SAS/STAT®
User's Guide

2026.04 - 2026.07*

* This document might apply to additional versions of the software. Open this document in SAS Help Center and click on the version in the banner to see all available versions.

SAS® Documentation
July 16, 2026

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2026. *SAS/STAT® User's Guide*. Cary, NC: SAS Institute Inc.

SAS/STAT® User's Guide

Copyright © 2026, SAS Institute Inc., Cary, NC, USA

All Rights Reserved. Produced in the United States of America.

For a hard-copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

The scanning, uploading, and distribution of this book via the internet or any other means without the permission of the publisher is illegal and punishable by law. Please purchase only authorized electronic editions and do not participate in or encourage electronic piracy of copyrighted materials. Your support of others' rights is appreciated.

July 2026

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.

SAS software may be provided with certain third-party software, including but not limited to open source software, which is licensed under its applicable third-party software license agreement. For license information about third-party software distributed with SAS software, refer to Third-Party Software Reference | SAS Support.

v_041-P1:statug

Contents

Chapter 1. Procedures by Category . . . 1
Chapter 2. Introduction . . . 13
Chapter 3. Introduction to Statistical Modeling with SAS/STAT Software . . . 23
Chapter 4. Introduction to Regression Procedures . . . 69
Chapter 5. Introduction to Analysis of Variance Procedures . . . 109
Chapter 6. Introduction to Mixed Modeling Procedures . . . 121
Chapter 7. Introduction to Causal Analysis Procedures . . . 135
Chapter 8. Introduction to Bayesian Analysis Procedures . . . 147
Chapter 9. Introduction to Categorical Data Analysis Procedures . . . 185
Chapter 10. Introduction to Multivariate Procedures . . . 197
Chapter 11. Introduction to Discriminant Procedures . . . 203
Chapter 12. Introduction to Clustering Procedures . . . 211
Chapter 13. Introduction to Scoring, Standardization, and Ranking Procedures . . . 251
Chapter 14. Introduction to Survival Analysis Procedures . . . 253
Chapter 15. Introduction to Survey Procedures . . . 261
Chapter 16. The Four Types of Estimable Functions . . . 275
Chapter 17. Introduction to Nonparametric Analysis . . . 293
Chapter 18. Introduction to Structural Equation Modeling with Latent Variables . . . 301
Chapter 19. Introduction to Power and Sample Size Analysis . . . 389
Chapter 20. Shared Concepts and Topics . . . 407
Chapter 21. Shared Concepts in High-Performance Computing . . . 567
Chapter 22. Shared Concepts and Topics in High-Performance Statistical Procedures . . . 571
Chapter 23. Using the Output Delivery System . . . 591
Chapter 24. Statistical Graphics Using ODS . . . 665
Chapter 25. ODS Graphics Template Modification . . . 843
Chapter 26. Customizing the Kaplan-Meier Survival Plot . . . 935
Chapter 27. The ACECLUS Procedure . . . 1027
Chapter 28. The ADAPTIVEREG Procedure . . . 1057
Chapter 29. The ANOVA Procedure . . . 1117
Chapter 30. The BCHOICE Procedure . . . 1175
Chapter 31. The BGLIMM Procedure . . . 1275
Chapter 32. The BOXPLOT Procedure . . . 1423
Chapter 33. The CALIS Procedure . . . 1503
Chapter 34. The CANCORR Procedure . . . 2167
Chapter 35. The CANDISC Procedure . . . 2197
Chapter 36. The CATMOD Procedure . . . 2225
Chapter 37. The CAUSALGRAPH Procedure . . . 2353
Chapter 38. The CAUSALMED Procedure . . . 2419
Chapter 39. The CAUSALTRT Procedure . . . 2503
Chapter 40. The CLUSTER Procedure . . . 2563
Chapter 41. The CORRESP Procedure . . . 2649
Chapter 42. The DISCRIM Procedure . . . 2721
Chapter 43. The DISTANCE Procedure . . . 2813
Chapter 44. The FACTOR Procedure . . . 2859
Chapter 45. The FASTCLUS Procedure . . . 2969

Chapter 46. The FMM Procedure . . . 3019
Chapter 47. The FREQ Procedure . . . 3145
Chapter 48. The GAM Procedure . . . 3375
Chapter 49. The GAMPL Procedure . . . 3431
Chapter 50. The GEE Procedure . . . 3505
Chapter 51. The GENMOD Procedure . . . 3567
Chapter 52. The GLIMMIX Procedure . . . 3775
Chapter 53. The GLM Procedure . . . 4143
Chapter 54. The GLMMOD Procedure . . . 4331
Chapter 55. The GLMPOWER Procedure . . . 4351
Chapter 56. The GLMSELECT Procedure . . . 4413
Chapter 57. The HPCANDISC Procedure . . . 4577
Chapter 58. The HPFMM Procedure . . . 4607
Chapter 59. The HPGENSELECT Procedure . . . 4727
Chapter 60. The HPLMIXED Procedure . . . 4799
Chapter 61. The HPLOGISTIC Procedure . . . 4843
Chapter 62. The HPMIXED Procedure . . . 4909
Chapter 63. The HPNLMOD Procedure . . . 4979
Chapter 64. The HPPLS Procedure . . . 5015
Chapter 65. The HPPRINCOMP Procedure . . . 5053
Chapter 66. The HPQUANTSELECT Procedure . . . 5081
Chapter 67. The HPREG Procedure . . . 5129
Chapter 68. The HPSPLIT Procedure . . . 5179
Chapter 69. The ICLIFETEST Procedure . . . 5261
Chapter 70. The ICPHREG Procedure . . . 5311
Chapter 71. The INBREED Procedure . . . 5381
Chapter 72. The IRT Procedure . . . 5407
Chapter 73. The KDE Procedure . . . 5501
Chapter 74. The KRIGE2D Procedure . . . 5547
Chapter 75. The LATTICE Procedure . . . 5623
Chapter 76. The LIFEREG Procedure . . . 5633
Chapter 77. The LIFETEST Procedure . . . 5751
Chapter 78. The LOESS Procedure . . . 5867
Chapter 79. The LOGISTIC Procedure . . . 5937
Chapter 80. The MCMC Procedure . . . 6211
Chapter 81. The MDS Procedure . . . 6511
Chapter 82. The MI Procedure . . . 6551
Chapter 83. The MIANALYZE Procedure . . . 6703
Chapter 84. The MIXED Procedure . . . 6761
Chapter 85. The MODECLUS Procedure . . . 6961
Chapter 86. The MULTTEST Procedure . . . 7041
Chapter 87. The NESTED Procedure . . . 7115
Chapter 88. The NLIN Procedure . . . 7129
Chapter 89. The NLMIXED Procedure . . . 7237
Chapter 90. The NPAR1WAY Procedure . . . 7355
Chapter 91. The ORTHOREG Procedure . . . 7433
Chapter 92. The PHREG Procedure . . . 7463
Chapter 93. The PLAN Procedure . . . 7741

Chapter 94. The PLM Procedure . . . 7777
Chapter 95. The PLS Procedure . . . 7837
Chapter 96. The POWER Procedure . . . 7895
Chapter 97. The Power and Sample Size Application . . . 8153
Chapter 98. The PRINCOMP Procedure . . . 8155
Chapter 99. The PRINQUAL Procedure . . . 8199
Chapter 100. The PROBIT Procedure . . . 8259
Chapter 101. The PSMATCH Procedure . . . 8365
Chapter 102. The QUANTLIFE Procedure . . . 8497
Chapter 103. The QUANTREG Procedure . . . 8529
Chapter 104. The QUANTSELECT Procedure . . . 8637
Chapter 105. The REG Procedure . . . 8721
Chapter 106. The RMSTREG Procedure . . . 8911
Chapter 107. The ROBUSTREG Procedure . . . 8955
Chapter 108. The RSREG Procedure . . . 9049
Chapter 109. The SCORE Procedure . . . 9089
Chapter 110. The SEQDESIGN Procedure . . . 9111
Chapter 111. The SEQTEST Procedure . . . 9325
Chapter 112. The SIM2D Procedure . . . 9493
Chapter 113. The SIMNORMAL Procedure . . . 9553
Chapter 114. The SPP Procedure . . . 9569
Chapter 115. The STDIZE Procedure . . . 9635
Chapter 116. The STDRATE Procedure . . . 9671
Chapter 117. The STEPDISC Procedure . . . 9743
Chapter 118. The SURVEYFREQ Procedure . . . 9767
Chapter 119. The SURVEYIMPUTE Procedure . . . 9893
Chapter 120. The SURVEYLOGISTIC Procedure . . . 9993
Chapter 121. The SURVEYMEANS Procedure . . . 10101
Chapter 122. The SURVEYPHREG Procedure . . . 10207
Chapter 123. The SURVEYREG Procedure . . . 10309
Chapter 124. The SURVEYSELECT Procedure . . . 10411
Chapter 125. The TPSPLINE Procedure . . . 10497
Chapter 126. The TRANSREG Procedure . . . 10553
Chapter 127. The TREE Procedure . . . 10777
Chapter 128. The TTEST Procedure . . . 10803
Chapter 129. The VARCLUS Procedure . . . 10901
Chapter 130. The VARCOMP Procedure . . . 10933
Chapter 131. The VARIOGRAM Procedure . . . 10959
Appendix A. Special SAS Data Sets . . . 11089
Appendix B. Sashelp Data Sets . . . 11105
Appendix C. Known Issues in Using CAS Tables with SAS/STAT Procedures . . . 11137

Acknowledgments

Many people make significant and continuing contributions to the development of SAS software products. The following are some of those who have contributed significant amounts of their time to help us make improvements to SAS/STAT software. (The organizations might not represent current affiliations.) These efforts include research and consulting, testing, and reviewing documentation. We are grateful to these members of the statistical community for their involvement, and to the many others not mentioned here for their feedback, suggestions, and consulting.

Alan Agresti, University of Florida (retired)
Greg M. Allenby, Ohio State University
Paul Allison, University of Pennsylvania
Xinming An, University of North Carolina, Chapel Hill
Peter Austin, Institute for Clinical Evaluative Sciences
John Barnard Jr., Cleveland Clinic Foundation
Douglas Bates, University of Wisconsin
David Binder, David Binder Research (deceased)

Suzette Blanchard, Frontier Science Technology Research Foundation
Donna J. Brogan, Emory University (retired)
Anu Brookins, Brookins Consultancy
Wilbert P. Byrd, Clemson University
Vincent Carey, Harvard University
Sally Carson, RAND Corporation
Love Casanova, CSC-FSG
Helene Cavior, Abacus Concepts
Rao Chaganty, Old Dominion University
George Chao, DuPont Merck Pharmaceutical Company
Colin Chen, Fannie Mae
Daniel M. Chilko, West Virginia University

Marc Cohen, Aktana
Robert Cohen, SAS Institute (retired)
Richard Cutler, Utah State University
Jan de Leeuw, University of California, Los Angeles (retired)
Dave DeLong, Duke University (retired)

Alex Dmitrienko, Mediana Inc.
Sandra Donaghy, North Carolina State University
David B. Duncan, Johns Hopkins University (deceased)
David B. Dunson, Duke University

Thomas Eagle, Eagle Analytics of California Inc.
Paul Eilers, Leiden University
Scott Emerson, University of Washington
Jane Evans, SAS Institute (retired)
Michael Farrell, Oak Ridge National Laboratory
Stewart Fossceco, SLF Consulting
Michael Friendly, York University
Rudolf J. Freund, Texas A&M University
Wayne A. Fuller, Iowa State University

Andrzej Galecki, University of Michigan
A. Ronald Gallant, Duke University

Joseph Gardiner, Michigan State University
Charles Gates, Texas A&M University
Thomas M. Gerig, North Carolina State University (retired)
Francis Giesbrecht, North Carolina State University (deceased)
Walter R. Gilks, University of Leeds
Harvey J. Gold, North Carolina State University (retired)
Kenneth Goldberg, Centocor Inc.
Robert J. Gray, Harvard University
Donald Guthrie, University of California, Los Angeles
Gerald Hajian, Schering-Plough Research Institute
Bob Hamer, University of North Carolina, Chapel Hill (deceased)
Frank E. Harrell Jr., Vanderbilt University
Wolfgang M. Hartmann, SAS Institute (retired)
Walter Harvey, Ohio State University
Douglas Hawkins, University of Minnesota
Xuming He, University of Illinois, Urbana-Champaign
Ronald W. Helms, Rho Inc.
Joseph Hilbe, Arizona State University
Gerry Hobbs, West Virginia University
Ronald R. Hocking, Texas A&M University
Nicholas J. Horton, Amherst College
Julian Horwich, CAMP Conference Company
Jason C. Hsu, Ohio State University (retired)
David Hurst, University of Alabama, Birmingham (deceased)
Joseph G. Ibrahim, University of North Carolina, Chapel Hill
Emilio A. Icaza, Louisiana State University
Jun Jie, Purdue University
Gordon Johnston, SAS Institute (retired)
Jörg Kaufmann, Bayer Schering Pharma AG
William Kennedy, Iowa State University
Bart Killam, SAS Institute (retired)
Jae Kwang Kim, Iowa State University
Gary Koch, University of North Carolina, Chapel Hill
Roger Koenker, University of Illinois, Urbana-Champaign
Alexander Kolovos, SpaceTimeWorks LLC
Kenneth L. Koonce, Louisiana State University
Warren F. Kuhfeld, SAS Institute (retired)
Rich La Valley, Strategic Technology Solutions
Russell V. Lenth, University of Iowa
Charles Lin, US Census Bureau
Danyu Lin, University of North Carolina, Chapel Hill
Ardell C. Linnerud, North Carolina State University (deceased)
Ramon C. Littell, University of Florida
Jun Liu, Infia ML
George MacKenzie, University of Oregon (deceased)
Alfio Marazzi, University of Lausanne
Brian Marx, Louisiana State University
J. Jack McArdle, University of Southern California
Roderick P. McDonald, Macquarie University
J. Philip Miller, Washington University Medical School
George Milliken, Kansas State University
Robert J. Monroe, North Carolina State University (deceased)
Mary Butler Moore, University of Florida
Jorge Morel, University of Maryland, Baltimore
David Morganstein, Westat
Robert D. Morrison, Oklahoma State University
Keith Muller, University of Florida
Meltem Narter
Nagaraj Neerchal, University of Maryland, Baltimore
Ralph G. O'Brien, Cleveland Clinic Foundation (retired)

Kenneth Offord, Mayo Clinic
Christopher R. Olinger, d-Wise Technologies
Christopher J. Paciorek, Harvard University
Robert Parks, Washington University
Richard M. Patterson, Auburn University (deceased)
Virginia Patterson, University of Tennessee
Limin Peng, Emory University
Cliff Pereira, Oregon State University
Hans-Peter Piepho, University of Hohenheim
Joseph Pingenot, Google
Jose Pinheiro, Johnson & Johnson
Edward Pollak, Iowa State University (deceased)
Stephen Portnoy, University of Illinois, Urbana-Champaign
John Preisser, University of North Carolina, Chapel Hill
C. H. Proctor, North Carolina State University (deceased)
Bahjat Qaqish, University of North Carolina, Chapel Hill
Dana Quade, University of North Carolina, Chapel Hill
Bill Raynor, Kimberly-Clark
Georgia Roberts, Statistics Canada (retired)
Robert N. Rodriguez, SAS Institute (retired)
James Roger, GlaxoSmithKline (retired)
Peter E. Rossi, University of California, Los Angeles
Peter Rousseeuw, University of Antwerp
Donald Rubin, Harvard University
Joseph L. Schafer, Pennsylvania State University
Robert Schechter, AstraZeneca
Shayle Searle, Cornell University (deceased)
Pat Hermes Smith, Ciba-Geigy
Roger Smith, USDA
Ying So, SAS Institute (retired)
Phil Spector, University of California, Berkeley
Michael Speed, Texas A&M University
William Stanish, Statistical Insight
Maura E. Stokes, SAS Institute (retired)
Rodney Strand, Orion Enterprises LLC
Walter Stroup, University of Nebraska
Elizabeth Stuart, Johns Hopkins University
Robert Teichman, ICI Americas Inc.
Terry M. Therneau, Mayo Clinic
Tyler J. VanderWeele, Harvard University
Edward Vonesh, Northwestern University
Grace Wahba, University of Wisconsin–Madison
Glenn Ware, University of Georgia
Peter H. Westfall, Texas Tech University
Edward W. Whitehorne, CI Partners LLC
Philip Whittall, Unilever (retired)
William Wigton, USDA
William Wilson, University of North Florida
Dong Xiang
Victor Yohai, University of Buenos Aires
Forrest W. Young, University of North Carolina, Chapel Hill (deceased)
Keming Yu, Brunel University London
Ke-Hai Yuan, University of Notre Dame
Yang C. Yuan, SAS Institute (retired)
Ruben Zamar, University of British Columbia
Scott Zeger, Johns Hopkins University

The final responsibility for the SAS System lies with SAS alone. We hope that you will always let us know your opinions about the SAS System and its documentation. It is through your participation that SAS software is continuously improved.

Chapter 1
Procedures by Category

Contents

Overview . . . 2
Analysis of Variance . . . 2
Bayesian Analysis . . . 2
Categorical Data Analysis . . . 3
Causal Analysis . . . 3
Cluster Analysis . . . 3
Descriptive Statistics . . . 4
Discriminant Analysis . . . 4
Distribution Analysis . . . 4
Exact Inference . . . 5
Finite Mixture Models . . . 5
Group Sequential Design and Analysis . . . 5
Longitudinal Data Analysis . . . 6
Market Research . . . 6
Missing Data Analysis . . . 6
Mixed Models . . . 7
Model Selection . . . 7
Multivariate Analysis . . . 7
Nonlinear Regression . . . 8
Nonparametric Analysis . . . 8
Nonparametric Regression . . . 8
Postprocessing . . . 9
Power and Sample Size . . . 9
Predictive Modeling . . . 9
Psychometric Analysis . . . 10
Quantile Regression . . . 10
Regression . . . 10
Robust Regression . . . 11
Spatial Analysis . . . 11
Standardization . . . 11
Structural Equation Models . . . 12
Survey Sampling and Analysis . . . 12
Survival Analysis . . . 12

Overview

The following sections list all SAS/STAT procedures by their use in different types of analysis. Procedures that can be used for multiple purposes are listed more than once.

Analysis of Variance

These procedures model classification effects in linear models—that is, how distinct groups in your data behave differently. For more information, see Chapter 5, "Introduction to Analysis of Variance Procedures," Chapter 16, "The Four Types of Estimable Functions," and Chapter 20, "Shared Concepts and Topics."

The ANOVA Procedure
The CATMOD Procedure
The GLM Procedure
The GLMMOD Procedure
The INBREED Procedure
The LATTICE Procedure
The NESTED Procedure
The PLAN Procedure
The TTEST Procedure

Bayesian Analysis

These procedures have features for analyzing data by using Bayesian methods. For more information, see Chapter 8, "Introduction to Bayesian Analysis Procedures."

The BCHOICE Procedure
The BGLIMM Procedure
The FMM Procedure
The GENMOD Procedure
The HPFMM Procedure
The LIFEREG Procedure
The MCMC Procedure
The PHREG Procedure

Categorical Data Analysis

These procedures analyze how data are grouped or categorized. For more information, see Chapter 9, "Introduction to Categorical Data Analysis Procedures."

The CATMOD Procedure
The FMM Procedure
The FREQ Procedure
The GENMOD Procedure
The HPFMM Procedure
The LOGISTIC Procedure
The PROBIT Procedure

Causal Analysis

These procedures are used to study causal effects in observational data.

The CAUSALGRAPH Procedure
The CAUSALMED Procedure
The CAUSALTRT Procedure
The PSMATCH Procedure

Cluster Analysis

These procedures detect groups in data such that observations within the same cluster are more similar to each other than to observations in different clusters. For more information, see Chapter 12, "Introduction to Clustering Procedures."

The ACECLUS Procedure
The CLUSTER Procedure
The DISTANCE Procedure
The FASTCLUS Procedure
The MODECLUS Procedure
The TREE Procedure
The VARCLUS Procedure

Descriptive Statistics

These procedures perform simple descriptive analyses.

The BOXPLOT Procedure
The FREQ Procedure
The STDIZE Procedure
The STDRATE Procedure

Discriminant Analysis

These procedures build models to discriminate data into known groups. For more information, see Chapter 11, "Introduction to Discriminant Procedures."

The CANDISC Procedure
The DISCRIM Procedure
The HPCANDISC Procedure
The STEPDISC Procedure

Distribution Analysis

This procedure performs kernel density estimation.

The KDE Procedure

Exact Inference

These procedures have features for performing exact categorical analysis, which is useful when asymptotic assumptions are not met.

The FREQ Procedure
The GENMOD Procedure
The LOGISTIC Procedure
The MULTTEST Procedure
The NPAR1WAY Procedure

Finite Mixture Models

These procedures fit models that are probabilistic mixtures of multiple simpler models.

The FMM Procedure
The HPFMM Procedure

Group Sequential Design and Analysis

These procedures perform design and analysis for group sequential adaptive clinical trials.

The SEQDESIGN Procedure
The SEQTEST Procedure

Longitudinal Data Analysis

These procedures model data that are observed longitudinally—that is, over time.

The GEE Procedure
The GENMOD Procedure
The GLIMMIX Procedure
The HPLMIXED Procedure
The MIXED Procedure

Market Research

These procedures build models for assessing the preferences and choices of consumers and potential consumers.

The BCHOICE Procedure
The CORRESP Procedure
The MDS Procedure
The PHREG Procedure
The PRINQUAL Procedure
The TRANSREG Procedure

Missing Data Analysis

These procedures have special features for handling unobserved values in data in building and analyzing models.

The CALIS Procedure
The GEE Procedure
The MCMC Procedure
The MI Procedure
The MIANALYZE Procedure
The SURVEYIMPUTE Procedure

Mixed Models

These procedures implement models and inferences for the variance and correlation in the data as well as for the expected value. For more information, see Chapter 6, "Introduction to Mixed Modeling Procedures."

The GLIMMIX Procedure
The HPMIXED Procedure
The MIXED Procedure
The NLMIXED Procedure
The PHREG Procedure
The VARCOMP Procedure

Model Selection

These procedures build models by using variable or effect selection techniques.

The GLMSELECT Procedure
The HPGENSELECT Procedure
The QUANTSELECT Procedure

Multivariate Analysis

These procedures characterize and model multidimensional data. For more information, see Chapter 10, "Introduction to Multivariate Procedures."

The CANCORR Procedure
The CORRESP Procedure
The FACTOR Procedure
The HPPRINCOMP Procedure
The MDS Procedure
The PRINCOMP Procedure
The PRINQUAL Procedure
The SIMNORMAL Procedure

Nonlinear Regression

These procedures fit nonlinear models to data.

The HPNLMOD Procedure
The NLIN Procedure
The TRANSREG Procedure

Nonparametric Analysis

These procedures perform analyses that relax typical parametric or distributional assumptions, including analyses based on ranks. For more information, see Chapter 17, "Introduction to Nonparametric Analysis."

The FREQ Procedure
The KDE Procedure
The NPAR1WAY Procedure

Nonparametric Regression

These procedures fit regression models that are less predefined and typically more flexible than more common parametric regression models.

The ADAPTIVEREG Procedure
The GAM Procedure
The GAMPL Procedure
The HPSPLIT Procedure
The LOESS Procedure
The TPSPLINE Procedure

Postprocessing

These procedures use the analytic results from other procedures to perform further analyses. For more information, see Chapter 13, "Introduction to Scoring, Standardization, and Ranking Procedures."

The PLM Procedure
The SCORE Procedure

Power and Sample Size

These procedures evaluate the statistical power of statistical tests for given data and the amount of data required to attain a specified level of power. For more information, see Chapter 19, "Introduction to Power and Sample Size Analysis."

The GLMPOWER Procedure
The POWER Procedure

Predictive Modeling

These procedures fit models for predicting new data.

The ADAPTIVEREG Procedure
The GLMSELECT Procedure
The HPGENSELECT Procedure
The HPLOGISTIC Procedure
The HPPLS Procedure
The HPREG Procedure
The PLS Procedure
The TRANSREG Procedure

Psychometric Analysis

These procedures fit models especially useful in analyzing psychological measurements.

The CALIS Procedure
The CORRESP Procedure
The FACTOR Procedure
The HPPRINCOMP Procedure
The IRT Procedure
The MDS Procedure
The PRINCOMP Procedure
The TRANSREG Procedure

Quantile Regression

These procedures fit models for specified quantiles of response rather than merely the expected value.

The HPQUANTSELECT Procedure
The QUANTLIFE Procedure
The QUANTREG Procedure
The QUANTSELECT Procedure

Regression

These procedures fit various forms of linear and nonlinear models. For more information, see Chapter 4, "Introduction to Regression Procedures."

The HPNLMOD Procedure
The HPPLS Procedure
The NLIN Procedure
The ORTHOREG Procedure
The PLM Procedure
The PLS Procedure
The REG Procedure
The RSREG Procedure
The TRANSREG Procedure

Robust Regression

These procedures implement regression methods that are robust with respect to outliers and other distributional violations in the data.

The HPQUANTSELECT Procedure
The QUANTREG Procedure
The QUANTSELECT Procedure
The ROBUSTREG Procedure

Spatial Analysis

These procedures perform analyses of data that have a two-dimensional geometric structure.

The KRIGE2D Procedure
The SIM2D Procedure
The SPP Procedure
The VARIOGRAM Procedure

Standardization

This procedure centers and scales data.

The STDIZE Procedure

Structural Equation Models

This procedure analyzes the structural relationships between observed and latent variables. For more information, see Chapter 18, "Introduction to Structural Equation Modeling with Latent Variables."

The CALIS Procedure

Survey Sampling and Analysis

These procedures create and analyze data for survey samples. For more information, see Chapter 15, "Introduction to Survey Procedures."

The SURVEYFREQ Procedure
The SURVEYIMPUTE Procedure
The SURVEYLOGISTIC Procedure
The SURVEYMEANS Procedure
The SURVEYPHREG Procedure
The SURVEYREG Procedure
The SURVEYSELECT Procedure

Survival Analysis

These procedures analyze time-to-event data. For more information, see Chapter 14, "Introduction to Survival Analysis Procedures." For information about modifying the Kaplan-Meier plot of the survivor function, see Chapter 26, "Customizing the Kaplan-Meier Survival Plot."

The ICLIFETEST Procedure
The ICPHREG Procedure
The LIFEREG Procedure
The LIFETEST Procedure
The PHREG Procedure
The QUANTLIFE Procedure
The RMSTREG Procedure
The SEQDESIGN Procedure
The SEQTEST Procedure
The SURVEYPHREG Procedure

Chapter 2
Introduction

Contents

Overview of SAS/STAT Software **13**
Experimental Software **14**
About This Book **14**
Chapter Organization 14
Syntax Conventions 15
Typographical Conventions 16
Options Used in Examples 17
Where to Turn for More Information **17**
Accessing the SAS/STAT Sample Library 17
Sashelp Data Sets 18
SAS Technical Support Services 18
Related SAS Software **19**
SAS/IML Software 19
Base SAS Software 19
SAS/ETS Software 20
SAS/GRAPH Software 20
SAS/OR Software 21
SAS/QC Software 21

Overview of SAS/STAT Software

SAS/STAT software provides comprehensive statistical tools for a wide range of statistical analyses, including analysis of variance, categorical data analysis, cluster analysis, multiple imputation, multivariate analysis, nonparametric analysis, power and sample size computations, psychometric analysis, regression, survey data analysis, and survival analysis. A few examples include nonlinear mixed models, generalized linear models, correspondence analysis, and robust regression. The software is constantly being updated to reflect new methodology. SAS/STAT software provides more than 90 procedures for statistical analysis.

SAS/STAT software also provides high-performance predictive modeling tools that have been specially developed to take advantage of parallel processing in multithreaded single-machine mode. Predictive modeling methods include regression, logistic regression, generalized linear models, generalized additive models, linear mixed models, nonlinear models, and decision trees. The procedures provide model selection, dimension reduction, and identification of important variables whenever this is appropriate for the analysis.

In addition to the high-performance statistical procedures described in this book, SAS/STAT includes high-performance utility procedures, which are described in *Base SAS Procedures Guide: High-Performance Procedures*.

Experimental Software

Experimental software is sometimes included as part of a production-release product. It is provided to (sometimes targeted) customers in order to obtain feedback. All experimental uses are marked Experimental in this document. Whenever an experimental procedure, statement, or option is used, a message is printed to the SAS log to indicate that it is experimental.

The design and syntax of experimental software might change before any production release. Experimental software has been tested prior to release, but it has not necessarily been tested to production-quality standards, and so should be used with care.

About This Book

Since SAS/STAT software is a part of the SAS System, this book assumes that you are familiar with Base SAS software and with the books *SAS Programmers Guide: Essentials* and the *Base SAS Procedures Guide*. It also assumes that you are familiar with basic SAS System concepts such as creating SAS data sets with the DATA step and manipulating SAS data sets with the procedures in Base SAS software (for example, the PRINT and SORT procedures).

Chapter Organization

This book is organized as follows.

Chapter 2, this chapter, provides an overview of SAS/STAT software and summarizes related information, products, and services. The remaining introductory chapters (Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," through Chapter 18, "Introduction to Structural Equation Modeling with Latent Variables") provide an introduction to the broad areas covered by SAS/STAT software.

Chapter 19, "Introduction to Power and Sample Size Analysis," provides documentation for the Power and Sample Size Application (PSS).

Chapter 20, "Shared Concepts and Topics," provides information about topics that are common to multiple procedures. Topics include parameterization of model effects, the EFFECT statement, and the NLOPTIONS statement. This chapter also documents the following statements that are used for postfitting analysis and are common across many modeling procedures: EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, SLICE, STORE, and TEST.

Chapter 21, "Shared Concepts in High-Performance Computing," describes common concepts that are used by SAS/STAT high-performance procedures.

Chapter 22, "Shared Concepts and Topics in High-Performance Statistical Procedures," describes common syntax elements that are supported by SAS/STAT high-performance procedures.

Chapter 23, "Using the Output Delivery System," explains the fundamentals of using the Output Delivery System (ODS) to manage your SAS output.

Chapter 24, "Statistical Graphics Using ODS," Chapter 25, "ODS Graphics Template Modification," and Chapter 26, "Customizing the Kaplan-Meier Survival Plot," describe the extension to ODS that enables many statistical procedures to create statistical graphics as easily as they create tables.

Subsequent chapters describe the SAS procedures that make up SAS/STAT software. These chapters appear in alphabetical order by procedure name and are organized as follows:

- The "Overview" section provides a brief description of the analysis provided by the procedure.
- The "Getting Started" section provides a quick introduction to the procedure through a simple example.
- The "Syntax" section describes the SAS statements and options that control the procedure.
- The "Details" section discusses methodology and miscellaneous details, such as ODS tables and ODS graphics.
- The "Examples" section contains examples that use the procedure.
- The "References" section contains references for the methodology and for examples of the procedure.

Following the chapters on the SAS/STAT procedures, Appendix A, "Special SAS Data Sets," documents the special SAS data sets that are associated with SAS/STAT procedures and Appendix B, "Sashelp Data Sets," documents Sashelp data sets that are used in this book.

Syntax Conventions

Each procedure's "Syntax" section follows the conventions that are described in this section. Consider the following MODEL and CONTRAST statements from the GLIMMIX procedure:

MODEL *response* < **(** *response-options* **)** > **=** < *fixed-effects* > < / *model-options* > **;**

CONTRAST '*label*' *contrast-specification* < , . . . > **=** < *fixed-effects* > < / *options* > **;**

These statements demonstrate the syntax conventions that are described in the following list:

UPPERCASE BOLD is used for keywords in lists of SAS statements and options in "Syntax" sections. When you type a keyword in SAS code, you type it as shown (although any mix of uppercase and lowercase is valid). In the two preceding examples, the statement names (MODEL and CONTRAST) are keywords. Note that keywords are displayed only in uppercase (not bold) when they are used in text.

oblique is used in syntax definitions and in text to represent arguments for which you supply a value. The preceding MODEL statement indicates that *response*, *response-options*, *fixed-effects*, and *model-options* are arguments for which you can supply values. The values that you can supply are defined later in the description of the MODEL statement.

< > (angle brackets) identify optional arguments. Arguments that are not enclosed in angle brackets are required. In the preceding MODEL statement, you must supply a value for *response* because it is not enclosed in angle brackets. However, supplying values for *response-options*, *fixed-effects*, and *model-options* is optional.

. . .	(ellipsis dots) indicate that the preceding argument can be repeated. Sometimes the argument is shown again for clarity. The ". . ." in the preceding CONTRAST statement indicate that you can supply additional *contrast-specifications*. The comma before the ellipsis dots indicates that each additional *contrast-specification* must be preceded by a comma.
'*value*'	(straight quotes around a *value*) indicate that the *value* must be enclosed in quotation marks (which can be single or double quotes). In the preceding CONTRAST statement, straight quotes around *label* indicate that you must use quotation marks when you specify a label.
( )	(parentheses) indicate arguments that must be grouped together. In the preceding MODEL statement, you must type parentheses around the *response-options*. Some statements do not require parentheses when you specify only one value for the argument that is shown in parentheses; these cases are indicated in the statement description.
|	(vertical bar) indicates that you can choose one value from a group of values. Values that are separated by a vertical bar are mutually exclusive. A vertical bar indicates mutually exclusive values for an option or indicates aliases for an option name.
;	(semicolon) indicates the end of a statement.

Other special characters—such as an equal sign (=), tilde (∼), colon (:), and slash (/)—indicate where in the syntax you must type those characters.

Typographical Conventions

This book uses several type styles for presenting information. The following list explains the meaning of the typographical conventions used in this book:

UPPERCASE ROMAN	is used for SAS statements, options, and other SAS language elements when they appear in the text. However, you can enter these elements in your own SAS programs in lowercase, uppercase, or a mixture of the two.
VariableName	is used for the names of variables and data sets when they appear in the text.
bold	is used to refer to matrices and vectors.
italic	is used for terms that are defined in the text, for emphasis, and for references to publications.
`monospace`	is used for example code. In most cases, this book uses lowercase type for SAS code.

Options Used in Examples

The HTMLBLUE style is used to create the graphs and the HTML tables that appear in the online documentation. The PEARLJ style is used to create the PDF tables that appear in the documentation. A style template controls stylistic elements such as colors, fonts, and presentation attributes. You can specify a style template in an ODS destination statement as follows:

```
ods html style=HTMLBlue;
. . .
ods html close;

ods pdf style=PearlJ;
. . .
ods pdf close;
```

For more information about styles, see Chapter 24, "Statistical Graphics Using ODS."

Most of the PDF tables are produced by using the following SAS System option:

```
options papersize=(6.5in 9in);
```

If you run the examples, you might get slightly different output. This is a function of the SAS System options that are used and the precision that your computer uses for floating-point calculations.

Where to Turn for More Information

This section describes other sources of information about SAS/STAT software.

Accessing the SAS/STAT Sample Library

The SAS/STAT sample library includes many examples that illustrate the use of SAS/STAT software. These examples include the examples in this documentation; at the beginning of each example in this documentation, you can find a link to the sample code that is used for it.

You can also access the SAS/STAT sample library at http://support.sas.com/documentation/onlinedoc/stat/ex_code/index.html.

Sashelp Data Sets

SAS provides over 200 data sets in the Sashelp library. These data sets are available for you to use for examples and for testing code. For example, the following step uses the Sashelp.Class data set:

```
proc reg data=sashelp.class;
   model weight = height;
quit;
```

You do not need to provide a DATA step to use Sashelp data sets.

The following steps list all of the data sets that are available in Sashelp:

```
ods select none;
proc contents data=sashelp._all_;
   ods output members=m;
run;
ods select all;

proc print;
   where memtype = 'DATA';
run;
```

The results of these steps (over 200 data set names) are not displayed.

The following steps provide detailed information about the Sashelp data sets:

```
proc contents data=sashelp._all_;
run;
```

The results of this step (hundreds of pages of PROC CONTENTS information) are not displayed. See Chapter B, "Sashelp Data Sets," for more information about Sashelp data sets.

SAS Technical Support Services

As with all SAS products, the SAS Technical Support staff is available to respond to problems and answer technical questions regarding the use of SAS/STAT software. Go to http://support.sas.com/techsup for more information.

Related SAS Software

Many features not found in SAS/STAT software are available in other parts of the SAS System. If you do not find something you need in SAS/STAT software, try looking for the feature in the following SAS software products.

SAS/IML Software

SAS/IML software gives you access to a powerful and flexible programming language (Interactive Matrix Language) in a dynamic, interactive environment. The fundamental object of the language is a data matrix. You can use SAS/IML software interactively (at the statement level) to see results immediately, or you can store statements in a module and execute them later. The programming is dynamic because necessary activities such as memory allocation and dimensioning of matrices are done automatically. SAS/IML software is of interest to users of SAS/STAT software because it enables you to program your methods in the SAS System.

Base SAS Software

The features provided by SAS/STAT software are in addition to the features provided by Base SAS software. Many data management and reporting capabilities you will need are part of Base SAS software. See *SAS Programmers Guide: Essentials* and *Base SAS Procedures Guide* for documentation of Base SAS software.

Base SAS software includes many useful SAS procedures. Base SAS procedures are documented in the *Base SAS Procedures Guide*. The following is a list of Base SAS procedures you might find useful:

CORR	computes correlations.
RANK	computes rankings or order statistics.
STANDARD	standardizes variables to a fixed mean and variance.
MEANS	computes descriptive statistics and summarizes or collapses data over cross sections.
TABULATE	prints descriptive statistics in tabular format.
UNIVARIATE	computes descriptive statistics.

ODS Graphics

Base SAS software provides the following:

- The SG family of procedures provides a simple syntax for creating stand-alone statistical graphics. These procedures include SGPLOT, SGSCATTER, and SGPANEL, which provide a simple and convenient syntax for producing many types of displays. They are particularly convenient for exploring and presenting data. See the *SAS ODS Graphics: Procedures Guide* for more information.

- The GTL (Graph Template Language) and the SGRENDER procedure provide a powerful syntax for creating customized graphs. See the *SAS Graph Template Language: User's Guide* and the *SAS Graph Template Language: Reference* for more information. You can also use the GTL to modify the SAS templates that are provided for use with SAS/STAT procedures. See Chapter 25, "ODS Graphics Template Modification," for more information about template modification.

See Chapter 24, "Statistical Graphics Using ODS," for more information about ODS Graphics.

SAS DATA Step

The DATA step is your primary tool for reading and processing data in the SAS System. The DATA step provides a powerful general purpose programming language that enables you to perform all kinds of data processing tasks. The DATA step is documented in *SAS Programmers Guide: Essentials*.

Base SAS Procedures

Base SAS software includes many useful SAS procedures. Base SAS procedures are documented in the *Base SAS Procedures Guide*. The following is a list of Base SAS procedures you might find useful:

CORR	computes correlations.
RANK	computes rankings or order statistics.
STANDARD	standardizes variables to a fixed mean and variance.
MEANS	computes descriptive statistics and summarizes or collapses data over cross sections.
TABULATE	prints descriptive statistics in tabular format.
UNIVARIATE	computes descriptive statistics.

SAS/ETS Software

SAS/ETS software provides SAS procedures for econometrics and time series analysis. It includes capabilities for forecasting, systems modeling and simulation, seasonal adjustment, and financial analysis and reporting. In addition, SAS/ETS software includes an interactive time series forecasting system.

SAS/GRAPH Software

SAS/GRAPH software includes procedures that create two- and three-dimensional plots and charts.

SAS/OR Software

SAS/OR software provides SAS procedures for operations research and project planning and includes a point-and-click interface to project management. Its capabilities include the following:

- solving transportation problems
- linear, integer, and mixed-integer programming
- nonlinear programming
- scheduling projects
- plotting Gantt charts
- drawing network diagrams
- solving optimal assignment problems
- network flow programming

SAS/OR software might be of interest to users of SAS/STAT software for its mathematical programming features. In particular, the NLP procedure in SAS/OR software solves nonlinear programming problems, and it can be used for constrained and unconstrained maximization of user-defined likelihood functions.

SAS/QC Software

SAS/QC software provides a variety of procedures for statistical quality control and quality improvement. SAS/QC software includes procedures for the following:

- Shewhart control charts
- cumulative sum control charts
- moving average control charts
- process capability analysis
- Pareto charts
- experimental design

Chapter 3
Introduction to Statistical Modeling with SAS/STAT Software

Contents

Overview: Statistical Modeling . . . **24**
- Statistical Models . . . 24
- Classes of Statistical Models . . . 27
  - Linear and Nonlinear Models . . . 27
  - Regression Models and Models with Classification Effects . . . 28
  - Univariate and Multivariate Models . . . 30
  - Fixed, Random, and Mixed Models . . . 31
  - Generalized Linear Models . . . 33
  - Latent Variable Models . . . 34
  - Bayesian Models . . . 36
- Classical Estimation Principles . . . 37
  - Least Squares . . . 38
  - Likelihood . . . 40
  - Inference Principles for Survey Data . . . 42

Statistical Background . . . **44**
- Hypothesis Testing and Power . . . 44
- Important Linear Algebra Concepts . . . 44
- Expectations of Random Variables and Vectors . . . 51
- Mean Squared Error . . . 54
- Linear Model Theory . . . 55
  - Finding the Least Squares Estimators . . . 56
  - Analysis of Variance . . . 57
  - Estimating the Error Variance . . . 59
  - Maximum Likelihood Estimation . . . 59
  - Estimable Functions . . . 60
  - Test of Hypotheses . . . 60
  - Residual Analysis . . . 63
  - Sweep Operator . . . 66

References . . . **67**

Overview: Statistical Modeling

The majority of procedures in SAS/STAT software are dedicated to solving problems in statistical modeling. The goal of this chapter is to provide a roadmap to statistical models and to modeling tasks, enabling you to make informed choices about the appropriate modeling context and tool. This chapter also introduces important terminology, notation, and concepts used throughout this documentation. Subsequent introductory chapters discuss model families and related procedures.

It is difficult to capture the complexity of statistical models in a simple scheme, so the classification used here is necessarily incomplete. It is most practical to classify models in terms of simple criteria, such as the presence of random effects, the presence of nonlinearity, characteristics of the data, and so on. That is the approach used here. After a brief introduction to statistical modeling in general terms, the chapter describes a number of model classifications and relates them to modeling tools in SAS/STAT software.

Statistical Models

Deterministic and Stochastic Models

Purely mathematical models, in which the relationships between inputs and outputs are captured entirely in deterministic fashion, can be important theoretical tools but are impractical for describing observational, experimental, or survey data. For such phenomena, researchers usually allow the model to draw on stochastic as well as deterministic elements. When the uncertainty of realizations leads to the inclusion of random components, the resulting models are called *stochastic* models. A *statistical* model, finally, is a stochastic model that contains *parameters*, which are unknown constants that need to be estimated based on assumptions about the model and the observed data.

There are many reasons why statistical models are preferred over deterministic models. For example:

- Randomness is often introduced into a system in order to achieve a certain balance or representativeness. For example, random assignment of treatments to experimental units allows unbiased inferences about treatment effects. As another example, selecting individuals for a survey sample by random mechanisms ensures a representative sample.
- Even if a deterministic model can be formulated for the phenomenon under study, a stochastic model can provide a more parsimonious and more easily comprehended description. For example, it is possible in principle to capture the result of a coin toss with a deterministic model, taking into account the properties of the coin, the method of tossing, conditions of the medium through which the coin travels and of the surface on which it lands, and so on. A very complex model is required to describe the simple outcome—heads or tails. Alternatively, you can describe the outcome quite simply as the result of a stochastic process, a Bernoulli variable that results in heads with a certain probability.
- It is often sufficient to describe the average behavior of a process, rather than each particular realization. For example, a regression model might be developed to relate plant growth to nutrient availability. The explicit aim of the model might be to describe how the average growth changes with nutrient availability, not to predict the growth of an individual plant. The support for the notion of averaging in a model lies in the nature of expected values, describing typical behavior in the presence of randomness. This, in turn, requires that the model contain stochastic components.

The defining characteristic of statistical models is their dependence on parameters and the incorporation of stochastic terms. The properties of the model and the properties of quantities derived from it must be studied in a long-run, average sense through expectations, variances, and covariances. The fact that the parameters of the model must be estimated from the data introduces a stochastic element in applying a statistical model: because the model is not deterministic but includes randomness, parameters and related quantities derived from the model are likewise random. The properties of parameter estimators can often be described only in an asymptotic sense, imagining that some aspect of the data increases without bound (for example, the number of observations or the number of groups).

The process of estimating the parameters in a statistical model based on your data is called *fitting* the model. For many classes of statistical models there are a number of procedures in SAS/STAT software that can perform the fitting. In many cases, different procedures solve identical estimation problems—that is, their parameter estimates are identical. In some cases, the same model parameters are estimated by different statistical principles, such as least squares versus maximum likelihood estimation. Parameter estimates obtained by different methods typically have different statistical properties—distribution, variance, bias, and so on. The choice between competing estimation principles is often made on the basis of properties of the estimators. Distinguishing properties might include (but are not necessarily limited to) computational ease, interpretive ease, bias, variance, mean squared error, and consistency.

Model-Based and Design-Based Randomness

A statistical model is a description of the data-generating mechanism, not a description of the specific data to which it is applied. The aim of a model is to capture those aspects of a phenomenon that are relevant to inquiry and to explain how the data could have come about as a realization of a random experiment. These relevant aspects might include the genesis of the randomness and the stochastic effects in the phenomenon under study. Different schools of thought can lead to different model formulations, different analytic strategies, and different results. Coarsely, you can distinguish between a viewpoint of *innate* randomness and one of *induced* randomness. This distinction leads to model-based and design-based inference approaches.

In a design-based inference framework, the random variation in the observed data is induced by random *selection* or random *assignment*. Consider the case of a survey sample from a finite population of size N; suppose that $\mathcal{F}_N = \{y_i : i \in U_N\}$ denotes the finite set of possible values and U_N is the index set $U_N = \{1, 2, \ldots, N\}$. Then a sample S, a subset of U_N, is selected by probability rules. The realization of the random experiment is the selection of a particular set S; the associated values selected from $\mathcal{F}_N$ are considered fixed. If properties of a design-based sampling estimator are evaluated, such as bias, variance, and mean squared error, they are evaluated with respect to the distribution induced by the sampling mechanism.

Design-based approaches also play an important role in the analysis of data from controlled experiments by randomization tests. Suppose that k treatments are to be assigned to kr homogeneous experimental units. If you form k sets of r units with equal probability, and you assign the jth treatment to the tth set, a completely randomized experimental design (CRD) results. A design-based view treats the potential response of a particular treatment for a particular experimental unit as a constant. The stochastic nature of the error-control design is induced by randomly selecting one of the potential responses.

Statistical models are often used in the design-based framework. In a survey sample the model is used to motivate the choice of the finite population parameters and their sample-based estimators. In an experimental design, an assumption of additivity of the contributions from treatments, experimental units, observational errors, and experimental errors leads to a linear statistical model. The approach to statistical inference where statistical models are used to construct estimators and their properties are evaluated with respect to

the distribution induced by the sample selection mechanism is known as *model-assisted inference* (Särndal, Swensson, and Wretman 1992).

In a purely model-based framework, the only source of random variation for inference comes from the unknown variation in the responses. Finite population values are thought of as a realization of a superpopulation model that describes random variables $Y_1, Y_2, \ldots$. The observed values $y_1, y_2, \ldots$ are realizations of these random variables. A model-based framework does not imply that there is only one source of random variation in the data. For example, mixed models might contain random terms that represent selection of effects from hierarchical (super-) populations at different granularity. The analysis takes into account the hierarchical structure of the random variation, but it continues to be model based.

A design-based approach is implicit in SAS/STAT procedures whose name commences with SURVEY, such as the SURVEYFREQ, SURVEYMEANS, SURVEYREG, SURVEYLOGISTIC, and SURVEYPHREG procedures. Inferential approaches are model based in other SAS/STAT procedures. For more information about analyzing survey data with SAS/STAT software, see Chapter 15, "Introduction to Survey Procedures."

Model Specification

If the model is accepted as a description of the data-generating mechanism, then its parameters are estimated using the data at hand. Once the parameter estimates are available, you can apply the model to answer questions of interest about the study population. In other words, the model becomes the lens through which you view the problem itself, in order to ask and answer questions of interest. For example, you might use the estimated model to derive new predictions or forecasts, to test hypotheses, to derive confidence intervals, and so on.

Obviously, the model must be "correct" to the extent that it sufficiently describes the data-generating mechanism. Model selection, diagnosis, and discrimination are important steps in the model-building process. This is typically an iterative process, starting with an initial model and refining it. The first important step is thus to formulate your knowledge about the data-generating process and to express the real observed phenomenon in terms of a statistical model. A statistical model describes the distributional properties of one or more variables, the *response* variables. The extent of the required distributional specification depends on the model, estimation technique, and inferential goals. This description often takes the simple form of a model with additive error structure:

$$\text{response} = \text{mean} + \text{error}$$

In mathematical notation this simple model equation becomes

$$Y = f(x_1, \ldots, x_k; \beta_1, \ldots, \beta_p) + \epsilon$$

In this equation Y is the *response* variable, often also called the *dependent* variable or the *outcome* variable. The terms $x_1, \ldots, x_k$ denote the values of k regressor variables, often termed the *covariates* or the "independent" variables. The terms $\beta_1, \ldots, \beta_p$ denote parameters of the model, unknown constants that are to be estimated. The term ϵ denotes the random disturbance of the model; it is also called the residual term or the error term of the model.

In this simple model formulation, stochastic properties are usually associated only with the ϵ term. The covariates $x_1, \ldots, x_k$ are usually known values, not subject to random variation. Even if the covariates are measured with error, so that their values are in principle random, they are considered fixed in most models fit by SAS/STAT software. In other words, stochastic properties under the model are derived conditional on the

xs. If ϵ is the only stochastic term in the model, and if the errors have a mean of zero, then the function $f(\cdot)$ is the *mean function* of the statistical model. More formally,

$$\mathrm{E}[Y] = f(x_1, \ldots, x_k; \beta_1, \ldots, \beta_p)$$

where $\mathrm{E}[\cdot]$ denotes the expectation operator.

In many applications, a simple model formulation is inadequate. It might be necessary to specify not only the stochastic properties of a single error term, but also how model errors associated with different observations relate to each other. A simple additive error model is typically inappropriate to describe the data-generating mechanism if the errors do not have zero mean or if the variance of observations depends on their means. For example, if Y is a Bernoulli random variable that takes on the values 0 and 1 only, a regression model with additive error is not meaningful. Models for such data require more elaborate formulations involving probability distributions.

Classes of Statistical Models

Linear and Nonlinear Models

A statistical estimation problem is nonlinear if the estimating equations—the equations whose solution yields the parameter estimates—depend on the parameters in a nonlinear fashion. Such estimation problems typically have no closed-form solution and must be solved by iterative, numerical techniques.

Nonlinearity in the mean function is often used to distinguish between linear and nonlinear models. A model has a nonlinear mean function if the derivative of the mean function with respect to the parameters depends on at least one other parameter. Consider, for example, the following models that relate a response variable Y to a single regressor variable x:

$$\mathrm{E}[Y|x] = \beta_0 + \beta_1 x$$
$$\mathrm{E}[Y|x] = \beta_0 + \beta_1 x + \beta_2 x^2$$
$$\mathrm{E}[Y|x] = \beta + x/\alpha$$

In these expressions, $\mathrm{E}[Y|x]$ denotes the expected value of the response variable Y at the fixed value of x. (The conditioning on x simply indicates that the predictor variables are assumed to be non-random. Conditioning is often omitted for brevity in this and subsequent chapters.)

The first model in the previous list is a simple linear regression (SLR) model. It is linear in the parameters β_0 and β_1 since the model derivatives do not depend on unknowns:

$$\frac{\partial}{\beta_0}(\beta_0 + \beta_1 x) = 1$$
$$\frac{\partial}{\beta_1}(\beta_0 + \beta_1 x) = x$$

The model is also linear in its relationship with x (a straight line). The second model is also linear in the parameters, since

$$\frac{\partial}{\beta_0}\left(\beta_0 + \beta_1 x + \beta_2 x^2\right) = 1$$
$$\frac{\partial}{\beta_1}\left(\beta_0 + \beta_1 x + \beta_2 x^2\right) = x$$
$$\frac{\partial}{\beta_2}\left(\beta_0 + \beta_1 x + \beta_2 x^2\right) = x^2$$

However, this second model is *curvilinear*, since it exhibits a curved relationship when plotted against x. The third model, finally, is a nonlinear model since

$$\frac{\partial}{\beta}\left(\beta + x/\alpha\right) = 1$$
$$\frac{\partial}{\alpha}\left(\beta + x/\alpha\right) = -\frac{x}{\alpha^2}$$

The second of these derivatives depends on a parameter α. A model is nonlinear if it is not linear in at least one parameter. Only the third model is a nonlinear model. A graph of $\mathrm{E}[Y]$ versus the regressor variable thus does not indicate whether a model is nonlinear. A curvilinear relationship in this graph can be achieved by a model that is linear in the parameters.

Nonlinear mean functions lead to nonlinear estimation. It is important to note, however, that nonlinear estimation arises also because of the estimation principle or because the model structure contains nonlinearity in other parts, such as the covariance structure. For example, fitting a simple linear regression model by minimizing the sum of the absolute residuals leads to a nonlinear estimation problem despite the fact that the mean function is linear.

Regression Models and Models with Classification Effects

A linear regression model in the broad sense has the form

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\mathbf{Y}$ is the vector of response values, $\mathbf{X}$ is the matrix of regressor effects, $\boldsymbol{\beta}$ is the vector of regression parameters, and $\boldsymbol{\epsilon}$ is the vector of errors or residuals. A regression model in the narrow sense—as compared to a classification model—is a linear model in which all regressor effects are continuous variables. In other words, each effect in the model contributes a single column to the $\mathbf{X}$ matrix and a single parameter to the overall model. For example, a regression of subjects' weight (Y) on the regressors age (x_1) and body mass index (bmi, x_2) is a regression model in this narrow sense. In symbolic notation you can write this regression model as

weight = age + bmi + error

This symbolic notation expands into the statistical model

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \epsilon_i$$

Single parameters are used to model the effects of age (β_1) and bmi (β_2), respectively.

A classification effect, on the other hand, is associated with possibly more than one column of the **X** matrix. Classification with respect to a variable is the process by which each observation is associated with one of k levels; the process of determining these k levels is referred to as *levelization* of the variable. Classification variables are used in models to identify experimental conditions, group membership, treatments, and so on. The actual values of the classification variable are not important, and the variable can be a numeric or a character variable. What is important is the association of discrete values or levels of the classification variable with groups of observations. For example, in the previous illustration, if the regression also takes into account the subjects' gender, this can be incorporated in the model with a two-level classification variable. Suppose that the values of the gender variable are coded as 'F' and 'M', respectively. In symbolic notation the model

weight = age + bmi + gender + error

expands into the statistical model

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \tau_1 \mathrm{I}(\mathrm{gender} = {}'\mathtt{F}') + \tau_2 \mathrm{I}(\mathrm{gender} = {}'\mathtt{M}') + \epsilon_i$$

where I(gender='F') is the indicator function that returns 1 if the value of the gender variable is 'F' and 0 otherwise. Parameters τ_1 and τ_2 are associated with the gender classification effect. This form of parameterizing the gender effect in the model is only one of several different methods of incorporating the levels of a classification variable in the model. This form, the so-called singular parameterization, is the most general approach, and it is used in the GLM, MIXED, and GLIMMIX procedures. Alternatively, classification effects with various forms of nonsingular parameterizations are available in such procedures as GENMOD and LOGISTIC. See the documentation for the individual SAS/STAT procedures on their respective facilities for parameterizing classification variables and the section "Parameterization of Model Effects" on page 412 in Chapter 20, "Shared Concepts and Topics," for general details.

Models that contain only classification effects are often identified with *analysis of variance* (ANOVA) models, because ANOVA methods are frequently used in their analysis. This is particularly true for experimental data where the model effects comprise effects of the treatment and error-control design. However, classification effects appear more widely than in models to which analysis of variance methods are applied. For example, many mixed models, where parameters are estimated by restricted maximum likelihood, consist entirely of classification effects but do not permit the sum of squares decomposition typical for ANOVA techniques.

Many models contain both continuous and classification effects. For example, a continuous-by-class effect consists of at least one continuous variable and at least one classification variable. Such effects are convenient, for example, to vary slopes in a regression model by the levels of a classification variable. Also, recent enhancements to linear modeling syntax in some SAS/STAT procedures (including GLIMMIX and GLMSELECT) enable you to construct sets of columns in **X** matrices from a single continuous variable. An example is modeling with splines where the values of a continuous variable x are expanded into a spline basis that occupies multiple columns in the **X** matrix. For purposes of the analysis you can treat these columns as a single unit or as individual, unrelated columns. For more details, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

Univariate and Multivariate Models

A multivariate statistical model is a model in which multiple response variables are modeled jointly. Suppose, for example, that your data consist of heights (h_i) and weights (w_i) of children, collected over several years (t_i). The following separate regressions represent two univariate models:

$$
\begin{aligned}
w_i &= \beta_{w0} + \beta_{w1} t_i + \epsilon_{wi} \\
h_i &= \beta_{h0} + \beta_{h1} t_i + \epsilon_{hi}
\end{aligned}
$$

In the univariate setting, no information about the children's heights "flows" to the model about their weights and vice versa. In a multivariate setting, the heights and weights would be modeled jointly. For example:

$$
\begin{aligned}
\mathbf{Y}_i = \begin{bmatrix} w_i \\ h_i \end{bmatrix} &= \mathbf{X}\boldsymbol{\beta} + \begin{bmatrix} \epsilon_{wi} \\ \epsilon_{hi} \end{bmatrix} \\
&= \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}_i \\
\boldsymbol{\epsilon}_i &\sim \left(\mathbf{0}, \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{bmatrix} \right)
\end{aligned}
$$

The vectors $\mathbf{Y}_i$ and $\boldsymbol{\epsilon}_i$ collect the responses and errors for the two observation that belong to the same subject. The errors from the same child now have the correlation

$$
\mathrm{Corr}[\epsilon_{wi}, \epsilon_{hi}] = \frac{\sigma_{12}}{\sqrt{\sigma_1^2 \sigma_2^2}}
$$

and it is through this correlation that information about heights "flows" to the weights and vice versa. This simple example shows only one approach to modeling multivariate data, through the use of covariance structures. Other techniques involve seemingly unrelated regressions, systems of linear equations, and so on.

Multivariate data can be coarsely classified into three types. The response vectors of *homogeneous multivariate data* consist of observations of the same attribute. Such data are common in repeated measures experiments and longitudinal studies, where the same attribute is measured repeatedly over time. Homogeneous multivariate data also arise in spatial statistics where a set of geostatistical data is the incomplete observation of a single realization of a random experiment that generates a two-dimensional surface. One hundred measurements of soil electrical conductivity collected in a forest stand compose a single observation of a 100-dimensional homogeneous multivariate vector. *Heterogeneous multivariate* observations arise when the responses that are modeled jointly refer to different attributes, such as in the previous example of children's weights and heights. There are two important subtypes of heterogeneous multivariate data. In *homocatanomic multivariate data* the observations come from the same distributional family. For example, the weights and heights might both be assumed to be normally distributed. With *heterocatanomic multivariate data* the observations can come from different distributional families. The following are examples of heterocatanomic multivariate data:

- For each patient you observe blood pressure (a continuous outcome), the number of prior episodes of an illness (a count variable), and whether the patient has a history of diabetes in the family (a binary outcome). A multivariate model that models the three attributes jointly might assume a lognormal distribution for the blood pressure measurements, a Poisson distribution for the count variable and a Bernoulli distribution for the family history.

- In a study of HIV/AIDS survival, you model jointly a patient's CD4 cell count over time—itself a homogeneous multivariate outcome—and the survival of the patient (event-time data).

Fixed, Random, and Mixed Models

Each term in a statistical model represents either a *fixed effect* or a *random effect*. Models in which all effects are fixed are called fixed-effects models. Similarly, models in which all effects are random—apart from possibly an overall intercept term—are called random-effects models. Mixed models, then, are those models that have fixed-effects and random-effects terms. In matrix notation, the linear fixed, linear random, and linear mixed model are represented by the following model equations, respectively:

$$\begin{aligned}
\mathbf{Y} &= \mathbf{X}\boldsymbol{\beta} \qquad\quad + \boldsymbol{\epsilon} \\
\mathbf{Y} &= \qquad\quad \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon} \\
\mathbf{Y} &= \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}
\end{aligned}$$

In these expressions, $\mathbf{X}$ and $\mathbf{Z}$ are design or regressor matrices associated with the fixed and random effects, respectively. The vector $\boldsymbol{\beta}$ is a vector of fixed-effects parameters, and the vector $\boldsymbol{\gamma}$ represents the random effects. The mixed modeling procedures in SAS/STAT software assume that the random effects $\boldsymbol{\gamma}$ follow a normal distribution with variance-covariance matrix $\mathbf{G}$ and, in most cases, that the random effects have mean zero.

Random effects are often associated with classification effects, but this is not necessary. As an example of random regression effects, you might want to model the slopes in a growth model as consisting of two components: an overall (fixed-effects) slope that represents the slope of the average individual, and individual-specific random deviations from the overall slope. The $\mathbf{X}$ and $\mathbf{Z}$ matrix would then have column entries for the regressor variable associated with the slope. You are modeling fixed and randomly varying regression coefficients.

Having random effects in your model has a number of important consequences:

- Some observations are no longer uncorrelated but instead have a covariance that depends on the variance of the random effects.
- You can and should distinguish between the inference spaces; inferences can be drawn in a broad, intermediate, and narrow inference space. In the narrow inference space, conclusions are drawn about the particular values of the random effects selected in the study. The broad inference space applies if inferences are drawn with respect to all possible levels of the random effects. The intermediate inference space can be applied for effects consisting of more than one random term, when inferences are broad with respect to some factors and narrow with respect to others. In fixed-effects models, there is no corresponding concept to the broad and intermediate inference spaces.
- Depending on the structure of $\mathbf{G}$ and $\mathrm{Var}[\boldsymbol{\epsilon}]$ and also subject to the balance in your data, there might be no closed-form solution for the parameter estimates. Although the model is linear in $\boldsymbol{\beta}$, iterative estimation methods might be required to estimate all parameters of the model.
- Certain concepts, such as least squares means and Type III estimable functions, are meaningful only for fixed effects.
- By using random effects, you are modeling variation through variance. Variation in data simply implies that things are not equal. Variance, on the other hand, describes a feature of a random variable. Random effects in your model are random variables: they model variation through variance.

It is important to properly determine the nature of the model effects as fixed or random. An effect is either fixed or random by its very nature; it is improper to consider it fixed in one analysis and random in another

depending on what type of results you want to produce. If, for example, a treatment effect is random and you are interested in comparing treatment means, and only the levels selected in the study are of interest, then it is not appropriate to model the treatment effect as fixed so that you can draw on least squares mean analysis. The appropriate strategy is to model the treatment effect as random and to compare the solutions for the treatment effects in the narrow inference space.

In determining whether an effect is fixed or random, it is helpful to inquire about the *genesis* of the effect. If the levels of an effect are randomly sampled, then the effect is a random effect. The following are examples:

- In a large clinical trial, drugs A, B, and C are applied to patients in various clinical centers. If the clinical centers are selected at random from a population of possible clinics, their effect on the response is modeled with a random effect.
- In repeated measures experiments with people or animals as subjects, subjects are declared to be random because they are selected from the larger population to which you want to generalize.
- Fertilizers could be applied at a number of levels. Three levels are randomly selected for an experiment to represent the population of possible levels. The fertilizer effects are random effects.

Quite often it is not possible to select effects at random, or it is not known how the values in the data became part of the study. For example, suppose you are presented with a data set consisting of student scores in three school districts, with four to ten schools in each district and two to three classrooms in each school. How do you decide which effects are fixed and which are random? As another example, in an agricultural experiment conducted in successive years at two locations, how do you decide whether location and year effects are fixed or random? In these situations, the fixed or random nature of the effect might be debatable, bearing out the adage that "one modeler's fixed effect is another modeler's random effect." However, this fact does not constitute license to treat as random those effects that are clearly fixed, or vice versa.

When an effect cannot be randomized or it is not known whether its levels have been randomly selected, it can be a random effect if its impact on the outcome variable is of a stochastic nature—that is, if it is the realization of a random process. Again, this line of thinking relates to the genesis of the effect. A random year, location, or school district effect is a placeholder for different environments that cannot be selected at random but whose effects are the cumulative result of many individual random processes. Note that this argument does not imply that effects are random because the experimenter does not know much about them. The key notion is that effects represent something, whether or not that something is known to the modeler. Broadening the inference space beyond the observed levels is thus possible, although you might not be able to articulate what the realizations of the random effects represent.

A consequence of having random effects in your model is that some observations are no longer uncorrelated but instead have a covariance that depends on the variance of the random effect. In fact, in some modeling applications random effects might be used not only to model heterogeneity in the parameters of a model, but also to induce correlations among observations. The typical assumption about random effects in SAS/STAT software is that the effects are normally distributed.

For more information about mixed modeling tools in SAS/STAT software, see Chapter 6, "Introduction to Mixed Modeling Procedures."

Generalized Linear Models

A class of models that has gained increasing importance in the past several decades is the class of generalized linear models. The theory of generalized linear models originated with Nelder and Wedderburn (1972); Wedderburn (1974), and was subsequently made popular in the monograph by McCullagh and Nelder (1989). This class of models extends the theory and methods of linear models to data with nonnormal responses. Before this theory was developed, modeling of nonnormal data typically relied on transformations of the data, and the transformations were chosen to improve symmetry, homogeneity of variance, or normality. Such transformations have to be performed with care because they also have implications for the error structure of the model, Also, back-transforming estimates or predicted values can introduce bias.

Generalized linear models also apply a transformation, known as the *link function*, but it is applied to a deterministic component, the mean of the data. Furthermore, generalized linear models take the distribution of the data into account, rather than assuming that a transformation of the data leads to normally distributed data to which standard linear modeling techniques can be applied.

To put this generalization in place requires a slightly more sophisticated model setup than that required for linear models for normal data:

- The *systematic* component is a linear predictor similar to that in linear models, $\eta = \mathbf{x}'\boldsymbol{\beta}$. The linear predictor is a linear function in the parameters. In contrast to the linear model, η does not represent the mean function of the data.

- The *link function* $g(\cdot)$ relates the linear predictor to the mean, $g(\mu) = \eta$. The link function is a monotonic, invertible function. The mean can thus be expressed as the inversely linked linear predictor, $\mu = g^{-1}(\eta)$. For example, a common link function for binary and binomial data is the logit link, $g(t) = \log\{t/(1-t)\}$. The mean function of a generalized linear model with logit link and a single regressor can thus be written as

$$\log\left\{\frac{\mu}{1-\mu}\right\} = \beta_0 + \beta_1 x$$
$$\mu = \frac{1}{1+\exp\{-\beta_0 - \beta_1 x\}}$$

  This is known as a logistic regression model.

- The *random component* of a generalized linear model is the distribution of the data, assumed to be a member of the exponential family of distributions. Discrete members of this family include the Bernoulli (binary), binomial, Poisson, geometric, and negative binomial (for a given value of the scale parameter) distribution. Continuous members include the normal (Gaussian), beta, gamma, inverse Gaussian, and exponential distribution.

The standard linear model with normally distributed error is a special case of a generalized linear model; the link function is the identity function and the distribution is normal.

Latent Variable Models

Latent variable modeling involves variables that are not observed directly in your research. It has a relatively long history, dating back from the measure of general intelligence by common factor analysis (Spearman 1904) to the emergence of modern-day structural equation modeling (Jöreskog 1973; Keesling 1972; Wiley 1973).

Latent variables are involved in almost all kinds of regression models. In a broad sense, all additive error terms in regression models are latent variables simply because they are not measured in research. Hereafter, however, a narrower sense of latent variables is used when referring to latent variable models. Latent variables are *systematic* unmeasured variables that are also referred to as *factors*. For example, in the following diagram a simple relation between Emotional Intelligence and Career Achievement is shown:

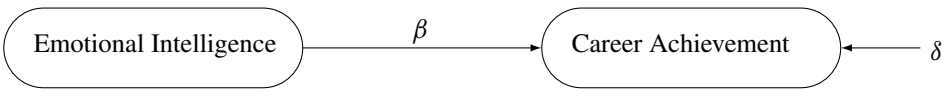

In the diagram, both Emotional Intelligence and Career Achievement are treated as latent factors. They are hypothetical constructs in your model. You hypothesize that Emotional Intelligence is a "causal factor" or predictor of Career Achievement. The symbol β represents the regression coefficient or the effect of Emotional Intelligence on Career Achievement. However, the "causal relationship" or prediction is not perfect. There is an error term δ, which accounts for the unsystematic part of the prediction. You can represent the preceding diagram by using the following linear equation:

$$\mathrm{CA} = \beta \mathrm{EI} + \delta$$

where CA represents Career Achievement and EI represents Emotional Intelligence. The means of the latent factors in the linear model are arbitrary, and so they are assumed to be zero. The error variable δ also has a zero mean with an unknown variance. This equation represents the so-called "structural model," where the "true" relationships among latent factors are theorized.

In order to model this theoretical model with latent factors, some observed variables must somehow relate to these factors. This calls for the measurement models for latent factors. For example, Emotional Intelligence could be measured by some established tests. In these tests, individuals are asked to respond to certain special situations that involve stressful decision making, personal confrontations, and so on. Their responses to these situations are then rated by experts or a standardized scoring system. Suppose there are three such tests and the test scores are labeled as X1, X2 and X3, respectively. The measurement model for the latent factor Emotional Intelligence is specified as follows:

$$\begin{aligned} \mathrm{X1} &= a_1 \mathrm{EI} + e_1 \\ \mathrm{X2} &= a_2 \mathrm{EI} + e_2 \\ \mathrm{X3} &= a_3 \mathrm{EI} + e_3 \end{aligned}$$

where a_1, a_2, and a_3 are regression coefficients and e_1, e_2, and e_3 are measurement errors. Measurement errors are assumed to be independent of the latent factors EI and CA. In the measurement model, X1, X2, and X3 are called the indicators of the latent variable EI. These observed variables are assumed to be centered in the model, and therefore no intercept terms are needed. Each of the indicators is a scaled measurement of the latent factor EI plus a unique error term.

Similarly, you need to have a measurement model for the latent factor CA. Suppose that there are four observed indicators Y1, Y2, Y3, and Y4 (for example, Job Status) for this latent factor. The measurement

model for CA is specified as follows:

$$\begin{aligned} \text{Y1} &= a_4\text{CA} + e_4 \\ \text{Y2} &= a_5\text{CA} + e_5 \\ \text{Y3} &= a_6\text{CA} + e_6 \\ \text{Y4} &= a_7\text{CA} + e_7 \end{aligned}$$

where a_4, a_5, a_6, and a_7 are regression coefficients and e_4, e_5, e_6, and e_7 are error terms. Again, the error terms are assumed to be independent of the latent variables EI and CA, and Y1, Y2, Y3, and Y4 are centered in the equations.

Given the data for the measured variables, you analyze the structural and measurement models simultaneously by the structural equation modeling techniques. In other words, estimation of β, a_1–a_7, and other parameters in the model are carried out simultaneously in the modeling.

Modeling involving the use of latent factors is quite common in social and behavioral sciences, personality assessment, and marketing research. Hypothetical constructs, although not observable, are very important in building theories in these areas.

Another use of latent factors in modeling is to "purify" the predictors in regression analysis. A common assumption in linear regression models is that predictors are measured without errors. That is, in the following linear equation x is assumed to have been measured without errors:

$$y = \alpha + \beta x + \epsilon$$

However, if x has been contaminated with measurement errors that cannot be ignored, the estimate of β might be biased severely so that the true relationship between x and y would be masked.

A measurement model for x provides a solution to such a problem. Let F_x be a "purified" version of x. That is, F_x is the "true" measure of x without measurement errors, as described in the following equation:

$$x = F_x + \delta$$

where δ represents a random measurement error term. Now, the linear relationship of interest is specified in the following new linear regression equation:

$$y = \alpha + \beta F_x + \epsilon$$

In this equation, F_x, which is now free from measurement errors, replaces x in the original equation. With measurement errors taken into account in the simultaneous fitting of the measurement and the new regression equations, estimation of β is unbiased; hence it reflects the true relationship much better.

Certainly, introducing latent factors in models is not a "free lunch." You must pay attention to the identification issues induced by the latent variable methodology. That is, in order to estimate the parameters in structural equation models with latent variables, you must set some identification constraints in these models. There are some established rules or conventions that would lead to proper model identification and estimation. See Chapter 18, "Introduction to Structural Equation Modeling with Latent Variables," for examples and general details.

In addition, because of the nature of latent variables, estimation in structural equation modeling with latent variables does not follow the same form as that of linear regression analysis. Instead of defining the estimators

in terms of the data matrices, most estimation methods in structural equation modeling use the fitting of the first- and second- order moments. Hence, estimation principles described in the section "Classical Estimation Principles" on page 37 do not apply to structural equation modeling. However, you can see the section "Estimation Criteria" on page 1805 in Chapter 33, "The CALIS Procedure," for details about estimation in structural equation modeling with latent variables.

Bayesian Models

Statistical models based on the classical (or *frequentist*) paradigm treat the parameters of the model as fixed, unknown constants. They are not random variables, and the notion of probability is derived in an objective sense as a limiting relative frequency. The Bayesian paradigm takes a different approach. Model parameters are random variables, and the probability of an event is defined in a subjective sense as the degree to which you believe that the event is true. This fundamental difference in philosophy leads to profound differences in the statistical content of estimation and inference. In the frequentist framework, you use the data to best estimate the unknown value of a parameter; you are trying to pinpoint a value in the parameter space as well as possible. In the Bayesian framework, you use the data to update your beliefs about the *behavior* of the parameter to assess its distributional properties as well as possible.

Suppose you are interested in estimating θ from data $\mathbf{Y} = [Y_1, \ldots, Y_n]$ by using a statistical model described by a density $p(\mathbf{y}|\theta)$. Bayesian philosophy states that θ cannot be determined exactly, and uncertainty about the parameter is expressed through probability statements and distributions. You can say, for example, that θ follows a normal distribution with mean 0 and variance 1, if you believe that this distribution best describes the uncertainty associated with the parameter.

The following steps describe the essential elements of Bayesian inference:

1. A probability distribution for θ is formulated as $\pi(\theta)$, which is known as the *prior* distribution, or just the prior. The prior distribution expresses your beliefs, for example, on the mean, the spread, the skewness, and so forth, about the parameter prior to examining the data.
2. Given the observed data $\mathbf{Y}$, you choose a statistical model $p(\mathbf{y}|\theta)$ to describe the distribution of $\mathbf{Y}$ given θ.
3. You update your beliefs about θ by combining information from the prior distribution and the data through the calculation of the *posterior* distribution, $p(\theta|\mathbf{y})$.

The third step is carried out by using Bayes' theorem, from which this branch of statistical philosophy derives its name. The theorem enables you to combine the prior distribution and the model in the following way:

$$p(\theta|\mathbf{y}) = \frac{p(\theta, \mathbf{y})}{p(\mathbf{y})} = \frac{p(\mathbf{y}|\theta)\pi(\theta)}{p(\mathbf{y})} = \frac{p(\mathbf{y}|\theta)\pi(\theta)}{\int p(\mathbf{y}|\theta)\pi(\theta)\, d\theta}$$

The quantity $p(\mathbf{y}) = \int p(\mathbf{y}|\theta)\pi(\theta)\, d\theta$ is the normalizing constant of the posterior distribution. It is also the marginal distribution of $\mathbf{Y}$, and it is sometimes called the marginal distribution of the data.

The likelihood function of θ is any function proportional to $p(\mathbf{y}|\theta)$—that is, $L(\theta) \propto p(\mathbf{y}|\theta)$. Another way of writing Bayes' theorem is

$$p(\theta|\mathbf{y}) = \frac{L(\theta)\pi(\theta)}{\int L(\theta)\pi(\theta)\, d\theta}$$

The marginal distribution $p(\mathbf{y})$ is an integral; therefore, provided that it is finite, the particular value of the integral does not yield any additional information about the posterior distribution. Hence, $p(\theta|\mathbf{y})$ can be written up to an arbitrary constant, presented here in proportional form, as

$$p(\theta|\mathbf{y}) \propto L(\theta)\pi(\theta)$$

Bayes' theorem instructs you how to update existing knowledge with new information. You start from a prior belief $\pi(\theta)$, and, after learning information from data $\mathbf{y}$, you change or update the belief on θ and obtain $p(\theta|\mathbf{y})$. These are the essential elements of the Bayesian approach to data analysis.

In theory, Bayesian methods offer a very simple alternative to statistical inference—all inferences follow from the posterior distribution $p(\theta|\mathbf{y})$. However, in practice, only the most elementary problems enable you to obtain the posterior distribution analytically. Most Bayesian analyses require sophisticated computations, including the use of simulation methods. You generate samples from the posterior distribution and use these samples to estimate the quantities of interest.

Both Bayesian and classical analysis methods have their advantages and disadvantages. Your choice of method might depend on the goals of your data analysis. If prior information is available, such as in the form of expert opinion or historical knowledge, and you want to incorporate this information into the analysis, then you might consider Bayesian methods. In addition, if you want to communicate your findings in terms of probability notions that can be more easily understood by nonstatisticians, Bayesian methods might be appropriate. The Bayesian paradigm can provide a framework for answering specific scientific questions that a single point estimate cannot sufficiently address. On the other hand, if you are interested in estimating parameters and in formulating inferences based on the properties of the parameter estimators, then there is no need to use Bayesian analysis. When the sample size is large, Bayesian inference often provides results for parametric models that are very similar to the results produced by classical, frequentist methods.

For more information, see Chapter 8, "Introduction to Bayesian Analysis Procedures."

Classical Estimation Principles

An estimation principle captures the set of rules and procedures by which parameter estimates are derived. When an estimation principle "meets" a statistical model, the result is an *estimation problem*, the solution of which are the parameter estimates. For example, if you apply the estimation principle of least squares to the SLR model $Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$, the estimation problem is to find those values $\widehat{\beta}_0$ and $\widehat{\beta}_1$ that minimize

$$\sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

The solutions are the least squares estimators.

The two most important classes of estimation principles in statistical modeling are the least squares principle and the likelihood principle. All principles have in common that they provide a metric by which you measure the distance between the data and the model. They differ in the nature of the metric; least squares relies on a geometric measure of distance, while likelihood inference is based on a distance that measures plausibility.

Least Squares

The idea of the ordinary least squares (OLS) principle is to choose parameter estimates that minimize the squared distance between the data and the model. In terms of the general, additive model,

$$Y_i = f(x_{i1}, \ldots, x_{ik}; \beta_1, \ldots, \beta_p) + \epsilon_i$$

the OLS principle minimizes

$$\text{SSE} = \sum_{i=1}^{n} \left(y_i - f(x_{i1}, \ldots, x_{ik}; \beta_1, \ldots, \beta_p)\right)^2$$

The least squares principle is sometimes called "nonparametric" in the sense that it does not require the distributional specification of the response or the error term, but it might be better termed "distributionally agnostic." In an additive-error model it is only required that the model errors have zero mean. For example, the specification

$$\begin{aligned} Y_i &= \beta_0 + \beta_1 x_i + \epsilon_i \\ \text{E}[\epsilon_i] &= 0 \end{aligned}$$

is sufficient to derive ordinary least squares (OLS) estimators for β_0 and β_1 and to study a number of their properties. It is easy to show that the OLS estimators in this SLR model are

$$\begin{aligned} \widehat{\beta}_1 &= \left(\sum_{i=1}^{n} (Y_i - \overline{Y}) \sum_{i=1}^{n} (x_i - \overline{x})\right) \Big/ \sum_{i=1}^{n} (x_i - \overline{x})^2 \\ \widehat{\beta}_0 &= \overline{Y} - \widehat{\beta}_1 \overline{x} \end{aligned}$$

Based on the assumption of a zero mean of the model errors, you can show that these estimators are unbiased, $\text{E}[\widehat{\beta}_1] = \beta_1$, $\text{E}[\widehat{\beta}_0] = \beta_0$. However, without further assumptions about the distribution of the ϵ_i, you cannot derive the variability of the least squares estimators or perform statistical inferences such as hypothesis tests or confidence intervals. In addition, depending on the distribution of the ϵ_i, other forms of least squares estimation can be more efficient than OLS estimation.

The conditions for which ordinary least squares estimation is efficient are zero mean, homoscedastic, uncorrelated model errors. Mathematically,

$$\begin{aligned} \text{E}[\epsilon_i] &= 0 \\ \text{Var}[\epsilon_i] &= \sigma^2 \\ \text{Cov}[\epsilon_i, \epsilon_j] &= 0 \text{ if } i \neq j \end{aligned}$$

The second and third assumption are met if the errors have an iid distribution—that is, if they are independent and identically distributed. Note, however, that the notion of stochastic independence is stronger than that of absence of correlation. Only if the data are normally distributed does the latter imply the former.

The various other forms of the least squares principle are motivated by different extensions of these assumptions in order to find more efficient estimators.

Weighted Least Squares

The objective function in weighted least squares (WLS) estimation is

$$\mathrm{SSE}_w = \sum_{i=1}^{n} w_i \left(Y_i - f(x_{i1}, \ldots, x_{ik}; \beta_1, \ldots, \beta_p)\right)^2$$

where w_i is a weight associated with the ith observation. A situation where WLS estimation is appropriate is when the errors are uncorrelated but not homoscedastic. If the weights for the observations are proportional to the reciprocals of the error variances, $\mathrm{Var}[\epsilon_i] = \sigma^2/w_i$, then the weighted least squares estimates are best linear unbiased estimators (BLUE). Suppose that the weights w_i are collected in the diagonal matrix $\mathbf{W}$ and that the mean function has the form of a linear model. The weighted sum of squares criterion then can be written as

$$\mathrm{SSE}_w = (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})' \mathbf{W} (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})$$

which gives rise to the weighted normal equations

$$(\mathbf{X}'\mathbf{W}\mathbf{X})\boldsymbol{\beta} = \mathbf{X}'\mathbf{W}\mathbf{Y}$$

The resulting WLS estimator of $\boldsymbol{\beta}$ is

$$\widehat{\boldsymbol{\beta}}_w = \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-} \mathbf{X}'\mathbf{W}\mathbf{Y}$$

Iteratively Reweighted Least Squares

If the weights in a least squares problem depend on the parameters, then a change in the parameters also changes the weight structure of the model. Iteratively reweighted least squares (IRLS) estimation is an iterative technique that solves a series of weighted least squares problems, where the weights are recomputed between iterations. IRLS estimation can be used, for example, to derive maximum likelihood estimates in generalized linear models.

Generalized Least Squares

The previously discussed least squares methods have in common that the observations are assumed to be uncorrelated—that is, $\mathrm{Cov}[\epsilon_i, \epsilon_j] = 0$, whenever $i \neq j$. The weighted least squares estimation problem is a special case of a more general least squares problem, where the model errors have a general covariance matrix, $\mathrm{Var}[\boldsymbol{\epsilon}] = \boldsymbol{\Sigma}$. Suppose again that the mean function is linear, so that the model becomes

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim (\mathbf{0}, \boldsymbol{\Sigma})$$

The generalized least squares (GLS) principle is to minimize the generalized error sum of squares

$$\mathrm{SSE}_g = (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})' \boldsymbol{\Sigma}^{-1} (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})$$

This leads to the generalized normal equations

$$(\mathbf{X}'\boldsymbol{\Sigma}^{-1}\mathbf{X})\boldsymbol{\beta} = \mathbf{X}'\boldsymbol{\Sigma}^{-1}\mathbf{Y}$$

and the GLS estimator

$$\widehat{\boldsymbol{\beta}}_g = \left(\mathbf{X}'\boldsymbol{\Sigma}^{-1}\mathbf{X}\right)^{-} \mathbf{X}'\boldsymbol{\Sigma}^{-1}\mathbf{Y}$$

Obviously, WLS estimation is a special case of GLS estimation, where $\boldsymbol{\Sigma} = \sigma^2\mathbf{W}^{-1}$—that is, the model is

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim \left(\mathbf{0}, \sigma^2\mathbf{W}^{-1}\right)$$

Likelihood

There are several forms of likelihood estimation and a large number of offshoot principles derived from it, such as pseudo-likelihood, quasi-likelihood, composite likelihood, etc. The basic likelihood principle is *maximum likelihood*, which asks to estimate the model parameters by those quantities that maximize the likelihood function of the data. The likelihood function is the joint distribution of the data, but in contrast to a probability mass or density function, it is thought of as a function of the parameters, given the data. The heuristic appeal of the maximum likelihood estimates (MLE) is that these are the values that make the observed data "most likely." Especially for discrete response data, the value of the likelihood function is the ordinate of a probability mass function, even if the likelihood is not a probability function. Since a statistical model is thought of as a representation of the data-generating mechanism, what could be more preferable as parameter estimates than those values that make it most likely that the data at hand will be observed?

Maximum likelihood estimates, if they exist, have appealing statistical properties. Under fairly mild conditions, they are best-asymptotic-normal (BAN) estimates—that is, their asymptotic distribution is normal, and no other estimator has a smaller asymptotic variance. However, their statistical behavior in finite samples is often difficult to establish, and you have to appeal to the asymptotic results that hold as the sample size tends to infinity. For example, maximum likelihood estimates are often biased estimates and the bias disappears as the sample size grows. A famous example is random sampling from a normal distribution. The corresponding statistical model is

$$
\begin{aligned}
Y_i &= \mu + \epsilon_i \\
\epsilon_i &\sim \text{iid}N(0, \sigma^2)
\end{aligned}
$$

where the symbol $\sim$ is read as "is distributed as" and iid is read as "independent and identically distributed." Under the normality assumption, the density function of y_i is

$$
f(y_i; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2}\left(\frac{y_i - \mu}{\sigma}\right)^2\right\}
$$

and the likelihood for a random sample of size n is

$$
L(\mu, \sigma^2; \mathbf{y}) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2}\left(\frac{y_i - \mu}{\sigma}\right)^2\right\}
$$

Maximizing the likelihood function $L(\mu, \sigma^2; \mathbf{y})$ is equivalent to maximizing the log-likelihood function $\log L = l(\mu, \sigma^2; \mathbf{y})$,

$$
\begin{aligned}
l(\mu, \sigma^2; \mathbf{y}) &= \sum_{i=1}^{n} -\frac{1}{2}\left(\log\{2\pi\} + \frac{(y_i - \mu)^2}{\sigma^2} + \log\{\sigma^2\}\right) \\
&= -\frac{1}{2}\left(n\log\{2\pi\} + n\log\{\sigma^2\} + \sum_{i=1}^{n}(y_i - \mu)^2/\sigma^2\right)
\end{aligned}
$$

The maximum likelihood estimators of μ and σ^2 are thus

$$
\widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} Y_i = \overline{Y} \qquad \widehat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \widehat{\mu})^2
$$

The MLE of the mean μ is the sample mean, and it is an unbiased estimator of μ. However, the MLE of the variance σ^2 is not an unbiased estimator. It has bias

$$\mathrm{E}\left[\widehat{\sigma}^2 - \sigma^2\right] = -\frac{1}{n}\sigma^2$$

As the sample size n increases, the bias vanishes.

For certain classes of models, special forms of likelihood estimation have been developed to maintain the appeal of likelihood-based statistical inference and to address specific properties that are believed to be shortcomings:

- The bias in maximum likelihood parameter estimators of variances and covariances has led to the development of restricted (or residual) maximum likelihood (REML) estimators that play an important role in mixed models.
- Quasi-likelihood methods do not require that the joint distribution of the data be specified. These methods derive estimators based on only the first two moments (mean and variance) of the joint distributions and play an important role in the analysis of correlated data.
- The idea of composite likelihood is applied in situations where the likelihood of the vector of responses is intractable but the likelihood of components or functions of the full-data likelihood are tractable. For example, instead of the likelihood of $\mathbf{Y}$, you might consider the likelihood of pairwise differences $Y_i - Y_j$.
- The pseudo-likelihood concept is also applied when the likelihood function is intractable, but the likelihood of a related, simpler model is available. An important difference between quasi-likelihood and pseudo-likelihood techniques is that the latter make distributional assumptions to obtain a likelihood function in the pseudo-model. Quasi-likelihood methods do not specify the distributional family.
- The penalized likelihood principle is applied when additional constraints and conditions need to be imposed on the parameter estimates or the resulting model fit. For example, you might augment the likelihood with conditions that govern the smoothness of the predictions or that prevent overfitting of the model.

Least Squares or Likelihood

For many statistical modeling problems, you have a choice between a least squares principle and the maximum likelihood principle. Table 3.1 compares these two basic principles.

Table 3.1 Least Squares and Maximum Likelihood

Criterion	Least Squares	Maximum Likelihood
Requires specification of joint distribution of data	No, but in order to perform confirmatory inference (tests, confidence intervals), a distributional assumption is needed, or an appeal to asymptotics.	Yes, no progress can be made with the genuine likelihood principle without knowing the distribution of the data.

Table 3.1 *continued*

Criterion	Least Squares	Maximum Likelihood
All parameters of the model are estimated	No. In the additive-error type models, least squares provides estimates of only the parameters in the mean function. The residual variance, for example, must be estimated by some other method—typically by using the mean squared error of the model.	Yes
Estimates always exist	Yes, but they might not be unique, such as when the **X** matrix is singular.	No, maximum likelihood estimates do not exist for all estimation problems.
Estimators are biased	Unbiased, provided that the model is correct—that is, the errors have zero mean.	Often biased, but asymptotically unbiased
Estimators are consistent	Not necessarily, but often true. Sometimes estimators are consistent even in a misspecified model, such as when misspecification is in the covariance structure.	Almost always
Estimators are best linear unbiased estimates (BLUE)	Typically, if the least squares assumptions are met.	Not necessarily: estimators are often nonlinear in the data and are often biased.
Asymptotically most efficient	Not necessarily	Typically
Easy to compute	Yes	No

Inference Principles for Survey Data

Design-based and model-assisted statistical inference for survey data requires that the randomness due to the selection mechanism be taken into account. This can require special estimation principles and techniques.

The SURVEYMEANS, SURVEYFREQ, SURVEYREG, SURVEYLOGISTIC, and SURVEYPHREG procedures support design-based and/or model-assisted inference for sample surveys. Suppose π_i is the selection probability for unit i in sample S. The inverse of the inclusion probability is known as sampling weight and is denoted by w_i. Briefly, the idea is to apply a relationship that exists in the population to the sample and to take into account the sampling weights. For example, to estimate the finite population total $T_N = \sum_{i \in U_N} y_i$ based on the sample S, you can accumulate the sampled values while properly weighting: $\widehat{T}_\pi = \sum_{i \in S} w_i y_i$. It is easy to verify that $\widehat{T}_\pi$ is design-unbiased in the sense that $\mathrm{E}[\widehat{T}_\pi | \mathcal{F}_N] = T_N$ (see Cochran 1977).

When a statistical model is present, similar ideas apply. For example, if β_{N0} and β_{N1} are finite population

quantities for a simple linear regression working model that minimize the sum of squares

$$\sum_{i \in U_N} (y_i - \beta_{0N} - \beta_{1N} x_i)^2$$

in the population, then the sample-based estimators $\widehat{\beta}_{0S}$ and $\widehat{\beta}_{1S}$ are obtained by minimizing the weighted sum of squares

$$\sum_{i \in S} w_i (y_i - \hat{\beta}_{0S} - \hat{\beta}_{1S} x_i)^2$$

in the sample, taking into account the inclusion probabilities.

In model-assisted inference, weighted least squares or pseudo-maximum likelihood estimators are commonly used to solve such estimation problems. Maximum pseudo-likelihood or weighted maximum likelihood estimators for survey data maximize a sample-based estimator of the population likelihood. Assume a working model with uncorrelated responses such that the finite population log-likelihood is

$$\sum_{i \in U_N} l(\theta_{1N}, \ldots, \theta_{pN}; y_i),$$

where $\theta_{1N}, \ldots, \theta_{pN}$ are finite population quantities. For independent sampling, one possible sample-based estimator of the population log likelihood is

$$\sum_{i \in S} w_i l(\theta_{1N}, \ldots, \theta_{pN}; y_i)$$

Sample-based estimators $\widehat{\theta}_{1S}, \ldots, \widehat{\theta}_{pS}$ are obtained by maximizing this expression.

Design-based and model-based statistical analysis might employ the same statistical model (for example, a linear regression) and the same estimation principle (for example, weighted least squares), and arrive at the same estimates. The design-based estimation of the precision of the estimators differs from the model-based estimation, however. For complex surveys, design-based variance estimates are in general different from their model-based counterpart. The SAS/STAT procedures for survey data (SURVEYMEANS, SURVEYFREQ, SURVEYREG, SURVEYLOGISTIC, and SURVEYPHREG procedures) compute design-based variance estimates for complex survey data. See the section "Variance Estimation" on page 271, in Chapter 15, "Introduction to Survey Procedures," for details about design-based variance estimation.

Statistical Background

Hypothesis Testing and Power

In statistical hypothesis testing, you typically express the belief that some effect exists in a population by specifying an alternative hypothesis H_1. You state a null hypothesis H_0 as the assertion that the effect does *not* exist and attempt to gather evidence to reject H_0 in favor of H_1. Evidence is gathered in the form of sample data, and a statistical test is used to assess H_0. If H_0 is rejected but there really is *no* effect, this is called a *Type I error*. The probability of a Type I error is usually designated "alpha" or α, and statistical tests are designed to ensure that α is suitably small (for example, less than 0.05).

If there is an effect in the population but H_0 is *not* rejected in the statistical test, then a *Type II error* has been committed. The probability of a Type II error is usually designated "beta" or β. The probability $1 - \beta$ of avoiding a Type II error—that is, correctly rejecting H_0 and achieving statistical significance, is called the *power* of the test.

An important goal in study planning is to ensure an acceptably high level of power. Sample size plays a prominent role in power computations because the focus is often on determining a sufficient sample size to achieve a certain power, or assessing the power for a range of different sample sizes.

There are several tools available in SAS/STAT software for power and sample size analysis. PROC POWER covers a variety of analyses such as t tests, equivalence tests, confidence intervals, binomial proportions, multiple regression, one-way ANOVA, survival analysis, logistic regression, and the Wilcoxon rank-sum test. PROC GLMPOWER supports more complex linear models. The Power and Sample Size application provides a user interface and implements many of the analyses supported in the procedures.

Important Linear Algebra Concepts

A *matrix* $\mathbf{A}$ is a rectangular array of numbers. The *order* of a matrix with n rows and k columns is $(n \times k)$. The element in row i, column j of $\mathbf{A}$ is denoted as a_{ij}, and the notation $\left[a_{ij}\right]$ is sometimes used to refer to the two-dimensional row-column array

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1k} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2k} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nk} \end{bmatrix} = \left[a_{ij}\right]$$

A *vector* is a one-dimensional array of numbers. A *column vector* has a single column ($k = 1$). A *row vector* has a single row ($n = 1$). A *scalar* is a matrix of order (1×1)—that is, a single number. A *square* matrix has the same row and column order, $n = k$. A *diagonal* matrix is a square matrix where all off-diagonal elements are zero, $a_{ij} = 0$ if $i \neq j$. The *identity* matrix $\mathbf{I}$ is a diagonal matrix with $a_{ii} = 1$ for all i. The *unit vector* $\mathbf{1}$ is a vector where all elements are 1. The *unit matrix* $\mathbf{J}$ is a matrix of all 1s. Similarly, the elements of the null vector and the null matrix are all 0.

Basic matrix operations are as follows:

Addition — If $\mathbf{A}$ and $\mathbf{B}$ are of the same order, then $\mathbf{A}+\mathbf{B}$ is the matrix of elementwise sums,

$$\mathbf{A}+\mathbf{B}=\left[a_{ij}+b_{ij}\right]$$

Subtraction — If $\mathbf{A}$ and $\mathbf{B}$ are of the same order, then $\mathbf{A}-\mathbf{B}$ is the matrix of elementwise differences,

$$\mathbf{A}-\mathbf{B}=\left[a_{ij}-b_{ij}\right]$$

Dot product — The dot product of two n-vectors $\mathbf{a}$ and $\mathbf{b}$ is the sum of their elementwise products,

$$\mathbf{a}\cdot\mathbf{b}=\sum_{i=1}^{n}a_i b_i$$

The dot product is also known as the *inner product* of $\mathbf{a}$ and $\mathbf{b}$. Two vectors are said to be *orthogonal* if their dot product is zero.

Multiplication — Matrices $\mathbf{A}$ and $\mathbf{B}$ are said to be conformable for $\mathbf{AB}$ multiplication if the number of columns in $\mathbf{A}$ equals the number of rows in $\mathbf{B}$. Suppose that $\mathbf{A}$ is of order $(n\times k)$ and that $\mathbf{B}$ is of order $(k\times p)$. The product $\mathbf{AB}$ is then defined as the $(n\times p)$ matrix of the dot products of the ith row of $\mathbf{A}$ and the jth column of $\mathbf{B}$,

$$\mathbf{AB}=\left[\mathbf{a_i}\cdot\mathbf{b_j}\right]_{n\times p}$$

Transposition — The transpose of the $(n\times k)$ matrix $\mathbf{A}$ is denoted as $\mathbf{A}'$ and is obtained by interchanging the rows and columns,

$$\mathbf{A}'=\begin{bmatrix} a_{11} & a_{21} & a_{31} & \cdots & a_{n1} \\ a_{12} & a_{22} & a_{23} & \cdots & a_{n2} \\ a_{13} & a_{23} & a_{33} & \cdots & a_{n3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{1k} & a_{2k} & a_{3k} & \cdots & a_{nk} \end{bmatrix}=\left[a_{ji}\right]$$

A *symmetric* matrix is equal to its transpose, $\mathbf{A}=\mathbf{A}'$. The inner product of two $(n\times 1)$ column vectors $\mathbf{a}$ and $\mathbf{b}$ is $\mathbf{a}\cdot\mathbf{b}=\mathbf{a}'\mathbf{b}$.

Matrix Inversion

Regular Inverses

The right inverse of a matrix $\mathbf{A}$ is the matrix that yields the identity when $\mathbf{A}$ is postmultiplied by it. Similarly, the left inverse of $\mathbf{A}$ yields the identity if $\mathbf{A}$ is premultiplied by it. $\mathbf{A}$ is said to be invertible and $\mathbf{B}$ is said to be the inverse of $\mathbf{A}$, if $\mathbf{B}$ is its right and left inverse, $\mathbf{BA}=\mathbf{AB}=\mathbf{I}$. This requires $\mathbf{A}$ to be square and nonsingular. The inverse of a matrix $\mathbf{A}$ is commonly denoted as $\mathbf{A}^{-1}$. The following results are useful in manipulating inverse matrices (assuming both $\mathbf{A}$ and $\mathbf{C}$ are invertible):

$$\begin{aligned}
\mathbf{AA}^{-1}&=\mathbf{A}^{-1}\mathbf{A}=\mathbf{I}\\
\left(\mathbf{A}'\right)^{-1}&=\left(\mathbf{A}^{-1}\right)'\\
\left(\mathbf{A}^{-1}\right)^{-1}&=\mathbf{A}\\
(\mathbf{AC})^{-1}&=\mathbf{C}^{-1}\mathbf{A}^{-1}\\
\operatorname{rank}(\mathbf{A})&=\operatorname{rank}\left(\mathbf{A}^{-1}\right)
\end{aligned}$$

If $\mathbf{D}$ is a diagonal matrix with nonzero entries on the diagonal—that is, $\mathbf{D} = \operatorname{diag}(d_1, \ldots, d_n)$—then $\mathbf{D}^{-1} = \operatorname{diag}(1/d_1, \ldots, 1/d_n)$. If $\mathbf{D}$ is a block-diagonal matrix whose blocks are invertible, then

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_2 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{D}_3 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{D}_n \end{bmatrix} \qquad \mathbf{D}^{-1} = \begin{bmatrix} \mathbf{D}_1^{-1} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_2^{-1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{D}_3^{-1} & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{D}_n^{-1} \end{bmatrix}$$

In statistical applications the following two results are particularly important, because they can significantly reduce the computational burden in working with inverse matrices.

Partitioned Matrix Suppose $\mathbf{A}$ is a nonsingular matrix that is partitioned as

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}$$

Then, provided that all the inverses exist, the inverse of $\mathbf{A}$ is given by

$$\mathbf{A}^{-1} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix}$$

where $\mathbf{B}_{11} = \left(\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{21}\right)^{-1}$, $\mathbf{B}_{12} = -\mathbf{B}_{11}\mathbf{A}_{12}\mathbf{A}_{22}^{-1}$, $\mathbf{B}_{21} = -\mathbf{A}_{22}^{-1}\mathbf{A}_{21}\mathbf{B}_{11}$, and $\mathbf{B}_{22} = \left(\mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\right)^{-1}$.

Patterned Sum Suppose $\mathbf{R}$ is $(n \times n)$ nonsingular, $\mathbf{G}$ is $(k \times k)$ nonsingular, and $\mathbf{B}$ and $\mathbf{C}$ are $(n \times k)$ and $(k \times n)$ matrices, respectively. Then the inverse of $\mathbf{R} + \mathbf{BGC}$ is given by

$$(\mathbf{R} + \mathbf{BGC})^{-1} = \mathbf{R}^{-1} - \mathbf{R}^{-1}\mathbf{B}\left(\mathbf{G}^{-1} + \mathbf{C}\mathbf{R}^{-1}\mathbf{B}\right)^{-1}\mathbf{C}\mathbf{R}^{-1}$$

This formula is particularly useful if $k << n$ and $\mathbf{R}$ has a simple form that is easy to invert. This case arises, for example, in mixed models where $\mathbf{R}$ might be a diagonal or block-diagonal matrix, and $\mathbf{B} = \mathbf{C}'$.

Another situation where this formula plays a critical role is in the computation of regression diagnostics, such as in determining the effect of removing an observation from the analysis. Suppose that $\mathbf{A} = \mathbf{X}'\mathbf{X}$ represents the crossproduct matrix in the linear model $\mathrm{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{\beta}$. If $\mathbf{x}_i'$ is the ith row of the $\mathbf{X}$ matrix, then $(\mathbf{X}'\mathbf{X} - \mathbf{x}_i\mathbf{x}_i')$ is the crossproduct matrix in the same model with the ith observation removed. Identifying $\mathbf{B} = -\mathbf{x}_i$, $\mathbf{C} = \mathbf{x}_i'$, and $\mathbf{G} = \mathbf{I}$ in the preceding inversion formula, you can obtain the expression for the inverse of the crossproduct matrix:

$$\left(\mathbf{X}'\mathbf{X} - \mathbf{x}_i\mathbf{x}_i'\right)^{-1} = \left(\mathbf{X}'\mathbf{X}\right)^{-1} + \frac{\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{x}_i\mathbf{x}_i'\left(\mathbf{X}'\mathbf{X}\right)^{-1}}{1 - \mathbf{x}_i'\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{x}_i}$$

This expression for the inverse of the reduced data crossproduct matrix enables you to compute "leave-one-out" deletion diagnostics in linear models without refitting the model.

Generalized Inverse Matrices

If $\mathbf{A}$ is rectangular (not square) or singular, then it is not invertible and the matrix $\mathbf{A}^{-1}$ does not exist. Suppose you want to find a solution to simultaneous linear equations of the form

$$\mathbf{Ab} = \mathbf{c}$$

If $\mathbf{A}$ is square and nonsingular, then the unique solution is $\mathbf{b} = \mathbf{A}^{-1}\mathbf{c}$. In statistical applications, the case where $\mathbf{A}$ is $(n \times k)$ rectangular is less important than the case where $\mathbf{A}$ is a $(k \times k)$ square matrix of rank less than k. For example, the normal equations in ordinary least squares (OLS) estimation in the model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ are

$$\left(\mathbf{X}'\mathbf{X}\right)\boldsymbol{\beta} = \mathbf{X}'\mathbf{Y}$$

A *generalized inverse* matrix is a matrix $\mathbf{A}^{-}$ such that $\mathbf{A}^{-}\mathbf{c}$ is a solution to the linear system. In the OLS example, a solution can be found as $\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{Y}$, where $\left(\mathbf{X}'\mathbf{X}\right)^{-}$ is a generalized inverse of $\mathbf{X}'\mathbf{X}$.

The following four conditions are often associated with generalized inverses. For the square or rectangular matrix $\mathbf{A}$ there exist matrices $\mathbf{G}$ that satisfy

$$\begin{array}{llcl}
\text{(i)} & \mathbf{AGA} & = & \mathbf{A} \\
\text{(ii)} & \mathbf{GAG} & = & \mathbf{G} \\
\text{(iii)} & (\mathbf{AG})' & = & \mathbf{AG} \\
\text{(iv)} & (\mathbf{GA})' & = & \mathbf{GA}
\end{array}$$

The matrix $\mathbf{G}$ that satisfies all four conditions is unique and is called the Moore-Penrose inverse, after the first published work on generalized inverses by Moore (1920) and the subsequent definition by Penrose (1955). Only the first condition is required, however, to provide a solution to the linear system above.

Pringle and Rayner (1971) introduced a numbering system to distinguish between different types of generalized inverses. A matrix that satisfies only condition (i) is a g_1-inverse. The g_2-inverse satisfies conditions (i) and (ii). It is also called a *reflexive* generalized inverse. Matrices satisfying conditions (i)–(iii) or conditions (i), (ii), and (iv) are g_3-inverses. Note that a matrix that satisfies the first three conditions is a right generalized inverse, and a matrix that satisfies conditions (i), (ii), and (iv) is a left generalized inverse. For example, if $\mathbf{B}$ is $(n \times k)$ of rank k, then $\left(\mathbf{B}'\mathbf{B}\right)^{-1}\mathbf{B}'$ is a left generalized inverse of $\mathbf{B}$. The notation g_4-inverse for the Moore-Penrose inverse, satisfying conditions (i)–(iv), is often used by extension, but note that Pringle and Rayner (1971) do not use it; rather, they call such a matrix "the" generalized inverse.

If the $(n \times k)$ matrix $\mathbf{X}$ is rank-deficient—that is, $\operatorname{rank}(\mathbf{X}) < \min\{n, k\}$—then the system of equations

$$\left(\mathbf{X}'\mathbf{X}\right)\boldsymbol{\beta} = \mathbf{X}'\mathbf{Y}$$

does not have a unique solution. A particular solution depends on the choice of the generalized inverse. However, some aspects of the statistical inference are invariant to the choice of the generalized inverse. If $\mathbf{G}$ is a generalized inverse of $\mathbf{X}'\mathbf{X}$, then $\mathbf{XGX}'$ is invariant to the choice of $\mathbf{G}$. This result comes into play, for example, when you are computing predictions in an OLS model with a rank-deficient $\mathbf{X}$ matrix, since it implies that the predicted values

$$\mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{y}$$

are invariant to the choice of $\left(\mathbf{X}'\mathbf{X}\right)^{-}$.

Matrix Differentiation

Taking the derivative of expressions involving matrices is a frequent task in statistical estimation. Objective functions that are to be minimized or maximized are usually written in terms of model matrices and/or vectors whose elements depend on the unknowns of the estimation problem. Suppose that $\mathbf{A}$ and $\mathbf{B}$ are real matrices whose elements depend on the scalar quantities β and θ—that is, $\mathbf{A} = \left[a_{ij}(\beta, \theta)\right]$, and similarly for $\mathbf{B}$.

The following are useful results in finding the derivative of elements of a matrix and of functions involving a matrix. For more in-depth discussion of matrix differentiation and matrix calculus, see, for example, Magnus and Neudecker (1999) and Harville (1997).

The derivative of $\mathbf{A}$ with respect to β is denoted $\dot{\mathbf{A}}_\beta$ and is the matrix of the first derivatives of the elements of $\mathbf{A}$:

$$\dot{\mathbf{A}}_\beta = \frac{\partial}{\partial\beta}\mathbf{A} = \left[\frac{\partial a_{ij}(\beta, \theta)}{\partial\beta}\right]$$

Similarly, the second derivative of $\mathbf{A}$ with respect to β and θ is the matrix of the second derivatives

$$\ddot{\mathbf{A}}_{\beta\theta} = \frac{\partial^2}{\partial\beta\partial\theta}\mathbf{A} = \left[\frac{\partial^2 a_{ij}(\beta, \theta)}{\partial\beta\partial\theta}\right]$$

The following are some basic results involving sums, products, and traces of matrices:

$$\begin{aligned}
\frac{\partial}{\partial\beta}c_1\mathbf{A} &= c_1\dot{\mathbf{A}}_\beta \\
\frac{\partial}{\partial\beta}(\mathbf{A}+\mathbf{B}) &= \dot{\mathbf{A}}_\beta + \dot{\mathbf{B}}_\beta \\
\frac{\partial}{\partial\beta}(c_1\mathbf{A}+c_2\mathbf{B}) &= c_1\dot{\mathbf{A}}_\beta + c_2\dot{\mathbf{B}}_\beta \\
\frac{\partial}{\partial\beta}\mathbf{AB} &= \mathbf{A}\dot{\mathbf{B}}_\beta + \dot{\mathbf{A}}_\beta\mathbf{B} \\
\frac{\partial}{\partial\beta}\mathrm{trace}(\mathbf{A}) &= \mathrm{trace}\left(\dot{\mathbf{A}}_\beta\right) \\
\frac{\partial}{\partial\beta}\mathrm{trace}(\mathbf{AB}) &= \mathrm{trace}\left(\mathbf{A}\dot{\mathbf{B}}_\beta\right) + \mathrm{trace}\left(\dot{\mathbf{A}}_\beta\mathbf{B}\right)
\end{aligned}$$

The next set of results is useful in finding the derivative of elements of $\mathbf{A}$ and of functions of $\mathbf{A}$, if $\mathbf{A}$ is a

nonsingular matrix:

$$\begin{aligned}
\frac{\partial}{\partial\beta}\mathbf{x}'\mathbf{A}^{-1}\mathbf{x} &= -\,\mathbf{x}'\mathbf{A}^{-1}\dot{\mathbf{A}}_\beta\mathbf{A}^{-1}\mathbf{x} \\
\frac{\partial}{\partial\beta}\mathbf{A}^{-1} &= -\,\mathbf{A}^{-1}\dot{\mathbf{A}}_\beta\mathbf{A}^{-1} \\
\frac{\partial}{\partial\beta}|\mathbf{A}| &= |\mathbf{A}|\,\mathrm{trace}\left(\mathbf{A}^{-1}\dot{\mathbf{A}}_\beta\right) \\
\frac{\partial}{\partial\beta}\log\{|\mathbf{A}|\} &= \frac{1}{|\mathbf{A}|}\,\frac{\partial}{\partial\beta}\mathbf{A} = \mathrm{trace}\left(\mathbf{A}^{-1}\dot{\mathbf{A}}_\beta\right) \\
\frac{\partial^2}{\partial\beta\partial\theta}\mathbf{A}^{-1} &= -\,\mathbf{A}^{-1}\ddot{\mathbf{A}}_{\beta\theta}\mathbf{A}^{-1} + \mathbf{A}^{-1}\dot{\mathbf{A}}_\beta\mathbf{A}^{-1}\dot{\mathbf{A}}_\theta\mathbf{A}^{-1} + \mathbf{A}^{-1}\dot{\mathbf{A}}_\theta\mathbf{A}^{-1}\dot{\mathbf{A}}_\beta\mathbf{A}^{-1} \\
\frac{\partial^2}{\partial\beta\partial\theta}\log\{|\mathbf{A}|\} &= \mathrm{trace}\left(\mathbf{A}^{-1}\ddot{\mathbf{A}}_{\beta\theta}\right) - \mathrm{trace}\left(\mathbf{A}^{-1}\dot{\mathbf{A}}_\beta\mathbf{A}^{-1}\dot{\mathbf{A}}_\theta\right)
\end{aligned}$$

Now suppose that $\mathbf{a}$ and $\mathbf{b}$ are column vectors that depend on β and/or θ and that $\mathbf{x}$ is a vector of constants. The following results are useful for manipulating derivatives of linear and quadratic forms:

$$\begin{aligned}
\frac{\partial}{\partial\mathbf{x}}\mathbf{a}'\mathbf{x} &= \mathbf{a} \\
\frac{\partial}{\partial\mathbf{x}'}\mathbf{B}\mathbf{x} &= \mathbf{B} \\
\frac{\partial}{\partial\mathbf{x}}\mathbf{x}'\mathbf{B}\mathbf{x} &= \left(\mathbf{B}+\mathbf{B}'\right)\mathbf{x} \\
\frac{\partial^2}{\partial\mathbf{x}\partial\mathbf{x}'}\mathbf{x}'\mathbf{B}\mathbf{x} &= \mathbf{B}+\mathbf{B}'
\end{aligned}$$

Matrix Decompositions

To decompose a matrix is to express it as a function—typically a product—of other matrices that have particular properties such as orthogonality, diagonality, triangularity. For example, the Cholesky decomposition of a symmetric positive definite matrix $\mathbf{A}$ is $\mathbf{C}\mathbf{C}' = \mathbf{A}$, where $\mathbf{C}$ is a lower-triangular matrix. The spectral decomposition of a symmetric matrix is $\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}'$, where $\mathbf{D}$ is a diagonal matrix and $\mathbf{P}$ is an orthogonal matrix.

Matrix decomposition play an important role in statistical theory as well as in statistical computations. Calculations in terms of decompositions can have greater numerical stability. Decompositions are often necessary to extract information about matrices, such as matrix rank, eigenvalues, or eigenvectors. Decompositions are also used to form special transformations of matrices, such as to form a "square-root" matrix. This section briefly mentions several decompositions that are particularly prevalent and important.

LDU, LU, and Cholesky Decomposition

Every square matrix $\mathbf{A}$, whether it is positive definite or not, can be expressed in the form $\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{U}$, where $\mathbf{L}$ is a unit lower-triangular matrix, $\mathbf{D}$ is a diagonal matrix, and $\mathbf{U}$ is a unit upper-triangular matrix. (The diagonal elements of a unit triangular matrix are 1.) Because of the arrangement of the matrices, the decomposition is called the LDU decomposition. Since you can absorb the diagonal matrix into the triangular

matrices, the decomposition

$$\mathbf{A} = \mathbf{L}\mathbf{D}^{1/2}\mathbf{D}^{1/2}\mathbf{U} = \mathbf{L}^*\mathbf{U}^*$$

is also referred to as the LU decomposition of $\mathbf{A}$.

If the matrix $\mathbf{A}$ is positive definite, then the diagonal elements of $\mathbf{D}$ are positive and the LDU decomposition is unique. If $\mathbf{A}$ is also symmetric, then the unique decomposition takes the form $\mathbf{A} = \mathbf{U}'\mathbf{D}\mathbf{U}$, where $\mathbf{U}$ is unit upper-triangular and $\mathbf{D}$ is diagonal with positive elements. Absorbing the square root of $\mathbf{D}$ into $\mathbf{U}$, $\mathbf{C} = \mathbf{D}^{1/2}\mathbf{U}$, the decomposition is known as the *Cholesky* decomposition of a positive-definite matrix:

$$\mathbf{A} = \mathbf{U}'\mathbf{D}^{1/2}\,\mathbf{D}^{1/2}\mathbf{U} = \mathbf{C}'\mathbf{C}$$

where $\mathbf{C}$ is upper triangular.

If $\mathbf{A}$ is symmetric but only nonnegative definite of rank k, rather than being positive definite of full rank, then it has an extended Cholesky decomposition as follows. Let $\mathbf{C}^*$ denote the lower-triangular matrix such that

$$\mathbf{C}^* = \begin{bmatrix} \mathbf{C}_{k\times k} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Then $\mathbf{A} = \mathbf{C}\mathbf{C}'$.

Spectral Decomposition

Suppose that $\mathbf{A}$ is an $(n \times n)$ symmetric matrix. Then there exists an orthogonal matrix $\mathbf{Q}$ and a diagonal matrix $\mathbf{D}$ such that $\mathbf{A} = \mathbf{Q}\mathbf{D}\mathbf{Q}'$. Of particular importance is the case where the orthogonal matrix is also orthonormal—that is, its column vectors have unit norm. Denote this orthonormal matrix as $\mathbf{P}$. Then the corresponding diagonal matrix—$\mathbf{\Lambda} = \mathrm{diag}(\lambda_i, \ldots, \lambda_n)$, say—contains the eigenvalues of $\mathbf{A}$. The spectral decomposition of $\mathbf{A}$ can be written as

$$\mathbf{A} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}' = \sum_{i=1}^{n} \lambda_i \mathbf{p}_i \mathbf{p}_i'$$

where $\mathbf{p}_i$ denotes the ith column vector of $\mathbf{P}$. The right-side expression decomposes $\mathbf{A}$ into a sum of rank-1 matrices, and the weight of each contribution is equal to the eigenvalue associated with the ith eigenvector. The sum furthermore emphasizes that the rank of $\mathbf{A}$ is equal to the number of nonzero eigenvalues.

Harville (1997, p. 538) refers to the spectral decomposition of $\mathbf{A}$ as the decomposition that takes the previous sum one step further and accumulates contributions associated with the distinct eigenvalues. If $\lambda_i^*, \ldots, \lambda_k^*$ are the distinct eigenvalues and $\mathbf{E}_j = \sum \mathbf{p}_i \mathbf{p}_i'$, where the sum is taken over the set of columns for which $\lambda_i = \lambda_j^*$, then

$$\mathbf{A} = \sum_{i=1}^{k} \lambda_j^* \mathbf{E}_j$$

You can employ the spectral decomposition of a nonnegative definite symmetric matrix to form a "square-root" matrix of $\mathbf{A}$. Suppose that $\mathbf{\Lambda}^{1/2}$ is the diagonal matrix containing the square roots of the λ_i. Then $\mathbf{B} = \mathbf{P}\mathbf{\Lambda}^{1/2}\mathbf{P}'$ is a square-root matrix of $\mathbf{A}$ in the sense that $\mathbf{B}\mathbf{B} = \mathbf{A}$, because

$$\mathbf{B}\mathbf{B} = \mathbf{P}\mathbf{\Lambda}^{1/2}\mathbf{P}'\mathbf{P}\mathbf{\Lambda}^{1/2}\mathbf{P}' = \mathbf{P}\mathbf{\Lambda}^{1/2}\mathbf{\Lambda}^{1/2}\mathbf{P}' = \mathbf{P}\mathbf{\Lambda}\mathbf{P}'$$

Generating the Moore-Penrose inverse of a matrix based on the spectral decomposition is also simple. Denote as $\mathbf{\Delta}$ the diagonal matrix with typical element

$$\delta_i = \begin{cases} 1/\lambda_i & \lambda_i \neq 0 \\ 0 & \lambda_i = 0 \end{cases}$$

Then the matrix $\mathbf{P\Delta P}' = \sum \delta_i \mathbf{p}_i \mathbf{p}_i'$ is the Moore-Penrose (g_4-generalized) inverse of $\mathbf{A}$.

Singular-Value Decomposition

The singular-value decomposition is related to the spectral decomposition of a matrix, but it is more general. The singular-value decomposition can be applied to any matrix. Let $\mathbf{B}$ be an $(n \times p)$ matrix of rank k. Then there exist orthogonal matrices P and Q of order $(n \times n)$ and $(p \times p)$, respectively, and a diagonal matrix $\mathbf{D}$ such that

$$\mathbf{P}'\mathbf{BQ} = \mathbf{D} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$$

where $\mathbf{D}_1$ is a diagonal matrix of order k. The diagonal elements of $\mathbf{D}_1$ are strictly positive. As with the spectral decomposition, this result can be written as a decomposition of $\mathbf{B}$ into a weighted sum of rank-1 matrices

$$\mathbf{B} = -\mathbf{PDQ}' = \sum_{i=1}^{n} d_i \mathbf{p}_i \mathbf{q}_i'$$

The scalars $d_1, \ldots, d_k$ are called the *singular values* of the matrix $\mathbf{B}$. They are the positive square roots of the nonzero eigenvalues of the matrix $\mathbf{B}'\mathbf{B}$. If the singular-value decomposition is applied to a symmetric, nonnegative definite matrix $\mathbf{A}$, then the singular values $d_1, \ldots, d_n$ are the nonzero eigenvalues of $\mathbf{A}$ and the singular-value decomposition is the same as the spectral decomposition.

As with the spectral decomposition, you can use the results of the singular-value decomposition to generate the Moore-Penrose inverse of a matrix. If $\mathbf{B}$ is $(n \times p)$ with singular-value decomposition $\mathbf{PDQ}'$, and if $\mathbf{\Delta}$ is a diagonal matrix with typical element

$$\delta_i = \begin{cases} 1/d_i & |d_i| \neq 0 \\ 0 & d_i = 0 \end{cases}$$

then $\mathbf{Q\Delta P}'$ is the g_4-generalized inverse of $\mathbf{B}$.

Expectations of Random Variables and Vectors

If Y is a discrete random variable with mass function $p(y)$ and support (possible values) $y_1, y_2, \ldots$, then the expectation (expected value) of Y is defined as

$$\mathrm{E}[Y] = \sum_{j=1}^{\infty} y_j \, p(y_j)$$

provided that $\sum |y_j| p(y_j) < \infty$, otherwise the sum in the definition is not well-defined. The expected value of a function $h(y)$ is similarly defined: provided that $\sum |h(y_j)| p(y_j) < \infty$,

$$\mathrm{E}[h(Y)] = \sum_{j=1}^{\infty} h(y_j) \, p(y_j)$$

For continuous random variables, similar definitions apply, but summation is replaced by integration over the support of the random variable. If X is a continuous random variable with density function $f(x)$, and $\int |x| f(x)\, dx < \infty$, then the expectation of X is defined as

$$\mathrm{E}[X] = \int_{-\infty}^{\infty} x f(x)\, dx$$

The expected value of a random variable is also called its *mean* or its first moment. A particularly important function of a random variable is $h(Y) = (Y - \mathrm{E}[Y])^2$. The expectation of $h(Y)$ is called the *variance* of Y or the second central moment of Y. When you study the properties of multiple random variables, then you might be interested in aspects of their joint distribution. The covariance between random variables Y and X is defined as the expected value of the function $(Y - \mathrm{E}[Y])(X - \mathrm{E}[X])$, where the expectation is taken under the bivariate joint distribution of Y and X:

$$\mathrm{Cov}[Y, X] = \mathrm{E}[(Y - \mathrm{E}[Y])(X - \mathrm{E}[X])] = \mathrm{E}[YX] - \mathrm{E}[Y]\mathrm{E}[X] = \int\int x\, y f(x, y)\ dx\, dy - \mathrm{E}[Y]\mathrm{E}[X]$$

The *covariance* between a random variable and itself is the variance, $\mathrm{Cov}[Y, Y] = \mathrm{Var}[Y]$.

In statistical applications and formulas, random variables are often collected into vectors. For example, a random sample of size n from the distribution of Y generates a random vector of order $(n \times 1)$,

$$\mathbf{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix}$$

The expected value of the $(n \times 1)$ random vector $\mathbf{Y}$ is the vector of the means of the elements of $\mathbf{Y}$:

$$\mathbf{E}[\mathbf{Y}] = [\mathrm{E}\,[Y_i]] = \begin{bmatrix} \mathrm{E}[Y_1] \\ \mathrm{E}[Y_2] \\ \vdots \\ \mathrm{E}[Y_n] \end{bmatrix}$$

It is often useful to directly apply rules about working with means, variances, and covariances of random vectors. To develop these rules, suppose that $\mathbf{Y}$ and $\mathbf{U}$ denote two random vectors with typical elements $Y_1, \ldots, Y_n$ and $U_1, \ldots, U_k$. Further suppose that $\mathbf{A}$ and $\mathbf{B}$ are constant (nonstochastic) matrices, that $\mathbf{a}$ is a constant vector, and that the c_i are scalar constants.

The following rules enable you to derive the mean of a linear function of a random vector:

$$\begin{aligned} \mathrm{E}[\mathbf{A}] &= \mathbf{A} \\ \mathrm{E}[\mathbf{AY} + \mathbf{a}] &= \mathbf{A}\mathrm{E}[\mathbf{Y}] + \mathbf{a} \\ \mathrm{E}[\mathbf{Y} + \mathbf{U}] &= \mathrm{E}[\mathbf{Y}] + \mathrm{E}[\mathbf{U}] \end{aligned}$$

The *covariance matrix* of $\mathbf{Y}$ and $\mathbf{U}$ is the $(n \times k)$ matrix whose typical element in row i, column j is the covariance between Y_i and U_j. The covariance matrix between two random vectors is frequently denoted

with the Cov "operator."

$$\begin{aligned}\mathrm{Cov}[\mathbf{Y},\mathbf{U}] &= \left[\mathrm{Cov}[Y_i,U_j]\right] \\ &= \mathrm{E}\left[(\mathbf{Y}-\mathrm{E}[\mathbf{Y}])\,(\mathbf{U}-\mathrm{E}[\mathbf{U}])'\right] = \mathrm{E}\left[\mathbf{Y}\mathbf{U}'\right] - \mathrm{E}[\mathbf{Y}]\mathrm{E}[\mathbf{U}]' \\ &= \begin{bmatrix} \mathrm{Cov}[Y_1,U_1] & \mathrm{Cov}[Y_1,U_2] & \mathrm{Cov}[Y_1,U_3] & \cdots & \mathrm{Cov}[Y_1,U_k] \\ \mathrm{Cov}[Y_2,U_1] & \mathrm{Cov}[Y_2,U_2] & \mathrm{Cov}[Y_2,U_3] & \cdots & \mathrm{Cov}[Y_2,U_k] \\ \mathrm{Cov}[Y_3,U_1] & \mathrm{Cov}[Y_3,U_2] & \mathrm{Cov}[Y_3,U_3] & \cdots & \mathrm{Cov}[Y_3,U_k] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}[Y_n,U_1] & \mathrm{Cov}[Y_n,U_2] & \mathrm{Cov}[Y_n,U_3] & \cdots & \mathrm{Cov}[Y_n,U_k] \end{bmatrix}\end{aligned}$$

The *variance matrix* of a random vector $\mathbf{Y}$ is the covariance matrix between $\mathbf{Y}$ and itself. The variance matrix is frequently denoted with the Var "operator."

$$\begin{aligned}\mathrm{Var}[\mathbf{Y}] &= \mathrm{Cov}[\mathbf{Y},\mathbf{Y}] = \left[\mathrm{Cov}[Y_i,Y_j]\right] \\ &= \mathrm{E}\left[(\mathbf{Y}-\mathrm{E}[\mathbf{Y}])\,(\mathbf{Y}-\mathrm{E}[\mathbf{Y}])'\right] = \mathrm{E}\left[\mathbf{Y}\mathbf{Y}'\right] - \mathrm{E}[\mathbf{Y}]\mathrm{E}[\mathbf{Y}]' \\ &= \begin{bmatrix} \mathrm{Cov}[Y_1,Y_1] & \mathrm{Cov}[Y_1,Y_2] & \mathrm{Cov}[Y_1,Y_3] & \cdots & \mathrm{Cov}[Y_1,Y_n] \\ \mathrm{Cov}[Y_2,Y_1] & \mathrm{Cov}[Y_2,Y_2] & \mathrm{Cov}[Y_2,Y_3] & \cdots & \mathrm{Cov}[Y_2,Y_n] \\ \mathrm{Cov}[Y_3,Y_1] & \mathrm{Cov}[Y_3,Y_2] & \mathrm{Cov}[Y_3,Y_3] & \cdots & \mathrm{Cov}[Y_3,Y_n] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}[Y_n,Y_1] & \mathrm{Cov}[Y_n,Y_2] & \mathrm{Cov}[Y_n,Y_3] & \cdots & \mathrm{Cov}[Y_n,Y_n] \end{bmatrix} \\ &= \begin{bmatrix} \mathrm{Var}[Y_1] & \mathrm{Cov}[Y_1,Y_2] & \mathrm{Cov}[Y_1,Y_3] & \cdots & \mathrm{Cov}[Y_1,Y_n] \\ \mathrm{Cov}[Y_2,Y_1] & \mathrm{Var}[Y_2] & \mathrm{Cov}[Y_2,Y_3] & \cdots & \mathrm{Cov}[Y_2,Y_n] \\ \mathrm{Cov}[Y_3,Y_1] & \mathrm{Cov}[Y_3,Y_2] & \mathrm{Var}[Y_3] & \cdots & \mathrm{Cov}[Y_3,Y_n] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}[Y_n,Y_1] & \mathrm{Cov}[Y_n,Y_2] & \mathrm{Cov}[Y_n,Y_3] & \cdots & \mathrm{Var}[Y_n] \end{bmatrix}\end{aligned}$$

Because the variance matrix contains variances on the diagonal and covariances in the off-diagonal positions, it is also referred to as the *variance-covariance matrix* of the random vector $\mathbf{Y}$.

If the elements of the covariance matrix $\mathrm{Cov}[\mathbf{Y},\mathbf{U}]$ are zero, the random vectors are uncorrelated. If $\mathbf{Y}$ and $\mathbf{U}$ are normally distributed, then a zero covariance matrix implies that the vectors are stochastically independent. If the off-diagonal elements of the variance matrix $\mathrm{Var}[\mathbf{Y}]$ are zero, the elements of the random vector $\mathbf{Y}$ are uncorrelated. If $\mathbf{Y}$ is normally distributed, then a diagonal variance matrix implies that its elements are stochastically independent.

Suppose that $\mathbf{A}$ and $\mathbf{B}$ are constant (nonstochastic) matrices and that c_i denotes a scalar constant. The following results are useful in manipulating covariance matrices:

$$\begin{aligned}\mathrm{Cov}[\mathbf{A}\mathbf{Y},\mathbf{U}] &= \mathbf{A}\mathrm{Cov}[\mathbf{Y},\mathbf{U}] \\ \mathrm{Cov}[\mathbf{Y},\mathbf{B}\mathbf{U}] &= \mathrm{Cov}[\mathbf{Y},\mathbf{U}]\mathbf{B}' \\ \mathrm{Cov}[\mathbf{A}\mathbf{Y},\mathbf{B}\mathbf{U}] &= \mathbf{A}\mathrm{Cov}[\mathbf{Y},\mathbf{U}]\mathbf{B}' \\ \mathrm{Cov}[c_1\mathbf{Y}_1 + c_2\mathbf{U}_1, c_3\mathbf{Y}_2 + c_4\mathbf{U}_2] &= c_1c_3\mathrm{Cov}[\mathbf{Y}_1,\mathbf{Y}_2] + c_1c_4\mathrm{Cov}[\mathbf{Y}_1,\mathbf{U}_2] \\ &\quad + c_2c_3\mathrm{Cov}[\mathbf{U}_1,\mathbf{Y}_2] + c_2c_4\mathrm{Cov}[\mathbf{U}_1,\mathbf{U}_2]\end{aligned}$$

Since $\mathrm{Cov}[\mathbf{Y},\mathbf{Y}] = \mathrm{Var}[\mathbf{Y}]$, these results can be applied to produce the following results, useful in manipulating variances of random vectors:

$$\begin{aligned}
\mathrm{Var}[\mathbf{A}] &= \mathbf{0} \\
\mathrm{Var}[\mathbf{AY}] &= \mathbf{A}\mathrm{Var}[\mathbf{Y}]\mathbf{A}' \\
\mathrm{Var}[\mathbf{Y}+\mathbf{x}] &= \mathrm{Var}[\mathbf{Y}] \\
\mathrm{Var}[\mathbf{x}'\mathbf{Y}] &= \mathbf{x}'\mathrm{Var}[\mathbf{Y}]\mathbf{x} \\
\mathrm{Var}[c_1\mathbf{Y}] &= c_1^2\mathrm{Var}[\mathbf{Y}] \\
\mathrm{Var}[c_1\mathbf{Y}+c_2\mathbf{U}] &= c_1^2\mathrm{Var}[\mathbf{Y}] + c_2^2\mathrm{Var}[\mathbf{U}] + 2c_1c_2\mathrm{Cov}[\mathbf{Y},\mathbf{U}]
\end{aligned}$$

Another area where expectation rules are helpful is quadratic forms in random variables. These forms arise particularly in the study of linear statistical models and in linear statistical inference. Linear inference is statistical inference about linear function of random variables, even if those random variables are defined through nonlinear models. For example, the parameter estimator $\widehat{\boldsymbol{\theta}}$ might be derived in a nonlinear model, but this does not prevent statistical questions from being raised that can be expressed through linear functions of $\boldsymbol{\theta}$; for example,

$$H_0\colon \begin{cases} \theta_1 - 2\theta_2 = 0 \\ \theta_2 - \theta_3 = 0 \end{cases}$$

if $\mathbf{A}$ is a matrix of constants and $\mathbf{Y}$ is a random vector, then

$$\mathrm{E}[\mathbf{Y}'\mathbf{AY}] = \mathrm{trace}(\mathbf{A}\mathrm{Var}[\mathbf{Y}]) + \mathrm{E}[\mathbf{Y}]'\mathbf{A}\mathrm{E}[\mathbf{Y}]$$

Mean Squared Error

The mean squared error is arguably the most important criterion used to evaluate the performance of a predictor or an estimator. (The subtle distinction between predictors and estimators is that random variables are predicted and constants are estimated.) The mean squared error is also useful to relay the concepts of bias, precision, and accuracy in statistical estimation. In order to examine a mean squared error, you need a target of estimation or prediction, and a predictor or estimator that is a function of the data. Suppose that the target, whether a constant or a random variable, is denoted as U. The mean squared error of the estimator or predictor $T(\mathbf{Y})$ for U is

$$\mathrm{MSE}\left[T(\mathbf{Y});U\right] = \mathrm{E}\left[(T(\mathbf{Y})-U)^2\right]$$

The reason for using a squared difference to measure the "loss" between $T(\mathbf{Y})$ and U is mostly convenience; properties of squared differences involving random variables are more easily examined than, say, absolute differences. The reason for taking an expectation is to remove the randomness of the squared difference by averaging over the distribution of the data.

Consider first the case where the target U is a constant—say, the parameter β—and denote the mean of the estimator $T(\mathbf{Y})$ as μ_T. The mean squared error can then be decomposed as

$$\begin{aligned}
\mathrm{MSE}[T(\mathbf{Y});\beta] &= \mathrm{E}\left[(T(\mathbf{Y})-\beta)^2\right] \\
&= \mathrm{E}\left[(T(\mathbf{Y})-\mu_T)^2\right] - \mathrm{E}\left[(\beta-\mu_T)^2\right] \\
&= \mathrm{Var}[T(\mathbf{Y})] + (\beta-\mu_T)^2
\end{aligned}$$

The mean squared error thus comprises the variance of the estimator and the squared bias. The two components can be associated with an estimator's precision (small variance) and its accuracy (small bias).

If $T(\mathbf{Y})$ is an unbiased estimator of β—that is, if $\mathrm{E}[T(\mathbf{Y})] = \beta$—then the mean squared error is simply the variance of the estimator. By choosing an estimator that has minimum variance, you also choose an estimator that has minimum mean squared error among all unbiased estimators. However, as you can see from the previous expression, bias is also an "average" property; it is defined as an expectation. It is quite possible to find estimators in some statistical modeling problems that have smaller mean squared error than a minimum variance unbiased estimator; these are estimators that permit a certain amount of bias but improve on the variance. For example, in models where regressors are highly collinear, the ordinary least squares estimator continues to be unbiased. However, the presence of collinearity can induce poor precision and lead to an erratic estimator. Ridge regression stabilizes the regression estimates in this situation, and the coefficient estimates are somewhat biased, but the bias is more than offset by the gains in precision.

When the target U is a random variable, you need to carefully define what an unbiased prediction means. If the statistic and the target have the same expectation, $\mathrm{E}[U] = \mathrm{E}[T(\mathbf{Y})]$, then

$$\mathrm{MSE}\left[T(\mathbf{Y}); U\right] = \mathrm{Var}[T(\mathbf{Y})] + \mathrm{Var}[U] - 2\mathrm{Cov}[T(\mathbf{Y}), U]$$

In many instances the target U is a new observation that was not part of the analysis. If the data are uncorrelated, then it is reasonable to assume in that instance that the new observation is also not correlated with the data. The mean squared error then reduces to the sum of the two variances. For example, in a linear regression model where U is a new observation Y_0 and $T(\mathbf{Y})$ is the regression estimator

$$\widehat{Y}_0 = \mathbf{x}_0' \left(\mathbf{X}'\mathbf{X}\right)^{-1} \mathbf{X}'\mathbf{Y}$$

with variance $\mathrm{Var}[Y_0] = \sigma^2 \mathbf{x}_0' \left(\mathbf{X}'\mathbf{X}\right)^{-1} \mathbf{x}_0$, the mean squared prediction error for Y_0 is

$$\mathrm{MSE}\left[\widehat{Y}; Y_0\right] = \sigma^2 \left(\mathbf{x}_0' \left(\mathbf{X}'\mathbf{X}\right)^{-1} \mathbf{x}_0 + 1\right)$$

and the mean squared prediction error for predicting the mean $\mathrm{E}[Y_0]$ is

$$\mathrm{MSE}\left[\widehat{Y}; \mathrm{E}[Y_0]\right] = \sigma^2 \mathbf{x}_0' \left(\mathbf{X}'\mathbf{X}\right)^{-1} \mathbf{x}_0$$

Linear Model Theory

This section presents some basic statistical concepts and results for the linear model with homoscedastic, uncorrelated errors in which the parameters are estimated by ordinary least squares. The model can be written as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \qquad \boldsymbol{\epsilon} \sim (0, \sigma^2 \mathbf{I})$$

where $\mathbf{Y}$ is an $(n \times 1)$ vector and $\mathbf{X}$ is an $(n \times k)$ matrix of known constants. The model equation implies the following expected values:

$$\mathrm{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{\beta}$$

$$\mathrm{Var}[\mathbf{Y}] = \sigma^2 \mathbf{I} \Leftrightarrow \mathrm{Cov}[Y_i, Y_j] = \begin{cases} \sigma^2 & i = j \\ 0 & \text{otherwise} \end{cases}$$

Finding the Least Squares Estimators

Finding the least squares estimator of $\boldsymbol{\beta}$ can be motivated as a calculus problem or by considering the geometry of least squares. The former approach simply states that the OLS estimator is the vector $\widehat{\boldsymbol{\beta}}$ that minimizes the objective function

$$\mathrm{SSE} = (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})$$

Applying the differentiation rules from the section "Matrix Differentiation" on page 48 leads to

$$\begin{aligned}\frac{\partial}{\partial\boldsymbol{\beta}}\mathrm{SSE} &= \frac{\partial}{\partial\boldsymbol{\beta}}(\mathbf{Y}'\mathbf{Y} - 2\mathbf{Y}'\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta}) \\ &= \mathbf{0} - 2\mathbf{X}'\mathbf{Y} + 2\mathbf{X}'\mathbf{X}\boldsymbol{\beta} \\ \frac{\partial^2}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}}\mathrm{SSE} &= \mathbf{X}'\mathbf{X}\end{aligned}$$

Consequently, the solution to the *normal equations*, $\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \mathbf{X}'\mathbf{Y}$, solves $\frac{\partial}{\partial\boldsymbol{\beta}}\mathrm{SSE} = 0$, and the fact that the second derivative is nonnegative definite guarantees that this solution minimizes SSE. The geometric argument to motivate ordinary least squares estimation is as follows. Assume that $\mathbf{X}$ is of rank k. For any value of $\boldsymbol{\beta}$, such as $\tilde{\boldsymbol{\beta}}$, the following identity holds:

$$\mathbf{Y} = \mathbf{X}\tilde{\boldsymbol{\beta}} + \left(\mathbf{Y} - \mathbf{X}\tilde{\boldsymbol{\beta}}\right)$$

The vector $\mathbf{X}\tilde{\boldsymbol{\beta}}$ is a point in a k-dimensional subspace of R^n, and the residual $(\mathbf{Y} - \mathbf{X}\tilde{\boldsymbol{\beta}})$ is a point in an $(n-k)$-dimensional subspace. The OLS estimator is the value $\widehat{\boldsymbol{\beta}}$ that minimizes the distance of $\mathbf{X}\tilde{\boldsymbol{\beta}}$ from $\mathbf{Y}$, implying that $\mathbf{X}\widehat{\boldsymbol{\beta}}$ and $(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})$ are orthogonal to each other; that is,

$(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'\mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{0}$. This in turn implies that $\widehat{\boldsymbol{\beta}}$ satisfies the normal equations, since

$$\widehat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{Y} = \widehat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{X}\widehat{\boldsymbol{\beta}} \Leftrightarrow \mathbf{X}'\mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}'\mathbf{Y}$$

Full-Rank Case

If $\mathbf{X}$ is of full column rank, the OLS estimator is unique and given by

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}$$

The OLS estimator is an unbiased estimator of $\boldsymbol{\beta}$—that is,

$$\begin{aligned}\mathrm{E}[\widehat{\boldsymbol{\beta}}] &= \mathrm{E}\left[\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}\right] \\ &= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathrm{E}[\mathbf{Y}] = \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}\end{aligned}$$

Note that this result holds if $\mathrm{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{\beta}$; in other words, the condition that the model errors have mean zero is sufficient for the OLS estimator to be unbiased. If the errors are homoscedastic and uncorrelated, the OLS estimator is indeed the *best* linear unbiased estimator (BLUE) of $\boldsymbol{\beta}$—that is, no other estimator that is a linear function of $\mathbf{Y}$ has a smaller mean squared error. The fact that the estimator is unbiased implies that no other linear estimator has a smaller variance. If, furthermore, the model errors are normally distributed, then the OLS estimator has minimum variance among all unbiased estimators of $\boldsymbol{\beta}$, whether they are linear or not. Such an estimator is called a *uniformly minimum variance unbiased estimator*, or UMVUE.

Rank-Deficient Case

In the case of a rank-deficient $\mathbf{X}$ matrix, a generalized inverse is used to solve the normal equations:

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{Y}$$

Although a g_1-inverse is sufficient to solve a linear system, computational expedience and interpretation of the results often dictate the use of a generalized inverse with reflexive properties (that is, a g_2-inverse; see the section "Generalized Inverse Matrices" on page 47 for details). Suppose, for example, that the $\mathbf{X}$ matrix is partitioned as $\mathbf{X} = [\mathbf{X}_1 \;\; \mathbf{X}_2]$, where $\mathbf{X}_1$ is of full column rank and each column in $\mathbf{X}_2$ is a linear combination of the columns of $\mathbf{X}_1$. The matrix

$$\mathbf{G}_1 = \begin{bmatrix} \left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1} & \left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\mathbf{X}_2 \\ -\mathbf{X}_2'\mathbf{X}_1\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1} & \mathbf{0} \end{bmatrix}$$

is a g_1-inverse of $\mathbf{X}'\mathbf{X}$ and

$$\mathbf{G}_2 = \begin{bmatrix} \left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$$

is a g_2-inverse. If the least squares solution is computed with the g_1-inverse, then computing the variance of the estimator requires additional matrix operations and storage. On the other hand, the variance of the solution that uses a g_2-inverse is proportional to G_2.

$$\begin{aligned} \mathrm{Var}\left[\mathbf{G}_1\mathbf{X}'\mathbf{Y}\right] &= \sigma^2\mathbf{G}_1\mathbf{X}'\mathbf{X}\mathbf{G}_1 \\ \mathrm{Var}\left[\mathbf{G}_2\mathbf{X}'\mathbf{Y}\right] &= \sigma^2\mathbf{G}_2\mathbf{X}'\mathbf{X}\mathbf{G}_2 = \sigma^2\mathbf{G}_2 \end{aligned}$$

If a generalized inverse $\mathbf{G}$ of $\mathbf{X}'\mathbf{X}$ is used to solve the normal equations, then the resulting solution is a biased estimator of $\boldsymbol{\beta}$ (unless $\mathbf{X}'\mathbf{X}$ is of full rank, in which case the generalized inverse is "the" inverse), since $\mathrm{E}\left[\widehat{\boldsymbol{\beta}}\right] = \mathbf{G}\mathbf{X}'\mathbf{X}\boldsymbol{\beta}$, which is not in general equal to $\boldsymbol{\beta}$.

If you think of estimation as "estimation without bias," then $\widehat{\boldsymbol{\beta}}$ is the estimator of something, namely $\mathbf{G}\mathbf{X}\boldsymbol{\beta}$. Since this is not a quantity of interest and since it is not unique—it depends on your choice of $\mathbf{G}$—Searle (1971, p. 169) cautions that in the less-than-full-rank case, $\widehat{\boldsymbol{\beta}}$ is a solution to the normal equations and "nothing more."

Analysis of Variance

The identity

$$\mathbf{Y} = \mathbf{X}\tilde{\boldsymbol{\beta}} + \left(\mathbf{Y} - \mathbf{X}\tilde{\boldsymbol{\beta}}\right)$$

holds for all vectors $\tilde{\boldsymbol{\beta}}$, but only for the least squares solution is the residual $(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})$ orthogonal to the predicted value $\mathbf{X}\widehat{\boldsymbol{\beta}}$. Because of this orthogonality, the additive identity holds not only for the vectors themselves, but also for their lengths (Pythagorean theorem):

$$||\mathbf{Y}||^2 = ||\mathbf{X}\widehat{\boldsymbol{\beta}}||^2 + ||(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}})||^2$$

Note that $\mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y} = \mathbf{H}\mathbf{Y}$ and note that $\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}} = (\mathbf{I} - \mathbf{H})\mathbf{Y} = \mathbf{M}\mathbf{Y}$. The matrices $\mathbf{H}$ and $\mathbf{M} = \mathbf{I} - \mathbf{H}$ play an important role in the theory of linear models and in statistical computations. Both are

projection matrices—that is, they are symmetric and idempotent. (An idempotent matrix $\mathbf{A}$ is a square matrix that satisfies $\mathbf{AA} = \mathbf{A}$. The eigenvalues of an idempotent matrix take on the values 1 and 0 only.) The matrix $\mathbf{H}$ projects onto the subspace of R^n that is spanned by the columns of $\mathbf{X}$. The matrix $\mathbf{M}$ projects onto the orthogonal complement of that space. Because of these properties you have $\mathbf{H}' = \mathbf{H}$, $\mathbf{HH} = \mathbf{H}$, $\mathbf{M}' = \mathbf{M}$, $\mathbf{MM} = \mathbf{M}$, $\mathbf{HM} = \mathbf{0}$.

The Pythagorean relationship now can be written in terms of $\mathbf{H}$ and $\mathbf{M}$ as follows:

$$||\mathbf{Y}||^2 = \mathbf{Y}'\mathbf{Y} = ||\mathbf{HY}||^2 + ||\mathbf{MY}||^2 = \mathbf{Y}'\mathbf{H}'\mathbf{HY} + \mathbf{Y}'\mathbf{M}'\mathbf{MY} = \mathbf{Y}'\mathbf{HY} + \mathbf{Y}'\mathbf{MY}$$

If $\mathbf{X}'\mathbf{X}$ is deficient in rank and a generalized inverse is used to solve the normal equations, then you work instead with the projection matrices $\mathbf{H} = \mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'$. Note that if $\mathbf{G}$ is a generalized inverse of $\mathbf{X}'\mathbf{X}$, then $\mathbf{XGX}'$, and hence also $\mathbf{H}$ and $\mathbf{M}$, are invariant to the choice of $\mathbf{G}$.

The matrix $\mathbf{H}$ is sometimes referred to as the "hat" matrix because when you premultiply the vector of observations with $\mathbf{H}$, you produce the fitted values, which are commonly denoted by placing a "hat" over the $\mathbf{Y}$ vector, $\widehat{\mathbf{Y}} = \mathbf{HY}$.

The term $\mathbf{Y}'\mathbf{Y}$ is the uncorrected total sum of squares (SST) of the linear model, $\mathbf{Y}'\mathbf{MY}$ is the error (residual) sum of squares (SSR), and $\mathbf{Y}'\mathbf{HY}$ is the uncorrected model sum of squares. This leads to the analysis of variance table shown in Table 3.2.

Table 3.2 Analysis of Variance with Uncorrected Sums of Squares

Source	df	Sum of Squares
Model	$\mathrm{rank}(\mathbf{X})$	$\mathrm{SSM} = \mathbf{Y}'\mathbf{HY} = \widehat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{Y}$
Residual	$n - \mathrm{rank}(\mathbf{X})$	$\mathrm{SSR} = \mathbf{Y}'\mathbf{MY} = \mathbf{Y}'\mathbf{Y} - \widehat{\boldsymbol{\beta}}\mathbf{X}'\mathbf{Y} = \sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2$
Uncorr. Total	n	$\mathrm{SST} = \mathbf{Y}'\mathbf{Y} = \sum_{i=1}^{n} Y_i^2$

When the model contains an intercept term, then the analysis of variance is usually corrected for the mean, as shown in Table 3.3.

Table 3.3 Analysis of Variance with Corrected Sums of Squares

Source	df	Sum of Squares
Model	$\mathrm{rank}(\mathbf{X}) - 1$	$\mathrm{SSM}_c = \widehat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{Y} - n\overline{Y}^2 = \sum_{i=1}^{n}\left(\widehat{Y}_i - \overline{Y}_i\right)^2$
Residual	$n - \mathrm{rank}(\mathbf{X})$	$\mathrm{SSR} = \mathbf{Y}'\mathbf{MY} = \mathbf{Y}'\mathbf{Y} - \widehat{\boldsymbol{\beta}}\mathbf{X}'\mathbf{Y} = \sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2$
Corrected Total	$n - 1$	$\mathrm{SST}_c = \mathbf{Y}'\mathbf{Y} - n\overline{Y}^2 = \sum_{i=1}^{n}\left(Y_i - \overline{Y}\right)^2$

The *coefficient of determination*, also called the R-square statistic, measures the proportion of the total

variation explained by the linear model. In models with intercept, it is defined as the ratio

$$R^2 = 1 - \frac{\mathrm{SSR}}{\mathrm{SST}_c} = 1 - \frac{\sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2}{\sum_{i=1}^{n}\left(Y_i - \overline{Y}\right)^2}$$

In models without intercept, the R-square statistic is a ratio of the uncorrected sums of squares

$$R^2 = 1 - \frac{\mathrm{SSR}}{\mathrm{SST}} = 1 - \frac{\sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2}{\sum_{i=1}^{n} Y_i^2}$$

Estimating the Error Variance

The least squares principle does not provide for a parameter estimator for σ^2. The usual approach is to use a method-of-moments estimator that is based on the sum of squared residuals. If the model is correct, then the mean square for error, defined to be SSR divided by its degrees of freedom,

$$\begin{aligned}\widehat{\sigma}^2 &= \frac{1}{n - \mathrm{rank}(\mathbf{X})}\left(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\right)'\left(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\right)\\ &= \mathrm{SSR}/(n - \mathrm{rank}(\mathbf{X}))\end{aligned}$$

is an unbiased estimator of σ^2.

Maximum Likelihood Estimation

To estimate the parameters in a linear model with mean function $\mathrm{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{\beta}$ by maximum likelihood, you need to specify the distribution of the response vector $\mathbf{Y}$. In the linear model with a continuous response variable, it is commonly assumed that the response is normally distributed. In that case, the estimation problem is completely defined by specifying the mean and variance of $\mathbf{Y}$ in addition to the normality assumption. The model can be written as $\mathbf{Y} \sim N(\mathbf{X}\boldsymbol{\beta}, \sigma^2\mathbf{I})$, where the notation $N(\mathbf{a}, \mathbf{V})$ indicates a multivariate normal distribution with mean vector $\mathbf{a}$ and variance matrix $\mathbf{V}$. The log likelihood for $\mathbf{Y}$ then can be written as

$$l(\boldsymbol{\beta}, \sigma^2; \mathbf{y}) = -\frac{n}{2}\log\{2\pi\} - \frac{n}{2}\log\{\sigma^2\} - \frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

This function is maximized in $\boldsymbol{\beta}$ when the sum of squares $(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$ is minimized. The maximum likelihood estimator of $\boldsymbol{\beta}$ is thus identical to the ordinary least squares estimator. To maximize $l(\boldsymbol{\beta}, \sigma^2; \mathbf{y})$ with respect to σ^2, note that

$$\frac{\partial l(\boldsymbol{\beta}, \sigma^2; \mathbf{y})}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

Hence the MLE of σ^2 is the estimator

$$\begin{aligned}\widehat{\sigma}^2_M &= \frac{1}{n}\left(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\right)'\left(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\right)\\ &= \mathrm{SSR}/n\end{aligned}$$

This is a biased estimator of σ^2, with a bias that decreases with n.

Estimable Functions

A function $\mathbf{L}\boldsymbol{\beta}$ is said to be estimable if there exists a linear combination of the expected value of $\mathbf{Y}$, such as $\mathbf{K}\mathrm{E}[\mathbf{Y}]$, that equals $\mathbf{L}\boldsymbol{\beta}$. Since $\mathrm{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{\beta}$, the definition of estimability implies that $\mathbf{L}\boldsymbol{\beta}$ is estimable if there is a matrix $\mathbf{K}$ such that $\mathbf{L} = \mathbf{KX}$. Another way of looking at this result is that the rows of $\mathbf{X}$ form a generating set from which all estimable functions can be constructed.

The concept of estimability of functions is important in the theory and application of linear models because hypotheses of interest are often expressed as linear combinations of the parameter estimates (for example, hypotheses of equality between parameters, $\beta_1 = \beta_2 \Leftrightarrow \beta_1 - \beta_2 = 0$). Since estimability is not related to the particular value of the parameter estimate, but to the row space of $\mathbf{X}$, you can test only hypotheses that consist of estimable functions. Further, because estimability is not related to the value of $\boldsymbol{\beta}$ (Searle 1971, p. 181), the choice of the generalized inverse in a situation with rank-deficient $\mathbf{X}'\mathbf{X}$ matrix is immaterial, since

$$\mathbf{L}\widehat{\boldsymbol{\beta}} = \mathbf{KX}\widehat{\boldsymbol{\beta}} = \mathbf{KX}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{Y}$$

where $\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}$ is invariant to the choice of generalized inverse.

$\mathbf{L}\boldsymbol{\beta}$ is estimable if and only if $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}(\mathbf{X}'\mathbf{X}) = \mathbf{L}$ (see, for example, Searle 1971, p. 185). If $\mathbf{X}$ is of full rank, then the *Hermite* matrix $(\mathbf{X}'\mathbf{X})^{-}(\mathbf{X}'\mathbf{X})$ is the identity, which implies that all linear functions are estimable in the full-rank case.

See Chapter 16, "The Four Types of Estimable Functions," for many details about the various forms of estimable functions in SAS/STAT.

Test of Hypotheses

Consider a general linear hypothesis of the form $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{d}$, where $\mathbf{L}$ is a $(k \times p)$ matrix. It is assumed that $\mathbf{d}$ is such that this hypothesis is linearly consistent—that is, that there exists *some* $\boldsymbol{\beta}$ for which $\mathbf{L}\boldsymbol{\beta} = \mathbf{d}$. This is always the case if $\mathbf{d}$ is in the column space of $\mathbf{L}$, if $\mathbf{L}$ has full row rank, or if $\mathbf{d} = \mathbf{0}$; the latter is the most common case. Since many linear models have a rank-deficient $\mathbf{X}$ matrix, the question arises whether the hypothesis is testable. The idea of testability of a hypothesis is—not surprisingly—connected to the concept of estimability as introduced previously. The hypothesis $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{d}$ is testable if it consists of estimable functions.

There are two important approaches to testing hypotheses in statistical applications—the reduction principle and the linear inference approach. The reduction principle states that the validity of the hypothesis can be inferred by comparing a suitably chosen summary statistic between the model at hand and a reduced model in which the constraint $\mathbf{L}\boldsymbol{\beta} = \mathbf{d}$ is imposed. The linear inference approach relies on the fact that $\widehat{\boldsymbol{\beta}}$ is an estimator of $\boldsymbol{\beta}$ and its stochastic properties are known, at least approximately. A test statistic can then be formed using $\widehat{\boldsymbol{\beta}}$, and its behavior under the restriction $\mathbf{L}\boldsymbol{\beta} = \mathbf{d}$ can be ascertained.

The two principles lead to identical results in certain—for example, least squares estimation in the classical linear model. In more complex situations the two approaches lead to similar but not identical results. This is the case, for example, when weights or unequal variances are involved, or when $\widehat{\boldsymbol{\beta}}$ is a nonlinear estimator.

Reduction Tests

The two main reduction principles are the sum of squares reduction test and the likelihood ratio test. The test statistic in the former is proportional to the difference of the residual sum of squares between the reduced model and the full model. The test statistic in the likelihood ratio test is proportional to the difference of the log likelihoods between the full and reduced models. To fix these ideas, suppose that you are fitting the

model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2\mathbf{I})$. Suppose that SSR denotes the residual sum of squares in this model and that SSR_H is the residual sum of squares in the model for which $\mathbf{L}\boldsymbol{\beta} = \mathbf{d}$ holds. Then under the hypothesis the ratio

$$(\mathrm{SSR}_H - \mathrm{SSR})/\sigma^2$$

follows a chi-square distribution with degrees of freedom equal to the rank of $\mathbf{L}$. Maybe surprisingly, the residual sum of squares in the full model is distributed independently of this quantity, so that under the hypothesis,

$$F = \frac{(\mathrm{SSR}_H - \mathrm{SSR})/\mathrm{rank}(\mathbf{L})}{\mathrm{SSR}/(n - \mathrm{rank}(\mathbf{X}))}$$

follows an F distribution with $\mathrm{rank}(\mathbf{L})$ numerator and $n - \mathrm{rank}(\mathbf{X})$ denominator degrees of freedom. Note that the quantity in the denominator of the F statistic is a particular estimator of σ^2—namely, the unbiased moment-based estimator that is customarily associated with least squares estimation. It is also the restricted maximum likelihood estimator of σ^2 if $\mathbf{Y}$ is normally distributed.

In the case of the likelihood ratio test, suppose that $l(\widehat{\boldsymbol{\beta}}, \widehat{\sigma}^2; \mathbf{y})$ denotes the log likelihood evaluated at the ML estimators. Also suppose that $l(\widehat{\boldsymbol{\beta}}_H, \widehat{\sigma}^2_H; \mathbf{y})$ denotes the log likelihood in the model for which $\mathbf{L}\boldsymbol{\beta} = \mathbf{d}$ holds. Then under the hypothesis the statistic

$$\lambda = 2\left(l(\widehat{\boldsymbol{\beta}}, \widehat{\sigma}^2; \mathbf{y}) - l(\widehat{\boldsymbol{\beta}}_H, \widehat{\sigma}^2_H; \mathbf{y})\right)$$

follows approximately a chi-square distribution with degrees of freedom equal to the rank of $\mathbf{L}$. In the case of a normally distributed response, the log-likelihood function can be profiled with respect to $\boldsymbol{\beta}$. The resulting profile log likelihood is

$$l(\widehat{\sigma}^2; \mathbf{y}) = -\frac{n}{2}\log\{2\pi\} - \frac{n}{2}\left(\log\{\widehat{\sigma}^2\}\right)$$

and the likelihood ratio test statistic becomes

$$\lambda = n\left(\log\left\{\widehat{\sigma}^2_H\right\} - \log\left\{\widehat{\sigma}^2\right\}\right) = n\left(\log\{\mathrm{SSR}_H\} - \log\{\mathrm{SSR}\}\right) = n\left(\log\{\mathrm{SSR}_H/\mathrm{SSR}\}\right)$$

The preceding expressions show that, in the case of normally distributed data, both reduction principles lead to simple functions of the residual sums of squares in two models. As Pawitan (2001, p. 151) puts it, there is, however, an important difference not in the computations but in the statistical content. The least squares principle, where sum of squares reduction tests are widely used, does not require a distributional specification. Assumptions about the distribution of the data are added to provide a framework for confirmatory inferences, such as the testing of hypotheses. This framework stems directly from the assumption about the data's distribution, or from the sampling distribution of the least squares estimators. The likelihood principle, on the other hand, requires a distributional specification at the outset. Inference about the parameters is implicit in the model; it is the result of further *computations* following the estimation of the parameters. In the least squares framework, inference about the parameters is the result of further *assumptions*.

Linear Inference

The principle of linear inference is to formulate a test statistic for $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{d}$ that builds on the linearity of the hypothesis about $\boldsymbol{\beta}$. For many models that have linear components, the estimator $\mathbf{L}\widehat{\boldsymbol{\beta}}$ is also linear in $\mathbf{Y}$. It is then simple to establish the distributional properties of $\mathbf{L}\widehat{\boldsymbol{\beta}}$ based on the distributional assumptions about

$\mathbf{Y}$ or based on large-sample arguments. For example, $\widehat{\boldsymbol{\beta}}$ might be a nonlinear estimator, but it is known to asymptotically follow a normal distribution; this is the case in many nonlinear and generalized linear models.

If the sampling distribution or the asymptotic distribution of $\widehat{\boldsymbol{\beta}}$ is normal, then one can easily derive quadratic forms with known distributional properties. For example, if the random vector $\mathbf{U}$ is distributed as $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then $\mathbf{U}'\mathbf{A}\mathbf{U}$ follows a chi-square distribution with $\mathrm{rank}(\mathbf{A})$ degrees of freedom and noncentrality parameter $1/2\boldsymbol{\mu}'\mathbf{A}\boldsymbol{\mu}$, provided that $\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}\boldsymbol{\Sigma} = \mathbf{A}\boldsymbol{\Sigma}$.

In the classical linear model, suppose that $\mathbf{X}$ is deficient in rank and that $\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{Y}$ is a solution to the normal equations. Then, if the errors are normally distributed,

$$\widehat{\boldsymbol{\beta}} \sim N\left(\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{X}\boldsymbol{\beta}, \sigma^2\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-}\right)$$

Because $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{d}$ is testable, $\mathbf{L}\boldsymbol{\beta}$ is estimable, and thus $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X} = \mathbf{L}$, as established in the previous section. Hence,

$$\mathbf{L}\widehat{\boldsymbol{\beta}} \sim N\left(\mathbf{L}\boldsymbol{\beta}, \sigma^2\mathbf{L}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{L}'\right)$$

The conditions for a chi-square distribution of the quadratic form

$$(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d})'\left(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'\right)^{-}(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d})$$

are thus met, provided that

$$(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-}\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-}\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}' = (\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-}\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'$$

This condition is obviously met if $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'$ is of full rank. The condition is also met if $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'^{-}$ is a reflexive inverse (a g_2-inverse) of $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}$.

The test statistic to test the linear hypothesis $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{d}$ is thus

$$F = \frac{(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d})'\left(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'\right)^{-}(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d})/\mathrm{rank}(\mathbf{L})}{\mathrm{SSR}/(n - \mathrm{rank}(\mathbf{X}))}$$

and it follows an F distribution with $\mathrm{rank}(\mathbf{L})$ numerator and $n - \mathrm{rank}(\mathbf{X})$ denominator degrees of freedom under the hypothesis.

This test statistic looks very similar to the F statistic for the sum of squares reduction test. This is no accident. If the model is linear and parameters are estimated by ordinary least squares, then you can show that the quadratic form $(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d})'\left(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'\right)^{-}(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d})$ equals the differences in the residual sum of squares, $\mathrm{SSR}_H - \mathrm{SSR}$, where SSR_H is obtained as the residual sum of squares from OLS estimation in a model that satisfies $\mathbf{L}\boldsymbol{\beta} = \mathbf{d}$. However, this correspondence between the two test formulations does not apply when a different estimation principle is used. For example, assume that $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \mathbf{V})$ and that $\boldsymbol{\beta}$ is estimated by generalized least squares:

$$\widehat{\boldsymbol{\beta}}_g = \left(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{Y}$$

The construction of $\mathbf{L}$ matrices associated with hypotheses in SAS/STAT software is frequently based on the properties of the $\mathbf{X}$ matrix, not of $\mathbf{X}'\mathbf{V}^{-}\mathbf{X}$. In other words, the construction of the $\mathbf{L}$ matrix is governed only by the design. A sum of squares reduction test for $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ that uses the generalized residual sum of squares $(\mathbf{Y} - \widehat{\boldsymbol{\beta}}_g)'\mathbf{V}^{-1}(\mathbf{Y} - \widehat{\boldsymbol{\beta}}_g)$ is not identical to a linear hypothesis test with the statistic

$$F^* = \frac{\widehat{\boldsymbol{\beta}}_g'\mathbf{L}'\left(\mathbf{L}\left(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right)^{-}\mathbf{L}'\right)^{-}\mathbf{L}\widehat{\boldsymbol{\beta}}_g}{\mathrm{rank}(\mathbf{L})}$$

Furthermore, $\mathbf{V}$ is usually unknown and must be estimated as well. The estimate for $\mathbf{V}$ depends on the model, and imposing a constraint on the model would change the estimate. The asymptotic distribution of the statistic F^* is a chi-square distribution. However, in practical applications the F distribution with $\mathrm{rank}(\mathbf{L})$ numerator and ν denominator degrees of freedom is often used because it provides a better approximation to the sampling distribution of F^* in finite samples. The computation of the denominator degrees of freedom ν, however, is a matter of considerable discussion. A number of methods have been proposed and are implemented in various forms in SAS/STAT (see, for example, the degrees-of-freedom methods in the MIXED and GLIMMIX procedures).

The general form of an F statistic for testing a linear hypotheses $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ is

$$F = \hat{\boldsymbol{\beta}}'\mathbf{L}'\left(\mathbf{L}\left(\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\right)\mathbf{L}'\right)^{-}\mathbf{L}\hat{\boldsymbol{\beta}}/\mu$$

where $\mu = \mathrm{rank}\left(\mathbf{L}\left(\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\right)\mathbf{L}'\right)$. The preceding development assumes that the estimated variance of $\hat{\boldsymbol{\beta}}$, $\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})$, is of the form $\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}$ for some estimate $\mathbf{V}$ of the variance of $\mathbf{Y}$. In this case, estimability ensures that this F statistic has a unique value no matter which kind of generalized inverse is used to compute it. However, when a residual-based sandwich estimator is used to estimate $\mathbf{V}$, estimability does not ensure uniqueness. In this case, the F value is invariant to the choice of the generalized inverse if and only if $\mathbf{L}$ is estimable and $\mathbf{L}'\left[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{L}'\right]^{-}\left[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{L}'\right] = \mathbf{L}'$.

Although it is extremely rare, it is possible in practice that the preceding uniqueness condition is not satisfied. For example, if the number of subjects (clusters) is less than the number of nonsingular parameters in the model and a residual-based sandwich variance estimator, $\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})$, is used to estimate $\mathbf{V}(\hat{\boldsymbol{\beta}})$, then the matrix of coefficients for testing the overall null does not satisfy the uniqueness condition. If this condition is not satisfied, then the F statistic for testing $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ is not invariant to the choice of the g_2-inverse of $\mathbf{L}\left(\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\right)\mathbf{L}'$. In practical applications, F is compared with an F distribution with μ numerator and ν denominator degrees of freedom, where ν depends on how the variance of $\hat{\boldsymbol{\beta}}$ is estimated. But the F value and therefore the inference might be different when a different g_2-inverse is used. This F test is not recommended when the uniqueness condition is not satisfied. An alternative approach would be to increase the number of subjects (clusters) or to find a parsimonious model so that the number of parameters is less than the number of subjects (clusters).

Residual Analysis

The model errors $\boldsymbol{\epsilon} = \mathbf{Y} - \mathbf{X}\boldsymbol{\beta}$ are unobservable. Yet important features of the statistical model are connected to them, such as the distribution of the data, the correlation among observations, and the constancy of variance. It is customary to diagnose and investigate features of the model errors through the fitted residuals $\widehat{\boldsymbol{\epsilon}} = \mathbf{Y} - \widehat{\mathbf{Y}} = \mathbf{Y} - \mathbf{HY} = \mathbf{MY}$. These residuals are projections of the data onto the null space of $\mathbf{X}$ and are also referred to as the "raw" residuals to contrast them with other forms of residuals that are transformations of $\widehat{\boldsymbol{\epsilon}}$. For the classical linear model, the statistical properties of $\widehat{\boldsymbol{\epsilon}}$ are affected by the features of that projection and can be summarized as follows:

$$\begin{aligned} \mathrm{E}[\widehat{\boldsymbol{\epsilon}}] &= \mathbf{0} \\ \mathrm{Var}[\widehat{\boldsymbol{\epsilon}}] &= \sigma^2\mathbf{M} \\ \mathrm{rank}(\mathbf{M}) &= n - \mathrm{rank}(\mathbf{X}) \end{aligned}$$

Furthermore, if $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2\mathbf{I})$, then $\widehat{\boldsymbol{\epsilon}} \sim N(\mathbf{0}, \sigma^2\mathbf{M})$.

Because $\mathbf{M} = \mathbf{I} - \mathbf{H}$, and the "hat" matrix $\mathbf{H}$ satisfies $\partial\widehat{\mathbf{Y}}/\partial\mathbf{Y}$, the hat matrix is also the leverage matrix of the model. If h_{ii} denotes the ith diagonal element of $\mathbf{H}$ (the leverage of observation i), then the leverages are bounded in a model with intercept, $1/n \leq h_{ii} \leq 1$. Consequently, the variance of a raw residual is less than that of an observation: $\mathrm{Var}[\widehat{\epsilon}_i] = \sigma^2(1 - h_{ii}) < \sigma^2$. In applications where the variability of the data is estimated from fitted residuals, the estimate is invariably biased low. An example is the computation of an empirical semivariogram based on fitted (detrended) residuals.

More important, the diagonal entries of $\mathbf{H}$ are not necessarily identical; the residuals are heteroscedastic. The "hat" matrix is also not a diagonal matrix; the residuals are correlated. In summary, the only property that the fitted residuals $\widehat{\boldsymbol{\epsilon}}$ share with the model errors is a zero mean. It is thus commonplace to use transformations of the fitted residuals for diagnostic purposes.

Raw and Studentized Residuals

A *standardized* residual is a raw residual that is divided by its standard deviation:

$$\widehat{\epsilon}_i^* = \frac{Y_i - \widehat{Y}_i}{\sqrt{\mathrm{Var}[Y_i - \widehat{Y}_i]}} = \frac{\widehat{\epsilon}_i}{\sqrt{\sigma^2(1 - h_{ii})}}$$

Because σ^2 is unknown, residual standardization is usually not practical. A *studentized* residual is a raw residual that is divided by its estimated standard deviation. If the estimate of the standard deviation is based on the same data that were used in fitting the model, the residual is also called an *internally studentized* residual:

$$\widehat{\epsilon}_{is} = \frac{Y_i - \widehat{Y}_i}{\sqrt{\widehat{\mathrm{Var}}[Y_i - \widehat{Y}_i]}} = \frac{\widehat{\epsilon}_i}{\sqrt{\widehat{\sigma}^2(1 - h_{ii})}}$$

If the estimate of the residual's variance does not involve the ith observation, it is called an *externally studentized* residual. Suppose that $\widehat{\sigma}^2_{-i}$ denotes the estimate of the residual variance obtained without the ith observation; then the externally studentized residual is

$$\widehat{\epsilon}_{ir} = \frac{\widehat{\epsilon}_i}{\sqrt{\widehat{\sigma}^2_{-i}(1 - h_{ii})}}$$

Scaled Residuals

A scaled residual is simply a raw residual divided by a scalar quantity that is not an estimate of the variance of the residual. For example, residuals divided by the standard deviation of the response variable are scaled and referred to as Pearson or Pearson-type residuals:

$$\widehat{\epsilon}_{ic} = \frac{Y_i - \widehat{Y}_i}{\sqrt{\widehat{\mathrm{Var}}[Y_i]}}$$

In generalized linear models, where the variance of an observation is a function of the mean μ and possibly of an extra scale parameter, $\mathrm{Var}[Y] = a(\mu)\phi$, the Pearson residual is

$$\widehat{\epsilon}_{iP} = \frac{Y_i - \widehat{\mu}_i}{\sqrt{a(\widehat{\mu})}}$$

because the sum of the squared Pearson residuals equals the Pearson X^2 statistic:

$$X^2 = \sum_{i=1}^{n} \widehat{\epsilon}_{iP}^2$$

When the scale parameter ϕ participates in the scaling, the residual is also referred to as a Pearson-type residual:

$$\widehat{\epsilon}_{iP} = \frac{Y_i - \widehat{\mu}_i}{\sqrt{a(\widehat{\mu})\phi}}$$

Other Residuals

You might encounter other residuals in SAS/STAT software. A "leave-one-out" residual is the difference between the observed value and the residual obtained from fitting a model in which the observation in question did not participate. If $\widehat{Y}_i$ is the predicted value of the ith observation and $\widehat{Y}_{i,-i}$ is the predicted value if Y_i is removed from the analysis, then the "leave-one-out" residual is

$$\widehat{\epsilon}_{i,-i} = Y_i - \widehat{Y}_{i,-i}$$

Since the sum of the squared "leave-one-out" residuals is the PRESS statistic (prediction sum of squares; Allen 1974), $\widehat{\epsilon}_{i,-i}$ is also called the PRESS residual. The concept of the PRESS residual can be generalized if the deletion residual can be based on the removal of sets of observations. In the classical linear model, the PRESS residual for case deletion has a particularly simple form:

$$\widehat{\epsilon}_{i,-i} = Y_i - \widehat{Y}_{i,-i} = \frac{\widehat{\epsilon}_i}{1 - h_{ii}}$$

That is, the PRESS residual is simply a scaled form of the raw residual, where the scaling factor is a function of the leverage of the observation.

When data are correlated, $\mathrm{Var}[\mathbf{Y}] = \mathbf{V}$, you can scale the vector of residuals rather than scale each residual separately. This takes the covariances among the observations into account. This form of scaling is accomplished by forming the Cholesky root $\mathbf{C}'\mathbf{C} = \mathbf{V}$, where $\mathbf{C}'$ is a lower-triangular matrix. Then $\mathbf{C}'^{-1}\mathbf{Y}$ is a vector of uncorrelated variables with unit variance. The Cholesky residuals in the model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ are

$$\widehat{\boldsymbol{\epsilon}}_C = \mathbf{C}'^{-1}\left(\mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\right)$$

In generalized linear models, the fit of a model can be measured by the scaled deviance statistic D^*. It measures the difference between the log likelihood under the model and the maximum log likelihood that is achievable. In models with a scale parameter ϕ, the deviance is $D = \phi \times D^* = \sum_{i=1}^{n} d_i$. The deviance residuals are the signed square roots of the contributions to the deviance statistic:

$$\widehat{\epsilon}_{id} = \mathrm{sign}\{y_i - \widehat{\mu}_i\}\sqrt{d_i}$$

Sweep Operator

The sweep operator (Goodnight 1979) is closely related to Gauss-Jordan elimination and the Forward Doolittle procedure. The fact that a sweep operation can produce a generalized inverse by in-place mapping with minimal storage and that its application invariably leads to some form of matrix inversion is important, but this observation does not do justice to the pervasive relevance of sweeping to statistical computing. In this section the sweep operator is discussed as a conceptual tool for further insight into linear model operations. Consider the nonnegative definite, symmetric, partitioned matrix

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{12}' & \mathbf{A}_{22} \end{bmatrix}$$

Sweeping a matrix consists of performing a series of row operations akin to Gauss-Jordan elimination. Basic row operations are the multiplication of a row by a constant and the addition of a multiple of one row to another. The sweep operator restricts row operations to pivots on the diagonal elements of a matrix; further details about the elementary operations can be found in Goodnight (1979). The process of sweeping the matrix $\mathbf{A}$ on its leading partition is denoted as $\mathrm{Sweep}(\mathbf{A}, \mathbf{A}_{11})$ and leads to

$$\mathrm{Sweep}(\mathbf{A}, \mathbf{A}_{11}) = \begin{bmatrix} \mathbf{A}_{11}^{-} & \mathbf{A}_{11}^{-}\mathbf{A}_{12} \\ -\mathbf{A}_{12}'\mathbf{A}_{11}^{-} & \mathbf{A}_{22} - \mathbf{A}_{12}'\mathbf{A}_{11}^{-}\mathbf{A}_{12} \end{bmatrix}$$

If the kth row and column are set to zero when the pivot is zero (or in practice, less than some singularity tolerance), the generalized inverse in the leading position of the swept matrix is a reflexive, g_2-inverse. Suppose that the crossproduct matrix of the linear model is augmented with a "Y-border" as follows:

$$\mathbf{C} = \begin{bmatrix} \mathbf{X}'\mathbf{X} & \mathbf{X}'\mathbf{Y} \\ \mathbf{Y}'\mathbf{X} & \mathbf{Y}'\mathbf{Y} \end{bmatrix}$$

Then the result of sweeping on the rows of $\mathbf{X}$ is

$$\begin{aligned} \mathrm{Sweep}(\mathbf{C}, \mathbf{X}) &= \begin{bmatrix} (\mathbf{X}'\mathbf{X})^{-} & (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y} \\ -\mathbf{Y}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-} & \mathbf{Y}'\mathbf{Y} - \mathbf{Y}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y} \end{bmatrix} \\ &= \begin{bmatrix} (\mathbf{X}'\mathbf{X})^{-} & \widehat{\boldsymbol{\beta}} \\ -\widehat{\boldsymbol{\beta}} & \mathbf{Y}'\mathbf{M}\mathbf{Y} \end{bmatrix} = \begin{bmatrix} (\mathbf{X}'\mathbf{X})^{-} & \widehat{\boldsymbol{\beta}} \\ -\widehat{\boldsymbol{\beta}} & \mathrm{SSR} \end{bmatrix} \end{aligned}$$

The "Y-border" has been transformed into the least squares solution and the residual sum of squares.

Partial sweeps are common in model selection. Suppose that the $\mathbf{X}$ matrix is partitioned as $[\mathbf{X}_1 \ \mathbf{X}_2]$, and consider the augmented crossproduct matrix

$$\mathbf{C} = \begin{bmatrix} \mathbf{X}_1'\mathbf{X}_1 & \mathbf{X}_1'\mathbf{X}_2 & \mathbf{X}_1'\mathbf{Y} \\ \mathbf{X}_2'\mathbf{X}_1 & \mathbf{X}_2'\mathbf{X}_2 & \mathbf{X}_2'\mathbf{Y} \\ \mathbf{Y}'\mathbf{X}_1 & \mathbf{Y}'\mathbf{X}_2 & \mathbf{Y}'\mathbf{Y} \end{bmatrix}$$

Sweeping on the $\mathbf{X}_1$ partition yields

$$\mathrm{Sweep}(\mathbf{C}, \mathbf{X}_1) = \begin{bmatrix} (\mathbf{X}_1'\mathbf{X}_1)^{-} & (\mathbf{X}_1'\mathbf{X}_1)^{-}\mathbf{X}_1'\mathbf{X}_2 & (\mathbf{X}_1'\mathbf{X}_1)^{-}\mathbf{X}_1'\mathbf{Y} \\ -\mathbf{X}_2'\mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-} & \mathbf{X}_2'\mathbf{M}_1\mathbf{X}_2 & \mathbf{X}_2'\mathbf{M}_1\mathbf{Y} \\ -\mathbf{Y}'\mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-} & -\mathbf{Y}'\mathbf{M}_1\mathbf{X}_2 & \mathbf{Y}'\mathbf{M}_1\mathbf{Y} \end{bmatrix}$$

where $\mathbf{M}_1 = \mathbf{I} - \mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-}\mathbf{X}_1'$. The entries in the first row of this partition are the generalized inverse of $\mathbf{X}'\mathbf{X}$, the coefficients for regressing $\mathbf{X}_2$ on $\mathbf{X}_1$, and the coefficients for regressing $\mathbf{Y}$ on $\mathbf{X}_1$. The diagonal

entries $\mathbf{X}_2'\mathbf{M}_1\mathbf{X}_2$ and $\mathbf{Y}'\mathbf{M}_1\mathbf{Y}$ are the sum of squares and crossproduct matrices for regressing $\mathbf{X}_2$ on $\mathbf{X}_1$ and for regressing $\mathbf{Y}$ on $\mathbf{X}_1$, respectively. As you continue to sweep the matrix, the last cell in the partition contains the residual sum of square of a model in which $\mathbf{Y}$ is regressed on all columns swept up to that point.

The sweep operator is not only useful to conceptualize the computation of least squares solutions, Type I and Type II sums of squares, and generalized inverses. It can also be used to obtain other statistical information. For example, adding the logarithms of the pivots of the rows that are swept yields the log determinant of the matrix.

References

Allen, D. M. (1974). "The Relationship between Variable Selection and Data Augmentation and a Method of Prediction." *Technometrics* 16:125–127.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Harville, D. A. (1997). *Matrix Algebra from a Statistician's Perspective*. New York: Springer-Verlag.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Jöreskog, K. G. (1973). "A General Method for Estimating a Linear Structural Equation System." In *Structural Equation Models in the Social Sciences*, edited by A. S. Goldberger and O. D. Duncan, 85–112. New York: Academic Press.

Keesling, J. W. (1972). "Maximum Likelihood Approaches to Causal Analysis." Ph.D. diss., University of Chicago.

Magnus, J. R., and Neudecker, H. (1999). *Matrix Differential Calculus with Applications in Statistics and Econometrics*. New York: John Wiley & Sons.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Moore, E. H. (1920). "On the Reciprocal of the General Algebraic Matrix." *Bulletin of the American Mathematical Society* 26:394–395.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Pawitan, Y. (2001). *In All Likelihood: Statistical Modelling and Inference Using Likelihood*. Oxford: Clarendon Press.

Penrose, R. A. (1955). "A Generalized Inverse for Matrices." *Proceedings of the Cambridge Philosophical Society* 51:406–413.

Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Spearman, C. (1904). "General Intelligence Objectively Determined and Measured." *American Journal of Psychology* 15:201–293.

Wedderburn, R. W. M. (1974). "Quasi-likelihood Functions, Generalized Linear Models, and the Gauss-Newton Method." *Biometrika* 61:439–447.

Wiley, D. E. (1973). "The Identification Problem for Structural Equation Models with Unmeasured Variables." In *Structural Equation Models in the Social Sciences*, edited by A. S. Goldberger and O. D. Duncan, 69–83. New York: Academic Press.

Chapter 4
Introduction to Regression Procedures

Contents

Overview: Regression Procedures **70**
- Introduction 70
- Introductory Example: Linear Regression 75
- Model Selection Methods 80
- Linear Regression: The REG Procedure 82
- Model Selection: The GLMSELECT Procedure 83
- Response Surface Regression: The RSREG Procedure 83
- Partial Least Squares Regression: The PLS Procedure 84
- Generalized Linear Regression 84
  - Contingency Table Data: The CATMOD Procedure 85
  - Generalized Linear Models: The GENMOD Procedure 85
  - Generalized Linear Mixed Models: The GLIMMIX Procedure 85
  - Logistic Regression: The LOGISTIC Procedure 85
  - Discrete Event Data: The PROBIT Procedure 85
  - Correlated Data: The GENMOD and GLIMMIX Procedures 85
- Ill-Conditioned Data: The ORTHOREG Procedure 86
- Quantile Regression: The QUANTREG and QUANTSELECT Procedures 86
- Nonlinear Regression: The NLIN and NLMIXED Procedures 87
- Nonparametric Regression 88
  - Adaptive Regression: The ADAPTIVEREG Procedure 88
  - Generalized Additive Models: The GAM and GAMPL Procedures 88
  - Local Regression: The LOESS Procedure 89
  - Thin Plate Smoothing Splines: The TPSPLINE Procedure 89
- Robust Regression: The ROBUSTREG Procedure 89
- Regression with Transformations: The TRANSREG Procedure 90
- Interactive Features in the CATMOD, GLM, and REG Procedures 90

Statistical Background in Linear Regression **91**
- Linear Regression Models 91
- Parameter Estimates and Associated Statistics 92
- Predicted and Residual Values 96
- Testing Linear Hypotheses 98
- Multivariate Tests 98
- Comments on Interpreting Regression Statistics 102

References **106**

Overview: Regression Procedures

This chapter provides an overview of SAS/STAT procedures that perform regression analysis. The REG procedure provides extensive capabilities for fitting linear regression models that involve individual numeric independent variables. Many other procedures can also fit regression models, but they focus on more specialized forms of regression, such as robust regression, generalized linear regression, nonlinear regression, nonparametric regression, quantile regression, regression modeling of survey data, regression modeling of survival data, and regression modeling of transformed variables. The SAS/STAT procedures that can fit regression models include the ADAPTIVEREG, CATMOD, GAM, GAMPL, GENMOD, GLIMMIX, GLM, GLMSELECT, HPGENSELECT, HPLOGISTIC, HPNLMOD, HPQUANTSELECT, HPREG, LIFEREG, LOESS, LOGISTIC, MIXED, NLIN, NLMIXED, ORTHOREG, PHREG, PLS, PROBIT, QUANTREG, QUANTSELECT, REG, ROBUSTREG, RSREG, SURVEYLOGISTIC, SURVEYPHREG, SURVEYREG, TPSPLINE, and TRANSREG procedures. A number of procedures in SAS/ETS software also fit regression models, including the ARIMA, AUTOREG, COUNTREG, MDC, MODEL, PANEL, PDLREG, QLIM, SYSLIN, and VARMAX procedures.

Introduction

In a linear regression model, the mean of a response variable **Y** is a function of parameters and covariates in a statistical model. The many forms of regression models have their origin in the characteristics of the response variable (discrete or continuous, normally or nonnormally distributed), assumptions about the form of the model (linear, nonlinear, or generalized linear), assumptions about the data-generating mechanism (survey, observational, or experimental data), and estimation principles. Some models contain classification (or CLASS) variables that enter the model not through their values but through their levels. For an introduction to linear regression models, see Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software." For information that is common to many of the regression procedures, see Chapter 20, "Shared Concepts and Topics." The following procedures, listed in alphabetical order, perform at least one type of regression analysis.

ADAPTIVEREG fits multivariate adaptive regression spline models. This is a nonparametric regression technique that combines both regression splines and model selection methods. PROC ADAPTIVEREG produces parsimonious models that do not overfit the data and thus have good predictive power. It supports CLASS variables. For more information, see Chapter 28, "The ADAPTIVEREG Procedure."

CATMOD analyzes data that can be represented by a contingency table. It fits linear models to functions of response frequencies, and it can be used for linear and logistic regression. PROC CATMOD supports CLASS variables. For more information, see Chapter 9, "Introduction to Categorical Data Analysis Procedures," and Chapter 36, "The CATMOD Procedure."

GAM fits generalized additive models by using the backfitting method of Hastie and Tibshirani (1990). PROC GAM uses smoothing splines to model each unknown additive term. It supports CLASS variables. For more information, see Chapter 48, "The GAM Procedure."

NOTE: As an alternative to PROC GAM, the GAMPL procedure provides an approach that searches for models that have optimal degrees of freedom for each smoother. It is also more computationally efficient than PROC GAM, especially for large problems.

GAMPL — fits generalized additive models by using the penalized likelihood method of Wood (2006). PROC GAMPL uses low-rank regression splines to model each unknown additive term. It supports CLASS variables. For more information, see Chapter 49, "The GAMPL Procedure."

GENMOD — fits generalized linear models. It is especially suited for responses that have discrete outcomes, and it performs logistic regression and Poisson regression in addition to fitting generalized estimating equations for repeated measures data. PROC GENMOD supports CLASS variables and provides Bayesian analysis capabilities. For more information, see Chapter 9, "Introduction to Categorical Data Analysis Procedures," and Chapter 51, "The GENMOD Procedure."

GLIMMIX — uses likelihood-based methods to fit generalized linear mixed models. It can perform simple, multiple, polynomial, and weighted regression, in addition to many other analyses. It can fit linear mixed models, which have random effects, and models that do not have random effects. PROC GLIMMIX supports CLASS variables. For more information, see Chapter 52, "The GLIMMIX Procedure."

GLM — uses the method of least squares to fit general linear models. It can perform simple, multiple, polynomial, and weighted regression in addition to many other analyses. It has many of the same input/output capabilities as PROC REG, but it does not provide as many diagnostic tools or allow interactive changes in the model or data. PROC GLM supports CLASS variables. For more information, see Chapter 5, "Introduction to Analysis of Variance Procedures," and Chapter 53, "The GLM Procedure."

GLMSELECT — performs variable selection in the framework of general linear models. It supports CLASS variables (like PROC GLM) and model selection (like PROC REG). A variety of model selection methods are available, including forward, backward, stepwise, LASSO, and least angle regression. PROC GLMSELECT provides a variety of selection and stopping criteria. For more information, see Chapter 56, "The GLMSELECT Procedure."

HPGENSELECT — is a high-performance procedure that provides model fitting and model building for generalized linear models. It fits models for standard distributions in the exponential family, such as the normal, Poisson, and Tweedie distributions. In addition, it fits multinomial models for ordinal and nominal responses, and it fits zero-inflated Poisson and negative binomial models for count data. For all these models, PROC HPGENSELECT provides forward, backward, and stepwise variable selection. It supports CLASS variables. For more information, see Chapter 59, "The HPGENSELECT Procedure."

HPLOGISTIC — is a high-performance procedure that fits logistic regression models for binary, binomial, and multinomial data. It fits logistic regression models in the broader sense; it permits several link functions and can handle ordinal and nominal data that have more than two response categories (multinomial data). PROC HPLOGISTIC supports CLASS variables. For more information, see Chapter 61, "The HPLOGISTIC Procedure."

HPNLMOD is a high-performance procedure that uses either nonlinear least squares or maximum likelihood to fit nonlinear regression models. It enables you to specify the model by using SAS programming statements, which give you greater flexibility in modeling the relationship between the response variable and independent (regressor) variables than SAS procedures that use a more structured MODEL statement. For more information, see Chapter 63, "The HPNLMOD Procedure."

HPQUANTSELECT is a high-performance procedure that fits quantile regression models and performs effect selection. Quantile regression is a systematic statistical methodology for modeling conditional quantile functions of a response variable on explanatory covariate effects. PROC HPQUANTSELECT supports CLASS variables. For more information, see Chapter 66, "The HPQUANTSELECT Procedure."

HPREG is a high-performance procedure that fits and performs model selection for ordinary linear least squares models. The supported models are standard independently and identically distributed (iid) general linear models, which can contain main effects that consist of both continuous and classification variables and the interaction effects of these variables. PROC HPREG offers extensive capabilities for customizing the model selection by using a wide variety of selection and stopping criteria, from traditional and computationally efficient significance-level-based criteria to more computationally intensive validation-based criteria. It also provides a variety of regression diagnostics that are conditional on the selected model. PROC HPREG supports CLASS variables. For more information, see Chapter 67, "The HPREG Procedure."

LIFEREG fits parametric models to failure-time data that might be right-censored. These types of models are commonly used in survival analysis. PROC LIFEREG supports CLASS variables and provides Bayesian analysis capabilities. For more information, see Chapter 14, "Introduction to Survival Analysis Procedures," and Chapter 76, "The LIFEREG Procedure."

LOESS uses a local regression method to fit nonparametric models. It is suitable for modeling regression surfaces in which the underlying parametric form is unknown and for which robustness in the presence of outliers is required. For more information, see Chapter 78, "The LOESS Procedure."

LOGISTIC fits logistic models for binomial and ordinal outcomes. It provides a wide variety of model selection methods and computes numerous regression diagnostics. PROC LOGISTIC supports CLASS variables. For more information, see Chapter 9, "Introduction to Categorical Data Analysis Procedures," and Chapter 79, "The LOGISTIC Procedure."

MIXED uses likelihood-based techniques to fit linear mixed models. It can perform simple, multiple, polynomial, and weighted regression, in addition to many other analyses. It can fit linear mixed models, which have random effects, and models that do not have random effects. PROC MIXED supports CLASS variables. For more information, see Chapter 84, "The MIXED Procedure."

NLIN uses the method of nonlinear least squares to fit general nonlinear regression models. Several different iterative methods are available. For more information, see Chapter 88, "The NLIN Procedure."

NLMIXED uses the method of maximum likelihood to fit general nonlinear mixed regression models. PROC NLMIXED enables you to specify a custom objective function for parameter estimation and to fit models with or without random effects. For more information, see Chapter 89, "The NLMIXED Procedure."

ORTHOREG uses the Gentleman-Givens computational method to perform regression. For ill-conditioned data, PROC ORTHOREG can produce more-accurate parameter estimates than procedures such as PROC GLM and PROC REG. PROC ORTHOREG supports CLASS variables. For more information, see Chapter 91, "The ORTHOREG Procedure."

PHREG fits Cox proportional hazards regression models to survival data. PROC PHREG supports CLASS variables and provides Bayesian analysis capabilities. For more information, see Chapter 14, "Introduction to Survival Analysis Procedures," and Chapter 92, "The PHREG Procedure."

PLS performs partial least squares regression, principal component regression, and reduced rank regression, along with cross validation for the number of components. PROC PLS supports CLASS variables. For more information, see Chapter 95, "The PLS Procedure."

PROBIT performs probit regression in addition to logistic regression and ordinal logistic regression. PROC PROBIT is useful when the dependent variable is either dichotomous or polychotomous and the independent variables are continuous. PROC PROBIT supports CLASS variables. For more information, see Chapter 100, "The PROBIT Procedure."

QUANTREG uses quantile regression to model the effects of covariates on the conditional quantiles of a response variable. PROC QUANTREG supports CLASS variables. For more information, see Chapter 103, "The QUANTREG Procedure."

QUANTSELECT provides variable selection for quantile regression models. Selection methods include forward, backward, stepwise, and LASSO. The procedure provides a variety of selection and stopping criteria. PROC QUANTSELECT supports CLASS variables. For more information, see Chapter 104, "The QUANTSELECT Procedure."

REG performs linear regression with many diagnostic capabilities. PROC REG produces fit, residual, and diagnostic plots; heat maps; and many other types of graphs. PROC REG enables you to select models by using any one of nine methods, and you can interactively change both the regression model and the data that are used to fit the model. For more information, see Chapter 105, "The REG Procedure."

ROBUSTREG uses Huber M estimation and high breakdown value estimation to perform robust regression. PROC ROBUSTREG is suitable for detecting outliers and providing resistant (stable) results in the presence of outliers. PROC ROBUSTREG supports CLASS variables. For more information, see Chapter 107, "The ROBUSTREG Procedure."

RSREG builds quadratic response-surface regression models. PROC RSREG analyzes the fitted response surface to determine the factor levels of optimum response and performs a ridge analysis to search for the region of optimum response. For more information, see Chapter 108, "The RSREG Procedure."

SURVEYLOGISTIC uses the method of maximum likelihood to fit logistic models for binary and ordinal outcomes to survey data. PROC SURVEYLOGISTIC supports CLASS variables.

For more information, see Chapter 15, "Introduction to Survey Procedures," and Chapter 120, "The SURVEYLOGISTIC Procedure."

SURVEYPHREG fits proportional hazards models for survey data by maximizing a partial pseudo-likelihood function that incorporates the sampling weights. The SURVEYPHREG procedure provides design-based variance estimates, confidence intervals, and tests for the estimated proportional hazards regression coefficients. PROC SURVEYPHREG supports CLASS variables. For more information, see Chapter 15, "Introduction to Survey Procedures," Chapter 14, "Introduction to Survival Analysis Procedures," and Chapter 122, "The SURVEYPHREG Procedure."

SURVEYREG uses elementwise regression to fit linear regression models to survey data by generalized least squares. PROC SURVEYREG supports CLASS variables. For more information, see Chapter 15, "Introduction to Survey Procedures," and Chapter 123, "The SURVEYREG Procedure."

TPSPLINE uses penalized least squares to fit nonparametric regression models. PROC TPSPLINE makes no assumptions of a parametric form for the model. For more information, see Chapter 125, "The TPSPLINE Procedure."

TRANSREG fits univariate and multivariate linear models, optionally with spline, Box-Cox, and other nonlinear transformations. Models include regression and ANOVA, conjoint analysis, preference mapping, redundancy analysis, canonical correlation, and penalized B-spline regression. PROC TRANSREG supports CLASS variables. For more information, see Chapter 126, "The TRANSREG Procedure."

Several SAS/ETS procedures also perform regression. The following procedures are documented in the *SAS/ETS User's Guide*:

ARIMA uses autoregressive moving-average errors to perform multiple regression analysis. For more information, see Chapter 7, "ARIMA Procedure" (*SAS/ETS User's Guide*).

AUTOREG implements regression models that use time series data in which the errors are autocorrelated. For more information, see Chapter 8, "AUTOREG Procedure" (*SAS/ETS User's Guide*).

COUNTREG analyzes regression models in which the dependent variable takes nonnegative integer or count values. For more information, see Chapter 11, "COUNTREG Procedure" (*SAS/ETS User's Guide*).

MDC fits conditional logit, mixed logit, heteroscedastic extreme value, nested logit, and multinomial probit models to discrete choice data. For more information, see Chapter 23, "MDC Procedure" (*SAS/ETS User's Guide*).

MODEL handles nonlinear simultaneous systems of equations, such as econometric models. For more information, see Chapter 24, "MODEL Procedure" (*SAS/ETS User's Guide*).

PANEL analyzes a class of linear econometric models that commonly arise when time series and cross-sectional data are combined. For more information, see Chapter 25, "PANEL Procedure" (*SAS/ETS User's Guide*).

PDLREG fits polynomial distributed lag regression models. For more information, see Chapter 26, "PDLREG Procedure" (*SAS/ETS User's Guide*).

QLIM	analyzes limited dependent variable models in which dependent variables take discrete values or are observed only in a limited range of values. For more information, see Chapter 27, "QLIM Procedure" (*SAS/ETS User's Guide*).
SYSLIN	handles linear simultaneous systems of equations, such as econometric models. For more information, see Chapter 35, "SYSLIN Procedure" (*SAS/ETS User's Guide*).
VARMAX	performs multiple regression analysis for multivariate time series dependent variables by using current and past vectors of dependent and independent variables as predictors, with vector autoregressive moving-average errors, and with modeling of time-varying heteroscedasticity. For more information, see Chapter 42, "VARMAX Procedure" (*SAS/ETS User's Guide*).

Introductory Example: Linear Regression

Regression analysis models the relationship between a response or outcome variable and another set of variables. This relationship is expressed through a statistical model equation that predicts a *response variable* (also called a *dependent variable* or *criterion*) from a function of *regressor variables* (also called *independent variables, predictors, explanatory variables, factors,* or *carriers*) and *parameters*. In a linear regression model, the predictor function is linear in the parameters (but not necessarily linear in the regressor variables). The parameters are estimated so that a measure of fit is optimized. For example, the equation for the *i*th observation might be

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

where Y_i is the response variable, x_i is a regressor variable, β_0 and β_1 are unknown parameters to be estimated, and ϵ_i is an error term. This model is called the simple linear regression (SLR) model, because it is linear in β_0 and β_1 and contains only a single regressor variable.

Suppose you are using regression analysis to relate a child's weight to the child's height. One application of a regression model that contains the response variable Weight is to predict a child's weight for a known height. Suppose you collect data by measuring heights and weights of 19 randomly selected schoolchildren. A simple linear regression model that contains the response variable Weight and the regressor variable Height can be written as

$$\text{Weight}_i = \beta_0 + \beta_1 \text{Height}_i + \epsilon_i$$

where

Weight_i	is the response variable for the *i*th child
Height_i	is the regressor variable for the *i*th child
β_0, β_1	are the unknown regression parameters
ϵ_i	is the unobservable random error associated with the *i*th observation

The data set Sashelp.class, which is available in the Sashelp library, identifies the children and their observed heights (the variable Height) and weights (the variable Weight). The following statements perform the regression analysis:

```
ods graphics on;
proc reg data=sashelp.class;
   model Weight = Height;
run;
```

Figure 4.1 displays the default tabular output of PROC REG for this model. Nineteen observations are read from the data set, and all observations are used in the analysis. The estimates of the two regression parameters are $\widehat{\beta}_0 = -143.02692$ and $\widehat{\beta_1} = 3.89903$. These estimates are obtained by the least squares principle. For more information about the principle of least squares estimation and its role in linear model analysis, see the sections "Classical Estimation Principles" and "Linear Model Theory" in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software." Also see an applied regression text such as Draper and Smith (1998); Daniel and Wood (1999); Johnston and DiNardo (1997); Weisberg (2005).

Figure 4.1 Regression for Weight and Height Data

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Number of Observations Read	19
Number of Observations Used	19

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	7193.24912	7193.24912	57.08	<.0001
Error	17	2142.48772	126.02869		
Corrected Total	18	9335.73684			

Root MSE	11.22625	**R-Square**	0.7705
Dependent Mean	100.02632	**Adj R-Sq**	0.7570
Coeff Var	11.22330		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

Based on the least squares estimates shown in Figure 4.1, the fitted regression line that relates height to weight is described by the equation

$$\widehat{\text{Weight}} = -143.02692 + 3.89903 \times \text{Height}$$

The "hat" notation is used to emphasize that $\widehat{\text{Weight}}$ is not one of the original observations but a value predicted under the regression model that has been fit to the data. In the least squares solution, the following residual sum of squares is minimized and the achieved criterion value is displayed in the analysis of variance table as the error sum of squares (2142.48772):

$$\text{SSE} = \sum_{i=1}^{19} (\text{Weight}_i - \beta_0 - \beta_1 \text{Height}_i)^2$$

Figure 4.2 displays the fit plot that is produced by ODS Graphics. The fit plot shows the positive slope of the fitted line. The average weight of a child changes by $\widehat{\beta}_1 = 3.89903$ units for each unit change in height. The 95% confidence limits in the fit plot are pointwise limits that cover the mean weight for a particular height with probability 0.95. The prediction limits, which are wider than the confidence limits, show the pointwise limits that cover a new observation for a given height with probability 0.95.

Figure 4.2 Fit Plot for Regression of Weight on Height

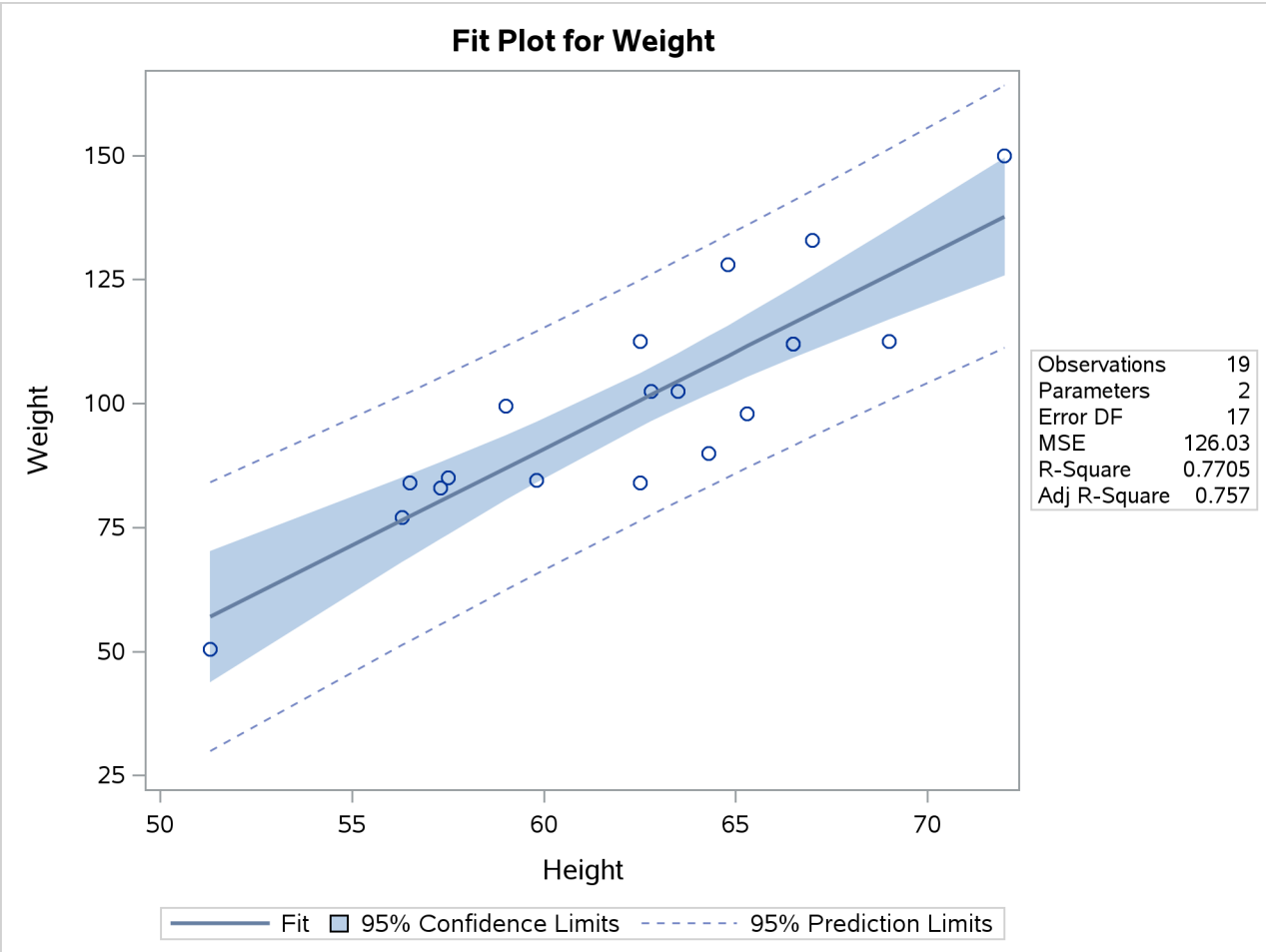

Regression is often used in an exploratory fashion to look for empirical relationships, such as the relationship between Height and Weight. In this example, Height is not the cause of Weight. You would need a controlled experiment to confirm the relationship scientifically. For more information, see the section "Comments on Interpreting Regression Statistics" on page 102. A separate question from whether there is a cause-and-effect relationship between the two variables that are involved in this regression is whether the simple linear regression model adequately describes the relationship among these data. If the SLR model makes the usual assumptions about the model errors ϵ_i, then the errors should have zero mean and equal variance and be uncorrelated. Because the children were randomly selected, the observations from different children are not correlated. If the mean function of the model is correctly specified, the fitted residuals $\mathrm{Weight}_i - \widehat{\mathrm{Weight}}_i$ should scatter around the zero reference line without discernible structure. The residual plot in Figure 4.3 confirms this.

Figure 4.3 Residual Plot for Regression of Weight on Height

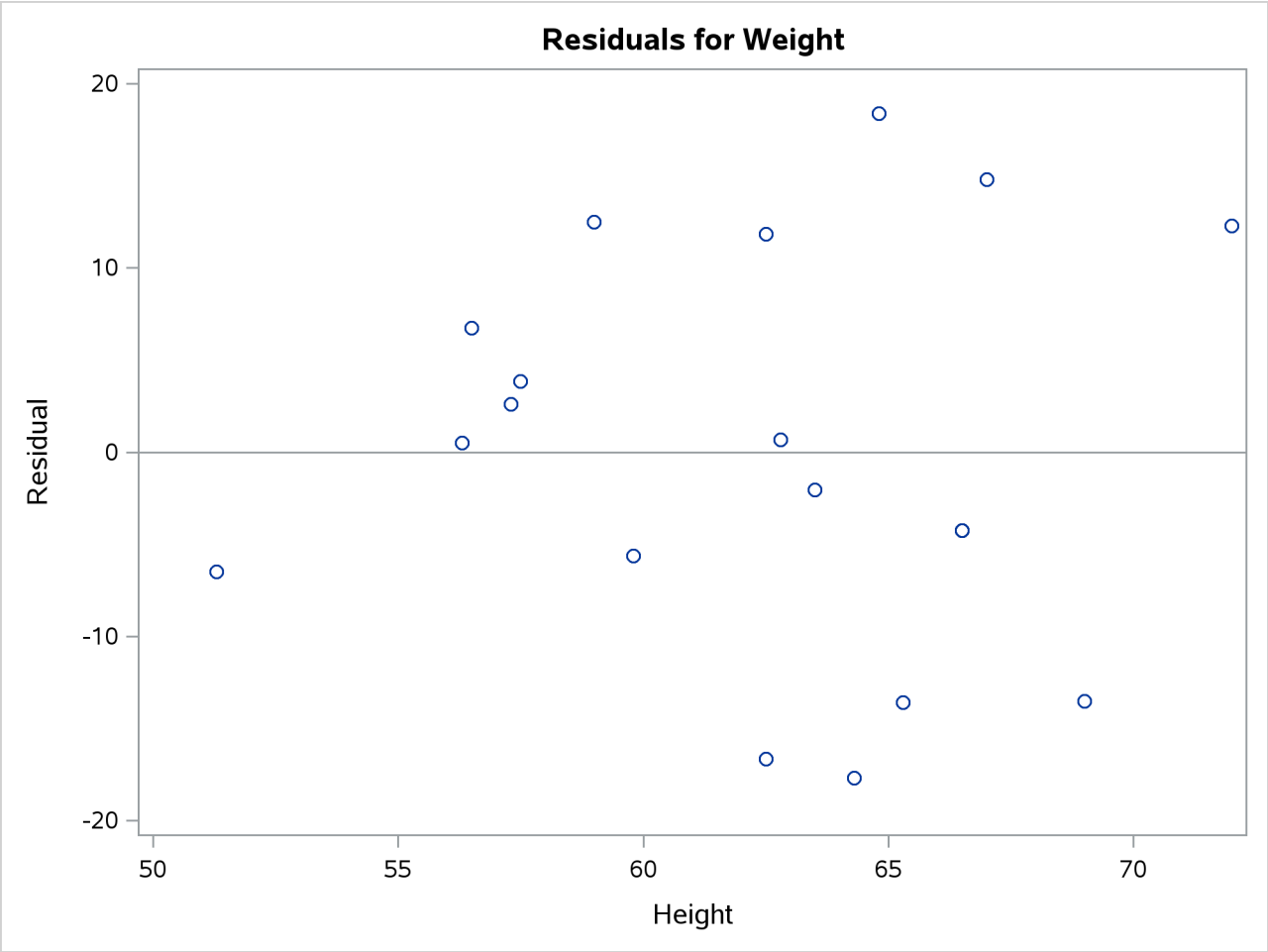

The panel of regression diagnostics in Figure 4.4 provides an even more detailed look at the model-data agreement. The graph in the upper left panel repeats the raw residual plot in Figure 4.3. The plot of the RSTUDENT residuals shows externally studentized residuals that take into account heterogeneity in the variability of the residuals. RSTUDENT residuals that exceed the threshold values of ± 2 often indicate outlying observations. The residual-by-leverage plot shows that two observations have high leverage—that is, they are unusual in their height values relative to the other children. The normal-probability Q-Q plot in the second row of the panel shows that the normality assumption for the residuals is reasonable. The plot of the Cook's D statistic shows that observation 15 exceeds the threshold value, indicating that the observation for this child has a strong influence on the regression parameter estimates.

Figure 4.4 Panel of Regression Diagnostics

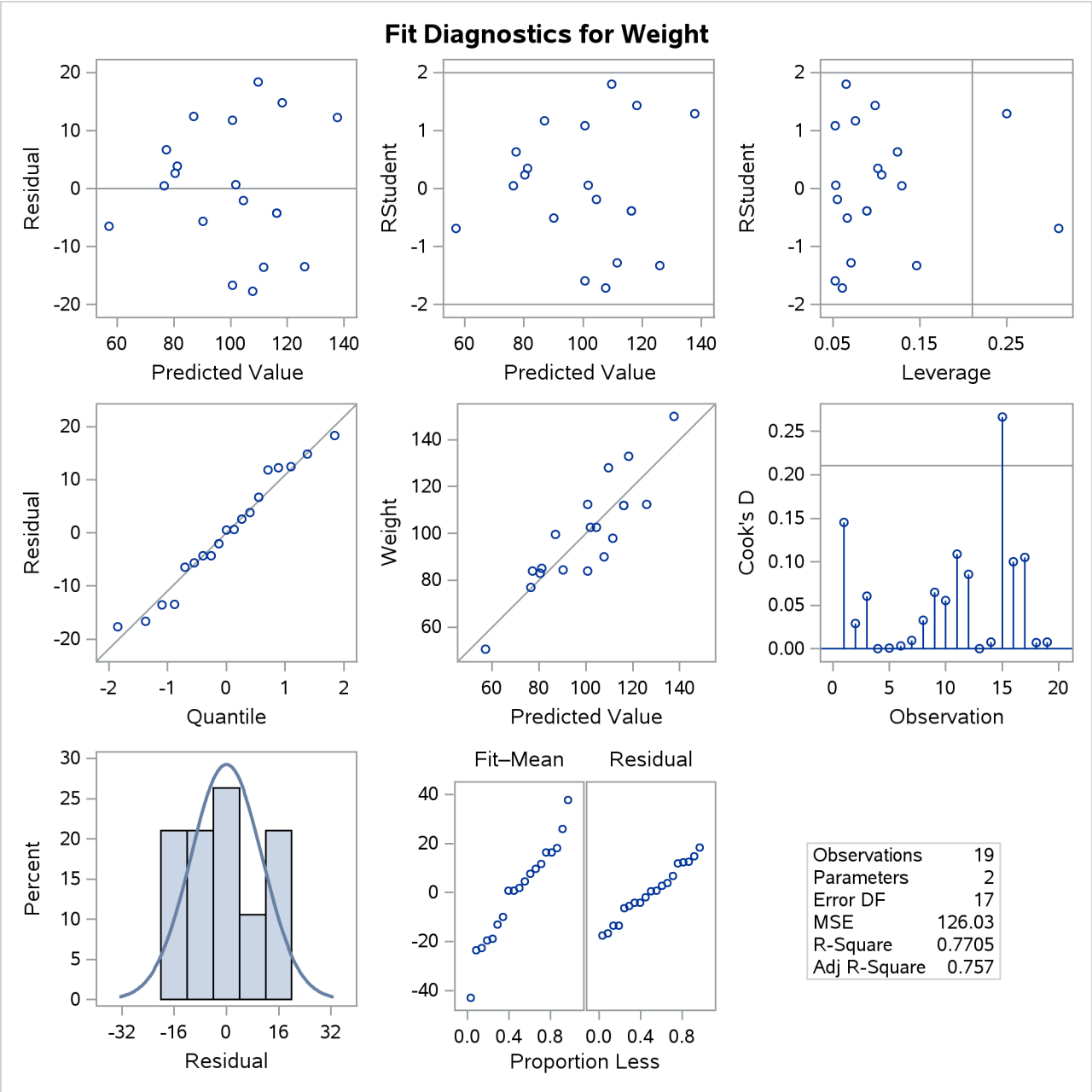

For more information about the interpretation of regression diagnostics and about ODS statistical graphics with PROC REG, see Chapter 105, "The REG Procedure."

SAS/STAT regression procedures produce the following information for a typical regression analysis:

- parameter estimates that are derived by using the least squares criterion
- estimates of the variance of the error term
- estimates of the variance or standard deviation of the sampling distribution of the parameter estimates
- tests of hypotheses about the parameters

SAS/STAT regression procedures can produce many other specialized diagnostic statistics, including the following:

- collinearity diagnostics to measure how strongly regressors are related to other regressors and how this relationship affects the stability and variance of the estimates (REG procedure)
- influence diagnostics to measure how each individual observation contributes to determining the parameter estimates, the SSE, and the fitted values (GENMOD, GLM, LOGISTIC, MIXED, NLIN, PHREG, REG, and RSREG procedures)
- lack-of-fit diagnostics that measure the lack of fit of the regression model by comparing the error variance estimate to another pure error variance that does not depend on the form of the model (CATMOD, LOGISTIC, PROBIT, and RSREG procedures)
- diagnostic plots that check the fit of the model (GLM, LOESS, PLS, REG, RSREG, and TPSPLINE procedures)
- predicted and residual values, and confidence intervals for the mean and for an individual value (GLIMMIX, GLM, LOESS, LOGISTIC, NLIN, PLS, REG, RSREG, TPSPLINE, and TRANSREG procedures)
- time series diagnostics for equally spaced time series data that measure how closely errors might be related across neighboring observations. These diagnostics can also measure functional goodness of fit for data that are sorted by regressor or response variables (REG and SAS/ETS procedures).

Many SAS/STAT procedures produce general and specialized statistical graphics through ODS Graphics to diagnose the fit of the model and the model-data agreement, and to highlight observations that strongly influence the analysis. Figure 4.2, Figure 4.3, and Figure 4.4, for example, show three of the ODS statistical graphs that are produced by PROC REG by default for the simple linear regression model. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the ODS statistical graphs available with a SAS/STAT procedure, see the PLOTS option in the "Syntax" section for the PROC statement and the "ODS Graphics" section in the "Details" section of the individual procedure documentation.

Model Selection Methods

Statistical model selection (or model building) involves forming a model from a set of regressor variables that fits the data well but without overfitting. Models are overfit when they contain too many unimportant regressor variables. Overfit models are too closely molded to a particular data set. As a result, overfit models have unstable regression coefficients and are quite likely to have poor predictive power. Guided, numerical variable selection methods offer one approach to building models in situations where many potential regressor variables are available for inclusion in a regression model.

Both the REG and GLMSELECT procedures provide extensive options for model selection in ordinary linear regression models.[1] PROC GLMSELECT provides the most modern and flexible options for model selection. PROC GLMSELECT provides more selection options and criteria than PROC REG, and PROC

[1] The QUANTSELECT, PHREG, and LOGISTIC procedures provide model selection for quantile, proportional hazards, and logistic regression, respectively.

GLMSELECT also supports CLASS variables. For more information about PROC GLMSELECT, see Chapter 56, "The GLMSELECT Procedure." For more information about PROC REG, see Chapter 105, "The REG Procedure."

SAS/STAT procedures provide the following model selection options for regression models:

NONE performs no model selection. This method uses the full model given in the MODEL statement to fit the model. This selection method is available in the GLMSELECT, LOGISTIC, PHREG, QUANTSELECT, and REG procedures.

FORWARD uses a forward-selection algorithm to select variables. This method starts with no variables in the model and adds variables one by one to the model. At each step, the variable that is added is the one that most improves the fit of the model. You can also specify groups of variables to treat as a unit during the selection process. An option enables you to specify the criterion for inclusion. This selection method is available in the GLMSELECT, LOGISTIC, PHREG, QUANTSELECT, and REG procedures.

BACKWARD uses a backward-elimination algorithm to select variables. This method starts with a full model and eliminates variables one by one from the model. At each step, the variable that makes the smallest contribution to the model is deleted. You can also specify groups of variables to treat as a unit during the selection process. An option enables you to specify the criterion for exclusion. This selection method is available in the GLMSELECT, LOGISTIC, PHREG, QUANTSELECT, and REG procedures.

STEPWISE uses a stepwise-regression algorithm to select variables; it combines the forward-selection and backward-elimination steps. This method is a modification of the forward-selection method in that variables already in the model do not necessarily stay there. You can also specify groups of variables to treat as a unit during the selection process. Again, options enable you to specify criteria for entering the model and for remaining in the model. This selection method is available in the GLMSELECT, LOGISTIC, PHREG, QUANTSELECT, and REG procedures.

LASSO adds and deletes parameters by using a version of ordinary least squares in which the sum of the absolute regression coefficients is constrained. If the model contains CLASS variables, then these CLASS variables are split. This selection method is available in the GLMSELECT and QUANTSELECT procedures.

LAR uses least angle regression to select variables. Like forward selection, this method starts with no effects in the model and adds effects. The parameter estimates at any step are "shrunk" when compared to the corresponding least squares estimates. If the model contains CLASS variables, then these CLASS variables are split. This selection method is available in PROC GLMSELECT.

MAXR uses maximum R-square improvement to select models. This method tries to find the best one-variable model, the best two-variable model, and so on. The MAXR method differs from the STEPWISE method in that it evaluates many more models. The MAXR method considers all possible variable exchanges before making any exchange. The STEPWISE method might remove the "worst" variable without considering what the "best" remaining variable might accomplish, whereas MAXR considers what the "best" remaining variable might accomplish. Consequently, model building based on the maximum R-square improvement usually takes much longer than stepwise model building. This selection method is available in PROC REG.

MINR uses minimum R-square improvement to select models. This method closely resembles MAXR, but the switch that is chosen is the one that produces the smallest increase in R square. This selection method is available in PROC REG.

RSQUARE selects a specified number of models that have the highest R square in each of a range of model sizes. This selection method is available in PROC REG.

CP selects a specified number of models that have the lowest C_p within a range of model sizes. This selection method is available in PROC REG.

ADJRSQ selects a specified number of models that have the highest adjusted R square within a range of model sizes. This selection method is available in PROC REG.

SCORE selects models based on a branch-and-bound algorithm that finds a specified number of models that have the highest likelihood score for all possible model sizes, from one up to a model that contains all the explanatory effects. This selection method is available in PROC LOGISTIC and PROC PHREG.

Linear Regression: The REG Procedure

PROC REG is a general-purpose procedure for linear regression that does the following:

- handles simple and multiple regression models
- provides nine model selection methods
- allows interactive changes both in the model and in the data that are used to fit the model
- allows linear equality restrictions on parameters
- tests linear hypotheses and multivariate hypotheses
- produces collinearity diagnostics, influence diagnostics, and partial regression leverage plots
- saves estimates, predicted values, residuals, confidence limits, and other diagnostic statistics in output SAS data sets
- generates plots of fit, of data, and of various statistics
- uses data, correlations, or crossproducts for input

Regression with the REG and GLM Procedures

The REG and GLM procedures are closely related; they make the same assumptions about the basic model and use the same estimation principles. Both procedures estimate parameters by ordinary or weighted least squares and assume homoscedastic, uncorrelated model errors with zero mean. An assumption of normality of the model errors is not necessary for parameter estimation, but it is implied in confirmatory inference based on the parameter estimates—that is, the computation of tests, p-values, and confidence and prediction intervals.

PROC GLM provides a CLASS statement for the levelization of classification variables; see the section "Parameterization of Model Effects" on page 412 in Chapter 20, "Shared Concepts and Topics," on the parameterization of classification variables in statistical models. In most cases, you should directly use PROC GLM, PROC GLMSELECT, or some other procedure when you fit models that have classification variables. However, you can fit models that have classification variables in PROC REG by first coding the classification variables by using PROC GLMSELECT, PROC TRANSREG, PROC GLMMOD, or some other method, and then including the coded variables in the MODEL statement in PROC REG.

Most of the statistics based on predicted and residual values that are available in PROC REG are also available in PROC GLM. However, PROC REG provides more diagnostic information. In addition, PROC GLM allows only one model and does not provide model selection.

Both PROC REG and PROC GLM are interactive, in that they do not stop after processing a RUN statement. Both procedures accept statements until a QUIT statement is submitted. For more information about interactive procedures, see the section "Interactive Features in the CATMOD, GLM, and REG Procedures" on page 90.

Model Selection: The GLMSELECT Procedure

PROC GLMSELECT performs model selection in the framework of general linear models. A variety of model selection methods are available, including forward, backward, stepwise, the LASSO method of Tibshirani (1996), and the related least angle regression method of Efron et al. (2004). The GLMSELECT procedure offers extensive capabilities for customizing the selection by providing a wide variety of selection and stopping criteria, including significance level–based and validation-based criteria. The procedure also provides graphical summaries of the selection process.

PROC GLMSELECT compares most closely with PROC REG and PROC GLM. PROC REG supports a variety of model selection methods but does not provide a CLASS statement. PROC GLM provides a CLASS statement but does not provide model selection methods. PROC GLMSELECT fills this gap. PROC GLMSELECT focuses on the standard general linear model for univariate responses with independently and identically distributed errors. PROC GLMSELECT provides results (tables, output data sets, and macro variables) that make it easy to explore the selected model in more detail in a subsequent procedure such as PROC REG or PROC GLM.

Response Surface Regression: The RSREG Procedure

PROC RSREG fits a quadratic response-surface model, which is useful in searching for factor values that optimize a response. The following features in PROC RSREG make it preferable to other regression procedures for analyzing response surfaces:

- automatic generation of quadratic effects
- a lack-of-fit test
- solutions for critical values of the surface
- eigenvalues of the associated quadratic form
- a ridge analysis to search for the direction of optimum response

Partial Least Squares Regression: The PLS Procedure

PROC PLS fits models by using any of a number of linear predictive methods, including partial least squares (PLS). Ordinary least squares regression, as implemented in the GLM or REG procedure, has the single goal of minimizing sample response prediction error by seeking linear functions of the predictors that explain as much variation in each response as possible. The techniques that are implemented in PROC PLS have the additional goal of accounting for variation in the predictors under the assumption that directions in the predictor space that are well sampled should provide better prediction for *new* observations when the predictors are highly correlated. All the techniques that are implemented in PROC PLS work by extracting successive linear combinations of the predictors, called *factors* (also called *components* or *latent vectors*), which optimally address one or both of these two goals—explaining response variation and explaining predictor variation. In particular, the method of partial least squares balances the two objectives, seeking factors that explain both response variation and predictor variation.

Generalized Linear Regression

As outlined in the section "Generalized Linear Models" on page 33 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," the class of generalized linear models generalizes the linear regression models in two ways:

- by allowing the data to come from a distribution that is a member of the exponential family of distributions
- by introducing a link function, $g(\cdot)$, that provides a mapping between the linear predictor $\eta = \mathbf{x}'\boldsymbol{\beta}$ and the mean of the data, $g(\mathrm{E}[Y]) = \eta$. The link function is monotonic, so that $\mathrm{E}[Y] = g^{-1}(\eta)$; $g^{-1}(\cdot)$ is called the inverse link function.

One of the most commonly used generalized linear regression models is the logistic model for binary or binomial data. Suppose that Y denotes a binary outcome variable that takes on the values 1 and 0 with the probabilities π and $1-\pi$, respectively. The probability π is also referred to as the "success probability," supposing that the coding $Y = 1$ corresponds to a success in a Bernoulli experiment. The success probability is also the mean of Y, and one of the aims of logistic regression analysis is to study how regressor variables affect the outcome probabilities or functions thereof, such as odds ratios.

The logistic regression model for π is defined by the linear predictor $\eta = \mathbf{x}'\boldsymbol{\beta}$ and the logit link function:

$$\mathrm{logit}(\Pr(Y = 0)) = \log\left(\frac{\pi}{1-\pi}\right) = \mathbf{x}'\boldsymbol{\beta}$$

The inversely linked linear predictor function in this model is

$$\Pr(Y = 0) = \frac{1}{1+\exp(-\eta)}$$

The dichotomous logistic regression model can be extended to multinomial (polychotomous) data. Two classes of models for multinomial data can be fit by using procedures in SAS/STAT software: models for ordinal data that rely on cumulative link functions, and models for nominal (unordered) outcomes that rely on generalized logits. The next sections briefly discuss SAS/STAT procedures for logistic regression. For more

information about the comparison of the procedures mentioned there with respect to analysis of categorical responses, see Chapter 9, "Introduction to Categorical Data Analysis Procedures."

The SAS/STAT procedures CATMOD, GENMOD, GLIMMIX, LOGISTIC, and PROBIT can fit generalized linear models for binary, binomial, and multinomial outcomes.

Contingency Table Data: The CATMOD Procedure

PROC CATMOD fits models to data that are represented by a contingency table by using weighted least squares and a variety of link functions. Although PROC CATMOD also provides maximum likelihood estimation for logistic regression, PROC LOGISTIC is more efficient for most analyses.

Generalized Linear Models: The GENMOD Procedure

PROC GENMOD is a generalized linear modeling procedure that estimates parameters by maximum likelihood. It uses CLASS and MODEL statements to form the statistical model and can fit models to binary and ordinal outcomes. The GENMOD procedure does not fit generalized logit models for nominal outcomes. However, PROC GENMOD can solve generalized estimating equations (GEE) to model correlated data and can perform a Bayesian analysis.

Generalized Linear Mixed Models: The GLIMMIX Procedure

PROC GLIMMIX fits generalized linear mixed models. If the model does not contain random effects, PROC GLIMMIX fits generalized linear models by using the method of maximum likelihood. In the class of logistic regression models, PROC GLIMMIX can fit models to binary, binomial, ordinal, and nominal outcomes.

Logistic Regression: The LOGISTIC Procedure

PROC LOGISTIC fits logistic regression models and estimates parameters by maximum likelihood. The procedure fits the usual logistic regression model for binary data in addition to models with the cumulative link function for ordinal data (such as the proportional odds model) and the generalized logit model for nominal data. PROC LOGISTIC offers a number of variable selection methods and can perform conditional and exact conditional logistic regression analysis.

Discrete Event Data: The PROBIT Procedure

PROC PROBIT calculates maximum likelihood estimates of regression parameters and the natural (or threshold) response rate for quantal response data from biological assays or other discrete event data. PROC PROBIT fits probit, logit, ordinal logistic, and extreme value (or gompit) regression models.

Correlated Data: The GENMOD and GLIMMIX Procedures

When a generalized linear model is formed by using distributions other than the binary, binomial, or multinomial distribution, you can use the GENMOD and GLIMMIX procedures for parameter estimation and inference.

Both PROC GENMOD and PROC GLIMMIX can accommodate correlated observations, but they use different techniques. PROC GENMOD fits correlated data models by using generalized estimating equations that rely on a first- and second-moment specification for the response data and a working correlation assumption. With PROC GLIMMIX, you can model correlations between the observations either by

specifying random effects in the conditional distribution that induce a marginal correlation structure or by direct modeling of the marginal dependence. PROC GLIMMIX uses likelihood-based techniques to estimate parameters.

PROC GENMOD supports a Bayesian analysis through its BAYES statement.

With PROC GLIMMIX, you can vary the distribution or link function from one observation to the next.

To fit a generalized linear model by using a distribution that is not available in the GENMOD or GLIMMIX procedure, you can use PROC NLMIXED and use SAS programming statements to code the log-likelihood function of an observation.

Ill-Conditioned Data: The ORTHOREG Procedure

PROC ORTHOREG performs linear least squares regression by using the Gentleman-Givens computational method (Gentleman 1972, 1973), and it can produce more accurate parameter estimates for ill-conditioned data. PROC GLM and PROC REG produce very accurate estimates for most problems. However, if you have very ill-conditioned data, consider using PROC ORTHOREG. The collinearity diagnostics in PROC REG can help you determine whether PROC ORTHOREG would be useful.

Quantile Regression: The QUANTREG and QUANTSELECT Procedures

PROC QUANTREG models the effects of covariates on the conditional quantiles of a response variable by using quantile regression. PROC QUANTSELECT performs quantile regression with model selection.

Ordinary least squares regression models the relationship between one or more covariates X and the conditional mean of the response variable $\mathrm{E}[Y|X = x]$. Quantile regression extends the regression model to conditional quantiles of the response variable, such as the 90th percentile. Quantile regression is particularly useful when the rate of change in the conditional quantile, expressed by the regression coefficients, depends on the quantile. An advantage of quantile regression over least squares regression is its flexibility in modeling data that have heterogeneous conditional distributions. Data of this type occur in many fields, including biomedicine, econometrics, and ecology.

Features of PROC QUANTREG include the following:

- simplex, interior point, and smoothing algorithms for estimation
- sparsity, rank, and resampling methods for confidence intervals
- asymptotic and bootstrap methods to estimate covariance and correlation matrices of the parameter estimates
- Wald and likelihood ratio tests for the regression parameter estimates
- regression quantile spline fits

The QUANTSELECT procedure shares most of its syntax and output format with PROC GLMSELECT and PROC QUANTREG. Features of PROC QUANTSELECT include the following:

- a variety of selection methods, including forward, backward, stepwise, and LASSO

- a variety of criteria, including AIC, AICC, ADJR1, SBC, significance level, testing ACL, and validation ACL
- effect selection both for individual quantile levels and for the entire quantile process
- a SAS data set that contains the design matrix
- macro variables that enable you to easily specify the selected model by using a subsequent PROC QUANTREG step

Nonlinear Regression: The NLIN and NLMIXED Procedures

Recall from Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," that a nonlinear regression model is a statistical model in which the mean function depends on the model parameters in a nonlinear function. The SAS/STAT procedures that can fit general, nonlinear models are the NLIN and NLMIXED procedures. These procedures have the following similarities:

- Nonlinear models are fit by iterative methods.
- You must provide an expression for the model by using SAS programming statements.
- Analytic derivatives of the objective function with respect to the parameters are calculated automatically.
- A grid search is available to select the best starting values for the parameters from a set of starting points that you provide.

The procedures have the following differences:

- PROC NLIN estimates parameters by using nonlinear least squares; PROC NLMIXED estimates parameters by using maximum likelihood.
- PROC NLMIXED enables you to construct nonlinear models that contain normally distributed random effects.
- PROC NLIN requires that you declare all model parameters in the PARAMETERS statement and assign starting values. PROC NLMIXED determines the parameters in your model from the PARAMETER statement and the other modeling statements. It is not necessary to supply starting values for all parameters in PROC NLMIXED, but it is highly recommended.
- The residual variance is not a parameter in models that are fit by using PROC NLIN, but it is a parameter in models that are fit by using PROC NLMIXED.
- The default iterative optimization method in PROC NLIN is the Gauss-Newton method; the default method in PROC NLMIXED is the quasi-Newton method. Other optimization techniques are available in both procedures.

Nonlinear models are fit by using iterative techniques that begin from starting values and attempt to iteratively improve on the estimates. There is no guarantee that the solution that is achieved when the iterative algorithm converges will correspond to a global optimum.

Nonparametric Regression

Parametric regression models express the mean of an observation as a function of the regressor variables $x_1, \ldots, x_k$ and the parameters $\beta_1, \ldots, \beta_p$:

$$\mathrm{E}[Y] = f(x_1, \ldots, x_k; \beta_1, \ldots, \beta_p)$$

Not only do nonparametric regression techniques relax the assumption of linearity in the regression parameters, but they also do not require that you specify a precise functional form for the relationship between response and regressor variables. Consider a regression problem in which the relationship between response Y and regressor X is to be modeled. It is assumed that $\mathrm{E}[Y_i] = g(x_i) + \epsilon_i$, where $g(\cdot)$ is an unspecified regression function. Two primary approaches in nonparametric regression modeling are as follows:

- Approximate $g(x_i)$ locally by a parametric function that is constructed from information in a local neighborhood of x_i.
- Approximate the unknown function $g(x_i)$ by a smooth, flexible function and determine the necessary smoothness and continuity properties from the data.

The SAS/STAT procedures ADAPTIVEREG, GAM, GAMPL, LOESS, and TPSPLINE fit nonparametric regression models by one of these methods.

Adaptive Regression: The ADAPTIVEREG Procedure

PROC ADAPTIVEREG fits multivariate adaptive regression splines as defined by Friedman (1991). The method is a nonparametric regression technique that combines regression splines and model selection methods. It does not assume parametric model forms and does not require specification of knot values for constructing regression spline terms. Instead, it constructs spline basis functions in an adaptive way by automatically selecting appropriate knot values for different variables and obtains reduced models by applying model selection techniques. PROC ADAPTIVEREG supports models that contain classification variables.

Generalized Additive Models: The GAM and GAMPL Procedures

Generalized additive models are nonparametric models in which one or more regressor variables are present and can make different smooth contributions to the mean function. For example, if $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{ik}]$ is a vector of k regressor for the ith observation, then an additive model represents the mean function as

$$\mathrm{E}[Y] = f_0 + f_1(x_{i1}) + f_2(x_{i2}) + \cdots + f_3(x_{i3})$$

The individual functions f_j can have a parametric or nonparametric form. If all f_j are parametric, the additive model is fully parametric. If some f_j are nonparametric, the additive model is semiparametric. Otherwise, the additive model is fully nonparametric.

The generalization of additive models is akin to the generalization of linear models: nonnormal data are accommodated by explicitly modeling the distribution of the data as a member of the exponential family and by applying a monotonic link function that provides a mapping between the predictor and the mean of the data.

PROC GAM uses smoothing splines or local regression smoothers to represent each individual function f_j. Because of the large number of parameters that are involved, PROC GAM uses the backfitting algorithm

(Hastie and Tibshirani 1990) to fit f_j by using partial residuals $y - \sum_i \hat{f}_{i:i \neq j}$ while keeping other fitted terms $\hat{f}_{i:i \neq j}$ temporarily fixed. It cycles over all smoothers until convergence.

PROC GAMPL uses low-rank regression splines (Wood 2003) to represent each individual function f_j. It reduces the number of parameters from the number of unique data points to a number that defines the upper limit of the smoothing complexity. Thus the number of parameters for regression splines is much smaller than the number of parameters for smoothing splines. PROC GAMPL also assumes an individual smoothing penalty for each f_j and then fits all additive models simultaneously.

Because of their different smoothers and general model fitting approaches, PROC GAM and PROC GAMPL do not usually yield similar results. The methods that PROC GAM uses are limited to small data sets. In contrast, you can use PROC GAMPL to efficiently analyze larger data sets.

Local Regression: The LOESS Procedure

PROC LOESS implements a local regression approach for estimating regression surfaces that was pioneered by Cleveland, Devlin, and Grosse (1988). No assumptions about the parametric form of the entire regression surface are made by PROC LOESS. Only a parametric form of the local approximation is specified by the user. Furthermore, PROC LOESS is suitable when there are outliers in the data and a robust fitting method is necessary.

Thin Plate Smoothing Splines: The TPSPLINE Procedure

PROC TPSPLINE decomposes the regressor contributions to the mean function into parametric components and into smooth functional components. Suppose that the regressor variables are collected into the vector $\mathbf{x}$ and that this vector is partitioned as $\mathbf{x} = [\mathbf{x}_1' \ \mathbf{x}_2']'$. The relationship between Y and $\mathbf{x}_2$ is linear (parametric), and the relationship between Y and $\mathbf{x}_1$ is nonparametric. PROC TPSPLINE fits models of the form

$$\mathrm{E}[Y] = g(\mathbf{x}_1) + \mathbf{x}_2'\boldsymbol{\beta}$$

The function $g(\cdot)$ can be represented as a sequence of spline basis functions.

The parameters are estimated by a penalized least squares method. The penalty is applied to the usual least squares criterion to obtain a regression estimate that fits the data well and to prevent the fit from attempting to interpolate the data (fit the data too closely).

Robust Regression: The ROBUSTREG Procedure

PROC ROBUSTREG implements algorithms to detect outliers and provide resistant (stable) results in the presence of outliers. The ROBUSTREG procedure provides four such methods: M estimation, LTS estimation, S estimation, and MM estimation.

- M estimation, which was introduced by Huber (1973), is the simplest approach both computationally and theoretically. Although it is not robust with respect to leverage points, it is still used extensively in analyzing data for which it can be assumed that the contamination is mainly in the response direction.
- Least trimmed squares (LTS) estimation is a high breakdown value method that was introduced by Rousseeuw (1984). The breakdown value is a measure of the proportion of observations that are contaminated by outliers that an estimation method can withstand and still maintain its robustness.

- S estimation is a high breakdown value method that was introduced by Rousseeuw and Yohai (1984). When the breakdown value is the same, it has a higher statistical efficiency than LTS estimation.
- MM estimation, which was introduced by Yohai (1987), combines high breakdown value estimation and M estimation. It has the high breakdown property of S estimation but a higher statistical efficiency.

For diagnostic purposes, PROC ROBUSTREG also implements robust leverage-point detection based on the robust Mahalanobis distance. The robust distance is computed by using a generalized minimum covariance determinant (MCD) algorithm.

Regression with Transformations: The TRANSREG Procedure

PROC TRANSREG fits linear models to data. In addition, PROC TRANSREG can find nonlinear transformations of the data and fit a linear model to the transformed data. This is in contrast to PROC REG and PROC GLM, which fit linear models to data, and PROC NLIN, which fits nonlinear models to data. PROC TRANSREG fits a variety of models, including the following:

- ordinary regression and ANOVA
- metric and nonmetric conjoint analysis
- metric and nonmetric vector and ideal point preference mapping
- simple, multiple, and multivariate regression with optional variable transformations
- redundancy analysis with optional variable transformations
- canonical correlation analysis with optional variable transformations
- simple and multiple regression models with a Box-Cox (1964) transformation of the dependent variable
- regression models with penalized B-splines (Eilers and Marx 1996)

Interactive Features in the CATMOD, GLM, and REG Procedures

The CATMOD, GLM, and REG procedures are interactive: they do not stop after processing a RUN statement. You can submit more statements to continue the analysis that was started by the previous statements. Both interactive and noninteractive procedures stop if a DATA step, a new procedure step, or a QUIT or ENDSAS statement is submitted. Noninteractive SAS procedures also stop if a RUN statement is submitted.

Placement of statements is critical in interactive procedures, particularly for ODS statements. For example, if you submit the following steps, no ODS OUTPUT data set is created:

```
proc reg data=sashelp.class;
   model weight = height;
run;

ods output parameterestimates=pm;
proc glm data=sashelp.class;
   class sex;
   model weight = sex | height / solution;
run;
```

You can cause the PROC GLM step to create an output data set by adding a QUIT statement to the PROC REG step or by moving the ODS OUTPUT statement so that it follows the PROC GLM statement. For more information about the placement of RUN, QUIT, and ODS statements in interactive procedures, see Example 23.6 in Chapter 23, "Using the Output Delivery System."

Statistical Background in Linear Regression

The remainder of this chapter outlines the way in which many SAS/STAT regression procedures calculate various regression quantities. The discussion focuses on the linear regression models. For general statistical background about linear statistical models, see the section "Linear Model Theory" in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software."

Exceptions and further details are documented for individual procedures.

Linear Regression Models

In matrix notation, a linear model is written as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\mathbf{X}$ is the $(n \times k)$ design matrix (rows are observations and columns are the regressors), $\boldsymbol{\beta}$ is the $(k \times 1)$ vector of unknown parameters, and $\boldsymbol{\epsilon}$ is the $(n \times 1)$ vector of unobservable model errors. The first column of $\mathbf{X}$ is usually a vector of ones and is used to estimate the intercept term.

The statistical theory of linear models is based on strict classical assumptions. Ideally, you measure the response after controlling all factors in an experimentally determined environment. If you cannot control the factors experimentally, some tests must be interpreted as being conditional on the observed values of the regressors.

Other assumptions are as follows:

- The form of the model is correct (all important explanatory variables have been included). This assumption is reflected mathematically in the assumption of a zero mean of the model errors, $\mathrm{E}[\boldsymbol{\epsilon}] = \mathbf{0}$.
- Regressor variables are measured without error.
- The expected value of the errors is 0.
- The variance of the error (and thus the dependent variable) for the ith observation is σ^2/w_i, where w_i is a known weight factor. Usually, $w_i = 1$ for all i and thus σ^2 is the common, constant variance.
- The errors are uncorrelated across observations.

When hypotheses are tested, or when confidence and prediction intervals are computed, an additional assumption is made that the errors are normally distributed.

Parameter Estimates and Associated Statistics

Least Squares Estimators

Least squares estimators of the regression parameters are found by solving the normal equations

$$\left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)\boldsymbol{\beta} = \mathbf{X}'\mathbf{W}\mathbf{Y}$$

for the vector $\boldsymbol{\beta}$, where $\mathbf{W}$ is a diagonal matrix of observed weights. The resulting estimator of the parameter vector is

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{W}\mathbf{Y}$$

This is an unbiased estimator, because

$$\begin{aligned}\mathrm{E}\left[\widehat{\boldsymbol{\beta}}\right] &= \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{W}\mathrm{E}[\mathbf{Y}] \\ &= \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{W}\mathbf{X}\boldsymbol{\beta} = \boldsymbol{\beta}\end{aligned}$$

Notice that the only assumption necessary in order for the least squares estimators to be unbiased is that of a zero mean of the model errors. If the estimator is evaluated at the observed data, it is referred to as the least squares estimate,

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{W}\mathbf{Y}$$

If the standard classical assumptions are met, the least squares estimators of the regression parameters are the best linear unbiased estimators (BLUE). In other words, the estimators have minimum variance in the class of estimators that are unbiased and that are linear functions of the responses. If the additional assumption of normally distributed errors is satisfied, then the following are true:

- The statistics that are computed have the proper sampling distributions for hypothesis testing.
- Parameter estimators are normally distributed.
- Various sums of squares are distributed proportional to chi-square, at least under proper hypotheses.
- Ratios of estimators to standard errors follow the Student's t distribution under certain hypotheses.
- Appropriate ratios of sums of squares follow an F distribution for certain hypotheses.

When regression analysis is used to model data that do not meet the assumptions, you should interpret the results in a cautious, exploratory fashion. The significance probabilities under these circumstances are unreliable.

Box (1966) and Mosteller and Tukey (1977, Chapters 12 and 13) discuss the problems that are encountered in regression data, especially when the data are not under experimental control.

Estimating the Precision

Assume for the present that $\mathbf{X}'\mathbf{W}\mathbf{X}$ has full column rank k (this assumption is relaxed later). The variance of the error terms, $\mathrm{Var}[\epsilon_i] = \sigma^2$, is then estimated by the mean square error,

$$s^2 = \mathrm{MSE} = \frac{\mathrm{SSE}}{n-k} = \frac{1}{n-k}\sum_{i=1}^{n} w_i \left(y_i - \mathbf{x}_i'\boldsymbol{\beta}\right)^2$$

where $\mathbf{x}_i'$ is the ith row of the design matrix $\mathbf{X}$. The residual variance estimate is also unbiased: $\mathrm{E}[s^2] = \sigma^2$.

The covariance matrix of the least squares estimators is

$$\mathrm{Var}\left[\widehat{\boldsymbol{\beta}}\right] = \sigma^2 \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1}$$

An estimate of the covariance matrix is obtained by replacing σ^2 with its estimate, s^2 in the preceding formula. This estimate is often referred to as COVB in SAS/STAT modeling procedures:

$$\mathrm{COVB} = \widehat{\mathrm{Var}}\left[\widehat{\boldsymbol{\beta}}\right] = s^2 \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1}$$

The correlation matrix of the estimates, often referred to as CORRB, is derived by scaling the covariance matrix: Let $\mathbf{S} = \mathrm{diag}\left(\left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1}\right)^{-\frac{1}{2}}$. Then the correlation matrix of the estimates is

$$\mathrm{CORRB} = \mathbf{S}\left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1}\mathbf{S}$$

The estimated standard error of the ith parameter estimator is obtained as the square root of the ith diagonal element of the COVB matrix. Formally,

$$\mathrm{STDERR}\left(\widehat{\boldsymbol{\beta}}_i\right) = \sqrt{\left[s^2(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\right]_{ii}}$$

The ratio

$$t = \frac{\widehat{\boldsymbol{\beta}}_i}{\mathrm{STDERR}\left(\widehat{\boldsymbol{\beta}}_i\right)}$$

follows a Student's t distribution with $(n-k)$ degrees of freedom under the hypothesis that β_i is zero and provided that the model errors are normally distributed.

Regression procedures display the t ratio and the significance probability, which is the probability under the hypothesis $H: \beta_i = 0$ of a larger absolute t value than was actually obtained. When the probability is less than some small level, the event is considered so unlikely that the hypothesis is rejected.

Type I SS and Type II SS measure the contribution of a variable to the reduction in SSE. Type I SS measure the reduction in SSE as that variable is entered into the model in sequence. Type II SS are the increment in SSE that results from removing the variable from the full model. Type II SS are equivalent to the Type III and Type IV SS that are reported in PROC GLM. If Type II SS are used in the numerator of an F test, the test is equivalent to the t test for the hypothesis that the parameter is zero. In polynomial models, Type I SS measure the contribution of each polynomial term after it is orthogonalized to the previous terms in the model. The four types of SS are described in Chapter 16, "The Four Types of Estimable Functions."

Coefficient of Determination

The coefficient of determination in a regression model, also known as the R-square statistic, measures the proportion of variability in the response that is explained by the regressor variables. In a linear regression model with intercept, R square is defined as

$$R^2 = 1 - \frac{\mathrm{SSE}}{\mathrm{SST}}$$

where SSE is the residual (error) sum of squares and SST is the total sum of squares corrected for the mean. The adjusted R-square statistic is an alternative to R square that takes into account the number of parameters in the model. The adjusted R-square statistic is calculated as

$$\mathrm{ADJRSQ} = 1 - \frac{n-i}{n-p}\left(1 - R^2\right)$$

where n is the number of observations that are used to fit the model, p is the number of parameters in the model (including the intercept), and i is 1 if the model includes an intercept term, and 0 otherwise.

R-square statistics also play an important indirect role in regression calculations. For example, the proportion of variability that is explained by regressing all other variables in a model on a particular regressor can provide insights into the interrelationship among the regressors.

Tolerances and variance inflation factors measure the strength of interrelationships among the regressor variables in the model. If all variables are orthogonal to each other, both tolerance and variance inflation are 1. If a variable is very closely related to other variables, the tolerance approaches 0 and the variance inflation gets very large. Tolerance (TOL) is 1 minus the R square that results from the regression of the other variables in the model on that regressor. Variance inflation (VIF) is the diagonal of $(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}$, if $(\mathbf{X}'\mathbf{W}\mathbf{X})$ is scaled to correlation form. The statistics are related as

$$\mathrm{VIF} = \frac{1}{\mathrm{TOL}}$$

Explicit and Implicit Intercepts

A linear model contains an *explicit* intercept if the **X** matrix contains a column whose nonzero values do not vary, usually a column of ones. Many SAS/STAT procedures automatically add this column of ones as the first column in the **X** matrix. Procedures that support a NOINT option in the MODEL statement provide the capability to suppress the automatic addition of the intercept column.

In general, models without an intercept should be the exception, especially if your model does not contain classification (CLASS) variables. An overall intercept is provided in many models to adjust for the grand total or overall mean in your data. A simple linear regression without intercept, such as

$$\mathrm{E}[Y_i] = \beta_1 x_i + \epsilon_i$$

assumes that Y has mean zero if X takes on the value zero. This might not be a reasonable assumption.

If you explicitly suppress the intercept in a statistical model, the calculation and interpretation of your results can change. For example, the exclusion of the intercept in the following PROC REG statements leads to a different calculation of the R-square statistic. It also affects the calculation of the sum of squares in the analysis of variance for the model. For example, the model and error sum of squares add up to the uncorrected total sum of squares in the absence of an intercept.

```
proc reg;
   model y = x / noint;
quit;
```

Many statistical models contain an *implicit* intercept, which occurs when a linear function of one or more columns in the **X** matrix produces a column of constant, nonzero values. For example, the presence of a CLASS variable in the GLM parameterization always implies an intercept in the model. If a model contains an implicit intercept, then adding an intercept to the model does not alter the quality of the model fit, but it changes the interpretation (and number) of the parameter estimates.

The way in which the implicit intercept is detected and accounted for in the analysis depends on the procedure. For example, the following statements in PROC GLM lead to an implied intercept:

```
proc glm;
   class a;
   model y = a / solution noint;
run;
```

Whereas the analysis of variance table uses the uncorrected total sum of squares (because of the NOINT option), the implied intercept does not lead to a redefinition or recalculation of the R-square statistic (compared to the model without the NOINT option). Also, because the intercept is implied by the presence of the CLASS variable a in the model, the same error sum of squares results whether the NOINT option is specified or not.

A different approach is taken, for example, by PROC TRANSREG. The ZERO=NONE option in the CLASS parameterization of the following statements leads to an implicit intercept model:

```
proc transreg;
   model ide(y) = class(a / zero=none) / ss2;
run;
```

The analysis of variance table or the regression fit statistics are not affected in PROC TRANSREG. Only the interpretation of the parameter estimates changes because of the way in which the intercept is accounted for in the model.

Implied intercepts not only occur when classification effects are present in the model. They also occur with B-splines and other sets of constructed columns.

Models Not of Full Rank

If the **X** matrix is not of full rank, then a generalized inverse can be used to solve the normal equations to minimize the SSE:

$$\widehat{\boldsymbol{\beta}} = \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{W}\mathbf{Y}$$

However, these estimates are not unique, because there are an infinite number of solutions that correspond to different generalized inverses. PROC REG and other regression procedures choose a nonzero solution for all variables that are linearly independent of previous variables and a zero solution for other variables. This approach corresponds to using a generalized inverse in the normal equations, and the expected values of the estimates are the Hermite normal form of $\mathbf{X}'\mathbf{W}\mathbf{X}$ multiplied by the true parameters:

$$\mathrm{E}\left[\widehat{\boldsymbol{\beta}}\right] = \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-}\left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)\boldsymbol{\beta}$$

Degrees of freedom for the estimates that correspond to singularities are not counted (reported as zero). The hypotheses that are not testable have t tests displayed as missing. The message that the model is not of full rank includes a display of the relations that exist in the matrix.

For information about generalized inverses and their importance for statistical inference in linear models, see the sections "Generalized Inverse Matrices" and "Linear Model Theory" in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software."

Predicted and Residual Values

After the model has been fit, predicted and residual values are usually calculated, graphed, and output. The predicted values are calculated from the estimated regression equation; the raw residuals are calculated as the observed value minus the predicted value. Often other forms of residuals, such as studentized or cumulative residuals, are used for model diagnostics. Some procedures can calculate standard errors of residuals, predicted mean values, and individual predicted values.

Consider the ith observation, where $\mathbf{x}_i'$ is the row of regressors, $\widehat{\boldsymbol{\beta}}$ is the vector of parameter estimates, and s^2 is the estimate of the residual variance (the mean squared error). The *leverage* value of the ith observation is defined as

$$h_i = w_i \mathbf{x}_i' (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1} \mathbf{x}_i$$

where $\mathbf{X}$ is the design matrix for the observed data, $\mathbf{x}_i'$ is an arbitrary regressor vector (possibly but not necessarily a row of $\mathbf{X}$), $\mathbf{W}$ is a diagonal matrix of observed weights, and w_i is the weight corresponding to $\mathbf{x}_i'$.

Then the predicted mean and the standard error of the predicted mean are

$$\begin{aligned} \widehat{y}_i &= \mathbf{x}_i' \widehat{\boldsymbol{\beta}} \\ \mathrm{STDERR}(\widehat{y}_i) &= \sqrt{s^2 h_i / w_i} \end{aligned}$$

The standard error of the individual (future) predicted value y_i is

$$\mathrm{STDERR}(y_i) = \sqrt{s^2(1 + h_i)/w_i}$$

If the predictor vector $\mathbf{x}_i$ corresponds to an observation in the analysis data, then the raw residual for that observation and the standard error of the raw residual are defined as

$$\begin{aligned} \mathrm{RESID}_i &= y_i - \mathbf{x}_i' \widehat{\boldsymbol{\beta}} \\ \mathrm{STDERR}(\mathrm{RESID}_i) &= \sqrt{s^2(1 - h_i)/w_i} \end{aligned}$$

The *studentized residual* is the ratio of the raw residual and its estimated standard error. Symbolically,

$$\mathrm{STUDENT}_i = \frac{\mathrm{RESID}_i}{\mathrm{STDERR}(\mathrm{RESID}_i)}$$

Two types of intervals provide a measure of confidence for prediction: the *confidence* interval for the mean value of the response, and the *prediction* (or *forecasting*) interval for an individual observation. As discussed in the section "Mean Squared Error" in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT

Software," both intervals are based on the mean squared error of predicting a target based on the result of the model fit. The difference in the expressions for the confidence interval and the prediction interval occurs because the target of estimation is a constant in the case of the confidence interval (the mean of an observation) and the target is a random variable in the case of the prediction interval (a new observation).

For example, you can construct a confidence interval for the ith observation that contains the true mean value of the response with probability $1-\alpha$. The upper and lower limits of the confidence interval for the mean value are

$$\text{LowerM} = \mathbf{x}_i'\widehat{\boldsymbol{\beta}} - t_{\alpha/2,\nu}\sqrt{s^2 h_i/w_i}$$
$$\text{UpperM} = \mathbf{x}_i'\widehat{\boldsymbol{\beta}} + t_{\alpha/2,\nu}\sqrt{s^2 h_i/w_i}$$

where $t_{\alpha/2,\nu}$ is the tabulated t quantile with degrees of freedom equal to the degrees of freedom for the mean squared error, $\nu = n - \text{rank}(\mathbf{X})$.

The limits for the prediction interval for an individual response are

$$\text{LowerI} = \mathbf{x}_i'\widehat{\boldsymbol{\beta}} - t_{\alpha/2,\nu}\sqrt{s^2(1+h_i)/w_i}$$
$$\text{UpperI} = \mathbf{x}_i'\widehat{\boldsymbol{\beta}} + t_{\alpha/2,\nu}\sqrt{s^2(1+h_i)/w_i}$$

Influential observations are those that, according to various criteria, appear to have a large influence on the analysis. One measure of influence, Cook's D, measures the change to the estimates that results from deleting an observation,

$$\text{COOKD}_i = \frac{1}{k}\text{STUDENT}_i^2\left(\frac{\text{STDERR}(\widehat{y}_i)}{\text{STDERR}(\text{RESID}_i)}\right)^2$$

where k is the number of parameters in the model (including the intercept). For more information, see Cook (1977, 1979).

The *predicted residual* for observation i is defined as the residual for the ith observation that results from dropping the ith observation from the parameter estimates. The sum of squares of predicted residual errors is called the *PRESS statistic*:

$$\text{PRESID}_i = \frac{\text{RESID}_i}{1-h_i}$$
$$\text{PRESS} = \sum_{i=1}^{n} w_i \text{PRESID}_i^2$$

Testing Linear Hypotheses

Testing of linear hypotheses based on estimable functions is discussed in the section "Test of Hypotheses" on page 60 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," and the construction of special sets of estimable functions corresponding to Type I–Type IV hypotheses is discussed in Chapter 16, "The Four Types of Estimable Functions." In linear regression models, testing of general linear hypotheses follows along the same lines. Test statistics are usually formed based on sums of squares that are associated with the hypothesis in question. Furthermore, when $\mathbf{X}$ is of full rank—as is the case in many regression models—the consistency of the linear hypothesis is guaranteed.

Recall from Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," that the general form of a linear hypothesis for the parameters is $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{d}$, where $\mathbf{L}$ is $q \times k$, $\boldsymbol{\beta}$ is $k \times 1$, and $\mathbf{d}$ is $q \times 1$. To test this hypothesis, you take the linear function with respect to the parameter estimates: $\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d}$. This linear function in $\widehat{\boldsymbol{\beta}}$ has variance

$$\mathrm{Var}\left[\mathbf{L}\widehat{\boldsymbol{\beta}}\right] = \mathbf{L}\mathrm{Var}[\widehat{\boldsymbol{\beta}}]\mathbf{L}' = \sigma^2\mathbf{L}\left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-}\mathbf{L}'$$

The *sum of squares due to the hypothesis* is a simple quadratic form:

$$\mathrm{SS}(H) = \mathrm{SS}(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d}) = \left(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d}\right)'\left(\mathbf{L}\left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-}\mathbf{L}'\right)^{-1}\left(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{d}\right)$$

If this hypothesis is testable, then $\mathrm{SS}(H)$ can be used in the numerator of an F statistic:

$$F = \frac{\mathrm{SS}(H)/q}{s^2} = \frac{\mathrm{SS}(\mathbf{L}\mathbf{b} - \mathbf{d})/q}{s^2}$$

If $\widehat{\boldsymbol{\beta}}$ is normally distributed, which follows as a consequence of normally distributed model errors, then this statistic follows an F distribution with q numerator degrees of freedom and $n - \mathrm{rank}(\mathbf{X})$ denominator degrees of freedom. Note that it was assumed in this derivation that $\mathbf{L}$ is of full row rank q.

Multivariate Tests

Multivariate hypotheses involve several dependent variables in the form

$$H\colon \mathbf{L}\boldsymbol{\beta}\mathbf{M} = \mathbf{d}$$

where $\mathbf{L}$ is a linear function on the regressor side, $\boldsymbol{\beta}$ is a matrix of parameters, $\mathbf{M}$ is a linear function on the dependent side, and $\mathbf{d}$ is a matrix of constants. The special case (handled by PROC REG) in which the constants are the same for each dependent variable is expressed as

$$(\mathbf{L}\boldsymbol{\beta} - \mathbf{c}\mathbf{j})\mathbf{M} = \mathbf{0}$$

where $\mathbf{c}$ is a column vector of constants and $\mathbf{j}$ is a row vector of ones. The special case in which the constants are 0 is then

$$\mathbf{L}\boldsymbol{\beta}\mathbf{M} = \mathbf{0}$$

These multivariate tests are covered in detail in Morrison (2004); Timm (2002); Mardia, Kent, and Bibby (1979); Bock (1975); and other works cited in Chapter 10, "Introduction to Multivariate Procedures."

Notice that in contrast to the tests discussed in the preceding section, $\boldsymbol{\beta}$ here is a matrix of parameter estimates. Suppose that the matrix of estimates is denoted as $\mathbf{B}$. To test the multivariate hypothesis, construct two matrices, $\mathbf{H}$ and $\mathbf{E}$, that correspond to the numerator and denominator of a univariate F test:

$$\mathbf{H} = \mathbf{M}'(\mathbf{LB} - \mathbf{cj})'(\mathbf{L}(\mathbf{X}'\mathbf{WX})^{-}\mathbf{L}')^{-1}(\mathbf{LB} - \mathbf{cj})\mathbf{M}$$
$$\mathbf{E} = \mathbf{M}'\left(\mathbf{Y}'\mathbf{WY} - \mathbf{B}'(\mathbf{X}'\mathbf{WX})\mathbf{B}\right)\mathbf{M}$$

Four test statistics, based on the eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$ or $(\mathbf{E}+\mathbf{H})^{-1}\mathbf{H}$, are formed. Let λ_i be the ordered eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$ (if the inverse exists), and let ξ_i be the ordered eigenvalues of $(\mathbf{E}+\mathbf{H})^{-1}\mathbf{H}$. It happens that $\xi_i = \lambda_i/(1+\lambda_i)$ and $\lambda_i = \xi_i/(1-\xi_i)$, and it turns out that $\rho_i = \sqrt{\xi_i}$ is the ith canonical correlation.

Let p be the rank of $(\mathbf{H}+\mathbf{E})$, which is less than or equal to the number of columns of $\mathbf{M}$. Let q be the rank of $\mathbf{L}(\mathbf{X}'\mathbf{WX})^{-}\mathbf{L}'$. Let v be the error degrees of freedom, and let $s = \min(p,q)$. Let $m = (|p-q|-1)/2$, and let $n = (v-p-1)/2$. Then the following statistics test the multivariate hypothesis in various ways, and their p-values can be approximated by F distributions. Note that in the special case that the rank of $\mathbf{H}$ is 1, all these F statistics are the same and the corresponding p-values are exact, because in this case the hypothesis is really univariate.

Wilks' Lambda

If

$$\Lambda = \frac{\det(\mathbf{E})}{\det(\mathbf{H}+\mathbf{E})} = \prod_{i=1}^{n}\frac{1}{1+\lambda_i} = \prod_{i=1}^{n}(1-\xi_i)$$

then

$$F = \frac{1-\Lambda^{1/t}}{\Lambda^{1/t}}\cdot\frac{rt-2u}{pq}$$

is approximately F distributed, where

$$r = v - \frac{p-q+1}{2}$$
$$u = \frac{pq-2}{4}$$
$$t = \begin{cases} \sqrt{\frac{p^2q^2-4}{p^2+q^2-5}} & \text{if } p^2+q^2-5>0 \\ 1 & \text{otherwise} \end{cases}$$

The degrees of freedom are pq and $rt-2u$. The distribution is exact if $\min(p,q) \le 2$. (See Rao 1973, p. 556.)

Pillai's Trace

If

$$V = \operatorname{trace}\left(\mathbf{H}(\mathbf{H}+\mathbf{E})^{-1}\right) = \sum_{i=1}^{n}\frac{\lambda_i}{1+\lambda_i} = \sum_{i=1}^{n}\xi_i$$

then

$$F = \frac{2n+s+1}{2m+s+1}\cdot\frac{V}{s-V}$$

is approximately F distributed with $s(2m+s+1)$ and $s(2n+s+1)$ degrees of freedom.

Hotelling-Lawley Trace

If

$$U = \text{trace}\left(\mathbf{E}^{-1}\mathbf{H}\right) = \sum_{i=1}^{n} \lambda_i = \sum_{i=1}^{n} \frac{\xi_i}{1-\xi_i}$$

then for $n > 0$

$$F = (U/c)((4 + (pq+2)/(b-1))/(pq))$$

is approximately F distributed with pq and $4 + (pq+2)/(b-1)$ degrees of freedom, where $b = (p + 2n)(q+2n)/(2(2n+1)(n-1))$ and $c = (2 + (pq+2)/(b-1))/(2n)$; while for $n \leq 0$

$$F = \frac{2(sn+1)U}{s^2(2m+s+1)}$$

is approximately F with $s(2m+s+1)$ and $2(sn+1)$ degrees of freedom.

Roy's Maximum Root

If $\Theta = \lambda_1$, then

$$F = \Theta\frac{v-r+q}{r}$$

where $r = \max(p,q)$ is an upper bound on F that yields a lower bound on the significance level. Degrees of freedom are r for the numerator and $v - r + q$ for the denominator.

Tables of critical values for these statistics are found in Pillai (1960).

Exact Multivariate Tests

If you specify the MSTAT=EXACT option in the appropriate statement, p-values for three of the four tests (Wilks' lambda, the Hotelling-Lawley trace, and Roy's greatest root) are computed exactly, and the p-values for the fourth test (Pillai's trace) are based on an F approximation that is more accurate (but occasionally slightly more liberal) than the default. The exact p-values for Roy's greatest root benefit the most, because in this case the F approximation provides only a lower bound for the p-value. If you use the F-based p-value for this test in the usual way, declaring a test significant if $p < 0.05$, then your decisions might be very liberal. For example, instead of the nominal 5% Type I error rate, such a procedure can easily have an actual Type I error rate in excess of 30%. By contrast, basing such a procedure on the exact p-values results in the appropriate 5% Type I error rate, under the usual regression assumptions.

The MSTAT=EXACT option is supported in the ANOVA, CANCORR, CANDISC, GLM, and REG procedures.

The exact p-values are based on the following sources:

- Wilks' lambda: Lee (1972); Davis (1979)
- Pillai's trace: Muller (1998)
- Hotelling-Lawley trace: Davis (1970, 1980)

- Roy's greatest root: Davis (1972); Pillai and Flury (1984)

Note that, although the MSTAT=EXACT *p*-value for Pillai's trace is still approximate, it has "substantially greater accuracy" than the default approximation (Muller 1998).

Because most of the MSTAT=EXACT *p*-values are not based on the *F* distribution, the columns in the multivariate tests table that correspond to this approximation—in particular, the *F* value and the numerator and denominator degrees of freedom—are no longer displayed, and the column that contains the *p*-values is labeled "P Value" instead of "Pr > F." Suppose, for example, that you use the following PROC ANOVA statements to perform a multivariate analysis of an archaeological data set:

```
data Skulls;
   input Loc $20. Basal Occ Max;
   datalines;
Minas Graes, Brazil  2.068 2.070 1.580
Minas Graes, Brazil  2.068 2.074 1.602
Minas Graes, Brazil  2.090 2.090 1.613
Minas Graes, Brazil  2.097 2.093 1.613
Minas Graes, Brazil  2.117 2.125 1.663
Minas Graes, Brazil  2.140 2.146 1.681
Matto Grosso, Brazil 2.045 2.054 1.580
Matto Grosso, Brazil 2.076 2.088 1.602
Matto Grosso, Brazil 2.090 2.093 1.643
Matto Grosso, Brazil 2.111 2.114 1.643
Santa Cruz, Bolivia  2.093 2.098 1.653
Santa Cruz, Bolivia  2.100 2.106 1.623
Santa Cruz, Bolivia  2.104 2.101 1.653
;

proc anova data=Skulls;
   class Loc;
   model Basal Occ Max = Loc / nouni;
   manova h=Loc;
   ods select MultStat;
run;
```

The default multivariate tests, based on the *F* approximations, are shown in Figure 4.5

Figure 4.5 Default Multivariate Tests

The ANOVA Procedure
Multivariate Analysis of Variance

MANOVA Test Criteria and F Approximations for the Hypothesis of No Overall Loc Effect
H = Anova SSCP Matrix for Loc
E = Error SSCP Matrix

S=2 M=0 N=3

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.60143661	0.77	6	16	0.6032
Pillai's Trace	0.44702843	0.86	6	18	0.5397
Hotelling-Lawley Trace	0.58210348	0.75	6	9.0909	0.6272
Roy's Greatest Root	0.35530890	1.07	3	9	0.4109

NOTE: F Statistic for Roy's Greatest Root is an upper bound.
NOTE: F Statistic for Wilks' Lambda is exact.

If you specify MSTAT=EXACT in the MANOVA statement, as in the following statements, then the displayed output is the much simpler table shown in Figure 4.6:

```
proc anova data=Skulls;
   class Loc;
   model Basal Occ Max = Loc / nouni;
   manova h=Loc / mstat=exact;
   ods select MultStat;
run;
```

Figure 4.6 Multivariate Tests with MSTAT=EXACT

The ANOVA Procedure
Multivariate Analysis of Variance

MANOVA Tests for the Hypothesis of No Overall Loc Effect
H = Anova SSCP Matrix for Loc
E = Error SSCP Matrix

S=2 M=0 N=3

Statistic	Value	P-Value
Wilks' Lambda	0.60143661	0.6032
Pillai's Trace	0.44702843	0.5521
Hotelling-Lawley Trace	0.58210348	0.6337
Roy's Greatest Root	0.35530890	0.7641

Notice that the *p*-value for Roy's greatest root is substantially larger in the new table and correspondingly more in line with the *p*-values for the other tests.

If you reference the underlying ODS output object for the table of multivariate statistics, it is important to note that its structure does not depend on the value of the MSTAT= option. In particular, it always contains columns that correspond to both the default MSTAT=FAPPROX and the MSTAT=EXACT tests. Moreover, because the MSTAT=FAPPROX tests are relatively cheap to compute, the columns that correspond to them are always filled in, even though they are not displayed when you specify MSTAT=EXACT. On the other hand, for MSTAT=FAPPROX (which is the default), the column of exact *p*-values contains missing values and is not displayed.

Comments on Interpreting Regression Statistics

In most applications, regression models are merely useful approximations. Reality is often so complicated that you cannot know what the true model is. You might have to choose a model more on the basis of what variables can be measured and what kinds of models can be estimated than on a rigorous theory that explains how the universe really works. However, even in cases where theory is lacking, a regression model can be an excellent predictor of the response if the model is carefully formulated from a large sample. The interpretation of statistics such as parameter estimates might nevertheless be highly problematic.

Statisticians usually use the word "prediction" in a technical sense. Prediction in this sense does not refer to "predicting the future" (statisticians call that *forecasting*) but rather to guessing the response from the values of the regressors in an observation taken under the same circumstances as the sample from which the regression equation was estimated. If you developed a regression model for predicting consumer preferences in 1997, it might not give very good predictions in 2017 no matter how well it did in 1997. If you want

to predict the future, your model must include whatever relevant factors might change over time. If the process that you are studying does in fact change over time, you must take observations at several, perhaps many, different times. Analysis of such data is the province of SAS/STAT procedures such as MIXED and GLIMMIX and SAS/ETS procedures such as AUTOREG and STATESPACE. For more information about modeling serial correlation in longitudinal, repeated measures, or time series data with SAS/STAT mixed modeling procedures, see Chapter 52, "The GLIMMIX Procedure," and Chapter 84, "The MIXED Procedure." For more information about the AUTOREG and STATESPACE procedures, see the *SAS/ETS User's Guide*.

The comments in the rest of this section are directed toward linear least squares regression. For more detailed discussions of the interpretation of regression statistics, see Darlington (1968); Mosteller and Tukey (1977); Weisberg (1985); Younger (1979).

Interpreting Parameter Estimates from a Controlled Experiment

Parameter estimates are easiest to interpret in a controlled experiment in which the regressors are manipulated independently of one another. In a well-designed experiment, such as a randomized factorial design with replications in each cell, you can use lack-of-fit tests and estimates of the standard error of prediction to determine whether the model describes the experimental process with adequate precision. If so, a regression coefficient estimates the amount by which the mean response changes when the regressor is changed by one unit while all the other regressors are unchanged. However, if the model involves interactions or polynomial terms, it might not be possible to interpret individual regression coefficients. For example, if the equation includes both linear and quadratic terms for a given variable, you cannot change the value of the linear term without also changing the value of the quadratic term. Sometimes it might be possible to recode the regressors, such as by using orthogonal polynomials, to simplify the interpretation.

If the nonstatistical aspects of the experiment are also treated with sufficient care (such as by using placebos and double-blind procedures), then you can state conclusions in causal terms; that is, this change in a regressor causes that change in the response. Causality can never be inferred from statistical results alone or from an observational study.

If the model that you fit is not the true model, then the parameter estimates can depend strongly on the particular values of the regressors that are used in the experiment. For example, if the response is actually a quadratic function of a regressor but you fit a linear function, then the estimated slope can be a large negative value if you use only small values of the regressor, a large positive value if you use only large values of the regressor, or near zero if you use both large and small regressor values. When you report the results of an experiment, it is important to include the values of the regressors. It is also important to avoid extrapolating the regression equation outside the range of regressors in the sample.

Interpreting Parameter Estimates from an Observational Study

In an observational study, parameter estimates can be interpreted as the expected difference in response of two observations that differ by one unit on the regressor in question and that have the same values for all other regressors. You cannot make inferences about "changes" in an observational study because you have not actually changed anything. It might not be possible even in principle to change one regressor independently of all the others. Nor can you draw conclusions about causality without experimental manipulation.

If you conduct an observational study and you do not know the true form of the model, interpretation of parameter estimates becomes even more convoluted. A coefficient must then be interpreted as an average over the sampled population of expected differences in response of observations that differ by one unit on only one regressor. The considerations that are discussed under controlled experiments for which the true model is not known also apply.

Comparing Parameter Estimates

Two coefficients in the same model can be directly compared only if the regressors are measured in the same units. You can make any coefficient large or small just by changing the units. For example, if you convert a regressor from feet to miles, the parameter estimate is multiplied by 5,280.

Sometimes standardized regression coefficients are used to compare the effects of regressors that are measured in different units. Standardized estimates are defined as the estimates that result when all variables are standardized to a mean of 0 and a variance of 1. Standardized estimates are computed by multiplying the original estimates by the sample standard deviation of the regressor variable and dividing by the sample standard deviation of the dependent variable.

Standardizing the variables makes the standard deviation the unit of measurement. This makes sense only if the standard deviation is a meaningful quantity, which usually is the case only if the observations are sampled from a well-defined population. In a controlled experiment, the standard deviation of a regressor depends on the values of the regressor that are selected by the experimenter. Thus, you can make a standardized regression coefficient large by using a large range of values for the regressor.

In some applications you might be able to compare regression coefficients in terms of the practical range of variation of a regressor. Suppose that each independent variable in an industrial process can be set to values only within a certain range. You can rescale the variables so that the smallest possible value is 0 and the largest possible value is 1. Then the unit of measurement for each regressor is the maximum possible range of the regressor, and the parameter estimates are comparable in that sense. Another possibility is to scale the regressors in terms of the cost of setting a regressor to a particular value so that comparisons can be made in monetary terms.

Correlated Regressors

In an experiment, you can often select values for the regressors such that the regressors are orthogonal (not correlated with one another). Orthogonal designs have enormous advantages in interpretation. With orthogonal regressors, the parameter estimate for a given regressor does not depend on which other regressors are included in the model, although other statistics such as standard errors and p-values might change.

If the regressors are correlated, it becomes difficult to disentangle the effects of one regressor from another, and the parameter estimates can be highly dependent on which regressors are used in the model. Two correlated regressors might be nonsignificant when tested separately but highly significant when considered together. If two regressors have a correlation of 1.0, it is impossible to separate their effects.

It might be possible to recode correlated regressors to make interpretation easier. For example, if X and Y are highly correlated, they could be replaced in a linear regression by $X + Y$ and $X - Y$ without changing the fit of the model or statistics for other regressors.

Errors in the Regressors

If the measurements of the regressors contain errors, the parameter estimates must be interpreted with respect to the measured values of the regressors, not the true values. A regressor might be statistically nonsignificant when measured with errors even though it would have been highly significant if measured accurately.

Probability Values (p-Values)

Probability values (p-values) do not necessarily measure the importance of a regressor. An important regressor can have a large (nonsignificant) p-value if the sample is small, if the regressor is measured over a narrow range, if there are large measurement errors, or if another closely related regressor is included in the equation. An unimportant regressor can have a very small p-value in a large sample. Computing a confidence interval for a parameter estimate gives you more useful information than just looking at the p-value, but confidence intervals do not solve problems of measurement errors in the regressors or highly correlated regressors.

Interpreting R Square

R square is usually defined as the proportion of variance of the response that is predictable from (can be explained by) the regressor variables. It might be easier to interpret $\sqrt{1 - R^2}$, which is approximately the factor by which the standard error of prediction is reduced by the introduction of the regressor variables.

R square is easiest to interpret when the observations, including the values of both the regressors and the response, are randomly sampled from a well-defined population. Nonrandom sampling can greatly distort the R square. For example, excessively large values of R square can be obtained by omitting from the sample any observations that have regressor values near the mean.

In a controlled experiment, the R square depends on the values that are chosen for the regressors. A wide range of regressor values generally yields a larger R square than a narrow range. In comparing the results of two experiments on the same variables but with different ranges for the regressors, you should look at the standard error of prediction (root mean square error) rather than the R square.

Whether a given R-square value is considered to be large or small depends on the context of the particular study. A social scientist might consider an R square of 0.30 to be large, whereas a physicist might consider an R square of 0.98 to be small.

You can always get an R square arbitrarily close to 1.0 by including a large number of completely unrelated regressors in the equation. If the number of regressors is close to the sample size, the R square is very biased. In such cases, the adjusted R square and related statistics discussed by Darlington (1968) are less misleading.

If you fit many different models and choose the model that has the largest R square, all the statistics are biased and the p-values for the parameter estimates are not valid. You must use caution in interpreting the R square for models that have no intercept term. As a general rule, no-intercept models should be fit only when theoretical justification exists and the data appear to fit a no-intercept framework. R square in those cases is measuring something different (see Kvalseth 1985).

Incorrect Data Values

All regression statistics can be seriously distorted by a single incorrect data value. A decimal point in the wrong place can completely change the parameter estimates, the R square, and other statistics. It is important to check your data for outliers and influential observations. Residual and influence diagnostics are particularly useful in this regard.

When a data point is declared as influential or as an outlier as measured by a particular model diagnostic, this does not imply that the case should be excluded from the analysis. The label "outlier" does not have a negative connotation. It means that the data point is unusual with respect to the model at hand. If your data follow a strong curved trend and you fit a linear regression, then many data points might be labeled as outliers not because they are "bad" or incorrect data values but because your model is not appropriate.

References

Allen, D. M. (1971). "Mean Square Error of Prediction as a Criterion for Selecting Variables." *Technometrics* 13:469–475.

Allen, D. M., and Cady, F. B. (1982). *Analyzing Experimental Data by Regression*. Belmont, CA: Lifetime Learning Publications.

Belsley, D. A., Kuh, E., and Welsch, R. E. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. New York: John Wiley & Sons.

Bock, R. D. (1975). *Multivariate Statistical Methods in Behavioral Research*. New York: McGraw-Hill.

Box, G. E. P. (1966). "The Use and Abuse of Regression." *Technometrics* 8:625–629.

Box, G. E. P., and Cox, D. R. (1964). "An Analysis of Transformations." *Journal of the Royal Statistical Society, Series B* 26:211–234.

Cleveland, W. S., Devlin, S. J., and Grosse, E. (1988). "Regression by Local Fitting." *Journal of Econometrics* 37:87–114.

Cook, R. D. (1977). "Detection of Influential Observations in Linear Regression." *Technometrics* 19:15–18.

Cook, R. D. (1979). "Influential Observations in Linear Regression." *Journal of the American Statistical Association* 74:169–174.

Daniel, C., and Wood, F. S. (1999). *Fitting Equations to Data: Computer Analysis of Multifactor Data*. 2nd ed. New York: John Wiley & Sons.

Darlington, R. B. (1968). "Multiple Regression in Psychological Research and Practice." *Psychological Bulletin* 69:161–182.

Davis, A. W. (1970). "Differential Equation of Hotelling's Generalized T^2." *Annals of Statistics* 39:815–832.

Davis, A. W. (1972). "On the Marginal Distributions of the Latent Roots of the Multivariate Beta Matrix." *Biometrika* 43:1664–1670.

Davis, A. W. (1979). "On the Differential Equation for Meijer's $G_{p,p}^{p,0}$ Function, and Further Tables of Wilks's Likelihood Ratio Criterion." *Biometrika* 66:519–531.

Davis, A. W. (1980). "Further Tabulation of Hotelling's Generalized T^2." *Communications in Statistics—Simulation and Computation* 9:321–336.

Draper, N. R., and Smith, H. (1998). *Applied Regression Analysis*. 3rd ed. New York: John Wiley & Sons.

Durbin, J., and Watson, G. S. (1951). "Testing for Serial Correlation in Least Squares Regression." *Biometrika* 37:409–428.

Efron, B., Hastie, T. J., Johnstone, I. M., and Tibshirani, R. (2004). "Least Angle Regression." *Annals of Statistics* 32:407–499. With discussion.

Eilers, P. H. C., and Marx, B. D. (1996). "Flexible Smoothing with B-Splines and Penalties." *Statistical Science* 11:89–121. With discussion.

Freund, R. J., and Littell, R. C. (1986). *SAS System for Regression*. 1986 ed. Cary, NC: SAS Institute Inc.

Freund, R. J., Littell, R. C., and Spector, P. C. (1991). *SAS System for Linear Models*. Cary, NC: SAS Institute Inc.

Friedman, J. H. (1991). "Multivariate Adaptive Regression Splines." *Annals of Statistics* 19:1–67.

Gentleman, W. M. (1972). *Basic Procedures for Large, Sparse, or Weighted Least Squares Problems*. Technical Report CSRR-2068, University of Waterloo, Ontario.

Gentleman, W. M. (1973). "Least Squares Computations by Givens Transformations without Square Roots." *Journal of the Institute of Mathematics and Its Applications* 12:329–336.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Hastie, T. J., and Tibshirani, R. J. (1990). *Generalized Additive Models*. New York: Chapman & Hall.

Hawkins, D. M. (1980). "A Note on Fitting a Regression without an Intercept Term." *American Statistician* 34:233.

Hosmer, D. W., Jr., and Lemeshow, S. (1989). *Applied Logistic Regression*. New York: John Wiley & Sons.

Huber, P. J. (1973). "Robust Regression: Asymptotics, Conjectures, and Monte Carlo." *Annals of Statistics* 1:799–821.

Johnston, J., and DiNardo, J. (1997). *Econometric Methods*. 4th ed. New York: McGraw-Hill.

Kennedy, W. J., Jr., and Gentle, J. E. (1980). *Statistical Computing*. New York: Marcel Dekker.

Kvalseth, T. O. (1985). "Cautionary Note about R^2." *American Statistician* 39:279–285.

Lee, Y.-S. (1972). "Some Results on the Distribution of Wilks's Likelihood-Ratio Criterion." *Biometrika* 59:649–664.

Mallows, C. L. (1973). "Some Comments on C_P." *Technometrics* 15:661–675.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Morrison, D. F. (2004). *Multivariate Statistical Methods*. 4th ed. New York: Duxbury Press.

Mosteller, F., and Tukey, J. W. (1977). *Data Analysis and Regression*. Reading, MA: Addison-Wesley.

Muller, K. E. (1998). "A New F Approximation for the Pillai-Bartlett Trace under H_0." *Journal of Computational and Graphical Statistics* 7:131–137.

Neter, J., and Wasserman, W. (1974). *Applied Linear Statistical Models*. Homewood, IL: Irwin.

Pillai, K. C. S. (1960). *Statistical Table for Tests of Multivariate Hypotheses*. Manila: University of Philippines Statistical Center.

Pillai, K. C. S., and Flury, B. N. (1984). "Percentage Points of the Largest Characteristic Root of the Multivariate Beta Matrix." *Communications in Statistics—Theory and Methods* 13:2199–2237.

Pindyck, R. S., and Rubinfeld, D. L. (1981). *Econometric Models and Econometric Forecasts*. 2nd ed. New York: McGraw-Hill.

Rao, C. R. (1973). *Linear Statistical Inference and Its Applications*. 2nd ed. New York: John Wiley & Sons.

Rawlings, J. O. (1988). *Applied Regression Analysis: A Research Tool*. Pacific Grove, CA: Wadsworth & Brooks/Cole Advanced Books & Software.

Rousseeuw, P. J. (1984). "Least Median of Squares Regression." *Journal of the American Statistical Association* 79:871–880.

Rousseeuw, P. J., and Yohai, V. (1984). "Robust Regression by Means of S-Estimators." In *Robust and Nonlinear Time Series Analysis*, edited by J. Franke, W. Härdle, and R. D. Martin, 256–274. Vol. 26 of Lecture Notes in Statistics. Berlin: Springer-Verlag.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B* 58:267–288.

Timm, N. H. (2002). *Applied Multivariate Analysis*. New York: Springer.

Weisberg, S. (1985). *Applied Linear Regression*. 2nd ed. New York: John Wiley & Sons.

Weisberg, S. (2005). *Applied Linear Regression*. 3rd ed. New York: John Wiley & Sons.

Wood, S. (2003). "Thin Plate Regression Splines." *Journal of the Royal Statistical Society, Series B* 65:95–114.

Wood, S. (2006). *Generalized Additive Models*. Boca Raton, FL: Chapman & Hall/CRC.

Yohai, V. J. (1987). "High Breakdown Point and High Efficiency Robust Estimates for Regression." *Annals of Statistics* 15:642–656.

Younger, M. S. (1979). *Handbook for Linear Regression*. North Scituate, MA: Duxbury Press.

Chapter 5
Introduction to Analysis of Variance Procedures

Contents

Overview: Analysis of Variance Procedures **109**
Procedures That Perform Sum of Squares Analysis of Variance 111
Procedures That Perform General Analysis of Variance 112
Statistical Details for Analysis of Variance **112**
From Sums of Squares to Linear Hypotheses 112
Tests of Effects Based on Expected Mean Squares 114
Analysis of Variance for Fixed-Effect Models **115**
PROC GLM for General Linear Models 115
PROC ANOVA for Balanced Designs 115
Comparing Group Means 115
PROC TTEST for Comparing Two Groups 116
Analysis of Variance for Categorical Data and Generalized Linear Models **116**
Nonparametric Analysis of Variance **117**
Constructing Analysis of Variance Designs **118**
For More Information **118**
References **118**

Overview: Analysis of Variance Procedures

The statistical term "analysis of variance" is used in a variety of circumstances in statistical theory and applications. In the *narrowest* sense, and the original sense of the phrase, it signifies a decomposition of a variance into contributing components. This was the sense used by R. A. Fisher when he defined the term to mean the expression of genetic variance as a sum of variance components due to environment, heredity, and so forth:

$$\sigma^2 = \sigma_1^2 + \sigma_2^2 + \cdots + \sigma_p^2$$

In this sense of the term, the SAS/STAT procedures that fit variance component models, such as the GLIMMIX, HPMIXED, MIXED, NESTED, and VARCOMP procedures, are "true" analysis of variance procedures.

Analysis of variance methodology in a slightly broader sense—and the sense most frequently understood today—applies the idea of an additive decomposition of variance to an additive decomposition of *sums of*

squares, whose expected values are functionally related to components of variation. A collection of sums of squares that measure and can be used for inference about meaningful features of a model is called a *sum of squares analysis of variance*, whether or not such a collection is an additive decomposition. In a linear model, the decomposition of sums of squares can be expressed in terms of projections onto orthogonal subspaces spanned by the columns of the design matrix $\mathbf{X}$. This is the general approach followed in the section "Analysis of Variance" on page 57 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software." Depending on the statistical question at hand, the projections can be formulated based on estimable functions, with different types of estimable functions giving rise to different types of sums of squares. Note that not all sum of squares analyses necessarily correspond to additive decompositions. For example, the Type III sums of squares often test hypotheses about the model that are more meaningful than those corresponding to the Type I sums of squares. But while the Type I sums of squares additively decompose the sum of squares due to all model contributions, the Type III sums of squares do not necessarily add up to any useful quantity. The four types of estimable functions in SAS/STAT software, their interpretation, and their construction are discussed in Chapter 16, "The Four Types of Estimable Functions." The application of sum of squares analyses is not necessarily limited to models with classification effects (factors). The methodology also applies to linear regression models that contain only continuous regressor variables.

An even broader sense of the term "analysis of variance" pertains to statistical models that contain classification effects (factors), and in particular, to models that contain *only* classification effects. Any statistical approach that measures features of such a model and can be used for inference is called a *general analysis of variance*. Thus the procedures for general analysis of variance in SAS/STAT are considered to be those that can fit statistical models containing factors, whether the data are experimental or observational. Some procedures for general analysis of variance have a statistical estimation principle that gives rise to a sum of squares analysis as discussed previously; others express a factor's contribution to the model fit in some other form. Note that this view of analysis of variance includes, for example, maximum likelihood estimation in generalized linear models with the GENMOD procedure, restricted maximum likelihood estimation in linear mixed models with the MIXED procedure, the estimation of variance components with the VARCOMP procedure, the comparison of means of groups with the TTEST procedure, and the nonparametric analysis of rank scores with the NPAR1WAY procedure, and so on.

In summary, analysis of variance in the contemporary sense of statistical modeling and analysis is more aptly described as *analysis of variation*, the study of the influences on the variation of a phenomenon. This can take, for example, the following forms:

- an analysis of variance table based on sums of squares followed by more specific inquiries into the relationship among factors and their levels
- a deviance decomposition in a generalized linear model
- a series of Type III tests followed by comparisons of least squares means in a mixed model

Procedures That Perform Sum of Squares Analysis of Variance

The flagship procedure in SAS/STAT software for linear modeling with sum of squares analysis techniques is the GLM procedure. It handles most standard analysis of variance problems. The following list provides descriptions of PROC GLM and other procedures that are used for more specialized situations:

ANOVA
: performs analysis of variance, multivariate analysis of variance, and repeated measures analysis of variance for *balanced* designs. PROC ANOVA also performs multiple comparison tests on arithmetic means.

GLM
: performs analysis of variance, regression, analysis of covariance, repeated measures analysis, and multivariate analysis of variance. PROC GLM produces several diagnostic measures, performs tests for random effects, provides contrasts and estimates for customized hypothesis tests, provides tests for means adjusted for covariates, and performs multiple-comparison tests on both arithmetic and adjusted means.

LATTICE
: computes the analysis of variance and analysis of simple covariance for data from an experiment with a lattice design. PROC LATTICE analyzes balanced square lattices, partially balanced square lattices, and some rectangular lattices.

MIXED
: performs mixed model analysis of variance and repeated measures analysis of variance via covariance structure modeling. When you choose one of the method-of-moment estimation techniques, the MIXED procedure produces an analysis of variance table with sums of squares, mean squares, and expected mean squares. PROC MIXED constructs statistical tests and intervals, enables customized contrasts and estimates, and computes empirical Bayes predictions.

NESTED
: performs analysis of variance and analysis of covariance for purely nested random models.

ORTHOREG
: performs regression by using the Gentleman-Givens computational method. For ill-conditioned data, PROC ORTHOREG can produce more accurate parameter estimates than other procedures, such as PROC GLM. See Chapter 91, "The ORTHOREG Procedure," for more information.

TRANSREG
: fits univariate and multivariate linear models, optionally with spline and other nonlinear transformations. Models include ordinary regression and ANOVA, multiple and multivariate regression, metric and nonmetric conjoint analysis, metric and nonmetric vector and ideal point preference mapping, redundancy analysis, canonical correlation, and response surface regression. See Chapter 126, "The TRANSREG Procedure," for more information.

VARCOMP
: estimates variance components for random or mixed models. If you choose the METHOD=TYPE1 or METHOD=GRR option, the VARCOMP procedure produces an analysis of variance table with sums of squares that correspond to the random effects in your models.

Procedures That Perform General Analysis of Variance

Many procedures in SAS/STAT enable you to incorporate classification effects into your model and to perform statistical inferences for experimental factors and their interactions. These procedures do not necessarily rely on sums of squares decompositions to perform these inferences. Examples of such procedures are the CATMOD, GENMOD, GLIMMIX, LOGISTIC, NPAR1WAY, and TTEST procedures. In fact, any one of the more than two dozen SAS/STAT modeling procedures that include a CLASS statement can be said to perform analysis of variance in this general sense. For more information about individual procedures, refer to their corresponding chapters in this documentation.

The following section discusses procedures in SAS/STAT that compute analysis of variance in models with classification factors in the narrow sense—that is, they produce analysis of variance tables and form F tests based on sums of squares, mean squares, and expected mean squares.

The subsequent sections discuss procedures that perform statistical inference in models with classification effects in the broader sense.

The following section also presents an overview of some of the fundamental features of analysis of variance. Subsequent sections describe how this analysis is performed with procedures in SAS/STAT software. For more detail, see the chapters for the individual procedures. Additional sources are described in the section References.

Statistical Details for Analysis of Variance

From Sums of Squares to Linear Hypotheses

Analysis of variance (ANOVA) is a technique for analyzing data in which one or more *response* (or *dependent* or simply Y) variables are measured under various conditions identified by one or more classification variables. The combinations of levels for the classification variables form the cells of the design for the data. This design can be the result of a controlled experiment or the result of an observational study in which you observe factors and factor level combinations in an uncontrolled environment. For example, an experiment might measure weight change (the dependent variable) for men and women who participated in three different weight-loss programs. The six cells of the design are formed by the six combinations of gender (men, women) and program (A, B, C).

In an analysis of variance, the variation in the response is separated into variation attributable to differences between the classification variables and variation attributable to random error. An analysis of variance constructs tests to determine the significance of the classification effects. A typical goal in such an analysis is to compare means of the response variable for various combinations of the classification variables.

The least squares principle is central to computing sums of squares in analysis of variance models. Suppose that you are fitting the linear model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ and that the error terms satisfy the usual assumptions (uncorrelated, zero mean, homogeneous variance). Further, suppose that $\mathbf{X}$ is partitioned according to several model effects, $\mathbf{X} = [\mathbf{X}_1\ \mathbf{X}_2 \cdots \mathbf{X}_k]$. If $\widehat{\boldsymbol{\beta}}$ denotes the ordinary least squares solution for this model, then the sum of squares attributable to the overall model can be written as

$$\mathrm{SSM} = \widehat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{Y} = \mathbf{Y}'\mathbf{H}\mathbf{Y}$$

where $\mathbf{H}$ is the "hat" matrix $\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'$. (This model sum of squares is not yet corrected for the presence of an explicit or implied intercept. This adjustment would consist of subtracting $n\overline{Y}^2$ from SSM.) Because of the properties of the hat matrix $\mathbf{H}$, you can write $\mathbf{X}' = \mathbf{X}'\mathbf{H}$ and $\mathbf{HX} = \mathbf{X}$. The (uncorrected) model sum of squares thus can also be written as

$$\mathrm{SSM} = \widehat{\boldsymbol{\beta}}'(\mathbf{X}'\mathbf{X})\widehat{\boldsymbol{\beta}}$$

This step is significant, because it demonstrates that sums of squares can be identified with quadratic functions in the least squares coefficients. The generalization of this idea is to do the following:

- consider hypotheses of interest in an analysis of variance model
- express the hypotheses in terms of linear estimable functions of the parameters
- compute the sums of squares associated with the estimable function
- construct statistical tests based on the sums of squares

Decomposing a model sum of squares into sequential, additive components, testing the significance of experimental factors, comparing factor levels, and performing other statistical inferences fall within this generalization. Suppose that $\mathbf{L}\boldsymbol{\beta}$ is an estimable function (see the section "Estimable Functions" on page 60 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," and Chapter 16, "The Four Types of Estimable Functions," for details). The sum of squares associated with the hypothesis $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ is

$$\mathrm{SS}(H) = \mathrm{SS}(\mathbf{L}\boldsymbol{\beta} = \mathbf{0}) = \widehat{\boldsymbol{\beta}}'\mathbf{L}'\left(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'\right)^{-1}\mathbf{L}\widehat{\boldsymbol{\beta}}$$

One application would be to form sums of squares associated with the different components of $\mathbf{X}$. For example, you can form a matrix $\mathbf{L}_2$ matrix such that $\mathbf{L}_2\boldsymbol{\beta} = \mathbf{0}$ tests the effect of adding the columns for $\mathbf{X}_2$ to an empty model or to test the effect of adding $\mathbf{X}_2$ to a model that already contains $\mathbf{X}_1$.

These sums of squares can also be expressed as the difference between two residual sums of squares, since $\mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ can be thought of as a (linear) restriction on the parameter estimates in the model:

$$\mathrm{SS}(H) = \mathrm{SSR}(\text{constrained model}) - \mathrm{SSR}(\text{full model})$$

If, in addition to the usual assumptions mentioned previously, the model errors are assumed to be normally distributed, then $\mathrm{SS}(H)$ follows a distribution that is proportional to a chi-square distribution. This fact, and the independence of $\mathrm{SS}(H)$ from the residual sum of squares, enables you to construct F tests based on sums of squares in least squares models.

The extension of sum of squares analysis of variance to general analysis of variance for classification effects depends on the fact that the distributional properties of quadratic forms in normal random variables are well understood. It is not necessary to first formulate a sum of squares to arrive at an exact or even approximate F test. The generalization of the expression for $\mathrm{SS}(H)$ is to form test statistics based on quadratic forms

$$\widehat{\boldsymbol{\beta}}'\mathbf{L}'\,\mathrm{Var}\left[\mathbf{L}\widehat{\boldsymbol{\beta}}\right]^{-1}\mathbf{L}\widehat{\boldsymbol{\beta}}$$

that follow a chi-square distribution if $\widehat{\boldsymbol{\beta}}$ is normally distributed.

Tests of Effects Based on Expected Mean Squares

Statistical tests in analysis of variance models can be constructed by comparing independent mean squares. To test a particular null hypothesis, you compute the ratio of two mean squares that have the same expected value under that hypothesis; if the ratio is much larger than 1, then that constitutes significant evidence against the null. In particular, in an analysis of variance model with fixed effects only, the expected value of each mean square has two components: quadratic functions of fixed parameters and random variation. For example, for a fixed effect called A, the expected value of its mean square is

$$\mathrm{E}[\mathrm{MS(A)}] = \mathrm{Q}(\boldsymbol{\beta}) + \sigma^2$$

where σ^2 is the common variance of the ϵ_i.

Under the null hypothesis of no A effect, the fixed portion $\mathrm{Q}(\boldsymbol{\beta})$ of the expected mean square is zero. This mean square is then compared to another mean square—say, MS(E)—that is independent of the first and has the expected value σ^2. The ratio of the two mean squares

$$F = \frac{\mathrm{MS(A)}}{\mathrm{MS(E)}}$$

has an F distribution under the null hypothesis.

When the null hypothesis is false, the numerator term has a larger expected value, but the expected value of the denominator remains the same. Thus, large F values lead to rejection of the null hypothesis. The probability of getting an F value at least as large as the one observed given that the null hypothesis is true is called the *significance probability value* (or the p-value). A p-value of less than 0.05, for example, indicates that data with *no* A effect will yield F values as large as the one observed less than 5% of the time. This is usually considered moderate evidence that there *is* a real A effect. Smaller p-values constitute even stronger evidence. Larger p-values indicate that the effect of interest is less than random noise. In this case, you can conclude either that there is no effect at all or that you do not have enough data to detect the differences being tested.

The actual pattern in expected mean squares of terms related to fixed quantities ($\mathrm{Q}(\boldsymbol{\beta})$) and functions of variance components depends on which terms in your model are fixed effects and which terms are random effects. This has bearing on how F statistics can be constructed. In some instances, exact tests are not available, such as when a linear combination of expected mean squares is necessary to form a proper denominator for an F test and a Satterthwaite approximation is used to determine the degrees of freedom of the approximation. The GLM and MIXED procedures can generate tables of expected mean squares and compute degrees of freedom by Satterthwaite's method. The MIXED and GLIMMIX procedures can apply Satterthwaite approximations and other degrees-of-freedom computations more widely than in analysis of variance models. See the section "Fixed, Random, and Mixed Models" on page 31 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," for a discussion of fixed versus random effects in statistical models.

Analysis of Variance for Fixed-Effect Models

PROC GLM for General Linear Models

The GLM procedure is the flagship tool for classical analysis of variance in SAS/STAT software. It performs analysis of variance by using least squares regression to fit general linear models. Among the statistical methods available in PROC GLM are regression, analysis of variance, analysis of covariance, multivariate analysis of variance, repeated measures analysis, and partial correlation analysis.

While PROC GLM can handle most common analysis of variance problems, other procedures are more efficient or have more features than PROC GLM for certain specialized analyses, or they can handle specialized models that PROC GLM cannot. Much of the rest of this chapter is concerned with comparing PROC GLM to other procedures.

PROC ANOVA for Balanced Designs

When you design an experiment, you choose how many experimental units to assign to each combination of levels (or cells) in the classification. In order to achieve good statistical properties and simplify the computations, you typically attempt to assign the same number of units to every cell in the design. Such designs are called *balanced designs*.

In SAS/STAT software, you can use the ANOVA procedure to perform analysis of variance for balanced data. The ANOVA procedure performs computations for analysis of variance that assume the balanced nature of the data. These computations are simpler and more efficient than the corresponding general computations performed by PROC GLM. Note that PROC ANOVA can be applied to certain designs that are not balanced in the strict sense of equal numbers of observations for all cells. These additional designs include all one-way models, regardless of how unbalanced the cell counts are, as well as Latin squares, which do not have data in all cells. In general, however, the ANOVA procedure is recommended only for balanced data. **If you use ANOVA to analyze a design that is not balanced, you must assume responsibility for the validity of the output.** You are responsible for recognizing incorrect results, which might include negative values reported for the sums of squares. If you are not certain that your data fit into a balanced design, then you probably need the framework of general linear models in the GLM procedure.

Comparing Group Means

The F test for a classification factor that has more than two levels tells you whether the level effects are significantly different from each other, but it does not tell you which levels differ from which other levels.

If the level comparisons are expressed through differences of the arithmetic cell means, you can use the MEANS statement in the GLM and ANOVA procedures for comparison. If arithmetic means are not appropriate for comparison, for example, because your data are unbalanced or means need to be adjusted for other model effects, then you can use the LSMEANS statement in the GLIMMIX, GLM, and MIXED procedures for level comparisons.

If you have specific comparisons in mind, you can use the CONTRAST statement in these procedures to make these comparisons. However, if you make many comparisons that use some given significance level (0.05, for example), you are more likely to make a type 1 error (incorrectly rejecting a hypothesis that the means are equal) simply because you have more chances to make the error.

Multiple-comparison methods give you more detailed information about the differences among the means and enable you to control error rates for a multitude of comparisons. A variety of multiple-comparison methods are available with the MEANS statement in both the ANOVA and GLM procedures, as well as the LSMEANS statement in the GLIMMIX, GLM, and MIXED procedures. These are described in detail in the section "Multiple Comparisons" on page 4231 in Chapter 53, "The GLM Procedure," and in Chapter 52, "The GLIMMIX Procedure," and Chapter 84, "The MIXED Procedure."

PROC TTEST for Comparing Two Groups

If you want to perform an analysis of variance and have only one classification variable with two levels, you can use PROC TTEST. In this special case, the results generated by PROC TTEST are equivalent to the results generated by PROC ANOVA or PROC GLM.

You can use PROC TTEST with balanced or unbalanced groups. In addition to the test assuming equal variances, PROC TTEST also performs a Satterthwaite test assuming unequal variances.

The TTEST procedure also performs equivalence tests, computes confidence limits, and supports both normal and lognormal data. If you have an AB/BA crossover design with no carryover effects, then you can use the TTEST procedure to analyze the treatment and period effects.

The PROC NPAR1WAY procedure performs nonparametric analogues to *t* tests. See Chapter 17, "Introduction to Nonparametric Analysis," for an overview and Chapter 90, "The NPAR1WAY Procedure," for details on PROC NPAR1WAY.

Analysis of Variance for Categorical Data and Generalized Linear Models

A *categorical variable* is defined as one that can assume only a limited number of values. For example, a person's gender is a categorical variable that can assume one of two values. Variables with levels that simply name a group are said to be measured on a *nominal scale*. Categorical variables can also be measured using an *ordinal scale*, which means that the levels of the variable are ordered in some way. For example, responses to an opinion poll are usually measured on an ordinal scale, with levels ranging from "strongly disagree" to "no opinion" to "strongly agree."

For two categorical variables, one measured on an ordinal scale and one measured on a nominal scale, you can assign scores to the levels of the ordinal variable and test whether the mean scores for the different levels of the nominal variable are significantly different. This process is analogous to performing an analysis of variance on continuous data, which can be performed by PROC CATMOD. If there are *n* nominal variables, rather than 1, then PROC CATMOD can perform an *n*-way analysis of variance of the mean scores.

For two categorical variables measured on a nominal scale, you can test whether the distribution of the first variable is significantly different for the levels of the second variable. This process is an analysis of variance

of proportions, rather than means, and can be performed by PROC CATMOD. The corresponding *n*-way analysis of variance can also be performed by PROC CATMOD.

See Chapter 9, "Introduction to Categorical Data Analysis Procedures," and Chapter 36, "The CATMOD Procedure," for more information.

The GENMOD procedure uses maximum likelihood estimation to fit generalized linear models. This family includes models for categorical data such as logistic, probit, and complementary log-log regression for binomial data and Poisson regression for count data, as well as continuous models such as ordinary linear regression, gamma, and inverse Gaussian regression models. PROC GENMOD performs analysis of variance through likelihood ratio and Wald tests of fixed effects in generalized linear models, and provides contrasts and estimates for customized hypothesis tests. It performs analysis of repeated measures data with generalized estimating equation (GEE) methods.

See Chapter 9, "Introduction to Categorical Data Analysis Procedures," and Chapter 51, "The GENMOD Procedure," for more information.

Nonparametric Analysis of Variance

Analysis of variance is sensitive to the distribution of the error term. If the error term is not normally distributed, the statistics based on normality can be misleading. The traditional test statistics are called *parametric tests* because they depend on the specification of a certain probability distribution except for a set of free parameters. Parametric tests are said to depend on distributional assumptions. Nonparametric methods perform the tests without making any strict distributional assumptions. Even if the data are distributed normally, nonparametric methods are often almost as powerful as parametric methods.

Most nonparametric methods are based on taking the ranks of a variable and analyzing these ranks (or transformations of them) instead of the original values. The NPAR1WAY procedure performs a nonparametric one-way analysis of variance. Other nonparametric tests can be performed by taking ranks of the data (using the RANK procedure) and using a regular parametric procedure (such as GLM or ANOVA) to perform the analysis. Some of these techniques are outlined in the description of PROC RANK in *SAS Programmers Guide: Essentials* and in Conover and Iman (1981).

Constructing Analysis of Variance Designs

Analysis of variance is most often used for data from designed experiments. You can use the PLAN procedure to construct designs for many experiments. For example, PROC PLAN constructs designs for completely randomized experiments, randomized blocks, Latin squares, factorial experiments, certain balanced incomplete block designs, and balanced crossover designs.

Randomization, or randomly assigning experimental units to cells in a design and to treatments within a cell, is another important aspect of experimental design. For either a new or an existing design, you can use PROC PLAN to randomize the experimental plan.

Additional features for design of experiments are available in SAS/QC software. The FACTEX and OPTEX procedures can construct a wide variety of designs, including factorials, fractional factorials, and D-optimal or A-optimal designs. These procedures provide features for randomizing and replicating designs; saving the design in an output data set; and interactively changing the design by changing its size, use of blocking, or the search strategies used. For more information, see the *SAS/QC User's Guide*.

For More Information

Analysis of variance was pioneered by Fisher (1925). For a general introduction to analysis of variance, see an intermediate statistical methods textbook such as Steel and Torrie (1980); Snedecor and Cochran (1980); Milliken and Johnson (1984); Mendenhall (1968); John (1971); Ott (1977); Kirk (1968). A classic source is Scheffé (1959). Freund, Littell, and Spector (1991) bring together a treatment of these statistical methods and SAS/STAT software procedures. Schlotzhauer and Littell (1997) cover how to perform *t* tests and one-way analysis of variance with SAS/STAT procedures. Texts on linear models include Searle (1971); Graybill (1976); Hocking (1985). Kennedy and Gentle (1980) survey the computing aspects. Other references appear in the reference section.

References

Conover, W. J., and Iman, R. L. (1981). "Rank Transformations as a Bridge between Parametric and Nonparametric Statistics." *American Statistician* 35:124–129.

Fisher, R. A. (1925). *Statistical Methods for Research Workers*. Edinburgh: Oliver & Boyd.

Freund, R. J., Littell, R. C., and Spector, P. C. (1991). *SAS System for Linear Models*. Cary, NC: SAS Institute Inc.

Graybill, F. A. (1976). *Theory and Applications of the Linear Model*. North Scituate, MA: Duxbury Press.

Hocking, R. R. (1985). *The Analysis of Linear Models*. Monterey, CA: Brooks/Cole.

John, P. W. M. (1971). *Statistical Design and Analysis of Experiments*. New York: Macmillan.

Kennedy, W. J., Jr., and Gentle, J. E. (1980). *Statistical Computing*. New York: Marcel Dekker.

Kirk, R. E. (1968). *Experimental Design: Procedures for the Behavioral Sciences*. Monterey, CA: Brooks/Cole.

Mendenhall, W. (1968). *Introduction to Linear Models and the Design and Analysis of Experiments*. Belmont, CA: Duxbury Press.

Milliken, G. A., and Johnson, D. E. (1984). *Designed Experiments.* Vol. 1 of Analysis of Messy Data. Belmont, CA: Lifetime Learning Publications.

Ott, L. (1977). *Introduction to Statistical Methods and Data Analysis*. 2nd ed. Belmont, CA: Duxbury Press.

Scheffé, H. (1959). *The Analysis of Variance*. New York: John Wiley & Sons.

Schlotzhauer, S. D., and Littell, R. C. (1997). *SAS System for Elementary Statistical Analysis*. 2nd ed. Cary, NC: SAS Institute Inc.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Snedecor, G. W., and Cochran, W. G. (1980). *Statistical Methods*. 7th ed. Ames: Iowa State University Press.

Steel, R. G. D., and Torrie, J. H. (1980). *Principles and Procedures of Statistics*. 2nd ed. New York: McGraw-Hill.

Chapter 6
Introduction to Mixed Modeling Procedures

Contents

Overview: Mixed Modeling Procedures **121**
Types of Mixed Models **123**
Linear, Generalized Linear, and Nonlinear Mixed Models 123
Linear Mixed Model 123
Generalized Linear Mixed Model 124
Nonlinear Mixed Model 125
Models for Clustered and Hierarchical Data 126
Models with Subjects and Groups 127
Linear Mixed Models **128**
Comparing the MIXED and GLM Procedures 129
Comparing the MIXED and HPMIXED Procedures 130
Generalized Linear Mixed Models **130**
Comparing the GENMOD and GLIMMIX Procedures 131
Nonlinear Mixed Models: The NLMIXED Procedure **132**
Bayesian Analysis: The BGLIMM Procedure **132**
References **132**

Overview: Mixed Modeling Procedures

A mixed model is a model that contains fixed and random effects. Since all statistical models contain some stochastic component and many models contain a residual error term, the preceding sentence deserves some clarification. The classical linear model $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ contains the parameters $\boldsymbol{\beta}$ and the random vector $\boldsymbol{\epsilon}$. The vector $\boldsymbol{\beta}$ is a vector of fixed-effects parameters; its elements are unknown constants to be estimated from the data. A mixed model in the narrow sense also contains random effects, which are unobservable random variables. If the vector of random effects is denoted by $\boldsymbol{\gamma}$, then a linear mixed model can be written as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

In a broader sense, mixed modeling and mixed model software is applied to special cases and generalizations of this model. For example, a purely random effects model, $\mathbf{Y} = \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$, or a correlated-error model, $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, is subsumed by mixed modeling methodology.

Over the last few decades virtually every form of classical statistical model has been enhanced to accommodate random effects. The linear model has been extended to the linear mixed model, generalized linear models have been extended to generalized linear mixed models, and so on. In parallel with this trend, SAS/STAT

software offers a number of classical and contemporary mixed modeling tools. The aim of this chapter is to provide a brief introduction and comparison of the procedures for mixed model analysis (in the broad sense) in SAS/STAT software. The theory and application of mixed models are discussed at length in many monographs, including Milliken and Johnson (1992); Diggle, Liang, and Zeger (1994); Davidian and Giltinan (1995); Verbeke and Molenberghs (1997, 2000); Vonesh and Chinchilli (1997); Demidenko (2004); Molenberghs and Verbeke (2005); and Littell et al. (2006).

The following procedures in SAS/STAT software can perform mixed and random effects analysis to various degrees:

BGLIMM fits Bayesian generalized linear mixed models by using Markov chain Monte Carlo methods. The BGLIMM procedure shares similar syntax with the MIXED and GLIMMIX procedures. Instead of maximizing the likelihood function to obtain maximum likelihood estimates, PROC BGLIMM draws samples from the joint posterior distribution of all parameters, fixed effects as well as random effects. The procedure produces estimates of posterior marginal distributions of the fixed-effects parameters that can be used for statistical inferences.

HPMIXED fits linear mixed models by sparse-matrix techniques. The HPMIXED procedure is designed to handle large mixed model problems, such as the solution of mixed model equations with thousands of fixed-effects parameters and random-effects solutions.

GLIMMIX fits generalized linear mixed models by likelihood-based techniques. As in the MIXED procedure, covariance structures are modeled parametrically. The GLIMMIX procedure also has built-in capabilities for mixed model smoothing and joint modeling of heterocatanomic multivariate data.

GLM is primarily a tool for fitting linear models by least squares. The GLM procedure has some capabilities for including random effects in a statistical model and for performing statistical tests in mixed models. Repeated measures analysis is also possible with the GLM procedure, assuming unstructured covariance modeling. Estimation methods for covariance parameters in PROC GLM are based on the method of moments, and a portion of its output applies only to the fixed-effects model.

LATTICE computes the analysis of variance and analysis of simple covariance for data from an experiment with a lattice design. PROC LATTICE analyzes balanced square lattices, partially balanced square lattices, and some rectangular lattices. Analyses performed with the LATTICE procedure can also be performed as mixed models for complete or incomplete block designs with the MIXED procedure.

MIXED performs mixed model analysis and repeated measures analysis by way of structured covariance models. The MIXED procedure estimates parameters by likelihood or moment-based techniques. You can compute mixed model diagnostics and influence analysis for observations and groups of observations. The default fitting method maximizes the restricted likelihood of the data under the assumption that the data are normally distributed and any missing data are missing at random. This general framework accommodates many common correlated-data methods, including variance component models and repeated measures analyses.

NESTED performs analysis of variance and analysis of covariance for purely nested random-effects models. Because of its customized algorithms, PROC NESTED can be useful for large data sets with nested random effects.

NLMIXED fits mixed models in which the fixed or random effects enter nonlinearly. The NLMIXED procedure requires that you specify components of your mixed model via programming statements. Some built-in distributions enable you to easily specify the conditional distribution of the data, given the random effects.

VARCOMP estimates variance components for random or mixed models.

The focus in the remainder of this chapter is on procedures designed for random effects and mixed model analysis: the GLIMMIX, HPMIXED, MIXED, NESTED, NLMIXED, and VARCOMP procedures. The important distinction between fixed and random effects in statistical models is addressed in the section "Fixed, Random, and Mixed Models" on page 31, in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software."

Types of Mixed Models

Linear, Generalized Linear, and Nonlinear Mixed Models

The linear model shown at the beginning of this chapter was incomplete because the distributional properties of the random variables and their relationship were not specified. In this section the specification of the models is completed and the three model classes, linear mixed models (LMM), generalized linear mixed models (GLMM), and nonlinear mixed models (NLMM), are delineated.

Linear Mixed Model

It is a defining characteristic of the class of linear mixed models (LMM), the class of generalized linear mixed models (GLMM), and the class of nonlinear mixed models (NLMM) that the random effects are normally distributed. In the linear mixed model, this also applies to the error term; furthermore, the errors and random effects are uncorrelated. The standard linear mixed model (LMM) is thus represented by the following assumptions:

$$\begin{aligned} \mathbf{Y} &= \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon} \\ \boldsymbol{\gamma} &\sim N(\mathbf{0}, \mathbf{G}) \\ \boldsymbol{\epsilon} &\sim N(\mathbf{0}, \mathbf{R}) \\ \mathrm{Cov}[\boldsymbol{\gamma}, \boldsymbol{\epsilon}] &= \mathbf{0} \end{aligned}$$

The matrices $\mathbf{G}$ and $\mathbf{R}$ are covariance matrices for the random effects and the random errors, respectively. A *G-side* random effect in a linear mixed model is an element of $\boldsymbol{\gamma}$, and its variance is expressed through an element in $\mathbf{G}$. An *R-side* random variable is an element of $\boldsymbol{\epsilon}$, and its variance is an element of $\mathbf{R}$. The GLIMMIX, HPMIXED, and MIXED procedures express the $\mathbf{G}$ and $\mathbf{R}$ matrices in parametric form—that is, you structure the covariance matrix, and its elements are expressed as functions of some parameters, known as the *covariance parameters* of the mixed models. The NLMIXED procedure also parameterizes the covariance structure, but you accomplish this with programming statements rather than with predefined syntax.

Since the right side of the model equation contains multiple random variables, the stochastic properties of $\mathbf{Y}$ can be examined by conditioning on the random effects, or through the marginal distribution. Because of

the linearity of the G-side random effects and the normality of the random variables, the conditional and the marginal distribution of the data are also normal with the following mean and variance matrices:

$$\begin{aligned}\mathbf{Y}|\boldsymbol{\gamma} &\sim N(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}, \mathbf{R})\\ \mathbf{Y} &\sim N(\mathbf{X}\boldsymbol{\beta}, \mathbf{V})\\ \mathbf{V} &= \mathbf{ZGZ}' + \mathbf{R}\end{aligned}$$

Parameter estimation in linear mixed models is based on likelihood or method-of-moment techniques. The default estimation method in PROC MIXED, and the only method available in PROC HPMIXED, is restricted (residual) maximum likelihood, a form of likelihood estimation that accounts for the parameters in the fixed-effects structure of the model to reduce the bias in the covariance parameter estimates. Moment-based estimation of the covariance parameters is available in the MIXED procedure through the METHOD= option in the PROC MIXED statement. The moment-based estimators are associated with sums of squares, expected mean squares (EMS), and the solution of EMS equations.

Parameter estimation by likelihood-based techniques in linear mixed models maximizes the marginal (restricted) log likelihood of the data—that is, the log likelihood is formed from $\mathbf{Y} \sim N(\mathbf{X}\boldsymbol{\beta}, \mathbf{V})$. This is a model for $\mathbf{Y}$ with mean $\mathbf{X}\boldsymbol{\beta}$ and covariance matrix $\mathbf{V}$, a correlated-error model. Such *marginal models* arise, for example, in the analysis of time series data, repeated measures, or spatial data, and are naturally subsumed into the linear mixed model family. Furthermore, some mixed models have an equivalent formulation as a correlated-error model, when both give rise to the same marginal mean and covariance matrix. For example, a mixed model with a single variance component is identical to a correlated-error model with compound-symmetric covariance structure, provided that the common correlation is positive.

Generalized Linear Mixed Model

In a generalized linear mixed model (GLMM) the G-side random effects are part of the linear predictor, $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}$, and the predictor is related nonlinearly to the conditional mean of the data

$$\mathrm{E}[\mathbf{Y}|\boldsymbol{\gamma}] = g^{-1}(\boldsymbol{\eta}) = g^{-1}(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma})$$

where $g^{-1}(\cdot)$ is the inverse link function. The conditional distribution of the data, given the random effects, is a member of the exponential family of distributions, such as the binary, binomial, Poisson, gamma, beta, or chi-square distribution. Because the normal distribution is also a member of the exponential family, the class of the linear mixed models is a subset of the generalized linear mixed models. In order to completely specify a GLMM, you need to do the following:

1. Formulate the linear predictor, including fixed and random effects.
2. Choose a link function.
3. Choose the distribution of the response, conditional on the random effects, from the exponential family.

As an example, suppose that s pairs of twins are randomly selected in a matched-pair design. One of the twins in each pair receives a treatment and the outcome variable is some binary measure. This is a study with s clusters (subjects) and each cluster is of size 2. If Y_{ij} denotes the binary response of twin $j = 1, 2$ in cluster i, then a linear predictor for this experiment could be

$$\eta_{ij} = \beta_0 + \tau x_{ij} + \gamma_i$$

where x_{ij} denotes a regressor variable that takes on the value 1 for the treated observation in each pair, and 0 otherwise. The γ_i are pair-specific random effects that model heterogeneity across sets of twins and that induce a correlation between the members of each pair. By virtue of random sampling the sets of twins, it is reasonable to assume that the γ_i are independent and have equal variance. This leads to a diagonal **G** matrix,

$$\mathrm{Var}[\boldsymbol{\gamma}] = \mathrm{Var}\begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \vdots \\ \gamma_s \end{bmatrix} = \begin{bmatrix} \sigma_\gamma^2 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_\gamma^2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_\gamma^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_\gamma^2 \end{bmatrix}$$

A common link function for binary data is the logit link, which leads in the second step of model formulation to

$$\mathrm{E}\left[Y_{ij}|\gamma_i\right] = \mu_{ij}|\gamma_i = \frac{1}{1+\exp\{-\eta_{ij}\}}$$

$$\mathrm{logit}\left\{\frac{\mu_{ij}|\gamma_i}{1-\mu_{ij}|\gamma_i}\right\} = \eta_{ij}$$

The final step, choosing a distribution from the exponential family, is automatic in this example; only the binary distribution comes into play to model the distribution of $Y_{ij}|\gamma_i$.

As for the linear mixed model, there is a marginal model in the case of a generalized linear mixed model that results from integrating the joint distribution over the random effects. This marginal distribution is elusive for many GLMMs, and parameter estimation proceeds by either approximating the model or by approximating the marginal integral. Details of these approaches are described in the section "Generalized Linear Mixed Models Theory" on page 3922, in Chapter 52, "The GLIMMIX Procedure."

A marginal model, one that models correlation through the **R** matrix and does not involve G-side random effects, can also be formulated in the GLMM family; such models are the extension of the correlated-error models in the linear mixed model family. Because nonnormal distributions in the exponential family exhibit a functional mean-variance relationship, fully parametric estimation is not possible in such models. Instead, estimating equations are formed based on first-moment (mean) and second-moment (covariance) assumptions for the marginal data. The approaches for modeling correlated nonnormal data via generalized estimating equations (GEE) fall into this category (see, for example, Liang and Zeger 1986; Zeger and Liang 1986).

Nonlinear Mixed Model

In a nonlinear mixed model (NLMM), the fixed and/or random effects enter the conditional mean function nonlinearly. If the mean function is a general, nonlinear function, then it is customary to assume that the conditional distribution is normal, such as in modeling growth curves or pharmacokinetic response. This is not a requirement, however.

An example of a nonlinear mixed model is the following logistic growth curve model for the *j*th observation

of the ith subject (cluster):

$$f(\boldsymbol{\beta}, \boldsymbol{\gamma}_i, x_{ij}) = \frac{\beta_1 + \gamma_{i1}}{1 + \exp[-(x_{ij} - \beta_2)/(\beta_3 + \gamma_{i2})]}$$

$$Y_{ij} = f(\boldsymbol{\beta}, \boldsymbol{\gamma}_i, x_{ij}) + \epsilon_{ij}$$

$$\begin{bmatrix} \gamma_{i1} \\ \gamma_{i2} \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix} \right)$$

$$Y_{ij} | \gamma_{i1}, \gamma_{i2} \sim N(0, \sigma_\epsilon^2)$$

The inclusion of R-side covariance structures in GLMM and NLMM models is not as straightforward as in linear mixed models for the following reasons:

- The normality of the conditional distribution in the LMM enables straightforward modeling of the covariance structure because the mean structure and covariance structure are not functionally related.
- The linearity of the random effects in the LMM leads to a marginal distribution that incorporates the **R** matrix in a natural and meaningful way.

To incorporate R-side covariance structures when random effects enter nonlinearly or when the data are not normally distributed requires estimation approaches that rely on linearizations of the mixed model. Among such estimation methods are the pseudo-likelihood methods that are available with the GLIMMIX procedure. Generalized estimating equations also solve this marginal estimation problem for nonnormal data; these are available with the GENMOD procedure.

Models for Clustered and Hierarchical Data

Mixed models are often applied in situations where data are clustered, grouped, or otherwise hierarchically organized. For example, observations might be collected by randomly selecting schools in a school district, then randomly selecting classrooms within schools, and then randomly selecting students within the classroom. A longitudinal study might randomly select individuals and then repeatedly take measurements on them. In the first example, a school is a cluster of observations, which consists of smaller clusters (classrooms) and so on. In the longitudinal example the observations for a particular individual form a cluster. Mixed models are popular analysis tools for hierarchically organized data for the following reasons:

- The selection of groups is often performed randomly, so that the associated effects are random effects.
- The data from different clusters are independent by virtue of the random selection or by assumption.
- The observations from the same cluster are often correlated, such as the repeated observations in a repeated measures or longitudinal study.
- It is often believed that there is heterogeneity in model parameters across subjects; for example, slopes and intercepts might differ across individuals in a longitudinal growth study. This heterogeneity, if due to stochastic sources, can be modeled with random effects.

A linear mixed models with clustered, hierarchical structure can be written as a special case of the general linear mixed model by introducing appropriate subscripts. For example, a mixed model with one type of clustering and s clusters can be written as

$$\mathbf{Y}_i = \mathbf{X}_i \boldsymbol{\beta} + \mathbf{Z}_i \boldsymbol{\gamma}_i + \boldsymbol{\epsilon}_i \quad i = 1, \ldots, s$$

In SAS/STAT software, the clusters are referred to as *subjects*, and the effects that define clusters in your data can be specified with the SUBJECT= option in the GLIMMIX, HPMIXED, MIXED, and NLMIXED procedures. The vector $\mathbf{Y}_i$ collects the n_i observations for the ith subject. In certain disciplines, the organization of a hierarchical model is viewed in a *bottom-up* form, where the measured observations represent the first level, these are collected into units at the second level, and so forth. In the school data example, the bottom-up approach considers a student's score as the level-1 observation, the classroom as the level-2 unit, and the school district as the level-3 unit (if these were also selected from a population of districts).

The following points are noteworthy about mixed models with SUBJECT= specification:

- A SUBJECT= option is available in the RANDOM statements of the GLIMMIX, HPMIXED, MIXED, and NLMIXED procedures and in the REPEATED statement of the MIXED and HPMIXED procedures.
- A SUBJECT= specification is required in the NLMIXED and HPMIXED procedures. It is not required with any other mixed modeling procedure in SAS/STAT software.
- Specifying models with subjects is usually more computationally efficient in the MIXED and GLIMMIX procedures, especially if the SUBJECT= effects are identical or contained within each other. The computational efficiency of the HPMIXED procedure is not dependent on SUBJECT= effects in the manner in which the MIXED and GLIMMIX procedures are affected.
- There is no limit to the number of SUBJECT= effects with the MIXED, HPMIXED, and GLIMMIX procedures—that is, you can achieve an arbitrary depth of the nesting.

Models with Subjects and Groups

The concept of a subject as a unit of clustering observations in a mixed model has been described in the preceding section. This concept is important for mixed modeling with the GLIMMIX, HPMIXED, MIXED, and NLMIXED procedures. Observations from two subjects are considered uncorrelated in the analysis. Observations from the same subject are potentially correlated, depending on your specification of the covariance structure. Random effects at the subject level always lead to correlation in the marginal distribution of the observations that belong to the subject.

The GLIMMIX, HPMIXED, and MIXED procedures also support the notion of a GROUP= effect in the specification of the covariance structure. Like a subject effect, a G-side group effect identifies independent random effects. In addition to a subject effect, the group effect assumes that the realizations of the random effects correspond to draws from different distributions; in other words, each level of the group effect is associated with a different set of covariance parameters. For example, the following statements in any of these procedures fit a random coefficient model with fixed intercept and slope and subject-specific random intercept and slope:

```
class id;
model y = x;
random intercept x / subject=id;
```

The interpretation of the RANDOM statement is that for each ID an independent draw is made from a bivariate normal distribution with zero mean and a diagonal covariance matrix. In the following statements (in any of these procedures) these independent draws come from different bivariate normal distributions depending on the value of the grp variable.

```
class id grp;
model y = x;
random intercept x / subject=id group=grp;
```

Adding GROUP= effects in your model increases the flexibility to model heterogeneity in the covariance parameters, but it can add numerical complexity to the estimation process.

Linear Mixed Models

You can fit linear mixed models in SAS/STAT software with the GLM, GLIMMIX, HPMIXED, LATTICE, MIXED, NESTED, and VARCOMP procedures.

The procedure specifically designed for statistical estimation in linear mixed models is the MIXED procedure. To fit the linear mixed model

$$\begin{aligned} \mathbf{Y} &= \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon} \\ \boldsymbol{\gamma} &\sim N(\mathbf{0}, \mathbf{G}) \\ \boldsymbol{\epsilon} &\sim N(\mathbf{0}, \mathbf{R}) \\ \mathrm{Cov}[\boldsymbol{\gamma}, \boldsymbol{\epsilon}] &= \mathbf{0} \end{aligned}$$

with the MIXED procedure, you specify the fixed-effects design matrix $\mathbf{X}$ in the MODEL statement, the random-effects design matrix $\mathbf{Z}$ in the RANDOM statement, the covariance matrix of the random effects $\mathbf{G}$ with options (SUBJECT=, GROUP=, TYPE=) in the RANDOM statement, and the $\mathbf{R}$ matrix in the REPEATED statement.

By default, covariance parameters are estimated by restricted (residual) maximum likelihood. In supported models, the METHOD=TYPE1, METHOD=TYPE2, and METHOD=TYPE3 options lead to method-of-moment-based estimators and analysis of variance. The MIXED procedure provides an extensive list of diagnostics for mixed models, from various residual graphics to observationwise and groupwise influence diagnostics.

The NESTED procedure performs an analysis of variance in nested random effects models. The VARCOMP procedure can be used to estimate variance components associated with random effects in random and mixed models. The LATTICE procedure computes analysis of variance for balanced and partially balanced square lattices. You can fit the random and mixed models supported by these procedures with the MIXED procedure as well. Some specific analyses, such as the analysis of Gauge R & R studies in the VARCOMP procedure (Burdick, Borror, and Montgomery 2005), are unique to the specialized procedures.

The GLIMMIX procedure can fit most of the models that you can fit with the MIXED procedure, but it does not offer method-of-moment-based estimation and analysis of variance in the narrow sense. Also, PROC

GLIMMIX does not support the same array of covariance structures as the MIXED procedure and does not support a sampling-based Bayesian analysis. An in-depth comparison of the GLIMMIX and MIXED procedures can be found in the section "Comparing the GLIMMIX and MIXED Procedures" on page 3977, in Chapter 52, "The GLIMMIX Procedure."

Comparing the MIXED and GLM Procedures

Random- and mixed-effects models can also be fitted with the GLM procedure, but the philosophy is different from that of PROC MIXED and other dedicated mixed modeling procedures. The following lists important differences between the GLM and MIXED procedures in fitting random and mixed models:

- The default estimation method for covariance parameters in the MIXED procedure is restricted maximum likelihood. Covariance parameters are estimated by the method of moments by solving expressions for expected mean squares.
- In the GLM procedure, fixed and random effects are listed in the MODEL statement. Only fixed effects are listed in the MODEL statement of the MIXED procedure. In the GLM procedure, random effects must be repeated in the RANDOM statement.
- You can request tests for model effects by adding the TEST option in the RANDOM statement of the GLM procedure. PROC GLM then constructs exact tests for random effects if possible and constructs approximate tests if exact tests are not possible. For details on how the GLM procedure constructs tests for random effects, see the section "Computation of Expected Mean Squares for Random Effects" on page 4257, in Chapter 53, "The GLM Procedure." Tests for fixed effects are constructed by the MIXED procedure as Wald-type F tests, and the degrees of freedom for these tests can be determined by a variety of methods.
- Some of the output of the GLM procedure applies only to the fixed effects part of the model, whether a RANDOM statement is specified or not.
- Variance components are independent in the GLM procedure and covariance matrices are generally unstructured. The default covariance structure for variance components in the MIXED procedure is also a variance component structure, but the procedure offers a large number of parametric structures to model covariation among random effects and observations.

Comparing the MIXED and HPMIXED Procedures

The HPMIXED procedure is designed to solve large mixed model problems by using sparse matrix techniques. The largeness of a mixed model can take many forms: a large number of observations, large number of columns in the **X** matrix, a large number of random effects, or a large number of covariance parameters. The province of the HPMIXED procedure is parameter estimation, inference, and prediction in mixed models with large **X** and/or **Z** matrices, many observations, but relatively few covariance parameters.

The models that you can fit with the HPMIXED procedure are a subset of the models available with the MIXED procedure. The HPMIXED procedure supports only a limited number of types of covariance structure in the RANDOM and REPEATED statements in order to balance performance and generality.

To some extent, the generality of the MIXED procedure precludes it from serving as a high-performance computing tool for all the model-data scenarios that the procedure can potentially estimate parameters for. For example, although efficient sparse algorithms are available to estimate variance components in large mixed models, the computational configuration changes profoundly when, for example, standard error adjustments and degrees of freedom by the Kenward-Roger method are requested.

Generalized Linear Mixed Models

Generalized linear mixed models can be fit with the GLIMMIX and NLMIXED procedures in SAS/STAT software. The GLIMMIX procedure is specifically designed to fit this class of models and offers syntax very similar to the syntax of other linear modeling procedures, such as the MIXED procedure. Consider a generalized linear model with linear predictor and link function

$$\mathrm{E}[\mathbf{Y}|\boldsymbol{\gamma}] = g^{-1}(\boldsymbol{\eta}) = g^{-1}(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma})$$

and distribution in the exponential family. The fixed-effects design matrix **X** is specified in the MODEL statement of the GLIMMIX procedure, and the random-effects design matrix **Z** is specified in the RANDOM statement, along with the covariance matrix of the random effects and the covariance matrix of R-side random variables. The link function and (conditional) distribution are determined by defaults or through options in the MODEL statement.

The GLIMMIX procedure can fit heterocatanomic multivariate data—that is, data that stem from different distributions. For example, one measurement taken on a patient might be a continuous, normally distributed outcome, whereas another measurement might be a binary indicator of medical history. The GLIMMIX procedure also provides capabilities for mixed model smoothing and mixed model splines.

The GLIMMIX procedure offers an extensive array of postprocessing features to produce output statistics and to perform linear inference. The ESTIMATE and LSMESTIMATE statements support multiplicity-adjusted *p*-values for the protection of the familywise Type-I error rate. The LSMEANS statement supports the slicing of interactions, simple effect differences, and ODS statistical graphs for group comparisons.

The default estimation technique in the GLIMMIX procedure depends on the class of models fit. For linear mixed models, the default technique is restricted maximum likelihood, as in the MIXED procedure. For generalized linear mixed models, the estimation is based on linearization methods (pseudo-likelihood) or on integral approximation by adaptive quadrature or Laplace methods.

The NLMIXED procedure facilitates the fitting of generalized linear mixed models through several built-in distributions from the exponential family (binary, binomial, gamma, negative binomial, and Poisson). You have to code the linear predictor and link function with SAS programming statements and assign starting values to all parameters, including the covariance parameters. Although you are not required to specify starting values with the NLMIXED procedure (because the procedure assigns a default value of 1.0 to every parameter not explicitly given a starting value), it is highly recommended that you specify good starting values. The default estimation technique of the NLMIXED procedure, an adaptive Gauss-Hermite quadrature, is also available in the GLIMMIX procedure through the METHOD=QUAD option in the PROC GLIMMIX statement. The Laplace approximation that is available in the NLMIXED procedure by setting QPOINTS=1 is available in the GLIMMIX procedure through the METHOD=LAPLACE option.

Comparing the GENMOD and GLIMMIX Procedures

The GENMOD and GLIMMIX procedures can fit generalized linear models and estimate the parameters by maximum likelihood. For multinomial data, the GENMOD procedure fits cumulative link models for ordinal data. The GLIMMIX procedure fits these models and generalized logit models for nominal data.

When data are correlated, you can use the REPEATED statement in the GENMOD procedure to fit marginal models via generalized estimating equations. A working covariance structure is assumed, and the standard errors of the parameter estimates are computed according to an empirical ("sandwich") estimator that is robust to the misspecification of the covariance structure. Marginal generalized linear models for correlated data can also be fit with the GLIMMIX procedure by specifying the random effects as R-side effects. The empirical covariance estimators are available through the EMPIRICAL= option in the PROC GLIMMIX statement. The essential difference between the estimation approaches taken by the GLIMMIX procedure and generalized estimating equations is that the latter approach estimates the covariance parameters by the method of moments, whereas the GLIMMIX procedure uses likelihood-based techniques.

The GENMOD procedure supports nonsingular parameterizations of classification variables through its CLASS statement. The GLIMMIX procedure supports only the standard, GLM-type singular parameterization of CLASS variables. For the differences between these parameterizations, see the section "Parameterization of Model Effects" on page 412, in Chapter 20, "Shared Concepts and Topics."

Nonlinear Mixed Models: The NLMIXED Procedure

PROC NLMIXED handles models in which the fixed or random effects enter nonlinearly. It requires that you specify a conditional distribution of the data given the random effects, with available distributions including the normal, binomial, and Poisson. You can alternatively code your own distribution with SAS programming statements. Under a normality assumption for the random effects, PROC NLMIXED performs maximum likelihood estimation via adaptive Gaussian quadrature and a dual quasi-Newton optimization algorithm. Besides standard maximum likelihood results, you can obtain empirical Bayes predictions of the random effects and estimates of arbitrary functions of the parameters with delta-method standard errors. PROC NLMIXED has a wide variety of applications; two of the most common applications are nonlinear growth curves and overdispersed binomial data.

Bayesian Analysis: The BGLIMM Procedure

PROC BGLIMM provides Bayesian inference for generalized linear mixed models. Bayesian analysis is different from the classical (or frequentist) approach in the sense that the Bayesian paradigm treats all parameters (including fixed effects and random effects) as random variables, and the objective is to derive inference from the joint posterior distribution of all parameters. PROC BGLIMM uses syntax similar to that of the MIXED and GLIMMIX procedures (the CLASS, MODEL, RANDOM, REPEATED, and ESTIMATE statements) to specify a GLMM, and it uses Markov chain Monte Carlo methods to draw samples from the joint posterior distribution. You can use PROC BGLIMM to fit multilevel nested and non-nested random-effects models and to handle repeated measures data and missing data problems.

References

Burdick, R. K., Borror, C. M., and Montgomery, D. C. (2005). *Design and Analysis of Gauge R&R Studies: Making Decisions with Confidence Intervals in Random and Mixed ANOVA Models*. Philadelphia, PA, and Alexandria, VA: SIAM and ASA.

Davidian, M., and Giltinan, D. M. (1995). *Nonlinear Models for Repeated Measurement Data*. New York: Chapman & Hall.

Demidenko, E. (2004). *Mixed Models: Theory and Applications*. New York: John Wiley & Sons.

Diggle, P. J., Liang, K.-Y., and Zeger, S. L. (1994). *Analysis of Longitudinal Data*. Oxford: Clarendon Press.

Laird, N. M., and Ware, J. H. (1982). "Random-Effects Models for Longitudinal Data." *Biometrics* 38:963–974.

Liang, K.-Y., and Zeger, S. L. (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika* 73:13–22.

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and Schabenberger, O. (2006). *SAS for Mixed Models*. 2nd ed. Cary, NC: SAS Institute Inc.

Milliken, G. A., and Johnson, D. E. (1992). *Designed Experiments.* Vol. 1 of Analysis of Messy Data. Reprint edition. New York: Chapman & Hall.

Molenberghs, G., and Verbeke, G. (2005). *Models for Discrete Longitudinal Data*. New York: Springer.

Verbeke, G., and Molenberghs, G., eds. (1997). *Linear Mixed Models in Practice: A SAS-Oriented Approach*. New York: Springer.

Verbeke, G., and Molenberghs, G. (2000). *Linear Mixed Models for Longitudinal Data*. New York: Springer.

Vonesh, E. F., and Chinchilli, V. M. (1997). *Linear and Nonlinear Models for the Analysis of Repeated Measurements*. New York: Marcel Dekker.

Zeger, S. L., and Liang, K.-Y. (1986). "Longitudinal Data Analysis for Discrete and Continuous Outcomes." *Biometrics* 42:121–130.

Chapter 7
Introduction to Causal Analysis Procedures

Contents

Overview of Causal Analysis **135**
The Causal Analysis Procedures **137**
PROC CAUSALGRAPH 137
PROC CAUSALMED 138
PROC CAUSALTRT 139
PROC PSMATCH 141
Using the Causal Analysis Procedures **143**
Using PROC CAUSALGRAPH before Using Other Causal Analysis Procedures . . . 143
Using PROC PSMATCH Repeatedly to Ensure Proper Covariate Balancing 144
Using PROC CAUSALTRT or PROC PSMATCH to Estimate Causal Effects 144
References **145**

Overview of Causal Analysis

The study of causal relationships is one of the most important endeavors in empirical science. Knowledge of whether a *treatment* (or an *intervention*) would have a causal effect on an *outcome* enables you to make informed decisions or take actions that could lead to anticipated results about the outcome. All the following examples involve an investigation of the putative causal effect of a treatment variable on an outcome variable:

- Does smoking cause lung cancer?
- Can a new policy boost the small business economy?
- Does attending a private high school lead to greater success in college than attending a public high school?

When you use randomized controlled trials (RCTs) to study causal relationships, statistical procedures for estimating causal effects are relatively easy to apply. In a simple RCT, each subject (or research unit) is randomly assigned to either the treatment or control condition. The purpose of randomization is to make the two groups comparable by design. Then the observed differences in the outcomes between the treatment and control groups are attributed solely to the causal effects of the treatment. You can use statistical procedures such as the ANOVA, GENMOD, GLM, LIFEREG, REG, and TTEST procedures in SAS/STAT software, among others, to estimate the causal effects and test statistical significance.

But when conducting RCT is not practical, you would need to draw causal inferences from observational data or imperfectly randomized experiments. With observational data, the statistical associations among the

observed variables reflect not only causal influences but also noncausal, or confounding, factors. This is the explanation behind the maxim that association is not causation. To infer causal effects from observational data, you might need to rely on specialized methods that can somehow address the confounding associations and other causal inference issues. In this regard, several procedures available in SAS/STAT can mitigate these issues and help make valid causal inferences:

- The CAUSALGRAPH procedure applies the theory of graphical causal models to guide you with valid modeling strategies for estimating causal effects.
- The CAUSALMED procedure posits a mediator in the causal process through which a treatment causally affects an outcome, and it estimates the direct, indirect, and various causal mediation effects.
- The CAUSALTRT procedure estimates causal treatment effects by propensity score weighting, outcome regression, and doubly robust methods.
- The PSMATCH procedure provides a variety of tools for propensity score analysis and produces output data sets to which you can apply propensity score matching, weighting, and stratification methods in order to estimate causal effects.

The main features and usages of these procedures are described in the next few sections. For an introduction to causal inference with data from observational or imperfectly randomized experiments, see Berzuini, Dawid, and Bernardinelli (2012), Hernán and Robins (2020), Imbens and Rubin (2015), and Morgan and Winship (2015) and references therein.

For valid causal inference and effect estimation from data to be possible, some assumptions must be made about the statistical techniques and methods that these procedures use. These assumptions are laid out within the potential (or counterfactual) outcomes framework (Neyman, Dabrowska, and Speed 1990; Pearl 2001; Robins and Greenland 1992; Rubin 1990) or the structural causal model framework (Pearl 2009b). It is important to emphasize that these assumptions are in addition to those of standard statistical modeling, such as independent observations or distributional assumptions. It is essential for the validity of the causal inferences that are obtained through the causal procedures that these assumptions be tenable. For more information about these assumptions and the related theoretical frameworks, see the following sections for the individual procedures:

- PROC CAUSALGRAPH: "Causal Graphs and Potential Outcomes" on page 2384
- PROC CAUSALMED: "Causal Mediation Effects: Theory, Definitions, and Effect Decompositions" on page 2455
- PROC CAUSALTRT: "Causal Effects: Definitions, Assumptions, and Identification" on page 2533
- PROC PSMATCH: "Observational Studies Contrasted with Randomized Trials" on page 8404 and "Propensity Score Analysis" on page 8405

The Causal Analysis Procedures

PROC CAUSALGRAPH

The CAUSALGRAPH procedure examines the structure of graphical causal models and suggests statistical strategies or steps that enable researchers to estimate causal effects that have valid causal interpretations.

Causal models are encoded in the form of directed acyclic graphs (DAGs) (Pearl 2009a, b), which are the primary input for the procedure. In the following example, you input a causal model in the MODEL statement and inquire about the identification conditions for the causal effect of PFAS on Duration in the IDENTIFY statement:

```
proc causalgraph;
   model "MyModel"
      Age    ==> Parity PFAS Education,
      Parity ==> PrevBF Duration PFAS,
      PrevBF ==> PFAS Duration,
      PFAS   ==> Duration,
      Education ==> Duration Employment PFAS BMI Alcohol Smoking,
      Employment ==> Duration PFAS BMI Alcohol Smoking,
      BMI Alcohol Smoking ==> Duration;
   identify PFAS ==> Duration;
run;
```

PROC CAUSALGRAPH interprets and encodes the (causal) directional relationships among the variables that you specify in the MODEL statement. It then applies the causal DAG theory to determine appropriate statistical strategies that you can use to estimate the causal effect of PFAS on Duration. For example, PROC CAUSALGRAPH outputs the valid sets of covariates that you can use for adjustments in outcome regression analysis or for matching in propensity score analysis. By using regression adjustment or propensity score matching techniques to estimate the target causal treatment effects, you can minimize the biasing effects due to the confounding covariates.

PROC CAUSALGRAPH provides a number of criteria for identifying causal treatment effects:

- constructive backdoor criterion
- backdoor criterion
- parents-of-treatment criterion
- parents-of-outcome criterion
- joint ancestor criterion
- instrumental variables

To identify the sets of adjustment covariates or instrumental variables, you can either request that the procedure list these sets (as in the preceding example) or test the validity of the sets that you specify. For more information about different criteria of identification, see the METHOD= option in Chapter 37, "The CAUSALGRAPH Procedure."

PROC CAUSALGRAPH supports the specification of multiple models, multiple treatments, and multiple outcomes. Other main features of PROC CAUSALGRAPH include the following:

- identification of joint causal effects from multiple treatments on multiple outcomes
- enumeration of all the observationally testable assumptions that are encoded by a causal model
- listing of causal and noncausal treatment-to-outcome proper paths
- ability to import and export saved models
- ODS graphics to visualize causal models

For more information about PROC CAUSALGRAPH, see Chapter 37, "The CAUSALGRAPH Procedure," and Thompson (2019).

PROC CAUSALMED

The CAUSALMED procedure estimates causal mediation effects in situations where the observed effects might have been confounded, such as in observational studies.

In causal mediation analysis, a treatment variable T is posited to causally affect an outcome variable Y through a mediation process that is reflected by the values of a mediation variable M. The total causal effect of T on Y is decomposed into two parts:

- an indirect effect through the mediation process, denoted as $T \rightarrow M \rightarrow Y$
- a direct effect not through the mediation process, denoted as $T \rightarrow Y$

To estimate causal mediation effects, PROC CAUSALMED uses the regression adjustment approach proposed by Valeri and VanderWeele (2013) and VanderWeele (2014). For a comprehensive review, see VanderWeele (2015). In this approach, you specify parametric models for the outcome variable and the mediator variable. In addition, confounding covariate effects are dealt with during model estimation.

For example, the following statements specify a causal mediation analysis in which you investigate the causal effects of the treatment variable Smoking (smoking behavior of mother) on the outcome variable Death (infant death):

```
proc causalmed data=birthwgt;
   class LowBirthWgt Smoking Death AgeGroup Race Drinking;
   mediator LowBirthWgt = Smoking;
   model Death = LowBirthWgt | Smoking;
   covar AgeGroup Race Drinking;
run;
```

You specify the outcome regression model in the MODEL statement and the mediation model in the MEDIATOR statement. Together, these models specify that the causal effect of Smoking on Death is mediated by the mediator variable LowBirthWgt (low birth weight) and that the interaction effect, as well as the main effects of Smoking and LowBirthWgt, is included in the outcome model. In addition, you specify in the COVAR statement that the covariates AgeGroup, Race, and Drinking might have confounded the

observed effects. These confounding effects would then be adjusted for when PROC CAUSALMED fits the outcome and mediator models and estimates various causal mediation effects.

The main results from PROC CAUSALMED include the total effect of T on Y, the natural indirect effect of T on Y that is mediated by M, the natural direct effect of T on Y that is not mediated by M, and the percentage of total effect that is due to mediation. Standard errors for these estimates are computed by analytic formulas or bootstrapping.

PROC CAUSALMED supports binary and continuous variables for the mediator and the following combinations of data type and model for the outcome response:

- continuous outcomes that are fitted by linear models
- time-to-event outcomes that are fitted by accelerated failure time or Cox proportional hazards models
- count data that are fitted by Poisson or negative binomial models
- binary outcomes that are fitted by generalized linear models with log or logit links

Other main features of PROC CAUSALMED include the following:

- estimation of the controlled direct effect, portion eliminated, percentage due to interaction, and various types of mediation effects and percentages
- computation of various two-way, three-way, and four-way effect decompositions of causal mediation effects
- estimation of conditional causal mediation effects for the subgroups of interest

For more information about PROC CAUSALMED, see Chapter 38, "The CAUSALMED Procedure," and Yung, Lamm, and Zhang (2018)

PROC CAUSALTRT

The CAUSALTRT procedure estimates the average causal effect of a binary treatment T on a continuous or discrete outcome Y, where the observed effect might have been confounded, such as in observational studies.

Two types of modeling are used in PROC CAUSALTRT. One is outcome modeling by a generalized linear model. The other is propensity score modeling of the treatment variable by a logistic regression model (Guo and Fraser 2015; Imbens and Rubin 2015). For example, the following statements specify both outcome and treatment modeling for estimating the causal treatment effect of Drug on Diabetes (both are assumed to be binary here):

```
proc causaltrt data=drugs;
   class Gender;
   psmodel Drug = Age Gender BMI;
   model Diabetes = Age Gender BMI;
run;
```

In the PSMODEL statement, you specify that the treatment variable is Drug and the corresponding propensity score model contains Age, Gender, and BMI as predictors of the treatment variable. The propensity scores in this model refer to the probabilities of drug use in individuals. In the MODEL statement, you specify that the outcome is Diabetes, which is modeled by Age, Gender, BMI, and Drug. Notice that the treatment effect of Drug on Diabetes is made implicit here. With the modeling of both the outcome and propensity scores, PROC CAUSALTRT uses a doubly robust estimation method (Lunceford and Davidian 2004) to estimate the potential outcome means, the average causal treatment effect, and their standard errors.

In fact, depending on whether or not you specify the effects on the right-hand side of the equal sign in the PSMODEL and MODEL statements, PROC CAUSALTRT estimates causal treatment effects by four different types of methods:

- specification of effects in both the PSMODEL and MODEL statements: doubly robust methods such as the augmented inverse probability weighting method and the inverse probability weighted regression adjustment method
- specification of effects only in the PSMODEL statement: three types of inverse probability weighting methods—basic, with ratio adjustment, and with ratio and scale adjustments
- specification of effects only in the MODEL statement: regression adjustment method
- specification of effects in neither the PSMODEL nor MODEL statement: unadjusted effect by computing the observed outcome mean difference between the treatment and control groups

The CAUSALTRT procedure can estimate two types of causal effects: the average treatment effect (ATE) and the average treatment effect for the treated (ATT). However, not all estimation methods can estimate both types of causal effects.

Other main features of PROC CAUSALTRT include the following:

- a class of generalized linear models for modeling continuous and binary outcomes
- asymptotic and bootstrap methods for computing standard errors and confidence intervals
- diagnostic plots for the propensity score model, including various plots of propensity scores or weights
- output data sets for propensity scores, inverse probability weights, and the predicted potential outcomes

For more information about PROC CAUSALTRT, see Chapter 39, "The CAUSALTRT Procedure," and Lamm and Yung (2017).

PROC PSMATCH

The PSMATCH procedure provides a variety of tools for propensity score analysis (Rosenbaum and Rubin 1983). It outputs data sets that contain pertinent propensity score information that you can use in subsequent outcome analyses to estimate causal effects.

PROC PSMATCH conducts propensity score analysis of a binary treatment variable T (say, T=1 indicates the treatment level), which is posited to have a causal effect on an outcome variable Y. You fit a propensity score model to the data so as to predict the propensity scores from a set of pretreatment (or baseline) characteristics. A commonly used form of the propensity score model is logistic regression, which is also assumed by PROC PSMATCH. For an introduction to propensity score analysis, see Guo and Fraser (2015).

For example, the following statements specify a propensity score analysis of the treatment variable Music (which indicates musical training in subjects):

```
proc psmatch data=School;
   class Music Gender;
   psmodel Music = Gender Absence;
   match method=optimal;
   output out(obs=match)=OutMatch;
run;
```

In the PSMODEL statement, the propensity score model specifies that the probability of receiving musical training is determined by the pretreatment characteristics Gender and Absence. In the MATCH statement, you request an optimal one-to-one matching of subjects between the treated and control groups. PROC PSMATCH then selects subsets of observations from the original data so that the selected observations in the treated and control groups match their propensity score distributions as closely as possible.

In the OUTPUT statement, you request that PROC PSMATCH output a data set called OutMatch. This output data set contains the original data as well as information about the matched subjects. For the current example, PROC PSMATCH creates a weighting variable named _MATCHWGT_ to indicate the matched subjects. You can then use this data set to estimate the causal treatment effect of Music on the outcome variable of interest.

For example, you can use PROC TTEST to estimate the causal treatment effect of Music on GPA (representing academic performance) and to test the statistical significance of the estimated causal effect by specifying the following statements:

```
proc ttest data=OutMatch;
   class Music;
   var GPA;
   weight _MATCHWGT_;
run;
```

PROC PSMATCH implements various propensity score methods, which are summarized as follows:

- propensity score matching method: PROC PSMATCH outputs matched observation weights for all subjects; subjects that are not matched are indicated by zero weights. Each set of matched subjects in the treated and control groups is also indicated by distinct identification numbers.
- propensity score weighting method: PROC PSMATCH outputs weights that are computed on the basis of the predicted propensity scores for all subjects.

- propensity score stratification method: PROC PSMATCH outputs stratum identification numbers for subjects. It also provides stratum weights that you can use to combine causal effect estimates from separate outcome analyses for the strata that PROC PSMATCH creates.

In all these propensity score methods, PROC PSMATCH outputs weights that you need to use in subsequent outcome analyses for estimating causal treatment effects. In addition, PROC PSMATCH computes different types of weights that are appropriate for computing the average treatment effect (ATE) and the average treatment effect for the treated (ATT), respectively.

A very important step in propensity score analysis is to evaluate the balance in covariates after you fit a propensity score model. PROC PSMATCH provides many numerical and graphical tools that you can use to assess the balance, including these:

- standardized mean differences between treated and control groups in the covariates
- percentage reductions of absolute mean differences after matching or weighting
- comparisons of distributions of covariates and propensity scores before and after matching or weighting

When you apply a propensity score–based adjustment method that is derived from a good propensity model, you expect good balance between the treated and control groups for all pretreatment covariates. If you are not satisfied with the covariate balance, refitting the propensity score model by other statistical strategies would be needed.

Other main features of the PSMATCH include the following:

- input of propensity scores that are estimated by methods outside PROC PSMATCH (for more information about inputting propensity scores, see the PSDATA statement of Chapter 101, "The PSMATCH Procedure")
- various matching methods: greedy nearest-neighbor matching, optimal matching, and matching with replacement

For more information about PROC PSMATCH, see Chapter 101, "The PSMATCH Procedure," and Yuan, Yung, and Stokes (2017).

Using the Causal Analysis Procedures

The causal analysis procedures that are described in this chapter help solve different kinds of causal inference problems. Certainly, this does not mean that you would use only one of these procedures exclusively. This section discusses how you might use these causal analysis procedures together and how these procedures work with other procedures to facilitate causal effect estimation.

Using PROC CAUSALGRAPH before Using Other Causal Analysis Procedures

Loosely speaking, being able to identify a causal effect means that you have a *valid* statistical strategy that infers or estimates the causal effect from the data, whether you collect them from observational studies or randomized experiments. When you use the CAUSALTRT or CAUSALMED procedure to estimate various types of causal effects or when you use the PSMATCH procedure to conduct propensity score modeling for matching, weighting, or stratification, you essentially assume the identification of the causal effect in question.

Practically, the identification assumption in these procedures is about whether you have measured and included a valid set of covariates in the regression model of the outcome variable and/or in the propensity score model for the treatment variable. In many situations, researchers might simply assume that a set of background (pretreatment) characteristics is a valid set of covariates in the model(s).

However, from the perspective of causal graph theory, the inclusion of all presumed background characteristics might not be an efficient or even valid modeling strategy to estimate causal effects. The primary purpose of PROC CAUSALGRAPH is to find valid modeling strategies or test the validity of a proposed strategy. When researchers can qualitatively specify the data generating process (including unmeasured variables and/or latent constructs) that explains how the treatment and outcome variables are related in the data, PROC CAUSALGRAPH can output valid measured covariate sets for regression adjustment or propensity score modeling.

Therefore, when you can reasonably assume the qualitative structure of the underlying data generating process, it is recommended that you use PROC CAUSALGRAPH to study the identification of causal effects before you use other causal analysis procedures. For illustrations of such a practice, see the examples in Thompson, Lamm, and Yung (2020) and Example 2 in Lamm, Thompson, and Yung (2019).

Using PROC PSMATCH Repeatedly to Ensure Proper Covariate Balancing

You use a propensity score model to estimate the probability that an individual receives a treatment. Then you use the estimated probabilities to create a matched, weighted, or stratified data set to which you apply a subsequent outcome analysis to estimate the causal effect of interest. Thus, a valid propensity model and an appropriate propensity score–based adjustment is critical to a valid estimation of the causal effect.

Checking covariate balance that results from a propensity score–based method (matching, weighting, or stratification) is perhaps the most common practice to justify the appropriateness of the propensity score–based adjustment. Although PROC PSMATCH provides many numerical and graphical tools for checking covariate balance, you can apply only a single logistic regression to model the propensity scores in a given analysis. If the covariate balance is not satisfactory, you might want to try other propensity score models that can lead to better covariate balancing. There are several possibilities for exploring suitable propensity score models:

- Add higher-order effects to the existing logistic regression of propensity scores.
- Replace the covariates in the existing logistic regression of propensity scores with a new set of covariates.
- Model the propensity scores by a technique other than logistic regression, possibly with a new set of covariates and/or a new set of covariate effects.

You can use PROC PSMATCH to explore the first two possibilities, but you must rely on other statistical procedures to explore the third possibility. Fitting a propensity score model in an external procedure might also be necessary when there is complete or quasi-complete separation of the data and the maximum likelihood estimates for the logistic regression model do not exist or are not unique.

When these other procedures produce the estimated propensity scores, you can input the propensity scores into PROC PSMATCH by specifying the PSDATA statement. PROC PSMATCH then examines the covariate balance by using these input propensity scores and completes the remaining tasks of the specified propensity score analysis. For illustrations of using the PSDATA statement, see Example 101.8 of Chapter 101, "The PSMATCH Procedure," and Example 1 of Lamm, Thompson, and Yung (2019).

Using PROC CAUSALTRT or PROC PSMATCH to Estimate Causal Effects

Both PROC CAUSALTRT and PROC PSMATCH use the logistic regression model to conduct propensity score analyses. If you specify the same propensity score model for these two procedures, you will get the same set of model fitting results and hence the same set of estimated propensity scores. Both procedures enable you to estimate the average treatment effect (ATE) and the average treatment effect for the treated (ATT). In addition, both procedures provide numerical and graphical tools for examining covariate balancing. Nonetheless, the two procedures have several distinct features that would suit different modeling situations.

You should consider using PROC CAUSALTRT in the following situations:

- You have a reliable outcome model to use for estimating causal effects by either an outcome regression method or a doubly robust method (for more information, see the METHOD= option and the MODEL statement in Chapter 39, "The CAUSALTRT Procedure").

- You are confident about the appropriateness of the specified propensity score model and want to estimate the standard errors of causal effects by taking the estimation of propensity scores into account.

You should consider using PROC PSMATCH in the following situations:

- You need to use the propensity score matching or stratification method (for more information, see the MATCH statement and the STRATA statement in Chapter 101, "The PSMATCH Procedure").
- You need to explore different propensity score models to achieve adequate covariate balance before conducting an outcome analysis (for more information, see the section "Using PROC PSMATCH Repeatedly to Ensure Proper Covariate Balancing" on page 144).

References

Berzuini, C., Dawid, P., and Bernardinelli, L., eds. (2012). *Causality: Statistical Perspectives and Applications*. Chichester, UK: John Wiley & Sons.

Guo, S., and Fraser, M. W. (2015). *Propensity Score Analysis: Statistical Methods and Applications*. 2nd ed. Thousand Oaks, CA: Sage Publications.

Hernán, M. A., and Robins, J. M. (2020). *Causal Inference: What If*. Boca Raton, FL: Chapman & Hall/CRC.

Imbens, G. W., and Rubin, D. B. (2015). *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. New York: Cambridge University Press.

Lamm, M., Thompson, W. C., and Yung, Y.-F. (2019). "Building a Propensity Score Model with SAS/STAT Software: Planning and Practice." In *Proceedings of the SAS Global Forum 2019 Conference*. Cary, NC: SAS Institute Inc. https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2019/3056-2019.pdf.

Lamm, M., and Yung, Y.-F. (2017). "Estimating Causal Effects from Observational Data with the CAUSALTRT Procedure." In *Proceedings of the SAS Global Forum 2017 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings17/SAS0374-2017.pdf.

Lunceford, J. K., and Davidian, M. (2004). "Stratification and Weighting via the Propensity Score in Estimation of Causal Treatment Effects: A Comparative Study." *Statistics in Medicine* 23:2937–2960.

Morgan, S. L., and Winship, C. (2015). *Counterfactuals and Causal Inference: Methods and Principles for Social Research*. 2nd ed. New York: Cambridge University Press.

Neyman, J., Dabrowska, D. M., and Speed, T. P. (1990). "On the Application of Probability Theory to Agricultural Experiments: Essay on Principles, Section 9." *Statistical Science* 5:465–472. Translated and edited by Dabrowska and Speed from the Polish original by Neyman (1923).

Pearl, J. (2001). "Direct and Indirect Effects." In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, edited by J. Breese and D. Koller, 411–420. San Francisco: Morgan Kaufmann.

Pearl, J. (2009a). "Causal Inference in Statistics: An Overview." *Statistics Surveys* 3:96–146.

Pearl, J. (2009b). *Causality: Models, Reasoning, and Inference*. 2nd ed. Cambridge: Cambridge University Press.

Robins, J. M., and Greenland, S. (1992). "Identifiability and Exchangeability for Direct and Indirect Effects." *Epidemiology* 3:143–155.

Rosenbaum, P. R., and Rubin, D. B. (1983). "The Central Role of the Propensity Score in Observational Studies for Causal Effects." *Biometrika* 70:41–55.

Rubin, D. B. (1990). "Comment: Neyman (1923) and Causal Inference in Experiments and Observational Studies." *Statistical Science* 5:472–480.

Thompson, W. C. (2019). "Causal Graph Analysis with the CAUSALGRAPH Procedure." In *Proceedings of the SAS Global Forum 2019 Conference*. Cary, NC: SAS Institute Inc. https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2019/2998-2019.pdf.

Thompson, W. C., Lamm, M., and Yung, Y.-F. (2020). "Causal Effect Estimands: Interpretation, Identification, and Computation." In *Proceedings of the SAS Global Forum 2020 Conference*. Cary, NC: SAS Institute Inc. https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2020/4322-2020.pdf.

Valeri, L., and VanderWeele, T. J. (2013). "Mediation Analysis Allowing for Exposure-Mediator Interactions and Causal Interpretation: Theoretical Assumptions and Implementation with SAS and SPSS Macros." *Psychological Methods* 18:137–150.

VanderWeele, T. J. (2014). "A Unification of Mediation and Interaction: A 4-Way Decomposition." *Epidemiology* 25:749–761.

VanderWeele, T. J. (2015). *Explanation in Causal Inference: Methods for Mediation and Interaction*. New York: Oxford University Press.

Yuan, Y., Yung, Y.-F., and Stokes, M. (2017). "Propensity Score Methods for Causal Inference with the PSMATCH Procedure." In *Proceedings of the SAS Global Forum 2017 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings17/SAS0332-2017.pdf.

Yung, Y.-F., Lamm, M., and Zhang, W. (2018). "Causal Mediation Analysis with the CAUSALMED Procedure." In *Proceedings of the SAS Global Forum 2018 Conference*. Cary, NC: SAS Institute Inc. https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2018/1991-2018.pdf.

Chapter 8
Introduction to Bayesian Analysis Procedures

Contents

Overview **147**
Introduction **148**
Background in Bayesian Statistics **150**
Prior Distributions 150
Bayesian Inference 152
Bayesian Analysis: Advantages and Disadvantages 154
Markov Chain Monte Carlo Method 155
Assessing Markov Chain Convergence 162
Summary Statistics 175
A Bayesian Reading List **178**
Introductory Books 178
Intermediate-Level Books 179
Advanced Titles 179
Books Motivated by Statistical Applications and Data Analysis 180
References **180**

Overview

SAS/STAT software provides Bayesian capabilities in seven procedures: BCHOICE, BGLIMM, FMM, GENMOD, LIFEREG, MCMC, and PHREG. The FMM, GENMOD, LIFEREG, and PHREG procedures provide Bayesian analysis in addition to the standard frequentist analyses that they have always performed. Thus, these procedures provide convenient access to Bayesian modeling and inference for finite mixture models, generalized linear models, accelerated life failure models, Cox regression models, and piecewise constant baseline hazard models (also known as piecewise exponential models). The BCHOICE procedure provides Bayesian analysis for discrete choice models. The BGLIMM procedure performs Bayesian analysis for generalized linear mixed-effects models (GLMMs). The MCMC procedure is a general procedure that fits a wide-range of Bayesian models that can contain, for example, arbitrary priors and likelihood functions.

This chapter provides an overview of Bayesian statistics; describes specific sampling algorithms used in these procedures; and discusses posterior inference and convergence diagnostics computations. Sources that provide in-depth treatment of Bayesian statistics can be found at the end of this chapter, in the section "A Bayesian Reading List" on page 178. Additional chapters contain syntax, details, and examples for the individual procedures BCHOICE (see Chapter 30, "The BCHOICE Procedure"), BGLIMM (see Chapter 31, "The BGLIMM Procedure"), FMM (see Chapter 46, "The FMM Procedure"), GENMOD (see Chapter 51, "The GENMOD Procedure"), LIFEREG (see Chapter 76, "The LIFEREG Procedure"), MCMC (see Chapter 80, "The MCMC Procedure"), and PHREG (see Chapter 92, "The PHREG Procedure").

Introduction

The most frequently used statistical methods are known as *frequentist* (or *classical*) methods. These methods assume that unknown parameters are fixed constants, and they define probability by using limiting relative frequencies. It follows from these assumptions that probabilities are objective and that you cannot make probabilistic statements about parameters because they are fixed. Bayesian methods offer an alternative approach; they treat parameters as random variables and define probability as "degrees of belief" (that is, the probability of an event is the degree to which you believe the event is true). It follows from these postulates that probabilities are subjective and that you can make probability statements about parameters. The term "Bayesian" comes from the prevalent usage of Bayes' theorem, which was named after the Reverend Thomas Bayes, an eighteenth century Presbyterian minister. Bayes was interested in solving the question of inverse probability: after observing a collection of events, what is the probability of one event?

Suppose you are interested in estimating θ from data $\mathbf{y} = \{y_1, \ldots, y_n\}$ by using a statistical model described by a density $p(\mathbf{y}|\theta)$. Bayesian philosophy states that θ cannot be determined exactly, and uncertainty about the parameter is expressed through probability statements and distributions. You can say that θ follows a normal distribution with mean 0 and variance 1, if it is believed that this distribution best describes the uncertainty associated with the parameter. The following steps describe the essential elements of Bayesian inference:

1. A probability distribution for θ is formulated as $\pi(\theta)$, which is known as the *prior* distribution, or just the prior. The prior distribution expresses your beliefs (for example, on the mean, the spread, the skewness, and so forth) about the parameter before you examine the data.
2. Given the observed data $\mathbf{y}$, you choose a statistical model $p(\mathbf{y}|\theta)$ to describe the distribution of $\mathbf{y}$ given θ.
3. You update your beliefs about θ by combining information from the prior distribution and the data through the calculation of the *posterior* distribution, $p(\theta|\mathbf{y})$.

The third step is carried out by using Bayes' theorem, which enables you to combine the prior distribution and the model in the following way:

$$p(\theta|\mathbf{y}) = \frac{p(\theta, \mathbf{y})}{p(\mathbf{y})} = \frac{p(\mathbf{y}|\theta)\pi(\theta)}{p(\mathbf{y})} = \frac{p(\mathbf{y}|\theta)\pi(\theta)}{\int p(\mathbf{y}|\theta)\pi(\theta)d\theta}$$

The quantity

$$p(\mathbf{y}) = \int p(\mathbf{y}|\theta)\pi(\theta)d\theta$$

is the normalizing constant of the posterior distribution. This quantity $p(\mathbf{y})$ is also the marginal distribution of $\mathbf{y}$, and it is sometimes called the marginal distribution of the data. The likelihood function of θ is any function proportional to $p(\mathbf{y}|\theta)$; that is, $L(\theta) \propto p(\mathbf{y}|\theta)$. Another way of writing Bayes' theorem is as follows:

$$p(\theta|\mathbf{y}) = \frac{L(\theta)\pi(\theta)}{\int L(\theta)\pi(\theta)d\theta}$$

The marginal distribution $p(\mathbf{y})$ is an integral. As long as the integral is finite, the particular value of the integral does not provide any additional information about the posterior distribution. Hence, $p(\theta|\mathbf{y})$ can be written up to an arbitrary constant, presented here in proportional form as:

$$p(\theta|\mathbf{y}) \propto L(\theta)\pi(\theta)$$

Simply put, Bayes' theorem tells you how to update existing knowledge with new information. You begin with a prior belief $\pi(\theta)$, and after learning information from data $\mathbf{y}$, you change or update your belief about θ and obtain $p(\theta|\mathbf{y})$. These are the essential elements of the Bayesian approach to data analysis.

In theory, Bayesian methods offer simple alternatives to statistical inference—all inferences follow from the posterior distribution $p(\theta|\mathbf{y})$. In practice, however, you can obtain the posterior distribution with straightforward analytical solutions only in the most rudimentary problems. Most Bayesian analyses require sophisticated computations, including the use of simulation methods, such as the Markov chain Monte Carlo (MCMC) methods. You generate samples from the posterior distribution and use these samples to estimate the quantities of interest. PROC MCMC uses a self-tuning Metropolis algorithm (see the section "Metropolis and Metropolis-Hastings Algorithms" on page 156). The GENMOD, LIFEREG, and PHREG procedures use the Gibbs sampler (see the section "Gibbs Sampler" on page 157). The BCHOICE and FMM procedure use a combination of the Gibbs sampler and the latent variable sampler. The BGLIMM procedure uses the Gibbs sampler, the Hamiltonian Monte Carlo sampler, and the Gamerman algorithm for generalized linear mixed-effects models. An important aspect of any analysis is assessing the convergence of the Markov chains. Inferences that are based on nonconverged Markov chains can be both inaccurate and misleading.

Both Bayesian and classical methods have their advantages and disadvantages. From a practical point of view, your choice of method depends on what you want to accomplish with your data analysis. If you have prior information (either expert opinion or historical knowledge) that you want to incorporate into the analysis, then you should consider Bayesian methods. In addition, if you want to communicate your findings in terms of probability notions that can be more easily understood by nonstatisticians, Bayesian methods might be appropriate. The Bayesian paradigm can often provide a framework for answering specific scientific questions that a single point estimate cannot sufficiently address. Alternatively, if you are interested only in estimating parameters based on the likelihood, then numerical optimization methods, such as the Newton-Raphson method, can give you very precise estimates and there is no need to use a Bayesian analysis. For further discussions of the relative advantages and disadvantages of Bayesian analysis, see the section "Bayesian Analysis: Advantages and Disadvantages" on page 154.

Background in Bayesian Statistics

Prior Distributions

A prior distribution of a parameter is the probability distribution that represents your uncertainty about the parameter before the current data are examined. Multiplying the prior distribution and the likelihood function together leads to the posterior distribution of the parameter. You use the posterior distribution to carry out all inferences. You cannot carry out any Bayesian inference or perform any modeling without using a prior distribution.

Objective Priors versus Subjective Priors

Bayesian probability measures the degree of belief that you have in a random event. By this definition, probability is highly subjective. It follows that all priors are *subjective priors*. Not everyone agrees with this notion of subjectivity when it comes to specifying prior distributions. There has long been a desire to obtain results that are objectively valid. Within the Bayesian paradigm, this can be somewhat achieved by using prior distributions that are "objective" (that is, that have a minimal impact on the posterior distribution). Such distributions are called *objective* or *noninformative* priors (see the next section). However, while noninformative priors are very popular in some applications, they are not always easy to construct. See DeGroot and Schervish (2002, Section 1.2) and Press (2003, Section 2.2) for more information about interpretations of probability. See Berger (2006) and Goldstein (2006) for discussions about objective Bayesian versus subjective Bayesian analysis.

Noninformative Priors

Roughly speaking, a prior distribution is noninformative if the prior is "flat" relative to the likelihood function. Thus, a prior $\pi(\theta)$ is noninformative if it has minimal impact on the posterior distribution of θ. Other names for the noninformative prior are *vague*, *diffuse*, and *flat* prior. Many statisticians favor noninformative priors because they appear to be more objective. However, it is unrealistic to expect that noninformative priors represent total ignorance about the parameter of interest. In some cases, noninformative priors can lead to *improper posteriors* (nonintegrable posterior density). You cannot make inferences with improper posterior distributions. In addition, noninformative priors are often not invariant under transformation; that is, a prior might be noninformative in one parameterization but not necessarily noninformative if a transformation is applied.

See Box and Tiao (1973) for a more formal development of noninformative priors. See Kass and Wasserman (1996) for techniques for deriving noninformative priors.

Improper Priors

A prior $\pi(\theta)$ is said to be improper if

$$\int \pi(\theta) d\theta = \infty$$

For example, a uniform prior distribution on the real line, $\pi(\theta) \propto 1$, for $-\infty < \theta < \infty$, is an improper prior. Improper priors are often used in Bayesian inference since they usually yield noninformative priors

and proper posterior distributions. Improper prior distributions can lead to posterior impropriety (improper posterior distribution). To determine whether a posterior distribution is proper, you need to make sure that the normalizing constant $\int p(\mathbf{y}|\theta)p(\theta)d\theta$ is finite for all $\mathbf{y}$. If an improper prior distribution leads to an improper posterior distribution, inference based on the improper posterior distribution is invalid.

The BCHOICE, BGLIMM, GENMOD, LIFEREG, and PHREG procedures allow the use of improper priors—that is, the flat prior on the real line—for regression coefficients. These improper priors do not lead to any improper posterior distributions in the models that these procedures fit. PROC MCMC allows the use of any prior, as long as the distribution is programmable using DATA step functions. However, PROC MCMC does not verify that the posterior distribution is integrable. You must verify this yourself. For PROC BGLIMM, a GLMM model contains various types of parameters: the coefficients for the fixed effects; the coefficients for the random effects; and the parameters for the covariance matrices, including both the G-side and R-side covariance matrices. When an improper prior is used for parameters in a covariance matrix, PROC BGLIMM does not check whether the posterior is proper. For more information, see the section "Prior Distributions" on page 1344 of the BGLIMM procedure chapter.

Informative Priors

An informative prior is a prior that is not dominated by the likelihood and that has an impact on the posterior distribution. If a prior distribution dominates the likelihood, it is clearly an informative prior. These types of distributions must be specified with care in actual practice. On the other hand, the proper use of prior distributions illustrates the power of the Bayesian method: information gathered from the previous study, past experience, or expert opinion can be combined with current information in a natural way. See the "Examples" sections of the GENMOD and PHREG procedure chapters for instructions about constructing informative prior distributions. Most other Bayesian procedures have discussion about informative priors in their documentation.

Conjugate Priors

A prior is said to be a conjugate prior for a family of distributions if the prior and posterior distributions are from the same family, which means that the form of the posterior has the same distributional form as the prior distribution. For example, if the likelihood is binomial, $y \sim \mathrm{Bin}(n, \theta)$, a conjugate prior on θ is the beta distribution; it follows that the posterior distribution of θ is also a beta distribution. Other commonly used conjugate prior/likelihood combinations include the normal/normal, gamma/Poisson, gamma/gamma, and gamma/beta cases. The development of conjugate priors was partially driven by a desire for computational convenience—conjugacy provides a practical way to obtain the posterior distributions. The BCHOICE, BGLIMM, and MCMC procedures use conjugate sampling whenever applicable. The other Bayesian procedures do not use conjugacy in posterior sampling.

Jeffreys' Prior

A very useful prior is Jeffreys' prior (Jeffreys 1961). It satisfies the local uniformity property: a prior that does not change much over the region in which the likelihood is significant and does not assume large values outside that range. It is based on the Fisher information matrix. Jeffreys' prior is defined as

$$\pi(\theta) \propto |I(\theta)|^{1/2}$$

where $|\ |$ denotes the determinant and $I(\theta)$ is the Fisher information matrix based on the likelihood function $p(\mathbf{y}|\theta)$:

$$I(\theta) = -E\left[\frac{\partial^2 \log p(\mathbf{y}|\theta)}{\partial\theta^2}\right]$$

Jeffreys' prior is locally uniform and hence noninformative. It provides an automated scheme for finding a noninformative prior for any parametric model $p(\mathbf{y}|\theta)$. Another appealing property of Jeffreys' prior is that it is invariant with respect to one-to-one transformations. The invariance property means that if you have a locally uniform prior on θ and $\phi(\theta)$ is a one-to-one function of θ, then $p(\phi(\theta)) = \pi(\theta) \cdot |\phi'(\theta)|^{-1}$ is a locally uniform prior for $\phi(\theta)$. This invariance principle carries through to multidimensional parameters as well. While Jeffreys' prior provides a general recipe for obtaining noninformative priors, it has some shortcomings: the prior is improper for many models, and it can lead to improper posterior in some cases; and the prior can be cumbersome to use in high dimensions. PROC GENMOD calculates Jeffreys' prior automatically for any generalized linear model. You can set it as your prior density for the coefficient parameters, and it does not lead to improper posteriors. You can construct Jeffreys' prior for a variety of statistical models in the MCMC procedure. See the section "Example 80.4: Logistic Regression Model with Jeffreys' Prior" on page 6400 in Chapter 80, "The MCMC Procedure," for an example. PROC MCMC does not guarantee that the corresponding posterior distribution is proper, and you need to exercise extra caution in this case.

Bayesian Inference

Bayesian inference about θ is primarily based on the posterior distribution of θ. There are various ways in which you can summarize this distribution. For example, you can report your findings through point estimates. You can also use the posterior distribution to construct hypothesis tests or probability statements.

Point Estimation and Estimation Error

Classical methods often report the maximum likelihood estimator (MLE) or the method of moments estimator (MOME) of a parameter. In contrast, Bayesian approaches often use the posterior mean. The definition of the posterior mean is given by

$$E(\theta|\mathbf{y}) = \int \theta \; p(\theta|\mathbf{y}) \; d\theta$$

Other commonly used posterior estimators include the posterior median, defined as

$$\theta\colon P(\theta \geq \text{median}|\mathbf{y}) = P(\theta \leq \text{median}|\mathbf{y}) = \frac{1}{2}$$

and the posterior mode, defined as the value of θ that maximizes $p(\theta|\mathbf{y})$.

The variance of the posterior density (simply referred to as the *posterior variance*) describes the uncertainty in the parameter, which is a random variable in the Bayesian paradigm. A Bayesian analysis typically uses the posterior variance, or the posterior standard deviation, to characterize the dispersion of the parameter. In multidimensional models, covariance or correlation matrices are used.

If you know the distributional form of the posterior density of interest, you can report the exact posterior point estimates. When models become too difficult to analyze analytically, you have to use simulation algorithms, such as the MCMC method to obtain posterior estimates (see the section "Markov Chain Monte Carlo Method" on page 155). All of the Bayesian procedures rely on MCMC to obtain all posterior estimates. Using only a finite number of samples, simulations introduce an additional level of uncertainty to the accuracy of the estimates. *Monte Carlo standard error (MCSE)*, which is the standard error of the posterior mean estimate, measures the simulation accuracy. See the section "Standard Error of the Mean Estimate" on page 176 for more information.

The posterior standard deviation and the MCSE are two completely different concepts: the posterior standard deviation describes the uncertainty in the parameter, while the MCSE describes only the uncertainty in the parameter estimate as a result of MCMC simulation. The posterior standard deviation is a function of the sample size in the data set, and the MCSE is a function of the number of iterations in the simulation.

Hypothesis Testing

Suppose you have the following null and alternative hypotheses: H_0 is $\theta \in \Theta_0$ and H_1 is $\theta \in \Theta_0^c$, where Θ_0 is a subset of the parameter space and Θ_0^c is its complement. Using the posterior distribution $\pi(\theta|\mathbf{y})$, you can compute the posterior probabilities $P(\theta \in \Theta_0|\mathbf{y})$ and $P(\theta \in \Theta_0^c|\mathbf{y})$, or the probabilities that H_0 and H_1 are true, respectively. One way to perform a Bayesian hypothesis test is to accept the null hypothesis if $P(\theta \in \Theta_0|\mathbf{y}) \geq P(\theta \in \Theta_0^c|\mathbf{y})$ and vice versa, or to accept the null hypothesis if $P(\theta \in \Theta_0|\mathbf{y})$ is greater than a predefined threshold, such as 0.75, to guard against falsely accepted null distribution.

It is more difficult to carry out a point null hypothesis test in a Bayesian analysis. A point null hypothesis is a test of $H_0: \theta = \theta_0$ versus $H_1: \theta \neq \theta_0$. If the prior distribution $\pi(\theta)$ is a continuous density, then the posterior probability of the null hypothesis being true is 0, and there is no point in carrying out the test. One alternative is to restate the null to be a small interval hypothesis: $\theta \in \Theta_0 = (\theta_0 - a, \theta_0 + a)$, where a is a very small constant. The Bayesian paradigm can deal with an interval hypothesis more easily. Another approach is to give a mixture prior distribution to θ with a positive probability of p_0 on θ_0 and the density $(1 - p_0)\pi(\theta)$ on $\theta \neq \theta_0$. This prior ensures a nonzero posterior probability on θ_0, and you can then make realistic probabilistic comparisons. For more detailed treatment of Bayesian hypothesis testing, see Berger (1985).

Interval Estimation

The Bayesian set estimates are called *credible sets*, which are also known as *credible intervals*. This is analogous to the concept of confidence intervals used in classical statistics. Given a posterior distribution $p(\theta|\mathbf{y})$, A is a credible set for θ if

$$P(\theta \in A|\mathbf{y}) = \int_A p(\theta|\mathbf{y})d\theta$$

For example, you can construct a 95% credible set for θ by finding an interval, A, over which $\int_A p(\theta|\mathbf{y}) = 0.95$.

You can construct credible sets that have equal tails. A $100(1-\alpha)\%$ equal-tail interval corresponds to the $100(\alpha/2)$th and $100(1-\alpha/2)$th percentiles of the posterior distribution. Some statisticians prefer this interval because it is invariant under transformations. Another frequently used Bayesian credible set is called the *highest posterior density* (HPD) interval.

A $100(1-\alpha)\%$ HPD interval is a region that satisfies the following two conditions:

1. The posterior probability of that region is $100(1-\alpha)\%$.
2. The minimum density of any point within that region is equal to or larger than the density of any point outside that region.

The HPD is an interval in which most of the distribution lies. Some statisticians prefer this interval because it is the smallest interval.

One major distinction between Bayesian and classical sets is their interpretation. The Bayesian probability reflects a person's subjective beliefs. Following this approach, a statistician can make the claim that θ is inside a credible interval with measurable probability. This property is appealing because it enables you to make a direct probability statement about parameters. Many people find this concept to be a more natural way of understanding a probability interval, which is also easier to explain to nonstatisticians. A confidence interval, on the other hand, enables you to make a claim that the interval covers the true parameter. The interpretation reflects the uncertainty in the sampling procedure; a confidence interval of $100(1-\alpha)\%$ asserts that, in the long run, $100(1-\alpha)\%$ of the realized confidence intervals cover the true parameter.

Bayesian Analysis: Advantages and Disadvantages

Bayesian methods and classical methods both have advantages and disadvantages, and there are some similarities. When the sample size is large, Bayesian inference often provides results for parametric models that are very similar to the results produced by frequentist methods. Some advantages to using Bayesian analysis include the following:

- It provides a natural and principled way of combining prior information with data, within a solid decision theoretical framework. You can incorporate past information about a parameter and form a prior distribution for future analysis. When new observations become available, the previous posterior distribution can be used as a prior. All inferences logically follow from Bayes' theorem.
- It provides inferences that are conditional on the data and are exact, without reliance on asymptotic approximation. Small sample inference proceeds in the same manner as if one had a large sample. Bayesian analysis also can estimate any functions of parameters directly, without using the "plug-in" method (a way to estimate functionals by plugging the estimated parameters in the functionals).
- It obeys the likelihood principle. If two distinct sampling designs yield proportional likelihood functions for θ, then all inferences about θ should be identical from these two designs. Classical inference does not in general obey the likelihood principle.
- It provides interpretable answers, such as "the true parameter θ has a probability of 0.95 of falling in a 95% credible interval."
- It provides a convenient setting for a wide range of models, such as hierarchical models and missing data problems. MCMC, along with other numerical methods, makes computations tractable for virtually all parametric models.

There are also disadvantages to using Bayesian analysis:

- It does not tell you how to select a prior. There is no correct way to choose a prior. Bayesian inferences require skills to translate subjective prior beliefs into a mathematically formulated prior. If you do not proceed with caution, you can generate misleading results.
- It can produce posterior distributions that are heavily influenced by the priors. From a practical point of view, it might sometimes be difficult to convince subject matter experts who do not agree with the validity of the chosen prior.
- It often comes with a high computational cost, especially in models with a large number of parameters. In addition, simulations provide slightly different answers unless the same random seed is used. Note that slight variations in simulation results do not contradict the early claim that Bayesian inferences are exact. The posterior distribution of a parameter is exact, given the likelihood function and the priors, while simulation-based estimates of posterior quantities can vary due to the random number generator used in the procedures.

For more in-depth treatments of the pros and cons of Bayesian analysis, see Berger (1985, Sections 4.1 and 4.12), Berger and Wolpert (1988), Bernardo and Smith (1994, with a new edition coming out), Carlin and Louis (2000, Section 1.4), Robert (2001, Chapter 11), and Wasserman (2004, Section 11.9).

The following sections provide detailed information about the Bayesian methods provided in SAS.

Markov Chain Monte Carlo Method

The Markov chain Monte Carlo (MCMC) method is a general simulation method for sampling from posterior distributions and computing posterior quantities of interest. MCMC methods sample successively from a target distribution. Each sample depends on the previous one, hence the notion of the Markov chain. A Markov chain is a sequence of random variables, $\theta^1, \theta^2, \ldots$, for which the random variable θ^t depends on all previous θs only through its immediate predecessor θ^{t-1}. You can think of a Markov chain applied to sampling as a mechanism that traverses randomly through a target distribution without having any memory of where it has been. Where it moves next is entirely dependent on where it is now.

Monte Carlo, as in Monte Carlo integration, is mainly used to approximate an expectation by using the Markov chain samples. In the simplest version

$$\int_S g(\theta)p(\theta)d\theta \cong \frac{1}{n}\sum_{t=1}^{n} g(\theta^t)$$

where $g(\cdot)$ is a function of interest and θ^t are samples from $p(\theta)$ on its support S. This approximates the expected value of $g(\theta)$. The earliest reference to MCMC simulation occurs in the physics literature. Metropolis and Ulam (1949) and Metropolis et al. (1953) describe what is known as the Metropolis algorithm (see the section "Metropolis and Metropolis-Hastings Algorithms" on page 156). The algorithm can be used to generate sequences of samples from the joint distribution of multiple variables, and it is the foundation of MCMC. Hastings (1970) generalized their work, resulting in the Metropolis-Hastings algorithm. Geman and Geman (1984) analyzed image data by using what is now called Gibbs sampling (see the section "Gibbs Sampler" on page 157). These MCMC methods first appeared in the mainstream statistical literature in Tanner and Wong (1987).

The Markov chain method has been quite successful in modern Bayesian computing. Only in the simplest Bayesian models can you recognize the analytical forms of the posterior distributions and summarize inferences directly. In moderately complex models, posterior densities are too difficult to work with directly. With the MCMC method, it is possible to generate samples from an arbitrary posterior density $p(\theta|\mathbf{y})$ and to use these samples to approximate expectations of quantities of interest. Several other aspects of the Markov chain method also contributed to its success. Most importantly, if the simulation algorithm is implemented correctly, the Markov chain is guaranteed to converge to the target distribution $p(\theta|\mathbf{y})$ under rather broad conditions, regardless of where the chain was initialized. In other words, a Markov chain is able to improve its approximation to the true distribution at each step in the simulation. Furthermore, if the chain is run for a very long time (often required), you can recover $p(\theta|\mathbf{y})$ to any precision. Also, the simulation algorithm is easily extensible to models with a large number of parameters or high complexity, although the "curse of dimensionality" often causes problems in practice.

Properties of Markov chains are discussed in Feller (1968), Breiman (1968), and Meyn and Tweedie (1993). Ross (1997) and Karlin and Taylor (1975) give a non-measure-theoretic treatment of stochastic processes, including Markov chains. For conditions that govern Markov chain convergence and rates of convergence, see Amit (1991), Applegate, Kannan, and Polson (1990), Chan (1993), Geman and Geman (1984), Liu, Wong, and Kong (1991a, b), Rosenthal (1991a, b), Tierney (1994), and Schervish and Carlin (1992). Besag (1974) describes conditions under which a set of conditional distributions gives a unique joint distribution. Tanner (1993), Gilks, Richardson, and Spiegelhalter (1996), Chen, Shao, and Ibrahim (2000), Liu (2001), Gelman et al. (2004), Robert and Casella (2004), and Congdon (2001, 2003, 2005) provide both theoretical and applied treatments of MCMC methods. You can also see the section "A Bayesian Reading List" on page 178 for a list of books with varying levels of difficulty of treatment of the subject and its application to Bayesian statistics.

Metropolis and Metropolis-Hastings Algorithms

The Metropolis algorithm is named after its inventor, the American physicist and computer scientist Nicholas C. Metropolis. The algorithm is simple but practical, and it can be used to obtain random samples from any arbitrarily complicated target distribution of any dimension that is known up to a normalizing constant.

Suppose you want to obtain T samples from a univariate distribution with probability density function $f(\theta|\mathbf{y})$. Suppose θ^t is the tth sample from f. To use the Metropolis algorithm, you need to have an initial value θ^0 and a symmetric *proposal* density $q(\theta^{t+1}|\theta^t)$. For the $(t+1)$ iteration, the algorithm generates a sample from $q(\cdot|\cdot)$ based on the current sample θ^t, and it makes a decision to either accept or reject the new sample. If the new sample is accepted, the algorithm repeats itself by starting at the new sample. If the new sample is rejected, the algorithm starts at the current point and repeats. The algorithm is self-repeating, so it can be carried out as long as required. In practice, you have to decide the total number of samples needed in advance and stop the sampler after that many iterations have been completed.

Suppose $q(\theta_{\text{new}}|\theta^t)$ is a symmetric distribution. The proposal distribution should be an easy distribution from which to sample, and it must be such that $q(\theta_{\text{new}}|\theta^t) = q(\theta^t|\theta_{\text{new}})$, meaning that the likelihood of jumping to θ_{new} from θ^t is the same as the likelihood of jumping back to θ^t from θ_{new}. The most common choice of the proposal distribution is the normal distribution $N(\theta^t, \sigma)$ with a fixed σ. The Metropolis algorithm can be summarized as follows:

1. Set $t = 0$. Choose a starting point θ^0. This can be an arbitrary point as long as $f(\theta^0|\mathbf{y}) > 0$.
2. Generate a new sample, θ_{new}, by using the proposal distribution $q(\cdot|\theta^t)$.

3. Calculate the following quantity:

$$r = \min\left\{\frac{f(\theta_{\text{new}}|\mathbf{y})}{f(\theta^t|\mathbf{y})}, 1\right\}$$

4. Sample u from the uniform distribution $U(0, 1)$.
5. Set $\theta^{t+1} = \theta_{\text{new}}$ if $u < r$; otherwise set $\theta^{t+1} = \theta^t$.
6. Set $t = t + 1$. If $t < T$, the number of desired samples, return to step 2. Otherwise, stop.

Note that the number of iteration keeps increasing regardless of whether a proposed sample is accepted.

This algorithm defines a chain of random variates whose distribution will converge to the desired distribution $f(\theta|\mathbf{y})$, and so from some point forward, the chain of samples is a sample from the distribution of interest. In Markov chain terminology, this distribution is called the *stationary distribution* of the chain, and in Bayesian statistics, it is the posterior distribution of the model parameters. The reason that the Metropolis algorithm works is beyond the scope of this documentation, but you can find more detailed descriptions and proofs in many standard textbooks, including Roberts (1996) and Liu (2001). The random-walk Metropolis algorithm is used in the MCMC procedure.

You are not limited to a symmetric random-walk proposal distribution in establishing a valid sampling algorithm. A more general form, the Metropolis-Hastings (MH) algorithm, was proposed by Hastings (1970). The MH algorithm uses an asymmetric proposal distribution: $q(\theta_{\text{new}}|\theta^t) \neq q(\theta^t|\theta_{\text{new}})$. The difference in its implementation comes in calculating the ratio of densities:

$$r = \min\left\{\frac{f(\theta_{\text{new}}|\mathbf{y})q(\theta^t|\theta_{\text{new}})}{f(\theta^t|\mathbf{y})q(\theta_{\text{new}}|\theta^t)}, 1\right\}$$

Other steps remain the same.

The extension of the Metropolis algorithm to a higher-dimensional $\boldsymbol{\theta}$ is straightforward. Suppose $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_k)'$ is the parameter vector. To start the Metropolis algorithm, select an initial value for each θ_k and use a multivariate version of proposal distribution $q(\cdot|\cdot)$, such as a multivariate normal distribution, to select a k-dimensional new parameter. Other steps remain the same as those previously described, and this Markov chain eventually converges to the target distribution of $f(\boldsymbol{\theta}|\mathbf{y})$. Chib and Greenberg (1995) provide a useful tutorial on the algorithm.

Gibbs Sampler

The Gibbs sampler, named by Geman and Geman (1984) after the American physicist Josiah W. Gibbs, is a special case of the "Metropolis and Metropolis-Hastings Algorithms" on page 156 in which the proposal distributions exactly match the posterior conditional distributions and proposals are accepted 100% of the time. Gibbs sampling requires you to decompose the joint posterior distribution into full conditional distributions for each parameter in the model and then sample from them. The sampler can be efficient when the parameters are not highly dependent on each other and the full conditional distributions are easy to sample from. Some researchers favor this algorithm because it does not require an instrumental proposal distribution

as Metropolis methods do. However, while deriving the conditional distributions can be relatively easy, it is not always possible to find an efficient way to sample from these conditional distributions.

Suppose $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)'$ is the parameter vector, $p(\mathbf{y}|\boldsymbol{\theta})$ is the likelihood, and $\pi(\boldsymbol{\theta})$ is the prior distribution. The full posterior conditional distribution of $\pi(\theta_i|\theta_j, i \neq j, \mathbf{y})$ is proportional to the joint posterior density; that is,

$$\pi(\theta_i|\theta_j, i \neq j, \mathbf{y}) \propto p(\mathbf{y}|\boldsymbol{\theta})\pi(\boldsymbol{\theta})$$

For instance, the one-dimensional conditional distribution of θ_1 given $\theta_j = \theta_j^*, 2 \leq j \leq k$, is computed as the following:

$$\pi(\theta_1|\theta_j = \theta_j^*, 2 \leq j \leq k, \mathbf{y}) = p(\mathbf{y}|(\boldsymbol{\theta} = (\theta_1, \theta_2^*, \ldots, \theta_k^*)')\pi(\boldsymbol{\theta} = (\theta_1, \theta_2^*, \ldots, \theta_k^*)')$$

The Gibbs sampler works as follows:

1. Set $t = 0$, and choose an arbitrary initial value of $\boldsymbol{\theta}^{(0)} = \{\theta_1^{(0)}, \ldots, \theta_k^{(0)}\}$.
2. Generate each component of $\boldsymbol{\theta}$ as follows:
   - Draw $\theta_1^{(t+1)}$ from $\pi(\theta_1|\theta_2^{(t)}, \ldots, \theta_k^{(t)}, \mathbf{y})$
   - Draw $\theta_2^{(t+1)}$ from $\pi(\theta_2|\theta_1^{(t+1)}, \theta_3^{(t)}, \ldots, \theta_k^{(t)}, \mathbf{y})$
   - ...
   - Draw $\theta_k^{(t+1)}$ from $\pi(\theta_k|\theta_1^{(t+1)}, \ldots, \theta_{k-1}^{(t+1)}, \mathbf{y})$
3. Set $t = t + 1$. If $t < T$, the number of desired samples, return to step 2. Otherwise, stop.

The name "Gibbs" was introduced by Geman and Geman (1984). Gelfand et al. (1990) first used Gibbs sampling to solve problems in Bayesian inference. See Casella and George (1992) for a tutorial on the sampler. Most of the Bayesian procedures update parameters by using the Gibbs sampler.

Adaptive Rejection Sampling Algorithm

The GENMOD, LIFEREG, and PHREG procedures use the adaptive rejection sampling (ARS) algorithm to sample parameters sequentially from their univariate full conditional distributions. The ARS algorithm is a rejection algorithm that was originally proposed by Gilks and Wild (1992). Given a log-concave density (the log of the density is concave), you can construct an envelope for the density by using linear segments. You then use the linear segment envelope as a proposal density (it becomes a piecewise exponential density on the original scale and is easy to generate samplers from) in the rejection sampling.

The log-concavity condition is met in some of the models that are fit by the procedures. For example, the posterior densities for the regression parameters in the generalized linear models are log-concave under flat priors. When this condition fails, the ARS algorithm calls for an additional Metropolis-Hastings step (Gilks, Best, and Tan 1995), and the modified algorithm becomes the adaptive rejection Metropolis sampling

(ARMS) algorithm. The GENMOD, LIFEREG, and PHREG procedures can recognize whether a model is log-concave and select the appropriate sampler for the problem at hand.

Although samples obtained from the ARMS algorithm often exhibit less dependence with lower autocorrelations, the algorithm could have a high computational cost because it requires repeated evaluations of the objective function (usually five to seven repetitions) at each iteration for each univariate parameter.[1]

Implementation the ARMS algorithm in the GENMOD, LIFEREG, and PHREG procedures is based on code that is provided by Walter R. Gilks, University of Leeds (Gilks 2003). For a detailed description and explanation of the algorithm, see Gilks and Wild (1992); Gilks, Best, and Tan (1995).

Slice Sampler

The slice sampler (Neal 2003), like the ARMS algorithm, is a general algorithm that can be used to sample parameters from their target distribution. As with the ARMS algorithm, the only requirement of the slice sampler is the ability to evaluate the objective function (the unnormalized conditional distribution in a Gibbs step, for example) at a given parameter value. In theory, you can draw a random number from any given distribution as long as you can first obtain a random number uniformly under the curve of that distribution. Treat the area under the curve of $p(\theta)$ as a two-dimensional space that is defined by the θ-axis and the Y-axis, the latter being the axis for the density function. You draw uniformly in that area, obtain a two-dimensional vector of (θ_i, y_i), ignore the y_i, and keep the θ_i. The θ_i's are distributed according to the right density.

To solve the problem of sampling uniformly under the curve, Neal (2003) proposed the idea of slices (hence the name of the sampler), which can be explained as follows:

1. Start the algorithm at θ_0.
2. Calculate the objective function $p(\theta_0)$ and draw a line between $y = 0$ and $y = p(\theta_0)$, which defines a vertical slice. You draw a uniform number, y_1, on this slice, between $(0, p(\theta_0))$.
3. Draw a horizontal line at y_1 and find the two points where the line intercepts with the curve, (L_1, R_1). These two points define a horizontal slice. Draw a uniform number, x_1, on this slice, between (L_1, R_1).
4. Repeat steps 2 and 3 many times.

The challenging part of the algorithm is finding the horizontal slice (L_i, R_i) at each iteration. The closed form expressions of $p_L^{-1}(y_i)$ and $p_R^{-1}(y_i)$ are virtually impossible to obtain analytically in most problems. Neal (2003) proved that although exact solutions would be nice, devising a search algorithm that finds portions of this horizontal slice is sufficient for the sampler to work. The search algorithm is based on the rejection method to expand and contract, when needed.

The sampler is implemented as an optional algorithm in the MCMC procedure, where you can use it to draw either model parameters or random-effects parameters. As with the ARMS algorithm, only the univariate version of the slice sampler is implemented. The slice sampler requires repeated evaluations of the objective function; this happens in the search algorithm to identify each horizontal slice at every iteration. Hence, the computational cost could be high if each evaluation of the objective function requires one pass through the entire data set.

[1] The extension to the multivariate ARMS algorithm is possible in theory but problematic in practice because the computational cost associated with constructing a multidimensional hyperbola envelop is often prohibitive.

Independence Sampler

Another type of Metropolis algorithm is the "independence" sampler. It is called the independence sampler because the proposal distribution in the algorithm does not depend on the current point as it does with the random-walk Metropolis algorithm. For this sampler to work well, you want to have a proposal distribution that mimics the target distribution and have the acceptance rate be as high as possible.

1. Set $t = 0$. Choose a starting point θ^0. This can be an arbitrary point as long as $f(\theta^0|\mathbf{y}) > 0$.
2. Generate a new sample, θ_{new}, by using the proposal distribution $q(\cdot)$. The proposal distribution does not depend on the current value of θ^t.
3. Calculate the following quantity:

$$r = \min\left\{\frac{f(\theta_{\text{new}}|\mathbf{y})/q(\theta_{\text{new}})}{f(\theta^t|\mathbf{y})/q(\theta^t)}, 1\right\}$$

4. Sample u from the uniform distribution $U(0, 1)$.
5. Set $\theta^{t+1} = \theta_{\text{new}}$ if $u < r$; otherwise set $\theta^{t+1} = \theta^t$.
6. Set $t = t + 1$. If $t < T$, the number of desired samples, return to step 2. Otherwise, stop.

A good proposal density should have thicker tails than those of the target distribution. This requirement sometimes can be difficult to satisfy especially in cases where you do not know what the target posterior distributions are like. In addition, this sampler does not produce independent samples as the name seems to imply, and sample chains from independence samplers can get stuck in the tails of the posterior distribution if the proposal distribution is not chosen carefully. The MCMC procedure uses the independence sampler.

Gamerman Algorithm

The Gamerman algorithm, named after the inventor Dani Gamerman, is a special case of the Metropolis algorithm (see the section "Metropolis and Metropolis-Hastings Algorithms" on page 156) in which the proposal distribution is derived from one iteration of the iterative weighted least squares (IWLS) algorithm. As the name suggests, a weighted least squares algorithm is carried out inside an iteration loop. For each iteration, a set of weights for the observations is used in the least squares fit. The weights are constructed by applying a weight function to the current residuals. The proposal distribution uses the current iteration's values of the parameters to form the proposal distribution from which to generate a proposed random value (Gamerman 1997).

The multivariate sampling algorithm is simple but practical, and can be used to obtain random samples from the posterior distribution of the regression parameters in a generalized linear model (GLM). See "Generalized Linear Regression" on page 84 in Chapter 4, "Introduction to Regression Procedures," for further details on generalized linear regression models. See McCullagh and Nelder (1989) for a discussion of transformed observations and diagonal matrix of weights pertaining to IWLS.

The BCHOICE, BGLIMM, and GENMOD procedures use the Gamerman algorithm to sample parameters from their multivariate posterior conditional distributions. Each procedure has a detailed description of the algorithm in its own documentation.

Hamiltonian Monte Carlo Sampler

The Hamiltonian Monte Carlo (HMC) algorithm, also known as the hybrid Monte Carlo algorithm, is a version of the Metropolis algorithm that uses gradients information and auxiliary momentum variables to draw samples from the posterior distribution (Neal 2011). The algorithm uses Hamiltonian dynamics to enable distant proposals in the Metropolis algorithm, making it efficient in many scenarios. The HMC algorithm is applicable only to continuous parameters.

HMC translates the target density function to a potential energy function and adds an auxiliary momentum variable $\mathbf{r}$ for each model parameter $\boldsymbol{\theta}$. The resulting joint density has the form

$$p(\boldsymbol{\theta}, \mathbf{r}) \propto p(\boldsymbol{\theta}) \exp\left(-\frac{1}{2}\mathbf{r}'\mathbf{r}\right)$$

where $p(\boldsymbol{\theta})$ is the posterior of the parameters $\boldsymbol{\theta}$ (up to a normalizing constant). HMC draws from the joint space of $(\boldsymbol{\theta}, \mathbf{r})$, throws away $\mathbf{r}$, and retains $\boldsymbol{\theta}$ as samples from $p(\boldsymbol{\theta})$. The algorithm uses the idea of Hamiltonian dynamics in preserving the total energy of a physics system, in which $\boldsymbol{\theta}$ is part of the potential energy function and $\mathbf{r}$ is part of the kinetic energy (velocity). As the velocity changes, the potential energy changes accordingly, leading to the movements in the parameter space.

At each iteration, HMC first generates the momentum variables $\mathbf{r}$, usually from standard normal distributions, that are independent of $\boldsymbol{\theta}$. Then the algorithm follows with a Metropolis update that includes many steps along a trajectory while maintaining the total energy of the system. One of the most frequently used methods in moving along this trajectory is the leapfrog method, which involves L steps with a step size ϵ,

$$\begin{aligned}
\mathbf{r}^{t+\epsilon/2} &= \mathbf{r}^{t} + (\epsilon/2) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}^{t}) \\
\boldsymbol{\theta}^{t+\epsilon} &= \boldsymbol{\theta}^{t} + \epsilon \mathbf{r}^{t+\epsilon/2} \\
\mathbf{r}^{t+\epsilon} &= \mathbf{r}^{t+\epsilon/2} + (\epsilon/2) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}^{t+\epsilon})
\end{aligned}$$

where $\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta})$ is the gradient of the log-posterior with respect to $\boldsymbol{\theta}$. After L steps, the proposed state $(\boldsymbol{\theta}^*, \mathbf{r}^*)$ is accepted as the next state of the Markov chain with probability $\min\{1, p(\boldsymbol{\theta}^*, \mathbf{r}^*)/p(\boldsymbol{\theta}, \mathbf{r})\}$.

Although HMC can lead to rapid convergence, it also heavily relies on two requirements: the gradient calculation of the logarithm of the posterior density and carefully selected tuning parameters, in step size ϵ and number of steps L. Step sizes that are too large or too small can lead to overly low or overly high acceptance rates, both of which affect the convergence of the Markov chain. Large L leads to large trajectory length ($\epsilon \cdot L$), which can move the parameters back to their original positions. Small L limits the movement of the chain. To tune ϵ and L, you usually want to run a few trials, starting with a small number of L and an ϵ with a value around 0.1. See Neal (2011) for advice on tuning these parameters.

An example of adaptive HMC with automatic tuning of ϵ and L is the No-U-Turn Sampler (NUTS; Hoffman and Gelman 2014). The NUTS algorithm uses a doubling process to build a binary tree whose leaf nodes correspond to the states of the parameters and momentum variables. The initial tree has a single node with no heights ($j = 0$). The doubling process expands the tree either left or right in a binary fashion, and in each direction, the algorithm takes 2^j leapfrog steps of size ϵ. Obviously, as the height of the tree (j) increases, the computational cost increases dramatically. The tree expands until one sampling trajectory makes a U-turn and starts to revisit parameter space that has been already explored. The NUTS algorithm tunes ϵ so that the actual acceptance rate during the doubling process is close to a predetermined target acceptance probability δ (usually set to 0.6 or higher). When the tuning stage ends, the NUTS algorithm proceeds to the main sampling stage and starts to draw posterior samples that have a fixed ϵ value. Increasing the targeted acceptance probability δ can often improve mixing, but it can also slow down the process significantly. For more information about the NUTS algorithm and its efficiency, see Hoffman and Gelman (2014).

Burn-In, Thinning, and Markov Chain Samples

Burn-in refers to the practice of discarding an initial portion of a Markov chain sample so that the effect of initial values on the posterior inference is minimized. For example, suppose the target distribution is $N(0, 1)$ and the Markov chain was started at the value 10^6. The chain might quickly travel to regions around 0 in a few iterations. However, including samples around the value 10^6 in the posterior mean calculation can produce substantial bias in the mean estimate. In theory, if the Markov chain is run for an infinite amount of time, the effect of the initial values decreases to zero. In practice, you do not have the luxury of infinite samples. In practice, you assume that after t iterations, the chain has reached its target distribution and you can throw away the early portion and use the good samples for posterior inference. The value of t is the burn-in number.

With some models you might experience poor mixing (or slow convergence) of the Markov chain. This can happen, for example, when parameters are highly correlated with each other. Poor mixing means that the Markov chain slowly traverses the parameter space (see the section "Visual Analysis via Trace Plots" on page 163 for examples of poorly mixed chains) and the chain has high dependence. High sample autocorrelation can result in biased Monte Carlo standard errors. A common strategy is to *thin* the Markov chain in order to reduce sample autocorrelations. You thin a chain by keeping every kth simulated draw from each sequence. You can safely use a thinned Markov chain for posterior inference as long as the chain converges. It is important to note that thinning a Markov chain can be wasteful because you are throwing away a $\frac{k-1}{k}$ fraction of all the posterior samples generated. MacEachern and Berliner (1994) show that you always get more precise posterior estimates if the entire Markov chain is used. However, other factors, such as computer storage or plotting time, might prevent you from keeping all samples.

To use the BCHOICE, BGLIMM, FMM, GENMOD, LIFEREG, MCMC, and PHREG procedures, you need to determine the total number of samples to keep ahead of time. This number is not obvious and often depends on the type of inference you want to make. Mean estimates do not require nearly as many samples as small-tail percentile estimates. In most applications, you might find that keeping a few thousand iterations is sufficient for reasonably accurate posterior inference. In all four procedures, the relationship between the number of iterations requested, the number of iterations kept, and the amount of thinning is as follows:

$$\text{kept} = \left[\frac{\text{requested}}{\text{thinning}}\right]$$

where [ ] is the rounding operator.

Assessing Markov Chain Convergence

Simulation-based Bayesian inference requires using simulated draws to summarize the posterior distribution or calculate any relevant quantities of interest. You need to treat the simulation draws with care. There are usually two issues. First, you have to decide whether the Markov chain has reached its stationary, or the desired posterior, distribution. Second, you have to determine the number of iterations to keep after the Markov chain has reached stationarity. Convergence diagnostics help to resolve these issues. Note that many diagnostic tools are designed to verify a necessary but not sufficient condition for convergence. There are no conclusive tests that can tell you when the Markov chain has converged to its stationary distribution. You should proceed with caution. Also, note that you should check the convergence of *all* parameters, and not just those of interest, before proceeding to make any inference. With some models, certain parameters can

appear to have very good convergence behavior, but that could be misleading due to the slow convergence of other parameters. If some of the parameters have bad mixing, you cannot get accurate posterior inference for parameters that appear to have good mixing. See Cowles and Carlin (1996) and Brooks and Roberts (1998) for discussions about convergence diagnostics.

Visual Analysis via Trace Plots

Trace plots of samples versus the simulation index can be very useful in assessing convergence. The trace tells you if the chain has not yet converged to its stationary distribution—that is, if it needs a longer burn-in period. A trace can also tell you whether the chain is mixing well. A chain might have reached stationarity if the distribution of points is not changing as the chain progresses. The aspects of stationarity that are most recognizable from a trace plot are a relatively constant mean and variance. A chain that mixes well traverses its posterior space rapidly, and it can jump from one remote region of the posterior to another in relatively few steps. Figure 8.1 through Figure 8.4 display some typical features that you might see in trace plots. The traces are for a parameter called γ.

Figure 8.1 Essentially Perfect Trace for γ

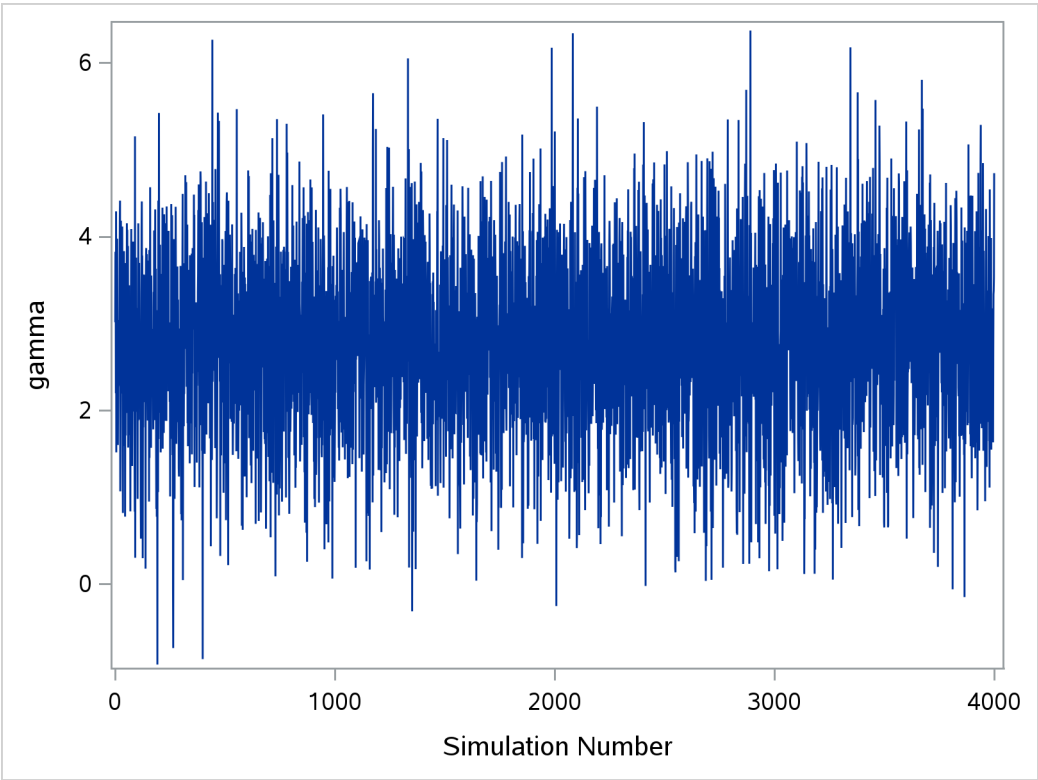

Figure 8.1 displays a "perfect" trace plot. Note that the center of the chain appears to be around the value 3, with very small fluctuations. This indicates that the chain could have reached the right distribution. The chain is mixing well; it is exploring the distribution by traversing to areas where its density is very low. You can conclude that the mixing is quite good here.

Figure 8.2 Initial Samples of γ Need to be Discarded

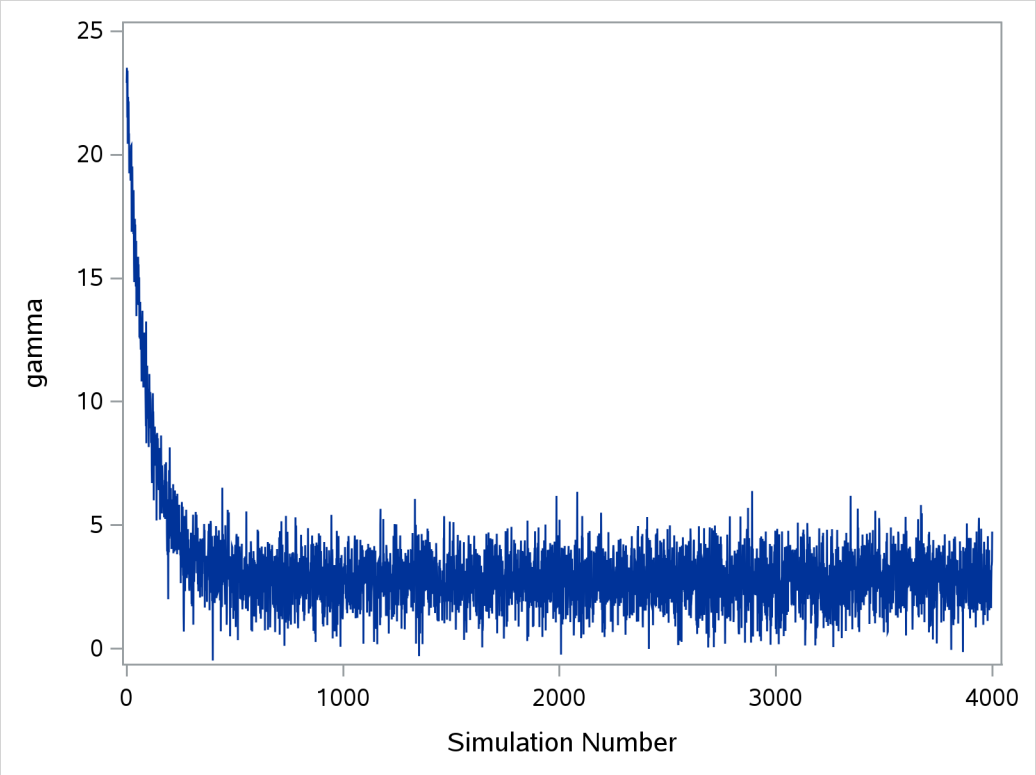

Figure 8.2 displays a trace plot for a chain that starts at a very remote initial value and makes its way to the targeting distribution. The first few hundred observations should be discarded. This chain appears to be mixing very well locally. It travels relatively quickly to the target distribution, reaching it in a few hundred iterations. If you have a chain that looks like this, you would want to increase the burn-in sample size. If you need to use this sample to make inferences, you would want to use only the samples toward the end of the chain.

Figure 8.3 Marginal Mixing for γ

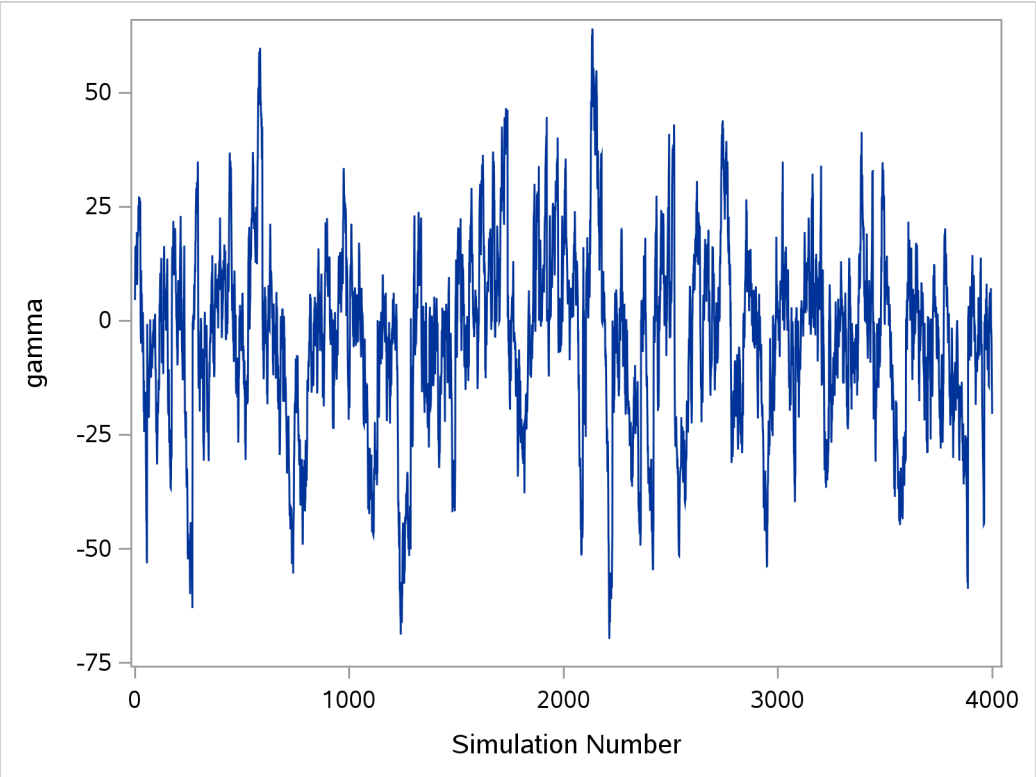

Figure 8.3 demonstrates marginal mixing. The chain is taking only small steps and does not traverse its distribution quickly. This type of trace plot is typically associated with high autocorrelation among the samples. To obtain a few thousand independent samples, you need to run the chain for much longer.

Figure 8.4 Bad Mixing, Nonconvergence of γ

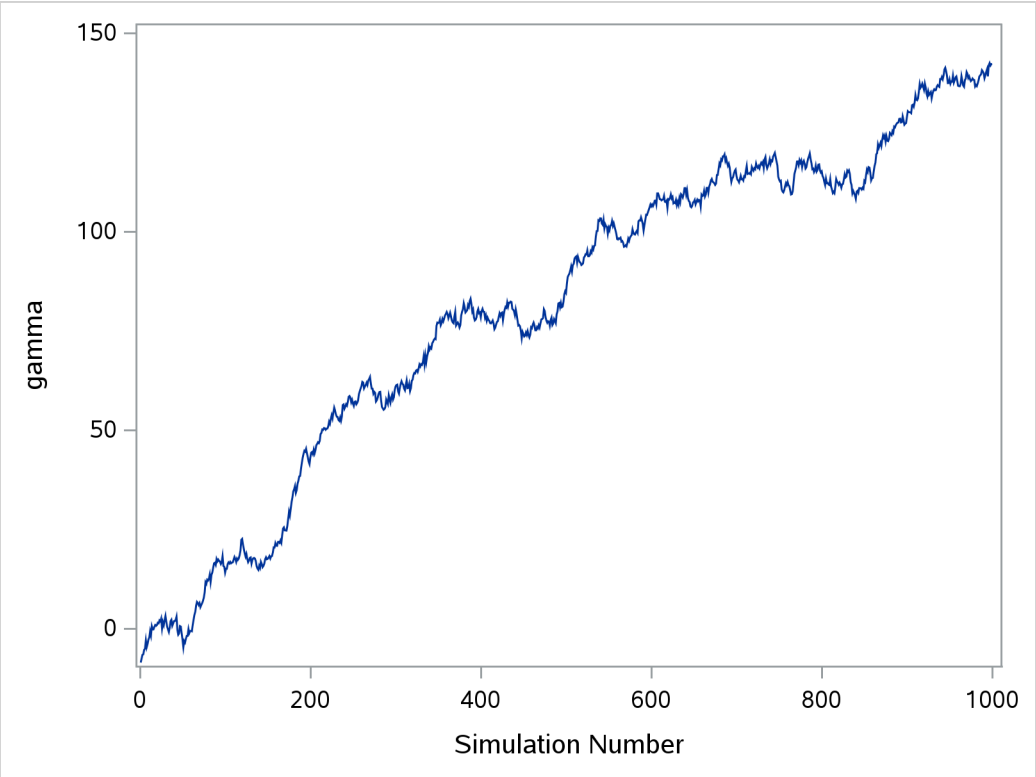

The trace plot shown in Figure 8.4 depicts a chain with serious problems. It is mixing very slowly, and it offers no evidence of convergence. You would want to try to improve the mixing of this chain. For example, you might consider reparameterizing your model on the log scale. Run the Markov chain for a long time to see where it goes. This type of chain is entirely unsuitable for making parameter inferences.

Statistical Diagnostic Tests

The Bayesian procedures include several statistical diagnostic tests that can help you assess Markov chain convergence. For a detailed description of each of the diagnostic tests, see the following subsections. Table 8.1 provides a summary of the diagnostic tests and their interpretations.

Table 8.1 Convergence Diagnostic Tests Available in the Bayesian Procedures

Name	Description	Interpretation of the Test
Gelman-Rubin	Uses parallel chains with dispersed initial values to test whether they all converge to the same target distribution. Failure could indicate the presence of a multi-mode posterior distribution (different chains converge to different local modes) or the need to run a longer chain (burn-in is yet to be completed).	One-sided test based on a variance ratio test statistic. Large $\widehat{R}_c$ values indicate rejection.

Table 8.1 *continued*

Name	Description	Interpretation of the test
Geweke	Tests whether the mean estimates have converged by comparing means from the early and latter part of the Markov chain.	Two-sided test based on a z-score statistic. Large absolute z values indicate rejection.
Heidelberger-Welch (stationarity test)	Tests whether the Markov chain is a covariance (or weakly) stationary process. Failure could indicate that a longer Markov chain is needed.	One-sided test based on a Cramér–von Mises statistic. Small p-values indicate rejection.
Heidelberger-Welch (half-width test)	Reports whether the sample size is adequate to meet the required accuracy for the mean estimate. Failure could indicate that a longer Markov chain is needed.	If a relative half-width statistic is greater than a predetermined accuracy measure, this indicates rejection.
Raftery-Lewis	Evaluates the accuracy of the estimated (desired) percentiles by reporting the number of samples needed to reach the desired accuracy of the percentiles. Failure could indicate that a longer Markov chain is needed.	If the total samples needed are fewer than the Markov chain sample, this indicates rejection.
Autocorrelation	Measures dependency among Markov chain samples.	High correlations between long lags indicate poor mixing.
Effective sample size	Relates to autocorrelation; measures mixing of the Markov chain.	Large discrepancy between the effective sample size and the simulation sample size indicates poor mixing.

Gelman and Rubin Diagnostics

Gelman and Rubin diagnostics (Gelman and Rubin 1992; Brooks and Gelman 1997) are based on analyzing multiple simulated MCMC chains by comparing the variances within each chain and the variance between chains. Large deviation between these two variances indicates nonconvergence.

Define $\{\theta^t\}$, where $t = 1, \ldots, n$, to be the collection of a single Markov chain output. The parameter θ^t is the tth sample of the Markov chain. For notational simplicity, θ is assumed to be single dimensional in this section.

Suppose you have M parallel MCMC chains that were initialized from various parts of the target distribution. Each chain is of length n (after discarding the burn-in). For each θ^t, the simulations are labeled as θ_m^t, where $t = 1, \ldots, n$ and $m = 1, \ldots, M$. The between-chain variance B and the within-chain variance W are calculated as

$$
\begin{aligned}
B &= \frac{n}{M-1}\sum_{m=1}^{M}(\bar{\theta}_m^{\cdot}-\bar{\theta}_{\cdot}^{\cdot})^2, \quad \text{where } \bar{\theta}_m^{\cdot}=\frac{1}{n}\sum_{t=1}^{n}\theta_m^t, \quad \bar{\theta}_{\cdot}^{\cdot}=\frac{1}{M}\sum_{m=1}^{M}\bar{\theta}_m^{\cdot} \\
W &= \frac{1}{M}\sum_{m=1}^{M}s_m^2, \quad \text{where } s_m^2=\frac{1}{n-1}\sum_{t=1}^{n}(\theta_m^t-\bar{\theta}_m^{\cdot})^2
\end{aligned}
$$

The posterior marginal variance, $\mathrm{var}(\theta|\mathbf{y})$, is a weighted average of W and B. The estimate of the variance is

$$
\widehat{V}=\frac{n-1}{n}W+\frac{M+1}{nM}B
$$

If all M chains have reached the target distribution, this posterior variance estimate should be very close to the within-chain variance W. Therefore, you would expect to see the ratio $\widehat{V}/W$ be close to 1. The square root of this ratio is referred to as the *potential scale reduction factor* (PSRF). A large PSRF indicates that the between-chain variance is substantially greater than the within-chain variance, so that longer simulation is needed. If the PSRF is close to 1, you can conclude that each of the M chains has stabilized, and they are likely to have reached the target distribution.

A refined version of PSRF is calculated, as suggested by Brooks and Gelman (1997), as

$$
\widehat{\mathrm{R}}_c=\sqrt{\frac{\hat{d}+3}{\hat{d}+1}\cdot\frac{\widehat{V}}{W}}=\sqrt{\frac{\hat{d}+3}{\hat{d}+1}\left(\frac{n-1}{n}+\frac{M+1}{nM}\frac{B}{W}\right)}
$$

where

$$
\hat{d}=\frac{2\widehat{V}^2}{\widehat{\mathrm{Var}}(\widehat{V})}
$$

and

$$
\begin{aligned}
\widehat{\mathrm{Var}}(\widehat{V}) &= \left(\frac{n-1}{n}\right)^2\frac{1}{M}\widehat{\mathrm{Var}}(s_m^2)+\left(\frac{M+1}{nM}\right)^2\frac{2}{M-1}B^2 \\
&\quad +2\frac{(M+1)(n-1)}{n^2M}\frac{n}{M}\left(\widehat{\mathrm{cov}}(s_m^2,(\bar{\theta}_m^{\cdot})^2)-2\bar{\theta}_{\cdot}^{\cdot}\widehat{\mathrm{cov}}(s_m^2,\bar{\theta}_m^{\cdot})\right)
\end{aligned}
$$

All the Bayesian procedures also produce an upper $100(1-\alpha/2)\%$ confidence limit of $\widehat{\mathrm{R}}_c$. Gelman and Rubin (1992) showed that the ratio B/W in $\widehat{\mathrm{R}}_c$ has an F distribution with degrees of freedom $M-1$ and $2W^2M/\widehat{\mathrm{Var}}(s_m^2)$. Because you are concerned only if the scale is large, not small, only the upper $100(1-\alpha/2)\%$ confidence limit is reported. This is written as

$$\sqrt{\left(\frac{n-1}{n}+\frac{M+1}{nM}\cdot F_{1-\alpha/2}\left(M-1,\frac{2W^2}{\widehat{\mathrm{Var}}(s_m^2)/M}\right)\right)\cdot\frac{\hat{d}+3}{\hat{d}+1}}$$

In the Bayesian procedures, you can specify the number of chains that you want to run. Typically three chains are sufficient. The first chain is used for posterior inference, such as mean and standard deviation; the other $M-1$ chains are used for computing the diagnostics and are discarded afterward. This test can be computationally costly, because it prolongs the simulation M-fold.

It is best to choose different initial values for all M chains. The initial values should be as dispersed from each other as possible so that the Markov chains can fully explore different parts of the distribution before they converge to the target. Similar initial values can be risky because all of the chains can get stuck in a local maximum; that is something this convergence test cannot detect. If you do not supply initial values for all the different chains, the procedures generate them for you.

Geweke Diagnostics

The Geweke test (Geweke 1992) compares values in the early part of the Markov chain to those in the latter part of the chain in order to detect failure of convergence. The statistic is constructed as follows. Two subsequences of the Markov chain $\{\theta^t\}$ are taken out, with $\{\theta_1^t : t = 1,\ldots,n_1\}$ and $\{\theta_2^t : t = n_a,\ldots,n\}$, where $1 < n_1 < n_a < n$. Let $n_2 = n - n_a + 1$, and define

$$\bar{\theta}_1 = \frac{1}{n_1}\sum_{t=1}^{n_1}\theta^t \quad\text{and}\quad \bar{\theta}_2 = \frac{1}{n_2}\sum_{t=n_a}^{n}\theta^t$$

Let $\hat{s}_1(0)$ and $\hat{s}_2(0)$ denote consistent spectral density estimates at zero frequency (see the subsection "Spectral Density Estimate at Zero Frequency" on page 169 for estimation details) for the two MCMC chains, respectively. If the ratios n_1/n and n_2/n are fixed, $(n_1+n_2)/n < 1$, and the chain is stationary, then the following statistic converges to a standard normal distribution as $n \to \infty$:

$$Z_n = \frac{\bar{\theta}_1 - \bar{\theta}_2}{\sqrt{\frac{\hat{s}_1(0)}{n_1}+\frac{\hat{s}_2(0)}{n_2}}}$$

This is a two-sided test, and large absolute z-scores indicate rejection.

Spectral Density Estimate at Zero Frequency

For one sequence of the Markov chain $\{\theta_t\}$, the relationship between the h-lag covariance sequence of a time series and the spectral density, f, is

$$s_h = \frac{1}{2\pi}\int_{-\pi}^{\pi}\exp(\mathrm{i}\omega h)\,f(\omega)\,d\omega$$

where i indicates that ωh is the complex argument. Inverting this Fourier integral,

$$f(\omega) = \sum_{h=-\infty}^{\infty} s_h \exp(-\mathrm{i}\omega h) = s_0 \left(1 + 2\sum_{h=1}^{\infty} \rho_h \cos(\omega h)\right)$$

It follows that

$$f(0) = \sigma^2 \left(1 + 2\sum_{h=1}^{\infty} \rho_h\right)$$

which gives an autocorrelation adjusted estimate of the variance. In this equation, σ^2 is the naive variance estimate of the sequence $\{\theta_t\}$ and ρ_h is the lag h autocorrelation. Due to obvious computational difficulties, such as calculation of autocorrelation at infinity, you cannot effectively estimate $f(0)$ by using the preceding formula. The usual route is to first obtain the *periodogram* $p(\omega)$ of the sequence, and then estimate $f(0)$ by smoothing the estimated periodogram. The periodogram is defined to be

$$p(\omega) = \frac{1}{n}\left[\left(\sum_{t=1}^{n} \theta_t \sin(\omega t)\right)^2 + \left(\sum_{t=1}^{n} \theta_t \cos(\omega t)\right)^2\right]$$

The procedures use the following way to estimate $\hat{f}(0)$ from p (Heidelberger and Welch 1981). In $p(\omega)$, let $\omega = \omega_k = 2\pi k/n$ and $k = 1, \ldots, [\frac{n}{2}]$.[2] A smooth spectral density in the domain of $(0, \pi]$ is obtained by fitting a gamma model with the log link function, using $p(\omega_k)$ as response and $x_1(\omega_k) = \sqrt{3}(4\omega_k/(2\pi) - 1)$ as the only regressor. The predicted value $\hat{f}(0)$ is given by

$$\hat{f}(0) = \exp(\hat{\beta}_0 - \sqrt{3}\hat{\beta}_1)$$

where $\hat{\beta}_0$ and $\hat{\beta}_1$ are the estimates of the intercept and slope parameters, respectively.

Heidelberger and Welch Diagnostics

The Heidelberger and Welch test (Heidelberger and Welch 1981, 1983) consists of two parts: a stationary portion test and a half-width test. The stationarity test assesses the stationarity of a Markov chain by testing the hypothesis that the chain comes from a covariance stationary process. The half-width test checks whether the Markov chain sample size is adequate to estimate the mean values accurately.

Given $\{\theta^t\}$, set $S_0 = 0$, $S_n = \sum_{t=1}^{n} \theta^t$, and $\bar{\theta} = (1/n)\sum_{t=1}^{n} \theta^t$. You can construct the following sequence with s coordinates on values from $\frac{1}{n}, \frac{2}{n}, \ldots, 1$:

$$B_n(s) = (S_{[ns]} - [ns]\bar{\theta})/(n\hat{p}(0))^{1/2}$$

[2]This is equivalent to the fast Fourier transformation of the original time series θ_t.

where [ ] is the rounding operator, and $\hat{p}(0)$ is an estimate of the spectral density at zero frequency that uses the second half of the sequence (see the section "Spectral Density Estimate at Zero Frequency" on page 169 for estimation details). For large n, B_n converges in distribution to a Brownian bridge (Billingsley 1986). So you can construct a test statistic by using B_n. The statistic used in these procedures is the Cramer–von Mises statistic[3]; that is $\int_0^1 B_n(s)^2 ds = \mathrm{CVM}(B_n)$. As $n \to \infty$, the statistic converges in distribution to a standard Cramer–von Mises distribution. The integral $\int_0^1 B_n(s)^2 ds$ is numerically approximated using Simpson's rule.

Let $y_i = B_n(s)^2$, where $s = 0, \frac{1}{n}, \ldots, \frac{n-1}{n}, 1$, and $i = ns = 0, 1, \ldots, n$. If n is even, let $m = n/2$; otherwise, let $m = (n-1)/2$. The Simpson's approximation to the integral is

$$\int_0^1 B_n(s)^2 ds \approx \frac{1}{3n}\left[y_0 + 4(y_1 + \cdots + y_{2m-1}) + 2(y_2 + \cdots + y_{2m-2}) + y_{2m}\right]$$

Note that Simpson's rule requires an even number of intervals. When n is odd, y_n is set to be 0 and the value does not contribute to the approximation.

This test can be performed repeatedly on the same chain, and it helps you identify a time t when the chain has reached stationarity. The whole chain, $\{\theta^t\}$, is first used to construct the Cramer–von Mises statistic. If it passes the test, you can conclude that the entire chain is stationary. If it fails the test, you drop the initial 10% of the chain and redo the test by using the remaining 90%. This process is repeated until either a time t is selected or it reaches a point where there are not enough data remaining to construct a confidence interval (the cutoff proportion is set to be 50%).

The part of the chain that is deemed stationary is put through a half-width test, which reports whether the sample size is adequate to meet certain accuracy requirements for the mean estimates. Running the simulation less than this length of time would not meet the requirement, while running it longer would not provide any additional information that is needed. The statistic calculated here is the *relative half-width* (RHW) of the confidence interval. The RHW for a confidence interval of level $1-\alpha$ is

$$\mathrm{RHW} = \frac{z_{(1-\alpha/2)} \cdot (\hat{s}_n/n)^{1/2}}{\hat{\theta}}$$

where $z_{(1-\alpha/2)}$ is the z-score of the $100(1-\alpha/2)$th percentile (for example, $z_{(1-\alpha/2)} = 1.96$ if $\alpha = 0.05$), $\hat{s}_n$ is the variance of the chain estimated using the spectral density method (see explanation in the section "Spectral Density Estimate at Zero Frequency" on page 169), n is the length, and $\hat{\theta}$ is the estimated mean. The RHW quantifies accuracy of the $1-\alpha$ level confidence interval of the mean estimate by measuring the ratio between the sample standard error of the mean and the mean itself. In other words, you can stop the Markov chain if the variability of the mean stabilizes with respect to the mean. An implicit assumption is that large means are often accompanied by large variances. If this assumption is not met, then this test can produce false rejections (such as a small mean around 0 and large standard deviation) or false acceptance (such as a very large mean with relative small variance). As with any other convergence diagnostics, you might want to exercise caution in interpreting the results.

[3] The von Mises distribution was first introduced by Von Mises (1918). The density function is $p(\theta|\mu\kappa) \sim M(\mu,\kappa) = [2\pi I_0(\kappa)]^{-1}\exp(\kappa\cos(\theta-\mu))$ $(0 \le \theta \le 2\pi)$, where the function $I_0(\kappa)$ is the modified Bessel function of the first kind and order zero, defined by $I_0(\kappa) = (2\pi)^{-1}\int_0^{2\pi}\exp(\kappa\cos(\theta-\mu))d\theta$.

The stationarity test is one-sided; rejection occurs when the p-value is greater than $1-\alpha$. To perform the half-width test, you need to select an α level (the default of which is 0.05) and a predetermined tolerance value ϵ (the default of which is 0.1). If the calculated RHW is greater than ϵ, you conclude that there are not enough data to accurately estimate the mean with $1-\alpha$ confidence under tolerance of ϵ.

Raftery and Lewis Diagnostics

If your interest lies in posterior percentiles, you want a diagnostic test that evaluates the accuracy of the estimated percentiles. The Raftery-Lewis test (Raftery and Lewis 1992, 1995) is designed for this purpose. Notation and deductions here closely resemble those in Raftery and Lewis (1995).

Suppose you are interested in a quantity θ_q such that $P(\theta \le \theta_q|\mathbf{y}) = q$, where q can be an arbitrary cumulative probability, such as 0.025. This θ_q can be empirically estimated by finding the $100nq$th number of the sorted $\{\theta^t\}$. Let $\hat{\theta}_q$ denote the estimand, which corresponds to an estimated probability $P(\theta \le \hat{\theta}_q) = \hat{P}_q$. Because the simulated posterior distribution converges to the true distribution as the simulation sample size grows, $\hat{\theta}_q$ can achieve any degree of accuracy if the simulator is run for a very long time. However, running too long a simulation can be wasteful. Alternatively, you can use coverage probability to measure accuracy and stop the chain when a certain accuracy is reached.

A stopping criterion is reached when the estimated probability is within $\pm r$ of the true cumulative probability q, with probability s, such as $P(\hat{P}_q \in (q-r, q+r)) = s$. For example, suppose you want the coverage probability s to be 0.95 and the amount of tolerance r to be 0.005. This corresponds to requiring that the estimate of the cumulative distribution function of the 2.5th percentile be estimated to within ± 0.5 percentage points with probability 0.95.

The Raftery-Lewis diagnostics test finds the number of iterations, M, that need to be discarded (burn-ins) and the number of iterations needed, N, to achieve a desired precision. Given a predefined cumulative probability q, these procedures first find $\hat{\theta}_q$, and then they construct a binary $0-1$ process $\{Z_t\}$ by setting $Z_t = 1$ if $\theta^t \le \hat{\theta}_q$ and 0 otherwise for all t. The sequence $\{Z_t\}$ is itself not a Markov chain, but you can construct a subsequence of $\{Z_t\}$ that is approximately Markovian if it is sufficiently k-thinned. When k becomes reasonably large, $\{Z_t^{(k)}\}$ starts to behave like a Markov chain.

Next, the procedures find this thinning parameter k. The number k is estimated by comparing the Bayesian information criterion (BIC) between two Markov models: a first-order and a second-order Markov model. A jth-order Markov model is one in which the current value of $\{Z_t^{(k)}\}$ depends on the previous j values. For example, in a second-order Markov model,

$$
\begin{aligned}
& p\left(Z_t^{(k)} = z_t | Z_{t-1}^{(k)} = z_{t-1}, Z_{t-2}^{(k)} = z_{t-2}, \ldots, Z_0^{(k)} = z_0\right) \\
= \quad & p\left(Z_t^{(k)} = z_t | Z_{t-1}^{(k)} = z_{t-1}, Z_{t-2}^{(k)} = z_{t-2}\right)
\end{aligned}
$$

where $z_i = \{0,1\}, i = 0, \ldots, t$. Given $\{Z_t^{(k)}\}$, you can construct two transition count matrices for a second-order Markov model:

	$z_t = 0$		$z_t = 1$		
	$z_{t-1} = 0$	$z_{t-1} = 1$		$z_{t-1} = 0$	$z_{t-1} = 1$
$z_{t-2} = 0$	w_{000}	w_{010}	$z_{t-2} = 0$	w_{001}	w_{011}
$z_{t-2} = 1$	w_{100}	w_{110}	$z_{t-2} = 1$	w_{101}	w_{111}

For each k, the procedures calculate the BIC that compares the two Markov models. The BIC is based on a likelihood ratio test statistic that is defined as

$$G_k^2 = 2\sum_{i=0}^{1}\sum_{j=0}^{1}\sum_{l=0}^{1} w_{ijl}\log\frac{w_{ijl}}{\hat{w}_{ijl}}$$

where $\hat{w}_{ijl}$ is the expected cell count of w_{ijl} under the null model, the first-order Markov model, where the assumption $(Z_t^{(k)} \perp Z_{t-2}^{(k)})|Z_{t-1}^{(k)}$ holds. The formula for the expected cell count is

$$\hat{w}_{ijl} = \frac{\sum_i w_{ijl}\cdot\sum_l w_{ijl}}{\sum_i\sum_l w_{ijl}}$$

The BIC is $G_k^2 - 2\log(n_k - 2)$, where n_k is the k-thinned sample size (every kth sample starting with the first), with the last two data points discarded due to the construction of the second-order Markov model. The thinning parameter k is the smallest k for which the BIC is negative. When k is found, you can estimate a transition probability matrix between state 0 and state 1 for $\{Z_t^{(k)}\}$:

$$Q = \begin{pmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{pmatrix}$$

Because $\{Z_t^{(k)}\}$ is a Markov chain, its equilibrium distribution exists and is estimated by

$$\pi = (\pi_0, \pi_1) = \frac{(\beta, \alpha)}{\alpha+\beta}$$

where $\pi_0 = P(\theta \le \theta_q|\mathbf{y})$ and $\pi_1 = 1-\pi_0$. The goal is to find an iteration number m such that after m steps, the estimated transition probability $P(Z_m^{(k)} = i|Z_0^{(k)} = j)$ is within ϵ of equilibrium π_i for $i, j = 0, 1$. Let $e_0 = (1, 0)$ and $e_1 = 1 - e_0$. The estimated transition probability after step m is

$$P(Z_m^{(k)} = i|Z_0^{(k)} = j) = e_j\left[\begin{pmatrix} \pi_0 & \pi_1 \\ \pi_0 & \pi_1 \end{pmatrix} + \frac{(1-\alpha-\beta)^m}{\alpha+\beta}\begin{pmatrix} \alpha & -\alpha \\ -\beta & \beta \end{pmatrix}\right]e_j'$$

which holds when

$$m = \frac{\log\left(\frac{(\alpha+\beta)\epsilon}{\max(\alpha,\beta)}\right)}{\log(1-\alpha-\beta)}$$

assuming $1-\alpha-\beta > 0$.

Therefore, by time m, $\{Z_t^{(k)}\}$ is sufficiently close to its equilibrium distribution, and you know that a total size of $M = mk$ should be discarded as the burn-in.

Next, the procedures estimate N, the number of simulations needed to achieve desired accuracy on percentile estimation. The estimate of $P(\theta \le \theta_q|\mathbf{y})$ is $\bar{Z}_n^{(k)} = \frac{1}{n}\sum_{t=1}^{n} Z_t^{(k)}$. For large n, $\bar{Z}_n^{(k)}$ is normally distributed with mean q, the true cumulative probability, and variance

$$\frac{1}{n}\frac{(2-\alpha-\beta)\alpha\beta}{(\alpha+\beta)^3}$$

$P(q - r \le \bar{Z}_n^{(k)} \le q + r) = s$ is satisfied if

$$n = \frac{(2-\alpha-\beta)\alpha\beta}{(\alpha+\beta)^3}\left\{\frac{\Phi^{-1}\left(\frac{s+1}{2}\right)}{r}\right\}^2$$

Therefore, $N = nk$.

By using similar reasoning, the procedures first calculate the minimal number of iterations needed to achieve the desired accuracy, assuming the samples are independent:

$$N_{min} = \left\{\Phi^{-1}\left(\frac{s+1}{2}\right)\right\}^2 \frac{q(1-q)}{r^2}$$

If $\{\theta^t\}$ does not have that required sample size, the Raftery-Lewis test is not carried out. If you still want to carry out the test, increase the number of Markov chain iterations.

The ratio N/N_{min} is sometimes referred to as the *dependence factor*. It measures deviation from posterior sample independence: the closer it is to 1, the less correlated are the samples. There are a few things to keep in mind when you use this test. This diagnostic tool is specifically designed for the percentile of interest and does not provide information about convergence of the chain as a whole (Brooks and Roberts 1999). In addition, the test can be very sensitive to small changes. Both N and N_{min} are inversely proportional to r^2, so you can expect to see large variations in these numbers with small changes to input variables, such as the desired coverage probability or the cumulative probability of interest. Last, the time until convergence for a parameter can differ substantially for different cumulative probabilities.

Autocorrelations

The sample autocorrelation of lag h for a parameter θ is defined in terms of the sample autocovariance function:

$$\hat{\rho}_h(\theta) = \frac{\hat{\gamma}_h(\theta)}{\hat{\gamma}_0(\theta)}, \quad |h| < n$$

The sample autocovariance function of lag h of θ is defined by

$$\hat{\gamma}_h(\theta) = \frac{1}{n-h}\sum_{t=1}^{n-h}\left(\theta^{t+h} - \bar{\theta}\right)\left(\theta^t - \bar{\theta}\right), \quad 0 \le h < n$$

Effective Sample Size

You can use autocorrelation and trace plots to examine the mixing of a Markov chain. A closely related measure of mixing is the effective sample size (ESS) (Kass et al. 1998).

ESS is defined as follows:

$$\text{ESS} = \frac{n}{\tau} = \frac{n}{1 + 2\sum_{k=1}^{\infty}\rho_k(\theta)}$$

where n is the total sample size and $\rho_k(\theta)$ is the autocorrelation of lag k for θ. The quantity τ is referred to as the autocorrelation time. To estimate τ, the Bayesian procedures first find a cutoff point k after which the autocorrelations are very close to zero, and then sum all the ρ_k up to that point. The cutoff point k is such that $|\rho_k| < \min\{0.01, 2s_k\}$, where s_k is the estimated standard deviation:

$$s_k = \sqrt{\left(\frac{1}{n}\left(1 + 2\sum_{j=1}^{k-1}\hat{\rho}_j^2(\theta)\right)\right)}$$

ESS and τ are inversely proportional to each other, and low ESS or high τ indicates bad mixing of the Markov chain.

Summary Statistics

Let $\boldsymbol{\theta}$ be a p-dimensional parameter vector of interest: $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_p\}$. For each $i \in \{1, \ldots, p\}$, there are n observations: $\theta_i = \{\theta_i^t,\ t = 1, \ldots, n\}$.

Mean

The posterior mean is calculated by using the following formula:

$$E(\theta_i|\mathbf{y}) \approx \bar{\theta}_i = \frac{1}{n}\sum_{t=1}^{n}\theta_i^t, \text{ for } i = 1, \ldots, n$$

Standard Deviation

Sample standard deviation (expressed in variance term) is calculated by using the following formula:

$$\mathrm{Var}(\theta_i|\mathbf{y}) \approx s_i^2 = \frac{1}{n-1}\sum_{t=1}^{n}\left(\theta_i^t - \bar{\theta}_i\right)^2$$

Standard Error of the Mean Estimate

Suppose you have n iid samples, the mean estimate is $\bar{\theta}_i$, and the sample standard deviation is s_i. The standard error of the estimate is $\hat{\sigma}_i/\sqrt{n}$. However, positive autocorrelation (see the section "Autocorrelations" on page 174 for a definition) in the MCMC samples makes this an underestimate. To take account of the autocorrelation, the Bayesian procedures correct the standard error by using effective sample size (see the section "Effective Sample Size" on page 175).

Given an effective sample size of m, the standard error for $\bar{\theta}_i$ is $\hat{\sigma}_i/\sqrt{m}$. The procedures use the following formula (expressed in variance term):

$$\widehat{\mathrm{Var}}(\bar{\theta}_i) = \frac{1 + 2\sum_{k=1}^{\infty}\rho_k(\theta_i)}{n} \cdot \frac{\sum_{t=1}^{n}\left(\theta_i^t - \bar{\theta}_i\right)^2}{(n-1)}$$

The standard error of the mean is also known as the Monte Carlo standard error (MCSE). The MCSE provides a measurement of the accuracy of the posterior estimates, and small values do not necessarily indicate that you have recovered the true posterior mean.

Percentiles

Sample percentiles are calculated using Definition 5 (see Chapter 3, "The UNIVARIATE Procedure" (*Base SAS Procedures Guide: Statistical Procedures*)).

Correlation

Correlation between θ_i and θ_j is calculated as

$$r_{ij} = \frac{\sum_{t=1}^{n}\left(\theta_i^t - \bar{\theta}_i\right)\left(\theta_j^t - \bar{\theta}_j\right)}{\sqrt{\sum_t\left(\theta_i^t - \bar{\theta}_i\right)^2\sum_t\left(\theta_j^t - \bar{\theta}_j\right)^2}}$$

Covariance

Covariance θ_i and θ_j is calculated as

$$s_{ij} = \sum_{t=1}^{n} \left(\theta_i^t - \bar{\theta}_i\right)\left(\theta_j^t - \bar{\theta}_j\right)/(n-1)$$

Equal-Tail Credible Interval

Let $\pi\left(\theta_i|\mathbf{y}\right)$ denote the marginal posterior cumulative distribution function of θ_i. A $100\left(1-\alpha\right)\%$ Bayesian equal-tail credible interval for θ_i is $\left(\theta_i^{\alpha/2}, \theta_i^{1-\alpha/2}\right)$, where $\pi\left(\theta_i^{\alpha/2}|\mathbf{y}\right) = \frac{\alpha}{2}$, and $\pi\left(\theta_i^{1-\alpha/2}|\mathbf{y}\right) = 1 - \frac{\alpha}{2}$. The interval is obtained using the empirical $\frac{\alpha}{2}$th and $(1-\frac{\alpha}{2})$th percentiles of $\left\{\theta_i^t\right\}$.

Highest Posterior Density (HPD) Interval

For a definition of an HPD interval, see the section "Interval Estimation" on page 153. The procedures use the Chen-Shao algorithm (Chen and Shao 1999; Chen, Shao, and Ibrahim 2000) to estimate an empirical HPD interval of θ_i:

1. Sort $\left\{\theta_i^t\right\}$ to obtain the ordered values:

   $$\theta_{i(1)} \leq \theta_{i(2)} \leq \cdots \leq \theta_{i(n)}$$

2. Compute the $100\left(1-\alpha\right)\%$ credible intervals:

   $$R_j\left(n\right) = \left(\theta_{i(j)}, \theta_{i(j+[(1-\alpha)n])}\right)$$

   for $j = 1, 2, \ldots, n - [(1-\alpha)\,n]$.

3. The $100\left(1-\alpha\right)\%$ HPD interval, denoted by $R_{j*}\left(n\right)$, is the one with the smallest interval width among all credible intervals.

Deviance Information Criterion (DIC)

The deviance information criterion (DIC) (Spiegelhalter et al. 2002) is a model assessment tool, and it is a Bayesian alternative to Akaike's information criterion (AIC) and the Bayesian information criterion (BIC, also known as the Schwarz criterion). The DIC uses the posterior densities, which means that it takes the prior information into account. The criterion can be applied to nonnested models and models that have non-iid data. Calculation of the DIC in MCMC is trivial—it does not require maximization over the parameter space, like the AIC and BIC. A smaller DIC indicates a better fit to the data set.

Letting $\boldsymbol{\theta}$ be the parameters of the model, the deviance information formula is

$$\mathrm{DIC} = \overline{D(\boldsymbol{\theta})} + p_D = D(\overline{\boldsymbol{\theta}}) + 2p_D$$

where

$D(\boldsymbol{\theta}) = 2\left(\log(f(\mathbf{y})) - \log(p(\mathbf{y}|\boldsymbol{\theta}))\right)$: deviance

where

$p(\mathbf{y}|\boldsymbol{\theta})$: likelihood function with the normalizing constants.

$f(\mathbf{y})$: a standardizing term that is a function of the data alone. This term is constant with respect to the parameter and is irrelevant when you compare different models that have the same likelihood function. Since the term cancels out in DIC comparisons, its calculation is often omitted.

NOTE: You can think of the deviance as the difference in twice the log likelihood between the saturated, $f(\mathbf{y})$, and fitted, $p(\mathbf{y}|\boldsymbol{\theta})$, models.

$\overline{\boldsymbol{\theta}}$: posterior mean, approximated by $\frac{1}{n}\sum_{t=1}^{n}\boldsymbol{\theta}^t$

$\overline{D(\boldsymbol{\theta})}$: posterior mean of the deviance, approximated by $\frac{1}{n}\sum_{t=1}^{n} D(\boldsymbol{\theta}^t)$. The expected deviation measures how well the model fits the data.

$D(\overline{\boldsymbol{\theta}})$: deviance evaluated at $\bar{\boldsymbol{\theta}}$, equal to $-2\log(p(\mathbf{y}|\bar{\boldsymbol{\theta}}))$. It is the deviance evaluated at your "best" posterior estimate.

p_D: effective number of parameters. It is the difference between the measure of fit and the deviance at the estimates: $\overline{D(\boldsymbol{\theta})} - D(\overline{\boldsymbol{\theta}})$. This term describes the complexity of the model, and it serves as a penalization term that corrects deviance's propensity toward models with more parameters.

A Bayesian Reading List

This section lists a number of Bayesian textbooks of varying levels.

Introductory Books

Berry, D. A. (1996). *Statistics: A Bayesian Perspective*. Belmont, CA: Duxbury Press.

Bolstad, W. M. (2016). *Introduction to Bayesian Statistics*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

DeGroot, M. H., and Schervish, M. J. (2011). *Probability and Statistics*. 4th ed. Reading, MA: Addison-Wesley.

Gamerman, D., and Lopes, H. F. (2006). *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*. 2nd ed. Boca Raton, FL: Chapman & Hall/CRC.

Ghosh, J. K., Delampady, M., and Samanta, T. (2006). *An Introduction to Bayesian Analysis: Theory and Methods*. New York: Springer-Verlag.

Lee, P. M. (2012). *Bayesian Statistics: An Introduction*. 4th ed. London: Edward Arnold.

Lesaffre, E., Lawson A. B. (2012). *Bayesian Biostatistics*. Chichester, UK: John Wiley & Sons.

Sivia, D. S. (2006). *Data Analysis: A Bayesian Tutorial*. 2nd ed. Oxford: Oxford University Press.

Stone, J. V. (2013). *Bayes' Rule: A Tutorial Introduction to Bayesian Analysis*. Sheffield, UK: Sebtel Press.

Intermediate-Level Books

Box, G. E. P., and Tiao, G. C. (1992). *Bayesian Inference in Statistical Analysis*. New York: John Wiley & Sons.

Chen, M.-H., Shao, Q.-M., and Ibrahim, J. G. (2000). *Monte Carlo Methods in Bayesian Computation*. New York: Springer-Verlag.

Gelman, A., and Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge: Cambridge University Press.

Goldstein, M., and Wooff, D. A. (2007). *Bayes Linear Statistics: Theory and Methods*. Chichester, UK: John Wiley & Sons.

Harney, H. L. (2016). *Bayesian Inference: Data Evaluation and Decisions*. 2nd ed. Berlin: Springer-Verlag.

Kadane, J. B. (2011). *Principles of Uncertainty*. London: Chapman & Hall/CRC.

Leonard, T., and Hsu, J. S. J. (1999). *Bayesian Methods: An Analysis for Statisticians and Interdisciplinary Researchers*. Cambridge: Cambridge University Press.

Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. New York: Springer-Verlag.

Marin, J.-M., and Robert, C. P. (2007). *Bayesian Core: A Practical Approach to Computational Bayesian Statistics*. New York: Springer-Verlag.

Press, S. J. (2002). *Subjective and Objective Bayesian Statistics: Principles, Models, and Applications*. 2nd ed. New York: Wiley-Interscience.

Robert, C. P. (2001). *The Bayesian Choice*. 2nd ed. New York: Springer-Verlag.

Robert, C. P., and Casella, G. (2004). *Monte Carlo Statistical Methods*. 2nd ed. New York: Springer-Verlag.

Tanner, M. A. (1996). *Tools for Statistical Inference: Methods for the Exploration of Posterior Distributions and Likelihood Functions*. 3rd ed. New York: Springer-Verlag.

Advanced Titles

Berger, J. O. (1993). *Statistical Decision Theory and Bayesian Analysis*. 2nd ed. New York: Springer-Verlag.

Bernardo, J. M., and Smith, A. F. M. (2007). *Bayesian Theory*. 2nd ed. Chichester, UK: John Wiley & Sons.

Damien, P., Dellaportas, P., Polson, N. G., and Stephens, D. A. (2013) *Bayesian Theory and Applications*. Oxford: Oxford University Press.

De Finetti, B. (1992). *Theory of Probability: A Critical Introductory Treatment*. Chichester, UK: John Wiley & Sons.

Jeffreys, H. (1998). *Theory of Probability*. 3rd ed. Oxford: Oxford University Press.

Kadane, J. B. (2016). *Pragmatics of Uncertainty*. London: Chapman & Hall/CRC.

O'Hagan, A., and Foster, J. J. (2004). *Bayesian Inference*. 2nd ed. Volume 2B of Kendall's Advanced Theory of Statistics. London: Edward Arnold.

Savage, L. J. (1954). *The Foundations of Statistics*. New York: John Wiley & Sons.

Books Motivated by Statistical Applications and Data Analysis

Carlin, B. P., and Louis, T. A. (2008). *Bayesian Methods for Data Analysis*. 3rd ed. London: Chapman & Hall.

Congdon, P. (2014). *Applied Bayesian Modeling*. 2nd ed. New York: John Wiley & Sons.

Congdon, P. (2005). *Bayesian Models for Categorical Data*. Chichester, UK: John Wiley & Sons.

Congdon, P. (2006). *Bayesian Statistical Modeling*. 2nd ed. New York: John Wiley & Sons.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, D., and Rubin, D. B. (2013). *Bayesian Data Analysis*. 3nd ed. London: Chapman & Hall.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996). *Markov Chain Monte Carlo in Practice*. London: Chapman & Hall.

References

Amit, Y. (1991). "On Rates of Convergence of Stochastic Relaxation for Gaussian and Non-Gaussian Distributions." *Journal of Multivariate Analysis* 38:82–99.

Applegate, D. L., Kannan, R., and Polson, N. (1990). *Random Polynomial Time Algorithms for Sampling from Joint Distributions*. Technical report, School of Computer Science, Carnegie Mellon University.

Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis*. 2nd ed. New York: Springer-Verlag.

Berger, J. O. (2006). "The Case for Objective Bayesian Analysis." *Bayesian Analysis* 3:385–402. http://www.stat.cmu.edu/bayesworkshop/2005/berger.pdf.

Berger, J. O., and Wolpert, R. (1988). *The Likelihood Principle*. 2nd ed. Hayward, CA: Institute of Mathematical Statistics.

Bernardo, J. M., and Smith, A. F. M. (1994). *Bayesian Theory*. New York: John Wiley & Sons.

Besag, J. (1974). "Spatial Interaction and the Statistical Analysis of Lattice Systems." *Journal of the Royal Statistical Society, Series B* 36:192–236.

Billingsley, P. (1986). *Probability and Measure*. 2nd ed. New York: John Wiley & Sons.

Box, G. E. P., and Tiao, G. C. (1973). *Bayesian Inference in Statistical Analysis*. New York: John Wiley & Sons.

Breiman, L. (1968). *Probability*. Reading, MA: Addison-Wesley.

Brooks, S. P., and Gelman, A. (1997). "General Methods for Monitoring Convergence of Iterative Simulations." *Journal of Computational and Graphical Statistics* 7:434–455.

Brooks, S. P., and Roberts, G. O. (1998). "Assessing Convergence of Markov Chain Monte Carlo Algorithms." *Statistics and Computing* 8:319–335.

Brooks, S. P., and Roberts, G. O. (1999). "On Quantile Estimation and Markov Chain Monte Carlo Convergence." *Biometrika* 86:710–717.

Carlin, B. P., and Louis, T. A. (2000). *Bayes and Empirical Bayes Methods for Data Analysis*. 2nd ed. London: Chapman & Hall.

Casella, G., and George, E. I. (1992). "Explaining the Gibbs Sampler." *American Statistician* 46:167–174.

Chan, K. S. (1993). "Asymptotic Behavior of the Gibbs Sampler." *Journal of the American Statistical Association* 88:320–326.

Chen, M.-H., and Shao, Q.-M. (1999). "Monte Carlo Estimation of Bayesian Credible and HPD Intervals." *Journal of Computational and Graphical Statistics* 8:69–92.

Chen, M.-H., Shao, Q.-M., and Ibrahim, J. G. (2000). *Monte Carlo Methods in Bayesian Computation*. New York: Springer-Verlag.

Chib, S., and Greenberg, E. (1995). "Understanding the Metropolis-Hastings Algorithm." *American Statistician* 49:327–335.

Congdon, P. (2001). *Bayesian Statistical Modeling*. Chichester, UK: John Wiley & Sons.

Congdon, P. (2003). *Applied Bayesian Modeling*. New York: John Wiley & Sons.

Congdon, P. (2005). *Bayesian Models for Categorical Data*. Chichester, UK: John Wiley & Sons.

Cowles, M. K., and Carlin, B. P. (1996). "Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review." *Journal of the American Statistical Association* 91:883–904.

DeGroot, M. H., and Schervish, M. J. (2002). *Probability and Statistics*. 3rd ed. Reading, MA: Addison-Wesley.

Feller, W. (1968). *An Introduction to Probability Theory and Its Applications*. 3rd ed. New York: John Wiley & Sons.

Gamerman, D. (1997). "Sampling from the Posterior Distribution in Generalized Linear Models." *Statistics and Computing* 7:57–68.

Gelfand, A. E., Hills, S. E., Racine-Poon, A., and Smith, A. F. M. (1990). "Illustration of Bayesian Inference in Normal Data Models Using Gibbs Sampling." *Journal of the American Statistical Association* 85:972–985.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. 2nd ed. London: Chapman & Hall.

Gelman, A., and Rubin, D. B. (1992). "Inference from Iterative Simulation Using Multiple Sequences." *Statistical Science* 7:457–472.

Geman, S., and Geman, D. (1984). "Stochastic Relaxation, Gibbs Distribution, and the Bayesian Restoration of Images." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6:721–741.

Geweke, J. (1992). "Evaluating the Accuracy of Sampling-Based Approaches to Calculating Posterior Moments." In *Bayesian Statistics*, vol. 4, edited by J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, 169–193. Oxford: Clarendon Press.

Gilks, W. R. (2003). "Adaptive Metropolis Rejection Sampling (ARMS)." Software from MRC Biostatistics Unit, Cambridge, UK. http://www.maths.leeds.ac.uk/~wally.gilks/adaptive.rejection/web_page/Welcome.html.

Gilks, W. R., Best, N. G., and Tan, K. K. C. (1995). "Adaptive Rejection Metropolis Sampling within Gibbs Sampling." *Journal of the Royal Statistical Society, Series C* 44:455–472.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996). *Markov Chain Monte Carlo in Practice*. London: Chapman & Hall.

Gilks, W. R., and Wild, P. (1992). "Adaptive Rejection Sampling for Gibbs Sampling." *Journal of the Royal Statistical Society, Series C* 41:337–348.

Goldstein, M. (2006). "Subjective Bayesian Analysis: Principles and Practice." *Bayesian Analysis* 3:403–420. http://www.stat.cmu.edu/bayesworkshop/2005/goldstein.pdf.

Hastings, W. K. (1970). "Monte Carlo Sampling Methods Using Markov Chains and Their Applications." *Biometrika* 57:97–109.

Heidelberger, P., and Welch, P. D. (1981). "A Spectral Method for Confidence Interval Generation and Run Length Control in Simulations." *Communications of the ACM* 24:233–245.

Heidelberger, P., and Welch, P. D. (1983). "Simulation Run Length Control in the Presence of an Initial Transient." *Operations Research* 31:1109–1144.

Hoffman, M. D., and Gelman, A. (2014). "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research* 15:1351–1381.

Jeffreys, H. (1961). *Theory of Probability*. 3rd ed. Oxford: Oxford University Press.

Karlin, S., and Taylor, H. (1975). *A First Course in Stochastic Processes*. 2nd ed. Orlando, FL: Academic Press.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998). "Markov Chain Monte Carlo in Practice: A Roundtable Discussion." *American Statistician* 52:93–100.

Kass, R. E., and Wasserman, L. (1996). "Formal Rules of Selecting Prior Distributions: A Review and Annotated Bibliography." *Journal of the American Statistical Association* 91:343–370.

Liu, C., Wong, W. H., and Kong, A. (1991a). *Correlation Structure and Convergence Rate of the Gibbs Sampler (I): Application to the Comparison of Estimators and Augmentation Scheme*. Technical report, Department of Statistics, University of Chicago.

Liu, C., Wong, W. H., and Kong, A. (1991b). *Correlation Structure and Convergence Rate of the Gibbs Sampler (II): Applications to Various Scans*. Technical report, Department of Statistics, University of Chicago.

Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. New York: Springer-Verlag.

MacEachern, S. N., and Berliner, L. M. (1994). "Subsampling the Gibbs Sampler." *American Statistician* 48:188–190.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). "Equation of State Calculations by Fast Computing Machines." *Journal of Chemical Physics* 21:1087–1092.

Metropolis, N., and Ulam, S. (1949). "The Monte Carlo Method." *Journal of the American Statistical Association* 44:335–341.

Meyn, S. P., and Tweedie, R. L. (1993). *Markov Chains and Stochastic Stability*. Berlin: Springer-Verlag.

Neal, R. M. (2003). "Slice Sampling." *Annals of Statistics* 31:705–757.

Neal, R. M. (2011). "MCMC Using Hamiltonian Dynamics." In *Handbook of Markov Chain Monte Carlo*, edited by S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng, 113–161. Boca Raton, FL: CRC Press.

Press, S. J. (2003). *Subjective and Objective Bayesian Statistics*. New York: John Wiley & Sons.

Raftery, A. E., and Lewis, S. M. (1992). "One Long Run with Diagnostics: Implementation Strategies for Markov Chain Monte Carlo." *Statistical Science* 7:493–497.

Raftery, A. E., and Lewis, S. M. (1995). "The Number of Iterations, Convergence Diagnostics, and Generic Metropolis Algorithms." In *Markov Chain Monte Carlo in Practice*, edited by W. R. Gilks, D. J. Spiegelhalter, and S. Richardson, 115–130. London: Chapman & Hall.

Robert, C. P. (2001). *The Bayesian Choice*. 2nd ed. New York: Springer-Verlag.

Robert, C. P., and Casella, G. (2004). *Monte Carlo Statistical Methods*. 2nd ed. New York: Springer-Verlag.

Roberts, G. O. (1996). "Markov Chain Concepts Related to Sampling Algorithms." In *Markov Chain Monte Carlo in Practice*, edited by W. R. Gilks, D. J. Spiegelhalter, and S. Richardson, 45–58. London: Chapman & Hall.

Rosenthal, J. S. (1991a). *Rates of Convergence for Data Augmentation on Finite Sample Spaces*. Technical report, Department of Mathematics, Harvard University.

Rosenthal, J. S. (1991b). *Rates of Convergence for Gibbs Sampling for Variance Component Models*. Technical report, Department of Mathematics, Harvard University.

Ross, S. M. (1997). *Simulation*. 2nd ed. Orlando, FL: Academic Press.

Schervish, M. J., and Carlin, B. P. (1992). "On the Convergence of Successive Substitution Sampling." *Journal of Computational and Graphical Statistics* 1:111–127.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and Van der Linde, A. (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society, Series B* 64:583–616. With discussion.

Tanner, M. A. (1993). *Tools for Statistical Inference: Methods for the Exploration of Posterior Distributions and Likelihood Functions*. New York: Springer-Verlag.

Tanner, M. A., and Wong, W. H. (1987). "The Calculation of Posterior Distributions by Data Augmentation." *Journal of the American Statistical Association* 82:528–540.

Tierney, L. (1994). "Markov Chains for Exploring Posterior Distributions." *Annals of Statistics* 22:1701–1762.

Von Mises, R. (1918). "Über die 'Ganzzahligkeit' der Atomgewichte und verwandte Fragen [About the 'integer nature' of atomic masses and related questions]." *Physikalische Zeitschrift* 19:490–500.

Wasserman, L. (2004). *All of Statistics: A Concise Course in Statistical Inference*. New York: Springer-Verlag.

Chapter 9
Introduction to Categorical Data Analysis Procedures

Contents

Overview: Categorical Data Analysis Procedures **185**
Introduction **187**
Sampling Frameworks and Distribution Assumptions **188**
Simple Random Sampling: One Population 188
Stratified Simple Random Sampling: Multiple Populations 189
Observational Data: Analyzing the Entire Population 190
Randomized Experiments 191
Relaxation of Sampling Assumptions 192
Comparison of PROC FREQ and the Modeling Procedures **192**
Comparison of Modeling Procedures **193**
Logistic Regression 194
References **195**

Overview: Categorical Data Analysis Procedures

There are two approaches to performing categorical data analyses. The first computes statistics based on tables defined by *categorical variables* (variables that assume only a limited number of discrete values), performs hypothesis tests about the association between these variables, and requires the assumption of a randomized process; following Stokes, Davis, and Koch (2000), call these methods *randomization procedures*. The other approach investigates the association by modeling a categorical response variable, regardless of whether the explanatory variables are continuous or categorical; call these methods *modeling procedures*. Several procedures in SAS/STAT software can be used for the analysis of categorical data.

The randomization procedures are:

FREQ builds frequency tables or contingency tables and can produce numerous statistics. For one-way frequency tables, it can perform tests for equal proportions, specified proportions, or the binomial proportion. For contingency tables, it can compute various tests and measures of association and agreement including chi-square statistics, odds ratios, correlation statistics, Fisher's exact test for any size two-way table, kappa, and trend tests. In addition, it performs stratified analysis, computing Cochran-Mantel-Haenszel statistics and estimates of the common relative risk. Exact p-values and confidence intervals are available for various test statistics and measures. See Chapter 47, "The FREQ Procedure," for more information.

SURVEYFREQ incorporates complex sample designs to analyze one-way, two-way, and multiway crosstabulation tables. Estimates population totals and proportions and performs tests of goodness-of-fit and independence. See Chapter 15, "Introduction to Survey Procedures," and Chapter 118, "The SURVEYFREQ Procedure," for more information.

The modeling procedures, which require a categorical response variable, are:

CATMOD fits linear models to functions of categorical data, facilitating such analyses as regression, analysis of variance, linear modeling, log-linear modeling, logistic regression, and repeated measures analysis. Maximum likelihood estimation is used for the analysis of logits and generalized logits, and weighted least squares analysis is used for fitting models to other response functions. Iterative proportional fitting (IPF), which avoids the need for parameter estimation, is available for fitting hierarchical log-linear models when there is a single population. See Chapter 36, "The CATMOD Procedure," for more information.

GENMOD fits generalized linear models with maximum-likelihood methods. This family includes logistic, probit, and complementary log-log regression models for binomial data, Poisson and negative binomial regression models for count data, and multinomial models for ordinal response data. It performs likelihood ratio and Wald tests for Type I, Type III, and user-defined contrasts. It analyzes repeated measures data with generalized estimating equation (GEE) methods. Bayesian analysis capabilities for generalized linear models are also available. See Chapter 51, "The GENMOD Procedure," for more information.

GLIMMIX fits generalized linear mixed models with maximum-likelihood methods. If the model does not contain random effects, the GLIMMIX procedure fits generalized linear models by the method of maximum likelihood. This family includes logistic, probit, and complementary log-log regression models for binomial data, Poisson and negative binomial regression models for count data, and multinomial models for ordinal response data. See Chapter 52, "The GLIMMIX Procedure," for more information.

LOGISTIC fits linear logistic regression models for discrete response data with maximum-likelihood methods. It provides four variable selection methods, computes regression diagnostics, and compares and outputs receiver operating characteristic curves. It can also perform stratified conditional logistic regression analysis for binary response data and exact conditional regression analysis for binary and nominal response data. The logit link function in the logistic regression models can be replaced by the probit function or the complementary log-log function. See Chapter 79, "The LOGISTIC Procedure," for more information.

PROBIT fits models with probit, logit, or complementary log-log links for quantal assay or other discrete event data. It is mainly designed for dose-response analysis with a natural response rate. It computes the fiducial limits for the dose variable and provides various graphical displays for the analysis. See Chapter 100, "The PROBIT Procedure," for more information.

SURVEYLOGISTIC fits logistic models for binary and ordinal outcomes to survey data by maximum likelihood, incorporating complex survey sample designs. See Chapter 120, "The SURVEYLOGISTIC Procedure," for more information.

Also see Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," and Chapter 4, "Introduction to Regression Procedures," for more information about all the modeling and regression procedures.

Other procedures that can be used for categorical data analysis and modeling are:

CORRESP performs simple and multiple correspondence analyses, using a contingency table, Burt table, binary table, or raw categorical data as input. See Chapter 10, "Introduction to Multivariate Procedures," and Chapter 41, "The CORRESP Procedure," for more information.

PRINQUAL performs a principal component analysis of qualitative and/or quantitative data, and multidimensional preference analysis. See Chapter 10, "Introduction to Multivariate Procedures," and Chapter 99, "The PRINQUAL Procedure," for more information.

TRANSREG fits univariate and multivariate linear models, optionally with spline and other nonlinear transformations. Models include ordinary regression and ANOVA, multiple and multivariate regression, metric and nonmetric conjoint analysis, metric and nonmetric vector and ideal point preference mapping, redundancy analysis, canonical correlation, and response surface regression. See Chapter 4, "Introduction to Regression Procedures," and Chapter 126, "The TRANSREG Procedure," for more information.

Introduction

A *categorical variable* is a variable that assumes only a limited number of discrete values. The measurement scale for a categorical variable is unrestricted. It can be *nominal*, which means that the observed levels are not ordered. It can be *ordinal*, which means that the observed levels are ordered in some way. Or it can be *interval*, which means that the observed levels are ordered and numeric and that any interval of one unit on the scale of measurement represents the same amount, regardless of its location on the scale. One example of a categorical variable is litter size; another is the number of times a subject has been married. A variable that lies on a nominal scale is sometimes called a *qualitative* or *classification variable*.

Categorical data result from observations on multiple subjects where one or more categorical variables are observed for each subject. If there is only one categorical variable, then the data are generally represented by a *frequency table*, which lists each observed value of the variable and its frequency of occurrence.

If there are two or more categorical variables, then a subject's *profile* is defined as the subject's observed values for each of the variables. Such categorical data can be represented by a frequency table that lists each observed profile and its frequency of occurrence.

If there are exactly two categorical variables, then the data are often represented by a two-dimensional *contingency table*, which has one row for each level of variable 1 and one column for each level of variable 2. The intersections of rows and columns, called *cells*, correspond to variable profiles, and each cell contains the frequency of occurrence of the corresponding profile.

If there are more than two categorical variables, then the data can be represented by a *multidimensional contingency table*. There are two commonly used methods for displaying such tables, and both require that the variables be divided into two sets.

- In the first method, one set contains a row variable and a column variable for a two-dimensional contingency table, and the second set contains all of the other variables. The variables in the second set are used to form a set of profiles. Thus, the data are represented as a series of two-dimensional

contingency tables, one for each profile. This is the data representation used by PROC FREQ. For example, if you request tables for RACE*SEX*AGE*INCOME, the FREQ procedure represents the data as a series of contingency tables: the row variable is AGE, the column variable is INCOME, and the combinations of levels of RACE and SEX form a set of profiles.

- In the second method, one set contains the independent variables, and the other set contains the dependent variables. Profiles based on the independent variables are called *population profiles*, whereas those based on the dependent variables are called *response profiles*. A two-dimensional contingency table is then formed, with one row for each population profile and one column for each response profile. Since any subject can have only one population profile and one response profile, the contingency table is uniquely defined. This is the data representation used by the modeling procedures.

NOTE: Modeling procedures for categorical data analysis only require that the response variable be categorical—the explanatory variables are allowed to be continuous or categorical. However, note that PROC CATMOD was designed to handle contingency table data, and it does not efficiently handle continuous covariates.

Sampling Frameworks and Distribution Assumptions

This section discusses the sampling frameworks and distribution assumptions for the modeling and randomization procedures.

Simple Random Sampling: One Population

Suppose you take a simple random sample of 100 people and ask each person the following question, "Of the three colors red, blue, and green, which is your favorite?" You then tabulate the results in a frequency table as shown in Table 9.1.

Table 9.1 One-Way Frequency Table

	Favorite Color			
	Red	**Blue**	**Green**	**Total**
Frequency	52	31	17	100
Proportion	0.52	0.31	0.17	1.00

In the population you are sampling, you assume there is an unknown probability that a population member, selected at random, would choose any given color. In order to estimate that probability, you use the sample proportion

$$p_j = \frac{n_j}{n}$$

where n_j is the frequency of the jth response and n is the total frequency.

Because of the random variation inherent in any random sample, the frequencies have a probability distribution representing their relative frequency of occurrence in a hypothetical series of samples. For a simple random

sample, the distribution of frequencies for a frequency table with three levels is as follows. The probability that the first frequency is n_1, the second frequency is n_2, and the third is $n_3 = n - n_1 - n_2$, is given by

$$\Pr(n_1, n_2, n_3) = \frac{n!}{n_1! n_2! n_3!} \pi_1^{n_1} \pi_2^{n_2} \pi_3^{n_3}$$

where π_j is the true probability of observing the *j*th response level in the population.

This distribution, called the *multinomial distribution*, can be generalized to any number of response levels. The special case of two response levels is called the *binomial distribution*.

Simple random sampling is the type of sampling required by the (non-survey) modeling procedures when there is one population. The modeling procedures use the multinomial distribution to estimate a probability vector and its covariance matrix. If the sample size is sufficiently large, then the probability vector is approximately normally distributed as a result of central limit theory. This result is used to compute appropriate test statistics for the specified statistical model.

Stratified Simple Random Sampling: Multiple Populations

Suppose you take two simple random samples, 50 men and 50 women, and ask the same question as before. You are now sampling two different populations that may have different response probabilities. The data can be tabulated as shown in Table 9.2.

Table 9.2 Two-Way Contingency Table: Sex by Color

	Favorite Color			
Sex	**Red**	**Blue**	**Green**	**Total**
Male	30	10	10	50
Female	20	10	20	50
Total	50	20	30	100

Note that the row marginal totals (50, 50) of the contingency table are fixed by the sampling design, but the column marginal totals (50, 20, 30) are random. There are six probabilities of interest for this table, and they are estimated by the sample proportions

$$p_{ij} = \frac{n_{ij}}{n_i}$$

where n_{ij} denotes the frequency for the *i*th population and the *j*th response and n_i is the total frequency for the *i*th population. For this contingency table, the sample proportions are shown in Table 9.3.

Table 9.3 Table of Sample Proportions by Sex

	Favorite Color			
Sex	**Red**	**Blue**	**Green**	**Total**
Male	0.60	0.20	0.20	1.00
Female	0.40	0.20	0.40	1.00

The probability distribution of the six frequencies is the *product multinomial distribution*

$$\Pr(n_{11}, n_{12}, n_{13}, n_{21}, n_{22}, n_{23}) = \frac{n_1! n_2! \pi_{11}^{n_{11}} \pi_{12}^{n_{12}} \pi_{13}^{n_{13}} \pi_{21}^{n_{21}} \pi_{22}^{n_{22}} \pi_{23}^{n_{23}}}{n_{11}! n_{12}! n_{13}! n_{21}! n_{22}! n_{23}!}$$

where π_{ij} is the true probability of observing the jth response level in the ith population. The product multinomial distribution is simply the product of two or more individual multinomial distributions since the populations are independent. This distribution can be generalized to any number of populations and response levels.

Stratified simple random sampling is the type of sampling required by the modeling procedures when there is more than one population. The product multinomial distribution is used to estimate a probability vector and its covariance matrix. If the sample sizes are sufficiently large, then the probability vector is approximately normally distributed as a result of central limit theory, and this result is used to compute appropriate test statistics for the specified statistical model. The statistics are known as Wald statistics, and they are approximately distributed as chi-square when the null hypothesis is true.

Observational Data: Analyzing the Entire Population

Sometimes the observed data do not come from a random sample but instead represent a complete set of observations on some population. For example, suppose a class of 100 students is classified according to sex and favorite color. The results are shown in Table 9.4.

In this case, you could argue that all of the frequencies are fixed since the entire population is observed; therefore, there is no sampling error. On the other hand, you could hypothesize that the observed table has only fixed marginals and that the cell frequencies represent one realization of a conceptual process of assigning color preferences to individuals. The assignment process is open to hypothesis, which means that you can hypothesize restrictions on the joint probabilities.

Table 9.4 Two-Way Contingency Table: Sex by Color

	Favorite Color			
Sex	**Red**	**Blue**	**Green**	**Total**
Male	16	21	20	57
Female	12	20	11	43
Total	28	41	31	100

The usual hypothesis (sometimes called *randomness*) is that the distribution of the column variable (Favorite Color) does not depend on the row variable (Sex). This implies that, for each row of the table, the assignment process corresponds to a simple random sample (without replacement) from the finite population represented by the column marginal totals (or by the column marginal subtotals that remain after sampling other rows). The hypothesis of randomness implies that the probability distribution on the frequencies in the table is the *hypergeometric distribution*.

If the same row and column variables are observed for each of several populations, then the probability distribution of all the frequencies can be called the *multiple hypergeometric distribution.* Each population is called a *stratum*, and an analysis that draws information from each stratum and then summarizes across

them is called a *stratified analysis* (or a *blocked analysis* or a *matched analysis*). PROC FREQ does such a stratified analysis, computing test statistics and measures of association.

In general, the populations are formed on the basis of cross-classifications of independent variables. Stratified analysis is a method of adjusting for the effect of these variables without being forced to estimate parameters for them. Note that PROC LOGISTIC can perform analyses on stratified tables as well, using the usual modeling procedure assumptions, by using conditional or exact conditional logistic regression.

The multiple hypergeometric distribution is the one used by PROC FREQ for the computation of Cochran-Mantel-Haenszel statistics. These statistics are in the class of *randomization model test statistics*, which require minimal assumptions for their validity. PROC FREQ uses the multiple hypergeometric distribution to compute the mean and the covariance matrix of a function vector in order to measure the deviation between the observed and expected frequencies with respect to a particular type of alternative hypothesis. If the cell frequencies are sufficiently large, then the function vector is approximately normally distributed as a result of central limit theory, and PROC FREQ uses this result to compute a quadratic form that has a chi-square distribution when the null hypothesis is true.

Randomized Experiments

Consider a *randomized experiment* in which patients are assigned to one of two treatment groups according to a randomization process that allocates 50 patients to each group. After a specified period of time, each patient's status (cured or not cured) is recorded. Suppose the data shown in Table 9.5 give the results of the experiment. The null hypothesis is that the two treatments are equally effective. Under this hypothesis, treatment is a randomly assigned label that has no effect on the cure rate of the patients. But this implies that each row of the table represents a simple random sample from the finite population whose cure rate is described by the column marginal totals. Therefore, the column marginals (58, 42) are fixed under the hypothesis. Since the row marginals (50, 50) are fixed by the allocation process, the hypergeometric distribution is induced on the cell frequencies. Randomized experiments can also be specified in a stratified framework, and Cochran-Mantel-Haenszel statistics can be computed relative to the corresponding multiple hypergeometric distribution.

Table 9.5 Two-Way Contingency Table: Treatment by Status

	Status		
Treatment	**Cured**	**Not Cured**	**Total**
1	36	14	50
2	22	28	50
Total	58	42	100

Relaxation of Sampling Assumptions

As indicated previously, the modeling procedures assume that the data are from a stratified simple random sample, so they use the product multinomial distribution. If the data are not from such a sample, then in many cases it is still possible to use a modeling procedure by arguing that each row of the contingency table *does* represent a simple random sample from some hypothetical population. The extent to which the inferences are generalizable depends on the extent to which the hypothetical population is perceived to resemble the target population.

Similarly, the Cochran-Mantel-Haenszel statistics use the multiple hypergeometric distribution, which requires fixed row and column marginal totals in each contingency table. If the sampling process does not yield a table with fixed margins, then it is usually possible to fix the margins through conditioning arguments similar to the ones used by Fisher when he developed the Exact Test for 2×2 tables. In other words, if you want fixed marginal totals, you can generally make your analysis conditional on those observed totals.

For more information on sampling models for categorical data, see Bishop, Fienberg, and Holland (1975, Chapter 13) and Agresti (2002, Chapter 1.2).

Comparison of PROC FREQ and the Modeling Procedures

PROC FREQ is used primarily to investigate the relationship between two variables; any confounding variables are taken into account by stratification rather than by parameter estimation. Modeling procedures are used to investigate the relationship among many variables, all of which are integrated into a parametric model.

When a modeling procedure estimates the covariance matrix of the frequencies, it assumes that the frequencies were obtained by a stratified simple random-sampling procedure. However, some modeling procedures can handle different sampling methods. PROC CATMOD can analyze input data that consists of a function vector and a covariance matrix, so you can estimate the covariance matrix of the frequencies in the appropriate manner before modeling the data. PROC SURVEYLOGISTIC can analyze data from a completely different, but known, sampling scheme.

For the FREQ procedure, Fisher's Exact Test and Cochran-Mantel-Haenszel (CMH) statistics are based on the hypergeometric distribution, which corresponds to fixed marginal totals. However, by conditioning arguments, these tests are generally applicable to a wide range of sampling procedures. Similarly, the Pearson and likelihood-ratio chi-square statistics can be derived under a variety of sampling situations.

PROC FREQ can do some traditional nonparametric analysis (such as the Kruskal-Wallis test and Spearman's correlation) since it can generate rank scores internally. Fisher's Exact Test and the CMH statistics are also inherently nonparametric. However, the main vehicle for nonparametric analyses in the SAS System is the NPAR1WAY procedure.

A large sample size is required for the validity of the chi-square distributions, the standard errors, and the covariance matrices for PROC FREQ and the modeling procedures. If sample size is a problem, then PROC FREQ has the advantage with its CMH statistics because it does not use any degrees of freedom to estimate parameters for confounding variables. In addition, PROC FREQ can compute exact p-values for any two-way table, provided that the sample size is sufficiently small in relation to the size of the table. It can also produce exact p-values for many tests, including the test of binomial proportions, the Cochran-Armitage test for trend,

and the Jonckheere-Terpstra test for ordered differences among classes. PROC LOGISTIC can perform exact conditional logistic regression and Firth's penalized-likelihood regression to compensate for small sample sizes.

See the procedure chapters for more information. In addition, some well-known texts that deal with analyzing categorical data are listed in the "References" section of this chapter.

Comparison of Modeling Procedures

The CATMOD, GENMOD, GLIMMIX, LOGISTIC, PROBIT, and SURVEYLOGISTIC procedures can all be used for statistical modeling of categorical data.

The CATMOD procedure treats all explanatory (independent) variables as classification variables by default, and you specify continuous covariates in the DIRECT statement. The other procedures treat covariates as continuous by default, and you specify the classification variables in the CLASS statement.

The CATMOD procedure provides weighted least squares estimation of many response functions, such as means, cumulative logits, and proportions, and you can also compute and analyze other response functions that can be formed from the proportions corresponding to the rows of a contingency table. In addition, a user can input and analyze a set of response functions and user-supplied covariance matrix with weighted least squares. PROC CATMOD also provides maximum likelihood estimation for binary and polytomous logistic regression.

The GENMOD procedure is also a general statistical modeling tool which fits generalized linear models to data; it fits several useful models to categorical data including logistic regression, the proportional odds model, and Poisson and negative binomial regression for count data. The GENMOD procedure also provides a facility for fitting generalized estimating equations to correlated response data that are categorical, such as repeated dichotomous outcomes. The GENMOD procedure fits models using maximum likelihood estimation. PROC GENMOD can perform Type I and Type III tests, and it provides predicted values and residuals. Bayesian analysis capabilities for generalized linear models are also available.

The GLIMMIX procedure fits many of the same models as the GENMOD procedure but also allows the inclusion of random effects. The GLIMMIX procedure fits models using maximum likelihood estimation.

The LOGISTIC procedure is specifically designed for logistic regression. It performs the usual logistic regression analysis for dichotomous outcomes and it fits the proportional odds model and the generalized logit model for ordinal and nominal outcomes, respectively, by the method of maximum likelihood. This procedure has capabilities for a variety of model-building techniques, including stepwise, forward, and backward selection. It computes predicted values, the receiver operating characteristics (ROC) curve and the area beneath the curve, and a number of regression diagnostics. It can create output data sets containing these values and other statistics. PROC LOGISTIC can perform a conditional logistic regression analysis (matched-set and case-controlled) for binary response data. For small data sets, PROC LOGISTIC can perform exact conditional logistic regression. Firth's bias-reducing penalized-likelihood method can also be used in place of conditional and exact conditional logistic regression.

The PROBIT procedure is designed for quantal assay or other discrete event data. In additional to performing the logistic regression analysis, it can estimate the threshold response rate. PROC PROBIT can also estimate the values of independent variables that yield a desired response.

The SURVEYLOGISTIC procedure performs logistic regression for binary, ordinal, and nominal responses under a specified complex sampling scheme, instead of the usual stratified simple random sampling.

Stokes, Davis, and Koch (2012) provide substantial discussion of these procedures, particularly the use of the FREQ, LOGISTIC, GENMOD, and CATMOD procedures for statistical modeling.

Logistic Regression

Dichotomous Response

You have many choices of performing logistic regression in the SAS System. The CATMOD, GENMOD, GLIMMIX, LOGISTIC, PROBIT, and SURVEYLOGISTIC procedures fit the usual logistic regression model.

PROC CATMOD might not be efficient when there are continuous independent variables with large numbers of different values. For a continuous variable with a very limited number of values, PROC CATMOD might still be useful.

PROC GLIMMIX enables you to specify random effects in the models; in particular, you can fit a random-intercept logistic regression model.

PROC LOGISTIC provides the capability of model-building and performs conditional and exact conditional logistic regression. It can also use Firth's bias-reducing penalized likelihood method.

PROC PROBIT enables you to estimate the natural response rate and compute fiducial limits for the dose variable.

The LOGISTIC, GENMOD, GLIMMIX, PROBIT, and SURVEYLOGISTIC procedures can analyze summarized data by enabling you to input the numbers of events and trials; the ratio of events to trials must be between 0 and 1.

Ordinal Response

PROC LOGISTIC fits the proportional odds model to the ordinal response data by default, PROC PROBIT fits this model if you specify the logistic distribution, and PROC GENMOD and PROC GLIMMIX fit this model if you specify the CLOGIT link and the multinomial distribution. PROC CATMOD fits the cumulative logit or adjacent-category logit response functions.

Nominal Response

When the response variable is nominal, there is no concept of ordering of the response values. Response functions called *generalized logits* can be fit by the CATMOD, GLIMMIX, and LOGISTIC procedures. PROC CATMOD fits this model by default; PROC GLIMMIX and PROC LOGISTIC require you to specify the GLOGIT link.

Numerical Differences

Differences in the way the models are parameterized and fit might result in different parameter estimates if you perform logistic regression in each of these procedures.

- Parameter estimates from the procedures can differ in sign depending on the ordering of response levels, which you can change if you want.
- The parameter estimates associated with a categorical independent variable might differ among the procedures, since the estimates depend on the coding of the indicator variables in the design matrix. By default, the design matrix column produced by PROC CATMOD and PROC LOGISTIC for a binary independent variable is coded using the values 1 and –1 (deviation from the mean coding, which is a full-rank parameterization). The same column produced by the CLASS statement of PROC GENMOD, PROC GLIMMIX, and PROC PROBIT is coded using 1 and 0 (GLM coding, which is less-than-full-rank parameterization). As a result, the parameter estimate printed by PROC LOGISTIC is one-half of the estimate produced by PROC GENMOD. Both PROC GENMOD and PROC LOGISTIC allow you to select either a full-rank parameterization or the less-than-full-rank parameterization. The GLIMMIX and PROBIT procedures allow only the less-than-full-rank parameterization for the CLASS variables. The CATMOD procedure allows only full-rank parameterizations. See the "Details" sections in the chapters on the CATMOD, GENMOD, GLIMMIX, LOGISTIC, and PROBIT procedures for more information on the generation of the design matrices used by these procedures. See Chapter 20, "Shared Concepts and Topics," for a general discussion of the various parameterizations.
- The maximum-likelihood algorithm used differs among the procedures. PROC LOGISTIC uses the Fisher's scoring method by default, while PROC PROBIT, PROC GENMOD, PROC GLIMMIX, and PROC CATMOD use the Newton-Raphson method. The parameter estimates should be the same for all three procedures, and the standard errors should be the same for the logistic model. For the normal and extreme-value (Gompertz) distributions in PROC PROBIT, which correspond to the probit and cloglog links, respectively, in PROC GENMOD and PROC LOGISTIC, the standard errors might differ. In general, tests computed using the standard errors from the Newton-Raphson method are more conservative.
- The LOGISTIC, GENMOD, GLIMMIX, and PROBIT procedures can fit a cumulative regression model for ordinal response data by using maximum-likelihood estimation. PROC LOGISTIC and PROC GENMOD use a different parameterization from that of PROC PROBIT, which results in different intercept parameters. Estimates of the slope parameters, however, should be the same for both procedures. The estimated standard errors of the slope estimates are slightly different between the procedures because of the different computational algorithms used as default.

References

Agresti, A. (1984). *Analysis of Ordinal Categorical Data*. New York: John Wiley & Sons.

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Agresti, A. (2007). *An Introduction to Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1975). *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, MA: MIT Press.

Collett, D. (2003). *Modelling Binary Data*. 2nd ed. London: Chapman & Hall.

Cox, D. R., and Snell, E. J. (1989). *The Analysis of Binary Data*. 2nd ed. London: Chapman & Hall.

Dobson, A. (1990). *An Introduction to Generalized Linear Models*. London: Chapman & Hall.

Fleiss, J. L., Levin, B., and Paik, M. C. (2003). *Statistical Methods for Rates and Proportions*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Freeman, D. H., Jr. (1987). *Applied Categorical Data Analysis*. New York: Marcel Dekker.

Grizzle, J. E., Starmer, C. F., and Koch, G. G. (1969). "Analysis of Categorical Data by Linear Models." *Biometrics* 25:489–504.

Hosmer, D. W., Jr., and Lemeshow, S. (2000). *Applied Logistic Regression*. 2nd ed. New York: John Wiley & Sons.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2000). *Categorical Data Analysis Using the SAS System*. 2nd ed. Cary, NC: SAS Institute Inc.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2012). *Categorical Data Analysis Using SAS*. 3rd ed. Cary, NC: SAS Institute Inc.

Chapter 10
Introduction to Multivariate Procedures

Contents

Overview: Multivariate Procedures 197
Comparison of the PRINCOMP and FACTOR Procedures 199
Comparison of the PRINCOMP and PRINQUAL Procedures 199
Comparison of the PRINCOMP and CORRESP Procedures 200
Comparison of the PRINQUAL and CORRESP Procedures 200
Comparison of the TRANSREG and PRINQUAL Procedures 200
References 201

Overview: Multivariate Procedures

The procedures discussed in this chapter investigate relationships among variables without designating some as independent and others as dependent. Principal component analysis and common factor analysis examine relationships within a single set of variables, whereas canonical correlation looks at the relationship between two sets of variables. The following is a brief description of SAS/STAT multivariate procedures:

CORRESP performs simple and multiple correspondence analyses, with a contingency table, Burt table, binary table, or raw categorical data as input. Correspondence analysis is a weighted form of principal component analysis that is appropriate for frequency data. The results are displayed in plots and tables and are also available in output data sets.

PRINCOMP performs a principal component analysis and outputs standardized or unstandardized principal component scores. The results are displayed in plots and tables and are also available in output data sets.

PRINQUAL performs a principal component analysis of qualitative data and multidimensional preference analysis. The results are displayed in plots and are also available in output data sets.

FACTOR performs principal component and common factor analyses with rotations and outputs component scores or estimates of common factor scores. The results are displayed in plots and tables and are also available in output data sets.

CANCORR performs a canonical correlation analysis and outputs canonical variable scores. The results are displayed in tables and are also available in output data sets for plotting.

Many other SAS/STAT procedures can also analyze multivariate data—for example, the CATMOD, GLM, REG, CALIS, and TRANSREG procedures as well as the procedures for clustering and discriminant analysis.

The purpose of *principal component analysis* (Rao 1964) is to derive a small number of linear combinations (principal components) of a set of variables that retain as much of the information in the original variables as possible. Often a small number of principal components can be used in place of the original variables for plotting, regression, clustering, and so on. Principal component analysis can also be viewed as an attempt to uncover approximate linear dependencies among variables.

The purpose of *common factor analysis* (Mulaik 1972) is to explain the correlations or covariances among a set of variables in terms of a limited number of unobservable, latent variables. The latent variables are not generally computable as linear combinations of the original variables. In common factor analysis, it is assumed that the variables are linearly related if not for uncorrelated random error or *unique variation* in each variable; both the linear relations and the amount of unique variation can be estimated.

Principal component and common factor analysis are often followed by rotation of the components or factors. *Rotation* is the application of a nonsingular linear transformation to components or common factors to aid interpretation.

The purpose of *canonical correlation analysis* (Mardia, Kent, and Bibby 1979) is to explain or summarize the relationship between two sets of variables by finding a small number of linear combinations from each set of variables that have the highest possible between-set correlations. Plots of the canonical variables can be useful in examining multivariate dependencies. If one of the two sets of variables consists of dummy variables generated from a classification variable, the canonical correlation is equivalent to canonical discriminant analysis (see Chapter 35, "The CANDISC Procedure"). If both sets of variables are dummy variables, canonical correlation is equivalent to simple correspondence analysis.

The purpose of *correspondence analysis* (Lebart, Morineau, and Warwick 1984; Greenacre 1984; Nishisato 1980) is to summarize the associations between a set of categorical variables in a small number of dimensions. Correspondence analysis computes scores on each dimension for each row and column category in a contingency table. Plots of these scores show the relationships among the categories.

The PRINQUAL procedure obtains linear and nonlinear transformations of variables by using the method of alternating least squares (Young 1981) to optimize properties of the transformed variables' covariance or correlation matrix. PROC PRINQUAL nonlinearly transforms variables, improving their fit to a principal component model. The name, PRINQUAL, for principal components of qualitative data, comes from the special case analysis of fitting a principal component model to nominal and ordinal scale of measurement variables (Young, Takane, and de Leeuw 1978). However, PROC PRINQUAL also has facilities for smoothly transforming continuous variables. All of PROC PRINQUAL's transformations are also available in the TRANSREG procedure, which fits regression models with nonlinear transformations. PROC PRINQUAL can also perform metric and nonmetric multidimensional preference (MDPREF) analyses (Carroll 1972) and produce plots of the results.

Comparison of the PRINCOMP and FACTOR Procedures

Although PROC FACTOR can be used for common factor analysis, the default method is principal components. PROC FACTOR produces the same results as PROC PRINCOMP except that scoring coefficients from PROC FACTOR are normalized to give principal component scores with unit variance, whereas PROC PRINCOMP by default produces principal component scores with variance equal to the corresponding eigenvalue. PROC PRINCOMP can also compute scores standardized to unit variance. Both procedures produce graphical results through ODS Graphics.

PROC PRINCOMP has the following advantages over PROC FACTOR:

- PROC PRINCOMP is slightly faster if a small number of components is requested.
- PROC PRINCOMP can analyze somewhat larger problems in a fixed amount of memory.
- PROC PRINCOMP can output scores from an analysis of a partial correlation or covariance matrix.
- PROC PRINCOMP is simpler to use.

PROC FACTOR has the following advantages over PROC PRINCOMP for principal component analysis:

- PROC FACTOR produces more output.
- PROC FACTOR does rotations.

If you want to perform a common factor analysis, you must use PROC FACTOR instead of PROC PRINCOMP. Principal component analysis should never be used if a common factor solution is desired (Dziuban and Harris 1973; Lee and Comrey 1979).

Comparison of the PRINCOMP and PRINQUAL Procedures

The PRINCOMP procedure performs principal component analysis. The PRINQUAL procedure finds linear and nonlinear transformations of variables to optimize properties of the transformed variables' covariance or correlation matrix. One property is the sum of the first n eigenvalues, which is a measure of the fit of a principal component model with n components. Use PROC PRINQUAL to find nonlinear transformations of your variables or to perform a multidimensional preference analysis. Use PROC PRINCOMP to fit a principal component model to your data or to PROC PRINQUAL's output data set. PROC PRINCOMP produces a report of the principal component analysis, a number of graphical displays, and output data sets. PROC PRINQUAL produces only a few graphs and an output data set.

Comparison of the PRINCOMP and CORRESP Procedures

As summarized previously, PROC PRINCOMP performs a principal component analysis of interval-scaled data. PROC CORRESP performs correspondence analysis, which is a weighted form of principal component analysis that is appropriate for frequency data. If your data are categorical, use PROC CORRESP instead of PROC PRINCOMP. Both procedures produce graphical displays of the results with ODS Graphics. The plots produced by PROC CORRESP graphically show relationships among the categories of the categorical variables.

Comparison of the PRINQUAL and CORRESP Procedures

Both PROC PRINQUAL and PROC CORRESP can be used to summarize associations among variables measured on a nominal scale. PROC PRINQUAL searches for a single nonlinear transformation of the original scoring of each nominal variable that optimizes some aspect of the covariance matrix of the transformed variables. For example, PROC PRINQUAL could be used to find scorings that maximize the fit of a principal component model with one component. PROC CORRESP uses the crosstabulations of nominal variables, not covariances, and produces multiple scores for each category of each nominal variable. The main conceptual difference between PROC PRINQUAL and PROC CORRESP is that PROC PRINQUAL assumes that the categories of a nominal variable correspond to values of a single underlying interval variable, whereas PROC CORRESP assumes that there are multiple underlying interval variables and therefore uses different category scores for each dimension of the correspondence analysis. Scores from PROC CORRESP on the first dimension match the single set of PROC PRINQUAL scores (with appropriate standardizations for both analyses).

Comparison of the TRANSREG and PRINQUAL Procedures

Both the TRANSREG and PRINQUAL procedures are data transformation procedures that have many of the same transformations. These procedures can either directly perform the specified transformation (such as taking the logarithm of the variable) or search for an optimal transformation (such as a spline with a specified number of knots). Both procedures can use an iterative, alternating least squares analysis. Both procedures create an output data set that can be used as input to other procedures. PROC PRINQUAL displays relatively little output, whereas PROC TRANSREG displays many results. PROC TRANSREG has two sets of variables, usually dependent and independent, and it fits linear models such as ordinary regression and ANOVA, multiple and multivariate regression, metric and nonmetric conjoint analysis, metric and nonmetric vector and ideal point preference mapping, redundancy analysis, canonical correlation, and response surface regression. In contrast, PROC PRINQUAL has one set of variables, fits a principal component model or multidimensional preference analysis, and can also optimize other properties of a correlation or covariance matrix. PROC TRANSREG performs hypothesis testing and can be used to code experimental designs prior to their use in other analyses. PROC TRANSREG can also perform Box-Cox transformations and fit models with smoothing spline and penalized B-spline transformations.

See Chapter 4, "Introduction to Regression Procedures," for comparisons of the TRANSREG and REG procedures.

References

Carroll, J. D. (1972). "Individual Differences and Multidimensional Scaling." In *Multidimensional Scaling: Theory and Applications in the Behavioral Sciences*, vol. 1, edited by R. N. Shepard, A. K. Romney, and S. B. Nerlove, 105–155. New York: Seminar Press.

Dziuban, C. D., and Harris, C. W. (1973). "On the Extraction of Components and the Applicability of the Factor Model." *American Educational Research Journal* 10:93–99.

Greenacre, M. J. (1984). *Theory and Applications of Correspondence Analysis*. London: Academic Press.

Lebart, L., Morineau, A., and Warwick, K. M. (1984). *Multivariate Descriptive Statistical Analysis: Correspondence Analysis and Related Techniques for Large Matrices*. New York: John Wiley & Sons.

Lee, H. B., and Comrey, A. L. (1979). "Distortions in a Commonly Used Factor Analytic Procedure." *Multivariate Behavioral Research* 14:301–321.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Mulaik, S. A. (1972). *The Foundations of Factor Analysis*. New York: McGraw-Hill.

Nishisato, S. (1980). *Analysis of Categorical Data: Dual Scaling and Its Applications*. Toronto: University of Toronto Press.

Rao, C. R. (1964). "The Use and Interpretation of Principal Component Analysis in Applied Research." *Sankhyā, Series A* 26:329–358.

Young, F. W. (1981). "Quantitative Analysis of Qualitative Data." *Psychometrika* 46:357–388.

Young, F. W., Takane, Y., and de Leeuw, J. (1978). "The Principal Components of Mixed Measurement Level Multivariate Data: An Alternating Least Squares Method with Optimal Scaling Features." *Psychometrika* 43:279–281.

Chapter 11
Introduction to Discriminant Procedures

Contents

Overview: Discriminant Procedures **203**
Background: Discriminant Procedures **204**
Example: Contrasting Univariate and Multivariate Analyses **205**
References **209**

Overview: Discriminant Procedures

The SAS procedures for discriminant analysis fit data with one classification variable and several quantitative variables. The purpose of discriminant analysis can be to find one or more of the following:

- a mathematical rule, or *discriminant function*, for guessing to which class an observation belongs, based on knowledge of the quantitative variables only
- a set of linear combinations of the quantitative variables that best reveals the differences among the classes
- a subset of the quantitative variables that best reveals the differences among the classes

The SAS discriminant procedures are as follows:

DISCRIM computes various discriminant functions for classifying observations. Linear or quadratic discriminant functions can be used for data with approximately multivariate normal within-class distributions. Nonparametric methods can be used without making any assumptions about these distributions.

CANDISC performs a canonical analysis to find linear combinations of the quantitative variables that best summarize the differences among the classes.

STEPDISC uses forward selection, backward elimination, or stepwise selection to try to find a subset of quantitative variables that best reveals differences among the classes.

Background: Discriminant Procedures

The term *discriminant analysis* (Fisher 1936; Cooley and Lohnes 1971; Tatsuoka 1971; Kshirsagar 1972; Lachenbruch 1975, 1979; Gnanadesikan 1977; Klecka 1980; Hand 1981, 1982; Silverman 1986) refers to several different types of analyses. Classificatory discriminant analysis is used to classify observations into two or more known groups on the basis of one or more quantitative variables. Classification can be done by either a parametric method or a nonparametric method in the DISCRIM procedure. A parametric method is appropriate only for approximately normal within-class distributions. The method generates either a linear discriminant function (the within-class covariance matrices are assumed to be equal) or a quadratic discriminant function (the within-class covariance matrices are assumed to be unequal).

When the distribution within each group is not assumed to have any specific distribution or is assumed to have a distribution different from the multivariate normal distribution, nonparametric methods can be used to derive classification criteria. These methods include the kernel method and nearest-neighbor methods. The kernel method uses uniform, normal, Epanechnikov, biweight, or triweight kernels in estimating the group-specific density at each observation. The within-group covariance matrices or the pooled covariance matrix can be used to scale the data.

The performance of a discriminant function can be evaluated by estimating error rates (probabilities of misclassification). Error count estimates and posterior probability error rate estimates can be evaluated with PROC DISCRIM. When the input data set is an ordinary SAS data set, the error rates can also be estimated by cross validation.

In multivariate statistical applications, the data collected are largely from distributions different from the normal distribution. Various forms of nonnormality can arise, such as qualitative variables or variables with underlying continuous but nonnormal distributions. If the multivariate normality assumption is violated, the use of parametric discriminant analysis might not be appropriate. When a parametric classification criterion (linear or quadratic discriminant function) is derived from a nonnormal population, the resulting error rate estimates might be biased.

If your quantitative variables are not normally distributed, or if you want to classify observations on the basis of categorical variables, you should consider using the CATMOD or LOGISTIC procedure to fit a categorical linear model with the classification variable as the dependent variable. Press and Wilson (1978) compare logistic regression and parametric discriminant analysis and conclude that logistic regression is preferable to parametric discriminant analysis in cases for which the variables do not have multivariate normal distributions within classes. However, if you do have normal within-class distributions, logistic regression is less efficient than parametric discriminant analysis. Efron (1975) shows that with two normal populations having a common covariance matrix, logistic regression is between one-half and two-thirds as effective as the linear discriminant function in achieving asymptotically the same error rate.

Do not confuse discriminant analysis with cluster analysis. All varieties of discriminant analysis require prior knowledge of the classes, usually in the form of a sample from each class. In cluster analysis, the data do not include information about class membership; the purpose is to construct a classification. See Chapter 12, "Introduction to Clustering Procedures."

Canonical discriminant analysis is a dimension-reduction technique related to principal components and canonical correlation, and it can be performed by both the CANDISC and DISCRIM procedures. A discriminant criterion is always derived in PROC DISCRIM. If you want canonical discriminant analysis without the use of a discriminant criterion, you should use PROC CANDISC. Stepwise discriminant analysis is a variable-selection technique implemented by the STEPDISC procedure. After selecting a subset of

variables with PROC STEPDISC, use any of the other discriminant procedures to obtain more detailed analyses. PROC CANDISC and PROC STEPDISC perform hypothesis tests that require the within-class distributions to be approximately normal, but these procedures can be used descriptively with nonnormal data.

Another alternative to discriminant analysis is to perform a series of univariate one-way ANOVAs. All three discriminant procedures provide summaries of the univariate ANOVAs. The advantage of the multivariate approach is that two or more classes that overlap considerably when each variable is viewed separately might be more distinct when examined from a multivariate point of view.

Example: Contrasting Univariate and Multivariate Analyses

Consider an artificial data set with two classes of observations indicated by 'H' and 'O'. The following statements generate and plot the data:

```
data random;
   drop n;

   Group = 'H';
   do n = 1 to 20;
      x = 4.5 + 2 * normal(57391);
      y = x + .5 + normal(57391);
      output;
   end;

   Group = 'O';
   do n = 1 to 20;
      x = 6.25 + 2 * normal(57391);
      y = x - 1 + normal(57391);
      output;
   end;

run;

proc sgplot noautolegend;
   scatter y=y x=x / markerchar=group group=group;
run;
```

The plot is shown in Figure 11.1.

Figure 11.1 Groups for Contrasting Univariate and Multivariate Analyses

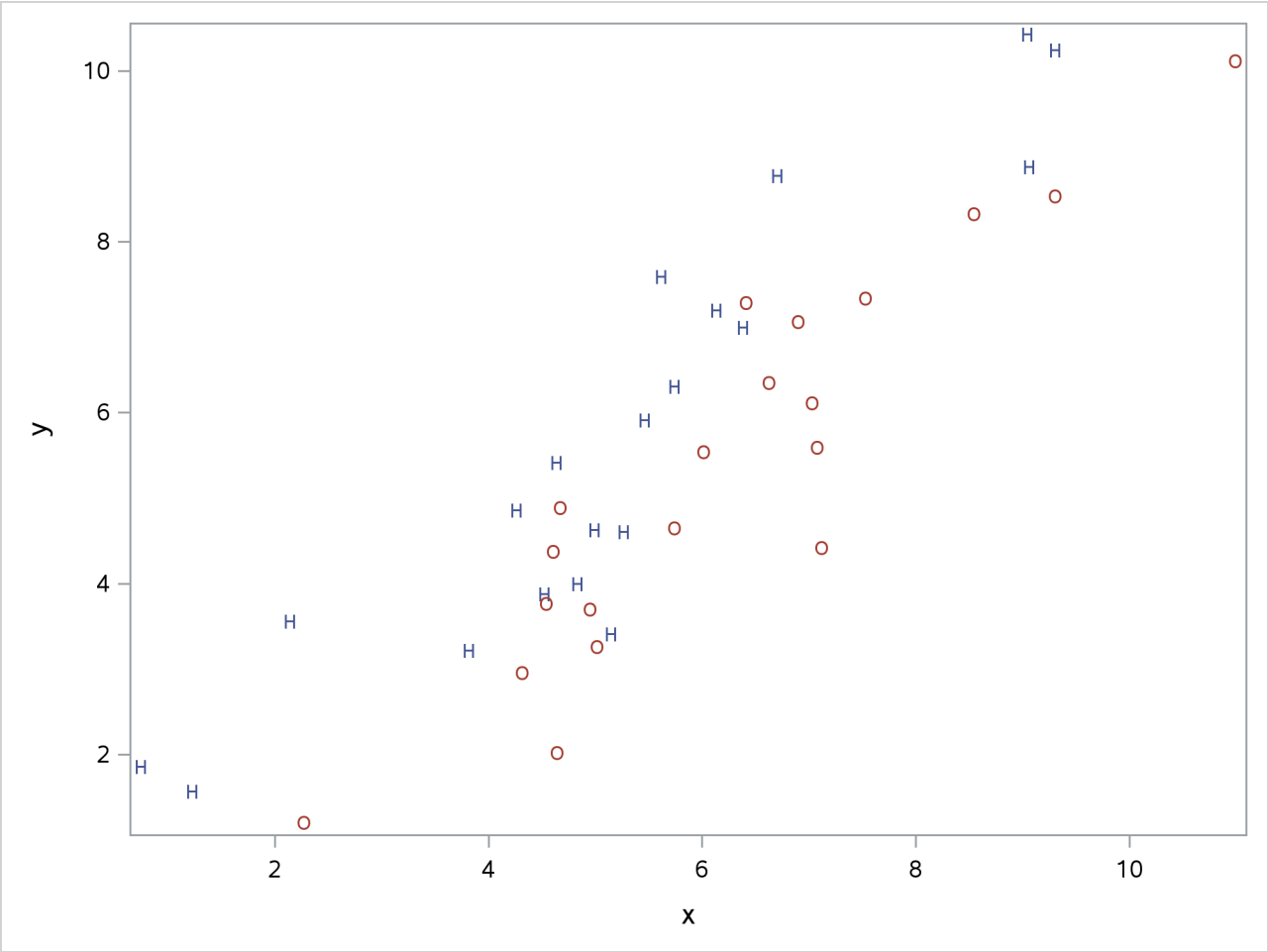

The following statements perform a canonical discriminant analysis and display the results in Figure 11.2:

```
proc candisc anova;
   class Group;
   var x y;
run;
```

Figure 11.2 Contrasting Univariate and Multivariate Analyses

The CANDISC Procedure

Total Sample Size	40	**DF Total**	39
Variables	2	**DF Within Classes**	38
Classes	2	**DF Between Classes**	1

Number of Observations Read	40
Number of Observations Used	40

Figure 11.2 *continued*

Class Level Information

Group	Variable Name	Frequency	Weight	Proportion
H	H	20	20.0000	0.500000
O	O	20	20.0000	0.500000

The CANDISC Procedure

Univariate Test Statistics

F Statistics, Num DF=1, Den DF=38

Variable	Total Standard Deviation	Pooled Standard Deviation	Between Standard Deviation	R-Square	R-Square / (1-RSq)	F Value	Pr > F
x	2.1776	2.1498	0.6820	0.0503	0.0530	2.01	0.1641
y	2.4215	2.4486	0.2047	0.0037	0.0037	0.14	0.7105

Average R-Square	
Unweighted	0.0269868
Weighted by Variance	0.0245201

Multivariate Statistics and Exact F Statistics

S=1 M=0 N=17.5

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.64203704	10.31	2	37	0.0003
Pillai's Trace	0.35796296	10.31	2	37	0.0003
Hotelling-Lawley Trace	0.55754252	10.31	2	37	0.0003
Roy's Greatest Root	0.55754252	10.31	2	37	0.0003

The CANDISC Procedure

	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq) Eigenvalue	Difference	Proportion	Cumulative
1	0.598300	0.589467	0.102808	0.357963	0.5575		1.0000	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero

	Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
1	0.64203704	10.31	2	37	0.0003

Note: The F statistic is exact.

The CANDISC Procedure

Total Canonical Structure

Variable	Can1
x	-0.374883
y	0.101206

Figure 11.2 *continued*

Between Canonical Structure	
Variable	**Can1**
x	-1.000000
y	1.000000

Pooled Within Canonical Structure	
Variable	**Can1**
x	-0.308237
y	0.081243

The CANDISC Procedure

Total-Sample Standardized Canonical Coefficients	
Variable	**Can1**
x	-2.625596855
y	2.446680169

Pooled Within-Class Standardized Canonical Coefficients	
Variable	**Can1**
x	-2.592150014
y	2.474116072

Raw Canonical Coefficients	
Variable	**Can1**
x	-1.205756217
y	1.010412967

Class Means on Canonical Variables	
Group	**Can1**
H	0.7277811475
O	-.7277811475

The univariate R squares are very small, 0.0503 for x and 0.0037 for y, and neither variable shows a significant difference between the classes at the 0.10 level.

The multivariate test for differences between the classes is significant at the 0.0003 level. Thus, the multivariate analysis has found a highly significant difference, whereas the univariate analyses failed to achieve even the 0.10 level. The raw canonical coefficients for the first canonical variable, Can1, show that the classes differ most widely on the linear combination -1.205756217 x + 1.010412967 y or approximately y

- 1.2 x. The R square between Can1 and the CLASS variable is 0.357963 as given by the squared canonical correlation, which is much higher than either univariate R square.

In this example, the variables are highly correlated within classes. If the within-class correlation were smaller, there would be greater agreement between the univariate and multivariate analyses.

References

Cooley, W. W., and Lohnes, P. R. (1971). *Multivariate Data Analysis*. New York: John Wiley & Sons.

Efron, B. (1975). "The Efficiency of Logistic Regression Compared to Normal Discriminant Analysis." *Journal of the American Statistical Association* 70:892–898.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Gnanadesikan, R. (1977). *Methods for Statistical Data Analysis of Multivariate Observations*. New York: John Wiley & Sons.

Hand, D. J. (1981). *Discrimination and Classification*. New York: John Wiley & Sons.

Hand, D. J. (1982). *Kernel Discriminant Analysis*. New York: Research Studies Press.

Klecka, W. R. (1980). *Discriminant Analysis.* Vol. 07-019 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Kshirsagar, A. M. (1972). *Multivariate Analysis*. New York: Marcel Dekker.

Lachenbruch, P. A. (1975). *Discriminant Analysis*. New York: Hafner Publishing.

Lachenbruch, P. A. (1979). "Discriminant Analysis." *Biometrics* 35:69–85.

Press, S. J., and Wilson, S. (1978). "Choosing between Logistic Regression and Discriminant Analysis." *Journal of the American Statistical Association* 73:699–705.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. New York: Chapman & Hall.

Tatsuoka, M. M. (1971). *Multivariate Analysis*. New York: John Wiley & Sons.

Chapter 12
Introduction to Clustering Procedures

Contents

Overview: Clustering Procedures **211**
Clustering Variables **213**
Clustering Observations **214**
Methods for Clustering Observations **215**
Well-Separated Clusters 215
Poorly Separated Clusters 217
Multinormal Clusters of Unequal Size and Dispersion 225
Elongated Multinormal Clusters 233
Nonconvex Clusters 240
The Number of Clusters **243**
References **246**

Overview: Clustering Procedures

You can use SAS clustering procedures to cluster the observations or the variables in a SAS data set. Both hierarchical and disjoint clusters can be obtained. Only numeric variables can be analyzed directly by the procedures, although the DISTANCE procedure can compute a distance matrix that uses character or numeric variables.

The purpose of cluster analysis is to place objects into groups, or clusters, suggested by the data, not defined a priori, such that objects in a given cluster tend to be similar to each other in some sense, and objects in different clusters tend to be dissimilar. You can also use cluster analysis to summarize data rather than to find "natural" or "real" clusters; this use of clustering is sometimes called *dissection* (Everitt 1980).

Any generalization about cluster analysis must be vague because a vast number of clustering methods have been developed in several different fields, with different definitions of clusters and similarity among objects. The variety of clustering techniques is reflected by the variety of terms used for cluster analysis: botryology, classification, clumping, competitive learning, morphometrics, nosography, nosology, numerical taxonomy, partitioning, Q-analysis, systematics, taximetrics, taxonorics, typology, unsupervised pattern recognition, vector quantization, and winner-take-all learning. Good (1977) has also suggested aciniformics and agminatics.

Several types of clusters are possible:

- Disjoint clusters place each object in one and only one cluster.

- Hierarchical clusters are organized so that one cluster can be entirely contained within another cluster, but no other kind of overlap between clusters is allowed.
- Overlapping clusters can be constrained to limit the number of objects that belong simultaneously to two clusters, or they can be unconstrained, allowing any degree of overlap in cluster membership.
- Fuzzy clusters are defined by a probability or grade of membership of each object in each cluster. Fuzzy clusters can be disjoint, hierarchical, or overlapping.

The data representations of objects to be clustered also take many forms. The most common are as follows:

- a square distance or similarity matrix, in which both rows and columns correspond to the objects to be clustered. A correlation matrix is an example of a similarity matrix.
- a coordinate matrix, in which the rows are observations and the columns are variables, as in the usual SAS multivariate data set. The observations, the variables, or both can be clustered.

The SAS procedures for clustering are oriented toward disjoint or hierarchical clusters from coordinate data, distance data, or a correlation or covariance matrix. The following procedures are used for clustering:

CLUSTER — performs hierarchical clustering of observations by using eleven agglomerative methods applied to coordinate data or distance data and draws tree diagrams, which are also called *dendrograms* or *phenograms*.

FASTCLUS — finds disjoint clusters of observations by using a k-means method applied to coordinate data. PROC FASTCLUS is especially suitable for large data sets.

MODECLUS — finds disjoint clusters of observations with coordinate or distance data by using nonparametric density estimation. It can also perform approximate nonparametric significance tests for the number of clusters.

VARCLUS — performs both hierarchical and disjoint clustering of variables by using oblique multiple-group component analysis and draws tree diagrams, which are also called *dendrograms* or *phenograms*.

TREE — draws tree diagrams, also called *dendrograms* or *phenograms*, by using output from the CLUSTER or VARCLUS procedure. PROC TREE can also create a data set indicating cluster membership at any specified level of the cluster tree.

The following procedures are useful for processing data prior to the actual cluster analysis:

ACECLUS — attempts to estimate the pooled within-cluster covariance matrix from coordinate data without knowledge of the number or the membership of the clusters (Art, Gnanadesikan, and Kettenring 1982). PROC ACECLUS outputs a data set containing canonical variable scores to be used in the cluster analysis proper.

DISTANCE — computes various measures of distance, dissimilarity, or similarity between the observations (rows) of a SAS data set. PROC DISTANCE also provides various nonparametric and parametric methods for standardizing variables. Different variables can be standardized with different methods.

PRINCOMP — performs a principal component analysis and outputs principal component scores.

STDIZE standardizes variables by using any of a variety of location and scale measures, including mean and standard deviation, minimum and range, median and absolute deviation from the median, various M-estimators and A-estimators, and some scale estimators designed specifically for cluster analysis.

Massart and Kaufman (1983) is the best elementary introduction to cluster analysis. Other important texts are Anderberg (1973); Sneath and Sokal (1973); Duran and Odell (1974); Hartigan (1975); Titterington, Smith, and Makov (1985); McLachlan and Basford (1988); Kaufman and Rousseeuw (1990). Hartigan (1975); Spath (1980) give numerous FORTRAN programs for clustering. Any prospective user of cluster analysis should study the Monte Carlo results of Milligan (1980); Milligan and Cooper (1985); Cooper and Milligan (1988). Important references on the statistical aspects of clustering include MacQueen (1967); Wolfe (1970); Scott and Symons (1971); Hartigan (1977, 1978, 1981, 1985a); Symons (1981); Everitt (1981); Sarle (1983); Bock (1985); Thode, Mendell, and Finch (1988). Bayesian methods have important advantages over maximum likelihood; see Binder (1978, 1981); Banfield and Raftery (1993); Bensmail et al. (1997). For fuzzy clustering, see Bezdek (1981); Bezdek and Pal (1992). The signal-processing perspective is provided by Gersho and Gray (1992). For a discussion of the fragmented state of the literature on cluster analysis, see Blashfield and Aldenderfer (1978).

Clustering Variables

Factor rotation is often used to cluster variables, but the resulting clusters are fuzzy. It is preferable to use PROC VARCLUS if you want hard (nonfuzzy), disjoint clusters. Factor rotation is better if you want to be able to find overlapping clusters. It is often a good idea to try both PROC VARCLUS and PROC FACTOR with an oblique rotation, compare the amount of variance explained by each, and see how fuzzy the factor loadings are and whether there seem to be overlapping clusters.

You can use PROC VARCLUS to harden a fuzzy factor rotation; use PROC FACTOR to create an output data set containing scoring coefficients and initialize PROC VARCLUS with this data set as follows:

```
proc factor rotate=promax score outstat=fact;
run;

proc varclus initial=input proportion=0;
run;
```

You can use any rotation method instead of the PROMAX method. The SCORE and OUTSTAT= options are necessary in the PROC FACTOR statement. PROC VARCLUS reads the correlation matrix from the data set created by PROC FACTOR. The INITIAL=INPUT option tells PROC VARCLUS to read initial scoring coefficients from the data set. The option PROPORTION=0 keeps PROC VARCLUS from splitting any of the clusters.

Clustering Observations

PROC CLUSTER is easier to use than PROC FASTCLUS because one run produces results from one cluster up to as many as you like. You must run PROC FASTCLUS once for each number of clusters.

The time required by PROC FASTCLUS is roughly proportional to the number of observations, whereas the time required by PROC CLUSTER with most methods varies with the square or cube of the number of observations. Therefore, you can use PROC FASTCLUS with much larger data sets than PROC CLUSTER.

If you want to hierarchically cluster a data set that is too large to use with PROC CLUSTER directly, you can have PROC FASTCLUS produce, for example, 50 clusters, and let PROC CLUSTER analyze these 50 clusters instead of the entire data set. The MEAN= data set produced by PROC FASTCLUS contains two special variables:

- The variable _FREQ_ gives the number of observations in the cluster.
- The variable _RMSSTD_ gives the root mean square across variables of the cluster standard deviations.

These variables are automatically used by PROC CLUSTER to give the correct results when clustering clusters. For example, you could specify Ward's minimum variance method Ward (1963):

```
proc fastclus maxclusters=50 mean=temp;
   var x y z;
run;

ods graphics on;
proc cluster method=ward outtree=tree;
   var x y z;
run;
```

Or you could specify Wong's hybrid method (Wong 1982):

```
proc fastclus maxclusters=50 mean=temp;
   var x y z;
run;

ods graphics on;
proc cluster method=density hybrid outtree=tree;
   var x y z;
run;
```

More detailed examples are given in Chapter 40, "The CLUSTER Procedure."

Characteristics of Methods for Clustering Observations

Many simulation studies comparing various methods of cluster analysis have been performed. In these studies, artificial data sets containing known clusters are produced using pseudo-random-number generators. The data sets are analyzed by a variety of clustering methods, and the degree to which each clustering method recovers the known cluster structure is evaluated. See Milligan (1981) for a review of such studies. In most of these studies, the clustering method with the best overall performance has been either average linkage or Ward's minimum variance method. The method with the poorest overall performance has almost invariably been single linkage. However, in many respects, the results of simulation studies are inconsistent and confusing.

When you attempt to evaluate clustering methods, it is essential to realize that most methods are biased toward finding clusters possessing certain characteristics related to size (number of members), shape, or dispersion. Methods based on the least squares criterion (Sarle 1982), such as k-means and Ward's minimum variance method, tend to find clusters with roughly the same number of observations in each cluster. Average linkage is somewhat biased toward finding clusters of equal variance. Many clustering methods tend to produce compact, roughly hyperspherical clusters and are incapable of detecting clusters with highly elongated or irregular shapes. The methods with the least bias are those based on nonparametric density estimation such as single linkage and density linkage.

Most simulation studies have generated compact (often multivariate normal) clusters of roughly equal size or dispersion. Such studies naturally favor average linkage and Ward's method over most other hierarchical methods, especially single linkage. It would be easy, however, to design a study that uses elongated or irregular clusters in which single linkage would perform much better than average linkage or Ward's method (see some of the following examples). Even studies that compare clustering methods that use "realistic" data might unfairly favor particular methods. For example, in all the data sets used by Mezzich and Solomon (1980), the clusters established by field experts are of equal size. When interpreting simulation or other comparative studies, you must, therefore, decide whether the artificially generated clusters in the study resemble the clusters you suspect might exist in your data in terms of size, shape, and dispersion. If, like many people doing exploratory cluster analysis, you have no idea what kinds of clusters to expect, you should include at least one of the relatively unbiased methods, such as density linkage, in your analysis.

The rest of this section consists of a series of examples that illustrate the performance of various clustering methods under various conditions. The first, and simplest, example shows a case of well-separated clusters. The other examples show cases of poorly separated clusters, clusters of unequal size, parallel elongated clusters, and nonconvex clusters.

Well-Separated Clusters

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/intclus.sas`.)

If the population clusters are sufficiently well separated, almost any clustering method performs well, as demonstrated in the following example, which uses single linkage. In this and subsequent examples, the output from the clustering procedures is not shown, but cluster membership is displayed in scatter plots. The following SAS statements produce Figure 12.1:

```
data compact;
   keep x y;
   n=50; scale=1;
   mx=0; my=0; link generate;
   mx=8; my=0; link generate;
   mx=4; my=8; link generate;
   stop;
generate:
   do i=1 to n;
      x=rannor(1)*scale+mx;
      y=rannor(1)*scale+my;
      output;
   end;
   return;
run;

proc cluster data=compact outtree=tree method=single noprint;
run;

proc tree noprint out=out n=3;
   copy x y;
run;

ods graphics on / attrpriority=none;

proc sgplot noautolegend;
   title 'Single Linkage Cluster Analysis: '
         'Well-Separated, Compact Clusters';
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.1 Well-Separated, Compact Clusters: PROC CLUSTER METHOD=SINGLE

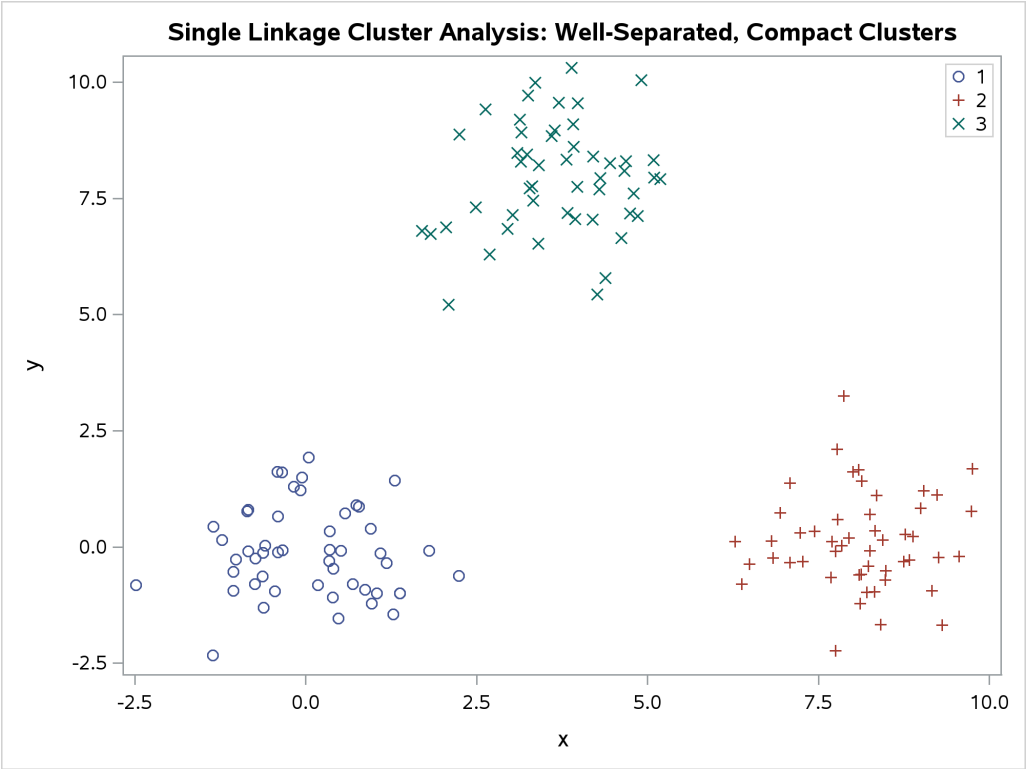

The ATTRPRIORITY=NONE option in the ODS GRAPHICS statement differentiates clusters by both marker shape and color. By default, ATTRPRIORITY=COLOR; so the default markers are circles and clusters are differentiated only by colors.

Poorly Separated Clusters

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/intclus.sas.)

To see how various clustering methods differ, you must examine a more difficult problem than that of the previous example.

The following data set is similar to the first except that the three clusters are much closer together. This example demonstrates the use of PROC FASTCLUS and five hierarchical methods available in PROC CLUSTER. To help you compare methods, this example plots true, generated clusters. Also included is a bubble plot of the density estimates obtained in conjunction with two-stage density linkage in PROC CLUSTER.

The following SAS statements produce Figure 12.2:

```
data closer;
   keep x y c;
   n=50; scale=1;
   mx=0; my=0; c=3; link generate;
```

```
   mx=3; my=0; c=1; link generate;
   mx=1; my=2; c=2; link generate;
   stop;
generate:
   do i=1 to n;
      x=rannor(9)*scale+mx;
      y=rannor(9)*scale+my;
      output;
   end;
   return;
run;

title 'True Clusters for Data Containing Poorly Separated, Compact Clusters';
proc sgplot noautolegend;
   scatter y=y x=x / group=c;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.2 Poorly Separated, Compact Clusters: Plot of True Clusters

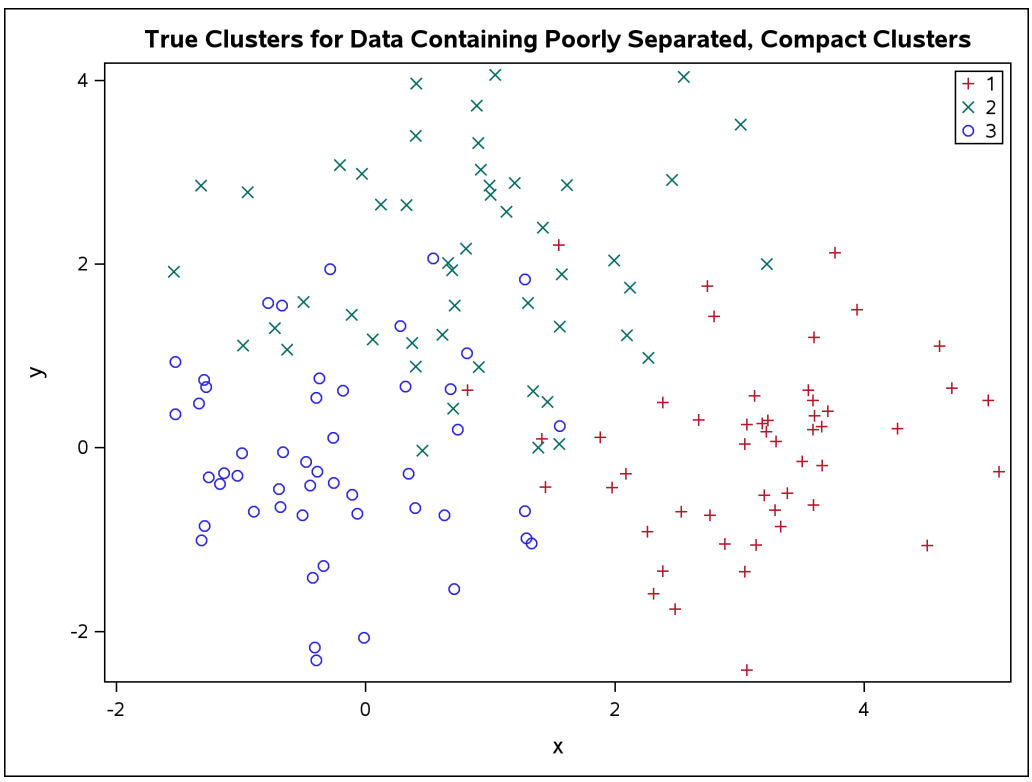

The following statements use the FASTCLUS procedure to find three clusters and then use the SGPLOT procedure to plot the clusters. The following statements produce Figure 12.3:

```
proc fastclus data=closer out=out maxc=3 noprint;
   var x y;
   title 'FASTCLUS Analysis: '
         'Poorly Separated, Compact Clusters';
run;
```

```
proc sgplot noautolegend;
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.3 Poorly Separated, Compact Clusters: PROC FASTCLUS

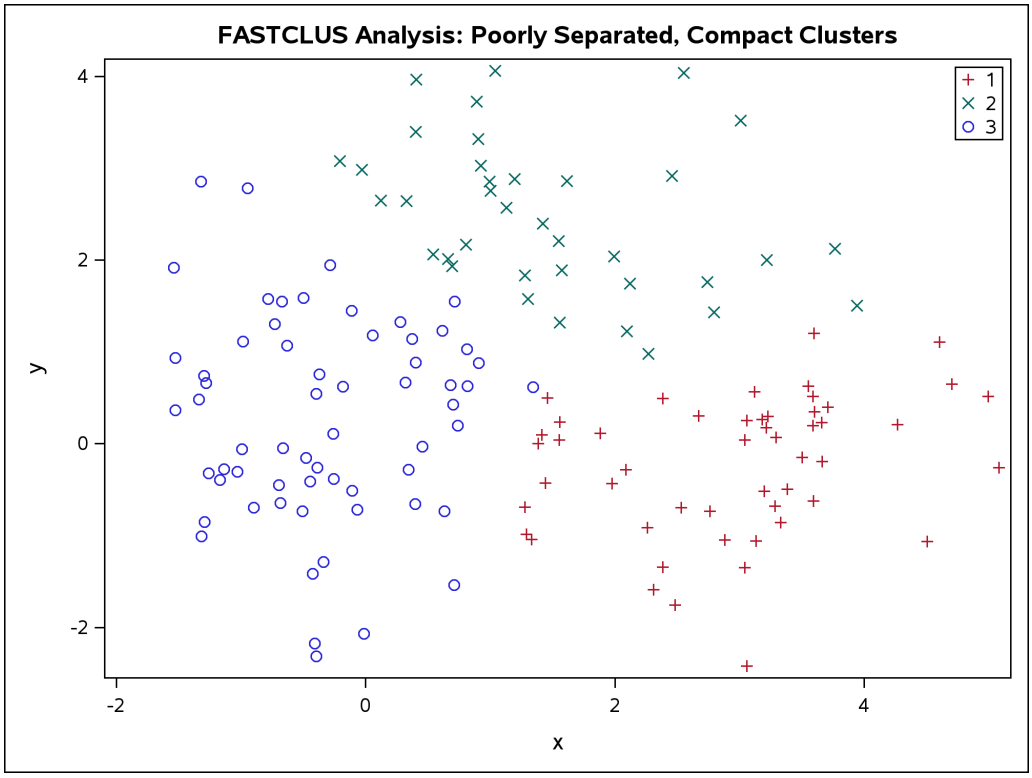

The following SAS statements produce Figure 12.4:

```
proc cluster data=closer outtree=tree method=ward noprint;
   var x y;
run;

proc tree noprint out=out n=3;
   copy x y;
   title 'Ward''s Minimum Variance Cluster Analysis: '
         'Poorly Separated, Compact Clusters';
run;

proc sgplot noautolegend;
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.4 Poorly Separated, Compact Clusters: PROC CLUSTER METHOD=WARD

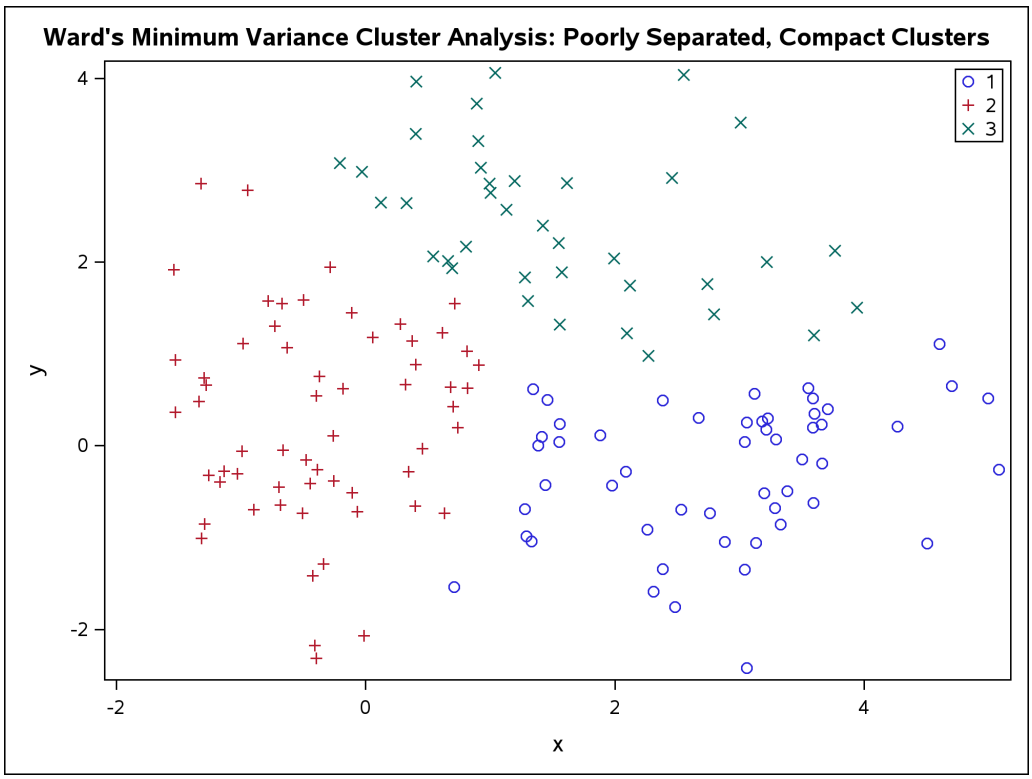

The following SAS statements produce Figure 12.5:

```
proc cluster data=closer outtree=tree method=average noprint;
   var x y;
run;

proc tree noprint out=out n=3 dock=5;
   copy x y;
   title 'Average Linkage Cluster Analysis: '
         'Poorly Separated, Compact Clusters';
run;

proc sgplot noautolegend;
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.5 Poorly Separated, Compact Clusters: PROC CLUSTER METHOD=AVERAGE

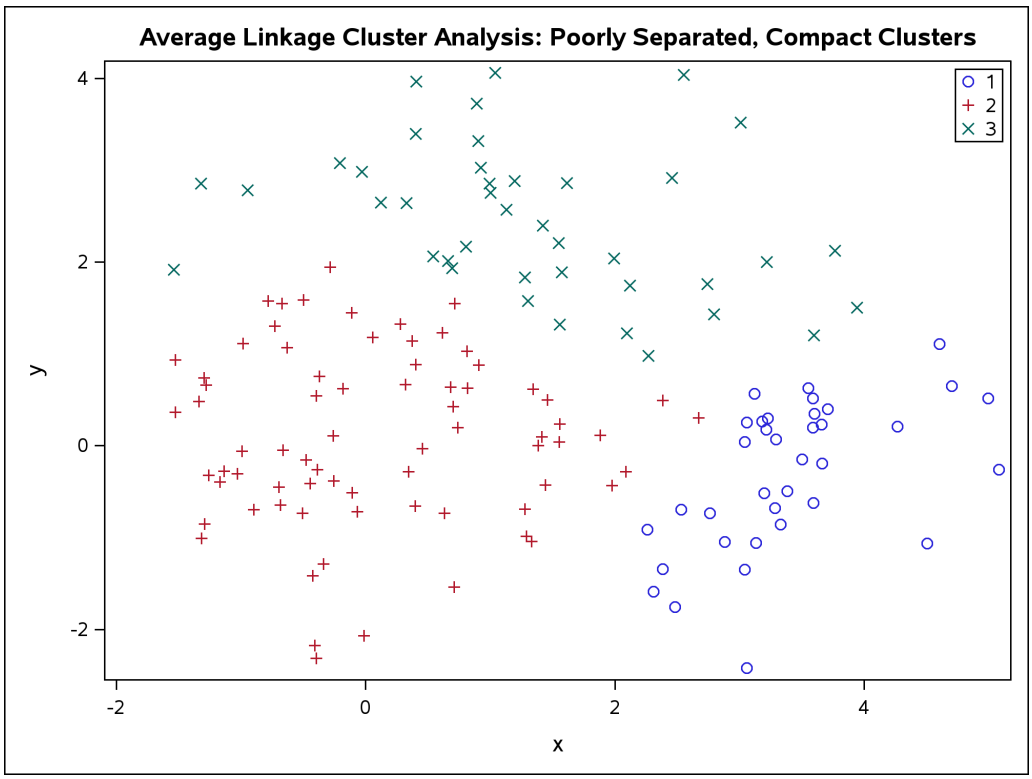

The following SAS statements produce Figure 12.6:

```
proc cluster data=closer outtree=tree method=centroid noprint;
   var x y;
run;

proc tree noprint out=out n=3 dock=5;
   copy x y;
   title 'Centroid Cluster Analysis: '
         'Poorly Separated, Compact Clusters';
run;

proc sgplot noautolegend;
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.6 Poorly Separated, Compact Clusters: PROC CLUSTER METHOD=CENTROID

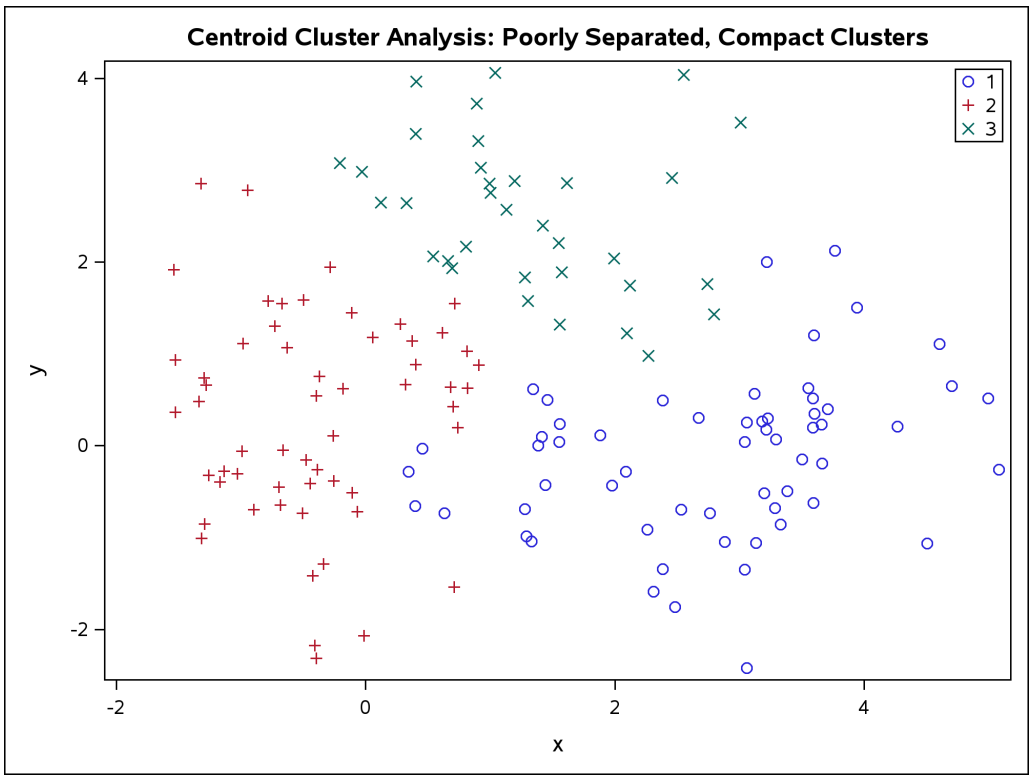

The following SAS statements produce Figure 12.7 and Figure 12.8:

```
proc cluster data=closer outtree=tree method=twostage k=10 noprint;
   var x y;
run;

proc tree noprint out=out n=3;
   copy x y _dens_;
   title 'Two-Stage Density Linkage Cluster Analysis: '
         'Poorly Separated, Compact Clusters';
run;

proc sgplot noautolegend;
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;

proc sgplot noautolegend;
   title 'Estimated Densities for Data Containing Poorly Separated, '
         'Compact Clusters';
   bubble y=y x=x size=_dens_ / nofill lineattrs=graphdatadefault;
run;
```

Figure 12.7 Poorly Separated, Compact Clusters: PROC CLUSTER METHOD=TWOSTAGE

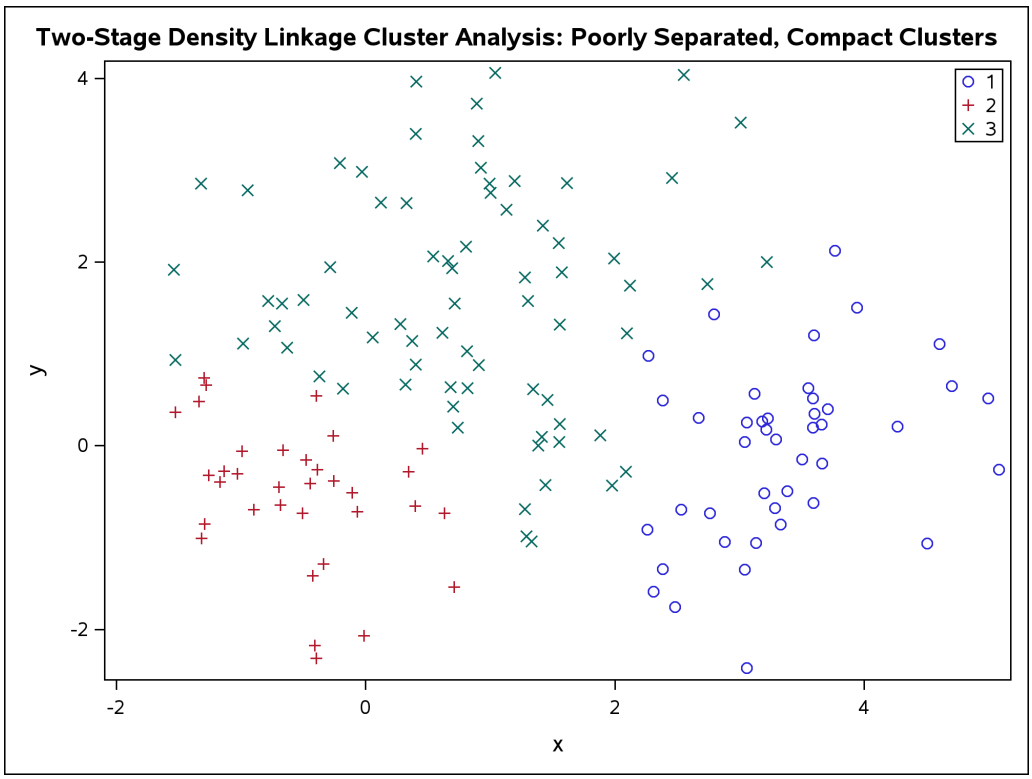

Figure 12.8 Poorly Separated, Compact Clusters: PROC CLUSTER METHOD=TWOSTAGE

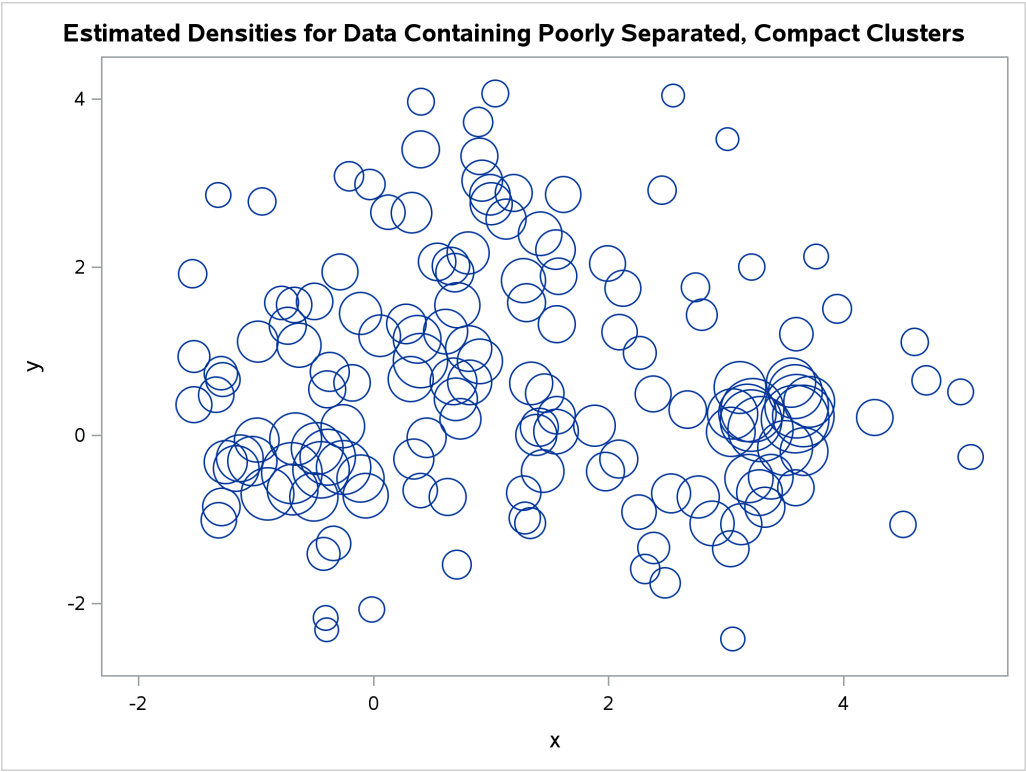

In two-stage density linkage, each cluster is a region surrounding a local maximum of the estimated probability density function. If you think of the estimated density function as a landscape with mountains and valleys, each mountain is a cluster, and the boundaries between clusters are placed near the bottoms of the valleys.

The following SAS statements produce Figure 12.9:

```
proc cluster data=closer outtree=tree method=single noprint;
   var x y;
run;

proc tree data=tree noprint out=out n=3 dock=5;
   copy x y;
   title 'Single Linkage Cluster Analysis: '
         'Poorly Separated, Compact Clusters';
run;

proc format;
   value out . = 'Outlier';
run;

proc sgplot noautolegend;
   styleattrs datasymbols=(circle plus x circlefilled);
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
   format cluster out.;
run;
```

Figure 12.9 Poorly Separated, Compact Clusters: PROC CLUSTER METHOD=SINGLE

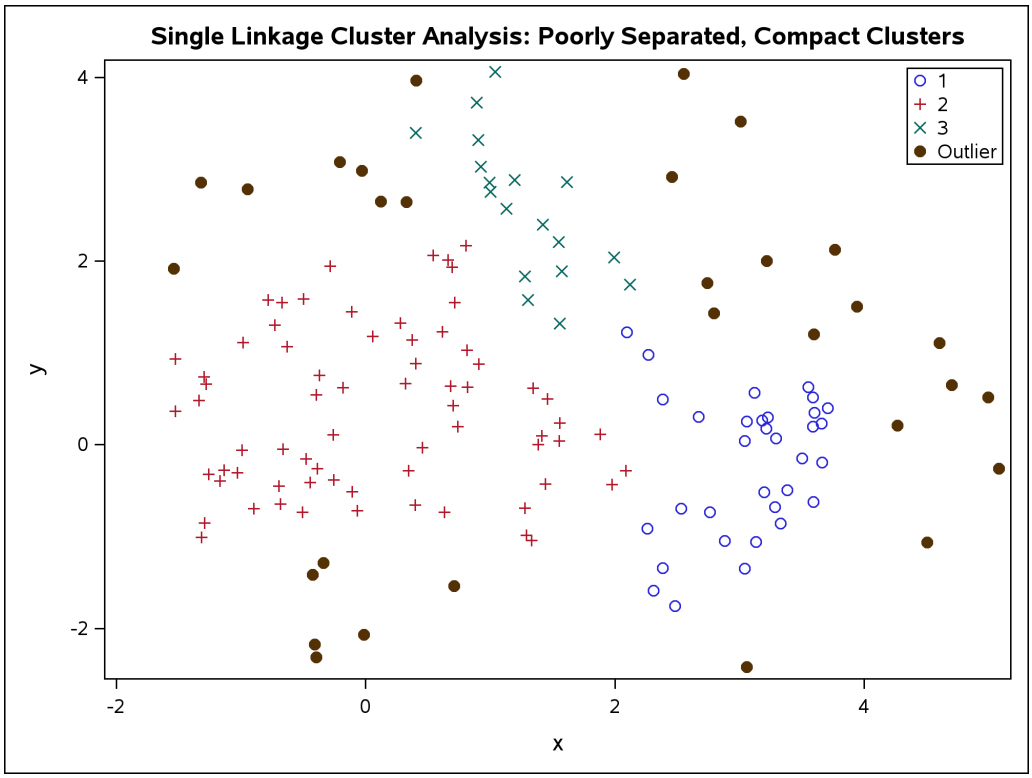

The two least squares methods, PROC FASTCLUS and Ward's, yield the most uniform cluster sizes and the best recovery of the true clusters. This result is expected since these two methods are biased toward recovering compact clusters of equal size. With average linkage, the lower-left cluster is too large; with the centroid method, the lower-right cluster is too large; and with two-stage density linkage, the top cluster is too large. The single linkage analysis resembles average linkage except for the large number of outliers resulting from the DOCK= option in the PROC TREE statement; the outliers (which are designated by missing values in the Cluster variable) are plotted as filled circles.

Multinormal Clusters of Unequal Size and Dispersion

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/intclus.sas.)

In this example, there are three multinormal clusters that differ in size and dispersion. PROC FASTCLUS and five of the hierarchical methods available in PROC CLUSTER are used. To help you compare methods, the true, generated clusters are plotted.

The following SAS statements produce Figure 12.10:

```
data unequal;
   keep x y c;
   mx=1; my=0; n=20; scale=.5; c=1; link generate;
   mx=6; my=0; n=80; scale=2.; c=3; link generate;
   mx=3; my=4; n=40; scale=1.; c=2; link generate;
   stop;
generate:
   do i=1 to n;
      x=rannor(1)*scale+mx;
      y=rannor(1)*scale+my;
      output;
   end;
   return;
run;

title 'True Clusters for Data Containing Multinormal Clusters of Unequal Size';
proc sgplot noautolegend;
   scatter y=y x=x / group=c;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.10 Generated Clusters of Unequal Size

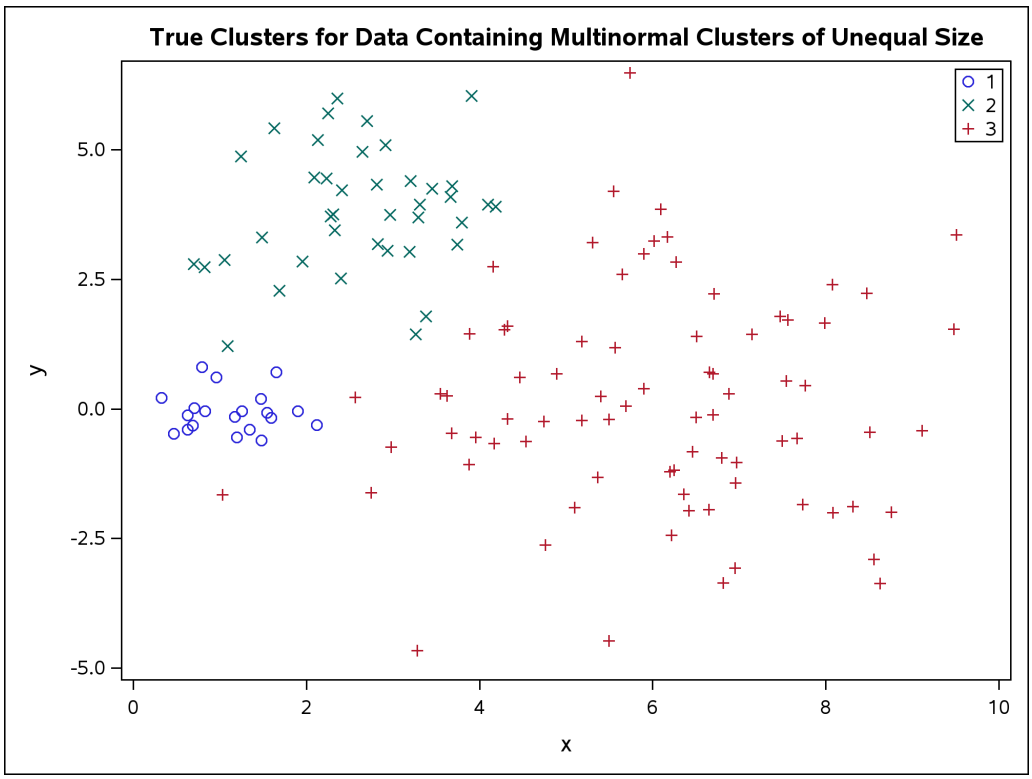

The following statements use the FASTCLUS procedure to find three clusters and then use the SGPLOT procedure to plot the clusters. The following statements produce Figure 12.11:

```
proc fastclus data=unequal out=out maxc=3 noprint;
   var x y;
   title 'FASTCLUS Analysis: Compact Clusters of Unequal Size';
run;

proc sgplot noautolegend;
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.11 Compact Clusters of Unequal Size: PROC FASTCLUS

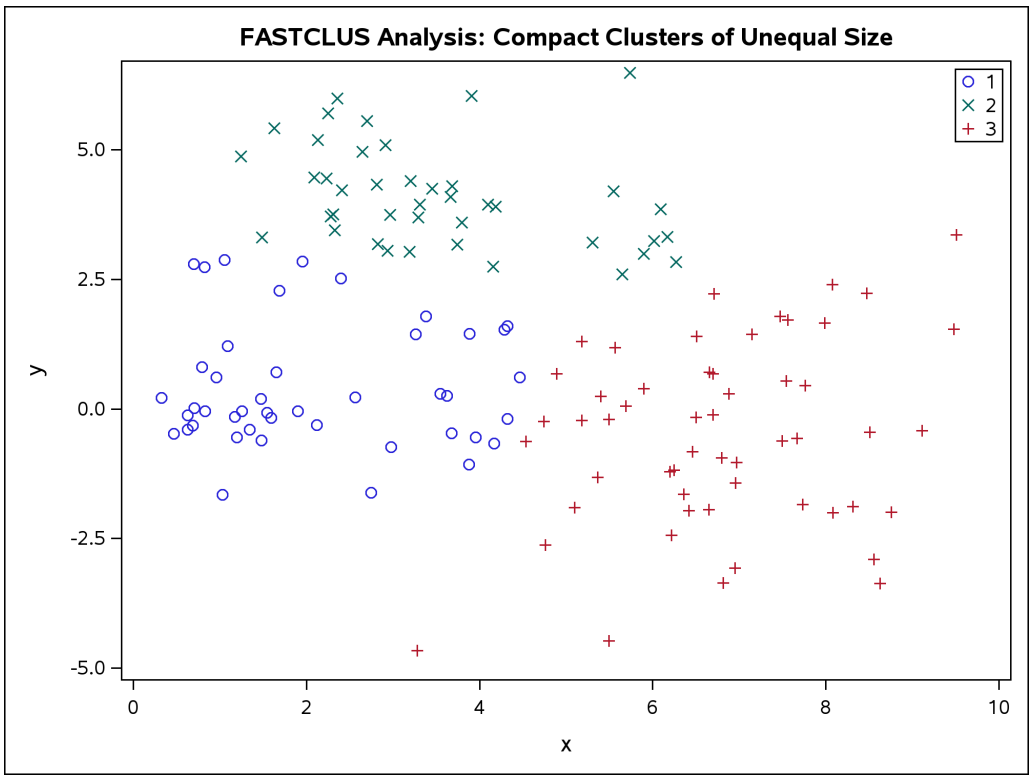

The following SAS statements produce Figure 12.12:

```
proc cluster data=unequal outtree=tree method=ward noprint;
   var x y;
run;

proc tree noprint out=out n=3;
   copy x y;
   title 'Ward''s Minimum Variance Cluster Analysis: '
         'Compact Clusters of Unequal Size';
run;

proc sgplot noautolegend;
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.12 Compact Clusters of Unequal Size: PROC CLUSTER METHOD=WARD

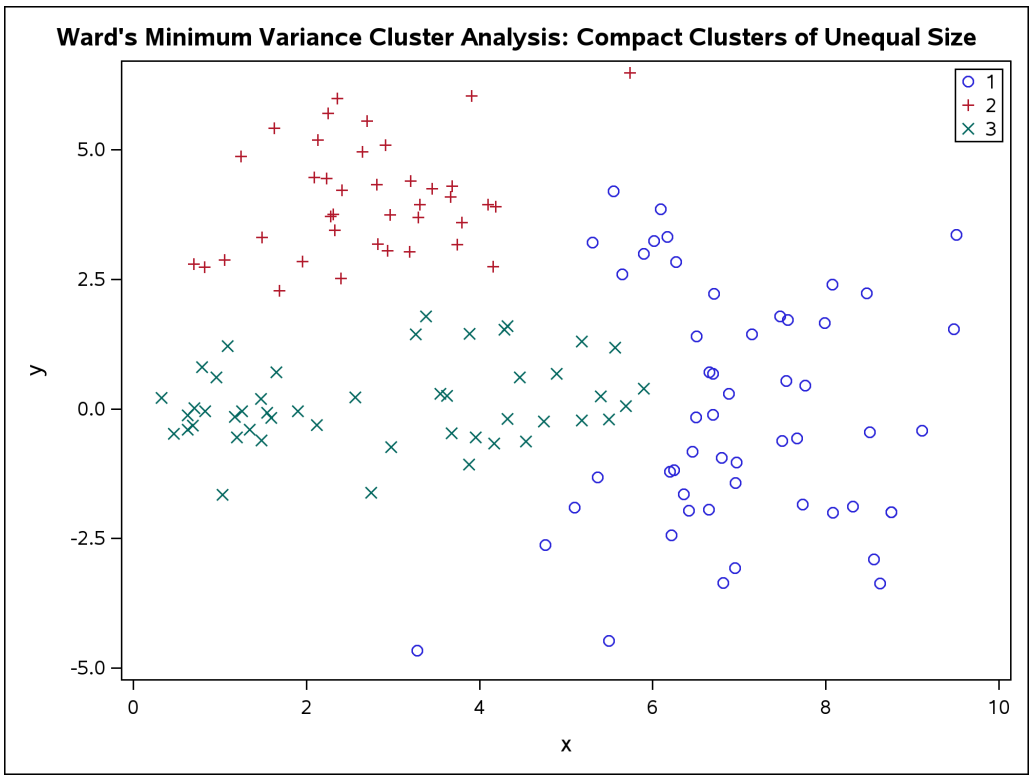

The following SAS statements produce Figure 12.13:

```
proc cluster data=unequal outtree=tree method=average noprint;
   var x y;
run;

proc tree noprint out=out n=3 dock=5;
   copy x y;
   title 'Average Linkage Cluster Analysis: '
         'Compact Clusters of Unequal Size';
run;

proc sgplot noautolegend;
   styleattrs datasymbols=(circle plus x circlefilled);
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
   format cluster out.;
run;
```

Figure 12.13 Compact Clusters of Unequal Size: PROC CLUSTER METHOD=AVERAGE

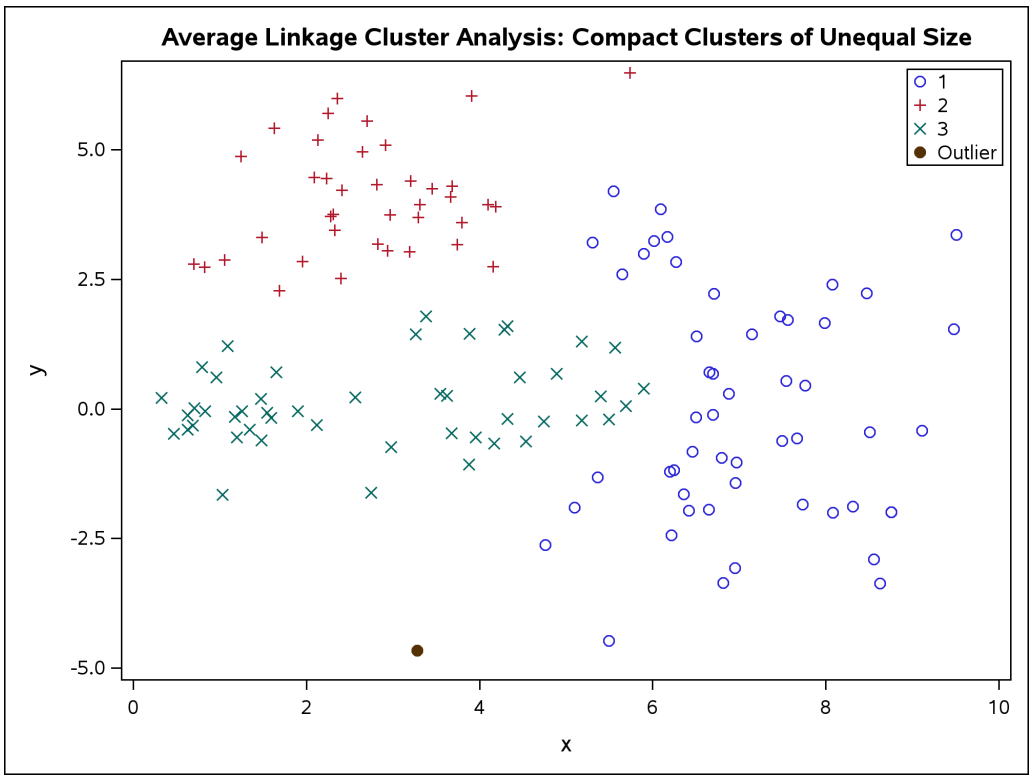

The following SAS statements produce Figure 12.14:

```
proc cluster data=unequal outtree=tree method=centroid noprint;
   var x y;
run;

proc tree noprint out=out n=3 dock=5;
   copy x y;
   title 'Centroid Cluster Analysis: '
         'Compact Clusters of Unequal Size';
run;

proc sgplot noautolegend;
   styleattrs datasymbols=(circle plus x circlefilled);
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
   format cluster out.;
run;
```

Figure 12.14 Compact Clusters of Unequal Size: PROC CLUSTER METHOD=CENTROID

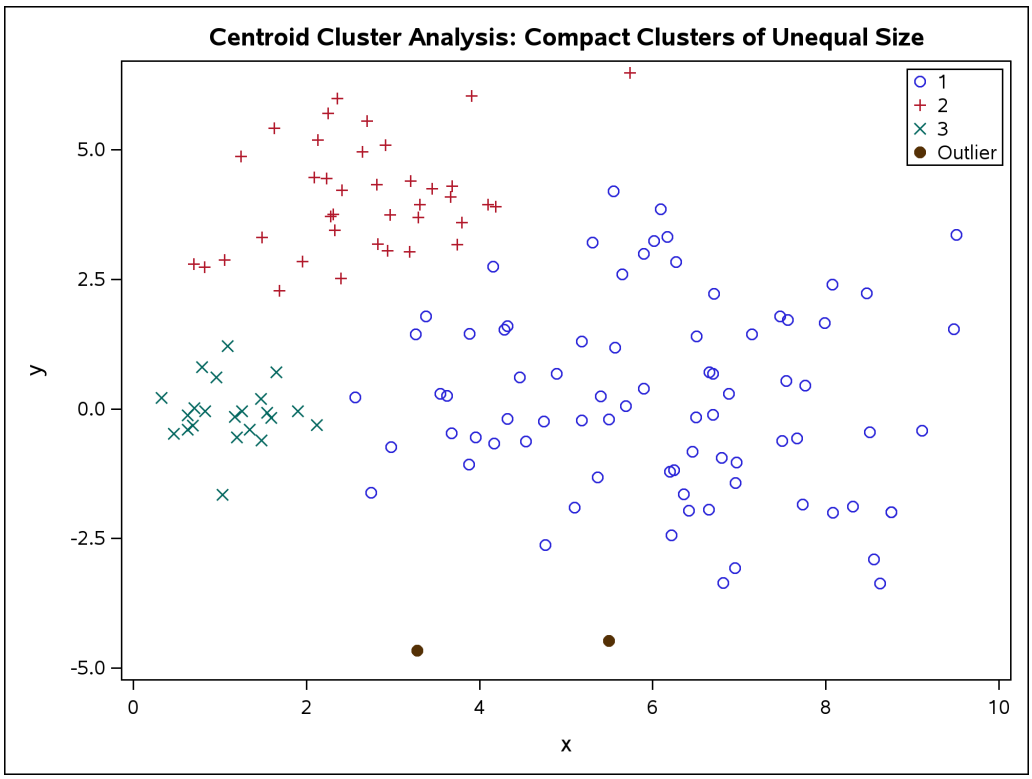

The following SAS statements produce Figure 12.15 and Figure 12.16:

```
proc cluster data=unequal outtree=tree method=twostage k=10 noprint;
   var x y;
run;

proc tree noprint out=out n=3;
   copy x y _dens_;
   title 'Two-Stage Density Linkage Cluster Analysis: '
         'Compact Clusters of Unequal Size';
run;

proc sgplot noautolegend;
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;

proc sgplot noautolegend;
   title 'Estimated Densities for Data Containing '
         'Compact Clusters of Unequal Size';
   bubble y=y x=x size=_dens_ / nofill lineattrs=graphdatadefault;
run;
```

Figure 12.15 Compact Clusters of Unequal Size: PROC CLUSTER METHOD=TWOSTAGE

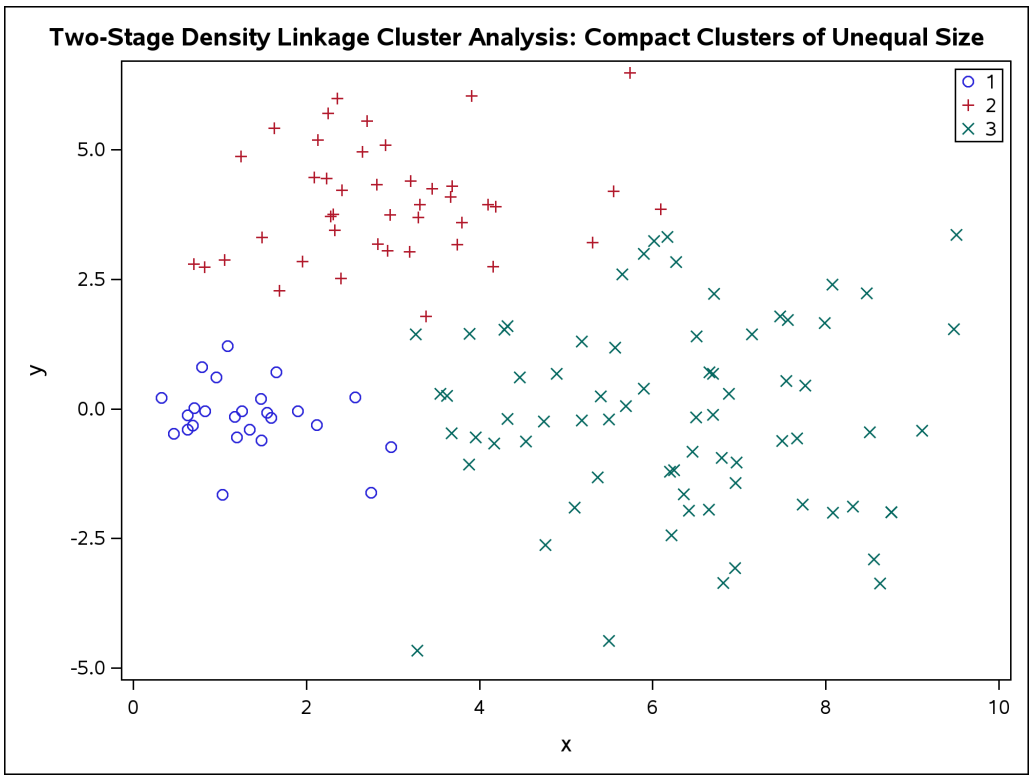

Figure 12.16 Compact Clusters of Unequal Size: PROC CLUSTER METHOD=TWOSTAGE

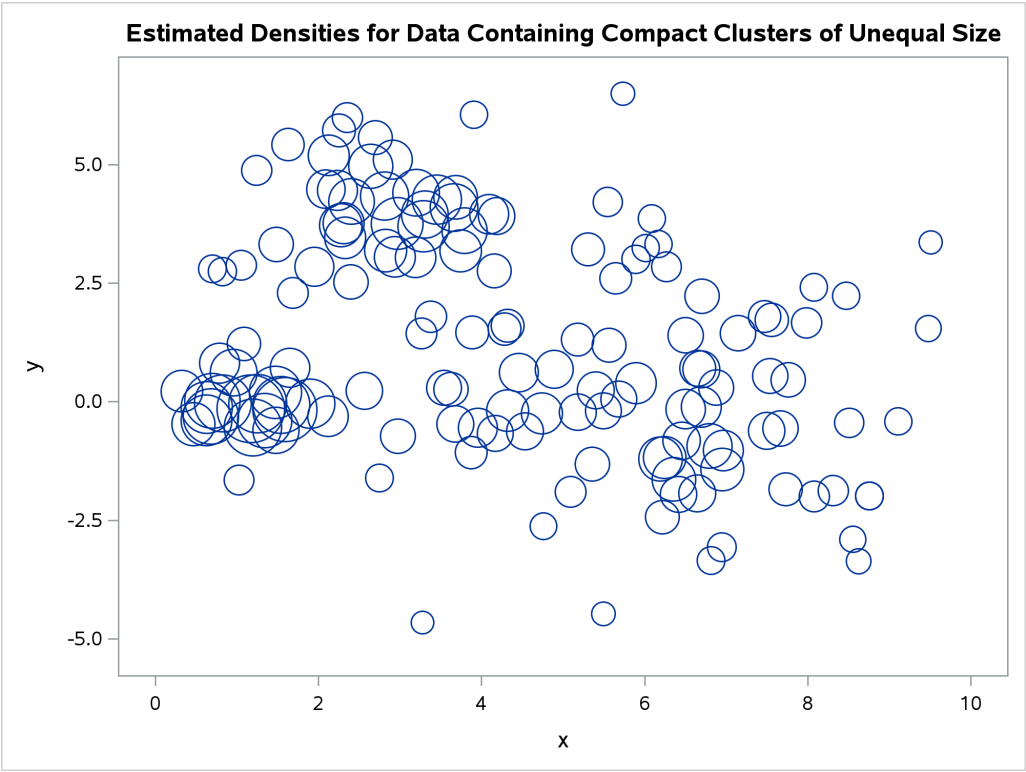

The following SAS statements produce Figure 12.17:

```
proc cluster data=unequal outtree=tree method=single noprint;
   var x y;
run;

proc tree data=tree noprint out=out n=3 dock=5;
   copy x y;
   title 'Single Linkage Cluster Analysis: '
         'Compact Clusters of Unequal Size';
run;

proc sgplot noautolegend;
   styleattrs datasymbols=(circle plus x circlefilled);
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
   format cluster out.;
run;
```

Figure 12.17 Compact Clusters of Unequal Size: PROC CLUSTER METHOD=SINGLE

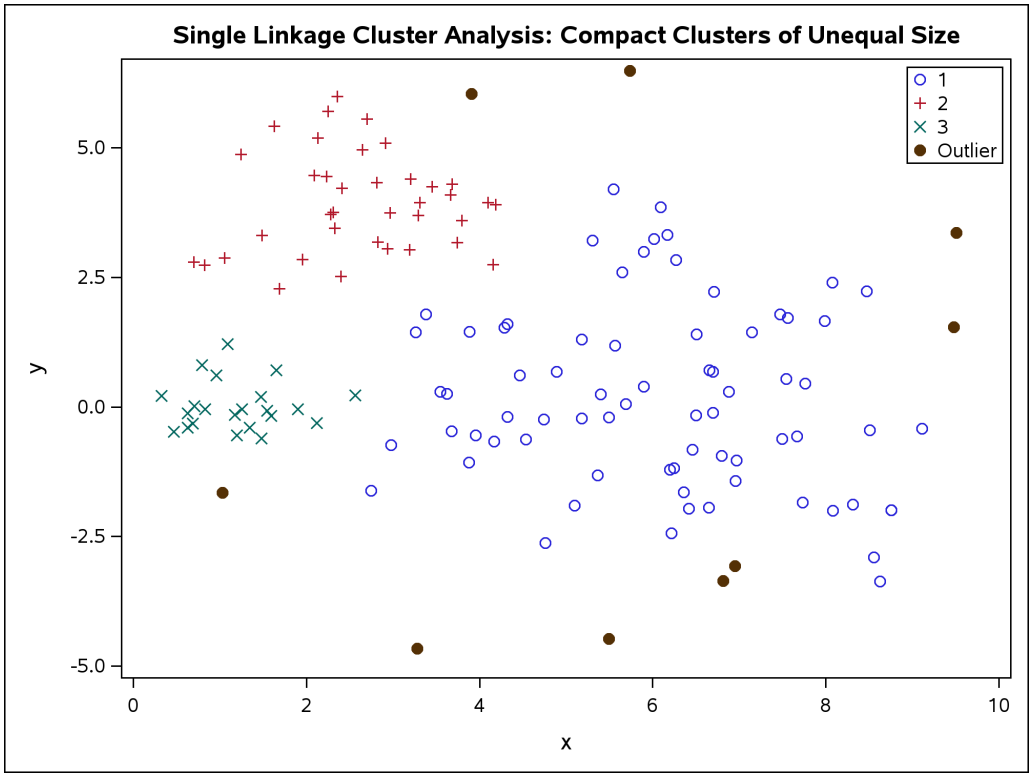

In the PROC FASTCLUS analysis, the smallest cluster, in the bottom-left portion of the plot, has stolen members from the other two clusters, and the upper-left cluster has also acquired some observations that rightfully belong to the larger, lower-right cluster. With Ward's method, the upper-left cluster is separated correctly, but the lower-left cluster has taken a large bite out of the lower-right cluster. For both of these methods, the clustering errors are in accord with the biases of the methods to produce clusters of equal size. In the average linkage analysis, both the upper-left and lower-left clusters have encroached on the lower-right

cluster, thereby making the variances more nearly equal than in the true clusters. The centroid method, which lacks the size and dispersion biases of the previous methods, obtains an essentially correct partition.

Two-stage density linkage does almost as well, even though the compact shapes of these clusters favor the traditional methods. Single linkage also produces excellent results.

Elongated Multinormal Clusters

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/intclus.sas.)

In this example, the data are sampled from two highly elongated multinormal distributions with equal covariance matrices. The following SAS statements produce Figure 12.18:

```
data elongate;
   keep x y;
   ma=8; mb=0; link generate;
   ma=6; mb=8; link generate;
   stop;
generate:
   do i=1 to 50;
      a=rannor(7)*6+ma;
      b=rannor(7)+mb;
      x=a-b;
      y=a+b;
      output;
   end;
   return;
run;

proc fastclus data=elongate out=out maxc=2 noprint;
run;

proc sgplot noautolegend;
   title 'FASTCLUS Analysis: Parallel Elongated Clusters';
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Notice that PROC FASTCLUS found two clusters, as requested by the MAXC= option. However, it attempted to form spherical clusters, which are obviously inappropriate for these data.

Figure 12.18 Parallel Elongated Clusters: PROC FASTCLUS

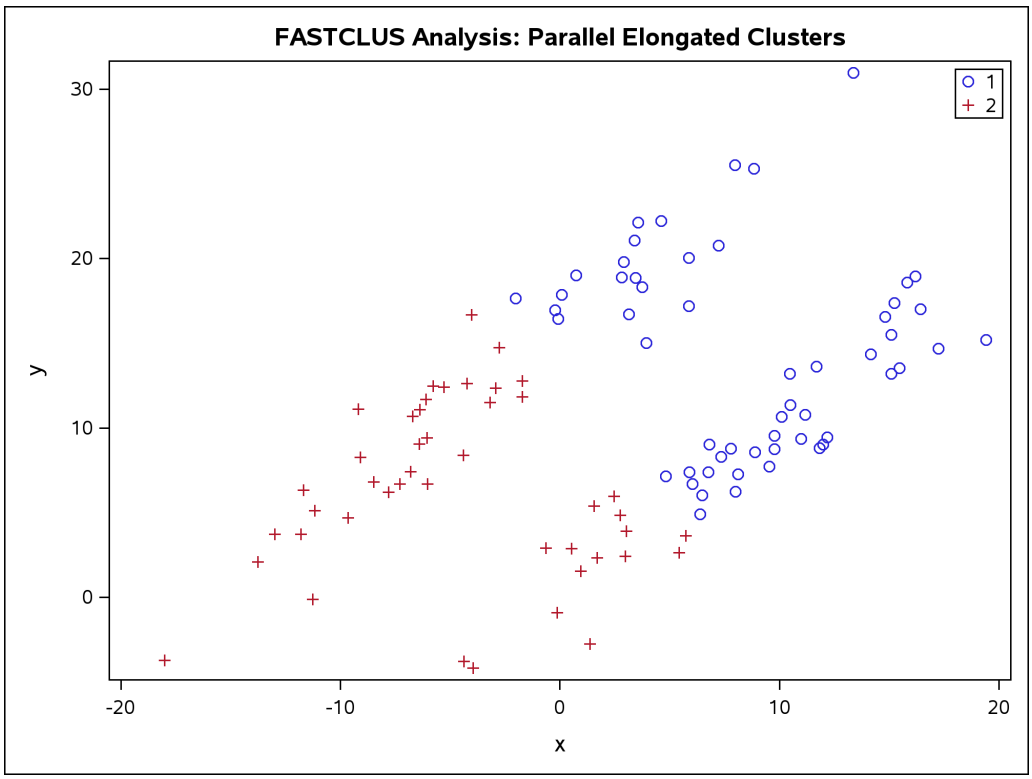

The following SAS statements produce Figure 12.19:

```
proc cluster data=elongate outtree=tree method=average noprint;
run;

proc tree noprint out=out n=2 dock=5;
   copy x y;
run;

proc sgplot noautolegend;
   title 'Average Linkage Cluster Analysis: '
         'Parallel Elongated Clusters';
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.19 Parallel Elongated Clusters: PROC CLUSTER METHOD=AVERAGE

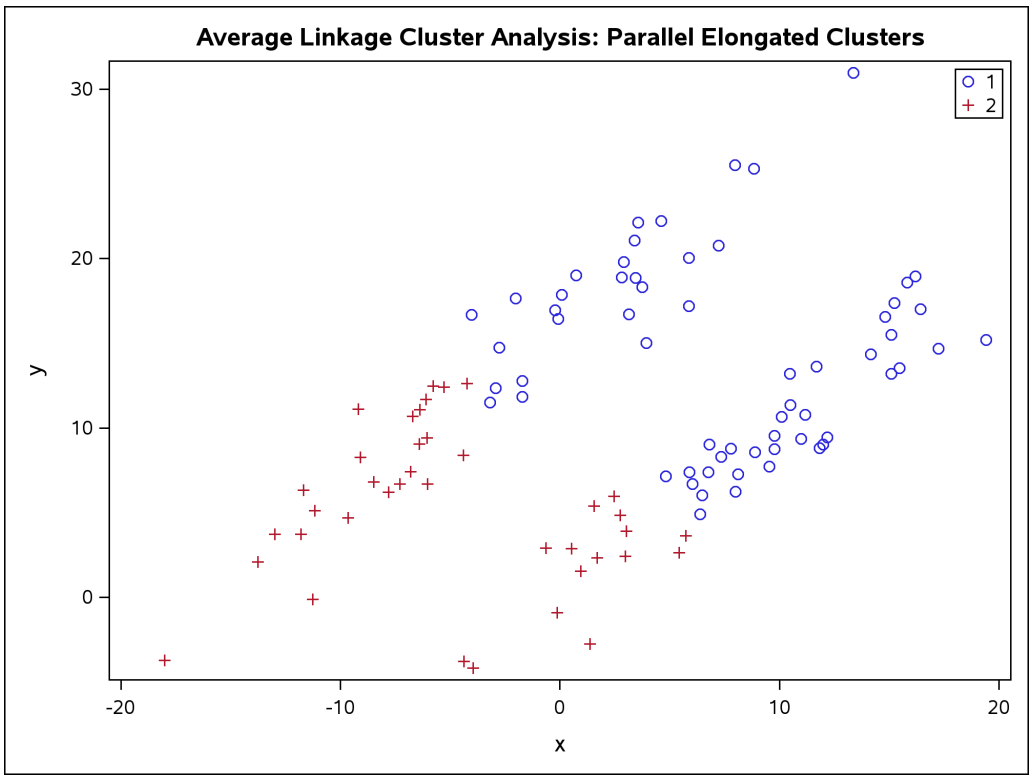

The following SAS statements produce Figure 12.20:

```
proc cluster data=elongate outtree=tree method=twostage k=10 noprint;
run;

proc tree noprint out=out n=2;
   copy x y;
run;

proc sgplot noautolegend;
   title 'Two-Stage Density Linkage Cluster Analysis: '
         'Parallel Elongated Clusters';
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.20 Parallel Elongated Clusters: PROC CLUSTER METHOD=TWOSTAGE

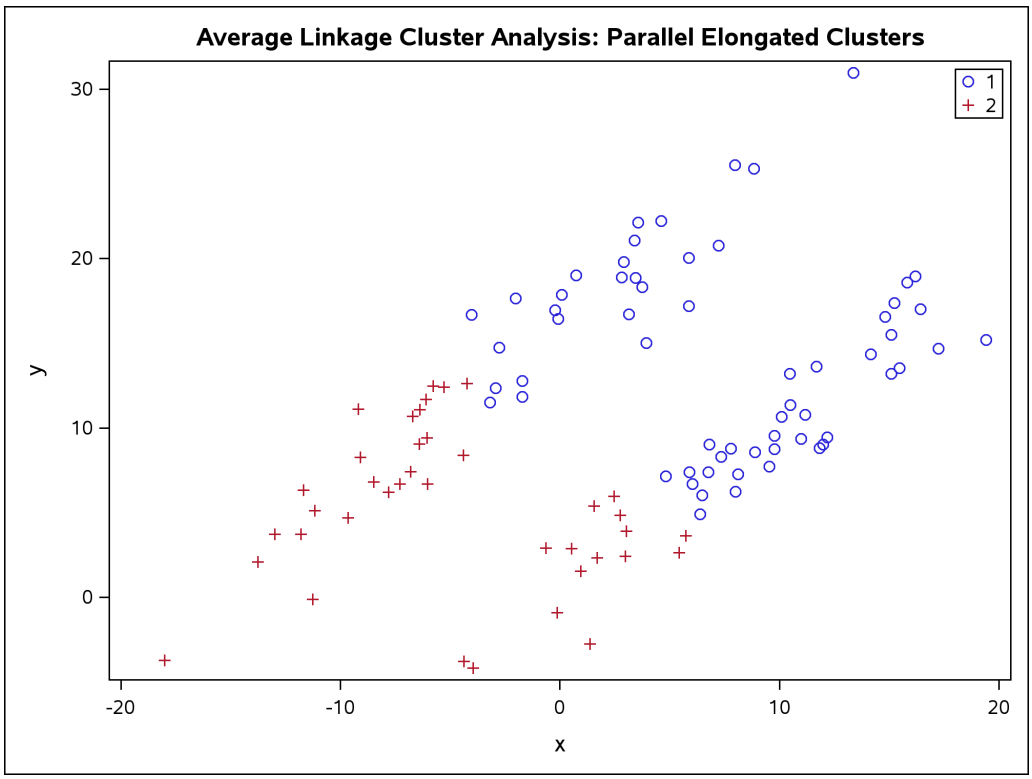

PROC FASTCLUS and average linkage fail miserably. Ward's method and the centroid method (not shown) produce almost the same results. Two-stage density linkage, however, recovers the correct clusters. Single linkage (not shown) finds the same clusters as two-stage density linkage except for some outliers.

In this example, the population clusters have equal covariance matrices. If the within-cluster covariances are known, the data can be transformed to make the clusters spherical so that any of the clustering methods can find the correct clusters. But when you are doing a cluster analysis, you do not know what the true clusters are, so you cannot calculate the within-cluster covariance matrix. Nevertheless, it is sometimes possible to estimate the within-cluster covariance matrix without knowing the cluster membership or even the number of clusters, using an approach invented by Art, Gnanadesikan, and Kettenring (1982). A method for obtaining such an estimate is available in the ACECLUS procedure.

In the following analysis, PROC ACECLUS transforms the variables X and Y into the canonical variables Can1 and Can2. The latter are plotted and then used in a cluster analysis by Ward's method. The clusters are then plotted with the original variables X and Y.

The following SAS statements produce Figure 12.21 and Figure 12.22:

```
proc aceclus data=elongate out=ace p=.1;
   var x y;
   title 'ACECLUS Analysis: Parallel Elongated Clusters';
run;

proc sgplot noautolegend;
   title 'Data Containing Parallel Elongated Clusters';
   title2 'After Transformation by PROC ACECLUS';
```

```
   scatter y=can2 x=can1;
   xaxis label='Canonical Variable 1';
   yaxis label='Canonical Variable 2';
run;
```

Figure 12.21 Parallel Elongated Clusters: PROC ACECLUS

ACECLUS Analysis: Parallel Elongated Clusters

The ACECLUS Procedure

Approximate Covariance Estimation for Cluster Analysis

Observations	100	**Proportion**	0.1000
Variables	2	**Converge**	0.00100

Means and Standard Deviations

Variable	Mean	Standard Deviation
x	2.6406	8.3494
y	10.6488	6.8420

COV: Total Sample Covariances

	x	y
x	69.71314819	24.24268934
y	24.24268934	46.81324861

Initial Within-Cluster Covariance Estimate = Full Covariance Matrix

Threshold = 0.328478

Iteration History

Iteration	RMS Distance	Distance Cutoff	Pairs Within Cutoff	Convergence Measure
1	2.000	0.657	672.0	0.673685
2	9.382	3.082	716.0	0.006963
3	9.339	3.068	760.0	0.008362
4	9.437	3.100	824.0	0.009656
5	9.359	3.074	889.0	0.010269
6	9.267	3.044	955.0	0.011276
7	9.208	3.025	999.0	0.009230
8	9.230	3.032	1052.0	0.011394
9	9.226	3.030	1091.0	0.007924
10	9.173	3.013	1121.0	0.007993

WARNING: Iteration limit exceeded.

Figure 12.21 *continued*

ACE: Approximate Covariance Estimate Within Clusters	x	y
x	9.299329632	8.215362614
y	8.215362614	8.937753936

Eigenvalues of Inv(ACE)*(COV-ACE)	Eigenvalue	Difference	Proportion	Cumulative
1	36.7091	33.1672	0.9120	0.9120
2	3.5420		0.0880	1.0000

Eigenvectors (Raw Canonical Coefficients)	Can1	Can2
x	-.748392	0.109547
y	0.736349	0.230272

Standardized Canonical Coefficients	Can1	Can2
x	-6.24866	0.91466
y	5.03812	1.57553

Figure 12.22 Parallel Elongated Clusters after Transformation by PROC ACECLUS

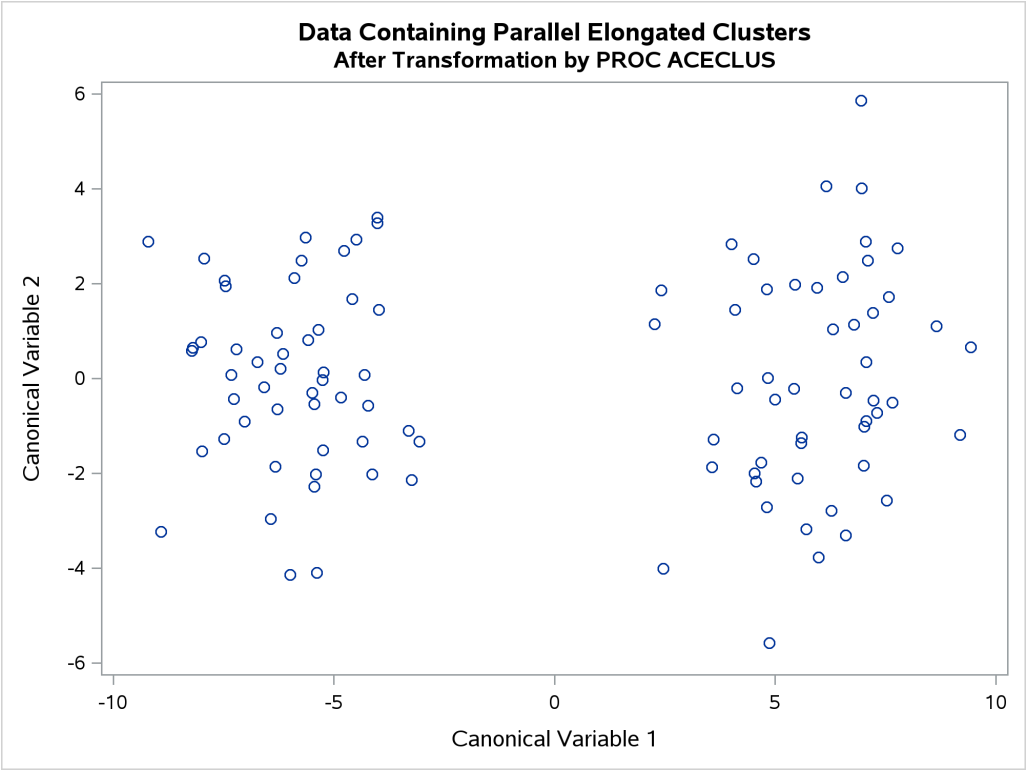

The following SAS statements produce Figure 12.23:

```
proc cluster data=ace outtree=tree method=ward noprint;
   var can1 can2;
   copy x y;
run;

proc tree noprint out=out n=2;
   copy x y;
run;

proc sgplot noautolegend;
   title 'Ward''s Minimum Variance Cluster Analysis: '
         'Parallel Elongated Clusters';
   title2 'After Transformation by PROC ACECLUS';
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.23 Transformed Data Containing Parallel Elongated Clusters: PROC CLUSTER METHOD=WARD

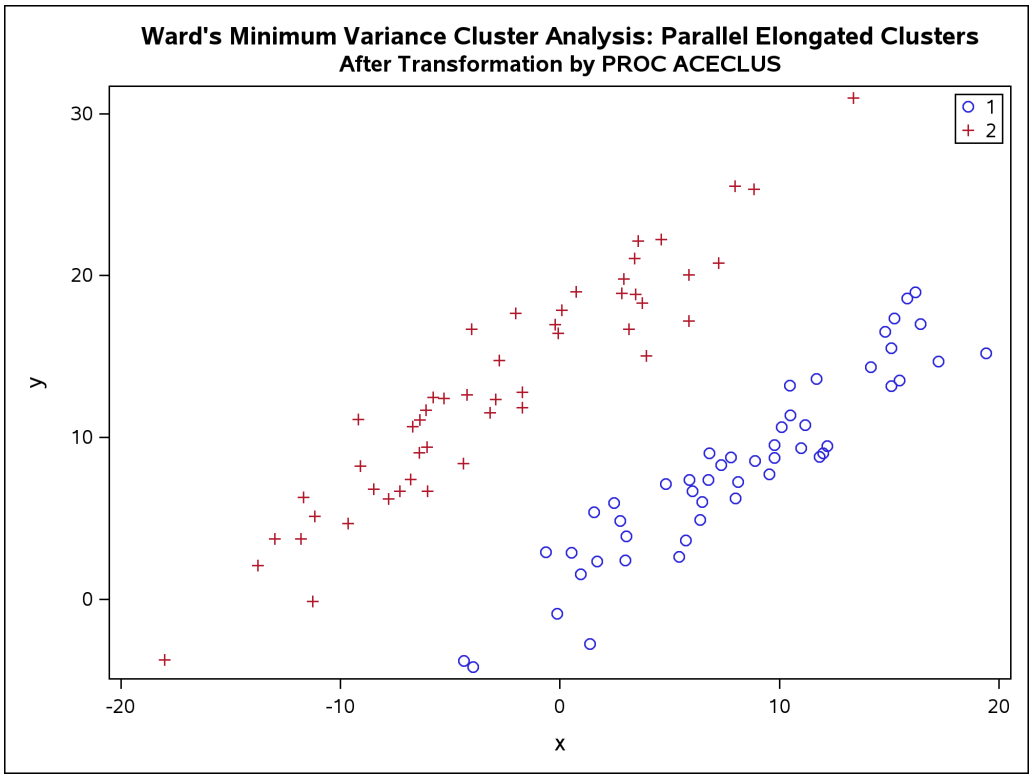

Nonconvex Clusters

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/intclus.sas.)

If the population clusters have very different covariance matrices, using PROC ACECLUS is of no avail. Although methods exist for estimating multinormal clusters with unequal covariance matrices (Wolfe 1970; Symons 1981; Everitt and Hand 1981; Titterington, Smith, and Makov 1985; McLachlan and Basford 1988), these methods tend to have serious problems with initialization and might converge to degenerate solutions. For unequal covariance matrices or radically nonnormal distributions, the best approach to cluster analysis is through nonparametric density estimation, as in density linkage. The next example illustrates population clusters with nonconvex density contours. The following SAS statements produce Figure 12.24:

```
data noncon;
   keep x y;
   do i=1 to 100;
      a=i*.0628319;
      x=cos(a)+(i>50)+rannor(7)*.1;
      y=sin(a)+(i>50)*.3+rannor(7)*.1;
      output;
   end;
run;

proc fastclus data=noncon out=out maxc=2 noprint;
run;

proc sgplot noautolegend;
   title 'FASTCLUS Analysis: Nonconvex Clusters';
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.24 Nonconvex Clusters: PROC FASTCLUS

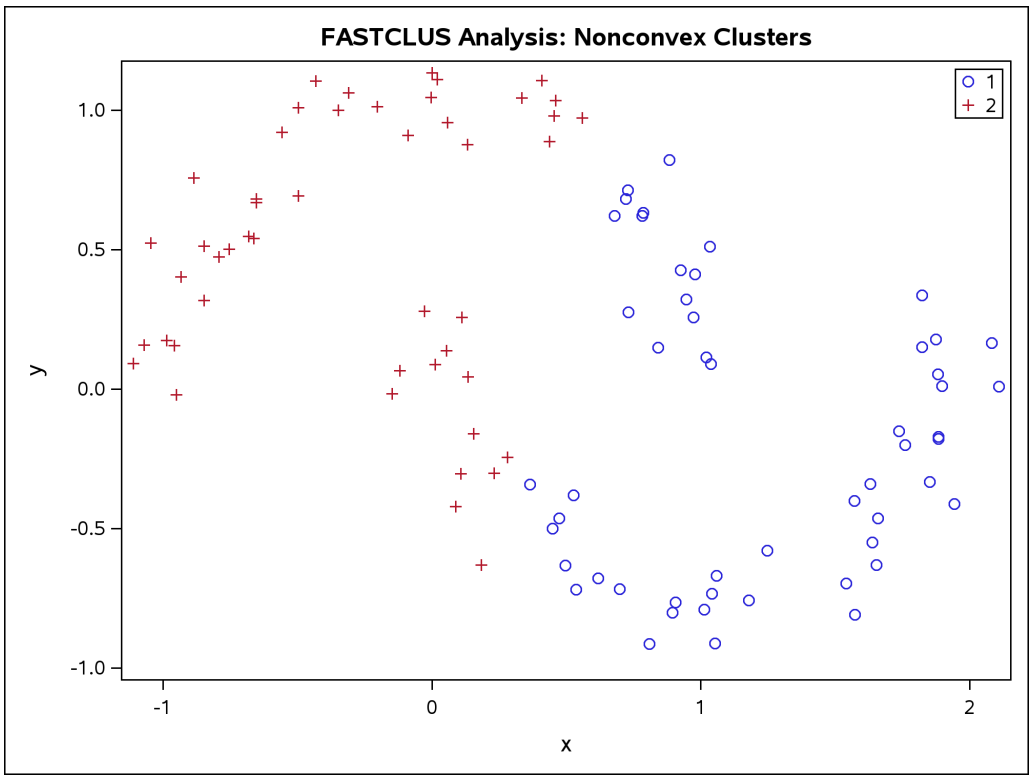

The following SAS statements produce Figure 12.25:

```
proc cluster data=noncon outtree=tree method=centroid noprint;
run;

proc tree noprint out=out n=2 dock=5;
   copy x y;
run;

proc sgplot noautolegend;
   title 'Centroid Cluster Analysis: Nonconvex Clusters';
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.25 Nonconvex Clusters: PROC CLUSTER METHOD=CENTROID

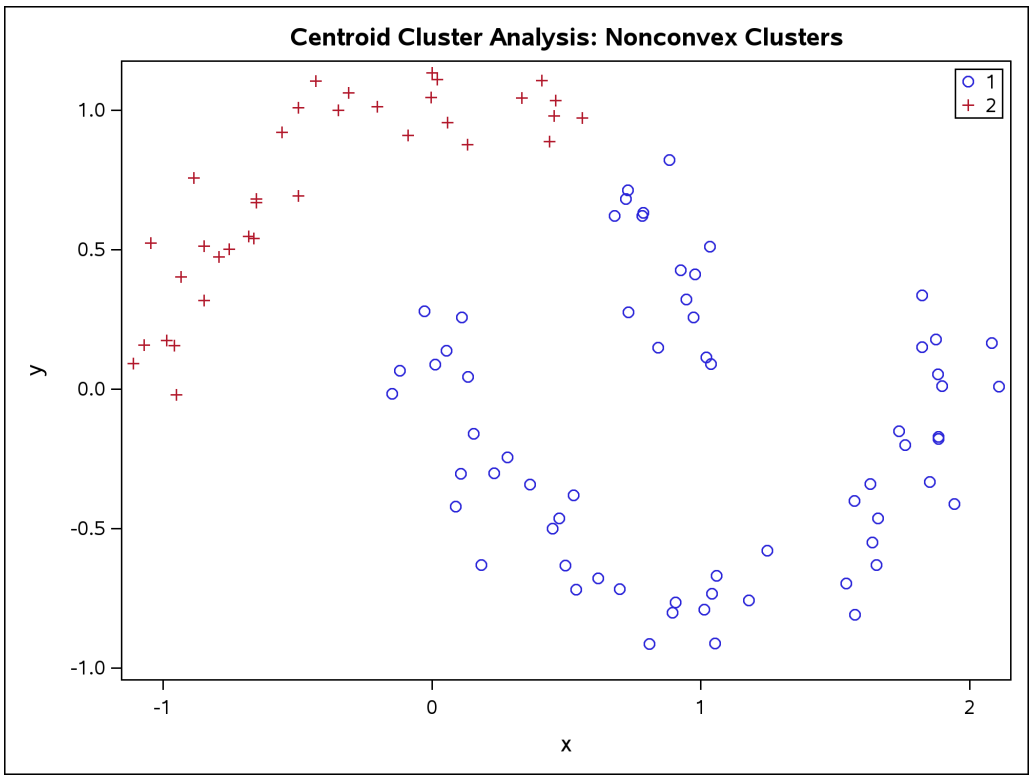

The following SAS statements produce Figure 12.26:

```
proc cluster data=noncon outtree=tree method=twostage k=10 noprint;
run;

proc tree noprint out=out n=2;
   copy x y;
run;

proc sgplot noautolegend;
   title 'Two-Stage Density Linkage Cluster Analysis: Nonconvex Clusters';
   scatter y=y x=x / group=cluster;
   keylegend / location=inside position=topright sortorder=ascending
               across=1 noopaque title='';
run;
```

Figure 12.26 Nonconvex Clusters: PROC CLUSTER METHOD=TWOSTAGE

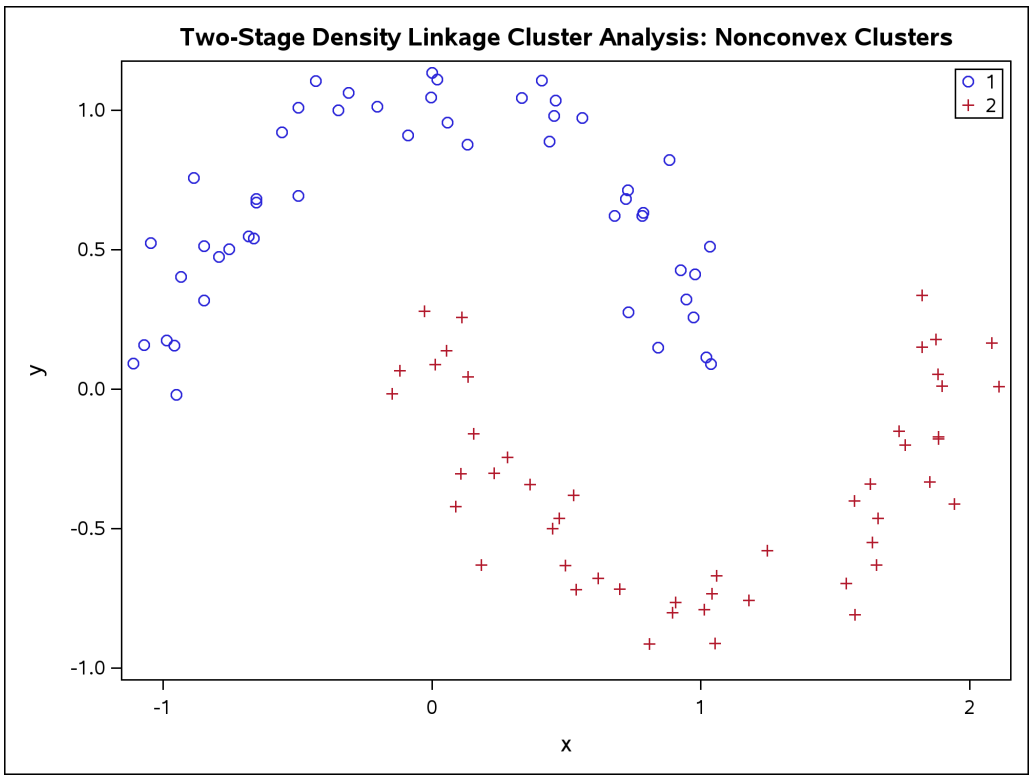

Ward's method and average linkage (not shown) do better than PROC FASTCLUS but not as well as the centroid method. Two-stage density linkage recovers the correct clusters, as does single linkage (not shown).

The preceding examples are intended merely to illustrate some of the properties of clustering methods in common use. If you intend to perform a cluster analysis, you should consult more systematic and rigorous studies of the properties of clustering methods, such as Milligan (1980).

The Number of Clusters

There are no completely satisfactory methods that can be used for determining the number of population clusters for any type of cluster analysis (Everitt 1979; Hartigan 1985a; Bock 1985).

If your purpose in clustering is dissection—that is, to summarize the data without trying to uncover real clusters—it might suffice to look at R square for each variable and pooled over all variables. Plots of R square against the number of clusters are useful.

It is always a good idea to look at your data graphically. If you have only two or three variables, use PROC SGPLOT to make scatter plots identifying the clusters. With more variables, use PROC CANDISC to compute canonical variables for plotting.

Ordinary significance tests, such as analysis of variance F tests, are not valid for testing differences between clusters. Since clustering methods attempt to maximize the separation between clusters, the assumptions of the usual significance tests, parametric or nonparametric, are drastically violated. For example, if you take a

sample of 100 observations from a single univariate normal distribution, have PROC FASTCLUS divide it into two clusters, and run a t test between the clusters, you usually obtain a p-value of less than 0.0001. For the same reason, methods that purport to test for clusters against the null hypothesis that objects are assigned randomly to clusters (such as McClain and Rao 1975 and Klastorin 1983) are useless.

Most valid tests for clusters either have intractable sampling distributions or involve null hypotheses for which rejection is uninformative. For clustering methods based on distance matrices, a popular null hypothesis is that all permutations of the values in the distance matrix are equally likely (Ling 1973; Hubert 1974). Using this null hypothesis, you can do a permutation test or a rank test. The trouble with the permutation hypothesis is that, with any real data, the null hypothesis is implausible even if the data do not contain clusters. Rejecting the null hypothesis does not provide any useful information (Hubert and Baker 1977).

Another common null hypothesis is that the data are a random sample from a multivariate normal distribution (Wolfe 1970, 1978; Duda and Hart 1973; Lee 1979). The multivariate normal null hypothesis arises naturally in normal mixture models (Titterington, Smith, and Makov 1985; McLachlan and Basford 1988). Unfortunately, the likelihood ratio test statistic does not have the usual asymptotic χ^2 distribution because the regularity conditions do not hold. Approximations to the asymptotic distribution of the likelihood ratio have been suggested Wolfe (1978), but the adequacy of these approximations is debatable (Everitt 1981; Thode, Mendell, and Finch 1988). For small samples, bootstrapping seems preferable (McLachlan and Basford 1988). Bayesian inference provides a promising alternative to likelihood ratio tests for the number of mixture components for both normal mixtures and other types of distributions (Binder 1978, 1981; Banfield and Raftery 1993; Bensmail et al. 1997).

The multivariate normal null hypothesis is better than the permutation null hypothesis, but it is not satisfactory because there is typically a high probability of rejection if the data are sampled from a distribution with lower kurtosis than a normal distribution, such as a uniform distribution. The tables in Englemann and Hartigan (1969), for example, generally lead to rejection of the null hypothesis when the data are sampled from a uniform distribution. Hawkins, Muller, and ten Krooden (1982, pp. 337–340) discuss a highly conservative Bonferroni method for the use of hypothesis testing. The conservativeness of this approach might compensate to some extent for the liberalness exhibited by tests based on normal distributions when the population is uniform.

Perhaps a better null hypothesis is that the data are sampled from a uniform distribution (Hartigan 1978; Arnold 1979; Sarle 1983) The uniform null hypothesis leads to conservative error rates when the data are sampled from a strongly unimodal distribution such as the normal. However, in two or more dimensions and depending on the test statistic, the results can be very sensitive to the shape of the region of support of the uniform distribution. Sarle (1983) suggests using a hyperbox with sides proportional in length to the singular values of the centered coordinate matrix.

Given that the uniform distribution provides an appropriate null hypothesis, there are still serious difficulties in obtaining sampling distributions. Some asymptotic results are available (Hartigan 1978, 1985a; Pollard 1981; Bock 1985) for the within-cluster sum of squares, the criterion that PROC FASTCLUS and Ward's minimum variance method attempt to optimize. No distributional theory for finite sample sizes has yet appeared. Currently, the only practical way to obtain sampling distributions for realistic sample sizes is by computer simulation.

Arnold (1979) used simulation to derive tables of the distribution of a criterion based on the determinant of the within-cluster sum of squares matrix $|\mathbf{W}|$. Both normal and uniform null distributions were used. Having obtained clusters with either PROC FASTCLUS or PROC CLUSTER, you can compute Arnold's criterion with the ANOVA or CANDISC procedure. Arnold's tables provide a conservative test because

PROC FASTCLUS and PROC CLUSTER attempt to minimize the trace of **W** rather than the determinant. Marriott (1971, 1975) also provides useful information about $|\mathbf{W}|$ as a criterion for the number of clusters.

Sarle (1983) used extensive simulations to develop the cubic clustering criterion (CCC), which can be used for crude hypothesis testing and estimating the number of population clusters. The CCC is based on the assumption that a uniform distribution on a hyperrectangle will be divided into clusters shaped roughly like hypercubes. In large samples that can be divided into the appropriate number of hypercubes, this assumption gives very accurate results. In other cases the approximation is generally conservative. For details about the interpretation of the CCC, consult Sarle (1983).

Milligan and Cooper (1985) and Cooper and Milligan (1988) compared 30 methods of estimating the number of population clusters by using four hierarchical clustering methods. The three criteria that performed best in these simulation studies with a high degree of error in the data were a pseudo F statistic developed by Caliński and Harabasz (1974), a statistic referred to as $J_e(2)/J_e(1)$ by Duda and Hart (1973) that can be transformed into a pseudo t^2 statistic, and the cubic clustering criterion. The pseudo F statistic and the CCC are displayed by PROC FASTCLUS; these two statistics and the pseudo t^2 statistic, which can be applied only to hierarchical methods, are displayed by PROC CLUSTER. It might be advisable to look for consensus among the three statistics—that is, local peaks of the CCC and pseudo F statistic combined with a small value of the pseudo t^2 statistic and a larger pseudo t^2 for the next cluster fusion. It must be emphasized that these criteria are appropriate only for compact or slightly elongated clusters, preferably clusters that are roughly multivariate normal.

Recent research has tended to deemphasize mixture models in favor of nonparametric models in which clusters correspond to modes in the probability density function. Hartigan and Hartigan (1985) and Hartigan (1985b) developed a test of unimodality versus bimodality in the univariate case.

Nonparametric tests for the number of clusters can also be based on nonparametric density estimates. This approach requires much weaker assumptions than mixture models, namely, that the observations are sampled independently and that the distribution can be estimated nonparametrically. Silverman (1986) describes a bootstrap test for the number of modes using a Gaussian kernel density estimate, but problems have been reported with this method under the uniform null distribution. Further developments in nonparametric methods are given by Müller and Sawitzki (1991); Minnotte (1992); Polonik (1993). All of these methods suffer from heavy computational requirements.

One useful descriptive approach to the number-of-clusters problem is provided by Wong and Schaack (1982) based on a kth-nearest-neighbor density estimate. The kth-nearest-neighbor clustering method developed by Wong and Lane (1983) is applied with varying values of k. Each value of k yields an estimate of the number of modal clusters. If the estimated number of modal clusters is constant for a wide range of k values, there is strong evidence of at least that many modes in the population. A plot of the estimated number of modes against k can be highly informative. Attempts to derive a formal hypothesis test from this diagnostic plot have met with difficulties, but a simulation approach similar to Silverman (1986) does seem to work Girman (1994). The simulation, of course, requires considerable computer time.

PROC MODECLUS uses a less expensive approximate nonparametric test for the number of clusters. This test sacrifices statistical efficiency for computational efficiency. The method for conducting significance tests is described in the chapter on the MODECLUS procedure. This method has the following useful features:

- No distributional assumptions are required.
- The choice of smoothing parameter is not critical since you can try any number of different values.
- The data can be coordinates or distances.

- Time and space requirements for the significance tests are no worse than those for obtaining the clusters.
- The power is high enough to be useful for practical purposes.

The method for computing the p-values is based on a series of plausible approximations. There are as yet no rigorous proofs that the method is infallible. Neither are there any asymptotic results. However, simulations for sample sizes ranging from 20 to 2000 indicate that the p-values are almost always conservative. The only case discovered so far in which the p-values are liberal is a uniform distribution in one dimension for which the simulated error rates exceed the nominal significance level only slightly for a limited range of sample sizes.

References

Anderberg, M. R. (1973). *Cluster Analysis for Applications*. New York: Academic Press.

Arnold, S. J. (1979). "A Test for Clusters." *Journal of Marketing Research* 16:545–551.

Art, D., Gnanadesikan, R., and Kettenring, J. R. (1982). "Data-Based Metrics for Cluster Analysis." *Utilitas Mathematica* 75–99.

Banfield, J. D., and Raftery, A. E. (1993). "Model-Based Gaussian and Non-Gaussian Clustering." *Biometrics* 49:803–821.

Bensmail, H., Celeux, G., Raftery, A. E., and Robert, C. P. (1997). "Inference in Model-Based Cluster Analysis." *Statistics and Computing* 7:1–10.

Bezdek, J. C. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum.

Bezdek, J. C., and Pal, S. K. (1992). *Fuzzy Models for Pattern Recognition*. New York: IEEE Press.

Binder, D. A. (1978). "Bayesian Cluster Analysis." *Biometrika* 65:31–38.

Binder, D. A. (1981). "Approximations to Bayesian Clustering Rules." *Biometrika* 68:275–285.

Blashfield, R. K., and Aldenderfer, M. S. (1978). "The Literature on Cluster Analysis." *Multivariate Behavioral Research* 13:271–295.

Bock, H. H. (1985). "On Some Significance Tests in Cluster Analysis." *Journal of Classification* 2:77–108.

Caliński, T., and Harabasz, J. (1974). "A Dendrite Method for Cluster Analysis." *Communications in Statistics—Theory and Methods* 3:1–27.

Cooper, M. C., and Milligan, G. W. (1988). "The Effect of Error on Determining the Number of Clusters." In *Proceedings of the International Workshop on Data Analysis, Decision Support, and Expert Knowledge Representation in Marketing and Related Areas of Research*, 319–328. Berlin: Springer-Verlag.

Duda, R. O., and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. New York: John Wiley & Sons.

Duran, B. S., and Odell, P. L. (1974). *Cluster Analysis*. New York: Springer-Verlag.

Englemann, L., and Hartigan, J. A. (1969). "Percentage Points of a Test for Clusters." *Journal of the American Statistical Association* 64:1647–1648.

Everitt, B. S. (1979). "Unresolved Problems in Cluster Analysis." *Biometrics* 35:169–181.

Everitt, B. S. (1980). *Cluster Analysis*. 2nd ed. London: Heineman Educational Books.

Everitt, B. S. (1981). "A Monte Carlo Investigation of the Likelihood Ratio Test for the Number of Components in a Mixture of Normal Distributions." *Multivariate Behavioral Research* 16:171–80.

Everitt, B. S., and Hand, D. J. (1981). *Finite Mixture Distributions*. London: Chapman & Hall.

Gersho, A., and Gray, R. M. (1992). *Vector Quantization and Signal Compression*. Boston: Kluwer Academic.

Girman, C. J. (1994). "Cluster Analysis and Classification Tree Methodology as an Aid to Improve Understanding of Benign Prostatic Hyperplasia." Ph.D. diss., Department of Biostatistics, University of North Carolina at Chapel Hill.

Good, I. J. (1977). *The Botryology of Botryology, in Classification and Clustering*. Edited by J. Van Ryzin. New York: Academic Press.

Hartigan, J. A. (1975). *Clustering Algorithms*. New York: John Wiley & Sons.

Hartigan, J. A. (1977). "Distribution Problems in Clustering." In *Classification and Clustering*, edited by J. V. Ryzin, 45–72. New York: Academic Press.

Hartigan, J. A. (1978). "Asymptotic Distributions for Clustering Criteria." *Annals of Statistics* 6:117–131.

Hartigan, J. A. (1981). "Consistency of Single Linkage for High-Density Clusters." *Journal of the American Statistical Association* 76:388–394.

Hartigan, J. A. (1985a). "Statistical Theory in Clustering." *Journal of Classification* 2:63–76.

Hartigan, J. A., and Hartigan, P. M. (1985). "The Dip Test of Unimodality." *Annals of Statistics* 13:70–84.

Hartigan, P. M. (1985b). "Algorithm AS 217: Computation of the Dip Statistic to Test for Unimodality." *Journal of the Royal Statistical Society, Series C* 34:320–325.

Hawkins, D. M., Muller, M. W., and ten Krooden, J. A. (1982). "Cluster Analysis." In *Topics in Applied Multivariate Analysis*, edited by D. M. Hawkins, 303–356. Cambridge: Cambridge University Press.

Hubert, L. J. (1974). "Approximate Evaluation Techniques for the Single-Link and Complete-Link Hierarchical Clustering Procedures." *Journal of the American Statistical Association* 69:698–704.

Hubert, L. J., and Baker, F. B. (1977). "An Empirical Comparison of Baseline Models for Goodness-of-Fit in r-Diameter Hierarchical Clustering." In *Classification and Clustering*, edited by J. Van Ryzin, 131–153. New York: Academic Press.

Kaufman, L., and Rousseeuw, P. J. (1990). *Finding Groups in Data*. New York: John Wiley & Sons.

Klastorin, T. D. (1983). "Assessing Cluster Analysis Results." *Journal of Marketing Research* 20:92–98.

Lee, K. L. (1979). "Multivariate Tests for Clusters." *Journal of the American Statistical Association* 74:708–714.

Ling, R. F. (1973). "A Probability Theory of Cluster Analysis." *Journal of the American Statistical Association* 68:159–169.

MacQueen, J. B. (1967). "Some Methods for Classification and Analysis of Multivariate Observations." *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* 1:281–297.

Marriott, F. H. C. (1971). "Practical Problems in a Method of Cluster Analysis." *Biometrics* 27:501–514.

Marriott, F. H. C. (1975). "Separating Mixtures of Normal Distributions." *Biometrics* 31:767–769.

Massart, D. L., and Kaufman, L. (1983). *The Interpretation of Analytical Chemical Data by the Use of Cluster Analysis*. New York: John Wiley & Sons.

McClain, J. O., and Rao, V. R. (1975). "CLUSTISZ: A Program to Test for the Quality of Clustering of a Set of Objects." *Journal of Marketing Research* 12:456–460.

McLachlan, G. J., and Basford, K. E. (1988). *Mixture Models*. New York: Marcel Dekker.

Mezzich, J. E., and Solomon, H. (1980). *Taxonomy and Behavioral Science*. New York: Academic Press.

Milligan, G. W. (1980). "An Examination of the Effect of Six Types of Error Perturbation on Fifteen Clustering Algorithms." *Psychometrika* 45:325–342.

Milligan, G. W. (1981). "A Review of Monte Carlo Tests of Cluster Analysis." *Multivariate Behavioral Research* 16:379–407.

Milligan, G. W., and Cooper, M. C. (1985). "An Examination of Procedures for Determining the Number of Clusters in a Data Set." *Psychometrika* 50:159–179.

Minnotte, M. C. (1992). "A Test of Mode Existence with Applications to Multimodality." Ph.D. diss., Department of Statistics, Rice University.

Müller, D. W., and Sawitzki, G. (1991). "Excess Mass Estimates and Tests for Multimodality." *Journal of the American Statistical Association* 86:738–746.

Pollard, D. (1981). "Strong Consistency of k-Means Clustering." *Annals of Statistics* 9:135–140.

Polonik, W. (1993). *Measuring Mass Concentrations and Estimating Density Contour Clusters—an Excess Mass Approach*. Technical Report 7, Beiträge zur Statistik, Universität Heidelberg.

Sarle, W. S. (1982). "Cluster Analysis by Least Squares." In *Proceedings of the Seventh Annual SAS Users Group International Conference*, 651–653. Cary, NC: SAS Institute Inc.

Sarle, W. S. (1983). *Cubic Clustering Criterion*. Technical Report A-108, SAS Institute Inc., Cary, NC.

Scott, A. J., and Symons, M. J. (1971). "Clustering Methods Based on Likelihood Ratio Criteria." *Biometrics* 27:387–397.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. New York: Chapman & Hall.

Sneath, P. H. A., and Sokal, R. R. (1973). *Numerical Taxonomy*. San Francisco: W. H. Freeman.

Spath, H. (1980). *Cluster Analysis Algorithms*. Chichester, UK: Ellis Horwood.

Symons, M. J. (1981). "Clustering Criteria and Multivariate Normal Mixtures." *Biometrics* 37:35–43.

Thode, H. C., Jr., Mendell, N. R., and Finch, S. J. (1988). "Simulated Percentage Points for the Null Distribution of the Likelihood Ratio Test for a Mixture of Two Normals." *Biometrics* 44:1195–1201.

Titterington, D. M., Smith, A. F. M., and Makov, U. E. (1985). *Statistical Analysis of Finite Mixture Distributions*. New York: John Wiley & Sons.

Ward, J. H. (1963). "Hierarchical Grouping to Optimize an Objective Function." *Journal of the American Statistical Association* 58:236–244.

Wolfe, J. H. (1970). "Pattern Clustering by Multivariate Mixture Analysis." *Multivariate Behavioral Research* 5:329–350.

Wolfe, J. H. (1978). "Comparative Cluster Analysis of Patterns of Vocational Interest." *Multivariate Behavioral Research* 13:33–44.

Wong, M. A. (1982). "A Hybrid Clustering Method for Identifying High-Density Clusters." *Journal of the American Statistical Association* 77:841–847.

Wong, M. A., and Lane, T. (1983). "A *k*th Nearest Neighbor Clustering Procedure." *Journal of the Royal Statistical Society, Series B* 45:362–368.

Wong, M. A., and Schaack, C. (1982). "Using the *k*th Nearest Neighbor Clustering Procedure to Determine the Number of Subpopulations." *Proceedings of the Statistical Computing Section, American Statistical Association* 40–48.

Chapter 13
Introduction to Scoring, Standardization, and Ranking Procedures

Contents

Overview: Scoring, Standardization, and Ranking Procedures **251**

Overview: Scoring, Standardization, and Ranking Procedures

Several SAS/STAT procedures are utilities that produce an output data set with new variables that are transformations of data in the input data set. SAS/STAT software includes four of these procedures. The RANK procedure produces rank scores across observations, the SCORE procedure constructs functions across the variables, and the STANDARD and STDIZE procedures transform each variable individually.

RANK
: ranks the observations of each numeric variable and outputs ranks or rank scores. For a complete discussion of the RANK procedure, see the *Base SAS Procedures Guide: Statistical Procedures*.

SCORE
: constructs new variables that are linear combinations of old variables according to a scoring data set. This procedure is used with the FACTOR procedure and other procedures that output scoring coefficients.

STANDARD
: standardizes variables to a given mean and standard deviation. For a complete discussion of PROC STANDARD, see the *Base SAS Procedures Guide: Statistical Procedures*.

STDIZE
: standardizes variables by subtracting a location measure and dividing by a scale measure. A variety of location and scale measures are provided. Such measures include the mean, median, Huber's estimate, Tukey's biweight estimate, and Andrew's wave estimate.

Chapter 14
Introduction to Survival Analysis Procedures

Contents

Overview **253**
Survival Analysis Procedures **254**
Parametric Accelerated Failure Time Models: The LIFEREG Procedure 255
Nonparametric Methods for Right-Censored Data: The LIFETEST Procedure 255
Nonparametric Methods for Interval-Censored Data: The ICLIFETEST Procedure . . 256
Proportional Hazards Regression for Interval-Censored Data: The ICPHREG Procedure 256
Quantile Regression: The QUANTLIFE Procedure 256
Cox Regression and Extensions: The PHREG Procedure 256
Restricted Mean Survival Time Regression: The RMSTREG Procedure 257
Cox Regression for Survey Data: The SURVEYPHREG Procedure 257
Survival Analysis with SAS/STAT Procedures **257**
Bayesian Survival Analysis with SAS/STAT Procedures **258**
References **259**

Overview

Data that measure lifetime, or the length of time until the occurrence of an event, are called *lifetime, failure time,* or *survival* data. For example, a variable of interest might be the lifetime of diesel engines, the length of time a person stays at a job, or the survival time for heart transplant patients. Such data have special considerations that must be incorporated into any analysis.

Survival data consist of a response (event time, failure time, or survival time) variable that measures the duration of time until a specified event occurs and possibly a set of independent variables that are thought to be associated with the failure time variable. These independent variables (concomitant variables, covariates, or prognostic factors) can be either discrete, such as sex or race, or continuous, such as age or temperature. The system that gives rise to the event of interest can be biological (as with most medical data) or physical (as with engineering data). The purpose of survival analysis is to model the underlying distribution of the failure time variable and to assess the dependence of the failure time variable on the independent variables.

An intrinsic characteristic of survival data is the possibility of censoring of observations (that is, the actual time until the event is not observed). Such censoring can arise from withdrawal by a subject from the experiment or termination of the experiment. Because the response is usually a duration, some of the possible events might not yet have occurred when the period of data collection ends. For example, clinical trials are conducted over a finite period of time, with staggered entry of patients. That is, patients enter a clinical trial over time, and thus the length of follow-up varies by patient; consequently, the time to the event for all patients in the study might not be ascertained. In addition, some of the responses might be lost to follow-up

(for example, a participant might move or refuse to continue to participate) before data collection ends. In either case, only a lower bound on the failure time of the censored observations is known. Such observations are said to be *right-censored*. Thus, an additional variable is incorporated into the analysis to indicate which failure times are observed event times and which are censored times. More generally, the failure time might be known only to be smaller than a given value (*left-censored*) or known only to be within a given interval (*interval-censored*). Many possible censoring schemes arise in survival analysis. Maddala (1983) discusses several related types of censoring situations, and Kalbfleisch and Prentice (1980) also discuss several censoring schemes. Data that contains censored observations cannot be analyzed by ignoring the censored observations because, among other considerations, the longer-lived individuals are usually more likely to be right-censored. The method of analysis must take the censoring into account and correctly use both the censored observations and the uncensored observations.

Another characteristic of survival data is that the response cannot be negative. This suggests that a transformation of the survival time, such as a log transformation, might be necessary or that specialized methods might be more appropriate than those that assume a normal distribution of the error term. It is especially important to check any underlying assumptions as part of the analysis, because some of the models that are used are very sensitive to these assumptions.

Survival Analysis Procedures

The following SAS/STAT procedures are specifically designed for analyzing survival data:

ICLIFETEST computes nonparametric estimates of survivor functions for interval-censored data. You can use this procedure to compare the underlying survival distributions of two or more samples of interval-censored data.

ICPHREG fits proportional hazards regression models to interval-censored data. You can select a piecewise constant function as the baseline hazard function, or you can model the cumulative baseline hazard function by using a cubic spline or a discrete function.

LIFEREG fits parametric models to failure time data that can be left-censored, right-censored, or interval-censored. The log of the survival time is modeled as a linear effect of covariates and a random disturbance term, the distribution of which includes the Weibull, log-normal, and log-logistic distributions.

LIFETEST computes the Kaplan-Meier estimate of a survivor function and provides the log-rank test to compare the underlying hazards of two or more samples of right-censored data. You can also use this procedure to study the association between the failure time and a number of concomitant variables.

PHREG fits the Cox proportional hazards model and its extensions, which include the multiplicative intensity model, the shared frailty model, and models for competing-risks data.

QUANTLIFE performs quantile regression for survival data by modeling the quantiles of the lifetime variable as a function of the covariates. Because lifetime distributions are usually more skewed, the quantiles of the lifetime are more informative than the mean for summarizing the lifetime distribution.

RMSTREG	fits generalized linear models to the restricted mean survival time (RMST). The RMST, defined as the expected value of the time-to-event variable up to a prespecified time, is an interpretable summary measure of the time-to-event outcome.
SURVEYPHREG	is a Cox modeling procedure similar to PROC PHREG, appropriate for analyzing data that are collected from a survey sample.

The SEVERITY procedure in SAS/ETS software is also a survival analysis procedure.

Parametric Accelerated Failure Time Models: The LIFEREG Procedure

The LIFEREG procedure fits parametric accelerated failure time models to survival data that can be left-, right-, or interval-censored. The parametric model is of the form

$$y = \mathbf{x}'\boldsymbol{\beta} + \sigma\epsilon$$

where y is usually the log of the failure time variable, $\mathbf{x}$ is a vector of covariate values, $\boldsymbol{\beta}$ is a vector of unknown regression parameters, σ is an unknown scale parameter, and ϵ is an error term. The baseline distribution of the error term can be specified as one of several possible distributions, including (but not limited to) the log-normal, log-logistic, and Weibull distributions. Texts that discuss these parametric models include Kalbfleisch and Prentice (1980); Lawless (1982); Nelson (1990); Meeker and Escobar (1998). For more information about PROC LIFEREG, see Chapter 76, "The LIFEREG Procedure."

Nonparametric Methods for Right-Censored Data: The LIFETEST Procedure

The LIFETEST procedure computes nonparametric estimates of the survival distribution function. You can request either the product-limit (Kaplan and Meier 1958) or the life-table (actuarial) estimate of the distribution. Cox and Oakes (1984) and Kalbfleisch and Prentice (1980) provide good discussions of the product-limit estimator, and Lee (1992) and Elandt-Johnson and Johnson (1980) include detailed discussions of the life-table estimator. PROC LIFETEST computes nonparametric tests to compare the survival curves of two or more groups. The procedure also computes rank tests of association between the survival time variable and other concomitant variables, as given in Kalbfleisch and Prentice (1980, Chapter 6). For more information about PROC LIFETEST, see Chapter 77, "The LIFETEST Procedure."

Nonparametric Methods for Interval-Censored Data: The ICLIFETEST Procedure

The ICLIFETEST procedure computes nonparametric estimates of the survival functions and examines the equality of the survival functions via statistical tests. PROC ICLIFETEST is intended primarily for handling interval-censored data, whereas the LIFETEST procedure deals exclusively with right-censored data. You can use PROC ICLIFETEST to analyze data that are left-censored, interval-censored, or right-censored. However, if the data to be analyzed contain only exact or right-censored observations, it is recommended that PROC LIFETEST be used because it provides specialized methods for dealing with right-censored data. For more information about PROC ICLIFETEST, see Chapter 69, "The ICLIFETEST Procedure."

Proportional Hazards Regression for Interval-Censored Data: The ICPHREG Procedure

The ICPHREG procedure fits proportional hazards regression models to interval-censored data, including left-censored data and right-censored data as special cases. You can select a piecewise constant function as the baseline hazard, you can model the log of the cumulative baseline hazard function by using a cubic spline (Royston and Parmar 2002), or you can assume a discrete cumulative hazard function in which jumps are identified by the Turnbull method (Finkelstein 1986). PROC ICPHREG estimates the regression coefficients and the hazard parameters by maximizing the full likelihood function. For more information about PROC ICPHREG, see Chapter 70, "The ICPHREG Procedure."

Quantile Regression: The QUANTLIFE Procedure

The QUANTLIFE procedure explores how the conditional quantile of the failure time variable depends on covariates. Quantile regression provides a flexible way to capture heterogeneous effects, in the sense that the tails and the central location of the conditional distributions can vary differently with the covariates. Thus, quantile regression offers a powerful tool in survival analysis, where the lifetimes are skewed and extreme survival times can be of special interest (Koenker and Geling 2001; Huang 2010). For more information about PROC QUANTLIFE, see Chapter 102, "The QUANTLIFE Procedure."

Cox Regression and Extensions: The PHREG Procedure

The PHREG procedure fits the proportional hazards model of Cox (1972, 1975) to survival data that might be right-censored. The Cox model is a semiparametric model in which the hazard function of the survival time is given by

$$\lambda(t;\mathbf{x}) = \lambda_0(t)e^{\boldsymbol{\beta}'\mathbf{x}(t)}$$

where $\lambda_0(t)$ is an unspecified baseline hazard function, $\mathbf{x}(t)$ is a vector of covariate values (possibly time-dependent), and $\boldsymbol{\beta}$ is a vector of unknown regression parameters. The model is referred to as a semiparametric model, because part of the model involves the unspecified baseline function over time (which has an infinite dimension) and the other part involves a finite number of regression parameters. Texts that discuss the Cox

regression models include Collett (1994); Cox and Oakes (1984); Kalbfleisch and Prentice (1980); Lawless (1982). Extensions of the Cox model are discussed in Therneau and Grambsch (2000); Andersen et al. (1992); Fleming and Harrington (1991); Fine and Gray (1999). For more information about PROC PHREG, see Chapter 92, "The PHREG Procedure."

Restricted Mean Survival Time Regression: The RMSTREG Procedure

The RSMTREG procedure analyzes time-to-event data by performing a regression analysis on the restricted mean survival time (RMST), which is the expected value of the time-to-event variable up to a prespecified time point. The regression model is a generalized linear model that models the RMST directly and supports classification and interaction effects, along with the linear and log link functions. Models are fitted by using methods based on the estimating equations, particularly the pseudovalue (PV) method (Andersen, Hansen, and Klein 2004) and the inverse probability censoring weighting (IPCW) method (Tian, Zhao, and Wei 2014).

Cox Regression for Survey Data: The SURVEYPHREG Procedure

The SURVEYPHREG procedure fits the Cox proportional hazards model to sample survey data. The procedure is similar to the PHREG procedure, except that it incorporates complex sample design information in the analysis. The proportional hazards regression coefficients are estimated by maximizing a partial pseudo-log-likelihood function that incorporates the sampling weights. PROC SURVEYPHREG provides design-based variance estimates, confidence intervals, and tests for the estimated regression coefficients. For more information about PROC SURVEYPHREG, see Chapter 122, "The SURVEYPHREG Procedure."

Survival Analysis with SAS/STAT Procedures

The typical goal in survival analysis is to characterize the distribution of the survival time for a given population, to compare the survival distributions among different groups, or to study the relationship between the survival time and some concomitant variables.

A first step in analyzing of a set of survival data is to use the LIFETEST or ICLIFETEST procedure to compute and plot the estimate of the distribution of the survival time. In many applications, you often have several survival curves to compare. For example, you might want to compare the survival experiences of patients who receive different treatments for their disease. You can investigate the relationship between covariates and the survival time by computing estimates of the survival distribution function within strata that are defined by the covariates. In particular, if the proportional hazards model is appropriate, the estimates of the log(-log(SURVIVAL)) plotted against the log(TIME) should give approximately parallel lines, where SURVIVAL is the survival distribution estimate and TIME is the failure time variable. In addition, these lines should be approximately straight if the Weibull model is appropriate.

You can use knowledge of the association between failure time and the concomitant variables to select covariates for further investigation. The LIFETEST procedure computes linear rank statistics by using Wilcoxon or log-rank scores. These statistics and their estimated covariance matrix can be used with the REG procedure and the METHOD=RSQUARE option to find the subset of variables that produce the largest

joint test statistic for association. An illustration of this methodology is given in Example 77.1 of Chapter 77, "The LIFETEST Procedure."

Another approach to examining the relationship between survival time and the concomitant variables is through a regression model in which the survival time has a distribution that depends on the concomitant variables. The regression coefficients can be interpreted as describing the direction and strength of the effect of each explanatory variable on the survival time.

In many biological systems, the Cox model might be a reasonable description of the relationship between the distribution of the survival time and the prognostic factors. You use PROC PHREG to fit the Cox regression model. The regression coefficient is interpreted as the increase of the log-hazard ratio that results in the increase of one unit in the covariate. However, the underlying hazard function is left unspecified, and, as in any other model, the results can be misleading if the proportional hazards assumptions do not hold.

When you have interval-censored data, it is difficult to fit the semiparametric Cox regression models. But you can use the ICPHREG procedure to fit a different proportional hazards model, and the regression coefficients of such a model can still be interpreted as log-hazard ratios.

Accelerated failure time models are popular for fitting survival data from physical systems. In many cases, the underlying survival distribution is known empirically. You use PROC LIFEREG to fit these parametric models. Also, PROC LIFEREG can accommodate data that contain left-censored or interval-censored observations, which PROC PHREG does not allow.

A common technique for checking the validity of a regression model is to embed it in a larger model and use the likelihood ratio test to check whether the reduction to the actual model is valid. Other techniques include examining the residuals. Both PROC LIFEREG and PROC PHREG produce predicted values, residuals, and other computed values that can be used to assess the model's adequacy.

Bayesian Survival Analysis with SAS/STAT Procedures

You can request Bayesian analysis of survival models in the LIFEREG and PHREG procedures. In addition to enabling you to fit the Cox model, PROC PHREG also enables you to fit a piecewise exponential model. In Bayesian analysis, the model parameters are treated as random variables, and inference about parameters is based on the posterior distribution of the parameters. A posterior distribution is a weighted likelihood function of the data with a prior distribution of the parameters by using the Bayes theorem. The prior distribution enables you to incorporate into your analysis knowledge or experience of the likely range of values of the parameters of interest. You can specify normal or uniform prior distributions for the model regression coefficients in both the LIFEREG and PHREG procedures. In addition, you can specify a gamma or improper prior distribution for the scale or variance parameter in PROC LIFEREG. For the piecewise exponential model in PROC PHREG, you can specify normal or uniform prior distributions for the log-hazard parameters; alternatively, you can specify gamma or improper prior distributions for the hazard parameters. If you have no prior knowledge of the parameter values, you can use a noninformative prior distribution, and the results of a Bayesian analysis are very similar to those of a classical analysis based on maximum likelihood.

A closed form of the posterior distribution is often not feasible, and a Markov chain Monte Carlo method is used to simulate samples from the posterior distribution. You can perform inference by using the simulated samples, for example, to estimate the probability that a function of the parameters of interest lies within a specified range of values.

For an introduction to the basic concepts of Bayesian statistics, see Chapter 8, "Introduction to Bayesian Analysis Procedures." For a discussion of the advantages and disadvantages of Bayesian analysis, see "Bayesian Analysis: Advantages and Disadvantages" on page 154 in Chapter 8, "Introduction to Bayesian Analysis Procedures." For more information about Bayesian analysis, including guidance about choosing prior distributions, see Ibrahim, Chen, and Sinha (2001); Gelman et al. (2004); Gilks, Richardson, and Spiegelhalter (1996).

References

Andersen, P. K., Borgan, Ø., Gill, R. D., and Keiding, N. (1992). *Statistical Models Based on Counting Processes*. New York: Springer-Verlag.

Andersen, P. K., Hansen, M. G., and Klein, J. P. (2004). "Regression Analysis of Restricted Mean Survival Time Based on Pseudo-observations." *Lifetime Data Analysis* 10:335–350.

Collett, D. (1994). *Modelling Survival Data in Medical Research*. London: Chapman & Hall.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220. With discussion.

Cox, D. R. (1975). "Partial Likelihood." *Biometrika* 62:269–276.

Cox, D. R., and Oakes, D. (1984). *Analysis of Survival Data*. London: Chapman & Hall.

Elandt-Johnson, R. C., and Johnson, N. L. (1980). *Survival Models and Data Analysis*. New York: John Wiley & Sons.

Fine, J. P., and Gray, R. J. (1999). "A Proportional Hazards Model for the Subdistribution of a Competing Risk." *Journal of the American Statistical Association* 94:496–509.

Finkelstein, D. M. (1986). "A Proportional Hazards Model for Interval-Censored Failure Time Data." *Biometrics* 42:845–854.

Fleming, T. R., and Harrington, D. P. (1991). *Counting Processes and Survival Analysis*. New York: John Wiley & Sons.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. 2nd ed. London: Chapman & Hall.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996). *Markov Chain Monte Carlo in Practice*. London: Chapman & Hall.

Huang, Y. (2010). "Quantile Calculus and Censored Regression." *Annals of Statistics* 38:1607–1637.

Ibrahim, J. G., Chen, M.-H., and Sinha, D. (2001). *Bayesian Survival Analysis*. New York: Springer-Verlag.

Kalbfleisch, J. D., and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley & Sons.

Kaplan, E. L., and Meier, P. (1958). "Nonparametric Estimation from Incomplete Observations." *Journal of the American Statistical Association* 53:457–481.

Koenker, R., and Geling, O. (2001). "Reappraising Medfly Longevity: A Quantile Regression Survival Analysis." *Journal of the American Statistical Association* 96:458–468.

Lawless, J. F. (1982). *Statistical Methods and Methods for Lifetime Data*. New York: John Wiley & Sons.

Lee, E. T. (1992). *Statistical Methods for Survival Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Maddala, G. S. (1983). *Limited-Dependent and Qualitative Variables in Econometrics*. New York: Cambridge University Press.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Nelson, W. (1990). *Accelerated Testing: Statistical Models, Test Plans, and Data Analyses*. New York: John Wiley & Sons.

Royston, P., and Parmar, M. K. B. (2002). "Flexible Parametric Proportional-Hazards and Proportional-Odds Models for Censored Survival Data, with Application to Prognostic Modelling and Estimation of Treatment Effects." *Statistics in Medicine* 21:2175–2197.

Therneau, T. M., and Grambsch, P. M. (2000). *Modeling Survival Data: Extending the Cox Model*. New York: Springer-Verlag.

Tian, L., Zhao, L., and Wei, L. J. (2014). "Predicting the Restricted Mean Event Time with the Subject's Baseline Covariates in Survival Analysis." *Biostatistics* 15:222–233.

Chapter 15
Introduction to Survey Sampling and Analysis Procedures

Contents

Overview: Survey Sampling and Analysis Procedures **262**
The Survey Procedures **265**
PROC SURVEYSELECT 265
PROC SURVEYIMPUTE 266
PROC SURVEYMEANS 266
PROC SURVEYFREQ 267
PROC SURVEYREG 267
PROC SURVEYLOGISTIC 268
PROC SURVEYPHREG 268
Survey Design Specification **269**
Population 269
Stratification 270
Clustering 270
Multistage Sampling 270
Sampling Weights 270
Population Totals and Sampling Rates 271
Variance Estimation **271**
Example: Survey Sampling and Analysis Procedures **272**
Sample Selection 272
Survey Data Analysis 273
References **274**

Overview: Survey Sampling and Analysis Procedures

This chapter introduces the SAS/STAT procedures for survey sampling and describes how you can use these procedures to analyze survey data.

Researchers often use sample survey methodology to obtain information about a large population by selecting and measuring a sample from that population. Because of variability among items, researchers apply probability-based scientific designs to select the sample. This reduces the risk of a distorted view of the population and enables statistically valid inferences to be made from the sample. For more information about statistical sampling and analysis of complex survey data, see Lohr (2022); Kalton (1983); Cochran (1977); Kish (1965). To select probability-based random samples from a study population, you can use the SURVEYSELECT procedure, which provides a variety of probability sampling methods. To perform imputation of missing values in survey data, you can use the SURVEYIMPUTE procedure, which provides donor-based imputation methods. To analyze sample survey data, you can use the SURVEYMEANS, SURVEYFREQ, SURVEYREG, SURVEYLOGISTIC, and SURVEYPHREG procedures, which incorporate the sample design into the analyses.

Many SAS/STAT procedures, such as the MEANS, FREQ, GLM, LOGISTIC, and PHREG procedures, can compute sample means, produce crosstabulation tables, and estimate regression relationships. However, in most of these procedures, statistical inference is based on the assumption that the sample is drawn from an infinite population by simple random sampling. If the sample is in fact selected from a finite population by using a complex survey design, these procedures generally do not calculate the estimates and their variances according to the design actually used. Using analyses that are not appropriate for your sample design can lead to incorrect statistical inferences.

The SURVEYMEANS, SURVEYFREQ, SURVEYREG, SURVEYLOGISTIC, and SURVEYPHREG procedures properly analyze complex survey data by taking into account the sample design. You can use these procedures for multistage or single-stage designs, with or without stratification, and with or without unequal weighting. The survey analysis procedures provide a choice of variance estimation methods, which include Taylor series linearization, balanced repeated replication (BRR), bootstrap, and jackknife.

Table 15.1 briefly describes the SAS/STAT sampling and analysis procedures.

Table 15.1 Survey Sampling and Analysis Procedures

PROC SURVEYSELECT	
Selection methods	Simple random sampling (without replacement) Unrestricted random sampling (with replacement) Balanced bootstrap Systematic Sequential Bernoulli Poisson Probability proportional to size (PPS) sampling, with and without replacement PPS systematic PPS for two units per stratum PPS sequential with minimum replacement

Table 15.1 *continued*

Allocation methods	Proportional Optimal Neyman
Sampling tools	Stratified sampling Cluster sampling Replicated sampling Serpentine sorting Random assignment
PROC SURVEYIMPUTE	
Imputation methods	Single and multiple hot-deck Approximate Bayesian bootstrap Fully efficient fractional Two-stage fully efficient fractional Fractional hot-deck
PROC SURVEYMEANS	
Statistics	Means and totals Proportions Quantiles Geometric means Ratios Standard errors Confidence limits
Analyses	Hypothesis tests Domain analysis Comparison of domain means Poststratification
Graphics	Histograms Box plots Summary panel plots Domain box plots
PROC SURVEYFREQ	
Tables	One-way frequency tables Two-way and multiway crosstabulation tables Estimates of totals and proportions Standard errors Confidence limits
Analyses	Tests of goodness of fit Tests of independence Risks and risk differences Odds ratios and relative risks Kappa coefficients
Graphics	Weighted frequency and percent plots

Table 15.1 *continued*

	Mosaic plots Odds ratio, relative risk, and risk difference plots Kappa plots
PROC SURVEYREG	
Analyses	Linear regression model fitting Regression coefficients Covariance matrices Confidence limits Hypothesis tests Estimable functions Contrasts Least squares means (LS-means) of effects Custom hypothesis tests among LS-means Regression with constructed effects Predicted values and residuals Domain analysis
Graphics	Fit plots
PROC SURVEYLOGISTIC	
Analyses	Cumulative logit regression model fitting Logit, probit, and complementary log-log link functions Generalized logit regression model fitting Regression coefficients Covariance matrices Confidence limits Hypothesis tests Odds ratios Estimable functions Contrasts Least squares means (LS-means) of effects Custom hypothesis tests among LS-means Regression with constructed effects Model diagnostics Domain analysis
PROC SURVEYPHREG	
Analyses	Proportional hazards regression model fitting Breslow and Efron likelihoods Regression coefficients Covariance matrices Confidence limits Hypothesis tests Hazard ratios Contrasts Predicted values and standard errors

Table 15.1 *continued*

	Martingale, Schoenfeld, score, and deviance residuals
	Domain analysis

The Survey Procedures

The SURVEYSELECT procedure provides methods for probability sample selection. The SURVEYIMPUTE procedure provides donor-based imputation methods for handling missing values in survey data. The SURVEYMEANS, SURVEYFREQ, SURVEYREG, SURVEYLOGISTIC, and SURVEYPHREG procedures provide statistical analyses for survey data. The following sections contain brief descriptions of these procedures. For more information, see the chapters for the individual procedures.

PROC SURVEYSELECT

The SURVEYSELECT procedure provides a variety of methods for selecting probability-based random samples. The procedure can select a simple random sample or can sample according to a complex multistage sample design that includes stratification, clustering, and unequal probabilities of selection. When you use probability sampling, each unit in the survey population has a known, positive probability of selection. This property of probability sampling avoids selection bias and enables you to use statistical theory to make valid inferences from the sample to the survey population.

PROC SURVEYSELECT provides methods for both equal-probability sampling and probability proportional to size (PPS) sampling. Available equal-probability sampling methods include simple random sampling (without replacement) and unrestricted random sampling (with replacement) in addition to systematic, sequential, Bernoulli, and balanced bootstrap selection.

In PPS sampling, a unit's selection probability is proportional to its size measure. PPS sampling is often used in cluster sampling, where you select clusters (groups of sampling units) of varying size in the first stage of selection. Available PPS methods include without replacement, with replacement, systematic, and sequential with minimum replacement. PROC SURVEYSELECT can apply these selection methods for stratified, clustered, and replicated sample designs.

For stratified sampling, PROC SURVEYSELECT provides survey design methods to allocate the total sample size among the strata. Available allocation methods include proportional, Neyman, and optimal allocation. Optimal allocation maximizes the estimation precision within the available resources by taking into account stratum sizes, costs, and variances.

PROC SURVEYSELECT also provides random assignment (partitioning).

For more information, see Chapter 124, "The SURVEYSELECT Procedure."

PROC SURVEYIMPUTE

The SURVEYIMPUTE procedure provides the following donor-based imputation methods for handling missing values in survey data: single and multiple hot-deck imputation, approximate Bayesian bootstrap imputation, fully efficient fractional imputation, two-stage fully efficient fractional imputation, and fractional hot-deck imputation. When you use fully efficient fractional imputation or fractional hot-deck imputation, PROC SURVEYIMPUTE produces replicate weights that appropriately account for the imputation. You can use these replicate weights in any survey analysis procedure to correctly estimate the variance for an estimator that uses the imputed data. For more information about fully efficient fractional imputation, see Kim and Shao (2014, sections 8.4 and 8.5) and Fuller (2009, section 5.2).

For more information, see Chapter 119, "The SURVEYIMPUTE Procedure."

PROC SURVEYMEANS

The SURVEYMEANS procedure produces estimates of population means and totals from sample survey data. The procedure also computes estimates of proportions for categorical variables, estimates of quantiles for continuous variables, estimates of geometric means for positive continuous variables, and ratio estimates of means and proportions. PROC SURVEYMEANS provides standard errors, confidence limits, and *t* tests for these statistics.

PROC SURVEYMEANS provides domain analysis, which computes estimates for domains (subpopulations) in addition to estimates for the entire study population. PROC SURVEYMEANS also provides comparisons between domain means. Formation of domains can be unrelated to the sample design, and therefore the domain sample sizes can be random variables. Domain analysis takes this variability into account by using the entire sample to estimate the variance of domain estimates. Domain analysis is also known as subgroup analysis, subpopulation analysis, and subdomain analysis.

PROC SURVEYMEANS also performs poststratification, which adjusts the sampling weights so that their distribution matches known auxiliary information. Poststratification is often used to improve the efficiency of the analysis and adjust for nonresponse. PROC SURVEYMEANS provides poststratified analyses and also produces poststratification weights that can be used in the other survey analysis procedures. For more information about poststratification, see Fuller (2009); Lohr (2022); Wolter (2007); Rao, Yung, and Hidiroglou (2002).

PROC SURVEYMEANS uses ODS Graphics to create graphs as part of its output. Available statistical graphics include histograms and summary panel plots for continuous variables, box plots, and domain box plots.

For more information, see Chapter 121, "The SURVEYMEANS Procedure."

PROC SURVEYFREQ

The SURVEYFREQ procedure produces one-way to *n*-way frequency and crosstabulation tables from sample survey data. These tables include estimates of population totals, population proportions (overall proportions, and row and column proportions), and the corresponding standard errors. Confidence limits, coefficients of variation, and design effects are also available. The procedure provides the following types of design-adjusted confidence limits for proportions: Wald, logit, modified Wilson (score), and modified Clopper-Pearson (exact).

For one-way frequency tables, PROC SURVEYFREQ provides Rao-Scott chi-square goodness-of-fit tests, which are adjusted for the sample design. You can test a null hypothesis of equal proportions for a one-way frequency table, or you can input custom null hypothesis proportions for the test. For two-way frequency tables, PROC SURVEYFREQ provides design-adjusted tests of independence or no association between the row and column variables. These tests include the Rao-Scott chi-square test, the Rao-Scott likelihood ratio test, the Wald chi-square test, and the Wald log-linear chi-square test.

For 2×2 tables, PROC SURVEYFREQ computes estimates and confidence limits for risks (row proportions), the risk difference, the odds ratio, and relative risks. For square tables, PROC SURVEYFREQ computes simple and weighted kappa coefficients.

PROC SURVEYFREQ uses ODS Graphics to create graphs as part of its output. The procedure provides weighted frequency and percent plots, which can be displayed as bar charts or dot plots in various formats. For two-way tables, PROC SURVEYFREQ also provides mosaic plots. For multiway tables, PROC SURVEYFREQ produces odds ratio, relative risk, risk difference, and kappa coefficient plots.

For more information, see Chapter 118, "The SURVEYFREQ Procedure."

PROC SURVEYREG

The SURVEYREG procedure performs regression analysis for sample survey data. The procedure fits linear models and computes regression coefficients and their variance-covariance matrices. PROC SURVEYREG enables you to specify classification effects by using the same syntax that the GLM procedure uses.

PROC SURVEYREG provides hypothesis tests for the model effects. The procedure also provides custom hypothesis tests for linear combinations of the regression parameters. The procedure computes confidence limits for the parameter estimates and also for any specified linear functions of the regression parameters. The procedure can produce an output data set that contains the predicted values from the linear regression, their standard errors and confidence limits, and the residuals.

PROC SURVEYREG also performs regression analysis for domains.

PROC SURVEYREG uses ODS Graphics to create graphs as part of its output. For models that depend on at most one regressor excluding the intercept, the procedure produces fit plots, which can be displayed as bubble plots or heat maps. In bubble plots, the bubble area is proportional to the observation's weight. In heat maps, the heat color represents the sum of the weights at the corresponding location.

For more information, see Chapter 123, "The SURVEYREG Procedure."

PROC SURVEYLOGISTIC

The SURVEYLOGISTIC procedure provides logistic regression analysis for sample survey data. Logistic regression analysis investigates the relationship between discrete responses and a set of explanatory variables. PROC SURVEYLOGISTIC fits linear logistic regression models for discrete response survey data by the method of maximum likelihood and incorporates the sample design into the analysis. The SURVEYLOGISTIC procedure enables you to specify categorical classification variables (also known as CLASS variables) as explanatory variables in the model by using the same syntax for main effects and interactions as in the GLM and LOGISTIC procedures.

The following link functions are available for regression in PROC SURVEYLOGISTIC: the cumulative logit function (CLOGIT), the generalized logit function (GLOGIT), the probit function (PROBIT), and the complementary log-log function (CLOGLOG). The procedure performs maximum likelihood estimation of the regression coefficients with either the Fisher scoring algorithm or the Newton-Raphson algorithm.

PROC SURVEYLOGISTIC also performs logistic regression analysis for domains.

For more information, see Chapter 120, "The SURVEYLOGISTIC Procedure."

PROC SURVEYPHREG

The SURVEYPHREG procedure performs regression analysis based on the Cox proportional hazards model for sample survey data. Cox's semiparametric model is widely used in the analysis of survival data to estimate hazard rates when explanatory variables are available. The regression coefficients are estimated by maximizing a pseudo-partial-likelihood function that incorporates the sampling weights. The procedure provides design-based variance estimates, confidence intervals, and tests for the estimated regression coefficients.

PROC SURVEYPHREG provides hypothesis tests for the model effects. The procedure also provides custom hypothesis tests for linear combinations of the regression parameters. The procedure computes hazard ratios and their confidence limits. The procedure can produce several observation-level output statistics, such as predicted values and their standard errors, martingale residuals, Schoenfeld residuals, score residuals, and deviance residuals.

PROC SURVEYPHREG also performs proportional hazards regressions for domains.

For more information, see Chapter 122, "The SURVEYPHREG Procedure."

Survey Design Specification

Survey sampling is the process of selecting a probability-based sample from a finite population according to a sample design. You then collect data from these selected units and use them to estimate characteristics of the entire population.

A *sample design* encompasses the rules and operations by which you select sampling units from the population and the computation of sample statistics, which are estimates of the population values of interest. The objective of your survey often determines appropriate sample designs and valid data collection methodology. A complex sample design can include stratification, clustering, multiple stages of selection, and unequal weighting. The survey procedures can be used for single-stage designs or for multistage designs, with or without stratification, and with or without unequal weighting.

To analyze your survey data with the SURVEYMEANS, SURVEYFREQ, SURVEYREG, SURVEYLOGISTIC, and SURVEYPHREG procedures, you need to specify sample design information for the procedures. This information can include design strata, clusters, and sampling weights. All the survey analysis procedures use the same syntax for specifying sample design information. You provide sample design information by using the STRATA, CLUSTER, and WEIGHT statements, and by specifying the RATE= or TOTAL= option in the PROC statement.

If you provide replicate weights for replication variance estimation, you do not need to specify a STRATA or CLUSTER statement. Otherwise, you should specify STRATA and CLUSTER statements whenever your design includes stratification and clustering.

When there are clusters (PSUs) in the sample design, the procedures estimate variance by using the PSUs, as described in the section "Variance Estimation" on page 271. For a multistage sample design, the procedures use only the first stage of the sample design for variance estimation. Therefore, the required input includes only first-stage cluster (PSU) and first-stage stratum identification. You do not need to input design information about any additional stages of sampling.

The following sections provide brief descriptions of basic sample design concepts and terminology used in the survey procedures. For more information, see Lohr (2022); Kalton (1983); Cochran (1977); Kish (1965).

Population

Population refers to the target population, which is the group of units (individuals or elements) of interest for study. Often, the primary objective is to estimate certain characteristics of this population, which are called *population values*. A *sampling unit* is an individual or element in the target population. A *sample* is a subset of the population that is selected for the study.

Before you use the survey procedures, you should have a well-defined target population, sampling units, and an appropriate sample design.

In order to select a sample according to your sample design, you need to have a list of sampling units in the population. This is called a *sampling frame*. PROC SURVEYSELECT uses probability-based selection methods to select a sample from a sampling frame.

Stratification

Stratified sampling involves selecting samples independently within strata, which are nonoverlapping subgroups of the survey population. Stratification controls the distribution of the sample size in the strata. It is widely used to meet a variety of survey objectives. For example, with stratification you can ensure adequate sample sizes for subgroups of interest, including small subgroups, or you can use stratification to improve the precision of overall estimates. To improve precision, units within strata should be as homogeneous as possible for the characteristics of interest.

Clustering

Cluster sampling involves selecting clusters, which are groups of sampling units. For example, clusters might be schools, hospitals, or geographical areas, and sampling units might be students, patients, or citizens. Cluster sampling can provide efficiency in frame construction and other survey operations. However, it can also result in a loss in precision of your estimates, compared to a nonclustered sample of the same size. To minimize this effect, units within clusters should be as heterogeneous as possible for the characteristics of interest.

Multistage Sampling

In *multistage sampling*, you select an initial (first-stage) sample that is based on groups of elements in the population, which are called *primary sampling units* (*PSUs*).

Then you create a second-stage sample by drawing a subsample from each selected PSU in the first-stage sample. By repeating this operation, you can select a higher-stage sample. If you include all the elements from the selected primary sampling units, then the two-stage sample is a cluster sample.

Sampling Weights

Sampling weights, which are also known as *survey weights*, are positive values associated with the units in your sample. Ideally, the weight of a sampling unit should be the "frequency" that the sampling unit represents in the target population.

Often, sampling weights are the reciprocals of the selection probabilities for the sampling units. When you use PROC SURVEYSELECT, the procedure generates the sampling weight component for each stage of the design, and you can multiply these sampling weight components to obtain the final sampling weights. Sometimes, sampling weights also include nonresponse adjustments, poststratification, or regression adjustments by using supplemental information.

When the sampling units have unequal weights, you must provide the weights to the survey analysis procedures. If you do not specify sampling weights, the procedures use equal weights in the analyses.

Population Totals and Sampling Rates

For Taylor series and bootstrap variance estimation, the survey procedures include a finite population correction if you provide either the sampling rate or the population total. For BRR, jackknife, and replicate weight variance estimation, the procedures do not use a finite population correction.

The sampling rate is the ratio of the sample size (the number of sampling units in the sample) n to the population size (the total number of sampling units in the target population) N, $f = n/N$. This ratio is also called the *sampling fraction*. If you select a sample without replacement, the extra efficiency compared to selecting a sample with replacement can be measured by the *finite population correction* (*fpc*) factor, (1-f). For fairly small sampling fractions, it is appropriate to ignore this correction. For more information, see Cochran (1977) and Kish (1965).

To specify the sampling rate when your design has multiple stages, you should input the first-stage sampling rate, which is the ratio of the number of PSUs in the sample to the total number of PSUs in the study population. To specify the population total when your design has multiple stages, you should input the total number of PSUs in the population.

Variance Estimation

The survey analysis procedures provide a choice of variance estimation methods for complex survey designs. In addition to the Taylor series linearization method, the procedures offer the following replication-based (resampling) methods: balanced repeated replication (BRR), bootstrap, and delete-1 jackknife. These variance estimation methods usually give similar, satisfactory results (Lohr 2022; Särndal, Swensson, and Wretman 1992; Wolter 2007). The choice of a variance estimation method can depend on the sample design used, the sample design information available, the parameters to be estimated, and computational issues. For more information, see Lohr (2022).

The Taylor series linearization method is appropriate for all designs where the first-stage sample is selected with replacement, or where the first-stage sampling fraction is small, as it often is in practice. The Taylor series method obtains a linear approximation for the estimator and then uses the variance estimate for this approximation to estimate the variance of the estimate itself (Fuller 1975; Woodruff 1971). When there are clusters (PSUs) in the sample design, the procedures estimate the variance from the variation among the PSUs. When the design is stratified, the procedures pool stratum variance estimates to compute the overall variance estimate.

For a multistage sample design, the Taylor series method uses only the first stage of the sample design. Therefore, the required input includes only first-stage cluster (PSU) and first-stage stratum identification. You do not need to input design information about any additional stages of sampling.

Replication methods for variance estimation draw multiple replicates (or subsamples) from the full sample by following a specific resampling scheme. Commonly used resampling schemes include balanced repeated replication (BRR), bootstrap, and jackknife. The parameter of interest is estimated from each replicate, and the variability among the replicate estimates is used to estimate the overall variance of the parameter estimate.

The BRR variance estimation method requires a stratified sample design with two PSUs in each stratum. Each replicate is obtained by deleting one PSU per stratum according to the corresponding Hadamard matrix

and adjusting the original weights for the remaining PSUs. The adjusted weights are called *replicate weights*. The survey procedures also provide Fay's method, which is a modification of the BRR method.

The bootstrap method creates replicate samples by using with-replacement simple random sampling of PSUs. If the sample design is stratified, bootstrap samples are selected independently within each stratum, and each stratum must contain at least two PSUs. In each replicate, the original sampling weights are adjusted to compute bootstrap replicate weights that reflect the full sample.

The jackknife method deletes one PSU at a time from the full sample to create replicates, and modifies the original weights to obtain replicate weights. The total number of replicates equals the number of PSUs. If the sample design is stratified, each stratum must contain at least two PSUs, and the jackknife method is applied separately within each stratum.

Instead of having the survey procedures generate replicate weights for the analysis, you can directly input your own replicate weights. This can be useful if you need to do multiple analyses with the same set of replicate weights, or if you have access to replicate weights without complete design information.

See the chapters on the survey procedures for complete details. For more information about variance estimation for sample survey data, see Lohr (2022); Wolter (2007); Särndal, Swensson, and Wretman (1992); Lee, Forthofer, and Lorimor (1989); Cochran (1977); Kish (1965); Hansen, Hurwitz, and Madow (1953).

Example: Survey Sampling and Analysis Procedures

This section demonstrates how you can use the survey procedures to select a probability-based sample and then analyze the survey data to make inferences about the population. The analyses include descriptive statistics and regression analysis. This example is a survey of income and expenditures for a group of households in North Carolina and South Carolina. The goals of the survey are as follows:

- Estimate total income and total living expenses
- Estimate the median income and the median living expenses
- Investigate the linear relationship between income and living expenses

Sample Selection

To select a sample by using PROC SURVEYSELECT, you provide a SAS data set that contains the sampling frame (the list of units from which the sample is to be selected). You can also specify the selection method, the sample size or sampling rate, and other selection parameters. PROC SURVEYSELECT selects the sample and produces an output data set that contains the selected units, their selection probabilities, and their sampling weights. For more information, see Chapter 124, "The SURVEYSELECT Procedure."

In this example, the sample design is a stratified sample design, with households as the sampling units and selection by simple random sampling. The SAS data set HHFrame contains the sampling frame, which is the list of households in the survey population. The sampling frame is stratified by the variables State and Region. Within strata, households are selected by simple random sampling. The following PROC SURVEYSELECT statements select a probability sample of households according to this sample design:

```
proc surveyselect data=HHFrame out=HHSample
                  method=srs n=(3, 5, 3, 6, 2);
   strata State Region;
run;
```

The STRATA statement names the stratification variables State and Region. In the PROC SURVEYSELECT statement, the DATA= option names the SAS data set HHFrame as the input data set (or sampling frame) from which to select the sample. The OUT= option stores the sample in the SAS data set named HHSample. The METHOD=SRS option specifies simple random sampling as the sample selection method. The N= option specifies the stratum sample sizes.

The SURVEYSELECT procedure then selects a stratified random sample of households and produces the output data set HHSample, which contains the selected households together with their selection probabilities and sampling weights. The data set HHSample also contains the sampling unit identification variable Id and the stratification variables State and Region from the input data set HHFrame.

Survey Data Analysis

You can use the SURVEYMEANS and SURVEYREG procedures to estimate population values and perform regression analyses for survey data. The following example briefly shows the capabilities of these procedures. For more information, see Chapter 121, "The SURVEYMEANS Procedure," and Chapter 123, "The SURVEYREG Procedure."

The following PROC SURVEYMEANS statements estimate the total income and living expenses for the survey population based on the data from the stratified sample design:

```
proc surveymeans data=HHSample sum median;
   var Income Expense;
   strata State Region;
   weight Weight;
run;
```

The PROC SURVEYMEANS statement invokes the procedure, and the DATA= option names the SAS data set HHSample as the input data set to be analyzed. The keywords SUM and MEDIAN request estimates of population totals and medians.

The VAR statement specifies the two analysis variables Income and Expense. The STRATA statement names the stratification variables State and Region. The WEIGHT statement specifies the sampling weight variable Weight.

You can use PROC SURVEYREG to perform regression analysis for survey data. Suppose that, in order to explore the relationship between household income and living expenses in the survey population, you choose the following linear model:

$$\text{Expense} = \alpha + \beta * \text{Income} + \text{error}$$

The following PROC SURVEYREG statements fit this linear model for the survey population based on the data from the stratified sample design:

```
proc surveyreg data=HHSample;
   strata State Region ;
   model  Expense = Income;
   weight Weight;
run;
```

The STRATA statement names the stratification variables State and Region. The MODEL statement specifies the model, with Expense as the dependent variable and Income as the independent variable. The WEIGHT statement specifies the sampling weight variable Weight.

References

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Fuller, W. A. (1975). "Regression Analysis for Sample Survey." *Sankhyā, Series C* 37:117–132.

Fuller, W. A. (2009). *Sampling Statistics*. Hoboken, NJ: John Wiley & Sons.

Hansen, M. H., Hurwitz, W. N., and Madow, W. G. (1953). *Sample Survey Methods and Theory.* 2 vols. New York: John Wiley & Sons.

Kalton, G. (1983). *Introduction to Survey Sampling.* Vol. 07-035 of University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Kim, J. K., and Shao, J. (2014). *Statistical Methods for Handling Incomplete Data*. Boca Raton, FL: CRC Press.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Lee, E. S., Forthofer, R. N., and Lorimor, R. J. (1989). *Analyzing Complex Survey Data.* Vol. 07-071 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Lohr, S. L. (2022). *Sampling: Design and Analysis*. 3rd ed. Boca Raton, FL: CRC Press.

Rao, J. N. K., Yung, W., and Hidiroglou, M. A. (2002). "Estimating Equations for the Analysis of Survey Data Using Poststratification Information." *Sankhyā, Series A* 64:364–378.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Wolter, K. M. (2007). *Introduction to Variance Estimation*. 2nd ed. New York: Springer.

Woodruff, R. S. (1971). "A Simple Method for Approximating the Variance of a Complicated Estimate." *Journal of the American Statistical Association* 66:411–414.

Chapter 16
The Four Types of Estimable Functions

Contents

Overview **275**
Estimability **276**
General Form of an Estimable Function 277
Introduction to Reduction Notation 278
Examples 279
Estimable Functions **282**
Type I SS and Estimable Functions 282
Type II SS and Estimable Functions 283
Type III and IV SS and Estimable Functions 287
References **291**

Overview

Many regression and analysis of variance procedures in SAS/STAT label tests for various effects in the model as Type I, Type II, Type III, or Type IV. These four types of hypotheses might not always be sufficient for a statistician to perform all desired inferences, but they should suffice for the vast majority of analyses. This chapter explains the hypotheses involved in each of the four test types. For additional discussion, see Freund, Littell, and Spector (1991) or Milliken and Johnson (1984).

The primary context of the discussion is testing linear hypotheses in least squares regression and analysis of variance, such as with PROC GLM. In this context, tests correspond to hypotheses about linear functions of the true parameters and are evaluated using sums of squares of the estimated parameters. Thus, there will be frequent references to Type I, II, III, and IV (estimable) functions and corresponding Type I, II, III, and IV sums of squares, or simply SS.

Estimability

Given a response or dependent variable $\mathbf{Y}$, predictors or independent variables $\mathbf{X}$, and a linear expectation model $\mathrm{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{\beta}$ relating the two, a primary analytical goal is to estimate or test for the significance of certain linear combinations of the elements of $\boldsymbol{\beta}$. For least squares regression and analysis of variance, this is accomplished by computing linear combinations of the observed $\mathbf{Y}$s. An unbiased linear estimate of a specific linear function of the individual βs, say $\mathbf{L}\boldsymbol{\beta}$, is a linear combination of the $\mathbf{Y}$s that has an expected value of $\mathbf{L}\boldsymbol{\beta}$. Hence, the following definition:

> A linear combination of the parameters $\mathbf{L}\boldsymbol{\beta}$ is estimable if and only if a linear combination of the $\mathbf{Y}$s exists that has expected value $\mathbf{L}\boldsymbol{\beta}$.

Any linear combination of the $\mathbf{Y}$s, for instance $\mathbf{KY}$, will have expectation $\mathrm{E}[\mathbf{KY}] = \mathbf{KX}\boldsymbol{\beta}$. Thus, the expected value of any linear combination of the $\mathbf{Y}$s is equal to that same linear combination of the rows of $\mathbf{X}$ multiplied by $\boldsymbol{\beta}$. Therefore,

> $\mathbf{L}\boldsymbol{\beta}$ is estimable if and only if there is a linear combination of the rows of $\mathbf{X}$ that is equal to $\mathbf{L}$—that is, if and only if there is a $\mathbf{K}$ such that $\mathbf{L} = \mathbf{KX}$.

Thus, the rows of $\mathbf{X}$ form a generating set from which any estimable $\mathbf{L}$ can be constructed. Since the row space of $\mathbf{X}$ is the same as the row space of $\mathbf{X}'\mathbf{X}$, the rows of $\mathbf{X}'\mathbf{X}$ also form a generating set from which all estimable $\mathbf{L}$s can be constructed. Similarly, the rows of $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$ also form a generating set for $\mathbf{L}$.

Therefore, if $\mathbf{L}$ can be written as a linear combination of the rows of $\mathbf{X}$, $\mathbf{X}'\mathbf{X}$, or $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$, then $\mathbf{L}\boldsymbol{\beta}$ is estimable.

In the context of least squares regression and analysis of variance, an estimable linear function $\mathbf{L}\boldsymbol{\beta}$ can be estimated by $\mathbf{L}\widehat{\boldsymbol{\beta}}$, where $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y}$. From the general theory of linear models, the unbiased estimator $\mathbf{L}\widehat{\boldsymbol{\beta}}$ is, in fact, the *best* linear unbiased estimator of $\mathbf{L}\boldsymbol{\beta}$, in the sense of having minimum variance as well as maximum likelihood when the residuals are normal. To test the hypothesis that $\mathbf{L}\boldsymbol{\beta} = \mathbf{0}$, compute the sum of squares

$$\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}) = (\mathbf{L}\widehat{\boldsymbol{\beta}})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}\mathbf{L}\widehat{\boldsymbol{\beta}}$$

and form an F test with the appropriate error term. Note that in contexts more general than least squares regression (for example, generalized and/or mixed linear models), linear hypotheses are often tested by analogous sums of squares of the estimated linear parameters $(\mathbf{L}\widehat{\boldsymbol{\beta}})'(\mathrm{Var}[\mathbf{L}\widehat{\boldsymbol{\beta}}])^{-1}\mathbf{L}\widehat{\boldsymbol{\beta}}$.

General Form of an Estimable Function

This section demonstrates a shorthand technique for displaying the generating set for any estimable **L**. Suppose

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \boldsymbol{\beta} = \begin{bmatrix} \mu \\ A_1 \\ A_2 \\ A_3 \end{bmatrix}$$

X is a generating set for **L**, but so is the smaller set

$$\mathbf{X}^* = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$\mathbf{X}^*$ is formed from **X** by deleting duplicate rows.

Since all estimable **L**s must be linear functions of the rows of $\mathbf{X}^*$ for $\mathbf{L}\boldsymbol{\beta}$ to be estimable, an **L** for a single-degree-of-freedom estimate can be represented symbolically as

$$L1 \times (1\ 1\ 0\ 0) + L2 \times (1\ 0\ 1\ 0) + L3 \times (1\ 0\ 0\ 1)$$

or

$$\mathbf{L} = (L1 + L2 + L3,\ L1,\ L2,\ L3)$$

For this example, $\mathbf{L}\boldsymbol{\beta}$ is estimable if and only if the first element of **L** is equal to the sum of the other elements of **L** or if

$$\mathbf{L}\boldsymbol{\beta} = (L1 + L2 + L3) \times \mu + L1 \times A_1 + L2 \times A_2 + L3 \times A_3$$

is estimable for any values of *L1*, *L2*, and *L3*.

If other generating sets for **L** are represented symbolically, the symbolic notation looks different. However, the inherent nature of the rules is the same. For example, if row operations are performed on $\mathbf{X}^*$ to produce an identity matrix in the first 3×3 submatrix of the resulting matrix

$$\mathbf{X}^{**} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

then $\mathbf{X}^{**}$ is also a generating set for **L**. An estimable **L** generated from $\mathbf{X}^{**}$ can be represented symbolically as

$$\mathbf{L} = (L1,\ L2,\ L3,\ L1 - L2 - L3)$$

Note that, again, the first element of **L** is equal to the sum of the other elements.

With multiple generating sets available, the question arises as to which one is the best to represent $\mathbf{L}$ symbolically. Clearly, a generating set containing a minimum of rows (of full row rank) and a maximum of zero elements is desirable.

The generalized g_2-inverse $(\mathbf{X}'\mathbf{X})^-$ of $\mathbf{X}'\mathbf{X}$ computed by the modified sweep operation (Goodnight 1979) has the property that $(\mathbf{X}'\mathbf{X})^-\mathbf{X}'\mathbf{X}$ usually contains numerous zeros. For this reason, in PROC GLM the nonzero rows of $(\mathbf{X}'\mathbf{X})^-\mathbf{X}'\mathbf{X}$ are used to represent $\mathbf{L}$ symbolically.

If the generating set represented symbolically is of full row rank, the number of symbols $(L1, L2, \ldots)$ represents the maximum rank of any testable hypothesis (in other words, the maximum number of linearly independent rows for any $\mathbf{L}$ matrix that can be constructed). By letting each symbol in turn take on the value of 1 while the others are set to 0, the original generating set can be reconstructed.

Introduction to Reduction Notation

Reduction notation can be used to represent differences in sums of squares (SS) for two models. The notation $R(\mu, A, B, C)$ denotes the complete main-effects model for effects A, B, and C. The notation

$$R(A \mid \mu, B, C)$$

denotes the difference between the model SS for the complete main-effects model containing A, B, and C and the model SS for the reduced model containing only B and C.

In other words, this notation represents the differences in model SS produced by

```
proc glm;
   class a b c;
   model y = a b c;
run;
```

and

```
proc glm;
   class b c;
   model y = b c;
run;
```

As another example, consider a regression equation with four independent variables. The notation $R(\beta_3, \beta_4 \mid \beta_1, \beta_2)$ denotes the differences in model SS between

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \epsilon$$

and

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \epsilon$$

This is the difference in the model SS for the models produced, respectively, by

```
model y = x1 x2 x3 x4;
```

and

```
model y = x1 x2;
```

The following examples demonstrate the ability to manipulate the symbolic representation of a generating set. Note that any operations performed on the symbolic notation have corresponding row operations that are performed on the generating set itself.

Examples

A One-Way Classification Model

For the model

$$Y = \mu + A_i + \epsilon \qquad i = 1, 2, 3$$

the general form of estimable functions $\mathbf{L}\boldsymbol{\beta}$ is (from the previous example)

$$\mathbf{L}\boldsymbol{\beta} = L1 \times \mu + L2 \times A_1 + L3 \times A_2 + (L1 - L2 - L3) \times A_3$$

Thus,

$$\mathbf{L} = (L1, L2, L3, L1 - L2 - L3)$$

Tests involving only the parameters A_1, A_2, and A_3 must have an $\mathbf{L}$ of the form

$$\mathbf{L} = (0, L2, L3, -L2 - L3)$$

Since this $\mathbf{L}$ for the A parameters involves only two symbols, hypotheses with at most two degrees of freedom can be constructed. For example, letting $(L2, L3)$ be $(1, 0)$ and $(0, 1)$, respectively, yields

$$\mathbf{L} = \begin{bmatrix} 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

The preceding $\mathbf{L}$ can be used to test the hypothesis that $A_1 = A_2 = A_3$. For this example, any $\mathbf{L}$ with two linearly independent rows with column 1 equal to zero produces the same sum of squares. For example, a joint test for linear and quadratic effects of A

$$\mathbf{L} = \begin{bmatrix} 0 & 1 & 0 & -1 \\ 0 & 1 & -2 & 1 \end{bmatrix}$$

gives the same SS. In fact, for any $\mathbf{L}$ of full row rank and any nonsingular matrix $\mathbf{K}$ of conformable dimensions,

$$\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = 0) = \mathrm{SS}(H_0\colon \mathbf{KL}\boldsymbol{\beta} = 0)$$

A Three-Factor Main-Effects Model

Consider a three-factor main-effects model involving the CLASS variables A, B, and C, as shown in Table 16.1.

Table 16.1 Three-Factor Main-Effects Model

Obs	*A*	*B*	*C*
1	1	2	1
2	1	1	2
3	2	1	3
4	2	2	2
5	2	2	2

The general form of an estimable function is shown in Table 16.2.

Table 16.2 General Form of an Estimable Function for Three-Factor Main-Effects Model

Parameter	Coefficient
μ (Intercept)	*L1*
A1	*L2*
A2	*L1* – *L2*
B1	*L4*
B2	*L1* – *L4*
C1	*L6*
C2	*L1* + *L2* – *L4* – 2 × *L6*
C3	–*L2* + *L4* + *L6*

Since only four symbols (*L1*, *L2*, *L4*, and *L6*) are involved, any testable hypothesis will have at most four degrees of freedom. If you form an **L** matrix with four linearly independent rows according to the preceding rules, then testing $\mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ is equivalent to testing that $\mathrm{E}[\mathbf{Y}]$ is uniformly 0. Symbolically,

$$\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = 0) = R(\mu, A, B, C)$$

In a main-effects model, the usual hypothesis of interest for a main effect is the equality of all the parameters. In this example, it is not possible to unambiguously test such a hypothesis because of confounding: any test for the equality of the parameters for any one of A, B, or C will necessarily involve the parameters for the other two effects. One way to proceed is to construct a maximum rank hypothesis (MRH) involving only the parameters of the main effect in question. This can be done using the general form of estimable functions. Note the following:

- To get an MRH involving only the parameters of A, the coefficients of **L** associated with μ, *B1*, *B2*, *C1*, *C2*, and *C3* must be equated to zero. Starting at the top of the general form, let *L1* = 0, then *L4* = 0, then *L6* = 0. If *C2* and *C3* are not to be involved, then *L2* must also be zero. Thus, *A1* – *A2* is not estimable; that is, the MRH involving only the A parameters has zero rank and $R(A \mid \mu, B, C) = 0$.
- To obtain the MRH involving only the B parameters, let *L1* = *L2* = *L6* = 0. But then to remove *C2* and *C3* from the comparison, *L4* must also be set to 0. Thus, *B1* – *B2* is not estimable and $R(B \mid \mu, A, C) = 0$.

- To obtain the MRH involving only the C parameters, let $L1 = L2 = L4 = 0$. Thus, the MRH involving only C parameters is

$$C1 - 2 \times C2 + C3 = K \qquad \text{(for any } K\text{)}$$

or any multiple of the left-hand side equal to K. Furthermore,

$$\text{SS}(H_0\text{: } C1 - 2 \times C2 + C3 = 0) = R(C \mid \mu, A, B)$$

A Multiple Regression Model

Suppose

$$\text{E}[Y] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

where the $\mathbf{X}'\mathbf{X}$ matrix has full rank. The general form of estimable functions is as shown in Table 16.3.

Table 16.3 General Form of Estimable Functions for a Multiple Regression Model When $\mathbf{X}'\mathbf{X}$ Matrix Is of Full Rank

Parameter	Coefficient
β_0	$L1$
β_1	$L2$
β_2	$L3$
β_3	$L4$

For example, to test the hypothesis that $\beta_2 = 0$, let $L1 = L2 = L4 = 0$ and let $L3 = 1$. Then $\text{SS}(\mathbf{L}\boldsymbol{\beta} = \mathbf{0}) = R(\beta_2 \mid \beta_0, \beta_1, \beta_3)$. In this full-rank case, all parameters, as well as any linear combination of parameters, are estimable.

Suppose, however, that $X3 = 2x_1 + 3x_2$. The general form of estimable functions is shown in Table 16.4.

Table 16.4 General Form of Estimable Functions for a Multiple Regression Model When $\mathbf{X}'\mathbf{X}$ Matrix Is Not of Full Rank

Parameter	Coefficient
β_0	$L1$
β_1	$L2$
β_2	$L3$
β_3	$2 \times L2 + 3 \times L3$

For this example, it is possible to test H_0: $\beta_0 = 0$. However, β_1, β_2, and β_3 are not jointly estimable; that is,

$$\begin{aligned} R(\beta_1 \mid \beta_0, \beta_2, \beta_3) &= 0 \\ R(\beta_2 \mid \beta_0, \beta_1, \beta_3) &= 0 \\ R(\beta_3 \mid \beta_0, \beta_1, \beta_2) &= 0 \end{aligned}$$

Estimable Functions

Type I SS and Estimable Functions

In PROC GLM, the Type I SS and the associated hypotheses they test are byproducts of the modified sweep operator used to compute a generalized g_2-inverse of $\mathbf{X}'\mathbf{X}$ and a solution to the normal equations. For the model $\mathrm{E}[Y] = x_1\beta_1 + x_2\beta_2 + x_3\beta_3$, the Type I SS for each effect are as follows:

Effect	Type I SS
x_1	$R(\beta_1)$
x_2	$R(\beta_2 \mid \beta_1)$
x_3	$R(\beta_3 \mid \beta_1, \beta_2)$

Note that some other SAS/STAT procedures compute Type I hypotheses by sweeping $\mathbf{X}'\mathbf{X}$ (for example, PROC MIXED and PROC GLIMMIX), but their test statistics are not necessarily equivalent to the results of using those procedures to fit models that contain successively more effects.

The Type I SS are model-order dependent; each effect is adjusted only for the preceding effects in the model.

There are numerous ways to obtain a Type I hypothesis matrix $\mathbf{L}$ for each effect. One way is to form the $\mathbf{X}'\mathbf{X}$ matrix and then reduce $\mathbf{X}'\mathbf{X}$ to an upper triangular matrix by row operations, skipping over any rows with a zero diagonal. The nonzero rows of the resulting matrix associated with x_1 provide an $\mathbf{L}$ such that

$$\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}) = R(\beta_1)$$

The nonzero rows of the resulting matrix associated with x_2 provide an $\mathbf{L}$ such that

$$\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}) = R(\beta_2 \mid \beta_1)$$

The last set of nonzero rows (associated with x_3) provide an $\mathbf{L}$ such that

$$\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}) = R(\beta_3 \mid \beta_1, \beta_2)$$

Another more formalized representation of Type I generating sets for x_1, x_2, and x_3, respectively, is

$$\begin{array}{rcl}
\mathbf{G}_1 & = & (\ \mathbf{X}_1'\mathbf{X}_1 \ \mid\ \mathbf{X}_1'\mathbf{X}_2 \ \mid\ \mathbf{X}_1'\mathbf{X}_3\ ) \\
\mathbf{G}_2 & = & (\ 0 \ \mid\ \mathbf{X}_2'\mathbf{M}_1\mathbf{X}_2 \ \mid\ \mathbf{X}_2'\mathbf{M}_1\mathbf{X}_3\ ) \\
\mathbf{G}_3 & = & (\ 0 \ \mid\ 0 \ \mid\ \mathbf{X}_3'\mathbf{M}_2\mathbf{X}_3\ )
\end{array}$$

where

$$\mathbf{M}_1 = \mathbf{I} - \mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-}\mathbf{X}_1'$$

and

$$\mathbf{M}_2 = \mathbf{M}_1 - \mathbf{M}_1\mathbf{X}_2(\mathbf{X}_2'\mathbf{M}_1\mathbf{X}_2)^{-}\mathbf{X}_2'\mathbf{M}_1$$

Using the Type I generating set $\mathbf{G}_2$ (for example), if an $\mathbf{L}$ is formed from linear combinations of the rows of $\mathbf{G}_2$ such that $\mathbf{L}$ is of full row rank and of the same row rank as $\mathbf{G}_2$, then $\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}) = R(\beta_2 \mid \beta_1)$.

In the GLM procedure, the Type I estimable functions displayed symbolically when the E1 option is requested are

$$\begin{aligned}
\mathbf{G}_1^* &= (\mathbf{X}_1'\mathbf{X}_1)^-\mathbf{G}_1 \\
\mathbf{G}_2^* &= (\mathbf{X}_2'\mathbf{M}_1\mathbf{X}_2)^-\mathbf{G}_2 \\
\mathbf{G}_3^* &= (\mathbf{X}_3'\mathbf{M}_2\mathbf{X}_3)^-\mathbf{G}_3
\end{aligned}$$

As can be seen from the nature of the generating sets $\mathbf{G}_1$, $\mathbf{G}_2$, and $\mathbf{G}_3$, only the Type I estimable functions for β_3 are guaranteed not to involve the β_1 and β_2 parameters. The Type I hypothesis for β_2 can (and often does) involve β_3 parameters, and likewise the Type I hypothesis for β_1 often involves β_2 and β_3 parameters.

There are, however, a number of models for which the Type I hypotheses are considered appropriate. These are as follows:

- balanced ANOVA models specified in proper sequence (that is, interactions do not precede main effects in the MODEL statement and so forth)
- purely nested models (specified in the proper sequence)
- polynomial regression models (in the proper sequence)

Type II SS and Estimable Functions

For main-effects models and regression models, the general form of estimable functions can be manipulated to provide tests of hypotheses involving only the parameters of the effect in question. The same result can also be obtained by entering each effect in turn as the last effect in the model and obtaining the Type I SS for that effect. These are the *Type II SS*. Using a modified reversible sweep operator, it is possible to obtain the Type II SS without actually refitting the model.

Thus, the **Type II SS correspond to the R notation in which each effect is adjusted for all other appropriate effects**. For a regression model such as

$$\mathrm{E}[Y] = x_1\beta_1 + x_2\beta_2 + x_3\beta_3$$

the Type II SS correspond to

Effect	Type II SS
x_1	$R(\beta_1 \mid \beta_2, \beta_3)$
x_2	$R(\beta_2 \mid \beta_1, \beta_3)$
x_3	$R(\beta_3 \mid \beta_1, \beta_2)$

For a main-effects model (A, B, and C as classification variables), the Type II SS correspond to

Effect	Type II SS
A	$R(A \mid B, C)$
B	$R(B \mid A, C)$
C	$R(C \mid A, B)$

As the discussion in the section "A Three-Factor Main-Effects Model" on page 280 indicates, for regression and main-effects models the Type II SS provide an MRH for each effect that does not involve the parameters of the other effects.

In order to see what effects are appropriate to adjust for in computing Type II estimable functions, note that for models involving interactions and nested effects, in the absence of a priori parametric restrictions, it is not possible to obtain a test of a hypothesis for a main effect free of parameters of higher-level interactions effects with which the main effect is involved. It is reasonable to assume, then, that any test of a hypothesis concerning an effect should involve the parameters of that effect and only those other parameters with which that effect is involved. The concept of effect containment helps to define this involvement.

Contained Effect

Given two effects *F1* and *F2*, *F1* is said to be *contained in F2* provided that the following two conditions are met:

- Both effects involve the same continuous variables (if any).
- *F2* has more CLASS variables than *F1* does, and if *F1* has CLASS variables, they all appear in *F2*.

Note that the intercept effect μ is contained in all pure CLASS effects, but it is not contained in any effect involving a continuous variable. No effect is contained by μ.

Type II, Type III, and Type IV estimable functions rely on this definition, and they all have one thing in common: the estimable functions involving an effect *F1* also involve the parameters of all effects that contain *F1*, and they do not involve the parameters of effects that do not contain *F1* (other than *F1*).

Hypothesis Matrix for Type II Estimable Functions

The Type II estimable functions for an effect *F1* have an $\mathbf{L}$ (before reduction to full row rank) of the following form:

- All columns of $\mathbf{L}$ associated with effects not containing *F1* (except *F1*) are zero.
- The submatrix of $\mathbf{L}$ associated with effect *F1* is $(\mathbf{X}_1'\mathbf{M}\mathbf{X}_1)^-(\mathbf{X}_1'\mathbf{M}\mathbf{X}_1)$.
- Each of the remaining submatrices of $\mathbf{L}$ associated with an effect *F2* that contains *F1* is $(\mathbf{X}_1'\mathbf{M}\mathbf{X}_1)^-(\mathbf{X}_1'\mathbf{M}\mathbf{X}_2)$.

In these submatrices,

$$\begin{aligned}
\mathbf{X}_0 &= \text{the columns of } \mathbf{X} \text{ whose associated effects do not contain } F1 \\
\mathbf{X}_1 &= \text{the columns of } \mathbf{X} \text{ associated with } F1 \\
\mathbf{X}_2 &= \text{the columns of } \mathbf{X} \text{ associated with an } F2 \text{ effect that contains } F1 \\
\mathbf{M} &= \mathbf{I} - \mathbf{X}_0(\mathbf{X}_0'\mathbf{X}_0)^-\mathbf{X}_0'
\end{aligned}$$

For the model

```
class A B;
model Y = A B A*B;
```

the Type II SS correspond to

$$R(A \mid \mu, B), \quad R(B \mid \mu, A), \quad R(A * B \mid \mu, A, B)$$

for effects A, B, and $A * B$, respectively. For the model

```
class A B C;
model Y =  A B(A) C(A B);
```

the Type II SS correspond to

$$R(A \mid \mu), \quad R(B(A) \mid \mu, A), \quad R(C(AB) \mid \mu, A, B(A))$$

for effects A, $B(A)$ and $C(A\ B)$, respectively. For the model

```
model Y = x x*x;
```

the Type II SS correspond to

$$R(X \mid \mu, X * X) \text{ and } R(X * X \mid \mu, X)$$

for x and $x * x$, respectively.

Note that, as in the situation for Type I tests, PROC MIXED and PROC GLIMMIX compute Type I hypotheses by sweeping $\mathbf{X}'\mathbf{X}$, but their test statistics are not necessarily equivalent to the results of sequentially fitting with those procedures models that contain successively more effects; while PROC TRANSREG computes tests labeled as being Type II by leaving out each effect in turn, but the specific linear hypotheses associated with these tests might not be precisely the same as the ones derived from successively sweeping $\mathbf{X}'\mathbf{X}$.

Example of Type II Estimable Functions

For a 2×2 factorial with w observations per cell, the general form of estimable functions is shown in Table 16.5. Any nonzero values for $L2$, $L4$, and $L6$ can be used to construct $\mathbf{L}$ vectors for computing the Type II SS for A, B, and $A * B$, respectively.

Table 16.5 General Form of Estimable Functions for 2×2 Factorial

Effect	Coefficient
μ	$L1$
$A1$	$L2$
$A2$	$L1 - L2$
$B1$	$L4$
$B2$	$L1 - L4$
$AB11$	$L6$
$AB12$	$L2 - L6$
$AB21$	$L4 - L6$
$AB22$	$L1 - L2 - L4 + L6$

For a balanced 2×2 factorial with the same number of observations in every cell, the Type II estimable functions are shown in Table 16.6.

Table 16.6 Type II Estimable Functions for Balanced 2×2 Factorial

	Coefficients for Effect		
Effect	***A***	***B***	***A * B***
μ	0	0	0
A1	*L2*	0	0
A2	–*L2*	0	0
B1	0	*L4*	0
B2	0	–*L4*	0
AB11	$0.5 \times L2$	$0.5 \times L4$	*L6*
AB12	$0.5 \times L2$	$-0.5 \times L4$	–*L6*
AB21	$-0.5 \times L2$	$0.5 \times L4$	–*L6*
AB22	$-0.5 \times L2$	$-0.5 \times L4$	*L6*

Now consider an unbalanced 2×2 factorial with two observations in every cell except the *AB22* cell, which contains only one observation. The general form of estimable functions is the same as if it were balanced, since the same effects are still estimable. However, the Type II estimable functions for *A* and *B* are not the same as they were for the balanced design. The Type II estimable functions for this unbalanced 2×2 factorial are shown in Table 16.7.

Table 16.7 Type II Estimable Functions for Unbalanced 2×2 Factorial

	Coefficients for Effect		
Effect	***A***	***B***	***A * B***
μ	0	0	0
A1	*L2*	0	0
A2	–*L2*	0	0
B1	0	*L4*	0
B2	0	–*L4*	0
AB11	$0.6 \times L2$	$0.6 \times L4$	*L6*
AB12	$0.4 \times L2$	$-0.6 \times L4$	–*L6*
AB21	$-0.6 \times L2$	$0.4 \times L4$	–*L6*
AB22	$-0.4 \times L2$	$-0.4 \times L4$	*L6*

By comparing the hypothesis being tested in the balanced case to the hypothesis being tested in the unbalanced case for effects *A* and *B*, you can note that the Type II hypotheses for *A* and *B* are dependent on the cell frequencies in the design. For unbalanced designs in which the cell frequencies are not proportional to the background population, the Type II hypotheses for effects that are contained in other effects are of questionable value.

However, if an effect is not contained in any other effect, the Type II hypothesis for that effect is an MRH

that does not involve any parameters except those associated with the effect in question.

Thus, Type II SS are appropriate for the following models:

- any balanced model
- any main-effects model
- any pure regression model
- an effect not contained in any other effect (regardless of the model)

In addition to the preceding models, Type II SS are generally accepted by most statisticians for purely nested models.

Type III and IV SS and Estimable Functions

When an effect is contained in another effect, the Type II hypotheses for that effect are dependent on the cell frequencies. The philosophy behind both the Type III and Type IV hypotheses is that the hypotheses tested for any given effect should be the same for all designs with the same general form of estimable functions.

To demonstrate this concept, recall the hypotheses being tested by the Type II SS in the balanced 2×2 factorial shown in Table 16.6. Those hypotheses are precisely the ones that the Type III and Type IV hypotheses employ for all 2×2 factorials that have at least one observation per cell. The Type III and Type IV hypotheses for a design without missing cells usually differ from the hypothesis employed for the same design with missing cells since the general form of estimable functions usually differs.

Many SAS/STAT procedures can perform tests of Type III hypotheses, but only PROC GLM offers Type IV tests as well.

Type III Estimable Functions

Type III hypotheses are constructed by working directly with the general form of estimable functions. The following steps are used to construct a hypothesis for an effect *F1*:

1. For every effect in the model except *F1* and those effects that contain *F1*, equate the coefficients in the general form of estimable functions to zero.

   If *F1* is not contained in any other effect, this step defines the Type III hypothesis (as well as the Type II and Type IV hypotheses). If *F1* is contained in other effects, go on to step 2. (See the section "Type II SS and Estimable Functions" on page 283 for a definition of when effect *F1* is contained in another effect.)

2. If necessary, equate new symbols to compound expressions in the *F1* block in order to obtain the simplest form for the *F1* coefficients.

3. Equate all symbolic coefficients outside the *F1* block to a linear function of the symbols in the *F1* block in order to make the *F1* hypothesis orthogonal to hypotheses associated with effects that contain *F1*.

By once again observing the Type II hypotheses being tested in the balanced 2 × 2 factorial, it is possible to verify that the *A* and *A* * *B* hypotheses are orthogonal and also that the *B* and *A* * *B* hypotheses are orthogonal. This principle of orthogonality between an effect and any effect that contains it holds for all balanced designs. Thus, construction of Type III hypotheses for any design is a logical extension of a process that is used for balanced designs.

The Type III hypotheses are precisely the hypotheses being tested by programs that reparameterize using the usual assumptions (for example, constraining all parameters for an effect to sum to zero). When no missing cells exist in a factorial model, Type III SS coincide with Yates' weighted squares-of-means technique. When cells are missing in factorial models, the Type III SS coincide with those discussed in Harvey (1960) and Henderson (1953).

The following discussion illustrates the construction of Type III estimable functions for a 2 × 2 factorial with no missing cells.

To obtain the *A* * *B* interaction hypothesis, start with the general form and equate the coefficients for effects μ, *A*, and *B* to zero, as shown in Table 16.8.

Table 16.8 Type III Hypothesis for *A* * *B* Interaction

Effect	General Form	*L1* = *L2* = *L4* = 0
μ	*L1*	0
A1	*L2*	0
A2	*L1* – *L2*	0
B1	*L4*	0
B2	*L1* – *L4*	0
AB11	*L6*	*L6*
AB12	*L2* – *L6*	–*L6*
AB21	*L4* – *L6*	–*L6*
AB22	*L1* – *L2* – *L4* + *L6*	*L6*

The last column in Table 16.8 represents the form of the MRH for *A* * *B*.

To obtain the Type III hypothesis for *A*, first start with the general form and equate the coefficients for effects μ and *B* to zero (let *L1* = *L4* = 0). Next let *L6* = *K* × *L2*, and find the value of K that makes the *A* hypothesis orthogonal to the *A* * *B* hypothesis. In this case, *K* = 0.5. Each of these steps is shown in Table 16.9.

In Table 16.9, the fourth column (under *L6* = *K* × *L2*) represents the form of all estimable functions not involving μ, *B1*, or *B2*. The prime difference between the Type II and Type III hypotheses for *A* is the way *K* is determined. Type II chooses *K* as a function of the cell frequencies, whereas Type III chooses *K* such that the estimable functions for *A* are orthogonal to the estimable functions for *A* * *B*.

Table 16.9 Type III Hypothesis for *A*

Effect	General Form	*L1* = *L4* = 0	*L6* = *K* × *L2*	*K*= 0.5
μ	*L1*	0	0	0
A1	*L2*	*L2*	*L2*	*L2*
A2	*L1* – *L2*	–*L2*	–*L2*	–*L2*

Table 16.9 *continued*

Effect	General Form	$L1 = L4 = 0$	$L6 = K \times L2$	$K = 0.5$
$B1$	$L4$	0	0	0
$B2$	$L1 - L4$	0	0	0
$AB11$	$L6$	$L6$	$K \times L2$	$0.5 \times L2$
$AB12$	$L2 - L6$	$L2 - L6$	$(1 - K) \times L2$	$0.5 \times L2$
$AB21$	$L4 - L6$	$-L6$	$-K \times L2$	$-0.5 \times L2$
$AB22$	$L1 - L2 - L4 + L6$	$-L2 + L6$	$-(1 - K) \times L2$	$-0.5 \times L2$

An example of Type III estimable functions in a 3×3 factorial with unequal cell frequencies and missing diagonals is given in Table 16.10 (N_1 through N_6 represent the nonzero cell frequencies).

Table 16.10 3×3 Factorial Design with Unequal Cell Frequencies and Missing Diagonals

		B		
		1	2	3
A	1		N_1	N_2
	2	N_3		N_4
	3	N_5	N_6	

For any nonzero values of N_1 through N_6, the Type III estimable functions for each effect are shown in Table 16.11.

Table 16.11 Type III Estimable Functions for 3×3 Factorial Design with Unequal Cell Frequencies and Missing Diagonals

Effect	A	B	$A * B$
μ	0	0	0
$A1$	$L2$	0	0
$A2$	$L3$	0	0
$A3$	$-L2 - L3$	0	0
$B1$	0	$L5$	0
$B2$	0	$L6$	0
$B3$	0	$-L5 - L6$	0
$AB12$	$0.667 \times L2 + 0.333 \times L3$	$0.333 \times L5 + 0.667 \times L6$	$L8$
$AB13$	$0.333 \times L2 - 0.333 \times L3$	$-0.333 \times L5 - 0.667 \times L6$	$-L8$
$AB21$	$0.333 \times L2 + 0.667 \times L3$	$0.667 \times L5 + 0.333 \times L6$	$-L8$
$AB23$	$-0.333 \times L2 + 0.333 \times L3$	$-0.667 \times L5 - 0.333 \times L6$	$L8$
$AB31$	$-0.333 \times L2 - 0.667 \times L3$	$0.333 \times L5 - 0.333 \times L6$	$L8$
$AB32$	$-0.667 \times L2 - 0.333 \times L3$	$-0.333 \times L5 + 0.333 \times L6$	$-L8$

Type IV Estimable Functions

By once again looking at the Type II hypotheses being tested in the balanced 2×2 factorial (see Table 16.6), you can see another characteristic of the hypotheses employed for balanced designs: the coefficients of lower-order effects are averaged across each higher-level effect involving the same subscripts. For example, in the A hypothesis, the coefficients of *AB11* and *AB12* are equal to one-half the coefficient of *A1*, and the coefficients of *AB21* and *AB22* are equal to one-half the coefficient of *A2*. With this in mind, the basic concept used to construct Type IV hypotheses is that the coefficients of any effect, say *F1*, are distributed equitably across higher-level effects that contain *F1*. When missing cells occur, this same general philosophy is adhered to, but care must be taken in the way the distributive concept is applied.

Construction of Type IV hypotheses begins as does the construction of the Type III hypotheses. That is, for an effect *F1*, equate to zero all coefficients in the general form that do not belong to *F1* or to any other effect containing *F1*. If *F1* is not contained in any other effect, then the Type IV hypothesis (and Type II and III) has been found. If *F1* is contained in other effects, then simplify, if necessary, the coefficients associated with *F1* so that they are all free coefficients or functions of other free coefficients in the *F1* block.

To illustrate the method of resolving the free coefficients outside the *F1* block, suppose that you are interested in the estimable functions for an effect A and that A is contained in AB, AC, and ABC. (In other words, the main effects in the model are A, B, and C.)

With missing cells, the coefficients of intermediate effects (here they are AB and AC) do not always have an equal distribution of the lower-order coefficients, so the coefficients of the highest-order effects are determined first (here it is ABC). Once the highest-order coefficients are determined, the coefficients of intermediate effects are automatically determined.

The following process is performed for each free coefficient of A in turn. The resulting symbolic vectors are then added together to give the Type IV estimable functions for A.

1. Select a free coefficient of A, and set all other free coefficients of A to zero.
2. If any of the levels of A have zero as a coefficient, equate all of the coefficients of higher-level effects involving that level of A to zero. This step alone usually resolves most of the free coefficients remaining.
3. Check to see if any higher-level coefficients are now zero when the coefficient of the associated level of A is not zero. If this situation occurs, the Type IV estimable functions for A are not unique.
4. For each level of A in turn, if the A coefficient for that level is nonzero, count the number of times that level occurs in the higher-level effect. Then equate each of the higher-level coefficients to the coefficient of that level of A divided by the count.

An example of a 3×3 factorial with four missing cells (N_1 through N_5 represent positive cell frequencies) is shown in Table 16.12.

Table 16.12 3×3 Factorial Design with Four Missing Cells

		B		
		1	2	3
A	1	N_1	N_2	
	2	N_3	N_4	
	3			N_5

The Type IV estimable functions are shown in Table 16.13.

Table 16.13 Type IV Estimable Functions for 3 × 3 Factorial Design with Four Missing Cells

Effect	*A*	*B*	*A* * *B*
μ	0	0	0
A1	–*L3*	0	0
A2	*L3*	0	0
A3	0	0	0
B1	0	*L5*	0
B2	0	–*L5*	0
B3	0	0	0
AB11	–0.5 × *L3*	0.5 × *L5*	*L8*
AB12	–0.5 × *L3*	–0.5 × *L5*	–*L8*
AB21	0.5 × *L3*	0.5 × *L5*	–*L8*
AB22	0.5 × *L3*	–0.5 × *L5*	*L8*
AB33	0	0	0

A Comparison of Type III and Type IV Hypotheses

For the vast majority of designs, Type III and Type IV hypotheses for a given effect are the same. Specifically, they are the same for any effect *F1* that is not contained in other effects for any design (with or without missing cells). For factorial designs with no missing cells, the Type III and Type IV hypotheses coincide for all effects. When there are missing cells, the hypotheses can differ. By using the GLM procedure, you can study the differences in the hypotheses and then decide on the appropriateness of the hypotheses for a particular model.

The Type III hypotheses for three-factor and higher completely nested designs with unequal *N*s in the lowest level differ from the Type II hypotheses; however, the Type IV hypotheses do correspond to the Type II hypotheses in this case.

When missing cells occur in a design, the Type IV hypotheses might not be unique. If this occurs in PROC GLM, you are notified, and you might need to consider defining your own specific comparisons.

References

Freund, R. J., Littell, R. C., and Spector, P. C. (1991). *SAS System for Linear Models*. Cary, NC: SAS Institute Inc.

Goodnight, J. H. (1978). *Tests of Hypotheses in Fixed-Effects Linear Models*. Technical Report R-101, SAS Institute Inc., Cary, NC.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Harvey, W. R. (1960). *Least-Squares Analysis of Data with Unequal Subclass Frequencies*. Technical Report ARS 20-8, Agriculture Research Service, U.S. Department of Agriculture.

Henderson, C. R. (1953). "Estimation of Variance and Covariance Components." *Biometrics* 9:226–252.

Milliken, G. A., and Johnson, D. E. (1984). *Designed Experiments.* Vol. 1 of Analysis of Messy Data. Belmont, CA: Lifetime Learning Publications.

Chapter 17
Introduction to Nonparametric Analysis

Contents

Overview: Nonparametric Analysis . . . **293**
- Testing for Normality . . . 294
- Comparing Distributions . . . 294

One-Sample Tests . . . **294**
Two-Sample Tests . . . **295**
- Comparing Two Independent Samples . . . 295
- Comparing Two Related Samples . . . 296

Tests for *k* Samples . . . **297**
- Comparing *k* Independent Samples . . . 297
- Comparing *k* Dependent Samples . . . 298

Measures of Correlation and Associated Tests . . . **298**
Obtaining Ranks . . . **298**
Kernel Density Estimation . . . **299**
References . . . **299**

Overview: Nonparametric Analysis

In statistical inference, or hypothesis testing, the traditional tests are called *parametric tests* because they depend on the specification of a probability distribution (such as the normal) except for a set of free parameters. Parametric tests are said to depend on distributional assumptions. *Nonparametric tests*, on the other hand, do not require any strict distributional assumptions. Even if the data are distributed normally, nonparametric methods are often almost as powerful as parametric methods.

Many nonparametric methods analyze the ranks of a variable rather than the original values. Procedures such as PROC NPAR1WAY calculate the ranks for you and then perform appropriate nonparametric tests. However, there are some situations in which you use a procedure such as PROC RANK to calculate ranks and then use another procedure to perform the appropriate test. See the section "Obtaining Ranks" on page 298 for details.

Although the NPAR1WAY procedure is specifically targeted for nonparametric analysis, many other procedures also perform nonparametric analyses. Some general references on nonparametrics include Hollander and Wolfe (1999); Conover (1999); Gibbons and Chakraborti (2010); Hettmansperger (1984); Randles and Wolfe (1979); Lehmann and D'Abrera (2006).

Testing for Normality

Many parametric tests assume an underlying normal distribution for the population. If your data do not meet this assumption, you might prefer to use a nonparametric analysis.

Base SAS software provides several tests for normality in the UNIVARIATE procedure. Depending on your sample size, PROC UNIVARIATE performs the Kolmogorov-Smirnov, Shapiro-Wilk, Anderson-Darling, and Cramér-von Mises tests. For more information, see the chapter "The UNIVARIATE Procedure" in the *Base SAS Procedures Guide: Statistical Procedures*.

Comparing Distributions

To test the hypothesis that two or more groups of observations have identical distributions, use the NPAR1WAY procedure, which provides empirical distribution function (EDF) statistics. The procedure calculates the Kolmogorov-Smirnov test, the Cramér-von Mises test, and, when the data are classified into only two samples, the Kuiper test. Exact *p*-values are available for the two-sample Kolmogorov-Smirnov test. To obtain these tests, use the EDF option in the PROC NPAR1WAY statement. See Chapter 90, "The NPAR1WAY Procedure," for details.

One-Sample Tests

Base SAS software provides two one-sample tests in the UNIVARIATE procedure: a sign test and the Wilcoxon signed rank test. Both tests are designed for situations where you want to make an inference about the location (median) of a population. For example, suppose you want to test whether the median resting pulse rate of marathon runners differs from a specified value.

By default, both of these tests examine the hypothesis that the median of the population from which the sample is drawn is equal to a specified value, which is zero by default. The Wilcoxon signed rank test requires that the distribution be symmetric; the sign test does not require this assumption. These tests can also be used for the case of two related samples; see the section "Comparing Two Independent Samples" on page 295 for more information.

These two tests are automatically provided by the UNIVARIATE procedure. For details, formulas, and examples, see the chapter "The UNIVARIATE Procedure" in the *Base SAS Procedures Guide: Statistical Procedures*.

Two-Sample Tests

This section describes tests appropriate for two independent samples (for example, two groups of subjects given different treatments) and for two related samples (for example, before-and-after measurements on a single group of subjects). Related samples are also referred to as paired samples or matched pairs.

Comparing Two Independent Samples

SAS/STAT software provides several nonparametric tests for location and scale differences for two independent samples.

When you perform these tests, your data should consist of a random sample of observations from two different populations. Your goal is to compare either the location parameters (medians) or the scale parameters of the two populations. For example, suppose your data consist of the number of days in the hospital for two groups of patients: those who received a standard surgical procedure and those who received a new, experimental surgical procedure. These patients are a random sample from the population of patients who have received the two types of surgery. Your goal is to decide whether the median hospital stays differ for the two populations.

Tests in the NPAR1WAY Procedure

The NPAR1WAY procedure provides the following location tests: Wilcoxon rank sum test (Mann-Whitney U test), median test, Savage test, and Van der Waerden (normal scores) test. Note that the Wilcoxon rank sum test can also be obtained from the FREQ procedure. PROC NPAR1WAY provides Hodges-Lehmann estimation of the location shift between two samples, including asymptotic (Moses) and exact confidence limits.

In addition, PROC NPAR1WAY produces the following tests for scale differences: Siegel-Tukey test, Ansari-Bradley test, Klotz test, and Mood test. PROC NPAR1WAY also provides the Conover test, which can be used to test for differences in both location and scale.

Additionally, PROC NPAR1WAY provides tests that use the input data observations as scores, enabling you to produce a wide variety of tests. You can construct any scores for your data with the DATA step, and then PROC NPAR1WAY computes the corresponding linear rank test. You can directly analyze the raw data this way, producing the permutation test known as Pitman's test.

When data are sparse, skewed, or heavily tied, the usual asymptotic tests might not be appropriate. In these situations, exact tests might be suitable for analyzing your data. The NPAR1WAY procedure can produce exact p-values for all of the two-sample tests for location and scale differences.

See Chapter 90, "The NPAR1WAY Procedure," for details, formulas, and examples of these tests.

Tests in the FREQ Procedure

The FREQ procedure provides nonparametric tests that compare the location of two groups and that test for independence between two variables.

The situation in which you want to compare the location of two groups of observations corresponds to a table with two rows. In this case, the asymptotic Wilcoxon rank sum test can be obtained by using SCORES=RANK in the TABLES statement and by looking at either of the following:

- the Mantel-Haenszel statistic in the list of tests for no association. This is labeled as "Mantel Haenszel Chi-Square," and PROC FREQ displays the statistic, the degrees of freedom, and the p-value. To obtain this statistic, specify the CHISQ option in the TABLES statement.
- the CMH statistic 2 in the section on Cochran-Mantel-Haenszel statistics. PROC FREQ displays the statistic, the degrees of freedom, and the p-value. To obtain this statistic, specify the CMH2 option in the TABLES statement.

When you test for independence, the question being answered is whether the two variables of interest are related in some way. For example, you might want to know if student scores on a standard test are related to whether students attended a public or private school. One way to think of this situation is to consider the data as a two-way table; the hypothesis of interest is whether the rows and columns are independent. In the preceding example, the groups of students would form the two rows, and the scores would form the columns. The special case of a two-category response (Pass/Fail) leads to a 2×2 table; the case of more than two categories for the response (A/B/C/D/F) leads to a $2 \times c$ table, where c is the number of response categories.

For testing whether two variables are independent, PROC FREQ provides Fisher's exact test. For a 2×2 table, PROC FREQ automatically provides Fisher's exact test when you specify the CHISQ option in the TABLES statement. For a $2 \times c$ table, use the FISHER option in the EXACT statement to obtain the test.

See Chapter 47, "The FREQ Procedure," for details, formulas, and examples of these tests.

Comparing Two Related Samples

SAS/STAT software provides the following nonparametric tests for comparing the locations of two related samples:

- Wilcoxon signed rank test
- sign test
- McNemar's test

The first two tests are available in the UNIVARIATE procedure, and the last test is available in the FREQ procedure. When you perform these tests, your data should consist of pairs of measurements for a random sample from a single population. For example, suppose your data consist of SAT scores for students before and after attending a course on how to prepare for the SAT. The pairs of measurements are the scores before and after the course, and the students should be a random sample of students who attended the course. Your goal in analysis is to decide whether the median change in scores is significantly different from zero.

Tests in the UNIVARIATE Procedure

By default, PROC UNIVARIATE performs a Wilcoxon signed rank test and a sign test. To use these tests on two related samples, perform the following steps:

1. In the DATA step, create a new variable that contains the differences between the two related variables.
2. Run PROC UNIVARIATE, using the new variable in the VAR statement.

See the chapter "The UNIVARIATE Procedure" in the *Base SAS Procedures Guide: Statistical Procedures* for details and examples of these tests.

Tests in the FREQ Procedure

The FREQ procedure can be used to obtain McNemar's test, which is simply another special case of a Cochran-Mantel-Haenszel statistic (and also of the sign test). The AGREE option in the TABLES statement produces this test for 2×2 tables, and exact p-values are also available for this test. See Chapter 47, "The FREQ Procedure," for more information.

Tests for k Samples

Comparing k Independent Samples

One goal in comparing k independent samples is to determine whether the location parameters (medians) of the populations are different. Another goal is to determine whether the scale parameters for the populations are different. For example, suppose new employees are randomly assigned to one of three training programs. At the end of the program, the employees are given a standard test that provides a rating score of their job ability. The goal of analysis is to compare the median scores for the three groups and decide whether the differences are real or due to chance alone.

To compare k independent samples, either the NPAR1WAY or the FREQ procedure provides a Kruskal-Wallis test. PROC NPAR1WAY also provides the Savage, median, and Van der Waerden (normal scores) tests. In addition, PROC NPAR1WAY produces the following tests for scale differences: Siegel-Tukey test, Ansari-Bradley test, Klotz test, and Mood test. PROC NPAR1WAY also provides the Conover test, which can be used to test for differences in both location and scale. Note that you can obtain exact p-values for all of these tests.

Additionally, you can specify the SCORES=DATA option to use the input data observations as scores. This enables you to produce a very wide variety of tests. You can construct any scores for your data with the DATA step, and then PROC NPAR1WAY computes the corresponding linear rank and one-way ANOVA tests. You can also analyze the raw data with the SCORES=DATA option; for two-sample data, this permutation test is known as Pitman's test.

See Chapter 90, "The NPAR1WAY Procedure," for details, formulas, and examples.

To produce a Kruskal-Wallis test in the FREQ procedure, use SCORES=RANK and the CMH2 option in the TABLES statement. Then, look at the second Cochran-Mantel-Haenszel statistic (labeled "Row Mean Scores Differ") to obtain the Kruskal-Wallis test. The FREQ procedure also provides the Jonckheere-Terpstra test, which is more powerful than the Kruskal-Wallis test for comparing k samples against ordered alternatives. The exact test is also available. In addition, you can obtain a ridit analysis, developed by Bross (1958), by specifying SCORES=RIDIT or SCORES=MODRIDIT in the TABLES statement in the FREQ procedure. See Chapter 47, "The FREQ Procedure," for more information.

Comparing k Dependent Samples

Friedman's test enables you to compare the locations of three or more dependent samples. You can obtain Friedman's chi-square with the FREQ procedure by using the CMH2 option with SCORES=RANK and by looking at the second CMH statistic in the output. For an example, see Chapter 47, "The FREQ Procedure," which also contains formulas and other details about the CMH statistics. For a discussion of how to use the RANK and GLM procedures to obtain Friedman's test, see Ipe (1987).

Measures of Correlation and Associated Tests

The CORR procedure in Base SAS software provides several nonparametric measures of association and associated tests. It computes Spearman's rank-order correlation, Kendall's tau-*b*, and Hoeffding's measure of dependence, and it provides tests for each of these statistics. PROC CORR also computes Spearman's partial rank-order correlation and Kendall's partial tau-*b*. Finally, PROC CORR computes Cronbach's coefficient alpha for raw and standardized variables. This statistic can be used to estimate the reliability coefficient. For a general discussion of correlations, formulas, interpretation, and examples, see the chapter "The CORR Procedure" in the *Base SAS Procedures Guide: Statistical Procedures*.

The FREQ procedure also provides some nonparametric measures of association: gamma, Kendall's tau-*b*, Stuart's tau-*c*, Somers' *D*, and the Spearman rank correlation. The output includes the measure, the asymptotic standard error, confidence limits, and the asymptotic test that the measure equals zero. Exact tests are also available for some of these measures. For more information, see Chapter 47, "The FREQ Procedure."

Obtaining Ranks

The primary procedure for obtaining ranks is the RANK procedure in Base SAS software. Note that the PRINQUAL and TRANSREG procedures also provide rank transformations. With all three of these procedures, you can create an output data set and use it as input to another SAS/STAT procedure or to the IML procedure. For more information, see the chapter "The RANK Procedure" in the *Base SAS Procedures Guide*. Also see Chapter 99, "The PRINQUAL Procedure," and Chapter 126, "The TRANSREG Procedure."

In addition, you can specify SCORES=RANK in the TABLES statement in the FREQ procedure. PROC FREQ then uses ranks to perform the analyses requested and generates nonparametric analyses.

For more discussion of the rank transform, see Iman and Conover (1979); Conover and Iman (1981); Hora and Conover (1984); Iman, Hora, and Conover (1984); Hora and Iman (1988); Iman (1988).

Kernel Density Estimation

The KDE procedure performs either univariate or bivariate kernel density estimation. Statistical *density estimation* involves approximating a hypothesized probability density function from observed data. *Kernel density estimation* is a nonparametric technique for density estimation in which a known density function (the kernel) is averaged across the observed data points to create a smooth approximation.

PROC KDE uses a Gaussian density as the kernel, and its assumed variance determines the smoothness of the resulting estimate. PROC KDE outputs the kernel density estimate to a SAS data set, which you can then use with other procedures for plotting or analysis. PROC KDE also computes a variety of common statistics, including estimates of the percentiles of the hypothesized probability density function.

For more information, see Chapter 73, "The KDE Procedure."

References

Agresti, A. (2007). *An Introduction to Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Bross, I. D. J. (1958). "How to Use Ridit Analysis." *Biometrics* 14:18–38.

Conover, W. J. (1999). *Practical Nonparametric Statistics*. 3rd ed. New York: John Wiley & Sons.

Conover, W. J., and Iman, R. L. (1981). "Rank Transformations as a Bridge between Parametric and Nonparametric Statistics." *American Statistician* 35:124–129.

Gibbons, J. D., and Chakraborti, S. (2010). *Nonparametric Statistical Inference*. 5th ed. New York: Chapman & Hall.

Hajek, J. (1969). *A Course in Nonparametric Statistics*. San Francisco: Holden-Day.

Hettmansperger, T. P. (1984). *Statistical Inference Based on Ranks*. New York: John Wiley & Sons.

Hollander, M., and Wolfe, D. A. (1999). *Nonparametric Statistical Methods*. 2nd ed. New York: John Wiley & Sons.

Hora, S. C., and Conover, W. J. (1984). "The *F* Statistic in the Two-Way Layout with Rank-Score Transformed Data." *Journal of the American Statistical Association* 79:668–673.

Hora, S. C., and Iman, R. L. (1988). "Asymptotic Relative Efficiencies of the Rank-Transformation Procedure in Randomized Complete Block Designs." *Journal of the American Statistical Association* 83:462–470.

Iman, R. L. (1988). "The Analysis of Complete Blocks Using Methods Based on Ranks." In *Proceedings of the Thirteenth Annual SAS Users Group International Conference*, 970–978. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI88/Sugi-13-196%20Iman.pdf.

Iman, R. L., and Conover, W. J. (1979). "The Use of the Rank Transform in Regression." *Technometrics* 21:499–509.

Iman, R. L., Hora, S. C., and Conover, W. J. (1984). "Comparison of Asymptotically Distribution-Free Procedures for the Analysis of Complete Blocks." *Journal of the American Statistical Association* 79:674–685.

Ipe, D. (1987). "Performing the Friedman Test and the Associated Multiple Comparison Test Using PROC GLM." In *Proceedings of the Twelfth Annual SAS Users Group International Conference*, 1146–1148. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI87/Sugi-12-209%20Ipe.pdf.

Lehmann, E. L., and D'Abrera, H. J. M. (2006). *Nonparametrics: Statistical Methods Based on Ranks*. Rev. ed. New York: Springer.

Randles, R. H., and Wolfe, D. A. (1979). *Introduction to the Theory of Nonparametric Statistics*. New York: John Wiley & Sons.

Chapter 18
Introduction to Structural Equation Modeling with Latent Variables

Contents

Overview of Structural Equation Modeling with Latent Variables **302**
Testing Covariance Patterns **304**
Testing Built-In Covariance Patterns in PROC CALIS 307
Direct and Implied Covariance Patterns 308
Regression with Measurement Errors **308**
Simple Linear Regression 309
Errors-in-Variables Regression 310
Regression with Measurement Errors in X and Y 314
Model Identification **317**
Illustration of Model Identification: Spleen Data 318
Path Diagrams and Path Analysis **323**
A Simplified Path Diagram for the Spleen Data 325
Producing Path Diagrams from the CALIS Procedure 327
Some Measurement Models **330**
H4: Full Measurement Model for Lord Data 331
H3: Congeneric (One-Factor) Model for Lord Data 335
H2: Two-Factor Model with Parallel Tests for Lord Data 337
H1: One-Factor Model with Parallel Tests for Lord Data 340
The FACTOR and RAM Modeling Languages **342**
Specifying the Full Measurement Model (H4) by the FACTOR Modeling Language: Lord Data 342
Specifying the Parallel Tests Model (H2) by the FACTOR Modeling Language: Lord Data 345
Specifying the Parallel Tests Model (H2) by the RAM Modeling Language: Lord Data 347
A Combined Measurement-Structural Model **349**
Career Aspiration (Analysis 1) 350
Career Aspiration (Analysis 2) 358
Career Aspiration (Analysis 3) 361
Fitting LISREL Models by the LISMOD Modeling Language **366**
The Measurement Model for y 368
The Measurement Model for x 369
The Structural Model 370
Fit Summary of the LISMOD Model for Career Aspiration Analysis 3 373
Some Important PROC CALIS Features **377**

Modeling Languages for Specifying Models 377
Estimation Methods 377
Statistical Inference 379
Multiple-Group Analysis 380
Goodness-of-Fit Statistics 381
Customizable Fit Summary Table 382
Standardized Solution 382
Statistical Graphics 382
Testing Parametric Functions 383
Effect Analysis 383
Model Modifications 384
Optimization Methods 384
Other Commonly Used Options 385
Comparison of the CALIS and FACTOR Procedures for Exploratory Factor Analysis **385**
Comparison of the CALIS and SYSLIN Procedures **386**
References **387**

Overview of Structural Equation Modeling with Latent Variables

Structural equation modeling includes analysis of covariance structures and mean structures, fitting systems of linear structural equations, factor analysis, and path analysis. In terms of the mathematical and statistical techniques involved, these various types of analyses are more or less interchangeable because the underlying methodology is based on analyzing the mean and covariance structures. However, the different analysis types emphasize different aspects of the analysis.

The analysis of covariance structures refers to the formulation of a model for the observed variances and covariances among a set of variables. The model expresses the variances and covariances as functions of some basic parameters. Similarly, the analysis of mean structures refers to the formulation of a model for the observed means. The model expresses the means as functions of some basic parameters. Usually, the covariance structures are of primary interest. However, sometimes the mean structures are analyzed simultaneously with the covariance structures in a model.

Corresponding to this kind of abstract formulation of mean and covariance structure analysis, PROC CALIS offers you two matrix-based modeling languages for specifying your model:

- MSTRUCT: a matrix-based model specification language that enables you to directly specify the parameters in the covariance and mean model matrices
- COSAN: a general matrix-based model specification language that enables you to specify a very wide class of mean and covariance structure models in terms of matrix expressions

Instead of focusing directly on the mean and covariance structures, other generic types of structural equation modeling emphasize more about the functional relationships among variables. Mean and covariance structures are still the means of these analyses, but they are usually implied from the structural relationships, rather than being directly specified as in the COSAN or MSTRUCT modeling languages.

In linear structural equations, the model is formulated as a system of equations that relates several random variables with assumptions about the variances and covariances of the random variables. The variables involved in the system of linear structural equations could be observed (manifest) or latent. Causal relationships between variables are hypothesized in the model.

When all observed variables in the model are hypothesized as indicator measures of underlying latent factors and the main interest is about studying the structural relations among the latent factors, it is a modeling scenario for factor-analysis or LISREL (Keesling 1972; Wiley 1973; Jöreskog 1973). PROC CALIS provides you two modeling languages that are closely related to this type of modeling scenario:

- FACTOR: a non-matrix-based model specification language that supports both exploratory and confirmatory factor analysis, including orthogonal and oblique factor rotations
- LISMOD: a matrix-based model specification language that enables you to specify the parameters in the LISREL model matrices

When causal relationships among observed and latent variables are freely hypothesized so that the observed variables are not limited to the roles of being measured indicators of latent factors, it is a modeling scenario for general path modeling (path analysis). In general path modeling, the model is formulated as a path diagram, in which arrows that connect variables represent variances, covariances, and path coefficients (effects). Depending on the way you represent the path diagram, you can use any of the following three different modeling languages in PROC CALIS:

- PATH: a non-matrix-based language that enables you to specify path-like relationships among variables
- RAM: a matrix-based language that enables you to specify the paths, variances, and covariance parameters in terms of the RAM model matrices (McArdle and McDonald 1984)
- LINEQS: an equation-based language that uses linear equations to specify functional or path relationships among variables (for example, the EQS model by Bentler 1995)

Although various types of analyses are put into distinct classes (with distinct modeling languages), with careful parameterization and model specification, it is possible to apply any of these modeling languages to the same analysis. For example, you can use the PATH modeling language to specify a confirmatory factor-analysis model, or you can use the LISMOD modeling language to specify a general path model. However, for some situations some modeling languages are easier to use than others. See the section "Which Modeling Language?" on page 1530 in Chapter 33, "The CALIS Procedure," for a detailed discussion of the modeling languages supported in PROC CALIS.

Loehlin (1987) provides an excellent introduction to latent variable models by using path diagrams and structural equations. A more advanced treatment of structural equation models with latent variables is given by Bollen (1989). Fuller (1987) provides a highly technical statistical treatment of measurement-error models.

This chapter illustrates applications of PROC CALIS, describes some of the main modeling features of PROC CALIS, and compares the CALIS procedure with the FACTOR and the SYSLIN procedures.

Testing Covariance Patterns

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex5.sas.)

The most basic use of PROC CALIS is testing covariance patterns. Consider a repeated-measures experiment where individuals are tested for their motor skills at three different time points. No treatments are introduced between these tests. The three test scores are denoted as X_1, X_2, and X_3, respectively. These test scores are likely correlated because the same set of individuals has been used. More specifically, the researcher wants to test the following pattern of the population covariance matrix $\mathbf{\Sigma}$:

$$\mathbf{\Sigma} = \begin{pmatrix} \phi & \theta & \theta \\ \theta & \phi & \theta \\ \theta & \theta & \phi \end{pmatrix}$$

Because there are no treatments between the tests, this pattern assumes that the distribution of motor skills stays more or less the same over time, as represented by the same ϕ for the diagonal elements of $\mathbf{\Sigma}$. The covariances between the test scores for motor skills also stay the same, as represented by the same θ for all the off-diagonal elements of $\mathbf{\Sigma}$.

Suppose you summarize your data in a covariance matrix, which is stored in the following SAS data set:

```
data motor(type=cov);
   input _type_ $ _name_ $ x1 x2 x3;
   datalines;
COV     x1      3.566   1.342   1.114
COV     x2      1.342   4.012   1.056
COV     x3      1.114   1.056   3.776
N        .         36      36      36
;
```

The diagonal elements are somewhat close to each other but are not the same. The off-diagonal elements are also very close to each other but are not the same. Could these observed differences be due to chance? Given the sample covariance matrix, can you test the hypothesized patterned covariance matrix in the population?

Setting up this patterned covariance model in PROC CALIS is straightforward with the MSTRUCT modeling language:

```
proc calis data=motor;
   mstruct var  = x1-x3;
   matrix _cov_ = phi
                  theta phi
                  theta theta phi;
run;
```

In the VAR= option in the MSTRUCT statement, you specify that x1–x3 are the variables in the covariance matrix. Next, you specify the elements of the patterned covariance matrix in the MATRIX statement with the _COV_ keyword. Because the covariance matrix is symmetric, you need to specify only the lower triangular elements in the MATRIX statement. You use phi for the parameters of all diagonal elements and theta for the parameters of all off-diagonal elements. Matrix elements with the same parameter name are implicitly

constrained to be equal. Hence, this is the patterned covariance matrix that you want to test. Some output results from PROC CALIS are shown in Figure 18.1.

Figure 18.1 Fit Summary

Fit Summary	
Chi-Square	0.3656
Chi-Square DF	4
Pr > Chi-Square	0.9852

MSTRUCT _COV_ Matrix: Estimate/StdErr/t-value/p-value			
	x1	**x2**	**x3**
x1	3.7847	1.1707	1.1707
	0.5701	0.5099	0.5099
	6.6383	2.2960	2.2960
	<.0001	0.0217	0.0217
	[phi]	[theta]	[theta]
x2	1.1707	3.7847	1.1707
	0.5099	0.5701	0.5099
	2.2960	6.6383	2.2960
	0.0217	<.0001	0.0217
	[theta]	[phi]	[theta]
x3	1.1707	1.1707	3.7847
	0.5099	0.5099	0.5701
	2.2960	2.2960	6.6383
	0.0217	0.0217	<.0001
	[theta]	[theta]	[phi]

First, PROC CALIS shows that the chi-square test for the model fit is 0.3656 (df = 4, p=0.9852). Because the chi-square test is not significant, it supports the hypothesized patterned covariance model. Next, PROC CALIS shows the estimates in the covariance matrix under the hypothesized model. The estimates for the diagonal elements are all 3.7847, and the estimates for off-diagonal elements are all 1.1707. Estimates of standard errors and t values for these covariance and variance parameters are also shown.

The MSTRUCT modeling language in PROC CALIS enables you to test various kinds of covariance and mean patterns, including matrices with fixed or constrained values. For example, consider a population covariance model in which correlations among the motor test scores are hypothesized to be zero. In other words, the covariance pattern is:

$$\mathbf{\Sigma} = \begin{pmatrix} \phi_1 & 0 & 0 \\ 0 & \phi_2 & 0 \\ 0 & 0 & \phi_3 \end{pmatrix}$$

Essentially, this diagonally-patterned covariance model means that the data are randomly and independently generated for x1–x3 under the multivariate normal distribution. Only the variances of the variables are parameters in the model, and the variables are not correlated at all.

You can use the MSTRUCT modeling language of PROC CALIS to fit this diagonally-patterned covariance matrix to the data for motor skills, as shown in the following statements:

```
proc calis data=motor;
   mstruct var  = x1-x3;
   matrix _cov_ = phi1
                     0.   phi2
                     0.      0.    phi3;
run;
```

Some of the output is shown in Figure 18.2.

Figure 18.2 Fit Summary: Testing Uncorrelatedness

Fit Summary	
Chi-Square	9.2939
Chi-Square DF	3
Pr > Chi-Square	0.0256

MSTRUCT _COV_ Matrix: Estimate/StdErr/t-value/p-value

	x1	x2	x3
x1	3.5660 0.8524 4.1833 <.0001 [phi1]	0	0
x2	0	4.0120 0.9591 4.1833 <.0001 [phi2]	0
x3	0	0	3.7760 0.9026 4.1833 <.0001 [phi3]

PROC CALIS shows that the chi-square test for the model fit is 9.2939 (df=3, p=0.0256). Because the chi-square test is significant, it does not support the patterned covariance model that postulates zero correlations among the variables. This conclusion is consistent with what is already known—the motor test scores should be somewhat correlated because they are measurements over time for the same group of individuals.

The output also shows the estimates of variances under the model. Each diagonal element of the covariance matrix has a distinct estimate because different parameters have been hypothesized under the patterned covariance model.

Testing Built-In Covariance Patterns in PROC CALIS

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex5.sas`.)

Some covariance patterns are well-known in multivariate statistics. For example, testing the diagonal pattern for a covariance matrix in the preceding section is a test of uncorrelatedness between the observed variables. Under the multivariate normal assumption, this test is also a test of independence between the observed variables. This test of independence is routinely applied in maximum likelihood factor analysis for testing the zero common factor hypothesis for the observed variables. For testing such a well-known covariance pattern, PROC CALIS provides an efficient way of specifying a model. With the COVPATTERN= option, you can invoke the built-in covariance patterns in PROC CALIS without the MSTRUCT model specifications, which could become laborious when the number of variables are large.

For example, to test the diagonal pattern (uncorrelatedness) of the motor skills, you can simply use the following specification:

```
proc calis data=motor covpattern=uncorr;
run;
```

The COVPATTERN=UNCORR option in the PROC CALIS statement invokes the diagonally patterned covariance matrix for the motor skills. PROC CALIS then generates the appropriate free parameters for this built-in covariance pattern. As a result, the MATRIX statement is not needed for specifying the free parameters, as it is if you use explicit MSTRUCT model specifications. Some of the output for using the COVPATTERN= option is shown in Figure 18.3.

Figure 18.3 Fit Summary: Testing Uncorrelatedness with the COVPATTERN= Option

Fit Summary	
Chi-Square	8.8071
Chi-Square DF	3
Pr > Chi-Square	0.0320

MSTRUCT _COV_ Matrix: Estimate/StdErr/t-value/p-value			
	x1	**x2**	**x3**
x1	3.5660 0.8524 4.1833 <.0001 [_varparm_1]	0	0
x2	0	4.0120 0.9591 4.1833 <.0001 [_varparm_2]	0
x3	0	0	3.7760 0.9026 4.1833 <.0001 [_varparm_3]

In the second table of Figure 18.3, the estimates of variances and their standard errors are the same as those

shown in Figure 18.2. The only difference is that the parameter names (for example, _varparm_1) for the variances in Figure 18.3 are generated by PROC CALIS, instead of being specified as those in Figure 18.2.

However, the current chi-square test for the model fit is 8.8071 (df=3, p=0.0320), which is different from that in Figure 18.2 for testing the same covariance pattern. The reason is that the chi-square correction due to Bartlett (1950) has been applied automatically to the current built-in covariance pattern testing. Theoretically, this corrected chi-square value is more accurate. Therefore, in addition to its efficiency in specification, the built-in covariance pattern with the COVPATTERN= option offers an extra advantage in the automatic chi-square correction.

The COVPATTERN= option supports many other built-in covariance patterns. For details, see the COVPATTERN= option. See also the MEANPATTERN= option for testing built-in mean patterns.

Direct and Implied Covariance Patterns

You have seen how you can use PROC CALIS to test covariance patterns directly. Basically, you can specify the parameters in the covariance and mean matrices directly by using the MSTRUCT modeling language, which is invoked by the MSTRUCT statement. You can also use the COVPATTERN= option to test some built-in covariance patterns in PROC CALIS. To handle more complicated covariance and mean structures that are products of several model matrices, you can use the COSAN modeling language. The COSAN modeling language is too powerful to consider in this introductory chapter, but see the COSAN statement and the section "The COSAN Model" on page 1734 in Chapter 33, "The CALIS Procedure."

This section considers the fitting of patterned covariances matrix directly by using the MSTRUCT and the MATRIX statements or by the COVPATTERN= option. However, in most applications of structural equation modeling, the covariance patterns are not specified directly but are implied from the linear structural relationships among variables. The next few sections show how you can use other modeling languages in PROC CALIS to specify structural equation models with implied mean and covariance structures.

Regression with Measurement Errors

In this section, you start with a linear regression model and learn how the regression equation can be specified in PROC CALIS. The regression model is then extended to include measurement errors in the predictors and in the outcome variables. Problems with model identification are introduced.

Simple Linear Regression

Consider fitting a linear equation to two observed variables, Y and X. Simple linear regression uses the following model form:

$$Y = \alpha + \beta X + E_Y$$

The model makes the following assumption:

$$\mathrm{Cov}(X, E_Y) = 0$$

The parameters α and β are the intercept and regression coefficient, respectively, and E_Y is an error term. If the values of X are fixed, the values of E_Y are assumed to be independent and identically distributed realizations of a normally distributed random variable with mean zero and variance $\mathrm{Var}(E_Y)$. If X is a random variable, X and E_Y are assumed to have a bivariate normal distribution with zero correlation and variances $\mathrm{Var}(X)$ and $\mathrm{Var}(E_Y)$, respectively. Under either set of assumptions, the usual formulas hold for the estimates of the intercept and regression coefficient and their standard errors. (See Chapter 4, "Introduction to Regression Procedures.")

In the REG procedure, you can fit a simple linear regression model with a MODEL statement that lists only the names of the manifest variables, as shown in the following statements:

```
proc reg;
   model Y = X;
run;
```

You can also fit this model with PROC CALIS, but the syntax is different. You can specify the simple linear regression model in PROC CALIS by using the LINEQS modeling language, as shown in the following statements:

```
proc calis;
   lineqs
      Y = beta * X + Ey;
run;
```

LINEQS stands for "LINear EQuationS." You invoke the LINEQS modeling language by using the LINEQS statement in PROC CALIS. In the LINEQS statement, you specify the linear equations of your model. The LINEQS statement syntax is similar to the mathematical equation that you would write for the model. An obvious difference between the LINEQS and the PROC REG model specification is that in LINEQS you can name the parameter involved (for example, beta) and you also specify the error term explicitly. The additional syntax required by the LINEQS statement seems to make the model specification more time-consuming and cumbersome. However, this inconvenience is minor and is offset by the modeling flexibility of the LINEQS modeling language (and of PROC CALIS, generally). As you proceed to more examples in this chapter, you will find the benefits of specifying parameter names for more complicated models with constraints. You will also find that specifying parameter names for unconstrained parameters is optional. Using parameter names in the current example is for the ease of reference in the current discussion.

You might wonder whether an intercept term is missing in the LINEQS statement and where you should put the intercept term if you want to specify it. The intercept term, which is considered as a mean structure parameter in the context of structural equation modeling, is usually omitted when statistical inferences can be drawn from analyzing the covariance structures alone. However, this does not mean that the regression

equation has a default fixed-zero intercept in the LINEQS specification. Rather, it means only that the mean structures are saturated and are not estimated in the covariance structure model. Therefore, in the preceding LINEQS specification, the intercept term α is implicitly assumed in the model. It is not of primary interest and is not estimated.

However, if you want to estimate the intercept, you can specify it in the LINEQS equations, as shown in the following specification:

```
proc calis;
   lineqs
      Y = alpha * Intercept + beta * X + Ey;
run;
```

In this LINEQS statement, alpha represents the intercept parameter α and intercept represents an internal "variable" that has a fixed value of 1 for each observation. With this specification, an estimate of α is displayed in the PROC CALIS output results. However, estimation results for other parameters are the same as those from the specification without the intercept term. Therefore, for regression models that have no latent factors, the intercept terms are usually ignored in specifying CALIS models without affecting the interpretation of regression effect estimates.

However, if the model has latent factors and the mean parameters are being interpreted, the intercept terms should be included. The next section explains how you can specify the intercepts and the associated mean and covariance parameters for some traditional errors-in-variables models.

Errors-in-Variables Regression

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex1.sas.)

For ordinary unconstrained regression models, there is no reason to use PROC CALIS instead of PROC REG. But suppose that the predictor variable X is a random variable that is contaminated by errors (especially measurement errors), and you want to estimate the linear relationship between the true, error-free scores. The following model takes this kind of measurement errors into account:

$$\begin{aligned} Y &= \alpha + \beta F_X + E_Y \\ X &= F_X + E_X \end{aligned}$$

The model assumes the following:

$$\mathrm{Cov}(F_X, E_Y) = \mathrm{Cov}(F_X, E_X) = \mathrm{Cov}(E_X, E_Y) = 0$$

There are two equations in the model. The first one is the so-called structural model, which describes the relationships between Y and the true score predictor F_X. This equation is your main interest. However, F_X is a latent variable that has not been observed. Instead, what you have observed for this predictor is X, which is the contaminated version of F_X with measurement error or other errors, denoted by E_X, added. This measurement process is described in the second equation, or the so-called measurement model. By analyzing the structural and measurement models (or the two linear equations) simultaneously, you want to estimate the true score effect β.

The assumption that the error terms E_X and E_Y and the latent variable F_X are jointly uncorrelated is of critical importance in the model. This assumption must be justified on substantive grounds such as the physical properties of the measurement process. If this assumption is violated, the estimators might be severely biased and inconsistent.

You can express the current errors-in-variables model by the LINEQS modeling language as shown in the following statements:

```
proc calis;
   lineqs
      Y  = alpha * Intercept + beta * Fx + Ey,
      X  =     0 * Intercept +    1 * Fx + Ex;
   mean
      Fx;
run;
```

In this specification, you need to specify only the equations involved without specifying the assumptions about the correlations among Fx, Ey, and Ex. In the LINEQS modeling language, you should always name latent factors with the 'F' or 'f' prefix (for example, Fx) and error terms with the 'E' or 'e' prefix (for example, Ey and Ex). Given this LINEQS notation, latent factors and error terms, by default, are uncorrelated in the model.

Notice that the intercept in the equation for X in the LINEQS statement is specified as a fixed zero. In addition, the MEAN statement specifies that the mean of latent factor Fx is a free parameter. These specifications override the default of free intercept parameters for observed variables (for X, in this case) and fixed zero means for latent factors (for Fx, in this case)—they are comparable to the conventional parameterization of the errors-in-variables regression models.

Consider an example of an errors-in-variables regression model. Fuller (1987, pp. 18–19) analyzes a data set from Voss (1969) that involves corn yields (Y) and available soil nitrogen (X) for which there is a prior estimate of the measurement error for soil nitrogen Var(E_X) of 57. The scientific question is: how does nitrogen affect corn yields? The linear prediction of corn yields by nitrogen should be based on a measure of nitrogen that is not contaminated with measurement error. Hence, the errors-in-variables model is applied. F_X in the model represents the "true" nitrogen measure, X represents the observed measure of nitrogen, which has a true score component F_X and an error component E_X. Given that the measurement error for soil nitrogen Var(E_X) is 57, you can specify the errors-in-variables regression model with the following statements in PROC CALIS:

```
data corn(type=cov);
   input _type_ $ _name_ $ y x;
   datalines;
cov    y    87.6727    .
cov    x    104.8818   304.8545
mean   .    97.4545    70.6364
n      .    11         11
;

proc calis;
   lineqs
      Y  = alpha * Intercept + beta * Fx + Ey,
      X  =     0 * Intercept +    1 * Fx + Ex;
   variance
      Ex = 57;
```

```
    mean
       Fx;
run;
```

In the LINEQS statement, the intercept in the equation for X is set to a fixed zero. In the VARIANCE statement, the variance of Ex (measurement error for X) is specified as the constant value 57. In the MEAN statement, the means of F_X is specified as a free parameter. PROC CALIS produces the estimates shown in Figure 18.4.

Figure 18.4 Errors-in-Variables Model for Corn Data

Linear Equations

y =	67.5641	(**)	Intercept	+	0.4232	(**)	Fx + 1.0000	Ey
x =	0		Intercept	+	1.0000		Fx + 1.0000	Ex

Effects in Linear Equations

Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
y	Intercept	alpha	67.56409	11.93888	5.6592	<.0001
y	Fx	beta	0.42316	0.16582	2.5520	0.0107
x	Intercept		0			
x	Fx		1.00000			

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**Ex**		57.00000			
Latent	**Fx**	_Add1	247.85450	136.33508	1.8180	0.0691
Error	**Ey**	_Add2	43.29105	23.92488	1.8095	0.0704

Mean Parameters

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Latent	**Fx**	_Parm1	70.63640	5.52136	12.7933	<.0001

In Figure 18.4, the estimate of beta is 0.4232 with a standard error estimate of 0.1658. The t value is 2.552. It is significant at the 0.05 α-level when compared to the critical value of the standard normal variate (that is, the z table). Also shown in Figure 18.4 are the estimated variances of Fx, Ey, and their estimated standard errors. The names of these parameters have the prefix '_Add'. They are added by PROC CALIS as default parameters. By employing some conventional rules for setting default parameters, PROC CALIS makes your model specification much easier and concise. For example, you do not need to specify each error variance parameter manually if it is not constrained in the model. However, you can specify these parameters explicitly if you desire. Note that in Figure 18.4, the variance of Ex is shown to be 57 without a standard error estimate because it is a fixed constant in the model. Finally, the last table shows that the intercept estimate of Fx is 70.636, which is the same as the sample mean of X, as expected.

What if you did not model the measurement error in the predictor X? That is, what is the estimate of beta if you use ordinary regression of Y on X, as described by the equation in the section "Simple Linear Regression" on page 309? You can specify such a linear regression model easily by the LINEQS modeling language. Here, you specify this linear regression model as a special case of the errors-in-variables model. That is, you constrain the variance of measurement error E_x to 0 in the preceding LINEQS model specification to form the linear regression model, as shown in the following statements:

```
proc calis;
   lineqs
      Y  = alpha * Intercept + beta * Fx + Ey,
      X  =     0 * Intercept +    1 * Fx + Ex;
   variance
      Ex = 0;
   mean
      Fx;
run;
```

Fixing the variance of Ex to zero forces the equality of X and F_X in the measurement model so that this "new" errors-in-variables model is in fact an ordinary regression model. PROC CALIS produces the estimation results in Figure 18.5.

Figure 18.5 Ordinary Regression Model for Corn Data: Zero Measurement Error in X

Linear Equations

y =	73.1528	(**)	Intercept	+	0.3440	(**)	Fx + 1.0000	Ey
x =	0		Intercept	+	1.0000		Fx + 1.0000	Ex

Effects in Linear Equations

Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
y	Intercept	alpha	73.15283	9.46542	7.7284	<.0001
y	Fx	beta	0.34404	0.13009	2.6447	0.0082
x	Intercept		0			
x	Fx		1.00000			

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**Ex**		0			
Latent	**Fx**	_Add1	304.85450	136.33508	2.2361	0.0253
Error	**Ey**	_Add2	51.58928	23.07143	2.2361	0.0253

Mean Parameters

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Latent	**Fx**	_Parm1	70.63640	5.52136	12.7933	<.0001

The estimate of beta is now 0.3440, which is an underestimate of the effect of nitrogen on corn yields given the presence of nonzero measurement error in X, where the estimate of beta is 0.4232.

Regression with Measurement Errors in X and Y

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex1.sas.)

What if there are also measurement errors in the outcome variable Y? How can you write such an extended model? The following model would take measurement errors in both X and Y into account:

$$\begin{aligned} F_Y &= \alpha + \beta F_X + D_{Fy} \\ Y &= F_Y + E_Y \\ X &= F_X + E_X \end{aligned}$$

with the following assumption:

$$\begin{aligned} \mathrm{Cov}(F_X, D_{Fy}) &= \mathrm{Cov}(F_X, E_Y) = \mathrm{Cov}(F_X, E_X) = \mathrm{Cov}(F_Y, E_Y) \\ &= \mathrm{Cov}(F_Y, E_X) = \mathrm{Cov}(E_X, E_Y) = \mathrm{Cov}(E_X, D_{Fy}) \\ &= \mathrm{Cov}(E_Y, D_{Fy}) = 0 \end{aligned}$$

Again, the first equation, expressing the relationship between two latent true-score variables, defines the structural or causal model. The next two equations express the observed variables in terms of a true score plus error; these two equations define the measurement model. This is essentially the same form as the so-called LISREL model (Keesling 1972; Wiley 1973; Jöreskog 1973), which has been popularized by the LISREL program (Jöreskog and Sörbom 1988). Typically, there are several X and Y variables in a LISREL model. For the moment, however, the focus is on the current regression form in which there is only a single predictor and a single outcome variable. The LISREL model is considered in the section "Fitting LISREL Models by the LISMOD Modeling Language" on page 366.

With the intercept term left out for modeling, you can use the following statements for fitting the regression model with measurement errors in both X and Y:

```
proc calis data=corn;
   lineqs
      Fy  = alpha * Intercept + beta * Fx + DFy,
      Y   = 0     * Intercept + 1.   * Fy + Ey,
      X   = 0     * Intercept + 1.   * Fx + Ex;
   mean
      Fx;
run;
```

Again, you do not need to specify the zero-correlation assumptions in the LINEQS model because they are set by default given the latent factors and errors in the LINEQS modeling language. When you run this model, PROC CALIS issues the following warning:

```
WARNING: Estimation problem not identified: More parameters to
         estimate ( 7 ) than the total number of mean and
         covariance elements ( 5 ).
```

The seven parameters in the model include alpha, beta, the mean of Fx, and the variances for the exogenous variables Fx, DFy, Ey, and Ex, which are treated as free parameters by default in PROC CALIS. You have seven parameters to estimate, but the information for estimating these seven parameters comes from the three unique elements in the sample covariance matrix and two sample means for X and Y. Hence, your model is in the so-called underidentification situation. Model identification is discussed in more detail in the section "Model Identification" on page 317.

To make the current model identified, you can put constraints on some parameters. This reduces the number of independent parameters to estimate in the model. In the errors-in-variables model for the corn data, the variance of Ex (measurement error for X) is given as the constant value 57, which was obtained from a previous study. This could still be applied in the current model with measurement errors in both X and Y. In addition, if you are willing to accept the assumption that the structural equation model is (almost) deterministic, then the variance of Dfy could be set to 0. With these two parameter constraints, the current model is just-identified. That is, you can now estimate five free parameters from five distinct covariance and mean elements in the data. The following statements show the LINEQS model specification for this just-identified model:

```
proc calis data=corn;
   lineqs
      Fy  = alpha * Intercept + beta * Fx + DFy,
      Y   = 0     * Intercept + 1.   * Fy + Ey,
      X   = 0     * Intercept + 1.   * Fx + Ex;
   variance
      Ex  = 57.,
      Dfy = 0.;
   mean
      Fx;
run;
```

Figure 18.6 shows the estimation results.

Figure 18.6 Regression Model with Measurement Errors in X and Y for Corn Data

Linear Equations

Fy	=	67.5641	(**)	Intercept	+	0.4232	(**)	Fx + 1.0000	DFy
y	=	0		Intercept	+	1.0000		Fy + 1.0000	Ey
x	=	0		Intercept	+	1.0000		Fx + 1.0000	Ex

Effects in Linear Equations

Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Fy	Intercept	alpha	67.56409	11.93888	5.6592	<.0001
Fy	Fx	beta	0.42316	0.16582	2.5520	0.0107
y	Intercept		0			
y	Fy		1.00000			
x	Intercept		0			
x	Fx		1.00000			

Figure 18.6 *continued*

Estimates for Variances of Exogenous Variables						
Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	Ex		57.00000			
Disturbance	DFy		0			
Latent	Fx	_Add1	247.85450	136.33508	1.8180	0.0691
Error	Ey	_Add2	43.29105	23.92488	1.8095	0.0704

Mean Parameters						
Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Latent	Fx	_Parm1	70.63640	5.52136	12.7933	<.0001

In Figure 18.6, the estimate of beta is 0.4232, which is basically the same as the estimate for beta in the errors-in-variables model shown in Figure 18.4. Other estimates, including alpha, the mean of Fx, and the variances for Fx and Ey, also match for the two models. In fact, it is not difficult to show mathematically that the current constrained model with measurements errors in both Y and X is equivalent to the errors-in-variables model for the corn data. The numerical results merely confirm this fact.

It is important to emphasize that the equivalence shown here is not a general statement about the current model with measurement errors in X and Y and the errors-in-variables model. Essentially, the equivalence of the two models as applied to the corn data is due to those constraints imposed on the measurement error variances for DFy and Ex. The more important implication from these two analyses is that for the model with measurement errors in both X and Y, you need to set more parameter constraints to make the model identified. Some constraints might be substantively meaningful, while others might need strong or risky assumptions.

For example, setting the variance of Ex to 57 is substantively meaningful because it is based on a prior study. However, setting the variance of Dfy to 0 implies the acceptance of the deterministic structural model, which could be a rather risky assumption in most practical situations. It turns out that using these two constraints together for the model identification of the regression with measurement errors in both X and Y does not give you more substantively important information than what the errors-in-variables model has already given you (compare Figure 18.6 with Figure 18.4). Therefore, the set of identification constraints you use might be important in at least two aspects. First, it might lead to an identified model if you set them properly. Second, given that the model is identified, the meaningfulness of your model depends on how reasonable your identification constraints are.

The two identification constraints set on the regression model with measurement errors in both X and Y make the model identified. But they do not lead to model estimates that are more informative than that of the errors-in-variables regression. Some other sets of identification constraints, if available, might have been more informative. For example, if there were a prior study about the measurement error variance of corn yields (Y), a fixed constant for the variance of Ey could have been set, instead of the unrealistic zero variance constraint of Dfy. This way the estimation results of the regression model with measurement errors in both X and Y would offer you something different from the errors-in-variables regression.

Setting identification constraints could be based on convention or other arguments. See the section "Illustration of Model Identification: Spleen Data" on page 318 for an example where model identification is attained by setting constant error variances for X and Y in the model. For the corn data, you have seen that fixing the error variance of the predictor variable led to model identification of the errors-in-variables model. In this case, prior knowledge about the measurement error variance is necessary. This necessity is partly due to

the fact that each latent true score variable has only one observed variable as its indicator measure. When you have more measurement indicators for the same latent factor, fixing the measurement error variances to constants for model identification would not be necessary. This is the modeling scenario assumed by the LISREL model (see the section "Fitting LISREL Models by the LISMOD Modeling Language" on page 366), of which the confirmatory factor model is a special case. The confirmatory factor model is described and illustrated in the section "The FACTOR and RAM Modeling Languages" on page 342.

Model Identification

As discussed in the preceding section, if you try to fit the errors-in-variables model with measurement errors in both X and Y without applying certain constraints, the model is not identified and you cannot obtain unique estimates of the parameters. For example, the errors-in-variables model with measurement errors in both X and Y has five parameters (one coefficient β and four variances). The covariance matrix of the observed variables Y and X has only three elements that are free to vary, since $\mathrm{Cov}(Y,X)=\mathrm{Cov}(X,Y)$. Therefore, the covariance structure can be expressed as three equations in five unknown parameters. Since there are fewer equations than unknowns, there are many different sets of values for the parameters that provide a solution for the equations. Such a model is said to be underidentified.

If the number of parameters equals the number of free elements in the covariance matrix, then there might exist a unique set of parameter estimates that exactly reproduce the observed covariance matrix. In this case, the model is said to be just-identified or saturated.

If the number of parameters is less than the number of free elements in the covariance matrix, there might exist no set of parameter estimates that reproduces the observed covariance matrix exactly. In this case, the model is said to be overidentified. Various statistical criteria, such as maximum likelihood, can be used to choose parameter estimates that approximately reproduce the observed covariance matrix. If you use ML, FIML, GLS, or WLS estimation, PROC CALIS can perform a statistical test of the goodness of fit of the model under the certain statistical assumptions.

If the model is just-identified or overidentified, it is said to be identified. If you use ML, FIML, GLS, or WLS estimation for an identified model, PROC CALIS can compute approximate standard errors for the parameter estimates. For underidentified models, PROC CALIS obtains approximate standard errors by imposing additional constraints resulting from the use of a generalized inverse of the Hessian matrix.

You cannot guarantee that a model is identified simply by counting the parameters. For example, for any latent variable, you must specify a numeric value for the variance, or for some covariance involving the variable, or for a coefficient of an indicator variable. Otherwise, the scale of the latent variable is indeterminate, and the model is underidentified regardless of the number of parameters and the size of the covariance matrix. As another example, an exploratory factor analysis with two or more common factors is always underidentified because you can rotate the common factors without affecting the fit of the model.

PROC CALIS can usually detect an underidentified model by computing the approximate covariance matrix of the parameter estimates and checking whether any estimate is linearly related to other estimates (Bollen 1989, pp. 248–250), in which case PROC CALIS displays equations showing the linear relationships among the estimates. Another way to obtain empirical evidence regarding the identification of a model is to run the analysis several times with different initial estimates to see whether the same final estimates are obtained. Bollen (1989) provides detailed discussions of conditions for identification in a variety of models.

Illustration of Model Identification: Spleen Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex2.sas.)

When your model involves measurement errors in variables and you need to use latent true scores in the regression or structural equation, you might encounter some model identification problems in estimation if you do not put certain identification constraints in the model. An example is shown in the section "Regression with Measurement Errors in X and Y" on page 314 for the corn data. You "solved" the problem by assuming a deterministic model with perfect prediction in the structural model. However, this assumption could be very risky and does not lead to estimation results that are substantively different from the model with measurement error only in X.

This section shows how you can apply another set of constraints to make the measurement model with errors in both X and Y identified without assuming the deterministic structural model. First, the identification problem is illustrated here again in light of the PROC CALIS diagnostics.

The following example is inspired by Fuller (1987, pp. 40–41). The hypothetical data are counts of two types of cells in spleen samples: cells that form rosettes and nucleated cells. It is reasonable to assume that counts have a Poisson distribution; hence, the square roots of the counts should have a constant error variance of 0.25. You can use PROC CALIS to fit this regression model with measurement errors in X and Y to the data. (See the section "Regression with Measurement Errors in X and Y" on page 314 for model definitions.) However, before fitting this target model, it is illustrative to see what would happen if you do not assume the constant error variance.

The following statements show the LINEQS specification of an errors-in-variables regression model for the square roots of the counts without constraints on the parameters:

```
data spleen;
   input rosette nucleate;
   sqrtrose=sqrt(rosette);
   sqrtnucl=sqrt(nucleate);
   datalines;
4 62
5 87
5 117
6 142
8 212
9 120
12 254
13 179
15 125
19 182
28 301
51 357
;

proc calis data=spleen;
   lineqs factrose = beta * factnucl + disturb,
          sqrtrose =        factrose + err_rose,
          sqrtnucl =        factnucl + err_nucl;
   variance
```

```
            factnucl = v_factnucl,
            disturb  = v_disturb,
            err_rose = v_rose,
            err_nucl = v_nucl;
  run;
```

This model is underidentified. You have five parameters to estimate in the model, but the number of distinct covariance elements is only three.

In the LINEQS statement, you specify the structural equation and then two measurement equations. In the structural equation, the variables factrose and factnucl are latent true scores for the corresponding measurements in sqrtrose and sqrtnucl, respectively. The structural equation represents the true variable relationship of interest. You name the regression coefficient parameter as beta and the error term as disturb in the structural model. (For structural equations, you can use names with prefix 'D' or 'd' to denote error terms.) The variance of factnucl and the variance of disturb are also parameters in the model. You name these variance parameters as v_factnucl and v_disturb in the VARIANCE statement. Therefore, you have three parameters in the structural equation.

In the measurement equations, the observed variables sqrtrose and sqrtnucl are specified as the sums of their corresponding true latent scores and error terms, respectively. The error variances are also parameters in the model. You name them as v_rose and v_nucl in the VARIANCE statement. Now, together with the three parameters in the structural equation, you have a total of five parameters in your model.

All variance specifications in the VARIANCE statement are actually optional in PROC CALIS. They are free parameters by default. In this example, it is useful to name these parameters so that explicit references to these parameters can be made in the following discussion.

PROC CALIS displays the following warning when you fit this underidentified model:

```
   WARNING: Estimation problem not identified: More parameters to
            estimate ( 5 ) than the total number of mean and
            covariance elements ( 3 ).
```

In this warning, the three covariance elements refer to the sample variances of sqrtrose and sqrtnucl and their covariance. PROC CALIS diagnoses the parameter indeterminacy as follows:

```
   NOTE: Covariance matrix for the estimates is not full rank.
   NOTE: The variance of some parameter estimates is zero or
         some parameter estimates are linearly related to other
         parameter estimates as shown in the following equations:

v_rose       =      -0.147856   +       0.447307   *   v_disturb

v_nucl       =    -110.923690   -       0.374367   *   beta       +
                    10.353896   *   v_factnucl     +   1.536613   *   v_disturb
```

With the warning and the notes, you are now certain that the model is underidentified and you cannot interpret your parameter estimates meaningfully.

Now, to make the model identified, you set the error variances to 0.25 in the VARIANCE statement, as shown in the following specification:

```
proc calis data=spleen residual;
   lineqs factrose = beta * factnucl + disturb,
          sqrtrose =        factrose + err_rose,
          sqrtnucl =        factnucl + err_nucl;
   variance
          factnucl = v_factnucl,
          disturb  = v_disturb,
          err_rose = 0.25,
          err_nucl = 0.25;
run;
```

In the specification, you use the RESIDUAL option in the PROC CALIS statement to request the residual analysis. An annotated fit summary is shown in Figure 18.7.

Figure 18.7 Spleen Data: Annotated Fit Summary for the Just-Identified Model

Fit Summary	
Chi-Square	0.0000
Chi-Square DF	0
Pr > Chi-Square	.

You notice that the model fit chi-square is 0 and the corresponding degrees of freedom is also 0. This indicates that your model is "just" identified, or your model is saturated—you have three distinct elements in the sample covariance matrix for the estimation of three parameters in the model. In the PROC CALIS results, you no longer see the warning message about underidentification or any notes about linear dependence in parameters.

For just-identified or saturated models like the current case, you expect to get zero residuals in the covariance matrix, as shown in Figure 18.8:

Figure 18.8 Spleen Data: Residuals for the Just-identified Model

Raw Residual Matrix		
	sqrtrose	sqrtnucl
sqrtrose	0.00000	0.00000
sqrtnucl	0.00000	0.00000

Residuals are the differences between the fitted covariance matrix and the sample covariance matrix. When the residuals are all zero, the fitted covariance matrix matches the sample covariance matrix perfectly (the parameter estimates reproduce the sample covariance matrix exactly).

You can now interpret the estimation results of this just-identified model, as shown in Figure 18.9:

Figure 18.9 Spleen Data: Parameter Estimated for the Just-Identified Model

Linear Equations						
factrose	=	0.3907	(**)	factnucl	+ 1.0000	disturb
sqrtrose	=	1.0000		factrose	+ 1.0000	err_rose
sqrtnucl	=	1.0000		factnucl	+ 1.0000	err_nucl

Figure 18.9 *continued*

Effects in Linear Equations						
Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
factrose	factnucl	beta	0.39074	0.07708	5.0692	<.0001
sqrtrose	factrose		1.00000			
sqrtnucl	factnucl		1.00000			

Estimates for Variances of Exogenous Variables						
Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Latent	**factnucl**	v_factnucl	10.50458	4.58577	2.2907	0.0220
Disturbance	**disturb**	v_disturb	0.38153	0.28556	1.3361	0.1815
Error	**err_rose**		0.25000			
	err_nucl		0.25000			

Notice that because the error variance parameters for variables err_rose and err_nucl are fixed constants in the model, there are no standard error estimates for them in Figure 18.9. For the current application, the estimation results of the just-identified model are those you would interpret and report. However, to completely illustrate model identification, an additional constraint is imposed to show an overidentified model. In the section "Regression with Measurement Errors in X and Y" on page 314, you impose a zero-variance constraint on the disturbance variable Dfy for the model identification. Would this constraint be necessary here for the spleen data too? The answer is no because with the two constraints on the variances of err_rose and err_nucl, the model has already been meaningfully specified and identified. Adding more constraints such as a zero variance for disturb would make the model overidentified unnecessarily. The following statements show the specification of such an overidentified model for the spleen data:

```
proc calis data=spleen residual;
   lineqs factrose = beta * factnucl + disturb,
          sqrtrose =        factrose + err_rose,
          sqrtnucl =        factnucl + err_nucl;
   variance
          factnucl = v_factnucl,
          disturb  = 0.,
          err_rose = 0.25,
          err_nucl = 0.25;
run;
```

An annotated fit summary table for the overidentified model is shown in Figure 18.10.

Figure 18.10 Spleen Data: Annotated Fit Summary for the Overidentified Model

Fit Summary	
Chi-Square	5.2522
Chi-Square DF	1
Pr > Chi-Square	0.0219
Standardized RMR (SRMR)	0.0745
Adjusted GFI (AGFI)	0.1821
RMSEA Estimate	0.6217
Bentler Comparative Fit Index	0.6535

The chi-square is 5.2522 (df=1, p=0.0219). Overall, the model does not provide a good fit. The sample size is so small that the p-value of the chi-square test should not be taken to be accurate, but to get a small p-value with such a small sample indicates that it is possible that the model is seriously deficient.

This same conclusion can be drawn by looking at other fit indices in the table. In Figure 18.10, several fit indices are computed for the model. For example, the standardized root mean square residual (SRMSR) is 0.0745 and the adjusted goodness of fit (AGFI) is 0.1821. By conventions, a good model should have an SRMSR smaller than 0.05 and an AGFI larger than 0.90. The root mean square error of approximation (RMSEA) (Steiger and Lind 1980) is 0.6217, but an RMSEA below 0.05 is recommended for a good model fit (Browne and Cudeck 1993). The comparative fit index (CFI) is 0.6535, which is also low as compared to the acceptable level at 0.90.

When you fit an overidentified model, usually you do not find estimates that match the sample covariance matrix exactly. The discrepancies between the fitted covariance matrix and the sample covariance matrix are shown as residuals in the covariance matrix, as shown in Figure 18.11.

Figure 18.11 Spleen Data: Residuals for the Overidentified Model

Raw Residual Matrix

	sqrtrose	sqrtnucl
sqrtrose	0.28345	-0.11434
sqrtnucl	-0.11434	0.04613

As you can see in Figure 18.11, the residuals are nonzero. This indicates that the parameter estimates do not reproduce the sample covariance matrix exactly. For overidentified models, nonzero residuals would be the norm rather than exception, but the general goal is to find the "best" set of estimates so that the residuals are as small as possible.

The parameter estimates are shown in Figure 18.12.

Figure 18.12 Spleen Data: Parameter Estimated for the Overidentified Model

Linear Equations

factrose	=	0.4034	(**)	factnucl	+ 1.0000	disturb
sqrtrose	=	1.0000		factrose	+ 1.0000	err_rose
sqrtnucl	=	1.0000		factnucl	+ 1.0000	err_nucl

Effects in Linear Equations

Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
factrose	factnucl	beta	0.40340	0.05078	7.9439	<.0001
sqrtrose	factrose		1.00000			
sqrtnucl	factnucl		1.00000			

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Latent	**factnucl**	v_factnucl	10.45846	4.56608	2.2905	0.0220
Disturbance	**disturb**		0			
Error	**err_rose**		0.25000			
	err_nucl		0.25000			

The estimate of beta in this model is 0.4034. Given that the model fit is bad and the zero variance for the error term disturb is unreasonable, beta could have been overestimated in the current overidentified model, as compared with the just-identified model, where the estimate of beta is only 0.3907. In summary, both the fit summary and the estimation results indicate that the zero variance for disturb in the overidentified model for the spleen data has been imposed unreasonably.

The purpose of the current illustration is not that you should not consider an overidentified model for your data in general. Quite the opposite, in practical structural equation modeling it is usually the overidentified models that are of the paramount interest. You can test or gauge the model fit of overidentified models. Good overidentified models enable you to establish scientific theories that are precise and general. However, most fit indices are not meaningful when applied to just-identified saturated models. Also, even though you always get zero residuals for just-identified saturated models, those models usually are not precise enough to be a scientific theory.

The overidentified model for the spleen data highlights the importance of setting meaningful identification constraints. Whether your resulting model is just-identified or overidentified, it is recommended that you do the following:

- Give priorities to those identification constraints that are derived from prior studies, substantive grounds, or mathematical basis.
- Avoid making unnecessary identification constraints that might bias your model estimation.

Path Diagrams and Path Analysis

Sections "Errors-in-Variables Regression" on page 310, "Regression with Measurement Errors in X and Y" on page 314, and "Illustration of Model Identification: Spleen Data" on page 318 show how you can specify models by means of equations in the LINEQS modeling language. This section shows you how to specify models that are represented by path diagrams. The PATH modeling language of PROC CALIS is the main tool for this purpose.

Complicated models are often easier to understand when they are expressed as path diagrams. One advantage of path diagrams over equations is that variances and covariances can be shown directly in the path diagram. Loehlin (1987) provides a detailed discussion of path diagrams. Another advantage is that the path diagram can be transcribed easily into the PATH modeling language supported by PROC CALIS.

A path diagram for the spleen data is shown in Figure 18.13. It explicitly shows all latent variables (including error terms) and variances of exogenous variables.

Figure 18.13 Path Diagram: Spleen Data

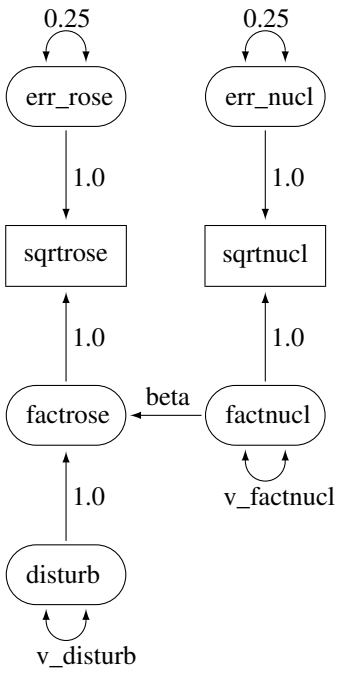

The path diagram shown in Figure 18.13 is essentially a graphical representation of the same just-identified model for the spleen data that is described in the section "Illustration of Model Identification: Spleen Data" on page 318. In path diagrams, it is customary to write the names of manifest or observed variables in rectangles and the names of latent variables in ovals. For example, sqrtrose and sqrtnucl are observed variables in the path diagram, while all others are latent variables.

The effects (the regression coefficients) in each equation are indicated by drawing arrows from the predictor variables to the outcome variable. For example, the path from factnucl to factrose is labeled with the regression coefficient beta in the path diagram shown in Figure 18.13. Other paths are labeled with fixed coefficients (or effects) of 1.

Variances of exogenous variables are drawn as double-headed arrows in Figure 18.13. For example, the variance of disturb is shown as a double-headed arrow pointing to the variable itself and is named v_disturb. Variances of the err_nucl and err_rose are also drawn as double-headed arrows but are labeled with fixed constants 0.25.

The path diagram shown in Figure 18.13 matches the features in the LINEQS model closely. For example, the error terms are depicted explicitly and their paths (regression coefficients) that connect to the associated endogenous variables are marked with fixed constants 1, reflecting the same specification in the equations of the LINEQS model. However, you can simplify the path diagram by using McArdle's RAM (reticular action model) notation (McArdle and McDonald 1984), as described in the following section.

A Simplified Path Diagram for the Spleen Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex2.sas.)

The main simplification in the path diagram is to drop all the error terms in the model. Instead, error variances are treated as residual (or partial) variances for the endogenous variables in the model or path diagram. Hence, in the path diagrams for RAM models, error variances are also represented by double-headed arrows directly attached to the endogenous variables, which is the same way you represent variances for the exogenous variables. The RAM model convention leads to a simplified representation of the path diagram for the spleen data, as shown in Figure 18.14.

Figure 18.14 Simplified Path Diagram: Spleen

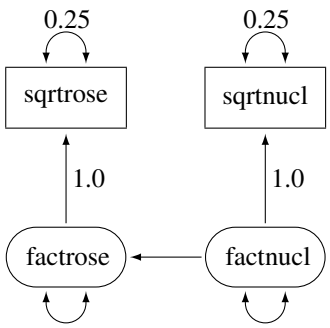

Another simplification done in Figure 18.14 is the omission of the parameter labeling in the path diagram. This simplification is not a part of the RAM notation. It is just a convention in PROC CALIS that you can omit the unconstrained parameter names without affecting the meaning of the model. Hence, the parameter names beta, v_disturb, and v_factnucl are no longer necessary in the simplified path diagram Figure 18.14. As you can see, this convention makes the task of model specification considerably simpler and easier.

The following statements show the specification of the simplified path diagram in Figure 18.14:

```
proc calis data=spleen;
   path
      sqrtrose <=== factrose   = 1.0,
      sqrtnucl <=== factnucl   = 1.0,
      factrose <=== factnucl   ;
   pvar
      sqrtrose = 0.25,      /* error variance for sqrtrose */
      sqrtnucl = 0.25,      /* error variance for sqrtnucl */
      factrose,             /* disturbance/error variance for factrose */
      factnucl;             /* variance of factnucl */
run;
```

The PATH statement invokes the PATH modeling language of PROC CALIS. In the PATH modeling language, each entry of specification corresponds to either a single- or double- headed arrow specification in the path diagram shown in Figure 18.14, as explained in the following:

- The PATH statement enables you to specify each of the single-headed arrows (paths) as path entries, which are separated by commas. You have three single-headed arrows in the path diagram and therefore you have three path entries in the PATH statement. The path entries "sqrtrose <=== factrose" and "sqrtnucl <=== factnucl" are followed by the constant 1, indicating fixed path coefficients. The path "factrose <=== factnucl" is also specified, but without giving a fixed value or a parameter name. By default, this path entry is associated with a free parameter for the effect or path coefficient.

- The PVAR statement enables you to specify each of the double-headed arrows with both heads pointing to the ***same*** variable, exogenous or endogenous. This type of arrows represents variances or error variances. You have four such double-headed arrows in the path diagram, and therefore there are four corresponding entries under the PVAR statement. Two of them are assigned with fixed constants (0.25), and the remaining two (error variance of factrose and variance of factnucl) are free variance parameters.

- The PCOV statement enables you to specify each of the double-headed arrows with its heads pointing to ***different*** variables, exogenous or endogenous. This type of arrows represents covariances between variables or their error terms. You do not have this type of double-headed arrows in the current path diagram, and therefore you do not need a PCOV statement for the corresponding model specification.

The estimation results are shown in Figure 18.15. Essentially, these are exactly the same estimation results as those that result from the LINEQS modeling language for the just-identified model in section "Illustration of Model Identification: Spleen Data" on page 318.

Figure 18.15 Spleen Data: RAM Model

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
sqrtrose	**<===**	**factrose**		1.00000			
sqrtnucl	**<===**	**factnucl**		1.00000			
factrose	**<===**	**factnucl**	beta	0.39074	0.07708	5.0692	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**sqrtrose**		0.25000			
	sqrtnucl		0.25000			
	factrose	v_disturb	0.38153	0.28556	1.3361	0.1815
Exogenous	**factnucl**	v_factnucl	10.50458	4.58577	2.2907	0.0220

Notice in Figure 18.15 that the path coefficient for path "factrose <=== factnucl" is given a parameter name _Parm1, which is generated automatically by PROC CALIS. This is the same beta parameter of the LINEQS model in the section "Illustration of Model Identification: Spleen Data" on page 318. Also, the variance parameters _Parm2 and _Parm3 in Figure 18.15 are the same v_disturb and v_factnucl parameters, respectively, in the preceding LINEQS model.

In PROC CALIS, using parameter names to specify free parameters is optional. Parameter names are generated for free parameters by default. Or, if you choose parameter names for your own convenience, you can do so without changing the model specification. For example, you can specify the preceding PATH model equivalently by adding the desired parameter names, as shown in the following statements:

```
proc calis data=spleen;
   path
      sqrtrose <=== factrose   = 1.0,
      sqrtnucl <=== factnucl   = 1.0,
      factrose <=== factnucl   = beta;
   pvar
      sqrtrose = 0.25,       /* error variance for sqrtrose */
      sqrtnucl = 0.25,       /* error variance for sqrtnucl */
      factrose = v_disturb,  /* disturbance/error variance for factrose */
      factnucl = v_factnucl; /* variance of factnucl */
run;
```

A path diagram provides you an easy and conceptual way to represent your model, while the PATH modeling language in PROC CALIS offers you an easy way to input your path diagram in a non-graphical fashion. This is especially useful for models with more complicated path structures. See the section "A Combined Measurement-Structural Model" on page 349 for a more elaborated example of the PATH model application.

The section "Some Measurement Models" on page 330 provides examples of the PATH model applied to classical test theory.

Producing Path Diagrams from the CALIS Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex2.sas.)

This section shows how you can produce high-quality graphical output for path diagrams from the CALIS procedure. Using the spleen data and the model that is represented by the path diagram in Figure 18.13 or Figure 18.14, the following statements produce the default path diagram in the output:

```
ods graphics on;

proc calis data=spleen plots=pathdiagram;
   path
      sqrtrose <=== factrose   = 1.0,
      sqrtnucl <=== factnucl   = 1.0,
      factrose <=== factnucl   ;
   pvar
      sqrtrose = 0.25,      /* error variance for sqrtrose */
      sqrtnucl = 0.25,      /* error variance for sqrtnucl */
      factrose,             /* disturbance/error variance for factrose */
      factnucl;             /* variance of factnucl */
run;

ods graphics off;
```

The ODS GRAPHICS ON statement enables the ODS Graphics, and the PLOTS=PATHDIAGRAM requests that the default path diagram of the unstandardized solution, which is shown in Figure 18.16, be output.

Figure 18.16 Default Path Diagram for the Spleen Data

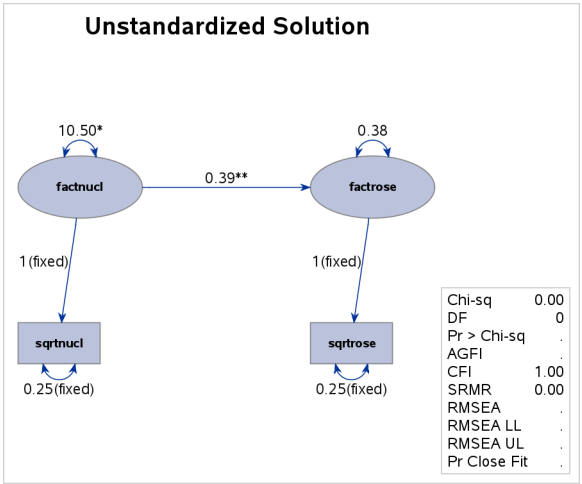

The output path diagram in Figure 18.16 uses a similar representation scheme as the simplified path diagram that is shown in Figure 18.14, except that the output path diagram in Figure 18.16 is for the unstandardized solution rather than the initial specification. The output path diagram in Figure 18.16 shows the final parameter estimates and indicates their statistical significance. Estimates that are significant at the 0.05 α-level are flagged with a single asterisk, and estimates that are significant at the 0.01 α-level are flagged with two asterisks. Hence, the path diagram shows that factnucl has a significant effect on factrose. In addition to showing the free parameter estimates, the path diagram shows the fixed parameter values and marks them as "fixed." The path diagram also includes a fit summary table to show the model fit statistics. However, because the model in Figure 18.16 is a saturated model, interpretations of the fit statistics are not meaningful.

PROC CALIS provides a wide range of options in the PATHDIAGRAM statement that you can use to customize the output path diagrams. This section illustrates some simple customization options. For example, because the fit summary statistics are not meaningful in the preceding model for the Spleen data, you can omit the fit summary table in the path diagram output by adding the following PATHDIAGRAM statement to the preceding CALIS run:

```
pathdiagram nofittable title='No Fit Summary';
```

The NOFITTABLE option suppresses the printing of the fit summary table, and the TITLE= option specifies the customized title. The resulting path diagram is shown in Figure 18.17.

Figure 18.17 Path Diagram for the Spleen Data: Fit Summary Table Omitted

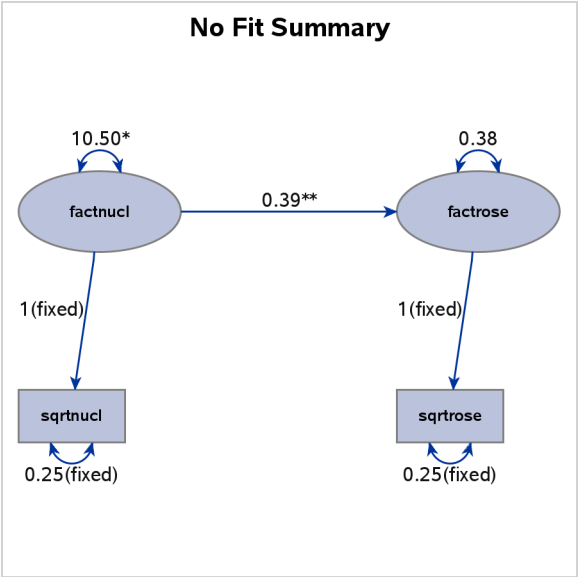

You can also change the styles of the path diagrams by using options in the PATHDIAGRAM statement. For example, to produce a path diagram for the initial specification that has the same style as the one shown in Figure 18.13, you can use the following PATHDIAGRAM statement:

```
proc calis data=spleen;
   path
      sqrtrose <=== factrose   = 1.0,
      sqrtnucl <=== factnucl   = 1.0,
      factrose <=== factnucl   = beta;
   pvar
      sqrtrose = 0.25,       /* error variance for sqrtrose */
      sqrtnucl = 0.25,       /* error variance for sqrtnucl */
      factrose = v_disturb,  /* disturbance/error variance for factrose */
      factnucl = v_factnucl; /* variance of factnucl */
   pathdiagram diagram=initial useerr;
run;
```

In the PATHDIAGRAM statement, the DIAGRAM=INITIAL option requests the path diagram for the initial model specification, and the USEERR option requests that the error variables be included in the path diagram. The resulting path diagram is shown in Figure 18.18.

Figure 18.18 Initial Path Diagram for the Spleen Data: Error Terms Included

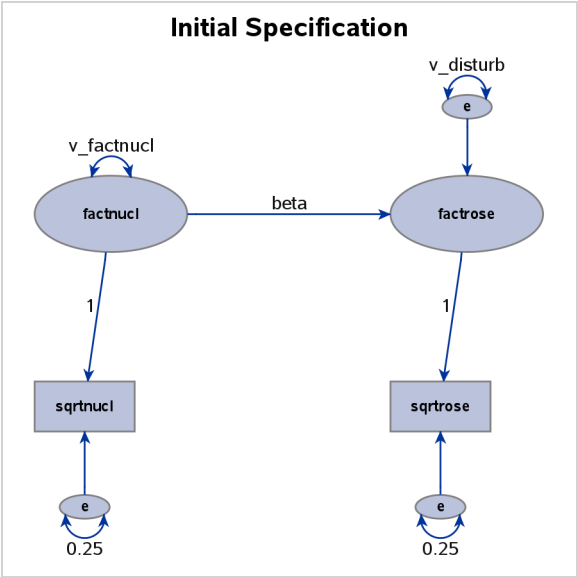

All error terms in Figure 18.18 are labeled by "e." Except for the orientation and the omission of the fixed unit constants in the paths from the error variables, this path diagram matches the style that is used in Figure 18.13.

For more information about path diagram customizations, see the PATHDIAGRAM statement and the section "Path Diagrams: Layout Algorithms, Default Settings, and Customization" on page 1784 in Chapter 33, "The CALIS Procedure." For introductory examples of creating and editing path diagrams from the CALIS procedure, see Yung (2014).

Some Measurement Models

In the section "Regression with Measurement Errors in X and Y" on page 314, outcome variables and predictor variables are assumed to have been measured with errors. In order to study the true relationships among the true scores variables, models for measurement errors are also incorporated into the estimation. The context of applications is that of regression or econometric analysis.

In the social and behavioral sciences, the same kind of model is developed in the context of test theory or item construction for measuring cognitive abilities, personality traits, or other latent variables. This kind of modeling is better-known as measurement models or confirmatory factor analysis (these two terms are interchangeable) in the psychometric field. Usually, applications in the social and behavioral sciences involve a much larger number of observed variables. This section considers some of these measurement or confirmatory factor-analytic models. For illustration purposes, only a handful of variables are used in the examples. Applications that use the PATH modeling language in PROC CALIS are described.

H4: Full Measurement Model for Lord Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex3.sas.)

Psychometric test theory involves many kinds of models that relate scores on psychological and educational tests to latent variables that represent intelligence or various underlying abilities. The following example uses data on four vocabulary tests from Lord (1957). Tests W and X have 15 items each and are administered with very liberal time limits. Tests Y and Z have 75 items and are administered under time pressure. The covariance matrix is read by the following DATA step:

```
data lord(type=cov);
   input _type_ $ _name_ $ W X Y Z;
   datalines;
n   . 649       .       .       .
cov W 86.3979   .       .       .
cov X 57.7751 86.2632   .       .
cov Y 56.8651 59.3177 97.2850   .
cov Z 58.8986 59.6683 73.8201 97.8192
;
```

The psychometric model of interest states that W and X are determined by a single common factor F_1, and Y and Z are determined by a single common factor F_2. The two common factors are expected to have a positive correlation, and it is desired to estimate this correlation. It is convenient to assume that the common factors have unit variance, so their correlation will be equal to their covariance. The error terms for all the manifest variables are assumed to be uncorrelated with each other and with the common factors. The model equations are

$$\begin{aligned} W &= \beta_W F_1 + E_W \\ X &= \beta_X F_1 + E_X \\ Y &= \beta_Y F_2 + E_Y \\ Z &= \beta_Z F_2 + E_Z \end{aligned}$$

with the following assumptions:

$$\begin{aligned} \mathrm{Var}(F_1) &= \mathrm{Var}(F_2) = 1 \\ \mathrm{Cov}(F_1, F_2) &= \rho \\ \mathrm{Cov}(E_W, E_X) &= \mathrm{Cov}(E_W, E_Y) = \mathrm{Cov}(E_W, E_Z) = \mathrm{Cov}(E_X, E_Y) \\ &= \mathrm{Cov}(E_X, E_Z) = \mathrm{Cov}(E_Y, E_Z) = \mathrm{Cov}(E_W, F_1) \\ &= \mathrm{Cov}(E_W, F_2) = \mathrm{Cov}(E_X, F_1) = \mathrm{Cov}(E_X, F_2) \\ &= \mathrm{Cov}(E_Y, F_1) = \mathrm{Cov}(E_Y, F_2) = \mathrm{Cov}(E_Z, F_1) \\ &= \mathrm{Cov}(E_Z, F_2) = 0 \end{aligned}$$

The corresponding path diagram is shown in Figure 18.19.

Figure 18.19 Path Diagram: Lord Data

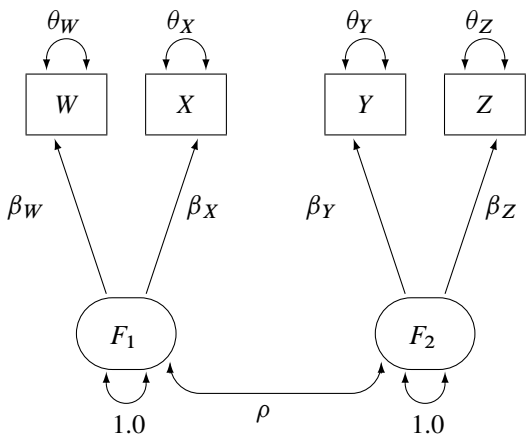

In Figure 18.19, error terms are not explicitly represented, but error variances for the observed variables are represented by double-headed arrows that point to the variables. The error variance parameters in the model are labeled with θ_W, θ_X, θ_Y, and θ_Z, respectively, for the four observed variables. In the terminology of confirmatory factor analysis, these four variables are called indicators of the corresponding latent factors F_1 and F_2.

Figure 18.19 represents the model equations clearly. It includes all the variables and the parameters in the diagram. However, sometimes researchers represent the same model with a simplified path diagram in which unconstrained parameters are not labeled, as shown in Figure 18.20.

Figure 18.20 Simplified Path Diagram: Lord Data

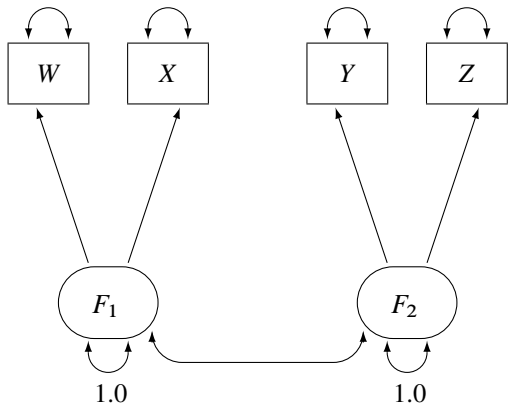

This simplified representation is also compatible with the PATH modeling language of PROC CALIS. In fact, this might be an easier starting point for modelers. With the following rules, the conversion from the path diagram to the PATH model specification is very straightforward:

- Each single-headed arrow in the path diagram is specified in the PATH statement.

- Each double-headed arrow that points to a single variable is specified in the PVAR statement.
- Each double-headed arrow that points to two distinct variables is specified in the PCOV statement.

Hence, you can convert the simplified path diagram in Figure 18.20 easily to the following PATH model specification:

```
proc calis data=lord;
   path
      W <=== F1,
      X <=== F1,
      Y <=== F2,
      Z <=== F2;
   pvar
      F1 = 1.0,
      F2 = 1.0,
      W X Y Z;
   pcov
      F1 F2;
run;
```

In this specification, you do not need to specify the parameter names. However, you do need to specify fixed values specified in the path diagram. For example, the variances of F1 and F2 are both fixed at 1 in the PVAR statement.

These fixed variances are applied solely for the purpose of model identification. Because F1 and F2 are latent variables and their scales are arbitrary, fixing their scales are necessary for model identification. Beyond these two identification constraints, none of the parameters in the model is constrained. Therefore, this is referred to as the "full" measurement model for the Lord data.

An annotated fit summary is displayed in Figure 18.21.

Figure 18.21 Fit Summary, H4: Full Model With Two Factors for Lord Data

Fit Summary	
Chi-Square	0.7030
Chi-Square DF	1
Pr > Chi-Square	0.4018
Standardized RMR (SRMR)	0.0030
Adjusted GFI (AGFI)	0.9946
RMSEA Estimate	0.0000
Bentler Comparative Fit Index	1.0000

The chi-square value is 0.7030 (df=1, p=0.4018). This indicates that you cannot reject the hypothesized model. The standardized root mean square error (SRMSR) is 0.003, which is much smaller than the conventional 0.05 value for accepting good model fit. Similarly, the RMSEA value is virtually zero, indicating an excellent fit. The adjusted GFI (AGFI) and Bentler comparative fit index are close to 1, which also indicate an excellent model fit.

The estimation results are displayed in Figure 18.22.

Figure 18.22 Estimation Results, H4: Full Model With Two Factors for Lord Data

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
W	<===	F1	_Parm1	7.50066	0.32339	23.1939	<.0001
X	<===	F1	_Parm2	7.70266	0.32063	24.0235	<.0001
Y	<===	F2	_Parm3	8.50947	0.32694	26.0273	<.0001
Z	<===	F2	_Parm4	8.67505	0.32560	26.6430	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	F1		1.00000			
	F2		1.00000			
Error	W	_Parm5	30.13796	2.47037	12.1998	<.0001
	X	_Parm6	26.93217	2.43065	11.0802	<.0001
	Y	_Parm7	24.87396	2.35986	10.5404	<.0001
	Z	_Parm8	22.56264	2.35028	9.6000	<.0001

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
F1	F2	_Parm9	0.89855	0.01865	48.1800	<.0001

All estimates are shown with estimates of standard errors in Figure 18.22. They are all statistically significant, supporting nontrivial relationships between the observed variables and the latent factors. Notice that each free parameter in the model has been named automatically in the output. For example, the path coefficient from F1 to W is named _Parm1.

Two results in Figure 18.22 are particularly interesting. First, in the table for estimates of the path coefficients, _Parm1 and _Parm2 values form one cluster, while _Parm3 and _Parm4 values from another cluster. This seems to indicate that the effects from F1 on the indicators W and X could have been the same in the population and the effects from F2 on the indicators Y and Z could also have been the same in the population. Another interesting result is the estimate for the correlation between F1 and F2 (both were set to have variance 1). The correlation estimate (_Parm9 in the Figure 18.22) is 0.8986. It is so close to 1 that you wonder whether F1 and F2 could have been the same factor in the population. These estimation results can be used to motivate additional analyses for testing the suggested constrained models against new data sets. However, for illustration purposes, the same data set is used to demonstrate the additional model fitting in the subsequent sections.

In an analysis of these data by Jöreskog and Sörbom (1979, pp. 54–56) (see also Loehlin 1987, pp. 84–87),

four hypotheses are considered:

H_1: One-factor model with parallel tests

$\rho = 1$

$\beta_W = \beta_X$ and $\mathrm{Var}(E_W) = \mathrm{Var}(E_X)$

$\beta_Y = \beta_Z$ and $\mathrm{Var}(E_Y) = \mathrm{Var}(E_Z)$

H_2: Two-factor model with parallel tests

$\beta_W = \beta_X$ and $\mathrm{Var}(E_W) = \mathrm{Var}(E_X)$

$\beta_Y = \beta_Z$ and $\mathrm{Var}(E_Y) = \mathrm{Var}(E_Z)$

H_3: Congeneric model: One factor without assuming parallel tests

$\rho = 1$

H_4: Full model: Two factors without assuming parallel tests

These hypotheses are ordered such that the latter models are less constrained. The hypothesis H_4 is the full model that has been considered in this section. The hypothesis H_3 specifies that there is really just one common factor instead of two; in the terminology of test theory, W, X, Y, and Z are said to be congeneric. Setting the correlation ρ between F1 and F2 to 1 makes the two factors indistinguishable. The hypothesis H_2 specifies that W and X have the same true scores and have equal error variance; such tests are said to be parallel. The hypothesis H_2 also requires Y and Z to be parallel. Because ρ is not constrained to 1 in H_2, two factors are assumed for this model. The hypothesis H_1 says that W and X are parallel tests, Y and Z are parallel tests, and all four tests are congeneric (with ρ also set to 1).

H3: Congeneric (One-Factor) Model for Lord Data

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex3.sas`.)

The path diagram for this congeneric (one-factor) model is shown in Figure 18.23.

Figure 18.23 H3: Congeneric (One-Factor) Model for Lord Data

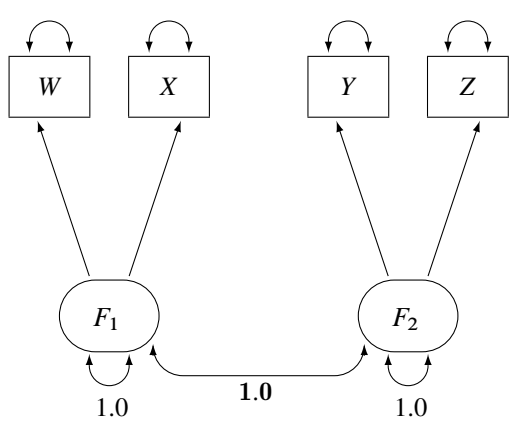

The only difference between the current path diagram in Figure 18.23 for the congeneric (one-factor) model and the preceding path diagram in Figure 18.20 for the full (two-factor) model is that the double-headed path that connects F1 and F2 is fixed to 1 in the current path diagram. Accordingly, you need to modify only slightly the preceding PROC CALIS specification to form the new model specification, as shown in the following statements:

```
proc calis data=lord;
   path
      W <=== F1,
      X <=== F1,
      Y <=== F2,
      Z <=== F2;
   pvar
      F1 = 1.0,
      F2 = 1.0,
      W X Y Z;
   pcov
      F1 F2 = 1.0;
run;
```

This specification sets the covariance between F1 and F2 to 1.0 in the PCOV statement. An annotated fit summary is displayed in Figure 18.24.

Figure 18.24 Fit Summary, H3: Congeneric (One-Factor) Model for Lord Data

Fit Summary	
Chi-Square	36.2095
Chi-Square DF	2
Pr > Chi-Square	<.0001
Standardized RMR (SRMR)	0.0277
Adjusted GFI (AGFI)	0.8570
RMSEA Estimate	0.1625
Bentler Comparative Fit Index	0.9766

The chi-square value is 36.2095 ($df = 2$, $p < 0.0001$). This indicates that you can reject the hypothesized model at the 0.01 α-level. The standardized root mean square error (SRMSR) is 0.0277, which indicates a good fit. Bentler's comparative fit index is 0.9766, which is also a good model fit. However, the adjusted GFI (AGFI) is 0.8570, which is not very impressive. Also, the RMSEA value is 0.1625, which is too large to be an acceptable model. Therefore, the congeneric model might not be the one you want to use.

The estimation results are displayed in Figure 18.25. Because the model does not fit well, the corresponding estimation results are not interpreted.

Figure 18.25 Estimation Results, H3: Congeneric (One-Factor) Model for Lord Data

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
W	<===	F1	_Parm1	7.10470	0.32177	22.0801	<.0001
X	<===	F1	_Parm2	7.26908	0.31826	22.8397	<.0001
Y	<===	F2	_Parm3	8.37344	0.32542	25.7314	<.0001
Z	<===	F2	_Parm4	8.51060	0.32409	26.2600	<.0001

Figure 18.25 *continued*

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	F1		1.00000			
	F2		1.00000			
Error	W	_Parm5	35.92111	2.41467	14.8762	<.0001
	X	_Parm6	33.42373	2.31037	14.4668	<.0001
	Y	_Parm7	27.17043	2.24621	12.0961	<.0001
	Z	_Parm8	25.38887	2.20837	11.4966	<.0001

Covariances Among Exogenous Variables					
Var1	Var2	Estimate	Standard Error	t Value	Pr > |t|
F1	F2	1.00000			

Perhaps a more natural way to specify the model under hypothesis H_3 is to use only one factor in the PATH model, as shown in the following statements:

```
proc calis data=lord;
   path
      W <=== F1,
      X <=== F1,
      Y <=== F1,
      Z <=== F1;
   pvar
      F1 = 1.0,
      W X Y Z;
run;
```

This produces essentially the same results as the specification with two factors that have perfect correlation.

H2: Two-Factor Model with Parallel Tests for Lord Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex3.sas.)

The path diagram for the two-factor model with parallel tests is shown in Figure 18.26.

Figure 18.26 H2: Two-Factor Model with Parallel Tests for Lord Data

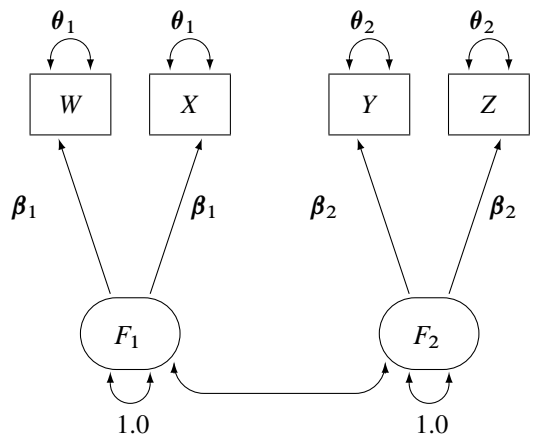

The hypothesis H_2 requires that variables or tests under each factor are "interchangeable." In terms of the measurement model, several pairs of parameters must be constrained to have equal estimates. That is, under the parallel-test model W and X should have the same effect or path coefficient β_1 from their common factor F1, and they should also have the same measurement error variance θ_1. Similarly, Y and Z should have the same effect or path coefficient β_2 from their common factor F2, and they should also have the same measurement error variance θ_2. These constraints are labeled in Figure 18.26.

You can impose each of these equality constraints by giving the same name for the parameters involved in the PATH model specification. The following statements specify the path diagram in Figure 18.26:

```
proc calis data=lord;
   path
      W <=== F1   = beta1,
      X <=== F1   = beta1,
      Y <=== F2   = beta2,
      Z <=== F2   = beta2;
   pvar
      F1  = 1.0,
      F2  = 1.0,
      W X = 2 * theta1,
      Y Z = 2 * theta2;
   pcov
      F1 F2;
run;
```

Note that the specification **2*theta1** in the PVAR statement means that theta1 is specified twice for the error variances of the two variables W and X. Similarly for the specification **2*theta2**. An annotated fit summary is displayed in Figure 18.27.

Figure 18.27 Fit Summary, H2: Two-Factor Model with Parallel Tests for Lord Data

Fit Summary	
Chi-Square	1.9335
Chi-Square DF	5
Pr > Chi-Square	0.8583
Standardized RMR (SRMR)	0.0076
Adjusted GFI (AGFI)	0.9970
RMSEA Estimate	0.0000
Bentler Comparative Fit Index	1.0000

The chi-square value is 1.9335 (df=5, p=0.8583). This indicates that you cannot reject the hypothesized model H2. The standardized root mean square error (SRMSR) is 0.0076, which indicates a very good fit. Bentler's comparative fit index is 1.0000. The adjusted GFI (AGFI) is 0.9970, and the RMSEA is close to zero. All results indicate that this is a good model for the data.

The estimation results are displayed in Figure 18.28.

Figure 18.28 Estimation Results, H2: Two-Factor Model with Parallel Tests for Lord Data

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
W	**<===**	**F1**	beta1	7.60099	0.26844	28.3158	<.0001
X	**<===**	**F1**	beta1	7.60099	0.26844	28.3158	<.0001
Y	**<===**	**F2**	beta2	8.59186	0.27967	30.7215	<.0001
Z	**<===**	**F2**	beta2	8.59186	0.27967	30.7215	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	**F1**		1.00000			
	F2		1.00000			
Error	**W**	theta1	28.55545	1.58641	18.0000	<.0001
	X	theta1	28.55545	1.58641	18.0000	<.0001
	Y	theta2	23.73200	1.31844	18.0000	<.0001
	Z	theta2	23.73200	1.31844	18.0000	<.0001

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
F1	**F2**	_Parm1	0.89864	0.01865	48.1801	<.0001

Notice that because you explicitly specify the parameter names for the path coefficients (that is, beta1 and beta2), they are used in the output shown in Figure 18.28. The correlation between F1 and F2 is 0.8987, which is a very high correlation that suggests F1 and F2 might have been the same factor in the population. The next section sets this value to one so that the current model becomes a one-factor model with parallel tests.

H1: One-Factor Model with Parallel Tests for Lord Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex3.sas.)

The path diagram for the one-factor model with parallel tests is shown in Figure 18.29.

Figure 18.29 H1: One-Factor Model with Parallel Tests for Lord Data

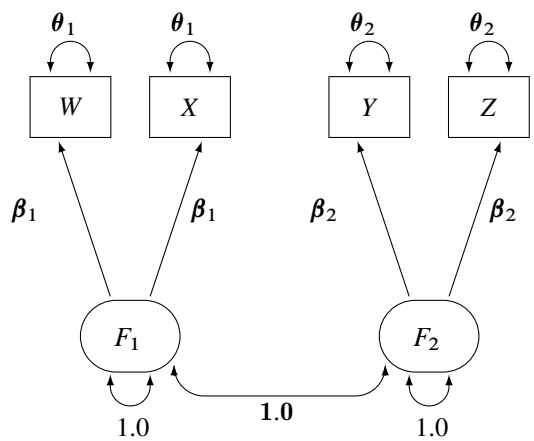

The hypothesis H_1 differs from H_2 in that F1 and F2 have a perfect correlation in H_1. This is indicated by the fixed value 1.0 for the double-headed path that connects F1 and F2 in Figure 18.29. Again, you need only minimal modification of the preceding specification for H_2 to specify the path diagram in Figure 18.29, as shown in the following statements:

```
proc calis data=lord;
   path
      W <=== F1   = beta1,
      X <=== F1   = beta1,
      Y <=== F2   = beta2,
      Z <=== F2   = beta2;
   pvar
      F1  = 1.0,
      F2  = 1.0,
      W X = 2 * theta1,
      Y Z = 2 * theta2;
   pcov
      F1 F2 = 1.0;
run;
```

The only modification of the preceding specification is in the PCOV statement, where you put a constant 1 for the covariance between F1 and F2. An annotated fit summary is displayed in Figure 18.30.

Figure 18.30 Fit Summary, H1: One-Factor Model with Parallel Tests for Lord Data

Fit Summary	
Chi-Square	37.3337
Chi-Square DF	6
Pr > Chi-Square	<.0001
Standardized RMR (SRMR)	0.0286
Adjusted GFI (AGFI)	0.9509
RMSEA Estimate	0.0898
Bentler Comparative Fit Index	0.9785

The chi-square value is 37.3337 (df=6, p<0.0001). This indicates that you can reject the hypothesized model H1 at the 0.01 α-level. The standardized root mean square error (SRMSR) is 0.0286, the adjusted GFI (AGFI) is 0.9509, and Bentler's comparative fit index is 0.9785. All these indicate good model fit. However, the RMSEA is 0.0898, which does not support an acceptable model for the data.

The estimation results are displayed in Figure 18.31.

Figure 18.31 Estimation Results, H1: One-Factor Model with Parallel Tests for Lord Data

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
W	**<===**	**F1**	beta1	7.18623	0.26598	27.0180	<.0001
X	**<===**	**F1**	beta1	7.18623	0.26598	27.0180	<.0001
Y	**<===**	**F2**	beta2	8.44198	0.28000	30.1494	<.0001
Z	**<===**	**F2**	beta2	8.44198	0.28000	30.1494	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	**F1**		1.00000			
	F2		1.00000			
Error	**W**	theta1	34.68865	1.64634	21.0701	<.0001
	X	theta1	34.68865	1.64634	21.0701	<.0001
	Y	theta2	26.28513	1.39955	18.7812	<.0001
	Z	theta2	26.28513	1.39955	18.7812	<.0001

Covariances Among Exogenous Variables

Var1	Var2	Estimate	Standard Error	t Value	Pr > |t|
F1	**F2**	1.00000			

The goodness-of-fit tests for the four hypotheses are summarized in the following table.

Hypothesis	Number of Parameters	χ^2	Degrees of Freedom	p-value	$\hat{\rho}$
H_1	4	37.33	6	< .0001	1.0
H_2	5	1.93	5	0.8583	0.8986
H_3	8	36.21	2	< .0001	1.0
H_4	9	0.70	1	0.4018	0.8986

Recall that the estimates of ρ for H_2 and H_4 are almost identical, about 0.90, indicating that the speeded and unspeeded tests are measuring almost the same latent variable. However, when ρ was set to 1 in H_1 and H_3 (both one-factor models), both hypotheses were rejected. Hypotheses H_2 and H_4 (both two-factor models) seem to be consistent with the data. Since H_2 is obtained by adding four constraints (for the requirement of parallel tests) to H_4 (the full model), you can test H_2 versus H_4 by computing the differences of the chi-square statistics and their degrees of freedom, yielding a chi-square of 1.23 with four degrees of freedom, which is obviously not significant. In a sense, the chi-square difference test means that representing the data by H_2 would not be significantly worse than representing the data by H_4. In addition, because H_2 offers a more precise description of the data (with the assumption of parallel tests) than H_4, it should be chosen because of its simplicity. In conclusion, the two-factor model with parallel tests provides the best explanation of the data.

The FACTOR and RAM Modeling Languages

In the section "Some Measurement Models" on page 330, you use the path diagram to represent the measurement models for data with cognitive tests and then you use the PATH modeling language to specify the model in PROC CALIS. You could have used other types of modeling languages for specifying the same model. In this section, the FACTOR and the RAM modeling languages are illustrated.

Specifying the Full Measurement Model (H4) by the FACTOR Modeling Language: Lord Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex3.sas.)

The measurement models described in the section "Some Measurement Models" on page 330 are also known as confirmatory factor models. PROC CALIS has a specific modeling language, called FACTOR, for confirmatory factor models. You can use this modeling language for both exploratory and confirmatory factor analysis.

For example, the full measurement model H4 in the section "H4: Full Measurement Model for Lord Data" on page 331 can be specified equivalently by the FACTOR modeling language with the following statements:

```
proc calis data=lord;
   factor
      F1  ===>  W  X,
      F2  ===>  Y  Z;
   pvar
```

```
      F1 = 1.0,
      F2 = 1.0,
      W X Y Z;
   cov
      F1 F2;
run;
```

In the specification, you use the FACTOR statement to invoke the FACTOR modeling language. In the FACTOR statement, you specify the paths from the latent factors to the measurement indicators. For example, F1 has two paths to its indicators, W and X. Similarly, F2 has two paths to its indicators, Y and Z. Next, you use the PVAR statement to specify the variances, which is exactly the same way you use the PATH model specification in the section "H4: Full Measurement Model for Lord Data" on page 331. Lastly, you use the COV statement to specify the covariance among the factors, much like you use the PCOV statement to specify the same covariance in the PATH model specification.

Given the same confirmatory factor model, there is a major difference between the paths specified by the PATH statement and the paths specified by the FACTOR statement. In the FACTOR statement, each path must start with a latent factor followed by a right arrow and the variable list. In the PATH statement, each path can start or end with an observed or latent variable, and the direction of the arrow can be left or right.

The fit summary table for the FACTOR model is shown in Figure 18.32:

Figure 18.32 Fit Summary of the Full Confirmatory Factor Model for Lord Data

Fit Summary	
Chi-Square	0.7030
Chi-Square DF	1
Pr > Chi-Square	0.4018
Standardized RMR (SRMR)	0.0030
Adjusted GFI (AGFI)	0.9946
RMSEA Estimate	0.0000
Bentler Comparative Fit Index	1.0000

This is exactly the same fit summary as shown in Figure 18.21, which is for the PATH model specification. Therefore, this confirms that the same model is being fit by the FACTOR model specification.

The estimation results are shown in Figure 18.33.

Figure 18.33 Estimation Results of Full Confirmatory Factor Model for Lord Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	F1	F2
W	7.5007 0.3234 23.1939 <.0001 [_Parm1]	0
X	7.7027 0.3206 24.0235 <.0001 [_Parm2]	0
Y	0	8.5095 0.3269 26.0273 <.0001 [_Parm3]
Z	0	8.6751 0.3256 26.6430 <.0001 [_Parm4]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value		
	F1	F2
F1	1.0000	0.8986 0.0186 48.1800 <.0001 [_Parm9]
F2	0.8986 0.0186 48.1800 <.0001 [_Parm9]	1.0000

Error Variances

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
W	_Parm5	30.13796	2.47037	12.1998	<.0001
X	_Parm6	26.93217	2.43065	11.0802	<.0001
Y	_Parm7	24.87396	2.35986	10.5404	<.0001
Z	_Parm8	22.56264	2.35028	9.6000	<.0001

Again, these are the same estimates as those shown in Figure 18.22, which is for the PATH model specification. The FACTOR results displayed in Figure 18.33 are arranged differently though. No paths are shown there. The relationships between the latent factors and its indicators are shown in matrix form. The factor variance and covariances are also shown in matrix form.

Specifying the Parallel Tests Model (H2) by the FACTOR Modeling Language: Lord Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex3.sas.)

In the section "H2: Two-Factor Model with Parallel Tests for Lord Data" on page 337, you fit a two-factor model with parallel tests for the Lord data by the PATH modeling language in PROC CALIS. Some paths and error variance are constrained under the PATH model. You can also specify this parallel tests model by the FACTOR modeling language, as shown in the following statements:

```
proc calis data=lord;
   factor
      F1  ===>  W  X    = 2 * beta1,
      F2  ===>  Y  Z    = 2 * beta2;
   pvar
      F1  = 1.0,
      F2  = 1.0,
      W X = 2 * theta1,
      Y Z = 2 * theta2;
   cov
      F1 F2;
run;
```

In this specification, you specify some parameters explicitly. You apply the parameter beta1 to the loadings of both W and X on F1. This means that F1 has the same amount of effect on W and X. Similarly, you apply the parameter beta2 to the loadings of Y and Z on F2. The constraints on the error variances for W, X, Y, and Z in this FACTOR model specification are done in the same way as in the PATH model specification in the section "H2: Two-Factor Model with Parallel Tests for Lord Data" on page 337.

The fit summary table for this parallel tests model is shown in Figure 18.34.

Figure 18.34 Fit Summary of the Confirmatory Factor Model with Parallel Tests for Lord Data

Fit Summary	
Chi-Square	1.9335
Chi-Square DF	5
Pr > Chi-Square	0.8583
Standardized RMR (SRMR)	0.0076
Adjusted GFI (AGFI)	0.9970
RMSEA Estimate	0.0000
Bentler Comparative Fit Index	1.0000

All the fit indices shown in Figure 18.34 for the FACTOR model match the corresponding PATH model results displayed in Figure 18.27. All the estimation results in Figure 18.35 for the FACTOR model are the same as those for the corresponding PATH model in Figure 18.28.

Figure 18.35 Estimation Results of the Confirmatory Factor Model with Parallel Tests for Lord Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	F1	F2
W	7.6010 0.2684 28.3158 <.0001 [beta1]	0
X	7.6010 0.2684 28.3158 <.0001 [beta1]	0
Y	0	8.5919 0.2797 30.7215 <.0001 [beta2]
Z	0	8.5919 0.2797 30.7215 <.0001 [beta2]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value		
	F1	F2
F1	1.0000	0.8986 0.0187 48.1801 <.0001 [_Parm1]
F2	0.8986 0.0187 48.1801 <.0001 [_Parm1]	1.0000

Error Variances

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
W	theta1	28.55545	1.58641	18.0000	<.0001
X	theta1	28.55545	1.58641	18.0000	<.0001
Y	theta2	23.73200	1.31844	18.0000	<.0001
Z	theta2	23.73200	1.31844	18.0000	<.0001

Specifying the Parallel Tests Model (H2) by the RAM Modeling Language: Lord Data

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex3.sas`.)

In the preceding section, you use the FACTOR modeling language of PROC CALIS to specify the parallel tests model. This model has also been specified by the PATH modeling language in the section "H2: Two-Factor Model with Parallel Tests for Lord Data" on page 337. The two specifications are equivalent; they lead to the same model fitting and estimation results. The main reason for providing two different types of modeling languages in PROC CALIS is that different researchers come from different fields of applications. Some researchers might be more comfortable with the confirmatory factor tradition, and some might equate structural equation models with path diagrams for variables.

PROC CALIS has still another modeling language that is closely related to the path diagram representation: the RAM model specification. In this section, the parallel tests model (H2) described in "H2: Two-Factor Model with Parallel Tests for Lord Data" on page 337 is used to illustrate the RAM model specification in PROC CALIS.

The path diagram for this model is reproduced in Figure 18.36.

Figure 18.36 H2: Two-Factor Model with Parallel Tests for Lord Data

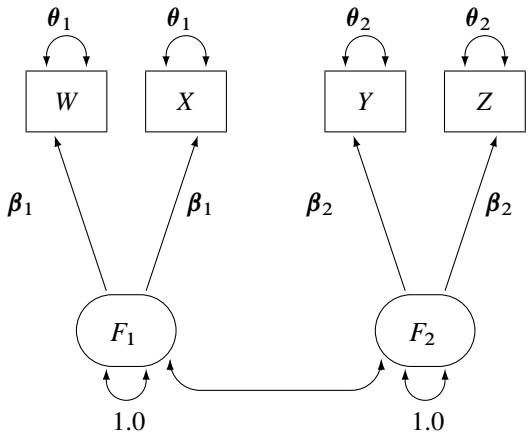

The path diagram in Figure 18.36 can be readily transcribed into the RAM model specification by following these simple rules:

- Each single- or double- headed path corresponds to an entry in the RAM model specification.
- The single-headed paths are specified with the _A_ path type or matrix keyword.
- The double-headed paths are specified with the _P_ path type or matrix keyword.

At this point, you do not need to define the RAM model matrices _A_ and _P_, as long as you recognize that they are used as keywords to distinguish different path types. There are 11 single- or double- headed paths in Figure 18.36, and therefore you expect to specify these 11 elements in the RAM model, as shown in the following statements:

```
proc calis data=lord;
   ram var = W X Y Z F1 F2, /* W=1, X=2, Y=3, Z=4, F1=5, F2=6*/
       _A_   1  5  beta1,
       _A_   2  5  beta1,
       _A_   3  6  beta2,
       _A_   4  6  beta2,
       _P_   5  5  1.0,
       _P_   6  6  1.0,
       _P_   1  1  theta1,
       _P_   2  2  theta1,
       _P_   3  3  theta2,
       _P_   4  4  theta2,
       _P_   5  6  ;
run;
```

In this specification, the RAM statement invokes the RAM modeling language. The first option is the VAR= option where you specify the variables, observed and latent, in the model. The order in the VAR= variable list represents the order of these variables in the RAM model matrices. For this example, W is 1, X is 2, and so on. Next, you specify 11 RAM entries for the 11 path elements in the path diagram shown in Figure 18.36.

The first four entries are for the single-headed paths. They all begin with the _A_ keyword. In each of these _A_ entries, you specify the variable number of the outcome variable (being pointed at), and then the variable number of the predictor variable. At the end of the entry, you can specify a parameter name, a fixed value, an initial value, or nothing. In this example, all the _A_ entries are specified with parameter names. The first two paths are constrained because they use the same parameter name beta1. The next two paths are constrained because they use the same parameter name beta2.

The rest of the RAM entries in the example are of the _P_ type, which is for the specification of variances or covariances in the RAM model (the double-headed arrows in the path diagram). The _P_ entry with [5,5] is for the variance of the fifth variable, F1, on the VAR= list. This variance is fixed at 1.0 in the model, and so is the variance of the sixth variable, F2, in the next _P_ entry.

The next four _P_ entries are for the specification of error variances of the observed variables W, X, Y, and Z. You use the desired parameter names for constraining these parameters, as required in the parallel test model.

The last _P_ entry in the RAM statement is for the covariance between the fifth variable (F1) and the sixth variable (F2). You specify neither a parameter name nor a fixed value at the end of this entry. By default, this empty parameter specification is treated as a free parameter in the model. A parameter name for this entry is generated by PROC CALIS.

The fit summary for this RAM model is shown in Figure 18.37, and the estimation results are shown in Figure 18.38.

Figure 18.37 Fit Summary of RAM Model with Parallel Tests for Lord Data

Fit Summary	
Chi-Square	1.9335
Chi-Square DF	5
Pr > Chi-Square	0.8583
Standardized RMR (SRMR)	0.0076
Adjusted GFI (AGFI)	0.9970
RMSEA Estimate	0.0000
Bentler Comparative Fit Index	1.0000

Figure 18.38 Estimation Results of RAM Model with Parallel Tests for Lord Data

RAM Pattern and Estimates								
Matrix	Row		Column		Parameter	Estimate	Standard Error	t Value	Pr > |t|
_A_ (1)	W	1	F1	5	beta1	7.60099	0.26844	28.3158	<.0001
	X	2	F1	5	beta1	7.60099	0.26844	28.3158	<.0001
	Y	3	F2	6	beta2	8.59186	0.27967	30.7215	<.0001
	Z	4	F2	6	beta2	8.59186	0.27967	30.7215	<.0001
_P_ (2)	F1	5	F1	5		1.00000			
	F2	6	F2	6		1.00000			
	W	1	W	1	theta1	28.55545	1.58641	18.0000	<.0001
	X	2	X	2	theta1	28.55545	1.58641	18.0000	<.0001
	Y	3	Y	3	theta2	23.73200	1.31844	18.0000	<.0001
	Z	4	Z	4	theta2	23.73200	1.31844	18.0000	<.0001
	F1	5	F2	6	_Parm1	0.89864	0.01865	48.1801	<.0001

Again, the model fit and the estimation results match those from the PATH model specification in Figure 18.27 and Figure 18.28, and those from the FACTOR model specification in Figure 18.34 and Figure 18.35.

A Combined Measurement-Structural Model

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas`.)

To illustrate a more complex model, this example uses some well-known data from Haller and Butterworth (1960). Various models and analyses of these data are given by Duncan, Haller, and Portes (1968); Jöreskog and Sörbom (1988); Loehlin (1987).

The study concerns the career aspirations of high school students and how these aspirations are affected by close friends. The data are collected from 442 seventeen-year-old boys in Michigan. There are 329 boys in the sample who named another boy in the sample as a best friend. The data from these 329 boys paired with the data from their best friends are analyzed.

The method of data collection introduces two statistical problems. First, restricting the analysis to boys whose best friends are in the original sample causes the reduced sample to be biased. Second, since the data from a given boy might appear in two or more observations, the observations are not independent. Therefore, any statistical conclusions should be considered tentative. It is difficult to accurately assess the effects of the dependence of the observations on the analysis, but it could be argued on intuitive grounds that since each observation has data from two boys and since it seems likely that many of the boys appear in the data set at least twice, the effective sample size might be as small as half of the reported 329 observations.

The correlation matrix, taken from Jöreskog and Sörbom (1988), is shown in the following DATA step:

```
title 'Peer Influences on Aspiration: Haller & Butterworth (1960)';
data aspire(type=corr);
   _type_='corr';
   input _name_ $ riq rpa rses roa rea fiq fpa fses foa fea;
   label riq='Respondent: Intelligence'
```

```
          rpa='Respondent: Parental Aspiration'
          rses='Respondent: Family SES'
          roa='Respondent: Occupational Aspiration'
          rea='Respondent: Educational Aspiration'
          fiq='Friend: Intelligence'
          fpa='Friend: Parental Aspiration'
          fses='Friend: Family SES'
          foa='Friend: Occupational Aspiration'
          fea='Friend: Educational Aspiration';
   datalines;
riq   1.      .      .      .      .      .      .      .      .      .
rpa   .1839  1.      .      .      .      .      .      .      .      .
rses  .2220  .0489  1.      .      .      .      .      .      .      .
roa   .4105  .2137  .3240  1.      .      .      .      .      .      .
rea   .4043  .2742  .4047  .6247  1.      .      .      .      .      .
fiq   .3355  .0782  .2302  .2995  .2863  1.      .      .      .      .
fpa   .1021  .1147  .0931  .0760  .0702  .2087   1.      .      .      .
fses  .1861  .0186  .2707  .2930  .2407  .2950  -.0438  1.      .      .
foa   .2598  .0839  .2786  .4216  .3275  .5007   .1988  .3607  1.      .
fea   .2903  .1124  .3054  .3269  .3669  .5191   .2784  .4105  .6404  1.
;
```

Career Aspiration (Analysis 1)

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas.)

The model analyzed by Jöreskog and Sörbom (1988) is displayed in the path diagram in Figure 18.39.

Figure 18.39 Path Diagram: Career Aspiration – Jöreskog and Sörbom (1988)

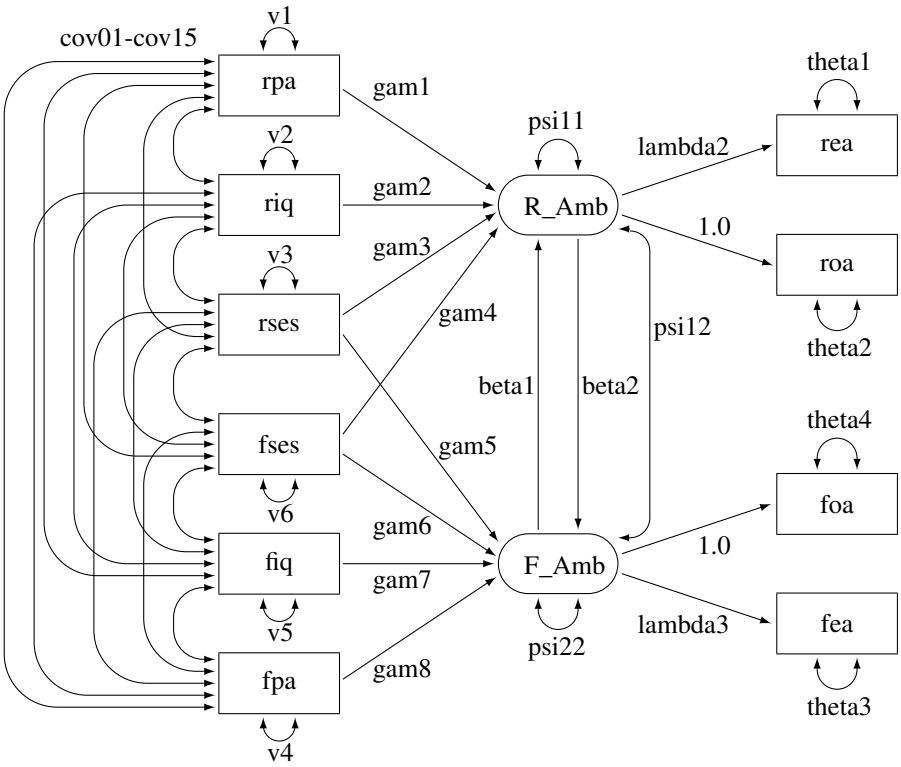

Two latent variables, R_Amb and F_Amb, represent the respondent's level of ambition and his best friend's level of ambition, respectively. The model states that the respondent's ambition (R_Amb) is determined by his intelligence (riq) and socioeconomic status (rses), his perception of his parents' aspiration for him (rpa), and his friend's socioeconomic status (fses) and ambition (F_Amb). It is assumed that his friend's intelligence (fiq) and parental aspiration (fpa) affect the respondent's ambition only indirectly through the friend's ambition (F_Amb). Ambition is indicated by the manifest variables of occupational (roa) and educational aspiration (rea), which are assumed to have uncorrelated residuals. The path coefficient from ambition to occupational aspiration is set to 1.0 to determine the scale of the ambition latent variable.

The path diagram shown in Figure 18.39 appears to be very complicated. Sometimes when researchers draw their path diagram with a lot of variables, they omit the covariances among exogenous variables for overall clarity. For example, the double-headed paths that represent cov01–cov15 in Figure 18.39 can be omitted. In addition, unconstrained variance and error variance parameters in the path diagram could be omitted without losing the pertinent information in the path diagram. For example, variance parameters v1–v6 and error variance parameters theta1–theta4 in Figure 18.39 detract from the main focus on the functional relationships of the model.

These omissions in the path diagram are in fact inconsequential when you transcribe them into the PATH model in PROC CALIS. The reason is that PROC CALIS employs several useful default parameterization rules that make the model specification process much easier and more intuitive. Here are the sets of default covariance structure parameters in the PATH modeling language:

- variances for all exogenous (observed or latent) variables

- error variances of all endogenous (observed or latent) variables
- covariances among all exogenous (observed or latent, excluding error) variables

For example, these rules for setting default covariance structure parameters mean that the following sets of parameters in Figure 18.39 are optional in the path diagram representation and in the corresponding PATH model specification:

- v1–v6
- theta1–theta4, psi11, and psi22
- cov01–cov15

Note that the double-headed path labeled with psi12, which is a covariance parameter among error terms for R_Amb and F_Amb, is not a default parameter. As a result, it must be represented in the path diagram and in the PATH model specification.

Another simplification is to omit the unconstrained parameter names in the path diagram. In the PATH model specification, an "unnamed" parameter is a free parameter by default—there is no need to give unique names to denote free parameters. With all the mentioned simplifications, you can depict your path diagram simply as the one in Figure 18.40.

Figure 18.40 Simplified Path Diagram for Career Aspiration : Analysis 1

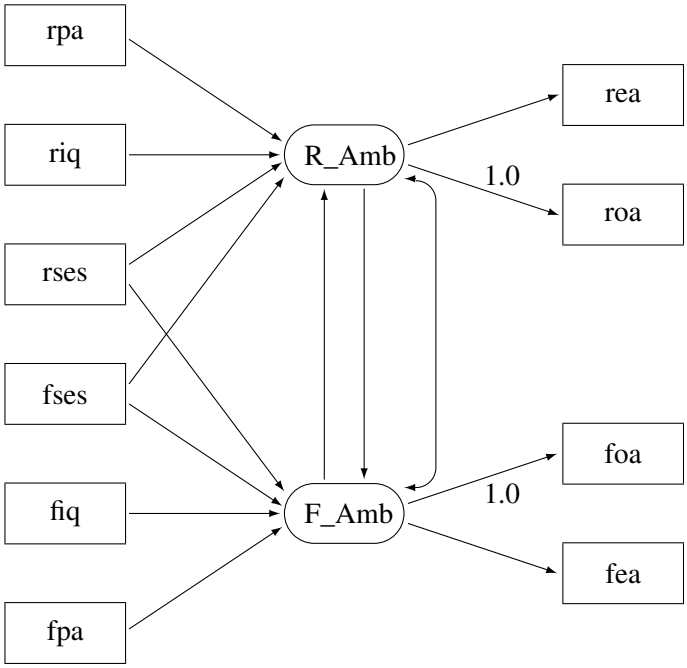

The simplified path diagram in Figure 18.40 is readily transcribed into the PATH model as shown in the following statements:

```
proc calis data=aspire nobs=329;
   path
      /* structural model of influences */
      R_Amb <=== rpa     ,
      R_Amb <=== riq     ,
      R_Amb <=== rses    ,
      R_Amb <=== fses    ,
      F_Amb <=== rses    ,
      F_Amb <=== fses    ,
      F_Amb <=== fiq     ,
      F_Amb <=== fpa     ,
      R_Amb <=== F_Amb   ,
      F_Amb <=== R_Amb   ,

      /* measurement model for aspiration */
      rea <===  R_Amb      ,
      roa <===  R_Amb  = 1.,
      foa <===  F_Amb  = 1.,
      fea <===  F_Amb    ;
   pcov
      R_Amb  F_Amb;
run;
```

Again, because you have 15 paths (single- or double- headed) in the path diagram, you expect that there are 15 entries in the PATH and the PCOV statements. Essentially, in this PATH model specification you specify all the functional relationships (single-headed arrows) in the path diagram and the covariance of error terms (double-headed arrows) for R_Amb and F_Amb.

Since this TYPE=CORR data set does not contain an observation with _TYPE_=N giving the sample size, it is necessary to specify the NOBS= option in the PROC CALIS statement.

The fit summary is displayed in Figure 18.41, and the estimation results are displayed in Figure 18.42.

Figure 18.41 Career Aspiration Data: Fit Summary for Analysis 1

Fit Summary	
Chi-Square	26.6972
Chi-Square DF	15
Pr > Chi-Square	0.0313
Standardized RMR (SRMR)	0.0202
Adjusted GFI (AGFI)	0.9428
RMSEA Estimate	0.0488
Akaike Information Criterion	106.6972
Schwarz Bayesian Criterion	258.5395
Bentler Comparative Fit Index	0.9859

The model fit chi-square value is 26.6972 (df=15, p=0.0313). From the hypothesis testing point of view, this result says that this is an extreme sample given the model is true; therefore, the model should be rejected. But in social and behavioral sciences, you rarely abandon a model purely on the ground of chi-square significance test. The main reason is that you might only need to find a model that is approximately true, but the hypothesis testing framework is for testing exact model representation in the population. To determine whether a model is good or bad, you usually consult other fit indices. Several fit indices are shown in Figure 18.41.

The standardized RMSR is 0.0202. The RMSEA value is 0.0488. Both of these indices are smaller than 0.05, which indicate good model fit by convention. The adjusted GFI is 0.9428, and the comparative fit index is 0.9859. Again, values greater than 0.9 for these indices indicate good model fit by convention. Therefore, you can conclude that this is a good model for the data. Akaike's information criterion (AIC) and the Schwarz Bayesian criterion are also shown. You cannot interpret these values directly, but they are useful for model comparison given the same data, as shown in later sections.

Figure 18.42 Career Aspiration Data: Estimation Results for Analysis 1

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
R_Amb	**<===**	**rpa**	_Parm01	0.16122	0.03879	4.1560	<.0001
R_Amb	**<===**	**riq**	_Parm02	0.24965	0.04398	5.6763	<.0001
R_Amb	**<===**	**rses**	_Parm03	0.21840	0.04420	4.9415	<.0001
R_Amb	**<===**	**fses**	_Parm04	0.07184	0.04971	1.4453	0.1484
F_Amb	**<===**	**rses**	_Parm05	0.05754	0.04812	1.1956	0.2318
F_Amb	**<===**	**fses**	_Parm06	0.21278	0.04169	5.1042	<.0001
F_Amb	**<===**	**fiq**	_Parm07	0.32451	0.04352	7.4562	<.0001
F_Amb	**<===**	**fpa**	_Parm08	0.14832	0.03645	4.0696	<.0001
R_Amb	**<===**	**F_Amb**	_Parm09	0.19816	0.10228	1.9374	0.0527
F_Amb	**<===**	**R_Amb**	_Parm10	0.21893	0.11125	1.9680	0.0491
rea	**<===**	**R_Amb**	_Parm11	1.06268	0.09014	11.7894	<.0001
roa	**<===**	**R_Amb**		1.00000			
foa	**<===**	**F_Amb**		1.00000			
fea	**<===**	**F_Amb**	_Parm12	1.07558	0.08131	13.2287	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	**riq**	_Add01	1.00000	0.07809	12.8062	<.0001
	rpa	_Add02	1.00000	0.07809	12.8062	<.0001
	rses	_Add03	1.00000	0.07809	12.8062	<.0001
	fiq	_Add04	1.00000	0.07809	12.8062	<.0001
	fpa	_Add05	1.00000	0.07809	12.8062	<.0001
	fses	_Add06	1.00000	0.07809	12.8062	<.0001
Error	**roa**	_Add07	0.41215	0.05122	8.0458	<.0001
	rea	_Add08	0.33614	0.05210	6.4519	<.0001
	foa	_Add09	0.40460	0.04618	8.7606	<.0001
	fea	_Add10	0.31120	0.04593	6.7759	<.0001
	R_Amb	_Add11	0.28099	0.04623	6.0778	<.0001
	F_Amb	_Add12	0.22806	0.03850	5.9233	<.0001

Figure 18.42 *continued*

Covariances Among Exogenous Variables						
Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
rpa	riq	_Add13	0.18390	0.05614	3.2756	0.0011
rses	riq	_Add14	0.22200	0.05656	3.9250	<.0001
rses	rpa	_Add15	0.04890	0.05528	0.8846	0.3764
fiq	riq	_Add16	0.33550	0.05824	5.7606	<.0001
fiq	rpa	_Add17	0.07820	0.05538	1.4120	0.1580
fiq	rses	_Add18	0.23020	0.05666	4.0628	<.0001
fpa	riq	_Add19	0.10210	0.05550	1.8395	0.0658
fpa	rpa	_Add20	0.11470	0.05558	2.0638	0.0390
fpa	rses	_Add21	0.09310	0.05545	1.6789	0.0932
fpa	fiq	_Add22	0.20870	0.05641	3.7000	0.0002
fses	riq	_Add23	0.18610	0.05616	3.3135	0.0009
fses	rpa	_Add24	0.01860	0.05523	0.3368	0.7363
fses	rses	_Add25	0.27070	0.05720	4.7323	<.0001
fses	fiq	_Add26	0.29500	0.05757	5.1244	<.0001
fses	fpa	_Add27	-0.04380	0.05527	-0.7925	0.4281

In Figure 18.42, some of the paths do not show significance. That is, fses does not seem to be a good indicator of a respondent's ambition R_Amb and rses does not seem to be a good indicator of a friend's ambition F_Amb. The t values are 1.445 and 1.195, respectively, which are much smaller than the nominal 1.96 value at the 0.05 α-level of significance. Other paths are either significant or marginally significant.

You should be very cautious about interpreting the current analysis results for two reasons. First, as mentioned previously the data consist of dependent observations, and it was not certain how the issue could have been addressed beyond setting the sample size to half of the actual size. Second, structural equation modeling methodology is mainly applicable when you analyze covariance structures. When you input a correlation matrix for analysis, there is no guarantee that the statistical tests and standard error estimates are applicable. You should view the interpretations made here just as an exercise of applying structural equation modeling.

In Output 18.42, all parameter names are generated by PROC CALIS. Alternatively, you can also name these parameters in your PATH model specification. The following shows a PATH model specification that corresponds to the complete path diagram shown in Figure 18.39:

```
proc calis data=aspire nobs=329;
   path
      /* structural model of influences */
      rpa    ===>  R_Amb   =  gam1,
      riq    ===>  R_Amb   =  gam2,
      rses   ===>  R_Amb   =  gam3,
      fses   ===>  R_Amb   =  gam4,
      rses   ===>  F_Amb   =  gam5,
      fses   ===>  F_Amb   =  gam6,
      fiq    ===>  F_Amb   =  gam7,
      fpa    ===>  F_Amb   =  gam8,
      F_Amb  ===>  R_Amb   = beta1,
      R_Amb  ===>  F_Amb   = beta2,

      /* measurement model for aspiration */
```

```
      R_Amb  ===>  rea     = lambda2,
      R_Amb  ===>  roa     = 1.,
      F_Amb  ===>  foa     = 1.,
      F_Amb  ===>  fea     = lambda3;
   pvar
      R_Amb = psi11,
      F_Amb = psi22,
      rpa riq rses fpa fiq fses = v1-v6,
      rea roa fea foa = theta1-theta4;
   pcov
      R_Amb F_Amb = psi12,
      rpa riq rses fpa fiq fses = cov01-cov15;
run;
```

In this specification, the names of the parameters correspond to those used by Jöreskog and Sörbom (1988). Compared with the simplified version of the same model specification, you name 27 more parameters in the current specification. You have to be careful with this many parameters. If you inadvertently repeat the use of some parameter names, you will have unexpected constraints in the model.

The results from this analysis are displayed in Figure 18.43.

Figure 18.43 Career Aspiration Data: Estimation Results with Designated Parameter Names (Analysis 1)

PATH List						
Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
rpa	===>	R_Amb	gam1	0.16122	0.03879	4.1560	<.0001
riq	===>	R_Amb	gam2	0.24965	0.04398	5.6763	<.0001
rses	===>	R_Amb	gam3	0.21840	0.04420	4.9415	<.0001
fses	===>	R_Amb	gam4	0.07184	0.04971	1.4453	0.1484
rses	===>	F_Amb	gam5	0.05754	0.04812	1.1956	0.2318
fses	===>	F_Amb	gam6	0.21278	0.04169	5.1042	<.0001
fiq	===>	F_Amb	gam7	0.32451	0.04352	7.4562	<.0001
fpa	===>	F_Amb	gam8	0.14832	0.03645	4.0696	<.0001
F_Amb	===>	R_Amb	beta1	0.19816	0.10228	1.9374	0.0527
R_Amb	===>	F_Amb	beta2	0.21893	0.11125	1.9680	0.0491
R_Amb	===>	rea	lambda2	1.06268	0.09014	11.7894	<.0001
R_Amb	===>	roa		1.00000			
F_Amb	===>	foa		1.00000			
F_Amb	===>	fea	lambda3	1.07558	0.08131	13.2287	<.0001

Figure 18.43 *continued*

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**R_Amb**	psi11	0.28099	0.04623	6.0778	<.0001
	F_Amb	psi22	0.22806	0.03850	5.9233	<.0001
Exogenous	**rpa**	v1	1.00000	0.07809	12.8062	<.0001
	riq	v2	1.00000	0.07809	12.8062	<.0001
	rses	v3	1.00000	0.07809	12.8062	<.0001
	fpa	v4	1.00000	0.07809	12.8062	<.0001
	fiq	v5	1.00000	0.07809	12.8062	<.0001
	fses	v6	1.00000	0.07809	12.8062	<.0001
Error	**rea**	theta1	0.33614	0.05210	6.4519	<.0001
	roa	theta2	0.41215	0.05122	8.0458	<.0001
	fea	theta3	0.31120	0.04593	6.7759	<.0001
	foa	theta4	0.40460	0.04618	8.7606	<.0001

Covariances Among Exogenous Variables						
Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
rpa	**riq**	cov01	0.18390	0.05614	3.2756	0.0011
rpa	**rses**	cov02	0.04890	0.05528	0.8846	0.3764
riq	**rses**	cov03	0.22200	0.05656	3.9250	<.0001
rpa	**fpa**	cov04	0.11470	0.05558	2.0638	0.0390
riq	**fpa**	cov05	0.10210	0.05550	1.8395	0.0658
rses	**fpa**	cov06	0.09310	0.05545	1.6789	0.0932
rpa	**fiq**	cov07	0.07820	0.05538	1.4120	0.1580
riq	**fiq**	cov08	0.33550	0.05824	5.7606	<.0001
rses	**fiq**	cov09	0.23020	0.05666	4.0628	<.0001
fpa	**fiq**	cov10	0.20870	0.05641	3.7000	0.0002
rpa	**fses**	cov11	0.01860	0.05523	0.3368	0.7363
riq	**fses**	cov12	0.18610	0.05616	3.3135	0.0009
rses	**fses**	cov13	0.27070	0.05720	4.7323	<.0001
fpa	**fses**	cov14	-0.04380	0.05527	-0.7925	0.4281
fiq	**fses**	cov15	0.29500	0.05757	5.1244	<.0001

These are the same results as displayed in Figure 18.42 for the simplified PATH model specification. The only differences are the arrangement of estimation results and the naming of the parameters.

Career Aspiration (Analysis 2)

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas.)

Jöreskog and Sörbom (1988) present more detailed results from a second analysis in which two constraints are imposed:

- The coefficients that connect the latent ambition variables are equal.
- The covariance of the disturbances of the ambition variables is zero.

Applying these constraints to Figure 18.40, you get the path diagram displayed in Figure 18.44.

Figure 18.44 Path Diagram for Career Aspiration : Analysis 2

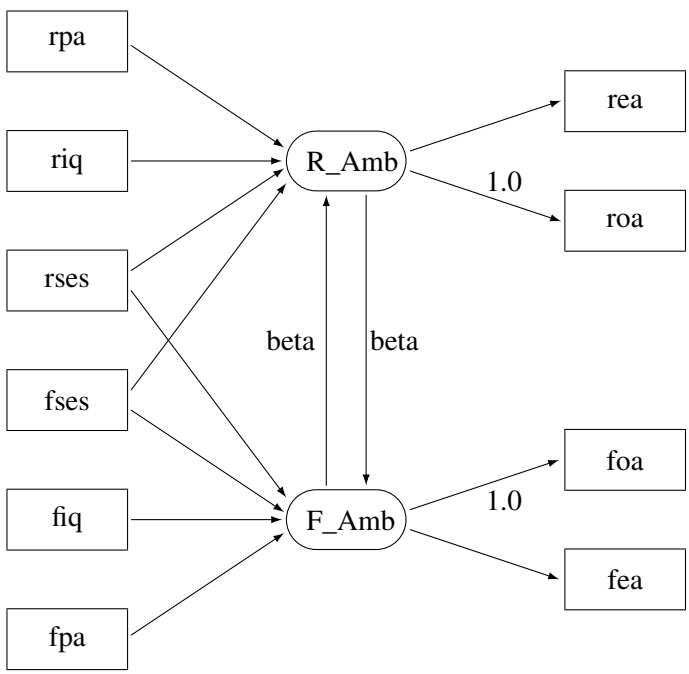

In Figure 18.44, the double-headed path that connected R_Amb and F_Amb no longer exists. Also, the single-headed paths between R_Amb and F_Amb are both labeled with beta, indicating the required constrained effects in the model. The path diagram in Figure 18.44 is transcribed into the PATH model in the following statements:

```
proc calis data=aspire nobs=329;
   path
      /* structural model of influences */
      rpa    ===> R_Amb         ,
      riq    ===> R_Amb         ,
      rses   ===> R_Amb         ,
      fses   ===> R_Amb         ,
      rses   ===> F_Amb         ,
      fses   ===> F_Amb         ,
      fiq    ===> F_Amb         ,
      fpa    ===> F_Amb         ,
      F_Amb  ===> R_Amb   = beta,
      R_Amb  ===> F_Amb   = beta,

      /* measurement model for aspiration */
      R_Amb  ===> rea         ,
      R_Amb  ===> roa     = 1.,
      F_Amb  ===> foa     = 1.,
      F_Amb  ===> fea     ;
run;
```

The only differences between the current specification and the preceding specification for Analysis 1 are the labeling of two paths with the same parameter beta and the deletion of PCOV statement where the covariance of R_Amb and F_Amb was specified in Analysis 1. The fit summary of the current model is displayed in Figure 18.45, and the estimation results are displayed in Figure 18.46.

Figure 18.45 Career Aspiration Data: Fit Summary for Analysis 2

Fit Summary	
Chi-Square	26.8987
Chi-Square DF	17
Pr > Chi-Square	0.0596
Standardized RMR (SRMR)	0.0203
Adjusted GFI (AGFI)	0.9492
RMSEA Estimate	0.0421
Akaike Information Criterion	102.8987
Schwarz Bayesian Criterion	247.1489
Bentler Comparative Fit Index	0.9880

The model fit chi-square value is 26.8987 (df=17, p=0.0596). The standardized RMSR and the RMSEA are both less than 0.05, while the adjusted GFI and comparative fit index are both bigger than 0.9. All these indicate a good model fit, but how does this model (Analysis 2) compare with that in Analysis 1?

The difference between the chi-square values for Analyses 1 and 2 is $26.8987 - 26.6972 = 0.2015$ with two degrees of freedom, which is far from significant. This indicates that the restricted model (Analysis 2) fits as well as the unrestricted model (Analysis 1). The AIC is 102.8987, and the SBC is 247.149. Both of these values are smaller than that of Analysis 1 (106.697 for AIC and 258.540 for SBC), and hence they indicate that the current model is a better one.

Figure 18.46 Career Aspiration Data: Estimation Results for Analysis 2

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
rpa	**===>**	**R_Amb**	_Parm01	0.16367	0.03872	4.2274	<.0001
riq	**===>**	**R_Amb**	_Parm02	0.25395	0.04186	6.0673	<.0001
rses	**===>**	**R_Amb**	_Parm03	0.22115	0.04187	5.2822	<.0001
fses	**===>**	**R_Amb**	_Parm04	0.07728	0.04149	1.8626	0.0625
rses	**===>**	**F_Amb**	_Parm05	0.06840	0.03868	1.7681	0.0770
fses	**===>**	**F_Amb**	_Parm06	0.21839	0.03948	5.5320	<.0001
fiq	**===>**	**F_Amb**	_Parm07	0.33063	0.04116	8.0331	<.0001
fpa	**===>**	**F_Amb**	_Parm08	0.15204	0.03636	4.1817	<.0001
F_Amb	**===>**	**R_Amb**	beta	0.18007	0.03912	4.6031	<.0001
R_Amb	**===>**	**F_Amb**	beta	0.18007	0.03912	4.6031	<.0001
R_Amb	**===>**	**rea**	_Parm09	1.06097	0.08921	11.8923	<.0001
R_Amb	**===>**	**roa**		1.00000			
F_Amb	**===>**	**foa**		1.00000			
F_Amb	**===>**	**fea**	_Parm10	1.07359	0.08063	13.3150	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	**riq**	_Add01	1.00000	0.07809	12.8062	<.0001
	rpa	_Add02	1.00000	0.07809	12.8062	<.0001
	rses	_Add03	1.00000	0.07809	12.8062	<.0001
	fiq	_Add04	1.00000	0.07809	12.8062	<.0001
	fpa	_Add05	1.00000	0.07809	12.8062	<.0001
	fses	_Add06	1.00000	0.07809	12.8062	<.0001
Error	**roa**	_Add07	0.41205	0.05103	8.0740	<.0001
	rea	_Add08	0.33764	0.05178	6.5204	<.0001
	foa	_Add09	0.40381	0.04608	8.7643	<.0001
	fea	_Add10	0.31337	0.04574	6.8517	<.0001
	R_Amb	_Add11	0.28113	0.04640	6.0587	<.0001
	F_Amb	_Add12	0.22924	0.03889	5.8939	<.0001

Figure 18.46 *continued*

Covariances Among Exogenous Variables						
Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
rpa	riq	_Add13	0.18390	0.05614	3.2756	0.0011
rses	riq	_Add14	0.22200	0.05656	3.9250	<.0001
rses	rpa	_Add15	0.04890	0.05528	0.8846	0.3764
fiq	riq	_Add16	0.33550	0.05824	5.7606	<.0001
fiq	rpa	_Add17	0.07820	0.05538	1.4120	0.1580
fiq	rses	_Add18	0.23020	0.05666	4.0628	<.0001
fpa	riq	_Add19	0.10210	0.05550	1.8395	0.0658
fpa	rpa	_Add20	0.11470	0.05558	2.0638	0.0390
fpa	rses	_Add21	0.09310	0.05545	1.6789	0.0932
fpa	fiq	_Add22	0.20870	0.05641	3.7000	0.0002
fses	riq	_Add23	0.18610	0.05616	3.3135	0.0009
fses	rpa	_Add24	0.01860	0.05523	0.3368	0.7363
fses	rses	_Add25	0.27070	0.05720	4.7323	<.0001
fses	fiq	_Add26	0.29500	0.05757	5.1244	<.0001
fses	fpa	_Add27	-0.04380	0.05527	-0.7925	0.4281

Like Analysis 1, the same two paths in the current analysis are not significant. That is, fses does not seem to be a good indicator of a respondent's ambition R_Amb, and rses does not seem to be a good indicator of a friend's ambition F_Amb. The t values are 1.862 and 1.768, respectively.

Career Aspiration (Analysis 3)

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas`.)

Loehlin (1987) points out that the models considered are unrealistic in at least two respects. First, the variables of parental aspiration, intelligence, and socioeconomic status are assumed to be measured without error. Loehlin adds uncorrelated measurement errors to the model and assumes, for illustrative purposes, that the reliabilities of these variables are known to be 0.7, 0.8, and 0.9, respectively. In practice, these reliabilities would need to be obtained from a separate study of the same or a very similar population. If these constraints are omitted, the model is not identified. However, constraining parameters to a constant in an analysis of a correlation matrix might make the chi-square goodness-of-fit test inaccurate, so there is more reason to be skeptical of the p-values. Second, the error terms for the respondent's aspiration are assumed to be uncorrelated with the corresponding terms for his friend. Loehlin introduces a correlation between the two educational aspiration error terms and between the two occupational aspiration error terms. These additions produce the path diagram for Loehlin's model shown in Figure 18.47.

Figure 18.47 Path Diagram for Career Aspiration: Analysis 3

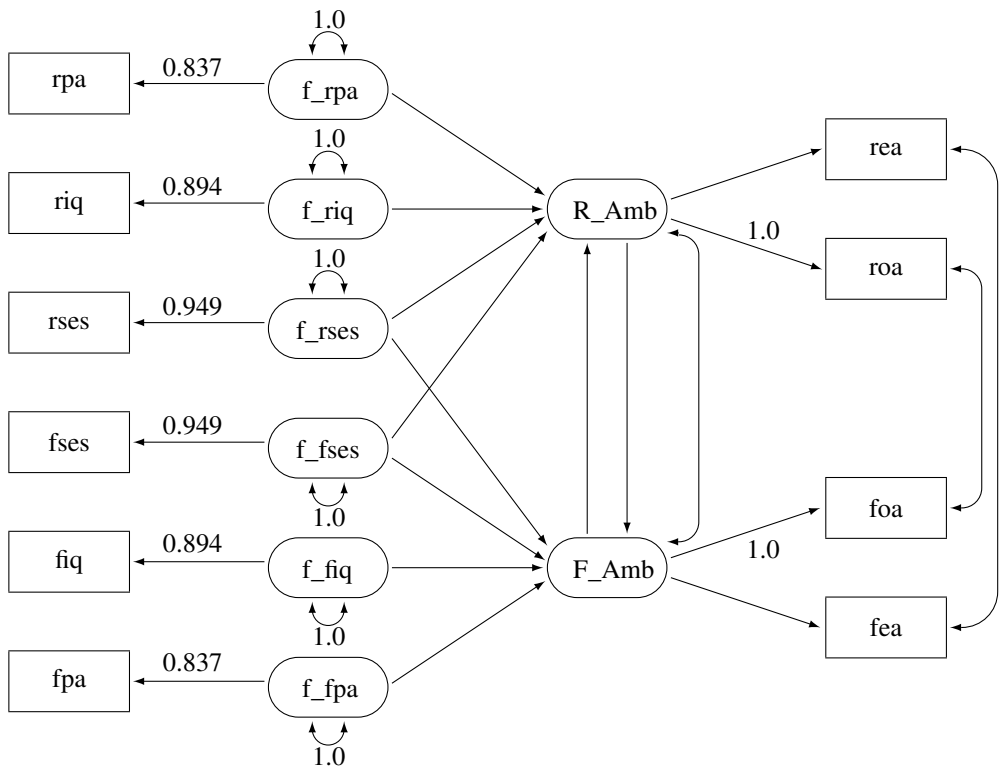

In Figure 18.47, the observed variables rpa, riq, rses, fses, fiq, and fpa are all measured with measurement errors. Their true scores counterparts f_rpa, f_riq, f_rses, f_fses, f_fiq, and f_fpa are latent variables in the model. Path coefficients from these latent variables to the observed variables are fixed coefficients, indicating the square roots of the theoretical reliabilities in the model. These latent variables, rather than the observed counterparts, serve as predictors of the ambition factors R_Amb and F_Amb in the current model (Analysis 3). The error terms for these two latent factors are correlated, as indicated by a double-headed path (arrow) that connects the two factors. Correlated errors for the occupational aspiration variables (roa and foa) and the educational aspiration variables (rea and fea) are also shown in Figure 18.47. Again, these correlated errors are represented by two double-headed paths (arrows) in the path diagram.

You use the following statements to specify the path model for Analysis 3:

```
proc calis data=aspire nobs=329;
   path
      /* measurement model for intelligence and environment */
      rpa     <===  f_rpa    = 0.837,
      riq     <===  f_riq    = 0.894,
      rses    <===  f_rses   = 0.949,
      fses    <===  f_fses   = 0.949,
      fiq     <===  f_fiq    = 0.894,
      fpa     <===  f_fpa    = 0.837,

      /* structural model of influences */
      f_rpa   ===>  R_Amb,
```

```
      f_riq   ===>  R_Amb,
      f_rses  ===>  R_Amb,
      f_fses  ===>  R_Amb,
      f_rses  ===>  F_Amb,
      f_fses  ===>  F_Amb,
      f_fiq   ===>  F_Amb,
      f_fpa   ===>  F_Amb,
      F_Amb   ===>  R_Amb,
      R_Amb   ===>  F_Amb,

      /* measurement model for aspiration */
      R_Amb   ===>  rea        ,
      R_Amb   ===>  roa    = 1.,
      F_Amb   ===>  foa    = 1.,
      F_Amb   ===>  fea    ;
   pvar
      f_rpa f_riq f_rses f_fses f_fiq f_fpa = 6 * 1.0;
   pcov
      R_Amb F_Amb  ,
      rea  fea     ,
      roa  foa     ;
run;
```

In this specification, the measurement model for the six intelligence and environment variables are added. They are the first six paths in the PATH statement. Fixed constants are set for these path coefficients so as to make the measurement model identified and to set the required reliabilities of these measurement indicators. The structural model of influences and the measurement model for aspiration are the same as specified in Analysis 1. (See the section "Career Aspiration (Analysis 1)" on page 350.) All the correlated errors are specified in the PCOV statement.

The fit summary of the current model is displayed in Figure 18.48.

Figure 18.48 Career Aspiration Data: Fit Summary for Analysis 3

Fit Summary	
Chi-Square	12.0132
Chi-Square DF	13
Pr > Chi-Square	0.5266
Standardized RMR (SRMR)	0.0149
Adjusted GFI (AGFI)	0.9692
RMSEA Estimate	0.0000
Akaike Information Criterion	96.0132
Schwarz Bayesian Criterion	255.4476
Bentler Comparative Fit Index	1.0000

Since the p-value for the chi-square test is 0.5266, this model clearly cannot be rejected. Both the standardized RMSR and the RMSEA are very small, and both the adjusted GFI and the comparative fit index are high. All these point to an excellent model fit. However, Schwarz's Bayesian criterion for this model (SBC = 255.4476) is somewhat larger than for Jöreskog and Sörbom (1988) Analysis 2 in Figure 18.45 (SBC = 247.1489), suggesting that a more parsimonious model would be desirable.

The estimation results are displayed in Figure 18.49.

Figure 18.49 Career Aspiration Data: Estimation Results for Analysis 3

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
rpa	<===	f_rpa		0.83700			
riq	<===	f_riq		0.89400			
rses	<===	f_rses		0.94900			
fses	<===	f_fses		0.94900			
fiq	<===	f_fiq		0.89400			
fpa	<===	f_fpa		0.83700			
f_rpa	===>	R_Amb	_Parm01	0.18370	0.05044	3.6420	0.0003
f_riq	===>	R_Amb	_Parm02	0.28004	0.06139	4.5618	<.0001
f_rses	===>	R_Amb	_Parm03	0.22616	0.05223	4.3300	<.0001
f_fses	===>	R_Amb	_Parm04	0.08698	0.05476	1.5883	0.1122
f_rses	===>	F_Amb	_Parm05	0.06327	0.05219	1.2124	0.2254
f_fses	===>	F_Amb	_Parm06	0.21539	0.05121	4.2060	<.0001
f_fiq	===>	F_Amb	_Parm07	0.35387	0.06741	5.2497	<.0001
f_fpa	===>	F_Amb	_Parm08	0.16876	0.04934	3.4205	0.0006
F_Amb	===>	R_Amb	_Parm09	0.11898	0.11396	1.0441	0.2964
R_Amb	===>	F_Amb	_Parm10	0.13022	0.12067	1.0791	0.2805
R_Amb	===>	rea	_Parm11	1.08399	0.09417	11.5105	<.0001
R_Amb	===>	roa		1.00000			
F_Amb	===>	foa		1.00000			
F_Amb	===>	fea	_Parm12	1.11630	0.08627	12.9394	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	f_rpa		1.00000			
	f_riq		1.00000			
	f_rses		1.00000			
	f_fses		1.00000			
	f_fiq		1.00000			
	f_fpa		1.00000			
Error	riq	_Add01	0.20874	0.07832	2.6652	0.0077
	rpa	_Add02	0.29584	0.07774	3.8057	0.0001
	rses	_Add03	0.09887	0.07803	1.2671	0.2051
	roa	_Add04	0.42307	0.05243	8.0695	<.0001
	rea	_Add05	0.32707	0.05452	5.9988	<.0001
	fiq	_Add06	0.19989	0.07674	2.6048	0.0092
	fpa	_Add07	0.29988	0.07807	3.8409	0.0001
	fses	_Add08	0.10324	0.07824	1.3195	0.1870
	foa	_Add09	0.42240	0.04730	8.9310	<.0001
	fea	_Add10	0.28716	0.04804	5.9776	<.0001
	R_Amb	_Add11	0.25418	0.04469	5.6874	<.0001
	F_Amb	_Add12	0.19698	0.03814	5.1653	<.0001

Figure 18.49 *continued*

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
f_riq	f_rpa	_Add13	0.24677	0.07519	3.2820	0.0010
f_rses	f_rpa	_Add14	0.06183	0.06945	0.8903	0.3733
f_rses	f_riq	_Add15	0.26351	0.06687	3.9408	<.0001
f_fses	f_rpa	_Add16	0.02382	0.06952	0.3427	0.7318
f_fses	f_riq	_Add17	0.22136	0.06648	3.3298	0.0009
f_fses	f_rses	_Add18	0.30156	0.06359	4.7421	<.0001
f_fiq	f_rpa	_Add19	0.10853	0.07362	1.4742	0.1404
f_fiq	f_riq	_Add20	0.42476	0.07219	5.8837	<.0001
f_fiq	f_rses	_Add21	0.27250	0.06660	4.0914	<.0001
f_fiq	f_fses	_Add22	0.34922	0.06771	5.1576	<.0001
f_fpa	f_rpa	_Add23	0.15789	0.07873	2.0056	0.0449
f_fpa	f_riq	_Add24	0.13084	0.07418	1.7639	0.0778
f_fpa	f_rses	_Add25	0.11516	0.06978	1.6505	0.0988
f_fpa	f_fses	_Add26	-0.05622	0.06971	-0.8065	0.4200
f_fpa	f_fiq	_Add27	0.27867	0.07530	3.7008	0.0002

Covariances Among Errors

Error of	Error of	Parameter	Estimate	Standard Error	t Value	Pr > |t|
R_Amb	F_Amb	_Parm13	-0.00936	0.05010	-0.1867	0.8519
rea	fea	_Parm14	0.02308	0.03139	0.7355	0.4621
roa	foa	_Parm15	0.11206	0.03258	3.4399	0.0006

Like Analyses 1 and 2, two paths that concern the validity of the indicators in the current analysis do not show significance. That is, f_fses does not seem to be a good indicator of a respondent's ambition R_Amb, and f_rses does not seem to be a good indicator of a friend's ambition F_Amb. The *t* values are 1.588 and 1.212, respectively. In addition, in the current model (Analysis 3), the structural relationships between the ambition factors do not show significance. The *t* value for the path from the friend's ambition factor F_Amb on the respondent's ambition factor R_Amb is only 1.044, while the *t* value for the path from the respondent's ambition factor R_Amb on the friend's ambition factor F_Amb is only 1.079. These cast doubts on the validity of the structural model and perhaps even the entire model.

Fitting LISREL Models by the LISMOD Modeling Language

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas.)

The model described in the section "Career Aspiration (Analysis 3)" on page 361 provides a good example of the LISREL model. In PROC CALIS, the LISREL model specifications are supported by a matrix-based language called LISMOD (LISREL model). In this section, the path diagram in Figure 18.47 is specified by the LISMOD modeling language of PROC CALIS. See the section "Career Aspiration (Analysis 3)" on page 361 for detailed descriptions of the model.

In order to understand the LISMOD modeling language of PROC CALIS, some basic understanding of the LISREL model is necessary. In a LISREL model, variables are classified into four distinct classes:

- ξ is a vector of exogenous (independent) latent variables in the model. They are specified in XI= variable list in the LISMOD statement.
- η is a vector of endogenous (dependent) latent variables in the model. They are specified in ETA= variable list in the LISMOD statement.
- x is a vector of observed indicator variables for ξ in the model. They are specified in XVAR= variable list in the LISMOD statement.
- y is a vector of observed indicator variables for η in the model. They are specified in YVAR= variable list in the LISMOD statement.

For detailed descriptions of the LISMOD modeling language, see the LISMOD statement and the section "The LISMOD Model and Submodels" on page 1752 in Chapter 33, "The CALIS Procedure." To successfully set up a LISMOD model in PROC CALIS, you first need to recognize these classes of variables in your model. For the path diagram in Figure 18.47, it is not difficult to see the following:

- ξ is the vector of the intelligence and environmental factors: f_rpa, f_riq, f_rses, f_fses, f_fiq, and f_fpa. These variables are exogenous because no single-headed arrows point to them.
- η is the vector of the ambition factors: R_Amb, and F_Amb. They are endogenous because each of them has at least one single-headed arrow pointing to it.
- x is the vector of the observed indicator variables for the intelligence and environmental factors ξ. These indicators are rpa, riq, rses, fses, fiq, and fpa.
- y is the vector of observed indicator variables for the ambition factors η. These indicators are rea, roa, foa, and fea.

In LISMOD, you do not need to define error terms explicitly as latent variables. The parameters in LISMOD are defined as entries in various model matrices. The following statements specify the LISMOD model for the diagram in Figure 18.47:

```
proc calis data=aspire nobs=329;
   lismod
      xi   = f_rpa f_riq f_rses f_fses f_fiq f_fpa,
      eta  = R_Amb F_Amb,
      xvar = rpa riq rses fses fiq fpa,
      yvar = rea roa foa fea;

   /* measurement model for aspiration */
   matrix _lambday_ [1,1], [2,1] = 1.0, [3,2] = 1.0, [4,2];
   matrix _thetay_  [4,1], [3,2];

   /* measurement model for intelligence and environment */
   matrix _lambdax_ [1,1] = 0.837 0.894 0.949 0.949 0.894 0.837;

   /* structural model of influences */
   matrix _beta_ [2,1],[1,2];
   matrix _gamma_ [1,1 to 4], [2,3 to 6];

   /* Covariances among Eta-variables */
   matrix _psi_ [2,1];

   /* Fixed variances for Xi-variables */
   matrix _phi_ [1,1] = 6 * 1.0;
run;
```

The LISMOD statement invokes the LISMOD modeling language of PROC CALIS. In the LISMOD statement, you list the four classes of variables in the model in the XI=, ETA=, XVAR=, and YVAR= variable lists, respectively. After you define the four classes of variables, you use several MATRIX statements to specify the model matrices and the parameters in the model.

Basically, there are three model components in the LISMOD specification: two measurement models and one structural model. The first measurement model specifies the functional relationships between observed variables y (YVAR= variables) and the endogenous (dependent) latent factors η (ETA= variables). The second measurement model specifies the functional relationships between observed variables x and (XVAR= variables) and the exogenous (independent) latent factors ξ (XI= variables). The structural model specifies the relationships between the endogenous and exogenous latent variables η and ξ. To facilitate the discussion of these model components and the corresponding LISMOD model specification, some initial model output from PROC CALIS are shown.

The Measurement Model for y

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas.)

The first component of the LISMOD specification is the measurement model for y, as shown in the following equation:

$$y = \mathbf{\Lambda_y}\eta + \epsilon$$

In the context of covariance structure analysis, without loss of generality, it is assumed that y and η are centered so that there is no intercept term in the equation. This equation essentially states that y is a function of the true scores vector η plus the error term ϵ, which is independent of η. The model matrices involved in this measurement model are $\mathbf{\Lambda_y}$ (effects of η on y) and $\mathbf{\Theta_y}$, which is the covariance matrix of ϵ.

For the career aspiration data, you specify the following two MATRIX statements for this measurement model:

```
matrix _lambday_ [1,1], [2,1] = 1.0, [3,2] = 1.0, [4,2];
matrix _thetay_  [4,1], [3,2];
```

The first matrix statement is for matrix $\mathbf{\Lambda_y}$. You specify four parameters in this matrix. The [1,1] and [4,2] elements are free parameters, and the [2,1] and [3,2] elements have fixed values of 1. You do not specify other elements in this matrix. By default, unspecified elements in the $\mathbf{\Lambda_y}$ matrix are fixed zeros. You can check your initial model specification of this matrix, as shown in the Figure 18.50.

Figure 18.50 Career Aspiration Analysis 3: Initial Measurement Model for y

Initial _LAMBDAY_ Matrix		
	R_Amb	F_Amb
rea	. [_Parm01]	0
roa	1.0000	0
foa	0	1.0000
fea	0	. [_Parm02]

Initial _THETAY_ Matrix				
	rea	roa	foa	fea
rea	. [_Add07]	0	0	. [_Parm13]
roa	0	. [_Add08]	. [_Parm14]	0
foa	0	. [_Parm14]	. [_Add09]	0
fea	. [_Parm13]	0	0	. [_Add10]

NOTE: Parameters with prefix '_Add' are added by PROC CALIS.

In Figure 18.50, the initial _LAMBDAY_ matrix is a 4×2 matrix. The _LAMBDAY_ matrix contains

information about the relationships between the row indicator variables y (YVAR= variables) and the column factors η (ETA= variables). As specified in the MATRIX statement for _LAMBDAY_, the [1,1] and [4,2] are free parameters named _Parm01 and _Parm02, respectively. These parameter names are generated by PROC CALIS. Fixed values 1.0 appear in the [2,1] and [3,2] elements. These fixed values are used to identify the scales of the latent variables R_Amb and F_Amb.

The _THETAY_ matrix in Figure 18.50 is the covariance matrix among the error terms for the y-variables (YVAR= variables). This is a 4×4 matrix for the four measured indicators. As specified in the MATRIX statement for _THETAY_, the [4,1] and [3,2] elements are free parameters named _Parm13 and _Parm14, respectively. Because _THETAY_ is a symmetric matrix, elements [1,4] and [2,3] are also implicitly specified as parameters in this model matrix.

As shown in Figure 18.50, PROC CALIS adds four default free parameters to the _THETAY_ matrix. On the diagonal of the _THETAY_ matrix, parameters _Add07, _Add08, _Add09, and _Add10 are added as default free parameters by PROC CALIS automatically. In general, error variances are default free parameters in PROC CALIS. You do not have to specify them but you can specify them if you want to, especially when you need to set fixed values or other constraints on them.

The Measurement Model for x

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas.)

The second component of the LISMOD specification is the measurement model for x, as shown in the following equation:

$$x = \mathbf{\Lambda_x}\xi + \delta$$

The measurement model for x is similar to that for y. Assuming that x and ξ are centered, this equation states that x is a function of the true scores vector ξ plus the error term δ, which is independent of ξ. The model matrices involved in this measurement model are $\mathbf{\Lambda_x}$ (effects of ξ on x) and $\mathbf{\Theta_x}$, which is the covariance matrix of δ.

For the career aspiration data, you specify the following MATRIX statement for this measurement model:

```
matrix _lambdax_ [1,1] = 0.837 0.894 0.949 0.949 0.894 0.837;
```

Figure 18.51 shows the output related to the specification of the measurement model for x.

Figure 18.51 Career Aspiration Analysis 3: Initial Measurement Model for x

Initial _LAMBDAX_ Matrix						
	f_rpa	f_riq	f_rses	f_fses	f_fiq	f_fpa
rpa	0.8370	0	0	0	0	0
riq	0	0.8940	0	0	0	0
rses	0	0	0.9490	0	0	0
fses	0	0	0	0.9490	0	0
fiq	0	0	0	0	0.8940	0
fpa	0	0	0	0	0	0.8370

Figure 18.51 *continued*

Initial _THETAX_ Matrix						
	rpa	riq	rses	fses	fiq	fpa
rpa	. [_Add01]	0	0	0	0	0
riq	0	. [_Add02]	0	0	0	0
rses	0	0	. [_Add03]	0	0	0
fses	0	0	0	. [_Add04]	0	0
fiq	0	0	0	0	. [_Add05]	0
fpa	0	0	0	0	0	. [_Add06]

NOTE: Parameters with prefix '_Add' are added by PROC CALIS.

In Figure 18.51, the initial _LAMBDAX_ matrix is a 6×6 matrix. The _LAMBDAX_ matrix contains information about the relationships between the row indicator variables x (XVAR= variables) and the column factors ξ (XI= variables). As specified in the MATRIX statement for _LAMBDAX_, the diagonal elements are filled with the fixed values provided. The [1,1] specification in the MATRIX statement for _LAMBDAX_ provides the starting element for the subsequent parameter list to fill in. In this case, the list contains six fixed values, and PROC CALIS proceeds from [1,1] to [2,2], [3,3] and so on until the entire list of parameters is consumed. This kind of notation is a shortcut of the following equivalent specification:

```
matrix _lambdax_ [1,1]=0.837, [2,2]=0.894, [3,3]=0.949,
                 [4,4]=0.949, [5,5]=0.894, [6,6]=0.837;
```

PROC CALIS provides many different kinds of shortcuts in specifying matrix elements. See the MATRIX statement of Chapter 33, "The CALIS Procedure," for details.

At the bottom of Figure 18.51, the initial _THETAX_ matrix is shown. Even though you did not specify any elements of this matrix in any MATRIX statements, the diagonal elements of this matrix are set as default parameters by PROC CALIS. Default parameters added by PROC CALIS are all denoted by names with the prefix '_Add'.

The Structural Model

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas.)

The last component of the LISMOD specification is the structural model that describes the relationship between η and ξ, as shown in the following equation:

$$\eta = \boldsymbol{\beta}\eta + \boldsymbol{\Gamma}\xi + \zeta$$

In this equation, η is endogenous (dependent) and ξ is exogenous (independent). Variables in η can have effects among themselves. Their effects are specified in the $\boldsymbol{\beta}$ matrix. The effects of ξ on η are specified in the $\boldsymbol{\Gamma}$ matrix. Finally, the error term for the structural relationships is denoted by ζ, which is independent of ξ.

There are four model matrices assumed in the structural model. $\boldsymbol{\beta}$ and $\boldsymbol{\Gamma}$ are matrices for the effects of variables. In addition, matrix $\boldsymbol{\Psi}$ denotes the covariance matrix for the error term ζ, and matrix $\boldsymbol{\Phi}$ denotes the covariance matrix of ξ.

For the career aspiration data, you use the following MATRIX statements for the structural model:

```
matrix _beta_ [2,1],[1,2];
matrix _gamma_ [1,1 to 4], [2,3 to 6];
matrix _psi_ [2,1];
matrix _phi_ [1,1] = 6 * 1.0;
```

In Figure 18.52, initial _BETA_ and _GAMMA_ matrices are shown.

Figure 18.52 Career Aspiration Analysis 3: Initial Structural Equations

Initial _BETA_ Matrix		
	R_Amb	F_Amb
R_Amb	0	. [_Parm12]
F_Amb	. [_Parm11]	0

Initial _GAMMA_ Matrix						
	f_rpa	f_riq	f_rses	f_fses	f_fiq	f_fpa
R_Amb	. [_Parm03]	. [_Parm04]	. [_Parm05]	. [_Parm06]	0	0
F_Amb	0	0	. [_Parm07]	. [_Parm08]	. [_Parm09]	. [_Parm10]

In Figure 18.52, the _BETA_ matrix contains information about the relationships among the η-variables (ETA= variables). Both the row and column variables of the _BETA_ matrix refer to the list of η-variables. The row variables receive effects from the column variables. You specify two parameters in the _BETA_ matrix: element [2,1] is the effect of R_Amb on F_Amb, and element [2,1] is the effect of F_Amb on R_Amb. Other effects are fixed zeros in this matrix.

The _GAMMA_ matrix contains information about the relationships between the η-variables (ETA= variables) and the ξ-variables (XI= variables). The row variables are the η-variables, which receive effects from the column ξ-variables. You specify eight free parameters in this matrix. These eight parameters represent the eight path coefficients from ξ (the intelligence and environment factors) to the η variables (the ambition factors), as shown in the path diagram in Figure 18.47. A shortcut in the MATRIX statement syntax for the _GAMMA_ matrix has been used. That is, `[1, 1 to 4]` means the [1,1], [1,2], [1,3], and [1,4] elements, and `[2, 3 to 6]` means the [2,3], [2,4], [2,5], and [2,6] elements. All these elements are free parameters in the model and free parameter names are generated for these elements.

Figure 18.53 shows the initial _PSI_ and _PHI_ matrices.

Figure 18.53 Career Aspiration Analysis 3: Initial Variances and Covariances

Initial _PSI_ Matrix	R_Amb	F_Amb
R_Amb	. [_Add11]	. [_Parm15]
F_Amb	. [_Parm15]	. [_Add12]

NOTE: Parameters with prefix '_Add' are added by PROC CALIS.

Initial _PHI_ Matrix	f_rpa	f_riq	f_rses	f_fses	f_fiq	f_fpa
f_rpa	1.0000	. [_Add13]	. [_Add14]	. [_Add16]	. [_Add19]	. [_Add23]
f_riq	. [_Add13]	1.0000	. [_Add15]	. [_Add17]	. [_Add20]	. [_Add24]
f_rses	. [_Add14]	. [_Add15]	1.0000	. [_Add18]	. [_Add21]	. [_Add25]
f_fses	. [_Add16]	. [_Add17]	. [_Add18]	1.0000	. [_Add22]	. [_Add26]
f_fiq	. [_Add19]	. [_Add20]	. [_Add21]	. [_Add22]	1.0000	. [_Add27]
f_fpa	. [_Add23]	. [_Add24]	. [_Add25]	. [_Add26]	. [_Add27]	1.0000

NOTE: Parameters with prefix '_Add' are added by PROC CALIS.

The _PSI_ matrix contains information about the covariances of error terms for the η-variables, which are endogenous in the structural model. There are two η-variables in the model—the two ambition factors R_Amb and F_Amb. You specify the [2,1] element as a free parameter in the MATRIX statement for _PSI_. This means that the error covariance between R_Amb and F_Amb is a free parameter to estimate in the model. In Figure 18.53, both [2,1] and [1,2] elements are named as _Parm15 because _PSI_ is a symmetric matrix. Again, the diagonal elements of this covariance matrix, which are for the error variances of the ambition factors, are default free parameters in PROC CALIS. These parameters are named with the prefix _Add.

Finally, the _PHI_ matrix contains information about the covariances among the exogenous latent factors in the structural model. For the _PHI_ matrix, you fix all the diagonal elements to 1 in the MATRIX statement for _PHI_. This makes the latent variable scales identified. These fixed values are echoed in the output of the initial _PHI_ matrix shown in Figure 18.53. In addition, all covariances among latent exogenous variables are set to be free parameters by default.

Fit Summary of the LISMOD Model for Career Aspiration Analysis 3

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icalex4.sas`.)

Figure 18.54 shows the fit summary of the LISMOD model. All these fit index values match those from using the PATH model specification of the same model, as shown in Figure 18.48. Therefore, you are confident that the current LISMOD model specification is equivalent to the PATH model specification shown in the section "Career Aspiration (Analysis 3)" on page 361.

Figure 18.54 Career Aspiration Analysis 3: Fit Summary of the LISMOD Model

Fit Summary	
Chi-Square	12.0132
Chi-Square DF	13
Pr > Chi-Square	0.5266
Standardized RMR (SRMR)	0.0149
Adjusted GFI (AGFI)	0.9692
RMSEA Estimate	0.0000
Akaike Information Criterion	96.0132
Schwarz Bayesian Criterion	255.4476
Bentler Comparative Fit Index	1.0000

Estimation results are shown in Figure 18.55, Figure 18.56, and Figure 18.57, respectively for the measurement model for y, measurement model for x, and the structural model. These are the same estimation results as those from the equivalent PATH model specification in Figure 18.49. However, estimates in the LISMOD model are now arranged in the matrix form, with standard error estimates and t values shown.

Figure 18.55 Career Aspiration Analysis 3: Estimation of Measurement Model for y

LAMBDAY Matrix: Estimate/StdErr/t-value/p-value		
	R_Amb	**F_Amb**
rea	1.0840 0.0942 11.5105 <.0001 [_Parm01]	0
roa	1.0000	0
foa	0	1.0000
fea	0	1.1163 0.0863 12.9394 <.0001 [_Parm02]

Figure 18.55 *continued*

THETAY Matrix: Estimate/StdErr/t-value/p-value	rea	roa	foa	fea
rea	0.3271 0.0545 5.9988 <.0001 [_Add07]	0	0	0.0231 0.0314 0.7355 0.4621 [_Parm13]
roa	0	0.4231 0.0524 8.0695 <.0001 [_Add08]	0.1121 0.0326 3.4399 0.000582 [_Parm14]	0
foa	0	0.1121 0.0326 3.4399 0.000582 [_Parm14]	0.4224 0.0473 8.9310 <.0001 [_Add09]	0
fea	0.0231 0.0314 0.7355 0.4621 [_Parm13]	0	0	0.2872 0.0480 5.9776 <.0001 [_Add10]

Figure 18.56 Career Aspiration Analysis 3: Estimation of Measurement Model for x

LAMBDAX Matrix: Estimate/StdErr/t-value/p-value	f_rpa	f_riq	f_rses	f_fses	f_fiq	f_fpa
rpa	0.8370	0	0	0	0	0
riq	0	0.8940	0	0	0	0
rses	0	0	0.9490	0	0	0
fses	0	0	0	0.9490	0	0
fiq	0	0	0	0	0.8940	0
fpa	0	0	0	0	0	0.8370

Figure 18.56 *continued*

THETAX Matrix: Estimate/StdErr/t-value/p-value						
	rpa	**riq**	**rses**	**fses**	**fiq**	**fpa**
rpa	0.2958 0.0777 3.8057 0.000141 [_Add01]	0	0	0	0	0
riq	0	0.2087 0.0783 2.6652 0.007695 [_Add02]	0	0	0	0
rses	0	0	0.0989 0.0780 1.2671 0.2051 [_Add03]	0	0	0
fses	0	0	0	0.1032 0.0782 1.3195 0.1870 [_Add04]	0	0
fiq	0	0	0	0	0.1999 0.0767 2.6048 0.009192 [_Add05]	0
fpa	0	0	0	0	0	0.2999 0.0781 3.8409 0.000123 [_Add06]

Figure 18.57 Career Aspiration Analysis 3: Estimation of Structural Model

BETA Matrix: Estimate/StdErr/t-value/p-value		
	R_Amb	**F_Amb**
R_Amb	0	0.1190 0.1140 1.0441 0.2964 [_Parm12]
F_Amb	0.1302 0.1207 1.0791 0.2805 [_Parm11]	0

Figure 18.57 *continued*

GAMMA Matrix: Estimate/StdErr/t-value/p-value						
	f_rpa	f_riq	f_rses	f_fses	f_fiq	f_fpa
R_Amb	0.1837	0.2800	0.2262	0.0870	0	0
	0.0504	0.0614	0.0522	0.0548		
	3.6420	4.5618	4.3300	1.5883		
	0.000271	<.0001	<.0001	0.1122		
	[_Parm03]	[_Parm04]	[_Parm05]	[_Parm06]		
F_Amb	0	0	0.0633	0.2154	0.3539	0.1688
			0.0522	0.0512	0.0674	0.0493
			1.2124	4.2060	5.2497	3.4205
			0.2254	<.0001	<.0001	0.000625
			[_Parm07]	[_Parm08]	[_Parm09]	[_Parm10]

PSI Matrix: Estimate/StdErr/t-value/p-value		
	R_Amb	F_Amb
R_Amb	0.2542	-0.009355
	0.0447	0.0501
	5.6874	-0.1867
	<.0001	0.8519
	[_Add11]	[_Parm15]
F_Amb	-0.009355	0.1970
	0.0501	0.0381
	-0.1867	5.1653
	0.8519	<.0001
	[_Parm15]	[_Add12]

PHI Matrix: Estimate/StdErr/t-value/p-value						
	f_rpa	f_riq	f_rses	f_fses	f_fiq	f_fpa
f_rpa	1.0000	0.2468	0.0618	0.0238	0.1085	0.1579
		0.0752	0.0695	0.0695	0.0736	0.0787
		3.2820	0.8903	0.3427	1.4742	2.0056
		0.001031	0.3733	0.7318	0.1404	0.0449
		[_Add13]	[_Add14]	[_Add16]	[_Add19]	[_Add23]
f_riq	0.2468	1.0000	0.2635	0.2214	0.4248	0.1308
	0.0752		0.0669	0.0665	0.0722	0.0742
	3.2820		3.9408	3.3298	5.8837	1.7639
	0.001031		<.0001	0.000869	<.0001	0.0778
	[_Add13]		[_Add15]	[_Add17]	[_Add20]	[_Add24]
f_rses	0.0618	0.2635	1.0000	0.3016	0.2725	0.1152
	0.0695	0.0669		0.0636	0.0666	0.0698
	0.8903	3.9408		4.7421	4.0914	1.6505
	0.3733	<.0001		<.0001	<.0001	0.0988
	[_Add14]	[_Add15]		[_Add18]	[_Add21]	[_Add25]
f_fses	0.0238	0.2214	0.3016	1.0000	0.3492	-0.0562
	0.0695	0.0665	0.0636		0.0677	0.0697
	0.3427	3.3298	4.7421		5.1576	-0.8065
	0.7318	0.000869	<.0001		<.0001	0.4200
	[_Add16]	[_Add17]	[_Add18]		[_Add22]	[_Add26]
f_fiq	0.1085	0.4248	0.2725	0.3492	1.0000	0.2787
	0.0736	0.0722	0.0666	0.0677		0.0753
	1.4742	5.8837	4.0914	5.1576		3.7008
	0.1404	<.0001	<.0001	<.0001		0.000215
	[_Add19]	[_Add20]	[_Add21]	[_Add22]		[_Add27]
f_fpa	0.1579	0.1308	0.1152	-0.0562	0.2787	1.0000
	0.0787	0.0742	0.0698	0.0697	0.0753	
	2.0056	1.7639	1.6505	-0.8065	3.7008	
	0.0449	0.0778	0.0988	0.4200	0.000215	
	[_Add23]	[_Add24]	[_Add25]	[_Add26]	[_Add27]	

Some Important PROC CALIS Features

In this section, some of the main features of PROC CALIS are introduced. Emphasis is placed on showing how these features are useful in practical structural equation modeling.

Modeling Languages for Specifying Models

PROC CALIS provides several modeling languages to specify a model. Different modeling languages in PROC CALIS are signified by the main model specification statement used. In this chapter, you have seen examples of the FACTOR, LINEQS, LISMOD, MSTRUCT, PATH, and RAM modeling languages. Depending on your modeling philosophy and the type of the model, you can choose a modeling language that is most suitable for your application. For example, models specified using structural equations can be transcribed directly into the LINEQS statement. Models that are hypothesized using path diagrams can be described easily in the PATH or RAM statement. First-order confirmatory or exploratory factor models are most conveniently specified by using the FACTOR and MATRIX statements. Traditional LISREL models are supported through the LISMOD and MATRIX statements. Finally, patterned covariance and mean models can be specified directly by the MSTRUCT and MATRIX statements, or by the COVPATTERN= and MEANPATTERN= options.

For most applications in structural equation modeling, the PATH and LINEQS statements are the easiest to use. For testing the built-in covariance and mean patterns of PROC CALIS, the use of the COVPATTERN= and the MEANPATTERN= options are the most efficient. In other cases, the FACTOR, LISMOD, MSTRUCT, or RAM statement might be more suitable. For very general matrix model specifications, you can use the COSAN modeling language. See the COSAN statement and the section "The COSAN Model" on page 1734 in Chapter 33, "The CALIS Procedure," for details about the COSAN modeling language. See also the section "Which Modeling Language?" on page 1530 in Chapter 33, "The CALIS Procedure," for a more detailed discussion about the use of different modeling languages.

Estimation Methods

The CALIS procedure provides six methods of estimation specified by the METHOD= option:

DWLS	diagonally weighted least squares
FIML	full-information maximum likelihood
GLS	normal theory generalized least squares
ML	maximum likelihood for multivariate normal distributions
MLSB	maximum likelihood with Satorra-Bentler adjustments
ULS	unweighted least squares
WLS	weighted least squares for arbitrary distributions

Each estimation method is based on finding parameter estimates that minimize a discrepancy (badness-of-fit) function, which measures the difference between the observed sample covariance matrix and the fitted (predicted) covariance matrix, given the model and the parameter estimates. The difference between the observed sample mean vector and the fitted (predicted) mean vector is also taken into account when the mean

structures are modeled. See the section "Estimation Criteria" on page 1805 in Chapter 33, "The CALIS Procedure," for formulas, or see Loehlin (1987, pp. 54–62) and Bollen (1989, pp. 104–123) for further discussion.

The default estimation is METHOD=ML, which is the most popular method for applications. The option METHOD=GLS usually produces very similar results to those produced by METHOD=ML. If your data contain random missing values and it is important to use the information from those incomplete observations, you might want to use the FIML method, which provides a sound treatment of missing values in data. METHOD=ML and METHOD=FIML are essentially the same method when you do not have missing values (see Example 33.16 of Chapter 33, "The CALIS Procedure"). Asymptotically, ML and GLS are the same. Both methods assume a multivariate normal distribution in the population. The WLS method with the default weight matrix is equivalent to the asymptotically distribution-free (ADF) method, which yields asymptotically normal estimates regardless of the distribution in the population. When the multivariate normal assumption is in doubt, especially if the variables have high kurtosis, you should seriously consider the WLS method. When a correlation matrix is analyzed, only WLS can produce correct standard error estimates. However, in order to use the WLS method with the expected statistical properties, the sample size must be large. Several thousand observations might be a minimum requirement.

If you do not have a large enough sample to use the ADF method, you can consider two viable ML alternatives to deal with nonnormal distributions. Both require raw data input. The first alternative is to use a robust ML estimation by specifying the ROBUST option. This method downweights outlying observations that depart from the multivariate normal distribution during estimation. Hence, the estimation is robust to outliers. The second alternative is to use ML estimation with the Satorra-Bentler adjustments (METHOD=MLSB) for nonnormality (Satorra and Bentler 1994). This method takes the nonnormality into account in computing chi-square model fit statistics and standard errors. As a result, you can still have accurate statistical inferences about model fit and the parameter estimates even if data are nonnormal.

The ULS and DWLS methods yield reasonable estimates under less restrictive assumptions. You can apply these methods to normal or nonnormal situations or to covariance or correlation matrices. The drawback is that the statistical qualities of the estimates seem to be unknown. For this reason, PROC CALIS does not provide standard errors or test statistics with these two methods.

You cannot use METHOD=ML, MLSB, or GLS if the observed covariance matrix is singular. You can either remove variables that are involved in the linear dependencies or use less restrictive estimation methods such as ULS. Specifying METHOD=ML or MLSB assumes that the predicted covariance matrix is nonsingular. If ML or MLSB fails because of a singular predicted covariance matrix, you need to examine whether the model specification led to the singularity. If so, modify the model specification to eliminate the problem. If not, you probably need to use a different estimation method.

Except when you use robust ML, METHOD=MLSB, or METHOD=WLS, you should remove outliers and try to transform variables that are skewed or heavy-tailed. This is because all other estimation methods depend on the sample covariance matrix, and the sample covariance matrix is a poor estimator for distributions with high kurtosis (Bollen 1989, pp. 415–418; Huber 1981; Hampel et al. 1986). PROC CALIS displays estimates of univariate and multivariate kurtosis (Bollen 1989, pp. 418–425) if you specify the KURTOSIS option in the PROC CALIS statement.

If you analyze raw data, you can use the RESIDUAL option to detect outliers and leverage observations. You can also use the PLOTS=CASERESID option to display various high-quality graphical plots for case-level residuals, outliers, and leverage observations. If outliers are present, you can either remove them from the analysis or consider using robust estimation that downweights the outliers during estimation.

See the section "Estimation Criteria" on page 1805 in Chapter 33, "The CALIS Procedure," for details about the estimation criteria. See the section "Case-Level Residuals, Outliers, Leverage Observations, and Residual Diagnostics" on page 1837 of the same Chapter for details about case-level residual diagnostics.

Statistical Inference

When you specify the ML, MLSB, FIML, GLS, or WLS estimation with appropriate models, PROC CALIS can compute the following:

- a chi-square goodness-of-fit test of the specified model versus the alternative that the data are from a population with unconstrained covariance matrix (Loehlin 1987, pp. 62–64; Bollen 1989, pp. 110, 115, 263–269)
- approximate standard errors of the parameter estimates (Bollen 1989, pp. 109, 114, 286), displayed with the STDERR option
- various modification indices, requested via the MODIFICATION or MOD option, that give the approximate change in the chi-square statistic that would result from removing constraints on the parameters or constraining additional parameters to zero (Bollen 1989, pp. 293–303)

If you have two models such that one model results from imposing constraints on the parameters of the other, you can test the constrained model against the more general model by fitting both models with PROC CALIS. If the constrained model is correct, the difference between the chi-square goodness of fit statistics for the two models has an approximate chi-square distribution with degrees of freedom equal to the difference between the degrees of freedom for the two models (Loehlin 1987, pp. 62–67; Bollen 1989, pp. 291–292).

All of the test statistics and standard errors computed under ML and GLS depend on the assumption of multivariate normality. Normality is a much more important requirement for data with random independent variables than it is for fixed independent variables. If the independent variables are random, distributions with high kurtosis tend to give liberal tests and excessively small standard errors, while low kurtosis tends to produce the opposite effects (Bollen 1989, pp. 266–267, 415–432).

All test statistics and standard errors computed by PROC CALIS are based on asymptotic theory and should not be trusted in small samples. There are no firm guidelines on how large a sample must be for the asymptotic theory to apply with reasonable accuracy. Some simulation studies have indicated that problems are likely to occur with sample sizes less than 100 (Loehlin 1987, pp. 60–61; Bollen 1989, pp. 267–268). Extrapolating from experience with multiple regression would suggest that the sample size should be at least 5 to 20 times the number of parameters to be estimated in order to get reliable and interpretable results. The WLS method might even require that the sample size be over several thousand.

The asymptotic theory requires the parameter estimates to be in the interior of the parameter space. If you do an analysis with inequality constraints and one or more constraints are active at the solution (for example, if you constrain a variance to be nonnegative and the estimate turns out to be zero), the chi-square test and standard errors might not provide good approximations to the actual sampling distributions.

For modeling correlation structures, the only theoretically correct method is the WLS method with the default ASYCOV=CORR option. For other methods, standard error estimates for modeling correlation structures might be inaccurate even for sample sizes as large as 400. The chi-square statistic is generally the same regardless of which matrix is analyzed, provided that the model involves no scale-dependent constraints.

However, if the purpose is to obtain reasonable parameter estimates for the correlation structures only, then you might also find other estimation methods useful.

If you fit a model to a correlation matrix and the model constrains one or more elements of the predicted matrix to equal 1.0, the degrees of freedom of the chi-square statistic must be reduced by the number of such constraints. PROC CALIS attempts to determine which diagonal elements of the predicted correlation matrix are constrained to a constant, but it might fail to detect such constraints in complicated models, particularly when programming statements are used. If this happens, you should add parameters to the model to release the constraints on the diagonal elements.

Multiple-Group Analysis

PROC CALIS supports multiple-group multiple-model analysis. You can fit the same covariance (and mean) structure model to several independent groups (data sets). Or, you can fit several different but constrained models to the independent groups (data sets). In PROC CALIS, you can use the GROUP statements to define several independent groups and the MODEL statements to define several different models. For example, the following statements show a multiple-group analysis by PROC CALIS:

```
proc calis;
   group 1 / data=set1;
   group 2 / data=set2;
   group 3 / data=set3;
   model 1 / group=1,2;
      path
         y <=== x   = beta ,
         x <=== z   = gamma;
   model 2 / group=3;
      path
         y <=== x   = beta,
         x <=== z   = alpha;
run;
```

In this specification, you conduct a three-group analysis. You define two PATH models. You fit Model 1 to Groups 1 and 2 and Model 2 to Group 3. The two models are constrained for the **`y <=== x`** path because they use the same path coefficient parameter beta. Other parameters in the models are not constrained.

To facilitate model specification by model referencing, you can use the REFMODEL statement to specify models based on model referencing. For example, the previous example can be specified equivalently as the following statements:

```
proc calis;
   group 1 / data=set1;
   group 2 / data=set2;
   group 3 / data=set3;
   model 1 / group=1,2;
      path
         y <=== x   = beta ,
         x <=== z   = gamma;
   model 2 / group=3;
      refmodel 1;
      renameparm gamma=alpha;
```

```
run;
```

The current specification differs from the preceding specification in the definition of Model 2. In the current specification, Model 2 is making reference to Model 1. Basically, this means that the *explicit* specification in Model 1 is transferred to Model 2. However, the RENAMEPARM statement requests a name change for gamma, which becomes a new parameter named alpha in Model 2. Hence, Model 2 and Model 1 are not the same. They are constrained by the same path coefficient beta for the **y <=== x** path, but they have different path coefficients for the **x <=== z** path.

Model referencing by the REFMODEL statement offers you an efficient and concise way to define models based on the similarities and differences between models. The advantages become more obvious when you have several large models in multiple-group analysis and each model differs just a little bit from each other.

Goodness-of-Fit Statistics

In addition to the chi-square test, there are many other statistics for assessing the goodness of fit of the predicted correlation or covariance matrix to the observed matrix.

Akaike's information criterion (AIC, Akaike 1987) and Schwarz's Bayesian criterion (SBC, Schwarz 1978) are useful for comparing models with different numbers of parameters—the model with the smallest value of AIC or SBC is considered best. Based on both theoretical considerations and various simulation studies, SBC seems to work better, since AIC tends to select models with too many parameters when the sample size is large.

There are many descriptive measures of goodness of fit that are scaled to range approximately from zero to one: the goodness-of-fit index (GFI) and GFI adjusted for degrees of freedom (AGFI) (Jöreskog and Sörbom 1988), centrality (McDonald 1989), and the parsimonious fit index (James, Mulaik, and Brett 1982). Bentler and Bonett (1980) and Bollen (1986) have proposed measures for comparing the goodness of fit of one model with another in a descriptive rather than inferential sense.

The root mean squared error approximation (RMSEA) proposed by Steiger and Lind (1980) does not assume a true model being fitted to the data. It measures the discrepancy between the fitted model and the covariance matrix in the population. For samples, RMSEA and confidence intervals can be estimated. Statistical tests for determining whether the population RMSEAs fall below certain specified values are available (Browne and Cudeck 1993). In the same vein, Browne and Cudeck (1993) propose the expected cross validation index (ECVI), which measures how good a model is for predicting future sample covariances. Point estimate and confidence intervals for ECVI are also developed.

None of these measures of goodness of fit are related to the goodness of prediction of the structural equations. Goodness of fit is assessed by comparing the observed correlation or covariance and mean matrices with the matrices computed from the model and parameter estimates. Goodness of prediction is assessed by comparing the actual values of the endogenous variables with their predicted values, usually in terms of root mean squared error or proportion of variance accounted for (R square). For latent endogenous variables, root mean squared error and R square can be estimated from the fitted model.

Customizable Fit Summary Table

Because there are so many fit indices that PROC CALIS can display and researchers prefer certain sets of fit indices, PROC CALIS enables you to customize the set of fit indices to display. For example, you can use the following statement to limit the set of fit indices to display:

```
fitindex on(only) = [chisq SRMSR RMSEA AIC];
```

With this statement, only the model-fit chi-square, standardized root mean square residual, root mean square error of approximation, and Akaiki's information criterion are displayed in your output. You can also save all your fit index values in an output data file by adding the OUTFIT= option in the FITINDEX statement. This output data file contains all available fit index values even if you have limited the set of fit indices to display in the listing output.

Standardized Solution

In many applications in social and behavioral sciences, measurement scales of variables are arbitrary. Although it should not be viewed as a universal solution, some researchers resort to the standardized solution for interpreting estimation results. PROC CALIS computes the standardized solutions for all models (except for COSAN) automatically. Standard error estimates are also produced for standardized solutions so that you can examine the statistical significance of the standardized estimates too.

However, equality or linear constraints on parameters are almost always set on the unstandardized variables. These parameter constraints are not preserved when the estimation solution is standardized. This would add difficulties in interpreting standardized estimates when your model is defined meaningfully with constraints on the unstandardized variables.

A general recommendation is to make sure your variables are measured on "comparable" scales (it does not necessarily mean that they are mean- and variance-standardized) for the analysis. But what makes different kinds of variables "comparable" is an ongoing philosophical issue.

Some researchers might totally abandon the concept of standardized solutions in structural equation modeling. If you prefer to turn off the standardized solutions in PROC CALIS, you can use the NOSTAND option in the PROC CALIS statement.

Statistical Graphics

PROC CALIS provides the following high-quality statistical graphics:

- Customizable path diagrams by using the PLOTS=PATHDIAGRAM option in the PROC CALIS statement or options in the PATHDIAGRAM statement. Path diagrams provide intuitive representations of your models and estimation results.
- Various case-level residual diagnostic plots and outlier-by-leverage plots by using the RESIDUAL and PLOTS=CASERESID options. You can use these plots to locate data points that are considered to be model outliers or influential observations.

- Residual distributions of sample moments by using the RESIDUAL and PLOTS=RESIDUAL options. You can use these plots to check whether moment residuals are distributed approximately like a normal variate. You can also use these plots to locate problematic variables that do not fit well.

Testing Parametric Functions

Oftentimes, researchers might have a priori hypotheses about the parameters in their models. After knowing the model fit is satisfactory, they want to test those hypotheses under the model. PROC CALIS provides two statements for testing these kinds of hypotheses. The TESTFUNC statement enables you to test each parametric function separately, and the SIMTESTS statement enables you to test parametric functions jointly (and separately). For example, assuming that effect1, effect2, effect3, and effect4 are parameters in your model, the following SIMTESTS statement tests the joint hypothesis test1, which consists of two component hypotheses diff_effect and sum_effect:

```
SIMTESTS  test1 = (diff_effect sum_ffect);
diff_effect = effect1 - effect2;
sum_effect  = effect3 + effect4;
```

To make test1 well-defined, each of the component hypotheses diff_effect and sum_effect is assumed to be defined as a parametric function by some SAS programming statements. In the specification, diff_effect represents the difference between effect1 and effect2, and sum_effect represents the sum of effect3 and effect4. Hence, the component hypotheses being tested are:

$$H_1: \quad \text{diff_effect} = \text{effect1} - \text{effect2} = 0$$
$$H_2: \quad \text{sum_effect} = \text{effect3} + \text{effect4} = 0$$

Effect Analysis

In structural equation modeling, effects from one variable to other variables can be direct or indirect. For example, in the following path diagram x has a direct effect on z in addition to an indirect effect on z via y:

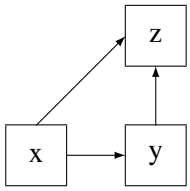

However, y has only a direct effect (but no indirect effect) on z. In cases like this, researchers are interested in computing the total, direct, and indirect effects from x and y to z. You can use the EFFPART option in the PROC CALIS statement to request this kind of effect partitioning in your model. Total, direct, and indirect effects are displayed, together with their standard error estimates. If your output contains standardized

results (default), the standardized total, direct, and indirect effects and their standard error estimates are also displayed. With the EFFPART option, effects analysis is applied to all variables (excluding error terms) in your model.

In large models with many variables, researchers might want to analyze the effects only for a handful of variables. In this regard, PROC CALIS provides you a way to do a customized version of effect analysis. For example, the following EFFPART statement requests the effect partitioning of x1 and x2 on y1 and y2, even though there might be many more variables in the model:

```
effpart   x1 x2 ===> y1 y2;
```

See the EFFPART statement of Chapter 33, "The CALIS Procedure," for details.

Model Modifications

When you fit a model and the model fit is not satisfactory, you might want to know what you could do to improve the model. The LM (Lagrange multiplier) tests in PROC CALIS can help you improve the model fit by testing the potential free parameters in the model. To request the LM tests, you can use the MODIFICATION option in the PROC CALIS statement.

The LM test results contain lists of parameters, organized according to their types. In each list, the potential parameter with the greatest model improvement is shown first. Adding these new parameters improves the model fit approximately by the amount of the corresponding LM statistic.

Sometimes, researchers might have a target set of parameters they want to test in the LM tests. PROC CALIS offers a flexible way that you can customize the set the parameters for the LM tests. See the LMTESTS statement for details.

In addition, the Wald statistics produced by PROC CALIS suggest whether any parameters in your model can be dropped (or fixed to zero) without significantly affecting the model fit. You can request the Wald statistics with the MODIFICATION option in the PROC CALIS statement.

Optimization Methods

PROC CALIS uses a variety of nonlinear optimization algorithms for computing parameter estimates. These algorithms are very complicated and do not always work for every data set. PROC CALIS generally informs you when the computations fail, usually by displaying an error message about the iteration limit being exceeded. When this happens, you might be able to correct the problem simply by increasing the iteration limit (MAXITER= and MAXFUNC=). However, it is often more effective to change the optimization method (OMETHOD=) or initial values. For more details, see the section "Use of Optimization Techniques" on page 1854 in Chapter 33, "The CALIS Procedure," and see Bollen (1989, pp. 254–256).

PROC CALIS might sometimes converge to a local optimum rather than the global optimum. To gain some protection against local optima, you can run the analysis several times with different initial estimates. The RANDOM= option in the PROC CALIS statement is useful for generating a variety of initial estimates.

Other Commonly Used Options

Other commonly used options in the PROC CALIS statement include the following:

- INMODEL= to input model specification from a data set, usually created by the OUTMODEL= option
- MEANSTR to analyze the mean structures
- NOBS to specify the number of observations
- NOPARMNAME to suppress the printing of parameter names
- NOSE to suppress the display of approximate standard errors
- OUTMODEL= to output model specification and estimation results to an external file for later use (for example, fitting the same model to other data sets)
- RESIDUAL to display residual correlations or covariances

Comparison of the CALIS and FACTOR Procedures for Exploratory Factor Analysis

Both the CALIS and the FACTOR procedures can fit exploratory factor models. However, there are several notable differences:

- By default, PROC FACTOR analyzes the correlation matrix, while PROC CALIS analyzes the covariance matrix.
- PROC FACTOR and PROC CALIS use different parameterizations in the initial factor solution. PROC CALIS uses a lower triangle pattern on the factor loading matrix (a confirmatory factor pattern) in the initial unrotated solution, while PROC FACTOR use certain matrix constraints in the initial unrotated solution. All other things being equal, PROC CALIS and PROC FACTOR might give the same solution after the same factor rotation.
- Because of the way it parameterizes, PROC FACTOR is usually more efficient computationally. PROC CALIS uses a more general algorithm that might not be computationally optimal for exploratory factor analysis.

Comparison of the CALIS and SYSLIN Procedures

The SYSLIN procedure in SAS/ETS software can fit certain kinds of path models and linear structural equation models. PROC CALIS differs from PROC SYSLIN in that PROC CALIS is more general in the use of latent variables in the models. Latent variables are unobserved, hypothetical variables, as distinct from manifest variables, which are the observed data. PROC SYSLIN allows at most one latent variable, the error term, in each equation. PROC CALIS allows several latent variables to appear in an equation—in fact, all the variables in an equation can be latent as long as there are other equations that relate the latent variables to manifest variables.

Both the CALIS and SYSLIN procedures enable you to specify a model as a system of linear equations. When there are several equations, a given variable might be a dependent variable in one equation and an independent variable in other equations. Therefore, additional terminology is needed to describe unambiguously the roles of variables in the system. Variables with values that are determined jointly and simultaneously by the system of equations are called *endogenous variables*. Variables with values that are determined outside the system—that is, in a manner separate from the process described by the system of equations—are called exogenous variables. The purpose of the system of equations is to explain the variation of each endogenous variable in terms of exogenous variables or other endogenous variables or both. See Loehlin (1987, p. 4) for further discussion of endogenous and exogenous variables. In the econometric literature, error and disturbance terms are usually distinguished from exogenous variables, but in systems with more than one latent variable in an equation, the distinction is not always clear.

In PROC SYSLIN, endogenous variables are identified by the ENDOGENOUS statement. In PROC CALIS, endogenous variables are identified by the procedure automatically after you specify the model. With different modeling languages, the identification of endogenous variables by PROC CALIS is done by different sets of rules. For example, when you specify structural equations by using the LINEQS modeling language in PROC CALIS, endogenous variables are assumed to be those that appear on the left-hand sides of the equations; a given variable can appear on the left-hand side of at most one equation. When you specify your model by using the PATH modeling language in PROC CALIS, endogenous variables are those variables pointed to by arrows at least once in the path specifications.

PROC SYSLIN provides many methods of estimation, some of which are applicable only in special cases. For example, ordinary least squares estimates are suitable in certain kinds of systems but might be statistically biased and inconsistent in other kinds. PROC CALIS provides three major methods of estimation that can be used with most models. Both the CALIS and SYSLIN procedures can do maximum likelihood estimation, which PROC CALIS calls ML and PROC SYSLIN calls FIML. PROC SYSLIN can be much faster than PROC CALIS in those special cases for which it provides computationally efficient estimation methods. However, PROC CALIS has a variety of sophisticated algorithms for maximum likelihood estimation that might be much faster than FIML in PROC SYSLIN.

PROC CALIS can impose a wider variety of constraints on the parameters, including nonlinear constraints, than can PROC SYSLIN. For example, PROC CALIS can constrain error variances or covariances to equal specified constants, or it can constrain two error variances to have a specified ratio.

References

Akaike, H. (1987). "Factor Analysis and AIC." *Psychometrika* 52:317–332.

Bartlett, M. S. (1950). "Tests of Significance in Factor Analysis." *British Journal of Psychology* 3:77–85.

Bentler, P. M. (1995). *EQS: Structural Equations Program Manual.* Program Version 5.0. Encino, CA: Multivariate Software.

Bentler, P. M., and Bonett, D. G. (1980). "Significance Tests and Goodness of Fit in the Analysis of Covariance Structures." *Psychological Bulletin* 88:588–606.

Bollen, K. A. (1986). "Sample Size and Bentler and Bonett's Nonnormed Fit Index." *Psychometrika* 51:375–377.

Bollen, K. A. (1989). *Structural Equations with Latent Variables*. New York: John Wiley & Sons.

Browne, M. W., and Cudeck, R. (1993). "Alternative Ways of Assessing Model Fit." In *Testing Structural Equation Models*, edited by K. A. Bollen and J. S. Long, 136–162. Newbury Park, CA: Sage Publications.

Duncan, O. D., Haller, A. O., and Portes, A. (1968). "Peer Influences on Aspirations: A Reinterpretation." *American Journal of Sociology* 74:119–137.

Fuller, W. A. (1987). *Measurement Error Models*. New York: John Wiley & Sons.

Haller, A. O., and Butterworth, C. E. (1960). "Peer Influences on Levels of Occupational and Educational Aspiration." *Social Forces* 38:289–295.

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J., and Stahel, W. A. (1986). *Robust Statistics: The Approach Based on Influence Functions*. New York: John Wiley & Sons.

Huber, P. J. (1981). *Robust Statistics*. New York: John Wiley & Sons.

James, L. R., Mulaik, S. A., and Brett, J. M. (1982). *Causal Analysis*. Beverly Hills, CA: Sage Publications.

Jöreskog, K. G. (1973). "A General Method for Estimating a Linear Structural Equation System." In *Structural Equation Models in the Social Sciences*, edited by A. S. Goldberger and O. D. Duncan, 85–112. New York: Academic Press.

Jöreskog, K. G., and Sörbom, D. (1979). *Advances in Factor Analysis and Structural Equation Models*. Cambridge, MA: Abt Books.

Jöreskog, K. G., and Sörbom, D. (1988). *LISREL 7: A Guide to the Program and Applications*. Chicago: SPSS.

Keesling, J. W. (1972). "Maximum Likelihood Approaches to Causal Analysis." Ph.D. diss., University of Chicago.

Loehlin, J. C. (1987). *Latent Variable Models: An Introduction to Factor, Path, and Structural Analysis*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Lord, F. M. (1957). "A Significance Test for the Hypothesis That Two Variables Measure the Same Trait Except for Errors of Measurement." *Psychometrika* 22:207–220.

McArdle, J. J., and McDonald, R. P. (1984). "Some Algebraic Properties of the Reticular Action Model for Moment Structures." *British Journal of Mathematical and Statistical Psychology* 37:234–251.

McDonald, R. P. (1989). "An Index of Goodness-of-Fit Based on Noncentrality." *Journal of Classification* 6:97–103.

Satorra, A., and Bentler, P. M. (1994). "Corrections to Test Statistics and Standard Errors in Covariance Structure Analysis." In *Latent Variables Analysis: Applications for Developmental Research*, edited by A. von Eye and C. C. Clogg, 399–419. Thousand Oaks, CA: Sage Publications.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Steiger, J. H., and Lind, J. C. (1980). "Statistically Based Tests for the Number of Common Factors." Paper presented at the annual meeting of the Psychometric Society, Iowa City, IA.

Voss, R. E. (1969). "Response by Corn to NPK Fertilization on Marshall and Monona Soils as Influenced by Management and Meteorological Factors." Ph.D. diss., Iowa State University.

Wiley, D. E. (1973). "The Identification Problem for Structural Equation Models with Unmeasured Variables." In *Structural Equation Models in the Social Sciences*, edited by A. S. Goldberger and O. D. Duncan, 69–83. New York: Academic Press.

Yung, Y.-F. (2014). "Creating Path Diagrams That Impress: A New Graphical Capability of the CALIS Procedure." SAS Institute Inc., Cary, NC. http://support.sas.com/rnd/app/stat/papers/2014/yungpd2014.pdf.

Chapter 19
Introduction to Power and Sample Size Analysis

Contents

Overview **390**
Coverage of Statistical Analyses 391
Statistical Background **392**
Hypothesis Testing, Power, and Confidence Interval Precision 392
Standard Hypothesis Tests 392
Equivalence and Noninferiority 392
Confidence Interval Precision 393
Computing Power and Sample Size 393
Power and Study Planning **394**
Components of Study Planning 394
Effect Size 395
Uncertainty and Sensitivity Analysis 395
SAS Tools for Power and Sample Size Analysis **396**
Basic Graphs (PROC POWER, PROC GLMPOWER, and Power and Sample Size in SAS Studio) 397
Highly Customized Graphs (PROC POWER and PROC GLMPOWER) 399
Formatted Tables (%POWTABLE Macro) 401
Customized Power Formulas (DATA Step) 402
Empirical Power Simulation (DATA Step, SAS/STAT Software) 403
References **404**

Overview

Power and sample size analysis optimizes the resource usage and design of a study, improving chances of conclusive results with maximum efficiency. The standard statistical testing paradigm implicitly assumes that Type I errors (mistakenly concluding significance when there is no true effect) are more costly than Type II errors (missing a truly significant result). This might be appropriate for your situation, or the relative costs of the two types of error might be reversed. For example, in screening experiments for drug development, it is often less damaging to carry a few false positives forward for follow-up testing than to miss potential leads. Power and sample size analysis can help you achieve your desired balance between Type I and Type II errors. By using optimal designs and sample sizes, you can improve your chances of detecting effects that might otherwise have been ignored, save money and time, and perhaps minimize risks to subjects.

Relevant tools for power and sample size analysis include the following:

- the GLMPOWER procedure in SAS/STAT
- the POWER procedure in SAS/STAT
- the %POWTABLE macro
- the power and sample size steps in SAS Studio
- various procedures, statements, and functions in Base SAS and SAS/STAT for developing customized formulas and simulations

These tools, discussed in detail in the section "SAS Tools for Power and Sample Size Analysis" on page 396, deal exclusively with *prospective* analysis—that is, planning for a future study. This is in contrast to *retrospective* analysis for a past study, which is not supported by the main tools. Although retrospective analysis is more convenient to perform, it is often uninformative or misleading, especially when power is computed directly based on observed data.

The goals of prospective power and sample size analysis include the following:

- determining the sample size required to get a significant result with adequate probability (power)
- characterizing the power of a study to detect a meaningful effect
- computing the probability of achieving the desired precision of a confidence interval, or the sample size required to ensure this probability
- conducting what-if analyses to assess how sensitive the power or required sample size is to other factors

The phrase *power analysis* is used for the remainder of this document as a shorthand to represent any or all of these goals. For more information about the GLMPOWER procedure, see Chapter 55, "The GLMPOWER Procedure." For more information about the POWER procedure, see Chapter 96, "The POWER Procedure." For more information about the power and sample size steps available in SAS Studio flows, see the steps whose names end in "Statistical Power" and are documented in the "Generating Statistics" section of *SAS Studio: Working with Flows*.

Coverage of Statistical Analyses

The GLMPOWER procedure covers power analysis for Type III F tests and contrasts of fixed effects in univariate and multivariate linear models. For univariate models, you can specify covariates, which can be continuous or categorical. For multivariate models, you can choose among Wilks' likelihood ratio, Hotelling-Lawley trace, and Pillai's trace F tests for multivariate analysis of variance (MANOVA) and among uncorrected, Greenhouse-Geisser, Huynh-Feldt, and Box conservative F tests for the univariate approach to repeated measures. Tests and contrasts that involve random effects are not supported.

The POWER procedure covers power analysis for the following:

- t tests, equivalence tests, and confidence intervals for means
- tests, equivalence tests, and confidence intervals for binomial proportions
- multiple regression
- tests of correlation and partial correlation
- one-way analysis of variance
- rank tests for comparing two survival curves
- Cox proportional hazards regression
- logistic regression with binary response
- Wilcoxon Mann-Whitney rank-sum test
- extensions of existing analyses that involve the chi-square, F, t, or normal distribution, or the distribution of the correlation coefficient under multivariate normality

The extensions of existing analyses consist of scalar multipliers and custom values for primary noncentrality and degrees of freedom. Important use cases include the following:

- Wald and likelihood ratio tests in generalized linear models that have nominal, count, or ordinal responses
- sample size inflation that is caused by correlated predictors
- sample size deflation that is caused by correlation between the covariates and the response

Examples of generalized linear models include Poisson regression, logistic regression, and zero-inflated models.

The power and sample size steps available in SAS Studio flows cover a large subset of the analyses in the GLMPOWER and POWER procedures.

Statistical Background

Hypothesis Testing, Power, and Confidence Interval Precision

Standard Hypothesis Tests

In statistical hypothesis testing, you typically express the belief that some effect exists in a population by specifying an alternative hypothesis H_1. You state a null hypothesis H_0 as the assertion that the effect does *not* exist and attempt to gather evidence to reject H_0 in favor of H_1. Evidence is gathered in the form of sample data, and a statistical test is used to assess H_0. If H_0 is rejected but there really is *no* effect, this is called a *Type I error*. The probability of a Type I error is usually designated "alpha" or α, and statistical tests are designed to ensure that α is suitably small (for example, less than 0.05).

If there is an effect in the population but H_0 is *not* rejected in the statistical test, then a *Type II error* has been committed. The probability of a Type II error is usually designated "beta" or β. The probability $1 - \beta$ of avoiding a Type II error (that is, correctly rejecting H_0 and achieving statistical significance) is called the *power* of the test.

Most, but not all, of the power analyses in the GLMPOWER and POWER procedures are based on such standard hypothesis tests.

Equivalence and Noninferiority

Whereas the standard two-sided hypothesis test for a parameter μ (such as a mean difference) aims to demonstrate that it is significantly different than a null value μ_0:

$$H_0{:}\mu = \mu_0$$
$$H_1{:}\mu \neq \mu_0$$

an equivalence test instead aims to demonstrate that it is *significantly similar* to some value, expressed in terms of a range θ_L, θ_U around that value:

$$H_0{:}\mu < \theta_L \quad \text{or} \quad \mu > \theta_U$$
$$H_1{:}\theta_L \leq \mu \leq \theta_U$$

Whereas the standard one-sided hypothesis test for μ (say, the upper one-sided test) aims to demonstrate that it is significantly greater than μ_0:

$$H_0{:}\mu \leq \mu_0$$
$$H_1{:}\mu > \mu_0$$

a corresponding noninferiority test aims to demonstrate that it is *not significantly less* than μ_0, expressed in terms of a margin $\delta > 0$:

$$H_0{:}\mu \leq \mu_0 - \delta$$
$$H_1{:}\mu > \mu_0 - \delta$$

Corresponding forms of these hypotheses with the inequalities reversed apply to lower one-sided noninferiority tests (sometimes called *nonsuperiority* tests).

The POWER procedure performs power analyses for equivalence tests for one-sample, paired, and two-sample tests of normal and lognormal mean differences and ratios. It also supports noninferiority tests for a variety of analyses of means, proportions, and correlation, both directly (with a MARGIN= option representing δ) and indirectly (with an option for a custom null value representing the sum or difference of μ_0 and δ).

Confidence Interval Precision

An analysis of confidence interval precision is analogous to a traditional power analysis, with *CI Half-Width* taking the place of effect size and *Prob(Width)* taking the place of power. The *CI Half-Width* is the margin of error associated with the confidence interval, the distance between the point estimate and an endpoint. The *Prob(Width)* is the probability of obtaining a confidence interval with *at most* a target half-width.

The POWER procedure performs confidence interval precision analyses for *t*-based confidence intervals for one-sample, paired, and two-sample designs, and for several varieties of confidence intervals for a binomial proportion.

Computing Power and Sample Size

For some statistical models and tests, power analysis calculations are exact—that is, they are based on a mathematically accurate formula that expresses power in terms of the other components. Such formulas typically involve either enumeration or noncentral versions of the distribution of the test statistic.

When a power computation is based on a noncentral t, F, or chi-square distribution, the noncentrality parameter generally has the same form as the test statistic, with the conjectured population parameters in place of their corresponding estimators.

For example, the test statistic for a two-sample t test is computed as follows:

$$t = N^{\frac{1}{2}}(w_1 w_2)^{\frac{1}{2}} \left(\frac{\bar{x}_2 - \bar{x}_1 - \mu_0}{s_p} \right)$$

where N is the total sample size, w_1 and w_2 are the group allocation weights, $\bar{x}_1$ and $\bar{x}_2$ are the sample means, μ_0 is the null mean difference, and s_p is the pooled standard deviation. Under the null hypothesis, the statistic $F = t^2$ is distributed as $F(1, N-2)$. In general, F has a noncentral F distribution $F(1, N-2, \delta^2)$ where

$$\delta = N^{\frac{1}{2}}(w_1 w_2)^{\frac{1}{2}} \left(\frac{\mu_{\text{diff}} - \mu_0}{\sigma} \right)$$

and μ_{diff} and σ are the (unknown) true mean difference and common group standard deviation, respectively. Note that the square-root noncentrality δ is exactly the same as the t statistic except that the estimators of mean difference and standard deviation are replaced by their corresponding true population values.

The power for the two-sided two-sample t test with significance level α is computed as

$$P\left(F \geq F_{1-\alpha}(1, N-2)\right)$$

where F is distributed as $F(1, N-2, \delta^2)$ and $F_{1-\alpha}(1, N-2)$ is the $100(1-\alpha)\%$ quantile of the central F distribution with 1 and $N-2$ degrees of freedom. See the section "Customized Power Formulas (DATA Step)" on page 402 for an example of the implementation of this formula in the DATA step.

In the absence of exact mathematical results, approximate formulas can sometimes be used. When neither exact power computations nor reasonable approximations are possible, simulation provides an increasingly viable alternative. You specify values for model parameters and use them to randomly generate a large number of hypothetical data sets. Applying the statistical test to each data set, you estimate power with the percentage of times the null hypothesis is rejected. While the simulation approach is computationally intensive, faster computing makes this less of an issue. A simulation-based power analysis is always a valid option, and, with a large number of data set replications, it can often be more accurate than approximations. See the section "Empirical Power Simulation (DATA Step, SAS/STAT Software)" on page 403 for an example of an empirical power simulation.

Sample size is usually computed by iterative numerical methods because it often cannot be expressed in closed form as a function of the other parameters. Sample size tends to appear in both a noncentrality parameter and a degrees of freedom term for the critical value.

Power and Study Planning

Power analysis is most effective when performed at the study planning stage, and as such it encourages early collaboration between researcher and statistician. It also focuses attention on effect sizes and variability in the underlying scientific process, concepts that both researcher and statistician should consider carefully at this stage.

There are many factors involved in a power analysis, such as the research objective, design, data analysis method, power, sample size, Type I error, variability, and effect size. By performing a power analysis, you can learn about the relationships between these factors, optimizing those that are under your control and exploring the implications of those that are fixed or unknown.

Components of Study Planning

Even when the research questions and study design seem straightforward, the ensuing power analysis can seem technically daunting. It is often helpful to break the process down into five components:

- **Study Design**: What is the structure of the planned design? This must be clearly and completely specified. What groups and treatments ("cells" and "factors" of the design) are going to be assessed, and what will be the relative sizes of those cells? How is each case going to be studied—that is, what is the primary outcome measure ("dependent variable")? Will covariates be measured and included in the statistical model?

- **Scenario Model**: What are your beliefs about patterns in the data? Imagine that you had unlimited time and resources to execute the study design, so that you could gather an "infinite data set." Characterize that infinite data set as best you can using a mathematical model, realizing that it will be a simplification of reality. Alternatively, as is common with complex linear models, you may decide to construct an "exemplary" data set that mimics the infinite data set. However you do this, your scenario model should

capture the key features of the study design and the main relationships among the primary outcome variables and study factors.

- **Effects and Variability**: What exactly are the "signals and noises" in the patterns you suspect? Set specific values for the parameters of your scenario model, keeping at most one unspecified. It is often enlightening to consider a variety of realistic possibilities for the key values by performing a sensitivity analysis, to explore the consequences of competing views on what the infinite data set might look like.
- **Statistical Method**: How will you cast your model in statistical terms and conduct the eventual data analysis? Define the statistical models and procedures that will be used to embody the study design and estimate and/or test the effects central to the research question. What significance levels will be used? Will one- or two-sided tests be used?
- **Aim of Assessment**: Finally, what needs to be determined in the power analysis? Most often you want to examine the statistical powers obtained across the various scenarios for the effects, variability, alternative varieties of the statistical procedures to be used, and the feasible total sample sizes. Sometimes the goal is to find sample size values that provide given levels of power, say 85%, 90%, or 95%.

Effect Size

There is some confusion in practice about how to postulate the effect size. One alternative is to specify the effect size that represents minimal clinical significance; then the result of the power analysis reveals the chances of detecting a minimally meaningful effect size. Often this minimal effect size is so small that it requires excessive resources to detect. Another alternative is to make an educated guess of the true underlying effect size. Then the power analysis determines the chance of detecting the effect size that is believed to be true. The choice is ultimately determined by the research goals. Finally, you can specify a collection of possible values, perhaps spanning the range between minimally meaningful effects and larger surmised effects.

You can arrive at values for required quantities in a power analysis, such as effect sizes and measures of variability, in many different ways. For example, you can use pilot data, results of previous studies reported in literature, educated guesses derived from theory, or educated guesses derived from partial data (a small sample or even just quantiles).

Uncertainty and Sensitivity Analysis

Uncertainty is a fact of life in any power analysis, because at least some of the numbers used are best guesses of unknown values. The result of a power calculation, whether it be achieved power or required sample size or something else, serves only as a point estimate, conditional on the conjectured values of the other components. It is not feasible in general to quantify the variability involved in using educated guesses or undocumented results to specify these components. If observed data are used, relevant adjustments for variability in the data tend to be problematic in the sense of producing confidence intervals for power that are too wide for practical use. But there is a useful way for you to characterize the uncertainty in your power analysis, and also discover the extent to which statistical power is affected by each component. You can posit a reasonable range for each input component, vary each one within its range, and observe the variety of results in the form of tables or graphs.

SAS Tools for Power and Sample Size Analysis

This section demonstrates how you can use the different SAS power analysis tools mentioned in the section "Overview" on page 390 to generate graphs, tables, and narratives; implement your own power formulas; and simulate empirical power.

Suppose you want to compute the power of a two-sample *t* test. You conjecture that the mean difference is between 5 and 6 and that the common group standard deviation is between 12 and 18. You plan to use a significance level between 0.05 and 0.1 and a sample size between 100 and 200. The following SAS statements use the POWER procedure to compute the power for these scenarios:

```
proc power;
   twosamplemeans test=diff
      meandiff = 5 6
      stddev = 12 18
      alpha = 0.05 0.1
      ntotal = 100 200
      power = .;
run;
```

Figure 19.1 shows the results. Depending on the plausibility of the various combinations of input parameter values, the power ranges between 0.379 and 0.970.

Figure 19.1 PROC POWER Tabular Output

The POWER Procedure
Two-Sample t Test for Mean Difference

Computed Power					
Index	Alpha	Mean Diff	Std Dev	N Total	Power
1	0.05	5	12	100	0.541
2	0.05	5	12	200	0.834
3	0.05	5	18	100	0.280
4	0.05	5	18	200	0.498
5	0.05	6	12	100	0.697
6	0.05	6	12	200	0.940
7	0.05	6	18	100	0.379
8	0.05	6	18	200	0.650
9	0.10	5	12	100	0.664
10	0.10	5	12	200	0.902
11	0.10	5	18	100	0.397
12	0.10	5	18	200	0.623
13	0.10	6	12	100	0.799
14	0.10	6	12	200	0.970
15	0.10	6	18	100	0.505
16	0.10	6	18	200	0.759

The following seven sections illustrate additional ways of displaying these results using the different SAS tools.

Basic Graphs (PROC POWER, PROC GLMPOWER, and Power and Sample Size in SAS Studio)

If you include a PLOT statement, the GLMPOWER and POWER procedures produce standard power curves, which represent any multivalued input parameters that have varying line styles, symbols, colors, panels, or any combination of these. The power and sample size steps available in SAS Studio flows also provide options that enable you to produce power curves. If ODS Graphics is enabled, then graphs are created using ODS Graphics; otherwise, traditional graphs are produced.

To display default power curves for the preceding PROC POWER call, add the PLOT statement with no arguments as follows:

```
ods graphics on;

proc power plotonly;
   twosamplemeans test=diff
      meandiff = 5 6
      stddev = 12 18
      alpha = 0.05 0.1
      ntotal = 100 200
      power = .;
   plot;
run;

ods graphics off;
```

The ODS GRAPHICS ON statement enables ODS Graphics.

Figure 19.2 shows the results. Note that the line style varies by the significance level α, the symbol varies by the mean difference, and the panel varies by standard deviation.

Figure 19.2 PROC POWER Default Graphical Output

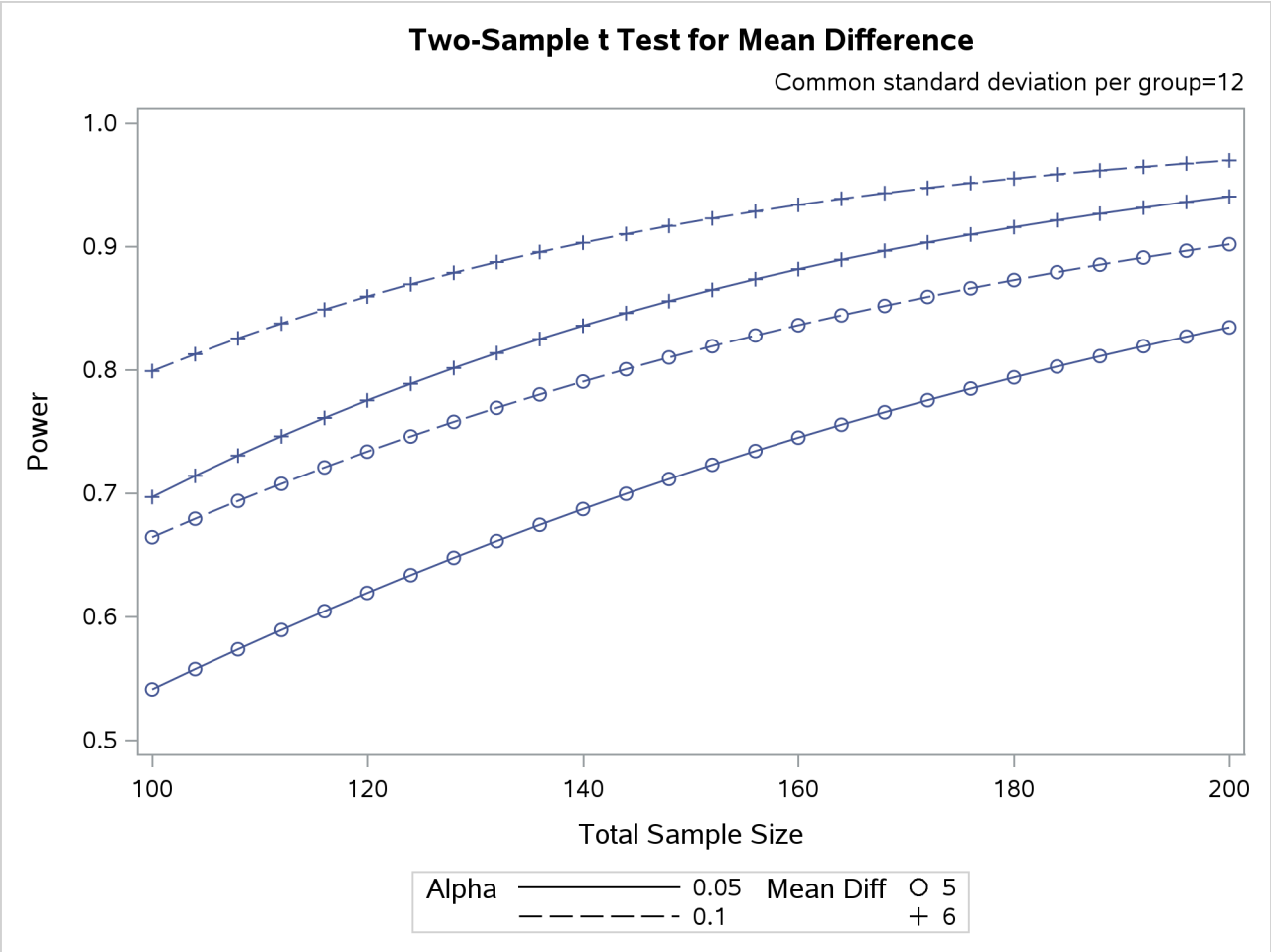

Figure 19.2 *continued*

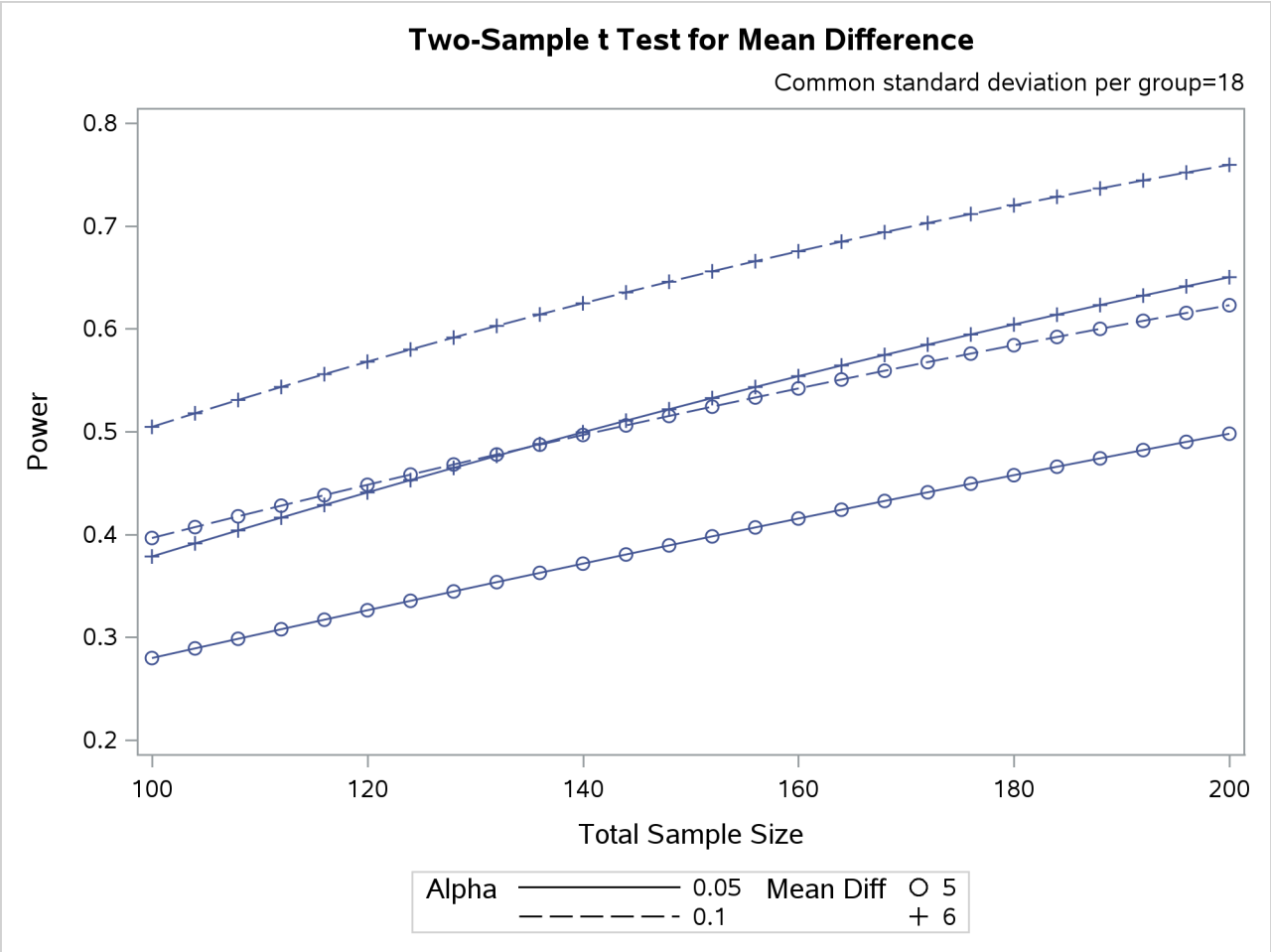

Highly Customized Graphs (PROC POWER and PROC GLMPOWER)

Example 96.8 of Chapter 96, "The POWER Procedure," demonstrates various ways you can modify and enhance plots created in the GLMPOWER or POWER procedures:

- assigning analysis parameters to axes
- fine-tuning a sample size axis
- adding reference lines
- linking plot features to analysis parameters
- choosing key (legend) styles
- modifying symbol locations

For example, replace the default PLOT statement with the following statement to modify the graphical results in Figure 19.2 to lower the minimum sample size to 60, show a reference line at power=0.9 with

corresponding sample size values, distinguish standard deviation by color instead of panel, and swap the roles of α and mean difference:

```
plot
   min=60
   yopts=(ref=0.9 crossref=yes)
   vary(color by stddev, linestyle by meandiff, symbol by alpha);
```

Figure 19.3 shows the results. The plot reveals that only the scenarios with the largest mean difference and smallest standard deviation achieve a power of at least 0.9 for this sample size range.

Figure 19.3 PROC POWER Customized Graphical Output

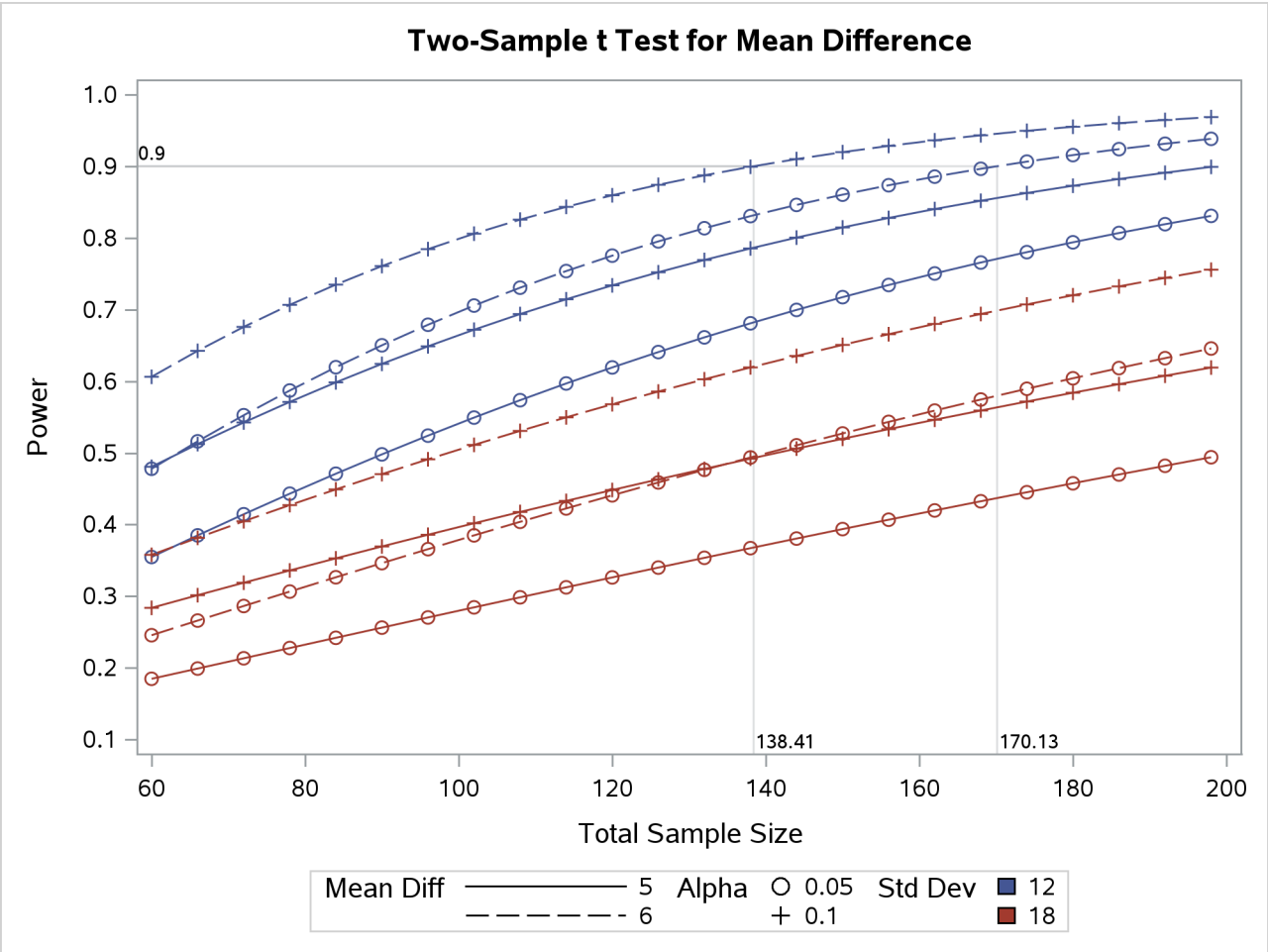

Formatted Tables (%POWTABLE Macro)

The %POWTABLE macro renders the output of the POWER and GLMPOWER procedures in rectangular form, and it optionally produces simplified results using weighted means across chosen variables. PROC REPORT and the Output Delivery System (ODS) are used to generate the tables. Base SAS and SAS/STAT 9.1 or higher versions are required.

You can run the %POWTABLE macro for the output in Figure 19.1 to display the results in a form more suitable for quickly discerning relationships among parameters. First use the ODS OUTPUT statement to assign the "Output" table produced by the POWER procedure to a data set as follows:

```
ods output output=powdata;
```

Next, specify the same PROC POWER statements that generate Figure 19.1. Finally, use the %POWTABLE macro to assign analysis parameters to table dimensions. To create a table of computed power values with mean difference assigned to rows, sample size and α assigned to columns, and standard deviation assigned to "panels" (rendered by default as rows separated by blank lines), specify the following statements:

```
%powtable ( Data = powdata,
            Entries = power,
            Rows = meandiff,
            Cols = ntotal alpha,
            Panels = stddev )
```

Figure 19.4 shows the results.

Figure 19.4 %POWTABLE Macro Output

The POWTABLE Macro

Entries are Power

		N Total			
		100		200	
		Alpha		Alpha	
		0.05	0.10	0.05	0.10
Std Dev	Mean Diff				
12	5	0.541	0.664	0.834	0.902
	6	0.697	0.799	0.940	0.970
18	5	0.280	0.397	0.498	0.623
	6	0.379	0.505	0.650	0.759

Customized Power Formulas (DATA Step)

If you want to perform a power computation for an analysis that is not currently supported directly in SAS/STAT tools, and you have a power formula, then you can program the formula in the DATA step.

For purposes of illustration, here is the power formula in the section "Computing Power and Sample Size" on page 393 implemented in the DATA step to compute power for the *t* test example:

```
data tpow;
   do meandiff = 5, 6;
      do stddev = 12, 18;
         do alpha = 0.05, 0.1;
            do ntotal = 100, 200;
               ncp = ntotal * 0.5 * 0.5 * meandiff**2 / stddev**2;
               critval = finv(1-alpha, 1, ntotal-2, 0);
               power = sdf('f', critval, 1, ntotal-2, ncp);
               output;
            end;
         end;
      end;
   end;
run;

proc print data=tpow;
run;
```

The output is shown in Figure 19.5.

Figure 19.5 Customized Power Formula (DATA Step)

Obs	meandiff	stddev	alpha	ntotal	ncp	critval	power
1	5	12	0.05	100	4.3403	3.93811	0.54102
2	5	12	0.05	200	8.6806	3.88885	0.83447
3	5	12	0.10	100	4.3403	2.75743	0.66434
4	5	12	0.10	200	8.6806	2.73104	0.90171
5	5	18	0.05	100	1.9290	3.93811	0.27981
6	5	18	0.05	200	3.8580	3.88885	0.49793
7	5	18	0.10	100	1.9290	2.75743	0.39654
8	5	18	0.10	200	3.8580	2.73104	0.62287
9	6	12	0.05	100	6.2500	3.93811	0.69689
10	6	12	0.05	200	12.5000	3.88885	0.94043
11	6	12	0.10	100	6.2500	2.75743	0.79895
12	6	12	0.10	200	12.5000	2.73104	0.96985
13	6	18	0.05	100	2.7778	3.93811	0.37857
14	6	18	0.05	200	5.5556	3.88885	0.65012
15	6	18	0.10	100	2.7778	2.75743	0.50459
16	6	18	0.10	200	5.5556	2.73104	0.75935

Empirical Power Simulation (DATA Step, SAS/STAT Software)

You can obtain a highly accurate power estimate by simulating the power empirically. You need to use this approach for analyses that are not supported directly in SAS/STAT tools and for which you lack a power formula. But the simulation approach is also a viable alternative to existing power approximations. A high number of simulations will yield a more accurate estimate than a non-exact power approximation.

Although exact power computations for the two-sample *t* test are supported in several of the SAS/STAT tools, suppose for purposes of illustration that you want to simulate power for the continuing *t* test example. This section describes how you can use the DATA step and SAS/STAT software to do this.

The simulation involves generating a large number of data sets according to the distributions defined by the power analysis input parameters, computing the relevant *p*-value for each data set, and then estimating the power as the proportion of times that the *p*-value is significant.

The following statements compute a power estimate along with a 95% confidence interval for power for the first scenario in the two-sample *t* test example, with 10,000 simulations:

```
%let meandiff =     5;
%let stddev   =    12;
%let alpha    =  0.05;
%let ntotal   =   100;
%let nsim     = 10000;

data simdata;
   call streaminit(123);
   do isim = 1 to &nsim;
      do i = 1 to floor(&ntotal/2);
         group = 1;
         y = rand('normal', 0        , &stddev);
         output;
         group = 2;
         y = rand('normal', &meandiff, &stddev);
         output;
      end;
   end;
run;

ods exclude all;
proc ttest data=simdata;
   ods output ttests=tests;
   by isim;
   class group;
   var y;
run;
ods exclude none;

data tests;
   set tests;
   where method="Pooled";
   issig = probt < &alpha;
run;
```

```
proc freq data=tests;
   ods select binomial;
   tables issig / binomial(level='1');
run;
```

First the DATA step is used to randomly generate *nsim* = 10,000 data sets based on the *meandiff*, *stddev*, and *ntotal* parameters and the normal distribution, consistent with the assumptions underlying the two-sample *t* test. These data sets are contained in a large SAS data set called simdata indexed by the variable isim.

The CALL STREAMINIT(123) statement initializes the random number generator with a specific sequence and ensures repeatable results for purposes of this example. (**NOTE:** Skip this step when you are performing actual power simulations.)

The TTEST procedure is run using isim as a BY variable, with the ODS EXCLUDE ALL statement to suppress output. The ODS OUTPUT statement saves the "TTests" table to a data set called tests. The *p*-values are contained in a column called probt.

The subsequent DATA step defines a variable called issig to flag the significant *p*-values.

Finally, the FREQ procedure computes the empirical power estimate as the estimate of P(issig = 1) and provides approximate and exact confidence intervals for this estimate.

Figure 19.6 shows the results. The estimated power is 0.5388 with 95% confidence interval (0.5290, 0.5486). Note that the exact power of 0.541 shown in the first row in Figure 19.1 is contained within this tight confidence interval.

Figure 19.6 Simulated Power (DATA Step, SAS/STAT Software)

The FREQ Procedure

Binomial Proportion issig = 1	
Proportion	0.5388
ASE	0.0050
95% Lower Conf Limit	0.5290
95% Upper Conf Limit	0.5486
Exact Conf Limits	
95% Lower Conf Limit	0.5290
95% Upper Conf Limit	0.5486

References

Castelloe, J. M. (2000). "Sample Size Computations and Power Analysis with the SAS System." In *Proceedings of the Twenty-Fifth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi25/25/st/25p265.pdf.

Castelloe, J. M., and O'Brien, R. G. (2001). "Power and Sample Size Determination for Linear Models." In *Proceedings of the Twenty-Sixth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi26/p240-26.pdf.

Lenth, R. V. (2001). "Some Practical Guidelines for Effective Sample Size Determination." *American Statistician* 55:187–193.

Muller, K. E., and Benignus, V. A. (1992). "Increasing Scientific Power with Statistical Power." *Neurotoxicology and Teratology* 14:211–219.

O'Brien, R. G., and Castelloe, J. M. (2007). "Sample-Size Analysis for Traditional Hypothesis Testing: Concepts and Issues." In *Pharmaceutical Statistics Using SAS: A Practical Guide*, edited by A. Dmitrienko, C. Chuang-Stein, and R. D'Agostino, 237–271. Cary, NC: SAS Institute Inc.

O'Brien, R. G., and Muller, K. E. (1993). "Unified Power Analysis for *t*-Tests through Multivariate Hypotheses." In *Applied Analysis of Variance in Behavioral Science*, edited by L. K. Edwards, 297–344. New York: Marcel Dekker.

Chapter 20
Shared Concepts and Topics

Contents

Levelization of Classification Variables **409**
Parameterization of Model Effects **412**
GLM Parameterization of Classification Variables and Effects 412
Intercept 412
Regression Effects 412
Main Effects 413
Interaction Effects 413
Nested Effects 414
Continuous-Nesting-Class Effects 414
Continuous-by-Class Effects 415
General Effects 415
Other Parameterizations 416
Using CAS Tables **420**
Procedures Incompatible with Input CAS Tables 421
Options Incompatible with Input CAS Tables 421
Special Data Sets Incompatible with CAS Tables 421
Special Requirements for Processing Some CAS Tables 422
Statistical Results Not Reliably Reproducible with Input CAS Tables 422
Observations Might Not Be Identified Reliably with Input CAS Tables 423
Best Practices for Using CAS Tables with SAS/STAT Procedures 423
CODE Statement **424**
Syntax: CODE Statement 425
EFFECT Statement **427**
Collection Effects 429
Lag Effects 429
Multimember Effects 432
Polynomial Effects 434
Spline Effects 437
Splines and Spline Bases 441
Truncated Power Function Basis 442
B-Spline Basis 443
Natural Cubic Spline Basis 445
ODS Table Names 446
EFFECTPLOT Statement **447**
Syntax: EFFECTPLOT Statement 447

Dictionary of Options 450
ODS Graphics: EFFECTPLOT Statement 459
Examples: EFFECTPLOT Statement 460
Example 20.1: A Saddle Surface 460
Example 20.2: Unbalanced Two-Way ANOVA 463
Example 20.3: Logistic Regression 470
ESTIMATE Statement **475**
Syntax: ESTIMATE Statement 476
Positional and Nonpositional Syntax for Coefficients in Linear Functions 486
Joint Hypothesis Tests with Complex Alternatives, the Chi-Bar-Square Statistic 489
ODS Table Names: ESTIMATE Statement 490
ODS Graphics: ESTIMATE Statement 491
LSMEANS Statement **492**
Syntax: LSMEANS Statement 493
Construction of Least Squares Means 507
Setting Covariate Values 509
Changing the Weighting Scheme 510
Estimability of LS-Means 510
ODS Table Names: LSMEANS Statement 511
ODS Graphics: LSMEANS Statement 511
LSMESTIMATE Statement **513**
Syntax: LSMESTIMATE Statement 514
ODS Table Names: LSMESTIMATE Statement 524
ODS Graphics: LSMESTIMATE Statement 524
MARGINS Statement **525**
Syntax: MARGINS Statement 526
Predictive Margins Compared with LS-Means 532
ODS Table Names: MARGINS Statement 533
NLOPTIONS Statement **533**
Syntax: NLOPTIONS Statement 533
Choosing an Optimization Algorithm 545
First- or Second-Order Algorithms 545
Algorithm Descriptions 546
SLICE Statement **550**
Syntax: SLICE Statement 551
ODS Table Names: SLICE Statement 552
ODS Graphics: SLICE Statement 553
STORE Statement **554**
Syntax: STORE Statement 555
TEST Statement **556**
Syntax: TEST Statement 556
ODS Table Names: TEST Statement 557
Programming Statements **558**
Convergence Status **560**
References **562**

This chapter introduces a number of concepts that are common to two or more SAS/STAT procedures. Most sections display a listing of the procedures for which the shared topic is relevant.

Levelization of Classification Variables

A classification variable is a variable that enters the statistical analysis or model not through its values, but through its levels. The process of associating values of a variable with levels is termed *levelization*.

This section covers in particular procedures that support a CLASS statement for specifying classification variables. Some of the concepts discussed also apply to procedures that use different syntax to request levelization of variables (for example, the CLASS() expansion in the TRANSREG procedure).

During the process of levelization, observations that share the same value are assigned to the same level. The manner in which values are grouped can be affected by the inclusion of formats. The sort order of the levels can be determined with the ORDER= option in the procedure statement. With the GENMOD, GLMSELECT, and LOGISTIC procedures, you can also control the sort order separately for each variable in the CLASS statement.

Consider the data on nine observations in Table 20.1. The variable A is integer valued, and the variable X is a continuous variable with a missing value for the fourth observation. The fourth and fifth columns of Table 20.1 apply two different formats to the variable X.

Table 20.1 Example Data for Levelization

Obs	A	x	FORMAT x 3.0	FORMAT x 3.1
1	2	1.09	1	1.1
2	2	1.13	1	1.1
3	2	1.27	1	1.3
4	3	.	.	.
5	3	2.26	2	2.3
6	3	2.48	2	2.5
7	4	3.34	3	3.3
8	4	3.34	3	3.3
9	4	3.14	3	3.1

By default, levelization of the variables groups observations by the formatted value of the variable, except for numerical variables for which no explicit format is provided. Numerical variables for which no explicit format is provided are sorted by their internal value. The levelization of the four columns in table Table 20.1 leads to the level assignment in Table 20.2.

Table 20.2 Values and Levels

	A		X		FORMAT x 3.0		FORMAT x 3.1	
Obs	**Value**	**Level**	**Value**	**Level**	**Value**	**Level**	**Value**	**Level**
1	2	1	1.09	1	1	1	1.1	1
2	2	1	1.13	2	1	1	1.1	1
3	2	1	1.27	3	1	1	1.3	2
4	3	2	.	.	.	.	.	.
5	3	2	2.26	4	2	2	2.3	3
6	3	2	2.48	5	2	2	2.5	4
7	4	3	3.34	7	3	3	3.3	6
8	4	3	3.34	7	3	3	3.3	6
9	4	3	3.14	6	3	3	3.1	5

The ORDER= option in the PROC statement specifies the sort order for the levels of CLASS variables. When ORDER=FORMATTED (which is the default) is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. To order numeric class levels with no explicit format by their BEST12. formatted values, you can specify the BEST12. format explicitly for the CLASS variables.

The following table shows how values of the ORDER= option are interpreted.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

For FORMATTED and INTERNAL values, the sort order is machine dependent. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

The GLMSELECT, LOGISTIC, and GENMOD procedures support a MISSING option in the CLASS statement. When this option is in effect, missing values (. for a numeric variable and blanks for a character variable) are included in the levelization and are assigned a level. Table 20.4 displays the results of levelizing the values in Table 20.1 when the MISSING option is in effect.

Table 20.4 Values and Levels with MISSING Option

	A		X		FORMAT x 3.0		FORMAT x 3.1	
Obs	**Value**	**Level**	**Value**	**Level**	**Value**	**Level**	**Value**	**Level**
1	2	1	1.09	2	1	2	1.1	2
2	2	1	1.13	3	1	2	1.1	2
3	2	1	1.27	4	1	2	1.3	3
4	3	2	.	1	.	1	.	1
5	3	2	2.26	5	2	3	2.3	4
6	3	2	2.48	6	2	3	2.5	5
7	4	3	3.34	8	3	4	3.3	7
8	4	3	3.34	8	3	4	3.3	7
9	4	3	3.14	7	3	4	3.1	6

When the MISSING option is not specified, or for procedures whose CLASS statement does not support this option, it is important to understand the implications of missing values for your statistical analysis. When a SAS/STAT procedure levelizes the CLASS variables, an observation for which a CLASS variable has a missing value is excluded from the analysis. This is true regardless of whether the variable is used to form the statistical model. Consider, for example, the case where some observations contain missing values for variable A but the records for these observations are otherwise complete with respect to all other variables in the statistical models. The analysis results from the following statements do not include any observations for which variable A contains missing values, even though A is not specified in the MODEL statement:

```
class A B;
model y = B x B*x;
```

Many statistical procedures print a “Number of Observations” table that shows the number of observations read from the data set and the number of observations used in the analysis. Pay careful attention to this table—especially when your data set contains missing values—to ensure that no observations are unintentionally excluded from the analysis.

Parameterization of Model Effects

The general form of a linear regression model is defined in the section "Regression Models and Models with Classification Effects" on page 28 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

This section describes how matrices of regressor effects such as **X** are constructed in SAS/STAT software. These constructions (*parameterization* rules) apply to regression models, models with classification effects, generalized linear models, and mixed models. The simplest and most general parameterization rules are the ones used in the GLM procedure, and they are discussed first. Several procedures also support alternate parameterizations of classification variables, including the CATMOD, GENMOD, GLMSELECT, LOGISTIC, PHREG, SURVEYLOGISTIC, and SURVEYPHREG procedures. These are discussed after the GLM parameterization of classification variables and model effects.

All modeling procedures that have a CLASS statement support classification variables and effects, and those procedures that additionally support the supplemental parameterizations have a PARAM= option in the CLASS statement.

GLM Parameterization of Classification Variables and Effects

This section applies to the following procedures:
GAM, GENMOD, GLIMMIX, GLM, GLMPOWER, GLMSELECT, LIFEREG, LOGISTIC, MI, MIXED, MULLTEST, ORTHOREG, PHREG, PLS, QUANTREG, ROBUSTREG, SURVEYLOGISTIC, and SURVEYPHREG.

Intercept

By default, SAS/STAT linear models automatically include a column of 1s in **X** which corresponds to an intercept parameter. In many procedures you can use the NOINT option in the MODEL statement to suppress this intercept. For example, the NOINT option is useful when the MODEL statement contains a classification effect and you want the parameter estimates to be in terms of the mean response for each level of that effect.

Regression Effects

Numeric variables or polynomial terms that involve them can be included in the model as regression effects (covariates). The actual values of such terms are included as columns of the relevant model matrices. You can use the bar operator with a regression effect to generate polynomial effects. For example, X|X|X expands to X X*X X*X*X, which is a cubic model.

Main Effects

If a classification variable has m levels, the GLM parameterization generates m columns for its main effect in the model matrix. Each column is an indicator variable for a given level. The order of the columns is the sort order of the values of their levels and frequently can be controlled with the ORDER= option in the procedure or CLASS statement.

Table 20.5 is an example where β_0 denotes the intercept and A and B are classification variables with two and three levels, respectively.

Table 20.5 Example of Main Effects

Data		**I**	A		B		
A	B	β_0	A1	A2	B1	B2	B3
1	1	1	1	0	1	0	0
1	2	1	1	0	0	1	0
1	3	1	1	0	0	0	1
2	1	1	0	1	1	0	0
2	2	1	0	1	0	1	0
2	3	1	0	1	0	0	1

Typically, there are more columns for these effects than there are degrees of freedom to estimate them. In other words, the GLM parameterization of main effects is *singular*.

Interaction Effects

Often a model includes interaction (crossed) effects to account for how the effect of a variable changes with the values of other variables. With an interaction, the terms are first reordered to correspond to the order of the variables in the CLASS statement. Thus, B*A becomes A*B if A precedes B in the CLASS statement. Then, the GLM parameterization generates columns for all combinations of levels that occur in the data. The order of the columns is such that the rightmost variables in the interaction change faster than the leftmost variables (Table 20.6). In the MIXED and GLIMMIX procedures, which support both fixed- and random-effects models, empty columns (that is, columns that would contain all 0s) are not generated for fixed effects, but they are generated for random effects.

Table 20.6 Example of Interaction Effects

Data		**I**	A		B			A*B					
A	B	β_0	A1	A2	B1	B2	B3	A1B1	A1B2	A1B3	A2B1	A2B2	A2B3
1	1	1	1	0	1	0	0	1	0	0	0	0	0
1	2	1	1	0	0	1	0	0	1	0	0	0	0
1	3	1	1	0	0	0	1	0	0	1	0	0	0
2	1	1	0	1	1	0	0	0	0	0	1	0	0
2	2	1	0	1	0	1	0	0	0	0	0	1	0
2	3	1	0	1	0	0	1	0	0	0	0	0	1

In the preceding matrix, main-effects columns are not linearly independent of crossed-effects columns; in fact, the column space for the crossed effects contains the space of the main effect.

When your model contains many interaction effects, you might be able to code them more parsimoniously by using the bar operator (|). The bar operator generates all possible interaction effects. For example, A|B|C expands to A B A*B C A*C B*C A*B*C. To eliminate higher-order interaction effects, use the at sign (@) in conjunction with the bar operator. For instance, A|B|C|D@2 expands to A B A*B C A*C B*C D A*D B*D C*D.

Nested Effects

Nested effects are generated in the same manner as crossed effects. Hence, the design columns generated by the following two statements are the same (but the ordering of the columns is different):

```
model Y=A B(A);
model Y=A A*B;
```

The nesting operator in SAS/STAT software is more of a notational convenience than an operation distinct from crossing. Nested effects are typically characterized by the property that the nested variables never appear as main effects. The order of the variables within nesting parentheses is made to correspond to the order of these variables in the CLASS statement. The order of the columns is such that variables outside the parentheses index faster than those inside the parentheses, and the rightmost nested variables index faster than the leftmost variables (Table 20.7).

Table 20.7 Example of Nested Effects

Data		I	A		B(A)					
A	B	β_0	A1	A2	B1A1	B2A1	B3A1	B1A2	B2A2	B3A2
1	1	1	1	0	1	0	0	0	0	0
1	2	1	1	0	0	1	0	0	0	0
1	3	1	1	0	0	0	1	0	0	0
2	1	1	0	1	0	0	0	1	0	0
2	2	1	0	1	0	0	0	0	1	0
2	3	1	0	1	0	0	0	0	0	1

Continuous-Nesting-Class Effects

When a continuous variable nests or crosses with a classification variable, the design columns are constructed by multiplying the continuous values into the design columns for the classification effect (Table 20.8).

Table 20.8 Example of Continuous-Nesting-Class Effects

Data		I	A		X(A)	
X	A	β_0	A1	A2	X(A1)	X(A2)
21	1	1	1	0	21	0
24	1	1	1	0	24	0
22	1	1	1	0	22	0
28	2	1	0	1	0	28
19	2	1	0	1	0	19
23	2	1	0	1	0	23

This model estimates a separate intercept and a separate slope for X within each level of A.

Continuous-by-Class Effects

Continuous-by-class effects generate the same design columns as continuous-nesting-class effects. Table 20.9 shows the construction of the X*A effect. The two columns for this effect are the same as the columns for the X(A) effect in Table 20.8.

Table 20.9 Example of Continuous-by-Class Effects

Data		**I**	**X**	A		X*A	
X	A	β_0	X	A1	A2	X*A1	X*A2
21	1	1	21	1	0	21	0
24	1	1	24	1	0	24	0
22	1	1	22	1	0	22	0
28	2	1	28	0	1	0	28
19	2	1	19	0	1	0	19
23	2	1	23	0	1	0	23

You can use continuous-by-class effects together with pure continuous effects to test for homogeneity of slopes.

General Effects

An example that combines all the effects is X1*X2*A*B*C(D E). The continuous list comes first, followed by the crossed list, followed by the nested list in parentheses. You should be aware of the sequencing of parameters when you use statements that depend on the ordering of parameters. Such statements include CONTRAST and ESTIMATE statements, which are used in a number of procedures to estimate and test functions of the parameters.

Effects might be renamed by the procedure to correspond to ordering rules. For example, B*A(E D) might be renamed A*B(D E) to satisfy the following:

- Classification variables that occur outside parentheses (crossed effects) are sorted in the order in which they appear in the CLASS statement.
- Variables within parentheses (nested effects) are sorted in the order in which they appear in the CLASS statement.

The sequencing of the parameters generated by an effect can be described by which variables have their levels indexed faster:

- Variables in the crossed list index faster than variables in the nested list.
- Within a crossed or nested list, variables to the right index faster than variables to the left.

For example, suppose a model includes four effects—A, B, C, and D—each having two levels, 1 and 2. If the CLASS statement is

```
class A B C D;
```

then the order of the parameters for the effect B*A(C D), which is renamed A*B(C D), is

$$\begin{array}{llll} A_1B_1C_1D_1 \rightarrow & A_1B_2C_1D_1 \rightarrow & A_2B_1C_1D_1 \rightarrow & A_2B_2C_1D_1 \rightarrow \\ A_1B_1C_1D_2 \rightarrow & A_1B_2C_1D_2 \rightarrow & A_2B_1C_1D_2 \rightarrow & A_2B_2C_1D_2 \rightarrow \\ A_1B_1C_2D_1 \rightarrow & A_1B_2C_2D_1 \rightarrow & A_2B_1C_2D_1 \rightarrow & A_2B_2C_2D_1 \rightarrow \\ A_1B_1C_2D_2 \rightarrow & A_1B_2C_2D_2 \rightarrow & A_2B_1C_2D_2 \rightarrow & A_2B_2C_2D_2 \end{array}$$

Note that first the crossed effects B and A are sorted in the order in which they appear in the CLASS statement so that A precedes B in the parameter list. Then, for each combination of the nested effects in turn, combinations of A and B appear. The B effect changes fastest because it is rightmost in the cross list. Then A changes next fastest, and D changes next fastest. The C effect changes most slowly because it is leftmost in the nested list.

Other Parameterizations

This section applies to the following procedures:
CATMOD, GENMOD, GLMSELECT, LOGISTIC, PHREG, and SURVEYPHREG.

Some SAS/STAT procedures, including GENMOD, GLMSELECT, and LOGISTIC, support nonsingular parameterizations for classification effects. A variety of these nonsingular parameterizations are available. In most of these procedures you use the PARAM= option in the CLASS statement to specify the parameterization.

Consider a model with one CLASS variable A that has four levels, 1, 2, 5, and 7. Details of the possible choices for the PARAM= option follow.

EFFECT

Three columns are created to indicate group membership of the nonreference levels. For the reference level, all three dummy variables have a value of –1. For example, if the reference level is 7 (REF=7), the design matrix columns for A are as follows.

Effect Coding			
	Design Matrix		
A	**A1**	**A2**	**A5**
1	1	0	0
2	0	1	0
5	0	0	1
7	–1	–1	–1

Parameter estimates of CLASS main effects that use the effect coding scheme estimate the difference in the effect of each nonreference level compared to the average effect over all four levels.

The EFFECT parameterization is the default parameterization in the CATMOD procedure. See the section "Generation of the Design Matrix" on page 2284, in Chapter 36, "The CATMOD Procedure," for further details about parameterization of model effects with the CATMOD procedure.

GLM As in the GLM procedure, four columns are created to indicate group membership. The design matrix columns for A are as follows.

GLM Coding				
	Design Matrix			
A	**A1**	**A2**	**A5**	**A7**
1	1	0	0	0
2	0	1	0	0
5	0	0	1	0
7	0	0	0	1

Parameter estimates of CLASS main effects that use the GLM coding scheme estimate the difference in the effects of each level compared to the last level. See the previous section for details about the GLM parameterization of model effects.

ORDINAL | THERMOMETER Three columns are created to indicate group membership of the higher levels of the effect. For the first level of the effect (which for A is 1), all three dummy variables have a value of 0. The design matrix columns for A are as follows.

Ordinal Coding			
	Design Matrix		
A	**A2**	**A5**	**A7**
1	0	0	0
2	1	0	0
5	1	1	0
7	1	1	1

The first level of the effect is a control or baseline level. Parameter estimates of CLASS main effects, using the ORDINAL coding scheme, estimate the differences between effects of successive levels. When the parameters have the same sign, the effect is monotonic across the levels.

POLYNOMIAL | POLY Three columns are created. The first represents the linear term (x), the second represents the quadratic term $\left(x^2\right)$, and the third represents the cubic term $\left(x^3\right)$, where x is the level value. If the CLASS levels are not numeric, they are translated into 1, 2, 3, ... according to their sort order. The design matrix columns for A are as follows.

Polynomial Coding			
	Design Matrix		
A	APOLY1	APOLY2	APOLY3
1	1	1	1
2	2	4	8
5	5	25	125
7	7	49	343

REFERENCE | REF Three columns are created to indicate group membership of the nonreference levels. For the reference level, all three dummy variables have a value of 0. For example, if the reference level is 7 (REF=7), the design matrix columns for A are as follows.

Reference Coding			
	Design Matrix		
A	A1	A2	A5
1	1	0	0
2	0	1	0
5	0	0	1
7	0	0	0

Parameter estimates of CLASS main effects that use the reference coding scheme estimate the difference in the effect of each nonreference level compared to the effect of the reference level.

The REFERENCE parameterization is also available through the MODEL statement in the CATMOD procedure. See the section "Generation of the Design Matrix" on page 2284, in Chapter 36, "The CATMOD Procedure," for further details about parameterization of model effects with the CATMOD procedure.

ORTHEFFECT The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=EFFECT. The design matrix columns for A are as follows.

Orthogonal Effect Coding			
	Design Matrix		
A	AOEFF1	AOEFF2	AOEFF3
1	1.41421	–0.81650	–0.57735
2	0	1.63299	–0.57735
5	0	0	1.73205
7	–1.41421	–0.81649	–0.57735

ORTHORDINAL | ORTHOTHERM The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=ORDINAL. The design matrix columns for A are as follows.

Orthogonal Ordinal Coding			
	Design Matrix		
A	**AOORD1**	**AOORD2**	**AOORD3**
1	–1.73205	0	0
2	0.57735	–1.63299	0
5	0.57735	0.81650	–1.41421
7	0.57735	0.81650	1.41421

ORTHPOLY The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=POLY. The design matrix columns for A are as follows.

Orthogonal Polynomial Coding			
	Design Matrix		
A	**AOPOLY1**	**AOPOLY2**	**AOPOLY5**
1	–1.15311	0.90712	–0.92058
2	–0.73380	–0.54041	1.47292
5	0.52414	–1.37034	–0.92058
7	1.36277	1.00363	0.36823

ORTHREF The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=REFERENCE. The design matrix columns for A are as follows.

Orthogonal Reference Coding			
	Design Matrix		
A	**AOREF1**	**AOREF2**	**AOREF3**
1	1.73205	0	0
2	–0.57735	1.63299	0
5	–0.57735	–0.81650	1.41421
7	–0.57735	–0.81650	–1.41421

Using CAS Tables

SAS/STAT procedures can be used on both SAS data sets and SAS Cloud Analytic Services (CAS) tables. In general, you can input a CAS table when option syntax calls for a *SAS-data-set*. The most common usage is that of the DATA=*SAS-data-set* option, in which you can specify either a SAS data set or a CAS table. In general, when the order of observations does not play a role in a statistical analysis, the results from inputting a CAS table match what you would get from the same data stored as a SAS data set.

However, CAS tables and SAS data sets are not always interchangeable. Traditionally, SAS/STAT procedures have been developed for use with SAS data sets, which can be thought of as a collection of observations arranged as ordered rows of data. But a CAS table has no intrinsic order of observations. When you use a CAS table as an input data set, the order of the observations as received by a SAS/STAT procedure is dynamically determined by various computational factors and can be different each time those data are retrieved.

Broadly speaking, the lack of intrinsic order of CAS tables can affect all statistical operations that use finite-precision arithmetic. Although most of the time such a numerical precision issue is minor and unnoticeable, there could be statistical methods and algorithms that cannot produce the intended or even correct statistical results unless the observations are ordered in a particular way or the data-ordering information is provided.

Table 20.10 summarizes various known issues in using CAS tables as input for SAS/STAT procedures. These issues are described in more detail in the next few sections, which are followed by an important section that recommends best practices for using CAS tables with SAS/STAT procedures.

Table 20.10 Primary Issues in Using CAS Tables with SAS/STAT Procedures

Issue	Procedure's Behavior
A procedure is incompatible with input CAS tables	Exits with an error message
An option is incompatible with input CAS tables	Exits with an error message or ignores the option and issues a warning message
A special data set is incompatible with CAS tables	Exits with an error message
Special requirements for processing a CAS table properly are not satisfied	Exits with an error message
Some statistical results are not reliably reproducible using input CAS tables	Issues a note to the log or suppresses the affected results
Observations might not be identified reliably with input CAS tables	Issues a note or warning to the log

Procedures Incompatible with Input CAS Tables

Because they rely on the order of observations to produce correct results from the specialized computational or statistical methods, the following procedures do not support CAS table input: BCHOICE, BOXPLOT, CALIS, INBREED, KRIGE2D, MCMC, MDS, NESTED, NLMIXED, SIM2D, SPP, and VARIOGRAM. For these procedures, you must instead input SAS data sets that have the required or intended order of observations.

Options Incompatible with Input CAS Tables

Some procedure options assume a definite data order for a proper execution of the intended statistical analysis. For example, the ORDER=DATA option is not compatible with input CAS tables because the order of observations in a CAS table is not fixed. Attempting to use such an incompatible option would result in an error. To avoid the error, you must input a SAS data set instead.

For more information about incompatible options in various procedures, see the section "Options Incompatible with Input CAS Tables" on page 11138 of Chapter C, "Known Issues in Using CAS Tables with SAS/STAT Procedures."

Special Data Sets Incompatible with CAS Tables

To process the data correctly, some procedures require that an input (or output) data set be stored as a special data set (rather than as a raw data set). Such a special data set might rely on the order of observations to preserve some implicit information about the data. For example, when reading a TYPE=DISTANCE data set, PROC CLUSTER assumes that the observations (rows) in the data set are ordered in the same way as the objects that are represented as variables (columns). However, because a definite order of the observations in CAS tables cannot be assumed, PROC CLUSTER issues an error when you attempt to use a CAS table as the DATA= data set that is a TYPE=DISTANCE data set. In this case, you must instead input a TYPE=DISTANCE SAS data set that has the required or intended order of observations.

For more information about incompatible special data sets, see the section "Special Data Sets Incompatible with CAS Tables" on page 11139 of Chapter C, "Known Issues in Using CAS Tables with SAS/STAT Procedures."

Special Requirements for Processing Some CAS Tables

To process an input CAS table properly or to interpret the output results correctly, special requirements are imposed in some procedures. Failing to satisfy these requirements results in an error. However, the requirements are not imposed when you use a SAS data set, for which a procedure can assume a persistent order of observations.

For example, for correct processing of the prior information, the GENMOD procedure requires the specification of the _NAME_ variable when you use a CAS table as the INPUT= data set for the normal priors in the BAYES statement. Alternatively, you can use a SAS data set that has the required or intended order of observations for the INPUT= data set.

For more information about special requirements when you use CAS tables in SAS/STAT procedures, see the section "Special Requirements for Processing Some CAS Tables" on page 11140 of Chapter C, "Known Issues in Using CAS Tables with SAS/STAT Procedures."

Statistical Results Not Reliably Reproducible with Input CAS Tables

Inputting data from CAS tables does not guarantee a definite order of observations. When a procedure analyzes CAS data in real time, the order of observations that are loaded into memory might change each time you input the same CAS table. As a result, if a statistical analysis method depends on the order of the observations, an analysis that uses the same CAS table repeatedly might not reproduce exactly the same results. Hence, a nonreproducibility issue arises.

In some situations, the nonreproducibility issue is reflected by (mostly) minor numerical differences in the results from different runs, although all results are still considered to be statistically valid. Simulation-based methods, such as the bootstrap, are a typical example here. Random samples that are drawn by using simulation-based methods depend on the order of the observations in the input data set. Hence, numerical results based on analyzing these random samples are not reliably reproducible with input CAS tables. It is important to note that the nonreproducibility issue does not go away even if you specify a starting seed in a procedure's SEED= option (whenever available).

Other situations where the nonreproducibility issue arises can be reflected by different qualitative solutions for repeated runs of an analysis with the same CAS table. For example, clustering observations are usually based on comparing distances among objects and clusters in a sequential manner. If there are ties in distances between objects or clusters in the analysis, the order of observations might then affect the results of the tie-breaking algorithm for clustering objects or clusters. Hence, the final clustering results might not be reliably reproducible with input CAS tables.

Whether nonreproducibility poses severe limitations for the use of CAS tables should be judged case by case. Depending on the purpose of a particular statistical analysis, the degree of the nonreproducibility, the trustworthiness of the statistical method, and even your tolerance of the stochastic nature of statistical results, you might or might not view nonreproducibility as a critical problem of statistical analyses or inferences. Nonetheless, SAS/STAT procedures issue a note to the log about potential nonreproducibility when you use CAS tables as input data.

For more information about the nonreproducibility issue with the use of CAS tables, see the section "Statistical Results Not Reliably Reproducible with Input CAS Tables" on page 11141 of Chapter C, "Known Issues in Using CAS Tables with SAS/STAT Procedures."

Observations Might Not Be Identified Reliably with Input CAS Tables

Because you cannot expect the order of observations to be persistent or definite when a SAS/STAT procedure reads data from a CAS table, the observation or case numbers, which reflect the order of observations, do not have a definite mapping to the observations in the CAS table. Thus, issues arise when a SAS/STAT procedure analyzes CAS data input and uses observation or case numbers to identify or label observations in the output results. For example, identifying outlying or influential observations is an important task in regression diagnostics. With a CAS table as the input data set, the observation numbers in the output might not be used to identify the observations correctly.

In general, using observation numbers to identify observations is neither a good nor a reliable practice. Using a CAS table as the input data set accentuates the identification problem. SAS/STAT procedures issue a note to the log to alert you to the identification problem when you use a CAS table as the input data set.

An effective strategy to address this problem is to specify identification variables in the ID statement, which many SAS/STAT procedures support. The output results might then label the observations by the values of the identification variables.

For more information about issues related to using observation numbers in output results of SAS/STAT procedures, see the section "Observations Might Not Be Identified Reliably" on page 11142 of Chapter C, "Known Issues in Using CAS Tables with SAS/STAT Procedures."

Best Practices for Using CAS Tables with SAS/STAT Procedures

You can avoid some potential problems that are related to using CAS tables with SAS/STAT procedures by following these best practices:

Specify the necessary TYPE= option for a CAS table that is of a special data type

Unlike SAS data sets, CAS tables do not store the data type information, which is critical for procedures to properly process input data sets of special data types. See Chapter A, "Special SAS Data Sets," for more information about special data sets.

You must use the TYPE= option to specify the data type of an input CAS table when it is not an ordinary raw data set. For example, DATA=myCAS.covmat(TYPE=CORR) specifies that the CAS table named covmat contains correlations as data. Other examples of special data types that you must specify explicitly for input CAS tables include TYPE=ACE, COV, DISTANCE, EST, LINEAR, TREE, and so on.

Specify ID variables for the identification of observations

Some SAS/STAT procedures display numerical results, graphical plots, or output data sets that contain observation numbers. For example, you might use the observation numbers in a regression diagnostic plot to identify extreme observations. Or you might use the observation numbers in an output data set for clustering results to identify observations in clusters.

As discussed in the section "Observations Might Not Be Identified Reliably with Input CAS Tables" on page 423, using observation numbers to identify observations is neither a good nor a reliable practice, especially when you use a CAS table as the input data set. Specifying ID variables in the ID statement (if supported) enables a procedure to use the ID variable values to identify or label the observations correctly.

Avoid saving output data sets of special data types as CAS tables

Often, the OUT= or OUTSTAT= data sets that many SAS/STAT procedures create are of special data types, such as TYPE=CORR, COV, DISTANCE, EST, LINEAR, TREE, and so on. See Chapter A, "Special SAS Data Sets," for more information about special data sets.

The order of observations in these special data sets might be critical for clear interpretations and proper subsequent analyses. Saving these special output data sets to CAS tables, especially when internal variables such as _NAME_ and _TYPE_ are not created by a procedure, might not retain the intended data order persistently and therefore is not recommended. For example, you might not be able to visualize a symmetric correlation matrix when you print an output CAS table that contain elements of a correlation matrix. Although many SAS/STAT procedures create special internal variables such as _NAME_ and _TYPE_ that you can use to recover the order of the data, in general you should avoid saving special output data sets as CAS tables.

Avoid saving output data sets as CAS tables if the order of observations is important for clear interpretation or proper subsequent analyses

No matter what the output data type is, the same issue of no persistent ordering of observations applies to any output results that you save to a CAS table. This also applies to any output data sets that you create from ODS tables by using the ODS OUTPUT statement. For example, some procedures produce output data sets that contain transformations of the original observations. If the output data set is written as a CAS table, you might not be able to map the transformed observations to the original observations reliably.

CODE Statement

This section applies to the following procedures:
GENMOD, GLIMMIX, GLM, GLMSELECT, HPGENSELECT, HPLOGISTIC, HPQUANTREG, HPREG, LOGISTIC, MIXED, PHREG, PLM, QUANTSELECT, and REG.

The CODE statement enables you to write SAS DATA step code to a file or catalog entry for computing predicted values of the fitted model. This code can then be included in a DATA step to score new data. For example, in the following program, the CODE statement writes the code for predicting the outcome of a logistic regression model to the file *mycode.sas*. The file is subsequently included in a DATA step to score the sashelp.Bmt data.

```
proc logistic data=sashelp.Bmt;
   class Group;
   model Status=Group;
   code file='mycode.sas';
run;

data Score;
   set sashelp.Bmt;
   %include mycode;
run;
```

Syntax: CODE Statement

CODE < *options* > **;**

Table 20.11 summarizes the *options* you can specify in the CODE statement.

Table 20.11 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

You cannot specify both the FILE= and CATALOG= options. If you specify neither, the SAS scoring code is written to the SAS log. You can specify the following *options* in the CODE statement.

CATALOG=*library.catalog.entry.type*

CAT=*library.catalog.entry.type*

specifies where to write the generated code in the form of library.catalog.entry.type. The compound name can have from one to four levels. The default library is determined by the USER= SAS system option, which by default is WORK. The default *entry* is SASCODE, and the default *type* is SOURCE.

DUMMIES | NODUMMIES

specifies whether to keep dummy variables that represent the CLASS levels in the data set. The default is NODUMMIES, which specifies that dummy variables not be retained.

ERROR | NOERROR

specifies whether to generate code to compute the error function. The default is NOERROR, which specifies that the error function not be generated.

FILE=*filename*

names the external file that saves the generated code. When enclosed in a quoted string (for example, FILE="c:\mydir\scorecode.sas"), this option specifies the path for writing the code to an external file. You can also specify unquoted SAS file names of no more than eight characters for *filename*. If the *filename* is assigned as a *fileref* in a Base SAS FILENAME statement, the file specified in the FILENAME statement is opened. The special *filerefs* LOG and PRINT are always assigned. If the specified *filename* is not an assigned *fileref*, the specified value for *filename* is concatenated with a *.txt* extension before the file is opened. For example, if FOO is not an assigned *fileref*, FILE=FOO causes *FOO.txt* to be opened. If *filename* has more than eight characters, an error message is printed.

FORMAT=*format*

specifies the format for the regression coefficients and other numerical values that do not have a format from the input data set. The default *format* is BEST20.

GROUP=*group-name*

specifies the group identifier for group processing. The *group-name* should be a valid SAS name of no more than 16 characters. It is used to construct array names and statement labels in the generated code.

IMPUTE

imputes the predicted values according to an intercept-only model for observations with missing or invalid covariate values. For a continuous response, the predicted value is the mean of the response variable; for a categorical response, the predicted values are the proportions of the response categories. When the IMPUTE option is specified, the scoring code also creates a variable named _WARN_ that contains one or more single-character codes that indicate problems in computing predicted values. The character codes used in _WARN_ go in the following positions:

Table 20.12 _WARN_ Variable Codes

Code	Column	Meaning
M	1	Missing covariate value
U	2	Unrecognized covariate category

LINESIZE=*value*

LS=*value*

specifies the line size for the generated code. The default is 78. The permissible range is 78 to 254.

LOOKUP=*lookup-method*

specifies the algorithm for looking up CLASS levels. You can specify the following *lookup-methods*:

AUTO

selects the LINEAR algorithm if a CLASS variable has fewer than five categories; otherwise, the BINARY algorithm is used. This is the default.

BINARY

uses a binary search. This method is fast, but might produce incorrect results and the normalized category values might contain characters that collate in different orders in ASCII and EBCDIC, if you generate the code on an ASCII machine and execute the code on an EBCDIC machine or vice versa.

LINEAR

uses a linear search with IF statements that have categories in the order of the class levels. This method is slow if there are many categories.

SELECT

uses a SELECT statement.

The default is LOOKUP=AUTO.

RESIDUAL | NORESIDUAL

specifies whether to generate code to compute residual values. If you request code for residuals and then score a data set that does not contain target values, the residuals will have missing values. The default is NORESIDUAL, which specifies that the code for residuals not be generated.

EFFECT Statement

This section applies to the following procedures:
GLIMMIX, GLMSELECT, HPMIXED, LOGISTIC, ORTHOREG, PHREG, PLS, QUANTLIFE, QUANTREG, QUANTSELECT, ROBUSTREG, SURVEYLOGISTIC, and SURVEYREG.

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412. For example, the terms A, B, x, A*x, A*B, and sub in the following statements define fixed, random, and subject effects of the usual type in a mixed model:

```
proc glimmix;
   class A B sub;
   model y = A B x A*x;
   random A*B / subject=sub;
run;
```

A constructed effect, on the other hand, is assigned through the EFFECT statement. For example, in the following program, the EFFECT statement defines a constructed effect named spl:

```
proc glimmix;
   class A B SUB;
   effect spl = spline(x);
   model y = A B A*spl;
   random A*B / subject=sub;
run;
```

The columns of spl are formed from the data set variable x as a cubic B-spline basis with three equally spaced interior knots.

Each constructed effect corresponds to a collection of columns that are referred to by using the name you supply. You can specify multiple EFFECT statements, and all EFFECT statements must precede the MODEL statement.

The general syntax for the EFFECT statement with *effect-specification* is

EFFECT *effect-name* = *effect-type* (*var-list* < / *effect-options* >) **;**

The name of the effect is specified after the EFFECT keyword. This name can appear in only one EFFECT statement and cannot be the name of a variable in the input data set. The *effect-type* is specified after an equal sign, followed by a list of variables within parentheses which are used in constructing the effect.

Effect-options that are specific to an *effect-type* can be specified after a slash (/) following the variable list. The following *effect-types* are available and are discussed in the following sections:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 20.13 summarizes the *options* available in the EFFECT statement.

Table 20.13 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial

Table 20.13 *continued*

Option	Description
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

Collection Effects

EFFECT *name*=**COLLECTION (***var-list* < / **DETAILS** >**) ;**

You use a collection effect to define a set of variables that are treated as a single effect with multiple degrees of freedom. The variables in *var-list* can be continuous or classification variables. The columns in the design matrix that are contributed by a collection effect are the design columns of its constituent variables in the order in which they appear in the definition of the collection effect. If you specify the DETAILS option, then a table that shows the constituents of the collection effect is displayed.

Lag Effects

EFFECT *name*=**LAG (***variable* / **WITHIN=***variable* **PERIOD=***variable lag-options***) ;**

EFFECT *name*=**LAG (***variable* / **WITHIN=(***var-list***) PERIOD=***variable lag-options***) ;**

A lag effect is a classification effect for the *variable* that is specified after the keyword LAG. A lag effect represents the effect of a previous value of the lagged variable when the observations of this variable are inherently ordered. A typical example where lag effects are useful is a study in which different subjects are given sequences of treatments and you want to investigate whether the treatment in the previous period is important in understanding the outcome in the current period. You can do this by including a lagged treatment effect in your model.

The precise definition of a lag effect depends on a subdivision of the data into disjoint subsets and an ordering into units of the observations within a subset. The subsets are often called subjects, and they are specified in the required WITHIN= option. The units are often called periods, and they are specified in the required PERIOD= option. For an observation that belongs to a particular subject at a particular period, the design matrix columns of the lagged variable are the usual design matrix columns of that variable except for the observation at the preceding period for that subject. Observations at the initial period do not have a preceding value, and so the design matrix columns of the lag effect for these observations are set to 0. You can also define lag effects where the number of periods that are lagged is greater than 1. If the number of periods that are lagged is n, then the design matrix columns of observations in periods less than or equal to n are set to 0. The design matrix columns that correspond to a subject at period p, where $p > n$, are the usual design matrix columns of the lagged variable for that subject at period $p - n$.

In a valid lag design there is at most one observation for a particular period and subject. For example, the following set of treatments by subject and period form a valid lag design:

Subject	Period	Treatment
Sheila	1	B
Joey	1	A
Athena	1	A
Gelindo	1	A
Sheila	2	C
Joey	2	A
Athena	2	.
Gelindo	2	B
Sheila	3	B
Joey	3	C
Athena	3	A
Gelindo	3	B

A convenient way to represent the organization of observations into subjects and periods is to form the lag design matrix. The rows and columns of this matrix correspond to the subjects and periods, respectively. The lag design matrix entry is the treatment for the corresponding subject and period.

The associated lag design matrix is

	Period		
Subject	1	2	3
Athena	A		A
Gelindo	A	B	B
Joey	A	A	C
Sheila	B	C	B

Note that the subject Athena did not receive a treatment at period 2, and so the corresponding entry in the lag design matrix is missing. The following statements define a lag effect for this lag design:

```
CLASS treatment;
EFFECT Lag = LAG( treatment / WITHIN=subject PERIOD=period);
```

When GLM coding is used for the variable treatment, the design matrix columns Lag_A, Lag_B, and Lag_C for the constructed effect Lag are as follows:

Subject	Period	Treatment	Lag_A	Lag_B	Lag_C
Athena	1	A	0	0	0
Athena	2		1	0	0
Athena	3	A	.	.	.
Gelindo	1	A	0	0	0
Gelindo	2	B	1	0	0
Gelindo	3	B	0	1	0
Joey	1	A	0	0	0
Joey	2	A	1	0	0
Joey	3	C	1	0	0
Sheila	1	B	0	0	0
Sheila	2	C	0	1	0
Sheila	3	B	0	0	1

The design matrix columns for each subject at period 1 are all 0 because there are no lagged observations for period 1. You can also see that the design matrix columns at period 3 for subject Athena are missing because Athena did not receive a treatment at period 2. Nevertheless, the design matrix columns for Athena at period 2 are nonmissing and correspond to the treatment "A" that she received in period 1.

You must specify the following required options:

PERIOD=*variable*
specifies the period variable of the lag design. The number of periods is the number of unique formatted values of the *variable*, and the ordering of the period is formed by sorting these formatted values in ascending order. You must specify this option.

WITHIN=(*var-list***) | *variable***
specifies a variable (or a list of variables within parentheses) that defines the subject grouping of the lag design. If there is only one WITHIN= *variable*, then the parentheses are not required. Each subject is defined by the unique set of formatted values of the variables in the WITHIN= list. The subjects are sorted in ascending lexicographic order. You must specify this option.

You can also specify the following optional *lag-options*:

DESIGNROLE=*variable*
specifies a numeric variable that subsets observations into two groups: a group in which the value of *variable* is nonzero and a group in which the value of *variable* is zero. The observations in the first group are used to form the lag design matrix that is used in fitting the model. The lag design that corresponds to the second group is used when observations in the input data set that do not belong to the first group are scored. This option is useful when you want to obtain predicted values in an output data set for observations that are not used in fitting the model. If you do not specify this option, then all observations are assigned to the first group.

DETAILS
requests a table that shows the lag design matrix of the lag effect.

NLAG=*n*
specifies the number of lags. By default NLAG=1.

Multimember Effects

EFFECT *name*=**MULTIMEMBER (***var-list* < / *mm-options* >**) ;**

EFFECT *name*=**MM (***var-list* < / *mm-options* >**) ;**

A multimember effect is formed from one or more classification variables in such a way that each observation can be associated with one or more levels of the union of the levels of the classification variables. In other words, a multimember effect is a classification-type effect with possibly more than one nonzero column entry for each observation. Multimember effects are useful, for example, in modeling the following:

- nurses' effects on patient recovery in hospitals
- teachers' effects on student scores
- lineage effects in genetic studies. See Example 52.16 in Chapter 52, "The GLIMMIX Procedure," for an application with random multimember effects in a genetic diallel experiment.

The levels of a multimember effect consist of the union of formatted values of the variables that define this effect. Each such level contributes one column to the design matrix. For each observation, the value that corresponds to each level of the multimember effect in the design matrix is the number of times that this level occurs for the observation.

For example, the following data provide teacher information and end-of-year test scores for students after two semesters:

Student	Score	Teacher1	Teacher2
Mary	87	Tobias	Cohen
Tom	89	Rodriguez	Tobias
Fred	82	Cohen	Cohen
Jane	88	Tobias	.
Jack	99	.	.

For example, Mary had different teachers in the two semesters, Fred had the same teacher in both semesters, and Jane received instruction only in the first semester.

You can model the effect of the teachers on student performance by using a multimember effect specified as follows:

```
CLASS teacher1 teacher2;
EFFECT teacher = MM(teacher1 teacher2);
```

The levels of the teacher effect are Cohen, Rodriguez, and Tobias, and the associated design matrix columns are as follows:

Student	Cohen	Rodriguez	Tobias
Mary	1	0	1
Tom	0	1	1
Fred	2	0	0
Jane	0	0	1
Jack	.	.	.

You can specify the following *mm-options* after a slash (/):

DETAILS
requests a table that shows the levels of the multimember effect.

NOEFFECT
specifies that, for observations with all missing levels of the multimember variables, the values in the corresponding design matrix columns be set to zero. If, in the preceding example, the teacher effect is defined by

```
EFFECT teacher = MM(teacher1 teacher2 / noeffect);
```

then the associated design matrix columns' values for Jack are all zero. This enables you to include Jack in the analysis even though there is no effect of teachers on his performance.

A situation where it is important to designate observations as having no effect due to a classification variable is the analysis of crossover designs, where lagged treatment levels are used to model the carryover effects of treatments between periods. Since there is no carryover effect for the first period, the treatment lag effect in a crossover design can be modeled with a multimember effect that consists of a single classification variable and the NOEFFECT option, as in the following statements:

```
CLASS Treatment lagTreatment;
EFFECT Carryover = MM(lagTreatment / noeffect);
```

The lagTreatment variable contains a missing value for the first period. Otherwise, it contains the value of the treatment variable for the preceding period.

STDIZE
specifies that for each observation, the entries in the design matrix that corresponds to the multimember effect be scaled to have a sum of one.

WEIGHT=*wght-list*
specifies numeric variables used to weigh the contributions of each of the classification effects that define the constructed multimember effect. The number of variables in *wght-list* must match the number of classification variables that define the effect.

Polynomial Effects

EFFECT *name*=**POLYNOMIAL (***var-list* < / *polynomial-options* >**) ;**

EFFECT *name*=**POLY (***var-list* < / *polynomial-options* >**) ;**

The variables in *var-list* must be numeric. A design matrix column is generated for each term of the specified polynomial. By default, each of these terms is treated as a separate effect for the purpose of model building. For example, the statements

```
proc glmselect;
   effect MyPoly = polynomial(x1-x3/degree=2);
   model y = MyPoly;
run;
```

yield the identical analysis to the statements

```
proc glmselect;
   model y = x1 x2 x3 x1*x1 x1*x2 x1*x3 x2*x2 x2*x3 x3*x3;
run;
```

You can specify the following *polynomial-options* after a slash (/):

DEGREE=*n*

specifies the degree of the polynomial. The degree must be a positive integer. The degree is typically a small integer, such as 1, 2, or 3. The default is DEGREE=1.

DETAILS

requests a table that shows the details of the specified polynomial, including the number of terms generated. If you also specify the STANDARDIZE option, then a table that shows the standardization details is also produced.

LABELSTYLE=(*style-opts***)**

LABELSTYLE=*style-opt*

specifies how the terms in the polynomial are labeled. By default, powers are shown with ˆ as the exponentiation operator and * as the multiplication operator. For example, a polynomial term such as $x_1^3 x_2 x_3^2$ is labeled x1ˆ3*x2*x3ˆ2. You can change the style of the label by using the following *style-opts* within parentheses. If you specify a single *style-opt*, then you can omit the enclosing parentheses.

EXPAND

specifies that each variable with an exponent greater than 1 be written as products of that variable. For example, the term $x_1^3 x_2 x_3^2$ receives the label x1*x1*x1*x2*x3*x3.

EXPONENT < =*quoted string* >

specifies that each variable with an exponent greater than 1 be written using exponential notation. By default, the symbol ˆ is used as the exponentiation operator. If you supply the optional quoted string after an equal sign, then that string is used as the exponentiation operator. For example, if you specify

```
LABELSTYLE=(EXPONENT="**")
```

then the term $x_1^3 x_2 x_3^2$ receives the label x1**3*x2*x3**2.

INCLUDENAME

specifies that the name of the effect followed by an underscore be used as a prefix for term labels. For example, the following statement generates terms with labels MyPoly_x1 and MyPoly_x1^2:

```
EFFECT MyPoly=POLYNOMIAL(x1/degree=2 labelstyle=INCLUDENAME)
```

The INCLUDENAME option is ignored if you also specify the NOSEPARATE option in the EFFECT=POLYNOMIAL statement.

PRODUCTSYMBOL=NONE | *quoted string*

specifies that the supplied string be used as the product symbol. For example, the following statement generates terms with labels x1, x2, and x1 x2:

```
EFFECT MyPoly=POLYNOMIAL(x1 x2 / degree=2 mdegree=1
                                 labelstyle=(PRODUCTSYMBOL=" "))
```

If you specify PRODUCTSYMBOL=NONE, then the labels are formed by juxtaposing the constituent variable names.

MDEGREE=*n*

specifies the maximum degree of any variable in a term of the polynomial. This degree must be a positive integer. The default is the degree of the specified polynomial. For example, the following statement generates the terms x_1, x_2, x_1^2, x_1x_2, x_2^2, $x_1^2x_2$, $x_1x_2^2$ and $x_1^2x_2^2$:

```
EFFECT MyPoly=POLYNOMIAL(x1 x2/degree=4 MDEGREE=2);
```

NOSEPARATE

specifies that the polynomial be treated as a single effect with multiple degrees of freedom. The effect name that you specify is used as the constructed effect name, and the labels of the terms are used as labels of the corresponding parameters.

STANDARDIZE < (*centerscale-opts*) > < = *standardize-opt* >

specifies that the variables that define the polynomial be standardized. By default, the standardized variables receive prefix "s_" in the variable names.

You can use the following *centerscale-opts* to specify how the center and scale are estimated:

METHOD=MOMENTS

specifies that the center be estimated by the variable mean and the scale be estimated by the standard deviation. If a weight variable is specified using a WEIGHT statement, the observations with invalid weights are ignored when forming the mean and standard deviation, but the weights are otherwise not used. Only observations that are used in performing the analysis are used for the standardization.

METHOD=RANGE

specifies that the center be estimated by the midpoint of the variable range and the scale be estimated as half the variable range. Any observation that has a missing value for any regressor used in the model is ignored when computing the range of variables in a polynomial effect. Observations with valid regressor values but missing or invalid values of frequency variables, weight variables, or dependent variables are used in computing variable ranges. The default (if you do not specify the METHOD= suboption) is METHOD=RANGE.

METHOD=WMOMENTS

is the same as METHOD=MOMENTS except that weighted means and weighted standard deviations are used.

Let

$$
\begin{aligned}
n &= \text{number of observations used in the analysis} \\
w &= \text{weight variable} \\
f &= \text{frequency variable} \\
x &= \text{variable to be standardized} \\
x_{(n)} &= \mathrm{Max}_{i=1}^{n}(x_i) \\
x_{(1)} &= \mathrm{Min}_{i=1}^{n}(x_i) \\
F &= \text{sum of frequencies} \\
&= \Sigma_{i=1}^{n} f_i \\
\mathrm{WF} &= \text{sum of weighted frequencies} \\
&= \Sigma_{i=1}^{n} w_i f_i
\end{aligned}
$$

Table 20.14 shows how the center and scale are computed for each of the supported methods.

Table 20.14 Center and Scale Estimates by Method

Method	Center	Scale
Range	$(x_{(n)} + x_{(1)})/2$	$(x_{(n)} - x_{(1)})/2$
Moments	$\bar{x} = \Sigma_{i=1}^{n} f_i x_i / F$	$\sqrt{\Sigma_{i=1}^{n} f_i (x_i - \bar{x})^2/(F-1)}$
WMoments	$\bar{x}_w = \Sigma_{i=1}^{n} w_i f_i x_i / \mathrm{WF}$	$\sqrt{\Sigma_{i=1}^{n} w_i f_i (x_i - \bar{x}_w)^2/(F-1)}$

PREFIX=NONE | *quoted-string*

specifies the prefix that is appended to standardized variables when forming the term labels. If you omit this option, the default prefix is "s_". If you specify PREFIX=NONE, then standardized variables are not prefixed.

You can control whether the standardization is to center, scale, or both center and scale by specifying a *standardize-opt*:

CENTER

specifies that variables be centered but not scaled. For a variable x,

$$\text{s_x} = x - \text{center}$$

CENTERSCALE

specifies that variables be centered and scaled. This is the default if you do not specify a *standardization-opt*. For a variable x,

$$\text{s_x} = \frac{x - \text{center}}{\text{scale}}$$

NONE

specifies that no standardization be performed.

SCALE

specifies that variables be scaled but not centered. For a variable x,

$$\text{s_x} = \frac{x}{\text{scale}}$$

Spline Effects

This section discusses the construction of spline effects through the EFFECT statement. You can also include spline effects in statistical models by other means. The TRANSREG procedure has dedicated facilities for including regression splines in your model and controlling the construction of the splines. For example, you can use the TRANSREG procedure to fit a spline function but restrict the function to be always increasing or decreasing (monotone). See the section "Using Splines and Knots" on page 10632 in Chapter 126, "The TRANSREG Procedure," for more information about using splines with the TRANSREG procedure. The GAM and TPSPLINE procedures also can model the effects of regressor variables in terms of smooth functions that are generated from spline bases. For more information see Chapter 48, "The GAM Procedure," and Chapter 125, "The TPSPLINE Procedure."

A spline effect expands variables into spline bases whose form depends on the options that you specify. You can find details about regression splines and spline bases in the section "Splines and Spline Bases" on page 441. You request a spline effect with the syntax

EFFECT *name*=**SPLINE (***var-list* < / *spline-options* >**) ;**

The variables in *var-list* must be numeric. Design matrix columns are generated separately for each of these variables, and the set of columns is collectively referred to with the specified name. By default, the spline basis that is generated for each variable is a cubic B-spline basis with three equally spaced knots positioned between the minimum and maximum values of that variable. This yields by default seven design matrix columns for each of the variables in the SPLINE effect.

You can specify the following *spline-options* after a slash (/):

BASIS=BSPLINE

specifies a B-spline basis for the spline expansion. For splines of degree d defined with n knots, this basis consists of $n + d + 1$ columns. In order to completely specify the B-spline basis, d left-side boundary knots and $\max\{d, 1\}$ right-side boundary knots are also required. See the suboptions KNOTMETHOD=, DATABOUNDARY, KNOTMIN=, and KNOTMAX= for details about how to specify the positions of both the internal and boundary knots. This is the default if you do not specify the BASIS= suboption.

BASIS=TPF(*options*)

specifies a truncated power function basis for the spline expansion. For splines of degree d defined with n knots for a variable x, this basis consists of an intercept, polynomials $x, x^2, \ldots, x^d$ and one truncated power function for each of the n knots. Unlike the B-spline basis, no boundary knots are required. See the suboption KNOTMETHOD= for details about how you can specify the position of the internal knots.

You can modify the number of columns when you request BASIS=TPF with the following *options*:

NOINT

excludes the intercept column.

NOPOWERS

excludes the intercept and polynomial columns.

DATABOUNDARY

specifies that the extremes of the data be used as boundary knots when building a B-spline basis.

DEGREE=*n*

specifies the degree of the spline transformation. The degree must be a nonnegative integer. The degree is typically a small integer, such as 0, 1, 2, or 3. The default is DEGREE=3.

DETAILS

requests tables that show the knot locations and the knots associated with each spline basis function.

KNOTMAX=*value*

specifies that, for each variable in the EFFECT statement, the right-side boundary knots be equally spaced starting at the maximum of the variable and ending at the specified value. This option is ignored for variables whose maximum value is greater than the specified value or if the DATABOUNDARY option is also specified.

KNOTMETHOD=*knot-method*<(*knot-options*)>

specifies how to construct the knots for spline effects. You can choose from the following *knot-methods* and affect the knot construction further with the method-specific *knot-options*:

EQUAL<(*n*)>

specifies that n equally spaced knots be positioned between the extremes of the data. The default is $n = 3$. For a B-spline basis, any needed boundary knots continue to be equally spaced unless the DATABOUNDARY option has also been specified. KNOTMETHOD=EQUAL is the default if no *knot-method* is specified.

LIST(*number-list***)**
specifies the list of internal knots to be used in forming the spline basis columns. For a B-spline basis, the data extremes are used as boundary knots.

LISTWITHBOUNDARY(*number-list***)**
specifies the list of all knots that are used in forming the spline basis columns. When you use a truncated power function basis, this list is interpreted as the list of internal knots. When you use a B-spline basis of degree d, then the first d entries are used as left-side boundary knots and the last $\mathrm{MAX}(d, 1)$ entries in the list are used as right-side boundary knots.

MULTISCALE< (*multiscale-options***) >**
specifies that multiple B-spline bases be generated, corresponding to sets with an increasing number of internal knots. As you increase the number of internal knots, the spline basis you generate is able to approximate features of the data at finer scales. So, by generating bases at multiple scales, you facilitate the modeling of both coarse- and fine-grained features of the data. For scale i, the spline basis corresponds to 2^i equally spaced internal knots. By default, the bases for scales 0–7 are generated. For each scale, a separate spline effect is generated. The name of the constructed spline effect at scale i is formed by appending _Si to the effect name that you specify in the EFFECT statement. If you specify multiple variables in the EFFECT statement, then spline bases are generated separately for each variable at each scale and the name of the corresponding effect is obtained by appending the variable name followed by _Si to the name in the EFFECT statement. For example, the following statement generates effects named spl_x1_S0, spl_x1_S1, spl_x1_S2, . . . , spl_x1_S7 and spl_x2_S1, spl_x2_S2, . . . , spl_x2_S7:

```
EFFECT spl = spline(x1 x2 / knotmethod=multiscale);
```

The MULTISCALE option is ignored if you specify the BASIS=TPF *spline-option*. The MULTISCALE option is not available for spline effects that are specified in the RANDOM statement of the GLIMMIX procedure.

You can control which scales are included with the following *multiscale-options*:

STARTSCALE=*n*
specifies the start scale, where *n* is a positive integer. The default is STARTSCALE=0.

ENDSCALE=*n*
specifies the end scale, where *n* is a positive integer. The default is ENDSCALE=7.

PERCENTILELIST(*percentile-list***)**
requests that internal knots for the variable or variables named in the EFFECT statement be placed at the specific percentiles that you list. For example, the following statement specifies five internal knots at the particular percentiles of the variable x that are suggested by Harrell (2001):

```
EFFECT spl = spline(x / knotmethod=percentilelist(5 27.5 50 72.5 95));
```

Alternatively, you can use standard numeric list syntax to specify only the endpoints and increment of your percentiles:

```
EFFECT spl = spline(x / knotmethod=percentilelist(5 to 95 by 22.5));
```

PERCENTILES(*n*)

requests that internal knots be placed at *n* equally spaced percentiles of the variable or variables named in the EFFECT statement. For example, the following statement positions internal knots at the deciles of the variable x. For a B-spline basis, the extremes of the data are used as boundary knots:

```
EFFECT spl = spline(x / knotmethod=percentiles(9));
```

RANGEFRACTIONS(*fraction-list*)

requests that internal knots be placed at each fraction of the ranges of the variables in the EFFECT statement. For example, if variable x1 ranges between 1 and 3, and variable x2 ranges between 0 and 20, then the following EFFECT statement uses internal knots 1.2, 2, and 2.5 for variable x1 and internal knots 2, 10, and 15 for variable x2:

```
EFFECT spl = spline(x1 x2 / knotmethod=rangefractions(.1 .5 .75));
```

For a B-spline basis, the data extremes are used as boundary knots.

KNOTMIN=*value*

specifies that for each variable in the EFFECT statement, the left-side boundary knots be equally spaced starting at the specified value and ending at the minimum of the variable. This option is ignored for variables whose minimum value is less than the specified value or if the DATABOUNDARY option is also specified.

NATURALCUBIC

specifies a natural cubic spline basis for the spline expansion. Natural cubic splines, also known as restricted cubic splines, are cubic splines that are constrained to be linear beyond the extreme knots. The natural cubic spline basis that is produced by the EFFECT statement is obtained by starting from the unrestricted truncated power function cubic spline basis that is defined with n distinct knots and imposes the linearity constraints beyond the extreme knots. This basis consists of an intercept, the polynomial x, and $n-2$ functions that are all linear beyond the largest knot. The ith function, $i = 1, 2, \ldots, n-2$, is zero to the left of the ith knot, which is called the "break knot." See the section "Splines and Spline Bases" on page 441 for details of this basis. You can use the NOINT and NOPOWERS suboptions of the BASIS=TPF option to suppress the intercept and polynomial x when forming the columns of the natural cubic spline basis. When you specify the NATURALCUBIC option, the options BASIS=BSPLINE, DATABOUNDARY, DEGREE=, and KNOTMETHOD=MULTISCALE are not applicable.

SEPARATE

specifies that when multiple variables are specified in the EFFECT statement, the spline basis for each variable be treated as a separate effect. The names of these separated effects are formed by appending an underscore followed by the name of the variable to the name that you specify in the EFFECT statement. For example, the effect names generated with the following statement are spl_x1 and spl_x2:

```
EFFECT spl = spline(x1 x2 / separate);
```

In procedures that support variable selection, such as the GLMSELECT procedure, these two effects can enter or leave the model independently during the selection process. Separated effects are not supported in the RANDOM statement of the GLIMMIX procedure.

SPLIT

specifies that each individual column in the design matrix that corresponds to the spline effect be treated as a separate effect that can enter or leave the model independently. Names for these split effects are generated by appending the variable name and an index for each column to the name that you specify in the EFFECT statement. For example, the effects generated for the spline effect in the following statement are spl_x1:1, spl_x1:2, . . . , spl_x1:7 and spl_x2:1, spl_x2:2, . . . , spl_x2:7:

```
EFFECT spl = spline(x1 x2 / split);
```

The SPLIT option is not supported in the GLIMMIX procedure.

Splines and Spline Bases

This section provides details about the construction of spline bases with the EFFECT statement. A spline function is a piecewise polynomial function in which the individual polynomials have the same degree and connect smoothly at join points whose abscissa values, referred to as knots, are prespecified. You can use spline functions to fit curves to a wide variety of data.

A spline of degree 0 is a step function with steps located at the knots. A spline of degree 1 is a piecewise linear function where the lines connect at the knots. A spline of degree 2 is a piecewise quadratic curve whose values and slopes coincide at the knots. A spline of degree 3 is a piecewise cubic curve whose values, slopes, and curvature coincide at the knots. Visually, a cubic spline is a smooth curve, and it is the most commonly used spline when a smooth fit is desired. Note that when no knots are used, splines of degree d are simply polynomials of degree d.

More formally, suppose you specify knots $k_1 < k_2 < k_3 < \cdots < k_n$. Then a spline of degree $d \geq 0$ is a function $S(x)$ with $d - 1$ continuous derivatives such that

$$S(x) = \begin{cases} P_0(x) & x < k_1 \\ P_i(x) & k_i \leq x < k_{i+1};\ i = 1, 2, \ldots, n-1 \\ P_n(x) & x \geq k_n \end{cases}$$

where each $P_i(x)$ is a polynomial of degree d. The requirement that $S(x)$ has $d - 1$continuous derivatives is satisfied by requiring that the function values and all derivatives up to order $d - 1$ of the adjacent polynomials at each knot match.

A counting argument yields the number of parameters that define a spline with n knots. There are $n + 1$ polynomials of degree d, giving $(n + 1)(d + 1)$ coefficients. However, there are d restrictions at each of the n knots, so the number of free parameters is $(n + 1)(d + 1) - nd = n + d + 1$. In mathematical terminology this says that the dimension of the vector space of splines of degree d on n distinct knots is $n + d + 1$. If you have $n + d + 1$ basis vectors, then you can fit a curve to your data by regressing your dependent variable by

using this basis for the corresponding design matrix columns. In this context, such a spline is known as a regression spline. The EFFECT statement provides a simple mechanism for obtaining such a basis.

If you remove the restriction that the knots of a spline must be distinct and allow repeated knots, then you can obtain functions with less smoothness and even discontinuities at the repeated knot location. For a spline of degree d and a repeated knot with multiplicity $m \leq d$, the piecewise polynomials that join such a knot are required to have only $d - m$ matching derivatives. Note that this increases the number of free parameters by $m - 1$ but also decreases the number of distinct knots by $m - 1$. Hence the dimension of the vector space of splines of degree d with n knots is still $n + d + 1$, provided that any repeated knot has a multiplicity less than or equal to d.

The EFFECT statement provides support for the commonly used *truncated power function* basis and *B-spline* basis. With exact arithmetic and by using the complete basis, you obtain the same fit with either of these bases. The following sections provide details about constructing spline bases for the space of splines of degree d with n knots that satisfies $k_1 \leq k_2 \leq k_3 < \cdots \leq k_n$.

Truncated Power Function Basis

A truncated power function for a knot k_i is a function defined by

$$t_i(x) = \begin{cases} 0 & x < k_i \\ (x - k_i)^d & x \geq k_i \end{cases}$$

Figure 20.1 shows such functions for $d = 1$ and $d = 3$ with a knot at $x = 1$.

Figure 20.1 Truncated Power Functions with Knot at $x = 1$

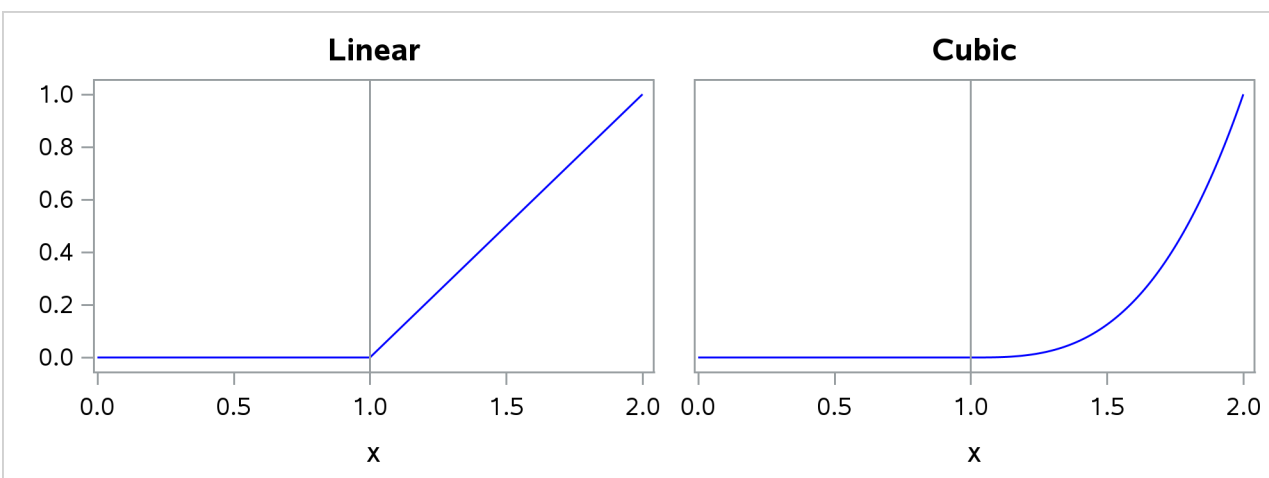

The name is derived from the fact that these functions are shifted power functions that get truncated to zero to the left of the knot. These functions are piecewise polynomial functions with two pieces whose function values and derivatives of all orders up to $d - 1$ are zero at the defining knot. Hence these functions are splines of degree d. It is easy to see that these n functions are linearly independent. However, they do not form a basis, because such a basis requires $n + d + 1$ functions. The usual way to add $d + 1$ additional basis functions is to use the polynomials $1, x, x^2, \ldots, x^d$. These $d + 1$ functions together with the n truncated power functions $t_i(x), i = 1, 2, \ldots, n$ form the truncated power basis.

Note that each time a knot is repeated, the associated exponent used in the corresponding basis function

is reduced by 1. For example, for splines of degree d with three repeated knots $k_i = k_{i+1} = k_{i+2}$ the corresponding basis functions are $t_i(x) = (x - k_i)_+^d$, $t_{i+1}(x) = (x - k_i)_+^{d-1}$, and $t_{i+2}(x) = (x - k_i)_+^{d-2}$. Provided that the multiplicity of each repeated knot is less than or equal to the degree, this construction continues to yield a basis for the associated space of splines.

The main advantage of the truncated power function basis is the simplicity of its construction and the ease of interpreting the parameters in a model that corresponds to these basis functions. However, there are two weaknesses when you use this basis for regression. These functions grow rapidly without bound as x increases, resulting in numerical precision problems when the x data span a wide range. Furthermore, many or even all of these basis functions can be nonzero when evaluated at some x value, resulting in a design matrix with few zeros that precludes the use of sparse matrix technology to speed up computation. This weakness can be addressed by using a B-spline basis.

B-Spline Basis

A B-spline basis can be built by starting with a set of Haar basis functions, which are functions that are 1 between adjacent knots and 0 elsewhere, and then applying a simple linear recursion relationship d times, yielding the $n + d + 1$ needed basis functions. For the purpose of building the B-spline basis, the n prespecified knots are referred to as internal knots. This construction requires d additional knots, known as boundary knots, to be positioned to the left of the internal knots, and $\mathrm{MAX}(d, 1)$ boundary knots to be positioned to the right of the internal knots. The actual values of these boundary knots can be arbitrary. The EFFECT statement provides several methods for placing the needed boundary knots, including the common method of using repeated values of the data extremes as the boundary knots. The boundary knot placement affects the precise form of the basis functions that are generated, but it does not affect the following two desirable properties:

1. The B-spline basis functions are nonzero over an interval that spans at most $d + 2$ knots. This yields design matrix columns each of whose rows contain at most $d + 2$ adjacent nonzero entries.

2. The computation of the basis functions at any x value is numerically stable and does not require evaluating powers of this value.

The following figures show the B-spline bases defined on [0, 1] with four equally spaced internal knots at 0.2, 0.4, 0.6, and 0.8.

Figure 20.2 shows a linear B-spline basis. Note that this basis consists of six functions each of which is nonzero over an interval that spans at most three knots.

Figure 20.2 Linear B-Spline Basis with Four Equally Spaced Interior Knots

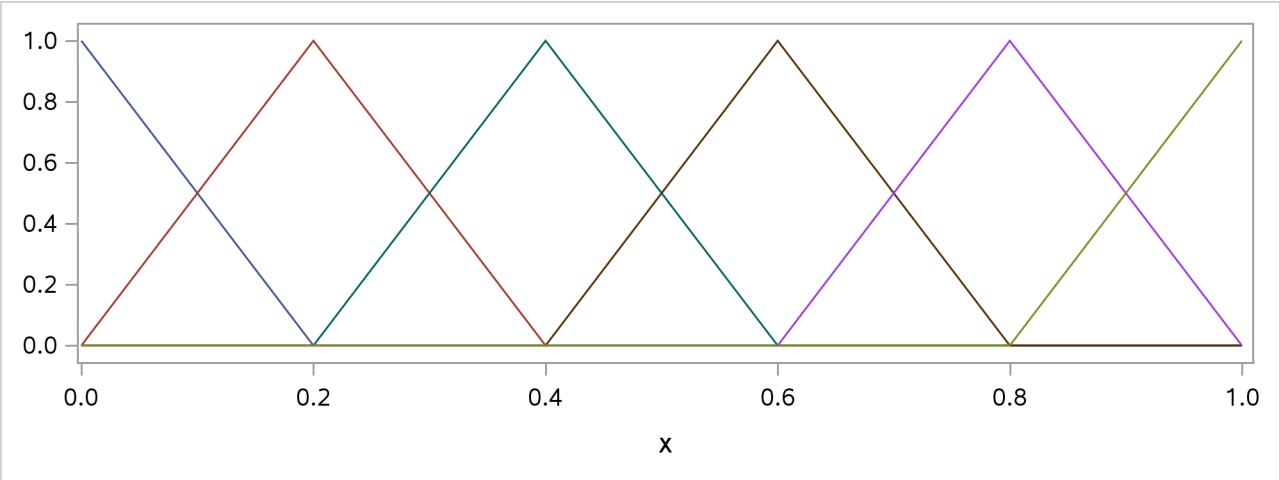

Figure 20.3 shows a cubic B-spline basis where the needed boundary knots are positioned at $x = 0$ and $x = 1$. Note that this basis consists of eight functions, each of which is nonzero over an interval spanning at most five knots.

Figure 20.3 Cubic B-Spline Basis with Four Equally Spaced Interior Knots

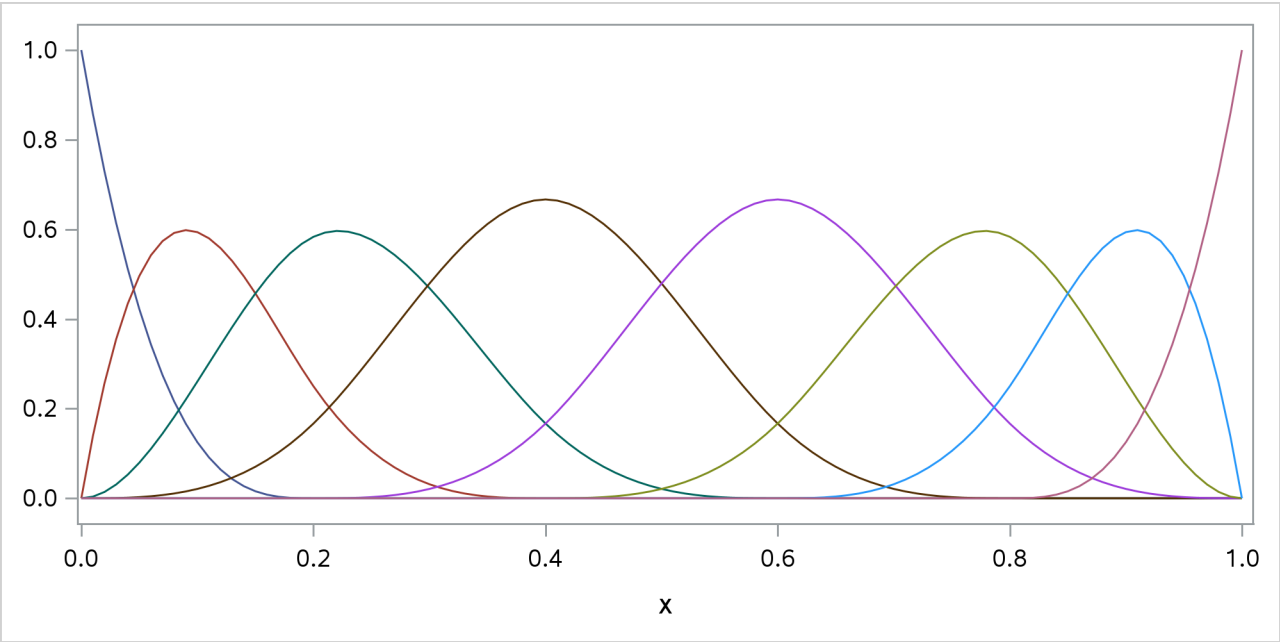

Figure 20.4 shows a different cubic B-spline basis where the needed left-side boundary knots are positioned at –0.6, –0.4, –0.2, and 0. The right-side boundary knots are positioned at 1, 1.2, 1.4, and 1.6. Note that, as in the basis shown in Figure 20.3, this basis consists of eight functions, each of which is nonzero over an interval spanning at most five knots. The different positioning of the boundary knots has merely changed the shape of the individual basis functions.

Figure 20.4 Cubic B-Spline Basis with Equally Spaced Boundary and Interior Knots

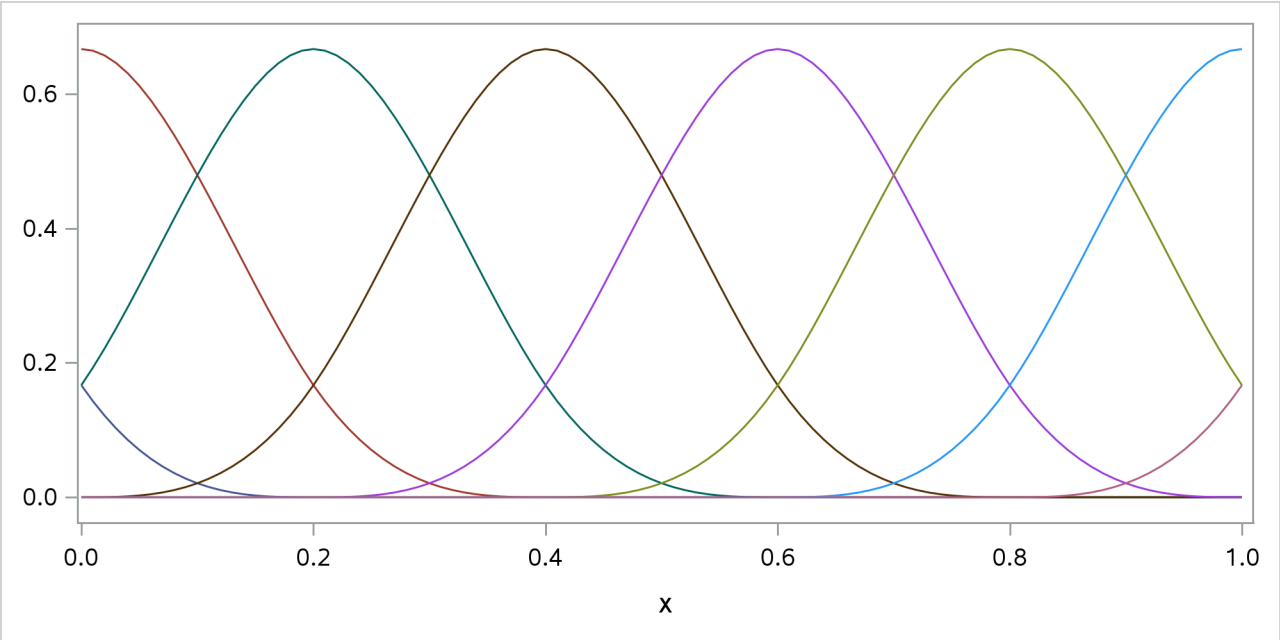

You can find details about this construction in Hastie, Tibshirani, and Friedman (2001).

Natural Cubic Spline Basis

Natural cubic splines are cubic splines with the additional restriction that the splines are required to be linear beyond the extreme knots. Some authors prefer the terminology "restricted cubic splines" to "natural cubic splines." The space of unrestricted cubic splines on n knots has the dimension $n + 4$. Imposing the restrictions that the cubic polynomials beyond the first and last knot reduce to linear polynomials reduces the number of degrees of freedom by 4, and so a basis for the natural cubic splines consists of n functions. Starting from the truncated power function basis for the unrestricted cubic splines, you can obtain a reduced basis by imposing linearity constraints. You can find details about this construction in Hastie, Tibshirani, and Friedman (2001). Figure 20.5 shows this natural cubic spline basis defined on $[0, 1]$ with four equally spaced internal knots at 0.2, 0.4, 0.6, and 0.8. Note that this basis consists of four basis functions that are all linear beyond the extreme knots at 0.2 and 0.8.

Figure 20.5 Natural Cubic Spline Basis with Four Equally Spaced Knots

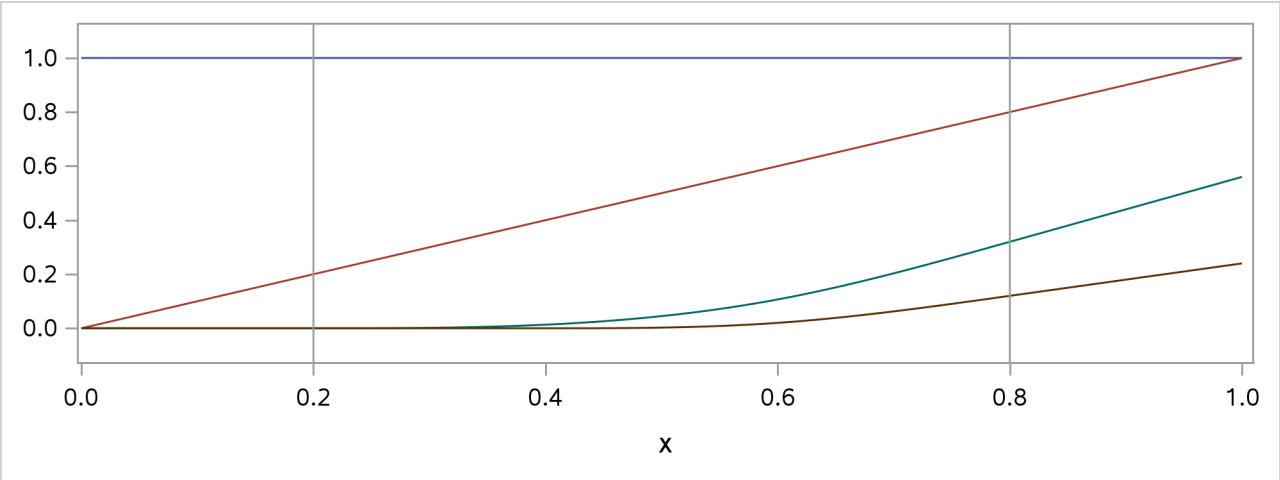

ODS Table Names

Each constructed effect produces one or more informative tables when the DETAILS option is also specified. Each table has a name associated with it. You can use this name to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 20.15. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 20.15 ODS Tables Produced by the EFFECT Statement

Table Name	Description	Option
BSplineDetails	B-spline basis details	BASIS=BSPLINE
CollectionLevelInfo	Levels of collection effects	
LagDesign	Subject-by-period design for lag effects	
LagInfo	Number of lags and lag variable information	
MMLevelInfo	Levels of multimember effects	
PolynomialDetails	Number of variables and columns, polynomial degree, and standardization method	
PolynomialScaling	Centering and scaling details	STANDARDIZE
SplineKnots	Knot and boundary knot values	
TPFSplineDetails	Truncated power function spline basis details	BASIS=TPF or NATURALCUBIC

EFFECTPLOT Statement

This statement applies to the following SAS/STAT procedures: GEE, GENMOD, LOGISTIC, ORTHOREG, PLM, and PROBIT.

The EFFECTPLOT statement produces a display (*effect plot*) of a complex fitted model and provides options for changing and enhancing the display. One simple effect plot is the display for a linear regression of the response Y on a single predictor X: the regression line is drawn with the predicted response on the Y axis and the covariate on the X axis. The regression line can be enhanced by displaying the observations and adding confidence and prediction limits. When your model is more complicated—with more continuous and categorical covariates, nestings and interactions, and link functions—the effect plots display the behavior of some covariates over their ranges while holding other covariates at some fixed values; this can enable easier interpretation and explanation of the resulting model.

By default, a single plot is produced based on the type of response variable and the number of continuous and classification covariates in the model. You can also specify options to do the following:

- select the variables to display in the plots
- produce multiple plots based on the following: the levels of classification covariates; the minimum, maximum, mean, or middle (midrange) value of continuous covariates; and specified values of the covariates
- specify different fixed values for continuous and classification covariates that are not displayed in the plot
- panel and unpanel plots
- select variables to slice or group by
- display (or remove from display) observations and confidence limits

Syntax: EFFECTPLOT Statement

EFFECTPLOT < *plot-type* < **(***plot-definition-options***)** > > < / *options* > **;**

The available *plot-types* and their *plot-definition-options* are described in Table 20.16. Table 20.18 lists the *options* that can be specified after a slash (/) for any *plot-type*, and Table 20.19 lists additional *options* that enhance specific *plot-types*. Full descriptions of the *plot-definition-options* and the other *options* are provided in the section "Dictionary of Options" on page 450.

Table 20.16 *Plot-Types* and *Plot-Definition-Options*

Plot-Type and Description	Plot-Definition-Options
BOX Displays a box plot of continuous response data at each level of a CLASS effect, with predicted values superimposed and connected by a line. This is an alternative to the INTERACTION *plot-type*.	PLOTBY= variable or CLASS effect X= CLASS variable or effect
CONTOUR Displays a contour plot of predicted values against two continuous covariates.	PLOTBY= variable or CLASS effect X= continuous variable Y= continuous variable
FIT Displays a curve of predicted values versus a continuous variable.	PLOTBY= variable or CLASS effect X= continuous variable
INTERACTION Displays a plot of predicted values (possibly with error bars) versus the levels of a CLASS effect. The predicted values are connected with lines and can be grouped by the levels of another CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= CLASS variable or effect
MOSAIC Displays a mosaic plot of predicted values using up to three CLASS effects.	PLOTBY= variable or CLASS effect X= CLASS effects
SLICEFIT Displays a curve of predicted values versus a continuous variable grouped by the levels of a CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= continuous variable

By default, a single plot is produced based on the type of response variable and the number of continuous and classification covariates in the model, as shown in Table 20.17. If you have a polytomous response model, then the response variable is treated as the grouping classification variable in this table. If your model does not fit into Table 20.17, then a default plot is not produced; however, specifying the *plot-type* argument displays a plot with the extra continuous covariates fixed at their mean values and the extra classification covariates fixed at their reference levels.

Table 20.17 Default *Plot-Types*

Number of Covariates		Type of Response Variable	
Classification	**Continuous**	**Continuous or Binary**	**Polytomous**
1	0	INTERACTION	INTERACTION with groups
2	0	INTERACTION with groups	None
2	0	INTERACTION with nested covariate	None
0	1	FIT	SLICEFIT
0	2	CONTOUR	None
1	1	SLICEFIT	None

Table 20.18 and Table 20.19 list the *options* that can be specified after a slash (/) to enhance the effect plots.

Table 20.18 Available *Options* for All *Plot-Types*

AT< *args* >	ILINK	NOOFFSET	NROWS=[ab]	PREDLABEL=
ATLEN=	INDIVIDUAL[a]	NCOLS=[ab]	OBS< (*options*) >	UNPACK
ATORDER=	LINK[b]	NOOBS[ab]	PLOTBYLEN=	

[a] Not available for the BOX *plot-type*.
[b] Not available for the MOSAIC *plot-type*.

NOTE: If your model contains an offset variable and the NOOFFSET option is not specified or not valid, then the predicted values are computed only at the observations. In this case, the FIT and SLICEFIT *plot-types* display scatter plots of the predicted values, the CONTOUR *plot-type* displays the residuals against two continuous covariates but with no fitted surface, the INTERACTION *plot-type* does not connect the predicted values with lines, and the BOX *plot-type* is unchanged.

Table 20.19 Additional *Options* for Each *Plot-Type*

Plot-Type	***Options***
BOX	BOXWIDTH=, CLUSTER, CLUSTERGRID=, CONNECT, NOCLUSTER, NOCONNECT, NOLABELOUTLIER, YRANGE=
CONTOUR	EXTEND=, GRIDSIZE=
FIT	ALPHA=, EXTEND=, GRIDSIZE=, NOCLI, NOCLM, NOLIMITS, SMOOTH, YRANGE=
INTERACTION	ALPHA=, CLI, CLM, CLUSTER, CLUSTERGRID=, CONNECT, LIMITS, NOCLUSTER, NOCONNECT, POLYBAR, YRANGE=
MOSAIC	ADDCELL, BIN, EQUAL, NOBORDER, TYPE=
SLICEFIT	ALPHA=, CLI, CLM, EXTEND=, GRIDSIZE=, LIMITS, YRANGE=

Dictionary of Options

You can specify the following EFFECTPLOT *options*.

ADDCELL<=*value***>**
adds *value* to the weight of every cell in the MOSAIC *plot-type*. You can specify *value* as any nonnegative number. If you do not specify a *value*, then *value*=0.5. This enables you to add some dimension to zero frequency cells.

ALPHA=*value*
specifies the significance level, $0 \leq$ *value* ≤ 1, for producing $100(1 - value/2)\%$ prediction and confidence limits. By default, *value*=0.05.

AT < *contopt* > < *classopt* > < (*varspec1*< *varspec2*... >) >
where the available *contopt* keywords are **MEAN**, **MIN**, **MAX**, and **MIDRANGE**, and the available *classopt* keywords are **ALL**, **REF**, and **BALANCE**. The *varspec* arguments have the following two forms:

variable< **(CODED)** >=*varopt*
class-variable(*level*)=*value*

where a *varopt* is *contopts*, *number-list*, *classopts*, or *'class-level'*... *'class-level'*.

The AT option specifies values at which to fix continuous and CLASS variables when they are not used in X=, Y=, SLICEBY=, or PLOTBY= effects. The *contopt* keyword fixes continuous variables at their mean, minimum, maximum, or midrange= $\frac{1}{2}(\text{minimum} + \text{maximum})$; the default is to use the mean. The *classopt* keyword either fixes a CLASS variable at its reference (last) level, indicates that all levels of the CLASS variable should be processed, or specifies a special *plug-in* coding as described in the following paragraphs; the default is to use the reference level. The *varopt* values enable you to specify *contopt* and *classopt* keywords or to specify lists of numbers or class levels. You can specify a CLASS variable only once in the AT specification, but you can specify a continuous variable multiple times; for example, the following syntax is valid when X is a continuous variable:

```
effectplot / at(x=min max x=0 to 2 by 1 x=2 5 7);
```

Duplicate AT values are suppressed, so the last X=2 value is ignored.

You can also specify *coded plug-in values* for CLASS variable levels when computing the predicted values $\mathbf{x}'\boldsymbol{\beta}$. For example, suppose a CLASS variable A with two levels={0,1} is in the model. Then instead of using the coding for A in the x vector by specifying **AT(A=all)**, **AT(A=ref)** or **AT(A='0' '1')**, you can specify a numeric list to plug in. For example, if the proportion of A's that equal 0 in the data set is 0.3, then you can input the proportions for all levels of the variable by specifying **AT(A(CODED)=0.3 0.7)**. Under GLM coding, A=0 is coded as "1 0" and A=1 is coded as "0 1", so the plug-in specification replaces both of these codings with "0.3 0.7". Under REFERENCE coding, A=0 is coded as "1" and A=1 is coded as "0", so this specification replaces both of these codings with "0.3" followed by "0.7"; however, if another variable is nested within A, then only "0.3" is used. To plug in values, you must specify a multiple of the number of parameters used for the CLASS variable or, if a variable is nested within the CLASS variable, a multiple of the number of levels of the CLASS variable.

There are two shorthand methods available for specifying coded plug-in values for CLASS variables. The BALANCE keyword enables you to set all levels of a CLASS variable to the reciprocal of the number of its levels; for example, if variable A has k levels, then specifying **AT(A=BALANCE)** sets all levels of the coding of A to $\frac{1}{k}$. You can also set one *level* of the CLASS variable to a specific *value* and set all the other levels to 0; for example, specifying **AT(A(2)=1)** sets the second level of A to 1 and all other levels to 0.

The coded plug-in values are distributed through the rest of the model effects in the following fashion. If a variable is nested within a plug-in variable, then its coding is multiplied by the plug-in value for the level it is nested in. If a variable interacts with a plug-in variable, its coding is multiplied by the appropriate plug-in value for the level it is interacting with. Lag, multimember, polynomial, and spline constructed effects are affected only by interactions and nestings. If the plug-in variable is part of a collection effect, then its values are replaced by the plug-in values; collection effects are also affected by interactions and nestings.

The AT levels are used for computing the predicted values. If the OBS option is also specified, then all observations are still displayed in all the plots. For example, if you specify the options **AT(A='1') OBS**, then the fitted values are computed by using A=1, but all the observations are displayed with their predicted values computed at their observed level of A. If you want to display only a subset of the observations based on the levels of a CLASS variable, then you must specify either the PLOTBY= option or the OBS(BYAT) option.

ATLEN=*n*

specifies the maximum length ($1 \leq n \leq 256$) of the levels of the AT variables that are displayed in footnotes and headers. By default, up to 256 characters of the CLASS levels are displayed, and the continuous AT levels are displayed with a BEST format that has a width greater than or equal to 5, which distinguishes each level. **CAUTION:** If the levels of your AT variables are not unique when the first n characters are displayed, then the levels are combined in the plots but not in the underlying computations. Also, at most n characters for continuous AT variables are displayed.

ATORDER=ASCENDING | DESCENDING

uses the AT values for continuous variables in ascending or descending order as specified. By default, values are used in the order of their first appearance in the AT option.

BIN

displays the statistic for the MOSAIC *plot-type* with a discrete coloring scheme.

BOXWIDTH=*value*

specifies the relative width ($0 \leq value \leq 1$) of the boxes of the BOX *plot-type*.

CLI

displays normal (Wald) prediction limits. This option is available only for normal distributions with identity links. If your model is from a Bayesian analysis, then sampling-based intervals are computed; for more information, see the section "Analysis Based on Posterior Estimates" on page 7807 in Chapter 94, "The PLM Procedure."

CLM

displays confidence limits. These are computed as the normal (Wald) confidence limits for the linear predictor, and if the ILINK option is specified, the limits are also back-transformed by the inverse link function. If your model is from a Bayesian analysis, then sampling-based intervals are computed; for more information, see the section "Analysis Based on Posterior Estimates" on page 7807 in Chapter 94, "The PLM Procedure."

CLUSTER< =*percent* >

modifies the BOX and INTERACTION *plot-types* by displaying the levels of the SLICEBY= effect side by side. You can specify *percent* as a percentage of half the distance between X levels. The *percent* value must be between 0.1 and 1; the default *percent* depends on the number of X levels, the number of levels of the SLICEBY= effect, and the number of levels of the PLOTBY= effect for INTERACTION *plot-types*. You can remove default clustering by specifying the NOCLUSTER option.

CLUSTERGRID< =YES | NO >

modifies the BOX and INTERACTION *plot-types* by separating the X levels with grid lines when the levels of the SLICEBY= effect are clustered.

CONNECT

modifies the BOX and INTERACTION *plot-types* by connecting the predicted values with a line. You can remove default connecting lines by specifying the NOCONNECT option.

EQUAL

causes every cell in the MOSAIC *plot-type* to have the same dimensions.

EXTEND=DATA | *value*

extends continuous covariate axes by $value \times \frac{1}{2} range$ in both directions, where $range$ is the range of the X axis. Specifying the DATA keyword displays curves to the range of the data within the appropriate SLICEBY=, PLOTBY=, and AT level. For the CONTOUR *plot-type*, *value*=0.05 by default; other *plot-types* set the default value to 0. When constructed effects are present, only the EXTEND=DATA option is available.

GRIDSIZE=*n*

specifies the resolution of curves by computing the predicted values at *n* equally spaced values on the X axis and specifies the resolution of surfaces by computing the predicted values on an $n \times n$ grid of points. Default values are $n = 200$ for curves and bands, $n = 50$ for surfaces, and $n = 2$ for lines. If results of a Bayesian or bootstrap analysis are being displayed, then the defaults are $n = 500000/B$, where B is the number of samples, the upper limit is equal to the usual defaults, and the lower limit is equal to 20.

ILINK

displays the fit on the scale of the inverse link function. In particular, the results are displayed on the probability scale for logistic regression. By default, a procedure displays the fit on either the link or inverse link scale.

INDIVIDUAL

displays individual probabilities for polytomous response models with cumulative links on the scale of the inverse link function. This option is not available when the LINK option is specified, and confidence limits are not available when you specify this option.

LIMITS

invokes the CLI and CLM options.

LINK

displays the fit on the scale of the link function—that is, the linear predictor. Probabilities or observed proportions near 0 and 1 are transformed to ±20. By default, a procedure displays the fit on either the link or inverse link scale.

NCOLS=*n*

specifies the maximum number of columns in a paneled plot. This option is not available with the BOX *plot-type*.

The default choice of NROWS= and NCOLS= is based on the number of PLOTBY= and AT levels. If only one plot is displayed in a panel, then NROWS=1 and NCOLS=1 and the plots are produced as if you specified only the UNPACK option. If only two plots are displayed in a panel, then NROWS=1 and NCOLS=2. For all other cases, a 2×2, 2×3, or 3×3 panel is chosen based on how much of the last panel is used, with ties going to the larger panels. For example, if 14 plots are being created, then this requires either four 2×2 panels with 50% of the last panel filled, three 2×3 panels with 33% of the last panel filled, or two 3×3 panels with 55% of the last panel filled; in this case, the 3×3 panels are chosen.

If you specify both the NROWS= and NCOLS= options, then those are the values used. However, if you specify only one of the options but have fewer plots, then the panel size is reduced; for example, if you specify NROWS=6 but have only four plots, then a plot that has four rows and one column is produced.

NOBORDER

removes the border from the cells in the MOSAIC *plot-type*. Otherwise, the color of the cells that were not observed in the data set is hidden by the border.

NOCLI

suppresses the prediction limits.

NOCLM

suppresses the confidence limits.

NOCLUSTER

modifies the BOX and INTERACTION *plot-types* by preventing the side-by-side display of the levels of the SLICEBY= effect.

NOCLUSTERGRID

removes the grid lines that separate the levels of the CLASS covariates on the X axis when the SLICEBY= effect is clustered by default.

NOCONNECT

modifies the BOX and INTERACTION *plot-types* by suppressing the line that connects the predicted values.

NOLABELOUTLIER

modifies the BOX *plot-types* by suppressing the labels that are displayed on the outliers.

NOLIMITS

invokes the NOCLI and NOCLM options.

NOOBS

suppresses the display of observations and overrides the specification of the OBS= option.

NOOFFSET

MOFF

moves the offset for a Poisson regression model to the response side of the equation. If the ILINK option is also specified, then the rate is displayed on the Y axis; if the LINK option is also specified, then the log of the rate is displayed on the Y axis. Without the NOOFFSET option, the predicted values are computed and displayed only for the observations.

NROWS=*n*

specifies the maximum number of rows in a paneled plot. This option is not available with the BOX *plot-type*. For more information, see the NCOLS= option.

OBS< (*obs-options***) >**

displays observations in the effect plots. An input data set is required; hence the OBS option is not available with PROC PLM. The OBS option is overridden by the NOOBS option. When the ILINK option is specified with binary response variables, then either the observed proportions or a coded value of the response is displayed. For polytomous response variables, the observed values are overlaid onto the fitted curves unless the LOCATION= option is specified. Whether or not observations are displayed by default depends on the procedure. If the PLOTBY= option is specified, then the observations that are displayed in each plot are from the corresponding PLOTBY= level for classification effects; for continuous effects, all observations are displayed in every plot.

You can specify the following *obs-options*:

BYAT

subsets the observations by AT level and by the PLOTBY= level. If you specify the PLOTBY= option without specifying this option, the observations are displayed in the plots that correspond to their PLOTBY= level without regard to any classification variables specified in the AT option. However, for FIT *plot-types* a distance can be computed and displayed (for more information, see the DISTANCE option). This option is ignored when there are no AT variables.

CDISPLAY=NONE | OUTLINE | GRADIENT | OUTLINEGRADIENT

controls the display of observations in contour plots. The keyword OUTLINE displays the observations as circles, GRADIENT displays gradient-colored dots, OUTLINEGRADIENT displays gradient-filled-circles, and NONE suppresses the display of the observations. The default is CDISPLAY=OUTLINEGRADIENT.

CGRADIENT=RESIDUAL | DEPENDENT

specifies what the gradient shading of the observed values in the CONTOUR *plot-type* represents. The RESIDUAL keyword shades the observations by the raw residual value and displays the fitted surface as a line contour plot. The DEPENDENT keyword shades the observations by the response variable value and displays the fitted surface as a contour shaded on the same scale. The default is CGRADIENT=DEPENDENT.

DEPTH=*depth*

specifies the number of overlapping observations that can be distinguished by adjusting their transparency; you can specify $1 \leq \textit{depth} \leq 100$. By default, DEPTH=1. The DEPTH= option is available with the FIT, SLICEFIT, and INTERACTION *plot-types*.

DISTANCE

displays observations in FIT *plot-types* with a color gradient that indicates how far the observation is from the AT and PLOTBY= level. This option is ignored unless an AT or PLOTBY= option is specified.

The distance is computed as the square root of the following number: for each continuous AT and PLOTBY= variable, add the square of the difference from the observed value divided by the range of the variable; for each CLASS AT and PLOTBY= variable, add 1 if the CLASS levels are different. Thus the largest possible distance is the square root of the number of AT and PLOTBY= variables. Observations at zero distance are displayed by using the darkest color, and the color fades as the distance increases.

The unpacked panels compute the maximum distance within each panel and hence do not use the same gradient across all panels. Also, the PANELS *panel-type* computes the maximum distance within each PLOTBY= level, so a different gradient is used for each PLOTBY= level. All other *panel-types* compute the maximum distance across all observations and therefore use the same gradient in every plot.

FITATCLASS

computes fitted values only for class levels that are observed in the data set. This option is ignored when the GLM parameterization is used.

FRINGE

displays observations in a fringe (rug) plot at the bottom of the plot. This option is available only with the FIT and SLICEFIT *plot-types*.

JITTER< (*jitter-options***) >**

shifts (*jitters*) the observations. By default, the jittering in the X direction is achieved by adding a random number that is generated according to a normal distribution with mean=0 and standard deviation=*x-jitter*/2 and truncating at $\pm$*x-jitter*, where *x-jitter*=0.01 times the range of the X axis; the jittering in the Y direction is performed independently but in the same fashion. The JITTER option is not available with the BOX *plot-type*. You can specify the following *jitter-options*:

FACTOR=*factor* sets the jitter to *factor* times the range of the axis, and jitters in both the X and Y directions. You can specify $0 \leq \textit{factor} \leq 1$.

SEED=*seed* specifies an integer to use as the initial seed for the random number generator. If you do not specify a seed, or if you specify a value less than or equal to zero, then the time of day from the computer clock is used to generate an initial seed.

X=*x-jitter* sets the jitter to *x-jitter* for the X direction; the jitter in the Y direction is assumed to be 0 unless the Y= option is also specified. You can specify *x-jitter* ≥ 0. The X= option is not available for the INTERACTION *plot-type*. This option is ignored if the FACTOR= option is also specified.

Y=*y-jitter* sets the jitter to *y-jitter* for the Y direction; the jitter in the X direction is assumed to be 0 unless the X= option is also specified. You can specify *y-jitter* ≥ 0. This option is ignored if the FACTOR= option is also specified.

LABEL< =OBS >
labels markers with their observation number.

LOCATION=*location*
specifies where the observed values for polytomous response models are displayed when the SLICEBY= variable is the response. This option is available only with the SLICEFIT and INTERACTION *plot-types*. The observations are always displayed at their appropriate X-axis value, but their Y-axis location can depend on the specification of the YRANGE= option or on the minimum and maximum computed predicted values in addition to the specified *location*. You can specify the following *locations*:

BOTTOM< =*factor* > displays the first response level at the minimum predicted value, and displays succeeding response levels above the first level at *factor* $\times range$ intervals, where $range$ is the range of the predicted values. You can specify $0 \leq$ *factor* ≤ 1, but the largest usable value, which corresponds to LOCATION=SPREAD, is *factor*=$\frac{1}{k}$, where $k+1$ is the number of response levels that are displayed. By default, *factor*=0.03.

CURVE displays the observations for polytomous response models at their predicted values. For displays on the LINK scale, the reference level is displayed at the maximum value. This method is the default.

FIRST displays the observations for a response level at the first displayed predicted value for that response level.

MAX displays the observations for a response level at the maximum displayed predicted value for that response level.

MIDDLE displays the observations for a response level at the middle of the displayed predicted values for that response level.

MIN displays the observations for a response level at the minimum displayed predicted value for that response level.

SPREAD displays the observations with the response levels evenly spread across the Y axis.

TOP< =*factor* > displays the last response level at the maximum predicted value, and displays preceding response levels below the last level at *factor* $\times range$ intervals, where $range$ is the range of the predicted values. You can specify $0 \leq$ *factor* ≤ 1, but the largest usable value, which corresponds to LOCATION=SPREAD, is *factor*=$\frac{1}{k}$, where k+1 is the number of response levels that are displayed. By default, *factor*=0.03.

PLOTBY< (*panel-type*) >=*effect*< =*numeric-list* >

specifies a variable or CLASS effect at whose levels the predicted values are computed and the plots are displayed. You can specify the response variable as the *effect* for polytomous response models. The *panel-type* argument specifies the method in which the plots are grouped for the display. You can specify the following *panel-types*:

COLUMNS

specifies that the columns within each panel correspond to different levels of the PLOTBY= effect and hence the rows correspond to different AT levels.

PACK

specifies that plots be displayed in the panels as they are produced, with no control over the placement of the PLOTBY= and AT levels.

PANELS | LEVELS

specifies that each level of the PLOTBY= effect begin a new panel of plots and the AT levels define the plots within the panels.

ROWS

specifies that the rows within each panel correspond to different levels of the PLOTBY= effect and hence the columns correspond to different AT levels.

This option is ignored with the BOX *plot-type*; box plots are always displayed in an unpacked fashion, grouped by the PLOTBY= and AT levels. If you specify a continuous variable as the *effect*, then either you can specify a *numeric-list* of values at which to display that variable or, by default, five equally spaced values from the minimum variable value to its maximum are displayed.

The default *panel-type* is based on the number of PLOTBY= and AT levels, as shown in the following table.

Number of PLOTBY= Levels	Number of AT Levels	Resulting *panel-type*
1	1	(UNPACK)
>1	1	PACK
1	>1	PACK
2	>1	ROWS
3	>1	COLUMNS
>3	>1	PANELS

The default dimensions of the panels are also based on the number of PLOTBY= and AT levels; for more information, see the NCOLS= option.

Specification of the *panel-type* is honored except in the following cases. If you specify a *panel-type* but produce only one plot, specify the NROWS=1 and NCOLS=1 options, or specify the UNPACK option, then the plots are produced as if you specified only the UNPACK option. If you specify the PANELS *panel-type* with only one AT level, then the plots are produced with the UNPACK option. However, if you specify the PANELS *panel-type* but the PLOTBY= effect has only one level, then the *panel-type* is changed to PACK.

PLOTBYLEN=*n*

specifies the maximum length ($1 \leq n \leq 256$) of the levels of the PLOTBY= variables, which are displayed in footnotes and headers. By default, up to 256 characters of the CLASS levels are displayed. **CAUTION:** If the levels of your PLOTBY= variables are not unique when the first *n* characters are displayed, then the levels are combined in the plots but not in the underlying computations.

POLYBAR

displays polytomous response data as a stacked histogram whose bar heights are defined by the individual predicted value. Your response variable must be the *effect* that is specified in the SLICEBY= option. If you specify the INDIVIDUAL option, then the histogram bars are displayed side by side. If you specify the CLM option, then error bars are displayed on the side-by-side histogram bars.

PREDLABEL='*label***'**

specifies a label to be displayed on the Y axis. The default Y-axis label is determined by your model. For the CONTOUR *plot-type*, this option changes the title to "*label* for Y."

SHOWCLEGEND

displays the gradient legend for the CONTOUR *plot-type*. This option has no effect when the OBS(CGRADIENT=RESIDUAL) option is also specified.

SLICEBY=NONE | *effect*< =*numeric-list*>

displays the fitted values at the different levels of the specified variable or CLASS effect. You can specify the response variable as the *effect* for polytomous response models. Use this option to modify the SLICEFIT, INTERACTION, and BOX *plot-types*. If you specify a continuous variable as the *effect*, then either you can specify a *numeric-list* of values at which to display that variable or, by default, five equally spaced values from the minimum variable value to its maximum are displayed. The NONE keyword is available for preventing the INTERACTION *plot-type* from slicing by a second classification covariate. The SLICEBY=NONE option is not available for the SLICEFIT *plot-type*, because that is the same as the FIT *plot-type*. The BOX *plot-type* accepts only classification effects.

SMOOTH

overlays a loess smooth on the FIT *plot-type* for models that have only one continuous predictor. This option is not available for binary or polytomous response models.

TYPE=PREDICTED | PARQUET | GOF

specifies the type of display for the MOSAIC *plot-types*. For effects that are specified as in the X= option, the TYPE=PREDICTED and TYPE=GOF mosaic plots create cells by dividing the X axis proportional to the total weight in each level of the *x-effect*, then dividing the Y axis according to the weight in each level of the *y-effect* within the *x-effect* levels, and dividing the X2 axis according to the weight in each level of the *x2-effect* within the *x-effect* and *y-effect* levels. The TYPE=PARQUET plot uses the predicted probabilities instead of the weights to determine the dimensions of the cells.

The default TYPE=PREDICTED mosaic plot colors the cells according to their predicted values (probabilities for binary and polytomous response models) computed at the AT and PLOTBY levels. The TYPE=GOF plot displays the Pearson goodness-of-fit statistic as in Friendly (2000), with the expected value computed at the AT and PLOTBY levels. For a cell i_y defined by the axis levels, the PLOTBY and AT levels, and the response level y, let W_{i_y} be the sum of all the weights of the observations in that cell, let $W_i = \sum_y W_{i_y}$ be the sum of the weights across all response levels, and let $\hat{Y}_y$ be the predicted response for that cell, where y is the event level for binary response models, and

$\hat{Y}_y = \Pr(Y = y|i_y)$ for binary and multinomial models. Then the Pearson goodness-of-fit statistic is computed as

$$\frac{W_{i_y} - W_i \hat{Y}}{\sqrt{W_i \hat{Y}}}$$

The TYPE=GOF plot is not available when you have continuous covariates in the model. The TYPE=PARQUET plot shades the cells with their observed weights and is available only with binary or polytomous response data.

UNPACK

suppresses paneling. By default, multiple plots can appear in some output *panels*. Specify UNPACK to display each plot separately.

X=*effect* | **(***x-effect* **<** *y-effect* **<** *x2-effect* **> >)**

specifies values to display on the X axis. For the BOX and INTERACTION *plot-types*, *effect* can be a CLASS effect in the MODEL statement. For the FIT, SLICEFIT, and CONTOUR *plot-types*, *effect* can be any continuous variable in the model. For the MOSAIC *plot-types*, you can specify CLASS effects (or the response variable if you have a multinomial model) as the *effect* or *x-effect* to display on the X axis, as the *y-effect* to display on the Y axis, and as the *x2-effect* to display on the X2 (upper) axis.

Y=*args*

specifies values to display on the Y axis for the CONTOUR *plot-type*. The Y= argument can be any continuous variable in the model.

YRANGE=CLIP | **(<** *min* **><** *,max* **>)**

displays the predicted values on the Y axis in the range [*min*,*max*]. The YRANGE=CLIP option has the same effect as specifying the minimum predicted value as *min* and the maximum predicted value as *max*. The axis might extend beyond your specified values. By default, when the Y axis displays predicted probabilities, the entire Y axis, [0,1], is displayed. This option is useful if your predicted probabilities are all contained in some subset of this range. This option is not available with the CONTOUR *plot-type*.

ODS Graphics: EFFECTPLOT Statement

To produce the EFFECTPLOT displays, ODS Graphics must be enabled. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." The available graph names are provided in Table 20.20.

Table 20.20 Graphs Produced by the EFFECTPLOT Statement

ODS Graph Name	Plot Description
BoxFitPlot	A box plot of the responses at each level of one classification effect, overlaid with a plot of the predicted values
ContourFitPlot	A contour plot of the fitted surface against two continuous covariates
ContourFitPanel	A panel of ContourFitPlots

Table 20.20 *continued*

ODS Graph Name	Plot Description
FitPlot	A curve of the predicted values plotted against one continuous covariate
FitPanel	A panel of FitPlots
InteractionPlot	A plot of the predicted values (connected by a line) against one classification effect, possibly for each level of a second classification effect
InteractionPanel	A panel of InteractionPlots
MosaicFitPlot	A mosaic plot of the predicted values categorized by one to three classification effects
SliceFitPlot	A curve of the predicted values against one continuous covariate for each level of a second classification covariate
SliceFitPanel	A panel of SliceFitPlots

Examples: EFFECTPLOT Statement

Example 20.1: A Saddle Surface

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icomep1.sas.)

Myers (1976) analyzes an experiment reported by Frankel (1961) that is aimed at maximizing the yield of mercaptobenzothiazole (MBT) by varying processing time and temperature. Myers uses a two-factor model in which the estimated surface does not have a unique optimum. The objective is to find the settings of time and temperature in the processing of the chemical that maximize the yield. The following statements create the data set d:

```
data d;
   input Time Temp MBT @@;
   label Time = "Reaction Time (Hours)"
         Temp = "Temperature (Degrees Centigrade)"
         MBT  = "Percent Yield Mercaptobenzothiazole";
   datalines;
 4.0  250  83.8    20.0  250  81.7    12.0  250  82.4
12.0  250  82.9    12.0  220  84.7    12.0  280  57.9
12.0  250  81.2     6.3  229  81.3     6.3  271  83.1
17.7  229  85.3    17.7  271  72.7     4.0  250  82.0
;
```

In the following statements, the ORTHOREG procedure fits a response surface regression model to the data and uses the EFFECTPLOT statement to create a slice of the response surface. The FIT *plot-type* requests plots of the predicted yield against the Time variable, and the PLOTBY= option specifies that the Temp variable is fixed at five equally spaced values so that five fitted regression curves are displayed in Output 20.1.1.

```
ods graphics on;
proc orthoreg data=d;
   model MBT=Time|Time|Temp|Temp@2;
   effectplot fit(x=time plotby=temp);
run;
ods graphics off;
```

The displays in Output 20.1.1 show that the slope of the surface changes as the temperature increases.

Output 20.1.1 Panel of Fit Plots

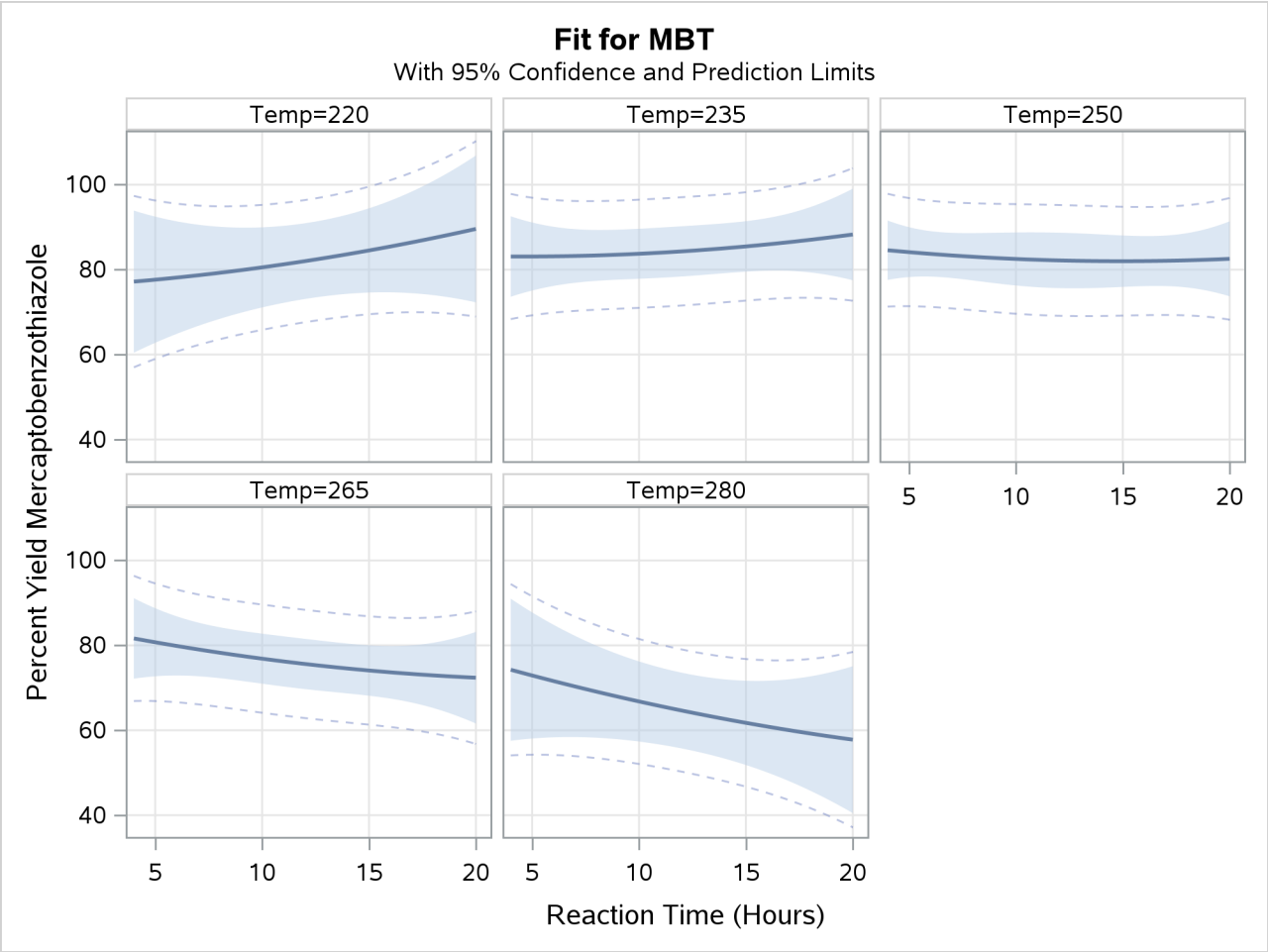

It might be more informative to see these results in one graphic, so the following statements specify the SLICEFIT *plot-type* to overlay plots of the predicted yield versus time, fixed at several values of temperature. In this case, the SLICEBY= option is specified to explicitly use the same four temperatures as used in the experiment.

```
ods graphics on;
proc orthoreg data=d;
   model MBT=Time|Time|Temp|Temp@2;
   effectplot slicefit(x=time sliceby=temp=229 250 271 280);
run;
ods graphics off;
```

Output 20.1.2 shows that to optimize the yield you should choose either low temperatures and long times or high temperatures and short times.

Output 20.1.2 Fit Plot Grouped (Sliced) by Temp

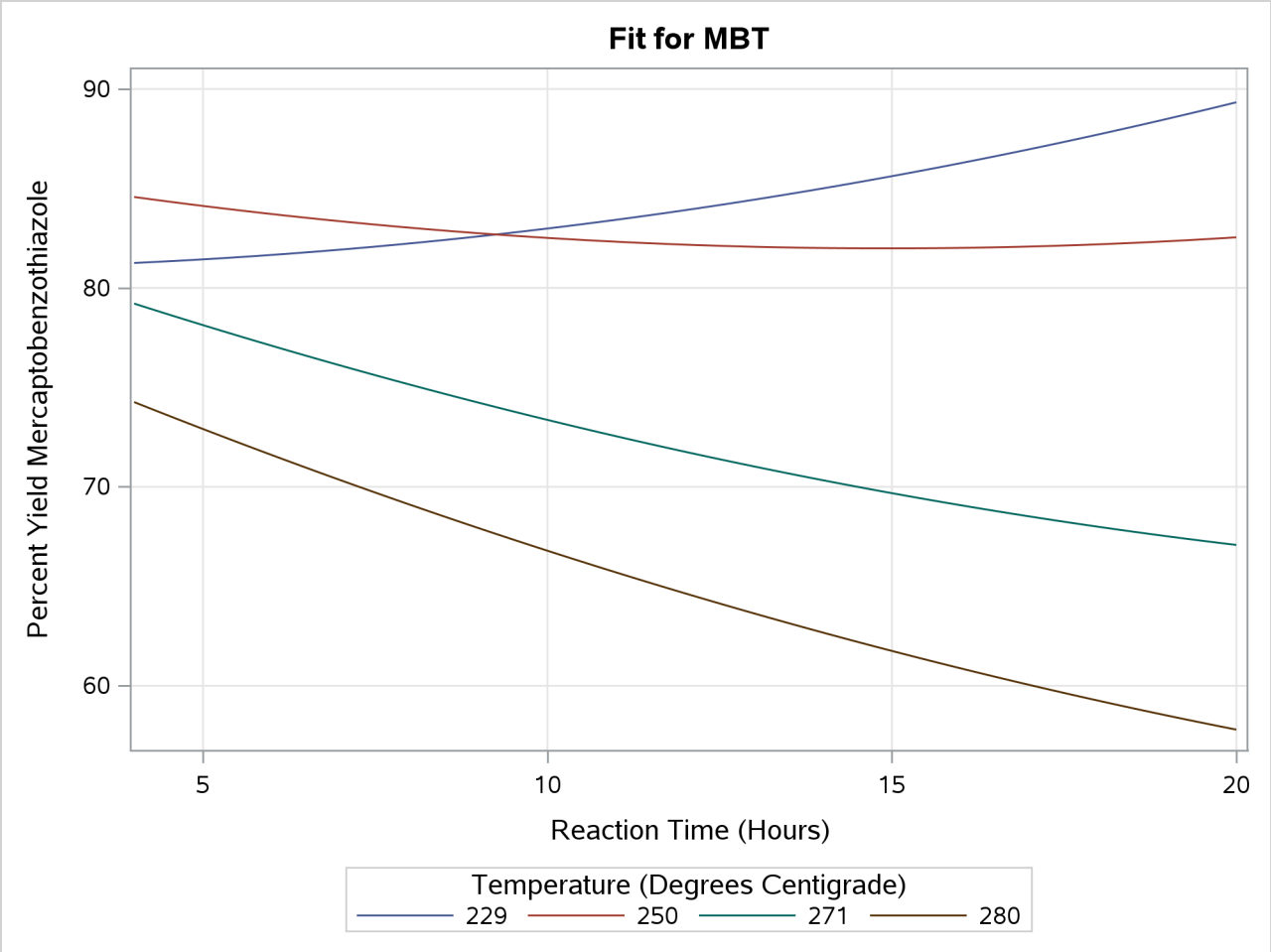

Another plot might explain the reason for these conflicting results more clearly. The following statements produce the default EFFECTPLOT statement display, enhanced by the OBS(JITTER) option to jitter the observations so that you can see the replicated points:

```
ods graphics on;
proc orthoreg data=d;
   model MBT=Time|Time|Temp|Temp@2;
   effectplot / obs(jitter(seed=39393));
run;
ods graphics off;
```

Output 20.1.3 shows that the reason for the changing slopes is that the surface is at a saddle point. This surface does not have an optimum point.

Output 20.1.3 Contour Fit Plot with Jittered Observations

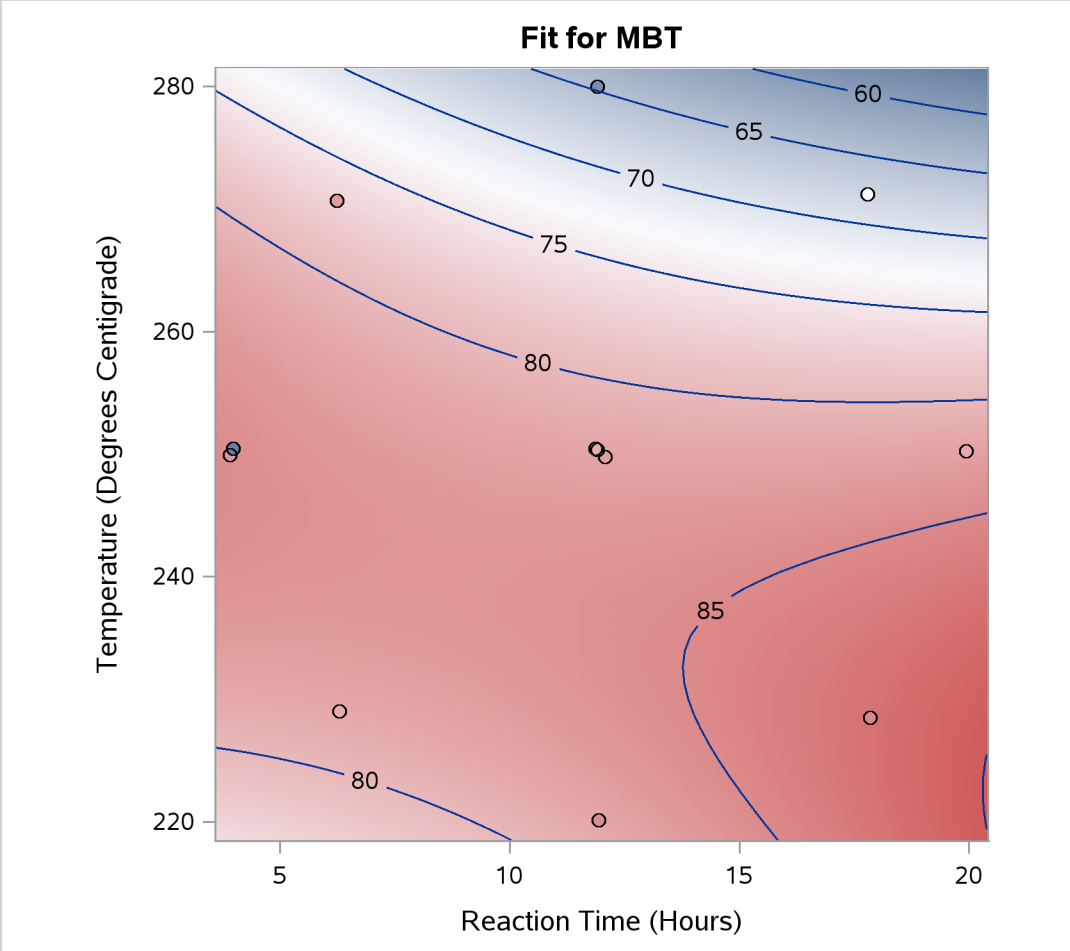

Example 20.2: Unbalanced Two-Way ANOVA

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icomep2.sas.)

This example uses data from Kutner (1974, p. 98) to illustrate a two-way analysis of variance. The original data source is Afifi and Azen (1972, p. 166). The following statements create the data set a:

```
data a;
   input drug disease @;
   do i=1 to 6;
      input y @;
      output;
   end;
   datalines;
1 1 42 44 36 13 19 22
1 2 33  . 26  . 33 21
1 3 31 -3  . 25 25 24
2 1 28  . 23 34 42 13
2 2  . 34 33 31  . 36
```

```
2 3  3 26 28 32  4 16
3 1  .  .  1 29  . 19
3 2  . 11  9  7  1 -6
3 3 21  1  .  9  3  .
4 1 24  .  9 22 -2 15
4 2 27 12 12 -5 16 15
4 3 22  7 25  5 12  .
;
```

In the following statements, PROC GENMOD fits two classification variables and their interaction to Y. The first EFFECTPLOT statement displays the default graphic, which plots the predicted values against Disease for each of the three Drug levels. The OBS option also displays the observations in the plot. The second EFFECTPLOT statement modifies the default to plot the predicted values against Drug for each of the three Disease levels. The CLM option is specified to produce 95% confidence bars for the means.

```
ods graphics on;
proc genmod data=a;
   class drug disease;
   model y=disease drug disease*drug / d=n;
   effectplot / obs;
   effectplot interaction(sliceby=disease) / clm;
run;
ods graphics off;
```

In Output 20.2.1, the default interaction plot is produced, and the observations are also displayed. From this plot, you can compare the performance of the drugs for a given disease. The predicted values are connected with a line to provide something for your eye to follow; obviously a line has no intrinsic meaning in this graphic. Drugs 3 and 4 are consistently outperformed by the first two drugs.

Output 20.2.1 Interaction Plot: Default with Observations

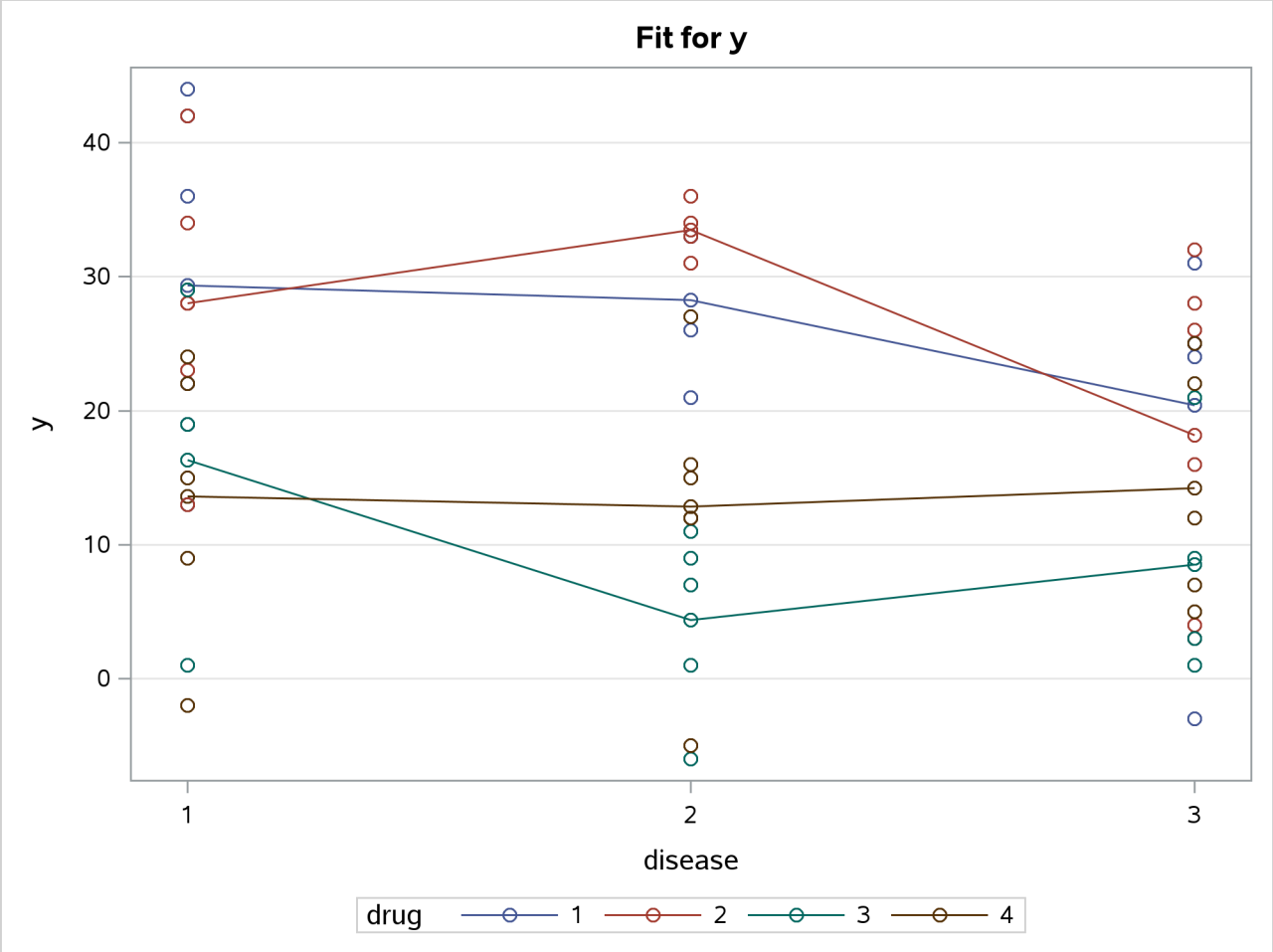

By default, the first classification variable is displayed on the X axis and the second classification variable is used for grouping. Specifying the SLICEBY=DISEASE option in the second EFFECTPLOT statement does the reverse: it displays the classification variable that has the most levels on the X axis, and slices by fewer levels, resulting in a more readable display. Output 20.2.2 shows how well a given drug performs for each disease.

Output 20.2.2 Interaction Plot with Specified SLICEBY= Effect

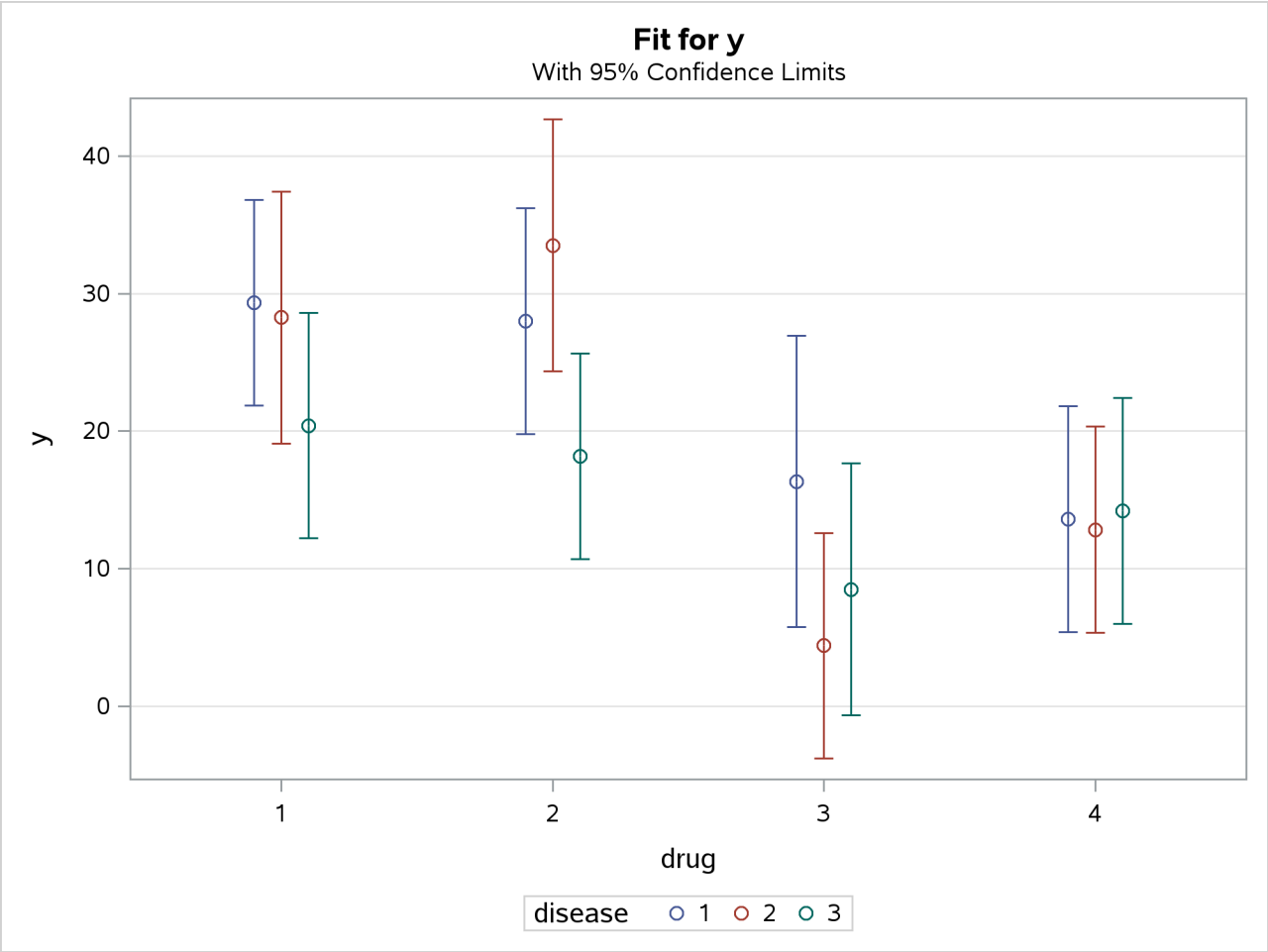

In the following statements, the BOX *plot-type* is requested to display box plots of the predictions by each combination of drug and disease. The second EFFECTPLOT statement displays the same information by using an INTERACTION *plot-type* and specifies the OBS option to display the individual observations. The third EFFECTPLOT statement displays the predictions in a MOSAIC plot. The fourth EFFECTPLOT statement creates an interaction plot of predictions versus drug for each of the Disease levels and displays them in a panel.

```
ods graphics on;
proc genmod data=a;
   class drug disease;
   model y=drug disease drug*disease / d=n;
   effectplot box;
   effectplot interaction(x=drug*disease) / obs;
   effectplot mosaic;
   effectplot interaction(plotby=disease);
run;
ods graphics off;
```

In the box plot in Output 20.2.3, the predicted values are displayed as circles; they coincide with the mean of the data at each level, all of which are displayed as diamonds. The predicted values are again connected by

lines. It is difficult to draw any conclusions from this graphic.

Output 20.2.3 Box Fit Plot

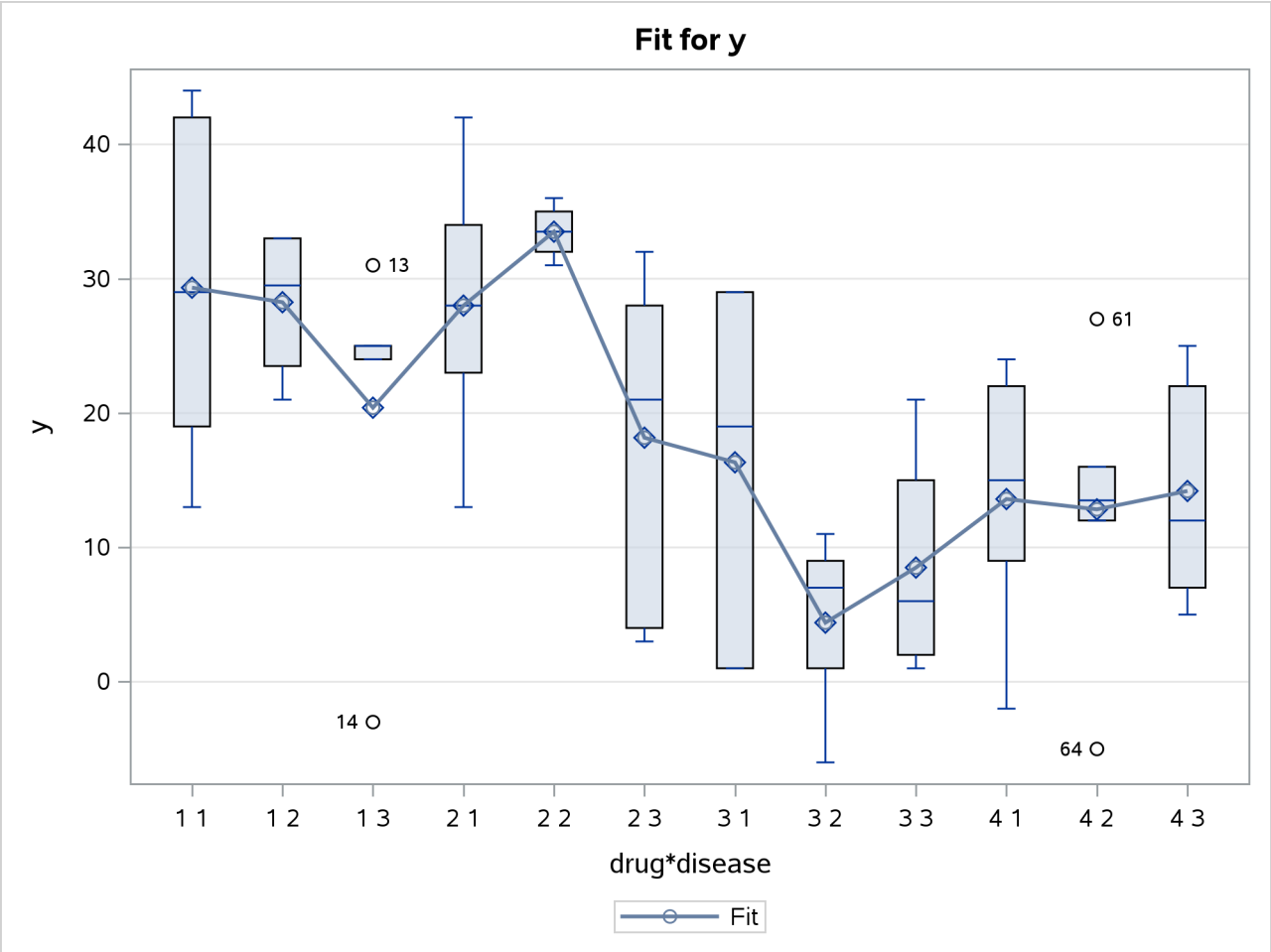

Output 20.2.4 shows the interaction plot at every combination of Drug and Disease. This plot is identical to the box plot in Output 20.2.3, except the boxes are replaced by the actual observations. Again, it is difficult to see any pattern in the plot.

Output 20.2.4 Interaction Plot with Specified X= Effect

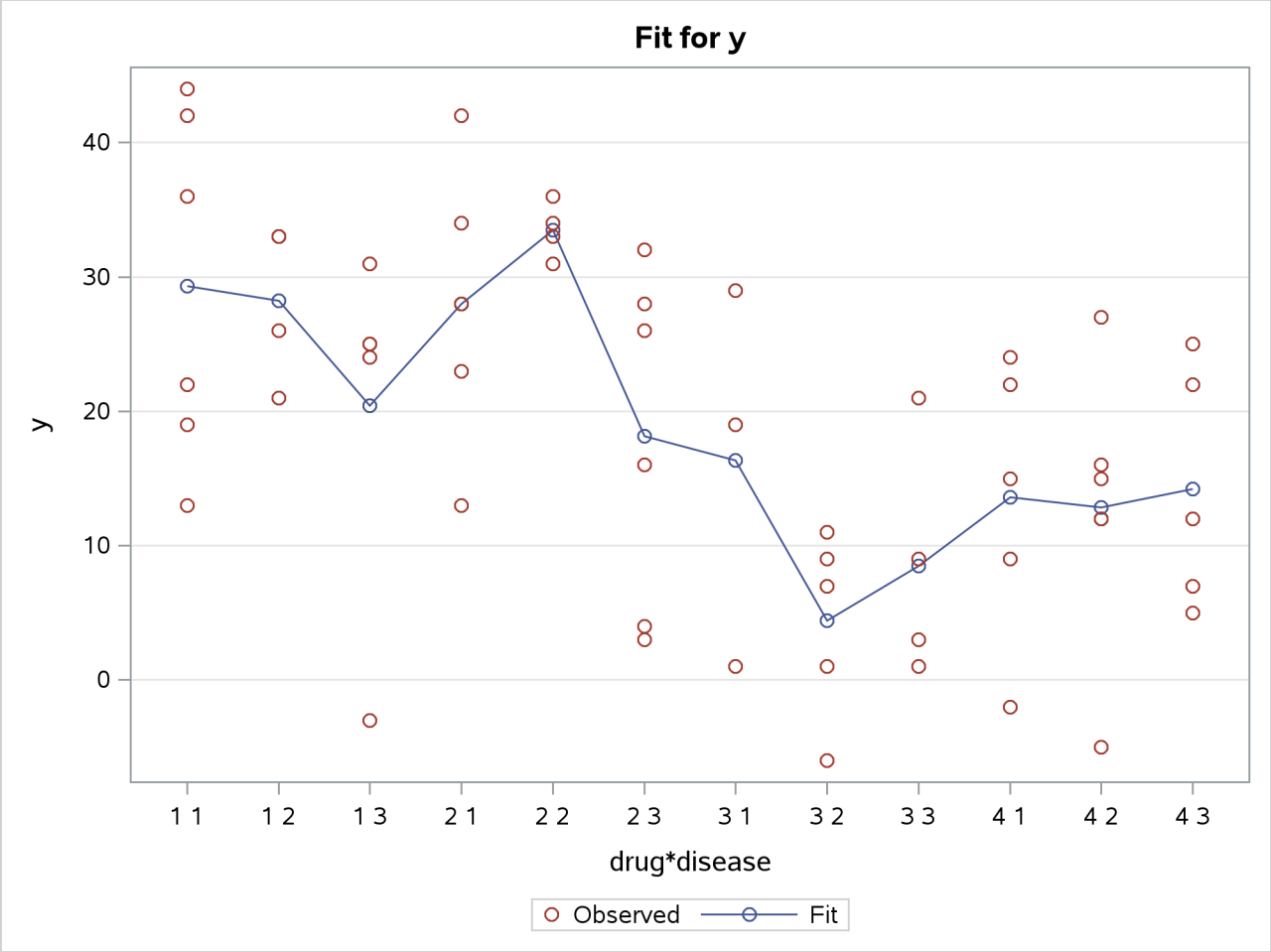

Output 20.2.5 displays the mosaic plot. You can see that drugs 1 and 2 consistently outperform the other two drugs.

Output 20.2.5 Mosaic Plot

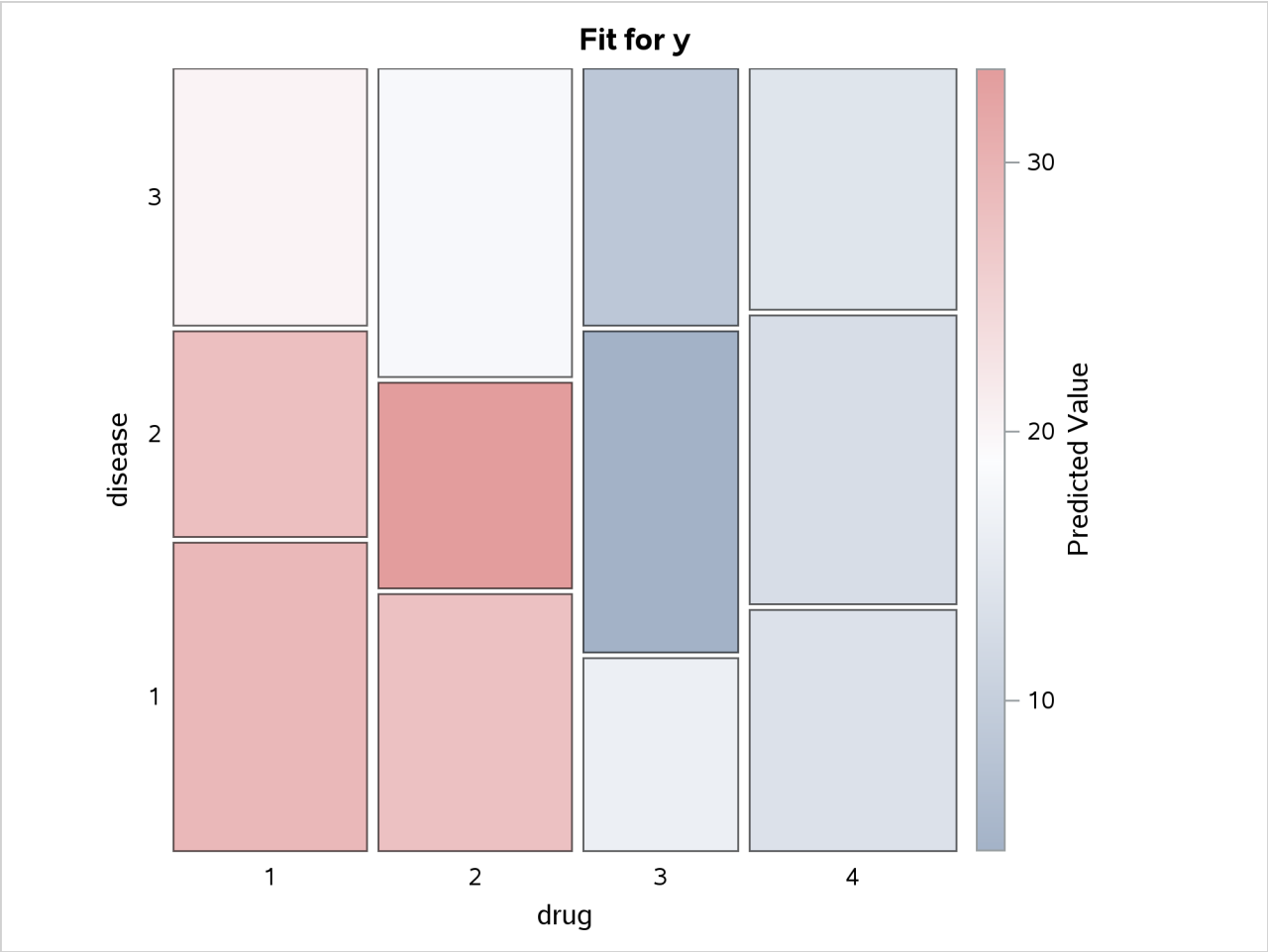

Output 20.2.6 groups the observations by Disease, and for each disease displays the effectiveness of the four drugs in a panel of plots.

Output 20.2.6 Interaction Plot with Specified PLOTBY= Effect

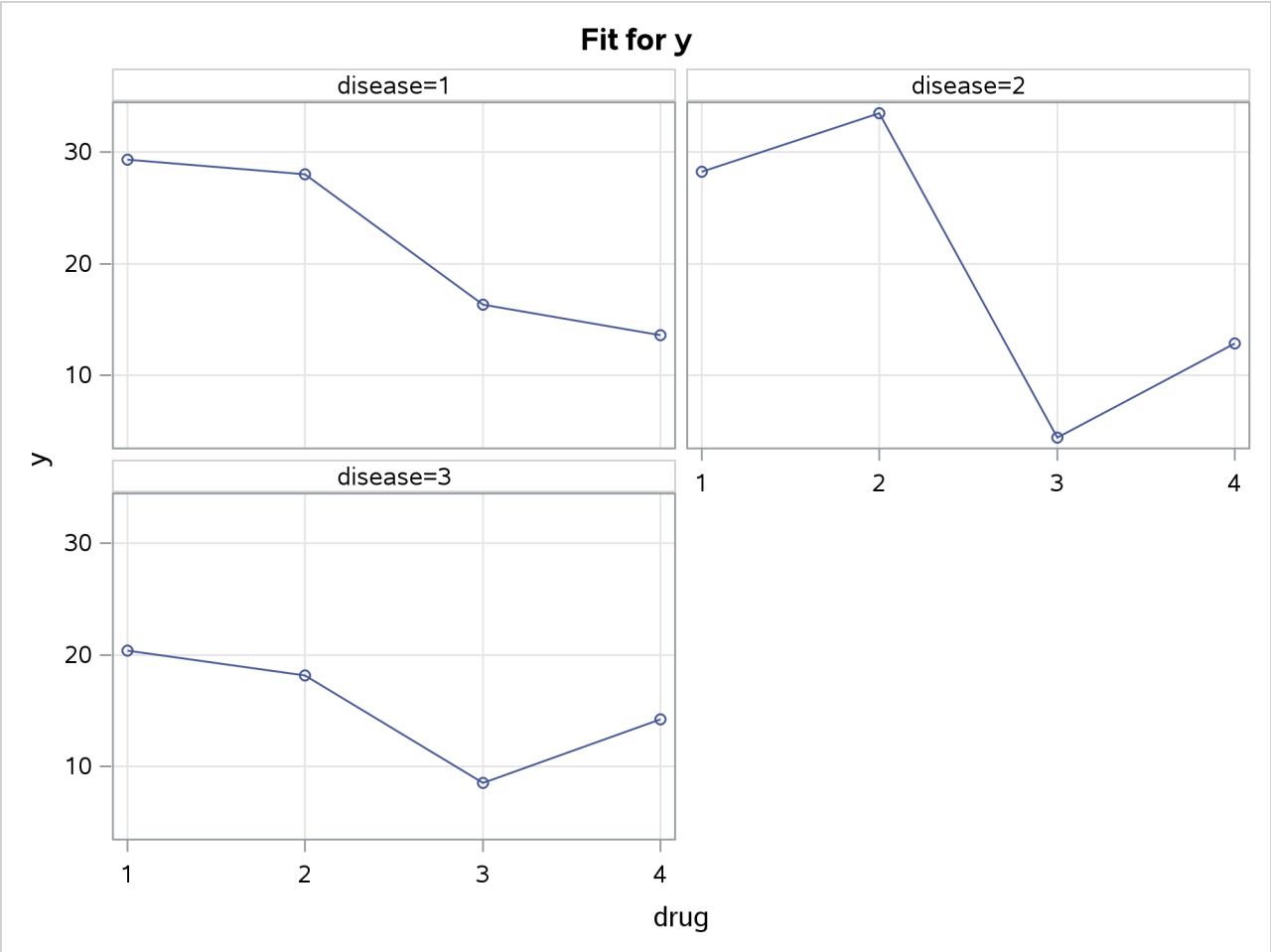

Example 20.3: Logistic Regression

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icomep3.sas.)

Consider a study of the analgesic effects of treatments on elderly patients with neuralgia. Two test treatments and a placebo are compared. The response variable is whether the patient reported pain or not. Researchers recorded the age and gender of 60 patients and the duration of complaint before the treatment began. The following DATA step creates the data set Neuralgia:

```
data Neuralgia;
   input Treatment $ Sex $ Age Duration Pain $ @@;
   datalines;
P  F  68   1  No   B  M  74  16  No  P  F  67  30  No
P  M  66  26  Yes  B  F  67  28  No  B  F  77  16  No
A  F  71  12  No   B  F  72  50  No  B  F  76   9  Yes
A  M  71  17  Yes  A  F  63  27  No  A  F  69  18  Yes
B  F  66  12  No   A  M  62  42  No  P  F  64   1  Yes
A  F  64  17  No   P  M  74   4  No  A  F  72  25  No
P  M  70   1  Yes  B  M  66  19  No  B  M  59  29  No
```

```
A  F  64  30  No   A  M  70  28  No  A  M  69   1  No
B  F  78   1  No   P  M  83   1  Yes B  F  69  42  No
B  M  75  30  Yes  P  M  77  29  Yes P  F  79  20  Yes
A  M  70  12  No   A  F  69  12  No  B  F  65  14  No
B  M  70   1  No   B  M  67  23  No  A  M  76  25  Yes
P  M  78  12  Yes  B  M  77   1  Yes B  F  69  24  No
P  M  66   4  Yes  P  F  65  29  No  P  M  60  26  Yes
A  M  78  15  Yes  B  M  75  21  Yes A  F  67  11  No
P  F  72  27  No   P  F  70  13  Yes A  M  75   6  Yes
B  F  65   7  No   P  F  68  27  Yes P  M  68  11  Yes
P  M  67  17  Yes  B  M  70  22  No  A  M  65  15  No
P  F  67   1  Yes  A  M  67  10  No  P  F  72  11  Yes
A  F  74   1  No   B  M  80  21  Yes A  F  69   3  No
;
```

The Neuralgia data set contains five variables. The Pain variable is the response. A specification of Pain=Yes indicates that the patient felt pain, and Pain=No indicates that the patient did not feel pain. The variable Treatment is a categorical variable with three levels: A and B represent the two test treatments, and P represents the placebo treatment. The gender of the patients is given by the categorical variable Sex. The variable Age is the age of the patients, in years, when treatment began. The duration of complaint, in months, before the treatment began is given by the variable Duration.

In the following statements, a complex model that includes classification and continuous covariates and an interaction term is fit to the Neuralgia data. When you try to create a default effect plot from this model, computations stop because the best type of plot cannot easily be determined.

```
ods graphics on;
proc logistic data=Neuralgia;
   class Treatment Sex / param=ref;
   model Pain= Treatment|Sex Age Duration;
   effectplot;
run;
ods graphics off;
```

To produce an effect plot for this model, you need to first choose the type of plot to be created. In this case, since there are both classification and continuous covariates on the model, a SLICEFIT *plot-type* displays the first continuous covariate (Age) on the X axis and displays fit curves that correspond to each level of the first classification covariate (Treatment). The following statements produce Output 20.3.1.

```
ods graphics on;
proc logistic data=Neuralgia;
   class Treatment Sex / param=ref;
   model Pain= Treatment|Sex Age Duration;
   effectplot slicefit;
run;
ods graphics off;
```

By default, effect plots from PROC LOGISTIC are displayed on the probability scale. The predicted values are computed at the mean of the Duration variable, 16.73, and at the reference level of the Sex variable, M. Observations are also displayed on the sliced-fit plot in Output 20.3.1. While the display of binary responses can give you a feel for the spread of the data, it does not enable you to evaluate the fit of the model.

Output 20.3.1 Default Fit Plot Sliced by Treatment

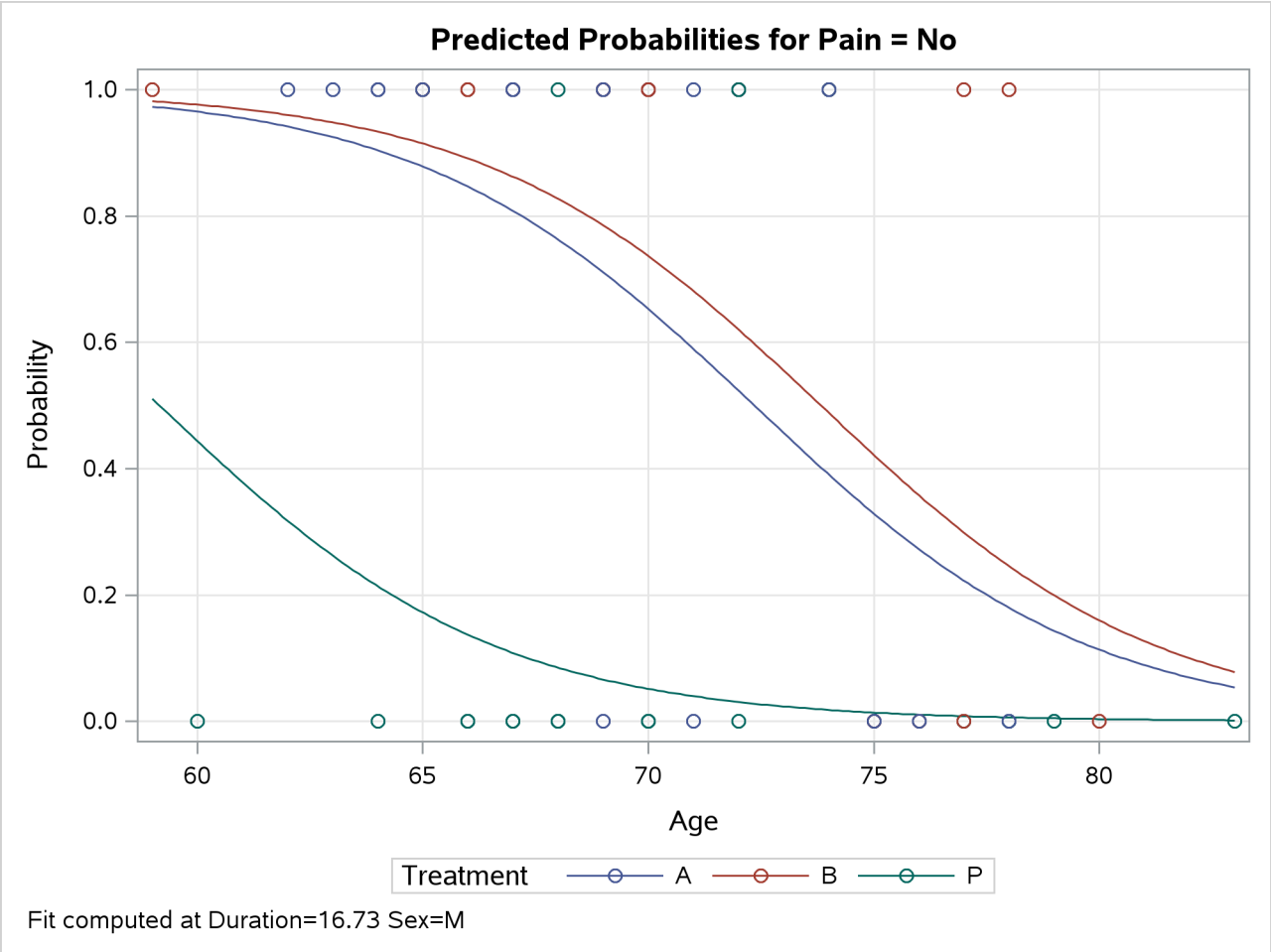

In the following statements, an INTERACTION *plot-type* is specified for the Treatment variable, with the Sex effect chosen for grouping the fits. The Age and Duration variables are set to their mean values for computing the predicted values. The NOOBS option suppresses the display of the binary observations on this plot. The LINK option is specified to display the fit on the LOGIT scale; if there is no interaction between Treatment and Sex, then the resulting curves shown in Output 20.3.2 will have similar slopes across the treatments.

```
ods graphics on;
proc logistic data=Neuralgia;
   class Treatment Sex / param=ref;
   model Pain= Treatment|Sex Age Duration;
   effectplot interaction(x=Treatment sliceby=Sex) / noobs link;
run;
ods graphics off;
```

In Output 20.3.2, the slopes of the lines seem “parallel” across the treatments, corroborating the nonsignificance of the interaction terms.

Output 20.3.2 Interaction Plot of an Interaction Effect

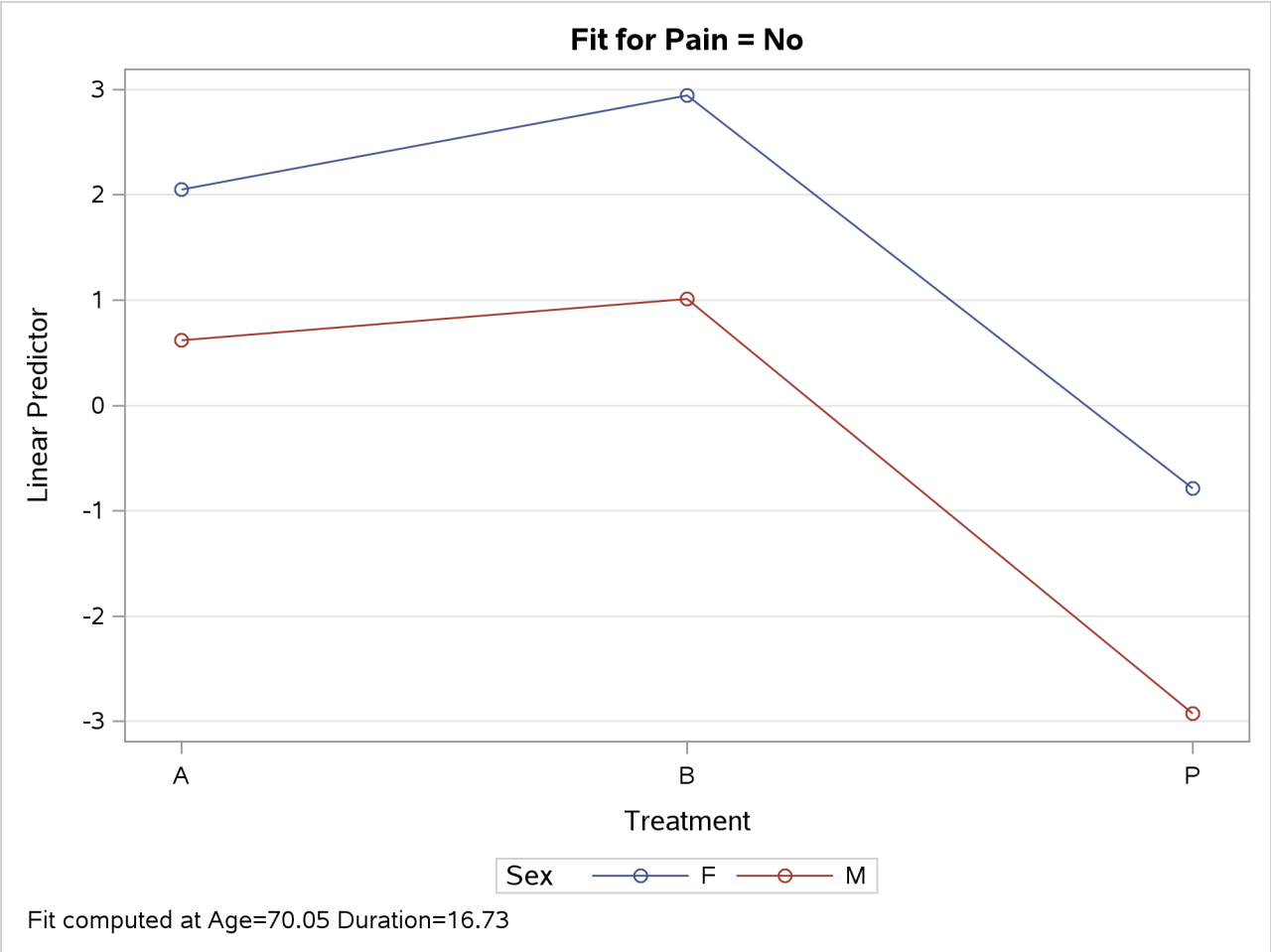

In the following statements, the interaction effect is removed, and the Duration variable is investigated further. The PLOTBY(ROWS)= option displays the Sex levels in the rows of a panel of plots, and the AT option computes the fits for several values of the Duration main effect in the columns of the panel. The OBS(FRINGE) option moves the observations to a fringe (rug) plot at the bottom of the plot, the observations are subsetted and displayed according to the value of the PLOTBY= variable, and the JITTER option makes overlaid fringes more visible. A STORE statement is also specified to save the model information for a later display. These statements produce Output 20.3.3.

```
ods graphics on;
proc logistic data=Neuralgia;
   class Treatment Sex / param=ref;
   model Pain= Treatment Sex Age Duration;
   effectplot slicefit(sliceby=Treatment plotby(rows)=Sex)
      / at(Duration=min midrange max) obs(fringe jitter(seed=39393));
   store logimodel;
run;
ods graphics off;
```

The predicted probability curves in Output 20.3.3 look very similar across the different values of the Duration variable, which agrees with the nonsignificance of Duration in this model. The fringe plot displays only

female patients in the SEX=F row of the panel and displays only male patients in the SEX=M row, because the PLOTBY=SEX option subsets the observations.

Output 20.3.3 Sliced-Fit Plot with AT Option

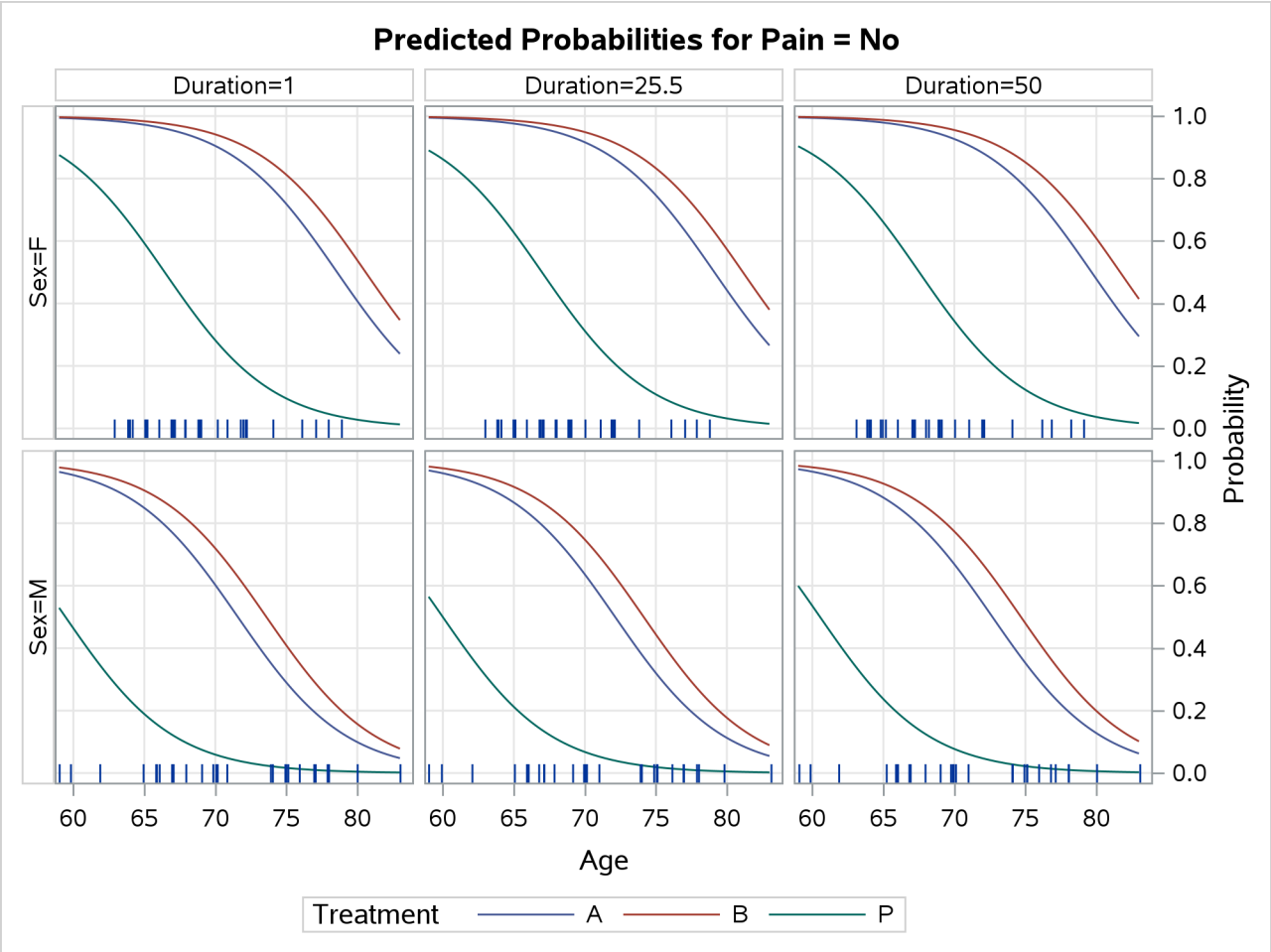

The following statements use the stored model and the PLM procedure to display a panel of contour plots:

```
ods graphics on;
proc plm restore=logimodel;
   effectplot contour(plotby=Treatment) / at(Sex=all);
run;
ods graphics off;
```

Output 20.3.4 again confirms that Duration is not significant.

Output 20.3.4 Contour Fit Panel

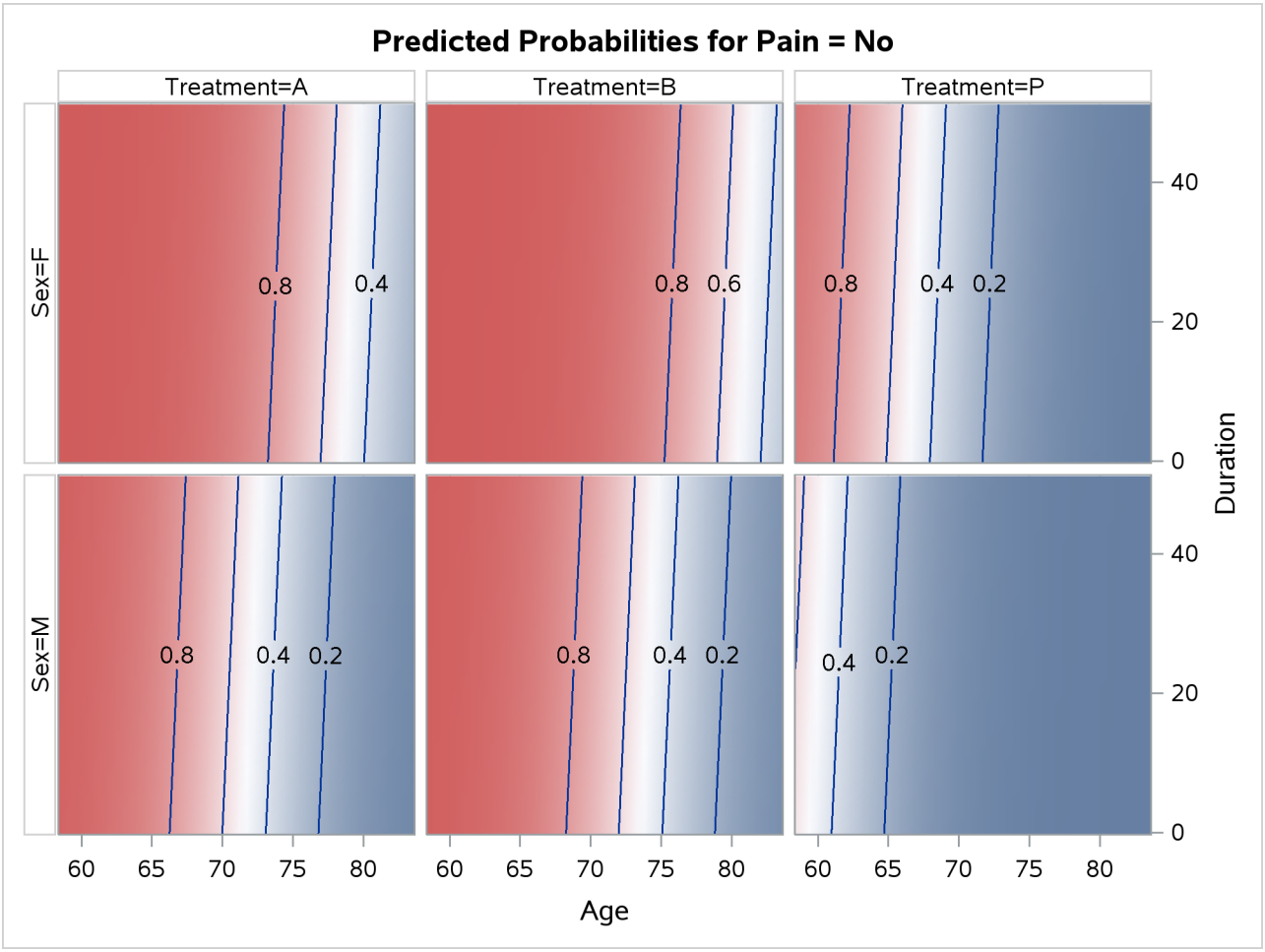

ESTIMATE Statement

This statement documentation applies to the following SAS/STAT procedures:
GEE, LIFEREG, LOGISTIC, ORTHOREG, PHREG, PLM, PROBIT, QUANTREG, RMSTREG, SURVEYLOGISTIC, SURVEYPHREG, and SURVEYREG. It also applies to the RELIABILITY procedure in SAS/QC software.
The ESTIMATE statement in the BGLIMM, GENMOD, GLIMMIX, GLM, and MIXED procedures are documented in the respective procedure chapters.

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Syntax: ESTIMATE Statement

ESTIMATE < '*label*' > *estimate-specification* < **(DIVISOR=***n***)** >
< **,** < '*label*' > *estimate-specification* < **(DIVISOR=***n***)** > > < **,** . . . >
< / *options* > **;**

The basic element of the ESTIMATE statement is the *estimate-specification*, which consists of model effects and their coefficients. An *estimate-specification* takes the general form

effect name < *effect values* . . . >

The following elements can appear in the ESTIMATE statement:

'*label*' is an optional label that identifies the particular row of the estimate in the output.

effect identifies an effect that appears in the MODEL statement. The keyword INTERCEPT can be used as an effect when an intercept is fitted in the model. You do not need to include all effects that are in the MODEL statement.

values are constants that are elements of the **L** matrix and are associated with the fixed and random effects. There are two basic methods of specifying the entries of the **L** matrix. The traditional representation—also known as the positional syntax—relies on entering coefficients in the position they assume in the **L** matrix. For example, in the following statements the elements of **L** that are associated with the b main effect receive a 1 in the first position and a –1 in the second position:

```
class a b;
model y = a b a*b;
estimate 'B at A2' b 1 -1  a*b 0  0  1 -1;
```

The elements that are associated with the interaction receive a 1 in the third position and a –1 in the fourth position. In order to specify coefficients correctly for the interaction term, you need to know how the levels of a and b vary in the interaction, which is governed by the order of the variables in the CLASS statement. For multinomial response models, you must specify the same number of values as there are parameters within a response function.

The nonpositional syntax is designed to make it easier to enter coefficients for interactions and is necessary to enter coefficients for effects that are constructed with the EFFECT statement. In square brackets you enter the coefficient followed by the associated levels of the CLASS variables. If B has two levels and A has three levels, the previous ESTIMATE statement, by using nonpositional syntax for the interaction term, becomes the following statement:

```
estimate 'B at A2' b 1 -1 a*b [1, 2 1] [-1, 2 2];
```

The previous statement assigns value 1 to the interaction where A is at level 2 and B is at level 1, and it assigns –1 to the interaction where both classification variables are at level 2. The comma that separates the entry for the **L** matrix from the level indicators is optional.

Further details about the nonpositional contrast syntax and its use with constructed effects can be found in the section "Positional and Nonpositional Syntax for Coefficients in Linear Functions" on page 486.

Based on the *estimate-specifications* in your ESTIMATE statement, the procedure constructs the matrix $\mathbf{L}$ to test the hypothesis $H: \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$. The procedure supports nonpositional syntax for the coefficients of model effects in the ESTIMATE statement. For details see the section "Positional and Nonpositional Syntax for Coefficients in Linear Functions" on page 486.

The procedure then produces for each row $\mathbf{l}$ of $\mathbf{L}$ an approximate t test of the hypothesis $H: \mathbf{l}\boldsymbol{\beta} = 0$. You can also obtain multiplicity-adjusted p-values and confidence limits for multirow estimates with the ADJUST= option.

Note that multirow estimates are permitted. The procedure constructs a default label if a label is not specified.

If the procedure finds the estimate to be nonestimable, then it displays "Non-est" for the estimate entry.

Table 20.21 summarizes important options in the ESTIMATE statement. All ESTIMATE options are subsequently discussed in alphabetical order.

Table 20.21 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1 - \alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the $\mathbf{L}$ matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers

Table 20.21 *continued*

Option	Description
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

You can specify the following options in the ESTIMATE statement after a slash (/).

ADJDFE=SOURCE

ADJDFE=ROW

specifies how denominator degrees of freedom are determined when *p*-values and confidence limits are adjusted for multiple comparisons with the ADJUST= option. When you do not specify the ADJDFE= option, or when you specify ADJDFE=SOURCE, the denominator degrees of freedom for multiplicity-adjusted results are the denominator degrees of freedom for the final effect that is listed in the ESTIMATE statement from the "Type III" table.

The ADJDFE=ROW setting is useful if you want multiplicity adjustments to take into account that denominator degrees of freedom are not constant across estimates. For example, this can be the case when the denominator degrees of freedom are computed by the Satterthwaite method or according to Kenward and Roger (1997).

The ADJDFE= option has an effect only in mixed models that use these degree-of-freedom methods. It is not supported by the procedures that perform chi-square-based inference (LOGISTIC, PHREG, and SURVEYLOGISTIC).

ADJUST=BON

ADJUST=SCHEFFE

ADJUST=SIDAK

ADJUST=SIMULATE<(*sim-options*)>

ADJUST=T

requests a multiple comparison adjustment for the *p*-values and confidence limits for the estimates. The adjusted quantities are produced in addition to the unadjusted quantities. Adjusted confidence limits are produced if the CL or ALPHA= option is in effect. For a description of the adjustments, see Chapter 53, "The GLM Procedure," and Chapter 86, "The MULTTEST Procedure," and the documentation for the ADJUST= option in the LSMEANS statement.

If the STEPDOWN option is in effect, the *p*-values are further adjusted in a step-down fashion.

ALPHA=*number*

requests that a *t* type confidence interval be constructed with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05. If the "Estimates" table shows infinite degrees of freedom, then the confidence interval is a *z* type interval.

CATEGORY=*category-options*

specifies how to construct estimates and multiplicity corrections for models that contain multinomial response data (ordinal or nominal). This option is also important for constructing sets of estimable functions for F or chi-square tests with the JOINT option.

The *category-options* are used to indicate how response variable levels are treated in constructing the estimable functions. Possible values for the *category-options* are the following:

JOINT

computes the estimable functions for nonredundant categories and displays the estimates for all these categories in a single table. For example, a three-row ESTIMATE statement in a model that has three response categories displays six estimable functions in one table. The same estimates are computed by the SEPARATE *category-option*.

SEPARATE

computes the estimable functions for nonredundant categories and displays the estimates in a separate table for each category. For example, a three-row ESTIMATE statement in a model that has three response categories displays two tables, each with three estimable functions. The same estimates are computed by the JOINT *category-option*.

quoted-value-list

computes the estimable functions only for the specified list of values. The list must consist of formatted values of the response categories, and you must specify an *estimate-specification* for each response category in the list.

Consider the following ESTIMATE statements in the LOGISTIC procedure for an ordinal model that has response categories ‘vg’, ‘g’, ‘m’, ‘b’, and ‘vb’. Because there are five response categories, there are four nonredundant categories for the cumulative link model.

```
proc logistic data=icecream;
   class brand / param=glm;
   model taste(order=data) = brand / link=logit;
   freq count;

   estimate brand 1 -1,
            intercept 1 brand  0 1 / category='m','vg';

   estimate intercept 1 brand 1    / category=joint
                                     adjust=simulate(seed=1);

   estimate brand 1 -1,
            brand 1 1 -2           / category=separate
                                     adjust=bon;
run;
```

The first ESTIMATE statement requests a two-row estimable function: the first row compares the first two brands, and the second row estimates the second brand. The result is produced for two of the four nonredundant response categories. The second ESTIMATE statement produces estimates for the first brand in all four nonredundant categories and displays them in a single table. The multiplicity adjustment with p-value computation by simulation treats the four estimable functions as a unit for familywise Type I error protection. The third ESTIMATE statement computes a two-row estimable

function: the first row again compares the first two brands, and the second row compares the average of the first two brands to the third. The result is displayed in separate tables, one for each nonredundant category. The Bonferroni adjustment in this statement applies to a family of two tests that correspond to the two-row estimable function. Four Bonferroni adjustments for sets of size two are performed.

The CATEGORY= option is supported only by the procedures that support generalized linear modeling (GEE, LOGISTIC, and SURVEYLOGISTIC) and by PROC PLM when it is used to perform statistical analyses on item stores created by these procedures.

CHISQ

requests that chi-square tests be performed in addition to F tests, when you request an F test with the JOINT option. This option has no effect in procedures that produce chi-square statistics by default.

CL

requests that t type confidence limits be constructed. If the procedure shows the degrees of freedom in the "Estimates" table as infinite, then the confidence limits are z intervals. The confidence level is 0.95 by default, and you can change the confidence level with the ALPHA= option. The confidence intervals are adjusted for multiplicity when you specify the ADJUST= option. However, if a step-down p-value adjustment is requested with the STEPDOWN option, only the p-values are adjusted for multiplicity.

CORR

displays the estimated correlation matrix of the linear combination of the parameter estimates.

COV

displays the estimated covariance matrix of the linear combination of the parameter estimates.

DF=*number*

specifies the degrees of freedom for the t test and confidence limits. This option is not supported by the procedures that perform chi-square-based inference (LOGISTIC, PHREG, and SUVEYLOGISTIC).

DIVISOR=*value-list*

specifies a list of values by which to divide the coefficients so that fractional coefficients can be entered as integer numerators. If you do not specify *value-list*, a default value of 1.0 is assumed. Missing values in the *value-list* are converted to 1.0.

If the number of elements in *value-list* exceeds the number of rows of the estimate, the extra values are ignored. If the number of elements in *value-list* is less than the number of rows of the estimate, the last value in *value-list* is copied forward.

If you specify a row-specific divisor as part of the specification of the estimate row, this value multiplies the corresponding divisor that is implied by the *value-list*. For example, the following statement divides the coefficients in the first row by 8, and the coefficients in the third and fourth row by 3:

```
estimate 'One vs. two'   A 2 -2  (divisor=2),
         'One vs. three' A 1  0 -1         ,
         'One vs. four'  A 3  0  0 -3      ,
         'One vs. five'  A 1  0  0  0  -1  / divisor=4,.,3;
```

Coefficients in the second row are not altered.

E

requests that the **L** matrix coefficients be displayed.

EXP

requests exponentiation of the estimate. When you model data with the logit, cumulative logit, or generalized logit link functions, and the estimate represents a log odds ratio or log cumulative odds ratio, the EXP option produces an odds ratio. In proportional hazards model, this option produces estimates of hazard ratios. If you specify the CL or ALPHA= option, the (adjusted) confidence bounds are also exponentiated.

The EXP option is supported only by PROC PHREG, PROC SURVEYPHREG, the procedures that support generalized linear modeling (GEE, LOGISTIC and SURVEYLOGISTIC), and by PROC PLM when it is used to perform statistical analyses on item stores created by these procedures.

ILINK

requests that the estimate and its standard error also be reported on the scale of the mean (the inverse linked scale). The computation of the inverse linked estimate depends on the estimation mode. For example, if the analysis is based on a posterior sample when a BAYES statement is present, the inversely linked estimate is the average of the inversely linked values across the sample of posterior parameter estimates. If the analysis is not based on a sample of parameter estimates, the procedure computes the value on the mean scale by applying the inverse link to the estimate. The interpretation of this quantity depends on the *effect values* specified in your ESTIMATE statement and on the link function. For example, in a model for binary data with logit link the following statements compute

$$\frac{1}{1+\exp\{-(\alpha_1-\alpha_2)\}}$$

where α_1 and α_2 are the fixed-effects solutions that are associated with the first two levels of the classification effect A:

```
class A;
model y = A / dist=binary link=logit;
estimate 'A one vs. two' A 1 -1 / ilink;
```

This quantity is not the difference of the probabilities that are associated with the two levels,

$$\pi_1-\pi_2=\frac{1}{1+\exp\{-\beta_0-\alpha_1\}}-\frac{1}{1+\exp\{-\beta_0-\alpha_2\}}$$

The standard error of the inversely linked estimate is based on the delta method. If you also specify the CL option, the procedure computes confidence limits for the estimate on the mean scale. In multinomial models for nominal data, the limits are obtained by the delta method. In other models they are obtained from the inverse link transformation of the confidence limits for the estimate. The ILINK option is specific to an ESTIMATE statement.

The ILINK option is supported only by the procedures that support generalized linear modeling (GEE, LOGISTIC and SURVEYLOGISTIC) and by PROC PLM when it is used to perform statistical analyses on item stores created by these procedures.

JOINT< (*joint-test-options*) >

requests that a joint F or chi-square test be produced for the rows of the estimate. The JOINT option in the ESTIMATE statement essentially replaces the CONTRAST statement. For multinomial response models, a joint test is constructed for each response function.

When the LOWERTAILED or the UPPERTAILED options are in effect, or if the BOUNDS option described below is in effect, the JOINT option produces the chi-bar-square statistic according to Silvapulle and Sen (2004). This statistic uses a simulation-based approach to compute p-values in situations where the alternative hypotheses of the estimable functions are not simple two-sided hypotheses. See the section "Joint Hypothesis Tests with Complex Alternatives, the Chi-Bar-Square Statistic" on page 489 for more information about this test statistic.

You can specify the following *joint-test-options* in parentheses:

ACC=γ

specifies the accuracy radius for determining the necessary sample size in the simulation-based approach of Silvapulle and Sen (2004) for tests with order restrictions. The value of γ must be strictly between 0 and 1; the default value is 0.005.

EPS=ϵ

specifies the accuracy confidence level for determining the necessary sample size in the simulation-based approach of Silvapulle and Sen (2004) for tests with order restrictions. The value of ϵ must be strictly between 0 and 1; the default value is 0.01.

LABEL='*label***'**

assigns an identifying label to the joint test. If you do not specify a label, the first nondefault label for the ESTIMATE rows is used to label the joint test.

NOEST

ONLY

performs only the F or chi-square test and suppresses other results from the ESTIMATE statement. This option is useful for emulating the CONTRAST statement that is available in other procedures.

NSAMP=*n*

specifies the number of samples for the simulation-based method of Silvapulle and Sen (2004). If *n* is not specified, it is constructed from the values of the ALPHA=α, the ACC=γ, and the EPS=ϵ options. With the default values for γ, ϵ, and α (0.005, 0.01, and 0.05, respectively), NSAMP=12,604 by default.

CHISQ

adds a chi-square test if the procedure produces an F test by default.

BOUNDS=*value-list*

specifies boundary values for the estimable linear function. The null value of the hypothesis is always zero. If you specify a positive boundary value z, the hypotheses are $H\colon \theta = 0$, $H_a\colon\colon \theta > 0$ with the added constraint that $\theta < z$. The same is true for negative boundary values. The alternative hypothesis is then $H_a\colon \theta < 0$ subject to the constraint $\theta > -|z|$. If you specify a missing value, the hypothesis is assumed to be two-sided. The BOUNDS option enables you to specify sets of one- and two-sided joint hypotheses. If all values in *value-list* are set to missing, the procedure performs a simulation-based p-value calculation for a two-sided test.

LOWER

LOWERTAILED

requests that the p-value for the t test be based only on values that are less than the test statistic. A two-tailed test is the default. A lower-tailed confidence limit is also produced if you specify the CL or ALPHA= option.

Note that for ADJUST=SCHEFFE the one-sided adjusted confidence intervals and one-sided adjusted p-values are the same as the corresponding two-sided statistics, because this adjustment is based on only the right tail of the F distribution.

If you request a joint test with the JOINT option, then a one-sided left-tailed order restriction is applied to all estimable functions, and the corresponding chi-bar-square statistic of Silvapulle and Sen (2004) is computed in addition to the two-sided, standard, F or chi-square statistic. See the JOINT option for how to control the computation of the simulation-based chi-bar-square statistic.

NOFILL

suppresses the automatic fill-in of coefficients of higher-order effects.

PLOTS=*plot-options*

produces ODS statistical graphics of the distribution of estimable functions if the procedure performs the analysis in a sampling-based mode. For example, this is the case when procedures support a BAYES statement and perform a Bayesian analysis. The estimable functions are then computed for each of the posterior parameter estimates, and the "Estimates" table reports simple descriptive statistics for the evaluated functions. The PLOTS= option enables you in this situation to visualize the distribution of the estimable function. The following *plot-options* are available:

ALL

produces all possible plots with their default settings.

BOXPLOT< (*boxplot-options*) **>**

produces box plots of the distribution of the estimable function across the posterior sample. A separate box is generated for each estimable function, and all boxes appear on a single graph by default. You can affect the appearance of the box plot graph with the following options:

ORIENTATION=VERTICAL | HORIZONTAL

ORIENT=VERT | HORIZ

specifies the orientation of the boxes. The default is vertical orientation of the box plots.

NPANELPOS=*number*

specifies how to break the series of box plots across multiple panels. If the NPANELPOS option is not specified, or if *number* equals zero, then all box plots are displayed in a single graph; this is the default. If a negative number is specified, then exactly up to |*number*| of box plots are displayed per panel. If *number* is positive, then the number of boxes per panel is balanced to achieve small variation in the number of box plots per graph.

DISTPLOT< (*distplot-options*) **>**

DIST< (*distplot-options*) **>**

generates panels of histograms with a kernel density overlaid. A separate plot in each panel contains the results for each estimable function. You can specify the following *distplot-options* in parentheses:

BOX | NOBOX
controls the display of a horizontal box plot of the estimable function's distribution across the posterior sample below the graph. The BOX option is enabled by default.

HIST | NOHIST
controls the display of the histogram of the estimable function's distribution across the posterior sample. The HIST option is enabled by default.

NORMAL | NONORMAL
controls the display of a normal density estimate on the graph. The NONORMAL option is enabled by default.

KERNEL | NOKERNEL
controls the display of a kernel density estimate on the graph. The KERNEL option is enabled by default.

NROWS=*number*
specifies the highest number of rows in a panel. The default is 3.

NCOLS=*number*
specifies the highest number of columns in a panel. The default is 3.

UNPACK
unpacks the panel into separate graphics.

NONE
does not produce any plots.

SEED=*number*
specifies the seed for the sampling-based components of the computations for the ESTIMATE statement (for example, chi-bar-square statistics and simulated p-values). The value of *number* must be an integer. The seed is used to start the pseudo-random number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. There could be multiple ESTIMATE statements with SEED= specifications and there could be other statements that can supply a random number seed. Since the procedure has only one random number stream, the initial seed is shown in the SAS log.

SINGULAR=*number*
tunes the estimability checking. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the largest absolute value of the elements of $\mathbf{v}$. If ABS($\mathbf{L} - \mathbf{LT}$) is greater than c**number* for any row of $\mathbf{L}$ in the contrast, then $\mathbf{L}\boldsymbol{\beta}$ is declared nonestimable. Here, $\mathbf{T}$ is the Hermite form matrix $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$, and c is ABS($\mathbf{L}$), except when it equals 0, and then c is 1. The value for *number* must be between 0 and 1; the default is 1E–4.

STEPDOWN< *(step-down-options)* >
requests that multiplicity adjustments for the p-values of estimates be further adjusted in a step-down fashion. Step-down methods increase the power of multiple testing procedures by taking advantage of the fact that a p-value is never declared significant unless all smaller p-values are also declared significant. The STEPDOWN adjustment combined with ADJUST=BON corresponds to the methods of Holm (1979) and "Method 2" of Shaffer (1986); this is the default. Using step-down-adjusted p-values combined with ADJUST=SIMULATE corresponds to the method of Westfall (1997).

If the ESTIMATE statement is applied with a STEPDOWN option in a mixed model where the degrees-of-freedom method is that of Kenward and Roger (1997) or of Satterthwaite, then step-down-adjusted *p*-values are produced only if the ADJDFE=ROW option is in effect.

Also, the STEPDOWN option affects only *p*-values, not confidence limits. For ADJUST=SIMULATE, the generalized least squares hybrid approach of Westfall (1997) is used to increase Monte Carlo accuracy. You can specify the following *step-down-options* in parentheses after the STEPDOWN option:

MAXTIME=*n*

specifies the time (in seconds) to be spent computing the maximal logically consistent sequential subsets of equality hypotheses for TYPE=LOGICAL. The default is MAXTIME=60. If the MAXTIME value is exceeded, the adjusted tests are not computed. When this occurs, you can try increasing the MAXTIME value. However, note that there are common multiple comparisons problems for which this computation requires a huge amount of time—for example, all pairwise comparisons between more than 10 groups. In such cases, try to use TYPE=FREE (the default) or TYPE=LOGICAL(*n*) for small *n*.

ORDER=PVALUE

ORDER=ROWS

specifies the order in which the step-down tests to be performed. ORDER=PVALUE is the default, with estimates being declared significant only if all estimates with smaller (unadjusted) *p*-values are significant. If you specify ORDER=ROWS, then significances are evaluated in the order in which they are specified in the syntax.

REPORT

specifies that a report on the step-down adjustment be displayed, including a listing of the sequential subsets (Westfall 1997) and, for ADJUST=SIMULATE, the step-down simulation results.

TYPE=LOGICAL< (*n*) >

TYPE=FREE

specifies how step-down adjustment are made. If you specify TYPE=LOGICAL, the step-down adjustments are computed by using maximal logically consistent sequential subsets of equality hypotheses (Shaffer 1986; Westfall 1997). Alternatively, for TYPE=FREE, sequential subsets are computed ignoring logical constraints. The TYPE=FREE results are more conservative than those for TYPE=LOGICAL, but they can be much more efficient to produce for many estimates. For example, it is not feasible to take logical constraints between all pairwise comparisons of more than about 10 groups. For this reason, TYPE=FREE is the default.

However, you can reduce the computational complexity of taking logical constraints into account by limiting the depth of the search tree used to compute them, specifying the optional depth parameter as a number *n* in parentheses after TYPE=LOGICAL. As with TYPE=FREE, results for TYPE=LOGICAL(*n*) are conservative relative to the true TYPE=LOGICAL results. But even for TYPE=LOGICAL(0) they can be appreciably less conservative than TYPE=FREE, and they are computationally feasible for much larger numbers of estimates. If you do not specify *n* or if *n* = –1, the full search tree is used.

TESTVALUE=*value-list*

TESTMEAN=*value-list*

specifies the value under the null hypothesis for testing the estimable functions in the ESTIMATE statement. The rules for specifying the *value-list* are very similar to those for specifying the divisor list in the DIVISOR= option. If no TESTVALUE= is specified, all tests are performed as $H: \mathbf{L}\boldsymbol{\beta} = 0$. Missing values in the *value-list* also are translated to zeros. If you specify fewer values than rows in the ESTIMATE statement, the last value in *value-list* is carried forward.

The TESTVALUE= option affects only p-values from individual, joint, and multiplicity-adjusted tests. It does not affect confidence intervals.

The TESTVALUE option is not available for the multinomial distribution, and the values are ignored when you perform a sampling-based (Bayesian) analysis.

UPPER

UPPERTAILED

requests that the p-value for the t test be based only on values that are greater than the test statistic. A two-tailed test is the default. An upper-tailed confidence limit is also produced if you specify the CL or ALPHA= option.

Note that for ADJUST=SCHEFFE the one-sided adjusted confidence intervals and one-sided adjusted p-values are the same as the corresponding two-sided statistics, because this adjustment is based on only the right tail of the F distribution.

If you request a joint test with the JOINT option, then a one-sided right-tailed order restriction is applied to all estimable functions, and the corresponding chi-bar-square statistic of Silvapulle and Sen (2004) is computed in addition to the two-sided, standard, F or chi-square statistic. See the JOINT option for how to control the computation of the simulation-based chi-bar-square statistic.

Positional and Nonpositional Syntax for Coefficients in Linear Functions

When you define custom linear hypotheses with the ESTIMATE statement, the procedure sets up an **L** vector or matrix that conforms to the model effect solutions. (Note that the following remarks also apply to the LSMESTIMATE statement, where you specify coefficients of the matrix **K** which is then converted into a coefficient matrix that conforms to the model effects solutions.)

There are two methods for specifying the entries in a coefficient matrix (hereafter simply referred to as the **L** matrix); they are called the positional and nonpositional methods. In the positional form, which is the traditional method, you provide a list of values that occupy the elements of the **L** matrix that is associated with the effect in question in the order in which the values are listed. For traditional model effects that consist of continuous and classification variables, the positional syntax is simpler in some cases (main effects) and more cumbersome in others (interactions). When you work with effects that are constructed through the EFFECT statement, the nonpositional syntax is essential.

For example, consider the following two-way model with interactions where factors A and B have three and two levels, respectively:

```
proc logistic;
   class a b;
   model y = a b a*b;
run;
```

To test the difference of the B levels at the second level of A with an ESTIMATE statement (a slice), you need to assign coefficients 1 and –1 to the levels of B and to the levels of the interaction where A is at the second level. Two examples of equivalent ESTIMATE statements that use positional and nonpositional syntax are as follows:

```
estimate 'B at A2' b 1 -1 a*b 0  0  1 -1      ;
estimate 'B at A2' b 1 -1 a*b [1 2 1] [-1 2 2];
```

Because A precedes B in the CLASS statement, the levels of the interaction are formed as $\alpha_1\beta_1, \alpha_1\beta_2, \alpha_2\beta_1, \alpha_2\beta_2, \ldots$. If B precedes A in the CLASS statement, you need to modify the coefficients accordingly:

```
proc logistic;
   class b a;
   model y = a b a*b;
   estimate 'B at A2' b 1 -1 a*b 0  1  0  0 -1     ;
   estimate 'B at A2' b 1 -1 a*b [1  1 2] [-1  2 2];
   estimate 'B at A2' b 1 -1 a*b [1, 1 2] [-1, 2 2];
run;
```

You can optionally separate the **L** value entry from the level indicators with a comma, as in the last ESTIMATE statement.

The general syntax for defining coefficients with the nonpositional syntax is as follows:

effect-name [*multiplier* <,> *level-values*] . . . < [*multiplier* <,> *level-values*] >

The first entry in square brackets is the multiplier that is applied to the elements of **L** for the effect after the *level-values* have been resolved and any necessary action that forms **L** has been taken.

The *level-values* are organized in a specific form:

- The number of entries should equal the number of terms that are needed to construct the effect. For effects that do not contain any constructed effects, this number is simply the number of terms in the name of the effect.
- Values of continuous variables that are needed for the construction of the **L** matrix precede the level indicators of CLASS variables.
- If the effect involves constructed effects, then you need to provide as many continuous and classification variables as are needed for the effect formation. For example, if a collection effect is defined as

  ```
  class c;
  effect v = collection(x1 x2 c);
  ```

  then a proper nonpositional syntax would be

```
v [0.5,  0.2 0.3 3]
```

- If an effect contains both regular terms (old-style effects) and constructed effects, then the order of the coefficients is as follows: continuous values for old-style effects, class levels for classification variables in old-style effects, continuous values for constructed effects, and finally class levels that are needed for constructed effects. Assume that C has four levels so that effect v contributes six elements to the **L** matrix. When the procedure resolves this syntax, the values 0.2 and 0.3 are assigned to the positions for x1 and x2 and a 1 is associated with the third level of C. The resulting vector is then multiplied by 0.5 to produce

  $$[0.1 \quad 0.15 \quad 0 \quad 0 \quad 0.5 \quad 0]$$

Note that you enter the **levels** of the classification variables in the square brackets, not their formatted values. The ordering of the levels of classification variables can be gleaned from the "Class Level Information" table.

To specify values for continuous variables, simply give their value as one of the terms in the effect. The nonpositional syntax in the following ESTIMATE statement is read as "1 times the value 0.4 in the column that is associated with level 2 of A"

```
proc phreg;
   class a / param=glm;
   model y = a a*x / s;
   lsmeans a / e at x=0.4;
   estimate 'A2 at x=0.4' intercept 1 a 0 1 a*x [1,0.4 2] / e;
run;
```

Because the value before the comma serves as a multiplier, the same estimable function could also be constructed with the following statements:

```
estimate 'A2 at x=0.4' intercept 1 a 0 1 a*x [ 4,  0.1 2];
estimate 'A2 at x=0.4' intercept 1 a 0 1 a*x [ 2,  0.2 2];
estimate 'A2 at x=0.4' intercept 1 a 0 1 a*x [-1, -0.4 2];
```

Note that continuous variables that are needed to construct an effect are always listed before any CLASS variables.

When you work with constructed effects, the nonpositional syntax works in the same way. For example, the following model contains a classification effect and a B-spline. The first two ESTIMATE statements produce predicted values for level 1 of C when the continuous variable x takes on the values 20 and 10, respectively.

```
proc orthoreg;
   class c;
   effect spl = spline(x / knotmethod=equal(5));
   model y = c spl;
   estimate 'C = 1 @ x=20' intercept 1 c 1 spl [1,20],
            'C = 1 @ x=10' intercept 1 c 1 spl [1,10];
   estimate 'Difference'   spl [1,20] [-1,10];
run;
```

In this example, the ORTHOREG procedure computes the spline coefficients for the first ESTIMATE statement based on $x = 20$, and similarly in the second statement for $x = 10$. The third ESTIMATE statement computes the difference of the predicted values. Because the spline effect does not interact with the

classification variable, this difference does not depend on the level of C. If such an interaction is present, you can estimate the difference in predicted values for a given level of C by using the nonpositional syntax. Because the effect C*spl contains both old-style terms (C) and a constructed effect, you specify the values for the old-style terms before assigning values to constructed effects.

```
proc orthoreg;
   class c;
   effect spl = spline(x / knotmethod=equal(5));
   model y = spl*c;
   estimate 'C2 = 1, x=20' intercept 1 c*spl [1,1 20];
   estimate 'C2 = 2, x=20' intercept 1 c*spl [1,2 20];
   estimate 'C diff at x=20' c*spl [1,1 20] [-1,2 20];
run;
```

It is recommended that you add the E option to the ESTIMATE or LSMESTIMATE statement to verify that the **L** matrix is formed according to your expectations.

In any row of an ESTIMATE statement you can choose positional and nonpositional syntax separately for each effect. However, you cannot mix the two forms of syntax for coefficients of a single effect. For example, the following statement is not proper because both forms of syntax are used for the interaction effect:

```
estimate 'A1B1 - A1B2' b 1 -1  a*b 0 1  [-1, 1 2];
```

Joint Hypothesis Tests with Complex Alternatives, the Chi-Bar-Square Statistic

Silvapulle and Sen (2004) propose a test statistic for testing hypotheses where the null or the alternative hypothesis or both involve inequalities. You can test special cases of these hypotheses with the JOINT option in the ESTIMATE and the LSMESTIMATE statement. Consider the k estimable functions $\mathbf{L}\boldsymbol{\beta}$ and the hypotheses $H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ and $H_a\colon \mathbf{L}\boldsymbol{\beta} \geq \mathbf{0}$. The alternative hypothesis defines a convex cone $\mathcal{C}$ at the origin. Suppose that under the null hypothesis $\mathbf{L}\widehat{\boldsymbol{\beta}}$ follows a multivariate normal distribution with mean $\mathbf{0}$ and variance $\mathbf{V}$. The restricted alternative prevents you from using the usual F or chi-square test machinery, since the distribution of the test statistic under the alternative might not follow the usual rules. Silvapulle and Sen (2004) coined a statistic that takes into account the projection of the observed estimate onto the convex cone formed by the alternative parameter space. This test statistic is called the chi-bar-square statistic, and p-values are obtained by simulation; see, in particular, Chapter 3.4 in Silvapulle and Sen (2004).

Briefly, let $\mathbf{U}$ be a multivariate normal random variable with mean $\mathbf{0}$ and variance matrix $\mathbf{V}$. The chi-bar-square statistic is the random variable

$$\begin{aligned} \overline{\chi}^2 &= \mathbf{U}'\mathbf{V}^{-1}\mathbf{U} - Q \\ Q &= \min_{\boldsymbol{\theta}\in C}(\mathbf{U}-\boldsymbol{\theta})'\mathbf{V}^{-1}(\mathbf{U}-\boldsymbol{\theta}) \end{aligned}$$

and it can be motivated by a geometric argument. The quadratic form in Q is the $\mathbf{V}$-projection of $\mathbf{U}$ onto the cone $\mathcal{C}$. Suppose that this projected point is $\tilde{\mathbf{U}}$. If $\mathbf{U} \in \mathcal{C}$, then $Q = 0$ and $\tilde{\mathbf{U}} = \mathbf{U}$. If $\mathbf{U}$ is completely outside of the cone $\mathcal{C}$, then $\tilde{\mathbf{U}}$ is a point on the surface of the cone. Similarly, $\mathbf{U}'\mathbf{V}^{-1}\mathbf{U}$ is the length of the segment from the origin to $\mathbf{U}$ in the $\mathbf{V}$-space with norm $||x|| = (\mathbf{x}'\mathbf{V}^{-1}\mathbf{x})^{1/2}$. If you apply the Pythagorean theorem, you can see that the chi-bar-square statistic measures the length of the segment from the origin to the projected point $\tilde{\mathbf{U}}$ in $\mathcal{C}$.

To calculate p-values for chi-bar-square statistics, a simulation-based approach is taken. Consider again the set of k estimable functions $\mathbf{L}\boldsymbol{\beta}$ with estimate $\mathbf{L}\widehat{\boldsymbol{\beta}} = \mathbf{U}$ and variance $\mathbf{L}\mathrm{Var}[\hat{\boldsymbol{\beta}}]\mathbf{L}' = \mathbf{V}$.

First, the observed value of the statistic is computed as

$$\overline{\chi}^2_{obs} = \mathbf{U}'\mathbf{V}^{-1}\mathbf{U} - Q$$

Then, n independent random samples $\mathbf{Z}_1, \ldots, \mathbf{Z}_n$ are drawn from an $N(\mathbf{0}, \mathbf{V})$ distribution and the following chi-bar-statistics are computed for the sample:

$$\begin{aligned}
\overline{\chi}^2_1 &= \mathbf{Z}_1'\mathbf{V}^{-1}\mathbf{Z}_1 - \min_{\boldsymbol{\theta}\in C}(\mathbf{Z}_1 - \boldsymbol{\theta})'\mathbf{V}^{-1}(\mathbf{Z}_1 - \boldsymbol{\theta}) \\
&\vdots \\
\overline{\chi}^2_n &= \mathbf{Z}_n'\mathbf{V}^{-1}\mathbf{Z}_n - \min_{\boldsymbol{\theta}\in C}(\mathbf{Z}_n - \boldsymbol{\theta})'\mathbf{V}^{-1}(\mathbf{Z}_n - \boldsymbol{\theta})
\end{aligned}$$

The p-value is estimated by the fraction of simulated statistics that are greater than or equal to the observed value $\overline{\chi}^2_{obs}$.

Notice that unless $\mathbf{U}$ is interior to the cone $\mathcal{C}$, finding the value of Q requires the solution to a quadratic optimization problem. When k is large, or when many simulations are requested, the computation of p-values for chi-bar-square statistics might require considerable computing time.

ODS Table Names: ESTIMATE Statement

Each table created by the ESTIMATE statement has a name associated with it, and you can use this name to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 20.22. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 20.22 ODS Tables Produced by the ESTIMATE Statement

Table Name	Description	Required Option
Coef	**L** matrix coefficients	E
Estimates	ESTIMATE statement results	Default
Contrasts	Joint test results	JOINT

ODS Graphics: ESTIMATE Statement

This section describes the use of ODS Graphics for creating statistical graphs of the distribution of estimable functions with the ESTIMATE statement. The plots can be produced only in association with the LIFEREG and PHREG procedures, which can perform Bayesian analysis. The plots are available via these procedures directly, and also via PROC PLM when it is run using an item store that was created by these procedures.

To request these graphs you must do the following:

- ensure that ODS Graphics is enabled
- use a BAYES statement with PROC LIFEREG or PROC PHREG, or use PROC PLM to perform statistical analysis on an item store that was saved from a Bayesian analysis
- request plots with the PLOTS= option in the ESTIMATE statement

For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." The available graphs are summarized in Table 20.23.

Table 20.23 Graphs Produced by the ESTIMATE Statement

ODS Graph Name	Plot Description	Required Option
BoxPlot	Displays box plots of estimable functions across a posterior sample.	PLOTS=BOXPLOT
DistPanel	Displays panels of histograms with kernel density curves overlaid. Each plot contains the results for the posterior sample of each estimable function.	PLOTS=DISTPLOT
DistPlot	Displays a histogram with a kernel density curve overlaid. The plot contains the results for the posterior sample of the estimable function.	PLOTS=DISTPLOT(UNPACK)

For details about the *plot-options* of the ESTIMATE statement, see the PLOTS= option in the section "ESTIMATE Statement" on page 475.

LSMEANS Statement

This statement documentation applies to the following procedures:
GEE, GENMOD, LIFEREG, LOGISTIC, ORTHOREG, PHREG, PLM, PROBIT, RMSTREG, SURVEYLOGISTIC, SURVEYPHREG, and SURVEYREG. It also applies to the RELIABILITY procedure in SAS/QC software.
The GLIMMIX, GLM, and MIXED procedures also support LSMEANS statements. The relevant statement documentation for these procedures can be found in the specific procedure chapter.

The LSMEANS statement computes least squares means (LS-means) of fixed effects. In the GLM, MIXED, and GLIMMIX procedures, LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Thus it is important not to interpret the name with a strict association with least squares estimation. Least squares is the predominant estimation technique for the type of models in which LS-means were first applied. Their interpretation and importance reaches beyond the least squares principle, however. A more appropriate approach to LS-means views them as linear combinations of the parameter estimates that are constructed in such a way that they correspond to average predicted values in a population where the levels of classification variables are balanced.

This contemporary—and historically correct—interpretation of the concept of least squares means underlines their importance in all classes of models where predicted values are reasonably formed as linear combinations of the parameter estimates. LS-means distinguish themselves from general estimable functions in that they take the structure for the model and data into account through the structure of the $\mathbf{X}$ and $\mathbf{X}'\mathbf{X}$ matrix in your model. For example, in a generalized linear model the structure of the $\mathbf{X}$ matrix informs the analysis about the possible levels of classification variables and predictions on the linear (the linked) scale are computed as $\mathbf{x}'\boldsymbol{\beta}$. LS-means are thus meaningful quantities in such models when the linear estimable function that corresponds to an averaged prediction is constructed on the linked scale. For example, in a binomial model with logit link, the least squares means are predicted population margins of the logits. You can then transform the least squares means to the data scale with the ILINK option, and you can display differences of least squares means in terms of odds ratios with the ODDSRATIO option. The underlying principle—unless you perform a Bayesian analysis—is to construct the estimates or their differences on the linked scale and to apply appropriate transformations in a second step.

Least squares means computations are also supported for multinomial models.

LS-means are computed as $\mathbf{L}\boldsymbol{\beta}$ where the $\mathbf{L}$ matrix that is constructed to compute the predicted values is the same as the $\mathbf{L}$ matrix that is formed in PROC GLM.

Each LS-mean is computed as $\mathbf{L}\widehat{\boldsymbol{\beta}}$, where $\mathbf{L}$ is the coefficient matrix that is associated with the least squares mean and $\widehat{\boldsymbol{\beta}}$ is the estimate of the fixed-effects parameter vector. The approximate standard error for the LS-mean is computed as the square root of $\mathbf{L}\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}]\mathbf{L}'$. The approximate variance matrix of the fixed-effects estimates depends on the estimation method.

Syntax: LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

LS-means can be computed for any effect in the statistical model that involves only CLASS variables. You can specify multiple effects in one LSMEANS statement or in multiple LSMEANS statements, and all LSMEANS statements must appear after the MODEL statement. If you do not specify *model-effects*, the options in the LSMEANS statement are applied to all suitable model effects.

As in the ESTIMATE statement, the **L** matrix is tested for estimability; if this test fails, the procedure displays "Non-est" for the LS-means entries. Note that linear functions of LS-means, such as differences, can be estimable, even if the means themselves are not estimable. Estimability checks for differences are thus applied separately from checks for the means.

Assuming the LS-mean is estimable, the procedure constructs an approximate t test to test the null hypothesis that the associated population quantity equals zero.

Table 20.24 summarizes important options in the LSMEANS statement. All LSMEANS options are subsequently discussed in alphabetical order.

Table 20.24 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion

Table 20.24 *continued*

Option	Description
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

You can specify the following options in the LSMEANS statement after a slash (/):

ADJDFE=ROW

ADJDFE=SOURCE

specifies how denominator degrees of freedom are determined when *p*-values and confidence limits are adjusted for multiple comparisons with the ADJUST= option. When you do not specify the ADJDFE= option or when you specify ADJDFE=SOURCE, the denominator degrees of freedom for multiplicity-adjusted results are the denominator degrees of freedom for the LS-mean effect in the "Type III Tests of Fixed Effects" table. When you specify ADJDFE=ROW, the denominator degrees of freedom for multiplicity-adjusted results correspond to the degrees of freedom that are displayed in the DF column of the "Differences of Least Squares Means" table.

The ADJDFE=ROW setting is particularly useful if you want multiplicity adjustments to take into account that denominator degrees of freedom are not constant across LS-mean differences.

In one-way models with heterogeneous variance, combining certain ADJUST= options with the ADJDFE=ROW option corresponds to particular methods of performing multiplicity adjustments in the presence of heteroscedasticity. For example, the following statements fit a heteroscedastic one-way model and perform Dunnett's T3 method (Dunnett 1980), which is based on the studentized maximum modulus (ADJUST=SMM):

```
proc glimmix;
   class A;
   model y = A / ddfm=satterth;
   random _residual_ / group=A;
   lsmeans A / adjust=smm adjdfe=row;
run;
```

If you combine the ADJDFE=ROW option with ADJUST=SIDAK, the multiplicity adjustment corresponds to the T2 method of Tamhane (1979), and ADJUST=TUKEY corresponds to the method of Games-Howell (Games and Howell 1976). Note that ADJUST=TUKEY gives the exact results for the case of fractional degrees of freedom in the one-way model, but it does not take into account that the degrees of freedom are subject to variability. A more conservative method, such as ADJUST=SMM, might protect the overall error rate better.

Unless the ADJUST= option is specified in the LSMEANS statement, the ADJDFE= option has no effect. The option is not supported by the procedures that perform chi-square-based inference (GENMOD, LOGISTIC, PHREG, and SURVEYLOGISTIC).

ADJUST=BON

ADJUST=DUNNETT

ADJUST=NELSON

ADJUST=SCHEFFE

ADJUST=SIDAK

ADJUST=SIMULATE<(*sim-options*)>

ADJUST=SMM | GT2

ADJUST=TUKEY

requests a multiple comparison adjustment for the *p*-values and confidence limits for the differences of LS-means. The adjusted quantities are produced in addition to the unadjusted quantities. By default, the procedure performs all pairwise differences. If you specify ADJUST=DUNNETT, the procedure analyzes all differences with a control level. If you specify ADJUST=NELSON, ANOM differences are taken. The ADJUST= option implies the DIFF option.

The BON (Bonferroni) and SIDAK adjustments involve correction factors described in Chapter 53, "The GLM Procedure," and Chapter 86, "The MULTTEST Procedure"; also see Westfall and Young (1993) and Westfall et al. (1999). When you specify ADJUST=TUKEY and your data are unbalanced, the procedure uses the approximation described in Kramer (1956) and identifies the adjustment as "Tukey-Kramer" in the results. Similarly, when you specify ADJUST=DUNNETT or ADJUST=NELSON and the LS-means are correlated, the procedure uses the factor-analytic covariance approximation described in Hsu (1992) and identifies the adjustment in the results as "Dunnett-Hsu" or "Nelson-Hsu," respectively. The approximation derives an approximate "effective sample sizes" for which exact critical values are computed. Computing the exact adjusted *p*-values and critical values for unbalanced designs can be computationally intensive, in particular for ADJUST=NELSON. A simulation-based approach, as specified by the ADJUST=SIM option, while nondeterministic, can provide inferences that are sufficiently accurate in much less time. The preceding references also describe the SCHEFFE and SMM adjustments.

Nelson's adjustment applies only to the analysis of means (Ott 1967; Nelson 1982, 1991, 1993), where LS-means are compared against an average LS-mean. It does not apply to all pairwise differences of

least squares means. See the DIFF=ANOM option for more details regarding the analysis of means with the procedure.

The SIMULATE adjustment computes adjusted p-values and confidence limits from the simulated distribution of the maximum or maximum absolute value of a multivariate t random vector. All covariance parameters, except the residual scale parameter, are fixed at their estimated values throughout the simulation, potentially resulting in some underdispersion. The simulation estimates q, the true $(1-\alpha)$ quantile, where $1-\alpha$ is the confidence coefficient. The default α is 0.05, and you can change this value with the ALPHA= option in the LSMEANS statement.

The number of samples is set so that the tail area for the simulated q is within γ of $1-\alpha$ with $100(1-\epsilon)\%$ confidence. In equation form,

$$\Pr(|F(\widehat{q})-(1-\alpha)| \leq \gamma) = 1-\epsilon$$

where $\hat{q}$ is the simulated q and F is the true distribution function of the maximum; see Edwards and Berry (1987) for details. By default, $\gamma = 0.005$ and $\epsilon = 0.01$, placing the tail area of $\hat{q}$ within 0.005 of 0.95 with 99% confidence. You can specify the following *sim-options* in parentheses after the ADJUST=SIMULATE option:

ACC=*value*
specifies the target accuracy radius γ of a $100(1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. The default value is ACC=0.005.

EPS=*value*
specifies the value ϵ for a $100 \times (1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. The default value for the accuracy confidence is 99%, which corresponds to EPS=0.01.

NSAMP=*n*
specifies the sample size for the simulation. By default, *n* is set by a formula that is based on the values of the target accuracy radius γ and accuracy confidence $100 \times (1-\epsilon)\%$ for an interval for the true probability content of the estimated $(1-\alpha)$ quantile. With the default values for γ, ϵ, and α (0.005, 0.01, and 0.05, respectively), NSAMP=12,604 by default. If you omit the NSAMP= option or specify NSAMP=0, the default sample size is used.

SEED=*number*
specifies an integer that is used to start the pseudo-random number generator for the simulation. If you do not specify a seed, or specify a value less than or equal to zero, the seed is by default generated from reading the time of day from the computer's clock.

THREADS
specifies that the computational work for the simulation be divided into parallel threads, where the number of threads is the value of the SAS system option CPUCOUNT=. For large simulations (as specified directly using the NSAMP= *sim-option* or indirectly using the ACC= or EPS= *sim-options*), parallel processing can markedly speed up the computation of adjusted p-values and confidence intervals. However, because the parallel processing has different pseudo-random number streams, the precise results are different from the default ones, which are computed in sequence rather than in parallel. This option overrides the SAS system option THREADS | NOTHREADS.

NOTHREADS

specifies that the computational work for the simulation be performed in sequence rather than in parallel. NOTHREADS is the default. This option overrides the SAS system option THREADS | NOTHREADS.

If the STEPDOWN option is in effect, the p-values are further adjusted in a step-down fashion. For certain options and data, this adjustment is exact under an iid $N(0, \sigma^2)$ model for the dependent variable, in particular for the following:

- for ADJUST=DUNNETT when the means are uncorrelated
- for ADJUST=TUKEY with STEPDOWN(TYPE=LOGICAL) when the means are balanced and uncorrelated.

The first case is a consequence of the nature of the successive step-down hypotheses for comparisons with a control; the second uses an extension of the maximum studentized range distribution appropriate for partition hypotheses (Royen 1989). Finally, for STEPDOWN(TYPE=FREE), ADJUST=TUKEY employs the Royen (1989) extension in such a way that the resulting p-values are conservative.

ALPHA=*number*

requests that a t type confidence interval be constructed for each of the LS-means with confidence level $1 - \mathit{number}$. The value of *number* must be between 0 and 1; the default is 0.05.

ASC

ASCENDING

sorts LS-means in ascending order in results that you display by specifying the LINES option. When you specify the ASCENDING option, LS-means that are displayed in graphical and tabular form are also sorted in ascending order.

AT *variable***=***value*

AT (*var-list***)=(***value-list***)**

AT MEANS

modifies the values of the covariates that are used in computing LS-means. By default, all covariate effects are set equal to their mean values for computation of standard LS-means. The AT option enables you to assign arbitrary values to the covariates. Additional columns in the output table indicate the values of the covariates.

If there is an effect that contains two or more covariates, the AT option sets the effect equal to the product of the individual means rather than the mean of the product (as with standard LS-means calculations). The AT MEANS option sets covariates equal to their mean values (as with standard LS-means) and incorporates this adjustment to crossproducts of covariates.

As an example, consider the following statements:

```
class A;
model Y = A x1 x2 x1*x2;
lsmeans A;
lsmeans A / at means;
lsmeans A / at x1=1.2;
lsmeans A / at (x1 x2)=(1.2 0.3);
```

For the first two LSMEANS statements, the LS-means coefficient for x1 is $\overline{x}_1$ (the mean of x1) and for x2 is $\overline{x}_2$ (the mean of x2). However, for the first LSMEANS statement, the coefficient for x1*x2 is $\overline{x_1 x_2}$, but for the second LSMEANS statement, the coefficient is $\overline{x}_1 \times \overline{x}_2$. The third LSMEANS statement sets the coefficient for x1 equal to 1.2 and leaves it at $\overline{x}_2$ for x2, and the final LSMEANS statement sets these values to 1.2 and 0.3, respectively.

Even if you specify a WEIGHT variable, the unweighted covariate means are used for the covariate coefficients if there is no AT specification. If you specify the AT option, WEIGHT or FREQ variables are taken into account as follows. The weighted covariate means are then used for the covariate coefficients for which no explicit AT values are given, or if you specify AT MEANS. Observations that do not contribute to the analysis because of a missing dependent variable are included in computing the covariate means. Use the E option in conjunction with the AT option to check that the modified LS-means coefficients are the ones you want.

The AT option is disabled if you specify the BYLEVEL option.

BYLEVEL

requests that separate margins be computed for each level of the LSMEANS effect.

The standard LS-means have equal coefficients across classification effects. The BYLEVEL option changes these coefficients to be proportional to the observed margins. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in the input data set. In this case, the resulting LS-means are actually equal to raw means for fixed-effects models and certain balanced random-effects models, but their estimated standard errors account for the covariance structure that you have specified. If a WEIGHT statement is specified, the procedure uses weighted margins to construct the LS-means coefficients.

If the AT option is specified, the BYLEVEL option disables it.

CL

requests that *t* type confidence limits be constructed for each of the LS-means. The confidence level is 0.95 by default; this can be changed with the ALPHA= option. If you specify an ADJUST= option, then the confidence limits are adjusted for multiplicity. But if you also specify STEPDOWN, then only *p*-values are step-down adjusted, not the confidence limits.

CORR

displays the estimated correlation matrix of the least squares means as part of the "Least Squares Means" table.

COV

displays the estimated covariance matrix of the least squares means as part of the "Least Squares Means" table.

DF=*number*

specifies the degrees of freedom for the *t* test and confidence limits. The default is the denominator degrees of freedom taken from the "Type III Tests" table that corresponds to the LS-means effect. The option is not supported by the procedures that perform chi-square-based inference (GENMOD, LOGISTIC, PHREG and SURVEYLOGISTIC).

DIFF<=*difftype*>

PDIFF<=*difftype*>

requests that differences of the LS-means be displayed. You can use one of the following optional *difftype* values to specify which differences to produce:

ALL

requests all pairwise differences; this is the default.

ANOM

requests differences between each LS-mean and the average LS-mean, as in the *analysis of means* (Ott 1967). The average is computed as a weighted mean of the LS-means, the weights being inversely proportional to the diagonal entries of the $\mathbf{L}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{L}'$ matrix. If LS-means are nonestimable, this design-based weighted mean is replaced with an equally weighted mean. Note that the ANOM procedure in SAS/QC software implements both tables and graphics for the analysis of means with a variety of response types. For one-way designs and normal data with identity link, the DIFF=ANOM computations are equivalent to the results of PROC ANOM. If the LS-means being compared are uncorrelated, exact adjusted p-values and critical values for confidence limits can be computed in the analysis of means; see Nelson (1982, 1991, 1993) and Guirguis and Tobias (2004) in addition to the documentation for the ADJUST=NELSON option.

CONTROL

requests differences with a control, which, by default, is the first valid level of each of the specified LSMEANS effects. For example, suppose the effects A and B are classification variables, both of them have two levels 1 and 2, and the A=1, B=1 cell is missing. Unless the procedure supports a MISSING option in the CLASS statement and the option is in effect, the following LSMEANS statement uses the level (1,2) of A*B as the control:

```
lsmeans A*B / diff=control;
```

Nevertheless, you can still specify a valid level as the control—for example, (2,1) of A*B. To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the CONTROL keyword. For example, if the effects A, B, and C are classification variables, each having two levels, 1 and 2, the following LSMEANS statement specifies the (1,2) level of A*B and the (2,1) level of B*C as controls:

```
lsmeans A*B B*C / diff=control('1' '2', '2' '1');
```

For multiple effects, the results depend upon the order of the list, and so you should check the output to make sure that the controls are correct.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*.

CONTROLL

tests whether the noncontrol levels are significantly smaller than the control; the upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing.

CONTROLU

tests whether the noncontrol levels are significantly larger than the control; the upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

If you want to perform multiple comparison adjustments on the differences of LS-means, you must specify the ADJUST= option.

The differences of the LS-means are displayed in a table titled "Differences of Least Squares Means."

E

requests that the **L** matrix coefficients for the LSMEANS effects be displayed.

EXP

requests exponentiation of the LS-means or LS-mean differences. When you model data with the logit, cumulative logit, or generalized logit link functions, and the estimate represents a log odds ratio or log cumulative odds ratio, the EXP option produces an odds ratio. In proportional hazards model, the exponentiation of the LS-mean differences produces estimates of hazard ratios. If you specify the CL or ALPHA= option, the (adjusted) confidence bounds are also exponentiated.

The EXP option is supported only by PROC PHREG, PROC SURVEYPHREG, PROC RMSTREG, the procedures that support generalized linear modeling (GEE, GENMOD, LOGISTIC, and SURVEYLOGISTIC), and PROC PLM when it is used to perform statistical analyses on item stores that are created by these procedures.

ILINK

requests that estimates and their standard errors in the "Least Squares Means" table also be reported on the scale of the mean (the inverse linked scale). This enables you to obtain estimates of predicted probabilities and their standard errors in logistic models, for example. The option is specific to an LSMEANS statement. If you also specify the CL option, the procedure computes confidence intervals for the predicted means by applying the inverse link transform to the confidence limits on the linked (linear) scale. Standard errors on the inverse linked scale are computed by the delta method.

The ILINK option is supported only by the procedures that support generalized linear modeling (GEE, GENMOD, LOGISTIC and SURVEYLOGISTIC) and by PROC PLM when it is used to perform statistical analyses on item stores that are created by these procedures.

LINES

presents results of comparisons between all pairs of least squares means by listing the LS-means in descending order and indicating nonsignificant subsets by line segments beside the corresponding LS-means. When all differences have the same variance, these comparison lines are guaranteed to accurately reflect the inferences that are based on the corresponding tests, which compare the respective p-values to the value of the ALPHA= option (0.05 by default). However, equal variances might not be the case for differences between LS-means. If the variances are not all the same, then the comparison lines might be conservative, in the sense that if you base your inferences on the lines alone, you will detect fewer significant differences than the tests indicate. If there are any such differences, the procedure lists the pairs of LS-means that are inferred to be significantly different by the tests but not by the comparison lines. However, even though the variances in many cases are unequal, they are similar enough that the comparison lines accurately reflect the test inferences.

If ODS Graphics is enabled, then the LINES display is produced in graphical form by default. To display it in tabular form, use the LINESTABLE option.

LINESTABLE

presents the results that are described in the LINES option as a table.

MEANS | NOMEANS

determines whether to print the least squares means themselves. For most procedure, MEANS is the default behavior. For example, the NOMEANS option is the default for the PHREG procedure. You can then use the MEANS option to produce the table of least squares means, if desired.

ODDSRATIO

OR

requests that LS-mean differences (DIFF, ADJUST= options) are also reported in terms of odds ratios. The ODDSRATIO option is ignored unless you use either the logit, cumulative logit, or generalized logit link function. If you specify the CL or ALPHA= option, confidence intervals for the odds ratios are also computed. These intervals are adjusted for multiplicity when you specify the ADJUST= option.

The ODDSRATIO option is supported only by the procedures that support generalized linear modeling (GEE, GENMOD, LOGISTIC and SURVEYLOGISTIC) and by PROC PLM when it is used to perform statistical analyses on item stores created by these procedures.

OBSMARGINS< =*OM-data-set* **>**

OM< =*OM-data-set* **>**

specifies a potentially different weighting scheme for the computation of LS-means coefficients. The standard LS-means have equal coefficients across classification effects; however, the OM option changes these coefficients to be proportional to those found in the *OM-data-set*. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins that are observed in *OM-data-set*.

By default, *OM-data-set* is the same as the analysis data set. You can optionally specify another data set that describes the population for which you want to make inferences. This data set must contain all model variables except for the dependent variable (which is ignored if it is present). In addition, the levels of all CLASS variables must be the same as those that occur in the analysis data set. If a level of a classification effect in the original data set is not present in the *OM-data-set*, the LS-means for that level are undefined. The corresponding rows of the LSMeans table are displayed as missing. Specifying an *OM-data-set* enables you to construct arbitrarily weighted LS-means.

In computing the observed margins, the procedure uses all observations for which there are no missing or invalid independent variables, including those for which there are missing dependent variables. Also, if you use a WEIGHT statement, the procedure computes weighted margins to construct the LS-means coefficients. If your data are balanced, the LS-means are unchanged by the OM option.

The BYLEVEL option modifies the observed-margins LS-means. Instead of computing the margins across all of the *OM-data-set*, the procedure computes separate margins for each level of the LSMEANS effect in question. In this case the resulting LS-means are actually equal to raw means for fixed-effects models and certain balanced random-effects models, but their estimated standard errors account for the covariance structure that you have specified.

You can use the E option in conjunction with either the OM or BYLEVEL option to verify that the modified LS-means coefficients are the ones you want. It is possible that the modified LS-means are not estimable when the standard ones are estimable, or vice versa.

PDIFF
is the same as the DIFF option.

PLOT | PLOTS< =*plot-request*< (*options*) > >

PLOT | PLOTS< =(*plot-request*< (*options*) > < . . . *plot-request*< (*options*) > >) >
requests that graphics related to least squares means be produced via ODS Graphics, provided that ODS Graphics is enabled and the *plot-request* does not conflict with other options in the LSMEANS statement. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

The available options and suboptions are as follows:

ALL
requests that the default plots that correspond to this LSMEANS statement be produced. The default plot depends on the options in the statement.

ANOMPLOT

ANOM
requests an analysis-of-means display in which least squares means are compared to an average least squares mean. Least squares mean ANOM plots are produced only for those model effects that are listed in LSMEANS statements and have options that do not contradict with the display. For example, the following statements produce analysis-of-mean plots for effects A and C:

```
lsmeans A / diff=anom plot=anom;
lsmeans B / diff      plot=anom;
lsmeans C /           plot=anom;
```

The DIFF option in the second LSMEANS statement implies all pairwise differences.

BOXPLOT< *boxplot-options* >
produces box plots of the distribution of the least squares mean or least squares mean differences across a posterior sample. For example, this plot is available in procedures that support a Bayesian analysis through the BAYES statement.

A separate box is generated for each estimable function, and all boxes appear on a single graph by default. You can affect the appearance of the box plot graph with the following options:

ORIENTATION=VERTICAL | HORIZONTAL

ORIENT=VERT | HORIZ
specifies the orientation of the boxes. The default is vertical orientation of the box plots.

NPANELPOS=*number*
specifies how to break the series of box plots across multiple panels. If the NPANELPOS option is not specified, or if *number* equals zero, then all box plots are displayed in a single graph; this is the default. If a negative number is specified, then exactly up to |*number*| of box plots are displayed per panel. If *number* is positive, then the number of boxes per panel is balanced to achieve small variation in the number of box plots per graph.

CONTROLPLOT

CONTROL

requests a display in which least squares means are visually compared against a reference level. These plots are produced only for statements with options that are compatible with control differences. For example, the following statements produce control plots for effects A and C:

```
lsmeans A / diff=control('1') plot=control;
lsmeans B / diff              plot=control;
lsmeans C                     plot=control;
```

The DIFF option in the second LSMEANS statement implies all pairwise differences.

DIFFPLOT< (*diffplot-options*) >

DIFFOGRAM< (*diffplot-options*) >

DIFF< (*diffplot-options*) >

requests a display of all pairwise least squares mean differences and their significance. The display is also known as a "mean-mean scatter plot" when it is based on arithmetic means (Hsu 1996; Hsu and Peruggia 1994). For each comparison a line segment, centered at the LS-means in the pair, is drawn. The length of the segment corresponds to the projected width of a confidence interval for the least squares mean difference. Segments that fail to cross the 45-degree reference line correspond to significant least squares mean differences.

LS-mean difference plots are produced only for statements with options that are compatible with the display. For example, the following statements request differences against a control level for the A effect, all pairwise differences for the B effect, and the least squares means for the C effect:

```
lsmeans A / diff=control('1') plot=diff;
lsmeans B / diff              plot=diff;
lsmeans C                     plot=diff;
```

The DIFF= type in the first statement is incompatible with a display of all pairwise differences.

You can specify the following *diffplot-options*:

ABS

determines the positioning of the line segments in the plot. This is the default *diffplot-options*. When the ABS option is in effect, all line segments are shown on the same side of the reference line.

NOABS

determines the positioning of the line segments in the plot. The NOABS option separates comparisons according to the sign of the difference.

CENTER

marks the center point for each comparison. This point corresponds to the intersection of two least squares means.

NOLINES
suppresses the display of the line segments that represent the confidence bounds for the differences of the least squares means. The NOLINES option implies the CENTER option. The default is to draw line segments in the upper portion of the plot area without marking the center point.

DISTPLOT< *distplot-options* >

DIST< *distplot-options* >
generates panels of histograms with a kernel density overlaid if the analysis has access to a set of posterior parameter estimates. For example, this plot is available in procedures that support a Bayesian analysis through the BAYES statement. A separate plot in each panel contains the results for each least squares mean or least squares mean differences. You can specify the following *distplot-options* in parentheses:

BOX | NOBOX
controls the display of a horizontal box plot of the estimable function's distribution across the posterior sample below the graph. The BOX option is enabled by default.

HIST | NOHIST
controls the display of the histogram of the estimable function's distribution across the posterior sample. The HIST option is enabled by default.

NORMAL | NONORMAL
controls the display of a normal density estimate on the graph. The NONORMAL option is enabled by default.

KERNEL | NOKERNEL
controls the display of a kernel density estimate on the graph. The KERNEL option is enabled by default.

NROWS=*number*
specifies the highest number of rows in a panel. The default is 3.

NCOLS=*number*
specifies the highest number of columns in a panel. The default is 3.

UNPACK
unpacks the panel into separate graphics.

LINESPLOT< **(WSCALE=***wfactor* **HSCALE=***hfactor***)** >
modifies the dimensions of the LS-means comparison plot that is produced by the LINES option. The default dimensions of the plot vary depending on aspects such as the number of groups, the number of CLASS variables in the effect, and the number of parallel lines that are needed to represent comparisons. You can change the defaults by specifying the following options:

HSCALE=*hfactor*
scales the default height of the plot by *hfactor*, which must be a positive number. For example, specifying HSCALE=2 makes the plot twice as high as it would be by default.

WSCALE=*wfactor*
scales the default width of the plot by *wfactor*, which must be a positive number. For example, specifying WSCALE=2 makes the plot twice as wide as it would be by default.

MEANPLOT<(*meanplot-options***)>**
requests displays of the least squares means.

The following *meanplot-options* control the display of the least squares means.

ASCENDING
displays the least squares means in ascending order. This option has no effect if means are displayed in separate plots.

CL
displays upper and lower confidence limits for the least squares means. By default, 95% limits are drawn. You can change the confidence level with the ALPHA= option. Confidence limits are drawn by default if the CL option is specified in the LSMEANS statement.

CLBAND
displays confidence limits as bands. This option implies the JOIN option.

DESCENDING
displays the least squares means in descending order. This option has no effect if means are displayed in separate plots.

ILINK
requests that means (and confidence limits) be displayed on the inverse linked scale.

JOIN
CONNECT
connects the least squares means with lines. This option is implied by the CLBAND option. If the effect contains nested variables and a SLICEBY= effect contains classification variables that appear as crossed effects, this option is ignored.

SLICEBY=*fixed-effect*
specifies an effect by which to group the means in a single plot. For example, the following statement requests a plot in which the levels of A are placed on the horizontal axis and the means that belong to the same level of B are joined by lines:

```
lsmeans A*B / plot=meanplot(sliceby=b join);
```

Unless the LS-mean effect contains at least two classification variables, the SLICEBY= option has no effect. The *fixed-effect* does not have to be an effect in your MODEL statement, but it must consist entirely of classification variables and it must be contained in the LS-mean effect.

PLOTBY=*fixed-effect*

specifies an effect by which to break interaction plots into separate displays. For example, the following statement requests for each level of C one plot of the A*B cell means that are associated with that level of C:

```
lsmeans A*B*C / plot=meanplot(sliceby=b plotby=c clband);
```

In each plot, levels of A are displayed on the horizontal axis, and confidence bands are drawn around the means that share the same level of B.

The PLOTBY= option has no effect unless the LS-mean effect contains at least three classification variables. The *fixed-effect* does not have to be an effect in the MODEL statement, but it must consist entirely of classification variables and it must be contained in the LS-mean effect.

NONE

requests that no plots be produced.

When LS-mean calculations are adjusted for multiplicity by using the ADJUST= option, the plots are adjusted accordingly.

SEED=*number*

specifies the seed for the sampling-based components of the computations for the LSMEANS statement (for example, chi-bar-square statistics and simulated p-values). The value of *number* must be an integer. The seed is used to start the pseudo-random-number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. Note that there could be multiple LSMEANS statements with SEED= specifications and there could be other statements that can supply a random number seed. Since the procedure has only one random number stream, the initial seed is shown in the SAS log.

SINGULAR=*number*

tunes the estimability checking. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the largest absolute value of the elements of $\mathbf{v}$. If ABS($\mathbf{K}' - \mathbf{K}'\mathbf{T}$) is greater than c*$number$ for any row of $\mathbf{K}'$ in the contrast, then $\mathbf{K}'\boldsymbol{\beta}$ is declared nonestimable. Here, $\mathbf{T}$ is the Hermite form matrix $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$, and c is ABS($\mathbf{K}'$), except when it equals 0, and then c is 1. The value for *number* must be between 0 and 1; the default is 1E–4.

STEPDOWN< (*step-down-options***) >**

requests that multiple comparison adjustments for the p-values of LS-mean differences be further adjusted in a step-down fashion. Step-down methods increase the power of multiple comparisons by taking advantage of the fact that a p-value is never declared significant unless all smaller p-values are also declared significant. The STEPDOWN adjustment combined with ADJUST=BON corresponds to the methods of Holm (1979) "Method 2" of Shaffer (1986); this is the default. Using step-down-adjusted p-values combined with ADJUST=SIMULATE corresponds to the method of Westfall (1997).

If the denominator degrees of freedom are computed by the Kenward-Roger (Kenward and Roger 1997) or Satterthwaite method in a mixed model, then step-down-adjusted p-values are produced only if the ADJDFE=ROW option is in effect.

Also, STEPDOWN affects only p-values, not confidence limits. For ADJUST=SIMULATE, the generalized least squares hybrid approach of Westfall (1997) is used to increase Monte Carlo accuracy.

You can specify the following *step-down-options* in parentheses:

MAXTIME=*n*

specifies the time (in seconds) to be spent computing the maximal logically consistent sequential subsets of equality hypotheses for TYPE=LOGICAL. The default is MAXTIME=60. If the MAXTIME value is exceeded, the adjusted tests are not computed. When this occurs, you can try increasing the MAXTIME value. However, note that there are common multiple comparisons problems for which this computation requires a huge amount of time—for example, all pairwise comparisons between more than 10 groups. In such cases, try to use TYPE=FREE (the default) or TYPE=LOGICAL(*n*) for small *n*.

REPORT

specifies that a report on the step-down adjustment be displayed, including a listing of the sequential subsets (Westfall 1997) and, for ADJUST=SIMULATE, the step-down simulation results.

TYPE=LOGICAL< (*n*) >

TYPE=FREE

specifies how step-down adjustment are made. If you specify TYPE=LOGICAL, the step-down adjustments are computed by using maximal logically consistent sequential subsets of equality hypotheses (Shaffer 1986; Westfall 1997). Alternatively, for TYPE=FREE, sequential subsets are computed ignoring logical constraints. The TYPE=FREE results are more conservative than those for TYPE=LOGICAL, but they can be much more efficient to produce for many comparisons. For example, it is not feasible to take logical constraints between all pairwise comparisons of more than 10 groups. For this reason, TYPE=FREE is the default.

However, you can reduce the computational complexity of taking logical constraints into account by limiting the depth of the search tree used to compute them, specifying the optional depth parameter as a number n in parentheses after TYPE=LOGICAL. As with TYPE=FREE, results for TYPE=LOGICAL(n) are conservative relative to the true TYPE=LOGICAL results. But even for TYPE=LOGICAL(0) they can be appreciably less conservative than TYPE=FREE, and they are computationally feasible for much larger numbers of comparisons. If you do not specify n or if $n = -1$, the full search tree is used.

Construction of Least Squares Means

To construct a least squares mean (LS-mean) for a particular level of a particular effect, construct a row vector **L** according to the following rules and use it in an ESTIMATE statement to compute the value of the LS-mean:

1. Set all L_i that correspond to covariates (continuous variables) to their mean value.
2. Consider effects that are contained by the particular effect. (For a definition of *containing*, see Chapter 16, "The Four Types of Estimable Functions.") Set the L_i that correspond to levels associated with the particular level equal to 1. Set all other L_i in these effects equal to 0.
3. Consider the particular effect. Set the L_i that correspond to the particular level equal to 1. Set the L_i that correspond to other levels equal to 0.

4. Consider the effects that contain the particular effect. If these effects are not nested within the particular effect, then set the L_i that correspond to the particular level to $1/k$, where k is the number of such columns. If these effects are nested within the particular effect, then set the L_i that correspond to the particular level to $1/(k_1 k_2)$, where k_1 is the number of nested levels within this combination of nested effects and k_2 is the number of such combinations. For L_i that correspond to other levels, use 0.

5. Consider other effects that are not yet considered. For each effect that has no nested factors, set all L_i that correspond to this effect to $1/j$, where j is the number of levels in the effect. For each effect that has nested factors, set all L_i that correspond to this effect to $1/(j_1 j_2)$, where j_1 is the number of nested levels within a particular combination of nested effects and j_2 is the number of such combinations.

The consequence of these rules is that the sum of the Xs within any classification effect is 1. This set of Xs forms a linear combination of the parameters that is checked for estimability before it is evaluated.

For example, consider the following model:

```
proc orthoreg;
   class A B C;
   model Y=A B A*B C Z;
   lsmeans A B A*B C;
run;
```

Assume A has 3 levels, B has 2 levels, and C has 2 levels, and assume that every combination of levels of A and B exists in the data. Assume also that Z is a continuous variable with an average of 12.5. Then the least squares means are computed by the following linear combinations of the parameter estimates:

		A			B		A*B						C		
	μ	1	2	3	1	2	11	12	21	22	31	32	1	2	Z
LSM( )	1	1/3	1/3	1/3	1/2	1/2	1/6	1/6	1/6	1/6	1/6	1/6	1/2	1/2	12.5
LSM(A1)	1	1	0	0	1/2	1/2	1/2	1/2	0	0	0	0	1/2	1/2	12.5
LSM(A2)	1	0	1	0	1/2	1/2	0	0	1/2	1/2	0	0	1/2	1/2	12.5
LSM(A3)	1	0	0	1	1/2	1/2	0	0	0	0	1/2	1/2	1/2	1/2	12.5
LSM(B1)	1	1/3	1/3	1/3	1	0	1/3	0	1/3	0	1/3	0	1/2	1/2	12.5
LSM(B2)	1	1/3	1/3	1/3	0	1	0	1/3	0	1/3	0	1/3	1/2	1/2	12.5
LSM(AB11)	1	1	0	0	1	0	1	0	0	0	0	0	1/2	1/2	12.5
LSM(AB12)	1	1	0	0	0	1	0	1	0	0	0	0	1/2	1/2	12.5
LSM(AB21)	1	0	1	0	1	0	0	0	1	0	0	0	1/2	1/2	12.5
LSM(AB22)	1	0	1	0	0	1	0	0	0	1	0	0	1/2	1/2	12.5
LSM(AB31)	1	0	0	1	1	0	0	0	0	0	1	0	1/2	1/2	12.5
LSM(AB32)	1	0	0	1	0	1	0	0	0	0	0	1	1/2	1/2	12.5
LSM(C1)	1	1/3	1/3	1/3	1/2	1/2	1/6	1/6	1/6	1/6	1/6	1/6	1	0	12.5
LSM(C2)	1	1/3	1/3	1/3	1/2	1/2	1/6	1/6	1/6	1/6	1/6	1/6	0	1	12.5

Setting Covariate Values

By default, all covariate effects are set equal to their mean values for computation of standard LS-means. The AT option in the LSMEANS statement enables you to set the covariates to whatever values you consider interesting.

If any effect contains two or more covariates, the AT option sets the effect equal to the product of the individual means rather than the mean of the product (as with standard LS-means calculations). The AT MEANS option leaves covariates equal to their mean values (as with standard LS-means) and incorporates this adjustment to crossproducts of covariates.

For example, the following statements specify a model with a classification variable A and two continuous variables, x1 and x2:

```
class A;
model y = A x1 x2 x1*x2;
```

The coefficients for the continuous effects with various AT specifications are shown in the following table.

Syntax	x1	x2	x1*x2
`lsmeans A;`	$\overline{x_1}$	$\overline{x_2}$	$\overline{x_1 x_2}$
`lsmeans A / at means;`	$\overline{x_1}$	$\overline{x_2}$	$\overline{x_1} \cdot \overline{x_2}$
`lsmeans A / at x1=1.2;`	1.2	$\overline{x_2}$	$1.2 \cdot \overline{x_2}$
`lsmeans A / at (x1 x2)=(1.2 0.3);`	1.2	0.3	$1.2 \cdot 0.3$

For the first two LSMEANS statements, the A LS-mean coefficient for x1 is $\overline{x_1}$ (the mean of x1) and for x2 is $\overline{x_2}$ (the mean of x2). However, the coefficient for x1*x2 is $\overline{x_1 x_2}$ for the first LSMEANS statement, but it is $\overline{x_1} \cdot \overline{x_2}$ for the second LSMEANS statement. The third LSMEANS statement sets the coefficient for x1 equal to 1.2 and leaves the coefficient for x2 at $\overline{x_2}$, and the final LSMEANS statement sets these values to 1.2 and 0.3, respectively.

The covariate means used in computing the LS-means are affected by the AT option in the LSMEANS statement as follows:

- If you use a WEIGHT statement:
  - If you do not specify the AT option, unweighted covariate means are used for the covariate coefficients.
  - If you specify the AT option, weighted covariate means are used for the covariate coefficients for which no explicit AT values are specified (thus for all the covariate means if you specify AT MEANS).
- If any observations have missing dependent variable values:
  - If you do not specify the AT option, these observations are not included in computing the covariate means.
  - If you specify the AT option, these observations are included unless they form a missing cell (a combination of CLASS variables all of whose responses are missing).

These conditions are summarized in Table 20.25. You can use the E option in conjunction with the AT option to verify that the modified LS-means coefficients are the ones you want.

Table 20.25 Treatment of Covariate Means with and without the AT Option

	Without AT	With AT
WEIGHT statement	Weights not used	Weights used
Observations with missing dependent variable values	Observations not included	Observations included

The AT option is disabled if you specify the BYLEVEL option, in which case the coefficients for the covariates are set equal to their means within each level of the LS-mean effect in question.

Changing the Weighting Scheme

The standard LS-means have equal coefficients across classification effects; however, the OBSMARGINS option in the LSMEANS statement changes these coefficients to be proportional to those found in the specified data set, or in the input data set if you do not specify a data set for observed margins. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in the specified data set.

The computation of observed margins uses all observations for which there are no missing independent variables, including those for which there are missing dependent variables. Also, if there is a WEIGHT variable, weighted margins are used to construct the LS-means coefficients. If the observed margins data set is balanced or if you specify a simple one-way model, the LS-means will be unchanged by the OBSMARGINS option.

The BYLEVEL option modifies the observed-margins LS-means. Instead of computing the margins across the entire data set, separate margins are computed for each level of the LS-mean effect in question. The resulting LS-means are actually equal to raw means in this case. The BYLEVEL option disables the AT option if it is specified.

You might want to use the E option in conjunction with either the OBSMARGINS or BYLEVEL option to verify that the modified LS-means coefficients are the ones you want. It is possible that the modified LS-means are not estimable when the standard ones are, or vice versa.

Estimability of LS-Means

LS-means are defined as certain linear combinations of the parameters. As such, it is possible for them to be inestimable. In fact, it is possible for a pair of LS-means to be both inestimable but their difference estimable. When this happens, only the entries that correspond to the estimable difference are computed and displayed in the Diffs table. If ADJUST=SIMULATE is specified when there are inestimable LS-means differences, adjusted results for all differences are displayed as missing.

ODS Table Names: LSMEANS Statement

Each table created by the LSMEANS statement has a name associated with it, and you can use this name to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 20.26. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 20.26 ODS Tables Produced by the LSMEANS Statement

Table Name	Description	Required Option
Coef	**L** matrix coefficients	E
Diffs	Differences of LS-means	DIFF or ADJUST= or STEPDOWN
LSMeans	LS-means	Default
LSMLines	Lines display for LS-means	LINES

ODS Graphics: LSMEANS Statement

This section describes the use of ODS Graphics for creating graphics that are related to LS-means in procedures that support the common LSMEANS statement. There are two groups of available plots: those that can be produced by all procedures that support this statement, and those that can be produced only in association with the three procedures that can perform Bayesian analysis (PROC GENMOD, PROC LIFEREG, and PROC PHREG). Plots that are associated with the Bayesian analysis are available via these procedures directly, and also by using PROC PLM with an item store that was created by these procedures.

Plots in the first group depict the LS-means and their differences; when LS-mean comparisons are adjusted for multiplicity by using the ADJUST= option, the plots are adjusted accordingly. To request plots in this group, ODS Graphics must be enabled and you must request plots with the appropriate PLOTS= option in the LSMEANS statement. Plots in the second group depict the posterior sample distribution of LS-means and their differences. To request plots in this group, you must also use a BAYES statement with PROC GENMOD, PROC PHREG, or PROC LIFEREG, or you must use PROC PLM to perform statistical analysis on an item store that was saved from a Bayesian analysis.

For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." The available graphs are summarized in Table 20.27 and Table 20.28.

Table 20.27 Graphs Produced by All Procedures That Support the Common LSMEANS Statement

ODS Graph Name	Plot Description	Required Option
AnomPlot	Requests an analysis of means display in which least squares means are compared to an average least squares mean	PLOTS=ANOM

Table 20.27 *continued*

ODS Graph Name	Plot Description	Required Option
ControlPlot	Requests a display in which least squares means are compared to a reference level	PLOTS=CONTROL
DiffPlot	Displays all pairwise least squares mean differences and their significance. This plot is also known as a "mean-mean scatter plot" when based on arithmetic means.	PLOTS=DIFF
LinesPlot	Displays comparisons between least squares means, with lines that indicate nonsignificant subsets	PLOTS=LINESPLOT
MeanPlot	Displays least squares means	PLOTS=MEANPLOT

Table 20.28 Graphs Produced by Procedures That Support the LSMEANS Statement and Bayesian Analysis

ODS Graph Name	Plot Description	Required Option
BoxPlot	Displays box plots of LS-means or LS-mean differences across a posterior sample	PLOTS=BOXPLOT
DistPanel	Displays panels of histograms with kernel density curves overlaid. Each plot contains the results for the posterior sample of each LS-mean or LS-mean difference	PLOTS=DISTPLOT
DistPlot	Displays a histogram with a kernel density curve overlaid. The plot contains the results for the posterior sample of the LS-mean or LS-mean difference.	PLOTS=DISTPLOT(UNPACK)
LinesPlot	Displays a plot of LS-mean comparison lines	LINES

For details about the *plot-options* of the LSMEANS statement, see the PLOTS= option in the section "LSMEANS Statement" on page 492. For more details about the DIFFPLOT in particular, see the section "Graphics for LS-Mean Comparisons" on page 3997 in Chapter 52, "The GLIMMIX Procedure."

LSMESTIMATE Statement

This statement documentation applies to the following SAS/STAT procedures:
GEE, GENMOD, LIFEREG, LOGISTIC, MIXED, ORTHOREG, PHREG, PLM, PROBIT, RMSTREG, SURVEYLOGISTIC, SURVEYPHREG, and SURVEYREG. It also applies to the RELIABILITY procedure in SAS/QC software. The LSMESTIMATE statement in the GLIMMIX procedure is documented in Chapter 52, "The GLIMMIX Procedure."

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means. In contrast to the LSMEANS statement, the LSMESTIMATE statement does not produce the least squares means or their differences; instead, you can estimate any linear function of the least squares means (including the means themselves or their differences). In contrast to the linear functions that are constructed with the ESTIMATE statement, you do not specify coefficients for the individual parameter estimates. Instead, with the LSMESTIMATE statement you specify coefficients for the least squares means; these are then converted for you into estimable functions for the parameter estimates.

The LSMESTIMATE statement thus combines important and convenient features of the LSMEANS and the ESTIMATE statement. As with the LSMEANS statement, the following conditions are true:

- You need to specify only a single effect; the mapping into linear estimable functions in terms of the parameter estimates is performed by the procedure.
- You can use the AT=, BYLEVEL, and OBSMARGINS options to affect the computation of the underlying least squares means.

As with the ESTIMATE statement you can do the following:

- specify multiple-row linear combinations.
- perform multiplicity corrections to control the familywise Type I error probability with the ADJUST= option.
- construct general linear functions of the least squares means.
- perform joint F or chi-square tests with or without order restrictions through the JOINT option.
- rely on positional or nonpositional syntax to specify coefficients for linear functions. For details about using nonpositional syntax, see the section "Positional and Nonpositional Syntax for Coefficients in Linear Functions" on page 486.

The computation of an LSMESTIMATE involves two coefficient matrices. Suppose that there are n_l levels for a valid least squares means effect (an effect that is part of your model and consists of classification variables only). Then the LS-means are formed as $\mathbf{L}_1\widehat{\boldsymbol{\beta}}$, where $\mathbf{L}_1$ is a $(n_l \times p)$ coefficient matrix. The $(k \times n_l)$ coefficient matrix $\mathbf{K}$ is formed from the *values* that you supply in the k rows of the LSMESTIMATE statement. The least squares means estimates then represent the $(k \times 1)$ vector

$$\mathbf{K}\mathbf{L}_1\boldsymbol{\beta} = \mathbf{L}\boldsymbol{\beta}$$

For multinomial response models, you must specify the same number of values as there are LS-means within a response function.

Because the analytic features and capabilities of the LSMESTIMATE statement are an amalgam of the LSMEANS and the ESTIMATE statement, the syntax of the statement follows the same pattern.

Syntax: LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < **DIVISOR=***n* >
< **,** < '*label*' > *values* < **DIVISOR=***n* > > < **,** . . . >
< / *options* > **;**

In contrast to a multirow estimate in the ESTIMATE statement, you specify only a single effect in the LSMESTIMATE statement. The row labels are optional and follow the *model-effect* specification. For example, the following statements fit a split-split-plot design and compare the average of the third and fourth LS-mean of the whole-plot factor A to the first LS-mean of the factor:

```
proc glimmix;
   class a b block;
   model y = a b a*b /  s;
   random int a / sub=block;
   lsmestimate A 'a1 vs avg(a3,a4)' 2 0 -1 -1 divisor=2;
run;
```

The order in which coefficients are assigned to the least squares means corresponds to the order in which they are displayed in the "Least Squares Means" table. You can use the ELSM option to see how coefficients are matched to levels of the fixed effect.

The optional DIVISOR=*n* specification enables you to assign a separate divisor to each row of the LSMESTIMATE. You can also assign divisor values through the DIVISOR= option. See the description of the DIVISOR= option that follows for the interaction between the two ways of specifying divisors.

Table 20.29 summarizes important options in the LSMESTIMATE statement. All LSMESTIMATE options are subsequently discussed in alphabetical order.

Table 20.29 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking

Table 20.29 *continued*

Option	Description
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint *F* or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

You can specify the following options in the LSMESTIMATE statement after a slash (/):

ADJDFE=SOURCE

ADJDFE=ROW

specifies how denominator degrees of freedom are determined when *p*-values and confidence limits are adjusted for multiple comparisons with the ADJUST= option. When you do not specify the ADJDFE= option or when you specify ADJDFE=SOURCE, the denominator degrees of freedom for multiplicity-adjusted results are the denominator degrees of freedom for the LS-mean effect in the "Type III Tests of Fixed Effects" table.

The ADJDFE=ROW setting is useful if you want multiplicity adjustments to take into account that denominator degrees of freedom are not constant across estimates. For example, this can be the case when the denominator degrees of freedom are computed by the Satterthwaite or Kenward-Roger method (Kenward and Roger 1997) in a mixed model.

The ADJDFE= option is not supported by the procedures that perform chi-square-based inference (GENMOD, LOGISTIC, PHREG and SURVEYLOGISTIC).

ADJUST=BON

ADJUST=SCHEFFE

ADJUST=SIDAK

ADJUST=SIMULATE< (*sim-options*) >

ADJUST=T

requests a multiple comparison adjustment for the *p*-values and confidence limits for the LS-mean estimates. The adjusted quantities are produced in addition to the unadjusted *p*-values and confidence limits. Adjusted confidence limits are produced if the CL or ALPHA= option is in effect. For a description of the adjustments, see Chapter 53, "The GLM Procedure," and Chapter 86, "The MULTTEST Procedure," in addition to the documentation for the ADJUST= option in the LSMEANS statement.

Not all adjustment methods of the LSMEANS statement are available for the LSMESTIMATE statement. Multiplicity adjustments in the LSMEANS statement are designed specifically for differences of least squares means.

If you specify the STEPDOWN option, the *p*-values are further adjusted in a step-down fashion.

ALPHA=*number*

requests that a *t* type confidence interval be constructed for each of the LS-means with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

AT *variable***=***value*

AT (*var-list***)=(***value-list***)**

AT MEANS

modifies the values of the covariates used in computing LS-means. See the AT option in the LSMEANS statement for details.

BYLEVEL

requests that the procedure compute separate margins for each level of the LSMEANS effect.

The standard LS-means have equal coefficients across classification effects. The BYLEVEL option changes these coefficients to be proportional to the observed margins. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in the input data set. In this case, the resulting LS-means are actually equal to raw means for fixed-effects models and certain balanced random-effects models, but their estimated standard errors account for the covariance structure that you have specified. If a WEIGHT statement is specified, the procedure uses weighted margins to construct the LS-means coefficients.

If the AT option is specified, the BYLEVEL option disables it.

CATEGORY=*category-options*

specifies how to construct estimates and multiplicity corrections for models with multinomial data (ordinal or nominal). This option is also important for constructing sets of estimable functions for *F* tests with the JOINT option.

The *category-options* indicate how response variable levels are treated in constructing the estimable functions. Possible value for the *category-options* are the following:

JOINT
computes the estimable functions for nonredundant categories and displays the estimates for all these categories in a single table. For example, a three-row LSMESTIMATE statement in a model that has three response categories displays six estimable functions in one table. The same estimates are computed by the SEPARATE *category-option*.

SEPARATE
computes the estimable functions for nonredundant categories and displays the estimates in a separate table for each category. For example, a three-row LSMESTIMATE statement in a model that has three response categories displays two tables, each with three estimable functions. The same estimates are computed by the JOINT *category-option*.

quoted-value-list
computes the estimable functions only for the specified list of values. The list must consist of formatted values of the response categories.

For further details about using the CATEGORY= option in models for multinomial data, see the documentation for the CATEGORY= option in the ESTIMATE statement.

The CATEGORY= option is supported only by the procedures that support generalized linear modeling (GEE, GENMOD, LOGISTIC, and SURVEYLOGISTIC) and by PROC PLM when it is used to perform statistical analyses on item stores that were created by these procedures.

CHISQ
requests that chi-square tests be performed in addition to F tests, when you request an F test with the JOINT option. This option has no effect in procedures that produce chi-square statistics by default.

CL
requests that t type confidence limits be constructed for each of the LS-means. The confidence level is 0.95 by default; this can be changed with the ALPHA= option. If you specify an ADJUST= option, then the confidence limits are adjusted for multiplicity. But if you also specify STEPDOWN, then only p-values are step-down adjusted, not the confidence limits.

CORR
displays the estimated correlation matrix of the linear combination of the least squares means.

COV
displays the estimated covariance matrix of the linear combination of the least squares means.

DF=*number*
specifies the degrees of freedom for the tests and confidence limits. The option is not supported by the procedures that perform chi-square-based inference (GENMOD, LOGISTIC, PHREG, and SURVEYLOGISTIC).

DIVISOR=*value-list*
specifies a list of values by which to divide the coefficients so that fractional coefficients can be entered as integer numerators. If you do not specify *value-list*, a default value of 1.0 is assumed. Missing values in the *value-list* are converted to 1.0.

If the number of elements in *value-list* exceeds the number of rows of the estimate, the extra values are ignored. If the number of elements in *value-list* is less than the number of rows of the estimate, the last value in *value-list* is carried forward.

If you specify a row-specific divisor as part of the specification of the estimate row, this value multiplies the corresponding value in the *value-list*. For example, the following statement divides the coefficients in the first row by 8, and the coefficients in the third and fourth row by 3:

```
lsmestimate A 'One vs. two'   8 -8   divisor=2,
              'One vs. three' 1  0 -1         ,
              'One vs. four'  3  0  0 -3      ,
              'One vs. five'  3  0  0  0  -3  / divisor=4,.,3;
```

Coefficients in the second row are not altered.

E

requests that the **L** coefficients of the estimable function be displayed. These are the coefficients that apply to the fixed-effect parameter estimates. The E option displays the coefficients that you would need to enter in an equivalent ESTIMATE statement.

ELSM

requests that the **K** matrix coefficients be displayed. These are the coefficients that apply to the LS-means. This option is useful to ensure that you assigned the coefficients correctly to the LS-means.

EXP

requests exponentiation of the least squares means estimate. When you model data with the logit link function and the estimate represents a log odds ratio, the EXP option produces an odds ratio. If you specify the CL or ALPHA= option, the (adjusted) confidence limits for the estimate are also exponentiated.

The EXP option is supported only by PROC PHREG, PROC SURVEYPHREG, the procedures that support generalized linear modeling (GEE, GENMOD, LOGISTIC, and SURVEYLOGISTIC), and by PROC PLM when it is used to perform statistical analyses on item stores that were created by these procedures.

ILINK

requests that the estimate and its standard error also be reported on the scale of the mean (the inverse linked scale). The computation of the inverse linked estimate depends on the estimation mode. For example, if the analysis is based on a posterior sample when a BAYES statement is present, the inversely linked estimate is the average of the inversely linked values across the sample of posterior parameter estimates. If the analysis is not based on a sample of parameter estimates, the procedure computes the value on the mean scale by applying the inverse link to the estimate.

The interpretation of the inversely linked quantity depends on the coefficients that are specified in your LSMESTIMATE statement and the link function. For example, in a model for binary data with logit link the following LSMESTIMATE statement computes

$$q = \frac{1}{1 + \exp\{-(\tau_1 - \tau_2)\}}$$

where τ_1 and τ_2 are the least squares means that are associated with the first two levels of the classification effect A:

```
proc logistic;
   class A / param=glm;
   model y = A / dist=binary link=logit;
   lsmestimate A 1 -1 / ilink;
run;
```

The quantity q is not the difference of the probabilities associated with the two levels,

$$\pi_1 - \pi_2 = \frac{1}{1+\exp\{-\tau_1\}} - \frac{1}{1+\exp\{-\tau_2\}}$$

The standard error of the inversely linked estimate is based on the delta method. If you also specify the CL or ALPHA= option, the procedure computes confidence intervals for the inversely linked estimate. These intervals are obtained by applying the inverse link to the confidence intervals on the linked scale.

The ILINK option is supported only by the procedures that support generalized linear modeling (GEE, GENMOD, LOGISTIC, and SURVEYLOGISTIC) and by PROC PLM when it is used to perform statistical analyses on item stores that were created by these procedures.

JOINT< (*joint-test-options*) >

requests that a joint F or chi-square test be produced for the rows of the estimate. For multinomial response models, a joint test is constructed for each response function. For more information about the simulation-based p-value calculation, see the section "Joint Hypothesis Tests with Complex Alternatives, the Chi-Bar-Square Statistic" on page 489. You can specify the following *joint-test-options* in parentheses:

ACC=γ

specifies the accuracy radius for determining the necessary sample size in the simulation-based approach of Silvapulle and Sen (2004) for tests with order restrictions. The value of γ must be strictly between 0 and 1; the default value is 0.005.

EPS=ϵ

specifies the accuracy confidence level for determining the necessary sample size in the simulation-based approach of Silvapulle and Sen (2004) for F tests with order restrictions. The value of ϵ must be strictly between 0 and 1; the default value is 0.01.

LABEL='*label***'**

assigns an identifying label to the joint test. If you do not specify a label, the first nondefault label for the ESTIMATE rows is used to label the joint test.

NOEST

ONLY

performs only the joint test and suppresses other results from the ESTIMATE statement. This option is useful for emulating the CONTRAST statement that is available in other procedures.

NSAMP=*n*

specifies the number of samples for the simulation-based method of Silvapulle and Sen (2004). If *n* is not specified, it is constructed from the values of the ALPHA=α, the ACC=γ, and the EPS=ϵ options. With the default values for γ, ϵ, and α (0.005, 0.01, and 0.05, respectively), NSAMP=12,604 by default.

CHISQ

adds a chi-square test if the procedure produces an F test by default.

BOUNDS=*value-list*

specifies boundary values for the estimable linear function. The null value of the hypothesis is always zero. If you specify a positive boundary value z, the hypotheses are $H: \theta = 0$, $H_a:: \theta > 0$ with the added constraint that $\theta < z$. The same is true for negative boundary values. The alternative hypothesis is then $H_a: \theta < 0$ subject to the constraint $\theta > -|z|$. If you specify a missing value, the hypothesis is assumed to be two-sided. The BOUNDS option enables you to specify sets of one- and two-sided joint hypotheses. If all values in *value-list* are set to missing, the procedure performs a simulation-based p-value calculation for a two-sided test.

LOWER

LOWERTAILED

requests that the p-value for the t test be based only on values that are less than the test statistic. A two-tailed test is the default. A lower-tailed confidence limit is also produced if you specify the CL or ALPHA= option.

Note that for ADJUST=SCHEFFE the one-sided adjusted confidence intervals and one-sided adjusted p-values are the same as the corresponding two-sided statistics, because this adjustment is based on only the right tail of the F distribution.

If you request an F test with the JOINT option, then a one-sided left-tailed order restriction is applied to all estimable functions, and the corresponding chi-bar-square statistic of Silvapulle and Sen (2004) is computed in addition to the two-sided, standard, F or chi-square statistic. See the JOINT option for how to control the computation of the simulation-based chi-bar-square statistic.

OBSMARGINS< =*OM-data-set* **>**

OM< =*OM-data-set* **>**

specifies a potentially different weighting scheme for the computation of LS-means coefficients. The standard LS-means have equal coefficients across classification effects; however, the OM option changes these coefficients to be proportional to those found in the *OM-data-set*. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in *OM-data-set*. See the OBSMARGINS option in the LSMEANS statement for further details.

PLOTS=*plot-options*

produces ODS statistical graphics of the distribution of estimable functions if the procedure performs the analysis in a sampling-based mode. For example, this is the case when procedures support a BAYES statement and perform a Bayesian analysis. The estimable functions are then computed for each of the posterior parameter estimates, and the "Least Squares Means Estimates" table reports simple descriptive statistics for the evaluated functions. In this situation, the PLOTS= option enables you to visualize the distribution of the estimable function. The following *plot-options* are available:

ALL

produces all possible plots with their default settings.

BOXPLOT< (*boxplot-options*) >

produces box plots of the distribution of the estimable function across the posterior sample. A separate box plot is generated for each estimable function and all box plots appear on a single graph by default. You can affect the appearance of the box plot graph with the following options:

ORIENTATION=VERTICAL | HORIZONTAL

ORIENT=VERT | HORIZ

specifies the orientation of the boxes. The default is vertical orientation of the box plots.

NPANELPOS=*number*

specifies how to break the series of box plots across multiple panels. If the NPANELPOS option is not specified, or if *number* equals zero, then all box plots are displayed in a single graph; this is the default. If a negative number is specified, then exactly up to |*number*| of box plots are displayed per panel. If *number* is positive, then the number of boxes per panel is balanced to achieve small variation in the number of box plots per graph.

DISTPLOT< (*distplot-options*) >

DIST< (*distplot-options*) >

generates panels of histograms with a kernel density overlaid. A separate plot in each panel contains the results for each estimable function. You can specify the following *distplot-options* in parentheses:

BOX | NOBOX

controls the display of a horizontal box plot below the histogram. The BOX option is enabled by default.

HIST | NOHIST

controls the display of the histogram of the estimable function's distribution across the posterior sample. The HIST option is enabled by default.

NORMAL | NONORMAL

controls the display of a normal density estimate on the graph. The NONORMAL option is enabled by default.

KERNEL | NOKERNEL

controls the display of a kernel density estimate on the graph. The KERNEL option is enabled by default.

NROWS=*number*

specifies the highest number of rows in a panel. The default is 3.

NCOLS=*number*

specifies the highest number of columns in a panel. The default is 3.

UNPACK
unpacks the panel into separate graphics.

NONE
does not produce any plots.

SEED=*number*
specifies the seed for the sampling-based components of the computations for the LSMESTIMATE statement (for example, chi-bar-square statistics and simulated *p*-values). The value of *number* must be an integer. The seed is used to start the pseudo-random-number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. Note that there could be multiple LSMESTIMATE statements with SEED= specifications and there could be other statements that can supply a random number seed. Since the procedure has only one random number stream, the initial seed is shown in the SAS log.

SINGULAR=*number*
tunes the estimability checking as documented for the SINGULAR= option in the ESTIMATE statement.

STEPDOWN< (*step-down-options***) >**
requests that multiplicity adjustments for the *p*-values of estimable functions be further adjusted in a step-down fashion. Step-down methods increase the power of multiple testing procedures by taking advantage of the fact that a *p*-value is never declared significant unless all smaller *p*-values are also declared significant. The STEPDOWN adjustment combined with ADJUST=BON corresponds to the methods of Holm (1979) and "Method 2" of Shaffer (1986); this is the default. Using step-down-adjusted *p*-values combined with ADJUST=SIMULATE corresponds to the method of Westfall (1997).

If the ESTIMATE statement is applied with a STEPDOWN option in a mixed model where the degrees-of-freedom method is that of Kenward and Roger (1997) or of Satterthwaite, then step-down-adjusted *p*-values are produced only if the ADJDFE=ROW option is in effect.

Also, the STEPDOWN option affects only *p*-values, not confidence limits. For ADJUST=SIMULATE, the generalized least squares hybrid approach of Westfall (1997) is used to increase Monte Carlo accuracy.

You can specify the following *step-down-options* in parentheses:

MAXTIME=*n*
specifies the time (in seconds) to be spent computing the maximal logically consistent sequential subsets of equality hypotheses for TYPE=LOGICAL. The default is MAXTIME=60. If the MAXTIME value is exceeded, the adjusted tests are not computed. When this occurs, you can try increasing the MAXTIME value. However, note that there are common multiple comparisons problems for which this computation requires a huge amount of time—for example, all pairwise comparisons between more than 10 groups. In such cases, try to use TYPE=FREE (the default) or TYPE=LOGICAL(*n*) for small *n*.

ORDER=PVALUE

ORDER=ROWS

specifies the order in which the step-down tests are performed. ORDER=PVALUE is the default, with LS-mean estimates being declared significant only if all LS-mean estimates with smaller (unadjusted) p-values are significant. If you specify ORDER=ROWS, then significances are evaluated in the order in which they are specified.

REPORT

specifies that a report on the step-down adjustment be displayed, including a listing of the sequential subsets (Westfall 1997) and, for ADJUST=SIMULATE, the step-down simulation results.

TYPE=LOGICAL<(*n*)>

TYPE=FREE

specifies how step-down adjustment are made. If you specify TYPE=LOGICAL, the step-down adjustments are computed by using maximal logically consistent sequential subsets of equality hypotheses (Shaffer 1986; Westfall 1997). Alternatively, for TYPE=FREE, sequential subsets are computed ignoring logical constraints. The TYPE=FREE results are more conservative than those for TYPE=LOGICAL, but they can be much more efficient to produce for many estimates. For example, it is not feasible to take logical constraints between all pairwise comparisons of more than about 10 groups. For this reason, TYPE=FREE is the default.

However, you can reduce the computational complexity of taking logical constraints into account by limiting the depth of the search tree used to compute them, specifying the optional depth parameter as a number n in parentheses after TYPE=LOGICAL. As with TYPE=FREE, results for TYPE=LOGICAL(n) are conservative relative to the true TYPE=LOGICAL results. But even for TYPE=LOGICAL(0), they can be appreciably less conservative than TYPE=FREE, and they are computationally feasible for much larger numbers of estimates. If you do not specify n or if $n = -1$, the full search tree is used.

TESTVALUE=*value-list*

TESTMEAN=*value-list*

specifies the value under the null hypothesis for testing the estimable functions in the LSMESTIMATE statement. The rules for specifying the *value-list* are very similar to those for specifying the divisor list in the DIVISOR= option. If no TESTVALUE= is specified, all tests are performed as $H: \mathbf{L}\boldsymbol{\beta} = 0$. Missing values in the *value-list* also are translated to zeros. If you specify fewer values than rows in the LSMESTIMATE statement, the last value in *value-list* is carried forward.

The TESTVALUE= option affects only p-values from individual, joint, and multiplicity-adjusted tests. It does not affect confidence intervals.

The TESTVALUE option is not available for the multinomial distribution, and the values are ignored when you perform a sampling-based (Bayesian) analysis.

UPPER

UPPERTAILED

requests that the p-value for the t test be based only on values that are greater than the test statistic. A two-tailed test is the default. An upper-tailed confidence limit is also produced if you specify the CL or ALPHA= option.

Note that for ADJUST=SCHEFFE the one-sided adjusted confidence intervals and one-sided adjusted p-values are the same as the corresponding two-sided statistics, because this adjustment is based on only the right tail of the F distribution.

If you request a joint test with the JOINT option, then a one-sided right-tailed order restriction is applied to all estimable functions, and the corresponding chi-bar-square statistic of Silvapulle and Sen (2004) is computed in addition to the two-sided, standard, F or chi-square statistic. See the JOINT option for how to control the computation of the simulation-based chi-bar-square statistic.

ODS Table Names: LSMESTIMATE Statement

Each table created by the LSMESTIMATE statement has a name associated with it, and you can use this name to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 20.30. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 20.30 ODS Tables Produced by the LSMESTIMATE Statement

Table Name	Description	Required Option
Coef	**L** matrix coefficients or **K** matrix coefficients	E or ELSM
LSMEstimates	Estimates among LS-means	Default
Contrasts	Joint test results for LS-means estimates	JOINT

ODS Graphics: LSMESTIMATE Statement

This section describes the use of ODS for creating statistical graphs of the distribution of LS-means and LS-mean differences with the LSMESTIMATE statement. The plots can be produced only in association with the three procedures that can perform Bayesian analysis (PROC GENMOD, PROC LIFEREG, and PROC PHREG). The plots are available via these procedures directly, and also via PROC PLM when run using an item store that was created by these procedures. To request these graphs, you must do the following:

- ensure that ODS Graphics is enabled
- use a BAYES statement with PROC GENMOD, PROC LIFEREG, or PROC PHREG, or use PROC PLM to perform statistical analysis on an item store that was saved from a Bayesian analysis
- request plots with the PLOTS= option in the LSMESTIMATE statement

For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." The available graphs are summarized in Table 20.31.

Table 20.31 Graphs Produced by the LSMESTIMATE Statement

ODS Graph Name	Plot Description	Required Option
BoxPlot	Displays box plots of LS-means or LS-mean differences across a posterior sample.	PLOTS=BOXPLOT
DistPanel	Displays panels of histograms with kernel density curves overlaid. Each plot contains the results for the posterior sample of each LS-mean or LS-mean difference	PLOTS=DISTPLOT
DistPlot	Displays a histogram with a kernel density curve overlaid. The plot contains the results for the posterior sample of the LS-mean or LS-mean difference.	PLOTS=DISTPLOT(UNPACK)

For details about the *plot-options* of the LSMESTIMATE statement, see the PLOTS= option in the section "LSMESTIMATE Statement" on page 513.

MARGINS Statement

This section applies to the following procedures: GEE, GENMOD, GLIMMIX, LOGISTIC, SURVEYLOGISTIC, SURVEYPHREG, and SURVEYREG.

The MARGINS statement computes predictive margins of fixed effects. You can compute predictive margins for any effect in the MODEL statement that involves only classification variables.

The predictive margin for a specific level (group) of a fixed effect represents the average predicted response if all the observations in the data set were in that group (Lane and Nelder 1982; Chang, Gelman, and Pagano 1982). You compute the predictive margin by fixing this effect at the specified level for all observations and averaging the predicted responses. The standard error of the predictive margin is computed by using the delta method.

For example, consider the model

$$g(y_{ij}) = \alpha_i + \beta x_{ij}$$

where g is the link function; $\alpha_i, i = 1, \ldots, K$, is the effect of the ith level of the classification effect α; and x_{ij} is the covariate for the jth observation in the ith level of the effect α, $j = 1, \ldots, n_i$. Denote the number of observations in the ith level as n_i, denote the frequency of the jth observation in the ith level as f_{ij} and the total frequency as $F = \sum_{i=1}^{K} \sum_{j=1}^{n_i} f_{ij}$, and denote the estimated model parameters at level k of effect α as

$(\hat{\alpha}_k, \hat{\beta})$. Then the predictive margin for level k of effect α is

$$\frac{1}{F}\sum_{i=1}^{K}\sum_{j=1}^{n_i} f_{ij}\, g^{-1}(\hat{\alpha}_k, \hat{\beta} x_{ij})$$

Note that when the USEWEIGHT=TRUE option is in effect, f_{ij} is the product of the frequency and weight of the observation.

By definition, predictive margins are covariate-adjusted marginal means, as are LS-means. These two approaches are discussed in the section "Predictive Margins Compared with LS-Means" on page 532.

Lane and Nelder (1982) and Chang, Gelman, and Pagano (1982) discuss how to use predictive margins to standardize predictions that are based on nonlinear regression models. Graubard and Korn (1999) show the computation of predictive margins in survey analysis.

Syntax: MARGINS Statement

MARGINS *fixed-effects* < / *options* > **;**

You can specify multiple effects in one or multiple MARGINS statements, and all MARGINS statements must appear after the MODEL statement. Predictive margin computations are not supported for t or lognormal distributions or for constructed effects.

Some procedures construct an approximate t test to test the null hypothesis that the associated population quantity equals zero; other procedures perform chi-square-based inference. By default, the denominator degrees of freedom for the t test are the same as those displayed for the effect in the "Type III Tests of Fixed Effects" table. Table 20.32 summarizes the options available in the MARGINS statement. All options for the MARGINS statement are discussed in alphabetical order after the table.

Table 20.32 MARGINS Statement Options

Option	Description
Construction and Computation of Predictive Margins	
AT	Modifies the covariate value in computing predictive margins
DIFF	Computes differences of predictive margins
SLICEBY=	Partitions tests of interaction effects
SLICEDIFF	Computes differences of sliced predictive margins and determines the types of differences
USEWEIGHT	Specifies whether to use WEIGHT variables to compute predictive margins
Degrees of Freedom and *p*-Values	
ADJUST=	Specifies the method of multiple comparison adjustment of predictive margin differences
ALPHA=α	Specifies the confidence level $(1-\alpha)$
DF=	Assigns a specific value to degrees of freedom for t tests and confidence limits
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion

Table 20.32 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for predictive margins and/or predictive margin differences

You can specify the following *options* in the MARGINS statement after a slash (/).

ADJUST=BON
ADJUST=DUNNETT
ADJUST=SCHEFFE
ADJUST=SIDAK
ADJUST=SIMULATE< (*simoptions*) >
ADJUST=SMM | GT2
ADJUST=TUKEY

performs a multiple comparison adjustment for the p-values and confidence limits for the differences of predictive margins. The adjusted quantities are produced in addition to the unadjusted quantities. By default, the procedures perform all pairwise differences. If you specify ADJUST=DUNNETT, the procedures analyze all differences between the margins and the control level. The ADJUST= option does not imply the DIFF option.

The BON (Bonferroni) and SIDAK adjustments involve correction factors described in Chapter 53, "The GLM Procedure," and Chapter 86, "The MULTTEST Procedure"; also see Westfall and Young (1993) and Westfall et al. (1999). When you specify ADJUST=TUKEY and your data are unbalanced, the procedures use the approximation described in Kramer (1956) and identify the adjustment as "Tukey-Kramer" in the results. Similarly, when you specify ADJUST=DUNNETT and the predictive margins are correlated, the procedures use the factor-analytic covariance approximation described in Hsu (1992) and identify the adjustment as "Dunnett-Hsu" in the results. The approximation derives an approximate "effective sample sizes" for which exact critical values are computed. Note that computing the exact adjusted p-values and critical values for unbalanced designs can be computationally intensive. A simulation-based approach, specified by the ADJUST=SIM option, is nondeterministic but can provide sufficiently accurate inferences in much less time. The preceding references also describe the SCHEFFE and SMM adjustments.

The SIMULATE adjustment computes adjusted p-values and confidence limits from the simulated distribution of the maximum or maximum absolute value of a multivariate t random vector. All covariance parameters, except the residual scale parameter, are fixed at their estimated values throughout the simulation, potentially resulting in some underdispersion. The simulation estimates q, the true $(1-\alpha)$ quantile, where $1-\alpha$ is the confidence coefficient. The default value of α is 0.05, and you can change this value by specifying the ALPHA= option in the MARGINS statement.

The number of samples is set so that the tail area for the simulated q is within γ of $1-\alpha$ with $100(1-\epsilon)\%$ confidence. In equation form,

$$\Pr(|F(\widehat{q}) - (1-\alpha)| \leq \gamma) = 1 - \epsilon$$

where $\hat{q}$ is the simulated q and F is the true distribution function of the maximum; see Edwards and Berry (1987) for details. By default, $\gamma = 0.005$ and $\epsilon = 0.01$, placing the tail area of $\hat{q}$ within 0.005 of

0.95 with 99% confidence. The ACC= and EPS= *simoptions* reset γ and ϵ, respectively; the NSAMP= *simoption* sets the sample size directly; and the SEED= *simoption* specifies an integer to use in starting the pseudorandom number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated by reading the time of day from the computer's clock. For additional descriptions of these and other simulation options, see the section "LSMEANS Statement" on page 4172 in Chapter 53, "The GLM Procedure."

If you specify the STEPDOWN option, the p-values are further adjusted in a step-down fashion. For certain options and data, this adjustment is exact under an iid $N(0, \sigma^2)$ model for the dependent variable, in particular for the following:

- for ADJUST=DUNNETT when the means are uncorrelated
- for ADJUST=TUKEY with STEPDOWN(TYPE=LOGICAL) when the means are balanced and uncorrelated

The first case is a consequence of the nature of the successive step-down hypotheses for comparisons with a control; the second case uses an extension of the maximum studentized range distribution appropriate for partition hypotheses (Royen 1989). Finally, when STEPDOWN(TYPE=FREE), the ADJUST=TUKEY option uses the Royen (1989) extension in such a way that the resulting p-values are conservative.

ALPHA=*number*

constructs confidence intervals for each of the predictive margins with significance level 1 – *number*. The value of *number* must be between 0 and 1; by default, ALPHA=0.05.

AT *variable***=***value*

AT (*variable-list***)=(***value-list***)**

AT MEANS

enables you to modify the values of the covariates that are used in computing predictive margins. By default, all covariate effects are set equal to their observed values for computation of predictive margins. The AT option enables you to assign arbitrary values to the covariates. Additional columns in the output table indicate the values of the covariates.

If there is an effect that contains two or more covariates, the AT MEANS option sets the effect equal to the product of the individual means rather than the mean of the product.

As an example, consider the following invocation of PROC GLIMMIX:

```
proc glimmix;
   class A;
   model Y = A x1 x2 x1*x2;
   margins A;
   margins A / at means;
   margins A / at x1=1.2;
   margins A / at (x1 x2)=(1.2 0.3);
run;
```

The first MARGINS statement sets x1 and x2 to their observed values for each observation. In the second MARGINS statement, x1 is set to $\overline{x}_1$ (the mean of x1) and x2 is set to $\overline{x}_2$ (the mean of x2). Their interaction is set to $\overline{x}_1 \times \overline{x}_2$. The third MARGINS statement sets x1 to 1.2 and x2 to its observed values, and the final MARGINS statement sets these variables to 1.2 and 0.3, respectively.

CL

constructs *t*-type confidence limits for each of the predictive margins. If the procedure enables you to specify the DDFM=NONE option, then the procedure uses infinite degrees of freedom for this test, essentially computing a *z* interval. The confidence level is 0.95 by default; you can change this by specifying the ALPHA= option. If you specify an ADJUST= option, then the confidence limits are adjusted for multiplicity, but if you also specify the STEPDOWN option, then only *p*-values are adjusted in a step-down fashion, not the confidence limits.

DF=*number*

specifies the degrees of freedom for the *t* test and confidence limits. The default is the denominator degrees of freedom from the "Type III Tests of Fixed Effects" table corresponding to the margins effect. This option has no effect on procedures that perform chi-square-based inference.

DIFF<=*difftype***>**

PDIFF<=*difftype***>**

displays differences of the predictive margins. The optional *difftype* specifies which differences to display; the *difftype* values are ALL, CONTROL, CONTROLL, and CONTROLU. The ALL *difftype* (the default) displays all pairwise differences. The CONTROL *difftype* computes differences between each margin and a control; by default, the control is the first level of each effect that you specify in the MARGINS statement.

To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the CONTROL keyword. For example, for the classification variables A, B, and C, each having two levels, 1 and 2, the following MARGINS statement specifies the (1,2) level of A*B and the (2,1) level of B*C as controls:

```
margins A*B B*C / diff=control('1' '2', '2' '1');
```

For an effect that has multiple classification variables, the quoted formatted values are assigned to these variables in the order in which they appear in the CLASS statement.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*. The CONTROLL *difftype* tests whether the noncontrol levels are significantly smaller than the control; the upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing. Conversely, the CONTROLU *difftype* tests whether the noncontrol levels are significantly larger than the control; the upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

If you want to perform multiple comparison adjustments on the differences of predictive margins, you must specify the ADJUST= option.

SLICEBY=*fixed-effect* **|** **(***fixed-effects***)**

specifies effects by which to partition interaction MARGINS effects. This option produces tests of margins for each level of the specified slice effect.

For example, suppose that the margins effect is A*B. To test the margins of A*B for each level of A, you can specify the following statement:

```
margins A*B / sliceby=A;
```

The SLICEBY= option produces an F test (or a chi-square test for procedures that perform chi-square-based inference) that tests the simultaneous equality of the margins at a fixed level of the slice effect A. To display differences of the margins while holding one or more factors at a fixed level, specify the SLICEDIFF option.

SLICEDIFF<=*difftype*>

constructs the specified type of differences for the sliced margins and tests them against zero. The *difftype* values are ALL, CONTROL, CONTROLL, and CONTROLU. The ALL *difftype* constructs all pairwise differences; it is the default. The CONTROL, CONTROLL, and CONTROLU *difftypes* construct the differences between the margins and the control. Whereas the SLICEBY= option tests the simultaneous equality of the margins at a fixed level of the slice effect, the SLICEDIFF option tests pairwise differences of these margins. This enables you to perform multiple comparisons among the levels of one factor at a fixed level of the other factor.

For example, assume that, in a certain design, factors A and B have $a = 4$ and $b = 3$ levels, respectively. Consider the following statements:

```
proc glimmix;
   class A B;
   model y = A B A*B;
   margins A*B / sliceby=A;
   margins A*B / sliceby=A slicediff=all;
run;
```

The first MARGINS statement produces four F tests, one per level of A. Denote the three margins that correspond to the first level of A as $\mathbf{m}_{a1}^{(1)}$, $\mathbf{m}_{a1}^{(2)}$, and $\mathbf{m}_{a1}^{(3)}$. Then the first F test performs a test of the two-degrees-of-freedom hypothesis

$$H: \begin{cases} \mathbf{m}_{a1}^{(1)} - \mathbf{m}_{a1}^{(2)} = 0 \\ \mathbf{m}_{a1}^{(1)} - \mathbf{m}_{a1}^{(3)} = 0 \end{cases}$$

The SLICEDIFF=ALL option performs tests of the difference between all pairs of these three margins. In the example this corresponds to tests of the form

$$H: \mathbf{m}_{a1}^{(1)} - \mathbf{m}_{a1}^{(2)} = 0$$
$$H: \mathbf{m}_{a1}^{(1)} - \mathbf{m}_{a1}^{(3)} = 0$$
$$H: \mathbf{m}_{a1}^{(2)} - \mathbf{m}_{a1}^{(3)} = 0$$

In the example, with $a = 4$ and $b = 3$, the second MARGINS statement produces four sets of predictive margin differences. Within each set, factor A is held fixed at a particular level, and each set consists of three comparisons.

For computing differences between the margins and the control, the default control is the first level of each effect that you specify in the MARGINS statement. To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the keyword CONTROL.

For example, if the effects A and B are classification variables, each having three levels (1, 2, and 3), the following MARGINS statement specifies the (1,3) level of A*B as the control:

```
margins A*B / sliceby=(A B)
              slicediff=control('1' '3');
```

This MARGINS statement first produces predictive margin differences while holding the levels of A fixed, and then produces predictive margin differences while holding the levels of B fixed. In the former case, level '3' of B serves as the control level. In the latter case, level '1' of A serves as the control.

Note that the control level specification is positional; for example, if you want to produce predictive margin differences while holding the levels of A fixed and using level '3' of B as the control level, then you still must specify a placeholder for the control level of A:

```
margins A*B / sliceby=A
              slicediff=control('1' '3');
```

For an effect that has multiple classification variables, the quoted formatted values are assigned to these variables in the order in which they appear in the CLASS statement.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*. The CONTROLL *difftype* tests whether the noncontrol levels are significantly smaller than the control; the upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing. Conversely, the CONTROLU *difftype* tests whether the noncontrol levels are significantly larger than the control; the upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

When you specify the ADJUST= option, the procedures also adjust the tests for multiplicity. The adjustment is based on the number of comparisons within each level of the SLICEBY= effect.

STEPDOWN< (*step-down-options*) >

further adjusts multiple comparison adjustments for the p-values of predictive margin differences in a step-down fashion. Step-down methods increase the power of multiple comparisons by taking advantage of the fact that a p-value is never declared significant unless all smaller p-values are also declared significant. Note that the step-down adjustment together with the ADJUST=BON option corresponds to the methods of Holm (1979) and "Method 2" of Shaffer (1986); this is the default. Using step-down-adjusted p-values combined with the ADJUST=SIMULATE option corresponds to the method of Westfall (1997).

The STEPDOWN option affects only p-values, not confidence limits. When ADJUST=SIMULATE, the generalized least squares hybrid approach of Westfall (1997) is used to increase Monte Carlo accuracy.

You can specify the following *step-down-options* in parentheses:

MAXTIME=*n*

specifies the time (in seconds) to spend computing the maximal logically consistent sequential subsets of equality hypotheses when TYPE=LOGICAL. By default, MAXTIME=60. If the specified value is exceeded, the adjusted tests are not computed. When this occurs, you can try increasing this value. However, note that there are common multiple comparison problems for which this computation requires a huge amount of time—for example, for all pairwise

comparisons of more than 10 groups. In such cases, try specifying TYPE=FREE (the default) or TYPE=LOGICAL(*n*) for small *n*.

TYPE=LOGICAL< (*n*) > | FREE

specifies whether to use logical constraints. When TYPE=LOGICAL, the step-down adjustments are computed by using maximal logically consistent sequential subsets of equality hypotheses (Shaffer 1986; Westfall 1997). When TYPE=FREE, sequential subsets are computed without logical constraints. The TYPE=FREE results are more conservative than those for TYPE=LOGICAL, but they can be much more efficient to produce for many comparisons. For example, it is not feasible to take logical constraints between all pairwise comparisons of more than 10 groups. For this reason, TYPE=FREE is the default.

However, you can reduce the computational complexity of taking logical constraints into account by limiting the depth of the search tree that you use to compute them, specifying the optional depth parameter as a number *n* in parentheses after TYPE=LOGICAL. As with TYPE=FREE, results for TYPE=LOGICAL(*n*) are more conservative than the true TYPE=LOGICAL results, but even for TYPE=LOGICAL(0) they can be appreciably less conservative than the TYPE=FREE results, and they are computationally feasible for much larger numbers of comparisons. If you do not specify *n*, or if $n = -1$, the full search tree is used.

USEWEIGHT< =TRUE | FALSE >

WEIGHT< =TRUE | FALSE >

WEIGHTED< =TRUE | FALSE >

specifies whether to take WEIGHT variables into account in computing predictive margins. Frequency variables are always used in the predictive margins computations. Specifying the USEWEIGHT option is the same as specifying USEWEIGHT=TRUE. See the individual procedure chapters to determine whether weights are used in the computations by default.

Predictive Margins Compared with LS-Means

The MARGINS statement computes predictive margins of fixed effects. The LSMEANS statement computes least squares means (LS-means) of fixed effects. Both predictive margins and LS-means are covariate-adjusted marginal means. They differ in the following ways:

- LS-means estimate the marginal means over a balanced population—that is, a population in which there are an equal number of observations for all possible effect level combinations. In contrast, predictive margins are computed over the observed distribution of the covariates. You can use the OM option in the LSMEANS statement to change the LS-means coefficients to be proportional to those found in the input data set. For linear models, this adjustment leads to LS-means that are equal to predictive margins if the margin effect is not part of any interaction effect.

- LS-means are constructed on the linear predictor scale—that is, the scale of $x\beta + z\gamma$. In contrast, predictive margins are computed on the mean scale—that is, the scale of $g^{-1}(x\beta + z\gamma)$. You can use the ILINK option in the LSMEANS statement to apply the inverse link transform to the LS-means. Note that in general the transformed LS-means are not equal to predictive margins for nonlinear models. They are equal only when the values of all covariates and classification variables are either specified or fixed in the computation.

ODS Table Names: MARGINS Statement

Each table that the MARGINS statement creates has a name associated with it. You refer to the table by this name when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 20.33. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 20.33 ODS Tables Produced by the MARGINS Statement

Table Name	Description	Required Option
Margins	Predictive margins	Default
MarginsDiffs	Differences of predictive margins	DIFF
MarginsSliceDiffs	Differences of sliced predictive margins	SLICEBY= or SLICEDIFF
MarginsSliceTests	Tests of sliced predictive margins	SLICEBY=

NLOPTIONS Statement

This section applies to the following procedures:
CALIS, GLIMMIX, HPMIXED, LOGISTIC, PHREG, SPP, SURVEYPHREG, and VARIOGRAM. See the individual procedure chapters for deviations from the common syntax and defaults shown here.

Syntax: NLOPTIONS Statement

The NLOPTIONS statement provides you with syntax to control aspects of the nonlinear optimizations in the CALIS, GLIMMIX, HPMIXED, LOGISTIC, PHREG, SPP, SURVEYPHREG, and VARIOGRAM procedures.

NLOPTIONS < *options* > **;**

The nonlinear optimization options are described in alphabetical order after Table 20.34, which summarizes the options by category. The notation used in describing the options is generic in the sense that $\boldsymbol{\psi}$ denotes the $p \times 1$ vector of parameters for the optimization and ψ_i is its ith element. The objective function being minimized, its $p \times 1$ gradient vector, and its $p \times p$ Hessian matrix are denoted as $f(\boldsymbol{\psi})$, $\mathbf{g}(\boldsymbol{\psi})$, and $\mathbf{H}(\boldsymbol{\psi})$, respectively. The gradient with respect to the ith parameter is denoted as $g_i(\boldsymbol{\psi})$. Superscripts in parentheses denote the iteration count; for example, $f(\boldsymbol{\psi})^{(k)}$ is the value of the objective function at iteration k. In the mixed model procedures, the parameter vector $\boldsymbol{\psi}$ might consist of fixed effects only, covariance parameters only, or fixed effects and covariance parameters. In the CALIS procedure, $\boldsymbol{\psi}$ consists of all independent parameters that are defined in the models and in the PARAMETERS statement.

Table 20.34 Options to Control Aspects of the Optimization

Option	Description
Optimization	
HESCAL=	Determines the type of Hessian scaling
INHESSIAN=	Specifies the start for approximated Hessian
LINESEARCH=	Specifies the line-search method
LSPRECISION=	Specifies the line-search precision
RESTART=	Specifies the iteration number for update restart
TECHNIQUE=	Determines the minimization technique
UPDATE=	Determines the update technique
Termination Criteria	
ABSCONV=	Tunes an absolute function convergence criterion
ABSFCONV=	Tunes an absolute function difference convergence criterion
ABSGCONV=	Tunes the absolute gradient convergence criterion
ABSXCONV=	Tunes the absolute parameter convergence criterion
FCONV=	Tunes the relative function convergence criterion
FCONV2=	Tunes another relative function convergence criterion
FSIZE=	Specifies the value used in the FCONV and GCONV criteria
GCONV=	Tunes the relative gradient convergence criterion
GCONV2=	Tunes another relative gradient convergence criterion
MAXFUNC=	Specifies the maximum number of function calls
MAXITER=	Specifies the maximum number of iterations
MAXTIME=	Specifies the upper limit for seconds of CPU time
MINITER=	Specifies the minimum number of iterations
XCONV=	Specifies the relative parameter convergence criterion
XSIZE=	Specifies the value used in the XCONV criterion
Step Length	
DAMPSTEP=	Dampens steps in a line search
INSTEP=	Specifies the initial trust region radius
MAXSTEP=	Specifies the maximum trust region radius
Printed Output	
PALL	Displays (almost) all printed output
PHISTORY	Displays optimization history
NOPRINT	Suppresses all printed output
Covariance Matrix Tolerances	
ASINGULAR=	Specifies the absolute singularity for inertia
MSINGULAR=	Specifies the relative M singularity for inertia
VSINGULAR=	Specifies the relative V singularity for inertia
Constraint Specifications	
LCEPSILON=	Specifies the range for active constraints
LCDEACT=	Specifies the LM tolerance for deactivating
LCSINGULAR=	Specifies the tolerance for dependent constraints

Table 20.34 *continued*

Option	Description
Remote Monitoring	
SOCKET=	Specifies the fileref for remote monitoring

ABSCONV=*r*

ABSTOL=*r*

specifies an absolute function convergence criterion: for minimization, termination requires $f(\boldsymbol{\psi}^{(k)}) \leq r$. The default value of r is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

ABSFCONV=*r* **<** *n* **>**

ABSFTOL=*r***<** *n* **>**

specifies an absolute function difference convergence criterion:

- For all techniques except NMSIMP (specified by the TECHNIQUE= option), termination requires a small change of the function value in successive iterations,

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

- The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}(k)$ is defined as the vertex with the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default value is $r = 0$. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

ABSGCONV=*r* **<** *n* **>**

ABSGTOL=*r***<** *n* **>**

specifies an absolute gradient convergence criterion:

- For all techniques except NMSIMP (specified by the TECHNIQUE= option), termination requires the maximum absolute gradient element to be small:

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

- This criterion is not used by the NMSIMP technique.

The default value is r = 1E–5. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

ABSXCONV=*r* **<** *n* **>**

ABSXTOL=*r***<** *n* **>**

specifies an absolute parameter convergence criterion:

- For all techniques except NMSIMP, termination requires a small Euclidean distance between successive parameter vectors,

$$\| \boldsymbol{\psi}^{(k)} - \boldsymbol{\psi}^{(k-1)} \|_2 \leq r$$

- For the NMSIMP technique, termination requires either a small length $\alpha^{(k)}$ of the vertices of a restart simplex,

  $$\alpha^{(k)} \leq r$$

  or a small simplex size,

  $$\delta^{(k)} \leq r$$

  where the simplex size $\delta^{(k)}$ is defined as the L1 distance from the simplex vertex $\boldsymbol{\xi}^{(k)}$ with the smallest function value to the other p simplex points $\boldsymbol{\psi}_l^{(k)} \neq \boldsymbol{\xi}^{(k)}$:

  $$\delta^{(k)} = \sum_{\boldsymbol{\psi}_l \neq y} \| \boldsymbol{\psi}_l^{(k)} - \boldsymbol{\xi}^{(k)} \|_1$$

The default is r = 1E–8 for the NMSIMP technique and r = 0 otherwise. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

ASINGULAR=*r*

ASING=*r*

specifies an absolute singularity criterion for the computation of the inertia (number of positive, negative, and zero eigenvalues) of the Hessian and its projected forms. The default value is the square root of the smallest positive double-precision value.

DAMPSTEP<=*r***>**

specifies that the initial step length value $\alpha^{(0)}$ for each line search (used by the QUANEW, CONGRA, or NEWRAP technique) cannot be larger than r times the step length value used in the former iteration. If the DAMPSTEP option is specified but r is not specified, the default is r = 2. The DAMPSTEP= option can prevent the line-search algorithm from repeatedly stepping into regions where some objective functions are difficult to compute or where they could lead to floating-point overflows during the computation of objective functions and their derivatives. The DAMPSTEP= option can save time-consuming function calls during the line searches of objective functions that result in very small steps.

FCONV=*r***<***n***>**

FTOL=*r***<***n***>**

specifies a relative function convergence criterion:

- For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations,

  $$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{\max(|f(\boldsymbol{\psi}^{(k-1)})|, \mathrm{FSIZE})} \leq r$$

  where FSIZE is defined by the FSIZE= option.
- The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex with the lowest function value and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default is $r = 10^{-\mathrm{FDIGITS}}$, where FDIGITS is by default $-\log_{10}\{\epsilon\}$ and ϵ is the machine precision. Some procedures, such as the GLIMMIX procedure, enable you to change the value with the FDIGITS= option in the PROC statement. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

FCONV2=*r*< *n*>
FTOL2=*r*< *n*>

specifies a second function convergence criterion:

- For all techniques except NMSIMP, termination requires a small predicted reduction,

  $$df^{(k)} \approx f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k)} + \mathrm{s}^{(k)})$$

  of the objective function. The predicted reduction

  $$\begin{aligned} df^{(k)} &= -\mathbf{g}^{(k)\prime}\mathrm{s}^{(k)} - \frac{1}{2}\mathrm{s}^{(k)\prime}\mathbf{H}^{(k)}\mathrm{s}^{(k)} \\ &= -\frac{1}{2}\mathrm{s}^{(k)\prime}\mathbf{g}^{(k)} \le r \end{aligned}$$

  is computed by approximating the objective function f by the first two terms of the Taylor series and substituting the Newton step,

  $$\mathrm{s}^{(k)} = -[\mathbf{H}^{(k)}]^{-1}\mathbf{g}^{(k)}$$

- For the NMSIMP technique, termination requires a small standard deviation of the function values of the $p+1$ simplex vertices $\boldsymbol{\psi}_l^{(k)}$, $l = 0, \ldots, p$,

  $$\sqrt{\frac{1}{n+1}\sum_l \left[f(\boldsymbol{\psi}_l^{(k)}) - \overline{f}(\boldsymbol{\psi}^{(k)})\right]^2} \le r$$

  where $\overline{f}(\boldsymbol{\psi}^{(k)}) = \frac{1}{p+1}\sum_l f(\boldsymbol{\psi}_l^{(k)})$. If there are p_{act} boundary constraints active at $\boldsymbol{\psi}^{(k)}$, the mean and standard deviation are computed only for the $n+1-p_{act}$ unconstrained vertices.

The default value is *r* = 1E–6 for the NMSIMP technique and *r*= 0 otherwise. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

FSIZE=*r*

specifies the FSIZE parameter of the relative function and relative gradient termination criteria. The default value is *r* = 0. For more details, see the FCONV= and GCONV= options.

GCONV=*r*< *n*>
GTOL=*r*< *n*>

specifies a relative gradient convergence criterion:

- For all techniques except CONGRA and NMSIMP, termination requires that the normalized predicted function reduction be small,

  $$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{\max(|f(\boldsymbol{\psi}^{(k)})|, \mathrm{FSIZE})} \le r$$

  where FSIZE is defined by the FSIZE= option. For the CONGRA technique (where a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

  $$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathrm{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \max(|f(\boldsymbol{\psi}^{(k)})|, \mathrm{FSIZE})} \le r$$

- This criterion is not used by the NMSIMP technique.

The default value is *r* = 1E–8. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

GCONV2=*r*< *n*>

GTOL2=*r*< *n*>

specifies another relative gradient convergence criterion:

- For least squares problems and the TRUREG, LEVMAR, NRRIDG, and NEWRAP techniques, the following criterion of Browne (1982) is used:

$$\max_j \frac{|\mathbf{g}_j(\boldsymbol{\psi}^{(k)})|}{\sqrt{f(\boldsymbol{\psi}^{(k)})\mathbf{H}^{(k)}_{j,j}}} \leq r$$

- This criterion is not used by the other techniques.

The default value is $r = 0$. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

HESCAL=0 | 1 | 2 | 3

HS=0 | 1 | 2 | 3

specifies the scaling version of the Hessian (or crossproduct Jacobian) matrix used in NRRIDG, TRUREG, LEVMAR, NEWRAP, or DBLDOG optimization.

If HS is not equal to 0, the first iteration and each restart iteration set the diagonal scaling matrix $D^{(0)} = \mathrm{diag}(d_i^{(0)})$:

$$d_i^{(0)} = \sqrt{\max(|H_{i,i}^{(0)}|, \epsilon)}$$

where $H_{i,i}^{(0)}$ are the diagonal elements of the Hessian (or crossproduct Jacobian). In every other iteration, the diagonal scaling matrix $D^{(0)} = \mathrm{diag}(d_i^{(0)})$ is updated depending on the HS option:

0 specifies that no scaling be done.

1 specifies the Moré (1978) scaling update:

$$d_i^{(k+1)} = \max\left[d_i^{(k)}, \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}\right]$$

2 specifies the Dennis, Gay, and Welsch (1981) scaling update:

$$d_i^{(k+1)} = \max\left[0.6 * d_i^{(k)}, \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}\right]$$

3 specifies that d_i be reset in each iteration:

$$d_i^{(k+1)} = \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}$$

In each scaling update, ϵ is the relative machine precision. The default value is HS=0. Scaling of the Hessian can be time-consuming in the case where general linear constraints are active.

INHESSIAN< =*r* >

INHESS< =*r* >

specifies how the initial estimate of the approximate Hessian is defined for the quasi-Newton techniques QUANEW and DBLDOG. There are two alternatives:

- If you do not use the *r* specification, the initial estimate of the approximate Hessian is set to the Hessian at $\boldsymbol{\psi}^{(0)}$.
- If you do use the *r* specification, the initial estimate of the approximate Hessian is set to the multiple of the identity matrix $r\mathbf{I}$.

By default (if you do not specify the option INHESSIAN=*r*), the initial estimate of the approximate Hessian is set to the multiple of the identity matrix $r\mathbf{I}$, where the scalar *r* is computed from the magnitude of the initial gradient.

INSTEP=*r*

SALPHA=*r*

RADIUS=*r*

reduces the length of the first trial step during the line search of the first iterations. For highly nonlinear objective functions, such as the EXP function, the default initial radius of the trust-region algorithm TRUREG or DBLDOG or the default step length of the line-search algorithms can result in arithmetic overflows. If this occurs, you should specify decreasing values of $0 < r < 1$ such as INSTEP=1E–1, INSTEP=1E–2, INSTEP=1E–4, and so on, until the iteration starts successfully.

- For trust-region algorithms (TRUREG or DBLDOG), the INSTEP= option specifies a factor $r > 0$ for the initial radius $\Delta^{(0)}$ of the trust region. The default initial trust-region radius is the length of the scaled gradient. This step corresponds to the default radius factor of $r = 1$.
- For line-search algorithms (NEWRAP, CONGRA, or QUANEW), the INSTEP= option specifies an upper bound for the initial step length for the line search during the first five iterations. The default initial step length is $r = 1$.
- For the Nelder-Mead simplex algorithm, by using TECH=NMSIMP, the INSTEP=*r* option defines the size of the start simplex.

LCDEACT=*r*

LCD=*r*

specifies a threshold *r* for the Lagrange multiplier that determines whether an active inequality constraint remains active or can be deactivated. For maximization, *r* must be greater than zero; for minimization, *r* must be smaller than zero. An active inequality constraint can be deactivated only if its Lagrange multiplier is less than the threshold value. The default value is

$$r = \pm \min(0.01, \max(0.1 \times \mathrm{ABSGCONV}, 0.001 \times \mathrm{gmax}^{(k)}))$$

where "+" is for maximization, "–" is for minimization, ABSGCONV is the value of the absolute gradient criterion, and $\mathrm{gmax}^{(k)}$ is the maximum absolute element of the gradient or the projected gradient.

LCEPSILON=*r*

LCEPS=*r*

LCE=*r*

specifies the range r for active and violated boundary constraints, where $r \geq 0$. If the point $\boldsymbol{\psi}^{(k)}$ satisfies the following condition, the constraint i is recognized as an active constraint:

$$|\sum_{j=1}^{k} a_{ij}\psi_j^{(k)} - b_i| \leq r \times (|b_i| + 1)$$

Otherwise, the constraint i is either an inactive inequality or a violated inequality or equality constraint. The default value is r = 1E–8. During the optimization process, the introduction of rounding errors can force the optimization to increase the value of r by a factor of 10^k for some k > 0. If this happens, it is indicated by a message displayed in the log.

LCSINGULAR=*r*

LCSING=*r*

LCS=*r*

specifies a criterion r, where $r \geq 0$, that is used in the update of the QR decomposition and that determines whether an active constraint is linearly dependent on a set of other active constraints. The default value is r = 1E–8. The larger r becomes, the more the active constraints are recognized as being linearly dependent. If the value of r is larger than 0.1, it is reset to 0.1.

LINESEARCH=*i*

LIS=*i*

specifies the line-search method for the CONGRA, QUANEW, and NEWRAP optimization techniques. See Fletcher (1987) for an introduction to line-search techniques. The value of i can be $1, \ldots, 8$ as follows. The default is LIS=2.

1 specifies a line-search method that needs the same number of function and gradient calls for cubic interpolation and cubic extrapolation; this method is similar to one used by the Harwell subroutine library.

2 specifies a line-search method that needs more function than gradient calls for quadratic and cubic interpolation and cubic extrapolation; this method is implemented as shown in Fletcher (1987) and can be modified to an exact line search by using the LSPRECISION= option. This is the default.

3 specifies a line-search method that needs the same number of function and gradient calls for cubic interpolation and cubic extrapolation; this method is implemented as shown in Fletcher (1987) and can be modified to an exact line search by using the LSPRECISION= option.

4 specifies a line-search method that needs the same number of function and gradient calls for stepwise extrapolation and cubic interpolation.

5 specifies a line-search method that is a modified version of LIS=4.

6 specifies a golden-section line search (Polak 1971), which uses only function values for linear approximation.

7 specifies a bisection line search (Polak 1971), which uses only function values for linear approximation.

8 specifies the Armijo line-search technique (Polak 1971), which uses only function values for linear approximation.

LSPRECISION=*r*
LSP=*r*

specifies the degree of accuracy that should be obtained by the line-search algorithms LIS=2 and LIS=3. Usually an imprecise line search is inexpensive and successful. For more difficult optimization problems, a more precise and expensive line search might be necessary (Fletcher 1987). The LIS=2 line-search method (which is the default for the NEWRAP, QUANEW, and CONGRA techniques) and the LIS=3 line-search method approach exact line search for small LSPRECISION= values. If you have numerical problems, try to decrease the LSPRECISION= value to obtain a more precise line search. The default values are shown in Table 20.35.

Table 20.35 Default Values for Line-Search Precision

TECH=	UPDATE=	LSP Default
QUANEW	DBFGS, BFGS	$r = 0.4$
QUANEW	DDFP, DFP	$r = 0.06$
CONGRA	All	$r = 0.1$
NEWRAP	No update	$r = 0.9$

For more details, see Fletcher (1987).

MAXFUNC=*i*
MAXFU=*i*

specifies the maximum number *i* of function calls in the optimization process. The default values are as follows:

- 125 for the TRUREG, NRRIDG, NEWRAP, and LEVMAR techniques
- 500 for the QUANEW and DBLDOG techniques
- 1000 for the CONGRA technique
- 3000 for the NMSIMP technique

Optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that is specified by the MAXFUNC= option.

MAXITER=*i*
MAXIT=*i*

specifies the maximum number *i* of iterations in the optimization process. The default values are as follows:

- 50 for the TRUREG, NRRIDG, NEWRAP, and LEVMAR techniques
- 200 for the QUANEW and DBLDOG techniques
- 400 for the CONGRA technique
- 1000 for the NMSIMP technique

These default values are also valid when *i* is specified as a missing value.

MAXSTEP=*r*< *n*>

specifies an upper bound for the step length of the line-search algorithms during the first *n* iterations. By default, *r* is the largest double-precision value and *n* is the largest integer available. Setting this option can improve the speed of convergence for the CONGRA, QUANEW, and NEWRAP techniques.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The time specified by the MAXTIME= option is checked only once at the end of each iteration. Therefore, the actual running time can be much longer than that specified by the MAXTIME= option. The actual running time includes the rest of the time needed to finish the iteration and the time needed to generate the output of the results. By default, CPU time is not limited.

MINITER=*i*

MINIT=*i*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

MSINGULAR=*r*

MSING=*r*

specifies a relative singularity criterion *r*, where $r > 0$, for the computation of the inertia (number of positive, negative, and zero eigenvalues) of the Hessian and its projected forms. The default value is 1E–12.

NOPRINT

suppresses output that is related to optimization, such as the iteration history. This option, along with all NLOPTIONS statement options for displayed output, are ignored by the GLIMMIX and HPMIXED procedures.

PALL

displays all optional output for optimization. This option is supported only by the CALIS and SURVEYPHREG procedures.

PHISTORY

PHIST

displays the optimization history. The PHISTORY option is implied if the PALL option is specified. The PHISTORY option is supported only by the CALIS and SURVEYPHREG procedures.

RESTART=*i*

REST=*i*

specifies that the QUANEW or CONGRA technique is restarted with a steepest search direction after at most *i* iterations, where $i > 0$. Default values are as follows:

- When TECHNIQUE=CONGRA and UPDATE=PB, restart is performed automatically; so *i* is not used.
- When TECHNIQUE=CONGRA and UPDATE≠PB, $i = \min(10p, 80)$, where *p* is the number of parameters.
- When TECHNIQUE=QUANEW, *i* is the largest integer available.

SINGULAR=*r*

SING=*r*

specifies the singularity criterion r, $0 \leq r \leq 1$, that is used for the inversion of the Hessian matrix. The default value is 1E–8.

SOCKET=*fileref*

specifies the fileref that contains the information needed for remote monitoring.

TECHNIQUE=*value*

TECH=*value*

OMETHOD=*value*

OM=*value*

specifies the optimization technique. You can find additional information about choosing an optimization technique in the section "Choosing an Optimization Algorithm" on page 545. Valid values for the TECHNIQUE= option are as follows:

- CONGRA
  performs a conjugate-gradient optimization, which can be more precisely specified with the UPDATE= option and modified with the LINESEARCH= option. When you specify this option, UPDATE=PB by default.
- DBLDOG
  performs a version of double-dogleg optimization, which can be more precisely specified with the UPDATE= option. When you specify this option, UPDATE=DBFGS by default.
- LEVMAR
  performs a highly stable, but for large problems memory- and time-consuming, Levenberg-Marquardt optimization technique, a slightly improved variant of the Moré (1978) implementation. You can also specify this technique with the alias LM or MARQUARDT. In the CALIS procedure, this is the default optimization technique if there are fewer than 40 parameters to estimate. The GLIMMIX and HPMIXED procedures do not support this optimization technique.
- NMSIMP
  performs a Nelder-Mead simplex optimization. The CALIS procedure does not support this optimization technique.
- NONE
  does not perform any optimization. This option can be used for the following:
  - to perform a grid search without optimization
  - to compute estimates and predictions that cannot be obtained efficiently with any of the optimization techniques
  - to obtain inferences for known values of the covariance parameters
- NEWRAP
  performs a Newton-Raphson optimization that combines a line-search algorithm with ridging. The line-search algorithm LIS=2 is the default method.
- NRRIDG
  performs a Newton-Raphson optimization with ridging. This is the default optimization technique in the SURVEYPHREG procedure.

- QUANEW
  performs a quasi-Newton optimization, which can be defined more precisely with the UPDATE= option and modified with the LINESEARCH= option.
- TRUREG
  performs a trust-region optimization.

UPDATE=*method*

UPD=*method*

specifies the update method for the quasi-Newton, double-dogleg, or conjugate-gradient optimization technique. Not every update method can be used with each optimizer.

The following are the valid methods for the UPDATE= option:

- BFGS
  performs the original Broyden, Fletcher, Goldfarb, and Shanno (BFGS) update of the inverse Hessian matrix.
- DBFGS
  performs the dual BFGS update of the Cholesky factor of the Hessian matrix. This is the default update method.
- DDFP
  performs the dual Davidon, Fletcher, and Powell (DFP) update of the Cholesky factor of the Hessian matrix.
- DFP
  performs the original DFP update of the inverse Hessian matrix.
- PB
  performs the automatic restart update method of Powell (1977) and Beale (1972).
- FR
  performs the Fletcher-Reeves update (Fletcher 1987).
- PR
  performs the Polak-Ribiere update (Fletcher 1987).
- CD
  performs a conjugate-descent update of Fletcher (1987).

VERSION=1 | 2

VS=1 | 2

specifies the version of the quasi-Newton optimization technique with nonlinear constraints.

VS=1 specifies the update of the μ vector as in Powell (1978b, a) (update like VF02AD).

VS=2 specifies the update of the μ vector as in Powell (1982b, a) (update like VMCWD).

The default is VERSION=2.

VSINGULAR=*r*

VSING=*r*

specifies a relative singularity criterion *r*, where *r* > 0, for the computation of the inertia (number of positive, negative, and zero eigenvalues) of the Hessian and its projected forms. The default value is *r* = 1E–8.

XCONV=*r*< *n* >

XTOL=*r*< *n* >

specifies the relative parameter convergence criterion:

- For all techniques except NMSIMP, termination requires a small relative parameter change in subsequent iterations:

$$\frac{\max_j |\psi_j^{(k)} - \psi_j^{(k-1)}|}{\max(|\psi_j^{(k)}|, |\psi_j^{(k-1)}|, \text{XSIZE})} \leq r$$

- For the NMSIMP technique, the same formula is used, but $\psi_j^{(k)}$ is defined as the vertex with the lowest function value and $\psi_j^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default value is *r* = 1E–8 for the NMSIMP technique and *r* = 0 otherwise. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

XSIZE=*r*

specifies the XSIZE parameter *r* of the relative parameter termination criterion, where $r \geq 0$. The default value is *r* = 0. For more details, see the XCONV= option.

Choosing an Optimization Algorithm

First- or Second-Order Algorithms

The factors that go into choosing a particular optimization technique for a particular problem are complex. Trial and error can be involved.

For many optimization problems, computing the gradient takes more computer time than computing the function value. Computing the Hessian sometimes takes *much* more computer time and memory than computing the gradient, especially when there are many decision variables. Unfortunately, optimization techniques that do not use some kind of Hessian approximation usually require many more iterations than techniques that do use a Hessian matrix, and, as a result, the total run time of these techniques is often longer. Techniques that do not use the Hessian also tend to be less reliable. For example, they can terminate more easily at stationary points than at global optima.

Table 20.36 shows which derivatives are required for each optimization technique.

Table 20.36 Derivatives Required

Algorithm	First-Order	Second-Order
LEVMAR	x	x
TRUREG	x	x
NEWRAP	x	x
NRRIDG	x	x
QUANEW	x	-
DBLDOG	x	-
CONGRA	x	-
NMSIMP	-	-

The second-derivative methods TRUREG, NEWRAP, and NRRIDG are best for small problems where the Hessian matrix is not expensive to compute. Sometimes the NRRIDG algorithm can be faster than the TRUREG algorithm, but TRUREG can be more stable. The NRRIDG algorithm requires only one matrix with $p(p+1)/2$ double words; TRUREG and NEWRAP require two such matrices. Here, p denotes the number of parameters in the optimization.

The first-derivative methods QUANEW and DBLDOG are best for medium-sized problems where the objective function and the gradient are much faster to evaluate than the Hessian. In general, the QUANEW and DBLDOG algorithms require more iterations than TRUREG, NRRIDG, and NEWRAP, but each iteration can be much faster. The QUANEW and DBLDOG algorithms require only the gradient to update an approximate Hessian, and they require slightly less memory than TRUREG or NEWRAP (essentially one matrix with $p(p+1)/2$ double words).

The first-derivative method CONGRA is best for large problems where the objective function and the gradient can be computed much faster than the Hessian and where too much memory is required to store the (approximate) Hessian. In general, the CONGRA algorithm requires more iterations than QUANEW or DBLDOG, but each iteration can be much faster. Because CONGRA requires only a factor of p double-word memory, many large applications can be solved only by CONGRA.

The no-derivative method NMSIMP is best for small problems where derivatives are not continuous or are very difficult to compute.

Each optimization method uses one or more convergence criteria that determine when it has converged. An algorithm is considered to have converged when any one of the convergence criteria is satisfied. For example, under the default settings, the QUANEW algorithm will converge if ABSGCONV < 1E–5, FCONV < $10^{-\mathrm{FDIGITS}}$, or GCONV < 1E–8.

Algorithm Descriptions

Trust Region Optimization (TRUREG)

The trust region method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function $f(\boldsymbol{\psi})$ have continuous first- and second-order derivatives inside the feasible region.

The trust region method iteratively optimizes a quadratic approximation to the nonlinear objective function within a hyperelliptic trust region with radius Δ that constrains the step size that corresponds to the quality of the quadratic approximation. The trust region method is implemented based on Dennis, Gay, and Welsch (1981); Gay (1983) and Moré and Sorensen (1983).

The trust region method performs well for small- to medium-sized problems, and it does not need many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the (dual) quasi-Newton or conjugate gradient algorithms might be more efficient.

Newton-Raphson Optimization with Line Search (NEWRAP)

The NEWRAP technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region. If second-order derivatives are computed efficiently and precisely, the NEWRAP method can perform well for medium-sized to large problems, and it does not need many function, gradient, and Hessian calls.

This algorithm uses a pure Newton step when the Hessian is positive definite and when the Newton step reduces the value of the objective function successfully. Otherwise, a combination of ridging and line search is performed to compute successful steps. If the Hessian is not positive definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive definite (Eskow and Schnabel 1991).

In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation (LIS=2).

Newton-Raphson Ridge Optimization (NRRIDG)

The NRRIDG technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

This algorithm uses a pure Newton step when the Hessian is positive definite and when the Newton step reduces the value of the objective function successfully. If at least one of these two conditions is not satisfied, a multiple of the identity matrix is added to the Hessian matrix.

The NRRIDG method performs well for small- to medium-sized problems, and it does not require many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the (dual) quasi-Newton or conjugate gradient algorithms might be more efficient.

Because the NRRIDG technique uses an orthogonal decomposition of the approximate Hessian, each iteration of NRRIDG can be slower than that of the NEWRAP technique, which works with a Cholesky decomposition. Usually, however, NRRIDG requires fewer iterations than NEWRAP.

Quasi-Newton Optimization (QUANEW)

The (dual) quasi-Newton method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$, and it does not need to compute second-order derivatives because they are approximated. It works well for medium-sized to moderately large optimization problems, where the objective function and the gradient are much faster to compute than the Hessian. However, in general, it requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which compute second-order derivatives. QUANEW is the default optimization algorithm because it provides an appropriate balance between the speed and stability required for most nonlinear mixed model applications.

The QUANEW technique is one of the following, depending upon the value of the UPDATE= option:

- the original quasi-Newton algorithm, which updates an approximation of the inverse Hessian
- the dual quasi-Newton algorithm, which updates the Cholesky factor of an approximate Hessian (this is the default)

You can specify four update formulas with the UPDATE= option:

- DBFGS performs the dual Broyden, Fletcher, Goldfarb, and Shanno (BFGS) update of the Cholesky factor of the Hessian matrix. This is the default.
- DDFP performs the dual Davidon, Fletcher, and Powell (DFP) update of the Cholesky factor of the Hessian matrix.
- BFGS performs the original BFGS update of the inverse Hessian matrix.
- DFP performs the original DFP update of the inverse Hessian matrix.

In each iteration, a line search is performed along the search direction to find an approximate optimum. The default line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit of the step size. Violating the left-side Goldstein condition can affect the positive definiteness of the quasi-Newton update. In that case, either the update is skipped or the iterations are restarted with an identity matrix, resulting in the steepest descent or ascent search direction. You can specify line-search algorithms other than the default with the LIS= option.

The QUANEW algorithm uses its own line-search technique. Of the options and parameters that control the line search for other algorithms, only the INSTEP= option applies here. In several applications, large steps in the first iterations are troublesome. You can use the INSTEP= option to impose an upper bound for the step size α during the first five iterations. You can also use the INHESSIAN= option to specify a different starting approximation for the Hessian. If you specify only the INHESSIAN option, the Cholesky factor of a (possibly ridged) finite-difference approximation of the Hessian is used to initialize the quasi-Newton update process.

Double-Dogleg Optimization (DBLDOG)

The double-dogleg optimization method combines the ideas of the quasi-Newton and trust region methods. In each iteration, the double-dogleg algorithm computes the step $\mathrm{s}^{(k)}$ as the linear combination of the steepest descent or ascent search direction $\mathrm{s}_1^{(k)}$ and a quasi-Newton search direction $\mathrm{s}_2^{(k)}$,

$$\mathrm{s}^{(k)} = \alpha_1 \mathrm{s}_1^{(k)} + \alpha_2 \mathrm{s}_2^{(k)}$$

The step is requested to remain within a prespecified trust region radius; see Fletcher (1987, p, 107). Thus, the DBLDOG subroutine uses the dual quasi-Newton update but does not perform a line search. You can specify two update formulas with the UPDATE= option:

- DBFGS performs the dual Broyden, Fletcher, Goldfarb, and Shanno update of the Cholesky factor of the Hessian matrix. This is the default.
- DDFP performs the dual Davidon, Fletcher, and Powell update of the Cholesky factor of the Hessian matrix.

The double-dogleg optimization technique works well for medium-sized to moderately large optimization problems, where the objective function and the gradient are much faster to compute than the Hessian. The implementation is based on Dennis and Mei (1979); Gay (1983), but it is extended for dealing with boundary and linear constraints. The DBLDOG technique generally requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which require second-order derivatives; however, each of the DBLDOG iterations is computationally cheap. Furthermore, the DBLDOG technique requires only gradient calls for the update of the Cholesky factor of an approximate Hessian.

Conjugate Gradient Optimization (CONGRA)

Second-order derivatives are not required by the CONGRA algorithm and are not even approximated. The CONGRA algorithm can be expensive in function and gradient calls, but it requires only $O(p)$ memory for unconstrained optimization. In general, many iterations are required to obtain a precise solution, but each of the CONGRA iterations is computationally cheap. You can specify four different update formulas for generating the conjugate directions by using the UPDATE= option:

- PB performs the automatic restart update method of Powell (1977) and Beale (1972). This is the default.
- FR performs the Fletcher-Reeves update (Fletcher 1987).
- PR performs the Polak-Ribiere update (Fletcher 1987).
- CD performs a conjugate-descent update of Fletcher (1987).

The default often behaves best for typical examples, whereas UPDATE=CD can perform poorly.

The CONGRA subroutine should be used for optimization problems with large p. For the unconstrained or boundary-constrained case, CONGRA requires only $O(p)$ bytes of working memory, whereas all other optimization methods require order $O(p^2)$ bytes of working memory. During p successive iterations, uninterrupted by restarts or changes in the working set, the conjugate gradient algorithm computes a cycle of p conjugate search directions. In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit for the step size. Other line-search algorithms can be specified with the LIS= option.

Nelder-Mead Simplex Optimization (NMSIMP)

The Nelder-Mead simplex method does not use any derivatives and does not assume that the objective function has continuous derivatives. The objective function itself needs to be continuous. This technique is quite expensive in the number of function calls, and it might be unable to generate precise results for $p \gg 40$.

The original Nelder-Mead simplex algorithm is implemented and extended to boundary constraints. This algorithm does not compute the objective for infeasible points, but it changes the shape of the simplex adapting to the nonlinearities of the objective function, which contributes to an increased speed of convergence. It uses a special termination criterion.

SLICE Statement

This statement applies to the following SAS/STAT procedures:
GEE, GENMOD, GLIMMIX, LIFEREG, LOGISTIC, MIXED, ORTHOREG, PHREG, PLM, PROBIT, RMSTREG, SURVEYLOGISTIC, SURVEYPHREG, and SURVEYREG. It also applies to the RELIABILITY procedure in SAS/QC software.

The SLICE statement is similar to the LSMEANS statement. You use it to perform inferences on model effects that consist entirely of classification variables. With the SLICE statement, these effects must be higher-order effects of at least two classification variables. The effect is then partitioned into subsets that correspond to variables used in forming the effect. You can use the same options as you use for the LSMEANS statement to perform an analysis for the partitions. This analysis is also known as an analysis of simple effects (Winer 1971).

By default, the interaction effect is partitioned by all main effects. For example, the following statements produce simple-effect differences among the A levels for each level of B and simple-effect differences among the B levels for each level of A:

```
class a b;
model y = a b a*b;
slice a*b / diff nof;
```

For example, if the *model-effect* is a three-way interaction effect, the default output includes comparisons of the two-way interaction means.

Suppose, for example, that the interaction effect A*B is significant in your analysis and that you want to test the effect of A for each level of B. The appropriate statement is

```
slice A*B / sliceBy = B;
```

This produces an F test for each level of B that compares the equality of the levels of A.

For example, assume that in a balanced design factors A and B have $a = 4$ and $b = 3$ levels, respectively. Consider the following statements:

```
class a b;
model y = a b a*b;
slice a*b / sliceby=a diff;
```

The SLICE statement produces four F tests, one per level of A. The first of these tests is constructed by extracting the three rows that correspond to the first level of A from the coefficient matrix for the A*B interaction. Call this matrix $\mathbf{L}_{a1}$ and its rows $\mathbf{l}_{a1}^{(1)}$, $\mathbf{l}_{a1}^{(2)}$, and $\mathbf{l}_{a1}^{(3)}$. The slice tests the two-degrees-of-freedom hypothesis

$$H: \begin{cases} \left(\mathbf{l}_{a1}^{(1)} - \mathbf{l}_{a1}^{(2)}\right)\boldsymbol{\beta} = 0 \\ \left(\mathbf{l}_{a1}^{(1)} - \mathbf{l}_{a1}^{(3)}\right)\boldsymbol{\beta} = 0 \end{cases}$$

In a balanced design, where μ_{ij} denotes the mean response if A is at level i and B is at level j, this hypothesis is equivalent to $H: \mu_{11} = \mu_{12} = \mu_{13}$. The DIFF option considers the three rows of $\mathbf{L}_{a1}$ in turn and performs

tests of the difference between pairs of rows. By default, all pairwise differences within the subset of **L** are considered; in the example this corresponds to tests of the form

$$H: \left(\mathbf{l}_{a1}^{(1)} - \mathbf{l}_{a1}^{(2)}\right) \boldsymbol{\beta} = 0$$
$$H: \left(\mathbf{l}_{a1}^{(1)} - \mathbf{l}_{a1}^{(3)}\right) \boldsymbol{\beta} = 0$$
$$H: \left(\mathbf{l}_{a1}^{(2)} - \mathbf{l}_{a1}^{(3)}\right) \boldsymbol{\beta} = 0$$

In the example, with $a = 4$ and $b = 3$, this produces four sets of least squares means differences. Within each set, factor A is held fixed at a particular level and each set consists of three comparisons.

Syntax: SLICE Statement

SLICE *model-effect* < / *options* > **;**

You can specify all options of the LSMEANS statement in the SLICE statement. The philosophy of the SLICE statement is to apply the analysis according to the options to the subsets of the **L** matrix that correspond to chosen partitions.

The following behavior differences between the SLICE and the LSMEANS statement are noteworthy:

- The specification of the *model-effect* is optional in the LSMEANS statement and required in the SLICE statement.
- Only a single SLICE *model-effect* can be specified before the option slash (/). However, you can specify multiple partitioning rules with the SLICEBY option.
- The MEANS option is the default for most procedures in the LSMEANS statement. For the SLICE statement, the default is the NOMEANS option.

Also, the three generalized linear modeling options: EXP, ILINK, and ODDSRATIO in the SLICE statement are additionally supported by PROC GLIMMIX and by PROC PLM when it is used to perform statistical analyses on item stores that were created by PROC GLIMMIX.

In addition to the options in the LSMEANS statement, you can specify the following options in the SLICE statement after the slash (/):

SLICEBY < = > *slice-specification*

SIMPLE < = > *slice-specification*

SLICEBY(*slice-specification* < , *slice-specification* < , . . . > >**)**

SIMPLE(*slice-specification* < , *slice-specification* < , . . . > >**)**

determines how to construct the partition of the least squares means for the *model-effect*. A *slice-specification* consists of an effect name followed by an optional list of formatted values. For example, the following statements creates partitions of the A*B interaction effect for all levels of variable A:

```
class a b;
model y = a b a*b;
slice a*b / sliceby=a;
```

The following statements produces two partitions of the interaction:

```
class a b;
model y = a b a*b;
slice a*b / sliceby(b='2' a='1') diff;
```

In the first partition the variable B takes on formatted value ‘2’. In the second partition the variable A takes on the formatted value ‘1’.

NOF

suppresses the F test for testing the mutual equality of the estimable functions in the partition.

ODS Table Names: SLICE Statement

Each table created by the SLICE statement has a name associated with it, and you can use this name to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 20.37. For more information about ODS, see Chapter 23, “Using the Output Delivery System.”

Table 20.37 ODS Tables Produced by the SLICE Statement

Table Name	Description	Required Option
Coef	**L** matrix coefficients	E
Slices	LS-means slices	MEANS
SliceDiffs	Simple differences of LS-means slices	DIFF or ADJUST= or STEPDOWN or NOF
SliceLines	Lines display for LS-means slices	LINES
SliceTests	Tests for LS-means slices	Default

ODS Graphics: SLICE Statement

This section describes the use of ODS Graphics for creating graphics that are related to LS-means in procedures that support the common SLICE statement. There are two groups of available plots: those that can be produced by all procedures that support this statement, and those that can be produced only in association with the three procedures that can perform Bayesian analysis (PROC GENMOD, PROC LIFEREG, and PROC PHREG). Plots that are associated with the Bayesian analysis are available via these procedures directly, and also by using PROC PLM with an item store that was created by these procedures.

Plots in the first group depict the LS-means and their differences; when LS-mean comparisons are adjusted for multiplicity by using the ADJUST= option, the plots are adjusted accordingly. To request plots in this group, ODS Graphics must be enabled and you must request plots with the appropriate PLOTS= option in the SLICE statement. Plots in the second group depict the posterior sample distribution of LS-means and their differences. To request plots in this group, you must also use a BAYES statement with PROC GENMOD, PROC PHREG, or PROC LIFEREG, or you must use PROC PLM to perform statistical analysis on an item store that was saved from a Bayesian analysis.

For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." The available graphs are summarized in Table 20.38 and Table 20.39.

Table 20.38 Graphs Produced by All Procedures That Support the Common SLICE Statement

ODS Graph Name	Plot Description	Required Option
AnomPlot	Requests an analysis of means display in which least squares means are compared to an average least squares mean	PLOTS=ANOM
ControlPlot	Requests a display in which least squares means are compared to a reference level	PLOTS=CONTROL
DiffPlot	Displays all pairwise least squares mean differences and their significance. This plot is also known as a "mean-mean scatter plot" when based on arithmetic means.	PLOTS=DIFF
LinesPlot	Displays comparisons between least squares means, with lines that indicate nonsignificant subsets	PLOTS=LINESPLOT
MeanPlot	Displays least squares means	PLOTS=MEANPLOT

Table 20.39 Graphs Produced by Procedures That Support the SLICE Statement and Bayesian Analysis

ODS Graph Name	Plot Description	Required Option
BoxPlot	Displays box plots of LS-means or LS-mean differences across a posterior sample	PLOTS=BOXPLOT
DistPanel	Displays panels of histograms with kernel density curves overlaid. Each plot contains the results for the posterior sample of each LS-mean or LS-mean difference	PLOTS=DISTPLOT
DistPlot	Displays a histogram with a kernel density curve overlaid. The plot contains the results for the posterior sample of the LS-mean or LS-mean difference.	PLOTS=DISTPLOT(UNPACK)
LinesPlot	Displays a plot of LS-mean comparison lines	LINES

For details about the *plot-options* of the SLICE statement, see the PLOTS= option in the section "LSMEANS Statement" on page 492. For more details about the DIFFPLOT in particular, see the section "Graphics for LS-Mean Comparisons" on page 3997 in Chapter 52, "The GLIMMIX Procedure."

STORE Statement

This statement applies to the following SAS/STAT procedures:
GEE, GENMOD, GLIMMIX, GLM, GLMSELECT, LIFEREG, LOGISTIC, MIXED, ORTHOREG, PHREG, PROBIT, REG, RMSTREG, SURVEYLOGISTIC, SURVEYPHREG, and SURVEYREG. It also applies to the RELIABILITY procedure in SAS/QC software.

The STORE statement requests that the procedure save the context and results of the statistical analysis into an item store. An item store is a binary file format that cannot be modified by the user. The contents of the item store can be processed with the PLM procedure. One example of item store technology is to perform a time-consuming analysis and to store its results by using the STORE statement. At a later time you can then perform specific statistical analysis tasks based on the saved results of the previous analysis, without having to fit the model again. The following statements show an example in which a mixed model is fit with the MIXED procedure and the postprocessing analysis is performed with the PLM procedure:

```
proc mixed data=MyBigDataSet;
   class Env A B sub;
   model y = A B x / ddfm=KenwardRoger;
   random int A*B / sub=Env;
```

```
   repeated / subject=Env*A*B type=AR(1);
   store sasuser.mixed;
run;

proc plm restore=sasuser.mixed;
   show cov Parms;
   lsmeans A B / diff;
   score data=NewData out=ScoreResults;
run;
```

The STORE statement in the PROC MIXED step requests that the MIXED procedure save those results that are needed to perform statistical tasks with the PLM procedure. For example, the MIXED procedure saves the necessary pieces of information that relate to the Kenward-Roger degree-of-freedom method. The results from the LSMEANS statement in the PROC PLM step thus apply this technique for calculating denominator degrees of freedom. The SHOW statement in the PLM procedure reveals the contents of the item store in terms of ODS tables, and the SCORE statement computes predicted values in a new data set. For more information about postprocessing tasks based on item stores, see the documentation for the PLM procedure.

NOTE: Item stores are not portable between operating systems and machines of different types. For example, if you create an item store in a Linux environment, you cannot use it in a Windows environment. Similarly, if you create an item store on a 32-bit system, you cannot use it on a 64-bit system.

Syntax: STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The *item-store-name* is a usual one- or two-level SAS name, like the names that are used for SAS data sets. If you specify a one-level name, then the item store resides in the WORK library and is deleted at the end of the SAS session. Since item stores usually are used to perform postprocessing tasks, typical usage specifies a two-level name of the form libname.membername.

If an item store by the same name as specified in the STORE statement already exists, the existing store is replaced.

You can add a custom label with the LABEL= option in the STORE statement after the slash (/). When the PLM procedure processes an item store, the label appears in the PROC PLM output along with other identifying information.

TEST Statement

This statement documentation applies to the following procedures:
ICPHREG, LIFEREG, ORTHOREG, PLM, PROBIT, RMSTREG, SURVEYPHREG, and SURVEYREG. It also applies to the RELIABILITY procedure in SAS/QC software.

The TEST statement enables you to perform *F* tests for model effects that test Type I, II, or Type III hypotheses. See Chapter 16, "The Four Types of Estimable Functions," for details about the construction of Type I, II, and III estimable functions.

Syntax: TEST Statement

TEST < *model-effects* > < / *options* > **;**

Table 20.40 summarizes options in the TEST statement.

Table 20.40 TEST Statement Options

Option	Description
CHISQ	Requests chi-square tests
DDF=	Specifies denominator degrees of freedom for fixed effects
E	Requests Type I, Type II, and Type III coefficients
E1	Requests Type I coefficients
E2	Requests Type II coefficients
E3	Requests Type III coefficients
HTYPE=	Indicates the type of hypothesis test to perform
INTERCEPT	Adds a row that corresponds to the overall intercept

You can specify the following options in the TEST statement after the slash (/):

CHISQ
requests that chi-square tests be performed for the relevant effects in addition to the *F* tests. Type III tests are the default; you can produce the Type I and Type II tests by using the HTYPE= option. This option has no effect when the procedure produces chi-square statistics by default.

DDF=*value-list*
DF=*value-list*
specifies the denominator degrees of freedom for the fixed effects. The *value-list* specification is a list of numbers or missing values (.) separated by commas. The order of degrees of freedom should match the order of the fixed effects that are specified in the TEST statement; otherwise it should match the order in which the effects appear in the "Type III Tests of Fixed Effects" table. If you want to retain the default degrees of freedom for a particular effect, use a missing value for its location in the list. In the following example, the first TEST statement assigns 3 denominator degrees of freedom to A and

4.7 to A*B, while those for B remain the same, and the second TEST statement assigns 5 denominator degrees of freedom to A and uses the default degrees of freedom for B.

```
model Y = A B A*B;
test  / ddf=3,.,4.7;
test  B A / ddf=.,5;
```

E

requests that Type I, Type II, and Type III **L** matrix coefficients be displayed for all relevant effects.

E1 | EI

requests that Type I **L** matrix coefficients be displayed for all relevant effects.

E2 | EII

requests that Type II **L** matrix coefficients be displayed for all relevant effects.

E3 | EIII

requests that Type III **L** matrix coefficients be displayed for all relevant effects.

HTYPE=*value-list*

indicates the type of hypothesis test to perform on the fixed effects. Valid entries for values in the *value-list* are 1, 2, and 3, which correspond to Type I, Type II, and Type III tests, respectively. The default value is 3.

INTERCEPT

INT

adds a row to the tables for Type I, II, and III tests that correspond to the overall intercept.

ODS Table Names: TEST Statement

Each table created by the TEST statement has a name associated with it, and you can use this name to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 20.41. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 20.41 ODS Tables Produced by the TEST Statement

Table Name	Description	Required Option
Coef	**L** matrix coefficients	E
Tests1	Type I tests of fixed effects	HTYPE=1
Tests2	Type II tests of fixed effects	HTYPE=2
Tests3	Type III tests of fixed effects	Default

Programming Statements

This section applies to the following procedures:
CALIS, GLIMMIX, MCMC, NLIN, NLMIXED, PHREG, and SURVEYPHREG.

The majority of the SAS/STAT modeling procedures can take advantage of the fact that the statistical model can easily be translated into programming syntax (statements and options). However, several procedures require additional flexibility in specifying models—for example, when the model contains general nonlinear functions, when it is necessary to specify complicated restrictions, or when user-supplied expressions need to be evaluated. Procedures that are listed at the beginning of the section support—in addition to the usual procedure statements and options—programming statements that can be used in the SAS DATA step.

The following are valid statements:

```
ABORT;
ARRAY arrayname < [ dimensions ] > < $ > < variables-and-constants >;
CALL name < (expression < , expression . . . >) >;
DELETE;
DO < variable = expression < TO expression > < BY expression > >
    < , expression < TO expression > < BY expression > > . . .
    < WHILE expression > < UNTIL expression >;
END;
GOTO statement-label;
IF expression;
IF expression THEN program-statement;
                ELSE program-statement;
variable = expression;
variable + expression;
LINK statement-label;
PUT < variable > < = > . . . ;
RETURN;
SELECT < (expression) >;
STOP;
SUBSTR(variable, index, length)= expression;
WHEN (expression)program-statement;
        OTHERWISE program-statement;
```

For the most part, these programming statements work the same as they do in the SAS DATA step, as documented in *SAS Programmers Guide: Essentials*. However, there are several differences:

- The ABORT statement does not allow any arguments.
- The DO statement does not allow a character index variable. Thus

```
do i = 1,2,3;
```

is supported, whereas the following statement is not supported:

```
do i = 'A','B','C';
```

- Not all procedures support LAG functionality. For example, the GLIMMIX procedure does not support lags.
- The PUT statement, used mostly for program debugging, supports only some of the features of the DATA step PUT statement, and it has some features that are not available with the DATA step PUT statement:
  - The PUT statement does not support line pointers, factored lists, iteration factors, overprinting, _INFILE_, the colon (:) format modifier, or "$".
  - The PUT statement does support expressions, but the expression must be enclosed in parentheses. For example, the following statement displays the square root of x:

    ```
    put (sqrt(x));
    ```

  - The PUT statement supports the item _PDV_ to display a formatted listing of all variables in the program. For example:

    ```
    put _pdv_;
    ```

- The WHEN and OTHERWISE statements enable you to specify more than one target statement. That is, DO/END groups are not necessary for multiple-statement WHENs. For example, the following syntax is valid:

```
select;
   when (exp1) stmt1;
               stmt2;
   when (exp2) stmt3;
               stmt4;
end;
```

- The LINK statement is used in a program to jump immediately to the label *statement-label* and to continue program execution at that point. It is not used to specify a link function in a generalized linear model.

Please consult the individual chapters for other, procedure-specific differences between programming statements and the SAS DATA step and for procedure-specific details, limitations, and rules.

When coding your programming statements, avoid defining variables that begin with an underscore (_), because they might conflict with internal variables that are created by procedures that support programming statements.

Convergence Status

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icomcs.sas.)

Most procedures that iterate create a table named ConvergenceStatus. The table contains a numeric column named Status and a character column named Reason. When iterations complete successfully, the status is 0. Values greater than 0 indicate something other than a successful convergence. Larger values indicate more severe problems. Different procedures produce different values for different reasons. Consult the Reason column and the SAS log for more information about nonzero values. Batch users can create an output data set from the ConvergenceStatus table and consult it to determine how to proceed with the analysis.

Sample values include the following:

0 converged; everything seemed fine
1 converged, but there was some minor problem or a reason to question the results
2 the iteration limit was reached without convergence, but everything seemed fine
3 an error prevented further iteration

The following steps illustrate how to display the convergence status:

```
data heights;
   input Family Gender $ Height @@;
   datalines;
1 F 67   1 F 66   1 F 64   1 M 71   1 M 72   2 F 63
2 F 63   2 F 67   2 M 69   2 M 68   2 M 70   3 F 63
3 M 64   4 F 67   4 F 66   4 M 67   4 M 67   4 M 69
;

proc mixed data=heights method=ml;
   ods output convergencestatus=cs;
   class Family Gender;
   model Height = Gender Family Family*Gender;
   repeated / type=un subject=family r;
run;

proc print data=cs;
   id status;
run;
```

The iteration history and convergence status table are displayed in Figure 20.3.5. The convergence status data set is displayed in Figure 20.3.6. The nonzero status indicates that there is a problem with the results.

Output 20.3.5 Iteration History and Convergence Status

The Mixed Procedure

Iteration History			
Iteration	**Evaluations**	**-2 Log Like**	**Criterion**
0	1	53.85649943	
1	2	59.68919298	0.21180916
2	1	56.39276784	0.47136098
3	1	50.81678998	6.59294844
4	1	47.19494707	92.10547924
5	1	44.30542619	804.32745158
6	1	40.83113890	11985.500198
7	1	37.08565532	323091.35113
8	1	32.24827115	692348728.58
9	1	30.79273954	4077848869.8
10	1	29.45204457	36428915810
11	1	28.13777320	367535832119
12	1	27.68391595	833511081505
13	1	27.58830036	991414360416
14	1	27.13471632	2.2672426E12
15	1	25.82899385	2.5300169E13
16	1	25.82606678	2.5434571E13
17	1	25.82582168	2.5451822E13
18	1	25.82541602	2.5468523E13
19	1	25.37222006	5.9337171E13
20	1	25.36073787	6.0621696E13
21	1	25.26509956	7.2484928E13
22	1	24.81204801	1.6959246E14
23	1	24.81196642	1.6959267E14
24	1	24.61066215	2.475507E14
25	1	24.60926992	2.4821615E14
26	1	24.15582290	5.8319199E14
27	1	24.06060182	6.9850403E14
28	1	22.75445239	8.2952665E15
29	20	22.75445239	8.2952665E15
30	20	22.75445239	8.2952665E15
31	20	22.75445239	8.2952665E15
32	20	22.75445239	8.2952665E15
33	20	22.75445239	8.2952665E15
34	20	22.75445239	8.2952665E15

WARNING: Stopped because of too many likelihood evaluations.

Output 20.3.6 Convergence Status Data Set

Status	Reason	pdG	pdH
1	WARNING: Stopped because of too many likelihood evaluations.	1	1

The results converge in the following example:

```
proc mixed data=heights method=ml;
   ods output convergencestatus=cs;
   class Family Gender;
   model Height = Gender Family Family*Gender;
   repeated / type=ar(1) subject=family r;
run;

proc print data=cs;
   id status;
run;
```

The iteration history, convergence status table, and convergence status data set are displayed in Figure 20.3.7.

Output 20.3.7 Iteration History and Convergence Status Table and Data Set

The Mixed Procedure

Iteration History

Iteration	Evaluations	-2 Log Like	Criterion
0	1	53.85649943	
1	2	53.46585196	0.00000000

Convergence criteria met.

Status	Reason	pdG	pdH
0	Convergence criteria met.	1	1

References

Afifi, A. A., and Azen, S. P. (1972). *Statistical Analysis: A Computer-Oriented Approach*. New York: Academic Press.

Beale, E. M. L. (1972). "A Derivation of Conjugate Gradients." In *Numerical Methods for Nonlinear Optimization*, edited by F. A. Lootsma, 39–43. London: Academic Press.

Browne, M. W. (1982). "Covariance Structures." In *Topics in Applied Multivariate Analysis*, edited by D. M. Hawkins, 72–141. Cambridge: Cambridge University Press.

Chang, I.-M., Gelman, R., and Pagano, M. (1982). "Corrected Group Prognostic Curves and Summary Statistics." *Journal of Chronic Diseases* 35:669–674.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981). "An Adaptive Nonlinear Least-Squares Algorithm." *ACM Transactions on Mathematical Software* 7:348–368.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Dunnett, C. W. (1980). "Pairwise Multiple Comparisons in the Unequal Variance Case." *Journal of the American Statistical Association* 75:796–800.

Edwards, D., and Berry, J. J. (1987). "The Efficiency of Simulation-Based Multiple Comparisons." *Biometrics* 43:913–928.

Eskow, E., and Schnabel, R. B. (1991). "Algorithm 695: Software for a New Modified Cholesky Factorization." *ACM Transactions on Mathematical Software* 17:306–312.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Fox, J. (1987). "Effect Displays for Generalized Linear Models." In *Sociological Methodology*, edited by C. C. Clogg, 347–361. Washington, DC: American Sociological Association.

Frankel, S. A. (1961). "Statistical Design of Experiments for Process Development of MBT." *Rubber Age* 89:453–458.

Friendly, M. (2000). *Visualizing Categorical Data*. Cary, NC: SAS Institute Inc.

Games, P. A., and Howell, J. F. (1976). "Pairwise Multiple Comparison Procedures with Unequal *n*'s and/or Variances: A Monte Carlo Study." *Journal of Educational Statistics* 1:113–125.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Goodnight, J. H., and Harvey, W. R. (1978). *Least-Squares Means in the Fixed-Effects General Linear Models*. Technical Report R-103, SAS Institute Inc., Cary, NC.

Graubard, B. I., and Korn, E. L. (1999). "Predictive Margins with Survey Data." *Biometrics* 55:652–659.

Guirguis, G. H., and Tobias, R. D. (2004). "On the Computation of the Distribution for the Analysis of Means." *Communications in Statistics—Simulation and Computation* 33:861–888.

Harrell, F. E., Jr. (2001). *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. New York: Springer.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Holm, S. (1979). "A Simple Sequentially Rejective Multiple Test Procedure." *Scandinavian Journal of Statistics* 6:65–70.

Hsu, J. C. (1992). "The Factor Analytic Approach to Simultaneous Inference in the General Linear Model." *Journal of Computational and Graphical Statistics* 1:151–168.

Hsu, J. C. (1996). *Multiple Comparisons: Theory and Methods*. London: Chapman & Hall.

Hsu, J. C., and Peruggia, M. (1994). "Graphical Representation of Tukey's Multiple Comparison Method." *Journal of Computational and Graphical Statistics* 3:143–161.

Kenward, M. G., and Roger, J. H. (1997). "Small Sample Inference for Fixed Effects from Restricted Maximum Likelihood." *Biometrics* 53:983–997.

Kramer, C. Y. (1956). "Extension of Multiple Range Tests to Group Means with Unequal Numbers of Replications." *Biometrics* 12:307–310.

Kutner, M. H. (1974). "Hypothesis Testing in Linear Models (Eisenhart Model)." *American Statistician* 28:98–100.

Lane, P. W., and Nelder, J. A. (1982). "Analysis of Covariance and Standardization as Instances of Prediction." *Biometrics* 38:613–621.

Moré, J. J. (1978). "The Levenberg-Marquardt Algorithm: Implementation and Theory." In *Lecture Notes in Mathematics*, vol. 30, edited by G. A. Watson, 105–116. Berlin: Springer-Verlag.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Myers, R. H. (1976). *Response Surface Methodology*. Blacksburg: Virginia Polytechnic Institute and State University.

Nelson, P. R. (1982). "Exact Critical Points for the Analysis of Means." *Communications in Statistics—Theory and Methods* 11:699–709.

Nelson, P. R. (1991). "Numerical Evaluation of Multivariate Normal Integrals with Correlations $\rho_{lj} = -\alpha_l \alpha_j$." In *Frontiers of Statistical Scientific Theory and Industrial Applications: Proceedings of the ICOSCO I Conference*, edited by A. Öztürk and E. C. van der Meulen, 97–114. Columbus, OH: American Sciences Press.

Nelson, P. R. (1993). "Additional Uses for the Analysis of Means and Extended Tables of Critical Values." *Technometrics* 35:61–71.

Ott, E. R. (1967). "Analysis of Means: A Graphical Procedure." *Industrial Quality Control* 24:101–109. Reprinted in *Journal of Quality Technology* 15 (1983): 10–18.

Polak, E. (1971). *Computational Methods in Optimization*. New York: Academic Press.

Powell, M. J. D. (1977). "Restart Procedures for the Conjugate Gradient Method." *Mathematical Programming* 12:241–254.

Powell, M. J. D. (1978a). "Algorithms for Nonlinear Constraints That Use Lagrangian Functions." *Mathematical Programming* 14:224–248.

Powell, M. J. D. (1978b). "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations." In *Lecture Notes in Mathematics*, vol. 630, edited by G. A. Watson, 144–175. Berlin: Springer-Verlag.

Powell, M. J. D. (1982a). "Extensions to Subroutine VF02AD." In *Systems Modeling and Optimization, Lecture Notes in Control and Information Sciences*, vol. 38, edited by R. F. Drenick and F. Kozin, 529–538. Berlin: Springer-Verlag.

Powell, M. J. D. (1982b). *VMCWD: A Fortran Subroutine for Constrained Optimization*. Technical Report DAMTP 1982/NA4, Department of Applied Mathematics and Theoretical Physics, University of Cambridge.

Royen, T. (1989). "Generalized Maximum Range Tests for Pairwise Comparisons of Several Populations." *Biometrical Journal* 31:905–929.

Shaffer, J. P. (1986). "Modified Sequentially Rejective Multiple Test Procedures." *Journal of the American Statistical Association* 81:826–831.

Silvapulle, M. J., and Sen, P. K. (2004). *Constrained Statistical Inference: Order, Inequality, and Shape Constraints*. New York: John Wiley & Sons.

Tamhane, A. C. (1979). "A Comparison of Procedures for Multiple Comparisons of Means with Unequal Variances." *Journal of the American Statistical Association* 74:471–480.

Westfall, P. H. (1997). "Multiple Testing of General Contrasts Using Logical Constraints and Correlations." *Journal of the American Statistical Association* 92:299–306.

Westfall, P. H., Tobias, R. D., Rom, D., Wolfinger, R. D., and Hochberg, Y. (1999). *Multiple Comparisons and Multiple Tests Using the SAS System*. Cary, NC: SAS Institute Inc.

Westfall, P. H., and Young, S. S. (1993). *Resampling-Based Multiple Testing: Examples and Methods for* p-*Value Adjustment*. New York: John Wiley & Sons.

Winer, B. J. (1971). *Statistical Principles in Experimental Design*. 2nd ed. New York: McGraw-Hill.

Chapter 21

Shared Concepts in High-Performance Computing

Contents

Overview 567
Single-Machine Mode 567
Single-Machine Data Access Mode 568
Output Data Sets 568
Working with Formats 568
PERFORMANCE Statement 569

Overview

This chapter describes common concepts that are used by SAS/STAT high-performance procedures. It also provides the syntax for the PERFORMANCE statement, which is common to all high-performance analytical procedures.

Single-Machine Mode

Single-machine mode is a computing model in which multiple processors or multiple cores are controlled by a single operating system and can access shared resources, such as disks and memory. In this book, single-machine mode refers to an application running multiple concurrent threads on a multicore machine in order to take advantage of parallel execution on multiple processing units. More simply, single-machine mode for high-performance analytical procedures means multithreading on the client machine.

All high-performance analytical procedures run in single-machine mode. The procedure uses the number of CPUs (cores) on the machine to determine the number of concurrent threads. High-performance analytical procedures use different methods to map core count to the number of concurrent threads, depending on the analytic task. Using one thread per core is not uncommon for the procedures that implement data-parallel algorithms.

Single-Machine Data Access Mode

When high-performance analytical procedures run in single-machine mode, they access data in the same way as traditional SAS procedures. They use Base SAS to access input and output SAS data sets.

Output Data Sets

Many procedures in SAS software add the variables from the input data set when an observationwise output data set is created. The assumption of high-performance analytical procedures is that the input data sets can be large and contain many variables. For performance reasons, the output data set contains the following:

- variables that are explicitly created by the statement
- variables that are listed in the ID statement, as described in Chapter 22, "Shared Concepts and Topics in High-Performance Statistical Procedures"
- distribution keys or hash keys that are transferred from the input data set

Including this information enables you to add to the output data set information necessary for subsequent SQL joins without copying the entire input data set to the output data set.

Working with Formats

You can use SAS formats and user-defined formats with high-performance analytical procedures as you can with other procedures in the SAS System.

High-performance analytical procedures examine the variables that are used in an analysis for association with user-defined formats. If you are running multiple high-performance analytical procedures in a SAS session and the analysis variables depend on user-defined formats, you can preprocess the formats. This step involves generating an XML stream (a file) of the formats and passing the stream to the high-performance analytical procedures.

Suppose that the following formats are defined in your SAS program:

```
proc format;
     value YesNo        1='Yes'        0='No';
     value checkThis    1='ThisisOne'  2='ThisisTwo';
     value $cityChar    1='Portage'    2='Kinston';
run;
```

The next group of SAS statements create the XML stream for the formats in the file *Myfmt.xml*, associate that file with the file reference myxml, and pass the file reference with the FMTLIBXML= option in the PROC HPLOGISTIC statement:

```
filename myxml 'Myfmt.xml';
libname  myxml XML92 xmltype=sasfmt tagset=tagsets.XMLsuv;
proc format cntlout=myxml.allfmts;
run;


proc hplogistic data=six fmtlibxml=myxml;
   class wheeze cit age;
   format wheeze best4. cit $cityChar.;
   model wheeze = cit age;
run;
```

Generation and destruction of the stream can be wrapped in convenience macros:

```
%macro Make_XMLStream(name=tempxml);
    filename &name 'fmt.xml';
    libname  &name XML92 xmltype=sasfmt tagset=tagsets.XMLsuv;
    proc format cntlout=&name..allfmts;
    run;
%mend;

%macro Delete_XMLStream(fref);
    %let rc=%sysfunc(fdelete(&fref));
%mend;
```

If you do not pass an XML stream to a high-performance analytical procedure that supports the FMTLIBXML= option, the procedure generates an XML stream as needed when it is invoked.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing, and requests detailed results about the performance characteristics of a high-performance analytical procedure.

You can specify the following *performance-options* in the PERFORMANCE statement:

BPC=*n*

specifies the number of bytes per character that is used in processing character strings in multibyte encodings. The default is the bytes per character of the encoding. The number of characters in a string is calculated as the byte length of the string divided by the bytes per character of the encoding. This will be incorrect when the strings in the multibyte encoding contain one or more single byte characters. In such cases, setting BPC=1 enables appropriate byte lengths to be used in processing such strings.

DETAILS

requests a table that shows a timing breakdown of the procedure steps.

NTHREADS=*n*

THREADS=*n*

specifies the number of threads for analytic computations and overrides the SAS system option THREADS | NOTHREADS. If you do not specify the NTHREADS= option, the number of threads is determined based on the number of CPUs on the host on which the analytic computations execute. The algorithm by which a CPU count is converted to a thread count is specific to the high-performance analytical procedure. Most procedures create one thread per CPU for the analytic computations.

By default, high-performance analytical procedures run in multiple concurrent threads unless multithreading has been turned off by the NOTHREADS system option or you force single-threaded execution by specifying NTHREADS=1. The largest number that can be specified for *n* is 256. Individual high-performance analytical procedures can impose more stringent limits if called for by algorithmic considerations.

Chapter 22
Shared Concepts and Topics in High-Performance Statistical Procedures

Contents

Common Features of SAS High-Performance Statistical Procedures **572**
Syntax Common to SAS High-Performance Statistical Procedures **572**
CLASS Statement 572
FREQ Statement 576
ID Statement 576
SELECTION Statement 576
VAR Statement 582
WEIGHT Statement 582
Model Selection **582**
Methods 582
Full Model Fitted 582
Forward Selection 582
Backward Elimination 585
Stepwise Selection 585
Forward-Swap Selection 587
Least Angle Regression 588
Lasso Selection 588
Adaptive Lasso Selection 589
References **589**

Common Features of SAS High-Performance Statistical Procedures

SAS high-performance statistical procedures behave in many ways like other procedures in the SAS System. This chapter provides details about and describes common syntax elements that are supported by many high-performance statistical procedures. Any deviation by a high-performance statistical procedure from the common syntax is documented in the specific chapter for the procedure.

Syntax Common to SAS High-Performance Statistical Procedures

CLASS Statement

CLASS *variable* < (*options*) >. . . < *variable* < (*options*) > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. These variables enter the analysis not through their values, but through levels to which the unique values are mapped. For more information about these mappings, see the section "Levelization of Classification Variables" on page 409.

If a CLASS statement is specified, it must precede the MODEL statement in high-performance statistical procedures that support a MODEL statement.

If the procedure permits a classification variable as a response (dependent variable or target), the response does not need to be specified in the CLASS statement.

You can specify options either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*.

You can specify the following values for either an *option* or a *global-option* (except for the HPLMIXED procedure, which does not support options in this statement):

DESCENDING

DESC

reverses the sort order of the classification variable. If both the DESCENDING and ORDER= options are specified, high-performance statistical procedures order the categories according to the ORDER= option and then reverse that order.

ORDER=DATA | FORMATTED | INTERNAL

ORDER=FREQ | FREQDATA | FREQFORMATTED | FREQINTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

Table 22.1 shows how high-performance statistical procedures interpret values of the ORDER= option.

Table 22.1 Sort Order

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count (levels that have more observations come earlier in the order)
FREQDATA	Order of descending frequency count, and within counts by order of appearance in the input data set when counts are tied
FREQFORMATTED	Order of descending frequency count, and within counts by formatted value when counts are tied
FREQINTERNAL	Order of descending frequency count, and within counts by unformatted (internal) value when counts are tied
INTERNAL	Unformatted value

For more information about sort order, see the chapter about the SORT procedure in *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REF=*'level'* | *keyword*

REFERENCE=*'level'* | *keyword*

specifies the reference level that is used when you specify PARAM=REFERENCE. For an individual (but not a global) variable REF= *option*, you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a REF= *option* or *global-option*, you can use one of the following *keywords*. The default is REF=LAST.

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

If you choose a reference level for any CLASS variable, all variables are parameterized in the reference parameterization for computational efficiency. In other words, high-performance statistical procedures apply a single parameterization method to all classification variables.

Suppose that the variable temp has three levels (`'hot'`, `'warm'`, and `'cold'`) and that the variable gender has two levels (`'M'` and `'F'`). The following statements fit a logistic regression model:

```
proc hplogistic;
   class gender(ref='F') temp;
   model y = gender gender*temp;
run;
```

Both CLASS variables are in reference parameterization in this model. The reference levels are `'F'` for the variable gender and `'warm'` for the variable temp, because the statements are equivalent to the following statements:

```
proc hplogistic;
   class gender(ref='F') temp(ref=last);
   model y = gender gender*temp;
run;
```

SPLIT

requests that the columns of the design matrix that correspond to any effect that contains a split classification variable can be selected to enter or leave a model independently of the other design columns of that effect. This option is specific to the HPREG procedure.

Suppose that the variable temp has three levels (`'hot'`, `'warm'`, and `'cold'`), that the variable gender has two levels (`'M'` and `'F'`), and that the variables are used in a PROC HPREG run as follows:

```
proc hpreg;
   class temp gender / split;
   model y = gender gender*temp;
run;
```

The two effects in the MODEL statement are split into eight independent effects. The effect "gender" is split into two effects that are labeled "gender_M" and "gender_F". The effect "gender*temp" is split into six effects that are labeled "gender_M*temp_hot", "gender_F*temp_hot", "gender_M*temp_warm", "gender_F*temp_warm", "gender_M*temp_cold", and "gender_F*temp_cold". The previous PROC HPREG step is equivalent to the following:

```
proc hpreg;
   model y = gender_M gender_F
             gender_M*temp_hot  gender_F*temp_hot
             gender_M*temp_warm gender_F*temp_warm
             gender_M*temp_cold gender_F*temp_cold;
run;
```

The SPLIT option can be used on individual classification variables. For example, consider the following PROC HPREG step:

```
proc hpreg;
   class temp(split) gender;
   model y = gender gender*temp;
run;
```

In this case, the effect "gender" is not split and the effect "gender*temp" is split into three effects, which are labeled "gender*temp_hot", "gender*temp_warm", and "gender*temp_cold". Furthermore, each of these three split effects now has two parameters that correspond to the two levels of "gender." The PROC HPREG step is equivalent to the following:

```
proc hpreg;
   class gender;
   model y = gender gender*temp_hot gender*temp_warm gender*temp_cold;
run;
```

You can specify the following *global-options*:

MISSING

treats missing values (".", ".A", . . . , ".Z" for numeric variables and blanks for character variables) as valid values for the CLASS variable.

If you do not specify the MISSING option, observations that have missing values for CLASS variables are removed from the analysis, even if the CLASS variables are not used in the model formulation.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. You can specify the following *keywords*:

GLM specifies a less-than-full-rank reference cell coding. This parameterization is used in, for example, the GLM, MIXED, and GLIMMIX procedures in SAS/STAT.

REFERENCE specifies a reference cell encoding. You can choose the reference value by specifying an option for a specific *variable* or set of *variables* in the CLASS statement, or designate the first or last ordered value by specifying a *global-option*. The default is REF=LAST.

For example, suppose that the variable temp has three levels (`'hot'`, `'warm'`, and `'cold'`), that the variable gender has two levels (`'M'` and `'F'`), and that the variables are used in a CLASS statement as follows:

```
class gender(ref='F') temp / param=ref;
```

Then `'F'` is used as the reference level for gender and `'warm'` is used as the reference level for temp.

The GLM parameterization is the default. For more information about how parameterization of classification variables affects the construction and interpretation of model effects, see the section "Parameterization of Model Effects" on page 412.

TRUNCATE<=*n***>**

specifies the truncation width of formatted values of CLASS variables when the optional *n* is specified.

If *n* is not specified, the TRUNCATE option requests that classification levels be determined by using no more than the first 16 characters of the formatted values of CLASS variables.

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence for each observation. High-performance statistical procedures that support the FREQ statement treat each observation as if it appeared f times, where f is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are transferred to output data sets that are created by high-performance statistical procedures, provided that the output data set contains one (or more) records per input observation. For example, when an OUTPUT statement is used to produce observationwise scores or prediction statistics, ID variables are added to the output data set.

By default, high-performance statistical procedures do not include all variables from the input data set in output data sets. In the following statements, a logistic regression model is fit and then scored. The input and output data are stored in the Greenplum database. The output data set contains three columns (p, account, trans_date) where p is computed during the scoring process and the account and transaction date are transferred from the input data set. (High-performance statistical procedures also transfer any distribution keys from the input to the output data.)

```
libname GPLib greenplm server=gpdca user=XXX password=YYY
              database=ZZZ;
proc hplogistic data=gplib.myData;
   class a b;
   model y = a b x1-x20;
   output out=gplib.scores pred=p;
   id account trans_date;
run;
```

SELECTION Statement

SELECTION < *options* > **;**

High-performance statistical procedures that support model selection use the SELECTION statement to control details about the model selection process. This statement is supported in different degrees by the HPGENSELECT, HPREG, and HPLOGISTIC procedures. The HPREG procedure supports the most complete set of options.

You can specify the following *options* in the SELECTION statement:

METHOD=NONE | *method*< *method-options* >

specifies the method used to select the model. You can also specify *method-options* that apply to the specified method by enclosing them in parentheses after the *method*. The default selection method (when the METHOD= option is not specified) is METHOD=STEPWISE.

The following *methods* are available and are explained in detail in the section "Methods" on page 582.

NONE specifies no model selection.

FORWARD specifies forward selection. This method starts with no effects in the model and adds effects.

BACKWARD specifies backward elimination. This method starts with all effects in the model and deletes effects.

STEPWISE specifies stepwise regression. This method is similar to the FORWARD method except that effects already in the model do not necessarily stay there.

FORWARDSWAP specifies forward-swap selection, which is an extension of the forward selection method. Before any addition step, all pairwise swaps of one effect in the model and one effect out of the current model that improve the selection criterion are made. When the selection criterion is R square, this method is the same as the MAXR method in the REG procedure in SAS/STAT software. The high-performance statistical procedure that supports this method is the HPREG procedure.

LAR specifies least angle regression. Like forward selection, this method starts by adding effects to an empty model. The parameter estimates at any step are "shrunk" when they are compared to the corresponding least squares estimates. If the model contains classification variables, then these classification variables are split. See the SPLIT option in the CLASS statement for details. The only high-performance statistical procedure that supports this method is the HPREG procedure.

LASSO adds and deletes parameters by using a version of ordinary least squares in which the sum of the absolute regression coefficients is constrained. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.

Table 22.2 lists the applicable *method-options* for each of these methods.

Table 22.2 Applicable *method-options* by *method*

method-option	FORWARD	BACKWARD	STEPWISE	FORWARDSWAP	LAR	LASSO
ADAPTIVE						x
CHOOSE =	x	x	x		x	x
COMPETITIVE			x			
CRITERION =	x	x	x	x		
FAST		x				
LSCOEFFS					x	x
MAXEFFECTS =	x		x	x	x	x
MAXSTEPS =	x	x	x	x	x	x
MINEFFECTS =		x				
SELECT =	x	x	x	x		

Table 22.2 *continued*

method-option	FORWARD	BACKWARD	STEPWISE	FORWARDSWAP	LAR	LASSO
SLENTRY =	x		x		x	x
SLSTAY =		x	x			x
STOP =	x	x	x	x	x	x

The syntax of the *method-options* that you can specify in parentheses after the SELECTION= option *method* follows. As described in Table 22.2, not all selection *method-options* are applicable to every METHOD= *method*.

ADAPTIVE <(GAMMA=*nonnegative number***)>**
requests that adaptive weights be applied to each of the coefficients when METHOD=LASSO. Ordinary least squares estimates of the model parameters are used to form the adaptive weights. You use the GAMMA= option to specify the power transformation that is applied to the parameters in forming the adaptive weights. The default value is GAMMA=1.

CHOOSE=*criterion*
chooses from the list of models (at each step of the selection process) the model that yields the best value of the specified criterion. If the optimal value of the specified criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. If you do not specify the CHOOSE= option, then the selected model is the model at the final step in the selection process. The criteria that are supported depend on the type of model that is being fit. For the supported criteria, see the chapters for the relevant high-performance statistical procedures.

COMPETITIVE
is applicable only as a *method-option* when METHOD=STEPWISE and the SELECT criterion is not SL. If you specify the COMPETITIVE option, then the SELECT= criterion is evaluated for all models in which an effect currently in the model is dropped or an effect not yet in the model is added. The effect whose removal from or addition to the model yields the maximum improvement to the SELECT= criterion is dropped or added. The HPREG procedure is the only high-performance statistical procedure that supports this option.

CRITERION=*criterion*
is an alias for the SELECT option.

FAST
implements the computational algorithm of Lawless and Singhal (1978) to compute a first-order approximation to the remaining slope estimates for each subsequent elimination of a variable from the model. When applied in backward selection, this option essentially leads to approximating the selection process as the selection process of a linear regression model in which the crossproducts matrix equals the Hessian matrix in the full model under consideration. The FAST option is available only when METHOD=BACKWARD in the HPLOGISTIC procedure. It is computationally efficient in logistic regression models because the model is not fit after removal of each effect.

LSCOEFFS

requests a hybrid version of the LAR and LASSO methods, in which the sequence of models is determined by the LAR or LASSO algorithm but the coefficients of the parameters for the model at any step are determined by using ordinary least squares.

MAXEFFECTS=*n*

specifies the maximum number of effects in any model that is considered during the selection process. This option is ignored with METHOD=BACKWARD. If at some step of the selection process the model contains the specified maximum number of effects, then no candidates for addition are considered.

MAXSTEPS=*n*

specifies the maximum number of selection steps that are performed. The default value of *n* is the number of effects in the MODEL statement when METHOD=FORWARD, METHOD=BACKWARD, or METHOD=LAR. The default is three times the number of effects when METHOD=STEPWISE or METHOD=LASSO.

MINEFFECTS=*n*

specifies the minimum number of effects in any model that is considered during backward selection. This option is ignored unless METHOD=BACKWARD is specified. The backward selection process terminates if, at some step of the selection process, the model contains the specified minimum number of effects.

SELECT=SL | *criterion*

specifies the criterion that the procedure uses to determine the order in which effects enter or leave at each step of the selection method. The criteria that are supported depend on type of model that is being fit. See the chapter for the relevant high-performance statistical procedure for the supported criteria.

The SELECT option is not valid when METHOD=LAR or METHOD=LASSO. You can use SELECT=SL to request the traditional approach, where effects enter and leave the model based on the significance level. When the value of the SELECT= option is not SL, the effect that is selected to enter or leave at any step of the selection process is the effect whose addition to or removal from the current model yields the maximum improvement in the specified criterion.

SLENTRY=*value*

SLE=*value*

specifies the significance level for entry when STOP=SL or SELECT=SL. The default is 0.05.

SLSTAY=*value*

SLS=*value*

specifies the significance level for staying in the model when STOP=SL or SELECT=SL. The default is 0.05.

STOP=SL | **NONE** | *criterion*

specifies a criterion that is used to stop the selection process. The criteria that are supported depend on the type of model that is being fit. For information about the supported criteria, see the chapter about the relevant high-performance statistical procedure.

If you do not specify the STOP= option but do specify the SELECT= option, then the criterion specified in the SELECT= option is also used as the STOP= criterion.

If you specify STOP=NONE, then the selection process stops if no suitable add or drop candidates can be found or if a size-based limit is reached. For example, if you specify STOP=NONE MAXEFFECTS=5, then the selection process stops at the first step that produces a model with five effects.

When STOP=SL, selection stops at the step where the significance level of the candidate for entry is greater than the SLENTRY= value for addition steps when METHOD=FORWARD or METHOD=STEPWISE and where the significance level of the candidate for removal is greater than the SLSTAY= value when METHOD=BACKWARD or METHOD=STEPWISE.

If you specify a criterion other than SL for the STOP= option, then the selection process stops if the selection process produces a local extremum of this criterion or if a size-based limit is reached. For example, if you specify STOP=AIC MAXSTEPS=5, then the selection process stops before step 5 if the sequence of models has a local minimum of the AIC criterion before step 5. The determination of whether a local minimum is reached is made on the basis of a stop horizon. The default stop horizon is 3, but you can change it by using the STOPHORIZON= option. If the stop horizon is *n* and the STOP= criterion at any step is better than the stop criterion at the next *n* steps, then the selection process terminates.

DETAILS=NONE | SUMMARY | ALL

DETAILS=STEPS< (CANDIDATES(ALL | *n*)) >

specifies the level of detail to be produced about the selection process. The default is DETAILS=SUMMARY.

The DETAILS=ALL and DETAILS=STEPS options produce the following output:

- tables that provide information about the model that is selected at each step of the selection process.
- entry and removal statistics for inclusion or exclusion candidates at each step. By default, only the top 10 candidates at each step are shown. If you specify STEPS(CANDIDATES(*n*)), then the best *n* candidates are shown. If you specify STEPS(CANDIDATES(ALL)), then all candidates are shown.
- a selection summary table that shows by step the effect that is added to or removed from the model in addition to the values of the SELECT, STOP, and CHOOSE criteria for the resulting model.
- a stop reason table that describes why the selection process stopped.
- a selection reason table that describes why the selected model was chosen.
- a selected effects table that lists the effects that are in the selected model.

The DETAILS=SUMMARY option produces only the selection summary, stop reason, selection reason, and selected effects tables.

HIERARCHY=NONE | SINGLE | SINGLECLASS

specifies whether and how the model hierarchy requirement is applied. You can specify that only classification effects, or both classification and continuous effects, be subject to the hierarchy requirement. The HIERARCHY= option is ignored unless you also specify one of the following options: METHOD=FORWARD, METHOD=BACKWARD, or METHOD=STEPWISE.

Model hierarchy refers to the requirement that, for any term to be in the model, all model effects that are contained in the term must be present in the model. For example, in order for the interaction A*B

to enter the model, the main effects A and B must be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model.

You can specify the following values:

NONE specifies that model hierarchy not be maintained. Any single effect can enter or leave the model at any given step of the selection process.

SINGLE specifies that only one effect enter or leave the model at one time, subject to the model hierarchy requirement. For example, suppose that the model contains the main effects A and B and the interaction A*B. In the first step of the selection process, either A or B can enter the model. In the second step, the other main effect can enter the model. The interaction effect can enter the model only when both main effects have already entered. Also, before A or B can be removed from the model, the A*B interaction must first be removed. All effects (CLASS and interval) are subject to the hierarchy requirement.

SINGLECLASS is the same as HIERARCHY=SINGLE except that only CLASS effects are subject to the hierarchy requirement.

The default value is HIERARCHY=NONE.

SCREEN < (*global-screen-options*) > < = *screen-options* >
requests that a subset of the effects specified in the MODEL statement be chosen as candidate effects for model selection. You use the *global-screen-options* and *screen-options* to specify how such a subset is chosen and to control the detail level of the associated output. The SCREEN option is fully documented in the section "SELECTION Statement" on page 5146 in Chapter 67, "The HPREG Procedure," which is the only high-performance statistical procedure that supports the SCREEN option.

SELECTION=NONE | BACKWARD | FORWARD | FORWARDSWAP | STEPWISE | LAR | LASSO
is an alias for the METHOD= option.

STOPHORIZON=*n*
specifies the number of consecutive steps at which the STOP= criterion must worsen in order for a local extremum to be detected. The default value is STOPHORIZON=3. The stop horizon value is ignored if you also specify STOP=NONE or STOP=SL. For example, suppose that STOP=AIC and the sequence of AIC values at steps 1 to 6 of a selection are 10, 7, 4, 6, 5, 2. If STOPHORIZON=2, then the AIC criterion is deemed to have a local minimum at step 3 because the AIC value at the next two steps are greater than the value 4 that occurs at step 3. However, if STOPHORIZON=3, then the value at step 3 is not deemed to be a local minimum because the AIC value at step 6 is lower than the AIC value at step 3.

VAR Statement

VAR *variable-list* **;**

Some high-performance statistical procedures (in particular procedures that do not support a MODEL statement) use a VAR statement to identify numerical variables for the analysis.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations with nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.

Model Selection

Methods

The model selection methods implemented in high-performance statistical procedures are specified in the METHOD= option in the SELECTION statement. The following methods are available, although specific procedures might support only a subset of these methods. Furthermore, the examples in this section refer to fit criteria that might not be supported by a specific procedure.

Full Model Fitted

When METHOD=NONE, the complete model that is specified in the MODEL statement is used to fit the model, and no effect selection is done.

Forward Selection

METHOD=FORWARD specifies the forward selection technique, which begins with just the intercept and then sequentially adds the effect that most improves the fit. The process terminates when no significant improvement can be obtained by adding any effect.

In the traditional implementation of forward selection, the statistic that is used to determine whether to add an effect is the significance level of a hypothesis test that reflects an effect's contribution to the model if it is included. At each step, the effect that is most significant is added. The process stops when the significance level for adding any effect is greater than some specified entry significance level.

An alternative approach to address the critical problem of when to stop the selection process is to assess the quality of the models that are produced by the forward selection method and choose the model from this sequence that "best" balances goodness of fit against model complexity. You can use several criteria for this purpose. These criteria fall into two groups—information criteria and criteria that are based on out-of-sample prediction performance.

You use the CHOOSE= option to specify the criterion for selecting one model from the sequence of models produced. If you do not specify a CHOOSE= criterion, then the model at the final step is the selected model.

For example, if you specify the following statement, then forward selection terminates at the step where no effect can be added at the 0.2 significance level:

```
selection method=forward(select=SL choose=AIC SLE=0.2);
```

However, the selected model is the first one that has the minimum value of Akaike's information criterion. In some cases, this minimum value might occur at a step much earlier than the final step. In other cases, the AIC might start increasing only if more steps are performed—that is, a larger value is used for the significance level for entry. If you want to minimize AIC, then too many steps are performed in the former case and too few in the latter case. To address this issue, high-performance statistical procedures enable you to specify a stopping criterion by using the STOP= option. When you specify a stopping criterion, forward selection continues until a local extremum of the stopping criterion in the sequence of models generated is reached. To be deemed a local extremum, a criterion value at a given step must be better than its value at the next n steps, where n is known as the "stop horizon." By default, the stop horizon is three steps, but you can change this by specifying the STOPHORIZON= option.

For example, if you specify the following statement, then forward selection terminates at the step where the effect to be added at the next step would produce a model that has an AIC statistic larger than the AIC statistic of the current model:

```
selection method=forward(select=SL stop=AIC) stophorizon=1;
```

In most cases, provided that the entry significance level is large enough that the local extremum of the named criterion occurs before the final step, specifying either of the following statements selects the same model, but more steps are done in the first case:

```
selection method=forward(select=SL choose=CRITERION);
selection method=forward(select=SL stop=CRITERION);
```

In some cases, there might be a better local extremum that cannot be reached if you specify the STOP= option but can be found if you use the CHOOSE= option. Also, you can use the CHOOSE= option in preference to the STOP= option if you want to examine how the named criterion behaves as you move beyond the step where the first local minimum of this criterion occurs.

You can specify both the CHOOSE= and STOP= options. You can also use these options together with options that specify size-based limits on the selected model. You might want to consider models that are generated by forward selection and have at most some fixed number of effects, but select from within this set based on a criterion that you specify. For example, specifying the following statements requests that forward selection continue until there are 20 effects in the final model and chooses among the sequence of models the one that has the largest value of the adjusted R-square statistic:

```
selection method=forward(stop=none maxeffects=20 choose=ADJRSQ);
```

You can also combine these options to select a model where one of two conditions is met. For example, the following statement chooses whatever occurs first between a local minimum of the sum of squares on validation data and a local minimum of the corrected Akaike's information criterion (AICC):

```
selection method=forward(stop=AICC choose=VALIDATE);
```

It is important to keep in mind that forward selection bases the decision about what effect to add at any step by considering models that differ by one effect from the current model. This search paradigm cannot

guarantee reaching a "best" subset model. Furthermore, the add decision is greedy in the sense that the effect that is deemed most significant is the effect that is added. However, if your goal is to find a model that is best in terms of some selection criterion other than the significance level of the entering effect, then even this one step choice might not be optimal. For example, the effect that you would add to get a model that has the smallest value of the Mallows' $C(p)$ statistic at the next step is not necessarily the same effect that is most significant based on a hypothesis test. High-performance statistical procedures enable you to specify the criterion to optimize at each step by using the SELECT= option. For example, the following statement requests that at each step the effect that is added be the one that produces a model that has the smallest value of the Mallows' $C(p)$ statistic:

```
selection method=forward(select=CP);
```

In the case where all effects are variables (that is, effects with one degree of freedom and no hierarchy), using ADJRSQ, AIC, AICC, BIC, CP, RSQUARE, or SBC as the selection criterion for forward selection produces the same sequence of additions. However, if the degrees of freedom contributed by different effects are not constant or if an out-of-sample prediction-based criterion is used, then different sequences of additions might be obtained.

You can use the SELECT= option together with the CHOOSE= and STOP= options. If you specify only the SELECT= criterion, then this criterion is also used as the stopping criterion. In the previous example where only the selection criterion is specified, not only do effects enter based on the Mallows' $C(p)$ statistic, but the selection terminates when the $C(p)$ statistic has a local minimum.

You can find discussion and references to studies about criteria for variable selection in Burnham and Anderson (2002), along with some cautions and recommendations.

Examples of Forward Selection Specifications

The following statement adds effects that at each step produce the lowest value of the SBC statistic and stops at the step where adding any effect would increase the SBC statistic:

```
selection method=forward stophorizon=1;
```

The following statement adds effects based on significance level and stops when all candidate effects for entry at a step have a significance level greater than the default entry significance level of 0.05:

```
selection=forward(select=SL);
```

The following statement adds effects based on significance level and stops at a step where adding any effect increases the error sum of squares computed on the validation data:

```
selection=forward(select=SL stop=validation) stophorizon=1;
```

The following statement adds effects that at each step produce the lowest value of the AIC statistic and stops at the first step whose AIC value is smaller than the AIC value at the next three steps:

```
selection=forward(select=AIC);
```

The following statement adds effects that at each step produce the largest value of the adjusted R-square statistic and stops at the step where the significance level that corresponds to the addition of this effect is greater than 0.2:

```
selection=forward(select=ADJRSQ stop=SL SLE=0.2);
```

Backward Elimination

METHOD=BACKWARD specifies the backward elimination technique. This technique starts from the full model, which includes all independent effects. Then effects are deleted one by one until a stopping condition is satisfied. At each step, the effect that shows the smallest contribution to the model is deleted.

In the traditional implementation of backward selection, the statistic that is used to determine whether to drop an effect is significance level. At any step, the least significant predictor is dropped and the process continues until all effects that remain in the model are significant at a specified stay significance level (SLS).

Just as with forward selection, you can use the SELECT= option to change the criterion that is used to assess effect contributions. You can also specify a stopping criterion in the STOP= option and use a CHOOSE= option to provide a criterion for selecting among the sequence of models produced. For more information, see the discussion in the section "Forward Selection" on page 582.

Examples of Backward Selection Specifications

The following statement removes effects that at each step produce the largest value of the Schwarz Bayesian information criterion (SBC) statistic and stops at the step where removing any effect increases the SBC statistic:

```
selection method=backward stophorizon=1;
```

The following statement bases removal of effects on significance level and stops when all candidate effects for removal at a step are significant at the default stay significance level of 0.05:

```
selection method=backward(select=SL);
```

The following statement bases removal of effects on significance level and stops when all effects in the model are significant at the 0.1 level. Finally, from the sequence of models generated, the chosen model is the one that produces the smallest average square error when scored on the validation data:

```
selection method=backward(select=SL choose=validate SLS=0.1);
```

The following statement applies in logistic regression models the fast backward technique of Lawless and Singhal (1978), a first-order approximation that has greater numerical efficiency than full backward selection:

```
selection method=backward(fast);
```

The fast technique fits an initial full logistic model and a reduced model after the candidate effects have been dropped. On the other hand, full backward selection fits a logistic regression model each time an effect is removed from the model.

Stepwise Selection

METHOD=STEPWISE specifies the stepwise method, which is a modification of the forward selection technique that differs in that effects already in the model do not necessarily stay there.

In the traditional implementation of stepwise selection method, the same entry and removal significance levels for the forward selection and backward elimination methods are used to assess contributions of effects as they are added to or removed from a model. If, at a step of the stepwise method, any effect in the model is

not significant at the SLSTAY= level, then the least significant of these effects is removed from the model and the algorithm proceeds to the next step. This ensures that no effect can be added to a model while some effect currently in the model is not deemed significant. Only after all necessary deletions have been accomplished can another effect be added to the model. In this case the effect whose addition is the most significant is added to the model and the algorithm proceeds to the next step. The stepwise process ends when none of the effects outside the model is significant at the SLENTRY= level and every effect in the model is significant at the SLSTAY= level. In some cases, neither of these two conditions for stopping is met and the sequence of models cycles. In this case, the stepwise method terminates at the end of the cycle.

Just as you can in forward selection and backward elimination, you can use the SELECT= option to change the criterion that is used to assess effect contributions. You can also use the STOP= option to specify a stopping criterion and use a CHOOSE= option to provide a criterion for selecting among the sequence of models produced. For more information, see the section "Forward Selection" on page 582.

For selection criteria other than significance level, the HPREG procedure optionally supports a further modification in the stepwise method. In the standard stepwise method, no effect can enter the model if removing any effect currently in the model would yield an improved value of the selection criterion. In the modification, you can use the COMPETITIVE option to specify that addition and deletion of effects should be treated competitively. The selection criterion is evaluated for all models that are produced by deleting an effect from the current model or by adding an effect to this model. The action that most improves the selection criterion is the action taken.

Examples of Stepwise Selection Specifications

The following statement requests stepwise selection based on the SBC criterion:

```
selection method=stepwise;
```

First, if removing any effect yields a model that has a lower SBC statistic than the current model, then the effect that produces the smallest SBC statistic is removed. If removing any effect increases the SBC statistic, then provided that adding some effect lowers the SBC statistic, the effect that produces the model that has the lowest SBC is added.

The following statement requests the traditional stepwise method:

```
selection=stepwise(select=SL)
```

First, if the removal of any effect in the model is not significant at the default stay level of 0.05, then the least significant effect is removed and the algorithm proceeds to the next step. Otherwise, the effect whose addition is the most significant is added, provided that it is significant at the default entry level of 0.05.

The following statement requests the traditional stepwise method, where effects enter and leave based on significance levels, but with the following extra check: if any effect to be added or removed yields a model whose SBC statistic is greater than the SBC statistic of the current model, then the stepwise method terminates at the current model.

```
selection method=stepwise(select=SL stop=SBC) stophorizon=1;
```

In this case, the entry and stay significance levels still play a role because they determine whether an effect is deleted from or added to the model. This extra check might result in the selection terminating before a local minimum of the SBC criterion is found.

The following statement selects effects to enter or drop as in the previous example except that the significance level for entry is now 0.1 and the significance level to stay is 0.08. From the sequence of models produced,

the selected model is chosen to yield the minimum AIC statistic:

```
selection method=stepwise(select=SL SLE=0.1 SLS=0.08 choose=AIC);
```

The following statement requests stepwise selection that is based on the AICC criterion and treats additions and deletions competitively:

```
selection method=stepwise(select=AICC competitive);
```

Each step evaluates the AICC statistics that correspond to the removal of any effect in the current model or the addition of any effect to the current model and chooses the addition or removal that produced the minimum value, provided that this minimum is lower than the AICC statistic of the current model.

The following statement requests stepwise selection that is based on the SBC criterion, treats additions and deletions competitively, and stops based on the average square error over the validation data:

```
selection=stepwise(select=SBC competitive stop=VALIDATE);
```

At any step, SBC statistics that correspond to the removal of any effect from the current model or the addition of any effect to the current model are evaluated. The addition or removal that produces the minimum SBC value is made. The average square error on the validation data for the model with this addition or removal is evaluated. The selection stops when the average square error so produced increases for three consecutive steps.

Forward-Swap Selection

METHOD=FORWARDSWAP specifies the forward-swap selection method, which is an extension of the forward selection method. The forward-swap selection method incorporates steps that improve a model by replacing an effect in the model with an effect that is not in the model. When the model selection criterion is R square, this method is the same as the maximum R-square improvement (MAXR) method that is implemented in the REG procedure in SAS/STAT software. You cannot use the effect significance level as the selection criterion for the forward-swap method.

The forward-swap selection method begins by finding the one-effect model that produces the best value of the selection criterion. Then another effect (the one that yields the greatest improvement in the selection criterion) is added. After the two-effect model is obtained, each of the effects in the model is compared to each effect that is not in the model. For each comparison, the forward-swap method determines whether removing one effect and replacing it with the other effect improves the selection criterion. After comparing all possible swaps, the forward-swap method makes the swap that produces the greatest improvement in the selection criterion. Comparisons begin again, and the process continues until the forward-swap method finds that no other swap could improve the selection criterion. Thus, the two-variable model that is produced is considered the “best” two-variable model that the technique can find. Another variable is then added to the model, and the comparing-and-swapping process is repeated to find the “best” three-variable model, and so on.

The difference between the stepwise selection method and the forward-swap selection method is that all swaps are evaluated before any addition is made in the forward-swap method. In the stepwise selection method, the “worst” effect might be removed without considering what adding the “best” remaining effects might accomplish. Because the forward-swap method needs to examine all possible pairwise effect swaps at each step of the selection process, the forward-swap method is much more computationally expensive than the stepwise selection method; it might not be appropriate for models that contain a large number of effects.

Least Angle Regression

METHOD=LAR specifies least angle regression (LAR), which is supported in the HPREG procedure. LAR was introduced by Efron et al. (2004). Not only does this algorithm provide a selection method in its own right, but with one additional modification, it can be used to efficiently produce LASSO solutions. Just like the forward selection method, the LAR algorithm produces a sequence of regression models in which one parameter is added at each step, terminating at the full least squares solution when all parameters have entered the model.

The algorithm starts by centering the covariates and response and scaling the covariates so that they all have the same corrected sum of squares. Initially all coefficients are zero, as is the predicted response. The predictor that is most correlated with the current residual is determined, and a step is taken in the direction of this predictor. The length of this step determines the coefficient of this predictor and is chosen so that some other predictor and the current predicted response have the same correlation with the current residual. At this point, the predicted response moves in the direction that is equiangular between these two predictors. Moving in this direction ensures that these two predictors continue to have a common correlation with the current residual. The predicted response moves in this direction until a third predictor has the same correlation with the current residual as the two predictors already in the model. A new direction is determined that is equiangular among these three predictors, and the predicted response moves in this direction until a fourth predictor, which has the same correlation with the current residual, joins the set. This process continues until all predictors are in the model.

As in other selection methods, the issue of when to stop the selection process is crucial. You can use the CHOOSE= option to specify a criterion for choosing among the models at each step. You can also use the STOP= option to specify a stopping criterion. These formulas use the approximation that at step k of the LAR algorithm, the model has k degrees of freedom. See Efron et al. (2004) for a detailed discussion of this so-called simple approximation.

A modification of LAR selection that is suggested in Efron et al. (2004) uses the LAR algorithm to select the set of covariates in the model at any step, but it uses ordinary least squares regression with just these covariates to obtain the regression coefficients. You can request this hybrid method by specifying the LSCOEFFS suboption of METHOD=LAR.

Lasso Selection

Method=LASSO specifies the least absolute shrinkage and selection operator (LASSO) method, which is supported in the HPREG procedure. LASSO arises from a constrained form of ordinary least squares regression where the sum of the absolute values of the regression coefficients is constrained to be smaller than a specified parameter. More precisely let $X = (x_1, x_2, \ldots, x_m)$ denote the matrix of covariates and let y denote the response, where the x_is have been centered and scaled to have unit standard deviation and mean zero and y has mean zero. Then for a given parameter t, the LASSO regression coefficients $\beta = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the following constrained optimization problem:

$$\text{minimize}||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t$$

Provided that the LASSO parameter t is small enough, some of the regression coefficients are exactly 0. Hence, you can view the LASSO as selecting a subset of the regression coefficients for each LASSO

parameter. By increasing the LASSO parameter in discrete steps, you obtain a sequence of regression coefficients in which the nonzero coefficients at each step correspond to selected parameters.

Early implementations (Tibshirani 1996) of LASSO selection used quadratic programming techniques to solve the constrained least squares problem for each LASSO parameter of interest. Later Osborne, Presnell, and Turlach (2000) developed a "homotopy method" that generates the LASSO solutions for all values of t. Efron et al. (2004) derived a variant of their algorithm for least angle regression that can be used to obtain a sequence of LASSO solutions from which all other LASSO solutions can be obtained by linear interpolation. This algorithm for METHOD=LASSO is used in PROC HPREG. It can be viewed as a stepwise procedure with a single addition to or deletion from the set of nonzero regression coefficients at any step.

As in the other selection methods that are supported by high-performance statistical procedures, you can use the CHOOSE= option to specify a criterion to choose among the models at each step of the LASSO algorithm. You can also use the STOP= option to specify a stopping criterion. For more information, see the discussion in the section "Forward Selection" on page 582. The model degrees of freedom that PROC GLMSELECT uses at any step of the LASSO are simply the number of nonzero regression coefficients in the model at that step. Efron et al. (2004) cite empirical evidence for doing this but do not give any mathematical justification for this choice.

A modification of LASSO selection suggested in Efron et al. (2004) uses the LASSO algorithm to select the set of covariates in the model at any step, but it uses ordinary least squares regression and just these covariates to obtain the regression coefficients. You can request this hybrid method by specifying the LSCOEFFS suboption of SELECTION=LASSO.

Adaptive Lasso Selection

Adaptive lasso selection is a modification of lasso selection; in adaptive lasso selection, weights are applied to each of the parameters in forming the lasso constraint (Zou 2006). More precisely, suppose that the response y has mean 0 and the regressors x are scaled to have mean 0 and common standard deviation. Furthermore, suppose that you can find a suitable estimator $\hat{\beta}$ of the parameters in the true model and you define a weight vector by $w = 1/|\hat{\beta}|^{\gamma}$, where $\gamma \geq 0$. Then the adaptive lasso regression coefficients $\beta = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the following constrained optimization problem:

$$\text{minimize}||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{m} |w_j \beta_j| \leq t$$

PROC HPREG uses the solution to the unconstrained least squares problem as the estimator $\hat{\beta}$. This is appropriate unless collinearity is a concern. If the regressors are collinear or nearly collinear, then Zou (2006) suggests using a ridge regression estimate to form the adaptive weights.

References

Burnham, K. P., and Anderson, D. R. (2002). *Model Selection and Multimodel Inference*. 2nd ed. New York: Springer-Verlag.

Efron, B., Hastie, T. J., Johnstone, I. M., and Tibshirani, R. (2004). "Least Angle Regression." *Annals of Statistics* 32:407–499. With discussion.

Lawless, J. F., and Singhal, K. (1978). "Efficient Screening of Nonnormal Regression Models." *Biometrics* 34:318–327.

Osborne, M. R., Presnell, B., and Turlach, B. A. (2000). "A New Approach to Variable Selection in Least Squares Problems." *IMA Journal of Numerical Analysis* 20:389–404.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B* 58:267–288.

Zou, H. (2006). "The Adaptive Lasso and Its Oracle Properties." *Journal of the American Statistical Association* 101:1418–1429.

Chapter 23
Using the Output Delivery System

Contents

Overview: Using the Output Delivery System **592**
- Output Defaults 592
  - HTML Output in the SAS Windowing Environment 592
  - LISTING Output in the SAS Windowing Environment 594
  - Assumptions about ODS Defaults in This Chapter 594
  - The HTMLBLUE Style 595
- Default Open Destination 595
  - Setting the Default Destination in the Results Tab 595
  - Setting the Default Destination in the SAS Registry 595
  - Setting the Default Destination in SAS System Options 596
  - Setting the Destination in ODS Statements 596
- Output Objects and ODS Destinations 598
- The ODS Statement 599
- Paths and Selection 600
- RUN-Group Processing 604
- The SAS Results Window 604
- The ODS PATH Statement 606
- The Master Template Store 606
- Controlling Output Appearance with Templates 607
- ODS and the NOPRINT Option 613

Examples: Using the Output Delivery System **614**
- Example 23.1: Creating HTML Output with ODS 614
- Example 23.2: Selecting ODS Tables for Display 616
- Example 23.3: Excluding ODS Tables from Display 619
- Example 23.4: Creating an Output Data Set from an ODS Table 621
- Example 23.5: Creating an Output Data Set: Subsetting the Data 624
- Example 23.6: RUN-Group Processing 626
- Example 23.7: ODS Output Data Sets and Using PROC TEMPLATE to Customize Output 630
- Example 23.8: HTML Output with Hyperlinks between Tables 642
- Example 23.9: HTML Output with Graphics and Hyperlinks 647
- Example 23.10: Correlation and Covariance Matrices 651

References **664**

Overview: Using the Output Delivery System

Most SAS procedures use the Output Delivery System (ODS) to manage their output. ODS enables you to do the following:

- display your output in hypertext markup language (HTML), rich text format (RTF), portable document format (PDF), PostScript, SAS listing, or other formats
- create SAS data sets directly from tables or graphs
- select or exclude individual pieces of output
- customize the layout, format, headers, and style of your output
- produce graphs with ODS Graphics (see Chapter 24, "Statistical Graphics Using ODS")

This chapter discusses some typical applications of ODS with SAS software. For complete documentation about the Output Delivery System, see the *SAS Output Delivery System: User's Guide*. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS," and Chapter 25, "ODS Graphics Template Modification."

Output Defaults

HTML output with ODS Graphics enabled is the default in the SAS windowing environment for Microsoft Windows and UNIX. LISTING output with ODS Graphics disabled is the default when you run SAS in batch mode or on the mainframe. Your actual defaults might be different due to your registry, system option, or configuration file settings. The following sections explain these defaults and how to change them.

HTML Output in the SAS Windowing Environment

The default destination in the SAS windowing environment is HTML and ODS Graphics is enabled by default.[1] These defaults have several advantages. Graphs are integrated with tables, and all output is displayed in the same HTML file. The HTML destination uses the HTMLBLUE style, which is an all-color style, that is designed to integrate tables and modern statistical graphics.

You can view and modify the default settings by selecting **Tools ► Options ► Preferences** from the menu at the top of the main SAS window. Then click the **Results** tab. You can remember this sequence using the mnemonic TOPR (pronounced "topper"). See Figure 23.1.

[1] HTML output with ODS Graphics enabled is the default in the SAS windowing environment for Microsoft Windows and UNIX, but not on the mainframe.

Figure 23.1 SAS Results Tab with the Default Settings

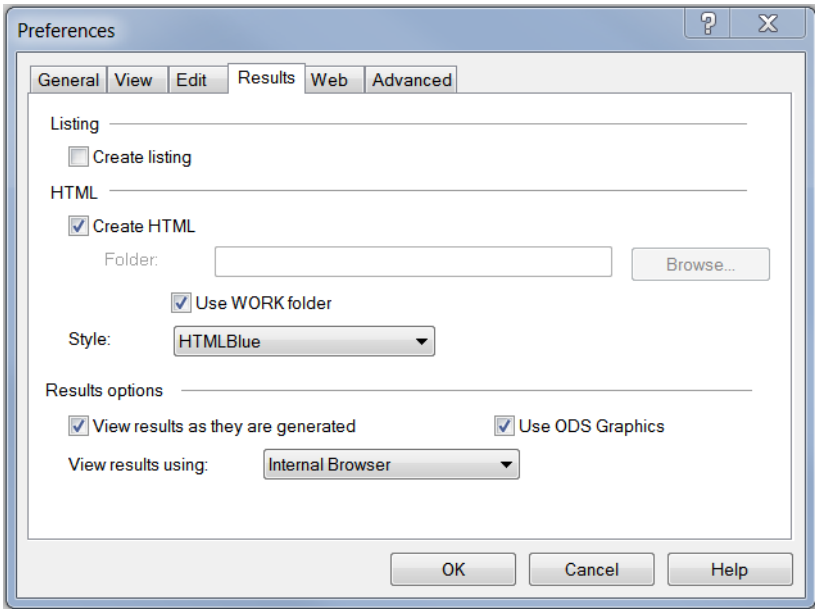

The default settings are as follows:

- HTML output is created when **Create HTML** is selected, and all output is viewed in the Results Viewer window.
- ODS Graphics is enabled when **Use ODS Graphics** is selected.
- The default style, HTMLBLUE, is selected from the **Style** list.
- Results are viewed in an internal SAS browser when **Internal browser** is selected.
- Graph image files are saved in the Work folder (not in your current folder) when **Use WORK folder** is selected.
- LISTING output is not created when the **Create listing** box is cleared.

In many cases, graphs are an integral part of a data analysis. However, when you run large computational programs (such as when you use procedures with many BY groups), you might not want to create graphs. In those cases, you should disable ODS Graphics, which will improve the performance of your program. You can disable and re-enable ODS Graphics in your SAS programs with the ODS GRAPHICS OFF and ODS GRAPHICS ON statements. You can also change the ODS Graphics default in the **Results** tab.

In the SAS windowing environment, the current folder is displayed in the status line at the bottom of the main SAS window. When **Use WORK folder** is cleared, graph image files are saved in the current folder and are available after your SAS session ends. They can accumulate with time and take up a great deal of space. When **Use WORK folder** is selected, graph image files are stored in the Work folder and are not available after your SAS session ends.

LISTING Output in the SAS Windowing Environment

In the LISTING destination, tables are displayed in monospace, and graphs are not integrated with tables. You can create LISTING output by selecting **Tools ► Options ► Preferences** from the menu at the top of the main SAS window. Then click the **Results** tab. Select **Create listing**, and clear **Create HTML**. See Figure 23.2. Tabular results are viewed in the Output window. Graphical results are viewed by selecting graphs in the Results window.

You can enable or disable ODS Graphics by default by using the **Use ODS Graphics** check box, and you can use the ODS GRAPHICS ON and ODS GRAPHICS OFF statements to enable and disable ODS Graphics in your SAS programs.

Figure 23.2 SAS Results Tab for LISTING Output

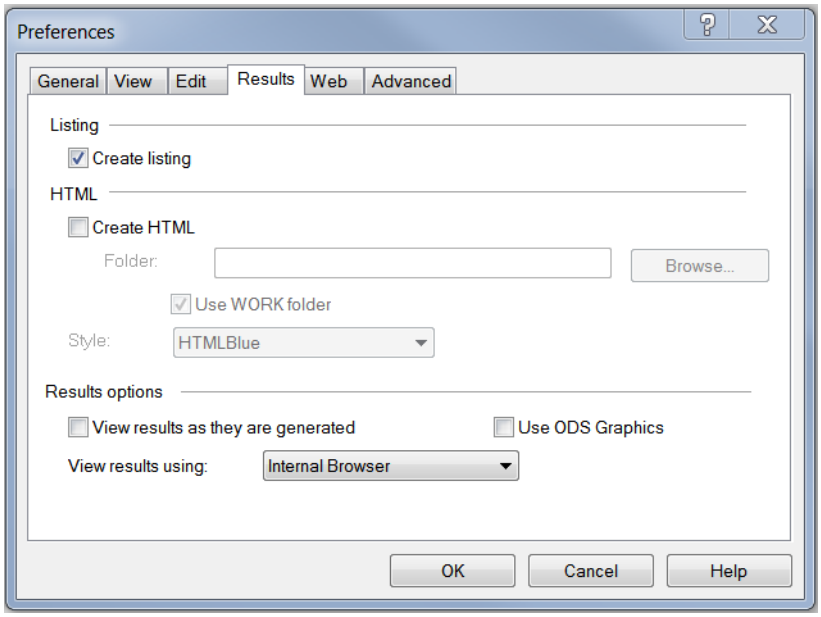

Assumptions about ODS Defaults in This Chapter

Default ODS settings (such as open destinations, styles, and whether or not ODS Graphics is enabled) vary depending on your operating system, registry settings, configuration file settings, system options, and whether you are using the SAS windowing environment or batch mode. By default, SAS/STAT documentation has two forms: PDF and HTML. By default, output is displayed by using the PDF destination and the PEARLJ style (for PDF documentation) or by using the HTML destination and the HTMLBLUE style (for HTML documentation). In most examples, ODS destination statements are not displayed, and you can run the example code to create output for any destination. In contrast, this chapter shows you how to work with destinations and styles, so it sometimes opens and closes destinations. In this chapter, when the open destinations are closed, the PDF and HTML destinations are opened at the end of the step so that destinations are available for subsequent output.

The HTMLBLUE Style

In the SAS windowing environment, the default ODS style for HTML output is the HTMLBLUE style. You can see examples of the HTMLBLUE style in this chapter in Output 23.1.1, Output 23.8.2, and Output 23.8.3. The HTMLBLUE style inherits most of its attributes from the STATISTICAL style, but it has a brighter appearance and color coordination between the tables and graphs. In the HTMLBLUE style, the dominant color is blue; in the DEFAULT style, the dominant color is gray. For a comparison of the HTMLBLUE style and other styles, see Chapter 24, "Statistical Graphics Using ODS."

Default Open Destination

By default, either the LISTING or the HTML destination is open. You can change the default destination three ways, which are described in the next three subsections. The fourth subsection of this section explains how to use ODS statements to open and close destinations.

Setting the Default Destination in the Results Tab

You can change the default destination in the SAS windowing environment by selecting **Tools ► Options ► Preferences** from the menu at the top of the main SAS window. Then select the **Results** tab. See sections "HTML Output in the SAS Windowing Environment" on page 592 and "LISTING Output in the SAS Windowing Environment" on page 594. Changing defaults in the SAS windowing environment affects only the SAS windowing environment; it does not affect batch jobs.

Setting the Default Destination in the SAS Registry

You can change the default destination by editing the SAS registry (see Figure 23.3) or by changing system options in your SAS configuration file.

Figure 23.3 SAS Registry Window

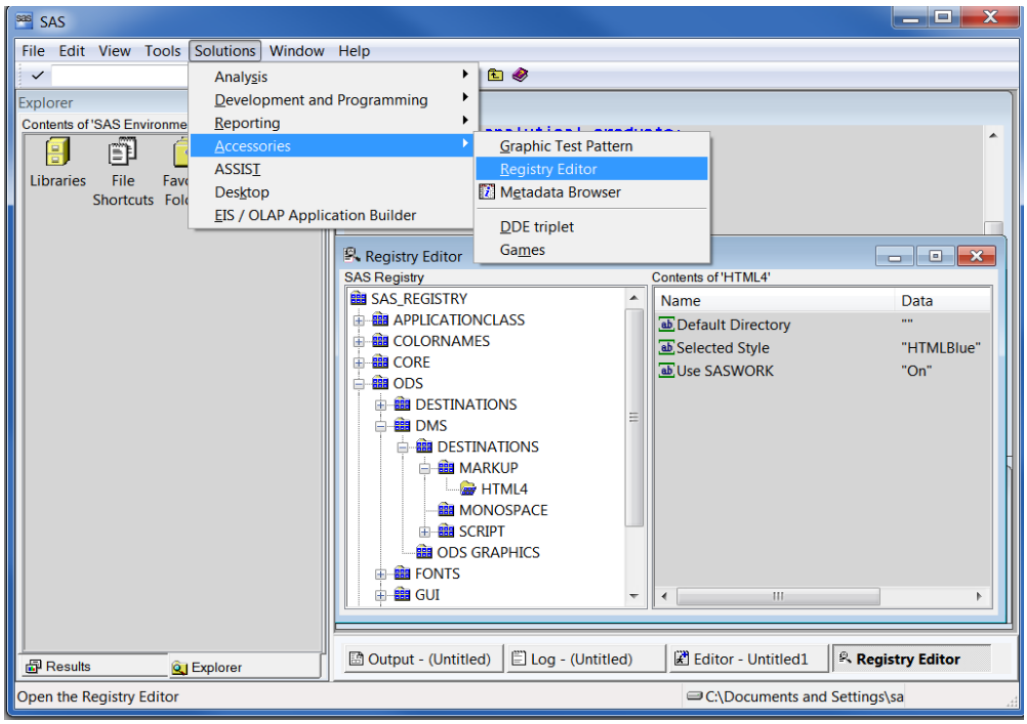

Registry customization is generally performed by more advanced users who have experience and knowledge about the SAS System and their operating environment. Incorrect registry entries can corrupt your SAS registry. For more information about SAS configuration files and the SAS registry, see the *SAS Companion* for your operating system.

Setting the Default Destination in SAS System Options

The ODSDEST system option controls the default destination. This option is specified only at SAS start-up time. Other relevant system options that correspond to entries in the **Results** tab in Figure 23.1 include ODSGRAPHICS (which specifies whether ODS Graphics is enabled by default) and ODSSTYLE (which specifies the default style for the HTML destination in the SAS windowing environment). See the *SAS System Options: Reference* for more information.

Setting the Destination in ODS Statements

You can use ODS destination statements to explicitly set destinations. These statements are described in this chapter and in detail in the *SAS Output Delivery System: User's Guide*. When you open a new destination, you should close all other open destinations unless you really need multiple destinations to be open. When multiple destinations are open, each piece of output is created multiple times, once per destination. Closing unneeded destinations increases efficiency.

You can create HTML output in any environment by using the ODS HTML statement as in the following example:

```
ods _all_ close;
ods html file='MyFile.html';

proc reg data=sashelp.class;
   model height=weight;
quit;

ods html close;
```

The first statement closes all open destinations. The second statement opens the HTML destination and specifies the HTML output file *MyFile.html*. The last statement closes the HTML destination.

You can create LISTING output in any environment by using the ODS LISTING statement as in the following example:

```
ods _all_ close;
ods listing;

proc reg data=sashelp.class;
   model height=weight;
quit;
```

The first statement closes all open destinations. The second statement opens the LISTING destination which sends output to the SAS listing. In this example, the LISTING destination is not closed so that subsequent steps can append more information to the listing.

If the LISTING destination is open, then you can simultaneously create LISTING and HTML output as follows:

```
* The ODS LISTING destination is not closed,
  which is not recommended for efficiency reasons;

ods html file='Reg.htm';

proc reg data=sashelp.class;
   model height=weight;
quit;

ods html close;
```

Sometimes you see ODS Graphics notes or warnings multiple times when multiple destinations are open. The messages appear once for each affected graph for each destination.

Output Objects and ODS Destinations

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgs1.sas.)

All SAS procedures produce *output objects* that ODS delivers to various *ODS destinations*, according to the default specifications for the procedure or according to your own specifications. Typically, you see the output objects displayed as tables, data sets, or graphs. Underlying all output (for example, a table of parameter estimates) are two component parts:

- the data component, which consists of the results computed by a SAS procedure
- the template, which contains the instructions for formatting and displaying the results

Each output object has an associated template, provided by the SAS System, that defines its presentation format. You can use the TEMPLATE procedure to view or alter these templates or to create new templates by changing the headers, formats, column order, and so on. For more information, see the chapter titled "The Template Procedure" in the *SAS Output Delivery System: User's Guide*.

You define the form that the output should take by specifying an ODS destination. Some supported destinations are as follows:

- LISTING, the standard SAS monospace listing
- HTML, for viewing in a browser
- RTF, for inclusion in Microsoft Word
- PDF, PostScript, and PCL, for high-fidelity printers
- OUTPUT, for saving results to SAS data sets
- DOCUMENT, for saving, modifying, and replaying your output

You can open multiple ODS destinations at the same time so that a single procedure step can produce output for multiple destinations. If you do not supply any ODS statements, ODS delivers all output to the default destination (which is usually LISTING or HTML). See the section "Output Defaults" on page 592 for more information about default destinations. You can specify an output style for each ODS destination. The style controls the foreground, background, colors, lines, fonts, and so on.

The following statements provide an example of temporarily closing all open destinations for PROC REG and then opening the PDF and HTML destinations for PROC PRINT. PROC REG with the ODS OUTPUT statement makes an output data set, Parms, from the parameter estimates table. Closing unneeded open destinations is not required, but it is done in many examples in this chapter for efficiency. Closing the superfluous destinations suppresses the generation of output that is not needed or used. This is particularly beneficial with graphics. This example uses the Sashelp.Class data set, one of the sample data sets in the Sashelp library that are automatically available for your use. The following statements produce Figure 23.4:

```
title 'Getting Started with ODS';

ods _all_ close;

proc reg data=sashelp.class;
   model height=weight;
```

```
   ods output ParameterEstimates=parms;
quit;

ods pdf;
ods html;

proc print noobs data=parms;
run;
```

The ODS OUTPUT statement contains an object name, an equal sign, and the name of the output SAS data set to create. You can use the ODS TRACE statement to find the object names. The ODS TRACE statement is described in the section "Paths and Selection" on page 600. Also see Example 23.4 for more information.

Figure 23.4 PROC REG Parameter Estimates Table

Getting Started with ODS

Model	Dependent	Variable	DF	Estimate	StdErr	tValue	Probt
MODEL1	Height	Intercept	1	42.57014	2.67989	15.89	<.0001
MODEL1	Height	Weight	1	0.19761	0.02616	7.55	<.0001

You could accomplish the same thing using ODS SELECT statements as follows:

```
ods select none;

proc reg data=sashelp.class;
   model height=weight;
   ods output ParameterEstimates=parms;
quit;

ods select all;

proc print noobs data=parms;
run;
```

You can specify ODS EXCLUDE ALL instead of ODS SELECT NONE and ODS EXCLUDE NONE instead of ODS SELECT ALL. These statements remain in effect until a new ODS SELECT or ODS EXCLUDE statement changes the selection list.

The ODS Statement

You use the ODS statement to provide instructions to ODS. You can use the ODS statement to specify options for different destinations, specify the output style, and select and exclude output. Here are some examples:

```
/* open the HTML destination with the HTMLBlue style */
ods html style=HTMLBlue;

/* select only the parameter estimates table */
ods select ParameterEstimates;

/* output the parameter estimates table to a SAS data set*/
ods output ParameterEstimates=Parms;
```

```
/* exclude the number of observations, ANOVA, and fit statistics tables */
ods exclude NObs ANOVA FitStatistics;
```

Paths and Selection

Each output object (tables, graphs, notes, and so on) produced by a SAS procedure has a name and a label. Each name is part of a name path. For example, PROC GLM has a table called ErrorSSCP, and the name path (fully qualified name) is **GLM.Repeated.MANOVA.Model.Error.ErrorSSCP**. Each level in the name path corresponds to a part of the PROC GLM hierarchy of output. Each piece of output also has a label and a label path. For example, the PROC GLM ErrorSSCP table is labeled **'SSCP Matrix'**, and the label path is **'The GLM Procedure'.'Repeated Measures Analysis'.'MANOVA'.'Model'.'Error'.'SSCP Matrix'**.

You need to know the name or label to select, exclude, or modify a table or graph. You can obtain this information in several ways:

- You can obtain names from the individual procedure documentation chapter or from the individual procedure section of the SAS online Help system. See the sections "ODS Table Names" and "ODS Graphics" from within the "Details" section of the procedure documentation chapter.
- You can use the SAS Results window to view the name of each piece of output that is created in your SAS session (see the section "The SAS Results Window" on page 604 for more information).
- You can use the ODS TRACE statement to find the name and label of each piece of output that is created in your SAS session. The ODS TRACE statement writes identifying information to the SAS log or listing for each generated output object.

If you are working interactively with reasonably small data sets, then the ODS TRACE statement is usually the most convenient way to find the names. Specify the ODS TRACE ON statement prior to the procedure statements that create the output for which you want information. For example, the following statements write the trace record for the output created in the REG procedure step:

```
ods trace on;
ods graphics on;
proc reg data=sashelp.class;
   model weight=height;
   model age=height;
quit;
ods trace off;
```

By default, the trace output is written to the SAS log. Some of the trace output from the previous step is as follows:

```
Output Added:
-------------
Name:       NObs
Label:      Number of Observations
Template:   Stat.Reg.NObs
```

```
 Path:       Reg.MODEL1.Fit.Weight.NObs
-------------


 Output Added:
-------------
 Name:       ANOVA
 Label:      Analysis of Variance
 Template:   Stat.REG.ANOVA
 Path:       Reg.MODEL1.Fit.Weight.ANOVA
-------------

 Output Added:
-------------
 Name:       FitStatistics
 Label:      Fit Statistics
 Template:   Stat.REG.FitStatistics
 Path:       Reg.MODEL1.Fit.Weight.FitStatistics
-------------

 Output Added:
-------------
 Name:       ParameterEstimates
 Label:      Parameter Estimates
 Template:   Stat.REG.ParameterEstimates
 Path:       Reg.MODEL1.Fit.Weight.ParameterEstimates
-------------

 Output Added:
-------------
 Name:       DiagnosticsPanel
 Label:      Fit Diagnostics
 Template:   Stat.REG.Graphics.DiagnosticsPanel
 Path:       Reg.MODEL1.ObswiseStats.Weight.DiagnosticPlots.DiagnosticsPanel
-------------


 Output Added:
-------------
 Name:       ResidualPlot
 Label:      Height
 Template:   Stat.REG.Graphics.ResidualPlot
 Path:       Reg.MODEL1.ObswiseStats.Weight.ResidualPlots.ResidualPlot
-------------

 Output Added:
-------------
 Name:       FitPlot
 Label:      Fit Plot
 Template:   Stat.REG.Graphics.Fit
 Path:       Reg.MODEL1.ObswiseStats.Weight.FitPlot
-------------

 Output Added:
-------------
```

```
Name:       NObs
Label:      Number of Observations
Template:   Stat.Reg.NObs
Path:       Reg.MODEL2.Fit.Age.NObs
-------------

.
.
.

Output Added:
-------------
Name:       FitPlot
Label:      Fit Plot
Template:   Stat.REG.Graphics.Fit
Path:       Reg.MODEL2.ObswiseStats.Age.FitPlot
-------------
```

Alternatively, you can specify the LISTING option (**ods trace on / listing;**), which writes the trace record, interleaved with the procedure output, to the LISTING destination (if it is open).

The trace record contains the name of each output object created and its associated label, template, and fully qualified name path. The label provides a description of the table or graph. The fully qualified name path shows the output hierarchy. (An example of the hierarchy is shown in Figure 23.5. The SAS Results window displays the labels, rather than the names of objects, but the hierarchy is the same for both names and labels.) The hierarchy has a level for the REG procedure, a level for the model (MODEL1 or MODEL2), a level for the fit results, a level for the dependent variable (Weight or Age), and a level for the name (for example, NObs, ANOVA, FitStatistics, ParameterEstimates).

When you specify ODS objects in an ODS statement, you can often omit the first few levels and instead use a partially qualified name path. A partially qualified name path consists of any part of the fully qualified name path that begins immediately after a period and continues to the end of the fully qualified name path. For example, the table **Reg.Model1.Fit.Weight.ParameterEstimates** can be referenced in any of the following ways:

ParameterEstimates	name
Weight.ParameterEstimates	partially qualified name path
Fit.Weight.ParameterEstimates	partially qualified name path
Model1.Fit.Weight.ParameterEstimates	partially qualified name path
Reg.Model1.Fit.Weight.ParameterEstimates	fully qualified name path

When a procedure creates multiple output objects that have the same name, as shown in the preceding trace output, you have several selection options for referring to the object. You can specify the name, a fully qualified name path, or a partially qualified name path in ODS statements such as ODS SELECT, ODS EXCLUDE, or ODS OUTPUT. You can also specify a WHERE clause. For example, you can specify any of the following statements (in addition to other possibilities) to display both tables of parameter estimates:

```
ods select ParameterEstimates;

ods select Weight.ParameterEstimates Age.ParameterEstimates;

ods select Reg.Model1.Fit.Weight.ParameterEstimates
```

```
            Reg.Model2.Fit.Age.ParameterEstimates;

ods select where = (_path_ ? 'Parameter');
```

The first ODS SELECT statement specifies the object name, which is shared by both tables. The second statement specifies a partially qualified name path for both tables. The third statement specifies the fully qualified name path for each table. The fourth statement selects every object that contains the string **'Parameter'** anywhere in its path.

In the first three statements, selection is case insensitive. Any combination of uppercase and lowercase letters works. This is not true in the fourth statement, which uses an ordinary SAS comparison of character strings. For case insensitivity in WHERE clause selection, use the LOWCASE function as in the following example:

```
ods select where = (lowcase(_path_) ? 'parameter');
```

You can also select objects based on a WHERE clause and the label path. The following statements turn on the trace record, display a label path in addition to the name path, and select all objects that have the string **'var'** in the label:

```
ods trace on / label;
ods select where = (lowcase(_label_) ? 'var');
```

A subset of the trace record for PROC REG with this ODS SELECT list, showing just the name path and label path, is as follows:

```
Path:       Reg.MODEL1.Fit.Weight.ANOVA
Label Path: 'The Reg Procedure'.'MODEL1'.'Fit'.Weight.'Analysis of Variance'
Path:       Reg.MODEL2.Fit.Age.ANOVA
Label Path: 'The Reg Procedure'.'MODEL2'.'Fit'.Age.'Analysis of Variance'
```

The ODS SELECT statement selects the ANOVA tables, because they have the string **'Analysis of Variance'** (which when lowercased contains **'var'**) in their labels. WHERE clause selection is also useful for selecting all of the objects within a group or level of the path hierarchy (the group **'MODEL2'** or **'Fit'**). You can specify any part of the name path or label path—for example, **'.Age.'** matches the variable Age and ignores any **'Age'** that might be in the middle of a word, **'2.F'** matches Model 2 fit tables and any other object that has the string **'2.F'** in its path, and so on.

ODS records the specified object names in its internal selection or exclusion list, and then it processes the output it receives. ODS maintains an overall selection or exclusion list that pertains to all ODS destinations, and it maintains a separate selection or exclusion list for each ODS destination. The list for a specific destination provides the primary filtering step. The restrictions that you specify in the overall list are added to the destination-specific lists.

Suppose, for example, that your LISTING exclusion list (that is, the list of objects you want to exclude from the LISTING destination) contains the FitStatistics table, which you specify with the following statement:

```
ods listing exclude FitStatistics;
```

Suppose also that your overall selection list (that is, the list of objects you want to select for all destinations) contains the tables ParameterEstimates and FitStatistics, which you specify with the following statement:

```
ods select ParameterEstimates FitStatistics;
```

ODS then sends only the ParameterEstimates and FitStatistics tables to all open destinations except the LISTING destination. It sends only the ParameterEstimates table to the LISTING destination because the table FitStatistics is excluded from that destination.

RUN-Group Processing

Some SAS procedures, such as PROC REG and PROC GLM, support RUN-group processing, which means that a RUN statement does not end the procedure. A QUIT statement explicitly ends such procedures. If you omit the QUIT statement, a PROC or a DATA statement implicitly ends such procedures. When you use ODS with procedures that support RUN-group processing, it is good programming practice to specify a QUIT statement at the end of the procedure. This causes ODS to clear the selection or exclusion list, and you are less likely to encounter unexpected results. See Example 23.6 for more information about RUN-group processing with interactive procedures.

The SAS Results Window

The SAS Results window contains a running record of the output from your SAS session. In the SAS windowing environment, select **View ► Results** to open the Results window. Figure 23.5 displays the Results window from the PROC REG step shown previously.

Figure 23.5 The Results Window from the SAS Explorer

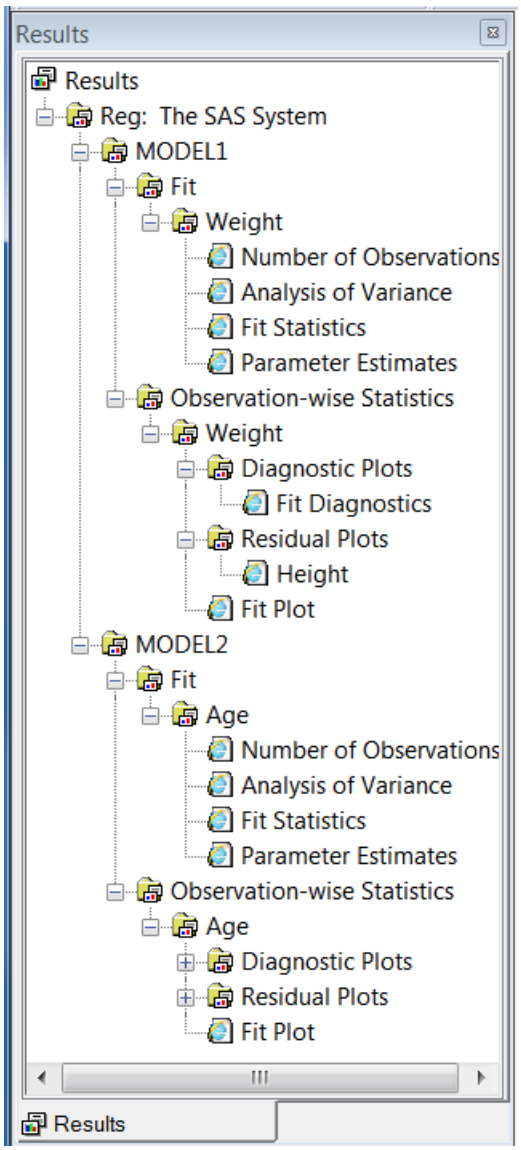

When you click the output names in the Results window, you link directly to the output in the Output Results window (for the HTML destination) or the Output window or graph viewer window (for the LISTING destination). The Results window contains an entry for each level of the label path and for each object. You can also use the Results window to determine the names of the templates that are associated with each object. Right-click the name and select **Properties**. You can see all the templates from the Results window by selecting **View ► Templates**. For SAS/STAT, select **Sashelp.Tmplstat**. Then select a procedure such as **REG** and a template such as **ParameterEstimates**.

The ODS PATH Statement

The ODS PATH statement controls where ODS stores new templates that you create and where ODS finds the templates that your programs use.[2] Compiled templates are stored in a template store, which is a type of item store. (An item store is a special type of SAS file.)

By default, the templates that you write are stored in Sasuser.Templat, and the templates that the SAS System provides are stored in Sashelp.Tmplmst and other Sashelp template stores. By default, ODS retrieves templates from Sashelp template stores unless you compile and store copies in Sasuser.Templat.

You can see the list of active template stores by submitting the following statement:

```
ods path show;
```

By default, the results are as follows:

```
Current ODS PATH list is:

1. SASUSER.TEMPLAT(UPDATE)
2. SASHELP.TMPLMST(READ)
```

You can see a list of all of the templates in a template store as follows:

```
proc template;
   list / store=sasuser.templat;
run;
```

See the sections "The Master Template Store" on page 606 and "Controlling Output Appearance with Templates" on page 607 for more information about the template search path and template stores.

The Master Template Store

By default, the ODS path includes the name of the template store that provides the templates that are shipped with the SAS system. You can use the ODS PATH SHOW statement to see the list of active template stores:

```
ods path show;
```

By default, the results are as follows:

```
Current ODS PATH list is:

1. SASUSER.TEMPLAT(UPDATE)
2. SASHELP.TMPLMST(READ)
```

The ODS PATH statement template search path name Sashelp.Tmplmst refers to the template stores that are shipped with the SAS System. More precisely, the ODS PATH name Sashelp.Tmplmst refers to multiple template store files including Sashelp.Tmplmst, Sashelp.Tmplstat, Sashelp.Tmplets, Sashelp.Tmplqc,

[2] Other types of paths include the name path and label path, which are discussed in the section "Paths and Selection" on page 600.

Sashelp.Tmpliml, and others. The name Sashelp.Tmplmst refers to both the entire template store that is shipped with the SAS System and one particular file in that template store. In earlier releases of the SAS System, there was just one template store file, namely Sashelp.Tmplmst, and there was a one-to-one correspondence between the ODS PATH statement name and the template store file name. Now there are multiple files (because certain products have their own template store files), but the ODS PATH statement syntax for selecting all of them is unchanged. You do not need to be concerned about this, and you should not specify any of these template stores individually. You simply specify Sashelp.Tmplmst to get the item stores that are shipped with the SAS System. However, you will see these other names sometimes in the SAS log and output when you are working with templates, so you need to know what that means.

If you see a template store name of the form Sashelp followed by a period and a name, it refers to part of the overall template store that you can reference with the name Sashelp.Tmplmst. For example, submit the following step:

```
proc template;
   source Stat.REG.ANOVA;
run;
```

The following is displayed in the SAS log file:

```
NOTE: Path 'Stat.Reg.ANOVA' is in: SASHELP.TMPLSTAT (via SASHELP.TMPLMST).
```

Submit the following step:

```
proc template;
   list Stat.REG;
   list ETS.ARIMA;
   list QC.Shewhart;
run;
```

The names Sashelp.Tmplstat, Sashelp.Tmplets, and Sashelp.Tmplqc are displayed in the headers of the tables that list the procedure templates.

Controlling Output Appearance with Templates

A template is a description of how output should appear when it is formatted. Templates describe several characteristics of the output, including headers, column ordering, style information, justification, and formats. Each object in the output has a template, and all SAS templates are stored in the Sashelp library. You can find the template associated with a particular output object or table column by using the ODS TRACE statement or the SAS Results window. You can create or modify a template with the TEMPLATE procedure. For example, you can specify different column headings or different orders of columns in a table.

There are a number of different types of templates including column and table templates, graphical templates, and style templates. A column or table template applies to the specific columns or tables that refer to the template. Graphical templates are discussed in more detail in Chapter 24, “Statistical Graphics Using ODS.” A style template applies to an entire SAS program, including all tables and graphs, and can be specified with the STYLE= option in a valid ODS destination, such as HTML, RTF, or PDF. You can specify a style as follows:

```
ods html style=HTMLBlue;
```

A style template controls stylistic elements such as colors, fonts, and presentation attributes. You can change the style to give your output different looks and color schemes. You can also refer to style information in table templates for individual headers and data cells. You can modify all types of templates with PROC TEMPLATE. For information about creating your own styles, see the *SAS Output Delivery System: User's Guide*.

You can display the contents of a template by running PROC TEMPLATE with a SOURCE statement and a template name, as in the following example:

```
proc template;
   source Stat.REG.ANOVA;
   source Stat.GLM.OverallANOVA;
run;
```

In many cases, a template definition is based at least in part on another template. When you see the PARENT=*template* option in a template definition, you need to look at the specified template to learn more about the rest of the template definition. To illustrate, consider the following PROC GLM step:

```
proc glm data=sashelp.class;
   model height=weight;
quit;
```

The ANOVA table from this step is displayed in Figure 23.6.

Figure 23.6 PROC GLM ANOVA Table with the Default Template

Getting Started with ODS

The GLM Procedure

Dependent Variable: Height

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	364.5762619	364.5762619	57.08	<.0001
Error	17	108.5879486	6.3875264		
Corrected Total	18	473.1642105			

The sums of squares and mean squares are presented with eight decimal places. You can change the templates to change the formats of those columns to use fewer decimal places. First, you can use the ODS TRACE statement when you run PROC GLM to determine the name of the template:

```
ods trace output;
proc glm data=sashelp.class;
   model height=weight;
quit;
ods trace off;
```

The trace output results include the following:

```
Output Added:
```

```
-------------
Name:       OverallANOVA
Label:      Overall ANOVA
Template:   stat.GLM.OverallANOVA
Path:       GLM.ANOVA.Height.OverallANOVA
-------------
```

From this, you can see that the template for the overall ANOVA table is **stat.GLM.OverallANOVA**. You can submit the following statements to see the overall ANOVA table template:

```
proc template;
   source stat.glm.overallanova;
run;
```

The results are as follows:

```
define table Stat.GLM.Overallanova;
   notes "Over-all ANOVA";
   top_space = 1;
   parent = Stat.GLM.ANOVA;
   double_space;
end;
```

The results show that this template inherits its definition from a parent template named **Stat.GLM.ANOVA**. Submit the following statements to see the parent template:

```
proc template;
   source stat.glm.anova;
run;
```

Some of the results are as follows:

```
define SS;
   parent = Stat.GLM.SS;
end;

define MS;
   parent = Stat.GLM.MS;
end;
```

These columns inherit their definitions from the parent columns named **Stat.GLM.SS** and **Stat.GLM.MS**. This is all of the information that you need to redefine these columns, but you can run PROC TEMPLATE again as follows to see more information about how these templates are defined:

```
proc template;
   source Stat.GLM.SS;
   source Stat.GLM.MS;
run;
```

The results are as follows:

```
define column Stat.GLM.Ss;
   notes "Parent for GLM ANOVA Sums of Squares columns";
   parent = Common.ANOVA.SS;
end;
define column Stat.GLM.Ms;
   notes "Parent for GLM ANOVA Mean Squares columns";
   parent = Common.ANOVA.MS;
end;
```

These columns inherit their definitions from the columns named **Common.ANOVA.SS** and **Common.ANOVA.MS**. You can run PROC TEMPLATE as follows to see their definitions:

```
proc template;
   source Common.ANOVA.SS;
   source Common.ANOVA.MS;
run;
```

The results are as follows:

```
define column Common.ANOVA.Ss;
   notes "Default ANOVA Sum of squares column";
   header = "Sum of Squares";
   translate _val_=._ into "";
end;
define column Common.ANOVA.Ms;
   notes "Default ANOVA Mean square column";
   header = "Mean Square";
   translate _val_=._ into "";
end;
```

You can redefine **Common.ANOVA.SS** and **Common.ANOVA.MS** to change all **SS** and **MS** columns in ANOVA tables. This would be the most general redefinition. More specifically, you can redefine **Stat.GLM.SS** and **Stat.GLM.MS** to change **SS** and **MS** columns in ANOVA tables produced by PROC GLM. Finally, and most specifically, you can change the **SS** and **MS** columns in just the overall ANOVA table template.

In this example, the **Stat.GLM.SS** and **Stat.GLM.MS** columns are redefined as follows, so that results are displayed with fewer decimal places:

```
proc template;
   edit Stat.GLM.SS;
      choose_format=max format_width=8;
   end;
   edit Stat.GLM.MS;
      choose_format=max format_width=8;
   end;
run;
```

The CHOOSE_FORMAT=MAX option along with the FORMAT_WIDTH=8 option chooses the format for each column based on the maximum value in that column and an overall width of eight. You are editing and not replacing the definition, so the column header and other information in the definition is not lost. The following step uses the new templates:

```
proc glm data=sashelp.class;
   model height=weight;
quit;
```

The new ANOVA results, using the edited templates, are shown in Figure 23.7. You can see that the original results in Figure 23.6 have eight decimal places, whereas the new results in Figure 23.7 have only five decimal places and an overall format width of eight.

Figure 23.7 PROC GLM ANOVA Table after Template Customization

Getting Started with ODS

The GLM Procedure

Dependent Variable: Height

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	364.5763	364.5763	57.08	<.0001
Error	17	108.5879	6.3875		
Corrected Total	18	473.1642			

The preceding PROC TEMPLATE step produces the following notes:

```
NOTE: Overwriting existing template/link: Stat.GLM.Ss
NOTE: COLUMN 'Stat.GLM.Ss' has been saved to: SASUSER.TEMPLAT
NOTE: Overwriting existing template/link: Stat.GLM.Ms
NOTE: COLUMN 'Stat.GLM.Ms' has been saved to: SASUSER.TEMPLAT
```

When you run PROC TEMPLATE to modify or edit a template, the template is stored by default in your Sasuser library. You can delete your custom template and restore the default template as follows:

```
proc template;
   delete Stat.GLM.SS / store=sasuser.templat;
   delete Stat.GLM.MS / store=sasuser.templat;
run;
```

The preceding PROC TEMPLATE step produces the following notes:

```
NOTE: 'Stat.GLM.SS' has been deleted from: SASUSER.TEMPLAT
NOTE: 'Stat.GLM.MS' has been deleted from: SASUSER.TEMPLAT
```

It is good practice to delete any template redefinitions that you do not want to be permanent, because otherwise they persist beyond the duration of your SAS session. The option STORE=SASUSER.TEMPLAT is not required. However, if you have administrator privileges on your computer, this option helps you ensure that you do not accidentally delete templates from Sashelp.Tmplmst.

You can modify the template search path with the ODS PATH statement—for example, so you can access these new templates in a later SAS session. This enables you to create a new default set of templates to modify the display format for all of your SAS output. You can specify the SHOW option in the ODS PATH statement to determine the current template search path. The following statements illustrate the template search path:

```
ods path show;
libname mytpls '.';
ods path (prepend) mytpls.template(update);
ods path show;

proc template;
   edit Stat.GLM.SS;
      choose_format=max format_width=8;
   end;
   edit Stat.GLM.MS;
      choose_format=max format_width=8;
   end;
run;
```

The results of the first statement are as follows:

```
Current ODS PATH list is:

1. SASUSER.TEMPLAT(UPDATE)
2. SASHELP.TMPLMST(READ)
```

This shows that the Sasuser.Templat template store is open for storing new templates and retrieving templates for use. After that, the Sashelp.Tmplmst template store is used, but it is open only for read access.[3] The LIBNAME and second ODS PATH statements add a template store to the front of this list in the current directory. The final ODS PATH SHOW statement shows the new template search path, which is as follows:

```
Current ODS PATH list is:

1. MYTPLS.TEMPLATE(UPDATE)
2. SASUSER.TEMPLAT(UPDATE)
3. SASHELP.TMPLMST(READ)
```

The PROC TEMPLATE step produces the following notes, which show that the templates are now stored in MYTPLS.TEMPLATE:

```
NOTE: Overwriting existing template/link: Stat.GLM.Ss
NOTE: COLUMN 'Stat.GLM.Ss' has been saved to: MYTPLS.TEMPLATE
NOTE: Overwriting existing template/link: Stat.GLM.Ms
NOTE: COLUMN 'Stat.GLM.Ms' has been saved to: MYTPLS.TEMPLATE
```

In all cases, the original template definitions in Sashelp.Tmplmst are not changed. You can delete your custom template and restore the default template as follows:

[3]Most SAS users cannot modify templates in Sashelp. However, if you have computer administrator privileges, you might be able to modify templates in Sashelp, so you should be careful to not do so.

```
proc template;
   delete Stat.GLM.SS / store=mytpls.template;
   delete Stat.GLM.MS / store=mytpls.template;
run;
```

If you would like all template modifications to be automatically deleted at the end or your SAS session, you can modify the template search path so that an updatable template store is placed in the Work directory in the front of the current path in either of the two equivalent ways:

```
ods path (prepend) work.templat(update);
ods path work.templat(update) sasuser.templat(update) sashelp.tmplmst(read);
```

Alternatively, you can replace Sasuser.Templat with Sashelp.Templat as follows:

```
ods path work.templat(update) sashelp.tmplmst(read);
```

When you are done, you can reset the default template search path as follows:

```
ods path reset;
```

ODS and the NOPRINT Option

Many SAS procedures support a NOPRINT option that you can use when you want to create an output data set without displaying any output. You use an option (such as the OUTEST= option or an OUTPUT statement with an OUT= option) in addition to the procedure's NOPRINT option to create a data set and suppress displayed output.

You can also use the ODS OUTPUT statement to create output data sets. However, if you specify the NOPRINT option, the procedure might not send any output to ODS. In most procedures that support a NOPRINT option, NOPRINT means no ODS. (However, there are a few procedures that for historical reasons still might produce some output even when NOPRINT is specified.) When you want to create output data sets through the ODS OUTPUT statement and you want to suppress the display of all output, specify the following statement instead of using the NOPRINT option:

```
ods select none;
```

Alternatively, you can close the active ODS destinations like this:

```
ods _all_ close;
```

ODS statements do not instruct a procedure to generate output. Instead, they specify how ODS should manage output after it is created. You must ensure that the proper procedure options are in effect, or the output is not generated. For example, the following statements do not create the requested data set Parms because the SOLUTION option is not specified in the MODEL statement:

```
proc glm data=sashelp.class;
   ods output ParameterEstimates=Parms;
   class sex;
   model height=sex;
quit;
```

Since PROC GLM did not create the table, ODS cannot make the output data set. When you execute these statements, the following message is displayed in the log:

```
WARNING: Output 'ParameterEstimates' was not created.
```

The following step creates the output data set:

```
proc glm data=sashelp.class;
   ods output ParameterEstimates=Parms;
   class sex;
   model height=sex / solution;
quit;
```

Examples: Using the Output Delivery System

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

This section provides examples of creating HTML output, selecting and excluding output, tracing ODS output, using the Results window, creating ODS output data sets, modifying templates, creating hyperlinks, and using ODS Graphics.

Example 23.1: Creating HTML Output with ODS

(View the complete code for this example (odsex1.sas) in the example repository.)

This example demonstrates how you can use the ODS HTML statement to display your output in HTML. The following statements create the data set Scores, which contains the golf scores of boys and girls in a physical education class:

```
title 'Comparing Group Means';

data Scores;
   input Gender $ Score @@;
   datalines;
f 75  f 76  f 80  f 77  f 80  f 77  f 73
m 82  m 80  m 85  m 85  m 78  m 87  m 82
;
```

The TTEST procedure is used to compare the scores. The ODS HTML statement specifies the name of the file to contain the HTML output. The following statements create the HTML file *ttest.htm*:

```
ods html body='ttest.htm' style=HTMLBlue;

proc ttest;
   class Gender;
   var Score;
run;

ods html close;
```

In some cases, the LISTING destination is open by default. See the section "Output Defaults" on page 592 for more information about default destinations. In this example, the ODS HTML statement opens the HTML destination. All open destinations receive the generated output. If you are in the SAS windowing environment and are using the internal browser, you do not need to close the HTML destination before viewing your output. However, when you write to an HTML file, you must specify the following statement before you can view your output in an external browser:

```
ods html close;
```

If you do not close the HTML destination, your HTML file might contain no output or incomplete output, or you might experience other unexpected results.

The following statements use ODS to display the output in HTML with a table of contents:

```
ods _all_ close;
ods html body='ttest.htm' contents='ttestc.htm' frame='ttestf.htm'
         style=HTMLBlue;
ods graphics on;

proc ttest;
   class Gender;
   var Score;
run;

ods html close;
ods html;
ods pdf;
```

The ODS _ALL_ CLOSE statement closes all open destinations. The ODS HTML statement specifies three files and the HTMLBLUE style of output. The BODY= option specifies the file that contains the SAS output. The CONTENTS= option specifies the file that contains the table of contents. The FRAME= option specifies the file that displays both the table of contents and the output. You can open the FRAME= file (*ttestf.htm*) in your browser to view the table of contents together with the generated output (see Output 23.1.1). By default, the HTML files are generated in your current working directory. You can instead specify a path, such as **frame='html/ttestf.htm'**, to store a file in a subdirectory.

If you specify the ODS HTML statement with only the BODY= argument, no table of contents is created. The table of contents contains the descriptive label for each output object produced in the PROC TTEST step. You can select any label in the table of contents, and the corresponding output is displayed on the right side of the browser window.

The ODS GRAPHICS ON statement enables ODS Graphics, which creates the graph displayed in Output 23.1.1. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Output 23.1.1 HTML Output with a Table of Contents and a Frame

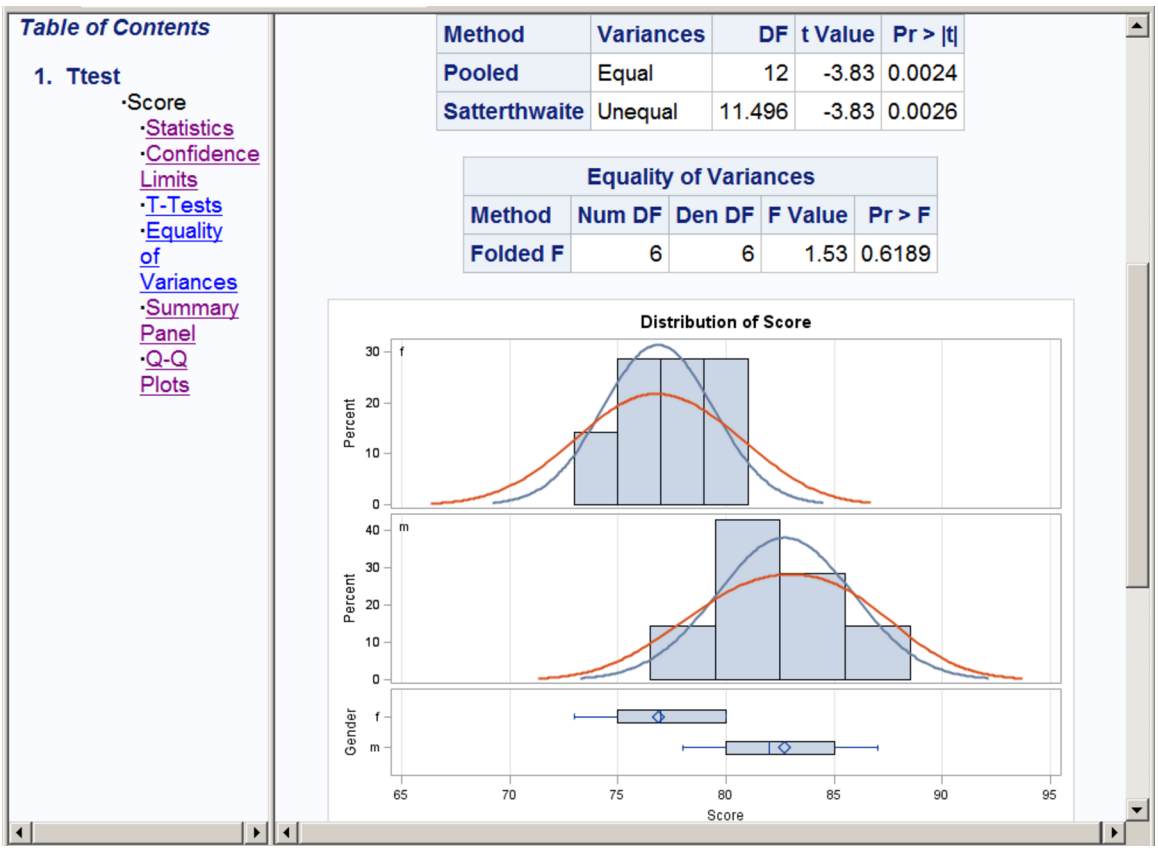

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	12	-3.83	0.0024
Satterthwaite	Unequal	11.496	-3.83	0.0026

Equality of Variances				
Method	Num DF	Den DF	F Value	Pr > F
Folded F	6	6	1.53	0.6189

Example 23.2: Selecting ODS Tables for Display

(View the complete code for this example (odsex2.sas) in the example repository.)

You can use the ODS SELECT statement to deliver only a subset of the tables or graphs to ODS destinations. The following statements create an input SAS data set and use PROC GLM to perform an analysis of an unbalanced two-way experimental design:

```
title 'Unbalanced Two-way Design';
data twoway;
   input Treatment Block y @@;
   datalines;
1 1 17   1 1 28   1 1 19   1 1 21   1 1 19   1 2 43
1 2 30   1 2 39   1 2 44   1 2 44   1 3 16
2 1 21   2 1 21   2 1 24   2 1 25   2 2 39   2 2 45
2 2 42   2 2 47   2 3 19   2 3 22   2 3 16
3 1 22   3 1 30   3 1 33   3 1 31   3 2 46   3 3 26
3 3 31   3 3 26   3 3 33   3 3 29   3 3 25
;
```

```
proc glm data=twoway;
   class Treatment Block;
   model y = Treatment | Block;
   means Treatment;
   lsmeans Treatment;
   ods select ModelANOVA Means;
   ods trace on;
   ods show;
run;
```

The ODS SELECT statement selects only two tables (ModelANOVA and Means) for display in the ODS destinations. In this example, no ODS destinations are explicitly opened. Therefore, only the default destination (usually LISTING or HTML) receives the procedure output. See the section "Output Defaults" on page 592 for more information about default destinations. The ODS SHOW statement displays the current overall selection list in the SAS log. The ODS SHOW statement is not required; it is used here simply to show the effects of the ODS SELECT statement. The results of the ODS SHOW statement are as follows:

```
Current OVERALL select list is:
1. ModelANOVA
2. Means
```

The ODS TRACE statement writes the trace record of the ODS output objects to the SAS log. The trace record is as follows:

```
Output Added:
-------------
Name:       ModelANOVA
Label:      Type I Model ANOVA
Template:   stat.GLM.Tests
Path:       GLM.ANOVA.y.ModelANOVA
-------------

Output Added:
-------------
Name:       ModelANOVA
Label:      Type III Model ANOVA
Template:   stat.GLM.Tests
Path:       GLM.ANOVA.y.ModelANOVA
-------------

Output Added:
-------------
Name:       Means
Label:      Means
Template:   stat.GLM.Means
Path:       GLM.Means.Treatment.Means
-------------
```

There are two tables with the name ModelANOVA. One contains the "Type I Model ANOVA" table, and the other contains the "Type III Model ANOVA" table. If you want to select only one of them, you can specify either of the labels in the ODS SELECT statement instead of the name. You specify one of the following:

```
ods select 'Type I Model ANOVA' Means;
ods select 'Type III Model ANOVA' Means;
```

In the following statements, the ODS SHOW statement writes the current overall selection list to the SAS log, the QUIT statement ends the PROC GLM step, and the second ODS SHOW statement writes the selection list to the log after PROC GLM terminates:

```
ods show;
quit;
ods show;
```

The results of these statements are as follows:

```
ods show;

 Current OVERALL select list is:
 1. ModelANOVA
 2. Means


quit;
ods show;

 Current OVERALL select list is: ALL
```

PROC GLM supports interactive RUN-group processing. Before the QUIT statement is executed, PROC GLM is active and the ODS selection list remains at its previous setting. The list includes only the two tables, ModelANOVA and Means. After the QUIT statement, when PROC GLM is no longer active, the selection list is reset to ALL. The displayed output, shown in Output 23.2.1, consists of the three selected tables (two ModelANOVA tables and the Means table). The LS-means results are not displayed even though an LSMEANS statement was specified. This is because the LS-means table, named LSMeans, is not specified in the ODS SELECT statement. Other tables are suppressed as well.

Output 23.2.1 Selected Tables from PROC GLM

Unbalanced Two-way Design

The GLM Procedure

Dependent Variable: y

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Treatment	2	8.060606	4.030303	0.24	0.7888
Block	2	2621.864124	1310.932062	77.95	<.0001
Treatment*Block	4	32.684361	8.171090	0.49	0.7460

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Treatment	2	266.130682	133.065341	7.91	0.0023
Block	2	1883.729465	941.864732	56.00	<.0001
Treatment*Block	4	32.684361	8.171090	0.49	0.7460

Output 23.2.1 *continued*

Unbalanced Two-way Design

The GLM Procedure

		y	
Level of Treatment	**N**	**Mean**	**Std Dev**
1	11	29.0909091	11.5104695
2	11	29.1818182	11.5569735
3	11	30.1818182	6.3058414

For more information about ODS exclusion and selection lists, see the section "The ODS Statement" on page 599.

Example 23.3: Excluding ODS Tables from Display

(View the complete code for this example (odsex3.sas) in the example repository.)

The following example demonstrates how you can use the ODS EXCLUDE statement to exclude particular objects from ODS destinations. This example also creates a SAS data set from the excluded table and uses it to create a specialized plot.

The data are from Hemmerle and Hartley (1973). The response variable consists of measurements from an oven experiment, and the model contains a fixed effect a and random effects b and a*b. The following statements create the input SAS data set:

```
title 'Oven Measurements';

data hh;
   input a b y @@;
   datalines;
1 1 237   1 1 254    1 1 246
1 2 178   1 2 179
2 1 208   2 1 178    2 1 187
2 2 146   2 2 145    2 2 141
3 1 186   3 1 183
3 2 142   3 2 125    3 2 136
;
```

The following ODS statements are submitted before the analysis, which will be done with the MIXED procedure:

```
ods _all_ close;
ods html body='mixed.htm' contents='mixedc.htm' frame='mixedf.htm'
         style=HTMLBlue;

ods exclude ParmSearch(persist);
ods show;
```

The ODS HTML statement specifies the file names to contain the output generated from the statements that follow. The ODS EXCLUDE statement excludes the table ParmSearch from display. Although the table is excluded from the displayed output, the information contained in the ParmSearch table is graphically summarized in a later step.

The PERSIST option in the ODS EXCLUDE statement excludes the object for the entire SAS session or until you execute an ODS SELECT statement or an ODS EXCLUDE NONE statement. If you omit the PERSIST option, the exclusion list is cleared when the procedure terminates. The resulting exclusion list is displayed next:

```
Current OVERALL exclude list is:
1. ParmSearch(PERSIST)
```

The MIXED procedure is run to fit the model:

```
proc mixed data=hh;
   class a b;
   model y = a;
   random b a*b;
   parms (17 to 20 by 0.1) (.3 to .4 by .005) (1.0);
   ods output ParmSearch=parms;
run;

ods show;
```

All output from PROC MIXED, except the ParmSearch table, is delivered to the HTML destination. The ODS OUTPUT statement outputs the table ParmSearch to a SAS data set called Parms.

The ODS SHOW statement again displays the overall current exclusion list after PROC MIXED has terminated. The results of the ODS SHOW statement are displayed next:

```
Current OVERALL exclude list is:
1. ParmSearch(PERSIST)
```

The ParmSearch table is saved in the Parms data set (as specified in the ODS OUTPUT statement). The following steps plot the surface of the residual log likelihood as a function of the covariance parameters and produce Output 23.3.1:

```
proc template;
   define statgraph surface;
      begingraph;
         layout overlay3d;
            surfaceplotparm x=CovP1 y=CovP2 z=ResLogLike;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=parms template=surface;
run;

ods html close;
```

PROC TEMPLATE is used to create a template for displaying the data as a three-dimensional surface plot. The plot is displayed with the ODS Graphics procedure SGRENDER. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Output 23.3.1 HTML Output from PROC MIXED

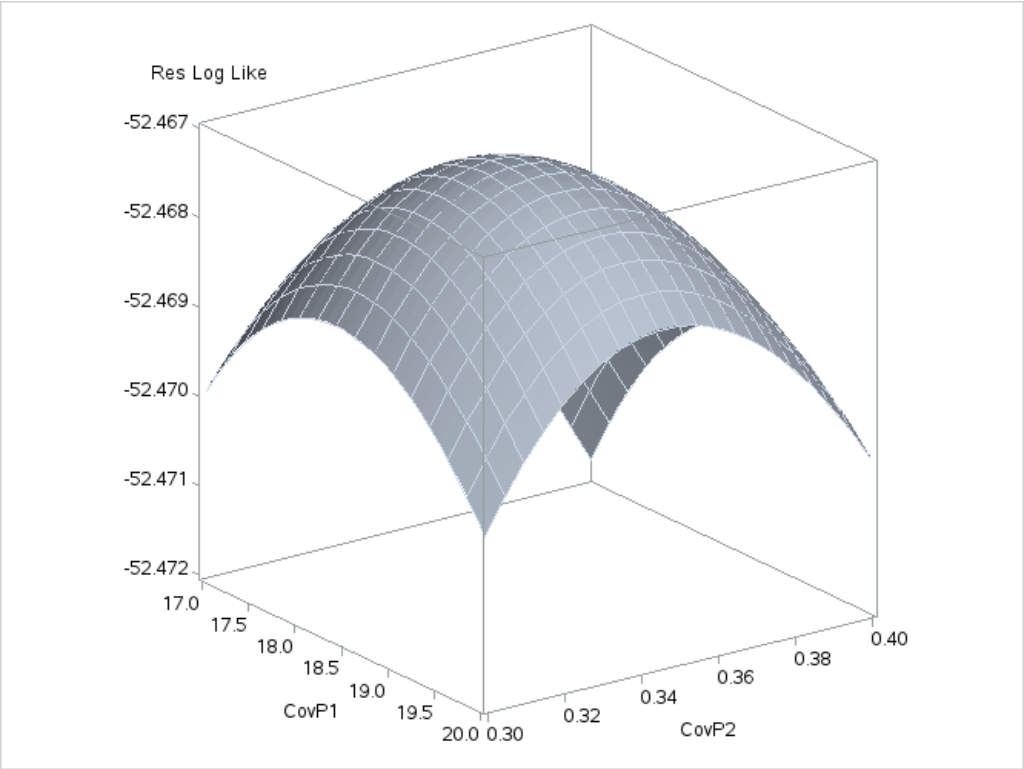

Example 23.4: Creating an Output Data Set from an ODS Table

(View the complete code for this example (odsex4.sas) in the example repository.)

In this example, the GENMOD procedure is used to perform Poisson regression, and part of the resulting procedure output is written to a SAS data set with the ODS OUTPUT statement. Insurance claims data are classified by two factors: age group (with two levels) and car type (with three levels). The following statements create the data set Insure:

```
title 'Insurance Claims';

data Insure;
   input n c Car $ Age;
   ln = log(n);
   datalines;
 500   42  Small  1
1200   37  Medium 1
 100    1  Large  1
 400  101  Small  2
 500   73  Medium 2
```

```
 300   14  Large  2
;
```

The variable n represents the number of insurance policyholders, and the variable c represents the number of insurance claims. The variable Car represents the type of car involved (classified into three groups), and the variable Age is the age of a policyholder (classified into two groups).

You can use PROC GENMOD to perform a Poisson regression analysis of these data with a log link function. Assume that the number-of-claims variable, c, has a Poisson probability distribution and the log of its mean, μ_i, is related to the factors Car and Age.

The following statements obtain the names of the objects produced by this PROC GENMOD run. The ODS TRACE statement lists the trace record. If you already know the names, such as by looking them up in the procedure documentation, you do not have to run this step. The following step displays the trace information:

```
ods trace on;

proc genmod data=insure;
   class car age;
   model c = car age / dist=poisson link=log offset=ln obstats;
run;

ods trace off;
```

The trace record from the SAS log is displayed next:

```
Output Added:
-------------
Name:       ModelInfo
Label:      Model Information
Template:   Stat.Genmod.ModelInfo
Path:       Genmod.ModelInfo
-------------

Output Added:
-------------
Name:       NObs
Label:      Number of observations summary
Template:   Stat.Genmod.NObs
Path:       Genmod.NObs
-------------

Output Added:
-------------
Name:       ClassLevels
Label:      Class Level Information
Template:   Stat.Genmod.Classlevels
Path:       Genmod.ClassLevels
-------------


Output Added:
-------------
Name:       ParmInfo
Label:      Parameter Information
```

```
Template:   Stat.Genmod.Parminfo
Path:       Genmod.ParmInfo
-------------

Output Added:
-------------
Name:       ModelFit
Label:      Criteria For Assessing Goodness Of Fit
Template:   stat.genmod.ModelFit
Path:       Genmod.ModelFit
-------------

Output Added:
-------------
Name:       ConvergenceStatus
Label:      Convergence Status
Template:   Stat.Genmod.ConvergenceStatus
Path:       Genmod.ConvergenceStatus
-------------

Output Added:
-------------
Name:       ParameterEstimates
Label:      Analysis Of Parameter Estimates
Template:   stat.genmod.parameterestimates
Path:       Genmod.ParameterEstimates
-------------

Output Added:
-------------
Name:       ObStats
Label:      Observation Statistics
Template:   Stat.Genmod.Obstats
Path:       Genmod.ObStats
-------------
```

In the following step, no output is displayed because the ODS SELECT NONE statement is included. The ODS OUTPUT statement writes the ODS table ObStats to a SAS data set named myObStats. All of the usual data set options, such as the KEEP= or RENAME= option, can be used in the ODS OUTPUT statement. Thus, to create the myObStats data set so that it contains only certain columns from the ObStats table, you can use the data set options as follows:

```
ods select none;
proc genmod data=insure;
   class car age;
   model c = car age / dist=poisson link=log offset=ln obstats;
   ods output ObStats=myObStats(keep=car age pred
                                rename=(pred=PredictedValue));
run;
```

The KEEP= data set option in the ODS OUTPUT statement specifies that only the variables Car, Age, and Pred are written to the data set. The RENAME= data set option changes the name of variable Pred to

PredictedValue. The following statements sort the output data set myObStats, select all output, and produce Output 23.4.1:

```
proc sort data=myObStats;
   by descending PredictedValue;
run;

ods select all;
proc print data=myObStats noobs;
   title2 'Values of Car, Age, and the Predicted Values';
run;
```

The ODS SELECT NONE statement remains in effect until it is explicitly canceled (for example, with the ODS SELECT ALL statement).

Output 23.4.1 The ObStats Table Created as a SAS Data Set

Insurance Claims
Values of Car, Age, and the Predicted Values

Car	Age	PredictedValue
Small	2	107.2011
Medium	2	67.025444
Medium	1	42.974556
Small	1	35.798902
Large	2	13.773459
Large	1	1.2265414

Example 23.5: Creating an Output Data Set: Subsetting the Data

(View the complete code for this example (odsex5.sas) in the example repository.)

This example demonstrates how you can create an output data set with the ODS OUTPUT statement and also use data set selection keywords to limit the output that ODS writes to a SAS data set. The data set, called Color, contains the eye color and hair color of children from two different regions of Europe. The data are recorded as cell counts, where the variable Count contains the number of children who exhibit each of the 15 combinations of eye and hair color. The following statements create the SAS data set:

```
title 'Hair Color of European Children';

data Color;
   input Region Eyes $ Hair $ Count @@;
   label Eyes  ='Eye Color'
         Hair  ='Hair Color'
         Region='Geographic Region';
   datalines;
1 blue  fair   23  1 blue  red     7  1 blue  medium 24
1 blue  dark   11  1 green fair   19  1 green red     7
1 green medium 18  1 green dark   14  1 brown fair   34
1 brown red     5  1 brown medium 41  1 brown dark   40
1 brown black   3  2 blue  fair   46  2 blue  red    21
```

```
2 blue  medium 44  2 blue  dark   40  2 blue  black   6
2 green fair   50  2 green red    31  2 green medium 37
2 green dark   23  2 brown fair   56  2 brown red    42
2 brown medium 53  2 brown dark   54  2 brown black  13
;
```

The following statements exclude all output and sort the observations in the Color data set by the Region variable:

```
ods select none;

proc sort data=Color;
   by Region;
run;
```

The following ODS OUTPUT statement creates the ChiSq table as a SAS data set named myStats:

```
ods output ChiSq=myStats(drop=Table
                         where=(Statistic =: 'Chi' or
                                Statistic =: 'Like'));
```

You specify the table name in the ODS OUTPUT statement.[4] The DROP= data set option excludes variables from the new data set. The WHERE= data set option selects observations for output to the new data set myStats—specifically, those that begin with **'Chi'** or **'Like'**.

The following statements create Output 23.5.1:

```
proc freq data=Color order=data;
   weight Count;
   tables Eyes*Hair / testp=(30 12 30 25 3);
   by Region;
run;

ods select all;
proc print data=myStats noobs;
run;
```

The FREQ procedure is used to create and analyze a crosstabulation table from the two categorical variables Eyes and Hair, for each value of the variable Region.

Output 23.5.1 Output Data Set from PROC FREQ and ODS

Hair Color of European Children

Region	Statistic	DF	Value	Prob
1	Chi-Square	8	12.6331	0.1251
1	Likelihood Ratio Chi-Square	8	14.1503	0.0779
2	Chi-Square	8	18.2839	0.0192
2	Likelihood Ratio Chi-Square	8	23.3021	0.0030

[4] You can obtain the names of the objects created by any procedure in the individual procedure documentation chapter or from the individual procedure section of the SAS online Help system. (See the "ODS Table Names" section in the "Details" section of the documentation.) You can also determine the names of objects with the ODS TRACE statement (see Example 23.4 and Example 23.2).

Example 23.6: RUN-Group Processing

(View the complete code for this example (odsex6.sas) in the example repository.)

Some SAS procedures, such as PROC REG and PROC GLM, are interactive. They enable you to submit a group of statements that end with a RUN statement, and then submit more statement groups, each of which ends with a RUN statement. Each group of statements with its RUN statement is called a RUN group. For example, the following PROC REG step has two RUN groups:

```
proc reg data=sashelp.class;
   var Age;
   model Weight = Height;
run;

   model Weight = Height Age;
run;
quit;
```

Interactive procedures can produce several blocks of output for each of several RUN groups. The procedure stays active until it processes a QUIT statement, it encounters a DATA or PROC statement, or the SAS session ends. However, ODS settings are cleared by default at RUN-group boundaries. You can specify the PERSIST= option to maintain ODS settings across RUN statements for procedures that support RUN-group processing.

Consider a data set that contains US population growth trends:

```
title1 'US Population Study';
title2 'Concatenating Two Tables into One Data Set';

data USPopulation;
   input Population @@;
   retain Year 1780;
   Year=Year+10;
   YearSq=Year*Year;
   Population=Population/1000;
   datalines;
3929 5308 7239 9638 12866 17069 23191 31443 39818 50155
62947 75994 91972 105710 122775 131669 151325 179323 203211
;
```

In the following analysis, which has two RUN groups, PROC REG is used to compute the covariance matrix of the estimates for two different models, and the covariance matrices are saved in a single SAS data set. The PERSIST=RUN option in the ODS OUTPUT statement is required to make this happen. The first RUN group creates a data set that contains the CovB table (the covariance matrix of the estimates):

```
proc reg data=USPopulation;
   ods output covb(persist=run)=Bmatrix;
   var YearSq;
   model Population = Year / covb;
run;
```

The MODEL statement defines the regression model and requests the CovB table. The RUN statement executes PROC REG and the model is fit, producing a covariance matrix of the estimates. The covariance

matrix has two rows and two columns and is shown in Output 23.6.1.

Output 23.6.1 CovB Table for the First Model

US Population Study
Concatenating Two Tables into One Data Set

The REG Procedure
Model: MODEL1
Dependent Variable: Population

Covariance of Estimates		
Variable	**Intercept**	**Year**
Intercept	20393.138485	-10.83821461
Year	-10.83821461	0.0057650078

In the next RUN group, the YearSq variable is added to the model and the model is fit again, producing a covariance matrix of the estimates that has three rows and three columns:

```
   add YearSq;
   print;
quit;
```

The CovB table for the second RUN group is displayed in Output 23.6.2.

Output 23.6.2 CovB Table for the Second Model

Covariance of Estimates			
Variable	**Intercept**	**Year**	**YearSq**
Intercept	711450.62615	-757.2493827	0.2013282695
Year	-757.2493827	0.8061328945	-0.000214361
YearSq	0.2013282695	-0.000214361	5.7010894E-8

If the PERSIST=RUN option is omitted, the selection list is cleared when the RUN statement is encountered, and only the first CovB table is selected. But because the PERSIST=RUN option is specified, the selection list remains in effect throughout the PROC REG step. This ensures that each of the CovB tables is selected and output. The following statements display the ODS OUTPUT data set and create Output 23.6.3:

```
proc print;
   id _run_;
   by _run_;
run;
```

Output 23.6.3 Results of the ODS OUTPUT Statement When the PERSIST Option Is Specified

US Population Study
Concatenating Two Tables into One Data Set

Run Group Number=1

Run	Model	Dependent	Variable	Intercept	Year	YearSq
1	MODEL1	Population	Intercept	20393.138485	-10.83821461	.
	MODEL1	Population	Year	-10.83821461	0.0057650078	.

Output 23.6.3 *continued*

Run	Model	Dependent	Variable	Intercept	Year	YearSq
2	MODEL1.1	Population	Intercept	711450.62615	-757.2493827	0.2013282695
	MODEL1.1	Population	Year	-757.2493827	0.8061328945	-0.000214361
	MODEL1.1	Population	YearSq	0.2013282695	-0.000214361	5.7010894E-8

Even though the two CovB tables do not have the same rows or columns, ODS automatically combines the two tables into one data set.

ODS Statement Placement with Interactive Procedures

Where you place a statement in an interactive procedure is critical. The following examples demonstrate the results of various placements of statements:

- If you submit the following steps to a new SAS session, to an existing SAS session after a QUIT statement, or to an existing SAS session after a noninteractive procedure has completed, the ODS OUTPUT statement creates a SAS data set that contains the PROC GLM parameter estimates:

```
ods output parameterestimates=pm;
proc glm data=sashelp.class;
   class sex;
   model weight = sex | height / solution;
run;
```

- If you submit the following steps, no ODS OUTPUT SAS data set is created:

```
proc reg data=sashelp.class;
   model weight = height;
run;

ods output parameterestimates=pm;
proc glm data=sashelp.class;
   class sex;
   model weight = sex | height / solution;
run;
```

  The preceding steps work as follows:

  - The first three statements (PROC REG, MODEL, and RUN) perform a simple regression analysis and display the parameter estimates table. This completes one RUN group.
  - PROC REG is still active when it encounters the ODS OUTPUT statement. But the first RUN group has completed, so the first parameter estimates table is no longer available for ODS to output.
  - PROC REG terminates because of the PROC GLM statement.
  - The ODS OUTPUT statement has not created an output data set when PROC REG terminates because no model was fit after the first RUN group.

  - SAS prints warning messages because the ODS OUTPUT statement could not output a parameter estimates table.
  - The ODS OUTPUT statement corresponds to the PROC REG step because PROC REG is still active when the statement is encountered. If PROC REG had ended first, then the ODS OUTPUT statement would apply to the following PROC GLM step.

- The ODS OUTPUT statement in the following example outputs the PROC GLM parameter estimates because the QUIT statement ends the PROC REG step:

```
proc reg data=sashelp.class;
   model weight = height;
quit;

ods output parameterestimates=pm;
proc glm data=sashelp.class;
   class sex;
   model weight = sex | height / solution;
run;
```

- The ODS OUTPUT statement in the following example outputs the PROC GLM parameter estimates because the ODS OUTPUT statement appears within the PROC GLM step:

```
proc reg data=sashelp.class;
   model weight = height;
run;

proc glm data=sashelp.class;
   ods output parameterestimates=pm;
   class sex;
   model weight = sex | height / solution;
run;
```

Explicitly Ending an Interactive Procedure

You can end interactive procedures such as PROC GLM and PROC REG by submitting a RUN statement and then a QUIT statement:

```
proc reg data=sashelp.class;
   model weight = height;
run; quit;
```

The RUN statement is superfluous, because the QUIT statement both runs the step and exits the procedure. Alternatively, you can end PROC GLM and PROC REG by submitting only a QUIT statement:

```
proc reg data=sashelp.class;
   model weight = height;
quit;
```

Some interactive procedures behave differently from PROC GLM and PROC REG. The GPLOT procedure does not produce results when you submit a QUIT statement but no RUN statement. You should end the IML procedure by submitting only a QUIT statement, because a RUN statement in PROC IML runs a module.

Example 23.7: ODS Output Data Sets and Using PROC TEMPLATE to Customize Output

(View the complete code for this example (odsex7.sas) in the example repository.)

You can use ODS statements, the DATA step, and PROC TEMPLATE to modify the appearance of your displayed tables or to display results in forms that are not directly produced by any procedure. The following example, similar to that given in Olinger and Tobias (1998), runs an analysis with PROC GLM. This example has several parts. It creates output data sets with the ODS OUTPUT statement, combines and manipulates those data sets, displays the results by using a standard SAS template, modifies a template by using PROC TEMPLATE, and displays the output data sets by using the modified template. Each step works toward the final goal of taking multiple tables and creating a custom display of those tables in a way that cannot be done directly by PROC GLM.

The following statements create a SAS data set named Histamine that contains the experimental data:

```
title1 'Histamine Study';

data Histamine;
   input Drug $12. Depleted $ hist0 hist1 hist3 hist5;
   logHist0 = log(hist0); logHist1 = log(Hist1);
   logHist3 = log(hist3); logHist5 = log(Hist5);
   datalines;
Morphine      N  .04  .20  .10  .08
Morphine      N  .02  .06  .02  .02
Morphine      N  .07 1.40  .48  .24
Morphine      N  .17  .57  .35  .24
Morphine      Y  .10  .09  .13  .14
Morphine      Y  .07  .07  .06  .07
Morphine      Y  .05  .07  .06  .07
Trimethaphan  N  .03  .62  .31  .22
Trimethaphan  N  .03 1.05  .73  .60
Trimethaphan  N  .07  .83 1.07  .80
Trimethaphan  N  .09 3.13 2.06 1.23
Trimethaphan  Y  .10  .09  .09  .08
Trimethaphan  Y  .08  .09  .09  .10
Trimethaphan  Y  .13  .10  .12  .12
Trimethaphan  Y  .06  .05  .05  .05
;
```

The data set comes from a preclinical drug experiment (Cole and Grizzle 1966). In order to study the effect of two different drugs on histamine levels in the blood, researchers administer the drugs to 13 animals and measure the levels of histamine in the animals' blood after 0, 1, 3, and 5 minutes. The response variable is the logarithm of the histamine level.

In the analysis that follows, PROC GLM is used to perform a repeated measures analysis, naming the drug and depletion status as between-subject factors in the MODEL statement and naming post-administration measurement time as the within-subject factor. For more information about this study and its analysis, see Example 53.7 in Chapter 53, "The GLM Procedure."

The following PROC GLM statements begin the analysis:

```
ods graphics off;
ods trace output;

proc glm data=Histamine;
   class Drug Depleted;
   model LogHist0--LogHist5 = Drug Depleted Drug*Depleted / nouni;
   repeated Time 4 (0 1 3 5) polynomial / summary printe;
quit;
```

The portion of the trace output that contains the fully qualified name paths is shown next:

```
Path:       GLM.Data.ClassLevels
Path:       GLM.Data.NObs
Path:       GLM.Repeated.RepeatedLevelInfo
Path:       GLM.Repeated.PartialCorr
Path:       GLM.Repeated.MANOVA.Model.Error.ErrorSSCP
Path:       GLM.Repeated.MANOVA.Model.Error.PartialCorr
Path:       GLM.Repeated.MANOVA.Model.Error.Sphericity
Path:       GLM.Repeated.MANOVA.Model.Time.MultStat
Path:       GLM.Repeated.MANOVA.Model.Time_Drug.MultStat
Path:       GLM.Repeated.MANOVA.Model.Time_Depleted.MultStat
Path:       GLM.Repeated.MANOVA.Model.Time_Drug_Depleted.MultStat
Path:       GLM.Repeated.BetweenSubjects.ModelANOVA
Path:       GLM.Repeated.WithinSubject.ModelANOVA
Path:       GLM.Repeated.WithinSubject.Epsilons
Path:       GLM.Repeated.Summary.Time_1.ModelANOVA
Path:       GLM.Repeated.Summary.Time_2.ModelANOVA
Path:       GLM.Repeated.Summary.Time_3.ModelANOVA
```

The goal here is to output the within-subjects multivariate statistics and the between-subjects ANOVA table to SAS data sets for use in subsequent steps. The following statements run the analysis and save the desired results to output data sets:

```
ods select none;

proc glm data=Histamine;
   class Drug Depleted;
   model LogHist0--LogHist5 = Drug Depleted Drug*Depleted / nouni;
   repeated Time 4 (0 1 3 5) polynomial / summary printe;
   ods output MultStat                   = HistWithin
              BetweenSubjects.ModelANOVA = HistBetween;
quit;

ods select all;
```

No output is displayed due to the ODS SELECT statements. The ODS OUTPUT statement creates two SAS data sets, named HistWithin and HistBetween, from the two ODS tables. This analysis creates the following tables:

```
Path:       GLM.Repeated.MANOVA.Model.Time.MultStat
Path:       GLM.Repeated.MANOVA.Model.Time_Drug.MultStat
Path:       GLM.Repeated.MANOVA.Model.Time_Depleted.MultStat
```

```
Path:        GLM.Repeated.MANOVA.Model.Time_Drug_Depleted.MultStat
Path:        GLM.Repeated.BetweenSubjects.ModelANOVA
```

Here is the full trace output for the model ANOVA table:

```
Output Added:
-------------
Name:        ModelANOVA
Label:       Type III Model ANOVA
Template:    stat.GLM.Tests
Path:        GLM.Repeated.BetweenSubjects.ModelANOVA
-------------
```

All of the multivariate test results are routed to the HistWithin data set because all multivariate test tables are named MultStat, even though they occur in different directories in the output directory hierarchy. Only the between-subject ANOVA table appears in the HistBetween data set, even though there are also other tables named ModelANOVA. ODS selects just the one specific table for the HistBetween data set because of the partial name path (**BetweenSubjects.ModelANOVA**) in the second specification. For more information about names and qualified path names, see the discussion in the section "The ODS Statement" on page 599.

The following statements show the names and the variable labels for the two data sets and produce Output 23.7.1:

```
proc contents data=HistBetween varnum;
   ods select position;
run;

proc contents data=HistWithin varnum;
   ods select position;
run;
```

Output 23.7.1 Variable Names and Labels for the Two Data Sets

Histamine Study

The CONTENTS Procedure

Variables in Creation Order					
#	**Variable**	**Type**	**Len**	**Format**	**Label**
1	Dependent	Char	15		
2	HypothesisType	Num	8	BEST8.	
3	Source	Char	20		
4	DF	Num	8	BEST6.	
5	SS	Num	8	12.8	Type III SS
6	MS	Num	8	12.8	Mean Square
7	FValue	Num	8	7.2	F Value
8	ProbF	Num	8	PVALUE6.4	Pr > F

Output 23.7.1 *continued*

Histamine Study

The CONTENTS Procedure

Variables in Creation Order					
#	Variable	Type	Len	Format	Label
1	Hypothesis	Char	32		
2	Error	Char	55		
3	Statistic	Char	22		
4	Value	Num	8	12.8	
5	FValue	Num	8	7.2	F Value
6	NumDF	Num	8	BEST6.	Num DF
7	DenDF	Num	8	BEST6.	Den DF
8	ProbF	Num	8	PVALUE6.4	Pr > F
9	PValue	Num	8	PVALUE6.4	P-Value

The following statements create a new data set that contains the two data sets created in the preceding PROC GLM step and display the results in Output 23.7.2:

```
title2 'The Combined Data Set';

data temp1;
   set HistBetween HistWithin;
run;

proc print label;
run;
```

Output 23.7.2 Listing of the Combined Data Set: Histamine Study

Histamine Study
The Combined Data Set

Obs	Dependent	HypothesisType	Source	DF
1	BetweenSubjects	3	Drug	1
2	BetweenSubjects	3	Depleted	1
3	BetweenSubjects	3	Drug*Depleted	1
4	BetweenSubjects	3	Error	11
5		.		.
6		.		.
7		.		.
8		.		.
9		.		.
10		.		.
11		.		.
12		.		.
13		.		.
14		.		.
15		.		.
16		.		.
17		.		.
18		.		.
19		.		.
20		.		.

Output 23.7.2 *continued*

Histamine Study
The Combined Data Set

Obs	Type III SS	Mean Square	F Value	Pr > F	Hypothesis	Error
1	5.99336243	5.99336243	2.71	0.1281		
2	15.44840703	15.44840703	6.98	0.0229		
3	4.69087508	4.69087508	2.12	0.1734		
4	24.34683348	2.21334850	_	_		
5	.	.	24.03	0.0001	Time	Error SSCP Matrix
6	.	.	24.03	0.0001	Time	Error SSCP Matrix
7	.	.	24.03	0.0001	Time	Error SSCP Matrix
8	.	.	24.03	0.0001	Time	Error SSCP Matrix
9	.	.	5.78	0.0175	Time_Drug	Error SSCP Matrix
10	.	.	5.78	0.0175	Time_Drug	Error SSCP Matrix
11	.	.	5.78	0.0175	Time_Drug	Error SSCP Matrix
12	.	.	5.78	0.0175	Time_Drug	Error SSCP Matrix
13	.	.	21.31	0.0002	Time_Depleted	Error SSCP Matrix
14	.	.	21.31	0.0002	Time_Depleted	Error SSCP Matrix
15	.	.	21.31	0.0002	Time_Depleted	Error SSCP Matrix
16	.	.	21.31	0.0002	Time_Depleted	Error SSCP Matrix
17	.	.	12.48	0.0015	Time_Drug_Depleted	Error SSCP Matrix
18	.	.	12.48	0.0015	Time_Drug_Depleted	Error SSCP Matrix
19	.	.	12.48	0.0015	Time_Drug_Depleted	Error SSCP Matrix
20	.	.	12.48	0.0015	Time_Drug_Depleted	Error SSCP Matrix

Output 23.7.2 *continued*

Histamine Study
The Combined Data Set

Obs	Statistic	Value	Num DF	Den DF	P-Value
1		.	.	.	.
2		.	.	.	.
3		.	.	.	.
4		.	.	.	.
5	Wilks' Lambda	0.11097706	3	9	.
6	Pillai's Trace	0.88902294	3	9	.
7	Hotelling-Lawley Trace	8.01087137	3	9	.
8	Roy's Greatest Root	8.01087137	3	9	.
9	Wilks' Lambda	0.34155984	3	9	.
10	Pillai's Trace	0.65844016	3	9	.
11	Hotelling-Lawley Trace	1.92774470	3	9	.
12	Roy's Greatest Root	1.92774470	3	9	.
13	Wilks' Lambda	0.12339988	3	9	.
14	Pillai's Trace	0.87660012	3	9	.
15	Hotelling-Lawley Trace	7.10373567	3	9	.
16	Roy's Greatest Root	7.10373567	3	9	.
17	Wilks' Lambda	0.19383010	3	9	.
18	Pillai's Trace	0.80616990	3	9	.
19	Hotelling-Lawley Trace	4.15915732	3	9	.
20	Roy's Greatest Root	4.15915732	3	9	.

The next steps are designed to produce a more parsimonious display of the most important information in Output 23.7.2. The next step creates a data set named HistTests. Only the observations from the input data sets that are needed for interpretation are included. The variable Hypothesis in the HistWithin data set is renamed Source, and the NumDF variable is renamed DF. The renamed variables correspond to the variable names found in the HistBetween data set. These names are chosen since the template for the ModelANOVA table is used in subsequent steps. An explicit length for the new variable Source is provided since the input variables, Hypothesis and Source, have different lengths. The following statements produce Output 23.7.3:

```
data HistTests;
   length Source $ 20;
   set HistBetween(where =(Source    ^= 'Error'))
       HistWithin (rename=(Hypothesis =  Source NumDF=DF)
                   where =(Statistic  = 'Hotelling-Lawley Trace'));
run;

proc print label;
   title2 'Listing of the Combined Data Set';
run;
```

Output 23.7.3 Listing of the HistTests Data Set: Histamine Study

Histamine Study
Listing of the Combined Data Set

Obs	Source	Dependent	HypothesisType	Num DF	Type III SS
1	Drug	BetweenSubjects	3	1	5.99336243
2	Depleted	BetweenSubjects	3	1	15.44840703
3	Drug*Depleted	BetweenSubjects	3	1	4.69087508
4	Time		.	3	.
5	Time_Drug		.	3	.
6	Time_Depleted		.	3	.
7	Time_Drug_Depleted		.	3	.

Obs	Mean Square	F Value	Pr > F	Error	Statistic	Value	Den DF	P-Value
1	5.99336243	2.71	0.1281			.	.	.
2	15.44840703	6.98	0.0229			.	.	.
3	4.69087508	2.12	0.1734			.	.	.
4	.	24.03	0.0001	Error SSCP Matrix	Hotelling-Lawley Trace	8.01087137	9	.
5	.	5.78	0.0175	Error SSCP Matrix	Hotelling-Lawley Trace	1.92774470	9	.
6	.	21.31	0.0002	Error SSCP Matrix	Hotelling-Lawley Trace	7.10373567	9	.
7	.	12.48	0.0015	Error SSCP Matrix	Hotelling-Lawley Trace	4.15915732	9	.

The amount of information contained in the HistTests data set is appropriate for interpreting the analysis; however, there is still extra information, and the information of interest is not being displayed in a compact or useful form. This data set consists of multiple tables, an ANOVA table with between-subjects information, and multivariate statistics tables with the variables renamed to match the names in the ANOVA table. This form was chosen so that the data set could be displayed using PROC GLM's ANOVA template. A template specifies how the data set should be displayed and which columns should be displayed. The output from the ODS TRACE statements shows that the template associated with PROC GLM's ANOVA table is named **`Stat.GLM.Tests`**. You can use the **`Stat.GLM.Tests`** template to display the SAS data set HistTests, as follows:

```
title2 'Listing of the Selections, Using a Standard Template';
proc sgrender data=histtests template=Stat.GLM.Tests;
run;
```

The SGRENDER procedure displays the DATA= data set with the specified TEMPLATE= template. (You can use PROC SGRENDER to display both graphs and tables.) The results are displayed in Output 23.7.4.

Output 23.7.4 Listing of the Data Set Using a Standard PROC GLM ANOVA Template

Histamine Study
Listing of the Selections, Using a Standard Template

Source	DF	SS	Mean Square	F Value	Pr > F
Drug	1	5.99336243	5.99336243	2.71	0.1281
Depleted	1	15.44840703	15.44840703	6.98	0.0229
Drug*Depleted	1	4.69087508	4.69087508	2.12	0.1734
Time	3	.	.	24.03	0.0001
Time_Drug	3	.	.	5.78	0.0175
Time_Depleted	3	.	.	21.31	0.0002
Time_Drug_Depleted	3	.	.	12.48	0.0015

Alternatively, you could display the results by using a DATA step as follows:

```
title2 'Listing of the Selections, Using a Standard Template';

data _null_;
   set histtests;
   file print ods=(template='Stat.GLM.Tests');
   put _ods_;
run;
```

The next steps create a final display of these results, this time by using a custom template. This example shows you how to use PROC TEMPLATE to do the following:

- redefine the format for the SS and Mean Square columns
- include the table title and footnote in the body of the table
- translate the missing values for SS and Mean Square in the rows that correspond to multivariate tests to asterisks
- add a footnote to a table
- add a column that depicts the level of significance of each effect

The following statements create a custom template:

```
proc template;
   define table CombinedTests;
      parent=Stat.GLM.Tests;

      header '#Histamine Study##';
      footer '#* - Test computed using Hotelling-Lawley trace';

      column Source DF SS MS FValue ProbF Star;

      define Source; width=20; end;
      define DF; format=bestd3.; end;
      define SS;
         parent=Stat.GLM.SS
         choose_format=max format_width=7;
         translate _val_ = . into '  *';
```

```
      end;
      define MS;
         parent=Stat.GLM.MS
         choose_format=max format_width=7;
         translate _val_ = . into '  *';
      end;
      define Star;
         compute as ProbF;
         translate _val_ <= 0.001 into 'xxx',
                   _val_ <= 0.01  into 'xx',
                   _val_ <= 0.05  into 'x',
                   _val_ >  0.05  into '';
         pre_space=1 width=3 just=l;
      end;
   end;
run;
```

The CHOOSE_FORMAT=MAX option along with FORMAT_WIDTH=7 chooses the format for each column based on the maximum value and an overall width of 7. Alternatively, you could have specified a format directly by specifying, for example, FORMAT=7.2 or FORMAT=D8.3. The TRANSLATE statements provide values to display in place of the original values. The first two TRANSLATE statements display missing values as an asterisk with leading blanks added to ensure alignment with the decimal place. The third TRANSLATE statement displays *p*-values greater than 0.05 as a blank, values greater than 0.01 but less than or equal to 0.05 as a single `'x'`, and so on. The ProbF column is printed twice—once in the usual way as a numeric column with a PVALUE format and once with a column of blanks or x's. For detailed information about PROC TEMPLATE, see the section "The Template Procedure" in the *SAS Output Delivery System: User's Guide*. The following statements use the customized template to display the HistTests data set:

```
title2 'Listing of the Selections, Using a Customized Template';

proc sgrender data=HistTests template=CombinedTests;
run;
```

The results are displayed in Output 23.7.5.

Output 23.7.5 Display of the Data Sets Using a Customized Template: Histamine Study

Histamine Study
Listing of the Selections, Using a Customized Template

Histamine Study						
Source	Num DF	Sum of Squares	Mean Square	F Value	Pr > F	
Drug	1	5.9934	5.9934	2.71	0.1281	
Depleted	1	15.4484	15.4484	6.98	0.0229	x
Drug*Depleted	1	4.6909	4.6909	2.12	0.1734	
Time	3	*	*	24.03	0.0001	xxx
Time_Drug	3	*	*	5.78	0.0175	x
Time_Depleted	3	*	*	21.31	0.0002	xxx
Time_Drug_Depleted	3	*	*	12.48	0.0015	xx
* - Test computed using Hotelling-Lawley trace						

These next steps display the same table, but this time changing the background color for the entire row to

highlight effects with p-values less than 0.001 and also those with p-values less than 0.01. The table is displayed three times. Output 23.7.6 displays the results by using bold green and yellow backgrounds and a bold font. Output 23.7.7 displays the results by using subtler cyan and yellow backgrounds and a bold font. Output 23.7.8 displays the results by using very subtle cyan and gray backgrounds and a normal font. This control is provided by the CELLSTYLE statement in PROC TEMPLATE. You can do many things with the CELLSTYLE statement to enhance your output. Several more are shown in other examples in this chapter. These next steps create the custom template with varying colors and fonts and display the results by using PROC SGRENDER:

```
%macro hilight(c1,c2);
   proc template;
      define table CombinedTests;
         parent=Stat.GLM.Tests;

         header '#Histamine Study##';
         footer '#* - Test computed using Hotelling-Lawley trace';

         column Source DF SS MS FValue ProbF;

         cellstyle probf <= 0.001 as {background=&c1},
                   probf <= 0.01  as {background=&c2};

         define DF; format=bestd3.; end;
         define SS;
            parent=Stat.GLM.SS
            choose_format=max format_width=7;
            translate _val_ = . into '  *';
         end;
         define MS;
            parent=Stat.GLM.MS
            choose_format=max format_width=7;
            translate _val_ = . into '  *';
         end;
      end;
   run;

   proc sgrender data=HistTests template=CombinedTests;
   run;
%mend;

title2;
ods _all_ close;
ods html style=HTMLBlue;

%hilight(CX22FF22 fontweight=bold, CXFFFF22 fontweight=bold)
%hilight(CXAAFFFF fontweight=bold, CXFFFFDD fontweight=bold)
%hilight(CXEEFAFA, CXEEEEEE)

ods html close;
ods html;
ods pdf;
```

Output 23.7.6 Rows Boldly Highlighted: Histamine Study

Histamine Study

Histamine Study					
Source	Num DF	Sum of Squares	Mean Square	F Value	Pr > F
Drug	1	5.9934	5.9934	2.71	0.1281
Depleted	1	15.4484	15.4484	6.98	0.0229
Drug*Depleted	1	4.6909	4.6909	2.12	0.1734
Time	**3**	*****	*****	**24.03**	**0.0001**
Time_Drug	3	*	*	5.78	0.0175
Time_Depleted	**3**	*****	*****	**21.31**	**0.0002**
Time_Drug_Depleted	**3**	*****	*****	**12.48**	**0.0015**
* - Test computed using Hotelling-Lawley trace					

Output 23.7.7 Rows Subtly Highlighted: Histamine Study

Histamine Study

Histamine Study					
Source	Num DF	Sum of Squares	Mean Square	F Value	Pr > F
Drug	1	5.9934	5.9934	2.71	0.1281
Depleted	1	15.4484	15.4484	6.98	0.0229
Drug*Depleted	1	4.6909	4.6909	2.12	0.1734
Time	**3**	*****	*****	**24.03**	**0.0001**
Time_Drug	3	*	*	5.78	0.0175
Time_Depleted	**3**	*****	*****	**21.31**	**0.0002**
Time_Drug_Depleted	**3**	*****	*****	**12.48**	**0.0015**
* - Test computed using Hotelling-Lawley trace					

Output 23.7.8 Rows Very Subtly Highlighted: Histamine Study

Histamine Study

Histamine Study					
Source	Num DF	Sum of Squares	Mean Square	F Value	Pr > F
Drug	1	5.9934	5.9934	2.71	0.1281
Depleted	1	15.4484	15.4484	6.98	0.0229
Drug*Depleted	1	4.6909	4.6909	2.12	0.1734
Time	3	*	*	24.03	0.0001
Time_Drug	3	*	*	5.78	0.0175
Time_Depleted	3	*	*	21.31	0.0002
Time_Drug_Depleted	3	*	*	12.48	0.0015
* - Test computed using Hotelling-Lawley trace					

All colors are specified in values of the form CX*rrggbb*, where the last six characters specify RGB (red, green, blue) values on the hexadecimal scale of 00 to FF (or 0 to 255 base 10). You can run the following step to see the correspondence between the integer and HEX formatting of values in the range 0 to 255:

```
data _null_;
   do color = 0 to 255;
      put color 3. +1 color hex2.;
   end;
run;
```

The results of this step are not shown. Hexadecimal values 0 through F represent the numbers 0 to 15. A hex value xy can be converted to an integer as follows: $16x + y$. For example, BC is $16 \times 11 + 12 = 188$. Common colors are CXFF0000 (red), CX00FF00 (green), CX0000FF (blue), CXFFFF00 (yellow, a mix of red and green), CXFF00FF (magenta, a mix of red and blue), CX00FFFF (cyan, a mix of green and blue), CXFFFFFF (white, a mix of red, green, and blue), CX000000 (black, no color), CXDDDDDD (very light gray), CX222222 (very dark gray), and so on. Colors become lighter as the RGB values increase and darker as they decrease. For example, cyan (CX00FFFF) can be lightened by increasing the red component from 00 to FF until eventually it becomes indistinguishable from white. It can be darkened by jointly decreasing the green and blue values until it becomes indistinguishable from black.

The three CELLSTYLE statements that set the colors after the macro variables are substituted are as follows:

```
cellstyle probf <= 0.001 as {background=CX22FF22 fontweight=bold},
          probf <= 0.01  as {background=CXFFFF22 fontweight=bold};
cellstyle probf <= 0.001 as {background=CXAAFFFF fontweight=bold},
          probf <= 0.01  as {background=CXFFFFDD fontweight=bold};
cellstyle probf <= 0.001 as {background=CXEEFAFA},
          probf <= 0.01  as {background=CXEEEEEE};
```

The first color, CX22FF22, for the smallest p-values in the first table is a bold green color. The first table uses almost pure green and pure yellow, but a little red and blue are added to slightly lighten the colors. The second table uses a cyan and yellow that are very light due to the addition of AA (170) red and DD (221) blue, respectively. The third table uses a cyan that is not much different from light gray, and a light gray that is not much different from white.

Example 23.8: HTML Output with Hyperlinks between Tables

(View the complete code for this example (odsex8.sas) in the example repository.)

This example demonstrates how you can use ODS to provide links between different parts of your HTML procedure output. This example creates a table where each row contains a link to another table with more information about that row.

Suppose that you are analyzing a 4×4 factorial experiment for an industrial process, testing for differences in the number of defective products that are manufactured by different machines and use different sources of raw material. The data set Experiment is created as follows:

```
title 'Product Defects Experiment';

data Experiment;
   do Supplier = 'A', 'B', 'C', 'D';
      do Machine = 1 to 4;
         do rep = 1 to 5;
            input Defects @@;
            output;
```

```
         end;
      end;
   end;
   datalines;
 2  6  3  3  6  8  6  6  4  4  4  2  4  0  4  5  5  7  8  5
13 12 12 11 12 16 15 14 14 13 11 10 12 12 10 13 13 14 15 12
 2  6  3  6  6  6  4  4  6  6  0  3  2  0  2  4  6  7  6  4
20 19 18 21 22 22 24 23 20 20 17 19 18 16 17 23 20 20 22 21
;
```

Suppose that you are interested in fitting a model to determine the effect that the supplier of raw material and machine type have on the number of defects in the products. If the F test for a factor is significant, you might want to follow up with a multiple-comparison test for the levels of that factor. The tables of interest are the model ANOVA and the multiple-comparison output. Since this is a balanced experiment, the ANOVA procedure computes the appropriate analysis. The following statements produce these tables and Figure 23.8.1:

```
ods graphics off;
ods _all_ close;
ods html body='anovab.htm' style=HTMLBlue anchor='anova1';
ods trace output;

proc anova data=Experiment;
   ods select ModelANOVA MCLines;
   class Supplier Machine;
   model Defects = Supplier Machine;
   means Supplier Machine / tukey;
quit;

ods html close;
ods html;
ods pdf;
```

All destinations are first closed to avoid generating the output multiple times. ODS writes the HTML output to the file *anovab.htm*. The ANCHOR= option specifies *anova1* as the root name for the HTML anchor tags. This means that within the HTML document, the URL for the first table will be *anova1*, the URL for the second table will be *anova2*, and so on.

Output 23.8.1 ANOVA and Multiple-Comparison Results: Histamine Study

Product Defects Experiment

The ANOVA Procedure

Dependent Variable: Defects

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Supplier	3	3441.637500	1147.212500	580.72	<.0001
Machine	3	163.137500	54.379167	27.53	<.0001

Output 23.8.1 *continued*

Product Defects Experiment

The ANOVA Procedure

Tukey's Studentized Range (HSD) Test for Defects

Means with the same letter are not significantly different.			
Tukey Grouping	Mean	N	Supplier
A	20.1000	20	D
B	12.7000	20	B
C	4.6000	20	A
C			
C	4.1500	20	C

Product Defects Experiment

The ANOVA Procedure

Tukey's Studentized Range (HSD) Test for Defects

Means with the same letter are not significantly different.			
Tukey Grouping	Mean	N	Machine
A	11.7500	20	2
A			
A	11.5000	20	4
B	10.1500	20	1
C	8.1500	20	3

The ODS trace output (not shown) shows that PROC ANOVA uses the **Stat.GLM.Tests** template to format the ANOVA table. The following statements demonstrate how you can link a row of the ANOVA table to the corresponding multiple-comparison table by modifying the table template, using the original values and the URLs for the second and third tables (*anova2* and *anova3*):

```
proc template;
   edit Stat.GLM.Tests;
      edit Source;
         cellstyle _val_ = 'Supplier' as {url="#ANOVA2"},
                   _val_ = 'Machine'  as {url="#ANOVA3"};
      end;
   end;
run;
```

This template uses the CELLSTYLE statement to alter the values in the **Source** column (**'Supplier'** and **'Machine'**) of the ANOVA tests table. The values of **'Supplier'** and **'Machine'** are displayed as hyperlinks in the HTML, and clicking them takes you to the links *anova2* and *anova3*, which are the

multiple-comparison tables.

You can find the value to use in the URL by viewing the HTML source file, *anovab.htm*. You can either open the HTML file in a text editor or view it in a browser window and select **View ► Source**. Search for `'<a name='` to find the URL names. The first table is *anova1*, the second is *anova2*, the third is *anova3*, and so on. If the ANCHOR= option had not been used in the ODS HTML statement, the names would have been *IDX*, *IDX1*, *IDX2*, and so on. If you do not use the ODS SELECT statement or if you do anything to change the tables that are produced, the names will be different. The statements create the *Supplier* label as a link that enables you to open the table of means from the "Tukey's Studentized Range Test for Defects" associated with the Supplier variable. Similarly, *Machine* provides a link to the table of means from the "Tukey's Studentized Range Test for Defects" associated with the Machine variable.

Next, the analysis is run again, this time using the modified template. The following statements produce the results:

```
ods _all_ close;
ods html body='anovab.htm' style=HTMLBlue anchor='anova1';

proc anova data=Experiment;
   ods select ModelANOVA MCLines;
   class Supplier Machine;
   model Defects = Supplier Machine;
   means Supplier Machine / tukey;
quit;

ods html close;
ods html;
ods pdf;
```

The ANOVA table is displayed in Output 23.8.2.

Output 23.8.2 HTML Output from PROC ANOVA: Linked Output

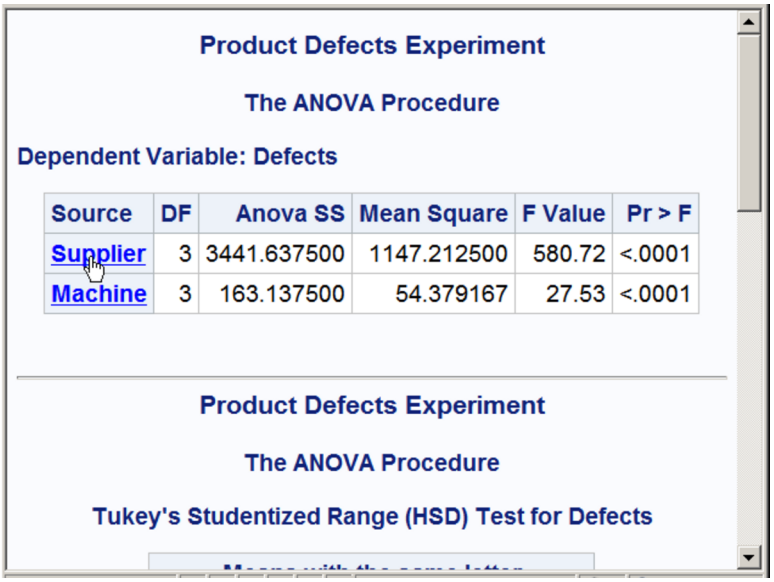

Product Defects Experiment

The ANOVA Procedure

Dependent Variable: Defects

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Supplier	3	3441.637500	1147.212500	580.72	<.0001
Machine	3	163.137500	54.379167	27.53	<.0001

Product Defects Experiment

The ANOVA Procedure

Tukey's Studentized Range (HSD) Test for Defects

The underlined text displayed in Output 23.8.2 shows the links, *Supplier* and *Machine*, that you created with

the modified template. When you click a link, the appropriate multiple-comparison table opens in your browser. Output 23.8.3 shows the table from the *Supplier* link.

Output 23.8.3 Linked Output: Multiple-Comparison Table from PROC ANOVA

Product Defects Experiment

The ANOVA Procedure

Tukey's Studentized Range (HSD) Test for Defects

Means with the same letter are not significantly different.			
Tukey Grouping	**Mean**	**N**	**Supplier**
A	20.1000	20	D
B	12.7000	20	B
C	4.6000	20	A
C			
C	4.1500	20	C

When you run the PROC TEMPLATE step shown previously, the following note is printed in the SAS log:

```
NOTE: TABLE 'Stat.GLM.Tests' has been saved to: SASUSER.TEMPLAT
```

You can see that there is a version of the template in Sasuser by running the following statements:

```
proc template;
   list Stat.GLM.Tests;
run;
```

These statements produce Output 23.8.4.

Output 23.8.4 Templates

Product Defects Experiment

Listing of: SASUSER.TEMPLAT
Path Filter is: Stat.GLM.Tests
Sort by: PATH/ASCENDING

Obs	Path	Type
1	Stat.GLM.Tests	Table

Listing of: SASHELP.TMPLSTAT
Path Filter is: Stat.GLM.Tests
Sort by: PATH/ASCENDING

Obs	Path	Type
1	Stat.GLM.Tests	Table

You can delete your custom template and restore the default template as follows:

```
proc template;
   delete Stat.GLM.Tests / store=sasuser.templat;
run;
```

The following note is printed in the SAS log:

```
NOTE: 'Stat.GLM.Tests' has been deleted from: SASUSER.TEMPLAT
```

Example 23.9: HTML Output with Graphics and Hyperlinks

(View the complete code for this example (odsex9.sas) in the example repository.)

This example demonstrates how you can use ODS to create links between each bar in a bar chart (Output 23.9.1) and other parts of the analysis (Output 23.9.2). The data in this example are selected from a larger experiment on the use of drugs in the treatment of leprosy (Snedecor and Cochran 1967, p. 422). Variables in the study are as follows:

Drug	two antibiotics (‘a’ and ‘d’) and a control (‘f’)
PreTreatment	a pretreatment score of leprosy bacilli
PostTreatment	a posttreatment score of leprosy bacilli

The data set is created as follows:

```
title 'Treatment of Leprosy';

data drugtest;
   input Drug $ PreTreatment PostTreatment @@;
   datalines;
a 11  6  a  8  0  a  5  2  a 14  8  a 19 11
a  6  4  a 10 13  a  6  1  a 11  8  a  3  0
d  6  0  d  6  2  d  7  3  d  8  1  d 18 18
d  8  4  d 19 14  d  8  9  d  5  1  d 15  9
f 16 13  f 13 10  f 11 18  f  9  5  f 21 23
f 16 12  f 12  5  f 12 16  f  7  1  f 12 20
;
```

The following statement opens the HTML destination:

```
ods _all_ close;
ods html body='glmb.htm' contents='glmc.htm' frame='glmf.htm' style=HTMLBlue;
```

The ODS HTML statement specifies the body file name, generates a table of contents for the output, and generates a frame to contain the body and table of contents. The following statements perform the analysis:

```
proc glm data=drugtest;
   class drug;
   model PostTreatment = drug | PreTreatment / solution;
   lsmeans drug / stderr pdiff;
   ods output LSMeans=lsmeans;
quit;
```

The ODS OUTPUT statement writes the table of LS-means to the data set named lsmeans. PROC GLM performs an analysis of covariance and computes LS-means for the variable Drug.

The following steps demonstrate how you can create links to connect the results of different analyses. In this example, the table of LS-means is graphically summarized in a horizontal bar chart. Each bar is linked to a plot that displays the relationship between the PostTreatment response variable and the PreTreatment variable for the drug that corresponds to the bar.

NOTE: PROC GLM can use ODS Graphics to create LS-means graphs that are different from the one constructed here. You do not have to run the following steps to get PROC GLM's standard LS-means plots.

The following DATA step creates a new variable named DrugClick that matches each drug value with an HTML file:

```
data lsmeans;
   set lsmeans;
   if drug='a' then DrugClick='drug1.htm';
   if drug='d' then DrugClick='drug2.htm';
   if drug='f' then DrugClick='drug3.htm';
run;
```

The variable DrugClick is used in the chart. The variable provides the connection information for linking the two parts of the analysis together. The files referred to in these statements are created in a later step. The following statements create the chart:

```
ods graphics / imagemap=yes height=2in width=6.4in;

proc sgplot data=lsmeans;
   title 'Chart of LS-Means for Drug Type';
   hbar drug / response=lsmean stat=mean
               url=drugclick;
   footnote j=l 'Click on the bar to see a plot of PostTreatment '
                'versus PreTreatment for the corresponding drug.';
   format lsmean 6.3;
run;

ods graphics off;
footnote;
ods html close;
```

The chart is created with the ODS Graphics procedure SGPLOT. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." The ODS GRAPHICS statement is not required before you run SG procedures. However, in this case, it is necessary to specify IMAGEMAP=YES so that the URL= option works properly. The size of the graph is also specified with the HEIGHT= and WIDTH= options. PROC SGPLOT is used, and the HBAR statement requests a horizontal bar chart for the variable Drug. The lengths of the bars represent the values of the LSMean variable. The URL= option specifies the variable DrugClick as the HTML linking variable. The FOOTNOTE statement provides text that indicates how to use the links in the graph.

The following statements provide the second analysis. The three files referred to by the DrugClick variable are created as follows:

```
ods html body='drug1.htm' newfile=page style=HTMLBlue;

proc sgplot data=drugtest;
   title 'Plot of PostTreatment versus PreTreatment';
   scatter y=PostTreatment x=PreTreatment;
   by drug notsorted;
run;
ods html close;
```

The NEWFILE= option in the ODS HTML statement creates a new HTML file for each page of output. (Page breaks occur only when a procedure explicitly starts a new page.) The NEWFILE= option also increments the file name numeric suffix for each new HTML file created, with the first file name corresponding to that given in the BODY= option, *drug1.htm*.

PROC SGPLOT is used, producing a plot of the variable PostTreatment versus the variable PreTreatment for each value of the Drug variable. Three plots are created, and each plot is contained in a separate HTML file. The files are named *drug1.htm*, *drug2.htm*, and *drug3.htm*. The file names match those file names specified as values of the DrugClick variable. By default, the HTML files are generated in your current working directory. You can instead specify a path, such as **`frame='html/drug2.htm'`**, to put a file in a subdirectory. The chart in Output 23.9.1 displays the difference in LS-means for each drug type. When you click on a bar that represents a value of the variable Drug, the browser opens the plot of PostTreatment versus PostTreatment variables that corresponds to that value of the variable Drug. Output 23.9.2 displays the plots for each drug type.

Output 23.9.1 Bar Chart of LS-Means by Drug Type with Links to Plots

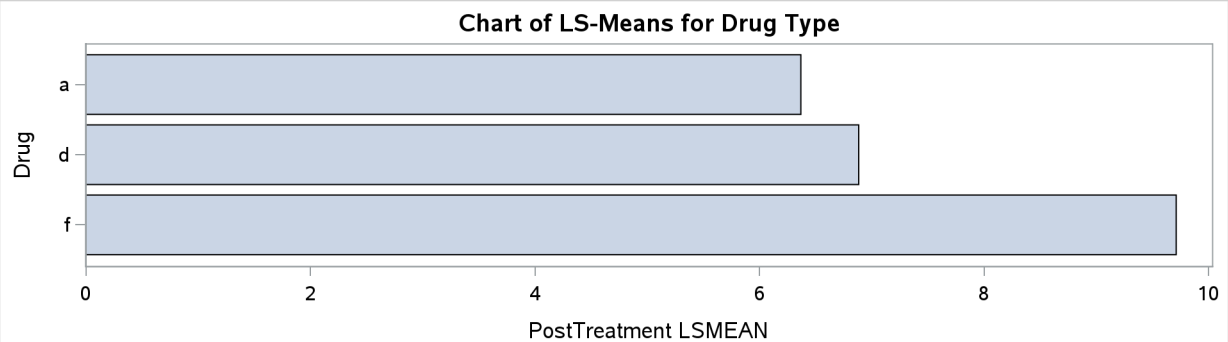

Click on the bar to see a plot of PostTreatment versus PreTreatment for the corresponding drug.

Output 23.9.2 Plots by Drug Type

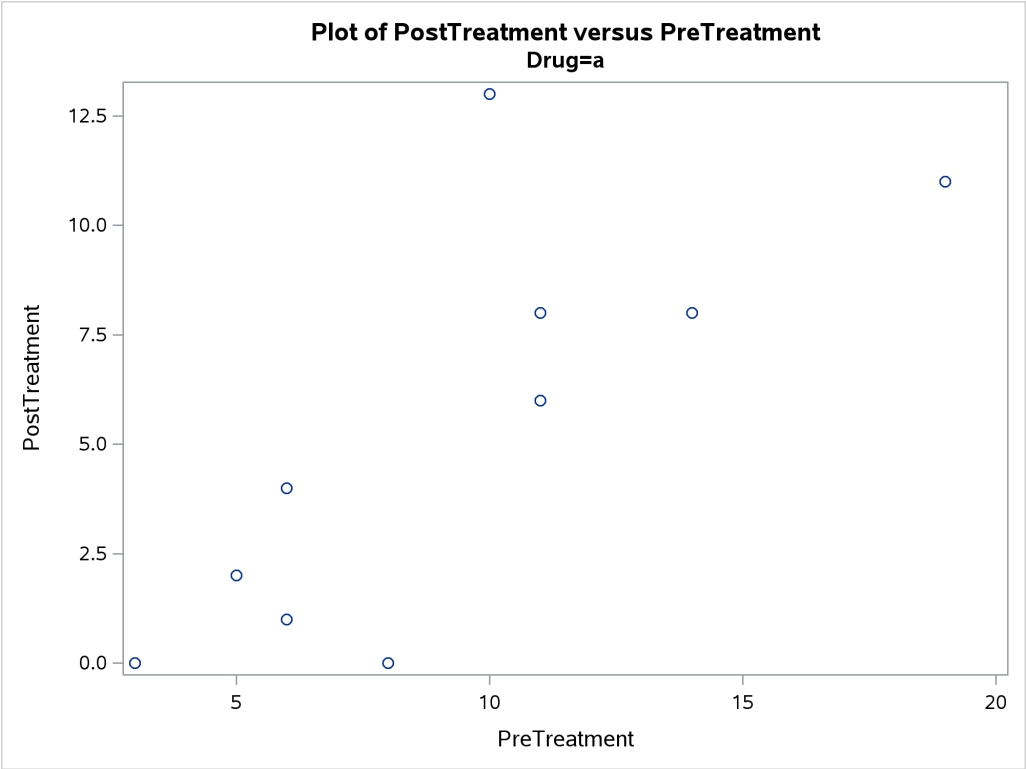

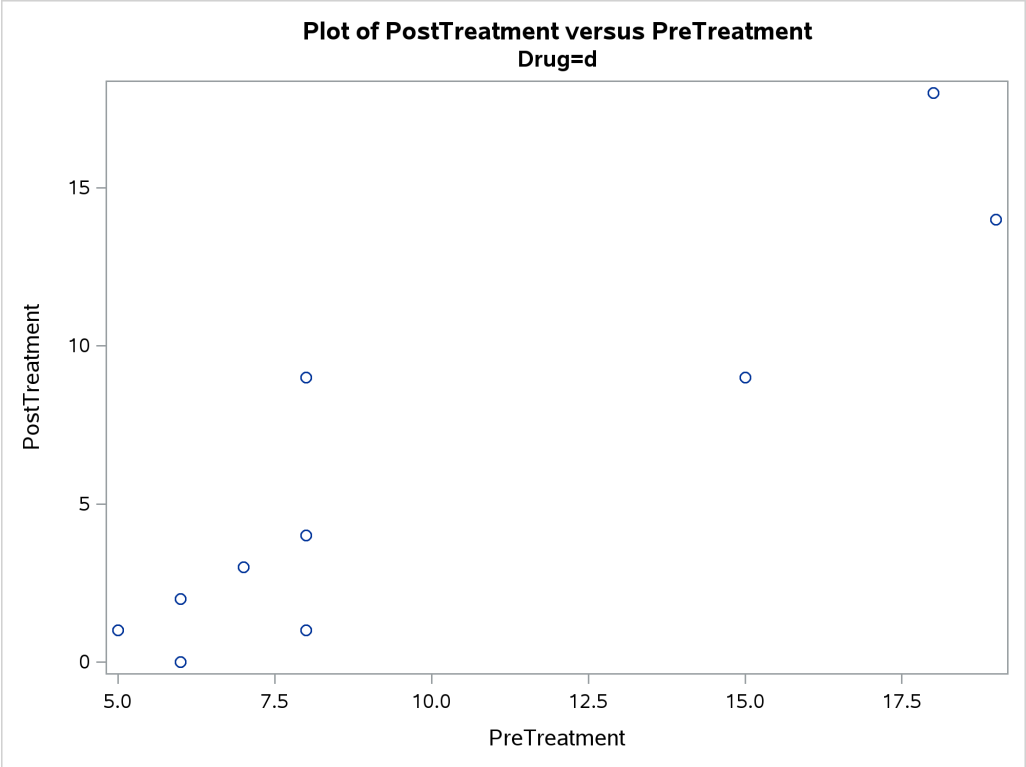

Output 23.9.2 *continued*

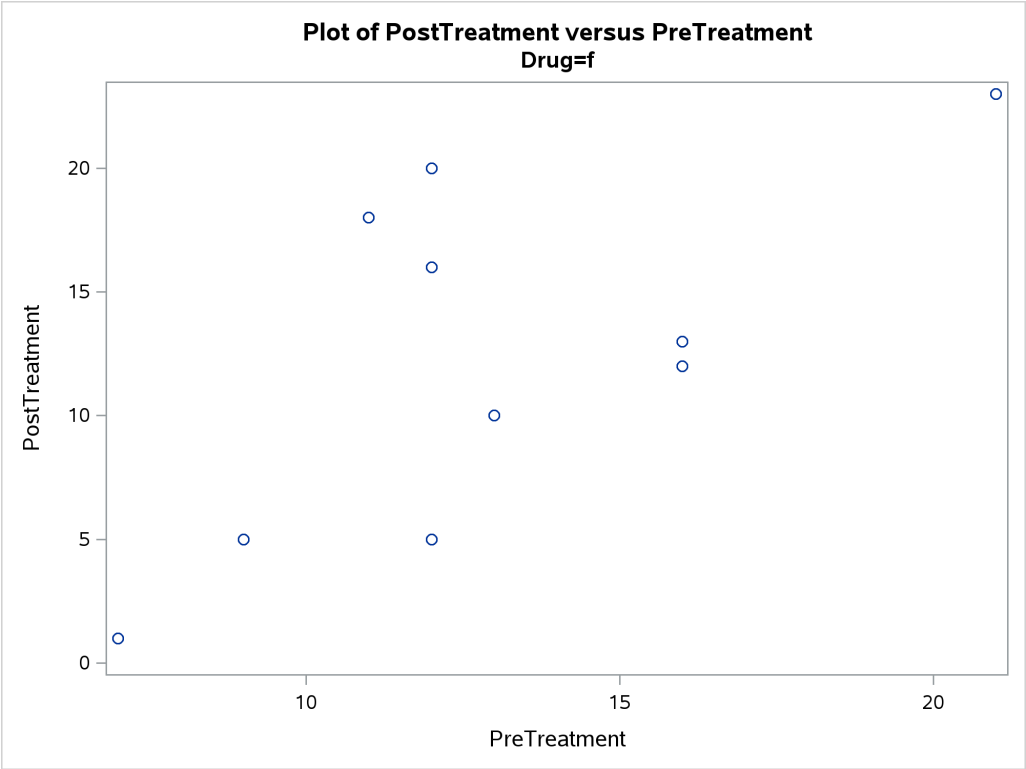

Example 23.10: Correlation and Covariance Matrices

(View the complete code for this example (odsex10.sas) in the example repository.)

This example demonstrates how you can use ODS to set the background color of individual cells in a table. The color is set to reflect the magnitude of the value in the cell. You can use color to call attention to larger values and to see the pattern in the data in a way that is hard to visualize just by looking at the numbers. This is illustrated with correlation and covariance matrices. The data for this first part of this example are ratings of automobiles. The following statements create the data set:

```
title 'Rating of Automobiles';

data cars;
   input Origin $ 1-8 Make $ 10-19 Model $ 21-36
         (MPG Reliability Acceleration Braking Handling Ride
          Visibility Comfort Quiet Cargo) (1.);
   datalines;
GMC       Buick      Century         3334444544
GMC       Buick      Electra         2434453555
GMC       Buick      Lesabre         2354353545

   ... more lines ...

GMC       Pontiac    Sunbird         3134533234
```

```
;
```

The following steps edit the template that PROC CORR uses to display the correlation matrix. The CELLSTYLE statement sets the background color to light gray for correlations equal to 1 or –1. Values less than –0.75 or greater than 0.75 are set to red. Values less than –0.50 or greater than 0.50 are set to blue. Values less than –0.25 or greater than 0.25 are set to cyan. Values in the range –0.25 to 0.25 are set to white. PROC CORR is then run using the custom template. Finally, the default template is restored. The following statements produce Output 23.10.1:

```
proc template;
   edit Base.Corr.StackedMatrix;
      column (RowName RowLabel) (Matrix) * (Matrix2);
      edit matrix;
         cellstyle _val_  = -1.00 as {backgroundcolor=CXEEEEEE},
                   _val_ <= -0.75 as {backgroundcolor=red},
                   _val_ <= -0.50 as {backgroundcolor=blue},
                   _val_ <= -0.25 as {backgroundcolor=cyan},
                   _val_ <=  0.25 as {backgroundcolor=white},
                   _val_ <=  0.50 as {backgroundcolor=cyan},
                   _val_ <=  0.75 as {backgroundcolor=blue},
                   _val_ <   1.00 as {backgroundcolor=red},
                   _val_  =  1.00 as {backgroundcolor=CXEEEEEE};
      end;
   end;
run;

ods _all_ close;
ods html body='corr.html' style=HTMLBlue;

proc corr data=cars noprob;
   ods select PearsonCorr;
run;

ods html close;
ods html;
ods pdf;

proc template;
   delete Base.Corr.StackedMatrix / store=sasuser.templat;
run;
```

Output 23.10.1 Correlation Matrix from PROC CORR

Rating of Automobiles

The CORR Procedure

Pearson Correlation Coefficients, N = 50										
	MPG	Reliability	Acceleration	Braking	Handling	Ride	Visibility	Comfort	Quiet	Cargo
MPG	1.00000	0.22003	-0.19454	0.41475	0.25594	-0.23705	0.67924	-0.06567	-0.49128	-0.03075
Reliability	0.22003	1.00000	-0.08512	0.25881	-0.09443	0.27406	0.33356	0.36607	0.45302	0.35261
Acceleration	-0.19454	-0.08512	1.00000	0.06688	0.07119	0.33888	-0.13280	0.06369	0.00934	-0.12112
Braking	0.41475	0.25881	0.06688	1.00000	0.22335	0.30309	0.44938	0.26165	0.00164	0.20880
Handling	0.25594	-0.09443	0.07119	0.22335	1.00000	0.12435	0.12599	-0.07516	-0.02418	-0.14274
Ride	-0.23705	0.27406	0.33888	0.30309	0.12435	1.00000	0.16114	0.75173	0.48498	0.39108
Visibility	0.67924	0.33356	-0.13280	0.44938	0.12599	0.16114	1.00000	0.29830	-0.18347	0.35585
Comfort	-0.06567	0.36607	0.06369	0.26165	-0.07516	0.75173	0.29830	1.00000	0.44917	0.53836
Quiet	-0.49128	0.45302	0.00934	0.00164	-0.02418	0.48498	-0.18347	0.44917	1.00000	0.33846
Cargo	-0.03075	0.35261	-0.12112	0.20880	-0.14274	0.39108	0.35585	0.53836	0.33846	1.00000

The preceding statements used a small number of discrete colors to show the range of values. In contrast, the following statements use a color gradient. The SAS autocall macro %Paint is available for generating the CELLSTYLE colors list with a list of interpolated colors. If your site has installed the autocall libraries supplied by the SAS System and uses the standard configuration of software supplied by the SAS System, you need to ensure that the SAS System option MAUTOSOURCE is in effect before you begin using autocall macros. The macros do not have to be included (for example, with a %INCLUDE statement). They can be called directly once they are properly installed. For more information about autocall libraries, see *SAS Macro Language: Reference*.

Usually, you can use the %Paint macro by specifying a list of values and a list of colors. Here is an example for values that range from 0 to 10:

```
%paint(values=0 to 10 by 0.5,
       colors=white cyan blue magenta red)

proc print data=colors;
run;
```

The %Paint macro prints the following information to the SAS log:

```
Legend:
        0 = White
      2.5 = Cyan
        5 = Blue
      7.5 = Magenta
       10 = Red
```

A value of 0 maps to white, a value of 2.5 maps to cyan, values in the range 0 to 2.5 map to colors in the range from white to cyan, and so on. The %Paint macro for this step creates an output data set, Colors, which is shown in Output 23.10.2.

Output 23.10.2 Color Interpolation

Rating of Automobiles

Obs	Start	_RGB_
1	0.0	CXFFFFFF
2	0.5	CXCBFFFF
3	1.0	CX97FFFF
4	1.5	CX63FFFF
5	2.0	CX2FFFFF
6	2.5	CX05FFFF
7	3.0	CX00D1FF
8	3.5	CX009CFF
9	4.0	CX0068FF
10	4.5	CX0034FF
11	5.0	CX0000FF
12	5.5	CX3400FF
13	6.0	CX6800FF
14	6.5	CX9C00FF
15	7.0	CXD100FF
16	7.5	CXFA00FF
17	8.0	CXFF00D1
18	8.5	CXFF009C
19	9.0	CXFF0068
20	9.5	CXFF0034
21	10.0	CXFF0000

This shows the color interpolation for a series of points. You could use a smaller BY value in the %Paint macro to get more points along the color gradient. However, a few dozen colors are usually sufficient for most purposes.

The following steps use the %Paint macro to create a color gradient for a correlation matrix, edit the template, display the results, and restore the default template:

```
%paint(values=-1 to 1 by 0.05, macro=setstyle,
       colors=CXEEEEEE red magenta blue cyan white
              cyan blue magenta red CXEEEEEE
              -1 -0.99 -0.75 -0.5 -0.25 0 0.25 0.5 0.75 0.99 1)

proc template;
   edit Base.Corr.StackedMatrix;
      column (RowName RowLabel) (Matrix) * (Matrix2);
      edit matrix;
         %setstyle(backgroundcolor)
      end;
   end;
run;

ods _all_ close;
ods html body='corr.html' style=HTMLBlue;
proc corr data=cars noprob;
   ods select PearsonCorr;
```

```
run;
ods html close;
ods html;
ods pdf;

proc template;
   delete Base.Corr.StackedMatrix / store=sasuser.templat;
run;
```

The VALUES= option creates a range of values from –1 to 1 with an increment of 0.05. The %Paint macro generates a CELLSTYLE **_val_ <=** *value* **as {backgroundcolor=** *color***},** line for each value in the list. Specifically, it generates a macro named SETSTYLE (from the MACRO= option) that contains the entire CELLSTYLE statement for use in PROC TEMPLATE. The argument to the macro is the option that you want to set. In this case, it is the background color. You could specify **foreground** instead to set the color of the numbers themselves. The first part of the generated statement is as follows:

```
cellstyle _val_<=-1 as {backgroundcolor=CXEFEEEE},
          _val_<=-0.95 as {backgroundcolor=CXFF0020},
          _val_<=-0.9 as {backgroundcolor=CXFF0062},
          _val_<=-0.85 as {backgroundcolor=CXFF008D},
          _val_<=-0.8 as {backgroundcolor=CXFF00CF},
```

The color mapping for a correlation matrix can be a bit more involved than it is for most tables. This is because you might want the maximum correlations, 1 and –1, to be displayed using colors outside the gradient that is used for other values. Usually, you specify the color list, and the %Paint macro maps the first color to the minimum value, the last color to the maximum value, and colors in between using equal increments and values based on the minimum and maximum. Alternatively, you can provide these values, as shown in this example. The legend, displayed in the SAS log, is as follows for the %Paint macro step:

```
Legend:
      -1 = CXEEEEEE
   -0.99 = Red
   -0.75 = Magenta
    -0.5 = Blue
   -0.25 = Cyan
       0 = White
    0.25 = Cyan
     0.5 = Blue
    0.75 = Magenta
    0.99 = Red
       1 = CXEEEEEE
```

Values in the range –0.99 to 0.99 follow the interpolation red to magenta to blue to cyan to white to cyan to blue to magenta to red. Of course, the actual correlations for these data do not span this entire range, so a pure red background does not appear in the matrix. Correlations of 1 and –1 are displayed as light gray. The resulting correlation matrix is displayed in Output 23.10.3. Notice that there are now a number of shades of colors, particularly shades of blues, not just a few discrete colors. The largest values are displayed in shades of purple and magenta.

Output 23.10.3 Correlation Matrix from PROC CORR with a Color Gradient

Rating of Automobiles

The CORR Procedure

Pearson Correlation Coefficients, N = 50

	MPG	Reliability	Acceleration	Braking	Handling	Ride	Visibility	Comfort	Quiet	Cargo
MPG	1.00000	0.22003	-0.19454	0.41475	0.25594	-0.23705	0.67924	-0.06567	-0.49128	-0.03075
Reliability	0.22003	1.00000	-0.08512	0.25881	-0.09443	0.27406	0.33356	0.36607	0.45302	0.35261
Acceleration	-0.19454	-0.08512	1.00000	0.06688	0.07119	0.33888	-0.13280	0.06369	0.00934	-0.12112
Braking	0.41475	0.25881	0.06688	1.00000	0.22335	0.30309	0.44938	0.26165	0.00164	0.20880
Handling	0.25594	-0.09443	0.07119	0.22335	1.00000	0.12435	0.12599	-0.07516	-0.02418	-0.14274
Ride	-0.23705	0.27406	0.33888	0.30309	0.12435	1.00000	0.16114	0.75173	0.48498	0.39108
Visibility	0.67924	0.33356	-0.13280	0.44938	0.12599	0.16114	1.00000	0.29830	-0.18347	0.35585
Comfort	-0.06567	0.36607	0.06369	0.26165	-0.07516	0.75173	0.29830	1.00000	0.44917	0.53836
Quiet	-0.49128	0.45302	0.00934	0.00164	-0.02418	0.48498	-0.18347	0.44917	1.00000	0.33846
Cargo	-0.03075	0.35261	-0.12112	0.20880	-0.14274	0.39108	0.35585	0.53836	0.33846	1.00000

Next, the same technique is used to display the covariance and correlation matrices of a heteroscedastic autoregressive model. The data are based on the famous growth measurement data of Pothoff and Roy (1964), but are modified here to illustrate the technique of painting the entries of a matrix. The data consist of four repeated growth measurements of 11 girls and 16 boys. The measurements from two adjacent children in the original data were combined and rearranged here to emulate a repeated measures sequence with eight observations. The following statements create the data set:

```
title 'Analysis of Repeated Growth Measures';

data pr;
   input Person Gender $ y1 y2 y3 y4 y5 y6 y7 y8;
   array y{8};
   do time=5,7,8,4,3,2,1;
      Response = y{time};
      Age      = time+7;
      output;
   end;
   datalines;
 1   F   21.0  20.0  21.5  23.0  21.0  21.5  24.0  25.5
 2   F   20.5  24.0  24.5  26.0  23.5  24.5  25.0  26.5
 3   F   21.5  23.0  22.5  23.5  20.0  21.0  21.0  22.5
 4   F   21.5  22.5  23.0  25.0  23.0  23.0  23.5  24.0
 5   F   20.0  21.0  22.0  21.5  16.5  19.0  19.0  19.5
 6   F   24.5  25.0  28.0  28.0  26.0  25.0  29.0  31.0
 7   M   21.5  22.5  23.0  26.5  23.0  22.5  24.0  27.5
 8   M   25.5  27.5  26.5  27.0  20.0  23.5  22.5  26.0
 9   M   24.5  25.5  27.0  28.5  22.0  22.0  24.5  26.5
10   M   24.0  21.5  24.5  25.5  23.0  20.5  31.0  26.0
11   M   27.5  28.0  31.0  31.5  23.0  23.0  23.5  25.0
12   M   21.5  23.5  24.0  28.0  17.0  24.5  26.0  29.5
13   M   22.5  25.5  25.5  26.0  23.0  24.5  26.0  30.0
;
```

The following statements create a macro that sets colors for the covariance matrix (SETSTYLE1), create a macro that sets colors for the correlation matrix (SETSTYLE2), edit the templates, run the analysis with PROC GLIMMIX, and restore the default templates:

```
* You need to run the analysis once to know that 20 is a good maximum;
%paint(values=0 to 20 by 0.25,
       colors=cyan blue magenta red, macro=setstyle1)

%paint(values=0 to 1 by 0.05,
       colors=cyan blue magenta red, macro=setstyle2)

proc template;
   edit Stat.Glimmix.V;
      column Subject Index Row Col;
      edit Col;
         %setstyle1(backgroundcolor)
      end;
   end;
   edit Stat.Glimmix.VCorr;
      column Subject Index Row Col;
      edit Col;
         %setstyle2(backgroundcolor)
      end;
   end;
run;

ods _all_ close;
ods html body='ar1.html' style=HTMLBlue;
proc glimmix data=pr;
   class person gender time;
   model response = gender age gender*age;
   random _residual_ / sub=person type=arh(1) v residual vcorr;
   ods select v vcorr;
run;
ods html close;
ods html;
ods pdf;

proc template;
   delete Stat.Glimmix.V / store=sasuser.templat;
   delete Stat.Glimmix.VCorr / store=sasuser.templat;
run;
```

The results are displayed in Output 23.10.4 and Output 23.10.5. Both the covariance and correlation matrices have a structure that is more obvious when colors are added to the display. In particular, the colors clearly show the banded structure of the correlation matrix.

Output 23.10.4 Heteroscedastic AR(1) Covariance Matrix

Analysis of Repeated Growth Measures

The GLIMMIX Procedure

Estimated V Matrix for Person 1

Row	Col1	Col2	Col3	Col4	Col5	Col6	Col7
1	19.1973	10.5505	7.3707	4.5756	2.4983	1.4814	0.9697
2	10.5505	11.8104	8.2508	5.1220	2.7966	1.6582	1.0855
3	7.3707	8.2508	11.7407	7.2885	3.9795	2.3596	1.5446
4	4.5756	5.1220	7.2885	9.2159	5.0318	2.9836	1.9530
5	2.4983	2.7966	3.9795	5.0318	5.5959	3.3181	2.1720
6	1.4814	1.6582	2.3596	2.9836	3.3181	4.0075	2.6232
7	0.9697	1.0855	1.5446	1.9530	2.1720	2.6232	3.4976

Output 23.10.5 Heteroscedastic AR(1) Correlation Matrix

Estimated V Correlation Matrix for Person 1

Row	Col1	Col2	Col3	Col4	Col5	Col6	Col7
1	1.0000	0.7007	0.4910	0.3440	0.2410	0.1689	0.1183
2	0.7007	1.0000	0.7007	0.4910	0.3440	0.2410	0.1689
3	0.4910	0.7007	1.0000	0.7007	0.4910	0.3440	0.2410
4	0.3440	0.4910	0.7007	1.0000	0.7007	0.4910	0.3440
5	0.2410	0.3440	0.4910	0.7007	1.0000	0.7007	0.4910
6	0.1689	0.2410	0.3440	0.4910	0.7007	1.0000	0.7007
7	0.1183	0.1689	0.2410	0.3440	0.4910	0.7007	1.0000

Alternatively, you could just use the %Paint macro to do the color interpolation and use its output data set to create other types of style effects. The following statements show one way to set the font to bold and set the foreground color based on the values of the covariances:

```
%let inc = 0.25;

%paint(values=0 to 20 by &inc, colors=blue magenta red)

data cntlin;
   set colors;
   fmtname = 'paintfmt';
   label = _rgb_;
   end = start + &inc;
   keep start end label fmtname;
run;

proc format cntlin=cntlin;
```

```
run;

proc template;
   edit Stat.Glimmix.V;
      column Subject Index Row Col;
      edit Col;
         style = {foreground=paintfmt8. font_weight=bold};
      end;
   end;
run;

ods _all_ close;
ods html body='ar1.html' style=HTMLBlue;
proc glimmix data=pr;
   class person gender time;
   model response = gender age gender*age;
   random _residual_ / sub=person type=arh(1) v residual;
   ods select v;
run;
ods html close;
ods html;
ods pdf;

proc template;
   delete Stat.Glimmix.V / store=sasuser.templat;
run;
title;
```

The %Paint macro creates the SAS data set Colors with the result of the interpolation. This data set can be processed to create a format. The DATA step creates a range of values from Start to End and assigns a color to Label based on the color computed by the %Paint macro. This data set is input to PROC FORMAT to create the format PAINTFMT. PROC TEMPLATE uses this format to set the color of the values in the table. The cell value is evaluated using the specified FOREGROUND= format for every cell in the table, and the appropriate color is assigned. PROC GLIMMIX does the analysis, and the results are displayed in Output 23.10.6.

Output 23.10.6 Heteroscedastic AR(1) Covariance Matrix

Analysis of Repeated Growth Measures

The GLIMMIX Procedure

Estimated V Matrix for Person 1							
Row	Col1	Col2	Col3	Col4	Col5	Col6	Col7
1	19.1973	10.5505	7.3707	4.5756	2.4983	1.4814	0.9697
2	10.5505	11.8104	8.2508	5.1220	2.7966	1.6582	1.0855
3	7.3707	8.2508	11.7407	7.2885	3.9795	2.3596	1.5446
4	4.5756	5.1220	7.2885	9.2159	5.0318	2.9836	1.9530
5	2.4983	2.7966	3.9795	5.0318	5.5959	3.3181	2.1720
6	1.4814	1.6582	2.3596	2.9836	3.3181	4.0075	2.6232
7	0.9697	1.0855	1.5446	1.9530	2.1720	2.6232	3.4976

Many other effects could be achieved by using this approach and different options in the STYLE= specification.

The following steps print the lower triangle of a correlation matrix:

```
ods select none;
proc corr data=sashelp.cars noprob;
   ods output PearsonCorr=p;
run;
ods select all;

data p2;
   set p end=eof;
   array __n[*] _numeric_;
   do __i = _n_ to dim(__n); __n[__i] = ._; end;
   if _n_ = 1 then do;
      call execute('data _null_; set p2;');
      call execute('file print ods=(template="Base.Corr.StackedMatrix"');
      call execute('columns=(rowname=variable');
   end;
   call execute(cats('matrix=',vname(__n[_n_]),'(generic)'));
   if eof then call execute('')); put _ods_; run;');
run;
```

The PROC CORR step suppresses all displayed output and creates an ODS output data set that contains the correlation matrix. The DATA step does two things: it modifies the correlation matrix so that all values on or above the diagonal are set to an underscore missing value, and it generates a second DATA step that contains ad hoc rendering code that displays the modified matrix. The template has a statement that translates the underscore missing values into blanks. The rendering code specifies the mapping between the template column name Rowname and the data set variable called Variable. This variable provides the row labels. Because the columns of a correlation matrix cannot be known until the procedure runs, the columns are designated as GENERIC in the template column definition. The ODS template has a single placeholder column named Matrix for each correlation matrix column. The rendering code declares the mappings between

the template generic column and the variables in the data set. In this example, the DATA P2 step uses CALL EXECUTE statements to generate the following DATA _NULL_ step (reformatted slightly from its original form):

```
data _null_;
   set p2;
   file print ods=(template="Base.Corr.StackedMatrix"
                   columns=(rowname=variable
                            matrix=MSRP(generic)
                            matrix=Invoice(generic)
                            matrix=EngineSize(generic)
                            matrix=Cylinders(generic)
                            matrix=Horsepower(generic)
                            matrix=MPG_City(generic)
                            matrix=MPG_Highway(generic)
                            matrix=Weight(generic)
                            matrix=Wheelbase(generic)
                            matrix=Length(generic)));
   put _ods_;
run;
```

CALL EXECUTE statements write the generated code to a buffer. The resulting DATA _NULL_ step executes after the DATA P2 step finishes.

The lower triangle is displayed in Output 23.10.7.

Output 23.10.7 Lower Triangle

Variable	MSRP	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Wheelbase	Length
MSRP										
Invoice	0.99913									
EngineSize	0.57175	0.56450								
Cylinders	0.64974	0.64523	0.90800							
Horsepower	0.82695	0.82375	0.78743	0.81034						
MPG_City	-0.47502	-0.47044	-0.70947	-0.68440	-0.67670					
MPG_Highway	-0.43962	-0.43459	-0.71730	-0.67610	-0.64720	0.94102				
Weight	0.44843	0.44233	0.80787	0.74221	0.63080	-0.73797	-0.79099			
Wheelbase	0.15200	0.14833	0.63652	0.54673	0.38740	-0.50728	-0.52466	0.76070		
Length	0.17204	0.16659	0.63745	0.54778	0.38155	-0.50153	-0.46609	0.69002	0.88919	

The following DATA step uses the same ODS OUTPUT data set from PROC CORR, p, and displays the lower triangle, omitting the first row and last column, which are blank:

```
data p2;
   set p end=eof;
   array __n[*] _numeric_;
   do __i = _n_ to dim(__n); __n[__i] = .; end;
   if _n_ = 1 then do;
      call execute('data _null_; set p2;');
      call execute('file print ods=(template="Base.Corr.StackedMatrix"');
      call execute('columns=(rowname=variable');
   end;
   if not eof then call execute(cats('matrix=',vname(__n[_n_]),'(generic)'));
```

```
   if eof then call execute('')); if _n_ ne 1 then put _ods_; run;');
run;
```

This DATA step contains two IF conditions, **if not eof then** and **if _n_ ne 1 then**, that omit the last column and first row, respectively. The lower triangle, without the first row and last column, is displayed in Output 23.10.8.

Output 23.10.8 Lower Triangle Omitting Blank Row and Column

Variable	MSRP	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Wheelbase
Invoice	0.99913								
EngineSize	0.57175	0.56450							
Cylinders	0.64974	0.64523	0.90800						
Horsepower	0.82695	0.82375	0.78743	0.81034					
MPG_City	-0.47502	-0.47044	-0.70947	-0.68440	-0.67670				
MPG_Highway	-0.43962	-0.43459	-0.71730	-0.67610	-0.64720	0.94102			
Weight	0.44843	0.44233	0.80787	0.74221	0.63080	-0.73797	-0.79099		
Wheelbase	0.15200	0.14833	0.63652	0.54673	0.38740	-0.50728	-0.52466	0.76070	
Length	0.17204	0.16659	0.63745	0.54778	0.38155	-0.50153	-0.46609	0.69002	0.88919

You can display the upper triangle instead of the lower triangle by replacing the first DO loop with the second:

```
* Display lower. Blank out the upper.;
do __i = _n_ to dim(__n); __n[__i] = ._; end;

* Display upper. Blank out the lower.;
do __i = 1 to _n_; __n[__i] = ._; end;
```

You also need to omit the last row and the first column (instead of the first row and the last column). The following steps change the format, display the upper triangle, and use the %Paint macro to display larger absolute values in red and values near zero in cyan:

```
%let inc = 0.01;
%paint(values=-1 to 1 by &inc, colors=red magenta cyan magenta red)

data cntlin;
   set colors;
   FmtName = 'paintfmt';
   Label = _rgb_;
   End = round(start + &inc, &inc);
   keep start end label fmtname;
run;

proc format cntlin=cntlin; run;

proc template;
   edit Base.Corr.StackedMatrix;
      column (RowName RowLabel) (Matrix);
      header 'Pearson Correlation Coefficients';
      edit matrix;format=5.2 style={foreground=paintfmt8. font_weight=bold};end;
   end;
quit;
```

```
ods select none;
proc corr data=sashelp.cars noprob;
   ods output PearsonCorr=p;
run;
ods select all;

ods _all_ close;
ods html body='upper.html' style=HTMLBlue;

data p2;
   set p end=eof;
   array __n[*] _numeric_;
   do __i = 1 to _n_; __n[__i] = ._; end;
   if _n_ = 1 then do;
      call execute('data _null_; set p2 end=eof;');
      call execute('file print ods=(template="Base.Corr.StackedMatrix"');
      call execute('columns=(rowname=variable');
   end;
   if _n_ ne 1 then call execute(cats('matrix=',vname(__n[_n_]),'(generic)'));
   if eof then call execute('')); if not eof then put _ods_; run;');
run;

ods html close;
ods html;
ods pdf;

proc template;
   delete Base.Corr.StackedMatrix / store=sasuser.templat;
quit;
```

The upper triangle of the correlation matrix is displayed in Output 23.10.9.

Output 23.10.9 Upper Triangle with Format Change and Color Changes

The SAS System

Pearson Correlation Coefficients

Variable	Invoice	EngineSize	Cylinders	Horsepower	MPG_City	MPG_Highway	Weight	Wheelbase	Length
MSRP	1.00	0.57	0.65	0.83	-0.48	-0.44	0.45	0.15	0.17
Invoice		0.56	0.65	0.82	-0.47	-0.43	0.44	0.15	0.17
EngineSize			0.91	0.79	-0.71	-0.72	0.81	0.64	0.64
Cylinders				0.81	-0.68	-0.68	0.74	0.55	0.55
Horsepower					-0.68	-0.65	0.63	0.39	0.38
MPG_City						0.94	-0.74	-0.51	-0.50
MPG_Highway							-0.79	-0.52	-0.47
Weight								0.76	0.69
Wheelbase									0.89

References

Cole, J. W. L., and Grizzle, J. E. (1966). "Applications of Multivariate Analysis of Variance to Repeated Measures Experiments." *Biometrics* 22:810–828.

Hemmerle, W. J., and Hartley, H. O. (1973). "Computing Maximum Likelihood Estimates for the Mixed AOV Model Using the W-Transformation." *Technometrics* 15:819–831.

Olinger, C. R., and Tobias, R. D. (1998). "It Chops, It Dices, It Makes Julienne Slices! ODS for Data Analysis—Output As-You-Like-It in Version 7." In *Proceedings of the Twenty-Third Annual SAS Users Group International Conference*, 1271–1291. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi23/Stats/p233.pdf.

Pothoff, R. F., and Roy, S. N. (1964). "A Generalized Multivariate Analysis of Variance Model Useful Especially for Growth Curve Problems." *Biometrika* 51:313–326.

Snedecor, G. W., and Cochran, W. G. (1967). *Statistical Methods*. 6th ed. Ames: Iowa State University Press.

Chapter 24
Statistical Graphics Using ODS

Contents

Introduction **667**
- Chapter Reading Guide 667
- Assumptions about ODS Defaults in This Chapter 668

Getting Started with ODS Statistical Graphics **669**
- Default Plots for Simple Linear Regression with PROC REG 669
- Survival Estimate Plot with PROC LIFETEST 672
- Contour and Surface Plots with PROC KDE 673
- Contour Plots with PROC KRIGE2D 675
- Partial Least Squares Plots with PROC PLS 677
- Box-Cox Transformation Plot with PROC TRANSREG 679
- LS-Means Diffogram with PROC GLIMMIX 680
- Principal Component Analysis Plots with PROC PRINCOMP 682
- Grouped Scatter Plot with PROC SGPLOT 684

A Primer on ODS Statistical Graphics **686**
- Enabling and Disabling ODS Graphics 687
- ODS Styles 688
- ODS Destinations 690
- Accessing Individual Graphs 692
- Specifying the Size and Resolution of Graphs 693
- Modifying Your Graphs 693
- Procedures That Support ODS Graphics 694
- Procedures That Support ODS Graphics and Traditional Graphics 695

Syntax **695**
- ODS GRAPHICS Statement 695
- ODS Destination Statements 698
- PLOTS= Option 699

Selecting and Viewing Graphs **701**
- Specifying an ODS Destination for Graphics 701
- Viewing Your Graphs in the SAS Windowing Environment 703
- Determining Graph Names and Labels 703
- Selecting and Excluding Graphs 706

Graphic Image Files **707**
- Image File Types 707
- Scalable Vector Graphics 708
- Naming Graphic Image Files 709

Saving Graphic Image Files . . . 711
Creating Graphs in Multiple Destinations . . . 713
Graph Size and Resolution . . . **714**
The Default Template Stores and the Template Search Path . . . **715**
ODS Styles . . . **717**
An Overview of ODS Styles . . . 717
Attribute Priority and Overriding How Groups Are Distinguished . . . 720
ODS Style Elements and Attributes . . . 726
Style Templates and Colors . . . 727
Some Common ODS Style Elements . . . 728
ODS Style Comparisons . . . 733
Modifying the HTMLBLUE Style . . . 755
ODS Style Template Modification Macro . . . 761
Varying Colors and Markers but Not Lines . . . 763
Changing the Default Markers and Lines . . . 765
Changing the Default Style . . . 773
Statistical Graphics Procedures . . . **775**
The SGPLOT Procedure . . . 775
The SGSCATTER Procedure . . . 776
The SGPANEL Procedure . . . 778
The SGRENDER Procedure . . . 780
Examples of ODS Statistical Graphics . . . **784**
Example 24.1: Creating Graphs with Tooltips in HTML . . . 784
Example 24.2: Creating Graphs for a Presentation . . . 786
Example 24.3: Creating Graphs in PostScript Files . . . 787
Example 24.4: Displaying Graphs Using the DOCUMENT Procedure . . . 790
Example 24.5: Customizing the Style for Box Plots . . . 793
Example 24.6: Splines . . . 796
Single Fit Function Using PROC SGPLOT . . . 796
Understanding Splines and Knots . . . 800
Sparse Data and Overfitting . . . 802
Multiple Fit Functions Using PROC SGPLOT . . . 804
Interpolation . . . 808
PROC TRANSREG . . . 811
The EFFECT and EFFECTPLOT Statements . . . 822
B-Splines . . . 825
Scoring . . . 831
Nonlinear Models . . . 835
PROC ADAPTIVEREG . . . 837
For More Information . . . 839
References . . . **840**

Introduction

Effective graphics are indispensable in modern statistical analysis. They reveal patterns, differences, and uncertainty that are not readily apparent in tabular output. Graphics provoke questions that stimulate deeper investigation, and they add visual clarity and rich content to reports and presentations. Statistical graphs are produced by ODS Graphics, which is an extension of ODS (the Output Delivery System). ODS manages procedure output (including both tables and graphs) and lets you display it in a variety of destinations, such as HTML and RTF. With ODS Graphics, statistical procedures produce graphs as automatically as they produce tables, and graphs are integrated with tables in the ODS output. ODS Graphics is available in procedures in SAS/STAT, Base SAS, SAS/ETS, SAS/QC, and other SAS products (see the section "Procedures That Support ODS Graphics" on page 694). ODS Graphics is automatically provided with Base SAS software.

ODS Graphics might or might not be enabled by default, depending on your operating system, whether you are in the SAS windowing environment, your registry, your system options, and your configuration file settings. For more information about default settings and enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687.

You can enable ODS Graphics by specifying the following statement:

```
ods graphics on;
```

When ODS Graphics is enabled, procedures that support ODS Graphics create the appropriate graphs, either by default or when you specify procedure options to request specific graphs. These options are documented in the "Syntax" section of each procedure chapter, and the "Details" section of each chapter provides an "ODS Graphics" subsection that lists the available graphs. Once ODS Graphics is enabled, it stays enabled for the duration of your SAS session unless you disable it.

You can disable ODS Graphics by specifying the following statement:

```
ods graphics off;
```

You might consider disabling ODS Graphics if your goal is solely to produce computational results. Often, you can enable ODS Graphics and then leave it enabled. Throughout this chapter, ODS Graphics is enabled only once per section.

Chapter Reading Guide

This chapter provides a basic introduction to ODS Graphics along with more detailed information. The following list provides a guide to reading this chapter:

- If you want to see a few of the many graphs that statistical procedures produce by using ODS Graphics, see the section "Getting Started with ODS Statistical Graphics" on page 669.
- If you are using ODS Graphics for the first time, read the section "A Primer on ODS Statistical Graphics" on page 686, which provides the minimum information that you need to get started.
- If you need to create plots of raw data or your own customized plots of statistical results, see the section "Statistical Graphics Procedures" on page 775, which describes SAS procedures that use ODS Graphics.

- If you need information about specialized topics such as accessing your graphs, making changes to your graphs, and working with ODS styles, see the section "Syntax" on page 695 and the sections "Examples of ODS Statistical Graphics" on page 784, "Selecting and Viewing Graphs" on page 701, "Graphic Image Files" on page 707, "Graph Size and Resolution" on page 714, "The Default Template Stores and the Template Search Path" on page 715, "ODS Styles" on page 717, and "Examples of ODS Statistical Graphics" on page 784.

If you are unfamiliar with ODS, see Chapter 23, "Using the Output Delivery System," for an introduction. For complete documentation about the Output Delivery System, see the *SAS Output Delivery System: User's Guide*. For an introduction to graph template modification, see Chapter 25, "ODS Graphics Template Modification." For more information about modifying the Kaplan and Meier (1958) plot in PROC LIFETEST, see Chapter 26, "Customizing the Kaplan-Meier Survival Plot." For an introduction to ODS Graphics, ODS styles, the Graph Template Language, the style template language, the statistical graphics procedures, and graph template modification, see Kuhfeld (2016). For complete documentation about ODS graph templates, see the *SAS Graph Template Language: User's Guide* and the *SAS Graph Template Language: Reference*. Also see the *SAS ODS Graphics: Procedures Guide* for information about the statistical graphics procedures and Kuhfeld (2015) for advance examples.

Assumptions about ODS Defaults in This Chapter

Default settings such as destinations and whether or not ODS Graphics is enabled vary depending on your operating system, registry settings, configuration file settings, and system options and whether you are using the SAS windowing environment or batch mode. For this reason, this chapter makes no assumptions about these defaults. Instead, destinations are often explicitly closed without assuming which destination (usually LISTING or HTML) is open, destinations are explicitly opened when needed, and ODS Graphics is explicitly enabled and disabled as needed. In some examples, when all destinations are closed, the LISTING destination is opened at the end of the step so that some destination is available for subsequent output. If you know the defaults for your environment, you do not need to use many of the ODS statements that are used in this chapter.

Getting Started with ODS Statistical Graphics

This section provides examples that illustrate the most basic uses of ODS Graphics by showing a few of the many plots that are produced by statistical procedures.

Default Plots for Simple Linear Regression with PROC REG

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

This example is based on the section "Getting Started: REG Procedure" on page 8724 in Chapter 105, "The REG Procedure." The Class data set that this example uses is available in the Sashelp library. The following statements use PROC REG to fit a simple linear regression model in which Weight is the response variable and Height is the independent variable:

```
ods graphics on;

proc reg data=sashelp.class;
   model Weight = Height;
quit;
```

The ODS GRAPHICS ON statement requests ODS Graphics in addition to the usual tabular output. The statement ODS GRAPHICS OFF is not used here, but it can be specified to disable ODS Graphics.

The graphical output consists of a fit diagnostics panel, a residual plot, and a fit plot. These plots are integrated with the tabular output and are shown in Figure 24.1, Figure 24.2, and Figure 24.3, respectively.

The results are displayed in the HTMLBLUE style. ODS styles control the colors and general appearance of all graphs and tables, and the SAS System provides several styles that are recommended for use with statistical graphics. The default style that you see when you run SAS depends on the ODS destination, system options, and SAS registry settings. For more information about styles, see the section "ODS Styles" on page 688 and the section "ODS Styles" on page 717.

Figure 24.1 Fit Diagnostics Panel

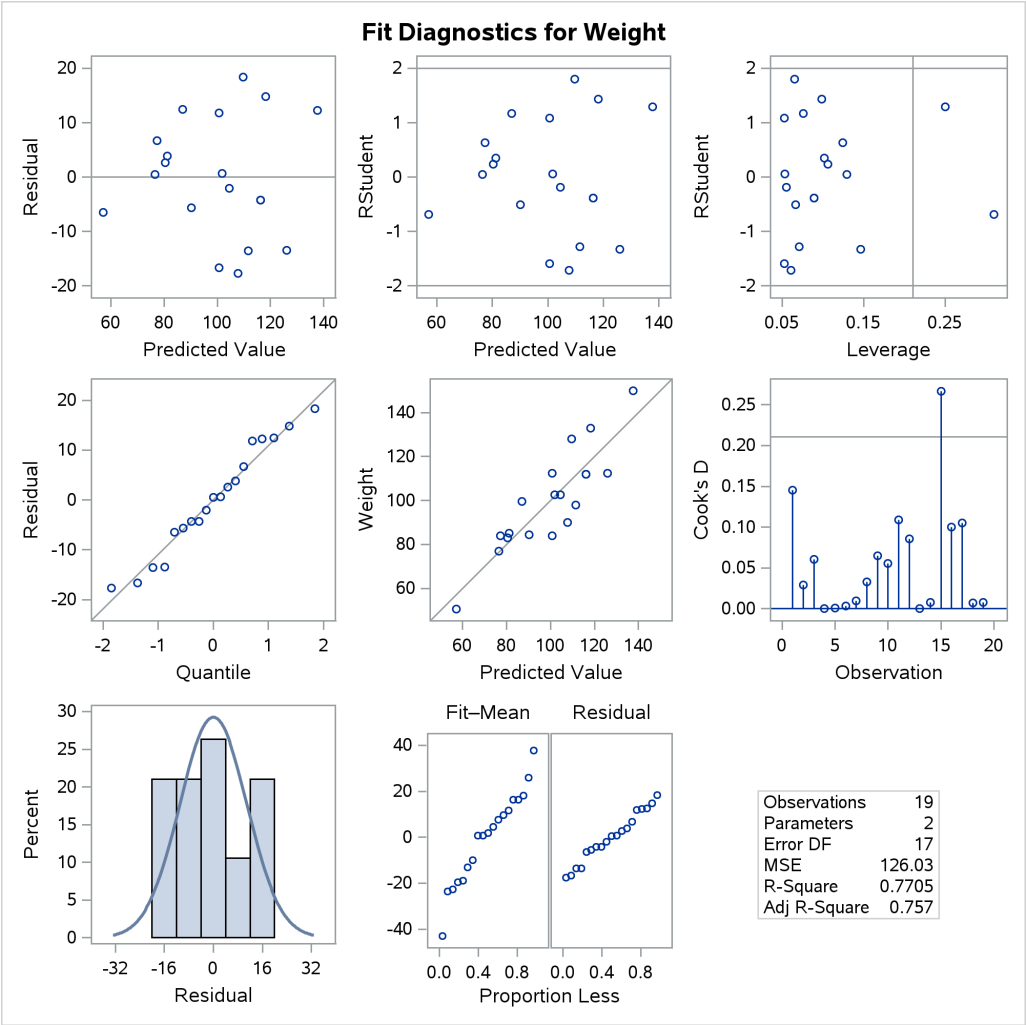

Figure 24.2 Residual Plot

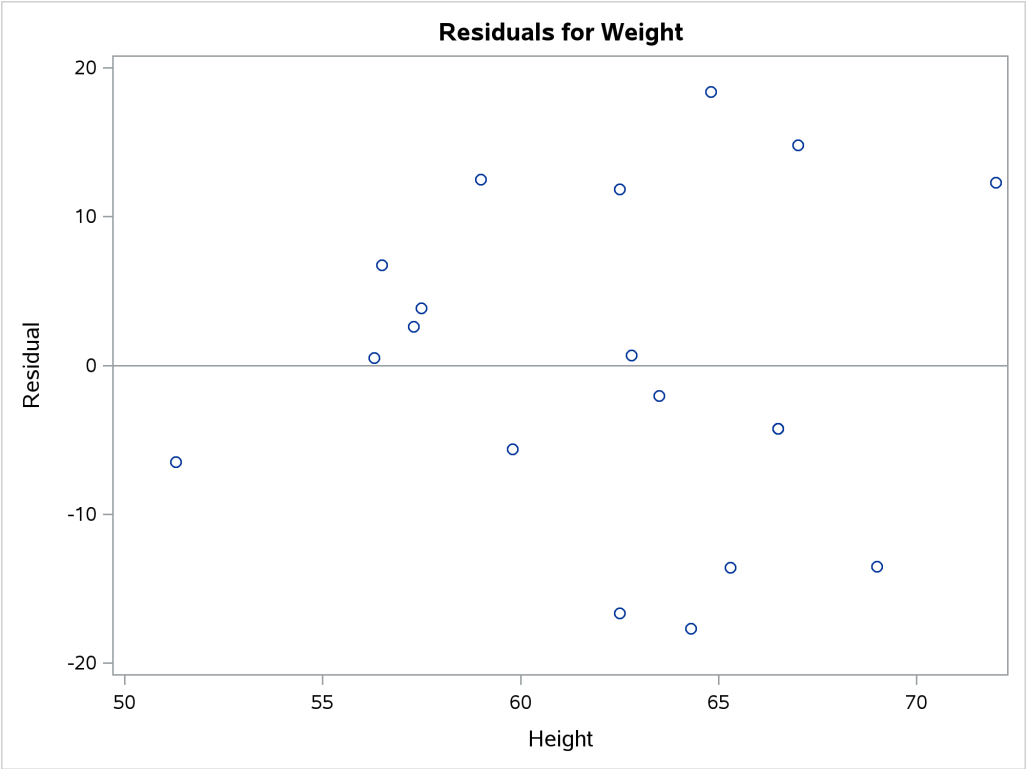

Figure 24.3 Fit Plot

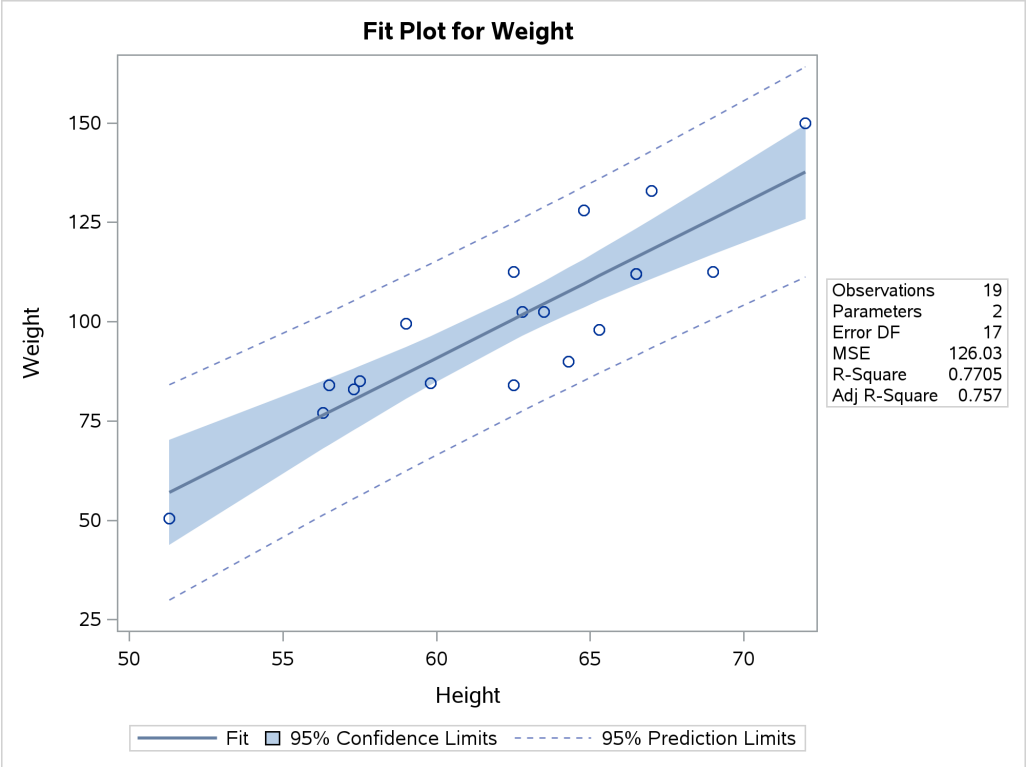

Survival Estimate Plot with PROC LIFETEST

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

This example is taken from Example 77.2 in Chapter 77, "The LIFETEST Procedure." It shows how to construct a product-limit survival estimate plot. Both the ODS GRAPHICS statement and procedure options are used to request the plot. This examples uses the bone marrow transplant data set, which is available from the Sashelp library. The data set contains disease-free times for three risk categories.

The following statements use PROC LIFETEST to compute the product-limit estimate of the survivor function for each risk category:

```
proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group / test=logrank;
run;
```

ODS Graphics was enabled in a previous step, and the PLOTS=SURVIVAL option requests a plot of the estimated survival curves. The CB=HW suboption requests Hall-Wellner confidence bands, and the TEST suboption displays the *p*-value for the log-rank test in a plot inset. For more information about modifying the Kaplan and Meier (1958) plot in PROC LIFETEST, see Chapter 26, "Customizing the Kaplan-Meier Survival Plot."

Figure 24.4 displays the plot. Patients in the AML-Low Risk group are disease-free longer than those in the ALL group, who in turn fare better than those in the AML-High Risk group.

Figure 24.4 Survival Plot

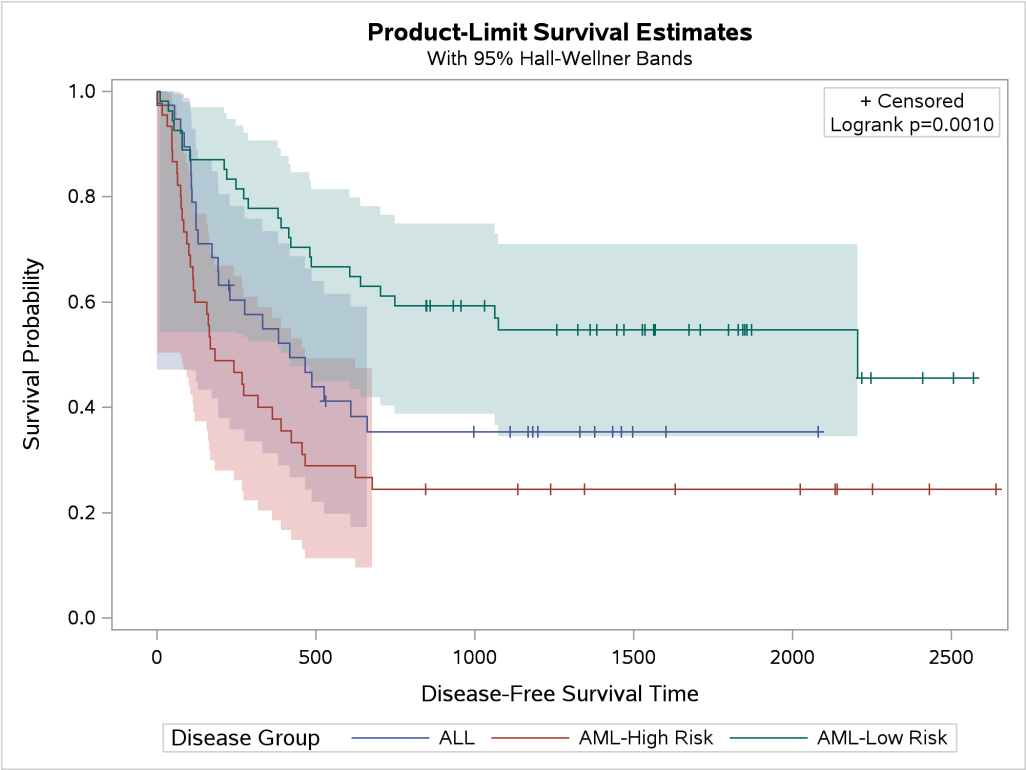

Contour and Surface Plots with PROC KDE

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

This example is taken from the section "Getting Started: KDE Procedure" on page 5502 in Chapter 73, "The KDE Procedure." Here, in addition to the ODS GRAPHICS statement, procedure options are used to request plots. The following statements simulate 1,000 observations from a bivariate normal density that has means (0,0), variances (10,10), and covariance 9:

```
data bivnormal;
   do i = 1 to 1000;
      z1 = rannor(104);
      z2 = rannor(104);
      z3 = rannor(104);
      x  = 3*z1+z2;
      y  = 3*z1+z3;
      output;
   end;
run;
```

The following statements request a bivariate kernel density estimate for the variables x and y:

```
proc kde data=bivnormal;
   bivar x y / plots=contour surface;
run;
```

The PLOTS= option requests a contour plot and a surface plot of the estimate (displayed in Figure 24.5 and Figure 24.6, respectively). For more information about the graphs available in PROC KDE, see the section "ODS Graphics" on page 5527 in Chapter 73, "The KDE Procedure."

Figure 24.5 Contour Plot of Estimated Density

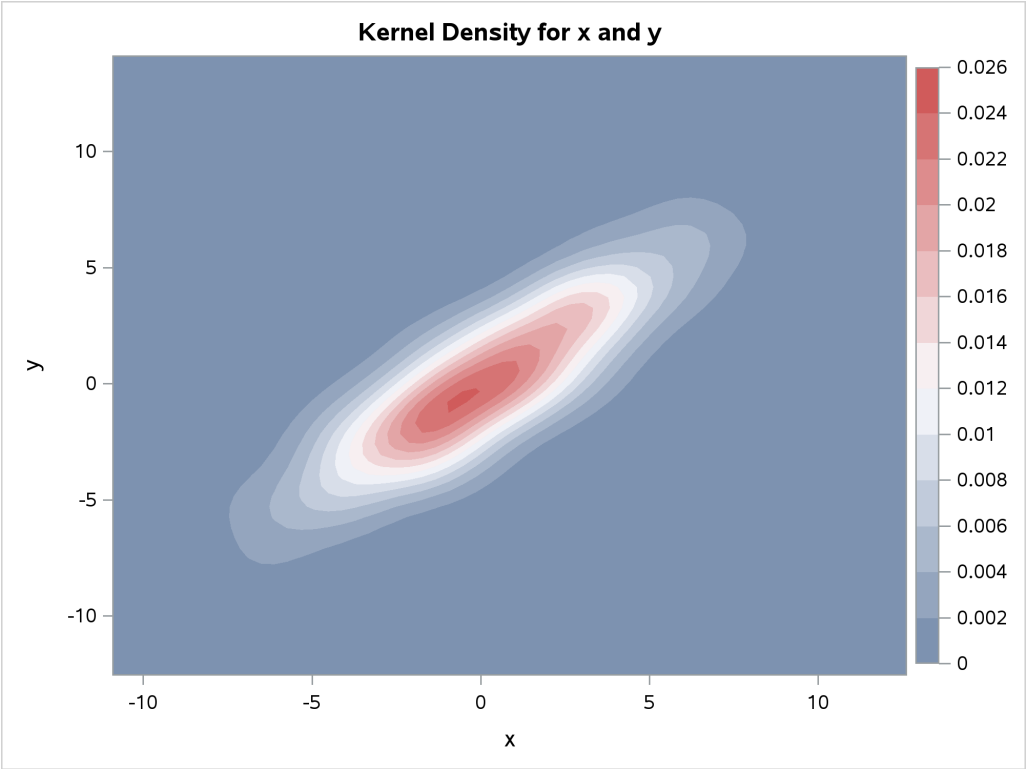

Figure 24.6 Surface Plot of Estimated Density

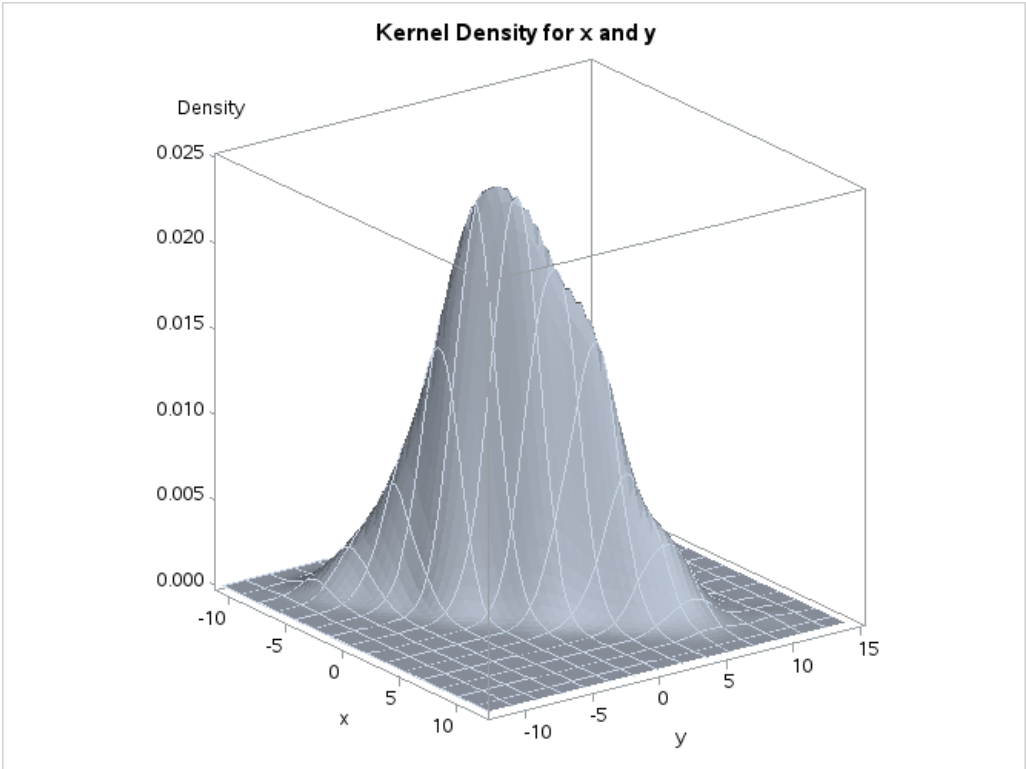

Contour Plots with PROC KRIGE2D

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

This example is taken from Example 74.2 in Chapter 74, "The KRIGE2D Procedure." The coal seam thickness data set is available from the Sashelp library. The following statements create a SAS data set that contains a copy of these data along with some artificially added missing data:

```
data thick;
   set sashelp.thick;
   if _n_ in (41, 42, 73) then thick = .;
run;
```

The following statements run PROC KRIGE2D:

```
proc krige2d data=thick outest=predictions
             plots=(observ(showmissing)
                    pred(fill=pred line=pred obs=linegrad)
                    pred(fill=se line=se obs=linegrad));
   coordinates xc=East yc=North;
   predict var=Thick r=60;
   model scale=7.2881 range=30.6239 form=gauss;
   grid x=0 to 100 by 2.5 y=0 to 100 by 2.5;
run;
```

The PLOTS=OBSERV(SHOWMISSING) option produces a scatter plot of the data along with the locations of any missing data. The PLOTS=PRED option produces maps of the kriging predictions and standard errors. Two instances of the PLOTS=PRED option are specified along with suboptions that customize the plots. The results are shown in Figure 24.7.

Figure 24.7 Spatial Distribution

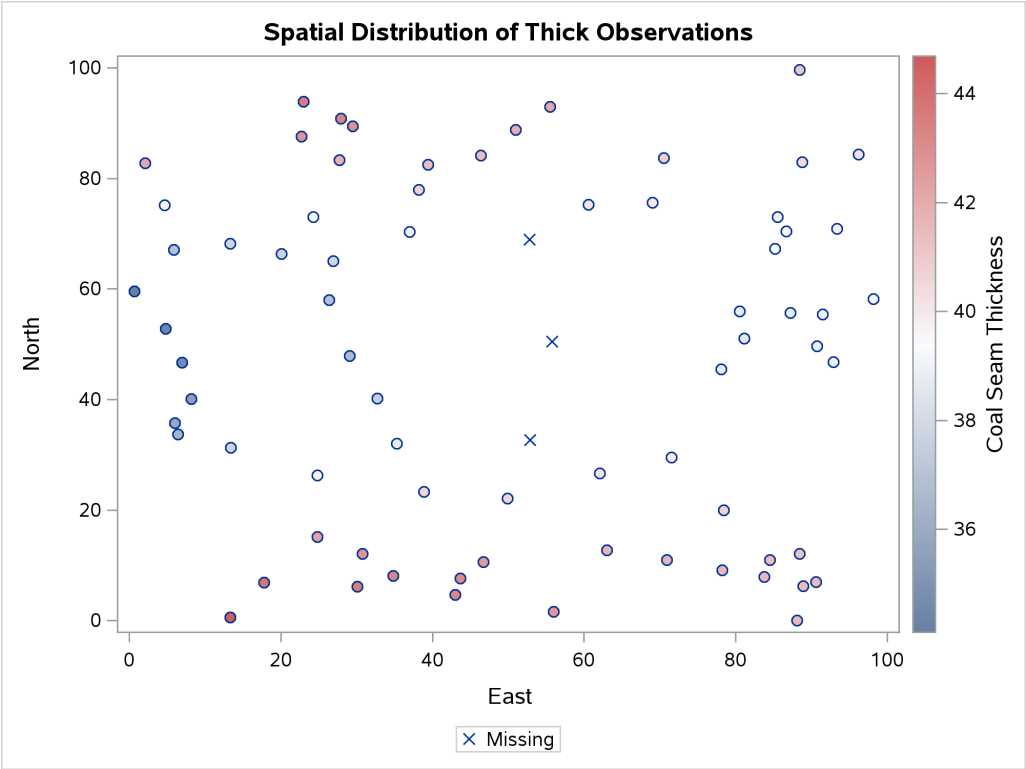

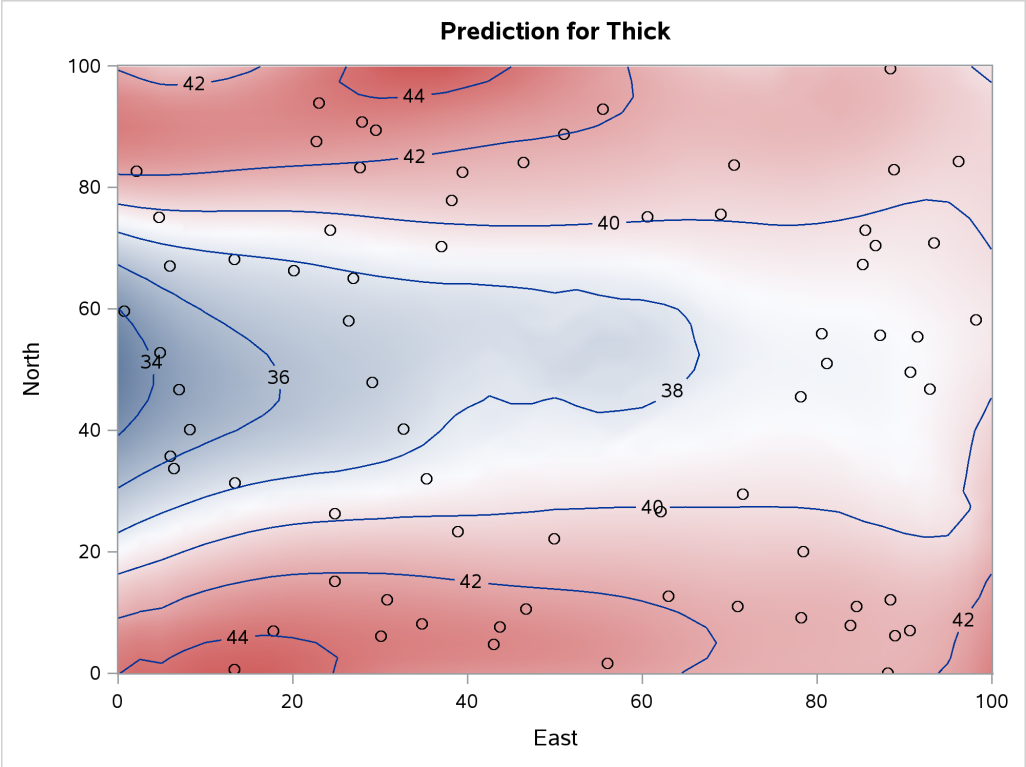

Figure 24.7 *continued*

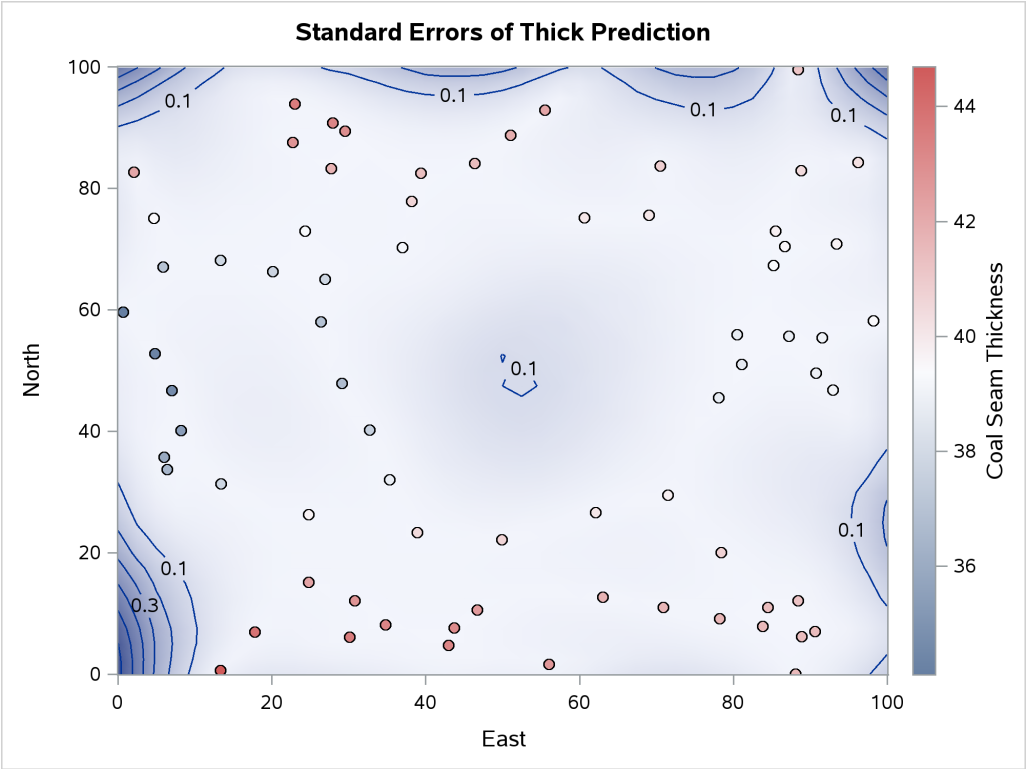

Partial Least Squares Plots with PROC PLS

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

This example is taken from the section "Getting Started: PLS Procedure" on page 7839 in Chapter 95, "The PLS Procedure." The following statements create a SAS data set that contains measurements of biological activity in the Baltic Sea:

```
data Sample;
   input obsnam $ v1-v27 ls ha dt @@;
   datalines;
EM1    2766 2610 3306 3630 3600 3438 3213 3051 2907 2844 2796
       2787 2760 2754 2670 2520 2310 2100 1917 1755 1602 1467
       1353 1260 1167 1101 1017         3.0110  0.0000   0.00
EM2    1492 1419 1369 1158  958  887  905  929  920  887  800

   ... more lines ...

;
```

The following statements run PROC PLS:

```
proc pls data=sample cv=split cvtest(seed=104);
   model ls ha dt = v1-v27;
run;
```

By default, the procedure produces a plot for the cross validation analysis and a correlation loading plot (see Figure 24.8).

Figure 24.8 Partial Least Squares

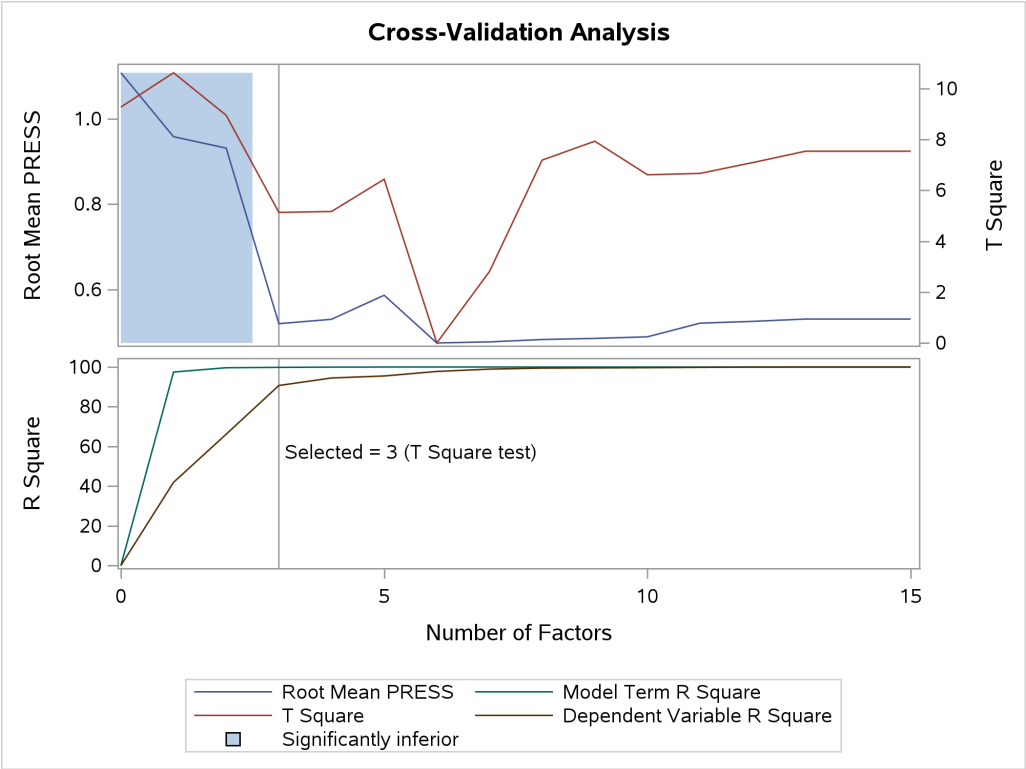

Figure 24.8 *continued*

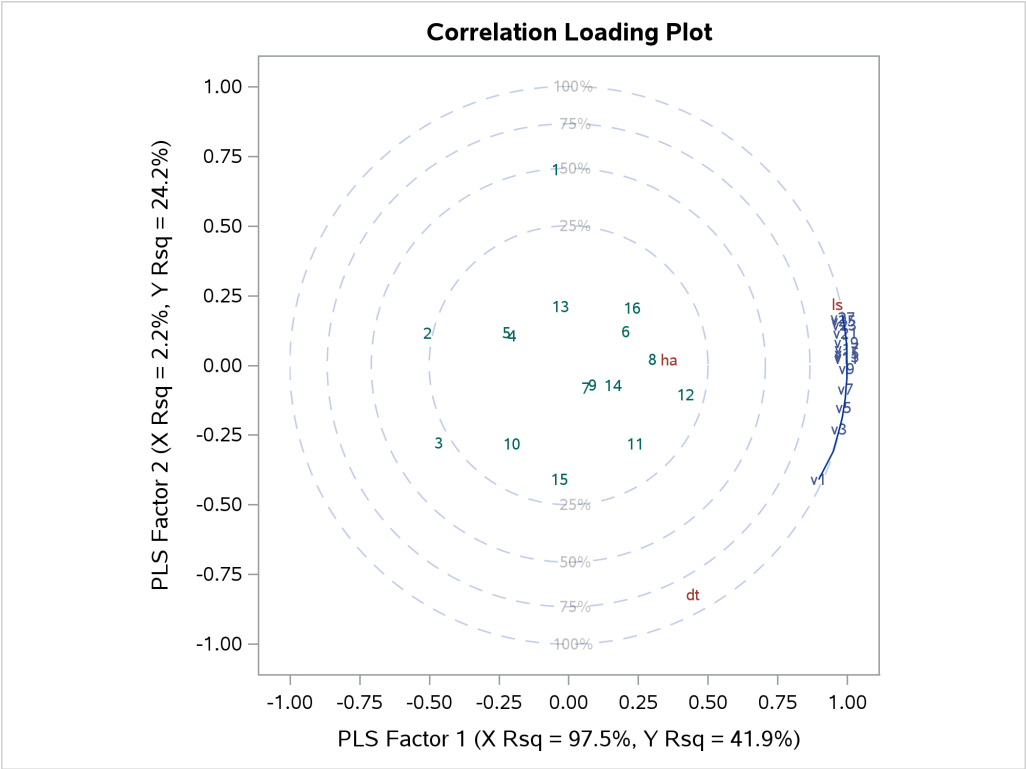

Box-Cox Transformation Plot with PROC TRANSREG

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

This example is taken from Example 126.2 in Chapter 126, "The TRANSREG Procedure." The following statements create a SAS data set that contains failure times for yarn:

```
proc format;
   value a -1 =   8 0 =   9 1 =  10;
   value l -1 = 250 0 = 300 1 = 350;
   value o -1 =  40 0 =  45 1 =  50;
run;

data yarn;
   input Fail Amplitude Length Load @@;
   format amplitude a. length l. load o.;
   label fail = 'Time in Cycles until Failure';
   datalines;
 674 -1 -1 -1    370 -1 -1  0    292 -1 -1  1    338  0 -1 -1
 266  0 -1  0    210  0 -1  1    170  1 -1 -1    118  1 -1  0

   ... more lines ...

;
```

The following statements run PROC TRANSREG:

```
proc transreg data=yarn;
   model BoxCox(fail / convenient lambda=-2 to 2 by 0.05) =
         qpoint(length amplitude load);
run;
```

The log-likelihood plot in Figure 24.9 suggests a Box-Cox transformation where $\lambda = 0$.

Figure 24.9 Box-Cox "Significant Effects"

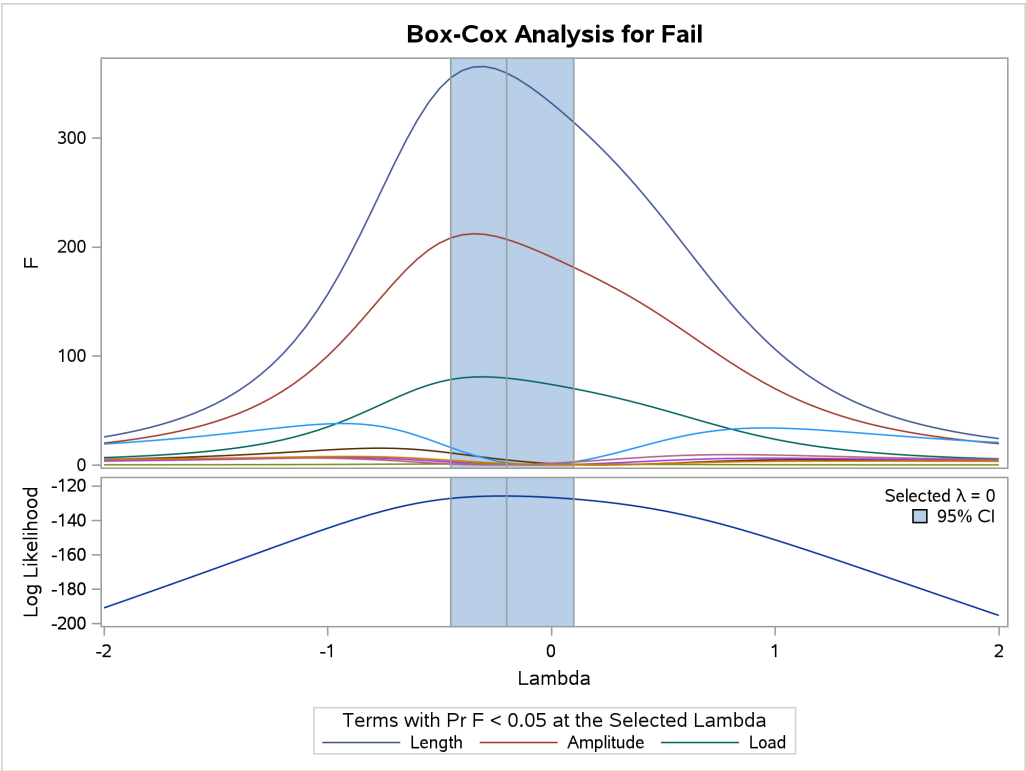

LS-Means Diffogram with PROC GLIMMIX

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

This example is taken from the section "Graphics for LS-Mean Comparisons" on page 3997 in Chapter 52, "The GLIMMIX Procedure." The following statements create a SAS data set that contains measurements from an experiment that investigates how snapdragons grow in various soils:

```
data plants;
   input Type $ @;
   do Block = 1 to 3;
      input StemLength @@;
      output;
   end;
```

```
   datalines;
Clarion    32.7 32.3 31.5   Clinton    32.1 29.7 29.1   Knox       35.7 35.9 33.1
ONeill     36.0 34.2 31.2   Compost    31.8 28.0 29.2   Wabash     38.2 37.8 31.9
Webster    32.5 31.1 29.7
;
```

The following statements run PROC GLIMMIX:

```
proc glimmix data=plants order=data plots=diffogram;
   class Block Type;
   model StemLength = Block Type;
   lsmeans Type;
run;
```

The PLOTS=DIFFOGRAM option produces a diffogram, shown in Figure 24.10, that displays all the pairwise least squares mean differences and indicates which are significant.

Figure 24.10 LS-Means Diffogram

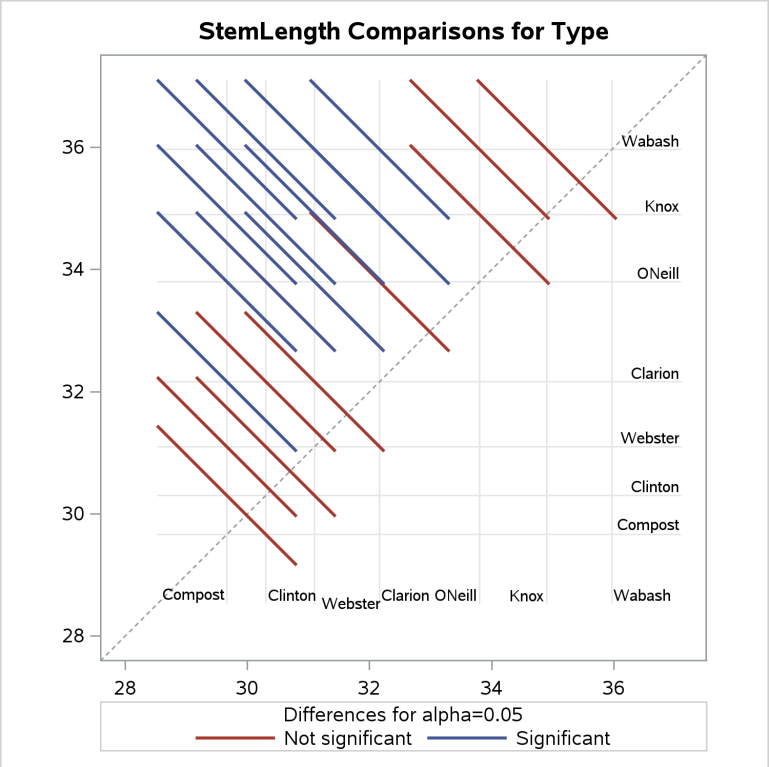

Principal Component Analysis Plots with PROC PRINCOMP

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

This example is taken from Example 98.3 in Chapter 98, "The PRINCOMP Procedure." The following statements create a SAS data set that contains job performance ratings of police officers:

```
options validvarname=any;

data Jobratings;
   input ('Communication Skills'n
          'Problem Solving'n
          'Learning Ability'n
          'Judgment Under Pressure'n
          'Observational Skills'n
          'Willingness to Confront Problems'n
          'Interest in People'n
          'Interpersonal Sensitivity'n
          'Desire for Self-Improvement'n
          'Appearance'n
          'Dependability'n
          'Physical Ability'n
          'Integrity'n
          'Overall Rating'n) (1.);
   datalines;
26838853879867
74758876857667

   ... more lines ...

;
```

The following statements run PROC PRINCOMP:

```
proc princomp data=Jobratings(drop='Overall Rating'n) n=2
              plots=(Matrix PatternProfile);
run;
```

The plots are requested by the PLOTS=(MATRIX PATTERNPROFILE) option. The results, shown in Figure 24.11, contain the default scree and variance-explained plots, along with a scatter plot matrix of component scores and a pattern profile plot.

Figure 24.11 Principal Component Analysis

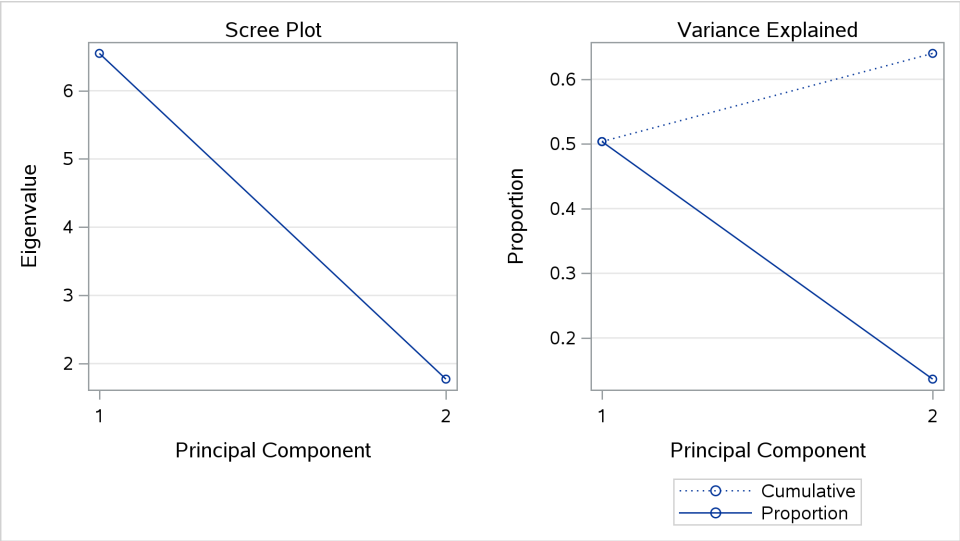

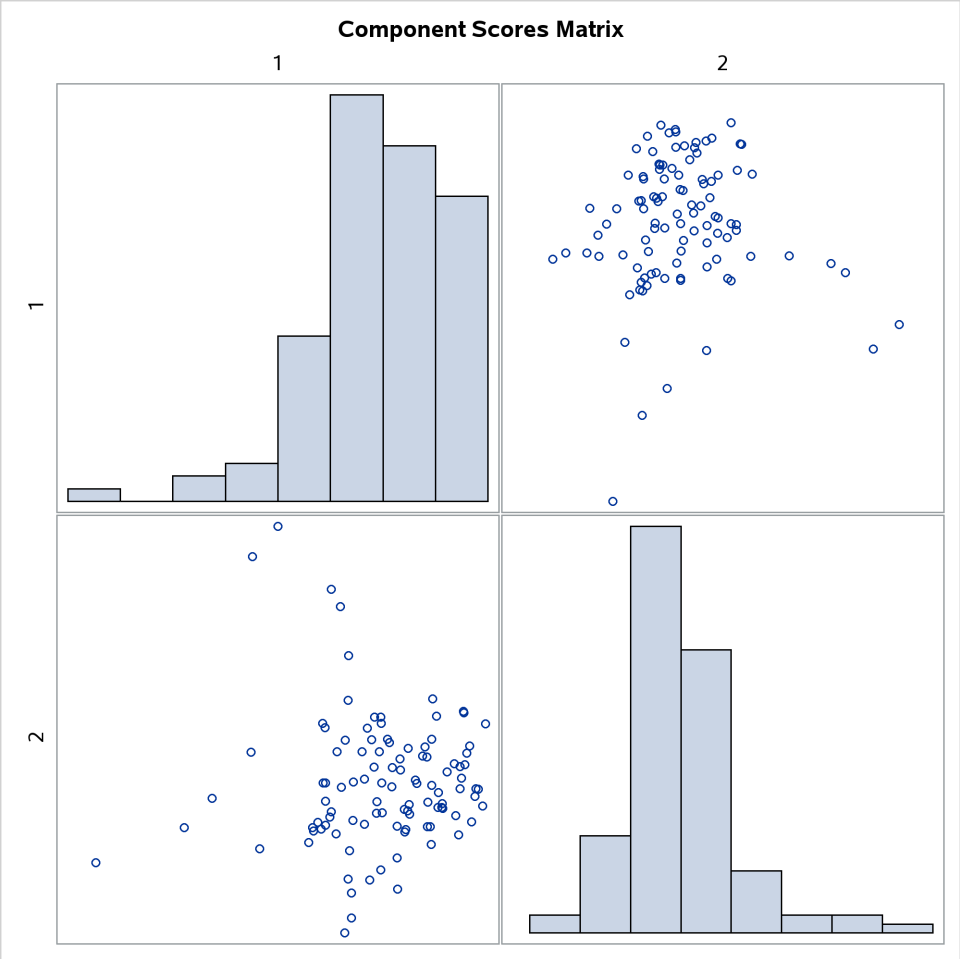

Figure 24.11 *continued*

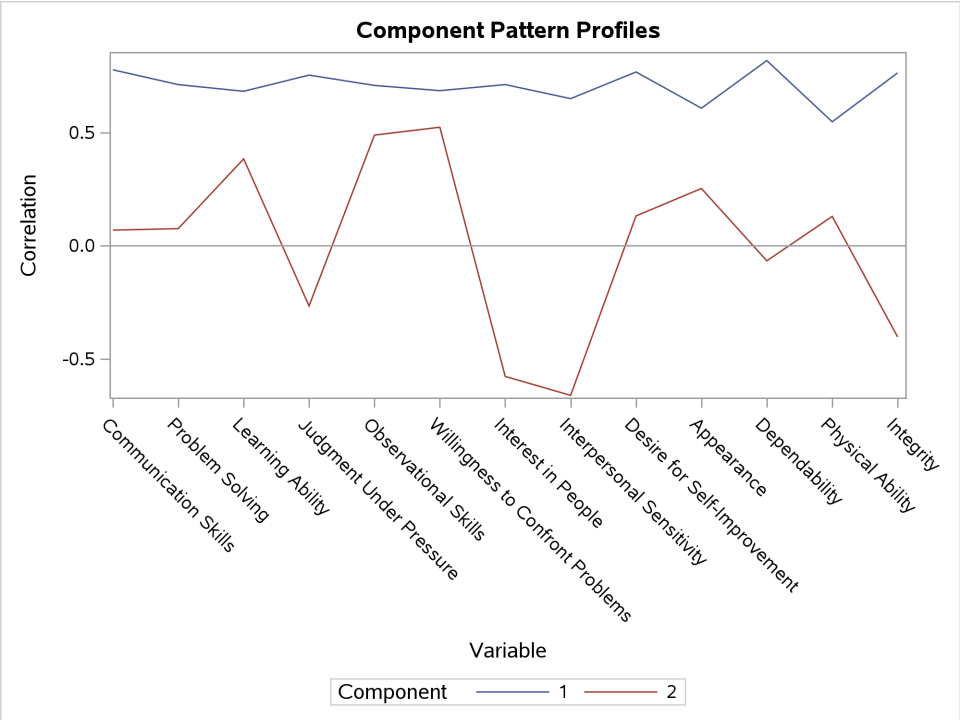

Grouped Scatter Plot with PROC SGPLOT

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas`.)

This example is taken from Example 42.1 in Chapter 42, "The DISCRIM Procedure." It uses the Fisher iris data set, which is available from the Sashelp library.

The following statements run PROC SGPLOT to make a scatter plot, grouped by iris species:

```
proc sgplot data=sashelp.iris;
   title 'Fisher (1936) Iris Data';
   scatter x=petallength y=petalwidth / group=species;
run;
```

The results are shown in Figure 24.12.

Figure 24.12 Iris Data

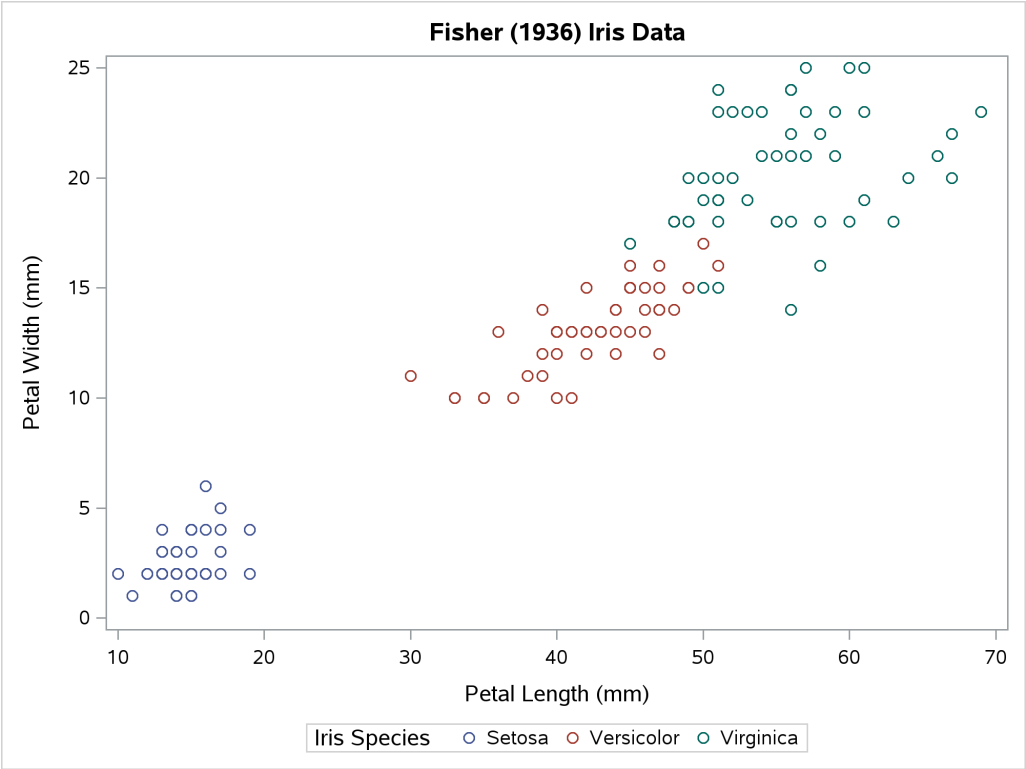

For more information about PROC SGPLOT (statistical graphics plot) and other SG procedures, see the section "Statistical Graphics Procedures" on page 775 and the *SAS ODS Graphics: Procedures Guide*. You do not need to enable ODS Graphics in order to use SG procedures (because making plots by using ODS Graphics is their sole function). However, you can use the ODS GRAPHICS statement to change options, as in the following example:

```
ods graphics on / attrpriority=none;

proc sgplot data=sashelp.iris;
   title 'Fisher (1936) Iris Data';
   styleattrs datasymbols=(circlefilled squarefilled starfilled);
   scatter x=petallength y=petalwidth / group=species markerattrs=(size=5px);
run;

ods graphics / reset;
```

The STYLEATTRS statement specifies the symbols for the three groups of observations. The results are shown in Figure 24.13. The groups in Figure 24.13 are distinguished by varying colors and symbols. In contrast, the groups in Figure 24.12 are distinguished only by colors. The ATTRPRIORITY=NONE option in the ODS GRAPHICS statement ensures that the second and third symbols are used. The HTMLBLUE style is an ATTRPRIORITY='Color' style, so by default, the symbol and line style attributes are not used to distinguish groups. The ATTRPRIORITY=NONE option in the ODS GRAPHICS statement overrides the ATTRPRIORITY='Color' option in the style, so that all three symbols are used. For more information, see the section "Attribute Priority and Overriding How Groups Are Distinguished" on page 720.

Figure 24.13 Iris Data Controlling the Symbols

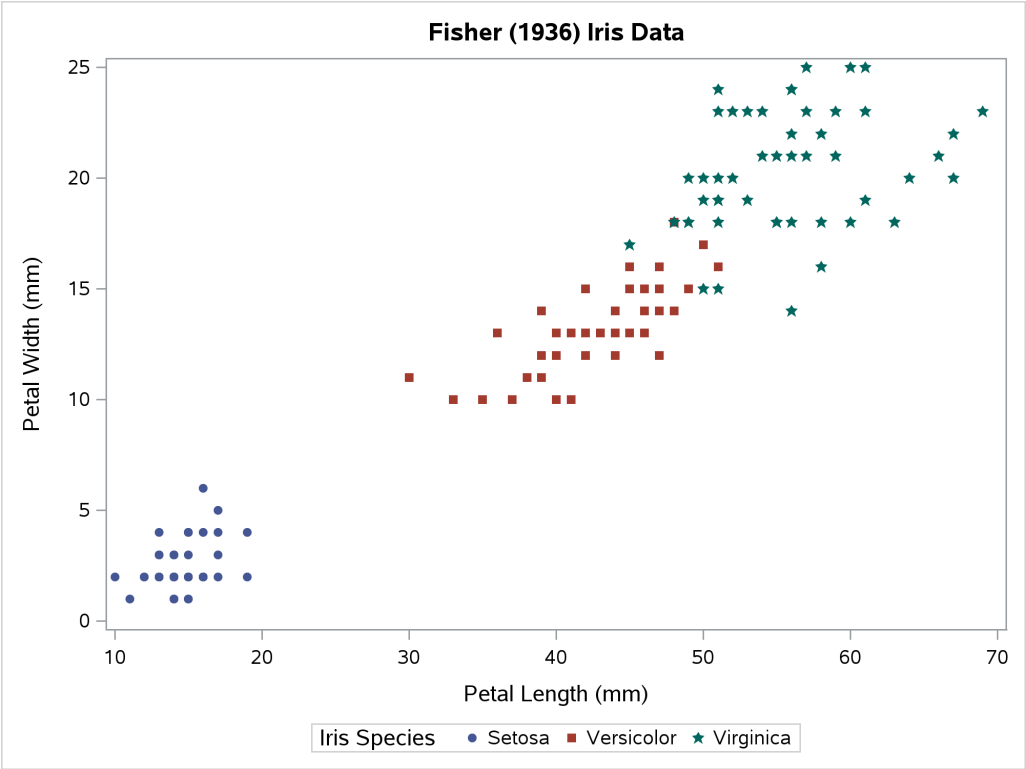

A Primer on ODS Statistical Graphics

You can enable ODS Graphics by specifying the following statement:

```
ods graphics on;
```

ODS Graphics remains enabled for all procedure steps until you disable it by submitting the following statement:

```
ods graphics off;
```

When ODS Graphics is enabled, creating graphical output with procedures is as simple as creating tabular output. For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687. For more information about the most commonly used ODS GRAPHICS statement options, see the section "Syntax" on page 695.

You can control your output by using the following ODS components:

- ODS destination statements (such as ODS HTML or ODS RTF), which specify where you want your graphs to be displayed. For an example of HTML output, see Figure 24.19. For a list of the supported destinations, see the section "ODS Destination Statements" on page 698. For more information about the most commonly used ODS destination statement options, see the section "Syntax" on page 695 .

- ODS SELECT and ODS EXCLUDE statements, which select and exclude graphs from your output. See the section "Selecting and Excluding Graphs" on page 706 for an example of how to select graphs.
- ODS OUTPUT statements, which create SAS data sets from the data object that is used to make the plot. See the section "Specifying an ODS Destination for Graphics" on page 701 for an example.
- procedure options, which specify what graphs to create. For each procedure, these options are described in the "Syntax" section of the procedure chapter. Usually, you use the PLOTS= option to control all graphs. The available graphs are listed in the "ODS Graphics" section, which is found in the "Details" section of each procedure chapter. Many graphs are produced by default.
- ODS style templates, which control the general appearance and consistency of all graphs and tables. You can modify the styles that SAS provides and override style information in several ways. For more information about styles, see the sections "ODS Styles" on page 688 and "ODS Styles" on page 717.
- ODS graph templates, which modify the layout and details of each graph. For more information about graph templates, see the section "Graph Templates" on page 844 in Chapter 25, "ODS Graphics Template Modification."

  NOTE: SAS provides a default template for each graph, so you do not need to know anything about templates in order to create statistical graphics.

You can also access individual graphs, control the resolution and size of graphs, and modify your graphs (as explained in the sections beginning with "Selecting and Viewing Graphs" on page 701). Alternatively, you can use special statistical graphics procedures to create custom graphs directly (see the section "Statistical Graphics Procedures" on page 775).

Enabling and Disabling ODS Graphics

You can enable ODS Graphics by specifying the following statement:

```
ods graphics on;
```

ODS Graphics remains enabled for all procedure steps until you disable it by submitting the following statement:

```
ods graphics off;
```

ODS Graphics might or might not be enabled by default. This depends on a number of factors. ODS Graphics is usually enabled by default in the SAS windowing environment; ODS Graphics is usually disabled by default when you invoke SAS in other ways. However, you can change these defaults in a number of ways. You can enable or disable ODS Graphics by default in an *autoexec.sas* file, a configuration file such as *SASV9.CFG*, or the SAS registry. You can change the default in the SAS windowing environment by selecting **Tools ► Options ► Preferences** from the main SAS window. Then on the **Results** tab, select the **Use ODS Graphics** check box to enable ODS Graphics by default, or clear the check box to disable ODS Graphics by default. You can also change the default output destination (HTML or LISTING) on the **Results** tab. For more information about default ODS Graphics settings and default destinations, see the section "HTML Output in the SAS Windowing Environment" on page 592 in Chapter 23, "Using the Output Delivery System."

When ODS Graphics is enabled, procedures that support ODS Graphics create graphs, either by default or when you specify procedure options for requesting specific graphs. Often, you can leave ODS Graphics enabled for the duration of your SAS session. However, you might consider disabling ODS Graphics if your goal is solely to produce computational results, particularly for large data sets or with many BY groups.

ODS Styles

ODS styles control the overall appearance of graphs and tables. They specify colors, fonts, line styles, symbol markers, and other attributes of graph elements. There are two types of ODS styles:

- ATTRPRIORITY='Color' style, which distinguishes groups of observations by color changes and not by line style or symbol changes.[1] ATTRPRIORITY='Color' styles include HTMLBLUE, PEARL, PEARLJ, and SAPPHIRE. If you want to control the markers or lines that are displayed for groups of observations when using an ATTRPRIORITY='Color' style, be sure to first specify the ATTRPRIORITY=NONE option in the ODS GRAPHICS statement or switch to an ATTRPRIORITY='None' style. For more information, see the sections "Attribute Priority and Overriding How Groups Are Distinguished" on page 720.

- ATTRPRIORITY='None' style, which distinguishes groups of observations by simultaneous color, marker, and line changes. Most ODS styles are ATTRPRIORITY='None' styles. They are compromise styles in the sense that some graph elements are intentionally overdistinguished to facilitate black-and-white printing. For example, fit lines that correspond to different classification levels are distinguished by both colors and line patterns. You can use the ATTRPRIORITY='Color' styles (such as HTMLBLUE, PEARL, PEARLJ, and SAPPHIRE) when you want groups to be distinguished only by color.

Although you can use any ODS style, only a few styles are usually used with ODS Graphics. They are described in Table 24.1.

[1] More precisely, an ATTRPRIORITY='Color' style such as HTMLBLUE distinguishes the first 12 groups of observations only by color. Markers and lines change for groups 13–24 and then again for groups 25–36. Figure 24.50 shows how colors, markers, and line styles change in the HTMLBLUE style, and Figure 24.49 shows how these change in most other ODS styles.

Table 24.1 ODS Styles Most Often Used with ODS Graphics

Style	Recommended Destinations	Attribute Priority	Description
ANALYSIS	HTML	None	A color style, with sans serif fonts, whose dominant colors are yellow, green, and tan. See Figure 24.22.
DEFAULT	HTML	None	A color style, with bold sans serif fonts, whose dominant colors are gray, blue, and white. See Figure 24.18.
HTMLBLUE	HTML	Color ✓	A color style, with sans serif fonts, whose dominant colors are shades of blue. See Figure 24.19. Default for HTML destination and SAS/STAT documentation.
HTMLBLUECML	HTML	None ✓	An ATTRPRIORITY='None' version of HTMLBLUE. See Figure 24.20.
JOURNAL, JOURNAL1A	PDF, PS, RTF, PRINTER	None ✓	A black-and-white style with sans serif fonts and filled areas. See Figure 24.23, Figure 24.28, and Figure 24.29.
JOURNAL2, JOURNAL2A	PDF, PS, RTF, PRINTER	None ✓	A black-and-white style, similar to JOURNAL but with empty areas. Grouped bar charts use crosshatching to show groups. See Figure 24.30 and Figure 24.31.
JOURNAL3, JOURNAL3A	PDF, PS, RTF, PRINTER	None ✓	A black-and-white style, similar to JOURNAL2 but with a mix of filled areas and crosshatching in grouped bar charts. See Figure 24.32 and Figure 24.33.
LISTING	HTML, LISTING	None	A color style, similar to DEFAULT but with a white background. See Figure 24.24. Default for the LISTING destination.
PEARL	PDF, PS, RTF, PRINTER	Color ✓	A color style, with sans serif fonts and a white background, whose dominant colors are shades of blue. See Figure 24.26 and Figure 24.34. Default for PDF destination.
PEARLJ	PDF, PS, RTF, PRINTER	Color ✓	A color style, with sans serif fonts and a white background, whose dominant colors are shades of blue. See Figure 24.35. Default for PDF tables in SAS/STAT documentation.
RTF	RTF	None	A color style, with serif (Times Roman) fonts, whose dominant colors are blue, white, and black. See Figure 24.25 and Figure 24.36. Default for RTF destination.
SAPPHIRE	PDF, PS, RTF, PRINTER	Color ✓	A color style, with sans serif fonts, a white background, and a light blue table heading background, whose dominant colors are shades of blue. See Figure 24.27 and Figure 24.37.
STATISTICAL	HTML	None	A color style, with sans serif fonts, whose dominant colors are blue, gray, and white. See Figure 24.21.

✓indicates newer styles that are recommended for use with statistical graphics.
JOURNAL# styles differ from JOURNAL#A styles in that the former use italic fonts in table headings.

You specify an ODS style by using the STYLE= option in the ODS destination statement. For example, the following statement creates RTF output and specifies the JOURNAL style:

```
ods rtf style=Journal;
```

The following statement sets the style for the LISTING destination:

```
ods listing style=HTMLBlue;
```

The style that is specified by the STYLE= option in the ODS LISTING statement applies only to graphs. SAS monospace format is used for tables.

More generally, you can modify the colors, fonts, and other attributes of graph elements in an ODS style by editing the style template. For more information, see the section "ODS Styles" on page 717 and *SAS Output Delivery System: User's Guide*. You can also perform ODS style modifications by using the %MODSTYLE SAS autocall macro. For more information, see the section "ODS Style Template Modification Macro" on page 761.

ODS Destinations

ODS can send your graphs and tables to a number of different destinations, including RTF (rich text format), HTML (hypertext markup language), LISTING (the SAS LISTING destination), DOCUMENT (the ODS document), and PDF (portable document format). You use an ODS statement to open a destination, as in the following examples:

```
ods html body='b.htm';
ods rtf;
ods listing;
ods document name=MyDoc(write);
ods pdf file="contour.pdf";
```

You can close destinations individually or all at once, as in the following examples:

```
ods html close;
ods rtf close;
ods listing close;
ods document close;
ods pdf close;
ods _all_ close;
```

For most ODS destinations (such as HTML, RTF, and PDF), graphs and tables are integrated in the output, and you view your output by using an appropriate viewer, such as a web browser for HTML. However, the LISTING destination is different. If you are using the LISTING destination in the SAS windowing environment, you view your graphs individually by clicking the graph icons in the Results window, shown in Figure 24.14. This action invokes a host-dependent graph viewer (for example, Microsoft Photo Editor in Windows). The graphs that ODS Graphics produces are *not* displayed along with traditional graphs in the Graph window.

Figure 24.14 SAS Results Window

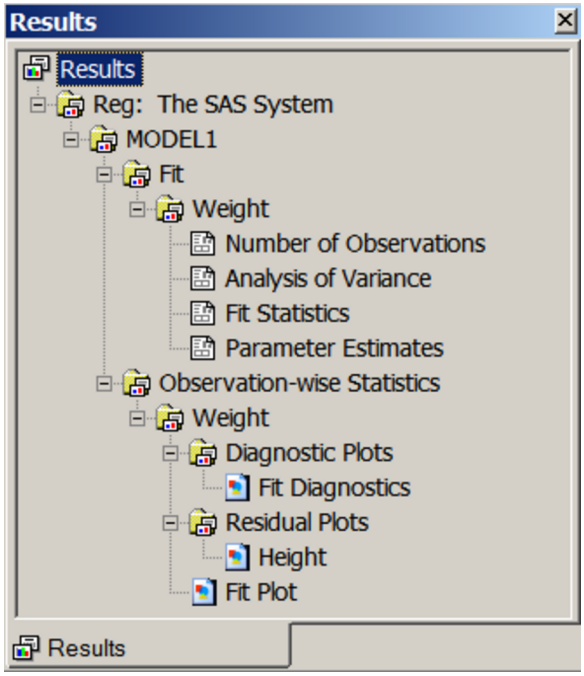

If you are using the SAS windowing environment and you prefer to view integrated output, you should use a destination such as HTML or RTF. In many cases, HTML is the default destination in the SAS windowing environment (see the section "HTML Output in the SAS Windowing Environment" on page 592 in Chapter 23, "Using the Output Delivery System"). You can change destinations in the SAS windowing environment by selecting **Tools ► Options ► Preferences** from the main SAS window and then clicking the **Results** tab.

You can prevent the Output window from appearing by using ODS statements to close the LISTING destination, as follows:

```
ods listing close;
ods html;
```

ODS creates a graph for every open destination. When you open a new destination, you should close all destinations that you do not need. Closing destinations makes your jobs run faster and use fewer resources, because fewer tables and graphs are produced.

Accessing Individual Graphs

If you are writing a paper or creating a presentation, you need to access your graphs individually. There are various ways to do this, depending on the ODS destination. Three particularly useful methods are as follows:

- If you are viewing RTF output, you can simply copy your graphs from the viewer and paste them into a Microsoft Word document or PowerPoint slide.
- If you are viewing HTML output, you can copy and paste your graphs from the viewer, or you can right-click the graph and save it to a file. For more information, see the section "Specifying the Size and Resolution of Graphs" on page 693.
- You can save your graphs in image files and then include them in a paper or presentation. For example, you can save your graphs as PNG files and include them in an HTML document or in a paper that you are writing in LaTeX.

You can specify the graphics image format and the file name in the ODS GRAPHICS statement. You can specify the graph resolution by using the IMAGE_DPI= option in an ODS destination statement. For example, the following statements, when submitted before a procedure step that produces multiple graphs, save the graphs in PostScript files named *myname.ps*, *myname1.ps*, and so on:

```
ods _all_ close;
ods latex;
ods graphics on / outputfmt=ps imagename='myname';
```

The following statements save the graphs in PNG files with a resolution of 300 dots per inch (DPI):

```
ods _all_ close;
ods html image_DPI=300;
ods graphics on;
```

For more information about the file types available with various destinations, how they are named, and how they are saved, see the section "Image File Types" on page 707. For more information about resolution, see the section "Specifying the Size and Resolution of Graphs" on page 693. For more information about scalable vector graphics, see the section "Scalable Vector Graphics" on page 708. If you are using the LISTING destination and the SAS windowing environment, you can copy from the viewer into a Microsoft Word document or PowerPoint slide.

Specifying the Size and Resolution of Graphs

Two factors to consider when you are creating graphs for a paper or presentation are the size of the graph and its resolution. You can specify the size of a graph in the ODS GRAPHICS statement. The following examples show typical ways to change the size of your graphs:

```
ods graphics on / width=6in;
ods graphics on / height=4in;
ods graphics on / width=4.5in height=3.5in;
```

You can change the resolution by specifying the IMAGE_DPI= option in any ODS destination statement, as in the following example:

```
ods html image_dpi=300;
```

The default resolution of graphs that you create by using the HTML and LISTING destinations is 96 DPI, whereas the default resolution in the RTF destination is 200 DPI. An increase in resolution often improves the quality of the graphs, but it also increases the size of the image file. For more information about graph size and resolution, see the section "Graph Size and Resolution" on page 714.

Modifying Your Graphs

Although ODS Graphics is designed to automate the creation of high-quality statistical graphics, you might occasionally need to modify your graphs. You can make persistent changes by modifying the ODS graph template for a particular plot. (For an introduction to graph template modification, see Chapter 25, "ODS Graphics Template Modification.") A graph template is a program, written in the Graph Template Language (GTL), that specifies the layout and details of a graph.

NOTE: The SAS System provides a template for each graph that it creates, so you do not need to know anything about templates in order to create statistical graphics.

Procedures That Support ODS Graphics

SAS procedures that support ODS Graphics include the following:

SAS/STAT

ACECLUS
ADAPTIVEREG
ANOVA
BCHOICE
BGLIMM
BOXPLOT
CALIS
CAUSALTRT
CLUSTER
CORRESP
FACTOR
FMM
FREQ
GAM
GAMPL
GEE
GENMOD
GLIMMIX
GLM
GLMPOWER
GLMSELECT
HPFMM
HPSPLIT
ICLIFETEST
ICPHREG
IRT
KDE
KRIGE2D
LIFEREG
LIFETEST
LOESS
LOGISTIC
MCMC
MDS
MI
MIXED
MULTTEST
NLIN
NPAR1WAY
ORTHOREG
PHREG
PLM
PLS
POWER
PRINCOMP
PRINQUAL
PROBIT
PSMATCH
QUANTLIFE
QUANTREG
QUANTSELECT
REG
RMSTREG
ROBUSTREG
RSREG
SEQDESIGN
SEQTEST
SIM2D
SPP
STDRATE
SURVEYFREQ
SURVEYLOGISTIC
SURVEYMEANS
SURVEYPHREG
SURVEYREG
TPSPLINE
TRANSREG
TTEST
VARCLUS
VARIOGRAM

Base SAS

CORR
FREQ
UNIVARIATE

SAS/QC

ANOM
CAPABILITY
CUSUM
MACONTROL
MVPDIAGNOSE
MVPMONITOR
MVPMODEL
PARETO
RAREEVENTS
RELIABILITY
SHEWHART

Other

HPF
HPFENGINE

SAS Risk Dimensions

SAS/ETS

ARIMA
AUTOREG
COPULA
COUNTREG
ENTROPY
ESM
EXPAND
HPCDM
HPQLIM
HPSEVERITY
MODEL
PANEL
PDLREG
QLIM
SEVERITY
SIMILARITY
SSM
SYSLIN
TIMEDATA
TIMEID
TIMESERIES
TMODEL
UCM
VARMAX
X12
X13

For information about the specific graphs that a particular procedure creates, see the PLOTS= option syntax and the “ODS Graphics” section in the corresponding procedure chapter. For the SAS/STAT procedures, the procedure names in the preceding table link to the “ODS Graphics” section.

Procedures That Support ODS Graphics and Traditional Graphics

A number of procedures that support ODS Graphics produced traditional graphics in previous releases of SAS. These include the UNIVARIATE procedure in Base SAS software; the LIFEREG, LIFETEST, and REG procedures in SAS/STAT software; and the ANOM, CAPABILITY, CUSUM, MACONTROL, PARETO, RELIABILITY, and SHEWHART procedures in SAS/QC software. All these procedures continue to produce traditional graphics, but in some cases they do so only when ODS Graphics is not enabled. For more information about the interaction between traditional graphics and ODS graphics in other procedures, see the documentation for those procedures.

Traditional graphs are saved in SAS graphics catalogs and are controlled by the GOPTIONS statement. In contrast, ODS Graphics produces graphs in standard image file formats (not graphics catalogs), and their appearance and layout are controlled by ODS styles and templates.

Syntax

The following sections document some of the most commonly used options in the ODS GRAPHICS statement (see the section "ODS GRAPHICS Statement" on page 695) and other ODS Graphics statements (see the section "ODS Destination Statements" on page 698). You can find the complete syntax in the *SAS Output Delivery System: User's Guide*. In addition, information about the PLOTS= option is provided in the section "PLOTS= Option" on page 699. All statistical procedures that produce ODS Graphics have a PLOTS= option that selects graphs and controls some aspects of the graphs.

ODS GRAPHICS Statement

ODS GRAPHICS < **OFF** | **ON** > < / *options* > **;**

The ODS GRAPHICS statement enables ODS to create graphs. You can enable ODS Graphics by using either of the following equivalent statements:

```
ods graphics on;
ods graphics;
```

You specify one of these statements before invoking your procedure, as illustrated in the examples beginning in the section "Default Plots for Simple Linear Regression with PROC REG" on page 669. Any procedure that supports ODS Graphics then produces graphs, either by default or when you specify procedure options to request particular graphs.

To disable ODS Graphics, specify the following statement:

```
ods graphics off;
```

ODS Graphics might or might not be enabled by default, depending on your operating system, whether you are in the SAS windowing environment, your registry, your system options, and your configuration file settings. For more information about default settings and enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687.

The following is a subset of the options, syntax, and capabilities available in the ODS GRAPHICS statement. For more information, see the *SAS Output Delivery System: User's Guide*.

ANTIALIAS=ON | OFF

controls the use of antialiasing to smooth the components of a graph. Without antialiasing, pixels are simply set or not set. With antialiasing, pixels at the edge of a line or other object are set to an intermediate color, which makes smoother and more professional-looking graphics. Text that is displayed in a graph is always antialiased. Antialiasing is very time-consuming for larger graphs, and its benefits decrease as the number of points increases, so it is disabled by default for plots that have many points. If the number of graph elements exceeds the ANTIALIASMAX= threshold (4,000 by default), then antialiasing is not used, even if you specify ANTIALIAS=ON. By default, ANTIALIAS=ON.

ANTIALIASMAX=*n*

specifies the maximum number of markers or lines to be antialiased before antialiasing is disabled. For example, if there are more than 4,000 markers and ANTIALIASMAX=4,000 (the default), then no markers are antialiased.

ATTRPRIORITY=COLOR | NONE

specifies the priority for cycling group attributes.

COLOR cycles through the lists of colors while holding the marker symbol and line pattern constant. When all the colors are exhausted, the marker symbol and line style attributes increment to the next element, and then the colors in the list are repeated.

NONE simultaneously cycles through colors, marker symbols, and line patterns.

You can specify ATTRPRIORITY=COLOR or ATTRPRIORITY=NONE in the ODS GRAPHICS statement to change the attribute priority for all graphs. You can specify ATTRPRIORITY=COLOR or ATTRPRIORITY=NONE in the BEGINGRAPH statement in the GTL to change the attribute priority for individual graphs. The default attribute priority is set by the ATTRPRIORITY='Color' or ATTRPRIORITY='None' option in the CLASS GRAPH statement in each style.

BORDER=ON | OFF

specifies whether to draw the graph with a border. By default, BORDER=ON.

BYLINE=FOOTNOTE | TITLE | NOBYLINE

specifies how the BY-group line is displayed in graphs when the analysis contains a BY statement. By default, a BY line is not displayed. Specify BYLINE=FOOTNOTE (recommended) to display the BY line as a left-justified graph footnote or BYLINE=TITLE (not recommended) to display the BY line as a centered graph title. Most graph templates control the placement of the BY line as follows:

```
if (_BYTITLE_)
   entrytitle _BYLINE_ / textattrs=GraphValueText;
else
   if (_BYFOOTNOTE_)
      entryfootnote halign=left _BYLINE_;
   endif;
endif;
```

You can modify the graph template if you want the BY line to be displayed in some other way. Because most graphs have titles and few graphs have footnotes, the BY line looks better when it is displayed as a footnote.

HEIGHT=*dimension*

specifies the height of the graph. By default, HEIGHT=480PX (480 pixels). You can also specify height in inches (for example, HEIGHT=5IN) or centimeters (for example, HEIGHT=12CM).

IMAGEMAP=ON | OFF

controls tooltip generation in the HTML destination. By default, IMAGEMAP=OFF, which means that no tooltips are generated. Tooltips are text boxes that appear in HTML output when you rest your mouse pointer over a part of the plot (see Example 24.1).

IMAGENAME=< *base-file-name* **>**

specifies the base image file name. The default is the name of the output object. You can determine the name of the output object by using the ODS TRACE statement (see the section "Determining Graph Names and Labels" on page 703). The base image file name should not include an extension. ODS automatically adds the increment value and the appropriate extension (which is specific to the output destination). See the section "Specifying Base File Names" on page 710 for an example.

LABELMAX=*n*

specifies the maximum number of labeled areas before labeling is disabled. For example, if LABELMAX=50, and there are more than 50 points that have labels, then no points are labeled. By default, LABELMAX=200.

MAXLEGENDAREA=*n*

specifies the maximum percentage of the overall graph area that a legend can occupy. By default, MAXLEGENDAREA=20. Larger legends are dropped from the display.

OUTPUTFMT=< *image-file-type* **| STATIC >**

specifies the image format for graphs. The OUTPUTFMT= option was previously named the IMAGEFMT= option. By default, OUTPUTFMT=STATIC, and ODS dynamically uses the best-quality static image format for the active output destination. The available image formats include BMP (Microsoft Windows device-independent bitmap), DIB (Microsoft Windows device-independent bitmap), EMF (Microsoft NT enhanced metafile), EPSI (Adobe encapsulated PostScript interchange), GIF (graphic interchange format), JFIF (JPEG file interchange format), JPEG (Joint Photographic Experts Group format), PBM (portable bitmap), PCD (Photo CD), PCL (Printer Command Language), PDF (portable document format), PICT (QuickDraw picture format), PNG (Portable Network Graphics), PS (PostScript image file format), SVG (Scalable Vector Graphics format), TIFF (Tagged Image File Format), WMF (Microsoft Windows Metafile format), XBM (X Bitmap), and XPM (X-Windows Pixelmap). If the specified image format is not valid for the active output destination, the device is automatically remapped to the default image format.

RESET< =*option* **>**

resets one or more ODS GRAPHICS options to their default settings. RESET and RESET=ALL are equivalent. If you want to reset more than one option, but not all the options, then you must specify RESET= separately for each option that you reset (for example, `ods graphics on / reset=antialias reset=index;`). The RESET= *options* include the following:

ALL	resets all resettable options to their defaults.
ANTIALIAS	resets the ANTIALIAS= option to its default.
ANTIALIASMAX	resets the ANTIALIASMAX= option to its default.
BORDER	resets the BORDER= option to its default.
INDEX	resets the index counter that is appended to static image files.

HEIGHT	resets the HEIGHT= option to its default.
IMAGEMAP	resets the IMAGEMAP= option to its default.
LABELMAX	resets the LABELMAX= option to its default.
SCALE	resets the SCALE= option to its default.
TIPMAX	resets the TIPMAX= option to its default.
WIDTH	resets the WIDTH= option to its default.

SCALE=ON | OFF
specifies whether the fonts and symbol markers are scaled proportionally to the size of the graph. By default, SCALE=ON. For examples, see Figure 24.69 and Figure 24.70.

SCALEMARKERS=ON | OFF
specifies whether markers are scaled in nested layouts. By default, SCALEMARKERS=ON.

TIPMAX=*n*
specifies the maximum number of distinct tooltips that are permitted before tooltips are disabled. Tooltips are text boxes that appear when you rest your mouse pointer over part of the plot. For example, if TIPMAX=400, and there are more than 400 points in a scatter plot, then no tooltips appear. By default, TIPMAX=500.

WIDTH=*dimension*
specifies the width of the graph. By default, WIDTH=640PX (640 pixels). You can also specify width in inches (for example, WIDTH=5IN) or centimeters (for example, WIDTH=12CM).

ODS Destination Statements

ODS has a number of statements that control the destination of ODS output. The ODS destination statements that are most commonly used in ODS Graphics are ODS DOCUMENT, ODS HTML, ODS LISTING, ODS PCL, ODS PDF, ODS PS, and ODS RTF. Specifying a statement opens a destination, unless the CLOSE option is specified. Each of the following statements opens an ODS destination:

```
ods html;
ods rtf;
ods html image_dpi=300;
ods listing style=HTMLBlue;
```

Each of the following statements closes an ODS destination:

```
ods html close;
ods rtf close;
ods listing close;
```

The following statement closes all open destinations:

```
ods _all_ close;
```

The following two options are commonly used in ODS destination statements to control aspects of ODS Graphics:

IMAGE_DPI=*dpi*
specifies the image resolution of graphical output, measured in number of dots per inch (DPI). The default varies depending on the destination. For example, the default is 96 for HTML and 200 for RTF.

STYLE=*style-name*
specifies the output style. Commonly used styles include HTMLBLUE, HTMLBLUECML, PEARL, PEARLJ, SAPPHIRE, DEFAULT, LISTING, STATISTICAL, JOURNAL, JOURNAL2, JOURNAL3, RTF, and ANALYSIS.

Other options provide ways to control the files that you create. For example, the following statement opens the HTML destination:

```
ods html body='b.html' contents='c.html' frame='a.html';
```

This statement also writes the body of the output to the file *b.html*, the table of contents to the file *c.html*, and an overall frame that contains both the contents and the output to the file *a.html*. Alternatively, you can specify FILE= instead of BODY=.

If you are using a destination for which individual graphs are created (for example, LISTING or HTML), you can use the GPATH= option to specify the directory where your graphics files are saved, as in the following example:

```
ods html gpath="C:\figures";
```

For more information about individual image files and options specified in the ODS destination statements, see the sections "Image File Types" on page 707, "Saving Graphic Image Files" on page 711, "LISTING Destination" on page 711, and "HTML Destination" on page 712. For complete information about the ODS destination statements, see the *SAS Output Delivery System: User's Guide*.

PLOTS= Option

Each statistical procedure that supports ODS Graphics has a PLOTS= option that you use to select graphs and specify some options. The syntax of the PLOTS= option is as follows:

PLOTS < (*global-plot-options*) > < = *plot-request*< (*options*) > > ;

PLOTS < (*global-plot-options*) > < =(*plot-request*< (*options*) > < . . . *plot-request*< (*options*) > >) > ;

The PLOTS= option has a common overall syntax for all statistical procedures, but the specific *global-plot-options*, *plot-requests*, and *plot-options* vary across procedures. This section discusses only a few of the options available in the PLOTS= option. For more information about the PLOTS= option, see the "Syntax" section for each procedure that produces ODS Graphics. There are only a limited number of details that you can control by using the PLOTS= option. Most graphical details are controlled either by graph templates (see the section "Graph Templates" on page 844 in Chapter 25, "ODS Graphics Template Modification") or by styles (see the section "ODS Styles" on page 717).

The PLOTS= option is usually specified in the PROC statement. However, for some procedures, certain analyses and hence certain plots can appear only if an additional statement is specified. These procedures might have a PLOTS= option in that other statement. For example, the PHREG procedure has a PLOTS= option in the BAYES statement, which is used to perform a Bayesian analysis. For more information, see the "Syntax" section of each procedure chapter. The following examples illustrate the syntax of the PLOTS= option:

```
plots=all
plots=none
plots=residuals
plots=residuals(smooth)
plots=(trace autocorr)
plots(unpack)
plots(unpack)=diagnostics
plots=diagnostics(unpack)
plots(only)=freqplot
plots=(scree(unpack) loadings(plotref) preloadings(flip))
plots(unpack maxparmlabel=0 stepaxis=number)=coefficients
plots(sigonly)=(rawprob adjusted(unpack))
```

Also see the "Getting Started" sections "Survival Estimate Plot with PROC LIFETEST" on page 672, "Contour and Surface Plots with PROC KDE" on page 673, "Contour Plots with PROC KRIGE2D" on page 675, "LS-Means Diffogram with PROC GLIMMIX" on page 680, and "Principal Component Analysis Plots with PROC PRINCOMP" on page 682 for examples of using the PLOTS= option.

The simplest PLOTS= specifications are of the form PLOTS=*plot-request* or PLOTS=(*plot-requests*). When there is more than one *plot-request*, the *plot-request* list must appear in parentheses. Each *plot-request* either requests a plot (for example, RESIDUALS) or provides a place to specify plot-specific options (for example, DIAGNOSTICS(UNPACK)). Some simple and typical *plot-requests* are explained next:

- PLOTS=ALL requests all plots that are relevant to the analysis. This does not mean that the procedure produces all plots that it can produce. Plots that are produced for one set of options might not appear when you specify PLOTS=ALL and a different set of options. In some cases, certain plots are not produced unless certain options or statements outside the PLOTS= option are specified.
- PLOTS=NONE disables ODS Graphics for just that step. You can use this option instead of specifying ODS GRAPHICS OFF before a procedure step and ODS GRAPHICS ON after the step when you want to suppress graphics for only that step.
- PLOTS=RESIDUALS requests a plot of residuals in a modeling procedure such as PROC REG.
- PLOTS=RESIDUALS(SMOOTH) requests the residuals plot along with a smooth fit function.
- PLOTS=(TRACE AUTOCORR) requests trace and autocorrelation plots in procedures that have Bayesian analysis options.

Global-plot-options appear in parentheses after the option name and before the equal sign. These options affect many or all of the plots. The UNPACK option is a commonly used *global-plot-option*. It specifies that plots that are usually produced containing multiple plots per panel (or "packed") should be unpacked so they appear in multiple panels that each contain one plot. The specification PLOTS(UNPACK)=(*plot-requests*) unpacks all paneled plots. The UNPACK option is also used as an option in a *plot-request* when you want to unpack only certain panels. For example, the option PLOTS=(DIAGNOSTICS(UNPACK) PARTIAL PREDICTIONS) unpacks only the diagnostics panel. In some cases, unpacked plots contain additional information that is not found in the smaller packed versions. The UNPACK option is not available for all *plot-requests*; it is available only for plots that have multiple panels by default.

Another commonly used *global-plot-option* is the ONLY option. Many procedures produce default plots, and additional plots can be requested in the PLOTS= option. Specifying PLOTS=(*plot-requests*) while omitting

the default plots does not prevent the default plots from being produced. You use the ONLY option when you want to see only the plots that are specifically cited in the *plot-request* list. Procedures that produce no default plots usually do not provide an ONLY option. You can use ODS SELECT and ODS EXCLUDE (see the section "Selecting and Excluding Graphs" on page 706) to select and exclude graphs, but in some situations the ONLY option is more convenient. It is usually more efficient to select plots by using the PLOTS(ONLY)= option, because the procedure does not do extra work to generate a plot that is excluded by the PLOTS(ONLY)= option. In contrast, ODS SELECT and ODS EXCLUDE have their effect after the procedure does the work to generate the plot.

Selecting and Viewing Graphs

This section describes techniques for selecting and viewing your graphs. Topics include the following:

- specifying an ODS destination for graphics
- viewing your graphs in the SAS windowing environment
- referring to graphs by name when using ODS
- selecting and excluding graphs from your output

Specifying an ODS Destination for Graphics

If you do not specify an ODS destination, then either the LISTING or HTML destination is used by default. Here is an example of how you can explicitly specify the HTML destination:

```
ods graphics on;
ods html;

proc reg data=sashelp.class;
   model Weight = Height;
quit;

ods html close;
```

This ODS HTML statement creates an HTML file that has a default name. For information about specifying a file name, see the section "Specifying a File for ODS Output" on page 702. Other destinations are specified in a similar way. For example, you can specify an RTF destination by using the following statements:

```
ods graphics on;
ods rtf;

. . .

ods rtf close;
```

The destinations that ODS supports for graphics are as follows:

Destination	Destination Family
DOCUMENT	
HTML	MARKUP
LATEX*	MARKUP
LISTING	
PCL	PRINTER
PDF	PRINTER
PS	PRINTER
RTF	

* The LATEX destination is experimental.

You can close all open destinations if you are interested only in displaying your output in a nondefault destination. For example, if you want to see your output only in the RTF destination, you can specify the following statements:

```
ods graphics on;
ods _all_ close;
ods rtf;

. . .

ods rtf close;
ods listing;
```

Closing unneeded destinations makes your jobs run faster and creates fewer files. More generally, it makes your jobs consume fewer resources, because a graph is created for every open destination. The last statement opens the LISTING destination after you are finished using the RTF destination.

You can also use the ODS OUTPUT destination to create an output data set from the data object that is used to make a plot. Here is an example:

```
ods graphics on;

proc reg data=sashelp.class;
   ods output fitplot=myfitplot;
   model Weight = Height;
quit;
```

Specifying a File for ODS Output

You can specify a file name for your output by using the FILE= option in the ODS destination statement, as in the following example:

```
ods html file="test.htm";
```

The output is written to the file *test.htm*, which is saved in the SAS current folder. At start-up, the SAS current folder is the same directory in which you started your SAS session. If you are using the SAS windowing environment, then the current folder is displayed in the status line at the bottom of the main SAS window. If you do not specify a file name for your output, then the SAS System provides a default file name, which depends on the ODS destination. This file is saved in the SAS current folder. You can always check the SAS log to verify the name of the file in which your output is saved. For example, suppose you specify the following statement:

```
ods html;
```

Then the following message is displayed in the SAS log:

```
NOTE: Writing HTML Body file: sashtml.htm
```

The default file names for each destination are specified in the SAS Registry. For example, Figure 24.71 shows that the default file name in the SAS Registry for the RTF destination is *sasrtf.rtf*. For more information, see the *SAS Companion* for your operating system.

Viewing Your Graphs in the SAS Windowing Environment

The mechanism for viewing graphs that are created by ODS can vary depending on your operating system, which viewers are installed on your computer, and the ODS destination that you select. If you do not specify an ODS destination, then the default destination is either HTML or LISTING.

If you are using the SAS windowing environment and the HTML destination, then the results are displayed by default in the SAS Results Viewer unless you are using an external browser. To use an external viewer, select **Tools ► Options ► Preferences** from the main SAS window. Then click the **Results** and **Web** tabs to make your selection.

If you are using the PS destination, you must use a PostScript viewer, such as GSview. For information about the windowing environment in a different operating system, see the *SAS Companion* for that operating system.

If you do not want to view the results as they are being generated, then select **Tools ► Options ► Preferences** from the main SAS window. Then on the **Results** tab, clear the **View results as they are generated** check box.

If you are using the SAS windowing environment and the LISTING destination, go to the Results window and find the icon for the corresponding graph. You can double-click the graph icon to display the graph in the default viewer that is configured on your computer for the corresponding image file type (see Figure 24.14).

Determining Graph Names and Labels

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas`.)

Procedures assign a name to each graph that they create using ODS Graphics. This name enables you to refer to ODS graphs in the same way that you refer to ODS tables (see the section "The ODS Statement" on page 599 in Chapter 23, "Using the Output Delivery System"). You can determine the names of graphs in several ways:

- You can look up graph names in the "ODS Graphics" section of chapters for procedures that use ODS Graphics. For example, see the section "ODS Graphics" on page 8852 in Chapter 105, "The REG Procedure."

- You can use the Results window to view the names of ODS graphs that are created in your SAS session. For more information, see the section "The SAS Results Window" on page 604 in Chapter 23, "Using the Output Delivery System."
- You can use the ODS TRACE ON statement to list the names of graphs that are created during your SAS session. This statement adds identifying information in the SAS log (or optionally in the SAS LISTING) for each graph that is produced. For more information, see the section "The ODS Statement" on page 599 in Chapter 23, "Using the Output Delivery System."

The graph name is not the same as the name of the image file that contains the graph (see the section "Naming Graphic Image Files" on page 709).

This example revisits the analysis that is described in the section "Contour and Surface Plots with PROC KDE" on page 673. To determine which output objects are created by ODS, you specify the ODS TRACE ON statement before the procedure statements as follows:

```
ods graphics on;
ods trace on;

proc kde data=bivnormal;
   bivar x y / plots=contour surface;
run;

ods trace off;
```

The trace record from the SAS log is as follows:

```
Output Added:
-------------
Name:       Inputs
Template:   Stat.KDE.Inputs
Path:       KDE.Bivar1.x_y.Inputs
-------------

Output Added:
-------------
Name:       Controls
Template:   Stat.KDE.Controls
Path:       KDE.Bivar1.x_y.Controls
-------------

Output Added:
-------------
Name:       ContourPlot
Label:      Contour Plot
Template:   Stat.KDE.Graphics.Contour
Path:       KDE.Bivar1.x_y.ContourPlot
-------------

Output Added:
-------------
Name:       SurfacePlot
Label:      Density Surface
```

```
Template:   Stat.KDE.Graphics.Surface
Path:       KDE.Bivar1.x_y.SurfacePlot
-------------
```

By default, PROC KDE creates table objects named Inputs and Controls, and it creates graph objects named ContourPlot and SurfacePlot. In addition to the name, the trace record provides the label, template, and path for each output object. Graph templates are distinguished from table templates by a naming convention that uses the procedure name in the second level and **Graphics** in the third level. For example, the fully qualified template name for the surface plot that PROC KDE creates is **Stat.KDE.Graphics.SurfacePlot**.

You can specify the LISTING option in the ODS TRACE ON statement to write the trace record to the LISTING destination as follows:

```
ods trace on / listing;
```

Each table and graph has a path (or name path), which was previously shown in the trace output. The path consists of the plot name, preceded by the names of one or more output groups. Each table and graph also has a label path, which can be seen by adding the LABEL option to the ODS TRACE ON statement, after a slash, as follows:

```
ods trace on / label;

proc kde data=bivnormal;
   bivar x y / plots=contour surface;
run;

ods trace off;
```

A portion of the trace output is shown next:

```
Path:       KDE.Bivar1.x_y.Inputs
Label Path: 'The KDE Procedure'.'Bivariate Analysis'.'x and y'.'KDE.Bivar1.x_y'

Path:       KDE.Bivar1.x_y.Controls
Label Path: 'The KDE Procedure'.'Bivariate Analysis'.'x and y'.'KDE.Bivar1.x_y'

Path:       KDE.Bivar1.x_y.ContourPlot
Label Path: 'The KDE Procedure'.'Bivariate Analysis'.'x and y'.'Contour Plot'

Path:       KDE.Bivar1.x_y.SurfacePlot
Label Path: 'The KDE Procedure'.'Bivariate Analysis'.'x and y'.'Density Surface'
```

The label path contains the information that you see in the HTML table of contents. Names are fixed (they do not vary and they are not data- or context-dependent). In contrast, labels often reflect data- or context-dependent information.

Selecting and Excluding Graphs

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

You can use the ODS SELECT and ODS EXCLUDE statements along with graph and table names to specify which ODS outputs to display. For more information about how to use these statements, see the section "The ODS Statement" on page 599 in Chapter 23, "Using the Output Delivery System."

This section shows several examples of selecting and excluding graphs by using the data set and trace output that are created in the section "Determining Graph Names and Labels" on page 703. The following statements use the ODS SELECT statement to select only two graphs, ContourPlot and SurfacePlot, for display in the output:

```
proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;
```

Equivalently, the following statements use the ODS EXCLUDE statement to exclude the two tables:

```
proc kde data=bivnormal;
   ods exclude Inputs Controls;
   bivar x y / plots=contour surface;
run;
```

You can select or exclude graphs by using either the name or the label. Labels must be specified in quotes. In the context of this example, the following two statements are equivalent:

```
ods select contourplot;
ods select 'Contour Plot';
```

You can also specify multiple levels of the path, as in the following examples:

```
ods select x_y.contourplot;
ods select 'x and y'.'Contour Plot';
ods select 'x and y'.contourplot;
ods select x_y.'Contour Plot';
```

You can mix name and label paths, as in the last two statements. All four of the preceding statements select the same plot. Furthermore, selection based directly on the names and labels is case-insensitive. The following statements all select the same plot:

```
ods select x_y.contourplot;
ods select 'x and y'.'Contour Plot';
ods select X_Y.CONTOURPLOT;
ods select 'X AND Y'.'CONTOUR PLOT';
```

It is sometimes useful to specify a WHERE clause in an ODS SELECT or ODS EXCLUDE statement. This enables you to specify expressions based on either the name path or the label path. You can base your selection on two automatic variables, _path_ and _label_. The following two statements select every object whose path contains the string 'plot' and every object whose label path contains the string 'plot', respectively, ignoring the case in the name and label:

```
ods select where = (lowcase(_path_)  ? 'plot');
ods select where = (lowcase(_label_) ? 'plot');
```

The question mark operator means that the second expression (the string 'plot') is contained in the first expression (the lowercase version of the name or label). For example, all the following names match 'plot' in the WHERE clause: plot, SurfacePlot, SURFACEPLOT, FitPlot, pLoTtInG, Splotch, and so on. Because WHERE clause selection is based on SAS string comparisons, selection is case-sensitive. The LOWCASE function is used to ensure a match even when the case of the specified string does not match the case of the actual name or label.

WHERE clauses are particularly useful when you want to select all the objects in a group. A group is a level of the name path or label path hierarchy before the last level. In the following step, all the objects whose name path contains 'DiagnosticPlots' are selected:

```
proc reg data=sashelp.class plots(unpack);
   ods select where = (_path_ ? 'DiagnosticPlots');
   model Weight = Height;
quit;
```

These are the plots that come from unpacking the PROC REG diagnostics panel of plots. All are in a group named 'DiagnosticPlots'.

Graphic Image Files

Accessing your graphs as individual image files is useful when you want to include them in various types of documents. The default image file type depends on the ODS destination, but you can specify other supported image file types. You can also specify the names for your graphic image files and the directory in which you want to save them. This section describes the image file types that ODS Graphics supports, and it explains how to name and save graphic image files.

Image File Types

If you are using the LISTING or HTML destination, your graphs are individually produced in a specific image file type, such as PNG. If you are using a destination in the PRINTER family or the RTF destination, the graphs are contained in the ODS output file and cannot be accessed as individual image files. However, you can open an RTF output file in Microsoft Word and then copy and paste the graphs into another document, such as a Microsoft PowerPoint presentation. This is illustrated in Example 24.2.

Table 24.2 shows the various ODS destinations that ODS Graphics supports, the viewer that is appropriate for displaying graphs in each destination, and the image file types that each destination supports.

Table 24.2 Destinations and Image File Types Supported by ODS Graphics

Destination	Destination Family	Recommended Viewer	Image File Types
DOCUMENT		Not applicable	Not applicable
HTML	MARKUP	Web browser	PNG (default), GIF, JPEG
LATEX	MARKUP	PostScript or PDF viewer after the LaTeX file is compiled	PostScript (default), EPSI, GIF, JPEG, PDF, PNG
LISTING		Default viewer in your system for the specified file type	PNG (default), BMP, DIB, EMF, EPSI, GIF, JFIF, JPEG, PBM, PS, TIFF, WMF
PCL	PRINTER	Not applicable	Contained in PRN file
PDF	PRINTER	PDF viewer, such as Adobe Reader	Contained in PDF file
PS	PRINTER	PostScript viewer, such as GSview	Contained in PostScript file
RTF		Word processor, such as Microsoft Word	Contained in RTF file

For destinations such as PDF and RTF, you can control the types of the images that the file contains even though individual files are not made for each image. The default image file type is PNG, and other image types are available. For more information, see the *SAS Output Delivery System: User's Guide*.

Scalable Vector Graphics

Scalable vector graphics output is supported in ODS Graphics. The output type support depends on the ODS destination that you use. You can specify the OUTPUTFMT= option in the ODS GRAPHICS statement to specify the output type for any destination. For destinations that generate vector graphics by default, you can get image output by specifying OUTPUTFMT=STATIC.

Vector graphics are not supported for all graph types. When vector graphics are requested but not supported, the graph automatically changes to image output. Vector graphics are not supported for the following graph types:

- three-dimensional graph
- contour plots that have smooth gradient fills
- graphs that have continuous legends
- graphs that have data skins (such as bars charts that appear to be shiny or have depth)
- graphs that have rotated annotation images
- graphs that have transparency (EMF and PS only)

The LISTING destination can generate all the supported forms of vector-based output: PDF, PS, EMF, SVG, and PCL. Each graph is generated in a separate file that can be included in a larger report. The default output format is a PNG image.

Like the LISTING destination, the ODS PRINTER destination can generate all the supported vector output types. The output format depends on the type of printer that you select. If you select the PDF, SVG, or PCL5C printer, vector-based output is automatically produced. However, if you select the PS or EMF printer, you need to set the OUTPUTFMT= option in the ODS GRAPHICS statement to PS or EMF, respectively, to

create vector-based output. By default, the output from this destination is contained in one file instead of individual files for each graph.

The PDF destination produces PDF vector output by default, except for the exceptions noted. You can specify OUTPUTFMT=STATIC in the ODS GRAPHICS statement to produce an embedded image in the PDF file.

The experimental LATEX destination produces PS vector output by default, except for the exceptions noted. You can specify OUTPUTFMT=STATIC in the ODS GRAPHICS statement to produce an embedded image in the PostScript file.

The RTF destination produces PNG image output by default. It also supports vector-based EMF output for this destination. You can specify OUTPUTFMT=EMF in the ODS GRAPHICS statement to select this output type. If one of the noted exceptions occurs, the output type for that graph changes to a PNG image type.

The HTML destination produces PNG image output by default. It also supports vector-based SVG output for this destination. You can specify OUTPUTFMT=SVG in the ODS GRAPHICS statement to select this output type. If one of the noted exceptions occurs, the output type for that graph changes to a PNG image type.

In most cases, the vector graphics file is much smaller than a comparable static image file. However, in some cases, the vector graphics file is larger than the image version. This is likely for scatter plots of data sets that have a large number of observations.

Naming Graphic Image Files

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

The following discussion applies to the destinations where ODS graphs are created as individual image files (for example, HTML and LISTING). The names of graphic image files are determined by a base file name, an index counter, and an extension. By default, the base file name is the ODS graph name (see the section "Determining Graph Names and Labels" on page 703). There is an index counter for each base file name. The extension indicates the image file type. The first time a graph object that has a particular base file name is created, the file name consists only of the base file name and the extension. If a graph that has the same base file name is created multiple times, then an index counter is appended to the base file name to avoid overwriting previously created images.

To illustrate, consider the following statements:

```
proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;
```

If you run this step at the beginning of a SAS session, the two graphic image files that are created are *ContourPlot.png* and *SurfacePlot.png*. If you immediately rerun these statements, then ODS creates the same graphs in different image files named *ContourPlot1.png* and *SurfacePlot1.png*. The next time, the image files are named *ContourPlot2.png* and *SurfacePlot2.png*. The index starts at 0, and 1 is added each time the same name is used. However, if the index is at 0, then 0 is not included in the file name.

Resetting the Index Counter

You can specify RESET=INDEX in the ODS GRAPHICS statement to reset the index counter. This is useful when you need to have predictable names. It is particularly useful when you are running a SAS program multiple times in the same session. The following statement resets the index:

```
ods graphics on / reset=index;
```

The index counter is reinitialized at the beginning of your SAS session or if you specify RESET=INDEX in the ODS GRAPHICS statement. Graphic image files that have the same name are overwritten.

Specifying Base File Names

You can specify a base file name for all your graphic image files by using the IMAGENAME= option in the ODS GRAPHICS statement as follows:

```
ods graphics on / imagename="MyName";
```

You can also specify RESET=INDEX as follows:

```
ods graphics on / reset=index imagename="MyName";
```

The IMAGENAME= option overrides the default base file name. In the preceding statement, the graphic image files are named *MyName*, *MyName1*, *MyName2*, and so on.

Specifying Image File Types

You can specify the image file type for the LISTING, HTML, or LATEX destination by specifying OUTPUTFMT= in the ODS GRAPHICS statement as follows:

```
ods graphics on / outputfmts=gif;
```

For more information, see the section "ODS GRAPHICS Statement" on page 695.

Naming Graphic Image Files with Multiple Destinations

Because the index counter depends only on the base file name, if you specify multiple ODS destinations for your output, then the index counter is increased independently of the destination. For example, the following statements create image files named *ContourPlot.png* and *SurfacePlot.png* that correspond to the LISTING destination and *ContourPlot1.png* and *SurfacePlot1.png* that correspond to the HTML destination:

```
ods listing;
ods html;
ods graphics on / reset;

proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;

ods _all_ close;
ods listing;
```

When you specify one of the destinations in the PRINTER family or the RTF destination, your ODS graphs are embedded in the document, so the index counter is not affected. For example, the following statements create the image files *ContourPlot.png* and *SurfacePlot.png* for the LISTING destinations but no image files for the RTF destination:

```
ods listing;
ods rtf;
ods graphics on / reset;

proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;

ods _all_ close;
```

Saving Graphic Image Files

Knowing where your graphic image files are saved and how they are named is particularly important if you are running in batch mode, if you have disabled the SAS Results window (see the section "Viewing Your Graphs in the SAS Windowing Environment" on page 703), or if you plan to access the files for inclusion in a paper or presentation. The following discussion assumes that you are running SAS on the Windows operating system. If you are running on a different operating system, see the *SAS Companion* for your operating system.

In the SAS windowing environment, the current folder is displayed in the status line at the bottom of the main SAS window. When the **Use WORK folder** check box is cleared on the **Results** tab (which you access by selecting **Tools ► Options ► Preferences** from the main SAS window), graphic image files are saved in the current folder and available after your SAS session ends. They can accumulate over time and take up a great deal of space. When **Use WORK folder** is selected, graphic image files are stored in the Work folder and are not available after your SAS session ends.

If you are running your SAS programs in batch mode, the graphs are saved by default in the same directory where you started your SAS session. For example, suppose the SAS current folder is *C:\myfiles*. If ODS Graphics is enabled, then your graphic image files are saved in the directory *C:\myfiles*. Traditional graphics are always saved in a catalog in your Work directory.

If you are using the LISTING, HTML, and LATEX destinations, you can specify a directory for saving your graphic image files. If you are using a destination in the PRINTER family or the RTF destination, you can specify a directory only for your output file. The remainder of this discussion provides details about each destination type.

LISTING Destination

If you are using the LISTING destination, the individual graphs are created as PNG files by default. You can use the GPATH= option in the ODS LISTING statement to specify the directory where your graphics files are saved. For example, if you want to save your graphic image files in the *C:\figures* directory, then you can specify the following:

```
ods listing gpath="C:\figures";
```

It is important to note that the GPATH= option applies only to ODS Graphics. It does not affect the behavior of graphs that are created by traditional SAS/GRAPH procedures.

HTML Destination

If you are using the HTML destination, the individual graphs are created as PNG files by default. You can use the PATH= and GPATH= options in the ODS HTML statement to specify the directory where your HTML and graphics files, respectively, are saved. These options also give you more control over your graphs. For example, if you want to save your HTML file named *test.htm* in the *C:\myfiles* directory, but you want to save your graphic image files in *C:\myfiles\png*, then you can specify the following:

```
ods html path  = "C:\myfiles"
         gpath = "C:\myfiles\png"
         file  = "test.htm";
```

When you specify the URL= suboption along with the GPATH= option, SAS creates relative paths for the links and references to the graphic image files in the HTML file. This is useful for building output files that are easily moved from one location to another. For example, the following statements create a relative path to the *png* directory in all the links and references that the file *test.htm* contains:

```
ods html path  = "C:\myfiles"
         gpath = "C:\myfiles\png" (url="png/")
         file  = "test.htm";
```

If you do not specify the URL= suboption, SAS creates absolute paths that are hard-coded in the HTML file. These can cause broken links if you move the files. For more information, see the ODS HTML statement in the *SAS Output Delivery System: User's Guide*.

LATEX Destination (Experimental)

LaTeX is a document preparation system for high-quality typesetting of mathematical and scientific material. The ODS LATEX statement produces output in the form of a LaTeX source file that is ready to compile in LaTeX. When you request ODS Graphics for a LATEX destination,[2] ODS creates the requested graphs as PostScript files by default, and the LaTeX source file includes references to these graphic image files. You can compile the LaTeX file, or you can ignore this file and simply access the individual PostScript files to include your graphs in a different LaTeX document, such as a paper that you are writing. You can specify the PATH= and GPATH= options in the ODS LATEX statement, as explained previously for the ODS HTML statement; see Example 24.3 for an illustration. The ODS LATEX statement is an alias for the ODS MARKUP statement with the TAGSET=LATEX option. For more information, see the *SAS Output Delivery System: User's Guide*.

The default image file type for the LATEX destination is PostScript. When you use LaTeX to compile your document, the graphics format for included images is PostScript. However, if you prefer to use pdfLaTeX, you can specify a different format, such as JPEG, PDF, or PNG, any of which can be directly included in your pdfLaTeX document. To specify one of these formats, you use the OUTPUTFMT= option in the ODS GRAPHICS statement. For more information, see the LaTeX documentation for the `graphicx` package.

[2]The LATEX destination is experimental.

Creating Graphs in Multiple Destinations

This section illustrates how to send your output to more than one destination by using a single execution of your SAS statements. For example, to create LISTING, HTML, and RTF output, you can specify the ODS LISTING, ODS HTML, and ODS RTF statements before your procedure statements. The ODS _ALL_ CLOSE statement closes all open destinations before and after the other statements are run.

```
ods _all_ close;
ods listing;
ods html;
ods rtf;

. . .

ods _all_ close;
```

You can also specify multiple instances of the same destination. For example, using the data in the section "Contour and Surface Plots with PROC KDE" on page 673, the following statements save the contour plot to the file *contour.pdf* and the surface plot to the file *surface.pdf*:

```
ods _all_ close;
ods pdf file="contour.pdf";
ods pdf select ContourPlot;
ods pdf(id=srf) file="surface.pdf";
ods pdf(id=srf) select SurfacePlot;
ods graphics on;

proc kde data=bivnormal;
   ods select ContourPlot SurfacePlot;
   bivar x y / plots=contour surface;
run;

ods _all_ close;
```

The ID= option assigns the name **srf** to the second instance of the PDF destination. Without the ID= option, the second ODS PDF statement closes the destination that was opened by the first ODS PDF statement, and it opens a new instance of the PDF destination. In that case, the file *contour.pdf* is not created. For more information, see the ODS PDF statement in the *SAS Output Delivery System: User's Guide*.

Graph Size and Resolution

ODS provides options for specifying the size and resolution of graphs. You can specify the size of a graph in the ODS GRAPHICS statement and the resolution in an ODS destination statement. There are two other ways to change the size of a graph, but they are rarely needed. The three methods are as follows:

- Usually, you specify the WIDTH= or HEIGHT= option (or both) in the ODS GRAPHICS statement to change the size of a graph.
- You can modify the size of a particular graph by specifying the dimensions in the DESIGNHEIGHT= and DESIGNWIDTH= options in the BEGINGRAPH statement in the template. Some templates contain the specification DESIGNWIDTH=DEFAULTDESIGNHEIGHT, which sets the width of the graph to the default height, or DESIGNHEIGHT=DEFAULTDESIGNWIDTH, which sets the height of the graph to the default width.
- You can modify the size of all your ODS graphs by specifying the dimensions in the OUTPUTHEIGHT= and OUTPUTWIDTH= options in the style template.

The following examples show typical ways to change the size of your graphs:

```
ods graphics on / width=6in;
ods graphics on / height=4in;
ods graphics on / width=4.5in height=3.5in;
```

The dimensions of the graph can be specified in pixels (such as 200PX), inches (such as 3IN), or centimeters (such as 8CM). The default dimensions of ODS Graphics are 640 pixels wide and 480 pixels high, and these values determine the default aspect ratio. The actual size of the graph in inches depends on your printer or display device. For example, if the resolution of your printer is 100 dots per inch (DPI) and you want a graph that is 4 inches wide, you should set the width to 400 pixels.

If you specify only one dimension, the other dimension is determined by the default aspect ratio—that is, height = 0.75 × width. For best results, you should create your graphs by using the exact size that is used to display the graphs in your paper or presentation. In other words, avoid generating graphs at one size and then expanding or shrinking them for inclusion in your document.

By default, fonts and symbol markers are automatically scaled along with the size of the graph. You can suppress this scaling by specifying the SCALE= option, as in the following example:

```
ods graphics on / scale=off;
```

The default resolution of graphs that are created in the HTML and LISTING destinations is 96 DPI, whereas the default for the RTF destination is 200 DPI. The 200 DPI value is recommended if you are copying and pasting graphs into a Microsoft PowerPoint presentation or a Microsoft Word document. Graphs in SAS/STAT documentation are usually generated at 300 DPI for display in PDF and 96 DPI for display in HTML.

You can change the resolution by using the IMAGE_DPI= option in any ODS destination statement, as in the following example:

```
ods html image_dpi=300;
```

An increase in resolution often improves the quality of the graphs, but it also greatly increases the size of the image file. Going from 96 DPI to 300 DPI increases the size of the image file by approximately a factor of $(300/96)^2 = 9.77$. Even when you are using a higher resolution for most of your graphs, you should consider using a lower resolution for some, such as contour plots, that create large files even at a lower resolution.

If you increase the resolution, you might need to compensate by reducing the size of the graph, as in the following example:

```
ods graphics on / width=4.5in  height=3.5in;
```

Increasing resolution also increases the amount of memory that is needed for your program to run. You can increase the amount of memory available to ODS Graphics by specifying an option when you invoke SAS, as in the following example:

```
-jreoptions '(-Xmx256m)'
```

You can modify the default amount of memory available to ODS Graphics by changing JREOPTIONS in your SAS configuration file to the settings –Xmx*nnn*m –Xms*nnn*m, where *nnn* is the amount of memory in megabytes. An example is –Xmx256m –Xms256m. In either case, the exact syntax varies depending on your operating system, and the amount of memory that you can allocate varies from system to system. For more information, see the *SAS Companion* for your operating system.

The Default Template Stores and the Template Search Path

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/odsgrin.sas.)

Compiled templates are stored in a template store, which is a type of item store. (An item store is a special type of SAS file.) You can see the list of template stores by submitting the following statement:

```
ods path show;
```

The results are as follows:

```
Current ODS PATH list is:

1. SASUSER.TEMPLAT(UPDATE)
2. SASHELP.TMPLMST(READ)
```

These results show that the default template search path consists of Sasuser.Templat followed by Sashelp.Tmplmst. You can add template stores that you create, or you can change the order in which the template stores are searched. For more information, see the sections "Saving Customized Templates" on page 853, "Using Customized Templates" on page 854, and "Reverting to the Default Templates" on page 854 in Chapter 25, "ODS Graphics Template Modification," and the sections "The ODS PATH Statement" on

page 606 and "Controlling Output Appearance with Templates" on page 607 in Chapter 23, "Using the Output Delivery System."

This section discusses the default template stores that you use when you have not modified the template search path by using the ODS PATH statement. By default, templates that you write are stored in Sasuser.Templat. If you store a modified template in Sasuser.Templat, ODS finds and uses your modified template. Otherwise, ODS finds the templates that it provides in Sashelp.Tmplmst. You can see a list of all the templates that you have modified as follows:

```
proc template;
   list / store=sasuser.templat;
run;
```

You can delete any template that you modified (so that ODS finds the default SAS template) by specifying it in a DELETE statement, as in the following statement:

```
proc template;
   delete Stat.REG.Graphics.ResidualPlot / store=sasuser.templat;
run;
```

Specifying STORE=SASUSER.TEMPLAT is not required. However, if you have administrator privileges on your computer, this option helps you ensure that you do not accidentally delete templates from Sashelp.Tmplmst. Unless you have administrator privileges, ODS never deletes a template in Sashelp.Tmplmst, so you can safely run the preceding step without the STORE= option, even if the template that you specify does not exist in Sasuser.Templat. You can run the following step to delete the entire Sasuser.Templat store of customized templates so that ODS uses only the templates that SAS supplies:

```
ods path sashelp.tmplmst(read);
proc datasets library=sasuser nolist;
   delete templat(memtype=itemstor);
run;
ods path sasuser.templat(update) sashelp.tmplmst(read);
```

It is good practice to delete templates that you have customized when you are done using them, so that they are not unexpectedly used later. For more information, see the section "Reverting to the Default Templates" on page 854 in Chapter 25, "ODS Graphics Template Modification."

ODS Styles

ODS styles control the overall appearance of your output. Usually, the only thing that you need to do with styles is specify them in an ODS destination statement, as in the following example:

```
ods html body='b.html' style=HTMLBlue;
```

However, you can also modify existing styles and even write your own styles. You can also specify style elements in custom templates that you write, or you can modify which style elements are used in templates that SAS supplies. This section provides an overview of ODS styles and style elements, which are the components of a style. It also describes how to customize a style template and how to specify a default style for your output. Only the most commonly used styles, style elements, and style changes are discussed here. For complete information about ODS styles, see the *SAS Output Delivery System: User's Guide*.

An Overview of ODS Styles

An ODS style template provides formatting information for specific visual aspects of your SAS output (see the section "ODS Style Elements and Attributes" on page 726). The appearance of tables and graphs is coordinated within a particular style. For tables, this information includes a list of fonts and a list of colors. Each font definition specifies a family, size, weight, and style. Colors are associated with common areas of output, including titles, footnotes, BY groups, table headings, and table cells. For graphs, ODS styles also control the appearance of graph elements, including lines, markers, fonts, and colors. ODS styles also include elements specific to statistical graphics, such as the style of fitted lines, confidence bands, and prediction limits. For more information about ODS styles, see Kuhfeld (2016) and the *SAS Output Delivery System: User's Guide*.

You can specify a style by using the STYLE= option in an ODS destination statement such as HTML, PDF, RTF, or PRINTER. You can also specify a style in the LISTING destination; however, this style affects graphs but not tables. Output that is produced by using different styles has the same content but a different appearance. For example, the following statement requests output that is produced by using the JOURNAL style:

```
ods rtf style=Journal;
```

You can use any ODS style or any style that you define yourself. The following statements list the names of all the styles and then display five of them:

```
proc template;
   list styles;
   source Styles.Default;
   source Styles.Statistical;
   source Styles.Journal;
   source Styles.RTF;
   source Styles.HTMLBlue;
run;
```

The results of this step (not shown) include a list of more than 50 styles in the SAS listing and five style templates in the SAS log. Style templates are often hundreds of lines long. For more information about style templates, see the section "Style Templates and Colors" on page 727.

Each ODS destination has its own default style, as shown in Table 24.1 and mentioned throughout this section. Most graphs in SAS/STAT documentation use the HTMLBLUE style. However, throughout this chapter, you can see examples of other styles. For more information about styles, see the *SAS Output Delivery System: User's Guide*.

The rest of this section describes a few ODS styles. There are many more ODS styles than are listed here. Most styles are not designed for statistical work. The following styles are used most often for statistical work.

- The HTMLBLUE style is a modern color style that is recommended for use in web pages or color print media. (See Figure 24.19 for an example.) The HTMLBLUE style is an ATTRPRIORITY='Color' style. The HTMLBLUE style inherits most of its attributes from the STATISTICAL style, which inherits some of its attributes from the DEFAULT style. The HTMLBLUE style has a brighter appearance than its parents, and it has color coordination between the tables and graphs. The dominant color is blue.

  The HTMLBLUE style is the default style for the HTML destination. It is also the default style in SAS/STAT documentation for tables displayed in the HTML format and graphs displayed in both the PDF and HTML formats.[3]

  Output that you create by using the HTMLBLUE style might not print well on black-and-white devices (particularly when you create graphs that have groups of observations). If you need an alternative to the HTMLBLUE style that varies colors, lines, and markers, use the HTMLBLUECML style or most other styles. If you need an alternative to the HTMLBLUE style that is designed for printer destinations such as PRINTER, PDF, PS, and RTF, see the PEARL, PEARLJ, and SAPPHIRE styles.

- The HTMLBLUECML style is a modern color style that is recommended for use in web pages or color print media. (See Figure 24.20 for an example.) It inherits most of its attributes from the HTMLBLUE style. The dominant color is blue. The HTMLBLUECML style is an ATTRPRIORITY='None' style. In graphs, groups of observations are distinguished by simultaneous color, line style, and symbol changes. If you need an alternative to the HTMLBLUECML style that is all color, use the HTMLBLUE style instead. Output that you create by using the HTMLBLUECML style might not print well on black-and-white devices.

- The PEARL style is a modern color style that is recommended for use in documents that are created by using printer destinations such as PRINTER, PDF, PS, and RTF. Graphs are displayed in color, and tables are displayed in black and white. (See Figure 24.26 for an example.) The PEARL style shares most of its attributes with the SAPPHIRE style. Both styles inherit most of their attributes from the HTMLBLUE style; hence the dominant color is blue. The differences are outside the graphs (in the tables and in the page background). The HTMLBLUE style has a very light blue background, and the PEARL style has a white background. The PEARL style also has a white background for row and column table headings, whereas the SAPPHIRE style has a light blue background. The PEARL and SAPPHIRE styles use fonts that are appropriate for printer destinations. The PEARL style is an ATTRPRIORITY='Color' style. In graphs, groups of observations are distinguished by color. Output that you create by using the PEARL style might not print well on black-and-white devices (particularly when you create graphs that have groups of observations).

- The PEARLJ style is a modern style that is recommended for use in documents that are created by using printer destinations such as PRINTER, PDF, PS, and RTF. Graphs are displayed in color, and tables are displayed in black and white. The PEARLJ style inherits all of its graphical attributes and most of its other attributes from the PEARL style. The two styles display tables differently. The

[3]The PEARLJ style is the default style for the PDF tables displayed in SAS/STAT documentation.

PEARLJ style uses smaller fonts and less cell padding, and it displays horizontal but not vertical lines. The PEARLJ style is the default style for tables that are displayed in the PDF version of SAS/STAT documentation. The PEARLJ style is an ATTRPRIORITY='Color' style. In graphs, groups of observations are distinguished by color. Output that you create by using the PEARL style might not print well on black-and-white devices (particularly when you create graphs that have groups of observations).

- The SAPPHIRE style is a modern color style that is recommended for use in documents that are created by using printer destinations such as PRINTER, PDF, PS, and RTF. (See Figure 24.27 for an example.) The SAPPHIRE style shares most of its attributes with the PEARL style. Both styles inherit most of their attributes from the HTMLBLUE style; hence the dominant color is blue. However, unlike the HTMLBLUE style (which has a very light blue background), the SAPPHIRE style has a white background. The SAPPHIRE style has a light blue background for row and column table headings, whereas the PEARL style has a white background. The SAPPHIRE and PEARL styles use fonts that are appropriate for printer destinations. The SAPPHIRE style is an ATTRPRIORITY='Color' style. In graphs, groups of observations are distinguished by color. Output that you create by using the SAPPHIRE style might not print well on black-and-white devices (particularly when you create graphs that have groups of observations).

- The JOURNAL family of styles (JOURNAL, JOURNAL2, and JOURNAL3) consists of black-and-white or grayscale styles that are recommended for graphs that appear in journals and other black-and-white publications. (See Figure 24.23 for an example of the JOURNAL style, see Figure 24.9 for an example of the JOURNAL2 style, and see Example 24.3 for a comparison of the three styles.) The JOURNAL1A, JOURNAL2A, and JOURNAL3A styles inherit most of their attributes from the JOURNAL, JOURNAL2, and JOURNAL3 styles, respectively, but use fewer italic fonts. For color alternatives to the JOURNAL family of styles, see the PEARL, PEARLJ, and SAPPHIRE styles.

- The DEFAULT style is a legacy color style. (See Figure 24.18 for an example.) Most other styles inherit some of their elements from this style. The DEFAULT style was the default style for the HTML destination in earlier SAS releases. The dominant color is gray. The DEFAULT style is an ATTRPRIORITY='None' style. In graphs, groups of observations are distinguished by simultaneous color, line style, and symbol changes. Output that you create by using the DEFAULT style might not print well on black-and-white devices.

- The STATISTICAL style is a legacy color style. (See Figure 24.21 for an example.) The STATISTICAL style inherits elements from the DEFAULT style, and it is similar in some ways (other than color) to the ANALYSIS style. The dominant colors are blue and gray. The STATISTICAL style is an ATTRPRIORITY='None' style. In graphs, groups of observations are distinguished by simultaneous color, line style, and symbol changes. Output that you create by using the STATISTICAL style might not print well on black-and-white devices.

- The ANALYSIS style is a legacy color style. (See Figure 24.22 for an example.) The ANALYSIS style inherits elements from the DEFAULT style, and it is similar in some ways (other than color) to the STATISTICAL style. The dominant colors are green and yellow. The ANALYSIS style is an ATTRPRIORITY='None' style. In graphs, groups of observations are distinguished by simultaneous color, line style, and symbol changes. Output that you create by using the ANALYSIS style might not print well on black-and-white devices.

- The RTF style is a legacy color style that is designed for graphs that will be inserted into a Microsoft Word document or PowerPoint slide. (See Figure 24.25 for an example of the RTF style, which is the

default style for the RTF destination.) The RTF style inherits elements from the DEFAULT style. The dominant color is gray. The ANALYSIS style is an ATTRPRIORITY='None' style. In graphs, groups of observations are distinguished by simultaneous color, line style, and symbol changes. Output that you create by using the RTF style might not print well on black-and-white devices. For alternatives to the RTF style that are brighter and less gray, see the PEARL, PEARLJ, and SAPPHIRE styles.

- The LISTING style is a legacy color style that is similar to the DEFAULT style, but the LISTING style has a lighter background. (See Figure 24.24 for an example.) It is the default style for the LISTING destination. The LISTING style inherits elements from the DEFAULT style. The dominant colors are black and white. The LISTING style is an ATTRPRIORITY='None' style. In graphs, groups of observations are distinguished by simultaneous color, line style, and symbol changes. Output that you create by using the LISTING style might not print well on black-and-white devices.

Attribute Priority and Overriding How Groups Are Distinguished

ODS styles have an ATTRPRIORITY= option with the specification ATTRPRIORITY='Color' or ATTRPRIORITY='None'. The ODS GRAPHICS statement has an ATTRPRIORITY= option with the specification ATTRPRIORITY=COLOR or ATTRPRIORITY=NONE. The default attribute priority for the style option is 'None'. The default attribute priority for the ODS GRAPHICS statement option is the attribute priority from the style. The ODS styles that distinguish groups by color alone are ATTRPRIORITY='Color' styles. The ODS styles that distinguish groups by varying colors, markers, and line patterns are ATTRPRIORITY='None' styles. The ATTRPRIORITY='Color' styles include HTMLBLUE, PEARL, PEARLJ, and SAPPHIRE. Most other styles are ATTRPRIORITY='None' styles. (For more information about styles and attribute priority, see the section "ODS Styles" on page 688.) You can override the ATTRPRIORITY= option in all styles by specifying one of the following statements:

```
ods graphics on / attrpriority=color;
ods graphics on / attrpriority=none;
```

You can reset the default ODS Graphics options as follows:

```
ods graphics on / reset;
```

The following steps illustrate three ways to make graphs and create the graphs in Figure 24.15:

```
ods graphics on / attrpriority=color;
proc sgplot data=sashelp.class;
   title 'Attribute Priority = Color';
   reg y=weight x=height / group=sex;
   keylegend / location=inside position=topleft across=1;
run;

ods graphics on / attrpriority=none;
proc sgplot data=sashelp.class;
   title 'Attribute Priority = None';
   reg y=weight x=height / group=sex;
   keylegend / location=inside position=topleft across=1;
run;

proc sgplot data=sashelp.class;
```

```
   title 'Solid Lines but Varying Markers';
   styleattrs datalinepatterns=(solid);
   reg y=weight x=height / group=sex;
   keylegend / location=inside position=topleft across=1;
run;
ods graphics on / reset;
```

Figure 24.15 Attribute Priority

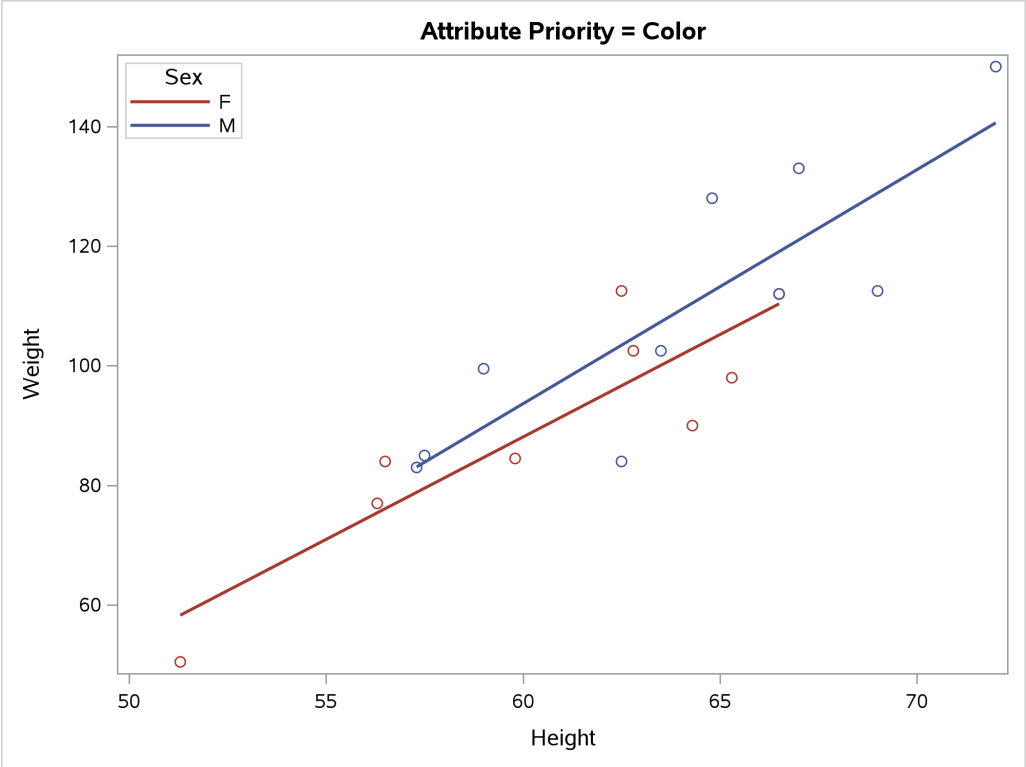

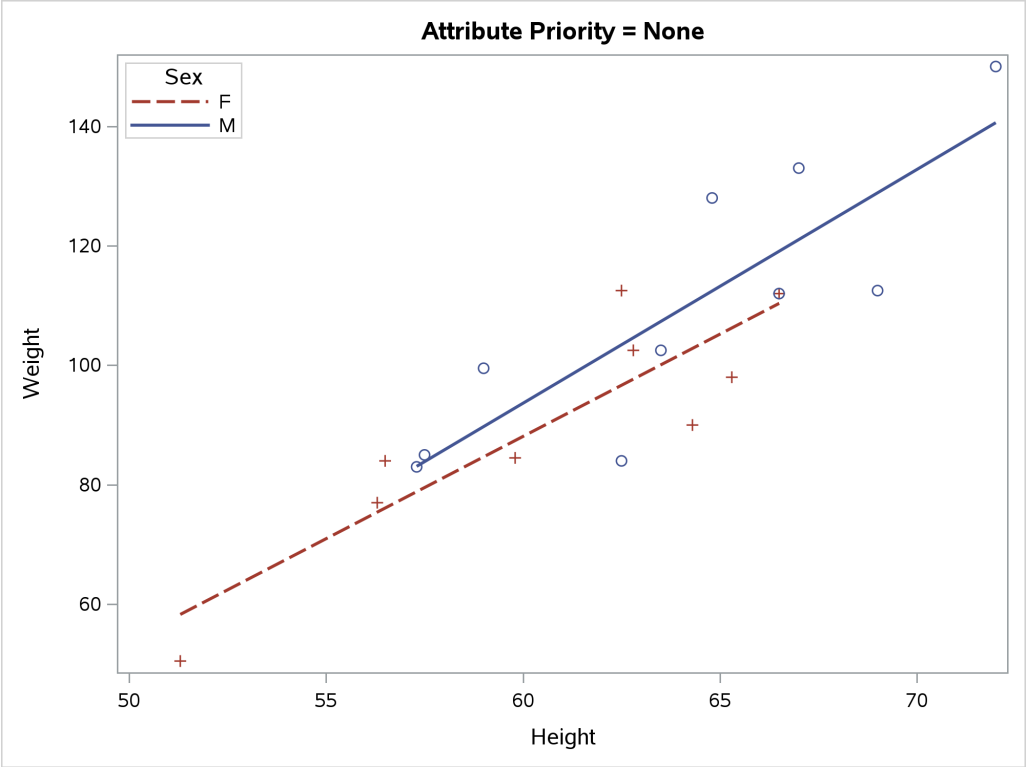

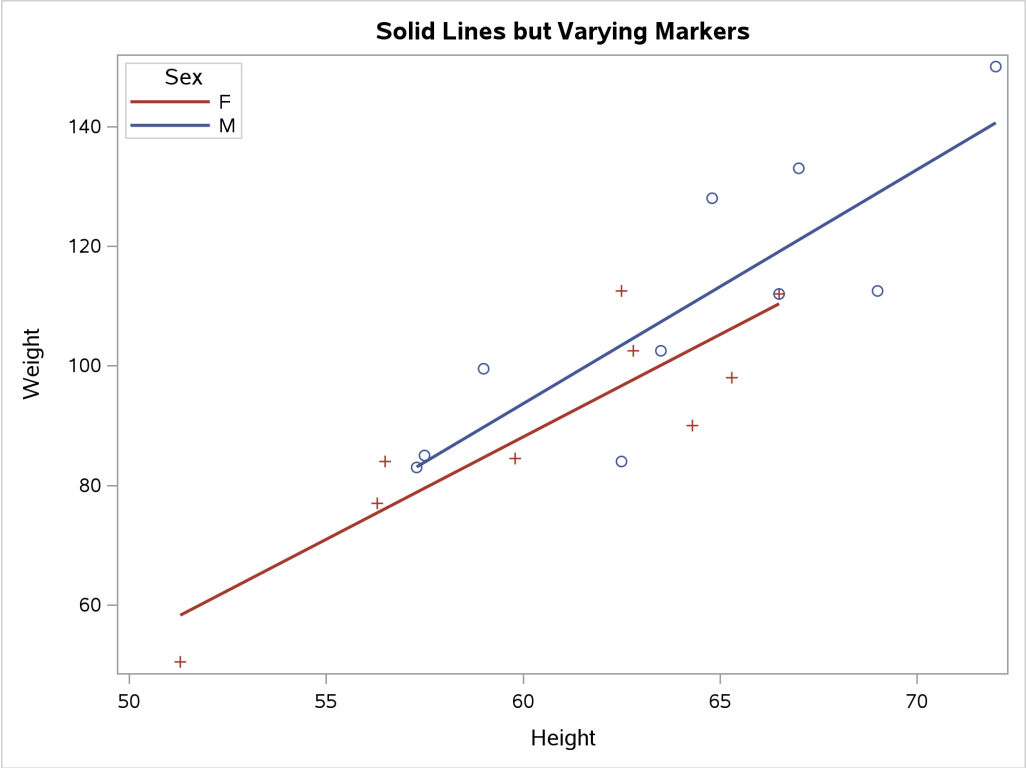

The settings that are shown in the last graph, which has solid lines but varying markers, often work well when there are multiple groups. You might not need to vary the line style, but varying the markers makes it easier

to distinguish the points. For example, it is easy to see in the last two graphs that one point corresponds to both a male and a female point.

You can modify the attribute priority in any style by specifying the ATTRPRIORITY= option, as in the following examples:

```
proc template;
   define style styles.Default;
      parent = styles.default;
      style Graph from Graph / attrpriority = "Color";
   end;
   define style styles.HTMLBlue;
      parent = styles.HTMLBlue;
      style Graph from Graph / attrpriority = "None";
   end;
run;
```

You can delete the modified style templates as follows:

```
proc template;
   delete styles.Default  / store=sasuser.templat;
   delete styles.HTMLBlue / store=sasuser.templat;
run;
```

In the statistical graphics (SG) procedures, you can override the style attributes and modify how you distinguish groups by using the STYLEATTRS statement:

```
ods graphics on / attrpriority=none;

proc sgplot data=sashelp.iris;
   title 'Fisher (1936) Iris Data';
   styleattrs datasymbols=(circlefilled squarefilled starfilled)
              datacontrastcolors=(red green blue);
   scatter x=petallength y=petalwidth / group=species markerattrs=(size=5px);
run;

ods graphics / reset;
```

The results are displayed in Figure 24.16.

Figure 24.16 Iris Data

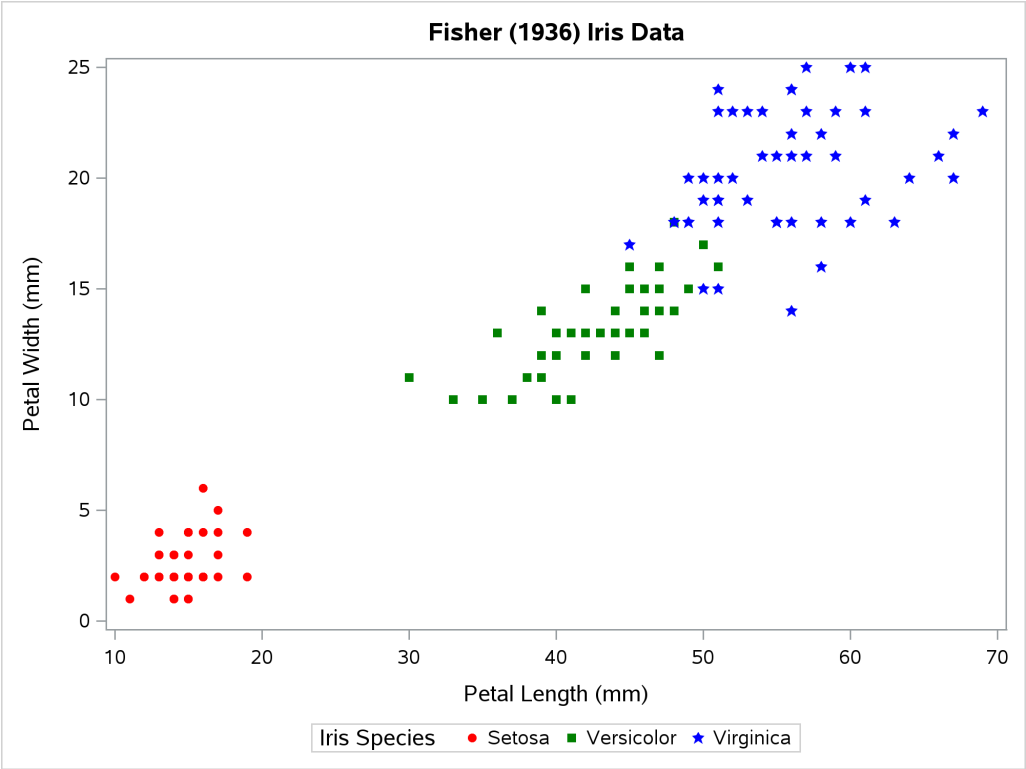

NOTE: Specifying more than one value in a STYLEATTRS statement option list works only when the attribute priority is 'None'. The STYLEATTRS statement overrides the default style elements, but it does not force the style elements **GraphData2**, **GraphData3**, and so on to be used.

The STYLEATTRS statement options are as follows:

DATACOLORS=(*color-list*)
specifies the fill colors. You can specify common color names or values of the form CX*rrggbb*, where the last six characters specify RGB (red, green, blue) values on the hexadecimal scale of 00 to FF (0 to 255, base 10).

DATACONTRASTCOLORS=(*color-list*)
specifies the contrast colors, which are used for lines and markers. You can specify common color names or values of the form CX*rrggbb*, where the last six characters specify RGB (red, green, blue) values on the hexadecimal scale of 00 to FF (0 to 255, base 10).

DATALINEPATTERNS=(*line-pattern-list*)
specifies the list of line patterns. Some of the available line patterns are **Solid**, **MediumDash**, **MediumDashShortDash**, **LongDash**, **DashDashDot**, **LongDashShortDash**, **DashDotDot**, **Dash**, **ShortDashDot**, **MediumDashDotDot**, and **ShortDash**.

DATASYMBOLS=(*marker-symbol-list*)

specifies the list of marker symbols. The available markers are **ArrowDown**, **Asterisk**, **Circle**, **CircleFilled**, **Diamond**, **DiamondFilled**, **GreaterThan**, **Hash**, **HomeDown**, **HomeDownFilled**, **Ibeam**, **LessThan**, **Plus**, **Square**, **SquareFilled**, **Star**, **StarFilled**, **Tack**, **Tilde**, **Triangle**, **TriangleDown**, **TriangleDownFilled**, **TriangleFilled**, **TriangleLeft**, **TriangleLeftFilled**, **TriangleRight**, **TriangleRightFilled**, **Union**, **X**, **Y**, and **Z**.

In the SG procedures, you can override the style attributes and modify how groups are distinguished by using attribute maps:

```
data myattrmap;
   retain ID 'Attr1' MarkerSymbol 'CircleFilled';
   input Value $ LineColor $ 3-11 MarkerColor $ 13-20;
   datalines;
F pink      cxFFCCEE
M lightblue blue
;

proc sgplot data=sashelp.class dattrmap=myattrmap;
   title 'Separate Fit by Sex';
   reg y=weight x=height / group=sex degree=3 attrid=Attr1;
run;
```

The results are displayed in Figure 24.17.

Figure 24.17 Color Changes with an Attribute Map

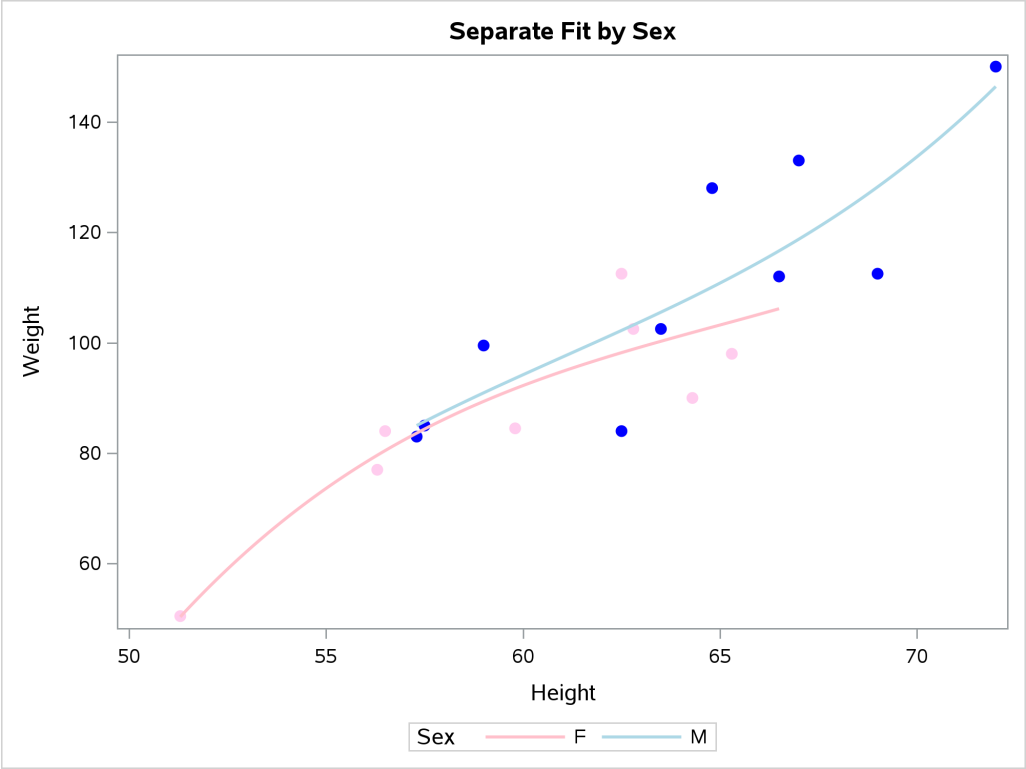

You can set the following variables and values in the PROC SGPLOT DATTRMAP=*SAS-data-set* option:

FillColor	color or style attribute
FillStyle	style element
ID	text string that contains the attribute map ID
LineColor	color or style attribute
LinePattern	line pattern or style attribute
LineStyle	style element
MarkerColor	color or style attribute
MarkerStyle	style element
MarkerSymbol	symbol name or style attribute
Value	text string that contains the group

ODS Style Elements and Attributes

An ODS style template is composed of a set of *style elements*. A style element is a collection of *style attributes* that applies to a particular feature or aspect of the output. A value is specified for each attribute in a style template. For example, **GraphFit** is the style element that is used for fit lines, and its attributes include **LineThickness**, **LineStyle**, **MarkerSize**, **MarkerSymbol**, **ContrastColor**, and **Color**.

In general, style templates control the overall appearance of ODS tables and graphs. For tables, style templates specify features such as background color, table borders, and color scheme, and they specify the fonts, sizes, and color for the text and values in a table and its headings. For graphs, style templates specify the following features:

- background color
- graph dimensions (height and width)
- borders
- line styles for axes and grid lines
- fonts, sizes, and colors for titles, footnotes, axis labels, axis values, and data labels (see the section "Modifying Graph Fonts in Styles" on page 768 for an illustration)
- marker symbols, colors, and sizes for data points and outliers
- line styles for needles
- line and curve styles for fitted models and predicted values (see the section "Modifying Other Graph Elements in Styles" on page 771 for an illustration)
- line and curve styles for confidence and prediction limits
- fill colors for histogram bars, confidence bands, and confidence ellipses
- colors for box plot features
- colors for surfaces
- color ramps for contour plots

The SAS System supplies a graph template for each graph that is created by statistical procedures. A graph template is a program that specifies the layout and details of a graph. For more information about templates, see the section "Graph Templates" on page 844 in Chapter 25, "ODS Graphics Template Modification." Some template options are specified by using a style reference of the form **style-element** or occasionally **style-element:attribute**. For example, the symbol, color, and size of markers for basic scatter plots are specified in a template SCATTERPLOT statement as follows:

```
scatterplot x=x y=y / markerattrs=GraphDataDefault;
```

The preceding statement specifies that the appearance for markers is controlled by the **GraphDataDefault** element. Consistent use of this element guarantees a common appearance of markers across all scatter plots, based on the style template that you are using.

In general, ODS Graphics features are determined by style element attributes unless they are overridden by a statement or option in the graph template. For example, suppose that a classification variable is specified in the GROUP= option in a SCATTERPLOT template statement as follows:

```
scatterplot x=X y=Y / group=GroupVar;
```

Then the colors for markers that correspond to the classification levels are assigned by using the style element attributes **GraphData1:ContrastColor** through **GraphData12:ContrastColor**.

Style templates are created and modified by using PROC TEMPLATE. For more information, see the *SAS Output Delivery System: User's Guide*. You need to understand the relationships between style elements and graph features if you want to create your own style template or modify a style template. These relationships are explained in the following sections.

Style Templates and Colors

The default style templates that the SAS System provides are stored in the *Styles* directory of Sashelp.Tmplmst. You can display, edit, and save style templates by using the same that are methods available for modifying graph and table templates, as explained in the section "The Default Template Stores and the Template Search Path" on page 715 and the series of sections beginning with the section "Displaying Templates" on page 850 in Chapter 25, "ODS Graphics Template Modification." In particular, you can display a style template by using one of the following methods:

- Open the Templates window by entering the command **odst** on the command line of the SAS windowing environment, expand the Sashelp.Tmplmst node under **Templates**, and then select **Styles** to display the contents of this folder.
- Use the SOURCE statement in PROC TEMPLATE.

For example, the following statements display the DEFAULT style template in the SAS log:

```
proc template;
   source Styles.Default;
run;
```

Some of the results are as follows:

```
define style Styles.Default;
   . . .
   class GraphColors
      "Abstract colors used in graph styles" /
      . . .
      'gconramp3cend' = cxFF0000
      'gconramp3cneutral' = cxFF00FF
```

```
        'gconramp3cstart' = cx0000FF
        . . .
        'gdata12' = cxDDD17E
        'gdata11' = cxB7AEF1
        'gdata10' = cx87C873
        'gdata9' = cxCF974B
        'gdata8' = cxCD7BA1
        'gdata6' = cxBABC5C
        'gdata7' = cx94BDE1
        'gdata4' = cxA9865B
        'gdata5' = cxB689CD
        'gdata3' = cx66A5A0
        'gdata2' = cxDE7E6F
        'gdata1' = cx7C95CA;
    . . .
```

The first part of this list shows that the shading for certain filled plots, such as some contour plots, goes from blue (**'gconramp3cstart' = cx0000FF**) to magenta (**'gconramp3cneutral' = cxFF00FF**) to red (**'gconramp3cend' = cxFF0000**). All colors are specified in values of the form CX*rrggbb*, where the last six characters specify RGB (red, green, blue) values on the hexadecimal scale of 00 to FF (0 to 255, base 10). The second part of the list (**'gdata1' = cx7C95CA**) shows that the dominant component of the **GraphData1** color is blue because the blue component of the color (CA, which corresponds to 202, base 10) is greater than both the green component (95, which corresponds to 149, base 10) and the red component (7C, which corresponds to 124, base 10).

You can change any part of the style and then submit the style back to the SAS System, after first submitting a PROC TEMPLATE statement. For more information about modifying, using, and restoring templates, see the sections "Saving Customized Templates" on page 853, "Using Customized Templates" on page 854, and "Reverting to the Default Templates" on page 854 in Chapter 25, "ODS Graphics Template Modification." The principles that are discussed in those sections apply to all templates—table, style, and graph.

Some Common ODS Style Elements

This section explains some common ODS style elements and produces most of the graphs that are displayed in the section "ODS Style Comparisons" on page 733.

The DEFAULT style is the parent for the styles that are used for statistical graphics work. You can see all the elements of the DEFAULT style by running the following step:

```
proc template;
   source styles.default;
run;
```

The source listing of the definition of the DEFAULT style is hundreds of lines long. If you run PROC TEMPLATE along with the SOURCE statement for most other styles, you see **parent = styles.default** (or in the case of the HTMLBLUE style, you see **parent = styles.statistical**, which inherits attributes from the DEFAULT style), and you do not see all the elements in the style unless you also add a slash and the EXPAND option to the SOURCE statement.

Only a few of the style elements are referenced in the templates that the SAS System provides for statistical procedures. The most commonly used style elements, along with the defaults for the noncolor attributes of

the DEFAULT style, are shown next (**`Color`** applies to filled areas, and **`ContrastColor`** applies to markers and lines):

`Graph`	graph size, outer border appearance, and background color **`Padding = 0`** **`BackgroundColor`**
`GraphConfidence`	primary fit confidence interval **`LineThickness = 1px`** **`LineStyle = 1`** **`MarkerSize = 7px`** **`MarkerSymbol = "triangle"`** **`ContrastColor`** **`Color`**
`GraphData1`	attributes related to first grouped data items **`MarkerSymbol = "circle"`** **`LineStyle = 1`** **`ContrastColor`** **`Color`**
`GraphData2`	attributes related to second grouped data items **`MarkerSymbol = "plus"`** **`LineStyle = 4`** **`ContrastColor`** **`Color`**
`GraphData3`	attributes related to third grouped data items **`MarkerSymbol = "X"`** **`LineStyle = 8`** **`ContrastColor`** **`Color`**
`GraphData4`	attributes related to fourth grouped data items **`MarkerSymbol = "triangle"`** **`LineStyle = 5`** **`ContrastColor`** **`Color`**
`GraphData`*n*	attributes related to *n*th grouped data items **`MarkerSymbol`** **`LineStyle`** **`ContrastColor`** **`Color`**
`GraphDataDefault`	attributes related to ungrouped data items **`EndColor`** **`NeutralColor`** **`StartColor`** **`MarkerSize = 7px`** **`MarkerSymbol = "circle"`** **`LineThickness = 1px`** **`LineStyle = 1`** **`ContrastColor`** **`Color`**

GraphFit	primary fit line or a normal density curve **LineThickness = 2px** **LineStyle = 1** **MarkerSize = 7px** **MarkerSymbol = "circle"** **ContrastColor** **Color**
GraphFit2	secondary fit line or a kernel density curve **LineThickness = 2px** **LineStyle = 4** **MarkerSize = 7px** **MarkerSymbol = "X"** **ContrastColor** **Color**
GraphGridLines	horizontal and vertical grid lines drawn at major tick marks **Displayopts = "auto"** **LineThickness = 1px** **LineStyle = 1** **ContrastColor** **Color**
GraphOutlier	outlier data for the graph **LineThickness = 2px** **LineStyle = 42** **MarkerSize = 7px** **MarkerSymbol = "circle"** **ContrastColor** **Color**
GraphPredictionLimits	fills for prediction limits **LineThickness = 1px** **LineStyle = 2** **MarkerSize = 7px** **MarkerSymbol = "chain"** **ContrastColor** **Color**
GraphReference	horizontal and vertical reference lines and drop lines **LineThickness = 1px** **LineStyle = 1** **ContrastColor**
GraphDataText	text font and color for point and line labels **Font = GraphFonts('GraphDataFont')** (where **'GraphDataFont' = ("<sans-serif>, <MTsans-serif>",7pt)**) **Color**
GraphValueText	text font and color for axis tick values and legend values **Font = GraphFonts('GraphValueFont')** (where **'GraphValueFont' = ("<sans-serif>, <MTsans-serif>",9pt)**) **Color**

GraphLabelText	text font and color for axis labels and legend title **Font = GraphFonts('GraphLabelFont')** (where **'GraphLabelFont' = ("<sans-serif>, <MTsans-serif>",10pt,bold)**) **Color**
GraphFootnoteText	text font and color for footnotes **Font = GraphFonts('GraphFootnoteFont')** (where **'GraphFootnoteFont' = ("<sans-serif>, <MTsans-serif>",10pt)**) **Color**
GraphTitleText	text font and color for titles **Font = GraphFonts('GraphTitleFont')** (where **'GraphTitleFont' = ("<sans-serif>, <MTsans-serif>",11pt,bold)**) **Color**
GraphWalls	vertical walls bounded by axes **LineThickness = 1px** **LineStyle = 1** **FrameBorder = on** **ContrastColor** **BackgroundColor** **Color**

You refer to these elements in graph templates **style-element** or **style-element:attribute** (for example, **GraphDataDefault:ContrastColor**). The default values are not shown for the color attributes because they are usually defined indirectly. For example, **Graph:BackgroundColor** (the color that fills the box outside the graph) is defined elsewhere in the style as **colors('docbg')**. The style also defines **'docbg' = color_list('bgA')** and **'bgA' = cxE0E0E0**. This shows that the background is a shade of gray that is much closer to white (CXFFFFFF) than to black (CX000000). You can see the background color in Figure 24.38. This shade of gray might seem darker (closer to CX000000) than you might expect based on only the RGB values. Your perception of a color change is not a linear function of the change in RGB values.

You can use the following program to see the color and other attributes for a number of style elements:

```
proc format;
   value vf 5 = 'GraphValueText';
run;

data x1;
   array y[20] y0 - y19;
   do x = 1 to 20; y[x] = x - 0.5; end;
   do x = 0 to 10 by 5; output; end;
   label y18 = 'GraphLabelText' x = 'GraphLabelText';
   format x y18 vf.;
run;

%macro d;
   %do i = 1 %to 12;
      reg y=y%eval(19-&i) x=x / lineattrs=GraphData&i markerattrs=GraphData&i
                                curvelabel="  GraphData&i" curvelabelpos=max;
```

```
   %end;
%mend;

%macro l(i, l);
   reg y=y&i x=x / lineattrs=&l markerattrs=&l curvelabel="  &l"
                   curvelabelpos=max;
%mend;

ods html style=default; * You can instead specify other destinations
                          such as LISTING, PDF, or RTF;
proc sgplot noautolegend data=x1;
   title 'GraphTitleText';
   %d
   %l(19, GraphDataDefault)
   %l( 6, GraphFit)
   %l( 5, GraphFit2)
   %l( 4, GraphPredictionLimits)
   %l( 3, GraphConfidence)
   %l( 2, GraphGridLines)
   %l( 1, GraphOutlier)
   %l( 0, GraphReference)
   xaxis values=(0 5 10);
run;
ods html close;
```

The results in Figure 24.38 display the attributes for a number of the elements of the DEFAULT style.

When there is a group or classification variable, the colors, markers, and lines that distinguish the groups are derived from the **GraphData***n* elements that are defined in the style. In the DEFAULT style, these are the elements **GraphData1** through **GraphData12**. There can be any number of groups, even though only 12 **GraphData***n* style elements are defined in the DEFAULT style. The following steps create a data set that contains 40 groups, display one line per group, and produce Figure 24.49:

```
data x2;
   do y = 40 to 1 by -1;
      group = 'Group' || put(41 - y, 2. -L);
      do x = 0 to 10 by 5;
         if x = 10 then do; z = 11; l = group; end;
         else           do; z = .;  l = ' ';   end;
         output;
      end;
   end;
run;

proc sgplot data=x2;
   title 'Colors, Markers, Lines Patterns for Groups';
   series  y=y x=x / group=group markers;
   scatter y=y x=z / group=group markerchar=l;
run;
```

The colors, markers, and line patterns in Figure 24.49 repeat in cycles. The **GraphData1** – **GraphData8** lines in Figure 24.38 exactly match the **Group1** – **Group8** lines in Figure 24.49. After that, there are differences due to the cyclic construction of the grouped style. This is explained next.

The DEFAULT style defines a marker symbol only in **GraphData1** through **GraphData7**. The seven

markers are circle, plus sign, X, triangle, square, asterisk, and diamond. With the explicit style reference in Figure 24.38, the actual symbol, when no symbol is specified, is the circle. This is what you see for **GraphData8** through **GraphData12**. With the group variable in Figure 24.49, the symbols repeat in cycles. Hence, **Group1**, **Group8**, **Group15**, and so on, are all circles. Similarly, **Group2**, **Group9**, **Group16**, and so on, are all plus signs. The DEFAULT style defines 11 different line styles for **GraphData1** through **GraphData11**. You specify line styles by specifying an integer. The default lines styles are 1, 4, 8, 5, 14, 26, 15, 20, 41, 42, and 2. Hence, **Group1**, **Group12**, **Group23**, and so on, all have the same line style, which is a solid line (line style 1). Similarly, **Group2**, **Group13**, **Group24**, and so on, all have line style 4. There are 12 different colors, so **Group1**, **Group13**, **Group25**, and so on, all have the same colors. Overall, there are $12 \times 11 \times 7 = 924$ color, line, and marker combinations that appear before any combination repeats. You can use the %MODSTYLE SAS autocall macro (see the section "ODS Style Template Modification Macro" on page 761) to conveniently change these style attributes.

The HTMLBLUE style is an all-color style for the first 12 groups of observations. Most analyses have fewer than 12 groups. Markers and lines change for groups 13–24 and then again for groups 25–36. Figure 24.50 shows how colors, markers, and line styles change in the HTMLBLUE style. Figure 24.49 and Figure 24.50 through Figure 24.59 show how these elements change in other styles.

ODS Style Comparisons

In this section, some of the most commonly used styles are compared in a series of figures, most of which were generated in the preceding section. Figure 24.18 through Figure 24.27 show tables and graphs in the HTML destination for each of eight styles, for the following analysis:

```
proc reg data=sashelp.class;
   model Weight = Height;
quit;
```

The PEARL, PEARLJ, RTF, SAPPHIRE, and six JOURNAL styles are compared by running the following steps for each of the ten styles and capturing output in the PDF destination:

```
options nonumber nodate;
ods proctitle off;

ods pdf body="fPearlJ.pdf" style=PearlJ startpage=never;
title "PearlJ";
proc means data=sashelp.class maxdec=2;
run;

proc sgplot data=sashelp.class;
   vbar age / group=sex;
run;
ods pdf close;
```

The results of these steps are displayed in Figure 24.28 through Figure 24.37.

Figure 24.38 through Figure 24.48 show some of the most common style elements. Figure 24.49 through Figure 24.56 show how groups of observations are displayed in the graph.

The style comparisons are as follows:

- The ANALYSIS style is displayed in Figure 24.22, Figure 24.42, and Figure 24.53.
- The DEFAULT style is displayed in Figure 24.18, Figure 24.38, and Figure 24.49.
- The HTMLBLUE style is displayed in Figure 24.19, Figure 24.39, and Figure 24.50.
- The HTMLBLUECML style is displayed in Figure 24.20, Figure 24.40, and Figure 24.51.
- The JOURNAL style is displayed in Figure 24.23, Figure 24.28, Figure 24.43, and Figure 24.54.
- The JOURNAL1A style is displayed in Figure 24.29.
- The JOURNAL2 style is displayed in Figure 24.30.
- The JOURNAL2A style is displayed in Figure 24.31.
- The JOURNAL3 style is displayed in Figure 24.32.
- The JOURNAL3A style is displayed in Figure 24.33.
- The LISTING style is displayed in Figure 24.24, Figure 24.44, and Figure 24.55.
- The PEARL style is displayed in Figure 24.26, Figure 24.34, Figure 24.46, and Figure 24.57.
- The PEARLJ style is displayed in Figure 24.35, Figure 24.47, and Figure 24.58.
- The RTF style is displayed in Figure 24.25, Figure 24.36, Figure 24.45, and Figure 24.56.
- The SAPPHIRE style is displayed in Figure 24.27, Figure 24.37, Figure 24.48, and Figure 24.59.
- The STATISTICAL style is displayed in Figure 24.21, Figure 24.41, and Figure 24.52.

Figure 24.18 Statistical Output with the DEFAULT Style and HTML Destination

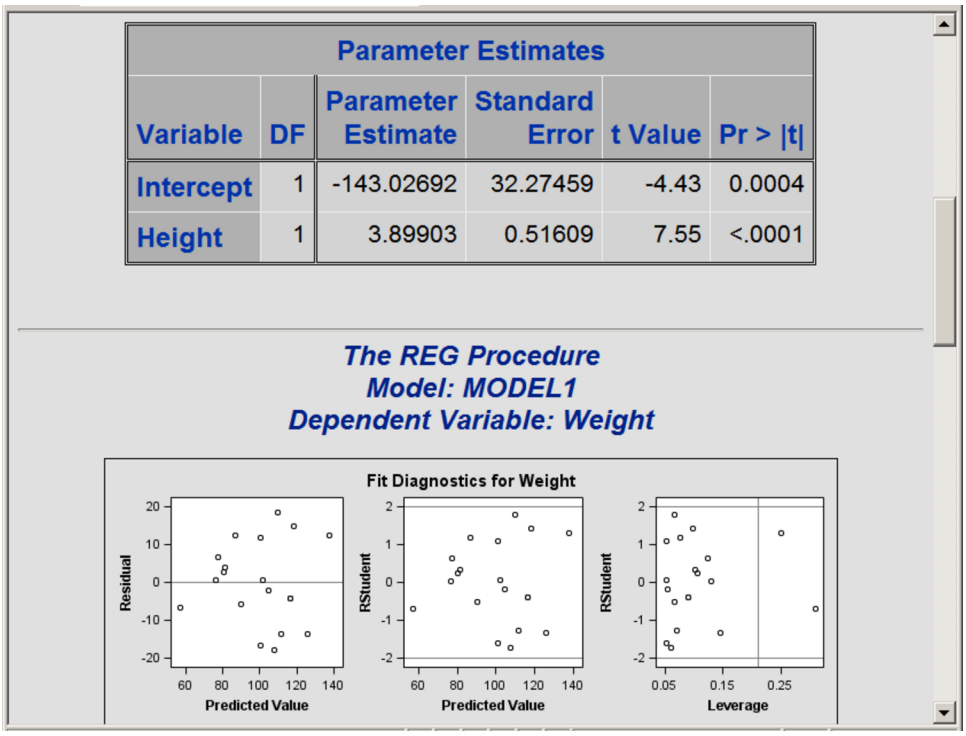

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.19 Statistical Output with the HTMLBLUE Style and HTML Destination

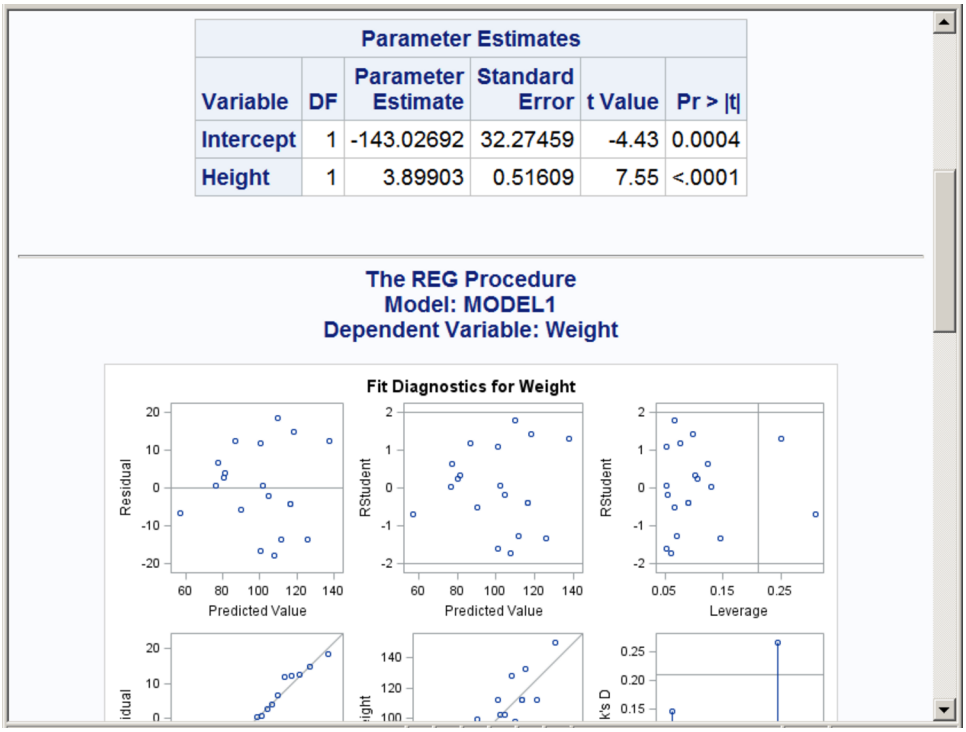

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.20 Statistical Output with the HTMLBLUECML Style and HTML Destination

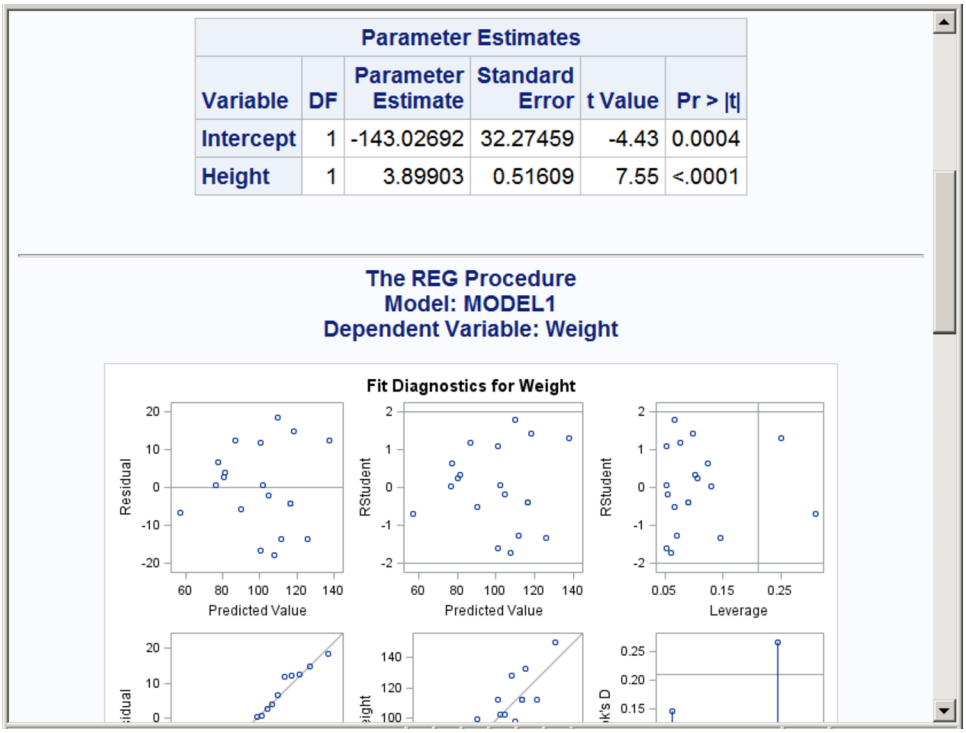

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.21 Statistical Output with the STATISTICAL Style and HTML Destination

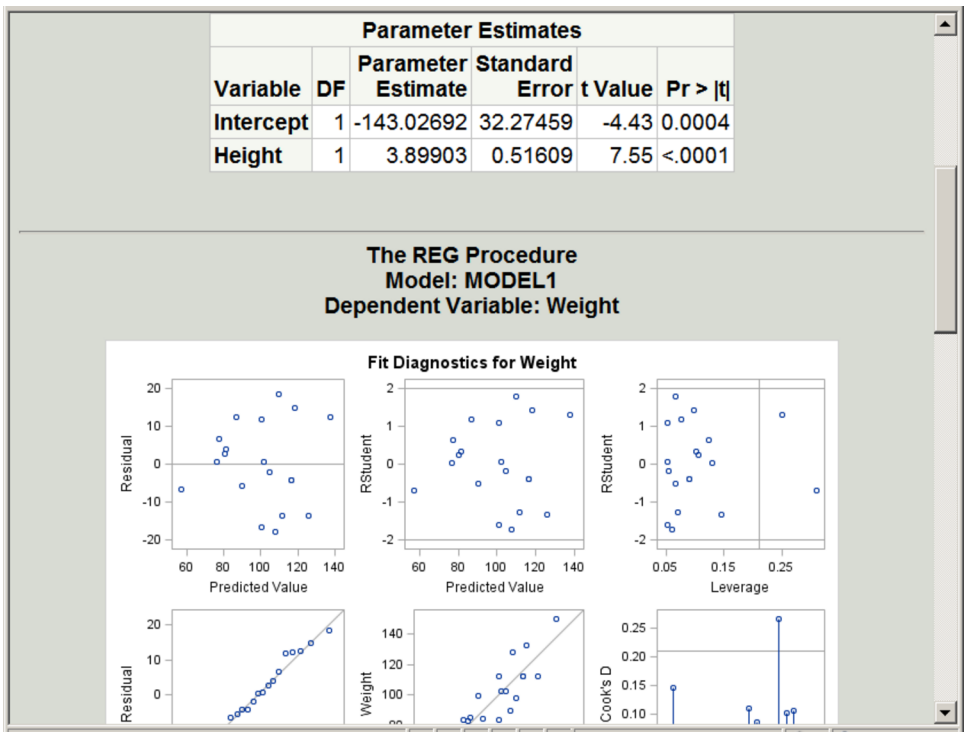

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.22 Statistical Output with the ANALYSIS Style and HTML Destination

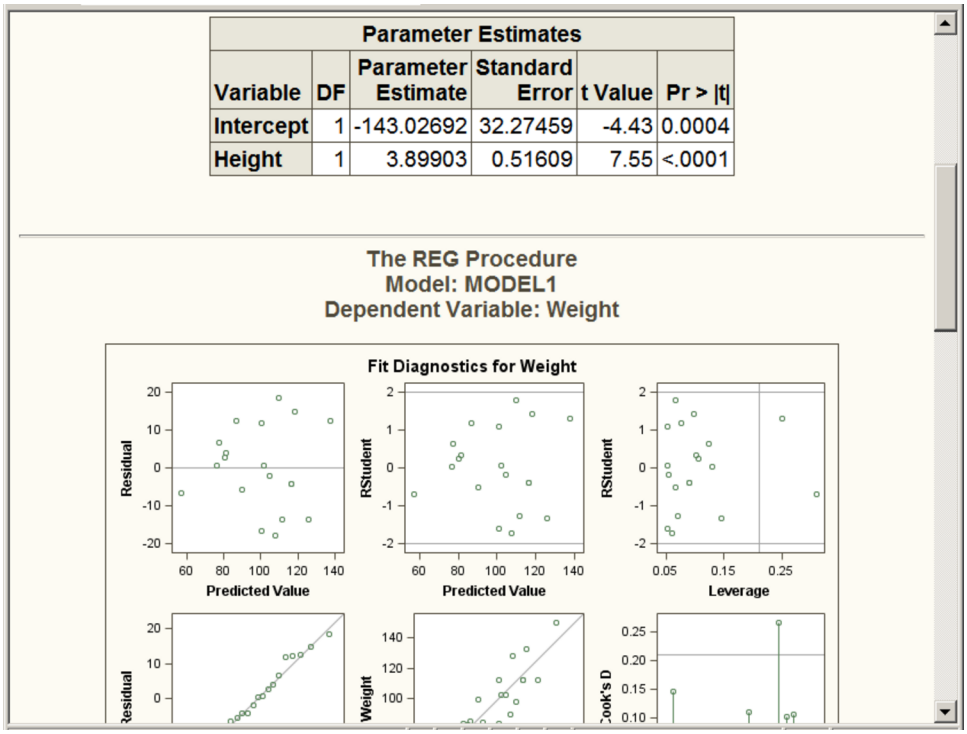

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.23 Statistical Output with the JOURNAL Style and HTML Destination

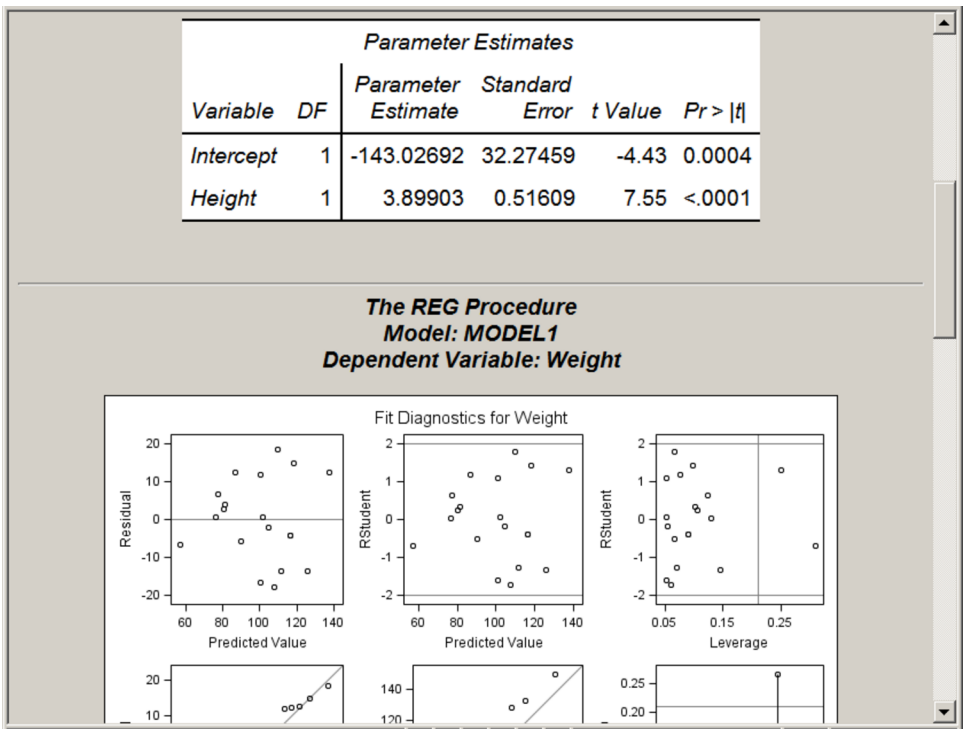

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.24 Statistical Output with the LISTING Style and HTML Destination

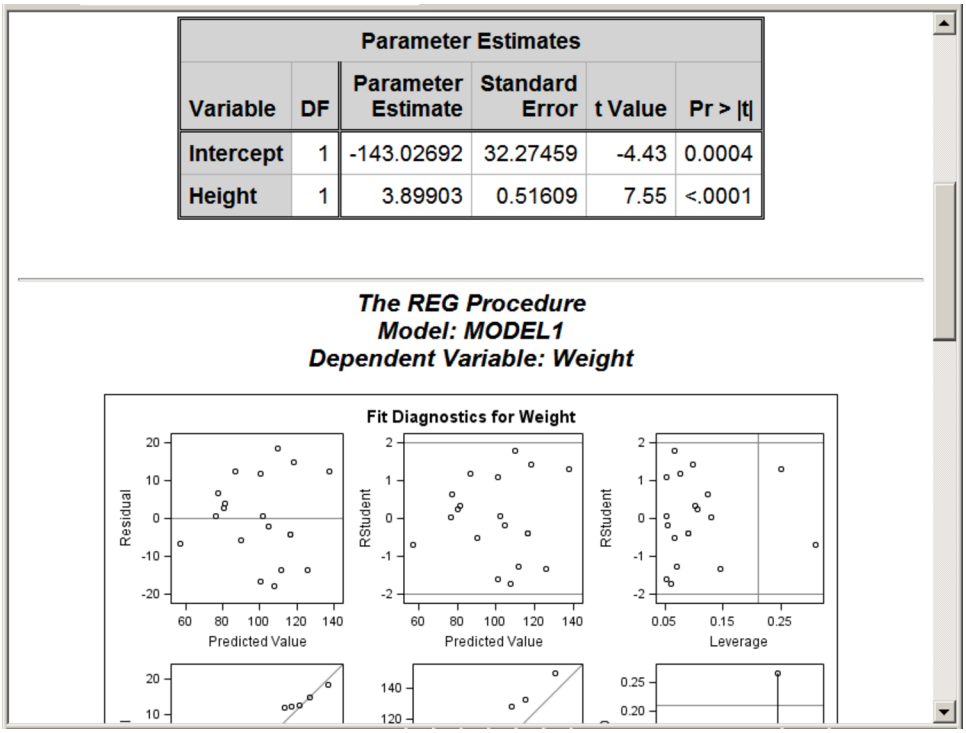

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.25 Statistical Output with the RTF Style and HTML Destination

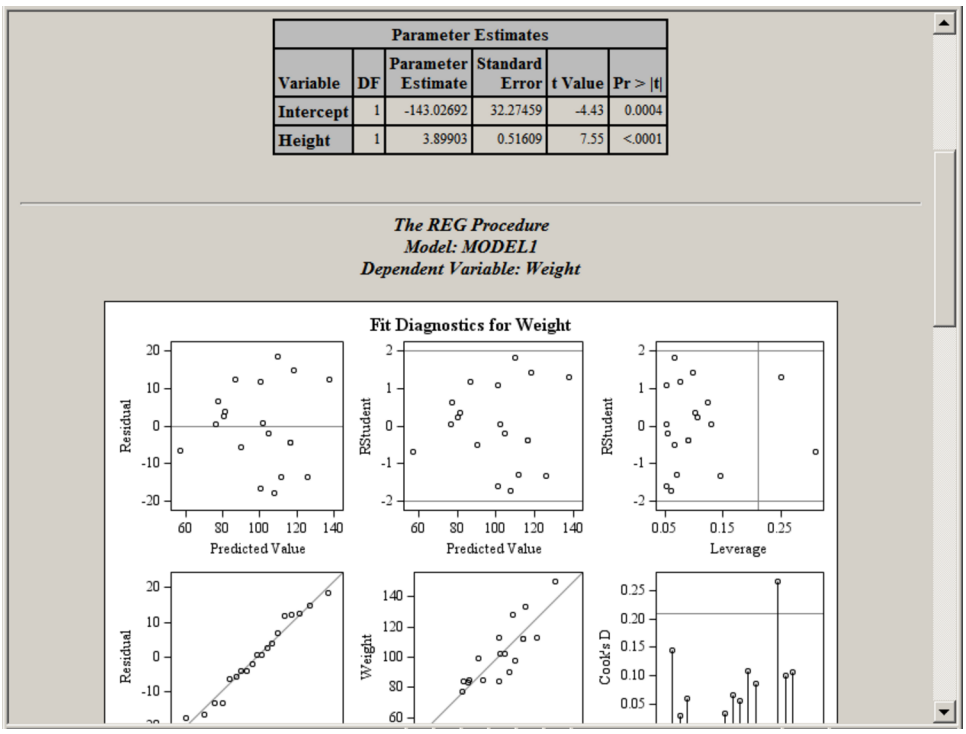

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.26 Statistical Output with the PEARL Style and HTML Destination

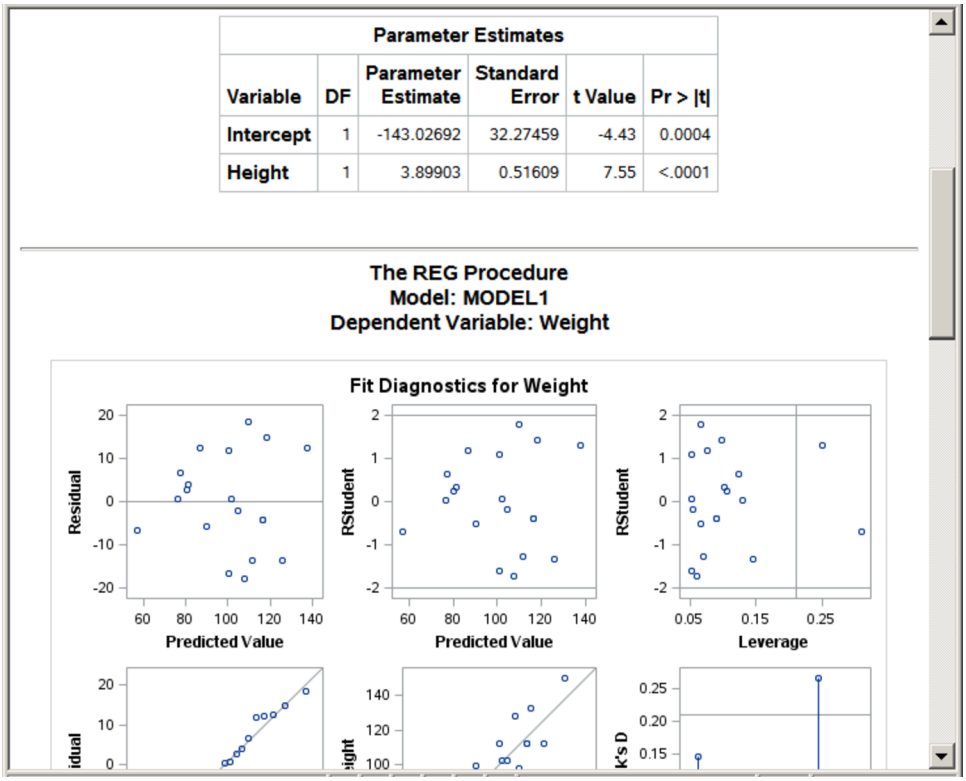

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.27 Statistical Output with the SAPPHIRE Style and HTML Destination

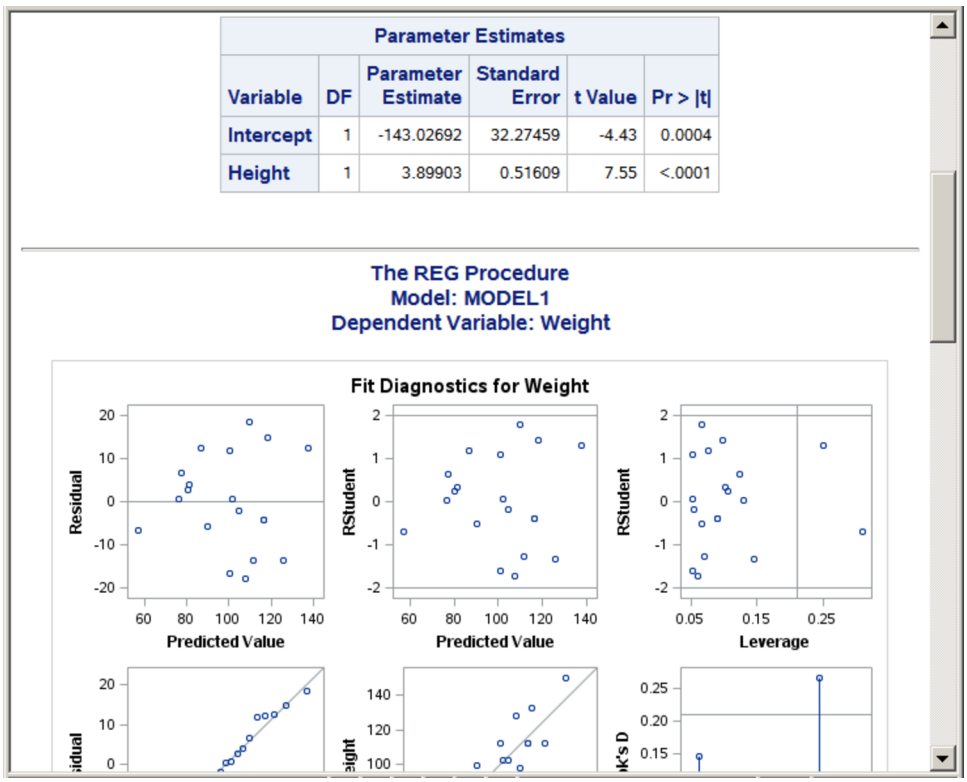

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Figure 24.28 JOURNAL Style and PDF Destination

Journal

Variable	*N*	*Mean*	*Std Dev*	*Minimum*	*Maximum*
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

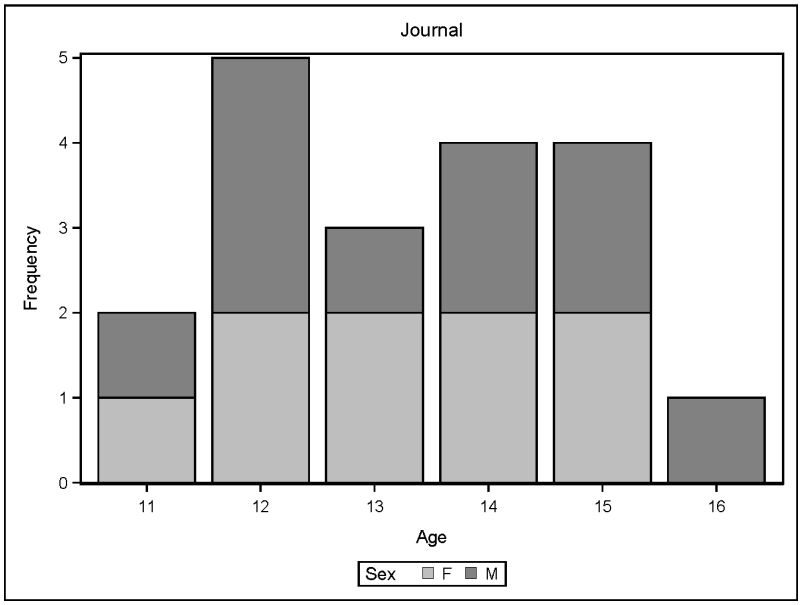

Figure 24.29 JOURNAL1A Style and PDF Destination

Journal1a

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

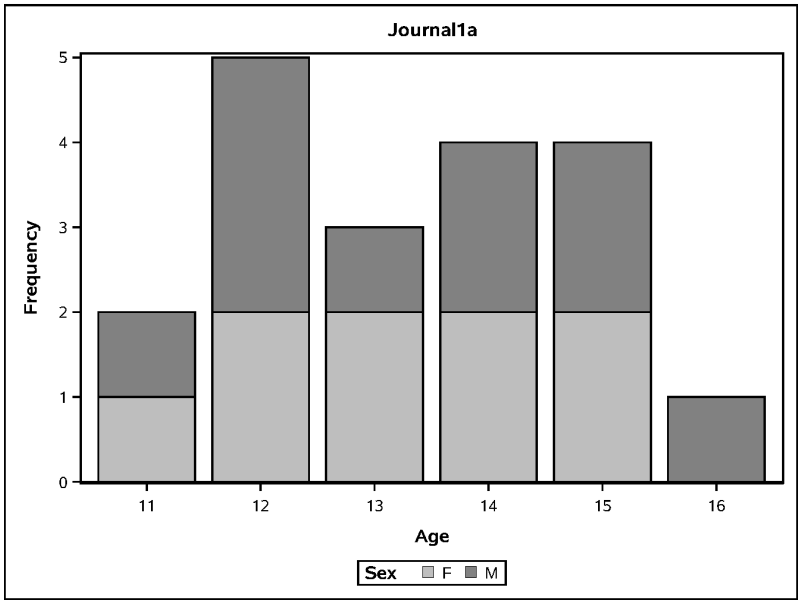

Figure 24.30 JOURNAL2 Style and PDF Destination

Journal2

Variable	*N*	*Mean*	*Std Dev*	*Minimum*	*Maximum*
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

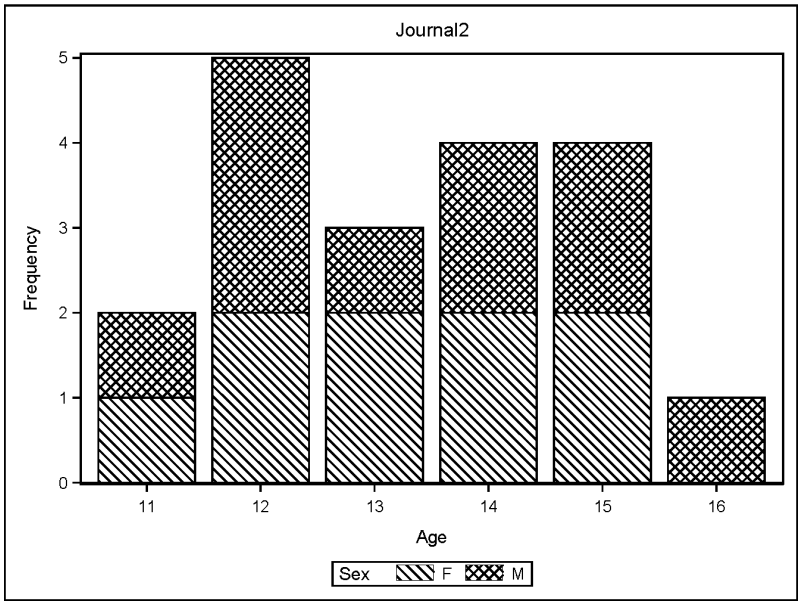

Figure 24.31 JOURNAL2A Style and PDF Destination

Journal2a

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

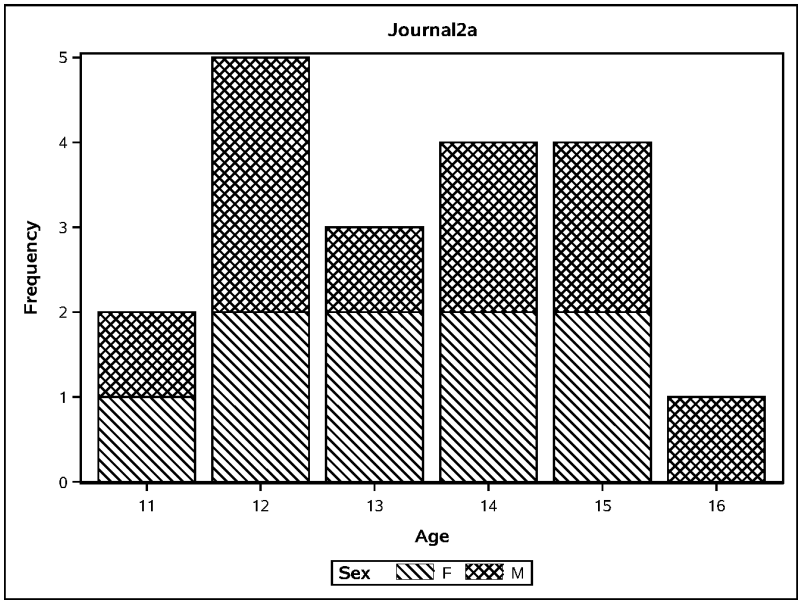

Figure 24.32 JOURNAL3 Style and PDF Destination

Journal3

Variable	*N*	*Mean*	*Std Dev*	*Minimum*	*Maximum*
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

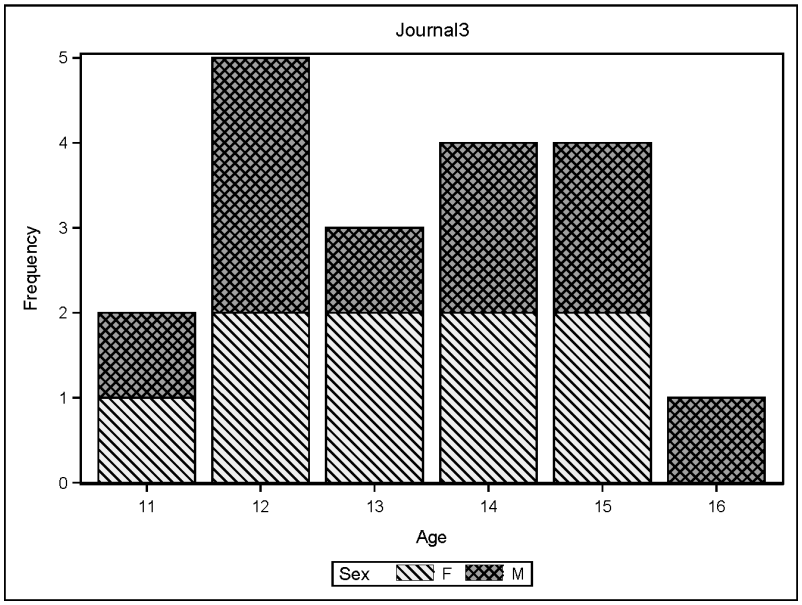

Figure 24.33 JOURNAL3A Style and PDF Destination

Journal3a

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

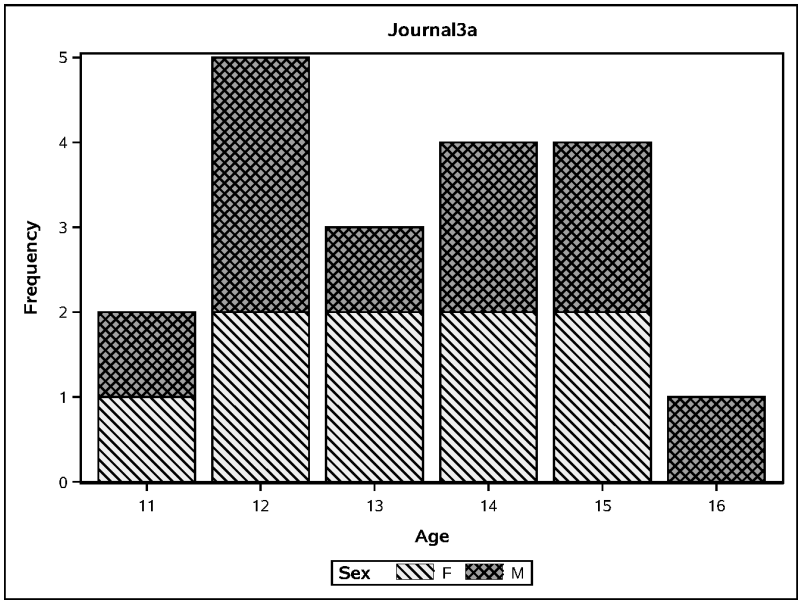

Figure 24.34 PEARL Style and PDF Destination

Pearl

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

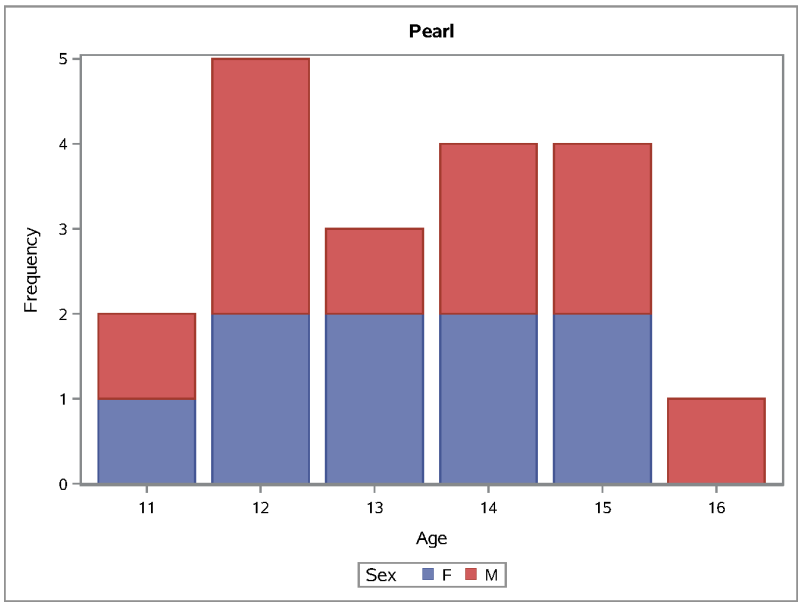

Figure 24.35 PEARLJ Style and PDF Destination

PearlJ

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

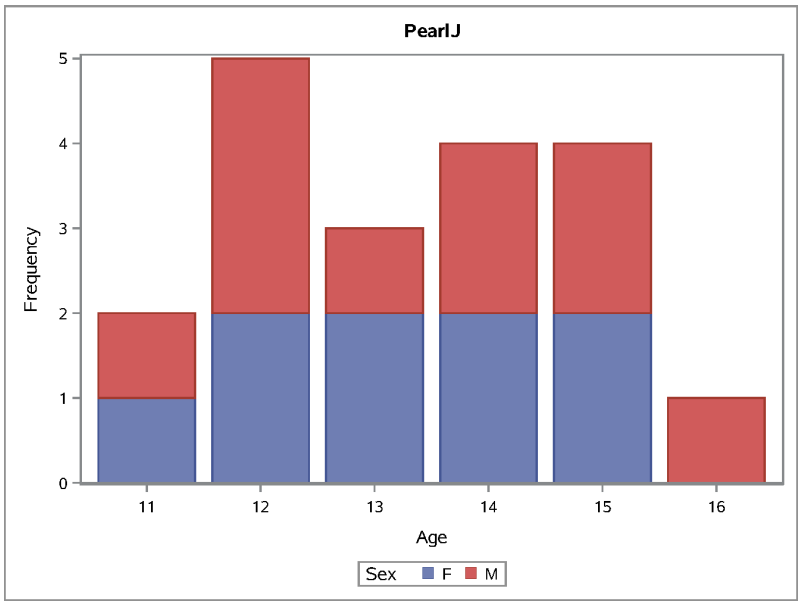

Figure 24.36 RTF Style and PDF Destination

RTF

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

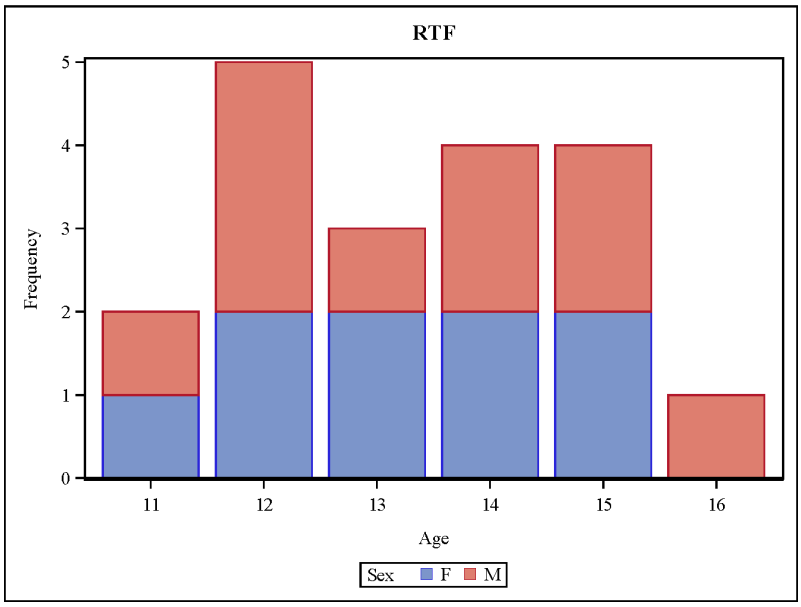

Figure 24.37 SAPPHIRE Style and PDF Destination

Sapphire

Variable	N	Mean	Std Dev	Minimum	Maximum
Age	19	13.32	1.49	11.00	16.00
Height	19	62.34	5.13	51.30	72.00
Weight	19	100.03	22.77	50.50	150.00

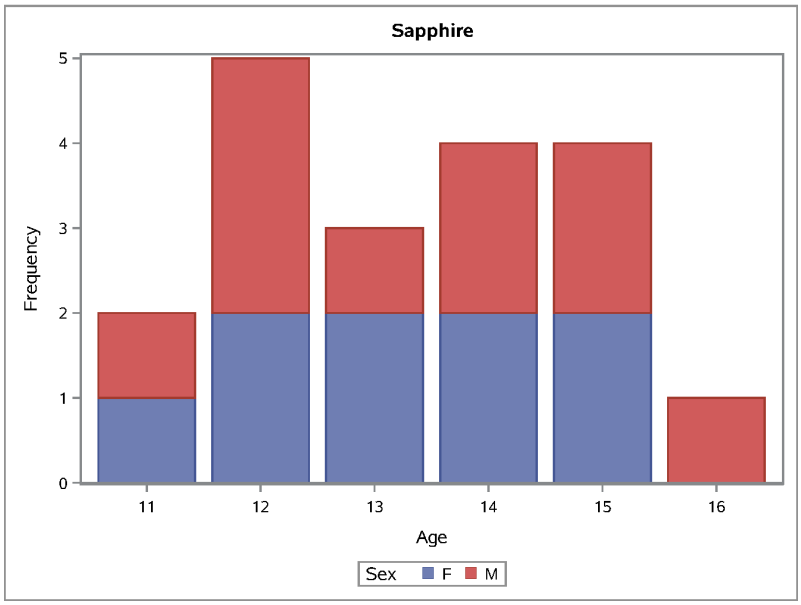

Figure 24.38 Attributes of Style Elements in the DEFAULT Style

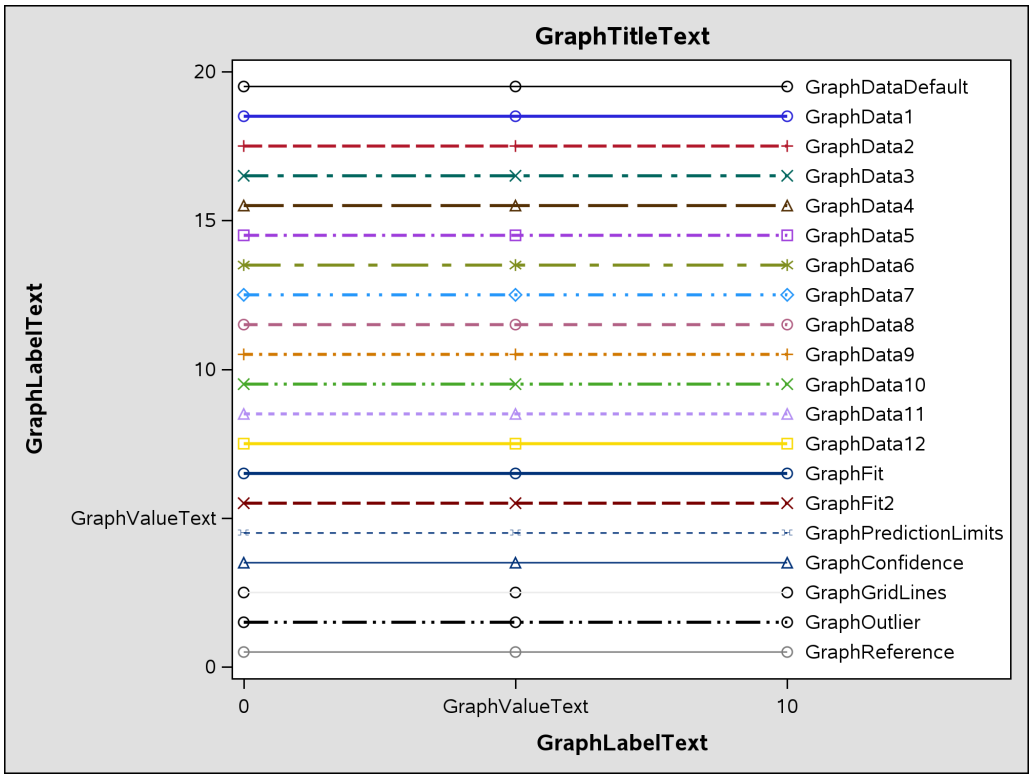

Figure 24.39 Attributes of Style Elements in the HTMLBLUE Style

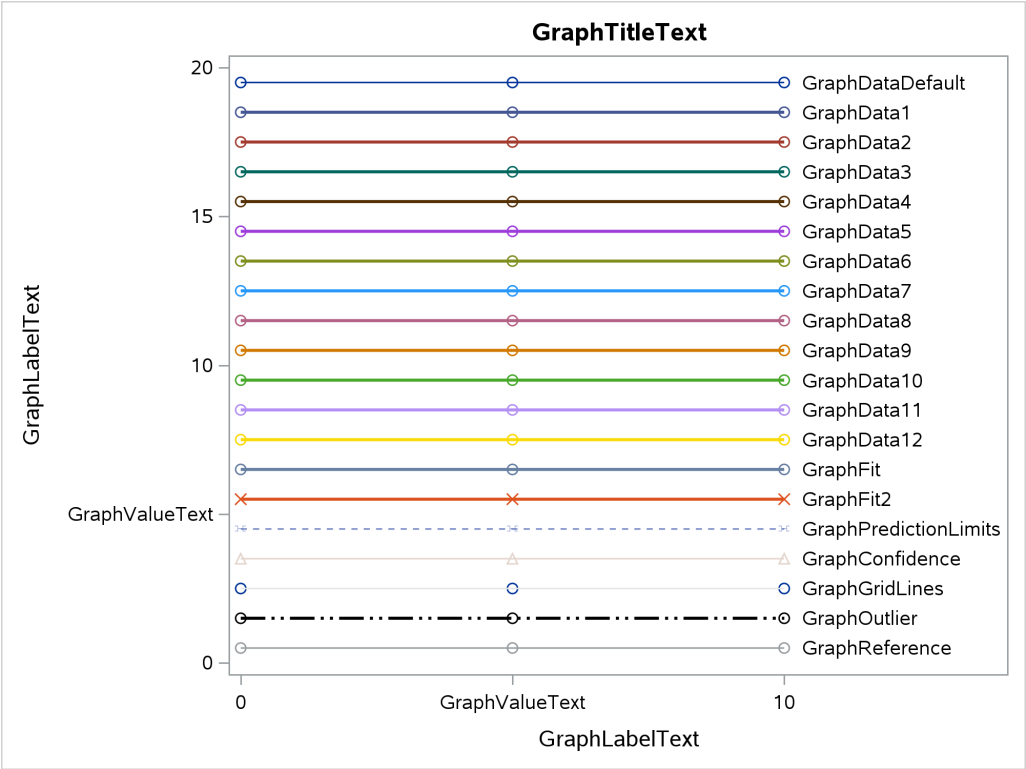

Figure 24.40 Attributes of Style Elements in the HTMLBLUECML Style

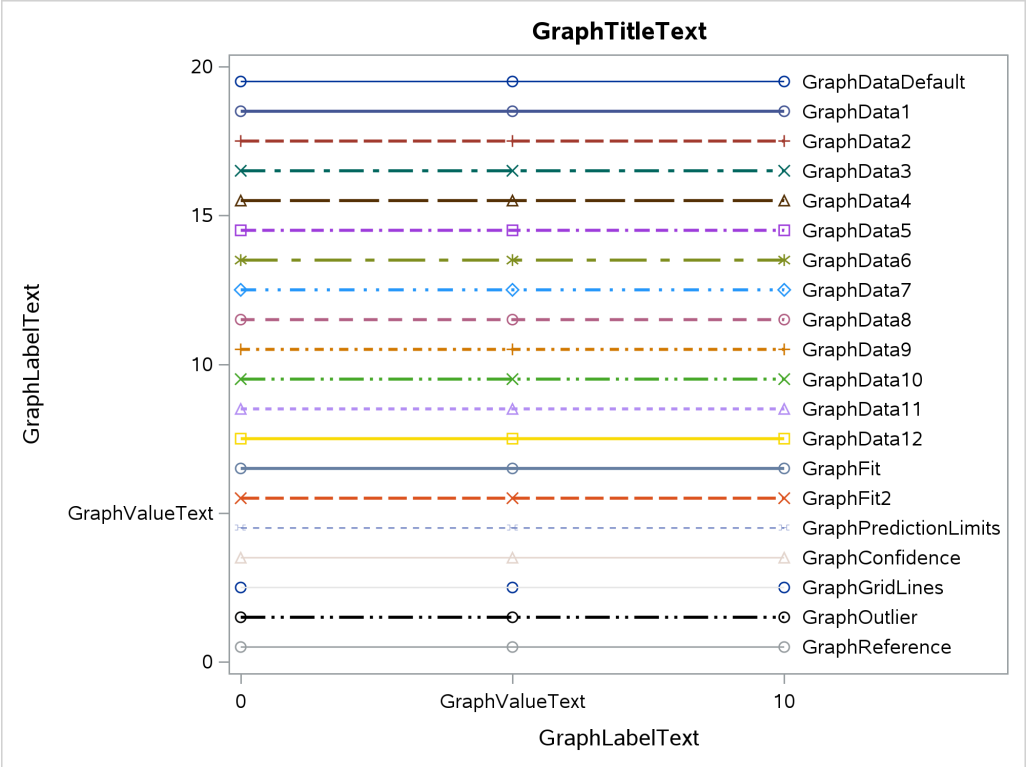

Figure 24.41 Attributes of Style Elements in the STATISTICAL Style

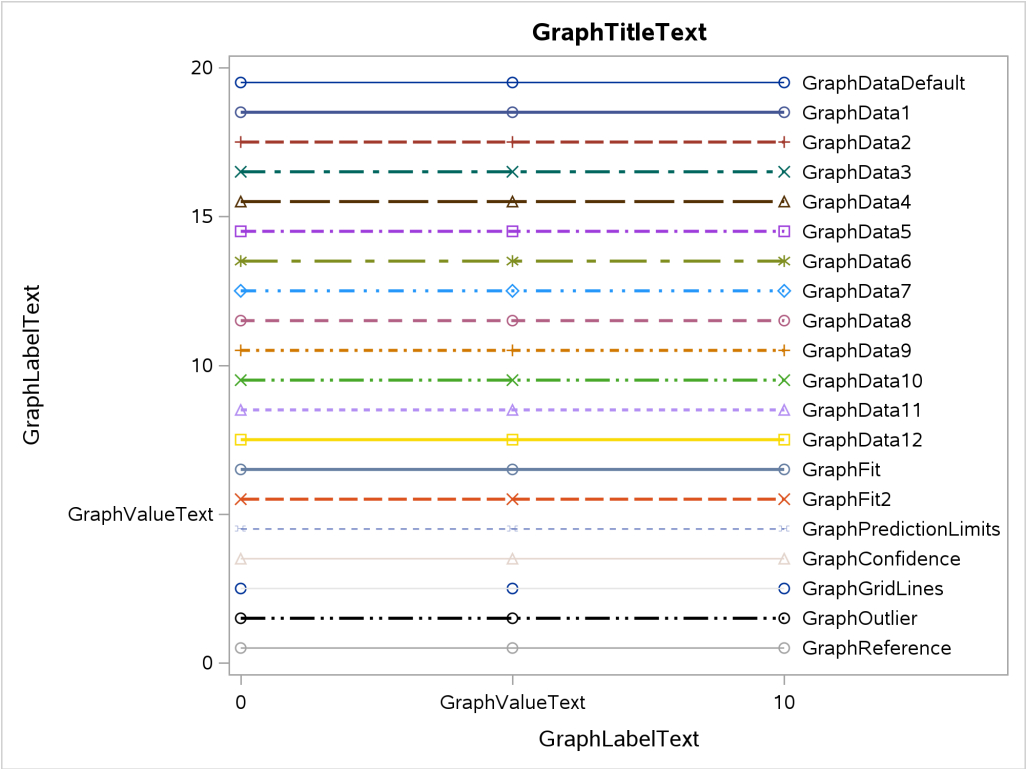

Figure 24.42 Attributes of Style Elements in the ANALYSIS Style

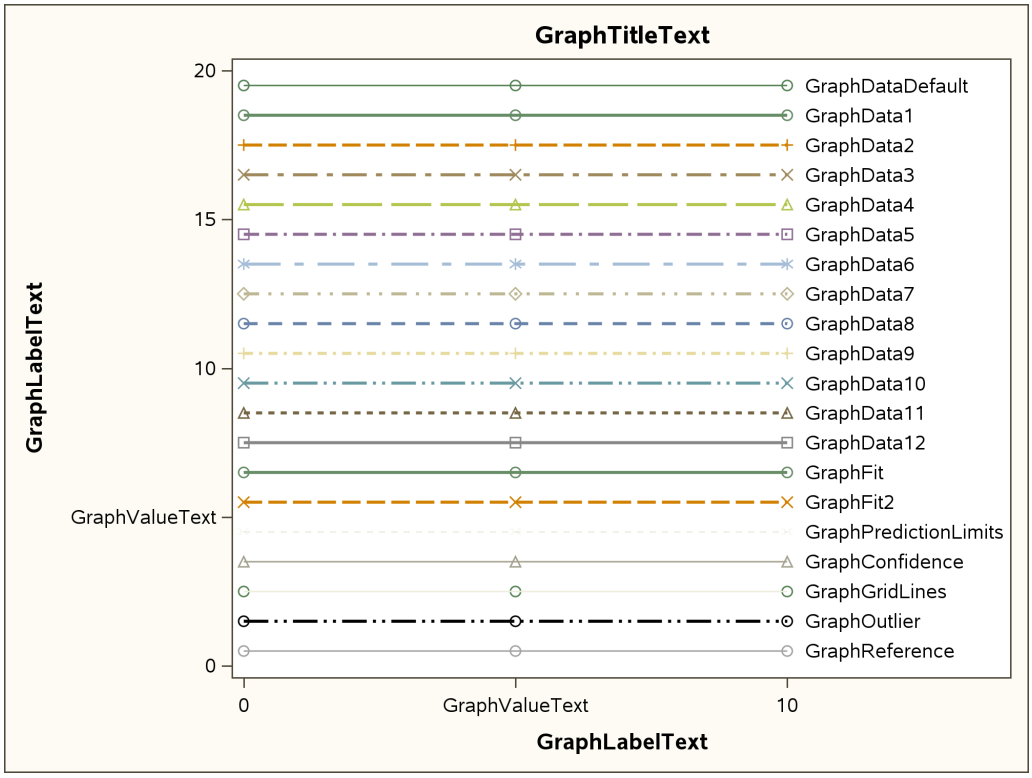

Figure 24.43 Attributes of Style Elements in the JOURNAL Style

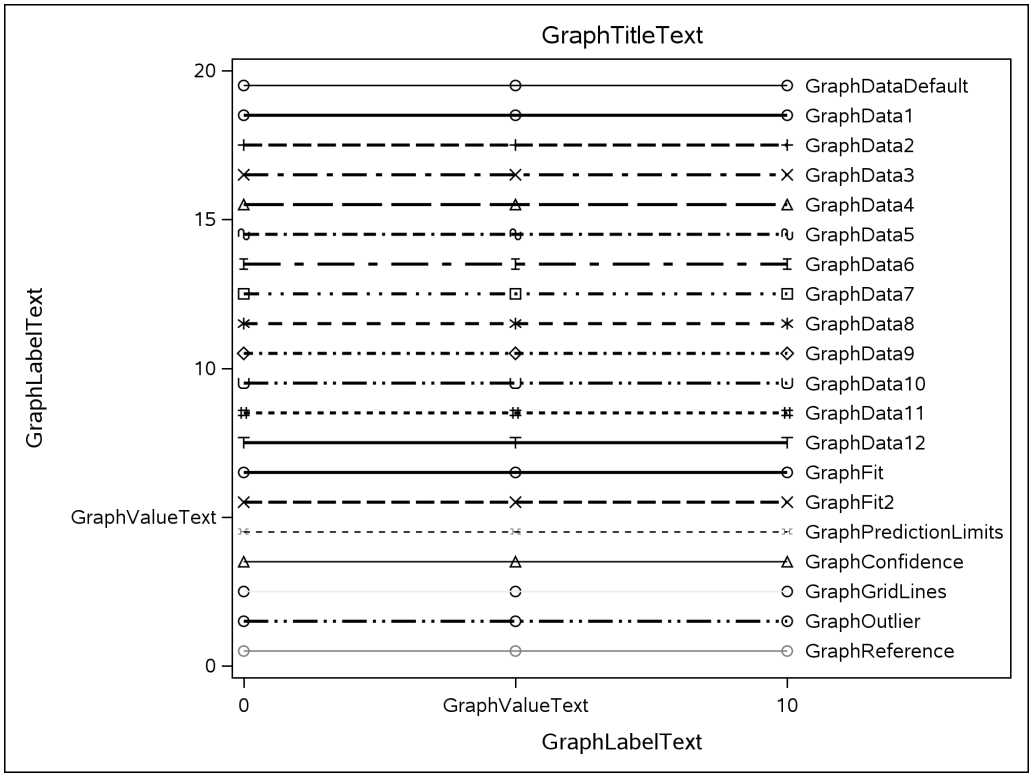

Figure 24.44 Attributes of Style Elements in the LISTING Style

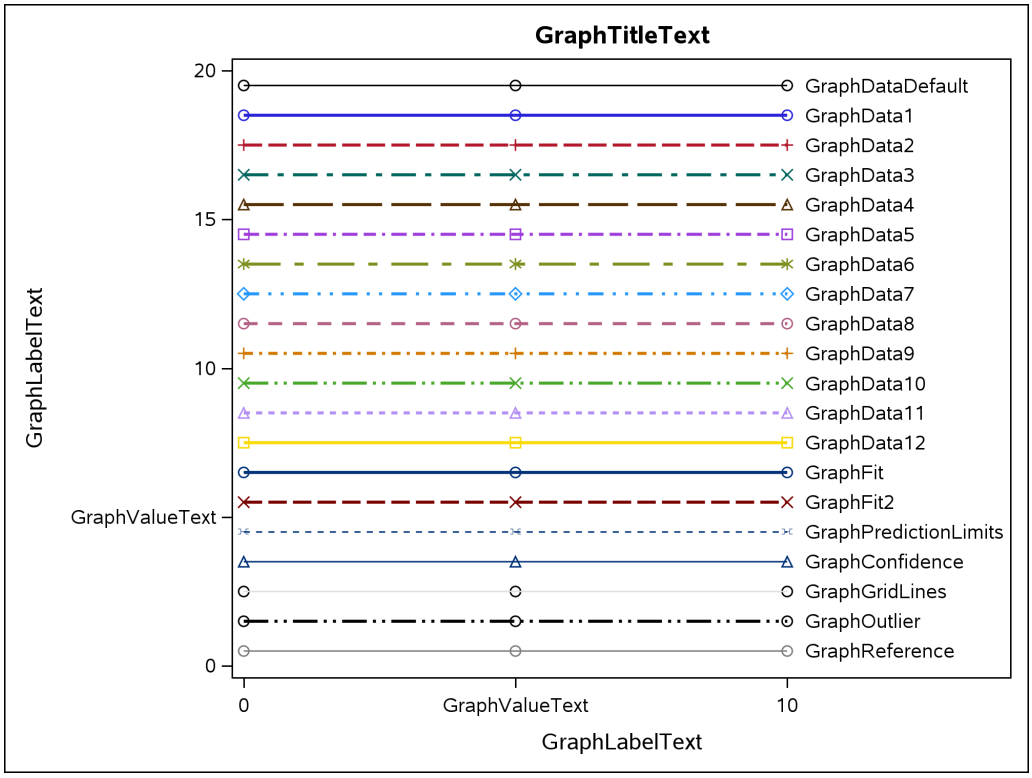

Figure 24.45 Attributes of Style Elements in the RTF Style

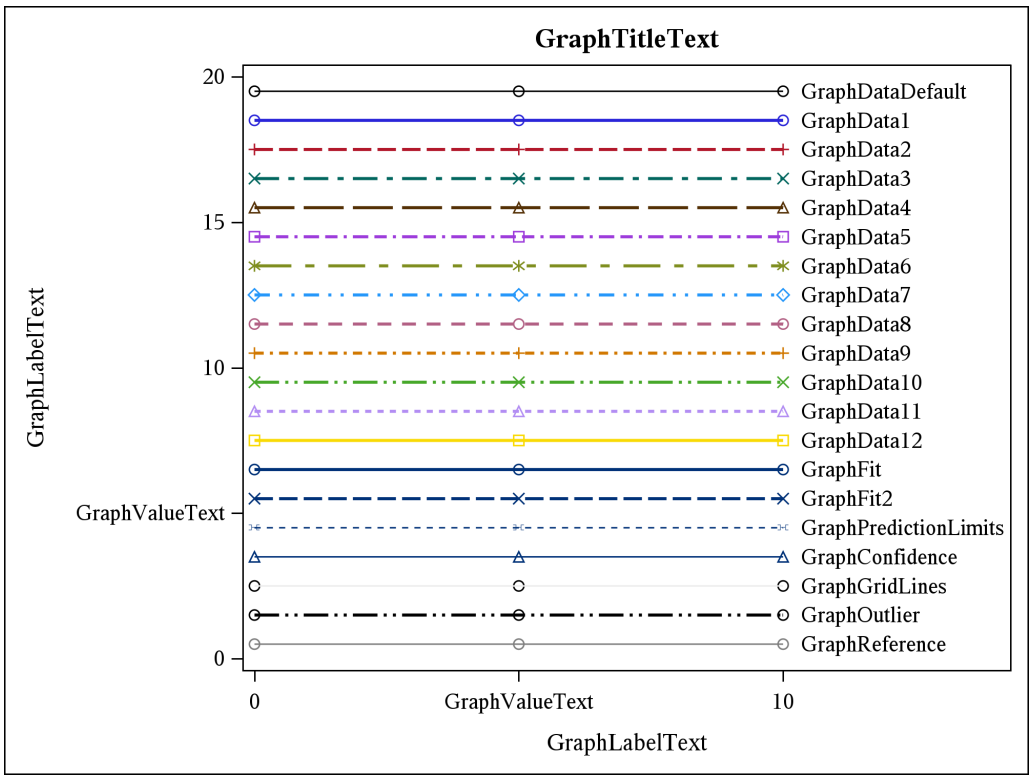

Figure 24.46 Attributes of Style Elements in the PEARL Style

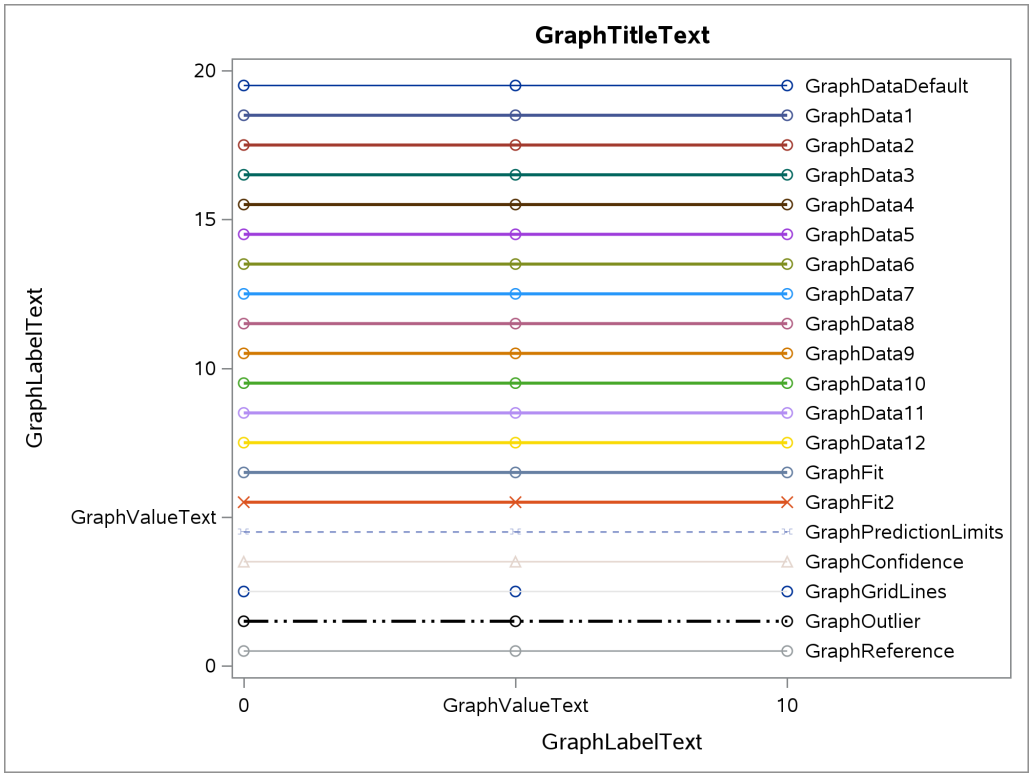

Figure 24.47 Attributes of Style Elements in the PEARLJ Style

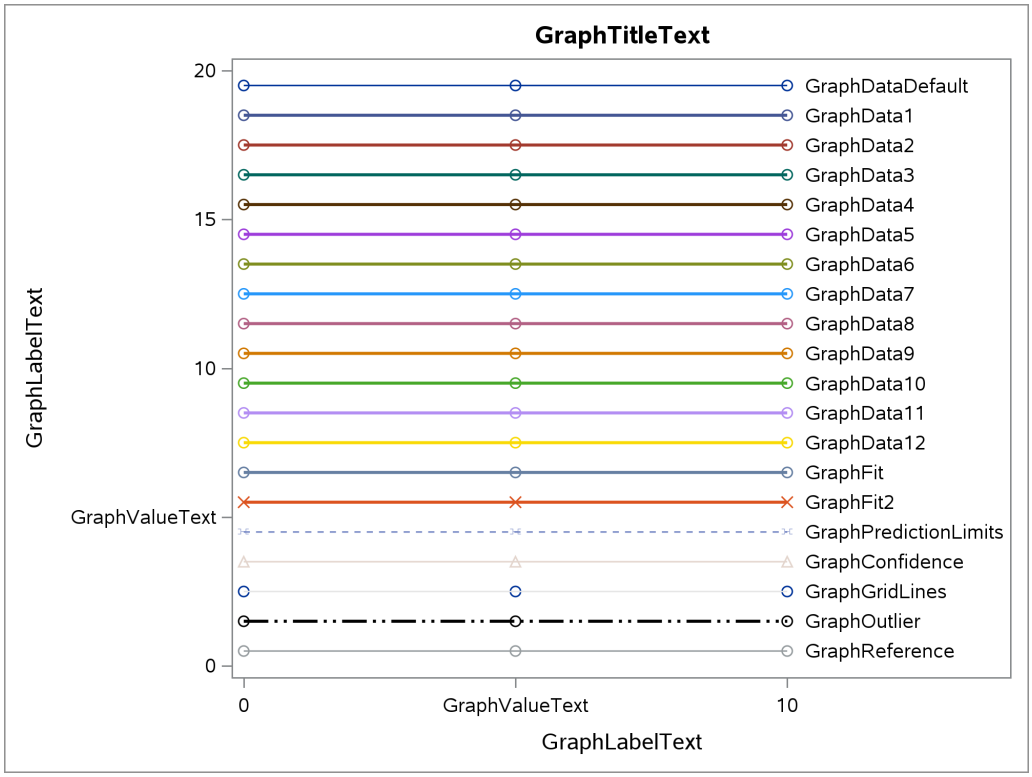

Figure 24.48 Attributes of Style Elements in the SAPPHIRE Style

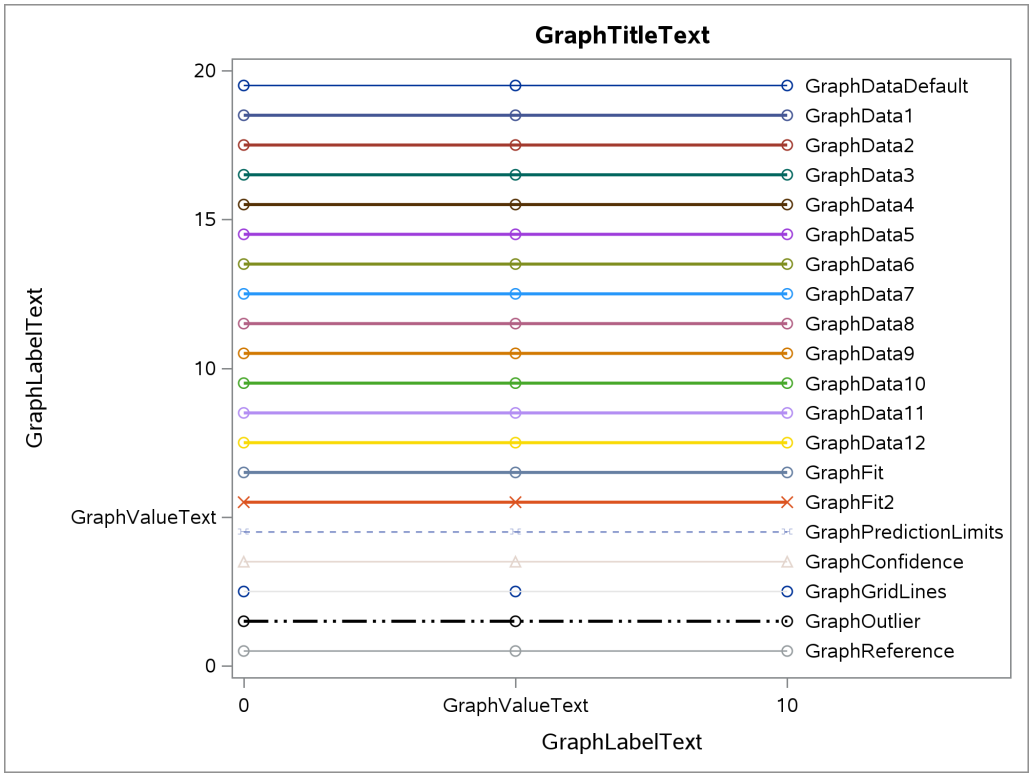

Figure 24.49 Markers, Lines, and Colors with Groups in the DEFAULT Style

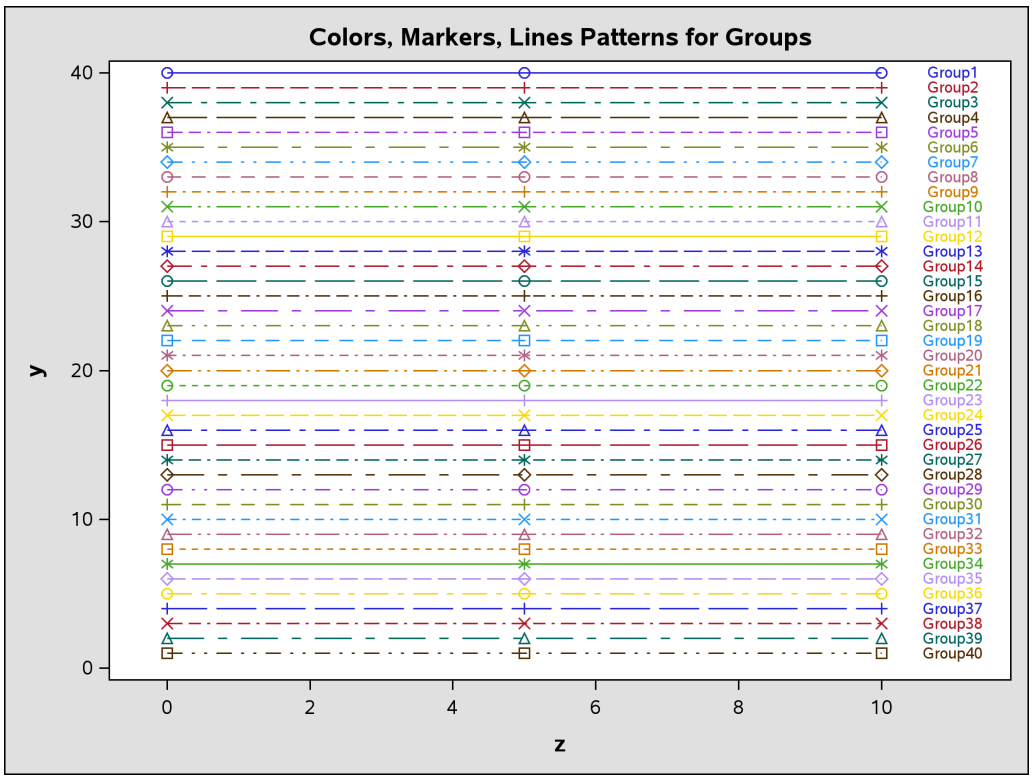

Figure 24.50 Markers, Lines, and Colors with Groups in the HTMLBLUE Style

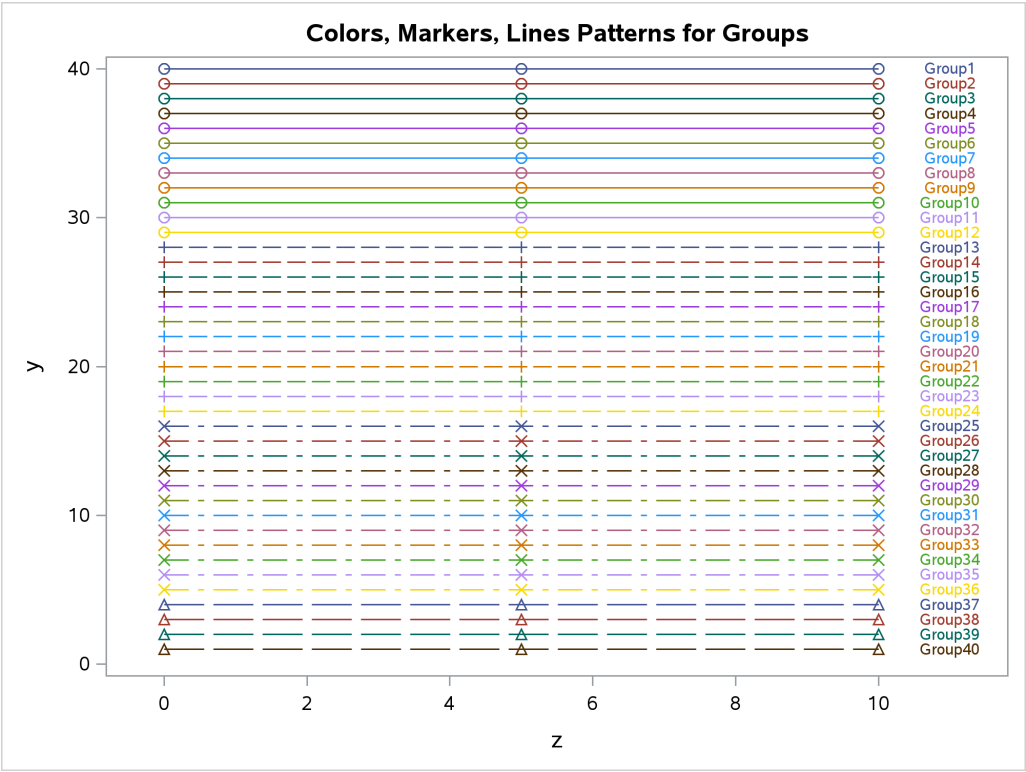

Figure 24.51 Markers, Lines, and Colors with Groups in the HTMLBLUECML Style

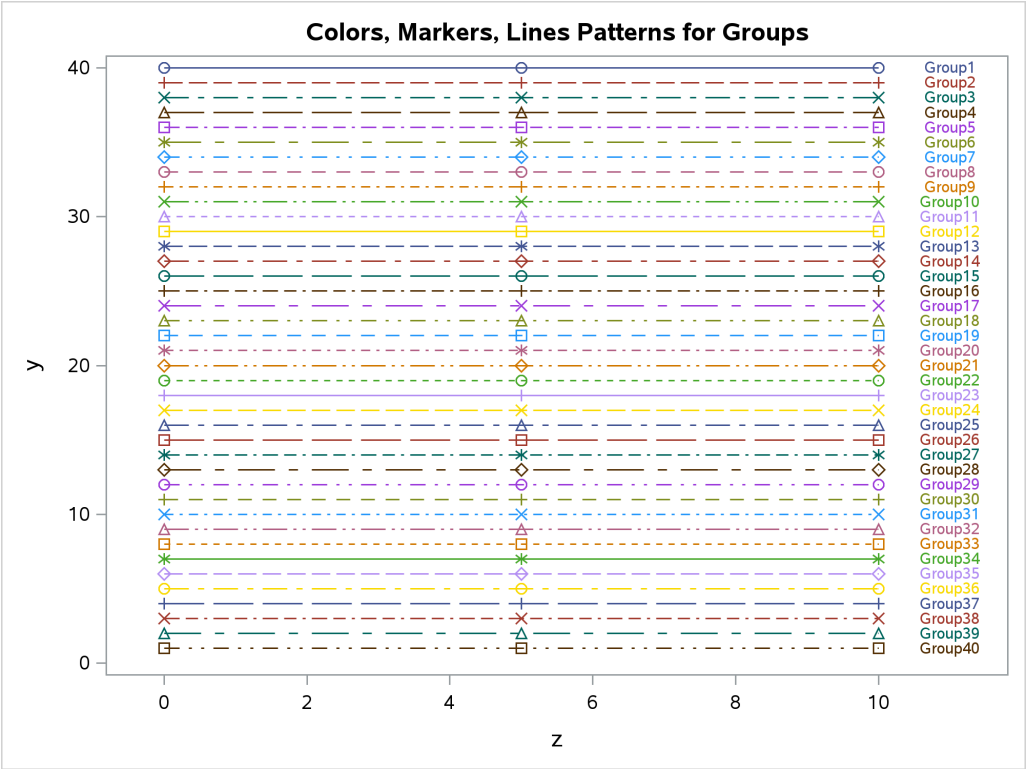

Figure 24.52 Markers, Lines, and Colors with Groups in the STATISTICAL Style

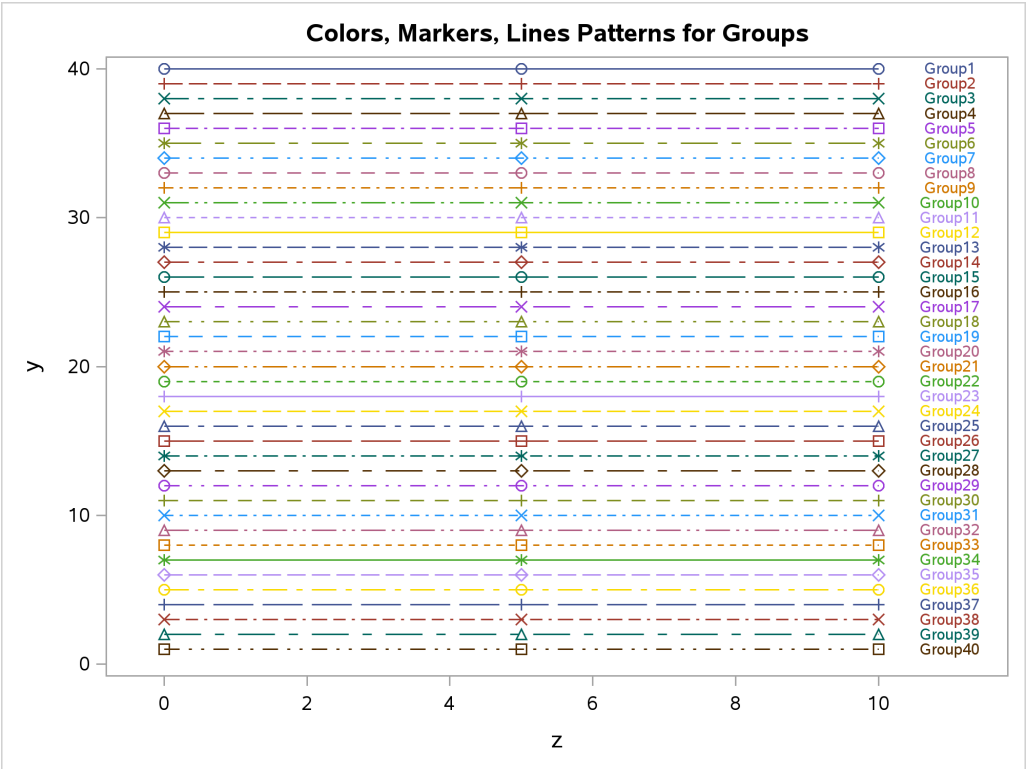

Figure 24.53 Markers, Lines, and Colors with Groups in the ANALYSIS Style

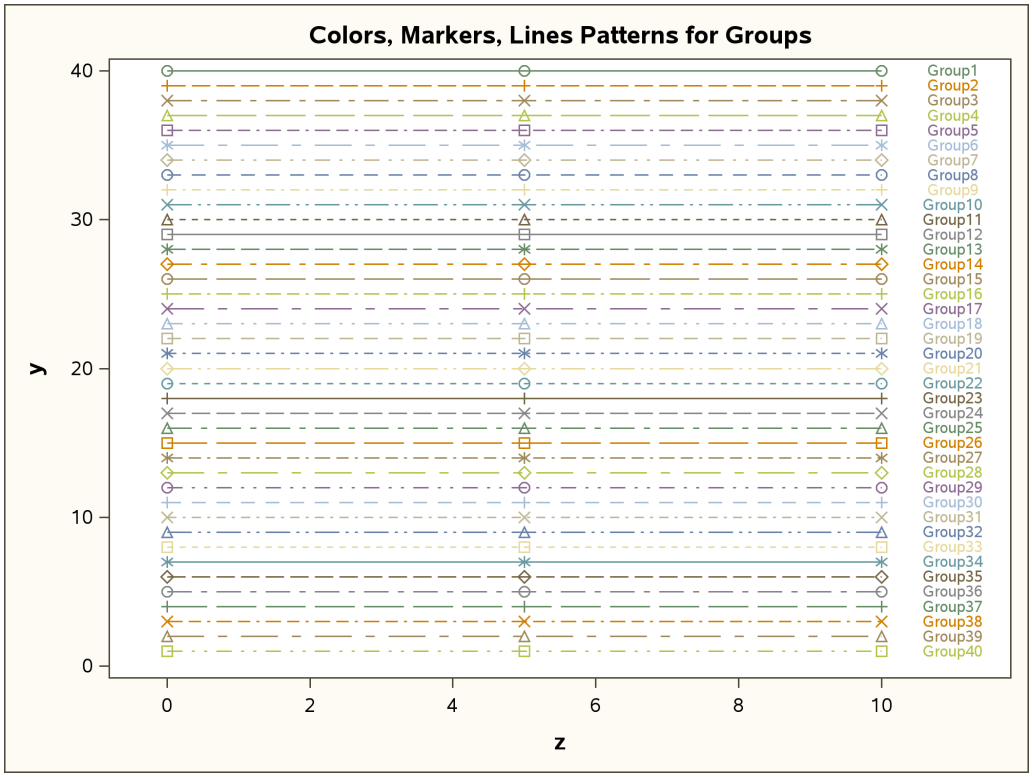

Figure 24.54 Markers, Lines, and Colors with Groups in the JOURNAL Style

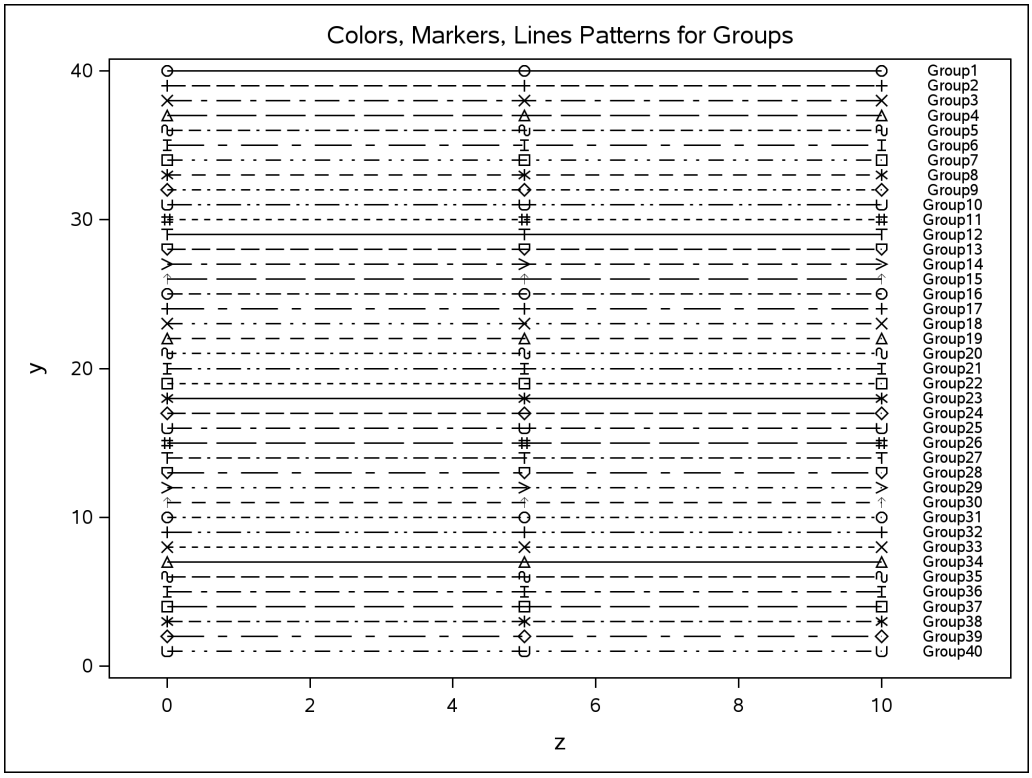

Figure 24.55 Markers, Lines, and Colors with Groups in the LISTING Style

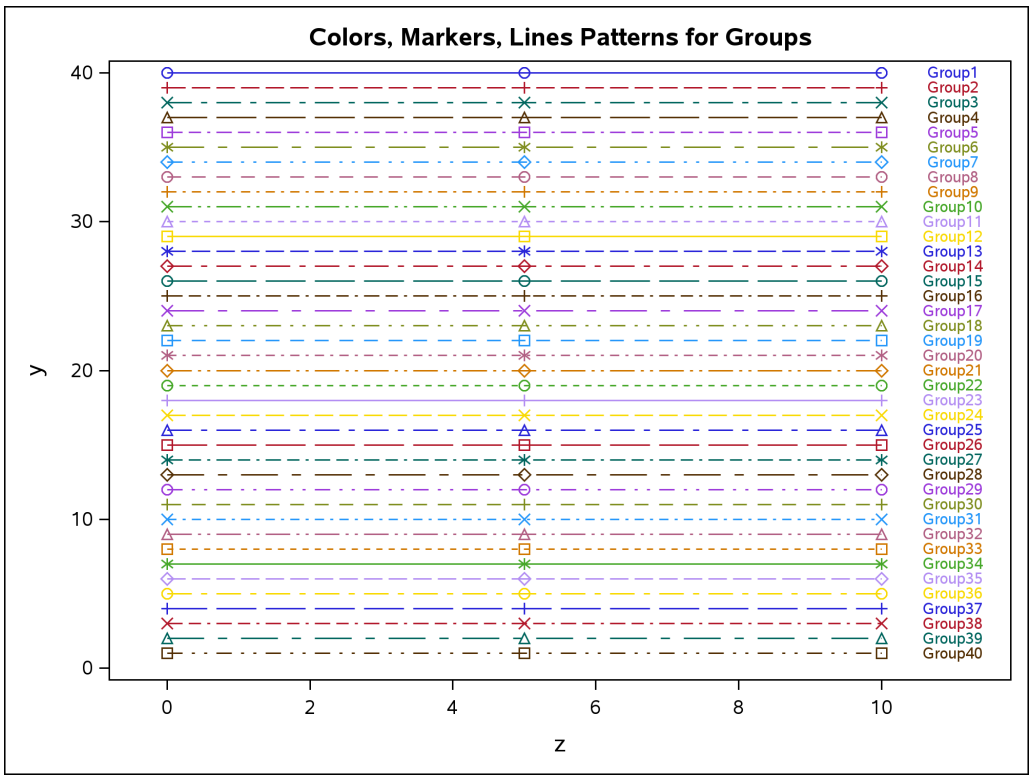

Figure 24.56 Markers, Lines, and Colors with Groups in the RTF Style

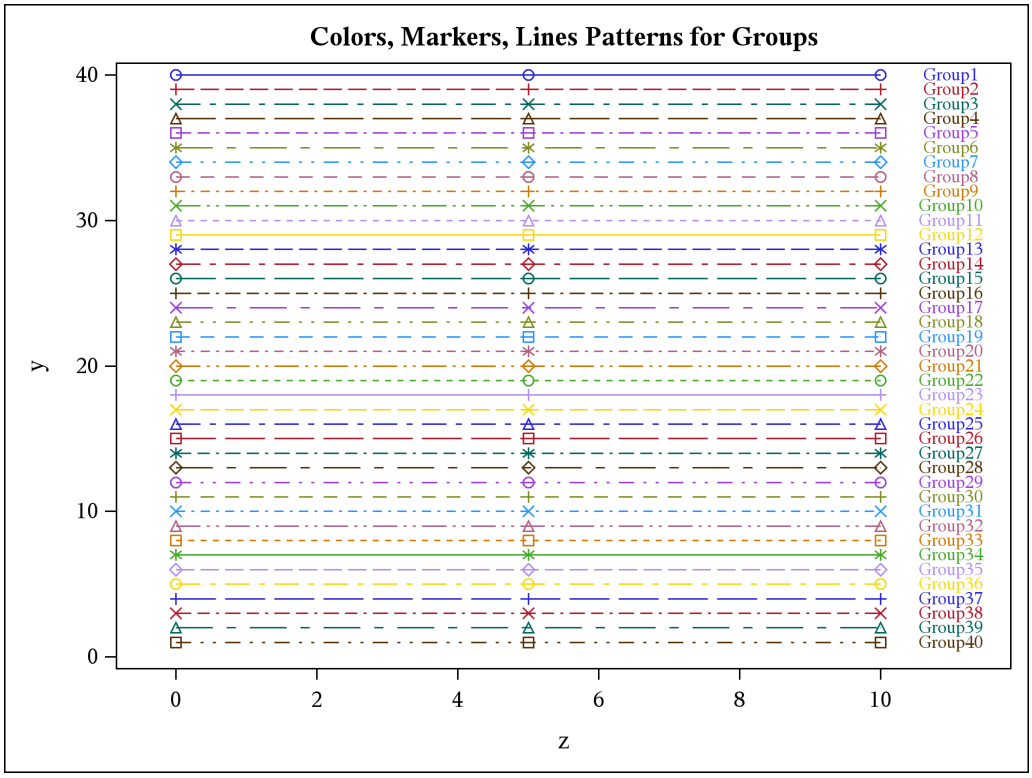

Figure 24.57 Markers, Lines, and Colors with Groups in the PEARL Style

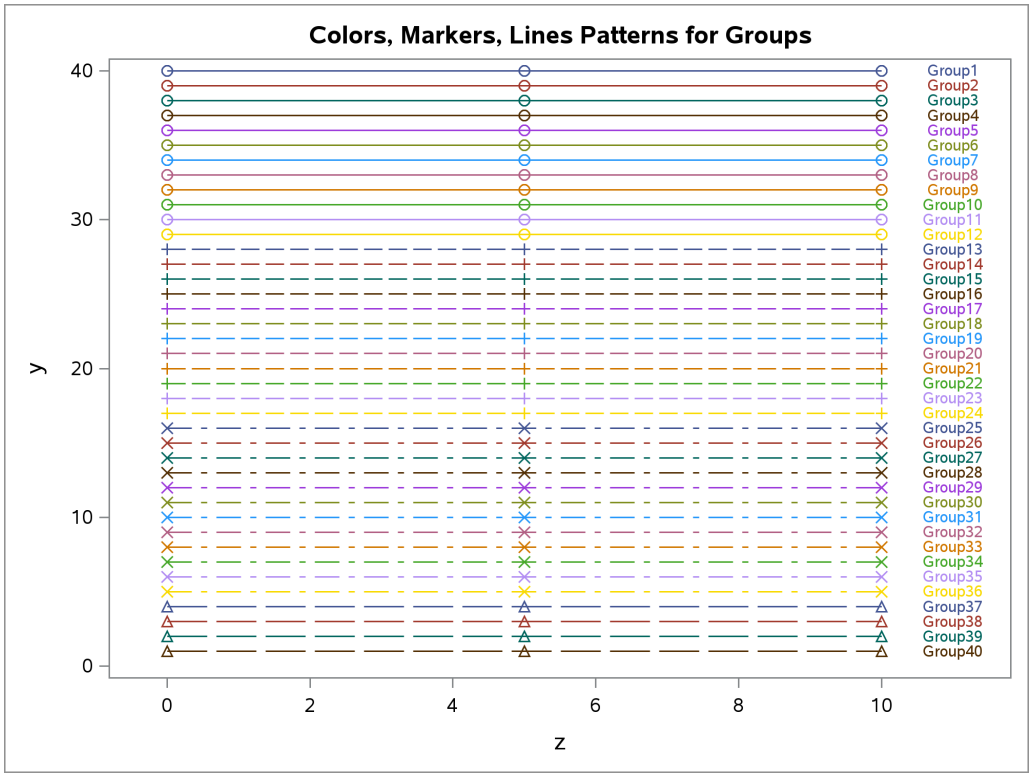

Figure 24.58 Markers, Lines, and Colors with Groups in the PEARLJ Style

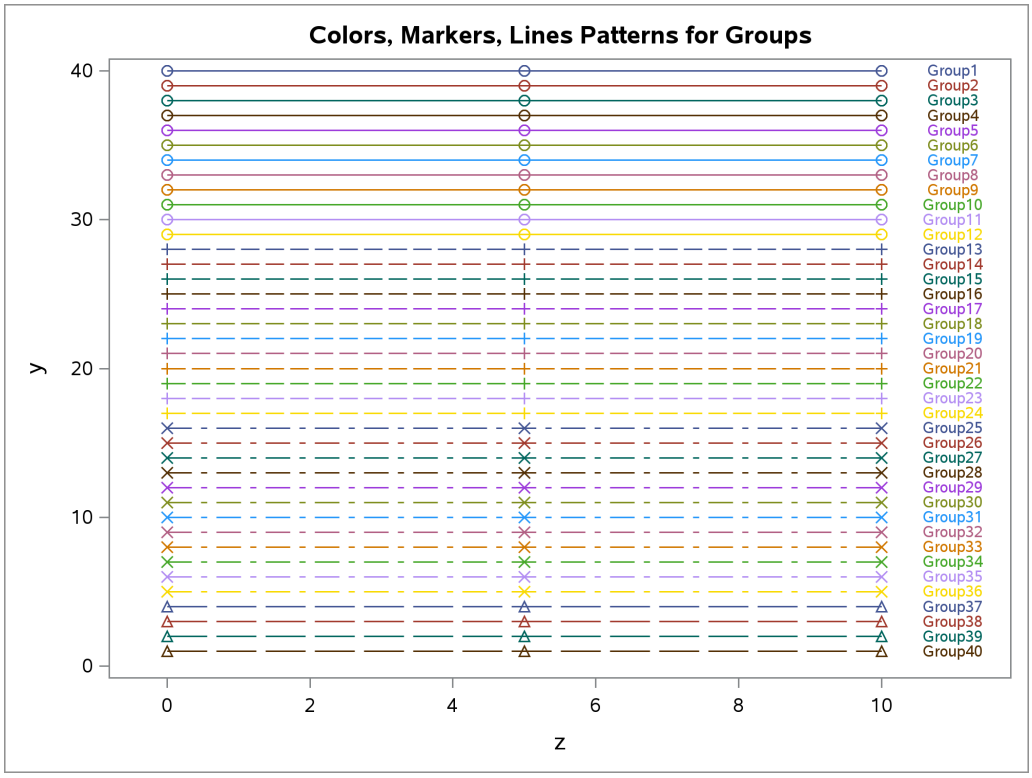

Figure 24.59 Markers, Lines, and Colors with Groups in the SAPPHIRE Style

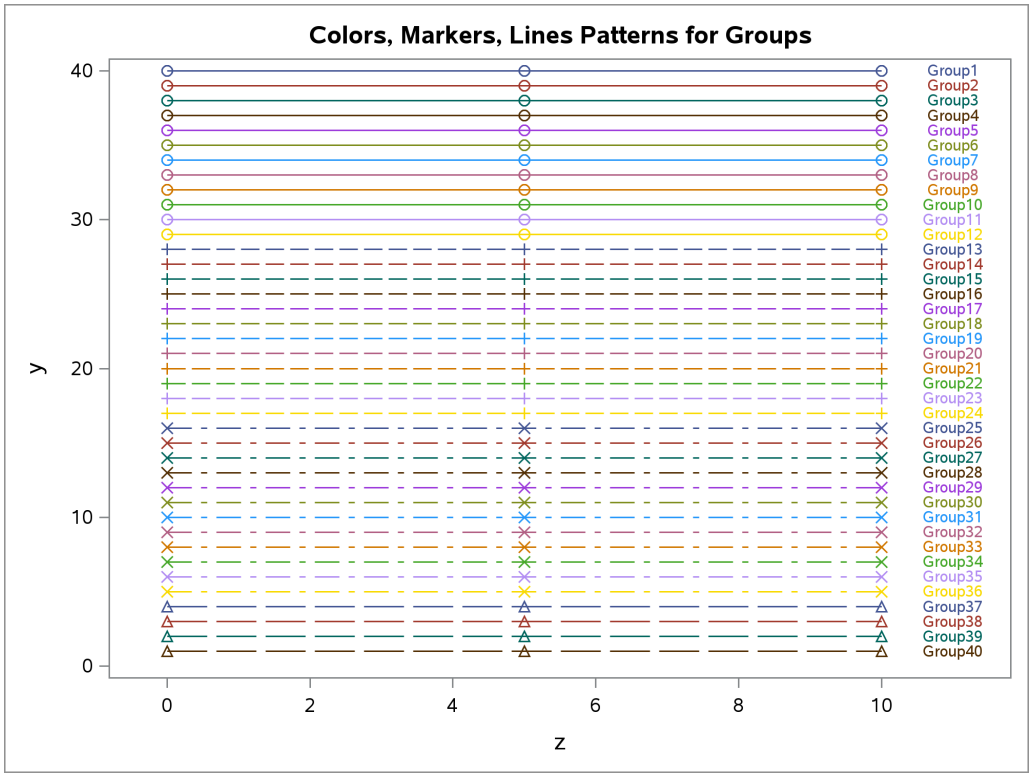

Modifying the HTMLBLUE Style

The HTMLBLUE style is an all-color style for the first 12 groups of observations. After each set of 12 groups, the line style and marker change for the next 12 groups. (See Figure 24.50.) The HTMLBLUECML style is a color style in which groups of observations are distinguished by simultaneous color, line style, and symbol changes. (See Figure 24.51.) For some graphs, you might want more differentiation than you get in an all-color style like HTMLBLUE but without the overkill differentiation of the HTMLBLUECML style and other styles. This section defines four new styles for this purpose:

HTMLBLUEL – line styles and colors vary together with fixed markers for each set of 11 groups.
HTMLBLUEM – markers and colors vary together with fixed line styles for each set of 11 groups.
HTMLBLUEFL – line styles and colors vary together with fixed filled markers for each set of 11 groups.
HTMLBLUEFM – filled markers and colors vary together with fixed line styles for each set of 5 groups.

The following statements show part of the style template for each of these styles:

```
define style Styles.HTMLBlueL;    parent = styles.htmlbluecml;
   style GraphFit2   from GraphFit2   /                          linestyle = 1;
   style GraphData1  from GraphData1  / markersymbol = "circle" linestyle = 1;
   style GraphData2  from GraphData2  / markersymbol = "circle" linestyle = 4;
   style GraphData3  from GraphData3  / markersymbol = "circle" linestyle = 8;
   style GraphData4  from GraphData4  / markersymbol = "circle" linestyle = 5;
   style GraphData5  from GraphData5  / markersymbol = "circle" linestyle = 14;
   style GraphData6  from GraphData6  / markersymbol = "circle" linestyle = 26;
```

```
   style GraphData7  from GraphData7  / markersymbol = "circle" linestyle = 15;
   style GraphData8  from GraphData8  / markersymbol = "circle" linestyle = 20;
   style GraphData9  from GraphData9  / markersymbol = "circle" linestyle = 41;
   style GraphData10 from GraphData10 / markersymbol = "circle" linestyle = 42;
   style GraphData11 from GraphData11 / markersymbol = "circle" linestyle = 2;
   style GraphData12 from GraphData12 / markersymbol = "square" linestyle = 1;
   style GraphData13 from GraphData1  / markersymbol = "square" linestyle = 4;
   style GraphData14 from GraphData2  / markersymbol = "square" linestyle = 8;
   style GraphData15 from GraphData3  / markersymbol = "square" linestyle = 5;
   . . .
end;
define style Styles.HTMLBlueM;    parent = styles.htmlbluecml;
   style GraphFit2   from GraphFit2   /                                linestyle = 1;
   style GraphData1  from GraphData1  / markersymbol = "circle"        linestyle = 1;
   style GraphData2  from GraphData2  / markersymbol = "square"        linestyle = 1;
   style GraphData3  from GraphData3  / markersymbol = "diamond"       linestyle = 1;
   style GraphData4  from GraphData4  / markersymbol = "asterisk"      linestyle = 1;
   style GraphData5  from GraphData5  / markersymbol = "plus"          linestyle = 1;
   style GraphData6  from GraphData6  / markersymbol = "triangle"      linestyle = 1;
   style GraphData7  from GraphData7  / markersymbol = "circlefilled"  linestyle = 1;
   style GraphData8  from GraphData8  / markersymbol = "starfilled"    linestyle = 1;
   style GraphData9  from GraphData9  / markersymbol = "squarefilled"  linestyle = 1;
   style GraphData10 from GraphData10 / markersymbol = "diamondfilled" linestyle = 1;
   style GraphData11 from GraphData11 / markersymbol = "trianglefilled" linestyle = 1;
   style GraphData12 from GraphData12 / markersymbol = "circle"        linestyle = 4;
   style GraphData13 from GraphData1  / markersymbol = "square"        linestyle = 4;
   style GraphData14 from GraphData2  / markersymbol = "diamond"       linestyle = 4;
   style GraphData15 from GraphData3  / markersymbol = "asterisk"      linestyle = 4;
   . . .
end;
define style Styles.HTMLBlueFL;    parent = styles.htmlbluecml;
   style GraphFit2   from GraphFit2   /                               linestyle = 1;
   style GraphData1  from GraphData1  / markersymbol = "circlefilled" linestyle = 1;
   style GraphData2  from GraphData2  / markersymbol = "circlefilled" linestyle = 4;
   style GraphData3  from GraphData3  / markersymbol = "circlefilled" linestyle = 8;
   style GraphData4  from GraphData4  / markersymbol = "circlefilled" linestyle = 5;
   style GraphData5  from GraphData5  / markersymbol = "circlefilled" linestyle = 14;
   style GraphData6  from GraphData6  / markersymbol = "circlefilled" linestyle = 26;
   style GraphData7  from GraphData7  / markersymbol = "circlefilled" linestyle = 15;
   style GraphData8  from GraphData8  / markersymbol = "circlefilled" linestyle = 20;
   style GraphData9  from GraphData9  / markersymbol = "circlefilled" linestyle = 41;
   style GraphData10 from GraphData10 / markersymbol = "circlefilled" linestyle = 42;
   style GraphData11 from GraphData11 / markersymbol = "circlefilled" linestyle = 2;
   style GraphData12 from GraphData12 / markersymbol = "starfilled"   linestyle = 1;
   style GraphData13 from GraphData1  / markersymbol = "starfilled"   linestyle = 4;
   style GraphData14 from GraphData2  / markersymbol = "starfilled"   linestyle = 8;
   style GraphData15 from GraphData3  / markersymbol = "starfilled"   linestyle = 5;
   . . .
end;
define style Styles.HTMLBlueFM;    parent = styles.htmlbluecml;
   style GraphFit2   from GraphFit2   /                                linestyle = 1;
   style GraphData1  from GraphData1  / markersymbol = "circlefilled"  linestyle = 1;
   style GraphData2  from GraphData2  / markersymbol = "starfilled"    linestyle = 1;
   style GraphData3  from GraphData3  / markersymbol = "squarefilled"  linestyle = 1;
```

```
   style GraphData4  from GraphData4  / markersymbol = "diamondfilled"  linestyle = 1;
   style GraphData5  from GraphData5  / markersymbol = "trianglefilled" linestyle = 1;
   style GraphData6  from GraphData6  / markersymbol = "circlefilled"   linestyle = 4;
   style GraphData7  from GraphData7  / markersymbol = "starfilled"     linestyle = 4;
   style GraphData8  from GraphData8  / markersymbol = "squarefilled"   linestyle = 4;
   style GraphData9  from GraphData9  / markersymbol = "diamondfilled"  linestyle = 4;
   style GraphData10 from GraphData10 / markersymbol = "trianglefilled" linestyle = 4;
   style GraphData11 from GraphData11 / markersymbol = "circlefilled"   linestyle = 8;
   style GraphData12 from GraphData12 / markersymbol = "starfilled"     linestyle = 8;
   style GraphData13 from GraphData1  / markersymbol = "squarefilled"   linestyle = 8;
   style GraphData14 from GraphData2  / markersymbol = "diamondfilled"  linestyle = 8;
   style GraphData15 from GraphData3  / markersymbol = "trianglefilled" linestyle = 8;
   . . .
end;
```

New **GraphData***n* style elements are created that inherit colors from the **GraphData1** through **GraphData12** style elements. The line styles and markers are explicitly set in the new style templates. The **style GraphFit2 from GraphFit2 / linestyle = 1** statement creates a solid second fit line. You can remove that statement if you prefer a dashed second fit line.

The following statements use SAS macros to generate these four new styles:

```
proc template;
   %let m = circle square diamond asterisk plus triangle circlefilled
            starfilled squarefilled diamondfilled trianglefilled;
   %let ls = 1 4 8 5 14 26 15 20 41 42 2;
   %macro makestyle;
      %let l = %eval(%sysfunc(mod(&k,12))+1);
      %let k = %eval(&k+1);
      style GraphData&k from GraphData&l /
            linestyle=%scan(&ls, &j) markersymbol="%scan(&m, &i)";
      %mend;
   define style styles.HTMLBlueL;
      parent=styles.htmlbluecml;
      style GraphFit2 from GraphFit2 / linestyle = 1;
      %macro htmlbluel;
         %let k = 0;
         %do i = 1 %to 11; %do j = 1 %to 11; %makestyle %end; %end;
         %mend;
      %htmlbluel
      end;
   define style styles.HTMLBlueM;
      parent=styles.htmlbluecml;
      style GraphFit2 from GraphFit2 / linestyle = 1;
      %macro htmlbluem;
         %let k = 0;
         %do j = 1 %to 11; %do i = 1 %to 11; %makestyle %end; %end;
         %mend;
      %htmlbluem
      end;
   %let m = circlefilled starfilled squarefilled diamondfilled trianglefilled;
   define style styles.HTMLBlueFL;
      parent=styles.htmlbluecml;
      style GraphFit2 from GraphFit2 / linestyle = 1;
```

```
      %macro htmlbluel;
         %let k = 0;
         %do i = 1 %to 5; %do j = 1 %to 11; %makestyle %end; %end;
         %mend;
      %htmlbluel
      end;
   define style styles.HTMLBlueFM;
      parent=styles.htmlbluecml;
      style GraphFit2 from GraphFit2 / linestyle = 1;
      %macro htmlbluem;
         %let k = 0;
         %do j = 1 %to 11; %do i = 1 %to 5; %makestyle %end; %end;
         %mend;
      %htmlbluem
   end;
run;
```

The %LET m statement provides the list of markers. The %LET ls statement provides the list of line styles. The MAKESTYLE macro makes the *k*th style element from the **GraphData***n* style element for $n = \text{mod}(k - 1, 12) + 1$. The remaining macros vary markers and line styles in the appropriate order over the elements in each list.

The following step, which is used in the section "ODS Style Comparisons" on page 733, is used along with the different styles to produce Figure 24.60 through Figure 24.63:

```
proc sgplot data=x2;
   title 'Colors, Markers, Lines Patterns for Groups';
   series  y=y x=x / group=group markers;
   scatter y=y x=z / group=group markerchar=l;
run;
```

Figure 24.60 Markers, Lines, and Colors with Groups in the HTMLBLUEL Style

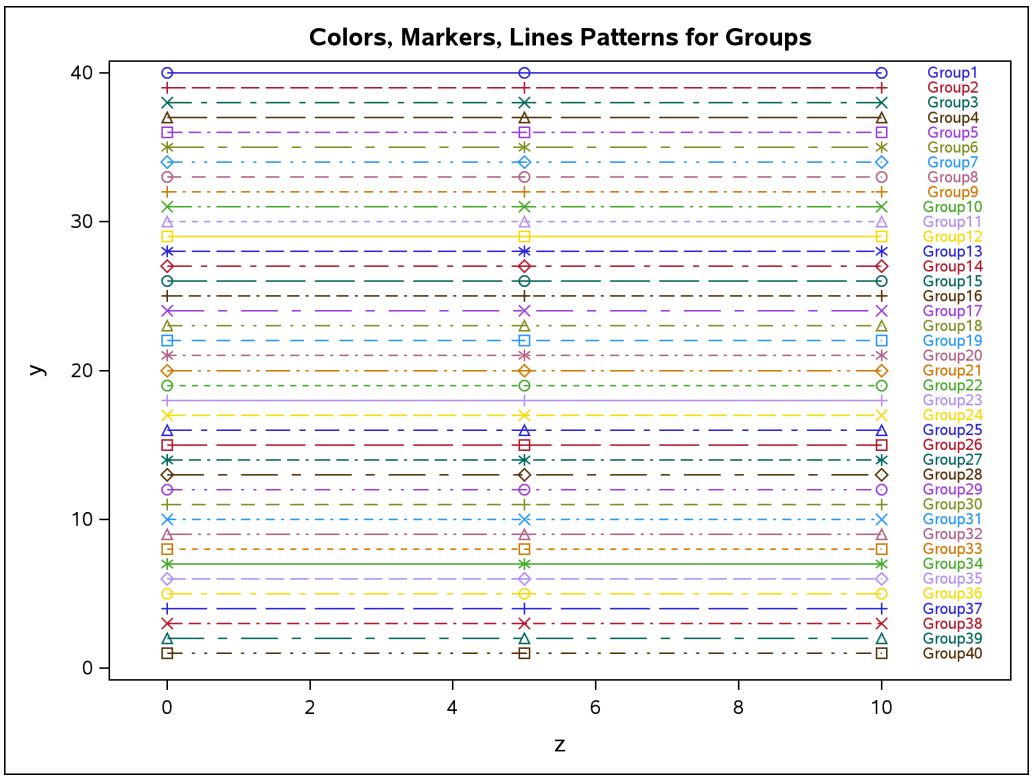

Figure 24.61 Markers, Lines, and Colors with Groups in the HTMLBLUEM Style

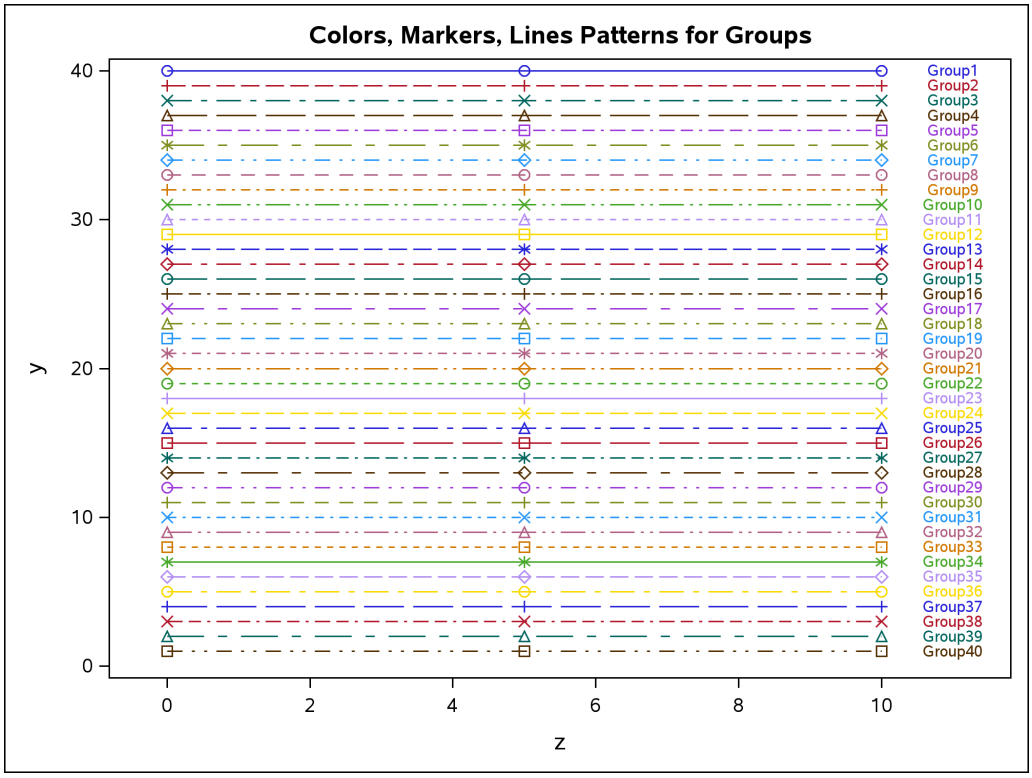

Figure 24.62 Markers, Lines, and Colors with Groups in the HTMLBLUEFL Style

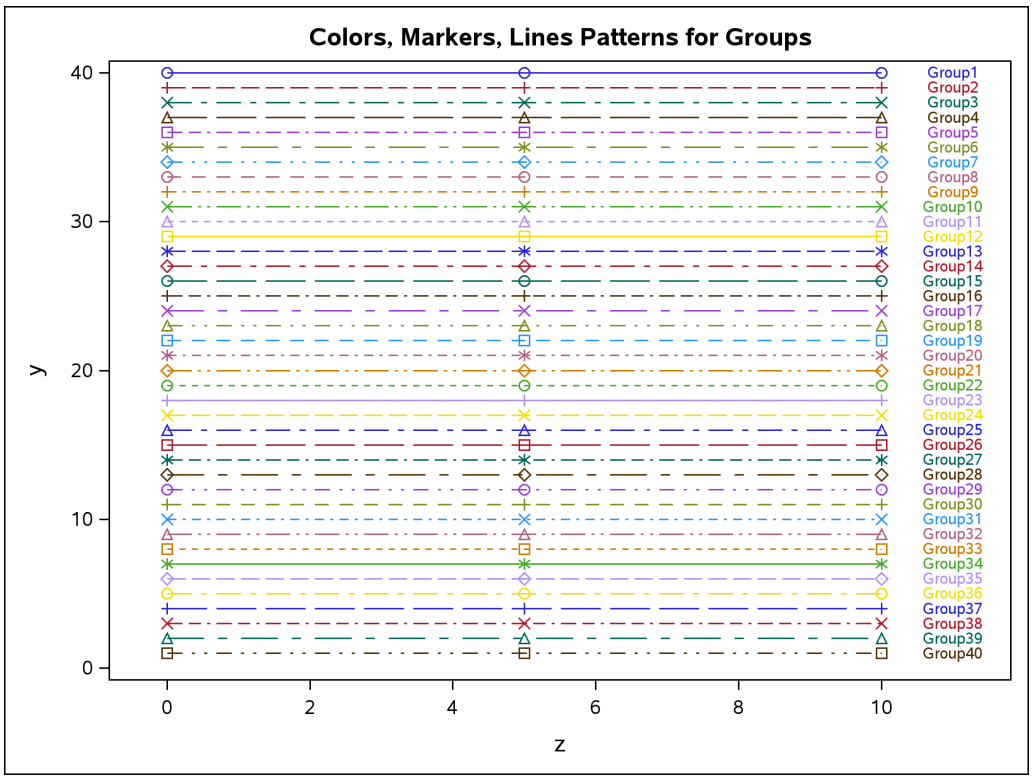

Figure 24.63 Markers, Lines, and Colors with Groups in the HTMLBLUEFM Style

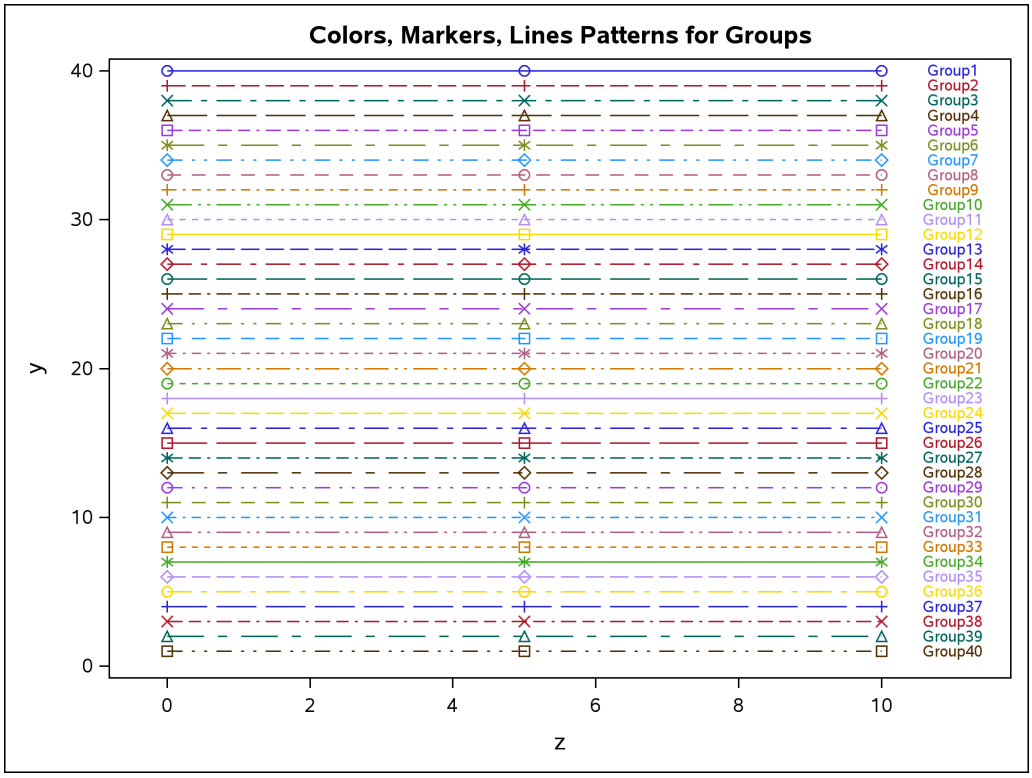

ODS Style Template Modification Macro

The **%ModStyle** macro provides easy ways to customize the style elements (**GraphData1**–**GraphData***n*) that control how groups of observations are distinguished. Examples of using the **%ModStyle** macro can be found in the section "Changing the Default Markers and Lines" on page 765. Also see Kuhfeld (2009) for more information about this macro. **NOTE:** Many of the things that you can do with the **%ModStyle** macro are more conveniently accomplished by using the ATTRPRIORITY= option in the ODS GRAPHICS statement and the options in the STYLEATTRS statement in PROC SGPLOT or the corresponding options in the BEGINGRAPH statement in PROC TEMPLATE. Specify the ATTRPRIORITY=NONE option in the ODS GRAPHICS statement or specify an ATTRPRIORITY=NONE style such as STATISTICAL in the PARENT= option if you want to distinguish groups by marker or line changes. For more information, see the sections "ODS Styles" on page 688 and "Grouped Scatter Plot with PROC SGPLOT" on page 684.

You do not need to include autocall macros (for example, by specifying a **%include** statement). You can call them directly after they are properly installed. If your site has installed the autocall libraries that are supplied by the SAS System and uses the standard configuration of SAS software, you need to ensure that the SAS System option MAUTOSOURCE is in effect before you can begin using the autocall macros. For more information about autocall libraries, see the *SAS Macro Language: Reference*. For information about installing autocall macros, consult your host documentation.

The **%ModStyle** macro has the following options:

COLORS=*color-list*

specifies a space-delimited list of colors for markers and lines. If you do not specify this option, then the colors from the parent style are used. You can specify the colors by using any SAS color notation, such as CX*rrggbb*.

COLORS=GRAYS generates seven distinguishable grayscale colors from blackest to whitest. The colors should be mixed up to be more easily distinguished when you need fewer colors, but you can do that in your own COLORS= list. The HLS (hue/light/saturation) coding generates colors by setting hue and saturation to 0 and incrementing the lightness for each gray. You can also use the keywords BLUES, PURPLES, MAGENTAS, REDS, ORANGES, YELLOWS, GREENS, and CYANS to generate seven colors that have a fixed hue and a saturation of AA (hex).

COLORS=SHADES INT generates seven colors as described previously, except that you specify an integer $0 \leq \text{INT} < 360$. (See *SAS/GRAPH: Reference*.) The available hues are GRAY (or GREY), BLUE=0, PURPLE=30, MAGENTA=60, RED=120, ORANGE=150, YELLOW=180, GREEN=240, and CYAN=300.

DISPLAY=*n*

specifies whether to display the generated template. By default, the template is not displayed. Specify DISPLAY=1 to display the generated template.

FILLCOLORS=*color-list*

specifies a space-delimited list of colors for bands and fills. If you do not specify this option, then the colors from the parent style are used.

Fill colors from the parent style are designed to work well with the colors from the parent style. If you specify a COLORS= list, then you might also want to redefine the FILLCOLORS= list. You need to have at least as many fill colors as you have colors (any extra fill colors are ignored). Two shortcuts are

available: FILLCOLORS=COLORS uses the COLORS= colors for the fills (your confidence bands should have transparency for this shortcut to be useful), and FILLCOLORS=LIGHTCOLORS modifies the lightness that is associated with each color generated by COLORS=SHADES (this is allowed only with COLORS=SHADES).

LINESTYLES=*line-style-list*

specifies a space-delimited list of line styles. The default is

```
LineStyles=Solid MediumDash MediumDashShortDash LongDash
           DashDashDot LongDashShortDash DashDotDot Dash
           ShortDashDot MediumDashDotDot ShortDash
```

Line style numbers can range from 1 to 46. Some line styles have names associated with them. You can specify either the name or the number for the following number/name pairs: 1 Solid, 2 ShortDash, 4 MediumDash, 5 LongDash, 8 MediumDashShortDash, 14 DashDashDot, 15 DashDotDot, 20 Dash, 26 LongDashShortDash, 34 Dot, 35 ThinDot, 41 ShortDashDot, and 42 MediumDashDotDot.

MARKERS=*marker-list*

specifies a space-delimited list of marker symbols. By default, **`Markers=Circle Plus X Triangle Square Asterisk Diamond`**. The available marker symbols are listed in the *SAS Graph Template Language: Reference*. Two shortcuts are available: MARKERS=FILLED is an alias for the specification **`Markers=CircleFilled TriangleFilled SquareFilled DiamondFilled StarFilled HomeDownFilled`**, and MARKERS=EMPTY is an alias for the specification **`Markers=Circle Triangle Square Diamond Star HomeDown`**.

NAME=*style-name*

specifies the name of the new style that you are creating. This name is used when you specify the style in an ODS destination statement (for example, ODS HTML STYLE=*style-name*). By default, NAME=NEWSTYLE.

NUMBEROFGROUPS=*n*

specifies *n*, the number of **`GraphData`***n* style elements to create. The **`GraphData1`**–**`GraphData`***n* style elements contain *n* combinations of colors, markers, and line styles. By default, 32 combinations are created.

PARENT=*style-name*

specifies the parent style. The new style inherits most of its attributes from the parent style. By default, PARENT=DEFAULT (which is the top-level parent style for all the styles that are recommended for statistical graphics). If your goal is to change colors or create an all-color style, you can use any style as the parent style. However, if your goal is to change markers or line styles without creating an all-color style, do not use ATTRPRIORITY='Color' style (such as HTMLBLUE, PEARL, PEARLJ, and SAPPHIRE) as a parent.

TYPE=*type-specification*

specifies how your new style cycles through colors, markers, and line styles. The values that are specified in the TYPE= option are case-sensitive ("by" is lowercase and the L, C, and M are uppercase). By default, TYPE=LMbyC.

These first three methods work well in all plots, because cycling line styles and markers together ensures that both scatter-plot markers and series plot lines are distinguishable:

CLM cycles through colors, line styles, and markers simultaneously. The first group uses the first color, line style, and marker; the second group uses the second color, line style, and marker; and so on. This is the method that most styles use; it corresponds to ATTRPRIORITY='None'.

LMbyC fixes line style and marker, cycles through colors, and then moves to the next line style and marker. This is the default; it creates a style where the first groups are distinguished entirely by color, which corresponds to ATTRPRIORITY='Color'.

CbyLM fixes color, cycles through line style and marker, and then moves to the next color. This option uses the smaller of the number of line styles or the number of markers when cycling within a color.

The following two methods might not work well in all plots:

CbyLbyM fixes color and line style, then cycles through markers, increments line style, and then cycles through markers again. After all line styles have been used, this option moves to the next color and continues.

LbyMbyC fixes line style and marker, then cycles through colors, increments marker, and then cycles through colors again. After all markers have been used, this option moves to the next line style and continues. This is closest to the legacy SAS/GRAPH method.

Varying Colors and Markers but Not Lines

Many styles are designed to make color plots where lines, markers, and groups of observations can be distinguished even when the plot is sent to a black-and-white printer. Hence, you can distinguish lines by both their colors and their line patterns. Similarly, you can distinguish markers by both their colors and their symbols. This is not true in the HTMLBLUE, PEARL, PEARLJ, and SAPPHIRE styles, which are all-color styles. You can easily make any style an all-color style by specifying the ATTRPRIORITY= option in the ODS GRAPHICS statement. For more information, see the section "Attribute Priority and Overriding How Groups Are Distinguished" on page 720.

Alternatively, you can make an all-color style by using the %MODSTYLE autocall macro. It creates a new style by modifying a parent style and reordering the colors, line patterns, and marker symbols in the **GraphData***n* style elements (see the section "Some Common ODS Style Elements" on page 728). The macro is documented in the section "ODS Style Template Modification Macro" on page 761.

The following example illustrates how you can use the macro and is taken from the section "Fitting a Curve through a Scatter Plot" on page 10556 in Chapter 126, "The TRANSREG Procedure." The data come from an experiment in which nitrogen oxide emissions from a single-cylinder engine are measured for various combinations of fuel and equivalence ratio.

The following statements fit separate curves for each group and produce Figure 24.64 and Figure 24.65:

```
ods graphics on;
ods html style=htmlblue; * You can instead specify other destinations
                           such as LISTING, PDF, or RTF;
proc transreg data=sashelp.Gas plots=fit(nocli noclm);
   model identity(nox) = class(Fuel / zero=none) * pbspline(EqRatio);
```

```
run;
ods html close;

%modstyle(parent=statistical, name=StatColor, linestyles=solid, type=CLM)
ods html style=StatColor;
proc transreg data=sashelp.Gas plots=fit(nocli noclm);
   model identity(nox) = class(Fuel / zero=none) * pbspline(EqRatio);
run;
ods html close;
```

The first PROC TRANSREG step uses the HTMLBLUE style (which is an ATTRPRIORITY='Color' style) to create the fit plot in Figure 24.64, which uses only colors to distinguish each group. Then the macro creates a new style called STATCOLOR, which inherits attributes from the STATISTICAL style (which is an ATTRPRIORITY='None' style).

When you specify TYPE=CLM and LINESTYLES=SOLID, the %MODSTYLE macro varies colors and markers but displays only solid lines. In Figure 24.65, which is created by using the modified style, the groups are differentiated by color and marker type.

Figure 24.64 Fit Plot with the HTMLBLUE Style

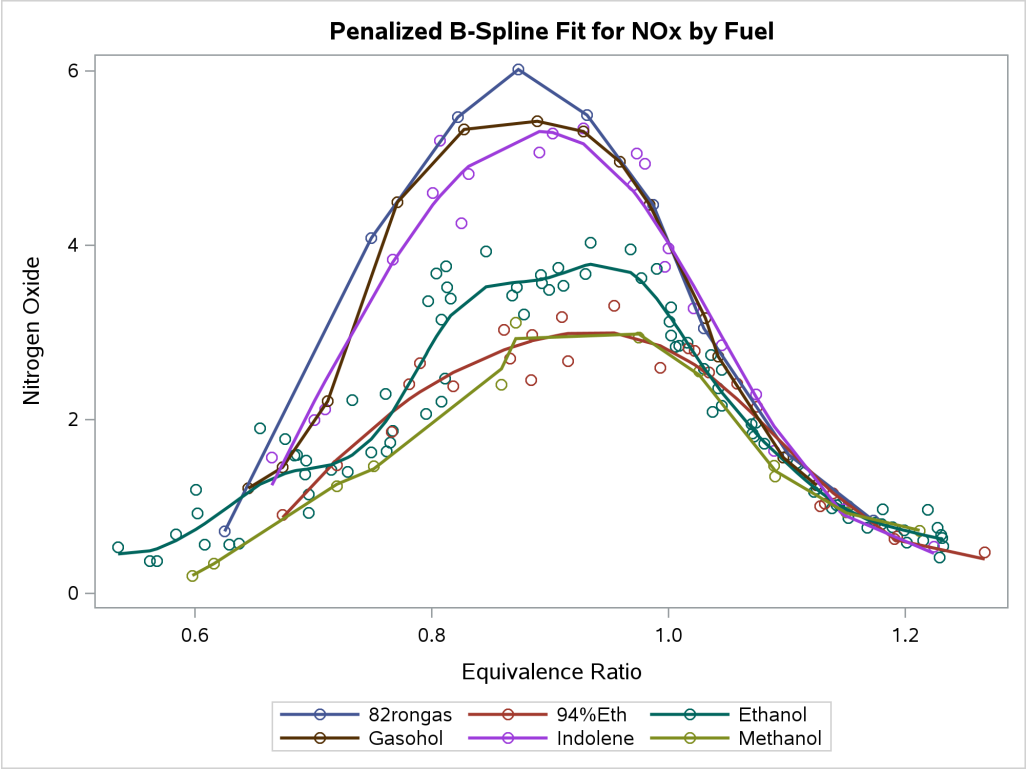

Figure 24.65 Fit Plot with the Modified Style

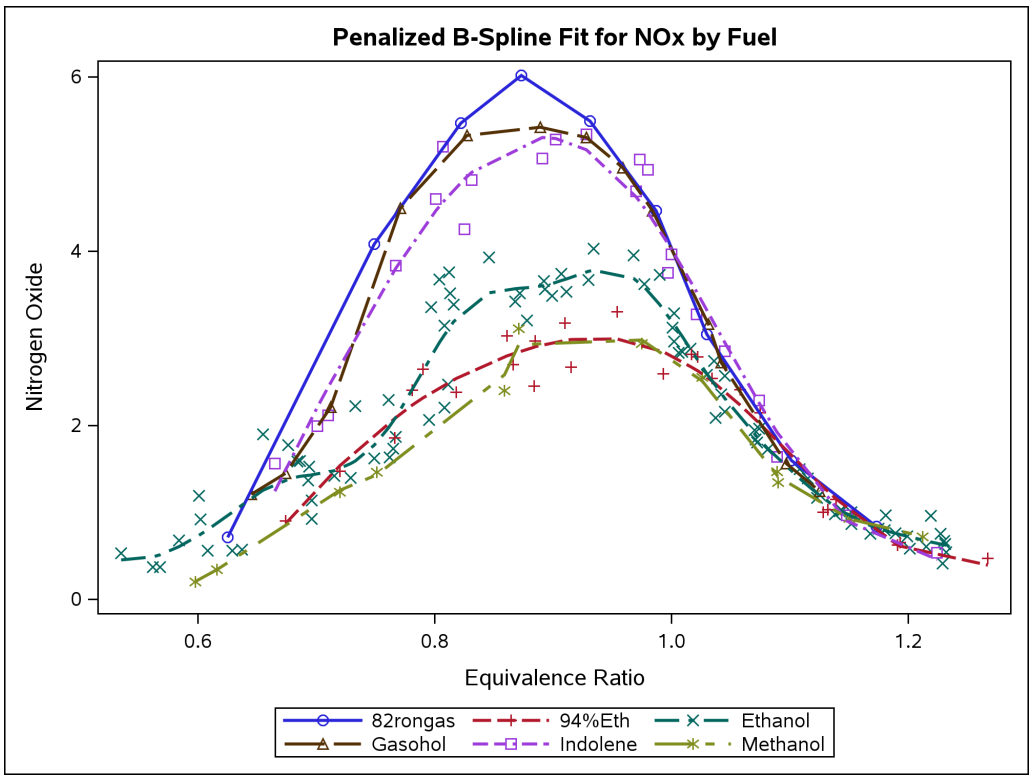

Changing the Default Markers and Lines

You can use the %MODSTYLE macro to change markers and line styles. This example creates a new style called MARKSTYLE that inherits attributes from the STATISTICAL style but uses a different set of markers. The following statements create artificial data, change the marker list, and display the results:

```
data x;
   do g = 1 to 12;
      do x = 1 to 10;
         y = 13 - g + sin(x * 0.1 * g);
         output;
      end;
   end;
run;

%modstyle(name=markstyle, parent=statistical, type=CLM,
          markers=star plus circle square diamond starfilled
                  circlefilled squarefilled diamondfilled)

ods html style=markstyle; * You can instead specify other destinations
                            such as LISTING, PDF, or RTF;
proc sgplot;
   title 'Modified Marker List';
   loess y=y x=x / group=g;
```

```
run;
ods html close;
```

The NAME= option specifies the new style name, and the PARENT= option specifies the parent style. The TYPE= option controls the method of cycling through colors, lines, and markers. The default, TYPE=LMbyC, fixes (holds constant) the line styles and markers while cycling through the color list. This example uses TYPE=CLM to cycle through colors, line styles, and markers (holding none of them constant). Other TYPE= values are described in the section "ODS Style Template Modification Macro" on page 761. The values that are specified in the TYPE= option are case-sensitive ("by" is lowercase and the L, C, and M are uppercase). The new marker list is specified in the MARKERS= option. The results are displayed in Figure 24.66. The marker list is reused in the tenth and subsequent groups because only nine markers are defined.

Figure 24.66 A Modified Style with a New List of Markers

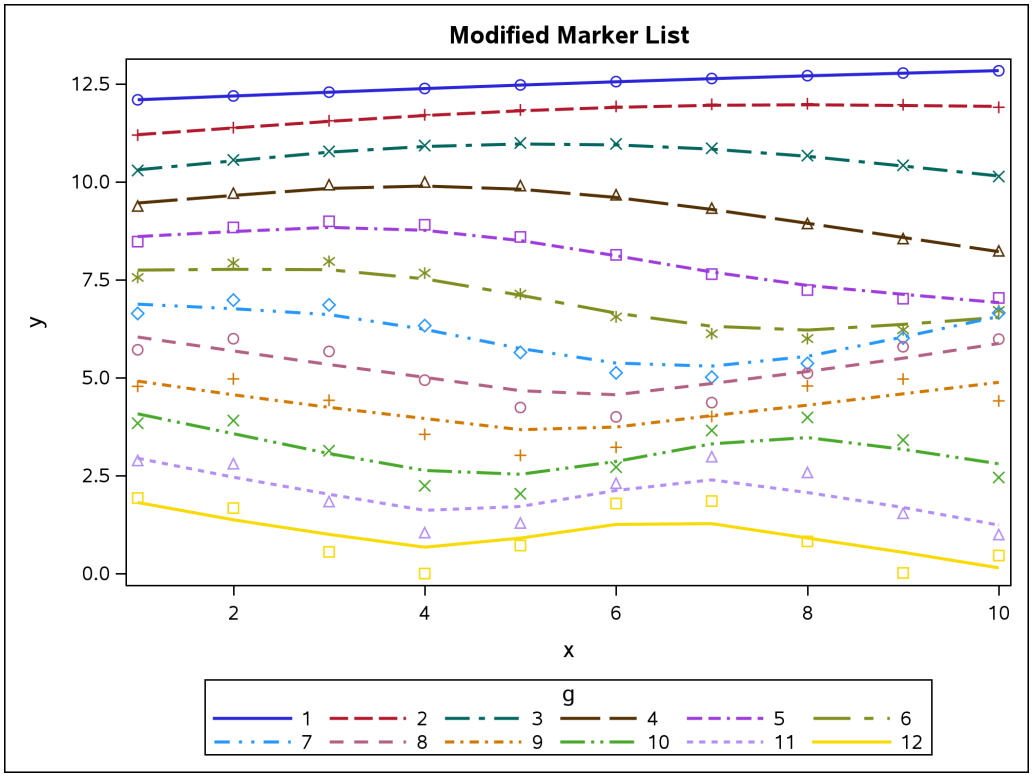

The following statements create a new style called LINESTYLE that inherits attributes from the STATISTICAL style and changes the line list:

```
%modstyle(name=linestyle, parent=statistical, type=CLM,
          linestyles=Solid LongDash MediumDash Dash ShortDash Dot ThinDot)

ods html style=linestyle; * You can instead specify other destinations
                            such as LISTING, PDF, or RTF;
proc sgplot;
   title 'Modified Line Style List';
   loess y=y x=x / group=g;
run;
ods html close;
```

The new line list is specified in the LINESTYLES= option. The results are displayed in Figure 24.67. In this example, each of the first seven groups uses a dash that is shorter than the dash in the previous group. The line list is reused in the eighth and subsequent groups because only seven line patterns are defined.

Figure 24.67 Modified Style with a New List of Line Styles

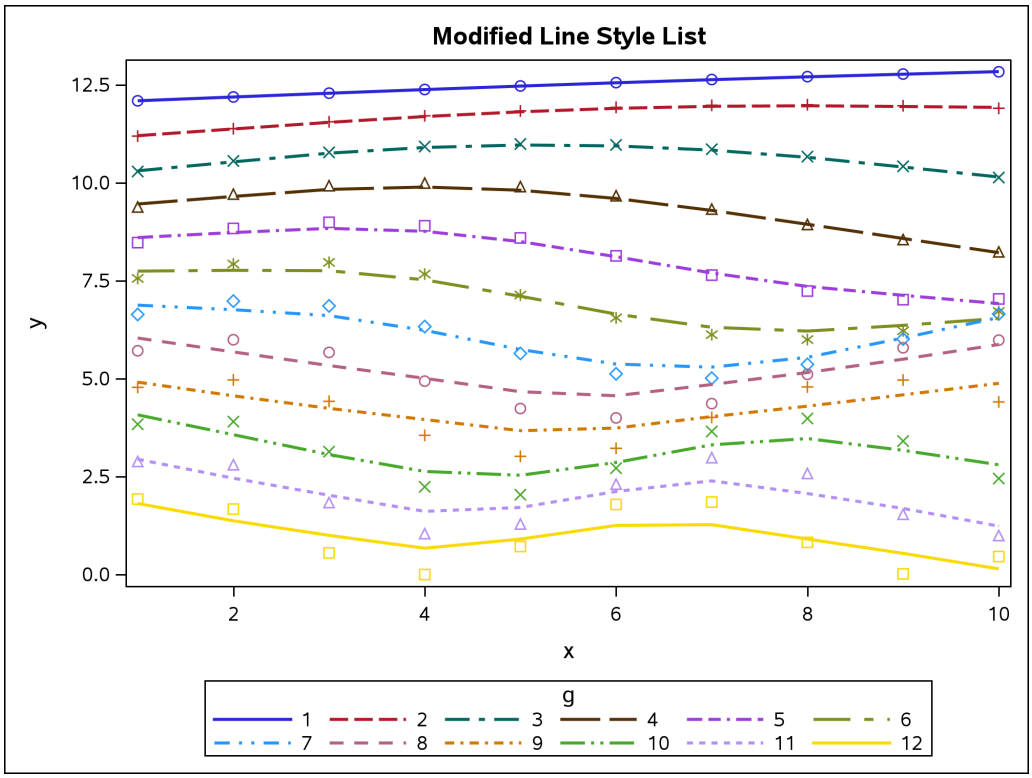

You can learn more about style modification by examining the new styles, as in the following example:

```
proc template;
   source styles.markstyle;
   source styles.linestyle;
run;
```

The results show the definitions of **GraphData1** through **GraphData32** that the macro created. An abridged listing of the results follows:

```
define style Styles.Markstyle;
   parent = Styles.statistical;
   . . .
   style GraphData1 /
      markersymbol = "star"
      linestyle = 1
      contrastcolor = ColorStyles('c1')
      color = FillStyles('f1');
   . . .
   style GraphData32 /
      markersymbol = "diamond"
      linestyle = 42
      contrastcolor = ColorStyles('c8')
```

```
      color = FillStyles('f8');
end;

define style Styles.Linestyle;
   parent = Styles.statistical;
   . . .
   style GraphData1 /
      markersymbol = "circle"
      linestyle = 1
      contrastcolor = ColorStyles('c1')
      color = FillStyles('f1');
   . . .
   style GraphData32 /
      markersymbol = "triangle"
      linestyle = 20
      contrastcolor = ColorStyles('c8')
      color = FillStyles('f8');
end;
```

You can use the NUMBEROFGROUPS= option in the %MODSTYLE macro to control the number of **GraphData***n* style elements that the new style creates.

Modifying Graph Fonts in Styles

You can modify an ODS style to customize the general appearance of plots that ODS Graphics produces, just as you can modify a style to customize the general appearance of ODS tables. This section shows you how to customize fonts that are used in graphs. The following step displays the HTMLBLUE style and its parent styles, STATISTICAL and DEFAULT:

```
proc template;
   source Styles.HTMLBlue / expand;
run;
```

If you search for “font”, you find the style elements that control graph fonts:

```
class GraphFonts /
   'NodeDetailFont' = ("<sans-serif>, <MTsans-serif>",7pt)
   'NodeInputLabelFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'NodeLabelFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'NodeTitleFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'GraphDataFont' = ("<sans-serif>, <MTsans-serif>",7pt)
   'GraphUnicodeFont' = ("<MTsans-serif-unicode>",9pt)
   'GraphValueFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'GraphLabel2Font' = ("<sans-serif>, <MTsans-serif>",10pt)
   'GraphLabelFont' = ("<sans-serif>, <MTsans-serif>",10pt)
   'GraphFootnoteFont' = ("<sans-serif>, <MTsans-serif>",10pt)
   'GraphTitleFont' = ("<sans-serif>, <MTsans-serif>",11pt,bold)
   'GraphTitle1Font' = ("<sans-serif>, <MTsans-serif>",14pt,bold)
   'GraphAnnoFont' = ("<sans-serif>, <MTsans-serif>",10pt);
```

The fonts **GraphTitle1Font** and **GraphLabel2Font** are not used by ODS Graphics. The following fonts are the ones that are usually used for the text in most graphs:

- **GraphDataFont** is the smallest font. It is used for text that needs to be small (labels for points in

scatter plots, labels for contours, and so on).

- **GraphValueFont** is the next-largest font. It is used for axis value (tick mark) labels and legend entry labels.
- **GraphLabelFont** is the next-largest font. It is used for axis labels and legend titles.
- **GraphFootnoteFont** is the next-largest font. It is used for all footnotes.
- **GraphTitleFont** is the largest font. It is used for all titles.
- **GraphUnicodeFont** is used for special characters. (See the section "Unicode and Special Characters" on page 878 in Chapter 25, "ODS Graphics Template Modification.")

The following statements define a style named NEWSTYLE that replaces the graph fonts in the STATISTICAL style by Times New Roman fonts, which are available in the Windows operating system:

```
proc template;
   define style Styles.NewStyle;
      parent=Styles.Statistical;
      class GraphFonts  /
         'GraphAnnoFont'     = ("<MTserif>, Times New Roman",10pt)
         'GraphDataFont'     = ("<MTserif>, Times New Roman", 7pt)
         'GraphUnicodeFont'  = ("<MTserif>, Times New Roman", 9pt)
         'GraphValueFont'    = ("<MTserif>, Times New Roman", 9pt)
         'GraphLabel2Font'   = ("<MTserif>, Times New Roman",10pt)
         'GraphLabelFont'    = ("<MTserif>, Times New Roman",10pt)
         'GraphFootnoteFont' = ("<MTserif>, Times New Roman",10pt)
         'GraphTitleFont'    = ("<MTserif>, Times New Roman",11pt)
         'GraphTitle1Font'   = ("<MTserif>, Times New Roman",14pt)
         'NodeTitleFont'     = ("<MTserif>, Times New Roman", 9pt)
         'NodeLabelFont'     = ("<MTserif>, Times New Roman", 9pt)
         'NodeLinkLabelFont' = ("<MTserif>, Times New Roman", 9pt)
         'NodeInputLabelFont'= ("<MTserif>, Times New Roman", 9pt)
         'NodeDetailFont'    = ("<MTserif>, Times New Roman", 7pt);
   end;
run;
```

For more information about the DEFINE, PARENT, and CLASS statements, see the *SAS Graph Template Language: Reference*.

The "Getting Started" section in Chapter 107, "The ROBUSTREG Procedure," creates the following data set to illustrate the use of PROC ROBUSTREG for robust regression:

```
data stack;
   input x1 x2 x3 y @@;
   datalines;
80 27 89 42   80 27 88 37   75 25 90 37   62 24 87 28   62 22 87 18
62 23 87 18   62 24 93 19   62 24 93 20   58 23 87 15   58 18 80 14
58 18 89 14   58 17 88 13   58 18 82 11   58 19 93 12   50 18 89  8
50 18 86  7   50 19 72  8   50 19 79  8   50 20 80  9   56 20 82 15
70 20 91 15
;
```

The following statements create a Q-Q plot that uses the HTMLBLUE style (see Figure 24.68) and the NEWSTYLE style (see Figure 24.69):

```
ods html style=HTMLBlue;
ods graphics on;
proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
ods html close;

ods html style=NewStyle;
proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
ods html close;
```

Figure 24.68 Q-Q Plot That Uses the HTMLBLUE Style

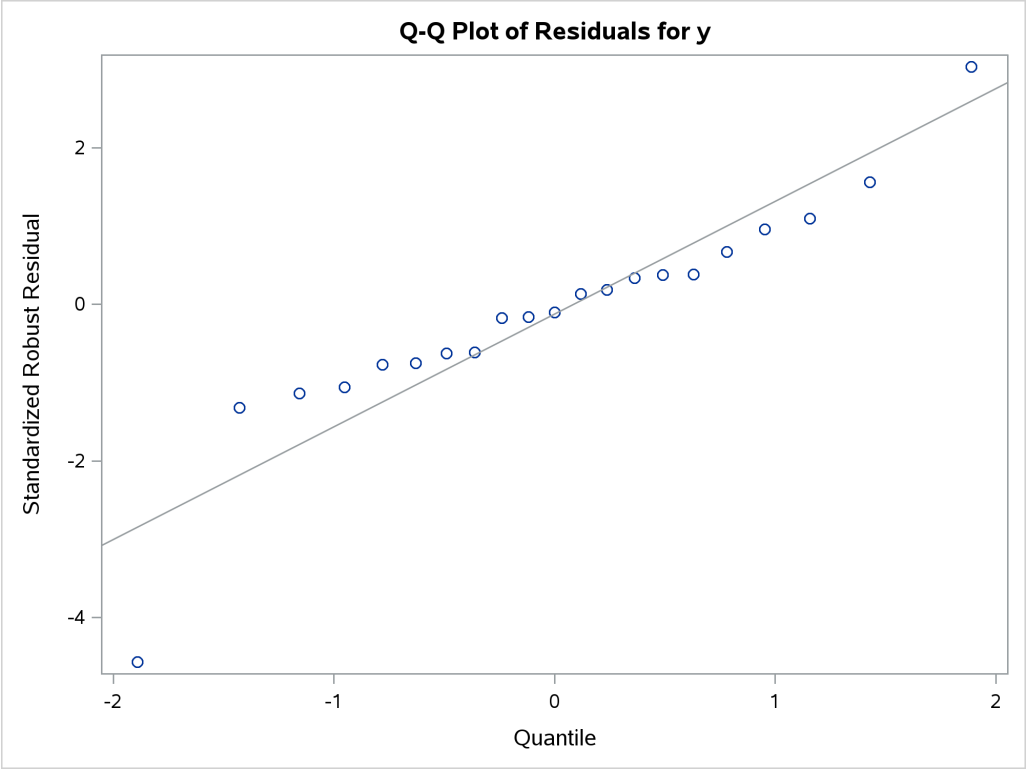

Figure 24.69 Q-Q Plot That Uses the NEWSTYLE Style

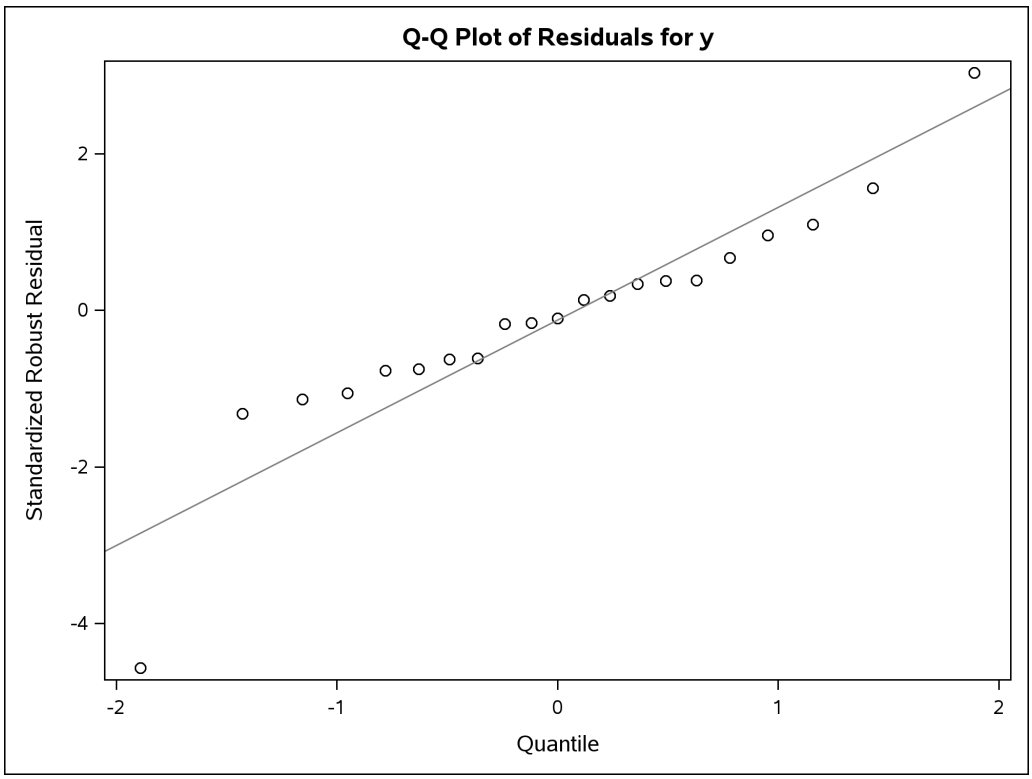

Although this example illustrates the use of a style with graphical output from a particular procedure, a style is applied to *all* your output (graphs and tables) in the destination for which you specify the style. For information about specifying a default style for all your output, see the section "Changing the Default Style" on page 773.

Modifying Other Graph Elements in Styles

This section illustrates how to modify other style elements for graphics, specifically the style element **GraphReference**, which controls the attributes of reference lines. You can run the following statements to learn more about the **GraphReference** style element:

```
proc template;
   source styles.HTMLBlue;
run;
```

The following are the first two lines of the source listing:

```
define style Styles.HTMLBlue;
   parent = styles.statistical;
```

There is no mention of **GraphReference** in the template source listing because **GraphReference** is inherited from a parent style. The following step displays the HTMLBLUE style and its parent styles:

```
proc template;
   source Styles.HTMLBlue / expand;
run;
```

The EXPAND option lists the styles in the following order: style of interest, then its parent, and then its grandparent, and so on. The HTMLBLUE style inherits attributes from the STATISTICAL style, which inherits attributes from the DEFAULT style. If you search the results from the top, you will find the most recent specification of a style element first. The **GraphReference** style element is defined as follows:

```
class GraphReference /
   linethickness = 1px
   linestyle = 1
   contrastcolor = GraphColors('greferencelines');
```

To specify a line thickness of 4 pixels for all reference lines, add the following statement to the definition of the NEWSTYLE style in the section "Modifying Graph Fonts in Styles" on page 768:

```
class GraphReference / linethickness=4px;
```

The following statements modify the style and produce the Q-Q plot shown in Figure 24.70:

```
proc template;
   define style Styles.NewStyle;
      parent=Styles.Statistical;
      class GraphFonts  /
         'GraphAnnoFont'     = ("<MTserif>, Times New Roman",10pt)
         'GraphDataFont'     = ("<MTserif>, Times New Roman", 7pt)
         'GraphUnicodeFont'  = ("<MTserif>, Times New Roman", 9pt)
         'GraphValueFont'    = ("<MTserif>, Times New Roman", 9pt)
         'GraphLabel2Font'   = ("<MTserif>, Times New Roman",10pt)
         'GraphLabelFont'    = ("<MTserif>, Times New Roman",10pt)
         'GraphFootnoteFont' = ("<MTserif>, Times New Roman",10pt)
         'GraphTitleFont'    = ("<MTserif>, Times New Roman",11pt)
         'GraphTitle1Font'   = ("<MTserif>, Times New Roman",14pt)
         'NodeTitleFont'     = ("<MTserif>, Times New Roman", 9pt)
         'NodeLabelFont'     = ("<MTserif>, Times New Roman", 9pt)
         'NodeLinkLabelFont' = ("<MTserif>, Times New Roman", 9pt)
         'NodeInputLabelFont'= ("<MTserif>, Times New Roman", 9pt)
         'NodeDetailFont'    = ("<MTserif>, Times New Roman", 7pt);
      class GraphReference / linethickness=4px;
   end;
run;

ods html style=NewStyle;
ods graphics on;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
ods html close;
```

Figure 24.70 Q-Q Plot That Uses the NEWSTYLE Style with a Thicker Line

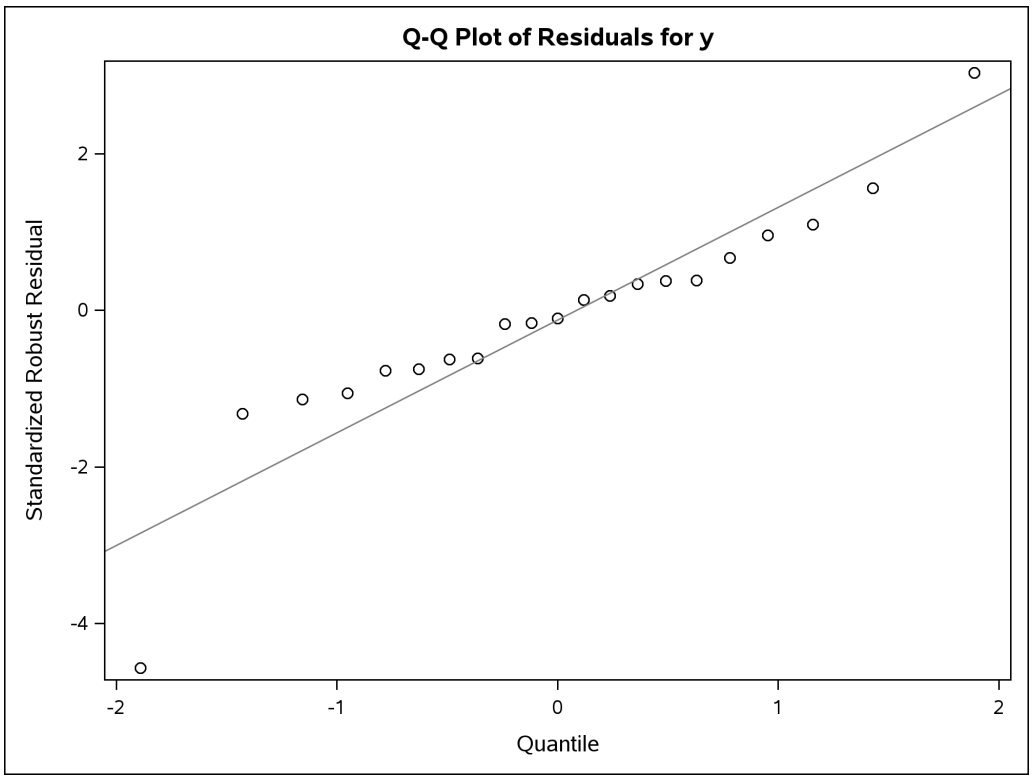

You can use this approach to modify other attributes of the line, such as **`LineStyle`** and **`ContrastColor`**. These style modifications apply to all graphs that display reference lines, not just Q-Q plots that are produced by PROC ROBUSTREG. You can control the attributes of specific graphs by modifying the graph template, as discussed in the section "Graph Templates" on page 844 in Chapter 25, "ODS Graphics Template Modification." Values that are specified directly in a graph template override style attributes.

When you are done with the NEWSTYLE style, you do not need to restore the HTMLBLUE style template, because you did not modify it. Rather, you inherited its attributes from the HTMLBLUE style.

Changing the Default Style

The default style for each ODS destination is specified in the SAS Registry. For example, the default style for the HTML destination is HTMLBLUE, and the default style for the RTF destination is RTF. You can specify a default style for all your output in a particular ODS destination. This is useful if you want to use a different ODS style, if you have modified one of the styles that the SAS System supplies (see the section "Style Templates and Colors" on page 727), or if you have defined your own style. For example, you can specify the JOURNAL style as the default style for RTF output.

The recommended approach for specifying a default style is as follows. Open the SAS Registry Editor by typing **regedit** on the command line. Expand the node **ODS ► DESTINATIONS** and select a destination (for example, select **RTF**). Double-click the **Selected Style** item, shown in Figure 24.71, and specify a style. This can be any style that the SAS System supplies or a user-defined style, as long as you can find it in the current template search path (for example, specify **Journal**). You can specify a default style for the other destinations in a similar way.

In a few cases, the default style is specified in more than one place. Assume you are using the SAS windowing environment and Microsoft Windows or UNIX in the following:

- If you expand the node **ODS ► DESTINATIONS ► HTML** (which refers to the obsolete HTML3 destination), you see that the **Selected Style** is DEFAULT.
- If you expand the node **ODS ► MARKUP ► HTML4**, you see that the **Selected Style** is HTMLBLUE.
- If you expand the node **ODS ► DMS ► DESTINATIONS ► MARKUP ► HTML4**, you see that the **Selected Style** is HTMLBLUE.

If you want to change the default style for the HTML destination, you need to change both HTML4 entries in the registry.

Figure 24.71 SAS Registry Editor

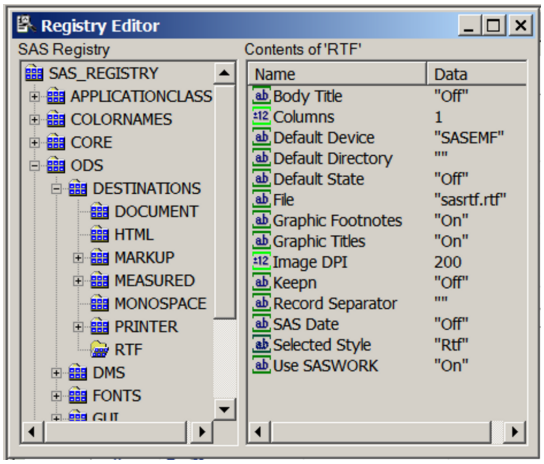

ODS searches sequentially through each element of the template search path for the first style template that matches the name of the style specified in the SAS Registry. It uses the first style template that it finds. (For more information about the template search path, see the sections "Saving Customized Templates" on page 853, "Using Customized Templates" on page 854, and "Reverting to the Default Templates" on page 854 in Chapter 25, "ODS Graphics Template Modification.") If you are specifying a customized style as your default style, the following are useful suggestions:

- If you save your style in Sasuser.Templat, verify that the name of your default style matches the name of the style specified in the SAS Registry. For example, suppose the RTF style is specified for the RTF destination in the SAS Registry. You can name your style RTF and save it in Sasuser.Templat. This blocks the RTF style in Sashelp.Tmplmst (provided that you did not alter the default template search path).
- If you save your style in a user-defined template store, verify that this template store is the first in the current template search path. Include the ODS PATH statement in your SAS autoexec file so that it is executed at start-up.

For the HTML destination, an alternative approach for specifying a default style is as follows. From the main SAS window, select **Tools ► Options ► Preferences**. On the **Results** tab, select the **Create HTML** check box and select a style from the **Style** list.

Statistical Graphics Procedures

Three Base SAS statistical graphics procedures use ODS Graphics and provide a convenient syntax for creating a variety of graphs from raw data or from procedure output:

SGSCATTER creates single-cell and multicell scatter plots and scatter plot matrices along with optional fits and ellipses.

SGPLOT creates single-cell plots along with a variety of plot and chart types.

SGPANEL creates single-page or multipage panels of plots and charts conditional on classification variables.

You do not need to enable ODS Graphics in order to use these SG (statistical graphics) procedures. In addition, the Base SAS SGRENDER procedure provides a way to create plots from graph templates that you modify or write yourself. For more information about the SG procedures and PROC SGRENDER, see the *SAS ODS Graphics: Procedures Guide* and Kuhfeld (2016).

These procedures do much more than make scatter plots. They can produce density plots, dot plots, needle plots, series plots, horizontal and vertical bar charts, histograms, and box plots. They can also compute and display loess fits, polynomial fits, penalized B-spline fits, reference lines, bands, and ellipses. PROC SGRENDER is the most flexible SG procedure because it uses the Graph Template Language (GTL). The syntax for the other SG procedures is much simpler than that of the GTL, so these procedures are recommended for creating most plots that are commonly required in statistical work.

The SGPLOT Procedure

PROC SGPLOT provides a simple way to make a variety of scatter plots. This example is taken from Example 78.4 in Chapter 78, "The LOESS Procedure." The ENSO data set, which contains information about differences in ocean pressure over time, is available from the Sashelp library.

The following statements create a scatter plot of points along with a penalized B-spline fit to the data and produce Figure 24.72:

```
proc sgplot data=sashelp.enso noautolegend;
   title 'Atmospheric Pressure Differences between '
         'Easter Island and Darwin, Australia';
   pbspline y=pressure x=year;
run;
```

Figure 24.72 Penalized B-Spline Fit with PROC SGPLOT

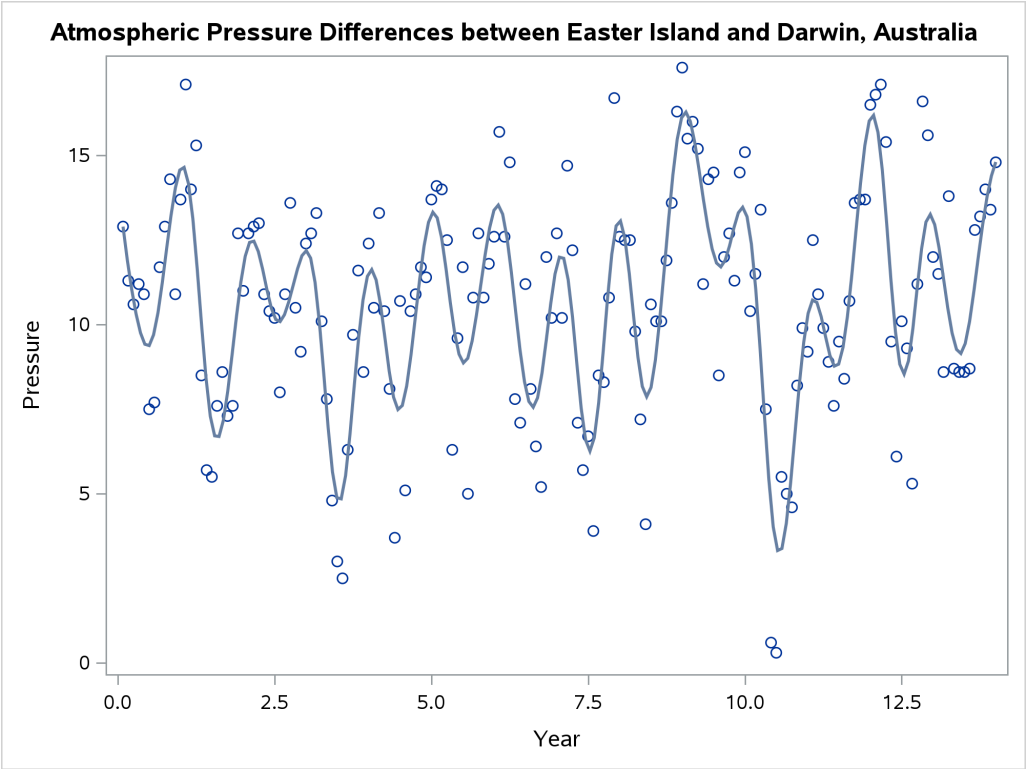

For more information about penalized B-splines, see Chapter 126, "The TRANSREG Procedure." Also see the section "Grouped Scatter Plot with PROC SGPLOT" on page 684 and Figure 24.12 for an example of a scatter plot that has groups of observations.

The SGSCATTER Procedure

You can use the SGSCATTER procedure to produce scatter plot matrices. The following step creates a scatter plot matrix from all the numeric variables in the Class data set (available in the Sashelp library) and produces Figure 24.73:

```
proc sgscatter data=sashelp.class;
   matrix _numeric_ / diagonal=(kernel histogram);
run;
```

The diagonal cells of Figure 24.73 contain a histogram and a kernel density fit. The off-diagonal cells contain all pairs of scatter plots.

Figure 24.73 Scatter Plot Matrix with PROC SGSCATTER

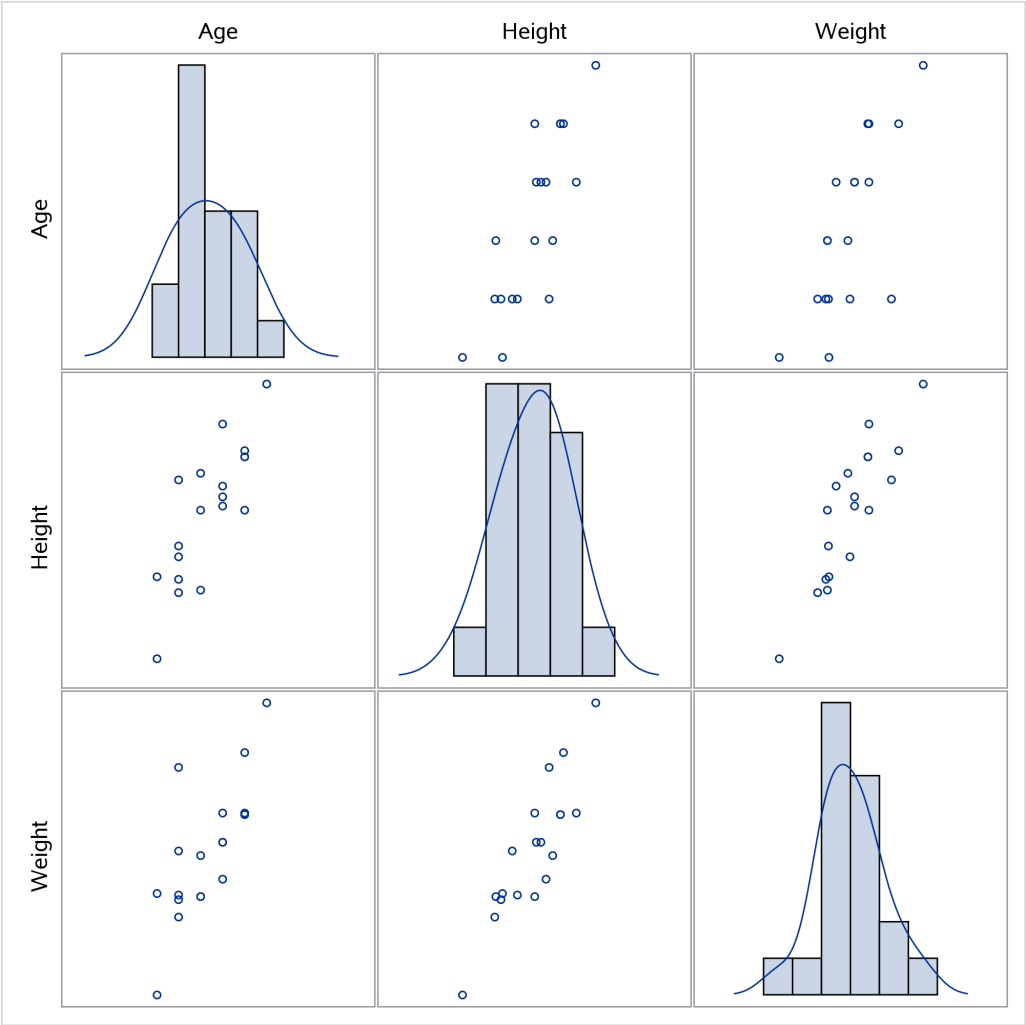

The MATRIX statement creates a symmetric $n \times n$ scatter plot matrix. Other statements are also available. The PLOT statement creates a panel that contains one or more individual scatter plots. The COMPARE statement creates a rectangular $m \times n$ scatter plot matrix. Linear and nonlinear fits can be added, and you can request many graphical features by specifying options.

The SGPANEL Procedure

The SGPANEL procedure creates paneled plots and charts that have one or more classification variables. Classification variables can be designated as row or column variables, or there can be multiple classifications. Graphs are drawn for each combination of the levels of classification variables, showing a subset of the data in each cell.

This example is taken from Example 52.6 in Chapter 52, "The GLIMMIX Procedure." The following statements create the input SAS data sets:

```
data times;
   input time1-time23;
   datalines;
122  150  166  179  219  247  276  296  324  354  380  445
478  508  536  569  599  627  655  668  723  751  781
;

data cows;
   if _n_ = 1 then merge times;
   array t{23} time1 - time23;
   array w{23} weight1 - weight23;
   input cow iron infection weight1-weight23 @@;
   do i=1 to 23;
      weight = w{i};
      tpoint = (t{i}-t{1})/10;
      output;
   end;
   keep cow iron infection tpoint weight;
   datalines;
 1 0 0  4.7    4.905  5.011  5.075  5.136  5.165  5.298  5.323
        5.416  5.438  5.541  5.652  5.687  5.737  5.814  5.799
        5.784  5.844  5.886  5.914  5.979  5.927  5.94
 2 0 0  4.868  5.075  5.193  5.22   5.298  5.416  5.481  5.521

   ... more lines ...

;
```

First, PROC GLIMMIX is run to fit the model, and then the results are prepared for plotting:

```
proc glimmix data=cows;
   t2 = tpoint / 100;
   class cow iron infection;
   model weight = iron infection iron*infection tpoint;
   random t2 / type=rsmooth subject=cow
                   knotmethod=kdtree(bucket=100 knotinfo);
   output out=gmxout pred(blup)=pred;
   nloptions tech=newrap;
run;

data plot;
   set gmxout;
   length Group $ 26;
```

```
   if      (iron=0) and (infection=0) then group='Control Group (n=4)';
   else if (iron=1) and (infection=0) then group='Iron - No Infection (n=3)';
   else if (iron=0) and (infection=1) then group='No Iron - Infection (n=9)';
   else group = 'Iron - Infection (n=10)';
run;

proc sort data=plot; by group cow;
run;
```

The following statements produce graphs of the observed data and fitted profiles in the four groups:

```
proc sgpanel data=plot noautolegend;
   title 'Radial Smoothing with Cow-Specific Trends';
   label tpoint='Time' weight='log(Weight)';
   panelby group / columns=2 rows=2;
   scatter x=tpoint y=weight;
   series  x=tpoint y=pred / group=cow lineattrs=GraphFit;
run;
```

The results are shown in Figure 24.74.

Figure 24.74 Fit Using PROC SGPANEL

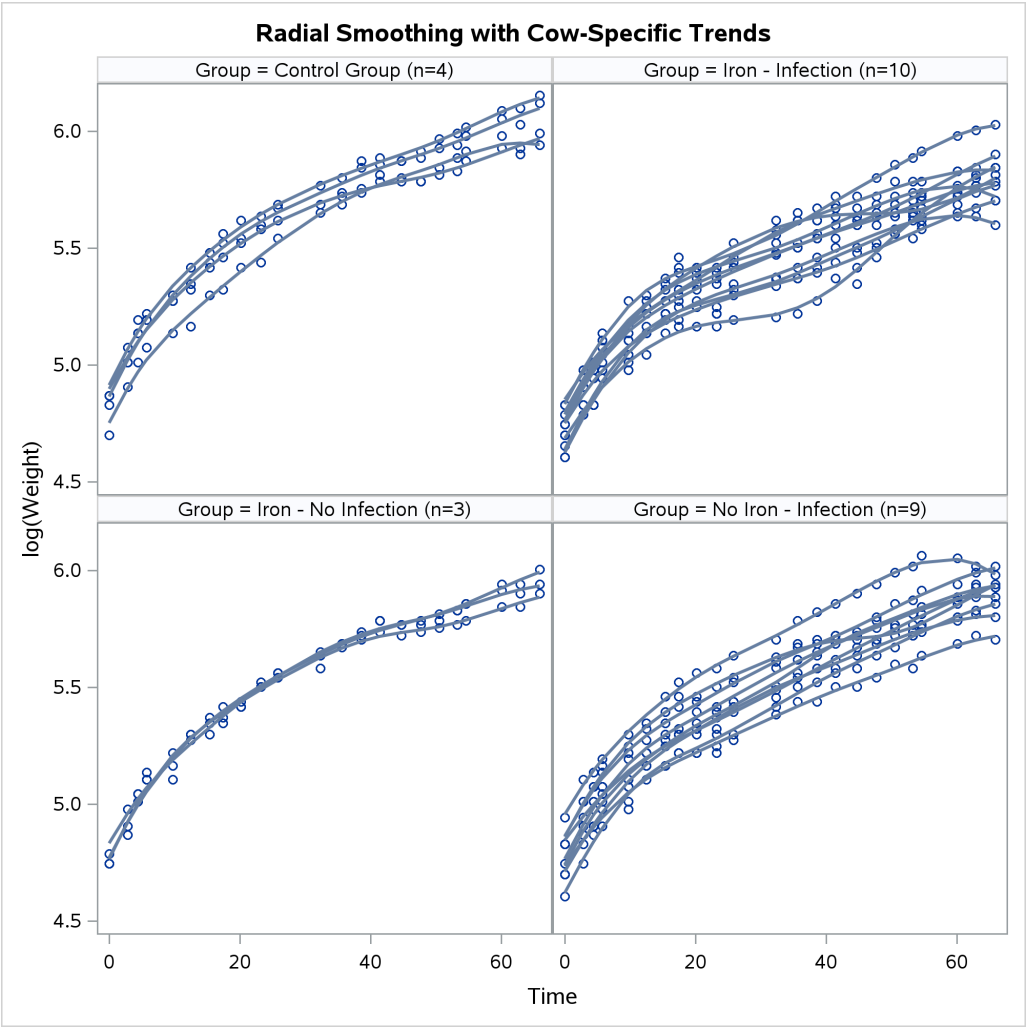

The SGRENDER Procedure

The SGRENDER procedure produces a graph from an input SAS data set and an ODS graph template. By using PROC SGRENDER and the Graph Template Language, you can create highly customized graphs. The following steps create a simple scatter plot of the Class data set (available in the Sashelp library) and produce Figure 24.75:

```
proc template;
   define statgraph Scatter;
      begingraph;
         entrytitle "Simple Scatter Plot of the Class Data Set";
         layout overlay;
            scatterplot y=weight x=height / datalabel=name;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=sashelp.class template=scatter;
run;
```

The template definition consists of an outer block that begins with a DEFINE statement and ends with an END statement. Inside that is a BEGINGRAPH/ENDGRAPH block. Inside that block, the ENTRYTITLE statement provides the plot title, and the LAYOUT OVERLAY block contains the statement or statements that define the graph. In this case, there is just a single SCATTERPLOT statement that names the Y-axis (vertical) variable, the X-axis (horizontal) variable, and an optional variable that contains labels for the points. The PROC SGRENDER statement simply specifies the input data set and the template. The real work in using PROC SGRENDER is writing the template.

Figure 24.75 Scatter Plot of Labeled Points with PROC SGRENDER

The following steps add a series of fit functions to the scatter plot and create a legend by adding statements to the **Scatter** template:

```
proc template;
   define statgraph Scatter;
      begingraph;
         entrytitle "Scatter Plot of the Class Data Set with Fit Functions";
         layout overlay;
            scatterplot y=weight x=height / datalabel=name;
            pbsplineplot y=weight x=height / name='pbs'
                         legendlabel='Penalized B-Spline'
                         lineattrs=GraphData1;
            regressionplot y=weight x=height / degree=1 name='line'
                         legendlabel='Linear Fit'
                         lineattrs=GraphData2;
            regressionplot y=weight x=height / degree=3 name='cubic'
                         legendlabel='Cubic Fit'
                         lineattrs=GraphData3;
            loessplot y=weight x=height / name='loess'
                         legendlabel='Loess Fit'
                         lineattrs=GraphData4;
            discretelegend 'pbs' 'line' 'cubic' 'loess';
         endlayout;
      endgraph;
   end;
run;
```

```
proc sgrender data=sashelp.class template=scatter;
run;
```

The line attributes for each function are specified in different style elements, **GraphData1** through **GraphData4**, so that the functions are adequately identified in the legend. The preceding statements create Figure 24.76.

Figure 24.76 Scatter Plot and Fit Functions with PROC SGRENDER

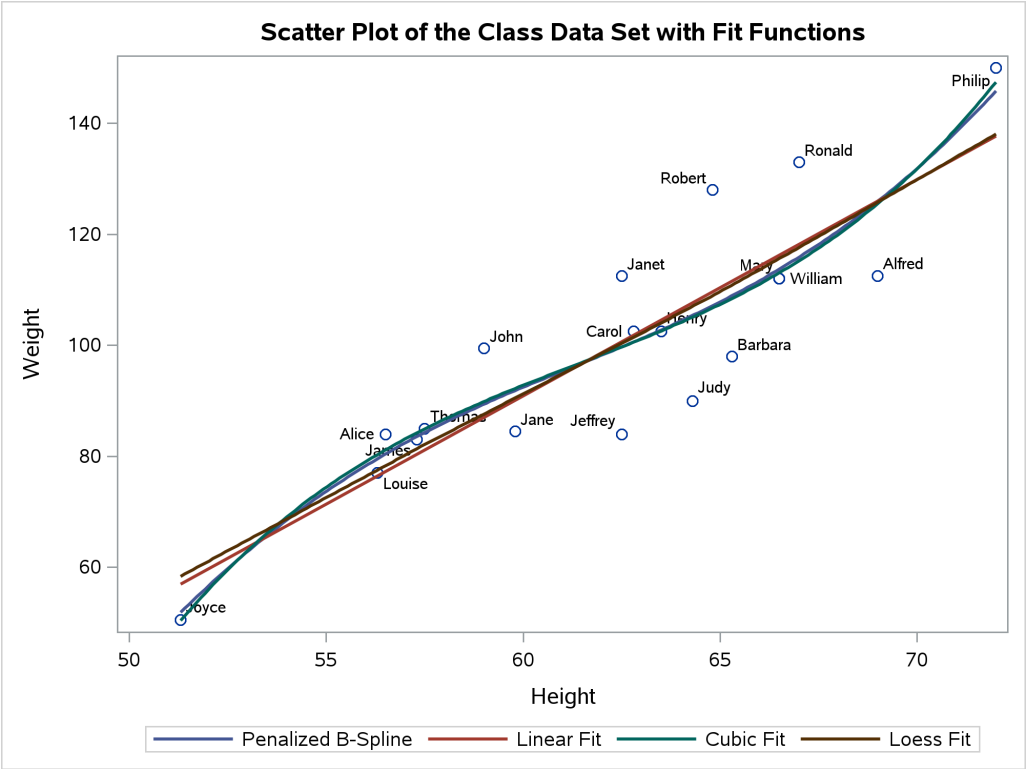

The following statements create a four-panel display of the Class data set and produce Figure 24.77:

```
proc template;
   define statgraph Panel;
      begingraph;
         entrytitle "Paneled Display of the Class Data Set";

         layout lattice / rows=2 columns=2 rowgutter=10 columngutter=10;

            layout overlay;
               scatterplot y=weight x=height;
               pbsplineplot y=weight x=height;
            endlayout;

            layout overlay / xaxisopts=(label='Weight');
               histogram weight;
            endlayout;

            layout overlay / yaxisopts=(label='Height');
```

```
            boxplot y=height;
         endlayout;

         layout overlay / xaxisopts=(offsetmin=0.1 offsetmax=0.1)
                          yaxisopts=(offsetmin=0.1 offsetmax=0.1);
            scatterplot  y=weight x=height / markercharacter=sex
               name='color' markercolorgradient=age;
            continuouslegend 'color'/ title='Age';
         endlayout;

      endlayout;
   endgraph;
end;
run;

proc sgrender data=sashelp.class template=panel;
run;
```

In this template, the outermost layout is a LAYOUT LATTICE block. It creates a 2 × 2 panel of plots that have a 10-pixel separation (or gutter) between pairs of plots. Inside the lattice are four LAYOUT OVERLAY blocks, each defining one of the graphs. The first is a simple scatter plot that contains a nonlinear penalized B-spline fit. The second is a histogram of the dependent variable Weight. The third is a box plot of the independent variable Height. The fourth simultaneously shows the height, weight, age, and sex of the students in the class. Each axis has an offset that is added at both the maximum and the minimum. This provides padding between the axes and the data.

Figure 24.77 Multiple Panels Using PROC SGRENDER

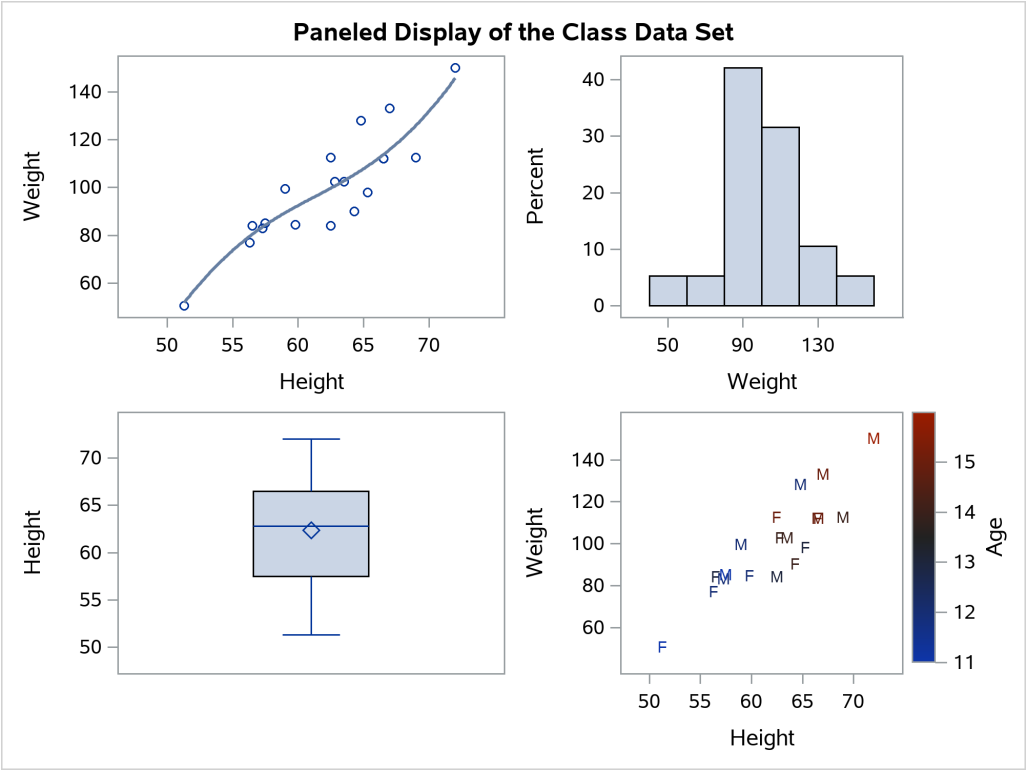

Many other types of graphs are available in the SG procedures. However, even the few examples that are provided here show the power and flexibility available for making professional-quality statistical graphics. For more information, see the *SAS Graph Template Language: User's Guide* and the *SAS ODS Graphics: Procedures Guide*.

Examples of ODS Statistical Graphics

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 24.1: Creating Graphs with Tooltips in HTML

(View the complete code for this example (odsgrex1.sas) in the example repository.)

This example demonstrates how to request graphs in HTML that are enhanced with tooltip displays, which appear when you move a mouse pointer over certain features of the graph. When you specify the HTML destination and IMAGEMAP=ON in the ODS GRAPHICS statement, an image map of coordinates for tooltips is generated along with the HTML output file. Individual graphs are saved as PNG files.

Example 84.2 and Example 84.8 in Chapter 84, "The MIXED Procedure," analyze a data set that has repeated growth measurements for 27 children. The following step creates the data set:

```
data pr;
   input Person Gender $ y1 y2 y3 y4 @@;
   y=y1; Age=8;  output;
   y=y2; Age=10; output;
   y=y3; Age=12; output;
   y=y4; Age=14; output;
   drop y1-y4;
   datalines;
 1  F  21.0  20.0  21.5  23.0      2  F  21.0  21.5  24.0  25.5
 3  F  20.5  24.0  24.5  26.0      4  F  23.5  24.5  25.0  26.5

   ... more lines ...

;
```

The following statements fit a mixed model that has random intercepts and slopes for each child:

```
ods _all_ close;
ods html body='b.html' style=HTMLBlue;
ods graphics on / imagemap=on;

proc mixed data=pr method=ml plots=boxplot;
   ods select 'Conditional Residuals by Gender';
   class Person Gender;
   model y = Gender Age Gender*Age;
   random intercept Age / type=un subject=Person;
run;
```

```
ods html close;
```

The PLOTS=BOXPLOT option in the PROC MIXED statement requests box plots of observed values and residuals for each classification main effect in the model (Gender and Person). Only the by-gender box plots are actually created because of the ODS SELECT statement, which uses the plot label to select the plot. Output 24.1.1 displays the results. Moving the mouse pointer over a box plot displays a tooltip that contains summary statistics for the class level. Graphics with tooltips are supported for only the HTML destination.

Output 24.1.1 Box Plot with Tooltips

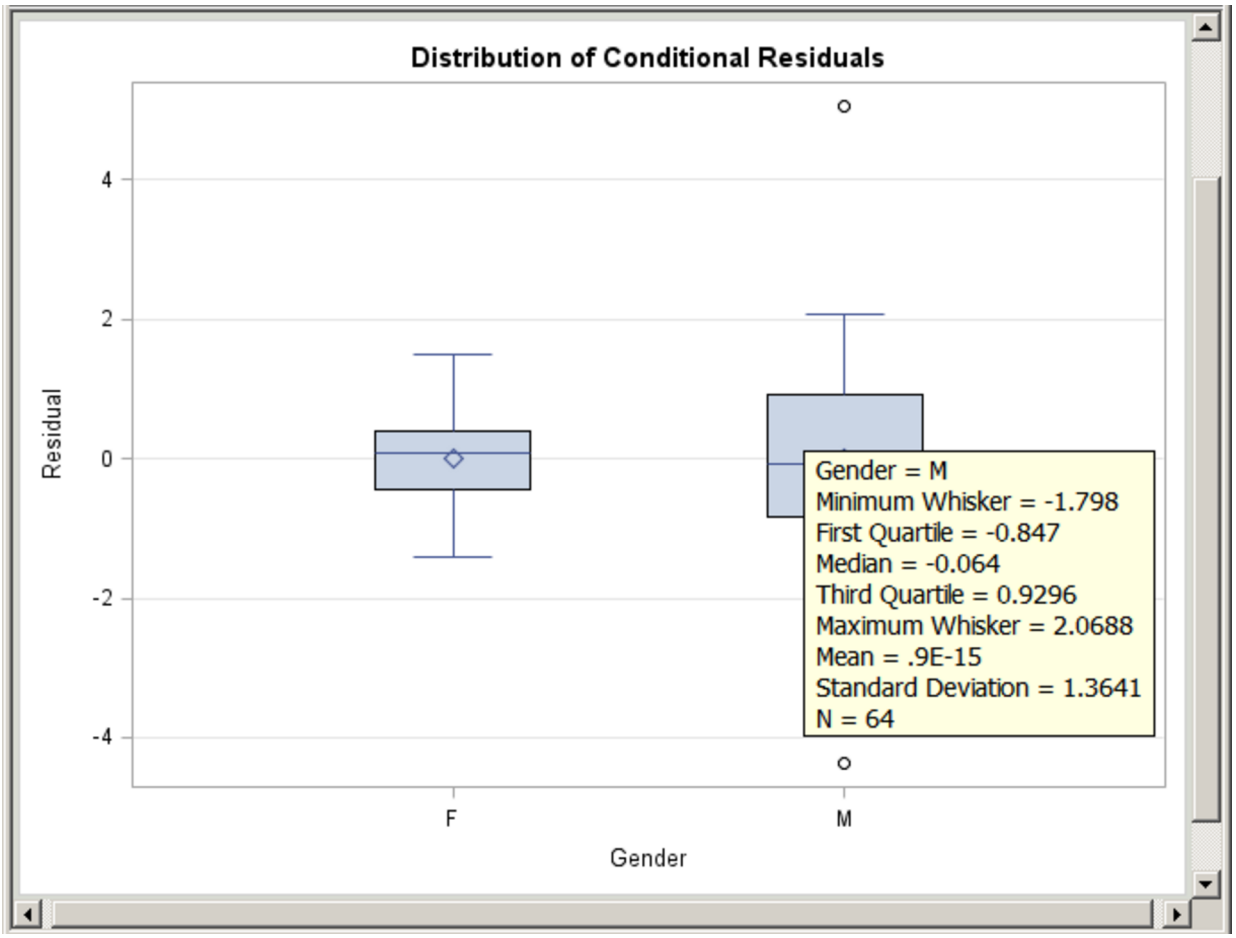

Example 24.2: Creating Graphs for a Presentation

(View the complete code for this example (odsgrex2.sas) in the example repository.)

The RTF destination provides an easy way to create graphs for inclusion in a paper or presentation. You can specify the ODS RTF statement to create a file that is easily imported into a document (such as Microsoft Word or WordPerfect) or a presentation (such as Microsoft PowerPoint).

The following statements request a loess fit and save the output in the file *loess.rtf*:

```
ods _all_ close;
ods rtf file="loess.rtf" style=HTMLBlue;
ods graphics on;

proc loess data=sashelp.enso;
   model pressure = year / clm residual;
run;

ods rtf close;
ods listing;
```

The output file includes various tables and the following plots: a plot of the selection criterion versus smoothing parameter, a fit plot with 95% confidence bands, a plot of residual by regressors, and a diagnostics panel. The fit plot is produced by using the HTMLBLUE style and is shown in Output 24.2.1.

Output 24.2.1 Loess Fit Plot with the HTMLBLUE Style

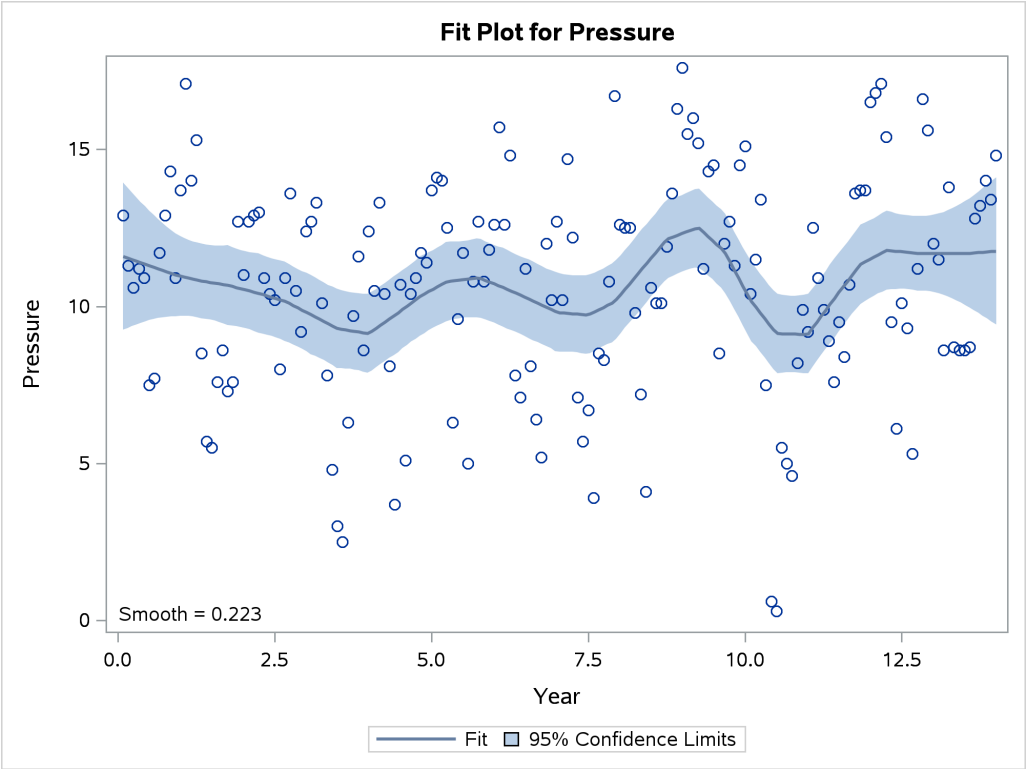

If you are running SAS on the Microsoft Windows operating system, you can open the RTF file in Microsoft

Word and simply copy and paste the graphs into Microsoft PowerPoint. In general, RTF output is convenient for exchange of graphical results between Windows applications through the clipboard.

Alternatively, if you use the LISTING or HTML destination, then your individual graphs are created as PNG files by default. You can insert these files in a Microsoft PowerPoint presentation. For information about how the image files are named and saved, see the sections "Naming Graphic Image Files" on page 709 and "Saving Graphic Image Files" on page 711.

Example 24.3: Creating Graphs in PostScript Files

(View the complete code for this example (odsgrex3.sas) in the example repository.)

This example illustrates how to create individual graphs in PostScript files. This is particularly useful when you want to include them in a LaTeX document.

The following statements close all open destinations, open the LATEX[4] destination with the JOURNAL style, and request a grouped bar chart for the Sashelp.Class data set:

```
ods graphics on / reset=index;
ods _all_ close;
ods latex style=Journal;

proc sgplot data=sashelp.class;
   vbar age / group=sex;
run;

ods latex close;
ods listing;
```

The JOURNAL style displays grayscale graphs that are suitable for a journal. When you specify the ODS LATEX destination, ODS creates a PostScript file for each individual graph in addition to a LaTeX source file that includes the tabular output and references to the PostScript files. By default, these files are saved in the SAS current folder. The bar chart shown in Output 24.3.1 is saved by default in a file named *SGPlot.ps*. For information about how graphic image files are named, see the section "Naming Graphic Image Files" on page 709. If both the default destination (LISTING or HTML) and the LATEX destination are open, then two files are created: *SGPlot.png* and *SGPlot1.ps*. If RESET=INDEX is not specified in the ODS GRAPHICS statement and you run the step again, the next names are based on an incremented index (*SGPlot2.png* and *SGPlot3.ps*).

[4]The LATEX destination is experimental.

Output 24.3.1 Bar Chart Using the JOURNAL Style

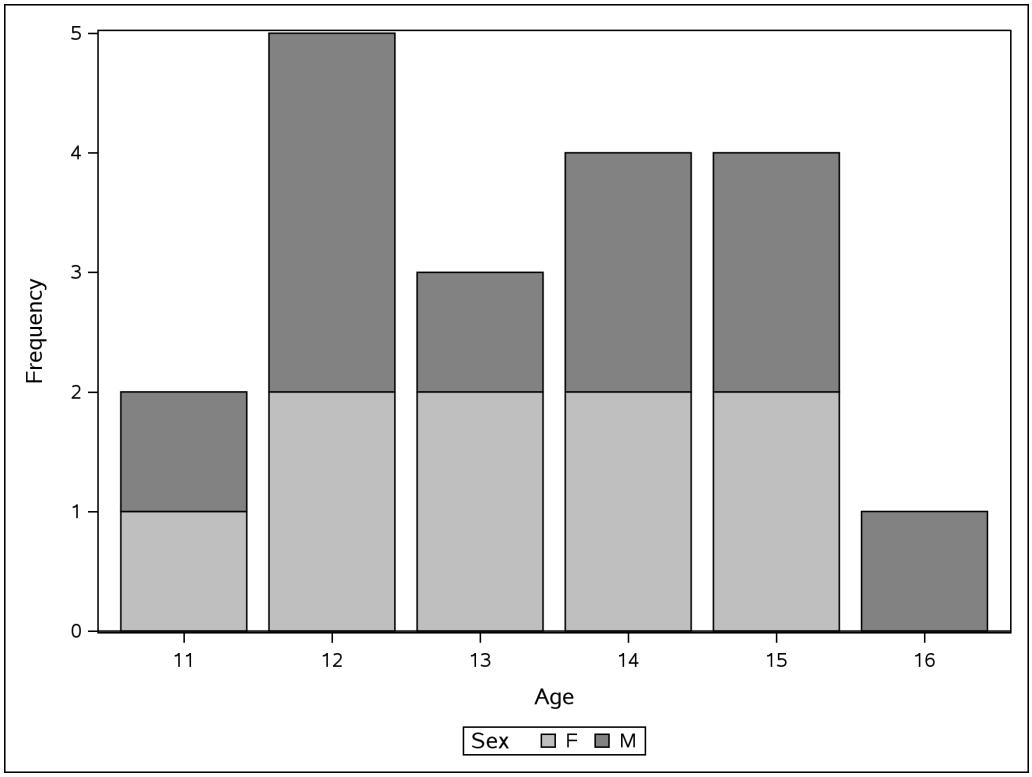

You can use the JOURNAL2 style for a different appearance—the bars are not shaded. Crosshatching is used to indicate group membership. The following step produces Output 24.3.2:

```
ods graphics on / reset=index;
ods _all_ close;
ods latex style=Journal2;

proc sgplot data=sashelp.class;
   vbar age / group=sex;
run;

ods latex close;
ods listing;
```

Output 24.3.2 Bar Chart Using the JOURNAL2 Style

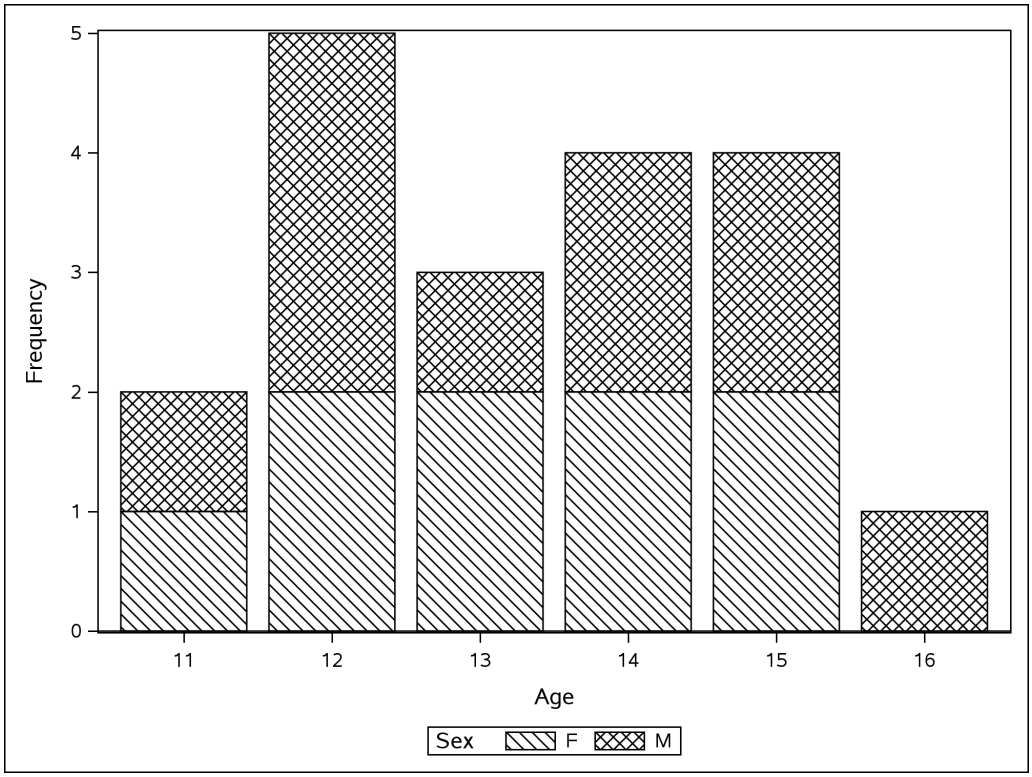

You can use the JOURNAL3 style for a different kind of appearance from the JOURNAL style. A mix of filled areas and crosshatching is used in grouped bar charts. The following step produces Output 24.3.3:

```
ods graphics on / reset=index;
ods _all_ close;
ods latex style=Journal3;

proc sgplot data=sashelp.class;
   vbar age / group=sex;
run;

ods latex close;
ods listing;
```

Output 24.3.3 Bar Chart Using the JOURNAL3 Style

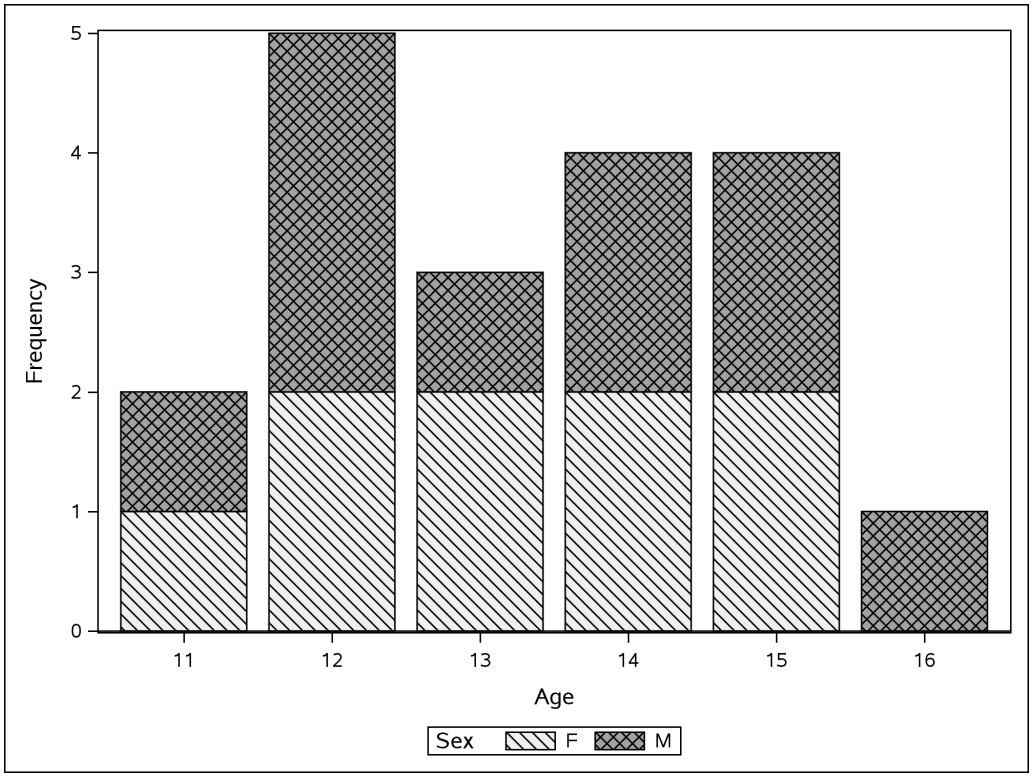

If you are writing a paper, you can include the graphs in your own LaTeX source file by referencing the names of the individual PostScript graphics files. In this situation, you might not find it necessary to use the LaTeX source file created by the SAS System. Alternatively, you can include PNG files in a LaTeX document, after using some other ODS destination (such as HTML) to create the PNG files.

Example 24.4: Displaying Graphs Using the DOCUMENT Procedure

(View the complete code for this example (odsgrex4.sas) in the example repository.)

This example illustrates the use of the ODS DOCUMENT destination and the DOCUMENT procedure to display your ODS graphs. You can use this approach whenever you want to generate and save your output (both tables and graphs) and then display it later, potentially in subsets or more than once. This approach is particularly useful when you want to display your output in multiple ODS destinations or when you want to use different styles without rerunning your SAS program. This approach is also useful when you want to break your output in separate parts for inclusion in different parts of a document such as a LaTeX file.

Consider again the data set Stack created by the following statements:

```
data stack;
   input x1 x2 x3 y @@;
   datalines;
80 27 89 42   80 27 88 37   75 25 90 37   62 24 87 28   62 22 87 18
62 23 87 18   62 24 93 19   62 24 93 20   58 23 87 15   58 18 80 14
58 18 89 14   58 17 88 13   58 18 82 11   58 19 93 12   50 18 89  8
```

```
50 18 86  7   50 19 72  8   50 19 79  8   50 20 80  9   56 20 82 15
70 20 91 15
;
```

The following statements request a Q-Q plot from PROC ROBUSTREG to analyze the Stack data:

```
ods graphics on;
ods _all_ close;
ods document name=QQDoc(write);

proc robustreg data=stack plots=qqplot;
   model y = x1 x2 x3;
quit;

ods document close;
ods listing;
```

The ODS DOCUMENT statement opens an ODS document named **QQDoc**. All the results—tables, graphs, titles, notes, footnotes, headings—are stored in the ODS document. None of them are displayed because no other destination is open. To display the Q-Q plot by using PROC DOCUMENT, you first need to determine its name. You can do this by specifying the ODS TRACE ON statement before the procedure statements (for more information, see the section "Determining Graph Names and Labels" on page 703). Alternatively, you can enter **odsdocuments** (or **odsd** for short) on the command line to open the Documents window, which you can then use to manage your ODS documents.

The following statements specify an HTML destination and display the residual Q-Q plot by using the REPLAY statement in PROC DOCUMENT:

```
ods html body='b.htm';

proc document name=QQDoc;
   ods select QQPlot;
   replay;
quit;

ods html close;
```

Subsequent steps can replay one or more objects from the same ODS document. By default, the REPLAY statement attempts to display every output object stored in the ODS document, but here only the Q-Q plot is displayed because it is specified by the ODS SELECT statement. The plot is displayed in Output 24.4.1.

Output 24.4.1 Q-Q Plot Displayed by PROC DOCUMENT

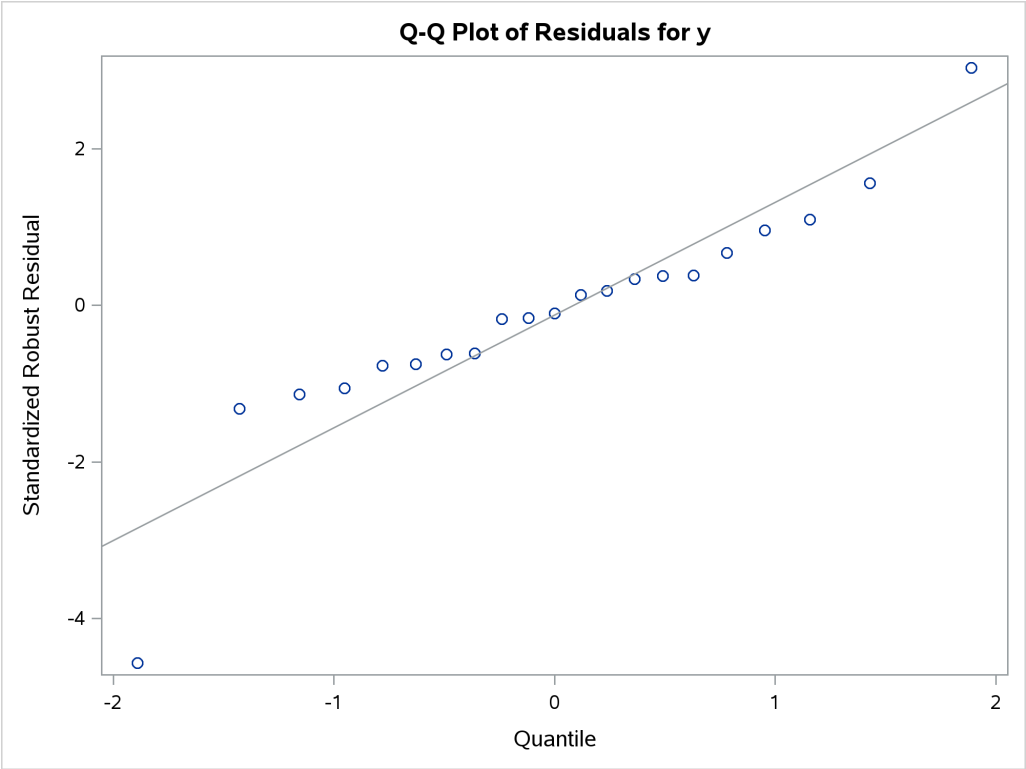

As an alternative to running PROC DOCUMENT along with an ODS SELECT statement, you can run PROC DOCUMENT and use a *document path* for the Q-Q plot in the REPLAY statement. This approach is preferable when the ODS document contains a large volume of output, so that PROC DOCUMENT does not attempt to process every piece of output that is stored in the ODS document.

You can determine the ODS document path for the Q-Q plot by specifying the LIST statement and LEVELS=ALL in PROC DOCUMENT as follows:

```
proc document name=QQDoc;
   list / levels=all;
quit;
```

The contents of the ODS document **QQDoc** are shown in Output 24.4.2.

Output 24.4.2 Contents of the ODS Document `QQDoc`

Listing of: \Work.Qqdoc\
Order by: Insertion
Number of levels: All

Obs	Path	Type
1	\Robustreg#1	Dir
2	\Robustreg#1\ModelInfo#1	Table
3	\Robustreg#1\NObs#1	Table
4	\Robustreg#1\ParmInfo#1	Table
5	\Robustreg#1\SummaryStatistics#1	Table
6	\Robustreg#1\ParameterEstimates#1	Table
7	\Robustreg#1\DiagSummary#1	Table
8	\Robustreg#1\DiagnosticPlots#1	Dir
9	\Robustreg#1\DiagnosticPlots#1\QQPlot#1	Graph
10	\Robustreg#1\GoodFit#1	Table

The ODS document path of the `QQPlot` entry in the `QQDoc` ODS document, as shown in Output 24.4.2, is `\Robustreg#1\DiagnosticPlots#1\QQPlot#1`.

You can use this path to display the residual Q-Q plot by using PROC DOCUMENT as follows:

```
proc document name=QQDoc;
   replay \Robustreg#1\DiagnosticPlots#1\QQPlot#1;
quit;
```

You can also determine the ODS document path from the Results window or Documents window. Right-click the object icon and select **Properties**.

The SAS/STAT documentation preparation process uses the ODS document. SAS output is saved in an ODS document that is then used in sections of the documentation, which is prepared using LaTeX. In general, when you send your output to the DOCUMENT destination, you can use PROC DOCUMENT to rearrange, duplicate, or remove output from the results of a procedure or a database query. For more information, see the ODS DOCUMENT statement in the section "Dictionary of ODS Language Statements" and the chapter "The DOCUMENT Procedure" in the *SAS Output Delivery System: User's Guide*.

Example 24.5: Customizing the Style for Box Plots

(View the complete code for this example (odsgrex5.sas) in the example repository.)

This example demonstrates how to modify the style for box plots. This example is taken from Example 24.1. The following step creates the data set:

```
data pr;
   input Person Gender $ y1 y2 y3 y4 @@;
   y=y1; Age=8;  output;
   y=y2; Age=10; output;
   y=y3; Age=12; output;
   y=y4; Age=14; output;
   drop y1-y4;
   datalines;
```

```
 1  F  21.0  20.0  21.5  23.0      2  F  21.0  21.5  24.0  25.5
 3  F  20.5  24.0  24.5  26.0      4  F  23.5  24.5  25.0  26.5

   ... more lines ...

;
```

The following step displays the HTMLBLUE style and its parent styles, STATISTICAL and DEFAULT:

```
proc template;
   source Styles.HTMLBlue / expand;
run;
```

If you search for “box”, you find the style element that controls some aspects of the box plot:

```
class GraphBox /
   capstyle = "serif"
   connect = "mean"
   displayopts = "fill caps median mean outliers";
```

You can learn more about the **GraphBox** style element and its attributes in the section on the BOXPLOT statement in the *SAS Graph Template Language: Reference* and in the section “ODS Style Elements” in the *SAS Output Delivery System: User’s Guide*.

The following statements create two new styles by modifying attributes of the **GraphBox** style element. The first style is a sparse style; the box is outlined (not filled), and the median is shown, but not the mean. In contrast, the second style produces a filled box, with caps on the whiskers that shows the mean, median, and outliers. In addition, the box is notched.

The following statements create the two styles:

```
proc template;
   define style BoxStyleSparse;
      parent=styles.HTMLBlue;
      style GraphBox / capstyle = "line" displayopts = "median";
   end;
   define style BoxStyleRich;
      parent=styles.HTMLBlue;
      style GraphBox / capstyle = "bracket"
            displayopts = "fill caps median mean outliers notches";
   end;
run;
```

The following steps run PROC MIXED and create box plots that use the two styles:

```
ods graphics on;
ods html style=boxstylesparse;

proc mixed data=pr method=ml plots=boxplot;
   ods select 'Conditional Residuals by Gender';
   class Person Gender;
   model y = Gender Age Gender*Age;
   random intercept Age / type=un subject=Person;
run;
ods html close;
```

```
ods html style=boxstylerich;

proc mixed data=pr method=ml plots=boxplot;
   ods select 'Conditional Residuals by Gender';
   class Person Gender;
   model y = Gender Age Gender*Age;
   random intercept Age / type=un subject=Person;
run;
ods html close;
```

The results from using the sparse style are displayed in Output 24.5.1, and the results from using the richer style are displayed in Output 24.5.2. See Output 24.1.1 in Example 24.1 for the results of using the HTMLBLUE style.

Output 24.5.1 Box Plot with the Sparse Style

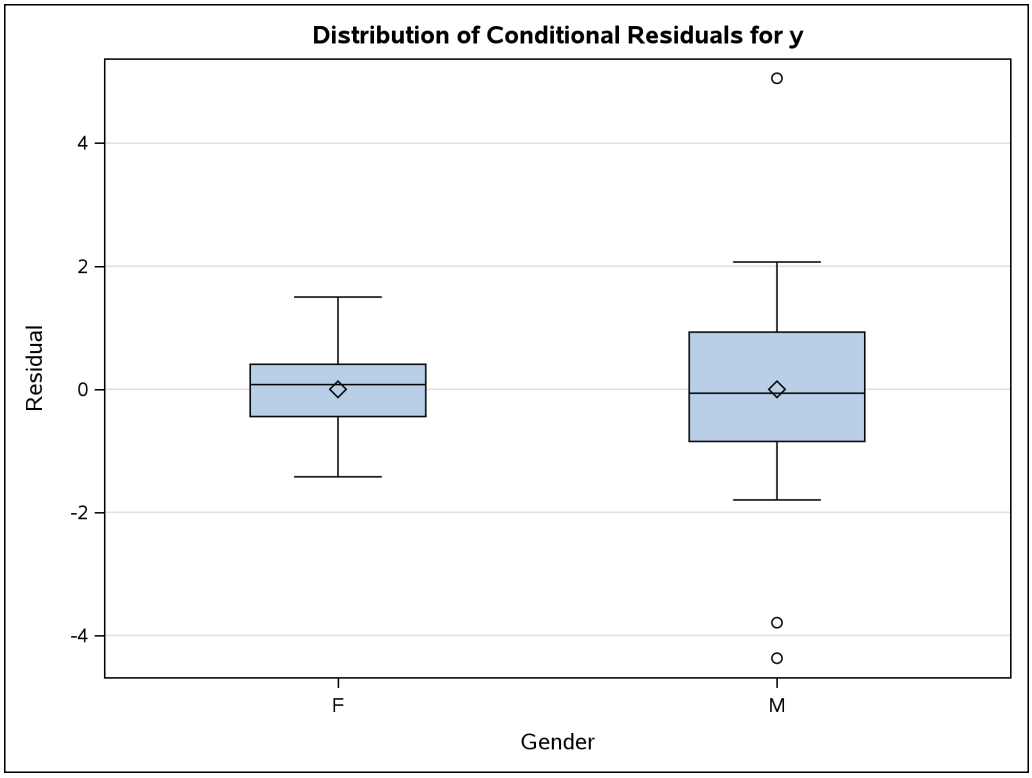

Output 24.5.2 Box Plot with the Richer Style

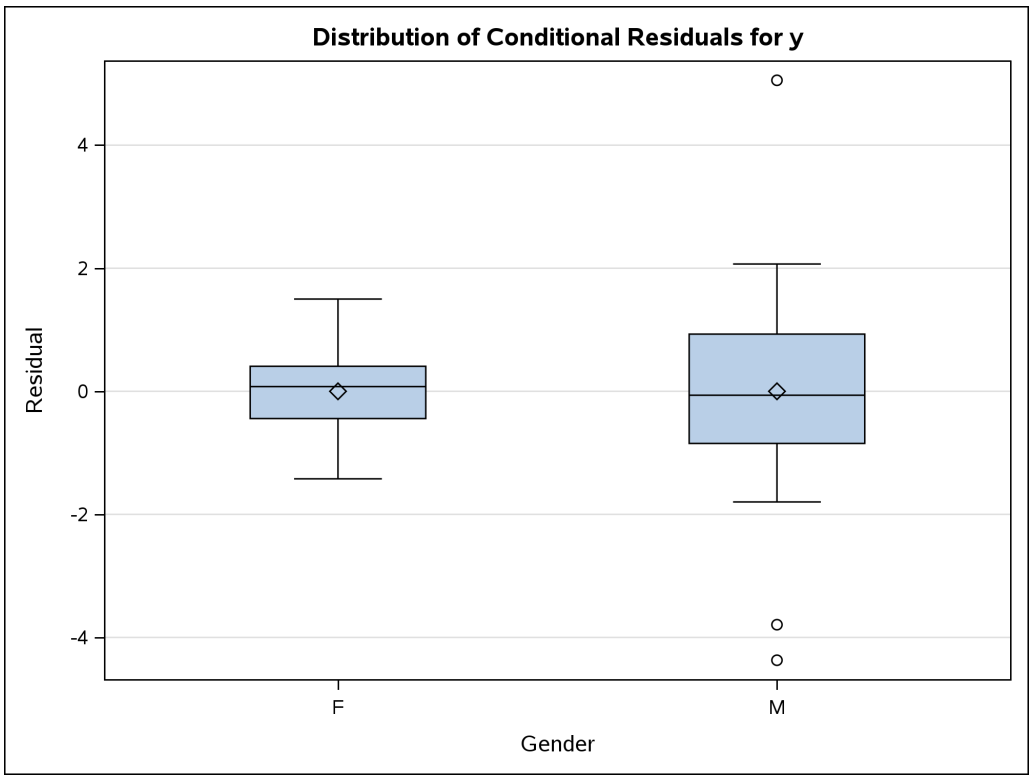

Example 24.6: Splines

(View the complete code for this example (odsgrex6.sas) in the example repository.)

This example illustrates several types of spline fit functions that are available in ODS Graphics and SAS/STAT. It begins by showing you how to use PROC SGPLOT to fit a single function through a scatter plot. Later examples use PROC SGPLOT to fit multiple functions by specifying a group variable. Subsequent sections show how to fit spline functions in SAS/STAT procedures, concentrating on PROC TRANSREG and the spline functionality in the EFFECT and EFFECTPLOT statements. PROC ADAPTIVEREG is also discussed. Topics include polynomials, polynomial splines (Smith 1979), B-splines (De Boor 1978), penalized B-splines (Eilers and Marx 1996), monotone splines (Winsberg and Ramsay 1980), natural cubic splines and restricted splines (Hastie, Tibshirani, and Friedman 2009), smoothing splines (Reinsch 1967), multivariate adaptive regression splines (Friedman 1991), and loess (Cleveland, Devlin, and Grosse 1988).

Single Fit Function Using PROC SGPLOT

Polynomial Fit Function

You can use PROC SGPLOT and the REG statement along with the DEGREE=3 option to fit a cubic polynomial function to your data. A cubic polynomial is smooth and has little freedom to follow nonlinear trends in the data. The model is

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \epsilon$$

The following step creates the plot in Output 24.6.1:

```
proc sgplot data=sashelp.heart noautolegend;
   reg y=weight x=height / markerattrs=(size=3px) degree=3;
run;
```

Output 24.6.1 Cubic Polynomial Fit Plot

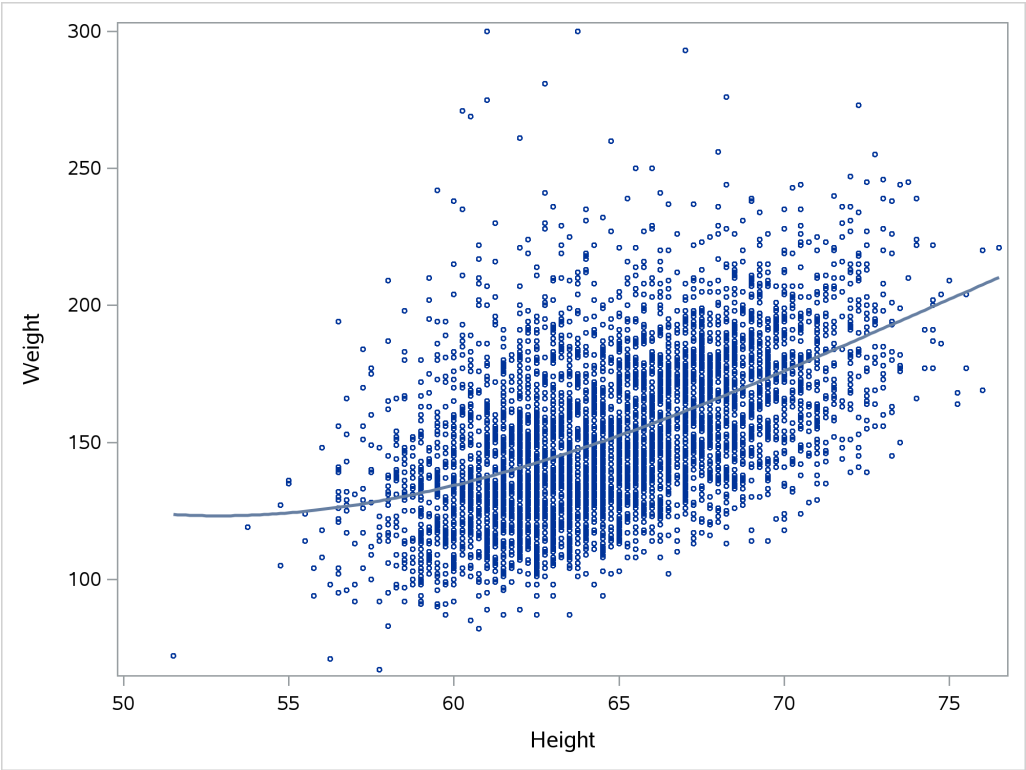

Penalized B-Spline Fit Function

You can use the PBSPLINE statement to fit penalized B-splines (Eilers and Marx 1996). Penalized B-splines draw a smooth curve through a scatter plot by using an automatic selection of the smoothing parameter. The following step creates the plot in Output 24.6.2:

```
ods graphics on / antialiasmax=6000;
proc sgplot data=sashelp.heart noautolegend;
   pbspline y=weight x=height / markerattrs=(size=3px);
run;
```

Output 24.6.2 Penalized B-Spline Fit Plot

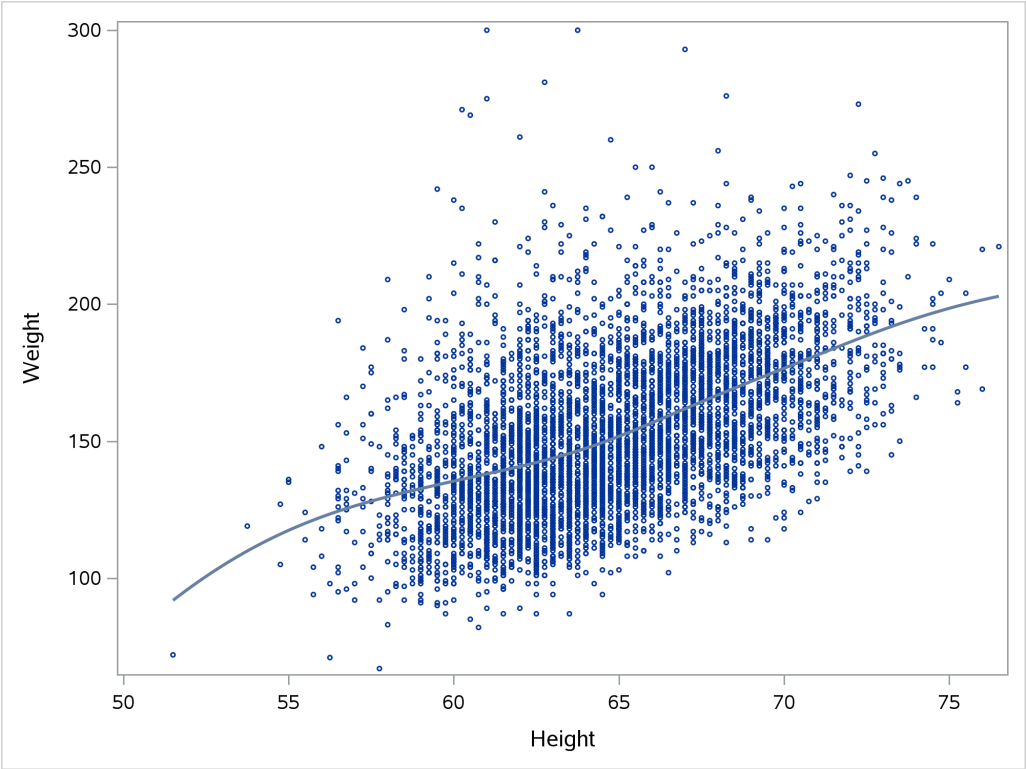

The resulting fit function is smooth and nonlinear. You do not need to know anything about the shape of the scatter plot. The PBSPLINE statement automatically finds a smooth fit while trying to guard against overfitting. It is not possible to write a simple equation for the model. For more information about how penalized B-splines work, see Chapter 126, "The TRANSREG Procedure."

NOTE: Antialiasing smooths the elements of a graph. The ANTIALIASMAX=6000 option enables antialiasing through 6,000 elements. By default, antialiasing is disabled after 4,000 elements.

Loess Fit Function

You can use the LOESS statement to find a loess fit function (Cleveland, Devlin, and Grosse 1988). Loess is a locally weighted scatter plot smoothing. The following step creates the plot in Output 24.6.3:

```
ods graphics on / loessmaxobs=6000;
proc sgplot data=sashelp.heart noautolegend;
   loess y=weight x=height / markerattrs=(size=3px);
run;
```

Output 24.6.3 Loess Fit Plot

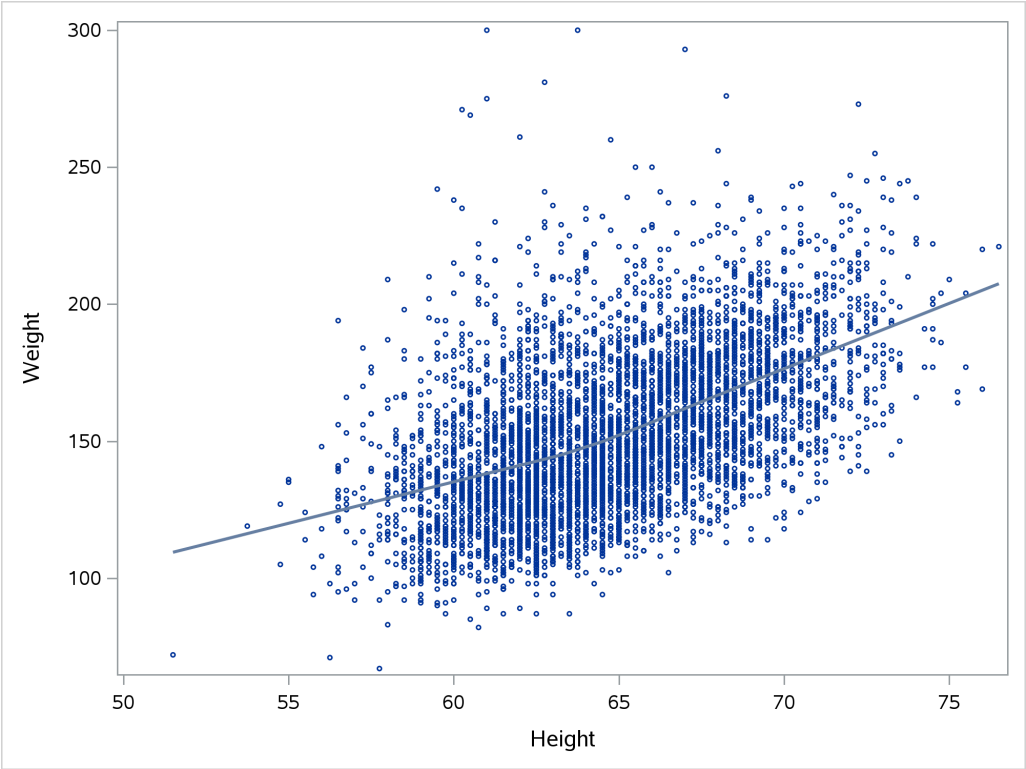

The loess fit is not a spline fit, but loess is similar to penalized B-splines in that it automatically tries to find a smooth fit while trying to guard against overfitting. It is not possible to write a simple equation for the model. For more information about loess, see Chapter 78, "The LOESS Procedure."

NOTE: Loess becomes computationally expensive with larger data sets. The LOESSOBSMAX=6000 option enables loess fits through 6,000 observations. By default, loess fits are disabled after 5,000 observations.

B-Spline Fit Function

You can use the PBSPLINE statement along with the option SMOOTH=0 to fit B-splines (De Boor 1978), which are equivalent to piecewise-polynomial splines. You specify SMOOTH=0 to disable all automatic smoothing. You specify the number of knots in the NKNOTS= option.[5] You use fewer knots to create smoother plots and more knots to enable greater curvature. The following step creates the plot in Output 24.6.4:

```
proc sgplot data=sashelp.heart noautolegend;
   pbspline y=weight x=height / smooth=0 nknots=5 markerattrs=(size=3px);
run;
```

[5] "Knots," without qualification, refer to interior knots—points inside the range of the x variable. Exterior knots, which are outside the range of the data, are explained in the section "Interior and Exterior Knots" on page 825.

Output 24.6.4 B-Spline Fit Plot

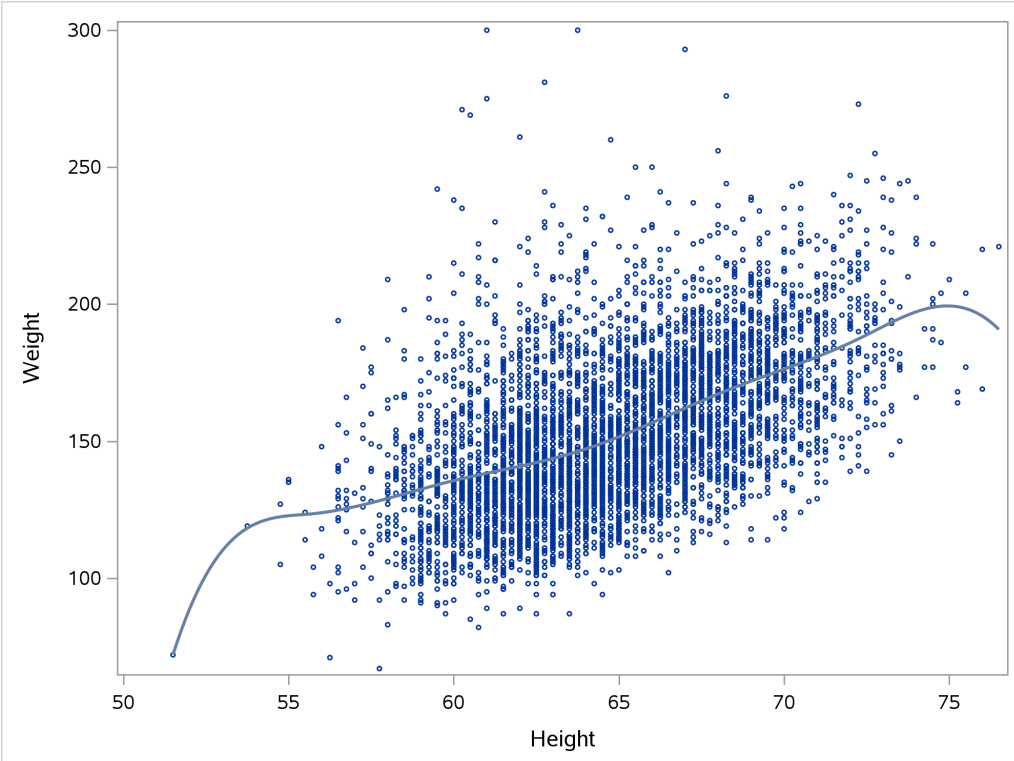

The resulting fit function is equivalent to those that you can obtain by using SPLINE (spline transformation), PSPLINE (polynomial-spline basis), or BSPLINE (B-spline basis) in the MODEL statement in PROC TRANSREG. Of all the functions shown in Output 24.6.2 through Output 24.6.4, the B-spline fit in Output 24.6.4 is most influenced by the extreme X values. The polynomial-spline model is

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 \beta_4 (x - k_1)_+^3 + \beta_5 (x - k_2)_+^3 + \beta_6 (x - k_3)_+^3 + \beta_8 (x - k_4)_+^3 + \beta_9 (x - k_5)_+^3 + \epsilon$$

The values k_1 through k_5 are the knots, which fall in the range of x. When $(x - k_i)$ is negative, $(x - k_i)_+ = 0$; otherwise, $(x - k_i)_+ = (x - k_i)$. The expression $(x - k_i)_+^p$ is called a truncated power function. Each knot term changes the cubic part of the polynomial as x advances beyond each knot.

For an introduction to piecewise-polynomial splines, see Smith (1979). For more information about how splines work, see Chapter 126, "The TRANSREG Procedure."

Understanding Splines and Knots

This section is optional; it shows some of the mathematical details of polynomial-spline models. As shown previously, the following is a cubic-polynomial regression model:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \epsilon$$

If you add n_k knots, it becomes a polynomial-spline regression model:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \sum_{i=1}^{n_k} \beta_{3+i}(x - k_i)_+^3 + \epsilon$$

Polynomial splines are easy to understand and describe. A curve has an overall intercept, linear portion, quadratic portion, and cubic portion. Then the cubic portion changes at each knot. Output 24.6.5 illustrates a spline that has knots at –2, 0, and 2. The blue function, $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$, extends from –5 to 5. The blue and red function, $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - -2)_+^3$, extends from –5 to almost 2 as it heads toward $y = \infty$. The red component first contributes to the overall function when $x - -2$ is positive. The blue, red, and green function, $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - -2)_+^3 + \beta_5 (x - 0)_+^3$, extends from –5 to just beyond 3 as it heads toward $y = -\infty$. The green component first contributes to the overall function when $x - 0$ is positive. The blue, red, green, and orange function, $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - -2)_+^3 + \beta_5 (x - 0)_+^3 + \beta_6 (x - 2)_+^3$, extends from –5 to almost 5 as it heads toward $y = \infty$. The orange component first contributes to the overall function when $x - 2$ is positive. Thus, $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - -2)_+^3 + \beta_5 (x - 0)_+^3 + \beta_6 (x - 2)_+^3$, which is highlighted in yellow, is the spline function, and it is composed of four component functions. The coefficients β_4, β_5, and β_6 are the change in the cubic portion of the spline. The intercept does not change; this makes the spline continuous. The linear and quadratic terms do not change; this makes the spline smooth.

Output 24.6.5 Polynomial-Spline Components

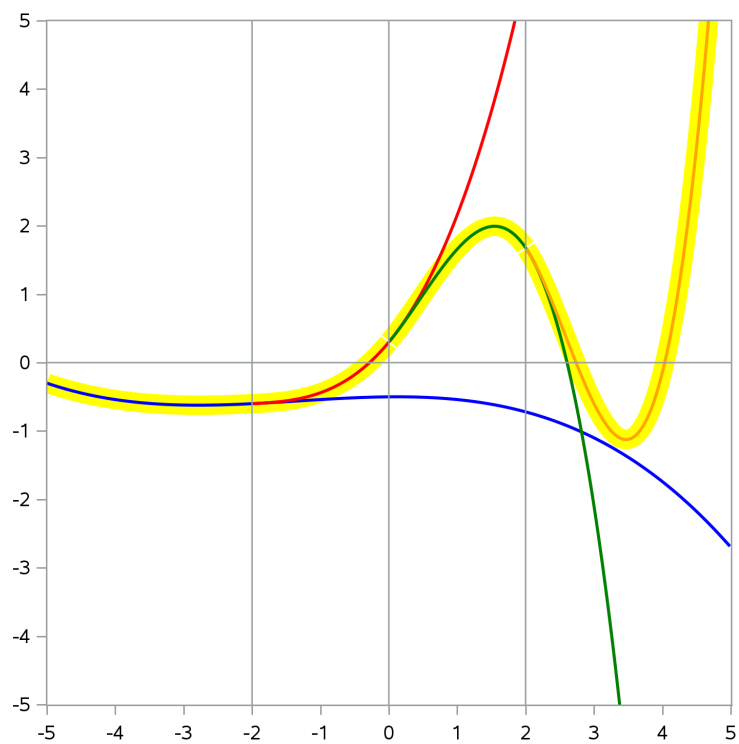

Mathematically, the cubic-polynomial spline is continuous, as are its first and second derivatives. Computationally, cubic-polynomial splines might be problematic, particularly for large data sets or when there are many knots. This is because some terms might be highly correlated, resulting in an unstable model.

In practice, B-splines are preferred over cubic-polynomial splines, although the two types of splines are equivalent. If $\mathbf{X}_p$ is a full-rank polynomial-spline basis and $\mathbf{X}_B$ is the corresponding full-rank B-spline basis, then there exists a matrix $\mathbf{T}$ such that $\mathbf{X}_B = \mathbf{X}_p\mathbf{T}$ and $\mathbf{X}_B\mathbf{T}^{-1} = \mathbf{X}_p$. For an illustration, see the section "B-Spline Basis" on page 827. The overall fit and R-square are the same, but because the basis columns of the $\mathbf{X}$ matrices are different, the regression coefficients are different. Regression coefficients for B-spline models are usually not interpretable.

Sparse Data and Overfitting

Penalized B-splines (and many other methods) do not work well with sparse data. For example, data sets might be small and sparse when you are following the results of a medical test for a single subject over time. You might want to fit a smooth function through the results rather than simply connect the dots. The following step creates the plot in Output 24.6.6:

```
proc sgplot data=sashelp.class(where=(sex='F'));
   pbspline y=weight x=height;
run;
```

Output 24.6.6 Overfit Sparse Data

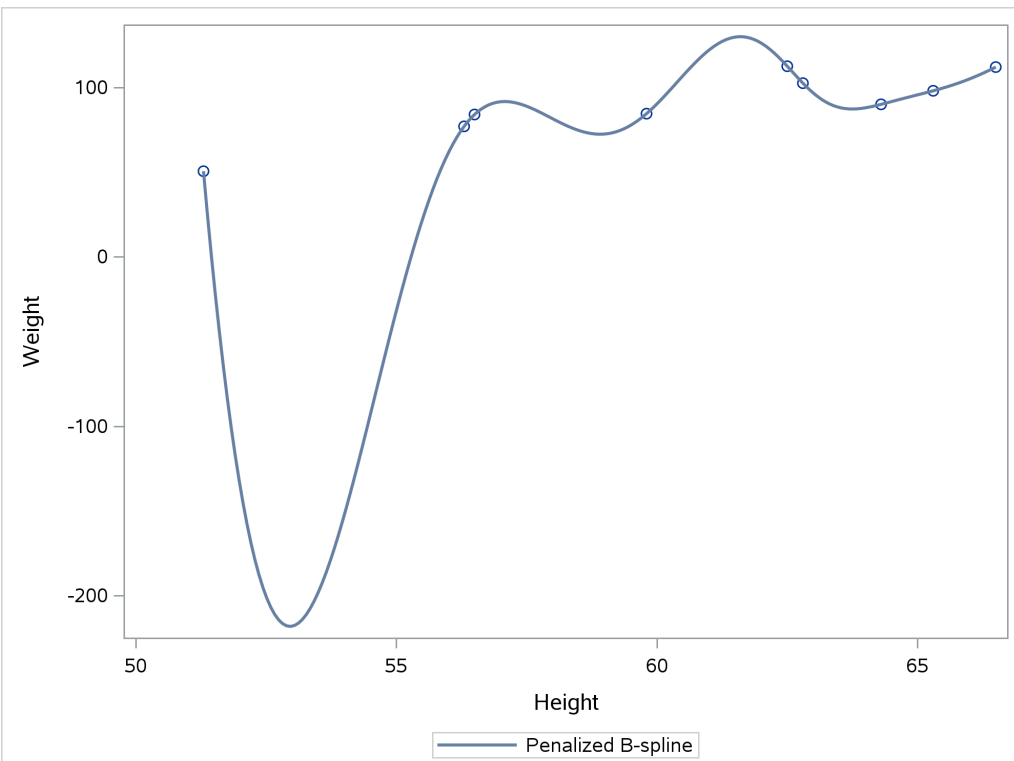

The X and Y coordinates of the fit function consist of 201 equally spaced X values that range from the minimum of X to its maximum. The Y values are computed from those X values and the coefficients. Penalized B-splines (and other spline methods) do not use the interpolation data when they compute the smoothing parameter and coefficients. First they analyze the actual data values, and then they compute the interpolated values.

The data are so sparse and the model has so many parameters that the computations fail to produce a

reasonable nonlinear fit function. You can see that the interpolated Y values extend well beyond the range of the data. You see this again in the section "Multiple Fit Functions Using PROC SGPLOT" on page 804, where you also see other remedies, including restricted splines. For information about interpolation, also see the section "Interpolation" on page 808.

A perfect connect-the-dots fit instead of the fit plot in Output 24.6.6 would look reasonable. In contrast, displaying the interpolated values more fully shows the spline function and shows that the results are overfit and not generalizable. You can manually set the smoothing parameter to get a better fit function. Larger values create a smoother fit function. The following step creates the plot in Output 24.6.7:

```
proc sgplot data=sashelp.class(where=(sex='F'));
   pbspline y=weight x=height / smooth=2e5;
run;
```

Output 24.6.7 Manually Set Smoothing Parameter

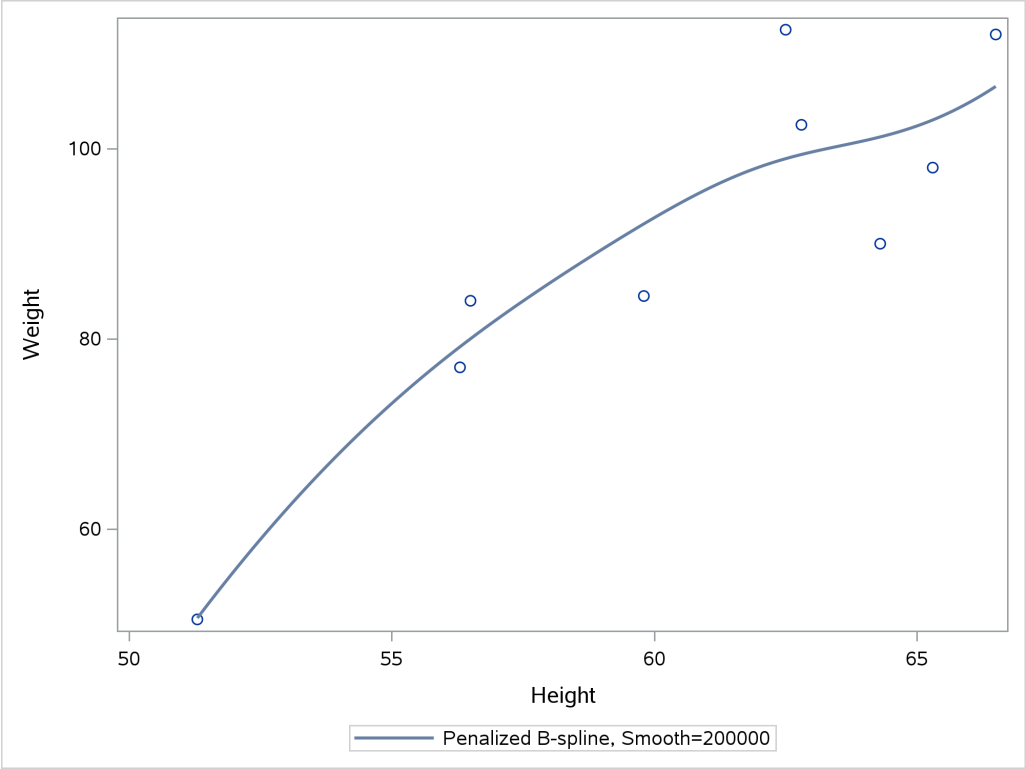

Usually, it takes some trial and error to set the smoothing parameter. You can use the penalized B-spline functionality in PROC TRANSREG to see which smoothing parameters it considers. PROC SGPLOT and PROC TRANSREG use the same code. Alternatively, you can specify fewer knots. By default, there are 100 knots. The following step specifies NKNOTS=20 and creates the plot in Output 24.6.8:

```
proc sgplot data=sashelp.class(where=(sex='F'));
   pbspline y=weight x=height / nknots=20;
run;
```

Output 24.6.8 Twenty Knots

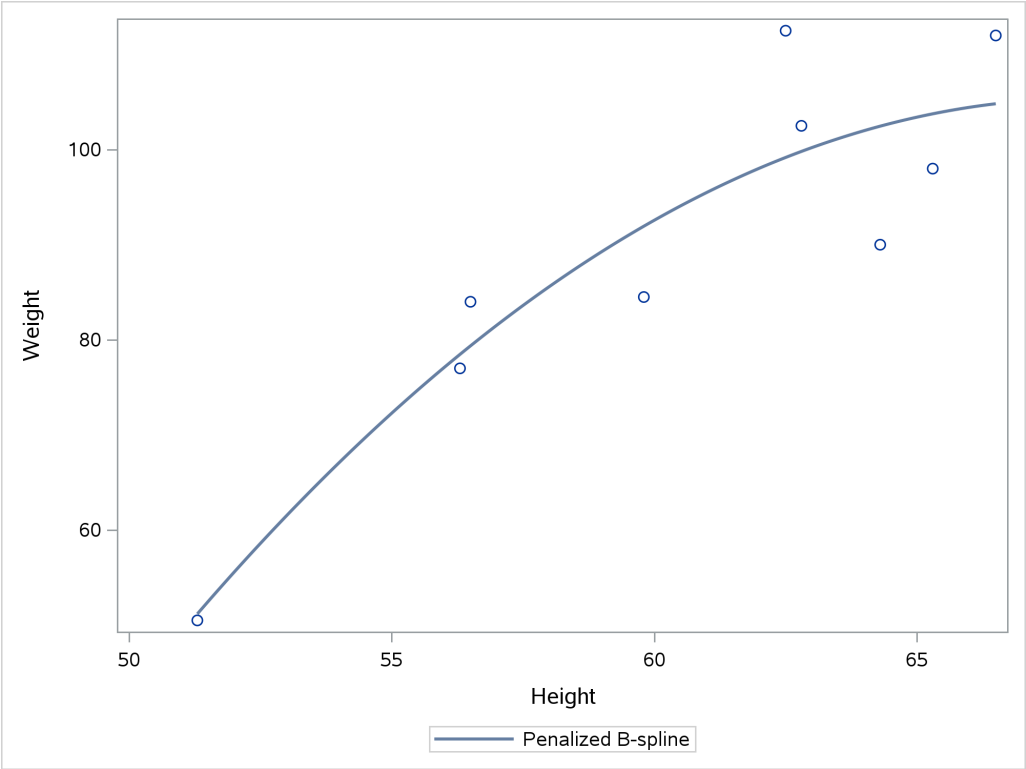

Multiple Fit Functions Using PROC SGPLOT

The next set of steps shows the same methods that were shown in the section "Single Fit Function Using PROC SGPLOT" on page 796 but this time using different data and a group variable.

Polynomial Fit Functions

You can use the REG statement with the DEGREE= and GROUP= options to fit multiple polynomial functions. The following step creates the plot in Output 24.6.9:

```
proc sgplot data=sashelp.gas;
   reg y=nox x=eqratio / degree=3 group=fuel markerattrs=(size=3px) name='a';
   keylegend 'a' / location=inside position=topright across=1;
run;
```

Output 24.6.9 Grouped Cubic Polynomial Fit Plot

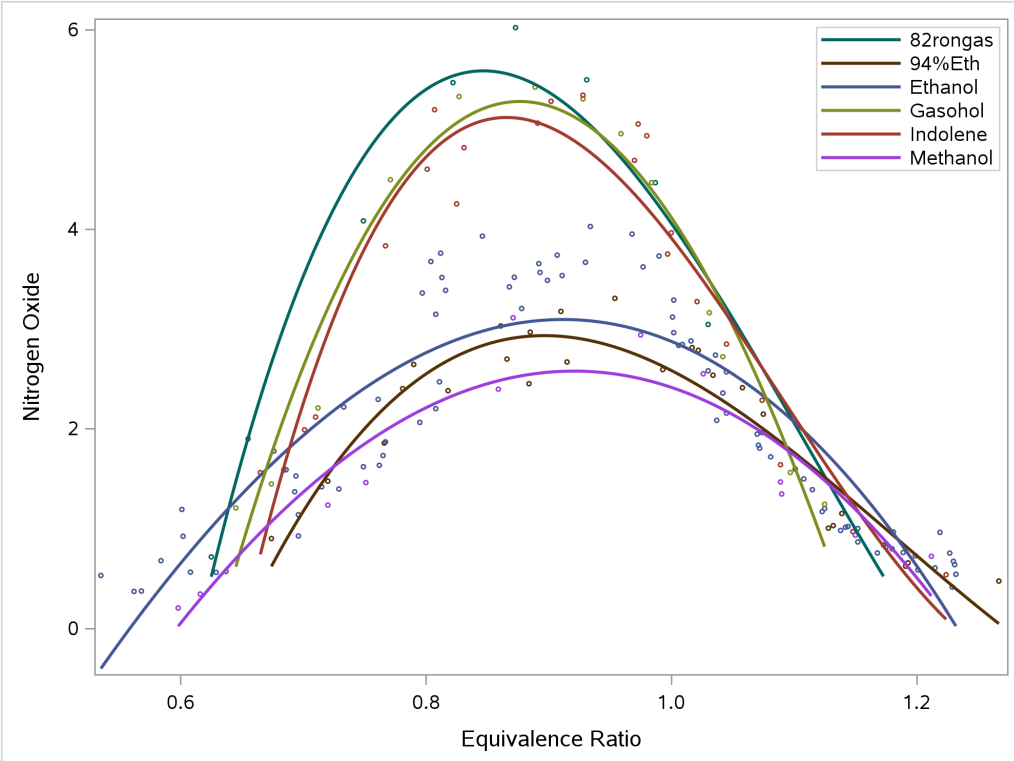

The plot in Output 24.6.9 contains six cubic polynomial functions, each independent of the others. Each function has the form

$$y_i = \beta_{i0} + \beta_{i1}x + \beta_{i2}x^2 + \beta_{i3}x^3 + \epsilon$$

If c_i is a binary indicator variable ($i = 1$ to 6) such that $\sum_{i=1}^{6} c_i = 1$, c_1 is 1 for `'82rongas'` and 0 for all other fuels, c_2 is 1 for `'94%Eth'` and 0 for all other fuels, and so on, then

$$y = \sum_{i=1}^{6} c_i \left[\beta_{i0} + \beta_{i1}x + \beta_{i2}x^2 + \beta_{i3}x^3\right] + \epsilon$$

Penalized B-Spline Fit Functions

You can use the PBSPLINE statement with the GROUP= option to fit multiple penalized B-splines. The following step creates the plot in Output 24.6.10:

```
proc sgplot data=sashelp.gas;
   pbspline y=nox x=eqratio / group=fuel markerattrs=(size=3px) name='a';
   keylegend 'a' / location=inside position=topright across=1;
run;
```

Output 24.6.10 Grouped Penalized B-Spline Fit Plot

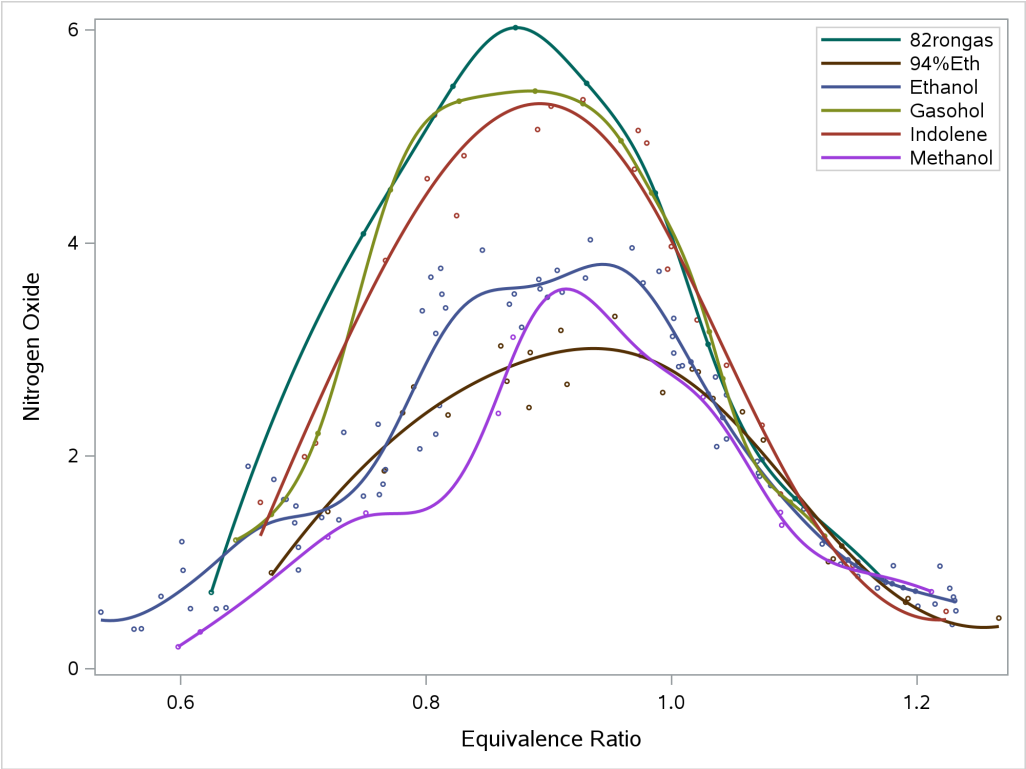

The penalized B-spline fit functions are less smooth than the polynomials.

Loess Fit Functions

You can use the LOESS statement with the GROUP= option to fit multiple loess curves. The following step creates the plot in Output 24.6.11:

```
proc sgplot data=sashelp.gas;
   loess y=nox x=eqratio / group=fuel markerattrs=(size=3px) name='a';
   keylegend 'a' / location=inside position=topright across=1;
run;
```

Output 24.6.11 Grouped Loess Fit Plot

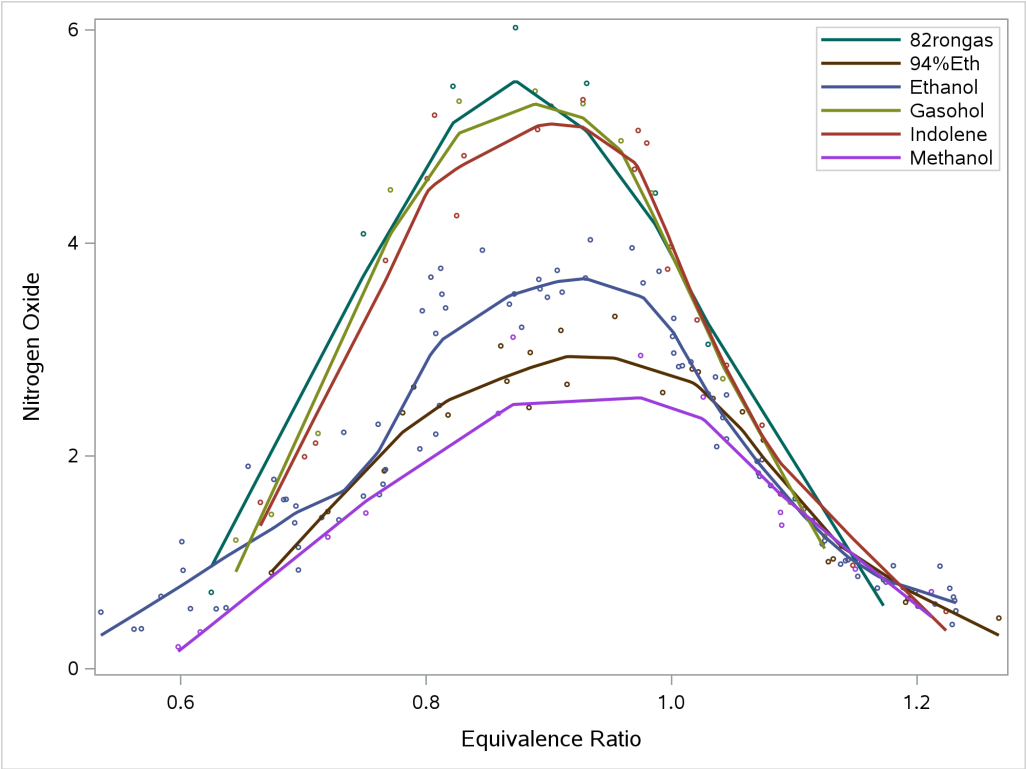

The loess spline fit functions are less smooth than the polynomials.

B-Spline Fit Functions

You can use the PBSPLINE statement, the SMOOTH=0 and NKNOTS= options, and the GROUP= option to fit multiple B-spline functions. The following step creates the plot in Output 24.6.12:

```
proc sgplot data=sashelp.gas;
   pbspline y=nox x=eqratio / group=fuel smooth=0 nknots=5
                              markerattrs=(size=3px) name='a';
   keylegend 'a' / location=inside position=topright across=1;
run;
```

Output 24.6.12 Grouped B-Spline Fit Plot

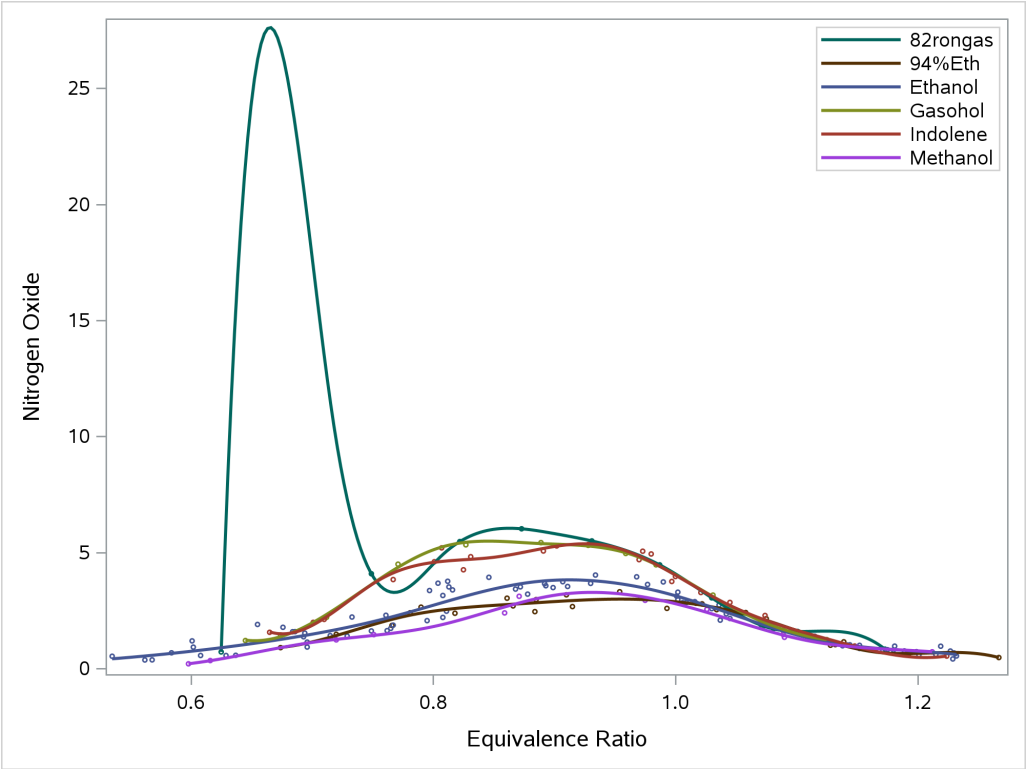

The polynomial-spline function is unsatisfactory for the `'82rongas'` function. This is because there is a large gap between the first two points. The function connects the dots between the first several points, and it is free to deviate from the data between points. This is a frequent concern with splines. Having fixed knots that apply to all functions, no matter how sparse they are, is problematic. In contrast, penalized B-splines and loess automatically smooth that section of that function. The polynomial functions do not have enough parameters to enable that level of nonlinearity.

Interpolation

You can examine the underlying ODS data object to better understand how PROC SGPLOT constructs the fit functions. The following step creates the plot, this time creating an ODS data set:

```
proc sgplot data=sashelp.gas;
   ods output sgplot=sg;
   pbspline y=nox x=eqratio / group=fuel smooth=0 nknots=5
                              markerattrs=(size=3px) name='a';
   keylegend 'a' / location=inside position=topright across=1;
run;
```

The next steps create and display a subset of the data:

```
data subset(drop=SORT_FUEL_RETAIN_ALL_);
   set sg;
   Obs = _n_;
   by PBSPLINE_EQRATIO_NOX_GROUP_S__GP fuel;
   if _N_ gt 169 then do; fuel = '_'; eqratio = ._; nox = ._; end;
```

```
   if first.fuel or last.fuel or first.PBSPLINE_EQRATIO_NOX_GROUP_S__GP or
      last.PBSPLINE_EQRATIO_NOX_GROUP_S__GP or obs = 169 then output;
   if lag(first.fuel) or lag(first.PBSPLINE_EQRATIO_NOX_GROUP_S__GP) then do;
      call missing(of PBSPLI: Fuel EqRatio NOx obs);
      if _n_ gt 169 then do; fuel = '_'; eqratio = ._; nox = ._; end;
      output; output; output;
   end;
run;

proc print noobs; id obs; run;
```

The results are shown in Output 24.6.13.

Output 24.6.13 Grouped Fit Function Data Object

Obs	PBSPLINE_ EQRATIO_NOX_ GROUP_S___X	PBSPLINE_ EQRATIO_NOX_ GROUP_S___Y	PBSPLINE_ EQRATIO_NOX_ GROUP_S__GP	Fuel	Eq Ratio	NOx
1	0.62500	0.7160	82rongas	82rongas	0.749	4.084
.	.	.			.	.
.	.	.			.	.
.	.	.			.	.
9	0.64692	22.8547	82rongas	82rongas	1.173	0.835
10	0.64966	24.2451	82rongas	94%Eth	0.993	2.593
.	.	.			.	.
.	.	.			.	.
.	.	.			.	.
34	0.71542	11.3943	82rongas	94%Eth	0.674	0.900
35	0.71816	10.4285	82rongas	Ethanol	1.152	0.866
.	.	.			.	.
.	.	.			.	.
.	.	.			.	.
124	0.96202	5.0358	82rongas	Ethanol	0.693	1.369
125	0.96476	4.9835	82rongas	Gasohol	0.645	1.207
.	.	.			.	.
.	.	.			.	.
.	.	.			.	.
137	0.99764	4.1538	82rongas	Gasohol	0.712	2.209
138	1.00038	4.0667	82rongas	Indolene	1.224	0.537
.	.	.			.	.
.	.	.			.	.
.	.	.			.	.
159	1.05792	2.1655	82rongas	Indolene	1.089	1.640
160	1.06066	2.0945	82rongas	Methanol	0.598	0.204
.	.	.			.	.
.	.	.			.	.
.	.	.			.	.
169	1.08532	1.6622	82rongas	Methanol	1.150	0.934

Output 24.6.13 *continued*

Obs	PBSPLINE_ EQRATIO_NOX_ GROUP_S___X	PBSPLINE_ EQRATIO_NOX_ GROUP_S___Y	PBSPLINE_ EQRATIO_NOX_ GROUP_S__GP	Fuel	Eq Ratio	NOx
201	1.173	0.83500	82rongas	_	_	_
202	0.674	0.90805	94%Eth	_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
402	1.267	0.47396	94%Eth	_	_	_
403	0.535	0.41974	Ethanol	_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
603	1.232	0.61021	Ethanol	_	_	_
604	0.645	1.22616	Gasohol	_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
804	1.125	1.24539	Gasohol	_	_	_
805	0.665	1.58007	Indolene	_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
1005	1.224	0.53708	Indolene	_	_	_
1006	0.598	0.20678	Methanol	_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
.	.	.		_	_	_
1206	1.212	0.72281	Methanol	_	_	_

Observations at the beginning and end of data groups are displayed. Missing values (ellipses) are displayed for other values. The first 169 observations contain the scatter plot variables Fuel, EqRatio, and NOx. After that, underscores indicate that those values are ignored. In the actual data set, which is too large to print in this example, observations 170 and beyond are excluded from the scatter plot because of missing values. All $6 \times 201 = 1{,}206$ observations contain interpolated coordinates for the six fit functions. The manufactured variable PBSPLINE_EQRATIO_NOX_GROUP_S__GP contains 201 copies of each of the six fuel values. The other manufactured variables, PBSPLINE_EQRATIO_NOX_GROUP_S___X and PBSPLINE_EQRATIO_NOX_GROUP_S___Y, provide the X and Y coordinates, respectively, for the curve for each fuel group.

The results of the following step (which are not shown) show that `'82rongas'` has only nine values:

```
proc freq data=sashelp.gas(where=(n(eqratio, nox) eq 2));
   tables fuel;
run;
```

Interpolation creates 201 interpolated values (200 line segments) from the minimum to the maximum by (maximum – minimum) / 200. You can specify the MAXPOINTS= option in the REG and PBSPLINE statements to change the number of interpolated values. Interpolation enables splines like the ones in Output 24.6.6 and Output 24.6.12—splines that have too many knots and too few values—to vary substantially from the original data. The automatic smoothing in penalized B-splines often prevents this variation from

happening, but not always. In most cases, it is good that ODS Graphics automatically interpolates, but not always. In the next section, you will see examples that do not use interpolation.

PROC TRANSREG

PROC TRANSREG was the first SAS/STAT modeling procedure to incorporate splines. Its syntax is different from that of other modeling procedures. It predates the EFFECT statement and options in the CLASS statement that other procedures now support.

B-Spline Fit Function

The following step creates a fit plot that has both a classification variable and a spline variable:

```
proc transreg data=sashelp.gas nomiss solve ss2 plots=fit(nocli noclm);
   ods select anova fitstatistics fitplot;
   model identity(nox) = spline(eqratio / nknots=5 evenly after) |
                         class(fuel / zero=none);
run;
```

The MODEL statement in PROC TRANSREG lists transformations, expansions, variables, and options. The IDENTITY transformation leaves the dependent variable unchanged. The CLASS expansion replaces the variable Fuel with a set of six binary variables, one for each type of fuel. The vertical bar in the MODEL statement creates an interaction between the SPLINE and CLASS variables. If there were no interaction, the SPLINE transformation would replace the variable EqRatio by a single new variable, which is a linear combination of the columns in a spline basis. Because there is an interaction between the SPLINE and CLASS variables, PROC TRANSREG finds a separate spline transformation for groups of values of the EqRatio variable. There is one group for each level of the CLASS variable (six total).

The model has an implicit intercept, because the ZERO=NONE option creates a binary variable for every level of the CLASS variable. The model has one parameter for each level of the CLASS variable (six total) and eight parameters (degree 3 plus 5 knots) for each spline transformation ($6 \times 8 = 48$ total), for a grand total of 54 parameters (53 model *df*). The AFTER option creates the knots for each term after the SPLINE and CLASS variables are combined to form the interaction term. The structural zeros (those that come from a zero in the binary CLASS variables) are ignored when the knots are found. The EVENLY option spaces the knots evenly; by default, knots are placed at the percentiles. The NOCLI and NOCLM options suppress the default prediction and confidence limits. The results are displayed in Output 24.6.14. Statistical procedures such as PROC TRANSREG display fit statistics in addition to plots. However, PROC TRANSREG does not automatically plot interpolated values. The functions are smoother when there are more data (such as for `'Ethanol'`) and less smooth for sparser functions (such as `'82rongas'`). The advantage of not interpolating is that the splines are less likely to leave the range of the data.

PROC SGPLOT finds separate fit functions by fitting separate models for each group. In contrast, PROC TRANSREG fits a single model, the one shown previously:

$$y = \sum_{i=1}^{6} c_i \left[\beta_{i0} + \beta_{i1}x + \beta_{i2}x^2 + \beta_{i3}x^3\right] + \epsilon$$

The two approaches are equivalent as long as the knots are the same. However, a procedure such as PROC TRANSREG gives you control over the model beyond anything that PROC SGPLOT gives you. For example, you can force the curves to be the same for each group by omitting the vertical bar from the MODEL statement (in other words, by omitting the interaction of the SPLINE and CLASS variables).

Output 24.6.14 Grouped Fit Function, Fit Statistics, and Plot

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	53	329.1232	6.209871	64.69	<.0001
Error	115	11.0388	0.095989		
Corrected Total	168	340.1619			

Root MSE	0.30982	**R-Square**	0.9675
Dependent Mean	2.34593	**Adj R-Sq**	0.9526
Coeff Var	13.20676		

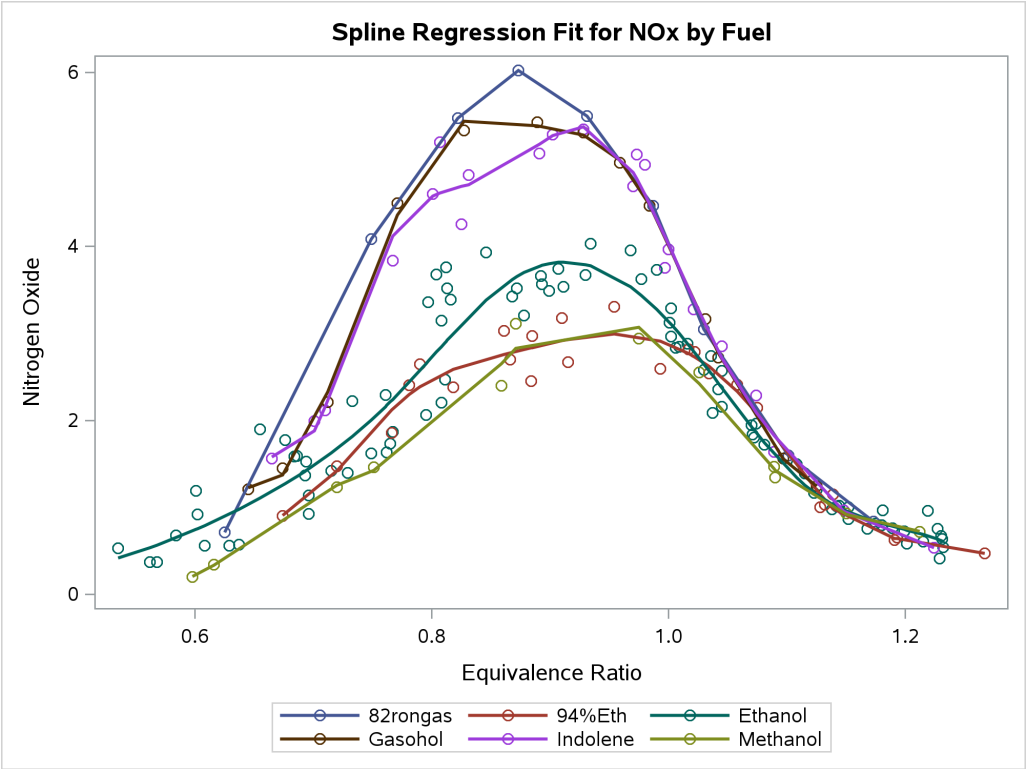

The following example shows how you can set up a data set for interpolation:

```
proc means data=sashelp.gas(where=(n(nox, eqratio))) noprint;
   class fuel;
   var eqratio;
   output out=m(where=(_type_ eq 1 and trim(_stat_) in ('MIN', 'MAX')));
run;
```

```
proc transpose data=m out=m2(drop=_:);
   by fuel;
   id _stat_;
   var eqratio;
run;

data gas(drop=min max);
   if _n_ = 1 then do i = 1 to n;
      set m2 nobs=n point=i;
      if fuel ne '82rongas' then
         do eqratio = min to max by (max - min) / 200; output; end;
   end;
   set sashelp.gas(where=(n(nox, eqratio)));
   output;
run;

proc transreg data=gas solve ss2 plots(interpolate)=fit(nocli noclm);
   ods select anova fitstatistics fitplot;
   model identity(nox) = class(fuel / zero=none) |
                         spline(eqratio / nknots=5 after evenly);
run;
```

PROC MEANS finds the minimum and maximum for each type of fuel. The WHERE= data set option ensures that only the observations without missing values in the quantitative variables are used. PROC TRANSPOSE arranges the minimum and maximum data set so that there is one observation for each fuel type. The Gas data set adds observations to provide interpolated fit, just as PROC SGPLOT does, except that no interpolated values are generated for the `'82rongas'` group. The results are displayed in Output 24.6.15.

Output 24.6.15 Interpolation

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	53	329.1232	6.209871	64.69	<.0001
Error	115	11.0388	0.095989		
Corrected Total	168	340.1619			

Root MSE	0.30982	**R-Square**	0.9675
Dependent Mean	2.34593	**Adj R-Sq**	0.9526
Coeff Var	13.20676		

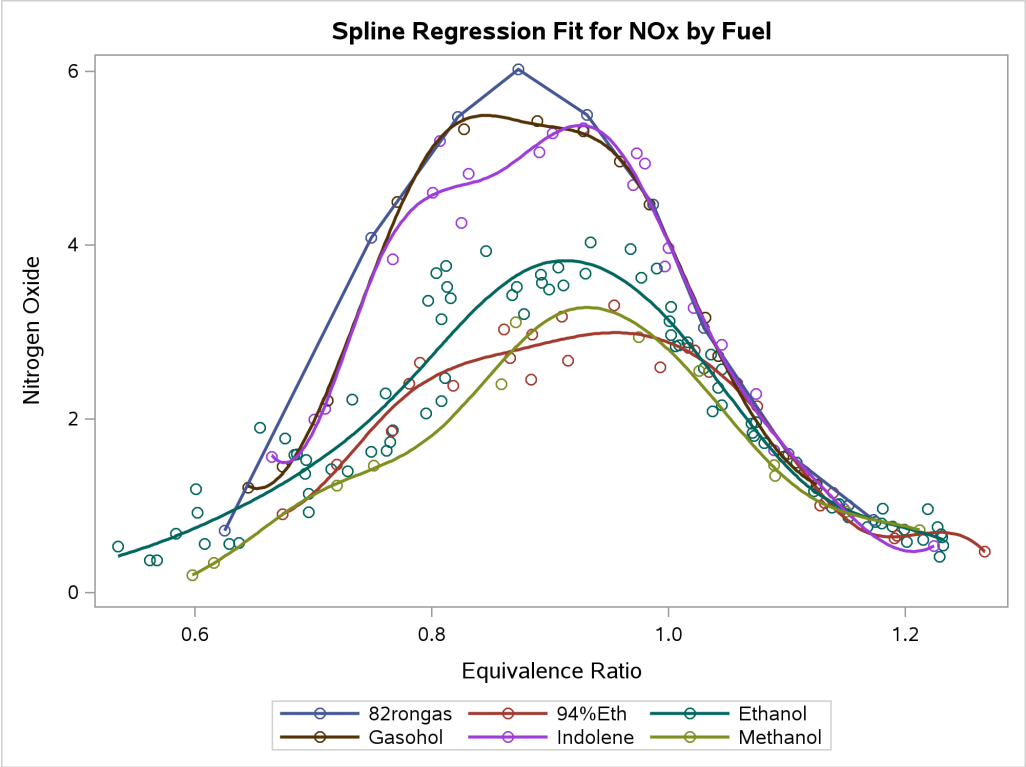

The first PROC TRANSREG step excludes observations that have missing values by using the NOMISS option and analyzes a data set that has 171 observations. The second PROC TRANSREG step excludes observations that have missing values by using a WHERE clause in a preceding DATA step and analyzes a data set that has 1,007 observations. All fit statistics match, because the two data sets contain exactly the same 171 observations that are analyzed. The second data set contains missing values in the variable NOx. Those observations are excluded in IDENTITY transformations. PROC TRANSREG has options for analyzing observations that have missing data, but they are not used for IDENTITY variables. PROC TRANSREG and many other procedures score observations that they exclude from the analysis when computing predicted values, residuals, confidence limits, and so on, as long as the relevant data are there to do the computations, even when those observations are not used in computing the sums of squares, degrees of freedom, coefficients, and so on. The PLOTS(INTERPOLATE) option specifies that those scored observations should be used in plotting the regression functions. When you compare the plots in Output 24.6.14 and Output 24.6.15, you see that the fit functions are the same for `'82rongas'` and are smoother in Output 24.6.15 for the other fuel types.

```
data gas(drop=min max);
   if _n_ = 1 then do i = 1 to n;
      set m2 nobs=n point=i;
      do eqratio = min to max by (max - min) / 200; output; end;
   end;
   set sashelp.gas(where=(n(nox, eqratio)));
   output;
run;

proc transreg data=gas solve ss2 plots(interpolate)=fit(nocli noclm);
   ods select anova fitstatistics fitplot;
   model identity(nox) = class(fuel / zero=none)
```

```
                                spline(eqratio / nknots=5 after evenly);
run;
```

Parallel Curves

The following steps fit a model without interactions and create the parallel fit functions that are displayed in Output 24.6.16.

Output 24.6.16 Interpolation

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	13	303.1972	23.32286	97.80	<.0001
Error	155	36.9647	0.23848		
Corrected Total	168	340.1619			

Root MSE	0.48835	**R-Square**	0.8913
Dependent Mean	2.34593	**Adj R-Sq**	0.8822
Coeff Var	20.81676		

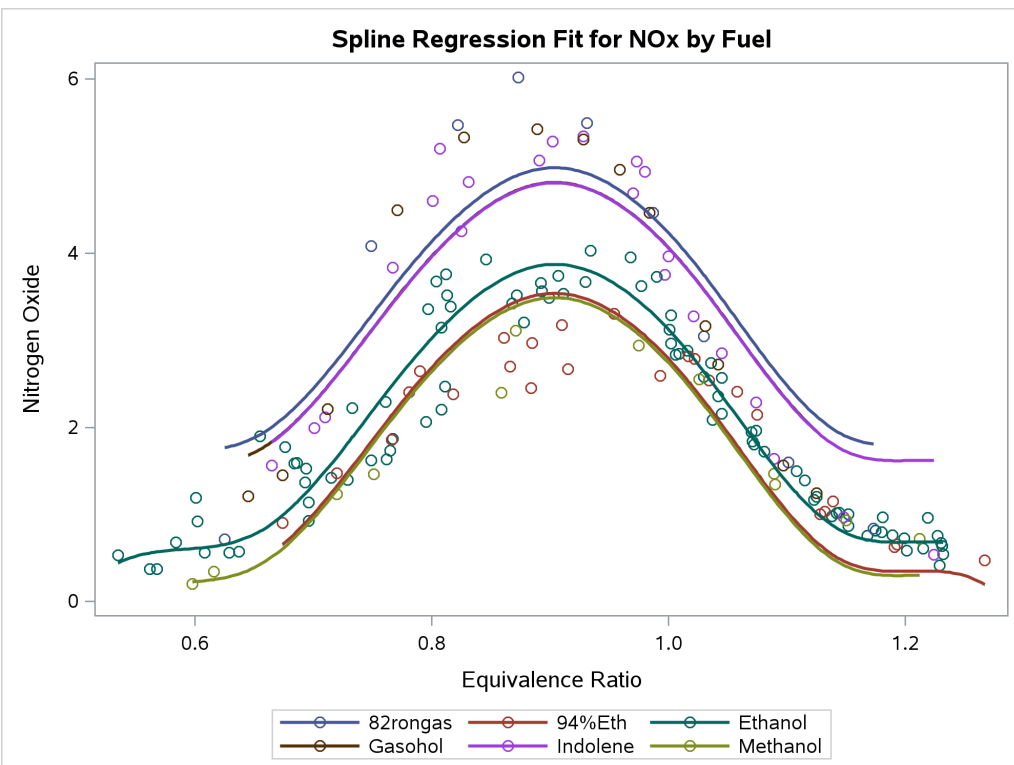

Because the plot has one spline curve and six intercepts, there is no need to guard against the lack of smoothness in the `'82rongas'` fuel type. R-square is smaller in this model because you did not fit a separate curve for each group. You cannot display results like this by using the statistical calculations in PROC SGPLOT; you must first compute the predicted values by using a procedure such as PROC TRANSREG.

Penalized B-Spline Fit Functions

The following step uses penalized B-splines and displays the results in Output 24.6.17:

```
proc transreg data=gas ss2 plots(interpolate)=fit(nocli noclm);
   ods select anova fitstatistics fitplot;
   model identity(nox) = class(fuel / zero=none) * pbspline(eqratio / after);
run;
```

Output 24.6.17 Interpolation

Univariate ANOVA Table, Penalized B-Spline Transformation

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	54.288	330.1741	6.081934	69.24	<.0001
Error	113.71	9.9878	0.087834		
Corrected Total	168	340.1619			

Root MSE	0.29637	R-Square	0.9706
Dependent Mean	2.34593	Adj R-Sq	0.9566
Coeff Var	12.63330		

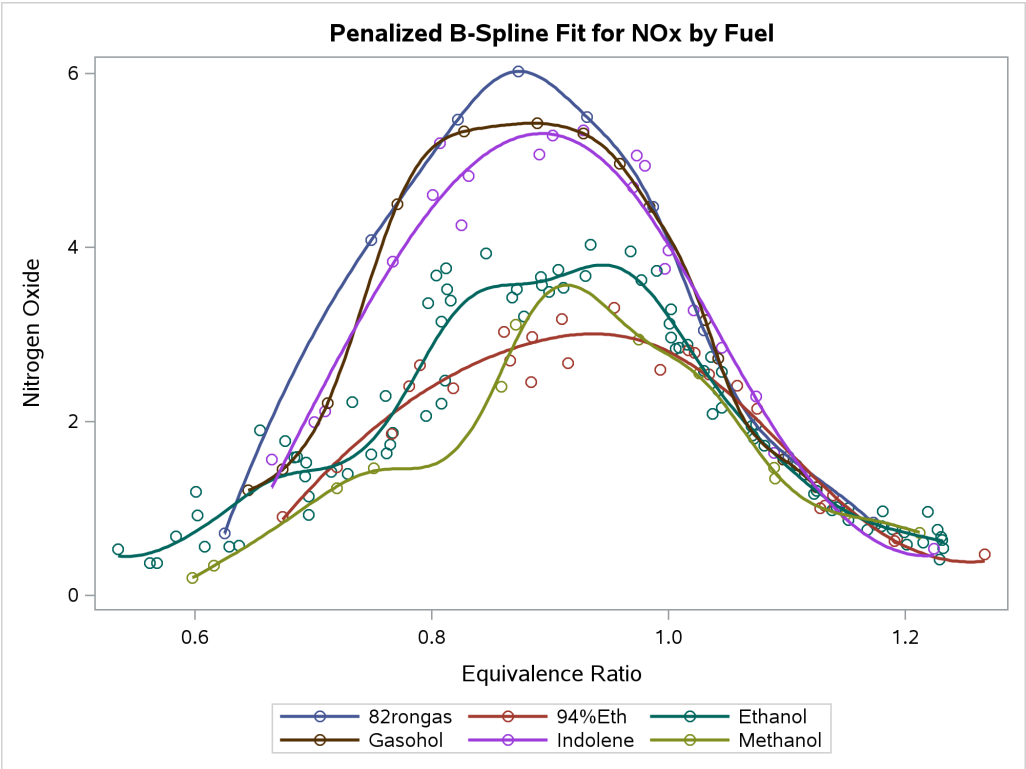

The MODEL statement uses an asterisk rather than a vertical bar to specify the interaction. PROC TRANSREG fits multiple types of models. There are linear models such as those shown previously where you can specify either a vertical bar or an asterisk, depending on the type of model that you want. There are also models whose results are computed by preprocessing. For these models, which include penalized

B-spline, smoothing spline, and Box-Cox models, you have no freedom to control the intercept terms. Each penalized B-spline has an intercept as part of the model.

Smoothing Spline Functions

PROC TRANSREG has other types of splines, including smoothing splines (Reinsch 1967). This type of spline was first available in PROC GPLOT. The following step fits separate smoothing splines for each fuel and displays the results in Output 24.6.18:

```
proc transreg data=gas solve ss2 plots(interpolate)=fit(nocli noclm);
   ods select anova fitstatistics fitplot;
   model identity(nox) = class(fuel / zero=none) *
                         smooth(eqratio / after sm=60);
run;
```

Output 24.6.18 Interpolation

Univariate ANOVA Table, Smooth Transformation

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	29.774	322.8643	10.84389	86.65	<.0001
Error	138.23	17.2977	0.12514		
Corrected Total	168	340.1619			

Root MSE	0.35375	**R-Square**	0.9491
Dependent Mean	2.34593	**Adj R-Sq**	0.9382
Coeff Var	15.07939		

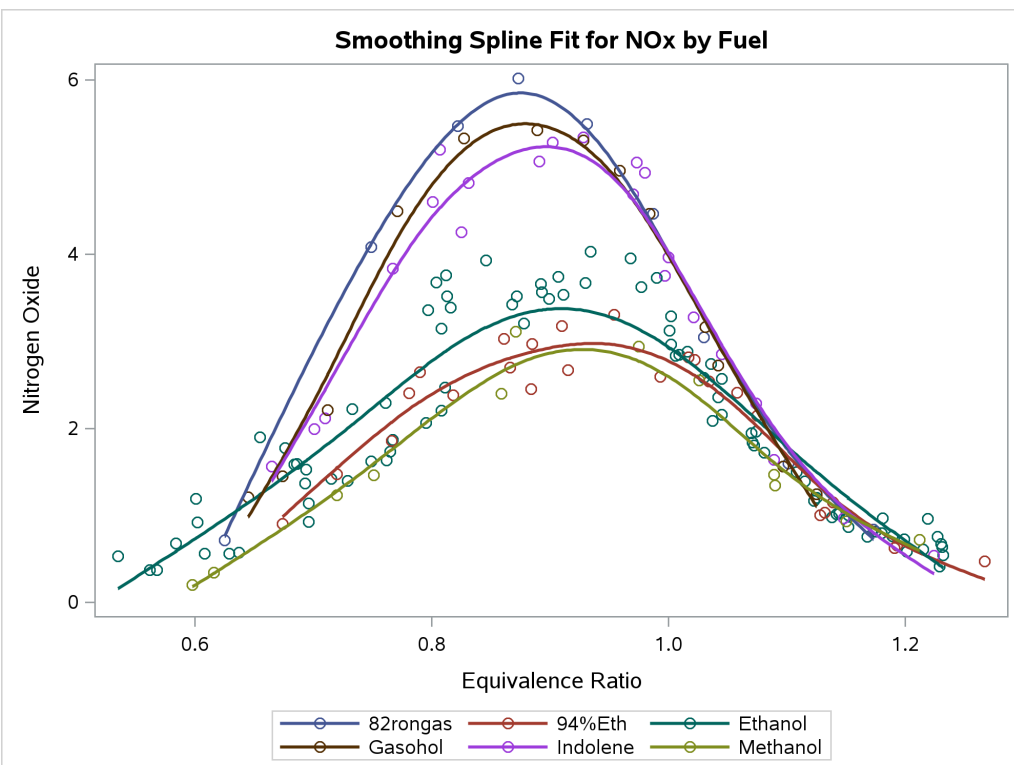

You use the SM= option to specify a smoothing parameter that ranges from 0 to 100. PROC TRANSREG does not pick an optimal smoothing parameter for you.

Monotone Splines

This example uses artificial data to illustrate monotone splines (Winsberg and Ramsay 1980):

```
data x;
   do i = 1 to 100;
      g = 1;
      x = 10 * uniform(17);
      y = x + 2 * sin(x) + normal(17);
      output;
      g = 2;
      x = 10 * uniform(17);
      y = 5 - x - 2 * cos(x) + normal(17);
      output;
   end;
run;

proc sgplot data=x;
   title 'Penalized-B-Spline';
   pbspline y=y x=x / group=g;
run;
title;

proc transreg data=x ss2 plots=fit(nocli noclm) maxiter=100;
   model identity(y) = class(g / zero=none) | mspline(x / nknots=10);
run;
```

The results are displayed in Output 24.6.19. The first plot, which is produced by PROC SGPLOT, shows the penalized B-spline fit functions, which are not monotonic. The second plot, which is produced by PROC TRANSREG, shows the monotone spline fit functions, which increase for the first group and decrease for the second group. Monotone splines are flat in areas where other splines increase and decrease. A monotone spline transformation of a variable x is always at least weakly increasing (that is, it is increasing or flat). The monotone spline fit function is always (at least weakly) increasing or decreasing, depending on the relationship between y and x. PROC TRANSREG is an iterative procedure. In many cases, you can specify the SOLVE option to directly compute a solution without iterations. However, iterations are required for monotone splines. Monotone splines are quadratic by default (DEGREE=2); most other splines are cubic by default.

Output 24.6.19 Penalized B-Splines and Monotone Splines

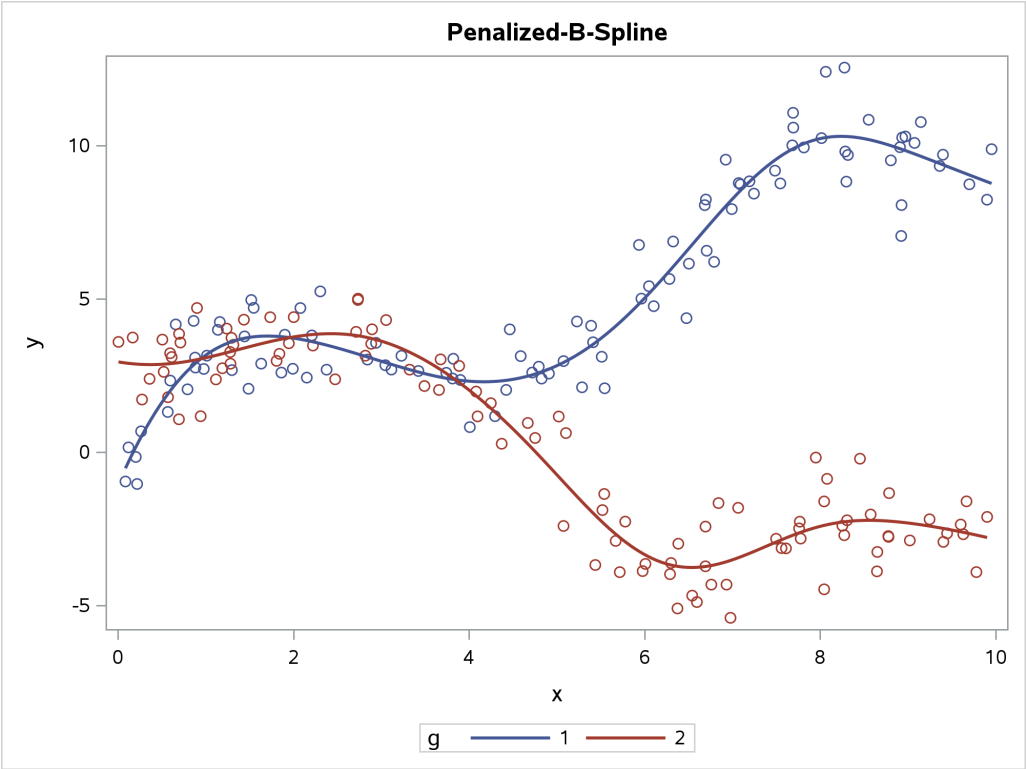

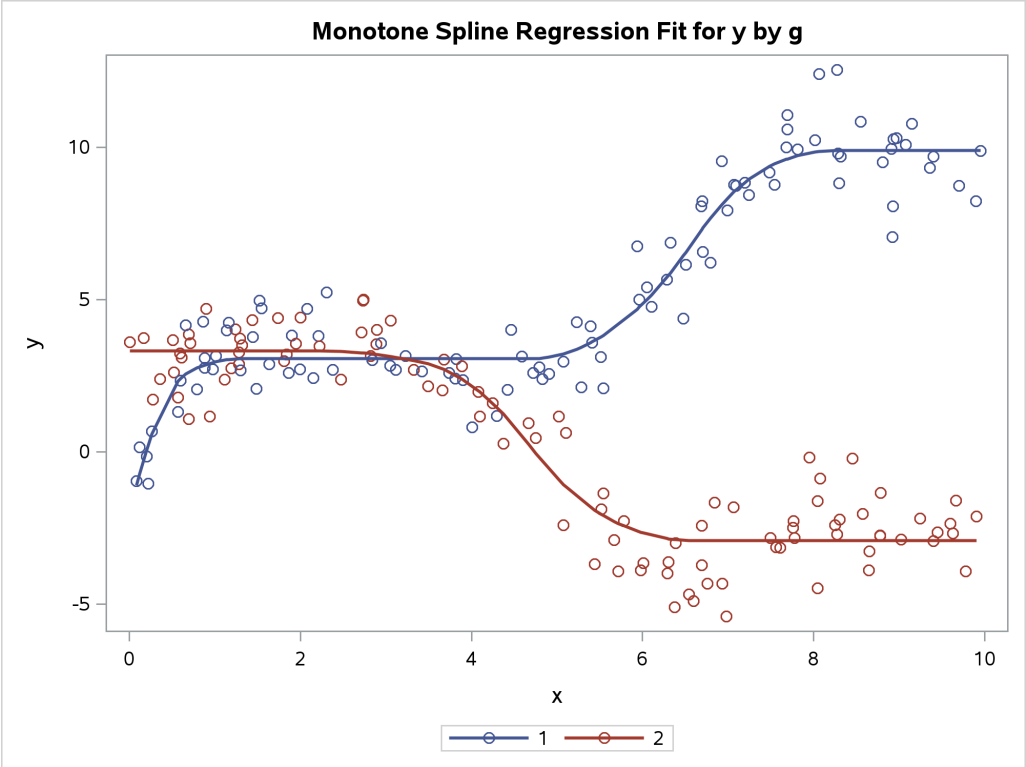

Piecewise-Linear Splines

You can specify the option DEGREE=1 along with splines or monotone splines for a piecewise-linear fit function:

```
proc transreg data=x ss2 plots=fit(nocli noclm) solve;
   model identity(y) = class(g / zero=none) | spline(x / nknots=10 degree=1);
run;

proc transreg data=x ss2 plots=fit(nocli noclm) maxiter=100;
   model identity(y) = class(g / zero=none) | mspline(x / nknots=10 degree=1);
run;
```

The results are shown in Output 24.6.20.

Output 24.6.20 Piecewise-Linear Splines

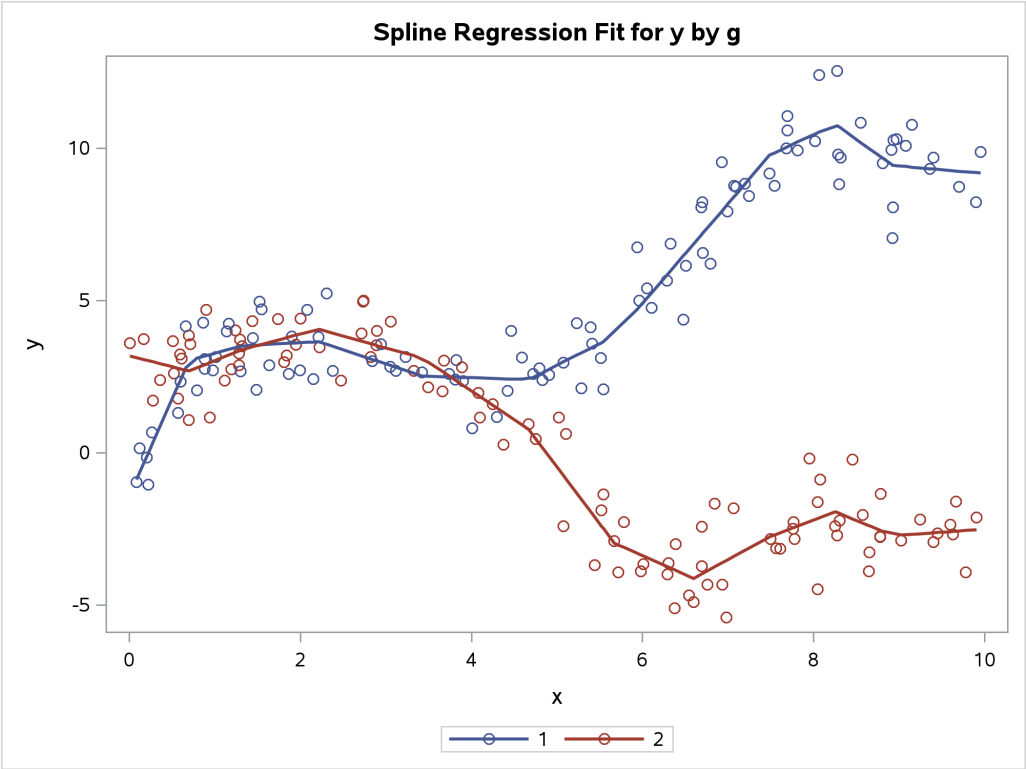

Output 24.6.20 *continued*

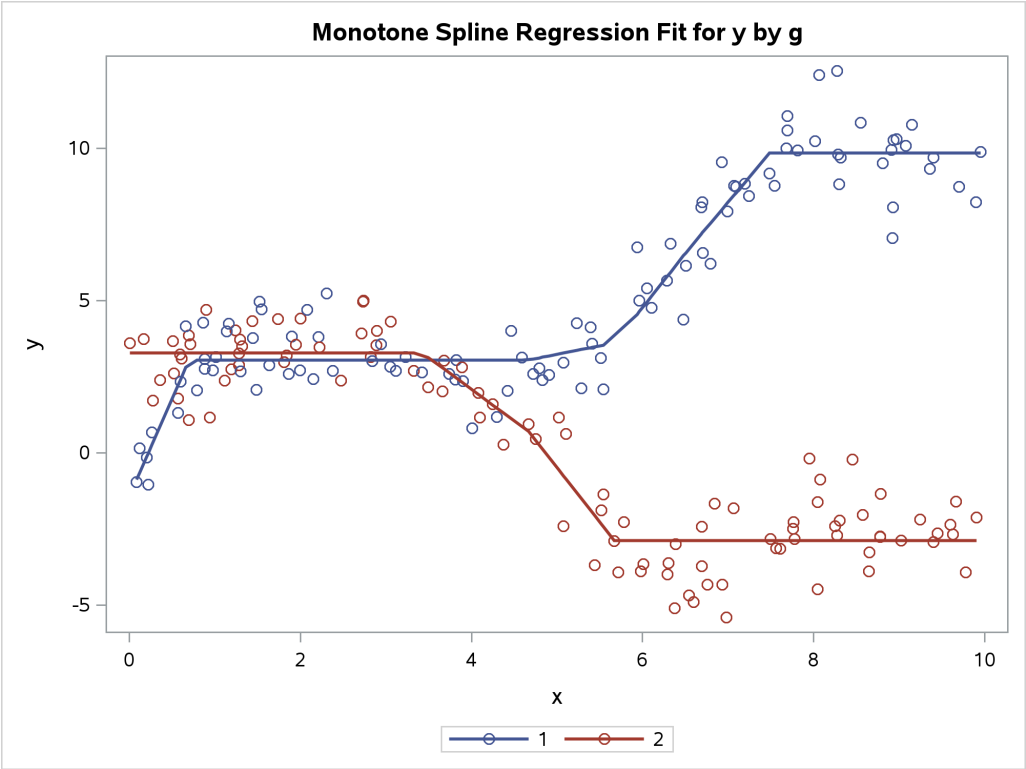

Outputting Splines

You can create an output data set from a SAS/STAT procedure such as PROC TRANSREG and then use PROC SGPLOT to display the results. This enables you to use the full power of the statistical procedure to fit models and customize the results without modifying a template. The following steps create the plot in Output 24.6.21, which has legend and marker customizations:

```
proc transreg data=x ss2 plots=fit(nocli noclm) maxiter=100;
   model identity(y) = class(g / zero=none) | mspline(x / nknots=10);
   output out=msp p replace;
run;

proc sort data=msp; by g x; run;

proc sgplot data=msp;
   title 'Transreg Output Data Set';
   scatter y=y  x=x / group=g markerattrs=(size=3px);
   series  y=py x=x / group=g name='a';
   keylegend 'a' / location=inside position=topleft across=1;
run;
title;
```

Output 24.6.21 Monotone Splines Displayed by PROC SGPLOT

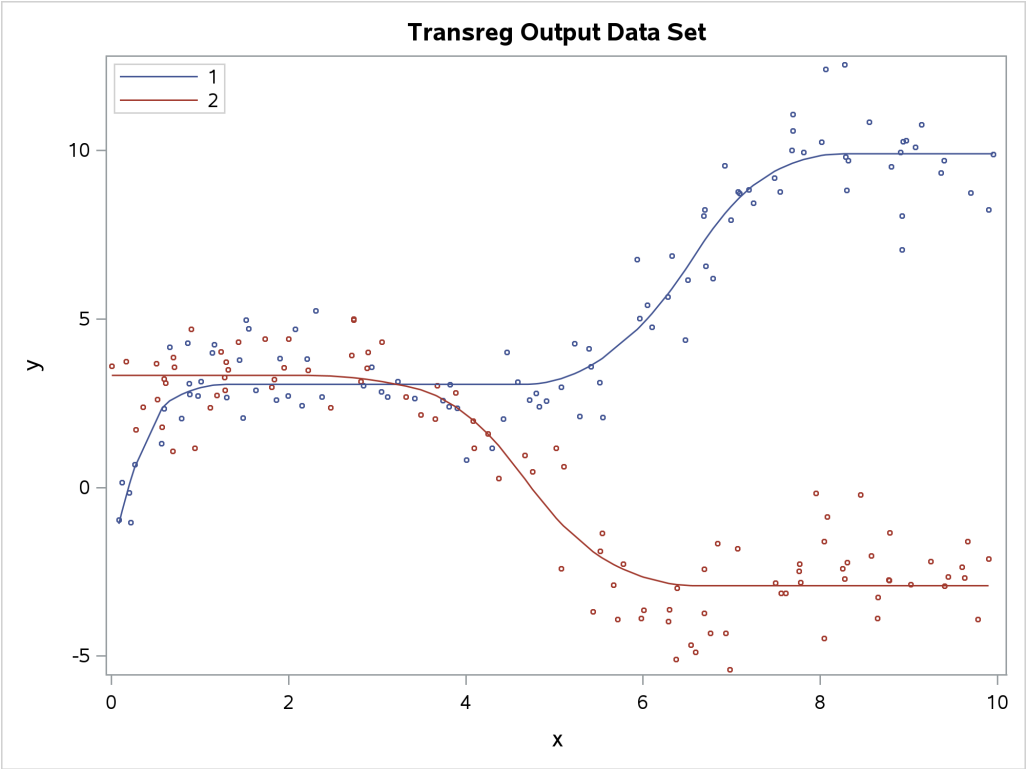

The EFFECT and EFFECTPLOT Statements

Splines are available in many other procedures when you use the EFFECT statement. The following procedures provide an EFFECT statement: GLIMMIX, GLMSELECT, HPMIXED, LOGISTIC, ORTHOREG, PHREG, PLM, PROBIT, QUANTLIFE, QUANTREG, QUANTSELECT, ROBUSTREG, SURVEYLOGISTIC, and SURVEYREG.

B-Spline Fit Function

The following step fits a spline that has five equally spaced knots:

```
proc orthoreg data=sashelp.heart;
   effect spl = spline(height / knotmethod=equal(5));
   model weight = spl;
   effectplot / obs;
run;
```

The EFFECT statement names the variable Height as a SPLINE variable and requests five evenly spaced knots. The name in the EFFECT statement, Spl, is then specified in the MODEL statement. Notice that the method is similar to the method in PROC TRANSREG, even though elements of the syntax are different. The EFFECTPLOT statement displays the fit function, and the OBS option adds the scatter plot to the graph. Note that the *plot-type*, which can appear before the slash, is not specified. The results are displayed in Output 24.6.22.

Output 24.6.22 EFFECT and EFFECTPLOT Statements

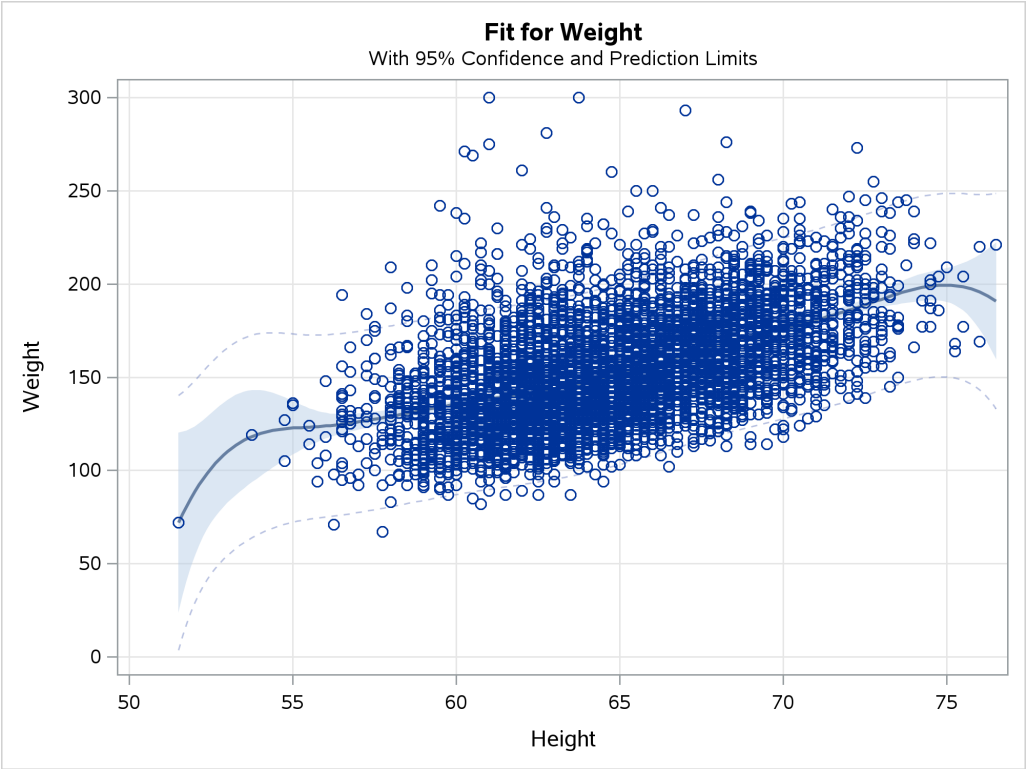

B-Spline Fit Functions

The EFFECTPLOT statement acts like PROC SGPLOT in that it automatically adds observations for interpolation (and in some cases extrapolation). The following step creates the plot in Output 24.6.23:

```
proc orthoreg data=sashelp.gas;
   effect spl = spline(eqratio / knotmethod=equal(3));
   class fuel;
   model nox = spl | fuel;
   effectplot / obs extend=data;
   ods output SliceFitPlot=sp;
run;
```

Output 24.6.23 EFFECT and EFFECTPLOT Statements with a CLASS Variable

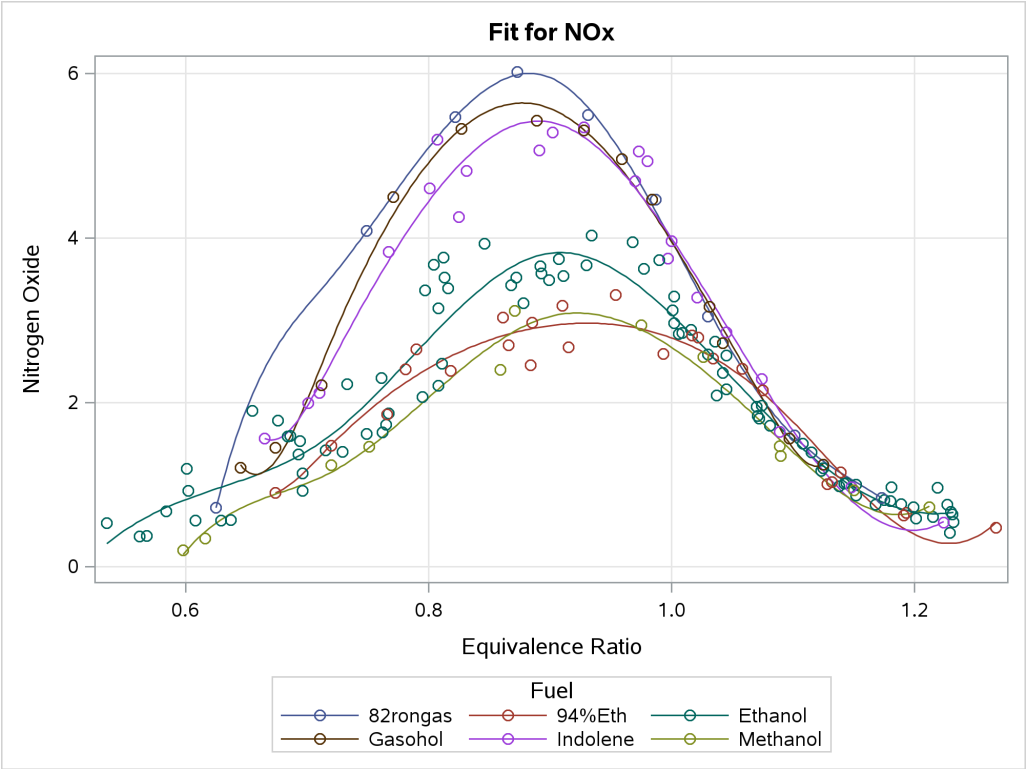

Note that the *plot-type*, which can appear before the slash, must not be specified for an interaction model like this. Furthermore, you must specify the EXTEND=DATA option for this type of plot. It causes the interpolation for plotted values to be between the minimum and maximum for each CLASS level, as is shown in the example that created the plot in Output 24.6.15. Without this option, the function for each group is generated through the range of the minimum and maximum of the spline variable EqRatio. Splines often rapidly head to plus or minus infinity as you extrapolate beyond the range of your data. The option does not affect the analysis; it affects only the plot construction. The number of knots was changed to three in this example. Five knots illustrate an even more extreme version of the same problem that was shown in Output 24.6.12. Because the `'82rongas'` function is sparse, that function is not well behaved when there are five knots.

Natural Cubic Splines (Restricted Splines)

You can get much better results by using natural cubic splines (Hastie, Tibshirani, and Friedman 2009), which are also known as restricted splines. The following step creates the plot in Output 24.6.24:

```
proc orthoreg data=sashelp.gas;
   effect spl = spline(eqratio / naturalcubic knotmethod=equal(5));
   class fuel;
   model nox = spl | fuel;
   effectplot / obs;
run;
```

Output 24.6.24 Natural Cubic (Restricted) Splines

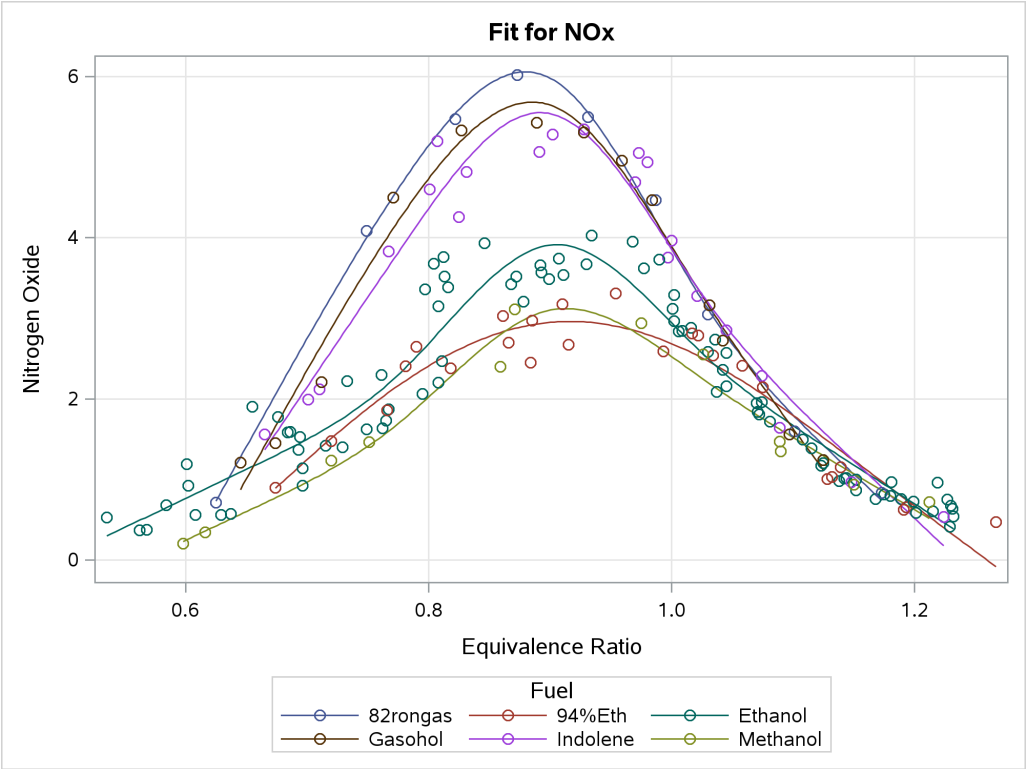

Restricted splines are linear beyond the interior knots, so they tend to be better behaved at the tails, particularly with a sparse CLASS level such as `'82rongas'`. There are 30 parameters in this restricted-spline model. In contrast, the full spline model has 51 parameters.

B-Splines

This section provides details about how B-splines work and the ways in which you can use them.

Interior and Exterior Knots

Polynomial splines have interior knots (knots that are inside the range of the minimum and maximum x). B-splines have interior knots and exterior knots (knots that are outside the range of the minimum and maximum x). For example, if x ranges from –2 to 2, a reasonable knot vector for three interior knots is $\mathbf{k} = (-4-\epsilon, -3-\epsilon, -2-\epsilon, -1, 0, 1, 2+\epsilon, 3+\epsilon, 4+\epsilon)$, where ϵ is a small value such as 10^{-12}. These knots are all but evenly spaced. The addition and subtraction of ϵ ensure that the first three exterior knots are less than the minimum and the last three exterior knots are greater than the maximum. Statistical procedures such as PROC TRANSREG and the procedures that support the EFFECT statement enable you to explicitly control both interior and exterior knots. Simpler options enable you to specify the number of knots and let the procedure determine their values. In many cases, fit functions based on slightly different knots are only slightly different. However, when you fit a model to one set of data and then score another set of data, you must ensure that you use the same knots for both purposes. The following procedure steps all fit the same model and use the knot vector $\mathbf{k}$:

```
data x2;
   do x = -2 to 2 by 0.1;
      y = x + sin(x) + normal(17);
      output;
   end;
run;

proc transreg data=x2 details ss2;
   model identity(y) = bspline(x / nknots=3 evenly=3);
run;

proc transreg data=x2 details ss2;
   model identity(y) = spline(x / nknots=3 evenly=3);
run;

%let k = -4.000000000001 -3.000000000001 -2.000000000001 -1 0 1
          2.000000000001  3.000000000001  4.000000000001;
proc transreg data=x2 details ss2;
   model identity(y) = bspline(x / knots=&k);
run;

proc orthoreg data=x2;
   effect spl = spline(x / knotmethod=equal(3));
   model  y = spl / noint;
run;

proc orthoreg data=x2;
   effect spl = spline(x / knotmethod=listwithboundary(&k));
   model  y = spl / noint;
run;
```

Although the models are the same, how the effects are grouped can vary. PROC TRANSREG with the DETAILS option displays the knots shown in Output 24.6.25. PROC TRANSREG with the BSPLINE expansion fits a model that has no intercept and seven 1-*df* B-spline effects. PROC TRANSREG with the SPLINE transformation fits a model that has an intercept and one 6-*df* B-spline effect. PROC ORTHOREG fits a model that has no intercept and seven 1-*df* B-spline effects.

PROC TRANSREG, because it has more than a 30-year history, does not create evenly spaced knots by default. By default, in the data set X2, the knot list is $\mathbf{k} = (-2-\epsilon, -2-\epsilon, -2-\epsilon, -1, 0, 1, 2+\epsilon, 2+\epsilon, 2+\epsilon)$. (Exterior knots are repeated.) Both repeated and all-but-equally-spaced exterior knots work, but the latter method better matches later software developments.

Output 24.6.25 Interior and Exterior Knots

Model Statement Specification Details				
Type	**DF**	**Variable**	**Description**	**Value**
Dep	1	Identity(y)	DF	1
Ind	6	Spline(x)	Degree	3
			Number of Knots	3
			Exterior Knots	-4.000000000001000E+00
				-3.000000000001000E+00
				-2.000000000001000E+00
			Interior Knots	-1.000000000000000E+00
				8.8817841970012000E-16
				1.0000000000000000E+00
			Exterior Knots	2.0000000000010000E+00
				3.0000000000010000E+00
				4.0000000000010000E+00
			Coefficients	-17.07078951
				0.20741011
				-1.77174923
				0.58192568
				0.54636248
				3.33130267
				-8.42096356

B-Spline Basis

The following four steps show four equivalent methods of putting the B-spline basis into a SAS data set:

```
proc transreg data=x2 details ss2;
   model identity(y) = bspline(x / nknots=3 evenly=3);
   output out=b1(keep=x_:) replace;
run;

proc glimmix data=x2 outdesign=b2(drop=x y
   rename=(%macro ren; %do i = 1 %to 7; _x&i=x_%eval(&i-1) %end; %mend; %ren));
   effect spl = spline(x / knotmethod=equal(3));
   model y = spl / noint;
run;

%let k = -4.000000000001 -3.000000000001 -2.000000000001 -1 0 1
          2.000000000001  3.000000000001  4.000000000001;

proc iml;
   use x2(keep=x); read all into x;
   b = bspline(x, 3, {&k});
   vname = 'x_0' : rowcatc('x_' || char(ncol(b)-1));
   create b3 from b[colname=vname]; append from b;
quit;

data b4(keep=x:);
   %let d   = 3;                        /* degree                  */
   %let nkn = %eval(&d * 2 + 3);        /* total number of knots   */
```

```
   %let nb  = %eval(&d + 1 + 3);        /* number of cols in basis */
   array k[&nkn] _temporary_ (&k);      /* knots                   */
   array b[&nb] x_0 - x_%eval(&nb - 1); /* basis                   */
   array w[%eval(2 * &d)];              /* work                    */
   set x2;
   do i = 1 to &nb; b[i] = 0; end;

   * find the index of first knot greater than current data value;
   do ki = 1 to &nkn while(k[ki] le x); end;
   kki = ki - &d - 1;

   * make the basis;
   b[1 + kki] = 1;
   do j = 1 to &d;
      w[&d + j] = k[ki + j - 1] - x;
      w[j] = x - k[ki - j];
      s = 0;
      do i = 1 to j;
         t = w[&d + i] + w[j + 1 - i];
         if t ne 0.0 then t = b[i + kki] / t;
         b[i + kki] = s + w[&d + i] * t;
         s = w[j + 1 - i] * t;
      end;
      b[j + 1 + kki] = s;
   end;
run;
```

The %REN macro is used to rename the PROC GLIMMIX variables, because different procedures have different rules for naming variables. Each data set has seven columns (intercept, degree 3, and three knots, although no column corresponds directly to these terms). The B-spline basis has at most order 4 (degree + 1) nonzero values in each row. This sparseness can be exploited by modeling code, which is one of the many appealing characteristics of B-splines. The following steps compare and display the bases:

```
options nolabel;
proc compare error note briefsummary criterion=1e-12
   data=b1 compare=b2 method=relative(1);
run;

proc compare error note briefsummary criterion=1e-12
   data=b1 compare=b3 method=relative(1);
run;

proc compare error note briefsummary criterion=1e-12
   data=b1 compare=b4(drop=x) method=relative(1);
run;
options label;

proc sgplot data=b4;
   %macro s; %do i = 0 %to &nb-1; series y=x_&i x=x; %end; %mend; %s
   yaxis label='B-Spline Basis Functions';
run;
```

The PROC COMPARE output (not displayed here) shows that the matrices are identical. The B-spline basis is plotted in Output 24.6.26.

Output 24.6.26 B-Spline Basis

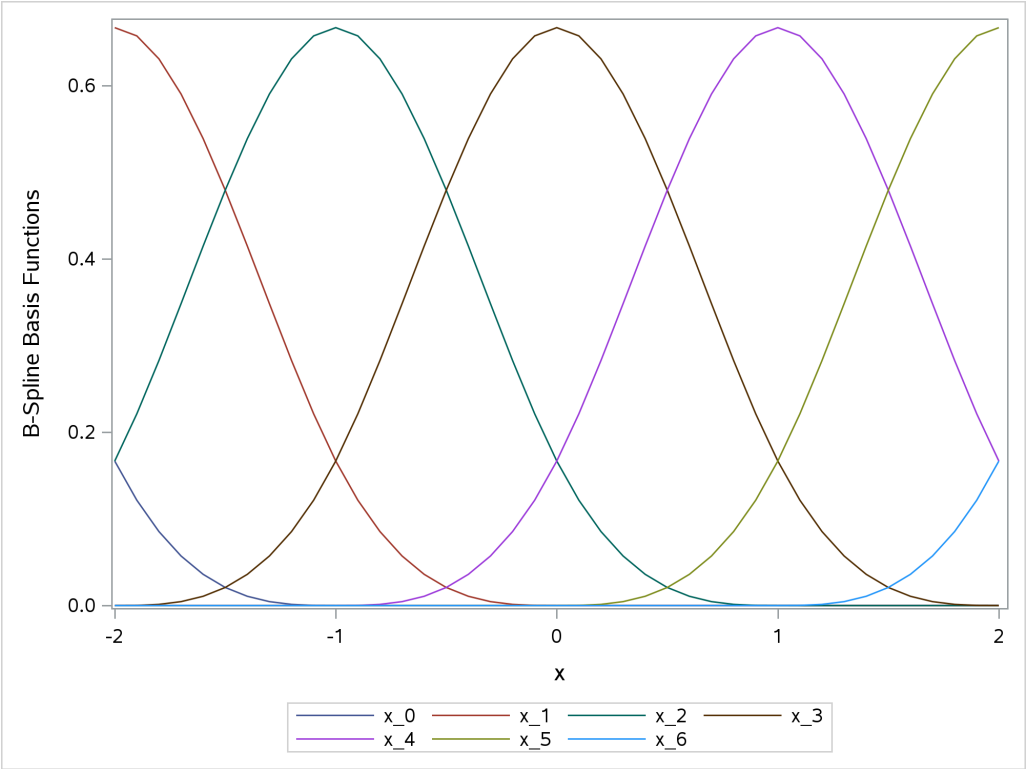

The next steps show that the polynomial-spline basis and the B-spline basis are linear transformations of each other:

```
proc transreg data=x2 details ss2;
   model identity(y) = pspline(x / name=(p) nknots=3);
   output out=p1(keep=p_:) replace;
run;

data both; merge p1 b1; run;

proc print data=both noobs;
   format _numeric_ bestd6.5 p_1 p_2 best6.;
run;

proc cancorr data=both;
  var x:;
  with p:;
run;
```

The polynomial spline (variables that begin with p_) and B-spline (variables that begin with x_) bases are displayed in Output 24.6.27. The variable p_1 matches the original X variable. All canonical correlations in Output 24.6.28 are 1.0. This shows that every vector in the space that is defined by the columns in the B-spline basis can be constructed from a linear combination of the columns of the polynomial-spline basis.

Output 24.6.27 Polynomial-Spline and B-Spline Bases

p_1	p_2	p_3	p_4	p_5	p_6	x_0	x_1	x_2	x_3	x_4	x_5	x_6
-2	4	-8	0	0	0	0.1667	0.6667	0.1667	0	0	0	0
-1.9	3.61	-6.859	0	0	0	0.1215	0.6572	0.2212	0.0002	0	0	0
-1.8	3.24	-5.832	0	0	0	0.0853	0.6307	0.2827	0.0013	0	0	0
-1.7	2.89	-4.913	0	0	0	0.0572	0.5902	0.3482	0.0045	0	0	0
-1.6	2.56	-4.096	0	0	0	0.0360	0.5387	0.4147	0.0107	0	0	0
-1.5	2.25	-3.375	0	0	0	0.0208	0.4792	0.4792	0.0208	0	0	0
-1.4	1.96	-2.744	0	0	0	0.0107	0.4147	0.5387	0.0360	0	0	0
-1.3	1.69	-2.197	0	0	0	0.0045	0.3482	0.5902	0.0572	0	0	0
-1.2	1.44	-1.728	0	0	0	0.0013	0.2827	0.6307	0.0853	0	0	0
-1.1	1.21	-1.331	0	0	0	0.0002	0.2212	0.6572	0.1215	0	0	0
-1	1	-1	0	0	0	0	0.1667	0.6667	0.1667	0	0	0
-0.9	0.81	-0.729	0.0010	0	0	0	0.1215	0.6572	0.2212	0.0002	0	0
-0.8	0.64	-0.512	0.0080	0	0	0	0.0853	0.6307	0.2827	0.0013	0	0
-0.7	0.49	-0.343	0.0270	0	0	0	0.0572	0.5902	0.3482	0.0045	0	0
-0.6	0.36	-0.216	0.0640	0	0	0	0.0360	0.5387	0.4147	0.0107	0	0
-0.5	0.25	-0.125	0.1250	0	0	0	0.0208	0.4792	0.4792	0.0208	0	0
-0.4	0.16	-0.064	0.2160	0	0	0	0.0107	0.4147	0.5387	0.0360	0	0
-0.3	0.09	-0.027	0.3430	0	0	0	0.0045	0.3482	0.5902	0.0572	0	0
-0.2	0.04	-0.008	0.5120	0	0	0	0.0013	0.2827	0.6307	0.0853	0	0
-0.1	0.01	-0.001	0.7290	0	0	0	0.0002	0.2212	0.6572	0.1215	0	0
0	0	0	1	0	0	0	0	0.1667	0.6667	0.1667	0	0
0.1	0.01	0.0010	1.3310	0.0010	0	0	0	0.1215	0.6572	0.2212	0.0002	0
0.2	0.04	0.0080	1.7280	0.0080	0	0	0	0.0853	0.6307	0.2827	0.0013	0
0.3	0.09	0.0270	2.1970	0.0270	0	0	0	0.0572	0.5902	0.3482	0.0045	0
0.4	0.16	0.0640	2.7440	0.0640	0	0	0	0.0360	0.5387	0.4147	0.0107	0
0.5	0.25	0.1250	3.3750	0.1250	0	0	0	0.0208	0.4792	0.4792	0.0208	0
0.6	0.36	0.2160	4.0960	0.2160	0	0	0	0.0107	0.4147	0.5387	0.0360	0
0.7	0.49	0.3430	4.9130	0.3430	0	0	0	0.0045	0.3482	0.5902	0.0572	0
0.8	0.64	0.5120	5.8320	0.5120	0	0	0	0.0013	0.2827	0.6307	0.0853	0
0.9	0.81	0.7290	6.8590	0.7290	0	0	0	0.0002	0.2212	0.6572	0.1215	0
1	1	1	8	1	0	0	0	0	0.1667	0.6667	0.1667	0
1.1	1.21	1.3310	9.2610	1.3310	0.0010	0	0	0	0.1215	0.6572	0.2212	0.0002
1.2	1.44	1.7280	10.648	1.7280	0.0080	0	0	0	0.0853	0.6307	0.2827	0.0013
1.3	1.69	2.1970	12.167	2.1970	0.0270	0	0	0	0.0572	0.5902	0.3482	0.0045
1.4	1.96	2.7440	13.824	2.7440	0.0640	0	0	0	0.0360	0.5387	0.4147	0.0107
1.5	2.25	3.3750	15.625	3.3750	0.1250	0	0	0	0.0208	0.4792	0.4792	0.0208
1.6	2.56	4.0960	17.576	4.0960	0.2160	0	0	0	0.0107	0.4147	0.5387	0.0360
1.7	2.89	4.9130	19.683	4.9130	0.3430	0	0	0	0.0045	0.3482	0.5902	0.0572
1.8	3.24	5.8320	21.952	5.8320	0.5120	0	0	0	0.0013	0.2827	0.6307	0.0853
1.9	3.61	6.8590	24.389	6.8590	0.7290	0	0	0	0.0002	0.2212	0.6572	0.1215
2	4	8	27	8	1	0	0	0	0	0.1667	0.6667	0.1667

Output 24.6.28 Canonical Correlations

Canonical Correlation Analysis

	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq) Eigenvalue	Difference	Proportion	Cumulative
1	1.000000	.	0.000000	1.000000	Infty	.	.	.
2	1.000000	.	0.000000	1.000000	Infty	.	.	.
3	1.000000	.	0.000000	1.000000	Infty	.	.	.
4	1.000000	.	0.000000	1.000000	Infty	.	.	.
5	1.000000	.	0.000000	1.000000	Infty	.	.	.
6	1.000000	.	0.000000	1.000000	Infty		.	.

Test of H0: The canonical correlations in the current row and all that follow are zero					
	Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
1	0.00000000	Infty	36	130.11	<.0001
2	0.00000000	Infty	25	112.95	<.0001
3	0.00000000	Infty	16	95.344	<.0001
4	0.00000000	Infty	9	78.03	<.0001
5	0.00000000	Infty	4	66	<.0001
6	0.00000000	Infty	1	34	<.0001

Scoring

Many procedures score observations that have missing values or have zero, missing, or invalid weights or frequencies. If the independent variables are all valid, then the procedure can compute predicted values. When the independent and dependent variables are all valid, then the procedure can also compute residuals. The following steps illustrate by using two simple regression models:

```
data class;
   set sashelp.class(rename=(height=Height1)) nobs=n;
   output;
   if _n_ = n then do;
      call missing(name, sex, height1);
      do age = 10 to 17; output; end;
   end;
run;

proc reg data=class;
   model height1 = age;
   output out=p1 p=p1y r=r1y;
run;

data class;
   f = 1;
   set sashelp.class(rename=(height=Height2)) nobs=n;
   output;
   if _n_ = n then do;
      call missing(f, name, sex);
```

```
      do age = 10 to 17; output; end;
   end;
run;

proc reg data=class;
   freq f;
   model height2 = age;
   output out=p2 p=p2y r=r2y;
run;

data all; merge p1 p2; run;

proc print data=all;
   var f name sex age height: p: r:;
   format p1y p2y r1y r2y 6.3;
run;
```

The results are displayed in Output 24.6.29.

Output 24.6.29 Scoring in a Simple Regression Model

Obs	f	Name	Sex	Age	Height1	Height2	p1y	p2y	r1y	r2y
1	1	Alfred	M	14	69.0	69.0	64.244	64.244	4.756	4.756
2	1	Alice	F	13	56.5	56.5	61.457	61.457	-4.957	-4.957
3	1	Barbara	F	13	65.3	65.3	61.457	61.457	3.843	3.843
4	1	Carol	F	14	62.8	62.8	64.244	64.244	-1.444	-1.444
5	1	Henry	M	14	63.5	63.5	64.244	64.244	-0.744	-0.744
6	1	James	M	12	57.3	57.3	58.670	58.670	-1.370	-1.370
7	1	Jane	F	12	59.8	59.8	58.670	58.670	1.130	1.130
8	1	Janet	F	15	62.5	62.5	67.031	67.031	-4.531	-4.531
9	1	Jeffrey	M	13	62.5	62.5	61.457	61.457	1.043	1.043
10	1	John	M	12	59.0	59.0	58.670	58.670	0.330	0.330
11	1	Joyce	F	11	51.3	51.3	55.882	55.882	-4.582	-4.582
12	1	Judy	F	14	64.3	64.3	64.244	64.244	0.056	0.056
13	1	Louise	F	12	56.3	56.3	58.670	58.670	-2.370	-2.370
14	1	Mary	F	15	66.5	66.5	67.031	67.031	-0.531	-0.531
15	1	Philip	M	16	72.0	72.0	69.818	69.818	2.182	2.182
16	1	Robert	M	12	64.8	64.8	58.670	58.670	6.130	6.130
17	1	Ronald	M	15	67.0	67.0	67.031	67.031	-0.031	-0.031
18	1	Thomas	M	11	57.5	57.5	55.882	55.882	1.618	1.618
19	1	William	M	15	66.5	66.5	67.031	67.031	-0.531	-0.531
20	.			10	.	66.5	53.095	53.095	.	13.405
21	.			11	.	66.5	55.882	55.882	.	10.618
22	.			12	.	66.5	58.670	58.670	.	7.830
23	.			13	.	66.5	61.457	61.457	.	5.043
24	.			14	.	66.5	64.244	64.244	.	2.256
25	.			15	.	66.5	67.031	67.031	.	-0.531
26	.			16	.	66.5	69.818	69.818	.	-3.318
27	.			17	.	66.5	72.605	72.605	.	-6.105

The first model excludes observations that have missing values. The second model excludes observations that have missing frequencies. The first model uses the variable Height1, which has missing values in the

extra observations that are to be scored. The predicted values and residual for the first model are displayed in the variables p1y and r1y, respectively. The second model uses the variable Height2, which has no missing values. The predicted values and residual for the second model are displayed in the variables p1y and r1y, respectively. The predicted values match for both models. The residuals match for the observations that do not have a missing height. For both models, the predicted values for the scored observations match the predicted values for the analysis observations that have the same age. Additionally, the procedure creates predicted values for heights that are not in the data set.

The following steps create observations for each fuel type that are to be scored:

```
proc means min max data=sashelp.gas;
   class fuel;
   var eqratio;
   output out=m(where=(_type_ eq 1 and _stat_ in ('MIN', 'MAX')));
run;

proc transpose data=m out=m2(drop=_:);
   var eqratio;
   by fuel;
   id _stat_;
run;

data score(drop=min max);
   set m2;
   do eqratio = min to max by (max - min) / 200; output; end;
run;
```

The following steps concatenate those observations to the data set and use PROC GLIMMIX to score them:

```
data gas;
   set sashelp.gas(where=(n(nox))) score;
run;

proc glimmix data=gas;
   effect spl = spline(eqratio / naturalcubic knotmethod=equal(5));
   class fuel;
   model nox = spl | fuel;
   output out=scored(where=(nmiss(nox))) pred=py;
run;
```

The next step displays the scores:

```
proc sgplot data=scored;
   series y=py x=eqratio / group=fuel;
run;
```

Output 24.6.30 Scored Observations

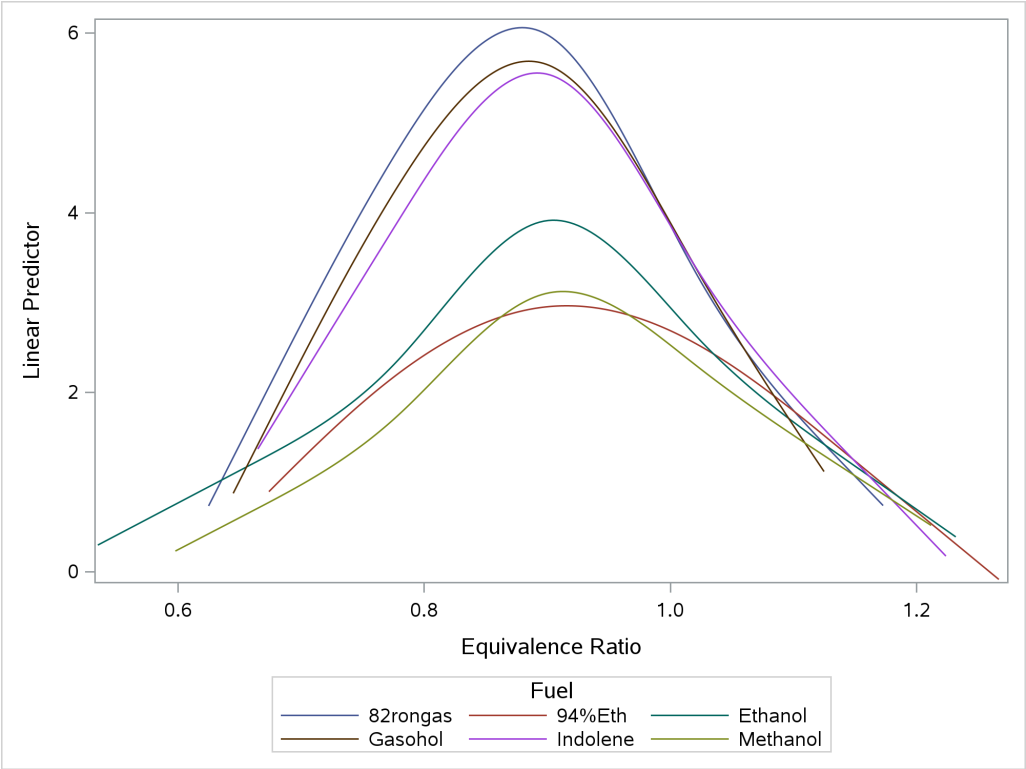

The following steps score the same observations by using the PLM procedure:

```
proc glimmix data=sashelp.gas;
   effect spl = spline(eqratio / naturalcubic knotmethod=equal(5));
   class fuel;
   model nox = spl | fuel;
   store SplineModel;
run;

proc plm restore=SplineModel;
   score data=score out=scored2 predicted=py;
run;

proc sgplot data=scored2;
   series y=py x=eqratio / group=fuel;
run;
```

The scores are displayed in Output 24.6.31. For more information about PROC PLM, see Chapter 94, "The PLM Procedure."

Output 24.6.31 Observations Scored by PROC PLM

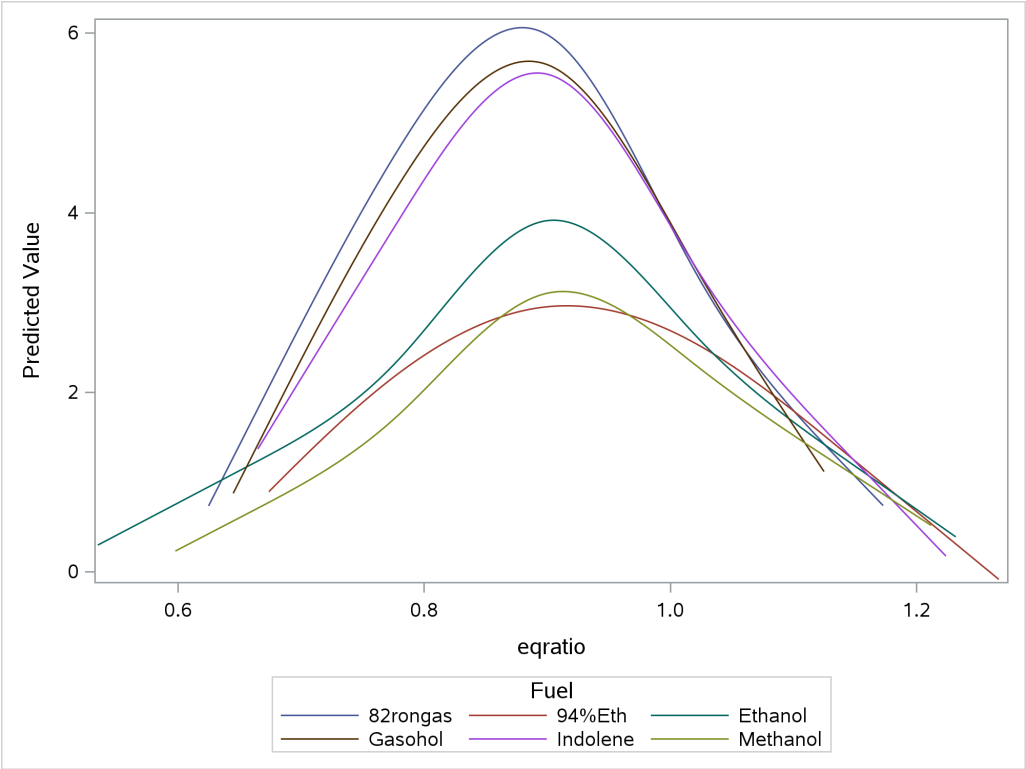

Nonlinear Models

The preceding examples all show linear models. The EFFECT statement and the SPLINE effect are available for both linear and nonlinear models. The following procedures provide an EFFECT statement: GLIMMIX, GLMSELECT, HPMIXED, LOGISTIC, ORTHOREG, PHREG, PLM, PROBIT, QUANTLIFE, QUANTREG, QUANTSELECT, ROBUSTREG, SURVEYLOGISTIC, and SURVEYREG. The following steps show how to use the EFFECT statement in logistic regression:

```
data Neuralgia;
   input Treatment $ Sex $ Age Duration Pain $ @@;
   datalines;
P  F  68   1  No   B  M  74  16  No   P  F  67  30  No   P  M  66  26  Yes
B  F  67  28  No   B  F  77  16  No   A  F  71  12  No   B  F  72  50  No
B  F  76   9  Yes  A  M  71  17  Yes  A  F  63  27  No   A  F  69  18  Yes
B  F  66  12  No   A  M  62  42  No   P  F  64   1  Yes  A  F  64  17  No
P  M  74   4  No   A  F  72  25  No   P  M  70   1  Yes  B  M  66  19  No
B  M  59  29  No   A  F  64  30  No   A  M  70  28  No   A  M  69   1  No
B  F  78   1  No   P  M  83   1  Yes  B  F  69  42  No   B  M  75  30  Yes
P  M  77  29  Yes  P  F  79  20  Yes  A  M  70  12  No   A  F  69  12  No
B  F  65  14  No   B  M  70   1  No   B  M  67  23  No   A  M  76  25  Yes
P  M  78  12  Yes  B  M  77   1  Yes  B  F  69  24  No   P  M  66   4  Yes
P  F  65  29  No   P  M  60  26  Yes  A  M  78  15  Yes  B  M  75  21  Yes
A  F  67  11  No   P  F  72  27  No   P  F  70  13  Yes  A  M  75   6  Yes
B  F  65   7  No   P  F  68  27  Yes  P  M  68  11  Yes  P  M  67  17  Yes
B  M  70  22  No   A  M  65  15  No   P  F  67   1  Yes  A  M  67  10  No
```

```
P  F  72  11  Yes  A  F  74   1  No   B  M  80  21  Yes  A  F  69   3  No
;

proc logistic data=Neuralgia outdesign=coded;
   class Treatment Sex;
   effect Age2      = spline(age / degree=2 knotmethod=equal(0));
   effect Duration2 = polynomial(duration / degree=2);
   model Pain= Treatment Sex Treatment*Sex Age2 Duration2 / expb;
run;

proc print data=coded(obs=10); run;
```

The parameter estimates are displayed in Output 24.6.32. There are three rows for the variable Age, which is designated as a quadratic spline with no knots (in other words, as a quadratic polynomial). There are three rows and two parameter estimates because the B-spline basis had $d + 1$ columns for a polynomial of degree d. The variable Duration is directly specified as a quadratic polynomial, and it has two rows in the table. Output 24.6.33 displays the first 10 coded observations. You can see that Age has values between 0 and 1, which is characteristic of a B-spline basis, and Duration is coded as an ordinary polynomial (duration and duration squared).

Output 24.6.32 Grouped Fit Function, Fit Statistics, and Plot

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq	Exp(Est)
Intercept		1	-22.1944	11.0096	4.0639	0.0438	0.000
Treatment	A	1	0.6601	0.6187	1.1381	0.2861	1.935
Treatment	B	1	2.0464	0.8893	5.2954	0.0214	7.740
Sex	F	1	0.8436	0.4179	4.0736	0.0436	2.325
Treatment*Sex	A F	1	-0.2576	0.5914	0.1897	0.6632	0.773
Treatment*Sex	B F	1	0.5762	0.7017	0.6743	0.4116	1.779
Age2	1	1	16.8907	6.6152	6.5194	0.0107	21654611
Age2	2	1	29.1886	14.6180	3.9871	0.0459	4.747E12
Age2	3	0	0	.	.	.	.
Duration		1	-0.2274	0.1667	1.8615	0.1724	0.797
Duration^2		1	0.00803	0.00562	2.0431	0.1529	1.008

Output 24.6.33 The First 10 Observations of the OUTDESIGN= Data Set

Obs	Pain	Intercept	TreatmentA	TreatmentB	SexF	TreatmentASexF	TreatmentBSexF	Age21	Age22	Age23	Duration	Duration_2
1	No	1	-1	-1	1	-1	-1	0.19531	0.73437	0.07031	1	1
2	No	1	0	1	-1	0	-1	0.07031	0.73437	0.19531	16	256
3	No	1	-1	-1	1	-1	-1	0.22222	0.72222	0.05556	30	900
4	Yes	1	-1	-1	-1	1	1	0.25087	0.70660	0.04253	26	676
5	No	1	0	1	1	0	1	0.22222	0.72222	0.05556	28	784
6	No	1	0	1	1	0	1	0.03125	0.68750	0.28125	16	256
7	No	1	1	0	1	1	0	0.12500	0.75000	0.12500	12	144
8	No	1	0	1	1	0	1	0.10503	0.74826	0.14670	50	2500
9	Yes	1	0	1	1	0	1	0.04253	0.70660	0.25087	9	81
10	Yes	1	1	0	-1	-1	0	0.12500	0.75000	0.12500	17	289

PROC ADAPTIVEREG

The ADAPTIVEREG procedure fits multivariate adaptive regression splines (Friedman 1991). Multivariate adaptive regression splines extend linear models to analyze nonlinear dependencies and produce parsimonious models that do not overfit the data and thus have good predictive power. This method is a nonparametric regression technique that combines both regression splines and model selection. It constructs spline basis functions in an adaptive way by automatically selecting appropriate knot values for different variables, and it obtains reduced models by applying model selection techniques. The method does not assume parametric model forms and does not require specification of knot values. For more information about PROC ADAPTIVEREG, see Kuhfeld and Cai (2013) and Chapter 28, "The ADAPTIVEREG Procedure." The following step displays the results in Output 24.6.34:

```
proc adaptivereg data=sashelp.gas plots=all details=bases;
   class fuel;
   model nox = eqratio | fuel;
run;
```

Output 24.6.34 Grouped Fit Function, Fit Statistics, and Plot

Basis Information	
Name	**Transformation**
Basis0	1
Basis1	Basis0*MAX(EqRatio - 0.915,0)
Basis2	Basis0*MAX(0.915 - EqRatio,0)
Basis3	Basis0*(Fuel = 'Indolene' OR Fuel = '82rongas' OR Fuel = 'Gasohol' OR Fuel = 'Ethanol')
Basis4	Basis0*NOT(Fuel = 'Indolene' OR Fuel = '82rongas' OR Fuel = 'Gasohol' OR Fuel = 'Ethanol')
Basis5	Basis0*(Fuel = 'Ethanol')
Basis6	Basis0*NOT(Fuel = 'Ethanol')
Basis7	Basis6*MAX(EqRatio - 0.808,0)
Basis8	Basis6*MAX(0.808 - EqRatio,0)
Basis9	Basis4*MAX(EqRatio - 0.827,0)
Basis10	Basis4*MAX(0.827 - EqRatio,0)
Basis11	Basis3*MAX(EqRatio - 1.144,0)
Basis12	Basis3*MAX(1.144 - EqRatio,0)
Basis13	Basis0*MAX(EqRatio - 0.954,0)
Basis14	Basis0*MAX(0.954 - EqRatio,0)
Basis15	Basis6*MAX(EqRatio - 1.128,0)
Basis16	Basis6*MAX(1.128 - EqRatio,0)
Basis17	Basis3*MAX(EqRatio - 0.693,0)
Basis18	Basis3*MAX(0.693 - EqRatio,0)
Basis19	Basis0*MAX(EqRatio - 0.846,0)
Basis20	Basis0*MAX(0.846 - EqRatio,0)

Output 24.6.34 *continued*

Regression Spline Model after Backward Selection					
Name	**Coefficient**	**Parent**	**Variable**	**Knot**	**Levels**
Basis0	2.8148		Intercept		
Basis2	-5.0396	Basis0	EqRatio	0.9150	
Basis3	1.8016	Basis0	Fuel		4 0 3 2
Basis5	-2.2310	Basis0	Fuel		2
Basis7	-7.2268	Basis6	EqRatio	0.8080	
Basis8	-13.5265	Basis6	EqRatio	0.8080	
Basis9	19.3139	Basis4	EqRatio	0.8270	
Basis10	7.5643	Basis4	EqRatio	0.8270	
Basis11	13.6667	Basis3	EqRatio	1.1440	
Basis13	-17.0561	Basis0	EqRatio	0.9540	
Basis15	7.4962	Basis6	EqRatio	1.1280	
Basis17	8.9758	Basis3	EqRatio	0.6930	
Basis19	-7.8762	Basis0	EqRatio	0.8460	

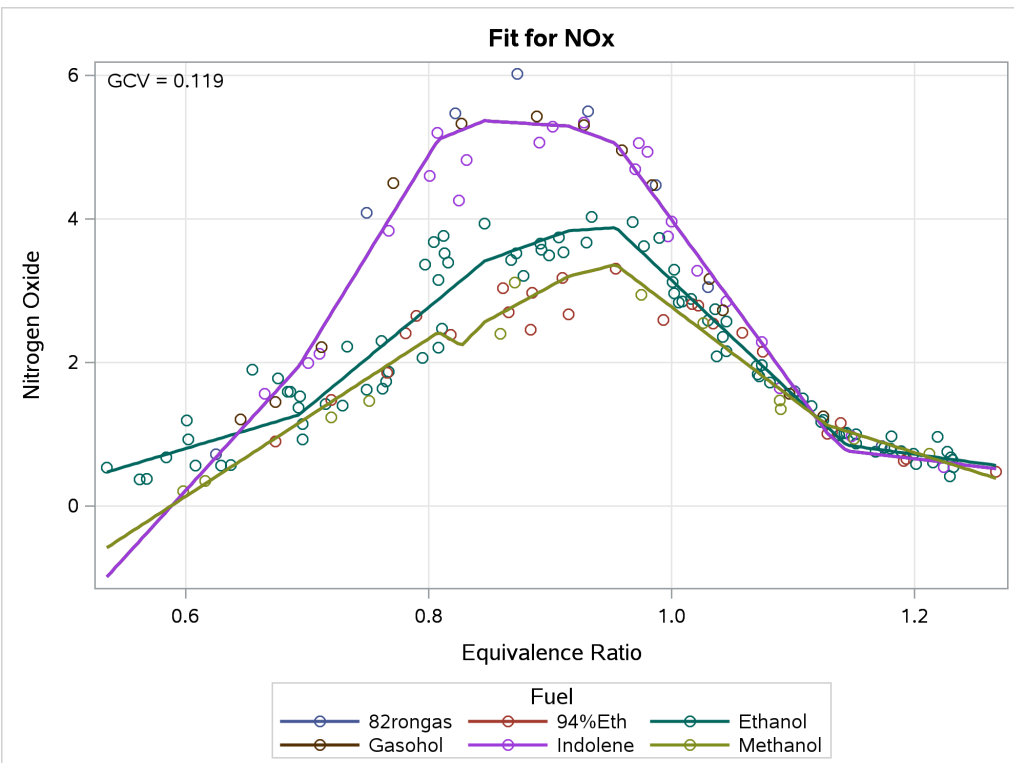

It is obvious from the plot in Output 24.6.34 that this analysis is different from those shown previously. Three splines are displayed even though there are still six types of fuel. Also, the functions are not smooth; they are piecewise linear. The first table shows that the terms that can enter the model include the following:

Basis0 is an intercept.

Basis1 is a linear truncated power function with a knot at 0.915. Like a hockey stick, this term is flat (0) up through 0.915 and then linearly increases as x increases beyond 0.915.

Basis2 is a linear truncated power function with a knot at 0.915. Like a reflection of the preceding hockey stick, this term linearly decreases as x increases to 0.915 and is flat (0) beyond 0.915.
Basis3 is a binary variable that is constructed by combining levels of the CLASS variable.
Basis4 is 0 when Basis3 is 1 and 0 otherwise.
Basis5 is a binary variable that corresponds to the `'Ethanol'` level of the CLASS variable.
Basis6 is 0 when Basis5 is 1 and 0 otherwise.

The remaining terms are interactions of preceding terms and other hockey-stick functions. Forward and backward selection creates a final model, which consists of a subset of the full set of basis functions. A model such as this, which is less smooth and treats groups of fuels the same, is likely to do better in scoring additional observations than a model that has more parameters (as many of the models shown previously have).

For More Information

The following procedures support the EFFECT statement:

GLIMMIX	Chapter 52, "The GLIMMIX Procedure"
GLMSELECT	Chapter 56, "The GLMSELECT Procedure"
HPMIXED	Chapter 62, "The HPMIXED Procedure"
LOGISTIC	Chapter 79, "The LOGISTIC Procedure"
ORTHOREG	Chapter 91, "The ORTHOREG Procedure"
PHREG	Chapter 92, "The PHREG Procedure"
PLM	Chapter 94, "The PLM Procedure"
PROBIT	Chapter 100, "The PROBIT Procedure"
QUANTLIFE	Chapter 102, "The QUANTLIFE Procedure"
QUANTREG	Chapter 103, "The QUANTREG Procedure"
QUANTSELECT	Chapter 104, "The QUANTSELECT Procedure"
ROBUSTREG	Chapter 107, "The ROBUSTREG Procedure"
SURVEYLOGISTIC	Chapter 120, "The SURVEYLOGISTIC Procedure"
SURVEYREG	Chapter 123, "The SURVEYREG Procedure"

The following procedures support splines by means other than the EFFECT statement:

ADAPTIVEREG	Chapter 28, "The ADAPTIVEREG Procedure"
GAM	Chapter 48, "The GAM Procedure"
TPSPLINE	Chapter 125, "The TPSPLINE Procedure"
TRANSREG	Chapter 126, "The TRANSREG Procedure"

For more information, also see the following sections in Chapter 20, "Shared Concepts and Topics":

"Parameterization of Model Effects" on page 412
"EFFECT Statement" on page 427
"Splines and Spline Bases" on page 441
"ODS Graphics: EFFECTPLOT Statement" on page 459

For more information, also see the following sections in Chapter 126, "The TRANSREG Procedure":

"Using Splines and Knots" on page 10632
"Scoring Spline Variables" on page 10643

"Penalized B-Splines" on page 10657
"Smoothing Splines" on page 10660
"SPLINE and MSPLINE Transformations" on page 10693
"SPLINE, BSPLINE, and PSPLINE Comparisons" on page 10696

For more information, also see the conference papers and other resources: Harrell (1988); Cai (2008); Bilenas and Herat (2016); Croxford (2016); Wicklin (2017).

References

Bilenas, J. V., and Herat, N. (2016). "Using Regression Splines in SAS STAT Procedures." In *Proceedings of SESUG 2016 Conference,* Paper BF-140. Cary, NC: SAS Institute Inc. http://analytics.ncsu.edu/sesug/2016/BF-140_Final_PDF.pdf.

Cai, W. (2008). "Fitting Generalized Additive Models with the GAM Procedure in SAS 9.2." In *Proceedings of the SAS Global Forum 2008 Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/forum2008/378-2008.pdf.

Cleveland, W. S., Devlin, S. J., and Grosse, E. (1988). "Regression by Local Fitting." *Journal of Econometrics* 37:87–114.

Croxford, R. (2016). "Restricted Cubic Spline Regression: A Brief Introduction." In *Proceedings of the SAS Global Forum 2016 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings16/5621-2016.pdf.

De Boor, C. (1978). *A Practical Guide to Splines*. New York: Springer-Verlag.

Eilers, P. H. C., and Marx, B. D. (1996). "Flexible Smoothing with *B*-Splines and Penalties." *Statistical Science* 11:89–121. With discussion.

Friedman, J. H. (1991). "Multivariate Adaptive Regression Splines." *Annals of Statistics* 19:1–67.

Harrell, F. E. (1988). "SAS Macros for Assisting with Survival and Risk Analysis, and Some SAS Procedures Useful for Multivariable Modeling."

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer-Verlag.

Kaplan, E. L., and Meier, P. (1958). "Nonparametric Estimation from Incomplete Observations." *Journal of the American Statistical Association* 53:457–481.

Kuhfeld, W. F. (2009). "Modifying ODS Statistical Graphics Templates in SAS 9.2." http://support.sas.com/rnd/app/papers/ods/modtmplt.pdf. Revision of paper by the same title that was published in the *Proceedings of the SAS Global Forum 2009 Conference*.

Kuhfeld, W. F. (2015). *Advanced ODS Graphics Examples*. Cary, NC: SAS Institute Inc. http://support.sas.com/documentation/prod-p/grstat/9.4/en/PDF/odsadvg.pdf.

Kuhfeld, W. F. (2016). *Basic ODS Graphics Examples*. Cary, NC: SAS Institute Inc. http://support.sas.com/documentation/prod-p/grstat/9.4/en/PDF/odsbasicg.pdf.

Kuhfeld, W. F., and Cai, W. (2013). "Introducing the New ADAPTIVEREG Procedure for Adaptive Regression." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings13/457-2013.pdf.

Reinsch, C. H. (1967). "Smoothing by Spline Functions." *Numerische Mathematik* 10:177–183.

Smith, P. L. (1979). "Splines as a Useful and Convenient Statistical Tool." *American Statistician* 33:57–62.

Wicklin, R. (2017). "Regression with Restricted Cubic Splines in SAS." April. https://blogs.sas.com/content/iml/2017/04/19/restricted-cubic-splines-sas.html.

Winsberg, S., and Ramsay, J. O. (1980). "Monotonic Transformations to Additivity Using Splines." *Biometrika* 67:669–674.

Chapter 25
ODS Graphics Template Modification

Contents

Graph Templates **844**
The Graph Template Language 845
Locating Templates 848
Displaying Templates 850
Editing Templates 851
Saving Customized Templates 853
Using Customized Templates 854
Reverting to the Default Templates 854
Graph Template Modification Macro 855
Examples of ODS Graphics Template Modification **860**
Example 25.1: Customizing Graphs Through Template Changes 860
Modifying Graph Titles and Axis Labels 860
Modifying Colors, Line Styles, and Markers 865
Modifying Tick Marks and Grid Lines 866
Modifying the Style to Show Grid Lines 868
Example 25.2: Adding Equations and Special Characters to Fit Plots 870
Simple Linear Regression 870
Cubic Fit Function 876
Unicode and Special Characters 878
Example 25.3: Customizing Panels 884
Example 25.4: Customizing Axes and Reference Lines 888
Example 25.5: Adding Text to Every Graph 896
Adding a Date and Project Stamp to a Few Graphs 898
Adding Data Set Information to a Graph 901
Adding a Date and Project Stamp to All Graphs 902
Example 25.6: PROC TEMPLATE Statement Order and Primary Plots 903
Example 25.7: Marginal Model Plots 908
%Marginal Macro Options 910
%Marginal Macro Examples 911
%Marginal Macro 917
Understanding Conditional Template Logic 922
Example 25.8: Changing Titles by Using the %GrTitle Macro 927
PROC LOGISTIC Example 927
%GrTitle Macro Details 929
References **933**

Graph Templates

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

This chapter discusses the graph template language and graph template modification in ODS Graphics. Be sure that you are familiar with Chapter 24, "Statistical Graphics Using ODS," before reading this chapter.

Graph templates control the layout and details of graphs produced with ODS Graphics. The SAS System provides a template for every graph produced by statistical procedures. Graph template definitions are written in the Graph Template Language (GTL). This powerful language includes statements for specifying plot layouts (such as lattices or overlays), plot types (such as scatter plots and histograms), and text elements (such as titles, footnotes, and insets). It also provides support for built-in computations (such as histogram binning) and the evaluation of expressions. Options are available for specifying colors, marker symbols, and other attributes of plot features.

Graphs, like all SAS output, are constructed from two underlying components, a data component (or data object) and a template. Procedures supply a table of data values and statistical results to plot. Together, the data object and the template form an output object that ODS displays in one or more output destinations. You can control this display by using the GTL to modify the template. With just a little knowledge of the GTL, you can modify or edit templates, even when you do not understand most of the syntax used in the template definition. See examples starting with Example 25.1.

NOTE: You do not need to know anything about the GTL to create statistical graphics.

This section provides an overview of the Graph Template Language. It also describes how to locate, display, edit, and save templates. A *template definition* is a set of SAS statements that is used together with PROC TEMPLATE to create a compiled template. In addition to graph templates, two other common types of templates are table templates and style templates. A table template describes how to display the output for an output object that is rendered as a table. A style template provides formatting information for visual aspects of your SAS output, including both tables and graphs. In most applications, you do not have to modify the templates that SAS supplies. However, when customization is necessary, you can modify the default template with the template language and PROC TEMPLATE.

Compiled templates are stored in a template store, which is a type of item store. (An item store is a special type of SAS file.) The default templates that SAS supplies are stored in template store in the Sashelp library. If you are using the SAS windowing environment, an easy way to display, edit, and save your templates is by using the Templates window. For an introduction to the graph template language, see Kuhfeld (2015, 2016).

For detailed information about managing templates, see the *SAS Output Delivery System: User's Guide* and the *SAS Graph Template Language: User's Guide*. For more information about the syntax of the graph template language, see the *SAS Graph Template Language: Reference*.

The Graph Template Language

(View the complete code for this example (templgs1.sas) in the example repository.)

Graph template definitions begin with a DEFINE STATGRAPH statement in PROC TEMPLATE, and they end with an END statement. Embedded in every graph template is a BEGINGRAPH/ENDGRAPH block, and embedded in that block are one or more LAYOUT blocks. You can specify the DYNAMIC statement to define dynamic variables (which the procedure uses to pass values to the template definition), the MVAR and NMVAR statements to define macro variables (which you can use to pass values to the template definition), and the NOTES statement to provide descriptive information about the graph. The default templates that SAS supplies for statistical procedures are often lengthy and complex, because they provide ODS Graphics with comprehensive and detailed information about graph construction. Here is one of the simpler graph templates for a statistical procedure:

```
define statgraph Stat.MDS.Graphics.Fit;
   notes "MDS Fit Plot";
   dynamic head _byline_ _bytitle_ _byfootnote_;
   begingraph / designwidth=defaultdesignheight;
      entrytitle HEAD;
      layout overlayequated / equatetype=square;
         scatterplot y=FITDATA x=FITDIST / markerattrs=(size=5px);
         lineparm slope=1 x=0 y=0 / clip=true extend=true
            lineattrs=GRAPHREFERENCE;
      endlayout;
      if (_BYTITLE_)
         entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
      else
         if (_BYFOOTNOTE_)
            entryfootnote halign=left _BYLINE_;
         endif;
      endif;
   endgraph;
end;
```

This template, which is supplied for the MDS procedure, creates a scatter plot of two variables, FitData and FitDist, along with a diagonal reference line that passes through the origin. The plot is square and the axes are equated so that a centimeter on one axis represents the same data range as a centimeter on the other axis. The plot title is provided by the evaluation of the dynamic variable Head, which is set by PROC MDS. It is not unusual for this plot to contain hundreds or even thousands of points, so a five-pixel marker is specified, which is smaller than the seven-pixel marker that most styles use by default.

The IF/ELSE construction is common to most templates and enables ODS Graphics to optionally display a BY line with BY-group information as either a footnote or a second title line.

The statements available in the graph template language can be classified as follows:

- Control statements specify the conditional or iterative flow of control. By default, flow of control is sequential. In other words, each statement is used in the order in which it appears.
- Layout statements specify the arrangement of the components of the graph. Layout statements are arranged in blocks that begin with a LAYOUT statement and end with an ENDLAYOUT statement.

The blocks can be nested. Within a layout block, there can be plot, text, and other statements that define one or more graph components. Options provide control for attributes of layouts and components.

- Plot statements specify a number of commonly used displays, including scatter plots, histograms, contour plots, surface plots, and box plots. Plot statements are always provided within a layout block. The plot statements include options to specify the data columns from the data object that is used in the graph. For example, in the SCATTERPLOT statement, there are mandatory X= and Y= arguments that specify which data columns are used for the X (horizontal) and Y (vertical) axes in the plot. (In the preceding example, FitData and FitDist are the names of columns in the data object that PROC MDS creates for this graph.) There is also a GROUP= option that specifies a data column as an optional classification variable.

- Text statements specify the descriptions that accompany graphs. An entry is any textual description, including titles, footnotes, and legends; it can include symbols to identify graph elements.

The following statements display another of the simpler template definitions—the definition of the scatter plot available in PROC KDE (see Figure 73.6.1 in Chapter 73, "The KDE Procedure"):

```
proc template;
   define statgraph Stat.KDE.Graphics.ScatterPlot;
      dynamic _VAR1NAME _VAR1LABEL _VAR2NAME _VAR2LABEL
              _byline_ _bytitle_ _byfootnote_;
      BeginGraph;
         EntryTitle "Distribution of " _VAR1NAME " by " _VAR2NAME;
         Layout Overlay / xaxisopts=(offsetmin=0.05 offsetmax=0.05)
                          yaxisopts=(offsetmin=0.05 offsetmax=0.05);
            ScatterPlot x=X y=Y / markerattrs=GRAPHDATADEFAULT;
         EndLayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
      EndGraph;
   end;
run;
```

Here, the PROC TEMPLATE and RUN statements have been added to show how to compile the template if you want to modify it. The DEFINE STATGRAPH statement in PROC TEMPLATE begins the graph template definition, and the END statement ends the definition. The DYNAMIC statement defines four dynamic variables that PROC KDE sets at run time. Three additional DYNAMIC variables control the BY line. The dynamic variables _Var1Name and _Var2Name provide the variable names for the title of the graph. The dynamic variables _Var1Label and _Var2Label contain the labels of the X and Y variables, respectively. The title of the graph is specified by the ENTRYTITLE statement. You can change the ENTRYTITLE statement to display variable labels instead of variable names in the title. You can also use any of the dynamic variables elsewhere in the graph template.

The axes are controlled by the LAYOUT OVERLAY statement inside the BEGINGRAPH/ENDGRAPH statement block. The offset options add a small amount of padding between the axes and the most extreme points. The SCATTERPLOT statement creates the scatter plot. The options in the SCATTERPLOT statement

are specified after the slash and control the marker attributes (symbol, color, and size). You can specify these attributes directly, as in the PROC MDS template, or more typically through style elements, as in the PROC KDE template. Style elements are defined in ODS styles, and elements can vary across different styles. For more information about ODS styles, see the section "ODS Styles" on page 717 in Chapter 24, "Statistical Graphics Using ODS." The ENDLAYOUT statement ends the main layout block. For information about the syntax of the graph template language, see the *SAS Graph Template Language: Reference*.

You can write your own templates and use them to display raw data or output from procedures. For example, consider the iris data from Example 42.1 of Chapter 42, "The DISCRIM Procedure." The iris data set is available from the Sashelp library.

The following statements create a template for a scatter plot of the variables PetalLength and PetalWidth with a legend:

```
proc template;
   define statgraph scatter;
      begingraph;
         entrytitle 'Fisher (1936) Iris Data';
         layout overlayequated / equatetype=fit;
            scatterplot x=petallength y=petalwidth /
                        group=species name='iris';
            layout gridded / autoalign=(topleft);
               discretelegend 'iris' / border=false opaque=false;
            endlayout;
         endlayout;
      endgraph;
   end;
run;
```

The layout is OVERLAYEQUATED, which equates the axes in the plot. However, unlike the PROC MDS template, which used EQUATETYPE=SQUARE to make a square plot, the EQUATETYPE=FIT option specifies that the lengths of the axes in this plot should fill the entire plotting area. A legend is placed internally in the top-left portion of the plot. There are three groups of observations, indicated by the three species, and each group is plotted with a separate color and symbol that depends on the ODS style. The legend identifies each group. The NAME= option provides the link between the SCATTERPLOT statement and the DISCRETELEGEND statement. An explicit link is needed since some graphical displays are based on multiple plotting statements.

The following step creates the plot by using the SGRENDER procedure, the Sashelp.Iris data set, and the custom template **scatter**:

```
proc sgrender data=sashelp.iris template=scatter;
run;
```

The syntax of PROC SGRENDER is very simple, because all of the graphical options appear in the template. The scatter plot in Figure 25.1 shows the results.

The intent of this example is to illustrate how you can write a template to create a scatterplot. PROC TEMPLATE and PROC SGRENDER provide you with the power to create highly customized displays. However, usually you can use the SGPLOT, SGSCATTER or SGPANEL procedures instead, which are much simpler to use. These procedures are discussed in the section "Statistical Graphics Procedures" on page 775 in Chapter 24, "Statistical Graphics Using ODS." See the section "Grouped Scatter Plot with PROC SGPLOT" on page 684 and Figure 24.12 in Chapter 24, "Statistical Graphics Using ODS," for an example

that plots these data with PROC SGPLOT.

Figure 25.1 Petal Width and Petal Length in Three Iris Species

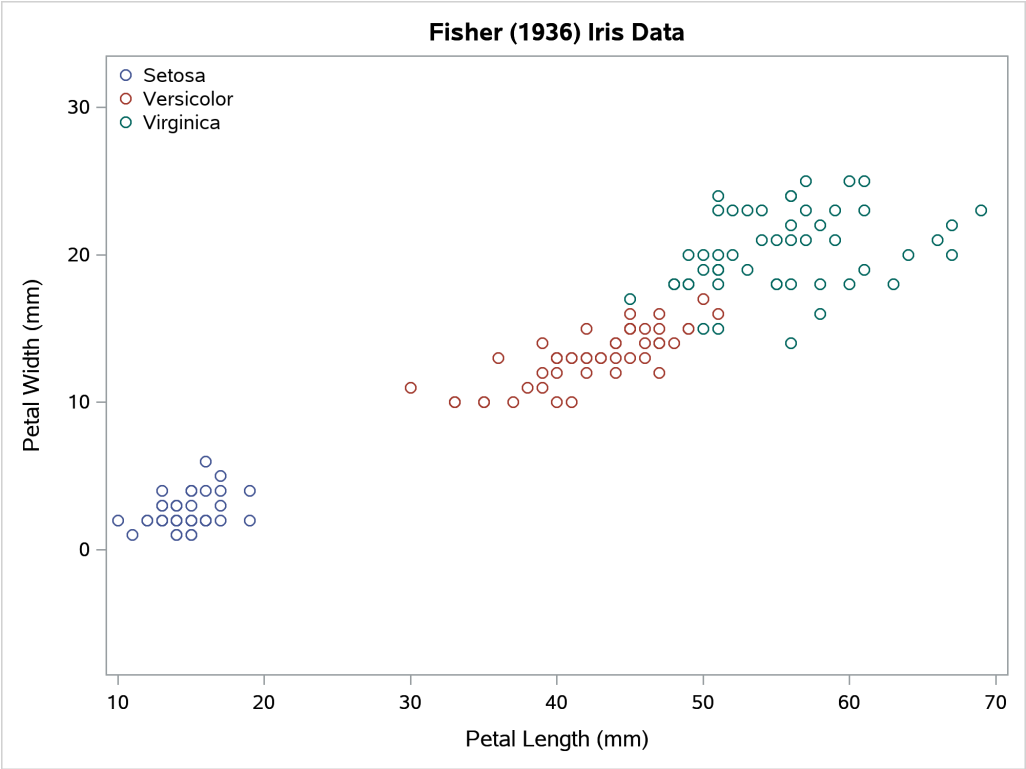

Locating Templates

(View the complete code for this example (templgs1.sas) in the example repository.)

Before you can customize a graph, you must determine which template is used to create the original graph. You can do this by submitting the ODS TRACE ON statement before the procedure statements that create the graph. The template name is displayed in the SAS log. Here is an example:

```
ods trace on;
ods graphics on;

proc reg data=sashelp.class;
   model Weight = Height;
quit;
```

The preceding statements create the following trace output, which provides information about both the graphs and tables produced by PROC REG:

```
Output Added:
-------------
Name:       NObs
Label:      Number of Observations
Template:   Stat.Reg.NObs
```

```
Path:       Reg.MODEL1.Fit.Weight.NObs
-------------

Output Added:
-------------
Name:       ANOVA
Label:      Analysis of Variance
Template:   Stat.REG.ANOVA
Path:       Reg.MODEL1.Fit.Weight.ANOVA
-------------

Output Added:
-------------
Name:       FitStatistics
Label:      Fit Statistics
Template:   Stat.REG.FitStatistics
Path:       Reg.MODEL1.Fit.Weight.FitStatistics
-------------

Output Added:
-------------
Name:       ParameterEstimates
Label:      Parameter Estimates
Template:   Stat.REG.ParameterEstimates
Path:       Reg.MODEL1.Fit.Weight.ParameterEstimates
-------------

Output Added:
-------------
Name:       DiagnosticsPanel
Label:      Fit Diagnostics
Template:   Stat.REG.Graphics.DiagnosticsPanel
Path:       Reg.MODEL1.ObswiseStats.Weight.DiagnosticPlots.DiagnosticsPanel
-------------

Output Added:
-------------
Name:       ResidualPlot
Label:      Height
Template:   Stat.REG.Graphics.ResidualPlot
Path:       Reg.MODEL1.ObswiseStats.Weight.ResidualPlots.ResidualPlot
-------------

Output Added:
-------------
Name:       FitPlot
Label:      Fit Plot
Template:   Stat.REG.Graphics.Fit
Path:       Reg.MODEL1.ObswiseStats.Weight.FitPlot
-------------
```

This is also illustrated in Example 25.1 and the section "The ODS Statement" on page 599 in Chapter 23, "Using the Output Delivery System."

Displaying Templates

(View the complete code for this example (templgs1.sas) in the example repository.)

Once you have found the name of a template, you can display its definition (source program) by using one of these methods:

- Open the Templates window by issuing the command **odstemplates** (**odst** for short) in the command line of the SAS windowing environment. The template window is shown in Figure 25.2. If you expand the Sashelp.Tmplstat node, you can view all the available templates and double-click any template icon to display its definition. This is illustrated in Example 25.1.
- Use the SOURCE statement in PROC TEMPLATE to display a template definition in the SAS log or write the definition to a file.

Figure 25.2 Requesting the Templates Window in the Command Line

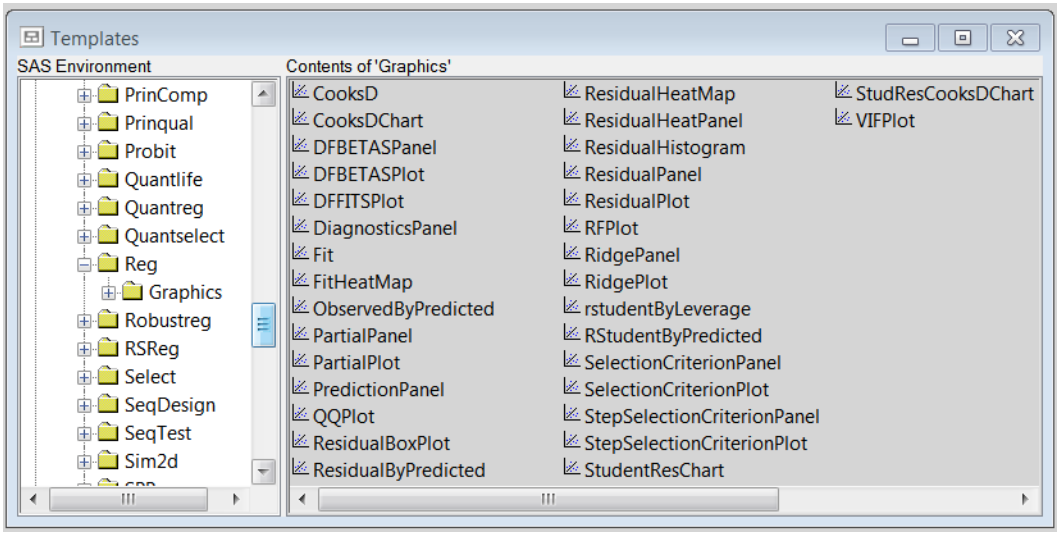

For example, the following statements display the template for the PROC REG residual plot:

```
proc template;
   source Stat.REG.Graphics.ResidualPlot;
run;
```

The template is displayed as follows:

```
define statgraph Stat.Reg.Graphics.ResidualPlot;
   notes "Residual Plot";
   dynamic _XVAR _SHORTXLABEL _TITLE _LOESSLABEL _DEPNAME _MODELLABEL _SMOOTH
      _byline_ _bytitle_ _byfootnote_;
   BeginGraph;
      entrytitle halign=left textattrs=GRAPHVALUETEXT _MODELLABEL halign=
         center textattrs=GRAPHTITLETEXT _TITLE " for " _DEPNAME;
      entrytitle textattrs=GRAPHVALUETEXT _LOESSLABEL;
      layout overlay / xaxisopts=(shortlabel=_SHORTXLABEL);
```

```
            referenceline y=0;
            scatterplot y=RESIDUAL x=_XVAR / primary=true rolename=(_tip1=
               OBSERVATION _id1=ID1 _id2=ID2 _id3=ID3 _id4=ID4 _id5=ID5) tip=(y
               x _tip1 _id1 _id2 _id3 _id4 _id5);
            if (EXISTS(_SMOOTH))
               loessplot y=_SMOOTH x=_XVAR / tiplabel=(y="Smoothed Residual");
            endif;
         endlayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
      EndGraph;
   end;
```

PROC TEMPLATE also tells you where the template is located. In this case, it prints the following note:

```
NOTE: Path 'Stat.Reg.Graphics.ResidualPlot' is in: SASHELP.TMPLSTAT
      (via SASHELP.TMPLMST).
```

The word “Path” in ODS has several meanings. In this context, it refers to any name or label hierarchy. In the note, the levels of the template name form a path. In the trace output, the levels of the plot name form a different path.

Editing Templates

You can modify the format and appearance of a particular graph by doing the following:

- Modify its template definition (source program).
- Submit the revised template to create a new compiled template, which is stored in a template store.
- Ensure that the ODS template search path includes the template store that contains your new template.

Template stores are designated read-only (such as Sashelp.Tmplmst) or updatable (such as Sasuser.Templat).

If you view the templates in an updatable template store from the Templates window, you can select **Open** or **Edit** from the pop-up menu. Either the Template Browser or Template Editor window opens. In the Template Editor window, you can make changes and submit the code directly. For read-only templates or when you select **Open**, the Template Browser window opens and you must copy the definition to an editor window to make changes. Since templates that SAS supplies are in the read-only Sashelp library, an easy way to obtain an editable program file is to use the SOURCE statement with the FILE= option in PROC TEMPLATE to write the template definition to a file as follows:

```
proc template;
   source Stat.REG.Graphics.ResidualPlot / file="residtpl.sas";
run;
```

By default, the file is saved in the SAS current folder. Alternatively, you can omit the slash and the FILE= option and copy and paste the source from the SAS log into an editor. Either way, you must add a PROC TEMPLATE statement before the generated source statements and optionally a RUN statement after the END statement before you submit your modified definition.

Graph templates are self-contained and do not support inheritance (via the PARENT= option) as do table templates. Consequently, the EDIT statement in PROC TEMPLATE is not supported for graph templates.

Here are some important points about what you can and cannot change in a template that SAS supplies while preserving its overall functionality:

- Do not change the template name. A statistical procedure can access only a predefined list of templates. If you change the name, the procedure cannot find your template. You must keep the original name and make sure that it is in a template store that is searched before Sashelp.Tmplmst. You control this with the ODS PATH statement. For more information about the template search path and the ODS PATH statement, see the section "The Default Template Stores and the Template Search Path" on page 715 in Chapter 24, "Statistical Graphics Using ODS," the section "Saving Customized Templates" on page 853, and subsequent sections.
- Do not change the names of columns. The underlying data object contains predefined column names that you must use. Be very careful if you change how a column is used in a template. Usually, columns are not interchangeable.
- Do not change the names of DYNAMIC variables. Procedures set values only for a predefined list of dynamic variables. Changing dynamic variable names can lead to runtime errors. Do not add dynamic variables, because the procedure cannot set their values. A few procedures document additional dynamic variables that can be defined in the template if you want to add more information to the output, such additional statistics in an inset table.
- Do not change the names of statements (for example, from a SCATTERPLOT to a NEEDLEPLOT or other type of plot).

You can change any of the following:

- You can add macro variables that behave like dynamic variables. They are resolved at the time that the statistical procedure is run, and not at the time that the template is compiled. They are defined with an MVAR or NMVAR statement at the beginning the template. You can set the value of each macro variable with a %LET statement before the statistical procedure is run. See Example 25.5. You can also move a variable from a DYNAMIC statement to an MVAR or NMVAR statement if you want to set it yourself rather than letting the procedure set it.
- You can change the graph size.
- You can change graph titles, footnotes, axis labels, and any other text that appears in the graph.
- You can change which plot features are displayed.
- You can change axis features, such as grid lines, offsets, view ports, tick value formatting, and so on.
- You can change the content and arrangement of insets (small tables of statistics embedded in some graphs).

- You can change the legend location, contents, border, background, title, and so on.

See the *SAS Graph Template Language: Reference* for information about the syntax of the statements in the Graph Template Language.

Saving Customized Templates

After you edit the template definition, you can submit your PROC TEMPLATE statements as you would any other SAS program. If you are using the Template Editor window, select **Submit** from the **Run** menu. See Example 25.1. Alternatively, submit your PROC TEMPLATE statements from the Program Editor. ODS automatically saves the compiled template in the first template store that it can update, according to the currently defined template search path. If you have not changed the template search path, then the modified template is saved in the Sasuser.Templat template store. You can use the following statement to display the current template search path:

```
ods path show;
```

The log messages for the default template search path are as follows:

```
Current ODS PATH list is:

1. SASUSER.TEMPLAT(UPDATE)
2. SASHELP.TMPLMST(READ)
```

The ODS PATH statement template search path name Sashelp.Tmplmst refers to the template stores that are shipped with the SAS System. More precisely, the ODS PATH name Sashelp.Tmplmst refers to multiple template store files, including Sashelp.Tmplmst, Sashelp.Tmplstat, Sashelp.Tmplets, Sashelp.Tmplqc, Sashelp.Tmpliml, and others. The name Sashelp.Tmplmst refers to both the entire template store that is shipped with the SAS System and a particular file in that template store. Earlier releases of the SAS System had just one template store file, Sashelp.Tmplmst, and there was a one-to-one correspondence between the ODS PATH statement name and the template store file name. Now there are multiple files (because certain products have their own template store files), but the ODS PATH statement syntax for selecting all of them is unchanged. You do not need to be concerned about this, and you should not specify any of these template stores individually. You simply specify Sashelp.Tmplmst to get the item stores that are shipped with the SAS System. However, you will sometimes see these other names in the SAS log and output when you are working with templates, so you need to know what that means.

If you want to store modified templates in another template store, you can use the ODS PATH statement to add that template store to the front of the list. To use these templates, you must make sure the template search path is set correctly before you attempt to access them in the other SAS sessions. See the section "Using Customized Templates" on page 854.

Using Customized Templates

When you create ODS output (either graphs or tables), ODS searches sequentially through each template store in the template search path for a template that matches the one requested. If you have not changed the default template search path, then ODS searches the Sasuser.Templat store first, then Sashelp.Tmplmst. ODS uses the first template that it finds with the requested name. **NOTE:** Templates with the same name can exist in more than one template store.

The ODS PATH statement specifies the template stores to search, as well as the order in which to search them. You can change the default template search path by using the ODS PATH statement. For example, the following statement sets the template search path so that the template store Work.Mystore is searched first, followed by Sashelp.Tmplmst:

```
ods path work.mystore(update) sashelp.tmplmst(read);
```

The UPDATE option provides both update access and read access to Work.Mystore. The READ option provides read-only access to Sashelp.Tmplmst. With this path, the template store Sasuser.Templat is no longer searched. You can verify this by using the following statement:

```
ods path show;
```

The log messages that the preceding statement generates are as follows:

```
Current ODS PATH list is:

1. WORK.MYSTORE(UPDATE)
2. SASHELP.TMPLMST(READ)
```

For more information, see the *SAS Output Delivery System: User's Guide* and the *SAS Graph Template Language: User's Guide*. Example 25.1 illustrates all the steps of displaying, editing, saving, and using customized templates.

Reverting to the Default Templates

(View the complete code for this example (templgs1.sas) in the example repository.)

Customized templates are stored in Sasuser.Templat or in some other template store that you create. The templates that SAS supplies are in the read-only template store Sashelp.Tmplmst. If you modify any of the supplied templates and you want to use the original default templates, you can change your template search path as follows:

```
ods path sashelp.tmplmst(read) sasuser.templat(update);
```

This way the default templates are found first. Alternatively, you can save all your customized templates in a user-defined template store (for example Mylib.Mystore). To access these templates, you submit the following statement before running your analysis:

```
libname mylib '.';
ods path mylib.mystore(update) sashelp.tmplmst(read);
```

When you are done, you can reset the default template search path as follows:

```
ods path reset;
```

This restores the template search path to its original state (**sasuser.templat(update) sashelp.tmplmst(read)**). You can also save your customized template as part of your SAS program. When you are done, you can delete your customized template from the Sasuser.Templat template store, as in the following statements:

```
proc template;
   delete Stat.REG.Graphics.ResidualPlot / store=sasuser.templat;
run;
```

The option STORE=SASUSER.TEMPLAT is not required. However, if you have administrator privileges on your computer, this option helps you ensure that you do not accidentally delete templates from Sashelp.Tmplmst.

The following note is printed in the SAS log:

```
NOTE: 'Stat.REG.Graphics.ResidualPlot' has been deleted from: SASUSER.TEMPLAT
```

You can run the following step to delete the entire Sasuser.Templat store of customized templates:

```
ods path sashelp.tmplmst(read);
proc datasets library=sasuser nolist;
   delete templat(memtype=itemstor);
run;
ods path sasuser.templat(update) sashelp.tmplmst(read);
```

Graph Template Modification Macro

(View the complete code for this example (templgs1.sas) in the example repository.)

You can use the **%ModTmplt** autocall macro to insert BY line information, titles, and footnotes in ODS Graphics. You can also use it to remove titles and perform other template modifications. For more information about this macro, see Kuhfeld (2009).

You do not have to include autocall macros (for example, with a **%include** statement). You can call them directly once they are properly installed. If your site has installed the autocall libraries that SAS supplies and uses the standard configuration of SAS supplied software, you need to ensure that the SAS system option MAUTOSOURCE is in effect to begin using the autocall macros. For more information about autocall libraries, see the *SAS Macro Language: Reference*. For more information about installing autocall macros, consult your host documentation.

The **%ModTmplt** macro has the following options:

BY=*by-variable-list*
specifies the list of BY variables. Also see BYLIST=. When graphs are produced (by default or when the STEPS= value contains 'G'), you must specify the BY= option. Otherwise, when you are only modifying the template, you do not need to specify the BY= option. **NOTE:** This option has been rendered obsolete by the BYLINE= option in the ODS GRAPHICS statement.

BYLIST=*by-statement-list*
specifies the full syntax of the BY statement. You can specify a full BY statement syntax including the DESCENDING or NOTSORTED options. If only BY variables are needed, specify only BY=. If you also need options, then specify the BY variables in the BY= option and the full syntax in the BYLIST= option (for example, specify BY=A B and BYLIST=A DESCENDING B). **NOTE:** This option has been rendered obsolete by the BYLINE= option in the ODS GRAPHICS statement.

DATA=*SAS-data-set*
specifies the input SAS data set. If you do not specify the DATA= option, the macro uses the most recently created SAS data set.

FILE=*filename*
specifies the file in which to store the original templates. This is a temporary file. You can specify either a quoted file name or the name from a FILENAME statement that you provide before you call the macro. The default is *"template.txt"*.

OPTIONS=*options*
specifies one or more of the following options (case is ignored):

LOG
displays a note in the SAS log when each BY group has finished.

FIRST
adds the ENTRYTITLE or ENTRYFOOTNOTE statements as the first titles or footnotes. By default, the statements are added after the last titles or footnotes. Most graph templates provided by SAS do not use footnotes; so this option usually affects only entry titles.

NOQUOTES
specifies that the values of the system titles and footnotes are to be moved to the ENTRYTITLE or ENTRYFOOTNOTE statements without the outer quotation marks. With OPTIONS=NOQUOTES, you can specify options in the titles or footnotes in addition to the text. However, you must ensure that you quote the text that provides the actual title or footnote.

The following is an example of an ordinary footnote:

```
footnote "My Footer";
```

With this FOOTNOTE statement and without OPTIONS=NOQUOTES, the macro creates the following ENTRYFOOTNOTE statement:

```
entryfootnote "My Footer";
```

The following footnotes are used with OPTIONS=NOQUOTES:

```
footnote 'halign=left "My Footer"';
footnote2 '"My Second Footer"';
```

With these FOOTNOTE statements and OPTIONS=NOQUOTES, the macro creates the following ENTRYFOOTNOTE statements:

```
entryfootnote halign=left "My Footer";
entryfootnote "My Second Footer";
```

REPLACE

replaces the unconditionally added entry titles and entry footnotes in the templates (those that are not part of IF or ELSE statements) with the system titles and footnotes. The system titles and footnotes are those that are specified in the TITLE or FOOTNOTE statements. You can instead use the TITLES=*SAS-data-set* option to specify titles and footnotes with a data set. If OPTIONS=REPLACE is specified, then OPTIONS=TITLES is ignored.

SOURCE

displays the generated source code. By default, the template source code is not displayed.

TITLES

displays the system titles and footnotes with the graphs. The system titles and footnotes are those that are specified in the TITLE or FOOTNOTE statements. You can instead use the TITLES=*SAS-data-set* option to specify titles and footnotes with a data set. If you also specify OPTIONS=FIRST, the system titles and footnotes are inserted before the previously existing entry titles and entry footnotes in the templates. Otherwise, they are inserted at the end.

You can specify OPTIONS=TITLES or OPTIONS=REPLACE, or insert BY lines, or do both. If you do both, and you do not like where the BY line is inserted relative to your titles and footnotes, just specify OPTIONS=NOQUOTES and _ByLine0 to place the BY line wherever you choose. The following TITLE statements illustrate:

```
title1 '"My First Title"';
title2 '_byline0';
title3 '"My Last Title"';
```

Also, you can embed BY information in a title or a footnote, again with OPTIONS=NOQUOTES. For example:

```
title '"Spline Fit By Sex, " _byline0';
```

When _ByLine0 is specified in any of the titles or footnotes, then the usual BY line is not added.

The following example removes all titles and footnotes:

```
footnote;
title;
%modtmplt(options=replace, template=Stat.Transreg.Graphics, steps=t)
```

STATEMENT=*entry-statement-fragment*

specifies the statement that contains the BY line that gets added to the template along with any statement options. The default is **`Statement=EntryFootNote halign=left TextAttrs=GraphValueText`**. Other examples include:

```
Statement=EntryTitle
Statement=EntryFootNote halign=left TextAttrs=GraphLabelText
```

STEPS=*steps*

specifies the macro steps to run. Case and white space are ignored. the macro modifies the templates (when 'T' is specified), produces the graphs for each BY group (when 'G' is specified), and deletes the modified templates (when 'D' is specified). The default is STEPS=TGD. You can instead have it perform a subset of these three tasks by specifying a subset of terms in the STEPS= option.

When you use the **`%ModTmplt`** macro to add BY lines, you usually do not need to delete the templates before you run your procedure again in the normal way. The template modification inserts the BY line through a macro variable and an MVAR statement. When the macro variable _ByLine0 is undefined, the ENTRYTITLE or ENTRYFOOTNOTE statement drops out as if it were not there at all.

STMTOPTS1= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS2= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS3= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS4= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS5= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS6= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS7= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS8= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS9= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >
STMTOPTS10= *n* **ADD | REPLACE | DELETE | BEFORE | AFTER** *statement-name* < *options* >

These ten options add or replace options in up to 10 selected statements. The following example illustrates:

```
%modtmplt(template=Stat.glm.graphics.residualhistogram, steps=t,
          stmtopts1=.  add  discretelegend  autoalign=(topleft),
          stmtopts2=1  add  densityplot     legendlabel='Normal Density',
          stmtopts3=2  add  densityplot     legendlabel='Kernel Density',
          stmtopts4=1  add  overlay         yaxisopts=(griddisplay=on)
                            yaxisopts=(label='Normal and Kernel Density'))

proc glm plots=diagnostics(unpack) data=sashelp.class;
   model weight = height;
run;

%modtmplt(template=Stat.glm.graphics.residualhistogram, steps=d)
```

These options require you to specify a series of values. The first value is the statement number (or missing to modify options on all statements that match the statement name). The second value is: ADD, REPLACE, DELETE, BEFORE, or AFTER. When the second value is ADD or REPLACE, it controls whether you add new options or replace existing options. Alternatively, the second value can be BEFORE or AFTER to add a new statement before or after the named statement. When the value is DELETE, the corresponding statement is deleted. The third value is a statement name. All remaining options are options for the statement named by the third value (with ADD and REPLACE) or for a new statement (with BEFORE and AFTER). In the STMTOPTS1= example, an option is added to all DISCRETELEGEND statements. In the STMTOPTS2= example, an option is added to the first DensityPlot statement. In the STMTOPTS4= example, an option is added to the LAYOUT OVERLAY statement. In most cases, the statement name is the first name that begins the statement. The LAYOUT statement is an exception. In the case of layouts, specify the second name (OVERLAY, GRIDDED, LATTICE, and so on) for the third value. Note that a statement such as **`if (expression) EntryTitle...;`** is an IF statement not an ENTRYTITLE statement.

If an option is specified multiple times on a GTL statement, the last specification overrides previous specifications. Hence, you do not need to know and respecify all of the options. You can just add an option to the end, and it overrides the previous value. You can use these options only to modify statements that contain a slash, and only to modify the options that come after the slash. Note that in STMTOPTS4=, the YAXISOPTS= option is specified twice. It could have been equivalently specified once as follows:

```
yaxisopts=(griddisplay=on label='Normal and Kernel Density'))
```

The actual specification adds the GRIDDISPLAY=ON to the Y axis options (which by default has only a label specification). The old label is unchanged until the LABEL= option in the second YAXISOPTS= specification overrides it. In other words, YAXISOPTS=(GRIDDISPLAY=ON) augments the old YAXISOPTS= option; it does not replace it.

The following steps delete the legend and instead provide a footnote:

```
%modtmplt(template=Stat.glm.graphics.residualhistogram, steps=t,
          stmtopts1=. delete discretelegend,
          stmtopts2=1 after  begingraph entryfootnote
                      textattrs=GraphLabelText(color=cx445694) 'Normal  '
                      textattrs=GraphLabelText(color=cxA23A2E) 'Kernel')

proc glm plots=diagnostics(unpack) data=sashelp.class;
   model weight = height;
run;

%modtmplt(template=Stat.glm.graphics.residualhistogram, steps=d)
```

TEMPLATE=*SAS-template*

specifies the name of the template to modify. You can specify just the first few levels to modify a series of templates. For example, to modify all of PROC REG's graph templates, specify TEMPLATE=**`Stat.Reg.Graphics`**. This option is required.

TITLES=*SAS-data-set*
specifies a data set that contains titles or footnotes or both. By default, when the system titles or footnotes are used (when OPTIONS=TITLES or OPTIONS=REPLACE is specified), PROC SQL is used to determine the titles and footnotes. You can instead create this data set yourself so that you can set the graph titles independently from the system titles and footnotes. The data set must contain two variables: Type (Type='T' for titles and Type='F' for footnotes), and Text, which contains the titles and footnotes. Other variables are ignored. Specify the titles and footnotes in the order in which you want them to appear.

TITLEOPTS=*entry-statement-options*
specifies the options for system titles and footnotes. For example, you can specify the HALIGN= and TEXTATTRS= options as in the STATEMENT= option. By default, no title options are used. With OPTIONS=NOQUOTES, you can specify options individually.

Examples of ODS Graphics Template Modification

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 25.1: Customizing Graphs Through Template Changes

(View the complete code for this example (templex1.sas) in the example repository.)

This example shows how to use PROC TEMPLATE to customize the appearance and content of an ODS graph. It is divided into several parts; each part illustrates a different aspect of the template that you can easily change. You are never required to change a template, but you can if you want to change aspects of the plot.

Modifying Graph Titles and Axis Labels

This section illustrates the discussion in the section "Graph Templates" on page 844 in the context of changing the default title and Y-axis label for a Q-Q plot created with PROC ROBUSTREG. The data set Stack is created by the following statements:

```
data stack;
   input  x1 x2 x3 y @@;
   datalines;
80  27  89  42    80  27  88  37    75  25  90  37    62  24  87  28
62  22  87  18    62  23  87  18    62  24  93  19    62  24  93  20
58  23  87  15    58  18  80  14    58  18  89  14    58  17  88  13
58  18  82  11    58  19  93  12    50  18  89   8    50  18  86   7
50  19  72   8    50  19  79   8    50  20  80   9    56  20  82  15
70  20  91  15
;
```

The following statements request a Q-Q plot for robust residuals created by PROC ROBUSTREG:

```
ods trace on;
ods graphics on;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;

ods trace off;
```

The Q-Q plot is shown in Output 25.1.1.

Output 25.1.1 Default Q-Q Plot from PROC ROBUSTREG

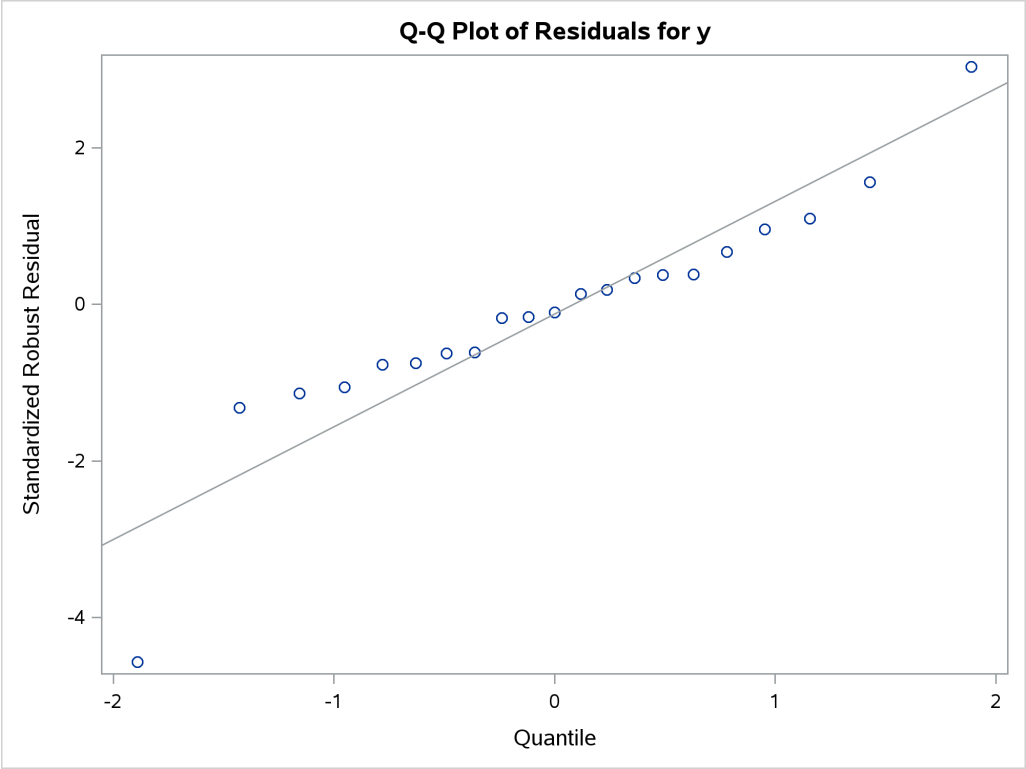

The ODS TRACE ON statement requests a record of all the ODS output objects created by PROC ROBUSTREG. The trace output is as follows:

```
Output Added:
-------------
Name:        QQPlot
Label:       Residual Q-Q Plot
Template:    Stat.Robustreg.Graphics.QQPlot
Path:        Robustreg.DiagnosticPlots.QQPlot
-------------
```

ODS Graphics creates the Q-Q plot from an ODS data object named QQPlot and a graph template named **Stat.Robustreg.Graphics.QQPlot**, which is the default template provided by SAS. Default templates

that SAS supplies are saved in the Sashelp.Tmplmst template store (see the section "Graph Templates" on page 844).

To display the default template definition, open the Templates window by typing **odstemplates** (or **odst** for short) in the command line. Expand Sashelp.Tmplstat and click the **Stat** folder. Next, open the **Robustreg** folder and then open the **Graphics** folder. Then right-click the **QQPlot** template icon and select **Open**. This opens the Template Browser. You can copy this template to an editor to edit it.

Alternatively, you can submit the following statements to display the **QQPlot** template definition in the SAS log:

```
proc template;
   source Stat.Robustreg.Graphics.QQPlot;
run;
```

The SOURCE statement specifies the template name. You can copy and paste the template source into the Program Editor and modify it. The template, with a PROC TEMPLATE and RUN statement added, is shown next:

```
proc template;
   define statgraph Stat.Robustreg.Graphics.QQPlot;
      notes "Q-Q Plot for Standardized Robust Residuals";
      dynamic _DEPLABEL Residual _byline_ _bytitle_ _byfootnote_;
      BeginGraph;
         ENTRYTITLE "Q-Q Plot of Residuals for " _DEPLABEL;
         Layout Overlay / yaxisopts=(label="Standardized Robust Residual")
            xaxisopts=(label="Quantile");
            SCATTERPLOT y=eval (SORT(DROPMISSING(RESIDUAL))) x=eval (
               PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
               N(RESIDUAL)))) / primary=true
               markerattrs=GRAPHDATADEFAULT
               rolename=(q=eval(
               PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
               N(RESIDUAL)))) s=eval (SORT(DROPMISSING(RESIDUAL))))
               tip=(q s) tiplabel=(q= "Quantile" s="Residual");
            lineparm slope=eval (STDDEV(RESIDUAL)) Y=eval (MEAN(RESIDUAL)) X=0 /
               clip=false lineattrs=GRAPHREFERENCE extend=true;
         EndLayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
      EndGraph;
   end;
run;
```

In the template, the default title of the Q-Q plot is specified by the ENTRYTITLE statement. The variable _DepLabel is a dynamic variable that provides the name of the dependent variable in the regression analysis. In this case, the name is y. In this template, the label for the axes are specified by the LABEL= suboption of the YAXISOPTS= option for the LAYOUT OVERLAY statement. In other templates, the axis labels come from the column labels of the X-axis and Y-axis columns of the data object. You can see these labels by

specifying ODS OUTPUT with the plot data object and running PROC CONTENTS with the resulting SAS data set.

Suppose you want to change the title to "Analysis of Residuals", and you want the Y-axis label to display the name of the dependent variable. First, replace the ENTRYTITLE statement with the following statement:

```
entrytitle "Analysis of Residuals";
```

Next, replace the LABEL= suboption with the following:

```
label=("Standardized Robust Residual for " _DEPLABEL)
```

You can use dynamic text variables such as _DepLabel in any text element.

You can then submit the modified template definition as you would any SAS program, for example, by selecting **Submit** from the **Run** menu. After submitting the PROC TEMPLATE statements, you should see the following message in the SAS log:

```
NOTE: STATGRAPH 'Stat.Robustreg.Graphics.QQPlot' has been
  saved to: SASUSER.TEMPLAT
```

For more information about graph templates and the graph template language, see the section "Graph Templates" on page 844.

Finally, resubmit the PROC ROBUSTREG statements to display the Q-Q plot created with your modified template. The following statements create Output 25.1.2:

```
proc template;
   define statgraph Stat.Robustreg.Graphics.QQPlot;
      notes "Q-Q Plot for Standardized Robust Residuals";
      dynamic _DEPLABEL Residual _byline_ _bytitle_ _byfootnote_;
      BeginGraph;
         entrytitle "Analysis of Residuals";
         Layout Overlay /
            yaxisopts=(label=("Standardized Robust Residual for " _DEPLABEL))
            xaxisopts=(label="Quantile");
            SCATTERPLOT y=eval (SORT(DROPMISSING(RESIDUAL))) x=eval (
               PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
               N(RESIDUAL)))) / primary=true
               markerattrs=GRAPHDATADEFAULT
               rolename=(q=eval(
               PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
               N(RESIDUAL)))) s=eval (SORT(DROPMISSING(RESIDUAL))))
               tip=(q s) tiplabel=(q= "Quantile" s="Residual");
            lineparm slope=eval (STDDEV(RESIDUAL)) Y=eval (MEAN(RESIDUAL)) X=0 /
               clip=false lineattrs=GRAPHREFERENCE extend=true;
         EndLayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
```

```
      EndGraph;
   end;
run;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
```

Output 25.1.2 Q-Q Plot with Modified Title and Y-Axis Label

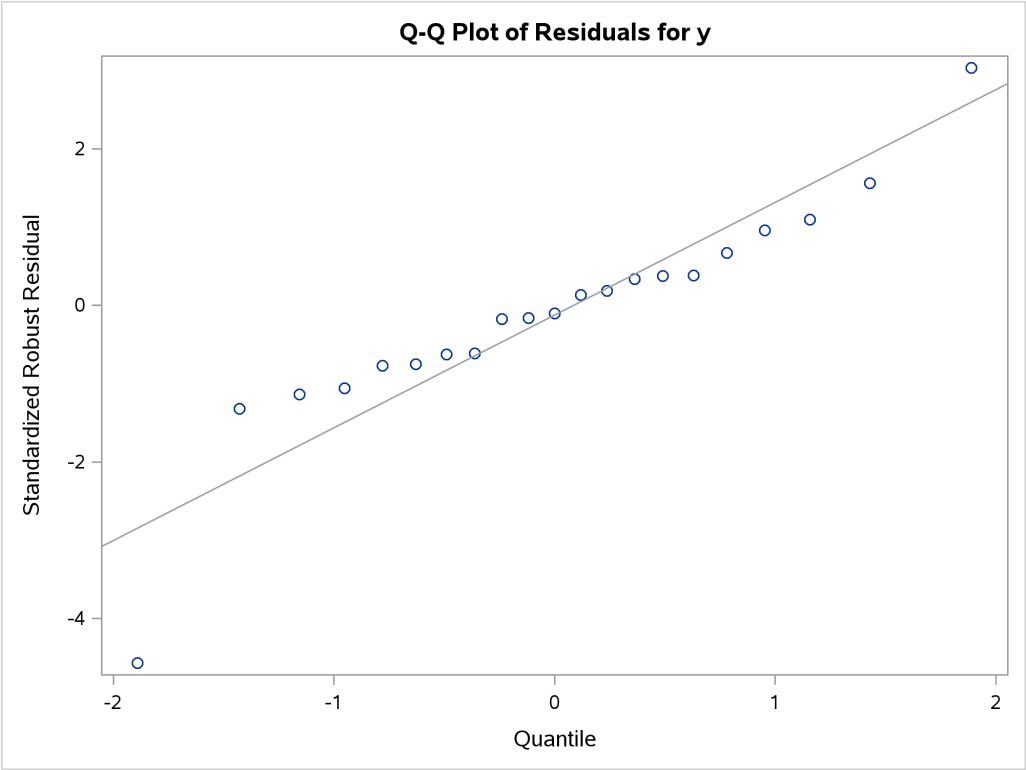

If you have not changed the default template search path, the modified template **QQPlot** is used automatically because Sasuser.Templat occurs before Sashelp.Tmplmst in the ODS search path. For more information about the template search path and the ODS PATH statement, see the sections "Saving Customized Templates" on page 853, "Using Customized Templates" on page 854, and "Reverting to the Default Templates" on page 854.

You do not need to rerun the PROC ROBUSTREG analysis after you modify a graph template if you have stored the plot in an ODS document. After you modify your template, you can submit the PROC DOCUMENT statements in Example 24.4 in Chapter 24, "Statistical Graphics Using ODS," to replay the Q-Q plot with the modified template. You can run the following statements to revert to the default template:

```
proc template;
   delete Stat.Robustreg.Graphics.QQPlot / store=sasuser.templat;
run;
```

Modifying Colors, Line Styles, and Markers

This section shows you how to customize colors, line attributes, and marker symbol attributes by modifying a graph template. In the **QQPlot** template definition shown previously, the SCATTERPLOT statement specifies a scatter plot of normal quantiles versus ordered standardized residuals. The attributes of the marker symbol in the scatter plot are specified by: **MarkerAttrs=GraphDataDefault**. This is a reference to the style element **GraphDataDefault**. For more information, see the section "ODS Style Elements and Attributes" on page 726 in Chapter 24, "Statistical Graphics Using ODS."

The actual value of the marker symbol depends on the style that you are using. In this case, since the HTMLBLUE style is used, the marker symbol is a circle. You can specify a filled circle as the marker symbol by overriding the symbol portion of the style specification as follows: **MarkerAttrs=GraphDataDefault(symbol=CircleFilled)**.

The value of the SYMBOL= option can be any valid marker symbol or a reference to a style attribute of the form **style-element:attribute**. It is recommended that you use style attributes because they are chosen to provide consistency and appropriate emphasis based on display principles for statistical graphics. If you specify values directly in a template, you are overriding the style and you run the risk of creating a graph that is inconsistent with the style template. For more information about the syntax of the Graph Template Language and style elements for graphics, see the *SAS Graph Template Language: Reference* and the *SAS Output Delivery System: User's Guide*.

Similarly, you can change the line color and pattern with the LINEATTRS= option in the LINEPARM statement. The LINEPARM statement displays a straight line specified by slope and intercept parameters. The following option changes the color of the line to red and the line pattern to dashed, by overriding those aspects of the style specification: **LineAttrs=GraphReference(color=red pattern=dash)**. To see the results, submit the modified template definition and the PROC ROBUSTREG statements as follows to create Output 25.1.3:

```
proc template;
   define statgraph Stat.Robustreg.Graphics.QQPlot;
      notes "Q-Q Plot for Standardized Robust Residuals";
      dynamic _DEPLABEL Residual _byline_ _bytitle_ _byfootnote_;
      BeginGraph;
         entrytitle "Analysis of Residuals";
         Layout Overlay /
            yaxisopts=(label=("Standardized Robust Residual for " _DEPLABEL))
            xaxisopts=(label="Quantile");
            SCATTERPLOT y=eval (SORT(DROPMISSING(RESIDUAL))) x=eval (
               PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
               N(RESIDUAL)))) / primary=true
               markerattrs=GraphDataDefault(symbol=CircleFilled)
               rolename=(q=eval(
               PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
               N(RESIDUAL)))) s=eval (SORT(DROPMISSING(RESIDUAL))))
               tip=(q s) tiplabel=(q= "Quantile" s="Residual");
            lineparm slope=eval (STDDEV(RESIDUAL)) Y=eval (MEAN(RESIDUAL)) X=0 /
               clip=false lineattrs=GraphReference(color=red pattern=dash)
               extend=true;
         EndLayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
```

```
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
      EndGraph;
   end;
run;

ods graphics on;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
```

Output 25.1.3 Q-Q Plot with Modified Marker Symbols and Line

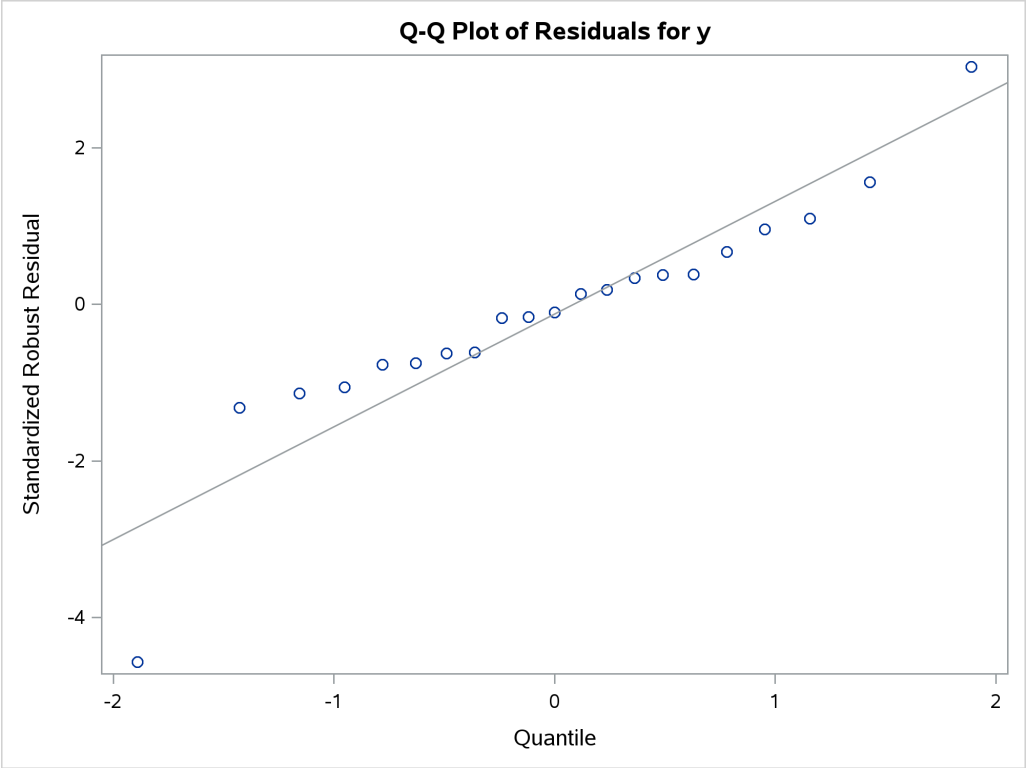

Alternatively, you can replay the plot with PROC DOCUMENT, as in Example 24.4 in Chapter 24, "Statistical Graphics Using ODS."

Modifying Tick Marks and Grid Lines

This section illustrates how to modify axis tick marks and control grid lines. For example, you can specify the following statement to request tick marks ranging from –4 to 2 in the Y-axis:

```
layout Overlay / yaxisopts=(linearopts=(tickvaluelist=(-4 -3 -2 -1 0 1 2)));
```

The LINEAROPTS= option is used for standard linearly scaled axes (as opposed to log-scaled axes). You use the TICKVALUELIST= to specify the tick marks.

You can control the grid lines by using the GRIDDISPLAY= suboption in the YAXISOPTS= option. Typically, you specify either GRIDDISPLAY=AUTO_OFF (grid lines are not displayed unless the **GraphGridLines** element in the current style contains **DisplayOpts="ON"**) or GRIDDISPLAY=AUTO_ON (grid lines are displayed unless the **GraphGridLines** element in the current style contains **DisplayOpts="OFF"**). Here, the template is modified by specifying GRIDDISPLAY=AUTO_ON for both axes. The following statements produce Output 25.1.4:

```
proc template;
   define statgraph Stat.Robustreg.Graphics.QQPlot;
      notes "Q-Q Plot for Standardized Robust Residuals";
      dynamic _DEPLABEL Residual _byline_ _bytitle_ _byfootnote_;
      BeginGraph;
         entrytitle "Analysis of Residuals";
         Layout Overlay /
            yaxisopts=(gridDisplay=Auto_On
                       linearopts=(tickvaluelist=(-4 -3 -2 -1 0 1 2))
                       label=("Standardized Robust Residual for " _DEPLABEL))
            xaxisopts=(gridDisplay=Auto_On label="Quantile");
            SCATTERPLOT y=eval (SORT(DROPMISSING(RESIDUAL))) x=eval (
               PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
               N(RESIDUAL)))) / primary=true
               markerattrs=GraphDataDefault(symbol=CircleFilled)
               rolename=(q=eval(
               PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
               N(RESIDUAL)))) s=eval (SORT(DROPMISSING(RESIDUAL))))
               tip=(q s) tiplabel=(q= "Quantile" s="Residual");
            lineparm slope=eval (STDDEV(RESIDUAL)) Y=eval (MEAN(RESIDUAL)) X=0 /
               clip=false lineattrs=GraphReference(color=red pattern=dash)
               extend=true;
         EndLayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
      EndGraph;
   end;
run;

ods graphics on;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
```

Output 25.1.4 Q-Q Plot with Modified Y-Axis Tick Marks and Grids

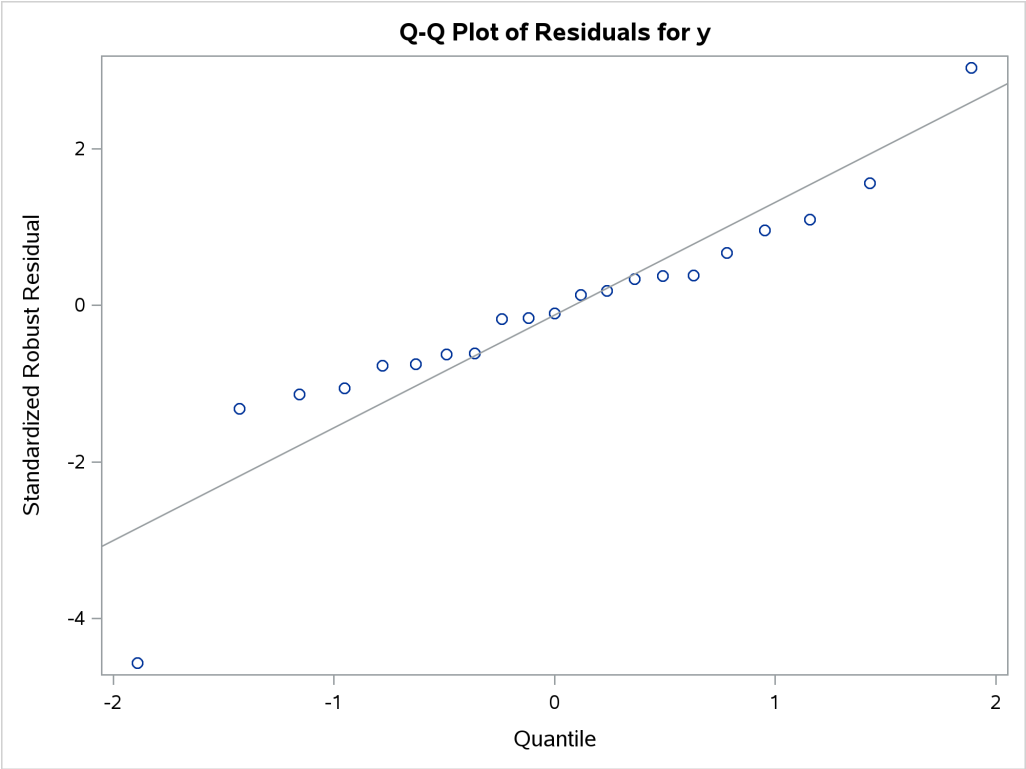

You can restore the default template by running the following step:

```
proc template;
   delete Stat.Robustreg.Graphics.QQPlot / store=sasuser.templat;
run;
```

For more information about grid lines, see the section "Modifying the Style to Show Grid Lines" on page 868.

Modifying the Style to Show Grid Lines

The section "Modifying Tick Marks and Grid Lines" on page 866 explains that grid lines in graphs are controlled both by template options and by the style. Some graphs never display grid lines because they would interfere with the display. Some graphs always display grid lines because they are a critical part of the display. In both cases, grid control is so important that the template writer is not willing to give control to the style. If you want to change the grid display setting for these graphs, you must edit their templates. However, most templates let the style control the grid lines. Either they do not display grid lines unless the style forces them on, or they display grid lines unless the style forces them off. The HTMLBLUE, STATISTICAL, DEFAULT, and most other styles use the setting **`DisplayOpts = "Auto"`**. Then templates that specify GRIDDISPLAY=AUTO_OFF (the default) do not display grid lines, and templates that specify GRIDDISPLAY=AUTO_ON do display grid lines. You can easily make a new style with **`DisplayOpts = "On"`** or **`DisplayOpts = "Off"`** if you would prefer to see grid lines more or less often. This example shows how to set **`DisplayOpts = "On"`**.

First, you need to find the part of the style that enables grid lines. The following step displays the HTMLBLUE style and uses the EXPAND option to display its parent styles, STATISTICAL and DEFAULT:

```
proc template;
   source Styles.HTMLBlue / expand;
run;
```

The advantage of displaying all three styles together is that you can do one search of the results. If grids are defined in the HTMLBLUE style, you will find that first. Otherwise, you will first find the definition in one of the parent styles. An abridged version of the results follows:

```
. . .
class GraphGridLines /
   displayopts = "auto"
   linethickness = 1px
   linestyle = 1
   contrastcolor = GraphColors('ggrid')
   color = GraphColors('ggrid');
. . .
```

You can use this to create a new style that inherits from the HTMLBLUE style, but sets the display options for grids to ON, as in the following example:

```
proc template;
   define style Styles.MyGrids;
      parent=styles.HTMLBlue;
      class GraphGridLines /
         displayopts = "on"
         linethickness = 1px
         linestyle = 1
         contrastcolor = GraphColors('ggrid')
         color = GraphColors('ggrid');
   end;
run;
```

You can use this new style as in the following example:

```
ods graphics on;
ods listing style=mygrids;

proc robustreg data=stack plots=qqplot;
   ods select QQPlot;
   model y = x1 x2 x3;
run;
```

The preceding statements produce Output 25.1.5, which shows the Q-Q plot with grid lines displayed. The default graph template that SAS supplies is used because the custom template created in the section "Modifying Tick Marks and Grid Lines" on page 866 is deleted at the end of that section.

Output 25.1.5 A Style that Makes Grid Lines the Typical Default

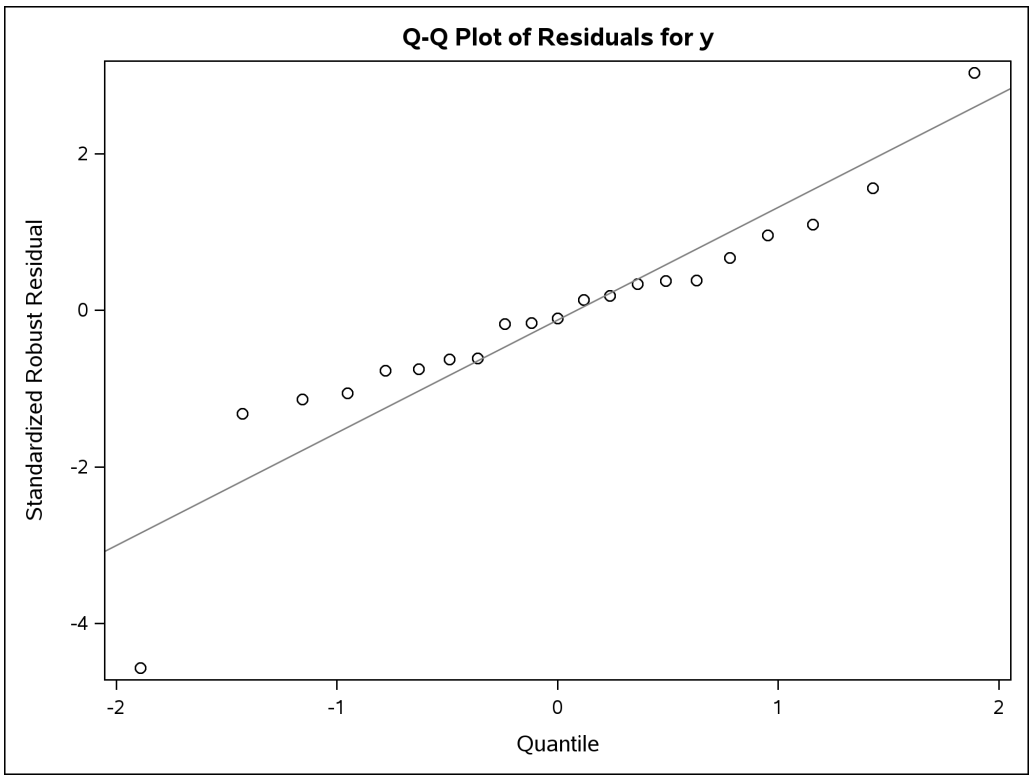

Example 25.2: Adding Equations and Special Characters to Fit Plots

(View the complete code for this example (templex2.sas) in the example repository.)

This example shows how to run the REG and TRANSREG procedures to get fit plots. The R square, mean, and equation for the regression model are output to data sets, and the results are processed and then displayed in subsequent fit plots. This example also illustrates Unicode and how to add special characters to graphs (for example, $\hat{\mu}$ and R^2). The Unicode Consortium `http://unicode.org/` provides a list of character codes at `http://www.unicode.org/charts/charindex.html`.

Simple Linear Regression

The following step runs PROC REG to fit a simple regression model and creates Output 25.2.2 and Output 25.2.1:

```
ods graphics on;
ods trace on;

proc reg data=sashelp.class;
   model weight = height;
run;
```

Output 25.2.1 PROC REG Output

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Number of Observations Read	19
Number of Observations Used	19

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	7193.24912	7193.24912	57.08	<.0001
Error	17	2142.48772	126.02869		
Corrected Total	18	9335.73684			

Root MSE	11.22625	**R-Square**	0.7705
Dependent Mean	100.02632	**Adj R-Sq**	0.7570
Coeff Var	11.22330		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

Output 25.2.2 PROC REG Fit Plot

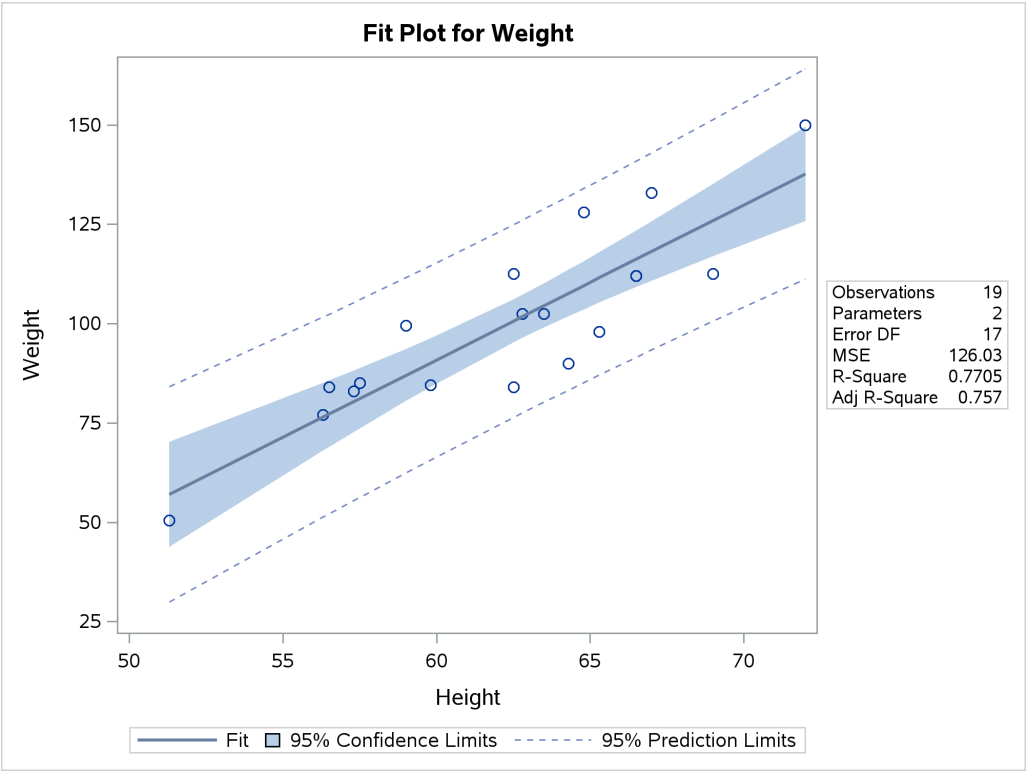

The fit statistics table following the "Analysis of Variance" table displays the R square and the mean. The last table displays the parameter estimates. This information is produced and processed for inclusion in the fit plot as follows:

```
proc reg data=sashelp.class;
   ods output fitstatistics=fs ParameterEstimates=c;
   model weight = height;
run;

data _null_;
   set fs;
   if _n_ = 1 then call symputx('R2'  , put(nvalue2, 4.2)   , 'G');
   if _n_ = 2 then call symputx('mean', put(nvalue1, best6.), 'G');
run;

data _null_;
   set c;
   length s $ 200;
   retain s ' ';
   if _n_ = 1 then
      s = trim(dependent) || ' = ' ||       /* dependent =             */
          put(estimate, best5. -L);         /* intercept               */
   else if abs(estimate) > 1e-8 then        /* skip zero coefficients  */
      s = catx(' ', s,                      /* string so far           */
               ifc(estimate < 0, '-', '+'), /* subtract (-) or add (+) */
               put(abs(estimate), best5.),  /* abs(coefficient)        */
               variable);                   /* variable name           */
   call symputx('formula', s, 'G');
run;
```

Two SAS data sets are made from the tabular output, and the R square, mean, and equation for the regression model are stored in macro variables. The following step uses PROC SGPLOT with an INSET statement to display the linear fit plot along with the R square, mean, and equation for the regression model:

```
proc sgplot data=sashelp.class;
   title 'Simple Linear Regression';
   inset "&formula"
         "R(*ESC*){sup '2'} = &r2"
         "(*ESC*){unicode mu}(*ESC*){unicode hat} = &mean" / position=topleft;
    reg y=weight x=height / clm cli;
run;
```

The results are displayed in Output 25.2.3.

Each separate string in the INSET statement is displayed in a separate line. The first string is the formula, which is generated in the second DATA step. The next string is the R square, and it consists of an 'R', an escaped superscript 2, and the value of R square (which is stored in a macro variable). The string for the mean consists of two Unicode specifications, one for the Greek letter μ, and one to put a hat over it. These special character specifications appear in quotes and are escaped with **(*ESC*)** so that they are processed as special characters rather than as literal text. Typically, you must escape special characters in quotes, and not escape them when they are not in quotes. See the section "Unicode and Special Characters" on page 878 for a list of a few of the more commonly used Unicode characters.

Output 25.2.3 Fit Plot from PROC SGPLOT with Equation

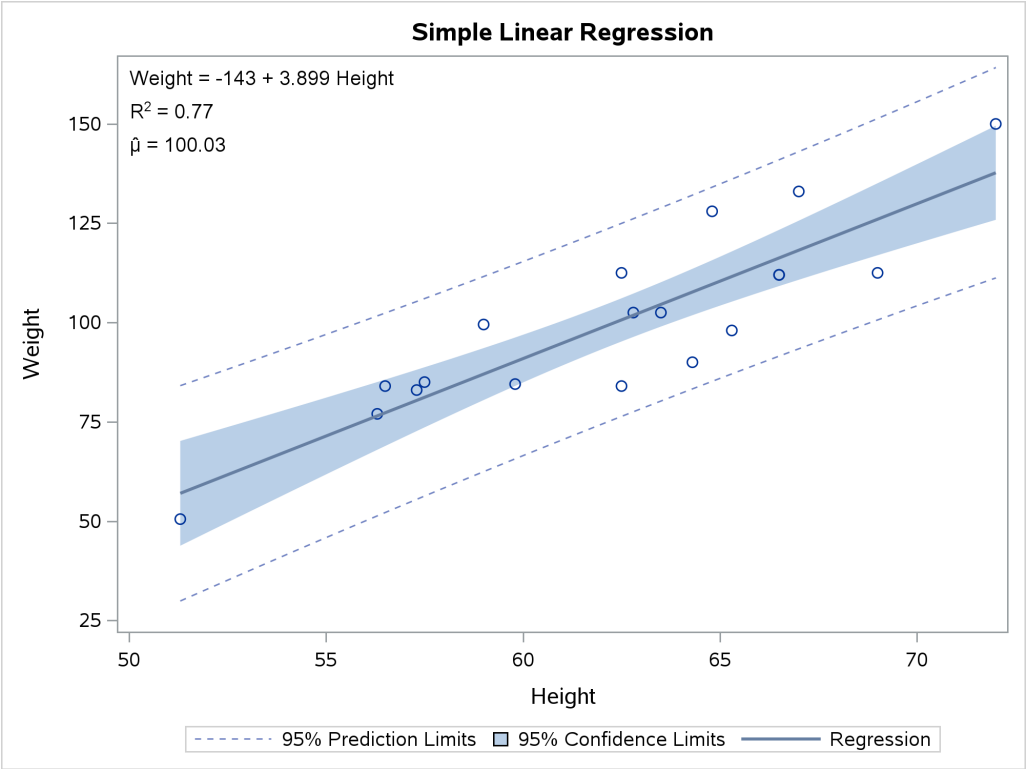

The same information can be added to the graph that PROC REG produces by adding the following statements to the PROC REG template for a fit plot:

```
mvar formula;

layout gridded / autoalign=(topleft topright bottomleft
                            bottomright);
   entry halign=left formula;
   entry halign=left "R"{sup '2'} " = " eval(put(_rsquare, 4.2));
   entry halign=left "(*ESC*){unicode mu}(*ESC*){unicode hat} = "
         eval(put(_depmean, best6.))
         / textattrs=GraphValueText
                     (family=GraphUnicodeText:FontFamily);
endlayout;
```

The MVAR statement names macro variables whose values are added to the graph. The MVAR statement is added to the PROC REG fit plot template near the top. The LAYOUT GRIDDED block creates a table that consists of the equation, R square, and mean. The LAYOUT GRIDDED block is added to the PROC REG fit plot template inside the LAYOUT OVERLAY block. The option **`autoalign=(topleft topright bottomleft bottomright)`** is used to position the table in a part of the graph that is open, first trying the top left corner.

In this example, in the LAYOUT GRIDDED block, two dynamic variables for R square, and the mean are used instead of the macro variables that were made in previous steps. The origin of the names of the two dynamic variables that are used in this example are revealed in the next step when the source code for the PROC REG fit plot is displayed. The first ENTRY statement creates a text line for the formula and left-justifies it. The

second ENTRY statement creates the R square line. It consists of a literal 'R', a specification for a superscript of 2 ({**sup 2**}), an equal sign surrounded by spaces, and the formatted value of the dynamic variable with the R square. The third ENTRY statement creates the mean line. It consists of two Unicode specifications, one for the Greek letter μ, and one to put a hat over it. These special character specifications appear in quotes (unlike the {**sup 2**}) and are escaped with **(*ESC*)** so that they are processed as special characters rather than as literal text. Typically, you must escape special characters in quotes, and not escape them when they are not in quotes. Note that **sup** along with **sub** (subscript) must not appear in quotes in the GTL, but they can appear in quotes in PROC SGPLOT (as was previously shown). The option **textattrs=GraphValueText (family=GraphUnicodeText:FontFamily)** is specified to ensure that a font that recognizes Unicode characters is used. See the section "Unicode and Special Characters" on page 878 for a list of a few of the more commonly used Unicode characters.

You can use the trace information from the PROC REG step (not shown) and the following step to display the template for the fit plot:

```
proc template;
   source Stat.Reg.Graphics.Fit;
run;
```

Some of the results are as follows:

```
if (_SHOWRSQUARE^=0)
   entry halign=left "R-Square" / valign=top;
   entry halign=right eval (PUT(_RSQUARE,BEST6.)) / valign=top;
endif;

if (_SHOWDEPMEAN^=0)
   entry halign=left "Dependent Mean" / valign=top;
   entry halign=right eval (PUT(_DEPMEAN,BEST6.)) / valign=top;
endif;
```

The preceding results show that the dynamic variables _RSquare and _DepMean contain the R square and the mean of the dependent variable. The MVAR statement and the LAYOUT GRIDDED block can be added to the template, and in the interest of maximizing graph size, the table of statistics can be removed, creating the following template:

```
proc template;
   define statgraph Stat.Reg.Graphics.Fit;
      notes "Fit Plot";
      mvar formula;
      dynamic _DEPLABEL _DEPNAME _MODELLABEL _SHOWSTATS _NSTATSCOLS _SHOWNObs
         _SHOWTOTFREQ _SHOWNParm _SHOWEDF _SHOWMSE _SHOWRSquare _SHOWAdjRSq
         _SHOWSSE _SHOWDepMean _SHOWCV _SHOWAIC _SHOWBIC _SHOWCP _SHOWGMSEP
         _SHOWJP _SHOWPC _SHOWSBC _SHOWSP _NObs _NParm _EDF _MSE _RSquare
         _AdjRSq _SSE _DepMean _CV _AIC _BIC _CP _GMSEP _JP _PC _SBC _SP
         _PREDLIMITS _CONFLIMITS _XVAR _SHOWCLM _SHOWCLI _WEIGHT _SHORTXLABEL
         _SHORTYLABEL _TITLE _TOTFreq _byline_ _bytitle_ _byfootnote_;
      BeginGraph;
         entrytitle halign=left textattrs=GRAPHVALUETEXT _MODELLABEL
            halign=center textattrs=GRAPHTITLETEXT _TITLE " for " _DEPNAME;
         layout Overlay / yaxisopts=(label=_DEPLABEL shortlabel=_SHORTYLABEL)
            xaxisopts=(shortlabel=_SHORTXLABEL);
            if (_SHOWCLM=1)
```

```
            BANDPLOT limitupper=UPPERCLMEAN limitlower=LOWERCLMEAN x=_XVAR
               / fillattrs=GRAPHCONFIDENCE connectorder=axis
               name="Confidence" LegendLabel=_CONFLIMITS;
         endif;
         if (_SHOWCLI=1)
            if (_WEIGHT=1)
               SCATTERPLOT y=PREDICTEDVALUE x=_XVAR / markerattrs=(size=0)
                  datatransparency=.6 yerrorupper=UPPERCL
                  yerrorlower=LOWERCL name="Prediction"
                  LegendLabel=_PREDLIMITS;
            else
               BANDPLOT limitupper=UPPERCL limitlower=LOWERCL x=_XVAR /
                  display=(outline) outlineattrs=GRAPHPREDICTIONLIMITS
                  connectorder=axis name="Prediction"
                  LegendLabel=_PREDLIMITS;
            endif;
         endif;
         SCATTERPLOT y=DEPVAR x=_XVAR / markerattrs=GRAPHDATADEFAULT
            primary=true rolename=(_tip1=OBSERVATION _id1=ID1 _id2=ID2
            _id3=ID3 _id4=ID4 _id5=ID5)
            tip=(y x _tip1 _id1 _id2 _id3 _id4 _id5);
         SERIESPLOT y=PREDICTEDVALUE x=_XVAR / lineattrs=GRAPHFIT
            connectorder=xaxis name="Fit" LegendLabel="Fit";
         if (_SHOWCLI=1 OR _SHOWCLM=1)
            DISCRETELEGEND "Fit" "Confidence" "Prediction" / across=3
               HALIGN=CENTER VALIGN=BOTTOM;
         endif;
         layout gridded / autoalign=(topleft topright bottomleft
                                     bottomright);
            entry halign=left formula;
            entry halign=left "R"{sup '2'} " = " eval(put(_rsquare, 4.2));
            entry halign=left "(*ESC*){unicode mu}(*ESC*){unicode hat} = "
                  eval(put(_depmean, best6.))
                  / textattrs=GraphValueText
                              (family=GraphUnicodeText:FontFamily);
         endlayout;
      endlayout;
      if (_BYTITLE_)
         entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
      else
         if (_BYFOOTNOTE_)
            entryfootnote halign=left _BYLINE_;
         endif;
      endif;
   EndGraph;
end;
run;
```

The following step uses the modified template to create Output 25.2.4:

```
proc reg data=sashelp.class;
   model weight = height;
run;
```

Output 25.2.4 PROC REG Fit Plot with the Equation

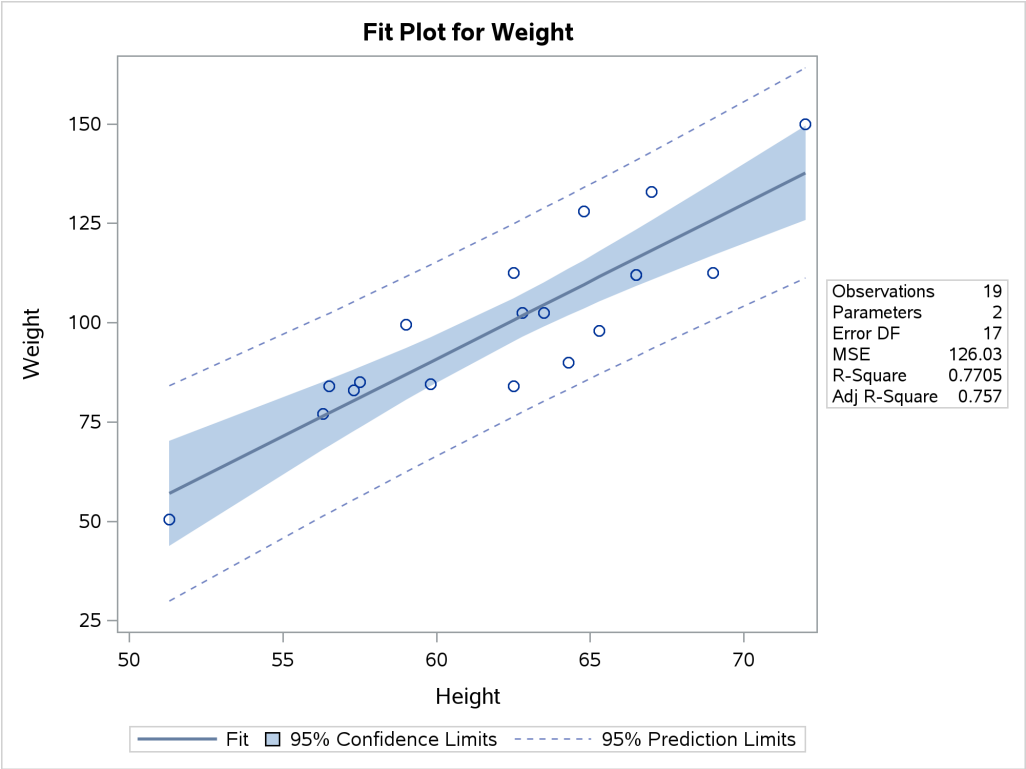

You can restore the default template by running the following step:

```
proc template;
   delete Stat.Reg.Graphics.Fit / store=sasuser.templat;
run;
```

Cubic Fit Function

The following steps run PROC TRANSREG to find a cubic fit function and display the equation in a plot generated by PROC SGPLOT:

```
proc transreg data=sashelp.class ss2;
   ods output fitstatistics=fs coef=c;
   model identity(weight) = pspline(height);
run;

data _null_;
   set fs;
   if _n_ = 1 then call symputx('R2'  , put(value2, 4.2)   , 'G');
   if _n_ = 2 then call symputx('mean', put(value1, best6.), 'G');
run;

data _null_;
   set c end=eof;
   length s $ 200 c $ 1;
   retain s ' ';
```

```
   if _n_ = 1 then
      s = scan(dependent, 2, '()') || ' = ' ||    /* dependent =             */
          put(coefficient, best5. -L);            /* intercept               */
   else if abs(coefficient) > 1e-8 then do;       /* skip zero coefficients  */
      s = catx(' ', s,                            /* string so far           */
               ifc(coefficient < 0, '-', '+'),    /* subtract (-) or add (+) */
               put(abs(coefficient), best5. -L ), /* abs(coefficient)        */
               scan(variable, 2, '._'));          /* variable name           */
      c = scan(variable, 2, '_');                 /* grab power              */
      if c ne '1' then                            /* skip power for linear   */
         s = cats(s, "(*ESC*){sup '", c, "'}");   /* add superscript         */
   end;
   if eof then call symputx('formula', trim(s), 'G');
run;

proc sgplot data=sashelp.class;
   title 'Cubic Fit Function';
   inset "&formula"
         "R(*ESC*){sup '2'} = &r2"
         "(*ESC*){unicode mu}(*ESC*){unicode hat} = &mean" / position=topleft;
    reg y=weight x=height / degree=3 cli clm;
run;
```

These steps create Output 25.2.5.

The PROC TRANSREG MODEL statement fits a model with an untransformed dependent variable and a cubic polynomial function of the independent variable. By default, PSPLINE fits a cubic polynomial spline with no knots, which is simply a cubic polynomial. The fit statistics and parameter estimates are output to data sets, and their values are stored in macro variables. There are three independent variables plus the intercept. Variable names and exponents are extracted from the TRANSREG parameter names of Pspline.Height_1, Pspline.Height_2, and Pspline.Height_3 by using the SCAN function. Exponents are added by using the specifications **"(*ESC*){sup '2'}"** and **"(*ESC*){sup '3'}"**, which are explained in more detail with the INSET statement.

PROC SGPLOT with an INSET statement makes the plot. Each separate string is displayed in a separate line. The first string is the formula, which is generated in the second DATA step. The next string is the R square: it consists of an 'R', an escaped superscript 2, and the value of R square (which is stored in a macro variable). The string for the mean consists of two Unicode specifications, one for the Greek letter μ, and one to put a hat over it. These special character specifications appear in quotes and are escaped with **(*ESC*)** so that they are processed as special characters rather than as literal text. Typically, you must escape special characters in quotes, and not escape them when they are not in quotes. For more information about Unicode characters, see section "Simple Linear Regression" on page 870. For a list of a few of the more commonly used Unicode characters, see the section "Unicode and Special Characters" on page 878.

Output 25.2.5 Cubic Fit Function with the Equation

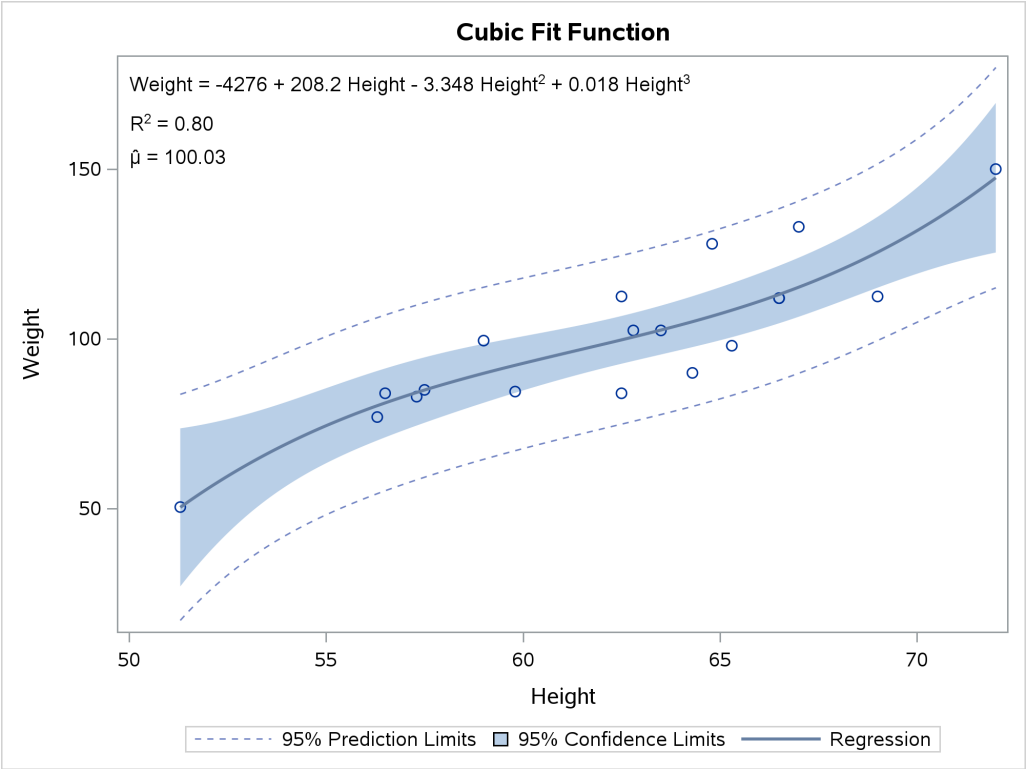

Unicode and Special Characters

The following steps illustrate Unicode specifications for a number of commonly used characters and create Output 25.2.6 and Output 25.2.7, which are charts of Unicode characters:

```
%let l = halign=left;
proc template;
   define statgraph class;
      begingraph / designheight=640px designwidth=520px;
         layout overlay / xaxisopts=(display=none) yaxisopts=(display=none);
            layout gridded / columns=3 autoalign=(topleft);
               entry &l textattrs=(weight=bold) 'Description';
               entry &l textattrs=(weight=bold) 'Displayed';
               entry &l textattrs=(weight=bold) "Unicode";
               entry &l 'R Square';
               entry &l 'R' {sup '2'};
               entry &l "'R' {sup '2'}";
               entry &l 'y hat sub i';
               entry &l 'y' {unicode hat}{sub 'i'};
               entry &l "'y' {unicode hat}{sub 'i'}";
               entry &l 'plus or minus';
               entry &l '7 ' {unicode '00B1'x} ' 2';
               entry &l "'7 ' {unicode '00B1'x} ' 2'";
               entry &l 'not equal to             ';
               entry &l 'a ' {unicode '2260'x} ' b';
               entry &l "'a ' {unicode '2260'x} ' b'";
```

```
entry &l 'less than or equal       ';
entry &l 'a ' {unicode '2264'x} ' b';
entry &l "'a ' {unicode '2264'x} ' b'";
entry &l 'greater than or equal    ';
entry &l 'b ' {unicode '2265'x} ' a';
entry &l "'b ' {unicode '2265'x} ' a'";
entry &l 'almost equal';
entry &l 'a ' {unicode '2248'x} ' b';
entry &l "'a ' {unicode '2248'x} ' b'";
entry &l 'infinity';
entry &l {unicode '221e'x};
entry &l "{unicode '221e'x}";
entry &l 'combining tilde';
entry &l 'El nin' {unicode tilde} 'o';
entry &l "'El nin' {unicode tilde} 'o'";
entry &l 'grave accent';
entry &l 'cre' {unicode '0300'x} 'me';
entry &l "'cre' {unicode '0300'x} 'me'";
entry &l 'circumflex, acute accent   ';
entry &l 'bru' {unicode '0302'x} 'le' {unicode '0301'x} 'e';
entry &l "'bru' {unicode '0302'x} 'le' {unicode '0301'x} 'e'";
entry &l 'degree sign';
entry &l '90' {unicode '00B0'x} ' angle';
entry &l "'90' {unicode '00B0'x} ' angle'";
entry &l 'euro sign';
entry &l {unicode '20AC'x} '100';
entry &l "{unicode '20AC'x} '100'";
entry &l 'cent sign';
entry &l '25' {unicode '00A2'x};
entry &l "'25' {unicode '00A2'x}";
entry &l 'copyright';
entry &l {unicode '00a9'x};
entry &l "{unicode '00a9'x}";
entry &l 'bullet';
entry &l {unicode '2022'x};
entry &l "{unicode '2022'x}";
entry &l 'alpha';
entry &l {unicode alpha} '   ' {unicode alpha_u};
entry &l "{unicode alpha} '   ' {unicode alpha_u}";
entry &l 'beta';
entry &l {unicode beta} '   ' {unicode beta_u};
entry &l "{unicode beta} '   ' {unicode beta_u}";
entry &l 'gamma';
entry &l {unicode gamma} '   ' {unicode gamma_u};
entry &l "{unicode gamma} '   ' {unicode gamma_u}";
entry &l 'delta';
entry &l {unicode delta} '   ' {unicode delta_u};
entry &l "{unicode delta} '   ' {unicode delta_u}";
entry &l 'epsilon';
entry &l {unicode epsilon} '   ' {unicode epsilon_u};
entry &l "{unicode epsilon} '   ' {unicode epsilon_u}";
entry &l 'zeta';
entry &l {unicode zeta} '   ' {unicode zeta_u};
entry &l "{unicode zeta} '   ' {unicode zeta_u}";
```

```
entry &l 'eta';
entry &l {unicode eta} '   ' {unicode eta_u};
entry &l "{unicode eta} '   ' {unicode eta_u}";
entry &l 'theta';
entry &l {unicode theta} '   ' {unicode theta_u};
entry &l "{unicode theta} '   ' {unicode theta_u}";
entry &l 'iota';
entry &l {unicode iota} '   ' {unicode iota_u};
entry &l "{unicode iota} '   ' {unicode iota_u}";
entry &l 'kappa';
entry &l {unicode kappa} '   ' {unicode kappa_u};
entry &l "{unicode kappa} '   ' {unicode kappa_u}";
entry &l 'lambda';
entry &l {unicode lambda} '   ' {unicode lambda_u};
entry &l "{unicode lambda} '   ' {unicode lambda_u}";
entry &l 'mu';
entry &l {unicode mu} '   ' {unicode mu_u};
entry &l "{unicode mu} '   ' {unicode mu_u}";
entry &l 'nu';
entry &l {unicode nu} '   ' {unicode nu_u};
entry &l "{unicode nu} '   ' {unicode nu_u}";
entry &l 'xi';
entry &l {unicode xi} '   ' {unicode xi_u};
entry &l "{unicode xi} '   ' {unicode xi_u}";
entry &l 'omicron';
entry &l {unicode omicron} '   ' {unicode omicron_u};
entry &l "{unicode omicron} '   ' {unicode omicron_u}";
entry &l 'pi';
entry &l {unicode pi} '   ' {unicode pi_u};
entry &l "{unicode pi} '   ' {unicode pi_u}";
entry &l 'rho';
entry &l {unicode rho} '   ' {unicode rho_u};
entry &l "{unicode rho} '   ' {unicode rho_u}";
entry &l 'sigma';
entry &l {unicode sigma} '   ' {unicode sigma_u};
entry &l "{unicode sigma} '   ' {unicode sigma_u}";
entry &l 'tau';
entry &l {unicode tau} '   ' {unicode tau_u};
entry &l "{unicode tau} '   ' {unicode tau_u}";
entry &l 'upsilon';
entry &l {unicode upsilon} '   ' {unicode upsilon_u};
entry &l "{unicode upsilon} '   ' {unicode upsilon_u}";
entry &l 'phi';
entry &l {unicode phi} '   ' {unicode phi_u};
entry &l "{unicode phi} '   ' {unicode phi_u}";
entry &l 'chi';
entry &l {unicode chi} '   ' {unicode chi_u};
entry &l "{unicode chi} '   ' {unicode chi_u}";
entry &l 'psi';
entry &l {unicode psi} '   ' {unicode psi_u};
entry &l "{unicode psi} '   ' {unicode psi_u}";
entry &l 'omega';
entry &l {unicode omega} '   ' {unicode omega_u};
entry &l "{unicode omega} '   ' {unicode omega_u}";
```

```
            endlayout;
            scatterplot y=weight x=height / markerattrs=(size=0);
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=sashelp.class template=class;
run;

%let sym = ArrowDown Asterisk Circle CircleFilled Diamond DiamondFilled
           GreaterThan Hash IBeam Plus Square SquareFilled Star StarFilled
           Tack Tilde Triangle TriangleFilled Union X Y Z;
%let uni = 2193 002A 25cb 25cf 25c7 2666 003e 0023 2336 002b 25a1 25a0 2606
           2605 22a4 223c 25b3 25b2 222a 0058 0059 005a;

data x;
   retain fmtname '$markers' x1 0 x2 0.1 x3 1 x4 0.9;
   length m $ 20 u $ 4 l label $ 30;
   do y     = 1 to 22;
      m     = scan("&sym", y);
      u     = scan("&uni", y);
      l     = cat("{unicode '", u, "'x}");
      label = catt('(*ESC*)', l);
      start = m;
      output;
   end;
run;

proc format cntlin=x;
run;

ods graphics on / attrpriority=none width=3in height=4in;

proc sgplot data=x noautolegend;
   title justify=left ' Markers' justify=right 'Unicode Characters';
   styleattrs datasymbols=(&sym) datacontrastcolors=(black);
   scatter x=x1 y=y / group=m;
   text    x=x2 y=y text=m / position=right;
   scatter x=x3 y=y / markerchar=start
                      markercharattrs=(size=9 family='Arial Unicode MS');
   text    x=x4 y=y text=l / position=right;
   format  start $markers.;
   xaxis   display=none offsetmin=0.05 offsetmax=0.27;
   yaxis   display=none offsetmin=0.01 offsetmax=0.05 reverse;
run;

ods graphics on / reset=all;
title;
```

Output 25.2.6 Commonly Used Unicode and Special Characters

Description	Displayed	Unicode
R Square	R^2	'R' {sup '2'}
y hat sub i	$\hat{y}_i$	'y' {unicode hat}{sub 'i'}
plus or minus	7 ± 2	'7 ' {unicode '00B1'x} ' 2'
not equal to	a ≠ b	'a ' {unicode '2260'x} ' b'
less than or equal	a ≤ b	'a ' {unicode '2264'x} ' b'
greater than or equal	b ≥ a	'b ' {unicode '2265'x} ' a'
almost equal	a ≈ b	'a ' {unicode '2248'x} ' b'
infinity	∞	{unicode '221e'x}
combining tilde	El niño	'El nin' {unicode tilde} 'o'
grave accent	crème	'cre' {unicode '0300'x} 'me'
circumflex, acute accent	brûlée	'bru' {unicode '0302'x} 'le' {unicode '0301'x} 'e'
degree sign	90° angle	'90' {unicode '00B0'x} ' angle'
euro sign	€100	{unicode '20AC'x} '100'
cent sign	25¢	'25' {unicode '00A2'x}
copyright	©	{unicode '00a9'x}
bullet	•	{unicode '2022'x}
alpha	α Α	{unicode alpha} ' ' {unicode alpha_u}
beta	β Β	{unicode beta} ' ' {unicode beta_u}
gamma	γ Γ	{unicode gamma} ' ' {unicode gamma_u}
delta	δ Δ	{unicode delta} ' ' {unicode delta_u}
epsilon	ε Ε	{unicode epsilon} ' ' {unicode epsilon_u}
zeta	ζ Ζ	{unicode zeta} ' ' {unicode zeta_u}
eta	η Η	{unicode eta} ' ' {unicode eta_u}
theta	θ Θ	{unicode theta} ' ' {unicode theta_u}
iota	ι Ι	{unicode iota} ' ' {unicode iota_u}
kappa	κ Κ	{unicode kappa} ' ' {unicode kappa_u}
lambda	λ Λ	{unicode lambda} ' ' {unicode lambda_u}
mu	μ Μ	{unicode mu} ' ' {unicode mu_u}
nu	ν Ν	{unicode nu} ' ' {unicode nu_u}
xi	ξ Ξ	{unicode xi} ' ' {unicode xi_u}
omicron	ο Ο	{unicode omicron} ' ' {unicode omicron_u}
pi	π Π	{unicode pi} ' ' {unicode pi_u}
rho	ρ Ρ	{unicode rho} ' ' {unicode rho_u}
sigma	σ Σ	{unicode sigma} ' ' {unicode sigma_u}
tau	τ Τ	{unicode tau} ' ' {unicode tau_u}
upsilon	υ Υ	{unicode upsilon} ' ' {unicode upsilon_u}
phi	ϕ Φ	{unicode phi} ' ' {unicode phi_u}
chi	χ Χ	{unicode chi} ' ' {unicode chi_u}
psi	ψ Ψ	{unicode psi} ' ' {unicode psi_u}
omega	ω Ω	{unicode omega} ' ' {unicode omega_u}

Output 25.2.7 Markers and Unicode Specifications

Markers		Unicode Characters	
↓	ArrowDown	↓	{unicode '2193'x}
✳	Asterisk	*	{unicode '002A'x}
○	Circle	○	{unicode '25cb'x}
●	CircleFilled	●	{unicode '25cf'x}
◇	Diamond	◇	{unicode '25c7'x}
◆	DiamondFilled	◆	{unicode '2666'x}
>	GreaterThan	>	{unicode '003e'x}
#	Hash	#	{unicode '0023'x}
I	IBeam	I	{unicode '2336'x}
+	Plus	+	{unicode '002b'x}
□	Square	□	{unicode '25a1'x}
■	SquareFilled	■	{unicode '25a0'x}
☆	Star	☆	{unicode '2606'x}
★	StarFilled	★	{unicode '2605'x}
T	Tack	T	{unicode '22a4'x}
∿	Tilde	~	{unicode '223c'x}
△	Triangle	△	{unicode '25b3'x}
▲	TriangleFilled	▲	{unicode '25b2'x}
∪	Union	∪	{unicode '222a'x}
X	X	X	{unicode '0058'x}
Y	Y	Y	{unicode '0059'x}
Z	Z	Z	{unicode '005a'x}

The Unicode Consortium `http://unicode.org/` provides a list of character codes at `http://www.unicode.org/charts/charindex.html`.

The following rules apply to Unicode and special character specifications in ODS graphics:

- Each character can be specified by looking up its code and specifying it as a hexadecimal constant. Example: {**unicode '221e'x**}.
- Lower case Greek letters can be specified by using names instead of hexadecimal constants. Example: {**unicode alpha**}.
- Upper case Greek letters can be specified by using names followed by **_u** instead of a hexadecimal constants. Example: {**unicode alpha_u**}.
- Superscript and subscript have special abbreviations. Examples: {**sup 2**} and {**sub 2**}.
- The **sup** and **sub** specifications must not appear escaped and in quotes in the GTL. They must appear outside of quotes.
- Some characters overprint the character that comes before. Example: **'El nin'** {**tilde**} **'o'**, which is equivalent to **'El nin'** {**unicode '0303'x**} **'o'** creates 'El niño'.
- Specifications inside quotes are escaped. Example: **"(*ESC*)**{**unicode beta**}**"**.
- Specifications outside quotes are not escaped. Example: {**unicode beta**}.

Example 25.3: Customizing Panels

(View the complete code for this example (templex3.sas) in the example repository.)

This example illustrates how to modify the regression fit diagnostics panel shown in Output 24.1 in Chapter 24, "Statistical Graphics Using ODS," so that it displays a subset of the component plots. The original panel consists of eight plots and a summary statistics box. The ODS trace output from PROC REG shown previously shows that the template for the diagnostics panel is **Stat.REG.Graphics.DiagnosticsPanel**. The following statements display the template:

```
proc template;
   source Stat.REG.Graphics.DiagnosticsPanel;
run;
```

An abridged version of the results is shown next:

```
define statgraph Stat.Reg.Graphics.DiagnosticsPanel;
   notes "Diagnostics Panel";
   dynamic . . .;
   BeginGraph / designheight=defaultDesignWidth;
      entrytitle halign=left textattrs=GRAPHVALUETEXT _MODELLABEL halign=
         center textattrs=GRAPHTITLETEXT "Fit Diagnostics" " for " _DEPNAME;
      layout lattice / columns=3 rowgutter=10 columngutter=10
                       shrinkfonts=true rows=3;
         layout overlay / xaxisopts=(shortlabel='Predicted');
            . . .
         endlayout;
         layout overlay / xaxisopts=(shortlabel='Predicted');
            . . .
         endlayout;
         layout overlay / xaxisopts=(label='Leverage' offsetmax=0.05)
            yaxisopts=(offsetmin=0.05 offsetmax=0.05);
            . . .
         endlayout;
         layout overlay / yaxisopts=(label="Residual" shortlabel="Resid")
            xaxisopts=(label="Quantile");
            . . .
         endlayout;
         layout overlayequated / xaxisopts=(shortlabel='Predicted')
            yaxisopts=(label=_DEPLABEL shortlabel="Observed") equatetype=square;
            . . .
         endlayout;
         layout overlay / xaxisopts=(linearopts=(integer=true) label=
            "Observation" shortlabel="Obs" offsetmax=0.05)
            yaxisopts=(offsetmin=0.05 offsetmax=0.05);
            . . .
         endlayout;
         layout overlay / xaxisopts=(label="Residual") yaxisopts=(label="Percent");
            . . .
         endlayout;
         layout lattice / columns=2 rows=1 rowdatarange=unionall columngutter=0;
            . . .
         endlayout;
```

```
            if (_SHOWSTATS =1)
               layout overlay;
               . . .
               endLayout;
            endif;
            if (_SHOWSTATS = 2)
               layout overlay / yaxisopts=(gridDisplay=auto_off label="Residual");
                  . . .
               endlayout;
            endif;
         endlayout;
         . . .
      EndGraph;
   end;
```

The outermost components of the template are a BEGINGRAPH/ENDGRAPH block with a lattice layout with ROWS=3 and COLUMNS=3 that defines the 3×3 panel of plots. Inside that are nine layouts, one for each cell, the last of which is conditionally defined. The LAYOUT statements define the components of the panel from left to right and top to bottom. You can eliminate some of the panels and produce a 2×2 panel as follows:

```
proc template;
   define statgraph Stat.Reg.Graphics.DiagnosticsPanel;
      notes "Diagnostics Panel";
      dynamic _DEPLABEL _DEPNAME _MODELLABEL _OUTLEVLABEL _TOTFREQ _NPARM _NOBS
         _OUTCOOKSDLABEL _SHOWSTATS _NSTATSCOLS _DATALABEL _SHOWNObs
         _SHOWTOTFREQ _SHOWNParm _SHOWEDF _SHOWMSE _SHOWRSquare _SHOWAdjRSq
         _SHOWSSE _SHOWDepMean _SHOWCV _SHOWAIC _SHOWBIC _SHOWCP _SHOWGMSEP
         _SHOWJP _SHOWPC _SHOWSBC _SHOWSP _EDF _MSE _RSquare _AdjRSq _SSE
         _DepMean _CV _AIC _BIC _CP _GMSEP _JP _PC _SBC _SP _byline_ _bytitle_
         _byfootnote_;
      BeginGraph / designheight=defaultDesignWidth;
         entrytitle halign=left textattrs=GRAPHVALUETEXT _MODELLABEL halign=
            center textattrs=GRAPHTITLETEXT "Fit Diagnostics" " for " _DEPNAME;
         layout lattice / columns=2 rowgutter=10 columngutter=10
            shrinkfonts=true rows=2;
            layout overlay / xaxisopts=(shortlabel='Predicted');
               referenceline y=0;
               scatterplot y=RESIDUAL x=PREDICTEDVALUE / primary=true datalabel=
                  _OUTLEVLABEL rolename=(_tip1=OBSERVATION _id1=ID1 _id2=ID2
                  _id3=ID3 _id4=ID4 _id5=ID5) tip=(y x _tip1 _id1 _id2 _id3 _id4
                  _id5);
            endlayout;
            layout overlay / yaxisopts=(label="Residual" shortlabel="Resid")
               xaxisopts=(label="Quantile");
               lineparm slope=eval (STDDEV(RESIDUAL)) y=eval (MEAN(RESIDUAL))
                  x=0 / clip=false extend=true lineattrs=GRAPHREFERENCE;
               scatterplot y=eval (SORT(DROPMISSING(RESIDUAL))) x=eval(
                  PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
                  N(RESIDUAL)))) / markerattrs=GRAPHDATADEFAULT primary=true
                  rolename=(s=eval (SORT(DROPMISSING(RESIDUAL))) nq=eval (
                  PROBIT((NUMERATE(SORT(DROPMISSING(RESIDUAL))) -0.375)/(0.25 +
                  N(RESIDUAL))))) tiplabel=(nq="Quantile" s="Residual")
                  tip=(nq s);
```

```
            endlayout;
            layout overlayequated / xaxisopts=(shortlabel='Predicted')
               yaxisopts=(label=_DEPLABEL shortlabel="Observed")
               equatetype=square;
               lineparm slope=1 x=0 y=0 / clip=true extend=true lineattrs=
                  GRAPHREFERENCE;
               scatterplot y=DEPVAR x=PREDICTEDVALUE / primary=true datalabel=
                  _OUTLEVLABEL rolename=(_tip1=OBSERVATION _id1=ID1 _id2=ID2
                  _id3=ID3 _id4=ID4 _id5=ID5) tip=(y x _tip1 _id1 _id2 _id3 _id4
                  _id5);
            endlayout;
            layout overlay / xaxisopts=(label="Residual") yaxisopts=(label=
               "Percent");
               histogram RESIDUAL / primary=true;
               densityplot RESIDUAL / name="Normal" legendlabel="Normal"
                  lineattrs=GRAPHFIT;
            endlayout;
         endlayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
      EndGraph;
   end;
run;

ods graphics on;

proc reg data=sashelp.class;
   model Weight = Height;
quit;
```

This template plots the residuals by predicted values, the Q-Q plot, the actual by predicted plot, and the residual histogram. The results are shown in Output 25.3.1.

Output 25.3.1 Diagnostics Panel with Four Plots

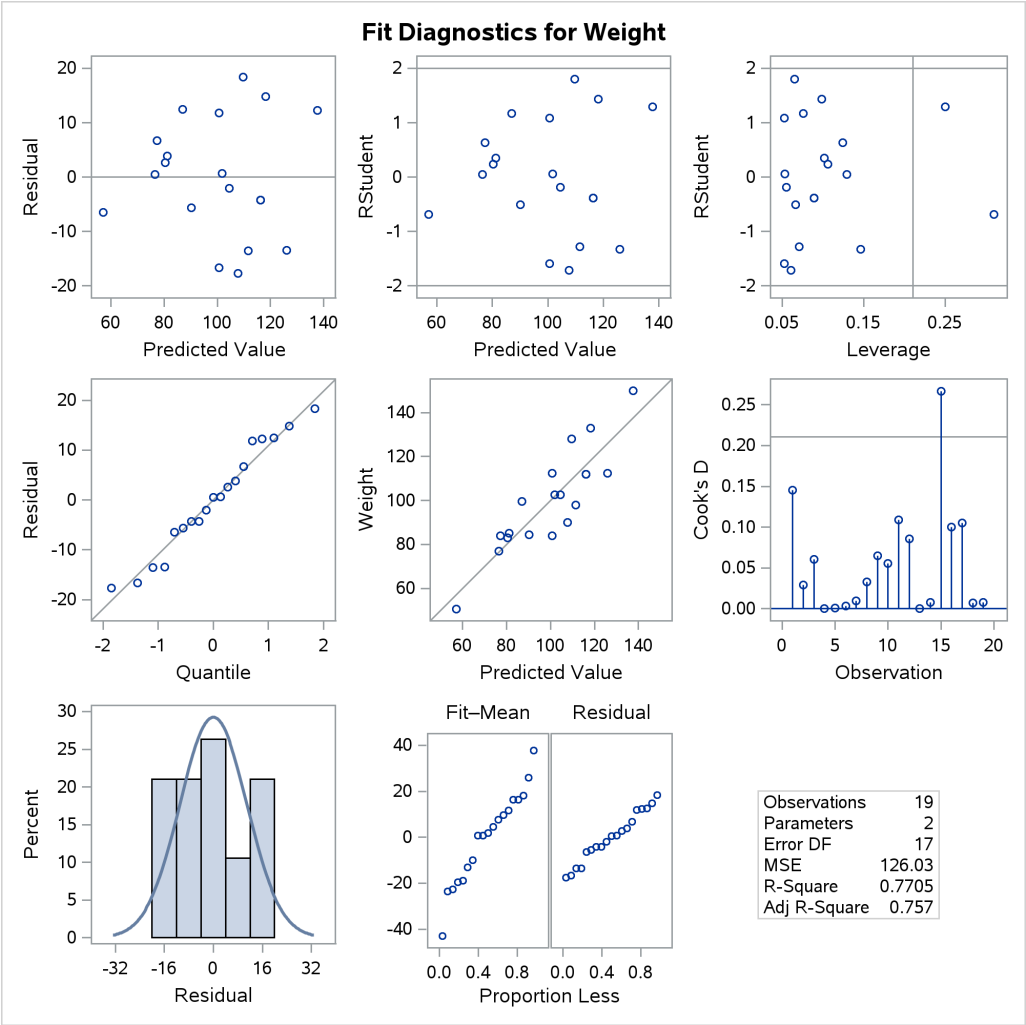

This new template is a straightforward modification of the original template. The COLUMNS=2 and ROWS=2 options in the LAYOUT LATTICE statement request a 2×2 lattice. The LAYOUT statement blocks for components 1, 3, 6, 8, and 9 are deleted. **NOTE:** You do not need to understand every aspect of a template to modify it if you can recognize the overall structure and a few key options.

You can restore the original template as follows:

```
proc template;
   delete Stat.REG.Graphics.DiagnosticsPanel / store=sasuser.templat;
run;
```

Example 25.4: Customizing Axes and Reference Lines

(View the complete code for this example (templex4.sas) in the example repository.)

This example illustrates several ways that you can change the plot axes in a scatter plot. The example uses PROC CORRESP to perform a correspondence analysis. It is taken from the section "Getting Started: CORRESP Procedure" on page 2651 in Chapter 41, "The CORRESP Procedure." It uses the following data:

```
title "Number of Ph.D.'s Awarded from 1973 to 1978";

data PhD;
   input Science $ 1-19 y1973-y1978;
   label y1973 = '1973'
         y1974 = '1974'
         y1975 = '1975'
         y1976 = '1976'
         y1977 = '1977'
         y1978 = '1978';
   datalines;
Life Sciences       4489 4303 4402 4350 4266 4361
Physical Sciences   4101 3800 3749 3572 3410 3234
Social Sciences     3354 3286 3344 3278 3137 3008
Behavioral Sciences 2444 2587 2749 2878 2960 3049
Engineering         3338 3144 2959 2791 2641 2432
Mathematics         1222 1196 1149 1003  959  959
;
```

The following steps perform the correspondence analysis and create Output 25.4.1:

```
ods graphics on;
ods trace on;

proc corresp data=PhD short;
   ods select configplot;
   var y1973-y1978;
   id Science;
run;
```

The trace output for this step (not shown) shows that the template for this plot is **Stat.Corresp.Graphics.Configuration**. The following step displays this template:

```
proc template;
   source Stat.Corresp.Graphics.Configuration;
run;
```

Output 25.4.1 Default Scatter Plot

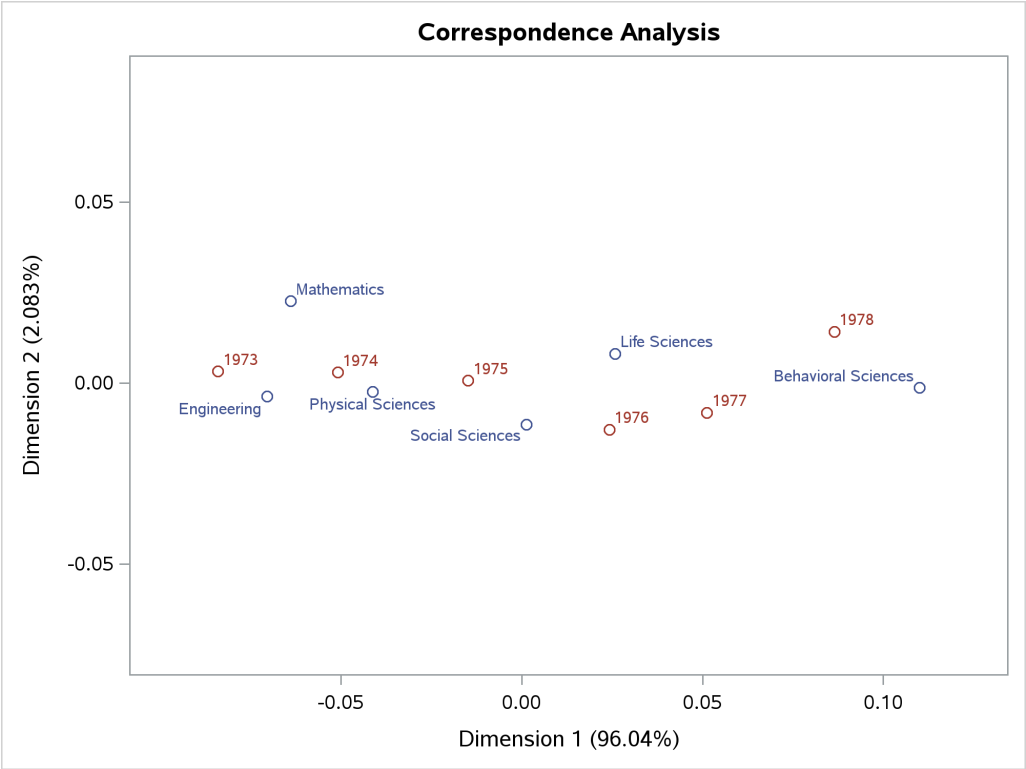

The results are as follows:

```
define statgraph Stat.Corresp.Graphics.Configuration;
   dynamic xVar yVar head legend _byline_ _bytitle_ _byfootnote_;
   begingraph;
      entrytitle HEAD;
      layout overlayequated / equatetype=fit xaxisopts=(offsetmin=0.1
         offsetmax=0.1) yaxisopts=(offsetmin=0.1 offsetmax=0.1);
         scatterplot y=YVAR x=XVAR / group=GROUP index=INDEX datalabel=LABEL
            name="Type" tip=(y x datalabel group) tiplabel=(group="Point");
         if (LEGEND)
            discretelegend "Type";
         endif;
      endlayout;
      if (_BYTITLE_)
         entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
      else
         if (_BYFOOTNOTE_)
            entryfootnote halign=left _BYLINE_;
         endif;
      endif;
   endgraph;
end;
```

You can add reference lines to the scatter plot at specified X and Y values by using the REFERENCELINE statement, as in the following example:

```
proc template;
   define statgraph Stat.Corresp.Graphics.Configuration;
      dynamic xVar yVar head legend _byline_ _bytitle_ _byfootnote_;
      begingraph;
         entrytitle HEAD;
         layout overlayequated / equatetype=fit xaxisopts=(offsetmin=0.1
            offsetmax=0.1) yaxisopts=(offsetmin=0.1 offsetmax=0.1);

            referenceline x=0;
            referenceline y=0;

            scatterplot y=YVAR x=XVAR / group=GROUP index=INDEX datalabel=LABEL
               name="Type" tip=(y x datalabel group) tiplabel=(group="Point");
            if (LEGEND)
               discretelegend "Type";
            endif;
         endlayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
      endgraph;
   end;
run;

proc corresp data=PhD short;
   ods select configplot;
   var y1973-y1978;
   id Science;
run;
```

When you modify templates, it is important to note that the order of the statements within the LAYOUT OVERLAYEQUATED block (or more typically, the LAYOUT OVERLAY block) is significant. Here, the reference lines are added before the scatter plot so that the reference lines are drawn before the scatter plot. Consequently, labels and markers that coincide with the reference lines are drawn over the reference lines. The results, with reference lines, are displayed in Output 25.4.2.

Output 25.4.2 Scatter Plot with Reference Lines Added

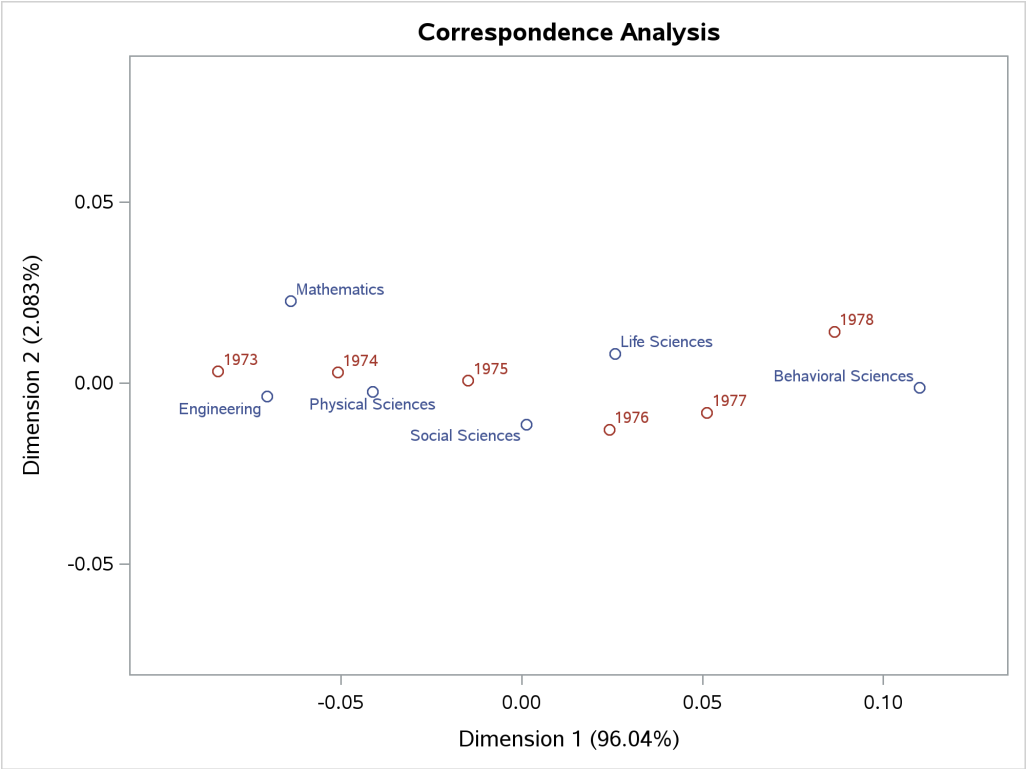

You can restore the default graph template as follows:

```
proc template;
   delete Stat.Corresp.Graphics.Configuration / store=sasuser.templat;
run;
```

The next steps show how you can change the style so that a frame is not shown:

```
proc template;
   define style noframe;
      parent=styles.htmlblue;
      style graphwalls from graphwalls / frameborder=off;
   end;
run;

ods listing style=noframe;

proc corresp data=PhD short;
   ods select configplot;
   var y1973-y1978;
   id Science;
run;
```

The results, shown in Output 25.4.3, display an X-axis and a Y-axis without a frame. Unlike the previous change, which affects only the ConfigPlot display, this change affects all plots created with the NOFRAME style.

Output 25.4.3 Scatter Plot with No Axis Frame

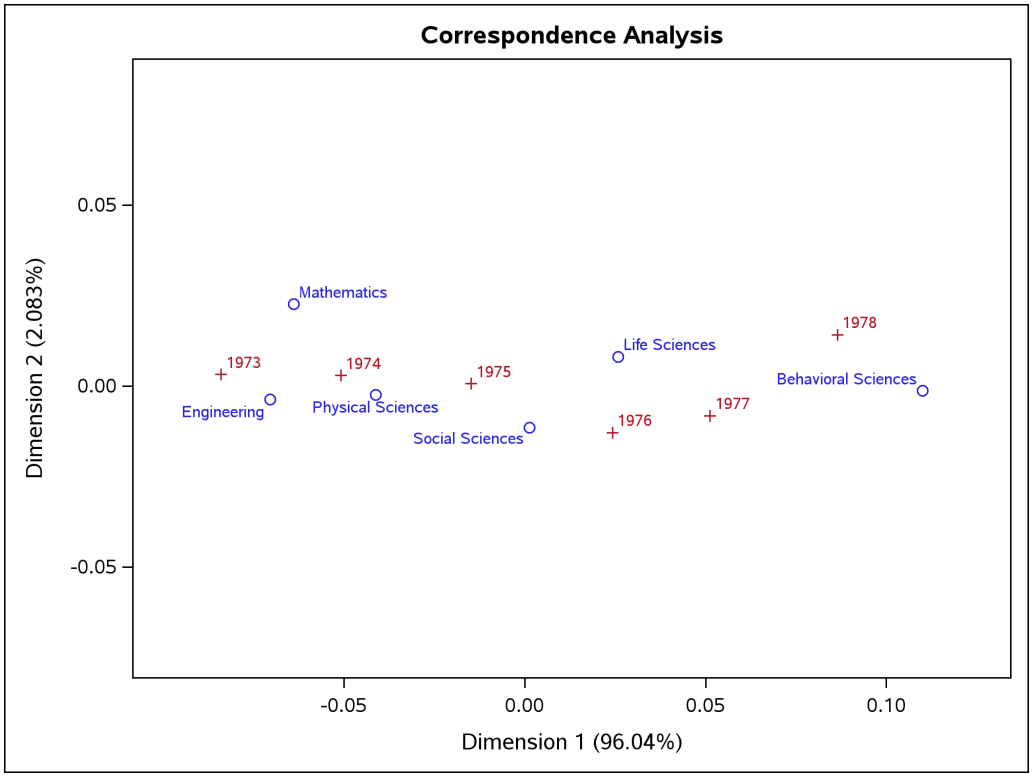

Alternatively, you can also add reference lines and delete the entire axis frame using the WALLDISPLAY=NONE and the DISPLAY= option in the graph template, as in the following example:

```
proc template;
   define statgraph Stat.Corresp.Graphics.Configuration;
      dynamic xVar yVar head legend _byline_ _bytitle_ _byfootnote_;
      begingraph;
         entrytitle HEAD;

         layout overlayequated / equatetype=fit walldisplay=none
            xaxisopts=(display=(tickvalues) offsetmin=0.1 offsetmax=0.1)
            yaxisopts=(display=(tickvalues) offsetmin=0.1 offsetmax=0.1);

            referenceline x=0;
            referenceline y=0;

            scatterplot y=YVAR x=XVAR / group=GROUP index=INDEX datalabel=LABEL
               name="Type" tip=(y x datalabel group) tiplabel=(group="Point");
            if (LEGEND)
               discretelegend "Type";
            endif;
         endlayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
```

```
            entryfootnote halign=left _BYLINE_;
         endif;
      endif;
   endgraph;
end;
run;

ods listing style=htmlblue;

proc corresp data=PhD short;
   ods select configplot;
   var y1973-y1978;
   id Science;
run;
```

The results are shown in Output 25.4.4.

Output 25.4.4 Scatter Plot with Internal Axes

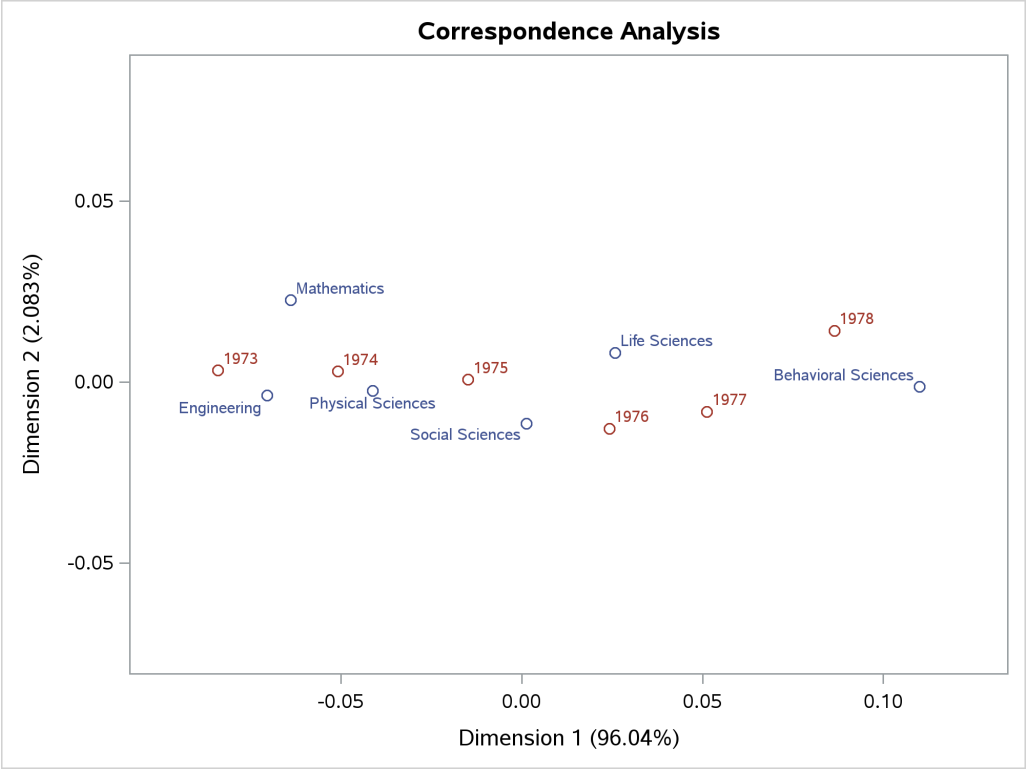

Instead of DISPLAY=(TICKVALUES), you can use DISPLAY=NONE (not shown) to remove the tick values from the display as well. You can change the tick values, as in the following example:

```
proc template;
   define statgraph Stat.Corresp.Graphics.Configuration;
      dynamic xVar yVar head legend _byline_ _bytitle_ _byfootnote_;
      begingraph;
         entrytitle HEAD;

         layout overlayequated / equatetype=fit
```

```
            commonaxisopts=(tickvaluelist=(0))
            xaxisopts=(offsetmin=0.1 offsetmax=0.1)
            yaxisopts=(offsetmin=0.1 offsetmax=0.1);


            referenceline x=0;
            referenceline y=0;

            scatterplot y=YVAR x=XVAR / group=GROUP index=INDEX datalabel=LABEL
               name="Type" tip=(y x datalabel group) tiplabel=(group="Point");
            if (LEGEND)
               discretelegend "Type";
            endif;
         endlayout;
         if (_BYTITLE_)
            entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
         else
            if (_BYFOOTNOTE_)
               entryfootnote halign=left _BYLINE_;
            endif;
         endif;
      endgraph;
   end;
run;

proc corresp data=PhD short;
   ods select configplot;
   var y1973-y1978;
   id Science;
run;
```

Since the axes in this plot are equated, the ticks are specified using the option **commonaxisopts = (tickvaluelist = (tick-value-list))**. This example only shows ticks at zero, but you can specify lists of values instead. The results are shown in Output 25.4.5.

Output 25.4.5 Scatter Plot with Tick Marks Specified

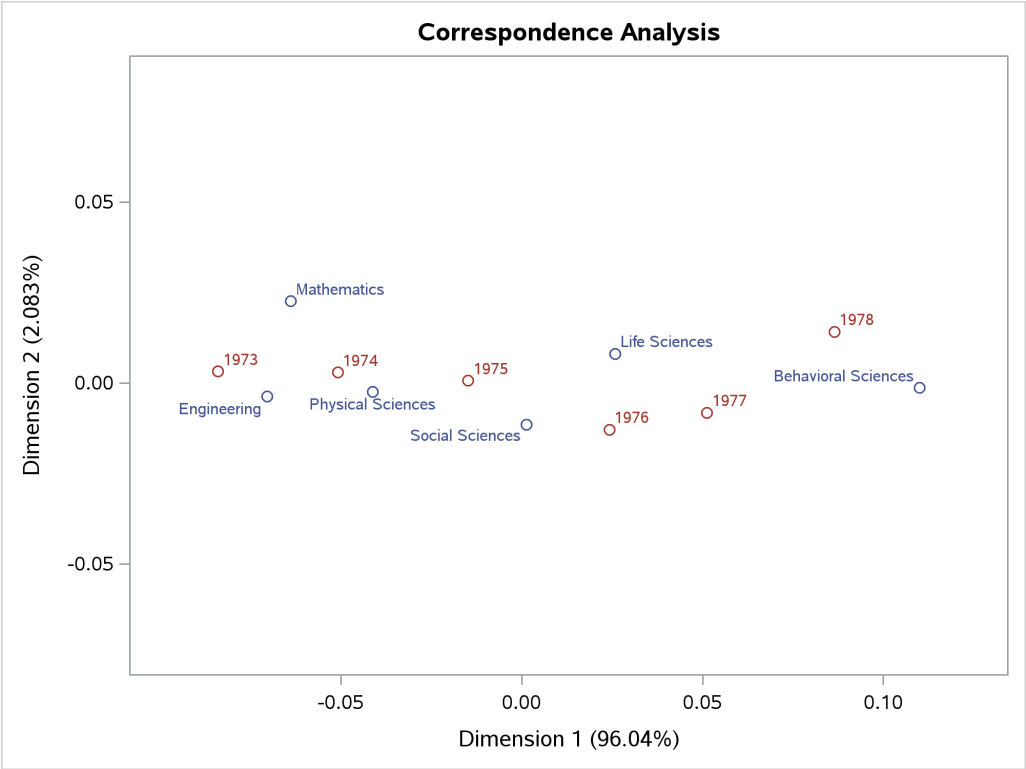

If the axes are not equated, then the tick value list is specified with the LINEAROPTS= option, as in the following statement:

```
layout overlay / xaxisopts=(linearopts=(viewmin=-0.1 viewmax=0.1
                                        tickvaluelist=(-0.1 0 0.1))
                               offsetmin=0.1 offsetmax=0.1)
                 yaxisopts=(linearopts=(viewmin=-0.1 viewmax=0.1
                                        tickvaluelist=(-0.1 0 0.1))
                               offsetmin=0.1 offsetmax=0.1);
```

The preceding statement uses the VIEWMIN= and VIEWMAX= options to specify the beginning and end of the data range that is shown. Specifying a tick value list does not extend or restrict the range of data shown in the plot. When axes share common options, it might be more convenient to use a macro to specify the options. The following two statements are equivalent to the preceding statement:

```
%let opts = linearopts=(viewmin=-0.1 viewmax=0.1
            tickvaluelist=(-0.1 0 0.1)) offsetmin=0.1 offsetmax=0.1;
layout overlay / xaxisopts=(&opts) yaxisopts=(&opts);
```

You can restore the default graph template as follows:

```
proc template;
   delete Stat.Corresp.Graphics.Configuration / store=sasuser.templat;
run;
```

Example 25.5: Adding Text to Every Graph

(View the complete code for this example (templex5.sas) in the example repository.)

This example shows how to add text to one or more graphs. For example, you can create a macro variable, with project and date information, as follows:

```
%let date = Project 17.104, &sysdate;
```

In order to add this information to a set of graphs, you need to first know the names of their templates. You can list the names of every graph template for SAS/STAT procedures or for a particular procedure as follows:

```
proc template;
   list stat     / where=(type='Statgraph');
   list stat.reg / where=(type='Statgraph');
run;
```

The results for PROC REG are shown in Output 25.5.1.

Output 25.5.1 PROC REG Templates

Listing of: SASHELP.TMPLSTAT
Path Filter is: Stat.Reg
Sort by: PATH/ASCENDING

Obs	Path	Type
1	Stat.Reg.Graphics.CooksD	Statgraph
2	Stat.Reg.Graphics.CooksDChart	Statgraph
3	Stat.Reg.Graphics.DFBETASPanel	Statgraph
4	Stat.Reg.Graphics.DFBETASPlot	Statgraph
5	Stat.Reg.Graphics.DFFITSPlot	Statgraph
6	Stat.Reg.Graphics.DiagnosticsPanel	Statgraph
7	Stat.Reg.Graphics.Fit	Statgraph
8	Stat.Reg.Graphics.FitHeatMap	Statgraph
9	Stat.Reg.Graphics.ObservedByPredicted	Statgraph
10	Stat.Reg.Graphics.PartialPanel	Statgraph
11	Stat.Reg.Graphics.PartialPlot	Statgraph
12	Stat.Reg.Graphics.PredictionPanel	Statgraph
13	Stat.Reg.Graphics.QQPlot	Statgraph
14	Stat.Reg.Graphics.RFPlot	Statgraph
15	Stat.Reg.Graphics.RStudentByPredicted	Statgraph
16	Stat.Reg.Graphics.ResidualBoxPlot	Statgraph
17	Stat.Reg.Graphics.ResidualByPredicted	Statgraph
18	Stat.Reg.Graphics.ResidualHeatMap	Statgraph
19	Stat.Reg.Graphics.ResidualHeatPanel	Statgraph
20	Stat.Reg.Graphics.ResidualHistogram	Statgraph
21	Stat.Reg.Graphics.ResidualPanel	Statgraph
22	Stat.Reg.Graphics.ResidualPlot	Statgraph
23	Stat.Reg.Graphics.RidgePanel	Statgraph
24	Stat.Reg.Graphics.RidgePlot	Statgraph
25	Stat.Reg.Graphics.SelectionCriterionPanel	Statgraph
26	Stat.Reg.Graphics.SelectionCriterionPlot	Statgraph
27	Stat.Reg.Graphics.StepSelectionCriterionPanel	Statgraph
28	Stat.Reg.Graphics.StepSelectionCriterionPlot	Statgraph
29	Stat.Reg.Graphics.StudResCooksDChart	Statgraph
30	Stat.Reg.Graphics.StudentResChart	Statgraph
31	Stat.Reg.Graphics.VIFPlot	Statgraph
32	Stat.Reg.Graphics.rstudentByLeverage	Statgraph

You can show the source for the graph templates for SAS/STAT procedures or for a particular procedure as follows:

```
options ls=96;
proc template;
   source stat     / where=(type='Statgraph');
   source stat.reg / where=(type='Statgraph');
options ls=80;
```

The results of this step are not shown. However, Example 25.3 shows a portion of the template for the PROC REG diagnostics panel. Here, the OPTIONS statement is used to set a line size of 96, which sometimes works better than the smaller default line size when showing the source for large and complicated templates.

An abridged version of the first few lines of the diagnostics panel template is displayed next:

```
define statgraph Stat.Reg.Graphics.DiagnosticsPanel;
   notes "Diagnostics Panel";
   dynamic . . .;
   BeginGraph / designheight=defaultDesignWidth;
      entrytitle halign=left textattrs=GRAPHVALUETEXT _MODELLABEL
         halign=center textattrs=GRAPHTITLETEXT "Fit Diagnostics"
         " for " _DEPNAME;
      . . .
```

Adding a Date and Project Stamp to a Few Graphs

You can add the project and date to the bottom of all graphs produced with PROC REG by putting a PROC TEMPLATE statement in front of the template source code, and adding an MVAR and ENTRYFOOTNOTE statement after every BEGINGRAPH statement, as in the following example:

```
proc template;
   define statgraph Stat.Reg.Graphics.DiagnosticsPanel;
      notes "Diagnostics Panel";
      dynamic . . .;
      BeginGraph / designheight=defaultDesignWidth;

         mvar date;
         entryfootnote halign=left textattrs=GraphValueText date;

         entrytitle halign=left textattrs=GRAPHVALUETEXT _MODELLABEL
            halign=center textattrs=GRAPHTITLETEXT "Fit Diagnostics"
            " for " _DEPNAME;
         . . .
```

The MVAR statement enables you to dynamically customize the template and graph at procedure run time, just as the DYNAMIC statement enables the procedure to dynamically customize the template and graph. With the MVAR statement, you can modify the template once and reuse that modification as the macro changes over time: Alternatively, you can modify the templates as follows:

```
entryfootnote halign=left textattrs=GRAPHVALUETEXT "&date";
```

However, you would then have to resubmit your templates every time the macro variable changed. The substitution for the macro variable date occurs at different times in the two preceding cases. In the former case, ODS looks for the value of the macro variable date at the time the template is used, and then the current date variable is used to set the text in the ENTRYFOOTNOTE statement, every time the template is used. In the latter case, SAS substitutes the value of the macro variable once, at the time that the PROC TEMPLATE step is executed.

The following steps use the Class data set and produce Output 25.5.2:

```
ods graphics on;

proc reg data=sashelp.class plots=fit(stats=none);
   model weight = height;
quit;
```

Output 25.5.2 PROC REG Plots with Project and Date Stamp

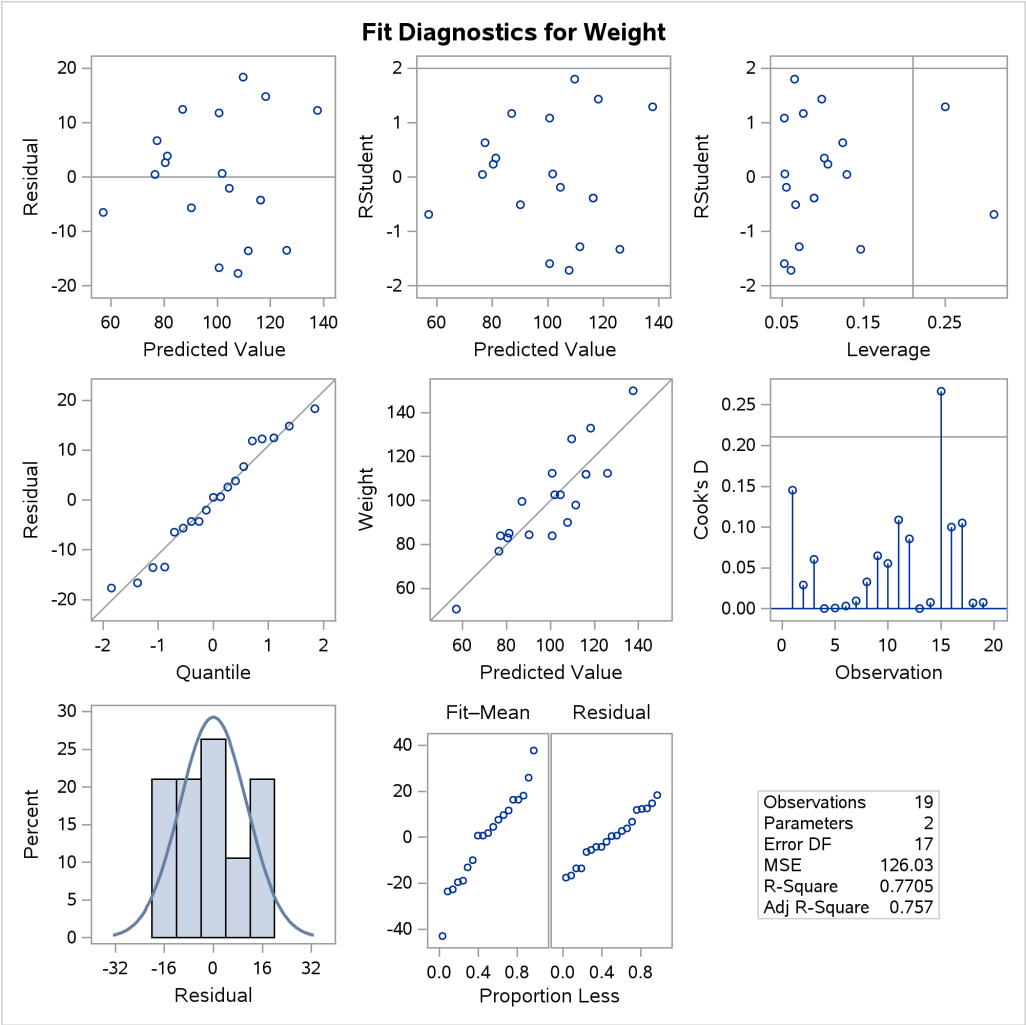

Output 25.5.2 *continued*

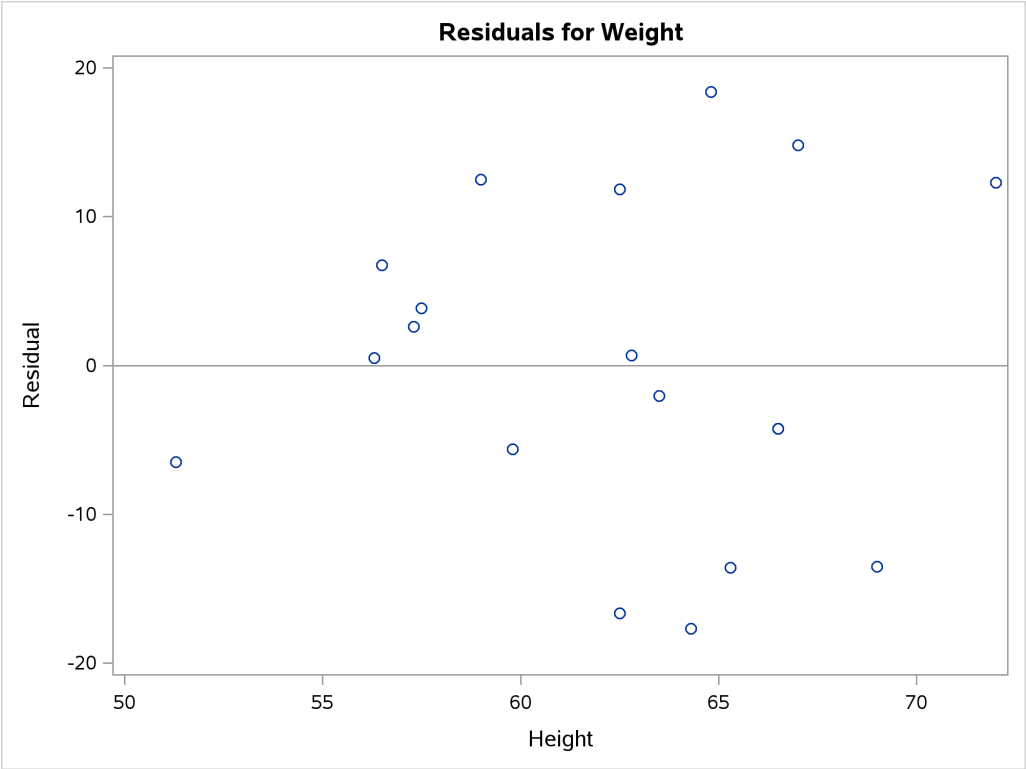

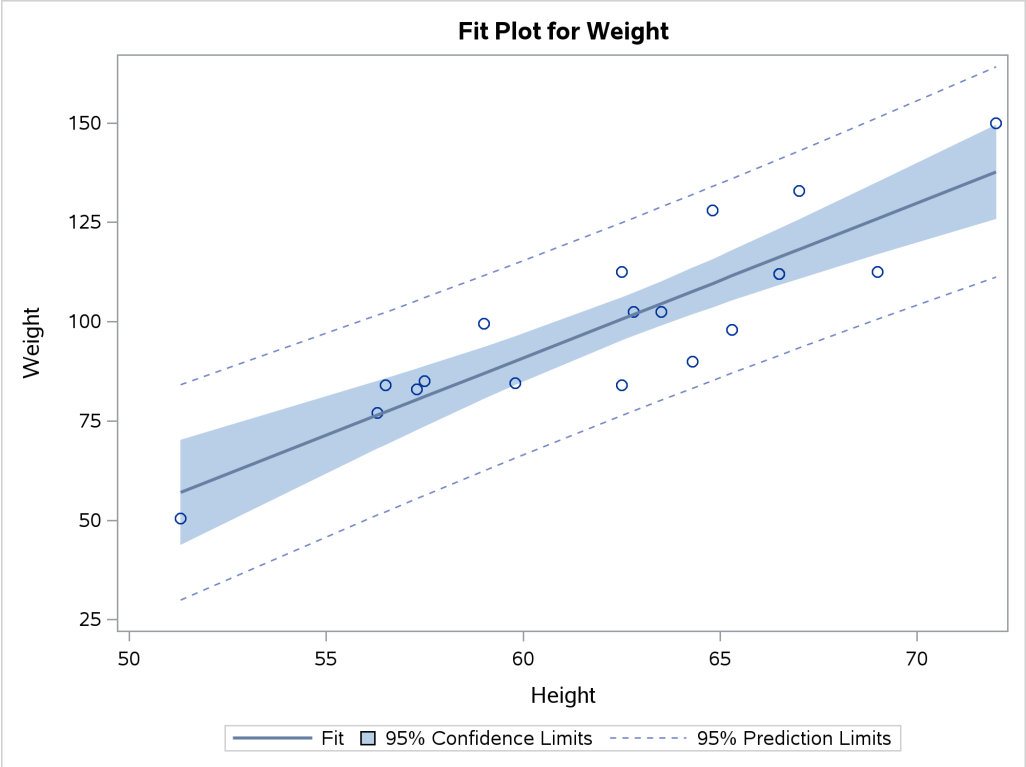

You can restore all of the default templates for PROC REG by running the following step:

```
proc template;
   delete stat.reg / store=sasuser.templat;
run;
```

Alternatively, you can specify **delete stat** to restore all SAS/STAT templates to their default definitions.

You can add text to the top or the bottom of a graph by using the ENTRYTITLE or the ENTRYFOOTNOTE statement, respectively. With both statements, you can put the text in the HALIGN=RIGHT, HALIGN=LEFT, or HALIGN=CENTER positions. You can add text to titles even if they already have a centered title. For example, the ENTRYTITLE statement in the diagnostic panel has text on the left (which is conditionally displayed) and a centered title:

```
entrytitle halign=left textattrs=GraphValueText _MODELLABEL
   halign=center textattrs=GraphTitleText "Fit Diagnostics"
   " for " _DEPNAME;
```

The current title can be followed by HALIGN=RIGHT and more text.

Adding Data Set Information to a Graph

You might, for example, want to add text to a set of graphs that indicates the most recently created data set. The following example shows you how you can do this with the syslast macro variable:

```
%let data = &syslast;

. . .

mvar data;
entrytitle halign=left textattrs=GraphValueText "Data: " data
   halign=center textattrs=GraphTitleText "Fit Diagnostics"
   " for " _DEPNAME;

. . .
```

Of course, this only makes sense when you are analyzing the last data set created. Alternatively, you can incorporate the name of the data set in the title, as in the following example:

```
%let data = &syslast;

. . .

mvar data;
entrytitle halign=center textattrs=GraphTitleText
           "Fit Diagnostics for Data Set " data;

. . .
```

Adding a Date and Project Stamp to All Graphs

Sometimes, you can automate the process of template modification. For example, you can automatically add an MVAR and ENTRYFOOTNOTE statement to every graph template, as in the following example:

```
ods path sashelp.tmplmst(read);
proc datasets library=sasuser nolist;
   delete templat(memtype=itemstor);
run;
ods path sasuser.templat(update) sashelp.tmplmst(read);

proc template;
   source / where=(type='Statgraph') file="tpls.sas";
run;

data _null_;
   infile 'tpls.sas' lrecl=256 pad;
   input line $ 1-256;
   file 'newtpls.sas';
   put line;
   line = left(lowcase(line));
   if line =: 'begingraph' then
      put 'mvar __date;' /
          'entryfootnote halign=left textattrs=GraphValueText __date;';

   file log;
   if index(line, '__date') then
      put 'ERROR: Name __date already used.' / line;
   if index(line,'entryfootnote') and not index(line,'_byline_') then put line;
run;

proc template;
   %include 'newtpls.sas' / nosource;
run;
```

These statements write all ODS graph templates to a file,[1] read that file, and write out a new file with an MVAR and ENTRYFOOTNOTE statement added after every BEGINGRAPH statement. Then these new templates are compiled with PROC TEMPLATE. Most graphs do not have footnotes. Those that do will now have multiple footnotes. You might want to manually combine them or write a more complicated program to handle them. These steps also assume that the name __date is not used anywhere. However, the program does check this and also lists all ENTRYFOOTNOTE statements. Be careful to check the SAS log to ensure that all templates compile without error. Also, before using templates that are automatically modified, make sure your modifications are reasonable.

You can delete Sasuser.Templat and hence all modified templates (assuming the default template search path) as follows:

[1] You might need to change the file name here and elsewhere in this chapter depending on your default working directory, your operating system, and the method you use to run SAS. For more information about specifying file names and directories, see the *SAS Companion* documentation for your operating system.

```
ods path sashelp.tmplmst(read);
proc datasets library=sasuser nolist;
   delete templat(memtype=itemstor);
run;
ods path sasuser.templat(update) sashelp.tmplmst(read);
```

Example 25.6: PROC TEMPLATE Statement Order and Primary Plots

(View the complete code for this example (templex6.sas) in the example repository.)

This example uses artificial data to illustrate two basic principles of template writing: that statement order matters and that one of the plotting statements is the primary statement. The data are a sample from a bivariate normal distribution. A custom graph template and PROC SGRENDER are used to plot the data along with vectors and ellipses. The plot consists of four components: a scatterplot of the data; vectors whose end points come from other variables in the data set; ellipses whose parameters are specified in the template; and reference lines whose locations are specified in the template. Initially, thick lines are used to show what happens at the places where the lines and points intersect.

The following steps create the input SAS data set:

```
data x;
   input x y;
   label x = 'Normal(0, 4)' y = 'Normal(0, 1)';
   datalines;
-4  0
 4  0
 0 -2
 0  2
;

data y(drop=i);
   do i = 1 to 2500;
      r1 = normal( 104 );
      r2 = normal( 104 ) * 2;
      output;
   end;
run;

data all;
   merge x y;
run;
```

The data set All contains four variables. The variables r1 and r2 contain the random data. These variables contain 2500 nonmissing observations. The data set also contains the variables x and y, which contain the end points for the vectors. These variables contain four nonmissing observations and 2496 observations that are all missing. A data set like this is not unusual when creating overlaid plots. Different overlays often require input data with very different sizes. First, the data are plotted by using a template that is deliberately constructed to demonstrate a number of problems that can occur with statement order.

The following steps create Output 25.6.1:

```
proc template;
   define statgraph Plot;
      begingraph;
         entrytitle 'Statement Order and the PRIMARY= Option';
         layout overlayequated / equatetype=fit;
            ellipseparm semimajor=eval(sqrt(4)) semiminor=1
                        slope=0 xorigin=0 yorigin=0 /
                        outlineattrs=GraphData2(pattern=solid thickness=5);
            ellipseparm semimajor=eval(2 * sqrt(4)) semiminor=2
                        slope=0 xorigin=0 yorigin=0 /
                        outlineattrs=GraphData5(pattern=solid thickness=5);
            vectorplot  y=y x=x xorigin=0 yorigin=0 /
                        arrowheads=false lineattrs=GraphFit(thickness=5);
            scatterplot y=r1 x=r2 /
                        markerattrs=(symbol=circlefilled size=3);
            referenceline x=0 / lineattrs=(thickness=3);
            referenceline y=0 / lineattrs=(thickness=3);
         endlayout;
      endgraph;
   end;
run;

ods listing style=listing;

proc sgrender data=all template=plot;
run;
```

Output 25.6.1 Statements Specified in a Nonoptimal Order

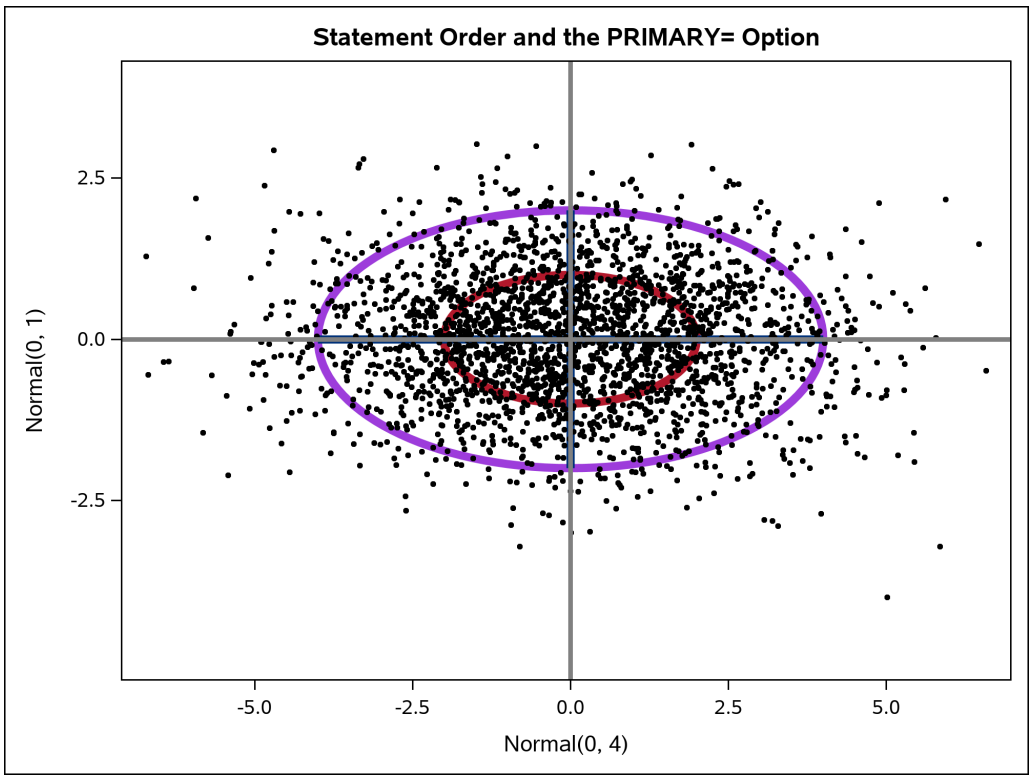

There are a number of problems with the plot in Output 25.6.1. The reference lines obliterate the vectors, and the data are on top of everything but the reference lines. It might be more reasonable to plot the reference lines first, the data next, the vectors next, and the ellipses last. The following steps do this and produce Output 25.6.2:

```
proc template;
   define statgraph Plot;
      begingraph;
         entrytitle 'Statement Order and the PRIMARY= Option';
         layout overlayequated / equatetype=fit;
            referenceline x=0 / lineattrs=(thickness=3);
            referenceline y=0 / lineattrs=(thickness=3);
            scatterplot y=r1 x=r2 /
                        markerattrs=(symbol=circlefilled size=3);
            vectorplot  y=y x=x xorigin=0 yorigin=0 /
                        arrowheads=false lineattrs=GraphFit(thickness=5);
            ellipseparm semimajor=eval(sqrt(4)) semiminor=1
                        slope=0 xorigin=0 yorigin=0 /
                        outlineattrs=GraphData2(pattern=solid thickness=5);
            ellipseparm semimajor=eval(2 * sqrt(4)) semiminor=2
                        slope=0 xorigin=0 yorigin=0 /
                        outlineattrs=GraphData5(pattern=solid thickness=5);
         endlayout;
      endgraph;
   end;
run;

ods listing style=listing;

proc sgrender data=all template=plot;
run;
```

Output 25.6.2 Statement Order Fixed

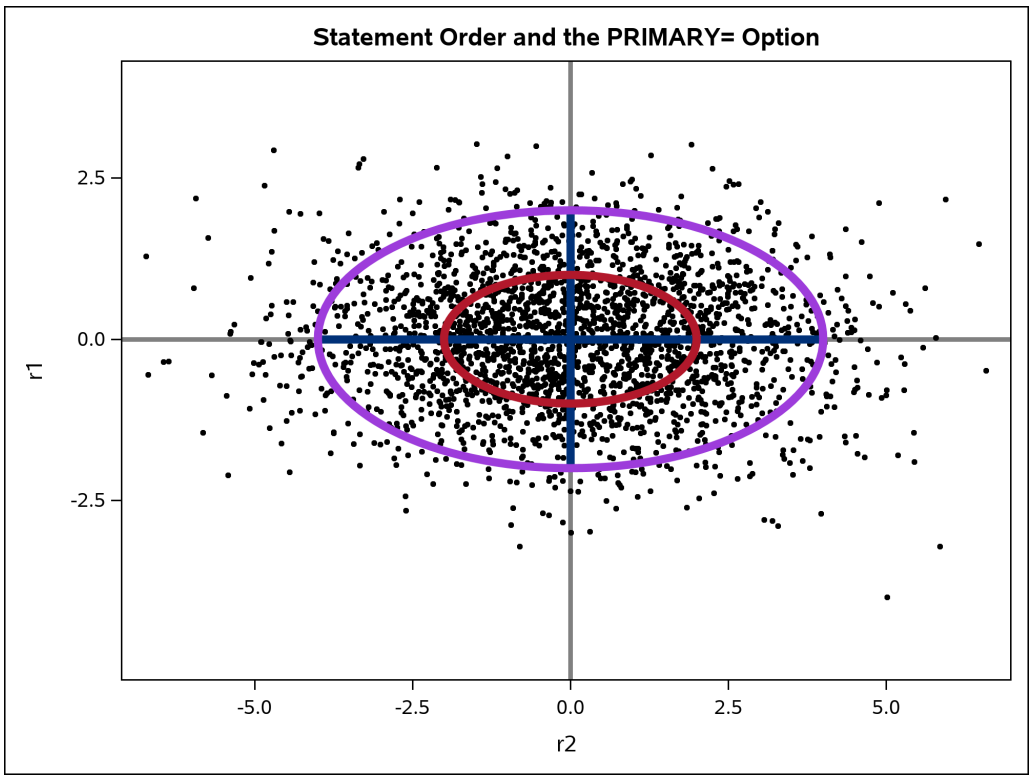

Output 25.6.2 looks better than Output 25.6.1, but the labels for the axes have changed. Output 25.6.1 has the labels of the variables x and y as axis labels, whereas Output 25.6.2 uses the names of the variables r1 and r2. This is because in the Output 25.6.1, the first plot is the vector plot of x and y (which have labels), and in Output 25.6.2, the first plot is the scatter plot of r1 and r2 (which do not have labels). By default, the first plot is the *primary plot*, and the primary plot is used to determine the axis type and labels. You can designate the vector plot as the primary plot with the PRIMARY=TRUE option.

The following statements make the final plot, this time with default line thicknesses, and produce Output 25.6.3:

```
proc template;
   define statgraph Plot;
      begingraph;
         entrytitle 'Statement Order and the PRIMARY= Option';
         layout overlayequated / equatetype=fit;
            referenceline x=0;
            referenceline y=0;
            scatterplot y=r1 x=r2 / markerattrs=(symbol=circlefilled size=3);

            vectorplot  y=y x=x xorigin=0 yorigin=0 / primary=true
                        arrowheads=false lineattrs=GraphFit;

            ellipseparm semimajor=eval(sqrt(4)) semiminor=1
                        slope=0 xorigin=0 yorigin=0 /
                        outlineattrs=GraphData2(pattern=solid);
            ellipseparm semimajor=eval(2 * sqrt(4)) semiminor=2
```

```
                        slope=0 xorigin=0 yorigin=0 /
                        outlineattrs=GraphData5(pattern=solid);
         endlayout;
      endgraph;
   end;
run;

ods listing style=listing;

proc sgrender data=all template=plot;
run;
```

Output 25.6.3 Statement Order Fixed and Primary Plot Specified

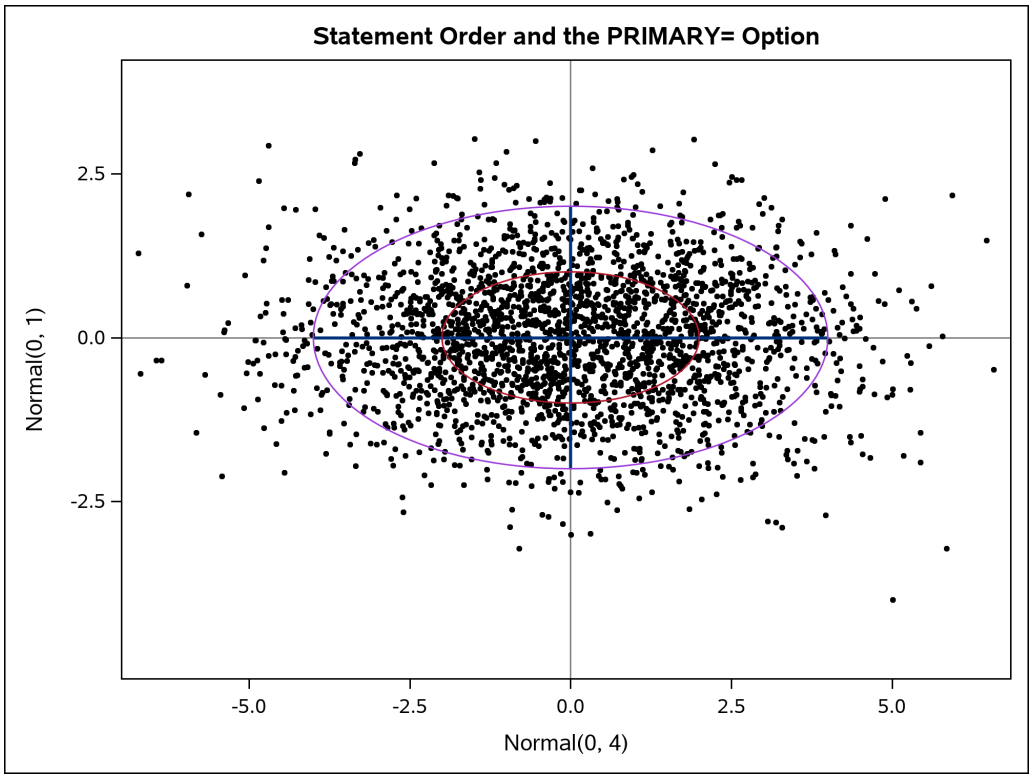

The axis labels in Output 25.6.3 and the overprinting of plot elements look better than in the previous plots. You can further adjust the line thicknesses if you want to emphasize or deemphasize components of this plot. The following list discusses the syntax of the GTL statements used in this example.

- The template has an ENTRYTITLE statement that specifies the title.
- The template has an equated overlay. This means that a centimeter on one axis represents the same data range as a centimeter on the other axis. This is done instead of the more common LAYOUT OVERLAY statement since with these data, the shape and geometry of the data have meaning even though the ranges of the two axis variables are different. The option EQUATETYPE=SQUARE is used to make a square plot, but since the X-axis variable has a larger range than the Y-axis variable, and since the default plot size is wider than high, EQUATETYPE=FIT is specified. The axes are equated but use the available space.

- A vertical reference line is drawn at X=0, and a horizontal reference line is drawn at Y=0.
- The scatter plot is based on the Y-axis variable r2 and the X-axis variable r1. The markers are filled circles with a size of three pixels. This is smaller than the default size and works well with a plot that displays many points.
- The vector plot is based on the Y-axis variable y and the X-axis variable x. The vectors are solid lines with no heads emanating from the origin (X=0 and Y=0). The color and other line attributes such as thickness come from the attributes of the **GraphFit** style element. This is the primary plot, so the default axis labels are the variable labels for the X= and Y= variables if they exist or the variable names if the variables do not have labels.
- The plot also displays two ellipses with X=0 and Y=0 at their center. Their widths are expressions, and their heights are constant. The expressions are not needed in this example; they are used to illustrate the syntax. The SEMIMAJOR= option specifies half the length of the major axis for the ellipse, and the SEMIMINOR= option specifies half the length of the minor axis for the ellipse. The SLOPE= option specifies the slope of the major axis for the ellipse. The colors of the ellipses and other line properties are based on the **GraphData2** and **GraphData5** style elements, but the line pattern attribute from the style is overridden.

Example 25.7: Marginal Model Plots

(View the complete code for this example (templex7.sas) in the example repository.)

Marginal model plots (proposed by Cook and Weisberg 1997 and discussed by Fox and Weisberg 2011) display the marginal relationship between the response and each predictor. Marginal model plots display the dependent variable on each vertical axis and each independent variable on a horizontal axis. There is one marginal model plot for each independent variable and one additional plot that displays the predicted values on the horizontal axis. Each plot contains a scatter plot of the two variables, a smooth fit function for the variables in the plot (labeled “Data”), and a function that displays the predicted values as a function of the horizontal axis variable (labeled “Model”). (See Output 25.7.1.) When the two functions are similar in each of the graphs, there is evidence that the model fits well. When the two functions differ in at least one of the graphs, there is evidence that the model does not fit well. The two functions correspond well for some independent variables and deviate for others largely because of the outlier, Pete Rose, the career hits leader.

Output 25.7.1 Marginal Model Plot for the 1986 Baseball Data

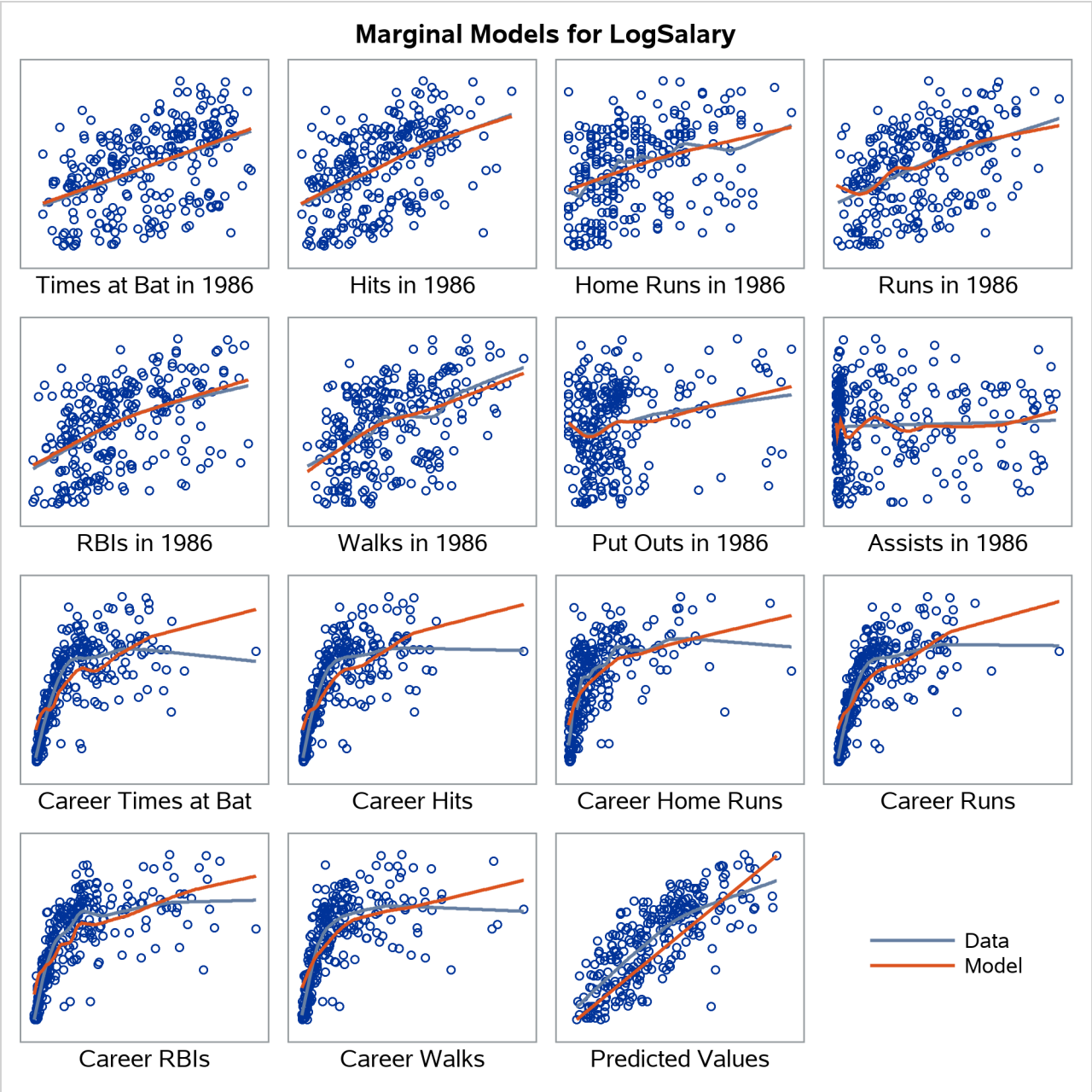

Marginal model plots provide useful diagnostic information for both linear and generalized linear models.You can use the **`%Marginal`** macro to produce marginal model plots after fitting a model. You must specify the dependent variable, the independent variables, and the variable that contains the predicted values. By default, the last SAS data set that is created is displayed, a loess fit function is used, and the macro chooses the number of graphs (from 2 to 15) to display in each panel and the size of the panel. Multiple panels are displayed when there are too many graphs to fit in a single panel. You can specify the number and size of the graphs in each panel rather than relying on the defaults.

The following steps illustrate the default settings:

```
%let vars =  nAtBat  nHits  nHome  nRuns  nRBI  nBB  nOuts nAssts
            CrAtBat CrHits CrHome CrRuns CrRbi CrBB;

proc glm noprint data=sashelp.baseball;
   class div;
   model logsalary = div &vars;
   output out=pvals p=p;
quit;

%marginal(independents=&vars, dependent=LogSalary, predicted=p)
```

Output 25.7.1 displays the results.

%Marginal Macro Options

You must specify the DEPENDENT=, INDEPENDENTS=, and PREDICTED= options. The remaining options have defaults and are not required. The macro provides the following options:

DATA=*SAS-data-set*
specifies the SAS data set to be displayed. If you do not specify this option, the **%Marginal** macro uses the most recently created SAS data set.

DEPENDENT=*variable*
specifies the dependent variable. You must specify this option.

GOPTS=*begingraph-statement-options*
specifies GTL BEGINGRAPH statement graph size options. The default size depends on the number of rows and columns in the panel.

Rows and Columns	Default GOPTS=
1×1, 2×2	DESIGNWIDTH=DEFAULTDESIGNHEIGHT
1×2	DESIGNHEIGHT=360PX
2×3	
3×4	DESIGNHEIGHT=520PX
3×3, 4×4, others	DESIGNHEIGHT=DEFAULTDESIGNWIDTH

INDEPENDENTS=*variable-list*
specifies the independent variables. You must specify this option.

PANEL=< *number-of-rows* > < *number-of-columns* >
specifies the number of rows and columns in the panel. Specify two values (rows then columns) or one value (when the number of rows equals the number of columns). For example, PANEL=2 3. Additional panels are automatically created when you have more graphs than available cells in the current panel. The default depends on both the number of independent variables and the number of graphs that can fit in a panel. For example, the default for 16 cells (14 independent variables, one predicted values plot, and a legend) is a 4×4 panel; the default for 17 cells (15 independent variables, one predicted values plot, and a legend) is two 3×3 panels (each panel has eight graphs and a legend that fills 18 cells). By default, the macro chooses to fill two panels that have fewer cells rather than to display more graphs in a first panel that has more cells and the last few graphs in the second panel. Multiple panels always have the same number of cells, although some cells might not be filled in the last panel. Defaults include 2×2, 2×3, 3×3, 3×4, and 4×4.

PREDICTED=*variable*
specifies the variable that contains the predicted values. You must specify this option.

SMOOTH=LOESS | PBSPLINE | REGRESSION
specifies the smoothing method.

- LOESS specifies a loess fit
- PBSPLINE specifies a penalized B-spline
- REGRESSION specifies a linear regression
- REGRESSION along with SMOOTHOPTS=DEGREE=3 specifies a cubic-polynomial regression.

By default, SMOOTH=LOESS.

SMOOTHOPTS=*smooth-options*
specifies GTL options for smoothing. These options are added to the GTL LOESSPLOT, PBSPLINEPLOT or REGRESSIONPLOT statement. By default, no options are specified.

%Marginal Macro Examples

The following steps show how to create displays that have varying numbers of graphs:

```
proc glm noprint data=sashelp.baseball;
   class div;
   model logsalary = div CrHome;
   output out=pvals p=p;
quit;

%marginal(independents=CrHome, dependent=LogSalary, predicted=p,
          panel=1 3, gopts=designheight=250px)

proc glm noprint data=sashelp.baseball;
   class div;
   model logsalary = div CrHome CrRuns;
   output out=pvals p=p;
quit;

%marginal(independents=CrHome CrRuns, dependent=LogSalary, predicted=p)

%let vars = nHits nHome nRuns CrHits CrHome CrRuns CrRbi;

proc glm noprint data=sashelp.baseball;
   class div;
   model logsalary = div &vars;
   output out=pvals p=p;
quit;

%marginal(independents=&vars, dependent=LogSalary, predicted=p)

proc glm noprint data=sashelp.baseball;
   class div;
   model logsalary = div CrAtBat CrHits CrHome;
   output out=pvals p=p;
```

```
quit;

%marginal(independents=CrAtBat CrHits CrHome,
          dependent=LogSalary, predicted=p, panel=1)
```

In the first model, the PANEL=1 3 and GOPTS=DESIGNHEIGHT=250PX options create a 1 × 3 display. The results are displayed in Output 25.7.2. In the second model, the default display for three graphs is 2 × 2. The results are displayed in Output 25.7.3. In the third model, the default display for eight graphs is 3 × 3. The results are displayed in Output 25.7.4. In the fourth model, the PANEL=1 option creates single graphs. The results are displayed in Output 25.7.5.

When you display graphs in panels, the legend occupies one of the cells. When you produce single graphs, the legend appears inside each graph.

Output 25.7.2 1 by 3 Display

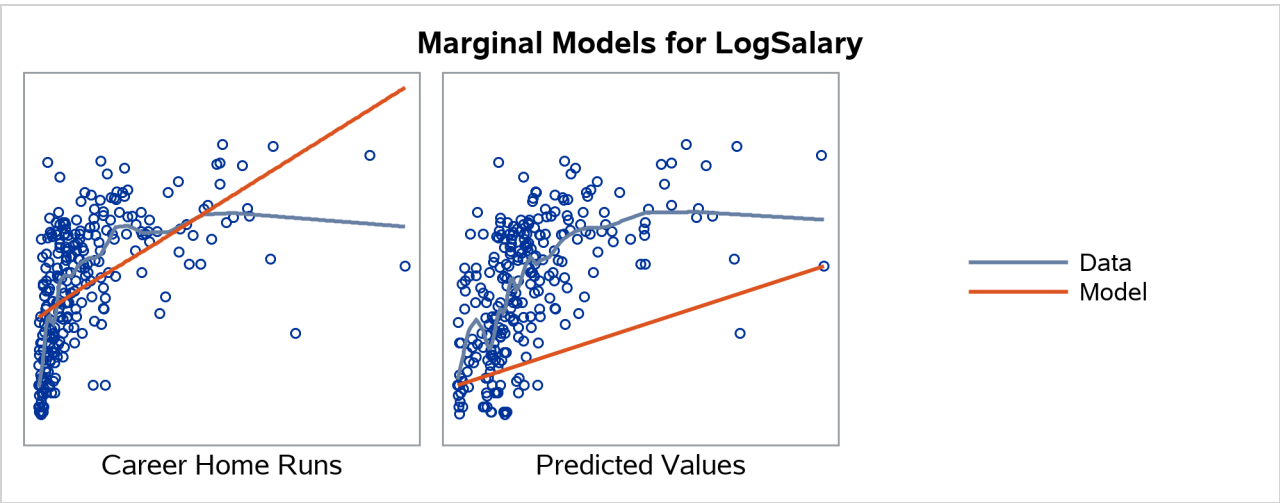

Output 25.7.3 2 by 2 Display

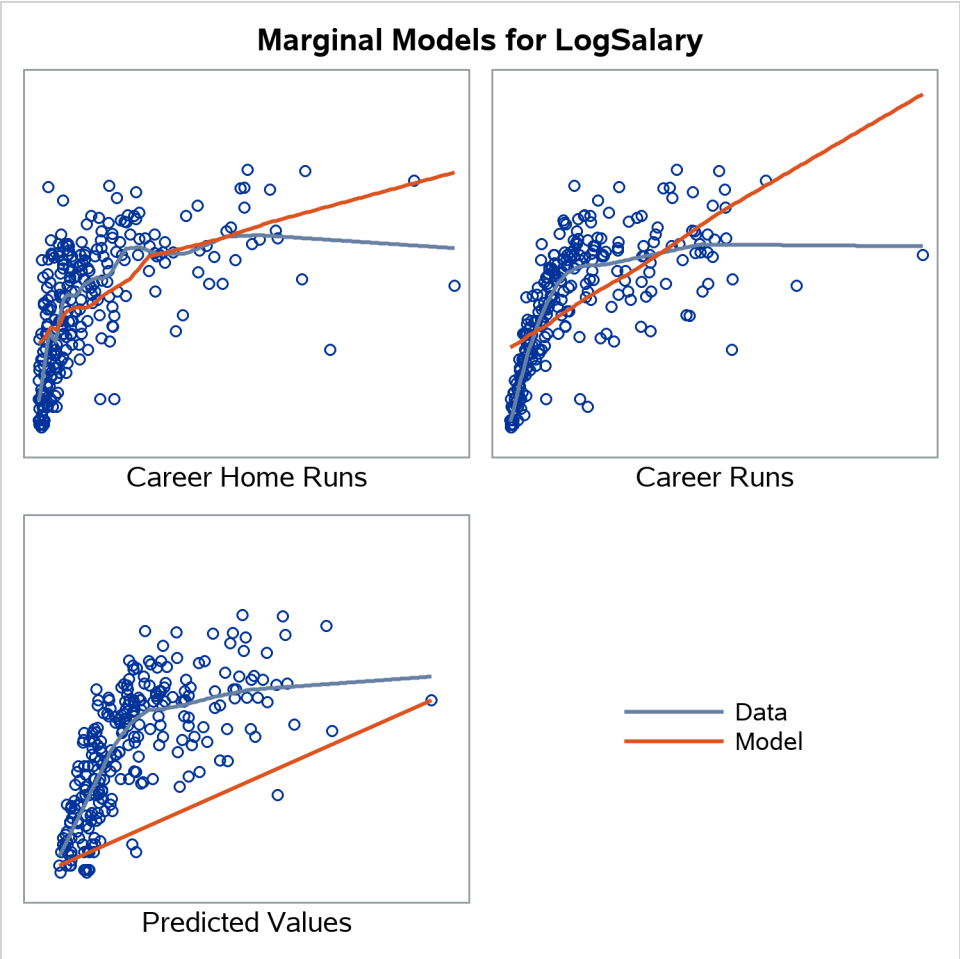

Output 25.7.4 3 by 3 Display

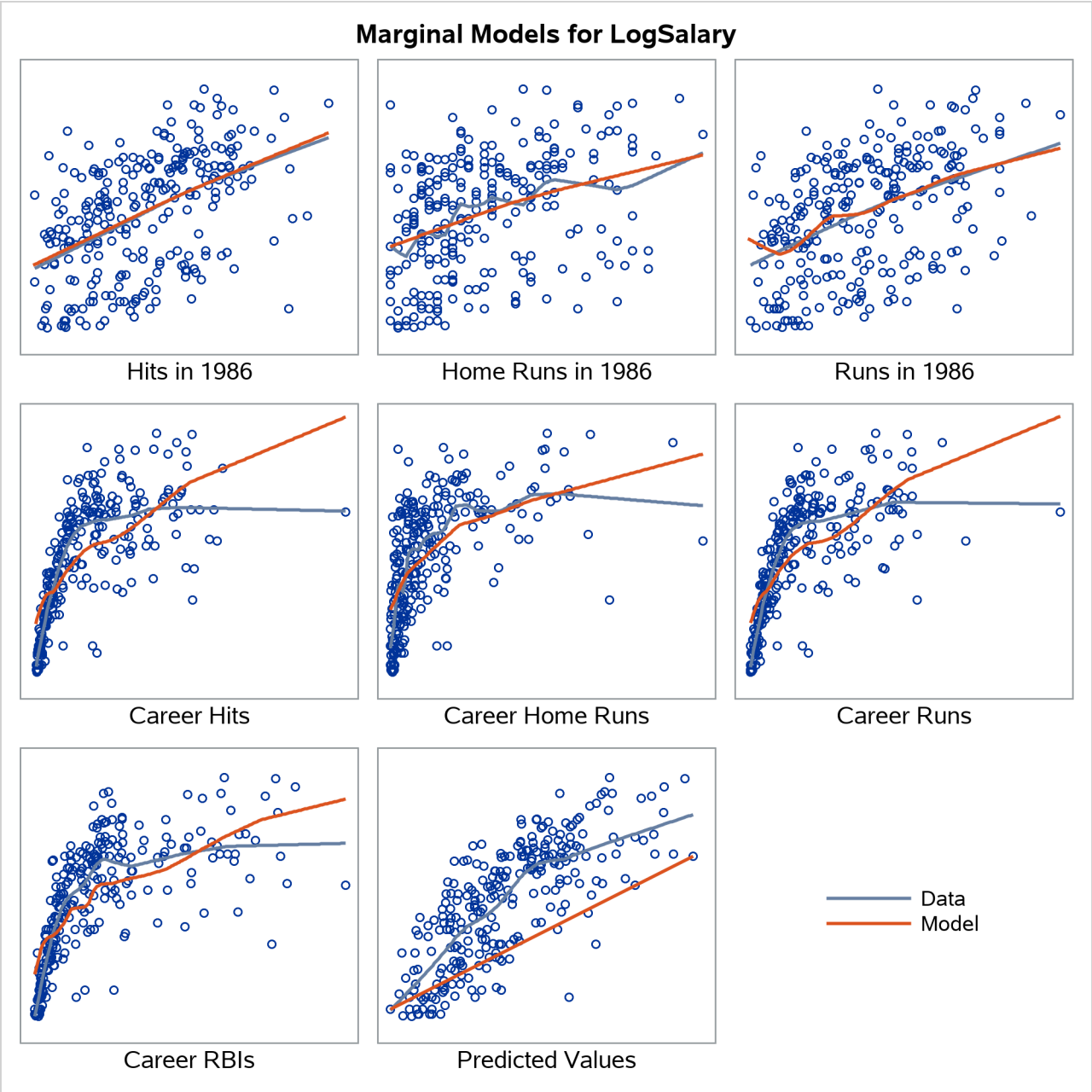

Output 25.7.5 Single-Graph Display

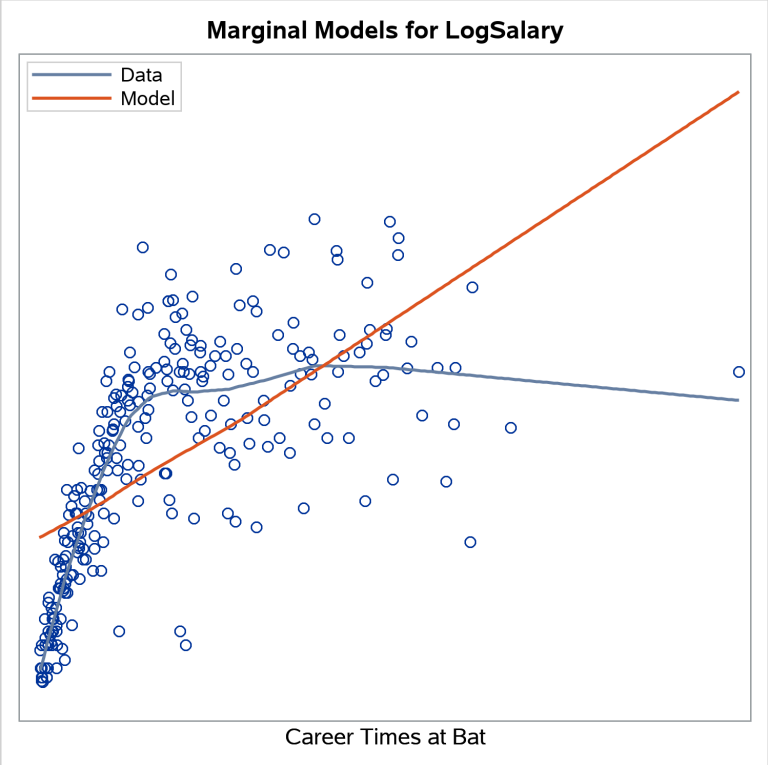

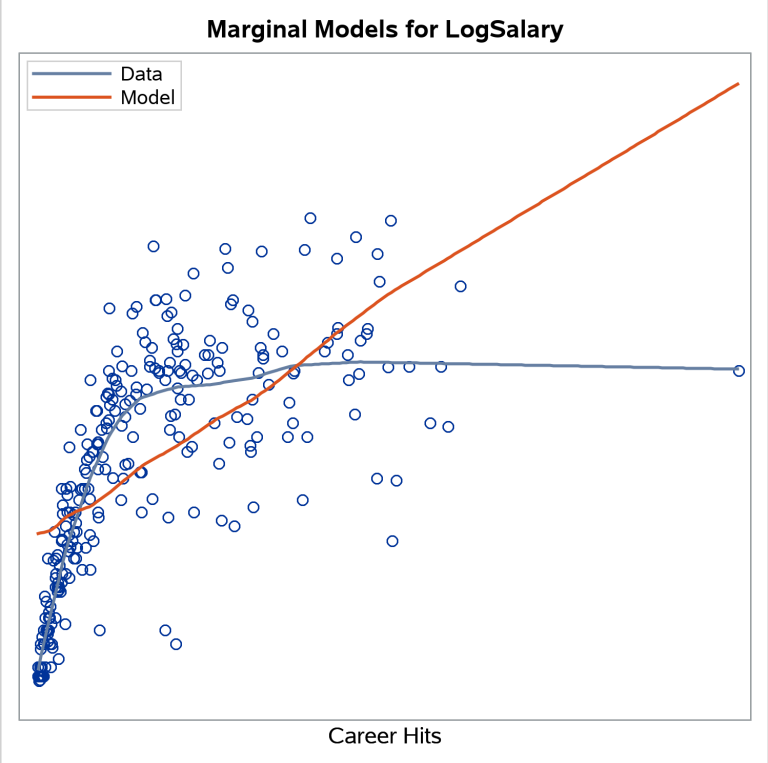

Output 25.7.5 *continued*

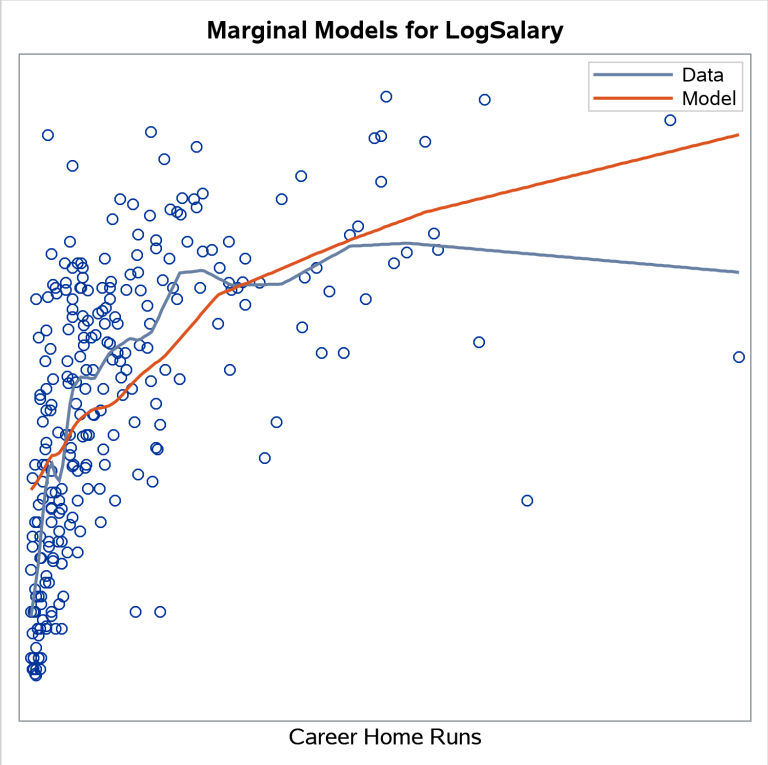

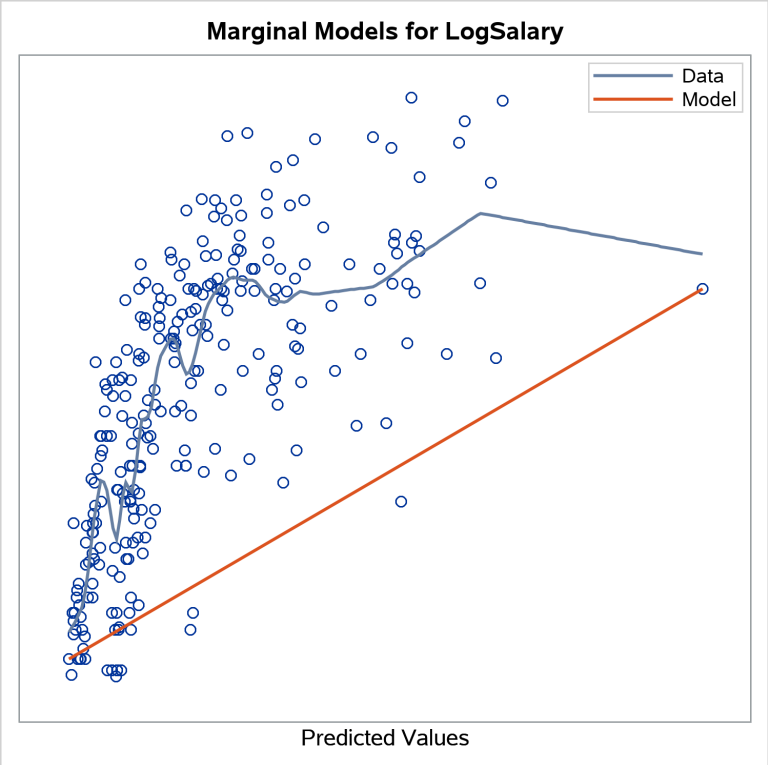

%Marginal Macro

You can use the **%Marginal** macro to create marginal model plots. The macro begins by initializing some macro variables and by checking the options. The macro gathers variable names, checks them, and stores them in the order that you specify. The macro determines the number of rows and columns in the panel along with the size. Next, it determines the endpoints for the predicted values plot. The macro constructs a custom graph template, and then one or more graphs are made using the template. The macro code follows.

```
                        /*----------------------------------------------*/
%macro marginal(        /*-------------Marginal Model Plots-------------*/
                        /*----------------------------------------------*/
       data=_last_,     /* Input SAS data set.                          */
       dependent=,      /* Dependent variable.                          */
       independents=,   /* Continuous independent variables.            */
       predicted=,      /* Predicted values variable.                   */
       smooth=loess,    /* LOESS, PBSPLINE, or REGRESSION.              */
       smoothopts=,     /* GTL options for smoothing.  They are added   */
                        /* to the GTL LOESSPLOT, PBSPLINEPLOT, or       */
                        /* REGRESSIONPLOT statement.                    */
       panel=,          /* Number of rows and columns in the panel.     */
                        /* The default depends on the number of         */
                        /* independent variables.  Specify two values   */
                        /* (rows columns) or one when rows equal        */
                        /* columns.  For example: PANEL=2 3.            */
       gopts=,          /* GTL BEGINGRAPH statement graph size options. */
                        /* The default depends on the number of         */
                        /* rows and columns in the panel.               */
                        /* Example: DESIGNHEIGHT=800px.                 */
       );;              /*----------------------------------------------*/

%local abort holdlast nvars i ncells savenote time tmpl paneled rows cols ettl;

%let time     = %sysfunc(datetime());
%let abort    = 0;
%let tmpl     = 0;
%let holdlast = &syslast;
%let savenote = %sysfunc(getoption(notes));

options nonotes;

data _null_; /* basic option processing and checking */
   length l $ 67;
   call symputx('ettl', quote('Marginal Models for ' || symget('dependent')));
   l = symget('data');
   if l = ' ' or lowcase(l) = '_last_' then  /* default, last data set */
      call symputx('data', symget('syslast'));

   l = symget('panel');
   r = input(scan(l, 1, ' '), best12.);
   c = input(scan(l, 2, ' '), best12.);
   if nmiss(c) and n(r) then c = r;
   call symputx('rows', r);
   call symputx('cols', c);
```

```
   if n(r,c) eq 2 and
      not (r = round(r) and c = round(c) and r ge 1 and c ge 1) then do;
      put "ERROR: PANEL=&panel must specify positive integers.";
      call symputx('abort', 1);
   end;

   l = lowcase(symget('smooth'));
   if not (trim(l) in ('regression', 'loess', 'pbspline')) then do;
      put 'ERROR: Invalid SMOOTH= option.  Specify: '
          'REGRESSION, LOESS, or PBSPLINE.';
      call symputx('abort', 1);
   end;

   if symget('predicted') eq ' ' or symget('dependent') eq ' ' or
      symget('independents') eq ' ' then do;
      put 'ERROR: The PREDICTED=, DEPENDENT=, and INDEPENDENTS= '
         'options ' 'must all ' 'be specified.';
      call symputx('abort', 1);
   end;

   if _error_ then call symputx('abort', 1);
run;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

data _null_;  /* check the variable names,                          */
   set &data; /* some problematic specifications generate SAS errors */
   &predicted = mean(of &predicted &dependent &independents);
   &dependent = mean(of &predicted &dependent &independents);
   if _error_ then do;
      put 'ERROR: All variables must be numeric.';
      _error_ = 0;
      call symputx('abort', 1);
   end;
   stop;
run;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

data __tmpcon; /* reorder vars if necessary */
   length &independents 8;
   set &data(obs=1 keep=&independents);
   if _error_ then call symputx('abort', 1);
run;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

proc contents data=__tmpcon noprint out=__tmpcon(keep=name varnum);
run;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

proc sort data=__tmpcon out=__tmpcon(drop=varnum);
   by varnum;
```

```
run;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

data __tmpcon(keep=name); /* augment name list, determine panel size */
   length gopts $ 32;
   set __tmpcon nobs=n;

   output;

   if n = _n_ then do;
      call symputx('nvars', n + 1);

      r = input(symget('rows'), best12.);
      c = input(symget('cols'), best12.);
      if n(r,c) eq 0 then do;
         %let i = 8,9,10,16,17,18,19,20,21,30,31,32;
         select;
            when(n le 2)          do; r = 2; c = 2; end;
            when(n in (3,4))      do; r = 2; c = 3; end;
            when(n in (5,6,7,15)) do; r = 3; c = 3; end;
            when(n in (&i))       do; r = 3; c = 4; end;
            otherwise             do; r = 4; c = 4; end;
         end;
         call symputx('rows', r);
         call symputx('cols', c);
      end;
      call symputx('paneled', r * c gt 1);
      if symget('gopts') eq ' ' then do;
         select;
            when(r eq 1 and c eq 1) gopts = 'designwidth=defaultdesignheight';
            when(r eq 1 and c eq 2) gopts = 'designheight=360px';
            when(r eq 2 and c eq 2) gopts = 'designwidth=defaultdesignheight';
            when(r eq 2 and c eq 3) gopts = ' ';
            when(r eq 3 and c eq 4) gopts = 'designheight=520px';
            otherwise               gopts = 'designheight=defaultdesignwidth';
         end;
         call symputx('gopts', gopts);
      end;
      name = ' '; output;
   end;

   if _error_ then call symputx('abort', 1);
run;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

data __tmpdat(keep=_x _y); /* get extremes for yhat series plot */
   set &data end=eof;
   retain _minx _maxx _miny _maxy .;
   if nmiss(_minx) then _minx = &predicted;
   if nmiss(_maxx) then _maxx = &predicted;
   if n(&predicted,_minx,&dependent) = 3 and &predicted lt _minx then do;
      _minx = &predicted; _miny = &dependent;
```

```
   end;
   if n(&predicted,_maxx,&dependent) = 3 and &predicted gt _maxx then do;
      _maxx = &predicted; _maxy = &dependent;
   end;
   if eof then do;
      _x = _minx; _y = _miny; output;
      _x = _maxx; _y = _maxy; output;
   end;
   if _error_ then call symputx('abort', 1);
run;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

data __tmpdat;
   merge &data __tmpdat;
   if _error_ then call symputx('abort', 1);
run;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

%let tmpl = 1;
proc template;
   define statgraph __marginal;
      dynamic %do i = 1 %to &rows*&cols - &paneled; _ivar&i %end; ncells pplot;
      begingraph / &gopts;
         entrytitle &ettl;
         legenditem type=line name='a' / lineattrs=GraphFit  label='Data';
         legenditem type=line name='b' / lineattrs=GraphFit2 label='Model';
         layout lattice / columns=&cols rows=&rows
                rowdatarange=unionall rowgutter=10 columngutter=10;
            %do i = 1 %to &rows * &cols - &paneled; /* ordinary cells */
               if(&i le ncells)
                  layout overlay / yaxisopts=(display=none)
                                   xaxisopts=(display=(label));
                     scatterplot  y=&dependent  x=_ivar&i;
                     &smooth.plot y=&dependent  x=_ivar&i / &smoothopts;
                     &smooth.plot y=&predicted  x=_ivar&i / &smoothopts
                                                            lineattrs=graphfit2;
                     %if not &paneled %then %do; /* not paneled?           */
                        layout gridded /         /* then put legend inside */
                           autoalign=(topright topleft bottomright bottomleft);
                           discretelegend 'a' 'b' / location=inside across=1;
                        endlayout;
                     %end;
                  endlayout;
               endif;
            %end;
            if(pplot) /* predicted values plot is handled differently */
               layout overlay / yaxisopts=(display=none)
                      xaxisopts=(display=(label) label='Predicted Values');
                  scatterplot  y=&dependent  x=&predicted;
                  &smooth.plot y=&dependent  x=&predicted / &smoothopts;
                  seriesplot   y=_y x=_x / lineattrs=graphfit2;
                  %if not &paneled %then %do;
```

```
                    layout gridded /
                           autoalign=(topright topleft bottomright bottomleft);
                       discretelegend 'a' 'b' / location=inside across=1;
                    endlayout;
                 %end;
              endlayout;
           endif;
           %if &paneled %then %do; /* legend in a cell by itself if paneled */
              layout overlay / yaxisopts=(display=none)
                               xaxisopts=(display=none);
                 discretelegend 'a' 'b' / location=inside across=1
                                          border=false;
              endlayout;
           %end;
        endlayout;
     endgraph;
  end;
run; quit;

%if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;

%let i = 1;
%do %while(&i le &nvars); /* create as many panels as is necessary */
   data _null_;
      set __tmpcon(firstobs=&i);
      length l $ 2000;
      retain l 'dynamic';
      if name ne ' ' then /* construct independent var list */
         l = catx(' ', l, '_ivar' || put(_n_, 3. -L), '=', quote(trim(name)));
      if name eq ' ' or _n_ eq (&rows * &cols - &paneled) then do;
         call symputx('pplot', name eq ' '); /* yhat plot in this panel? */
         call symputx('dynamics', l);
         call symputx('i', &i + _n_);
         call symputx('ncells', _n_ - (name = ' '));
         if _error_ then call symputx('abort', 1);
         stop;
      end;
      if _error_ then call symputx('abort', 1);
   run;

   %if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;


   proc sgrender data=__tmpdat template=__marginal;
      &dynamics ncells=&ncells pplot=&pplot;
   run;

   %if &syserr > 4 %then %let abort = 1; %if &abort %then %goto endit;
%end;

%endit:

proc datasets nolist; delete __tmpcon __tmpdat; run; quit;
```

```
%if &syserr > 4 %then %let abort = 1;

%if &tmpl and not %symexist(tmplsave) %then %do;
   proc template; delete __marginal; run; quit;
   %if &syserr > 4 %then %let abort = 1;
%end;

%let syslast = &holdlast;

options &savenote;

%if &abort %then %do;
   %if &syserr = 1016 or &syserr = 116  %then %put ERROR: Insufficient memory.;
   %if &syserr = 2000 or &syserr = 3000 %then
      %put ERROR: Syntax error.  Check your macro arguments for validity.;
   %put ERROR: The MARGINAL macro ended abnormally.;
%end;

%let time = %sysfunc(round(%sysevalf(%sysfunc(datetime()) - &time), 0.01));
%put NOTE: The MARGINAL macro used &time seconds.;

%mend;
```

Understanding Conditional Template Logic

You do not need to understand how the **%Marginal** macro works to use it. However, if you are interested, this section explains how this macro creates varying graph templates that depend on the number of variables and other options. This section can help you understand how you can use both GTL and macro conditional logic to construct flexible templates.

The **%Marginal** macro creates a graph template and then uses it one or more times to make graphs. The macro uses one block of code to create the template for all cases: single graphs, the default sizes (2×2, 2×3, 3×3, 3×4, and 4×4), and other sizes. It displays the legend inside a cell for multicell panels and inside the graph for single graphs. It constructs the final graph of predicted values in a different way from the way that it constructs the graphs that correspond to the independent variables. It does all of this in only 49 lines by using macro %DO loops and by using conditional logic (macro %IF statements and GTL IF and ENDIF statements) to populate cells. Before going into detail about the general logic, here are some specific examples.

For a 2×2 panel, the generated template is as follows:

```
proc template;
   define statgraph __marginal;
      dynamic _ivar1 _ivar2 _ivar3 ncells pplot;
      begingraph / designwidth=defaultdesignheight;
         entrytitle "Marginal Models for LogSalary";
         legenditem type=line name='a' / lineattrs=GRAPHFIT label='Data';
         legenditem type=line name='b' / lineattrs=GRAPHFIT2 label='Model';
         layout lattice / columns=2 rows=2 rowdatarange=unionall rowgutter=10
            columngutter=10;
            if (1 LE NCELLS)
               layout overlay / yaxisopts=(display=none)
                                xaxisopts=(display=(label));
```

```
                scatterplot y=LOGSALARY x=_IVAR1;
                loessplot y=LOGSALARY x=_IVAR1 /;
                loessplot y=P x=_IVAR1 / lineattrs=GRAPHFIT2;
             endlayout;
          endif;
          if (2 LE NCELLS)
             layout overlay / yaxisopts=(display=none)
                              xaxisopts=(display=(label));
                scatterplot y=LOGSALARY x=_IVAR2;
                loessplot y=LOGSALARY x=_IVAR2 /;
                loessplot y=P x=_IVAR2 / lineattrs=GRAPHFIT2;
             endlayout;
          endif;
          if (3 LE NCELLS)
             layout overlay / yaxisopts=(display=none)
                              xaxisopts=(display=(label));
                scatterplot y=LOGSALARY x=_IVAR3;
                loessplot y=LOGSALARY x=_IVAR3 /;
                loessplot y=P x=_IVAR3 / lineattrs=GRAPHFIT2;
             endlayout;
          endif;
          if (PPLOT)
             layout overlay / yaxisopts=(display=none)
                              xaxisopts=(display=(label)
                label='Predicted Values');
                scatterplot y=LOGSALARY x=P;
                loessplot y=LOGSALARY x=P /;
                seriesplot y=_Y x=_X / lineattrs=GRAPHFIT2;
             endlayout;
          endif;
          layout overlay / yaxisopts=(display=none)
                              xaxisopts=(display=none);
             discretelegend 'a' 'b' / location=inside across=1 border=false;
          endlayout;
       endlayout;
    endgraph;
 end;
quit;
```

The template can handle up to three independent variables (and no predicted values variable) or two independent variables along with one predicted values variable. (When there are three or more independent variables, the predicted values variable appears in a subsequent panel.) The following statements control which graphs are produced: IF (1 LE NCELLS), IF (2 LE NCELLS), IF (3 LE NCELLS), and IF (PPLOT).

This template can be used with the SGRENDER procedure and two independent variables as follows:

```
proc sgrender data=__tmpdat template=__marginal;
   dynamic _ivar1 = "nAtBat" _ivar2 = "nHits" ncells=2 pplot=1;
run;
```

Dynamic variables enable templates to be more flexible. The values of the dynamic variables are substituted into the template when PROC SGRENDER runs. The dynamic variables that begin with _ivar specify the names of the independent variables. Other dynamic variables specify that there are two independent variable cells (because ncells = 2, the third LAYOUT OVERLAY block is not used) and that the predicted values plot

is displayed (because pplot = 1, which is a true Boolean constant).

This template can be used with PROC SGRENDER and four independent variables as follows:

```
proc sgrender data=__tmpdat template=__marginal;
   dynamic _ivar1 = "nAtBat" _ivar2 = "nHits" _ivar3 = "nhome" ncells=3 pplot=0;
run;

proc sgrender data=__tmpdat template=__marginal;
   dynamic _ivar1 = "nrbi" ncells=1 pplot=1;
run;
```

When the dynamic variable ncells = 3, three independent variable plots are produced. When the dynamic variable pplot = 1, a predicted values plot is produced that follows the specified number of independent variable plots. The first step produces three independent variable plots, and the second step creates one independent variable plot and the predicted values plot. A legend appears in the cell after the last graph in both panels.

When you specify PANEL=1 to create single graphs, the generated template is as follows:

```
proc template;
   define statgraph __marginal;
      dynamic _ivar1 ncells pplot;
      begingraph / designwidth=defaultdesignheight;
         entrytitle "Marginal Models for LogSalary";
         legenditem type=line name='a' / lineattrs=GRAPHFIT label='Data';
         legenditem type=line name='b' / lineattrs=GRAPHFIT2 label='Model';
         layout lattice / columns=1 rows=1 rowdatarange=unionall
                          rowgutter=10 columngutter=10;
            if (1 LE NCELLS)
               layout overlay / yaxisopts=(display=none)
                                xaxisopts=(display=(label));
                  scatterplot y=LOGSALARY x=_IVAR1;
                  loessplot y=LOGSALARY x=_IVAR1 /;
                  loessplot y=P x=_IVAR1 / lineattrs=GRAPHFIT2;
                  layout gridded / autoalign=(topright topleft bottomright
                                              bottomleft);
                     discretelegend 'a' 'b' / location=inside across=1;
                  endlayout;
               endlayout;
            endif;
            if (PPLOT)
               layout overlay / yaxisopts=(display=none)
                       xaxisopts=(display=(label) label='Predicted Values');
                  scatterplot y=LOGSALARY x=P;
                  loessplot y=LOGSALARY x=P /;
                  seriesplot y=_Y x=_X / lineattrs=GRAPHFIT2;
                  layout gridded / autoalign=(topright topleft bottomright
                     bottomleft);
                     discretelegend 'a' 'b' / location=inside across=1;
                  endlayout;
               endlayout;
            endif;
         endlayout;
      endgraph;
```

```
   end;
quit;
```

Now, the legend appears inside each graph rather than in a separate cell. When you have two independent variables, the template is used in three PROC SGRENDER steps:

```
proc sgrender data=__tmpdat template=__marginal;
   dynamic _ivar1 = "nAtBat" ncells=1 pplot=0;
run;

proc sgrender data=__tmpdat template=__marginal;
   dynamic _ivar1 = "nHits" ncells=1 pplot=0;
run;

proc sgrender data=__tmpdat template=__marginal;
   dynamic ncells=0 pplot=1;
run;
```

When the dynamic variable ncells = 1, an independent variable plot is produced. When the dynamic variable pplot = 1, a predicted values plot is produced. The first two steps produce independent variable plots, and the third step creates a predicted values plot.

The part of the **%Marginal** macro that generates the template is as follows:

```
proc template;
   define statgraph __marginal;
      dynamic %do i = 1 %to &rows*&cols - &paneled; _ivar&i %end; ncells pplot;
      begingraph / &gopts;
         entrytitle &ettl;
         legenditem type=line name='a' / lineattrs=GraphFit  label='Data';
         legenditem type=line name='b' / lineattrs=GraphFit2 label='Model';
         layout lattice / columns=&cols rows=&rows
                rowdatarange=unionall rowgutter=10 columngutter=10;
            %do i = 1 %to &rows * &cols - &paneled; /* ordinary cells */
               if(&i le ncells)
                  layout overlay / yaxisopts=(display=none)
                                   xaxisopts=(display=(label));
                     scatterplot  y=&dependent  x=_ivar&i;
                     &smooth.plot y=&dependent  x=_ivar&i / &smoothopts;
                     &smooth.plot y=&predicted  x=_ivar&i / &smoothopts
                                                            lineattrs=graphfit2;
                     %if not &paneled %then %do; /* not paneled?          */
                        layout gridded /         /* then put legend inside */
                           autoalign=(topright topleft bottomright bottomleft);
                           discretelegend 'a' 'b' / location=inside across=1;
                           endlayout;
                        %end;
                     endlayout;
                  endif;
               %end;
            if(pplot) /* predicted values plot is handled differently */
               layout overlay / yaxisopts=(display=none)
                      xaxisopts=(display=(label) label='Predicted Values');
                  scatterplot  y=&dependent  x=&predicted;
                  &smooth.plot y=&dependent  x=&predicted / &smoothopts;
```

```
                    seriesplot   y=_y x=_x / lineattrs=graphfit2;
                    %if not &paneled %then %do;
                       layout gridded /
                              autoalign=(topright topleft bottomright bottomleft);
                          discretelegend 'a' 'b' / location=inside across=1;
                          endlayout;
                       %end;
                    endlayout;
                 endif;
              %if &paneled %then %do; /* legend in a cell by itself if paneled */
                 layout overlay / yaxisopts=(display=none)
                                  xaxisopts=(display=none);
                    discretelegend 'a' 'b' / location=inside across=1
                                             border=false;
                 endlayout;
              %end;
           endlayout;
        endgraph;
     end;
  quit;
```

The LAYOUT LATTICE statement in the GTL constructs a display of &rows and &cols (macro variables that are set in a preceding step). Inside the LAYOUT LATTICE block, there are LAYOUT OVERLAY blocks for the independent variables and for the predicted values, and optionally one for the legend. When the graph is a panel that contains multiple cells, the macro %DO loop generates &rows × &cols – 1 LAYOUT OVERLAY blocks for the independent variables. Additional code produces one LAYOUT OVERLAY block for the predicted values plot and a final LAYOUT OVERLAY block for the legend. This same template is used when there are multiple panels, and the LAYOUT OVERLAY block for the predicted values plot is suppressed in all but the last panel by the IF(PPLOT) statement.

When there is one cell per panel, two GTL IF statements control which type of graph is displayed. When there is one cell per panel, a macro %IF statement adds a legend inside the body of the plot, and a different macro %IF statement excludes the creation of a separate legend cell. When there are multiple cells per panel, the legend is added to the last cell and does not appear inside the graphs.

The macro language enables you to generate a long template by using a minimum number of statements. There is one LAYOUT OVERLAY block in the macro for independent variable plots. In contrast, the generated template can be much longer and has &rows × &cols – 1 LAYOUT OVERLAY blocks because of the macro %DO loop, each conditionally executed by a GTL IF statement. The template also contains two LAYOUT OVERLAY blocks that are outside the macro %DO loop. Although you would never know it by looking at the PROC TEMPLATE SOURCE statement template display, many of the multicell templates that SAS provides for use in its analytical procedures are created in a similar way by using macros and dynamic variables.

Example 25.8: Changing Titles by Using the %GrTitle Macro

(View the complete code for this example (templex8.sas) in the example repository.)

This example shows how to use a macro to change titles in graphs that are produced by analytical procedures. The **%GrTitle** macro modifies graph templates and makes it easy to replace one or more titles.

PROC LOGISTIC Example

The following steps run PROC LOGISTIC and create an analysis-of-means plot:

```
data Neuralgia;
   input Treatment $ Sex $ Age Duration Pain $ @@;
   datalines;
P F 68 1 No B M 74 16 No P F 67 30 No P M 66 26 Yes B F 67 28 No B F 77 16 No A
F 71 12 No B F 72 50 No B F 76 9 Yes A M 71 17 Yes A F 63 27 No A F 69 18 Yes B
F 66 12 No A M 62 42 No P F 64 1 Yes A F 64 17 No P M 74 4 No A F 72 25 No P M
70 1 Yes B M 66 19 No B M 59 29 No A F 64 30 No A M 70 28 No A M 69 1 No B F 78
1 No P M 83 1 Yes B F 69 42 No B M 75 30 Yes P M 77 29 Yes P F 79 20 Yes A M 70
12 No A F 69 12 No B F 65 14 No B M 70 1 No B M 67 23 No A M 76 25 Yes P M 78 12
Yes B M 77 1 Yes B F 69 24 No P M 66 4 Yes P F 65 29 No P M 60 26 Yes A M 78 15
Yes B M 75 21 Yes A F 67 11 No P F 72 27 No P F 70 13 Yes A M 75 6 Yes B F 65 7
No P F 68 27 Yes P M 68 11 Yes P M 67 17 Yes B M 70 22 No A M 65 15 No P F 67 1
Yes A M 67 10 No P F 72 11 Yes A F 74 1 No B M 80 21 Yes A F 69 3 No
;

ods trace on;
ods graphics on;
proc logistic data=Neuralgia;
   class Treatment Sex / param=glm;
   model Pain= Treatment|Sex Age;
   lsmeans Treatment / plots=anom;
run;
```

ODS displays the following trace output for the analysis-of-means plot:

```
Output Added:
-------------
Name:       AnomPlot
Label:      Treatment ANOM Plot
Template:   Stat.Logistic.Graphics.AnomPlot
Path:       Logistic.LSMeans.AnomPlot
-------------
```

The analysis-of-means plot is displayed in Output 25.8.1. Notice that it has two titles.

You can use the **%GrTitle** autocall macro to modify the way the graph template specifies titles:

```
%grtitle(path=Stat.Logistic.Graphics.AnomPlot)
```

The macro displays the following in the SAS log:

```
Stat.Logistic.Graphics.AnomPlot -> Stat.Graphics.AnomPlot
```

```
Graphics_AnomPlot
Graphics_AnomPlot2
```

Output 25.8.1 Analysis of Means

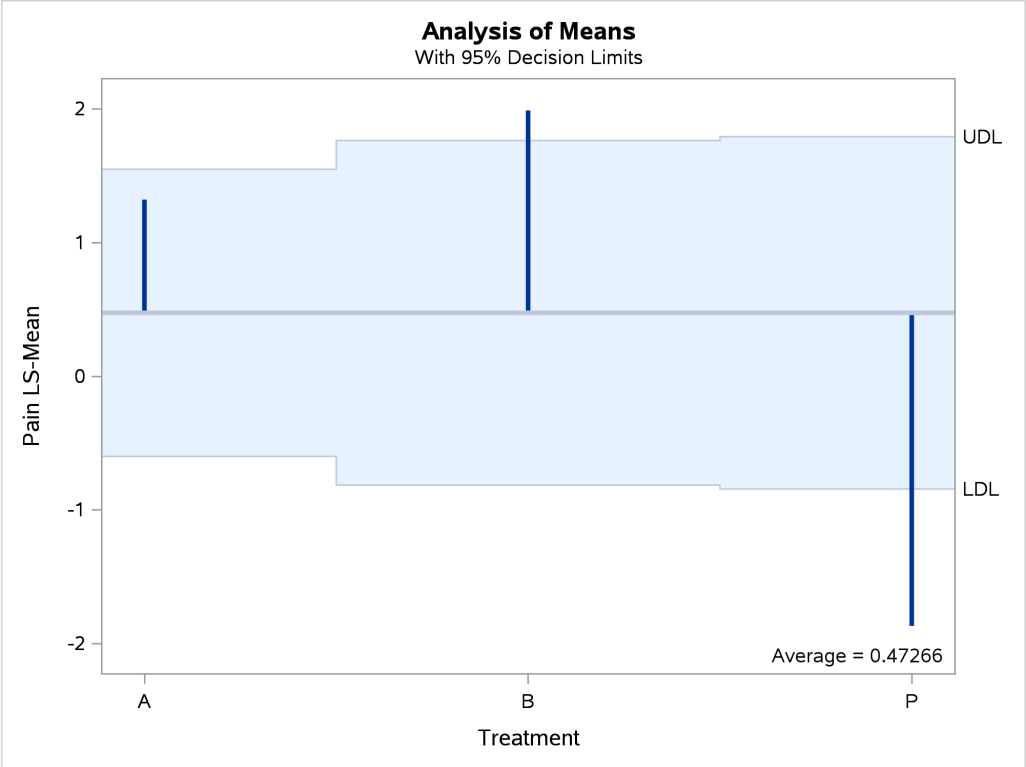

The macro begins by displaying the name of the template that you specified in the PATH= argument. In this case, two names are displayed because the specified template is a link. The macro follows links and modifies the actual graph template that is at the end of the link chain. The macro creates and displays the names of two macro variables (because there are two titles) after the template names. The modified template creates titles from the values of the macro variables when those variables exist; it uses the original titles when the macro variables do not exist. The names of the macro variables are constructed from the second and last levels of the template name. In many cases, these two levels are the procedure name and the graph name (but not in this case because of the link).

The following step creates the graph by using the modified template and the two macro variables:

```
%let Graphics_AnomPlot = Neuralgia Study;
%let Graphics_AnomPlot2 = Analysis of Means with 95% Decision Limits;
proc logistic data=Neuralgia;
   ods select anomplot;
   class Treatment Sex / param=glm;
   model Pain= Treatment|Sex Age;
   lsmeans Treatment / plots=anom;
run;
```

The modified graph is displayed in Output 25.8.2.

Output 25.8.2 Customized Titles

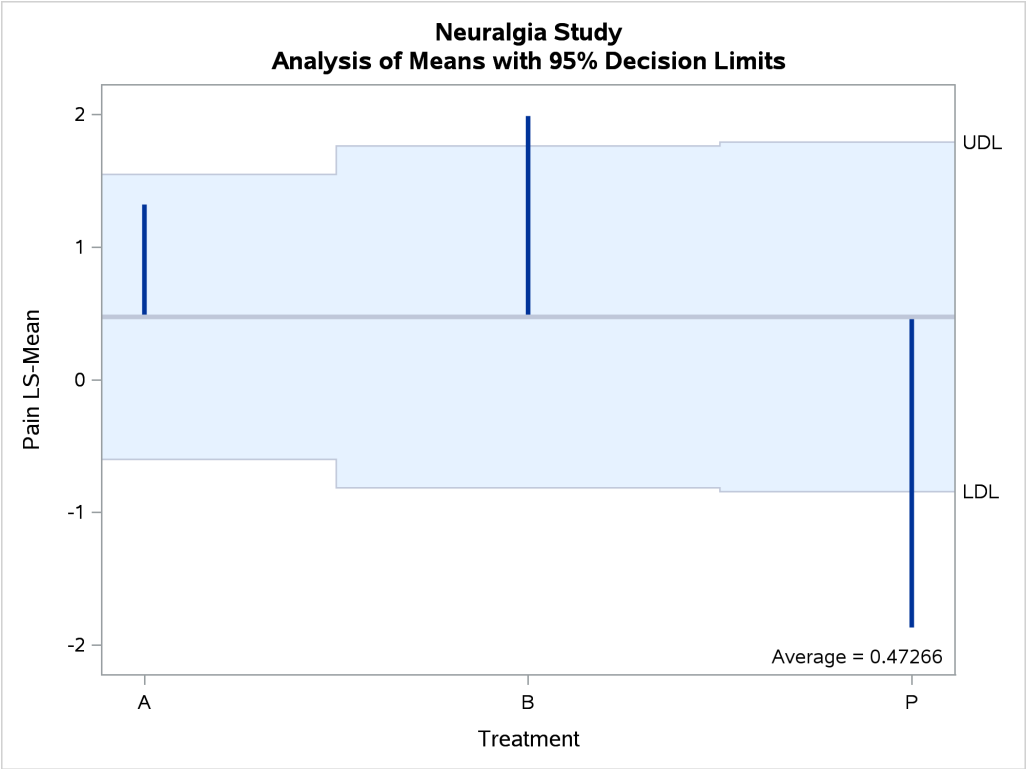

%GrTitle Macro Details

The **%GrTitle** macro does the following:

- The macro deletes all modified templates in the Work library.
- PROC TEMPLATE generates a list of each template that matches the specified path.
- PROC TEMPLATE writes each graph template to a file.
- DATA steps follow the links.
- A DATA step modifies the template code and submits it to SAS using CALL EXECUTE.

You can use OPTIONS=DISPLAY to display the ENTRYTITLE and NOTES statements:

```
%grtitle(path=Stat.Logistic.Graphics.AnomPlot, options=display)
```

In the analysis-of-means plot, the original template has two ENTRYTITLE statements:

```
entrytitle _TITLE;
entrytitle textattrs=GRAPHVALUETEXT _CLSTR;
```

Both statements rely on dynamic variables that PROC LOGISTIC sets and sends to the template. The macro modifies the template by replacing those two statements with the following:

```
mvar Graphics_AnomPlot Graphics_AnomPlot2;
if (EXISTS(GRAPHICS_ANOMPLOT))
   entrytitle GRAPHICS_ANOMPLOT;
else
   entrytitle _TITLE;
endif;
if (EXISTS(GRAPHICS_ANOMPLOT2))
   entrytitle GRAPHICS_ANOMPLOT2;
else
   entrytitle textattrs=GRAPHVALUETEXT _CLSTR;
endif;
```

The titles can come from either the new macro variables or the old dynamic variables. If a macro variable exists, then ODS uses the title that you specified in that macro variable. If a macro variable has a null value (`%LET Graphics_AnomPlot = ;`), that title is suppressed. Otherwise, ODS uses the original title. There are no ampersands in the modified template. ODS substitutes the macro variable values at the time that PROC LOGISTIC runs, not at the time that the macro runs PROC TEMPLATE.

The following step modifies another PROC LOGISTIC template:

```
%grtitle(path=Stat.Logistic.Graphics.PhatPanel, options=display)
```

The macro produces the following messages in the SAS log:

```
Stat.Logistic.Graphics.PhatPanel
   notes "Predicted Probability Plots";

   Logistic_PhatPanel
   entrytitle _TITLE;
```

The specified template is not a link, so one template name is displayed. The macro displays the NOTES statement, which provides information about the template's purpose. You can specify the macro variable Logistic_PhatPanel to modify the title.

The original template comes from one of the item stores in your ODS path (by default, either Sasuser.Templat or Sashelp.Tmplmst). The modified template is saved in the item store Work.Templat. You can delete that item store as follows:

```
%grtitle(options=delete)
```

You can modify multiple templates at once. The following three steps modify all templates, all STAT templates, and all PROC LOGISTIC templates, respectively. Additional templates are modified when there are links. Common prefixes for additional templates include: **Common**, **Stat.Bayes**, **Stat.Graphics**, and **Stat.Lmr**.

```
%grtitle;
%grtitle(path=stat)
%grtitle(path=stat.logistic)
```

By default, each macro call begins by deleting all previously modified templates. The following steps modify both the PROC REG and the PROC GLM templates:

```
%grtitle(path=stat.reg)
%grtitle(path=stat.glm, options=nodelete)
```

The first step deletes all the templates in Work.Templat and then modifies the PROC REG templates. The second step additionally modifies PROC GLM templates but without deleting the previously modified templates that are in Work.Templat.

The following step lists all the modified templates that are in the Work library:

```
proc template;
   list / store=work.templat;
quit;
```

The results are displayed in Output 25.8.3. The macro modified templates from **Common.Zreg**, **Stat.GLM**, **Stat.Graphics**, and **Stat.Reg**.

Output 25.8.3 Modified Templates from PROC REG and PROC GLM

Listing of: WORK.TEMPLAT
Path Filter is: *
Sort by: PATH/ASCENDING

Obs	Path	Type
1	Common	Dir
2	Common.Zreg	Dir
3	Common.Zreg.Graphics	Dir
4	Common.Zreg.Graphics.ConditionalFit	Statgraph
5	Common.Zreg.Graphics.ContourFit	Statgraph
6	Common.Zreg.Graphics.Fit	Statgraph
7	Common.Zreg.Graphics.FitBoxPlot	Statgraph
8	Common.Zreg.Graphics.IntPlot	Statgraph
9	Common.Zreg.Graphics.NestPlot	Statgraph
10	Common.Zreg.Graphics.Residual	Statgraph
11	Common.Zreg.Graphics.ResidualPanel	Statgraph
12	Common.Zreg.Graphics.SLICEFITPlot	Statgraph
13	Stat	Dir
14	Stat.GLM	Dir
15	Stat.GLM.Graphics	Dir
16	Stat.GLM.Graphics.BoxPlot	Statgraph
17	Stat.GLM.Graphics.CooksD	Statgraph
18	Stat.GLM.Graphics.FitSummary	Statgraph
19	Stat.GLM.Graphics.LSMLinesPlot	Statgraph
20	Stat.GLM.Graphics.MeanLinesPlot	Statgraph
21	Stat.GLM.Graphics.MeansBoxPlot	Statgraph
22	Stat.GLM.Graphics.ObservedByPredicted	Statgraph
23	Stat.GLM.Graphics.Qq	Statgraph
24	Stat.GLM.Graphics.RStudentByLeverage	Statgraph
25	Stat.GLM.Graphics.RStudentByPredicted	Statgraph
26	Stat.GLM.Graphics.ResidualByPredicted	Statgraph
27	Stat.GLM.Graphics.ResidualHistogram	Statgraph
28	Stat.GLM.Graphics.Rf	Statgraph
29	Stat.Graphics	Dir
30	Stat.Graphics.AnomPlot	Statgraph
31	Stat.Graphics.DiffPlot	Statgraph
32	Stat.Graphics.MeanPlot	Statgraph
33	Stat.Reg	Dir
34	Stat.Reg.Graphics	Dir
35	Stat.Reg.Graphics.CooksD	Statgraph
36	Stat.Reg.Graphics.CooksDChart	Statgraph
37	Stat.Reg.Graphics.DFBETASPanel	Statgraph
38	Stat.Reg.Graphics.DFBETASPlot	Statgraph
39	Stat.Reg.Graphics.DFFITSPlot	Statgraph
40	Stat.Reg.Graphics.DiagnosticsPanel	Statgraph
41	Stat.Reg.Graphics.Fit	Statgraph
42	Stat.Reg.Graphics.FitHeatMap	Statgraph
43	Stat.Reg.Graphics.ObservedByPredicted	Statgraph
44	Stat.Reg.Graphics.PartialPanel	Statgraph
45	Stat.Reg.Graphics.PartialPlot	Statgraph

Output 25.8.3 *continued*

Listing of: WORK.TEMPLAT
Path Filter is: *
Sort by: PATH/ASCENDING

Obs	Path	Type
46	Stat.Reg.Graphics.PredictionPanel	Statgraph
47	Stat.Reg.Graphics.QQPlot	Statgraph
48	Stat.Reg.Graphics.RFPlot	Statgraph
49	Stat.Reg.Graphics.RStudentByPredicted	Statgraph
50	Stat.Reg.Graphics.ResidualBoxPlot	Statgraph
51	Stat.Reg.Graphics.ResidualByPredicted	Statgraph
52	Stat.Reg.Graphics.ResidualHeatMap	Statgraph
53	Stat.Reg.Graphics.ResidualHeatPanel	Statgraph
54	Stat.Reg.Graphics.ResidualHistogram	Statgraph
55	Stat.Reg.Graphics.ResidualPanel	Statgraph
56	Stat.Reg.Graphics.ResidualPlot	Statgraph
57	Stat.Reg.Graphics.RidgePanel	Statgraph
58	Stat.Reg.Graphics.RidgePlot	Statgraph
59	Stat.Reg.Graphics.SelectionCriterionPanel	Statgraph
60	Stat.Reg.Graphics.SelectionCriterionPlot	Statgraph
61	Stat.Reg.Graphics.StepSelectionCriterionPanel	Statgraph
62	Stat.Reg.Graphics.StepSelectionCriterionPlot	Statgraph
63	Stat.Reg.Graphics.StudResCooksDChart	Statgraph
64	Stat.Reg.Graphics.StudentResChart	Statgraph
65	Stat.Reg.Graphics.VIFPlot	Statgraph
66	Stat.Reg.Graphics.rstudentByLeverage	Statgraph

References

Cook, R. D., and Weisberg, S. (1997). "Graphics for Assessing the Adequacy of Regression Models." *Journal of the American Statistical Association* 92:490–499.

Fox, J., and Weisberg, S. (2011). *An R Companion to Applied Regression*. Thousand Oaks, CA: Sage Publications.

Kuhfeld, W. F. (2009). "Modifying ODS Statistical Graphics Templates in SAS 9.2." http://support.sas.com/rnd/app/papers/ods/modtmplt.pdf. Revision of paper by the same title that was published in the *Proceedings of the SAS Global Forum 2009 Conference*.

Kuhfeld, W. F. (2015). *Advanced ODS Graphics Examples*. Cary, NC: SAS Institute Inc. http://support.sas.com/documentation/prod-p/grstat/9.4/en/PDF/odsadvg.pdf.

Kuhfeld, W. F. (2016). *Basic ODS Graphics Examples*. Cary, NC: SAS Institute Inc. http://support.sas.com/documentation/prod-p/grstat/9.4/en/PDF/odsbasicg.pdf.

Chapter 26
Customizing the Kaplan-Meier Survival Plot

Contents

Overview **936**
Controlling the Survival Plot by Specifying Procedure Options **937**
Enabling ODS Graphics and the Default Kaplan-Meier Plot 937
Individual Survival Plots 939
Hall-Wellner Confidence Bands and Homogeneity Test 940
Equal-Precision Bands 941
Displaying the Patients-at-Risk Table inside the Plot 943
Displaying the Patients-at-Risk Table outside the Plot 945
Modifying At-Risk Table Times 946
Reordering the Groups 949
Suppressing the Censored Observations 951
Failure Plots 952
Controlling the Survival Plot by Modifying Graph Templates **958**
The Modularized Templates 958
Changing the Plot Title 960
Modifying the Y Axis 961
Changing the Line Thickness 963
Changing the Group Color 964
Changing the Line Pattern 966
Changing the Font 967
Changing the Legend and Inset Position 969
Changing How the Censored Points Are Displayed 970
Adding a Y-Axis Reference Line 971
Adding Y-Axis Grid Lines 973
Changing the Homogeneity Test Inset 974
Changing the Second Title and Adding a Footnote 976
Adding a Custom Header above the At-Risk Table 978
Adding a Small Inset Table with Event Information 979
Adding an External Table with Event Information 981
Suppressing the Legend 983
Kaplan-Meier Plot with Event Table and Other Customizations 984
Displaying Percentages on the Y Axis 985
Compiled Template Cleanup 988
Graph Templates, Macros, and Macro Variables **988**
The Macro Variables 989

The Smaller Macros . . . 993
The Larger Macros . . . 993
Event Table Macros . . . 997
Dynamic Variables . . . **1000**
Dynamic Variables That Are Automatically Declared . . . 1000
Additional Dynamic Variables . . . 1001
Highly Customized Graphs: Unicode in the STRATA Statement Variable . . . **1002**
Unicode Values Require Alternative Processing . . . 1002
Using the GROUP= Option . . . 1004
Modifying the Data Object . . . 1005
Multiple Strata Variables and Reformatting the Legend . . . 1009
Style Templates . . . **1012**
Changing the Style . . . 1013
Color Priority Styles . . . 1014
Displaying a Style and Extracting Color Lists . . . 1015
Modifying Color Lists . . . 1018
Swapping Colors among Style Elements . . . 1019
Displaying a Style and Extracting Font Information . . . 1021
Displaying Other Style Elements . . . 1023
SAS Item Stores . . . **1025**
References . . . **1025**

Overview

The LIFETEST procedure is a nonparametric procedure for analyzing survival data. You can use PROC LIFETEST to compute the Kaplan-Meier curve (1958), which is a nonparametric maximum likelihood estimate of the survivor function. The Kaplan-Meier plot (also called the product-limit survival plot) is a popular tool in medical, pharmaceutical, and life sciences research. The Kaplan-Meier plot contains step functions that represent the Kaplan-Meier curves of different samples (strata). The Kaplan-Meier plot has many other features that you can add or change through procedure options, graph templates, and style templates. This chapter explores these features in detail but does not explain how to interpret the graphs or the underlying analysis. For more information about PROC LIFETEST and the Kaplan-Meier plot, see Chapter 77, "The LIFETEST Procedure."

This chapter shows you how to modify the Kaplan-Meier plot through a series of examples. It discusses four types of examples: specifying procedure options, modifying graph templates by using macro variables, modifying graph templates by using macros, and changing styles. Most examples do not go into detail about the tools that underlie the template changes. Each example is designed to be small, simple, self-contained, and easy to copy and use "as is" or with minor modifications. Subsequent sections provide more details about the macro variables and macros that are used to modify the graph templates. You can use the simple examples to make a wide variety of changes without reading or understanding the detailed descriptions at the end of this chapter.

Statistical procedures produce tables by using the Output Delivery System (ODS) and produce graphs by using ODS Graphics. Procedures produce graphs as automatically as they produce tables, and graphs and tables are integrated in the ODS output. Graphs that are produced by ODS Graphics are controlled by options, the data object (the matrix of information that is graphed), a style template, and a graph template. A style template is a SAS program that controls the overall appearance of graphs, including colors, line and marker styles, sizes, fonts, and so on. A graph template is a SAS program, written in the Graph Template Language (GTL), that provides a detailed specification of the layout and contents of each graph. Each graph that is created when ODS Graphics is enabled is controlled by a graph template.[1]

If you want to modify a graph template, you usually use the TEMPLATE procedure to display the template of interest, and then you copy it into your editor, modify it, and submit it to SAS to compile. Then, when you run your procedure, it uses the new template. The PROC LIFETEST survival plot is the only plot in SAS for which you have another alternative available for template modification. SAS provides the survival plot templates in a series of macros and macro variables that are modular and easier to modify than the original templates. This chapter provides numerous examples of using these macros and macro variables.

The data that are used in this chapter come from 137 bone marrow transplant patients in a study by Klein and Moeschberger (1997) and are available in the BMT data set in the Sashelp library. At the time of transplant, each patient is classified in one of three risk categories: ALL (acute lymphoblastic leukemia), AML (acute myelocytic leukemia)–Low Risk, and AML–High Risk. The endpoint of interest is the disease-free survival time, which is the time in days until death, relapse, or the end of the study. The variable Group represents the patient's risk category, the variable T represents the disease-free survival time, and the variable Status is the censoring indicator. A status of 1 indicates an event time, and a status of 0 indicates a censored time.

Controlling the Survival Plot by Specifying Procedure Options

This section provides a series of examples that use ODS Graphics and the PLOTS= option in the PROC LIFETEST statement to control the appearance of the survival plot. Other examples use formats and the ORDER= option to control the order of the groups.

Enabling ODS Graphics and the Default Kaplan-Meier Plot

You can use the following statements to enable ODS Graphics and run PROC LIFETEST:

```
ods graphics on;

proc lifetest data=sashelp.BMT;
   time T * Status(0);
   strata Group;
run;
```

ODS Graphics is enabled for this step and all subsequent steps until it is disabled. ODS Graphics remains enabled throughout the examples in this chapter.

[1] ODS Graphics might or might not be enabled by default. ODS Graphics is usually enabled by default in the SAS windowing environment and disabled when you invoke SAS in other ways. However, these defaults can be changed in a number of ways. ODS Graphics is enabled in the first example in this chapter by the ODS GRAPHICS ON statement and remains enabled throughout the chapter.

You specify in the TIME statement that the disease-free survival time is recorded in the variable T. You further specify that the variable Status indicates censoring and 0 indicates a censored time. Separate survivor functions are displayed for each group in the Group variable, which you specify in the STRATA statement.

The plot in Figure 26.1 consists of three step functions, one for each of the three groups of patients. The plot shows that patients in the AML–Low Risk group have longer disease-free survival than patients in the ALL and AML–High Risk groups.

Figure 26.1 Default Kaplan-Meier Plot

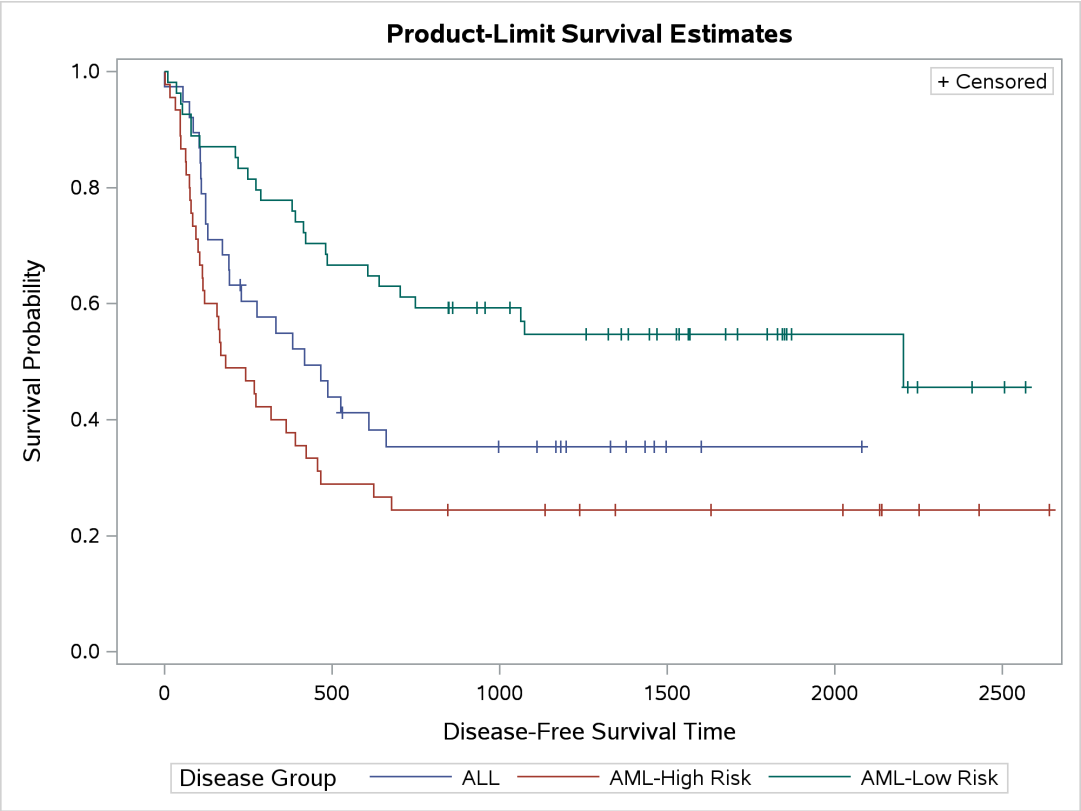

The following step, which explicitly specifies the default PLOTS=SURVIVAL option, is equivalent to the preceding step:

```
proc lifetest data=sashelp.BMT plots=survival;
   time T * Status(0);
   strata Group;
run;
```

The PLOTS= option enables you to control the graphs that a procedure produces. You can use it to request nondefault graphs and specify options for some graphs. You can specify graph names (PLOTS=SURVIVAL), graph options (PLOTS=SURVIVAL(ATRISK OUTSIDE)), and suboptions (PLOTS=SURVIVAL(ATRISK OUTSIDE(0.15))). The PLOTS= option is described in the section "PROC LIFETEST Statement" on page 5762 in Chapter 77, "The LIFETEST Procedure."

Individual Survival Plots

You can use the STRATA=INDIVIDUAL option to request individual survival plots. By default, the STRATA=OVERLAY option produces the plot of overlaid step functions displayed in Figure 26.1. You can run the same analysis but request the results in three separate graphs, one per patient group, as follows:

```
proc lifetest data=sashelp.BMT plots=survival(strata=individual);
   time T * Status(0);
   strata Group;
run;
```

The first of the three survival plots is displayed in Figure 26.2. To conserve space, the other graphs are not displayed.

Figure 26.2 One of Three Individual Plots

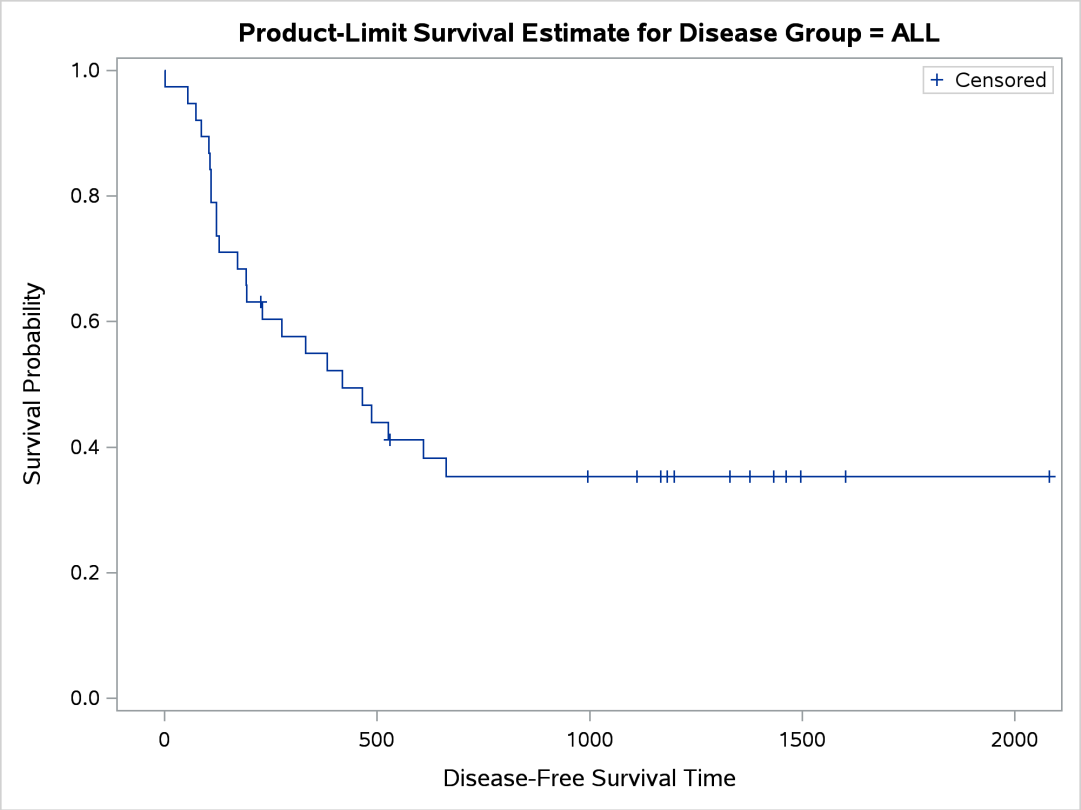

You can use the STRATA=PANEL option as follows to display the results in separate panels of a single graphical display:

```
proc lifetest data=sashelp.BMT plots=survival(strata=panel);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.3.

Figure 26.3 Individual Plots Displayed in a Panel

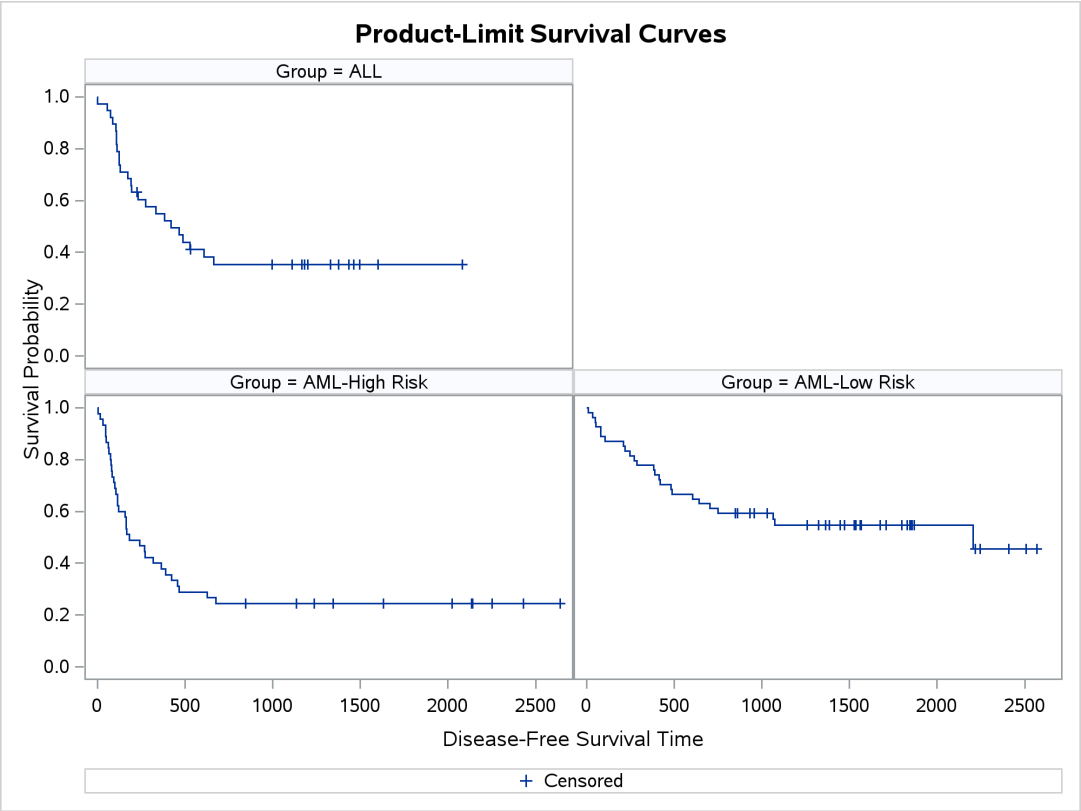

The rest of this chapter discusses overlaid plots such as the one displayed in Figure 26.1.

Hall-Wellner Confidence Bands and Homogeneity Test

You can use the following statements to add Hall-Wellner confidence bands (Hall and Wellner 1980) to Figure 26.1 and display the *p*-value from a test that the strata are homogeneous:

```
proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.4. The Hall-Wellner confidence bands extend to the last event times. The small *p*-value supports rejecting the hypothesis that the groups are homogeneous.

Figure 26.4 Confidence Bands and Homogeneity Test

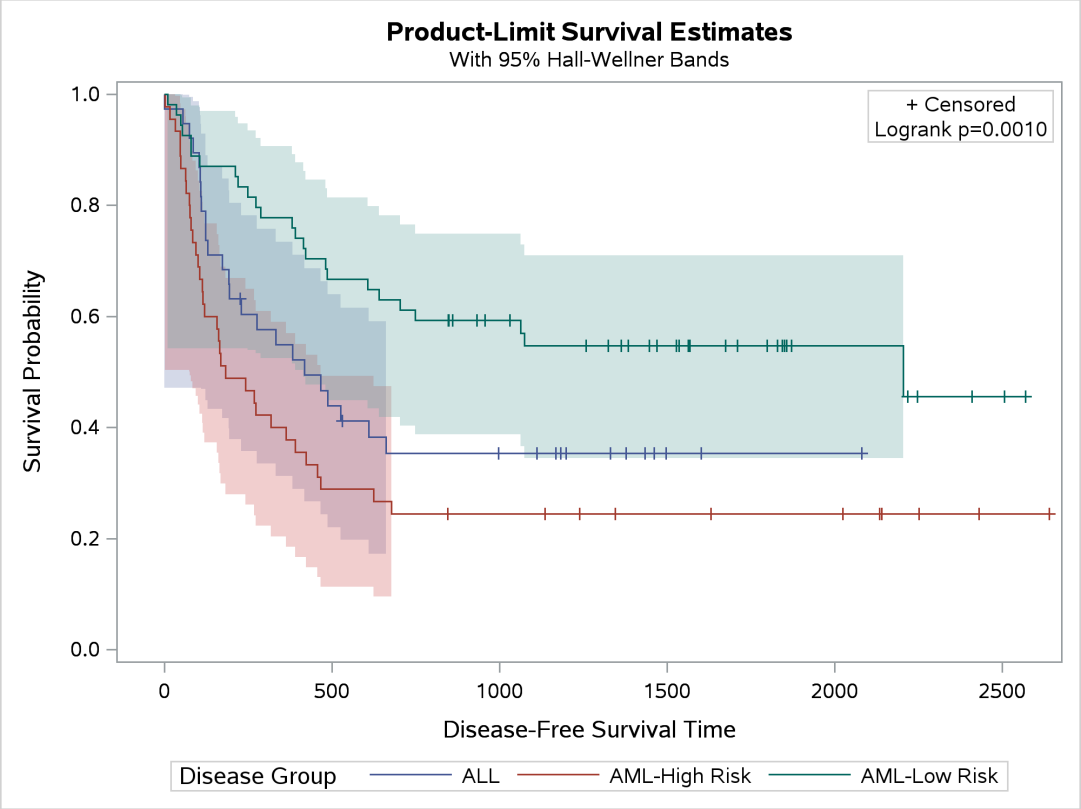

Equal-Precision Bands

You can use the following statements to add equal-precision bands to the plot:

```
proc lifetest data=sashelp.BMT plots=survival(cb=ep test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.5.

Figure 26.5 Equal-Precision Bands

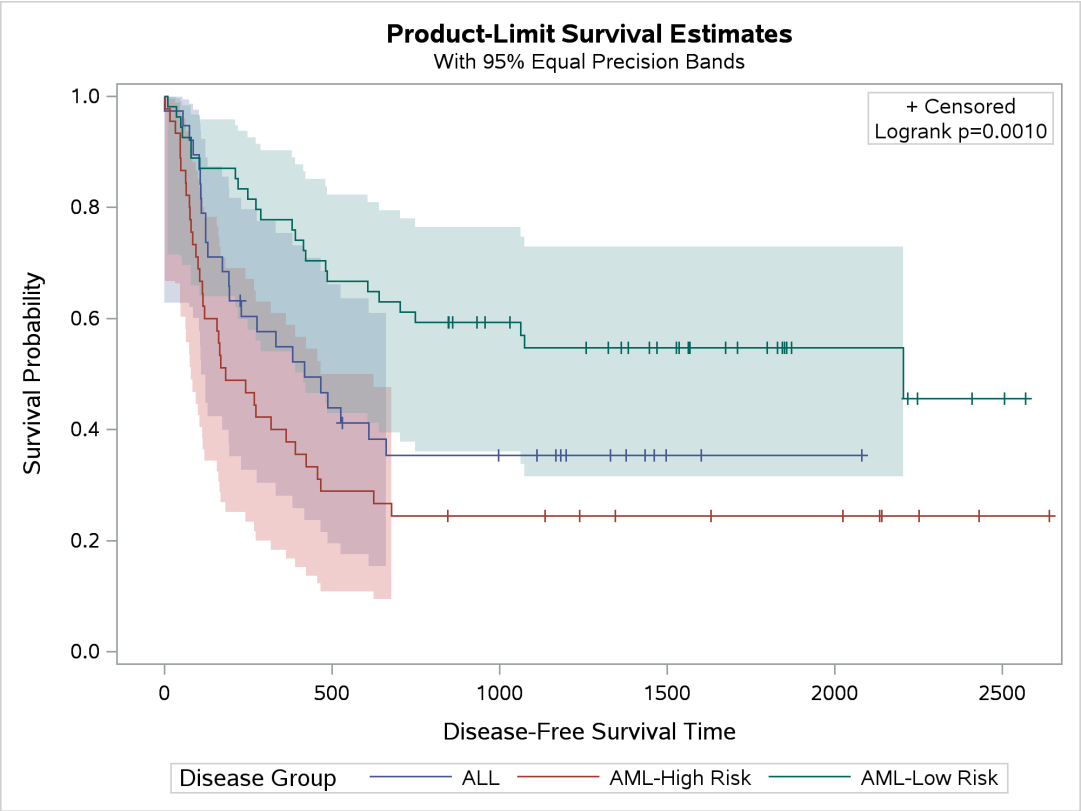

You can use the following statements to add both Hall-Wellner and equal-precision bands to the plot:

```
proc lifetest data=sashelp.BMT plots=survival(cb=all test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.6.

Figure 26.6 Hall-Wellner and Equal-Precision Bands

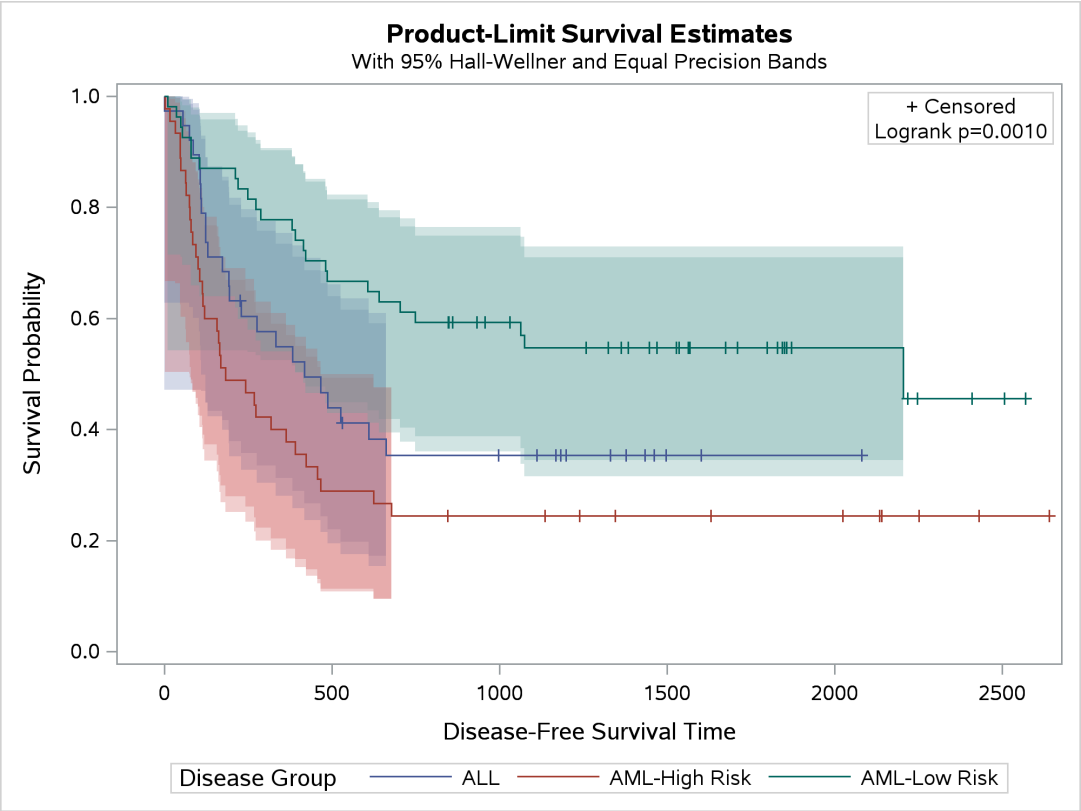

Displaying the Patients-at-Risk Table inside the Plot

You can add the patients-at-risk table to the Kaplan-Meier plot as follows:

```
proc lifetest data=sashelp.BMT plots=survival(cb=hw test atrisk);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.7. By default, the at-risk table is displayed inside the body of the plot. This table shows the number of patients who are at risk for each group for each of the different times. For these data, the default survival times at which at-risk values are displayed are 0 to 2500 by 500. You will see how to specify other values in subsequent examples.

Figure 26.7 At-Risk Table inside the Plot

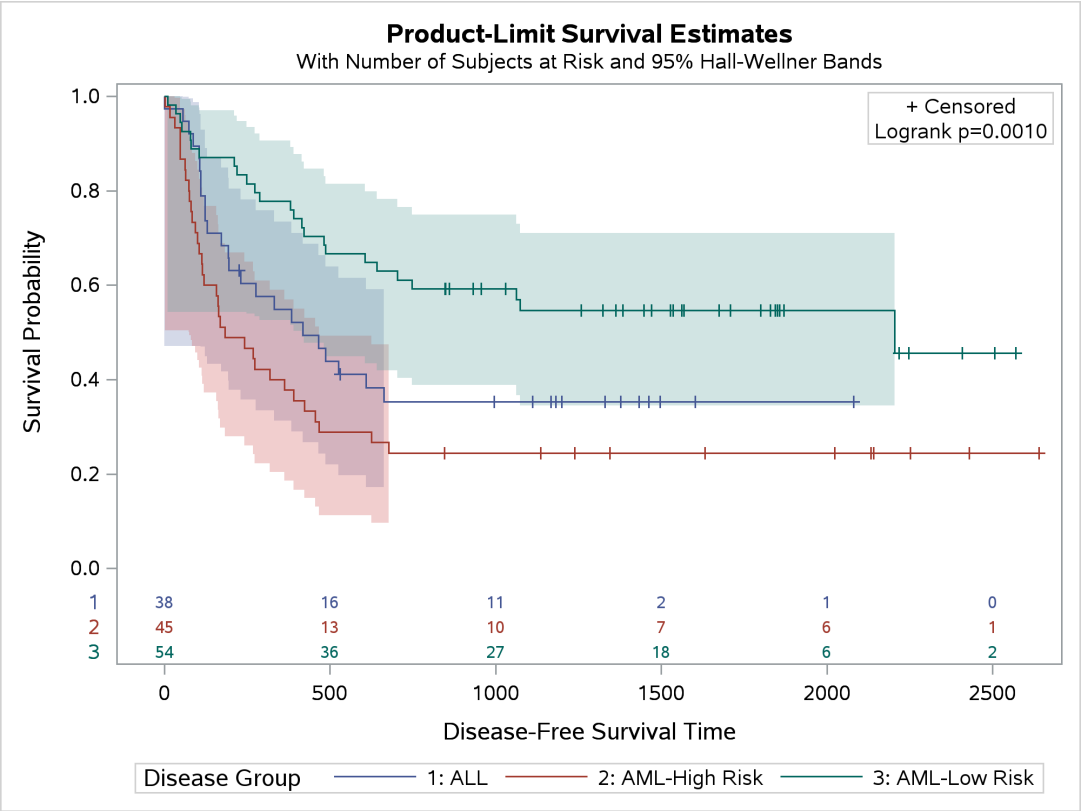

The group labels for the at-risk table are group numbers, and these numbers appear in the legend. Numbers are used rather than the actual labels because the length of the longest label (13) is greater than the default that is set by the maximum label length option (MAXLEN=12). You can display labels rather than the group numbers by specifying a MAXLEN= value equal to the maximum group label length as follows:

```
proc lifetest data=sashelp.BMT plots=survival(cb=hw test atrisk(maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.8. The legend entries and the order of the rows in the at-risk table correspond to the sort order of the values of the Group variable.

Figure 26.8 At-Risk Table with Labels

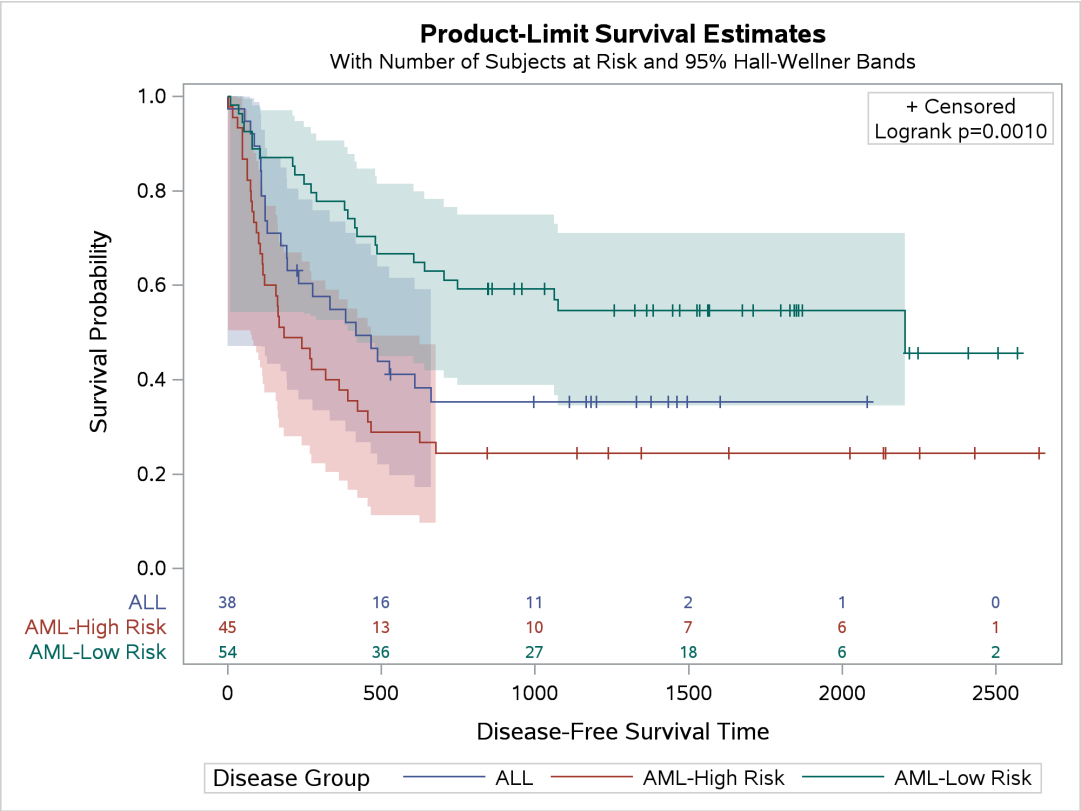

Displaying the Patients-at-Risk Table outside the Plot

You can use the PLOTS=SURVIVAL(OUTSIDE) option to display the at-risk table outside the body of the plot. The option OUTSIDE(0.15) reserves 15% of the vertical graph window for the at-risk table. This example illustrates that the PLOTS= option has options nested within options and options nested within those nested options. The following step produces the plot in Figure 26.9:

```
proc lifetest data=sashelp.BMT
              plots=survival(atrisk(maxlen=13 outside(0.15)));
   time T * Status(0);
   strata Group;
run;
```

Figure 26.9 Moving the At-Risk Table outside the Plot

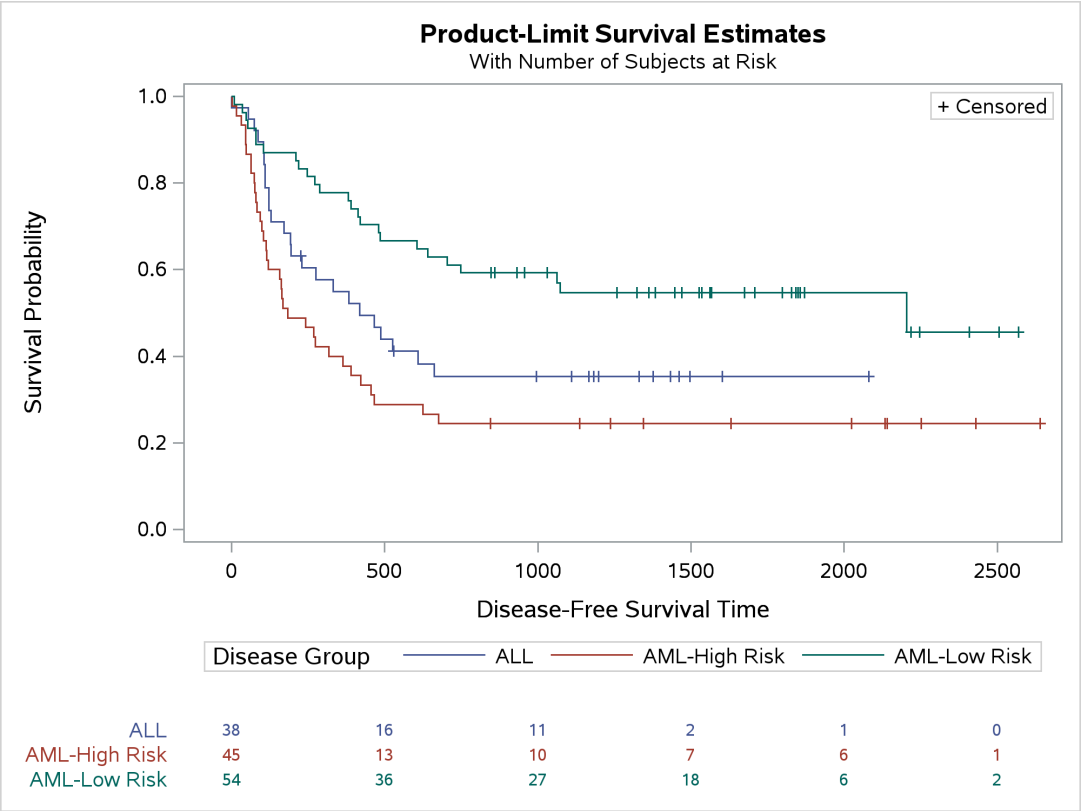

Modifying At-Risk Table Times

The following step explicitly controls the time values at which the at-risk values are displayed by using the PLOTS=SURVIVAL(ATRISK=0 TO 3000 BY 1000) option:

```
proc lifetest data=sashelp.BMT
              plots=survival(atrisk(maxlen=13 outside)=0 to 3000 by 1000);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.10.

Figure 26.10 Specifying At-Risk Values

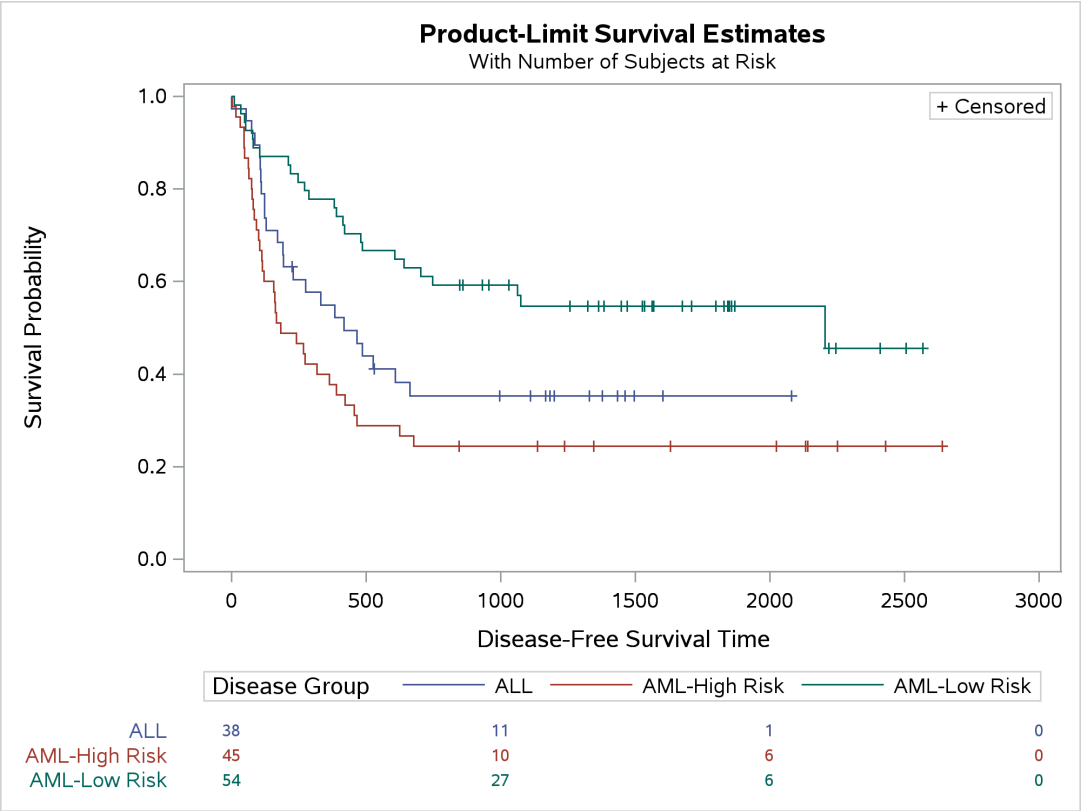

You can specify at-risk values that do not correspond to the original time axis tick marks. You can use the PLOTS=SURVIVAL(ATRISK(ATRISKTICK)) option to add tick marks that correspond to the specified at-risk values:

```
proc lifetest data=sashelp.BMT plots=survival(atrisk
   (atrisktick maxlen=13 outside)=0 500 750 1000 1250 1500 1750 2000 2500);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.11.

Figure 26.11 Controlling At-Risk Tick Marks

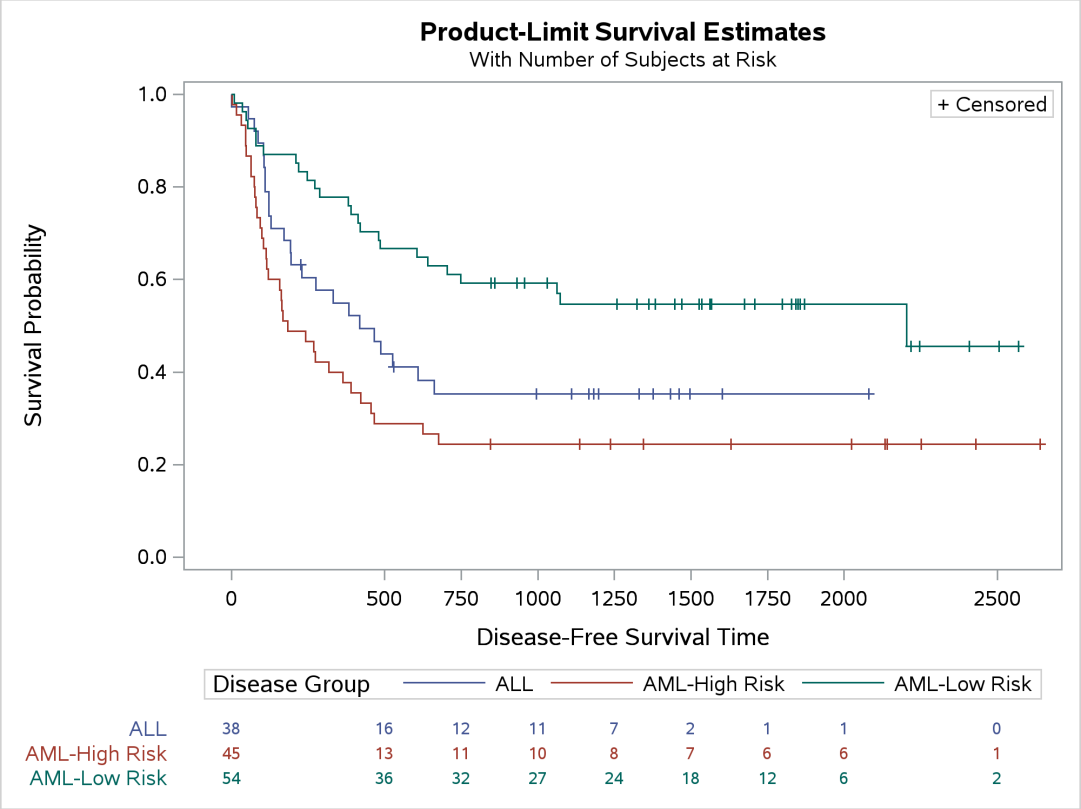

You can display tick values only at those times that are given in the ATRISK= list:

```
proc lifetest data=sashelp.BMT plots=survival(atrisk
   (atrisktickonly maxlen=13 outside)=0 1250 2500);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.12.

Figure 26.12 Controlling At-Risk Tick Marks

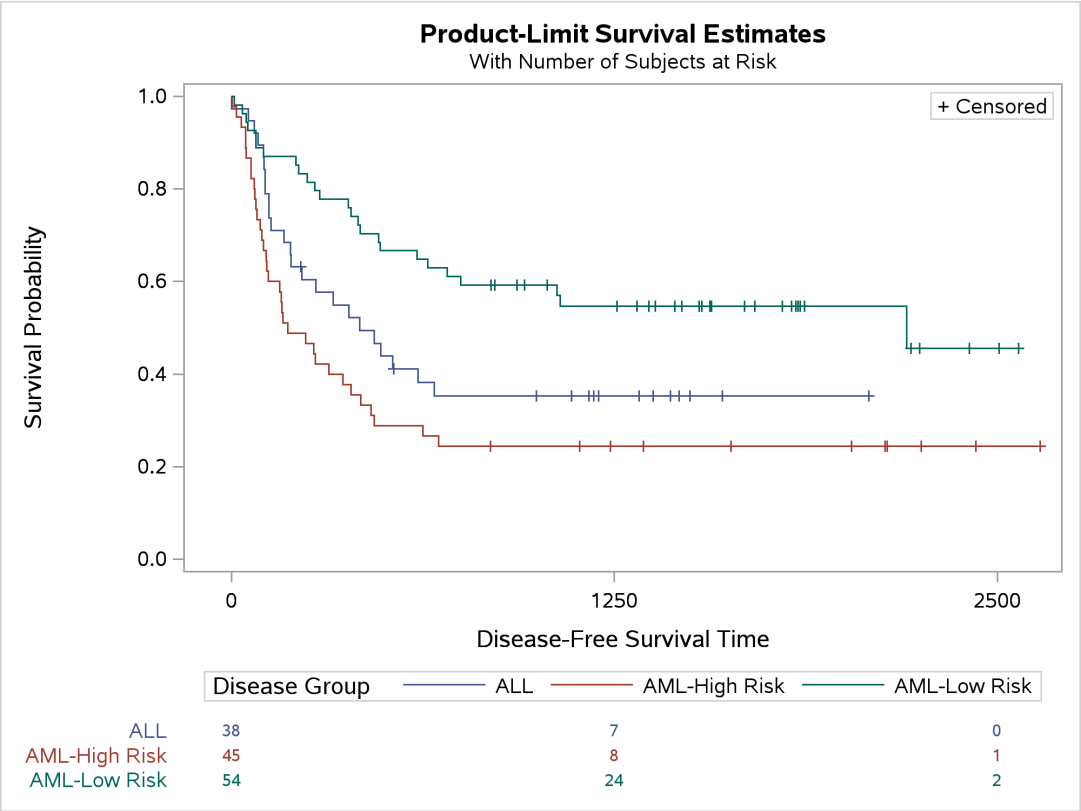

Reordering the Groups

You can change the order of the legend entries by first changing each original group value to a new value in the desired order and then running the analysis with a FORMAT statement to provide the original values. In this example, the order is changed to AML–Low Risk (the top function), followed by ALL (the middle function), followed by AML–High Risk. With this ordering, there is a clearer correspondence between the functions, the at-risk table, and the legend. The following steps illustrate this reordering:

```
proc format;
   invalue bmtnum   'AML-Low Risk' = 1  'ALL' = 2  'AML-High Risk' = 3;
   value   bmtfmt   1 = 'AML-Low Risk'  2 = 'ALL'  3 = 'AML-High Risk';
run;

data BMT(drop=g);
   set sashelp.BMT(rename=(group=g));
   Group = input(g, bmtnum.);
run;

proc lifetest data=BMT plots=survival(cl test atrisk(maxlen=13));
   time T * Status(0);
   strata Group / order=internal;
   format group bmtfmt.;
run;
```

The PROC FORMAT step has two statements. The INVALUE statement creates an informat that maps the values of the original Group variable into integers that have the correct order. The VALUE statement creates a format that maps the integers back to the original values. The informat is used with the INPUT function in the DATA step to create a new integer Group variable. The FORMAT statement assigns the BMTFMT format to the Group variable so that the actual risk groups are displayed in the analysis. You specify the ORDER=INTERNAL option in the STRATA statement to sort the Group values based on internal order (the order specified by the integers, which are the internal unformatted values). This example also illustrates the CL option, which displays pointwise confidence limits for the survival curve (instead of the Hall-Wellner confidence bands). The results are displayed in Figure 26.13.

Figure 26.13 Controlling Legend Order

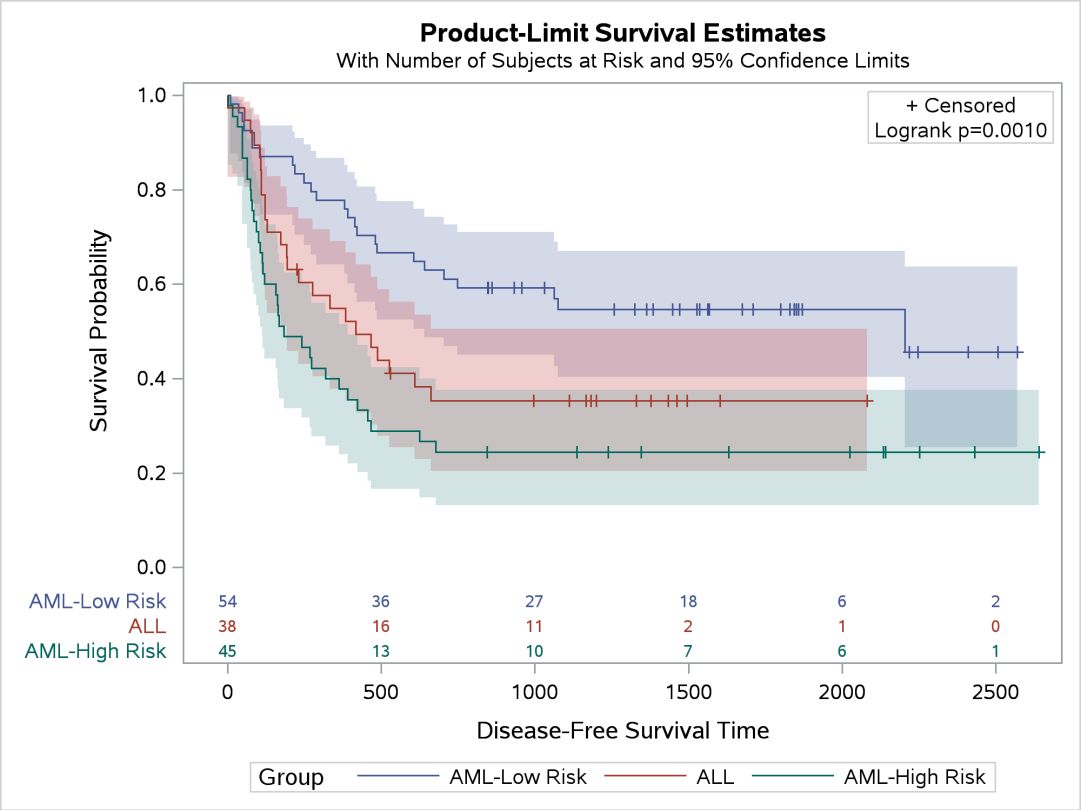

You can submit the following steps to display ALL first, followed by AML–Low Risk and then AML–High Risk:

```
proc format;
   invalue bmtnum 'ALL' = 1  'AML-Low Risk' = 2  'AML-High Risk' = 3;
   value   bmtfmt 1 = 'ALL'  2 = 'AML-Low Risk'  3 = 'AML-High Risk';
run;

data BMT(drop=g);
   set sashelp.BMT(rename=(group=g));
   Group = input(g, bmtnum.);
run;

proc lifetest data=BMT plots=survival(cl test atrisk(maxlen=13));
```

```
   time T * Status(0);
   strata Group / order=internal;
   format group bmtfmt.;
run;
```

The results are displayed in Figure 26.14.

Figure 26.14 Controlling Legend Order

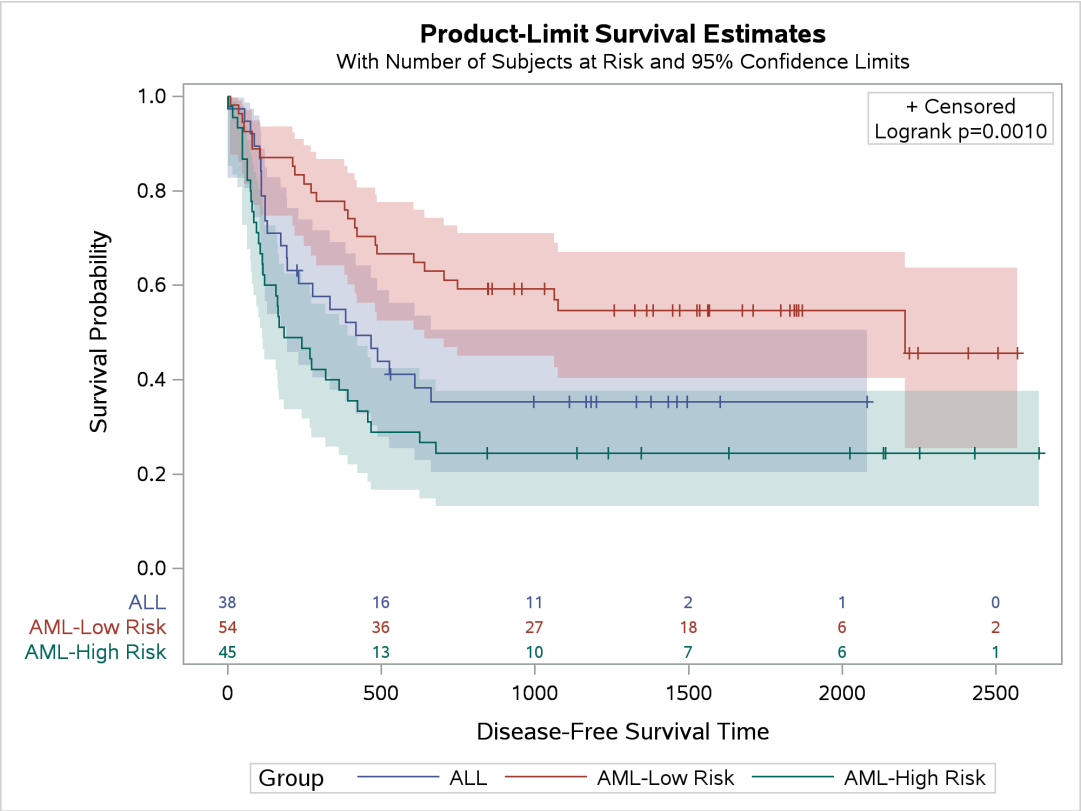

Suppressing the Censored Observations

You can use the PLOTS=SURVIVAL(NOCENSOR) option to suppress the display of censored observations as follows:

```
proc lifetest data=sashelp.BMT
              plots=survival(nocensor test atrisk(maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.15.

Figure 26.15 Censored Values Not Displayed

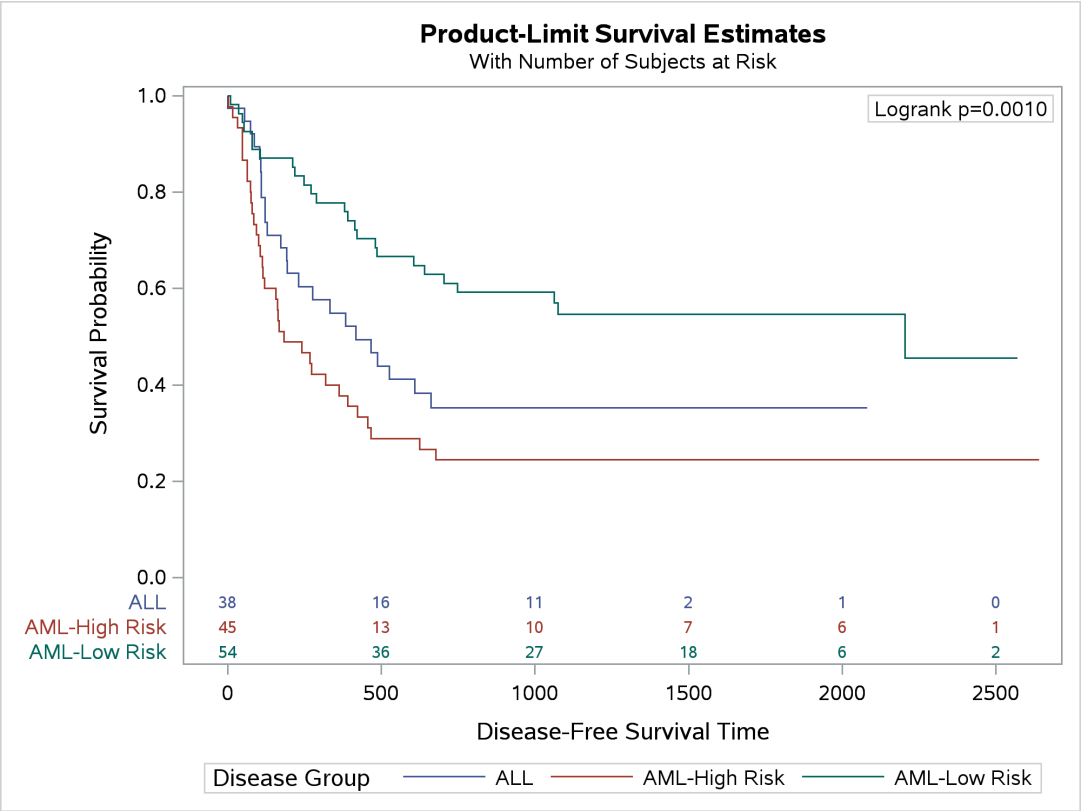

Failure Plots

All the discussion up to this point has been about survival plots. You can instead plot failure probabilities by using the PLOTS=SURVIVAL(FAILURE) option as follows:

```
proc lifetest data=sashelp.BMT
     plots=survival(cb=hw failure test atrisk(maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.16.

Figure 26.16 Failure Plot

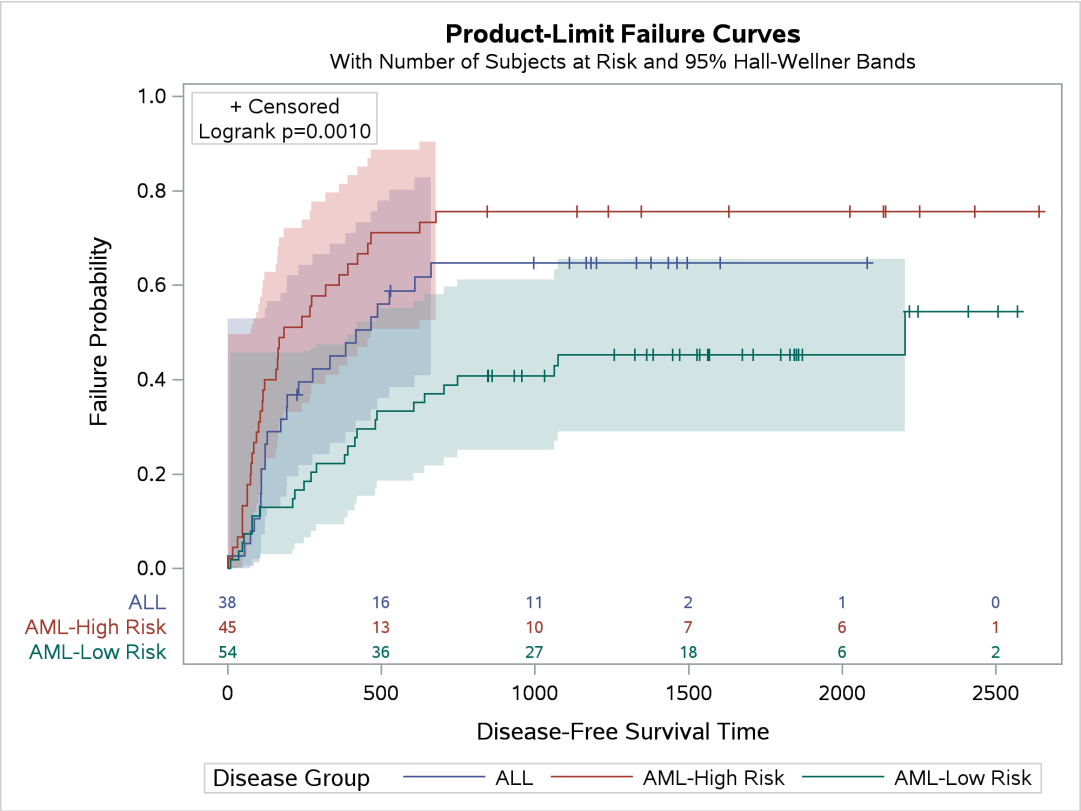

The section "Controlling the Survival Plot by Modifying Graph Templates" on page 958 provides macros for customizing the survival plot. That is the only plot in SAS that has customization macros. If you want to customize the failure plot or other plots, you might need to directly modify the graph template. The rest of this example shows how to programmatically modify the template for the failure plot by using a DATA step. This technique of writing a DATA step to modify a template was introduced by Kuhfeld (2015).

Customizing the Failure Plot Y Axis

This example shows how to display a Y axis that ranges from 0 to 0.8 instead of the customary 0 to 1. You begin by enabling ODS TRACE output and running the procedure. This step, unlike the preceding step, displays the at-risk table outside the graph.

```
ods trace on;
proc lifetest data=sashelp.BMT
   plots=survival(cb=hw failure test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The graph is displayed in Figure 26.17. It has the full Y-axis range of 0 to 1.

Figure 26.17 Failure Plot with At-Risk Table Outside

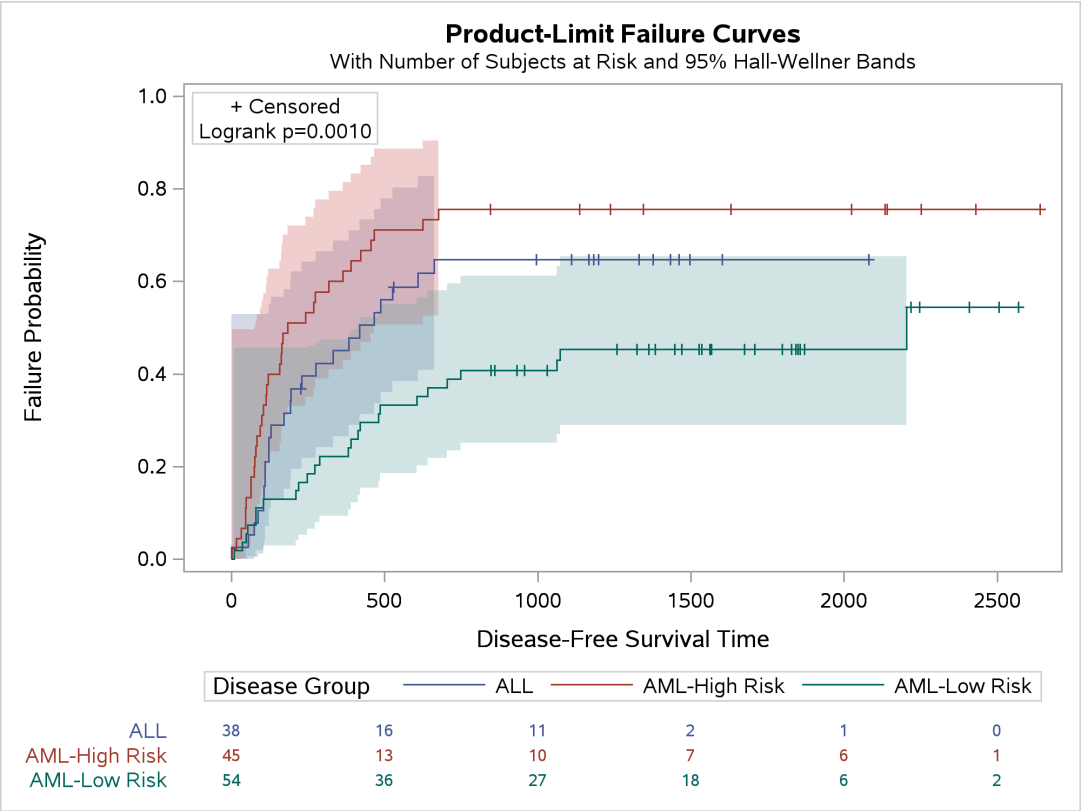

The output from the ODS TRACE ON statement (not displayed here) shows that the name of the graph template is **`Stat.Lifetest.Graphics.ProductLimitFailure2`**. You can write to a file the template that SAS provides for the failure plot by submitting the following step:

```
proc template;
   delete Stat.Lifetest.Graphics.ProductLimitFailure2;
   source Stat.Lifetest.Graphics.ProductLimitFailure2 / file='tpl.tpl';
quit;
```

The DELETE statement deletes any compiled templates that might be left from previous template modifications. The SOURCE statement writes the template to the file *tpl.tpl*. You might need to name your files differently, depending on your operating system and how you run SAS. You can use an editor to read the template, and then you can use a DATA step to modify it and change the Y-axis maximum. The option that you need to change is the VIEWMAX= option. You also need to submit a PROC TEMPLATE statement before submitting the template definition. Finally, you can submit a QUIT statement, although this is not required. You can do all this by submitting the following step:

```
data _null_;                                          /* Standard boilerplate   */
   infile 'tpl.tpl' end=eof;                          /* Standard boilerplate   */
   input;                                             /* Standard boilerplate   */
   if _n_ eq 1 then call execute('proc template;'); /* Standard boilerplate   */
   _infile_ = tranwrd(_infile_, 'viewmax=1',          /* Customization: var =   */
                                'viewmax=0.8');       /* tranwrd(var, from, to);*/
   call execute(_infile_);                            /* Standard boilerplate   */
   if eof then call execute('quit;');                 /* Standard boilerplate   */
```

```
run;                                                    /* Standard boilerplate   */
```

You are not required to specify variables in the INPUT statement. When you are processing text files, it is often convenient to instead use the automatic input buffer variable, _Infile_. The TRANWRD function does the translation and replaces the VIEWMAX= option. You must look at the original template to know how to write translation assignment statements. The CALL EXECUTE statements submit all the template code to a code buffer. SAS runs the program that is stored in the code buffer when the DATA step finishes. If you want to make different template modifications, you need to replace only the assignment statement. The rest of the DATA step remains the same. You can view the modified template by submitting the following statements:

```
proc template;
   source Stat.Lifetest.Graphics.ProductLimitFailure2 / store=sasuser.templat;
quit;
```

The SOURCE statement explicitly references the compiled template that is stored in the templat item store in the Sasuser library. Alternatively, you can write the modified template to a file, as in the following code, and then use a text comparison utility to see the changes between *tpl.tpl* and *tplmod.tpl*:

```
proc template;
   source Stat.Lifetest.Graphics.ProductLimitFailure2 /
     store=sasuser.templat file='tplmod.tpl';
quit;
```

The following step runs PROC LIFETEST, which uses the modified template:

```
proc lifetest data=sashelp.BMT
   plots=survival(cb=hw failure test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The graph is displayed in Figure 26.18.

Figure 26.18 Restricted Axis Range

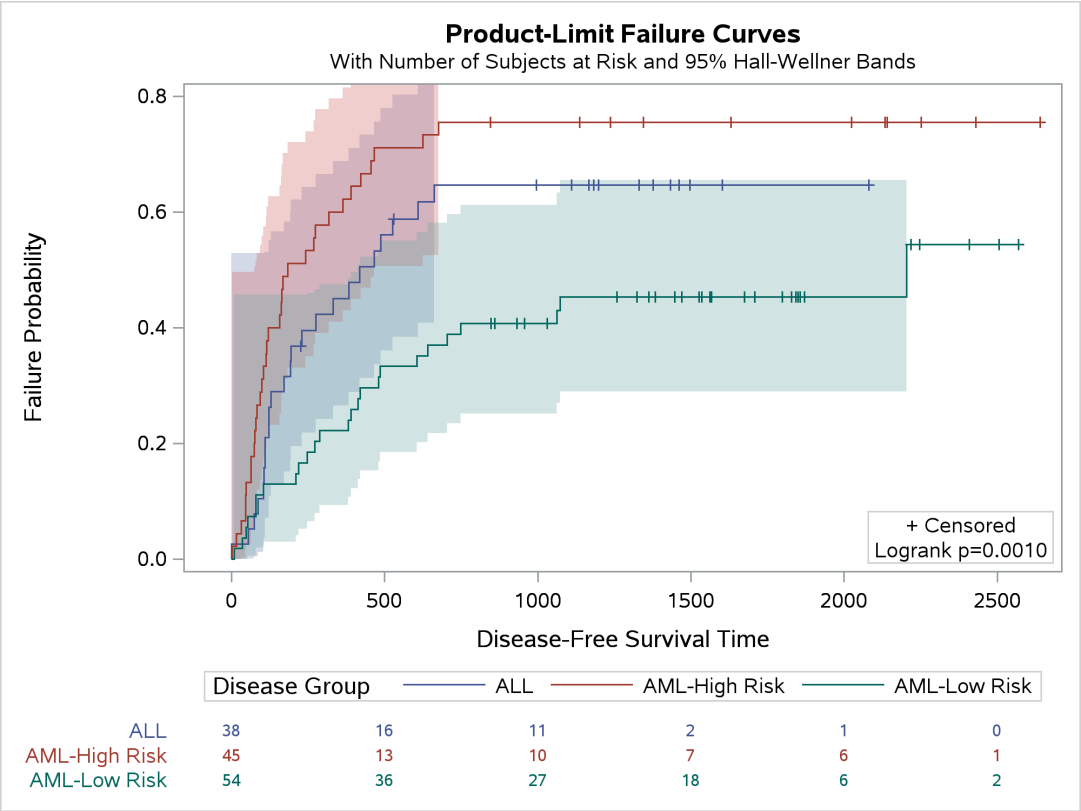

Converting the Y-Axis Scale to Percentages

The following steps continue using the *tpl.tpl* file, but this time they convert the Y-axis scale to percentages:

```
data _null_;
   infile 'tpl.tpl' end=eof;
   input;
   if _n_ eq 1 then call execute('proc template;');
   _infile_ = tranwrd(_infile_, 'viewmax=1', 'viewmax=100');
   _infile_ = tranwrd(_infile_, 'tickvaluelist=(0 .2 .4 .6 .8 1.0)',
                    'tickvaluelist=(0 20 40 60 80 100)');
   _infile_ = tranwrd(_infile_, '1-', '100-100*');
   _infile_ = tranwrd(_infile_, 'Failure Probability', 'Failure Percentage');
   call execute(_infile_);
   if eof then call execute('quit;');
run;

proc lifetest data=sashelp.BMT
   plots=survival(cb=hw failure test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The *tpl.tpl* file was not changed by the previous customization, so you can process it again to perform a new template modification. The DATA step changes the option VIEWMAX=1 to VIEWMAX=100; changes the

tick values to range from 0 to 100; and changes the expression for many Y-axis options, such as Y=EVAL(1–SURVIVAL) in the STEPPLOT statement, to Y=EVAL(100–100*SURVIVAL). The output data object variables for the Y axis are on a survival scale of 0 to 1. Subtracting the Y value from 1 transforms the scale to a failure scale. Subtracting 100 times the survival value from 100 converts the scale to a percentage scale. Finally, the DATA step changes the Y-axis label. Again, you cannot write code like this without first looking at the template that is stored in the *tpl.tpl* file. For example, you must correctly specify the case and spacing when you specify the string that is to be translated. The graph is displayed in Figure 26.19.

Figure 26.19 Percentages on the Y Axis

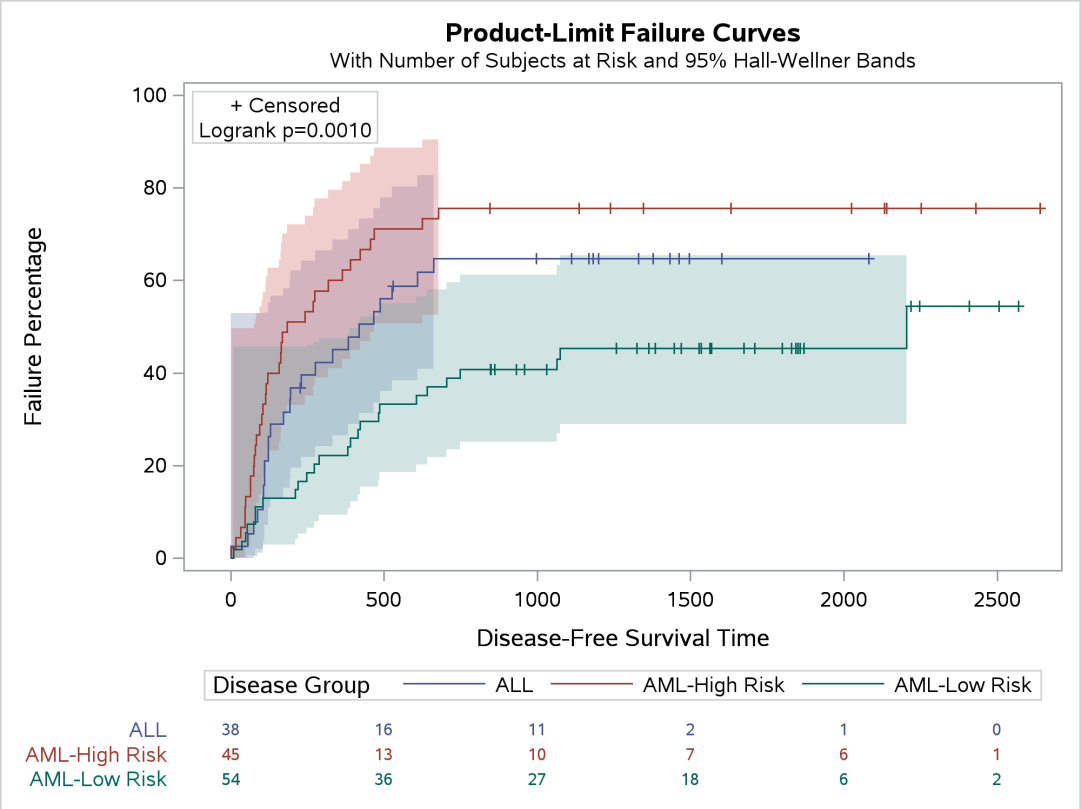

Controlling the Survival Plot by Modifying Graph Templates

The preceding section illustrates the PLOTS= options for controlling the survival plot. If you need to make modifications that are not shown in that section, this section shows how to modify the survival plot by using macros and macro variables to modify graph templates.

The Modularized Templates

SAS provides the two templates that are used to make the survival plot in a modularized form.[2] The modularized version of the two survival plot templates is available in the SAS sample library and is on the Web at http://support.sas.com/documentation/onlinedoc/stat/ex_code/151/templft.html.

The file defines the macro %ProvideSurvivalMacros, which defines a series of macros and macro variables. The %ProvideSurvivalMacros macro contains a %GLOBAL statement, a series of %LET statements, and several macro definitions. It ends with a call to the %CompileSurvivalTemplates macro (which is defined inside the %ProvideSurvivalMacros macro), which compiles the two survival plot templates.[3] By using these macros and macro variables, you can easily specify single changes that modify both templates. All the statements in this file are displayed and explained in more detail in the section "Graph Templates, Macros, and Macro Variables" on page 988.

The %ProvideSurvivalMacros macro provides a way to provide (and in subsequent steps restore) the default macros and macro variables. The macros and macro variables are designed so that you can make most changes by submitting just a few lines of SAS code. Hence, you should not modify any of the statements while they are inside the %ProvideSurvivalMacros macro. Rather, you should use this macro only to provide all the default macros and macro variables. You should modify the individual macros and macro variables outside the context of the %ProvideSurvivalMacros macro.[4] The reasons for this will become clearer as you work through the examples. Before you modify anything, you must submit the %ProvideSurvivalMacros macro definition from the sample library to SAS. You can both store the macros in a temporary file and submit them to SAS by submitting the following statements:

```
data _null_;
   %let url = //support.sas.com/documentation/onlinedoc/stat/ex_code/151;
   infile "http:&url/templft.html" device=url;

   file 'macros.tmp';
   retain pre 0;
   input;
   _infile_ = tranwrd(_infile_, '&amp;', '&');
   _infile_ = tranwrd(_infile_, '&lt;' , '<');
```

[2]The two templates that PROC LIFETEST uses are named **Stat.Lifetest.Graphics.ProductLimitSurvival** and **Stat.Lifetest.Graphics.ProductLimitSurvival2**.

[3]You might wonder why these macros are not simply made available in the SAS autocall library. The autocall library provides macros that you can run. In this context, you do not need to simply run a macro. You need to copy it, extract parts of it, modify those parts, and submit the modified statements. That is not convenient with the autocall library.

[4]However, there might be something that you *always* want to change. For example, if you always want the survival plot to be entitled 'Kaplan-Meier Plot', then you can modify the title once inside the %ProvideSurvivalMacros macro. This is not illustrated in this chapter. All examples illustrate ad hoc changes that are made outside the context of the %ProvideSurvivalMacros macro.

```
   if index(_infile_, '</pre>') then pre = 0;
   if pre then put _infile_;
   if index(_infile_, '<pre>')  then pre = 1;
run;

%inc 'macros.tmp' / nosource;
```

You might need to change the file name here and elsewhere in this chapter, depending on your default working directory, your operating system, and the method you use to run SAS. For more information about specifying file names and directories, see the *SAS Companion* documentation for your operating system.

The TRANWRD function calls change the HTML coding of the ampersand (`'&'`) and the less than sign `'<'`) to the actual characters (`'&'` and `'<'`).

Submitting these statements only defines the %ProvideSurvivalMacros macro. It does not make any of its component macros and macro variables available. The URL macro variable is used to avoid an overly long INFILE statement.

You can provide the default macros and macro variables by running the following macro:

```
%ProvideSurvivalMacros
```

Running this macro provides the default macros and macro variables (or restores them if you have previously submitted the %ProvideSurvivalMacros macro).[5] The %ProvideSurvivalMacrosmacro also runs the %CompileSurvivalTemplates macro and hence replaces any compiled survival plot templates that you might have created in the past. You can recompile the templates by submitting the following macro:

```
%CompileSurvivalTemplates
```

This macro runs PROC TEMPLATE and compiles the templates from all the macros and macro variables in the %ProvideSurvivalMacros macro along with any that you modified. Running this macro produces two compiled templates that are stored in a special SAS data file called an item store. For more information about SAS item stores, see the section "SAS Item Stores" on page 1025. Assuming that you have not modified your ODS path by using an ODS PATH statement, compiled templates are stored in an item store in the Sasuser library. Files in the Sasuser library persist across SAS sessions until they are deleted. When you are done with a modified template, it is wise to clean up all remnants of it by restoring the default macros and by deleting the modified templates from the Sasuser template item store. You can delete the modified templates (so that SAS can only find the original templates) by running the following step:

```
proc template;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival  /
          store=sasuser.templat;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival2 /
          store=sasuser.templat;
run;
```

This step deletes the compiled templates from the item store sasuser.templat. You can omit the STORE= option if you are using the default ODS path, but it is good practice to explicitly control which templates are deleted. Deleting the compiled templates does not change any of the macros or macro variables. Only the compiled templates (not the macros or macro variables) affect the graph when you run PROC LIFETEST. For more information about compiled templates, item stores, and cleanup, see the section "SAS Item Stores" on page 1025.

[5] Semicolons are not needed after a macro call like this one, so they are not used in these examples.

Changing the Plot Title

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

Here is a simple, complete program (except for retrieving the %ProvideSurvivalMacros macro from the sample library) with setup, macro variable modifications to change the title, and cleanup:

```
/*-- Original Macro Variable Definitions ----------------------------------
%let TitleText0 = METHOD " Survival Estimate";
%let TitleText1 = &titletext0 " for " STRATUMID;
%let TitleText2 = &titletext0 "s";
-------------------------------------------------------------------------*/

                                          /* Make the macros and macro      */
%ProvideSurvivalMacros                    /* variables available.           */

%let TitleText0 = "Kaplan-Meier Plot";    /* Change the title.              */
%let TitleText1 = &titletext0 " for " STRATUMID;
%let TitleText2 = &titletext0;

%CompileSurvivalTemplates                 /* Compile the templates with     */
                                          /* the new title.                 */

proc lifetest data=sashelp.BMT            /* Perform the analysis and make  */
              plots=survival(cb=hw test); /* the graph.                     */
   time T * Status(0);
   strata Group;
run;

%ProvideSurvivalMacros                    /* Optionally restore the default */
                                          /* macros and macro variables.    */

proc template;                            /* Delete the modified templates. */
   delete Stat.Lifetest.Graphics.ProductLimitSurvival  / store=sasuser.templat;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival2 / store=sasuser.templat;
run;
```

The results are displayed in Figure 26.20. You can see that the graph title is now 'Kaplan-Meier Plot'.

There are multiple title macro variables because two different types of plots are defined in the survival plot templates. The first macro variable, TitleText0, contains the text that is the same for both types of plots. The second macro variable, TitleText1, contains the title for the single-stratum case. The third macro variable, TitleText2, contains the title for the multiple-strata case. Both TitleText1 and TitleText2 use the common text defined in TitleText0. Both TitleText0 and TitleText2 were changed from their original definition; the definition of TitleText1 was copied from the %ProvideSurvivalMacros macro. You must provide all relevant %LET statements when you modify TitleText0. In this case it is TitleText0 and TitleText2, but it is easy to copy all three and then just modify what you need. Alternatively, when you know the number of strata, you can modify only TitleText1 or TitleText2.

Figure 26.20 Kaplan-Meier Plot Title Modification

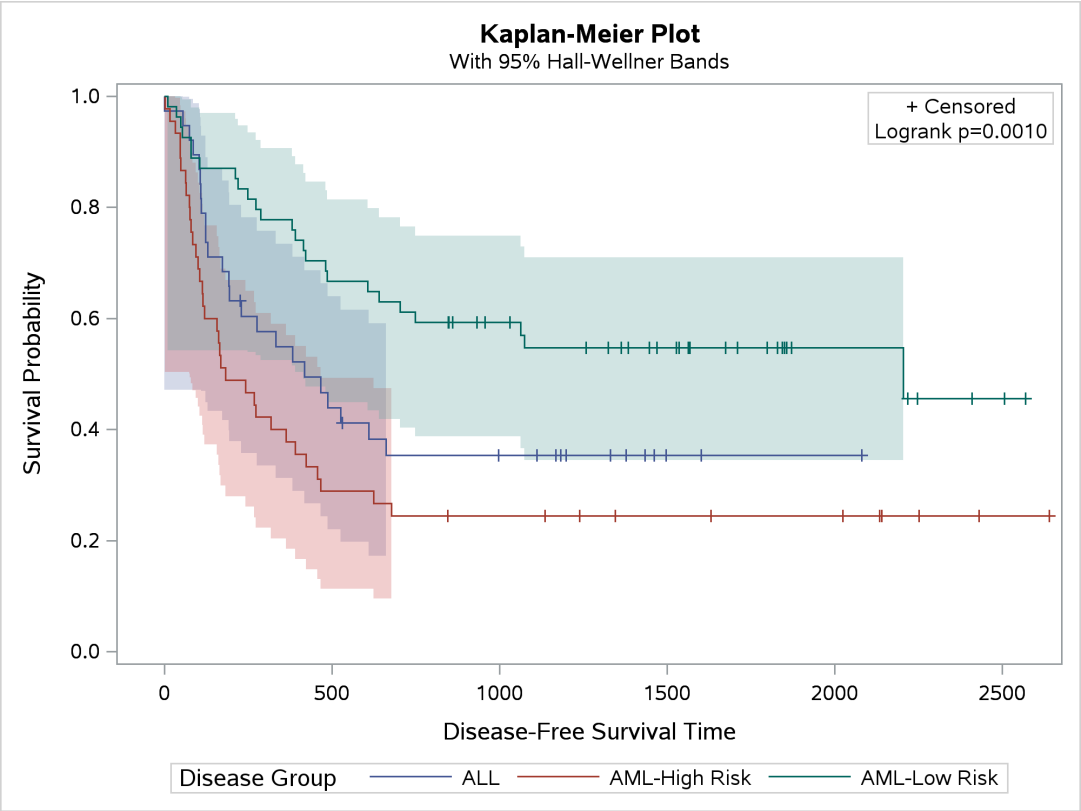

Modifying the Y Axis

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

The following statements modify the default tick value list for the Y axis from the default increment of 0.2 to have an increment of 0.25 and also change the Y-axis label to 'Survival':

```
/*-- Original Macro Variable Definitions ----------------------------------
%let yOptions   = label="Survival Probability" shortlabel="Survival"
                  linearopts=(viewmin=0 viewmax=1
                              tickvaluelist=(0 .2 .4 .6 .8 1.0));
-------------------------------------------------------------------------*/

%ProvideSurvivalMacros

%let yOptions = label="Survival"
                linearopts=(viewmin=0 viewmax=1
                            tickvaluelist=(0 .25 .5 .75 1));

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
```

```
run;
```

The results are displayed in Figure 26.21.

Figure 26.21 Y-Axis Modification

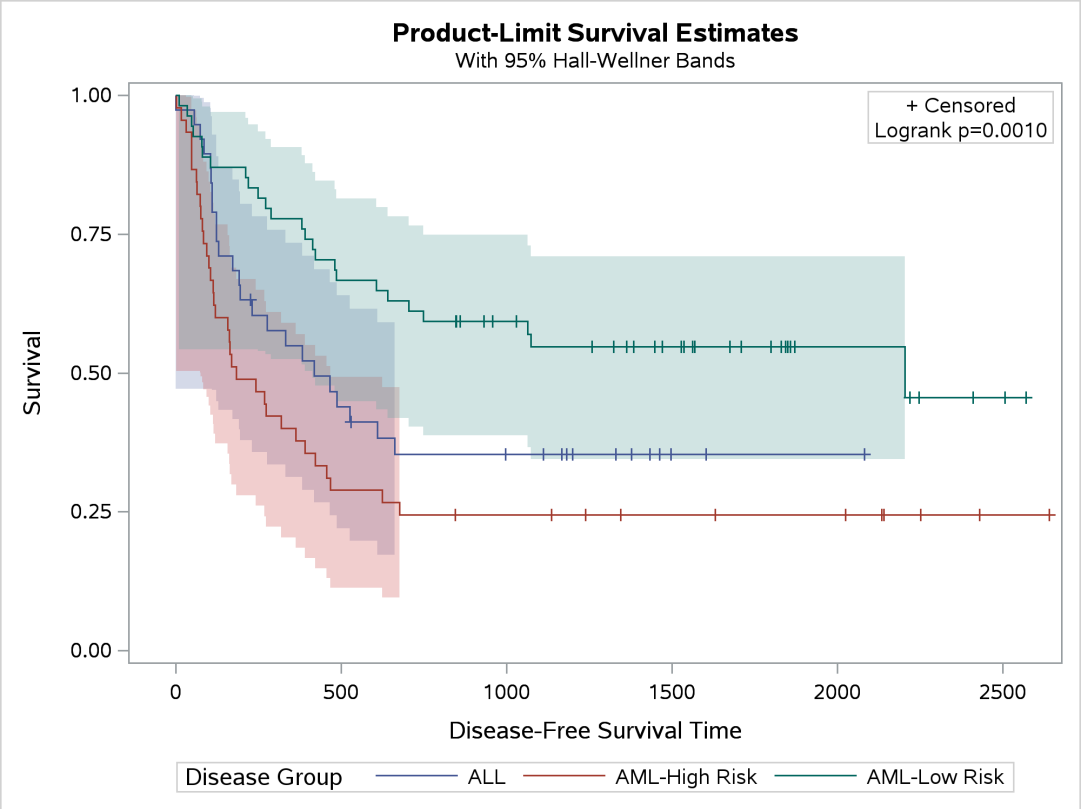

The following statements modify the Y axis so that tick marks start at 0.2:

```
%ProvideSurvivalMacros

%let yOptions = label="Survival"
                linearopts=(viewmin=0.2 viewmax=1
                            tickvaluelist=(0 .2 .4 .6 .8 1.0));

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

You only need to change the value of the VIEWMIN= option, in this case from 0 to 0.2. You do not need to modify the tick value list. The VIEWMIN= option (not the tick value list) controls the smallest value shown on the axis. The results are displayed in Figure 26.22.

Figure 26.22 Y Axis, First Tick Change

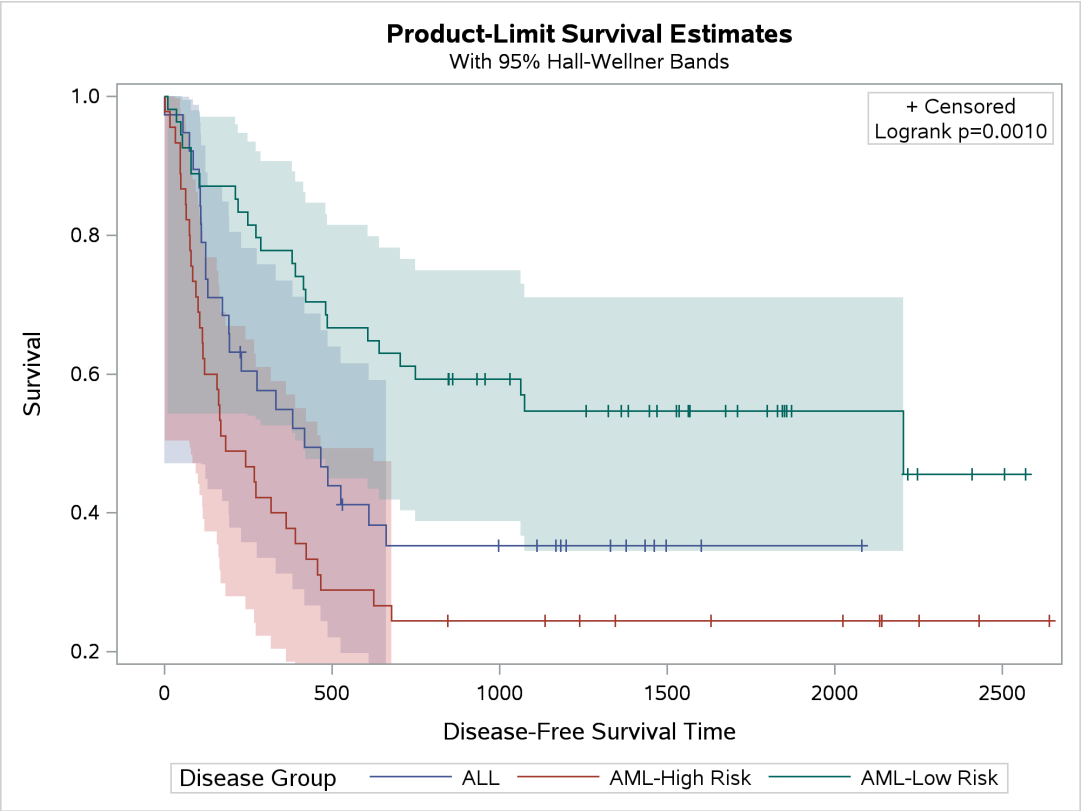

Changing the Line Thickness

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

The following steps modify the line thickness for the step functions in the survival plot:

```
%ProvideSurvivalMacros

%let StepOpts = lineattrs=(thickness=2.5);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.23.

Figure 26.23 Changing Line Thickness

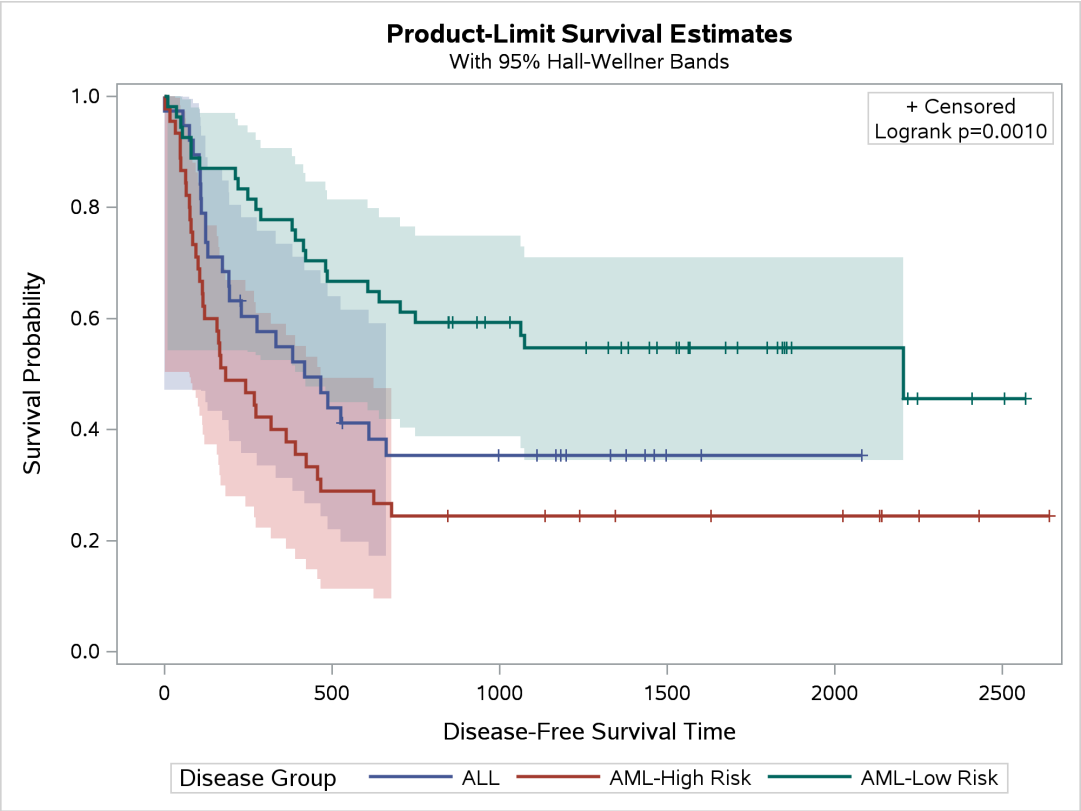

By default, the StepOpts macro variable is null.

Changing the Group Color

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

SAS styles control the colors displayed in graphs. The style elements **GraphData1**, **GraphData2**, ..., **GraphData12** control the appearance of groups of observations such as the survival step plots in the Kaplan-Meier plot. You can override these colors by using the GraphOpts macro variable (which is null by default). By default, the colors for the first three groups in the HMTLBlue style are shades of blue, red, and green. You can change them to a pure green, red, and blue as follows:

```
%ProvideSurvivalMacros

%let GraphOpts = DataContrastColors=(green red blue)
                 DataColors=(green red blue);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
```

```
run;
```

The results are displayed in Figure 26.24. The DATACONTRASTCOLORS= option specifies the contrast colors, which are used for markers and lines. The DATACOLORS= option specifies the colors, which are used for shaded areas such as confidence bands.

Figure 26.24 Named Color Specifications

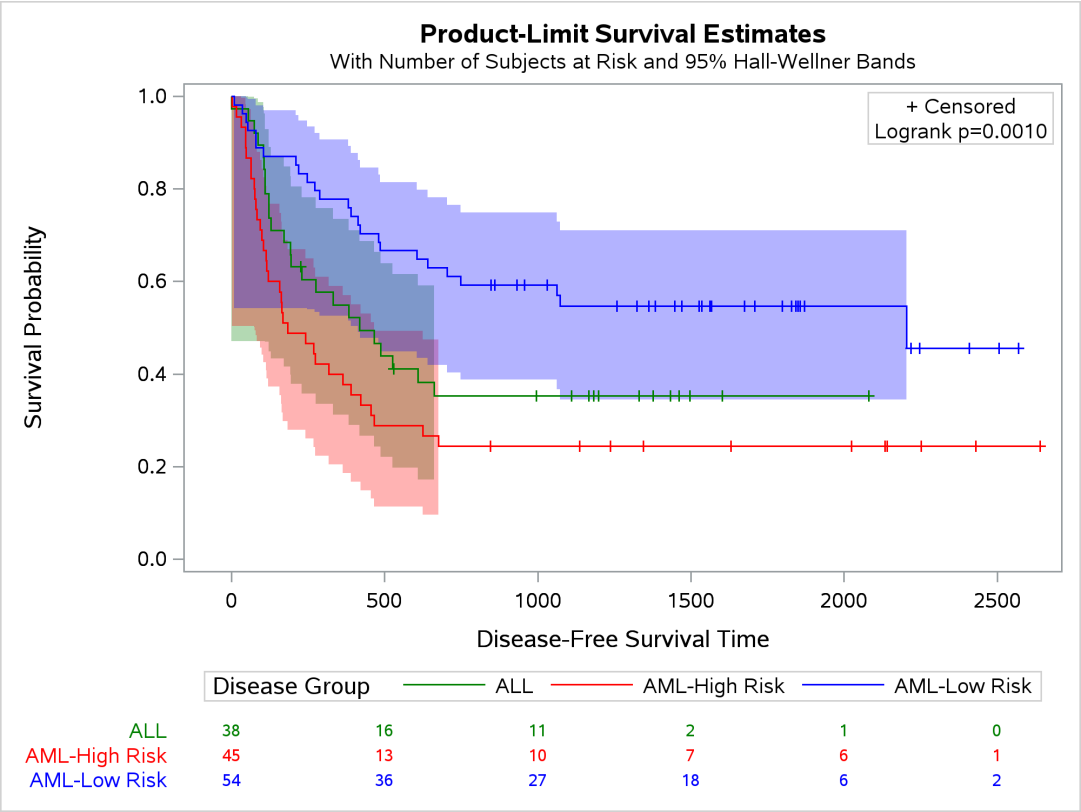

The original colors (as shown in Figure 26.47) are more subtle than those shown in Figure 26.24. If you want to change the order of the original colors by using this approach, then you need to know what they are so that you can specify them. The graph colors for the HTMLBlue and Statistical styles are extracted from the style in the section "Displaying a Style and Extracting Color Lists" on page 1015 and displayed in Figure 26.50. The section "Modifying Color Lists" on page 1018 shows you how to change the graph template to specify the original colors in a different order. The section "Swapping Colors among Style Elements" on page 1019 shows you how to use a macro to change a style template to specify the original colors in a different order (without having to extract and specify the color names).

Changing the Line Pattern

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

You can change the line patterns as follows:

```
%ProvideSurvivalMacros

%let GraphOpts = attrpriority=none
                 DataLinePatterns=(ShortDash MediumDash LongDash);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.25.

Figure 26.25 Changing the Line Patterns

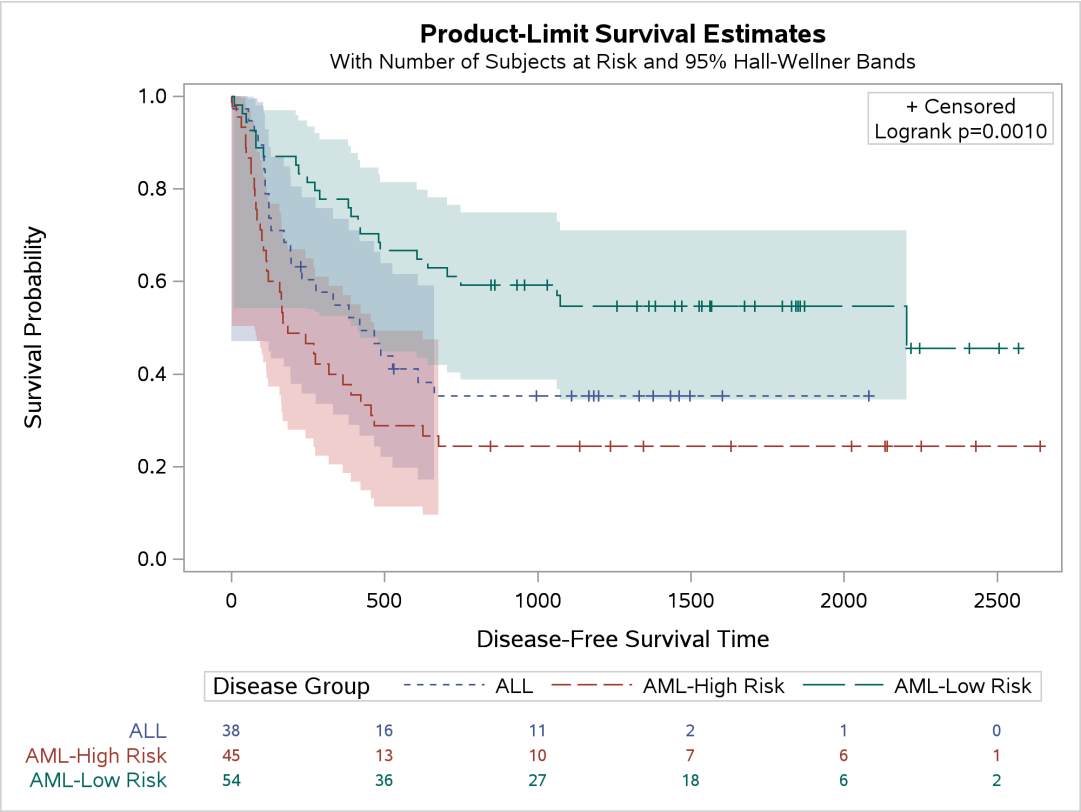

Other values for the DATALINEPATTERNS= option are provided in the section "The Macro Variables" on page 989. You must use the option ATTRPRIORITY=NONE when you want to have varying line patterns in an ATTRPRIORITY=COLOR style like HTMLBlue or Pearl. In an ATTRPRIORITY=COLOR style, groups

are not distinguished by line patterns, and the line patterns for second and subsequent groups match the line pattern for the first group.

Changing the Font

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

You can change the Y-axis, X-axis, and title fonts as follows:

```
%ProvideSurvivalMacros

/*-- Original Macro Variable Definitions ----------------------------------
%let TitleText0 = METHOD " Survival Estimate";
%let TitleText1 = &titletext0 " for " STRATUMID;
%let TitleText2 = &titletext0 "s";
%let yOptions   = label="Survival Probability"
                  shortlabel="Survival"
                  linearopts=(viewmin=0 viewmax=1
                              tickvaluelist=(0 .2 .4 .6 .8 1.0));
%let xOptions   = shortlabel=XNAME
                  offsetmin=.05
                  linearopts=(viewmax=MAXTIME tickvaluelist=XTICKVALS
                              tickvaluefitpolicy=XTICKVALFITPOL);
---------------------------------------------------------------------------*/

%let tatters    = textattrs=(size=12pt weight=bold family='arial');
%let TitleText0 = METHOD " Survival Estimate";
%let TitleText1 = &titletext0 " for " STRATUMID / &tatters;
%let TitleText2 = &titletext0 "s" / &tatters;

%let yOptions   = label="Survival Probability"
                  shortlabel="Survival"
                  labelattrs=(size=10pt weight=bold)
                  tickvalueattrs=(size=8pt)
                  linearopts=(viewmin=0 viewmax=1
                              tickvaluelist=(0 .2 .4 .6 .8 1.0));

%let xOptions   = shortlabel=XNAME
                  offsetmin=.05
                  labelattrs=(size=10pt weight=bold)
                  tickvalueattrs=(size=8pt)
                  linearopts=(viewmax=MAXTIME tickvaluelist=XTICKVALS
                              tickvaluefitpolicy=XTICKVALFITPOL);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.26.

Figure 26.26 Changing the Fonts

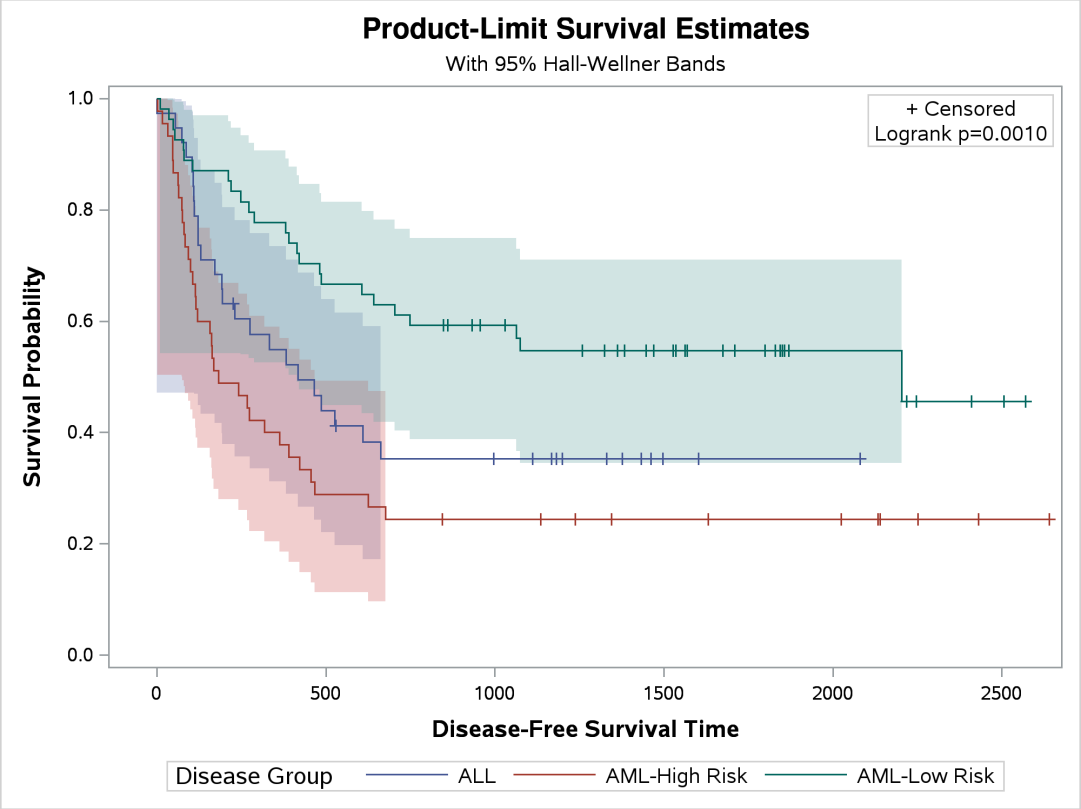

Font options include the following:

COLOR=*style-reference* | *color*
FAMILY=*style-reference* | *'string'*
SIZE=*style-reference* | *dimension*
STYLE=*style-reference* | **NORMAL** | **ITALIC**
WEIGHT=*style-reference* | **NORMAL** | **BOLD**

Fonts vary from installation to installation. Sample font strings include: 'Times New Roman', 'Courier New', 'Arial', and 'Calibri'. For more information about text and label attribute options, see *SAS Graph Template Language: Reference*. For information about changing fonts in ODS styles, see the section "Displaying a Style and Extracting Font Information" on page 1021. ODS Graphics can use a single style element in more than one place in a graph; this example shows how to change individual graph components.

Changing the Legend and Inset Position

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

This example shows you how to move the legend inside the plot (to the top right) and move the homogeneity test and censored value legend to the bottom right of the plot:

```
%ProvideSurvivalMacros
/*-- Original Macro Variable Definitions ----------------------------------
%let InsetOpts  = autoalign=(TOPRIGHT BOTTOMLEFT TOP BOTTOM)
                  border=true BackgroundColor=GraphWalls:Color Opaque=true;
%let LegendOpts = title=GROUPNAME location=outside;
-------------------------------------------------------------------------*/

%let InsetOpts  = autoalign=(BottomRight)
                  border=true BackgroundColor=GraphWalls:Color Opaque=true;
%let LegendOpts = title=GROUPNAME location=inside across=1 autoalign=(TopRight);
%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

This example shows you how to replace the AUTOALIGN=(TOPRIGHT BOTTOMLEFT TOP BOTTOM) option in the macro variable InsetOpts with AUTOALIGN=(BOTTOMRIGHT) and add the AUTOALIGN=(TOPRIGHT) option to the LegendOpts macro variable. You can also add the option ACROSS=1 to the LegendOpts macro variable to stack all legend entries vertically (with just one element in each row). The results are displayed in Figure 26.27.

Figure 26.27 Controlling Legend Placement

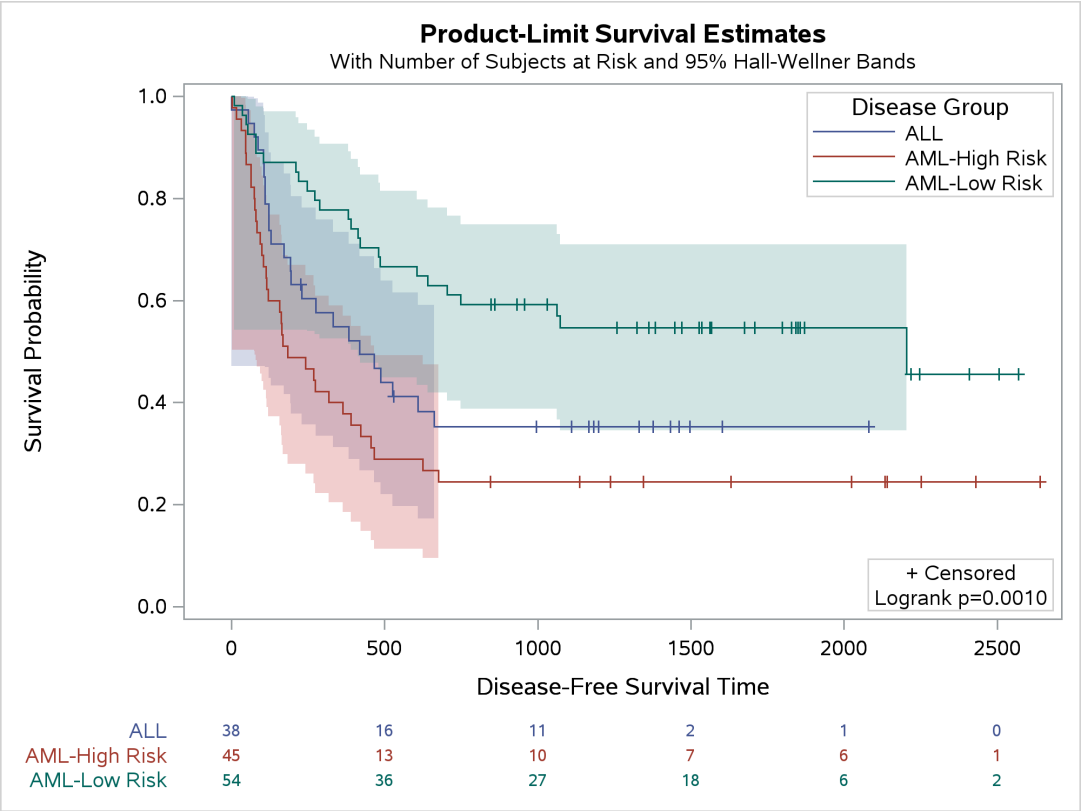

Changing How the Censored Points Are Displayed

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

By default, PROC LIFETEST displays a plus sign to indicate censoring. This example illustrates how to change the plus sign to a small filled circle in both the step plots and the inset box. The following steps change the template and create Output 26.28:

```
/*-- Original Macro Variable Definitions ----------------------------------
%let Censored   = markerattrs=(symbol=plus);
%let CensorStr  = "+ Censored";
-------------------------------------------------------------------------*/

%ProvideSurvivalMacros

%let censored  = markerattrs=(symbol=circlefilled size=3px);
%let censorstr = "(*ESC*){Unicode '25cf'x} Censored"
                 / textattrs=GraphValueText(family=GraphUnicodeText:FontFamily);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw atrisk(outside maxlen=13));
   time T * Status(0);
```

```
   strata Group;
run;
```

Figure 26.28 Survival Plot with a Modified Display of Censoring

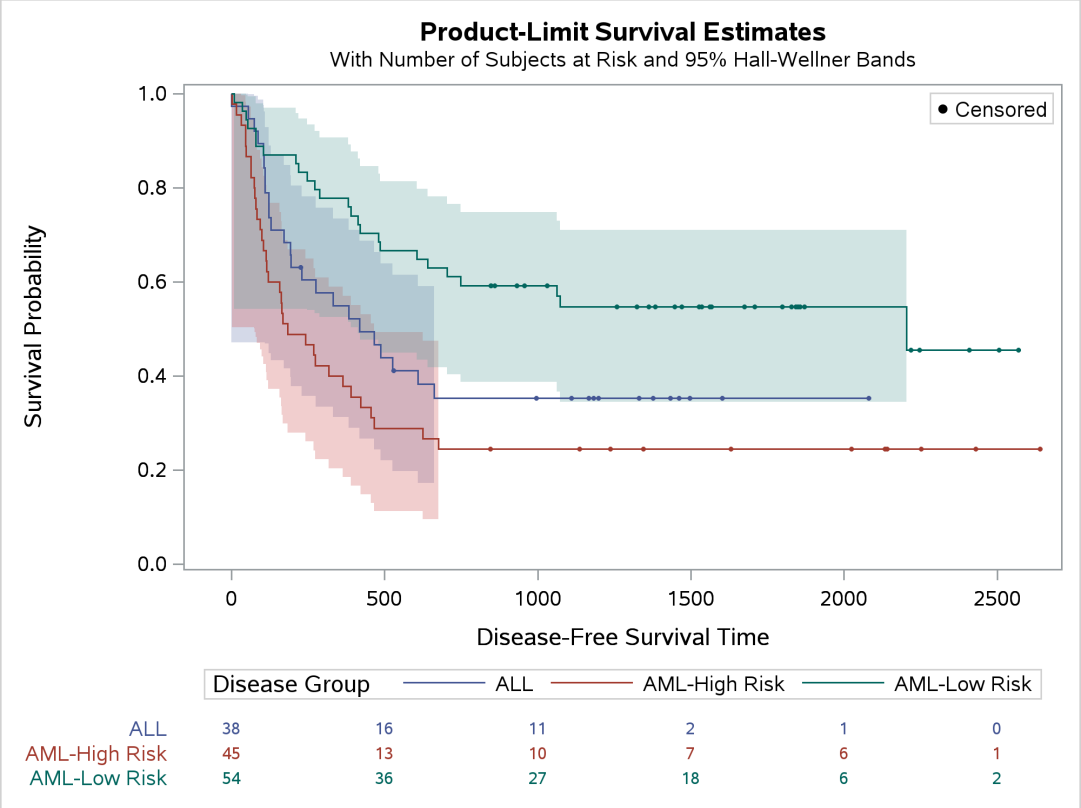

The Unicode Consortium (`http://unicode.org/`) provides a list of character codes. Also see Figure 25.2.7 in Chapter 25, "ODS Graphics Template Modification," for information about the Unicode specification for other markers. Although some Unicode characters are supported in some fonts, you should always specify a Unicode font when using special characters.

Adding a Y-Axis Reference Line

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

You can add a horizontal reference line to the survival plot by adding the following statement to the template:

```
referenceline y=0.5;
```

You can do this by using the %StmtsTop macro. By default, this macro is empty. You can use the %StmtsTop macro to add new statements to the beginning of the block of statements that define the appearance of the graph. In contrast, you can use the %StmtsBottom macro to provide statements at the end of the statement block. ODS Graphics draws statements in the order in which they appear; therefore, reference lines should be drawn first so they do not obscure other parts of the graph.

The following step creates the plot in Figure 26.29:

```
%ProvideSurvivalMacros
%macro StmtsTop;
   referenceline y=0.5;
%mend;
%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

Figure 26.29 Horizontal Reference Line

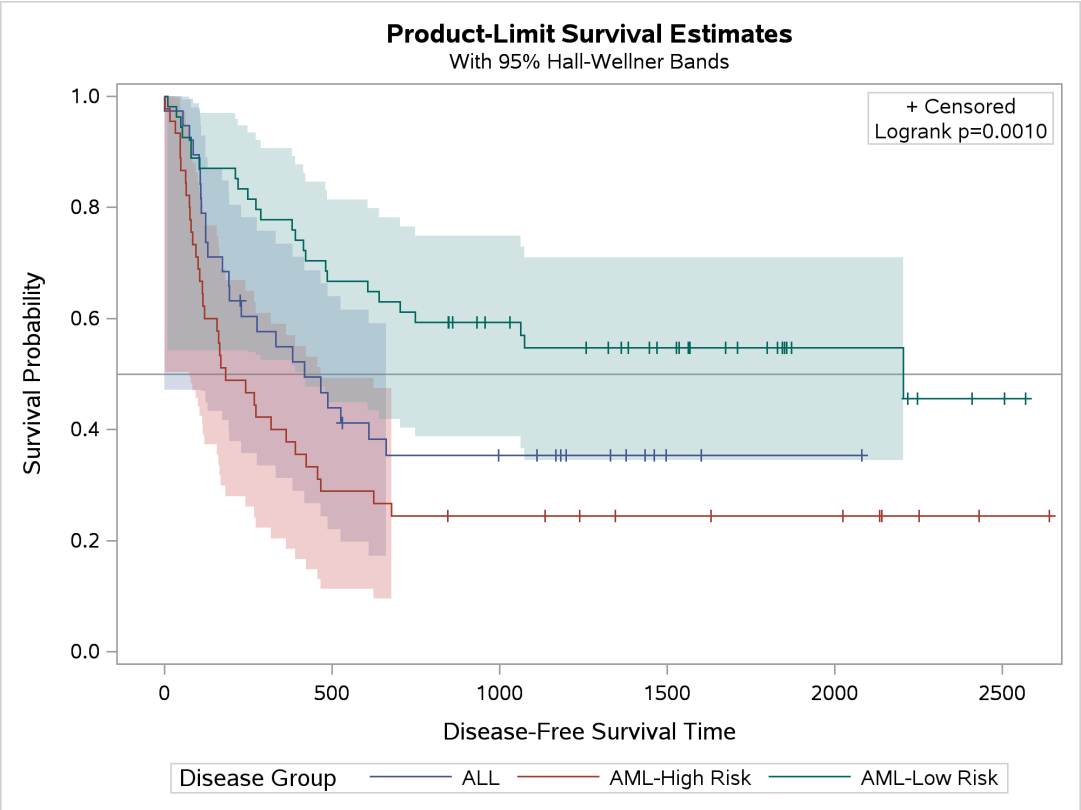

Adding Y-Axis Grid Lines

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

You can add horizontal grid lines to the survival plot by adding the following option to the Y-axis options:

```
griddisplay=on;
```

You can do this by modifying the yOptions macro variable. The following steps create the plot in Figure 26.30:

```
%ProvideSurvivalMacros

%let yoptions = &yoptions griddisplay=on;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

Figure 26.30 Horizontal Grid Lines

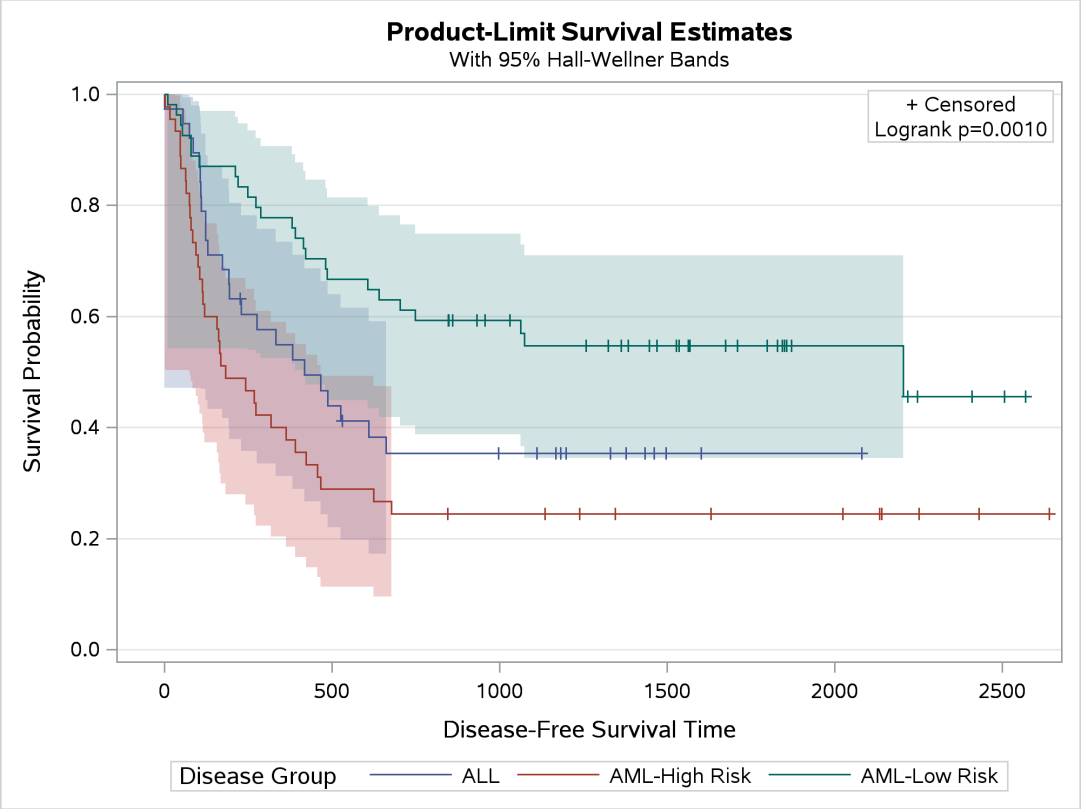

Changing the Homogeneity Test Inset

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

This example modifies the contents of the %pValue macro. The original %pValue macro definition is as follows:

```
%macro pValue;
   if (PVALUE < .0001)
      entry TESTNAME " p " eval (PUT(PVALUE, PVALUE6.4));
   else
      entry TESTNAME " p=" eval (PUT(PVALUE, PVALUE6.4));
   endif;
%mend;
```

The following example directly specifies the test name (replacing the internal name 'Logrank' with 'Log Rank') and adds blank spaces around the equal sign:

```
%ProvideSurvivalMacros

%macro pValue;
   if (PVALUE < .0001)
      entry "Log Rank p "   eval (PUT(PVALUE, PVALUE6.4));
   else
      entry "Log Rank p = " eval (PUT(PVALUE, PVALUE6.4));
   endif;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.31.

Figure 26.31 Cosmetic Inset Entry Change

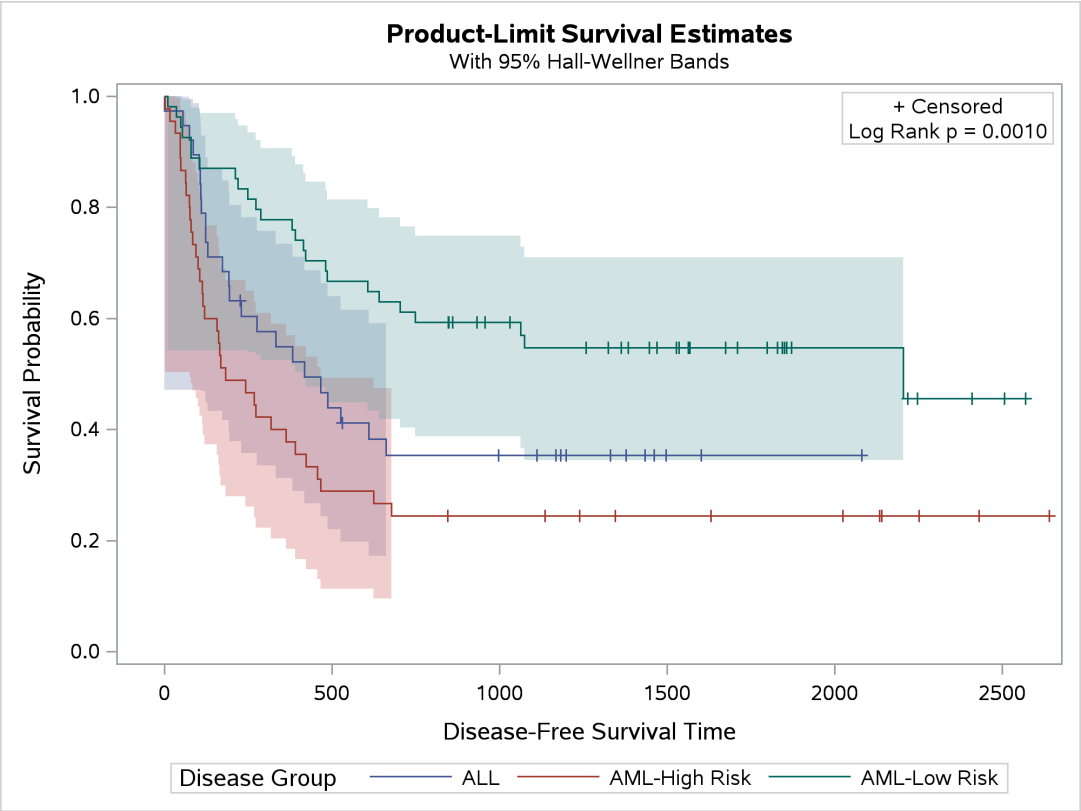

Because this template modification replaces a character string that is more appropriately set by PROC LIFETEST, you should clean up afterward as follows:

```
%ProvideSurvivalMacros

proc template;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival  /
          store=sasuser.templat;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival2 /
          store=sasuser.templat;
run;
```

Changing the Second Title and Adding a Footnote

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

The following steps add an ENTRYFOOTNOTE statement to the %StmtsBeginGraph macro and suppress the second title:

```
%ProvideSurvivalMacros

%let ntitles = 1;
%macro StmtsBeginGraph;
   entryfootnote halign=left "Acme Company, %sysfunc(date(),worddate.)" /
                 textattrs=GraphDataText;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.32.

Figure 26.32 Footnote but No Second Title

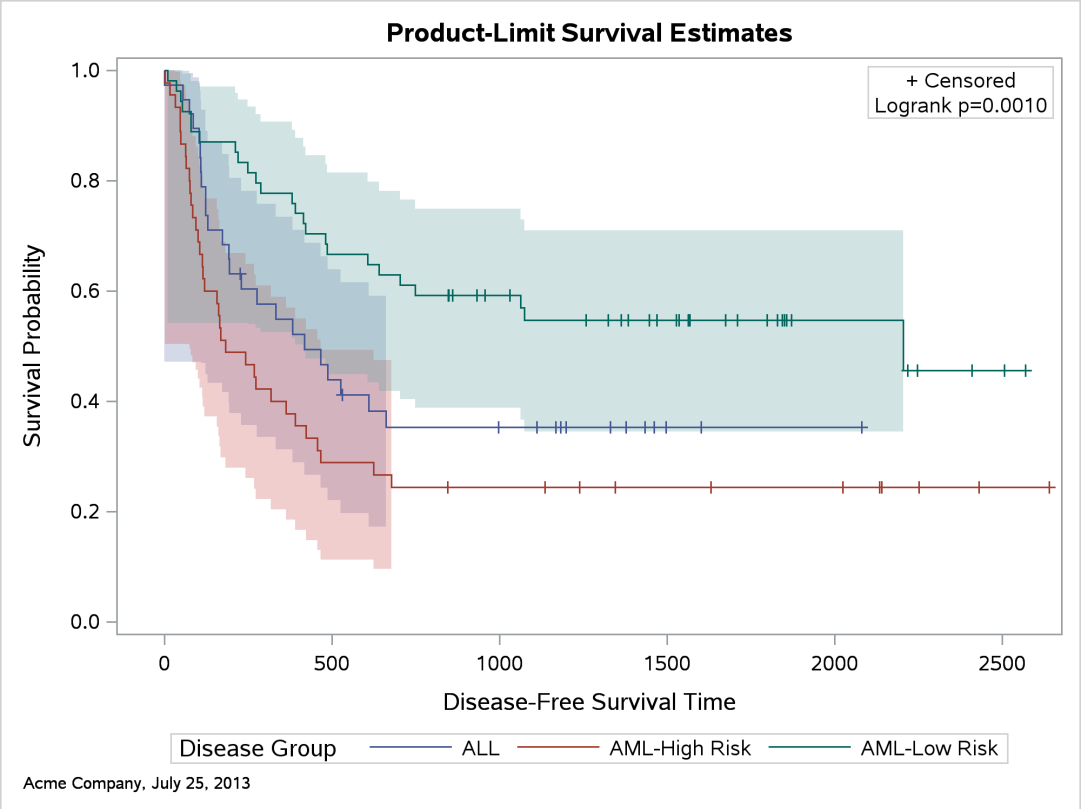

By default, the value of the nTitles macro variable is 2, and all titles are displayed. Setting nTitles to 1 suppresses the second title. You can add titles or footnotes to the plot by adding them to the %StmtsBeginGraph macro. This example adds a footnote that consists of a company name followed by the current date, formatted by using the WORDDATE format. The **GraphDataText** style element is used; it has a smaller font than the default style element, **GraphFootnoteText**.

The following steps replace the second title rather than adding a footnote:

```
%ProvideSurvivalMacros

%let ntitles = 1;

%macro StmtsBeginGraph;
entrytitle "Acme Company, %sysfunc(date(),worddate.)" /textattrs=GraphValueText;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
   plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.33. You can find the original definition of the second title in the %CompileSurvivalTemplates macro in the section "The Larger Macros" on page 993.

Figure 26.33 Replacing the Second Title

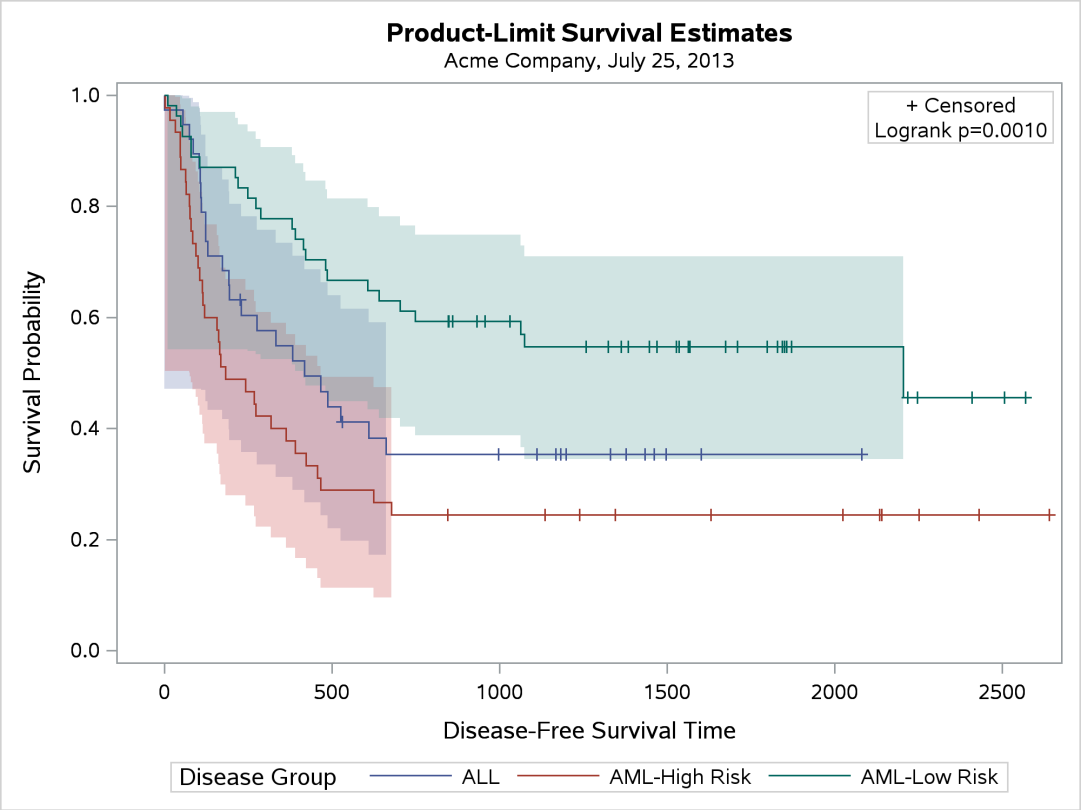

Adding a Custom Header above the At-Risk Table

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

The following steps add a header to the at-risk table by using the %StmtsBeginGraph macro and a DRAWTEXT statement:

```
%ProvideSurvivalMacros

%macro StmtsBeginGraph;
  * Coordinates are ad hoc, and there are many available coordinate systems.
    See the documentation for more information.;
  drawtext textattrs=(weight=Bold) 'Number at Risk' / x=5 y=14 width=9;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(atrisk outside maxlen=13);
   time T * Status(0);
   strata Group;
run;
```

The %StmtsBeginGraph macro adds statements near the top of the template. The DRAWTEXT statement adds text to the graph. The width of 9% was chosen so that the header would break between `'Number'` and `'at Risk'`. The other statements in the DRAW family include BEGINPOLYGON, BEGINPOLYLINE, DRAWARROW, DRAWIMAGE, DRAWLINE, DRAWOVAL, and DRAWRECTANGLE. They provide the same capabilities in graph templates that SG annotation provides in the SG procedures. For more information, see *SAS Graph Template Language: User's Guide*.

Figure 26.34 At-Risk Table Header

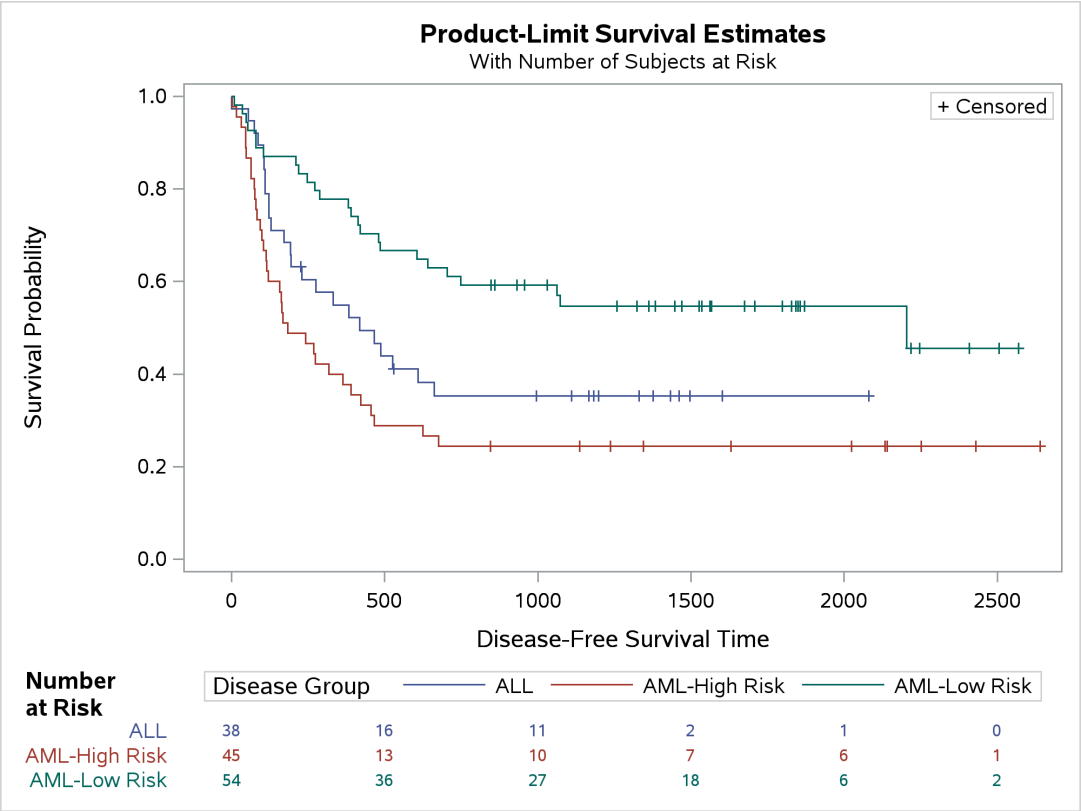

Adding a Small Inset Table with Event Information

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

This example shows you how to modify the template to produce the plot displayed in Output 26.35. This new plot has an inset table in the top right corner that shows the number of observations and the number of events in each stratum. The legend has been moved inside the plot and combined with the old inset table that shows the marker for censored observations.[6] Also, the title is changed to 'Kaplan-Meier Plot'.

```
%ProvideSurvivalMacros

%let TitleText2 = "Kaplan-Meier Plot";
%let LegendOpts = title="+ Censored"
                  location=inside autoalign=(Bottom);
%let InsetOpts  = ;

%macro StmtsBottom;
   dynamic %do i = 1 %to 3; StrVal&i NObs&i NEvent&i %end;;
   layout gridded / columns=3 border=TRUE autoalign=(TopRight);
      entry ""; entry "Event"; entry "Total";
```

[6]This legend is wide and might not be displayed if your graph is small. If the legend is not displayed, try increasing the size of the graph by specifying the WIDTH= or HEIGHT= option in the ODS GRAPHICS statement.

```
      %do i = 1 %to 3;
         %let t = / textattrs=GraphData&i;
         entry halign=right Strval&i &t; entry NEvent&i &t; entry NObs&i &t;
      %end;
   endlayout;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.35.

Figure 26.35 New Inset Table with Event Information

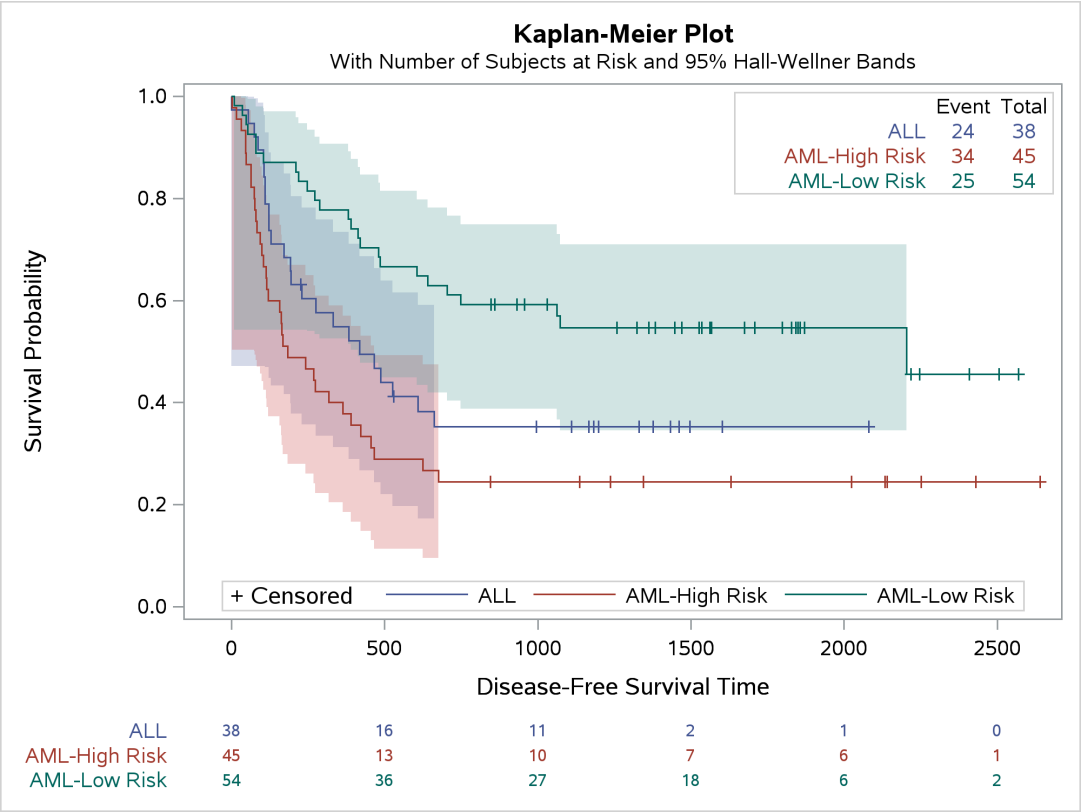

The macro variable TitleText2, which controls the title for the multiple-strata plot, is changed. You can change all three title macro variables, as is done in the construction of Figure 26.20, or you can change only TitleText2 when you have multiple overlaid strata, as in this example. The LegendOpts macro variable value was changed from TITLE=GROUPNAME LOCATION=OUTSIDE to display the censored value legend in place of the legend title and to display the legend inside the bottom of the plot. When the InsetOpts macro variable is null, the usual inset that contains the censored value and *p*-value is not displayed.

The %StmtsBottom macro (null by default) is replaced with a macro that creates the new inset table. This macro adds statements to the bottom of the templates. If you ignore for a moment most of the options, the

core of the generated statements is as follows:

```
dynamic StrVal1 NObs1 NEvent1 StrVal2 NObs2 NEvent2 StrVal3 NObs3 NEvent3;
layout gridded / columns=3;
   entry "";        entry "Event";   entry "Total";
   entry Strval1;   entry NEvent1;   entry NObs1;
   entry Strval2;   entry NEvent2;   entry NObs2;
   entry Strval3;   entry NEvent3;   entry NObs3;
endlayout;
```

The macro first constructs a DYNAMIC statement that includes the names of the dynamic variables that contain some of the results. PROC LIFETEST creates these dynamic variables and sets them to values, but you must declare them in your template before using them. For more information about these dynamic variables, see the section "Additional Dynamic Variables" on page 1001. The macro then constructs a 4×3 grid that contains a table consisting of a title line and a row for each stratum (which consists of the stratum label, the number of events, and the total number of subjects). The full layout that the %StmtsBottom macro generates, with all the options, is as follows:

```
dynamic StrVal1 NObs1 NEvent1 StrVal2 NObs2 NEvent2 StrVal3 NObs3 NEvent3;
layout gridded / columns=3 border=TRUE autoalign=(TopRight);
   entry "";
   entry "Event";
   entry "Total";
   entry halign=right Strval1 / textattrs=GraphData1;
   entry NEvent1 / textattrs=GraphData1;
   entry NObs1 / textattrs=GraphData1;
   entry halign=right Strval2 / textattrs=GraphData2;
   entry NEvent2 / textattrs=GraphData2;
   entry NObs2 / textattrs=GraphData2;
   entry halign=right Strval3 / textattrs=GraphData3;
   entry NEvent3 / textattrs=GraphData3;
   entry NObs3 / textattrs=GraphData3;
endlayout;
```

Adding an External Table with Event Information

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

This example adds a table to the plot that displays a summary of event information. The following statements create Figure 26.36:

```
%ProvideSurvivalMacros

%let GraphOpts = DesignHeight=DefaultDesignWidth;

%SurvivalSummaryTable

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw atrisk(outside maxlen=13));
   time T * Status(0);
```

```
   strata Group;
run;
```

Figure 26.36 External Event Table

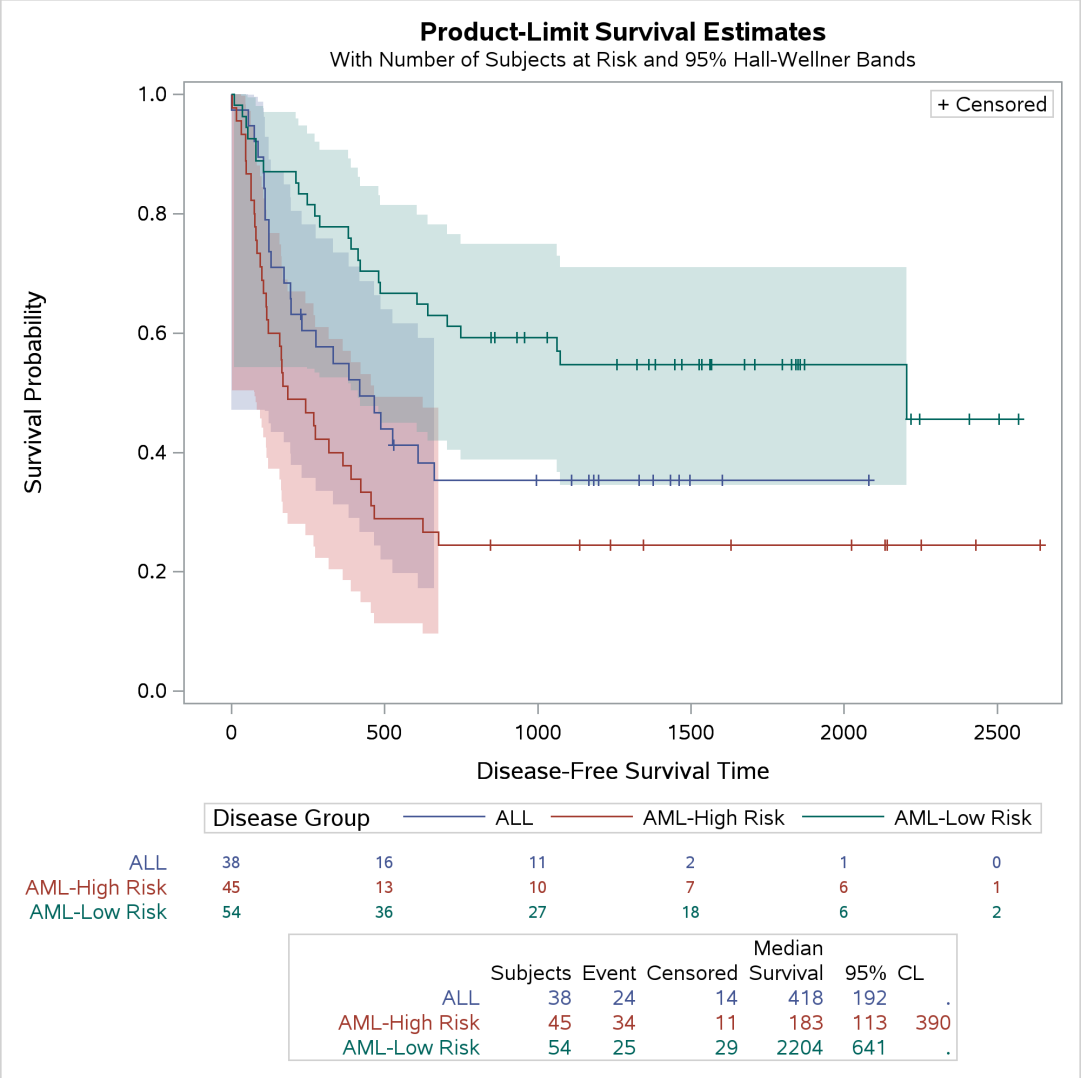

The GraphOpts macro variable specifies the option DESIGNHEIGHT=DEFAULTDESIGNWIDTH. At the default graph size (the size at which the graph is designed), this option sets the graph height to the default graph width of 640 pixels. The macro %SurvivalSummaryTable adds new statements to the graph templates that display the number of subjects, number of events, number of censored observations, median survival time, and 95% confidence limits for the median survival time. For more information about the %SurvivalSummaryTable macro, see the section "Event Table Macros" on page 997.

Suppressing the Legend

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

The plot in Figure 26.36 has a legend. However, the plot displays values in the tables by using colors that match the colors of the step functions, so you do not need the legend. The next statements show how to remove the legend:

```
%ProvideSurvivalMacros

%let GraphOpts  = DesignHeight=500px;
%let LegendOpts = ;

%SurvivalSummaryTable

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw atrisk(maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

Figure 26.37 Plot with Legend Removed

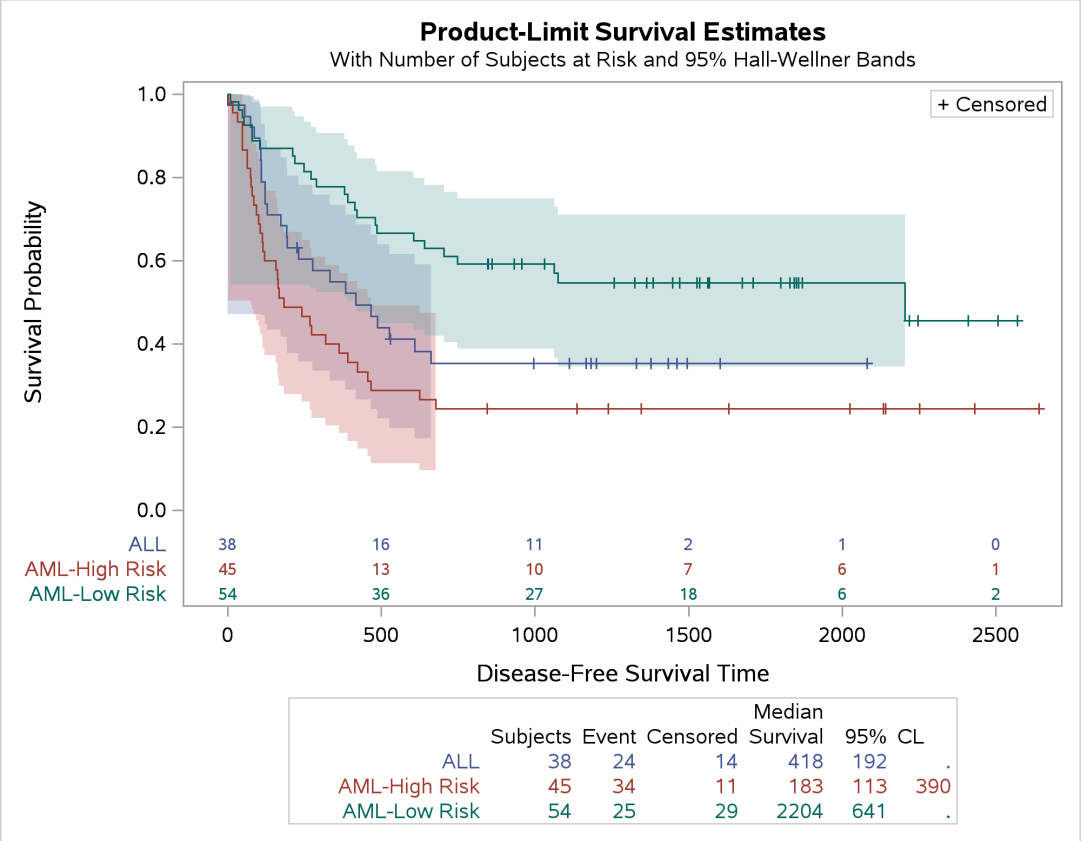

The legend is suppressed when the LegendOpts macro variable is null. This example also illustrates changing the design height to 500 pixels and moving the at-risk table back inside the body of the plot.

Kaplan-Meier Plot with Event Table and Other Customizations

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

This example combines a number of features from previous examples. The order of the strata levels in the tables is ALL, AML–Low Risk, and AML–High Risk (see the section "Reordering the Groups" on page 949). The title is set to 'Kaplan-Meier Plot' (see the section "Changing the Plot Title" on page 960). The second title line is suppressed (see the section "Changing the Second Title and Adding a Footnote" on page 976). The graph height is set to 500 pixels (see the section "Suppressing the Legend" on page 983). The legend and the inset box that contains the legend for censored values are both suppressed (see the sections "Suppressing the Legend" on page 983 and "Adding a Small Inset Table with Event Information" on page 979). The event table is displayed outside the plot (see the section "Adding an External Table with Event Information" on page 981) and the at-risk table is displayed inside the plot (see the section "Displaying the Patients-at-Risk Table inside the Plot" on page 943).

```
proc format;
   invalue bmtnum 'ALL' = 1  'AML-Low Risk' = 2  'AML-High Risk' = 3;
   value   bmtfmt 1 = 'ALL'  2 = 'AML-Low Risk'  3 = 'AML-High Risk';
run;

data BMT(drop=g);
   set sashelp.BMT(rename=(group=g));
   Group = input(g, bmtnum.);
run;

%ProvideSurvivalMacros

%let TitleText2 = "Kaplan-Meier Plot";
%let nTitles    = 1;
%let GraphOpts  = DesignHeight=500px;
%let LegendOpts = ;
%let InsetOpts  = ;

%SurvivalSummaryTable

%CompileSurvivalTemplates

proc lifetest data=BMT plots=survival(cb=hw atrisk(maxlen=13));
   time T * Status(0);
   strata Group / order=internal;
   format group bmtfmt.;
run;
```

The results are displayed in Figure 26.38.

Figure 26.38 Kaplan-Meier Plot with Extensive Customizations

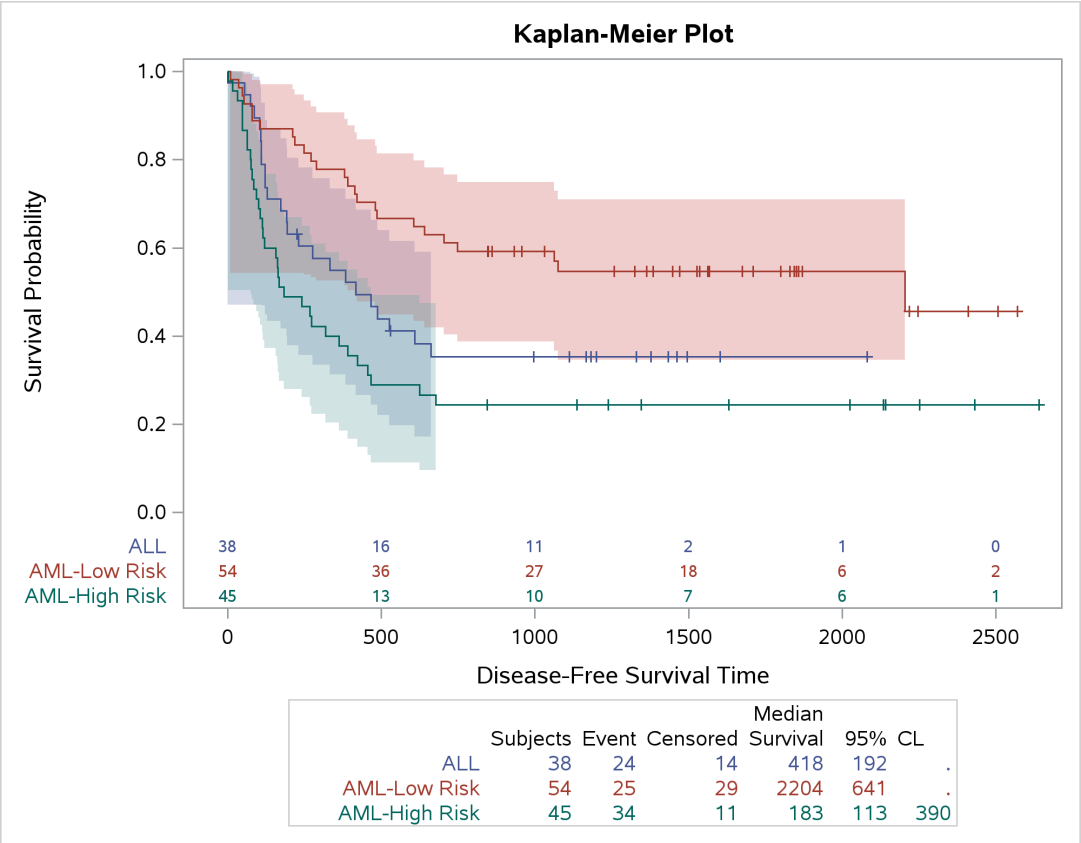

Displaying Percentages on the Y Axis

Begin by including the macros that you copied in the section "The Modularized Templates" on page 958.

There might be times when you want to use the customization macros *and* perform a subsequent template modification. If you want to make all the changes that appear in Figure 26.38 and display percentages on the Y axis, then you can modify the already-modified template. If you want to display only percentages, then you can omit the first round of template modifications. Begin by making the following template modifications, as in the section "Kaplan-Meier Plot with Event Table and Other Customizations" on page 984:

```
proc format;
   invalue bmtnum 'ALL' = 1  'AML-Low Risk' = 2  'AML-High Risk' = 3;
   value   bmtfmt 1 = 'ALL'  2 = 'AML-Low Risk'  3 = 'AML-High Risk';
run;

data BMT(drop=g);
   set sashelp.BMT(rename=(group=g));
   Group = input(g, bmtnum.);
run;

%ProvideSurvivalMacros
```

```
%let TitleText2 = "Kaplan-Meier Plot";
%let nTitles    = 1;
%let GraphOpts  = DesignHeight=500px;
%let LegendOpts = ;
%let InsetOpts  = ;

%SurvivalSummaryTable

%CompileSurvivalTemplates
```

Next, write the modified template to a file:

```
proc template;
   source Stat.Lifetest.Graphics.ProductLimitSurvival / file='temp.tmp';
quit;
```

A subsequent step reads the file, inserts a PROC TEMPLATE statement at the beginning, and inserts a QUIT statement at the end. Then it modifies several statements in between. Before seeing how it makes those modifications, you should look at the file *temp.tmp* and find the code that needs to be modified. The following are samples of the statement fragments that need to be modified:

```
yaxisopts=(label="Survival Probability"
linearopts=(viewmin=0 viewmax=1
tickvaluelist=(0 .2 .4 .6 .8 1.0)));
bandplot LimitUpper=EP_UCL LimitLower=EP_LCL x=TIME
bandplot LimitUpper=HW_UCL LimitLower=HW_LCL x=TIME
scatterplot y=CENSORED x=TIME
stepplot y=SURVIVAL x=TIME
```

Each of these statement fragments occurs in more than one place. There are ten BANDPLOT statements and two each of the SCATTERPLOT and STEPPLOT statements. You need to change `'Survival Probability'` to `'Survival Percentage'`, VIEWMAX=1 to VIEWMAX=100, and the tick value list from proportions to percentages. Finally, you need to change all the Y-axis variables (which are specified in many statements) from proportions to percentages by specifying an expression. For example, you need to change **`LimitUpper=EP_UCL LimitLower=EP_LCL`** to **`LimitUpper=eval(100*EP_UCL) LimitLower=eval(100*EP_LCL)`**. The following step makes all these changes:

```
data _null_;
   length var stmt $ 32;
   infile 'temp.tmp' end=eof;
   input;
   if _n_ eq 1 then call execute('proc template;');
   stmt = scan(_infile_, 1);
   if stmt in ('scatterplot', 'stepplot', 'bandplot') then do;
      var = scan(scan(_infile_, 2, '='), 1);
      _infile_ = tranwrd(_infile_, trim(var), cats('eval(100*',var,')'));
      if stmt = 'bandplot' then do;
         var = scan(scan(_infile_, 3, '='), 1);
         _infile_ = tranwrd(_infile_, trim(var), cats('eval(100*',var,')'));
      end;
   end;
   _infile_ = tranwrd(_infile_, 'Survival Probability', 'Survival Percentage');
   _infile_ = tranwrd(_infile_, '0 .2 .4 .6 .8 1.0', '0 20 40 60 80 100');
   _infile_ = tranwrd(_infile_, 'viewmax=1', 'viewmax=100');
```

```
   call execute(_infile_);
   if eof then call execute('quit;');
run;
```

Statements in this DATA step begin by doing the following: read the template from the file, submit a PROC TEMPLATE statement to a buffer, find the name of each statement, and find the name of the first variable (which for this template is always a Y-axis variable). The DATA step scans for the second string by using an equal sign as a delimiter and then scans that result for the first string by using the standard delimiters (which include blanks). This step extracts only the variable name that follows the equal sign. The TRANWRD function changes the variable name to an expression. Band plots have two Y-axis specifications (for the upper and lower limits), so the next code block makes the same change to the second specification. The final TRANWRD functions change the Y-axis label, the tick values, and the maximum Y-axis value. The second-last CALL EXECUTE statement submits to a buffer all the template statements, including the modified statements. At the end, the DATA step submits a QUIT statement to the buffer. When the DATA step finishes, SAS submits all the code in the buffer, which modifies the template. The following step uses the modified template and creates a plot that has percentages on the Y axis:

```
proc lifetest data=BMT plots=survival(cb=hw atrisk(maxlen=13));
   time T * Status(0);
   strata Group / order=internal;
   format group bmtfmt.;
run;
```

The graph is displayed in Figure 26.39.

Figure 26.39 Survival Percentages

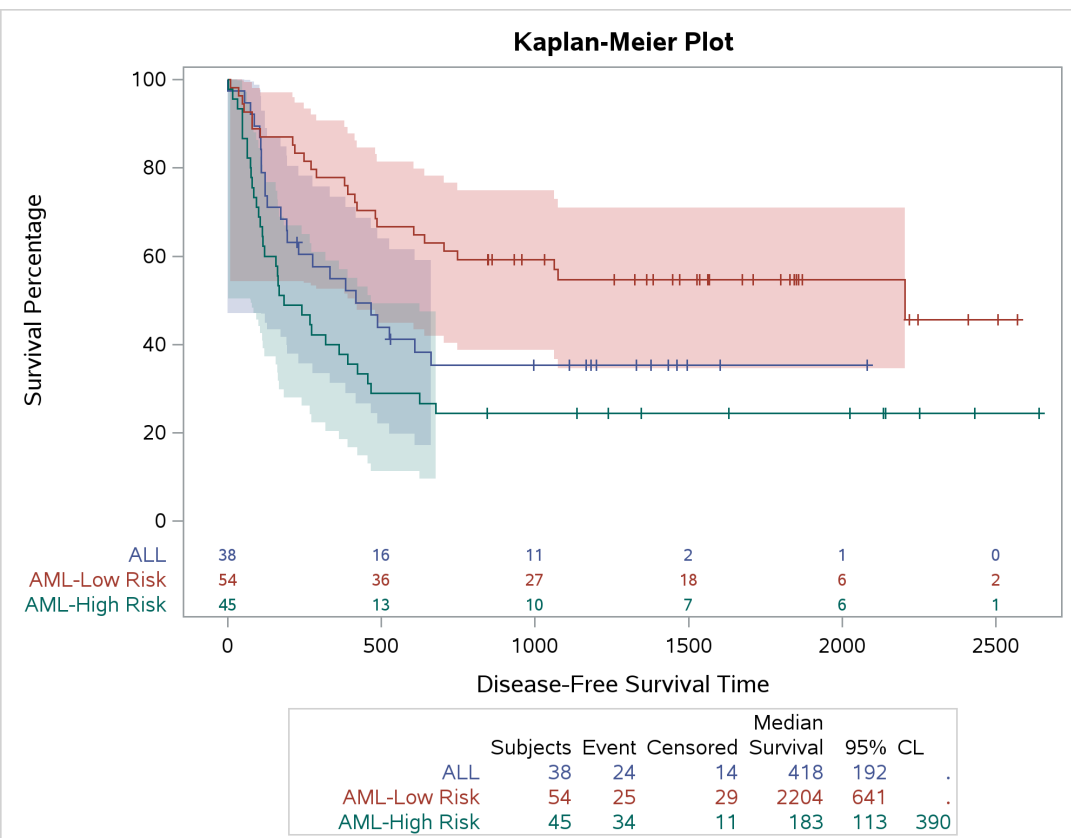

Compiled Template Cleanup

The following step restores all the default macros and macro variables and deletes the modified templates:

```
%ProvideSurvivalMacros

proc template;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival  /
          store=sasuser.templat;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival2 /
          store=sasuser.templat;
run;
```

For more information about deleting compiled templates, see the section "SAS Item Stores" on page 1025.

Graph Templates, Macros, and Macro Variables

The %ProvideSurvivalMacros macro and the macros and macro variables that it provides have the following properties:

- Many options, including most of the options that are specified in multiple places in the templates, are extracted to macro variables.
- The %CompileSurvivalTemplates macro provides the main body of the two templates. You can call it to compile the templates after making changes.
  - The template **Stat.Lifetest.Graphics.ProductLimitSurvival** provides the survival template when the at-risk table is inside the body of the plot.
  - The template **Stat.Lifetest.Graphics.ProductLimitSurvival2** provides the survival template when the at-risk table is outside the body of the plot.[7]

  The two templates share many statements, and a macro %DO loop creates both versions.
- The portion of the templates for the table for the *p*-values is stored in the macro %pValue.
- The portion of the templates for the single-stratum case is stored in the macro %SingleStratum.
- The portion of the templates for the multiple-strata case is stored in the macro %MultipleStrata.
- The macro %AtRiskLatticeStart begins the two-cell lattice that contains the plot above the table when the at-risk table is outside the body of the plot.
- The macro %AtRiskLatticeEnd ends the two-cell lattice that contains the plot and the table when the at-risk table is outside the body of the plot.

[7] The macros do not affect any graph that uses graph templates other than the two templates that are modified here. The macros do not affect the STRATA=PANEL plot that uses the template **Stat.Lifetest.Graphics.ProductLimitSurvivalPanel** or the failure plot that uses the template **Stat.Lifetest.Graphics.ProductLimitFailure**.

- Some empty macros (%StmtsBeginGraph, %StmtsTop, and %StmtsBottom) are provided to enable you to add statements and options to strategic places in the templates.
- The %SurvTabHeader, %SurvivalTable, and %SurvivalSummaryTable macros enable you to easily add more GTL statements to the Kaplan-Meier plot templates to display event information for each stratum.

This organization makes it easy to identify the relevant parts of the templates, modify these parts, and recompile the templates. A small portion of the %ProvideSurvivalMacros macro follows:

```
%macro ProvideSurvivalMacros;

   %global atriskopts bandopts censored censorstr classopts
           graphopts groups insetopts legendopts ntitles stepopts tiplabel
           tips titletext0 titletext1 titletext2 xoptions yoptions;

   %let TitleText0 = METHOD " Survival Estimate";
   %let TitleText1 = &titletext0 " for " STRATUMID;
   %let TitleText2 = &titletext0 "s";         /* plural: Survival Estimates */

   %let yOptions   = label="Survival Probability" shortlabel="Survival"
                     linearopts=(viewmin=0 viewmax=1
                                 tickvaluelist=(0 .2 .4 .6 .8 1.0));

   %let xOptions   = shortlabel=XNAME offsetmin=.05
                     linearopts=(viewmax=MAXTIME tickvaluelist=XTICKVALS
                                 tickvaluefitpolicy=XTICKVALFITPOL);
   . . .

   %macro CompileSurvivalTemplates; . . . %mend;
   %macro pValue;                   . . . %mend;
   %macro SingleStratum;            . . . %mend;
   %macro MultipleStrata;           . . . %mend;
   . . .
%CompileSurvivalTemplates
%mend;
```

The Macro Variables

The macros and macro variables are designed so that most of the time you need to modify only the macro variables and not the larger macros. However, you have the full flexibility to modify both. You can modify any of the following macro variables:

```
%let TitleText0 = METHOD " Survival Estimate";
%let TitleText1 = &titletext0 " for " STRATUMID;
%let TitleText2 = &titletext0 "s";         /* plural: Survival Estimates */
%let nTitles    = 2;

%let yOptions   = label="Survival Probability" shortlabel="Survival"
                  linearopts=(viewmin=0 viewmax=1
                              tickvaluelist=(0 .2 .4 .6 .8 1.0));
```

```
%let xOptions   = shortlabel=XNAME offsetmin=.05
                  linearopts=(viewmax=MAXTIME tickvaluelist=XTICKVALS
                              tickvaluefitpolicy=XTICKVALFITPOL);

%let Tips       = rolename=(_tip1= ATRISK _tip2=EVENT)
                  tiplabel=(_tip1="Number at Risk" _tip2="Observed Events")
                  tip=(x y _tip1 _tip2);
%let TipLabel   = tiplabel=(y="Survival Probability");
%let StepOpts   = ;

%let Groups     = group=STRATUM index=STRATUMNUM;

%let BandOpts   = displayTail=false &groups modelname="Survival";

%let InsetOpts  = autoalign=(TOPRIGHT BOTTOMLEFT TOP BOTTOM)
                  border=true BackgroundColor=GraphWalls:Color Opaque=true;
%let LegendOpts = title=GROUPNAME location=outside;

%let AtRiskOpts = display=(label) valueattrs=(size=7pt);
%let ClassOpts  = class=CLASSATRISK colorgroup=CLASSATRISK;

%let Censored   = markerattrs=(symbol=plus);
%let CensorStr  = "+ Censored";

%let GraphOpts  = ;
```

The %ProvideSurvivalMacros macro declares that these macro variables are global in scope, so you can assign values to them in your programs and have them affect the internal macros. These macro variables specify a variety of GTL options; for more information, see *SAS Graph Template Language: Reference*. The macro variables are as follows.

TitleText0 provides the common text that is used in the title for the single-stratum and multiple-strata cases. METHOD is a dynamic variable that PROC LIFETEST sets. In these examples, the value of METHOD is 'Product-Limit'; the product-limit method is also known as the Kaplan-Meier (1958) method.

TitleText1 provides the title text for the single-stratum title (relying on TitleText0).

TitleText2 provides the title text for the multiple-strata title (relying on TitleText0).

nTitles specifies the number of titles. Set the macro variable nTitles to 1 to suppress the second title line or 0 to suppress all title lines. You can add titles to the plot by adding ENTRYTITLE statements to the top of the %StmtsBeginGraph macro even when you suppress the usual titles by setting the nTitles macro variable to 0 or 1. By default, nTitles equals 2.

yOptions provides the Y-axis options. The LABEL= option provides the axis label. The SHORTLABEL= option provides the axis label for small plots when the LABEL= option label is too long. The LINEAROPTS= option specifies linear axis options. This and most other axes are linear axes; alternatives include log-scale axes. The VIEWMIN=0 and VIEWMAX=1 options ensure that the axis goes from 0 to 1 even when the actual results have a more restricted range. The TICKVALUELIST= option provides the tick values. Standard SAS number list abbreviations like 0 TO 1 BY 0.2 are not valid in the GTL.

xOptions provides the X-axis options. The LABEL= option is not provided, so the axis label comes from the column label in the ODS data object. You can add a LABEL= option or other axis options if you want. The SHORTLABEL= option provides the axis label for small plots when the label is too long. The short label comes from a dynamic variable that PROC LIFETEST provides. The OFFSETMIN= option ensures that there is extra space between the axis and the minimum tick mark. The LINEAROPTS= option specifies linear axis options. The VIEWMAX= option ensures that the axis goes to the value in the MAXTIME dynamic variable set by PROC LIFETEST. The TICKVALUELIST= option provides the tick values in a dynamic variable. The TICKVALUEFITPOLICY= option provides, in a dynamic variable, the approach for handling dense tick marks. Approaches include rotation, staggering, and thinning.

Tips provides options for tooltips for the step plots. Tooltips are text boxes that appear in HTML output when you rest your mouse pointer over part of the plot when the IMAGEMAP=ON option is specified in the ODS GRAPHICS statement. Tooltips are provided for the X- and Y-axis columns. Additional columns that are assigned roles (and hence are eligible to use as tooltips) include the at-risk and event columns. These columns are given the tooltip labels 'Number at Risk' and 'Observed Events'. Unless you are specifically interested in tooltips, you probably do not need to modify this macro variable.

TipLabel provides a label for the Y-axis tooltip. Unless you are specifically interested in tooltips, you do not need to modify this macro variable.

StepOpts provides options for the step functions. This macro variable is null by default. You can use this option to control the line thickness (for example, LINEATTRS=(THICKNESS=2.5)) and other aspects of the step functions.

Groups provides the name of the data object columns that provide group names and the index that provides the order of the group names. You will probably never need to modify this macro variable.

BandOpts provides the group information for band plots. You will probably never need to modify this macro variable.

InsetOpts provides options for the inset table that provides the censored value legend and the homogeneity test *p*-value. The AUTOALIGN= option specifies the places in the plot where the inset table can be positioned. If your preferred placement is somewhere other than the top right corner, you can modify the automatic placement list. The BORDER= option displays a border around this table. The BACKGROUNDCOLOR= option controls the table background. By default, it matches the background color for the rest of the plot by using the **`GraphWalls:Color`** style reference. The OPAQUE=TRUE option specifies an opaque table that hides any graphical elements that are behind the table. You can set the InsetOpts macro variable to null to suppress the usual inset that contains the censored value and *p*-value.

LegendOpts provides options for the external legend that identifies the strata. The title comes from a dynamic variable GroupName that the procedure sets. By default, the legend is outside the plot. Specify LOCATION=INSIDE and an AUTOALIGN= option such as the one provided in the InsetOpts macro variable if you want the legend to appear inside the plot. You can set the LegendOpts macro variable to null to suppress the legend.

AtRiskOpts provides options for the at-risk table. The option DISPLAY=(LABEL) limits the display to labels. VALUEATTRS=(SIZE=7PT) specifies a font size of seven points.

ClassOpts provides the options that are used in the at-risk table to distinguish groups of observations.

Censored provides the marker (a plus sign) that is displayed in the plot to indicate censored observations.

CensorStr provides the character for the inset table that shows how censored observations appear in the plot.

GraphOpts provides options for the template BEGINGRAPH statement. By default, the GraphOpts macro variable is null. The following options are particularly useful:

- **ATTRPRIORITY=AUTO | NONE | COLOR** specifies the priority for varying the attributes that distinguish groups of observations. AUTO honors the setting that is otherwise in effect. COLOR varies only the color attribute. NONE simultaneously varies colors, markers, and lines. Styles such as HMTLBlue and Pearl are ATTRPRIORITY=COLOR styles, whereas styles such as DEFAULT, Statistical, Listing, and RTF are ATTRPRIORITY=NONE styles.
- **DATACOLORS=(*color-list*)** specifies the list of colors (which control confidence bands) to replace the graph data colors from the GraphData1–GraphData*N* style elements.
- **DATACONTRASTCOLORS=(*color-list*)** specifies the list of contrast colors (which control markers and lines) to replace the graph data contrast colors from the GraphData1–GraphData*N* style elements.
- **DATALINEPATTERNS=(*line-pattern-list*)** specifies the list of line patterns to replace the graph data line patterns from the GraphData1–GraphData*N* style elements. There are 46 line patterns, and you can specify each pattern by using an integer in the range 1 to 46. Some patterns have names associated with them. You can specify either the name or the number for the following number/name pairs: 1 Solid, 2 ShortDash, 4 MediumDash, 5 LongDash, 8 MediumDashShortDash, 14 DashDashDot, 15 DashDotDot, 20 Dash, 26 LongDashShortDash, 34 Dot, 35 ThinDot, 41 ShortDashDot, and 42 MediumDashDotDot.
- **DESIGNHEIGHT=*height*** sets the graph height. You can set the graph height to the default graph width of 640 pixels by specifying the option DESIGNHEIGHT=DEFAULTDESIGNWIDTH. Or you can specify a size in pixels, such as DESIGNHEIGHT=500PX. Although the graph is designed at the specified height, you can resize it for the actual display by using the WIDTH= and HEIGHT= options in the ODS GRAPHICS statement. By default, DESIGNHEIGHT=480PX.
- **DESIGNWIDTH=*width*** sets the graph width. You can set the graph width to the default graph height of 480 pixels by specifying the option DESIGNWIDTH=DEFAULTDESIGNHEIGHT. Or you can specify a size in pixels, such as DESIGNWIDTH=600PX. Although the graph is designed at the specified width, you can resize it for the actual display by using the WIDTH= and HEIGHT= options in the ODS GRAPHICS statement. By default, DESIGNWIDTH=640PX.

The Smaller Macros

The %ProvideSurvivalMacros macro provides four small macros that are easy for you to modify:

```
%macro StmtsBeginGraph; %mend;
%macro StmtsTop;        %mend;
%macro StmtsBottom;     %mend;

%macro pValue;
   if (PVALUE < .0001)
      entry TESTNAME " p " eval (PUT(PVALUE, PVALUE6.4));
   else
      entry TESTNAME " p=" eval (PUT(PVALUE, PVALUE6.4));
   endif;
%mend;
```

By default, the %StmtsBeginGraph, %StmtsTop, and %StmtsBottom macros are empty. You can use them to add new statements to the BEGINGRAPH block or to the beginning or end of the block of statements that define the appearance of the graph.

The %pValue macro is used to control the display of the *p*-value from the homogeneity test.

The Larger Macros

The examples and information up to this point have illustrated how you can make simple changes to the survival plot. It is unlikely that you will ever have to do more than that. If you need to make changes to the overall layout of the graph, then you must modify one of the other macros. The %CompileSurvivalTemplates macro, which is the macro that compiles all the pieces that you modified, is as follows:

```
%macro CompileSurvivalTemplates;
   %local outside;
   proc template;
      %do outside = 0 %to 1;
         define statgraph
            Stat.Lifetest.Graphics.ProductLimitSurvival%scan(2,2-&outside);
            dynamic NStrata xName plotAtRisk
               %if %nrbquote(&censored) ne %then plotCensored;
               plotCL plotHW plotEP labelCL labelHW labelEP maxTime xtickVals
               xtickValFitPol rowWeights method StratumID classAtRisk
               plotTest GroupName Transparency SecondTitle TestName pValue
               _byline_ _bytitle_ _byfootnote_;
            BeginGraph %if %nrbquote(&graphopts) ne %then / &graphopts;;

            if (NSTRATA=1)
               %if &ntitles %then %do;
                  if (EXISTS(STRATUMID)) entrytitle &titletext1;
                  else                   entrytitle &titletext0;
                  endif;
               %end;

               %if &ntitles gt 1 %then %do;
```

```
                %if not &outside %then if (PLOTATRISK=1);
                   entrytitle "With Number of Subjects at Risk" /
                              textattrs=GRAPHVALUETEXT;
                %if not &outside %then %do; endif; %end;
             %end;

             %StmtsBeginGraph
             %AtRiskLatticeStart
             layout overlay / xaxisopts=(&xoptions) yaxisopts=(&yoptions);
                %StmtsTop
                %SingleStratum
                %StmtsBottom
             endlayout;
             %AtRiskLatticeEnd

          else
             %if &ntitles %then %do; entrytitle &titletext2; %end;
             %if &ntitles gt 1 %then %do;
                if (EXISTS(SECONDTITLE))
                   entrytitle SECONDTITLE / textattrs=GRAPHVALUETEXT;
                endif;
             %end;

             %StmtsBeginGraph
             %AtRiskLatticeStart
             layout overlay / xaxisopts=(&xoptions) yaxisopts=(&yoptions);
                %StmtsTop
                %MultipleStrata
                %StmtsBottom
             endlayout;
             %AtRiskLatticeEnd(class)

          endif;

          if (_BYTITLE_) entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
          else if (_BYFOOTNOTE_) entryfootnote halign=left _BYLINE_; endif;
          endif;
          EndGraph;
       end;
    %end;
 run;
%mend;
```

The macro %DO loop compiles the following two templates:

- **`Stat.Lifetest.Graphics.ProductLimitSurvival`** when the macro variable Outside is 0 and %SCAN(2,2-&OUTSIDE) is null
- **`Stat.Lifetest.Graphics.ProductLimitSurvival2`** when the macro variable Outside is 1 and %SCAN(2,2-&OUTSIDE) is 2

The primary difference between these templates is that when the macro variable Outside is 1, a LAYOUT LATTICE statement block is used to place the at-risk table outside the graph. When Outside is 1, the macros %AtRiskLatticeStart and %AtRiskLatticeEnd provide the LAYOUT LATTICE statement block (two cells,

plot above and at-risk table below) and the LAYOUT OVERLAY statement block for the at-risk table. The %AtRiskLatticeStart and %AtRiskLatticeEnd macros are defined as follows:

```
%macro AtRiskLatticeStart;
   %if &outside %then %do;
      layout lattice / rows=2 rowweights=ROWWEIGHTS
                       columndatarange=union rowgutter=10;
      cell;
   %end;
%mend;

%macro AtRiskLatticeEnd(useclassopts);
   %if &outside %then %do;
      endcell;
      cell;
         layout overlay / walldisplay=none xaxisopts=(display=none);
            axistable x=TATRISK value=ATRISK / &atriskopts
                      %if &useclassopts ne %then &classopts;;
         endlayout;
      endcell;
   endlayout;
   %end;
%mend;
```

The %CompileSurvivalTemplates macro relies on two other macros: %SingleStratum for the single-stratum case and %MultipleStrata for the multiple-strata case. The %SingleStratum macro is as follows:

```
%macro SingleStratum;
   if (PLOTHW=1 AND PLOTEP=0)
      bandplot LimitUpper=HW_UCL LimitLower=HW_LCL x=TIME /
         displayTail=false modelname="Survival" fillattrs=GRAPHCONFIDENCE
         name="HW" legendlabel=LABELHW;
   endif;
   if (PLOTHW=0 AND PLOTEP=1)
      bandplot LimitUpper=EP_UCL LimitLower=EP_LCL x=TIME /
         displayTail=false modelname="Survival" fillattrs=GRAPHCONFIDENCE
         name="EP" legendlabel=LABELEP;
   endif;
   if (PLOTHW=1 AND PLOTEP=1)
      bandplot LimitUpper=HW_UCL LimitLower=HW_LCL x=TIME /
         displayTail=false modelname="Survival" fillattrs=GRAPHDATA1
         datatransparency=.55 name="HW" legendlabel=LABELHW;
      bandplot LimitUpper=EP_UCL LimitLower=EP_LCL x=TIME /
         displayTail=false modelname="Survival" fillattrs=GRAPHDATA2
         datatransparency=.55 name="EP" legendlabel=LABELEP;
   endif;
   if (PLOTCL=1)
      if (PLOTHW=1 OR PLOTEP=1)
         bandplot LimitUpper=SDF_UCL LimitLower=SDF_LCL x=TIME /
            displayTail=false modelname="Survival" display=(outline)
            outlineattrs=GRAPHPREDICTIONLIMITS name="CL" legendlabel=LABELCL;
      else
         bandplot LimitUpper=SDF_UCL LimitLower=SDF_LCL x=TIME /
            displayTail=false modelname="Survival"
```

```
            fillattrs=GRAPHCONFIDENCE
            name="CL" legendlabel=LABELCL;
         endif;
      endif;

      stepplot y=SURVIVAL x=TIME / name="Survival" &tips legendlabel="Survival"
            &stepopts;

      if (PLOTCENSORED=1)
         scatterplot y=CENSORED x=TIME / &censored &tiplabel
            name="Censored" legendlabel="Censored";
      endif;

      if (PLOTCL=1 OR PLOTHW=1 OR PLOTEP=1)
         discretelegend "Censored" "CL" "HW" "EP" / location=outside
            halign=center;
      else
         if (PLOTCENSORED=1)
            discretelegend "Censored" / location=inside
                                        autoalign=(topright bottomleft);
         endif;
      endif;
      %if not &outside %then %do;
         if (PLOTATRISK=1)
            innermargin / align=bottom;
               axistable x=TATRISK value=ATRISK / &atriskopts;
            endinnermargin;
         endif;
      %end;
   %mend;
```

The %MultipleStrata macro is as follows:

```
   %macro MultipleStrata;
     if (PLOTHW=1)
         bandplot LimitUpper=HW_UCL LimitLower=HW_LCL x=TIME / &bandopts
                  datatransparency=Transparency;
      endif;
      if (PLOTEP=1)
         bandplot LimitUpper=EP_UCL LimitLower=EP_LCL x=TIME / &bandopts
                  datatransparency=Transparency;
      endif;
      if (PLOTCL=1)
         if (PLOTHW=1 OR PLOTEP=1)
            bandplot LimitUpper=SDF_UCL LimitLower=SDF_LCL x=TIME / &bandopts
                     display=(outline) outlineattrs=(pattern=ShortDash);
         else
            bandplot LimitUpper=SDF_UCL LimitLower=SDF_LCL x=TIME / &bandopts
                     datatransparency=Transparency;
         endif;
      endif;

      stepplot y=SURVIVAL x=TIME / &groups name="Survival" &tips &stepopts;
```

```
   if (PLOTCENSORED=1)
      scatterplot y=CENSORED x=TIME / &groups &tiplabel &censored;
   endif;

   %if not &outside %then %do;
      if (PLOTATRISK=1)
         innermargin / align=bottom;
            axistable x=TATRISK value=ATRISK / &atriskopts &classopts;
         endinnermargin;
      endif;
   %end;

   %if %nrbquote(&legendopts) ne %then %do;
      DiscreteLegend "Survival" / &legendopts;
   %end;

   %if %nrbquote(&insetopts) ne %then %do;
      if (PLOTCENSORED=1)
         if (PLOTTEST=1)
            layout gridded / rows=2 &insetopts;
               entry &censorstr;
               %pValue
            endlayout;
         else
            layout gridded / rows=1 &insetopts;
               entry &censorstr;
            endlayout;
         endif;
      else
         if (PLOTTEST=1)
            layout gridded / rows=1 &insetopts;
               %pValue
            endlayout;
         endif;
      endif;
   %end;

%mend;
```

Event Table Macros

All the macros and macro variables that have been described up to this point are used in defining the two survival plot graph templates. Some macros (%StmtsTop and %StmtsBottom) and macro variables (StepOpts and GraphOpts) are null and do not affect the generated template code, but all are resolved somewhere in the process of producing the templates. In contrast, the macros %SurvTabHeader, %SurvivalTable, and %SurvivalSummaryTable are not used by default. They are available for you to use to add more statements to the templates.

The %SurvTabHeader macro provides the headings for the event table:

```
%macro SurvTabHeader(multiple);
   %if &multiple %then %do; entry ""; %end;
   entry "";
   entry &r "Median";
   entry "";

   %if &multiple %then %do; entry ""; %end;
   entry &r "Subjects";
   entry &r "Event";
   entry &r "Censored";
   entry &r "Survival";
   entry &r PctMedianConfid;
   entry halign=left  "CL";
%mend;
```

This table is not displayed by default. There are two types of headings: one for multiple strata and one for a single stratum.

The %SurvivalTable macro provides the body of the event table:

```
%macro SurvivalTable;
   %local fmt r i t;
   %let fmt = bestd6.;
   %let r = halign = right;
   columnheaders;
      layout overlay / pad=(top=5);
         if(NSTRATA=1)
            layout gridded / columns=6 border=TRUE;
               dynamic PctMedianConfid NObs NEvent Median
                       LowerMedian UpperMedian;
               %SurvTabHeader(0)
               entry &r NObs;
               entry &r NEvent;
               entry &r eval(NObs-NEvent);
               entry &r eval(put(Median,&fmt));
               entry &r eval(put(LowerMedian,&fmt));
               entry &r eval(put(UpperMedian,&fmt));
            endlayout;
         else
            layout gridded / columns=7 border=TRUE;
               dynamic PctMedianConfid;
               %SurvTabHeader(1)
               %do i = 1 %to 10;
                  %let t = / textattrs=GraphData&i;
                  dynamic StrVal&i NObs&i NEvent&i Median&i
                          LowerMedian&i UpperMedian&i;
                  if (&i <= nstrata)
                     entry &r StrVal&i &t;
                     entry &r NObs&i &t;
                     entry &r NEvent&i &t;
                     entry &r eval(NObs&i-NEvent&i) &t;
                     entry &r eval(put(Median&i,&fmt)) &t;
                     entry &r eval(put(LowerMedian&i,&fmt)) &t;
                     entry &r eval(put(UpperMedian&i,&fmt)) &t;
```

```
                  endif;
               %end;
            endlayout;
         endif;
      endlayout;
   endcolumnheaders;
%mend;
```

The %SurvivalSummaryTable macro redefines the %AtRiskLatticeStart and %AtRiskLatticeEnd macros so that they provide the body of the event table:

```
%macro SurvivalSummaryTable;
   %macro AtRiskLatticeStart;
      layout lattice / columndatarange=union rowgutter=10
         rows=%if &outside %then 2 rowweights=ROWWEIGHTS;
              %else              1;;
      %if &outside %then %do; cell; %end;
   %mend;

   %macro AtRiskLatticeEnd(useclassopts);
      %if &outside %then %do;
         endcell;
         cell;
            layout overlay / walldisplay=none xaxisopts=(display=none);
               axistable x=TATRISK value=ATRISK / &atriskopts
                         %if &useclassopts ne %then &classopts;;
            endlayout;
         endcell;
      %end;
      %SurvivalTable
      endlayout;
   %mend;
%mend;
```

If you want to create an event table like the one displayed in Figure 26.36, you only need to call the %SurvivalSummaryTable macro. If you want to modify the table, then you need to modify the %SurvTabHeader and %SurvivalTable macros.

Dynamic Variables

Graph templates consist of instructions, written by SAS developers, in conjunction with SAS procedure code. However, SAS developers cannot fully provide some instructions when the template is written, because some elements of some graphs cannot be known until the procedure runs. For example, the legend title in a graph that has multiple strata corresponds to the label or name of the stratification variable, and the procedure calculates the *p*-value for the homogeneity test. SAS procedures create dynamic variables to provide some run-time information to graphs.[8] Some dynamic variables are set by the procedure and are declared in the template. Other dynamic variables are also set by the procedure, but you must declare them directly or through the template modification macros before you can use them.

Dynamic Variables That Are Automatically Declared

The primary dynamic variables are as follows:

ByFootNote	is a binary variable that, when true, displays the BY-group BY line as a footnote.
ByLine	is a character variable that provides the BY-group BY line.
ByTitle	is a binary variable that, when true, displays the BY-group BY line as a title.
ClassAtRisk	is a character variable that names the data object column that contains the classification (stratification) values.
GroupName	is a character variable that contains the stratification legend title.
LabelCL	is a character variable that contains the label for the confidence limits legend entry (including the percent sign).
LabelEP	is a character variable that contains the label for the equal-precision band legend entry (including the percent sign).
LabelHW	is a character variable that contains the label for the Hall-Wellner band legend entry (including the percent sign).
MaxTime	is a numeric variable that contains the maximum value to display on the X axis.
Method	is a character variable that contains the method for the plot title.
NStrata	is an integer variable that contains the number of strata.
PValue	is a numeric variable that contains the *p*-value for the homogeneity test.
PlotAtRisk	is a binary variable that, when true, is used to display the at-risk table.
PlotCensored	is a binary variable that, when true, displays the censored values on the step functions.
PlotCL	is a binary variable that, when true, displays the pointwise confidence limits.
PlotEP	is a binary variable that, when true, displays the equal-precision band.
PlotHW	is a binary variable that, when true, displays the Hall-Wellner confidence band.
PlotTest	is a binary variable that, when true, displays the *p*-value for the homogeneity test.

[8] Axis labels can be set directly in the template or at run time through dynamic variables or through data object column labels.

RowWeights	is a pair of relative heights of the plot and the external at-risk table.
SecondTitle	is a character variable that provides the second title line.
StratumID	is a character variable that provides the value of the stratification variable for the single stratum case.
TestName	is a character variable that provides the name of the homogeneity test (for example, 'logrank').
Transparency	is a numeric variable that provides the transparency for the confidence bands in the multiple strata case.
XName	is a character variable that contains a short label for the X axis, which might be used in place of the ordinary X-axis label when the ordinary label is long or the plot is small.
XtickValFitPol	is a character variable that contains the option for handling dense tick values on the X axis.
XtickVals	is a list of X-axis tick values.

Additional Dynamic Variables

PROC LIFETEST passes to the survival plots a number of dynamic variables that contain summary statistics. Some of these dynamic variables are used when you call the %SurvivalSummaryTable macro. Table 26.1 and Table 26.2 list these additional dynamic variables for the Kaplan-Meier curves and the life-table curves, respectively. These dynamic variables are not declared in the templates for the survival curves, but you can declare them and use them to enhance the default plots.[9] The names of the dynamic variables depend on the STRATA= suboption of the PLOTS=SURVIVAL option: STRATA=INDIVIDUAL produces a separate plot for each stratum, and STRATA=OVERALL produces one plot that has overlaid curves.

Table 26.1 Additional Dynamic Variables for `Stat.Graphics.ProductLimitSurvival`

STRATA=	Dynamic	Description
OVERLAY	StrVal*j*	Label for the *j*th stratum
	NObs*j*	Number of observations in the *j*th stratum
	NEvent*j*	Number of events in the *j*th stratum
	Median*j*	Median survival time of the *j*th stratum
	LowerMedian*j*	Lower median survival time of the *j*th stratum
	UpperMedian*j*	Upper median survival time of the *j*th stratum
	PctMedianConfid	Confidence of the median intervals, in percentage
INDIVIDUAL	NObs	Number of observations
	NEvent	Number of events
	Median	Median survival time
	LowerMedian	Lower median survival time
	UpperMedian	Upper median survival time
	PctMedianConfid	Confidence of the median intervals, in percentage

[9] Because the number of dynamic variables is a function of the number of strata, the template definition cannot automatically contain the correct number of dynamic variables.

Table 26.2 Additional Dynamic Variables for `Stat.Graphics.LifetableSurvival`

STRATA=	Dynamic	Description
OVERLAY	StrVal*j*	Label for the *j*th stratum
	NObs*j*	Number of observations in the *j*th stratum
	NEvent*j*	Number of events in the *j*th stratum
INDIVIDUAL	NObs	Number of observations
	NEvent	Number of events

Highly Customized Graphs: Unicode in the STRATA Statement Variable

This example shows you how to display Unicode characters in the survival plot legend. In this case, the legend displays the values of the STRATA statement variable. This example does not use the template customization macros. Instead, it takes a different approach to customization and shows you how to modify the survival plot data object and create the graph after PROC LIFETEST finishes. By using this technique, you can display in the survival plot Unicode characters that otherwise could not be displayed.

Unicode Values Require Alternative Processing

The following steps attempt to replace the ordinary hyphen with a longer dash (that is, they attempt to replace '-' with '—') and run PROC LIFETEST:

```
proc format;
   value $dfmt "AML-Low Risk"  = "AML(*ESC*){Unicode '2014'x}Low Risk"
               "AML-High Risk" = "AML(*ESC*){Unicode '2014'x}High Risk";
run;

ods graphics on;
proc lifetest data=sashelp.BMT plots=survival(cb=hw atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
   format group $dfmt.;
run;
```

The table and graph that are displayed in Output 26.40 show a difference between how ODS handles Unicode characters in tables and how ODS Graphics handles Unicode characters in graphs. When ODS creates tables and processes an escape sequence and a Unicode character in the value of a column, it substitutes the special character in the output. (This discussion does not apply to the LISTING destination.) When ODS Graphics creates graphs and processes an escape sequence and a Unicode character in the *formatted value* of a column, it substitutes the special character in the output. In other words, in ODS Graphics, you can use Unicode only with formats. This might seem puzzling, because this example does in fact use a format, but it does

not display the formatted value. It does not work as expected, because PROC LIFETEST does not send unformatted values and the format to ODS Graphics.

Figure 26.40 Unicode Displays Automatically in Tables but Not Graphs

The LIFETEST Procedure

Summary of the Number of Censored and Uncensored Values

Stratum	Group	Total	Failed	Censored	Percent Censored
1	ALL	38	24	14	36.84
2	AML—High Risk	45	34	11	24.44
3	AML—Low Risk	54	25	29	53.70
Total		137	83	54	39.42

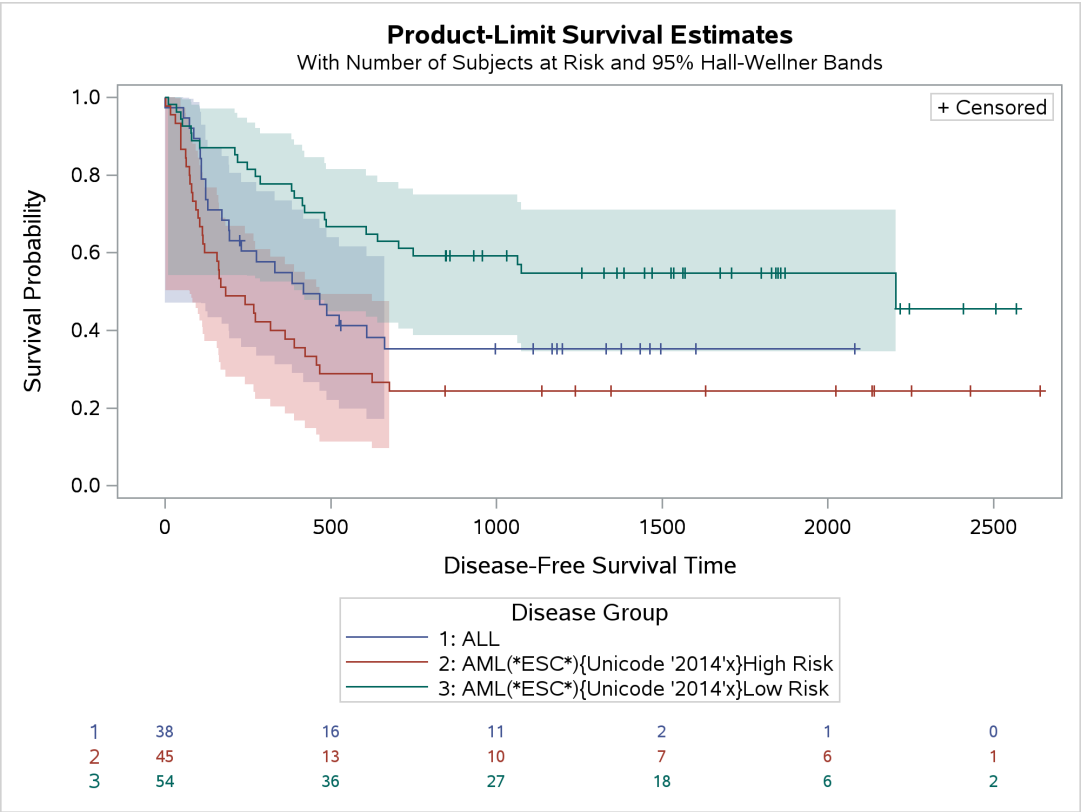

PROC LIFETEST and most other analytical procedures apply the formats at the time that the data are read and mostly process strings that contain the formatted values, not the raw values. If you want to see the Unicode characters in the graph, you must circumvent this behavior and send the raw values and the format to ODS Graphics. There are two ways to do this. The first uses the GROUP= option, and the second modifies the data object.

Using the GROUP= Option

You can display the GROUP= variable in the graph. The following step illustrates:

```
proc lifetest data=sashelp.BMT plots=survival(cb=hw atrisk(outside maxlen=13));
   time T * Status(0);
   strata / group=Group;
   format group $dfmt.;
run;
```

Some of the results are displayed in Output 26.41.

Figure 26.41 Select Output from Both PROC LIFETEST Runs

The LIFETEST Procedure

Summary of the Number of Censored and Uncensored Values

Stratum	Group	Total	Failed	Censored	Percent Censored
1	ALL	38	24	14	36.84
1	AML—High Risk	45	34	11	24.44
1	AML—Low Risk	54	25	29	53.70
Total		137	83	54	39.42

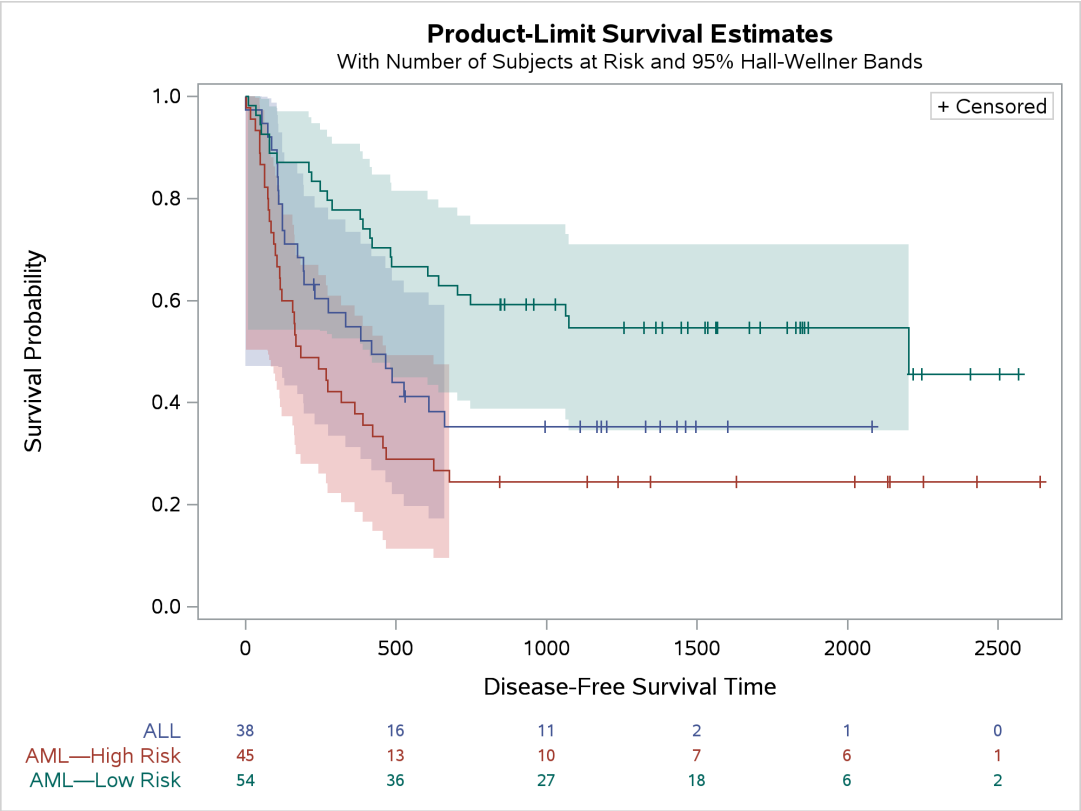

Modifying the Data Object

The second way to approach this problem requires you to modify the data object and use the ODS document. The first things you must do are to open an ODS document, save the results of the PROC LIFETEST analysis, save the data object that underlies the graph into a SAS data set, and close the ODS document. The following code illustrates:

```
ods document name=MyDoc (write);
proc lifetest data=sashelp.BMT plots=survival(cb=hw atrisk(outside maxlen=13));
   ods output survivalplot=sp;
   time T * Status(0);
   strata Group;
run;
ods document close;
```

The ODS document contains all the tables, graphs, titles, notes, and other components of the output. The following step lists the contents of the ODS document:

```
proc document name=MyDoc;
   list / levels=all;
quit;
```

The contents are displayed in Output 26.42.

Figure 26.42 Contents of the ODS Document

Listing of: \Work.Mydoc
Order by: Insertion
Number of levels: All

Obs	Path	Type
1	\Lifetest#1	Dir
2	\Lifetest#1\Stratum1#1	Dir
3	\Lifetest#1\Stratum1#1\ProductLimitEstimates#1	Table
4	\Lifetest#1\Stratum1#1\TimeSummary#1	Dir
5	\Lifetest#1\Stratum1#1\TimeSummary#1\SummaryNote#1	Note
6	\Lifetest#1\Stratum1#1\TimeSummary#1\Quartiles#1	Table
7	\Lifetest#1\Stratum1#1\TimeSummary#1\Means#1	Table
8	\Lifetest#1\Stratum2#1	Dir
9	\Lifetest#1\Stratum2#1\ProductLimitEstimates#1	Table
10	\Lifetest#1\Stratum2#1\TimeSummary#1	Dir
11	\Lifetest#1\Stratum2#1\TimeSummary#1\SummaryNote#1	Note
12	\Lifetest#1\Stratum2#1\TimeSummary#1\Quartiles#1	Table
13	\Lifetest#1\Stratum2#1\TimeSummary#1\Means#1	Table
14	\Lifetest#1\Stratum3#1	Dir
15	\Lifetest#1\Stratum3#1\ProductLimitEstimates#1	Table
16	\Lifetest#1\Stratum3#1\TimeSummary#1	Dir
17	\Lifetest#1\Stratum3#1\TimeSummary#1\SummaryNote#1	Note
18	\Lifetest#1\Stratum3#1\TimeSummary#1\Quartiles#1	Table
19	\Lifetest#1\Stratum3#1\TimeSummary#1\Means#1	Table
20	\Lifetest#1\CensoredSummary#1	Table
21	\Lifetest#1\StrataHomogeneity#1	Dir
22	\Lifetest#1\StrataHomogeneity#1\HomogeneityNote#1	Note
23	\Lifetest#1\StrataHomogeneity#1\HomStats#1	Table
24	\Lifetest#1\StrataHomogeneity#1\LogrankHomCov#1	Table
25	\Lifetest#1\StrataHomogeneity#1\WilcoxonHomCov#1	Table
26	\Lifetest#1\StrataHomogeneity#1\HomTests#1	Table
27	\Lifetest#1\SurvivalPlot#1	Graph

In the following step, you copy and paste the ODS document path for the survival plot into the OBDYNAM statement to create an ODS output data set that contains all the dynamic variables for the survival plot:

```
proc document name=MyDoc;
   ods output dynamics=dynamics;
   obdynam \Lifetest#1\SurvivalPlot#1;
quit;
```

The dynamic variables and their values are displayed in Output 26.43.[10]

[10]Some dynamic variables have numeric values, and some have character values. The ODS output data set (not shown) stores the character and formatted numeric values in the variable cValue1 and the unformatted numeric values in nValue1.

Figure 26.43 Dynamic Variables

Dynamics for: \Work.Mydoc\Lifetest#1\SurvivalPlot#1			
Name	**Value**	**Type**	**Namespace**
NSTRATA	3	Data	
PLOTHW	1	Data	
PLOTEP	0	Data	
PLOTCL	0	Data	
PLOTCENSORED	1	Data	
Transparency	0.7	Data	
PLOTTEST	0	Data	
BYTITLE		Data	
BYLINE		Data	
BYFOOTNOTE		Data	
METHOD	Product-Limit	Data	
ROWWEIGHTS	PREFERRED	Data	
XNAME	T	Data	
LABELHW	95% Hall-Wellner Band	Data	
SECONDTITLE	With Number of Subjects at Risk and 95% Hall-Wellner Bands	Data	
GROUPNAME	Disease Group	Data	
___NOBS___	156	Data	
___NOBS___	156	Column	HW_UCL
HW_UCL	HW_UCL	Column	HW_UCL
___NOBS___	156	Column	HW_LCL
HW_LCL	HW_LCL	Column	HW_LCL
___NOBS___	156	Column	Time
TIME	Time	Column	Time
___NOBS___	156	Column	Survival
SURVIVAL	Survival	Column	Survival
___NOBS___	156	Column	AtRisk
ATRISK	AtRisk	Column	AtRisk
___NOBS___	156	Column	Event
EVENT	Event	Column	Event
___NOBS___	156	Column	Censored
CENSORED	Censored	Column	Censored
___NOBS___	156	Column	tAtRisk
TATRISK	tAtRisk	Column	tAtRisk
___NOBS___	156	Column	Stratum
STRATUM	Stratum	Column	Stratum
CLASSATRISK	Stratum	Column	Stratum
___NOBS___	156	Column	StratumNum
STRATUMNUM	StratumNum	Column	StratumNum

You can use a DATA step as follows to generate a PROC SGRENDER statement; specify the ODS output data set, the template, and the format; and specify the names and values of all the dynamic variables:

```
data _null_;
   set dynamics(where=(label1 ne '___NOBS___')) end=eof;
   if _n_ = 1 then
      call execute('proc sgrender data=sp
```

```
                    template=Stat.Lifetest.Graphics.ProductLimitSurvival2;
                    format stratum $dfmt.;
                    dynamic');
   if cvalue1 ne ' ' then
      call execute(catx(' ', label1, '=',
                   ifc(n(nvalue1), cvalue1, quote(trim(cvalue1)))));
   if eof then call execute('; run;');
run;
```

This DATA step generates and executes a PROC SGRENDER step that creates the graph. The DATA step uses a series of CALL EXECUTE statements to populate the list of dynamic variables. The DATA step generates and submits to SAS the following code (which has been reformatted for display here):

```
proc sgrender data=sp
   template=Stat.Lifetest.Graphics.ProductLimitSurvival2;
   format stratum $dfmt.;
   dynamic
      NSTRATA = 3
      PLOTHW = 1
      PLOTEP = 0
      PLOTCL = 0
      PLOTCENSORED = 1
      Transparency = 0.7
      PLOTTEST = 0
      METHOD = "Product-Limit"
      ROWWEIGHTS = "PREFERRED"
      XNAME = "T"
      LABELHW = "95% Hall-Wellner Band"
      SECONDTITLE = "With Number of Subjects at Risk and 95% Hall-Wellner Bands"
      GROUPNAME = "Disease Group"
      CLASSATRISK = "Stratum";
run;
```

The survival plot, which now contains the long dashes, is displayed in Output 26.44.

Figure 26.44 Unicode in the Legend

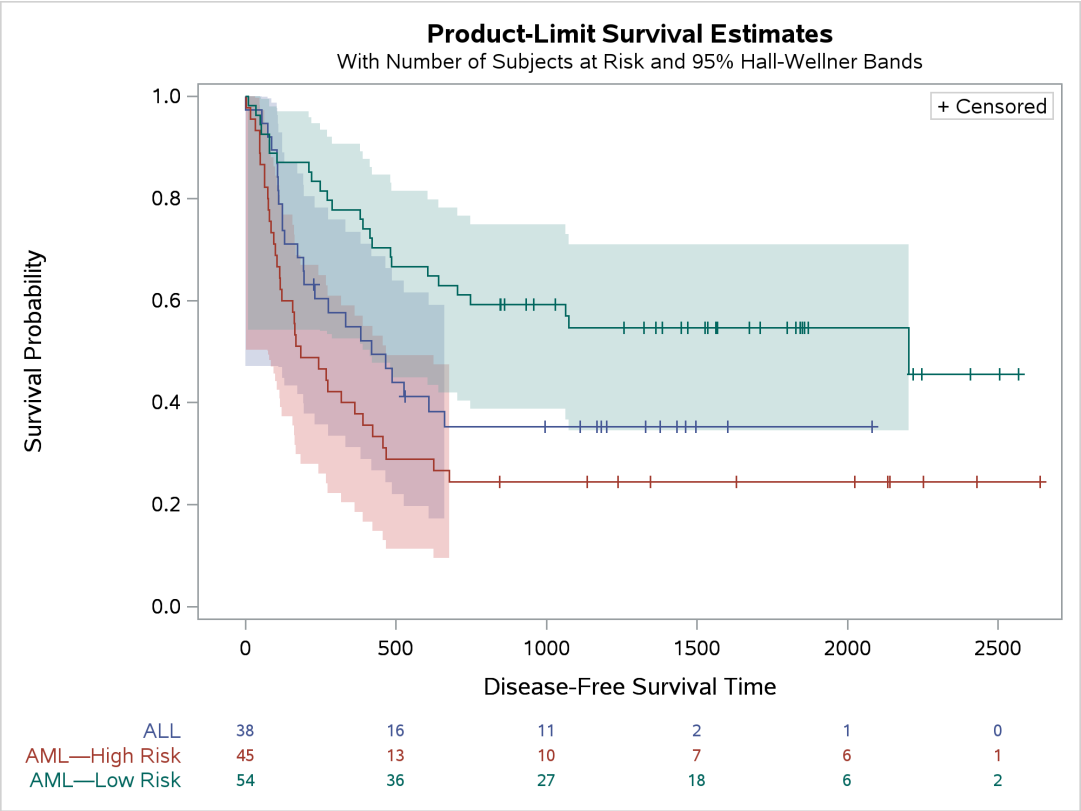

This example shows that you can save the data object and dynamic variables into SAS data sets. Then you can apply a format to the Stratum variable as you create the graph from those data sets. The advantage of this approach is that it provides full flexibility. For example, you can use it with multiple strata variables.

Multiple Strata Variables and Reformatting the Legend

This example reformats the Group variable from the Sashelp.BMT data set as follows to produce the same analysis from two strata variables:

```
data bmt;
   set sashelp.bmt;
   s1 = scan(group, 1, '-');
   s2 = scan(group, 2, '-');
   if s2 ne ' ' then s1 = cats(s1,'-');
   else              s2 = 'ALL';
run;

proc lifetest data=BMT plots=survival(cb=hw atrisk(outside maxlen=13));
   ods output survivalplot=sp;
   time T * Status(0);
   strata s1 s2;
run;
```

In practice, you would never create two strata variables from one. This is done here simply to illustrate the techniques for dealing with multiple strata variables in the context of a familiar example.

The graph in Output 26.45 contains variable names and equal signs. You can reformat the values in any way that you choose. For example, if there were Unicode specifications in each variable, you could handle it this way.

Figure 26.45 Unicode in the Legend

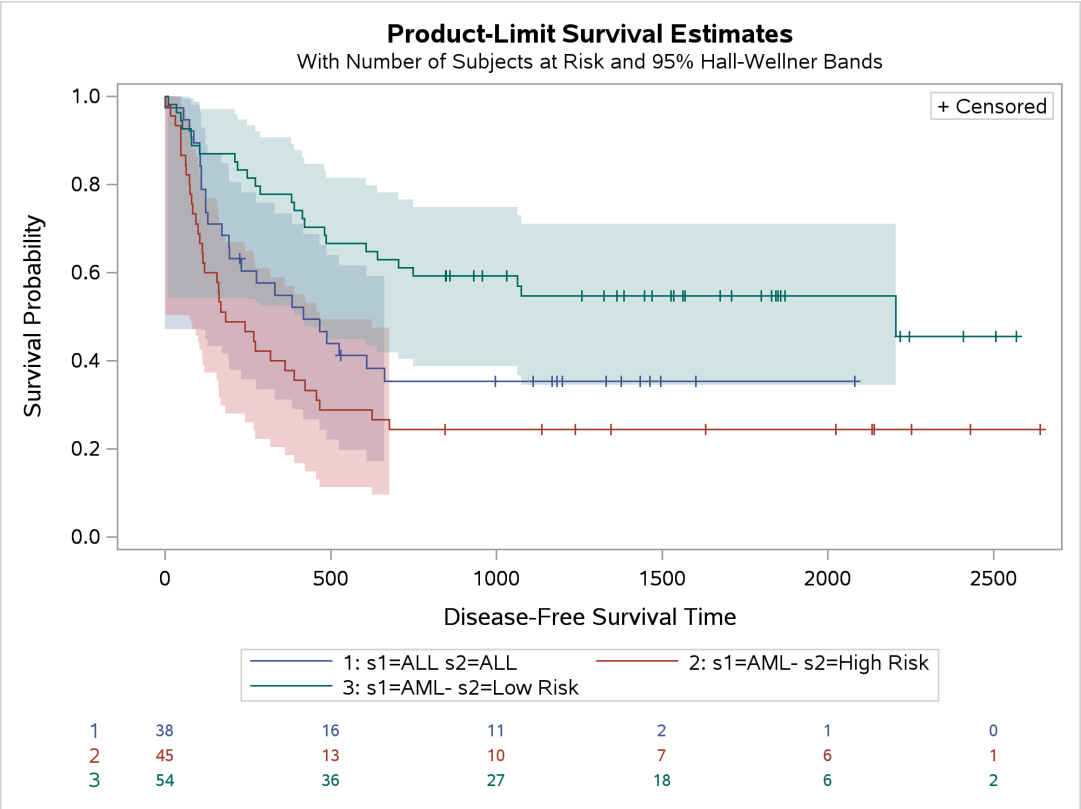

The next step creates a new format that contains two Unicode specifications:

```
proc format;
   value $d2fmt '1: s1=ALL s2=ALL'= 'ALL'
                "2: s1=AML- s2=High Risk" =
                "AML (*ESC*){Unicode '2014'x} High(*ESC*){Unicode '2012'x}Risk"
                "3: s1=AML- s2=Low Risk"  =
                "AML (*ESC*){Unicode '2014'x} Low(*ESC*){Unicode '2012'x}Risk";
run;
```

The rest of this example proceeds as the previous example did. You begin by capturing the output data object and dynamic variables. Then you create the graph by using the new format. The only difference is that you need to change the value of the dynamic variable ClassAtRisk so that the formatted value is displayed to the left of the subjects-at-risk table in place of the stratum number.

```
ods document name=MyDoc (write);
proc lifetest data=BMT plots=survival(cb=hw atrisk(outside maxlen=16));
   ods output survivalplot=sp;
   time T * Status(0);
   strata s1 s2;
run;
ods document close;

proc document name=MyDoc;
   ods output dynamics=dynamics;
   obdynam \Lifetest#1\SurvivalPlot#1;
quit;

data _null_;
   set dynamics(where=(label1 ne '___NOBS___')) end=eof;
   if _n_ = 1 then
      call execute('proc sgrender data=sp
                    template=Stat.Lifetest.Graphics.ProductLimitSurvival2;
                    format stratum $d2fmt.;
                    dynamic');
   if label1 eq 'CLASSATRISK' then cvalue1 = 'Stratum';
   if cvalue1 ne ' ' then
      call execute(catx(' ', label1, '=',
                   ifc(n(nvalue1), cvalue1, quote(trim(cvalue1)))));
   if eof then call execute('; run;');
run;
```

The graph is displayed in Output 26.46.

Figure 26.46 Reformatted Legend

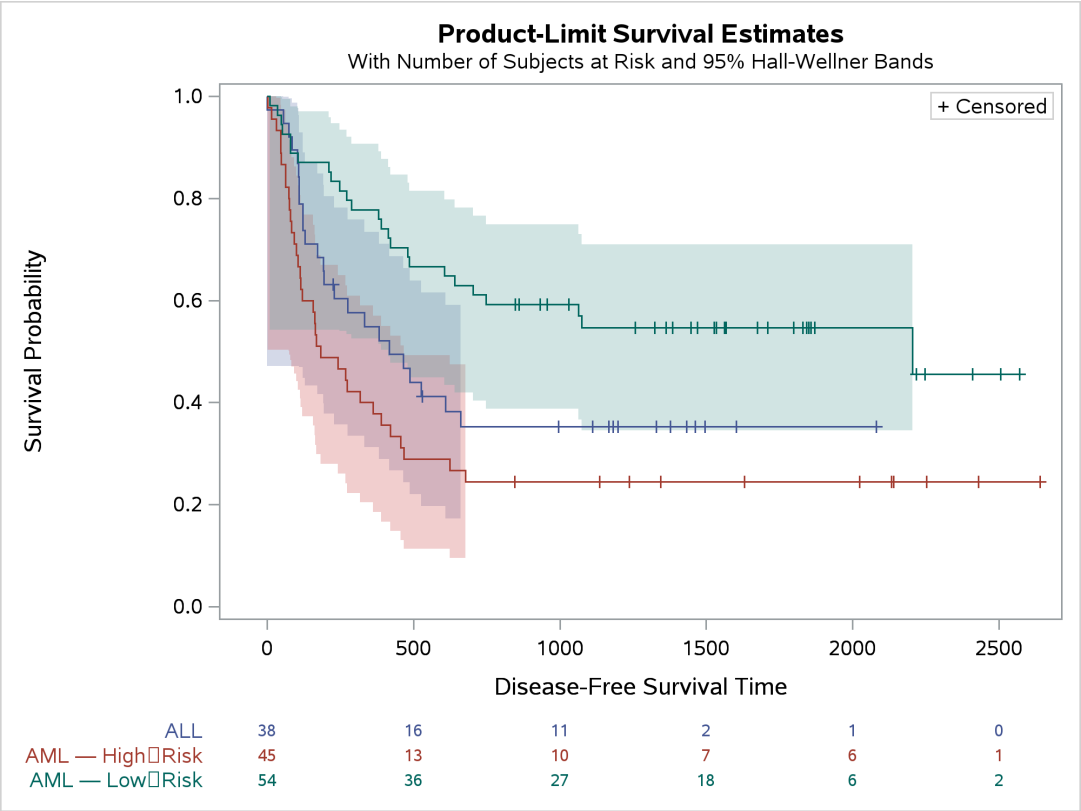

You can extend this example by specifying additional procedure options or by modifying the template. You can even add SG annotation.

Style Templates

Graphs that are produced by ODS Graphics are controlled by the data object (the matrix of information that is graphed), the graph template (the program that controls how a specific graph is constructed), and a style template (a program that controls the overall appearance of graphs, including colors, line and marker styles, sizes, fonts, and so on). Although it is rarely necessary, you can use different styles or modify styles to change the appearance of all graphs, including the survival plot. In the past, you could make certain Kaplan-Meier plot modifications only through style modifications. However, with the addition of the DATACOLORS=, DATACONTRASTCOLORS=, and DATALINEPATTERNS= options in the GTL, you no longer have to modify styles in order to modify how groups of observations are displayed. This section shows you how to change styles, extract group color and other information from styles, and modify styles.

Changing the Style

The graphs that have been displayed up to this point were all produced by using the HTMLBlue style, which is the default style for the HTML destination in the SAS windowing environment. This is an all-color style (because of the ATTRPRIORITY='Color' option); it does not rely on line style or marker changes to differentiate groups. You can switch to a style that varies colors, markers, and lines by specifying the STYLE= option in an ODS destination statement. You can use the HTMLBlueCML style as follows to make a graph whose line patterns differ:

```
ods html style=htmlbluecml image_dpi=300;
proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
ods html close;
```

The results are displayed in Figure 26.47. This example also illustrates specifying the IMAGE_DPI= option to control the resolution (measured in dots per inch, or DPI) of the image. All images in this chapter are created at 300 DPI. The default setting for the HTML destination is 100 DPI. Images that are created at 300 DPI are clearer than images created at 100 DPI, but they require about nine times as much disk space.

Figure 26.47 Line and Color Group Differentiation

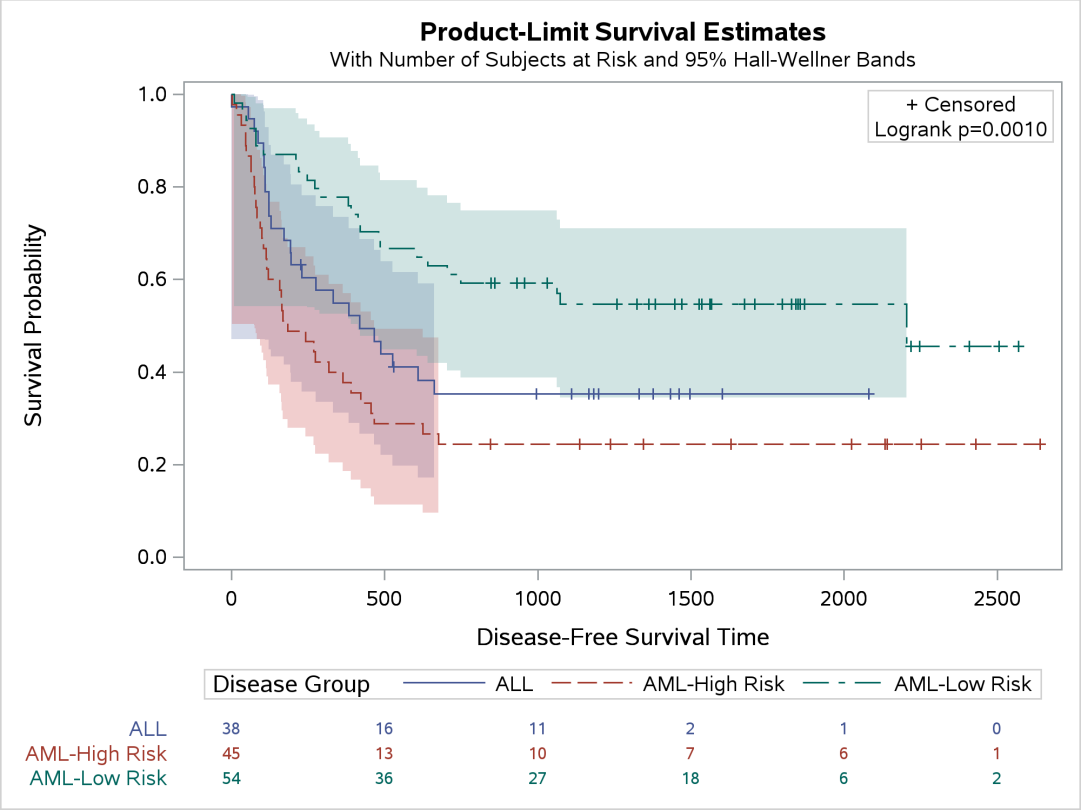

Color Priority Styles

You can use the HTMLBlue or Pearl style when you want to distinguish groups only by color. Alternatively, you can easily modify any other style to be an all-color style like HTMLBlue or Pearl by using the ATTRPRIORITY='Color' option:[11]

```
proc template;
   define style styles.ListingColor;
      parent = styles.Listing;
      style Graph from Graph / attrpriority = "Color";
   end;
run;
```

You need to specify the new style name in an ODS destination statement, as in the following:

```
ods html style=ListingColor image_dpi=300;
proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
ods html close;
```

The results are displayed in Figure 26.48.

Figure 26.48 ATTRPRIORITY='Color' Style

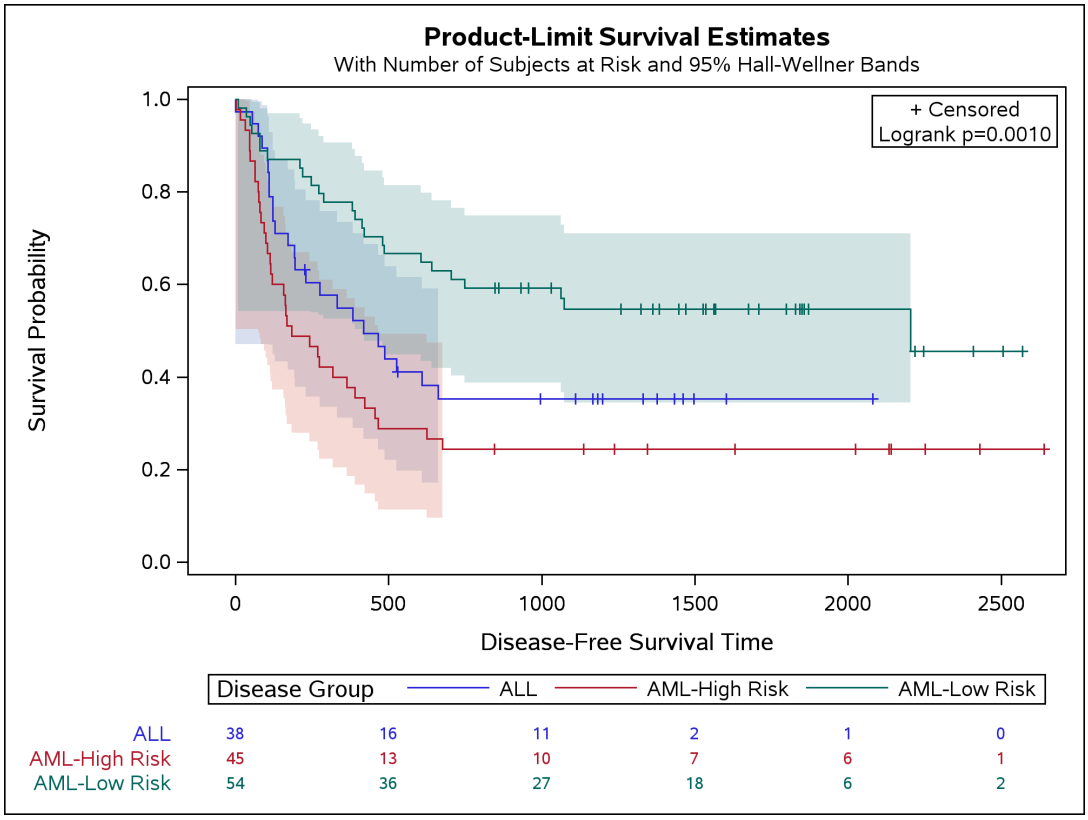

[11] The style option is ATTRPRIORITY=*quoted-string*, whereas the GTL option is ATTRPRIORITY=*keyword*.

Displaying a Style and Extracting Color Lists

You can use PROC TEMPLATE with the SOURCE statement to display a style as follows:

```
proc template;
   source styles.htmlblue;
run;
```

The results of this step, which are not shown, include the option PARENT=STYLES.STATISTICAL and do not include definitions of the colors (**gData1**, **gData2**, . . . , **gData12**) and contrast colors (**gcData1**, **gcData2**, . . . , **gcData12**). These are the color definitions that are used in the style elements **GraphData1**, **GraphData2**, . . . , **GraphData12**. You can examine the parent Statistical style as follows:

```
proc template;
   source styles.statistical;
run;
```

The results of this step are not shown because they are hard to interpret in their raw form, but the desired color definitions are included. You can submit the following statements to display the colors for the Statistical (and hence HTMLBlue) style in a more understandable form:

```
proc template;
   source styles.statistical / file='style.tmp';
run;

data colors;
   length element Color $ 20;
   infile 'style.tmp';
   input;
   if index(_infile_, 'data') then do;
      element = scan(_infile_, 1, ' ');
      Color = scan(_infile_, 3, ' ;');
      Type = ifc(index(element, 'gc'), 'Line', 'Fill') || ' Colors';
      i = input(compress(element, 'gcdat'';'), ?? 2.);
      if i then output;
   end;
run;

proc sort; by descending type i; run;

proc print; id type; by descending type; var color; run;
```

The results are displayed in Figure 26.49. All colors are specified in values of the form CX*rrggbb*, where the last six characters specify RGB (red, green, blue) values on the hexadecimal scale of 00 to FF (0 to 255, base 10).

Figure 26.49 HMTLBlue Style Colors List

Type=Line Colors

Type	Color
Line Colors	cx445694
	cxA23A2E
	cx01665E
	cx543005
	cx9D3CDB
	cx7F8E1F
	cx2597FA
	cxB26084
	cxD17800
	cx47A82A
	cxB38EF3
	cxF9DA04

Type	Color
Fill Colors	cx6F7EB3
	cxD05B5B
	cx66A5A0
	cxA9865B
	cxB689CD
	cxBABC5C
	cx94BDE1
	cxCD7BA1
	cxCF974B
	cx87C873
	cxB7AEF1
	cxDDD17E

You can use the following steps to display the **GraphData1** – **GraphData12** line and fill colors (contrast colors and colors, respectively):

```
data display;
   array y[12] y1 - y12;
   do i = 1 to 12; y[i] = i;      end;
   do x = 1 to 10; output;        end;
   do i = 1 to 12; y[i] = i + .5; end;
   do x = 1 to 10; output;        end;
run;

data _null_;
   set colors;
   call symputx(compress(type || put(i, 2.)), color);
run;

proc sgplot noautolegend data=display;
   %macro reg;
      title 'Line and Fill Colors, Respectively';
      %do i = 1 %to 12;
```

```
        reg y=y%eval(13-&i) x=x / lineattrs=GraphData&i clmattrs=GraphData&i
                    nomarkers clm  curvelabelpos=max
                    curvelabel="  GraphData&i &&LineColors&i &&FillColors&i";
      %end;
   %mend;
   %reg
   xaxis display=none;
   yaxis display=none;
run;

title;
```

The results are displayed in Figure 26.50. The colors in Figure 26.50 are richer than the colors in the bands in the survival plots because of the DATATRANSPARENCY= options in the BANDPLOT statements.

Figure 26.50 HTMLBlue Style Colors Display

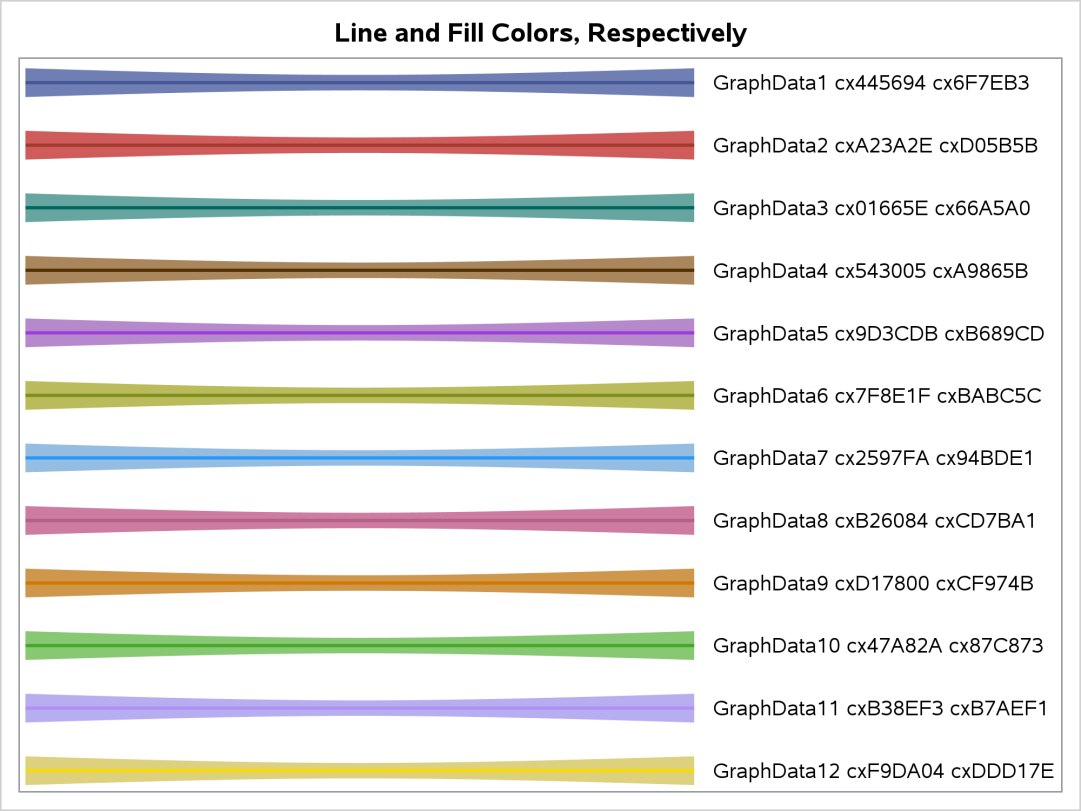

Modifying Color Lists

You can use the information in Figure 26.50 to specify the desired colors in the graph template. You can copy the third, second, and first colors from each list and switch the colors as follows:

```
%ProvideSurvivalMacros

%let GraphOpts = DataContrastColors=(cx01665E cxA23A2E cx445694)
                 DataColors=(cx66A5A0 cxD05B5B cx6F7EB3);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 26.51. The familiar colors are used, but they are now in a different order. The next section shows you how to modify a style template to change the color order without having to extract the original color names.

Figure 26.51 Swapping Colors from the Style

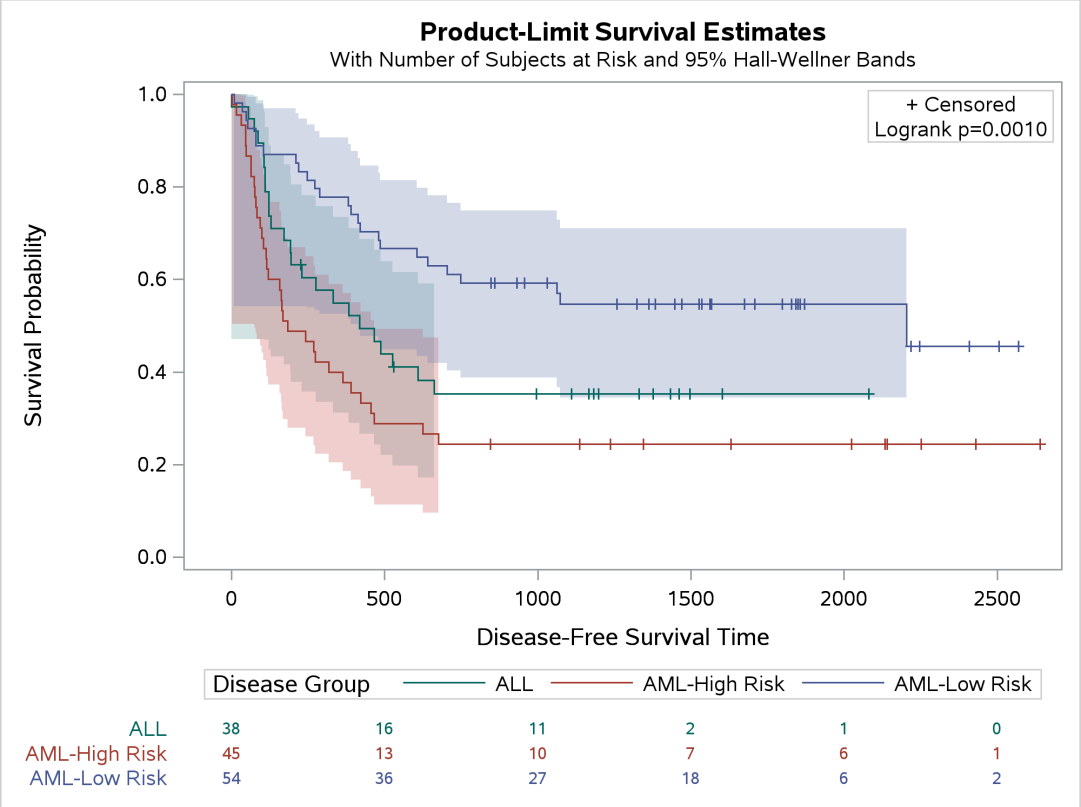

You can use the information in Figure 26.50 to modify the style template, but the next example shows an easier way.

Swapping Colors among Style Elements

You can modify the colors in a style as follows:

```
%macro reorder(from, to, list);

   proc template;
      define style styles.&to;
         parent=styles.&from;
         %do i = 1 %to 12;
            %let s = %scan(&list, &i);
            %if &s ne %then %do;
               style GraphData&i from GraphData&i /
                     contrastcolor = GraphColors("gcdata&s")
                     color = GraphColors("gdata&s");
            %end;
         %end;
      end;
   run;

%mend;

%reorder(htmlblue,   /* Parent style.                              */
         MyStyle,    /* New style to create.  Specify it in an ODS */
                     /* destination statement.                     */
         3 2 1)      /* Replace the first few GraphData colors     */
                     /* (1 2 3) with the colors from the specified */
                     /* GraphData style elements (3 2 1).          */
                     /* You can specify up to 12 integers in the   */
                     /* range 1 - 12.                              */
```

The %Reorder macro creates a new style called MyStyle that inherits most of its attributes from the HTMLBlue style. However, in the new style, the colors for groups 1, 2, and 3 have been replaced by the colors for groups 3, 2, and 1. In other words, the colors for **GraphData1** and **GraphData3** have been switched.

The following step creates the plot:

```
ods html style=mystyle;
proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
ods html close;
```

The survival plot is not shown, but it matches the plot in Figure 26.51.

The rest of this section is optional. It explains how you can directly modify colors in a style template when the %Reorder macro or the technique illustrated in the section "Changing the Group Color" on page 964 is not sufficient.

The source code for the MyStyle style (as generated by the %Reorder macro) is as follows:

```
proc template;
   define style Styles.MyStyle;
      parent = styles.htmlblue;
      style GraphData1 from GraphData1 /
         color = GraphColors('gdata3')
         contrastcolor = GraphColors('gcdata3');
      style GraphData2 from GraphData2 /
         color = GraphColors('gdata2')
         contrastcolor = GraphColors('gcdata2');
      style GraphData3 from GraphData3 /
         color = GraphColors('gdata1')
         contrastcolor = GraphColors('gcdata1');
   end;
run;
```

You can create a modified style that has direct color specifications by using the colors in Figure 26.49 as follows:

```
proc template;
   define style Styles.MyStyle;
      parent = styles.htmlblue;
      style GraphData1 from GraphData1 /
         color = cx66A5A0
         contrastcolor = cx01665E;
      style GraphData2 from GraphData2 /
         color = cxD05B5B
         contrastcolor = cxA23A2E;
      style GraphData3 from GraphData3 /
         color = cx6F7EB3
         contrastcolor = cx445694;
   end;
run;
```

You can define additional GraphData*N* style elements as well. For more information about how to define style elements, see the section "Displaying Other Style Elements" on page 1023.

You can delete the new style template as follows:

```
proc template;
   delete Styles.MyStyle / store=sasuser.templat;
run;
```

Displaying a Style and Extracting Font Information

You can use PROC TEMPLATE with the SOURCE statement to display a style and its parent styles in the SAS log:

```
proc template;
   source styles.htmlblue / expand;
run;
```

The results of this step are long and are not shown. You can write a copy of the style templates to a file as follows:

```
proc template;
   source styles.htmlblue / expand file='style.tmp';
run;
```

The EXPAND option writes the specified style (HTMLBlue), followed by its parent (Statistical), and followed by the parent's parent (DEFAULT) to the file. The following step extracts and displays the first place in the file that the graph fonts are defined (which make the final decision in the style template):

```
data _null_;
   infile 'style.tmp' pad;
   input line $char80.;
   file print;
   if index(lowcase(line), ' graphfonts ') then y + 1;
   if y then put line $char80.;
   if y and index(line, ';') then stop;
run;
```

The results are displayed in Figure 26.52.

Figure 26.52 Graph Font Definition

```
class GraphFonts /
   'NodeDetailFont' = ("<sans-serif>, <MTsans-serif>",7pt)
   'NodeInputLabelFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'NodeLabelFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'NodeTitleFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'GraphDataFont' = ("<sans-serif>, <MTsans-serif>",7pt)
   'GraphUnicodeFont' = ("<MTsans-serif-unicode>",9pt)
   'GraphValueFont' = ("<sans-serif>, <MTsans-serif>",9pt)
   'GraphLabel2Font' = ("<sans-serif>, <MTsans-serif>",10pt)
   'GraphLabelFont' = ("<sans-serif>, <MTsans-serif>",10pt)
   'GraphFootnoteFont' = ("<sans-serif>, <MTsans-serif>",10pt)
   'GraphTitleFont' = ("<sans-serif>, <MTsans-serif>",11pt,bold)
   'GraphTitle1Font' = ("<sans-serif>, <MTsans-serif>",14pt,bold)
   'GraphAnnoFont' = ("<sans-serif>, <MTsans-serif>",10pt);
```

If the **GraphFonts** style element is defined in the HTMLBlue style, then it will appear first in the file, followed by the definitions from the Statistical style and then the DEFAULT style. In this case, the **GraphFonts** style element is defined in the DEFAULT style (last in the file), which is overridden by a definition in the Statistical style (closer to the top of the file); that is the definition that is inherited by the HTMLBlue style and displayed in Figure 26.52.

The following step creates a new style, BigFont, that changes the **GraphLabelFont** style element from a regular 10-point font to a bold 12-point font and changes the **GraphValueFont** style element from a regular 9-point font to a bold 8-point font:

```
proc template;
   define style Styles.BigFont;
      parent = Styles.HTMLBlue;
      style graphfonts from graphfonts /
         'GraphLabelFont' = ("<sans-serif>, <MTsans-serif>",12pt,bold)
         'GraphValueFont' = ("<sans-serif>, <MTsans-serif>",8pt,bold);
   end;
run;
```

The following step creates the plot that is displayed in Output 26.53:

```
ods html style=BigFont;
proc lifetest data=sashelp.BMT plots=survival(maxlen=13 atrisk);
   time T * Status(0);
   strata Group;
run;
ods html close;
```

Figure 26.53 Font Modifications

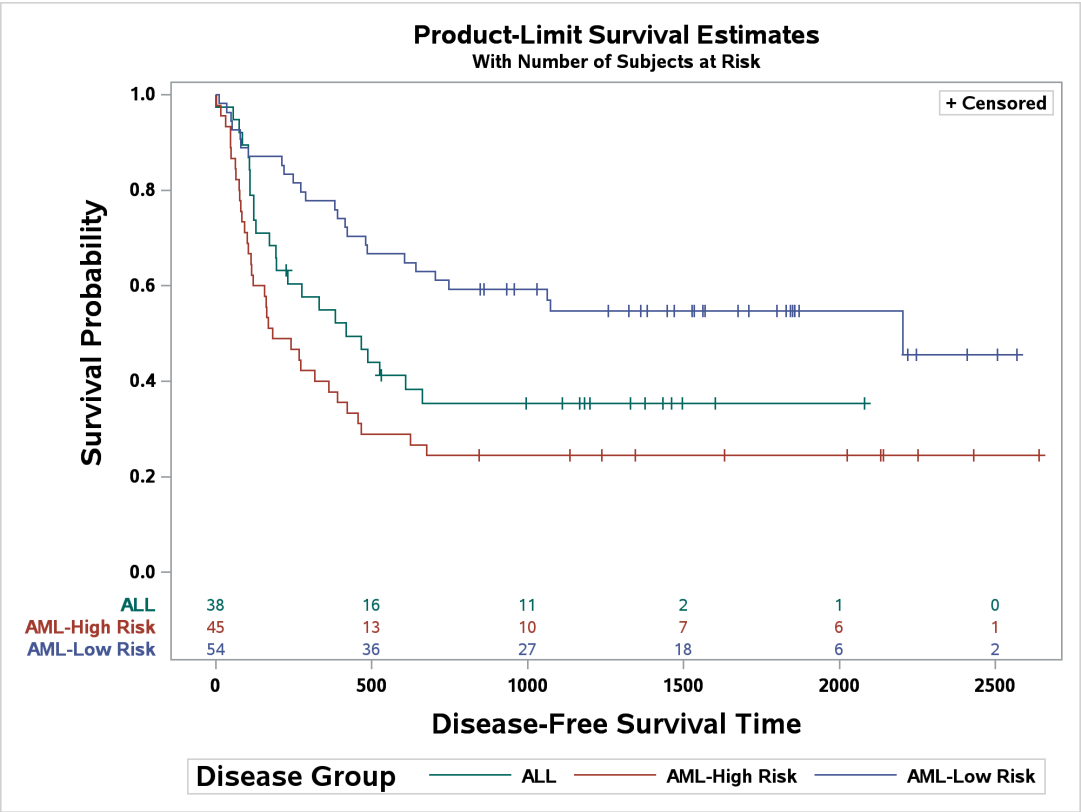

You can delete the new style template as follows:

```
proc template;
   delete Styles.BigFont / store=sasuser.templat;
run;
```

For information about making ad hoc font changes in the graph templates rather than making more global font changes in style templates, see the section "Changing the Font" on page 967.

Displaying Other Style Elements

You can use the approach from the previous example to display other style elements:

```
proc template;
   source styles.htmlblue / expand file='style.tmp';
run;

data _null_;
   infile 'style.tmp' pad;
   input line $char80.;
   file print;
   if index(lowcase(line), ' graphdata1 ') then y + 1;
   if y then put line $char80.;
   if y and index(line, ';') then stop;
run;
```

This example displays the **GraphData1** style element. The results are displayed in Figure 26.54.

Figure 26.54 GraphData1 Style Element

```
class GraphData1 /
   fillpattern = "L1"
   markersymbol = "circle"
   linestyle = 1
   contrastcolor = GraphColors('gcdata1')
   color = GraphColors('gdata1');
```

The following steps display all the **GraphFonts** style elements from all the styles:

```
proc template;
   source styles / file='style.tmp';
run;

data _null_;
   infile 'style.tmp' pad;
   length style $ 80;
   retain style;
   input line $char80.;
   file print;
   if index(lowcase(line), 'define style') then style = line;
   if index(lowcase(line), ' graphfonts ') then do;
      y + 1;
      put style $char80.;
   end;
```

```
    if y then put line $char80.;
    if index(line, ';') then y = 0;;
  run;
```

The results of this step are not displayed. You can use this approach to help you better understand the options that are available for modifying styles. The SOURCE statement specifies a single-level value of STYLES rather than a specific style name such as STYLES.HTMLBLUE, so all templates that begin with STYLES as the first level (all style templates) are written to the file. The DATA step displays all definitions of **GraphFonts** and the names of all styles that define the **GraphFonts** style element.

You can insert the name of another style element (in lowercase with a leading and trailing blank) in the preceding programs in place of 'graphdata1' or 'graphfonts'. After you display a style element, you can modify the definition and create a new style that uses the modified definition, as in the example in the section "Displaying a Style and Extracting Font Information" on page 1021. Some of the style elements that you might want to display and modify are listed in Table 26.3.

Table 26.3 Style Elements

AfterCaption	GraphAnnoText	GraphHeaderBackground	List3
Batch	GraphAxisLines	GraphHistogram	ListItem
Body	GraphBackground	GraphInitial	Note
BodyDate	GraphBand	GraphLabel2Text	NoteBanner
ByContentFolder	GraphBar	GraphLabelText	NoteContentFixed
Byline	GraphBlock	GraphLegendBackground	Output
BylineContainer	GraphBlockHeader	GraphMissing	PageNo
Caption	GraphBorderLines	GraphOther	Pages
Color_list	GraphBox	GraphOutlier	PagesProcLabel
Colors	GraphBoxMean	GraphOutlines	PagesTitle
Container	GraphBoxMedian	GraphOverflow	Paragraph
ContentFolder	GraphBoxWhisker	GraphPhaseBox	Parskip
ContentProcLabel	GraphClipping	GraphPrediction	PrePage
ContentTitle	GraphColors	GraphPredictionLimits	ProcTitle
Contents	GraphConfidence	GraphReference	ProcTitleFixed
Continued	GraphConfidence2	GraphRunTest	RowFooter
Data	GraphConnectLine	GraphSelection	RowFooterEmphasis
DataEmphasis	GraphContour	GraphStars	RowFooterEmphasisFixed
DataEmphasisFixed	GraphControlLimits	GraphTitle1Text	RowFooterFixed
DataFixed	GraphData1	GraphTitleText	RowFooterStrong
DataStrong	GraphData2	GraphUnderflow	RowFooterStrongFixed
DataStrongFixed	GraphData3	GraphUnicodeText	RowHeader
Date	GraphData4	GraphValueText	RowHeaderEmphasis
Document	GraphData5	GraphWalls	RowHeaderEmphasisFixed
DropShadowStyle	GraphData6	GraphZoneA	RowHeaderFixed
ErrorBanner	GraphData7	GraphZoneB	RowHeaderStrong
ErrorContentFixed	GraphData8	GraphZoneC	RowHeaderStrongFixed
ExtendedPage	GraphData9	Header	SysTitleAndFooterContainer
FatalBanner	GraphData10	HeaderEmphasis	SystemFooter
FatalContentFixed	GraphData11	HeaderEmphasisFixed	SystemTitle
Fonts	GraphData12	HeaderFixed	Table

Footer	GraphDataDefault	HeaderStrong	ThreeColorAltRamp
FooterEmphasis	GraphDataText	HeaderStrongFixed	ThreeColorRamp
FooterEmphasisFixed	GraphEllipse	HeadersAndFooters	TitleAndNoteContainer
FooterFixed	GraphError	Index	TitlesAndFooters
FooterStrong	GraphFinal	IndexItem	TwoColorAltRamp
FooterStrongFixed	GraphFit	IndexProcName	TwoColorRamp
Frame	GraphFit2	IndexTitle	UserText
Graph	GraphFloor	LayoutContainer	WarnBanner
GraphAltBlock	GraphFonts	LineContent	WarnContentFixed
GraphAnnoLine	GraphFootnoteText	List	
GraphAnnoShape	GraphGridLines	List2	

SAS Item Stores

In other sections of this chapter, you submit PROC TEMPLATE statements (either directly or through macros) to compile and save graph and style templates. Compiled templates are stored in special SAS files called item stores. Assuming that you have not modified your ODS path with an ODS PATH statement, the templates that you compile are stored in an item store in the Sasuser library. By default, all templates that SAS provides are stored in an item store in the Sashelp library. By default, the Sashelp item store has read access only; you cannot write to it. By default, the Sasuser item store has update access; you can both read and write to it. **CAUTION:** Never set the Sashelp item store to update or write access. If you do, and if you have administrator privileges on your computer, you could corrupt the Sashelp item store.

Assuming that the default ODS path is used, ODS first looks for a template in the Sasuser item store. If it does not find the template there, ODS next looks for the template in the Sashelp item store. Files in the Sasuser library persist across SAS sessions until you delete them. You can run the following step to delete the entire Sasuser item store (including all compiled graph and style templates that you added or modified) so that ODS uses only the templates the SAS System supplies:

```
ods path sashelp.tmplmst(read);
proc datasets library=sasuser nolist;
   delete templat(memtype=itemstor);
run;
ods path reset;
```

For more information about the ODS path and SAS item stores, see Chapter 24, "Statistical Graphics Using ODS."

References

Hall, W. J., and Wellner, J. A. (1980). "Confidence Bands for a Survival Curve from Censored Data." *Biometrika* 67:133–143.

Kaplan, E. L., and Meier, P. (1958). "Nonparametric Estimation from Incomplete Observations." *Journal of the American Statistical Association* 53:457–481.

Klein, J. P., and Moeschberger, M. L. (1997). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: Springer-Verlag.

Kuhfeld, W. F. (2015). *Advanced ODS Graphics Examples*. Cary, NC: SAS Institute Inc. http://support.sas.com/documentation/prod-p/grstat/9.4/en/PDF/odsadvg.pdf.

Chapter 27
The ACECLUS Procedure

Contents

Overview: ACECLUS Procedure **1028**
Background 1028
Getting Started: ACECLUS Procedure **1034**
Syntax: ACECLUS Procedure **1040**
PROC ACECLUS Statement 1040
BY Statement 1045
FREQ Statement 1046
VAR Statement 1046
WEIGHT Statement 1046
Details: ACECLUS Procedure **1047**
Missing Values 1047
Output Data Sets 1047
Computational Resources 1048
Displayed Output 1049
ODS Table Names 1050
ODS Graphics 1050
Example: ACECLUS Procedure **1051**
Example 27.1: Transformation and Cluster Analysis of Fisher Iris Data 1051
References **1055**

Overview: ACECLUS Procedure

The ACECLUS (approximate covariance estimation for clustering) procedure obtains approximate estimates of the pooled within-cluster covariance matrix when the clusters are assumed to be multivariate normal with equal covariance matrices. Neither cluster membership nor the number of clusters needs to be known. PROC ACECLUS is useful for preprocessing data to be subsequently clustered by the CLUSTER or FASTCLUS procedure.

Many clustering methods perform well with spherical clusters but poorly with elongated elliptical clusters (Everitt 1980, pp. 77–97). If the elliptical clusters have roughly the same orientation and eccentricity, you can apply a linear transformation to the data to yield a spherical within-cluster covariance matrix—that is, a covariance matrix proportional to the identity. Equivalently, the distance between observations can be measured in the metric of the inverse of the pooled within-cluster covariance matrix. The remedy is difficult to apply, however, because you need to know what the clusters are in order to compute the sample within-cluster covariance matrix. One approach is to estimate iteratively both cluster membership and within-cluster covariance (Wolfe 1970; Hartigan 1975). Another approach is provided by Art, Gnanadesikan, and Kettenring (1982). They have devised an ingenious method for estimating the within-cluster covariance matrix without knowledge of the clusters. The method can be applied before any of the usual clustering techniques, including hierarchical clustering methods.

First, Art, Gnanadesikan, and Kettenring (1982) obtain a decomposition of the total-sample sum-of-squares-and-crossproducts (SSCP) matrix into within-cluster and between-cluster SSCP matrices computed from pairwise differences between observations, rather than differences between observations and means. Then, they show how the within-cluster SSCP matrix based on pairwise differences can be approximated without knowing the number or the membership of the clusters. The approximate within-cluster SSCP matrix can be used to compute distances for cluster analysis, or it can be used in a canonical analysis similar to canonical discriminant analysis. For more information, see Chapter 35, "The CANDISC Procedure."

Art, Gnanadesikan, and Kettenring demonstrate by Monte Carlo calculations that their method can produce better clusters than the Euclidean metric even when the approximation to the within-cluster SSCP matrix is poor or the within-cluster covariances are moderately heterogeneous. The algorithm used by the ACECLUS procedure differs slightly from the algorithm used by Art, Gnanadesikan, and Kettenring. In the following sections, the PROC ACECLUS algorithm is described first; then, differences between PROC ACECLUS and the method used by Art, Gnanadesikan, and Kettenring are summarized.

Background

It is well known from the literature on nonparametric statistics that variances and, hence, covariances can be computed from pairwise differences instead of deviations from means. (For example, Puri and Sen (1971, pp. 51–52) show that the variance is a U statistic of degree 2.) Let $\mathbf{X} = (x_{ij})$ be the data matrix with n observations (rows) and v variables (columns), and let $\bar{x}_j$ be the mean of the jth variable. The sample covariance matrix $\mathbf{S} = (s_{jk})$ is usually defined as

$$s_{jk} = \frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)$$

The matrix **S** can also be computed as

$$s_{jk} = \frac{1}{n(n-1)} \sum_{i=2}^{n} \sum_{h=1}^{i-1} (x_{ij} - x_{hj})(x_{ik} - x_{hk})$$

Let $\mathbf{W} = (w_{jk})$ be the pooled within-cluster covariance matrix, q be the number of clusters, n_c be the number of observations in the cth cluster, and

$$d''_{ic} = \begin{cases} 1 & \text{if observation } i \text{ is in cluster } c \\ 0 & \text{otherwise} \end{cases}$$

The matrix **W** is normally defined as

$$w_{jk} = \frac{1}{n-q} \sum_{c=1}^{q} \sum_{i=1}^{n} d''_{ic}(x_{ij} - \bar{x}_{cj})(x_{ik} - \bar{x}_{ck})$$

where $\bar{x}_{cj}$ is the mean of the jth variable in cluster c. Let

$$d'_{ih} = \begin{cases} \frac{1}{n_c} & \text{if observations } i \text{ and } h \text{are in cluster } c \\ 0 & \text{otherwise} \end{cases}$$

The matrix **W** can also be computed as

$$w_{jk} = \frac{1}{n-q} \sum_{i=2}^{n} \sum_{h=1}^{i-1} d'_{ih}(x_{ij} - x_{hj})(x_{ik} - x_{hk})$$

If the clusters are not known, d'_{ih} cannot be determined. However, an approximation to **W** can be obtained by using instead

$$d_{ih} = \begin{cases} 1 & \text{if } \sum_{j=1}^{v} \sum_{k=1}^{v} m_{jk}(x_{ij} - x_{hj})(x_{ik} - x_{hk}) \le u^2 \\ 0 & \text{otherwise} \end{cases}$$

where u is an appropriately chosen value and $\mathbf{M} = (m_{jk})$ is an appropriate metric. Let $\mathbf{A} = (a_{jk})$ be defined as

$$a_{jk} = \frac{\sum_{i=2}^{n} \sum_{h=1}^{i-1} d_{ih}(x_{ij} - x_{hj})(x_{ik} - x_{hk})}{2 \sum_{i=2}^{n} \sum_{h=1}^{i-1} d_{ih}}$$

If all of the following conditions hold, **A** equals **W**:

- All within-cluster distances in the metric **M** are less than or equal to u.
- All between-cluster distances in the metric **M** are greater than u.
- All clusters have the same number of members n_c.

If the clusters are of unequal size, **A** gives more weight to large clusters than **W** does, but this discrepancy should be of little importance if the population within-cluster covariance matrices are equal. There might be large differences between **A** and **W** if the cutoff u does not discriminate between pairs in the same cluster and pairs in different clusters. Lack of discrimination might occur for one of the following reasons:

- The clusters are not well separated.
- The metric $\mathbf{M}$ or the cutoff u is not chosen appropriately.

In the former case, little can be done to remedy the problem. The remaining question concerns how to choose $\mathbf{M}$ and u. Consider $\mathbf{M}$ first. The best choice for $\mathbf{M}$ is $\mathbf{W}^{-1}$, but $\mathbf{W}$ is not known. The solution is to use an iterative algorithm:

1. Obtain an initial estimate of $\mathbf{A}$, such as the identity or the total-sample covariance matrix. See the INITIAL= option in the PROC ACECLUS statement for more information.
2. Let $\mathbf{M}$ equal $\mathbf{A}^{-1}$.
3. Recompute $\mathbf{A}$ by using the preceding formula.
4. Repeat steps 2 and 3 until the estimate stabilizes.

Convergence is assessed by comparing values of $\mathbf{A}$ on successive iterations. Let $\mathbf{A}_i$ be the value of $\mathbf{A}$ on the ith iteration and $\mathbf{A}_0$ be the initial estimate of $\mathbf{A}$. Let $\mathbf{Z}$ be a user-specified $v \times v$ matrix. See the METRIC= option in the PROC ACECLUS statement for more information. The convergence measure is

$$e_i = \frac{1}{v} \parallel \mathbf{Z}'(\mathbf{A}_i - \mathbf{A}_{i-1})\mathbf{Z} \parallel$$

where $\parallel \cdots \parallel$ indicates the Euclidean norm—that is, the square root of the sum of the squares of the elements of the matrix. In PROC ACECLUS, $\mathbf{Z}$ can be the identity or an inverse factor of $\mathbf{S}$ or diag($\mathbf{S}$). Iteration stops when e_i falls below a user-specified value. See the CONVERGE= option or the MAXITER= option in the PROC ACECLUS statement for more information.

The remaining question of how to choose u has no simple answer. In practice, you must try several different values. PROC ACECLUS provides four different ways of specifying u:

- You can specify a constant value for u. This method is useful if the initial estimate of $\mathbf{A}$ is quite good. See the ABSOLUTE option and the THRESHOLD= option in the PROC ACECLUS statement for more information.
- You can specify a threshold value $t > 0$ that is multiplied by the root mean square distance between observations in the current metric on each iteration to give u. Thus, the value of u changes from iteration to iteration. This method is appropriate if the initial estimate of $\mathbf{A}$ is poor. See the THRESHOLD= option in the PROC ACECLUS statement for more information.
- You can specify a value p, $0 < p < 1$, to be transformed into a distance u such that approximately a proportion p of the pairwise Mahalanobis distances between observations in a random sample from a multivariate normal distribution will be less than u in repeated sampling. The transformation can be computed only if the number of observations exceeds the number of variables, preferably by at least 10 percent. This method also requires a good initial estimate of $\mathbf{A}$. See the PROPORTION= option and the ABSOLUTE option in the PROC ACECLUS statement for more information.

- You can specify a value p, $0 < p < 1$, to be transformed into a value t that is then multiplied by $1/\sqrt{2v}$ times the root mean square distance between observations in the current metric on each iteration to yield u. The value of u changes from iteration to iteration. This method can be used with a poor initial estimate of $\mathbf{A}$. See the PROPORTION= option in the PROC ACECLUS statement for more information.

In most cases, the analysis should begin with the last method, using values of p between 0.5 and 0.01 and using the full covariance matrix as the initial estimate of $\mathbf{A}$.

Proportions p are transformed to distances t by using the formula

$$t^2 = 2v \left\{ \left[F^{-1}_{v,n-v}(p) \right]^{\frac{n-v}{n-1}} \right\}$$

where $F^{-1}_{v,n-v}$ is the quantile (inverse cumulative distribution) function of an F random variable with v and $n - v$ degrees of freedom. The squared Mahalanobis distance between a single pair of observations sampled from a multivariate normal distribution is distributed as $2v$ times an F random variable with v and $n - v$ degrees of freedom. The distances between two pairs of observations are correlated if the pairs have an observation in common. The quantile function is raised to the power given in the preceding formula to compensate approximately for the correlations among distances between pairs of observations that share a member. Monte Carlo studies indicate that the approximation is acceptable if the number of observations exceeds the number of variables by at least 10 percent.

If $\mathbf{A}$ becomes singular, step 2 in the iterative algorithm cannot be performed because $\mathbf{A}$ cannot be inverted. In this case, let $\mathbf{Z}$ be the matrix as defined in discussing the convergence measure, and let $\mathbf{Z}'\mathbf{A}\mathbf{Z} = \mathbf{R}'\mathbf{\Lambda}\mathbf{R}$, where $\mathbf{R}'\mathbf{R} = \mathbf{R}\mathbf{R}' = \mathbf{I}$ and $\mathbf{\Lambda} = (\lambda_{jk})$ is diagonal. Let $\mathbf{\Lambda}^* = (\lambda^*_{jk})$ be a diagonal matrix, where $\lambda^*_{jj} = \max(\lambda_{jj}, g\ \mathrm{trace}(\mathbf{\Lambda}))$, and $0 < g < 1$ is a user-specified singularity criterion (see the SINGULAR= option in the PROC ACECLUS statement for more information). Then $\mathbf{M}$ is computed as $\mathbf{Z}\mathbf{R}'(\mathbf{\Lambda}^*)^{-1}\mathbf{R}\mathbf{Z}'$.

The ACECLUS procedure differs from the method used by Art, Gnanadesikan, and Kettenring (1982) in several respects:

- The Art, Gnanadesikan, and Kettenring method uses the identity matrix as the initial estimate, whereas the ACECLUS procedure enables you to specify any symmetric matrix as the initial estimate and defaults to the total-sample covariance matrix. The default initial estimate in PROC ACECLUS is chosen to yield invariance under nonsingular linear transformations of the data but might sometimes obscure clusters that become apparent if the identity matrix is used.

- The Art, Gnanadesikan, and Kettenring method carries out all computations with SSCP matrices, whereas the ACECLUS procedure uses estimated covariance matrices because covariances are easier to interpret than crossproducts.

- The Art, Gnanadesikan, and Kettenring method uses the m pairs with the smallest distances to form the new estimate at each iteration, where m is specified by the user, whereas the ACECLUS procedure uses all pairs closer than a given cutoff value. Kettenring (1984) says that the m-closest-pairs method seems to give the user more direct control. PROC ACECLUS uses a distance cutoff because it yields a slight decrease in computer time and because in some cases, such as widely separated spherical clusters, the results are less sensitive to the choice of distance cutoff than to the choice of m. Much research remains to be done on this issue.

- The Art, Gnanadesikan, and Kettenring method uses a different convergence measure. Let $\mathbf{A}_i$ be computed on each iteration by using the m-closest-pairs method, and let $\mathbf{B}_i = \mathbf{A}_{i-1}^{-1}\mathbf{A}_i - \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. The convergence measure is equivalent to trace($\mathbf{B}_i^2$).

Analyses of the Fisher (1936) iris data, consisting of measurements of petal and sepal length and width for 50 specimens from each of three iris species, are summarized in Table 27.1. The number of misclassified observations out of 150 is given for four clustering methods:

- k-means as implemented in PROC FASTCLUS with MAXC=3, MAXITER=99, and CONV=0
- Ward's minimum variance method as implemented in PROC CLUSTER
- average linkage on Euclidean distances as implemented in PROC CLUSTER
- the centroid method as implemented in PROC CLUSTER

Each hierarchical analysis is followed by the TREE procedure with NCL=3 to determine cluster assignments at the three-cluster level. Clusters with 20 or fewer observations are discarded by using the DOCK=20 option. The observations in a discarded cluster are considered unclassified.

Each method is applied to the following data:

- the raw data
- the data standardized to unit variance by the STANDARD procedure
- two standardized principal components accounting for 95 percent of the standardized variance and having an identity total-sample covariance matrix, computed by the PRINCOMP procedure with the STD option
- four standardized principal components having an identity total-sample covariance matrix, computed by PROC PRINCOMP with the STD option
- the data transformed by PROC ACECLUS, using seven different settings of the PROPORTION= (P=) option
- four canonical variables having an identity pooled within-species covariance matrix, computed using the CANDISC procedure

Theoretically, the best results should be obtained by using the canonical variables from PROC CANDISC. PROC ACECLUS yields results comparable to those from PROC CANDISC for values of the PROPORTION= option ranging from 0.005 to 0.02. At PROPORTION=0.04, average linkage and the centroid method show some deterioration, but k-means and Ward's method continue to produce excellent classifications. At larger values of the PROPORTION= option, all methods perform poorly, although no worse than with four standardized principal components.

Table 27.1 Number of Misclassified and Unclassified Observations Using the Fisher (1936) Iris Data

	Clustering Method			
Data	***k*-means**	**Ward's**	**Average Linkage**	**Centroid**
Raw data	16	16	25+12	14
Standardized data	25	26	33+4	33+4
Two standardized principal components	29	31	30+9	27+32
Four standardized principal components	39	27	32+7	45+11
Transformed by ACECLUS, P=0.32	39	10+9	7+25	
Transformed by ACECLUS, P=0.16	39	18+9	7+19	7+26
Transformed by ACECLUS, P=0.08	19	9	3+13	5+16
Transformed by ACECLUS, P=0.04	4	5	1+19	3+12
Transformed by ACECLUS, P=0.02	4	3	3	3
Transformed by ACECLUS, P=0.01	4	4	3	4
Transformed by ACECLUS, P=0.005	4	4	4	4
Canonical variables	3	5	4	4+1

— A single number represents misclassified observations with no unclassified observations.

— Where two numbers are separated by a plus sign, the first is the number of observations; the second is the number of unclassified observations.

This example demonstrates the following:

- PROC ACECLUS can produce results as good as those from the optimal transformation.
- PROC ACECLUS can be useful even when the within-cluster covariance matrices are moderately heterogeneous.
- The choice of the distance cutoff as specified by the PROPORTION= or the THRESHOLD= option is important, and several values should be tried.
- Commonly used transformations such as standardization and principal components can produce poor classifications.

Although experience with the Art, Gnanadesikan, and Kettenring and PROC ACECLUS methods is limited, the results so far suggest that these methods help considerably more often than they hinder the subsequent cluster analysis, especially with normal-mixture techniques such as *k*-means and Ward's minimum variance method.

Getting Started: ACECLUS Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/aceclgs.sas.)

The following example demonstrates how you can use the ACECLUS procedure to obtain approximate estimates of the pooled within-cluster covariance matrix and to compute canonical variables for subsequent analysis. You use PROC ACECLUS to preprocess data before you cluster it by using the FASTCLUS or CLUSTER procedure.

Suppose you want to determine whether national figures for birth rates, death rates, and infant death rates can be used to determine certain types or categories of countries. You want to perform a cluster analysis to determine whether the observations can be formed into groups suggested by the data. Previous studies indicate that the clusters computed from this type of data can be elongated and elliptical. Thus, you need to perform a linear transformation on the raw data before the cluster analysis.

The following data[1] from Rouncefield (1995) are the birth rates, death rates, and infant death rates for 97 countries. The following statements create the SAS data set Poverty:

```
data poverty;
   input Birth Death InfantDeath Country &$20. @@;
   datalines;
24.7  5.7  30.8 Albania             12.5 11.9  14.4 Bulgaria
13.4 11.7  11.3 Czechoslovakia        12 12.4   7.6 Former E. Germany

   ... more lines ...

41.7 10.3    66 Zimbabwe
;
```

The data set Poverty contains the character variable Country and the numeric variables Birth, Death, and InfantDeath, which represent the birth rate per thousand, death rate per thousand, and infant death rate per thousand, respectively. The $20. format in the INPUT statement specifies that the variable Country is a character variable with a length of 20. The preceding & enables the reading of blanks in the middle of the country names. The double trailing at sign (@@) in the INPUT statement specifies that observations are input from each line until all values have been read.

It is often useful when beginning a cluster analysis to look at the data graphically. The following statements use the SGPLOT procedure to make a scatter plot of the variables Birth and Death.

```
proc sgplot data=poverty;
   scatter y=Death x=Birth;
run;
```

The plot, displayed in Figure 27.1, indicates the difficulty of dividing the points into clusters. Plots of the other variable pairs (not shown) display similar characteristics. The clusters that comprise these data might be poorly separated and elongated. Data with poorly separated or elongated clusters must be transformed.

[1] These data have been compiled from the United Nations *Demographic Yearbook* 1990 (United Nations publications, Sales No. E/F.91.XII.1, copyright 1991, United Nations, New York) and are reproduced with the permission of the United Nations.

Figure 27.1 Scatter Plot of Original Poverty Data: Birth Rate versus Death Rate

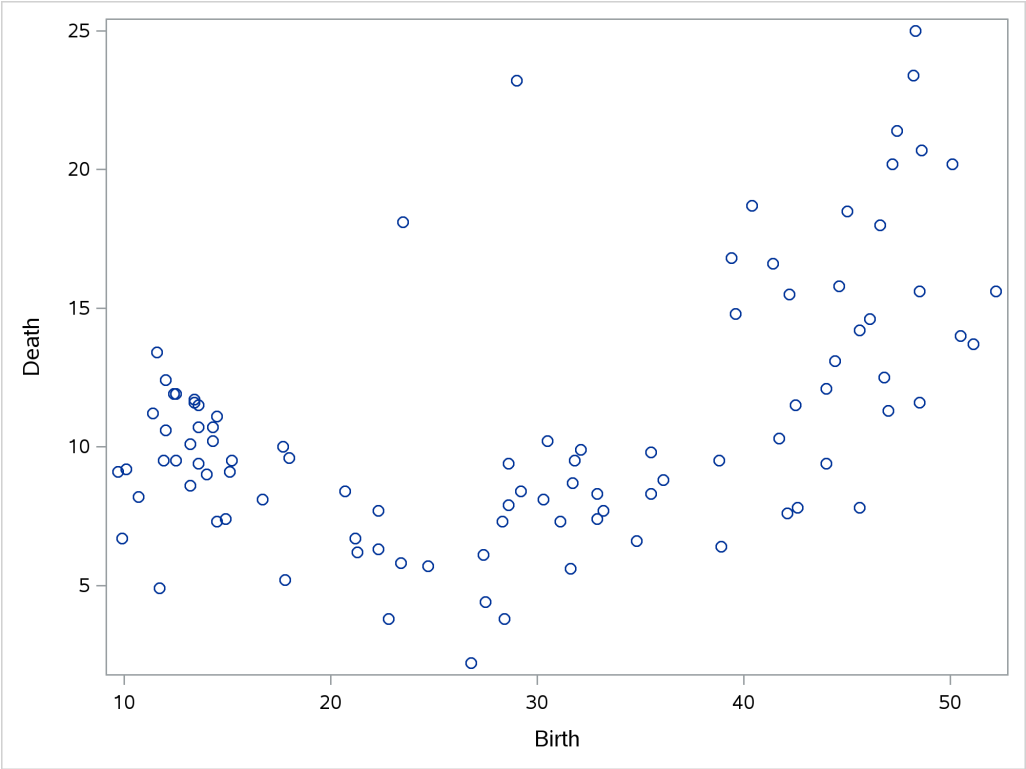

If you know the within-cluster covariances, you can transform the data to make the clusters spherical. However, since you do not know what the clusters are, you cannot calculate exactly the within-cluster covariance matrix. The ACECLUS procedure estimates the within-cluster covariance matrix to transform the data, even when you have no knowledge of cluster membership or the number of clusters.

The following statements perform the ACECLUS procedure transformation by using the SAS data set Poverty:

```
proc aceclus data=poverty out=ace proportion=.03;
   var Birth Death InfantDeath;
run;
```

The OUT= option creates an output data set called Ace to contain the canonical variable scores. The PROPORTION= option specifies that approximately 3 percent of the pairs are included in the estimation of the within-cluster covariance matrix. The VAR statement specifies that the variables Birth, Death, and InfantDeath are used in computing the canonical variables.

The results of this analysis are displayed in Figure 27.2 through Figure 27.5.

Figure 27.2 displays the number of observations, the number of variables, and the settings for the PROPORTION and CONVERGE options. The PROPORTION option is set at 0.03, as specified in the previous statements. The CONVERGE parameter is set at its default value of 0.001. Figure 27.2 next displays the means, standard deviations, and sample covariance matrix of the analysis variables.

Figure 27.2 Means, Standard Deviations, and Covariance Matrix from the ACECLUS Procedure

The ACECLUS Procedure

Approximate Covariance Estimation for Cluster Analysis

Observations	97	**Proportion**	0.0300
Variables	3	**Converge**	0.00100

Means and Standard Deviations

Variable	Mean	Standard Deviation
Birth	29.2299	13.5467
Death	10.8361	4.6475
InfantDeath	54.9010	45.9926

COV: Total Sample Covariances

	Birth	Death	InfantDeath
Birth	183.512951	30.610056	534.794969
Death	30.610056	21.599205	139.925900
InfantDeath	534.794969	139.925900	2115.317811

The type of matrix used for the initial within-cluster covariance estimate is displayed in Figure 27.3. In this example, that initial estimate is the full covariance matrix. The threshold value that corresponds to the PROPORTION=0.03 setting is given as 0.292815.

Figure 27.3 Table of Iteration History from the ACECLUS Procedure

Initial Within-Cluster Covariance Estimate = Full Covariance Matrix

Threshold = 0.292815

Iteration History

Iteration	RMS Distance	Distance Cutoff	Pairs Within Cutoff	Convergence Measure
1	2.449	0.717	385.0	0.552025
2	12.534	3.670	446.0	0.008406
3	12.851	3.763	521.0	0.009655
4	12.882	3.772	591.0	0.011193
5	12.716	3.723	628.0	0.008784
6	12.821	3.754	658.0	0.005553
7	12.774	3.740	680.0	0.003010
8	12.631	3.699	683.0	0.000676

Algorithm converged.

Figure 27.3 displays the iteration history. For each iteration, PROC ACECLUS displays the following measures:

- root mean square distance between all pairs of observations

- distance cutoff for including pairs of observations in the estimate of within-cluster covariances (equal to RMS*Threshold)
- number of pairs within the cutoff
- convergence measure

Figure 27.4 displays the approximate within-cluster covariance matrix and the table of eigenvalues from the canonical analysis. The first column of the eigenvalues table contains numbers for the eigenvectors. The next column of the table lists the eigenvalues of Inv(ACE)*(COV-ACE).

Figure 27.4 Approximate Within-Cluster Covariance Estimates

ACE: Approximate Covariance Estimate Within Clusters	Birth	Death	InfantDeath
Birth	5.94644949	-0.63235725	6.28151537
Death	-0.63235725	2.33464129	1.59005857
InfantDeath	6.28151537	1.59005857	35.10327233

Eigenvalues of Inv(ACE)*(COV-ACE)	Eigenvalue	Difference	Proportion	Cumulative
1	63.5500	54.7313	0.8277	0.8277
2	8.8187	4.4038	0.1149	0.9425
3	4.4149		0.0575	1.0000

The next three columns of the eigenvalue table (Figure 27.4) display measures of the relative size and importance of the eigenvalues. The first column lists the difference between each eigenvalue and its successor. The last two columns display the individual and cumulative proportions that each eigenvalue contributes to the total sum of eigenvalues.

The raw and standardized canonical coefficients are displayed in Figure 27.5. The coefficients are standardized by multiplying the raw coefficients with the standard deviation of the associated variable. The ACECLUS procedure uses these standardized canonical coefficients to create the transformed canonical variables, which are the linear transformations of the original input variables, Birth, Death, and InfantDeath.

Figure 27.5 Raw and Standardized Canonical Coefficients from the ACECLUS Procedure

Eigenvectors (Raw Canonical Coefficients)	Can1	Can2	Can3
Birth	0.125610	0.457037	0.003875
Death	0.108402	0.163792	0.663538
InfantDeath	0.134704	-.133620	-.046266

Standardized Canonical Coefficients	Can1	Can2	Can3
Birth	1.70160	6.19134	0.05249
Death	0.50380	0.76122	3.08379
InfantDeath	6.19540	-6.14553	-2.12790

The following statements invoke the CLUSTER procedure, using the SAS data set Ace created in the previous ACECLUS procedure:

```
proc cluster data=ace outtree=tree noprint method=ward;
   var can1 can2 can3 ;
   copy Birth--Country;
run;
```

The OUTTREE= option creates the output SAS data set Tree that is used in a subsequent step to display cluster membership. The NOPRINT option suppresses the display of the output. The METHOD= option specifies Ward's minimum-variance clustering method.

The VAR statement specifies that the canonical variables computed in the ACECLUS procedure are used in the cluster analysis. The COPY statement specifies that all the variables from the SAS data set Poverty (Birth—Country) are added to the output data set Tree.

The following statements use PROC TREE to create an output SAS data set called New. The NCLUSTERS= option specifies the number of clusters desired in the SAS data set New. The NOPRINT option suppresses the display of the output.

```
proc tree data=tree out=new nclusters=3 noprint;
   copy Birth Death InfantDeath can1 can2 ;
   id Country;
run;
```

The COPY statement copies the canonical variables Can1 and Can2 (computed in the preceding ACECLUS procedure) and the original analysis variables Birth, Death, and InfantDeath into the output SAS data set New.

The following statements invoke the SGPLOT procedure, using the SAS data set created by PROC TREE:

```
proc sgplot data=new;
   scatter y=Death x=Birth / group=cluster;
   keylegend / title="Cluster Membership";
run;

proc sgplot data=new;
   scatter y=can2 x=can1 / group=cluster;
   keylegend / title="Cluster Membership";
run;
```

The first PROC SGPLOT statement requests a scatter plot of the two variables Birth and Death, using the variable CLUSTER as the identification variable.

The second PROC SGPLOT statement requests a plot of the two canonical variables, using the value of the variable CLUSTER as the identification variable.

Figure 27.6 and Figure 27.7 display the separation of the clusters when three clusters are calculated.

Figure 27.6 Scatter Plot of Poverty Data, Identified by Cluster

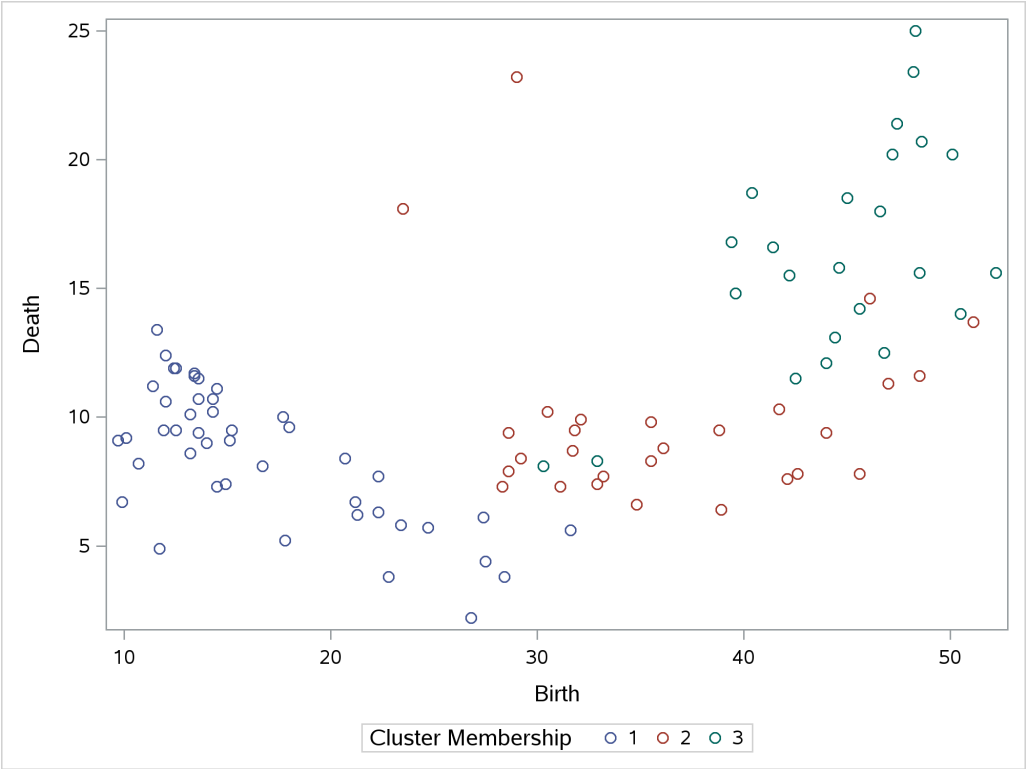

Figure 27.7 Scatter Plot of Canonical Variables

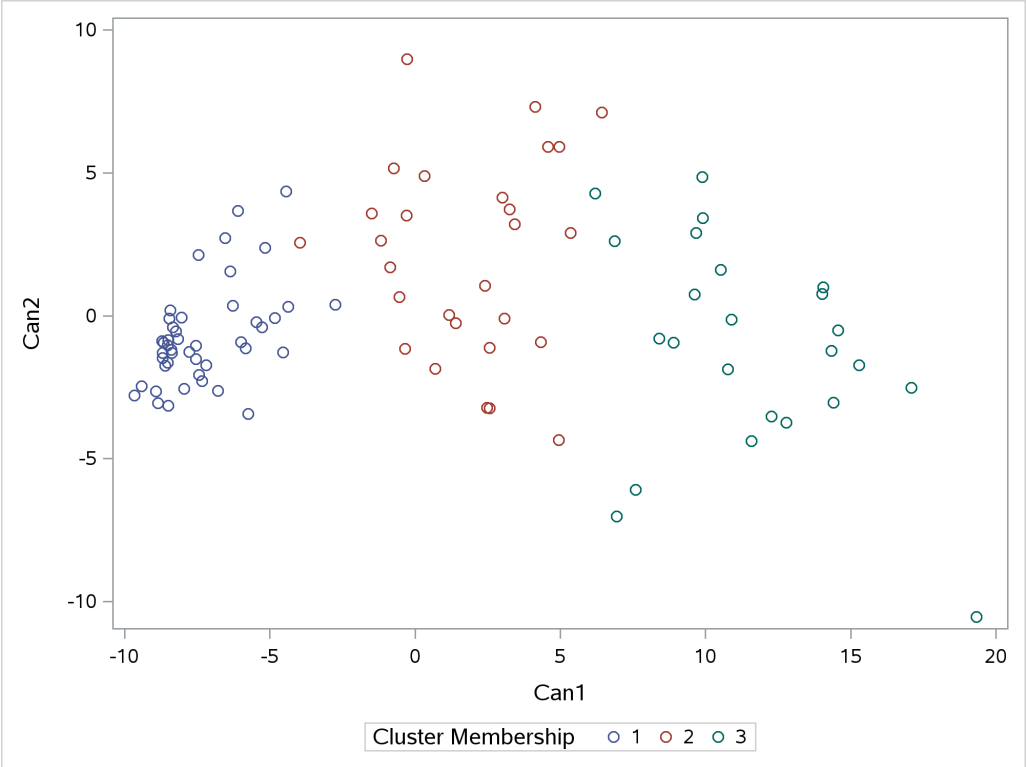

Syntax: ACECLUS Procedure

The following statements are available in the ACECLUS procedure:

```
PROC ACECLUS PROPORTION=p | THRESHOLD=t < options > ;
    BY variables ;
    FREQ variable ;
    VAR variables ;
    WEIGHT variable ;
```

Usually you need only the VAR statement in addition to the required PROC ACECLUS statement. The optional BY, FREQ, VAR, and WEIGHT statements are described in alphabetical order after the PROC ACECLUS statement.

PROC ACECLUS Statement

```
PROC ACECLUS PROPORTION=p | THRESHOLD=t < options > ;
```

The PROC ACECLUS statement invokes the ACECLUS procedure. Table 27.2 summarizes the options available in the PROC ACECLUS statement. These options are also discussed in the following sections. Note that, if you specify the METHOD=COUNT option, you must specify either the PROPORTION= or the MPAIRS= option. Otherwise, you must specify either the PROPORTION= or THRESHOLD= option.

Table 27.2 Summary of PROC ACECLUS Statement Options

Options	Description
Specify Clustering Options	
METHOD=	Specifies the clustering method
MPAIRS=	Specifies number of pairs for estimating within-cluster covariance (when you specify the option METHOD=COUNT)
PROPORTION=	Specifies proportion of pairs for estimating within-cluster covariance
THRESHOLD=	Specifies the threshold for including pairs in the estimation of the within-cluster covariance
Specify Input and Output Data Sets	
DATA=	Specifies input data set name
OUT=	Specifies output data set name
OUTSTAT=	Specifies output data set name containing various statistics
Specify Iteration Options	
ABSOLUTE	Uses absolute instead of relative threshold
CONVERGE=	Specifies convergence criterion
INITIAL=	Specifies initial estimate of within-cluster covariance matrix
MAXITER=	Specifies maximum number of iterations
METRIC=	Specifies metric in which computations are performed
SINGULAR=	Specifies singularity criterion

Table 27.2 *continued*

Options	Description
Specify Canonical Analysis Options	
N=	Specifies number of canonical variables
PREFIX=	Specifies prefix for naming canonical variables
Control Displayed Output	
NOPRINT	Suppresses the display of the output
PLOTS=	Specifies ODS Graphics details
PP	Produces a P-P plot of distances between pairs from last iteration
QQ	Produces a Q-Q plot of power transformation of distances between pairs from last iteration
SHORT	Omits all output except for iteration history and eigenvalue table

The following list provides details about the options.

ABSOLUTE

causes the THRESHOLD= value or the threshold computed from the PROPORTION= option to be treated absolutely rather than relative to the root mean square distance between observations. Use the ABSOLUTE option only when you are confident that the initial estimate of the within-cluster covariance matrix is close to the final estimate, such as when the INITIAL= option specifies a data set created by a previous execution of PROC ACECLUS by using the OUTSTAT= option.

CONVERGE=*c*

specifies the convergence criterion. By default, CONVERGE= 0.001. Iteration stops when the convergence measure falls below the value specified by the CONVERGE= option or when the iteration limit as specified by the MAXITER= option is exceeded, whichever happens first.

DATA=*SAS-data-set*

specifies the SAS data set to be analyzed. By default, PROC ACECLUS uses the most recently created SAS data set.

INITIAL=*name*

specifies the matrix for the initial estimate of the within-cluster covariance matrix. Valid values for *name* are as follows:

DIAGONAL | D uses the diagonal matrix of sample variances as the initial estimate of the within-cluster covariance matrix.

FULL | F uses the total-sample covariance matrix as the initial estimate of the within-cluster covariance matrix.

IDENTITY | I uses the identity matrix as the initial estimate of the within-cluster covariance matrix.

INPUT=*SAS-data-set* specifies a SAS data set from which to obtain the initial estimate of the within-cluster covariance matrix. The data set can be TYPE=CORR, COV, UCORR, UCOV, SSCP, or ACE, or it can be an ordinary SAS data set. See Appendix A, "Special SAS Data Sets," for descriptions of CORR, COV,

UCORR, UCOV, and SSCP data sets. See the section "Output Data Sets" on page 1047 for a description of ACE data sets.

If you do not specify the INITIAL= option, the default is the matrix specified by the METRIC= option. If neither the INITIAL= nor the METRIC= option is specified, INITIAL=FULL is used if there are enough observations to obtain a nonsingular total-sample covariance matrix; otherwise, INITIAL=DIAGONAL is used.

MAXITER=*n*

specifies the maximum number of iterations. By default, MAXITER=10.

METHOD=COUNT | C | THRESHOLD | T

specifies the clustering method. The METHOD=THRESHOLD option requests a method (also the default) that uses all pairs closer than a given cutoff value to form the estimate at each iteration. The METHOD=COUNT option requests a method that uses a number of pairs, m, with the smallest distances to form the estimate at each iteration.

METRIC=*name*

specifies the metric in which the computations are performed, implies the default value for the INITIAL= option, and specifies the matrix $\mathbf{Z}$ used in the formula for the convergence measure e_i and for checking singularity of the $\mathbf{A}$ matrix. Valid values for *name* are as follows:

DIAGONAL | D uses the diagonal matrix of sample variances $\mathrm{diag}(\mathbf{S})$ and sets $\mathbf{Z} = \mathrm{diag}(\mathbf{S})^{-\frac{1}{2}}$, where the superscript $-\frac{1}{2}$ indicates an inverse factor.

FULL | F uses the total-sample covariance matrix $\mathbf{S}$ and sets $\mathbf{Z} = \mathbf{S}^{-\frac{1}{2}}$.

IDENTITY | I uses the identity matrix $\mathbf{I}$ and sets $\mathbf{Z} = \mathbf{I}$.

If you do not specify the METRIC= option, METRIC=FULL is used if there are enough observations to obtain a nonsingular total-sample covariance matrix; otherwise, METRIC=DIAGONAL is used.

The option METRIC= is rather technical. It affects the computations in a variety of ways, but for well-conditioned data the effects are subtle. For most data sets, the METRIC= option is not needed.

MPAIRS=*m*

specifies the number of pairs to be included in the estimation of the within-cluster covariance matrix when METHOD=COUNT is requested. The values of m must be greater than 0 but less than or equal to $(totfq \times (totfq - 1)) / 2$, where $totfq$ is the sum of nonmissing frequencies specified in the FREQ statement. If there is no FREQ statement, $totfq$ equals the number of total nonmissing observations.

N=*n*

specifies the number of canonical variables to be computed. The default is the number of variables analyzed. N=0 suppresses the canonical analysis.

NOPRINT

suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*

creates an output SAS data set that contains all the original data as well as the canonical variables having an estimated within-cluster covariance matrix equal to the identity matrix. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTSTAT=*SAS-data-set*

specifies a TYPE=ACE output SAS data set that contains means, standard deviations, number of observations, covariances, estimated within-cluster covariances, eigenvalues, and canonical coefficients. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PLOTS < =*plot-request* **>**

PLOTS < =(*plot-request* **< . . .** *plot-request* **>) >**

specifies options that control the details of the plots. When you specify only one plot request, you can omit the parentheses.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc aceclus plots=all;
   var _numeric_;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The *plot-requests* include the following:

ALL

produces both P-P and Q-Q plots.

NONE

suppresses all plots.

PP

produces a P-P plot of distances between pairs of observations computed in the last iteration.

QQ

produces a Q-Q plot of a power transformation of the distances between pairs of observations computed in the last iteration. **CAUTION:** The Q-Q plot can require an enormous amount of computer time.

When ODS Graphics is not enabled, you can specify the PP and QQ options (but not PLOTS=(PP QQ)) to create printer plots.

By default, no plots are produced.

PROPORTION=*p*

PERCENT=*p*

P=*p*

specifies the percentage of pairs to be included in the estimation of the within-cluster covariance matrix. The value of *p* must be greater than 0. If *p* is greater than or equal to 1, it is interpreted as a percentage and divided by 100; PROPORTION=0.02 and PROPORTION=2 are equivalent. When you specify METHOD=THRESHOLD, a threshold value is computed from the PROPORTION= option under the assumption that the observations are sampled from a multivariate normal distribution.

When you specify METHOD=COUNT, the number of pairs, m, is computed from PROPORTION=*p* as

$$m = \text{floor}\left(\frac{p}{2} \times totfq \times (totfq - 1)\right)$$

where *totfq* is the number of total nonmissing observations.

PP

produces a P-P plot of distances between pairs of observations computed in the last iteration.

PREFIX=*name*

specifies a prefix for naming the canonical variables. By default the names are Can1, Can2, . . . , CAN*N*. If you specify PREFIX=ABC, the variables are named ABC1, ABC2, ABC3, and so on. The number of characters in the prefix plus the number of digits required to designate the variables should not exceed the name length defined by the VALIDVARNAME= system option. For more information about the VALIDVARNAME= system option, see *SAS System Options: Reference*.

QQ

produces a Q-Q plot of a power transformation of the distances between pairs of observations computed in the last iteration. **CAUTION:** The Q-Q plot can require an enormous amount of computer time.

SHORT

omits all items from the standard output except for the iteration history and the eigenvalue table.

SINGULAR=*g*

SING=*g*

specifies a singularity criterion $0 < g < 1$ for the total-sample covariance matrix $\mathbf{S}$ and the approximate within-cluster covariance estimate $\mathbf{A}$. The default is SINGULAR=1E–4.

THRESHOLD=*t*

T=*t*

specifies the threshold for including pairs of observations in the estimation of the within-cluster covariance matrix. A pair of observations is included if the Euclidean distance between them is less than or equal to *t* times the root mean square distance computed over all pairs of observations.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC ACECLUS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the ACECLUS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

If you specify the INITIAL=INPUT= option and the INITIAL=INPUT= data set does not contain any of the BY variables, the entire INITIAL=INPUT= data set provides the initial value for the matrix $\mathbf{A}$ for each BY group in the DATA= data set.

If the INITIAL=INPUT= data set contains some but not all of the BY variables, or if some BY variables do not have the same type or length in the INITIAL=INPUT= data set as in the DATA= data set, then PROC ACECLUS displays an error message and stops.

If all the BY variables appear in the INITIAL=INPUT= data set with the same type and length as in the DATA= data set, then each BY group in the INITIAL=INPUT= data set provides the initial value for $\mathbf{A}$ for the corresponding BY group in the DATA= data set. All BY groups in the DATA= data set must also appear in the INITIAL= INPUT= data set. The BY groups in the INITIAL=INPUT= data set must be in the same order as in the DATA= data set. If you specify NOTSORTED in the BY statement, identical BY groups must occur in the same order in both data sets. If you do not specify NOTSORTED, some BY groups can appear in the INITIAL= INPUT= data set, but not in the DATA= data set; such BY groups are not used in the analysis.

For more information about BY-group processing, see the “Grouping Data” section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If a variable in your data set represents the frequency of occurrence for the observation, include the name of that variable in the FREQ statement. The procedure then treats the data set as if each observation appears n times, where n is the value of the FREQ variable for the observation. If a value of the FREQ variable is not integral, it is truncated to the largest integer not exceeding the given value. Observations with FREQ values less than one are not included in the analysis. The total number of observations is considered equal to the sum of the FREQ variable.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the numeric variables to be analyzed. If the VAR statement is omitted, all numeric variables not specified in other statements are analyzed.

WEIGHT Statement

WEIGHT *variable* **;**

If you want to specify relative weights for each observation in the input data set, place the weights in a variable in the data set and specify that variable name in a WEIGHT statement. This is often done when the variance associated with each observation is different and the values of the weight variable are proportional to the reciprocals of the variances. The values of the WEIGHT variable can be nonintegral and are not truncated. An observation is used in the analysis only if the value of the WEIGHT variable is greater than zero.

The WEIGHT and FREQ statements have a similar effect, except in calculating the divisor of the $\mathbf{A}$ matrix.

Details: ACECLUS Procedure

Missing Values

Observations with missing values are omitted from the analysis and are given missing values for canonical variable scores in the OUT= data set.

Output Data Sets

OUT= Data Set

The OUT= data set contains all the variables in the original data set plus new variables containing the canonical variable scores. The N= option determines the number of new variables. The OUT= data set is not created if N=0. The names of the new variables are formed by concatenating the value given by the PREFIX= option (or the prefix CAN if the PREFIX= option is not specified) and the numbers 1, 2, 3, and so on. The OUT= data set can be used as input to PROC CLUSTER or PROC FASTCLUS. The cluster analysis should be performed on the canonical variables, not on the original variables.

OUTSTAT= Data Set

The OUTSTAT= data set is a TYPE=ACE data set containing the following variables:

- the BY variables, if any
- the two new character variables, _TYPE_ and _NAME_
- the variables analyzed—that is, those in the VAR statement, or, if there is no VAR statement, all numeric variables not listed in any other statement

Each observation in the new data set contains some type of statistic as indicated by the _TYPE_ variable. The values of the _TYPE_ variable are as follows:

MEAN	mean of each variable
STD	standard deviation of each variable
N	number of observations on which the analysis is based. This value is the same for each variable.
SUMWGT	sum of the weights if a WEIGHT statement is used. This value is the same for each variable.
COV	covariances between each variable and the variable named by the _NAME_ variable. The number of observations with _TYPE_=COV is equal to the number of variables being analyzed.
ACE	estimated within-cluster covariances between each variable and the variable named by the _NAME_ variable. The number of observations with _TYPE_=ACE is equal to the number of variables being analyzed.

EIGENVAL eigenvalues of INV(ACE)*(COV–ACE). If the N= option requests fewer than the maximum number of canonical variables, only the specified number of eigenvalues are produced, with missing values filling out the observation.

RAWSCORE raw canonical coefficients.

To obtain the canonical variable scores, these coefficients should be multiplied by the raw data centered by means obtained from the observation with _TYPE_='MEAN'.

SCORE standardized canonical coefficients. The _NAME_ variable contains the name of the corresponding canonical variable as constructed from the PREFIX= option. The number of observations with _TYPE_=SCORE equals the number of canonical variables computed.

To obtain the canonical variable scores, these coefficients should be multiplied by the standardized data, using means obtained from the observation with _TYPE_='MEAN' and standard deviations obtained from the observation with _TYPE_='STD'.

The OUTSTAT= data set can be used in the following conditions:

- to initialize another execution of PROC ACECLUS
- to compute canonical variable scores with the SCORE procedure
- as input to the FACTOR procedure, specifying METHOD=SCORE, to rotate the canonical variables

Computational Resources

Let

$$n = \text{number of observations}$$

$$v = \text{number of variables}$$

$$i = \text{number of iterations}$$

Memory

The memory in bytes required by PROC ACECLUS is approximately

$$8(2n(v+1) + 21v + 5v^2)$$

bytes. If you request the PP or QQ option, an additional $4n(n-1)$ bytes are needed.

Time

The time required by PROC ACECLUS is roughly proportional to

$$2nv^2 + 10v^3 + i\left(\frac{n^2v}{2} + nv^2 + 5v^3\right)$$

Displayed Output

Unless the SHORT option is specified, the ACECLUS procedure displays the following items:

- Means and Standard Deviations of the input variables
- the **S** matrix, labeled COV: Total Sample Covariances
- the name or value of the matrix used for the Initial Within-Cluster Covariance Estimate
- the Threshold value if the PROPORTION= option is specified

For each iteration, PROC ACECLUS displays the following items:

- the Iteration number
- RMS Distance, the root mean square distance between all pairs of observations
- the Distance Cutoff (u) for including pairs of observations in the estimate of the within-cluster covariances, which equals the RMS distance times the threshold
- the number of Pairs Within Cutoff
- the Convergence Measure (e_i) as specified by the METRIC= option

If the SHORT option is not specified, PROC ACECLUS also displays the **A** matrix, labeled ACE: Approximate Covariance Estimate Within Clusters.

The ACECLUS procedure displays a table of eigenvalues from the canonical analysis containing the following items:

- Eigenvalues of Inv(ACE)*(COV–ACE)
- the Difference between successive eigenvalues
- the Proportion of variance explained by each eigenvalue
- the Cumulative proportion of variance explained

If the SHORT option is not specified, PROC ACECLUS displays the following items:

- the Eigenvectors or raw canonical coefficients
- the standardized eigenvectors or standard canonical coefficients

ODS Table Names

PROC ACECLUS assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 27.3 ODS Tables Produced by PROC ACECLUS

ODS Table Name	Description	Statement	Option
ConvergenceStatus	Convergence status	PROC ACECLUS	Default
DataOptionInfo	Data and option information	PROC ACECLUS	Default
Eigenvalues	Eigenvalues of Inv(ACE)*(COV–ACE)	PROC ACECLUS	Default
Eigenvectors	Eigenvectors (raw canonical coefficients)	PROC ACECLUS	Default
InitWithin	Initial within-cluster covariance estimate	PROC ACECLUS	INITIAL=INPUT
IterHistory	Iteration history	PROC ACECLUS	Default
SimpleStatistics	Simple statistics	PROC ACECLUS	Default
StdCanCoef	Standardized canonical coefficients	PROC ACECLUS	Default
Threshold	Threshold value	PROC ACECLUS	PROPORTION=
TotSampleCov	Total sample covariances	PROC ACECLUS	Default
Within	Approximate covariance estimate within clusters	PROC ACECLUS	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC ACECLUS generates are listed in Table 27.4.

Table 27.4 Graphs Produced by PROC ACECLUS

ODS Graph Name	Plot Description
PPLOT	P-P plot
QQPLOT	Q-Q plot

Example: ACECLUS Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 27.1: Transformation and Cluster Analysis of Fisher Iris Data

(View the complete code for this example (aceclex1.sas) in the example repository.)

The iris data published by Fisher (1936) have been widely used for examples in discriminant analysis and cluster analysis. The sepal length, sepal width, petal length, and petal width are measured in millimeters on 50 iris specimens from each of three species, *Iris setosa, I. versicolor,* and *I. virginica*. Mezzich and Solomon (1980) discuss a variety of cluster analyses of the iris data.

In this example PROC ACECLUS is used to transform the iris data, which is available from the Sashelp library, and the clustering is performed by PROC FASTCLUS. Compare this with the example in Chapter 45, "The FASTCLUS Procedure." The results from the FREQ procedure display fewer misclassifications when PROC ACECLUS is used.

The following statements produce Output 27.1.1 through Output 27.1.5:

```
title 'Fisher (1936) Iris Data';

proc aceclus data=sashelp.iris out=ace p=.02 outstat=score;
   var SepalLength SepalWidth PetalLength PetalWidth ;
run;

proc sgplot data=ace;
   scatter y=can2 x=can1 / group=Species;
   keylegend / title="Species";
run;

proc fastclus data=ace maxc=3 maxiter=10 conv=0 out=clus;
   var can:;
run;

proc freq;
   tables cluster*Species;
run;
```

Output 27.1.1 Using PROC ACECLUS to Transform Fisher's Iris Data

Fisher (1936) Iris Data

The ACECLUS Procedure

Approximate Covariance Estimation for Cluster Analysis

Observations	150	**Proportion**	0.0200
Variables	4	**Converge**	0.00100

Means and Standard Deviations

Variable	Mean	Standard Deviation	Label
SepalLength	58.4333	8.2807	Sepal Length (mm)
SepalWidth	30.5733	4.3587	Sepal Width (mm)
PetalLength	37.5800	17.6530	Petal Length (mm)
PetalWidth	11.9933	7.6224	Petal Width (mm)

COV: Total Sample Covariances

	SepalLength	SepalWidth	PetalLength	PetalWidth
SepalLength	68.5693512	-4.2434004	127.4315436	51.6270694
SepalWidth	-4.2434004	18.9979418	-32.9656376	-12.1639374
PetalLength	127.4315436	-32.9656376	311.6277852	129.5609396
PetalWidth	51.6270694	-12.1639374	129.5609396	58.1006264

Initial Within-Cluster Covariance Estimate = Full Covariance Matrix

Threshold = 0.334211

Iteration History

Iteration	RMS Distance	Distance Cutoff	Pairs Within Cutoff	Convergence Measure
1	2.828	0.945	408.0	0.465775
2	11.905	3.979	559.0	0.013487
3	13.152	4.396	940.0	0.029499
4	13.439	4.491	1506.0	0.046846
5	13.271	4.435	2036.0	0.046859
6	12.591	4.208	2285.0	0.025027
7	12.199	4.077	2366.0	0.009559
8	12.121	4.051	2402.0	0.003895
9	12.064	4.032	2417.0	0.002051
10	12.047	4.026	2429.0	0.000971

Algorithm converged.

Output 27.1.2 Eigenvalues, Raw Canonical Coefficients, and Standardized Canonical Coefficients

ACE: Approximate Covariance Estimate Within Clusters				
	SepalLength	**SepalWidth**	**PetalLength**	**PetalWidth**
SepalLength	11.73342939	5.47550432	4.95389049	2.02902429
SepalWidth	5.47550432	6.91992590	2.42177851	1.74125154
PetalLength	4.95389049	2.42177851	6.53746398	2.35302594
PetalWidth	2.02902429	1.74125154	2.35302594	2.05166735

Eigenvalues of Inv(ACE)*(COV-ACE)				
	Eigenvalue	**Difference**	**Proportion**	**Cumulative**
1	63.7716	61.1593	0.9367	0.9367
2	2.6123	1.5561	0.0384	0.9751
3	1.0562	0.4167	0.0155	0.9906
4	0.6395		0.00939	1.0000

Eigenvectors (Raw Canonical Coefficients)					
		Can1	**Can2**	**Can3**	**Can4**
SepalLength	Sepal Length (mm)	-.012009	-.098074	-.059852	0.402352
SepalWidth	Sepal Width (mm)	-.211068	-.000072	0.402391	-.225993
PetalLength	Petal Length (mm)	0.324705	-.328583	0.110383	-.321069
PetalWidth	Petal Width (mm)	0.266239	0.870434	-.085215	0.320286

Standardized Canonical Coefficients					
		Can1	**Can2**	**Can3**	**Can4**
SepalLength	Sepal Length (mm)	-0.09944	-0.81211	-0.49562	3.33174
SepalWidth	Sepal Width (mm)	-0.91998	-0.00031	1.75389	-0.98503
PetalLength	Petal Length (mm)	5.73200	-5.80047	1.94859	-5.66782
PetalWidth	Petal Width (mm)	2.02937	6.63478	-0.64954	2.44134

Output 27.1.3 Plot of Transformed Iris Data: PROC SGPLOT

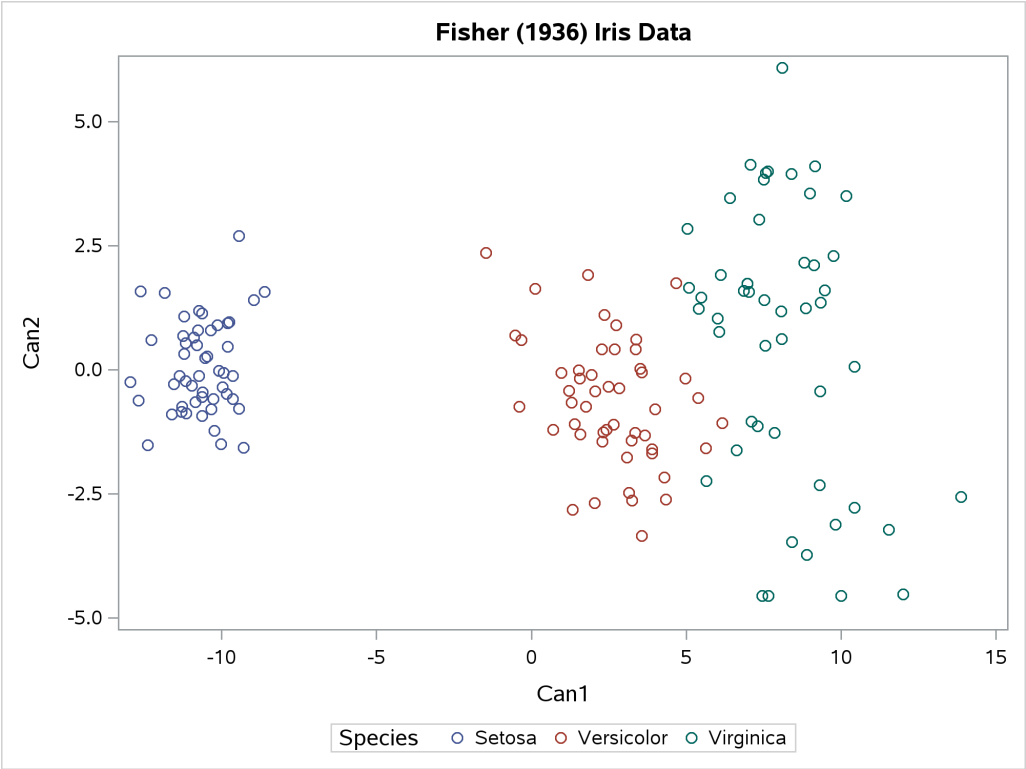

Output 27.1.4 Clustering of Transformed Iris Data: Partial Output from PROC FASTCLUS

Fisher (1936) Iris Data

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=3 Maxiter=10 Converge=0

Cluster Summary

Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	50	1.4138	5.3152		2	5.8580
2	50	1.8880	6.8298		1	5.8580
3	50	1.1016	5.2768		1	13.2845

Statistics for Variables

Variable	Total STD	Within STD	R-Square	RSQ/(1-RSQ)
Can1	8.04808	1.48537	0.966394	28.756658
Can2	1.90061	1.85646	0.058725	0.062389
Can3	1.43395	1.32518	0.157417	0.186826
Can4	1.28044	1.27550	0.021025	0.021477
OVER-ALL	4.24499	1.50298	0.876324	7.085666

Pseudo F Statistic = 520.80

Output 27.1.4 *continued*

Approximate Expected Over-All R-Squared =	0.80391

Cubic Clustering Criterion =	5.179

WARNING: The two values above are invalid for correlated variables.

Cluster Means				
Cluster	**Can1**	**Can2**	**Can3**	**Can4**
1	2.54528754	-0.59273569	-0.78905317	-0.26079612
2	8.12988211	0.52566663	0.51836499	0.14915404
3	-10.67516964	0.06706906	0.27068819	0.11164209

Cluster Standard Deviations				
Cluster	**Can1**	**Can2**	**Can3**	**Can4**
1	1.572366584	1.393565864	1.303411851	1.372050319
2	1.799159552	2.743869556	1.270344142	1.370523175
3	0.953761025	0.931943571	1.398456061	1.058217627

Output 27.1.5 Crosstabulation of Cluster by Species for Fisher's Iris Data: PROC FREQ

Fisher (1936) Iris Data

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of CLUSTER by Species			
		Species(Iris Species)		
CLUSTER(Cluster)	**Setosa**	**Versicolor**	**Virginica**	**Total**
1	0 0.00 0.00 0.00	48 32.00 96.00 96.00	2 1.33 4.00 4.00	50 33.33
2	0 0.00 0.00 0.00	2 1.33 4.00 4.00	48 32.00 96.00 96.00	50 33.33
3	50 33.33 100.00 100.00	0 0.00 0.00 0.00	0 0.00 0.00 0.00	50 33.33
Total	50 33.33	50 33.33	50 33.33	150 100.00

References

Art, D., Gnanadesikan, R., and Kettenring, J. R. (1982). "Data-Based Metrics for Cluster Analysis." *Utilitas Mathematica* 75–99.

Everitt, B. S. (1980). *Cluster Analysis*. 2nd ed. London: Heineman Educational Books.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Hartigan, J. A. (1975). *Clustering Algorithms*. New York: John Wiley & Sons.

Kettenring, J. R. (1984). Personal communication.

Mezzich, J. E., and Solomon, H. (1980). *Taxonomy and Behavioral Science*. New York: Academic Press.

Puri, M. L., and Sen, P. K. (1971). *Nonparametric Methods in Multivariate Analysis*. New York: John Wiley & Sons.

Rouncefield, M. (1995). "The Statistics of Poverty and Inequality." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Wolfe, J. H. (1970). "Pattern Clustering by Multivariate Mixture Analysis." *Multivariate Behavioral Research* 5:329–350.

Chapter 28
The ADAPTIVEREG Procedure

Contents

Overview: ADAPTIVEREG Procedure **1058**
Getting Started: ADAPTIVEREG Procedure **1059**
Syntax: ADAPTIVEREG Procedure **1066**
- PROC ADAPTIVEREG Statement 1067
- BY Statement 1074
- CLASS Statement 1075
- FREQ Statement 1076
- MODEL Statement 1076
- OUTPUT Statement 1082
- PARTITION Statement 1084
- SCORE Statement 1085
- WEIGHT Statement 1086

Details: ADAPTIVEREG Procedure **1086**
- Fitting Algorithms 1086
- Missing Values 1091
- ANOVA Decomposition 1092
- Computational Resources 1092
- ODS Table Names 1094
- ODS Graphics 1095

Examples: ADAPTIVEREG Procedure **1096**
- Example 28.1: Surface Fitting with Many Noisy Variables 1096
- Example 28.2: Fitting Data with Mixture Structures 1101
- Example 28.3: Predicting E-Mail Spam 1105
- Example 28.4: Nonparametric Poisson Model for Mackerel Egg Density 1110

References **1115**

Overview: ADAPTIVEREG Procedure

The ADAPTIVEREG procedure fits multivariate adaptive regression splines as defined by Friedman (1991b). The method is a nonparametric regression technique that combines both regression splines and model selection methods. It does not assume parametric model forms and does not require specification of knot values for constructing regression spline terms. Instead, it constructs spline basis functions in an adaptive way by automatically selecting appropriate knot values for different variables and obtains reduced models by applying model selection techniques.

PROC ADAPTIVEREG supports models with classification variables (Friedman 1991a) and offers options for improving modeling speed (Friedman 1993). PROC ADAPTIVEREG also extends the method to data with response variables that are distributed in the exponential family as suggested in Buja et al. (1991). The procedure can take advantage of multicore processors to distribute the computation to multiple threads.

SAS/STAT software offers various tools for nonparametric regression, including the GAM, LOESS, and TPSPLINE procedures. Typical nonparametric regression methods involve a large number of parameters in order to capture nonlinear trends in data; thus the model space is much larger than it is in more restricted parametric models. The fitting algorithms for nonparametric regression models are usually more complicated than for parametric regression models. Also, the sparsity of data in high dimensions often causes slow convergence or failure in many nonparametric regression methods. As the number of predictors increases, the model variance increases rapidly because of the sparsity. This phenomenon is referred as the "curse of dimensionality" (Bellman 1961). Hence, the LOESS and TPSPLINE procedures are limited to problems in low dimensions. PROC GAM fits generalized additive models with the additivity assumption. By using the local scoring algorithm (Hastie and Tibshirani 1990), PROC GAM can handle larger data sets than the other two procedures. However, the computation time for the local scoring algorithm to converge increases rapidly as data size grows, and the convergence for nonnormal distributions is not guaranteed. PROC ADAPTIVEREG uses the multivariate adaptive regression splines method, which is similar to the method used for the recursive partitioning models (Breiman et al. 1984). It creates an overfitted model first with the fast-update algorithm (Friedman 1991b); then prunes it back with the backward selection technique.

The main features of the ADAPTIVEREG procedure are as follows:

- supports classification variables with ordering options
- enables you to force effects in the final model or restrict variables in linear forms
- supports options for fast forward selection
- supports data with response variables that are distributed in the exponential family
- supports partitioning of data into training, validation, and testing roles
- provides leave-one-out and k-fold cross validation
- produces a graphical representation of the selection process, model fit, functional components, and fit diagnostics
- produces an output data set that contains predicted values and residuals
- produces an output data set that contains the design matrix of formed basis functions
- supports multiple SCORE statements

Getting Started: ADAPTIVEREG Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/adptgs1.sas.)

This example concerns city-cycle fuel efficiency and automobile characteristics for 361 vehicle models made from year 1970 to 1982. The data can be downloaded from the UCI Machine Learning Repository (Asuncion and Newman 2007). The following DATA step creates the data set autompg:

```
title 'Automobile MPG Study';
data Autompg;
   input MPG Cylinders Displacement Horsepower Weight
         Acceleration Year Origin Name $35.;
   datalines;
18.0 8 307.0   130.0   3504   12.0   70  1  Chevrolet Chevelle Malibu
15.0 8 350.0   165.0   3693   11.5   70  1  Buick Skylark 320
18.0 8 318.0   150.0   3436   11.0   70  1  Plymouth Satellite
16.0 8 304.0   150.0   3433   12.0   70  1  AMC Rebel SST

   ... more lines ...

44.0 4 97.00   52.00   2130   24.6   82  2  VW Pickup
32.0 4 135.0   84.00   2295   11.6   82  1  Dodge Rampage
28.0 4 120.0   79.00   2625   18.6   82  1  Ford Ranger
31.0 4 119.0   82.00   2720   19.4   82  1  Chevy S-10
;
```

There are nine variables in the data set. The response variable MPG is city-cycle mileage per gallon (MPG). Seven predictor variables (Cylinders, Displacement, HorsePower, Weight, Acceleration, Year, and Origin) provide vehicle attributes. Among them, Cylinders, Year, and Origin are categorical variables. The last variable, Name, contains the specific name of each vehicle model.

The dependency of vehicle fuel efficiency on various factors might be nonlinear. There might also be redundant predictor variables as a result of dependency structures within predictors. For example, a vehicle model with more cylinders is likely to have more horsepower. The objective of this example is to explore the nonlinear dependency structure and also to produce a parsimonious model that does not overfit and thus has good predictive power. The following invocation of the ADAPTIVEREG procedure fits an additive model with linear spline terms of continuous predictors. By default, PROC ADAPTIVEREG fits a nonparametric regression model that includes two-way interaction between spline basis functions. You can try models with even higher interaction orders by specifying the MAXORDER= option in the MODEL statement. For this particular data set, the sample size is relatively small. Restricting model complexity by specifying an additive model can both improve model interpretability and reduce model variance without sacrificing much predictive power. The additive model consists of terms of nonparametric transformations of variables. The transformation of each variable and the selection of transformed terms are performed in an adaptive and automatic way.

```
ods graphics on;

proc adaptivereg data=autompg plots=all;
   class cylinders year origin;
   model mpg = cylinders displacement horsepower
```

```
                weight acceleration year origin / additive;
run;
```

PROC ADAPTIVEREG summarizes important information about the model that you are fitting in Figure 28.1.

Figure 28.1 Model Information and Fit Controls

Automobile MPG Study

The ADAPTIVEREG Procedure

Model Information	
Data Set	WORK.AUTOMPG
Response Variable	MPG
Class Variables	Cylinders Year Origin
Distribution	Normal
Link Function	Identity

Fit Controls	
Maximum Number of Bases	21
Maximum Order of Interaction	1
Degrees of Freedom per Knot	2
Knot Separation Parameter	0.05
Variable Parsimony Parameter	0
Missing Value Handling	Include

In addition to listing classification variables in the "Model Information" table, PROC ADAPTIVEREG displays level information about the classification variables that are specified in the CLASS statement. The table in Figure 28.2 lists the levels of the classification variables Cylinders, Year, and Origin. Although the values of Cylinders and Year are naturally ordered, they are treated as ordinary classification variables.

Figure 28.2 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
Cylinders	5	3 4 5 6 8
Year	13	70 71 72 73 74 75 76 77 78 79 80 81 82
Origin	3	1 2 3

The "Fit Statistics" table (Figure 28.3) lists summary statistics of the fitted regression spline model. Because the final model is essentially a linear model, several fit statistics are reported as if the model were fitted with basis functions as predetermined effects. However, because the model selection process and the determination of basis functions are highly nonlinear, additional statistics that incorporate the extra source of degrees of freedom are also displayed. The statistics include effective degrees of freedom, the generalized cross validation (GCV) criterion, and the GCV R-square value.

Figure 28.3 Fit Statistics

Fit Statistics	
GCV	12.64769
GCV R-Square	0.79349
Effective Degrees of Freedom	25
R-Square	0.81770
Adjusted R-Square	0.81202
Mean Square Error	11.48378
Average Square Error	11.10869

The "Parameter Estimates" table (Figure 28.4) displays both parameter estimates for constructed basis functions and each function's construction components. The basis functions are constructed as two-way interaction terms from parent basis functions and transformations of variables. For continuous variables, the transformations are linear spline functions with knot values specified in the Knot column. For classification variables, the transformations are formed by dichotomizing the variables based on levels specified in the Levels column.

Figure 28.4 Parameter Estimates

Regression Spline Model after Backward Selection					
Name	**Coefficient**	**Parent**	**Variable**	**Knot**	**Levels**
Basis0	29.3788		Intercept		
Basis2	0.003577	Basis0	Weight	3139.00	
Basis3	-4.0349	Basis0	Horsepower	.	
Basis5	-0.05079	Basis3	Horsepower	102.00	
Basis6	0.1925	Basis3	Horsepower	102.00	
Basis7	2.6665	Basis0	Year		10 12 11 9 8 7 3 2
Basis9	-0.6600	Basis0	Displacement	85.0000	
Basis11	0.6394	Basis0	Displacement	97.0000	
Basis13	1.6047	Basis0	Acceleration	21.0000	
Basis14	0.5071	Basis0	Acceleration	21.0000	
Basis16	-0.2960	Basis3	Displacement	105.00	
Basis17	1.7761	Basis0	Origin		2
Basis19	-7.0066	Basis0	Cylinders		0

During the model construction and selection process, some basis function terms are removed. You can view the backward elimination process in the selection plot (Figure 28.5). The plot displays how the model sum of squared error and the corresponding GCV criterion change along with the backward elimination process. The sum of squared error increases as more basis functions are removed from the full model. The GCV criterion decreases at first when two basis functions are dropped and increases afterward. The vertical line indicates the selected model that has the minimum GCV value.

Figure 28.5 Selection Plot

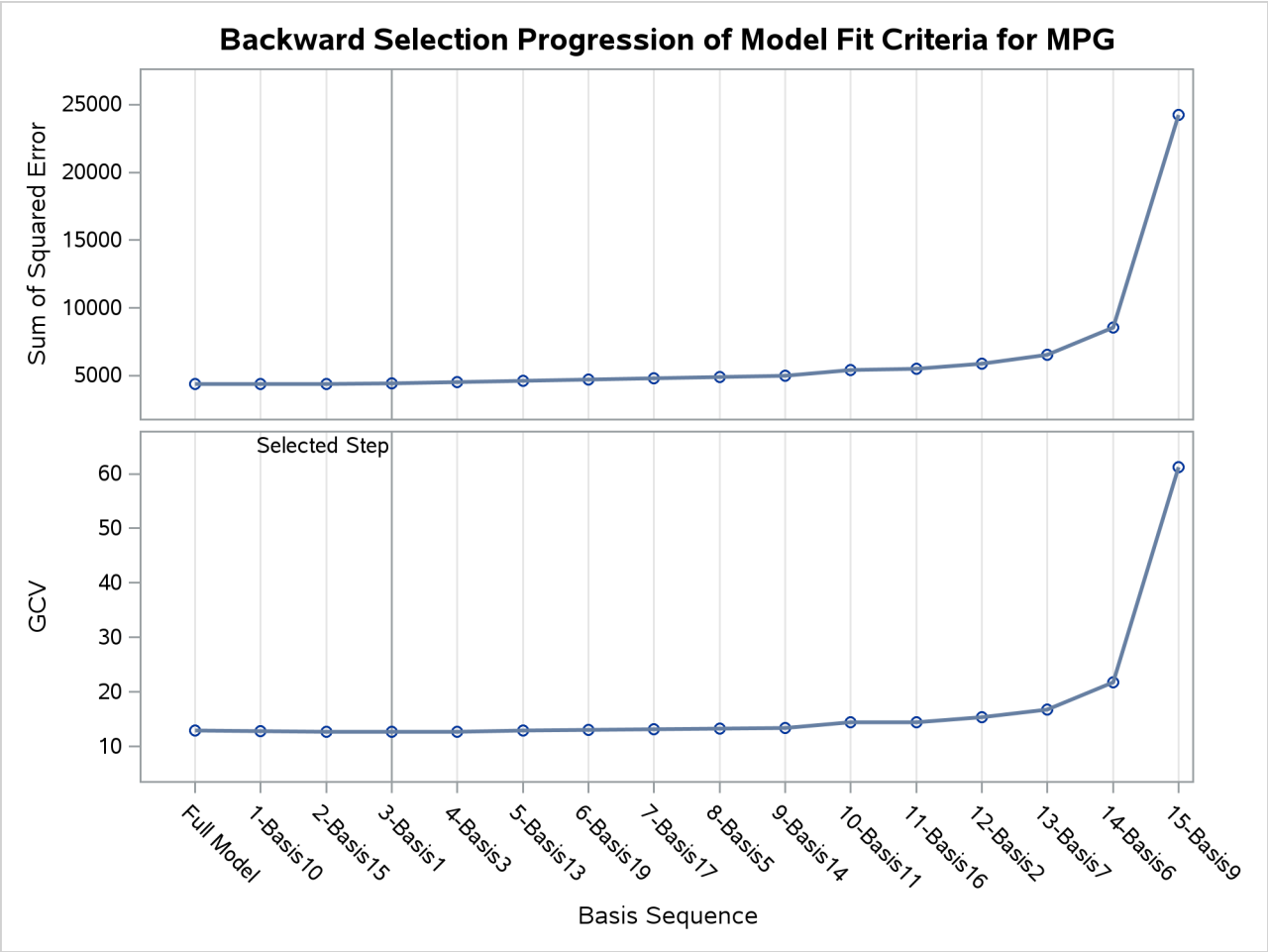

The formed model is an additive model. Basis functions of same variables can be grouped together to form functional components. The "ANOVA Decomposition" table (Figure 28.6) shows functional components and their contribution to the final model.

Figure 28.6 ANOVA Decomposition

ANOVA Decomposition

Functional Component	Number of Bases	DF	Change If Omitted Lack of Fit	Change If Omitted GCV
Weight	1	2	122.85	0.2131
Horsepower	2	4	746.28	1.8228
Year	1	2	620.89	1.6227
Displacement	3	6	463.12	0.8859
Acceleration	2	4	312.83	0.6090
Origin	1	2	133.52	0.2433
Cylinders	1	2	114.56	0.1897

Another criterion that focuses on the contribution of each individual variable is variable importance. It is defined to be the square root of the GCV value of a submodel from which all basis functions that involve a variable have been removed, minus the square root of the GCV value of the selected model, then scaled to have the largest importance value, 100. The table in Figure 28.7 lists importance values, sorted in descending order, for the variables that compose the selected model.

Figure 28.7 Variable Importance

Variable Importance		
Variable	Number of Bases	Importance
Horsepower	2	100.00
Year	1	89.34
Displacement	3	49.44
Acceleration	2	34.17
Origin	1	13.75
Weight	1	12.05
Cylinders	1	10.73

The component panels (Figure 28.8 and Figure 28.9) display the fitted functional components against their forming variables.

Figure 28.8 Component Panel 1

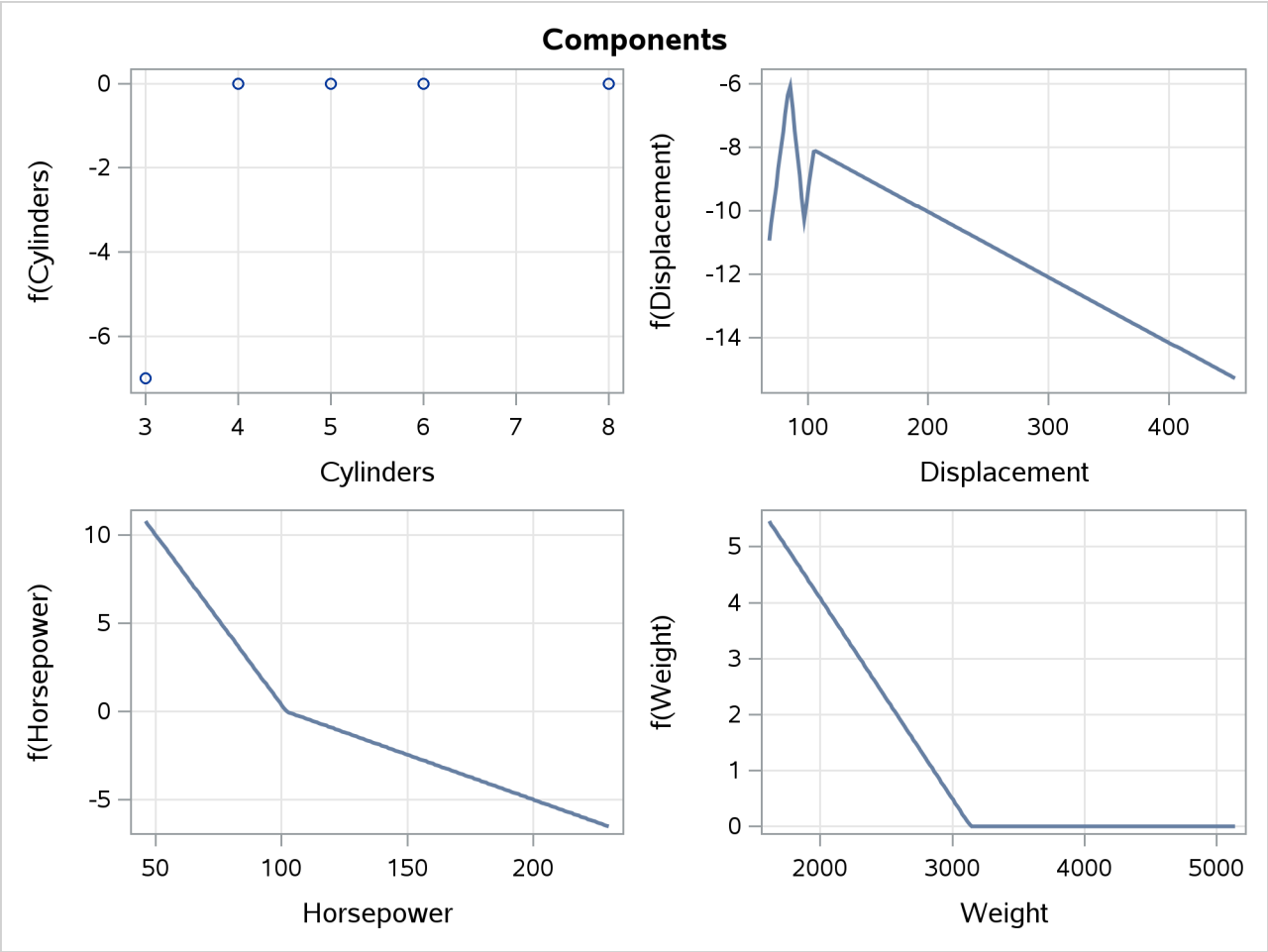

Figure 28.9 Component Panel 2

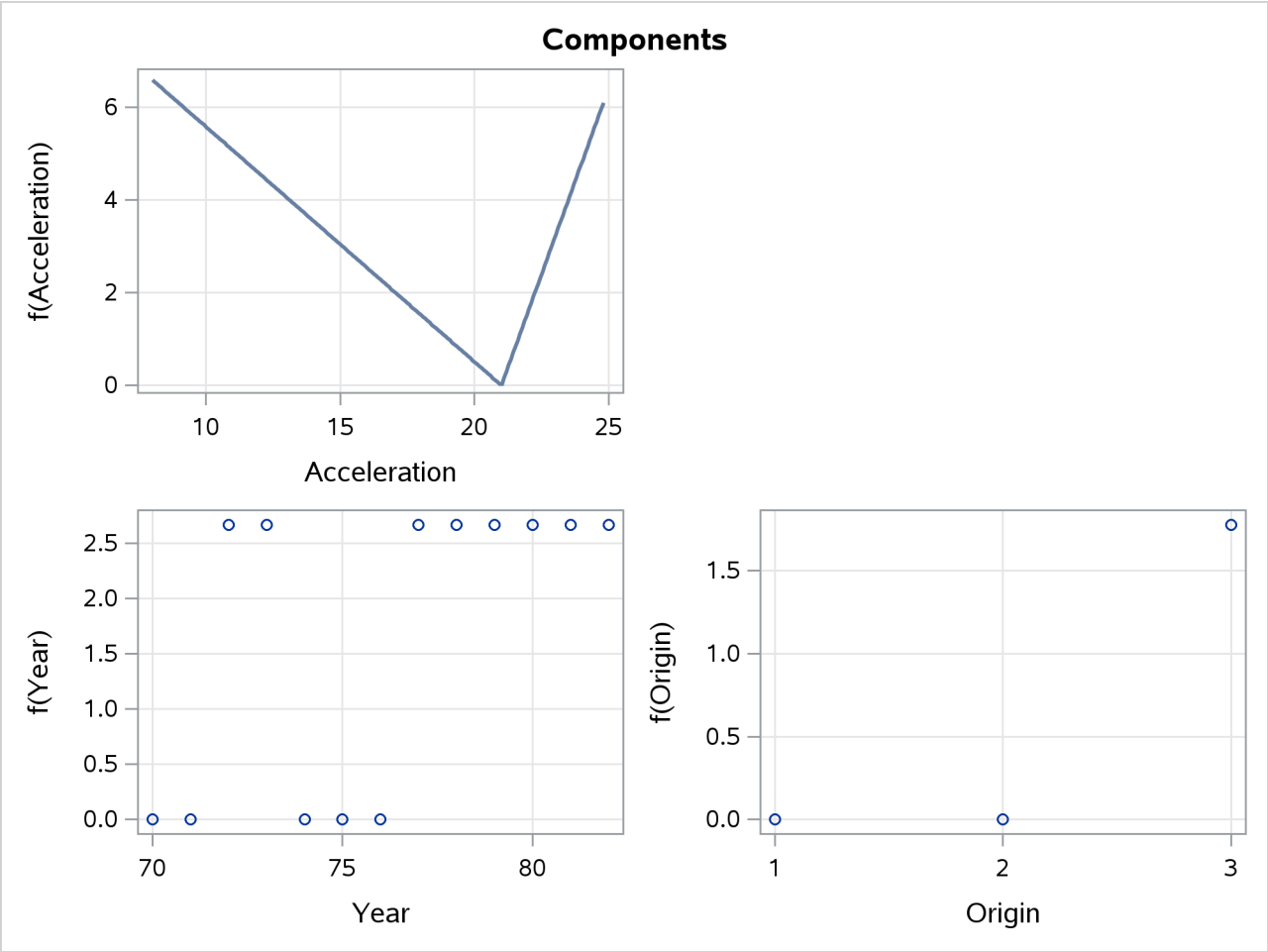

Figure 28.10 shows a panel of fit diagnostics for the selected model that indicate a reasonable fit.

PROC ADAPTIVEREG provides an adaptive way to fit parsimonious regression spline models. The nonparametric transformation of variables is automatically determined, and model selection methods are used to reduce model complexity. The final model based on piecewise linear splines is easy to interpret and highly portable. It can also be used to suggest parametric forms based on the nonlinear trend.

Figure 28.10 Diagnostics Panel

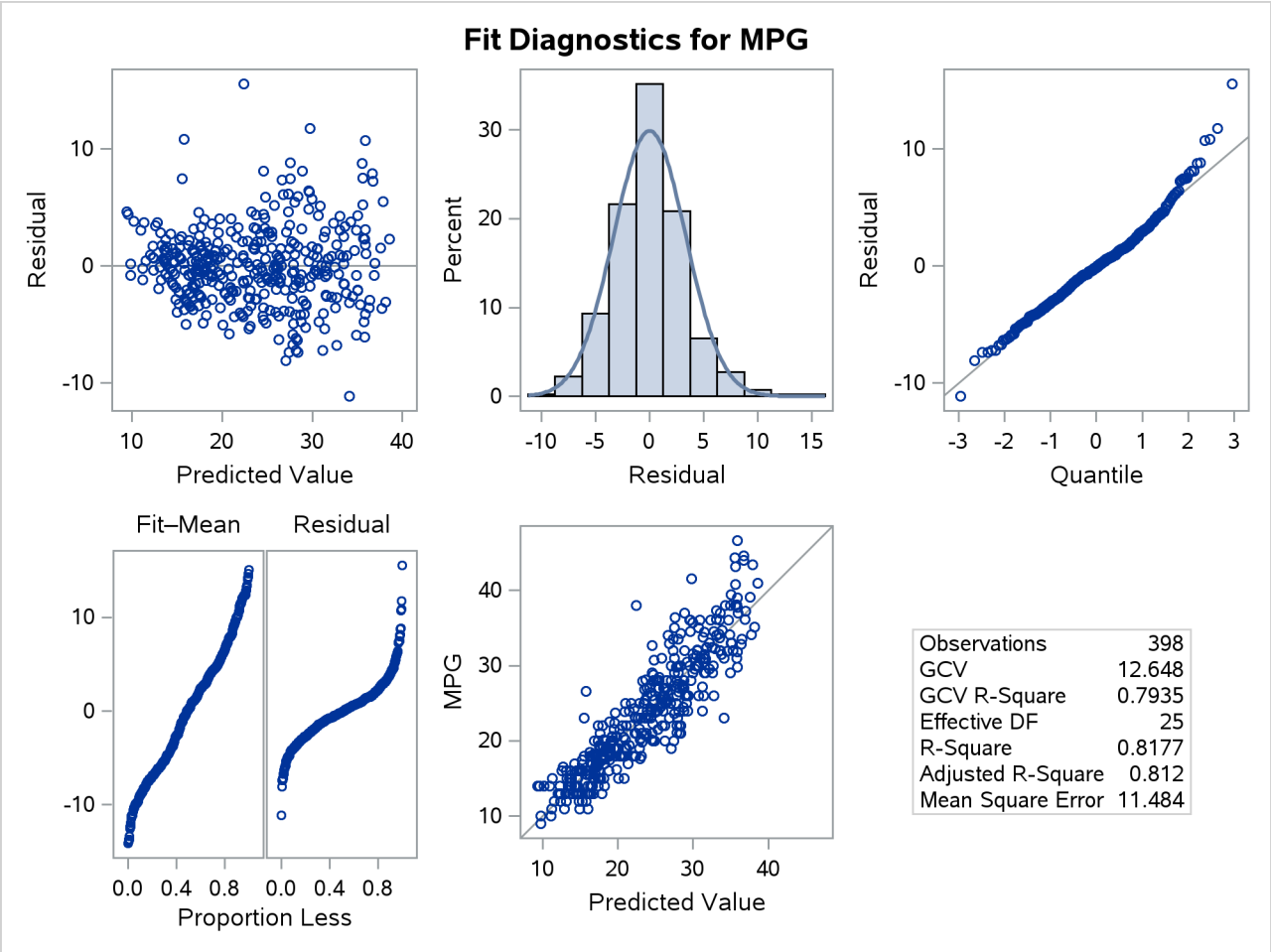

Syntax: ADAPTIVEREG Procedure

The following statements are available in the ADAPTIVEREG procedure:

PROC ADAPTIVEREG < *options* > **;**
  BY *variables* **;**
  CLASS *variables* < / *options* > **;**
  FREQ *variable* **;**
  MODEL *dependent* < **(***options***)** > **=** < *effects* >< / *options* > **;**
  OUTPUT < **OUT=***SAS-data-set* > < *keyword* < **(***keyword-options* **)** > < **=***name* > > . . .
    < *keyword* < **(***keyword-options* **)** > < **=***name* > > **;**
  PARTITION < *options* > **;**
  SCORE < **DATA=***SAS-data-set* > < **OUT=***SAS-data-set* >
    < *keyword* < **=***name* > >. . . < *keyword* < **=***name* > > **;**
  WEIGHT *variable* **;**

The syntax of the ADAPTIVEREG procedure is similar to that of other regression procedures in the SAS System. The PROC ADAPTIVEREG and MODEL statements are required, and the MODEL statement must appear after the CLASS statement if a CLASS statement is included. The SCORE statement can appear multiple times; all other statements can appear only once.

The following sections describe the PROC ADAPTIVEREG statement and then describe the other statements in alphabetical order.

PROC ADAPTIVEREG Statement

PROC ADAPTIVEREG < *options* > **;**

The PROC ADAPTIVEREG statement invokes the procedure.

Table 28.1 summarizes the *options* available in the PROC ADAPTIVEREG statement.

Table 28.1 PROC ADAPTIVEREG Statement Options

Option	Description
Data Set Options	
DATA=	Specifies the input SAS data set
TESTDATA=	Names a data set that contains test data
VALDATA=	Names a data set that contains validation data
Computational Options	
NLOPTIONS	Sets optimization parameters for fitting generalized linear models
SELFUZZ=	Sets the fuzzy comparison criterion in selection
SINGULAR=	Sets the singularity tolerance
Display Options	
NAMELEN=	Sets the length of effect names in tables and output data sets
PLOTS=	Controls plots produced through ODS Graphics
DETAILS=	Displays detailed modeling information
Other Options	
NOTHREADS	Requests the computation in single-threaded mode
OUTDESIGN=	Requests a data set that contains the design matrix
SEED=	Sets the seed used for pseudo-random number generation
NTHREADS=	Specifies the number of threads for the computation

You can specify the following *options*.

DATA=*SAS-data-set*
specifies the SAS data set to be read by PROC ADAPTIVEREG. If you do not specify the DATA= option, PROC ADAPTIVEREG uses the most recently created SAS data set.

DETAILS< =(*detail-options*) >
requests detailed model fitting information. You can specify the following *detail-options*:

BASES
displays the "Bases Information" table.

BWDSUMMARY
displays the "Backward Selection Summary" table.

FWDSUMMARY
displays the "Forward Selection Summary" table.

FWDPARAMS
displays the "Forward Selection Parameter Estimates" table.

If you do not specify a *detail-option*, PROC ADAPTIVEREG produces all the preceding tables by default.

NAMELEN=*number*
specifies the length to which long effect names are shortened. The default and minimum value is 20.

NLOPTIONS(*options*)
specifies options for the nonlinear optimization methods if you are applying the multivariate adaptive regression splines algorithm to generalized linear models. You can specify the following *options*:

ABSCONV=*r*
ABSTOL=*r*
specifies an absolute function convergence criterion by which minimization stops when $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of *r* is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

ABSFCONV=*r*
ABSFTOL=*r*
specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, termination requires a small change of the function value in successive iterations,

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex with the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default value is *r*=0.

ABSGCONV=*r*
ABSGTOL=*r*
specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small,

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the *j*th parameter. This criterion is not used by the NMSIMP technique. The default value is *r* = 1E–5.

FCONV=*r*

FTOL=*r*

specifies a relative function convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex with the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default is $r = 10^{-\mathrm{FDIGITS}}$, where FDIGITS is by default $-\log_{10}\{\epsilon\}$ and ϵ is the machine precision.

GCONV=*r*

GTOL=*r*

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, termination requires the normalized predicted function reduction to be small,

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the *CONGRA* technique (where a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \, |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The default value is r = 1E–8.

HESSIAN=*hessian-options*

specifies the Hessian matrix type used in the optimization of likelihood functions, if the Newton-Raphson technique is used. You can specify the following *hessian-options*:

EXPECTED

requests that the Hessian matrix in optimization be computed as the negative of the expected information matrix.

OBSERVED

requests that the Hessian matrix in optimization be computed as the negative of the observed information matrix. For many specified distribution families and link functions, the observed information matrix is equal to the expected information matrix.

The default is HESSIAN=EXPECTED.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, and NEWRAP: 125
- QUANEW and DBLDOG: 500
- CONGRA: 1000
- NMSIMP: 3000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that is specified by this option. You can select the optimization technique by specifying the TECHNIQUE= option.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, and NEWRAP: 50
- QUANEW and DBLDOG: 200
- CONGRA: 400
- NMSIMP: 1000

These default values also apply when *n* is specified as a missing value. You can select the optimization technique by specifying the TECHNIQUE= option.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The time is checked only at the end of each iteration. Therefore, the actual run time might be longer than the specified time. By default, CPU time is not limited.

MINITER=*n*

MINIT=*n*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

TECHNIQUE=*keyword*

specifies the optimization technique to obtain maximum likelihood estimates for nonnormal distributions. You can choose from the following techniques by specifying the appropriate *keyword*:

CONGRA performs a conjugate-gradient optimization.

DBLDOG performs a version of double-dogleg optimization.

NEWRAP performs a Newton-Raphson optimization that combines a line-search algorithm with ridging.

NMSIMP performs a Nelder-Mead simplex optimization.

NONE performs no optimization.

NRRIDG performs a Newton-Raphson optimization with ridging.

QUANEW performs a dual quasi-Newton optimization.

TRUREG performs a trust-region optimization.

The default is TECHNIQUE=NEWRAP.

For more information about these optimization methods, see the section "Choosing an Optimization Algorithm" on page 545 in Chapter 20, "Shared Concepts and Topics."

NOTHREADS

forces single-threaded execution of the analytic computations. This overrides the SAS system option THREADS | NOTHREADS. Specifying this option is equivalent to specifying the NTHREADS=1 option.

OUTDESIGN< (*options*) >=*SAS-data-set*

creates a data set that contains the design matrix of constructed basis functions. The design matrix column names consist of a prefix followed by an index. The default naming prefix is _X. The default output is the design matrix of basis functions after backward selection.

You can specify the following *options* in parentheses to control the content of the OUTDESIGN= data set:

BACKWARDMODEL | BACKWARD

produces the design matrix for the selected model after the backward selection.

FORWARDMODEL | FORWARD

produces the design matrix for the selected model after the forward selection.

PREFIX=*prefix*

requests that the design matrix column names consist of a prefix followed by an index.

STARTMODEL

produces the design matrix for the initial model specified in the MODEL statement.

PLOTS < (*global-plot-options*) > < = *plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < = (*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the plots produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. For example:

```
plots=all
plots=components(unpack)
plots(unpack)=(components diagnostics)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc adaptivereg plots=all;
   model y=x1 x2;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *global-plot-option*, which applies to all plots that the ADAPTIVEREG procedure generates:

UNPACK | UNPACKPANEL
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACK to get each plot individually. You can also specify UNPACK as a suboption with COMPONENTS and DIAGNOSTICS.

You can specify the following *plot-requests* and their options:

ALL
requests that all default plots be produced.

COMPONENTS < (*component-options*) >
plots a panel of functional components of the fitted model. You can specify the following *component-options*:

COMMONAXES
specifies that the functional component plots use a common vertical axis except for contour plots. This enables you to visually judge relative effect size.

UNPACK | UNPACKPANEL
displays the component plots individually.

DIAGNOSTICS < (UNPACK | UNPACKPANEL) >
produces a summary panel of fit diagnostics that consists of the following:

- residuals versus the predicted values
- a histogram of the residuals
- a normal quantile plot of the residuals
- a residual-fit (RF) plot that consists of side-by-side quantile plots of the centered fit and the residuals
- response values versus the predicted values

You can request the five plots in this panel as individual plots by specifying the UNPACK suboption. The fit diagnostics panel is not produced for dependent variable with nonnormal distributions.

FIT < (NODATA | NOOBS) >
produces a plot of the predicted values against the variables that form the selected model. By default, a scatter plot of the input data is overlaid. You can suppress the scatter plot by specifying the NODATA | NOOBS option.

The plot is not produced if the number of variables in the selected model exceeds two. The plot is not produced for dependent variables with nonnormal distributions.

NONE
suppresses all plots.

SELECTION< (*selection-panel-options*) >
plots a panel of model fit criteria. The panel consists of two plots. The upper plot shows the progression of the model lack-of-fit criterion as the selection process proceeds. The lower plot shows the progression of the model validation criterion as the selection process proceeds. By default, the selection panel shows the progression for the backward selection process. You can specify the following *selection-panel-options*:

BACKWARDMODEL | BACKWARD
displays the progression of model fit criteria for the backward selection process.

FORWARDMODEL | FORWARD
displays the progression of model fit criteria for the forward selection process.

SEED=*number*
specifies an integer used to start the pseudorandom number generator for random cross validation and random partitioning of data for training, testing, and validation. If you do not specify a seed, or if you specify a value less than or equal to 0, the seed is generated from the time of day, which is read from the computer's clock.

SELFUZZ=*number*
SELECTFUZZ=*number*
sets the fuzzy comparison criterion when PROC ADAPTIVEREG examines candidate basis functions in forward and backward selection stages. The fuzzy comparison criterion is also used in stepwise selection for CLASS variables. A candidate is considered to be the best one only when its improvement is better than the current optimum with the extra amount *number*. By default, *number* is 10^4 times the machine epsilon. The default *number* is approximately 10^{-11} on most machines.

SINGULAR=*number*
EPSILON=*number*
sets the tolerance for testing singularity of the $\mathbf{X}'\mathbf{W}\mathbf{X}$ matrix that is formed from the design matrix $\mathbf{X}$. Roughly, the test requires that a pivot be at least this number times the original diagonal value. By default, *number* is 10^7 times the machine epsilon. The default *number* is approximately 10^{-9} on most machines.

TESTDATA=*SAS-data-set*

names a SAS data set that contains test data. This data set must contain all the variables specified in the MODEL statement. Furthermore, when a BY statement is used and the TESTDATA=data set contains any of the BY variables, then the TESTDATA= data set must also contain all the BY variables sorted in the order of the BY variables. In this case, only the test data for a specific BY group are used with the corresponding BY group in the analysis data. If the TESTDATA= data set contains none of the BY variables, then the entire TESTDATA = data set is used with each BY group of the analysis data.

If you specify a TESTDATA= data set, then you cannot also specify a PARTITION statement to reserve observations for testing.

NTHREADS=*n*

specifies the number of threads for analytic computations and overrides the SAS system option THREADS | NOTHREADS. If you do not specify the NTHREADS= option or if you specify NTHREADS=0, the number of threads is determined based on the data size and the number of CPUs on the host on which the analytic computations execute. If the specified number of threads is more than the number of actual CPUs, PROC ADAPTIVEREG by default sets the value to the number of actual CPUs.

VALDATA=*SAS-data-set*

names a SAS data set that contains validation data. This data set must contain all the variables specified in the MODEL statement. Furthermore, when a BY statement is used and the VALDATA= data set contains any of the BY variables, then the VALDATA= data set must also contain all the BY variables sorted in the order of the BY variables. In this case, only the validation data for a specific BY group are used with the corresponding BY group in the analysis data. If the VALDATA= data set contains none of the BY variables, then the entire VALDATA = data set is used with each BY group of the analysis data.

If you specify a VALDATA= data set, then you cannot also specify a PARTITION statement to reserve observations for validation.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC ADAPTIVEREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the ADAPTIVEREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / *options* > **;**

The CLASS statement names the classification *variables* to be used in the analysis. Typical CLASS variables are Treatment, Sex, Race, Group, and Replication. If the CLASS statement is used, it must appear before the MODEL statement.

Classification variables can be either character or numeric. Class levels are determined from the formatted values of the *variables*. Thus, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

You can specify the following *options* for classification *variables*:

DESCENDING
DESC
reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC ADAPTIVEREG orders the categories according to the ORDER= option and then reverses that order.

ORDER=*order-type*
specifies the sort order for the categories of categorical variables. This ordering determines which parameters in the model correspond to each level in the data. When the default ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. Table 28.2 shows how PROC ADAPTIVEREG interprets values of the ORDER= option.

Table 28.2 Sort Order for Categorical Variables

order-type	**Levels Sorted By**
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
FREQDATA	Order of descending frequency count, and within counts by order of appearance in the input data set when counts are tied
FREQFORMATTED	Order of descending frequency count, and within counts by formatted value (as above) when counts are tied
FREQINTERNAL	Order of descending frequency count, and within counts by unformatted value when counts are tied
INTERNAL	Unformatted value

For the FORMATTED and INTERNAL values, the sort order is machine-dependent. If you specify the ORDER= option in the MODEL statement and the ORDER= option in the CLASS statement, the former takes precedence.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a variable that provides frequencies for each observation in the DATA= data set. Specifically, if n is the value of the FREQ variable for a given observation, then that observation is used n times.

The analysis produced using a FREQ statement reflects the expanded number of observations. You can produce the same analysis without the FREQ statement by first creating a new data set that contains the expanded number of observations. For example, if the value of the FREQ variable is 5 for the first observation, the first five observations in the new data set are identical. Each observation in the old data set is replicated n_i times in the new data set, where n_i is the value of the FREQ variable for that observation.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

MODEL Statement

MODEL *dependent* < **(***options***)** >**=**< *effects* > < / *options* > **;**

MODEL *events/trials* **=** < *effects* > < / *options* > **;**

The MODEL statement names the response variable and the explanatory effects, including covariates, main effects, interactions, and nested effects; see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure," for more information. If you omit the explanatory effects, the procedure fits an intercept-only model. You must specify exactly one MODEL statement.

You can specify two forms of the MODEL statement. The first form, referred to as *single-trial* syntax, is applicable to binary, ordinal, and nominal response data. The second form, referred to as *events/trials* syntax, is restricted to binary response data. You use the *single-trial* syntax when each observation in the DATA= data set contains information about only a single trial, such as a single subject in an experiment. When each observation contains information about multiple binary response trials, such as the counts of the number of observed subjects and the number of subjects who respond, then you can use the *events/trials* syntax.

In the *events/trials* syntax, you specify two variables that contain count data for a binomial experiment. These two variables are separated by a slash. The value of the first variable, *events*, is the number of positive responses (or events). The value of the second variable, *trials*, is the number of trials. The values of both *events* and (*trials*–*events*) must be nonnegative and the value of *trials* must be positive for the response to be valid.

In the *single-trial* syntax, you specify one variable (on the left side of the equal sign) as the response variable. This variable can be character or numeric. You can specify variable options specific to the response variable immediately after the response variable with parentheses around them.

For both forms of the MODEL statement, explanatory *effects* follow the equal sign. Variables can be either continuous or classification variables. Classification variables can be character or numeric, and they must be declared in the CLASS statement. When an *effect* is a classification variable, the procedure inserts a set of coded columns into the design matrix instead of directly entering a single column that contains the values of the variable.

Table 28.3 summarizes the *options* available in the MODEL statement.

Table 28.3 MODEL Statement Options

Option	Description
Response Variable Options	
DESCENDING	Reverses the order of the response categories
EVENT=	Specifies the event category for the binary response
ORDER=	Specifies the sort order for the binary response
REFERENCE=	Specifies the reference category for the binary response
Statistical Modeling Options	
ADDITIVE	Requests an additive model
ALPHA	Controls the knot selection
CVMETHOD=	Specifies how subsets for cross validation are formed
DFPERBASIS	Specifies degrees of freedom per basis function
DIST=	Specifies the distribution family
FAST	Controls the fast-forward selection algorithm
FORWARDONLY	Requests that the backward selection process be skipped
KEEP=	Specifies effects to be included in the final model
LINEAR=	Specifies linear effects to be examined in model selection
LINK=	Specifies the link function
MAXBASIS=	Specifies the maximum number of basis functions allowed
MAXORDER=	Specifies the maximum order of interactions allowed
NOMISS	Requests removal of missing values from modeling
OFFSET=	Specifies an offset for the linear predictor
VARPENALTY=	Specifies the penalty for variable reentry

You can specify the following *options* in the MODEL statement.

Response Variable Options

Response variable options determine how the ADAPTIVEREG procedure models probabilities for binary data. You can specify the following response variable options by enclosing them in parentheses after the response variable.

DESCENDING
DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC ADAPTIVEREG orders the response categories according to the ORDER= option and then reverses that order.

EVENT='*category*' | **FIRST** | **LAST**

specifies the event category for the binary response model. PROC ADAPTIVEREG models the probability of the event category. You can specify one of the following values for this option:

'*category*' specifies the formatted value of the reference category.

FIRST designates the first ordered category as the event.

LAST designates the last ordered category as the event.

The default is EVENT=FIRST.

One of the most common sets of response levels is $\{0, 1\}$, with 1 representing the event for which the probability is to be modeled. Consider the example where Y takes the value 1 for event and 0 for nonevent, and X is the explanatory variable. To specify the value 1 as the event category, use the following MODEL statement:

```
model Y (event='1') = X;
```

ORDER=*order-type*

specifies the sort order for the categories of categorical variables. This ordering determines which parameters in the model correspond to each level in the data. When the default ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. Table 28.4 shows how PROC ADAPTIVEREG interprets values of the ORDER= option.

Table 28.4 Sort Order for Categorical Variables

order-type	**Levels Sorted By**
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
FREQDATA	Order of descending frequency count, and within counts by order of appearance in the input data set when counts are tied
FREQFORMATTED	Order of descending frequency count, and within counts by formatted value (as above) when counts are tied
FREQINTERNAL	Order of descending frequency count, and within counts by unformatted value when counts are tied
INTERNAL	Unformatted value

For the FORMATTED and INTERNAL values, the sort order is machine-dependent. If you specify the ORDER= option in the MODEL statement and the ORDER= option in the CLASS statement, the former takes precedence.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' **| FIRST | LAST**

REF='*category*' **| FIRST | LAST**

specifies the reference category for the binary or multinomial response model. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify one of the following values for this option:

'*category*' specifies the formatted value of the reference category.

FIRST designates the first ordered category as the reference.

LAST designates the last ordered category as the reference.

The default is REFERENCE=LAST.

Model Options

You can specify the following model *options*.

ADDITIVE

requests an additive model for which only main effects are included in the fitted model. If you do not specify the ADDITIVE option, PROC ADAPTIVEREG fits a model that has both main effects and two-way interaction terms.

ALPHA=*number*

specifies the parameter that controls the number of knots considered for each variable. Friedman (1991b) uses the following as the number of observations between interior knots:

$$-\frac{2}{5}\log_2\left[-\frac{\log(1-\alpha)}{pn_m}\right]$$

Friedman also uses the following as the number of observations between extreme knots and the corresponding variable boundary values,

$$3-\log_2\frac{\alpha}{p}$$

where p is the number of variables and n_m is the number of observations for which a parent basis $\mathrm{B}_m > 0$. The value of α should be greater than 0 and less than 1. The default is ALPHA=0.05.

CVMETHOD=RANDOM <(*n***)>**

CVMETHOD=INDEX (*variable***)**

specifies the method for subdividing the training data into *n* parts when you request *n*-fold cross validation when you do backward selection. CVMETHOD=RANDOM assigns each training observation randomly to one of the *n* parts. CVMETHOD=INDEX(*variable*) assigns observations to parts based on the formatted value of the named *variable*. This input data set variable is treated as a classification

variable, and the number of parts *n* is the number of distinct levels of this variable. By optionally naming this variable in a CLASS statement, you can use the ORDER= option in the CLASS statement to control how this variable is levelized.

The value of *n* defaults to 5 with CVMETHOD=RANDOM.

DFPERBASIS=*d*

DF=*d*

specifies the degrees of freedom (*d*) that are "charged" for each basis function that is used in the lack-of-fit function for backward selection. Larger values of *d* lead to fewer spline knots and thus smoother function estimates. The default is DFPERBASIS=2.

DIST=*distribution-id*

specifies the distribution family used in the model.

If you do not specify a *distribution-id*, the ADAPTIVEREG procedure defaults to the normal distribution for continuous response variables and to the binary distribution for classification or character variables, unless the *events/trial* syntax is used in the MODEL statement. If you choose the *events/trial* syntax, the ADAPTIVEREG procedure defaults to the binomial distribution.

Table 28.5 lists the values of the DIST= option and the corresponding default link functions. For generalized linear models with these distributions, you can find expressions for the log-likelihood functions in the section "Log-Likelihood Functions" on page 3659 in Chapter 51, "The GENMOD Procedure."

Table 28.5 Values of the DIST= Option

distribution-id	**Aliases**	**Distribution**	**Default Link Function**
BINOMIAL		Binomial	Logit
GAMMA	GAM, G	Gamma	Reciprocal
GAUSSIAN	NORMAL, N, NOR	Normal	Identity
IGAUSSIAN	IG	Inverse Gaussian	Inverse squared (power(–2))
NEGBIN	NB	Negative binomial	Log
POISSON	POI	Poisson	Log

FAST< (*fast-options***) >**

improves the speed of the modeling. Because of the computation complexity in the original multivariate adaptive regression splines algorithm, Friedman (1993) proposes modifications to improve the speed by tuning several parameters. See the section "Fast Algorithm" on page 1090 for more information about the improvement of the multivariate adaptive regression splines algorithm. You can specify the following *fast-options*:

BETA=*beta*

specifies the "aging" factor in the priority queue of candidate parent bases. Larger values of *beta* result in low-improvement parents rising fast into top list of candidates. The default value is BETA=1.

H=*h*

specifies the parameter that controls how often the improvement is recomputed for a parent basis $\mathbf{B}_m$ over all candidate variables. Larger values of *h* cause fewer computations of improvement. The default value is H=1.

K=*k*

specifies the number of top candidates in the priority queue of parent bases for selecting new bases. Larger values of *k* cause more parent bases to be considered. The default is to use half of eligible parent bases at every iteration.

FORWARDONLY

skips the backward selection step after forward selection is finished.

KEEP=*effects*

specifies a list of variables to be included in the final model.

LINEAR=*effects*

specifies a list of variables to be considered without nonparametric transformation. They should appear in the linear form if they are selected.

LINK=*keyword*

specifies the link function in the model. Not all link functions are available for all distribution families. The *keywords* and expressions for the associated link functions are shown in Table 28.6.

Table 28.6 Link Functions in MODEL Statement of the ADAPTIVEREG Procedure

keyword	**Alias**	**Link Function**	$g(\mu) = \eta =$
IDENTITY	ID	Identity	μ
LOG		Log	$\log(\mu)$
LOGIT		Logit	$\log(\mu/(1-\mu))$
POWERMINUS2		Power with exponent –2	$1/\mu^2$
PROBIT	NORMIT	Probit	$\Phi^{-1}(\mu)$
RECIPROCAL	INVERSE	Reciprocal	$1/\mu$

MAXBASIS=*number*

specifies the maximum number of basis functions ($M_{\max}$) that can be used in the final model. The default value is the larger value between 21 and one plus two times the number of nonintercept effects specified in the MODEL statement.

MAXORDER=*number*

specifies the maximum interaction levels for effects that could potentially enter the model. The default value is MAXORDER=2.

NOMISS

excludes all observations with missing values from the model fitting. By default, the ADAPTIVEREG procedure takes the missingness into account when an explanatory variable has missing values. For more information about how PROC ADAPTIVEREG handles missing values, see the section "Missing Values" on page 1091.

OFFSET=*variable*

specifies an offset for the linear predictor. An offset plays the role of a predictor whose coefficient is known to be 1. For example, you can use an offset in a Poisson model when counts have been obtained in time intervals of different lengths. With a log link function, you can model the counts as Poisson variables with the logarithm of the time interval as the offset variable. The offset variable cannot appear in the CLASS statement or elsewhere in the MODEL statement.

VARPENALTY= γ

specifies the incremental penalty γ for increasing the number of variables in the adaptive regression model. To discourage a model with too many variables, at each iteration of the forward selection the model improvement is reduced by a factor of $(1 - \gamma)$ for any new variable that is introduced.

For highly collinear designs, the VARPENALTY= option helps PROC ADAPTIVEREG produce models that are nearly equivalent in terms of residual sum of squares but have fewer independent variables. Friedman (1991b) suggests the following values for γ:

0.0	no penalty (default value)
0.05	moderate penalty
0.1	heavy penalty

The best value depends on the specific situation. Some experimenting with different values is usually required. You should use this option with care.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *keyword* < **(***keyword-options* **)** > < **=***name* > > . . .
< *keyword* < **(***keyword-options* **)** > < **=***name* > > **;**

The OUTPUT statement creates a new SAS data set to contain diagnostic measures that are calculated for the selected model. If you do not specify a *keyword*, then the only diagnostic included is the predicted response.

All the variables in the original data set are included by the new data set, along with variables created in the OUTPUT statement. These new variables contain the values of a variety of statistics and diagnostic measures that are calculated for each observation in the data set. If you specify a BY statement, then a variable _BY_ that indexes the BY groups is included. For each observation, the value of _BY_ is the index of the BY group to which this observation belongs.

If you have requested n-fold cross validation, then a variable _CVINDEX_ is included in the output data set. For each observation that is used for model training, the value of _CVINDEX_ is i if that observation is omitted in forming the ith subset of the training data. See the CVMETHOD= for additional details. The value of _CVINDEX_ is 0 for all observations in the input data set that are not used for model training.

If you have partitioned the input data by using a PARTITION statement, then a character variable _ROLE_ is included in the output data set. For each observation the value of _ROLE_ is as follows:

ROLE	Observation Role
TEST	Testing
TRAIN	Training
VALIDATE	Validation

If you want to create a permanent SAS data set, you must specify a two-level name. For more information about permanent SAS data sets, see *SAS Programmers Guide: Essentials*.

Details about the specifications in the OUTPUT statement follow.

keyword < **(***keyword-options***)** >< **=***name* >

specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. You can use the *keyword-options* to control which type of a particular statistic to compute for generalized linear models. You can specify the following *keyword-options* for associated statistics:

ILINK computes the prediction on the scale of the data $\hat{\mu} = g^{-1}(\hat{\eta})$.

RAW requests the raw residual value $r = y - \hat{\eta}$.

PEARSON requests the Pearson residual value $r = (y - \hat{\eta})/\sqrt{V(\hat{\mu})}$.

DEVIANCE requests the deviance residual value $r = \text{sign}(y - \hat{\mu})\sqrt{\hat{d}^2}$.

You can specify a *keyword* for each desired statistic (see the following list of *keywords*), followed optionally by an equal sign, and a variable to contain the statistic.

If you specify *keyword***=***name*, the new variable that contains the requested statistic has the specified name. If you omit the optional **=***name* after a *keyword*, then the new variable name is formed by default names.

You can specify the following *keywords* for the corresponding statistics:

PREDICTED | PRED | P requests predicted values. The default name is Pred.

RESIDUAL | RESID | R requests residuals, calculated as ACTUAL – PREDICTED. The default name is Resid.

OUT=*SAS-data-set*

specifies the name of the new data set to contain the diagnostic measures. If the OUT= option is omitted, the procedure uses the DATA*n* convention to name the output data set.

PARTITION Statement

PARTITION < *options* > **;**

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. Either you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations for each role.

An alternative to using a PARTITION statement is to provide a variable named _ROLE_ in the input data set to define roles of observations in the input data. If you specify a PARTITION statement, then the _ROLE_ variable is ignored if it is present in the input data set. If you do not specify a PARTITION statement and the input data do not contain a variable named _ROLE_, then all observations in the input data set are assigned to model training.

You can specify the following mutually exclusive *options*:

ROLEVAR=*variable* **(< TEST=‘***value***’ > < TRAIN=‘***value***’ > < VALIDATE=‘***value***’ >)**

ROLE=*variable* **(< TEST=‘***value***’ > < TRAIN=‘***value***’ > < VALIDATE=‘***value***’ >)**

names the *variable* in the input data set whose values are used to assign roles to each observation. The formatted values of this variable that are used to assign observations roles are specified in the TEST=, TRAIN=, and VALIDATE= suboptions. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboptions are assigned to training. If you specify a TESTDATA= data set in the PROC ADAPTIVEREG statement, then you cannot also specify the TEST= suboption in the PARTITION statement. If you specify a VALDATA= data set in the PROC ADAPTIVEREG statement, then you cannot also specify the VALIDATE= suboption in the PARTITION statement.

FRACTION(< TEST=*fraction* **> < VALIDATE=***fraction* **>)**

randomly assigns training and validation roles to the observations in the input data according to the proportions that are specified by the fraction values in the TEST= and VALIDATE= suboptions. If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. If you specify a TESTDATA= data set in the PROC ADAPTIVEREG statement, then you cannot also specify the TEST= suboption in the PARTITION statement. If you specify a VALDATA= data set in the PROC ADAPTIVEREG statement, then you cannot also specify the VALIDATE= suboption in the PARTITION statement.

SCORE Statement

SCORE < **DATA=***SAS-data-set*> < **OUT=***SAS-data-set*>
< *keyword* < **=***name*> >. . . < *keyword* < **=***name*> > **;**

The SCORE statement creates a new SAS data set to contain predicted values and optionally residuals for data in a new data set that you name. If you do not specify a DATA= data set, then the input data are scored. If you want to predict multiple data sets, you can specify multiple SCORE statements. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

When you specify a BY statement, the DATA= data set must either contain all the BY variables sorted in the order of the BY variables or contain none of the BY variables. If the DATA= data set contains all the BY variables, then the model that is selected for a given BY group is used to score just the matching observations in that data set. If the DATA= set contains none of the BY variables, then the entire data set is scored for each BY group.

All observations in the DATA= data set are retained in the output data set. All the variables in the input data set are included in the output data set, along with variables that contain predicted values and optionally residuals.

You can specify the following arguments in the SCORE statement:

DATA=*SAS data set*
names the data set to be scored. If you omit this option, then the input data set that is named in the DATA= option in the PROC ADAPTIVEREG statement is scored.

keyword < **=***name*>
specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. Specify one of the following *keyword* for each desired statistic, followed optionally by an equal sign, and a variable to contain the statistic.

If you specify *keyword***=***name*, the new variable that contains the requested statistic has the specified name. If you omit the optional **=***name* after a *keyword*, then the new variable name is formed by using a prefix of one or more characters that identify the statistic, followed by an underscore (_), followed by the dependent variable name.

You can specify the following *keywords*, which represent the statistics shown:

PREDICTED | PRED | P includes predicted values in the output data set. The prefix for the default name is Pred.

RESIDUAL | RESID | R includes residuals (which are calculated as ACTUAL – PREDICTED), in the output data set. The prefix for the default name is Resid.

OUT=*SAS data set*
specifies the name of the new output data set. By default, PROC ADAPTIVEREG uses the DATA*n* convention to name the new data set.

WEIGHT Statement

WEIGHT *variable* **;**

When you specify a WEIGHT statement, each observation in the input data set is weighted by the value of *variable*. The value of *variable* can be nonintegral. Observations that have a negative, zero, or missing value for the WEIGHT variable are not used in model fitting.

Details: ADAPTIVEREG Procedure

Fitting Algorithms

The multivariate adaptive regression splines algorithm (Friedman 1991b) is a predictive modeling algorithm that combines nonparametric variable transformations with a recursive partitioning scheme.

The algorithm originates with Smith (1982), who proposes a nonparametric method that applies the model selection method (stepwise regression) to a large number of truncated power spline functions, which are evaluated at different knot values. This method constructs spline functions and selects relevant knot values automatically with the model selection method. However, the method is applicable only to problems in low dimensions. For multiple variables, the number of tensor products between spline basis functions is too large to fit even a single model. The multivariate adaptive regression splines algorithm avoids this situation by using forward selection to build the model gradually instead of using the full set of tensor products of spline basis functions.

Like the recursive partitioning algorithm, which has “growing” and “pruning” steps, the multivariate adaptive regression splines algorithm contains two stages: forward selection and backward selection. During the forward selection process, bases are created from interactions between existing parent bases and nonparametric transformations of continuous or classification variables as candidate effects. After the model grows to a certain size, the backward selection process begins by deleting selected bases. The deletion continues until the null model is reached, and then an overall best model is chosen based on some goodness-of-fit criterion. The next three subsections give details about the selection process and methods of nonparametric transformation of variables. The fourth subsection describes how the multivariate adaptive regression splines algorithm is applied to fit generalized linear models. The fifth subsection describes the fast algorithm (Friedman 1993) for speeding up the fitting process.

Forward Selection

The forward selection process in the multivariate adaptive regression splines algorithm is as follows:

1. Initialize by setting $\mathbf{B}_0 = \mathbf{1}$ and $M = 1$.
2. Repeat the following steps until the maximum number of bases $M_{\max}$ has been reached or the model cannot be improved by any combination of $\mathbf{B}_m$, $\mathbf{v}$, and t.
   a) Set the lack-of-fit criterion $\mathrm{LOF}^* = \infty$.
   b) For each selected basis: $\mathbf{B}_m, m \in \{0, \ldots, M-1\}$ do the following for each variable $\mathbf{v}$ that $\mathbf{B}_m$ does not consist of $\mathbf{v} \notin \{\mathbf{v}(k,m) | 1 \le k \le K_m\}$
      i. For each knot value (or a subset of categories) t of $\mathbf{v} : t \in \{\mathbf{v}\}$, form a model with all currently selected bases $\sum_{i=0}^{M-1} \mathbf{B}_i$ and two new bases: $\mathbf{B}_m \mathbf{T}_1(\mathbf{v}, t)$ and $\mathbf{B}_m \mathbf{T}_2(\mathbf{v}, t)$.
      ii. Compute the lack-of-fit criterion for the new model LOF.
      iii. If $\mathrm{LOF} < \mathrm{LOF}^*$, then update $\mathrm{LOF}^* = \mathrm{LOF}$, $m^* = m$, $\mathbf{v}^* = \mathbf{v}$, and $t^* = t$.
   c) Update the model by adding two bases that improve the most $\mathbf{B}_{m^*} \mathbf{T}_1(\mathbf{v}^*, t^*)$ and $\mathbf{B}_{m^*} \mathbf{T}_2(\mathbf{v}^*, t^*)$.
   d) Set $M = M + 2$.

The essential part of each iteration is to search a combination of $\mathbf{B}_m$, $\mathbf{v}$, and t such that adding two corresponding bases most improve the model. The objective of the forward selection step is to build a model that overfits the data. The lack-of-fit criterion for linear models is usually the residual sum of squares (RSS).

Backward Selection

The backward selection process in the multivariate adaptive regression splines algorithm is as follows:

1. Initialize by setting the overall lack-of-fit criterion: $\mathrm{LOF}^* = \infty$.
2. Repeat the following steps until the null model is reached. The final model is the best one that is found during the backward deletion process.
   a) For a selected basis $\mathbf{B}_m, m \in \{1, \ldots, M\}$:
      i. Compute the lack-of-fit criterion, LOF, for a model that excludes $\mathbf{B}_m$.
      ii. If $\mathrm{LOF} < \mathrm{LOF}^*$, save the model as the best one. Let $m^* = m$.
      iii. Delete $\mathbf{B}_{m^*}$ from the current model.
   b) Set $M = M - 1$.

The objective of the backward selection is to "prune" back the overfitted model to find the best model that has good predictive performance. So the lack-of-fit criteria that characterize model loyalty to original data are not appropriate. Instead, the multivariate adaptive regression splines algorithm uses a quantity similar to the generalized cross validation criterion. See the section "Goodness-of-Fit Criteria" on page 1089 for more information.

Variable Transformations

The type of transformation depends on the variable type:

- For a continuous variable, the transformation is a linear truncated power spline,

$$\mathbf{T}_1(\mathbf{v}, t) = (v - t)_+ = \begin{cases} v - t, & \text{if } v > t \\ 0, & \text{otherwise} \end{cases}$$

$$\mathbf{T}_2(\mathbf{v}, t) = [-(v - t)]_+ = \begin{cases} 0, & \text{if } v > t \\ t - v, & \text{otherwise} \end{cases}$$

  where t is a knot value for variable $\mathbf{v}$ and v is an observed value for $\mathbf{v}$. Instead of examining every unique value of $\mathbf{v}$, a series of knot values with a minimum span are used by assuming the smoothness of the underlying function. Friedman (1991b) uses the following formula to determine a reasonable number of counts between knots (span size). For interior knots, the span size is determined by

$$-\frac{2}{5}\log_2\left[-\frac{\log(1-\alpha)}{pn_m}\right]$$

  For boundary knots, the span size is determined by

$$3 - \log_2\frac{\alpha}{p}$$

  where α is the parameter that controls the knot density, p is the number of variables, and n_m is the number of observations that a parent basis $\mathbf{B}_m > 0$.

- For a classification variable, the transformation is an indicator function,

$$\mathbf{T}_1(\mathbf{v}, t) = \begin{cases} 1, & \text{if } v \in \{c_1, \ldots, c_t\} \\ 0, & \text{otherwise} \end{cases}$$

$$\mathbf{T}_2(\mathbf{v}, t) = \begin{cases} 0, & \text{if } v \in \{c_1, \ldots, c_t\} \\ 1, & \text{otherwise} \end{cases}$$

  where $\{c_1, \ldots, c_t\}$ is a subset of all categories of variable $\mathbf{v}$. The smoothing is applied to categorical variables by assuming that subsets of categories tend to have similar properties, analogous to the assumption that a local neighborhood has close predictions for continuous variables.

  If a categorical variable has k distinct categories, then there are a total of $2^{k-1} - 1$ possible subsets to consider. The computation cost is equal to all-subsets selection in regression, which is expensive for large k values. The multivariate adaptive regression splines algorithm use the stepwise selection method to select categories to form the subset $\{c_1, \ldots, c_t\}$. The method is still greedy, but it reduces computation and still yields reasonable final models.

Goodness-of-Fit Criteria

Like other nonparametric regression procedures, the multivariate adaptive regression splines algorithm can yield complicated models that involve high-order interactions in which many knot values or subsets are considered. Besides the basis functions, both the forward selection and backward selection processes are also highly nonlinear. Because of the trade-off between bias and variance, the complicated models that contain many parameters tend to have low bias but high variance. To select models that achieve good prediction performance, Craven and Wahba (1979) propose the widely used generalized cross validation criterion (GCV),

$$\mathrm{GCV} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}_i}{1 - \mathrm{trace}(\mathbf{S})/n}\right)^2 = \frac{\mathrm{RSS}}{n(1 - \mathrm{trace}(\mathbf{S})/n)^2}$$

where y is the response, $\hat{f}$ is an estimate of the underlying smooth function, and $\mathbf{S}$ is the smoothing matrix such that $\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}$. The effective degrees of freedom for the smoothing spline can be defined as $\mathrm{trace}(\mathbf{S})$. In the multivariate adaptive regression splines algorithm, Friedman (1991b) uses a similar quantity as the lack-of-fit criterion,

$$\mathrm{LOF} = \frac{\mathrm{RSS}}{n(1 - (M + d(M-1)/2)/n)^2}$$

where d is the degrees-of-freedom cost for each nonlinear basis function and M is total number of linearly independent bases in the model. Because any candidate model that is evaluated at each step of the multivariate adaptive regression splines algorithm is a linear model, M is actually the trace of the hat matrix. The only difference between the GCV criterion and the LOF criterion is the extra term $d(M-1)$. The corresponding effective degrees of freedom is defined as $M + d(M-1)/2$. The quantity d takes into account the extra nonlinearity in forming new bases, and it operates as a smoothing parameter. Larger values of d tend to result in smoother function estimates. Based on many practical experiments and some theoretic work (Owen 1991), Friedman suggests that the value of d is typically in the range of $[2, 4]$. For data that have complicated structures, the value of d could be much larger.

Alternatively, you can use the cross validation as the goodness-of-fit criterion or use a separate validation data set to select models and a separate testing data set to evaluate selected models.

Generalized Linear Models

Friedman (1991b) applies the multivariate adaptive regression splines algorithm to a logistic model by using the squared error loss between the response and inversely linked values in the goodness-of-fit criterion:

$$\sum_{i=1}^{n}\left(y_i - \frac{1}{1 + \exp(\mathbf{x}'\boldsymbol{\beta})}\right)^2$$

When a final model is obtained, the ordinary logistic model is fitted on selected bases. Some realizations of the multivariate adaptive regression splines algorithm ignore the distributional properties and derive model bases that are based on the least squares criterion. The reason to ignore the distributional properties or use least squares approximations is that examining the lack-of-fit criterion for each combination of $\mathbf{B}_m$, $\mathbf{v}$, and t is computationally formidable, because one generalized linear model fit involves multiple steps of weighted least squares. The ADAPTIVEREG procedure extends the multivariate adaptive regression splines algorithm to generalized linear models as suggested by Buja et al. (1991).

In the forward selection process, the ADAPTIVEREG procedure extends the algorithm in the following way. Suppose there are $(2k + 1)$ bases after iteration k Then a generalized linear model is fitted against the data by using the selected bases. Then the weighted least squares method uses the working weights and working response in the last step of the iterative reweighted least squares algorithm as the weight and response for selecting new bases in iteration $(k + 1)$. Then the residual chi-square statistic is used to select two new bases. This is similar to the forward selection scheme that the LOGISTIC procedure uses. For more information about the score chi-square statistic, see the section "Testing Individual Effects Not in the Model" on page 6040 in Chapter 79, "The LOGISTIC Procedure."

In the backward selection process, the ADAPTIVEREG procedure extends the algorithm in the following way. Suppose there are M bases in the selected model. The Wald chi-square statistic is used to determine which basis to delete. After one basis is selected for deletion, a generalized linear model is refitted with the remaining bases. This is similar to the backward deletion scheme that the LOGISTIC procedure uses. For more information about the Wald chi-square statistic, see the section "Testing Linear Hypotheses about the Regression Coefficients" on page 6059 in Chapter 79, "The LOGISTIC Procedure."

Accordingly, the lack-of-fit criterion in the forward selection for generalized linear models is the score chi-square statistic. For the lack-of-fit criterion in the backward selection process for generalized linear models, the residual sum of squares term is replaced by the model deviance.

Fast Algorithm

The original multivariate adaptive regression splines algorithm is computationally expensive. To improve the computation speed, Friedman (1993) proposes the fast algorithm. The essential idea of the fast algorithm is to reduce the number of combinations of $\mathbf{B}$, $\mathbf{v}$, and t that are examined at each step of forward selection.

Suppose there are $(2k + 1)$ bases that are formed after iteration k, where a parent basis $\mathbf{B}_m$ is selected to construct two new bases. Consider a queue with bases as its elements. At the top of the queue are the selected parent $\mathbf{B}_m$ and two newly constructed bases, $\mathbf{B}_{2k}$ and $\mathbf{B}_{2k+1}$. The rest of the queue is sorted based on the minimum lack-of-fit criterion for each basis:

$$J(\mathbf{B}_i) = \min_{\substack{\text{for all eligible } \mathbf{v} \\ \text{for all knot } t}} \mathrm{LOF}(\mathbf{v}, t|\mathbf{B}_i),\ i = 1, \ldots, 2k - 1$$

When k is not small, there are a relatively large number of bases in the model, and adding more bases is unlikely to dramatically improve the fit. Thus the ranking of the bases in the priority queue is not likely to change much during adjacent iterations. So the candidate parent bases can be restricted to the top K ones in the queue for iteration $(k + 1)$ After iteration k, the top bases have new $J(\mathbf{B}_i)$ values, whereas the values of the bottom bases are unchanged. The queue is reordered based on $J(\mathbf{B}_i)$ values. This corresponds to the K= option value for the FAST option in the MODEL statement.

To avoid losing the candidate bases that are ranked at the bottom of the queue and to allow them to rise back to the top, a natural "aging" factor is introduced into each basis. This is accomplished by defining the priority for each basis function to be

$$P(\mathbf{B}_i) = R(\mathbf{B}_i) + \beta(k_c - k_r)$$

where $R(\mathbf{B}_i)$ is the rank of ith basis in the queue, k_c is the current iteration number, and k_r is the number of the iteration where the $J(\mathbf{B}_i)$ value was last computed. The top K candidate bases are then sorted again based on this priority. Large β values cause bases that have low improvement during previous iterations to rise faster to the top of the list. This corresponds to the BETA= value for the FAST option in the MODEL statement.

For a candidate basis in the top of the priority queue, the minimum lack-of-fit criterion $J(\mathbf{B}_i)$ is recomputed for all eligible variables **v** for iteration $(k + 1)$. An optimal variable is likely to be the same as the one that was found during the previous iteration. So the fast multivariate adaptive regression splines algorithm introduces another factor H to save the computation cost. The factor specifies how often $J(\mathbf{B}_i)$ should be recomputed for all eligible variables. If $H = 1$, then optimization over all variables is done at each iteration when a parent basis is considered. If $H = 5$, the complete optimization is done after five iterations. For an iteration count less than the specified H, the optimization is done only for the optimal variable found in the last complete optimization. The only exceptions are the top three candidates, $\mathbf{B}_{2k-1}$ (which is the parent basis $\mathbf{B}_m$ used to construct two new bases) and two new ones, $\mathbf{B}_{2k}$ and $\mathbf{B}_{2k+1}$. The complete optimization for them is performed at each iteration. This corresponds to the H= option value for the FAST option in the MODEL statement.

Missing Values

When fitting a model, the ADAPTIVEREG procedure excludes observations that have missing values for the response variable, weight variable, or frequency variable. It also excludes observations with invalid response, weight, or frequency values. For observations that have valid response, weight, and frequency values but missing predictor values, the ADAPTIVEREG procedure can either include them in model fitting or exclude them.

By default, observations with missing values in the predictor variables are included in the model fitting. Suppose a variable **v** contains missing values. The ADAPTIVEREG procedure automatically forms two candidate bases, $\mathbf{B}_m$ and $\mathbf{B}_{m+1}$, in the forward selection step when variable **v** is considered. When v is missing, $\mathbf{B}_{m+1} = I(v \text{ is missing})$. When v is not missing, $\mathbf{B}_m = I(v \text{ is not missing})$. $I(\cdot)$ is a scalar-valued indicator function that returns a 1 when the argument is true and a 0 when the argument is false.

If the transformation of **v** with a parent basis $\mathbf{B}_i$ and a knot (or a subset) t turns out to be the best one during this iteration, then two more bases are added to the model:

$$\mathbf{B}_{m+2} = \mathbf{B}_i \mathbf{B}_m \mathbf{T}_1(v - t)$$

$$\mathbf{B}_{m+3} = \mathbf{B}_i \mathbf{B}_{m+1} \mathbf{T}_2(v - t)$$

The indicator function does not contribute to the interaction order of the constructed bases. This approach assumes that the missingness in the training data is representative of missingness in future data to be predicted.

Alternatively, you can specify the NOMISS option in the MODEL statement to exclude from the model fitting all observations that have missing values in the predictor variables.

ANOVA Decomposition

The model that is produced by the multivariate adaptive regression splines algorithm can be formed as

$$\begin{aligned}\hat{f}(\mathbf{x}) &= \beta_0 + \sum_{m=1}^{M} \beta_m \mathbf{B}_m \\ &= \beta_0 + \sum_{m=1}^{M} \beta_m \prod_{k=1}^{K_m} \mathbf{T}_m(\mathbf{x}_{k,m}, t_{k,m})\end{aligned}$$

Here $\hat{f}$ is the nonparametric estimate of the response variable in linear models and of the linked response variable in generalized linear models. M is the number of nonconstant bases. For each formed basis, K_m is the order of interaction, $\mathbf{T}_m$ is the variable transformation function that depends on the variable type, $\mathbf{x}_{k,m}$ is variable for the kth component of the basis, and $t_{k,m}$ is the corresponding knot value or subset categories for the variable.

The function estimate can be recast into the form

$$\hat{f}(\mathbf{x}) = \beta_0 + \sum_{i:K_m=1} f_i(\mathbf{x}_i) + \sum_{i,j:K_m=2} f_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \sum_{i,j,k:K_m=3} f_{ijk}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k) + \cdots$$

where f_i represents the sum of bases that involve a single variable $\mathbf{x}_i$, f_{ij} represents the sum of bases that involve two-way interactions between transformations of two variables, and so on. The univariate function f_i is a linear regression spline for variable $\mathbf{x}_i$, which represent the univariate contribution of $\mathbf{x}_i$ to the model. Let

$$f_{ij}^* = f_i(\mathbf{x}_i) + f_j(\mathbf{x}_j) + f_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

Then this bivariate function is a tensor product regression spline that represents the joint contribution by both $\mathbf{x}_i$ and $\mathbf{x}_j$. Multivariate functions can be formed similarly if higher-order interaction terms are present in the model. Because of its similarity to the analysis of variance for contingency tables, this is referred as the ANOVA decomposition of the multivariate adaptive regression splines model.

Computational Resources

The multivariate adaptive regression splines algorithm is computationally intensive and requires a significant amount of memory for large data sets. However, the core algorithm is fairly scalable, so you might expect performance improvement if you use multicore machines. You can even further improve the fitting speed by carefully tuning the parameters of the FAST option in the MODEL statement.

A general formula does not exist for predicting amount of memory that is required for PROC ADAPTIVEREG. The procedure uses logical utility files to store values that are associated with observations. If sufficient random access memory (RAM) is available, the utility files reside in RAM to allow fast access. Otherwise, the utility files are stored on hard drives, which have slower read/write speed.

Because of the model selection nature, the multivariate adaptive regression splines algorithm essentially fits a large number of candidate models with different sets of basis functions. The original prototype requires computation that is proportional to $pNM_{\max}^4$. The implemented algorithm takes advantage of the special structure of linear truncated power functions to reduce the computation to be proportional to $pNM_{\max}^3$. With the fast algorithm, the computations can be reduced even further.

To provide a feel for how the number of variables and the number of observations affect the fitting performance, a series of simulations are carried out on a server with a 12-way 2.6GHz AMD Opteron processor and 32GB of RAM. Data sets are created in different sizes with the number of variables ranging from 10 to 50 and the number of observations ranging from 100 to 5,000. For each data set, the true model is the same as the one used in Example 28.1. At each data size, the experiment is repeated three times to measure minimum running times and corresponding memory consumption. PROC ADAPTIVEREG sets the maximum number of basis function to 50 and uses all other default options. Figure 28.11 displays the results of the simulations.

Figure 28.11 Time and Memory Used

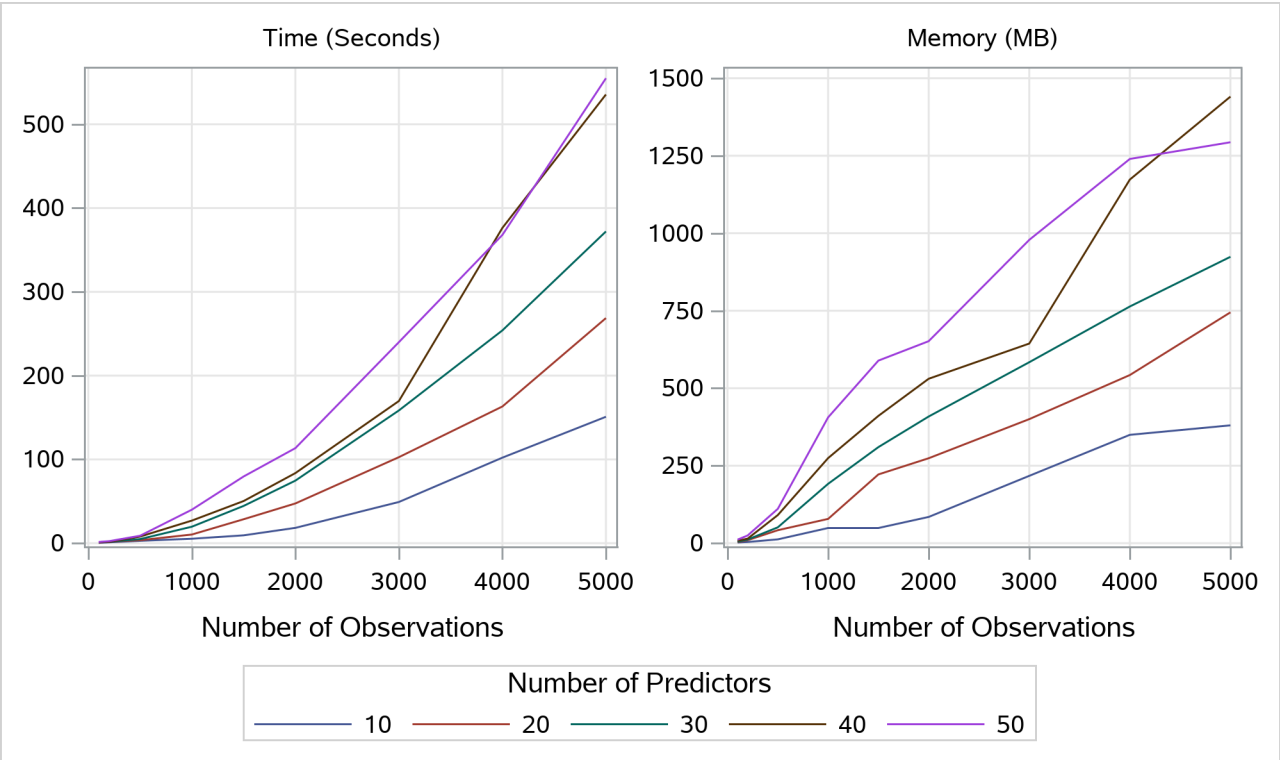

The graphs in Figure 28.11 show that computational times grow with respect to the number of observations at a speed that is slightly faster than linear. The growth of the memory consumption is close to linear. Also, the increment with respect to number of variables is approximately in fixed ratios.

To show how PROC ADAPTIVEREG scales as the number of CPUs grows, another series of experiments is performed with the following settings. SAS DATA steps use the same mechanism as in previous simulations to generate data sets. The number of observations range from 100 to 5,000, and the number of variables is 10. PROC ADAPTIVEREG fits models to these data sets with the maximum number of basis functions set to 50. Each fitting uses four threading settings with 1, 2, 4, and 8 CPUs.

Figure 28.12 displays the simulation results. The computation times scale well with the number of CPUs. The more CPUs you have, the less time you need to fit a model by using PROC ADAPTIVEREG.

Figure 28.12 Time Used with Different Number of Threads

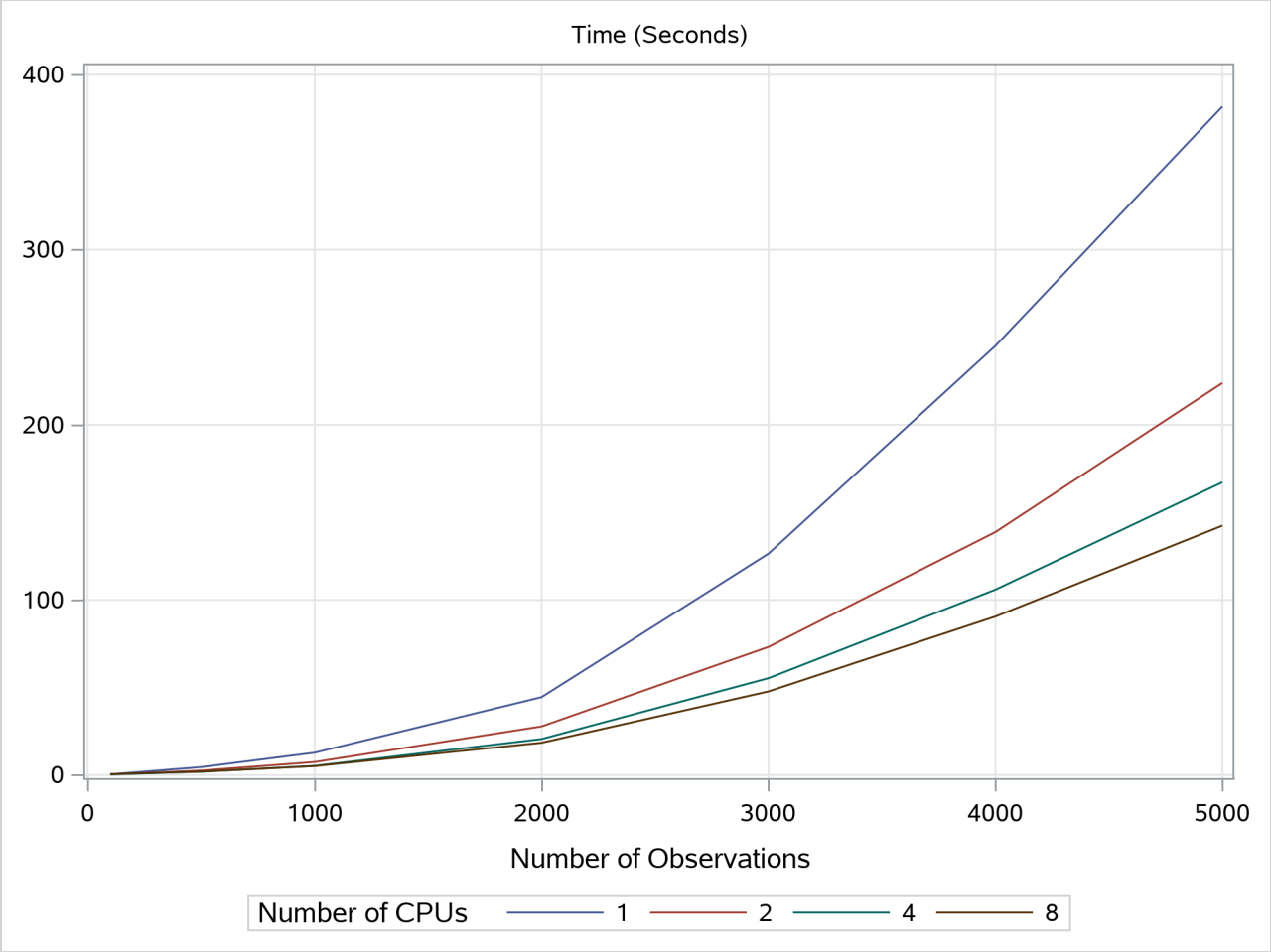

ODS Table Names

PROC ADAPTIVEREG assigns a name to each table that it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 28.7. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 28.7 ODS Tables Produced by PROC ADAPTIVEREG

ODS Table Name	Description	Statement	Option
ANOVA	ANOVA functional decomposition	PROC ADAPTIVEREG	Default
Bases	Bases transformation information	PROC ADAPTIVEREG	DETAILS=BASES
BWDParams	Parameter estimates after backward selection	PROC ADAPTIVEREG	Default

Table 28.7 *continued*

ODS Table Name	Description	Statement	Option
ClassInfo	Classification variable levels information	CLASS	Default
FitControls	Fit control parameters	PROC ADAPTIVEREG	Default
FitStatistics	Model fit statistics	PROC ADAPTIVEREG	Default
FWDParams	Parameter estimates after forward selection	PROC ADAPTIVEREG	DETAILS=FWDPARAMS
FWDSummary	Forward selection summary	PROC ADAPTIVEREG	DETAILS=FWDSUMMARY
ModelInfo	Model information	PROC ADAPTIVEREG	Default
NObs	Number of observations	PROC ADAPTIVEREG	Default
ResponseProfile	Response categories and category modeled	MODEL	DIST=BINARY
SelectionSummary	Backward selection summary	PROC ADAPTIVEREG	DETAILS=BWDSUMMARY
VarImp	Variable importance information	PROC ADAPTIVEREG	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You must also specify the PLOTS= option in the PROC ADAPTIVEREG statement.

PROC ADAPTIVEREG assigns a name to each graph that it creates using ODS. You can use these names to refer to the graphs when using ODS. The names are listed in Table 28.8.

Table 28.8 Graphs Produced by PROC ADAPTIVEREG

ODS Graph Name	Plot Description	PLOTS Option
BoxPlot	Box plot of the response at each level of one categorical predictor, overlaid with a plot of predicted values	FIT
ComponentPanel	Panel of partial prediction curves for components that contain up to two predictors	COMPONENTS
ContourPlot	Contour plot of the fitted surface by two continuous predictors overlaid on scatter plot of data	FIT
DiagnosticsPanel	Panel of fit diagnostics	DIAGNOSTICS
FitPlot	Plot of fitted values by single continuous predictor (or with one categorical variable) overlaid on scatter plot of data	FIT
IntPlot	Plot of fitted values by two categorical predictors overlaid on scatter plot of data	FIT
ObservedByPredicted	Dependent variable versus fitted values	DIAGNOSTICS(UNPACK)
QQPlot	Normal quantile plot of residuals	DIAGNOSTICS(UNPACK)
ResidualByPredicted	Residuals versus fitted values	DIAGNOSTICS(UNPACK)
ResidualHistogram	Histogram of fit residuals	DIAGNOSTICS(UNPACK)
RFPlot	Side-by-side plots of quantiles of centered fit and residuals	DIAGNOSTICS(UNPACK)
SelectionPlot	Model fit criteria by step	SELECTION

Examples: ADAPTIVEREG Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 28.1: Surface Fitting with Many Noisy Variables

(View the complete code for this example (adptex1.sas) in the example repository.)

This example shows how you can use PROC ADAPTIVEREG to fit a surface model from a data set that contains many nuisance variables.

Consider a simulated data set that contains a response variable and 10 continuous predictors. Each continuous predictor is sampled independently from the uniform distribution $U(0, 1)$. The true model is formed by x_1 and x_2:

$$y = \frac{40 \exp\left(8\left((x_1 - 0.5)^2 + (x_2 - 0.5)^2\right)\right)}{\exp\left(8\left((x_1 - 0.2)^2 + (x_2 - 0.7)^2\right)\right) + \exp\left(8\left((x_1 - 0.7)^2 + (x_2 - 0.2)^2\right)\right)}$$

The values of the response variable are generated by adding errors from the standard normal distribution $N(0, 1)$ to the true model. The generating mechanism is adapted from Gu et al. (1990). There are 400 generated observations in all. The following statements create an artificial data set:

```
data artificial;
   drop i;
   array X{10};
   do i=1 to 400;
      do j=1 to 10;
         X{j} = ranuni(1);
      end;
      Y = 40*exp(8*((X1-0.5)**2+(X2-0.5)**2))/
            (exp(8*((X1-0.2)**2+(X2-0.7)**2))+
             exp(8*((X1-0.7)**2+(X2-0.2)**2)))+rannor(1);
      output;
   end;
run;
```

The standard deviation for the response without noise is 3, whereas the standard deviation for the error term is 1. So the response variable Y has a signal-to-noise ratio of 3. When eight more variables are introduced, it is harder to search for the true model because of the extra variability that the nuisance variables create. The objective is to fit a nonparametric surface model that can well approximate the true model without experiencing much interference from the nuisance variables.

The following statements invoke the ADAPTIVEREG procedure to fit the model:

```
ods graphics on;

proc adaptivereg data=artificial plots=fit;
   model y=x1-x10;
run;
```

The PLOTS=FIT option in the PROC ADAPTIVEREG statement requests a fit plot. PROC ADAPTIVEREG might not produce the fit plot because the number of predictors in the final model is unknown. If the final model has no more than two variables, then the fit can be graphically presented.

PROC ADAPTIVEREG selects the two variables that form the true model (X1, X2) and does not include other nuisance variables. The "Fit Statistics" table (Output 28.1.1) lists summary statistics of the fitted surface model. The model has 27 effective degrees of freedom and 14 basis functions formed by X1 or X2 or both. The fit statistics suggest that this is a reasonable fit.

Output 28.1.1 Fit Statistics

The ADAPTIVEREG Procedure

Fit Statistics	
GCV	1.55656
GCV R-Square	0.86166
Effective Degrees of Freedom	27
R-Square	0.87910
Adjusted R-Square	0.87503
Mean Square Error	1.40260
Average Square Error	1.35351

Output 28.1.2 lists both parameter estimates and construction components (parent basis function, new variable, and optimal knot for the new variable) for the basis functions.

Output 28.1.2 Parameter Estimates

Regression Spline Model after Backward Selection				
Name	**Coefficient**	**Parent**	**Variable**	**Knot**
Basis0	12.3031		Intercept	
Basis1	13.1804	Basis0	X1	0.05982
Basis3	-23.4892	Basis0	X2	0.1387
Basis4	-171.03	Basis0	X2	0.1387
Basis5	-86.1867	Basis3	X1	0.6333
Basis7	-436.86	Basis4	X1	0.5488
Basis8	397.18	Basis4	X1	0.5488
Basis9	11.4682	Basis1	X2	0.6755
Basis10	-19.1796	Basis1	X2	0.6755
Basis13	126.84	Basis11	X1	0.6018
Basis14	40.8134	Basis11	X1	0.6018
Basis15	22.2884	Basis0	X1	0.7170
Basis17	-53.8746	Basis12	X1	0.2269
Basis19	598.89	Basis4	X1	0.2558

Output 28.1.3 shows all the ANOVA functional components that form the final model. The function estimate consists of two basis functions for each of X1 and X2 and nine bivariate functions of both variables. Because the true model contains the interaction between X1 and X2, PROC ADAPTIVEREG automatically selects many interaction terms.

Output 28.1.3 ANOVA Decomposition

ANOVA Decomposition				
			Change If Omitted	
Functional Component	**Number of Bases**	**DF**	**Lack of Fit**	**GCV**
X1	2	4	405.18	1.1075
X2	2	4	947.87	2.6348
X2 X1	9	18	2583.21	6.6187

To compute predictions for the contour plot of the fitted model, you can use the SCORE statement. The following statements produce the graph that shows both the true model and the fitted model:

```
data score;
   do X1=0 to 1 by 0.01;
      do X2=0 to 1 by 0.01;
         Y=40*exp(8*((X1-0.5)**2+(X2-0.5)**2))/
              (exp(8*((X1-0.2)**2+(X2-0.7)**2))+
               exp(8*((X1-0.7)**2+(X2-0.2)**2)));
         output;
      end;
   end;
```

```
run;

proc adaptivereg data=artificial;
   model y=x1-x10;
   score data=score out=scoreout;
run;

%let off0 = offsetmin=0 offsetmax=0;
%let off0 = xaxisopts=(&off0) yaxisopts=(&off0);
%let eopt = location=outside valign=top textattrs=graphlabeltext;
proc template;
   define statgraph surfaces;
      begingraph / designheight=360px;
         layout lattice/columns=2;
            layout overlay / &off0;
               entry "True Model" / &eopt;
               contourplotparm z=y y=x2 x=x1;
            endlayout;
            layout overlay / &off0;
               entry "Fitted Model" / &eopt;
               contourplotparm z=pred y=x2 x=x1;
            endlayout;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=scoreout template=surfaces;
run;
```

Output 28.1.4 displays surfaces for both the true model and the fitted model. The fitted model approximates the underlying true model well.

Output 28.1.4 True Model and Fitted Model

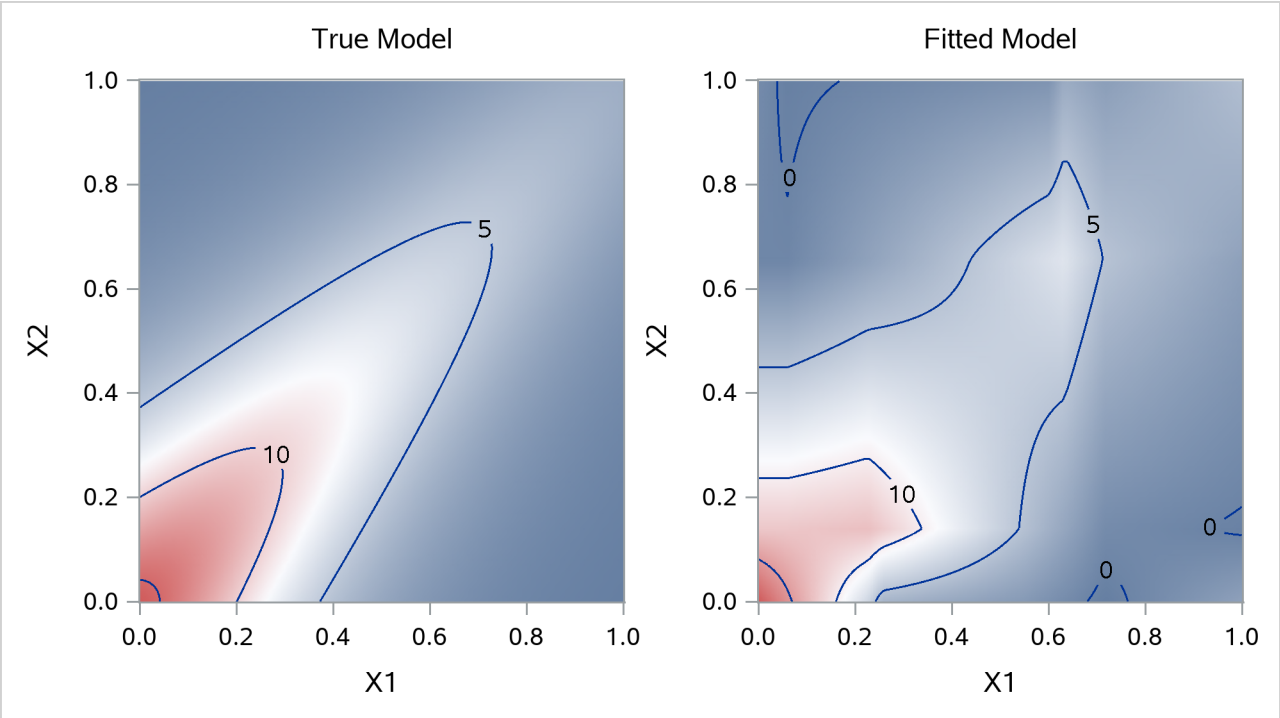

For high-dimensional data sets with complex underlying data-generating mechanisms, many different models can almost equally approximate the true mechanisms. Because of the sequential nature of the selection mechanism, any change in intermediate steps due to perturbations from local structures might yield completely different models. Therefore, PROC ADAPTIVEREG might find models that contain noisy variables. For example, if you change the random number seed in generating the data (as in the following statements), PROC ADAPTIVEREG might return different models with more variables. You can use the information from the variable importance table (Output 28.1.5) to aid further analysis.

```
data artificial;
   drop i;
   array x{10};
   do i=1 to 400;
      do j=1 to 10;
         x{j} = ranuni(12345);
      end;
      y = 40*exp(8*((x1-0.5)**2+(x2-0.5)**2))/
            (exp(8*((x1-0.2)**2+(x2-0.7)**2))+
             exp(8*((x1-0.7)**2+(x2-0.2)**2)))+rannor(1);
      output;
   end;
run;

proc adaptivereg data=artificial;
   model y=x1-x10;
run;
```

Output 28.1.5 shows that the variables X1 and X2 are two dominating factors for predicting the response, whereas the relative importance of the variable X8 compared to the other two is negligible. You might want

to remove the variable if you fit a new model.

Output 28.1.5 Variable Importance

The ADAPTIVEREG Procedure

Variable Importance		
Variable	Number of Bases	Importance
x1	13	100.00
x2	12	98.58
x8	2	0.22

Example 28.2: Fitting Data with Mixture Structures

(View the complete code for this example (adptex2.sas) in the example repository.)

This example shows how you can use PROC ADAPTIVEREG to fit a model from a data set that contains mixture structures. It also demonstrates how to use the CLASS statement.

Consider a simulated data set that contains a response variable and two predictors, one continuous and the other categorical. The continuous predictor is sampled from the uniform distribution $U(0,1)$, and the classification variable is sampled from $U(0,3)$ and then rounded to integers. The response variable is constructed from three different models that depend on the CLASS variable levels, with error sampled from the standard normal distribution.

$$y = \begin{cases} \exp(5(x-0.3)^2), & \text{if } c = 0 \\ \log(x - x^2), & \text{if } c = 1 \\ 7x, & \text{if } c = 2 \end{cases}$$

The following statements create the artificial data set Mixture:

```
data Mixture;
   drop i;
   do i=1 to 1000;
      X1 = ranuni(1);
      C1 = int(3*ranuni(1));
      if C1=0 then Y=exp(5*(X1-0.3)**2)+rannor(1);
      else if C1=1 then Y=log(X1*(1-X1))+rannor(1);
      else Y=7*X1+rannor(1);
      output;
   end;
run;
```

The standard deviation for the response without noise is 3.14. So the response variable Y in the data set Mixture has a signal-to-noise ratio of 3.14. With a classification variable and a continuous variable in the data set, the objective is to fit a nonparametric model that can reveal the underlying three different data-generating processes. The following statements use the ADAPTIVEREG procedure to fit the data:

```
ods graphics on;
proc adaptivereg data=Mixture plots=fit;
   class c1;
   model y=c1 x1;
run;
```

Because the data contain two explanatory variables, graphical presentation of the fitted model is possible. The PLOTS=FIT option in the PROC ADAPTIVEREG statement requests the fit plot. The CLASS statement specifies that C1 is a classification variable.

Output 28.2.1 displays the parameter estimates for the 13 selected basis functions after backward selection. For Basis1, the coefficient estimate is -4.3871. It is constructed from the intercept and the classification variable C1 at levels 0 and 1.

Output 28.2.1 Parameter Estimates

The ADAPTIVEREG Procedure

Regression Spline Model after Backward Selection					
Name	**Coefficient**	**Parent**	**Variable**	**Knot**	**Levels**
Basis0	5.3829		Intercept		
Basis1	-4.3871	Basis0	C1		1 0
Basis3	32.7761	Basis0	C1		1
Basis5	20.2859	Basis4	X1	0.7665	
Basis7	-11.4183	Basis2	X1	0.7665	
Basis8	-7.0758	Basis2	X1	0.7665	
Basis9	58.4911	Basis3	X1	0.5531	
Basis10	-71.6388	Basis3	X1	0.5531	
Basis11	-69.0764	Basis3	X1	0.04580	
Basis13	-119.71	Basis3	X1	0.9526	
Basis15	66.5733	Basis1	X1	0.9499	
Basis17	6.6681	Basis1	X1	0.5143	
Basis19	-185.21	Basis1	X1	0.9890	

Output 28.2.2 displays the fitted linear splines overlaid with the original data. PROC ADAPTIVEREG captures the three underlying data-generating processes. For observations with C1 at level 0, the shape of the fitted splines is quite similar to the exponential function. For observations with C1 at level 1, the shape of the fitted spline suggests a symmetric function along X1 with a symmetry point approximately equal to 0.5. The function at each side of the symmetry point is analogous to the logarithmic transformation. For the rest of the observations with C1 at level 2, PROC ADAPTIVEREG suggests a strict linear model. The fitted model is very close to the true model. PROC ADAPTIVEREG fits the model in an automatic and adaptive way, except that it needs the CLASS statement to name the classification variable.

Output 28.2.2 Raw Data and Fitted Model

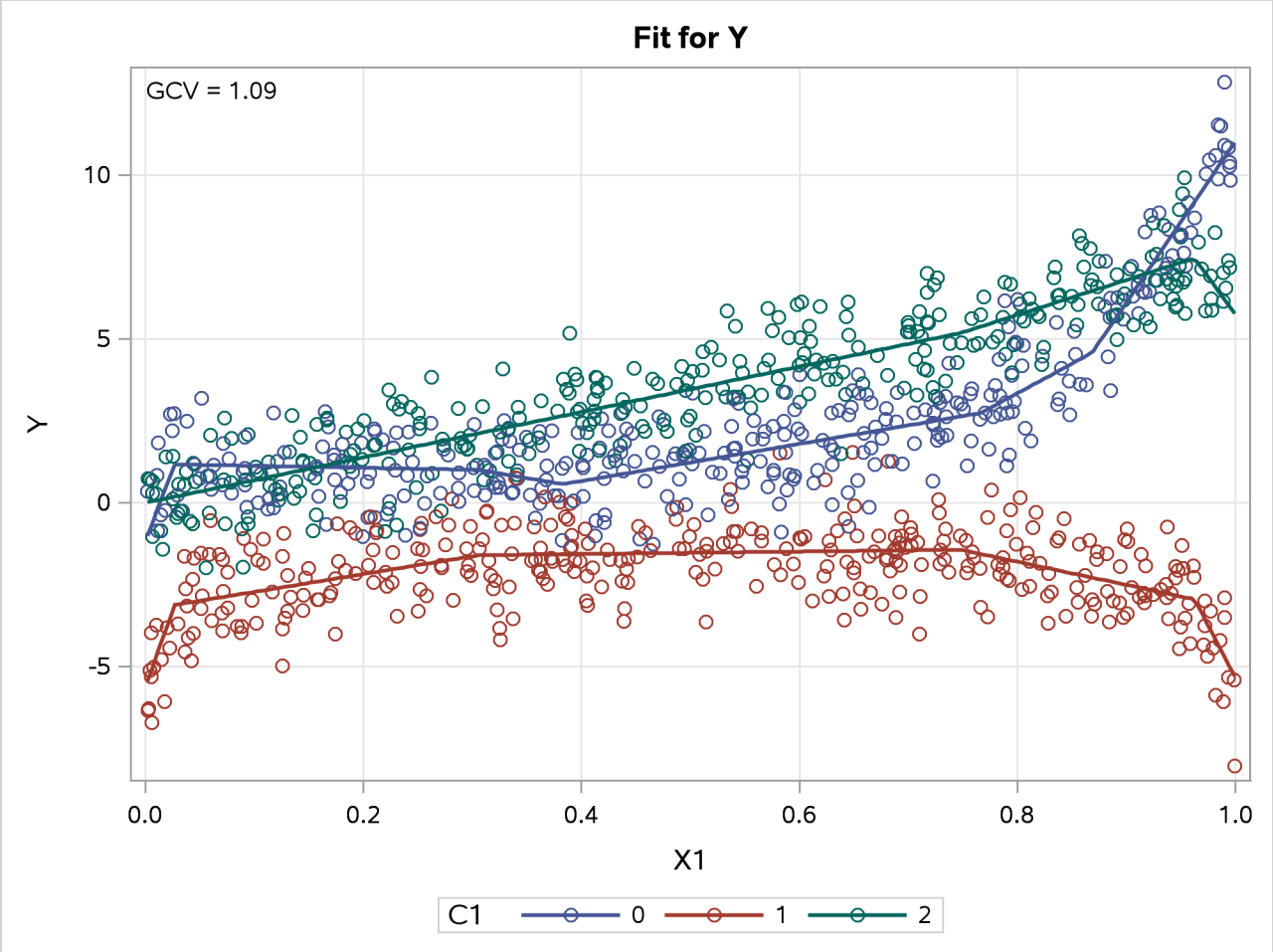

You might notice that some basis functions have their parent basis functions not listed in the parameter estimates table (Output 28.2.1). This is because their parent basis functions are dropped during the model selection process. You can view the complete set of basis functions used in the model selection by specifying the DETAILS=BASES option in the PROC ADAPTIVEREG statement, as in the following statements:

```
proc adaptivereg data=Mixture details=bases;
   class c1;
   model y=c1 x1;
run;
```

Output 28.2.3 Basis Function Information

The ADAPTIVEREG Procedure

Basis Information	
Name	**Transformation**
Basis0	1
Basis1	Basis0*(C1 = 1 OR C1 = 0)
Basis2	Basis0*NOT(C1 = 1 OR C1 = 0)
Basis3	Basis0*(C1 = 1)
Basis4	Basis0*NOT(C1 = 1)
Basis5	Basis4*MAX(X1 - 0.7665019053,0)
Basis6	Basis4*MAX(0.7665019053 - X1,0)
Basis7	Basis2*MAX(X1 - 0.7665019053,0)
Basis8	Basis2*MAX(0.7665019053 - X1,0)
Basis9	Basis3*MAX(X1 - 0.5530566455,0)
Basis10	Basis3*MAX(0.5530566455 - X1,0)
Basis11	Basis3*MAX(X1 - 0.045800759,0)
Basis12	Basis3*MAX(0.045800759 - X1,0)
Basis13	Basis3*MAX(X1 - 0.9526330293,0)
Basis14	Basis3*MAX(0.9526330293 - X1,0)
Basis15	Basis1*MAX(X1 - 0.9499325226,0)
Basis16	Basis1*MAX(0.9499325226 - X1,0)
Basis17	Basis1*MAX(X1 - 0.5142821095,0)
Basis18	Basis1*MAX(0.5142821095 - X1,0)
Basis19	Basis1*MAX(X1 - 0.9889635476,0)
Basis20	Basis1*MAX(0.9889635476 - X1,0)

You can produce a SAS DATA step for scoring new observations by using the information provided in the parameter estimate table and the basis information table, as shown in the following statements:

```
data New;
   basis1  = (c1=1 OR c1=0);
   basis3  = (c1=1);
   basis5  = NOT(c1=1)*MAX(x1-0.7665019053,0);
   basis7  = NOT(c1=1 OR c1=0)*MAX(x1-0.7665019053,0);
   basis8  = NOT(c1=1 OR c1=0)*MAX(0.7665019053-x1,0);
   basis9  = (c1=1)*MAX(x1-0.5530566455,0);
   basis10 = (c1=1)*MAX(0.5530566455-x1,0);
   basis11 = (c1=1)*MAX(x1-0.045800759,0);
   basis13 = (c1=1)*MAX(x1-0.9526330293,0);
   basis15 = (c1=1 OR c1=0)*MAX(x1-0.9499325226,0);
   basis17 = (c1=1 OR c1=0)*MAX(x1-0.5142821095,0);
   basis19 = (c1=1 OR c1=0)*MAX(x1-0.9889635476,0);
   pred    = 5.3829 - 4.3871*basis1 + 32.7761*basis3 +
             20.2859*basis5 - 11.4183*basis7 - 7.0758*basis8 +
             58.4911*basis9 - 71.6388*basis10 - 69.0764*basis11 -
             119.71*basis13 + 66.5733*basis15 + 6.6681*basis17 -
             185.21*basis19;
run;
```

Example 28.3: Predicting E-Mail Spam

(View the complete code for this example (adptex3.sas) in the example repository.)

This example shows how you can use PROC ADAPTIVEREG to fit a classification model for a data set with a binary response. It illustrates how you can use the PARTITION statement to create subsets of data for training and testing purposes. It also demonstrates how to use the OUTPUT statement. Finally, it shows how you can improve the modeling speed by changing some default settings.

This example concerns a study on classifying whether an e-mail is junk e-mail (coded as 1) or not (coded as 0). The data were collected in Hewlett-Packard labs and donated by George Forman. The data set contains 4,601 observations with 58 variables. The response variable is a binary indicator of whether an e-mail is considered spam or not. The 57 variables are continuous variables that record frequencies of some common words and characters in e-mails and lengths of uninterrupted sequences of capital letters. The data set is publicly available at the UCI Machine Learning repository (Asuncion and Newman 2007).

This example shows how you can use PROC ADAPTIVEREG to build a model with good predictive power and then use it to classify observations in independent data sets. PROC ADAPTIVEREG enables you to partition your data into subsets for training, validation, and testing. The training set is used to build models, the validation set is used to estimate prediction errors and select models, and the testing set is used independently to evaluate the final model. When the sample size is not large enough, sample reusing approaches are used instead, such as bootstrap and cross validation. For this data set, the sample size is sufficient to support a random partitioning. Because the GCV model selection criterion itself serves as an estimate of prediction error, this data set is split into two separate subsets. The training set is used to build the classification model, and the test set is used to evaluate the model. The PARTITION statement performs the random partitioning for you, as shown in the following statements:

```
proc adaptivereg data=sashelp.junkmail seed=10359;
   model class = Address     Addresses    All       Bracket    Business
                 CS          CapAvg       CapLong   CapTotal   Conference
                 Credit      Data         Direct    Dollar     Edu
                 Email       Exclamation  Font      Free       George
                 HP          HPL          Internet  Lab        Labs
                 Mail        Make         Meeting   Money      Order
                 Original    Our          Over      PM         Paren
                 Parts       People       Pound     Project    RE
                 Receive     Remove       Report    Semicolon  Table
                 Technology  Telnet       Will      You        Your
                 _000        _85          _415      _650       _857
                 _1999       _3D / additive dist=binomial;
   partition fraction(test=0.333);
   output out=spamout p(ilink);
run;
```

The FRACTION option in the PARTITION statement specifies that 33.3% of observations in the sashelp.junkmail data set are randomly selected to form the testing set while the rest of the data form the training set. If you want to use the same partitioning for further analysis, you can specify the seed for the random number generator so that the exact same random number stream can be duplicated. For the preceding statements, the seed is 10359, which is specified in the PROC ADAPTIVEREG statement. The response variable is a two-level variable. The ADDITIVE option specifies that this is an additive model without interactions between spline basis functions; this option makes the predictive model more interpretable.

The DIST=BINOMIAL option specifies the distribution of the response variable. The ILINK option in the OUTPUT statement requests predicted probabilities for each observation.

The "Model Information" table in Output 28.3.1 includes the distribution, link function, and the random number seed.

Output 28.3.1 Model Information

The ADAPTIVEREG Procedure

Model Information	
Data Set	SASHELP.JUNKMAIL
Response Variable	Class
Distribution	Binomial
Link Function	Logit
Random Number Seed	10359

The "Number of Observations" table in Output 28.3.2 lists the total number of observations used. It also lists number of observations for the training set and the test set.

Output 28.3.2 Number of Observations

Number of Observations Read	4601
Number of Observations Used	4601
Number of Observations Used for Training	3028
Number of Observations Used for Testing	1573

The response variable is a binary classification variable. PROC ADAPTIVEREG produces the "Response Profile" table in Output 28.3.3. The table shows the response level frequencies for the training set and the probability that PROC ADAPTIVEREG models.

Output 28.3.3 Response Profile

Response Profile		
Ordered Value	**Class**	**Total Frequency**
1	0	1844
2	1	1184

Probability modeled is Class='0'.

The "Fit Statistics" table in Output 28.3.3 shows that the final model for the training set contains large effective degrees freedom.

Output 28.3.4 Fit Statistics

Fit Statistics	
GCV	0.23427
GCV R-Square	0.82508
Effective Degrees of Freedom	173
Log Likelihood	-315.30998
Deviance (Train)	630.61996
Deviance (Test)	806.74112

To classify e-mails from the test set, the following rule is used. For each observation, the e-mail is classified as spam if the predicted probability of Class = '0' is greater than the predicted probability of Class = '1', and ham (a good e-mail) otherwise. Because the response is binary, you can classify an e-mail as spam if the predicted probability of Class = '0' is less than 0.5. The following statements evaluate classification errors:

```
data test;
   set spamout(where=(_ROLE_='TEST'));
   if ((pred>0.5 & class=0) | (pred<0.5 & class=1))
   then Error=0;
   else error=1;
run;

proc freq data=test;
   tables class*error/nocol;
run;
```

Output 28.3.5 shows the misclassification errors for all observations and observations of each response category. Compared to the results from other statistical learning algorithms that use different training subsets (Hastie, Tibshirani, and Friedman 2001), these results from PROC ADAPTIVEREG are competitive.

Output 28.3.5 Crosstabulation Table for Test Set Prediction

The FREQ Procedure

Frequency Percent Row Pct	Table of Class by Error			
	Class(0 - Not Junk, 1 - Junk)	Error 0	Error 1	Total
	0	885 56.26 93.75	59 3.75 6.25	944 60.01
	1	592 37.64 94.12	37 2.35 5.88	629 39.99
	Total	1477 93.90	96 6.10	1573 100.00

It takes approximately 300MB of memory and about 102 seconds to fit the model on a workstation with a 12-way 2.6GHz AMD Opteron processor. The following analyses illustrate how you can change some default settings to improve the modeling speed without sacrificing much predictive capability. As discussed in the section "Computational Resources" on page 1092, the computation cost for PROC ADAPTIVEREG is proportional to $pNM_{\max}^3$. For the same data set, you can significantly increase the modeling speed by

reducing the maximum number of basis functions that are allowed for the forward selection.

PROC ADAPTIVEREG uses 115 as the maximum number of basis functions. Suppose you want to set the maximum number to 61, which is approximately half the default value. The following program fits a multivariate adaptive regression splines model with MAXBASIS= set to 61. The same random number seed is used to get the exact same data partitioning.

```
proc adaptivereg data=sashelp.junkmail seed=10359;
   model class = Address     Addresses    All        Bracket    Business
                 CS          CapAvg       CapLong    CapTotal   Conference
                 Credit      Data         Direct     Dollar     Edu
                 Email       Exclamation  Font       Free       George
                 HP          HPL          Internet   Lab        Labs
                 Mail        Make         Meeting    Money      Order
                 Original    Our          Over       PM         Paren
                 Parts       People       Pound      Project    RE
                 Receive     Remove       Report     Semicolon  Table
                 Technology  Telnet       Will       You        Your
                 _000        _85          _415       _650       _857
                 _1999       _3D / maxbasis=61 additive dist=binomial;
   partition fraction(test=0.333);
   output out=spamout2 p(ilink);
run;
```

The "Fit Statistics" table in Output 28.3.6 displays summary statistics for the second model. The log likelihood of the second model is smaller than that of the first model, which is expected because the effective degrees of freedom is 95, much smaller than the effective degrees of freedom of the first model. This means that the fitted model is much simpler than the first model. Both the GCV and GCV R-square values show that the estimated prediction capability of the second model is slightly less than the first model.

Output 28.3.6 Fit Statistics

The ADAPTIVEREG Procedure

Fit Statistics	
GCV	0.27971
GCV R-Square	0.79115
Effective Degrees of Freedom	95
Log Likelihood	-397.32916
Deviance (Train)	794.65833
Deviance (Test)	682.79427

By predicting observations in the test set, the second model has an overall misclassification error of 5.28%, which is slightly lower than that of the first model. This shows that the predictive power of the second model is actually greater than of the first model due to reduced model complexity. The computation takes around 21 seconds on the same workstation and consumes approximately 170MB of memory. This is a significant improvement in both computation speed and memory cost.

You can further improve the modeling speed by using the FAST option in the MODEL statement. The FAST option avoids evaluating certain combinations of parent basis functions and variables. For example, you can specify the FAST(K=20) option so that in each forward selection iteration, PROC ADAPTIVEREG uses only the top 20 parent basis functions (based on their maximum improvement from the previous iteration) to construct and evaluate new basis functions. The underlying assumption, as discussed in the section "Fast

Algorithm" on page 1090, is that parent basis functions that offer low improvement at previous steps are less likely to yield new basis functions that offer large improvement at the current step. The following statements illustrate the FAST option:

```
proc adaptivereg data=sashelp.junkmail seed=10359;
   model class = Address     Addresses    All       Bracket    Business
                 CS          CapAvg       CapLong   CapTotal   Conference
                 Credit      Data         Direct    Dollar     Edu
                 Email       Exclamation  Font      Free       George
                 HP          HPL          Internet  Lab        Labs
                 Mail        Make         Meeting   Money      Order
                 Original    Our          Over      PM         Paren
                 Parts       People       Pound     Project    RE
                 Receive     Remove       Report    Semicolon  Table
                 Technology  Telnet       Will      You        Your
                 _000        _85          _415      _650       _857
                 _1999       _3D / maxbasis=61 fast(k=20) additive dist=binomial;
   partition fraction(test=0.333);
   output out=spamout3 p(ilink);
run;
```

The fitted model is the same as the second model. The computation time on the same workstation is even less at 19 seconds. You should tune the parameters for the FAST option with care because the underlying assumption does not always hold.

With this investigation, the second model can serve as a good classifier. It contains 26 variables. The "Variable Importance" table (Output 28.3.7) lists all variables and their importance values in descending order. Two variables in the model, George and Hp, are important factors in classifying e-mails as not spam. George Forman, the donor of the original data set, collected e-mails from filed work and personal e-mails at Hewlett-Packard labs. Thus these two variables are strong indicators of e-mails that are not spam. This confirms the results from the fitted multivariate adaptive regression splines model by PROC ADAPTIVEREG.

Output 28.3.7 Variable Importance

Variable Importance		
Variable	Number of Bases	Importance
George	1	100.00
HP	1	78.35
Edu	3	61.25
Remove	2	49.21
Exclamation	3	44.14
Free	2	34.18
Meeting	3	32.57
_1999	2	29.71
Dollar	2	28.30
Money	3	26.39
CapLong	3	24.41
Our	2	19.46
Semicolon	2	14.98
RE	2	13.52
Business	3	13.48
Over	3	12.63
CapTotal	3	12.50
Will	1	10.81
Pound	2	9.73
Internet	1	5.88
_000	1	4.57
You	2	3.17

Example 28.4: Nonparametric Poisson Model for Mackerel Egg Density

(View the complete code for this example (adptex4.sas) in the example repository.)

This example demonstrates how you can use PROC ADAPTIVEREG to fit a nonparametric Poisson regression model.

The example concerns a study of mackerel egg density. The data are a subset of the 1992 mackerel egg survey conducted over the Porcupine Bank west of Ireland. The survey took place in the peak spawning area. Scientists took samples by hauling a net up from deep sea to the sea surface. Then they counted the number of spawned mackerel eggs and used other geographic information to estimate the sizes and distributions of spawning stocks. The data set is used as an example in Bowman and Azzalini (1997).

The following SAS DATA step creates the data set Mackerel. This data set contains 634 observations and five variables. The response variable Egg_Count is the number of mackerel eggs collected from each sampling net. Longitude and Latitude are the location values in degrees east and north, respectively, of each sample station. Net_Area is the area of the sampling net in square meters. Depth records the sea bed depth in meters at the sampling location. And Distance is the distance in geographic degrees from the sample location to the continental shelf edge.

```
title 'Mackerel Egg Density Study';
data Mackerel;
   input Egg_Count Longitude Latitude  Net_Area Depth Distance;
   datalines;
   0     -4.65     44.57     0.242    4342    0.8395141177
   0     -4.48     44.57     0.242    4334    0.8591926336
   0      -4.3     44.57     0.242    4286    0.8930152895
   1     -2.87     44.02     0.242    1438    0.3956408691
   4     -2.07     44.02     0.242     166    0.0400088237
   3     -2.13     44.02     0.242     460    0.0974234463
   0     -2.27     44.02     0.242     810    0.2362566569

   ... more lines ...

  22     -4.22     46.25      0.19     205    0.1181120828
  21     -4.28     46.25      0.19     237     0.129990854
   0     -4.73     46.25      0.19    2500    0.3346500536
   5     -4.25     47.23      0.19     114     0.718192582
   3     -3.72     47.25      0.19     100    0.9944669778
   0     -3.25     47.25      0.19      64    1.2639918431
;
```

The response values are counts, so the Poisson distribution might be a reasonable model. The study of interest is the mackerel egg density, which can be formed as

$$\text{density} = E(\text{count})/\text{net_area}$$

This is equivalent to a Poisson regression with the response variable Egg_Count and an offset variable $\log(\text{net_area})$ and other covariates.

The following statements produce the plot of the mackerel egg density with respect to the sampling station location:

```
data temp;
   set mackerel;
   density = egg_count/net_area;
run;

%let off0 = offsetmin=0 offsetmax=0 linearopts=(thresholdmin=0 thresholdmax=0);
proc template;
   define statgraph surface;
      dynamic _title _z;
      begingraph / designwidth=defaultDesignHeight;
         entrytitle _title;
         layout overlay / xaxisopts=(&off0) yaxisopts=(&off0);
            contourplotparm z=_z y=latitude x=longitude / gridded=FALSE;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=temp template=surface;
   dynamic _title='Mackerel Egg Density' _z='density';
run;
```

Output 28.4.1 displays the mackerel egg density in the sampling area. The black hole in the upper right corner is due to missing values in that area.

Output 28.4.1 Mackerel Egg Density

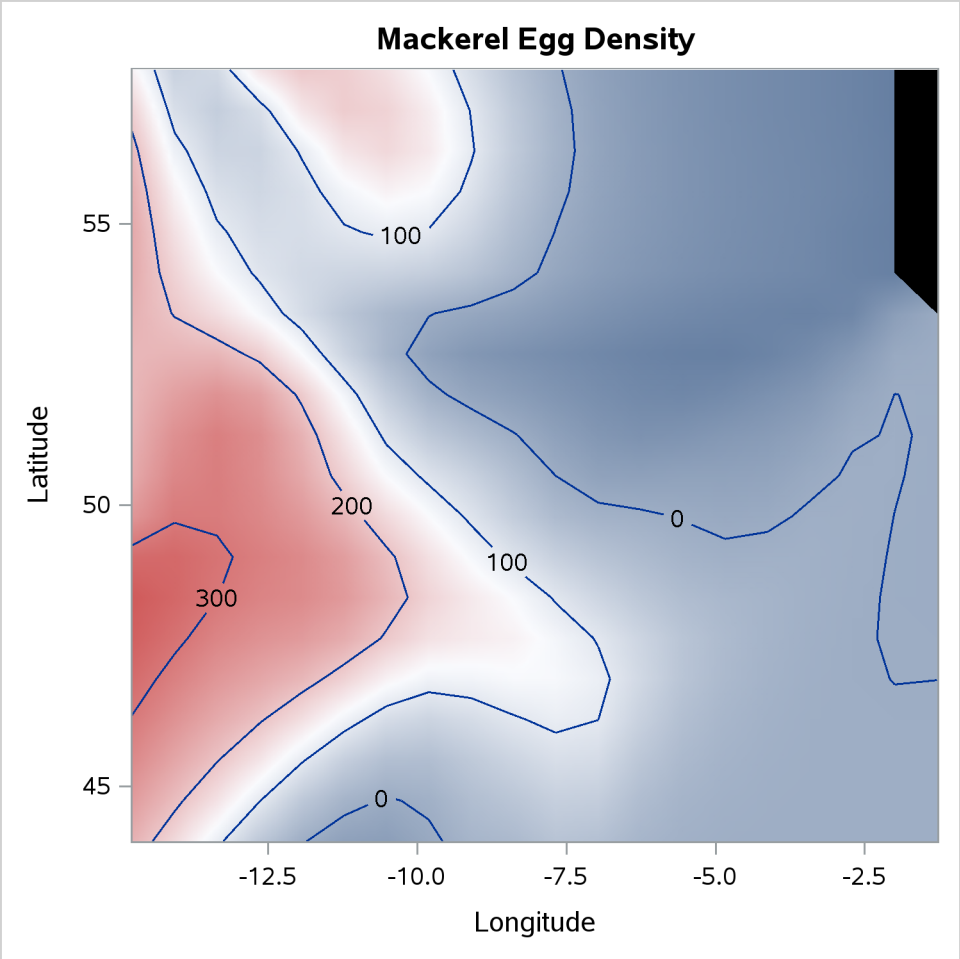

In this example, the dependent variable is the mackerel egg counts, the independent variables are the geographical information about each of the sampling stations, and the logarithm of the sampling area is the offset variable. The following statements fit the nonparametric Poisson regression model:

```
data mackerel;
   set mackerel;
   log_net_area = log(net_area);
run;

proc adaptivereg data=mackerel;
   model egg_count = longitude latitude depth distance
                     / offset=log_net_area dist=poisson;
   output out=mackerelout p(ilink);
run;
```

Output 28.4.2 lists basic model information such as the offset variable, distribution, and link function.

Output 28.4.2 Model Information

Mackerel Egg Density Study

The ADAPTIVEREG Procedure

Model Information	
Data Set	WORK.MACKEREL
Response Variable	Egg_Count
Offset Variable	log_net_area
Distribution	Poisson
Link Function	Log

Output 28.4.3 lists fit statistics for the final model.

Output 28.4.3 Fit Statistics

Fit Statistics	
GCV	6.94340
GCV R-Square	0.79204
Effective Degrees of Freedom	29
Log Likelihood	-2777.21279
Deviance	4008.60601

The final model consists of basis functions and interactions between basis functions of three geographic variables. Output 28.4.4 lists seven functional components of the final model, including three one-way spline transformations and four two-way spline interactions.

Output 28.4.4 ANOVA Decomposition

ANOVA Decomposition				
			Change If Omitted	
Functional Component	**Number of Bases**	**DF**	**Lack of Fit**	**GCV**
Longitude	3	6	2035.77	3.3216
Depth	1	2	420.59	0.6780
Latitude	1	2	265.05	0.4104
Longitude Latitude	2	4	199.17	0.2496
Depth Distance	3	6	552.75	0.8030
Depth Latitude	2	4	680.45	1.0723
Depth Longitude	2	4	415.77	0.6198

The "Variable Importance" table in Output 28.4.5 displays the relative variable importance among the four variables. Longitude is the most important one.

Output 28.4.5 Variable Importance

Variable Importance		
Variable	Number of Bases	Importance
Longitude	7	100.00
Depth	8	30.26
Latitude	5	18.93
Distance	3	8.56

The following steps create and display in Output 28.4.6 the predicted mackerel egg density over the spawning area.

```
data mackplot;
   set mackerelout;
   density = pred / net_area;
run;

proc sgrender data=mackplot template=surface;
   dynamic _title='Predicted Mackerel Egg Density'
           _z='density';
run;
```

Output 28.4.6 Predicted Mackerel Egg Density

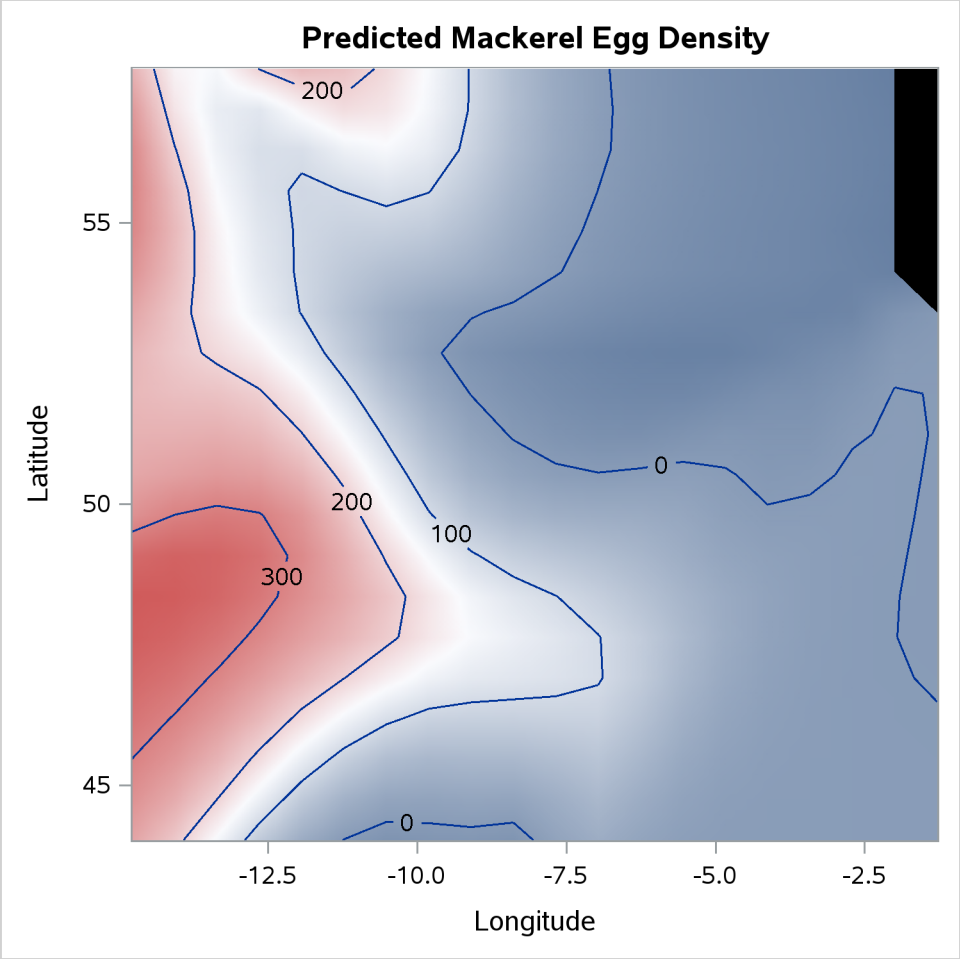

References

Asuncion, A., and Newman, D. J. (2007). "UCI Machine Learning Repository." http://archive.ics.uci.edu/ml/.

Bellman, R. E. (1961). *Adaptive Control Processes*. Princeton, NJ: Princeton University Press.

Bowman, A. W., and Azzalini, A. (1997). *Applied Smoothing Techniques for Data Analysis*. New York: Oxford University Press.

Breiman, L., Friedman, J., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Belmont, CA: Wadsworth.

Buja, A., Duffy, D., Hastie, T. J., and Tibshirani, R. (1991). "Discussion: Multivariate Adaptive Regression Splines." *Annals of Statistics* 19:93–99.

Craven, P., and Wahba, G. (1979). "Smoothing Noisy Data with Spline Functions." *Numerical Mathematics* 31:377–403.

Friedman, J. H. (1991a). *Estimating Functions of Mixed Ordinal and Categorical Variables Using Adaptive Splines*. Technical report, Stanford University.

Friedman, J. H. (1991b). "Multivariate Adaptive Regression Splines." *Annals of Statistics* 19:1–67.

Friedman, J. H. (1993). *Fast MARS*. Technical report, Stanford University.

Gu, C., Bates, D. M., Chen, Z., and Wahba, G. (1990). "The Computation of GCV Function through Householder Tridiagonalization with Application to the Fitting of Interaction Splines Models." *SIAM Journal on Matrix Analysis and Applications* 10:457–480.

Hastie, T. J., and Tibshirani, R. J. (1990). *Generalized Additive Models*. New York: Chapman & Hall.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Owen, A. (1991). "Discussion of 'Multivariate Adaptive Regression Splines' by J. H. Friedman." *Annals of Statistics* 19:102–112.

Smith, P. L. (1982). *Curve Fitting and Modeling with Splines Using Statistical Variable Selection Techniques*. Technical report, NASA Langley Research Center.

Chapter 29
The ANOVA Procedure

Contents

Overview: ANOVA Procedure **1118**
Getting Started: ANOVA Procedure **1119**
One-Way Layout with Means Comparisons 1119
Randomized Complete Block with One Factor 1123
Syntax: ANOVA Procedure **1126**
PROC ANOVA Statement 1127
ABSORB Statement 1129
BY Statement 1129
CLASS Statement 1130
FREQ Statement 1131
MANOVA Statement 1131
MEANS Statement 1135
MODEL Statement 1141
REPEATED Statement 1141
TEST Statement 1146
Details: ANOVA Procedure **1147**
Specification of Effects 1147
Using PROC ANOVA Interactively 1150
Missing Values 1150
Output Data Set 1151
Computational Method 1152
Displayed Output 1152
ODS Table Names 1154
ODS Graphics 1155
Examples: ANOVA Procedure **1156**
Example 29.1: Factorial Treatments in Complete Blocks 1156
Example 29.2: Alternative Multiple Comparison Procedures 1158
Example 29.3: Split Plot 1164
Example 29.4: Latin Square Split Plot 1166
Example 29.5: Strip-Split Plot 1168
References **1172**

Overview: ANOVA Procedure

The ANOVA procedure performs *analysis of variance* (ANOVA) for balanced data from a wide variety of experimental designs. In analysis of variance, a continuous response variable, known as a *dependent variable*, is measured under experimental conditions identified by classification variables, known as *independent variables*. The variation in the response is assumed to be due to effects in the classification, with random error accounting for the remaining variation.

The ANOVA procedure is one of several procedures available in SAS/STAT software for analysis of variance. The ANOVA procedure is designed to handle balanced data (that is, data with equal numbers of observations for every combination of the classification factors), whereas the GLM procedure can analyze both balanced and unbalanced data. Because PROC ANOVA takes into account the special structure of a balanced design, it is faster and uses less storage than PROC GLM for balanced data.

Use PROC ANOVA for the analysis of balanced data only, with the following exceptions: one-way analysis of variance, Latin square designs, certain partially balanced incomplete block designs, completely nested (hierarchical) designs, and designs with cell frequencies that are proportional to each other and are also proportional to the background population. These exceptions have designs in which the factors are all orthogonal to each other.

For further discussion, see Searle (1971, p. 138). PROC ANOVA works for designs with block diagonal $\mathbf{X}'\mathbf{X}$ matrices where the elements of each block all have the same value. The procedure partially tests this requirement by checking for equal cell means. However, this test is imperfect: some designs that cannot be analyzed correctly might pass the test, and designs that can be analyzed correctly might not pass. If your design does not pass the test, PROC ANOVA produces a warning message to tell you that the design is unbalanced and that the ANOVA analyses might not be valid; if your design is not one of the special cases described here, then you should use PROC GLM instead. Complete validation of designs is not performed in PROC ANOVA since this would require the whole $\mathbf{X}'\mathbf{X}$ matrix; if you are unsure about the validity of PROC ANOVA for your design, you should use PROC GLM.

CAUTION: If you use PROC ANOVA for analysis of unbalanced data, you must assume responsibility for the validity of the results.

The ANOVA procedure automatically produces graphics as part of its ODS output. For general information about ODS graphics, see the section "ODS Graphics" on page 1155 and Chapter 24, "Statistical Graphics Using ODS."

Getting Started: ANOVA Procedure

The following examples demonstrate how you can use the ANOVA procedure to perform analyses of variance for a one-way layout and a randomized complete block design.

One-Way Layout with Means Comparisons

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/anovain1.sas.)

A one-way analysis of variance considers one treatment factor with two or more treatment levels. The goal of the analysis is to test for differences among the means of the levels and to quantify these differences. If there are two treatment levels, this analysis is equivalent to a *t* test comparing two group means.

The assumptions of analysis of variance (Steel and Torrie 1980) are that treatment effects are additive and experimental errors are independently random with a normal distribution that has mean zero and constant variance.

The following example studies the effect of bacteria on the nitrogen content of red clover plants. The treatment factor is bacteria strain, and it has six levels. Five of the six levels consist of different *Rhizobium trifolii* bacteria cultures combined with a composite of five *Rhizobium meliloti* strains. The sixth level is a composite of the five *Rhizobium trifolii* strains with the composite of the *Rhizobium meliloti*. Red clover plants are inoculated with the treatments, and nitrogen content is later measured in milligrams. The data are derived from an experiment by Erdman (1946) and are analyzed in Chapters 7 and 8 of Steel and Torrie (1980). The following DATA step creates the SAS data set Clover:

```
title1 'Nitrogen Content of Red Clover Plants';
data Clover;
   input Strain $ Nitrogen @@;
   datalines;
3DOK1  19.4 3DOK1  32.6 3DOK1  27.0 3DOK1  32.1 3DOK1  33.0
3DOK5  17.7 3DOK5  24.8 3DOK5  27.9 3DOK5  25.2 3DOK5  24.3
3DOK4  17.0 3DOK4  19.4 3DOK4   9.1 3DOK4  11.9 3DOK4  15.8
3DOK7  20.7 3DOK7  21.0 3DOK7  20.5 3DOK7  18.8 3DOK7  18.6
3DOK13 14.3 3DOK13 14.4 3DOK13 11.8 3DOK13 11.6 3DOK13 14.2
COMPOS 17.3 COMPOS 19.4 COMPOS 19.1 COMPOS 16.9 COMPOS 20.8
;
```

The variable Strain contains the treatment levels, and the variable Nitrogen contains the response. The following statements produce the analysis:

```
ods graphics on;

proc anova data = Clover;
   class Strain;
   model Nitrogen = Strain;
   means Strain / tukey;
run;
```

The classification variable is specified in the CLASS statement. Note that, unlike the GLM procedure, PROC ANOVA does not allow continuous variables on the right-hand side of the model. The following figures display the output produced by these statements.

Figure 29.1 Class Level Information

Nitrogen Content of Red Clover Plants

The ANOVA Procedure

Class Level Information		
Class	**Levels**	**Values**
Strain	6	3DOK1 3DOK13 3DOK4 3DOK5 3DOK7 COMPOS

Number of Observations Read	30
Number of Observations Used	30

The "Class Level Information" table shown in Figure 29.1 lists the variables that appear in the CLASS statement, their levels, and the number of observations in the data set.

Figure 29.2 displays the ANOVA table, followed by some simple statistics and tests of effects.

Figure 29.2 ANOVA Table

Nitrogen Content of Red Clover Plants

The ANOVA Procedure

Dependent Variable: Nitrogen

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	847.046667	169.409333	14.37	<.0001
Error	24	282.928000	11.788667		
Corrected Total	29	1129.974667			

R-Square	Coeff Var	Root MSE	Nitrogen Mean
0.749616	17.26515	3.433463	19.88667

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Strain	5	847.0466667	169.4093333	14.37	<.0001

The degrees of freedom (DF) column should be used to check the analysis results. The model degrees of freedom for a one-way analysis of variance are the number of levels minus 1; in this case, 6 – 1 = 5. The Corrected Total degrees of freedom are always the total number of observations minus one; in this case 30 – 1 = 29. The sum of Model and Error degrees of freedom equals the Corrected Total.

With ODS Graphics enabled, PROC ANOVA also displays by default a plot that enables you to visualize the distribution of nitrogen content for each treatment, using a box plot of the nitrogen content values within each level of the treatment. This plot is shown in Figure 29.3.

Figure 29.3 Box Plot of Nitrogen Content for Each Treatment

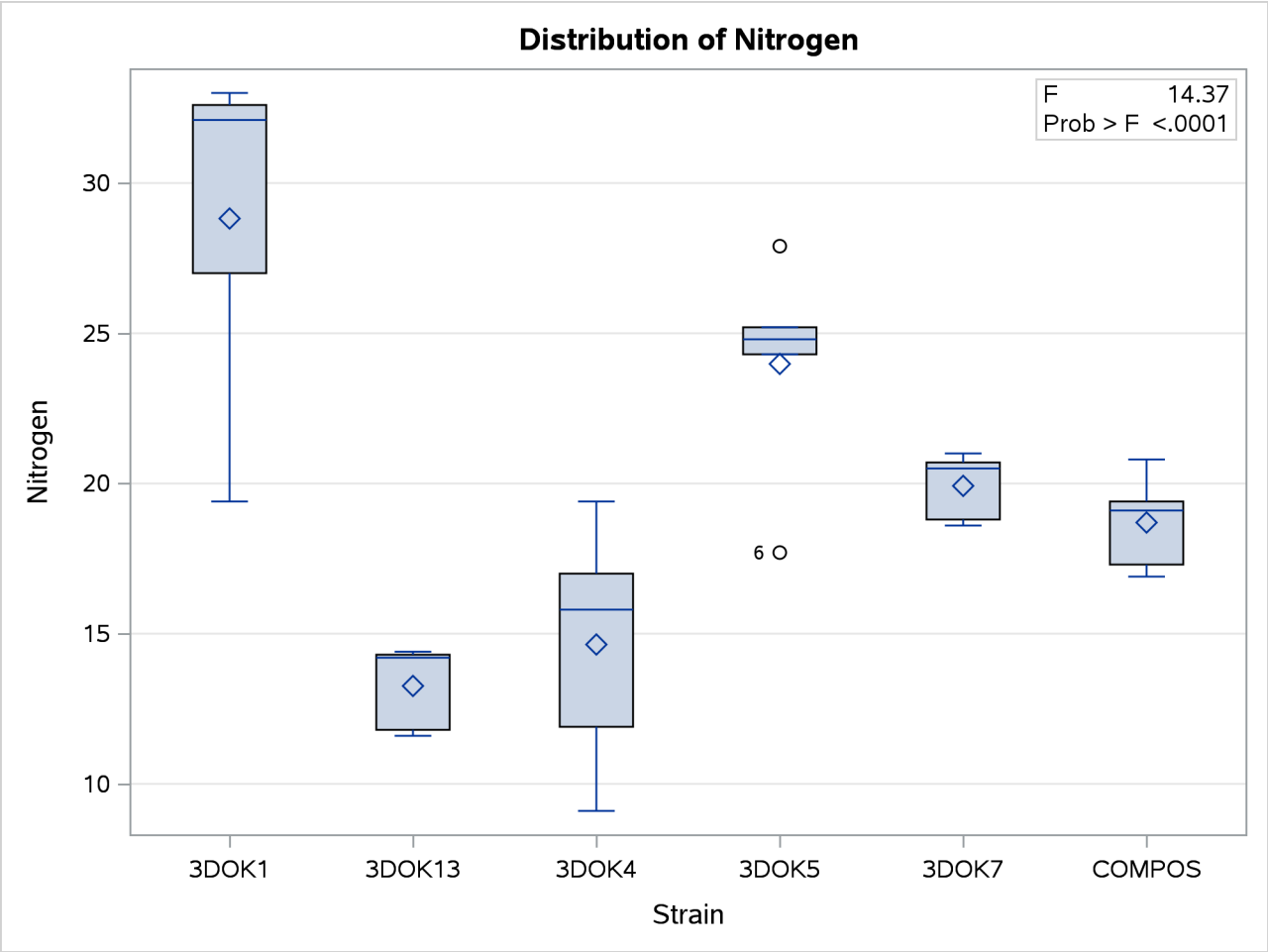

For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the ANOVA procedure, see the section "ODS Graphics" on page 1155.

The overall F test is significant ($F = 14.37$, $p < 0.0001$), indicating that the model as a whole accounts for a significant portion of the variability in the dependent variable. The F test for Strain is significant, indicating that some contrast between the means for the different strains is different from zero. Notice that the Model and Strain F tests are identical, since Strain is the only term in the model.

The significant F test for Strain suggests that there are differences among the bacterial strains, but it does not reveal any information about the nature of the differences. You can use mean comparison methods to gather further information. The MEANS statement requests comparisons of the means of the Strain levels by using Tukey's studentized range method. Results of Tukey's method are shown in Figure 29.4.

Figure 29.4 Tukey's Multiple Comparisons Method

Nitrogen Content of Red Clover Plants

The ANOVA Procedure

Tukey's Studentized Range (HSD) Test for Nitrogen

Note: This test controls the Type I experimentwise error rate, but it generally has a higher Type II error rate than REGWQ.

Alpha	0.05
Error Degrees of Freedom	24
Error Mean Square	11.78867
Critical Value of Studentized Range	4.37265
Minimum Significant Difference	6.7142

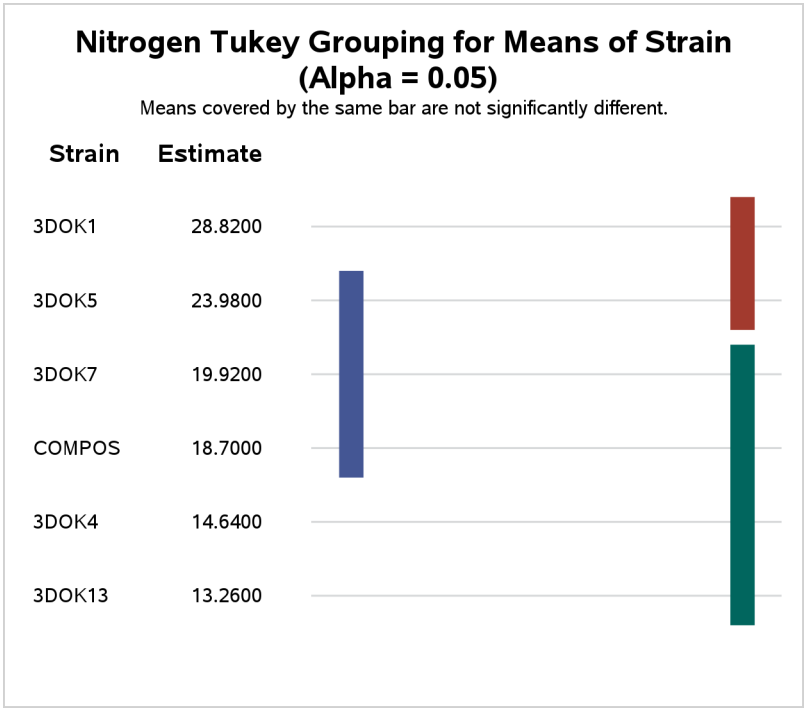

Examples of implications of the multiple comparisons results are as follows:

- Strain 3DOK1 fixes significantly more nitrogen than all but 3DOK5.
- While 3DOK5 is not significantly different from 3DOK1, it is also not significantly better than all the rest, though it is better than the bottom two groups.

Although the experiment has succeeded in separating the best strains from the worst, more experimentation is required in order to clearly distinguish the very best strain.

Randomized Complete Block with One Factor

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/anovain2.sas.)

This example illustrates the use of PROC ANOVA in analyzing a randomized complete block design. Researchers are interested in whether three treatments have different effects on the yield and worth of a particular crop. They believe that the experimental units are not homogeneous. So, a blocking factor is introduced that allows the experimental units to be homogeneous within each block. The three treatments are then randomly assigned within each block.

The data from this study are input into the SAS data set RCB:

```
title1 'Randomized Complete Block';
data RCB;
   input Block Treatment $ Yield Worth @@;
   datalines;
1 A 32.6 112   1 B 36.4 130   1 C 29.5 106
2 A 42.7 139   2 B 47.1 143   2 C 32.9 112
3 A 35.3 124   3 B 40.1 134   3 C 33.6 116
;
```

The variables Yield and Worth are continuous response variables, and the variables Block and Treatment are the classification variables. Because the data for the analysis are balanced, you can use PROC ANOVA to run the analysis.

The statements for the analysis are

```
proc anova data=RCB;
   class Block Treatment;
   model Yield Worth=Block Treatment;
run;
```

The Block and Treatment effects appear in the CLASS statement. The MODEL statement requests an analysis for each of the two dependent variables, Yield and Worth.

Figure 29.5 shows the "Class Level Information" table.

Figure 29.5 Class Level Information

Randomized Complete Block

The ANOVA Procedure

Class Level Information		
Class	**Levels**	**Values**
Block	3	1 2 3
Treatment	3	A B C

Number of Observations Read	9
Number of Observations Used	9

The "Class Level Information" table lists the number of levels and their values for all effects specified in the

CLASS statement. The number of observations in the data set are also displayed. Use this information to make sure that the data have been read correctly.

The overall ANOVA table for Yield in Figure 29.6 appears first in the output because it is the first response variable listed on the left side in the MODEL statement.

Figure 29.6 Overall ANOVA Table for Yield

Randomized Complete Block

The ANOVA Procedure

Dependent Variable: Yield

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	225.2777778	56.3194444	8.94	0.0283
Error	4	25.1911111	6.2977778		
Corrected Total	8	250.4688889			

R-Square	Coeff Var	Root MSE	Yield Mean
0.899424	6.840047	2.509537	36.68889

The overall F statistic is significant ($F = 8.94$, $p = 0.0283$), indicating that the model as a whole accounts for a significant portion of the variation in Yield and that you can proceed to evaluate the tests of effects.

The degrees of freedom (DF) are used to ensure correctness of the data and model. The Corrected Total degrees of freedom are one less than the total number of observations in the data set; in this case, 9 – 1 = 8. The Model degrees of freedom for a randomized complete block are $(b - 1) + (t - 1)$, where b = number of block levels and t = number of treatment levels. In this case, this formula leads to $(3 - 1) + (3 - 1) = 4$ model degrees of freedom.

Several simple statistics follow the ANOVA table. The R-Square indicates that the model accounts for nearly 90% of the variation in the variable Yield. The coefficient of variation (C.V.) is listed along with the Root MSE and the mean of the dependent variable. The Root MSE is an estimate of the standard deviation of the dependent variable. The C.V. is a unitless measure of variability.

The tests of the effects shown in Figure 29.7 are displayed after the simple statistics.

Figure 29.7 Tests of Effects for Yield

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Block	2	98.1755556	49.0877778	7.79	0.0417
Treatment	2	127.1022222	63.5511111	10.09	0.0274

For Yield, both the Block and Treatment effects are significant ($F = 7.79$, $p = 0.0417$ and $F = 10.09$, $p = 0.0274$, respectively) at the 95% level. From this you can conclude that blocking is useful for this variable and that some contrast between the treatment means is significantly different from zero.

Figure 29.8 shows the ANOVA table, simple statistics, and tests of effects for the variable Worth.

Figure 29.8 ANOVA Table for Worth

Randomized Complete Block

The ANOVA Procedure

Dependent Variable: Worth

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	1247.333333	311.833333	8.28	0.0323
Error	4	150.666667	37.666667		
Corrected Total	8	1398.000000			

R-Square	Coeff Var	Root MSE	Worth Mean
0.892227	4.949450	6.137318	124.0000

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Block	2	354.6666667	177.3333333	4.71	0.0889
Treatment	2	892.6666667	446.3333333	11.85	0.0209

The overall F test is significant ($F = 8.28$, $p = 0.0323$) at the 95% level for the variable Worth. The Block effect is not significant at the 0.05 level but is significant at the 0.10 confidence level ($F = 4.71$, $p = 0.0889$). Generally, the usefulness of blocking should be determined before the analysis. However, since there are two dependent variables of interest, and Block is significant for one of them (Yield), blocking appears to be generally useful. For Worth, as with Yield, the effect of Treatment is significant ($F = 11.85$, $p = 0.0209$).

Issuing the following command produces the Treatment means.

```
    means Treatment;
  run;
```

Figure 29.9 displays the treatment means and their standard deviations for both dependent variables.

Figure 29.9 Means of Yield and Worth

Randomized Complete Block

The ANOVA Procedure

		Yield		Worth	
Level of Treatment	N	Mean	Std Dev	Mean	Std Dev
A	3	36.8666667	5.22908532	125.000000	13.5277493
B	3	41.2000000	5.43415127	135.666667	6.6583281
C	3	32.0000000	2.19317122	111.333333	5.0332230

Syntax: ANOVA Procedure

The following statements are available in the ANOVA procedure:

```
PROC ANOVA <options> ;
    CLASS variables </ option> ;
    MODEL dependents = effects </ options> ;
    ABSORB variables ;
    BY variables ;
    FREQ variable ;
    MANOVA <test-options> </ detail-options> ;
    MEANS effects </ options> ;
    REPEATED factor-specification </ options> ;
    TEST <H=effects> E=effect ;
```

The PROC ANOVA, CLASS, and MODEL statements are required, and they must precede the first RUN statement. The CLASS statement must precede the MODEL statement. If you use the ABSORB, FREQ, or BY statement, it must precede the first RUN statement. The MANOVA, MEANS, REPEATED, and TEST statements must follow the MODEL statement, and they can be specified in any order. These four statements can also appear after the first RUN statement.

Table 29.1 summarizes the function of each statement (other than the PROC statement) in the ANOVA procedure:

Table 29.1 Statements in the ANOVA Procedure

Statement	Description
ABSORB	Absorbs classification effects in a model
BY	Specifies variables to define subgroups for the analysis
CLASS	Declares classification variables
FREQ	Specifies a frequency variable
MANOVA	Performs a multivariate analysis of variance
MEANS	Computes and compares means
MODEL	Defines the model to be fit
REPEATED	Performs multivariate and univariate repeated measures analysis of variance
TEST	Constructs tests that use the sums of squares for effects and the error term you specify

PROC ANOVA Statement

PROC ANOVA < *options* > **;**

The PROC ANOVA statement invokes the ANOVA procedure. Table 29.2 summarizes the *options* available in the PROC ANOVA statement.

Table 29.2 PROC ANOVA Statement Options

Option	Description
Specify Input and Output Data Sets	
DATA=	Specifies input SAS data set
MANOVA	Requests the multivariate mode of eliminating observations with missing values
MULTIPASS	Requests that the input data set be reread when necessary, instead of using a utility file
NAMELEN=	Specifies the length of effect names
NOPRINT	Suppresses the normal display of results
ORDER=	Specifies the sort order for the levels of the classification variables
OUTSTAT=	Names an output data set for information and statistics on each model effect
PLOTS	Controls the plots produced through ODS Graphics.

You can specify the following *options* in the PROC ANOVA statement:

DATA=*SAS-data-set*
names the SAS data set used by the ANOVA procedure. By default, PROC ANOVA uses the most recently created SAS data set.

MANOVA
requests the multivariate mode of eliminating observations with missing values. If any of the dependent variables have missing values, the procedure eliminates that observation from the analysis. The MANOVA option is useful if you use PROC ANOVA in interactive mode and plan to perform a multivariate analysis.

MULTIPASS
requests that PROC ANOVA reread the input data set, when necessary, instead of writing the values of dependent variables to a utility file. This option decreases disk space usage at the expense of increased execution times and is useful only in rare situations where disk space is at an absolute premium.

NAMELEN=*n*
specifies the length of effect names to be *n* characters long, where *n* is a value between 20 and 200 characters. The default length is 20 characters.

NOPRINT

suppresses the normal display of results. The NOPRINT option is useful when you want to create only the output data set with the procedure. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

OUTSTAT=*SAS-data-set*

names an output data set that contains sums of squares, degrees of freedom, *F* statistics, and probability levels for each effect in the model. If you use the CANONICAL option in the MANOVA statement and do not use an M= specification in the MANOVA statement, the data set also contains results of the canonical analysis. See the section "Output Data Set" on page 1151 for more information.

PLOTS < (MAXPOINTS=NONE | *number***) > < =NONE >**

PLOTS=NONE

controls the plots produced through ODS Graphics. When ODS Graphics is enabled, the ANOVA procedure can display a grouped box plot of the input data with groups defined by an effect in the model. Such a plot is produced by default if you have a one-way model, with only a single classification variable, or if you use a MEANS statement. Specify the PLOTS=NONE option to prevent these plots from being produced when ODS Graphics is enabled.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc anova data = Clover;
   class Strain;
   model Nitrogen = Strain;
run;
```

```
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The following option can be specified in parentheses after PLOTS.

MAXPOINTS=NONE | *number*
specifies that plots with elements that require processing of more than *number* points be suppressed. The default is MAXPOINTS=5000. This limit is ignored if you specify MAXPOINTS=NONE.

ABSORB Statement

ABSORB *variables* **;**

Absorption is a computational technique that provides a large reduction in time and memory requirements for certain types of models. The *variables* are one or more variables in the input data set.

For a main effect variable that does not participate in interactions, you can absorb the effect by naming it in an ABSORB statement. This means that the effect can be adjusted out before the construction and solution of the rest of the model. This is particularly useful when the effect has a large number of levels.

Several variables can be specified, in which case each one is assumed to be nested in the preceding variable in the ABSORB statement.

NOTE: When you use the ABSORB statement, the data set (or each BY group, if a BY statement appears) must be sorted by the variables in the ABSORB statement. Including an absorbed variable in the CLASS list or in the MODEL statement might produce erroneous sums of squares. If the ABSORB statement is used, it must appear before the first RUN statement or it is ignored.

When you use an ABSORB statement and also use the INT option in the MODEL statement, the procedure ignores the option but produces the uncorrected total sum of squares (SS) instead of the corrected total SS.

See the section "Absorption" on page 4224 in Chapter 53, "The GLM Procedure," for more information.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC ANOVA to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the ANOVA procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Since sorting the data changes the order in which PROC ANOVA reads observations, the sort order for the levels of the classification variables might be affected if you have also specified the ORDER=DATA option in the PROC ANOVA statement.

If the BY statement is used, it must appear before the first RUN statement, or it is ignored. When you use a BY statement, the interactive features of PROC ANOVA are disabled.

When both a BY and an ABSORB statement are used, observations must be sorted first by the variables in the BY statement, and then by the variables in the ABSORB statement.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(REF=** *option***)** > . . . < *variable* < **(REF=** *option***)** > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC ANOVA statement.

You can specify the following REF= option to indicate how the levels of an individual classification variable are to be ordered by enclosing it in parentheses after the variable name:

REF=*'level'* | **FIRST** | **LAST**

specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC ANOVA's singular parameterization. You can specify the *level* of the variable to use as the reference level; specify a value that corresponds to the formatted value of the variable if a format is assigned. Alternatively, you can specify REF=FIRST to designate that the first ordered level serve as the reference, or REF=LAST to designate that the last ordered level serve as the reference. To specify that REF=FIRST or REF=LAST be used for all classification variables, use the REF= *global-option* after the slash (/) in the CLASS statement.

You can specify the following *global-options* in the CLASS statement after a slash (/):

REF=FIRST | LAST

specifies a level of all classification variables to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC ANOVA's singular parameterization. Specify REF=FIRST to designate that the first ordered level for each classification variable serve as the reference. Specify REF=LAST to designate that the last ordered level serve as the reference. This option applies to all the variables specified in the CLASS statement. To specify different reference levels for different classification variables, use REF= options for individual variables.

TRUNCATE

specifies that class levels be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a variable that provides frequencies for each observation in the DATA= data set. Specifically, if n is the value of the FREQ variable for a given observation, then that observation is used n times.

The analysis produced by using a FREQ statement reflects the expanded number of observations. For example, means and total degrees of freedom reflect the expanded number of observations. You can produce the same analysis (without the FREQ statement) by first creating a new data set that contains the expanded number of observations. For example, if the value of the FREQ variable is 5 for the first observation, the first 5 observations in the new data set would be identical. Each observation in the old data set would be replicated n_i times in the new data set, where n_i is the value of the FREQ variable for that observation.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

If the FREQ statement is used, it must appear before the first RUN statement or it is ignored.

MANOVA Statement

MANOVA < *test-options* > < *detail-options* > **;**

If the MODEL statement includes more than one dependent variable, you can perform multivariate analysis of variance with the MANOVA statement. The *test-options* define which effects to test, while the *detail-options* specify how to execute the tests and what results to display.

When a MANOVA statement appears before the first RUN statement, PROC ANOVA enters a multivariate mode with respect to the handling of missing values; in addition to observations with missing independent variables, observations with *any* missing dependent variables are excluded from the analysis. If you want to use this mode of handling missing values but do not need any multivariate analyses, specify the MANOVA option in the PROC ANOVA statement.

Table 29.3 summarizes the *options* available in the MANOVA statement.

Table 29.3 MANOVA Statement Options

Option	Description
Test Options	
H=	Specifies hypothesis effects
E=	Specifies the error effect
M=	Specifies a transformation matrix for the dependent variables
MNAMES=	Provides names for the transformed variables
PREFIX=	Alternatively identifies the transformed variables
Detail Options	
CANONICAL	Displays a canonical analysis of the **H** and **E** matrices
MSTAT=	Specifies the method of evaluating the multivariate test statistics
ORTH	Orthogonalizes the rows of the transformation matrix
PRINTE	Displays the error SSCP matrix **E**
PRINTH	Displays the hypothesis SSCP matrix **H**
SUMMARY	Produces analysis-of-variance tables for each dependent variable

Test Options

You can specify the following *options* in the MANOVA statement as *test-options* in order to define which multivariate tests to perform.

H=*effects* **| INTERCEPT | _ALL_**

specifies effects in the preceding model to use as hypothesis matrices. For each SSCP matrix **H** associated with an effect, the H= specification computes an analysis based on the characteristic roots of $\mathbf{E}^{-1}\mathbf{H}$, where **E** is the matrix associated with the error effect. The characteristic roots and vectors are displayed, along with the Hotelling-Lawley trace, Pillai's trace, Wilks' lambda, and Roy's greatest root. By default, these statistics are tested with approximations based on the F distribution. To test them with exact (but computationally intensive) calculations, use the MSTAT=EXACT option.

Use the keyword INTERCEPT to produce tests for the intercept. To produce tests for all effects listed in the MODEL statement, use the keyword _ALL_ in place of a list of effects.

For background and further details, see the section "Multivariate Analysis of Variance" on page 4247 in Chapter 53, "The GLM Procedure."

E=*effect*

specifies the error effect. If you omit the E= specification, the ANOVA procedure uses the error SSCP (residual) matrix from the analysis.

M=*equation,. . . ,equation* **|** (*row-of-matrix,. . . ,row-of-matrix*)

specifies a transformation matrix for the dependent variables listed in the MODEL statement. The equations in the M= specification are of the form

$$\begin{aligned} c_1 \times dependent - variable \quad &\pm \quad c_2 \times dependent - variable \\ \cdots \quad &\pm \quad c_n \times dependent - variable \end{aligned}$$

where the c_i values are coefficients for the various *dependent-variables*. If the value of a given c_i is 1, it can be omitted; in other words $1 \times Y$ is the same as Y. Equations should involve two or more dependent variables. For sample syntax, see the section "Examples" on page 1134.

Alternatively, you can input the transformation matrix directly by entering the elements of the matrix with commas separating the rows, and parentheses surrounding the matrix. When this alternate form of input is used, the number of elements in each row must equal the number of dependent variables. Although these combinations actually represent the columns of the **M** matrix, they are displayed by rows.

When you include an M= specification, the analysis requested in the MANOVA statement is carried out for the variables defined by the equations in the specification, not the original dependent variables. If you omit the M= option, the analysis is performed for the original dependent variables in the MODEL statement.

If an M= specification is included without either the MNAMES= or the PREFIX= option, the variables are labeled MVAR1, MVAR2, and so forth by default.

For further information, see the section "Multivariate Analysis of Variance" on page 4247 in Chapter 53, "The GLM Procedure."

MNAMES=*names*

provides names for the variables defined by the equations in the M= specification. Names in the list correspond to the M= equations or the rows of the **M** matrix (as it is entered).

PREFIX=*name*

is an alternative means of identifying the transformed variables defined by the M= specification. For example, if you specify PREFIX=DIFF, the transformed variables are labeled DIFF1, DIFF2, and so forth.

Detail Options

You can specify the following *options* in the MANOVA statement after a slash as *detail-options*:

CANONICAL

produces a canonical analysis of the **H** and **E** matrices (transformed by the **M** matrix, if specified) instead of the default display of characteristic roots and vectors.

MSTAT=FAPPROX | EXACT

specifies the method of evaluating the multivariate test statistics. The default is MSTAT=FAPPROX, which specifies that the multivariate tests are evaluated by using the usual approximations based on the F distribution, as discussed in the "Multivariate Tests" section in Chapter 4, "Introduction to Regression Procedures." Alternatively, you can specify MSTAT=EXACT to compute exact p-values for three of the four tests (Wilks' lambda, the Hotelling-Lawley trace, and Roy's greatest root) and an improved F-approximation for the fourth (Pillai's trace). While MSTAT=EXACT provides better control of the significance probability for the tests, especially for Roy's Greatest Root, computations for the exact p-values can be appreciably more demanding, and are in fact infeasible for large problems (many dependent variables). Thus, although MSTAT=EXACT is more accurate for most data, it is not the default method. For more information about the results of MSTAT=EXACT, see the section "Multivariate Analysis of Variance" on page 4247 in Chapter 53, "The GLM Procedure."

ORTH

requests that the transformation matrix in the M= specification of the MANOVA statement be orthonormalized by rows before the analysis.

PRINTE

displays the error SSCP matrix **E**. If the **E** matrix is the error SSCP (residual) matrix from the analysis, the partial correlations of the dependent variables given the independent variables are also produced.

For example, the statement

```
manova / printe;
```

displays the error SSCP matrix and the partial correlation matrix computed from the error SSCP matrix.

PRINTH

displays the hypothesis SSCP matrix **H** associated with each effect specified by the H= specification.

SUMMARY

produces analysis-of-variance tables for each dependent variable. When no **M** matrix is specified, a table is produced for each original dependent variable from the MODEL statement; with an **M** matrix other than the identity, a table is produced for each transformed variable defined by the **M** matrix.

Examples

The following statements give several examples of using a MANOVA statement.

```
proc anova;
   class A B;
   model Y1-Y5=A B(A);
   manova h=A e=B(A) / printh printe;
   manova h=B(A) / printe;
   manova h=A e=B(A) m=Y1-Y2,Y2-Y3,Y3-Y4,Y4-Y5
          prefix=diff;

   manova h=A e=B(A) m=(1 -1  0  0  0,
                        0  1 -1  0  0,
                        0  0  1 -1  0,
                        0  0  0  1 -1) prefix=diff;
run;
```

The first MANOVA statement specifies A as the hypothesis effect and B(A) as the error effect. As a result of the PRINTH option, the procedure displays the hypothesis SSCP matrix associated with the A effect; and, as a result of the PRINTE option, the procedure displays the error SSCP matrix associated with the B(A) effect.

The second MANOVA statement specifies B(A) as the hypothesis effect. Since no error effect is specified, PROC ANOVA uses the error SSCP matrix from the analysis as the **E** matrix. The PRINTE option displays this **E** matrix. Since the **E** matrix is the error SSCP matrix from the analysis, the partial correlation matrix computed from this matrix is also produced.

The third MANOVA statement requests the same analysis as the first MANOVA statement, but the analysis is carried out for variables transformed to be successive differences between the original dependent variables. The PREFIX=DIFF specification labels the transformed variables as DIFF1, DIFF2, DIFF3, and DIFF4.

Finally, the fourth MANOVA statement has the identical effect as the third, but it uses an alternative form of the M= specification. Instead of specifying a set of equations, the fourth MANOVA statement specifies rows of a matrix of coefficients for the five dependent variables.

As a second example of the use of the M= specification, consider the following:

```
proc anova;
   class group;
   model dose1-dose4=group / nouni;
   manova h = group
          m = -3*dose1 -   dose2 +   dose3 + 3*dose4,
                 dose1 -   dose2 -   dose3 +   dose4,
                -dose1 + 3*dose2 - 3*dose3 +   dose4
          mnames = Linear Quadratic Cubic
          / printe;
run;
```

The M= specification gives a transformation of the dependent variables dose1 through dose4 into orthogonal polynomial components, and the MNAMES= option labels the transformed variables as LINEAR, QUADRATIC, and CUBIC, respectively. Since the PRINTE option is specified and the default residual matrix is used as an error term, the partial correlation matrix of the orthogonal polynomial components is also produced.

For further information, see the section "Multivariate Analysis of Variance" on page 4247 in Chapter 53, "The GLM Procedure."

MEANS Statement

MEANS *effects* < / *options* > **;**

PROC ANOVA can compute means of the dependent variables for any effect that appears on the right-hand side in the MODEL statement.

You can use any number of MEANS statements, provided that they appear after the MODEL statement. For example, suppose A and B each have two levels. Then, if you use the following statements

```
proc anova;
   class A B;
   model Y=A B A*B;
   means A B / tukey;
   means A*B;
run;
```

means, standard deviations, and Tukey's multiple comparison tests are produced for each level of the main effects A and B, and just the means and standard deviations for each of the four combinations of levels for A*B. Since multiple comparisons options apply only to main effects, the single MEANS statement

```
means A B A*B / tukey;
```

produces the same results.

Options are provided to perform multiple comparison tests for only main effects in the model. PROC ANOVA does not perform multiple comparison tests for interaction terms in the model; for multiple comparisons of interaction terms, see the LSMEANS statement in Chapter 53, "The GLM Procedure."

Table 29.4 summarizes the *options* available in the MEANS statement.

Table 29.4 Options Available in the MEANS Statement

Option	Description
Perform Multiple Comparison Tests	
BON	Performs Bonferroni t tests of differences between means for all main effect means
DUNCAN	Performs Duncan's multiple range test on all main effect means
DUNNETT	Performs Dunnett's two-tailed t test
DUNNETTL	Performs Dunnett's one-tailed t test, testing if any treatment is significantly less than the control
DUNNETTU	Performs Dunnett's one-tailed t test, testing if any treatment is significantly greater than the control
GABRIEL	Performs Gabriel's multiple-comparison procedure on all main effect means
REGWQ	Performs the Ryan-Einot-Gabriel-Welsch multiple range test
SCHEFFE	Performs Scheffé's multiple-comparison procedure
SIDAK	Performs pairwise t tests on differences between means with levels adjusted according to Sidak's inequality
SMM or GT2	Performs pairwise comparisons based on the studentized maximum modulus and Sidak's uncorrelated-t inequality
SNK	Performs the Student-Newman-Keuls multiple range test
T or LSD	Performs pairwise t tests
TUKEY	Performs Tukey's studentized range test (HSD)
WALLER	Performs the Waller-Duncan k-ratio t test
Specify Additional Details for Multiple Comparison Tests	
ALPHA=	Specifies the level of significance for comparisons among the means.
CLDIFF	Presents results from options as confidence intervals
CLM	Options as intervals for the mean of each level of the variables specified
E=	Specifies the error mean square used in the multiple comparisons
KRATIO=	Specifies the Type 1/Type 2 error seriousness ratio for the Waller-Duncan test
LINES	Presents results of options by listing the means in descending order and indicating nonsignificant subsets by line segments
LINESTABLE	Displays the results of the LINES option as a table
NOSORT	Prevents the means from being sorted into descending order
Test for Homogeneity of Variances	
HOVTEST	Requests a homogeneity of variance test
Compensate for Heterogeneous Variances	
WELCH	Requests the Welch (1951) variance-weighted one-way ANOVA

Descriptions of these *options* follow. For a further discussion of these *options*, see the section "Multiple Comparisons" on page 4231 in Chapter 53, "The GLM Procedure."

ALPHA=*p*

specifies the level of significance for comparisons among the means. By default, ALPHA=0.05. You can specify any value greater than 0 and less than 1.

BON

performs Bonferroni *t* tests of differences between means for all main effect means in the MEANS statement. See the CLDIFF and LINES options, which follow, for a discussion of how the procedure displays results.

CLDIFF

presents results of the BON, GABRIEL, SCHEFFE, SIDAK, SMM, GT2, T, LSD, and TUKEY options as confidence intervals for all pairwise differences between means, and the results of the DUNNETT, DUNNETTU, and DUNNETTL options as confidence intervals for differences with the control. The CLDIFF option is the default for unequal cell sizes unless the DUNCAN, REGWQ, SNK, or WALLER option is specified.

CLM

presents results of the BON, GABRIEL, SCHEFFE, SIDAK,SMM, T, and LSD options as intervals for the mean of each level of the variables specified in the MEANS statement. For all options except GABRIEL, the intervals are confidence intervals for the true means. For the GABRIEL option, they are *comparison intervals* for comparing means pairwise: in this case, if the intervals corresponding to two means overlap, the difference between them is insignificant according to Gabriel's method.

DUNCAN

performs Duncan's multiple range test on all main effect means given in the MEANS statement. See the LINES option for a discussion of how the procedure displays results.

DUNNETT < (*formatted-control-values***) >**

performs Dunnett's two-tailed *t* test, testing if any treatments are significantly different from a single control for all main effects means in the MEANS statement.

To specify which level of the effect is the control, enclose the formatted value in quotes in parentheses after the *keyword*. If more than one effect is specified in the MEANS statement, you can use a list of control values within the parentheses. By default, the first level of the effect is used as the control. For example,

```
means a / dunnett('CONTROL');
```

where CONTROL is the formatted control value of A. As another example,

```
means a b c / dunnett('CNTLA' 'CNTLB' 'CNTLC');
```

where CNTLA, CNTLB, and CNTLC are the formatted control values for A, B, and C, respectively.

DUNNETTL < (*formatted-control-value***) >**

performs Dunnett's one-tailed *t* test, testing if any treatment is significantly less than the control. Control level information is specified as described previously for the DUNNETT option.

DUNNETTU < (*formatted-control-value*) >

performs Dunnett's one-tailed *t* test, testing if any treatment is significantly greater than the control. Control level information is specified as described previously for the DUNNETT option.

E=*effect*

specifies the error mean square used in the multiple comparisons. By default, PROC ANOVA uses the residual Mean Square (MS). The effect specified with the E= option must be a term in the model; otherwise, the procedure uses the residual MS.

GABRIEL

performs Gabriel's multiple-comparison procedure on all main effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

GT2

see the SMM option.

HOVTEST

HOVTEST=BARTLETT

HOVTEST=BF

HOVTEST=LEVENE < (TYPE=ABS | SQUARE) >

HOVTEST=OBRIEN < (W=*number*) >

requests a homogeneity of variance test for the groups defined by the MEANS effect. You can optionally specify a particular test; if you do not specify a test, Levene's test (Levene 1960) with TYPE=SQUARE is computed. Note that this option is ignored unless your MODEL statement specifies a simple one-way model.

The HOVTEST=BARTLETT option specifies Bartlett's test (Bartlett 1937), a modification of the normal-theory likelihood ratio test.

The HOVTEST=BF option specifies Brown and Forsythe's variation of Levene's test (Brown and Forsythe 1974).

The HOVTEST=LEVENE option specifies Levene's test (Levene 1960), which is widely considered to be the standard homogeneity of variance test. You can use the TYPE= option in parentheses to specify whether to use the absolute residuals (TYPE=ABS) or the squared residuals (TYPE=SQUARE) in Levene's test. The default is TYPE=SQUARE.

The HOVTEST=OBRIEN option specifies O'Brien's test (O'Brien 1979), which is basically a modification of HOVTEST=LEVENE(TYPE=SQUARE). You can use the W= option in parentheses to tune the variable to match the suspected kurtosis of the underlying distribution. By default, W=0.5, as suggested by O'Brien (1979, 1981).

See the section "Homogeneity of Variance in One-Way Models" on page 4243 in Chapter 53, "The GLM Procedure," for more details on these methods. Example 53.10 in the same chapter illustrates the use of the HOVTEST and WELCH options in the MEANS statement in testing for equal group variances.

KRATIO=*value*

specifies the Type 1/Type 2 error seriousness ratio for the Waller-Duncan test. Reasonable values for KRATIO are 50, 100, and 500, which roughly correspond for the two-level case to ALPHA levels of 0.1, 0.05, and 0.01. By default, the procedure uses the default value of 100.

LINES

presents results of the BON, DUNCAN, GABRIEL, REGWQ, SCHEFFE, SIDAK,SMM, GT2, SNK, T, LSD TUKEY, and WALLER options by listing the means in descending order and indicating nonsignificant subsets by line segments beside the corresponding means. The LINES option is appropriate for equal cell sizes, for which it is the default. The LINES option is also the default if the DUNCAN, REGWQ, SNK, or WALLER option is specified, or if there are only two cells of unequal size. The LINES option cannot be used in combination with the DUNNETT, DUNNETTL, or DUNNETTU option.

If ODS Graphics is enabled, then the LINES display is produced in graphical form by default. To display the results in tabular form, use the LINESTABLE option.

In addition, PROC ANOVA restricts the number of overlapping groups of means to 24. If a mean belongs to more than 24 groups, the procedure issues an error message. You can either reduce the number of levels of the variable or use a multiple comparison test that allows the CLDIFF option rather than the LINES option.

NOTE: If the cell sizes are unequal, the harmonic mean of the cell sizes is used to compute the critical ranges. This approach is reasonable if the cell sizes are not too different, but it can lead to liberal tests if the cell sizes are highly disparate. In this case, you should not use the LINES option for displaying multiple comparisons results; use the CLDIFF options instead.

LINESTABLE

presents the results of the LINES option as a table.

LSD

see the T option.

NOSORT

prevents the means from being sorted into descending order when the CLDIFF or CLM option is specified.

REGWQ

performs the Ryan-Einot-Gabriel-Welsch multiple range test on all main effect means in the MEANS statement. See the LINES option for a discussion of how the procedure displays results.

SCHEFFE

performs Scheffé's multiple-comparison procedure on all main effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

SIDAK

performs pairwise t tests on differences between means with levels adjusted according to Sidak's inequality for all main effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

SMM

GT2

performs pairwise comparisons based on the studentized maximum modulus and Sidak's uncorrelated-*t* inequality, yielding Hochberg's GT2 method when sample sizes are unequal, for all main effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

SNK

performs the Student-Newman-Keuls multiple range test on all main effect means in the MEANS statement. See the LINES option for a discussion of how the procedure displays results.

T

LSD

performs pairwise *t* tests, equivalent to Fisher's least-significant-difference test in the case of equal cell sizes, for all main effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

TUKEY

performs Tukey's studentized range test (HSD) on all main effect means in the MEANS statement. (When the group sizes are different, this is the Tukey-Kramer test.) See the CLDIFF and LINES options for discussions of how the procedure displays results.

WALLER

performs the Waller-Duncan *k*-ratio *t* test on all main effect means in the MEANS statement. See the KRATIO= option for information about controlling details of the test, and see the LINES option for a discussion of how the procedure displays results.

WELCH

requests Welch's (1951) variance-weighted one-way ANOVA. This alternative to the usual analysis of variance for a one-way model is robust to the assumption of equal within-group variances. This option is ignored unless your MODEL statement specifies a simple one-way model.

Note that using the WELCH option merely produces one additional table consisting of Welch's ANOVA. It does not affect all of the other tests displayed by the ANOVA procedure, which still require the assumption of equal variance for exact validity.

See the section "Homogeneity of Variance in One-Way Models" on page 4243 in Chapter 53, "The GLM Procedure," for more details on Welch's ANOVA. Example 53.10 in the same chapter illustrates the use of the HOVTEST and WELCH options in the MEANS statement in testing for equal group variances.

MODEL Statement

MODEL *dependents* **=** *effects* < / *options* > **;**

The MODEL statement names the dependent variables and independent effects. The syntax of effects is described in the section "Specification of Effects" on page 1147. For any model effect involving classification variables (interactions as well as main effects), the number of levels cannot exceed 32,767. If no independent effects are specified, only an intercept term is fit. This tests the hypothesis that the mean of the dependent variable is zero. All variables in effects that you specify in the MODEL statement must appear in the CLASS statement because PROC ANOVA does not allow for continuous effects.

You can specify the following *options* in the MODEL statement; they must be separated from the list of independent effects by a slash.

EFFECTSIZE

adds measures of effect size to each analysis of variance table displayed by the procedure, except for those displayed by the MANOVA, REPEATED, and TEST statements. The effect size measures include the intraclass correlation and both estimates and confidence intervals for the noncentrality for the F test, as well as for the semipartial and partial correlation ratios. For more information about the computation and interpretation of these measures, see the section "Effect Size Measures for F Tests in GLM" on page 4220 in Chapter 53, "The GLM Procedure."

INTERCEPT

INT

displays the hypothesis tests associated with the intercept as an effect in the model. By default, the procedure includes the intercept in the model but does not display associated tests of hypotheses. Except for producing the uncorrected total SS instead of the corrected total SS, the INT option is ignored when you use an ABSORB statement.

NOUNI

suppresses the display of univariate statistics. You typically use the NOUNI option with a multivariate or repeated measures analysis of variance when you do not need the standard univariate output. The NOUNI option in a MODEL statement does not affect the univariate output produced by the REPEATED statement.

REPEATED Statement

REPEATED *factor-specification* < / *options* > **;**

When values of the dependent variables in the MODEL statement represent repeated measurements on the same experimental unit, the REPEATED statement enables you to test hypotheses about the measurement factors (often called *within-subject factors*), as well as the interactions of within-subject factors with independent variables in the MODEL statement (often called *between-subject factors*). The REPEATED statement provides multivariate and univariate tests as well as hypothesis tests for a variety of single-degree-of-freedom contrasts. There is no limit to the number of within-subject factors that can be specified. For more details, see the section "Repeated Measures Analysis of Variance" on page 4249 in Chapter 53, "The GLM Procedure."

The REPEATED statement is typically used for handling repeated measures designs with one repeated response variable. Usually, the variables on the left-hand side of the equation in the MODEL statement represent one repeated response variable.

This does not mean that only one factor can be listed in the REPEATED statement. For example, one repeated response variable (hemoglobin count) might be measured 12 times (implying variables Y1 to Y12 on the left-hand side of the equal sign in the MODEL statement), with the associated within-subject factors treatment and time (implying two factors listed in the REPEATED statement). See the section "Examples" on page 1145 for an example of how PROC ANOVA handles this case.

Designs with two or more repeated response variables can, however, be handled with the IDENTITY transformation; see Example 53.9 in Chapter 53, "The GLM Procedure," for an example of analyzing a doubly-multivariate repeated measures design.

When a REPEATED statement appears, the ANOVA procedure enters a multivariate mode of handling missing values. If any values for variables corresponding to each combination of the within-subject factors are missing, the observation is excluded from the analysis.

The simplest form of the REPEATED statement requires only a *factor-name*. With two repeated factors, you must specify the *factor-name* and number of levels (*levels*) for each factor. Optionally, you can specify the actual values for the levels (*level-values*), a *transformation* that defines single-degree-of freedom contrasts, and *options* for additional analyses and output. When more than one within-subject factor is specified, *factor-names* (and associated level and transformation information) must be separated by a comma in the REPEATED statement. These terms are described in the following section, "Syntax Details."

Syntax Details

Table 29.5 summarizes the *options* available in the REPEATED statement.

Table 29.5 PROC REPEATED Statement Options

Option	Description
CANONICAL	Performs a canonical analysis of the **H** and **E** matrices
MSTAT=FAPPROX	Specifies the method of evaluating the multivariate test statistics
NOM	Displays only the results of the univariate analyses
NOU	Displays only the results of the multivariate analyses
PRINTE	Displays the **E** matrix
PRINTH	Displays the **H** (SSCP) matrix
PRINTM	Displays the transformation matrices that define the contrasts
PRINTRV	Produces the characteristic roots and vectors
SUMMARY	Produces analysis-of-variance tables for each contrast
UEPSDEF=	Specifies the univariate F test adjustment

You can specify the following terms in the REPEATED statement.

factor-specification

The *factor-specification* for the REPEATED statement can include any number of individual factor specifications, separated by commas, of the following form:

factor-name levels < **(***level-values***)** > < *transformation* >

where

factor-name names a factor to be associated with the dependent variables. The name should not be the same as any variable name that already exists in the data set being analyzed and should conform to the usual conventions of SAS variable names.

When specifying more than one factor, list the dependent variables in the MODEL statement so that the within-subject factors defined in the REPEATED statement are nested; that is, the first factor defined in the REPEATED statement should be the one with values that change least frequently.

levels specifies the number of levels associated with the factor being defined. When there is only one within-subject factor, the number of levels is equal to the number of dependent variables. In this case, *levels* is optional. When more than one within-subject factor is defined, however, *levels* is required, and the product of the number of levels of all the factors must equal the number of dependent variables in the MODEL statement.

(*level-values*) specifies values that correspond to levels of a repeated-measures factor. These values are used to label output; they are also used as spacings for constructing orthogonal polynomial contrasts if you specify a POLYNOMIAL transformation. The number of level values specified must correspond to the number of levels for that factor in the REPEATED statement. Enclose the *level-values* in parentheses.

The following *transformation* keywords define single-degree-of-freedom contrasts for factors specified in the REPEATED statement. Since the number of contrasts generated is always one less than the number of levels of the factor, you have some control over which contrast is omitted from the analysis by which transformation you select. The only exception is the IDENTITY transformation; this transformation is not composed of contrasts, and it has the same degrees of freedom as the factor has levels. By default, the procedure uses the CONTRAST transformation.

CONTRAST< **(***ordinal-reference-level***)** >

generates contrasts between levels of the factor and a reference level. By default, the procedure uses the last level; you can optionally specify a reference level in parentheses after the keyword CONTRAST. The reference level corresponds to the ordinal value of the level rather than the level value specified. For example, to generate contrasts between the first level of a factor and the other levels, use

```
contrast(1)
```

HELMERT
generates contrasts between each level of the factor and the mean of subsequent levels.

IDENTITY
generates an identity transformation corresponding to the associated factor. This transformation is *not* composed of contrasts; it has n degrees of freedom for an n-level factor, instead of $n - 1$. This can be used for doubly-multivariate repeated measures.

MEAN< (*ordinal-reference-level*) >

generates contrasts between levels of the factor and the mean of all other levels of the factor. Specifying a reference level eliminates the contrast between that level and the mean. Without a reference level, the contrast involving the last level is omitted. See the CONTRAST transformation for an example.

POLYNOMIAL
generates orthogonal polynomial contrasts. Level values, if provided, are used as spacings in the construction of the polynomials; otherwise, equal spacing is assumed.

PROFILE
generates contrasts between adjacent levels of the factor.

For examples of the transformation matrices generated by these contrast transformations, see the section "Repeated Measures Analysis of Variance" on page 4249 in Chapter 53, "The GLM Procedure."

You can specify the following *options* in the REPEATED statement after a slash:

CANONICAL
performs a canonical analysis of the **H** and **E** matrices corresponding to the transformed variables specified in the REPEATED statement.

MSTAT=FAPPROX | EXACT
specifies the method of evaluating the multivariate test statistics. The default is MSTAT=FAPPROX, which specifies that the multivariate tests are evaluated by using the usual approximations based on the F distribution, as discussed in the "Multivariate Tests" section in Chapter 4, "Introduction to Regression Procedures." Alternatively, you can specify MSTAT=EXACT to compute exact p-values for three of the four tests (Wilks' lambda, the Hotelling-Lawley trace, and Roy's greatest root) and an improved F-approximation for the fourth (Pillai's trace). While MSTAT=EXACT provides better control of the significance probability for the tests, especially for Roy's Greatest Root, computations for the exact p-values can be appreciably more demanding, and are in fact infeasible for large problems (many dependent variables). Thus, although MSTAT=EXACT is more accurate for most data, it is not the default method. For more information about the results of MSTAT=EXACT, see the section "Multivariate Analysis of Variance" on page 4247 in Chapter 53, "The GLM Procedure."

NOM
displays only the results of the univariate analyses.

NOU
displays only the results of the multivariate analyses.

PRINTE
displays the **E** matrix for each combination of within-subject factors, as well as partial correlation matrices for both the original dependent variables and the variables defined by the transformations specified in the REPEATED statement. In addition, the PRINTE option provides sphericity tests for each set of transformed variables. If the requested transformations are not orthogonal, the PRINTE option also provides a sphericity test for a set of orthogonal contrasts.

PRINTH
displays the **H** (SSCP) matrix associated with each multivariate test.

PRINTM
displays the transformation matrices that define the contrasts in the analysis. PROC ANOVA always displays the **M** matrix so that the transformed variables are defined by the rows, not the columns, of the displayed **M** matrix. In other words, PROC ANOVA actually displays $\mathbf{M}'$.

PRINTRV
produces the characteristic roots and vectors for each multivariate test.

SUMMARY
produces analysis-of-variance tables for each contrast defined by the within-subjects factors. Along with tests for the effects of the independent variables specified in the MODEL statement, a term labeled MEAN tests the hypothesis that the overall mean of the contrast is zero.

UEPSDEF=*unbiased-epsilon-definition*
specifies the type of adjustment for the univariate F test that is displayed in addition to the Greenhouse-Geisser adjustment. The default is UEPSDEF=HFL, corresponding to the corrected form of the Huynh-Feldt adjustment (Huynh and Feldt 1976; Lecoutre 1991). Other alternatives are UEPSDEF=HF, the uncorrected Huynh-Feldt adjustment (the only available method in previous releases of SAS/STAT software), and UEPSDEF=CM, the adjustment of Chi et al. (2012). See the section "Hypothesis Testing in Repeated Measures Analysis" on page 4251 in Chapter 53, "The GLM Procedure," for details about these adjustments.

Examples

When specifying more than one factor, list the dependent variables in the MODEL statement so that the within-subject factors defined in the REPEATED statement are nested; that is, the first factor defined in the REPEATED statement should be the one with values that change least frequently. For example, assume that three treatments are administered at each of four times, for a total of twelve dependent variables on each experimental unit. If the variables are listed in the MODEL statement as Y1 through Y12, then the following REPEATED statement

```
repeated trt 3, time 4;
```

implies the following structure:

	Dependent Variables											
	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10	Y11	Y12
Value of trt	1	1	1	1	2	2	2	2	3	3	3	3
Value of time	1	2	3	4	1	2	3	4	1	2	3	4

The REPEATED statement always produces a table like the preceding one.

For more information about repeated measures analysis and about using the REPEATED statement, see the section "Repeated Measures Analysis of Variance" on page 4249 in Chapter 53, "The GLM Procedure."

TEST Statement

TEST < **H=** *effects* > **E=** *effect* **;**

Although an F value is computed for all SS in the analysis by using the residual MS as an error term, you can request additional F tests that use other effects as error terms. You need a TEST statement when a nonstandard error structure (as in a split plot) exists.

CAUTION: The ANOVA procedure does not check any of the assumptions underlying the F statistic. When you specify a TEST statement, you assume sole responsibility for the validity of the F statistic produced. To help validate a test, you might want to use the GLM procedure with the RANDOM statement and inspect the expected mean squares. In the GLM procedure, you can also use the TEST option in the RANDOM statement.

You can use as many TEST statements as you want, provided that they appear after the MODEL statement.

You can specify the following terms in the TEST statement.

H=*effects*
specifies which effects in the preceding model are to be used as hypothesis (numerator) effects.

E=*effect*
specifies one, and only one, effect to use as the error (denominator) term. The **E=** specification is required.

The following example uses two TEST statements and is appropriate for analyzing a split-plot design.

```
proc anova;
   class a b c;
   model y=a|b(a)|c;
   test h=a e=b(a);
   test h=c a*c e=b*c(a);
run;
```

Details: ANOVA Procedure

Specification of Effects

In SAS analysis-of-variance procedures, the variables that identify levels of the classifications are called *classification variables*, and they are declared in the CLASS statement. Classification variables are also called *categorical*, *qualitative*, *discrete*, or *nominal variables*. The values of a classification variable are called *levels*. Classification variables can be either numeric or character. This is in contrast to the *response* (or *dependent*) *variables*, which are continuous. Response variables must be numeric.

The analysis-of-variance model specifies *effects*, which are combinations of classification variables used to explain the variability of the dependent variables in the following manner:

- Main effects are specified by writing the variables by themselves in the CLASS statement: A B C. Main effects used as independent variables test the hypothesis that the mean of the dependent variable is the same for each level of the factor in question, ignoring the other independent variables in the model.
- Crossed effects (interactions) are specified by joining the CLASS variables with asterisks in the MODEL statement: A*B A*C A*B*C. Interaction terms in a model test the hypothesis that the effect of a factor does not depend on the levels of the other factors in the interaction.
- Nested effects are specified by following a main effect or crossed effect with a CLASS variable or list of CLASS variables enclosed in parentheses in the MODEL statement. The main effect or crossed effect is nested within the effects listed in parentheses: B(A) C*D(A B). Nested effects test hypotheses similar to interactions, but the levels of the nested variables are not the same for every combination within which they are nested.

The general form of an effect can be illustrated by using the CLASS variables A, B, C, D, E, and F:

$$A * B * C(D\ E\ F)$$

The crossed list should come first, followed by the nested list in parentheses. Note that no asterisks appear within the nested list or immediately before the left parenthesis.

Main Effects Models

For a three-factor main effects model with A, B, and C as the factors and Y as the dependent variable, the necessary statements are

```
proc anova;
   class A B C;
   model Y=A B C;
run;
```

Models with Crossed Factors

To specify interactions in a factorial model, join effects with asterisks as described previously. For example, these statements specify a complete factorial model, which includes all the interactions:

```
proc anova;
   class A B C;
   model Y=A B C A*B A*C B*C A*B*C;
run;
```

Bar Notation

You can shorten the specifications of a full factorial model by using bar notation. For example, the preceding statements can also be written

```
proc anova;
   class A B C;
   model Y=A|B|C;
run;
```

When the bar (|) is used, the expression on the right side of the equal sign is expanded from left to right by using the equivalents of rules 2–4 given in Searle (1971, p. 390). The variables on the right- and left-hand sides of the bar become effects, and the cross of them becomes an effect. Multiple bars are permitted. For instance, A | B | C is evaluated as follows:

$$
\begin{aligned}
\text{A | B | C} &\rightarrow \{\text{A | B}\}\text{ | C} \\
&\rightarrow \{\text{A B A*B}\}\text{ | C} \\
&\rightarrow \text{A B A*B C A*C B*C A*B*C}
\end{aligned}
$$

You can also specify the maximum number of variables involved in any effect that results from bar evaluation by specifying that maximum number, preceded by an @ sign, at the end of the bar effect. For example, the specification A | B | C@2 results in only those effects that contain two or fewer variables; in this case, A B A*B C A*C and B*C.

The following table gives more examples of using the bar and at operators.

A | C(B)	is equivalent to	A C(B) A*C(B)
A(B) | C(B)	is equivalent to	A(B) C(B) A*C(B)
A(B) | B(D E)	is equivalent to	A(B) B(D E)
A | B(A) | C	is equivalent to	A B(A) C A*C B*C(A)
A | B(A) | C@2	is equivalent to	A B(A) C A*C
A | B | C | D@2	is equivalent to	A B A*B C A*C B*C D A*D B*D C*D

Consult the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure," for further details on bar notation.

Nested Models

Write the effect that is nested within another effect first, followed by the other effect in parentheses. For example, if A and B are main effects and C is nested within A and B (that is, the levels of C that are observed are not the same for each combination of A and B), the statements for PROC ANOVA are

```
proc anova;
   class A B C;
   model y=A B C(A B);
run;
```

The identity of a level is viewed within the context of the level of the containing effects. For example, if City is nested within State, then the identity of City is viewed within the context of State.

The distinguishing feature of a nested specification is that nested effects never appear as main effects. Another way of viewing nested effects is that they are effects that pool the main effect with the interaction of the nesting variable.

See the "Automatic Pooling" section, which follows.

Models Involving Nested, Crossed, and Main Effects

Asterisks and parentheses can be combined in the MODEL statement for models involving nested and crossed effects:

```
proc anova;
   class A B C;
   model Y=A B(A) C(A) B*C(A);
run;
```

Automatic Pooling

In line with the general philosophy of the GLM procedure, there is no difference between the statements

```
model Y=A B(A);
```

and

```
model Y=A A*B;
```

The effect B becomes a nested effect by virtue of the fact that it does not occur as a main effect. If B is not written as a main effect in addition to participating in A*B, then the sum of squares that is associated with B is pooled into A*B.

This feature allows the automatic pooling of sums of squares. If an effect is omitted from the model, it is automatically pooled with all the higher-level effects containing the CLASS variables in the omitted effect (or within-error). This feature is most useful in split-plot designs.

Using PROC ANOVA Interactively

PROC ANOVA can be used interactively. After you specify a model in a MODEL statement and run PROC ANOVA with a RUN statement, a variety of statements (such as MEANS, MANOVA, TEST, and REPEATED) can be executed without PROC ANOVA recalculating the model sum of squares.

The section "Syntax: ANOVA Procedure" on page 1126 describes which statements can be used interactively. You can execute these interactive statements individually or in groups by following the single statement or group of statements with a RUN statement. Note that the MODEL statement cannot be repeated; the ANOVA procedure allows only one MODEL statement.

If you use PROC ANOVA interactively, you can end the procedure with a DATA step, another PROC step, an ENDSAS statement, or a QUIT statement. The syntax of the QUIT statement is

```
quit;
```

When you use PROC ANOVA interactively, additional RUN statements do not end the procedure but tell PROC ANOVA to execute additional statements.

When a WHERE statement is used with PROC ANOVA, it should appear before the first RUN statement. The WHERE statement enables you to select only certain observations for analysis without using a subsetting DATA step. For example, the statement **`where group ne 5`** omits observations with GROUP=5 from the analysis. See *SAS DATA Step Statements: Reference* for details about this statement.

When a BY statement is used with PROC ANOVA, interactive processing is not possible; that is, once the first RUN statement is encountered, processing proceeds for each BY group in the data set, and no further statements are accepted by the procedure.

Interactivity is also disabled when there are different patterns of missing values among the dependent variables. For details, see the section "Missing Values," which follows.

Missing Values

For an analysis involving one dependent variable, PROC ANOVA uses an observation if values are nonmissing for that dependent variable and for all the variables used in independent effects.

For an analysis involving multiple dependent variables without the MANOVA or REPEATED statement, or without the MANOVA option in the PROC ANOVA statement, a missing value in one dependent variable does not eliminate the observation from the analysis of other nonmissing dependent variables. For an analysis with the MANOVA or REPEATED statement, or with the MANOVA option in the PROC ANOVA statement, the ANOVA procedure requires values for all dependent variables to be nonmissing for an observation before the observation can be used in the analysis.

During processing, PROC ANOVA groups the dependent variables by their pattern of missing values across observations so that sums and cross products can be collected in the most efficient manner.

If your data have different patterns of missing values among the dependent variables, interactivity is disabled. This could occur when some of the variables in your data set have missing values and either of the following conditions obtain:

- You do not use the MANOVA option in the PROC ANOVA statement.

- You do not use a MANOVA or REPEATED statement before the first RUN statement.

Output Data Set

The OUTSTAT= option in the PROC ANOVA statement produces an output data set that contains the following:

- the BY variables, if any
- _TYPE_, a new character variable. This variable has the value 'ANOVA' for observations corresponding to sums of squares; it has the value 'CANCORR', 'STRUCTUR', or 'SCORE' if a canonical analysis is performed through the MANOVA statement and no M= matrix is specified.
- _SOURCE_, a new character variable. For each observation in the data set, _SOURCE_ contains the name of the model effect from which the corresponding statistics are generated.
- _NAME_, a new character variable. The variable _NAME_ contains the name of one of the dependent variables in the model or, in the case of canonical statistics, the name of one of the canonical variables (CAN1, CAN2, and so on).
- four new numeric variables, SS, DF, F, and PROB, containing sums of squares, degrees of freedom, F values, and probabilities, respectively, for each model or contrast sum of squares generated in the analysis. For observations resulting from canonical analyses, these variables have missing values.
- if there is more than one dependent variable, then variables with the same names as the dependent variables represent
  - for _TYPE_='ANOVA', the crossproducts of the hypothesis matrices
  - for _TYPE_='CANCORR', canonical correlations for each variable
  - for _TYPE_='STRUCTUR', coefficients of the total structure matrix
  - for _TYPE_='SCORE', raw canonical score coefficients

The output data set can be used to perform special hypothesis tests (for example, with the IML procedure in SAS/IML software), to reformat output, to produce canonical variates (through the SCORE procedure), or to rotate structure matrices (through the FACTOR procedure).

Computational Method

Let $\mathbf{X}$ represent the $n \times p$ design matrix. The columns of $\mathbf{X}$ contain only 0s and 1s. Let $\mathbf{Y}$ represent the $n \times 1$ vector of dependent variables.

In the GLM procedure, $\mathbf{X}'\mathbf{X}$, $\mathbf{X}'\mathbf{Y}$, and $\mathbf{Y}'\mathbf{Y}$ are formed in main storage. However, in the ANOVA procedure, only the diagonals of $\mathbf{X}'\mathbf{X}$ are computed, along with $\mathbf{X}'\mathbf{Y}$ and $\mathbf{Y}'\mathbf{Y}$. Thus, PROC ANOVA saves a considerable amount of storage as well as time. The memory requirements for PROC ANOVA are asymptotically linear functions of n^2 and nr, where n is the number of dependent variables and r the number of independent parameters.

The elements of $\mathbf{X}'\mathbf{Y}$ are cell totals, and the diagonal elements of $\mathbf{X}'\mathbf{X}$ are cell frequencies. Since PROC ANOVA automatically pools omitted effects into the next higher-level effect containing the names of the omitted effect (or within-error), a slight modification to the rules given by Searle (1971, p. 389) is used.

1. PROC ANOVA computes the sum of squares for each effect as if it is a main effect. In other words, for each effect, PROC ANOVA squares each cell total and divides by its cell frequency. The procedure then adds these quantities together and subtracts the correction factor for the mean (total squared over N).
2. For each effect involving two CLASS variable names, PROC ANOVA subtracts the SS for any main effect with a name that is contained in the two-factor effect.
3. For each effect involving three CLASS variable names, PROC ANOVA subtracts the SS for all main effects and two-factor effects with names that are contained in the three-factor effect. If effects involving four or more CLASS variable names are present, the procedure continues this process.

Displayed Output

PROC ANOVA first displays a table that includes the following:

- the name of each variable in the CLASS statement
- the number of different values or Levels of the CLASS variables
- the Values of the CLASS variables
- the Number of observations in the data set and the number of observations excluded from the analysis because of missing values, if any

PROC ANOVA then displays an analysis-of-variance table for each dependent variable in the MODEL statement. This table breaks down the Total Sum of Squares for the dependent variable into the portion attributed to the Model and the portion attributed to Error. It also breaks down the Mean Square term, which is the Sum of Squares divided by the degrees of freedom (DF). The analysis-of-variance table also lists the following:

- the Mean Square for Error (MSE), which is an estimate of σ^2, the variance of the true errors

- the F Value, which is the ratio produced by dividing the Mean Square for the Model by the Mean Square for Error. It tests how well the model as a whole (adjusted for the mean) accounts for the dependent variable's behavior. This F test is a test of the null hypothesis that all parameters except the intercept are zero.
- the significance probability associated with the F statistic, labeled "Pr > F"
- R-Square, R^2, which measures how much variation in the dependent variable can be accounted for by the model. The R square statistic, which can range from 0 to 1, is the ratio of the sum of squares for the model divided by the sum of squares for the corrected total. In general, the larger the R square value, the better the model fits the data.
- C.V., the coefficient of variation, which is often used to describe the amount of variation in the population. The C.V. is 100 times the standard deviation of the dependent variable divided by the Mean. The coefficient of variation is often a preferred measure because it is unitless.
- Root MSE, which estimates the standard deviation of the dependent variable. Root MSE is computed as the square root of Mean Square for Error, the mean square of the error term.
- the Mean of the dependent variable

For each effect (or source of variation) in the model, PROC ANOVA then displays the following:

- DF, degrees of freedom
- Anova SS, the sum of squares, and the associated Mean Square
- the F Value for testing the hypothesis that the group means for that effect are equal
- Pr > F, the significance probability value associated with the F Value

When you specify a TEST statement, PROC ANOVA displays the results of the requested tests. When you specify a MANOVA statement and the model includes more than one dependent variable, PROC ANOVA produces these additional statistics:

- the characteristic roots and vectors of $\mathbf{E}^{-1}\mathbf{H}$ for each $\mathbf{H}$ matrix
- the Hotelling-Lawley trace
- Pillai's trace
- Wilks' lambda
- Roy's greatest root

See Example 53.6 in Chapter 53, "The GLM Procedure," for an example of the MANOVA results. These MANOVA tests are discussed in Chapter 4, "Introduction to Regression Procedures."

ODS Table Names

PROC ANOVA assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 29.6. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 29.6 ODS Tables Produced by PROC ANOVA

ODS Table Name	Description	Statement / Option
AltErrTests	ANOVA tests with error other than MSE	TEST E=
Bartlett	Bartlett's homogeneity of variance test	MEANS / HOVTEST=BARTLETT
CanAnalysis	Canonical analysis	(MANOVA or REPEATED) / CANONICAL
CanCoef	Canonical coefficients	(MANOVA or REPEATED) / CANONICAL
CanStructure	Canonical structure	(MANOVA or REPEATED) / CANONICAL
CharStruct	Characteristic roots and vectors	(MANOVA / not CANONICAL) or (REPEATED / PRINTRV)
ClassLevels	Classification variable levels	CLASS statement
CLDiffs	Multiple comparisons of pairwise differences	MEANS / CLDIFF or DUNNETT or (Unequal cells and not LINES)
CLDiffsInfo	Information for multiple comparisons of pairwise differences	MEANS / CLDIFF or DUNNETT or (Unequal cells and not LINES)
CLMeans	Multiple comparisons of means with confidence/comparison interval	MEANS / CLM with (BON or GABRIEL or SCHEFFE or SIDAK or SMM or T or LSD)
CLMeansInfo	Information for multiple comparisons of means with confidence/comparison interval	MEANS / CLM
DependentInfo	Simultaneously analyzed dependent variables	Default when there are multiple dependent variables with different patterns of missing values
Epsilons	Greenhouse-Geisser and Huynh-Feldt epsilons	REPEATED statement
ErrorSSCP	Error SSCP matrix	(MANOVA or REPEATED) / PRINTE
FitStatistics	R-square, coefficient of variation, root MSE, and dependent mean	Default
HOVFTest	Homogeneity of variance ANOVA	MEANS / HOVTEST
HypothesisSSCP	Hypothesis SSCP matrix	(MANOVA or REPEATED) / PRINTE
MANOVATransform	Multivariate transformation matrix	MANOVA / M=

Table 29.6 *continued*

ODS Table Name	Description	Statement / Option
MCLines	Multiple comparisons LINES output	MEANS / LINES or ((DUNCAN or WALLER or SNK or REGWQ) and not (CLDIFF or CLM)) or (Equal cells and not CLDIFF)
MCLinesInfo	Information for multiple comparison LINES output	MEANS / LINES or ((DUNCAN or WALLER or SNK or REGWQ) and not (CLDIFF or CLM)) or (Equal cells and not CLDIFF)
MCLinesRange	Ranges for multiple range MC tests	MEANS / LINES or ((DUNCAN or WALLER or SNK or REGWQ) and not (CLDIFF or CLM)) or (Equal cells and not CLDIFF)
Means	Group means	MEANS statement
ModelANOVA	ANOVA for model terms	Default
MultStat	Multivariate tests	MANOVA statement
NObs	Number of observations	Default
OverallANOVA	Overall ANOVA	Default
PartialCorr	Partial correlation matrix	(MANOVA or REPEATED) / PRINTE
RepeatedLevelInfo	Correspondence between dependents and repeated measures levels	REPEATED statement
RepeatedTransform	Repeated transformation matrix	REPEATED (CONTRAST or HELMERT or MEAN or POLYNOMIAL or PROFILE)
Sphericity	Sphericity tests	REPEATED / PRINTE
Tests	Summary ANOVA for specified MANOVA H= effects	MANOVA / H= SUMMARY
Welch	Welch's ANOVA	MEANS / WELCH

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, if you specify a one-way analysis of variance model, with just one independent classification variable, or if you use a MEANS statement, then the ANOVA procedure will

produce a grouped box plot of the response values versus the classification levels. For an example of the box plot, see the section "One-Way Layout with Means Comparisons" on page 1119. Outliers are labeled by the observation number within the entire data set, including observations that are not used in the analysis. Therefore, the labels of the outliers in this box plot might differ from those in the box plot that is produced by the MEANS statement.

ODS Graph Names

PROC ANOVA produces a single graph, whose name you can use to refer to it in ODS. The name is listed in Table 29.7.

Table 29.7 ODS Graphic Produced by PROC ANOVA

ODS Graph Name	Plot Description
BoxPlot	Box plot of observed response values by classification levels

Examples: ANOVA Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 29.1: Randomized Complete Block With Factorial Treatment Structure

(View the complete code for this example (anovaex1.sas) in the example repository.)

This example uses statements for the analysis of a randomized block with two treatment factors occurring in a factorial structure. The data, from Neter, Wasserman, and Kutner (1990, p. 941), are from an experiment examining the effects of codeine and acupuncture on post-operative dental pain in male subjects. Both treatment factors have two levels. The codeine levels are a codeine capsule or a sugar capsule. The acupuncture levels are two inactive acupuncture points or two active acupuncture points. There are four distinct treatment combinations due to the factorial treatment structure. The 32 subjects are assigned to eight blocks of four subjects each based on an assessment of pain tolerance.

The data for the analysis are balanced, so PROC ANOVA is used. The data are as follows:

```
title1 'Randomized Complete Block With Two Factors';
data PainRelief;
   input PainLevel Codeine Acupuncture Relief @@;
   datalines;
1 1 1 0.0  1 2 1 0.5  1 1 2 0.6  1 2 2 1.2
2 1 1 0.3  2 2 1 0.6  2 1 2 0.7  2 2 2 1.3
3 1 1 0.4  3 2 1 0.8  3 1 2 0.8  3 2 2 1.6
4 1 1 0.4  4 2 1 0.7  4 1 2 0.9  4 2 2 1.5
```

```
5 1 1 0.6  5 2 1 1.0  5 1 2 1.5  5 2 2 1.9
6 1 1 0.9  6 2 1 1.4  6 1 2 1.6  6 2 2 2.3
7 1 1 1.0  7 2 1 1.8  7 1 2 1.7  7 2 2 2.1
8 1 1 1.2  8 2 1 1.7  8 1 2 1.6  8 2 2 2.4
;
```

The variable PainLevel is the blocking variable, and Codeine and Acupuncture represent the levels of the two treatment factors. The variable Relief is the pain relief score (the higher the score, the more relief the patient has).

The following statements invokes PROC ANOVA. The blocking variable and treatment factors appear in the CLASS statement. The bar between the treatment factors Codeine and Acupuncture adds their main effects as well as their interaction Codeine*Acupuncture to the model.

```
proc anova data=PainRelief;
   class PainLevel Codeine Acupuncture;
   model Relief = PainLevel Codeine|Acupuncture;
run;
```

The results from the analysis are shown in Output 29.1.1, Output 29.1.2, and Output 29.1.3.

Output 29.1.1 Class Level Information

Randomized Complete Block With Two Factors

The ANOVA Procedure

Class Level Information		
Class	**Levels**	**Values**
PainLevel	8	1 2 3 4 5 6 7 8
Codeine	2	1 2
Acupuncture	2	1 2

Number of Observations Read	32
Number of Observations Used	32

Output 29.1.2 ANOVA Table

Randomized Complete Block With Two Factors

The ANOVA Procedure

Dependent Variable: Relief

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	10	11.33500000	1.13350000	78.37	<.0001
Error	21	0.30375000	0.01446429		
Corrected Total	31	11.63875000			

R-Square	Coeff Var	Root MSE	Relief Mean
0.973902	10.40152	0.120268	1.156250

The Class Level Information and ANOVA table are shown in Output 29.1.1 and Output 29.1.2. The classification level information summarizes the structure of the design. It is good to check these consistently in search of errors in the DATA step. The overall *F* test is significant, indicating that the model accounts for a significant amount of variation in the dependent variable.

Output 29.1.3 Tests of Effects

Source	DF	Anova SS	Mean Square	F Value	Pr > F
PainLevel	7	5.59875000	0.79982143	55.30	<.0001
Codeine	1	2.31125000	2.31125000	159.79	<.0001
Acupuncture	1	3.38000000	3.38000000	233.68	<.0001
Codeine*Acupuncture	1	0.04500000	0.04500000	3.11	0.0923

Output 29.1.3 shows tests of the effects. The blocking effect is significant; hence, it is useful. The interaction between codeine and acupuncture is significant at the 90% level but not at the 95% level. The significance level of this test should be determined before the analysis. The main effects of both treatment factors are highly significant.

Example 29.2: Alternative Multiple Comparison Procedures

(View the complete code for this example (anovaex2.sas) in the example repository.)

The following is a continuation of the first example in the section "One-Way Layout with Means Comparisons" on page 1119. You are studying the effect of bacteria on the nitrogen content of red clover plants, and the analysis of variance shows a highly significant effect. The following statements create the data set and compute the analysis of variance as well as Tukey's multiple comparisons test for pairwise differences between bacteria strains; the results are shown in Figure 29.1, Figure 29.2, and Figure 29.4.

```
title1 'Nitrogen Content of Red Clover Plants';
data Clover;
   input Strain $ Nitrogen @@;
   datalines;
3DOK1  19.4 3DOK1  32.6 3DOK1  27.0 3DOK1  32.1 3DOK1  33.0
3DOK5  17.7 3DOK5  24.8 3DOK5  27.9 3DOK5  25.2 3DOK5  24.3
3DOK4  17.0 3DOK4  19.4 3DOK4   9.1 3DOK4  11.9 3DOK4  15.8
3DOK7  20.7 3DOK7  21.0 3DOK7  20.5 3DOK7  18.8 3DOK7  18.6
3DOK13 14.3 3DOK13 14.4 3DOK13 11.8 3DOK13 11.6 3DOK13 14.2
COMPOS 17.3 COMPOS 19.4 COMPOS 19.1 COMPOS 16.9 COMPOS 20.8
;

proc anova data=Clover;
   class Strain;
   model Nitrogen = Strain;
   means Strain / tukey;
run;
```

The interactivity of PROC ANOVA enables you to submit further MEANS statements without re-running the entire analysis. For example, the following command requests means of the Strain levels with Duncan's multiple range test and the Waller-Duncan *k*-ratio *t* test.

```
    means Strain / duncan waller;
  run;
```

Results of the Waller-Duncan k-ratio t test are shown in Output 29.2.1.

Output 29.2.1 Waller-Duncan K-ratio t Test

Nitrogen Content of Red Clover Plants

The ANOVA Procedure

Waller-Duncan K-ratio t Test for Nitrogen

Note: This test minimizes the Bayes risk under additive loss and certain other assumptions.

Kratio	100
Error Degrees of Freedom	24
Error Mean Square	11.78867
F Value	14.37
Critical Value of t	1.91873
Minimum Significant Difference	4.1665

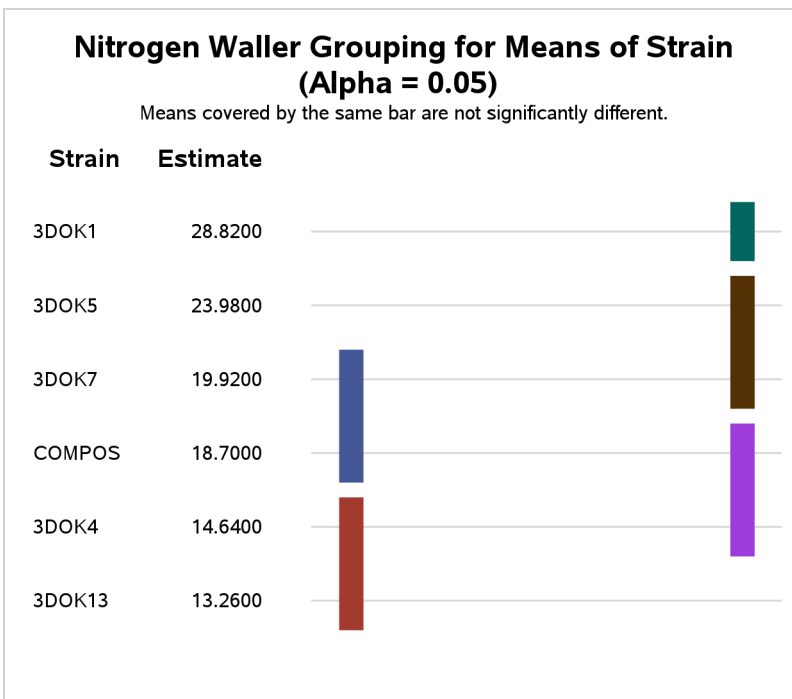

The Waller-Duncan k-ratio t test is a multiple range test. Unlike Tukey's test, this test does not operate on the principle of controlling Type I error. Instead, it compares the Type I and Type II error rates based on Bayesian principles (Steel and Torrie 1980).

The Waller Grouping column in Output 29.2.1 shows which means are significantly different. From this test, you can conclude the following:

- The mean nitrogen content for strain 3DOK1 is higher than the means for all other strains.
- The mean nitrogen content for strain 3DOK5 is higher than the means for COMPOS, 3DOK4, and 3DOK13.
- The mean nitrogen content for strain 3DOK7 is higher than the means for 3DOK4 and 3DOK13.
- The mean nitrogen content for strain COMPOS is higher than the mean for 3DOK13.
- Differences between all other means are not significant based on this sample size.

Output 29.2.2 shows the results of Duncan's multiple range test. Duncan's test is a result-guided test that compares the treatment means while controlling the comparison-wise error rate. You should use this test for planned comparisons only (Steel and Torrie 1980). The results and conclusions for this example are the same as for the Waller-Duncan *k*-ratio *t* test. This is not always the case.

Output 29.2.2 Duncan's Multiple Range Test

Nitrogen Content of Red Clover Plants

The ANOVA Procedure

Duncan's Multiple Range Test for Nitrogen

Note: This test controls the Type I comparisonwise error rate, not the experimentwise error rate.

Alpha	0.05
Error Degrees of Freedom	24
Error Mean Square	11.78867

Number of Means	**2**	**3**	**4**	**5**	**6**
Critical Range	4.482	4.707	4.852	4.954	5.031

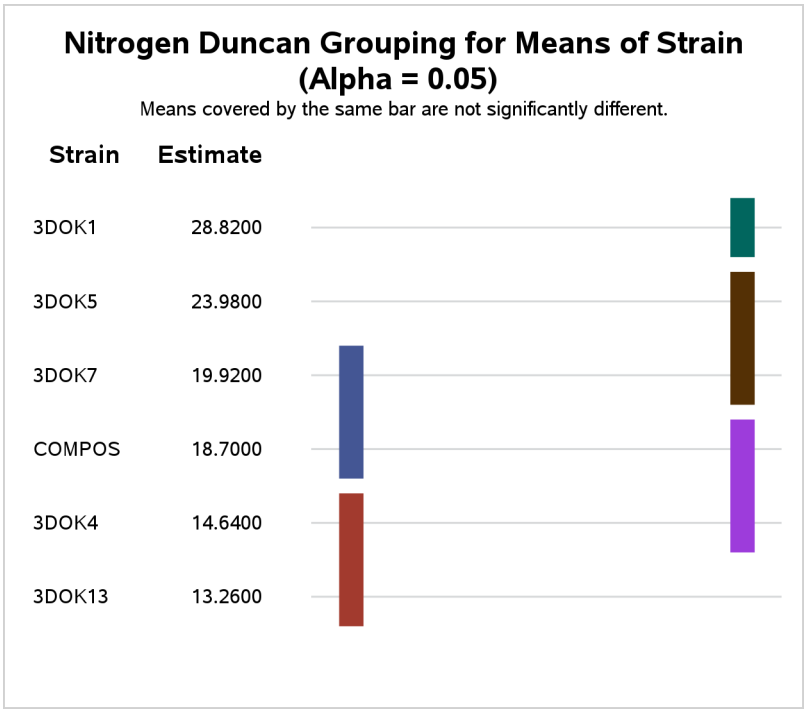

Tukey and Least Significant Difference (LSD) tests are requested with the following MEANS statement. The CLDIFF option requests confidence intervals for both tests.

```
    means Strain/ lsd tukey cldiff ;
run;
```

The LSD tests for this example are shown in Output 29.2.3, and they give the same results as the previous two multiple comparison tests. Again, this is not always the case.

Output 29.2.3 T Tests (LSD)

Nitrogen Content of Red Clover Plants

The ANOVA Procedure

t Tests (LSD) for Nitrogen

Alpha	0.05
Error Degrees of Freedom	24
Error Mean Square	11.78867
Critical Value of t	2.06390
Least Significant Difference	4.4818

Output 29.2.3 *continued*

Comparisons significant at the 0.05 level are indicated by ***.

Strain Comparison	Difference Between Means	95% Confidence Limits		
3DOK1 - 3DOK5	4.840	0.358	9.322	***
3DOK1 - 3DOK7	8.900	4.418	13.382	***
3DOK1 - COMPOS	10.120	5.638	14.602	***
3DOK1 - 3DOK4	14.180	9.698	18.662	***
3DOK1 - 3DOK13	15.560	11.078	20.042	***
3DOK5 - 3DOK1	-4.840	-9.322	-0.358	***
3DOK5 - 3DOK7	4.060	-0.422	8.542	
3DOK5 - COMPOS	5.280	0.798	9.762	***
3DOK5 - 3DOK4	9.340	4.858	13.822	***
3DOK5 - 3DOK13	10.720	6.238	15.202	***
3DOK7 - 3DOK1	-8.900	-13.382	-4.418	***
3DOK7 - 3DOK5	-4.060	-8.542	0.422	
3DOK7 - COMPOS	1.220	-3.262	5.702	
3DOK7 - 3DOK4	5.280	0.798	9.762	***
3DOK7 - 3DOK13	6.660	2.178	11.142	***
COMPOS - 3DOK1	-10.120	-14.602	-5.638	***
COMPOS - 3DOK5	-5.280	-9.762	-0.798	***
COMPOS - 3DOK7	-1.220	-5.702	3.262	
COMPOS - 3DOK4	4.060	-0.422	8.542	
COMPOS - 3DOK13	5.440	0.958	9.922	***
3DOK4 - 3DOK1	-14.180	-18.662	-9.698	***
3DOK4 - 3DOK5	-9.340	-13.822	-4.858	***
3DOK4 - 3DOK7	-5.280	-9.762	-0.798	***
3DOK4 - COMPOS	-4.060	-8.542	0.422	
3DOK4 - 3DOK13	1.380	-3.102	5.862	
3DOK13 - 3DOK1	-15.560	-20.042	-11.078	***
3DOK13 - 3DOK5	-10.720	-15.202	-6.238	***
3DOK13 - 3DOK7	-6.660	-11.142	-2.178	***
3DOK13 - COMPOS	-5.440	-9.922	-0.958	***
3DOK13 - 3DOK4	-1.380	-5.862	3.102	

If you only perform the LSD tests when the overall model F test is significant, then this is called Fisher's protected LSD test. Note that the LSD tests should be used for planned comparisons.

The TUKEY tests shown in Output 29.2.4 find fewer significant differences than the other three tests. This is not unexpected, as the TUKEY test controls the Type I experimentwise error rate. For a complete discussion of multiple comparison methods, see the section "Multiple Comparisons" on page 4231 in Chapter 53, "The GLM Procedure."

Output 29.2.4 Tukey's Studentized Range Test

Alpha	0.05
Error Degrees of Freedom	24
Error Mean Square	11.78867
Critical Value of Studentized Range	4.37265
Minimum Significant Difference	6.7142

Comparisons significant at the 0.05 level are indicated by *.**

Strain Comparison	Difference Between Means	Simultaneous 95% Confidence Limits		
3DOK1 - 3DOK5	4.840	-1.874	11.554	
3DOK1 - 3DOK7	8.900	2.186	15.614	***
3DOK1 - COMPOS	10.120	3.406	16.834	***
3DOK1 - 3DOK4	14.180	7.466	20.894	***
3DOK1 - 3DOK13	15.560	8.846	22.274	***
3DOK5 - 3DOK1	-4.840	-11.554	1.874	
3DOK5 - 3DOK7	4.060	-2.654	10.774	
3DOK5 - COMPOS	5.280	-1.434	11.994	
3DOK5 - 3DOK4	9.340	2.626	16.054	***
3DOK5 - 3DOK13	10.720	4.006	17.434	***
3DOK7 - 3DOK1	-8.900	-15.614	-2.186	***
3DOK7 - 3DOK5	-4.060	-10.774	2.654	
3DOK7 - COMPOS	1.220	-5.494	7.934	
3DOK7 - 3DOK4	5.280	-1.434	11.994	
3DOK7 - 3DOK13	6.660	-0.054	13.374	
COMPOS - 3DOK1	-10.120	-16.834	-3.406	***
COMPOS - 3DOK5	-5.280	-11.994	1.434	
COMPOS - 3DOK7	-1.220	-7.934	5.494	
COMPOS - 3DOK4	4.060	-2.654	10.774	
COMPOS - 3DOK13	5.440	-1.274	12.154	
3DOK4 - 3DOK1	-14.180	-20.894	-7.466	***
3DOK4 - 3DOK5	-9.340	-16.054	-2.626	***
3DOK4 - 3DOK7	-5.280	-11.994	1.434	
3DOK4 - COMPOS	-4.060	-10.774	2.654	
3DOK4 - 3DOK13	1.380	-5.334	8.094	
3DOK13 - 3DOK1	-15.560	-22.274	-8.846	***
3DOK13 - 3DOK5	-10.720	-17.434	-4.006	***
3DOK13 - 3DOK7	-6.660	-13.374	0.054	
3DOK13 - COMPOS	-5.440	-12.154	1.274	
3DOK13 - 3DOK4	-1.380	-8.094	5.334	

Example 29.3: Split Plot

(View the complete code for this example (anovaex3.sas) in the example repository.)

In some experiments, treatments can be applied only to groups of experimental observations rather than separately to each observation. When there are two nested groupings of the observations on the basis of treatment application, this is known as a *split plot design*. For example, in integrated circuit fabrication it is of interest to see how different manufacturing methods affect the characteristics of individual chips. However, much of the manufacturing process is applied to a relatively large wafer of material, from which many chips are made. Additionally, a chip's position within a wafer might also affect chip performance. These two groupings of chips—by wafer and by position-within-wafer—might form the *whole plots* and the *subplots*, respectively, of a split plot design for integrated circuits.

The following statements produce an analysis for a split-plot design. The CLASS statement includes the variables Block, A, and B, where B defines subplots within BLOCK*A whole plots. The MODEL statement includes the independent effects Block, A, Block*A, B, and A*B. The TEST statement asks for an F test of the A effect that uses the Block*A effect as the error term. The following statements produce Output 29.3.1 and Output 29.3.2:

```
title1 'Split Plot Design';
data Split;
   input Block 1 A 2 B 3 Response;
   datalines;
142 40.0
141 39.5
112 37.9
111 35.4
121 36.7
122 38.2
132 36.4
131 34.8
221 42.7
222 41.6
212 40.3
211 41.6
241 44.5
242 47.6
231 43.6
232 42.8
;

proc anova data=Split;
   class Block A B;
   model Response = Block A Block*A B A*B;
   test h=A e=Block*A;
run;
```

Output 29.3.1 Class Level Information and ANOVA Table

Split Plot Design

The ANOVA Procedure

Class Level Information		
Class	**Levels**	**Values**
Block	2	1 2
A	4	1 2 3 4
B	2	1 2

Number of Observations Read	16
Number of Observations Used	16

Split Plot Design

The ANOVA Procedure

Dependent Variable: Response

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	11	182.0200000	16.5472727	7.85	0.0306
Error	4	8.4300000	2.1075000		
Corrected Total	15	190.4500000			

R-Square	Coeff Var	Root MSE	Response Mean
0.955736	3.609007	1.451723	40.22500

First, notice that the overall F test for the model is significant.

Output 29.3.2 Tests of Effects

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Block	1	131.1025000	131.1025000	62.21	0.0014
A	3	40.1900000	13.3966667	6.36	0.0530
Block*A	3	6.9275000	2.3091667	1.10	0.4476
B	1	2.2500000	2.2500000	1.07	0.3599
A*B	3	1.5500000	0.5166667	0.25	0.8612

Tests of Hypotheses Using the Anova MS for Block*A as an Error Term					
Source	**DF**	**Anova SS**	**Mean Square**	**F Value**	**Pr > F**
A	3	40.19000000	13.39666667	5.80	0.0914

The effect of Block is significant. The effect of A is not significant: look at the F test produced by the TEST statement, not at the F test produced by default. Neither the B nor A*B effects are significant. The test for Block*A is irrelevant, as this is simply the main-plot error.

Example 29.4: Latin Square Split Plot

(View the complete code for this example (anovaex4.sas) in the example repository.)

The data for this example is taken from Smith (1951). A Latin square design is used to evaluate six different sugar beet varieties arranged in a six-row (Rep) by six-column (Column) square. The data are collected over two harvests. The variable Harvest then becomes a split plot on the original Latin square design for whole plots. The following statements produce Output 29.4.1, Output 29.4.2, and Output 29.4.3:

```
title1 'Sugar Beet Varieties';
title3 'Latin Square Split-Plot Design';
data Beets;
   do Harvest=1 to 2;
      do Rep=1 to 6;
         do Column=1 to 6;
            input Variety Y @;
            output;
         end;
      end;
   end;
   datalines;
3 19.1 6 18.3 5 19.6 1 18.6 2 18.2 4 18.5
6 18.1 2 19.5 4 17.6 3 18.7 1 18.7 5 19.9
1 18.1 5 20.2 6 18.5 4 20.1 3 18.6 2 19.2
2 19.1 3 18.8 1 18.7 5 20.2 4 18.6 6 18.5
4 17.5 1 18.1 2 18.7 6 18.2 5 20.4 3 18.5
5 17.7 4 17.8 3 17.4 2 17.0 6 17.6 1 17.6
3 16.2 6 17.0 5 18.1 1 16.6 2 17.7 4 16.3
6 16.0 2 15.3 4 16.0 3 17.1 1 16.5 5 17.6
1 16.5 5 18.1 6 16.7 4 16.2 3 16.7 2 17.3
2 17.5 3 16.0 1 16.4 5 18.0 4 16.6 6 16.1
4 15.7 1 16.1 2 16.7 6 16.3 5 17.8 3 16.2
5 18.3 4 16.6 3 16.4 2 17.6 6 17.1 1 16.5
;

proc anova data=Beets;
   class Column Rep Variety Harvest;
   model Y=Rep Column Variety Rep*Column*Variety
           Harvest Harvest*Rep
           Harvest*Variety;
   test h=Rep Column Variety e=Rep*Column*Variety;
   test h=Harvest            e=Harvest*Rep;
run;
```

Output 29.4.1 Class Level Information

Sugar Beet Varieties

Latin Square Split-Plot Design

The ANOVA Procedure

Class Level Information		
Class	**Levels**	**Values**
Column	6	1 2 3 4 5 6
Rep	6	1 2 3 4 5 6
Variety	6	1 2 3 4 5 6
Harvest	2	1 2

Number of Observations Read	72
Number of Observations Used	72

Output 29.4.2 ANOVA Table

Sugar Beet Varieties

Latin Square Split-Plot Design

The ANOVA Procedure

Dependent Variable: Y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	46	98.9147222	2.1503200	7.22	<.0001
Error	25	7.4484722	0.2979389		
Corrected Total	71	106.3631944			

R-Square	Coeff Var	Root MSE	Y Mean
0.929971	3.085524	0.545838	17.69028

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Rep	5	4.32069444	0.86413889	2.90	0.0337
Column	5	1.57402778	0.31480556	1.06	0.4075
Variety	5	20.61902778	4.12380556	13.84	<.0001
Column*Rep*Variety	20	3.25444444	0.16272222	0.55	0.9144
Harvest	1	60.68347222	60.68347222	203.68	<.0001
Rep*Harvest	5	7.71736111	1.54347222	5.18	0.0021
Variety*Harvest	5	0.74569444	0.14913889	0.50	0.7729

First, note from Output 29.4.2 that the overall model is significant.

Output 29.4.3 Tests of Effects

Tests of Hypotheses Using the Anova MS for Column*Rep*Variety as an Error Term					
Source	DF	Anova SS	Mean Square	F Value	Pr > F
Rep	5	4.32069444	0.86413889	5.31	0.0029
Column	5	1.57402778	0.31480556	1.93	0.1333
Variety	5	20.61902778	4.12380556	25.34	<.0001

Tests of Hypotheses Using the Anova MS for Rep*Harvest as an Error Term					
Source	DF	Anova SS	Mean Square	F Value	Pr > F
Harvest	1	60.68347222	60.68347222	39.32	0.0015

Output 29.4.3 shows that the effects for Rep and Harvest are significant, while the Column effect is not. The average Ys for the six different Varietys are significantly different. For these four tests, look at the output produced by the two TEST statements, not at the usual ANOVA procedure output. The Variety*Harvest interaction is not significant. All other effects in the default output should either be tested by using the results from the TEST statements or are irrelevant as they are only error terms for portions of the model.

Example 29.5: Strip-Split Plot

(View the complete code for this example (anovaex5.sas) in the example repository.)

In this example, four different fertilizer treatments are laid out in vertical strips, which are then split into subplots with different levels of calcium. Soil type is stripped across the split-plot experiment, and the entire experiment is then replicated three times. The dependent variable is the yield of winter barley. The data come from the notes of G. Cox and A. Rotti.

The input data are the 96 values of Y, arranged so that the calcium value (Calcium) changes most rapidly, then the fertilizer value (Fertilizer), then the Soil value, and, finally, the Rep value. Values are shown for Calcium (0 and 1); Fertilizer (0, 1, 2, 3); Soil (1, 2, 3); and Rep (1, 2, 3, 4). The following example produces Output 29.5.1, Output 29.5.2, Output 29.5.3, and Output 29.5.4.

```
title1 'Strip-split Plot';
data Barley;
   do Rep=1 to 4;
      do Soil=1 to 3; /* 1=d 2=h 3=p */
         do Fertilizer=0 to 3;
            do Calcium=0,1;
               input Yield @;
               output;
            end;
         end;
      end;
   end;
   datalines;
4.91 4.63 4.76 5.04 5.38 6.21 5.60 5.08
4.94 3.98 4.64 5.26 5.28 5.01 5.45 5.62
5.20 4.45 5.05 5.03 5.01 4.63 5.80 5.90
6.00 5.39 4.95 5.39 6.18 5.94 6.58 6.25
```

```
5.86 5.41 5.54 5.41 5.28 6.67 6.65 5.94
5.45 5.12 4.73 4.62 5.06 5.75 6.39 5.62
4.96 5.63 5.47 5.31 6.18 6.31 5.95 6.14
5.71 5.37 6.21 5.83 6.28 6.55 6.39 5.57
4.60 4.90 4.88 4.73 5.89 6.20 5.68 5.72
5.79 5.33 5.13 5.18 5.86 5.98 5.55 4.32
5.61 5.15 4.82 5.06 5.67 5.54 5.19 4.46
5.13 4.90 4.88 5.18 5.45 5.80 5.12 4.42
;

proc anova data=Barley;
   class Rep Soil Calcium Fertilizer;
   model Yield =
           Rep
           Fertilizer Fertilizer*Rep
           Calcium Calcium*Fertilizer Calcium*Rep(Fertilizer)
           Soil Soil*Rep
           Soil*Fertilizer Soil*Rep*Fertilizer
           Soil*Calcium Soil*Fertilizer*Calcium
           Soil*Calcium*Rep(Fertilizer);
   test h=Fertilizer                e=Fertilizer*Rep;
   test h=Calcium calcium*fertilizer e=Calcium*Rep(Fertilizer);
   test h=Soil                      e=Soil*Rep;
   test h=Soil*Fertilizer           e=Soil*Rep*Fertilizer;
   test h=Soil*Calcium
          Soil*Fertilizer*Calcium    e=Soil*Calcium*Rep(Fertilizer);
   means Fertilizer Calcium Soil Calcium*Fertilizer;
run;
```

Output 29.5.1 Class Level Information

Strip-split Plot

The ANOVA Procedure

Class Level Information		
Class	**Levels**	**Values**
Rep	4	1 2 3 4
Soil	3	1 2 3
Calcium	2	0 1
Fertilizer	4	0 1 2 3

Number of Observations Read	96
Number of Observations Used	96

Output 29.5.2 ANOVA Table

Strip-split Plot

The ANOVA Procedure

Dependent Variable: Yield

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	95	31.89149583	0.33569996	.	.
Error	0	0.00000000	.		
Corrected Total	95	31.89149583			

R-Square	Coeff Var	Root MSE	Yield Mean
1.000000	.	.	5.427292

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Rep	3	6.27974583	2.09324861	.	.
Fertilizer	3	7.22127083	2.40709028	.	.
Rep*Fertilizer	9	6.08211250	0.67579028	.	.
Calcium	1	0.27735000	0.27735000	.	.
Calcium*Fertilizer	3	1.96395833	0.65465278	.	.
Rep*Calcium(Fertili)	12	1.76705833	0.14725486	.	.
Soil	2	1.92658958	0.96329479	.	.
Rep*Soil	6	1.66761042	0.27793507	.	.
Soil*Fertilizer	6	0.68828542	0.11471424	.	.
Rep*Soil*Fertilizer	18	1.58698125	0.08816563	.	.
Soil*Calcium	2	0.04493125	0.02246562	.	.
Soil*Calcium*Fertili	6	0.18936042	0.03156007	.	.
Rep*Soil*Calc(Ferti)	24	2.19624167	0.09151007	.	.

Notice in Output 29.5.2 that the default tests against the residual error rate are all unavailable. This is because the Soil*Calcium*Rep(Fertilizer) term in the model takes up all the degrees of freedom, leaving none for estimating the residual error rate. This is appropriate in this case since the TEST statements give the specific error terms appropriate for testing each effect. Output 29.5.3 displays the output produced by the various TEST statements. The only significant effect is the Calcium*Fertilizer interaction.

Output 29.5.3 Tests of Effects

Tests of Hypotheses Using the Anova MS for Rep*Fertilizer as an Error Term

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Fertilizer	3	7.22127083	2.40709028	3.56	0.0604

Tests of Hypotheses Using the Anova MS for Rep*Calcium(Fertili) as an Error Term

Source	DF	Anova SS	Mean Square	F Value	Pr > F
Calcium	1	0.27735000	0.27735000	1.88	0.1950
Calcium*Fertilizer	3	1.96395833	0.65465278	4.45	0.0255

Output 29.5.3 *continued*

Tests of Hypotheses Using the Anova MS for Rep*Soil as an Error Term					
Source	**DF**	**Anova SS**	**Mean Square**	**F Value**	**Pr > F**
Soil	2	1.92658958	0.96329479	3.47	0.0999

Tests of Hypotheses Using the Anova MS for Rep*Soil*Fertilizer as an Error Term					
Source	**DF**	**Anova SS**	**Mean Square**	**F Value**	**Pr > F**
Soil*Fertilizer	6	0.68828542	0.11471424	1.30	0.3063

Tests of Hypotheses Using the Anova MS for Rep*Soil*Calc(Ferti) as an Error Term					
Source	**DF**	**Anova SS**	**Mean Square**	**F Value**	**Pr > F**
Soil*Calcium	2	0.04493125	0.02246562	0.25	0.7843
Soil*Calcium*Fertili	6	0.18936042	0.03156007	0.34	0.9059

Output 29.5.4 Results of MEANS statement

		Yield	
Level of Fertilizer	**N**	**Mean**	**Std Dev**
0	24	5.18416667	0.48266395
1	24	5.12916667	0.38337082
2	24	5.75458333	0.53293265
3	24	5.64125000	0.63926801

		Yield	
Level of Calcium	**N**	**Mean**	**Std Dev**
0	48	5.48104167	0.54186141
1	48	5.37354167	0.61565219

		Yield	
Level of Soil	**N**	**Mean**	**Std Dev**
1	32	5.54312500	0.55806369
2	32	5.51093750	0.62176315
3	32	5.22781250	0.51825224

			Yield	
Level of Calcium	**Level of Fertilizer**	**N**	**Mean**	**Std Dev**
0	**0**	12	5.34666667	0.45029956
0	**1**	12	5.08833333	0.44986530
0	**2**	12	5.62666667	0.44707806
0	**3**	12	5.86250000	0.52886027
1	**0**	12	5.02166667	0.47615569
1	**1**	12	5.17000000	0.31826233
1	**2**	12	5.88250000	0.59856077
1	**3**	12	5.42000000	0.68409197

Output 29.5.4 shows the results of the MEANS statement, displaying for various effects and combinations of effects, as requested. You can examine the Calcium*Fertilizer means to understand the interaction better.

In this example, you could reduce memory requirements by omitting the Soil*Calcium*Rep(Fertilizer) effect from the model in the MODEL statement. This effect then becomes the ERROR effect, and you can omit the last TEST statement in the statements shown earlier. The test for the Soil*Calcium effect is then given in the Analysis of Variance table in the top portion of output. However, for all other tests, you should look at the results from the TEST statement. In large models, this method might lead to significant reductions in memory requirements.

References

Bartlett, M. S. (1937). "Properties of Sufficiency and Statistical Tests." *Proceedings of the Royal Society of London, Series A* 160:268–282.

Brown, M. B., and Forsythe, A. B. (1974). "Robust Tests for Equality of Variances." *Journal of the American Statistical Association* 69:364–367.

Chi, Y.-Y., Gribbin, M. J., Lamers, Y., Gregory, J. F., III, and Muller, K. E. (2012). "Global Hypothesis Testing for High-Dimensional Repeated Measures Outcomes." *Statistics in Medicine* 31:724–742.

Erdman, L. W. (1946). "Studies to Determine If Antibiosis Occurs among Rhizobia." *Journal of the American Society of Agronomy* 38:251–258.

Fisher, R. A. (1942). *The Design of Experiments*. 3rd ed. Edinburgh: Oliver & Boyd.

Freund, R. J., Littell, R. C., and Spector, P. C. (1986). *SAS System for Linear Models*. 1986 ed. Cary, NC: SAS Institute Inc.

Graybill, F. A. (1961). *An Introduction to Linear Statistical Models.* Vol. 1. New York: McGraw-Hill.

Henderson, C. R. (1953). "Estimation of Variance and Covariance Components." *Biometrics* 9:226–252.

Huynh, H., and Feldt, L. S. (1976). "Estimation of the Box Correction for Degrees of Freedom from Sample Data in the Randomized Block and Split Plot Designs." *Journal of Educational Statistics* 1:69–82.

Lecoutre, B. (1991). "A Correction for the Epsilon Approximate Test with Repeated Measures Design with Two or More Independent Groups." *Journal of Educational Statistics* 16:371–372.

Levene, H. (1960). "Robust Tests for the Equality of Variance." In *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, edited by I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow, and H. B. Mann, 278–292. Palo Alto, CA: Stanford University Press.

Neter, J., Wasserman, W., and Kutner, M. H. (1990). *Applied Linear Statistical Models*. 3rd ed. Homewood, IL: Irwin.

O'Brien, R. G. (1979). "A General ANOVA Method for Robust Tests of Additive Models for Variances." *Journal of the American Statistical Association* 74:877–880.

O'Brien, R. G. (1981). "A Simple Test for Variance Effects in Experimental Designs." *Psychological Bulletin* 89:570–574.

Remington, R. D., and Schork, M. A. (1970). *Statistics with Applications to the Biological and Health Sciences*. Englewood Cliffs, NJ: Prentice-Hall.

Scheffé, H. (1959). *The Analysis of Variance*. New York: John Wiley & Sons.

Schlotzhauer, S. D., and Littell, R. C. (1987). *SAS System for Elementary Statistical Analysis*. Cary, NC: SAS Institute Inc.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Smith, W. G. (1951). Dissertation notes on Canadian Sugar Factories Ltd., Taber, Alberta.

Snedecor, G. W., and Cochran, W. G. (1967). *Statistical Methods*. 6th ed. Ames: Iowa State University Press.

Steel, R. G. D., and Torrie, J. H. (1980). *Principles and Procedures of Statistics*. 2nd ed. New York: McGraw-Hill.

Welch, B. L. (1951). "On the Comparison of Several Mean Values: An Alternative Approach." *Biometrika* 38:330–336.

Chapter 30
The BCHOICE Procedure

Contents

Overview: BCHOICE Procedure . . . **1176**
PROC BCHOICE Compared with Other SAS Procedures . . . 1177
Getting Started: BCHOICE Procedure . . . **1178**
A Simple Logit Model Example . . . 1178
A Logit Model Example with Random Effects . . . 1185
Syntax: BCHOICE Procedure . . . **1189**
PROC BCHOICE Statement . . . 1190
BY Statement . . . 1201
CLASS Statement . . . 1201
MODEL Statement . . . 1204
PREDDIST Statement . . . 1209
RANDOM Statement . . . 1210
RESTRICT Statement . . . 1213
Details: BCHOICE Procedure . . . **1215**
Discrete Choice Models . . . 1215
Types of Choice Models . . . 1216
Random Effects . . . 1221
Other Types of Choice Response . . . 1224
Identification and Specification . . . 1225
Gamerman Algorithm . . . 1229
Tuning the Random Walk Metropolis in Logit Models . . . 1231
Autocall Macros for Postprocessing . . . 1233
Regenerating Diagnostics Plots . . . 1235
Displayed Output . . . 1235
ODS Table Names . . . 1239
ODS Graphics . . . 1240
Examples: BCHOICE Procedure . . . **1240**
Example 30.1: Alternative-Specific and Individual-Specific Effects . . . 1240
Example 30.2: Nested Logit Modeling . . . 1244
Example 30.3: Probit Modeling . . . 1246
Example 30.4: A Random-Effects-Only Logit Model . . . 1248
Example 30.5: Heterogeneity Affected by Individual Characteristics . . . 1253
Example 30.6: Quantities of Interest and Willingness to Pay . . . 1258
Example 30.7: Predict the Choice Probabilities . . . 1261
Example 30.8: MaxDiff Choice Example . . . 1263
Example 30.9: Allocation Choice Example . . . 1268
References . . . **1273**

Overview: BCHOICE Procedure

The BCHOICE (Bayesian choice) procedure performs Bayesian analysis for discrete choice models. Discrete choice models are used in marketing research to model decision makers' choices among alternative products and services. The decision maker might be people, households, companies and so on, and the alternatives might be products, services, actions, or any other options or items about which choices must be made (Train 2009). The collection of alternatives that are available to the decision makers is called a choice set.

Discrete choice models are derived under the assumption of utility-maximizing behavior by decision makers. When individuals are asked to make one choice among a set of alternatives, they usually determine the level of utility that each alternative offers. The utility that individual i obtains from alternative j among J alternatives is denoted as

$$u_{ij} = v_{ij} + \epsilon_{ij}, \quad i = 1, \ldots, N \text{ and } j = 1, \ldots, J$$

where the subscript i is an index for the individuals, the subscript j is an index for the alternatives in a choice set, v_{ij} is a nonstochastic utility function that relates observed factors to the utility, and ϵ_{ij} is the error component that captures the unobserved characteristics of the utility. In discrete choice models, the observed part of the utility function is assumed to be linear in the parameters,

$$v_{ij} = \mathbf{x}'_{ij}\boldsymbol{\beta}$$

where $\mathbf{x}_{ij}$ is a p-dimensional design vector of observed attribute levels that relate to alternative j and $\boldsymbol{\beta}$ is the corresponding vector of fixed regression coefficients that indicate the utilities or part-worths of the attribute levels.

Decision makers choose the alternative that gives them the greatest utility. Let $\mathbf{y}_i$ be the multinomial response vector for the ith individual. The value y_{ij} takes 1 if the jth component of $\mathbf{u}_i = (u_{i1}, \ldots, u_{iJ})$ is the largest, and 0 otherwise:

$$\begin{aligned} u_{ij} &= \mathbf{x}'_{ij}\boldsymbol{\beta} + \epsilon_{ij} \\ y_{ij} &= \begin{cases} 1 & \text{if } u_{ij} \geq \max(\mathbf{u}_i) \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

Different specifications about the density of the error vector $\boldsymbol{\epsilon}_i = (\epsilon_{i1}, \ldots, \epsilon_{iJ})$ result in different types of choice models: logit, nested logit, and probit, as detailed in the section "Types of Choice Models" on page 1216. Logit and nested logit models have closed-form likelihood, whereas a probit model does not.

The past 15 years have seen a dramatic increase in using a Bayesian approach to develop new methods of analysis and models of consumer behavior. The milestone breakthroughs are Albert and Chib (1993) and McCulloch and Rossi (1994) for choice probit models, and Allenby and Lenk (1994) and Allenby (1997) for logit models that have normally distributed random effects (these models are called mixed logit models). Train (2009) extends the Bayesian procedure for mixed logit models to include nonnormal distributions, such as lognormal, uniform, and triangular distributions.

An issue in choice models is that the quantity of relevant data at the individual level is very limited. Respondents frequently become fatigued after answering 15 to 20 questions in a survey. Rossi, McCulloch, and Allenby (1996) and Allenby and Rossi (1999) show how to obtain information about individual-level

parameters within a model by using random taste variation. The lack of data at the individual level, combined with the desire to account for individual differences instead of treating all respondents alike, presents challenges in marketing research. Bayesian methods are ideally suited for analyzing limited data.

Bayesian methods have several advantages. First, Bayesian methods do not require optimization of any function. For probit models and logit models that have random effects, optimization of the likelihood function can be numerically difficult. Different starting values might lead to different maximization results. The problem of a local maximum versus a global maximum is another issue, because convergence is not guaranteed to find the global maximum. Second, Bayesian procedures enable consistency and efficiency to be achieved under more relaxed conditions. When the likelihood function does not have a closed form or there are too many parameters (as in models with random effects), simulation can be used to estimate the likelihood function. Maximization that is based on such a simulated likelihood function is consistent only if the number of draws in simulation rises with the sample size, and it is efficient only if the number of draws in simulation increases faster than the square root of the sample size. On the other hand, Bayesian methods are consistent for a fixed number of draws and are efficient if the number of draws goes up at any rate with the sample size.

For a short introduction to Bayesian analysis and related basic concepts, see Chapter 8, "Introduction to Bayesian Analysis Procedures." Also see the section "A Bayesian Reading List" on page 178 for a guide to Bayesian textbooks of varying degrees of difficulty. It follows from Bayes' theorem that a posterior distribution is proportional to the product of the likelihood function and the prior distribution of the parameter. When it is difficult to obtain the posterior distribution analytically, Bayesian methods often rely on simulations to generate samples from the posterior distribution, and they use the simulated draws to approximate the distribution and to make the inferences.

To use the BCHOICE procedure, you need to specify the type of model for the data. You can also supply a prior distribution for the parameters if you want something other than the default noninformative prior. PROC BCHOICE obtains samples from the corresponding posterior distributions, produces summary and diagnostic statistics, and saves the posterior samples in an output data set that can be used for further analysis. The procedure derives inferences from simulation rather than through analytic or numerical methods. You should expect slightly different answers from each run for the same problem, unless you use the same random number seed.

PROC BCHOICE Compared with Other SAS Procedures

The underlying structure of PROC BCHOICE is similar to the structure of PROC MCMC in that both procedures obtain samples from the posterior distributions and produce summary and diagnostic statistics when you specify the model or the priors or both. However, they differ in that PROC MCMC is a general-purpose Markov chain Monte Carlo (MCMC) simulation procedure that is designed to fit a wide range of Bayesian models, whereas PROC BCHOICE is designed specifically for discrete choice models. You can call PROC MCMC to analyze data that have any likelihood, prior, or hyperprior, as long as these functions can be programmed by using the SAS DATA step functions. For example, you can fit choice logit and nested logit models in PROC MCMC by using some SAS coding to specify the likelihood. PROC BCHOICE works only with choice models, but it is customized to fit special characteristics and features in a choice model. The syntax is quite different from PROC MCMC's syntax. PROC BCHOICE provides a CLASS statement to handle categorical variables, and it requires less complicated SAS coding for choice models. The default sampling method for choice logit models when direct sampling is not available is the Metropolis-Hastings method, which is based on the Gamerman approach, which in turn has proved to be often more efficient than

the random walk Metropolis algorithm that PROC MCMC uses. In addition, it is difficult to fit choice probit models in PROC MCMC.

For a standard logit choice model, you can use the TIES=BRESLOW option in the PHREG procedure. The approach has the same likelihood as PROC BCHOICE of fitting the data. You use the STRATA statement in PROC PHREG to specify how to define the choice set. However, this is a frequentist approach, and it does not work for a choice model with random effects.

Getting Started: BCHOICE Procedure

The first example in this section is a simple logit conjoint analysis that illustrates some basic features of PROC BCHOICE. The second example discusses a mixed logit model with individual-level random effects, which has become increasingly popular.

A Simple Logit Model Example

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/bchcgs1.sas.)

This example uses a standard logit model in which the error component, ϵ_{ij} $(j = 1, \ldots, J)$, is independently and identically distributed (iid) with the Type I extreme-value distribution, $\exp(-\exp(-\epsilon_{ij}))$. This assumption provides a convenient logit form for the choice probability (McFadden 1974):

$$P(y_{ij} = 1) = \frac{\exp(\mathbf{x}'_{ij}\boldsymbol{\beta})}{\sum_{k=1}^{J} \exp(\mathbf{x}'_{ik}\boldsymbol{\beta})}, \quad i = 1, \ldots, N \text{ and } j = 1, \ldots, J$$

The likelihood is formed by the product of the N independent multinomial distributions:

$$p(\mathbf{Y}|\boldsymbol{\beta}) = \prod_{i=1}^{N} \prod_{j=1}^{J} P(y_{ij} = 1)^{y_{ij}}$$

Suppose $\boldsymbol{\beta}$ has a normal prior

$$\pi(\boldsymbol{\beta}) = \mathrm{N}(\mathbf{0}, c\mathbf{I})$$

where $\mathbf{I}$ is the identity matrix and c is a scalar. c is often set to be large for a noninformative prior.

The posterior density of the parameter $\boldsymbol{\beta}$ is

$$p(\boldsymbol{\beta}|\mathbf{Y}) \propto p(\mathbf{Y}|\boldsymbol{\beta})\pi(\boldsymbol{\beta})$$

PROC BCHOICE obtains samples from the posterior distribution, produces summary and diagnostic statistics, and saves the posterior samples in an output data set that can be used for further analysis.

In this example (Kuhfeld 2010), each of 10 subjects is presented with eight different chocolate candies and asked to choose one. The eight candies consist of the 2^3 combinations of dark or milk chocolate, soft or chewy center, and nuts or no nuts. Each subject sees all eight alternatives and makes one choice. Experimental choice data such as these are usually analyzed by using a multinomial logit model.

The following statements read the data:

```
title 'Conjoint Analysis of Chocolate Candies';

data Chocs;
   input Subj Choice Dark Soft Nuts;
   datalines;
1 0 0 0 0
1 0 0 0 1
1 0 0 1 0
1 0 0 1 1
1 1 1 0 0
1 0 1 0 1

   ... more lines ...

10 0 1 0 0
10 1 1 0 1
10 0 1 1 0
10 0 1 1 1
;

proc print data=Chocs (obs=16);
run;
```

The data for the first two subjects are shown in Figure 30.1.

Figure 30.1 Data for the First Two Subjects

Conjoint Analysis of Chocolate Candies

Obs	Subj	Choice	Dark	Soft	Nuts
1	1	0	0	0	0
2	1	0	0	0	1
3	1	0	0	1	0
4	1	0	0	1	1
5	1	1	1	0	0
6	1	0	1	0	1
7	1	0	1	1	0
8	1	0	1	1	1
9	2	0	0	0	0
10	2	0	0	0	1
11	2	0	0	1	0
12	2	0	0	1	1
13	2	0	1	0	0
14	2	1	1	0	1
15	2	0	1	1	0
16	2	0	1	1	1

The data set contains 10 subjects and 80 observation lines. Each line of the data represents one alternative in the choice set for each subject. It is required that the response variable, which is Choice in this study, indicate the chosen alternative by the value 1 and the unchosen alternatives by the value 0. Dark is 1 for dark chocolate and 0 for milk chocolate; Soft is 1 for soft center and 0 for chewy center; Nuts is 1 if the candy

contains nuts and 0 if it does not contain nuts. In this example, subject 1 chose the fifth alternative (Choice=1, which is Dark/Chewy/No Nuts) among the eight alternatives in the choice set. All the chosen and unchosen alternatives must appear in the data set. If you have choice data in which all alternatives appear on one line, then you must rearrange the data in the correct form (that is, the data must contain one observation line for each alternative of each choice set for each subject).

The following statements fit a multinomial logit model:

```
ods graphics on;
proc bchoice data=Chocs outpost=Bsamp nmc=10000 thin=2 diag=(AutoCorr
     ESS MCSE) seed=124;
   class Dark(ref='0') Soft(ref='0') Nuts(ref='0') Subj;
   model Choice = Dark Soft Nuts / choiceset=(Subj) cprior=normal(var=1000);
run;
```

The ODS GRAPHICS ON statement invokes the ODS Graphics environment and displays the diagnostic plots, such as the trace and autocorrelation function plots of the posterior samples. For more information about ODS, see Chapter 24, "Statistical Graphics Using ODS."

The PROC BCHOICE statement invokes the procedure, and the DATA= option specifies the input data set Chocs. The OUTPOST= option requests an output data set called Bsamp to contain all the posterior samples. The NMC= option specifies the number of posterior simulation iterations in the main simulation loop after burn-in (the default number of burn-in iterations is 500). The THIN= option controls the thinning of the Markov chain and specifies to keep one of every two samples. Thinning is often used to reduce correlation among posterior sample draws. In this example, 5,000 simulated values are saved in the Bsamp data set. The DIAG=(AUTOCORR ESS MCSE) option requests that three convergence diagnostics be output to help determine whether the chain has converged: the autocorrelations, effective sample sizes, and Monte Carlo standard errors. The SEED= option specifies a seed for the random number generator, which guarantees the reproducibility of the random stream.

The CLASS statement names the classification variables to be used in the model. The CLASS statement must precede the MODEL statement (as it must in most other SAS procedures). The REF= option specifies the reference level.

The MODEL statement is required; it defines the dependent variable (y_{ij}) and independent variables (x_{ij}). To the left of the equal sign in the MODEL statement, you specify the dependent variable that indicates which alternatives are chosen and which ones are not chosen. The dependent variable Choice has the value 1 (chosen) or 0 (unchosen). You specify the independent variables after the equal sign. The independent variables often indicate attributes or characteristics of the alternatives in the choice set, such as Dark, Soft, and Nuts in this example. The CHOICESET= option specifies how a choice set is defined. In this example, CHOICESET=(Subj) because there is one and only one choice set per subject. The variable that you specify in the CHOICESET= option must be a classification variable that appears in the CLASS statement. Always use the CHOICESET= option to define the choice set.

The first table that PROC BCHOICE produces is the "Model Information" table, as shown in Figure 30.2. This table displays basic information about the analysis, such as the name of the input data set, response variable, model type, choice response type, sampling algorithm, burn-in size, simulation size, thinning number, and random number seed. The random number seed initializes the random number generators. If you repeat the analysis and use the same seed, you get an identical stream of random numbers. Here the response type is binary, where the response takes either 1 (chosen) or 0 (unchosen). Other types of choice response include MaxDiff and allocation, where the response can take other values.

Figure 30.2 Model Information

The BCHOICE Procedure

Model Information	
Data Set	WORK.CHOCS
Response Variable	Choice
Type of Model	Logit
Type of Choice Response	Binary
Fixed Effects Included	Yes
Random Effects Included	No
Sampling Algorithm	Gamerman Metropolis
Burn-In Size	500
Simulation Size	10000
Thinning	2
Random Number Seed	124
Number of Threads	1

The "Choice Sets Summary" table is displayed by default; you can use it to check the data entry. In this case, there are 10 choice sets, one for each subject. All 10 choice sets display the same pattern: they have a total of eight alternatives, one of which is chosen and seven of which are unchosen. Sometimes there can be more than one pattern among the choice sets (for example, some choice sets have nine alternatives). Logit models can have different patterns as long as each choice set has at least two alternatives in total and only one chosen alternative. If more than one alternative is chosen or no alternative is chosen, the choice set is invalid for the binary response. PROC BCHOICE deletes all the invalid choice sets and produces a warning message.

Figure 30.3 Choice Sets Summary

Choice Sets Summary				
Pattern	Choice Sets	Total Alternatives	Chosen Alternatives	Not Chosen
1	10	8	1	7

The next table is the "Number of Observations" table, as shown in Figure 30.4. This table lists the number of observations that are read from the DATA= data set and the number of nonmissing and valid observations that are used in the analysis. PROC BCHOICE does not impute missing values. If any missing value is encountered in a row of the DATA= data set, PROC BCHOICE skips that row and moves on to read the next row. If the missing value causes the corresponding choice set to be invalid, the entire choice set is discarded from analysis. In this example, all 80 observations are used.

Figure 30.4 Number of Observations

Number of Observations	
Number of Observations Read	80
Number of Observations Used	80

PROC BCHOICE reports posterior summary statistics (posterior means, standard deviations, and highest posterior density (HPD) intervals) for each parameter in the "Posterior Summaries and Intervals" table, as shown in Figure 30.5. For more information about posterior statistics, see the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Figure 30.5 PROC BCHOICE Posterior Summary Statistics

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Dark 1	5000	1.5308	0.7943	0.1848	3.2412
Soft 1	5000	-2.4312	0.9792	-4.4882	-0.8125
Nuts 1	5000	0.9671	0.7454	-0.3255	2.6675

Before you examine the posterior summary statistics and try to draw any conclusions, you might want to verify that the simulation has converged. Convergence diagnostics are essential to inferring from simulations that are based on Markov chains. If the Markov chain has not converged, all conclusions that are based on the samples might be misleading. PROC BCHOICE computes the effective sample size by default. There are a number of other convergence diagnostics to help you determine whether the chain has converged: the Monte Carlo standard errors, the autocorrelations at selected lags, and so on. These statistics are shown in Figure 30.6. For details and interpretations of these diagnostics, see the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

The "Posterior Autocorrelations" table shows that the autocorrelations among posterior samples reduce quickly. The "Effective Sample Sizes" table reports the number of effective sample sizes of the Markov chain. The "Monte Carlo Standard Errors" table indicates that the standard errors of the mean estimates for each of the variables are relatively small with respect to the posterior standard deviations. The values in the MCSE/SD column (ratios of the standard errors and the standard deviations) are small. This means that only a fraction of the posterior variability is caused by the simulation.

Figure 30.6 PROC BCHOICE Convergence Diagnostics

Posterior Autocorrelations				
Parameter	Lag 1	Lag 5	Lag 10	Lag 50
Dark 1	0.4495	0.1427	0.0603	-0.0095
Soft 1	0.5389	0.1518	0.0730	0.0449
Nuts 1	0.4136	0.1473	0.0259	-0.0264

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
Dark 1	1078.4	4.6365	0.2157
Soft 1	904.9	5.5253	0.1810
Nuts 1	1161.6	4.3043	0.2323

Monte Carlo Standard Errors			
Parameter	MCSE	Standard Deviation	MCSE/SD
Dark 1	0.0242	0.7943	0.0305
Soft 1	0.0326	0.9792	0.0332
Nuts 1	0.0219	0.7454	0.0293

PROC BCHOICE produces a number of graphs, shown in Figure 30.7, which also aid convergence diagnostic checks. The trace plots have two important aspects to examine. First, you want to check whether the mean of the Markov chain has stabilized and appears constant over the graph. Second, you want to check whether the

chain has good mixing and is "dense," in the sense that it quickly traverses the support of the distribution to explore both the tails and the mode areas efficiently. The plots show that the chains appear to have reached their stationary distributions.

Next, you want to examine the autocorrelation plots, which indicate the degree of autocorrelation for each of the posterior samples. High correlations usually imply slow mixing. Finally, the kernel density plots estimate the posterior marginal distributions for each parameter.

Figure 30.7 PROC BCHOICE Diagnostic Plots

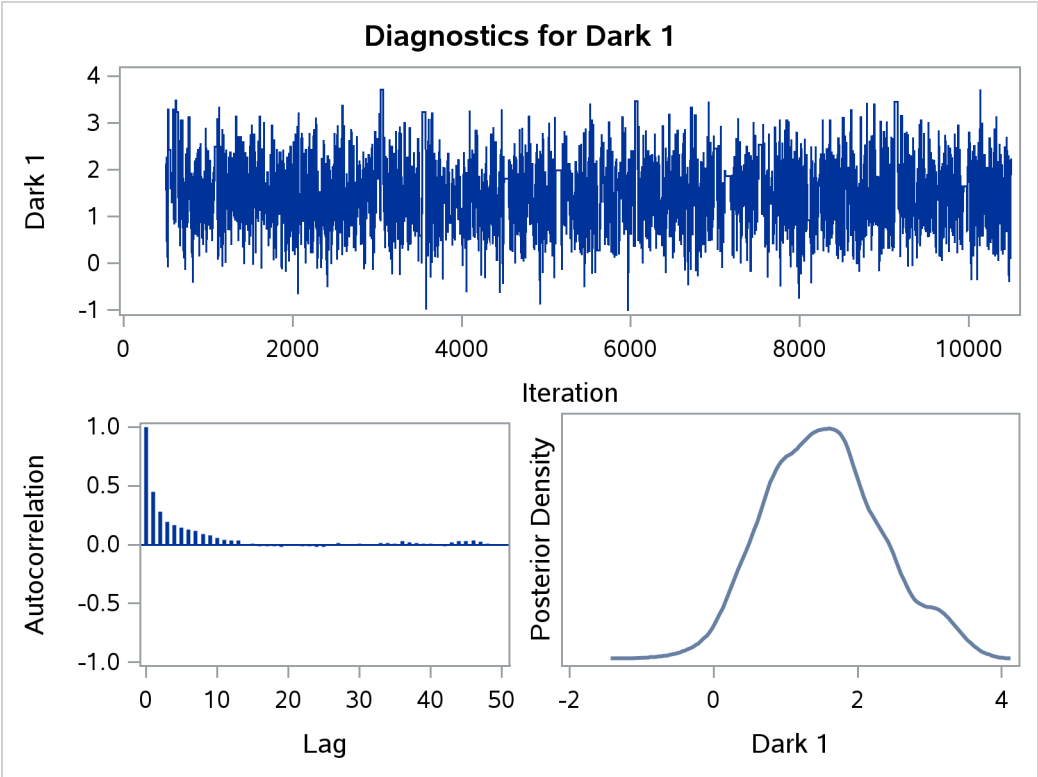

Figure 30.7 *continued*

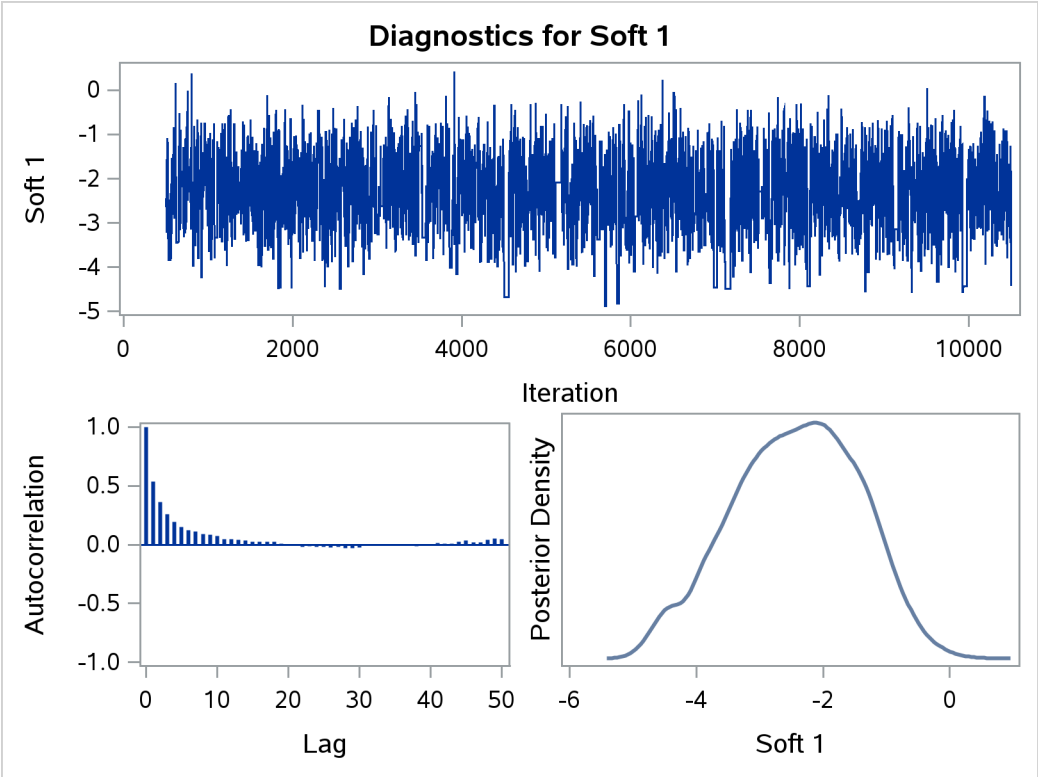

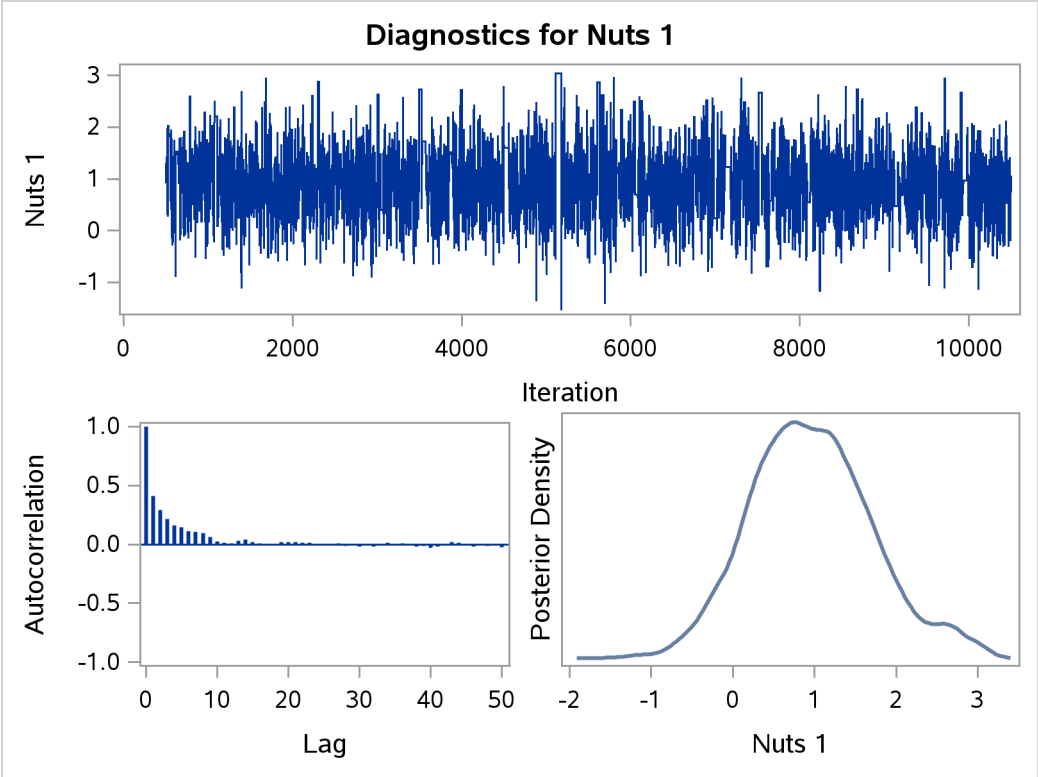

Because the chains seem to have reached their stationary distributions, you can go back to the posterior

summary table for results and conclusions. As seen in Figure 30.5, the part-worth for dark chocolate is 1.5 and the part-worth for milk chocolate (base category) is structural 0; the part-worth for soft center is –2.4 and the part-worth for chewy center is structural 0; the part-worth for containing nuts is 1.0 and the part-worth for no nuts is structural 0. A positive part-worth implies being more favorable. Hence, dark chocolate is preferred over milk chocolate, soft centers are less popular than chewy centers, and candies with nuts are more popular than candies without nuts.

A Logit Model Example with Random Effects

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/bchcgs2.sas.)

Choice models that have random effects (or random coefficients) provide solutions to create individual-level or group-specific utilities. Because people have different preferences, it can be misleading to roll the whole sample together into a single set of utilities. The desire to account for individual differences, instead of treating all respondents alike, provides challenges in marketing research. For logit models that have random effects, using frequentist methods to optimize of the likelihood function can be numerically difficult. Bayesian methods are ideally suited for analysis with random effects.

Choice models that have random effects generalize the standard choice models to incorporate individual-level effects. Let the utility that individual i obtains from alternative j in choice situation t ($t = 1, \ldots, T$) be

$$\begin{aligned} u_{ijt} &= \mathbf{x}'_{ijt}\boldsymbol{\beta} + \mathbf{z}'_{ijt}\boldsymbol{\gamma}_i + \epsilon_{ijt} \\ y_{ijt} &= 1 \text{ if } u_{ijt} \geq \max(u_{i1t}, u_{i2t}, \ldots, u_{iJt}) \\ &= 0 \text{ otherwise} \end{aligned}$$

where y_{ijt} is the observed choice for individual i and alternative j in choice situation t; $\mathbf{x}_{ijt}$ is the fixed design vector for individual i and alternative j in choice situation t; $\boldsymbol{\beta}$ are the fixed coefficients; $\mathbf{z}_{ijt}$ is the random design vector for individual i and alternative j in choice situation t; and $\boldsymbol{\gamma}_i$ are the random coefficients for individual i corresponding to $\mathbf{z}_{ijt}$.

It is assumed that each $\boldsymbol{\gamma}_i$ is drawn from a superpopulation and that this superpopulation is normal, $\boldsymbol{\gamma}_i \sim$ iid $\mathrm{N}(\mathbf{0}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}})$. An additional stage is added to the model in which a prior for $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$ is specified:

$$\begin{aligned} \pi(\boldsymbol{\gamma}_i) &= \mathrm{N}(\mathbf{0}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}) \\ \pi(\boldsymbol{\Omega}_{\boldsymbol{\gamma}}) &= \text{inverse Wishart}(\nu_0, \mathbf{V}_0) \end{aligned}$$

The covariance matrix $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$ characterizes the extent of heterogeneity among individuals. Large diagonal elements of $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$ indicate substantial heterogeneity in part-worths. Off-diagonal elements indicate patterns in the evaluation of attribute levels in pairs.

Consider a study that estimates the market demand for kitchen trash cans (Rossi 2013). There are four attributes, and each has two levels: touchless opening (Yes/No), material (Steel/Plastic), automatic trash bag replacement (Yes/No), and price (80/40). The number of all possible hypothetical types of trash cans is $2^4 = 16$. Including more attributes and more levels can easily become unmanageable. The study uses a fractional factorial design, in which the first three factors are set up to be a full factorial design and the fourth is generated as the product of the first three. This design confounds the three-way interaction with the effect of the fourth factor, shown in Table 30.1.

Table 30.1 Design for the Trash Can Study

Obs	Touchless	Steel	AutoBag	Price80
1	–1	–1	–1	–1
2	–1	–1	1	1
3	–1	1	–1	1
4	–1	1	1	–1
5	1	–1	–1	1
6	1	–1	1	–1
7	1	1	–1	–1
8	1	1	1	1

In Table 30.1, 1 means "Yes" and –1 means "No." This is a balanced design, in which each level appears the same number of times. This study assigns only two alternatives to a choice set by randomly sampling two rows from the previous table and giving each individual 10 choice sets (or choice tasks) to pick from. For more information about how to design a choice model efficiently, see Kuhfeld (2010).

Data were obtained by enrolling 104 people and assigning 10 choice tasks to each of them: for each task, the participants stated their preference between two types of trash cans. The following steps read in the data:

```
data Trashcan;
   input ID Task Choice Index Touchless Steel AutoBag Price80 @@;
   datalines;
1 1 1 1 0 1 1 0 1 1 0 2 1 1 0 0 1 2 0 1 0 0 0 0 1 2 1 2 1 1 1 1 1 3 0 1
0 0 0 0 1 3 1 2 1 1 0 0 1 4 0 1 0 1 0 1 1 4 1 2 1 0 0 1 1 5 1 1 0 1 1 0
1 5 0 2 1 0 0 1 1 6 0 1 0 0 1 1 1 6 1 2 1 1 1 1 1 7 0 1 1 0 0 1 1 7 1 2
1 1 0 0 1 8 0 1 0 0 1 1 1 8 1 2 0 1 1 0 1 9 1 1 0 0 1 1 1 9 0 2 1 0 0 1
1 10 0 1 0 1 1 0 1 10 1 2 1 0 1 0 2 1 1 1 0 1 1 0 2 1 0 2 1 1 0 0 2 2 0
1 0 0 0 0 2 2 1 2 1 1 1 1 2 3 0 1 0 0 0 0 2 3 1 2 1 1 0 0 2 4 0 1 0 1 0
1 2 4 1 2 1 0 0 1 2 5 1 1 0 1 1 0 2 5 0 2 1 0 0 1 2 6 1 1 0 0 1 1 2 6 0
2 1 1 1 1 2 7 1 1 1 0 0 1 2 7 0 2 1 1 0 0 2 8 1 1 0 0 1 1 2 8 0 2 0 1 1
0 2 9 1 1 0 0 1 1 2 9 0 2 1 0 0 1 2 10 1 1 0 1 1 0 2 10 0 2 1 0 1 0 3 1

   ... more lines ...

2 1 2 1 1 1 1 104 3 0 1 0 0 0 0 104 3 1 2 1 1 0 0 104 4 0 1 0 1 0 1 104
4 1 2 1 0 0 1 104 5 1 1 0 1 1 0 104 5 0 2 1 0 0 1 104 6 0 1 0 0 1 1 104
6 1 2 1 1 1 1 104 7 0 1 1 0 0 1 104 7 1 2 1 1 0 0 104 8 0 1 0 0 1 1 104
8 1 2 0 1 1 0 104 9 0 1 0 0 1 1 104 9 1 2 1 0 0 1 104 10 0 1 0 1 1 0 104
10 1 2 1 0 1 0
;

proc print data=Trashcan (obs=8);
run;
```

The data for the first four choice tasks are shown in Figure 30.8.

Figure 30.8 Data for the First Four Choice Tasks

Obs	ID	Task	Choice	Index	Touchless	Steel	AutoBag	Price80
1	1	1	1	1	0	1	1	0
2	1	1	0	2	1	1	0	0
3	1	2	0	1	0	0	0	0
4	1	2	1	2	1	1	1	1
5	1	3	0	1	0	0	0	0
6	1	3	1	2	1	1	0	0
7	1	4	0	1	0	1	0	1
8	1	4	1	2	1	0	0	1

In the data, ID is the individual's ID number, and Task indexes the number of choice tasks. The response is Choice, which states each individual's choice for each choice task. Touchless, Steel, AutoBag, and Price80 are the attribute variables; for each of them, 1 means "Yes" and 0 means "No." In the data, 0 replaces the –1 values that are shown in the design matrix in Table 30.1.

The following statements fit a logit model with random effects:

```
proc bchoice data=Trashcan seed=1 nmc=30000 thin=2 nthreads=4;
   class ID Task;
   model Choice = Touchless Steel AutoBag Price80 / choiceset=(ID Task);
   random Touchless Steel AutoBag Price80 / sub=ID monitor=(1 to 5) type=un;
run;
```

The NTHREADS option in the PROC BCHOICE statement specifies the number of threads to be used for running analytic computations and simulation simultaneously. Using four threads at the same time enhances the efficiency and reduces the run time. If you do not specify the NTHREADS option, the default number is 1. The maximum number of threads should not exceed the total number of CPUs on the host where the analytic computations execute.

The choice set is specified by ID (which identifies the participants) and by Task (which identifies each of the 10 choice tasks that are assigned to each participant). The variables ID and Task are needed in the CLASS statement because they define the choice set in the MODEL statement.

In addition to the MODEL statement for fixed effects, the RANDOM statement is added for random effects. Note that Touchless, Steel, AutoBag, and Price80 are listed as both fixed and random effects, so that their average part-worth values in the population are estimated via fixed effects and the deviation from the overall mean for each individual is presented through random effects. The SUB=ID argument in the RANDOM statement defines ID as a subject index for the random effects grouping, so that each person with a different ID has his own random effects. The MONITOR=(1 to 5) option requests to display the summary and diagnostics statistics of the random-effects parameter estimates for the first five subjects. By default, PROC BCHOICE does not print the summary and diagnostics statistics for any individual-level random-effects to save time and space. In models that have a large number of individual random effects (for example, tens of thousands individuals), it may take a long time to display the summary, diagnostics statistics, and plots for all the individual-level parameters, so be cautious when using the MONITOR option without providing a subset of the random-effects parameters.

The TYPE=UN option in the RANDOM statement specifies an unstructured covariance matrix for the random effects. The unstructured type provides a mechanism for estimating the correlation between the random effects. The TYPE=VC (variance components) option, which is a diagonal matrix and is the default structure, models a different variance component for each random effect.

Summary statistics for the fixed coefficients ($\boldsymbol{\beta}$), the covariance of the random coefficients ($\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$), and the random coefficients ($\boldsymbol{\gamma}_i$) for the first five individuals are shown in Figure 30.9.

Figure 30.9 Posterior Summary Statistics

The BCHOICE Procedure

Posterior Summaries and Intervals

Parameter	Subject	N	Mean	Standard Deviation	95% HPD Interval	
Touchless		15000	1.7043	0.2709	1.1645	2.2322
Steel		15000	1.0516	0.2608	0.5376	1.5691
AutoBag		15000	2.1722	0.3583	1.4886	2.8948
Price80		15000	-4.6321	0.6720	-5.9688	-3.4046
RECov Touchless, Touchless		15000	3.1145	1.0613	1.3441	5.2576
RECov Steel, Touchless		15000	-0.4727	0.8796	-2.1884	1.2334
RECov Steel, Steel		15000	2.6312	0.9565	1.1035	4.5817
RECov AutoBag, Touchless		15000	-0.6473	0.8811	-2.3803	1.1974
RECov AutoBag, Steel		15000	0.0595	0.7078	-1.4318	1.3748
RECov AutoBag, AutoBag		15000	3.5547	1.5837	0.9611	6.7248
RECov Price80, Touchless		15000	-1.3016	1.1857	-3.8046	0.7154
RECov Price80, Steel		15000	-1.5935	1.2052	-4.1428	0.6170
RECov Price80, AutoBag		15000	-2.2190	1.7156	-5.7073	0.7349
RECov Price80, Price80		15000	7.7452	3.4362	2.2287	14.3546
Touchless	ID 1	15000	0.5244	1.0525	-1.3366	2.7890
Steel	ID 1	15000	-0.3298	1.0799	-2.3057	1.9751
AutoBag	ID 1	15000	1.6216	1.3842	-0.9207	4.4860
Price80	ID 1	15000	-0.5394	2.2018	-5.2251	3.4324
Touchless	ID 2	15000	-1.7498	0.9341	-3.5965	0.0610
Steel	ID 2	15000	-1.4343	0.8572	-3.2176	0.2048
AutoBag	ID 2	15000	0.4767	1.3208	-2.0239	3.1743
Price80	ID 2	15000	4.6873	1.4955	1.9223	7.7535
Touchless	ID 3	15000	-0.6191	0.9695	-2.4024	1.4490
Steel	ID 3	15000	0.5779	0.9524	-1.2792	2.4187
AutoBag	ID 3	15000	-1.8737	1.2107	-4.2878	0.4407
Price80	ID 3	15000	3.7507	1.6747	0.4795	7.0672
Touchless	ID 4	15000	-0.5921	0.9679	-2.5222	1.2694
Steel	ID 4	15000	0.3152	1.0416	-1.7134	2.3286
AutoBag	ID 4	15000	0.9770	1.4250	-1.7103	3.9276
Price80	ID 4	15000	-1.9068	2.3717	-6.4256	2.8233
Touchless	ID 5	15000	-1.2689	1.1491	-3.4720	0.9828
Steel	ID 5	15000	1.6581	1.2220	-0.4731	4.2409
AutoBag	ID 5	15000	1.2695	1.5708	-1.7506	4.4762
Price80	ID 5	15000	-0.3706	2.3810	-5.0593	4.2134

The fixed effects (Touchless, Steel, AutoBag, and Price80) are shown in the first four rows. Across all the respondents in the data, the average part-worths for touchless opening, steel material, and automatic trash bag replacement are all positive, indicating that most people favor those features; the average part-worth for having to pay USD80 for a trash can instead of USD40 is negative (–4.6), which is very intuitive, because spending more money is usually unfavorable.

The covariance estimate of the random coefficients ($\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$) is displayed by the parameters whose label begins with "RECov":

$$\hat{\boldsymbol{\Omega}}_{\boldsymbol{\gamma}} = \begin{pmatrix} 3.11 & . & . & . \\ -0.47 & 2.63 & . & . \\ -0.65 & 0.06 & 3.55 & . \\ -1.30 & -1.59 & -2.22 & 7.75 \end{pmatrix}$$

where the dots refer to the corresponding elements in the lower part of the symmetric covariance matrix. The covariance estimate of the random coefficients ($\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$) characterizes the variability of part-worths across respondents. Some of the diagonal elements of the matrix are large. For example, the variance for price (labeled "RECov Price80, Price80") is quite large, indicating substantial unexplained difference in response to price. Off-diagonal elements of the matrix illustrate attribute levels that tend to be evaluated similarly (positive covariance) or differently (negative covariance) across all the respondents. The covariances between each of the attributes (Touchless, Steel, and AutoBag) and Price80 are all negative, implying that the respondents who prefer some of the new features are those who are also unwilling to pay a higher price for the trash can. Therefore, offering a discounted price might be a particularly effective method of introducing the new features to customers.

The next set of parameters that are displayed are the estimates for the individual-level random effects for the first five respondents (see Figure 30.9). These estimates are the deviation from the overall means (which are estimated via the fixed effects). The part-worth for touchless opening for the first respondent (who is labeled "ID 1" in the Subject column) is 1.7 + 0.5 = 2.2.

Allenby and Rossi (1999) and Rossi, Allenby, and McCulloch (2005) propose a hierarchical Bayesian random-effects model that is set up in a different way such that there are no fixed effects but only random effects. For more information about this type of model, see the section "Random Effects" on page 1221 and a follow-up example in "Example 30.4: A Random-Effects-Only Logit Model" on page 1248.

Syntax: BCHOICE Procedure

The following statements are available in the BCHOICE procedure. Items within < > are optional.

```
PROC BCHOICE <options> ;
    BY variables ;
    CLASS variable <(options)> <... variable <(options)> > </ options> ;
    MODEL response <(response-options)> = <fixed-effects> </ model-options> ;
    RANDOM random-effects </ options> ;
    PREDDIST OUTPRED=SAS-data-set <options> ;
```

The PROC BCHOICE and MODEL statements are required. The CLASS statement, if present, must precede the MODEL statement.

The rest of this section provides detailed syntax information for each statement, beginning with the PROC BCHOICE statement. The remaining statements are presented in alphabetical order.

PROC BCHOICE Statement

PROC BCHOICE *<options>* **;**

The PROC BCHOICE statement invokes the BCHOICE procedure.

Table 30.2 summarizes the *options* available in the PROC BCHOICE statement.

Table 30.2 PROC BCHOICE Statement Options

Option	Description
Basic Options	
DATA=	Names the input data set
GPU	Enables the use of graphics processing units (GPUs) to perform parallel computations
NBI=	Specifies the number of burn-in iterations
NMC=	Specifies the number of iterations, excluding the burn-in iterations
NTHREADS=	Specifies the number of threads to use
NTU=	Specifies the number of tuning iterations for the random walk sampler
OUTPOST=	Names the output data set to contain posterior samples of parameters
SEED=	Specifies the random seed for simulation
THIN=	Specifies the thinning rate
Sampling, Summary, Diagnostics, and Plotting Options	
ALGORITHM=	Specifies the algorithm to use to sample the posterior distribution
DIAGNOSTICS=	Controls the convergence diagnostics
DIC	Computes the deviance information criterion (DIC)
HITPROB	Outputs the average of estimated probabilities of chosen alternatives in the input data
PLOTS=	Controls plotting
STATISTICS=	Controls posterior statistics
Other Options	
ACCEPTTOL=	Specifies the acceptance rate tolerance for the random walk sampler
INF=	Specifies the machine numerical limit for infinity
LOGPOST	Calculates the logarithm of the posterior density and likelihood
MAXTUNE=	Specifies the maximum number of tuning loops for the random walk sampler
MCHISTORY=	Displays Markov chain sampling history
MINTUNE=	Specifies the minimum number of tuning loops for the random walk sampler
NOCLPRINT	Suppresses the “Class Level Information” table completely or partially
SCALE=	Specifies the initial scale applied to the proposal distribution for the random walk sampler
TARGACCEPT=	Specifies the target acceptance rate for the random walk sampler
TUNEWT=	Specifies the weight used in covariance updating for the random walk sampler

You can specify the following *options*.

ACCEPTTOL=*n*

specifies a tolerance for acceptance probabilities for the random walk sampler. By default, ACCEPTTOL=0.075. You can specify this option for logit models.

ALGORITHM=GAMERMAN | RWM | LATENT

ALG=GAMERMAN | RWM | LATENT

specifies the algorithm to use to sample the posterior distribution for the regression coefficients. You can specify the following algorithms:

GAMERMAN

uses the Metropolis-Hastings approach of Gamerman (1997). This is the default for logit models.

RWM

uses the random walk Metropolis algorithm along with the normal proposal, as suggested in Rossi, Allenby, and McCulloch (2005).

LATENT < (NTRY=*n***) >**

uses the latent variables via the data augmentation method as proposed in McCulloch and Rossi (1994). This approach avoids direct evaluation of the likelihood. This is the default for probit models. This option is ignored for logit models. You can specify the following suboption:

NTRY=*n*

specifies a positive integer that sets the maximum number of attempts for sampling from the truncated conditional distributions. By default, NTRY=20.

When possible, PROC BCHOICE samples directly from the full conditional distribution. Otherwise, the default sampling algorithm is the Gamerman algorithm for standard logit models, the random walk Metropolis algorithm for nested logit models, and the latent variables via the data augmentation method for probit models. Standard logit models can also use random walk Metropolis sampling algorithm if specified, while the other two types of models, nested logit models and probit models, have only their own default sampling algorithm.

DATA=*SAS-data-set*

specifies the input data set.

DIAGNOSTICS=NONE | (*keyword-list***)**

DIAG=NONE | (*keyword-list***)**

specifies options for convergence diagnostics. By default, PROC BCHOICE computes the effective sample sizes. The sample autocorrelations, Monte Carlo errors, Geweke test, Raftery-Lewis test, and Heidelberger-Welch test are also available. For more information about convergence diagnostics, see the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures." You can request all the diagnostic tests by specifying DIAGNOSTICS=ALL. You can suppress all the diagnostic tests by specifying DIAGNOSTICS=NONE.

You can specify one or more of the following *keyword-list* options:

ALL

computes all diagnostic tests and statistics. You can combine the option ALL with any other specific tests to modify test options. For example, DIAGNOSTICS=(ALL AUTOCORR(LAGS=(1 5 35))) computes all tests by using default settings and autocorrelations at lags 1, 5, and 35.

AUTOCORR < (*autocorrelation-options*) >

AC < (*autocorrelation-options*) >

computes default autocorrelations at lags 1, 5, 10, and 50 for each variable. You can choose other lags by using the following *autocorrelation-option*:

LAGS=(*numeric-list*)

specifies autocorrelation lags. The *numeric-list* must take positive integer values.

ESS

computes the effective sample sizes (Kass et al. 1998) of the posterior samples of each parameter. It also computes the correlation time and the efficiency of the chain for each parameter. Small values of ESS might indicate a lack of convergence. For more information, see the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

GEWEKE < (*Geweke-options*) >

computes the Geweke spectral density diagnostics; this is a two-sample t-test between the first f_1 portion (as specified by the FRAC1= option) and the last f_2 portion (as specified by the FRAC2= option) of the chain. For more information, see the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures." By default, FRAC1=0.1 and FRAC2=0.5, but you can choose other fractions by using the following *Geweke-options*:

FRAC1=*value*

F1=*value*

specifies the beginning proportion of the Markov chain. By default, FRAC1=0.1.

FRAC2=*value*

F2=*value*

specifies the end proportion of the Markov chain. By default, FRAC2=0.5.

HEIDELBERGER < (*Heidel-options*) >

HEIDEL < (*Heidel-options*) >

computes the Heidelberger-Welch diagnostic (which consists of a stationarity test and a half-width test) for each variable. The stationary diagnostic test tests the null hypothesis that the posterior samples are generated from a stationary process. If the stationarity test is passed, a half-width test is then carried out. For more information, see the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

You can also specify suboptions, such as DIAGNOSTICS=HEIDELBERGER(EPS=0.05), as follows:

EPS=*value*
specifies a small positive number ϵ such that if the half-width is less than ϵ times the sample mean of the retaining iterations, the half-width test is passed. By default, EPS=0.1.

HALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the half-width test. By default, HALPHA=0.05.

SALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the stationarity test. By default, SALPHA=0.05.

MAXLAG=*n*
specifies the maximum number of autocorrelation lags to use to compute the effective sample size; for more information, see the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The value of *n* is also used in the calculation of the Monte Carlo standard error; see the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures." By default, MAXLAG=MIN(500, MCsample/4), where MCsample is the Markov chain sample size that is kept after thinning—that is, MCsample $= \left[\frac{\mathrm{NMC}}{\mathrm{NTHIN}}\right]$. If *n* is too low, you might observe significant lags and the effective sample size cannot be calculated accurately. A warning message appears in the SAS log, and you can increase either the MAXLAG= option or the NMC= option, accordingly. Specifying this option implies the ESS and MCSE options.

MCSE
MCERROR
computes the Monte Carlo standard error for the posterior samples of each parameter.

NONE
suppresses all the diagnostic tests and statistics. This option is not recommended.

RAFTERY <(*Raftery-options***)>**
RL <(*Raftery-options***)>**
computes the Raftery-Lewis diagnostic, which evaluates the accuracy of the estimated quantile ($\hat{\theta}_Q$ for a given $\mathrm{Q} \in (0, 1)$) of a chain. $\hat{\theta}_Q$ can achieve any degree of accuracy when the chain is allowed to run for a long time. The algorithm stops when the estimated probability $\hat{P}_Q = \Pr(\theta \leq \hat{\theta}_Q)$ reaches within $\pm$R of the value Q with probability S; that is, $\Pr(\mathrm{Q}-\mathrm{R} \leq \hat{\mathrm{P}}_Q \leq \mathrm{Q}+\mathrm{R}) = \mathrm{S}$. For more information, see the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

You can specify Q, R, S, and a precision level ϵ for a stationary test by specifying the following *Raftery-options*—for example, DIAGNOSTICS=RAFTERY(QUANTILE=0.05):

ACCURACY=*value*
R=*value*
specifies a small positive number as the margin of error for measuring the accuracy of estimation of the quantile. By default, ACCURACY=0.005.

EPS=*value*
specifies the tolerance level (a small positive number) for the stationary test. By default, EPS=0.001.

PROB=*value*
S=*value*
specifies the probability of attaining the accuracy of the estimation of the quantile. By default, PROB=0.95.

QUANTILE=*value*
Q=*value*
specifies the order (a value between 0 and 1) of the quantile of interest. By default, QUANTILE=0.025.

DIC
computes the deviance information criterion (DIC). DIC is calculated by using the posterior mean estimates of the parameters. For more information, see the section "Deviance Information Criterion (DIC)" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

GPU< (TPB=*n***) >**
enables PROC BCHOICE to use graphics processing units (GPUs) to perform parallel computations for MCMC sampling. Currently, this option is enabled for logit models and probit models. If you specify a nested logit model, the GPU option is ignored. To use this option, the system must have an NVIDIA GPU and a compatible NVIDIA driver installed. PROC BCHOICE uses CUDA 11.8, and the NVIDIA V100 and P100 GPUs are supported.

You can specify the following suboption:

TPB=*n*
specifies a positive integer that sets the number of threads per block (TPB) for GPU computations. The number of threads per block must be one of the following values: 64, 128, 256, or 512. If you specify an invalid value, the closest valid value is used. By default, TPB=256.

HITPROB
HITPROBABILITY
calculates the average of estimated probabilities of all chosen alternatives in the input data set. You can use this average as a measure of goodness of fit.

INF=*value*
specifies the numerical definition of infinity in PROC BCHOICE. By default, INF=1E15. For example, PROC BCHOICE considers 1E16 to be outside the support of the normal distribution and assigns a missing value to the log density evaluation. The minimum *value* that is allowed is 1E10, and the maximum *value* that is allowed is 1E27.

LOGPOST
calculates the logarithm of the posterior density of the parameters and the likelihood at each iteration. As a result, the OUTPOST= data set will contain the LOGLIKE and LOGPOST variables. You can specify this option for logit models and nested logit models, but not probit models.

MAXTUNE=*n*

specifies an upper limit for the number of proposal tuning loops for the random walk sampler. You can specify this option for logit models; it is ignored for other models. By default, MAXTUNE=6.

MCHISTORY=BRIEF | DETAILED | NONE

MCHIST=BRIEF | DETAILED | NONE

controls the display of the Metropolis sampling history. This option is ignored for nested logit and probit models. You can specify the following values:

BRIEF

produces a summary output for the tuning, burn-in, and sampling history tables. No tuning history table is produced if there is no tuning stage. The tables show the following when applicable:

- Scale shows the scale, or the range of the scales, that is used in each random walk Metropolis block that has a normal distribution.
- Acceptance Rate shows the acceptance rate, or the range of the acceptance rates, for each Metropolis block.

DETAILED

produces detailed output of the tuning, burn-in, and sampling history tables, including scale values, acceptance probabilities, and blocking information. No tuning history table is produced if there is no tuning stage. Use this option with caution, especially in random-effects models that have a large number of random-effects groups, because it can produce copious output.

NONE

produces none of the tuning history, burn-in history, and sampling history tables.

By default, MCHISTORY=BRIEF.

MINTUNE=*n*

specifies a lower limit for the number of proposal tuning loops for the random walk sampler. You can specify this option for logit models; it is ignored for other models. By default, MINTUNE=2.

NBI=*n*

specifies the number of burn-in iterations to perform before beginning to save parameter estimate chains. By default, NBI=500. For more information, see the section "Burn-In, Thinning, and Markov Chain Samples" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

NMC=*n*

specifies the number of iterations in the main simulation loop. If you specify a data set in the OUTPOST= option, this is the number of posterior samples that will be saved for each parameter. This is the MCMC sample size if THIN=1. By default, NMC=5000.

NOCLPRINT<=*n***>**

suppresses the display of the "Class Level Information" table if you do not specify *n*. If you specify *n*, the values of the classification variables are displayed for only those variables whose number of levels is less than *n*. Specifying *n* helps reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NTHREADS=*n*

NTHREAD=*n*

specifies the number of threads(CPUs) for analytic computations and simulation to run simultaneously on. Multi-threading is the concurrent execution of threads. When multi-threading is possible, you can realize substantial performance gains compared to the performance you get from sequential (single-threaded) execution. The more threads there are, the faster the computation runs. But do not specify a number that is more than the number of CPUs on the host where the analytic computations execute.

PROC BCHOICE performs two types of threading. In sampling fixed-effects parameters, PROC BCHOICE allocates data to different threads and accumulates values from each thread; in sampling of random-effects parameters, each thread generates a subset of these parameters simultaneously at each iteration. Most sampling algorithms are threaded. If you do not specify the NTHREADS= option or if you specify NTHREADS=0, the default number of threads is 1.

NTU=*n*

specifies the number of iterations to use in each proposal tuning phase for the random walk sampler. You can specify this option for logit models; it is ignored for other models. By default, NTU=500.

OUTPOST=*SAS-data-set*

OUT=*SAS-data-set*

specifies an output data set to contain the posterior samples of all parameters and the iteration numbers. It contains the log of the posterior density (LOGPOST) and the log likelihood (LOGLIKE) if you specify the LOGPOST option. By default, no OUTPOST= data set is created.

PLOTS < (*global-plot-options***) > < = *plot-request* < (***options***) > >**

PLOTS < (*global-plot-options***) > < = (***plot-request* < (***options***) > < . . . *plot-request* < (***options***) > >) >**

controls the display of diagnostic plots. You can request three types of plots: trace plots, autocorrelation function plots, and kernel density plots. By default, the plots are displayed in panels unless you specify the *global-plot-option* UNPACK. Also, when you specify more than one type of plot, the plots are grouped by parameter unless you specify the *global-plot-option* GROUPBY=TYPE. When you specify only one *plot-request*, you can omit the parentheses around it, as shown in the following example:

```
plots=none
plots(unpack)=trace
plots=(trace density)
```

If ODS Graphics is enabled but you do not specify the PLOTS= option, then PROC BCHOICE produces, for each parameter, a panel that contains the trace plot, the autocorrelation function plot, and the density plot. This is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

You can specify the following *global-plot-options*:

FRINGE

adds a fringe plot to the horizontal axis of the density plot.

GROUPBY=PARAMETER | TYPE

GROUP=PARAMETER | TYPE

specifies how the plots are grouped when there is more than one type of plot. By default, GROUPBY=PARAMETER. You can specify the following values:

TYPE

groups the plots by type.

PARAMETER

groups the plots by parameter.

LAGS=*n*

specifies the number of autocorrelation lags that are used in plotting the ACF graph. By default, LAGS=50.

SMOOTH

smooths the trace plot by using a fitted penalized B-spline curve (Eilers and Marx 1996).

UNPACKPANEL

UNPACK

unpacks all paneled plots, so that each plot in a panel is displayed separately.

You can specify the following *plot-requests*:

ALL

requests all types of plots. PLOTS=ALL is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

AUTOCORR

ACF

displays the autocorrelation function plots for the parameters.

DENSITY

D

KERNEL

K

displays the kernel density plots for the parameters.

NONE

suppresses the display of all plots.

TRACE

T

displays the trace plots for the parameters.

Consider a model that has four parameters, X1–X4. The following list shows which plots are produced for various option settings:

- PLOTS=(TRACE AUTOCORR) displays the trace and autocorrelation plots for each parameter side by side, with two parameters per panel:

Display 1	Trace(X1)	Autocorr(X1)
	Trace(X2)	Autocorr(X2)
Display 2	Trace(X3)	Autocorr(X3)
	Trace(X4)	Autocorr(X4)

- PLOTS(GROUPBY=TYPE)=(TRACE AUTOCORR) displays all the paneled trace plots, followed by panels of autocorrelation plots:

Display 1	Trace(X1)	
	Trace(X2)	
Display 2	Trace(X3)	
	Trace(X4)	
Display 3	Autocorr(X1)	Autocorr(X2)
	Autocorr(X3)	Autocorr(X4)

- PLOTS(UNPACK)=(TRACE AUTOCORR) displays a separate trace plot and a separate correlation plot, parameter by parameter:

Display 1	Trace(X1)
Display 2	Autocorr(X1)
Display 3	Trace(X2)
Display 4	Autocorr(X2)
Display 5	Trace(X3)
Display 6	Autocorr(X3)
Display 7	Trace(X4)
Display 8	Autocorr(X4)

- PLOTS(UNPACK GROUPBY=TYPE)=(TRACE AUTOCORR) displays all the separate trace plots, followed by the separate autocorrelation plots:

Display 1	Trace(X1)
Display 2	Trace(X2)
Display 3	Trace(X3)
Display 4	Trace(X4)
Display 5	Autocorr(X1)
Display 6	Autocorr(X2)
Display 7	Autocorr(X3)
Display 8	Autocorr(X4)

SCALE=*value*

controls the initial multiplicative scale to the covariance matrix of the proposal distribution for the random walk–based Metropolis algorithm for logit models. By default, SCALE=2.93. For more information, see the section "Scale Tuning" on page 1231.

SEED=*n*

specifies the random number seed. By default, SEED=0, and PROC BCHOICE gets a random number seed from the system clock. Negative seed values are treated as the default. The largest possible value for the seed is $2^{31} - 1$. Analyses that use the same nonzero seed are reproducible. The seed value is reported in the "Model Information" table.

STATISTICS < (*global-stats-options***) > = NONE | ALL |** *stats-request*

STATS < (*global-stats-options***) > = NONE | ALL |** *stats-request*

specifies options for posterior statistics. By default, PROC BCHOICE computes the posterior mean, standard deviation, quantiles, and two 95% credible intervals: equal-tail and highest posterior density (HPD). Other available statistics include the posterior correlation and covariance. For more information, see the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures." You can request all the posterior statistics by specifying STATS=ALL. You can suppress all the calculations by specifying STATS=NONE.

You can specify the following *global-stats-options*:

ALPHA=*numeric-list*

specifies the α level for the equal-tail and HPD intervals. The value of α must be between 0 and 0.5. By default, ALPHA=0.05.

PERCENT=*numeric-list*

PERCENTAGE=*numeric-list*

calculates the posterior percentages. The *numeric-list* contains values between 0 and 100. By default, PERCENTAGE=(25 50 75).

You can specify the following *stats-requests*:

ALL

computes all posterior statistics. You can combine the ALL option with any other options. For example, STATS(ALPHA=(0.02 0.05 0.1))=ALL computes all statistics by using the default settings and intervals at α levels of 0.02, 0.05, and 0.1.

BRIEF

computes the posterior means, standard deviations, and $100(1-\alpha)\%$ HPD credible interval for each variable. By default, ALPHA=0.05, but you can use the global ALPHA= option to request other values. This is the default output for posterior statistics.

CORR

computes the posterior correlation matrix.

COV

computes the posterior covariance matrix.

INTERVAL

INT

computes the $100(1-\alpha)\%$ equal-tail and HPD credible intervals for each variable. For more information, see the sections "Equal-Tail Credible Interval" on page 177 and "Highest Posterior Density (HPD) Interval" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures." By default, ALPHA=0.05, but you can use the global ALPHA= option to request other intervals of any probabilities.

NONE

suppresses all the statistics.

SUMMARY

SUM

computes the posterior means, standard deviations, and percentile points for each variable. By default, the 25th, 50th, and 75th percentile points are produced, but you can use the global PERCENT= option to request specific percentile points.

TARGACCEPT=*value*

specifies the target acceptance rate for the random walk–based Metropolis algorithm for logit models. For more information, see the section "Metropolis and Metropolis-Hastings Algorithms" on page 156 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The numeric *value* must be between 0.01 and 0.99. By default, TARGACCEPT=0.45 for one parameter; TARGACCEPT=0.35 for two, three, or four parameters; and TARGACCEPT=0.234 for more than four parameters (Roberts, Gelman, and Gilks 1997; Roberts and Rosenthal 2001).

THIN=*n*

NTHIN=*n*

controls the thinning rate of the simulation. PROC BCHOICE keeps every *n*th simulation sample and discards the rest. All posterior statistics and diagnostics are calculated by using the thinned samples. By default, THIN=1. For more information, see the section "Burn-In, Thinning, and Markov Chain Samples" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

TUNEWT=*value*

specifies the multiplicative weight used in updating the covariance matrix of the proposal distribution for the random walk sampler for logit models. The numeric *value* must be between 0 and 1. By default, TUNEWT=0.75. For more information, see the section "Covariance Tuning" on page 1232.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC BCHOICE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the BCHOICE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(***options***)** > . . . < *variable* < **(***options***)** > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. Response variables do not need to be specified in the CLASS statement.

The CLASS statement must precede the MODEL statement. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*. You can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING

DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC BCHOICE orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (., ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values of the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so this option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC BCHOICE interprets values of the ORDER= option:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. You can specify any of the *keywords* shown in the following table. The default is PARAM=REF. Design matrix columns are created from CLASS variables according to the corresponding coding schemes.

Value of PARAM=	Coding
EFFECT	Effect coding
GLM	Less-than-full-rank reference cell coding (this *keyword* can be used only in a global option)
REFERENCE **REF**	Reference cell coding

All parameterizations are full rank, except for the GLM parameterization. The REF= option in the CLASS statement determines the reference level for EFFECT and REFERENCE coding and for their orthogonal parameterizations. It also indirectly determines the reference level for a singular GLM parameterization through the order of levels.

If a PARAM= option is specified as a variable option for some variables, then any variables for which PARAM= is not specified use either the EFFECT parameterization if the global PARAM= option is not specified, or the full-rank parameterization indicated in the global PARAM= option if specified. If the global PARAM=GLM option is specified and PARAM= is also specified for some variables, GLM parameterization is used for all variables.

REF=*'level'* | *keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*:

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n* >

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

MODEL Statement

MODEL *response* < (*response-options*) > **=** < *fixed-effects* > < / *model-options* > **;**

The MODEL statement is required; it defines the response (dependent) variable and the fixed effects. The response variable indicates the chosen alternative in a choice set by the value 1 and the unchosen alternatives by the value 0 when the response is binary. In the MaxDiff choice experiment, a respondent can pick one best item and one worst item in each choice set, where there are three valid integer values: 1 for the best alternative, –1 for the worst alternative, and 0 for the ones in the middle. In the allocation choice setting, the response takes a percent of preference for each alternative and the sum of all percents in each choice set is 100%. Any value of the response variable that is valid, including missing, will be replace by zero. The

fixed-effects determine the **X** matrix of the model. The specification of effects is the same as in the GLIMMIX and MIXED procedures, where you do not specify random effects in the MODEL statement.

Table 30.3 summarizes the options available in the MODEL statement. These are subsequently discussed in detail in alphabetical order by option category.

Table 30.3 MODEL Statement Options

Option	Description
ALTERIDX=	Specifies the order of all alternatives in a choice set
CHOICESET=	Specifies the variables for defining a choice set
CHOICETYPE=	Specifies what values the choice response variable takes
COEFFPRIOR=	Specifies the prior of the regression coefficients
COVPRIOR=	Specifies the prior of the covariance parameter for a probit model
COVTYPE=	Specifies the structure of the covariance matrix of the error difference for a probit model
INIT=	Controls the generation of initial values of the regression coefficients
LAMBDAPRIOR=	Specifies the prior of the log-sum coefficients for a nested logit model
NEST=	Defines the nonoverlapping nests for a nested logit model
SAMELAMBDA	Constrains the log-sum coefficients to be the same for all the nests in a nested logit model
TYPE=	Specifies the type of the model

ALTERIDX=*variable*

specifies the order of all alternatives in a choice set for a nested logit or probit model. Specify this option to make sure the alternatives appear in the same order in all choice sets. When you do not specify this option, PROC BCHOICE assumes that the alternatives are in the same order. The *variable* should take increasing positive integers starting from 1 within each choice set. You need to specify this option only for a nested logit or probit model; PROC BCHOICE ignores it for a logit model.

CHOICESET=(*variables***)**

specifies one or more *variables* for defining the choice sets. You must specify how the choice sets are constructed, and you can use more than one variable. PROC BCHOICE does not sort by the values of the choice set variable; rather, it considers the data to be from a new choice set whenever the value of the choice set variable changes from the previous observation.

CHOICETYPE=*choice-type*

specifies what values the choice outcome variable takes. You can specify the following *choice-types*:

BINARY

specifies that the choice response variable has a binary integer value: 1 for a chosen alternative and 0 for an unchosen alternative. Each choice set has at least two alternatives and only one chosen alternative.

MAXDIFF

specifies that the choice response variable takes one of the possible three integer values: 1 for the best alternative, –1 for the worst alternative, and 0 for the ones in the middle. Each respondent is shown a set of the possible items to be evaluated and chooses one best item and one worst item in each choice set.

ALLOCATION < (WEIGHT=*n* **| WEIGHT=***variable***) >**

specifies that the choice response variable takes a percentage of preference for each alternative and that the sum of percentages in each choice set is 100%. You can specify the following suboptions:

WEIGHT=*n*

specifies a positive integer to indicate how many times the allocated percentages of each choice set in the data should be applied. For example, if the respondent were asked to imagine the next 20 purchase occasions and to allocate a percentage of those 20 purchases for each alternative, you would use a weight of 20. The same weight is used for all the choice sets in the data. By default, WEIGHT=10.

WEIGHT=*variable*

specifies a *variable* from the DATA= data set to indicate how many times the allocated percentages of a choice set should be counted. This suboption allows varying weights among the choice sets. The *variable* takes nonmissing positive integers. This *variable* should be the same within each choice set, because the weight is supposed to be applied at the choice set level and is not specific to any alternative in a choice set.

By default, CHOICETYPE=BINARY.

COEFFPRIOR < (*bounded-prior-options***) >=UNIFORM | NORMAL**

CPRIOR < (*bounded-prior-options***) >=UNIFORM | NORMAL**

specifies the prior distribution for the regression coefficients. The default is the normal prior $\mathrm{N}(\mathbf{0}, 10^2\mathbf{I})$, where $\mathbf{I}$ is the identity matrix. You can specify the following prior distributions:

UNIFORM

CONSTANT

specifies the uniform prior for the regression coefficients. The uniform prior is supported for logit models but not probit models. If you specify UNIFORM for a probit model, the normal prior $\mathrm{N}(\mathbf{0}, 10^2\mathbf{I})$ is used.

NORMAL < (*normal-prior-options*) >
specifies the normal prior for the regression coefficients. You can specify the following *normal-prior-options*, enclosed in parentheses:

INPUT=*SAS-data-set*
specifies a SAS data set that contains the mean and covariance information of the normal prior. The data set must have a _TYPE_ variable to represent the type of each observation and a variable for each regression coefficient. If the data set also contains a _NAME_ variable, the values of this variable are used to identify the covariances for the _TYPE_='COV' observations; otherwise, the _TYPE_='COV' observations are assumed to be in the same order as the explanatory variables in the MODEL statement. PROC BCHOICE reads the mean vector from the observation for which _TYPE_='MEAN' and reads the covariance matrix from observations for which _TYPE_='COV'. For an independent normal prior, the variances can be specified with _TYPE_='VAR'; alternatively, the precisions (inverse of the variances) can be specified with _TYPE_='PRECISION'.

VAR < =*c* >
specifies the normal prior $\mathrm{N}(\mathbf{0}, c\mathbf{I})$, where $\mathbf{I}$ is the identity matrix and *c* is a scalar.

You can specify the following *bounded-prior-options*, enclosed in parentheses:

LOWER=*value-list*
assigns lower bounds of the prior distribution for the regression coefficients, following the order of the coefficient list. A missing value (.) sets the lower bound for a coefficient to the default value, negative infinity.

UPPER=*value-list*
assigns upper bounds of the prior distribution for the regression coefficients, following the order of the coefficient list. A missing value (.) sets the upper bound for a coefficient to the default value, positive infinity.

For example, the following statement specifies a uniform prior and assigns the lower bounds –1, $-\infty$, –3 and the upper bounds 1, 2, ∞ to the first, second, and third coefficients in the model, respectively:

```
model y = x / choiceset=(ID Index)
      coeffprior(lower=(-1 . -3) upper=(1 2 .))=uniform;
```

By default, infinity is defined as 1E15. You can change it by using the INF= option in the PROC BCHOICE statement.

The length of the list must match the number of fixed-effects parameters. If the list length is less than the number of fixed-effects parameters, no constraints are set for the remaining parameters. If the list length is greater than the number of fixed-effects parameters, the extra bounds are ignored. RESTRICT statements are ignored when you specify a *bounded-prior-option* in the COEFFPRIOR option.

COVPRIOR=IWISHART < (*options*) >

specifies an inverse Wishart prior distribution, IWISHART(*a*,*b*), for the covariance matrix for the vector of error differences. For models that do not have a covariance matrix for the error differences (the logit and nested logit models), this option is ignored.

You can specify the following *options*, enclosed in parentheses:

DF=*a*

specifies the degrees of freedom of the inverse Wishart distribution. The default is the number of alternatives in the choice set plus 2, which is equivalent to the dimension of the covariance matrix of the error differences plus 3.

SCALE=*b*

specifies $b\mathbf{I}$ for the scale parameter of the inverse Wishart distribution, where $\mathbf{I}$ is the identity matrix. The default is the number of alternatives in the choice set plus 2.

COVTYPE=UN | VC

specifies the covariance structure of the error difference vector for a probit model. Although a variety of structures are available, most applications call for either COVTYPE=VC or COVTYPE=UN for the error difference vector. COVTYPE=VC (variance components) models a different variance component for each error term. COVTYPE=UN (unstructured) specifies a full structured covariance matrix. The unstructured form accommodates any pattern of correlation in addition to fitting a different variance component for each error difference term.

INIT=*keyword-list* | (*numeric-list*)

INITIAL=*keyword-list* | (*numeric-list*)

specifies options for generating the initial values for the coefficients parameters that are specified as *fixed-effects* in the MODEL statement. By default, INIT=POSTMODE for logit models and INIT=PRIORMODE for probit models. You can specify the following *keywords*:

LIST=*numeric-list*

assigns the numbers to be the initial values of the fixed effects in the corresponding list order. The length of the list must be the same as the number of fixed effects. For example, the following statement assigns the values 1, 2, and 3 to the first, second, and third coefficients in the model and prints the table of initial values:

```
model y = x / choiceset=(ID Index) init=(list=(1 2 3) pinit);
```

If the length of the list is less than the number of fixed effects, the initial value of each remaining parameter will be replaced by the corresponding default initial value. For example, the corresponding mode of the posterior density is used for a logit model. If the length of the list is greater than the number of fixed effects, the extra ones are ignored.

PINIT

tabulates initial values for the fixed effects. (By default, PROC BCHOICE does not display the initial values.)

POSTMODE
uses the mode of the posterior density as the initial value of the parameter, if you do not provide one. If the mode does not exist or if it is on the boundary of the support of the density, the mean value is used. If you specify POSTMODE for a probit model, where the posterior density is difficult to obtain, PROC BCHOICE resets it to PRIORMODE.

PRIORMODE
uses the mode of the prior density as the initial value of the parameter.

LAMBDAPRIOR=SEMIFLAT < (*options*) >
LPRIOR=SEMIFLAT < (*options*) >
specifies a semi-flat prior distribution (Lahiri and Gao 2002) for the log-sum coefficient, λ, for each nest in a nested logit model. For models that are not nested logit, this option is ignored.

You can specify the following *option*, enclosed in parentheses:

PHI=*a*
specifies the parameter ϕ of the semi-flat prior. By default, $\phi = 0.8$.

NEST=(*numeric-list***)**
defines the nonoverlapping nests for a nested logit model. For a nested logit model, you must specify the nests for all the alternatives in the choice set. Otherwise, the standard logit model is assumed. The number of values in the list should match the number of alternatives in the choice sets, and each of the actual values represents the nest that the particular alternative goes to. For example, NEST=(1 2 1 1 2) arranges the first, third, and fourth alternatives in the first nest and the second and fifth alternatives in the second nest. Currently, this option can accommodate only two-level nested logit models.

SAMELAMBDA
constrains the log-sum coefficients to be the same for all the nests in a nested logit model.

TYPE=*keyword*
specifies the type of the model. You can specify the following *keywords*:

LOGIT
specifies a standard logit model.

NLOGIT
specifies a nested logit model. If you do not also specify the NEST= option to define the nests, this option is ignored, and a standard logit model is fit.

PROBIT
specifies a probit model.

PREDDIST Statement

PREDDIST OUTPRED=*SAS-data-set* < **COVARIATES=***SAS-data-set* > **;**

The PREDDIST statement creates a new SAS data set that contains random samples from the posterior predictive distribution of the choice probabilities. It enables you to get the expected choice probabilities of all the alternatives in a choice set.

The posterior predictive distribution is the distribution of unobserved observations (prediction) conditional on the observed data. Let $\mathbf{Y}$ be the observed data, $\mathbf{X}$ be the covariates, $\boldsymbol{\theta}$ be the parameter, and $\mathbf{Y}_{\text{pred}}$ be the unobserved data. The posterior predictive distribution is defined as follows:

$$\begin{aligned} p(\mathbf{Y}_{\text{pred}}|\mathbf{Y},\mathbf{X}) &= \int p(\mathbf{Y}_{\text{pred}},\boldsymbol{\theta}|\mathbf{Y},\mathbf{X})d\boldsymbol{\theta} \\ &= \int p(\mathbf{Y}_{\text{pred}}|\boldsymbol{\theta},\mathbf{Y},\mathbf{X})p(\boldsymbol{\theta}|\mathbf{Y},\mathbf{X})d\boldsymbol{\theta} \end{aligned}$$

Assuming that the observed and unobserved data are conditionally independent given $\boldsymbol{\theta}$, the posterior predictive distribution can be further simplified as follows:

$$p(\mathbf{Y}_{\text{pred}}|\mathbf{Y},\mathbf{X}) = \int p(\mathbf{Y}_{\text{pred}}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{Y},\mathbf{X})d\boldsymbol{\theta}$$

The posterior predictive distribution is an integral of the likelihood function $p(\mathbf{Y}_{\text{pred}}|\boldsymbol{\theta})$ with respect to the posterior distribution $p(\boldsymbol{\theta}|\mathbf{Y})$. The PREDDIST statement generates samples from a posterior predictive distribution based on draws from the posterior distribution of $\boldsymbol{\theta}$.

You can specify the following options:

COVARIATES=*SAS-data-set*

names the SAS data set to contain the sets of explanatory variable values for which the predictions are established. This data set must contain data that has the same variables used in the model. If you omit the COVARIATES= option, the DATA= data set that is specified in the PROC BCHOICE statement is used instead.

NALTER=*n*

NALTERNATIVE=*n*

specifies the number of alternatives in a choice set in the COVARIATES= data set. All choice sets in the data must have the same number of alternatives. You must specify this option if a COVARIATES= data set is given.

OUTPRED=*SAS-data-set*

creates an output data set to contain the samples from the posterior predictive distribution of the choice probability that each alternative is chosen from a choice set. The output data set are in the order of either the COVARIATES= data set or the DATA= data set specified in the PROC statement. Multi-threading and data deletion might cause the order to change.

RANDOM Statement

RANDOM *random-effects* < / *options* > **;**

The RANDOM statement defines the random effects in the choice model. You can use this statement to specify traditional variance component models and to specify random coefficients, which can be classification or continuous. You can specify only one RANDOM statement. Include the TYPE= option to define the covariance structure. PROC BCHOICE does not include the intercept in the RANDOM statement.

Table 30.4 summarizes the *options* available in the RANDOM statement. All *options* are subsequently discussed in alphabetical order.

Table 30.4 Summary of RANDOM Statement Options

Option	Description
COVPRIOR=	Specifies the prior distribution for the covariance of the random effects
MONITOR	Displays the summary, diagnostics statistics and plots of individual random effects
NOOUTPOST	Suppresses saving of the posterior samples of subject-level random effects to the OUTPOST= data set
OUTSUBCOEFF=	Creates an output data set to contain the estimates of subject-level random-effects coefficients
PRINTCOV=	Requests that the procedure always output the summary, diagnostics statistics, and plots for the covariance of the random effects
REMEAN	Models the prior mean of the random effects
SUBJECT=	Identifies the subjects in the model
TYPE=	Specifies the covariance structure

You can specify the following *options* in the RANDOM statement after a slash (/).

COVPRIOR=IWISHART < (*options*) >

specifies an inverse Wishart prior distribution, **IWISHART**(*a*,*b*), for the covariance matrix of the random effects.

You can specify the following *options*, enclosed in parentheses:

DF=*a*

specifies the degrees of freedom of the inverse Wishart distribution. The default is the dimension of the covariance matrix of the random effects plus 3.

SCALE=*b*

specifies $b\mathbf{I}$ for the scale parameter of the inverse Wishart distribution, where $\mathbf{I}$ is the identity matrix. The default is the dimension of the covariance matrix of the random effects plus 3.

MONITOR

MONITOR=(*numeric-list*)

MONITOR=RANDOM (*number*)

displays results (summary, diagnostics statistics and plots) for the individual-level random-effects parameters. By default, PROC BCHOICE does not print results for individual-level random-effects parameters to save time and space. In models that have a large number of individual random effects (for example, tens of thousands individuals), it may take a long time to display the summary, diagnostics statistics, and plots for all the individual-level parameters, so be cautious when using the MONITOR option.

You can monitor a subset of the random-effects parameters. You can provide a numeric list of the SUBJECT indexes, or PROC BCHOICE can randomly choose a subset of all subjects for you.

To monitor a list of random-effects parameters for certain subjects, you can provide their indexes as follows:

```
random x / subject=index monitor=(1 to 5 by 2 23 57);
```

PROC BCHOICE outputs results of random effects for subjects 1, 3, 5, 23, and 57. PROC BCHOICE can also randomly choose a subset of all the subjects to monitor, if you submit a statement such as the following:

```
random x / subject=index monitor=(random(12));
```

PROC BCHOICE outputs results of random effects for 12 randomly selected subjects. You control the sequence of the random indexes by specifying the SEED= option in the PROC BCHOICE statement.

When you specify the MONITOR option, it uses the specification of the STATISTICS= and PLOTS= options in the PROC BCHOICE statement.

NOOUTPOST

suppresses storing the posterior samples of individual random-effects parameters in the OUTPOST= data set. By default, PROC BCHOICE outputs the posterior samples of all random-effects parameters to the OUTPOST= data set. You can use the NOOUTPOST option to not save the random-effects parameters. In models that have a large number of individual random-effects (for example, tens of thousands individuals), PROC BCHOICE can run faster if it does not save the posterior samples of all the individual random-effects.

When you specify both the NOOUTPOST option and the MONITOR option, PROC BCHOICE outputs the list of variables that are monitored.

There is a limit on the maximum number of variables that can be saved to an OUTPOST= data set. If you run a large-scale random-effects model in which the number of parameters exceeds that limit, the NOOUTPOST option is invoked automatically and PROC BCHOICE does not save the individual random-effects draws to the output data set. You can use the MONITOR option to select a subset of the parameters to store in the OUTPOST= data set.

OUTSUBCOEFF

OUTSUBUTL

creates an output data set to contain the estimates of subject-level (individual) random-effects coefficients. PROC BCHOICE saves the posterior means of all individual random effects by subject to this data set. The columns in this data set are the random effects that are listed in the RANDOM statement, and the rows are the subjects that are specified in the SUBJECT= command of the RANDOM statement.

PRINTCOV

requests to always output the summary, diagnostics statistics and plots for the parameters in the covariance matrix of the random effects. By default, PROC BCHOICE produces the summary, diagnostics statistics and plots for the parameters in the covariance matrix of the random effects. However, if there are too many parameters, greater than 36, in the covariance matrix of random effects, then the summary, diagnostics statistics and plots are not displayed. You can use this option to have them printed if you would like to.

REMEAN

requests to estimate the mean of the individual random effects, so that the mean of the random effects, $\bar{\boldsymbol{\gamma}}$, is not assumed to be zero and is estimated. This option is recommended when there are no fixed effects specified in the MODEL statement.

You can also model the mean of the random effects as a function of individual characteristics, such as demographic variables. Rossi, McCulloch, and Allenby (1996) and Rossi, Allenby, and McCulloch (2005) propose adding another layer of flexibility to the random-effects-only model by allowing heterogeneity that is driven by observable (demographic) characteristics of the individuals. Effects specified in the REMEAN= option are not supposed to change within each SUBJECT= level in the RANDOM statement. PROC BCHOICE takes only the first value of all effects in the REMEAN= option within each SUBJECT= level.

The following REMEAN=(AGE GENDER) option in the RANDOM statement estimates the mean of the random effect, X, and models the mean as a function of Age and Gender:

```
random X / subject=Index remean=(Age Gender);
```

SUBJECT=*effect*

SUB=*effect*

identifies the subjects in the model for the random effects. A set of random effects will be estimated for each subject. PROC BCHOICE assumes complete independence across subjects; thus, for the RANDOM statement, the SUBJECT= option produces a block-diagonal structure that has identical blocks. Specifying a subject effect is equivalent to nesting all other effects in the RANDOM statement within the subject effect.

The *effect* in the RANDOM statement must be specified in the CLASS list. The *effect* can be continuous variables.

PROC BCHOICE does not sort by the values of the SUBJECT= variable; rather, it considers the data to be from a new subject or group whenever the value of the variable changes from the previous observation.

TYPE=UN | VC

specifies the covariance structure. Although a variety of structures are available, most applications call for either TYPE=VC or TYPE=UN. The TYPE=VC (variance components) option, which is the default structure, models a different variance component for each random effect. The TYPE=UN (unstructured) specifies a full structured covariance matrix for the random effects. The unstructured form accommodates any pattern of correlation between the random effects in addition to fitting a different variance component for each random effect.

RESTRICT Statement

RESTRICT < *'label'* > *fixed-effect operand operator* < *value* > **;**

RESTRICT < *'label'* > *fixed-effect constraint-list* **;**

The RESTRICT statement enables you to specify restrictions on a *fixed-effect* that is specified in a preceding MODEL statement for a logit model. You can specify either a boundary requirement or order constraints on the *fixed-effect*. Specify one RESTRICT statement for each boundary requirement or order constraint. You must specify all boundary requirements before any order constraints. PROC BCHOICE incorporates the specified restrictions in the analysis. The RESTRICT statement is not available for a nested logit or probit model, nor for any random effects.

For either type of restriction, you can specify an optional *label*, surrounded by single quotation marks, and you must specify the following argument:

fixed-effect

specifies a *fixed-effect* that matches the one specified in the preceding MODEL statement. You can specify only one *fixed-effect*. Use multiple RESTRICT statements if you need to specify multiple restrictions for a single *fixed-effect* or if you need to specify restrictions on multiple *fixed-effects*.

Use the following syntax to specify a boundary requirement:

RESTRICT < *'label'* > *fixed-effect operand operator* < *value* > **;**

You must specify the following arguments in addition to the *fixed-effect*:

operand

specifies what the operator applies to. For a continuous *fixed-effect*, specify 1; for a classification *fixed-effect*, specify a string of 0's and one 1, where 1 indicates the level of the *fixed-effect* that the operator applies to.

operator

specifies one of the following mathematical operators: **>**, **<**, **>=**, **<=**.

You can also specify the following optional argument:

value

specifies a *value* to which the operator applies. The default is 0.

The following statements specify three boundary requirements:

```
proc bchoice;
   class Brand Subj;
   model choice = Brand Price / choiceset = (Subj);
   restrict Brand 1 0 0 0 > 0;
   restrict Brand 0 1 0 0 < 1;
   restrict Price 1 < -1;
run;
```

The first RESTRICT statement restricts the part-worth (regression coefficient) that is associated with the first level of the variable Brand to be positive. The second RESTRICT statement restricts the part-worth that is associated with the second level of the variable Brand to be less than 1. The third RESTRICT statement restricts the part-worth that is associated with the variable Price to be less than –1. You can specify a boundary restriction for only one parameter in one RESTRICT statement. If you want to specify both an upper bound and a lower bound for a parameter, use two RESTRICT statements.

Use the following syntax to specify an order constraint:

RESTRICT < *'label'* > *fixed-effect constraint-list* **;**

You must specify the following argument in addition to the *fixed-effect*:

constraint-list
: specifies a list of order constraints for the classification variable that is specified in the preceding CLASS statement. The *constraint-list* can contain integers 0 through the total number of estimable levels for the classification variable, where 0 indicates a level that is ignored, 1 indicates the level that has the smallest part-worth (regression coefficient), 2 indicates the level that has the second-smallest part-worth, and so on.

In the following statements, the RESTRICT statement specifies that the first level of the variable Brand has the second-smallest part-worth, the second level has the smallest part-worth, the third level is ignored, and the fourth level has the largest part-worth:

```
proc bchoice;
   class Brand Subj;
   model choice = Brand Price / choiceset = (Subj);
   restrict Brand 2 1 0 3;
run;
```

Because reference coding is used, the fifth level of the variable is the reference level and is normalized to 0. No order index is needed for the reference level.

When you specify a constraint for a *fixed-effect*, the corresponding parameter is transformed to an unconstrained variable behind the scenes. The transformation is based on the constraint.

For example, if a *fixed-effect* β is specified to have a lower bound a, then β is transformed to an unbounded variable α:

$$\alpha = \log(\beta - a)$$

The inverse of the lower-bound transformation maps an unbounded variable α to a variable β that is bounded below by a:

$$\beta = \exp(\alpha) + a$$

The normal prior distribution is assumed on the unconstrained parameters. PROC BCHOICE samples the unconstrained parameters first and then exponentiates the unconstrained parameters when they enter the utility function.

Suppose you specify the following order constraint on a set of *fixed-effects*:

$$\beta_1 < \beta_2 < \cdots < \beta_K$$

PROC BCHOICE maps them to an unconstrained set of variables $(\alpha_1, \alpha_2, \ldots, \alpha_K)$ by setting

$$\alpha_k = \begin{cases} \beta_1 & \text{if } k = 1 \\ \log(\beta_k - \beta_{k-1}) & \text{if } 1 < k \leq K \end{cases}$$

Then, the inverse of this transformation is defined by the recursion:

$$\beta_k = \begin{cases} \alpha_1 & \text{if } k = 1 \\ \exp(\alpha_k) + \beta_{k-1} & \text{if } 1 < k \leq K \end{cases}$$

Details: BCHOICE Procedure

Discrete Choice Models

Discrete choice models are used in marketing research to model decision makers' choices among alternative products and services. The decision makers might be people, households, companies and so on, and the alternatives might be products, services, actions, or any other options or items about which choices must be made (Train 2009). The collection of alternatives that are available to the decision makers is called a choice set.

Discrete choice models are derived under the assumption of utility-maximizing behavior by the decision maker. When individuals are asked to make one choice among a set of alternatives, they usually determine the level of utility that each alternative offers. The utility that individual i obtains from alternative j among J alternatives is denoted as

$$u_{ij} = v_{ij} + \epsilon_{ij}, \quad i = 1, \ldots, N, \text{ and } j = 1, \ldots, J$$

where the subscript i is an index for the individuals, the subscript j is an index for the alternatives in a choice set, v_{ij} is a nonstochastic utility function that relates those observed factors to the utility, and ϵ_{ij} is the error component that captures the unobserved characteristics of the utility. In discrete choice models, the observed part of the utility function is assumed to be linear in the parameters,

$$v_{ij} = \mathbf{x}'_{ij}\boldsymbol{\beta}$$

where $\mathbf{x}_{ij}$ is a p-dimensional design vector of observed attribute levels that relate to alternative j and $\boldsymbol{\beta}$ is the corresponding vector of fixed regression coefficients that indicate the utilities or part-worths of the attribute levels.

Decision makers choose the alternative that gives them the greatest utility. Let $\mathbf{y}_i$ be the multinomial response vector for the ith individual. The value y_{ij} takes 1 if the jth component of $\mathbf{u}_i = (u_{i1}, \ldots, u_{iJ})$ is the largest, and 0 otherwise:

$$\begin{aligned} u_{ij} &= \mathbf{x}'_{ij}\boldsymbol{\beta} + \epsilon_{ij} \\ y_{ij} &= \begin{cases} 1 & \text{if } u_{ij} \geq \max(\mathbf{u}_i) \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

The probability that the individual i chooses alternative j is

$$\begin{aligned} P(y_{ij} = 1) &= \Pr\left(u_{ij} > u_{ik} \text{ for all } k \neq j\right) \\ &= \Pr\left(v_{ij} + \epsilon_{ij} > v_{ik} + \epsilon_{ik} \text{ for all } k \neq j\right) \\ &= \Pr\left(\epsilon_{ik} - \epsilon_{ij} < v_{ij} - v_{ik} \text{ for all } k \neq j\right) \\ &= \int_{\epsilon} \mathrm{I}(\epsilon_{ik} - \epsilon_{ij} < v_{ij} - v_{ik} \text{ for all } k \neq j) f(\boldsymbol{\epsilon}_i) d\boldsymbol{\epsilon}_i \end{aligned}$$

where I(.) is the indicator function and $f(\boldsymbol{\epsilon}_i)$ denotes the joint density of the error vector $\boldsymbol{\epsilon}_i = (\epsilon_{i1}, \ldots, \epsilon_{iJ})$. Different specifications about the density result in different types of choice models, as detailed in the next section. Logit and nested logit models have a closed form for this integral, whereas a probit model does not.

Types of Choice Models

Logit

In the multinomial logit (MNL) model, each ϵ_{ij} $(j = 1, \ldots, J)$ is independently and identically distributed (iid) with the Type I extreme-value distribution, $\exp(-\exp(-\epsilon_{ij}))$, which is also known as the Gumbel distribution. The essential part of the model is that the unobserved parts of utility are uncorrelated among all alternatives, in addition to having the same variance. This assumption provides a convenient logit form for the choice probability (McFadden 1974):

$$P(y_{ij} = 1) = \frac{\exp(\mathbf{x}'_{ij}\boldsymbol{\beta})}{\sum_{k=1}^{J} \exp(\mathbf{x}'_{ik}\boldsymbol{\beta})}, \quad i = 1, \ldots, N \text{ and } j = 1, \ldots, J$$

The MNL likelihood is formed by the product of the N independent multinomial distributions:

$$p(\mathbf{Y}|\boldsymbol{\beta}) = \prod_{i=1}^{N} \prod_{j=1}^{J} P(y_{ij} = 1)^{y_{ij}}$$

The MNL model is by far the easiest and most widely used discrete choice model. It is popular because the formula for the choice probability has a closed form and is readily interpretable. In his Nobel lecture, McFadden (2001) tells a history of the development of this path-breaking model.

Bayesian analysis for the MNL model requires the specification of a prior over the coefficient parameters $\boldsymbol{\beta}$ and computation of the posterior density. It is convenient to use a normal prior on $\boldsymbol{\beta}$, $\pi(\boldsymbol{\beta}) = \mathrm{N}(\bar{\boldsymbol{\beta}}, \boldsymbol{\Omega}_{\boldsymbol{\beta}})$. When the investigator has no strong prior beliefs about the location of the parameters, it is recommended that diffuse, but proper, priors be used.

The posterior density of the parameter $\boldsymbol{\beta}$ is

$$p(\boldsymbol{\beta}|\mathbf{Y}) \propto p(\mathbf{Y}|\boldsymbol{\beta})\pi(\boldsymbol{\beta})$$

Because discrete choice logit models fall under the framework of a generalized linear model (GLM) with a logit link, you can use Metropolis-Hastings sampling with the Gamerman approach for the sampling procedure. This approach extends the usual iterative weighted least squares method to include a sampling step based on the Metropolis-Hastings algorithm for GLMs. For more information about the Gamerman approach, see the section "Gamerman Algorithm" on page 1229. You can also incorporate this methodology for a hierarchical structure that has random effects. For more information, see the section "Logit with Random Effects" on page 1223. The Gamerman approach is the default sampling method when a direct sampling that uses a closed-form posterior distribution cannot be attained. The random walk Metropolis is also available when you specify ALGORITHM=RWM in the PROC BCHOICE statement.

The iid assumption about the error terms of the utilities of the different alternatives imposes the independence from irrelevant alternatives (IIA) property, which implies proportional substitution across alternatives. Sometimes this property might seem restrictive or unrealistic, prompting the need for more flexible models to overcome it.

Nested Logit

You derive the nested logit model by allowing the error components to be identical but nonindependent. Instead of being independent Type I extreme-value errors, the errors are assumed to have a generalized extreme-value (GEV) distribution. This model generalizes the standard logit model to allow for particular patterns of correlation in unobserved utility (McFadden 1978).

In a nested logit model, all the alternatives in a choice set can be partitioned into nests in such a way that the following conditions are true:

- The probability ratio of any two alternatives that are in the same nest is independent of the existence of all other alternatives. Hence, the IIA assumption holds within each nest.
- The probability ratio of any two alternatives that are in different nests is not independent of the existence of other alternatives in the two nests. Hence, the IIA assumption does not hold between different nests.

A nested logit example from Train (2009) best explains how a nested logit model works. Suppose the alternatives for commuting to work are driving an auto alone, carpooling, taking the bus, and taking the train. If you remove any alternative, the probabilities of the other alternatives increase. But by what percentage would each probability increase? Suppose the changes in probabilities occur as shown in Table 30.5.

Table 30.5 Example of IIA Holding within Nests of Alternatives

		Probabilities with One Alternative Removed			
Alternative	**Original Prob.**	**Auto Alone**	**Carpool**	**Bus**	**Train**
Auto alone	0.40	–	0.45(+12.5%)	0.52(+30.0%)	0.48(+20.0%)
Carpool	0.10	0.20(+100%)	–	0.13(+30.0%)	0.12(+20.0%)
Bus	0.30	0.48(+60.0%)	0.33(+10.0%)	–	0.40(+33.0%)
Train	0.20	0.32(+60.0%)	0.22(+10.0%)	0.35(+70.0%)	–

In Table 30.5, the probabilities for bus and train increase by the same proportion whenever you remove one of the other alternatives. Therefore, IIA is valid between bus and train, so they can be placed in the same nest called "transit." Similarly, auto alone and carpool are placed in one nest called "auto." IIA does not hold between these two nests.

A convenient way to represent the model and substitution patterns is to draw a tree diagram, where each branch denotes a subset of alternatives within which IIA holds and where each leaf on each branch denotes an alternative. The tree diagram for the commuting-to-work example is shown in Figure 30.10.

Figure 30.10 Two-Level Decision Tree

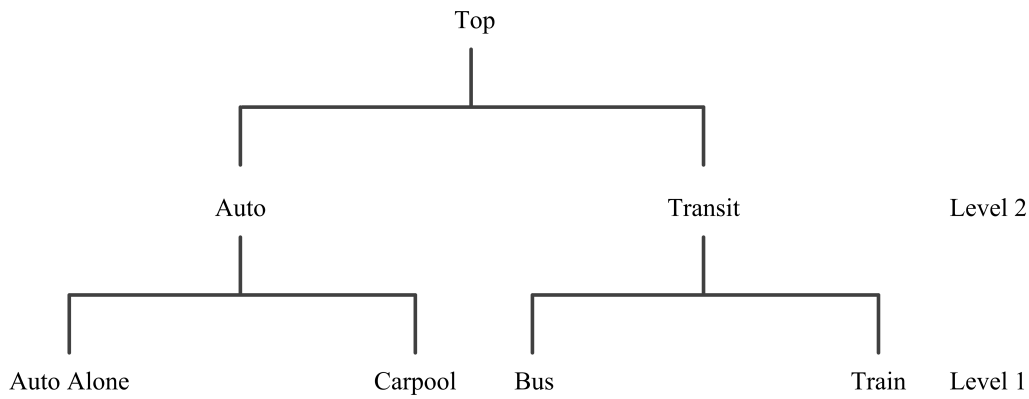

Although decision trees in nested logit analyses, such as the one shown in Figure 30.10, are often interpreted as implying that the higher-level decisions are made first, followed by decisions at lower levels, no such temporal ordering is necessarily required. A good way to think about nested logit models is that they are appropriate when there are groups of alternatives that are similar to each other in unobserved ways. In other words, nested logit models are appropriate when there is correlation between the alternatives in each nest but no correlation between the alternatives in different nests. Currently, PROC BCHOICE considers only two-level nested logit models.

In mathematical notation, let the set of alternatives be partitioned into K nonoverlapping subsets (nests) that are denoted by $S_1, S_2, \ldots, S_K$. The nested logit model is derived by assuming that the vector of the unobserved part of the utility, $\boldsymbol{\epsilon}_i = (\epsilon_{i1}, \ldots, \epsilon_{iJ})$, has a cumulative distribution

$$\exp(-\sum_{k=1}^{K}(\sum_{j \in S_k} \exp(-\epsilon_{ij}/\lambda_k))^{\lambda_k})$$

This is a type of generalized extreme value (GEV) distribution. For a standard logit model, each ϵ_{ij} is iid with an extreme-value distribution, whereas for a nested logit model, the marginal distribution of ϵ_{ij} is an extreme-value distribution. The ϵ_{ij} are correlated within nests. If alternatives j and m belong to the same nest, then ϵ_{ij} is correlated with ϵ_{im}. But if any two alternatives are in different nests, their unobserved part of utility is still independent. The parameter λ_k measures the degree of independence among alternatives in nest k. The higher the value of λ_k is, the less correlation there is. However, the correlation is actually more complicated than the parameter λ_k. The equation $\lambda_k = 1$ represents no correlation in nest k. If $\lambda_k = 1$ for all nests, the nested logit model reduces to the standard logit model.

The choice probability for alternative $j \in S_k$ has a closed form:

$$P(y_{ij} = 1) = \frac{\exp(\mathbf{x}'_{ij}\boldsymbol{\beta}/\lambda_k)(\sum_{m \in S_k} \exp(\mathbf{x}'_{im}\boldsymbol{\beta}/\lambda_k))^{\lambda_k - 1}}{\sum_{l=1}^{K}(\sum_{m \in S_l} \exp(\mathbf{x}'_{im}\boldsymbol{\beta}/\lambda_l))^{\lambda_l}}$$

From this form, the nested logit likelihood is derived as the product of the N multinomial distributions:

$$p(\mathbf{Y}|\boldsymbol{\beta},\boldsymbol{\lambda}) = \prod_{i=1}^{N}\prod_{j=1}^{J} P(y_{ij}=1)^{y_{ij}}$$

A prior for $\boldsymbol{\lambda} = (\lambda_1,\ldots,\lambda_K)$ is needed in addition to a prior for the coefficient parameters $\boldsymbol{\beta}$. The λ_k in nest k is often called the log-sum coefficient. The value of λ_k must be positive for the model to be consistent with utility-maximizing behavior. If $\lambda_k \in [0,1]$ for all k, the model is consistent with utility maximization for all possible values of the explanatory covariates, whereas the model is consistent only for some range of the covariates but not for all values if $\lambda_k > 1$. Kling and Herriges (1995) and Herriges and Kling (1996) suggest tests of consistency of nested logit with utility maximization when $\lambda_k > 1$. Train, McFadden, and Ben-Akiva (1987) show an example of models for which $\lambda_k > 1$. A negative value of λ_k should be avoided because the model will be inconsistent and imply that consumers are choosing the alternative to minimize their utilities.

For the nested logit model, noninformative priors are not ideal for $\boldsymbol{\lambda}$. Flat priors on different versions of the parameter space can yield different posterior distributions.

Lahiri and Gao (2002) suggest the following semi-flat priors by using the parameter ϕ for each λ:

$$\pi(\lambda) = \begin{cases} 0 & \text{if } \lambda \le 0 \\ \phi & \text{if } 0 < \lambda < 1 \\ \phi\exp[\frac{\phi}{1-\phi}(1-\lambda)] & \text{if } \lambda \ge 1 \end{cases}$$

PROC BCHOICE uses this prior with a default value of 0.8 for ϕ. You can specify other values for ϕ. Sims' priors are also attractive with various s values as follows:

$$\pi(\lambda) = \begin{cases} 0 & \text{if } \lambda \le 0 \\ s\lambda^{s-1}\exp(-\lambda^s) & \text{if } \lambda > 0 \end{cases}$$

In addition, a gamma or beta distribution is a good choice for the prior of $\boldsymbol{\lambda}$. However, PROC BCHOICE supports only semi-flat priors.

When the priors for the parameters $(\boldsymbol{\beta},\boldsymbol{\lambda})$ are specified, the posterior density is as follows:

$$p(\boldsymbol{\beta},\boldsymbol{\lambda}|\mathbf{Y}) \propto p(\mathbf{Y}|\boldsymbol{\beta},\boldsymbol{\lambda})\pi(\boldsymbol{\beta})\pi(\boldsymbol{\lambda})$$

Another generalization of the conditional logit model, the heteroscedastic extreme-value (HEV) model, is obtained by allowing independent but nonidentical errors that are distributed with a Type I extreme-value distribution (Bhat 1995). The HEV model permits different variances on the error components of utility across the alternatives. Currently, PROC BCHOICE does not consider this type of model.

Probit

The multinomial probit (MNP) model is derived when the errors, $\boldsymbol{\epsilon}_i' = (\epsilon_{i1},\epsilon_{i2},\ldots,\epsilon_{iJ})$, have a multivariate normal (MVN) distribution with a mean vector of $\mathbf{0}$ and a covariance matrix $\boldsymbol{\Sigma}$.

$$\begin{aligned} u_{ij} &= \mathbf{x}_{ij}'\boldsymbol{\beta} + \epsilon_{ij}, \quad \boldsymbol{\epsilon}_i \sim \mathrm{N}(0,\boldsymbol{\Sigma}), \quad i=1,\ldots,N \text{ and } j=1,\ldots,J \\ y_{ij} &= \begin{cases} 1 & \text{if } u_{ij} \ge \max(\mathbf{u}_i) \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where $\mathbf{\Sigma}$ is the J × J covariance matrix of the error vector. For a full covariance matrix, any pattern of correlation and heteroscedasticity can be accommodated. Thus, this model fits a very general error structure.

However, the choice probability, from which the likelihood is derived, does not have a closed form. McCulloch and Rossi (1994) propose an algorithm that is a multivariate version of the probit regression algorithm of Albert and Chib (1993) and construct a Gibbs sampler that is based on a Markov chain to draw directly from the exact posteriors of the MNP model. This approach avoids direct evaluation of the likelihood. Hence, it avoids problems that are associated with calculating choice probabilities that affect both the standard likelihood and the method of simulated moments approaches.

To avoid the parameter identification problems, a usual practice is to take differences against one particular alternative, because only differences in utility matter. Suppose you take differences against the last alternative in the choice set and you define $w_{ij} = u_{ij} - u_{iJ}, \tilde{\mathbf{x}}_{ij} = \mathbf{x}_{ij} - \mathbf{x}_{iJ}$, $\tilde{\epsilon}_{ij} = \epsilon_{ij} - \epsilon_{iJ}$, and $\tilde{\boldsymbol{\epsilon}}_i' = (\tilde{\epsilon}_{i1}, \tilde{\epsilon}_{i2}, \ldots, \tilde{\epsilon}_{iJ-1})$.

$$
\begin{aligned}
w_{ij} &= \tilde{\mathbf{x}}_{ij}'\boldsymbol{\beta} + \tilde{\epsilon}_{ij}, \quad \tilde{\boldsymbol{\epsilon}}_i \sim \mathrm{N}(0, \tilde{\mathbf{\Sigma}}), \quad i = 1, \ldots, N \text{ and } j = 1, \ldots, J-1 \\
y_{ij} &= \begin{cases} 1 & \text{if } w_{ij} \geq \max(0, \mathbf{w}_{i,-j}) \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

where $\tilde{\mathbf{\Sigma}}$ is the $(J-1) \times (J-1)$ covariance matrix of the vector of error differences, $\tilde{\boldsymbol{\epsilon}}_i$, and $\mathbf{w}_{i,-j} = (w_{i1}, \ldots, w_{i(j-1)}, w_{i(j+1)}, \ldots, w_{i(J-1)})$. The dimension has been reduced to $J-1$. McCulloch and Rossi (1994) call w_{ij} the latent variable. The introduction of the latent variables in what is known as the data augmentation step makes the application of a Gibbs sampler feasible. The order in which the alternatives appear in a choice set is important, so you should arrange the alternatives to appear in the same order in all choice sets.

A normal prior is used for $\boldsymbol{\beta}$ and an inverse Wishart prior on $\tilde{\mathbf{\Sigma}}$:

$$
\begin{aligned}
\pi(\boldsymbol{\beta}) &= \mathrm{N}(\bar{\boldsymbol{\beta}}, \mathbf{\Omega}_{\boldsymbol{\beta}}) \\
\pi(\tilde{\mathbf{\Sigma}}) &= \text{inverse Wishart}(\nu, \mathbf{V})
\end{aligned}
$$

Then sampling is carried out consecutively from the following three groups of conditional posterior distributions:

$$
\begin{aligned}
&(1) \quad (w_{ij} | \mathbf{w}_{i,-j}, \boldsymbol{\beta}, \tilde{\mathbf{\Sigma}}, \mathbf{Y}), \quad i = 1, \ldots, N \text{ and } j = 1, \ldots, J-1 \\
&(2) \quad (\boldsymbol{\beta} | \mathbf{W}, \tilde{\mathbf{\Sigma}}, \mathbf{Y}) \\
&(3) \quad (\tilde{\mathbf{\Sigma}} | \mathbf{W}, \boldsymbol{\beta}, \mathbf{Y})
\end{aligned}
$$

where $\mathbf{W}$ is obtained by stacking all $\mathbf{w}_i$. All three groups of conditional distributions have closed forms that are easily drawn from. Conditional (1) is truncated normals, (2) is a regular normal, and (3) is an inverse Wishart distribution. This sampling avoids the tuning stage that is required by most Metropolis-Hastings algorithms, and its analytically tractable complete data likelihood also makes it easy to embed probit models into more elaborate hierarchical structures, such as random-effects models. For more information, see McCulloch and Rossi (1994).

Originally, the multinomial probit model required burdensome computation compared to a family of multinomial choice models that are derived from the Gumbel distributed utility function, because the computation involves multidimensional integration (with dimension $J-1$) in the estimation process. In addition, the

multinomial probit model requires more parameters than other multinomial choice models. However, it fully relaxes the restrictive IIA assumption that is imposed in logit models and allows any pattern of substitution among alternatives. Because of the breakthrough Bayesian methods that Albert and Chib (1993) and McCulloch and Rossi (1994) introduced, the probit model can be estimated without the need to compute the choice probabilities. This advantage has greatly increased the popularity of the probit model.

Random Effects

Choice models that have random effects (or random coefficients) provide solutions to create individual-level or group-specific utilities. They are also referred as "mixed models" or "hybrid models." One of the greatest challenges in marketing research is to account for the diversity of preferences and sensitivities in the marketplace. Heterogeneity in individual preferences is the reason for differentiated product programs and market segmentation. As individual preferences become more diverse, it becomes less appropriate to consider the analyses in an aggregated way. Individual utilities are useful because they make segmentation easy and provide a way to detect groups. Because people have different preferences, it can be misleading to roll the whole sample together into a single set of utilities.

For example, imagine studying the popularity of a new brand. Some participants in the study who are interviewed love the new brand, whereas others dislike it. If you simply aggregate the data and look at the average, the conclusion is that the sample is ambivalent toward the new brand. This would be the least helpful conclusion that could be drawn, because it does not fit for anyone.

Choice models that have random effects generalize the standard choice models to incorporate individual-level effects. Let the utility that individual i obtains from alternative j in choice situation t ($t = 1, \ldots, T$) be

$$
\begin{aligned}
u_{ijt} &= \mathbf{x}'_{ijt}\boldsymbol{\beta} + \mathbf{z}'_{ijt}\boldsymbol{\gamma}_i + \epsilon_{ijt} \\
y_{ijt} &= \begin{cases} 1 & \text{if } u_{ijt} \geq \max(u_{i1t}, u_{i2t}, \ldots, u_{iJt}) \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

where y_{ijt} is the observed choice for individual i and alternative j in choice situation t; $\mathbf{x}_{ijt}$ is the fixed design vector for individual i and alternative j in choice situation t; $\boldsymbol{\beta}$ is the vector of fixed coefficients; $\mathbf{z}_{ijt}$ is the random design vector for individual i and alternative j in choice situation t; and $\boldsymbol{\gamma}_i$ is the vector of random coefficients for individual i that correspond to $\mathbf{z}_{ijt}$. Sometimes the random coefficients might be at a level different from the individual level. For example, it is common to assume that there are random coefficients at the household level or group level of participants. For the convenience of notation, random effects are assumed to be at the individual level.

In the random-effects model, it is assumed that each $\boldsymbol{\gamma}_i$ is drawn from a superpopulation and this superpopulation is normal, $\boldsymbol{\gamma}_i \sim \text{iid} \quad \text{N}(\mathbf{0}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}})$. An additional stage is added to the model where a prior for $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$ is specified:

$$
\begin{aligned}
\pi(\boldsymbol{\gamma}_i) &= \text{N}(\mathbf{0}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}) \\
\pi(\boldsymbol{\Omega}_{\boldsymbol{\gamma}}) &= \text{inverse Wishart}(\nu_0, \mathbf{V}_0)
\end{aligned}
$$

The covariance matrix $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$ characterizes the extent of heterogeneity among individuals. Large diagonal elements of $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$ indicate substantial heterogeneity in part-worths. Off-diagonal elements indicate patterns in the evaluation of attribute levels. For example, positive covariances specify pairs of attribute levels that tend

to be evaluated similarly across respondents. Product offerings that consist of these attribute levels are more strongly preferred or disliked by certain individuals.

In this setup, the prior mean is 0 for the random effects, meaning that the random effects either are truly around 0 or have been centered by the fixed effects. For random effects whose mean is not around 0, you can follow the usual practice of specifying them in the fixed effects. For example, if one random effect is price and you do not think that the population mean of price is around 0, then you should add price as a fixed effect as follows:

```
proc bchoice data=randeffdata;
   class subj set;
   model y = price / choiceset=(subj set);
   random  price / subject=subj;
run;
```

Thus, you obtain the estimate for the population mean of the price effect through the fixed effect, and you obtain the deviation from the population mean for each individual through random effects.

Allenby and Rossi (1999) and Rossi, Allenby, and McCulloch (2005) propose a hierarchical Bayesian random-effects model that is set up in a different way. In their model, there are no fixed effects but only random effects. This model, which is referred to as the random-effects-only model in the rest of this chapter, is as follows:

$$
\begin{aligned}
u_{ijt} &= \mathbf{z}'_{ijt}\boldsymbol{\gamma}_i + \epsilon_{ijt} \\
y_{ijt} &= \begin{cases} 1 & \text{if } u_{ijt} \geq \max(u_{i1t}, u_{i2t}, \ldots, u_{iJt}) \\ 0 & \text{otherwise} \end{cases} \\
\pi(\boldsymbol{\gamma}_i) &= \mathrm{N}(\bar{\boldsymbol{\gamma}}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}) \\
\pi(\boldsymbol{\Omega}_{\boldsymbol{\gamma}}) &= \text{inverse Wishart}(\nu_0, \mathbf{V}_0)
\end{aligned}
$$

where $\bar{\boldsymbol{\gamma}}$ is a mean vector of regression coefficients, which models the central location of distribution of the random coefficients; $\bar{\boldsymbol{\gamma}}$ represents the average part-worths across the respondents in the data. If you want to use this setup, specify the REMEAN option in the RANDOM statement to request estimation on $\bar{\boldsymbol{\gamma}}$ and do not specify any fixed effects in the MODEL statement.

Rossi, McCulloch, and Allenby (1996) and Rossi, Allenby, and McCulloch (2005) add another layer of flexibility to the random-effects-only model by allowing heterogeneity that is driven by observable (demographic) characteristics of the individuals. They model the prior mean of the random coefficients as a function of the individual's demographic variables (such as age and gender),

$$
\begin{aligned}
\pi(\boldsymbol{\gamma}_i) &= \mathrm{N}(\boldsymbol{\Gamma} d_i, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}) \\
\pi(\boldsymbol{\Omega}_{\boldsymbol{\gamma}}) &= \text{inverse Wishart}(\nu_0, \mathbf{V}_0)
\end{aligned}
$$

where d_i is a vector that consists of an intercept and some observable demographic variables. $\boldsymbol{\Gamma}$ is a matrix of regression coefficients, which affects the location of distribution of the random coefficients. $\boldsymbol{\Gamma}$ should be useful for identifying respondents who have part-worths that are different from those in the rest of the sample if some individual-level characteristics are included in d_i. This specification allows the preferences or intercepts to vary by both demographic variables and the slopes. If d_i consists of only an intercept, this model reduces to the previous one. For more information about how to sample $\boldsymbol{\Gamma}$, see Rossi, McCulloch, and Allenby (1996), Rossi, Allenby, and McCulloch (2005) (Section 2.12 in Chapter 2), and Rossi (2012).

Logit with Random Effects

The logit model with random effects consists of the fixed-coefficients parameters $\boldsymbol{\beta}$, the random-coefficients parameters $\boldsymbol{\gamma}_i$, and the covariance parameters for the random coefficients $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$. You can use the Metropolis-Hastings sampling with Gamerman approach to draw samples through the following three conditional posterior distributions:

(1) $(\boldsymbol{\beta}|\boldsymbol{\gamma}_i, \mathbf{Y})$
(2) $(\boldsymbol{\gamma}_i|\boldsymbol{\beta}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}, \mathbf{Y}) \quad i = 1, \ldots, N$
(3) $(\boldsymbol{\Omega}_{\boldsymbol{\gamma}}|\boldsymbol{\gamma}_i, \mathbf{Y})$

All chains are initialized with random effects that are set to 0 and a covariance matrix that is set to an identity matrix. Updating is done first for the fixed effects, $\boldsymbol{\beta}$, as a block to position the chain in the correct region of the parameter space. Then the random effects are updated, and finally the covariance of the random effects is updated. For more information, see Gamerman (1997) and the section "Gamerman Algorithm" on page 1229.

The hierarchical Bayesian random-effects-only model as proposed in Allenby and Rossi (1999) and Rossi, Allenby, and McCulloch (2005) contains the random-coefficients parameters $\boldsymbol{\gamma}_i$, the population mean of the random-coefficients parameters $\bar{\boldsymbol{\gamma}}$, and the covariance parameters for the random coefficients $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$. The sampling can be carried out by the following conditional posteriors:

(1) $(\boldsymbol{\gamma}_i|\bar{\boldsymbol{\gamma}}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}, \mathbf{Y}) \quad i = 1, \ldots, N$
(2) $(\bar{\boldsymbol{\gamma}}|\boldsymbol{\gamma}_i, \boldsymbol{\Omega}_{\boldsymbol{\gamma}})$
(3) $(\boldsymbol{\Omega}_{\boldsymbol{\gamma}}|\boldsymbol{\gamma}_i, \bar{\boldsymbol{\gamma}})$

The second and third conditional posteriors are easy to draw because they have direct sampling distributions: the second has a normal distribution with a mean of $\sum_{i=1}^{N} \boldsymbol{\gamma}_i/N$ and a covariance of $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}/N$; the third is an inverse Wishart$(\nu_0 + N, \mathbf{V}_0 + S)$, where $S = \sum_{i=1}^{N}(\boldsymbol{\gamma}_i - \bar{\boldsymbol{\gamma}})(\boldsymbol{\gamma}_i - \bar{\boldsymbol{\gamma}})'/N$. There is no closed form for the first conditional posterior. The Metropolis-Hastings sampling with Gamerman approach is the default sampling algorithm for it.

You can also use random walk Metropolis sampling when direct sampling is not an option, such as for the fixed-coefficients parameters $\boldsymbol{\beta}$ and random-coefficients parameters $\boldsymbol{\gamma}_i$. You can specify ALGORITHM=RWM to choose random walk Metropolis sampling. For the random-coefficients parameters $\boldsymbol{\gamma}_i$, Rossi, McCulloch, and Allenby (1996) and Rossi, Allenby, and McCulloch (2005) suggest random walk Metropolis sampling in which increments have covariance $s^2\boldsymbol{\Omega}_{\boldsymbol{\gamma}}^t$, where s is a scaling constant whose value is usually set at $\frac{2.93}{\sqrt{\dim(\boldsymbol{\gamma}_i)}}$ and $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}^t$ is the current draw of $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$.

The sampling for the random-effects-only setup is often faster, because the first conditional posterior for each i involves only the data for individual i and because the second and third conditional posteriors depend not on the data directly but only on the draws of $\boldsymbol{\gamma}_i$.

Probit with Random Effects

The probit model with random effects has the following parameters: the fixed-coefficients parameters $\boldsymbol{\beta}$, the covariance parameters for the error differences $\tilde{\boldsymbol{\Sigma}}$, the random-coefficients parameters $\boldsymbol{\gamma}_i$, and the covariance parameters for the random coefficients $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$. It has extra parameters $(\boldsymbol{\gamma}_i, \boldsymbol{\Omega}_{\boldsymbol{\gamma}})$ in addition to $(\boldsymbol{\beta}, \tilde{\boldsymbol{\Sigma}})$ in a fixed-effects-only model. You can conveniently adapt the Gibbs sampler proposed in McCulloch and Rossi (1994) to handle the model by appending the new parameters to the set of parameters that would be drawn for a probit model without random coefficients. The sampling can be carried out by the following conditional posteriors:

(1) $(w_{ij} | \mathbf{w}_{i,-j}, \boldsymbol{\beta}, \tilde{\boldsymbol{\Sigma}}, \boldsymbol{\gamma}_i, \mathbf{Y}) \quad i = 1, \ldots, N$ and $j = 1, \ldots, J-1$

(2) $(\boldsymbol{\beta} | \mathbf{W}, \tilde{\boldsymbol{\Sigma}}, \boldsymbol{\gamma}_i, \mathbf{Y})$

(3) $(\boldsymbol{\gamma}_i | \mathbf{W}, \boldsymbol{\beta}, \tilde{\boldsymbol{\Sigma}}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}, \mathbf{Y}) \quad i = 1, \ldots, N$

(4) $(\tilde{\boldsymbol{\Sigma}} | \mathbf{W}, \boldsymbol{\beta}, \mathbf{Y})$

(5) $(\boldsymbol{\Omega}_{\boldsymbol{\gamma}} | \mathbf{W}, \boldsymbol{\beta}, \tilde{\boldsymbol{\Sigma}}, \boldsymbol{\gamma}_i, \mathbf{Y})$

All the groups of conditional distributions have closed forms that are easily drawn from. Conditional (1) is truncated normal, (2) and (3) are normal, and (4) and (5) are inverse Wishart distribution. For more information, see McCulloch and Rossi (1994).

Other Types of Choice Response

So far, the focus has been on one type of choice response, where the response takes binary integer values: 1 for a chosen alternative and 0 for an unchosen alternative. Each choice set should have at least two alternatives and only one chosen alternative. This is the default choice-type.

However, there are other types of choice response that PROC BCHOICE supports.

MaxDiff

Louviere (1991) and Finn and Louviere (1992) developed a technique in which a person was shown a set or subset of the possible items and asked to indicate the least preferred item in a choice set, in addition to the traditional most preferred item. This approach is now called MaxDiff (maximum difference) or best-worst scaling (BWS).

A respondent can pick one best and one worst item in each choice set. The choice response variable takes one of the three integer values: 1 for the best alternative, –1 for the worst alternative, and 0 for the ones in the middle. MaxDiff questionnaires are relatively easy and straightforward. People are often much better at judging items at extremes than discriminating among items of medium importance or preference (Louviere 1991). MaxDiff is free from scale bias because the responses involve choosing items rather than expressing strength of preference. MaxDiff forces trade-offs of items and provides better discrimination among alternatives and between groups of the alternatives.

Following random utility theory, the MaxDiff experiments treat the choices of best alternatives as utility-maximizing behavior and the choices of worst alternatives as utility-minimizing behavior. Each choice set in a MaxDiff data set is coded as two separate sets: one for the best choice and one for the worst choice. In the first set, the only change is to replace the response value –1 with 0 for the worst choice. In the second set, all elements in the attribute design matrix are negated (multiplied by –1), the response value is changed from 1

to 0 for the best choice, and the response value is changed from –1 to 1 for the worst choice. The observed part of the utility function for the set of the best choice remains the same,

$$v_{ij} = \mathbf{x}'_{ij}\boldsymbol{\beta}$$

whereas the observed part of the utility function for the set of the worst choice has changed the sign,

$$v_{ij} = -\mathbf{x}'_{ij}\boldsymbol{\beta}$$

Then, you concatenate these two sets and estimate parameters by using a single model.

For MaxDiff data (where CHOICETYPE=MAXDIFF is specified), PROC BCHOICE expands each choice set to two separate sets: one for the best choice and one for the worst choice. You can do the data expansion by yourself and then treat the data as regular binary type of choice data without specifying CHOICETYPE=MAXDIFF. The former is recommended, although these two approaches are equivalent. If there are nonestimable parameters due to singularity when a classification covariate is specified, PROC BCHOICE is able to detect it; whereas if you expand the data as described above before calling PROC BCHOICE, those parameters (columns), which were singular before data expansion, are not singular any more (but they should be deleted).

Allocation

Binary choice data closely mimic actual purchase situations, but researchers sometimes want more information than just determining the best choice: for example, they might ask respondents to answer more fully by allocating a percentage to each alternative in a choice set. Alternatively, the respondents might be asked to think about the next 10 buying occasions and to state how many times they will pick each alternative in total. This type of choice experiment is called an allocation choice model.

In an allocation choice model, the choice takes a percentage of preference for each alternative, and the sum of percentages in each choice set is 100%. An allocation choice model can provide more information than choices alone, but it has its own disadvantages. One shortcoming is that it takes respondents longer to answer each question. Another is that respondents' mental processes of assigning percentages are not clear.

When the choice response is of the allocation type, you can specify the weight suboptions to indicate how many choice sets the allocated percentage should apply to. For example, if the respondent were asked to imagine the next 20 purchase occasions and to allocate a percentage of those 20 purchases for each alternative, you would specify WEIGHT=20. By default, WEIGHT=10.

Identification and Specification

Two aspects of identification in choice models play an important role in model specification: location shift and scale shift. For more information, see Chapter 2 in Train (2009).

Location Shift

Decision makers choose the alternative that gives them the greatest utility. If a constant is added to the utilities of all alternatives, the location of the utilities will change, but the decision maker will choose the same alternative.

The probability that the individual i chooses alternative j is

$$\begin{aligned} P(y_{ij} = 1) &= \Pr\left(u_{ij} > u_{ik} \text{ for all } k \neq j\right) \\ &= \Pr\left(u_{ij} - u_{ik} > 0 \text{ for all } k \neq j\right) \end{aligned}$$

Only the difference in utilities matters, not the absolute values (Train 2009). This explains why there is no overall intercept term in a choice model.

Alternative-Specific Main Effects

People often want to observe the main effect that is related to each of the alternatives. For example,

$$u_{ij} = \alpha_j + \mathbf{x}'_{ij}\boldsymbol{\beta} + \epsilon_{ij} \text{ for all } j$$

where α_j is the main effect that is related to alternative j, $\mathbf{x}_{ij}$ is the design variable vector that is specific to alternative j, and $\boldsymbol{\beta}$ are the corresponding coefficients of $\mathbf{x}_{ij}$. The main effect α_j implies the average effect for alternative j on utility of all factors that are not included in the model.

Consider a person who can take either a car or a bus to work. The form of utilities for taking a car or bus can be specified as follows:

$$\begin{aligned} u_{\text{car}} &= \alpha^1_{\text{car}} + \mathbf{x}'_{\text{car}}\boldsymbol{\beta} + \epsilon_{\text{car}} \\ u_{\text{bus}} &= \alpha^1_{\text{bus}} + \mathbf{x}'_{\text{bus}}\boldsymbol{\beta} + \epsilon_{\text{bus}} \end{aligned}$$

The preceding form is equivalent to the form

$$\begin{aligned} u_{\text{car}} &= \alpha^2_{\text{car}} + \mathbf{x}'_{\text{car}}\boldsymbol{\beta} + \epsilon_{\text{car}} \\ u_{\text{bus}} &= \alpha^2_{\text{bus}} + \mathbf{x}'_{\text{bus}}\boldsymbol{\beta} + \epsilon_{\text{bus}} \end{aligned}$$

as long as $\alpha^1_{\text{car}} - \alpha^1_{\text{bus}} = \alpha^2_{\text{car}} - \alpha^2_{\text{bus}}$. There are infinitely many values that will result in the same choice probabilities. The two main effects are not estimable.

To avoid this problem, you must normalize one of the main effects to 0. If you set α_{car} to 0, then α_{bus} is interpreted as the average effect on the utility of bus relative to car. PROC BCHOICE automatically normalizes for you by setting one of the main effects to 0 if you specify the alternative specific main effects in the CLASS statement. If you want to designate which alternative to set to 0, use the REF= option in the CLASS statement.

Individual-Specific Effects

Attributes of the alternatives vary, so they are also called alternative-specific effects. However, demographic variables, such as age, gender, and race, do not change among alternatives for an individual. These are individual-specific effects. They can enter the model only if they are specified in ways that create differences in utilities among alternatives.

Suppose you want to examine the effect of an individual's age on choosing car or bus:

$$\begin{aligned} u_{\text{car}} &= \mathbf{x}'_{\text{car}}\boldsymbol{\beta} + \theta_{\text{car}}\text{Age} + \epsilon_{\text{car}} \\ u_{\text{bus}} &= \alpha_{\text{bus}} + \mathbf{x}'_{\text{bus}}\boldsymbol{\beta} + \theta_{\text{bus}}\text{Age} + \epsilon_{\text{bus}} \end{aligned}$$

Because only differences in utility matter, you cannot estimate both θ_{car} and θ_{bus}. You need to set one of them to 0. You can do this by interacting Age with the alternative-specific main effects, one of which has been normalized to 0:

$$\begin{aligned} u_{\text{car}} &= \mathbf{x}'_{\text{car}}\boldsymbol{\beta} + \epsilon_{\text{car}} \\ u_{\text{bus}} &= \alpha_{\text{bus}} + \mathbf{x}'_{\text{bus}}\boldsymbol{\beta} + \theta_{\text{bus}}\text{Age} + \epsilon_{\text{bus}} \end{aligned}$$

where θ_{bus} is interpreted as the differential effect of age on the utility of bus versus car.

You need to make sure that individual-specific effects have interacted with some alternative-specific effects, so that there is no identification problem. PROC BCHOICE checks whether there is any variable that does not change among alternatives; it issues a warning message if any such variable is found.

Scale Shift

The scale of utility is also irrelevant for choice models. The alternative that has the largest utility is the largest, regardless of how the utilities are scaled. Multiplying the utilities by a constant does not change the choice probabilities.

Scale Normalization for Logit Models

Consider the utility function

$$u^1_{ij} = \mathbf{x}'_{ij}\boldsymbol{\beta} + \epsilon^1_{ij} \quad \text{for all} \quad j$$

where $\text{Var}(\epsilon^1_{ij}) = 1$. There is another utility function

$$u^2_{ij} = \mathbf{x}'_{ij}(\boldsymbol{\beta} * \sigma) + \epsilon^2_{ij} \quad \text{for all} \quad j$$

where $\text{Var}(\epsilon^2_{ij}) = \sigma^2$. These two utility functions are equivalent. The coefficients $(\boldsymbol{\beta} * \sigma)$ reflect the effect of the observed variables relative to the standard deviation of the unobserved factors (Train 2009).

In both standard and nested logit models, the error variance is equal to $\pi^2/6$, which is around 1.6. So the coefficients are larger by a factor of $\sqrt{1.6}$ than the coefficients of a model that has an error variance of 1.

Scale Normalization for Probit Models

Parameters in logit models are automatically normalized because of the specific variance assumption on the error term. In probit models, the covariance matrix of the error is estimated together with the $\boldsymbol{\beta}$ coefficients; therefore, the normalization is not done automatically. The researcher has to be careful in interpreting the results.

For notational convenience, use a four-alternative model. The error vector is $\boldsymbol{\epsilon}' = (\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4)$, which is assumed to have a normal distribution with 0 mean and a covariance matrix

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} & \sigma_{14} \\ . & \sigma_{22} & \sigma_{23} & \sigma_{24} \\ . & . & \sigma_{33} & \sigma_{34} \\ . & . & . & \sigma_{44} \end{pmatrix}$$

where the dots refer to the corresponding elements in the upper part of the symmetric matrix. There are J(J+1)/2 elements in the covariance matrix, but not all of them are estimable, because only the difference in the utilities matters.

A typical way of normalizing the covariance matrix is to reduce it by differencing the utilities with respect to the last alternative; some prefer the first alternative (Train 2009). You can choose which alternative to subtract by rearranging the data in the right order. Define the error differences as $(\tilde{\epsilon}_{14}, \tilde{\epsilon}_{24}, \tilde{\epsilon}_{34})$; the subscripts mean that the error difference is taken against the fourth one. The covariance matrix for the transformed new vector of error differences is of the form

$$\tilde{\mathbf{\Sigma}} = \begin{pmatrix} \tilde{\sigma}_{11} & \tilde{\sigma}_{12} & \tilde{\sigma}_{13} \\ . & \tilde{\sigma}_{22} & \tilde{\sigma}_{23} \\ . & . & \tilde{\sigma}_{33} \end{pmatrix}$$

where

$$\begin{aligned}
\tilde{\sigma}_{11} &= \sigma_{11} + \sigma_{44} - 2\sigma_{14} \\
\tilde{\sigma}_{22} &= \sigma_{22} + \sigma_{44} - 2\sigma_{24} \\
\tilde{\sigma}_{33} &= \sigma_{33} + \sigma_{44} - 2\sigma_{34} \\
\tilde{\sigma}_{12} &= \sigma_{12} + \sigma_{44} - \sigma_{14} - \sigma_{24} \\
\tilde{\sigma}_{13} &= \sigma_{13} + \sigma_{44} - \sigma_{14} - \sigma_{34} \\
\tilde{\sigma}_{23} &= \sigma_{23} + \sigma_{44} - \sigma_{24} - \sigma_{34}
\end{aligned}$$

Furthermore, the top left entry on the diagonal is set to 1:

$$\tilde{\tilde{\mathbf{\Sigma}}} = \begin{pmatrix} 1 & \tilde{\sigma}_{12}/\tilde{\sigma}_{11} & \tilde{\sigma}_{13}/\tilde{\sigma}_{11} \\ . & \tilde{\sigma}_{22}/\tilde{\sigma}_{11} & \tilde{\sigma}_{23}/\tilde{\sigma}_{11} \\ . & . & \tilde{\sigma}_{33}/\tilde{\sigma}_{11} \end{pmatrix}$$

PROC BCHOICE outputs estimates of $\tilde{\tilde{\mathbf{\Sigma}}}$.

In the multinomial logit (MNL) model, each ϵ_{ij} is independently and identically distributed (iid) with the Type I extreme-value distribution, and the covariance matrix is $\pi^2/6I$. The normalized covariance matrix of the vector of error differences would be

$$\begin{pmatrix} 1 & 0.5 & .5 \\ . & 1 & .5 \\ . & . & 1 \end{pmatrix}$$

This provides a method of model selection: you might want to fit a probit model first, check the normalized covariance matrix estimation to see how close it is to the preceding matrix, and then decide whether a simpler logit model would be appropriate.

Gamerman Algorithm

Discrete choice logit models fall in the framework of a generalized linear model (GLM) with a logit link. The Metropolis-Hastings sampling approach of Gamerman (1997) is well suited to this type of model.

In the GLM setting, the data are assumed to be independent with exponential family density

$$f(\mathbf{y}_i|\boldsymbol{\theta}_i) = \exp\left[(\mathbf{y}_i\boldsymbol{\theta}_i - b(\boldsymbol{\theta}_i))/\phi_i\right]c(\mathbf{y}_i, \phi_i)$$

The means that $\boldsymbol{\mu}_i = E(\mathbf{y}_i|\boldsymbol{\theta}_i)$ are related to the canonical parameters $\boldsymbol{\theta}_i$ via $\boldsymbol{\mu}_i = b'(\boldsymbol{\theta}_i)$ and to the regression coefficients via the link function

$$g(\boldsymbol{\mu}_i) = \eta_i = \mathbf{X}_i\boldsymbol{\beta}$$

The maximum likelihood (ML) estimator in a GLM and the asymptotic variance are obtained by iterative application of weighted least squares (IWLS) to transformed observations. Following McCullagh and Nelder (1989), define the transformed response as

$$\tilde{\mathbf{y}}_i(\boldsymbol{\beta}) = \eta_i + (\mathbf{y}_i - \boldsymbol{\mu}_i)g'(\boldsymbol{\mu}_i)$$

and define the corresponding weights as

$$\mathbf{W}_i^{-1}(\boldsymbol{\beta}) = b''(\boldsymbol{\theta}_i)[g'(\boldsymbol{\mu}_i)]^2$$

Suppose a normal prior is specified on $\boldsymbol{\beta}$, $\pi(\boldsymbol{\beta}) = \mathrm{N}(\bar{\boldsymbol{\beta}}, \boldsymbol{\Omega}_{\boldsymbol{\beta}})$. The posterior density is as follows:

$$\begin{aligned} p(\boldsymbol{\beta}|\mathbf{Y}) &\propto p(\mathbf{Y}|\boldsymbol{\beta})\pi(\boldsymbol{\beta}) \\ &\propto \exp\{\sum_{i=1}^{N}\frac{\mathbf{y}_i\boldsymbol{\theta}_i - b(\boldsymbol{\theta}_i)}{\phi_i} - \frac{1}{2}(\boldsymbol{\beta}-\bar{\boldsymbol{\beta}})'\boldsymbol{\Omega}_{\boldsymbol{\beta}}^{-1}(\boldsymbol{\beta}-\bar{\boldsymbol{\beta}})\} \end{aligned}$$

Gamerman (1997) proposes that Metropolis-Hastings sampling be combined with iterative weighted least squares as follows:

1. Start with $\boldsymbol{\beta}^{(0)}$ and $t = 1$.
2. Sample $\boldsymbol{\beta}^*$ from the proposal density $\mathbf{N}(\mathbf{m}^{(t)}, \mathbf{C}^{(t)})$, where

$$\begin{aligned} \mathbf{m}^{(t)} &= \{\boldsymbol{\Omega}_{\boldsymbol{\beta}}^{-1} + \mathbf{X}'\mathbf{W}(\boldsymbol{\beta}^{(t-1)})\mathbf{X}\}^{-1}\{\boldsymbol{\Omega}_{\boldsymbol{\beta}}^{-1}\bar{\boldsymbol{\beta}} + \mathbf{X}'\mathbf{W}(\boldsymbol{\beta}^{(t-1)})\tilde{\mathbf{Y}}(\boldsymbol{\beta}^{(t-1)})\} \\ \mathbf{C}^{(t)} &= \{\boldsymbol{\Omega}_{\boldsymbol{\beta}}^{-1} + \mathbf{X}'\mathbf{W}(\boldsymbol{\beta}^{(t-1)})\mathbf{X}\}^{-1} \end{aligned}$$

3. Accept $\boldsymbol{\beta}^*$ with probability

$$\alpha(\boldsymbol{\beta}_{(t-1)}, \boldsymbol{\beta}^*) = \min[1, \frac{p(\boldsymbol{\beta}^*|\mathbf{Y})q(\boldsymbol{\beta}^*, \boldsymbol{\beta}^{(t-1)})}{p(\boldsymbol{\beta}^{(t-1)}|\mathbf{Y})q(\boldsymbol{\beta}^{(t-1)}, \boldsymbol{\beta}^*)}]$$

   where $p(\boldsymbol{\beta}|\mathbf{Y})$ is the posterior density and $q(\boldsymbol{\beta}^*, \boldsymbol{\beta}^{(t-1)})$ and $q(\boldsymbol{\beta}^{(t-1)}, \boldsymbol{\beta}^*)$ are the transitional probabilities that are based on the proposal density $\mathbf{N}(\mathbf{m}^{(.)}, \mathbf{C}^{(.)})$. More specifically, $q(\boldsymbol{\beta}^*, \boldsymbol{\beta}^{(t-1)})$ is an $\mathbf{N}(\mathbf{m}^*, \mathbf{C}^*)$ density that is evaluated at $\boldsymbol{\beta}^{(t-1)}$, whereas $\mathbf{m}^*$ and $\mathbf{C}^*$ have the same expression as $\mathbf{m}^{(t)}$ and $\mathbf{C}^{(t)}$ but depend on $\boldsymbol{\beta}^*$ instead of $\boldsymbol{\beta}^{(t-1)}$. If $\boldsymbol{\beta}^*$ is not accepted, the chain stays with $\boldsymbol{\beta}^{(t-1)}$.

4. Set $t = t + 1$ and return to step 1.

You can extend this methodology to logit models that have random effects. If there are random effects, the link function is extended to

$$g(\boldsymbol{\mu}_i) = \eta_i = \mathbf{X}_i\boldsymbol{\beta} + \mathbf{Z}_i\boldsymbol{\gamma}_i$$

where the random effects are assumed to have a normal distribution, $\pi(\boldsymbol{\gamma}_i) = \mathrm{N}(\mathbf{0}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}})$, and $\pi(\boldsymbol{\Omega}_{\boldsymbol{\gamma}}) =$ inverse Wishart$(\nu_0, \mathbf{V}_0)$. The posterior density is

$$p(\boldsymbol{\beta}, \boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_N, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}|\mathbf{Y}) \propto p(\mathbf{Y}|\boldsymbol{\beta}, \boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_N, \boldsymbol{\Omega}_{\boldsymbol{\gamma}})\pi(\boldsymbol{\beta}) \prod_{i=1}^{N} \pi(\boldsymbol{\gamma}_i)\pi(\boldsymbol{\Omega}_{\boldsymbol{\gamma}})$$

The parameters are divided into blocks, $\boldsymbol{\beta}, \boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_N$, and $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$. For the fixed-effects $\boldsymbol{\beta}$ block, the conditional posterior has the same form, but the link changes to include $\mathbf{Z}_i\boldsymbol{\gamma}_i, i = 1, \ldots, N$, which are taken as known constants (offsets) at each iteration. The only change that is needed is to replace the transformed response $\tilde{\mathbf{y}}_i(\boldsymbol{\beta}^{(t-1)})$ with $\tilde{\mathbf{y}}_i(\boldsymbol{\beta}^{(t-1)}) - \mathbf{Z}_i\boldsymbol{\gamma}_i$ in step 2 of the previous Gamerman procedure.

For the random-effects $\boldsymbol{\gamma}_i$ block, the same Metropolis-Hastings sampling with the least square proposal can apply. The conditional posterior is

$$p(\boldsymbol{\gamma}_i|\mathbf{Y}, \boldsymbol{\beta}, \boldsymbol{\Omega}_{\boldsymbol{\gamma}}) \propto \exp\{\frac{\mathbf{y}_i\boldsymbol{\theta}_i - b(\boldsymbol{\theta}_i)}{\phi_i} - \frac{1}{2}\boldsymbol{\gamma}_i'\boldsymbol{\Omega}_{\boldsymbol{\gamma}}^{-1}\boldsymbol{\gamma}_i\}$$

The transformed response is now $\tilde{\mathbf{y}}_i(\boldsymbol{\gamma}_i^{(t-1)})$, and the proposal density is $\mathbf{N}(\mathbf{m}_i^{(t)}, \mathbf{C}_i^{(t)})$, where

$$\begin{aligned}
\mathbf{m}_i^{(t)} &= \{\boldsymbol{\Omega}_{\boldsymbol{\gamma}}^{-1} + \mathbf{Z}'\mathbf{W}(\boldsymbol{\gamma}_i^{(t-1)})\mathbf{Z}\}^{-1}\mathbf{Z}'\mathbf{W}(\boldsymbol{\gamma}_i^{(t-1)})\{\tilde{\mathbf{Y}}(\boldsymbol{\gamma}_i^{(t-1)}) - \mathbf{X}_i\boldsymbol{\beta}\} \\
\mathbf{C}^{(t)} &= \{\boldsymbol{\Omega}_{\boldsymbol{\gamma}}^{-1} + \mathbf{Z}'\mathbf{W}(\boldsymbol{\gamma}_i^{(t-1)})\mathbf{X}_i\}^{-1}
\end{aligned}$$

Finally, for the covariance matrix $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$ block, direct sampling from an inverse Wishart$(\nu_0 + N, \mathbf{V}_0 + S)$ is used, where $S = \sum_{i=1}^{N} \boldsymbol{\gamma}_i\boldsymbol{\gamma}_i'/N$.

The chain is initialized with random effects set to 0 and the covariance set to the identity matrix. Updating is done first for the fixed effects, $\boldsymbol{\beta}$, as a block to position the chain in the correct region of the parameter space. Then the random effects are updated, and finally the covariance of the random effects is updated. For more information about this algorithm, see Gamerman (1997).

Tuning the Random Walk Metropolis in Logit Models

When you use a random walk Metropolis for a logit model, you want to find a good proposal distribution for each sampling block (for a fixed-effects block or for each individual random-effects block). This process is called tuning. The tuning phase consists of a number of loops. The minimum number of loops is controlled by the MINTUNE= option, which has a default value of 2. The MAXTUNE= option controls the maximum number of tuning loops, and it has a default value of 6. The number of iterations in each loop is controlled by the NTU= option, which has a default value of 500. At the end of every loop, PROC BCHOICE examines the acceptance probability for each block. The acceptance probability is the percentage of proposals (one from each iteration) that have been accepted. If the probability falls within the acceptance tolerance range (see the section "Scale Tuning" on page 1231), the current configuration of the scale and covariance matrix is kept. Otherwise, the scale and covariance matrix are modified before the next tuning loop.

A good proposal distribution should resemble the actual posterior distribution of the parameters. Large-sample theory states that the posterior distribution of the parameters approaches a multivariate normal distribution (Gelman et al. 2004, Appendix B; Schervish 1995, Section 7.4). That is why a normal proposal distribution often works well in practice. The default proposal distribution in PROC BCHOICE is the normal distribution:

$$q_j(\theta_{\text{new}}|\theta^t) = \text{MVN}(\theta_{\text{new}}|\theta^t, c^2\boldsymbol{\Sigma})$$

Scale Tuning

The acceptance rate is closely related to the sampling efficiency of a Metropolis chain. For a random walk Metropolis, a high acceptance rate means that most new samples occur very close to the current data point. Their frequent acceptance means that the Markov chain is moving rather slowly and not exploring the parameter space fully. On the other hand, a low acceptance rate means that the proposed samples are often rejected; hence the chain is not moving much. An efficient Metropolis sampler has an acceptance rate that is neither too high nor too low. The scale c in the proposal distribution $q(\cdot|\cdot)$ effectively controls this acceptance probability. Roberts, Gelman, and Gilks (1997) showed that if both the target and proposal densities are normal, the optimal acceptance probability for the Markov chain should be around 0.45 in a single-dimensional problem, and asymptotically approaches 0.234 in higher dimensions.

The nature of stochastic simulations makes it impossible to fine-tune a set of variables such that the Metropolis chain has the exact desired acceptance rate. In addition, Roberts and Rosenthal (2001) empirically demonstrate that an acceptance rate between 0.15 and 0.5 is at least 80% efficient, so there is really no need to fine-tune the algorithms to reach an acceptance probability that is within small tolerance of the optimal values. PROC BCHOICE works with a probability range, which is $a \pm b$, where a and b are the values of the TARGACCEPT and ACCEPTTOL options, respectively, in the BCHOICE statement. The default value of TARGACCEPT is a function of the number of parameters in the model, as outlined in Roberts, Gelman, and Gilks (1997). By default, ACCEPTTOL=0.075. If the observed acceptance rate in a particular tuning loop is less than the lower bound of the range, the scale is reduced; if the observed acceptance rate is greater than the upper bound of the range, the scale is increased. During the tuning phase, a scale parameter in the normal distribution is adjusted as a function of the observed acceptance rate and the target acceptance rate. PROC BCHOICE uses the updating scheme

$$c_{\text{new}} = \frac{c_{\text{cur}} \cdot \Phi^{-1}(p_{\text{opt}}/2)}{\Phi^{-1}(p_{\text{cur}}/2)}$$

where c_{cur} is the current scale, p_{cur} is the current acceptance rate, and p_{opt} is the optimal acceptance probability.[1]

Covariance Tuning

To tune a covariance matrix, PROC BCHOICE takes a weighted average of the old proposal covariance matrix and the recently observed covariance matrix, based on a number of samples in the current loop that is specified in NTU= option. The TUNEWT=*w* option determines how much weight is put on the recently observed covariance matrix. The following formula is used to update the covariance matrix:

$$\text{COV}_{\text{new}} = w\,\text{COV}_{\text{cur}} + (1 - w)\text{COV}_{\text{old}}$$

For fixed effects:

- By default, PROC BCHOICE uses the mode of the posterior density as the initial value of the fixed parameter and uses a numerical optimization routine (such as the quasi-Newton method) to find a starting covariance matrix. The optimization is performed on the joint posterior distribution, and the covariance matrix is a quadratic approximation at the posterior mode. In some cases, this is a better and more efficient way of initializing the covariance matrix. However, there are cases (such as when the number of parameters is large) where the optimization could fail to find a matrix that is positive definite. In that case, the tuning covariance matrix is reset to the identity matrix.
- If you specify INIT=PRIORMODE or INIT=(LIST=*numeric-list*) to assign some initial values for the fixed effects in the MODEL statement, no numerical optimization is carried out and the proposal covariance matrix is set to the identity matrix divided by the number of parameters in the fixed effects sampling block. It can take a number of tuning phases before the proposal distribution is tuned to its optimal stage, because the Markov chain needs to spend time learning about the posterior covariance structure. If the posterior variances of your parameters vary by more than a few orders of magnitude, if the variances of your parameters are much different from 1, or if the posterior correlations are high, then the proposal tuning algorithm might have difficulty with forming an acceptable proposal distribution.

For random effects, the initial values are set to 0 and the proposal covariance matrix is set to the identity matrix divided by the number of parameters in each individual-level random-effects sampling block.

[1] Roberts, Gelman, and Gilks (1997) and Roberts and Rosenthal (2001) demonstrate that the relationship between acceptance probability and scale in a random walk Metropolis is $p = 2\Phi\left(-\sqrt{I}c/2\right)$, where c is the scale, p is the acceptance rate, Φ is the CDF of a standard normal, and $I \equiv E_f[(f'(x)/f(x))^2]$, $f(x)$ is the density function of samples. This relationship determines the updating scheme, with I being replaced by the identity matrix to simplify calculation.

Autocall Macros for Postprocessing

Although PROC BCHOICE provides a number of convergence diagnostic tests and posterior summary statistics, it performs the calculations only for the default tests and statistics or only if you specify the options in advance. If you want to analyze the posterior draws of unmonitored parameters or functions of the parameters that are calculated in later DATA step calls, you can use the autocall macros that are listed in Table 30.6.

Table 30.6 Postprocessing Autocall Macros

Macro	Description
%ESS	Effective sample sizes
%GEWEKE*	Geweke diagnostic
%HEIDEL*	Heidelberger-Welch diagnostic
%MCSE	Monte Carlo standard errors
%POSTACF	Autocorrelation
%POSTCOR	Correlation matrix
%POSTCOV	Covariance matrix
%POSTINT	Equal-tail and HPD intervals
%POSTSUM	Mean, standard deviation, and various quantiles
%RAFTERY	Raftery diagnostic
%SUMINT	Mean, standard deviation, and HPD interval
%TADPLOT	Trace plot, autocorrelation plot, and density plot

*The %GEWEKE and %HEIDEL macros use a different optimization routine than that used in PROC BCHOICE. As a result, there might be numerical differences in some cases, especially when the sample size is small.

Table 30.7 lists options that are shared by all postprocessing autocall macros. For macro-specific options, see Table 30.8.

Table 30.7 Shared Options

Option	Description
DATA=*SAS-data-set*	Names the input data set that contains posterior samples
OUT=*SAS-data-set*	Specifies a name for the output SAS data set to contain the results
PRINT=YES | NO	Displays the results (the default is YES)
VAR=*variable-list*	Specifies the variables on which to perform the calculation

Suppose the data set that contains posterior samples is called Post and the variables of interest are defined in the macro variable &PARMS. The following statements call the %ESS macro and calculate the effective sample sizes for each variable:

```
%ESS(data=Post, var=Alpha Beta U_1-U_17)
```

By default, the ESS estimates are displayed. You can choose not to display the result and instead use the following statement to save the output to a data set:

```
%ESS(data=Post, var=&parms, print=NO, out=eout)
```

Some of the macros can take additional options, which are listed in Table 30.8.

Table 30.8 Macro-Specific Options

Macro	Option	Description
%ESS	AUTOCORLAG=*numeric*	Specifies the maximum number of autocorrelation lags used in computing the ESS estimates. By default, AUTOCORLAG=MIN(500, NOBS/4), where NOBS is the sample size of the input data set.
	HIST=YES | NO	Displays a histogram of all ESS estimates. By default, HIST=NO.
%GEWEKE	FRAC1=*numeric*	Specifies the earlier portion of the Markov chain used in the test. By default, FRAC1=0.1.
	FRAC2=*numeric*	Specifies the latter portion of the Markov chain used in the test. By default, FRAC2=0.5.
%HEIDEL	SALPHA=*numeric*	Specifies the α level for the stationarity test. By default, SALPHA=0.05.
	HALPHA=*numeric*	Specifies the α level for the halfwidth test. By default, HALPHA=0.05.
	EPS=*numeric*	Specifies a small positive number ϵ such that if the halfwidth is less than ϵ times the sample mean of the remaining iterations, the halfwidth test is passed. By default, EPS=0.1.
%MCSE	AUTOCORLAG=*numeric*	Specifies the maximum number of autocorrelation lags used in computing the Monte Carlo standard error estimates. By default, AUTOCORLAG=MIN(500, NOBS/4), where NOBS is the sample size of the input data set.
%POSTACF	LAGS=*%str(numeric-list)*	Specifies which autocorrelation lags to calculate. The default values are 1, 5, 10, and 50.
%POSTINT	ALPHA=*value*	Specifies the α level ($0 < \alpha < 1$) for the interval estimates. By default, ALPHA=0.05.
%RAFTERY	Q=*numeric*	Specifies the order of the quantile of interest. By default, Q=0.025.
	R=*numeric*	Specifies the margin of error for measuring the accuracy of estimation of the quantile. By default, R=0.005.
	S=*numeric*	Specifies the probability of attaining the accuracy of the estimation of the quantile. By default, S=0.95.
	EPS=*numeric*	Specifies the tolerance level for the stationary test. By default, EPS=0.001.

For example, the following statement calculates and displays autocorrelation at lags 1, 6, 11, 50, and 100. Note that the lags in the *numeric-list* must be separated by commas.

```
%POSTACF(data=Post, var=&parms, lags=%str(1 to 15 by 5, 50, 100))
```

Regenerating Diagnostics Plots

By default, PROC BCHOICE generates three plots: the trace plot, the autocorrelation plot, and the kernel density plot. Unless ODS Graphics is enabled before the procedure is called, it is hard to generate the same graph afterward. Directly using the **Stat.BCHOICE.Graphics.TraceAutocorrDensity** template is not feasible. The easiest way to regenerate the same graph is to use the %TADPLOT autocall macro. The %TADPLOT macro requires you to specify an input data set (which is the output data set from a previous PROC BCHOICE call) and a list of variables that you want to plot.

Suppose that the output data set Postsamp contains posterior draws for the regression coefficients of Mode1, Mode2, and Mode3. If you want to examine these parameters graphically, you can use the following statements to regenerate the graphs:

```
%TADPLOT(data=Postsamp, var=Mode1 Mode2 Mode3)
```

Displayed Output

This section describes the displayed output from PROC BCHOICE. For a quick reference of all ODS table names, see the section "ODS Table Names" on page 1239. ODS tables are arranged in four groups, listed in the following sections: "Model- and Data-Related ODS Tables" on page 1235, "Sampling-Related ODS Tables" on page 1236, "ODS Tables Related to Posterior Statistics" on page 1237, and "ODS Tables Related to Convergence Diagnostics" on page 1238.

Model- and Data-Related ODS Tables

Model Information

The "Model Information" table (ODS table name ModelInfo) displays basic information about the model, such as the name of the input data set, response variable, model type, sampling algorithm, burn-in size, simulation size, thinning, and random number seed. The "Model Information" table is produced irrespective of the sampling technique.

The "Model Information" table also displays the random number seed that is used to initialize the random number generators. If you repeat the analysis and specify this seed value in the SEED= option in the PROC BCHOICE statement, you get an identical stream of random numbers. This table is displayed by default.

Class Level Information

The "Class Level Information" table (ODS table name ClassLevels) lists the levels of every variable that is specified in both the CLASS statement and the MODEL statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress the "Class Level Information" table completely or partially by specifying the NOCLPRINT option in the PROC BCHOICE statement.

Choice Sets Summary

The "Choice Sets Summary" table (ODS table name ChoiceSummary) is displayed by default and should be used to check the data entry. It shows the numbers of choice sets, the number of alternatives, and the number of chosen and unchosen alternatives. You should always check this summary table to ensure that the data are arrayed correctly. This table is displayed by default.

Number of Observations

The "NObs" table (ODS table name NOBS) shows the number of observations in the data set and the number of observations that are used in the analysis. By default, observations that have missing values are not used. This table is displayed by default.

Sampling-Related ODS Tables

Burn-In History

The "Burn-In History" table (ODS table name BurnInHistory) shows the scales and acceptance rates for each parameter block in the burn-in phase for logit models. The table is not displayed by default, but you can request it by specifying MCHISTORY=BRIEF or MCHISTORY=DETAILED in the PROC BCHOICE statement.

Parameters Initial Value

The "Parameters Initial" table (ODS table name ParametersInit) shows the initial value of each fixed regression coefficient. This table is not displayed by default, but you can request it by specifying the option INIT=PINIT in the MODEL statement.

Prior for Fixed Effects

The "Prior for Fixed Effects" table (ODS table name CoeffPrior) shows the prior distribution for the regression coefficients. The table is displayed by default for a model with fixed effects.

Sampling History

The "Sampling History" table (ODS table name SamplingHistory) shows the scales and acceptance rates for each parameter block in the main sampling phase for logit models. The table is not displayed by default, but you can request it by specifying MCHISTORY=BRIEF or MCHISTORY=DETAILED in the PROC BCHOICE statement.

Tuning History

The "Tuning History" table (ODS table name TuningHistory) shows the number of tuning phases used in establishing the proposal distribution. The table also displays the scales and acceptance rates for each parameter block at each of the tuning phases for logit models using random walk Metropolis algorithm. For more information about the self-adapting proposal tuning algorithm, see the section "Tuning the Random Walk Metropolis in Logit Models" on page 1231. The table is not displayed by default, but you can request it by specifying MCHISTORY=BRIEF or MCHISTORY=DETAILED in the PROC BCHOICE statement.

ODS Tables Related to Posterior Statistics

PROC BCHOICE calculates some essential posterior statistics and outputs them to a number of ODS tables. Some of the ODS tables are produced by default, and you can request others by specifying an option in the PROC BCHOICE statement. For more information about the calculations, see the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Summary and Interval Statistics

The "Posterior Summaries and Intervals" table (ODS table name PostSumInt) contains basic summary and interval statistics for each parameter. The table lists the number of posterior samples, the posterior mean and standard deviation estimates, and the highest posterior density (HPD) interval estimates. This table is displayed by default.

Summary Statistics

The "Posterior Summaries" table (ODS table name PostSummaries) contains basic statistics for each parameter. The table lists the number of posterior samples, the posterior mean and standard deviation estimates, and the percentile estimates. This table is not displayed by default, but you can request it by specifying the option STATISTICS=SUM.

Correlation Matrix

The "Posterior Correlation Matrix" table (ODS table name Corr) contains the posterior correlation of model parameters. This table is not displayed by default, but you can request it by specifying the option STATISTICS=CORR.

Covariance Matrix

The "Posterior Covariance Matrix" table (ODS table name Cov) contains the posterior covariance of model parameters. This table is not displayed by default, but you can request it by specifying the option STATISTICS=COV.

Deviance Information Criterion

The "Deviance Information Criterion" table (ODS table name DIC) contains the deviance information criterion (DIC) of the model. This table is not displayed by default, but you can request it by specifying the option DIC. For more information about the calculations, see the section "Deviance Information Criterion (DIC)" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Interval Statistics

The "Posterior Intervals" table (ODS table name PostIntervals) contains the equal-tail and highest posterior density (HPD) interval estimates for each parameter. The default α value is 0.05, and you can change it to other levels by using the STATISTICS= option. This table is not displayed by default, but you can request it by specifying the option STATISTICS=INT.

ODS Tables Related to Convergence Diagnostics

PROC BCHOICE provides convergence diagnostic tests that check for Markov chain convergence. The procedure produces a number of ODS tables that you can request and save individually. For information about calculation, see the section "Statistical Diagnostic Tests" on page 166 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Autocorrelation

The "Autocorrelations" table (ODS table name AUTOCORR) contains the first-order autocorrelations of the posterior samples for each parameter. The Parameter column states the name of the parameter. By default, PROC BCHOICE displays lag 1, 5, 10, and 50 estimates of the autocorrelations. You can request different autocorrelations by using the DIAGNOSTICS=AUTOCORR(LAGS=) option. This table is displayed by default.

Effective Sample Size

The "Effective Sample Sizes" table (ODS table name ESS) calculates the effective sample size of each parameter. For more information, see the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is displayed by default.

Monte Carlo Standard Errors

The "Monte Carlo Standard Errors" table (ODS table name MCSE) calculates the standard errors of the posterior mean estimate. For more information, see the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is not displayed by default, but you can request it by specifying the option DIAGNOSTICS=MCSE.

Geweke Diagnostics

The "Geweke Diagnostics" table (ODS table name Geweke) lists the results of the Geweke diagnostic test. For more information, see the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is not displayed by default, but you can request it by specifying DIAGNOSTICS=GEWEKE.

Heidelberger-Welch Diagnostics

The "Heidelberger-Welch Diagnostics" table (ODS table name Heidelberger) lists the results of the Heidelberger-Welch diagnostic test. The test consists of two parts: a stationary test and a half-width test. For more information, see the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is not displayed by default, but you can request it by specifying DIAGNOSTICS=HEIDEL.

Raftery-Lewis Diagnostics

The "Raftery-Lewis Diagnostics" table (ODS table name Raftery) lists the results of the Raftery-Lewis diagnostic test. For more information, see the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is not displayed by default, but you can request it by specifying DIAGNOSTICS=RAFTERY.

ODS Table Names

PROC BCHOICE assigns a name to each table that it creates. You can use these names to refer to the tables when you use the output delivery system (ODS) to select tables and create output data sets. These names are listed in Table 30.9. For more information about ODS, see Chapter 24, "Statistical Graphics Using ODS."

Table 30.9 ODS Tables Produced in PROC BCHOICE

ODS Table Name	Description	Statement or Option
AutoCorr	Autocorrelation statistics for each parameter	DIAG=AUTOCORR
BurnInHistory	History of burn-in phase sampling	MCHISTORY=BRIEF | DETAILED
ChoiceSummary	Choice sets summary	Default
ClassLevels	Class level information	Default
CoeffPrior	Prior information for fixed effects	Default
Corr	Correlation matrix of the posterior samples	STATS=CORR
Cov	Covariance matrix of the posterior samples	STATS=COV
DIC	Deviance information criterion	DIC
ESS	Effective sample size for each parameter	Default
Geweke	Geweke diagnostics for each parameter	DIAG=GEWEKE
Heidelberger	Heidelberger-Welch diagnostics for each parameter	DIAG=HEIDEL
MCSE	Monte Carlo standard error for each parameter	STATS=MCSE
ModelInfo	Model information	Default
NObs	Number of observations	Default
ParametersInit	Parameter initial values	INIT=PINIT
PostSumInt	Posterior statistics for each parameter, including sample size, mean, standard deviation, and HPD intervals	Default
PostIntervals	Equal-tail and HPD intervals for each parameter	STATS=INT
PostSummaries	Posterior statistics for each parameter, including sample size, mean, standard deviation, and percentiles	STATS=SUM
Raftery	Raftery-Lewis diagnostics for each parameter	DIAG=RAFTERY
SamplingHistory	History of main phase sampling	MCHISTORY=BRIEF | DETAILED
TuningHistory	History of proposal distribution tuning	MCHISTORY=BRIEF | DETAILED

ODS Graphics

You can refer by name to every graph that is produced through ODS Graphics. The names of the graphs that PROC BCHOICE generates are listed in Table 30.10.

Table 30.10 Graphs Produced by PROC BCHOICE

ODS Graph Name	Plot Description	Statement and Option
ADPanel	Autocorrelation function and density panel	PLOTS=(AUTOCORR DENSITY)
AutocorrPanel	Autocorrelation function panel	PLOTS=AUTOCORR
AutocorrPlot	Autocorrelation function plot	PLOTS(UNPACK)=AUTOCORR
DensityPanel	Density panel	PLOTS=DENSITY
DensityPlot	Density plot	PLOTS(UNPACK)=DENSITY
TAPanel	Trace and autocorrelation function panel	PLOTS=(TRACE AUTOCORR)
TADPanel	Trace, density, and autocorrelation function panel	PLOTS=(TRACE AUTOCORR DENSITY)
TDPanel	Trace and density panel	PLOTS=(TRACE DENSITY)
TracePanel	Trace panel	PLOTS=TRACE
TracePlot	Trace plot	PLOTS(UNPACK)=TRACE

Examples: BCHOICE Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 30.1: Alternative-Specific and Individual-Specific Effects

(View the complete code for this example (bchcex1.sas) in the example repository.)

In many situations, a choice model that includes characteristics of both the alternatives and the individuals is needed for investigating consumer choice.

Consider an example of travel demand. People are asked to choose among travel by auto, plane, or public transit (bus or train). The following SAS statements create the data set Travel. The variables AutoTime, PlanTime, and TranTime represent the total travel time that is required to get to a destination by using auto, plane, or public transit, respectively. The variable Age represents the age of each individual who is surveyed, and the variable Chosen contains each individual's choice of travel mode.

```
data Travel;
   input AutoTime PlanTime TranTime Age Chosen $;
   AgeCtr=Age-34;
   datalines;
10.0 4.5 10.5 32 Plane
5.5 4.0 7.5 13 Auto
4.5 6.0 5.5 41 Transit
3.5 2.0 5.0 41 Transit
1.5 4.5 4.0 47 Auto
10.5 3.0 10.5 24 Plane
7.0 3.0 9.0 27 Auto
9.0 3.5 9.0 21 Plane
4.0 5.0 5.5 23 Auto
22.0 4.5 22.5 30 Plane
7.5 5.5 10.0 58 Plane
11.5 3.5 11.5 36 Transit
3.5 4.5 4.5 43 Auto
12.0 3.0 11.0 33 Plane
18.0 5.5 20.0 30 Plane
23.0 5.5 21.5 28 Plane
4.0 3.0 4.5 44 Plane
5.0 2.5 7.0 37 Transit
3.5 2.0 7.0 45 Auto
12.5 3.5 15.5 35 Plane
1.5 4.0 2.0 22 Auto
;
```

In this example, the AutoTime, PlanTime, and TranTime variables apply to the alternatives, whereas Age is a characteristic of the individuals. AgeCtr, a centered version of Age, is created by subtracting the sample's mean age from each individual's age. To study how the choice depends on both the travel time and age of the individuals, you need to incorporate both types of variables.

Before you invoke PROC BCHOICE to fit a choice logit model, you must arrange your data in such a way that there is one observation for each combination of individual and alternative. In this example, let Subject identify the individuals, let TravTime represent the travel time for each mode of transportation, and let Choice have the value 1 if the alternative is chosen and 0 otherwise. The following SAS statements rearrange the data set Travel into a new data set, Travel2, and display the first nine observations:

```
data Travel2(keep=Subject Mode TravTime Age AgeCtr Choice);
   array Times[3] AutoTime PlanTime TranTime;
   array Allmodes[3] $ _temporary_ ('Auto' 'Plane' 'Transit');
   set Travel;
   Subject = _n_;
   do i = 1 to 3;
      Mode = Allmodes[i];
      TravTime = Times[i];
      Choice = (Chosen eq Mode);
      output;
   end;
run;
```

```
proc print data=Travel2 (obs=20);
   by Subject;
   id Subject;
run;
```

The data for the first nine observations is shown in Output 30.1.1.

Output 30.1.1 Data for the First Nine Observations

Subject	Age	AgeCtr	Mode	TravTime	Choice
1	32	-2	Auto	10.0	0
	32	-2	Plane	4.5	1
	32	-2	Transit	10.5	0

Subject	Age	AgeCtr	Mode	TravTime	Choice
2	13	-21	Auto	5.5	1
	13	-21	Plane	4.0	0
	13	-21	Transit	7.5	0

Subject	Age	AgeCtr	Mode	TravTime	Choice
3	41	7	Auto	4.5	0
	41	7	Plane	6.0	0
	41	7	Transit	5.5	1

Notice that each subject in the data set Travel corresponds to a block of three observations in the data set Travel2, one for each travel alternative. The response variable Choice indicates the chosen alternative by the value 1 and the unchosen alternative by the value 0; exactly one alternative is chosen. The following SAS statements invoke PROC BCHOICE to fit the choice logit model:

```
proc bchoice data=Travel2 seed=124;
   class Mode Subject / param=ref order=data;
   model Choice = Mode TravTime / choiceset=(Subject);
run;
```

The "Choice Sets Summary" table shows that there are 21 choice sets and that each consists of three alternatives and one chosen alternative (each subject chooses one out of the three travel modes). It seems that the data are arrayed correctly.

Output 30.1.2 Choice Sets Summary

The BCHOICE Procedure

Choice Sets Summary				
Pattern	Choice Sets	Total Alternatives	Chosen Alternatives	Not Chosen
1	21	3	1	2

Summary statistics are shown in Output 30.1.3.

Output 30.1.3 PROC BCHOICE Posterior Summary Statistics

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Mode Auto	5000	-0.1678	0.7440	-1.7017	1.2396
Mode Plane	5000	-1.8794	1.2683	-4.6055	0.3801
TravTime	5000	-0.5695	0.2047	-0.9943	-0.2328

When Transit is the reference mode (normalized to 0), the part-worth (posterior mean) of Auto, which is negative, might reflect that driving is more inconvenient than traveling by bus or train, and the negative part-worth of Plane might reflect that traveling by plane is more expensive than traveling by bus or train. However, both are only suggestive, because the 95% HPD intervals have 0 in them. The posterior mean of TravTime is negative, which makes sense because having to spend more time en route is often unfavorable.

To study the relationship between the choice of transportation and the age of people who make the choice, you need to create an interaction between AgeCtr and Mode. AgeCtr is not estimable by itself, because it is the same throughout a choice set for an individual. The following statements fit a model including the interaction between AgeCtr and Mode:

```
proc bchoice data=Travel2 seed=124;
   class Mode Subject / param=ref order=data;
   model Choice = Mode Mode*AgeCtr TravTime / choiceset=(Subject);
run;
```

Output 30.1.4 PROC BCHOICE Posterior Summary Statistics

The BCHOICE Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Mode Auto	5000	-0.2634	0.7883	-1.8072	1.1395
Mode Plane	5000	-2.8210	1.5370	-5.9228	-0.0686
AgeCtr*Mode Auto	5000	-0.0986	0.0678	-0.2182	0.0350
AgeCtr*Mode Plane	5000	0.0251	0.0775	-0.1268	0.1618
TravTime	5000	-0.7608	0.2564	-1.2943	-0.3473

The parameter estimate for Mode Auto reflects the part-worth of Auto for an individual of mean age (34 years old), whereas the parameter estimate for Mode Plane is the part-worth of Plane for an individual of mean age. There are two interaction effects: the first corresponds to the effect of a one-unit change in age on the probability of choosing Auto over Transit, and the second corresponds to the effect of a one-unit change in age on the probability of choosing Plane over Transit.

Example 30.2: Nested Logit Modeling

(View the complete code for this example (bchcex2.sas) in the example repository.)

The standard logit model imposes the restriction of the independence from irrelevant alternatives (IIA) property, which implies proportional substitution across alternatives. When the IIA assumption does not hold, models with more flexibility are needed.

One of the most widely used models is called the nested logit. In a nested logit model, all the alternatives in a choice set can be partitioned into nests in such a way that the following conditions are true:

- The ratio of any two alternatives that are in the same nest is independent of the existence of all other alternatives. Hence, the IIA assumption holds within each nest.
- The ratio of any two alternatives that are in different nests is not independent of the existence of other alternatives in the two nests. Hence, the IIA assumption does not hold between different nests.

For more information about nested logit models, see the section "Nested Logit" on page 1217.

In the previous example of travel demand data, people are asked to choose among travel by auto, plane, or public transit. It seems that auto and public transit are more similar to each other than either of them are to plane, because the probability of choosing auto and public transit might rise by about the same proportion whenever the option of taking a plane is unavailable. A nested logit model that places auto and public transit in one nest and plane in another nest might seem more reasonable than the standard logit model.

You specify a nested logit model by using the TYPE=NLOGIT option, and you allocate the nests by using the NEST= option. In the following code, NEST=(1 2 1) implies that travel alternatives 1 (auto) and 3 (public transit) are in the first nest and travel alternative 2 (plane) is in the second nest. There are two nests in total.

The deviance information criterion (DIC) is used to select the model. The "Deviance Information Criterion" table (ODS table name DIC) contains the DIC of the model. The table is not displayed by default, but you can request it by specifying the DIC option in the PROC BCHOICE statement. For more information about the calculations, see the section "Deviance Information Criterion (DIC)" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The DIC option requests the calculation of DIC.

```
proc bchoice data=Travel2 seed=124 nmc=20000 nthin=2 dic;
   class Mode Subject / param=ref order=data;
   model Choice = Mode TravTime / choiceset=(Subject) type=nlogit nest=(1 2 1);
run;
```

Output 30.2.1 PROC BCHOICE Posterior Summary Statistics and DIC

The BCHOICE Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Mode Auto	10000	-0.2398	0.6574	-1.5032	1.1475
Mode Plane	10000	-2.0295	1.1816	-4.2724	0.3156
TravTime	10000	-0.5638	0.2042	-0.9638	-0.1872
Lambda 1	10000	0.8884	0.3540	0.2338	1.5334

Output 30.2.1 *continued*

Deviance Information Criterion	
Dbar (Posterior Mean of Deviance)	32.946
Dmean (Deviance Evaluated at Posterior Mean)	30.311
pD (Effective Number of Parameters)	2.635
DIC (Smaller is Better)	35.581

There are two alternatives in the first nest and one alternative in the second nest. A nest that has only one alternative is degenerate; therefore its λ is not estimable. This explains why there is only one λ estimate in the output.

The following statements revisit the standard logit model (the default type) that is discussed in the previous example and request that the calculation of DIC be displayed:

```
proc bchoice data=Travel2 seed=124 nmc=20000 nthin=2 dic;
   class Mode Subject / param=ref order=data;
   model Choice = Mode TravTime / choiceset=(Subject);
run;
```

Output 30.2.2 PROC BCHOICE Posterior Summary Statistics and DIC

The BCHOICE Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Mode Auto	10000	-0.1999	0.7341	-1.6891	1.1612
Mode Plane	10000	-2.0470	1.3376	-4.6183	0.5479
TravTime	10000	-0.6000	0.2333	-1.0716	-0.1986

Deviance Information Criterion	
Dbar (Posterior Mean of Deviance)	33.356
Dmean (Deviance Evaluated at Posterior Mean)	30.550
pD (Effective Number of Parameters)	2.806
DIC (Smaller is Better)	36.161

The DIC values are very close. A smaller DIC value indicates a better fit to the data; hence, the nested logit model might be a little better.

Example 30.3: Probit Modeling

(View the complete code for this example (bchcex3.sas) in the example repository.)

A more general model is the probit model, which is derived under the assumption of jointly normal random utility components, where the error vector has a multivariate normal distribution with a mean vector of $\mathbf{0}$ and a covariance matrix $\mathbf{\Sigma}$. For a full covariance matrix, PROC BCHOICE can accommodate any pattern of correlation and heteroscedasticity. Thus, this model fits a very general error structure. For more information about probit models, see the section "Probit" on page 1219.

In Train (2009), a project team collected the following data from commuters about four available travel modes for their trips to work: car alone, carpool, bus, and railway. The time and cost of travel for each mode were determined for each commuter, based on the location of the commuter's home and workplace.

```
data Commuter;
   input Subject Mode Choice Cost Time @@;
   datalines;
1 1 1 1.51 18.5 1 2 0 2.34 26.34 1 3 0 1.8 20.87 1 4 0 2.36 30.03 2 1 0
6.06 31.31 2 2 0 2.9 34.26 2 3 0 2.24 67.18 2 4 1 1.86 60.29 3 1 1 5.79
22.55 3 2 0 2.14 23.26 3 3 0 2.58 63.31 3 4 0 2.75 49.17 4 1 1 1.87
26.09 4 2 0 2.57 29.9 4 3 0 1.9 19.75 4 4 0 2.27 13.47 5 1 1 2.5 4.7 5 2
0 1.72 12.41 5 3 0 2.69 43.09 5 4 0 2.97 39.74 6 1 1 4.73 3.07 6 2 0
0.62 9.22 6 3 0 1.85 12.83 6 4 0 2.31 43.54 7 1 1 4.73 13.14 7 2 0 0.6
17.77 7 3 0 2.43 54.09 7 4 0 2 42.22 8 1 1 5.35 52.9 8 2 0 2.91 48.78 8
3 0 2.61 69.16 8 4 0 2.78 53.25 9 1 0 4.41 61.06 9 2 0 1.59 62.13 9 3 1

   ... more lines ...

450 1 1 4.59 29.44 450 2 0 2.89 33.73 450 3 0 1.9 66.12 450 4 0 1.79
39.84 451 1 1 3.24 16.35 451 2 0 1.21 18.98 451 3 0 1.75 23.39 451 4 0
2.02 43.3 452 1 0 6.93 65.42 452 2 0 1.17 60.48 452 3 1 2.46 52.4 452 4
0 2.61 48.37 453 1 0 6.53 59.57 453 2 1 1.41 55.14 453 3 0 2.21 67.82
453 4 0 1.86 73.45
;

proc print data=Commuter(obs=20);
run;
```

The variable Mode has the value 1 for car alone, 2 for carpool, 3 for bus, and 4 for railway. The variable Choice is the response variable that represents the decision among the four travel modes for each commuter. The order in which the alternatives appear in a choice set is important: the variable Mode should have the value 1 in the first alternative, 2 in the second alternative, then 3 in the third alternative, and finally 4 in the last alternative. The data for the first five commuters are shown in Output 30.3.1.

Output 30.3.1 Data for the First Five Commuters

Obs	Subject	Mode	Choice	Cost	Time
1	1	1	1	1.51	18.50
2	1	2	0	2.34	26.34
3	1	3	0	1.80	20.87
4	1	4	0	2.36	30.03
5	2	1	0	6.06	31.31
6	2	2	0	2.90	34.26
7	2	3	0	2.24	67.18
8	2	4	1	1.86	60.29
9	3	1	1	5.79	22.55
10	3	2	0	2.14	23.26
11	3	3	0	2.58	63.31
12	3	4	0	2.75	49.17
13	4	1	1	1.87	26.09
14	4	2	0	2.57	29.90
15	4	3	0	1.90	19.75
16	4	4	0	2.27	13.47
17	5	1	1	2.50	4.70
18	5	2	0	1.72	12.41
19	5	3	0	2.69	43.09
20	5	4	0	2.97	39.74

The following statements fit a probit model by specifying TYPE=PROBIT. The BCHOICE procedure's implementation of the Gibbs sampler for the probit model exhibits a higher autocorrelation than that for the logit model. High autocorrelation is created by introducing the latent variable via data augmentation because of the dependence between the latent variable and the regression parameters. You might want to control the thinning rate of the simulation. For example, THIN=10 keeps every 10th sample in the simulation and discards the rest.

```
proc bchoice data=Commuter outpost=Commupostsamp thin=10 nmc=100000 seed=123;
   class Mode(ref='1') Subject;
   model Choice = Cost Time Mode / choiceset=(Subject) type=probit;
run;
```

Output 30.3.2 shows the summary statistics for the part-worth ($\boldsymbol{\beta}$) of each of the attributes (Cost, Time, Mode 2, Mode 3, and Mode 4) and the covariance of the error difference vector ($\tilde{\boldsymbol{\Sigma}}$), which is displayed by parameters labeled "Sigma 1 1," "Sigma 2 1," and so on.

Output 30.3.2 Posterior Summary Statistics

The BCHOICE Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Cost	10000	-0.4627	0.0758	-0.6091	-0.3147
Time	10000	-0.0495	0.00565	-0.0611	-0.0391
Mode 2	10000	-3.3920	0.7358	-4.8987	-2.1087
Mode 3	10000	-1.9953	0.2934	-2.5705	-1.4381
Mode 4	10000	-1.6257	0.2113	-2.0267	-1.2089
Sigma 1 1	10000	1.0000	0	1.0000	1.0000
Sigma 2 1	10000	1.2107	0.5764	0.0448	2.3046
Sigma 2 2	10000	4.5411	2.1736	1.4132	8.6471
Sigma 3 1	10000	0.5720	0.2150	0.1437	0.9933
Sigma 3 2	10000	1.7167	0.9071	0.0199	3.5824
Sigma 3 3	10000	1.3891	0.4796	0.5995	2.3361

It is well known that an identification problem exists in probit models, because location and scale transformations do not change the choices that are made. The solution to the location shift is differencing with respect to the last alternative in each choice set. (See the section "Probit" on page 1219.) After that, a scale shift problem remains, because the parameters $(c\boldsymbol{\beta}, c^2\tilde{\boldsymbol{\Sigma}})$ for any constant $c > 0$ are equivalent to $(\boldsymbol{\beta}, \tilde{\boldsymbol{\Sigma}})$. A solution to the scaling problem is to normalize the parameters with respect to one of the diagonal elements of the covariance of the error difference vector, $\tilde{\boldsymbol{\Sigma}}$. PROC BCHOICE reports $(\boldsymbol{\beta}/\sqrt{\sigma_{11}}, \tilde{\boldsymbol{\Sigma}}/\sigma_{11})$ at each draw, where σ_{11} is the first diagonal entry of $\tilde{\boldsymbol{\Sigma}}$. This explains why "Sigma 1 1" is always 1 in Output 30.3.2.

By the IIA property in a logit model, it is assumed that all alternatives are independent and have the same variance. Therefore, the normalized covariance matrix after differencing with respect to one of the alternatives is of the form

$$\tilde{\boldsymbol{\Sigma}} = \begin{pmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{pmatrix}$$

Obviously, this matrix is quite different from the estimated normalized covariance matrix for this data set. Fitting a standard logit model would be inappropriate.

Example 30.4: A Random-Effects-Only Logit Model

(View the complete code for this example (bchcex4.sas) in the example repository.)

This example revisits the trash can study that is described earlier in this chapter in the getting-started section "A Logit Model Example with Random Effects" on page 1185.

If you want to create a random-effects-only model using the random walk Metropolis sampling as suggested in Rossi, Allenby, and McCulloch (2005), you can add the ALG=RWM option to the PROC BCHOICE statement to use the random walk Metropolis sampling algorithm instead of the default Gamerman Metropolis sampling, add the REMEAN option to the RANDOM statement to request estimation of the nonzero mean of the random effects, and not specify any fixed effects in the MODEL statement as follows:

```
proc bchoice data=Trashcan outpost=Postsamp seed=123 nmc=50000 thin=2
             alg=rwm nthreads=4;
   class ID Task;
   model Choice = / choiceset=(ID Task);
   random Touchless Steel AutoBag Price80 / sub=ID remean
          monitor=(1 to 5) type=un;
run;
```

Summary statistics for the mean of the random coefficients ($\bar{\boldsymbol{\gamma}}$), the covariance of the random coefficients ($\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$), and the random coefficients ($\boldsymbol{\gamma}_i$) for the first five individuals are shown in Output 30.4.1.

Output 30.4.1 Posterior Summary Statistics

The BCHOICE Procedure

Posterior Summaries and Intervals						
Parameter	Subject	N	Mean	Standard Deviation	95% HPD Interval	
REMean Touchless		25000	1.7281	0.2711	1.1983	2.2651
REMean Steel		25000	1.0663	0.2661	0.5577	1.5998
REMean AutoBag		25000	2.2014	0.3520	1.5394	2.9100
REMean Price80		25000	-4.7018	0.6545	-6.0553	-3.4883
RECov Touchless, Touchless		25000	3.1178	1.1462	1.3552	5.3414
RECov Steel, Touchless		25000	-0.4517	0.9922	-2.1999	1.2156
RECov Steel, Steel		25000	2.6809	1.0342	1.0335	4.6468
RECov AutoBag, Touchless		25000	-0.7233	0.9521	-2.6063	1.0850
RECov AutoBag, Steel		25000	0.0301	0.8003	-1.4271	1.5548
RECov AutoBag, AutoBag		25000	3.6804	1.5597	1.1236	6.7726
RECov Price80, Touchless		25000	-1.3032	1.2372	-3.8305	0.8279
RECov Price80, Steel		25000	-1.6533	1.2926	-4.4595	0.5892
RECov Price80, AutoBag		25000	-2.3293	1.7191	-5.7326	0.6730
RECov Price80, Price80		25000	7.8969	3.3306	2.1101	14.3416
Touchless	ID 1	25000	2.2223	1.0613	0.1295	4.2930
Steel	ID 1	25000	0.7433	1.0823	-1.3053	2.9046
AutoBag	ID 1	25000	3.7970	1.4406	1.1535	6.7240
Price80	ID 1	25000	-5.2465	2.3491	-10.0027	-0.9522
Touchless	ID 2	25000	-0.0612	0.8563	-1.6726	1.6508
Steel	ID 2	25000	-0.3962	0.8279	-2.1014	1.1849
AutoBag	ID 2	25000	2.7067	1.3353	0.1149	5.3223
Price80	ID 2	25000	0.0271	1.3024	-2.5173	2.6137
Touchless	ID 3	25000	1.1029	0.9627	-0.7106	3.1099
Steel	ID 3	25000	1.6602	0.9599	-0.1585	3.5981
AutoBag	ID 3	25000	0.2961	1.1676	-2.0028	2.5169
Price80	ID 3	25000	-0.9568	1.5615	-4.0940	2.0521
Touchless	ID 4	25000	1.1556	0.9722	-0.7835	3.0279
Steel	ID 4	25000	1.3922	1.0206	-0.5989	3.4310
AutoBag	ID 4	25000	3.2143	1.4620	0.4988	6.1835
Price80	ID 4	25000	-6.6490	2.5043	-11.7925	-2.0763
Touchless	ID 5	25000	0.4804	1.1831	-1.7839	2.8515
Steel	ID 5	25000	2.7119	1.2432	0.4436	5.2051
AutoBag	ID 5	25000	3.4736	1.6239	0.5679	6.8042
Price80	ID 5	25000	-5.1156	2.5674	-10.0957	-0.0400

The average part-worths ($\bar{\boldsymbol{\gamma}}$) for touchless opening, steel material, automatic trash bag replacement, and price, which are labeled as "REMean Touchless," "REMean Steel," "REMean AutoBag," and "REMean Price80," are very similar to the estimates of the fixed effects in the previous model as shown in Figure 30.9, indicating the equivalence of these two setups. The covariance estimates of the random coefficients ($\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$) are also very similar.

The next set of parameters show the estimates for the individual random effects for the first five respondents (see Output 30.4.1). However, these estimates are no longer the deviation from the overall mean but are each

respondent's own effects. The part-worth for touchless opening for the first respondent (ID 1) is 2.2, which is close to the part-worth that is estimated in the model as shown in Figure 30.9.

A caterpillar plot is a side-by-side bar plot of the 95% intervals for multiple variables. A caterpillar plot is usually used to visualize and compare individual random-effects parameters, which can exist in large numbers in certain models. You can use the %CATER autocall macro to create a caterpillar plot. The %CATER macro requires you to specify an input data set and a list of variables that you want to plot.

In this example, the output data set Postsamp contains posterior draws for all the random-effects parameters. The following statements generate a caterpillar plot for Touchless:

```
%CATER(data=Postsamp, var=Touchless_ID_:)
```

Output 30.4.2 is a caterpillar plot of the individual random effects of touchless opening for the 104 participants in this study. It displays the heterogeneity in preferences for touchless opening among the sample. As shown, the individuals have diverse preferences for the feature of touchless opening for a trash can. Heterogeneity of preferences is an important aspect of the model, and ignoring its presence can lead to incorrect inferences.

Output 30.4.2 Caterpillar Plot of the Random-Effects Parameters

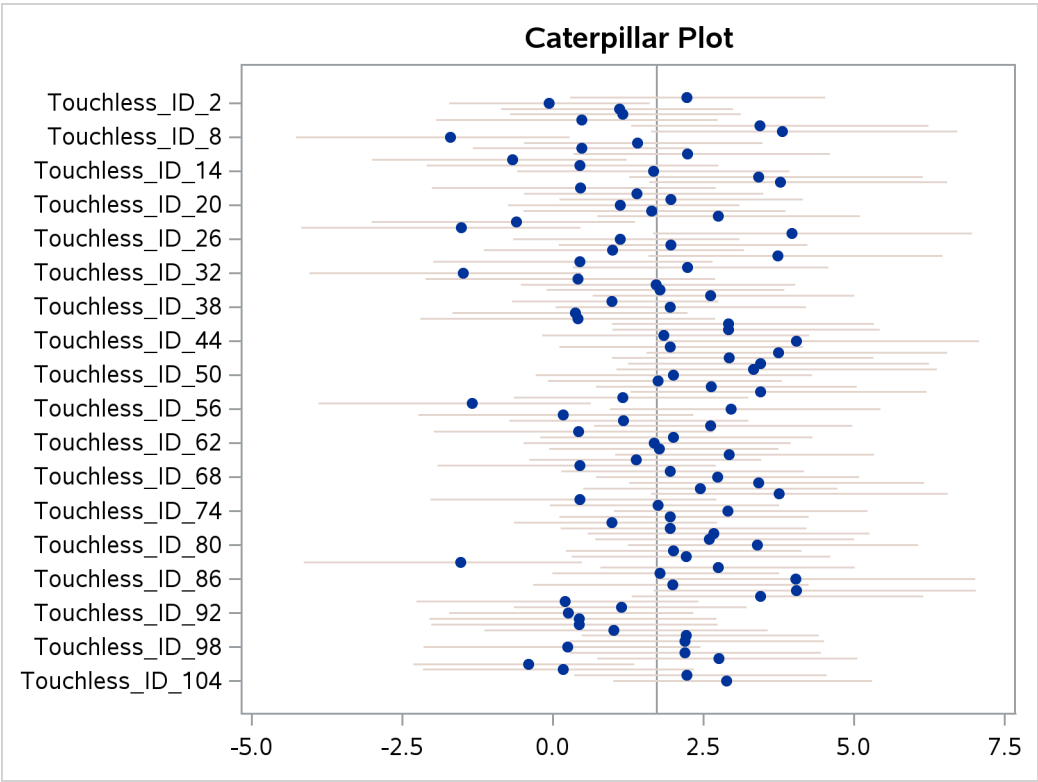

For marketing simulation purposes, you might want to obtain estimates of the random effects for all the respondents in order to derive other quantities of interest: you can add an OUTSUBCOEFF= option to create an output data set in which to save the estimates of subject-level (individual) random-effects coefficients. If you also want to obtain the parameter estimates at the population level, you can insert **`ods output PostSumInt=PopCoeff;`** before calling the BCHOICE procedure, as shown in the following statements. Inserting this statement produces a new SAS data set named PopCoeff in which to save the posterior statistics of the parameters (the population mean and covariance of the random effects), as listed in the "Posterior

Summaries and Intervals" table.

```
ods output PostSumInt=PopCoeff;
proc bchoice data=Trashcan outpost=Postsamp seed=123 nmc=50000 thin=2
             alg=rwm nthreads=4 plots=none;
   class ID Task;
   model Choice = / choiceset=(ID Task);
   random Touchless Steel AutoBag Price80 / sub=ID remean
          type=un outsubcoeff=SubCoeff;
run;

proc print data=SubCoeff (obs=20);
run;

proc print data=PopCoeff(drop=ParameterName);
run;
```

The first 20 observations of the SubCoeff data set are shown in Output 30.4.3. The columns in this data set list the random effects, "Touchless," "Steel," "AutoBag," and "Price80"; the rows are the subjects that are specified in the SUBJECT= option in the RANDOM statement. There are 104 rows (subjects) in total in this SubCoeff data set.

Output 30.4.3 Subject-Level Random-effects Coefficients

Obs	Subject	Touchless	Steel	AutoBag	Price80
1	ID 1	2.22231	0.74333	3.79703	-5.24650
2	ID 2	-0.06116	-0.39625	2.70671	0.02705
3	ID 3	1.10292	1.66015	0.29609	-0.95682
4	ID 4	1.15563	1.39224	3.21427	-6.64898
5	ID 5	0.48039	2.71190	3.47356	-5.11563
6	ID 6	3.43739	0.68635	1.09859	-6.28209
7	ID 7	3.80949	1.10457	1.51311	-5.08475
8	ID 8	-1.70400	-0.17082	2.78074	0.84940
9	ID 9	1.40801	1.66734	3.61761	-5.19075
10	ID 10	0.48025	0.69532	3.67706	-3.90143
11	ID 11	2.23090	0.69172	3.84367	-5.29062
12	ID 12	-0.67359	1.46822	3.15258	-5.15036
13	ID 13	0.44859	2.74937	3.43585	-5.14505
14	ID 14	1.67260	2.68559	0.44953	-5.83748
15	ID 15	3.41221	0.69942	0.96101	-6.14496
16	ID 16	3.77795	1.08635	1.52534	-5.06281
17	ID 17	0.46074	2.72260	3.45382	-5.10106
18	ID 18	1.39268	1.64135	3.59687	-5.14450
19	ID 19	1.96137	0.46858	3.46723	-6.72558
20	ID 20	1.11386	1.36409	3.27232	-6.63044

Output 30.4.4 shows the PopCoeff data set, which contains the posterior means, standard deviations, and HPD credible intervals of the population mean and the covariance of the random effects.

Output 30.4.4 Population Mean and Covariance of the Random Effects

Obs	Parameter	N	Mean	StdDev	HPDLower	HPDUpper
1	REMean Touchless	25000	1.7281	0.2711	1.1983	2.2651
2	REMean Steel	25000	1.0663	0.2661	0.5577	1.5998
3	REMean AutoBag	25000	2.2014	0.3520	1.5394	2.9100
4	REMean Price80	25000	-4.7018	0.6545	-6.0553	-3.4883
5	RECov Touchless, Touchless	25000	3.1178	1.1462	1.3552	5.3414
6	RECov Steel, Touchless	25000	-0.4517	0.9922	-2.1999	1.2156
7	RECov Steel, Steel	25000	2.6809	1.0342	1.0335	4.6468
8	RECov AutoBag, Touchless	25000	-0.7233	0.9521	-2.6063	1.0850
9	RECov AutoBag, Steel	25000	0.0301	0.8003	-1.4271	1.5548
10	RECov AutoBag, AutoBag	25000	3.6804	1.5597	1.1236	6.7726
11	RECov Price80, Touchless	25000	-1.3032	1.2372	-3.8305	0.8279
12	RECov Price80, Steel	25000	-1.6533	1.2926	-4.4595	0.5892
13	RECov Price80, AutoBag	25000	-2.3293	1.7191	-5.7326	0.6730
14	RECov Price80, Price80	25000	7.8969	3.3306	2.1101	14.3416

Example 30.5: Heterogeneity Affected by Individual Characteristics

(View the complete code for this example (bchcex5.sas) in the example repository.)

Rossi, Allenby, and McCulloch (2005) studied a scanner panel data about purchases of margarine. The data were first analyzed in Allenby and Rossi (1991) and are about purchases of ten brands of margarine. This example considers a subset of data about six margarine brands: Parkay stick, Blue Bonnet stick, Fleischmann's stick, a house brand stick, a generic stick, and Shedd's Spread tub. There are 313 households, which made a total of 3,405 purchases. Information about a few demographic characteristics of these households (income and family size) is expected to have effects on the central location of the distribution of heterogeneity.

The data set, which is called Sashelp.Margarin, comes from the SASHELP library.

```
proc print data=Sashelp.Margarin (obs=24);
   by HouseID Set;
   id HouseID Set;
run;
```

The data for the first four choice sets are shown in Figure 30.5.1.

Output 30.5.1 Data for the First Four Choice Sets

HouseID	Set	Choice	Brand	LogPrice	LogInc	FamSize
2100016	**1**	1	PPk	-0.41552	3.48124	2
		0	PBB	-0.40048	3.48124	2
		0	PFl	0.08618	3.48124	2
		0	PHse	-0.56212	3.48124	2
		0	PGen	-1.02165	3.48124	2
		0	PSS	-0.16252	3.48124	2

Output 30.5.1 *continued*

HouseID	Set	Choice	Brand	LogPrice	LogInc	FamSize
2100016	**2**	1	PPk	-0.46204	3.48124	2
		0	PBB	-0.40048	3.48124	2
		0	PFl	-0.01005	3.48124	2
		0	PHse	-0.56212	3.48124	2
		0	PGen	-1.02165	3.48124	2
		0	PSS	-0.16252	3.48124	2

HouseID	Set	Choice	Brand	LogPrice	LogInc	FamSize
2100016	**3**	1	PPk	-1.23787	3.48124	2
		0	PBB	-0.69315	3.48124	2
		0	PFl	-0.01005	3.48124	2
		0	PHse	-0.56212	3.48124	2
		0	PGen	-1.02165	3.48124	2
		0	PSS	-0.23572	3.48124	2

HouseID	Set	Choice	Brand	LogPrice	LogInc	FamSize
2100016	**4**	1	PPk	-0.47804	3.48124	2
		0	PBB	-0.49430	3.48124	2
		0	PFl	-0.01005	3.48124	2
		0	PHse	-0.56212	3.48124	2
		0	PGen	-1.02165	3.48124	2
		0	PSS	-0.16252	3.48124	2

The variable HouseID represents the household ID, and each household made at least five purchases, which are defined by Set. The variable Choice represents the choice made among the six margarine brands for each purchase or choice set. The variable Brand has the value PPK for Parkay stick, PBB for Blue Bonnet stick, PFL for Fleischmann's stick, PHse for the house brand stick, PGen for the generic stick, and PSS for Shedd's Spread tub. The variable LogPrice is the logarithm of the product price. The variables LogInc and variable FamSize provide information about household income and family size, respectively.

The following statements fit a random-effects-only logit model using Gamerman Metropolis sampling

```
proc bchoice data=Sashelp.Margarin seed=123 nmc=40000 thin=2
   nthreads=4 plots=none;
   class Brand(ref='PPk') HouseID Set;
   model Choice = / choiceset=(HouseID Set);
   random  Brand LogPrice / subject=HouseID remean=(LogInc FamSize)
           type=un monitor=(1);
run;
```

The REMEAN=(LOGINC FAMSIZE) option in the RANDOM statement requests estimation of the nonzero mean of the random effects, which is a function of household income and family size. No fixed effects are specified in the MODEL statement.

The NMC= option in the PROC BCHOICE statement requests to run the MCMC chain for 40,000 iterations, and the THIN=2 option keeps one of every two samples for analysis. If ODS Graphics is enabled and you do not specify the PLOTS option, then PROC BCHOICE produces, for each parameter, a panel that contains the trace plot, the autocorrelation function plot, and the kernel density plot by default. In models that have a large number of parameters, it will take a long time to display all the plots. So PLOTS=NONE is specified to stop

producing the plots. If you want to see the plots for some parameters, you can save the posterior samples to a SAS data set using the OUTPOST= option and then generate the plots using the %TADPLOT macro as explained in the section "Regenerating Diagnostics Plots" on page 1235.

Summary statistics for the mean matrix of the random coefficients ($\mathbf{\Gamma}$), the covariance of the random coefficients ($\mathbf{\Omega}_{\boldsymbol{\gamma}}$), and the random coefficients ($\boldsymbol{\gamma}_i$) for the first household are shown in Output 30.5.2.

Output 30.5.2 Posterior Summary Statistics

The BCHOICE Procedure

Posterior Summaries and Intervals						
Parameter	Subject	N	Mean	Standard Deviation	95% HPD Interval	
REMean Brand PBB		20000	-1.2079	0.6384	-2.4293	0.0686
REMean Brand PFl		20000	-3.2484	1.9276	-7.0351	0.5201
REMean Brand PGen		20000	-5.1130	1.2332	-7.6390	-2.8030
REMean Brand PHse		20000	-3.2595	0.9194	-5.0725	-1.4761
REMean Brand PSS		20000	0.0915	1.2127	-2.3015	2.4981
REMean LogPrice		20000	-3.4148	0.8359	-5.0397	-1.7620
REMean Brand PBB LogInc		20000	0.0529	0.2114	-0.3485	0.4811
REMean Brand PFl LogInc		20000	0.7596	0.6208	-0.4726	1.9749
REMean Brand PGen LogInc		20000	-0.5079	0.4019	-1.2977	0.2698
REMean Brand PHse LogInc		20000	0.0315	0.3029	-0.5949	0.5931
REMean Brand PSS LogInc		20000	-0.6315	0.4131	-1.4645	0.1555
REMean LogPrice LogInc		20000	-0.2837	0.2817	-0.8434	0.2631
REMean Brand PBB FamSize		20000	-0.0274	0.0959	-0.2180	0.1572
REMean Brand PFl FamSize		20000	-0.7357	0.3059	-1.3283	-0.1267
REMean Brand PGen FamSize		20000	0.5775	0.1824	0.2269	0.9428
REMean Brand PHse FamSize		20000	0.2365	0.1357	-0.0291	0.4997
REMean Brand PSS FamSize		20000	0.0528	0.1974	-0.3347	0.4425
REMean LogPrice FamSize		20000	0.1010	0.1273	-0.1542	0.3424
RECov Brand PBB, Brand PBB		20000	2.2081	0.3730	1.5261	2.9615
RECov Brand PFl, Brand PBB		20000	1.8598	0.9106	0.1374	3.6776
RECov Brand PFl, Brand PFl		20000	12.0894	3.9050	5.5544	20.0864
RECov Brand PGen, Brand PBB		20000	1.9842	0.5697	0.8563	3.0829
RECov Brand PGen, Brand PFl		20000	1.3300	1.9065	-2.4622	5.1401
RECov Brand PGen, Brand PGen		20000	8.3897	1.4835	5.5924	11.3433
RECov Brand PHse, Brand PBB		20000	1.5148	0.4402	0.6799	2.3928
RECov Brand PHse, Brand PFl		20000	2.1554	1.3869	-0.5797	4.9030
RECov Brand PHse, Brand PGen		20000	5.7576	0.9570	3.8799	7.6015
RECov Brand PHse, Brand PHse		20000	5.4834	0.8441	3.9355	7.1970
RECov Brand PSS, Brand PBB		20000	1.1860	0.6287	-0.0412	2.4223
RECov Brand PSS, Brand PFl		20000	0.6096	1.9471	-3.0709	4.6460
RECov Brand PSS, Brand PGen		20000	4.8189	1.1738	2.5960	7.1020
RECov Brand PSS, Brand PHse		20000	3.4484	0.8805	1.7068	5.1649
RECov Brand PSS, Brand PSS		20000	8.7098	1.8047	5.4222	12.2276
RECov LogPrice, Brand PBB		20000	-0.2260	0.3462	-0.8764	0.4813
RECov LogPrice, Brand PFl		20000	2.1909	0.9010	0.5220	4.0796
RECov LogPrice, Brand PGen		20000	-0.9989	0.6371	-2.2175	0.2793
RECov LogPrice, Brand PHse		20000	-0.4254	0.5043	-1.3751	0.6150
RECov LogPrice, Brand PSS		20000	0.1734	0.6757	-1.1897	1.4765
RECov LogPrice, LogPrice		20000	2.1279	0.5373	1.1697	3.2261
Brand PBB	HouseID 2100016	20000	-2.3546	1.0548	-4.4528	-0.3948
Brand PFl	HouseID 2100016	20000	-3.9792	2.7840	-9.4491	0.9694
Brand PGen	HouseID 2100016	20000	-6.5984	1.6472	-9.8601	-3.4056
Brand PHse	HouseID 2100016	20000	-2.9722	1.2014	-5.5148	-0.8522
Brand PSS	HouseID 2100016	20000	-3.3807	2.1701	-7.4577	0.7600
LogPrice	HouseID 2100016	20000	-4.6129	1.2361	-6.9996	-2.1437

Table 30.11 collects the posterior means and standard deviations of $\bar{\mathbf{\Gamma}}$ that are shown in Output 30.5.2. The first column corresponds to the parameters that are specified in the model, namely the brands and price. The second column shows the average part-worths of each brand (versus the brand, Parkay stick) and the price at LogInc=0 and FamSize=0. The LogInc and FamSize columns list the modifying effects on the preference for each brand and price by household income and family size, respectively. Larger families show more interest in the generic and house brands and tend to stay away from the Fleischmann's brand. For example, consider the part-worth estimates for Fleischmann's. The posterior mean for REMean Brand PFI FamSize (the Fleischmann's row and the Famsize column) is –0.74 with a standard deviation of 0.31, meaning that an additional unit increase in family size is associated with a reduction of 0.74 in the estimated part-worth for Fleischmann's. In general, the demographics of households are only weakly associated with preference for brand and price. These results are in good agreement with those of Rossi, Allenby, and McCulloch (2005).

Table 30.11 Posterior Mean and Standard Deviation of $\mathbf{\Gamma}$

Parameter	Intercept	LogInc	FamSize
Blue Bonnet	REMean Brand PBB –1.21 0.64	REMean Brand PBB LogInc 0.05 0.21	REMean Brand PBB FamSize –0.03 0.10
Fleisch-mann's	REMean Brand PFI –3.25 1.93	REMean Brand PFI LogInc 0.76 0.62	REMean Brand PFI FamSize –0.74 0.31
Generic	REMean Brand PGen –5.11 1.23	REMean Brand PGen LogInc –0.51 0.40	REMean Brand PGen FamSize 0.58 0.18
House	REMean Brand PHse –3.26 0.92	REMean Brand PHse LogInc 0.03 0.30	REMean Brand PHse FamSize 0.24 0.14
Shedd's Spread	REMean Brand PSS 0.09 1.21	REMean Brand PSS LogInc –0.63 0.41	REMean Brand PSS FamSize –0.05 0.20
LogPrice	REMean LogPrice –3.41 0.84	REMean LogPrice LogInc –0.28 0.28	REMean LogPrice FamSize 0.10 0.13

Because the demographic variables are not zero-centered, the Intercept column shows the average part-worths of each brand and price for households with LogInc=0 and FamSize=0, which are not very meaningful. It is better to center demographic variables by their means, so that the posterior means listed in the Intercept column can be interpreted as the part-worths of a household that has an average income and average size.

Nevertheless, you can obtain the utilities of households that have any income levels and sizes. For example, the average part-worth of the Fleischmann's brand for a household with average income (LogInc=3.1) and family size (FamSize=3) would be as follows, because the estimated LogInc coefficient is 0.76 and the estimated FamSize coefficient is –0.74 for Fleischmann's:

$$-3.25 + 0.76 \times 3.1 - 0.74 \times 3 = -3.11$$

You can obtain part-worths for all other brands and compare their popularity among average households.

The posterior means and standard deviations of the covariance matrix of the random coefficients ($\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$) are displayed by parameters that are labeled "RECov Brand PBB, Brand PBB," "RECov Brand PFI, Brand PBB," and so on. Some of the diagonal terms are fairly large, indicating that there is quite a bit of heterogeneity between households in margarine brand preference and price sensitivity. The covariance between the generic and house brands, "RECov Brand PHse, Brand PGen," is fairly large, suggesting that household preferences for these two brands are highly correlated.

The next set of parameters, which are displayed in Output 30.5.2, contain the estimates for the random effects for the first household.

Example 30.6: Quantities of Interest and Willingness to Pay

(View the complete code for this example (bchcex6.sas) in the example repository.)

It is easy to give a model an interpretation in terms of a probability ratio (PR). The PR of choosing alternative a instead of alternative b while holding all other attributes or conditions the same is

$$\mathrm{PR}_{ab} = \frac{P_a}{P_b} = \frac{\exp(\beta_a)}{\exp(\beta_b)} = \exp(\beta_a - \beta_b)$$

where P_a and P_b are the probabilities of choosing alternative a and b, respectively.

This example considers the fabric softener data simulated from Kuhfeld (2010). There are five fictitious fabric softener brands involved: Sploosh, Plumbbob, Platter, Moosey, and Another, where Another is the reference brand. Most brands were available at these three prices: 1.49, 1.99, and 2.49, except that the Another brand was only offered at 1.49.

```
data fabric;
   input Subj Set Choice Price Brand $ @@;
   datalines;
1       1       0       1.99    Sploosh
1       1       0       1.99    Plumbbob
1       1       1       1.99    Platter
1       1       0       2.49    Moosey
1       1       0       1.99    Another
1       2       0       2.49    Sploosh
1       2       0       1.49    Plumbbob
1       2       1       1.49    Platter
1       2       0       1.99    Moosey

   ... more lines ...

50      18      0       2.49    Sploosh
50      18      0       2.49    Plumbbob
50      18      1       1.99    Platter
50      18      0       2.49    Moosey
50      18      0       1.99    Another
;
```

The following statements fit a logit model with fixed effects only.

```
proc bchoice data=fabric seed=123 out=fabric_out;
   class Brand(ref='Another' order=data) Subj Set;
   model Choice = Brand Price / choiceset=(Subj Set);
run;
```

The posterior summary statistics are shown in Output 30.6.1. The Another brand is the base category, whose part-worth is normalized to zero. The most to least preferred brands are Platter, Moosey, Another, Plumbbob, and Sploosh. The average part-worth for having to pay an extra dollar is negative (–4.6).

Output 30.6.1 PROC BCHOICE Posterior Summary Statistics

The BCHOICE Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Brand Sploosh	5000	-1.3133	0.2141	-1.7429	-0.8967
Brand Plumbbob	5000	-0.4940	0.1818	-0.8256	-0.1232
Brand Platter	5000	2.0984	0.1457	1.8241	2.3891
Brand Moosey	5000	0.6273	0.1598	0.3313	0.9504
Price	5000	-4.6210	0.2177	-5.0567	-4.2141

The PR of choosing Sploosh as opposed to Another (which is the reference category) simplifies to

$$\mathrm{PR}_{(\text{Sploosh vs. Another})} = \frac{\exp(-1.31)}{\exp(0)} = 0.27$$

This ratio indicates that the likelihood of choosing Sploosh is 0.27 times that of choosing Another. Sploosh is less favorable than Another.

The PR of choosing Sploosh instead of Plumbbob is

$$\mathrm{PR}_{(\text{Sploosh vs. Plumbbob})} = \frac{\exp(-1.31)}{\exp(-0.49)} = 0.44$$

This shows that Sploosh is less preferred than Plumbbob at the same price.

You can derive the probabilities for any alternatives and any combinations of attributes, even imaginary ones. Suppose now that you want to also take price into consideration. For example, the PR of choosing Sploosh at \$1.49 as opposed to Plumbbob at \$1.99 is

$$\frac{\exp(\beta_{\text{Sploosh}} + 1.49 * \beta_{\text{Price}})}{\exp(\beta_{\text{Plumbbob}} + 1.99 * \beta_{\text{Price}})} = \frac{\exp(-1.31 - 1.49 * 4.62)}{\exp(-0.49 - 1.99 * 4.62)} = 4.44$$

This indicates that the likelihood of choosing Sploosh at \$1.49 is 4.44 times that of choosing Plumbbob at \$1.99. Offering some discount on price makes Sploosh more preferred than Plumbbob.

Although it is easy to obtain the point estimates, sometimes you might want to estimate other quantities, such as the standard deviations or various quantiles. If you are interested in making inference based on any quantities that are transformations of the random variables, you can do it either directly in PROC BCHOICE or by using the DATA step after you run the simulation.

Suppose you want to set the reference category as Plumbbob so that you can directly compare any other brand with it, you can specify REF='PLUMBBOB' in the CLASS statement and rerun the PROC BCHOICE simulation. You can also use the DATA step to calculate the quantities of interest. The following DATA step uses the simulated values of Brand_Sploosh and Brand_Plumbbob from the output data set fabric_out to create a new series of posterior samples for Sploosh_Plumbbob, a new variable that is created to calculate the PR of these two brands:

```
data Transout;
   set fabric_out;
   Sploosh_Plumbbob=exp(Brand_Sploosh - Brand_Plumbbob);
run;
```

Then you can use some autocall macros to analyze the posterior samples of Sploosh_Plumbbob. For example, the %SUMINT macro provides summary statistics and 95% HPD intervals as described in the section "Autocall Macros for Postprocessing" on page 1233.

```
%SUMINT(data=Transout, var=Sploosh_Plumbbob)
```

The results from using the %SUMINT macro are shown in Output 30.6.2. The posterior mean of the PR of choosing Sploosh over Plumbbob is about the same as the one that is computed just above. It also provides the standard deviation and 95% HPD interval, which indicates how spread out the distribution of the PR is.

Output 30.6.2 Sploosh versus Plumbbob

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Sploosh_Plumbbob	5000	0.4497	0.0907	0.2722	0.6327

The ratio of coefficients in most choice models has economic meanings. Part-worths are measure in utility units. Can you convert these to dollar equivalents? The answer is to compute the Willingness to Pay (WTP).

One's WTP for attribute a versus attribute b is the increase in purchase price that keeps the utilities equal. You can approximate WTP by dividing the difference in the attributes' part-worths by the price coefficient, given that the price coefficient is less than zero.

$$\text{WTP(a vs. b)} = \frac{\beta_a - \beta_b}{\beta_{\text{price}}}$$

If WTP(a vs. b) $=$ s and s > 0, you say that attribute b is preferred to attribute a when offered at the same price and one is willing to pay up to the amount s to stay with b; If WTP(a vs. b) $=$ s and s < 0, you say that attribute a is preferred to attribute b when offered at the same price and one is willing to pay up to the amount |s| to stay with a.

$$\text{WTP(Sploosh vs. Plumbbob)} = \frac{-1.31 + 0.49}{-4.62} = 0.18$$

Therefore, most subjects prefer Plumbbob over Sploosh at the same price; and they are willing to pay up to 0.18 dollars more to stay with Plumbbob.

You can even approximate the posterior distributions of the WTPs, comparing each brand to Another by using the DATA step:

```
data WTPout;
   set fabric_out;
   Sploosh_WTP = Brand_Sploosh/price;
   Plumbbob_WTP= Brand_Plumbbob/price;
   Platter_WTP = Brand_Platter/price;
   Moosey_WTP  = Brand_Moosey/price;
run;
```

The posterior distributions of the WTPs is plotted in Figure 30.6.3. The WTP of Sploosh versus Another and the WTP of Plumbbob versus Another are both located to the right of the zero line, meaning that Another is more popular than both Sploosh and Plumbbob; and the WTP of Sploosh versus Another is further away from the zero line compared to the WTP of Plumbbob versus Another, indicating that people are willing to pay more money to stay with Another before switching to Sploosh than switching to Plumbbob. Obtaining the whole distribution of the WTPs provides more information than having just the mean: it displays how well the mean estimates and how spread out the distribution is.

Output 30.6.3 Willingness to Pay Plot

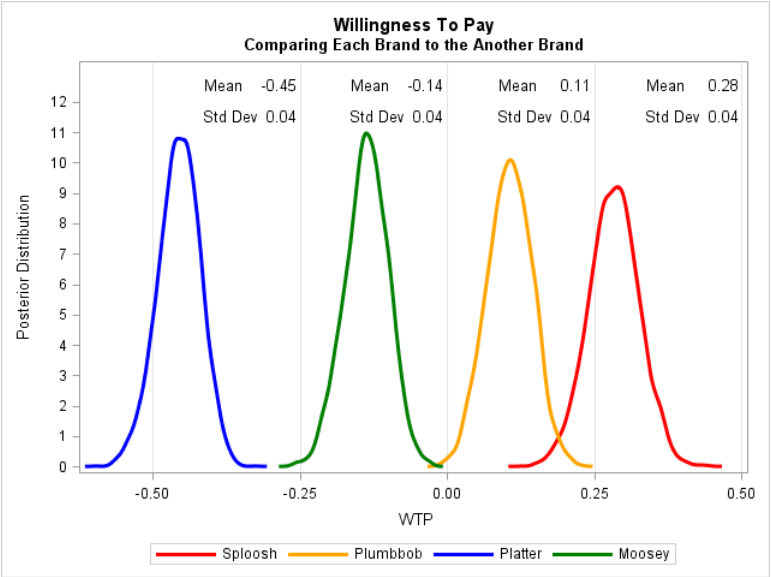

Example 30.7: Predict the Choice Probabilities

(View the complete code for this example (bchcex7.sas) in the example repository.)

This example shows how to obtain the posterior predictive distribution of the choice probability that each alternative is chosen from a choice set. The posterior predictive distribution enables you to get the expected choice probabilities of all the alternatives in the data, or even to predict market share for simulated or hypothetical products or marketplaces that do not directly reflect the choice set in the data.

Suppose you have a data set that contains the attribute variables (the design matrix) for all the alternatives in a choice set. For example, in the candy study earlier in the chapter, in the section "A Simple Logit Model Example" on page 1178, you can use the same eight alternatives: Dark is 1 for dark chocolate and 0 for soft

chocolate; Soft is 1 for soft center and 0 for chewy center; Nuts is 1 if the candy contains nuts and 0 if it contains no nuts. The following data set contains the eight alternatives:

```
data DesignMatrix;
   input Dark Soft Nuts;
   datalines;
0 0 0
0 0 1
0 1 0
0 1 1
1 0 0
1 0 1
1 1 0
1 1 1
;
```

You can use the PREDDIST statement, which obtains samples from the posterior predictive distribution of each of the choice probabilities by using the posterior samples of parameters in the model:

```
proc bchoice data=Chocs outpost=Bsamp nmc=10000 thin=2 seed=124;
   class Dark(ref='0') Soft(ref='0') Nuts(ref='0') Subj;
   model Choice = Dark Soft Nuts / choiceset=(Subj);
   preddist covariates=DesignMatrix nalter=8 outpred=Predout;
run;

%SUMINT(data=Predout, var=Prob_1_:)
```

In the PREDDIST statement, the COVARIATES= option names the data set to contain the explanatory variable values for which the predictions are established. This data set must contain data that have the same variables that the model uses. The NALTER= option specifies the number of alternatives in each choice set in the COVARIATES= data set. All choice sets in the data must have the same number of alternatives. If you omit the COVARIATES= option, the DATA= data set that you specify in the PROC BCHOICE statement is used instead. The OUTPRED= option creates an output data set to contain the samples from the posterior predictive distribution of the choice probabilities. Then you can use SAS autocall macros to analyze the posterior samples. For example, the %SUMINT macro provides summary statistics (means and standard deviations) and the 95% HPD intervals.

You can predict the choice probabilities by using the means of the posterior distributions. The results from using the %SUMINT macro are shown in Output 30.7.1. There is only one choice set for choice probability prediction, in which there are a total of eight alternatives. This explains the parameter names in the first column of the output, where the first number indexes the choice sets and the second number indexes the alternatives in each choice set. The most preferred chocolate candy is the sixth one, Dark/Chewy/Nuts, which takes about half the market.

Output 30.7.1 Choice of Chocolate Candies

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Prob_1_1	5000	0.0554	0.0425	0.00395	0.1434
Prob_1_2	5000	0.1309	0.0801	0.0136	0.2865
Prob_1_3	5000	0.00686	0.00896	0.000165	0.0244
Prob_1_4	5000	0.0158	0.0179	0.000448	0.0535
Prob_1_5	5000	0.2102	0.1033	0.0297	0.3974
Prob_1_6	5000	0.4946	0.1315	0.2407	0.7439
Prob_1_7	5000	0.0262	0.0279	0.000607	0.0830
Prob_1_8	5000	0.0601	0.0526	0.00329	0.1669

Example 30.8: MaxDiff Choice Example

(View the complete code for this example (bchcex8.sas) in the example repository.)

This example uses a MaxDiff type of choice experiment from the Sydney transport project. The study sought to help set priorities for future improvements in public transportation. The data were collected by the Center for the Study of Choice (CenSoC) at the University of Technology Sydney, and analyzed previously by Magidson and Vermunt (2015).

Respondents were asked to choose from among nine short-term improvements (nine alternatives). Table 30.12 lists the nine improvements that were evaluated.

Table 30.12 Alternatives in Sydney Transport Project

Index	Improvement Alternatives
1	More frequent off-peak trains between major centers
2	Improved peak rail capacity
3	More frequent bus service on major routes
4	Extension of light rail services
5	Integrated fares
6	Integrated ticketing
7	Real-time arrival information
8	New cycleways, more bike and scooter parking
9	Use of green power for trains

```
data Sydney;
   input RespID Set Choice Index @@;
   datalines;
5 1 1 2 5 1 0 4 5 1 -1 8 5 2 0 1 5 2 1 4 5 2 -1 5 5 3 0 4 5 3 -1 7 5 3 1
9 5 4 1 3 5 4 0 4 5 4 -1 6 5 5 -1 1 5 5 1 2 5 5 0 3 5 6 1 2 5 6 -1 5 5 6
0 7 5 7 1 2 5 7 -1 6 5 7 0 9 5 8 -1 1 5 8 0 8 5 8 1 9 5 9 0 5 5 9 -1 6 5
9 1 8 5 10 1 3 5 10 -1 7 5 10 0 8 5 11 1 1 5 11 0 6 5 11 -1 7 5 12 1 3 5
12 -1 5 5 12 0 9 6 1 1 2 6 1 0 4 6 1 -1 8 6 2 1 1 6 2 -1 4 6 2 0 5 6 3
-1 4 6 3 1 7 6 3 0 9 6 4 0 3 6 4 -1 4 6 4 1 6 6 5 -1 1 6 5 1 2 6 5 0 3 6
```

```
6 1 2 6 6 0 5 6 6 -1 7 6 7 1 2 6 7 0 6 6 7 -1 9 6 8 1 1 6 8 -1 8 6 8 0 9
6 9 0 5 6 9 1 6 6 9 -1 8 6 10 1 3 6 10 0 7 6 10 -1 8 6 11 1 1 6 11 0 6 6
11 -1 7 6 12 1 3 6 12 0 5 6 12 -1 9 7 1 1 2 7 1 0 4 7 1 -1 8 7 2 1 1 7 2

   ... more lines ...

350 1 -1 8 350 2 0 1 350 2 -1 4 350 2 1 5 350 3 0 4 350 3 -1 7 350 3 1 9
350 4 0 3 350 4 -1 4 350 4 1 6 350 5 -1 1 350 5 1 2 350 5 0 3 350 6 1 2
350 6 0 5 350 6 -1 7 350 7 1 2 350 7 0 6 350 7 -1 9 350 8 1 1 350 8 -1 8
350 8 0 9 350 9 0 5 350 9 1 6 350 9 -1 8 350 10 1 3 350 10 -1 7 350 10 0
8 350 11 0 1 350 11 1 6 350 11 -1 7 350 12 1 3 350 12 0 5 350 12 -1 9
;

proc print data=Sydney(obs=18);
run;
```

Specifically, each of the 200 respondents was given 12 choice tasks; each task consisted of a subset of three of these nine short-term improvements (alternatives). For each of these tasks, the respondent was asked to choose an alternative that should receive the highest priority (indicated by 1) and an alternative that should receive the lowest priority (indicated by –1).

The data for the first several choice sets are shown in Output 30.8.1. The variable RespID identifies each respondent. The variable Set indexes the 12 choice tasks that are given to each respondent. The variable Choice is the response variable that represents the decision: 1 for the highest priority, –1 for the lowest priority, and 0 for those in between. The variable Index has values from 1 to 9, each of which represents one of the nine improvement alternatives. This variable is listed in the CLASS statement as a classification variable, and the last level, 9, is the reference level. The respondent, with RespID=5, answered 12 choice questions, but the table shows only the first six choice tasks. This person picked the second alternative (improved peak rail capacity) as the highest priority, and the eighth alternative (new cycleways, more bike and scooter parking) as the lowest priority.

Output 30.8.1 Data for the First Several MaxDiff Choice Sets

Obs	RespID	Set	Choice	Index
1	5	1	1	2
2	5	1	0	4
3	5	1	-1	8
4	5	2	0	1
5	5	2	1	4
6	5	2	-1	5
7	5	3	0	4
8	5	3	-1	7
9	5	3	1	9
10	5	4	1	3
11	5	4	0	4
12	5	4	-1	6
13	5	5	-1	1
14	5	5	1	2
15	5	5	0	3
16	5	6	1	2
17	5	6	-1	5
18	5	6	0	7

The following statements fit a logit model with random effects for this MaxDiff choice experiment by specifying CHOICETYPE=MAXDIFF:

```
proc bchoice data=Sydney seed=1 nthreads=8 nmc=20000 thin=2 plots=none;
   class Index (ref=last) RespID Set;
   model Choice =  / choiceset=(RespID Set) choicetype=maxdiff;
   random Index / subject=RespID remean type=un;
run;
```

The NMC= option specifies the number of posterior simulation iterations after burn-in, and the THIN=2 option keeps one of every two samples for analysis. PLOTS=NONE is specified to suppress producing the plots to save time. The REMEAN option in the RANDOM statement requests estimation of the mean of the random effects. No fixed effects are specified in the MODEL statement.

The "Model Information" table in Output 30.8.2 shows that the type of choice response is MaxDiff.

Output 30.8.2 Model Information

The BCHOICE Procedure

Model Information	
Data Set	WORK.SYDNEY
Response Variable	Choice
Type of Model	Logit
Type of Choice Response	MaxDiff
Fixed Effects Included	No
Random Effects Included	Yes
Sampling Algorithm	Gamerman Metropolis
Burn-In Size	500
Simulation Size	20000
Thinning	2
Random Number Seed	1
Number of Threads	8

As shown in the "Choice Sets Summary" table in Output 30.8.3, there are a total of 2,400 choice sets (200 respondents, each with 12 choice sets), all of which display the same pattern: each has a total of three alternatives, one of which was chosen as the highest priority (best), another one was chosen as the lowest priority (worst), and the remaining one was not chosen.

Output 30.8.3 Choice Sets Summary

Choice Sets Summary for MaxDiff Data					
Pattern	Choice Sets	Total Alternatives	Chosen Best	Chosen Worst	Not Chosen
1	2400	3	1	1	1

Summary statistics for the mean of the random effects $\bar{\boldsymbol{\gamma}}$ (labeled as "REMEAN") and the covariance of the random effects $\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$ (labeled as "RECOV") are shown in Output 30.8.4. On average, the second improvement alternative (Index=2, improved peak rail capacity) has the largest part-worth value of 3.85, and the third improvement alternative (Index=3, more frequent bus service on major routes) has the next-largest part-worth

value. The part-worth value of the last alternative (Index=9) is structural 0. The ranking for the nine improvements from the highest priority to the lowest is 2, 3, 1, 5, 6, 9, 7, 4, and 8. It seems that in general people are most concerned about transportation on major routes during busy hours.

Output 30.8.4 PROC BCHOICE Posterior Summary Statistics

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
REMean Index 1	10000	1.4291	0.3042	0.8229	2.0192
REMean Index 2	10000	3.8554	0.3467	3.1852	4.5356
REMean Index 3	10000	3.0110	0.3108	2.4244	3.6337
REMean Index 4	10000	-0.8314	0.3217	-1.4560	-0.1999
REMean Index 5	10000	0.8937	0.3063	0.2722	1.4739
REMean Index 6	10000	0.7749	0.3183	0.1597	1.4073
REMean Index 7	10000	-0.5741	0.3278	-1.2157	0.0646
REMean Index 8	10000	-1.5394	0.3073	-2.1602	-0.9580
RECov Index 1, Index 1	10000	14.8555	2.1808	10.6273	19.1673
RECov Index 2, Index 1	10000	12.8307	2.1035	8.9311	17.1101
RECov Index 2, Index 2	10000	17.2483	2.6550	12.4063	22.6886
RECov Index 3, Index 1	10000	11.4254	1.9311	7.7419	15.2444
RECov Index 3, Index 2	10000	13.2266	2.1780	9.0158	17.4365
RECov Index 3, Index 3	10000	14.2821	2.1736	10.2170	18.6448
RECov Index 4, Index 1	10000	11.4926	1.9608	7.7660	15.3728
RECov Index 4, Index 2	10000	9.9519	2.0063	6.0542	13.8420
RECov Index 4, Index 3	10000	10.5877	1.9189	6.8862	14.3772
RECov Index 4, Index 4	10000	16.6336	2.4546	12.0061	21.5206
RECov Index 5, Index 1	10000	11.3054	1.9592	7.6339	15.2169
RECov Index 5, Index 2	10000	11.6876	2.0882	7.9589	16.0430
RECov Index 5, Index 3	10000	10.8130	1.9255	7.3641	14.8192
RECov Index 5, Index 4	10000	10.6285	1.9907	7.0367	14.7534
RECov Index 5, Index 5	10000	15.2144	2.2827	10.9980	19.7565
RECov Index 6, Index 1	10000	11.3246	1.9902	7.4668	15.1618
RECov Index 6, Index 2	10000	11.7650	2.1211	7.8721	16.0694
RECov Index 6, Index 3	10000	10.5438	1.9467	6.9584	14.4123
RECov Index 6, Index 4	10000	10.9368	2.0576	7.2383	15.1956
RECov Index 6, Index 5	10000	15.3265	2.2869	10.9703	19.7437
RECov Index 6, Index 6	10000	16.8353	2.4481	12.3717	21.7751
RECov Index 7, Index 1	10000	13.0307	2.0579	9.0988	17.1055
RECov Index 7, Index 2	10000	12.4041	2.1295	8.4571	16.6625
RECov Index 7, Index 3	10000	11.0318	1.9352	7.4545	14.9278
RECov Index 7, Index 4	10000	11.3357	2.0386	7.5180	15.4328
RECov Index 7, Index 5	10000	13.3085	2.1345	9.4358	17.5905
RECov Index 7, Index 6	10000	13.2184	2.1730	9.2114	17.5802
RECov Index 7, Index 7	10000	18.0400	2.5687	13.1280	23.0490
RECov Index 8, Index 1	10000	7.2473	1.6590	4.1723	10.5612
RECov Index 8, Index 2	10000	8.1009	1.8005	4.7421	11.6407
RECov Index 8, Index 3	10000	8.2427	1.7017	5.1639	11.6972
RECov Index 8, Index 4	10000	9.2959	1.8450	5.8923	12.9848
RECov Index 8, Index 5	10000	7.9252	1.7191	4.6876	11.2987
RECov Index 8, Index 6	10000	7.8835	1.7567	4.6412	11.3654
RECov Index 8, Index 7	10000	7.5465	1.7841	4.2349	11.1415
RECov Index 8, Index 8	10000	14.3947	2.2484	10.2301	18.7904

The covariance matrix ($\boldsymbol{\Omega}_{\boldsymbol{\gamma}}$) characterizes the extent of heterogeneity among individuals. Large diagonal elements of the covariance matrix indicate substantial heterogeneity in part-worths. Off-diagonal elements indicate patterns in the evaluation of alternatives. Most of the diagonal elements of the matrix are large, indicating that there is a large difference among people in the priorities they would set for future improvements in public transportation.

Example 30.9: Allocation Choice Example

(View the complete code for this example (bchcex9.sas) in the example repository.)

Consider an example of allocation choice data in the MAMR study.[2] In this study, researchers aimed to learn about oncologists' preferences on different scenarios of pharmaceutical therapies and benefits of these therapies in relation to the survival of patients with metastatic breast cancer.

```
data MAMR;
   input ID Set None TrtScheme RespRate Time Status TrtLine Risk Cost Alloc @@;
   datalines;
1     1     1     3     3     1     2     1     6     1     0.2
1     1     1     5     1     1     1     3     4     2     0
1     1     1     7     2     2     2     3     6     2     0
1     1     0     0     0     0     0     0     0     0     0.8
1     2     1     1     1     2     2     1     1     1     0.12
1     2     1     3     3     1     1     2     2     1     0.4
1     2     1     5     1     2     1     3     4     3     0
1     2     0     0     0     0     0     0     0     0     0.48
1     3     1     4     3     1     2     1     4     4     0.08

   ... more lines ...

146   10    0     0     0     0     0     0     0     0     0
146   11    1     1     2     1     1     3     6     1     0.34
146   11    1     2     2     2     1     3     3     1     0.28
146   11    1     1     2     2     2     1     1     1     0.38
146   11    0     0     0     0     0     0     0     0     0
;

proc print data=MAMR(obs=20);
run;
```

The data for the first several choice sets are shown in Output 30.9.1.

[2]The data were provided by Hernan Talledo, Ph.D., associate consultant at Datum International, Lima, Peru, and advanced market research professor at Universidad San Ignacio de Loyola and University of the Pacific in Lima; and Thomas C. Eagle, Ph.D, president of Eagle Analytics of California, Inc.

Output 30.9.1 Data for the First Several Allocation Choice Sets

Obs	ID	Set	None	TrtScheme	RespRate	Time	Status	TrtLine	Risk	Cost	Alloc
1	1	1	1	3	3	1	2	1	6	1	0.20
2	1	1	1	5	1	1	1	3	4	2	0.00
3	1	1	1	7	2	2	2	3	6	2	0.00
4	1	1	0	0	0	0	0	0	0	0	0.80
5	1	2	1	1	1	2	2	1	1	1	0.12
6	1	2	1	3	3	1	1	2	2	1	0.40
7	1	2	1	5	1	2	1	3	4	3	0.00
8	1	2	0	0	0	0	0	0	0	0	0.48
9	1	3	1	4	3	1	2	1	4	4	0.08
10	1	3	1	4	3	2	2	1	5	3	0.20
11	1	3	1	1	1	2	2	1	6	1	0.40
12	1	3	0	0	0	0	0	0	0	0	0.32
13	1	4	1	1	1	2	1	1	6	1	0.60
14	1	4	1	2	2	1	2	2	6	1	0.20
15	1	4	1	3	3	2	1	3	6	1	0.04
16	1	4	0	0	0	0	0	0	0	0	0.16
17	1	5	1	3	2	2	1	1	6	1	0.60
18	1	5	1	4	3	1	2	2	4	2	0.00
19	1	5	1	5	2	1	1	1	4	3	0.20
20	1	5	0	0	0	0	0	0	0	0	0.20

The variable ID represents the IDs for the 133 oncologists in the study. Each oncologist was presented with 11 choice tasks, each consisting of four scenarios of therapies (including the NONE option), and was asked to state the number of times he or she would pick each alternative. These numbers of times were then transformed into percentages, which sum to 100% within each choice task. The variable Alloc is the response variable that gives a percentage of preference for each alternative, and the sum of percentages in each choice set is 100%. Table 30.13 lists all the attributes and their levels.

Table 30.13 Attributes in the MAMR Study

Attribute	Label	Level	Description
TrtScheme	Treatment scheme	0	No treatment
		1	Taxotere + Adriamicina + SG 22.4 months + SLP 9.3 months
		2	Taxotere + Xeloda + SG 14.7 months + SLP 6.2 months
		3	Taxotere + Gemcitabina + SG 18.6 months + SLP 6.1 months
		4	Taxol + Avastin + SG 26.7 months + SLP 11.8 months
		5	Taxotere + Avastin + SG 83% a year + SLP 8.8 months
		6	Taxol weekly 90mg/m2 + SLP 6.2 months
		7	Taxotere 100mg/m2 + SG 15 months + SLP 6.5 months
RespRate	Response rate	1	40%
		2	50%
		3	60%
TrtTime	Treatment time	1	6 months
		2	Until cure

Table 30.13 *continued*

Attribute	Label	Level	Description
Status	Status	1	0–1 (better)
		2	>=2
TrtLine	Treatment lines	1	First line
		2	Second line
		3	Third line
Risk	Risk factors	1	Cardiovascular disease
		2	Hepatopathy moderate to severe
		3	Renal failure
		4	Thromboembolism history
		5	Arterial Hypertension (HTA)
		6	No risk
Cost	Cost per month	1	1,000
		2	3,000
		3	5,000
		4	7,000

The following statements fit a logit model with random effects for these allocation choice data by specifying CHOICETYPE=ALLOCATION:

```
proc bchoice data=MAMR seed=124 nthreads=8 nmc=20000 nthin=2 plots=none;
   class  TrtScheme(ref=first) RespRate Time Status TrtLine Risk Cost ID Set
          / ref=last;
   model  Alloc = / choiceset=(ID Set) choicetype=allocation;
   random TrtScheme RespRate Time Status TrtLine Risk Cost
          / subject=ID remean;
run;
```

The NMC= option requests to run the MCMC chain for 20,000 iterations, and the THIN=2 option keeps one of every two samples for analysis. PLOTS=NONE is specified, so the plots will not be produced to save time. The REMEAN option in the RANDOM statement requests estimation of the mean of the random effects.

When CHOICETYPE=ALLOCATION, you can specify a positive integer in the WEIGHT= suboption to indicate how many choice sets the allocated percentages should apply to. For example, if the respondent were asked to imagine the next 20 occasions and to allocate a percentage of those 20 occasions for each alternative, you would specify WEIGHT=20. By default, WEIGHT=10. In this example, the default (WEIGHT=10) is close to the nature of the original data collection.

The “Model Information” table, as shown in Output 30.9.2, shows that the type of choice response is allocation.

Output 30.9.2 Model Information

The BCHOICE Procedure

Model Information	
Data Set	WORK.MAMR
Response Variable	Alloc
Type of Model	Logit
Type of Choice Response	Allocation
Fixed Effects Included	No
Random Effects Included	Yes
Sampling Algorithm	Gamerman Metropolis
Burn-In Size	500
Simulation Size	20000
Thinning	2
Random Number Seed	124
Number of Threads	8

As shown in the "Choice Sets Summary" table in Output 30.9.3, there are a total of 1,463 choice sets, all of which display the same pattern: they have a total of four alternatives, and the sum of percentages is 100%. If the sum of percentages within one choice set is not equal to 100%, the choice set is invalid. PROC BCHOICE deletes all invalid choice sets and produces a warning message.

Output 30.9.3 Choice Sets Summary

Choice Sets Summary for Allocation Data				
Pattern	Choice Sets	Total Alternatives	Percent Sum(%)	Not Chosen(%)
1	1463	4	100	0

As you see in Output 30.9.4, TrtScheme, Status, TrtLine, and Risk have impact on the choices. It seems that treatments (TrtScheme) with shorter periods of SG and SLP follow-up are more preferred than those with longer follow-up times. The oncologists in the study tend to choose the therapies that would have the best performance status (Status); the part-worth for level 1 (better) of Status is positive compared to the referent level 2 of Status, which is set at 0. More important, risk factors (Risk) have a large influence on the choice of various treatments: the estimates of the part-worths for all risk factors are all negative and have relatively large values, reflecting that doctors try to avoid them as much as possible, especially treatments with serious side effects. One surprising result is that therapy cost does not play a role in the decision-making process. But when you realize that the respondents are doctors, not their patients, and that the study deals with metastatic cancer, it might not be hard to figure out why.

Output 30.9.4 PROC BCHOICE Posterior Summary Statistics

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
REMean TrtScheme 1	10000	-0.3178	0.1647	-0.6514	-0.00866
REMean TrtScheme 2	10000	-0.5021	0.1507	-0.8010	-0.2145
REMean TrtScheme 3	10000	-0.3978	0.1476	-0.6803	-0.1057
REMean TrtScheme 4	10000	0.1785	0.1480	-0.1180	0.4624
REMean TrtScheme 5	10000	-0.4356	0.1499	-0.7329	-0.1380
REMean TrtScheme 6	10000	0.1791	0.1746	-0.1612	0.5265
REMean TrtScheme 7	10000	-0.0555	0.1468	-0.3571	0.2126
REMean RespRate 1	10000	-0.0493	0.0781	-0.1982	0.1076
REMean RespRate 2	10000	-0.0348	0.0771	-0.1877	0.1147
REMean Time 1	10000	-0.0371	0.0717	-0.1765	0.1056
REMean Status 1	10000	0.1893	0.0741	0.0456	0.3344
REMean TrtLine 1	10000	0.4689	0.0835	0.3045	0.6285
REMean TrtLine 2	10000	0.2883	0.0806	0.1306	0.4465
REMean Risk 1	10000	-1.2123	0.1782	-1.5556	-0.8573
REMean Risk 2	10000	-0.8808	0.1398	-1.1627	-0.6108
REMean Risk 3	10000	-0.9594	0.1601	-1.2731	-0.6476
REMean Risk 4	10000	-1.4509	0.1794	-1.8124	-1.1090
REMean Risk 5	10000	-1.4393	0.1732	-1.7814	-1.1135
REMean Cost 1	10000	0.1649	0.1300	-0.0943	0.4162
REMean Cost 2	10000	-0.0330	0.1061	-0.2371	0.1751
REMean Cost 3	10000	-0.0936	0.1174	-0.3364	0.1253
REVar TrtScheme 1	10000	1.7718	0.3062	1.2306	2.4020
REVar TrtScheme 2	10000	1.0636	0.2037	0.6920	1.4722
REVar TrtScheme 3	10000	1.2343	0.2166	0.8439	1.6742
REVar TrtScheme 4	10000	1.9933	0.3025	1.4457	2.6027
REVar TrtScheme 5	10000	1.9061	0.3168	1.2955	2.5264
REVar TrtScheme 6	10000	1.3845	0.2915	0.8668	1.9748
REVar TrtScheme 7	10000	0.9070	0.1750	0.5886	1.2585
REVar RespRate 1	10000	0.4506	0.0673	0.3290	0.5842
REVar RespRate 2	10000	0.4533	0.0676	0.3295	0.5859
REVar Time 1	10000	0.4296	0.0625	0.3097	0.5496
REVar Status 1	10000	0.4670	0.0692	0.3404	0.6071
REVar TrtLine 1	10000	0.5918	0.0910	0.4225	0.7684
REVar TrtLine 2	10000	0.4901	0.0748	0.3570	0.6436
REVar Risk 1	10000	1.3516	0.3257	0.7761	2.0061
REVar Risk 2	10000	1.2343	0.2634	0.7722	1.7622
REVar Risk 3	10000	1.1511	0.2694	0.6700	1.6896
REVar Risk 4	10000	2.6897	0.5148	1.7562	3.7233
REVar Risk 5	10000	2.7399	0.4715	1.8703	3.6810
REVar Cost 1	10000	1.0116	0.1675	0.7030	1.3472
REVar Cost 2	10000	0.7321	0.1159	0.5201	0.9731
REVar Cost 3	10000	0.9321	0.1783	0.6065	1.2881

References

Albert, J. H., and Chib, S. (1993). "Bayesian Analysis of Binary and Polychotomous Response Data." *Journal of the American Statistical Association* 88:669–679.

Allenby, G. M. (1997). "An Introduction to Hierarchical Bayesian Modeling." American Marketing Association, Advanced Research Techniques Forum, Tutorial Notes.

Allenby, G. M., and Lenk, P. (1994). "Modeling Household Purchase Behavior with Logistic Normal Regression." *Journal of the American Statistical Association* 89:1218–1231.

Allenby, G. M., and Rossi, P. E. (1991). "Quality Perceptions and Asymmetric Switching between Brands." *Marketing Science* 10:185–205.

Allenby, G. M., and Rossi, P. E. (1999). "Marketing Models of Consumer Heterogeneity." *Journal of Econometrics* 89:57–78.

Bhat, C. R. (1995). "A Heteroscedastic Extreme Value Model of Intercity Travel Mode Choice." *Transportation Research* 29:471–483.

Eilers, P. H. C., and Marx, B. D. (1996). "Flexible Smoothing with *B*-Splines and Penalties." *Statistical Science* 11:89–121. With discussion.

Finn, A., and Louviere, J. J. (1992). "Determining the Appropriate Response to Evidence of Public Concern: The Case of Food Safety." *Journal of Public Policy and Marketing* 11:12–25.

Gamerman, D. (1997). "Sampling from the Posterior Distribution in Generalized Linear Models." *Statistics and Computing* 7:57–68.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. 2nd ed. London: Chapman & Hall.

Herriges, J. A., and Kling, C. L. (1996). "Testing the Consistency of Nested Logit Models with Utility Maximization." *Economics Letters* 50:33–39.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998). "Markov Chain Monte Carlo in Practice: A Roundtable Discussion." *American Statistician* 52:93–100.

Kling, C. L., and Herriges, J. A. (1995). "An Empirical Investigation of the Consistency of Nested Logit Models with Utility Maximization." *American Journal of Agricultural Economics* 77:875–884.

Kuhfeld, W. F. (2010). *Marketing Research Methods in SAS*. Technical report, SAS Institute Inc., Cary, NC. `http://support.sas.com/resources/papers/tnote/tnote_marketresearch.html`.

Lahiri, K., and Gao, J. (2002). "Bayesian Analysis of Nested Logit Model by Markov Chain Monte Carlo." *Journal of Econometrics* 11:103–133.

Louviere, J. J. (1991). "Best-Worst Scaling: A Model for the Largest Difference Judgements." Working paper, University of Alberta, Edmonton.

Magidson, J., and Vermunt, J. K. (2015). "Analyzing MaxDiff Data with Scale Factors: Applications of Latent GOLD Basic + Choice 5.1." Latent GOLD Tutorial 8A, Statistical Innovations Inc., Belmont, MA. http://www.statisticalinnovations.com/wp-content/uploads/LGChoice_tutorial_8A.pdf.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

McCulloch, R., and Rossi, P. E. (1994). "An Exact Likelihood Analysis of the Multinomial Probit Model." *Journal of Econometrics* 64:207–240.

McFadden, D. (1974). "Conditional Logit Analysis of Qualitative Choice Behavior." In *Frontiers in Econometrics*, edited by P. Zarembka, 105–142. New York: Academic Press.

McFadden, D. (1978). "Modelling the Choice of Residential Location." In *Spatial Interaction Theory and Planning Models*, edited by A. Karlqvist, L. Lundqvist, F. Snickars, and J. Weibull, 75–96. Amsterdam: North-Holland.

McFadden, D. (2001). "Economic Choices." *American Economic Review* 91:351–378.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). "Weak Convergence and Optimal Scaling of Random Walk Metropolis Algorithms." *Annals of Applied Probability* 7:110–120.

Roberts, G. O., and Rosenthal, J. S. (2001). "Optimal Scaling for Various Metropolis-Hastings Algorithms." *Statistical Science* 16:351–367.

Rossi, P. E. (2012). "Bayesm Package: Bayesian Inference for Marketing/Micro-econometrics." http://www.perossi.org/home/bsm-1.

Rossi, P. E. (2013). Personal communication.

Rossi, P. E., Allenby, G. M., and McCulloch, R. (2005). *Bayesian Statistics and Marketing*. Chichester, UK: John Wiley & Sons.

Rossi, P. E., McCulloch, R., and Allenby, G. M. (1996). "The Value of Purchase History Data in Target Marketing." *Marketing Science* 15:321–340.

Schervish, M. J. (1995). *Theory of Statistics*. New York: Springer-Verlag.

Train, K. E. (2009). *Discrete Choice Methods with Simulation*. 2nd ed. Cambridge: Cambridge University Press.

Train, K. E., McFadden, D. L., and Ben-Akiva, M. (1987). "The Demand for Local Telephone Service: A Fully Discrete Model of Residential Calling Patterns and Service Choice." *RAND Journal of Economics* 18:109–123.

Chapter 31
The BGLIMM Procedure

Contents

Overview: BGLIMM Procedure . . . **1277**
- Basic Features . . . 1277
- Notation for the Generalized Linear Mixed Model . . . 1278
- PROC BGLIMM Compared with Other SAS Procedures . . . 1280

Getting Started: BGLIMM Procedure . . . **1281**
- Logistic Regression with Random Intercepts . . . 1281

Syntax: BGLIMM Procedure . . . **1288**
- PROC BGLIMM Statement . . . 1289
- BY Statement . . . 1298
- CLASS Statement . . . 1298
- ESTIMATE Statement . . . 1301
- FREQ Statement . . . 1304
- LSMEANS Statement . . . 1304
- MODEL Statement . . . 1305
- PREDDIST Statement . . . 1314
- RANDOM Statement . . . 1316
- REPEATED Statement . . . 1327
- WEIGHT Statement . . . 1331

Details: BGLIMM Procedure . . . **1332**
- Generalized Linear Mixed Models . . . 1332
  - Response Probability Distributions . . . 1333
  - Likelihood . . . 1336
  - Scale and Dispersion Parameters . . . 1338
- How PROC BGLIMM Works . . . 1339
  - GLM Parameters . . . 1339
  - GLMM with Random Effects . . . 1340
  - Models with Missing Values . . . 1340
- Sampling Methods . . . 1341
  - Conjugate Sampling . . . 1341
  - Gamerman Algorithm . . . 1341
  - Hamiltonian Monte Carlo Sampler . . . 1343
  - Slice Sampler . . . 1344
- Prior Distributions . . . 1344
  - Prior for the Fixed-Effects Coefficients . . . 1344
  - Prior for the Random-Effects Coefficients . . . 1345
  - Prior for the G-Side Covariance . . . 1345

Prior for the Scale Parameter 1347
Prior for the R-Side Covariance 1347
Treatment of Subjects in the RANDOM Statement 1348
Initial Values of the Markov Chains 1349
Missing Data 1349
Multinomial Models 1350
Autocall Macros for Postprocessing 1351
Regenerating Diagnostics Plots 1353
Displayed Output 1354
Model- and Data-Related ODS Tables 1354
Sampling-Related ODS Tables 1355
ODS Tables Related to Posterior Statistics 1355
ODS Tables Related to Convergence Diagnostics 1357
ODS Table Names 1358
ODS Graphics 1360
Examples: BGLIMM Procedure **1361**
Example 31.1: Normal Regression with Repeated Measurements 1361
Example 31.2: Mating Experiment with Crossed Random Effects 1368
Example 31.3: Poisson Regression with Random Effects 1377
Example 31.4: Repeated Growth Measurements with Group Difference 1383
Example 31.5: Multinomial Distribution with Cumulative Links 1392
Example 31.6: Multinomial Generalized Logit Model for Nominal Response 1398
Example 31.7: Bayesian Networking Meta-analysis 1402
Arm-Based NMA for Binomial Outcomes 1403
Arm-Based NMA for Continuous Outcomes 1406
Example 31.8: Power Prior 1410
References **1419**

Overview: BGLIMM Procedure

The BGLIMM procedure is a high-performance, sampling-based procedure that provides Bayesian inference for generalized linear mixed models (GLMMs).

GLMMs are hierarchical models that combine a generalized linear model with normally distributed random effects. Conditional on the random effects, data have distributions in the exponential family (binary, binomial, Poisson, normal, gamma, and so on). GLMMs are widely used in practice and are especially useful in applications where the data consist of collections of units and are hierarchically structured. Variations of GLMMs offer modeling flexibility that enables you to capture the complex nature of real-world data.

In a classical (frequentist) approach, the fixed-effects parameters are considered fixed with an unknown mean and are the primary focus of inference. The random effects are treated as unobserved latent variables. Estimation is achieved by maximizing the marginal likelihood of the fixed-effects parameters while integrating out the random effects (Davidian and Giltinan 1995; Vonesh, Chinchilli, and Pu 1996). Typically, asymptotic normality is assumed in inference.

In contrast, the Bayesian approach estimates the joint posterior distribution of all parameters in a model, including all fixed- and random-effects parameters. The Monte Carlo method numerically integrates out the random effects and propagates the uncertainties to the marginal posterior of the fixed-effects parameters. PROC BGLIMM uses efficient Markov chain Monte Carlo (MCMC) sampling tools to estimate the posterior marginal distributions and use them for further inference.

For a short introduction to Bayesian analysis and related basic concepts, see Chapter 8, "Introduction to Bayesian Analysis Procedures." For discussions of the relative advantages and disadvantages of the Bayesian paradigm, see the section "Bayesian Analysis: Advantages and Disadvantages" on page 154 in Chapter 8, "Introduction to Bayesian Analysis Procedures." For a guide to Bayesian textbooks of varying degrees of difficulty, see the section "A Bayesian Reading List" on page 178 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

PROC BGLIMM uses syntax similar to that of PROC MIXED and PROC GLIMMIX in specifying a GLMM. You use the MODEL statement to specify the distribution and link function, the RANDOM statement to specify the random effects, the CLASS statement to specify categorical variables, the REPEATED statement to specify the correlation of longitudinal responses, and the ESTIMATE statements for inferences. PROC BGLIMM draws samples from the target distributions, computes summary and diagnostic statistics, and saves the posterior samples in an output data set that you can use for further analysis.

Basic Features

The basic features of PROC BGLIMM include the following:

- GLMMs with univariate or multivariate dimensional random effects
- nested or non-nested hierarchical models
- repeated measures models (balanced or unbalanced data) with normal data
- suite of covariance structures for random effects and residuals, including variance components, compound symmetry, unstructured, AR(1), Toeplitz, autoregressive, and many more

- multinomial models for ordinal and nominal outcomes
- built-in prior distributions for regression coefficients and covariance parameters
- the ability to model heterogeneity in covariance structures
- the ability to produce estimates and credible intervals for estimable linear combination of effects
- support for missing completely at random (MCAR) and missing at random (MAR) approaches in modeling missing data
- multithreading of optimal sampling algorithms for fast performance
- the ability to save posterior samples to an output data set for use in further inferences

PROC BGLIMM uses the Output Delivery System (ODS) to display and control the output. ODS enables you to convert any of the output from PROC BGLIMM to a SAS data set. For more information, see the section "ODS Table Names" on page 1358.

PROC BGLIMM uses ODS Graphics to create graphs as part of its output. For specific information about the statistical graphics available with the BGLIMM procedure, see the PLOTS options in PROC BGLIMM.

Notation for the Generalized Linear Mixed Model

This section introduces the mathematical notation that the chapter uses to describe the generalized linear mixed model. For a description of the statistical details and sampling algorithms, see the section "Details: BGLIMM Procedure" on page 1332.

First consider the simple normal linear model. The quantity of primary interest, y_i, is called the response or outcome variable for the ith individual. The variable $\mathbf{x}_i$ is the $1 \times p$ covariate vector for the fixed effects. The distribution of y_i given $\mathbf{x}_i$ is normal with a mean that is a linear function of $\mathbf{x}_i$,

$$y_i = \mathbf{x}_i \boldsymbol{\beta} + \epsilon_i, \quad i = 1, \ldots, I$$
$$\epsilon_i \sim N(\mathbf{0}, \sigma^2)$$

where $\boldsymbol{\beta}$ is a $p \times 1$ vector of regression coefficients (also known as fixed effects) and ϵ_i is the noise with a variance σ^2.

The normal linear model can be expanded to include random effects, and the model becomes a normal linear mixed model,

$$y_i = \mathbf{x}_i \boldsymbol{\beta} + \mathbf{z}_i \boldsymbol{\gamma}_i + \epsilon_i$$
$$\boldsymbol{\gamma}_i \sim N(\mathbf{0}, \mathbf{G}_i)$$
$$\epsilon_i \sim N(0, \sigma^2)$$

where $\boldsymbol{\gamma}_i$ is a $q \times 1$ vector of random effects, $\mathbf{z}_i$ is a $1 \times q$ matrix of covariates for the $\boldsymbol{\gamma}_i$, and $\mathbf{G}_i$ is the covariance matrix of the random effects $\boldsymbol{\gamma}_i$ ($\mathbf{G}$ is a block diagonal matrix where each block is $\mathbf{G}_i$).

When an individual i has n_i repeated measurements, the random-effects model for outcome vector $\mathbf{y}_i$ is given by

$$\mathbf{y}_i = \mathbf{X}_i \boldsymbol{\beta} + \mathbf{Z}_i \boldsymbol{\gamma}_i + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, I$$

where $\mathbf{y}_i$ is $n_i \times 1$, $\mathbf{X}_i$ is an $n_i \times p$ matrix of fixed covariates, $\boldsymbol{\beta}$ is a $p \times 1$ vector of regression coefficients (also known as fixed effects), $\boldsymbol{\gamma}_i$ is a $q \times 1$ vector of random effects, $\mathbf{Z}_i$ is an $n_i \times q$ matrix of covariates for the $\boldsymbol{\gamma}_i$, and $\boldsymbol{\epsilon}_i$ is an $n_i \times 1$ vector of random errors.

It is further assumed that

$$\boldsymbol{\gamma}_i \sim N(\mathbf{0}, \mathbf{G}_i)$$
$$\boldsymbol{\epsilon}_i \sim N(\mathbf{0}, \mathbf{R}_i)$$

where $\mathbf{G}_i$ is the covariance matrix of $\boldsymbol{\gamma}_i$ ($\mathbf{G}$ is a block diagonal matrix where each block is $\mathbf{G}_i$) and $\mathbf{R}_i$ is the covariance matrix of the residual errors for the ith subject ($\mathbf{R}$ is a block diagonal matrix where each block is $\mathbf{R}_i$).

There are cases where the relationship between the design matrix ($\mathbf{X}$ and $\mathbf{Z}$) and the expectation of the response is not linear, or where the distribution for the response is far from normal, even after transformation of the data. The class of generalized linear mixed models unifies the approaches that you need in order to analyze data in those cases. Let $\mathbf{y}$ be the collection of all $\mathbf{y}_i$; and let $\mathbf{X}$ and $\mathbf{Z}$ be the collection of all $\mathbf{X}_i$ and $\mathbf{Z}_i$, respectively. A generalized linear mixed model consists of the following:

- the linear predictor $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}$
- the link function $g(\cdot)$ that relates the linear predictor to the mean of the outcome via a monotone link function,

  $$\mathrm{E}[Y|\boldsymbol{\beta}, \boldsymbol{\gamma}] = g^{-1}(\boldsymbol{\eta}) = g^{-1}(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma})$$

  where $g(\cdot)$ is a differentiable monotone link function and $g^{-1}(\cdot)$ is its inverse
- a response distribution in the exponential family of distributions. The distribution can also depend on a scale parameter, ϕ.

The conditional distribution of the response variable, given $\boldsymbol{\gamma}$, is a member of the exponential family of distributions, including the normal distribution. You specify the distribution by using the DIST= option in the MODEL statement and specify the link function $g(\cdot)$ by using the LINK= option.

The BGLIMM procedure distinguishes two types of covariance structure: the "G-side" and the "R-side." The G-side matrix is the covariance matrix of the random effects; the R-side matrix is the covariance matrix of the residuals. Models without G-side effects are also known as marginal (or population-averaged) models.

The columns of $\mathbf{X}$ are constructed from effects that are listed on the right side in the MODEL statement. Columns of $\mathbf{Z}$ and the G-side covariance matrix $\mathbf{G}$ are constructed from the RANDOM statement. The R-side covariance matrix $\mathbf{R}$ is constructed from the REPEATED statement, or from the RANDOM statement with the RESIDUAL option.

By default, the $\mathbf{R}$ matrix is the scaled identity matrix, $\mathbf{R} = \phi\mathbf{I}$. The scale parameter ϕ is set to 1 if the distribution does not have a scale parameter, such as in the case of the binary, binomial, Poisson, and exponential distributions.

For the normal distribution, for which you can specify various types of covariance structure for $\mathbf{R}$, use the REPEATED statement. For example, to specify that the Time effect for each patient is an R-side effect with a first-order autoregressive covariance structure, use the following statement:

```
repeated Time / type=ar(1) subject=Patient;
```

Unknown quantities subject to estimation are the fixed-effects parameter vector $\boldsymbol{\beta}$, the random-effects parameter $\boldsymbol{\gamma}$, and the covariance parameters that constitute all unknowns in $\mathbf{G}$ and $\mathbf{R}$.

PROC BGLIMM Compared with Other SAS Procedures

Both PROC BGLIMM and PROC GENMOD can fit Bayesian generalized linear models. The posterior estimates are similar but not exact because of the nature of simulation and the different algorithms that the procedures use.

PROC MIXED and PROC GLIMMIX are two mixed modeling procedures that are related to PROC BGLIMM. All three procedures share some common syntax. PROC BGLIMM provides Bayesian solutions to GLMMs, whereas PROC MIXED and PROC GLIMMIX provide classical statistics solutions to the linear and generalized linear mixed-effects models, respectively.

Frequentist estimation methods rely on maximizing the marginal likelihood function, and inferences are often based on asymptotic theorems. This reflects frequentist assumptions in statistical inference—namely, that parameters are fixed quantities and uncertainties are the result of sampling.

PROC BGLIMM provides Bayesian solutions, and it estimates, via simulation, the joint posterior distributions of all parameters (fixed-effects, random-effects, covariance), conditional on the data. This approach reflects assumptions in the Bayesian paradigm—namely, that parameters are random and you use probability statements to quantify uncertainty associated with these random variables.

In practice, Bayesian estimates from PROC BGLIMM that use noninformative prior distributions closely resemble those that are obtained from PROC MIXED in linear mixed-effects models, a result due to linear model theory and the normality assumption. You can expect larger discrepancies between PROC BGLIMM and PROC GLIMMIX results, especially in situations where the normal approximation to the marginal likelihood function is inaccurate.

In terms of run time, the sampling-based BGLIMM procedure could run slower than the mixed modeling procedures. The dimension of the regression problem can also hinder MCMC convergence. On the other hand, if a complex hierarchical model requires high-dimensional and computationally costly integral approximation methods in mixed procedures, using PROC BGLIMM can lead to better performance.

Model specifications in PROC BGLIMM are mostly identical to those in PROC MIXED and PROC GLIMMIX. You can expect the same construction of the design matrix, the covariance types, and the group structure in PROC BGLIMM. One important difference is in how the procedures handle wrongly specified models, such as singular or nearly singular design matrices. PROC BGLIMM requires the associated covariance matrix to be strictly positive definite; PROC MIXED and PROC GLIMMIX might rely on numerical methods (such as the general inverse) to bypass such difficulties in obtaining estimates.

PROC BGLIMM is similar to PROC MCMC in that both procedures use MCMC methods to estimate posterior distributions. PROC MCMC is a more general procedure that can handle a wider range of Bayesian statistical models. For example, PROC MCMC enables you to fit a model that includes random effects that are not normally distributed, which is something that PROC BGLIMM does not support. But because PROC MCMC relies on more detailed user inputs to construct a model, it uses generic sampling algorithms that are not tailored to specific models. On the other hand, PROC BGLIMM provides convenient access to Bayesian

analysis of complex mixed models, with improved performance that results from using optimal sampling algorithms.

Getting Started: BGLIMM Procedure

This "Getting Started" section presents an example of a random-effects logistic model.

Logistic Regression with Random Intercepts

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/bglmmgs1.sas.)

Researchers investigated the performance of two medical procedures in a multicenter study. They randomly selected 15 centers for inclusion. One of the study goals was to compare the occurrence of side effects from the procedures. In each center, n_{A} patients were randomly selected and assigned to treatment group A, and n_{B} patients were randomly assigned to treatment group B. The following DATA step creates the data set, MultiCenter, for the analysis:

```
data MultiCenter;
   input Center Group$ N SideEffect @@;
   datalines;
 1  A  32  14   1  B  33  18
 2  A  30   4   2  B  28   8
 3  A  23  14   3  B  24   9
 4  A  22   7   4  B  22  10
 5  A  20   6   5  B  21  12
 6  A  19   1   6  B  20   3
 7  A  17   2   7  B  17   6
 8  A  16   7   8  B  15   9
 9  A  13   1   9  B  14   5
10  A  13   3  10  B  13   1
11  A  11   1  11  B  12   2
12  A  10   1  12  B   9   0
13  A   9   2  13  B   9   6
14  A   8   1  14  B   8   1
15  A   7   1  15  B   8   0
;
```

The variable Group identifies the two procedures, N is the number of patients who received a given procedure at a particular center, and SideEffect is the number of patients who reported side effects. Side effects ratios, from different centers, are shown in Figure 31.1. Data from group A are represented by solid circles; data from group B are represented by clear circles.

Figure 31.1 MultiCenter Data

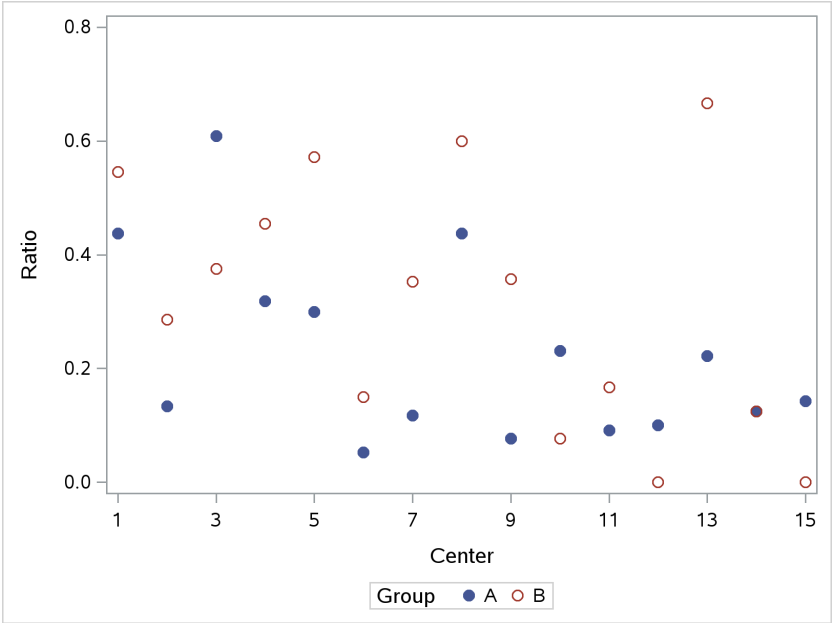

If y_{iA} and y_{iB} denote the number of patients at center i who report side effects for procedures A and B, respectively, then—for a given center—these are independent binomial random variables. To model the probability of having side effects from the two procedures, p_{iA} and p_{iB}, you need to account for the fixed group effect and the random selection of centers. One possibility is to assume a model that relates group and center effects linearly to the logit of the probabilities:

$$\log\left\{\frac{p_{iA}}{1-p_{iA}}\right\} = \beta_A + \gamma_i$$
$$\log\left\{\frac{p_{iB}}{1-p_{iB}}\right\} = \beta_B + \gamma_i$$

In this model, $\beta_A - \beta_B$ measures the difference in the logits of experiencing side effects, and the γ_i are independent random variables due to the random selection of centers. Observations from the same center receive the same adjustment, and these adjustments vary randomly from center to center, with variance $\mathrm{Var}[\gamma_i] = \sigma_c^2$.

Because p_{iA} is the conditional mean of the sample proportion, $\mathrm{E}[y_{iA}/n_{iA}|\gamma_i] = p_{iA}$, you can model the sample proportions as binomial ratios in a generalized linear mixed model. The following statements request this analysis under the assumption of normally distributed center effects with equal variance and a logit link function:

```
ods graphics on;
proc bglimm data=MultiCenter nmc=10000 thin=2 seed=976352
   plots=all;
   class Center Group;
   model SideEffect/N = Group / noint;
   random int / subject = Center;
run;
```

The ODS GRAPHICS ON statement invokes the ODS Graphics environment, and the PLOTS=ALL option displays the diagnostic plots—the trace, autocorrelation function, and kernel density plots of the posterior samples.

The PROC BGLIMM statement invokes the procedure, and the DATA= option specifies the input data set MultiCenter. The NMC= option specifies the number of posterior simulation iterations in the main simulation loop after burn-in (the default number of burn-in iterations is 500). The THIN= option controls the thinning of the Markov chain and specifies that one of every two samples be kept.[1] The SEED= option specifies a random number generator seed, which reproduces the results.

The CLASS statement treats the variables Center and Group as classification variables.

The MODEL statement specifies the response variable as a sample proportion by using the *events/trials* syntax. In terms of the previous formulas, SideEffect/N corresponds to $y_{i\mathrm{A}}/n_{i\mathrm{A}}$ for observations from group A and to $y_{i\mathrm{B}}/n_{i\mathrm{B}}$ for observations from group B. Note that because of the *events/trials* syntax, the BGLIMM procedure defaults to the binomial distribution, and that distribution's default link is the logit link. The MODEL statement specifies an independent fixed-effects variable (Group). An intercept is included in the fixed-effects model by default. You can remove it by using the NOINT option, so that you can obtain the estimates for both treatment groups A and B.

Along with the MODEL statement for fixed effects, the RANDOM statement is used for random effects. It specifies that the linear predictor contains an intercept term that randomly varies at the level of the Center effect. The SUBJECT=CENTER option in the RANDOM statement defines Center as a subject index for the random-effects grouping, so that each center has its own intercept. In other words, a random intercept is drawn separately and independently for each center in the study.

The "Model Information Table" in Figure 31.2 summarizes important information about the model that you fit and about aspects of the sampling technique. Information includes the input data set, response variable, likelihood distribution, link function, sampling algorithm, MCMC burn-in size, simulation size, thinning rate, and random number seed.

Figure 31.2 Model Information

The BGLIMM Procedure

Model Information	
Data Set	WORK.MULTICENTER
Response Variable	SideEffect
Distribution	Binomial
Link Function	Logit
Fixed Effects Included	Yes
Random Effects Included	Yes
Sampling Algorithm	Gamerman, Conjugate
Burn-In Size	500
Simulation Size	10000
Thinning	2
Random Number Seed	976352
Number of Threads	1

[1] Thinning is often used to reduce correlation among posterior sample draws. For more information about Markov chain sample size, burn-in, and thinning, see the section "Burn-In, Thinning, and Markov Chain Samples" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

PROC BGLIMM recognizes the variables SideEffect and N as the numerator and denominator in the *events/trials* syntax, respectively. The distribution—conditional on the random center effects—is binomial.

Figure 31.3 displays two tables. The "Class Level Information" table lists the levels of the variables that are specified in the CLASS statement and the ordering of the levels. The "Number of Observations" table displays the number of observations that are read and used in the analysis.

Figure 31.3 Class Level Information and Number of Observations

Class Level Information		
Class	**Levels**	**Values**
Center	15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Group	2	A B

Number of Observations	
Number of Observations Read	30
Number of Observations Used	30
Number of Events	155
Number of Trials	503

Two variables are listed in the CLASS statement. The Center variable has 15 levels, and the Group variable has 2 levels. Because the response is specified through the *events/trial* syntax, the "Number of Observations" table also contains the total number of events and trials that are used in the analysis.

In Figure 31.4, the "Posterior Summaries and Intervals" table lists the summary statistics (posterior means, standard deviations, and HPD intervals) for each parameter, the fixed coefficients ($\boldsymbol{\beta}$), and the variance of the random center intercepts (σ_c^2). Posterior summary statistics of random-effects parameters are not displayed by default. You can display them by using the MONITOR option in the RANDOM statement.

Figure 31.4 Posterior Summaries and Intervals

Posterior Summaries and Intervals					
Parameter	**N**	**Mean**	**Standard Deviation**	**95% HPD Interval**	
Group A	5000	-1.3895	0.3102	-2.0071	-0.7956
Group B	5000	-0.8839	0.2968	-1.4819	-0.3186
Random Var	5000	0.9184	0.4198	0.3024	1.7515

It is important to check for convergence of all model parameters, because chains that are not converged can lead to biased conclusions. PROC BGLIMM produces a number of convergence diagnostics. By default, it displays effective sample sizes (ESS; Kass et al. 1998) of each parameter (Figure 31.5). ESS is theoretically equivalent to the number of independent samples that are generated directly from a target distribution.

Figure 31.5 PROC BGLIMM Effective Sample Size

Effective Sample Sizes			
Parameter	**ESS**	**Autocorrelation Time**	**Efficiency**
Group A	637.5	7.8432	0.1275
Group B	649.1	7.7028	0.1298
Random Var	2213.4	2.2590	0.4427

.

PROC BGLIMM produces a number of graphs, shown in Figure 31.6, that also help your convergence diagnostic checks. The trace plots show the stability of the Markov chain over the simulation and a number of whiskers in the tail areas, indicating good mixing of the chain. The autocorrelation plots show a decrease in autocorrelations, and the kernel density plots estimate the posterior marginal distributions for each parameter.

Figure 31.6 PROC BGLIMM Diagnostic Plots

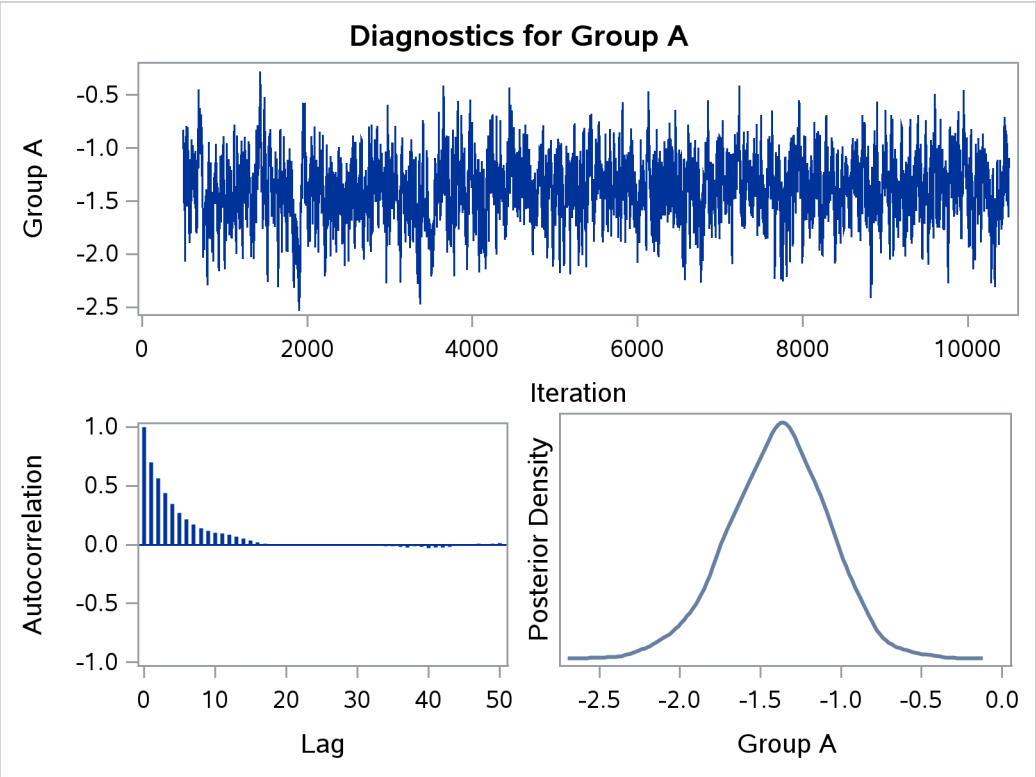

Figure 31.6 *continued*

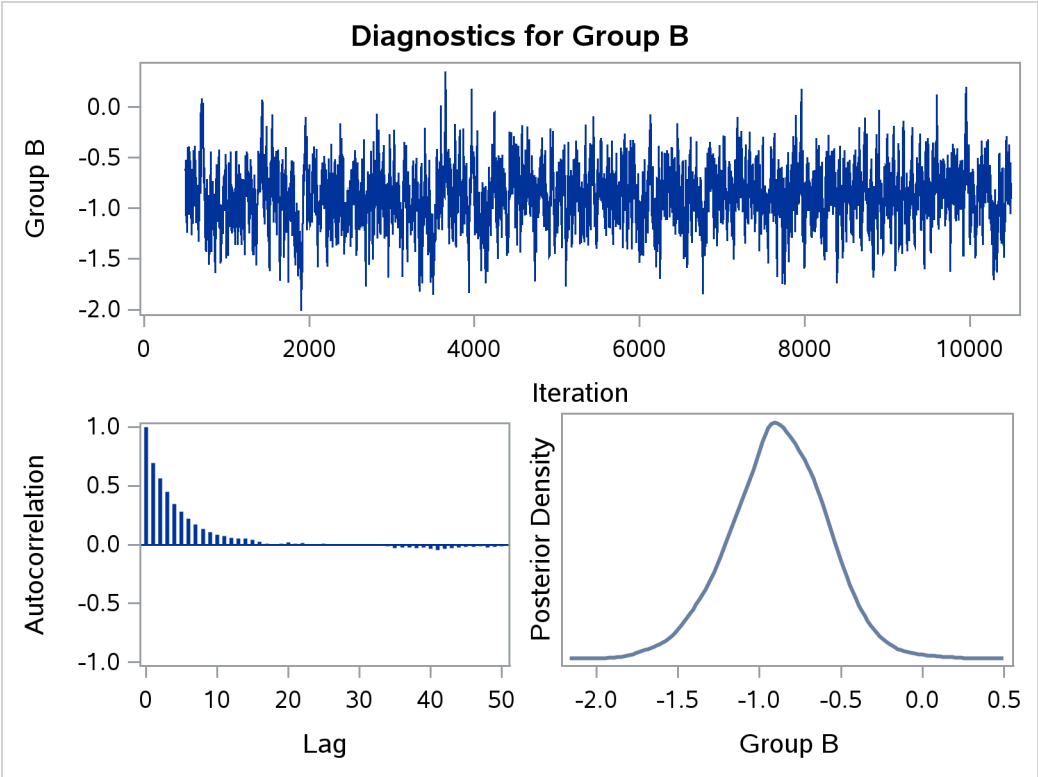

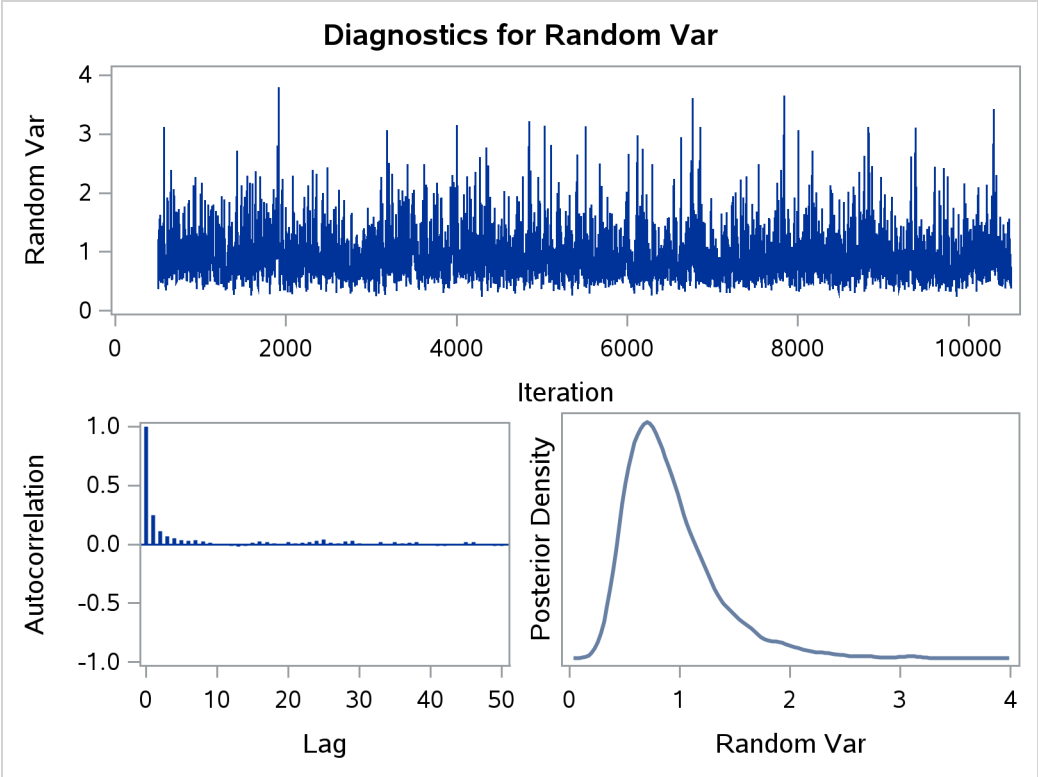

You can use the ESTIMATE statement as follows to compute the log of odds ratios between the two treatment

groups, A and B:

```
proc bglimm data=MultiCenter nmc=10000 thin=2 seed=976352
   outpost=CenterOut;
   class Center Group;
   model SideEffect/N = Group / noint;
   random int / subject=Center monitor;
   estimate "log OR" group 1 -1;
run;
```

The ESTIMATE statement computes $\beta_{\mathrm{A}} - \beta_{\mathrm{B}}$ by using every posterior draw of the parameters and saves the values in the OUTPOST= data set under the variable name Log_or.

The "Estimated Differences in the Logits" table in Figure 31.7 shows that the posterior mean of the log of odds ratio is around –0.5, with the 95% HPD interval all negative. This indicates less chance of developing side effects among patients who undergo procedure A than among those who undergo procedure B.

Figure 31.7 Estimated Differences in the Logits

The BGLIMM Procedure

Results from ESTIMATE Statements				
Label	Mean	Standard Deviation	95% HPD Interval	
log OR	-0.5056	0.2087	-0.9292	-0.1102

The ESTIMATE statement does not compute the difference in probabilities of side effects directly. You can use the posterior samples and compute the probability differences directly. The following DATA step computes the probability difference between the two groups and saves it to the variable pDiff:

```
data prob;
   set CenterOut;
   pDiff = logistic(group_a) - logistic(group_b);
run;
```

You can use the %SUMINT autocall macro to compute the posterior summary statistics of pDiff:

```
%sumint(data=prob, var=pDiff)

proc sgplot data=prob noautolegend;
   yaxis display=(nolabel noline noticks novalues);
   xaxis label="Difference of Probability in Treatment";
   density pDiff / type=kernel;
run;
```

The results are shown in Figure 31.8. The density plot of the posterior distribution is shown in Figure 31.9. There is a significant difference between the two groups.

Figure 31.8 Posterior Summary Statistics

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
pDiff	5000	-0.0920	0.0395	-0.1750	-0.0195

.

Figure 31.9 Posterior Density of the Difference in Probabilities

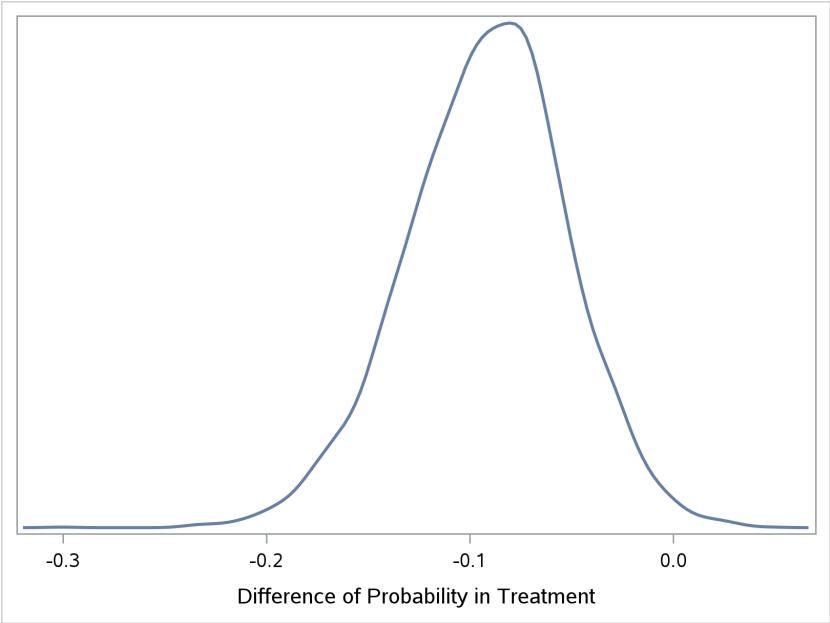

Syntax: BGLIMM Procedure

The following statements are available in the BGLIMM procedure. Items within angle brackets (< >) are optional.

PROC BGLIMM < *options* > **;**
  BY *variables* **;**
  CLASS *variable* < **(***options***)** >. . . < *variable* < **(***options***)** > > < / *global-options* > **;**
  ESTIMATE '*label*' *estimate-specification* < **(***divisor***=***n***)** > < / *options* > **;**
  FREQ *variable* **;**
  LSMEANS *fixed-effects* < / *options* > **;**
  MODEL *response* < **(***response-options***)** > **=** < *fixed-effects* > < / *model-options* > **;**
  MODEL *events* / *trials* **=** < *fixed-effects* > < / *model-options* > **;**
  PREDDIST < '*label*' > **OUTPRED=***SAS-data-set* < *options* > **;**
  RANDOM *random-effects* < / *options* > **;**
  REPEATED *repeated-effect* < / *options* > **;**
  WEIGHT *variable* **;**

The PROC BGLIMM statement and the MODEL statement are required. The CLASS statement must precede the MODEL statement. You can have multiple RANDOM statements in a program.

The ESTIMATE statement provides a mechanism for computing custom linear combination of the parameters. The PREDDIST statement generates samples from the posterior predictive distribution,

The following sections provide a description of each statement.

PROC BGLIMM Statement

PROC BGLIMM < *options* > **;**

The PROC BGLIMM statement invokes the procedure. Table 31.1 summarizes the available options in the PROC BGLIMM statement by function. The options are then described fully in alphabetical order.

Table 31.1 PROC BGLIMM Statement Options

Option	Description
Basic Options	
DATA=	Specifies the SAS input data set
NBI=	Specifies the number of burn-in iterations
NMC=	Specifies the number of iterations, excluding the burn-in iterations
NTHREADS=	Specifies the number of threads to use
OUTPOST=	Names the output data set to contain posterior samples of parameters
SAMEBYSEED=	Uses the same seed for each BY group
SEED=	Sets the seed for pseudorandom number generation
THIN=	Specifies the thinning rate
Display Options	
NOCLPRINT	Limits or suppresses the display of classification variable levels
MAXRESUBPRT	Suppresses the display of subject levels in the "Random Effect Information" table
Summary, Diagnostics, and Plotting Options	
DIAG=	Controls the convergence diagnostics
DIC	Computes the deviance information criterion (DIC)
PLOTS=	Controls plotting
STATS=	Controls posterior statistics
WAIC	Computes the Watanabe-Akaike information criterion (WAIC)
Other Options	
LOGPOST	Calculates the logarithm of the posterior density and likelihood
MISSING=	Indicates how to handle missing values
SINGCHOL=	Tunes the singularity criterion for Cholesky decomposition
SINGULAR=	Tunes the general singularity criterion

You can specify the following *options* in the PROC BGLIMM statement.

DATA=*SAS-data-set*

names the input data set for PROC BGLIMM to use. The default is the most recently created data set. Observations in this data set are used to compute the log-likelihood function.

DIAGNOSTICS=NONE | (*keyword-list*)

DIAG=NONE | (*keyword-list*)

specifies options for convergence diagnostics. By default, PROC BGLIMM computes the effective sample sizes. The sample autocorrelations, Monte Carlo errors, Geweke test, Raftery-Lewis test, and Heidelberger-Welch test are also available. You can request all the diagnostic tests by specifying DIAGNOSTICS=ALL. You can suppress all the diagnostic tests by specifying DIAGNOSTICS=NONE.

You can specify one or more of the following *keyword-list* options:

ALL

computes all diagnostic tests and statistics. You can combine this option with any other specific tests to modify test options. For example, DIAGNOSTICS=(ALL AUTOCORR(LAGS=(1 5 35))) computes all tests by using default settings and autocorrelations at lags 1, 5, and 35.

AUTOCORR < (*autocorrelation-options*) >

AC < (*autocorrelation-options*) >

computes default autocorrelations at lags 1, 5, 10, and 50 for each variable. You can choose other lags by using the following *autocorrelation-option*:

LAGS=(*numeric-list*)

specifies autocorrelation lags. The *numeric-list* takes only positive integer values.

ESS

computes the effective sample sizes (Kass et al. 1998) of the posterior samples of each parameter. It also computes the correlation time and the efficiency of the chain for each parameter. Small values of ESS might indicate a lack of convergence.

GEWEKE < (*Geweke-options*) >

computes the Geweke spectral density diagnostics; this is a two-sample t test between the first f_1 portion (as specified by the FRAC1= option) and the last f_2 portion (as specified by the FRAC2= option) of the chain. By default, FRAC1=0.1 and FRAC2=0.5, but you can choose other fractions by using the following *Geweke-options*:

FRAC1=*value*

F1=*value*

specifies the beginning proportion of the Markov chain. By default, FRAC1=0.1.

FRAC2=*value*

F2=*value*

specifies the end proportion of the Markov chain. By default, FRAC2=0.5.

HEIDELBERGER < (*Heidel-options*) >

HEIDEL < (*Heidel-options*) >

computes the Heidelberger-Welch diagnostic (which consists of a stationarity test and a halfwidth test) for each variable. The stationary diagnostic test tests the null hypothesis that the posterior samples are generated from a stationary process. If the stationarity test is passed, a halfwidth test is then carried out.

You can also specify the following *Heidel-options*, such as DIAGNOSTICS=HEIDELBERGER(EPS=0.05):

EPS=*value*
specifies a small positive number ϵ such that if the halfwidth is less than ϵ times the sample mean of the retaining iterations, the halfwidth test is passed. By default, EPS=0.1.

HALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the halfwidth test. By default, HALPHA=0.05.

SALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the stationarity test. By default, SALPHA=0.05.

MAXLAG=*number*
specifies the maximum number of autocorrelation lags to use to compute the effective sample size. The value of *number* is also used in the calculation of the Monte Carlo standard error. By default, MAXLAG=MIN(500, MCsample/4), where MCsample is the Markov chain sample size that is kept after thinning—that is, MCsample $= \left[\frac{\text{NMC}}{\text{NTHIN}}\right]$. If *number* is too low, you might observe significant lags, and the effective sample size cannot be calculated accurately. A warning message appears in the SAS log, and you can increase the value of either the MAXLAG= option or the NMC= option accordingly. Specifying this option implies the ESS and MCSE options.

MCSE
MCERROR
computes the Monte Carlo standard error for the posterior samples of each parameter.

NONE
suppresses all the diagnostic tests and statistics. This option is not recommended.

RAFTERY <(*Raftery-options***)>**
RL <(*Raftery-options***)>**
computes the Raftery-Lewis diagnostic, which evaluates the accuracy of the estimated quantile ($\hat{\theta}_Q$ for a given $Q \in (0,1)$) of a chain. $\hat{\theta}_Q$ can achieve any degree of accuracy when the chain is allowed to run for a long time. The algorithm stops when the estimated probability $\hat{P}_Q = \Pr(\theta \leq \hat{\theta}_Q)$ reaches within $\pm R$ of the value Q with probability S; that is, $\Pr(Q - R \leq \hat{P}_Q \leq Q + R) = S$.

You can specify Q, R, S, and a precision level ϵ for a stationarity test by specifying the following *Raftery-options*—for example, DIAGNOSTICS=RAFTERY(QUANTILE=0.05):

ACCURACY=*value*
R=*value*
specifies a small positive number as the margin of error for measuring the accuracy of estimation of the quantile. By default, ACCURACY=0.005.

EPS=*value*
specifies the tolerance level (a small positive number) for the stationarity test. By default, EPS=0.001.

PROB=*value*

S=*value*

specifies the probability of attaining the accuracy of the estimation of the quantile. By default, PROB=0.95.

QUANTILE=*value*

Q=*value*

specifies the order (a value between 0 and 1) of the quantile of interest. By default, QUANTILE=0.025.

DIC< (INCLUDE=*variable***) >**

computes the deviance information criterion (DIC). DIC is calculated by using the posterior mean estimates of the parameters.

The INCLUDE=*variable* specification selects observations that should be included for computing DIC. The *variable* is binary with values being either 0 or 1. The observations for which *variable*=1 are included to obtain DIC, whereas those observations for which *variable*=0 are not included.

LOGPOST

computes the logarithm of the posterior density of the parameters and the likelihood at each iteration. The LogLike and LogPost variables are saved in the OUTPOST= data set.

MAXRESUBPRT< =*number* **>**

limits the display of subject levels in the "Random Effect Information" table. If you specify a *number*, the values of the subject effect are displayed up to that many characters. This option may help reduce the size of the "Subject Values" column in the "Random Effect Information" table if the subject of a RANDOM statement has a long list of levels. By default, MAXRESUBPRT = 200.

MISSING=*keyword*

MISS=*keyword*

specifies how to handle missing values. For more information, see the section "Missing Data" on page 1349. PROC BGLIMM models missing response variables and discards observations that have missing covariates. You can specify the following *keywords*:

CC

COMPLETECASE

assumes a complete case analysis, so all observations that have missing variable values are discarded before the simulation.

CCMODELY

models the missing response variables and discards observations that have missing covariates.

By default, MISSING=CCMODELY.

NBI=*number*

specifies the number of burn-in iterations to perform before saving parameter estimate chains. By default, NBI=500.

NMC=*number*

specifies the number of iterations in the main simulation loop. If you specify a data set in the OUTPOST= option, *number* is the number of posterior samples that are saved for each parameter. This is the MCMC sample size if THIN=1. By default, NMC=5000.

NOCLPRINT< =*number* **>**

suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose list of levels is less than *number* characters. Specifying *number* helps reduce the size of the "Class Level Information" table if some classification variables have a large number of levels. By default, NOCLPRINT = 200.

NTHREADS=*number*

NTHREAD=*number*

specifies the number of threads (CPUs) on which to run analytic computations and simulations simultaneously. Multithreading is the use of more than one thread to perform computations concurrently. When multithreading is possible, you can realize substantial performance gains compared to the performance that you get from sequential (single-threaded) execution. The more threads there are, the faster the computation runs. But do not specify a *number* greater than the number of CPUs on the host where the analytic computations are performed.

PROC BGLIMM performs two types of threading. In sampling fixed-effects parameters, the procedure allocates data to different threads and accumulates values from each thread; in sampling of random-effects parameters, each thread generates a subset of these parameters simultaneously at each iteration. Most sampling algorithms are threaded. NTHREADS=–1 sets the number of available threads to the number of hyperthreaded cores available on the system. By default, NTHREADS=1.

OUTPOST=*SAS-data-set*

specifies an output data set to contain the posterior samples of all parameters and the iteration numbers. It contains the log of the posterior density (LOGPOST) and the log likelihood (LOGLIKE) if you specify the LOGPOST option. By default, no OUTPOST= data set is created.

PLOTS < (*global-plot-option***) > < =** *plot-requests* **< (***option***) > >**

controls the display of diagnostic plots. You can request three types of plots: trace plots, autocorrelation function plots, and kernel density plots. By default, the plots are displayed in panels unless you specify the *global-plot-option* UNPACK. Also, when you specify more than one type of plot, the plots are grouped by parameter unless you specify the *global-plot-option* GROUPBY=TYPE. When you specify only one *plot-request*, you can omit the parentheses around it, as shown in the following example:

```
plots=none
plots(unpack)=trace
plots=(trace density)
```

If ODS Graphics is enabled and you specify PLOTS=ALL, then PROC BGLIMM produces, for each parameter, a panel that contains the trace plot, the autocorrelation function plot, and the density plot. This is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

You can specify the following *global-plot-options*:

FRINGE
adds a fringe plot to the horizontal axis of the density plot.

GROUPBY=PARAMETER | TYPE
GROUP=PARAMETER | TYPE
specifies how the plots are grouped when there is more than one type of plot. By default, GROUPBY=PARAMETER. You can specify the following values:

PARAMETER
groups the plots by parameter.

TYPE
groups the plots by type.

LAGS=*number*
specifies the number of autocorrelation lags to use in plotting the ACF graph. By default, LAGS=50.

SMOOTH
smooths the trace plot by using a fitted penalized B-spline curve (Eilers and Marx 1996).

UNPACKPANEL
UNPACK
unpacks all paneled plots so that each plot in a panel is displayed separately.

You can specify the following *plot-requests*:

ALL
requests all types of plots. PLOTS=ALL is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

AUTOCORR
ACF
displays the autocorrelation function plots for the parameters.

DENSITY
D
KERNEL
K
displays the kernel density plots for the parameters.

NONE
suppresses the display of all plots.

TRACE
T
displays the trace plots for the parameters.

Consider a model that has four parameters, X1–X4. The following list shows which plots are produced for various option settings:

- PLOTS=(TRACE AUTOCORR) displays the trace and autocorrelation plots for each parameter side by side, with two parameters per panel:

Display 1	Trace(X1)	Autocorr(X1)
	Trace(X2)	Autocorr(X2)
Display 2	Trace(X3)	Autocorr(X3)
	Trace(X4)	Autocorr(X4)

- PLOTS(GROUPBY=TYPE)=(TRACE AUTOCORR) displays all the paneled trace plots, followed by panels of autocorrelation plots:

Display 1	Trace(X1)	
	Trace(X2)	
Display 2	Trace(X3)	
	Trace(X4)	
Display 3	Autocorr(X1)	Autocorr(X2)
	Autocorr(X3)	Autocorr(X4)

- PLOTS(UNPACK)=(TRACE AUTOCORR) displays a separate trace plot and a separate correlation plot, parameter by parameter:

Display 1	Trace(X1)
Display 2	Autocorr(X1)
Display 3	Trace(X2)
Display 4	Autocorr(X2)
Display 5	Trace(X3)
Display 6	Autocorr(X3)
Display 7	Trace(X4)
Display 8	Autocorr(X4)

- PLOTS(UNPACK GROUPBY=TYPE)=(TRACE AUTOCORR) displays all the separate trace plots, followed by the separate autocorrelation plots:

Display 1	Trace(X1)
Display 2	Trace(X2)
Display 3	Trace(X3)
Display 4	Trace(X4)
Display 5	Autocorr(X1)
Display 6	Autocorr(X2)
Display 7	Autocorr(X3)
Display 8	Autocorr(X4)

SAMEBYSEED

uses the same seed that you specify in the SEED= option to start the pseudorandom number generator in each BY group. If you omit this option, the initial seed for the next BY group is the one that is generated at the end of the previous BY group.

SEED=*number*

specifies an integer that is used to start the pseudorandom number generator. If you omit this option or if *number* ≤ 0, the seed is generated from the time of day, which is read from the computer's clock.

SINGCHOL=*number*

tunes the singularity criterion in Cholesky decomposition and matrix inversion operations. The default is 1E4 times the machine epsilon, or approximately 1E–12 on most computers.

SINGULAR=*number*

tunes the general singularity criterion applied by the procedure in divisions and inversions. The default is 1E4 times the machine epsilon, or approximately 1E–12 on most computers.

STATISTICS < (*global-stats-options***) > = NONE | ALL |** *stats-request*

STATS < (*global-stats-options***) > = NONE | ALL |** *stats-request*

specifies options for posterior statistics. By default, PROC BGLIMM computes the posterior mean, standard deviation, quantiles, and two 95% credible intervals: equal-tail and highest posterior density (HPD). Other available statistics include the posterior correlation and covariance. You can request all the posterior statistics by specifying STATS=ALL. You can suppress all the calculations by specifying STATS=NONE.

You can specify the following *global-stats-options*:

ALPHA=*numeric-list*

specifies the α level for the equal-tail and HPD intervals. The value of α must be between 0 and 0.5. By default, ALPHA=0.05.

PERCENT=*numeric-list*

PERCENTAGE=*numeric-list*

calculates the posterior percentages. The *numeric-list* contains values between 0 and 100, separated by spaces. By default, PERCENTAGE=(25 50 75).

You can specify the following *stats-requests*:

ALL

computes all posterior statistics. You can combine the ALL option with any other options. For example, STATS(ALPHA=(0.02 0.05 0.1))=ALL computes all statistics by using the default settings and intervals at α levels of 0.02, 0.05, and 0.1.

BRIEF

computes the posterior means, standard deviations, and $100(1-\alpha)\%$ HPD credible interval for each variable. By default, ALPHA=0.05, but you can use the global *global-stats-option* ALPHA= to specify other values. This is the default output for posterior statistics.

CORR

computes the posterior correlation matrix.

COV

computes the posterior covariance matrix.

INTERVAL

INT

computes the $100(1-\alpha)\%$ equal-tail and HPD credible intervals for each variable. By default, ALPHA=0.05, but you can use the *global-stats-option* ALPHA= to specify other intervals of any probabilities.

NONE

suppresses all the statistics.

SUMMARY

SUM

computes the posterior means, standard deviations, and percentile points for each variable. By default, the 25th, 50th, and 75th percentile points are produced, but you can use the *global-stats-option* PERCENT= to request specific percentile points.

THIN=*number*

NTHIN=*number*

controls the thinning rate of the simulation. PROC BGLIMM keeps every *n*th simulation sample and discards the rest. All posterior statistics and diagnostics are calculated by using the thinned samples. By default, THIN=1.

WAIC

computes the Watanabe-Akaike information criterion (WAIC), also known as the widely applicable information criterion. WAIC is proposed as an approximation to n-fold leave-one-out cross validation, which computes the posterior mean and variance of the likelihood and the log likelihood.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC BGLIMM to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the BGLIMM procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. Response variables do not need to be specified in the CLASS statement.

The CLASS statement must precede the MODEL statement. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*. You can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING
DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC BGLIMM orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (., ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values of the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so this option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC BGLIMM interprets values of the ORDER= option:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. You can specify any of the *keywords* shown in the following table.

The default is PARAM=GLM. Design matrix columns are created from CLASS variables according to the corresponding coding schemes.

Value of PARAM=	Coding
EFFECT	Effect coding
GLM	Less-than-full-rank reference cell coding (this *keyword* can be used only in a global option)
REFERENCE **REF**	Reference cell coding

All parameterizations are full rank, except for the GLM parameterization. The REF= option in the CLASS statement determines the reference level for EFFECT and REFERENCE coding and for their orthogonal parameterizations. It also indirectly determines the reference level for a singular GLM parameterization through the order of levels.

If a PARAM= option is specified as a variable option for some variables, then any variables for which PARAM= is not specified use either the EFFECT parameterization if the global PARAM= option is not specified, or the full-rank parameterization indicated in the global PARAM= option if specified. If the global PARAM=GLM option is specified and PARAM= is also specified for some variables, GLM parameterization is used for all variables.

REF=*'level'* | *keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*:

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n* >

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

ESTIMATE Statement

ESTIMATE '*label*' *estimate-specification* < **(***divisor***=***n***)** >< / *options* > **;**

The ESTIMATE statement provides a mechanism for computing custom linear combination of the parameters. The basic element of this statement is the *estimate-specification*, which consists of MODEL statement effects, random effects, and their coefficients. Specifically, an *estimate-specification* takes the form

< *fixed-effect values* . . . > < | *random-effect values* . . . >

You can estimate the linear combination of the parameters $\mathbf{L}'\boldsymbol{\phi}$, where $\mathbf{L}' = (\mathbf{K}'\,\mathbf{M}')$ and $\boldsymbol{\phi}' = (\boldsymbol{\beta}'\,\boldsymbol{\gamma}')$. Based on the *estimate-specification* in your ESTIMATE statement, PROC BGLIMM constructs the vector $\mathbf{L}' = [\mathbf{K}'\ \mathbf{M}']$, where $\mathbf{K}$ is associated with the fixed effects and $\mathbf{M}$ is associated with the G-side random effects.

PROC BGLIMM then produces for $\mathbf{L}'\boldsymbol{\phi}$ an estimate (by using the posterior mean), the standard deviation (by using the posterior standard deviation), and the HPD intervals. Results from all ESTIMATE statements are combined in the ODS table named Estimates.

The ESTIMATE statement has the following arguments:

label identifies the ESTIMATE statement in the table. A label is required for every ESTIMATE statement that you specify. Labels can be up to 32 characters and must be enclosed in quotation marks.

fixed-effect identifies an effect that appears in the MODEL statement. You can use the keyword INTERCEPT as an effect when you are fitting an intercept in the model. You do not need to include all effects that are specified in the MODEL statement.

random-effect identifies an effect that appears in the RANDOM statement. The first random effect must follow a vertical bar (|); however, you are not required to specify random effects.

values are constants that are elements of the $\mathbf{L}$ vector that are associated with the fixed and random effects.

The vector of $\mathbf{L}$ is specified in order. The $\mathbf{K}$ component of $\mathbf{L}$ is specified on the left side of the vertical bars (|). The $\mathbf{M}$ component of $\mathbf{L}$ is specified on the right side of the vertical bars. The estimability checking is necessary.

If PROC BGLIMM finds a portion of the specified estimate statement to be nonestimable, then it displays a message in the log.

In the following program, the first ESTIMATE statement compares the first level with the second level for the effect A, and the second ESTIMATE statement compares the first level with the third level for the effect A in a split-plot study where A has three levels and B has two levels:

```
estimate 'A 1 vs 2' A  1 -1 0   A*B  .5 .5 -.5 -.5 0 0;
estimate 'A 1 vs 3' A  1 0 -1   A*B  .5 .5 0 0 -.5 -.5;
```

Note that no random effects are specified in the preceding statement. The following statements make the same comparison for A when Block and A*Block are random effects:

```
estimate 'A 1 vs 2'
          A        1 -1 0
          A*B     .5 .5 -.5 -.5 0 0 |
          A*Block .25  .25  .25  .25
                 -.25 -.25 -.25 -.25
                   0    0    0    0 ;
estimate 'A 1 vs 3'
          A        1 0 -1
          A*B     .5 .5 0 0 -.5 -.5 |
          A*Block .25  .25  .25  .25
                   0    0    0    0
                 -.25 -.25 -.25 -.25 ;
```

The preceding statements do not contain coefficients for B and Block, because they cancel out in estimated differences between levels of A. Coefficients for B and Block are necessary to estimate the mean of one of the levels of A.

Table 31.2 summarizes the *options* available in the ESTIMATE statement after a slash (/).

Table 31.2 ESTIMATE Statement Options

Option	Description
BYCAT=	Reports estimates for each category of the response in the generalized logit model
DIVISOR=	Specifies a value to divide the coefficients
E	Prints the **L** matrix
EXP	Displays exponentiated estimates
GROUP	Sets up random-effects contrasts between different groups
ILINK	Computes and displays estimates on the inverse linked scale
SUBJECT	Sets up random-effects contrasts between different subjects

BYCATEGORY
BYCAT

reports estimates for each category of the response variable in the multinomial model for nominal data with the generalized logit link. The BYCATEGORY option has no effect unless your model is a generalized (mixed) logit model.

In this example, the response variable Style is multinomial with three categories. The following statements fit a generalized logit model relating the response to the two covariates: School and Program:

```
proc bglimm data=school;
   class School Program;
   model Style = School Program / dist=multinomial link=glogit;
   freq Count;
   estimate 'School 1 vs. 2' School 1 -1 / bycat;
   estimate 'School 1 vs. 2' School 1 -1;
```

```
run;
```

The first ESTIMATE statement compares the School effects separately for each nonredundant response category because of the BYCAT option. The second ESTIMATE statement compares the School effects for the first non-reference category only.

DIVISOR=*value*

specifies a value by which to divide the coefficients so that fractional coefficients can be entered as integer numerators. By default, DIVISOR=1.0.

EST

E

displays the **L** matrix coefficients.

EXP | ODDSRATIO | OR

requests the exponentiated version of the linear combination of the parameters. When you model data by using the logit link function, the estimate of the linear combination of the parameters represents a log odds ratio, and the EXP option produces an odds ratio. Based on the number of iterations in the main simulation loop, PROC BGLIMM takes the exponential of the linear combination of the parameters for each of the posterior samples and then calculates the posterior mean.

GROUP *coeffs*

sets up random-effects contrasts between different groups when you include a GROUP= variable in the RANDOM statement, as in the following example:

```
estimate 'Trt 1 vs 2 @ x=0.4' trt 1 -1  0 | x 0.4 / group 1 -1;
```

By default, ESTIMATE statement coefficients on random effects are distributed equally across groups.

ILINK

reports the parameter estimates on the scale of the mean (the inverse linked scale). PROC BGLIMM computes the value on the mean scale by applying the inverse link to the linear combination of the parameters for each of the posterior samples and then calculating the posterior mean. The interpretation of this quantity depends on the *fixed-effect values* and *random-effect values* that you specify in the ESTIMATE statement and on the link function. In a model for binary data with a logit link, for example, the following statements compute

$$\frac{1}{1+\exp\{-(\alpha_1-\alpha_2)\}}$$

```
proc bglimm data=Sales seed=9988;
   class A;
   model y = A / dist=binary link=logit;
   estimate 'A one vs. two' A 1 -1 / ilink;
run;
```

Here α_1 and α_2 are the fixed-effects solutions that are associated with the first two levels of the classification effect A. This quantity is not the difference of the probabilities associated with the two levels,

$$\pi_1 - \pi_2 = \frac{1}{1 + \exp\{-\beta_0 - \alpha_1\}} - \frac{1}{1 + \exp\{-\beta_0 - \alpha_2\}}$$

SUBJECT *coeffs*

sets up random-effects contrasts between different subjects when you include a SUBJECT= variable in the RANDOM statement. By default, ESTIMATE statement coefficients on random effects are distributed equally across subjects.

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable (in the input data set) that contains the frequency of occurrence for each observation. PROC BGLIMM treats each observation as if it appears f times, where f is the value of the FREQ variable for the observation.

The following *option* can be specified in the FREQ statement after a slash (/):

NOTRUNCATE

NOTRUNC

specifies that frequency values are not truncated to integers.

By default, the frequency value is truncated to an integer if it is not an integer. If the frequency value is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

LSMEANS Statement

LSMEANS *fixed-effects* < / *options* > **;**

The LSMEANS statement computes least squares means (LS-means) of fixed effects. LS-means estimate the marginal means over a balanced population. A more appropriate approach to LS-means views them as linear combinations of the parameter estimates that are constructed in such a way that they correspond to average predicted values in a population where the levels of classification variables are balanced.

The **L** matrix that is constructed to compute them is the same as the **L** matrix that is constructed in other SAS procedures. LS-means computations are not supported for multinomial models.

LS-means are constructed on the linked scale—that is, the scale on which the model effects are additive. For example, in a binomial model with a logit link, the LS-means are predicted population margins of the logits.

LS-means can be computed for any effect in the MODEL statement that involves only CLASS variables. You can specify multiple effects in one LSMEANS statement or in multiple LSMEANS statements, and all LSMEANS statements must appear after the MODEL statement.

The fixed-effects parameters that are used in the LSMEANS statement come directly from the posterior samples. They are marginal posterior samples (with random effects integrated out), not marginal predictive samples.

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *options* in the LSMEANS statement after a slash (/):

DIFF

PDIFF

displays differences of the LS-means in a table titled "Differences of Least Squares Means."

For multiple effects, the results depend on the order of the list, so you should check the output to make sure that the controls are correct.

EST

E

displays the **L** matrix coefficients (coefficients that are used to compute the LS-means).

ILINK

requests that estimates in the "Least Squares Means" table also be reported on the scale of the mean (the inverse linked scale). This option is specific to an LSMEANS statement.

The BGLIMM procedure applies the inverse link transform to the LS-mean that is reported in the Inverse Link Mean column. In a logistic model, for example, this implies that the value that is reported as the inversely linked estimate corresponds to a predicted probability that is based on an average estimable function (the function that produces the LS-mean on the linear scale).

MODEL Statement

MODEL *response* < **(***response-options***)** > **=** < *fixed-effects* > < / *model-options* > **;**

MODEL *events* / *trials* **=** < *fixed-effects* > < / *model-options* > **;**

The MODEL statement, which is required, names the dependent variable and the fixed effects. The *fixed-effects* determine the **X** matrix of the model (see the section "Notation for the Generalized Linear Mixed Model" for details). You specify effects in the same way as in other SAS procedures.

An intercept is included in the fixed-effects model by default. You can remove it by using the NOINT option.

You can specify the dependent variable by using either the *response* syntax or the *events/trials* syntax. The *events/trials* syntax is specific to models for binomial data. A binomial(n,π) variable is the sum of n independent Bernoulli trials with event probability π. Each Bernoulli trial results in either an event or a nonevent (with probability $1-\pi$). You use the *events/trials* syntax to indicate to the BGLIMM procedure that the Bernoulli outcomes are grouped. The value of the second variable, *trials*, gives the number n of Bernoulli trials. The value of the first variable, *events*, is the number of events out of n. The values of both *events* and (*trials*–*events*) must be nonnegative, and the value of trials must be positive. Observations for which these conditions are not met are excluded from the analysis. If the *events/trials* syntax is used, PROC BGLIMM defaults to the binomial distribution. The response is then the *events* variable. The *trials* variable

is accounted for in model fitting as an additional weight. If you use the *response* syntax, the procedure defaults to the normal distribution.

The MODEL statement uses two sets of options. The *response-options* determine how PROC BGLIMM models probabilities for binary, binomial, and multinomial data. The *model-options* control other aspects of model formation and inference. Table 31.3 summarizes these options, and subsequent sections describe them in detail.

Table 31.3 MODEL Statement Options

Option	Description
Response Variable Options	
DESCENDING	Reverses the order of response categories
EVENT=	Specifies the event category in binary and binomial models
ORDER=	Specifies the sort order for the response variable
REF=	Specifies the reference category in generalized logit models
Model Options	
COEFFPRIOR=	Specifies the prior of the fixed-effects coefficients
DIST=	Specifies the response distribution
SCALE=	Specifies a fixed value for the scale parameter
INIT=	Controls the generation of initial values of the fixed-effects coefficients
LINK=	Specifies the link function
NOINT	Excludes the fixed-effects intercept from the model
NOOUTPOST	Suppresses storing the posterior samples of missing responses in the OUTPOST= data set
OFFSET=	Specifies the offset variable
SCALEPRIOR=	Specifies the prior of the scale parameter

Response Variable Options

Response variable options determine how the BGLIMM procedure models probabilities for binary, binomial, and multinomial data.

You can specify the following *response-options* by enclosing them in parentheses after the *response* variable.

DESCENDING
DESC

reverses the order of the response categories. If you specify both the DESCENDING and ORDER= options, PROC BGLIMM orders the response categories according to the ORDER= option and then reverses that order.

EVENT='*category*' | FIRST | LAST

specifies the event category for the binary or binomial response model. PROC BGLIMM models the probability of the event category. This option has no effect when there are more than two response categories.

You can specify any of the following values:

'*category*'
specifies that observations whose value matches *category* (formatted, if a format is applied) in quotation marks represent events in the data. For example, the following statements specify that observations that have a formatted value of '1' represent events in the data. The probability that is modeled by PROC BGLIMM is thus the probability that the variable def takes the (formatted) value '1'.

```
proc bglimm data=MyData;
    class A B C;
    model Def(event ='1') = A B C;
run;
```

FIRST
designates the first ordered category as the event.

LAST
designates the last ordered category as the event.

By default, EVENT=FIRST.

ORDER=FORMATTED | FREQ | INTERNAL
specifies the sort order of the levels of the *response* variable. When ORDER=FORMATTED (the default) for numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC BGLIMM run or in the DATA step that created the data set), the levels are ordered by their internal (numeric) value. Table 31.4 shows the interpretation of the ORDER= option.

Table 31.4 Sort Order

ORDER=	Levels Sorted By
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count (levels that have the most observations come first in the order)
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For the FORMATTED and INTERNAL orders, the sort order is machine-dependent.

For more information about sort order, see the chapter about the SORT procedure in *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Language Reference: Concepts*.

REFERENCE='*category*' | FIRST | LAST
REF='*category*' | FIRST | LAST
specifies the reference category for the binary or binomial response model. Specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify any of the following values:

'*category*'
specifies that observations whose value matches *category* (formatted, if a format is applied) are designated as the reference.

FIRST
designates the first ordered category as the reference.

LAST
designates the last ordered category as the reference.

By default, REF=LAST.

Model Options

You can specify the following *model-options* in the MODEL statement after a slash (/):

COEFFPRIOR < (*bounded-prior-options*) > =CONSTANT | NORMAL < (*options*) >
CPRIOR < (*bounded-prior-options*) > =CONSTANT | NORMAL < (*options*) >
specifies the prior distribution for the fixed-effects coefficients. You can specify the following prior distributions:

CONSTANT
UNIFORM
specifies the noninformative and improper prior of a constant for the regression coefficients. This is the default prior distribution.

NORMAL < (*options*) >
specifies a normal prior distribution for the regression coefficients. When you specify COEFFPRIOR=NORMAL, the default normal distribution is $N(\mathbf{0}, 10^4\mathbf{I})$, where $\mathbf{I}$ is the identity matrix. You can specify the following *options* for the normal prior, enclosed in parentheses:

INPUT=*SAS-data-set*
specifies a SAS data set that contains the mean and covariance information of the normal prior. The data set must have a _TYPE_ variable to represent the type of each observation and a variable for each regression coefficient. If the data set also contains a _NAME_ variable, the values of this variable are used to identify the covariance for the _TYPE_='COV' observations; otherwise, the _TYPE_='COV' observations are assumed to be in the same order as the explanatory variables in the MODEL statement. PROC BGLIMM reads the mean vector from the observation for which _TYPE_='MEAN' and reads the covariance matrix from observations for which _TYPE_='COV'. For an independent normal prior, you can specify the variances by using _TYPE_='VAR'; alternatively, you can specify the precision (inverse of the variances) by using _TYPE_='PRECISION'.

VAR < =*c* >
specifies the normal prior $N(\mathbf{0}, c\mathbf{I})$, where $\mathbf{I}$ is the identity matrix and c is a scalar. By default, $c = 1e4$.

This VAR=*c* option specifies a normal prior with a single variance value that is equal to *c*; hence the covariance matrix is uncorrelated in the multivariate normal cases. If you want a specific mean and variance that are different from the default values, you can provide a SAS data set that contains the mean and variance of the normal prior by using the INPUT=*SAS-data-set* option. See the following example:

```
data Prior1;
   length _TYPE_ $4;
   input _TYPE_ $ Age Duration;
   datalines;
   Mean  1.0 -2.0
   Var   10  9
   ;

proc bglimm data=Neuralgia seed=123;
   model numPain = Age Duration / dist=binary cprior=normal(input=Prior1);
run;
```

Instead of providing values only for the variances, you might want to specify values for the whole covariance matrix. Here is an example:

```
data Prior2;
   length _TYPE_ $4 _NAME_ $13;
   input _TYPE_ $ _NAME_ $ Intercept Age Treatment_A;
   datalines;
   Mean .           1.0 -2.0  1.0
   Cov Intercept    1    0.5  0.2
   Cov Age          0.5  2    0.5
   Cov Treatment_A  0.2  0.5  3
   ;

proc bglimm data=Neuralgia seed=123;
   class Treatment;
   model numPain = Age Duration Treatment / dist=binary
         cprior=normal(input=Prior2);
run;
```

If there is a classification variable for which you want to specify a prior value, you need to name it by using both the variable name and one of its levels. For example, the last parameter is called "Treatment_A" instead of "Treatment". If any model parameter is not specified in the prior data, its prior mean and variance use the default values.

You can specify the following *bounded-prior-options* for truncated prior distributions, enclosed in parentheses:

LOWER=*value-list*
: assigns lower bounds of the prior distribution for the regression coefficients. The *value-list* follows the order of the coefficient list. A missing value (.) sets the lower bound for a coefficient to the default value, negative infinity.

UPPER=*value-list*
: assigns upper bounds of the prior distribution for the regression coefficients. The *value-list* follows the order of the coefficient list. A missing value (.) sets the upper bound for a coefficient to the default value, positive infinity.

For example, the following MODEL statement specifies a noninformative prior and assigns the lower bounds –1, $-\infty$, –3 and the upper bounds 1, 2, ∞ to the first, second, and third coefficients in the model, respectively:

```
proc bglimm data=Neuralgia seed=123;
   model numPain = Age Duration / dist=binary
         cprior(lower=(-1 . -3) upper=(1 2 .))=constant;
run;
```

The length of the list should match the number of fixed-effects parameters, including the intercept if the model has one. If the list length is less than the number of fixed-effects parameters, no constraints are set for the remaining parameters. If the list length is greater than the number of fixed-effects parameters, the extra bounds are ignored.

DISTRIBUTION=*keyword*

DIST=*keyword*

ERROR=*keyword*

ERR=*keyword*

specifies the response distribution for the model. The *keywords* and their associated distributions are shown in Table 31.5.

Table 31.5 Built-In Distribution Functions

DISTRIBUTION=	Distribution Function
BETA	Beta
BINARY	Binary
BINOMIAL	Binary or binomial
EXPONENTIAL | EXPO	Exponential
GAMMA | GAM	Gamma
GEOMETRIC | GEOM	Geometric
INVGAUSS | IG	Inverse Gaussian
MULTINOMIAL	Multinomial
NEGBINOMIAL | NEGBIN | NB	Negative binomial
NORMAL | GAUSSIAN | GAUSS	Normal
POISSON | POI	Poisson

If you do not specify a distribution, PROC BGLIMM defaults to the normal distribution for continuous response variables and to the binomial or multinomial distribution for classification or character variables, unless you use the *events/trial* syntax in the MODEL statement. If you use the *events/trial* syntax, the procedure defaults to the binomial distribution.

If you do not specify a link function in the LINK= option, a default link function is used. The default link function for each distribution and other commonly used link functions are shown in Table 31.6. You can also use the LINK= option to specify any link function shown in Table 31.7.

Table 31.6 Default and Commonly Used Link Functions

DISTRIBUTION=	Default Link Function	Other Commonly Used Link Functions
BETA	Logit	Probit, complementary log-log, log-log
BINARY	Logit	Probit, complementary log-log, log-log
BINOMIAL	Logit	Probit, complementary log-log, log-log
EXPONENTIAL | EXPO	Log	Reciprocal
GAMMA | GAM	Log	Reciprocal
GEOMETRIC | GEOM	Log	
INVGAUSS | IG	Reciprocal square	
MULTINOMIAL	Cumulative logit	Cumulative probit, cumulative log-log, generalized logit
NEGBINOMIAL | NEGBIN | NB	Log	
NORMAL | GAUSSIAN | GAUSS	Identity	Log
POISSON | POI	Log	

INIT=*keyword-list* | **(***numeric-list***)**

INITIAL=*keyword-list* | **(***numeric-list***)**

specifies options for generating the initial values for the coefficients parameters that you specify as *fixed-effects* in the MODEL statement. You can specify the following *keywords*:

LIST=*numeric-list*

assigns the numbers to use as the initial values of the fixed effects in the corresponding list order, including the intercept. The length of the *numeric-list* must be the same as the number of fixed effects. For example, the following statement assigns the values 1, 2, and 3 to the first, second, and third coefficients in the model and prints the table of initial values:

```
model y = x / init=(list=(1 2 3) pinit);
```

If the number of items in the *numeric-list* is less than the number of fixed effects, the initial value of each remaining parameter is replaced by the corresponding default initial value. For example, the corresponding mode of the posterior density is used. If the number of items in the *numeric-list* is greater than the number of fixed effects, the extra numbers are ignored.

PINIT

tabulates initial values for the fixed effects. (By default, PROC BGLIMM does not display the initial values.)

POSTMODE

uses the mode of the posterior density as the initial value of the parameter.

PRIORMODE

uses the mode of the prior density as the initial value of the parameter.

By default, INIT=POSTMODE.

LINK=*keyword*

specifies the link function for the model. The *keywords* and the associated link functions are shown in Table 31.7. Default and commonly used link functions for the available distributions are shown in Table 31.6.

Table 31.7 Built-In Link Functions

LINK=	Link Function	$g(\mu) = \eta =$
CUMCLL | CCLL	Cumulative Complementary log-log	$\log(-\log(1-\pi))$
CUMLOGIT | CLOGIT	Cumulative logit	$\log(\pi/(1-\pi))$
CUMLOGLOG	Cumulative log-log	$-\log(-\log(\pi))$
CUMPROBIT | CPROBIT	Cumulative probit	$\Phi^{-1}(\pi)$
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$
GLOGIT | GENLOGIT	Generalized logit	
IDENTITY | ID	Identity	μ
INVERSE | RECIPROCAL	Reciprocal	$1/\mu$
LOG	Logarithm	$\log(\mu)$
LOGIT	Logit	$\log(\mu/(1-\mu))$
LOGLOG	Log-log	$-\log(-\log(\mu))$
POWERMINUS2	Power with exponent –2	$1/\mu^2$
PROBIT	Probit	$\Phi^{-1}(\mu)$

$\Phi^{-1}(\cdot)$ in the probit and cumulative probit links denotes the quantile function of the standard normal distribution. For the other cumulative links, π denotes a cumulative category probability. The cumulative and generalized logit link functions are appropriate only for the multinomial distribution. When you choose a cumulative link function, PROC BGLIMM assumes that the data are ordinal. When you specify LINK=GLOGIT, the procedure assumes that the data are nominal (not ordered).

NOINT

excludes the intercept from the fixed-effects model. An intercept is included by default.

NOOUTPOST

suppresses storing the posterior samples of missing responses in the OUTPOST= data set. By default, PROC BGLIMM outputs the posterior samples of all missing responses to the OUTPOST= data set.

OFFSET=*variable*

specifies a *variable* to use as an offset to the linear predictor. An offset plays the role of an effect whose coefficient is known to be 1. The offset *variable* cannot be a classification variable or appear elsewhere in the MODEL statement. Observations that have missing values for the offset *variable* are excluded from the analysis.

SCALE=*options*

specifies a fixed value for the scale parameter that is denoted by ϕ in the log-likelihood functions. Not all exponential family distributions have a scale parameter. For the normal distribution, ϕ corresponds to the variance of the response. This option is available only to the normal distribution that has the identity link and has no repeated measurements.

You can specify the following *options*:

SCALE=*variable*

enables you to use a *variable* to specify what value the scale parameter takes for each observation in the DATA= data set. The variable value can be different from observation to observation.

SCALE=*c*

specifies a numerical value for the scale parameter of a distribution. The value is the same for all the observation lines in the DATA= data set. The specified value must be within the range of the scale parameter. By default, *c*=1.

By default, when you omit the SCALE= option, the scale is a random parameter whose prior distribution is specified in the SCALEPRIOR= option.

SCALEPRIOR=*prior-distribution*

SPRIOR=*prior-distribution*

specifies the prior distribution for the scale parameter, if the model has a scale parameter. For models that do not have a dispersion parameter (the Poisson and binomial), this option is ignored.

You can specify the one of following *prior-distributions*:

GAMMA< (*options***) >**

specifies a gamma prior $\mathrm{G}(a, b)$ with the density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. The default is $\mathrm{G}(10^{-4}, 10^{-4})$. You can set the parameters of the gamma distribution by specifying one or both of the following *options*, separated by a comma or a space:

SHAPE=*a*

specifies the shape parameter *a* of the gamma distribution. By default, SHAPE=0.0001.

ISCALE=*b*

specifies the inverse-scale parameter *b* of the gamma distribution. By default, ISCALE=0.0001.

IGAMMA< (*options***) >**

specifies an inverse gamma prior $\mathrm{IG}(a, b)$ with the density $f(t) = \frac{b^a}{\Gamma(a)} t^{-(a+1)}\mathrm{e}^{-b/t}$. The hyperparameters a and b are the shape and scale parameters of the inverse gamma distribution, respectively. The default is $\mathrm{IG}(2, 2)$ for the normal likelihood with an identity link, and $\mathrm{IG}(2.001, 0.001)$ for all other models. You can set the parameters of the inverse gamma distribution by specifying one or both of the following *options*, separated by a comma or a space:

SHAPE=*a*

specifies the shape parameter *a* of the inverse gamma distribution. By default, SHAPE=2 for the normal likelihood with an identity link, and SHAPE=2.001 for all other models.

SCALE=*b*

specifies the scale parameter *b* of the inverse gamma distribution. By default, SCALE=2 for the normal likelihood with an identity link, and SCALE=0.001 for all other models.

IMPROPER
specifies an improper prior with the density $f(t)$ proportional to t^{-1}.

UNIFORM< (*options*) >
UNIF< (*options*) >
specifies a uniform prior, $\mathrm{UNIFORM}(a, b)$, for the scale parameter. You can set the lower and upper bounds of the uniform distribution by specifying one or both of the following *options*, separated by a comma or a space:

LOWER=*a*
specifies the lower bound *a* of the uniform distribution. The lower bound must be nonnegative. By default, LOWER=0.

UPPER=*b*
specifies the upper bound *b* of the uniform distribution. The upper bound must be positive. By default, UPPER=1E10.

For the normal likelihood with an identity link, the prior for the scale parameter is always an inverse gamma, and the default is $\mathrm{IG}(2, 2)$. Any other prior specification is ignored if the prior is not an inverse gamma.

For any other likelihood or link function, you can choose a gamma prior, an inverse gamma prior, an improper prior, or a uniform prior. The default is the improper prior.

PREDDIST Statement

PREDDIST < '*label*' > **OUTPRED=***SAS-data-set* < **NSIM=***n* > < **COVARIATES=***SAS-data-set* >
< **STATISTICS=***options* > **;**

The PREDDIST statement creates a new SAS data set that contains random samples from the posterior predictive distribution of the response variable. The posterior predictive distribution is the distribution of unobserved observations (prediction) conditional on the observed data. Let $\mathbf{y}$ be the observed data, $\mathbf{X}$ be the covariates, θ be the parameter, and $\mathbf{y}_{\text{pred}}$ be the unobserved data. The posterior predictive distribution is defined as

$$\begin{aligned} p(\mathbf{y}_{\text{pred}}|\mathbf{y}, \mathbf{X}) &= \int p(\mathbf{y}_{\text{pred}}, \theta|\mathbf{y}, \mathbf{X}) d\theta \\ &= \int p(\mathbf{y}_{\text{pred}}|\theta, \mathbf{y}, \mathbf{X}) p(\theta|\mathbf{y}, \mathbf{X}) d\theta \end{aligned}$$

Given the assumption that the observed and unobserved data are conditional independent given θ, the posterior predictive distribution can be further simplified as

$$p(\mathbf{y}_{\text{pred}}|\mathbf{y}, \mathbf{X}) = \int p(\mathbf{y}_{\text{pred}}|\theta) p(\theta|\mathbf{y}, \mathbf{X}) d\theta$$

The posterior predictive distribution is an integral of the likelihood function $p(\mathbf{y}_{\text{pred}}|\theta)$ with respect to the posterior distribution $p(\theta|\mathbf{y})$. The PREDDIST statement generates samples from a posterior predictive distribution on the basis of draws from the posterior distribution of θ.

You can specify the following options:

COVARIATES=*SAS-data-set*

names the SAS data set that contains the sets of explanatory variable values for which the predictions are established. This data set must contain data that have the same variable names that are used in the likelihood function. If you omit this option, the DATA= data set that you specify in the PROC BGLIMM statement is used instead.

ILINK

outputs the inverse link function of the linear predictor for each observation.

LINP

outputs the linear predictors.

MILINK

outputs the inverse link function of the marginal linear predictor for each observation.

MLINP

outputs the marginal linear predictor for each observation.

NSIM=*n*

specifies the number of simulated predicted values. By default, *n* is the same as the NMC= option value that you specify in the PROC BGLIMM statement.

OUTPRED=*SAS-data-set*

creates an output data set to contain the samples from the posterior predictive distribution. The output variable names are listed as resp_1–resp_*m*, where resp is the name of the response variable and *m* is the number of observations in the COVARIATES= data set in the PREDDIST statement. If the COVARIATES= data set is not specified, *m* is the number of observations in the DATA= data set that you specify in the PROC BGLIMM statement.

Table 31.7 displays the *keywords* for the variables to be included in the OUTPRED= data set.

Table 31.7 Keywords for Variables in OUTPRED= Data Set

Keyword	Description	Expression	Variable Name
ILINK	Mean using inverse link	$g^{-1}(\eta_i)$	ILink
LINP	Linear predictor	$\eta_i = \mathbf{x}_i\boldsymbol{\beta} + \mathbf{z}_i\boldsymbol{\gamma}_i$	Linp
MILINK	Marginal mean using inverse link	$g^{-1}(\eta_i^m)$	MILnk
MLINP	Marginal linear predictor	$\eta_i^m = \mathbf{x}_i\boldsymbol{\beta}$	MLinp

STATISTICS< (*global-options***) > = NONE | ALL |***stats-request***

STATS< (*global-options***) > = NONE | ALL |***stats-request***

specifies options for calculating posterior statistics. This option works in exactly the same way as the STATISTICS= option in the PROC BGLIMM statement. By default, the STATS option takes the same value that you specify in the STATISTICS= option in the PROC BGLIMM statement.

RANDOM Statement

RANDOM *random-effects* < / *options* > **;**

Using notation from the section "Notation for the Generalized Linear Mixed Model" on page 1278, the RANDOM statement defines the **Z** matrix of the mixed model, the random effects in the $\boldsymbol{\gamma}$ vector, and the covariance structure of **G**.

The **Z** matrix is constructed exactly like the **X** matrix for the fixed effects, and the **G** matrix is constructed to correspond to the effects that constitute **Z**. The covariance structure of **G** is defined by using the TYPE= option. The random effects can be classification or continuous effects, and you can specify multiple RANDOM statements.

You can specify INTERCEPT (or INT) as a random effect to indicate the intercept. PROC BGLIMM does not include the intercept in the RANDOM statement by default as it does in the MODEL statement.

Table 31.8 summarizes the *options* available in the RANDOM statement. All *options* are then discussed in alphabetical order.

Table 31.8 RANDOM Statement Options

Option	Description
Construction and Sampling of Covariance Structure	
COVALG=	Specifies the sampling algorithm for the **G** matrix
COVPRIOR=	Specifies the prior for the **G** matrix
GROUP=	Varies the **G** covariance matrix by group
RESIDUAL	Designates a covariance structure as R-side
SUBJECT=	Identifies the subjects in the model
TYPE=	Specifies the type of the **G** matrix
Statistical Output	
G	Displays the estimated **G** matrix
GCORR	Displays the correlation matrix that corresponds to the estimated **G** matrix
MONITOR	Displays the summary, diagnostic statistics, and plots of individual random effects
NOOUTPOST	Suppresses storing the posterior samples of individual random effects in the OUTPOST= data set

You can specify the following *options* in the RANDOM statement after a slash (/).

COVALG=< (*algorithms*) >

specifies the sampling algorithm that is used to sample the parameters in the **G** matrix of the corresponding RANDOM statement, when conjugate sampling is unavailable. For more information, see the section "Sampling Methods" on page 1341. You can specify one of the following sampling *algorithms*:

NUTS< (*nuts-options*) >

applies the No-U-Turn Sampler (NUTS) of the Hamiltonian algorithm to sample the parameters in the **G** matrix of the corresponding RANDOM statement. The NUTS algorithm is a version of the adaptive Hamiltonian Monte Carlo algorithm with automatic tuning of the step size and number of steps in each iteration. For more information, see the section "Hamiltonian Monte Carlo Sampler" on page 1343. You can specify the following *nuts-options*:

DELTA=*value*

specifies the target acceptance rate during the tuning process of the NUTS algorithm. Increasing the *value* can often improve mixing, but it can also significantly slow down the sampling. By default, DELTA=0.6.

MAXHEIGHT=*value*

specifies the maximum height of the NUTS algorithm tree. The taller the tree, the more gradient evaluations per iteration the procedure calculates. The number of evaluations is 2^{height}. Usually, the height of a tree should be no more than 10 during the sampling stage, but it can go higher during the tuning stage. A larger number indicates that the algorithm is having difficulty converging. By default, MAXHEIGHT=10.

NOCONJCOV

uses the NUTS algorithm, instead of the conjugate sampler, on the scale parameter of the **G** matrix when the covariance type is AR(1) or ARMA(1,1).

NTU=*n*

specifies the number of tuning iterations for the NUTS algorithm to use. By default, *n* is the same as the NBI= option value that you specify in the PROC BGLIMM statement.

STEPSIZE=*value*

specifies the initial step size in the NUTS algorithm. By default, STEPSIZE=0.1.

SLICE< (*option*) >

applies the slice sampling algorithm to sample the parameters in the **G** matrix of the corresponding RANDOM statement. Slice sampling is a Markov chain Monte Carlo (MCMC) method that can be applied to sample any continuous parameter, as long as its unnormalized conditional distribution can be evaluated at any parameter value. For more information, see the section "Slice Sampler" on page 1344. You can specify the following *option*:

NOCONJCOV

uses the slice sampler, instead of the conjugate sampler, on the scale parameter of the **G** matrix when the covariance type is AR(1) or ARMA(1,1).

By default, COVALG=SLICE.

COVPRIOR=*prior-distribution*

specifies a prior distribution for the covariance matrix, **G**, of the random effects, when the **G** matrix is of the UN, UN(1), VC, or TOEP(1) type, where a conjugate sampler is used to sample the covariance matrix. This option is ignored for other covariance types, because you assign a flat prior to the **G** matrix if its type is not UN, UN(1), VC, or TOEP(1). For more information, see the section "Prior for the G-Side Covariance" on page 1345.

You can specify one of the following *prior-distributions*:

HALFCAUCHY < (SCALE=*a*) >

HCAUCHY < (SCALE=*a*) >

HC < (SCALE=*a*) >

specifies the prior to be a half-Cauchy distribution. The half-Cauchy prior is applied only to the diagonal terms (variances) of the **G** matrix. The off-diagonal terms of the **G** matrix are assumed to have a flat prior.

You can specify the scale parameter *a* of the half-Cauchy distribution. The scale parameter *a* must be positive. By default, SCALE=25.

HALFNORMAL < (VAR=*a*) >

HNORMAL < (VAR=*a*) >

HN < (VAR=*a*) >

specifies the prior to be a half-normal distribution. The half-normal prior is applied only to the diagonal terms (variances) of the **G** matrix. The off-diagonal terms of the **G** matrix are assumed to have a flat prior.

You can specify the variance *a* of the half-normal distribution. The variance *a* must be positive. By default, VAR=25.

IGAMMA < (*options*) >

specifies an inverse gamma prior, $\mathrm{IG}(a, b)$, with the density $f(t) = \frac{b^a}{\Gamma(a)} t^{-(a+1)} \mathrm{e}^{-b/t}$ for each diagonal term of the **G** matrix. It is the default prior for the covariance types UN(1), VC, and TOEP(1).

You can set the parameters of the inverse gamma distribution by specifying one or both of the following *options*, separated by a comma or a space:

SHAPE=*a*

specifies the shape parameter *a* of the inverse gamma distribution. By default, SHAPE=2.

SCALE=*b*

specifies the scale parameter *b* of the inverse gamma distribution. BY default, SCALE=2.

IWISHART < (*options*) >

IWISH < (*options*) >

IW < (*options*) >

specifies an inverse Wishart prior, $\mathrm{IWISHART}(a, \mathbf{S})$, for the **G** matrix of the random effects. The hyperparameters *a* and **S** are the degrees of freedom and scale matrix of an inverse Wishart distribution, respectively. The inverse Wishart prior is the default prior for the UN covariance type. The following examples show several ways to specify an inverse Wishart prior distribution:

```
covprior=iwishart
covprior=iwishart(scale=25)
covprior=iwishart(df=6, diagonal=(1 4 9))
```

You can set the parameters of the inverse Wishart distribution by specifying one or more of the following *options*, separated by a comma or a space:

DF=*a*
specifies the degrees of freedom of the inverse Wishart distribution. The default is the dimension of the **G** matrix plus 3.

SCALE=*b*
specifies $\mathbf{S} = b\mathbf{I}$ as the scale matrix of the inverse Wishart distribution, where **I** is the identity matrix. The default is the dimension of the **G** matrix plus 3. You can set the scale matrix **S** by specifying one of these *options*: SCALE=, DIAGONAL=, or COV=.

DIAGONAL=*numeric-list*
specifies a list of positive values for the diagonal terms in the scale matrix **S** of the inverse Wishart distribution. The default value of each diagonal term is the dimension of the **G** matrix plus 3. You can set the scale matrix **S** by specifying one of these *options*: SCALE=, DIAGONAL=, or COV=.

COV=*numeric-list*
specifies a list of values for the entries in the lower-triangle portion of the scale matrix **S** for the inverse Wishart distribution. The specified values of the diagonal terms must be positive, and the whole matrix must be positive definite. The order of the list is rowwise. The default setting is $\mathbf{S} = b\mathbf{I}$, where b is the dimension of the **G** matrix plus 3. You can set the scale matrix **S** by specifying one of these *options*: SCALE=, DIAGONAL=, or COV=.

SIWISHART < (*options*) >

SIWISH < (*options*) >

SIW < (*options*) >
specifies a scaled inverse Wishart prior for the **G** matrix of the random effects.

You can set the parameters of the scaled inverse Wishart distribution by specifying one or more of the following *options*, separated by a comma or a space:

DF=*a*
specifies the degrees of freedom *a* of the scaled inverse Wishart distribution. The default is the dimension of the **G** matrix of the random effects plus 3.

SCALE=*b*
specifies $b\mathbf{I}$ as the scale matrix of the scaled inverse Wishart distribution, where **I** is the identity matrix. The default is the dimension of the **G** matrix of the random effects plus 3.

VAR=*c*
specifies the variance parameter *c* of the normal prior for $\log(\delta_i)$. By default, VAR=1.

UNIFORM < (*options*) >

UNIF < (*options*) >
specifies a uniform prior, $\mathrm{UNIFORM}(a, b)$, for the **G** matrix of the random effects. The uniform prior is applied to standard deviations (the square root of the diagonal terms) of the **G** matrix.

You can set the lower and upper bounds of the uniform distribution by specifying one or both of the following *options*, separated by a comma or a space:

LOWER=*a*
specifies the lower bound *a* of the uniform distribution. The lower bound must be nonnegative. By default, LOWER=0.

UPPER=*b*
specifies the upper bound *b* of the uniform distribution. The upper bound must be positive. By default, UPPER=1E10.

G
displays the estimated **G** matrix for G-side random effects that are associated with this RANDOM statement. PROC BGLIMM displays blanks for values that are 0. The ODS table name is G.

GCORR
displays the correlation matrix that corresponds to the estimated **G** matrix for G-side random effects that are associated with this RANDOM statement. PROC BGLIMM displays blanks for values that are 0. The ODS table name is GCORR.

GROUP=*effect*
GRP=*effect*
identifies groups by which to vary the covariance parameters. All observations that have the same level of the *effect* have the same covariance parameters. Each new level of the grouping effect produces a new set of covariance parameters. You should exercise caution in properly defining the *effect*, because strange covariance patterns can result when it is misused. Also, the *effect* can greatly increase the number of estimated covariance parameters, which can adversely affect the sampling process.

The GROUP= *effect* must be specified in the CLASS statement.

MONITOR< =(*numeric-list* | **RANDOM (***number***)) >**
SOLUTION< =(*numeric-list* | **RANDOM (***number***)) >**
S< =(*numeric-list* | **RANDOM (***number***)) >**
displays results (summary, diagnostic statistics, and plots) for the individual-level random-effects parameters. By default, to save time and space, PROC BGLIMM does not print results for individual-level random-effects parameters. In models that have a large number of individual random effects (for example, tens of thousands), it can take a long time to display the summary, diagnostic statistics, and plots for all the individual-level parameters, so be cautious when using this option.

You can monitor a subset of the random-effects parameters. You can provide a numeric list of the SUBJECT indexes, or PROC BGLIMM can randomly choose a subset of all subjects for you.

To monitor a list of random-effects parameters for certain subjects, you can provide their indexes as follows:

```
random x / subject=index monitor=(1 to 5 by 2 23 57);
```

In this case, PROC BGLIMM outputs results of random effects for subjects 1, 3, 5, 23, and 57. If the number of items in the list is greater than the number of subjects, the extra list items are ignored.

PROC BGLIMM can also randomly choose a subset of all the subjects to monitor, if you submit a statement such as the following:

```
random x / subject=index monitor=(random(12));
```

In this case, PROC BGLIMM outputs results of random effects for 12 randomly selected subjects. You control the sequence of the random indexes by specifying the SEED= option in the PROC BGLIMM statement.

When you specify the MONITOR option, it uses the values that you specify in the STATISTICS= and PLOTS= options in the PROC BGLIMM statement.

NOOUTPOST

suppresses storing the posterior samples of individual random-effects parameters in the OUTPOST= data set. By default, PROC BGLIMM outputs the posterior samples of all random-effects parameters to the OUTPOST= data set. You can use this option to avoid saving the random-effects parameters. In models that have a large number of individual random effects (for example, tens of thousands), PROC BGLIMM can run faster if it does not save the posterior samples of all the individual random effects.

When you specify both the NOOUTPOST option and the MONITOR option, PROC BGLIMM outputs the list of variables that are monitored.

There is a limit on the maximum number of variables that you can save to an OUTPOST= data set. If you run a large-scale random-effects model in which the number of parameters exceeds that limit, the NOOUTPOST option is invoked automatically and PROC BGLIMM does not save the individual random-effects draws to the output data set. You can use the MONITOR option to select a subset of the parameters to store in the OUTPOST= data set.

RESIDUAL

RSIDE

specifies that the random effects listed in this statement be R-side effects. You use this option in the RANDOM statement if the covariance matrix is for the R-side. Specifying this option is equivalent to using the REPEATED statement. For example, if it is necessary to order the columns of the R-side AR(1) covariance structure by the Time variable, you can use the RESIDUAL option as in the following statements:

```
class time id;
random time / subject=id type=ar(1) residual;
```

SUBJECT=*effect*

SUB=*effect*

identifies the subjects in the model for the random effects.

A set of random effects is estimated for each subject. All variables in the *effect* must be declared as categorical variables in the CLASS statement. PROC BGLIMM assumes independence across subjects, conditional on other parameters in the model. Specifying a subject effect is equivalent to nesting all other effects in the RANDOM statement within the subject effect. Thus, for the RANDOM statement, the SUBJECT= option produces a block-diagonal structure that has identical blocks.

For more information about specifying a random effect with or without the SUBJECT= variable, see the section "Treatment of Subjects in the RANDOM Statement" on page 1348.

TYPE=*covariance-structure*

specifies the covariance structure of the **G** matrix for G-side effects.

Although a variety of structures are available, many applications call for either simple diagonal (TYPE=VC) or unstructured covariance matrices. The default structure is TYPE=VC (variance components), which models a different variance component for each random effect.

If you want different covariance structures in different parts of **G**, you must use multiple RANDOM statements with different TYPE= options.

Valid values for *covariance-structure* and their descriptions are provided in Table 31.9.

Table 31.9 Covariance Structures

Structure	Description	Parms	(i, j) Element
ANTE(1)	Antedependence	$2t-1$	$\sigma_i\sigma_j\prod_{k=i}^{j-1}\rho_k$
AR(1)	Autoregressive(1)	2	$\sigma^2\rho^{\lvert i-j\rvert}$
ARH(1)	Heterogeneous AR(1)	$t+1$	$\sigma_i\sigma_j\rho^{\lvert i-j\rvert}$
ARMA(1,1)	ARMA(1,1)	3	$\sigma^2[\gamma\rho^{\lvert i-j\rvert-1}1(i\neq j)+1(i=j)]$
CS	Compound symmetry	2	$\sigma_1+\sigma^2 1(i=j)$
CSH	Heterogeneous compound symmetry	$t+1$	$\sigma_i\sigma_j[\rho 1(i\neq j)+1(i=j)]$
FA(1)	Factor analytic	$2t$	$\lambda_i\lambda_j+d_i 1(i=j)$
HF	Huynh-Feldt	$t+1$	$(\sigma_i^2+\sigma_j^2)/2-\lambda 1(i\neq j)$
TOEP	Toeplitz	t	$\sigma_{\lvert i-j\rvert+1}$
TOEP(q)	Banded Toeplitz	q	$\sigma_{\lvert i-j\rvert+1}1(\lvert i-j\rvert<q)$
TOEPH	Heterogeneous Toeplitz	$2t-1$	$\sigma_i\sigma_j\rho_{\lvert i-j\rvert}$
TOEPH(q)	Banded heterogeneous Toeplitz	$t+q-1$	$\sigma_i\sigma_j\rho_{\lvert i-j\rvert}1(\lvert i-j\rvert<q)$
UN	Unstructured	$t(t+1)/2$	σ_{ij}
UN(q)	Banded	$\frac{q}{2}(2t-q+1)$	$\sigma_{ij}1(\lvert i-j\rvert<q)$
VC	Variance components	q	$\sigma_k^2 1(i=j)$ and i corresponds to kth effect

In Table 31.9, Parms refers to the number of covariance parameters in the structure, t is the overall dimension of the covariance matrix, q is the order parameter, and $1(A)$ equals 1 when A is true and 0 otherwise. For example, $1(i=j)$ equals 1 when $i=j$ and 0 otherwise, and $1(|i-j|<q)$ equals 1 when $|i-j|<q$ and 0 otherwise. For the TYPE=TOEPH structures, $\rho_0=1$.

ANTE(1)

specifies a first-order antedependence structure (Kenward 1987; Patel 1991) parameterized in terms of variances and correlation parameters. If t ordered random variables $\xi_1,\ldots,\xi_t$ have a first-order antedependence structure, then each ξ_j, $j>1$, is independent of all other $\xi_k, k<j$, given ξ_{j-1}. This Markovian structure is characterized by its inverse variance matrix, which is tridiagonal. Parameterizing an ANTE(1) structure for a random vector of size t requires $2t-1$ parameters: variances $\sigma_1^2,\ldots,\sigma_t^2$ and $t-1$ correlation parameters $\rho_1,\ldots,\rho_{t-1}$. The covariances

among random variables ξ_i and ξ_j are then constructed as

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sqrt{\sigma_i^2 \sigma_j^2} \prod_{k=i}^{j-1} \rho_k$$

PROC BGLIMM constrains the correlation parameters to satisfy $|\rho_k| < 1, \ \forall k$. For variable-order antedependence models see Macchiavelli and Arnold (1994).

AR(1)

specifies a first-order autoregressive structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \rho^{|i^* - j^*|}$$

The values i^* and j^* are derived for the *i*th and *j*th observations, respectively, and are not necessarily the observation numbers. For example, in the following statements, the values correspond to the class levels for the Time effect of the *i*th and *j*th observation within a particular subject:

```
proc bglimm;
   class time patient;
   model y = x x*x;
   random time / sub=patient type=ar(1);
run;
```

PROC BGLIMM imposes the constraint $|\rho| < 1$ for stationarity.

ARH(1)

specifies a heterogeneous first-order autoregressive structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sqrt{\sigma_i^2 \sigma_j^2}\, \rho^{|i^* - j^*|}$$

where $|\rho| < 1$. This covariance structure has the same correlation pattern as the TYPE=AR(1) structure, but the variances are allowed to differ.

ARMA(1,1)

specifies the first-order autoregressive moving average structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 & i = j \\ \sigma^2 \gamma \rho^{|i^* - j^*| - 1} & i \neq j \end{cases}$$

Here, ρ is the autoregressive parameter, γ models a moving average component, and σ^2 is a scale parameter. In the notation of Fuller (1976, p. 68), $\rho = \theta_1$ and

$$\gamma = \frac{(1 + b_1\theta_1)(\theta_1 + b_1)}{1 + b_1^2 + 2b_1\theta_1}$$

The example in Table 31.10 and $|b_1| < 1$ imply that

$$b_1 = \frac{\beta - \sqrt{\beta^2 - 4\alpha^2}}{2\alpha}$$

where $\alpha = \gamma - \rho$ and $\beta = 1 + \rho^2 - 2\gamma\rho$. PROC BGLIMM imposes the constraints $|\rho| < 1$ and $|\gamma| < 1$ for stationarity, although for some values of ρ and γ in this region, the resulting covariance matrix is not positive definite.

CS

specifies the compound symmetry structure, which has constant variance and constant covariance

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma_1 + \sigma^2 & i = j \\ \sigma_1 & i \neq j \end{cases}$$

The compound symmetry structure arises naturally with nested random effects, such as when subsampling error is nested within experimental error. Hierarchical random assignments or selections, such as subsampling or split-plot designs, give rise to compound symmetric covariance structures. This implies exchangeability of the observations in the subunit, leading to constant correlations between the observations. Compound symmetry structures are thus usually not appropriate for processes in which correlations decline according to some metric, such as spatial and temporal processes.

CSH

specifies the heterogeneous compound symmetry structure, which is an equicorrelation structure but allows for different variances,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sqrt{\sigma_i^2 \sigma_j^2} & i = j \\ \rho\sqrt{\sigma_i^2 \sigma_j^2} & i \neq j \end{cases}$$

FA(*1*)

specifies the factor-analytic structure with one factor (Jennrich and Schluchter 1986). This structure is of the form $\boldsymbol{\lambda}\boldsymbol{\lambda}' + \mathbf{D}$, where $\boldsymbol{\lambda}$ is a $t \times 1$ vector and $\mathbf{D}$ is a $t \times t$ diagonal matrix with t different parameters.

HF

specifies a covariance structure that satisfies the general Huynh-Feldt condition (Huynh and Feldt 1970). For a random vector that has t elements, this structure has $t + 1$ positive parameters and covariances

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma_i^2 & i = j \\ 0.5(\sigma_i^2 + \sigma_j^2) - \lambda & i \neq j \end{cases}$$

A covariance matrix $\boldsymbol{\Sigma}$ generally satisfies the Huynh-Feldt condition if it can be written as $\boldsymbol{\Sigma} = \boldsymbol{\tau}\mathbf{1}' + \mathbf{1}\boldsymbol{\tau}' + \lambda\mathbf{I}$. The preceding parameterization chooses $\tau_i = 0.5(\sigma_i^2 - \lambda)$. Several simpler covariance structures give rise to covariance matrices that also satisfy the Huynh-Feldt condition. For example, TYPE=CS, VC, and UN(1) are nested within TYPE=HF. Note also that the HF structure is nested within an unstructured covariance matrix.

TOEP

models a Toeplitz covariance structure. This structure can be viewed as an autoregressive structure whose order is equal to the dimension of the matrix,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 & i = j \\ \sigma_{|i-j|} & i \neq j \end{cases}$$

TOEP(*q*)

specifies a banded Toeplitz structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 & i = j \\ \sigma_{|i-j|} & |i-j| < q \end{cases}$$

This can be viewed as a moving average structure whose order is equal to $q - 1$. The specification TYPE=TOEP(1) is the same as $\sigma^2\mathbf{I}$, and it can be useful for specifying the same variance component for several effects.

TOEPH< (*q*) >

models a Toeplitz covariance structure. The correlations of this structure are banded as in the TOEP or TOEP(q) structures, but the variances are allowed to vary:

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma_i^2 & i = j \\ \rho_{|i-j|}\sqrt{\sigma_i^2\sigma_j^2} & i \neq j \end{cases}$$

The correlation parameters satisfy $|\rho_{|i-j|}| < 1$. If you specify the optional value q, the correlation parameters with $|i-j| \geq q$ are set to 0, creating a banded correlation structure. The specification TYPE=TOEPH(1) results in a diagonal covariance matrix with heterogeneous variances.

UN< (*q*) >

specifies a completely general (unstructured) covariance matrix that is parameterized directly in terms of variances and covariances,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma_{ij}$$

The variances are constrained to be nonnegative, and the covariances are unconstrained. This structure is constrained to be nonnegative definite. If you specify the order parameter q, then PROC BGLIMM estimates only the first q bands of the matrix, setting elements in all higher bands equal to 0.

VC

specifies standard variance components and is the default structure for both G-side and R-side covariance structures. In a G-side covariance structure, a distinct variance component is assigned to each effect. In an R-side structure, TYPE=VC is usually used only to add overdispersion effects or, with the GROUP= option, to specify a heterogeneous variance model.

Table 31.10 lists some examples of the structures in Table 31.9.

Table 31.10 Covariance Structure Examples

Description	Structure	Example
First-order antedependence	ANTE(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_1 & \sigma_1\sigma_3\rho_1\rho_2 \\ \sigma_2\sigma_1\rho_1 & \sigma_2^2 & \sigma_2\sigma_3\rho_2 \\ \sigma_3\sigma_1\rho_2\rho_1 & \sigma_3\sigma_2\rho_2 & \sigma_3^2 \end{bmatrix}$
First-order autoregressive	AR(1)	$\sigma^2\begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 \\ \rho & 1 & \rho & \rho^2 \\ \rho^2 & \rho & 1 & \rho \\ \rho^3 & \rho^2 & \rho & 1 \end{bmatrix}$

Table 31.10 *continued*

Description	Structure	Example
Heterogeneous AR(1)	ARH(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho^2 & \sigma_1\sigma_4\rho^3 \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho^2 \\ \sigma_3\sigma_1\rho^2 & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho^3 & \sigma_4\sigma_2\rho & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$
First-order autoregressive moving average	ARMA(1,1)	$\sigma^2 \begin{bmatrix} 1 & \gamma & \gamma\rho & \gamma\rho^2 \\ \gamma & 1 & \gamma & \gamma\rho \\ \gamma\rho & \gamma & 1 & \gamma \\ \gamma\rho^2 & \gamma\rho & \gamma & 1 \end{bmatrix}$
Compound symmetry	CS	$\begin{bmatrix} \sigma_1+\sigma^2 & \sigma_1 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1+\sigma^2 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma_1+\sigma^2 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma_1 & \sigma_1+\sigma^2 \end{bmatrix}$
Heterogeneous compound symmetry	CSH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho & \sigma_1\sigma_4\rho \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho \\ \sigma_3\sigma_1\rho & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho & \sigma_4\sigma_2\rho & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$
First-order factor analytic	FA(1)	$\begin{bmatrix} \lambda_1^2+d_1 & \lambda_1\lambda_2 & \lambda_1\lambda_3 & \lambda_1\lambda_4 \\ \lambda_2\lambda_1 & \lambda_2^2+d_2 & \lambda_2\lambda_3 & \lambda_2\lambda_4 \\ \lambda_3\lambda_1 & \lambda_3\lambda_2 & \lambda_3^2+d_3 & \lambda_3\lambda_4 \\ \lambda_4\lambda_1 & \lambda_4\lambda_2 & \lambda_4\lambda_3 & \lambda_4^2+d_4 \end{bmatrix}$
Huynh-Feldt	HF	$\begin{bmatrix} \sigma_1^2 & \frac{\sigma_1^2+\sigma_2^2}{2}-\lambda & \frac{\sigma_1^2+\sigma_3^2}{2}-\lambda \\ \frac{\sigma_2^2+\sigma_1^2}{2}-\lambda & \sigma_2^2 & \frac{\sigma_2^2+\sigma_3^2}{2}-\lambda \\ \frac{\sigma_3^2+\sigma_1^2}{2}-\lambda & \frac{\sigma_3^2+\sigma_2^2}{2}-\lambda & \sigma_3^2 \end{bmatrix}$
Toeplitz	TOEP	$\begin{bmatrix} \sigma^2 & \sigma_1 & \sigma_2 & \sigma_3 \\ \sigma_1 & \sigma^2 & \sigma_1 & \sigma_2 \\ \sigma_2 & \sigma_1 & \sigma^2 & \sigma_1 \\ \sigma_3 & \sigma_2 & \sigma_1 & \sigma^2 \end{bmatrix}$
Toeplitz with two bands	TOEP(2)	$\begin{bmatrix} \sigma^2 & \sigma_1 & 0 & 0 \\ \sigma_1 & \sigma^2 & \sigma_1 & 0 \\ 0 & \sigma_1 & \sigma^2 & \sigma_1 \\ 0 & 0 & \sigma_1 & \sigma^2 \end{bmatrix}$
Heterogeneous Toeplitz	TOEPH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_1 & \sigma_1\sigma_3\rho_2 & \sigma_1\sigma_4\rho_3 \\ \sigma_2\sigma_1\rho_1 & \sigma_2^2 & \sigma_2\sigma_3\rho_1 & \sigma_2\sigma_4\rho_2 \\ \sigma_3\sigma_1\rho_2 & \sigma_3\sigma_2\rho_1 & \sigma_3^2 & \sigma_3\sigma_4\rho_1 \\ \sigma_4\sigma_1\rho_3 & \sigma_4\sigma_2\rho_2 & \sigma_4\sigma_3\rho_1 & \sigma_4^2 \end{bmatrix}$
Unstructured	UN	$\begin{bmatrix} \sigma_1^2 & \sigma_{21} & \sigma_{31} & \sigma_{41} \\ \sigma_{21} & \sigma_2^2 & \sigma_{32} & \sigma_{42} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 & \sigma_{43} \\ \sigma_{41} & \sigma_{42} & \sigma_{43} & \sigma_4^2 \end{bmatrix}$

Table 31.10 *continued*

Description	Structure	Example
Banded main diagonal	UN(1)	$\begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & 0 \\ 0 & 0 & 0 & \sigma_4^2 \end{bmatrix}$
Variance components	VC (default)	$\begin{bmatrix} \sigma_B^2 & 0 & 0 & 0 \\ 0 & \sigma_B^2 & 0 & 0 \\ 0 & 0 & \sigma_{AB}^2 & 0 \\ 0 & 0 & 0 & \sigma_{AB}^2 \end{bmatrix}$

REPEATED Statement

REPEATED *repeated-effect* < / *options* > **;**

The REPEATED statement specifies the **R** matrix in the model. Its syntax is similar to that of the REPEATED statement in PROC MIXED. If you omit this statement, **R** is assumed to be equal to $\sigma^2\mathbf{I}$. The REPEATED statement is available only for the normal distribution with the identity link in this release.

Specifying a *repeated-effect* is required in order to inform PROC BGLIMM of the proper location of the observed repeated responses. The *repeated-effect* must contain only classification variables. You specify the SUBJECT= option to identify repeated measures for the same subject and set up the blocks of **R**. You can use the TYPE= option to define the covariance structure. The levels of the *repeated-effect* must be different for each observation within a subject; otherwise, PROC BGLIMM produces an error message.

Table 31.11 summarizes the *options* available in the REPEATED statement. All *options* are then discussed in alphabetical order.

Table 31.11 Summary of REPEATED Statement Options

Option	Description
Construction of Covariance Structure	
COVALG=	Specifies the sampling algorithm for the **R** matrix
COVPRIOR=	Specifies the prior for the **R** matrix
GROUP=	Defines an effect that specifies heterogeneity in the **R** covariance structure
SUBJECT=	Identifies the subjects in the R-side model
TYPE=	Specifies the type of the **R** matrix
Statistical Output	
NOPRINTCOV	Suppresses displaying the results for the covariance matrix parameters
R	Displays the estimated **R** matrix
RCORR	Display the correlation matrix that corresponds to the estimated **R** matrix

You can specify the following *options* in the REPEATED statement after a slash (/):

COVALG=<(*algorithms*)>

specifies the sampling algorithm that is used to sample the parameters in the **R** matrix of the residuals, when conjugate sampling is unavailable. For more information, see the section "Sampling Methods" on page 1341. You can specify one of the following sampling *algorithms*:

NUTS<(*nuts-options*)>

applies the No-U-Turn Sampler (NUTS) of the Hamiltonian algorithm to sample the parameters in the **R** matrix of the residuals. The NUTS algorithm is a version of the adaptive Hamiltonian Monte Carlo algorithm with automatic tuning of the step size and number of steps in each iteration. For more information, see the section "Hamiltonian Monte Carlo Sampler" on page 1343. You can specify the following *nuts-options*:

DELTA=*value*

specifies the target acceptance rate during the tuning process of the NUTS algorithm. Increasing the *value* can often improve mixing, but it can also significantly slow down the sampling. By default, DELTA=0.6.

MAXHEIGHT=*value*

specifies the maximum height of the NUTS algorithm tree. The taller the tree, the more gradient evaluations per iteration the procedure calculates. The number of evaluations is 2^{height}. Usually, the height of a tree should be no more than 10 during the sampling stage, but it can go higher during the tuning stage. A larger number indicates that the algorithm is having difficulty converging. By default, MAXHEIGHT=10.

NOCONJCOV

uses the NUTS algorithm, instead of the conjugate sampler, on the scale parameter of the **R** matrix when the covariance type is AR(1) or ARMA(1,1).

NTU=*n*

specifies the number of tuning iterations for the NUTS algorithm to use. By default, *n* is the same as the NBI= option value that you specify in the PROC BGLIMM statement.

STEPSIZE=*value*

specifies the initial step size in the NUTS algorithm. By default, STEPSIZE=0.1.

SLICE<(*option*)>

applies the slice sampling algorithm to sample the parameters in the **R** matrix of the residuals. Slice sampling is a Markov chain Monte Carlo (MCMC) method that can be applied to sample any continuous parameter, as long as its unnormalized conditional distribution can be evaluated at any parameter value. For more information, see the section "Slice Sampler" on page 1344. You can specify the following *option*:

NOCONJCOV

uses the slice sampler, instead of the conjugate sampler, on the scale parameter of the **R** matrix when the covariance type is AR(1) or ARMA(1,1).

These options are similar to those in the RANDOM statement. By default, COVALG=SLICE.

COVPRIOR=*prior-distribution*

specifies a prior distribution for the **R** matrix of the residuals, when the **R** matrix is of the UN, UN(1), VC, or TOEP(1) type, where a conjugate sampler is used. This option is ignored for other covariance types, because you assign a flat prior to the **R** matrix if its type is not UN, UN(1), VC, or TOEP(1).

You can specify one of the following *prior-distributions*:

HALFCAUCHY < (SCALE=*a***) >**

HCAUCHY < (SCALE=*a***) >**

HC < (SCALE=*a***) >**

specifies the prior to be a half-Cauchy distribution. The half-Cauchy prior is applied only to the diagonal terms (variances) of the **R** matrix. The off-diagonal terms of the **R** matrix are assumed to have a flat prior.

You can specify the scale parameter *a* of the half-Cauchy distribution. The scale parameter *a* must be positive. By default, SCALE=25.

HALFNORMAL < (VAR=*a***) >**

HNORMAL < (VAR=*a***) >**

HN < (VAR=*a***) >**

specifies the prior to be a half-normal distribution. The half-normal prior is applied only to the diagonal terms (variances) of the **R** matrix. The off-diagonal terms of the **R** matrix are assumed to have a flat prior.

You can specify the variance *a* of the half-normal distribution. The variance *a* must be positive. By default, VAR=25.

IGAMMA < (*options***) >**

specifies an inverse gamma prior, $\mathrm{IG}(a, b)$, with the density $f(t) = \frac{b^a}{\Gamma(a)} t^{-(a+1)} \mathrm{e}^{-b/t}$ for each diagonal term of the **R** matrix. It is the default prior for the covariance types UN(1), VC, and TOEP(1).

You can set the parameters for the inverse gamma distribution by specifying one or both of the following *options*, separated by a comma or a space:

SHAPE=*a*

specifies the shape parameter *a* of the inverse gamma distribution. By default, SHAPE=2.

SCALE=*b*

specifies the scale parameter *b* of the inverse gamma distribution. By default, SCALE=2.

IWISHART < (*options***) >**

IWISH < (*options***) >**

IW < (*options***) >**

specifies an inverse Wishart prior, $\mathrm{IWISHART}(a, b)$, for the **R** matrix. It is the default prior for the UN covariance type.

You can set the parameters of the inverse Wishart distribution by specifying one or both of the following *options*, separated by a comma or a space:

DF=*a*
specifies the degrees of freedom *a* of the inverse Wishart distribution. The default is the dimension of the **R** matrix plus 3.

SCALE=*b*
specifies $b\mathbf{I}$ as the scale matrix of the inverse Wishart distribution, where **I** is the identity matrix. The default is the dimension of the **R** matrix plus 3.

SIWISHART < (*options*) >
SIWISH < (*options*) >
SIW < (*options*) >
specifies a scaled inverse Wishart prior for the **R** matrix.

You can set the parameters of the scaled inverse Wishart distribution by specifying one or more of the following *options*, separated by a comma or a space:

DF=*a*
specifies the degrees of freedom *a* of the scaled inverse Wishart distribution. The default is the dimension of the **R** matrix plus 3.

SCALE=*b*
specifies $b\mathbf{I}$ as the scale matrix of the scaled inverse Wishart distribution, where **I** is the identity matrix. The default is the dimension of the **R** matrix plus 3.

VAR=*c*
specifies the variance parameter *c* of the normal prior for $\log(\delta_i)$. By default, VAR=1.

UNIFORM < (*options*) >
UNIF < (*options*) >
specifies a uniform prior, $\mathrm{UNIFORM}(a, b)$, for the **R** matrix. The uniform prior is applied to standard deviations (the square root of the diagonal terms) of the **R** matrix.

You can set the lower and upper bounds of the uniform distribution by specifying one or both of the following *options*, separated by a comma or a space:

LOWER=*a*
specifies the lower bound *a* of the uniform distribution. The lower bound must be nonnegative. By default, LOWER=0.

UPPER=*b*
specifies the upper bound *b* of the uniform distribution. The upper bound must be positive. By default, UPPER=1E10.

GROUP=*effect*
GRP=*effect*
defines an effect that specifies heterogeneity in the covariance structure of **R**. All observations that have the same level of the *effect* have the same covariance parameters. Each new level of the *effect* produces a new set of covariance parameters that has the same structure as the original group. You should exercise caution in properly defining the *effect*, because strange covariance patterns can result when it is misused. The *effect* can greatly increase the number of estimated covariance parameters.

The GROUP= *effect* must be specified as a variable in the CLASS statement.

NOPRINTCOV

suppresses displaying results (summary statistics, diagnostics, and plots) for the parameters for the covariance matrix. In models that have many repeated measurements, the number of parameters involved in the covariance matrix can be very large and then it can take a long time to display the summary, diagnostic statistics, and plots for all these parameters. By default, this option is off.

R

displays the estimated $\mathbf{R}$ matrix. The ODS table name is R.

RCORR

produces the correlation matrix that corresponds to the estimated $\mathbf{R}$ matrix. The ODS table name is RCorr.

SUBJECT=*effect*

SUB=*effect*

identifies the subjects for the blocking structure in $\mathbf{R}$. The repeated measures for the same subject must be together in the data. Complete independence is assumed across subjects; therefore, this option produces a block diagonal structure in $\mathbf{R}$ that has identical blocks. The *effect* must be specified as a categorical variable in the CLASS statement.

TYPE=*covariance-structure*

specifies the covariance structure of the $\mathbf{R}$ matrix. The SUBJECT= option defines the blocks of $\mathbf{R}$, and the TYPE= option specifies the structure of these blocks. Valid values for *covariance-structure* and their descriptions are provided in Table 31.9, and some examples are shown in Table 31.10. By default, TYPE=VC.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement identifies a *variable* in the input data set to provide a value to rescale each observation in a data set: the scale parameter is replaced by ϕ/w_i in the density, where w_i is the weight that is associated with the observation y_i. When there is not a scale parameter in the response distribution, w_i behaves the same as the frequency f_i by treating an observation as if it appears w_i times.

Observations that have relatively large weights have more influence in the analysis than observations that have smaller weights. An unweighted analysis is the same as a weighted analysis in which all weights are 1.

Observations with nonpositive or missing weights are not used in the analysis. If a WEIGHT statement is not included, all observations included in the analysis are assigned a weight of 1. The WEIGHT variable does not have to be an integer; it can have decimals.

Details: BGLIMM Procedure

Generalized Linear Mixed Models

First consider the simplest model: a normal linear model. The quantity of primary interest, y_i, is called the response or outcome variable for the ith individual. The variable $\mathbf{x}_i$ is the $1 \times p$ covariate vector for the fixed effects. The distribution of y_i given $\mathbf{x}_i$ is normal with a mean that is a linear function of $\mathbf{x}_i$,

$$y_i = \mathbf{x}_i \boldsymbol{\beta} + \epsilon_i, \quad i = 1, \ldots, I$$
$$\epsilon_i \sim N(\mathbf{0}, \sigma^2)$$

where $\boldsymbol{\beta}$ is a $p \times 1$ vector of regression coefficients (also known as fixed effects) and ϵ_i is the noise with a variance σ^2.

The normal linear model can be expanded to include random effects, and the model becomes a normal linear mixed model,

$$y_i = \mathbf{x}_i \boldsymbol{\beta} + \mathbf{z}_i \boldsymbol{\gamma}_i + \epsilon_i$$
$$\boldsymbol{\gamma}_i \sim N(\mathbf{0}, \mathbf{G}_i)$$
$$\epsilon_i \sim N(0, \sigma^2)$$

where $\boldsymbol{\gamma}_i$ is a $q \times 1$ vector of random effects, $\mathbf{z}_i$ is an $1 \times q$ matrix of covariates for the $\boldsymbol{\gamma}_i$, and $\mathbf{G}_i$ is the covariance matrix of the random effects $\boldsymbol{\gamma}_i$ ($\mathbf{G}$ is a block diagonal matrix where each block is $\mathbf{G}_i$).

When an individual i has n_i repeated measurements, the random-effects model for outcome vector $\mathbf{y}_i$ is given by

$$\mathbf{y}_i = \mathbf{X}_i \boldsymbol{\beta} + \mathbf{Z}_i \boldsymbol{\gamma}_i + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, I$$

where $\mathbf{y}_i$ is $n_i \times 1$, $\mathbf{X}_i$ is an $n_i \times p$ matrix of fixed covariates, $\boldsymbol{\beta}$ is a $p \times 1$ vector of regression coefficients (also known as fixed effects), $\boldsymbol{\gamma}_i$ is a $q \times 1$ vector of random effects, $\mathbf{Z}_i$ is an $n_i \times q$ matrix of covariates for the $\boldsymbol{\gamma}_i$, and $\boldsymbol{\epsilon}_i$ is an $n_i \times 1$ vector of random errors.

It is further assumed that

$$\boldsymbol{\gamma}_i \sim N(\mathbf{0}, \mathbf{G}_i)$$
$$\boldsymbol{\epsilon}_i \sim N(\mathbf{0}, \mathbf{R}_i)$$

where $\mathbf{G}_i$ is the covariance matrix of $\boldsymbol{\gamma}_i$ ($\mathbf{G}$ is a block diagonal matrix where each block is $\mathbf{G}_i$) and $\mathbf{R}_i$ is the covariance matrix of the residual errors for the ith subject ($\mathbf{R}$ is a block diagonal matrix where each block is $\mathbf{R}_i$).

There are cases where the relationship between the design matrix ($\mathbf{X}$ and $\mathbf{Z}$) and the expectation of the response is not linear, or where the distribution for the response is far from normal, even after transformation of the data. The class of generalized linear mixed models unifies the approaches that you need in order to analyze data in those cases. Let $\mathbf{y}$ be the collection of all $\mathbf{y}_i$, and let $\mathbf{X}$ and $\mathbf{Z}$ be the collection of all $\mathbf{X}_i$ and $\mathbf{Z}_i$, respectively. A generalized linear mixed model consists of the following:

- the linear predictor $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}$

- the link function $g(\cdot)$ that relates the linear predictor to the mean of the outcome via a monotone link function,

  $$\mathrm{E}[Y|\boldsymbol{\beta}, \boldsymbol{\gamma}] = g^{-1}(\boldsymbol{\eta}) = g^{-1}(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma})$$

  where $g(\cdot)$ is a differentiable monotone link function and $g^{-1}(\cdot)$ is its inverse.

- a response distribution in the exponential family of distributions. The distribution can also depend on a scale parameter, ϕ.

A density or mass function in the exponential family can be written as

$$f(y) = \exp\left\{\frac{y\theta - b(\theta)}{\phi} + c(y, f(\phi))\right\}$$

for some functions $b(\cdot)$ and $c(\cdot)$. The parameter θ is called the natural (canonical) parameter. The parameter ϕ is a scale parameter, and it is not present in all exponential family distributions. For example, in logistic regression and Poisson regression, $\phi = 1$.

The mean and variance of the data are related to the components of the density, $\mathrm{E}[Y] = \mu = b'(\theta)$, $\mathrm{Var}[Y] = \phi b''(\theta)$, where primes denote first and second derivatives. If you express θ as a function of μ, the relationship is known as the natural link function or the canonical link function. In other words, modeling data by using a canonical link assumes that $\theta = \mathbf{x}\boldsymbol{\beta} + \mathbf{z}\boldsymbol{\gamma}$; the effect contributions are additive on the canonical scale. The second derivative of $b(\cdot)$, expressed as a function of μ, is the variance function of the generalized linear model, $a(\mu) = b''(\theta(\mu))$. Note that because of this relationship, the distribution determines the variance function and the canonical link function. However, you cannot proceed in the opposite direction.

Likelihood-based inference is based on the marginal likelihood in which the random effects are integrated out. The integration requires numerical methods, such as Gaussian quadrature methods (which can be computationally costly) or Laplace methods (which are faster but not as accurate). The Bayesian approach estimates the joint distribution of all parameters in the model, and it is made possible by the Markov chain Monte Carlo (MCMC) methods. The presence of the random-effects parameters $\boldsymbol{\gamma}$ adds an extra sampling step to the Gibbs algorithm, thus eliminating the need to numerically integrate out $\boldsymbol{\gamma}$ to make inferences about $\boldsymbol{\beta}$. The MCMC methods produce marginal distribution estimates of all fixed-effects parameters, include the **G** and **R** covariance matrices, making estimation convenient.

Response Probability Distributions

Probability distributions of the response *y* in generalized linear models are usually parameterized in terms of the mean μ and dispersion parameter ϕ instead of the *natural parameter* θ. The probability distributions that are available in the BGLIMM procedure are shown in the following list. The PROC BGLIMM scale parameter and the variance of *y* are also shown.

- Beta:

$$
\begin{aligned}
f(y) &= \frac{\Gamma(p+q)}{\Gamma(p)\Gamma(q)} y^{p-1}(1-y)^{q-1} \quad \text{for } 0 < y < 1 \\
\mu &= \frac{p}{p+q} \\
\phi &= p+q \\
\text{scale} &= p+q \\
\mathrm{Var}(Y) &= \frac{pq}{(p+q)^2(p+q+1)}
\end{aligned}
$$

- Binary:

$$
\begin{aligned}
f(y) &= \mu^y(1-\mu)^{1-y} \quad \text{for } y = 0, 1 \\
\phi &= 1 \\
\mathrm{Var}(Y) &= \mu(1-\mu)
\end{aligned}
$$

- Binomial:

$$
\begin{aligned}
f(y) &= \binom{n}{r} \mu^r (1-\mu)^{n-r} \quad \text{for } y = \frac{r}{n},\ r = 0, 1, 2, \ldots, n \\
\phi &= 1 \\
\mathrm{Var}(Y) &= \frac{\mu(1-\mu)}{n}
\end{aligned}
$$

- Exponential:

$$
\begin{aligned}
f(y) &= \frac{1}{\mu} \exp\left(-\frac{y}{\mu}\right) \quad \text{for } 0 < y < \infty \\
\phi &= 1 \\
\mathrm{Var}(Y) &= \mu^2
\end{aligned}
$$

- Gamma:

$$
\begin{aligned}
f(y) &= \frac{1}{\Gamma(\nu)y} \left(\frac{y\nu}{\mu}\right)^{\nu} \exp\left(-\frac{y\nu}{\mu}\right) \quad \text{for } 0 < y < \infty \\
\phi &= \nu^{-1} \\
\text{scale} &= \nu \\
\mathrm{Var}(Y) &= \frac{\mu^2}{\nu}
\end{aligned}
$$

- Geometric: This is a special case of the negative binomial where $k = 1$.

$$\begin{aligned} f(y) &= \frac{(\mu)^y}{(1+\mu)^{y+1}} \quad \text{for } y = 0, 1, 2, \ldots \\ \phi &= 1 \\ \mathrm{Var}(Y) &= \mu(1+\mu) \end{aligned}$$

- Inverse Gaussian:

$$\begin{aligned} f(y) &= \frac{1}{\sqrt{2\pi y^3}\sigma} \exp\left[-\frac{1}{2y}\left(\frac{y-\mu}{\mu\sigma}\right)^2\right] \quad \text{for } 0 < y < \infty \\ \phi &= \sigma^2 \\ \text{scale} &= \sigma^2 \\ \mathrm{Var}(Y) &= \sigma^2\mu^3 \end{aligned}$$

- Multinomial:

$$\begin{aligned} f(y_1, y_2, \ldots, y_k) &= \frac{m!}{y_1! y_2! \cdots y_k!} p_1^{y_1} p_2^{y_2} \cdots p_k^{y_k} \\ \phi &= 1 \end{aligned}$$

- Negative binomial:

$$\begin{aligned} f(y) &= \frac{\Gamma(y+1/k)}{\Gamma(y+1)\Gamma(1/k)} \frac{(k\mu)^y}{(1+k\mu)^{y+1/k}} \quad \text{for } y = 0, 1, 2, \ldots \\ \phi &= 1 \\ \text{dispersion} &= k \\ \mathrm{Var}(Y) &= \mu + k\mu^2 \end{aligned}$$

- Normal (Gaussian):

$$\begin{aligned} f(y) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right] \quad \text{for } -\infty < y < \infty \\ \phi &= \sigma^2 \\ \text{scale} &= \sigma^2 \\ \mathrm{Var}(Y) &= \sigma^2 \end{aligned}$$

- Poisson:

$$
\begin{aligned}
f(y) &= \frac{\mu^y \mathrm{e}^{-\mu}}{y!} \quad \text{for } y = 0, 1, 2, \ldots \\
\phi &= 1 \\
\mathrm{Var}(Y) &= \mu
\end{aligned}
$$

The negative binomial and zero-inflated negative binomial distributions contain a parameter k, called the negative binomial dispersion parameter. This is not the same as the generalized linear model dispersion ϕ; rather, it is an additional distribution parameter that must be estimated or set to a fixed value.

For the binomial distribution, the response is the binomial proportion $y = events/trials$. The variance function is $V(\mu) = \mu(1-\mu)$, and the binomial trials parameter n is regarded as a weight w.

Likelihood

The BGLIMM procedure forms the log likelihoods of generalized linear models as

$$
L(\boldsymbol{\mu}, \phi; \mathbf{y}, f_i, w_i) = \sum_{i=1}^{n} f_i \, l(\mu_i, \phi; y_i, w_i)
$$

where $l(\mu_i, \phi; y_i)$ is the log-likelihood contribution of the ith observation with weight w_i and f_i is the value of the frequency variable. In the case where observations have weights, the scale parameter is replaced by ϕ/w_i in the density, where w_i is the weight that is associated with the observation y_i. For the determination of w_i and f_i, see the WEIGHT and FREQ statements.

The individual log-likelihood contributions for the various distributions are as follows:

Beta:

$$
\begin{aligned}
l(\mu_i, \phi; y_i, w_i) = \log &\left\{ \frac{\Gamma(\phi/w_i)}{\Gamma(\mu_i \phi/w_i)\Gamma((1-\mu_i)\phi/w_i)} \right\} \\
&+ (\mu_i \phi/w_i - 1) \log\{y_i\} \\
&+ ((1-\mu_i)\phi/w_i - 1) \log\{1 - y_i\}
\end{aligned}
$$

$\mathrm{Var}[Y] = \mu(1-\mu)/(1+\phi)$; $\phi > 0$. See Ferrari and Cribari-Neto (2004).

Binary:

$$
l(\mu_i, \phi; y_i, w_i) = w_i (y_i \log\{\mu_i\} + (1 - y_i) \log\{1 - \mu_i\})
$$

$\mathrm{Var}[Y] = \mu(1-\mu)$; $\phi \equiv 1$.

Binomial:

$$
\begin{aligned}
l(\mu_i, \phi; y_i, w_i) = &\; w_i \, (y_i \log\{\mu_i\} + (n_i - y_i) \log\{1 - \mu_i\}) \\
&+ w_i \, (\log\{\Gamma(n_i + 1)\} - \log\{\Gamma(y_i + 1)\} - \log\{\Gamma(n_i - y_i + 1)\})
\end{aligned}
$$

where y_i and n_i are the *events* and *trials* in the *events/trials* syntax, and $0 < \mu < 1$. $\mathrm{Var}[Y/n] = \mu(1-\mu)/n$; $\phi \equiv 1$.

Exponential:

$$l(\mu_i, \phi; y_i, w_i) = \begin{cases} -\log\{\mu_i\} - y_i/\mu_i & w_i = 1 \\ w_i \log\left\{\frac{w_i y_i}{\mu_i}\right\} - \frac{w_i y_i}{\mu_i} - \log\{y_i \Gamma(w_i)\} & w_i \neq 1 \end{cases}$$

$\mathrm{Var}[Y] = \mu^2$; $\phi \equiv 1$.

Gamma:

$$l(\mu_i, \phi; y_i, w_i) = w_i \phi \log\left\{\frac{w_i y_i \phi}{\mu_i}\right\} - \frac{w_i y_i \phi}{\mu_i} - \log\{y_i\} - \log\{\Gamma(w_i \phi)\}$$

$\mathrm{Var}[Y] = \phi\mu^2$; $\phi > 0$.

Geometric:

$$l(\mu_i, \phi; y_i, w_i) = y_i \log\left\{\frac{\mu_i}{w_i}\right\} - (y_i + w_i) \log\left\{1 + \frac{\mu_i}{w_i}\right\} + \log\left\{\frac{\Gamma(y_i + w_i)}{\Gamma(w_i)\Gamma(y_i + 1)}\right\}$$

$\mathrm{Var}[Y] = \mu + \mu^2$; $\phi \equiv 1$.

Inverse Gaussian:

$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left[\frac{w_i (y_i - \mu_i)^2}{y_i \phi \mu_i^2} + \log\left\{\frac{\phi y_i^3}{w_i}\right\} + \log\{2\pi\}\right]$$

$\mathrm{Var}[Y] = \phi\mu^3$; $\phi > 0$.

Multinomial:

$$l(\boldsymbol{\mu}_i, \phi; \mathbf{y}_i, w_i) = w_i \sum_{j=1}^{J} y_{ij} \log\{\mu_{ij}\}$$

$\phi \equiv 1$.

Negative binomial:

$$l(\mu_i, \phi; y_i, w_i) = y_i \log\left\{\frac{k\mu_i}{w_i}\right\} - (y_i + w_i/k) \log\left\{1 + \frac{k\mu_i}{w_i}\right\} + \log\left\{\frac{\Gamma(y_i + w_i/k)}{\Gamma(w_i/k)\Gamma(y_i + 1)}\right\}$$

$\mathrm{Var}[Y] = \mu + k\mu^2$; $k > 0$; $\phi \equiv 1$.

For a given k, the negative binomial distribution is a member of the exponential family. The parameter k is related to the scale of the data, because it is part of the variance function. However, it cannot be factored from the variance, as is the case with the ϕ parameter in many other distributions. The parameter k is designated as "Scale" in the output of PROC BGLIMM.

Normal (Gaussian):

$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{\phi} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\}\right]$$

$$\mathrm{Var}[Y] = \phi;\ \phi > 0.$$

Poisson:

$$l(\mu_i, \phi; y_i, w_i) = w_i(y_i \log\{\mu_i\} - \mu_i - \log\{\Gamma(y_i + 1)\})$$

$$\mathrm{Var}[Y] = \mu;\ \phi \equiv 1.$$

Define the parameter vector for the generalized linear model as $\boldsymbol{\theta} = \boldsymbol{\beta}$, if $\phi \equiv 1$, and as $\boldsymbol{\theta} = [\boldsymbol{\beta}', \phi]'$ otherwise. $\boldsymbol{\beta}$ denotes the fixed-effects parameters in the linear predictor. For the negative binomial distribution, the relevant parameter vector is $\boldsymbol{\theta} = [\boldsymbol{\beta}', k]'$. The gradient and Hessian of the negative log likelihood are then

$$\mathbf{g} = -\frac{\partial L(\boldsymbol{\theta}; \mathbf{y})}{\partial \boldsymbol{\theta}} \qquad \mathbf{H} = -\frac{\partial^2 L(\boldsymbol{\theta}; \mathbf{y})}{\partial \boldsymbol{\theta}\, \partial \boldsymbol{\theta}'}$$

Scale and Dispersion Parameters

The parameter ϕ in the log-likelihood functions is a scale parameter. McCullagh and Nelder (1989, p. 29) refer to it as the dispersion parameter. With the exception of the normal distribution, ϕ does not correspond to the variance of an observation; the variance of an observation in a generalized linear model is a function of ϕ and μ. In a generalized linear model, the BGLIMM procedure displays the estimate of ϕ as "Scale" in the "Posterior Summaries and Intervals" table. Note that the scale parameter is the same as that reported by the GLIMMIX procedure for almost all distributions, but it is different from that reported by the GENMOD procedure for some distributions of this scale (see the "Parameter Estimates" table in PROC GLIMMIX and PROC GENMOD). The scale that is reported by PROC GENMOD is sometimes a transformation of the dispersion parameter in the log-likelihood function. Table 31.11 displays the "Scale" entries that are reported by the three procedures in terms of the ϕ (or k) parameter.

Table 31.11 Scale Reported in Output Table

Distribution	PROC BGLIMM	PROC GLIMMIX	PROC GENMOD
Beta	$\widehat{\phi}$	$\widehat{\phi}$	N/A
Gamma	$1/\widehat{\phi}$	$\widehat{\phi}$	$1/\widehat{\phi}$
Inverse Gaussian	$\widehat{\phi}$	$\widehat{\phi}$	$\sqrt{\widehat{\phi}}$
Negative binomial	$\widehat{k}$	$\widehat{k}$	$\widehat{k}$
Normal	$\widehat{\phi} = \widehat{\mathrm{Var}}[y]$	$\widehat{\phi} = \widehat{\mathrm{Var}}[y]$	$\sqrt{\widehat{\phi}}$

How PROC BGLIMM Works

PROC BGLIMM is a simulation-based procedure that uses a variety of sampling algorithms to draw samples from the joint posterior distribution of parameters from a generalized linear mixed model (GLMM). Sampling methods include the conjugate sampler, direct sampler, Gamerman algorithm (a variation of the Metropolis-Hastings algorithm that is tailored to generalized linear models), and No-U-Turn Sampler (NUTS, a self-tuning variation of the Hamiltonian Monte Carlo (HMC) method).

For situations in which the conjugate samplers are used, see the section "Conjugate Sampling" on page 1341. The direct sampling method is used for missing values, where the sampling distribution is known. The Gamerman algorithm is used for both the fixed-effects and random-effects parameters in nonnormal models. The NUTS algorithm is used for covariance parameters when conjugacy is not available.

PROC BGLIMM updates parameters conditionally, through Gibbs sampling. The fixed-effects parameters $\boldsymbol{\beta}$ are drawn jointly at each iteration. The random-effect parameters (in a RANDOM statement) are updated by clusters, unless the SUBJECT= option is not specified. In that situation, the random-effects parameters from the same RANDOM statement are updated jointly (for more information about how the random-effects parameters can be parameterized differently with or without the presence of the SUBJECT= option, see the section "Treatment of Subjects in the RANDOM Statement" on page 1348). Missing data values are updated in sequence, and the G-side and the R-side covariance parameters are updated separately, in their full posterior conditionals.

The rest of this section describes how PROC BGLIMM computes the full conditional distributions in the Gibbs updating. Let $\boldsymbol{\theta} = \{\boldsymbol{\beta}, \mathbf{G}, \mathbf{R}\}$, the collection of all fixed-effects parameters and the covariance matrices; let $\boldsymbol{\gamma}$ denote random-effects parameters and $\boldsymbol{\gamma}_j$ denote the random-effects parameters from cluster j. For simplicity, it is assumed that there is only one random effect, thus omitting an extra subindex for $\boldsymbol{\gamma}$. The treatment of random effects is identical for effects in multiple RANDOM statements.

GLM Parameters

In generalized linear models that have only fixed effects, the log of the posterior density is

$$\log(p(\boldsymbol{\beta}|\mathbf{y}, \mathbf{R})) = \log(\pi(\boldsymbol{\beta})) + \sum_{i=1}^{n} \log(f(\mathbf{y}_i|\boldsymbol{\beta}, \mathbf{R}))$$

where $\boldsymbol{\beta}$ is the vector of fixed-effects parameters and $\log(\pi(\boldsymbol{\beta}))$ is the log of the joint prior density of $\boldsymbol{\beta}$. The log likelihood, $\log(f(\mathbf{y}_i|\boldsymbol{\beta}, \mathbf{R}))$, is computed for the ith observation. The summation reflects the assumption that all observations in the data set are independent. The response variable $\mathbf{y}_i$ can be a vector, and $\mathbf{R}$ can be either a scalar or a covariance. The logarithm of the prior distribution of $\mathbf{R}$ is not included because it is constant with respect to $\boldsymbol{\beta}$.

The objective function of $\mathbf{R}$ is similar to that of $\boldsymbol{\beta}$:

$$\log(p(\mathbf{R}|\mathbf{y}, \boldsymbol{\beta})) = \log(\pi(\mathbf{R})) + \sum_{i=1}^{n} \log(f(\mathbf{y}_i|\boldsymbol{\beta}, \mathbf{R}))$$

GLMM with Random Effects

In a random-effects model, the conditional distribution of $\boldsymbol{\beta}$ is similar to that of the fixed-effects-only model,

$$\log(p(\boldsymbol{\beta}|\boldsymbol{\gamma}, \mathbf{y}, \mathbf{R})) = \log(\pi(\boldsymbol{\beta})) + \sum_{i=1}^{n} \log(f(\mathbf{y}_i|\boldsymbol{\beta}, \boldsymbol{\gamma}, \mathbf{R}))$$

where the log-likelihood function now includes the random effects $\boldsymbol{\gamma}$. This construction reflects two PROC BGLIMM modeling settings: all random-effects parameters enter the likelihood function (linearly at the mean level), and the fixed-effects parameters cannot be hyperparameters of $\boldsymbol{\gamma}$ (hence no $\log(\pi(\gamma_j|\boldsymbol{\beta}))$ terms).

The conditional distribution of $\mathbf{R}$ again mirrors that of $\boldsymbol{\beta}$:

$$\log(p(\mathbf{R}|\boldsymbol{\gamma}, \mathbf{y}, \boldsymbol{\beta})) = \log(\pi(\mathbf{R})) + \sum_{i=1}^{n} \log(f(\mathbf{y}_i|\boldsymbol{\beta}, \boldsymbol{\gamma}, \mathbf{R}))$$

For $\boldsymbol{\gamma}_j$, the following conditional is used:

$$\log(p(\boldsymbol{\gamma}_j|\boldsymbol{\theta}, \mathbf{y})) = \log(\pi(\boldsymbol{\gamma}_j|\mathbf{G})) + \sum_{i \in \{j\text{th cluster}\}} \log(f(\mathbf{y}_i|\boldsymbol{\beta}, \boldsymbol{\gamma}_j, \mathbf{R}))$$

In this computation, only subjects from the *j*th cluster are used. This reflects the conditional independence assumption that the RANDOM statement makes. This simplification in the calculation makes updating the random-effects parameters computationally efficient and enables the procedure to handle random effects that contain large number of clusters just as easily.

The G-side covariance matrix $\mathbf{G}$ depends only on the random effects γ and not on the data or other parameters, $\boldsymbol{\beta}$ or $\mathbf{R}$,

$$\log(p(\mathbf{G}|\boldsymbol{\gamma})) = \log(\pi(\mathbf{G}))) + \sum_{j} \log(\pi(\boldsymbol{\gamma}_j|\mathbf{G}))$$

where $\pi(\mathbf{G})$ is the prior distribution of $\mathbf{G}$.

Models with Missing Values

Missing response values are treated as parameters by default and sampled in the MCMC simulation. This mechanism of modeling missing values is referred to as missing at random (MAR). You can delete all observations that contain missing values by using the MISSING=CC option in the PROC BGLIMM statement.

Suppose that

$$\mathbf{y} = \{\mathbf{y}_{\mathbf{obs}}, \mathbf{y}_{\mathbf{mis}}\}$$

The response variable $\mathbf{y}$ consists of n_1 observed values, $\mathbf{y}_{\mathbf{obs}}$, and n_2 missing values, $\mathbf{y}_{\mathbf{mis}}$. At each iteration, PROC BGLIMM samples every missing response value (by using the likelihood function as the sampling distribution). After these samples are drawn, the GLMM is reduced to a full data scenario with no missing data. PROC BGLIMM then proceeds to update $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, $\mathbf{G}$, and $\mathbf{R}$ sequentially, in the same way as described in the section "How PROC BGLIMM Works" on page 1339.

Sampling Methods

This section describes the sampling algorithms that PROC BGLIMM uses.

Conjugate Sampling

Conjugate prior distributions are a type of prior distribution in which the prior and posterior distributions are in the same family of distributions. For example, if you model an independently and identically distributed random variable y_i by using a normal likelihood with known variance σ^2,

$$y_i \sim \text{normal}(\mathbf{x}_i'\boldsymbol{\beta}, \sigma^2)$$

then a normal prior on $\boldsymbol{\beta}$,

$$\boldsymbol{\beta} \sim \text{normal}(\boldsymbol{\beta}_0, \boldsymbol{\Sigma}_0)$$

is a conjugate prior, because the posterior distribution of $\boldsymbol{\beta}$ is also a normal distribution, where the mean is $(\boldsymbol{\Sigma}_0^{-1} + \frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i')^{-1}\,(\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\beta}_0 + \frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{x}_i y_i)$ and the covariance is $(\boldsymbol{\Sigma}_0^{-1} + \frac{1}{\sigma^2}\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i')^{-1}$.

PROC BGLIMM uses conjugate samplers in the normal and multivariate normal cases, as shown in Table 31.12.

Table 31.12 Conjugate Sampling in PROC BGLIMM

Family	Parameter	Prior
Normal	$\boldsymbol{\beta}$	Multivariate normal
Normal	Variance σ^2	Inverse gamma
Multivariate normal	$\boldsymbol{\beta}$	Multivariate normal
Multivariate normal	Covariance $\boldsymbol{\Sigma}$	Inverse Wishart

The fixed-effects parameters $\boldsymbol{\beta}$ and the covariances $\mathbf{G}$ and $\mathbf{R}$ are sampled when applicable.

Gamerman Algorithm

The Gamerman algorithm (Gamerman 1997), which is named after the inventor Dani Gamerman, is a special case of the Metropolis algorithm in which the proposal distribution is derived from one iteration of the iterative weighted least squares (IWLS) algorithm. As the name suggests, a weighted least squares algorithm runs inside an iteration loop. For each iteration, a set of weights for the observations is used in the least squares fit. The weights are constructed by applying a weight function to the current residuals. The proposal distribution uses the current iteration's parameter values to form the proposal distribution from which to generate a proposed random value (Gamerman 1997).

The Gamerman algorithm is suitable for both GLM and GLMM models.

The maximum likelihood (ML) estimator in a GLM and the asymptotic variance are obtained by iterative application of weighted least squares (IWLS) to transformed observations. Following McCullagh and Nelder

(1989), define the transformed response as

$$\tilde{\mathbf{y}}_i(\boldsymbol{\beta}) = \eta_i + (\mathbf{y}_i - \boldsymbol{\mu}_i)g'(\boldsymbol{\mu}_i)$$

and define the corresponding weights as

$$\mathbf{W}_i^{-1}(\boldsymbol{\beta}) = b''(\boldsymbol{\theta}_i)[g'(\boldsymbol{\mu}_i)]^2$$

The Gamerman algorithm is summarized as follows:

1. Start with $\boldsymbol{\beta}^{(0)}$ and $t = 1$.
2. Sample $\boldsymbol{\beta}^*$ from the proposal density $\mathbf{N}(\mathbf{m}^{(t)}, \mathbf{C}^{(t)})$, where

$$\begin{aligned}\mathbf{m}^{(t)} &= \{\boldsymbol{\Omega}_{\boldsymbol{\beta}}^{-1} + \mathbf{X}'\mathbf{W}(\boldsymbol{\beta}^{(t-1)})\mathbf{X}\}^{-1}\{\boldsymbol{\Omega}_{\boldsymbol{\beta}}^{-1}\bar{\boldsymbol{\beta}} + \mathbf{X}'\mathbf{W}(\boldsymbol{\beta}^{(t-1)})\tilde{\mathbf{y}}(\boldsymbol{\beta}^{(t-1)})\}\\ \mathbf{C}^{(t)} &= \{\boldsymbol{\Omega}_{\boldsymbol{\beta}}^{-1} + \mathbf{X}'\mathbf{W}(\boldsymbol{\beta}^{(t-1)})\mathbf{X}\}^{-1}\end{aligned}$$

3. Accept $\boldsymbol{\beta}^*$ with probability

$$\alpha(\boldsymbol{\beta}_{(t-1)}, \boldsymbol{\beta}^*) = \min[1, \frac{p(\boldsymbol{\beta}^*|\mathbf{y})q(\boldsymbol{\beta}^*, \boldsymbol{\beta}^{(t-1)})}{p(\boldsymbol{\beta}^{(t-1)}|\mathbf{y})q(\boldsymbol{\beta}^{(t-1)}, \boldsymbol{\beta}^*)}]$$

   where $p(\boldsymbol{\beta}|\mathbf{y})$ is the posterior density and $q(\boldsymbol{\beta}^*, \boldsymbol{\beta}^{(t-1)})$ and $q(\boldsymbol{\beta}^{(t-1)}, \boldsymbol{\beta}^*)$ are the transitional probabilities that are based on the proposal density $\mathbf{N}(\mathbf{m}^{(.)}, \mathbf{C}^{(.)})$. More specifically, $q(\boldsymbol{\beta}^*, \boldsymbol{\beta}^{(t-1)})$ is an $\mathbf{N}(\mathbf{m}^*, \mathbf{C}^*)$ density that is evaluated at $\boldsymbol{\beta}^{(t-1)}$, whereas $\mathbf{m}^*$ and $\mathbf{C}^*$ have the same expression as $\mathbf{m}^{(t)}$ and $\mathbf{C}^{(t)}$ but depend on $\boldsymbol{\beta}^*$ instead of $\boldsymbol{\beta}^{(t-1)}$. If $\boldsymbol{\beta}^*$ is not accepted, the chain stays with $\boldsymbol{\beta}^{(t-1)}$.
4. Set $t = t + 1$ and return to step 1.

PROC BGLIMM uses this algorithm to draw samples for both the fixed-effects parameters $\boldsymbol{\beta}$ and the random-effects parameters $\boldsymbol{\gamma}$: the GLMM simplifies to a GLM when $\boldsymbol{\gamma}$ is conditioned on; similarly, for the ith cluster, the model for $\boldsymbol{\gamma}_i$ is simplified to a GLM when $\boldsymbol{\beta}$ are treated as known and conditioned on.

For the random-effects $\boldsymbol{\gamma}_i$ block, the same Metropolis-Hastings sampling with the least squares proposal can apply. The conditional posterior is

$$p(\boldsymbol{\gamma}_i|\mathbf{y}, \boldsymbol{\beta}, \mathbf{G}) \propto \exp\{\frac{\mathbf{y}_i\boldsymbol{\theta}_i - b(\boldsymbol{\theta}_i)}{\phi_i} - \frac{1}{2}\boldsymbol{\gamma}_i'\mathbf{G}^{-1}\boldsymbol{\gamma}_i\}$$

The transformed response is now $\tilde{\mathbf{y}}_i(\boldsymbol{\gamma}_i^{(t-1)})$, and the proposal density is $\mathbf{N}(\mathbf{m}_i^{(t)}, \mathbf{C}_i^{(t)})$, where

$$\begin{aligned}\mathbf{m}_i^{(t)} &= \{\mathbf{G}^{-1} + \mathbf{Z}'\mathbf{W}(\boldsymbol{\gamma}_i^{(t-1)})\mathbf{Z}\}^{-1}\mathbf{Z}'\mathbf{W}(\boldsymbol{\gamma}_i^{(t-1)})\{\tilde{\mathbf{y}}(\boldsymbol{\gamma}_i^{(t-1)}) - \mathbf{X}_i\boldsymbol{\beta}\}\\ \mathbf{C}^{(t)} &= \{\mathbf{G}^{-1} + \mathbf{Z}'\mathbf{W}(\boldsymbol{\gamma}_i^{(t-1)})\mathbf{X}_i\}^{-1}\end{aligned}$$

Hamiltonian Monte Carlo Sampler

The Hamiltonian Monte Carlo (HMC) algorithm, also known as the hybrid Monte Carlo algorithm, is a version of the Metropolis algorithm that uses gradient information and auxiliary momentum variables to draw samples from the posterior distribution (Neal 2011). The algorithm uses Hamiltonian dynamics to enable distant proposals in the Metropolis algorithm, making it efficient in many scenarios. The HMC algorithm is applicable only to continuous parameters.

HMC translates the target density function to a potential energy function and adds an auxiliary momentum variable $\mathbf{r}$ for each model parameter $\boldsymbol{\theta}$. The resulting joint density has the form

$$p(\boldsymbol{\theta}, \mathbf{r}) \propto p(\boldsymbol{\theta}) \exp\left(-\frac{1}{2}\mathbf{r}'\mathbf{r}\right)$$

where $p(\boldsymbol{\theta})$ is the posterior of the parameters $\boldsymbol{\theta}$ (up to a normalizing constant). HMC draws from the joint space of $(\boldsymbol{\theta}, \mathbf{r})$, discards $\mathbf{r}$, and retains $\boldsymbol{\theta}$ as samples from $p(\boldsymbol{\theta})$. The algorithm uses the idea of Hamiltonian dynamics in preserving the total energy of a physical system, in which $\boldsymbol{\theta}$ is part of the potential energy function and $\mathbf{r}$ is part of the kinetic energy (velocity). As the velocity changes, the potential energy changes accordingly, leading to the movements in the parameter space.

At each iteration, the HMC algorithm first generates the momentum variables $\mathbf{r}$, usually from standard normal distributions, that are independent of $\boldsymbol{\theta}$. Then the algorithm follows with a Metropolis update that includes many steps along a trajectory while maintaining the total energy of the system. One of the most common approaches in moving along this trajectory is the leapfrog method, which involves L steps with a step size ϵ,

$$\begin{aligned}
\mathbf{r}^{t+\epsilon/2} &= \mathbf{r}^{t} + (\epsilon/2) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}^{t}) \\
\boldsymbol{\theta}^{t+\epsilon} &= \boldsymbol{\theta}^{t} + \epsilon \mathbf{r}^{t+\epsilon/2} \\
\mathbf{r}^{t+\epsilon} &= \mathbf{r}^{t+\epsilon/2} + (\epsilon/2) \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}^{t+\epsilon})
\end{aligned}$$

where $\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta})$ is the gradient of the log posterior with respect to $\boldsymbol{\theta}$. After L steps, the proposed state $(\boldsymbol{\theta}^*, \mathbf{r}^*)$ is accepted as the next state of the Markov chain with probability $\min\{1, p(\boldsymbol{\theta}^*, \mathbf{r}^*)/p(\boldsymbol{\theta}, \mathbf{r})\}$.

Although HMC can lead to rapid convergence, it also heavily relies on two requirements: the gradient calculation of the logarithm of the posterior density and carefully selected tuning parameters, in step size ϵ and number of steps L. Step sizes that are too large or too small can lead to acceptance rates that are too low or too high, both of which affect the convergence of the Markov chain. A large L leads to a large trajectory length ($\epsilon \cdot L$), which can move the parameters back to their original positions. A small L limits the movement of the chain.

An example of an adaptive HMC algorithm with automatic tuning of ϵ and L is the No-U-Turn Sampler (NUTS; Hoffman and Gelman 2014). The NUTS algorithm uses a doubling process to build a binary tree whose leaf nodes correspond to the states of the parameters and momentum variables. The initial tree has a single node with no heights ($j = 0$). The doubling process expands the tree to either the left or right in a binary fashion, and in each direction, the algorithm takes 2^j leapfrog steps of size ϵ. Obviously, as the height of the tree (j) increases, the computational cost increases dramatically. The tree expands until one sampling trajectory makes a U-turn and starts to revisit parameter space that has been already explored. The NUTS algorithm tunes ϵ so that the actual acceptance rate during the doubling process is close to a predetermined target acceptance probability δ (usually set to 0.6 or higher). When the tuning stage ends, the NUTS algorithm proceeds to the main sampling stage and starts to draw posterior samples that have a fixed ϵ value. Increasing the targeted acceptance probability δ can often improve mixing, but it can also slow down the process significantly. For more information about the NUTS algorithm and its efficiency, see Hoffman and Gelman (2014).

Slice Sampler

The slice sampling algorithm (Neal 2003) is a general algorithm that you can use to sample parameters from their target distribution. The requirement of applying the slice sampler is the ability to evaluate the objective function (the unnormalized conditional distribution in a Gibbs step, for example) at a given parameter value. In theory, you can draw a random number from any given distribution as long as you can first obtain a random number uniformly under the curve of that distribution. Treat the area under the curve of $p(\theta)$ as a two-dimensional space that is defined by the θ axis and the Y axis; the latter is the axis for the density function. You uniformly draw a two-dimensional vector (θ_i, y_i) in this area, ignore the y_i, and keep the θ_i. The θ_i's are distributed according to the density function.

To solve the problem of sampling uniformly under the curve, Neal (2003) proposed the idea of slices (hence the name of the sampler), which can be explained as follows:

1. Start the algorithm at θ_0.
2. Calculate the objective function $p(\theta_0)$, and draw a line between $y = 0$ and $y = p(\theta_0)$, which defines a vertical slice. You draw a uniform number, y_1, on this slice, between 0 and $p(\theta_0)$.
3. Draw a horizontal line at y_1, and find the two points where the line intercepts with the curve, L_1 and R_1. These two points define a horizontal slice. Draw a uniform number, x_1, on this slice, between L_1 and R_1.
4. Repeat steps 2 and 3 many times.

The challenging part of the algorithm is finding the horizontal slice (L_i, R_i) at each iteration. The closed-form expressions of $p_L^{-1}(y_i)$ and $p_R^{-1}(y_i)$ are virtually impossible to obtain analytically in most problems. Neal (2003) proved that although exact solutions would be ideal, devising a search algorithm that finds portions of this horizontal slice is sufficient for the sampler to work. The search algorithm applies a rejection method to perform regional expansion and contraction, when needed, until the obtained sample is within the slice.

The sampler is implemented as the default algorithm in the BGLIMM procedure to draw the parameters in the **G** matrix and the **R** matrix, when conjugate sampling is unavailable.

Prior Distributions

A GLMM model contains various types of parameters: the coefficients for the fixed effects that are specified in the MODEL statement; the coefficients for the random effects that are specified in the RANDOM statement; and the parameters for the covariance matrices, including both the G-side and R-side covariance matrices.

Prior for the Fixed-Effects Coefficients

In GLMMs, flat priors on the fixed-effects coefficients ($\boldsymbol{\beta}$) are considered to be noninformative. The flat prior assigns equal likelihood for all possible values of the parameter, $\pi(\boldsymbol{\beta}) \propto 1$. This is the default prior for the fixed-effects coefficients in PROC BGLIMM.

In addition to the flat prior, a normal prior that has very large variance is also considered to be noninformative or weakly informative. The following statement specifies such a prior:

```
model y = x / cprior=normal(var=1e4);
```

This normal prior is noninformative because its variance value is sufficiently larger than the posterior variances of all the $\boldsymbol{\beta}$ parameters.

On the other hand, you can use an informative prior by making the variance small in the normal distribution. If you want to have a specific mean and covariance for the normal prior, you can provide a SAS data set that contains the mean and covariance information of the normal prior through the INPUT=*SAS-data-set* in the CPRIOR= option,

```
model y = x / cprior=normal(input=MyPrior);
```

where MyPrior is the name of a SAS data set.

Prior for the Random-Effects Coefficients

The random-effects coefficients $\boldsymbol{\gamma}$ are assumed to have the normal prior

$$\boldsymbol{\gamma} \sim \text{Normal}(\mathbf{0}, \mathbf{G})$$

where $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_n)^T$ and $\mathbf{G}$ is a block diagonal matrix where each block is $\mathbf{G}_i$. For a specific subject i, the random-effects coefficients $\boldsymbol{\gamma}_i$ are normally distributed with mean zero and the covariance matrix $\mathbf{G}_i$. When the dimension is the same, $\mathbf{G}_i = \mathbf{G}_j$ for any i and j.

You use the TYPE= option in the RANDOM statement to specify the types of the $\mathbf{G}$ matrix.

Prior for the G-Side Covariance

PROC BGLIMM supports the following prior distributions of $\mathbf{G}$:

- Inverse Wishart distribution: $\pi(\mathbf{G}_i) = \text{IW(a,bI)}$. You can specify this distribution as follows:

  ```
  random x / sub=Id covprior=iwishart(df=a, scale=b);
  ```

  The inverse Wishart distribution is a generalization of the inverse gamma distribution; when the dimension is 1, the inverse Wishart distribution is equivalent to an inverse gamma distribution. The inverse Wishart prior is frequently used in Bayesian analysis, in part because it is a conjugate prior for a normal covariance, which leads to efficient sampling. An inverse Wishart prior that has small degrees of freedom (interpreted as prior sample size) and small diagonal values of the scale matrix is considered to be weakly informative.

  The inverse Wishart prior applies to the covariance types UN, UN(1), VC, and TOEP(1). It is the default prior for the UN type.

- Inverse gamma distribution: $\pi(\sigma_g^2) = \text{IG(a,b)}$. You can specify this distribution as follows:

  ```
  random x / sub=Id covprior=igamma(shape=a, scale=b);
  ```

  The inverse gamma prior applies to the diagonal variance terms of a covariance that is of type UN, UN(1), VC, or TOEP(1). It is the default prior for the covariance types UN(1), VC, and TOEP(1).

- Uniform distribution: $\pi(\sigma_\gamma) = \frac{1}{\text{Upper}-\text{Lower}}$. You can specify this distribution as follows:

```
random x / sub=Id covprior=uniform(lower=, upper=);
```

The uniform prior is on the standard deviation, not the variance, terms of **G**. This prior applies to the diagonal standard deviation terms of a covariance that is of type UN, UN(1), VC, or TOEP(1).

- Half-Cauchy distribution: $\pi(\sigma_\gamma^2) = \text{half-Cauchy}(a)$. You can specify this distribution as follows:

```
random x / sub=Id covprior=halfcauchy(scale=a);
```

The half-Cauchy prior is a truncated prior whose lower bound is equal to zero. The prior applies to the variance terms of a covariance that is of type UN, UN(1), VC, or TOEP(1).

The half-Cauchy prior is a special case of the conditionally conjugate folded-noncentral-*t* distribution. The right tail of the half-Cauchy distribution is thick and decays slowly. Large a translates to weakly informative, a goes to infinity, and the half-Cauchy distribution becomes a uniform distribution.

- Half-normal distribution: $\pi(\sigma_\gamma^2) = \text{half-normal}(0, \sigma^2 = a)$. You can specify this distribution as follows:

```
random x / sub=Id covprior=halfnormal(var=a);
```

The half-normal prior is a truncated prior whose lower bound is equal to zero. The prior applies to the variance terms of a covariance that is of type UN, UN(1), VC, or TOEP(1).

- Scaled inverse Wishart distribution. You can specify this distribution as follows:

```
random x / sub=Id covprior=siwishart(df=a, scale=b, var=c);
```

This is a generalization of the inverse Wishart prior. The distribution decomposes the $\mathbf{G}_i$ matrix into variance and correlation components and specifies separate priors for each component (O'Malley and Zaslavsky 2008). Define $\boldsymbol{\Delta} = \text{diag}(\boldsymbol{\delta})$, where $\boldsymbol{\delta} = (\delta_1, \delta_2, \ldots, \delta_n)$ and $\delta_i > 0$; let $\boldsymbol{\Phi}$ be a positive definite matrix; and write $\mathbf{G}_i = \boldsymbol{\Delta}\boldsymbol{\Phi}\boldsymbol{\Delta}$,

$$\pi(\boldsymbol{\Delta}) = \text{IW(a, bI)}, \pi(\log(\delta_i)) = \text{normal}(0, \sigma_\delta^2)$$

where the hyperparameters a and b are the degrees of freedom and scale parameters, respectively, of the inverse Wishart prior; $\sigma_\delta^2 = c$ is the variance of the normal prior; and the default variance is 1. You can choose values for the hyperparameters a and b, and the default is the dimension of the $\mathbf{G}_i$ matrix plus 3 for both a and b.

Although $\boldsymbol{\Phi}$ determines the correlations, it is not constrained to be a correlation matrix, so the model is overparameterized. The associated correlation matrix is $\boldsymbol{\Omega}(\boldsymbol{\Phi}) = \text{diag}(\boldsymbol{\Phi})^{-\frac{1}{2}}\boldsymbol{\Phi}\text{diag}(\boldsymbol{\Phi})^{-\frac{1}{2}}$, where $\text{diag}(\boldsymbol{\Phi})$ retains the diagonal elements of $\boldsymbol{\Phi}$ but has zeros elsewhere.

Like the inverse Wishart prior, the scaled inverse Wishart prior applies to types UN, UN(1), VC, and TOEP(1).

For the covariance types AR(1) and ARMA(1,1), by default, a conjugate inverse gamma prior $\text{IG}(2.001, 0.001)$ is applied to the scale parameter of the **G** covariance matrix, and the flat prior is applied to other parameters of the **G** covariance matrix. If you specify the NOCONJCOV option in the RANDOM statement, the flat prior is applied to all parameters of the **G** covariance matrix. For all other covariance types, the flat prior is applied to all parameters of the **G** covariance matrix.

Prior for the Scale Parameter

In models that have a scale parameter (see Table 31.11), you can specify the following prior distribution:

- Inverse gamma distribution: $\pi(\phi) = \mathrm{IG}(a,b)$. You can specify this distribution as follows:

```
model y = x / scaleprior = igamma(shape=a, scale=b);
```

  The inverse gamma prior applies to all distributions that have a scale parameter.

- Gamma distribution: $\pi(\phi) = \mathrm{gamma}(a,b)$. You can specify this distribution as follows:

```
model y = x / scaleprior = gamma(shape=a, iscale=b);
```

  The gamma prior applies to the gamma, inverse gamma, and negative binomial distributions. It does not apply to the normal likelihood function with an identity link.

- Improper prior: $\pi(\phi) \propto 1/\phi$. You can specify this distribution as follows:

```
model y = x / scaleprior = improper;
```

  The improper prior applies to the gamma, inverse gamma, and negative binomial distributions. It does not apply to the normal likelihood function with an identity link.

Prior for the R-Side Covariance

You can specify one of the following prior distributions for the **R** covariance matrix: inverse gamma distribution, inverse Wishart distribution, half-Cauchy distribution, half-normal distribution, scaled inverse Wishart distribution, and uniform distribution. These distributions are the same as what you can specify for the **G** covariance matrix. The inverse Wishart distribution is the default prior for the UN type, and the inverse gamma distribution is the default prior for types UN(1), VC, and TOEP(1).

For the covariance types AR(1) and ARMA(1,1), by default, a conjugate inverse gamma prior $\mathrm{IG}(2.001, 0.001)$ is applied to the scale parameter of the **R** covariance matrix, and the flat prior is applied to other parameters of the **R** covariance matrix. If you specify the NOCONJCOV option in the REPEATED statement, the flat prior is applied to all parameters of the **R** covariance matrix. For all other covariance types, the flat prior is applied to all parameters of the **R** covariance matrix.

Treatment of Subjects in the RANDOM Statement

PROC BGLIMM supports the syntax of specifying random effects without specifying a SUBJECT= option, but this is different from specifying an intercept random effect and using the same random effects in the SUBJECT= option.

For example, suppose you have a nested effect **A(B)**, where both A and B are categorical variables. You can specify the following statement:

```
random A(B);
```

In the traditional mixed modeling procedures, such as PROC MIXED and PROC GLIMMIX, this specification is the same as

```
random int / subject = A(B);
```

But this is not the case in PROC BGLIMM. The difference is as follows:

- When you specify the SUBJECT= option in PROC BGLIMM, the procedure fits a normal prior (with default TYPE=VC),

  $$\pi(\gamma_j) \sim \text{normal}(0, \sigma_\gamma^2), \quad j = 1, \ldots, J$$

  where γ_j is the intercept from cluster j and J is the total number of unique clusters in effect **A(B)**. This is the expected way of specifying a prior on intercept random effects.

- When you omit the SUBJECT= option, PROC BGLIMM assumes that there is only one cluster (instead of J clusters). This cluster, which represents the entire data set, has nested (categorical) effects of **A(B)**.

  This is equivalent to treating **A(B)** as categorical fixed effects in the regression model (in the MODEL statement):

  ```
  model resp = A(B);
  ```

  The exception is that, instead of a noninformative or flat prior on the regression coefficients, a hierarchical normal prior is assumed (again, following the specification of the TYPE= covariance),

  $$\beta_j \sim \text{normal}(0, \sigma_\beta^2), \quad j = 1, \ldots, K$$

  where β_j (used here instead of γ_j to emphasize the different treatment in modeling by PROC BGLIMM) is a categorical coefficient for the jth level and K is less or equal to J, depending on the rank of the regression model.

  In other words, after you specify a random effect without using the SUBJECT= option, the effect becomes equivalent to a fixed effect (but with a shrinkage prior). The number of estimable parameters from that effect depends on the rank of the model, so you will often see that some of the cluster parameters are inestimable (hence, their value is displayed as zero).

In PROC MIXED and PROC GLIMMIX, the two specifications lead to identical numerical estimates, because neither procedure requires the design matrix (of the regression model) to be nonsingular. But PROC BGLIMM has this requirement, and this results in different posterior estimates.

If you want to specify a random-effects model in the strict Bayesian sense, use the SUBJECT= option in the RANDOM statement.

Initial Values of the Markov Chains

A GLMM model has various types of parameters: fixed-effects parameters (in the MODEL statement), random-effects parameters (in the RANDOM statement), parameters in the covariance matrix, and missing data variables in the response.

For the fixed-effects parameters, PROC BGLIMM generates initial values that are based on the optimization of the posterior density functions. To assign initial values to the fixed-effects parameters, you can use the INIT= option in the MODEL statement. If there are multiple fixed effects, you can provide a list of numbers, where the length of the list is the same as the dimension of the fixed effects. Each number is then given to all corresponding fixed-effects parameters in order.

For all the individual random-effects parameters, PROC BGLIMM sets the initial values to zero.

The initial values for either the **G** or **R** matrix are set to the identity matrix.

For missing data in the response, PROC BGLIMM uses the sample average of the nonmissing values as the initial value. The procedure requires some response value to be nonmissing—if all values of a particular variable are missing, PROC BGLIMM issues an error and stops.

Missing Data

When you have missing data, you can use the MISSING= option in the PROC BGLIMM statement as follows to specify how you want to handle the missing response values:[2]

- If you specify MISSING=CC (CC stands for complete cases), PROC BGLIMM discards all observations that have missing or partial missing values (for example, in a repeated measures model) before carrying out the simulation. This is equivalent to assuming that the missing values are missing completely at random (MCAR).
- If you specify MISSING=CCMODELY, PROC BGLIMM treats missing response values as parameters and includes the sampling of the missing data as part of the simulation. The procedure discards all observations that have missing covariates. This is equivalent to assuming that the missing values are missing at random (MAR).

Different types of missing data were first defined by Rubin (1976). For a comprehensive treatment of missing data analysis, see Little and Rubin (2002). PROC BGLIMM does not model the missing not at random (MNAR) type of missing data.

[2] A missing value is usually, although not necessarily, represented by a single period (.) in the input data set.

Multinomial Models

The BGLIMM procedure fits two kinds of models to multinomial data: one that has a cumulative link that applies to ordinal data, and one that has a generalized logit link that applies to nominal data.

PROC BGLIMM models the proportional-odds cumulative probabilities for ordinal data. This model uses cumulative probabilities up to a threshold, thereby making the whole range of ordinal categories binary at that threshold. If the nominal response has J categories and the ordering is natural, the associated probabilities are $\pi_1, \pi_2, \ldots, \pi_J$, and a cumulative probability of a response less than equal to j is

$$\log\left\{\frac{\Pr(Y \le j)}{\Pr(Y > j)}\right\} = \log\left\{\frac{\Pr(Y \le j)}{1 - \Pr(Y \le j)}\right\} = \log\left\{\frac{\pi_1 + \pi_2 + \cdots + \pi_j}{\pi_{j+1} + \cdots + \pi_J}\right\}$$

This describes the log odds of two cumulative probabilities: a less-than type and a greater-than type. This measures how likely the response is to be in category j or below, or how likely it is to be in a category higher than j.

The linear predictor depends on the response category only through the intercepts (cutoffs) $\alpha_1, \ldots, \alpha_{J-1}$,

$$\begin{aligned}
\eta_1 &= \alpha_1 + \mathbf{x}'\boldsymbol{\beta} + \mathbf{z}'\boldsymbol{\gamma} \\
\eta_2 &= \alpha_2 + \mathbf{x}'\boldsymbol{\beta} + \mathbf{z}'\boldsymbol{\gamma} \\
&\vdots \\
\eta_{J-1} &= \alpha_{J-1} + \mathbf{x}'\boldsymbol{\beta} + \mathbf{z}'\boldsymbol{\gamma} \\
\eta_J &= 0,
\end{aligned}$$

where $\alpha_1 \le \alpha_2 \cdots \le \alpha_{J-1}$ to ensure the cumulative property and $(\boldsymbol{\beta}, \boldsymbol{\gamma})$ remains the same across all levels of the response variable. The cumulative logits are formed as

$$\log\left\{\frac{\Pr(Y \le j)}{\Pr(Y > j)}\right\} = \eta_j = \alpha_j + \mathbf{x}'\boldsymbol{\beta} + \mathbf{z}'\boldsymbol{\gamma} = \alpha_j + \tilde{\eta}$$

The odds ratio that compares two conditions represented by the linear predictors η_{j1} and η_{j0} is then

$$\begin{aligned}
\psi\left(\eta_{j1}, \eta_{j0}\right) &= \exp\left(\eta_{j1} - \eta_{j0}\right) \\
&= \exp\left(\tilde{\eta}_1 - \tilde{\eta}_0\right)
\end{aligned}$$

and is independent of category. You might think of this as a set of parallel lines (or hyperplanes) with different intercepts. The proportional-odds condition forces the lines that correspond to each cumulative logit to be parallel.

In the generalized logit model, you model baseline-category logits. By default, the BGLIMM procedure chooses the last category as the baseline category or reference, but you can use another category as the reference. If the nominal response has J categories, both the fixed effects and the random effects in the linear predictor depend on the response category:

$$\begin{aligned}
\eta_1 &= \mathbf{x}'\boldsymbol{\beta}_1 + \mathbf{z}'\boldsymbol{\gamma}_1 \\
\eta_2 &= \mathbf{x}'\boldsymbol{\beta}_2 + \mathbf{z}'\boldsymbol{\gamma}_2 \\
&\vdots \\
\eta_{J-1} &= \mathbf{x}'\boldsymbol{\beta}_{J-1} + \mathbf{z}'\boldsymbol{\gamma}_{J-1} \\
\eta_J &= 0
\end{aligned}$$

The last category defaults to zero in order to ensure the estimability property. The baseline-category logit for category j is

$$\log(\pi_j/\pi_J) = \eta_j = \mathbf{x}'\boldsymbol{\beta}_j + \mathbf{z}'\boldsymbol{\gamma}_j$$

and

$$\pi_j = \frac{\exp(\eta_j)}{\sum_{k=1}^{J}\exp(\eta_k)}$$

Suppose that the two conditions to be compared are identified using subscripts 1 and 0. The log odds ratio of outcome j versus J for the two conditions is then

$$\begin{aligned}\log\left\{\psi\left(\eta_{j1},\eta_{j0}\right)\right\} &= \log\left\{\frac{\pi_{j1}/\pi_{J1}}{\pi_{j0}/\pi_{J0}}\right\} = \log\left\{\frac{\exp\{\eta_{j1}\}}{\exp\{\eta_{j0}\}}\right\}\\ &= \eta_{j1}-\eta_{j0}\end{aligned}$$

Note that the log odds ratios are again differences on the scale of the linear predictor, but they depend on the response category. The BGLIMM procedure determines the estimable functions whose differences represent log odds ratios, but it produces separate estimates for each nonreference response category.

Autocall Macros for Postprocessing

Although PROC BGLIMM provides a number of convergence diagnostic tests and posterior summary statistics, it performs the calculations only for the default tests and statistics or only if you specify the necessary options. If you want to analyze the posterior draws of unmonitored parameters or functions of the parameters that are calculated in later DATA step calls, you can use the autocall macros that are listed in Table 31.13.

Table 31.13 Postprocessing Autocall Macros

Macro	Description
%ESS	Effective sample sizes
%GEWEKE*	Geweke diagnostic
%HEIDEL*	Heidelberger-Welch diagnostic
%MCSE	Monte Carlo standard errors
%POSTACF	Autocorrelation
%POSTCOR	Correlation matrix
%POSTCOV	Covariance matrix
%POSTINT	Equal-tail and HPD intervals
%POSTSUM	Mean, standard deviation, and various quantiles
%RAFTERY	Raftery diagnostic
%SUMINT	Mean, standard deviation, and HPD interval
%TADPLOT	Trace plot, autocorrelation plot, and density plot

*The %GEWEKE and %HEIDEL macros use a different optimization routine than PROC BGLIMM uses. As a result, there might be numerical differences in some cases, especially when the sample size is small.

Table 31.14 lists options that are shared by all postprocessing autocall macros. For macro-specific options, see Table 31.15.

Table 31.14 Shared Options

Option	Description
DATA=*SAS-data-set*	Names the input data set that contains posterior samples
OUT=*SAS-data-set*	Specifies a name for the output SAS data set to contain the results
PRINT=YES | NO	Displays the results (the default is YES)
VAR=*variable-list*	Specifies the variables on which to perform the calculation

Suppose that the data set that contains posterior samples is called Post and the variables of interest are defined in the macro variable &PARMS. The following statement calls the %ESS macro and calculates the effective sample sizes for each variable:

```
%ESS(data=Post, var=Alpha Beta U_1-U_17)
```

By default, the ESS estimates are displayed. You can choose not to display the result and instead use the following statement to save the output to a data set:

```
%ESS(data=Post, var=&parms, print=NO, out=eout)
```

Some of the macros can take additional options, which are listed in Table 31.15.

Table 31.15 Macro-Specific Options

Macro	Option	Description
%ESS	AUTOCORLAG=*numeric*	Specifies the maximum number of autocorrelation lags used in computing the ESS estimates. By default, AUTOCORLAG=MIN(500, NOBS/4), where NOBS is the sample size of the input data set.
	HIST=YES | NO	Displays a histogram of all ESS estimates. By default, HIST=NO.
%GEWEKE	FRAC1=*numeric*	Specifies the earlier portion of the Markov chain used in the test. By default, FRAC1=0.1.
	FRAC2=*numeric*	Specifies the latter portion of the Markov chain used in the test. By default, FRAC2=0.5.
%HEIDEL	SALPHA=*numeric*	Specifies the α level for the stationarity test. By default, SALPHA=0.05.
	HALPHA=*numeric*	Specifies the α level for the halfwidth test. By default, HALPHA=0.05.
	EPS=*numeric*	Specifies a small positive number ϵ such that if the halfwidth is less than ϵ times the sample mean of the remaining iterations, the halfwidth test is passed. By default, EPS=0.1.

Table 31.15 *continued*

Option	Description	
%MCSE	AUTOCORLAG=*numeric*	Specifies the maximum number of autocorrelation lags used in computing the Monte Carlo standard error estimates. By default, AUTOCORLAG=MIN(500, NOBS/4), where NOBS is the sample size of the input data set.
%POSTACF	LAGS=*%str(numeric-list)*	Specifies which autocorrelation lags to calculate. The default values are 1, 5, 10, and 50.
%POSTINT	ALPHA=*value*	Specifies the α level ($0 < \alpha < 1$) for the interval estimates. By default, ALPHA=0.05.
%RAFTERY	Q=*numeric*	Specifies the order of the quantile of interest. By default, Q=0.025.
	R=*numeric*	Specifies the margin of error for measuring the accuracy of estimation of the quantile. By default, R=0.005.
	EPS=*numeric*	Specifies the tolerance level for the stationary test. By default, EPS=0.001.

For example, the following statement calculates and displays autocorrelation at lags 1, 6, 11, 50, and 100. Note that the lags in the *numeric-list* must be separated by commas.

```
%POSTACF(data=Post, var=&parms, lags=%str(1 to 15 by 5, 50, 100))
```

Regenerating Diagnostics Plots

By default, PROC BGLIMM generates three plots: the trace plot, the autocorrelation plot, and the kernel density plot. Unless ODS Graphics is enabled before the procedure is called, it is hard to generate the same graph afterward. Directly using the **Stat.Bglimm.Graphics.TraceAutocorrDensity** template is not feasible. The easiest way to regenerate the same graph is to use the %TADPLOT autocall macro. This macro requires you to specify an input data set (which is the output data set from a previous PROC BGLIMM call) and a list of variables that you want to plot.

Suppose that the output data set Postsamp contains posterior draws for the regression coefficients of Mode1, Mode2, and Mode3. If you want to examine these parameters graphically, you can use the following statement to regenerate the graphs:

```
%TADPLOT(data=Postsamp, var=Mode1 Mode2 Mode3)
```

Displayed Output

The following sections describe the output that the BGLIMM procedure produces by default. The output is organized into various tables, which are discussed in the order of their appearance.

Model- and Data-Related ODS Tables

Model Information

The "Model Information" table displays basic information about the model, such as the response variable, frequency variable, link function, and model category that the BGLIMM procedure determines on the basis of your input and options, sampling algorithm, burn-in size, simulation size, thinning, and random number seed. This table also displays the distribution of the data that PROC BGLIMM assumes. For information about the supported response distributions, see the description of the DIST= option.

Number of Observations

The "Number of Observations" table displays the number of observations that are read from the input data set and the number of observations that are used in the analysis. If the events/trials syntax is used, the number of events and trials is also displayed.

Response Profile

The "Response Profile" table displays the ordered values from which the BGLIMM procedure determines the probability that is modeled as an event in binary models. For each response category level, the frequency that is used in the analysis is reported. You can determine the ordering of the response values by specifying *response-options* in the MODEL statement. For binary models, the note that follows this table indicates which outcome is modeled as the event in binary models and which value serves as the reference category.

The "Response Profile" table is not produced for binomial (events/trials) data. You can find information about the number of events and trials in the "Number of Observations" table.

Class Level Information

The "Class Level Information" table lists the levels of every variable that you specify in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress this table completely or partially by specifying the NOCLPRINT= option in the PROC BGLIMM statement.

Random Effect Information

The "Random Effect Information" table lists some basic information, such as the subject name, number of subject levels, list of subject levels, and the covariance type, of a RANDOM statement. You should check this table to make sure that the random effects are specified correctly.

Estimate Coefficients

The "Coefficients for *label*" table, where *label* is the label that you specify in the ESTIMATE statement, displays the **L** matrix coefficients.

Least Squares Means Coefficients

The "Coefficients for *effect* Least Squares Means" table, where *effect* is the effect that you specify in the LSMEANS statement, displays the **L** matrix coefficients.

Sampling-Related ODS Tables

Parameters Initial Value

The "Parameters Initial" table (ODS table name ParametersInit) shows the initial value of each fixed regression coefficient. This table is not displayed by default. You can display it by specifying the INIT=PINIT option in the MODEL statement.

Constant Priors for Fixed Effects

The "Constant Priors for Fixed Effects" table (ODS table name ConstantCoeffPrior) shows the information for the constant prior for the fixed-effects coefficients. The table is displayed by default for a model that contains fixed effects.

Independent Normal Priors for Fixed Effects

The "Independent Normal Priors for Fixed Effects" table (ODS table name IndepNormalCoeffPrior) shows the information for the independent normal prior for the fixed-effects coefficients. The table is displayed when you specify that fixed effects have an independent normal prior.

Normal Priors for Fixed Effects

The "Normal Priors for Fixed Effects" table (ODS table name NormalCoeffPrior) shows the information for the normal prior for the fixed-effects coefficients. The table is displayed when you specify that fixed effects have a normal prior and are not independent.

Priors for Scale and Covariance Parameters

The "Priors for Scale and Covariance Parameters" table (ODS table name ScaleCovPrior) shows the prior distributions for the scale and covariance parameters. The table is displayed by default for a model that contains any scale and covariance parameters.

ODS Tables Related to Posterior Statistics

PROC BGLIMM calculates some essential posterior statistics and outputs them to a number of ODS tables. Some of the ODS tables are produced by default, and you can request others by specifying an option in the PROC BGLIMM statement. For more information about the calculations, see the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Summary and Interval Statistics

The "Posterior Summaries and Intervals" table (ODS table name PostSumInt) contains basic summary and interval statistics for each parameter. The table lists the number of posterior samples, the posterior mean and standard deviation estimates, and the highest posterior density (HPD) interval estimates. This table is displayed by default.

Summary Statistics

The "Posterior Summaries" table (ODS table name PostSummaries) contains basic statistics for each parameter. The table lists the number of posterior samples, the posterior mean and standard deviation estimates, and the percentile estimates. This table is not displayed by default. You can request it by specifying the STATISTICS=SUM option in the PROC BGLIMM statement.

Correlation Matrix

The "Posterior Correlation Matrix" table (ODS table name Corr) contains the posterior correlation of model parameters. This table is not displayed by default. You can display it by specifying the STATISTICS=CORR option in the PROC BGLIMM statement.

Covariance Matrix

The "Posterior Covariance Matrix" table (ODS table name Cov) contains the posterior covariance of model parameters. This table is not displayed by default. You can display it by specifying the STATISTICS=COV option in the PROC BGLIMM statement.

Deviance Information Criterion

The "Deviance Information Criterion" table (ODS table name DIC) contains the deviance information criterion (DIC) of the model. This table is not displayed by default. You can display it by specifying the DIC option in the PROC BGLIMM statement. For more information about the calculations, see the section "Deviance Information Criterion (DIC)" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Interval Statistics

The "Posterior Intervals" table (ODS table name PostIntervals) contains the equal-tail and highest posterior density (HPD) interval estimates for each parameter. The default α value is 0.05, and you can change it to other levels by using the STATISTICS= option in the PROC BGLIMM statement. This table is not displayed by default. You can display it by specifying the STATISTICS=INT option.

Estimated G Matrix

The "Estimated G Matrix" table (ODS table name G) contains the estimated $\mathbf{G}$ matrix. This table is not displayed by default. You can display it by specifying the G option in a RANDOM statement.

Estimated G Correlation Matrix

The "Estimated G Correlation Matrix" table (ODS table name GCorr) contains the correlation matrix that corresponds to the estimated $\mathbf{G}$ matrix. This table is not displayed by default. You can display it by specifying the GCORR option in a RANDOM statement.

Estimated R Matrix

The "Estimated R Matrix" table (ODS table name R) contains the $\mathbf{R}$ matrix for the REPEATED statement. This table is not displayed by default. You can request it by specifying the R option in a REPEATED statement.

Estimated R Correlation Matrix

The "Estimated R Correlation Matrix" table (ODS table name RCorr) contains the correlation matrix that corresponds to the estimated $\mathbf{R}$ matrix. This table is not displayed by default. You can display it by specifying the RCORR option in the REPEATED statement.

Results from ESTIMATE Statements

The "Results from ESTIMATE statements" table (ODS table name Estimates) contains basic summary and interval statistics for each linear combination of the parameters. The table lists the posterior mean and standard deviation estimates, as well as the highest posterior density (HPD) interval estimates. This table is displayed by default when you specify an ESTIMATE statement.

Least Squares Means

The "Least Squares Means" table (ODS table name LSMeans) contains basic summary and interval statistics for least squares means. The table lists the posterior mean and standard deviation estimates, as well as the highest posterior density (HPD) interval estimates. This table is displayed by default when you specify an LSMEANS statement.

Differences of Least Squares Means

The "*effect* Diffs" table (ODS table name Diffs), where *effect* is the effect that you specify in the LSMEANS statement, contains basic summary and interval statistics for differences of LS-means. The table lists the posterior mean and standard deviation estimates, as well as the highest posterior density (HPD) interval estimates. The table is displayed when you specify the DIFF option in the LSMEANS statement.

ODS Tables Related to Convergence Diagnostics

PROC BGLIMM provides convergence diagnostic tests that check for Markov chain convergence. It produces a number of ODS tables that you can display and save individually. For information about calculations, see the section "Statistical Diagnostic Tests" on page 166 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Autocorrelation

The "Autocorrelations" table (ODS table name AUTOCORR) contains the first-order autocorrelations of the posterior samples for each parameter. The Parameter column states the name of the parameter. By default, PROC BGLIMM displays lag 1, 5, 10, and 50 estimates of the autocorrelations. You can request different autocorrelations by specifying the DIAGNOSTICS=AUTOCORR(LAGS=) option in the PROC BGLIMM statement. This table is displayed by default.

Effective Sample Size

The "Effective Sample Sizes" table (ODS table name ESS) calculates the effective sample size of each parameter. For more information, see the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is displayed by default.

Monte Carlo Standard Errors

The "Monte Carlo Standard Errors" table (ODS table name MCSE) calculates the standard errors of the posterior mean estimate. For more information, see the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is not displayed by default. You can display it by specifying the DIAGNOSTICS=MCSE option in the PROC BGLIMM statement.

Geweke Diagnostics

The "Geweke Diagnostics" table (ODS table name Geweke) lists the results of the Geweke diagnostic test. For more information, see the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is not displayed by default. You can display it by specifying the DIAGNOSTICS=GEWEKE option in the PROC BGLIMM statement.

Heidelberger-Welch Diagnostics

The "Heidelberger-Welch Diagnostics" table (ODS table name Heidelberger) lists the results of the Heidelberger-Welch diagnostic test. The test consists of two parts: a stationarity test and a halfwidth test. For more information, see the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8,

"Introduction to Bayesian Analysis Procedures." This table is not displayed by default. You can display it by specifying the DIAGNOSTICS=HEIDEL option in the PROC BGLIMM statement.

Raftery-Lewis Diagnostics

The "Raftery-Lewis Diagnostics" table (ODS table name Raftery) lists the results of the Raftery-Lewis diagnostic test. For more information, see the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures." This table is not displayed by default. You can display it by specifying the DIAGNOSTICS=RAFTERY option in the PROC BGLIMM statement.

Watanabe-Akaike Information Criterion

The "Watanabe-Akaike Information Criterion" table (ODS table name WAIC) contains the Watanabe-Akaike information criterion (WAIC) of the model. This table is not displayed by default. You can display it by specifying the WAIC option in the PROC BGLIMM statement.

ODS Table Names

Each table that the BGLIMM procedure creates has a name that is associated with it. You must use this name to refer to the table when you use ODS statements. These names are listed in Table 31.16.

Table 31.16 ODS Tables Produced by PROC BGLIMM

Table Name	Description	Statement or Option
AutoCorr	Autocorrelation statistics for each parameter	DIAG=AUTOCORR
ClassLevels	Class level information	Default
Coef	**L** matrix coefficients	E option in ESTIMATE or LSMEANS
CoeffPrior	Constant prior information for fixed effects	Default
CoeffPrior	Independent normal prior information for fixed effects	COEFFPRIOR=NORMAL(VAR=*c*)
CoeffPrior	Normal prior information for fixed effects	COEFFPRIOR=NORMAL(INPUT=*SAS-data-set*)
Corr	Correlation matrix of the posterior samples	STATS=CORR
Cov	Covariance matrix of the posterior samples	STATS=COV
DIC	Deviance information criterion	DIC
Diffs	Differences of LS-means	DIFF option in LSMEANS
ESS	Effective sample size for each parameter	Default
Estimates	Results from ESTIMATE statements	ESTIMATE
G	Estimated **G** matrix	G
GCORR	Correlation matrix of the estimated **G** matrix	GCORR

Table 31.16 *continued*

Table Name	Description	Statement or Option
Geweke	Geweke diagnostics for each parameter	DIAG=GEWEKE
Heidelberger	Heidelberger-Welch diagnostics for each parameter	DIAG=HEIDEL
LSMeans	LS-means	LSMEANS
MCSE	Monte Carlo standard error for each parameter	DIAG=MCSE
ModelInfo	Model information	Default
NObs	Number of observations	Default
ParametersInit	Parameter initial values	INIT=PINIT
PostIntervals	Equal-tail and HPD intervals for each parameter	STATS=INT
PostSumInt	Posterior statistics for each parameter, including sample size, mean, standard deviation, and HPD intervals	Default
PostSummaries	Posterior statistics for each parameter, including sample size, mean, standard deviation, and percentiles	STATS=SUM
R	Estimated **R** matrix	R
Raftery	Raftery-Lewis diagnostics for each parameter	DIAG=RAFTERY
REInfo	Random Effect Information	Default
RCORR	Correlation matrix of the estimated **R** matrix	RCORR
ResponseProfile	Frequency counts for response categories	Default output in models with binary or multinomial responses
ScaleCovPrior	Priors for scale and covariance parameters	Default
WAIC	Watanabe-Akaike information criterion	WAIC

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, the BGLIMM procedure by default produces plots of the partial predictions for each spline term in the model. Use the PLOTS option in the PROC BGLIMM statement to control aspects of these plots.

PROC BGLIMM assigns a name to each graph that it creates by using ODS. You can use these names to refer to the graphs when using ODS. The names are listed in Table 31.17.

You can refer by name to every graph that is produced through ODS Graphics. The names of the graphs that PROC BGLIMM generates are listed in Table 31.17.

Table 31.17 Graphs Produced by PROC BGLIMM

ODS Graph Name	Plot Description	Option
ADPanel	Autocorrelation function and density panel	PLOTS=(AUTOCORR DENSITY)
AutocorrPanel	Autocorrelation function panel	PLOTS=AUTOCORR
AutocorrPlot	Autocorrelation function plot	PLOTS(UNPACK)=AUTOCORR
DensityPanel	Density panel	PLOTS=DENSITY
DensityPlot	Density plot	PLOTS(UNPACK)=DENSITY
TADPanel	Trace, density, and autocorrelation function panel	PLOTS=(TRACE AUTOCORR DENSITY)
TAPanel	Trace and autocorrelation function panel	PLOTS=(TRACE AUTOCORR)
TDPanel	Trace and density panel	PLOTS=(TRACE DENSITY)
TracePanel	Trace panel	PLOTS=TRACE
TracePlot	Trace plot	PLOTS(UNPACK)=TRACE

Examples: BGLIMM Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 31.1: Normal Regression with Repeated Measurements

(View the complete code for this example (bglmmex1.sas) in the example repository.)

This example illustrates how to use PROC BGLIMM to fit a repeated measures model.

The data were collected by a pharmaceutical company that examined the effects of three drugs on the respiratory capacity of patients with asthma. Treatments involved a standard drug (A), a test drug (C), and a placebo (P). Patients received each treatment on different days. The forced expiratory volume (FEV) of each patient was measured hourly for eight hours following treatment, and a baseline FEV was also recorded. Analysis in this section is based on the scenario that each patient received all three treatments on different visits. For more information about this data set (Fev1), see Littell et al. (2006, Chapter 5).

The following statements show the first few records of the Fev data set. Patient is the patient ID (total number 24), BaseVal is the baseline measurement, Drug is the drug type, Hour measures time (8 hours), and FEV is the response variable.

```
data Fev;
   input Patient Baseval Drug$ Hour FEV;
   datalines;
   201        2.46      a        1     2.68
   201        2.46      a        2     2.76
   201        2.46      a        3     2.50
   201        2.46      a        4     2.30
   201        2.46      a        5     2.14
   201        2.46      a        6     2.40
   201        2.46      a        7     2.33

   ... more lines ...

   232        2.88      p        1     3.04
   232        2.88      p        4     3.37
   232        2.88      p        5     2.69
   232        2.88      p        6     2.89
;
```

The data are unbalanced, meaning that some patients have missing measurements at certain time points. A subset of patient profiles is plotted in Output 31.1.1.

Output 31.1.1 Profile Plots from Six Patients

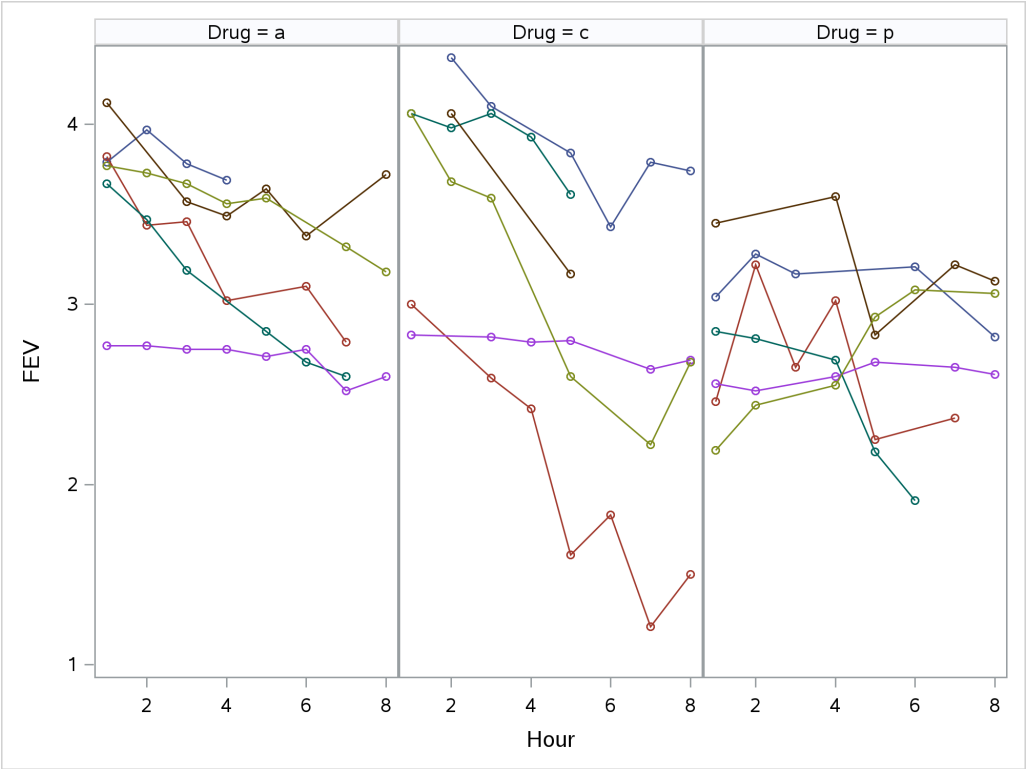

There are a number of ways that you can fit this model to account for the patient effect as well as the time (Hour) effect. For example, you can fit a random-effects model with three fixed effects (intercept, BaseVal, and Drug) and two random intercepts—one patient-level and one hour-level. This is referred to as Model 1 in this example.

Let Fev_{ijk} be the FEV value that is measured in the ith patient at jth hour who is in the (Drug = k) group, where $i = 1, \ldots, 24$, $j = 1, \ldots, 8$, and $k = \{\text{A, C, P}\}$.

$$
\begin{aligned}
\text{Fev}_{ijk} &= \mu_{ijk} + \gamma_{\text{patient},i} + \gamma_{\text{hour},j} + \epsilon_{ijk} \\
\mu_{ijk} &= \beta_0 + \beta_1 \cdot \text{baseval}_{ijk} + \beta_2 \cdot (\text{drug}_{ij,k=A}) + \beta_3 \cdot (\text{drug}_{ij,k=C}) \\
\gamma_{\text{patient},i} &\sim N(0, \sigma_{g_p^2}) \quad i = 1, \ldots, 24 \\
\gamma_{\text{hour},j} &\sim N(0, \sigma_{g_h^2}) \quad j = 1, \ldots, 8 \\
\epsilon_{ijk} &\sim N(0, \sigma_y^2)
\end{aligned}
$$

The following statements fit a model that has two random intercepts—one patient-level and one hour-level:

```
proc bglimm data=Fev nmc=50000 nbi=2000 seed=44672057
      outpost=Fev_mod1 ;
   class Drug Patient Hour;
   model FEV = BaseVal Drug;
   random int / subject=Hour covprior=uniform(lower=0,upper=2);
   random int / subject=Patient;
run;
```

The MODEL statement specifies a BaseVal effect and two Drug effects (categorical variables), and the two RANDOM statements add two random-intercept effects to the model. The prior for the shrinkage parameter of the Hour effect is taken to be a uniform(0, 2) distribution. The prior for the shrinkage parameter of the Patient effect is the default inverse gamma prior.

Output 31.1.2 show the posterior summary statistics of the fixed-effects parameters and the two random effects. The Scale parameter is the residual variance, Random1 Var is the G-side variance for the Hour effect, and Random2 Var is the G-side variance for the Patient effect.

Output 31.1.2 Posterior Summary Statistics from Model 1

The BGLIMM Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	50000	1.4844	0.3099	0.8861	2.0997
Baseval	50000	0.5008	0.1028	0.2993	0.7043
Drug a	50000	0.3237	0.0415	0.2427	0.4047
Drug c	50000	0.5245	0.0420	0.4423	0.6069
Drug p	0	.	.	.	.
Scale	50000	0.1184	0.00857	0.1016	0.1350
Random1 Var	50000	0.0600	0.0653	0.00787	0.1580
Random2 Var	50000	0.3646	0.1108	0.1853	0.5863

You can see that the Drug A effect (versus the placebo group) is 0.32 and the Drug C effect is 0.52. You can also, for example, use the ESTIMATE statement as follows to measure the effect difference between the two drug groups (the posterior mean difference is around –0.2, and the posterior standard deviation is 0.042):

```
estimate "Drug A vs C" drug 1 -1;
```

The second way of modeling these data (Model 2) is to treat the Hour effect as fixed effects, not as a random effect. And the Hour variable enters the main model as a categorical variable, as shown in the following statements:

```
proc bglimm data=Fev nmc=10000 seed=44672057
      outpost=Fev_mod2;
   class Drug Patient Hour;
   model FEV = BaseVal Drug Hour;
   random int / subject=Patient;
   estimate "A vs C" Drug 1 -1;
run;
```

This model has only one random effect (Patient). The posterior summary statistics of the fixed-effects parameters are displayed in Output 31.1.3.

Output 31.1.3 Posterior Summary Statistics from Model 2

The BGLIMM Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	10000	1.2592	0.3090	0.6002	1.8231
Baseval	10000	0.5031	0.1021	0.3111	0.7078
Drug a	10000	0.3247	0.0413	0.2475	0.4077
Drug c	10000	0.5253	0.0419	0.4412	0.6064
Drug p	0	.	.	.	.
Hour 1	10000	0.4261	0.0700	0.2869	0.5612
Hour 2	10000	0.4779	0.0703	0.3369	0.6130
Hour 3	10000	0.3679	0.0706	0.2317	0.5090
Hour 4	10000	0.2562	0.0692	0.1212	0.3926
Hour 5	10000	0.1094	0.0709	-0.0256	0.2529
Hour 6	10000	0.1140	0.0719	-0.0269	0.2540
Hour 7	10000	0.0312	0.0704	-0.1028	0.1749
Hour 8	0	.	.	.	.
Scale	10000	0.1185	0.00855	0.1031	0.1364
Random Var	10000	0.3653	0.1131	0.1815	0.5863

Although both models account for the Hour effect, they differ in the sense that the Hour random effects in Model 1 have a shared hyperprior distribution with a shrinkage parameter, whereas in Model 2 they are treated as independent fixed effects with a noninformative flat prior. Some statisticians prefer one or the other approach, because they offer different interpretations of the model. For a detailed discussion, see Gelman and Hill (2007, p. 245). The estimated effects in Model 1 and Model 2 are almost identical.

Model 3 is a minor variation of Model 2. You can consider an interaction main effect between Drug and Hour. The change in the SAS program is minor—you add a vertical bar (|) between the two effects to create two main effects and the cross effect between the two, as follows:

```
model FEV = BaseVal Drug|Hour;
```

Results from Model 3 are not displayed.

Models 1 through 3 all treat within-patient measurements as independent observations (the R-side covariance is of the VC type), which do not account for correlations. To model the longitudinal aspect of the measurements across hours, you can specify a nonidentity R-side covariance type, such as an unstructured covariance (Model 4),

$$\mathrm{Fev}_{ik} \sim \mathrm{MVN}(\boldsymbol{\mu}_{ik}, \mathbf{R})$$

where Fev_{ik} denotes all hourly observations from the ith patient who is taking drug k.

You can use the following PROC BGLIMM program to fit a repeated measures model with three fixed-effects (BaseVal, Drug, and Hour), one patient-level random effect, and an unstructured R-side covariance matrix:

```
proc bglimm data=Fev nmc=10000 seed=44672057
      outpost=Fev_mod4;
   class Drug Patient Hour;
   model FEV = BaseVal Drug Hour;
```

```
    random int / subject=Patient;
    repeated Hour / subject=Patient(Drug) type=un r rcorr;
run;
```

The REPEATED statement specifies that the Hour variable is the repeated measures variable, and an unstructured covariance is specified. The nested effects, **`Patient(Drug)`**, that are specified in the SUBJECT= option indicate that the repeats occur for each drug treatment within a patient ($3 \times 24 = 72$ in total). The R and RCORR options display the estimate $\mathbf{R}$ covariance and correlation matrices, respectively.

Output 31.1.4 displays the estimated covariance and correlation matrices.

Output 31.1.4 R Covariance and Correlation Matrices

The BGLIMM Procedure

Estimated R Matrix

Row	Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7	Col 8
1	0.3161	0.07613	0.07626	0.05901	0.03782	0.03430	0.01952	0.01276
2	0.07613	0.3073	0.07901	0.06354	0.04803	0.04112	0.02169	0.01940
3	0.07626	0.07901	0.3055	0.08042	0.06196	0.05621	0.04712	0.03937
4	0.05901	0.06354	0.08042	0.3017	0.06413	0.06189	0.05752	0.04989
5	0.03782	0.04803	0.06196	0.06413	0.3527	0.07724	0.08092	0.07187
6	0.03430	0.04112	0.05621	0.06189	0.07724	0.3405	0.07785	0.06671
7	0.01952	0.02169	0.04712	0.05752	0.08092	0.07785	0.3560	0.08926
8	0.01276	0.01940	0.03937	0.04989	0.07187	0.06671	0.08926	0.3549

Estimated R Correlation Matrix

Row	Col 1	Col 2	Col 3	Col 4	Col 5	Col 6	Col 7	Col 8
1	1.0000	0.2443	0.2454	0.1911	0.1133	0.1046	0.05821	0.03809
2	0.2443	1.0000	0.2579	0.2087	0.1459	0.1271	0.06557	0.05876
3	0.2454	0.2579	1.0000	0.2649	0.1887	0.1743	0.1429	0.1196
4	0.1911	0.2087	0.2649	1.0000	0.1966	0.1931	0.1755	0.1525
5	0.1133	0.1459	0.1887	0.1966	1.0000	0.2229	0.2283	0.2031
6	0.1046	0.1271	0.1743	0.1931	0.2229	1.0000	0.2236	0.1919
7	0.05821	0.06557	0.1429	0.1755	0.2283	0.2236	1.0000	0.2511
8	0.03809	0.05876	0.1196	0.1525	0.2031	0.1919	0.2511	1.0000

For easier visualization, you can use the correlation matrix to plot a heat map. The graph is displayed in Output 31.1.5. You can see an evenly declining correlation matrix away from the diagonal axis. This indicates stronger correlation with nearby time points, and the correlation becomes weaker as the time lag increases.

Output 31.1.5 Heat Map of the Correlation Matrix in Model 4

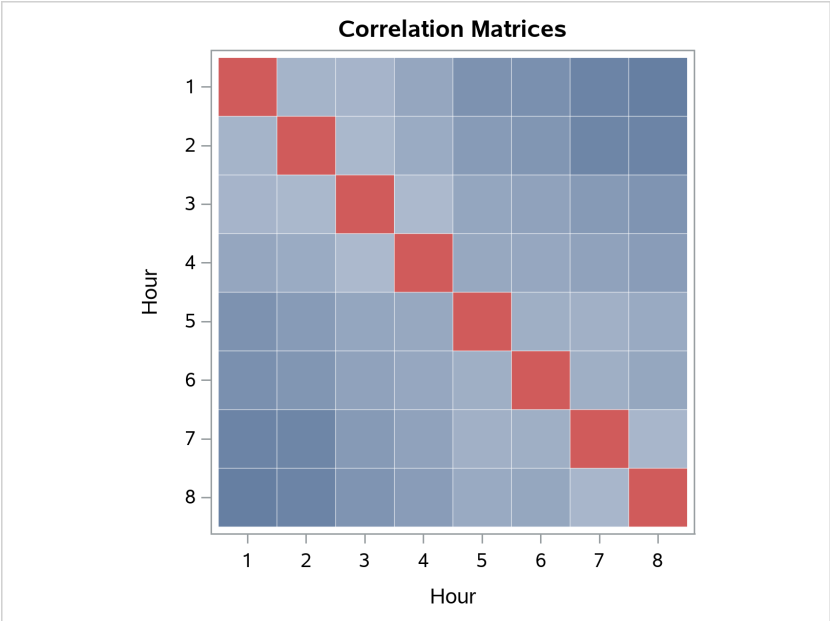

Suppose you are interested in estimating the drug response profiles (over time) of a hypothetical patient who has a known BaseVal FEV value. You can use PROC BGLIMM to estimate the predictive distribution of this patient.

The first step is to create the data set Pred, which contains the new covariates information:

```
data Pred;
   input Patient BaseVal Drug$ Hour FEV;
   datalines;
   1   2.46   a    1   .
   1   2.46   a    2   .
   1   2.46   a    3   .
   1   2.46   a    4   .
   1   2.46   a    5   .
   1   2.46   a    6   .
   1   2.46   a    7   .
   1   2.46   a    8   .
   1   2.46   c    1   .
   1   2.46   c    2   .
   1   2.46   c    3   .
   1   2.46   c    4   .
   1   2.46   c    5   .
   1   2.46   c    6   .
   1   2.46   c    7   .
   1   2.46   c    8   .
   1   2.46   p    1   .
   1   2.46   p    2   .
   1   2.46   p    3   .
   1   2.46   p    4   .
   1   2.46   p    5   .
   1   2.46   p    6   .
```

```
   1   2.46   p    7  .
   1   2.46   p    8  .
;
```

The Patient variable takes the value of 1. As long as the patient value is not a repeat of any existing Patient values in the Fev data set, PROC BGLIMM treats these observations as data from new patients. The BaseVal variable takes a value of 2.46 (the initial FEV value for this patient), the Drug variable takes three treatments, and the Hour variable goes from 1 to 8. The FEV variable takes a missing value, indicating that you want to make predictions in these scenarios.

Next, you concatenate the Pred data set with the original input data set to create a new data set to use in PROC BGLIMM. Model 4 is used here.

```
data Combined;
   set Fev Pred;
run;

proc bglimm data=Combined nmc=10000 seed=44672057
      outpost=Fev_mod4_pred;
   class Drug Patient Hour;
   model FEV = BaseVal Drug|Hour;
   random int / subject=Patient;
   repeated Hour / subject=Patient(Drug) type=un;
run;
```

PROC BGLIMM treats the 24 observations that have missing FEV values as missing data and generates predictive samples for them. The missing data variables have names that follow the convention of FEV_*n*, where *n* is the observation index of these values in the input data set. In this example, the 24 variables are named FEV_416 to FEV_439. There are 415 observation in the Fev data set.

Next, you use the %SUMINT autocall macro to compute the posterior summary statistics for the predicted values as follows. You can plot the predicted response profiles of this patient for different treatment over time. The results are shown in Output 31.1.6.

```
%sumint(data=Fev_mod4_pred, var=FEV_:, print=NO, out=pred_si)

data Comb_pred;
   merge Pred Pred_si;
   run;

proc sgpanel data=Comb_pred noautolegend;
   panelby Drug / columns=3 ;
   rowaxis label = "FEV";
   scatter x=Hour y=mean / group=Patient;
   series x=Hour y=mean / group=Patient;
run;
```

Output 31.1.6 Predicted Patient Profiles over Time for Different Treatments

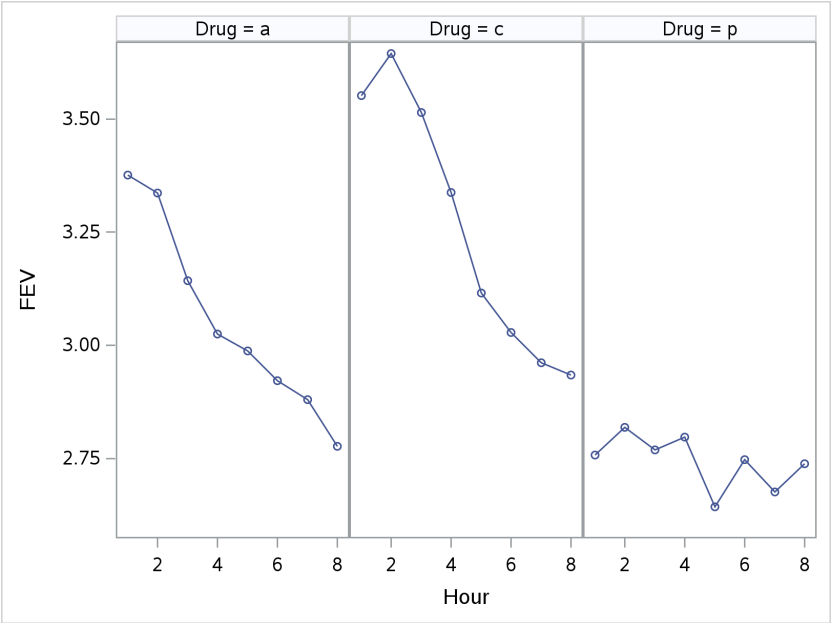

The first two panels in Output 31.1.6 both show an initial improvement, then a gradual decline of the FEV values over time. Drug C appears to have a better treatment effect than drug A. The third panel is the placebo group. It has a horizontal profile, which is what you would expect.

Example 31.2: Mating Experiment with Crossed Random Effects

(View the complete code for this example (bglmmex2.sas) in the example repository.)

This example illustrates how to use PROC BGLIMM to fit a non-nested logistic random-effects model and perform prediction-based inference.

McCullagh and Nelder (1989, Ch. 14.5) describe a mating experiment—conducted by S. Arnold and P. Verell at the University of Chicago—that involves two geographically isolated populations of mountain dusky salamanders. One goal of the experiment was to determine whether barriers to interbreeding have evolved as a result of the geographical isolation of the populations. In this case, matings within a population should be more successful than matings between the populations. The experiment, which was conducted in the summer of 1986, involved 40 animals—20 rough butt (R) and 20 whiteside (W) salamanders—with equal numbers of males and females. The animals were grouped into two sets of R males, two sets of R females, two sets of W males, and two sets of W females, so that each set contained five salamanders. Each set was mated against one rough butt and one whiteside set, creating eight crossings. Within the pairings of sets, each female salamander was paired with three male salamanders.

The following DATA step creates the data set, Salamander, for the analysis:

```
data Salamander;
   input Day Fpop$ Fnum Mpop$ Mnum Mating @@;
   datalines;
4  rb  1 rb  1 1  4  rb  2 rb  5 1
```

```
4  rb  3 rb  2 1  4  rb  4 rb  4 1
4  rb  5 rb  3 1  4  rb  6 ws  9 1
4  rb  7 ws  8 0  4  rb  8 ws  6 0
4  rb  9 ws 10 0  4  rb 10 ws  7 0
4  ws  1 rb  9 0  4  ws  2 rb  7 0
4  ws  3 rb  8 0  4  ws  4 rb 10 0

   ... more lines ...

24 ws  5 ws  7 0  24 ws  6 rb  1 0
24 ws  7 rb  5 1  24 ws  8 rb  3 0
24 ws  9 rb  4 0  24 ws 10 rb  2 0
;
```

The first observation in the first line, for example, indicates that rough butt female 1 was paired in the laboratory on day 4 of the experiment with rough butt male 1, and the pair mated. On the same day rough butt female 7 was paired with whiteside male 8, but the pair did not mate (the first observation in the fourth line).

The model that is adopted by many authors for these data contains fixed effects for gender and population. Let p_{RR}, p_{RW}, p_{WR}, and p_{WW} denote the mating probabilities between the populations, where the first subscript identifies the female partner and the second subscript identifies the male partner. Then, you model

$$\log\left\{\frac{p_{\mathrm{kl}}}{1-p_{\mathrm{kl}}}\right\} = \beta_{\mathrm{kl}} \quad k, l \in \{\mathrm{R}, \mathrm{W}\}$$

where β_{kl} denotes the average logit of mating between females of population k and males of population l.

The following statements fit a generalized model that contains fixed effects:

```
proc bglimm data=Salamander seed=725697;
   class Fpop Fnum Mpop Mnum;
   model Mating (event='1') = Fpop|Mpop / dist=binary;
run;
```

The response variable is the two-level variable Mating. Because it is coded as zeros and ones, and because by default PROC BGLIMM models the probability of the first level according to the response-level ordering, the EVENT='1' option instructs PROC BGLIMM to model the probability of a successful mating. The distribution of the mating variable is binary.

The first table in the output is the "Model Information" table, shown in Output 31.2.1. This table displays basic information about the data and model, such as the name of the input data set, response variable, distribution, link function, sampling algorithm, burn-in size, simulation size, thinning number, and random number seed. The random number seed initializes the random number generators. If you repeat the analysis and use the same seed, you get an identical stream of random numbers.

Output 31.2.1 Model Information

The BGLIMM Procedure

Model Information	
Data Set	WORK.SALAMANDER
Response Variable	Mating
Distribution	Binary
Link Function	Logit
Fixed Effects Included	Yes
Random Effects Included	No
Sampling Algorithm	Gamerman
Burn-In Size	500
Simulation Size	5000
Thinning	1
Random Number Seed	725697
Number of Threads	1

The "Class Level Information" table in Output 31.2.2 displays the levels of variables that are listed in the CLASS statement. Note that there are two female populations and two male populations.

Output 31.2.2 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
Fpop	2	rb ws
Fnum	10	1 2 3 4 5 6 7 8 9 10
Mpop	2	rb ws
Mnum	10	1 2 3 4 5 6 7 8 9 10

Output 31.2.3 displays the "Number of Observations" table. All 120 observations in the data set are used in the analysis. For data sets that have missing or invalid values, the number of observations that are used might be less than the number of observations that are read.

Output 31.2.3 Number of Observations

Number of Observations	
Number of Observations Read	120
Number of Observations Used	120

The "Response Profile" table, which is displayed only for binary data, lists the levels of the response variable and their order (Output 31.2.4). The table also provides information about which level of the response variable defines the event. Because of the EVENT='1' response variable option in the MODEL statement, the probability that is being modeled is that of the higher-ordered value.

Output 31.2.4 Response Profile

Response Profile		
Ordered Value	Mating	Total Frequency
1	0	50
2	1	70

Probability modeled is Mating ='1'.

PROC BGLIMM reports posterior summary statistics (posterior means, standard deviations, and highest posterior density (HPD) intervals) for each parameter in the "Posterior Summaries and Intervals" table, as shown in Output 31.2.5.

Output 31.2.5 Posterior Summary Statistics

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	5000	0.8845	0.4082	0.1391	1.7387
Fpop rb	5000	-0.1674	0.5659	-1.2915	0.8939
Fpop ws	0	.	.	.	.
Mpop rb	5000	-2.1235	0.5964	-3.3360	-1.0176
Mpop ws	0	.	.	.	.
Fpop rb*Mpop rb	5000	2.4627	0.8352	0.8322	4.0574
Fpop rb*Mpop ws	0	.	.	.	.
Fpop ws*Mpop rb	0	.	.	.	.
Fpop ws*Mpop ws	0	.	.	.	.

There are four levels in the Fpop*Mpop effect. An intercept is included by default in the fixed effects, so the last level of the Fpop*Mpop effect (pairing of a whiteside female and a whiteside male) is not estimable (this explains why the estimate for the last level is missing). The estimate for the intercept gives the logit for the last level of Fpop*Mpop effect—that is, the pairing of a whiteside female and a whiteside male. And the estimates for other three pairings are the deviates from the logit for the pairing of a whiteside female and a whiteside male. For example, the posterior mean of the estimate for "Fpop rb*Mop rb" is 0.16, meaning that the logit for the pairing of a rough butt female and a rough butt male is 0.16 higher than the logit for the pairing of a whiteside female and a whiteside male.

In a pairing with a female whiteside salamander, the logit drops sharply by 2.12 when the female is paired with a whiteside male from a different population instead of a male from her own population. If the same comparisons are made in pairs with whiteside males, then you also notice a drop of 0.17 in the logit if the female comes from a different population from that of the male. However, the change is considerably less.

This model can be fitted using a logistic regression with two random effects:

$$p_i = \frac{1}{1 + \exp(-(\beta_0 + \beta_1 \cdot \mathrm{Fpop_{rb}Mpop_{ws}} + \beta_2 \cdot \mathrm{Fpop_{ws}Mpop_{rb}} + \beta_3 \cdot \mathrm{Fpop_{rb}Mpop_{rb}} + \gamma_f + \gamma_m))}$$

$$\gamma_f \sim N(0, \sigma_f^2)$$

$$\gamma_m \sim N(0, \sigma_m^2)$$

This represents a logistic regression model that has a fixed intercept, an interaction, one random effect for the female group (for $f = 1, \ldots, 20$), and one random effect for the male group (for $m = 1, \ldots, 20$). You can use the following statements to fit the model:

```
proc bglimm data=salamander nmc=20000 seed=901214;
   class Fpop Fnum Mpop Mnum;
   model Mating (event='1') = Fpop|Mpop / dist=binary;
   random int / sub=Fpop*Fnum;
   random int / sub=Mpop*Mnum;
run;
```

The Fpop*Fnum effect in the first RANDOM statement creates a random intercept for each female animal. Because Fpop and Fnum are CLASS variables, the effect has 20 levels (10 rough butt and 10 whiteside females). Similarly, the Mpop*Mnum effect in the second RANDOM statement creates the random intercepts for the male animals. No TYPE= option is specified in the RANDOM statement, so the covariance structure defaults to TYPE=VC. The random effects and their levels are independent, and each effect has its own variance component. Because the conditional distribution of the data, conditioned on the random effects, is binary, no extra scale parameter (ϕ) is added. The RANDOM statements in PROC BGLIMM take two interaction effects: Fpop*Fnum and Mpop*Mnum. The SUB= option in the RANDOM statement in PROC BGLIMM supports a syntax for interaction effects.

The "Posterior Summaries and Intervals" table is shown in Output 31.2.6.

Output 31.2.6 Posterior Summary Statistics

The BGLIMM Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	20000	1.3187	0.7770	-0.2395	2.8228
Fpop rb	20000	-0.2836	0.9660	-2.1719	1.6527
Fpop ws	0	.	.	.	.
Mpop rb	20000	-3.1839	0.9695	-5.1256	-1.3199
Mpop ws	0	.	.	.	.
Fpop rb*Mpop rb	20000	3.7212	1.1306	1.4989	5.9243
Fpop rb*Mpop ws	0	.	.	.	.
Fpop ws*Mpop rb	0	.	.	.	.
Fpop ws*Mpop ws	0	.	.	.	.
Random1 Var	20000	2.1019	1.3387	0.3421	4.6481
Random2 Var	20000	1.0683	0.6602	0.2376	2.3319

The parameter labeled "Random1 Var" measures the variability among the 20 females (10 rough butt and 10 whiteside); the parameter labeled "Random2 Var" measures the variability among the 20 males (10 rough butt and 10 whiteside). It seems that there is more variability among the females than among the males.

In the previous program, two RANDOM statements are used to specify two random intercepts. Alternatively, you can combine the two SUBJECT= variables into a single RANDOM statement, as follows:

```
random Fpop*Fnum Mpop*Mnum;
```

This is an equivalent model specification. However, the posterior sampling sequence will be different: in programs that use the SUBJECT= option, the random-effects parameters are updated in sequence; if you omit the SUBJECT= option, all random effects from that RANDOM statement are updated together. Therefore, the posterior samples will be different, and you can also experience mixing and performance differences, depending on the model and the number of random-effects parameters involved.

The following ESTIMATE statements compute the four levels of the Fpop*Mpop effect, which are labeled as **"rb and rb"** and so on:

```
proc bglimm data=salamander nmc=20000 seed=901214 outpost=Sal_Out;
   class Fpop Fnum Mpop Mnum;
   model Mating (event='1') = Fpop|Mpop / dist=binary;
   random int / sub=Fpop*Fnum;
   random int / sub=Mpop*Mnum;
   estimate "rb and rb" Int 1 Fpop 1 0 Mpop 1 0 Fpop*Mpop 1;
   estimate "rb and ws" Int 1 Fpop 1 0 Mpop 0 1 Fpop*Mpop 0 1;
   estimate "ws and rb" Int 1 Fpop 0 1 Mpop 1 0 Fpop*Mpop 0 0 1;
   estimate "ws and ws" Int 1 Fpop 0 1 Mpop 0 1 Fpop*Mpop 0 0 0 1;
run;
```

These are some linear combinations of the fixed-effects parameters, which, if you choose not to use the ESTIMATE statement, you replicate by using the following DATA step manipulations:

```
data Post;
   set Sal_Out;
   rr = Intercept + Fpop_rb + Mpop_rb + Fpop_rb_Mpop_rb;
   rw = Intercept + Fpop_rb;
   wr = Intercept + Mpop_rb;
   ww = Intercept;
   drop Intercept__:;
run;
```

The RR variable in the Post data set is identical to what the first ESTIMATE statement produces. You can check by comparing RR with rb_and_rb, which was created by the first ESTIMATE statement in the earlier program.

What is more relevant is the need to transfer these linear combinations back to the scale of the data and to estimate different mating probabilities: p_{RR}, p_{RW}, p_{WR}, and p_{WW}. If the fitted model is a fixed-effects model, as in

$$\log\left\{\frac{p_{\mathrm{kl}}}{1-p_{\mathrm{kl}}}\right\} = \beta_{\mathrm{kl}} \quad \mathrm{k,l} \in \{\mathrm{R,W}\}$$

then the transformation back to p is simple:

$$p_{\mathrm{kl}} = \frac{\exp(\beta_{\mathrm{kl}})}{1+\exp(\beta_{\mathrm{kl}})}$$

And the programming is straightforward:

```
data Prob;
   set Post;
   p_rr_f = logistic(rr);
   p_rw_f = logistic(rw);
   p_wr_f = logistic(wr);
   p_ww_f = logistic(ww);
run;
```

But because the fitted model is a random-effects model,

$$\log\left\{\frac{p_{\mathrm{kl}}}{1-p_{\mathrm{kl}}}\right\} = \beta_{\mathrm{kl}} + \gamma_f + \gamma_m \quad \mathrm{k,l} \in \{\mathrm{R,W}\}$$

The transformation must account for the variability in the random effects. In other words, you must integrate out the uncertainties of the γ_f and γ_m in order to obtain the marginal posterior distribution of the p_{kl} parameters:

$$\pi(p_{\mathrm{kl}}|\mathrm{data}) = \int_{\gamma_f}\int_{\gamma_m} \pi(p_{\mathrm{kl}}, \gamma_f, \gamma_m|\mathrm{data}) \cdot \pi(\gamma_f) \cdot \pi(\gamma_m) d\gamma_m d\gamma_f$$

Most frequently and most conveniently, $\pi(\gamma_f)$ and $\pi(\gamma_m)$ are taken to be the distribution of the random effects, given the posterior samples of σ_f^2 and σ_m^2.

The following DATA step program uses the Monte Carlo method to integrate out the random effects and compute the marginal posterior of the mating probabilities:

```
data Prob;
   set Post;
   call streaminit(87925);
   g_f = rand("normal", 0, sqrt(random1_var));
   g_m = rand("normal", 0, sqrt(random2_var));
   p_rr = logistic(rr + g_f + g_m);
   g_f = rand("normal", 0, sqrt(random1_var));
   g_m = rand("normal", 0, sqrt(random2_var));
   p_rw = logistic(rw + g_f + g_m);
   g_f = rand("normal", 0, sqrt(random1_var));
   g_m = rand("normal", 0, sqrt(random2_var));
   p_wr = logistic(wr + g_f + g_m);
   g_f = rand("normal", 0, sqrt(random1_var));
   g_m = rand("normal", 0, sqrt(random2_var));
   p_ww = logistic(ww + g_f + g_m);
   drop g_f g_m;
run;
```

The CALL STREAMINIT statement ensures reproducibility; the RAND call draws samples from the random-effects prior distribution, conditional on posterior samples of σ_f^2 and σ_m^2. You compute the mean and apply the logistic transformation. This process is repeated for the computation of each individual probability. The g_f and g_m are temporary symbols that do not need to be saved after the sampling and computation are done.

The Prob data set contains 20,000 draws of p_rr, p_rw, p_wr, and p_ww, which are estimates of the four mating probabilities, respectively. You can use the following program to display the posterior distributions of the four mating probabilities:

```
proc sgplot data=Prob noborder;
  density p_rr / type=kernel legendlabel='p(rb & rb)'
     lineattrs=(pattern=solid);
  density p_rw / type=kernel legendlabel='p(rb & ws)'
     lineattrs=(pattern=ShortDash);
  density p_wr / type=kernel legendlabel='p(ws & rb)'
     lineattrs=(pattern=DashDotDot);
  density p_ww / type=kernel legendlabel='p(ws & ws)'
     lineattrs=(pattern=LongDash);
  keylegend / location=inside position=topright across=1;
  xaxis label="Probability";
  yaxis display=(nolabel noline noticks novalues);
run;
```

The posterior distributions of the mating probabilities are shown in Output 31.2.7. The probability of a female whiteside mating with a male rough butt (the dash-dotted line) is significantly lower than the other three probabilities. The smoothness at the boundaries (probability less than 0 or greater than 1) is an artifact of the kernel method that is used in the SGPLOT procedure.

Output 31.2.7 Marginal Posterior Distributions of Mating Probability

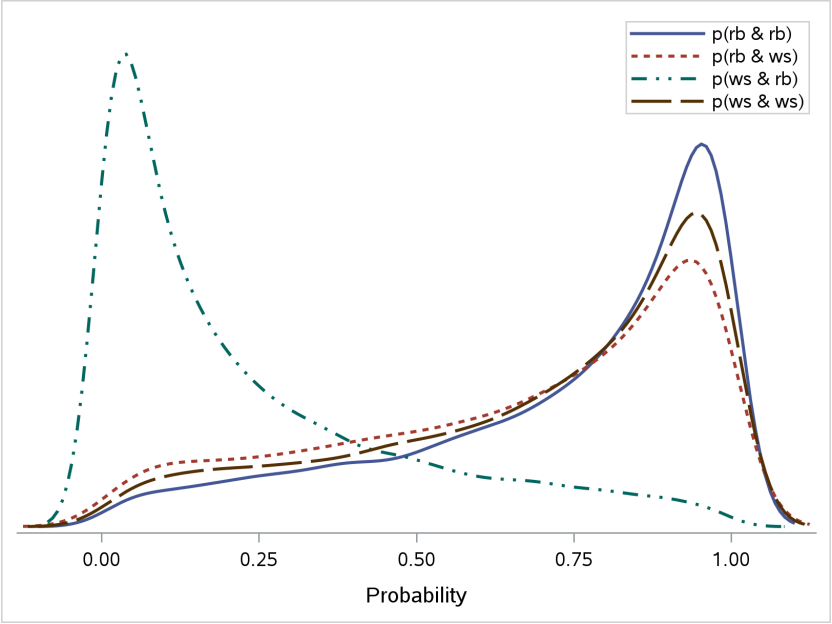

Considering the posterior distributions of these probabilities, you can make additional inferences rather easily. For example, you can estimate that the probability of a female rough butt mating with a male rough butt is greater than the probability of a female whiteside mating with a male whiteside.

To compare, Output 31.2.8 shows the posterior distributions of the mating probabilities by using logistic transformation on the fixed-effects parameters only, without accounting for the variability of the random effects. These distributions are clearly different from those shown in Output 31.2.7, with noticeably thinner tails, precisely because they do not account for additional variability in the model.

Output 31.2.8 Posterior Distributions of Mating Probability without Accounting for Random Effects

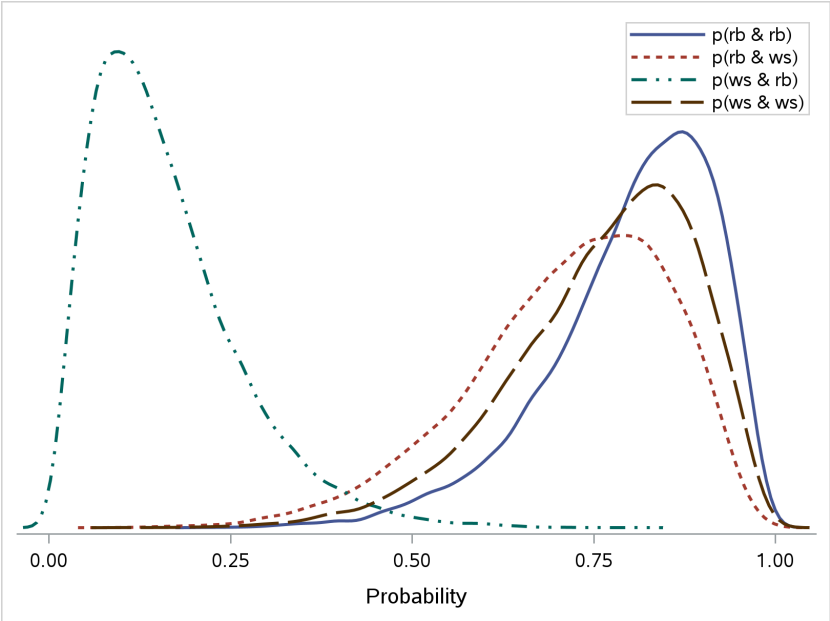

After the PREDDIST statement becomes available, you can obtain the posterior predictive distribution of the response and its related quantities for given covariates more conveniently. Suppose you have a data set that contains the covariates (design matrix) for the four mating groups:

```
data ForPreddist;
   input Fpop$ Mpop$ Fnum Mnum;
   datalines;
   rb  rb  11  11
   rb  ws  11  11
   ws  rb  11  11
   ws  ws  11  11
;
```

The Fpop and Mpop defines the four female and male combinations. The Fnum and Mnum variables have the value of 11, which is different from their values (1 to 10) in the DATA= data set salamander and this makes a new subject level for the RANDOM effects. PROC BGLIMM will treat these observations as data from a new subject and draw random effects from the prior (normal distribution with the current sampled variance) and use those samples to predict for the responses.

```
proc bglimm data=salamander nmc=20000 seed=901214 outpost=Sal_Out;
   class Fpop Fnum Mpop Mnum;
   model Mating (event='1') = Fpop|Mpop / dist=binary;
   random int / sub=Fpop*Fnum s;
   random int / sub=Mpop*Mnum s;
   preddist covariates=ForPreddist outpred=Preddist_Out ilink;
run;

%sumint(data=Preddist_Out, var=Mating: Ilink:)
```

In the PREDDIST statement, the OUTPRED= option creates a new SAS data set Preddist_Out that contains random samples from the posterior predictive distribution of the response variable and the ILINK option

computes the inverse link function of the linear predictor for each of the four mating groups, which is the mating probability. The %SUMINT autocall macro is used to compute the posterior summary statistics for the predicted responses and mating probabilities for the four groups. The results (not shown) are very similar to those obtained previously for the mating probabilities: p_{RR}, p_{RW}, p_{WR}, and p_{WW}.

Example 31.3: Poisson Regression with Random Effects

(View the complete code for this example (bglmmex3.sas) in the example repository.)

This example illustrates how to use PROC BGLIMM to fit a Poisson random-effects model with offsets. The example also discusses the use of the deviance information criterion (DIC) as a way to evaluate the fit of a model.

Clayton and Kaldor (1987, Table 1) present data on observed and expected cases of lip cancer in the 56 counties of Scotland between 1975 and 1980. The expected number of cases was determined by a separate multiplicative model that accounted for the age distribution in the counties. The goal of the analysis is to estimate the county-specific log-relative risks, also known as standardized mortality ratios (SMRs).

If y_i is the number of incident cases in county i and E_i is the expected number of incident cases, then the ratio of observed to expected counts, y_i/E_i, is the standardized mortality ratio. Clayton and Kaldor (1987) assume that there exists a relative risk, λ_i, that is specific to each county and is a random variable. Conditional on λ_i, the observed counts are independent Poisson variables with mean $E_i\lambda_i$.

An elementary mixed model for λ_i specifies only a random intercept for each county, in addition to a fixed intercept. Breslow and Clayton (1993), in their analysis of these data, also provide a covariate that measures the percentage of the population who work in agriculture, fishing, and forestry and therefore spend a lot of time exposed to sunlight. The expanded model for the region-specific relative risk in Breslow and Clayton (1993) is

$$\lambda_i = \exp\{\beta_0 + \beta_1 \mathrm{Employment}_i/10 + \gamma_i\}, \quad i = 1, \ldots, 56$$

where β_0 and β_1 are fixed effects and γ_i are county random effects.

Because the mean of the Poisson variates, conditional on the random effects, is $\mu_i = E_i\lambda_i$, applying a log link yields

$$\log\{\mu_i\} = \log\{E_i\} + \beta_0 + \beta_1 \mathrm{Employment}_i/10 + \gamma_i$$

The term $\log\{E_i\}$ is an offset, a regressor variable whose coefficient is known to be 1. Note that it is assumed that the E_i are known; they are not treated as random variables.

The following DATA step creates the data set LipCancer:

```
data LipCancer;
   input County Observed Expected Employment SMR;
   if (Observed > 0) then ExpCount = 100*Observed/SMR;
   else ExpCount = Expected;
   x    = Employment / 10;
   LogN = log(ExpCount);
   datalines;
 1  9  1.4 16 652.2
 2 39  8.7 16 450.3
```

```
 3 11  3.0 10 361.8
 4  9  2.5 24 355.7
 5 15  4.3 10 352.1
 6  8  2.4 24 333.3
 7 26  8.1 10 320.6
 8  7  2.3  7 304.3
 9  6  2.0  7 303.0

   ... more lines ...

53  1  5.7  1  17.4
54  1  7.0  1  14.2
55  0  4.2 16   0.0
56  0  1.8 10   0.0
;
```

The expected number of cases, ExpCount, is based on the observed standardized mortality ratio for counties that have lip cancer cases and based on the expected counts that are reported by Clayton and Kaldor (1987, Table 1) for the counties without such cases. The sum of the expected counts then equals the sum of the observed counts. The offset is created in the DATA step by using the following assignment statement:

```
LogN = log(ExpCount);
```

In addition, in the DATA step, you use the following statement to transform the covariate that measures the percentage of employment in agriculture, fisheries, and forestry to agree with the analysis of Breslow and Clayton (1993):

```
x = Employment / 10;
```

The following statements fit this Poisson model with an offset and a random intercept for each county:

```
proc bglimm data=LipCancer seed=10571042 nmc=10000
   outpost=LipCancer_Out;
   class County;
   model Observed = x / dist=poisson offset=LogN;
   random int / sub=County;
run;
```

In the MODEL statement, DIST=POISSON specifies that the response variable has a Poisson distribution with an offset, LogN. The offset is associated with the linear predictor through the OFFSET=LOGN option in the MODEL statement.

Along with the MODEL statement for fixed effects, the RANDOM statement is used for random effects. It specifies that the linear predictor contains an intercept term that randomly varies at the level of the County effect. The SUBJECT=COUNTY option in the RANDOM statement defines County as a subject index for the random-effects grouping, so that each county has its own intercept.

The “Model Information” table, the “Class Level Information” table, and the “Number of Observations” table are shown in Output 31.3.1. Note that there are 56 counties.

Output 31.3.1 Model Information

The BGLIMM Procedure

Model Information	
Data Set	WORK.LIPCANCER
Response Variable	Observed
Distribution	Poisson
Link Function	Log
Fixed Effects Included	Yes
Random Effects Included	Yes
Sampling Algorithm	Gamerman, Conjugate
Burn-In Size	500
Simulation Size	10000
Thinning	1
Random Number Seed	10571042
Number of Threads	1

Class Level Information

Class	Levels	Values
County	56	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56

Number of Observations	
Number of Observations Read	56
Number of Observations Used	56

The "Posterior Summaries and Intervals" table in Output 31.3.2 displays the estimates of β_0 and β_1 along with their standard deviations. The covariate that measures employment percentage in agriculture, fisheries, and forestry does not have zero in the 95% HPD interval, indicating that this covariate might be a surrogate for the exposure to sunlight, an important risk factor for lip cancer.

Output 31.3.2 Posterior Summaries and Intervals

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	10000	-0.5069	0.1712	-0.8502	-0.1835
x	10000	0.6900	0.1585	0.3785	0.9989
Random Var	10000	0.4763	0.1217	0.2702	0.7213

Looking at the estimate of the variance of the region-specific log-relative risks, you can see significant county-to-county heterogeneity in risks. If the covariate were removed from the analysis, as in Clayton and Kaldor (1987), the heterogeneity in county-specific risks would increase. (The fitted SMRs in Table 6 of Breslow and Clayton (1993) were obtained without the covariate x in the model.)

You can use the posterior data set LipCancer_Out to compute the predicted SMR value, which is

$$\mathrm{SMR}_i = 100 \cdot \exp(\beta_0 + \beta_1 x_i + \gamma_i)$$

where i denotes the county, β_0 is the intercept, β_1 is the parameter for the covariate x, and γ_i is the random effect for county i. The following DATA step computes the predicted SMR for each county and saves them in the data set SMR_Pred:

```
data SMR_PRED;
   array gamma[56] Intercept__County_1-Intercept__County_56;
   array SMR_pred[56];
   set LipCancer_Out;
      do i = 1 to 56;
         set LipCancer(rename=(x=data_x)) point=i;
         SMR_pred[i] = 100 * exp(Intercept + x * data_x + gamma[i]);
      end;
   keep smr_pred:;
   run;
```

You can compute the mean estimates for each SMR_Pred and plot them against the observed SMR values (Output 31.3.3) to get a sense of how the model fits the data, as follows:

```
%sumint(data=SMR_PRED, var=_numeric_, print=NO, out=SMR_SI)

data combine;
   merge LipCancer SMR_SI;
run;

proc sgplot data=combine noautolegend aspect=1;
   yaxis label="Predicted SMR" max=700;
   xaxis label="Observed SMR" max=700;
   text x=SMR y=mean text=employment;
   lineparm x=0 y=0 slope=1;
run;
```

Output 31.3.3 Observed and Predicted SMRs; Data Labels Indicate Employment Types

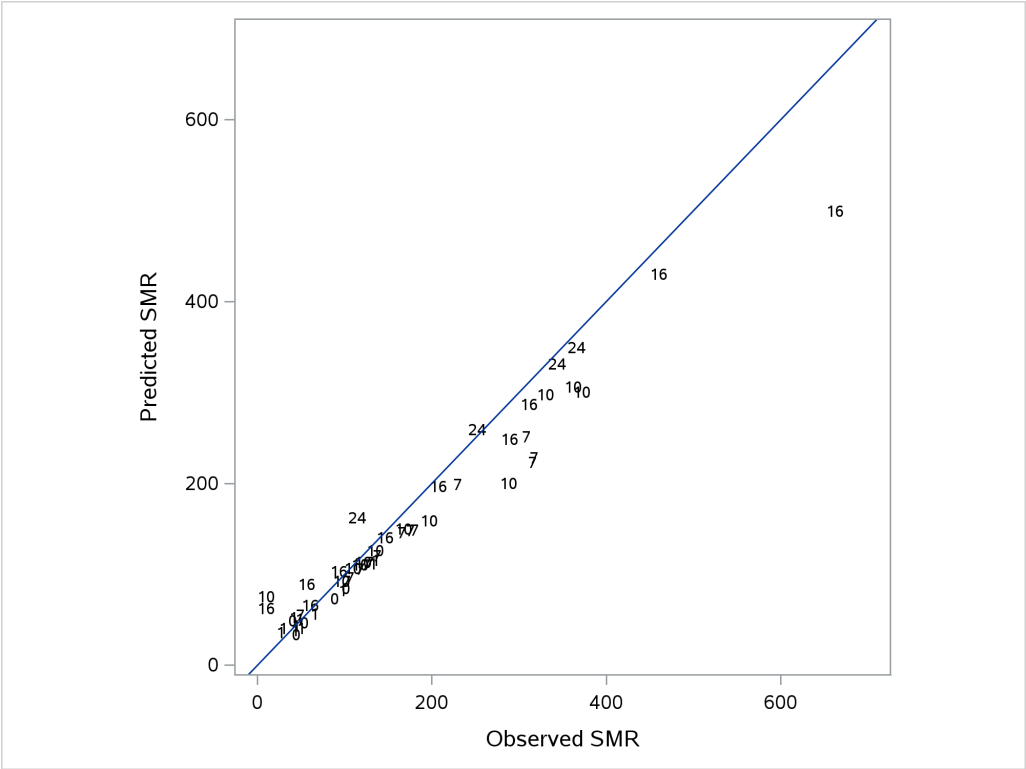

In contrast, you can fit a fixed-effects-only model to the data by using the following program (results not shown):

```
proc bglimm data=LipCancer seed=10571042 nmc=10000
   outpost=LipCancer_fOut;
   class County;
   model Observed = x / dist=poisson offset=LogN;
run;
```

The fixed-effects model provides a worse fit than the random-effects model (Output 31.3.4).

Output 31.3.4 Observed and Predicted SMRs; Fixed-Effects-Only Model

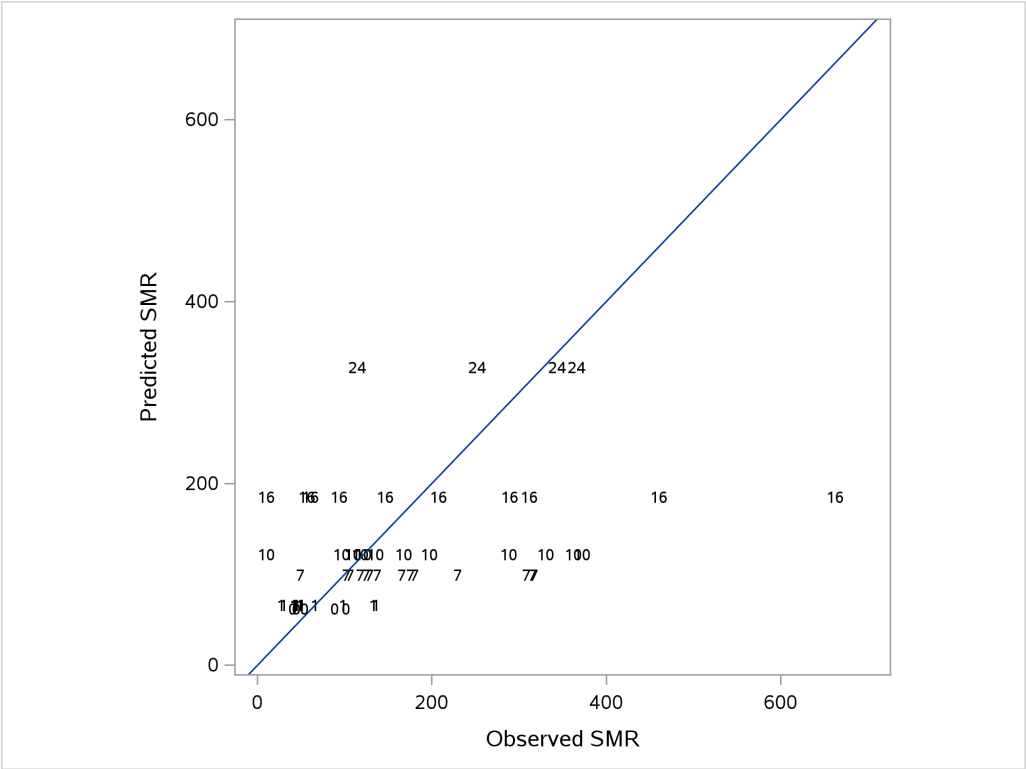

Another way to understand the fit of a model is to use the deviation information criterion (DIC), which you can specify by using the DIC option in the PROC BGLIMM call, as follows:

```
proc bglimm data=LipCancer seed=10571042 nmc=10000 DIC;
   class County;
   model Observed = x / dist=poisson offset=LogN;
   random int / sub=County;
run;

proc bglimm data=LipCancer seed=10571042 nmc=10000 DIC;
   class County;
   model Observed = x / dist=poisson offset=LogN;
run;
```

The DIC values are shown in Output 31.3.5 (random-effects model) and in Output 31.3.6 (fixed-effects-only model). The discrepancy suggests that the model that includes county-specific adjustments fits the data much better.

Output 31.3.5 DIC Values from Random-Effects Model

The BGLIMM Procedure

Deviance Information Criterion	
Dbar (Posterior Mean of Deviance)	267.919
Dmean (Deviance Evaluated at Posterior Mean)	226.337
pD (Effective Number of Parameters)	41.582
DIC (Smaller is Better)	309.501

Output 31.3.6 DIC Values from Fixed-Effects-Only Model

The BGLIMM Procedure

Deviance Information Criterion	
Dbar (Posterior Mean of Deviance)	449.038
Dmean (Deviance Evaluated at Posterior Mean)	447.023
pD (Effective Number of Parameters)	2.015
DIC (Smaller is Better)	451.053

You can further extend the model by including another random effect, the Employment-level random-intercept effect that accounts for employment-related variability, as follows:

```
proc bglimm data=LipCancer seed=10571042 nmc=10000;
   class County Employment;
   model Observed = x / dist=poisson offset=LogN;
   random int / sub=Employment;
   random int / sub=County;
run;
```

This type of overparameterized model can often encounter convergence difficulties and might not provide any more meaningful fitting than County-level random-effects models. Note that PROC BGLIMM produces an error message if the two RANDOM statements are switched; the procedure requires that the observation-level random effects be specified last.

Example 31.4: Repeated Growth Measurements with Group Difference

(View the complete code for this example (bglmmex4.sas) in the example repository.)

This example illustrates how to model heterogeneity in covariance structures by using grouped growth measurements data.

Changes in the distance (measured in mm) from the center of the pituitary gland to the pterygomaxillary fissure near the teeth are important in orthodontic therapy. The dental study of Pothoff and Roy (1964) consists of growth measurements of 27 children (11 girls and 16 boys) at ages 8, 10, 12, and 14.

The following DATA step creates the data set, PR, for the analysis:

```
data pr;
   input Person Gender $ y1 y2 y3 y4;
   Distance=y1; Age=8;  Time=1; output;
   Distance=y2; Age=10; Time=2; output;
   Distance=y3; Age=12; Time=3; output;
   Distance=y4; Age=14; Time=4; output;
   drop y1-y4;
   datalines;
 1   F   21.0    20.0    21.5    23.0
 2   F   21.0    21.5    24.0    25.5
 3   F   20.5    24.0    24.5    26.0
 4   F   23.5    24.5    25.0    26.5
 5   F   21.5    23.0    22.5    23.5
 6   F   20.0    21.0    21.0    22.5
```

```
 7   F   21.5    22.5    23.0    25.0
 8   F   23.0    23.0    23.5    24.0
 9   F   20.0    21.0    22.0    21.5
10   F   16.5    19.0    19.0    19.5
11   F   24.5    25.0    28.0    28.0
12   M   26.0    25.0    29.0    31.0
13   M   21.5    22.5    23.0    26.5
14   M   23.0    22.5    24.0    27.5
15   M   25.5    27.5    26.5    27.0
16   M   20.0    23.5    22.5    26.0
17   M   24.5    25.5    27.0    28.5
18   M   22.0    22.0    24.5    26.5
19   M   24.0    21.5    24.5    25.5
20   M   23.0    20.5    31.0    26.0
21   M   27.5    28.0    31.0    31.5
22   M   23.0    23.0    23.5    25.0
23   M   21.5    23.5    24.0    28.0
24   M   17.0    24.5    26.0    29.5
25   M   22.5    25.5    25.5    26.0
26   M   23.0    24.5    26.0    30.0
27   M   22.0    21.5    23.5    25.0
;
```

Questions of interest include the following:

- Does the distance from the center of the pituitary gland to the pterygomaxillary fissure change over time?
- What is the pattern of change?
- Does the pattern of change differ between boys and girls?
- How correlated are the repeated measurements across age for the same subject?
- Is the correlation similar for boys and girls?

Output 31.4.1 shows the distance growth plots for boys and girls as they age. The plots show apparent tracking (for children who start with small distance, the distances tend to stay small) and approximately linear growth. However, the growth rates are noticeably different for boys (faster) and girls (slower).

Output 31.4.1 Distance Growth Plot for Dental Data

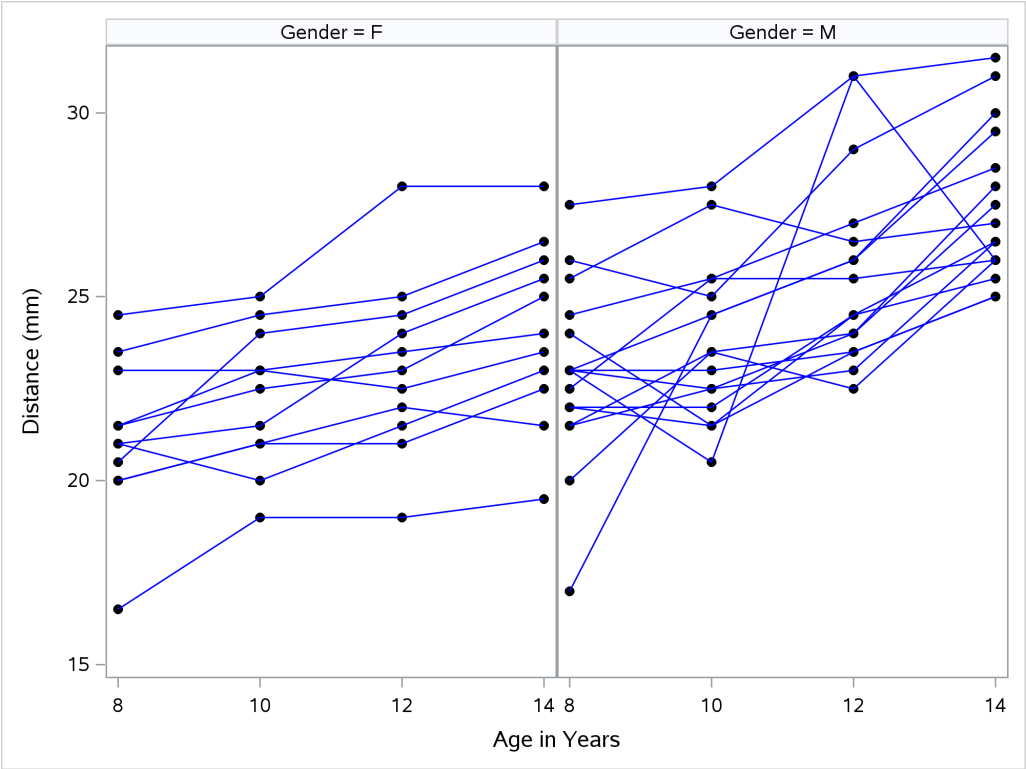

You can fit a linear repeated measures model to the data: the Age variable is the fixed-effect covariate, and the repeats are over the variable Time. The BY statement offers a convenient way to perform a separate and independent analysis between the boys' group and the girls' group. This accounts for different growth profiles for the two groups, because correlation can differ by gender.

The following PROC BGLIMM program fits two linear repeated measures models to the data:

```
proc bglimm data=pr seed=475193 outpost=pr_out;
   by Gender;
   class Person Time;
   model Distance = Age;
   repeated Time / subject=Person type=un r;
run;
```

The MODEL statement specifies the response variable Distance and the covariate Age, which models an overall linear growth trend over time.

The REPEATED statement specifies that the repeated measurements be taken over the Time variable. The repeated effect is required in a REPEATED statement, and it must be specified as a CLASS variable. The TYPE=UN option specifies an unstructured covariance for the residuals. The repeated measurements are grouped according to Person (the SUBJECT= variable), and the covariance type is unstructured. The R option displays the estimated covariance matrix of **R**.

Output 31.4.2 and Output 31.4.3 show the estimate covariance matrix for girls and boys, respectively. The point estimates are different, and you can get a more accurate sense of the degree of differences by plotting the posterior density estimates. This is shown in Output 31.4.4.

Output 31.4.2 Estimated Covariance Matrix of $\mathbf{R}$ for Girls

The BGLIMM Procedure

Gender=F

Estimated R Matrix				
Row	**Col 1**	**Col 2**	**Col 3**	**Col 4**
1	4.2652	2.8200	3.5671	3.5578
2	2.8200	3.5672	3.3555	3.3955
3	3.5671	3.3555	5.1359	4.5170
4	3.5578	3.3955	4.5170	5.4541

Output 31.4.3 Estimated Covariance Matrix of $\mathbf{R}$ for Boys

Gender=M

Estimated R Matrix				
Row	**Col 1**	**Col 2**	**Col 3**	**Col 4**
1	5.6814	1.9663	3.1060	1.4269
2	1.9663	4.4751	2.0155	2.4799
3	3.1060	2.0155	6.4496	2.8814
4	1.4269	2.4799	2.8814	4.2419

Output 31.4.4 Posterior Density Comparison of the $\mathbf{R}$ Covariance—Independent Analysis

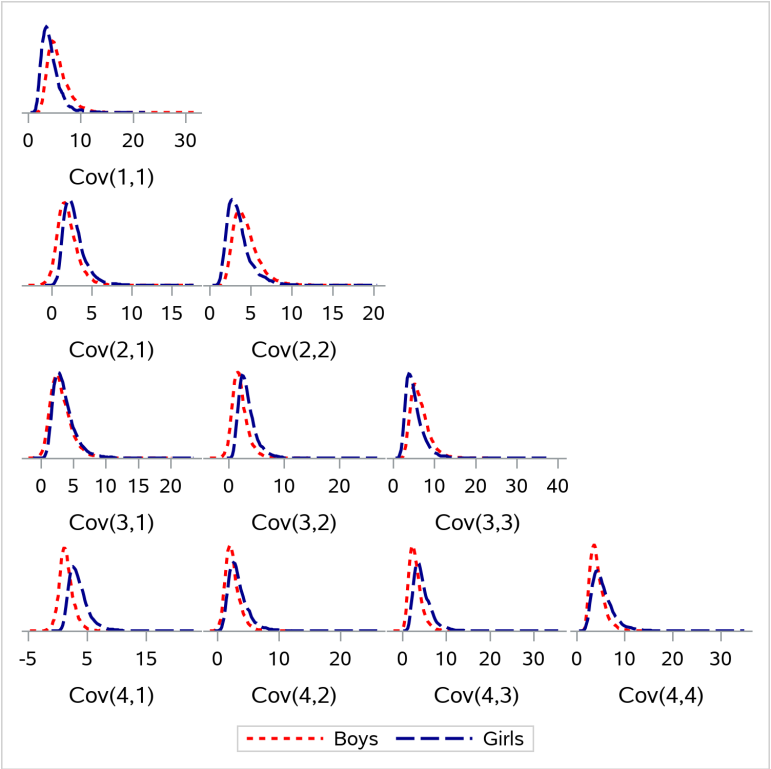

The independent analysis accounts for the heterogeneity in the data (different groups have different covariance structure), but the model cannot represent, for example, the Gender effect, after you separate the data set by Gender. Ideally, you want both to account for the heterogeneity and to fit a model that estimates the Gender effect,

$$\mathbf{y}_i \sim \mathrm{MVN}(\boldsymbol{\mu}_i, \mathbf{R}_k)$$

where $\mathbf{y}_i$ is the vector of observed measurements from the ith subject; $\boldsymbol{\mu}_i$ is the regression mean that has the Gender effect; and $k = \mathrm{F}, \mathrm{M}$, indicating that there are two distinct covariance structures for the two gender groups.

You can fit this model by using the following statements:

```
proc bglimm data=pr seed=475193 outpost=pr_out;
   class Person Gender Time;
   model Distance = Age|Gender;
   repeated Time / type=un subject=Person group=Gender r;
run;
```

The MODEL statement specifies the interaction main effect between Age and Gender, by using a shorthand notation (the vertical bar) for including Age, Gender, and Age*Gender. GROUP=GENDER (in the REPEATED statement) models a separate R-side covariance matrix for each gender. The R option prints the estimated covariance matrix of $\mathbf{R}$.

The results are shown in Output 31.4.5 and Output 31.4.6.

Output 31.4.5 Posterior Summaries and Intervals

The BGLIMM Procedure

Posterior Summaries and Intervals						
Parameter	Group	N	Mean	Standard Deviation	95% HPD Interval	
Intercept		5000	15.9070	1.1413	13.7924	18.2695
Age		5000	0.8254	0.0962	0.6374	1.0135
Gender F		5000	1.4767	1.4106	-1.1653	4.3705
Gender M		0	.	.	.	.
Age*Gender F		5000	-0.3439	0.1222	-0.5893	-0.1010
Age*Gender M		0	.	.	.	.
Residual UN(1,1)	Gender F	5000	4.2321	1.8217	1.6445	7.8020
Residual UN(2,1)	Gender F	5000	2.7878	1.4636	0.7293	5.7478
Residual UN(2,2)	Gender F	5000	3.5504	1.5892	1.3130	6.4988
Residual UN(3,1)	Gender F	5000	3.5348	1.7821	1.0138	6.9532
Residual UN(3,2)	Gender F	5000	3.3304	1.6826	0.8530	6.5813
Residual UN(3,3)	Gender F	5000	5.0936	2.2021	1.8895	9.1471
Residual UN(4,1)	Gender F	5000	3.5407	1.8271	0.9083	7.1369
Residual UN(4,2)	Gender F	5000	3.3792	1.7284	0.8135	6.6893
Residual UN(4,3)	Gender F	5000	4.4877	2.1267	1.3969	8.5377
Residual UN(4,4)	Gender F	5000	5.4479	2.3883	2.0321	9.9821
Residual UN(1,1)	Gender M	5000	5.7815	2.1501	2.5561	10.0795
Residual UN(2,1)	Gender M	5000	1.9706	1.3886	-0.3006	4.9087
Residual UN(2,2)	Gender M	5000	4.4630	1.6476	1.9748	7.6177
Residual UN(3,1)	Gender M	5000	3.1485	1.7035	0.3229	6.5712
Residual UN(3,2)	Gender M	5000	2.0219	1.4383	-0.4459	5.0039
Residual UN(3,3)	Gender M	5000	6.3990	2.2197	2.9795	10.8205
Residual UN(4,1)	Gender M	5000	1.4408	1.3267	-1.0554	4.1671
Residual UN(4,2)	Gender M	5000	2.5001	1.2867	0.4869	5.1913
Residual UN(4,3)	Gender M	5000	2.8916	1.4919	0.4865	6.0024
Residual UN(4,4)	Gender M	5000	4.2609	1.5520	1.9952	7.4605

Output 31.4.6 R-Side Covariance Matrix

Estimated R Matrix					
Group	Row	Col 1	Col 2	Col 3	Col 4
Gender F	1	4.2321	2.7878	3.5348	3.5407
Gender F	2	2.7878	3.5504	3.3304	3.3792
Gender F	3	3.5348	3.3304	5.0936	4.4877
Gender F	4	3.5407	3.3792	4.4877	5.4479
Gender M	1	5.7815	1.9706	3.1485	1.4408
Gender M	2	1.9706	4.4630	2.0219	2.5001
Gender M	3	3.1485	2.0219	6.3990	2.8916
Gender M	4	1.4408	2.5001	2.8916	4.2609

The "Posterior Summaries and Intervals" table in Output 31.4.5 lists posterior estimates for the fixed-effects and residual covariance estimates. The posterior mean estimate of the boys' intercept is 15.907, and that of the girls' intercept is $15.907 + 1.4767 = 17.3837$. Similarly, the estimate for the boys' slope is 0.8254, and

that of the girls' slope is $0.8254 - 0.3439 = 0.4815$. Thus the girls' starting point is greater than the boys' starting point, but the girls' growth rate is about half the boys' growth rate.

You can use ESTIMATE statements as follows to obtain the posterior distribution of these gender-related intercepts and slopes:

```
proc bglimm data=pr seed=475193 outpost=pr_out;
   class Person Gender Time;
   model Distance = Age|Gender;
   repeated Time / type=un subject=Person group=Gender r;
   estimate 'Girls Intercept' Int 1 Gender 1 0;
   estimate 'Boys Intercept'  Int 1 Gender 0 1;
   estimate 'Girls Slope' Age 1 Age*Gender 1 0;
   estimate 'Boys Slope' Age 1 Age*Gender 0 1;
run;
```

Output 31.4.7 lists the posterior summary and interval statistics of the transformed parameters. These values are saved in the OUTPOST= data set, under variable names of Girls_Intercept, Boys_Intercept, Girls_Slope, and Boys_Slope.

Output 31.4.7 Intercepts and Slopes

The BGLIMM Procedure

Results from ESTIMATE Statements				
Label	Mean	Standard Deviation	95% HPD Interval	
Girls Intercept	17.3838	0.8261	15.7077	18.9964
Boys Intercept	15.9070	1.1413	13.7924	18.2695
Girls Slope	0.4814	0.0735	0.3405	0.6304
Boys Slope	0.8254	0.0962	0.6374	1.0135

With these posterior estimates, you can use the following statements to directly infer the probability—for example, that the boys' growth rate is greater than the girls' growth rate. The results are shown in Output 31.4.8.

```
data prob;
   set pr_out;
   pDiff = boys_slope - girls_slope;
   prob = (pDiff > 0);
   keep pDiff prob;
run;

%sumint(data=prob, var=pDiff prob)
```

Output 31.4.8 Probability in Growth Rate Difference

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
pDiff	5000	0.3439	0.1222	0.1010	0.5893
prob	5000	0.9972	0.0528	1.0000	1.0000

The expected slope difference is 0.3439. Output 31.4.9 plots the distribution of the difference in slopes; the shaded area under the positive axis represents the probability that the difference is greater than zero. The area is greater than 99%, and the probability that the boys have a greater growth slope than the girls is over 99%.

Output 31.4.9 Visualizing the Slope Difference

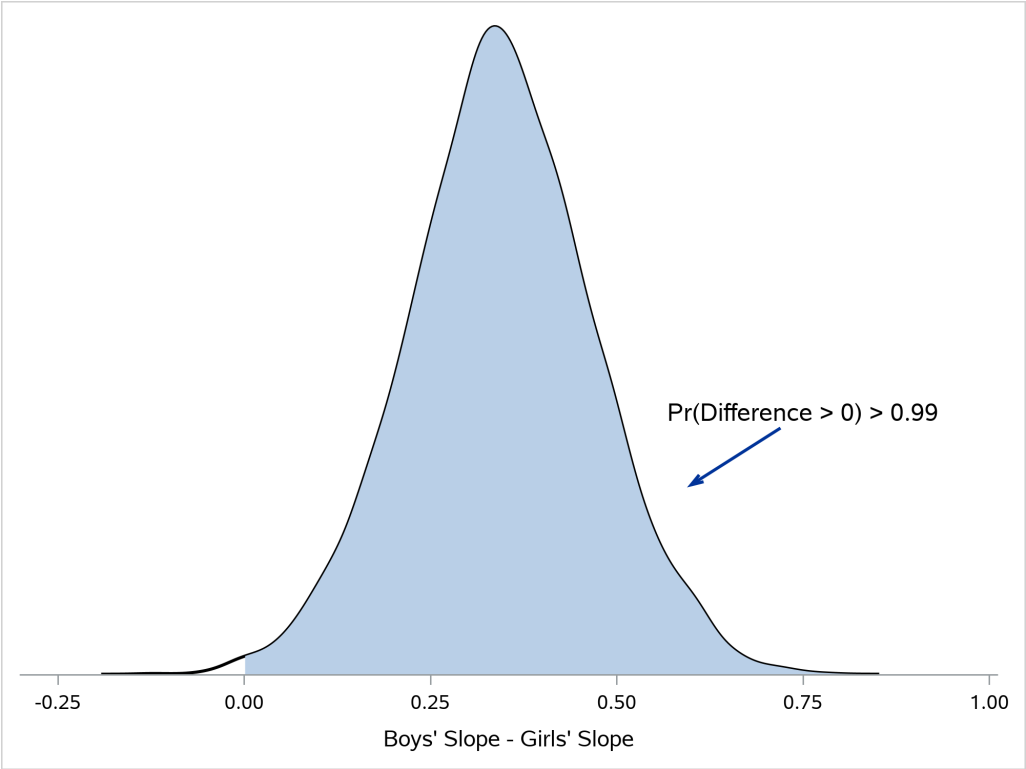

In Output 31.4.5, two of the estimates equal 0; this is a result of the overparameterized model that PROC BGLIMM uses. You can obtain a full-rank parameterization by using the following MODEL statement:

```
model y = Gender Gender*Age / noint;
```

The posterior estimates from the full-rank parameterization (program not shown) are listed in Output 31.4.10. The four estimates are the girls' intercept (17.3857), boys' intercept (15.9519), girls' slope (0.4803), and boys' slope (0.8212). You might find that this parameterization makes it easier to interpret the model.

Output 31.4.10 Posterior Summaries and Intervals from Full-Rank Parameterization

The BGLIMM Procedure

Posterior Summaries and Intervals						
Parameter	Group	N	Mean	Standard Deviation	95% HPD Interval	
Gender F		5000	17.3857	0.8215	15.7553	18.9998
Gender M		5000	15.9519	1.1635	13.6144	18.1640
Age*Gender F		5000	0.4803	0.0731	0.3371	0.6247
Age*Gender M		5000	0.8212	0.0982	0.6247	1.0101
Residual UN(1,1)	Gender F	5000	4.2270	1.8240	1.6174	7.8089
Residual UN(2,1)	Gender F	5000	2.7806	1.4563	0.7068	5.6685
Residual UN(2,2)	Gender F	5000	3.5446	1.5888	1.3287	6.5472
Residual UN(3,1)	Gender F	5000	3.5277	1.7762	1.0206	6.8974
Residual UN(3,2)	Gender F	5000	3.3243	1.6771	0.9705	6.6051
Residual UN(3,3)	Gender F	5000	5.0854	2.1934	1.9174	9.1690
Residual UN(4,1)	Gender F	5000	3.5329	1.8214	0.8502	6.9560
Residual UN(4,2)	Gender F	5000	3.3734	1.7233	0.9427	6.8214
Residual UN(4,3)	Gender F	5000	4.4801	2.1247	1.3600	8.5490
Residual UN(4,4)	Gender F	5000	5.4416	2.3974	2.0595	10.0870
Residual UN(1,1)	Gender M	5000	5.7797	2.1686	2.5756	10.0388
Residual UN(2,1)	Gender M	5000	1.9783	1.3982	-0.4409	4.7993
Residual UN(2,2)	Gender M	5000	4.4669	1.6497	1.9516	7.6345
Residual UN(3,1)	Gender M	5000	3.1495	1.6962	0.4838	6.6577
Residual UN(3,2)	Gender M	5000	2.0172	1.4267	-0.4760	4.9811
Residual UN(3,3)	Gender M	5000	6.3930	2.2058	3.0217	10.8284
Residual UN(4,1)	Gender M	5000	1.4432	1.3330	-0.9448	4.2283
Residual UN(4,2)	Gender M	5000	2.4933	1.2900	0.4163	5.1829
Residual UN(4,3)	Gender M	5000	2.8882	1.4818	0.5280	5.9364
Residual UN(4,4)	Gender M	5000	4.2620	1.5549	1.7624	7.2623

The posterior estimates of the covariance matrices (Output 31.4.6) are close to those from the independent model (Output 31.4.2 and Output 31.4.3), and you are now able to make inferences on Gender-related questions.

The first two ESTIMATE statements in the following program compute the intercept and slope differences in gender, a more intuitive specification than in the overparameterized model. The last two ESTIMATE statements compute the estimated mean distance for girls and boys at age 11, respectively.

```
proc bglimm data=pr seed=475193 outpost=pr_out;
   class Person Gender Time;
   model Distance = Gender Age*Gender / noint;
   repeated Time / type=un subject=Person group=Gender;
   estimate 'Intercept Difference' Gender 1 -1;
   estimate 'Slope Difference' Gender*Age 1 -1;
   estimate 'Girl at Age 11' Gender 1 0 Gender*Age 11 0;
   estimate 'Boy  at Age 11' Gender 0 1 Gender*Age 0 11;
run;
```

The results are shown in Output 31.4.11.

Output 31.4.11 Results from ESTIMATE Statements

The BGLIMM Procedure

Results from ESTIMATE Statements				
Label	Mean	Standard Deviation	95% HPD Interval	
Intercept Difference	1.4338	1.4336	-1.3417	4.2965
Slope Difference	-0.3409	0.1243	-0.5923	-0.1003
Girl at Age 11	22.6688	0.5266	21.6213	23.6831
Boy at Age 11	24.9852	0.4243	24.1859	25.8446

Example 31.5: Multinomial Distribution with Cumulative Links

(View the complete code for this example (bglmmex5.sas) in the example repository.)

This example shows how to analyze multinomial data in which the response variable has more than two categories and is ordinal. The BGLIMM procedure fits two kinds of link function for multinomial data: one with a cumulative link that applies to ordinal data, and the other with a generalized logit that applies to nominal data. For more information, see the section "Multinomial Models" on page 1350.

The data, which are shown in the following DATA step, are collected from a study (Vonesh 2012) that compares active treatment to placebo for patients who had shoulder pain after rotator-cuff surgery. The two treatments were randomly assigned to patients. The patients were asked how serious their pain was in both the morning and the afternoon for three days (hence six follow-up data points) after surgery. The response variable that is used to rate the pain score is an ordinal measure containing five categories, from 1 to 5 (low to severe pain) (Kiernan 2018).

```
data Shoulder_pain;
   input Trt$ Gender$ Age T1 T2 T3 T4 T5 T6 @@;
   ID = _n_;
   datalines;
y  f  64  1  1  1  1  1  1
y  m  41  3  2  1  1  1  1
y  f  77  3  2  2  2  1  1
y  f  54  1  1  1  1  1  1
y  f  66  1  1  1  1  1  1
y  m  56  1  2  1  1  1  1

   ... more lines ...

n  f  41  5  5  5  4  3  3
n  m  72  3  3  3  3  1  1
n  f  60  5  4  4  4  2  2
n  m  61  1  3  3  3  2  1
;
```

To analyze these multinomial data with PROC BGLIMM, you need to rearrange the data in univariate format. The following DATA step creates several observation lines from each record in the original data set, so each patient has six data lines, one for each of the six follow-up sessions.

```
data ShoulderData;
   set Shoulder_pain;
   array tt T1-T6;
   do over tt;
      Y    = tt;
      Time = _i_;
      output;
   end;
run;
```

You can add formats for treatment (Trt) and gender as follows to make the data more readable:

```
proc format;
   value $abc 'y' = 'Active'
              'n' = 'Placebo';
   value $xyz 'f' = 'Female'
              'm' = 'Male';
run;

proc print data=ShoulderData(obs=18);
   var ID Trt Gender Age Time y;
   format Trt $abc. Gender $xyz.;
run;
```

The resulting data are shown in Table 31.5.1 for the first three patients, who have ID=1, 2, and 3.

Output 31.5.1 First 18 Observations of the Shoulder_pain Data Set

Obs	ID	Trt	Gender	Age	Time	Y
1	1	Active	Female	64	1	1
2	1	Active	Female	64	2	1
3	1	Active	Female	64	3	1
4	1	Active	Female	64	4	1
5	1	Active	Female	64	5	1
6	1	Active	Female	64	6	1
7	2	Active	Male	41	1	3
8	2	Active	Male	41	2	2
9	2	Active	Male	41	3	1
10	2	Active	Male	41	4	1
11	2	Active	Male	41	5	1
12	2	Active	Male	41	6	1
13	3	Active	Female	77	1	3
14	3	Active	Female	77	2	2
15	3	Active	Female	77	3	2
16	3	Active	Female	77	4	2
17	3	Active	Female	77	5	1
18	3	Active	Female	77	6	1

Because the response categories are ordered, a proportional-odds model is chosen. The initial model is a proportional-odds model with the cumulative logit link (McCullagh 1980) and $J = 5$ categories. Separate intercepts (cutoffs) are modeled for the first $J - 1 = 4$ cumulative categories, and the intercepts are

monotonically increasing, which guarantees ordering of the cumulative probabilities and nonnegative category probabilities. The probability of observing a rating in at most category $j \leq 4$ is

$$\Pr(Y \leq j) = \frac{1}{1 + \exp(-\eta_j)}$$
$$\eta_j = \alpha_j + \beta_1 \text{Trt} + \beta_2 \text{Gender} + \beta_3 \text{Age} + \beta_4 \text{Time}$$

where α_j is the intercept for the jth response category and $\alpha_1 \leq \alpha_2 \leq \alpha_3 \leq \alpha_4$.

The following PROC BGLIMM statements fit a model that uses the cumulative logit. Later other links, such as the cumulative probit, are considered.

```
proc bglimm data=ShoulderData seed=8875 nmc=10000 thin=2 dic;
   class ID Trt Gender;
   model y  = Trt Gender Age Time / dist=multinomial link=clogit;
   format Trt $abc. Gender $xyz. ;
run;
```

The "Model Information" table, shown in Output 31.5.2, displays basic information about the data and model. In this case, the distribution is multinomial with ordered response categories, and the link function is cumulative logit.

Output 31.5.2 Model Information

The BGLIMM Procedure

Model Information	
Data Set	WORK.SHOULDERDATA
Response Variable	Y
Distribution	Multinomial (ordered)
Link Function	Cumulative Logit
Fixed Effects Included	Yes
Random Effects Included	No
Sampling Algorithm	Gamerman
Burn-In Size	500
Simulation Size	10000
Thinning	2
Random Number Seed	8875
Number of Threads	1

The "Response Profile" table (Output 31.5.3) lists all levels of the response variable, the ordering of the response variable, and a breakdown of the frequencies by category. The probabilities that are modeled are accumulated over the lower ordered values. You should review the "Response Profile" table to ensure that the categories are properly arranged and the desired outcome levels are modeled. In this table, response levels are arranged by ordered value. The lowest response level is assigned ordered value 1, the next-lowest response level is assigned ordered value 2, and so on. For a categorical response, the probability that is modeled is the probability of the response level that has the lower ordered value.

Output 31.5.3 Response Profile

Response Profile		
Ordered Value	Y	Total Frequency
1	1	131
2	2	40
3	3	35
4	4	28
5	5	12

Probabilities modeled are accumulated over the lower ordered values.

For multinomial models, the response-level ordering is important. With a cumulative link, the categories are assumed to be ordered by their ordered value in the "Response Profile" table. If the response variable is a character variable or has a format, check this table carefully to determine whether the ordered values reflect the correct ordinal scale.

The parameter estimates are shown in Output 31.5.4. The probabilities that are modeled here are accumulated at the lower end of the pain scale. There are four intercept terms: Intercept 1 defines the boundary between pain levels 1 and 2, Intercept 2 defines the boundary between pain levels 2 and 3, and so on. The intercept terms are in increasing order, and they correspond to the four cumulative logits that are defined for the response levels in the order shown in the "Response Profile" table.

Output 31.5.4 Posterior Summaries and Intervals

Posterior Summaries and Intervals						
Parameter	Y	N	Mean	Standard Deviation	95% HPD Interval	
Intercept 1	**1**	5000	-3.2178	0.6412	-4.3631	-1.8889
Intercept 2	**2**	5000	-2.2858	0.6158	-3.4348	-1.0306
Intercept 3	**3**	5000	-1.2499	0.6009	-2.3614	0.00273
Intercept 4	**4**	5000	0.2732	0.6365	-0.9863	1.5235
Trt Active	**.**	5000	1.9550	0.2801	1.4226	2.5186
Trt Placebo	**.**	0	.	.	.	.
Gender Female	**.**	5000	0.1213	0.2728	-0.4057	0.6705
Gender Male	**.**	0	.	.	.	.
Age	**.**	5000	0.0271	0.00856	0.0106	0.0441
Time	**.**	5000	0.2026	0.0771	0.0627	0.3625

The treatment (Trt) effect shows how far the boundaries move up or down under the two treatment groups, active treatment and placebo. In this case, the active treatment (labeled "Trt Active") is positive in relation to the placebo group (labeled "Trt Placebo"), which is the reference group and whose value is fixed at 0. Therefore, the active treatment group has a higher probability of a low pain score and a lower probability of severe pain than the placebo group. The fact that the Time effect estimate is positive indicates a reduction in pain over time. The results support an expected pain-decreasing process after surgery. The treatment and time effects play a role in the pain rating because their 95% HPD intervals do not contain the value 0, whereas age does not seem to affect the pain scoring because age's 95% HPD interval has 0 in it.

You can also use other link functions. For the multinomial distribution, these links are available: CUMLOGIT, CUMPROBIT, CUMLOGLOG, CUMCLL, and GLOGIT. In the following statements, the LINK=CPROBIT

option specifies that the cumulative probit link be used:

```
proc bglimm data=ShoulderData seed=8875 nmc=10000 thin=2;
   class ID Trt Gender;
   model y  = Trt Gender Age Time / dist=multinomial link=cprobit;
   format Trt $abc. Gender $xyz. ;
run;
```

Because the patients were selected at random, you can include patient-specific random effects in the model as follows:

$$\begin{aligned}\Pr(Y \le j) &= \frac{1}{1+\exp\{-\eta_j\}}\\ \eta_j &= \alpha_j + \beta_1\text{Trt} + \beta_2\text{Gender} + \beta_3\text{Age} + \beta_4\text{Time} + \gamma_i\\ \gamma_i &\sim \text{iid}\, N(0, \sigma_\gamma^2),\end{aligned}$$

where γ_i denotes the intercept for the ith patient and σ_γ^2 is the variance of individual random intercepts that measures the variability among all patients. Note that the random effects do not depend on the response category.

The following statements fit a model that has both the fixed covariates and the random patient-specific intercepts and produce the results in Output 31.5.5:

```
proc bglimm data=ShoulderData seed=8875 nmc=10000 thin=2 dic;
   class ID Trt Gender;
   model y  = Trt Gender Age Time / dist=multinomial link=clogit;
   random int / sub=ID s=(1 to 3);
   format Trt $abc. Gender $xyz. ;
run;
```

In the “Random Var” row of the “Posterior Summaries and Intervals” table in Output 31.5.5, you can see that the posterior mean of the variance of the random intercepts is about 5.6 and the 95% HPD interval does not include 0, implying that there is heterogeneity among subjects that needs to be accounted for.

Output 31.5.5 Posterior Summaries and Intervals

The BGLIMM Procedure

Posterior Summaries and Intervals							
Parameter	Subject	Y	N	Mean	Standard Deviation	95% HPD Interval	
Intercept 1		1	5000	-5.7518	1.7866	-9.3910	-2.3964
Intercept 2		2	5000	-4.1819	1.7548	-7.5569	-0.7251
Intercept 3		3	5000	-2.5194	1.7275	-6.0605	0.7211
Intercept 4		4	5000	-0.3470	1.7239	-3.6830	3.0815
Trt Active		.	5000	2.9931	0.8525	1.3478	4.6608
Trt Placebo		.	0	.	.	.	.
Gender Female		.	5000	0.2491	0.8678	-1.4435	1.9503
Gender Male		.	0	.	.	.	.
Age		.	5000	0.0515	0.0277	-0.00198	0.1086
Time		.	5000	0.3904	0.0936	0.1982	0.5649
Random Var		.	5000	5.5546	2.0659	2.4095	9.8033
Intercept	ID 1	.	5000	1.7409	1.6941	-1.3510	5.1678
Intercept	ID 2	.	5000	0.0638	1.1541	-2.3022	2.3000
Intercept	ID 3	.	5000	-2.8045	1.1375	-5.0011	-0.6100

In the following code, the ESTIMATE statement is added in order to compare the two treatment groups: active and placebo. The results are shown in Output 31.5.6. The EXP option produces its exponentiated version, which is the odds ratio in the case of using the logit link.

```
proc bglimm data=ShoulderData seed=8875 nmc=10000 thin=2 dic;
   class ID Trt Gender;
   model y  = Trt Gender Age Time / dist=multinomial link=clogit;
   random int / sub=ID s=(1 to 3);
   format Trt $abc. Gender $xyz. ;
   estimate 'Active vs Placebo' Trt 1 -1  / exp;
run;
```

The large value of the odds ratio indicates that the odds that the active treatment group is in lower pain categories is approximately 32 times the odds that the placebo group is in lower pain categories. Simply put, the odds ratio indicates approximately a 32-fold reduction in pain while controlling heterogeneity among subjects.

Output 31.5.6 Results from ESTIMATE Statement

The BGLIMM Procedure

Results from ESTIMATE Statements					
Label	Mean	Standard Deviation	95% HPD Interval		Exponentiated Mean
Active vs Placebo	2.9931	0.8525	1.3478	4.6608	32.0356

Example 31.6: Multinomial Generalized Logit Model for Nominal Response

(View the complete code for this example (bglmmex6.sas) in the example repository.)

This example uses data from a study in which educational researchers compared three different styles of mathematics instruction for third graders (Stokes, Davis, and Koch 2012). Students were rotated through three learning styles: a self-instruction mode largely based on computer use, a team approach in which students solved problems in groups, and a traditional class approach that is used at a regular school. The students were asked which style they preferred, and their responses were classified by the type of program that they were placed in (a regular school day versus a regular day supplemented with an after-school program in the afternoon). Researchers were interested in how other school programs influenced the effectiveness of the styles and how they affected the students' perceptions of the different styles.

Table 31.18 displays data that reflect the students' preferences of styles, cross-classified by their school and the program that they were placed in.

Table 31.18 Learning Style Preference Data

		Learning Style Preference		
School	**Program**	**Self**	**Team**	**Class**
1	Regular	10	17	26
1	Afternoon	5	12	50
2	Regular	21	17	26
2	Afternoon	16	12	36
3	Regular	15	15	16
3	Afternoon	12	12	20

The levels (self, team, and class) of the response variable Style have no essential ordering. Instead of fitting a model that uses cumulative logits, a model that uses generalized logits is more appropriate for nominal responses. There are three response levels, so you could form logits that compare self to class, or team to class, treating class as the reference level, because that is the traditional way of learning.

The following statements create the data set school and request the analysis:

```
data school;
   length Program $ 9;
   input School Program $ Style $ Count @@;
   datalines;
1 regular   self 10  1 regular   team 17  1 regular   class 26
1 afternoon self  5  1 afternoon team 12  1 afternoon class 50
2 regular   self 21  2 regular   team 17  2 regular   class 26
2 afternoon self 16  2 afternoon team 12  2 afternoon class 36
3 regular   self 15  3 regular   team 15  3 regular   class 16
3 afternoon self 12  3 afternoon team 12  3 afternoon class 20
;
```

A multinomial logistic regression is performed on the generalized logits. In the following statements, the LINK=GLOGIT option forms the generalized logits. You can choose a reference category by specifying the REF= option. The choice of reference category for generalized logit models affects the results. An interaction between School and Program is added to the two main effects. Because the data are in frequency/count

form, you need to indicate this to PROC BGLIMM: you do this by specifying the FREQ statement. If all frequencies are 1, you can omit the FREQ statement.

```
proc bglimm data=school seed=123 dic;
   freq Count;
   class School Program;
   model Style(ref="class")=School Program School*Program / dist=multinomial
         link=glogit;
run;
```

The "Response Profile" table (Output 31.6.1) contains the response information. In generalized logit models for unordered response, one category is chosen as the reference in the formulation of the generalized logits. By default, the linear predictor in the reference category is set to 0, and the reference category corresponds to the entry in the "Response Profile" table for the first ordered value. You can change the assignment of ordered values by specifying the DESCENDING and ORDER= options in the MODEL statement. You can choose a different reference category by specifying the REF= option. The choice of reference category for generalized logit models affects the results. It is sometimes recommended that you choose the category that has the highest frequency as the reference (see, for example, Brown and Prescott 1999, p. 160). Because the REF="CLASS" option is specified, the generalized logits are formed for the probability of "self" with respect to "class", and for the probability of "team" with respect to "class".

Output 31.6.1 Response Profile

The BGLIMM Procedure

Response Profile		
Ordered Value	Style	Total Frequency
1	class	174
2	self	79
3	team	85

Logits modeled use 'class' as the reference category.

The results are shown in Output 31.6.2. Essentially, the parameter space has the same structure that it would for modeling a single response function, except that it models two response functions: the first logit of "self" with respect to "class" (labeled "self" in the response Style column), and the second logit of "team" with respect to "class" (labeled "team" in the response Style column).

Output 31.6.2 Posterior Summaries and Intervals

Posterior Summaries and Intervals						
Parameter	Style	N	Mean	Standard Deviation	95% HPD Interval	
Intercept 1	self	5000	-0.0597	0.3621	-0.7973	0.6100
School 1 1	self	5000	-0.9194	0.5067	-2.0099	0.0362
School 2 1	self	5000	-0.1527	0.4731	-1.0109	0.8400
School 3 1	self	0	.	.	.	.
Program afternoon 1	self	5000	-0.4807	0.5253	-1.5254	0.4697
Program regular 1	self	0	.	.	.	.
School 1*Program afternoon 1	self	5000	-0.9416	0.8158	-2.6314	0.5057
School 1*Program regular 1	self	0	.	.	.	.
School 2*Program afternoon 1	self	5000	-0.1267	0.6779	-1.5019	1.0586
School 2*Program regular 1	self	0	.	.	.	.
School 3*Program afternoon 1	self	0	.	.	.	.
School 3*Program regular 1	self	0	.	.	.	.
Intercept 2	team	5000	-0.0635	0.3586	-0.7852	0.6055
School 1 2	team	5000	-0.3704	0.4695	-1.2872	0.5298
School 2 2	team	5000	-0.3760	0.4793	-1.3432	0.4804
School 3 2	team	0	.	.	.	.
Program afternoon 2	team	5000	-0.4812	0.5342	-1.6885	0.4042
Program regular 2	team	0	.	.	.	.
School 1*Program afternoon 2	team	5000	-0.5497	0.6954	-1.8526	0.8048
School 1*Program regular 2	team	0	.	.	.	.
School 2*Program afternoon 2	team	5000	-0.1973	0.7271	-1.6619	1.2083
School 2*Program regular 2	team	0	.	.	.	.
School 3*Program afternoon 2	team	0	.	.	.	.
School 3*Program regular 2	team	0	.	.	.	.

Because the interaction terms include 0 in the 95% HPD intervals, it is suspected that the interaction of "School*Program" does not affect the model. As shown in Output 31.6.3 and Output 31.6.4, the DIC for the model that contains the interaction is greater than the DIC for the main-effects model without an interaction. Hence the interaction term might be unnecessary.

Output 31.6.3 DIC for the Main-plus-Interaction Model

Deviance Information Criterion	
Dbar (Posterior Mean of Deviance)	677.372
Dmean (Deviance Evaluated at Posterior Mean)	665.230
pD (Effective Number of Parameters)	12.143
DIC (Smaller is Better)	689.515

Output 31.6.4 DIC for the Main-Effects Model

Deviance Information Criterion	
Dbar (Posterior Mean of Deviance)	674.981
Dmean (Deviance Evaluated at Posterior Mean)	666.965
pD (Effective Number of Parameters)	8.016
DIC (Smaller is Better)	682.996

The following statements fit the main-effects model. The first two ESTIMATE statements compare differences among the three schools, and the last ESTIMATE statement computes the difference between the two programs. The EXP option produces the odds ratio for each effect comparison. The BYCAT option reports estimates for each category of the response variable.

```
proc bglimm data=school seed=123 dic;
   freq Count;
   class School Program;
   model Style(ref="class")= School Program / dist=multinomial
         link=glogit;
   estimate 'School 1 vs 3' School 1 0 -1  / exp bycat;
   estimate 'School 2 vs 3' School 0 1 -1  / exp bycat;
   estimate 'Afternoon vs Regular' Program 1 -1  / exp bycat;
run;
```

The parameter posterior summaries and intervals of the main-effects model are shown in Output 31.6.5. The first section of parameters is for the first logit, which is of "self" versus "class", and the second section of parameters is for the second logit, which is of "team" versus "class". For the School effect, "School 1 1" is the effect of specifying a value of 1 for School for the first logit, "School 2 1" is the effect of specifying a value of 2 for School for the first logit, "School 1 2" is the effect of specifying a value of 1 for School for the second logit, and "School 2 2" is the effect of specifying a value of 2 for School for the second logit. For the Program effect, "Program afternoon 1" is the effect of specifying a value of "afternoon" for Program for the first logit, and "Program afternoon 2" is the effect of specifying a value of "afternoon" for Program for the second logit.

Output 31.6.5 Posterior Summaries and Intervals

The BGLIMM Procedure

Posterior Summaries and Intervals

Parameter	Style	N	Mean	Standard Deviation	95% HPD Interval	
Intercept 1	self	5000	0.0942	0.2878	-0.5241	0.6155
School 1 1	self	5000	-1.3376	0.3823	-2.0287	-0.5633
School 2 1	self	5000	-0.2390	0.3283	-0.8340	0.4287
School 3 1	self	0	.	.	.	.
Program afternoon 1	self	5000	-0.7696	0.2866	-1.3255	-0.2117
Program regular 1	self	0	.	.	.	.
Intercept 2	team	5000	0.0977	0.2852	-0.4775	0.6350
School 1 2	team	5000	-0.6725	0.3428	-1.3106	0.00307
School 2 2	team	5000	-0.4945	0.3400	-1.1672	0.1437
School 3 2	team	0	.	.	.	.
Program afternoon 2	team	5000	-0.7564	0.2673	-1.3210	-0.2522
Program regular 2	team	0	.	.	.	.

Table 31.19 displays the parameter estimates, which are arranged according to the logits that they reference. This is an effective way to show the results from an analysis of a multinomial generalized logits model.

Table 31.19 Parameter Estimates from the Main-Effects Model

	logit(self/class)		logit(team/class)	
Parameter	**Mean**	**StdDev**	**Mean**	**StdDev**
Intercept	0.09	0.29	0.10	0.29
School 1	-1.34	0.38	-0.67	0.34
School 2	-0.24	0.33	-0.49	0.34
Program afternoon	-0.77	0.29	-0.76	0.27

School with the value of 1 has the largest effect of the schools, particularly for the logit that compares "self" to "class". The Program covariate has nearly the same effect on both logits.

The results from the three ESTIMATE statements are shown in Output 31.6.6, which displays the differences among the three schools and the two programs. The Exponentiated Mean column shows the odds ratio for each effect comparison. Because the BYCAT option is specified, the effect comparison is made for each category of the response variable Style.

Output 31.6.6 Results from ESTIMATE Statements

Results from ESTIMATE Statements

Label	Style	Mean	Standard Deviation	95% HPD Interval		Exponentiated Mean
School 1 vs 3	**self**	-1.3376	0.3823	-2.0287	-0.5633	0.2823
School 1 vs 3	**team**	-0.6725	0.3428	-1.3106	0.00307	0.5415
School 2 vs 3	**self**	-0.2390	0.3283	-0.8340	0.4287	0.8306
School 2 vs 3	**team**	-0.4945	0.3400	-1.1672	0.1437	0.6464
Afternoon vs Regular	**self**	-0.7696	0.2866	-1.3255	-0.2117	0.4826
Afternoon vs Regular	**team**	-0.7564	0.2673	-1.3210	-0.2522	0.4863

Example 31.7: Bayesian Networking Meta-analysis

(View the complete code for this example (bglmmex7.sas) in the example repository.)

Networking meta-analysis (NMA) synthesizes direct and indirect evidence on multiple treatments from a collection of independent studies or randomized controlled trials (Lee 2014). There are two ways to set up the model: using the contrast-based NMA method (Dias and Ades 2016) and using the arm-based NMA method (Lin et al. 2017). In the popular arm-based NMA method, population-averaged treatment-specific parameters, such as absolute risks, are modeled and estimated. This example demonstrates how to use PROC BGLIMM to perform arm-based NMA for various types of responses.

Arm-Based NMA for Binomial Outcomes

Suppose that there are I studies on K treatments and that each study has a subset of the K treatments. In the ith study, the numbers of events and trials in treatment group k are denoted by y_{ik} and n_{ik}. The response y_{ik} is assumed to have a binomial distribution:

$$\begin{aligned} y_{ik} &\sim \text{Binomial}(n_{ik}, p_{ik}) \\ \Phi^{-1}(p_{ik}) &= \mu_k + \gamma_{ik} \\ (\gamma_{i1}, \gamma_{i2}, \ldots, \gamma_{iK})^T &\sim \text{MVN}(\mathbf{0}, \mathbf{G}) \end{aligned}$$

where μ_k represents the fixed main effect for treatment k; $\gamma_i = (\gamma_{i1}, \gamma_{i2}, \ldots, \gamma_{iK})^T$ are the random effects that follow a multivariate normal distribution (MVN) with a covariance matrix $\mathbf{G}$ and Φ is the standard normal cumulative density function that connects the event probability p_{ik} and the linear predictor $\mu_k + \gamma_{ik}$. You can also consider other link functions, but they do not have simple expressions for population-averaged absolute risks.

The absolute risk (event probability) of treatment k can be estimated as

$$p_k = \Phi(\mu_k / \sqrt{1 + G_k})$$

where G_k is the kth diagonal element in $\mathbf{G}$. Because this is a marginal expectation of p_{ik} given μ_k and G_k, you can interpret p_k as the population-averaged absolute risk of treatment k. With the estimates p_k for absolute risk, it is often of interest to estimate the risk difference (RD), odds ratio (OR), and risk ratio (RR), which are defined as follows:

$$\begin{aligned} \text{RD}_{kl} &= p_k - p_l \\ \text{OR}_{kl} &= \frac{p_k/(1-p_k)}{p_l/(1-p_l)} \\ \text{RR}_{kl} &= p_k/p_l \end{aligned}$$

Here is an example. The data in the following DATA step contain 24 studies on smoking cessation (Lin et al. 2017). There are four treatments (Treat): no contact (NC), self-help (SH), individual counseling (IC), and group counseling (GC). Event and Total are the number of events (successful smoking cessation) and the total number of participants, respectively.

```
data SmokeData;
   input Study Treat $ Event Total;
   datalines;
 1     NC      9       140
 1     IC     23       140
 1     GC     10       138
 2     SH     11        78
 2     IC     12        85
 2     GC     29       170
 3     NC     79       702

   ... more lines ...

22     GC     32       127
```

```
23     IC     12        76
23     GC     20        74
24     IC      9        55
24     GC      3        26
;
```

The following statements fit the model by using arm-based NMA for binomial responses:

```
proc bglimm data=SmokeData seed=1315 nbi=5000 nmc=100000 thin=10
            outpost=CSout;
   class Study Treat / order=data;
   model Event/Total = Treat / link=probit noint;
   random Treat / sub=Study type=cs g monitor=(1 to 2);
run;
```

The response is the proportion of events as binomial ratios, so the *events/trials* syntax is used, which in this example is Event/Total. The fixed effect Treat measures the main effect of each treatment, and the random effect Treat (clustered at Study) measures the deviation of each treatment's effect in a study from the main effect—that is, the overall mean of all studies. Note that the probit link is used to relate the linear predictor to the event probability. PROC BGLIMM includes many choices of link function, such as the logit and complementary log-log. The probit link (LINK=PROBIT) is used here both for its easy comparison with the results from other software packages that allow only the probit link (Lin et al. 2017) and for its closed-form solution of the absolute risk estimates. The covariance matrix of the random effects is compound symmetry by the TYPE=CS specification, because compound symmetry is one of the structures that are seen most often in this kind of meta-analysis. The G option prints the estimated covariance matrix of the random effects. The MONITOR=(1 TO 2) option prints the estimates of the random effects for the first two studies.

Output 31.7.1 displays the "Posterior Summaries and Intervals" table, which lists the summary statistics (posterior means, standard deviations, and HPD intervals) for the model parameters.

Output 31.7.1 Posterior Summaries and Intervals for the Smoking Cessation Data

The BGLIMM Procedure

Posterior Summaries and Intervals						
Parameter	Subject	N	Mean	Standard Deviation	95% HPD Interval	
Treat NC		10000	-1.5176	0.1226	-1.7591	-1.2791
Treat IC		10000	-1.2293	0.1133	-1.4577	-1.0069
Treat GC		10000	-0.8543	0.2034	-1.2416	-0.4508
Treat SH		10000	-1.1256	0.2012	-1.5248	-0.7407
Random Var		10000	0.2195	0.0776	0.0938	0.3769
Random CS		10000	0.00251	0.0541	-0.0896	0.1125
Treat NC	Study 1	10000	-0.00819	0.1915	-0.3673	0.3798
Treat IC	Study 1	10000	0.2318	0.1595	-0.0773	0.5498
Treat GC	Study 1	10000	-0.5428	0.2349	-1.0060	-0.0921
Treat SH	Study 1	10000	0.0267	0.4509	-0.8366	0.9311
Treat NC	Study 2	10000	0.00259	0.4413	-0.8924	0.8451
Treat IC	Study 2	10000	0.1286	0.1860	-0.2482	0.4802
Treat GC	Study 2	10000	-0.0951	0.2211	-0.5425	0.3265
Treat SH	Study 2	10000	0.0365	0.2410	-0.4236	0.5243

The fixed effects, labeled "Treat NC", "Treat IC", "Treat GC", and "Treat SH", measure the main effects of

the four treatments. "Treat NC" has the smallest value, indicating that the no-contact group has the lowest probability of quitting smoking in general; "Treat GC" has the largest value, indicating that group counseling is the best treatment for smoking cessation.

The next two parameters in Output 31.7.1, labeled "Random Var" and "Random CS", are the two parameters in the following covariance matrix of random effects. A compound-symmetry covariance matrix has constant variance and constant covariance.

$$\begin{bmatrix} \sigma_1 + \sigma^2 & \sigma_1 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 + \sigma^2 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma_1 + \sigma^2 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma_1 & \sigma_1 + \sigma^2 \end{bmatrix}$$

The parameters "Random Var" and "Random CS" correspond to σ^2 and σ_1, respectively, in the above matrix. Therefore, the kth diagonal term in **G** is the sum of "Random Var" and "Random CS".

You can calculate the absolute risk / event probability by using the equation

$$p_k = \Phi(\mu_k / \sqrt{1 + G_k})$$

where G_k is the variance of the kth treatment's random effect (that is, the kth diagonal element of the covariance matrix **G**). You can use a SAS DATA step to obtain the posterior samples of treatment-specific absolute event probabilities as follows, by using the posterior samples of the model parameters that are saved in the OUTPOST=CSOUT data set:

```
data CSoutP;
   set CSout;
   p_NC = probnorm(Treat_NC/sqrt(1+Random_Var+Random_CS));
   p_IC = probnorm(Treat_IC/sqrt(1+Random_Var+Random_CS));
   p_GC = probnorm(Treat_GC/sqrt(1+Random_Var+Random_CS));
   p_SH = probnorm(Treat_SH/sqrt(1+Random_Var+Random_CS));
run;
```

Output 31.7.2 plots the estimated posterior density of the absolute event probabilities (p_NC, p_SH, p_IC, p_GC). The no-contact group clearly has the lowest event probability, and its posterior density overlaps with the posterior densities of the other three groups only in a very small area. Group counseling offers the greatest chance to quit smoking; hence it is the best intervention in the study.

Output 31.7.2 Posterior Distributions of Absolute Event Probabilities

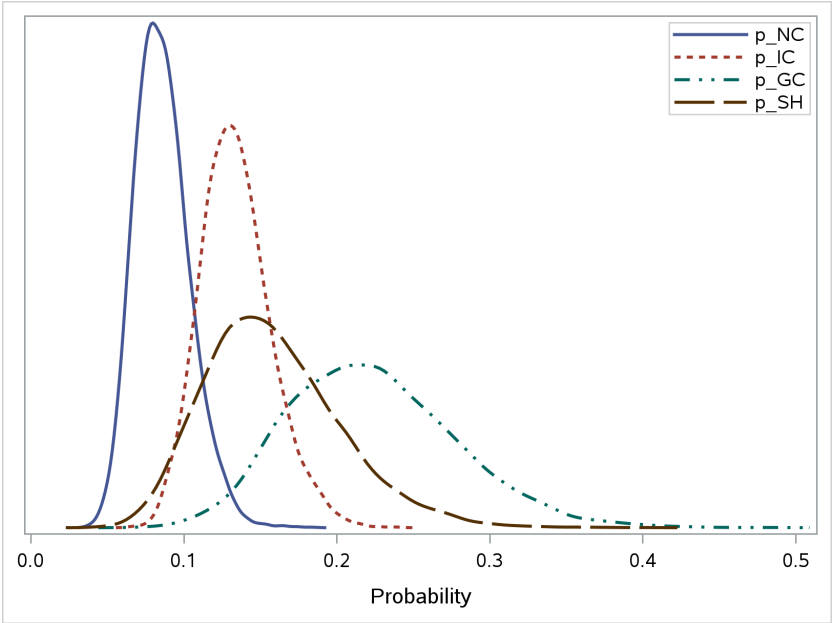

Arm-Based NMA for Continuous Outcomes

You can also perform arm-base NMA in studies that have continuous outcomes. In addition to the same notation with I studies on K treatments, the total number of participants in treatment group k is denoted by n_{ik}, sample mean $\bar{y}_{ik}$, and sample variance s_{ik}^2:

$$\begin{aligned} \bar{y}_{ik} &\sim \text{Normal}(\mu_{ik}, s_{ik}^2/n_{ik}) \\ \mu_{ik} &= \beta_k + \gamma_{ik} \\ (\gamma_{i1}, \gamma_{i2}, \ldots, \gamma_{iK})^T &\sim \text{MVN}(\mathbf{0}, \mathbf{G}) \end{aligned}$$

In this setup, β_k is of interest because it is the overall effect of treatment k, $E[\mu_{ik}|\beta_k, \sigma_k] = \beta_k$, and you can estimate the effect difference between treatments k and l, which is defined as $D_{kl} = \beta_k - \beta_l$.

In this example, the data contain seven studies that have a continuous outcome, which is the mean off-time reduction in patients who are given dopamine agonists as adjunct therapy for Parkinson's disease (Lin et al. 2017). There are five treatments (Trt): placebo (coded 1) is listed first and the other four treatments are active drugs (coded 2 to 5). The data set is as follows:

```
data Parkinson;
   input Sid Trt Mean SD Nstudy;
   datalines;
1   1   -1.22   3.70    54
1   3   -1.53   4.28    95
2   1   -0.70   3.70   172
2   2   -2.40   3.40   173
3   1   -0.30   4.40    76
3   2   -2.60   4.30    71
3   4   -1.20   4.30    81
```

```
4   3   -0.24   3.00   128
4   4   -0.59   3.00    72
5   3   -0.73   3.00    80
5   4   -0.18   3.00    46
6   4   -2.20   2.31   137
6   5   -2.50   2.18   131
7   4   -1.80   2.48   154
7   5   -2.10   2.99   143
;
```

In the data, Mean and SD represent the sample mean and sample standard deviation, respectively, of the continuous outcome for each treatment in the study.

To fit the model shown in the preceding equations, you use a DATA step as follows in order to obtain the sample variance for each treatment in the study:

```
data Parkinson2;
   set Parkinson;
   SVar= SD*SD/Nstudy;
   run;
```

Then you can use the SCALE= option as follows to specify the variance variable so that PROC BGLIMM reads in that variable and holds the variance of each response at a fixed value:

```
proc bglimm data=Parkinson2 seed=1315 nmc=50000 thin=10 outpost=PostSamples;
   class Trt Sid / order=internal;
   model mean = Trt / noint scale=SVar cprior=normal;
   random Trt / sub=Sid type=cs monitor=(1 to 2);
run;
```

The results are shown in Output 31.7.3 along with the posterior summary statistics for the model parameters. The variance parameter is not displayed in this table, because it is fixed at a known value for each treatment in the study.

Output 31.7.3 Posterior Summaries and Intervals for Parkinson's Data

The BGLIMM Procedure

Posterior Summaries and Intervals						
Parameter	Subject	N	Mean	Standard Deviation	95% HPD Interval	
Trt 1		5000	-0.7224	0.5926	-1.7988	0.5667
Trt 2		5000	-2.5525	0.6816	-3.9816	-1.2508
Trt 3		5000	-1.0800	0.5894	-2.3027	0.0715
Trt 4		5000	-1.3227	0.5020	-2.3581	-0.3296
Trt 5		5000	-1.8498	0.6726	-3.2085	-0.4942
Random Var		5000	0.3266	0.5261	0.000198	1.2168
Random CS		5000	1.1006	1.9663	-0.2783	3.7573
Trt 1	Sid 1	5000	-0.3863	0.6130	-1.6470	0.8055
Trt 2	Sid 1	5000	-0.2869	0.8083	-1.9034	1.3898
Trt 3	Sid 1	5000	-0.3883	0.6099	-1.5535	0.9171
Trt 4	Sid 1	5000	-0.2811	0.8171	-1.9340	1.4125
Trt 5	Sid 1	5000	-0.2900	0.7904	-1.8455	1.3164
Trt 1	Sid 2	5000	0.0496	0.6013	-1.1535	1.2659
Trt 2	Sid 2	5000	0.1269	0.6702	-1.3069	1.3822
Trt 3	Sid 2	5000	0.0704	0.7808	-1.5111	1.6367
Trt 4	Sid 2	5000	0.0845	0.7986	-1.5658	1.6240
Trt 5	Sid 2	5000	0.0744	0.8048	-1.5636	1.5804

Output 31.7.4 displays the 95% credible interval of each treatment effect. The interval lines for treatment 1 (placebo) and treatment 2 (drug 1) have very little overlap, showing that these two treatments have different effects on the outcome. The other three drugs do not have much difference from the placebo.

Output 31.7.4 Treatment-Specific Effects

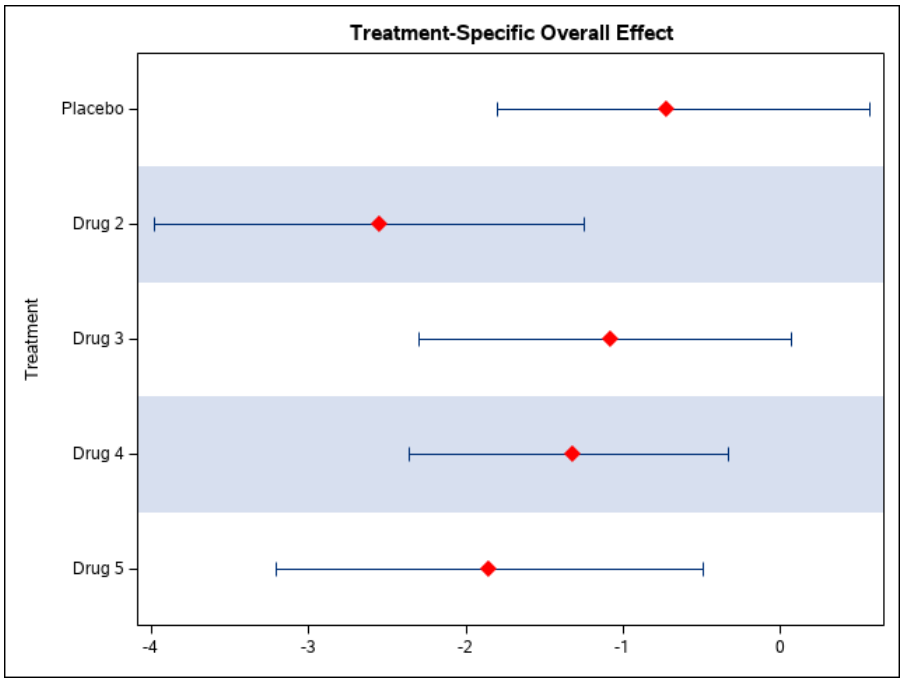

One quantity of interest is the overall effect difference of each drug compared to the placebo, controlling for the random effects. You can estimate the effect difference between each drug and the placebo conveniently by using the ESTIMATE statement as follows:

```
proc bglimm data=Parkinson2 seed=1315 nmc=50000 thin=10 outpost=PostSamples;
   class Trt Sid / order=internal;
   model mean = Trt / noint scale=SVar cprior=normal;
   random Trt / sub=Sid type=cs monitor=(1 to 2);
   estimate "Placebo_Drug2" Trt -1 1 0 0 0 / e;
   estimate "Placebo_Drug3" Trt -1 0 1 0 0 / e;
   estimate "Placebo_Drug4" Trt -1 0 0 1 0 / e;
   estimate "Placebo_Drug5" Trt -1 0 0 0 1 / e;
run;
```

Output 31.7.5 shows the results from the four ESTIMATE statements for comparison between each drug (coded 2 to 5) and the placebo.

Output 31.7.5 Effect Difference between Each Drug and Placebo

Results from ESTIMATE Statements				
Label	Mean	Standard Deviation	95% HPD Interval	
Placebo_Drug2	-1.8301	0.6191	-3.0728	-0.5609
Placebo_Drug3	-0.3576	0.6523	-1.6176	1.0036
Placebo_Drug4	-0.6002	0.6022	-1.7390	0.6686
Placebo_Drug5	-1.1273	0.7608	-2.6869	0.3665

Output 31.7.6 displays the credible interval of the effect difference for each drug compared to the placebo. The interval line for the effect difference between the placebo and drug 2 does not intersect with the vertical line at 0, implying that these two treatments have different effects on the outcome and that drug 2 leads to smaller off-time reduction than the placebo. The other three drugs do not differ much from the placebo. This confirms what is shown in Output 31.7.4.

Output 31.7.6 Effect Difference between Each Drug and Placebo

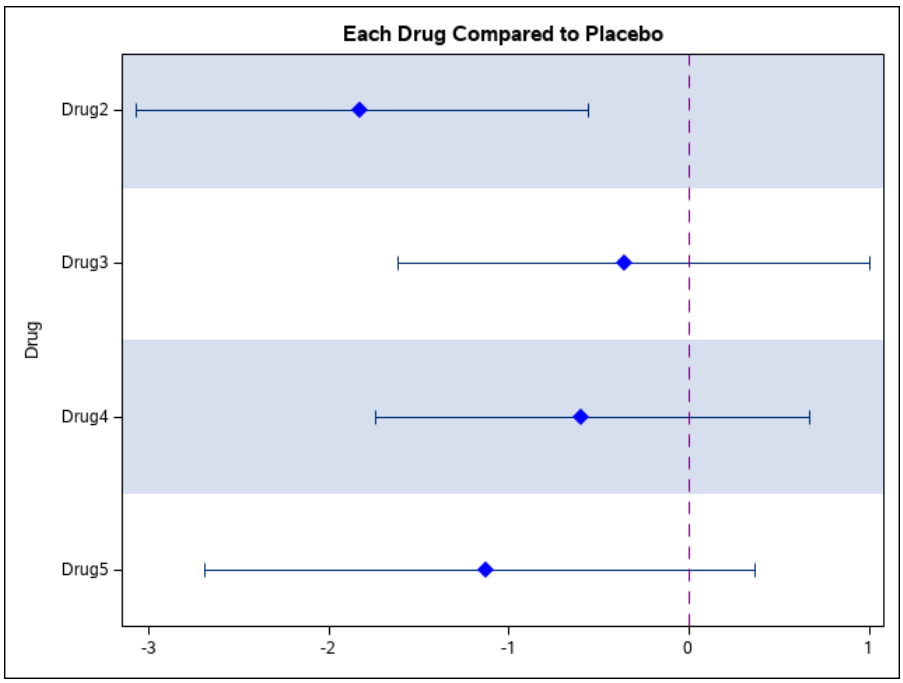

Example 31.8: Power Prior

(View the complete code for this example (bglmmex8.sas) in the example repository.)

Informative priors express specific information about parameters in a model. Although they often demonstrate the power of Bayesian analysis—in combining multiple sources of information—they are also a source of controversy, because analysts often view such priors as too personal and disagree with what it means to be informative. The power prior (Ibrahim and Chen 2000), which is a type of informative prior, is intended to be an automatic way to construct an informative prior that does not rely heavily on personal opinion: the power prior translates information from a data set, which can be a historical or previously observed data set, into distributional information about parameters that you can use for the analysis. For a detailed discussion on the theory and practice of the power prior, see Ibrahim et al. (2015). The power prior has long been seen as a useful class of informative priors for a variety of situations in which historical data are available.

The idea of the power prior is as follows. Suppose you want to infer on the parameter $\boldsymbol{\theta}$ by using a data set D, and you want to construct a prior distribution. Further suppose you have another data set, a historical data set D_0, that provides certain information about the parameter $\boldsymbol{\theta}$. For example, D_0 could come from an early data collection procedure that provides similar information about $\boldsymbol{\theta}$. The power prior enables you to use the D_0 data set to construct a prior on $\boldsymbol{\theta}$ that you can use to analyze D.

In addition to the D_0 data set, the power prior requires the specification of a scalar parameter a_0 that takes a value between 0 and 1. The prior has the formulation

$$\pi(\boldsymbol{\theta}|D_0, a_0) \propto L(\boldsymbol{\theta}|D_0)^{a_0}\pi_0(\boldsymbol{\theta})$$

where L is the likelihood function of the historical data set. The form of the power prior is similar to the form of a typical Bayesian analysis, where the posterior is the product of the likelihood function and a

(noninformative) prior distribution, referred to as $\pi_0(\boldsymbol{\theta})$. The difference is that for the power prior, the likelihood function is weighted by the parameter a_0, where $0 \leq a_0 \leq 1$. In this way, a weighted posterior distribution based on D_0 can now be used as a prior distribution for the current analysis that uses D. The corresponding posterior distribution is given by

$$\pi(\boldsymbol{\theta}|D, D_0, a_0) \propto L(\boldsymbol{\theta}|D)L(\boldsymbol{\theta}|D_0)^{a_0}\pi_0(\boldsymbol{\theta})$$

The following example demonstrates how to conduct a Bayesian analysis with a power prior for the benchmark approach that is a useful tool in toxicology. A benchmark dose (BMD) is defined as the dose of an environmental toxicant that corresponds to a prescribed change in response compared with the background response level.

The toxicological data consist of n binomial responses $\boldsymbol{y} = (y_1, \ldots, y_n)$, where $y_i \sim \text{binomial}(n_i, p_i)$, and where n_i is the number of animals tested and p_i is the probability that an animal gives an adverse response at dose level x_i:

$$p_i = \frac{\exp(\beta_0 + \beta_1 x_i)}{1 + \exp(\beta_0 + \beta_1 x_i)}, \quad i = 1, \ldots, n$$

This example uses two data sets. The first data set is from the Kociba study (Kociba et al. 1978), which is a lifetime feeding study of female and male Sprague Dawley rats. In this study, rats were placed in groups and tested at doses of 0, 1, 10, and 100 ng/kg/day of the drug tetrachlorodibenzodioxin (TCDD). The dosing was carried out by oral gavage, because the majority of human exposure to TCDD is oral. Inferences that were derived from the Kociba study had been used as the basis for risk assessments of the drug. The second data set is from the 1982 National Toxicology Program (NTP), a study in which groups of about 50 male rats, 50 female rats, and 50 male mice received TCDD as a suspension in 9:1 corn oil–acetone by gavage twice a week for two years. These are similar but not identical studies that can both provide information about how TCDD can increase the probability of adverse responses. Here, the Kociba data set is treated as a historical data set that provides information used in constructing a prior distribution for the NTP analysis.

```
data Kociba;
   input y n Dose;
datalines;
9  86  0
3  50  1
18 50  10
34 48  100
;

data NTP;
   input y n Dose;
datalines;
5  75  0
1  49  1.4
3  50  7.1
12 49  71
;
```

First, you can run a separate analysis of each data set and compare posterior estimates of the regression parameters to get a sense of how the data sets differ from each other:

```
proc bglimm data=Kociba seed=1181 nmc=10000;
   model y/n = Dose;
run;

proc bglimm data=NTP seed=1181 nmc=10000;
   model y/n = Dose;
run;
```

In the preceding code, PROC BGLIMM performs a logistic regression on each data set. The default flat priors are used for both the intercept β_0 and the slope β_1 of the variable Dose. The results are shown in Output 31.8.1 and Output 31.8.2, respectively, along with the posterior summary statistics for the two separate analyses. The posterior means of the slope parameter (Dose) are very similar in the two data sets, whereas the posterior means of the intercept are quite different.

Output 31.8.1 Posterior Summaries and Intervals for the Kociba Data

The BGLIMM Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	10000	-1.7828	0.2077	-2.1883	-1.3816
Dose	10000	0.0278	0.00394	0.0197	0.0351

Output 31.8.2 Posterior Summaries and Intervals for the NTP Data

The BGLIMM Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	10000	-3.0349	0.3761	-3.7461	-2.2948
Dose	10000	0.0264	0.00706	0.0142	0.0421

Output 31.8.3 displays the scatter plot of the joint posterior of the intercept and the Dose slope from the two data sets. The cluster of blue crosses is from the Kociba analysis, and the cluster of red circles is from the NTP analysis. You can see that the two posterior distributions are similar in their growth, or slope, estimates (Y axis) but different in their intercept estimates (X axis); this is consistent with the posterior summary statistics shown earlier.

Output 31.8.3 Joint Posterior Distributions from Two Separate Analyses

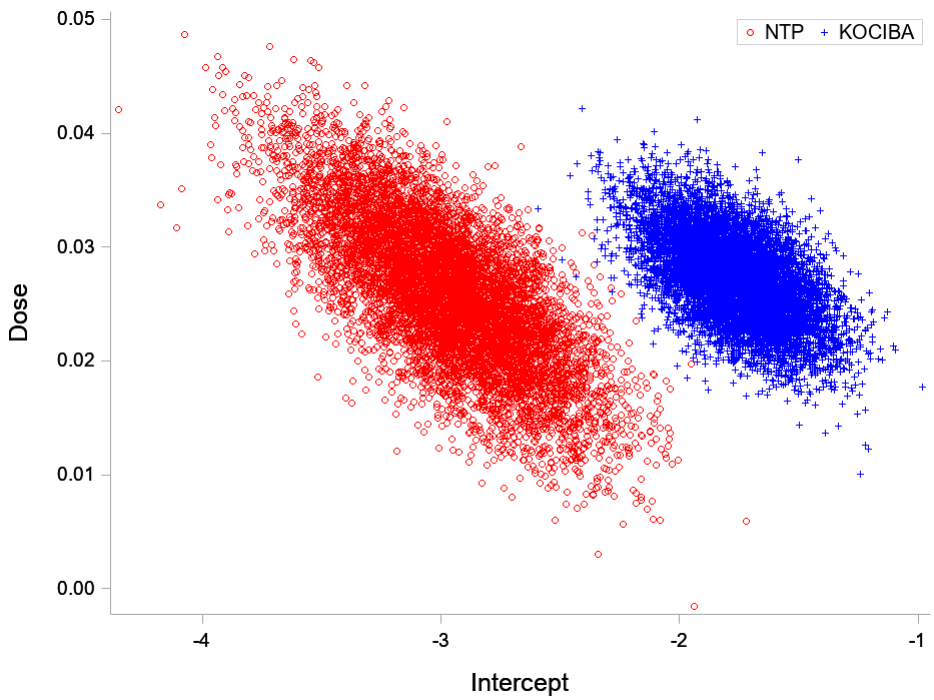

One purpose of the analysis is to predict the probability that an animal gives an adverse response at a dose level. For any dose value, the probability of an event can be computed using the logistic function of the linear predictor, $\beta_0 + \beta_1 x_i$. To predict the probability of an adverse response given a specific dose level (x), the estimates of the two regression parameters, $(\hat{\beta}_0, \hat{\beta}_1)$, are plugged in as follows:

$$\hat{p}(x) = \frac{\exp(\hat{\beta}_0 + \hat{\beta}_1 x)}{1 + \exp(\hat{\beta}_0 + \hat{\beta}_1 x)}$$

The predictive curves, from the two separate analyses, of the probability of reporting an adverse response for dose levels in the range of 0 to 100 are shown in Output 31.8.4.

Output 31.8.4 Predictive Curves from Two Separate Analyses

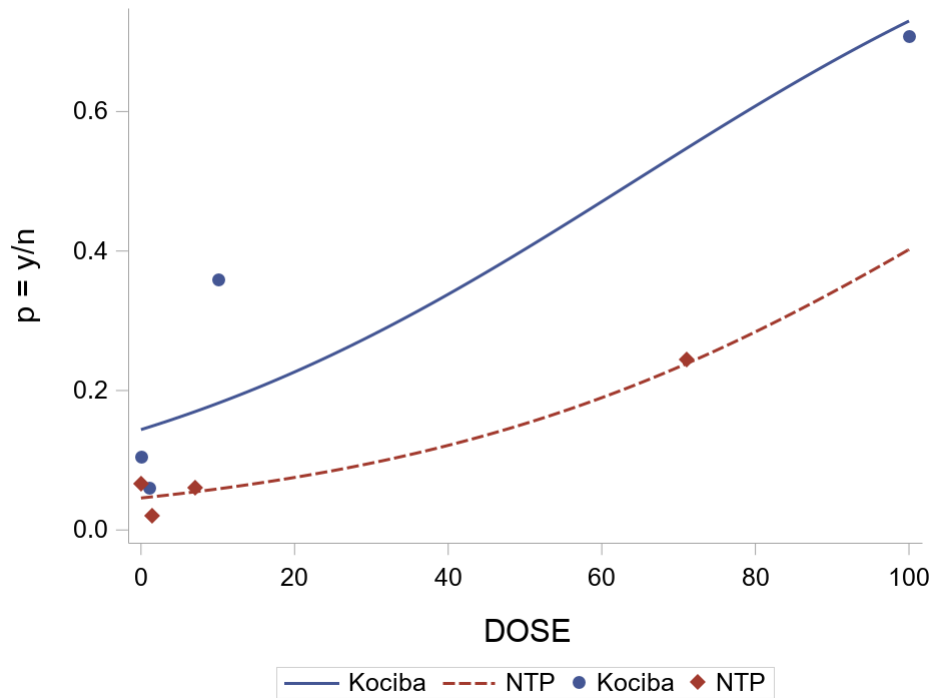

The solid blue line is the predictive curve for the Kociba analysis, and the dashed red line is the predictive curve for the NTP analysis; the round blue points represent the four observations in the Kociba data, and the diamond-shaped red points refer to the four observations in the NTP data. The predicted probability of an adverse response at any dose level seems to be different between these two studies.

Because the historical data are available, it is natural to incorporate these data into the comparison study by using the power prior to control the influence of the historical data on this study and to construct a suitable informative prior distribution on the model parameters. To use the power prior, you can use the Kociba study to construct the prior, and the posterior distribution of the analysis has the following form:

$$
\begin{aligned}
p(\theta|D, D_0, a_0) &\propto \prod_{i=1}^{n+n_0} f_i(y_i|\theta, x_i) \cdot \pi_0(\theta) \\
\text{where } f_i &= \begin{cases} f(y_i|\theta, x_i) & \text{for each } i \text{ in the current data set}, i = 1, \ldots, n \\ f(y_{0,i}|\theta, x_{0,i})^{a_0} & \text{for each } i \text{ in the historical data set}, i = 1, \ldots, n_0 \end{cases}
\end{aligned}
$$

where n is the sample size of the current data and n_0 is the sample size of the historical data. The form of the posterior distribution suggests that fitting the power prior is equivalent to combining two data sets. The posterior distribution is very much like a combined analysis that uses all data, albeit weighting those observations in D_0 with a_0.

The following DATA step merges the two data sets, creates a frequency variable a0, and assigns 1 for observations in the NTP data set and 0.3 for observations in the Kociba data set:

```
data Combined;
   set Kociba(in=i) NTP;
   a0 = 1;
   if i then a0 = 0.3;
   run;
```

Now you can use the Combined data set in PROC BGLIMM and perform a power prior analysis by using the FREQ statement:

```
proc bglimm data=Combined seed=1181 nmc=10000;
   model y/n = Dose;
   freq  a0 / notrunc;
run;
```

The FREQ statement assigns a frequency to each observation according to the a0 value: 0.3 to those in the Kociba data and 1 to those in the NTP data. The NOTRUNC option specifies that frequency values not be truncated to integers, to ensure that observations are properly weighted according to the power prior specification. By default, the frequency value would be truncated to an integer for any noninteger values, which is not what you want here, so the NOTRUNC option is needed.

The results are shown in Output 31.8.5 with the posterior summary statistics for the model parameters in the combined analysis that uses the power prior. The posterior mean of the intercept is between the posterior means of the intercept from the previous two separate analyses, as shown in Output 31.8.1 and Output 31.8.2, because the analysis of the current data incorporates some information from the historical data.

Output 31.8.5 Posterior Summaries and Intervals Using the Power Prior

The BGLIMM Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	10000	-2.6657	0.2654	-3.1773	-2.1574
Dose	10000	0.0275	0.00476	0.0188	0.0376

Output 31.8.6 displays the scatter plot of the posterior samples of the intercept and Dose slope coefficient from the combined analysis that uses the power prior (green), compared with the two previous separate analyses (blue and red). The combined analysis gives results that are between those from the two separate analyses; this is consistent with the findings in the posterior summary statistics shown previously.

Output 31.8.6 Posterior Distributions Using a Power Prior

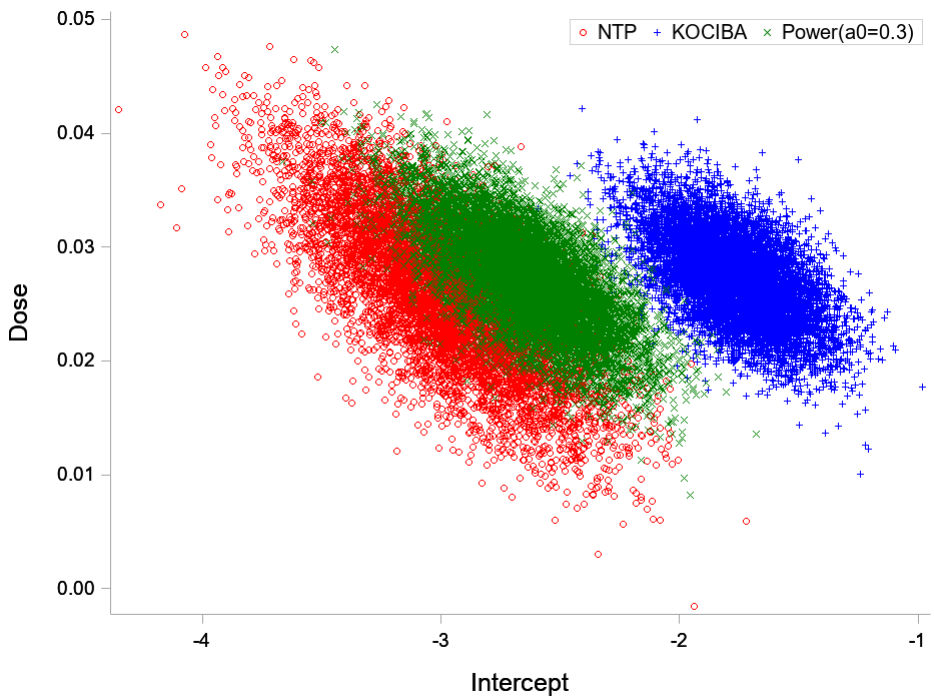

Output 31.8.7 displays the predictive curve of the adverse event probability from the joint data analysis that uses the power prior, in addition to the two predictive curves of the event probability from the two separate analyses. Similarly, the combined analysis that uses the power prior moves the current data closer to the historical data.

Output 31.8.7 Predictive Curves Using a Power Prior

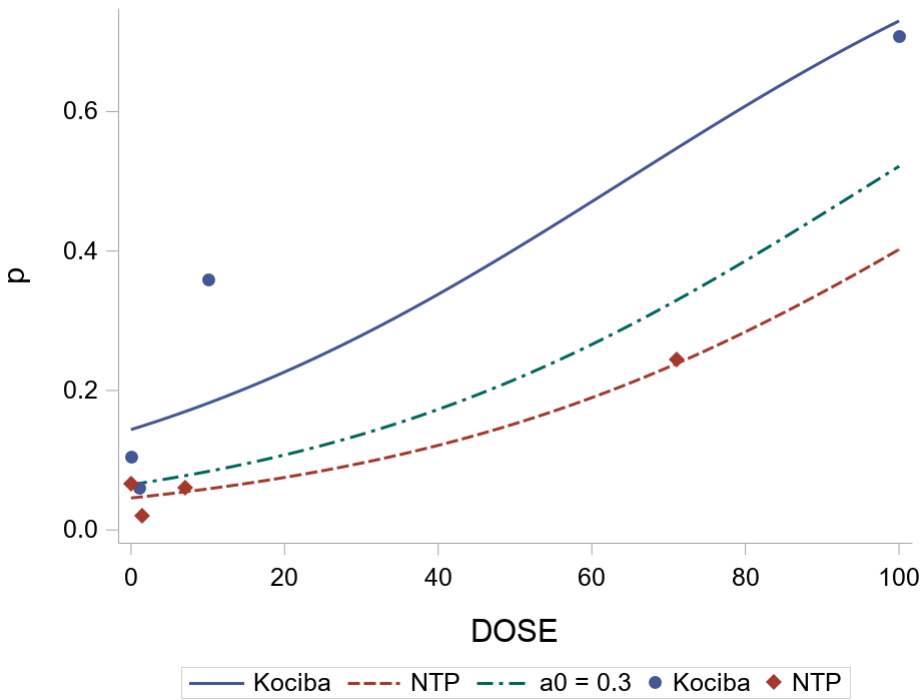

In the previous analysis, the selection of $a_0 = 0.3$ is arbitrary. In practice, you need to get a sense of what a proper level of borrowing from the historical data should be. The question is then how to select a proper a_0 value in a systematic way. Although it is possible to treat a_0 as a parameter (Neuenschwander, Branson, and Spiegelhalter 2009), the formulation requires a normalizing constant in the power prior, which can sometimes be challenging to compute. A viable alternative is to rely on a model-selection criterion, such as the deviance information criterion (DIC) (Ibrahim et al. 2015), to choose an optimal value of a_0. This can be achieved by repeating a power prior analysis by using different values of a_0 (for example, gridded values from 0 to 1) and by selecting the a_0 that produces the lowest DIC value. You can use the BY statement in PROC BGLIMM in this case: each BY group consists of the combined data set that has a different value of a_0 for the historical portion of the data. The following DATA step makes many copies of the combined data, with a_0 taking the values $0, 0.1, 0.2, 0.3, \ldots, 0.9, 1$ for the Kociba data set and taking the value 1 for the NTP data set. Then, the observations in the new SAS data set CombinedBy are sorted according to the val variable. All models can be run in PROC BGLIMM sequentially via the BY statement, which offers a convenient way to perform separate and independent analyses.

```
data CombinedBy;
   set Kociba(in=i) NTP;
   do val = 0 to 1 by 0.1;
      a0 = 1;
      dicIdx = 1;
      if i then do;
         a0 = val;
         dicIdx = 0;
         end;
      output;
      end;
```

```
proc sort;
   by val;
run;

ods output dic = dic0;
proc bglimm data=CombinedBy seed=1181 nmc=10000 dic(include=dicIdx);
   model y/n = Dose;
   freq  a0 / notrunc;
   by val;
run;
```

The BY statement repeats the power prior analysis according to unique values in the val variable. The DIC option requests the DIC calculation for each BY group. However, you cannot use the default DIC option here, because it leads to incorrect results. The reason is that the deviance in the DIC method (Spiegelhalter et al. 2002) depends only on the likelihood of the current data, which are from the NTP data set. Here, by default, PROC BGLIMM computes the DIC by using observations from both the NTP and Kociba data sets. Instead, specify the INCLUDE=DICIDX suboption in the DIC option to indicate which observations to include in the DIC calculation. This suboption specifies the DIC computation, including the observations for which the value of dicIdx is 1 (the NTP data) and excluding the observations for which the value of dicIdx value is 0 (the Kociba data). This way, the DIC values that PROC BGLIMM returns are correct.

Output 31.8.8 plots the DIC values together with their corresponding a_0 values. DIC reaches its minimum when a_0 is 0.1, suggesting that 0.1 might be an optimal choice for the prior weight. A small value of a_0 corresponds to a small amount of borrowing from the historical data set. This intuitively agrees with the early results: the joint posterior distributions (of beta0 and beta1) from the two independent analyses are well separated, indicating that the two data sets might contain very different information about the model parameters. Hence, you might not want to borrow too much information from the historical data set.

Output 31.8.8 DIC Values with Different Prior Weights

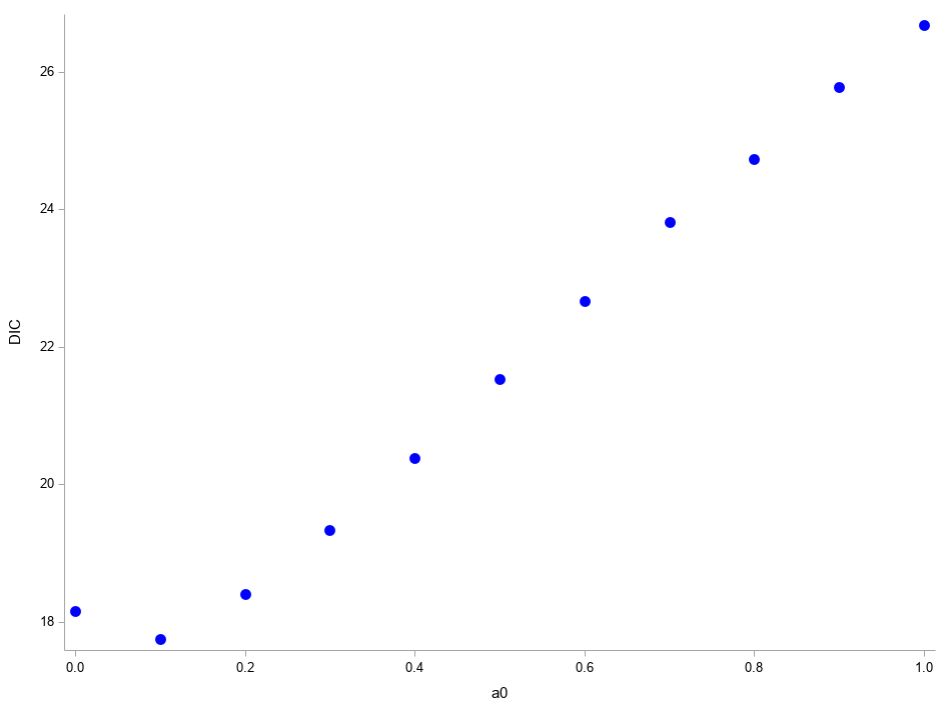

Next, the power prior analysis is conducted with $a_0 = 0.1$. The DATA step assigns $a_0 = 0.1$ to the observations in the Kociba data set and $a_0 = 1$ to those in the NTP data set, as follows:

```
data Combined;
   set Kociba(in=i) NTP;
   a0 = 1;
   if i then a0 = 0.1;
   run;

proc bglimm data=Combined seed=1181 nmc=10000;
   model y/n = Dose;
   freq  a0 / notrunc;
run;
```

The posterior summary statistics for the model parameters are shown in Output 31.8.9. The posterior mean of the intercept is a little farther away from the posterior mean of the intercept from the analysis that uses only the Kociba data than the posterior mean of the intercept in the previous power prior analysis in which $a_0 = 0.3$ (Output 31.8.5), because this analysis incorporates less information from the historical data in which $a_0 = 0.1$.

Output 31.8.9 Posterior Summaries and Intervals Using the Power Prior ($a_0 = 0.1$)

The BGLIMM Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Intercept	10000	-2.8936	0.3372	-3.5207	-2.2265
Dose	10000	0.0272	0.00605	0.0165	0.0402

The power prior is a useful class of informative priors when you want to use information that is contained in separate data sets. The FREQ statement and the INCLUDE= suboption in the DIC option give you a convenient way to perform the power prior analysis and select the a_0 weight. You can use them in all models that PROC BGLIMM supports. If you want to borrow information from multiple historical data sets, you can assign different a_0 values to the different data sets, or you can conduct a gridded DIC search over a larger combination of a_0 values to identify the optimal a_0 value for the level of information borrowing from the historical data, as this example demonstrates.

References

Breslow, N. E., and Clayton, D. G. (1993). "Approximate Inference in Generalized Linear Mixed Models." *Journal of the American Statistical Association* 88:9–25.

Brown, H., and Prescott, R. (1999). *Applied Mixed Models in Medicine*. New York: John Wiley & Sons.

Clayton, D. G., and Kaldor, J. (1987). "Empirical Bayes Estimates of Age-Standardized Relative Risks for Use in Disease Mapping." *Biometrics* 43:671–681.

Davidian, M., and Giltinan, D. M. (1995). *Nonlinear Models for Repeated Measurement Data*. New York: Chapman & Hall.

Dias, S., and Ades, A. E. (2016). "Absolute or Relative Effects? Arm-Based Synthesis of Trial Data." *Research Synthesis Methods* 7:23–28.

Eilers, P. H. C., and Marx, B. D. (1996). "Flexible Smoothing with *B*-Splines and Penalties." *Statistical Science* 11:89–121. With discussion.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). "Beta Regression for Modelling Rates and Proportions." *Journal of Applied Statistics* 31:799–815.

Fuller, W. A. (1976). *Introduction to Statistical Time Series*. New York: John Wiley & Sons.

Gamerman, D. (1997). "Sampling from the Posterior Distribution in Generalized Linear Models." *Statistics and Computing* 7:57–68.

Gelman, A., and Hill, J. (2007). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge: Cambridge University Press.

Hoffman, M. D., and Gelman, A. (2014). "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research* 15:1351–1381.

Huynh, H., and Feldt, L. S. (1970). "Conditions Under Which Mean Square Ratios in Repeated Measurements Designs Have Exact F-Distributions." *Journal of the American Statistical Association* 65:1582–1589.

Ibrahim, J. G., and Chen, M.-H. (2000). "Power Prior Distributions for Regression Models." *Statistical Science* 15:46–60.

Ibrahim, J. G., Chen, M.-H., Gwon, Y., and Chen, F. (2015). "The Power Prior: Theory and Applications." *Statistics in Medicine* 34:3724–3749.

Jennrich, R. I., and Schluchter, M. D. (1986). "Unbalanced Repeated-Measures Models with Structured Covariance Matrices." *Biometrics* 42:805–820.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998). "Markov Chain Monte Carlo in Practice: A Roundtable Discussion." *American Statistician* 52:93–100.

Kenward, M. G. (1987). "A Method for Comparing Profiles of Repeated Measurements." *Journal of the Royal Statistical Society, Series C* 36:296–308.

Kiernan, K. (2018). "Insights into Using the GLIMMIX Procedure to Model Categorical Outcomes with Random Effects." In *Proceedings of the SAS Global Forum 2018 Conference*. Cary, NC: SAS Institute Inc. https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2018/2179-2018.pdf.

Lee, A. W. (2014). "Review of Mixed Treatment Comparisons in Published Systematic Reviews Shows Marked Increase Since 2009." *Journal of Clinical Epidemiology* 67:138–143.

Lin, L., Zhang, J., Hodges, J. S., and Chu, H. (2017). "Performing Arm-Based Network Meta-analysis in R with the pcnetmeta Package." *Journal of Statistical Software* 80:1–25.

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and Schabenberger, O. (2006). *SAS for Mixed Models*. 2nd ed. Cary, NC: SAS Institute Inc.

Little, R. J. A., and Rubin, D. B. (2002). *Statistical Analysis with Missing Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Macchiavelli, R. E., and Arnold, S. F. (1994). "Variable Order Ante-dependence Models." *Communications in Statistics—Theory and Methods* 23:2683–2699.

McCullagh, P. (1980). "Regression Models for Ordinal Data." *Journal of the Royal Statistical Society, Series B* 42:109–142.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Neal, R. M. (2003). "Slice Sampling." *Annals of Statistics* 31:705–757.

Neal, R. M. (2011). "MCMC Using Hamiltonian Dynamics." In *Handbook of Markov Chain Monte Carlo*, edited by S. Brooks, A. Gelman, G. L. Jones, and X.-L. Meng, 113–161. Boca Raton, FL: CRC Press.

Neuenschwander, B., Branson, M., and Spiegelhalter, D. J. (2009). "A Note on the Power Prior." *Statistics in Medicine* 28:3562–3566.

O'Malley, A. J., and Zaslavsky, A. M. (2008). "Domain-Level Covariance Analysis for Multilevel Survey Data with Structured Nonresponse." *Journal of the American Statistical Association* 103:1405–1418.

Patel, H. I. (1991). "Analysis of Incomplete Data from a Clinical Trial with Repeated Measurements." *Biometrika* 78:609–619.

Pothoff, R. F., and Roy, S. N. (1964). "A Generalized Multivariate Analysis of Variance Model Useful Especially for Growth Curve Problems." *Biometrika* 51:313–326.

Rubin, D. B. (1976). "Inference and Missing Data." *Biometrika* 63:581–592.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and Van der Linde, A. (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society, Series B* 64:583–616. With discussion.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2012). *Categorical Data Analysis Using SAS*. 3rd ed. Cary, NC: SAS Institute Inc.

Vonesh, E. F. (2012). *Generalized Linear and Nonlinear Models for Correlated Data: Theory and Applications Using SAS*. Cary, NC: SAS Institute Inc.

Vonesh, E. F., Chinchilli, V. M., and Pu, K. (1996). "Goodness-of-Fit in Generalized Nonlinear Mixed-Effects Models." *Biometrics* 52:572–587.

Chapter 32
The BOXPLOT Procedure

Contents

Overview: BOXPLOT Procedure . . . **1424**
- Traditional Graphics and ODS Graphics . . . 1425

Getting Started: BOXPLOT Procedure . . . **1425**
- Creating Box Plots from Raw Data . . . 1426
- Creating Box Plots from Summary Data . . . 1428
- Saving Summary Data with Outliers . . . 1431

Syntax: BOXPLOT Procedure . . . **1434**
- PROC BOXPLOT Statement . . . 1434
- BY Statement . . . 1435
- ID Statement . . . 1436
- INSET Statement . . . 1436
- INSETGROUP Statement . . . 1439
- PLOT Statement . . . 1442

Details: BOXPLOT Procedure . . . **1467**
- Summary Statistics Represented by Box Plots . . . 1467
- Output Data Sets . . . 1467
- Input Data Sets . . . 1469
- Styles of Box Plots . . . 1472
- Percentile Definitions . . . 1473
- Missing Values . . . 1474
- Continuous Group Variables . . . 1474
- Positioning Insets . . . 1476
- Displaying Blocks of Data . . . 1481
- Clipping Extreme Values . . . 1483
- ODS Graphics . . . 1487

Examples: BOXPLOT Procedure . . . **1488**
- Example 32.1: Displaying Summary Statistics in a Box Plot . . . 1488
- Example 32.2: Using Box Plots to Compare Groups . . . 1489
- Example 32.3: Creating Various Styles of Box-and-Whiskers Plots . . . 1492
- Example 32.4: Creating Notched Box-and-Whiskers Plots . . . 1496
- Example 32.5: Creating Box-and-Whiskers Plots with Varying Widths . . . 1498
- Example 32.6: Creating Horizontal Box-and-Whiskers Plots . . . 1500

References . . . **1501**

Overview: BOXPLOT Procedure

The BOXPLOT procedure creates side-by-side box-and-whiskers plots of measurements that are organized in groups. A box-and-whiskers plot displays the mean, quartiles, and minimum and maximum observations for a group. Throughout this chapter, this type of plot, which can contain one or more box-and-whiskers plots, is referred to as a *box plot*.

The PLOT statement of the BOXPLOT procedure produces a box plot. You can specify more than one PLOT statement to produce multiple box plots. You can use options in the PLOT statement to do the following:

- control the style of the box-and-whiskers plots
- specify one of several methods for calculating quantile statistics (percentiles)
- add block legends and symbol markers to reveal stratification in data
- display vertical and horizontal reference lines
- control axis values and labels
- overlay the box plot with plots of additional variables
- control the layout and appearance of the plot

The INSET and INSETGROUP statements produce boxes or tables (referred to as *insets*) of summary statistics or other data on a box plot. An INSET statement produces an inset of statistics that pertain to the entire box plot. An INSETGROUP statement produces an inset that contains statistics that are calculated separately for each group. An INSET or INSETGROUP statement by itself does not produce a display; it must be used with a PLOT statement.

You can use options in an INSET or INSETGROUP statement to control insets in these ways:

- specify the position of the inset
- specify a header for the inset
- specify graphical enhancements, such as background colors, text colors, text height, text font, and drop shadows

Traditional Graphics and ODS Graphics

The BOXPLOT procedure can produce two kinds of graphical output:

- traditional graphics
- ODS Statistical Graphics output

Traditional graphics are saved in graphics catalogs with entry type GRSEG. Their appearance is controlled by global statements such as the GOPTIONS, AXIS, and SYMBOL statements (as described in *SAS/GRAPH: Reference*) and numerous specialized PLOT statement options. You must have a SAS/GRAPH license to produce traditional graphics.

ODS Statistical Graphics (or ODS Graphics for short) is an extension to the Output Delivery System (ODS). Graphs are produced in standard image file formats (such as PNG) instead of graphics catalogs, and the details of their appearance and layout are controlled by ODS styles and templates. When ODS Graphics is enabled (for example, with the ODS GRAPHICS ON statement) PROC BOXPLOT produces ODS Graphics output. Otherwise, it produces traditional graphics. See Chapter 24, "Statistical Graphics Using ODS," for a thorough discussion of ODS Graphics.

Global graphics statements (GOPTIONS, AXIS, and SYMBOL, for example) and PLOT statement options that specify details of graph appearance (such as CBOXFILL= and FONT=) are ignored when ODS Graphics is enabled. Some PLOT statement options do affect ODS Graphics output, as indicated in the section "PLOT Statement Options" on page 1442.

See the section "Getting Started: BOXPLOT Procedure" on page 1425 for examples that produce box plots via the traditional graphics system and ODS Graphics.

Getting Started: BOXPLOT Procedure

This section introduces the BOXPLOT procedure with simple examples that demonstrate commonly used options. Complete syntax for the BOXPLOT procedure is presented in the section "Syntax: BOXPLOT Procedure" on page 1434, and advanced examples are presented in the section "Examples: BOXPLOT Procedure" on page 1488.

Creating Box Plots from Raw Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/boxgs.sas.)

A petroleum company uses a turbine to heat water into steam that is pumped into the ground to make oil less viscous and easier to extract. This process occurs 20 times daily, and the amount of power (in kilowatts) used to heat the water to the desired temperature is recorded. The following statements create a SAS data set called Turbine that contains the power output measurements for 10 nonconsecutive days:

```
data Turbine;
   informat Day date7.;
   format Day date5.;
   label KWatts='Average Power Output';
   input Day @;
   do i=1 to 10;
      input KWatts @;
      output;
   end;
   drop i;
   datalines;
05JUL94 3196 3507 4050 3215 3583 3617 3789 3180 3505 3454
05JUL94 3417 3199 3613 3384 3475 3316 3556 3607 3364 3721
06JUL94 3390 3562 3413 3193 3635 3179 3348 3199 3413 3562
06JUL94 3428 3320 3745 3426 3849 3256 3841 3575 3752 3347
07JUL94 3478 3465 3445 3383 3684 3304 3398 3578 3348 3369
07JUL94 3670 3614 3307 3595 3448 3304 3385 3499 3781 3711
08JUL94 3448 3045 3446 3620 3466 3533 3590 3070 3499 3457
08JUL94 3411 3350 3417 3629 3400 3381 3309 3608 3438 3567
11JUL94 3568 2968 3514 3465 3175 3358 3460 3851 3845 2983
11JUL94 3410 3274 3590 3527 3509 3284 3457 3729 3916 3633
12JUL94 3153 3408 3741 3203 3047 3580 3571 3579 3602 3335
12JUL94 3494 3662 3586 3628 3881 3443 3456 3593 3827 3573
13JUL94 3594 3711 3369 3341 3611 3496 3554 3400 3295 3002
13JUL94 3495 3368 3726 3738 3250 3632 3415 3591 3787 3478
14JUL94 3482 3546 3196 3379 3559 3235 3549 3445 3413 3859
14JUL94 3330 3465 3994 3362 3309 3781 3211 3550 3637 3626
15JUL94 3152 3269 3431 3438 3575 3476 3115 3146 3731 3171
15JUL94 3206 3140 3562 3592 3722 3421 3471 3621 3361 3370
18JUL94 3421 3381 4040 3467 3475 3285 3619 3325 3317 3472
18JUL94 3296 3501 3366 3492 3367 3619 3550 3263 3355 3510
;
```

In the data set Turbine, each observation contains the date and the power output for a single heating. The first 20 observations contain the outputs for the first day, the second 20 observations contain the outputs for the second day, and so on. Because the variable Day classifies the observations into groups, it is referred to as the *group variable*. The variable KWatts contains the output measurements and is referred to as the *analysis variable*.

The following statements create a box plot that shows the distribution of power output for each day:

```
ods graphics on;
title 'Box Plot for Power Output';
proc boxplot data=Turbine;
   plot KWatts*Day;
run;
```

The input data set Turbine is specified with the DATA= option in the PROC BOXPLOT statement. The PLOT statement requests a box-and-whiskers plot for each group of data. After the keyword PLOT, you specify the analysis variable (in this case, KWatts), followed by an asterisk and the group variable (Day). The ODS GRAPHICS ON statement specified before the PROC BOXPLOT statement enables ODS Graphics, so the box plot is produced using ODS Graphics instead of traditional graphics. The box plot is shown in Figure 32.1.

Figure 32.1 Box Plot for Power Output Data

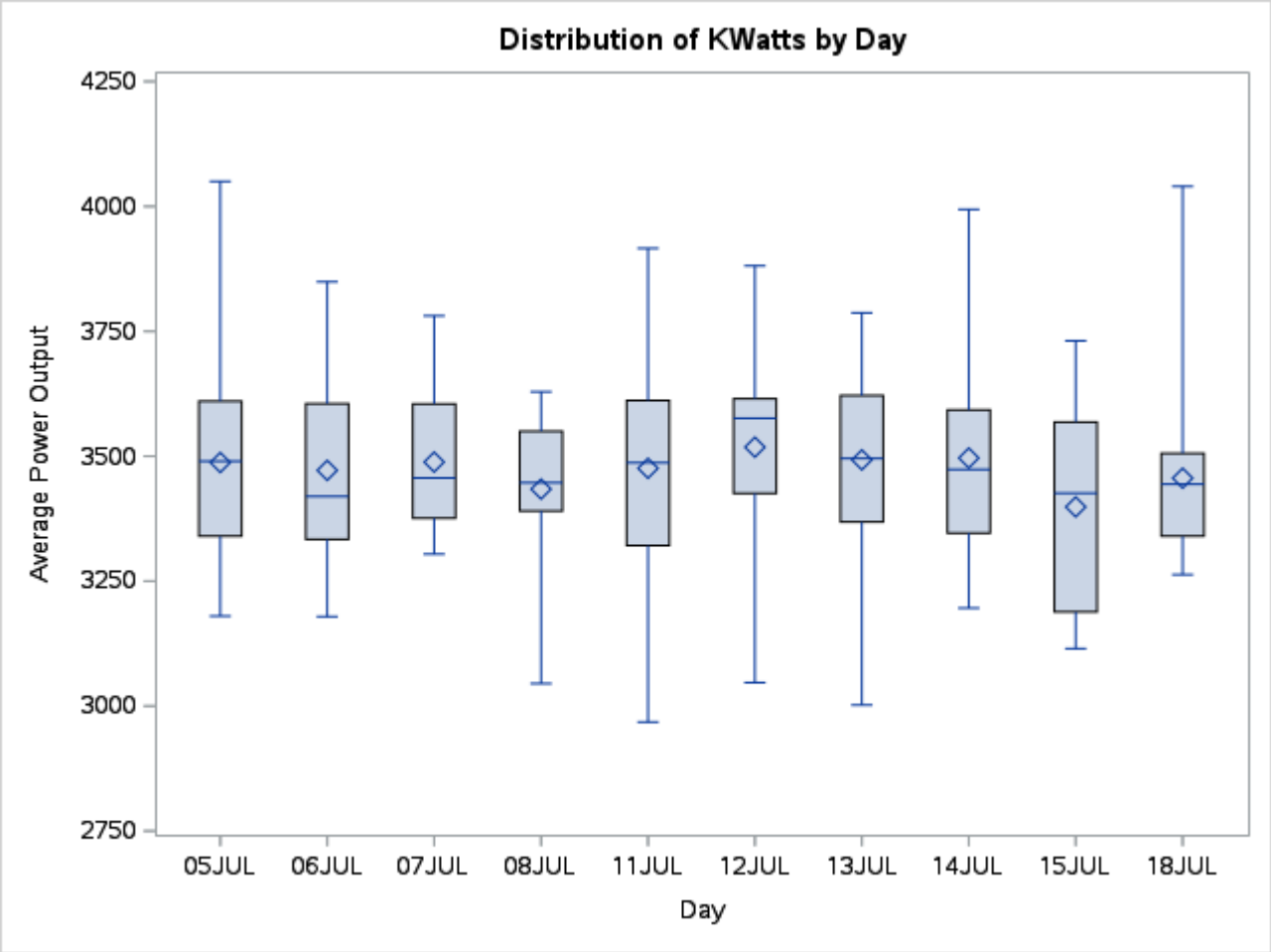

The box plot displayed in Figure 32.1 represents summary statistics for the analysis variable KWatts. Each of the 10 box-and-whiskers plots describes the variable KWatts for a particular day. The plot elements and the statistics they represent are as follows:

- The length of the box represents the interquartile range (the distance between the 25th and 75th percentiles).

- The symbol in the box interior represents the group mean.
- The horizontal line in the box interior represents the group median.
- The vertical lines (called *whiskers*) that issue from the box extend to the group minimum and maximum values.

Creating Box Plots from Summary Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/boxgs.sas.)

The previous example illustrates how you can create box plots from raw data. However, in some applications the data are provided as summary statistics. This example illustrates how you can use the BOXPLOT procedure with data of this type.

The following statements create the data set Oilsum, which provides the data from the preceding example in summarized form:

```
data Oilsum;
   input Day KWattsL KWatts1 KWattsX KWattsM
             KWatts3 KWattsH KWattsS KWattsN;
   informat Day date7. ;
   format Day date5. ;
   label Day    ='Date of Measurement'
         KWattsL='Minimum Power Output'
         KWatts1='25th Percentile'
         KWattsX='Average Power Output'
         KWattsM='Median Power Output'
         KWatts3='75th Percentile'
         KWattsH='Maximum Power Output'
         KWattsS='Standard Deviation of Power Output'
         KWattsN='Group Sample Size';
   datalines;
05JUL94 3180 3340.0 3487.40 3490.0 3610.0 4050 220.3 20
06JUL94 3179 3333.5 3471.65 3419.5 3605.0 3849 210.4 20
07JUL94 3304 3376.0 3488.30 3456.5 3604.5 3781 147.0 20
08JUL94 3045 3390.5 3434.20 3447.0 3550.0 3629 157.6 20
11JUL94 2968 3321.0 3475.80 3487.0 3611.5 3916 258.9 20
12JUL94 3047 3425.5 3518.10 3576.0 3615.0 3881 211.6 20
13JUL94 3002 3368.5 3492.65 3495.5 3621.5 3787 193.8 20
14JUL94 3196 3346.0 3496.40 3473.5 3592.5 3994 212.0 20
15JUL94 3115 3188.5 3398.50 3426.0 3568.5 3731 199.2 20
18JUL94 3263 3340.0 3456.05 3444.0 3505.5 4040 173.5 20
;
```

Oilsum contains exactly one observation for each group. Note that, as in the previous example, the groups are indexed by the variable Day. A listing of Oilsum is shown in Figure 32.2.

Figure 32.2 The Summary Data Set Oilsum

Box Plot for Power Output

Day	KWattsL	KWatts1	KWattsX	KWattsM	KWatts3	KWattsH	KWattsS	KWattsN
05JUL	3180	3340.0	3487.40	3490.0	3610.0	4050	220.3	20
06JUL	3179	3333.5	3471.65	3419.5	3605.0	3849	210.4	20
07JUL	3304	3376.0	3488.30	3456.5	3604.5	3781	147.0	20
08JUL	3045	3390.5	3434.20	3447.0	3550.0	3629	157.6	20
11JUL	2968	3321.0	3475.80	3487.0	3611.5	3916	258.9	20
12JUL	3047	3425.5	3518.10	3576.0	3615.0	3881	211.6	20
13JUL	3002	3368.5	3492.65	3495.5	3621.5	3787	193.8	20
14JUL	3196	3346.0	3496.40	3473.5	3592.5	3994	212.0	20
15JUL	3115	3188.5	3398.50	3426.0	3568.5	3731	199.2	20
18JUL	3263	3340.0	3456.05	3444.0	3505.5	4040	173.5	20

There are eight summary variables in Oilsum:

- KWattsL contains the group minima (low values).
- KWatts1 contains the 25th percentile (first quartile) for each group.
- KWattsX contains the group means.
- KWattsM contains the group medians.
- KWatts3 contains the 75th percentile (third quartile) for each group.
- KWattsH contains the group maxima (high values).
- KWattsS contains the group standard deviations.
- KWattsN contains the group sizes.

You can use this data set as input to the BOXPLOT procedure by specifying it with the HISTORY= option in the PROC BOXPLOT statement. Detailed requirements for HISTORY= data sets are presented in the section "HISTORY= Data Set" on page 1471.

The following statements produce a box plot of the summary data from the Oilsum data set:

```
ods graphics off;
title 'Box Plot for Power Output';
symbol value=dot color=salmon height=3 pct;
proc boxplot history=Oilsum;
   plot KWatts*Day / cframe   = vligb
                     cboxes   = dagr
                     cboxfill = ywh;
run;
goptions reset=symbol;
```

The ODS GRAPHICS OFF statement specified before the PROC BOXPLOT statement disables ODS Graphics, so the box plot is produced using traditional graphics. The SYMBOL statement and options specified after the slash (/) in the PLOT statement apply to traditional graphics. The CFRAME= option specifies the background color of the graph frame, the CBOXES= option specifies the color of the box outlines and whiskers, and the CBOXFILL= option specifies the color of the box interiors. For more information about SYMBOL statements and the colors specified in the PLOT statement options, see *SAS/GRAPH: Reference*. The resulting box plot is shown in Figure 32.3.

Figure 32.3 Traditional Graphics Box Plot

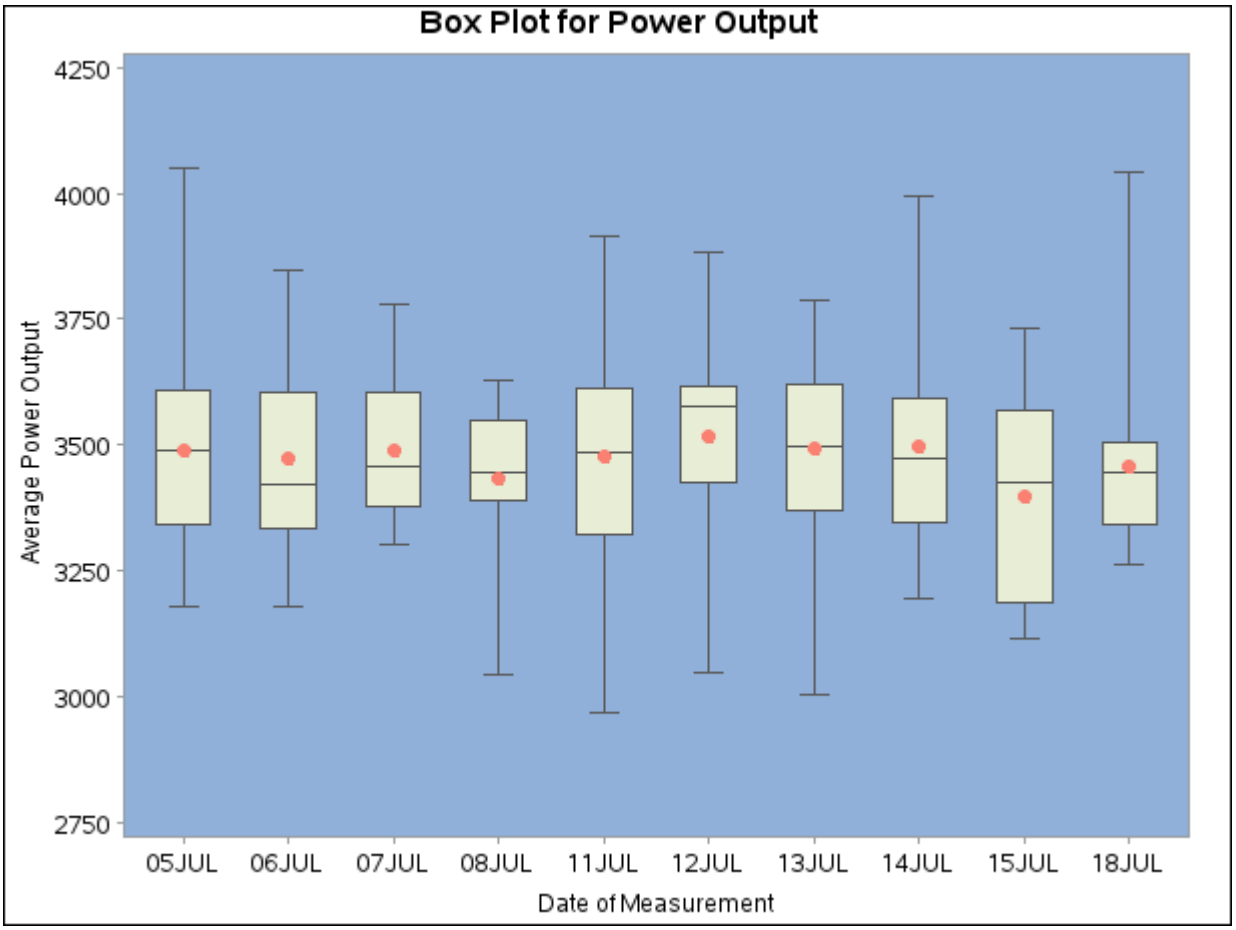

Saving Summary Data with Outliers

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/boxgs.sas.)

In a *schematic* box plot, outlier values within a group are plotted as separate points beyond the whiskers of the box-and-whiskers plot. See the section "Styles of Box Plots" on page 1472 and the description of the BOXSTYLE= option for a complete description of schematic box plots.

The following statements use the BOXSTYLE= option to produce a schematic box plot of the data from the Turbine data set. The OUTBOX= option creates a summary data set named OilSchematic.

```
title 'Schematic Box Plot for Power Output';
ods graphics on;
proc boxplot data=Turbine;
   plot KWatts*Day / boxstyle = schematic
                     outbox   = OilSchematic;
run;
```

The schematic box plot is shown in Figure 32.4. Note the outliers plotted for several of the groups.

Figure 32.4 Schematic Box Plot of Power Output

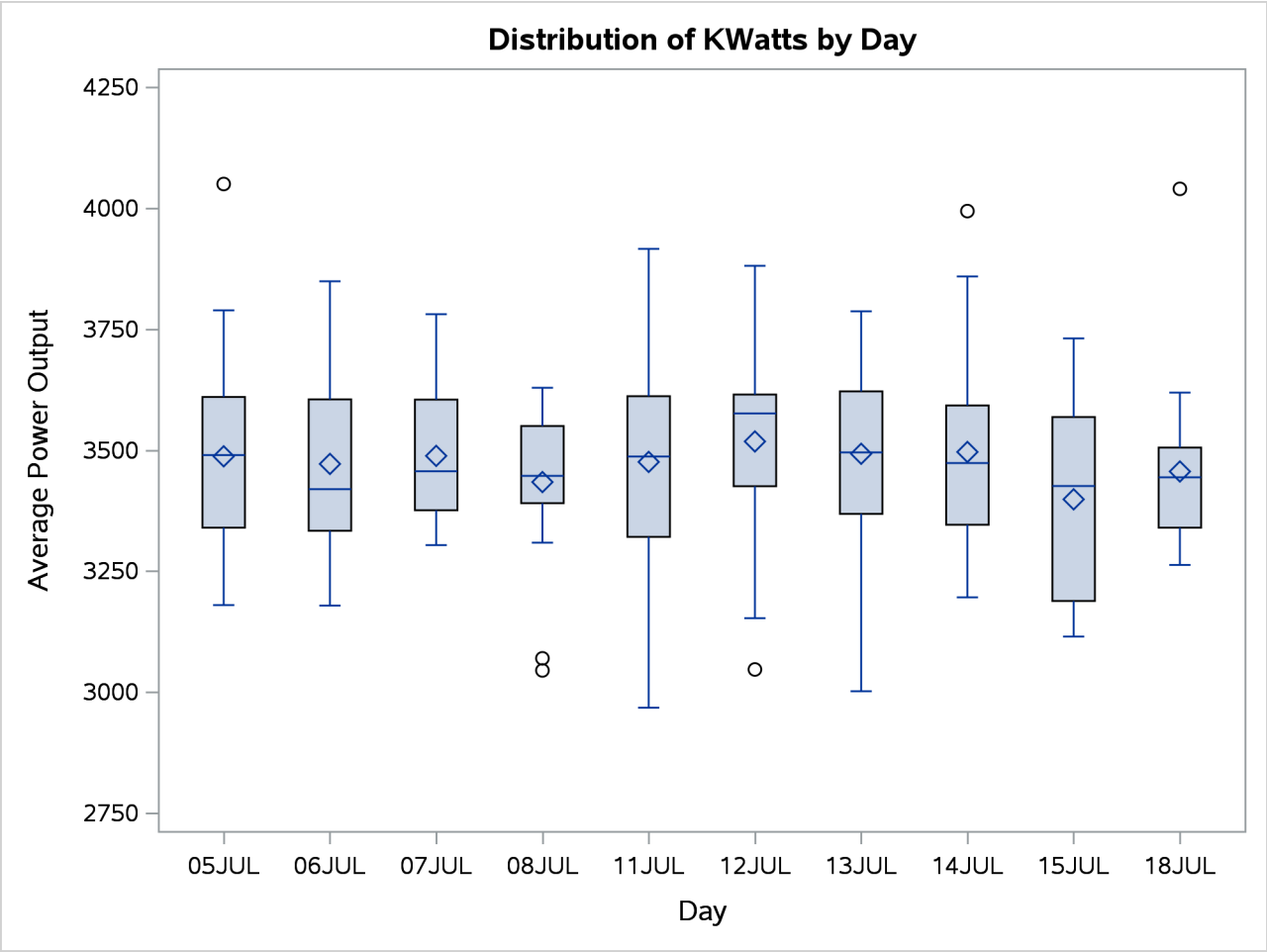

Whereas the Oilsum data set from the section "Creating Box Plots from Summary Data" on page 1428 contains a *variable* for each summary statistic and one observation per group, the OUTBOX= data set OilSchematic contains one *observation* for each summary statistic in each group. The _TYPE_ variable identifies the statistic and the _VALUE_ variable contains its value. In addition, the OilSchematic data set contains an observation that records each outlier value for each group. Figure 32.5 shows a partial listing of the OilSchematic data set.

Figure 32.5 The Summary Data Set OilSchematic

Schematic Box Plot for Power Output

Day	_VAR_	_TYPE_	_VALUE_
05JUL	KWatts	N	20.00
05JUL	KWatts	MIN	3180.00
05JUL	KWatts	Q1	3340.00
05JUL	KWatts	MEAN	3487.40
05JUL	KWatts	MEDIAN	3490.00
05JUL	KWatts	Q3	3610.00
05JUL	KWatts	MAX	4050.00
05JUL	KWatts	STDDEV	220.26
05JUL	KWatts	HIWHISKR	3789.00
05JUL	KWatts	HIGH	4050.00
06JUL	KWatts	N	20.00
06JUL	KWatts	MIN	3179.00
06JUL	KWatts	Q1	3333.50
06JUL	KWatts	MEAN	3471.65
06JUL	KWatts	MEDIAN	3419.50
06JUL	KWatts	Q3	3605.00
06JUL	KWatts	MAX	3849.00
06JUL	KWatts	STDDEV	210.43
07JUL	KWatts	N	20.00
07JUL	KWatts	MIN	3304.00
07JUL	KWatts	Q1	3376.00
07JUL	KWatts	MEAN	3488.30
07JUL	KWatts	MEDIAN	3456.50
07JUL	KWatts	Q3	3604.50
07JUL	KWatts	MAX	3781.00
07JUL	KWatts	STDDEV	147.02
08JUL	KWatts	N	20.00
08JUL	KWatts	MIN	3045.00
08JUL	KWatts	Q1	3390.50
08JUL	KWatts	MEAN	3434.20
08JUL	KWatts	MEDIAN	3447.00
08JUL	KWatts	Q3	3550.00
08JUL	KWatts	MAX	3629.00
08JUL	KWatts	STDDEV	157.64
08JUL	KWatts	LOWHISKR	3309.00
08JUL	KWatts	LOW	3070.00
08JUL	KWatts	LOW	3045.00
11JUL	KWatts	N	20.00
11JUL	KWatts	MIN	2968.00
11JUL	KWatts	Q1	3321.00

Observations with the _TYPE_ variable values “HIGH” and “LOW” contain outlier values. If you want to use a summary data set to re-create a schematic box plot, you *must* create an OUTBOX= data set in order to save the outlier data.

Syntax: BOXPLOT Procedure

The following statements are available in the BOXPLOT procedure:

```
PROC BOXPLOT options ;
    BY variables ;
    ID variables ;
    INSET keywords < / options > ;
    INSETGROUP keywords < / options > ;
    PLOT analysis-variable * group-variable < (block-variables) > < =symbol-variable > < / options > ;
```

Both the PROC BOXPLOT and PLOT statements are required. You can specify any number of PLOT statements within a single PROC BOXPLOT invocation.

PROC BOXPLOT Statement

```
PROC BOXPLOT options ;
```

The PROC BOXPLOT statement invokes the BOXPLOT procedure. Table 32.1 summarizes the *options* available in the PROC BOXPLOT statement.

Table 32.1 PROC BOXPLOT Statement Options

Option	Description
ANNOTATE=	Enhances traditional graphics box plots
BOX=	Names an input data set that contains group summary statistics and outlier values
DATA=	Names an input data set that contains raw data to be analyzed
GOUT=	Specifies the SAS catalog in which to save traditional graphics output
HISTORY=	Names an input data set that contains group summary statistics

The following *options* can appear in the PROC BOXPLOT statement.

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an ANNOTATE= type data set, as described in *SAS/GRAPH: Reference*, which enhances traditional graphics box plots requested in subsequent PLOT statements. **NOTE:** The ANNOTATE= option is ignored when ODS Graphics is enabled.

BOX=*SAS-data-set*

names an input data set that contains group summary statistics and outlier values. Typically, this data set is created as an OUTBOX= data set in a previous run of PROC BOXPLOT. Each group summary statistic or outlier value is recorded in a separate observation in a BOX= data set, so there are multiple observations per group. You cannot use a BOX= data set together with a DATA= or HISTORY= data set. If you do not specify one of these input data sets, the procedure uses the most recently created SAS data set as a DATA= data set.

DATA=*SAS-data-set*

names an input data set that contains raw data to be analyzed. You cannot use a DATA= data set together with a BOX= or HISTORY= data set. If you do not specify one of these input data sets, the procedure uses the most recently created SAS data set as a DATA= data set.

GOUT=*< libref. >output catalog*

specifies the SAS catalog in which to save traditional graphics output that is produced by the BOXPLOT procedure. If you omit the libref, PROC BOXPLOT looks for the catalog in the temporary library called WORK and creates the catalog if it does not exist. **NOTE:** The GOUT= option is ignored when ODS Graphics is enabled.

HISTORY=*SAS-data-set*

HIST=*SAS-data-set*

names an input data set that contains group summary statistics. Typically, this data set is created as an OUTHISTORY= data set in a previous run of PROC BOXPLOT, but it can also be created by using a SAS summarization procedure such as the MEANS procedure. The HISTORY= data set can contain only one observation for each value of the group variable. You cannot use a HISTORY= data set with a DATA= or BOX= data set. If you do not specify one of these three input data sets, PROC BOXPLOT uses the most recently created data set as a DATA= data set.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC BOXPLOT to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the BOXPLOT procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the “Grouping Data” section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

ID Statement

ID *variables* **;**

The ID statement specifies variables used to identify observations. The ID variables must be variables in the input data set.

If you specify the keyword SCHEMATICID or SCHEMATICIDFAR with the BOXSTYLE= option, the value of an ID variable is used to label each extreme observation. When you specify a BOX= data set, the label values come from the variable _ID_, if it is present in the data set. When you specify a DATA= or HISTORY= input data set, or a BOX= data set that does not contain the variable _ID_, the labels come from the first variable listed in the ID statement. If ID statement is specified, the outliers are not labeled.

INSET Statement

INSET *keywords* < / *options* > **;**

A PLOT statement in the BOXPLOT procedure can be followed by a series of INSET and INSETGROUP statements. Each INSET statement in that series produces one inset in the box plot produced by the preceding PLOT statement. If the box plot occupies multiple panels, the inset appears on each panel.

The data requested by using the *keywords* are displayed in the order in which they are specified. Summary statistics that are requested with an INSET statement are calculated by using the observations in all groups.

keywords identify summary statistics or other data to be displayed in the inset. By default, inset statistics are identified with appropriate labels, and numeric values are printed using appropriate formats. However, you can provide customized labels and formats. You provide the customized label by specifying the *keyword* for that statistic followed by an equal sign (=) and the label in quotes. Labels can have up to 24 characters. You provide the numeric format in parentheses after the *keyword*. Note that if you specify both a label and a format for a statistic, the label must appear before the format.

The available *keywords* are listed in Table 32.2.

options control the appearance of the inset. Most of these *options* apply only to traditional graphics and are ignored when ODS Graphics is enabled. Table 32.3 summarizes the *options* available in the INSET statement. It also lists *options* and identifies those that are valid when ODS Graphics is enabled. Complete descriptions for each *option* follow.

Table 32.2 INSET Statement Keywords

Keyword	Description
DATA=	(Label, value) pairs from *SAS-data-set*
MEAN	Mean of all observations
MIN	Minimum observed value
MAX	Maximum observed value
NMIN	Minimum group size
NMAX	Maximum group size
NOBS	Number of observations in box plot
STDDEV	Pooled standard deviation

The DATA= *keyword* specifies a SAS data set that contains (label, value) pairs to be displayed in an inset. The data set must contain the variables _LABEL_ and _VALUE_. _LABEL_ is a character variable of up to 24 characters whose values provide labels for inset entries. _VALUE_ can be character or numeric, and provides values displayed in the inset. The label and value from each observation in the DATA= data set occupy one line in the inset.

The *pooled standard deviation* requested with the STDDEV keyword is defined as

$$s_p = \sqrt{\frac{\sum_{i=1}^{N} s_i^2 (n_i - 1)}{\sum_{i=1}^{N} (n_i - 1)}}$$

where N is the number of groups, n_i is the size of the ith group, and s_i^2 is the variance of the ith group.

Table 32.3 INSET Statement Options

Option	Description	ODS Graphics
CFILL=	Specifies color of inset background	
CFILLH=	Specifies color of inset header background	
CFRAME=	Specifies color of inset frame	
CHEADER=	Specifies color of inset header text	
CSHADOW=	Specifies color of inset drop shadow	
CTEXT=	Specifies color of inset text	
DATA	Specifies data units for POSITION=(x, y) coordinates	
FONT=	Specifies font of inset text	
FORMAT=	Specifies format of values in inset	✓
HEADER=	Specifies inset header text	✓
HEIGHT=	Specifies height of inset and header text	
NOFRAME	Suppresses frame around inset	✓
POSITION=	Specifies position of inset	✓
REFPOINT=	Specifies reference point of inset positioned with POSITION=(x, y) coordinates	

Following are descriptions of the *options* that you can specify in the INSET statement after a slash (/). Only those *options* marked with † are applicable when ODS Graphics is enabled.

CFILL=*color* | **BLANK**

specifies the color of the inset background (including the header background if you do not specify the CFILLH= option).

If you do not specify the CFILL= option, then by default the background is empty. This means that items that overlap the inset (such as box-and-whiskers plots or reference lines) show through the inset. If you specify any value for the CFILL= option, then overlapping items no longer show through the inset. Specify CFILL=BLANK to leave the background uncolored and also to prevent items from showing through the inset.

CFILLH=*color*

specifies the color of the header background. By default, if you do not specify a CFILLH= color, the CFILL= color is used.

CFRAME=*color*

specifies the color of the frame around the inset. By default, the frame is the same color as the axis of the plot.

CHEADER=*color*

specifies the color of the header text. By default, if you do not specify a CHEADER= color, the INSET statement CTEXT= color is used.

CSHADOW=*color*

CS=*color*

specifies the color of the drop shadow. If you do not specify the CSHADOW= option, a drop shadow is not displayed.

CTEXT=*color*

CT=*color*

specifies the color of the text in the inset. By default, the inset text color is the same as the other text in the box plot.

DATA

specifies that data coordinates be used to position the inset with the POSITION= option. The DATA option is available only when you specify POSITION= (x, y), and it must be placed immediately after the coordinates (x, y). See the entry for the POSITION= option.

FONT=*font*

specifies the font of the text.

† FORMAT=*format*

specifies a format for all the values displayed in an inset. If you specify a format for a particular statistic, then this format overrides the format you specified with the FORMAT= option.

† HEADER='*string*'
specifies the header text. The *string* can be up to 40 characters. If you do not specify the HEADER= option, no header line appears in the inset.

HEIGHT=*value*
specifies the height of the inset and header text.

† NOFRAME
suppresses the frame drawn around the inset.

† POSITION=*position*

† POS=*position*
determines the position of the inset. The *position* can be a compass point keyword, a margin keyword, or (for traditional graphics) a pair of coordinates (x, y). You can specify coordinates in axis percent units or axis data units. For more information, see the section "Positioning Insets" on page 1476. By default, POSITION=NW, which positions the inset in the upper-left (northwest) corner of the plot.

REFPOINT=BR | BL | TR | TL

RP=BR | BL | TR | TL
specifies the reference point for an inset that is positioned by a pair of coordinates with the POSITION= option. Use the REFPOINT= option with POSITION= coordinates. The REFPOINT= option specifies which corner of the inset frame you want positioned at coordinates (x, y). The keywords BL, BR, TL, and TR represent bottom left, bottom right, top left, and top right, respectively. The default is REFPOINT=BL.

If you specify the position of the inset as a compass point or margin keyword, the REFPOINT= option is ignored.

INSETGROUP Statement

INSETGROUP *keywords* < / *options* > **;**

A PLOT statement in the BOXPLOT procedure can be followed by a series of INSET and INSETGROUP statements. Each INSETGROUP statement in that series displays statistics associated with individual groups in the box plot produced by the preceding PLOT statement. **NOTE:** The INSETGROUP statement is ignored when you specify the HORIZONTAL option with ODS Graphics enabled. No more than two INSETGROUP statements can be associated with a given PLOT statement: one that displays group statistics above the box plot and one that displays group statistics below it. The data requested by using the *keywords* are displayed in the order in which they are specified.

keywords
identify summary statistics to be displayed in the insets. By default, inset statistics are identified with appropriate labels, and numeric values are printed using appropriate formats. However, you can provide customized labels and formats. You provide the customized label by specifying the *keyword* for that statistic followed by an equal sign (=) and the label in quotes. Labels can have up to 24 characters. You provide the numeric format in parentheses after the *keyword*. Note that if you specify both a label and a format for a statistic, the label must appear before the format. The *keywords* are listed in Table 32.4.

options control the appearance of the insets. Table 32.5 lists all the *options* in the INSETGROUP statement. Complete descriptions for each *option* follow.

Table 32.4 INSETGROUP Statement Keywords

Keyword	Description
MEAN	Group mean
MIN	Group minimum value or low whisker value
MAX	Group maximum value or high whisker value
N	Number of observations in group
NHIGH	Number of outliers above upper fence
NLOW	Number of outliers below lower fence
NOUT	Total number of outliers in group
Q1	First quartile of group values
Q2	Second quartile of group values
Q3	Third quartile of group values
RANGE	Range of group values
STDDEV	Group standard deviation

NOTE: The NHIGH, NLOW, and NOUT keywords are not supported when ODS Graphics is enabled,

Table 32.5 summarizes the *options* available in the INSETGROUP statement. All of these *options* apply to traditional graphics only. They are ignored when ODS Graphics is enabled.

Table 32.5 INSETGROUP Statement Options

Option	Description
CFILL=	Specifies color of inset background
CFILLH=	Specifies color of inset header background
CFRAME=	Specifies color of inset frame
CHEADER=	Specifies color of inset header text
CTEXT=	Specifies color of inset text
FONT=	Specifies font of inset text
FORMAT=	Specifies format of values in inset
HEADER=	Specifies inset header text
HEIGHT=	Specifies height of inset and header text
NOFRAME	Suppresses frame around inset
POSITION=	Specifies position of inset

Following are descriptions of the *options* that you can specify in the INSETGROUP statement after a slash (/).

CFILL=*color*
specifies the color of the inset background (including the header background if you do not specify the CFILLH= option). If you do not specify the CFILL= option, then by default the background is empty.

CFILLH=*color*
specifies the color of the header background. By default, if you do not specify a CFILLH= color, the CFILL= color is used.

CFRAME=*color*
specifies the color of the frame around the inset. By default, the frame is the same color as the axis of the plot.

CHEADER=*color*
specifies the color of the header text. By default, if you do not specify a CHEADER= color, the CTEXT= color is used.

CTEXT=*color*

CT=*color*
specifies the color of the inset text. By default, the inset text color is the same as the other text in the plot.

FONT=*font*
specifies the font of the inset text. By default, the font is SIMPLEX.

FORMAT=*format*
specifies a format for all the values displayed in an inset. If you specify a format for a particular statistic, then this format overrides the format you specified with the FORMAT= option.

HEADER='*string*'
specifies the header text. The *string* can be up to 40 characters. If you do not specify the HEADER= option, no header line appears in the inset.

HEIGHT=*value*
specifies the height of the inset and header text.

NOFRAME
suppresses the frame drawn around the inset.

POSITION=*position*

POS=*position*
determines the position of the inset. Valid positions are TOP, TOPOFF, AXIS, and BOTTOM. By default, POSITION=TOP.

Position Keyword	Description
TOP	Top of plot, immediately above axis frame
TOPOFF	Top of plot, offset from axis frame
AXIS	Bottom of plot, immediately above horizontal axis
BOTTOM	Bottom of plot, below horizontal axis label

PLOT Statement

PLOT (*analysis-variables*)∗ *group-variable* < (*block-variables*) > < =*symbol-variable* >
< / *options* > ;

You can specify multiple PLOT statements after the PROC BOXPLOT statement. The components of the PLOT statement are as follows:

analysis-variables identify one or more variables to be analyzed. An analysis variable is required. If you specify more than one analysis variable, enclose the list in parentheses. For example, the following statements request distinct box plots for the variables Weight, Length, and Width:

```
proc boxplot data=Summary;
   plot (Weight Length Width)*Day;
run;
```

group-variable specifies the variable that identifies groups in the data. The group variable is required. In the preceding PLOT statement, Day is the group variable.

block-variables specify optional variables that group the data into blocks of consecutive groups. These blocks are labeled in a legend, and each block variable provides one level of labels in the legend.

symbol-variable specifies an optional variable whose levels (unique values) determine the symbol marker used to plot the means. Distinct symbol markers are displayed for points that correspond to the various levels of the symbol variable. You can specify the symbol markers with SYMBOL*n* statements (see *SAS/GRAPH: Reference* for complete details).

options enhance the appearance of the box plot, request additional analyses, save results in data sets, and so on. Complete descriptions of each option follow.

PLOT Statement Options

Many PLOT statement *options* apply only to traditional graphics and are ignored when ODS Graphics is enabled. Table 32.6 summarizes the *options* available in the PLOT statement. It also lists *options* by function and indicates which are applicable with ODS Graphics.

Table 32.6 PLOT Statement Options

Option	Description	ODS Graphics
Options for Controlling Box Appearance		
BOXCONNECT=	Connects features of adjacent box-and-whiskers plots with line segments	✓
BOXSTYLE=	Specifies style of box-and-whiskers plots	✓
BOXWIDTH=	Specifies width of box-and-whiskers plots	
BOXWIDTHSCALE=	Specifies that widths of box-and-whiskers plots vary proportionately to group size	✓
CBOXES=	Specifies color for outlines of box-and-whiskers plots	

Table 32.6 *continued*

Option	Description	ODS Graphics
CBOXFILL=	Specifies fill color for interior of box-and-whiskers plots	
IDCOLOR=	Specifies outlier symbol color in schematic box-and-whiskers plots	
IDCTEXT=	Specifies outlier label color in schematic box-and-whiskers plots	
IDFONT=	Specifies outlier label font in schematic box-and-whiskers plots	
IDHEIGHT=	Specifies outlier label height in schematic box-and-whiskers plots	
IDSYMBOL=	Specifies outlier symbol in schematic box-and-whiskers plots	
IDSYMBOLHEIGHT=	Specifies outlier symbol height in schematic box-and-whiskers plots	
LBOXES=	Specifies line types for outlines of box-and-whiskers plots	
NOSERIFS	Eliminates serifs from whiskers of box-and-whiskers plots	✓
NOTCHES	Specifies that box-and-whiskers plots be notched	✓
PCTLDEF=	Specifies percentile definition used for box-and-whiskers plots	✓
WHISKERPERCENTILE=	Specifies that box whiskers be drawn to percentile values	✓
Options for Plotting and Labeling Points		
ALLLABEL=	Labels means of box-and-whiskers plots	
CLABEL=	Specifies color for labels requested with ALLLABEL= option	
CCONNECT=	Specifies color for line segments requested with BOXCONNECT= option	
LABELANGLE=	Specifies angle for labels requested with ALLLABEL= option	
SYMBOLLEGEND=	Specifies LEGEND statement for levels of symbol variable	
SYMBOLORDER=	Specifies order in which symbols are assigned for levels of symbol variable	
Reference Line Options		
CHREF=	Specifies color for lines requested by HREF= option	
CVREF=	Specifies color for lines requested by VREF= option	
FRONTREF	Draws reference lines in front of boxes	
HREF=	Requests reference lines perpendicular to horizontal axis	✓
HREFLABELS=	Specifies labels for HREF= lines	✓
HREFLABPOS=	Specifies position of HREFLABELS= labels	
LHREF=	Specifies line type for HREF= lines	
LVREF=	Specifies line type for VREF= lines	
NOBYREF	Specifies that reference line information in a data set be applied uniformly to plots created for all BY groups	✓
VREF=	Requests reference lines perpendicular to vertical axis	✓
VREFLABELS=	Specifies labels for VREF= lines	✓

Table 32.6 *continued*

Option	Description	ODS Graphics
VREFLABPOS=	Specifies position of VREFLABELS= labels	
Block Variable Options		
BLOCKLABELPOS=	Specifies position of label for block variable legend	
BLOCKLABTYPE=	Specifies text size of block variable legend	
BLOCKPOS=	Specifies vertical position of block variable legend	✓
BLOCKREF	Uses vertical reference lines to delineate blocks that are defined by block variables	✓
BLOCKREFFILL	Fills the graph walls of blocks defined by block variables with colors that are determined by the BLOCKVAR= option	✓
BLOCKREP	Repeats identical consecutive labels in block variable legend	✓
BLOCKVALUEPOS=	Specifies the alignment of block values within the block legend	✓
CBLOCKLAB=	Specifies colors for filling frames enclosing block variable labels	
CBLOCKVAR=	Specifies colors for filling background of block variable legend	
Axis and Axis Label Options		
CAXIS=	Specifies color for axis lines and tick marks	
CFRAME=	Specifies fill color for frame for plot area	
CONTINUOUS	Produces horizontal axis for continuous group variable values (traditional graphics only)	
CTEXT=	Specifies color for tick mark values and axis labels	
HAXIS=	Specifies major tick mark values for horizontal axis	
HEIGHT=	Specifies height of axis label and axis legend text	
HMINOR=	Specifies number of minor tick marks between major tick marks on horizontal axis	
HOFFSET=	Specifies length of offset at both ends of horizontal axis	
NOHLABEL	Suppresses horizontal axis label	✓
NOTICKREP	Specifies that only first occurrence of repeated, adjacent character group values be labeled on horizontal axis	
NOVANGLE	Requests vertical axis labels that are strung out vertically	
SKIPHLABELS=	Specifies thinning factor for tick mark labels on horizontal axis	
TURNHLABELS	Requests horizontal tick labels that are strung out vertically	
VAXIS=	Specifies major tick mark values for vertical axis	✓
VFORMAT=	Specifies format for vertical axis tick marks	✓
VMINOR=	Specifies number of minor tick marks between major tick marks on vertical axis	
VOFFSET=	Specifies length of offset at both ends of vertical axis	
VZERO	Forces origin to be included in vertical axis	

Table 32.6 *continued*

Option	Description	ODS Graphics
WAXIS=	Specifies width of axis lines	
Input Data Set Options		
MISSBREAK	Specifies that a missing value between identical character group values signify the start of a new group	✓
Output Data Set Options		
OUTBOX=	Produces an output data set that contains group summary statistics and outlier values	✓
OUTHISTORY=	Produces an output data set that contains group summary statistics	✓
Graphical Enhancement Options		
ANNOTATE=	Specifies annotate data set that adds features to box plot	
BWSLEGEND	Displays a legend that identifies the function of group size specified with BOXWIDTHSCALE= option	
DESCRIPTION=	Specifies string that appears in description field of PROC GREPLAY master menu for traditional graphics box plot	
FONT=	Specifies font for labels and legends on plots	
HORIZONTAL	Requests a horizontal box plot with ODS Graphics	✓
HTML=	Specifies URLs to be associated with box-and-whiskers plots	
NAME=	Specifies name that appears in name field of PROC GREPLAY master menu for traditional graphics box plot	
NLEGEND	Requests legend that displays group sizes	
OUTHIGHHTML=	Specifies URLs to be associated with high outliers on box-and-whiskers plots	
OUTLOWHTML=	Specifies URLs to be associated with low outliers on box-and-whiskers plots	
PAGENUM=	Specifies form of label used in pagination	
PAGENUMPOS=	Specifies position of page number requested with PAGENUM= option	
Grid Options		
CGRID=	Specifies color for grid requested with ENDGRID or GRID option	
ENDGRID	Adds grid after last box-and-whiskers plot	
GRID	Adds grid to box plot	✓
LENDGRID=	Specifies line type for grid requested with ENDGRID option	
LGRID=	Specifies line type for grid requested with GRID option	
WGRID=	Specifies width of grid lines	
Plot Layout Options		
INTERVAL=	Specifies natural time interval between consecutive group positions when time, date, or datetime format is associated with numeric group variable	

Table 32.6 *continued*

Option	Description	ODS Graphics
INTSTART=	Specifies first major tick mark value on horizontal axis when date, time, or datetime format is associated with numeric group variable	
MAXPANELS=	Specifies maximum number of panels used for box plot	✓
NOCHART	Suppresses creation of box plot	✓
NOFRAME	Suppresses frame for plot area	
NPANELPOS=	Specifies number of group positions per panel	✓
REPEAT	Repeats last group position on panel as first group position of next panel	✓
TOTPANELS=	Specifies number of panels to be used to display box plot	✓
Overlay Options		
CCOVERLAY=	Specifies colors for line segments that connect points on overlays	
COVERLAY=	Specifies colors for points on overlays	
LOVERLAY=	Specifies line types for line segments that connect points on overlays	
NOOVERLAYLEGEND	Suppresses overlay legend	✓
OVERLAY=	Specifies variables to be plotted on overlays	✓
OVERLAYHTML=	Specifies URLs to be associated with overlay plot points	
OVERLAYID=	Specifies labels for overlay plot points	
OVERLAYLEGLAB=	Specifies label for overlay legend	✓
OVERLAYSYM=	Specifies symbols used for overlays	
OVERLAYSYMHT=	Specifies heights for overlay symbols	
WOVERLAY=	Specifies widths for line segments that connect points on overlays	
Clipping Options		
CCLIP=	Specifies color for plot symbol for clipped points	
CLIPFACTOR=	Determines extent to which extreme values are clipped	✓
CLIPLEGEND=	Specifies text for clipping legend	✓
CLIPLEGPOS=	Specifies position of clipping legend	
CLIPSUBCHAR=	Specifies substitution character for CLIPLEGEND= text	✓
CLIPSYMBOL=	Specifies plot symbol for clipped points	
CLIPSYMBOLHT=	Specifies symbol marker height for clipped points	
COVERLAYCLIP=	Specifies color for clipped points on overlays	
OVERLAYCLIPSYM=	Specifies symbol for clipped points on overlays	
OVERLAYCLIPSYMHT=	Specifies symbol height for clipped points on overlays	
Options for Box Plots Produced Using Styles		
BLOCKVAR=	Groups block legends whose backgrounds are filled with colors from style	✓
BOXES=	Groups boxes whose outlines are drawn with colors from style	
BOXFILL=	Groups boxes that are filled with colors from style	

Table 32.6 *continued*

Option	Description	ODS Graphics
Options for ODS Graphics Output		
ODSFOOTNOTE=	Specifies a footnote displayed in ODS Graphics output	✓
ODSFOOTNOTE2=	Specifies a secondary footnote displayed in ODS Graphics output	✓
ODSTITLE=	Specifies a title displayed in ODS Graphics output	✓
ODSTITLE2=	Specifies a secondary title displayed in ODS Graphics output	✓

Following are explanations of the *options* you can specify in the PLOT statement after a slash (/). Only those *options* marked with † are applicable when ODS Graphics is enabled.

ALLLABEL=VALUE | (*variable*)
labels the point plotted for the mean of each box-and-whiskers plot with the mean (when ALLLABEL=VALUE) or with the value of the ALLLABEL=*variable* from the input data set.

ANNOTATE=*SAS-data-set*
specifies an ANNOTATE= type data set, as described in *SAS/GRAPH: Reference*.

BLOCKLABELPOS=ABOVE | LEFT
specifies the position of a block variable label in the block legend. The keyword ABOVE places the label immediately above the legend, and LEFT places the label to the left of the legend. Use the keyword LEFT with labels that are short enough to fit in the margin of the plot; otherwise, they are truncated. By default, BLOCKLABELPOS=ABOVE.

BLOCKLABTYPE=SCALED | TRUNCATED | *height*
specifies how lengthy block variable values are treated when there is insufficient space to display them in the block legend. If you specify BLOCKLABTYPE=SCALED, the values are uniformly reduced in height so that they fit. If you specify BLOCKLABTYPE=TRUNCATED, lengthy values are truncated on the right until they fit. You can also specify a text height in vertical percent screen units for the values. By default, lengthy values are not displayed. For more information, see the section "Displaying Blocks of Data" on page 1481.

† BLOCKPOS=*n*
specifies the vertical position of the legend for the values of the block variables. Values of *n* and the corresponding positions are as follows. By default, BLOCKPOS=1.

n	Legend Position
1	Top of plot, offset from axis frame
2	Top of plot, immediately above axis frame
3	Bottom of plot, immediately above horizontal axis
4	Bottom of plot, below horizontal axis label

† BLOCKREF

uses vertical reference lines to delineate blocks that are defined by the block variables. For more information about using block variables, see the section "Displaying Blocks of Data" on page 1481. **NOTE:** This option is available only when ODS Graphics is enabled.

† BLOCKREFFILL

fills the graph walls of blocks that are defined by the block variables with colors that are determined by the BLOCKVAR= option. For more information about using block variables, see the section "Displaying Blocks of Data" on page 1481. **NOTE:** This option is available only when ODS Graphics is enabled.

† BLOCKREP

specifies that block variable values for all groups be displayed. By default, only the first block variable value in any block is displayed, and repeated block variable values are not displayed.

† BLOCKVALUEPOS=CENTER | LEFT | RIGHT

specifies the alignment of block variable values in the block legend. By default, BLOCKVALUEPOS=LEFT. For more information about using block variables, see the section "Displaying Blocks of Data" on page 1481. **NOTE:** This option is available only when ODS Graphics is enabled.

† BLOCKVAR=*variable* **|** (*variable-list*)

specifies variables whose values are used to assign colors for filling the background of the legend associated with block variables. A list of BLOCKVAR= variables must be enclosed in parentheses. BLOCKVAR= variables are matched with block variables by their order in the respective variable lists. Whereas the values of a CBLOCKVAR= variable are color names, values of a BLOCKVAR= variable are used to group block legends for assigning fill colors from the ODS style. Block legends that have the same BLOCKVAR= variable value are filled with the same color.

† BOXCONNECT=MEAN | MEDIAN | MAX | MIN | Q1 | Q3

† BOXCONNECT

specifies that the points in adjacent box-and-whiskers plots representing group means, medians, maximum values, minimum values, first quartiles, or third quartiles be connected with line segments. If the BOXCONNECT option is specified without a *keyword* that identifies the points to be connected, group means are connected. By default, no points are connected.

BOXES=(*variable***)**

specifies a variable whose values are used to assign colors for the outlines of box-and-whiskers plots. Whereas the values of a CBOXES= variable are color names, values of the BOXES= variable are used to group box-and-whiskers plots for assigning outline colors from the ODS style. The outlines of box-and-whiskers plots of groups that have the same BOXES= variable value are drawn using the same color.

BOXFILL=(*variable***)**

specifies a variable whose values are used to assign fill colors for box-and-whiskers plots. Whereas the values of a CBOXFILL= variable are color names, values of the BOXFILL= variable are used to group box-and-whiskers plots for assigning fill colors from the ODS style. Box-and-whiskers plots of groups that have the same BOXFILL= variable value are filled with the same color.

† BOXSTYLE=*keyword*

specifies the style of the box-and-whiskers plots displayed. If you specify BOXSTYLE=SKELETAL, the whiskers are drawn from the edges of the box to the extreme values of the group. This plot is sometimes referred to as a skeletal box-and-whiskers plot. By default, the whiskers are drawn with serifs. You can specify the NOSERIFS option to draw the whiskers without serifs.

In the following descriptions, the terms *fence* and *far fence* refer to the distance from the first and third quartiles (25th and 75th percentiles, respectively), expressed in terms of the interquartile range (IQR). For example, the lower fence is located at $1.5 \times \mathrm{IQR}$ below the 25th percentile; the upper fence is located at $1.5 \times \mathrm{IQR}$ above the 75th percentile. Similarly, the lower far fence is located at $3 \times \mathrm{IQR}$ below the 25th percentile; the upper far fence is located at $3 \times \mathrm{IQR}$ above the 75th percentile.

If you specify BOXSTYLE=SCHEMATIC, a whisker is drawn from the upper edge of the box to the largest observed value less than or equal to the upper fence, and another is drawn from the lower edge of the box to the smallest observed value greater than or equal to the lower fence. Serifs are added to the whiskers by default. Observations outside the fences are identified with a special symbol. For traditional graphics you can specify the shape and color for this symbol with the IDSYMBOL= and IDCOLOR= options. The default symbol is a square. This type of plot corresponds to the schematic box-and-whiskers plot described in Chapter 2 of Tukey (1977). See Figure 32.8 and the discussion in the section "Styles of Box Plots" on page 1472 for more information.

If you specify BOXSTYLE=SCHEMATICID, a schematic box-and-whiskers plot is displayed in which an ID variable value is used to label the symbol that marks each observation outside the upper and lower fences. A BOX= data set can contain a variable named _ID_ that is used as the ID variable. Otherwise, the first variable listed in the ID statement provides the labels.

If you specify BOXSTYLE=SCHEMATICIDFAR, a schematic box-and-whiskers plot is displayed in which the value of the ID variable is used to label the symbol that marks each observation outside the lower and upper far fences. Observations between the fences and the far fences are identified with a symbol but are not labeled with the ID variable.

Figure 32.6 illustrates the elements of a skeletal box-and-whiskers plot.

Figure 32.6 Skeletal Box-and-Whiskers Plot

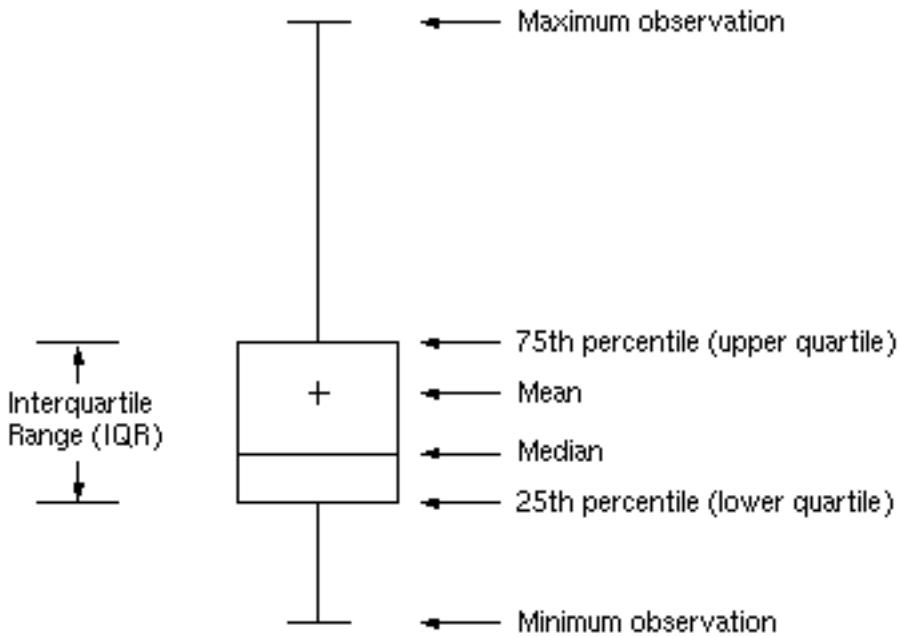

The skeletal style of the box-and-whiskers plot shown in Figure 32.6 is the default.

BOXWIDTH=*value*

specifies the width (in horizontal percent screen units) of the box-and-whiskers plots.

† BOXWIDTHSCALE=*value*

specifies that the widths of the box-and-whisker plots are to vary according to a particular function of the group size *n*. The function, $f(n)$, is determined by the specified *value* (≥ 0).

If you specify a positive *value*, $f(n) = n^{\text{value}}$. In particular, if you specify BOXWIDTHSCALE=1, $f(n) = n$. If you specify BOXWIDTHSCALE=0.5, $f(n) = \sqrt{n}$, as described by McGill, Tukey, and Larsen (1978).

If you specify BOXWIDTHSCALE=0, $f(n) = \log(n)$.

The box widths vary between minimum ($w_{\min}$) and maximum ($w_{\max}$) widths that are determined by the output destination. The width of the *i*th box is

$$w_i = w_{\min} + (w_{\max} - w_{\min}) \frac{f(n_i) - f(n_{\min})}{f(n_{\max}) - f(n_{\min})}$$

where $n_{\min}$ is the minimum group size and $n_{\max}$ is the maximum group size.

By default, the box widths are constant.

See Example 32.5 for an illustration of the BOXWIDTHSCALE= option.

You can specify the BWSLEGEND option to display a legend that identifies $f(n)$.

BWSLEGEND

displays a legend that identifies the function of group size *n* that is specified with the BOXWIDTHSCALE= option. No legend is displayed if all group sizes are equal. The BWSLEGEND option is not applicable unless you also specify the BOXWIDTHSCALE= option.

CAXIS=*color*

CAXES=*color*

CA=*color*

specifies the color for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement.

CBLOCKLAB=*color* | (*color-list*)

specifies fill colors for the frames that enclose the block variable labels in a block legend. By default, these areas are not filled. Colors in the CBLOCKLAB= list are matched with block variables in the order in which they appear in the PLOT statement.

CBLOCKVAR=*variable* | (*variable-list*)

specifies variables whose values are colors for filling the background of the legend associated with block variables. CBLOCKVAR= variables are matched with block variables by their order in the respective variable lists. Each CBLOCKVAR= variable must be a character variable of no more than eight characters in the input data set, and its values must be valid SAS/GRAPH color names (see *SAS/GRAPH: Reference* for complete details). A list of CBLOCKVAR= variables must be enclosed in parentheses.

The procedure matches the CBLOCKVAR= variables with block variables in the order specified. That is, each block legend is filled with the color value of the CBLOCKVAR= variable of the first observation in each block. In general, values of the *i*th CBLOCKVAR= variable are used to fill the block of the legend that corresponds to the *i*th block variable.

By default, fill colors are not used for the block variable legend. The CBLOCKVAR= option is available only when block variables are used in the PLOT statement.

CBOXES=*color* | (*variable*)

specifies the colors for the outlines of the box-and-whiskers plots created with the PLOT statement. You can use one of the following approaches:

- You can specify CBOXES=*color* to provide a single outline color for all the box-and-whiskers plots.
- You can specify CBOXES=(*variable*) to provide a distinct outline color for each box-and-whiskers plot as the value of the variable. The variable must be a character variable of up to eight characters in the input data set, and its values must be valid SAS/GRAPH color names (see *SAS/GRAPH: Reference* for complete details). The outline color of the plot displayed for a particular group is the value of the variable in the observations corresponding to this group. Note that if there are multiple observations per group in the input data set, the values of the variable should be identical for all the observations in a given group.

CBOXFILL=*color* | **(***variable***)**
specifies the interior fill colors for the box-and-whiskers plots. You can use one of the following approaches:

- You can specify CBOXFILL=*color* to provide a single color for all of the box-and-whiskers plots.
- You can specify CBOXFILL=(*variable*) to provide a distinct color for each box-and-whiskers plot as the value of the variable. The variable must be a character variable of up to eight characters in the input data set, and its values must be valid SAS/GRAPH color names (or the value EMPTY, which you can use to suppress color filling). See *SAS/GRAPH: Reference* for complete details. The interior color of the box displayed for a particular group is the value of the variable in the observations that correspond to this group. Note that if there are multiple observations per group in the input data set, the values of the variable should be identical for all the observations in a given group.

By default, the interiors are not filled.

CCLIP=*color*
specifies a color for the plotting symbol that is specified with the CLIPSYMBOL= option to mark clipped values. The default color is the color specified in the COLOR= option in the SYMBOL1 statement.

CCONNECT=*color*
specifies the color for line segments that connect points on the plot. The default color is the color specified in the COLOR= option in the SYMBOL1 statement. This option is not applicable unless you also specify the BOXCONNECT= option.

CCOVERLAY=(*color-list***)**
specifies the colors for line segments connecting points on overlay plots. Colors in the CCOVERLAY= list are matched with variables in the corresponding positions in the OVERLAY= list. By default, points are connected by line segments of the same color as the plotted points. You can specify the value NONE to suppress the line segments that connect points of an overlay plot.

CFRAME=*color*
specifies the color for filling the rectangle enclosed by the axes and the frame. By default, this area is not filled. The CFRAME= option cannot be used in conjunction with the NOFRAME option.

CGRID=*color*
specifies the color for the grid requested by the ENDGRID or GRID option. By default, the grid is the same color as the axes.

CHREF=*color*
specifies the color for the lines requested by the HREF= option.

CLABEL=*color*
specifies the color for labels produced by the ALLLABEL= option. The default color is the CTEXT= color.

† CLIPFACTOR=*factor*

requests clipping of extreme values on the box plot. The *factor* that you specify determines the extent to which these values are clipped, and it must be greater than 1.

For examples of the CLIPFACTOR= option, see Figure 32.17 and Figure 32.18. Related clipping options are CCLIP=, CLIPLEGEND=, CLIPLEGPOS=, CLIPSUBCHAR=, and CLIPSYMBOL=.

† CLIPLEGEND='*label*'

specifies the *label* for the legend that indicates the number of clipped boxes when the CLIPFACTOR= option is used. The *label* must be no more than 16 characters and must be enclosed in quotes. For an example, see Figure 32.18.

CLIPLEGPOS= TOP | BOTTOM

specifies the position for the legend that indicates the number of clipped boxes when the CLIPFACTOR= option is used. The keyword TOP or BOTTOM positions the legend at the top or bottom of the chart, respectively. Do not specify CLIPLEGPOS=TOP together with the BLOCKPOS=1 or BLOCKPOS=2 option. By default, CLIPLEGPOS=BOTTOM.

† CLIPSUBCHAR='*character*'

specifies a substitution character (such as '#') for the label provided with the CLIPLEGEND= option. The substitution character is replaced with the number of boxes that are clipped. For example, suppose that the following statements produce a chart in which three boxes are clipped:

```
proc boxplot data=Pistons;
   plot Diameter*Hour /
      clipfactor  = 1.5
      cliplegend  = 'Boxes clipped=#'
      clipsubchar = '#' ;
run;
```

Then the clipping legend displayed on the chart will be "Boxes clipped=3".

CLIPSYMBOL=*symbol*

specifies a plot symbol used to identify clipped points on the chart and in the legend when the CLIPFACTOR= option is used. You should use this option in conjunction with the CLIPFACTOR= option. The default *symbol* is CLIPSYMBOL=SQUARE.

CLIPSYMBOLHT=*value*

specifies the height for the symbol marker used to identify clipped points on the chart when the CLIPFACTOR= option is used. The default is the height specified with the H= option in the SYMBOL statement.

For general information about clipping options, see the section "Clipping Extreme Values" on page 1483.

CONTINUOUS

specifies that numeric group variable values be treated as continuous values. By default, the values of a numeric group variable are considered discrete values unless the HAXIS= option is specified. **NOTE:** The CONTINUOUS option is not supported for ODS Graphics output. For more information, see the discussion in the section "Continuous Group Variables" on page 1474.

COVERLAY=(*color-list*)
specifies the colors used to plot overlay variables. Colors in the COVERLAY= list are matched with variables in the corresponding positions in the OVERLAY= list.

COVERLAYCLIP=*color*
specifies the color used to plot clipped values on overlay plots when the CLIPFACTOR= option is used.

CTEXT=*color*
specifies the color for tick mark values and axis labels. The default color is the color specified in the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*
specifies the color for the lines requested by the VREF= option.

DESCRIPTION=‘*string*’
DES=‘*string*’
specifies a description of a box plot produced with traditional graphics. The description appears in the PROC GREPLAY master menu and can be no longer than 256 characters. The default description is the analysis variable name.

ENDGRID
adds a grid to the rightmost portion of the plot, beginning with the first labeled major tick mark position that follows the last box-and-whiskers plot. You can use the HAXIS= option to force space to be added to the horizontal axis.

FONT=*font*
specifies a font for labels and legends. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the GOPTIONS statement. See *SAS/GRAPH: Reference* for more information about the GOPTIONS statement.

FRONTREF
draws reference lines specified with the HREF= and VREF= options in front of box-and-whiskers plots. By default, reference lines are drawn behind the box-and-whiskers plots and can be obscured by filled boxes.

† GRID
adds a grid to the box plot. Grid lines are horizontal lines positioned at labeled major tick marks, and they cover the length and height of the plotting area.

HAXIS=*value-list* | AXIS*n*
specifies tick mark values for the horizontal (group) axis. If the group variable is numeric, the values must be numeric and equally spaced. If the group variable is character, values must be quoted strings of up to 16 characters. Optionally, you can specify an axis name defined in a previous AXIS statement. See *SAS/GRAPH: Reference* for more information about the AXIS statement.

If you are producing traditional graphics, specifying the HAXIS= option with a numeric group variable causes the group variable values to be treated as continuous values. For more information, see the description of the CONTINUOUS option and the discussion in the section “Continuous Group Variables” on page 1474. Numeric values can be given in an explicit or implicit list. If a date, time, or datetime format is associated with a numeric group variable, SAS datetime literals can be used. Examples of HAXIS= lists follow:

- haxis=0 2 4 6 8 10
- haxis=0 to 10 by 2
- haxis='LT12A' 'LT12B' 'LT12C' 'LT15A' 'LT15B' 'LT15C'
- haxis='20MAY88'D to '20AUG88'D by 7
- haxis='01JAN88'D to '31DEC88'D by 30

If the group variable is numeric, the HAXIS= list must span the group variable values. If the group variable is character, the HAXIS= list must include all of the group variable values. You can add group positions to the box plot by specifying HAXIS= values that are not group variable values.

If you specify a large number of HAXIS= values, some of these can be thinned to avoid collisions between tick mark labels. To avoid thinning, use one of the following methods.

- Shorten values of the group variable by eliminating redundant characters. For example, if your group variable has values LOT1, LOT2, LOT3, and so on, you can use the SUBSTR function in a DATA step to eliminate LOT from each value, and you can modify the horizontal axis label to indicate that the values refer to lots.
- Use the TURNHLABELS option to turn the labels vertically.
- Use the NPANELPOS= option to force fewer group positions per panel.

HEIGHT=*value*

specifies the height (in vertical screen percent units) of the text for axis labels and legends. This value takes precedence over the HTEXT= value specified in the GOPTIONS statement. This option is recommended for use with fonts specified with the FONT= option or with the FTEXT= option in the GOPTIONS statement. See *SAS/GRAPH: Reference* for complete information about the GOPTIONS statement.

HMINOR=*n*

HM=*n*

specifies the number of minor tick marks between major tick marks on the horizontal axis. Minor tick marks are not labeled. The default is HMINOR=0.

HOFFSET=*value*

specifies the length (in percent screen units) of the offset at both ends of the horizontal axis. You can eliminate the offset by specifying HOFFSET=0.

† HORIZONTAL

produces a horizontal box plot, with group variable values on the vertical axis and analysis variable values on the horizontal axis. The HORIZONTAL option is supported only with ODS Graphics. **NOTE:** When you specify the HORIZONTAL option, any INSETGROUP statements associated with the PLOT statement are ignored.

† HREF=*value-list*

HREF=*SAS-data-set*

draws reference lines perpendicular to the horizontal (group) axis on the box plot. You can use this option in the following ways:

- You can specify the values for the lines with an HREF= list. If the group variable is numeric, the values must be numeric. If the group variable is character, the values must be quoted strings of up to 16 characters. If the group variable is formatted, the values must be given as internal values. Examples of HREF= values follow:

```
href=5
href=5 10 15 20 25 30
href='Shift 1' 'Shift 2' 'Shift 3'
```

- You can specify reference line values as the values of a variable named _REF_ in an HREF= data set. The type and length of _REF_ must match those of the group variable specified in the PLOT statement. Optionally, you can provide labels for the lines as values of a variable named _REFLAB_, which must be a character variable of up to 16 characters. If you want distinct reference lines to be displayed in plots for different analysis variables specified in the PLOT statement, you must include a character variable named _VAR_, whose values are the analysis variable names. If you do not include the variable _VAR_, all of the lines are displayed in all of the plots. Each observation in an HREF= data set corresponds to a reference line. If BY variables are used in the input data set, the same BY variable structure must be used in the reference line data set unless you specify the NOBYREF option.

Unless the CONTINUOUS or HAXIS= option is specified, numeric group variable values are treated as discrete values, and only HREF= values that match these discrete values are valid. Other values are ignored.

† HREFLABELS='*label1*' **...**'*labeln*'

† HREFLABEL='*label1*' **...**'*labeln*'

† HREFLAB='*label1*' **...**'*labeln*'

specifies labels for the reference lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*

specifies the vertical position of the HREFLABELS= label, as described in the following table. By default, n=2.

HREFLABPOS=	Label Position
1	Along top of plot area
2	Staggered from top to bottom of plot area
3	Along bottom of plot area
4	Staggered from bottom to top of plot area

HTML=*variable*

specifies uniform resource locators (URLs) as values of the specified character variable (or formatted values of a numeric variable). These URLs are associated with box-and-whiskers plots when graphics output is directed into HTML. The value of the HTML= variable should be the same for each observation with a given value of the group variable.

IDCOLOR=*color*
specifies the color of the symbol marker used to identify outliers in schematic box-and-whiskers plots (that is, when you specify the keyword SCHEMATIC, SCHEMATICID, or SCHEMATICIDFAR with the BOXSTYLE= option). The default color is the color specified with the CBOXES= option.

IDCTEXT=*color*
specifies the color for the text used to label outliers when you specify the keyword SCHEMATICID or SCHEMATICIDFAR with the BOXSTYLE= option. The default value is the color specified with the CTEXT= option.

IDFONT=*font*
specifies the font for the text used to label outliers when you specify the keyword SCHEMATICID or SCHEMATICIDFAR with the BOXSTYLE= option. The default font is SIMPLEX.

IDHEIGHT=*value*
specifies the height for the text used to label outliers when you specify the keyword SCHEMATICID or SCHEMATICIDFAR with the BOXSTYLE= option. The default value is the height specified with the HTEXT= option in the GOPTIONS statement. See *SAS/GRAPH: Reference* for complete information about the GOPTIONS statement.

IDSYMBOL=*symbol*
specifies the symbol marker used to identify outliers in schematic box plots. The default symbol is SQUARE.

IDSYMBOLHEIGHT=*value*
specifies the height of the symbol marker used to identify outliers in schematic box plots.

INTERVAL=DAY | DTDAY | HOUR | MINUTE | MONTH | QTR | SECOND
specifies the natural time interval between consecutive group positions when a time, date, or datetime format is associated with a numeric group variable. By default, the INTERVAL= option uses the number of group positions per panel (screen or page) that you specify with the NPANELPOS= option. The default time interval *keywords* for various time formats are shown in the following table.

Format	Default Keyword	Format	Default Keyword
DATE	**DAY**	MONYY	**MONTH**
DATETIME	**DTDAY**	TIME	**SECOND**
DDMMYY	**DAY**	TOD	**SECOND**
HHMM	**HOUR**	WEEKDATE	**DAY**
HOUR	**HOUR**	WORDDATE	**DAY**
MMDDYY	**DAY**	YYMMDD	**DAY**
MMSS	**MINUTE**	YYQ	**QTR**

You can use the INTERVAL= option to modify the effect of the NPANELPOS= option, which specifies the number of group positions per panel. The INTERVAL= option enables you to match the scale of the horizontal axis to the scale of the group variable without having to associate a different format with the group variable.

For example, suppose that your formatted group values span an overall time interval of 100 days and a DATETIME format is associated with the group variable. Since the default interval for the DATETIME format is DTDAY and since NPANELPOS=25 by default, the plot is displayed with four panels.

Now, suppose that your data span an overall time interval of 100 hours and a DATETIME format is associated with the group variable. The plot for these data is created in a single panel, but the data occupy only a small fraction of the plot since the scale of the data (hours) does not match that of the horizontal axis (days). If you specify INTERVAL=HOUR, the horizontal axis is scaled for 25 hours, matching the scale of the data, and the plot is displayed with four panels.

You should use the INTERVAL= option only in conjunction with the CONTINUOUS or HAXIS= option, which produces a horizontal axis of continuous group variable values. For more information, see the descriptions of the CONTINUOUS and HAXIS= options, and the discussion in the section "Continuous Group Variables" on page 1474.

INTSTART=*value*

specifies the starting value for a numeric horizontal axis when a date, time, or datetime format is associated with the group variable. If the value specified is greater than the first group variable value, this option has no effect.

LABELANGLE=*angle*

specifies the angle at which labels requested with the ALLLABEL= option are drawn. A positive angle rotates the labels counterclockwise; a negative angle rotates them clockwise. By default, labels are oriented horizontally.

LBOXES=*linetype* **|** **(***variable***)**

specifies the line types for the outlines of the box-and-whiskers plots. You can use one of the following approaches:

- You can specify LBOXES=*linetype* to provide a single linetype for all of the box-and-whiskers plots.
- You can specify LBOXES=(*variable*) to provide a distinct line type for each box-and-whiskers plot. The variable must be a numeric variable in the input data set, and its values must be valid SAS/GRAPH linetype values (numbers that range from 1 to 46). The line type for the plot displayed for a particular group is the value of the variable in the observations that correspond to this group. Note that if there are multiple observations per group in the input data set, the values of the variable should be identical for all of the observations in a given group.

The default value is 1, which produces solid lines. See the description of the SYMBOL statement in *SAS/GRAPH: Reference* for more information about valid linetypes.

LENDGRID=*linetype*

specifies the line type for the grid requested with the ENDGRID option. The default value is 1, which produces a solid line. If you use the LENDGRID= option, you do not need to specify the ENDGRID option. See the description of the SYMBOL statement in *SAS/GRAPH: Reference* for more information about valid linetypes.

LGRID=*linetype*

specifies the line type for the grid requested with the GRID option. The default value is 1, which produces a solid line. If you use the LGRID= option, you do not need to specify the GRID option. See the description of the SYMBOL statement in *SAS/GRAPH: Reference* for more information about valid linetypes.

LHREF=*linetype*

LH=*linetype*

specifies the line type for reference lines requested with the HREF= option. The default value is 2, which produces a dashed line. See the description of the SYMBOL statement in *SAS/GRAPH: Reference* for more information about valid linetypes.

LOVERLAY=(*linetypes***)**

specifies line types for the line segments that connect points on overlay plots. Line types in the LOVERLAY= list are matched with variables in the corresponding positions in the OVERLAY= list.

LVREF=*linetype*

LV=*linetype*

specifies the line type for reference lines requested by the VREF= option. The default value is 2, which produces a dashed line. See the description of the SYMBOL statement in *SAS/GRAPH: Reference* for more information about valid linetypes.

† MAXPANELS=*n*

specifies the maximum number of panels used to display a box plot. By default, $n = 20$.

† MISSBREAK

determines how groups are formed when observations are read from a DATA= data set and a character group variable is provided. When you specify the MISSBREAK option, observations with missing values of the group variable are not processed. Furthermore, the next observation with a nonmissing value of the group variable is treated as the beginning observation of a new group even if this value is identical to the most recent nonmissing group value. In other words, by specifying the option MISSBREAK and by inserting an observation with a missing group variable value into a group of consecutive observations with the same group variable value, you can split the group into two distinct groups of observations.

By default (that is, when you omit the MISSBREAK option), observations with missing values of the group variable are not processed, and all remaining observations with the same consecutive value of the group variable are treated as a single group.

NAME='*string*'

specifies a name, not more than eight characters long, for a traditional graphics box plot. The name appears in the PROC GREPLAY master menu.

NLEGEND

requests a legend that displays group sizes. If the size is the same for each group, that number is displayed. Otherwise, the minimum and maximum group sizes are displayed.

† NOBYREF

specifies that the reference line information in an HREF= or VREF= data set be applied uniformly to box plots created for all the BY groups in the input data set. If you specify the NOBYREF option, you do not need to provide BY variables in the reference line data set. By default, you must provide BY variables.

† NOCHART

suppresses the creation of the box plot. You typically specify the NOCHART option when you are using the procedure to compute group summary statistics and save them in an output data set.

NOFRAME

suppresses the default frame drawn around the plot.

† NOHLABEL

suppresses the label for the horizontal (group) axis. Use the NOHLABEL option when the meaning of the axis is evident from the tick mark labels, such as when a date format is associated with the group variable.

† NOOVERLAYLEGEND

suppresses the legend for overlay plots that is displayed by default when the OVERLAY= option is specified.

† NOSERIFS

eliminates serifs from the whiskers of box-and-whiskers plots.

† NOTCHES

specifies that box-and-whiskers plots be notched. The endpoints of the notches are located at the median plus and minus $1.58(\mathrm{IQR}/\sqrt{n})$, where IQR is the interquartile range and n is the group size. The medians (central lines) of two box-and-whiskers plots are significantly different at approximately the 0.95 confidence level if the corresponding notches do not overlap.

See McGill, Tukey, and Larsen (1978) for more information. Figure 32.7 illustrates the NOTCHES option. Notice the folding effect at the bottom, which happens when the endpoint of a notch is beyond its corresponding quartile. This situation typically occurs when the group size is small.

Figure 32.7 Box Plot: The NOTCHES Option

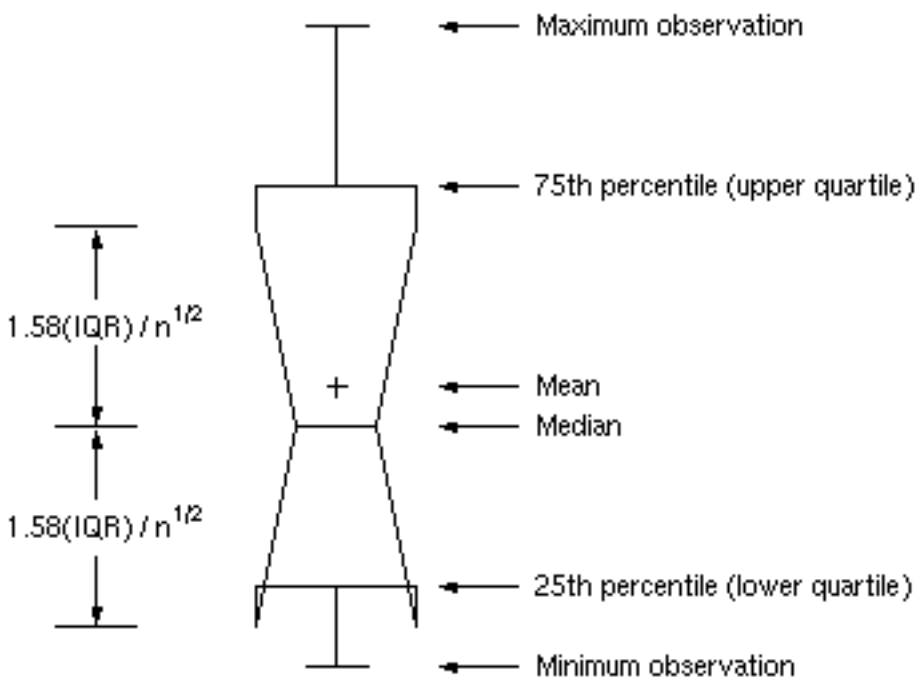

NOTICKREP

applies to character-valued group variables and specifies that only the first occurrence of repeated, adjacent group values be labeled on the horizontal axis.

NOVANGLE

requests that the vertical axis label be strung out vertically.

† NPANELPOS=*n*

NPANEL=*n*

specifies the number of group positions per panel. You typically specify the NPANELPOS= option to display more box-and-whiskers plots on a panel than the default number, which is $n = 25$.

You can specify a positive or negative number for *n*. The absolute value of *n* must be at least 5. If *n* is positive, the number of positions is adjusted so that it is approximately equal to *n* and so that all panels display approximately the same number of group positions. If *n* is negative, no balancing is done, and each panel (except possibly the last) displays approximately $|n|$ positions. In this case, the approximation is due only to axis scaling.

You can use the INTERVAL= option to change the effect of the NPANELPOS= option when a date or time format is associated with the group variable. The INTERVAL= option enables you to match the scale of the horizontal axis to the scale of the group variable without having to associate a different format with the group variable.

† ODSFOOTNOTE=FOOTNOTE | FOOTNOTE1 | '*string***'**

adds a footnote to ODS Graphics output. If you specify the FOOTNOTE (or FOOTNOTE1) keyword, the value of SAS FOOTNOTE statement is used the as the graph footnote. If you specify a quoted string, that is used as the footnote. The quoted string can contain any of the following escaped characters, which are replaced with the appropriate values from the analysis:

\n	analysis variable name
\l	analysis variable label (or name if the analysis variable has no label)
\x	group variable name
\s	group variable label (or name if the group variable has no label)

† ODSFOOTNOTE2=FOOTNOTE2 | '*string***'**

adds a secondary footnote to ODS Graphics output. If you specify the FOOTNOTE2 keyword, the value of SAS FOOTNOTE2 statement is used as the secondary graph footnote. If you specify a quoted string, that is used as the secondary footnote. The quoted string can contain any of the following escaped characters, which are replaced with the appropriate values from the analysis:

\n	analysis variable name
\l	analysis variable label (or name if the analysis variable has no label)
\x	group variable name
\s	group variable label (or name if the group variable has no label)

† ODSTITLE=TITLE | TITLE1 | NONE | DEFAULT | LABELFMT | '*string*'
specifies a title for ODS Graphics output.

TITLE (or TITLE1) uses the value of SAS TITLE statement as the graph title.

NONE suppresses all titles from the graph.

DEFAULT uses the default ODS Graphics title (a descriptive title that consists of the plot type and the process variable name.)

LABELFMT uses the default ODS Graphics title with the variable label instead of the variable name.

If you specify a quoted string, that is used as the graph title. The quoted string can contain any of the following escaped characters, which are replaced with the appropriate values from the analysis:

\n analysis variable name

\l analysis variable label (or name if the analysis variable has no label)

\x group variable name

\s group variable label (or name if the group variable has no label)

† ODSTITLE2=TITLE2 | '*string*'
specifies a secondary title for ODS Graphics output. If you specify the TITLE2 keyword, the value of SAS TITLE2 statement is used as the secondary graph title. If you specify a quoted string, that is used as the secondary title. The quoted string can contain any of the following escaped characters, which are replaced with the appropriate values from the analysis:

\n analysis variable name

\l analysis variable label (or name if the analysis variable has no label)

\x group variable name

\s group variable label (or name if the group variable has no label)

† OUTBOX=*SAS-data-set*
creates an output data set that contains group summary statistics and outlier values for a box plot. You can use an OUTBOX= data set as a BOX= input data set in a subsequent run of the procedure. See the section "OUTBOX= Data Set" on page 1467 for details.

OUTHIGHHTML=*variable*
specifies a variable whose values are URLs to be associated with outlier points above the upper fence on a schematic box plot when graphics output is directed into HTML.

† OUTHISTORY=*SAS-data-set*
creates an output data set that contains the group summary statistics. You can use an OUTHISTORY= data set as a HISTORY= input data set in a subsequent run of the procedure. See the section "OUTHISTORY= Data Set" on page 1468 for details.

OUTLOWHTML=*variable*
specifies a variable whose values are URLs to be associated with outlier points below the lower fence on a schematic box plot when graphics output is directed into HTML.

† OVERLAY=(*variable-list***)**
specifies variables to be plotted as overlays on the box plot. One value for each overlay variable is plotted at each group position. If there are multiple observations with the same group variable value in the input data set, the overlay variable values from the first observation in each group are plotted. By default, the points in an overlay plot are connected with line segments.

OVERLAYCLIPSYM=*symbol*
specifies the symbol used to plot clipped values on overlay plots when the CLIPFACTOR= option is used.

OVERLAYCLIPSYMHT=*value*
specifies the height for the symbol used to plot clipped values on overlay plots when the CLIPFACTOR= option is used.

OVERLAYHTML=(*variable-list***)**
specifies variables whose values are URLs to be associated with points on overlay plots when graphics output is directed into HTML. Variables in the OVERLAYHTML= list are matched with variables in the corresponding positions in the OVERLAY= list.

OVERLAYID=(*variable-list***)**
specifies variables whose formatted values are used to label points on overlays. Variables in the OVERLAYID= list are matched with variables in the corresponding positions in the OVERLAY= list. The value of the OVERLAYID= variable should be the same for each observation with a given value of the group variable.

† OVERLAYLEGLAB=‘*label*’
specifies the label displayed to the left of the overlay legend produced by the OVERLAY= option. The label can be up to 16 characters and must be enclosed in quotes. The default label is “Overlays:”.

OVERLAYSYM=(*symbol-list***)**
specifies symbols used to plot overlay variables. Symbols in the OVERLAYSYM= list are matched with variables in the corresponding positions in the OVERLAY= list.

OVERLAYSYMHT=(*value-list***)**
specifies the heights of symbols used to plot overlay variables. Symbol heights in the OVERLAYSYMHT= list are matched with variables in the corresponding positions in the OVERLAY= list.

PAGENUM=‘*string*’
specifies the form of the label used for pagination. The string can be up to 16 characters, and it must include one or two occurrences of the substitution character ‘#’. The first ‘#’ is replaced with the page number, and the optional second ‘#’ is replaced with the total number of pages.

The PAGENUM= option is useful when you are working with a large number of groups, resulting in multiple pages of output. For example, suppose that each of the following PLOT statements produces multiple pages:

```
proc boxplot data=Pistons;
   plot Diameter*Hour / pagenum='Page #';
   plot Diameter*Hour / pagenum='Page # of #';
   plot Diameter*Hour / pagenum='#/#';
run;
```

The third page produced by the first statement would be labeled "Page 3". The third page produced by the second statement would be labeled "Page 3 of 5". The third page produced by the third statement would be labeled "3/5".

By default, no page number is displayed.

PAGENUMPOS=TL | TR | BL | BR | TL100 | TR100 | BL0 | BR0

specifies where to position the page number requested with the PAGENUM= option. The keywords TL, TR, BL, and BR correspond to the positions top left, top right, bottom left, and bottom right, respectively. You can use the TL100 and TR100 keywords to ensure that the page number appears at the very top of a page when a title is displayed. The BL0 and BR0 keywords ensure that the page number appears at the very bottom of a page when footnotes are displayed.

The default value is BR.

† PCTLDEF=*index*

specifies one of five definitions used to calculate percentiles in the construction of box-and-whiskers plots. The index can be 1, 2, 3, 4, or 5. The five corresponding percentile definitions are discussed in the section "Percentile Definitions" on page 1473. The default index is 5.

† REPEAT

† REP

specifies that the horizontal axis of a plot that spans multiple panels be arranged so that the last group position on a panel is repeated as the first group position on the next panel. The REPEAT option facilitates cutting and pasting panels together. When a SAS DATETIME format is associated with the group variable, the REPEAT option is the default.

SKIPHLABELS=*n*

SKIPHLABEL=*n*

specifies the number *n* of consecutive tick mark labels, beginning with the second tick mark label, that are thinned (not displayed) on the horizontal (group) axis. For example, specifying SKIPHLABEL=1 causes every other label to be skipped. Specifying SKIPHLABEL=2 causes the second and third labels to be skipped, the fifth and sixth labels to be skipped, and so forth.

The default value of the SKIPHLABELS= option is the smallest value *n* for which tick mark labels do not collide. A specified *n* will be overridden to avoid collision. To reduce thinning, you can use the TURNHLABELS option.

SYMBOLLEGEND=LEGEND*n* | NONE

controls the legend for the levels of a symbol variable (see Example 32.2). You can specify SYMBOLLEGEND=LEGEND*n*, where *n* is the number of a LEGEND statement defined previously. You can specify SYMBOLLEGEND=NONE to suppress the default legend. See *SAS/GRAPH: Reference* for more information about the LEGEND statement.

SYMBOLORDER=DATA | INTERNAL | FORMATTED

SYMORD=DATA | INTERNAL | FORMATTED

specifies the order in which symbols are assigned for levels of the symbol variable. The DATA keyword assigns symbols to values in the order in which values appear in the input data set. The INTERNAL keyword assigns symbols based on sorted order of internal values of the symbol variable, and the FORMATTED keyword assigns them based on sorted formatted values. The default value is FORMATTED.

† TOTPANELS=*n*

specifies the total number of panels to be used to display the plot. This option overrides the NPANELPOS= option.

TURNHLABELS

TURNHLABEL

turns the major tick mark labels for the horizontal (group) axis so that they are arranged vertically. By default, labels are arranged horizontally.

Note that arranging the labels vertically might leave insufficient vertical space on the panel for a plot.

† VAXIS=*value-list*

VAXIS=AXIS*n*

specifies major tick mark values for the vertical axis of a box plot. The values must be listed in increasing order, must be evenly spaced, and must span the range of values displayed in the plot. You can specify the values with an explicit list or with an implicit list, as shown in the following example:

```
proc boxplot;
   plot Width*Hour / vaxis=0 2 4 6 8;
   plot Width*Hour / vaxis=0 to 8 by 2;
run;
```

You can also specify a previously defined AXIS statement with the VAXIS= option.

† VFORMAT=*format*

specifies a format to be used to display tick mark labels on the vertical axis of the box plot.

VMINOR=*n*

VM=*n*

specifies the number of minor tick marks between major tick marks on the vertical axis. Minor tick marks are not labeled. By default, VMINOR=0.

VOFFSET=*value*

specifies the length in percent screen units of the offset at the ends of the vertical axis.

† VREF=*value-list* | *SAS-data-set*

draws reference lines perpendicular to the vertical axis. You can use this option in the following ways:

- Specify the values for the lines with a VREF= list:

```
vref=20
vref=20 40 80
```

- Specify the values for the lines as the values of a numeric variable named _REF_ in a VREF= data set. Optionally, you can provide labels for the lines as values of a variable named _REFLAB_, which must be a character variable of up to 16 characters. If you want distinct reference lines to be displayed in plots for different analysis variables specified in the PLOT statement, you must include a character variable named _VAR_, whose values are the names of the analysis variables. If you do not include the variable _VAR_, all of the lines are displayed in all of the plots. Each observation in the VREF= data set corresponds to a reference line. If BY variables are used in the input data set, the same BY-variable structure must be used in the VREF= data set unless you specify the NOBYREF option.

† VREFLABELS=‘*label1*’ . . . ‘*labeln*’
† VREFLABEL=‘*label1*’ . . . ‘*labeln*’
† VREFLAB=‘*label1*’ . . . ‘*labeln*’

specifies labels for the reference lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*

specifies the horizontal position of the VREFLABELS= label, as described in the following table. By default, $n = 1$.

n	Label Position
1	Left-justified in plot area
2	Right-justified in plot area
3	Left-justified in right margin

VZERO

forces the origin to be included in the vertical axis for a box plot.

WAXIS=*n*

specifies the width in pixels for the axis and frame lines. By default, $n = 1$.

WGRID=*n*

specifies the width in pixels for grid lines requested with the ENDGRID and GRID options. By default, $n = 1$.

† WHISKERPERCENTILE=*p*

specifies that the whiskers of the box-and-whisker plots be drawn to the pth and $(100 - p)$th percentiles. For example, if you specify WHISKERPERCENTILE=10 the whiskers are drawn to the 10th and 90th percentiles. Observations that lie beyond the whiskers are outliers and there are no far outliers.

By default, whiskers are drawn to the minimum and maximum data values if the BOXSTYLE= value is SKELETAL (the default), and to the most extreme values within or equal to the lower and upper fences otherwise.

WOVERLAY=(*value-list*)
specifies the widths in pixels for the line segments connecting points on overlay plots. Widths in the WOVERLAY= list are matched with variables in the corresponding positions in the OVERLAY= list. By default, all overlay widths are 1.

Details: BOXPLOT Procedure

Summary Statistics Represented by Box Plots

Table 32.7 lists the summary statistics represented in each box-and-whiskers plot.

Table 32.7 Summary Statistics Represented by Box Plots

Group Summary Statistic	Feature of Box-and-Whiskers Plot
Maximum	Endpoint of upper whisker
Third quartile (75th percentile)	Upper edge of box
Median (50th percentile)	Line inside box
Mean	Symbol marker
First quartile (25th percentile)	Lower edge of box
Minimum	Endpoint of lower whisker

Note that you can request different box plot styles, as discussed in the section "Styles of Box Plots" on page 1472, and as illustrated in Example 32.2.

Output Data Sets

OUTBOX= Data Set

The OUTBOX= data set saves group summary statistics and outlier values. The following variables can be saved:

- the group variable
- the variable _VAR_, containing the analysis variable name
- the variable _TYPE_, identifying features of box-and-whiskers plots
- the variable _VALUE_, containing values of box-and-whiskers plot features
- the variable _ID_, containing labels for outliers
- the variable _HTML_, containing URLs associated with plot features

_ID_ is included in the OUTBOX= data set only if the keyword SCHEMATICID or SCHEMATICID-FAR is specified with the BOXSTYLE= option. _HTML_ is present only if one or more of the HTML=, OUTHIGHHTML=, and OUTLOWHTML= options are specified.

Each observation in an OUTBOX= data set records the value of a single feature of one group's box-and-whiskers plot, such as its mean. The _TYPE_ variable identifies the feature whose value is recorded in _VALUE_. Table 32.8 lists valid _TYPE_ variable values.

Table 32.8 Valid _TYPE_ Values in an OUTBOX= Data Set

TYPE	Description
N	Group size
MIN	Minimum group value
Q1	Group first quartile
MEDIAN	Group median
MEAN	Group mean
Q3	Group third quartile
MAX	Group maximum value
STDDEV	Group standard deviation
LOW	Low outlier value
HIGH	High outlier value
LOWHISKR	Low whisker value, if different from MIN
HIWHISKR	High whisker value, if different from MAX
FARLOW	Low far outlier value
FARHIGH	High far outlier value

Additionally, the following variables, if specified, are included:

- block variables
- symbol variable
- BY variables
- ID variables

OUTHISTORY= Data Set

The OUTHISTORY= data set saves group summary statistics. The following variables are saved:

- the group variable
- group minimum variables named by *analysis-variable* suffixed with *L*
- group first-quartile variables named by *analysis-variable* suffixed with 1
- group mean variables named by *analysis-variable* suffixed with *X*

- group median variables named by *analysis-variable* suffixed with *M*
- group third-quartile variables named by *analysis-variable* suffixed with 3
- group maximum variables named by *analysis-variable* suffixed with *H*
- group standard deviation variables named by *analysis-variable* suffixed with *S*
- group size variables named by *analysis-variable* suffixed with *N*

If an analysis variable name has the maximum length of 32 characters, PROC BOXPLOT forms summary statistic names from its first 16 characters, its last 15 characters, and the appropriate suffix.

Group summary variables are created for each analysis variable specified in the PLOT statement. For example, consider the following statements:

```
proc boxplot data=Steel;
   plot (Width Diameter)*Lot / outhistory=Summary;
run;
```

The data set Summary contains variables named Lot, WidthL, Width1, WidthM, WidthX, Width3, WidthH, WidthS, WidthN, DiameterL, Diameter1, DiameterM, DiameterX, Diameter3, DiameterH, DiameterS, and DiameterN.

Additionally, the following variables, if specified, are included:

- BY variables
- block variables
- symbol variable
- ID variables

Note that an OUTHISTORY= data set does not contain outlier values, and therefore cannot be used, in general, to save a schematic box plot. You can use an OUTBOX= data set to save a schematic box plot summary.

Input Data Sets

DATA= Data Set

You can read analysis variable measurements from a data set specified with the DATA= option in the PROC BOXPLOT statement. Each analysis variable specified in the PLOT statement must be a SAS variable in the data set. This variable provides measurements that are organized into groups indexed by the group variable. The group variable, specified in the PLOT statement, must also be a SAS variable in the DATA= data set. Each observation in a DATA= data set must contain a value for each analysis variable and a value for the group variable. If the ith group contains n_i measurements, there should be n_i consecutive observations for which the value of the group variable is the index of the ith group. For example, if each group contains 20 items and there are 30 groups, the DATA= data set should contain 600 observations. Other variables that can be read from a DATA= data set include the following:

- block variables
- symbol variable
- BY variables
- ID variables

BOX= Data Set

You can read group summary statistics and outlier information from a BOX= data set specified in the PROC BOXPLOT statement. This enables you to reuse OUTBOX= data sets that have been created in previous runs of the BOXPLOT procedure to reproduce schematic box plots.

A BOX= data set must contain the following variables:

- the group variable
- _VAR_, containing the analysis variable name
- _TYPE_, identifying features of box-and-whiskers plots
- _VALUE_, containing values of those features

Each observation in a BOX= data set records the value of a single feature of one group's box-and-whiskers plot, such as its mean. Consequently, a BOX= data set contains multiple observations per group. These must appear consecutively in the BOX= data set.

The _TYPE_ variable identifies the feature whose value is recorded in a given observation. Table 32.9 lists valid _TYPE_ variable values.

Table 32.9 Valid _TYPE_ Values in a BOX= Data Set

TYPE	Description
N	Group size
MIN	Group minimum value
Q1	Group first quartile
MEDIAN	Group median
MEAN	Group mean
Q3	Group third quartile
MAX	Group maximum value
STDDEV	Group standard deviation
LOW	Low outlier value
HIGH	High outlier value
LOWHISKR	Low whisker value, if different from MIN
HIWHISKR	High whisker value, if different from MAX
FARLOW	Low far outlier value
FARHIGH	High far outlier value

The features identified by _TYPE_ values N, MIN, Q1, MEDIAN, MEAN, Q3, and MAX are required for each group.

Other variables that can be read from a BOX= data set include the following:

- the variable _ID_, containing labels for outliers
- the variable _HTML_, containing URLs to be associated with features on box plots
- block variables
- symbol variable
- BY variables
- ID variables

When you specify the keyword SCHEMATICID or SCHEMATICIDFAR with the BOXSTYLE= option, values of _ID_ are used as outlier labels. If _ID_ does not exist in the BOX= data set, the values of the first variable listed in the ID statement are used.

HISTORY= Data Set

You can read group summary statistics from a HISTORY= data set specified in the PROC BOXPLOT statement. This enables you to reuse OUTHISTORY= data sets that have been created in previous runs of the BOXPLOT procedure or to read output data sets created with SAS summarization procedures, such as PROC UNIVARIATE.

Note that a HISTORY= data set does *not* contain outlier information. Therefore, in general you cannot reproduce a schematic box plot from summary statistics saved in an OUTHISTORY= data set. To save and reproduce schematic box plots, use OUTBOX= and BOX= data sets.

A HISTORY= data set must contain the following:

- the group variable
- a group minimum variable for each analysis variable
- a group first-quartile variable for each analysis variable
- a group median variable for each analysis variable
- a group mean variable for each analysis variable
- a group third-quartile variable for each analysis variable
- a group maximum variable for each analysis variable
- a group standard deviation variable for each analysis variable
- a group size variable for each analysis variable

The names of the group summary statistics variables must be the analysis variable name concatenated with the following special suffix characters.

Group Summary Statistic	Suffix Character
Group minimum	L
Group first quartile	1
Group median	M
Group mean	X
Group third quartile	3
Group maximum	H
Group standard deviation	S
Group size	N

For example, consider the following statements:

```
proc boxplot history=Summary;
   plot (Weight Yieldstrength) * Batch;
run;
```

The data set Summary must include the variables Batch, WeightL, Weight1, WeightM, WeightX, Weight3, WeightH, WeightS, WeightN, YieldstrengthL, Yieldstrength1, YieldstrengthM, YieldstrengthX, Yieldstrength3, YieldstrengthH, YieldstrengthS, and YieldstrengthN.

Note that if you specify an analysis variable whose name contains the maximum of 32 characters, the summary variable names must be formed from the first 16 characters and the last 15 characters of the analysis variable name, suffixed with the appropriate character.

These other variables can be read from a HISTORY= data set:

- block variables
- symbol variable
- BY variables
- ID variables

Styles of Box Plots

A box-and-whiskers plot is displayed for the measurements in each group on the box plot. The *skeletal* style of the box-and-whiskers plot shown in Figure 32.6 is the default. You can produce a *schematic* box plot by specifying the BOXSTYLE=SCHEMATIC option in the PLOT statement. Figure 32.8 illustrates a typical schematic box plot and the locations of the fences (which are not displayed in actual output). See the description of the BOXSTYLE= option for complete details.

Figure 32.8 Schematic Box-and-Whiskers Plot

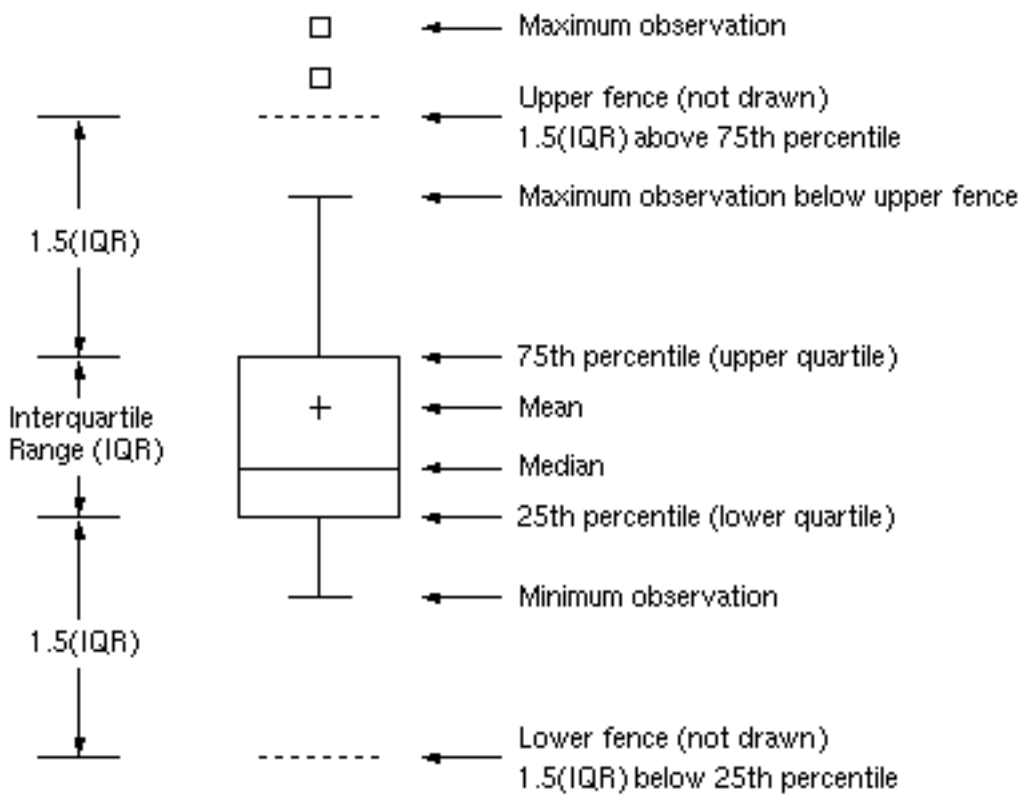

You can draw connecting lines between adjacent box-and-whiskers plots by using the BOXCONNECT=*keyword* option. For example, BOXCONNECT=MEAN connects the points that represent the means of adjacent groups. Other available keywords are MIN, Q1, MEDIAN, Q3, and MAX. Specifying BOXCONNECT without a keyword is equivalent to specifying BOXCONNECT=MEAN. You can specify the color for the connecting lines with the CCONNECT= option.

Percentile Definitions

You can use the PCTLDEF= option to specify one of five definitions for computing quantile statistics (percentiles). Suppose that n is the number of nonmissing values for a variable and that $x_1, x_2, \ldots, x_n$ represent the ordered values of the analysis variable. For the tth percentile, set $p = t/100$ and let

$$\begin{aligned} np &= j + g \quad \text{when PCTLDEF=1, 2, 3, or 5} \\ (n+1)p &= j + g \quad \text{when PCTLDEF=4} \end{aligned}$$

where j is the integer part of the quantity and g is the fractional part of the quantity.

The tth percentile (call it y) can be defined as follows:

PCTLDEF=1 weighted average at x_{np}

$$y = (1-g)x_j + gx_{j+1}$$

where x_0 is taken to be x_1.

PCTLDEF=2 observation numbered closest to np

$$y = x_i$$

where i is the integer part of $np + 1/2$

PCTLDEF=3 empirical distribution function

$$y = x_j \text{ if } g = 0$$

$$y = x_{j+1} \text{ if } g > 0$$

PCTLDEF=4 weighted average aimed at $x_{p(n+1)}$

$$y = (1 - g)x_j + gx_{j+1}$$

where x_{n+1} is taken to be x_n.

PCTLDEF=5 empirical distribution function with averaging

$$y = (x_j + x_{j+1})/2 \text{ if } g = 0$$

$$y = x_{j+1} \text{ if } g > 0$$

Missing Values

An observation read from an input data set is not analyzed if the value of the group variable is missing. For a particular analysis variable, an observation read from a DATA= data set is not analyzed if the value of the analysis variable is missing.

Continuous Group Variables

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/boxdet1.sas.)

By default, the PLOT statement treats numerical group variable values as *discrete* values and spaces the boxes evenly on the plot. The following statements produce the box plot in Figure 32.9:

```
ods graphics off;
title 'Box Plot for Power Output';
proc boxplot data=Turbine;
   plot KWatts*Day;
run;
```

Figure 32.9 Box Plot with Discrete Group Variable

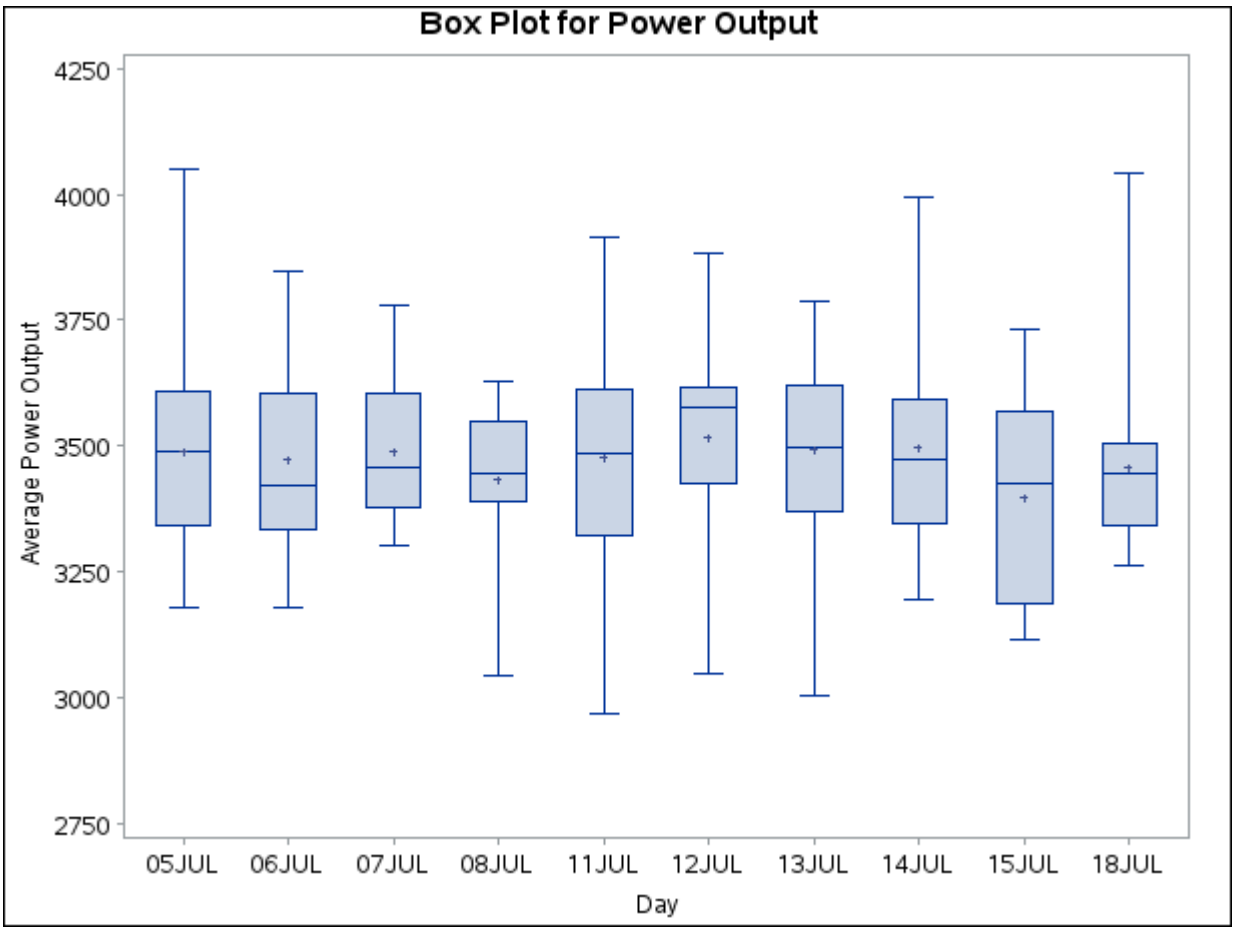

The labels on the horizontal axis in Figure 32.9 do not represent 10 consecutive days, but the box-and-whiskers plots are evenly spaced.

In order to treat the group variable as *continuous*, you can specify the CONTINUOUS or HAXIS= option when producing traditional graphics. Either option produces a box plot with a horizontal axis scaled for continuous group variable values. (ODS Graphics does not support a continuous group axis.)

The following statements produce the plot shown in Figure 32.10. The TURNHLABELS option orients the horizontal axis labels vertically so there is room to display them all.

```
title 'Box Plot for Power Output';
proc boxplot data=Turbine;
   plot KWatts*Day / turnhlabels
                     continuous;
run;
```

Figure 32.10 Box Plot with Continuous Group Variable

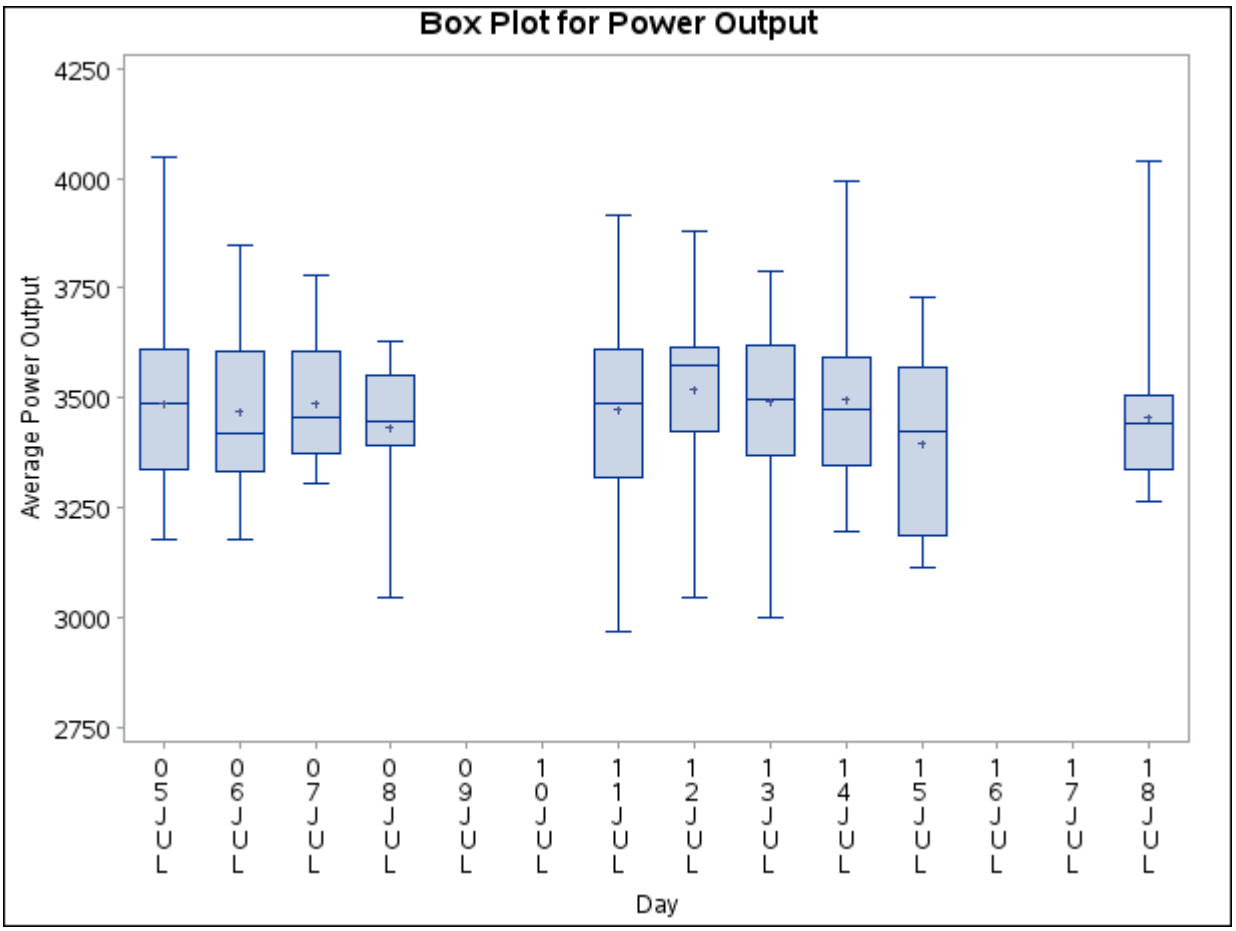

Note that the tick values on the horizontal axis represent consecutive days and that no box-and-whiskers plots are displayed for days when no turbine data were collected.

Positioning Insets

This section provides details on three different methods of positioning INSET boxes by using the POSITION= option. With the POSITION= option, you can specify the following:

- compass points
- keywords for margin positions
- coordinates in data units or percent axis units

Positioning the Inset Using Compass Points

You can specify the eight compass points (N, NE, E, SE, S, SW, W, and NW) as keywords for the POSITION= option. The default inset position is NW. The following statements create the display in Figure 32.11, which illustrates all eight compass positions:

```
title 'Box Plot for Power Output';
proc boxplot data=Turbine;
   plot KWatts*Day;
   inset nobs / height=2.5 cfill=blank header='NW' pos=nw;
   inset nobs / height=2.5 cfill=blank header='N ' pos=n ;
   inset nobs / height=2.5 cfill=blank header='NE' pos=ne;
   inset nobs / height=2.5 cfill=blank header='E ' pos=e ;
   inset nobs / height=2.5 cfill=blank header='SE' pos=se;
   inset nobs / height=2.5 cfill=blank header='S ' pos=s ;
   inset nobs / height=2.5 cfill=blank header='SW' pos=sw;
   inset nobs / height=2.5 cfill=blank header='W ' pos=w ;
run;
```

Figure 32.11 Insets Positioned Using Compass Points

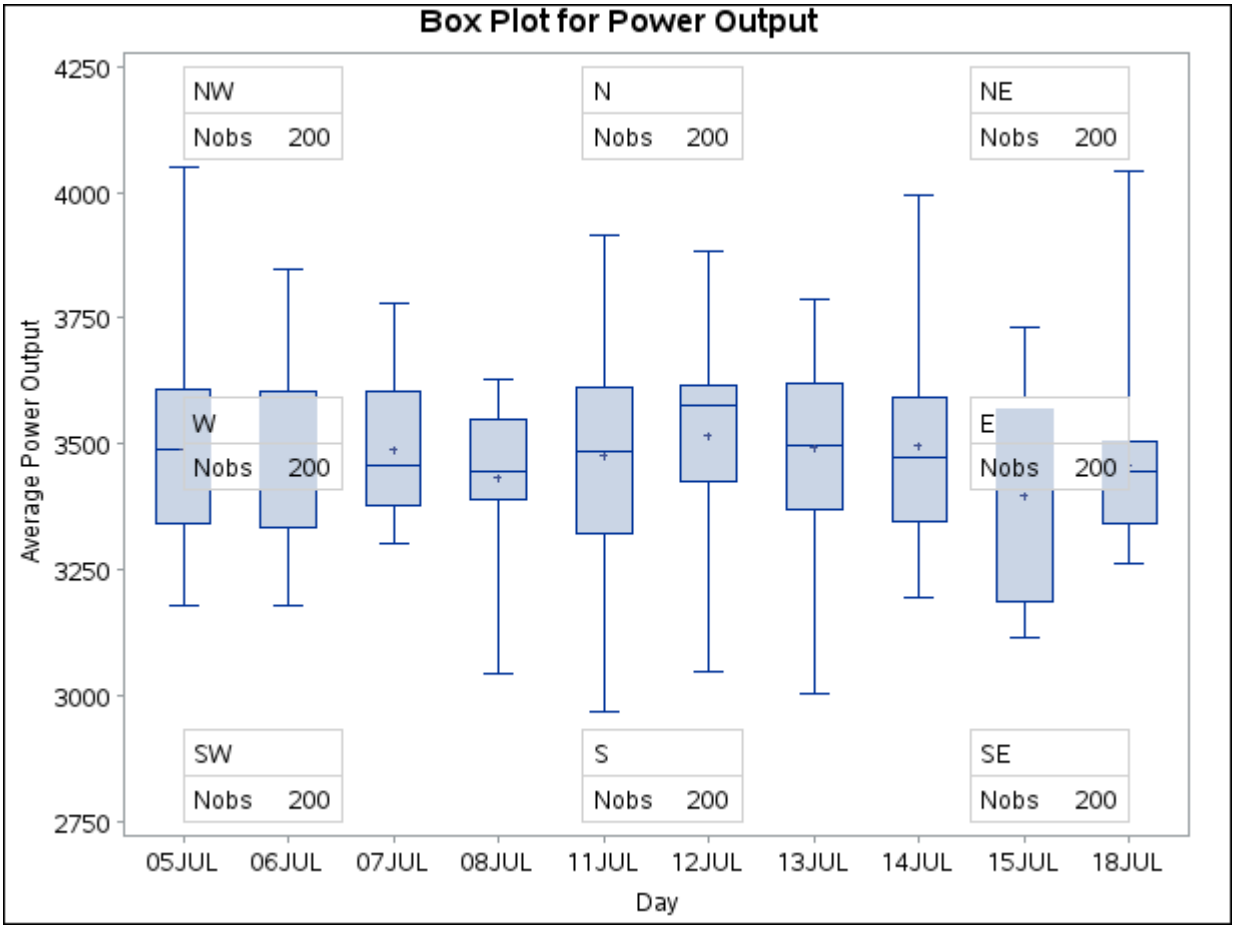

Positioning the Inset in the Margins

You can also use the INSET statement to position an inset in one of the four margins that surround the plot area by using the margin keyword LM, RM, TM, or BM, as illustrated in Figure 32.12.

Figure 32.12 Positioning Insets in the Margins

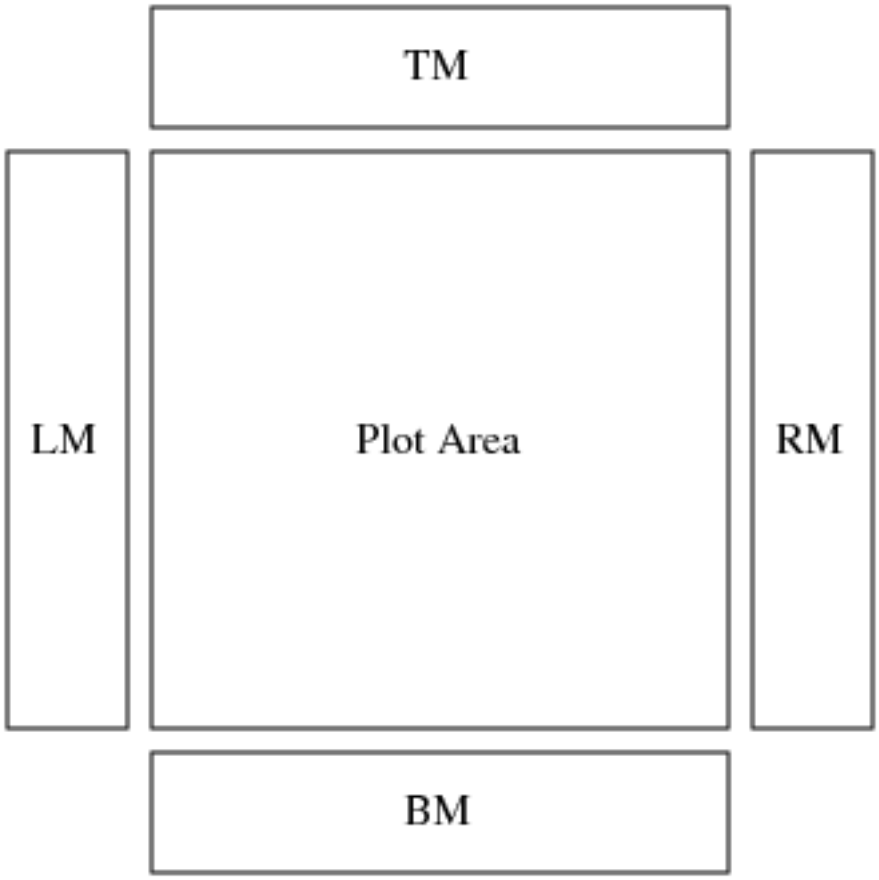

For an example of an inset placed in the top margin, see Output 32.1.1. Margin positions are recommended for insets that contain a large number of statistics. If you attempt to display a lengthy inset in the interior of the plot, it is likely that the inset will collide with the data display.

Positioning the Inset Using Coordinates

You can also specify the position of an inset with coordinates by using the POSITION= (x, y) option. You can specify coordinates in axis percent units (the default) or in axis data units.

Data Unit Coordinates

If you specify the DATA option immediately following the coordinates, the inset is positioned using axis data units. For example, the following statements place the bottom-left corner of the inset at 07JUL on the horizontal axis and 3950 on the vertical axis:

```
title 'Box Plot for Power Output';
proc boxplot data=Turbine;
   plot KWatts*Day;
   inset nobs /
      header   = 'Position=(07JUL,3950)'
      position = ('07JUL94'd, 3950) data;
run;
```

The box plot is displayed in Figure 32.13. By default, the specified coordinates determine the position of the bottom-left corner of the inset. You can change this reference point with the REFPOINT= option, as in the next example.

Figure 32.13 Inset Positioned Using Data Unit Coordinates

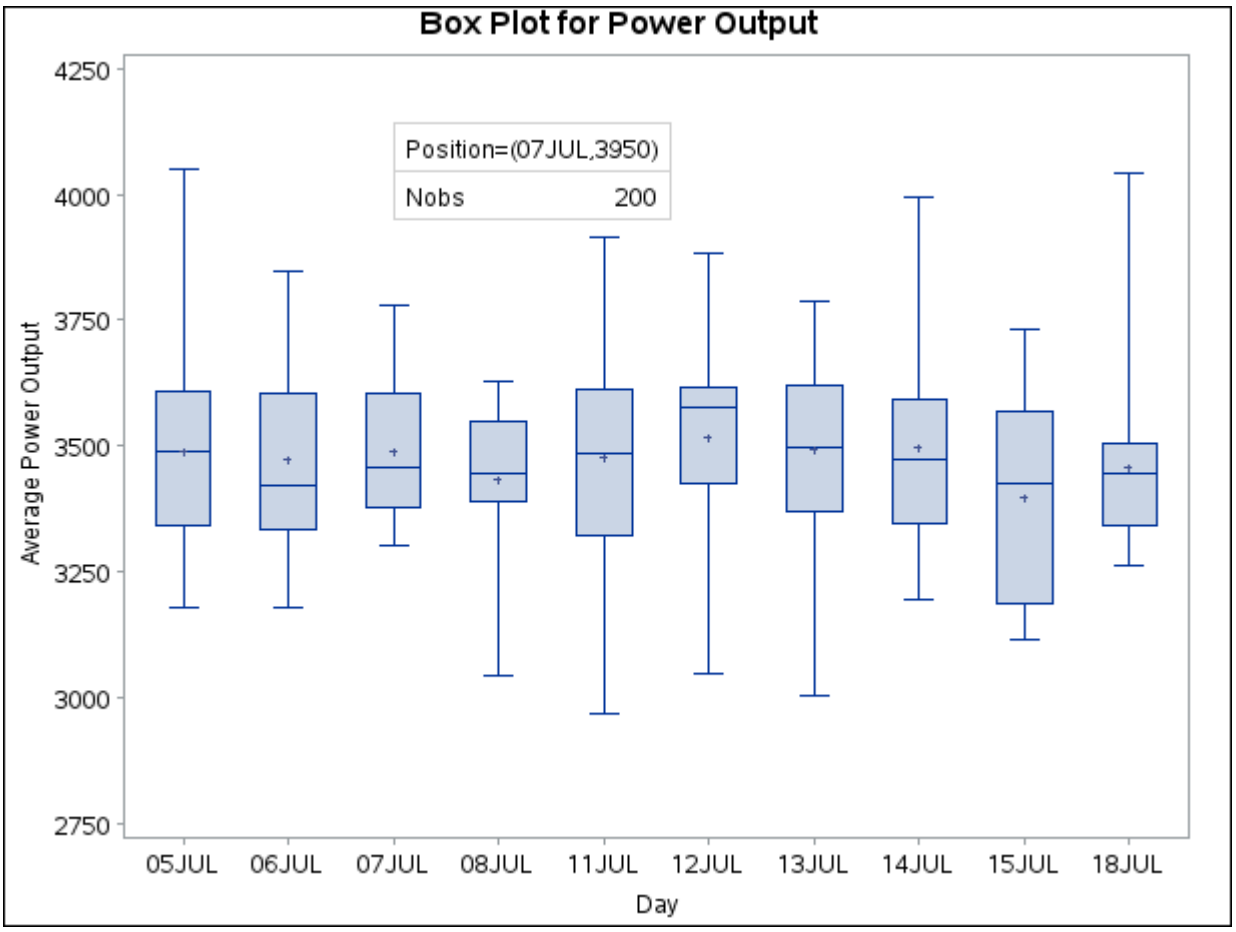

Axis Percent Unit Coordinates

If you do not use the DATA option, the inset is positioned using axis percent units. The coordinates of the bottom-left corner of the display are (0, 0), while the coordinates of the top-right corner are (100, 100). For example, the following statements create a box plot with two insets, both positioned using coordinates in axis percent units:

```
title 'Box Plot for Power Output';
proc boxplot data=Turbine;
   plot KWatts*Day;
   inset nmin / position = (5,25)
                header   = 'Position=(5,25)'
                height   = 3
                cfill    = ywh
                refpoint = tl;
   inset nmax / position = (95,95)
                header   = 'Position=(95,95)'
                height   = 3
                cfill    = ywh
                refpoint = tr;
run;
```

The display is shown in Figure 32.14. Notice that the REFPOINT= option is used to determine which corner of the inset is placed at the coordinates specified with the POSITION= option. The first inset has REFPOINT=TL, so the top-left corner of the inset is positioned 5% of the way across the horizontal axis and 25% of the way up the vertical axis. The second inset has REFPOINT=TR, so the top-right corner of the inset is positioned 95% of the way across the horizontal axis and 95% of the way up the vertical axis. Note also that coordinates in axis percent units must be *between* 0 and 100.

Figure 32.14 Inset Positioned Using Axis Percent Unit Coordinates

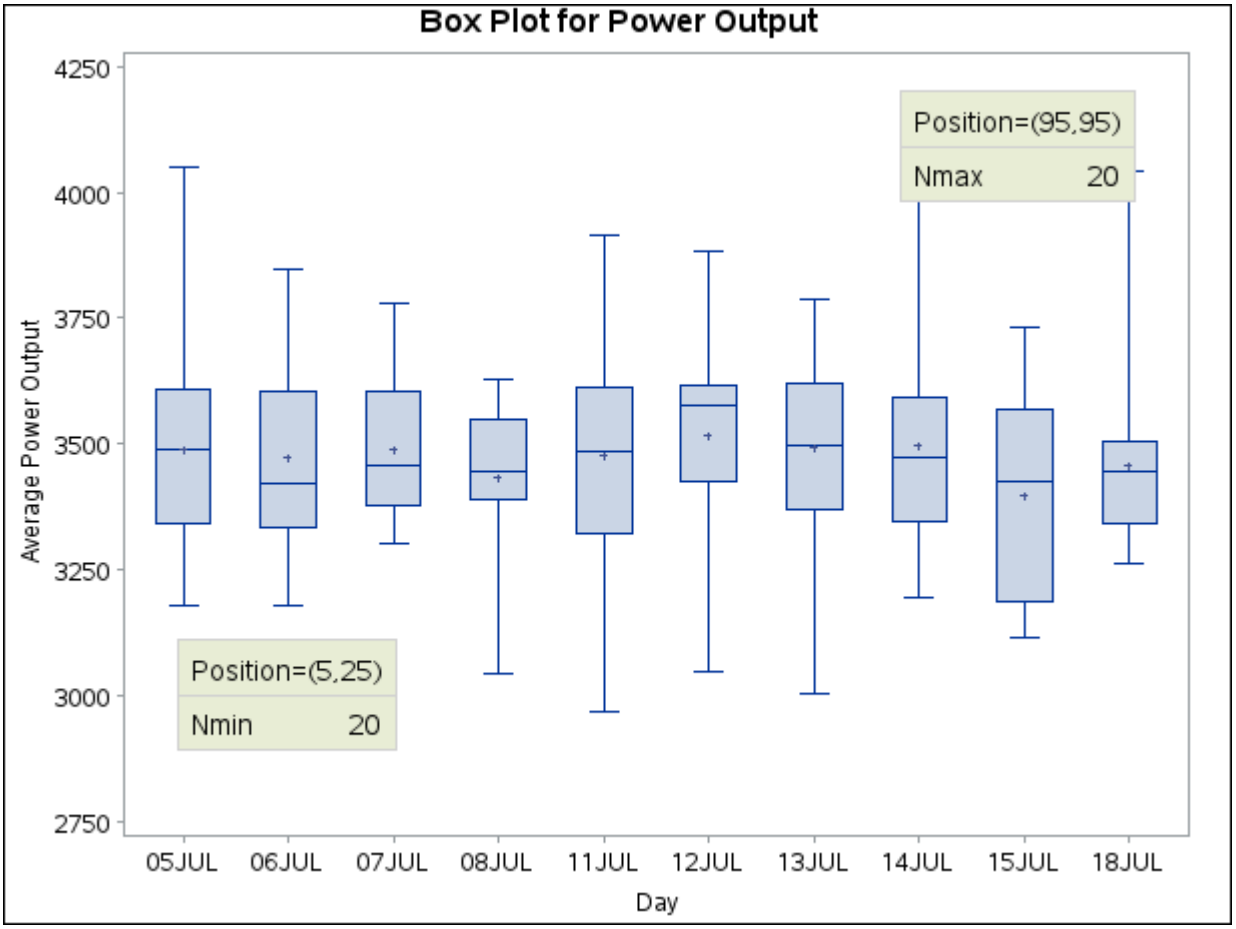

Displaying Blocks of Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/boxdet2.sas.)

To display data organized in blocks of consecutive observations, specify one or more *block variables* in parentheses after the group variable in the PLOT statement. The block variables must be variables in the input data set. The BOXPLOT procedure displays a legend that identifies blocks of consecutive observations with identical values of the block variables. The legend displays one track of values for each block variable that contains formatted values of the block variable.

The values of a block variable must be the same for all observations with the same value of the group variable. In other words, groups must be nested within blocks determined by block variables.

The following statements create a SAS data set that contains diameter measurements for a part produced on three different machines:

```
data Parts;
   length Machine $ 4;
   input Sample Machine $ @;
   do i= 1 to 4;
      input Diam @;
      output;
   end;
   drop i;
   datalines;
 1  A386  4.32 4.55 4.16 4.44
 2  A386  4.49 4.30 4.52 4.61
 3  A386  4.44 4.32 4.25 4.50
 4  A386  4.55 4.15 4.42 4.49
 5  A386  4.21 4.30 4.29 4.63
 6  A386  4.56 4.61 4.29 4.56
 7  A386  4.63 4.30 4.41 4.58
 8  A386  4.38 4.65 4.43 4.44
 9  A386  4.12 4.49 4.30 4.36
10  A455  4.45 4.56 4.38 4.51
11  A455  4.62 4.67 4.70 4.58
12  A455  4.33 4.23 4.34 4.58
13  A455  4.29 4.38 4.28 4.41
14  A455  4.15 4.35 4.28 4.23
15  A455  4.21 4.30 4.32 4.38
16  C334  4.16 4.28 4.31 4.59
17  C334  4.14 4.18 4.08 4.21
18  C334  4.51 4.20 4.28 4.19
19  C334  4.10 4.33 4.37 4.47
20  C334  3.99 4.09 4.47 4.25
21  C334  4.24 4.54 4.43 4.38
22  C334  4.23 4.48 4.31 4.57
23  C334  4.27 4.40 4.32 4.56
24  C334  4.70 4.65 4.49 4.38
;
```

The following statements create a box plot for the measurements in the Parts data set grouped into blocks by the block variable Machine:

```
ods graphics on;
title 'Box Plot for Diameter Grouped By Machine';
proc boxplot data=Parts;
   plot Diam*Sample (Machine) / odstitle = title;
   label Sample  = 'Sample Number'
         Machine = 'Machine'
         Diam    = 'Diameter';
run;
```

The ODSTITLE= option uses the title specified in the SAS title as the graph title. Note the LABEL statement used to provide labels for the axes and for the block legend. The plot is shown in Figure 32.15.

Figure 32.15 Box Plot Using a Block Variable

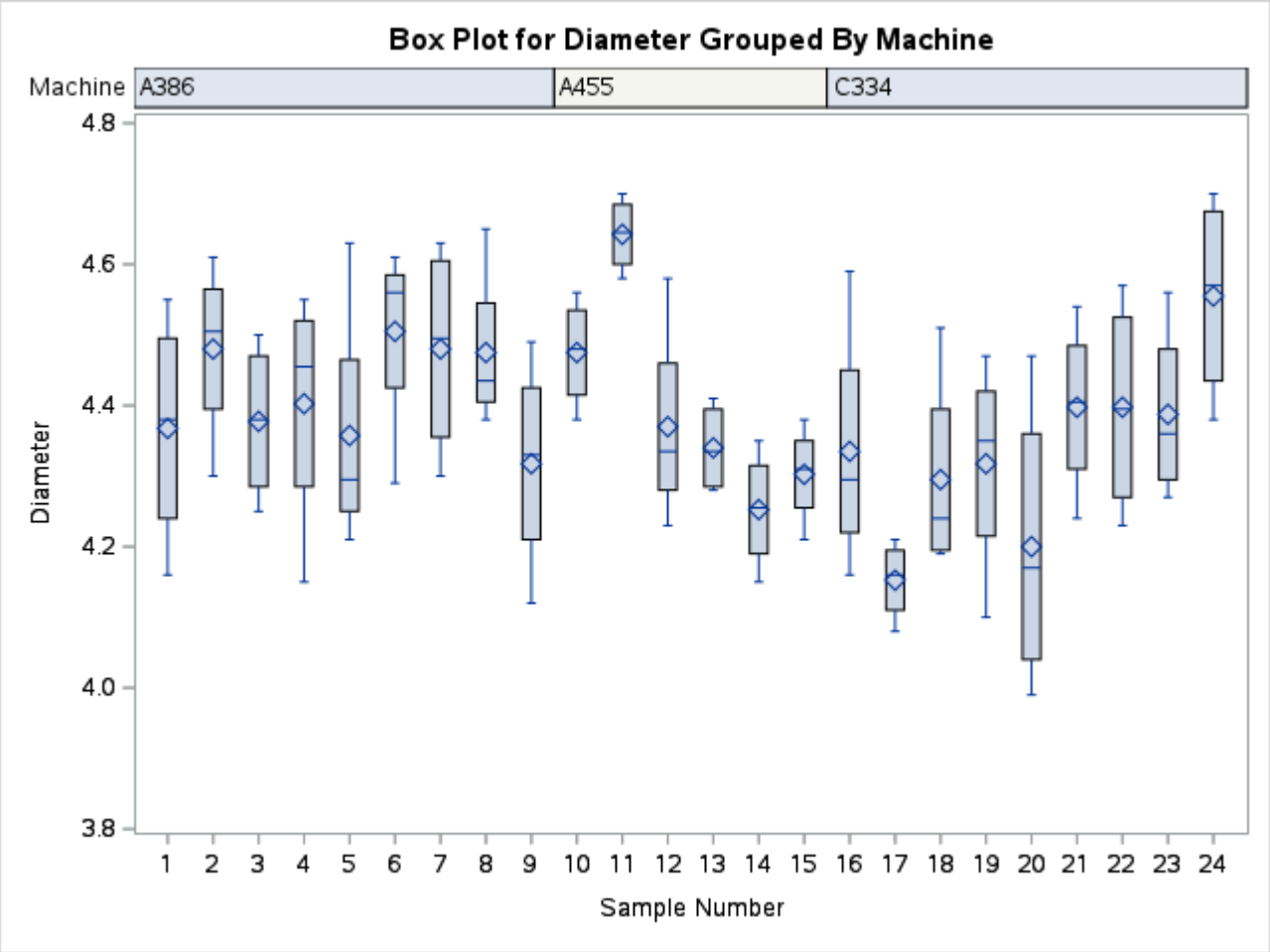

The unique consecutive values of Machine (A386, A455, and C334) are displayed in a legend above the plot. That is the default location of the block legend. You can control the position of the block legend with the BLOCKPOS= option. See the BLOCKPOS= option for details.

By default, block variable values that are too long to fit into the available space in a block legend are not displayed. You can specify the BLOCKLABTYPE= option to display lengthy labels. Specify

BLOCKLABTYPE=SCALED to scale down the text size of the values so they all fit. Use BLOCKLABTYPE=TRUNCATED to truncate lengthy values. You can also use BLOCKLABTYPE=*height* to specify a text height in vertical percent screen units for the values.

You can control the position of legend labels with the BLOCKLABELPOS= option. Valid BLOCKLABELPOS= values are ABOVE (the default, as shown in Figure 32.15) and LEFT.

Clipping Extreme Values

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/boxdet3.sas.)

By default a box plot's vertical axis is scaled to accommodate all the values in all groups. If the variation between groups is large with respect to the variation within groups, or if some groups contain extreme outlier values, the vertical axis scale can become so large that the box-and-whiskers plots are compressed. In such cases, you can clip the extreme values to produce a more readable plot, as illustrated in the following example.

A company produces copper tubing. The diameter measurements (in millimeters) for 15 batches of five tubes each are provided in the data set Newtubes:

```
data Newtubes;
   label Diameter='Diameter in mm';
   do Batch = 1 to 15;
      do  i = 1 to 5;
         input Diameter @@;
         output;
      end;
   end;
   datalines;
69.13  69.83  70.76  69.13  70.81
85.06  82.82  84.79  84.89  86.53
67.67  70.37  68.80  70.65  68.20
71.71  70.46  71.43  69.53  69.28
71.04  71.04  70.29  70.51  71.29
69.01  68.87  69.87  70.05  69.85
50.72  50.49  49.78  50.49  49.69
69.28  71.80  69.80  70.99  70.50
70.76  69.19  70.51  70.59  70.40
70.16  70.07  71.52  70.72  70.31
68.67  70.54  69.50  69.79  70.76
68.78  68.55  69.72  69.62  71.53
70.61  70.75  70.90  71.01  71.53
74.62  56.95  72.29  82.41  57.64
70.54  69.82  70.71  71.05  69.24
;
```

The following statements create a box plot of the tube diameters:

```
ods graphics on;
title 'Box Plot for New Copper Tubes' ;
proc boxplot data=Newtubes;
   plot Diameter*Batch / odstitle = title;
run;
```

The box plot is shown in Figure 32.16.

Figure 32.16 Compressed Box Plots

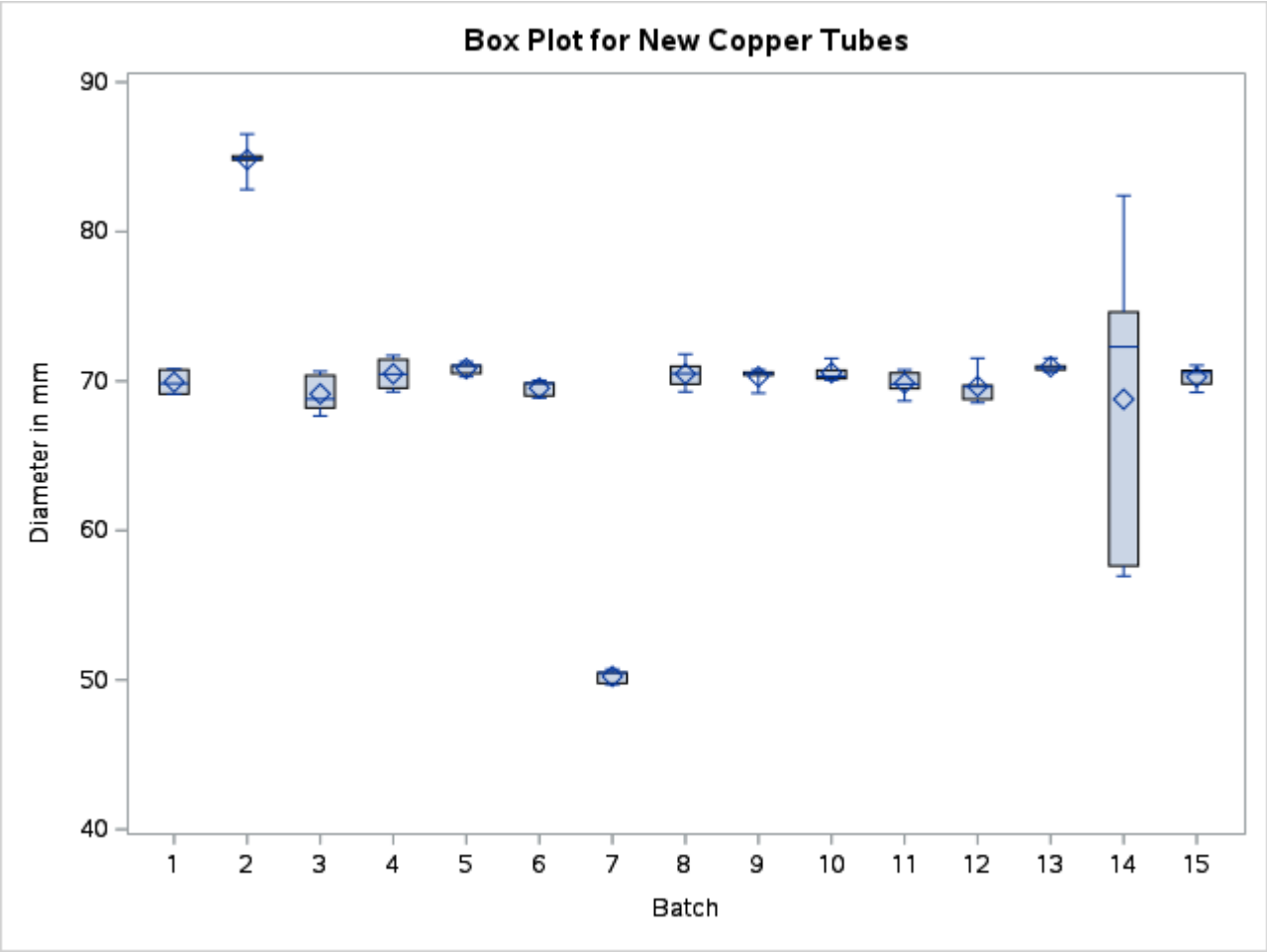

Note that the diameters in batch 2 are significantly larger, and those in batch 7 significantly smaller, than those in most of the other batches. The default vertical axis scaling causes the box-and-whiskers plots to be compressed.

You can produce a more useful box plot by specifying the CLIPFACTOR=*factor* option, where *factor* is a value greater than one. Clipping is applied as follows:

1. The mean of the first quartile values ($\overline{Q1}$) and the mean of the third quartile values ($\overline{Q3}$) are computed across all groups.

2. The following values define the clipping range:

$$y_{\max} = \overline{Q1} + (\overline{Q3} - \overline{Q1}) \times factor$$

and

$$y_{\min} = \overline{Q3} - (\overline{Q3} - \overline{Q1}) \times factor$$

Any statistic greater than $y_{\max}$ or less than $y_{\min}$ is ignored during vertical axis scaling.

NOTE:

- Clipping is applied only to the plotted statistics and not to the statistics saved in an output data set.
- A special symbol is used for clipped points (the default symbol is a square), and a legend that indicates the number of boxes that were clipped is added to the chart.

The following statements use a clipping factor of 1.5 to create a box plot of the same data plotted in Figure 32.16:

```
title 'Box Plot for New Copper Tubes' ;
proc boxplot data=Newtubes;
   plot Diameter*Batch /
      odstitle   = title
      clipfactor = 1.5;
run;
```

The clipped box plot is shown in Figure 32.17.

Figure 32.17 Box Plot with Clip Factor of 1.5

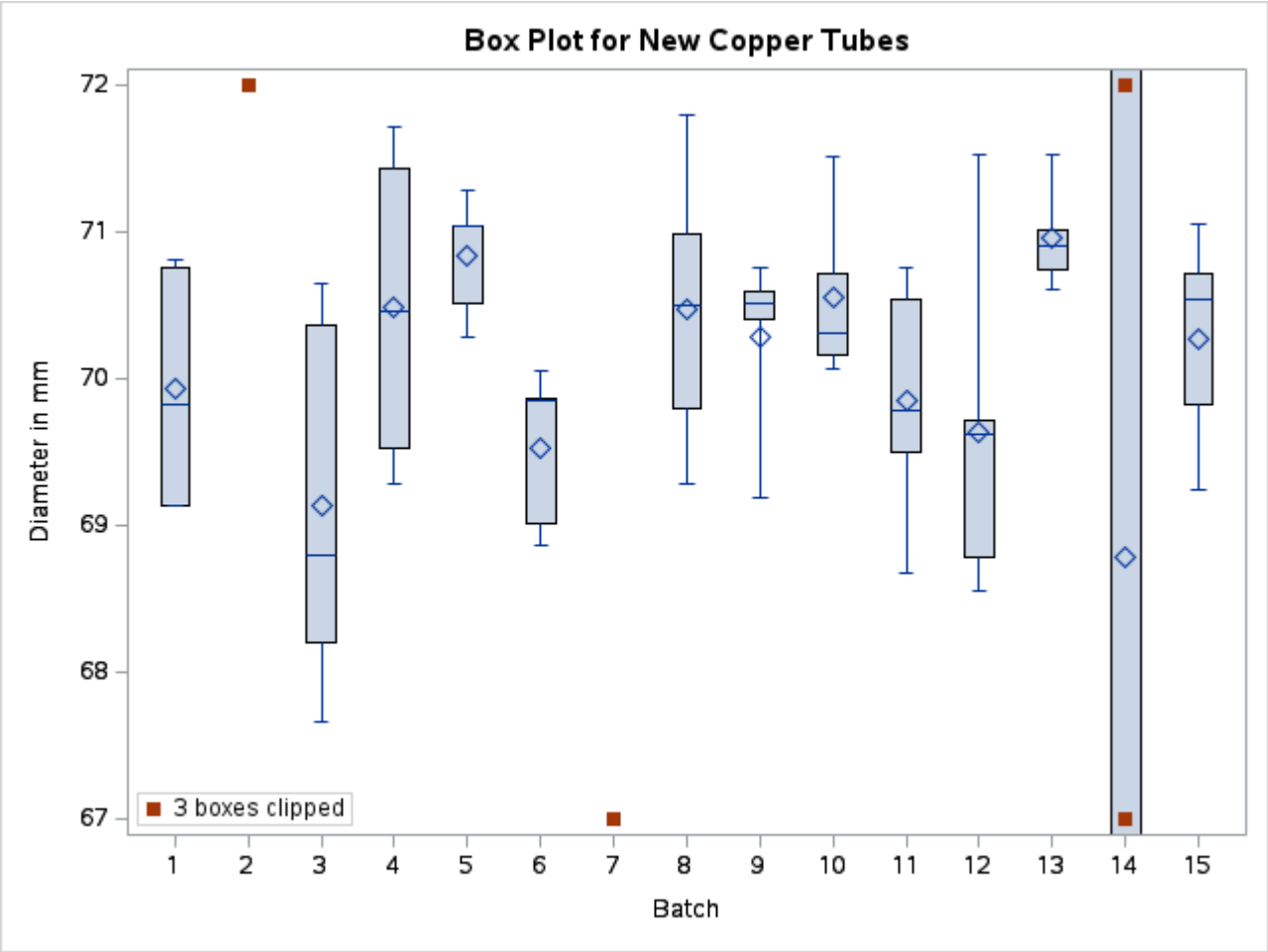

In Figure 32.17 the extreme values are clipped, making the box plot more readable. The box-and-whiskers plots for batches 2 and 7 are clipped completely, while the plot for batch 14 is clipped at both the top and bottom. Clipped points are marked with a square, and a clipping legend is added at the lower right of the display.

Other clipping options are available, as illustrated by the following statements:

```
title 'Box Plot for New Copper Tubes' ;
proc boxplot data=Newtubes;
   plot Diameter*Batch /
      odstitle    = title
      clipfactor  = 1.5
      cliplegend  = '# Clipped Boxes'
      clipsubchar = '#';
run;
```

The CLIPLEGEND= option requests a user-specified legend for the number of clipped boxes. Each occurrence in the legend of the character specified in the CLIPSUBCHAR= option is replaced by the number of clipped boxes.

Figure 32.18 shows the box plot with the modified clipping legend.

Figure 32.18 Box Plot with Clipping Options

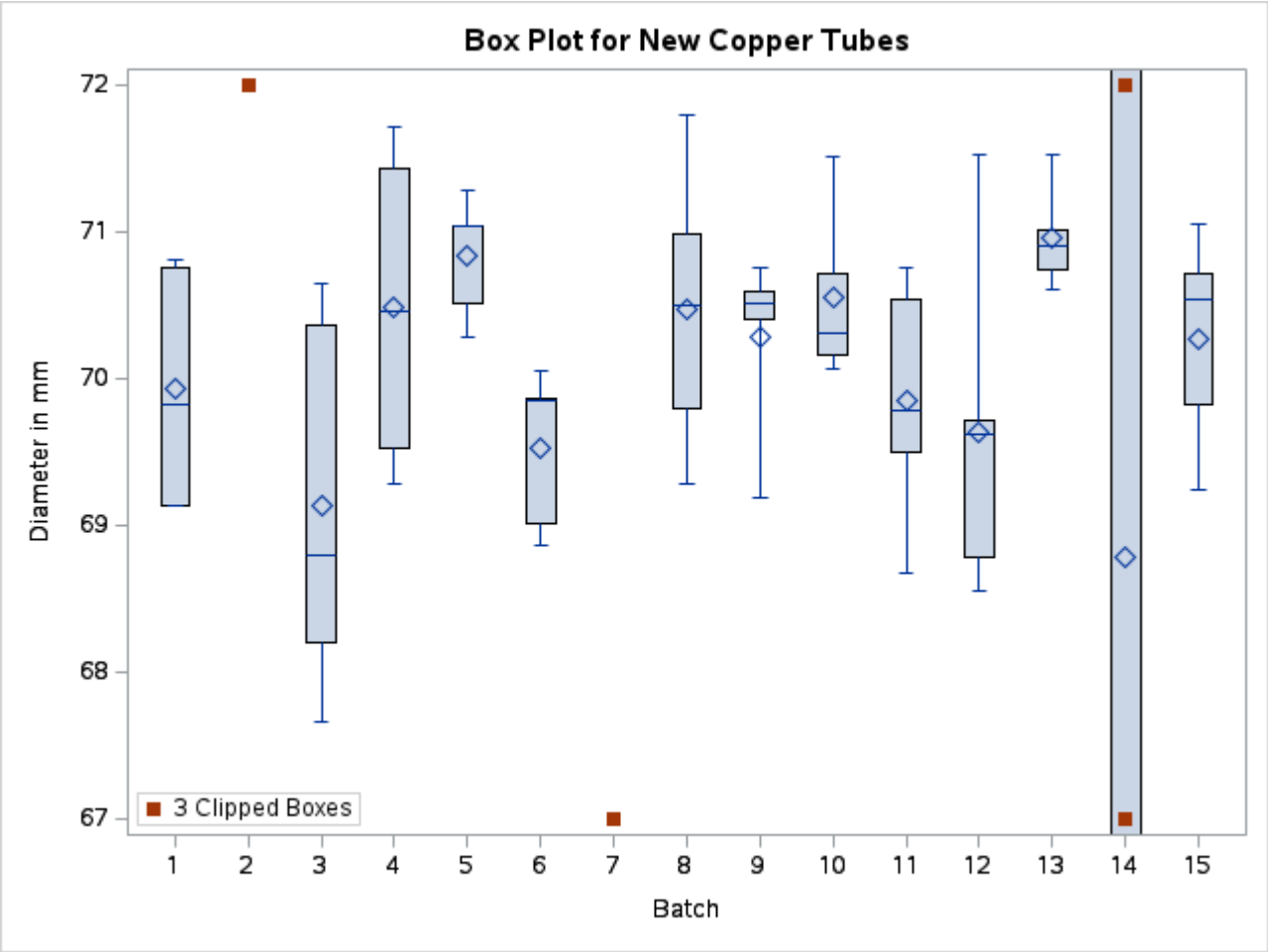

For more information about clipping options, see the appropriate entries in the section "PLOT Statement Options" on page 1442.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

The appearance of a box plot produced by using ODS Graphics is determined by the style associated with the ODS destination where the graph is produced. PLOT statement options used to control the appearance of traditional graphs are ignored for ODS Graphics output.

When producing ODS graphical displays, the PLOT statement assigns a name to each graph it creates. You can use this name to reference the graph when using ODS. The name is listed in Table 32.10.

Table 32.10 Graphs Produced by PROC BOXPLOT

ODS Graph Name	Plot Description
Boxplot	Box-and-whiskers plots for groups

Examples: BOXPLOT Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

This section provides advanced examples of the PLOT statement.

Example 32.1: Displaying Summary Statistics in a Box Plot

(View the complete code for this example (boxex1.sas) in the example repository.)

This example demonstrates how you can use the INSET and INSETGROUP statements to include tables of summary statistics in your box plots. The following statements produce a box plot of the Turbine data set from the section "Getting Started: BOXPLOT Procedure" on page 1425, augmented with insets that contains summary statistics:

```
ods graphics off;
symbol height=3 pct;
title 'Box Plot for Power Output';
proc boxplot data=Turbine;
   plot KWatts*Day;
   inset min mean max stddev /
      header = 'Overall Statistics'
      pos    = tm;
   insetgroup min max /
      header = 'Extremes by Day';
run;
```

The INSET statement produces an inset of overall summary statistics. The keywords listed before the slash (/) request the minimum, mean, maximum, and standard deviation computed over all days. The POS=TM option places the inset in the top margin of the plot.

The INSETGROUP statement produces an inset that contains statistics calculated for each group separately. The MIN and MAX keywords request the minimum and maximum observations from each day, respectively.

The resulting plot is shown in Output 32.1.1.

Output 32.1.1 Box Plot with Insets

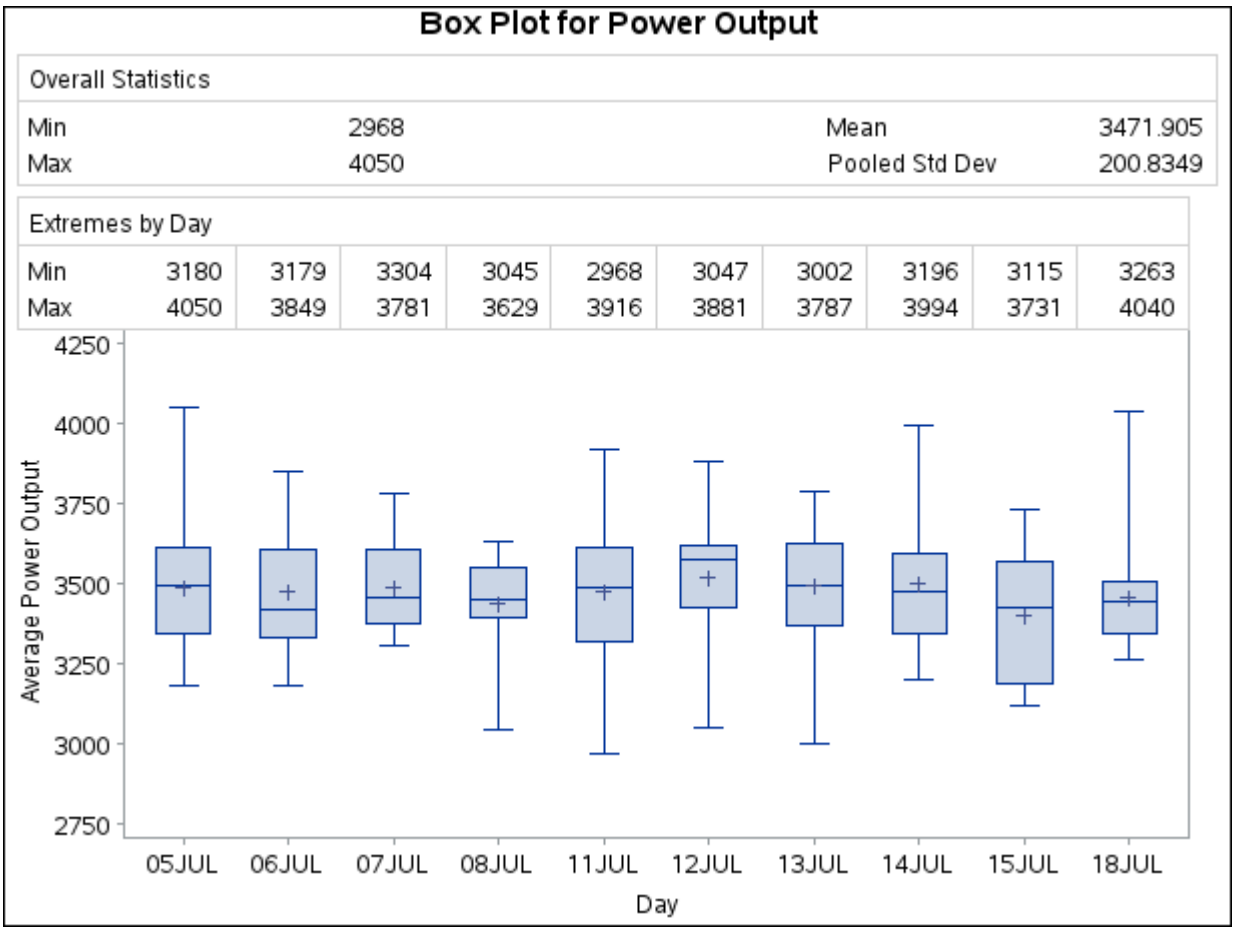

Example 32.2: Using Box Plots to Compare Groups

(View the complete code for this example (boxex2.sas) in the example repository.)

In this example a box plot is used to compare the delay times of airline flights during the Christmas holidays with the delay times prior to the holiday period. The following statements create a data set named Times with the delay times in minutes for 25 flights each day. When a flight is canceled, the delay is recorded as a missing value.

```
data Times;
   informat Day date7. ;
   format   Day date7. ;
   input Day @ ;
   do Flight=1 to 25;
      input Delay @ ;
      output;
   end;
   datalines;
16DEC88   4  12   2   2  18   5   6  21   0   0
          0  14   3   .   2   3   5   0   6  19
```

```
          7   4   9   5  10
17DEC88   1  10   3   3   0   1   5   0   .   .
          1   5   7   1   7   2   2  16   2   1
          3   1  31   5   0
18DEC88   7   8   4   2   3   2   7   6  11   3
          2   7   0   1  10   2   3  12   8   6
          2   7   2   4   5
19DEC88  15   6   9   0  15   7   1   1   0   2
          5   6   5  14   7  20   8   1  14   3
         10   0   1  11   7
20DEC88   2   1   0   4   4   6   2   2   1   4
          1  11   .   1   0   6   5   5   4   2
          2   6   6   4   0
21DEC88   2   6   6   2   7   7   5   2   5   0
          9   2   4   2   5   1   4   7   5   6
          5   0   4  36  28
22DEC88   3   7  22   1  11  11  39  46   7  33
         19  21   1   3  43  23   9   0  17  35
         50   0   2   1   0
23DEC88   6  11   8  35  36  19  21   .   .   4
          6  63  35   3  12  34   9   0  46   0
          0  36   3   0  14
24DEC88  13   2  10   4   5  22  21  44  66  13
          8   3   4  27   2  12  17  22  19  36
          9  72   2   4   4
25DEC88   4  33  35   0  11  11  10  28  34   3
         24   6  17   0   8   5   7  19   9   7
         21  17  17   2   6
26DEC88   3   8   8   2   7   7   8   2   5   9
          2   8   2  10  16   9   5  14  15   1
         12   2   2  14  18
;
```

In the following statements, the MEANS procedure is used to count the number of canceled flights for each day. This information is then added to the data set Times.

```
proc means data=Times noprint;
   var Delay;
   by Day;
   output out=Cancel nmiss=ncancel;
run;

data Times;
   merge Times Cancel;
   by Day;
run;
```

The following statements create a data set named Weather that contains information about possible causes for delays, and then merge this data set with the data set Times:

```
data Weather;
   informat Day date7. ;
   format   Day date7. ;
   length Reason $ 16 ;
   input Day Flight Reason & ;
```

```
   datalines;
16DEC88  8   Fog
17DEC88  18  Snow Storm
17DEC88  23  Sleet
21DEC88  24  Rain
21DEC88  25  Rain
22DEC88  7   Mechanical
22DEC88  15  Late Arrival
24DEC88  9   Late Arrival
24DEC88  22  Late Arrival
;

data Times;
   merge Times Weather;
   by Day Flight;
run;
```

The following statements create a box plot for the complete set of data:

```
ods graphics off;
symbol1 value=dot            c=salmon h=3 pct;
symbol2 value=squarefilled   c=vigb   h=3 pct;
symbol3 value=trianglefilled c=vig    h=3 pct;
title 'Box Plot for Airline Delays';
proc boxplot data=Times;
   plot Delay*Day = ncancel /
       nohlabel
       symbollegend = legend1;
   legend1 label = ('Cancellations:');
   label Delay = 'Delay in Minutes';
run;
goptions reset=symbol;
```

The level of the *symbol variable* ncancel determines the symbol marker for each group mean, and the SYMBOLLEGEND= option controls the appearance of the legend for the symbols. The NOHLABEL option suppresses the horizontal axis label. The resulting box plot is shown in Output 32.2.1.

Output 32.2.1 Box Plot for Airline Data

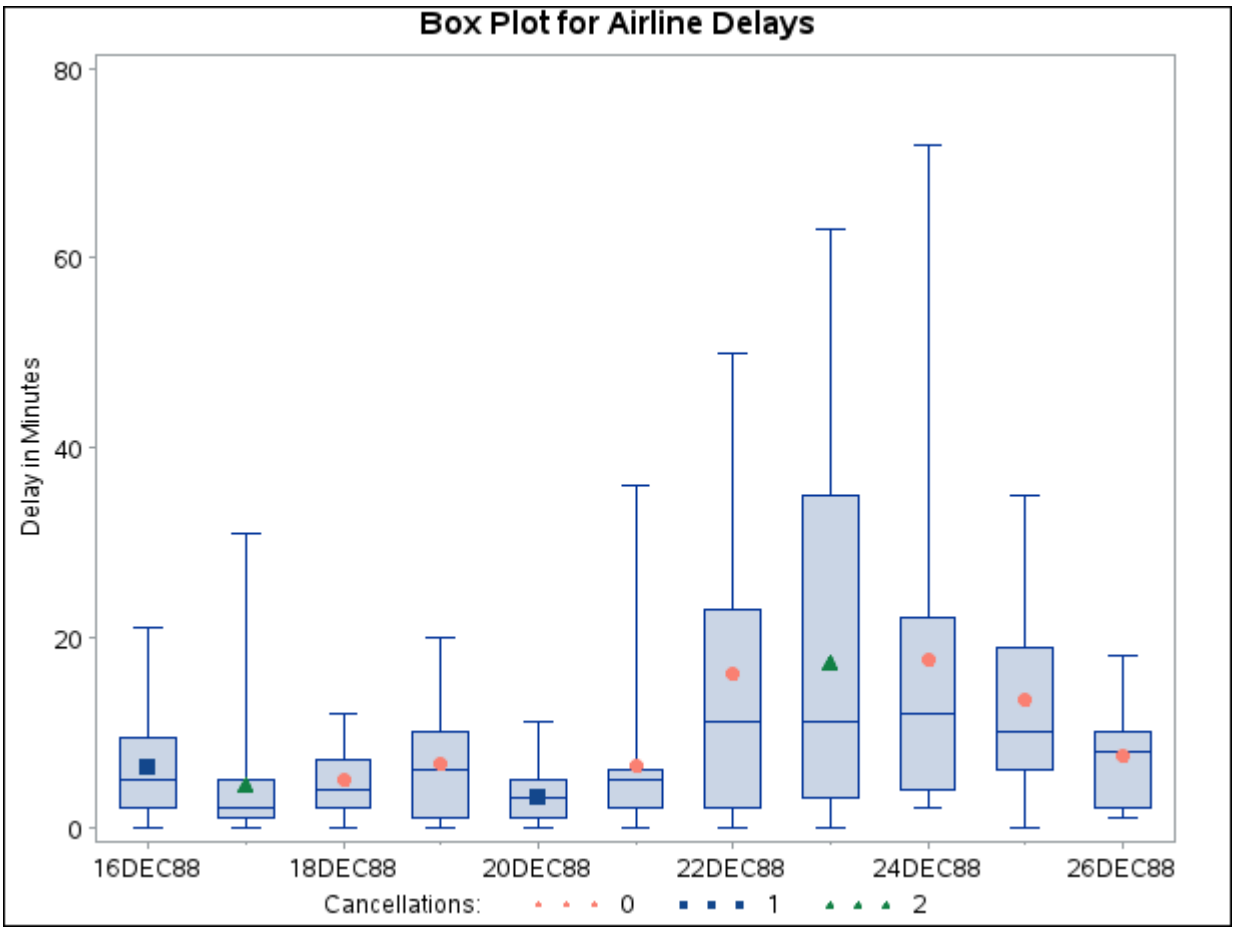

The delay distributions from December 22 through December 25 are drastically different from the delay distributions during the pre-holiday period. Both the mean delay and the variability of the delays are much greater during the holiday period.

Example 32.3: Creating Various Styles of Box-and-Whiskers Plots

(View the complete code for this example (boxex2.sas) in the example repository.)

This example uses the flight delay data of the preceding example to illustrate how you can create box plots with various styles of box-and-whiskers plots. The following statements create a plot that displays skeletal box-and-whiskers plots:

```
ods graphics on;
title 'Analysis of Airline Departure Delays';
title2 'BOXSTYLE=SKELETAL';
proc boxplot data=Times;
   plot Delay*Day /
      boxstyle = skeletal
      odstitle = title
      nohlabel;
```

```
   label Delay = 'Delay in Minutes';
run;
```

In a skeletal box-and-whiskers plot, the whiskers are drawn from the quartiles to the extreme values of the group. The skeletal box plot is the default style, so you can also produce a skeletal box plot by omitting the BOXSTYLE= option. Output 32.3.1 shows the skeletal box plot.

Output 32.3.1 BOXSTYLE=SKELETAL

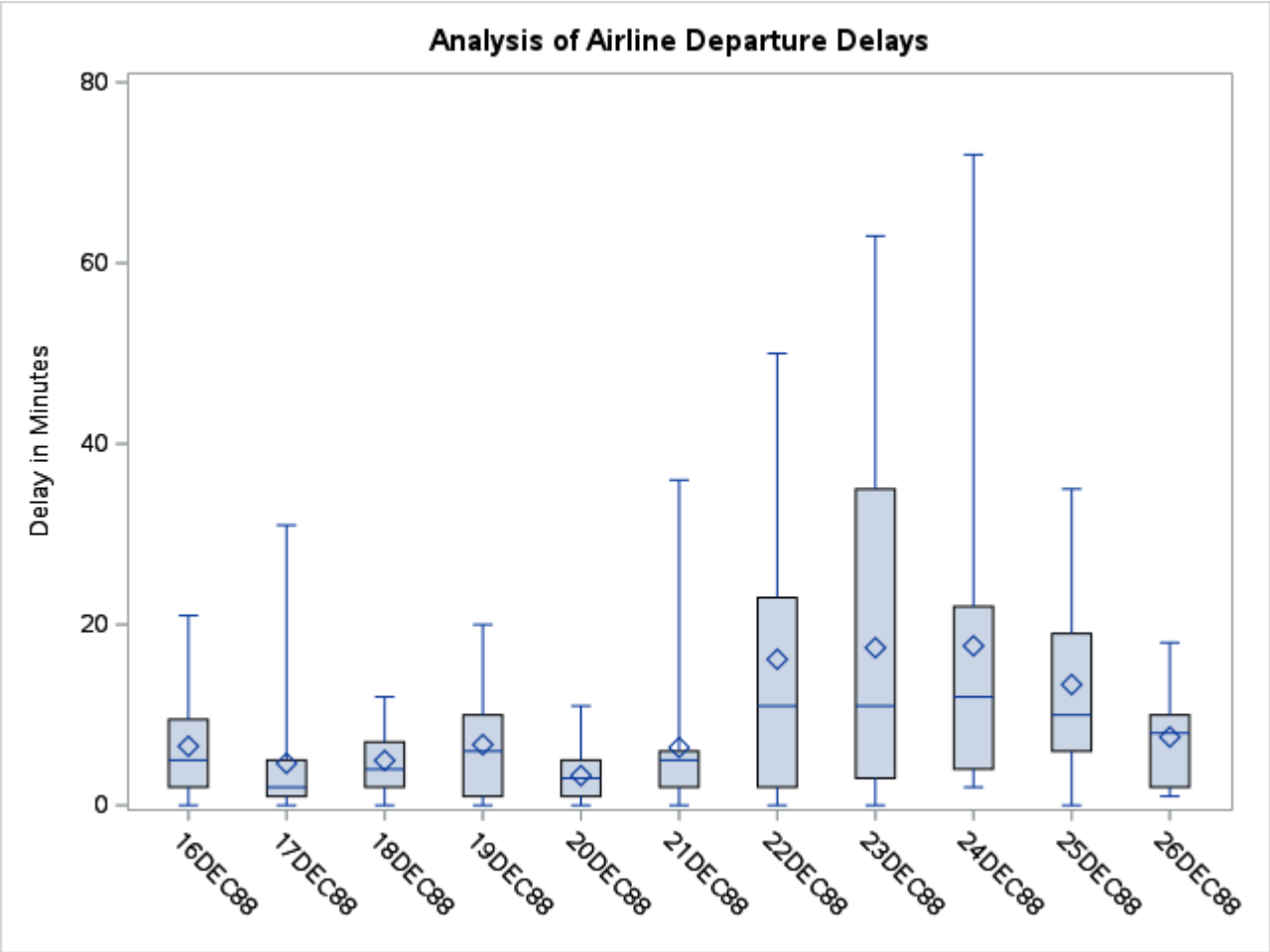

The following statements request a schematic box:

```
title 'Analysis of Airline Departure Delays';
title2 'BOXSTYLE=SCHEMATIC';
proc boxplot data=Times;
   plot Delay*Day /
      boxstyle = schematic
      odstitle = title
      nohlabel;
   label Delay = 'Delay in Minutes';
run;
```

When you specify BOXSTYLE=SCHEMATIC, the whiskers are drawn to the most extreme points in the group that lie within or equal to the *fences*. The *upper fence* is defined as the third quartile (represented by

the upper edge of the box) plus 1.5 times the interquartile range (IQR). The *lower fence* is defined as the first quartile (represented by the lower edge of the box) minus 1.5 times the interquartile range. Observations outside the fences are identified with a special symbol. The default symbol is a square, and you can specify the shape and color for this symbol with the IDSYMBOL= and IDCOLOR= options. Serifs are added to the whiskers by default. For further details, see the entry for the BOXSTYLE= option. The plot is shown in Output 32.3.2.

Output 32.3.2 BOXSTYLE=SCHEMATIC

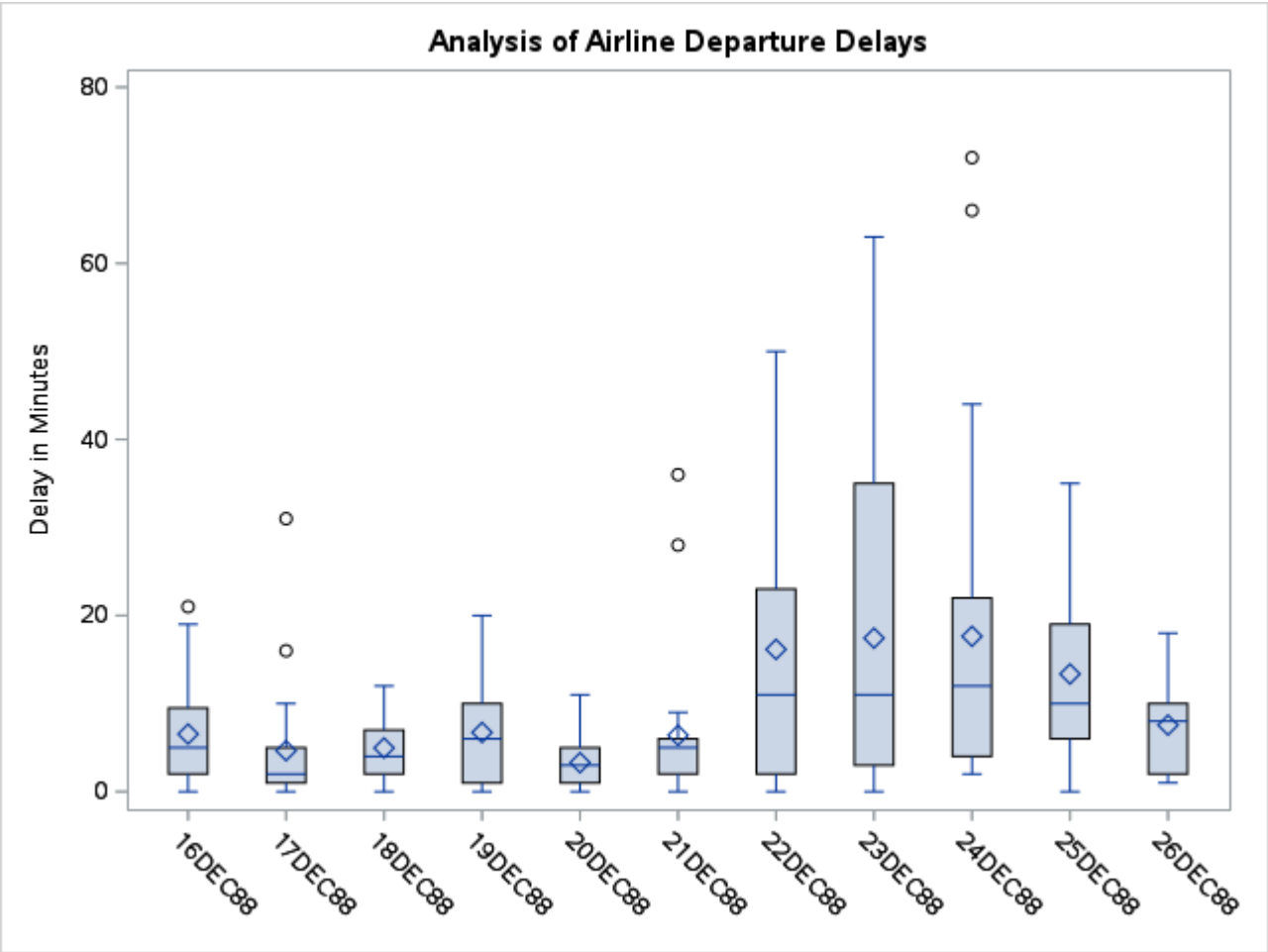

The following statements create a schematic box plot in which the observations outside the fences are labeled:

```
title 'Analysis of Airline Departure Delays';
title2 'BOXSTYLE=SCHEMATICID';
proc boxplot data=Times;
   plot Delay*Day /
      boxstyle  = schematicid
      odstitle  = title
      odstitle2 = title2
      nohlabel;
   id Reason;
   label Delay = 'Delay in Minutes';
run;
```

If you specify BOXSTYLE=SCHEMATICID, schematic box-and-whiskers plots are created and the value of the first ID variable (in this case, Reason) is used to label each observation outside the fences. The box plot is shown in Output 32.3.3.

Output 32.3.3 BOXSTYLE=SCHEMATICID

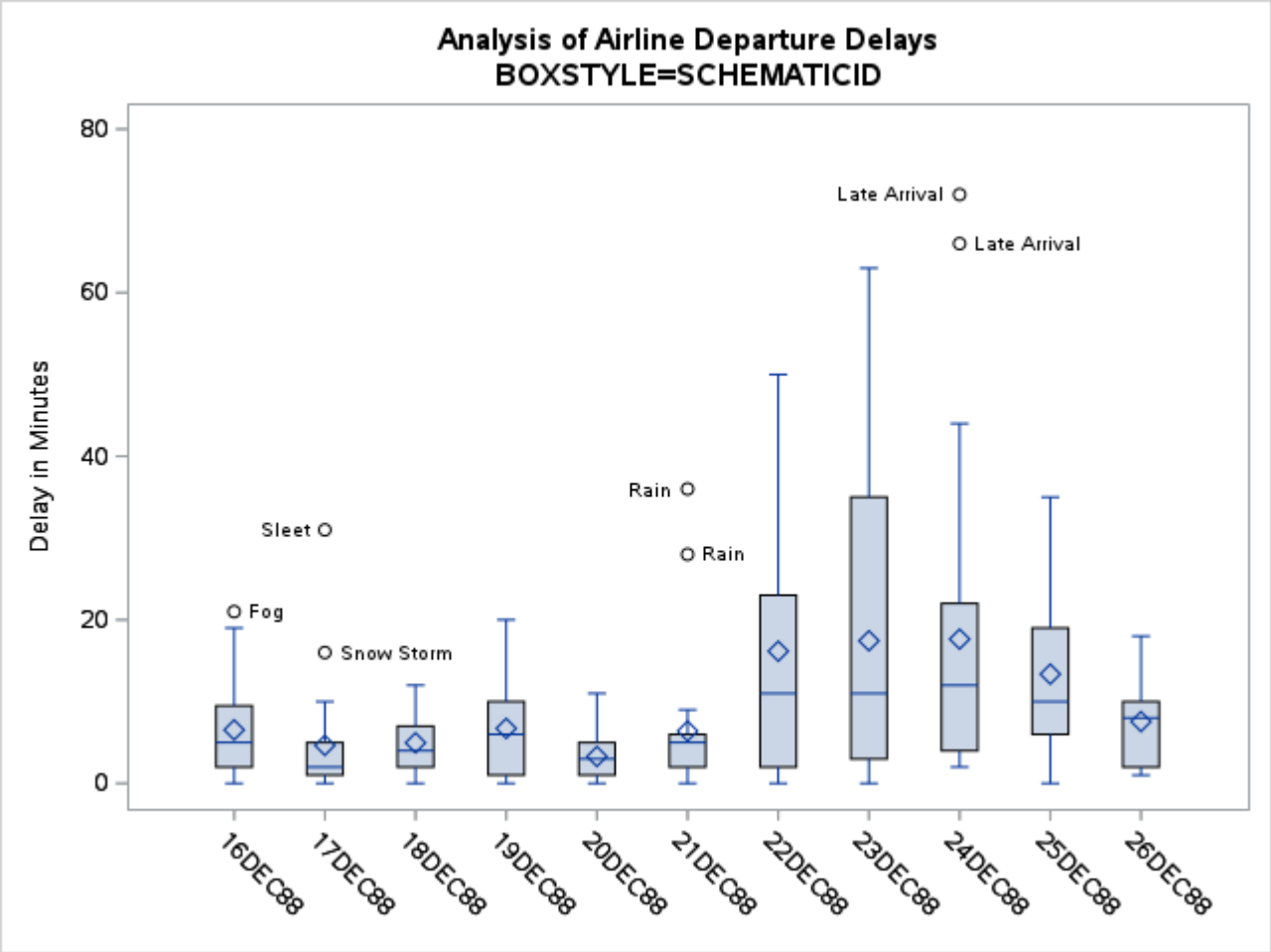

The following statements create a box plot with schematic box-and-whiskers plots in which only the extreme observations outside the fences are labeled:

```
title 'Analysis of Airline Departure Delays';
title2 'BOXSTYLE=SCHEMATICIDFAR';
proc boxplot data=Times;
   plot Delay*Day /
      boxstyle = schematicidfar
      odstitle  = title
      odstitle2 = title2
      nohlabel;
   id Reason;
   label Delay = 'Delay in Minutes';
run;
```

If you specify BOXSTYLE=SCHEMATICIDFAR, the value of the first ID variable is used to label each observation outside the lower and upper *far fences*. The lower and upper far fences are located 3×IQR below the 25th percentile and 3×IQR above the 75th percentile, respectively. Observations between the fences and the far fences are identified with a symbol but are not labeled. The box plot is shown in Output 32.3.4.

Output 32.3.4 BOXSTYLE=SCHEMATICIDFAR

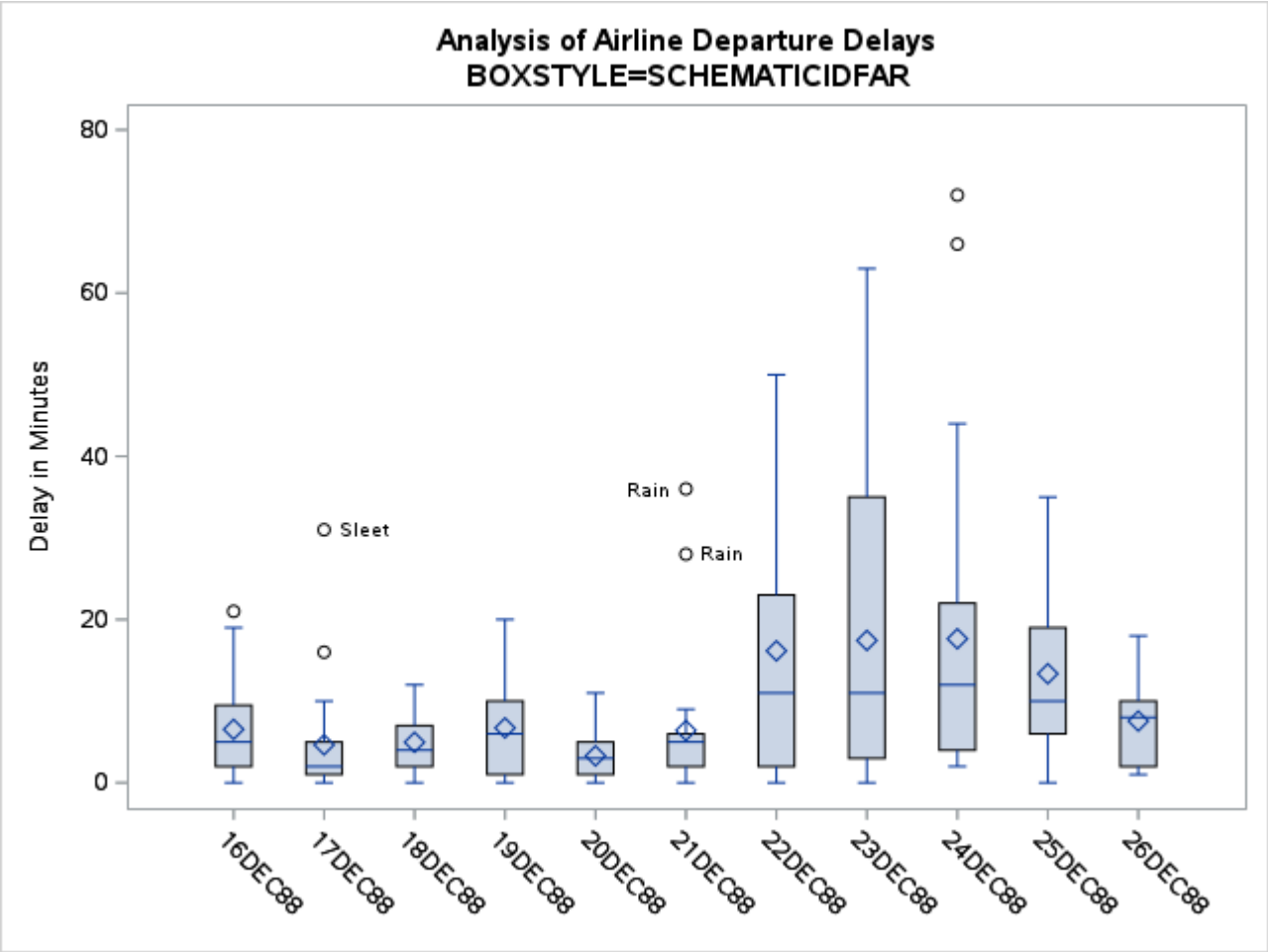

Example 32.4: Creating Notched Box-and-Whiskers Plots

(View the complete code for this example (boxex2.sas) in the example repository.)

The following statements use the flight delay data of Example 32.2 to create box-and-whiskers plots with notches:

```
proc boxplot data=Times;
   plot Delay*Day /
      boxstyle  = schematicid
      odstitle  = title
      odstitle2 = title2
      nohlabel
```

```
      notches;
   id Reason;
   label Delay = 'Delay in Minutes';
run;
```

The notches, requested with the NOTCHES option, measure the significance of the difference between two medians. The medians of two box plots are significantly different at approximately the 0.95 confidence level if the corresponding notches do not overlap. For example, in Output 32.4.1, the median for December 20 is significantly different from the median for December 24.

Output 32.4.1 Notched Side-by-Side Box-and-Whiskers Plots

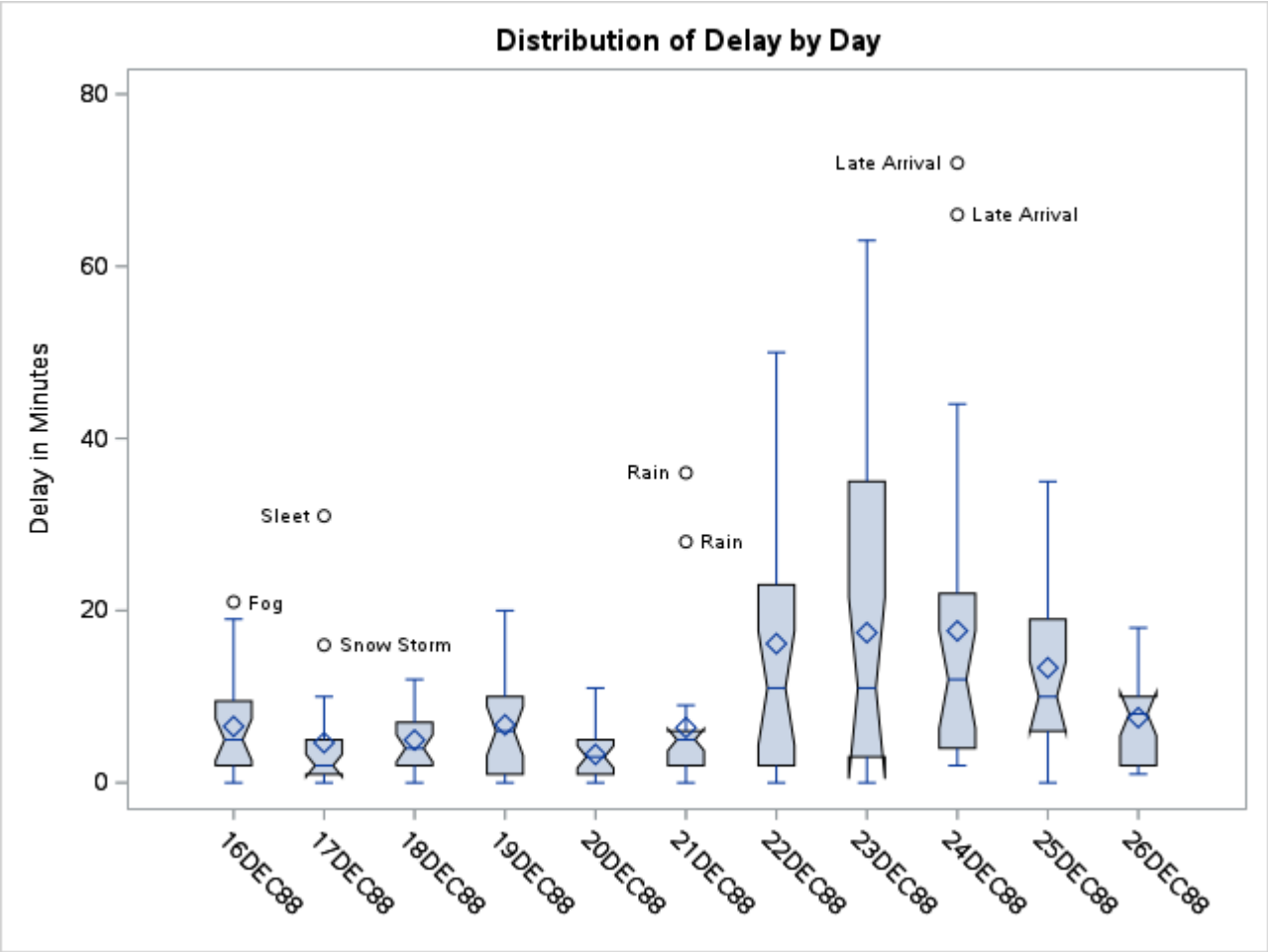

Example 32.5: Creating Box-and-Whiskers Plots with Varying Widths

(View the complete code for this example (boxex2.sas) in the example repository.)

This example shows how to create a box plot with box-and-whiskers plots whose widths vary proportionately with the group size. The following statements create a SAS data set named Times2 that contains flight departure delays (in minutes) recorded daily for eight consecutive days:

```
data Times2;
   label Delay = 'Delay in Minutes';
   informat Day date7. ;
   format   Day date7. ;
   input Day @ ;
   do Flight=1 to 25;
      input Delay @ ;
      output;
   end;
   datalines;
01MAR90   12  4   2   2  15   8   0  11   0   0
           0 12   3   .   2   3   5   0   6  25
           7  4   9   5  10
02MAR90   1   .   3   .   0   1   5   0   .   .
           1  5   7   .   7   2   2  16   2   1
           3  1  31   .   0
03MAR90   6   8   4   2   3   2   7   6  11   3
           2  7   0   1  10   2   5  12   8   6
           2  7   2   4   5
04MAR90  12   6   9   0  15   7   1   1   0   2
           5  6   5  14   7  21   8   1  14   3
          11  0   1  11   7
05MAR90   2   1   0   4   .   6   2   2   1   4
           1 11   .   1   0   .   5   5   .   2
           3  6   6   4   0
06MAR90   8   6   5   2   9   7   4   2   5   1
           2  2   4   2   5   1   3   9   7   8
           1  0   4  26  27
07MAR90   9   6   6   2   7   8   .   .  10   8
           0  2   4   3   .   .   .   7   .   6
           4  0   .   .   .
08MAR90   1   6   6   2   8   8   5   3   5   0
           8  2   4   2   5   1   6   4   5  10
           2  0   4   1   1
;
```

The following statements create a box plot with varying box widths:

```
title 'Analysis of Airline Departure Delays';
title2 'Using the BOXWIDTHSCALE= Option';
proc boxplot data=Times2;
   plot Delay*Day /
      boxstyle      = schematic
      odstitle      = title
      odstitle2     = title2
```

```
        boxwidthscale = 1
        nohlabel
        bwslegend;
run;
```

The BOXWIDTHSCALE=*value* option specifies that the widths of the box-and-whiskers plots vary in proportion to a particular function of the group size *n*. The function is determined by *value* and is identified on the box plot with a legend if the BWSLEGEND option is specified. The BOXWIDTHSCALE= option is useful in situations where the group sizes vary widely.

Output 32.5.1 shows the resulting box plot.

Output 32.5.1 Box Plot with Box-and-Whiskers Plots of Varying Widths

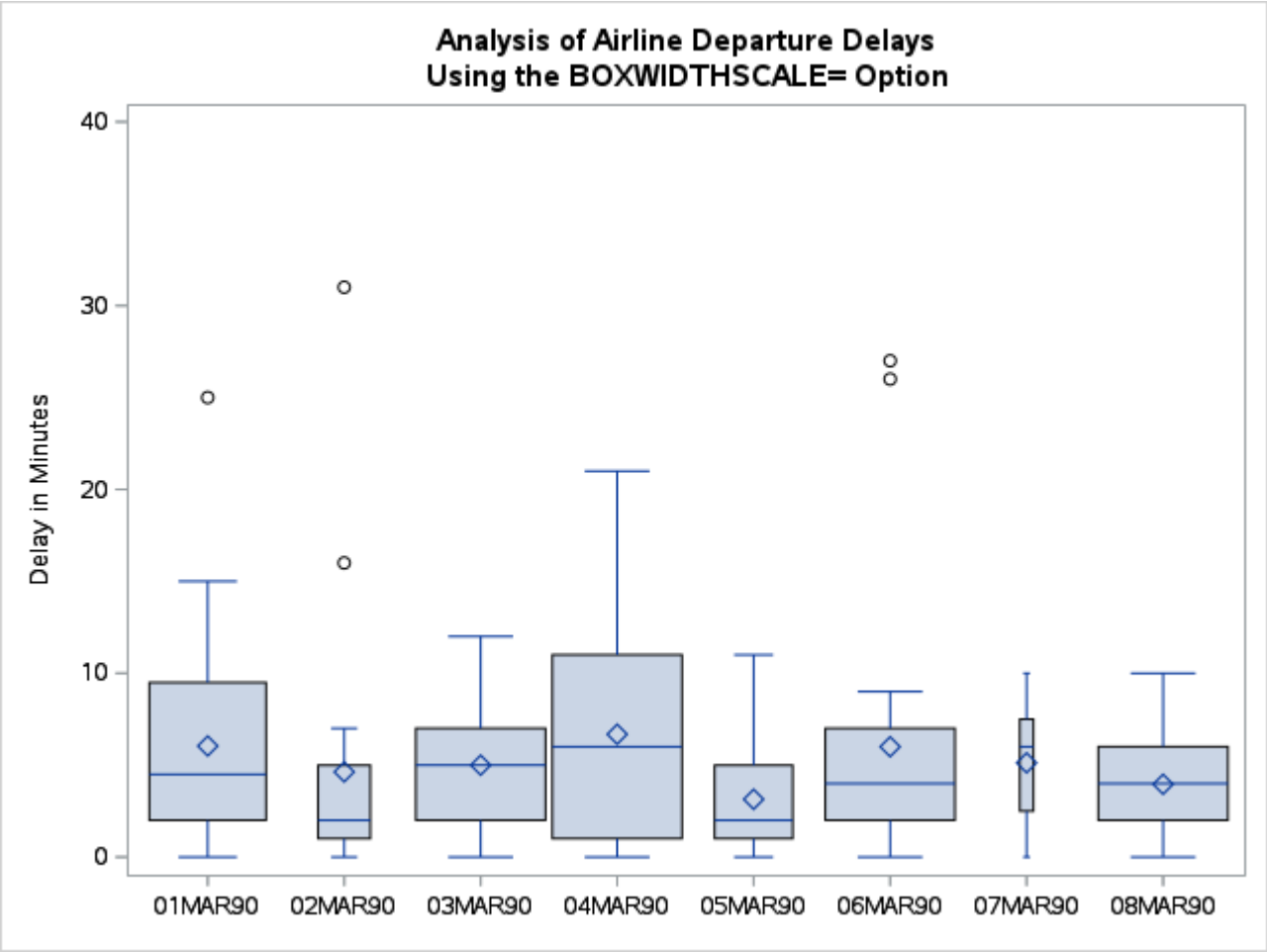

Example 32.6: Creating Horizontal Box-and-Whiskers Plots

(View the complete code for this example (boxex2.sas) in the example repository.)

The following statements use the HORIZONTAL option, which is supported only for ODS Graphics output, to produce a horizontal box plot:

```
proc boxplot data=Times;
   plot Delay*Day /
      boxstyle = schematic
      horizontal;
   label Delay = 'Delay in Minutes';
run;
```

The horizontal box plot is shown in Output 32.6.1.

Output 32.6.1 Horizontal Box Plot

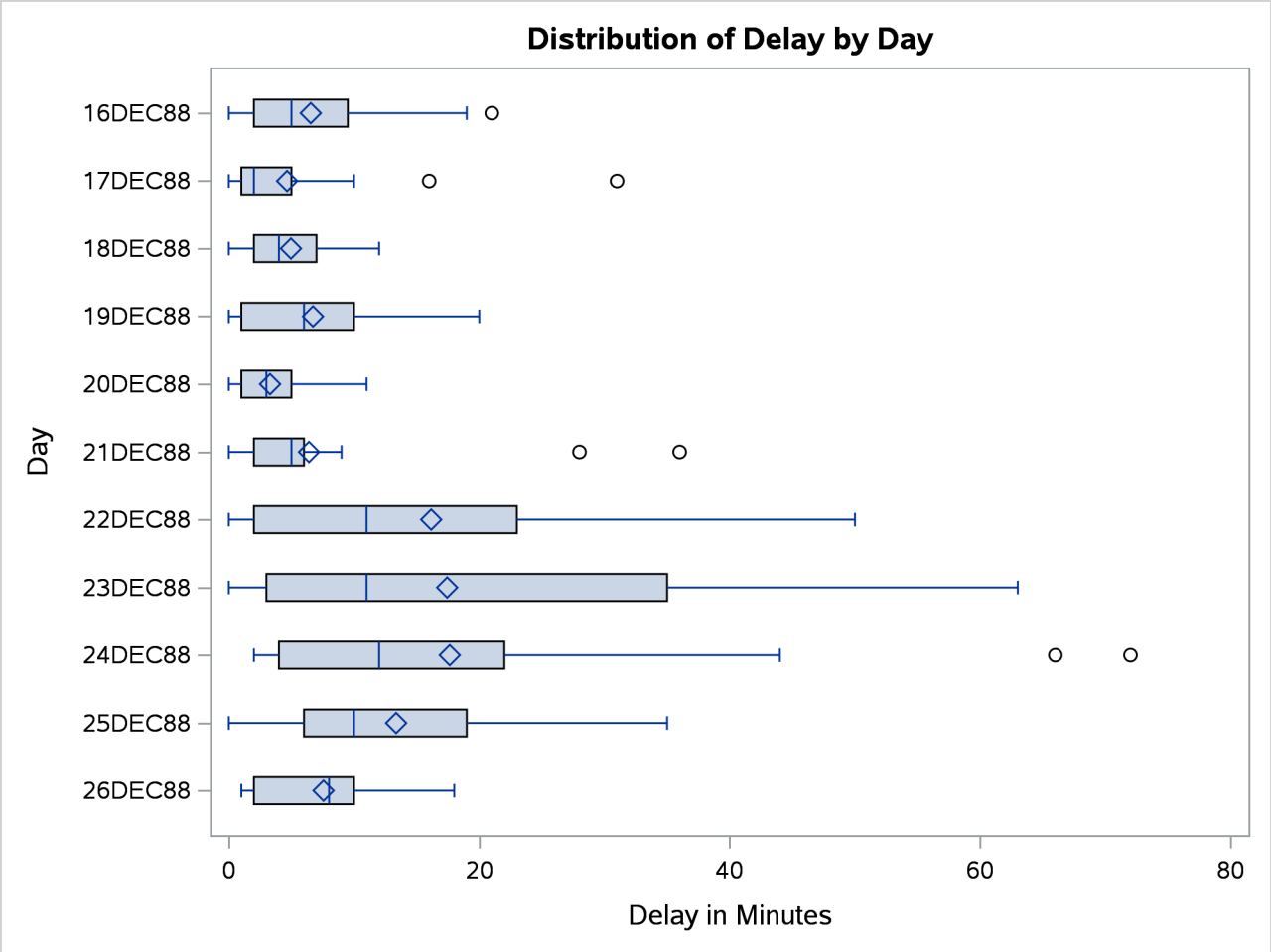

References

McGill, R., Tukey, J. W., and Larsen, W. A. (1978). "Variations of Box Plots." *American Statistician* 32:12–16.

Tukey, J. W. (1977). *Exploratory Data Analysis*. Reading, MA: Addison-Wesley.

Chapter 33
The CALIS Procedure

Contents

Overview: CALIS Procedure . . . **1506**
A Guide to the PROC CALIS Documentation . . . 1510
Getting Started: CALIS Procedure . . . **1521**
A Structural Equation Example . . . 1521
A Factor Model Example . . . 1527
Direct Covariance Structures Analysis . . . 1529
Which Modeling Language? . . . 1530
Syntax: CALIS Procedure . . . **1532**
Classes of Statements in PROC CALIS . . . 1533
Single-Group Analysis Syntax . . . 1536
Multiple-Group Multiple-Model Analysis Syntax . . . 1536
PROC CALIS Statement . . . 1537
AUXILIARY Statement . . . 1581
BOUNDS Statement . . . 1582
BY Statement . . . 1583
COSAN Statement . . . 1583
COV Statement . . . 1593
DETERM Statement . . . 1598
EFFPART Statement . . . 1599
FACTOR Statement . . . 1600
FITINDEX Statement . . . 1610
FREQ Statement . . . 1614
GROUP Statement . . . 1614
LINCON Statement . . . 1617
LINEQS Statement . . . 1618
LISMOD Statement . . . 1624
LMTESTS Statement . . . 1628
MATRIX Statement . . . 1638
MEAN Statement . . . 1651
MODEL Statement . . . 1654
MSTRUCT Statement . . . 1656
NLINCON Statement . . . 1659
NLOPTIONS Statement . . . 1659
OUTFILES Statement . . . 1660
PARAMETERS Statement . . . 1662
PARTIAL Statement . . . 1665

PATH Statement 1665
PATHDIAGRAM Statement 1675
PCOV Statement 1687
PVAR Statement 1690
RAM Statement 1692
REFMODEL Statement 1698
RENAMEPARM Statement 1700
SAS Programming Statements 1701
SIMTESTS Statement 1701
STD Statement 1702
STRUCTEQ Statement 1702
TESTFUNC Statement 1702
VAR Statement 1703
VARIANCE Statement 1706
VARNAMES Statement 1710
WEIGHT Statement 1711
Details: CALIS Procedure **1711**
Input Data Sets 1711
Output Data Sets 1715
Default Analysis Type and Default Parameterization 1732
The COSAN Model 1734
The FACTOR Model 1738
The LINEQS Model 1745
The LISMOD Model and Submodels 1752
The MSTRUCT Model 1761
The PATH Model 1763
The RAM Model 1769
Naming Variables and Parameters 1777
Setting Constraints on Parameters 1778
Automatic Variable Selection 1783
Path Diagrams: Layout Algorithms, Default Settings, and Customization 1784
Estimation Criteria 1805
Relationships among Estimation Criteria 1816
Gradient, Hessian, Information Matrix, and Approximate Standard Errors 1818
Counting the Degrees of Freedom 1822
Assessment of Fit 1824
Case-Level Residuals, Outliers, Leverage Observations, and Residual Diagnostics . . 1837
Latent Variable Scores 1841
Total, Direct, and Indirect Effects 1844
Standardized Solutions 1847
Modification Indices 1848
Missing Values and the Analysis of Missing Patterns 1850
Measures of Multivariate Kurtosis 1851
Initial Estimates 1853

Use of Optimization Techniques 1854
Computational Problems 1862
Displayed Output 1865
ODS Table Names 1870
ODS Graphics 1884
Examples: CALIS Procedure **1886**
Example 33.1: Estimating Covariances and Correlations 1886
Example 33.2: Estimating Covariances and Means Simultaneously 1891
Example 33.3: Testing Uncorrelatedness of Variables 1893
Example 33.4: Testing Covariance Patterns 1895
Example 33.5: Testing Some Standard Covariance Pattern Hypotheses 1897
Example 33.6: Linear Regression Model 1900
Example 33.7: Multivariate Regression Models 1904
Example 33.8: Measurement Error Models 1921
Example 33.9: Testing Specific Measurement Error Models 1927
Example 33.10: Measurement Error Models with Multiple Predictors 1933
Example 33.11: Measurement Error Models Specified as Linear Equations 1938
Example 33.12: Confirmatory Factor Models 1944
Example 33.13: Confirmatory Factor Models: Some Variations 1956
Example 33.14: Residual Diagnostics and Robust Estimation 1965
Example 33.15: The Full Information Maximum Likelihood Method 1978
Example 33.16: Comparing the ML and FIML Estimation 1987
Example 33.17: Path Analysis: Stability of Alienation 1992
Example 33.18: Simultaneous Equations with Mean Structures and Reciprocal Paths . 2006
Example 33.19: Fitting Direct Covariance Structures 2013
Example 33.20: Confirmatory Factor Analysis: Cognitive Abilities 2016
Example 33.21: Testing Equality of Two Covariance Matrices Using a Multiple-Group Analysis 2027
Example 33.22: Testing Equality of Covariance and Mean Matrices between Independent Groups 2032
Example 33.23: Illustrating Various General Modeling Languages 2055
Example 33.24: Testing Competing Path Models for the Career Aspiration Data . . . 2065
Example 33.25: Fitting a Latent Growth Curve Model 2077
Example 33.26: Higher-Order and Hierarchical Factor Models 2082
Example 33.27: Linear Relations among Factor Loadings 2096
Example 33.28: Multiple-Group Model for Purchasing Behavior 2104
Example 33.29: Fitting the RAM and EQS Models by the COSAN Modeling Language 2127
Example 33.30: Second-Order Confirmatory Factor Analysis 2141
Example 33.31: Linear Relations among Factor Loadings: COSAN Model Specification 2146
Example 33.32: Ordinal Relations among Factor Loadings 2151
Example 33.33: Longitudinal Factor Analysis 2154
References **2160**

Overview: CALIS Procedure

Structural equation modeling is an important statistical tool in social and behavioral sciences. Structural equations express relationships among a system of variables that can be either observed variables (manifest variables) or unobserved hypothetical variables (latent variables). For an introduction to latent variable models, see Loehlin (2004); Bollen (1989b); Everitt (1984), or Long (1983); and for manifest variables with measurement errors, see Fuller (1987).

In structural models, as opposed to functional models, all variables are taken to be random rather than having fixed levels. For maximum likelihood (ML, the default) and generalized least squares (GLS) estimation in PROC CALIS, the random variables are assumed to have an approximately multivariate normal distribution. Nonnormality, especially high kurtosis, can produce poor estimates and grossly incorrect standard errors and hypothesis tests, even in large samples. Consequently, the assumption of normality is much more important than in models with nonstochastic exogenous variables. You should remove outliers and consider transformations of nonnormal variables before using PROC CALIS with maximum likelihood (default) or generalized least squares estimation.

Alternatively, several approaches are available to deal with the nonnormality issue. If the number of observations is sufficiently large, you can use Browne's asymptotically distribution-free (ADF; Browne 1982) estimation method. However, there is no definite guideline for how large a sample needs to be in order to use ADF estimation. Simulation studies usually show that several thousand observations might be required. If you use maximum likelihood estimation, the Satorra-Bentler scaled chi-square test statistics and the associated sandwich-type standard error estimates (Satorra and Bentler 1994) might be a viable solution even if your data are not normal. In PROC CALIS, you can apply the Satorra-Bentler method by using the METHOD=MLSB option. In the psychometric literature, the Satorra-Bentler method is sometimes referred to as robust ML. However, PROC CALIS reserves the term "robust" for another estimation procedure. When you use METHOD=ML (default) with the ROBUST option, the iterative estimation downweights model outliers so that they have less impact on the estimation. Henceforth, this is the robust ML method that the CALIS procedure refers to.

When your data contain missing values, you might consider the full information maximum likelihood (FIML) method. Assuming that the data are missing at random (MAR; see Rubin 1976), the FIML method uses all available data to perform estimation within the maximum likelihood framework. In contrast, all other estimation methods perform listwise deletion of incomplete observations before estimation. Hence, when the data are scarce and you need to use as much information as possible, FIML estimation might provide a viable solution.

You can use the CALIS procedure to estimate parameters and test hypotheses for constrained and unconstrained problems in various situations, including but not limited to the following:

- exploratory and confirmatory factor analysis of any order
- linear measurement-error models or regression with errors in variables
- multiple and multivariate linear regression
- multiple-group structural equation modeling with mean and covariance structures
- path analysis and causal modeling

- latent curve modeling
- simultaneous equation models with reciprocal causation
- structured covariance and mean matrices in various forms

To specify models in PROC CALIS, you can use a variety of modeling languages:

- COSAN—a generalized version of the COSAN program (McDonald 1978, 1980), uses general mean and covariance structures to define models
- FACTOR—supports the input of latent factor and observed variable relations
- LINEQS—like the EQS program (Bentler 1995), uses equations to describe variable relationships
- LISMOD—uses LISREL (Jöreskog and Sörbom 1985) model matrices to define models
- MSTRUCT—supports direct parameterizations in the mean and covariance matrices
- PATH—provides an intuitive causal path specification interface
- RAM—uses the formulation of the reticular action model (McArdle and McDonald 1984) to define models
- REFMODEL—provides a quick way for model referencing and respecification

Various modeling languages are provided to suit a wide range of researchers' background and modeling philosophy. However, statistical situations might arise where one modeling language is more convenient than the others. This will be discussed in the section "Which Modeling Language?" on page 1530.

In addition to basic model specification, you can set various parameter constraints in PROC CALIS. Equality constraints on parameters can be achieved by simply giving the same parameter names in different parts of the model. Boundary (BOUNDS statement), linear (LINCON statement), and nonlinear (NLINCON statement) constraints are supported as well. If parameters in the model are dependent on additional parameters, you can define the dependence by using the PARAMETERS and the SAS programming statements.

Before the data are analyzed, researchers might be interested in studying some statistical properties of the data. PROC CALIS can provide the following statistical summary of the data:

- covariance and mean matrices and their properties
- descriptive statistics like means, standard deviations, univariate skewness, and kurtosis measures
- multivariate measures of kurtosis
- coverage of covariances and means, missing patterns summary, and means of the missing patterns when the FIML estimation is used
- weight matrix and its descriptive properties
- robust covariance and mean matrices with the robust methods

After a model is fitted and accepted by the researcher, PROC CALIS can provide the following supplementary statistical analysis:

- computing squared multiple correlations and determination coefficients
- direct and indirect effects partitioning with standard error estimates
- model modification tests such as Lagrange multiplier and Wald tests
- computing fit summary indices
- computing predicted moments of the model
- residual analysis on the covariances and means
- case-level residual diagnostics with graphical plots
- factor rotations
- standardized solutions with standard errors
- testing parametric functions, individually or simultaneously

When fitting a model, you need to choose an estimation method. The following estimation methods are supported in the CALIS procedure:

- diagonally weighted least squares (DWLS, with optional weight matrix input)
- full information maximum likelihood (FIML, which can treat observations with random missing values)
- generalized least squares (GLS, with optional weight matrix input)
- maximum likelihood (ML, for multivariate normal data); this is the default method
- maximum likelihood with Satorra-Bentler scaled model fit chi-square statistic and sandwich-type standard error estimation (MLSB)
- robust estimation with maximum likelihood model evaluation (ROBUST option with METHOD=ML)
- unweighted least squares (ULS)
- weighted least squares or asymptotically distribution-free method (WLS or ADF, with optional weight matrix input)

Estimation methods implemented in PROC CALIS do not exhaust all alternatives in the field. For example, the partial least squares (PLS) method is not implemented. See the section "Estimation Criteria" on page 1805 for details about estimation criteria used in PROC CALIS. Note that there is a SAS/STAT procedure called PROC PLS, which employs the partial least squares technique but for a different class of models than those of PROC CALIS. For general path analysis with latent variables, consider using PROC CALIS.

All estimation methods need some starting values for the parameter estimates. You can provide starting values for any parameters. If there is any estimate without a starting value provided, PROC CALIS determines the starting value by using one or any combination of the following methods:

- approximate factor analysis
- default initial values
- instrumental variable method
- matching observed moments of exogenous variables
- McDonald's method (McDonald and Hartmann 1992)
- ordinary least squares estimation
- random number generation, if a seed is provided
- two-stage least squares estimation

Although no methods for initial estimates are completely foolproof, the initial estimation methods provided by PROC CALIS behave reasonably well in most common applications.

With initial estimates, PROC CALIS will iterate the solutions so as to achieve the optimum solution as defined by the estimation criterion. This is a process known as optimization. Because numerical problems can occur in any optimization process, the CALIS procedure offers several optimization algorithms so that you can choose alternative algorithms when the one being used fails. The following optimization algorithms are supported in PROC CALIS:

- Levenberg-Marquardt algorithm (Moré 1978)
- trust-region algorithm (Gay 1983)
- Newton-Raphson algorithm with line search
- ridge-stabilized Newton-Raphson algorithm
- various quasi-Newton and dual quasi-Newton algorithms: Broyden-Fletcher-Goldfarb-Shanno and Davidon-Fletcher-Powell, including a sequential quadratic programming algorithm for processing nonlinear equality and inequality constraints
- various conjugate gradient algorithms: automatic restart algorithm of Powell (1977), Fletcher-Reeves, Polak-Ribiere, and conjugate descent algorithm of Fletcher (1980)
- iteratively reweighted least squares for robust estimation

In addition to the ability to save output tables as data sets by using the ODS OUTPUT statement, PROC CALIS supports the following types of output data sets so that you can save your analysis results for later use:

- OUTEST= data sets for storing parameter estimates and their covariance estimates
- OUTFIT= data sets for storing fit indices and some pertinent modeling information
- OUTMODEL= data sets for storing model specifications and final estimates
- OUTSTAT= data sets for storing descriptive statistics, robust covariances and means, residuals, predicted moments, and latent variable scores regression coefficients

- OUTWGT= data sets for storing the weight matrices used in the modeling

The OUTEST=, OUTMODEL=, and OUTWGT= data sets can be used as input data sets for subsequent analyses. That is, in addition to the input data provided by the DATA= option, PROC CALIS supports the following input data sets for various purposes in the analysis:

- INEST= data sets for providing initial parameter estimates. An INEST= data set could be an OUTEST= data set created from a previous analysis.
- INMODEL= data sets for providing model specifications and initial estimates. An INMODEL= data set could be an OUTMODEL= data set created from a previous analysis.
- INWGT= data sets for providing the weight matrices. An INWGT= data set could be an OUTWGT= data set created from a previous analysis.

The CALIS procedure uses ODS Graphics to create high-quality graphs as part of its output. You can produce the following graphical output by specifying the PLOTS= option or the PATHDIAGRAM statement:

- histogram for mean, covariance, or correlation residuals
- histogram for case-level residual M-distances
- case-level residual diagnostic plots such as residual by leverage plot, residual by predicted plot, PP-plot, and QQ-plot
- path diagram for initial model specification, unstandardized solution, or standardized solution

See Chapter 24, "Statistical Graphics Using ODS," for general information about ODS Graphics. See the section "ODS Graphics" on page 1884 and the PLOTS= option on page 1571 for specific information about the statistical graphics available with the CALIS procedure. For more information about producing customized path diagrams, see the options of the PATHDIAGRAM statement.

A Guide to the PROC CALIS Documentation

The CALIS procedure uses a variety of modeling languages to fit structural equation models. This chapter provides documentation for all of them. Additionally, some sections provide introductions to the model specification, the theory behind the software, and other technical details. While some introductory material and examples are provided, this chapter is not a textbook for structural equation modeling and related topics. For didactic treatment of structural equation models with latent variables, see Bollen (1989b) and Loehlin (2004).

Reading this chapter sequentially is not a good strategy for learning about PROC CALIS. This section provides a guide or "road map" to the rest of the PROC CALIS chapter, starting with the basics and continuing through more advanced topics. Many sections assume that you already have a basic understanding of structural equation modeling.

The following table shows three different skill levels of using the CALIS procedure (basic, intermediate, and advanced) and their milestones.

Level	Milestone	Starting Section
Basic	You are able to specify simple models, but might make mistakes.	"Guide to the Basic Skill Level" on page 1511
Intermediate	You are able to specify more sophisticated models with few syntactic and semantic mistakes.	"Guide to the Intermediate Skill Level" on page 1516
Advanced	You are able to use the advanced options provided by PROC CALIS.	"Guide to the Advanced Skill Level" on page 1518

In the next three sections, each skill level is discussed, followed by an introductory section of the reference topics that are not covered in any of the skill levels.

Guide to the Basic Skill Level

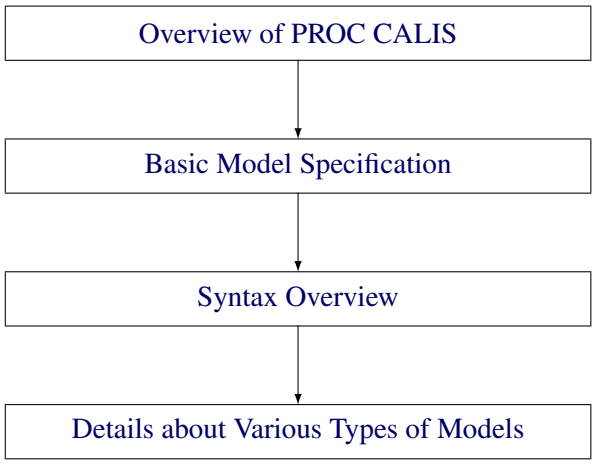

Overview of PROC CALIS

The section "Overview: CALIS Procedure" on page 1506 gives you an overall picture of the CALIS procedure but without the details.

Basic Model Specification

The structural equation example in the section "Getting Started: CALIS Procedure" on page 1521 provides the starting point to learn the basic model specification. You learn how to represent your theory by using a path diagram and then translate the diagram into the PATH model for PROC CALIS to analyze. Because the PATH modeling language is new, this example is useful whether or not you have previous experience with PROC CALIS. The PATH model is specified in the section "PATH Model" on page 1522. The corresponding results are shown and discussed in Example 33.17.

After you learn about the PATH modeling language and an example of its application, you can do either of the following:

- You can continue to learn more modeling languages in the section "Getting Started: CALIS Procedure" on page 1521.

- You can skip to the section "Syntax Overview" on page 1515 for an overview of the PROC CALIS syntax and learn other modeling languages at a later time.

You do not need to learn all of the modeling languages in PROC CALIS. Any one of the modeling languages (LINEQS, LISMOD, PATH, or RAM) is sufficient for specifying a very wide class of structural equation models. PROC CALIS provides different kinds of modeling languages because different researchers might have previously learned different modeling languages or approaches. To get a general idea about different kinds of modeling languages, the following subsections in the "Getting Started: CALIS Procedure" section are useful:

- LINEQS: Section "LINEQS Model" on page 1525
- RAM: Section "RAM Model" on page 1524
- LISMOD: Section "LISMOD Model" on page 1526
- FACTOR: Section "A Factor Model Example" on page 1527
- MSTRUCT: Section "Direct Covariance Structures Analysis" on page 1529

After studying the examples in the "Getting Started: CALIS Procedure" section, you can strengthen your understanding of the various modeling languages by studying more examples such as those in section "Examples: CALIS Procedure" on page 1886. Unlike the examples in the "Getting Started: CALIS Procedure" section, the examples in the "Examples: CALIS Procedure" section include the analysis results in addition to the explanations of the model specifications.

You can start with the following two sets of basic examples:

- MSTRUCT model examples
  The basic MSTRUCT model examples demonstrate the testing of covariance structures directly on the covariance matrices. Although the MSTRUCT model is not the most common structural equation models in applications, these MSTRUCT examples can help you understand the basic form of covariance structures and the corresponding specifications in PROC CALIS.
- PATH model examples
  The basic PATH model examples demonstrate how you can represent your model by path diagrams and by the PATH modeling language. These examples show the most common applications of structural equation modeling.

The following is a summary of the basic MSTRUCT model examples:

- "Example 33.1: Estimating Covariances and Correlations" on page 1886 shows how you can estimate the covariances and correlations with standard error estimates for the variables in your model. The model you fit is a saturated covariance structure model.
- "Example 33.2: Estimating Covariances and Means Simultaneously" on page 1891 extends Example 33.1 to include the mean structures in the model. The model you fit is a saturated mean and covariance structure model.

- "Example 33.3: Testing Uncorrelatedness of Variables" on page 1893 shows a very basic covariance structure model, in which the covariance structures can be specified directly. The variables in this model are uncorrelated. You learn how to specify the covariance pattern directly.
- "Example 33.4: Testing Covariance Patterns" on page 1895 extends Example 33.3 to include other covariance structures that you can specify directly.
- "Example 33.5: Testing Some Standard Covariance Pattern Hypotheses" on page 1897 illustrates the use of built-in covariance patterns supported by PROC CALIS.

The following is a summary of the basic PATH model examples:

- "Example 33.6: Linear Regression Model" on page 1900 shows how you can fit a linear regression model with the PATH modeling language of PROC CALIS. This example also introduces the path diagram representation of "causal" models. You compare results obtained from PROC CALIS and from the REG procedure, which is designed specifically for regression analysis.
- "Example 33.7: Multivariate Regression Models" on page 1904 extends Example 33.6 in several different ways. You fit covariance structure models with more than one predictor, with direct and indirect effects. This example also discusses how you can choose the "best" model for your data.
- "Example 33.8: Measurement Error Models" on page 1921 explores the case where the predictor in simple linear regression is measured with error. The concept of latent true score variable is introduced. You use PROC CALIS to fit a simple measurement error model.
- "Example 33.9: Testing Specific Measurement Error Models" on page 1927 extends Example 33.8 to test special measurement error models with constraints. By using PROC CALIS, you can constrain your measurement error models in many different ways. For example, you can constrain the error variances or the intercepts to test specific hypotheses.
- "Example 33.10: Measurement Error Models with Multiple Predictors" on page 1933 extends Example 33.8 to include more predictors in the measurement error models. The measurement errors in the predictors can be correlated in the model.

More elaborate examples about the MSTRUCT and PATH models are listed as follows:

- "Example 33.17: Path Analysis: Stability of Alienation" on page 1992 shows you how to specify a simple PATH model and interpret the basic estimation results. The results are shown in considerable detail. The output and analyses include: a model summary, an initial model specification, an initial estimation method, an optimization history and results, residual analyses, residual graphics, estimation results, squared multiple correlations, and standardized results.
- "Example 33.19: Fitting Direct Covariance Structures" on page 2013 shows you how to fit your covariance structures directly on the covariance matrix by using the MSTRUCT modeling language. You also learn how to use the FITINDEX statement to create a customized model fit summary and how to save the fit summary statistics into an external file.
- "Example 33.21: Testing Equality of Two Covariance Matrices Using a Multiple-Group Analysis" on page 2027 uses the MSTRUCT modeling language to illustrate a simple multiple-group analysis. You also learn how to use the ODS SELECT statement to customize your printed output.

- "Example 33.22: Testing Equality of Covariance and Mean Matrices between Independent Groups" on page 2032 uses the COVPATTERN= and MEANPATTERN= options to show some tests of equality of covariance and mean matrices between independent groups. It also illustrates how you can improve your model fit by the exploratory use of the Lagrange multiplier statistics for releasing equality constraints.
- "Example 33.24: Testing Competing Path Models for the Career Aspiration Data" on page 2065 illustrates how you can fit competing models by using the OUTMODEL= and INMODEL= data sets for transferring and modifying model information from one analysis to another. This example also demonstrates how you can choose the best model among several competing models for the same data.

After studying the PATH and MSTRUCT modeling languages, you are able to specify most commonly used structural equation models by using PROC CALIS. To broaden your scope of structural equation modeling, you can study some basic examples that use the FACTOR and LINEQS modeling languages. These basic examples are listed as follows:

- "Example 33.11: Measurement Error Models Specified as Linear Equations" on page 1938 explores another way to specify measurement error models in PROC CALIS. The LINEQS modeling language is introduced. You learn how to specify linear equations of the measurement error model by using the LINEQS statement. Unlike the PATH modeling language, in the LINEQS modeling language, you need to specify the error terms explicitly in the model specification.
- "Example 33.12: Confirmatory Factor Models" on page 1944 introduces a basic confirmatory factor model for test items. You use the FACTOR modeling language to specify the factor-variable relationships.
- "Example 33.13: Confirmatory Factor Models: Some Variations" on page 1956 extends Example 33.12 to include some variants of the confirmatory factor model. With the flexibility of the FACTOR modeling language, this example shows how you fit models with parallel items, tau-equivalent items, or partially parallel items.

More advanced examples that use the PATH, LINEQS, and FACTOR modeling languages are listed as follows:

- "Example 33.14: Residual Diagnostics and Robust Estimation" on page 1965 illustrates the use of several graphical residual plots to detect model outliers and leverage observations, to study the departures from the theoretical case-level residual distribution, and to examine the linearity and homoscedasticity of variance. In addition, this example illustrates the use of robust estimation technique to downweight the outliers and to estimate the model parameters.
- "Example 33.15: The Full Information Maximum Likelihood Method" on page 1978 shows how you can use the full information maximum likelihood (FIML) method to estimate your model when your data contain missing values. It illustrates the analysis of the data coverage of the sample variances, covariances, and means and the analysis of missing patterns and the mean profile. It also shows that the full information maximum likelihood method makes the maximum use of the available information from the data, as compared with the default ML (maximum likelihood) methods.

- "Example 33.16: Comparing the ML and FIML Estimation" on page 1987 discusses the similarities and differences between the ML and FIML estimation methods as implemented in PROC CALIS. It uses an empirical example to show how ML and FIML obtain the same estimation results when the data do not contain missing values.

- "Example 33.18: Simultaneous Equations with Mean Structures and Reciprocal Paths" on page 2006 is an econometric example that shows you how to specify models using the LINEQS modeling language. This example also illustrates the specification of reciprocal effects, the simultaneous analysis of the mean and covariance structures, the setting of bounds for parameters, and the definitions of metaparameters by using the PARAMETERS statement and SAS programming statements. You also learn how to shorten your output results by using some global display options such as the PSHORT and NOSTAND options in the PROC CALIS statement.

- "Example 33.20: Confirmatory Factor Analysis: Cognitive Abilities" on page 2016 uses the FACTOR modeling language to illustrate confirmatory factor analysis. In addition, you use the MODIFICATION option in the PROC CALIS statement to compute LM test indices for model modifications.

- "Example 33.25: Fitting a Latent Growth Curve Model" on page 2077 is an advanced example that illustrates the use of structural equation modeling techniques for fitting latent growth curve models. You learn how to specify random intercepts and random slopes by using the LINEQS modeling language. In addition to the modeling of the covariance structures, you also learn how to specify the mean structure parameters.

If you are familiar with the traditional Keesling-Wiley-Jöreskog measurement and structural models (Keesling 1972; Wiley 1973; Jöreskog 1973) or the RAM model (McArdle 1980), you can use the LISMOD or RAM modeling languages to specify structural equation models. The following example shows how to specify these types of models:

- "Example 33.23: Illustrating Various General Modeling Languages" on page 2055 extends Example 33.17, which uses the PATH modeling language, and shows how to use the other general modeling languages: RAM, LINEQS, and LISMOD. These modeling languages enable you to specify the same path model as in Example 33.17 and get equivalent results. This example shows the connections between the general modeling languages supported in PROC CALIS. A good understanding of Example 33.17 is a prerequisite for this example.

Once you are familiar with various modeling languages, you might wonder which modeling language should be used in a given situation. The section "Which Modeling Language?" on page 1530 provides some guidelines and suggestions.

Syntax Overview

The section "Syntax: CALIS Procedure" on page 1532 shows the syntactic structure of PROC CALIS. However, reading the "Syntax: CALIS Procedure" section sequentially might not be a good strategy. The statements used in PROC CALIS are classified in the section "Classes of Statements in PROC CALIS" on page 1533. Understanding this section is a prerequisite for understanding single-group and multiple-group analyses in PROC CALIS. Syntax for single-group analyses is described in the section "Single-Group Analysis Syntax" on page 1536, and syntax for multiple-group analyses is described in the section "Multiple-Group Multiple-Model Analysis Syntax" on page 1536.

You might also want to get an overview of the options in the PROC CALIS statement. However, you can skip the detailed listing of the PROC CALIS statement options. Most of these details serve as references, so you can consult them only when you need to. You can just read the summary tables for the available options in the PROC CALIS statement in the following subsections:

- "Data Set Options" on page 1537
- "Model and Estimation Options" on page 1538
- "Options for Fit Statistics" on page 1539
- "Options for Statistical Analysis" on page 1539
- "Global Display Options" on page 1540
- "Optimization Options" on page 1542

Details about Various Types of Models

Several subsections in the section "Details: CALIS Procedure" on page 1711 can help you gain a deeper understanding of the various types of modeling languages, as shown in the following table:

Language	Section
COSAN	"The COSAN Model" on page 1734
FACTOR	"The FACTOR Model" on page 1738
LINEQS	"The LINEQS Model" on page 1745
LISMOD	"The LISMOD Model and Submodels" on page 1752
MSTRUCT	"The MSTRUCT Model" on page 1761
PATH	"The PATH Model" on page 1763
RAM	"The RAM Model" on page 1769

The specification techniques you learn from the examples cover only parts of the modeling language. A more complete treatment of the modeling languages is covered in these subsections. In addition, you can also learn the mathematical models, model restrictions, and default parameterization of all supported modeling languages in these subsections. To get an overall idea about the default parameterization rules used in PROC CALIS, the section "Default Analysis Type and Default Parameterization" on page 1732 would be very useful. Understanding how PROC CALIS set default parameters would help you specify your models more efficiently and accurately.

Guide to the Intermediate Skill Level

At the intermediate level, you learn to minimize your mistakes in model specification and to establish more sophisticated modeling techniques. The following topics in the "Details: CALIS Procedure" section or elsewhere can help:

- The section "Naming Variables and Parameters" on page 1777 summarizes the naming rules and conventions for variable and parameter names in specifying models.
- The section "Setting Constraints on Parameters" on page 1778 covers various techniques of constraining parameters in model specifications.

- The section "Automatic Variable Selection" on page 1783 discusses how PROC CALIS treats variables in the models and variables in the data sets. It also discusses situations where the VAR statement specification is deemed necessary.
- The section "Computational Problems" on page 1862 discusses computational problems that occur quite commonly in structural equation modeling. It also discusses some possible remedies of the computational problem.
- The section "Missing Values and the Analysis of Missing Patterns" on page 1850 describes the default treatment of missing values.
- The statements REFMODEL on page 1698 and RENAMEPARM on page 1700 are useful when you need to make references to well-defined models when specifying a "new" model. See Example 33.28 for an application.

Revisit topics and examples covered at the basic level, as needed, to help you better understand the topics at the intermediate level.

You can also study the following more advanced examples:

- "Example 33.26: Higher-Order and Hierarchical Factor Models" on page 2082 is an advanced example for confirmatory factor analysis. It involves the specifications of higher-order and hierarchical factor models. Because higher-order factor models cannot be specified by the FACTOR modeling language, you need to use the LINEQS model specification instead. A second-order factor model and a bifactor model are fit. Linear constraints on parameters are illustrated by using the PARAMETERS statement and SAS programming statements. Relationships between the second-order factor model and the bifactor model are numerically illustrated.
- "Example 33.27: Linear Relations among Factor Loadings" on page 2096 is an advanced example of a first-order confirmatory factor analysis that uses the FACTOR modeling language. In this example, you learn how to use the PARAMETERS statement and SAS programming statements to set up dependent parameters in your model. You also learn how to specify the correlation structures for a specific confirmatory factor model.
- "Example 33.28: Multiple-Group Model for Purchasing Behavior" on page 2104 is a sophisticated example of analyzing a path model. The PATH modeling language is used. In this example, a two-group analysis of mean and covariance structures is conducted. You learn how to use the REFMODEL statement to reference properly defined models and the SIMTESTS statement to test a priori simultaneous hypotheses.
- "Example 33.29: Fitting the RAM and EQS Models by the COSAN Modeling Language" on page 2127 introduces the COSAN modeling language by connecting it with general RAM and EQS models. The model matrices of the RAM or EQS model are described. You specify these model matrices and the associated parameters in the COSAN modeling language.
- "Example 33.30: Second-Order Confirmatory Factor Analysis" on page 2141 constructs the covariance structure model of the second-order confirmatory factor model. You define the model matrices by using the COSAN modeling language.
- "Example 33.31: Linear Relations among Factor Loadings: COSAN Model Specification" on page 2146 shows how you can set linear constraints among model parameters under the COSAN model.

- "Example 33.32: Ordinal Relations among Factor Loadings" on page 2151 shows how you can set ordinal constraints among model parameters under the COSAN model.
- "Example 33.33: Longitudinal Factor Analysis" on page 2154 defines the covariance structures of a longitudinal factor model and shows how you can specify the covariance structure model with the COSAN modeling language.

Guide to the Advanced Skill Level

At the advanced level, you learn to use the advanced data analysis and output control tools supported by PROC CALIS.

Advanced Data Analysis Tools

The following advanced data analysis topics are discussed:

- Assessment of fit

  The section "Assessment of Fit" on page 1824 presents the fit indices used in PROC CALIS. However, the more important topics covered in this section are about how model fit indices are organized and used, how residuals can be used to gauge the fitting of individual parts of the model, and how the coefficients of determination are defined for equations.

  To customize your fit summary table, you can use the options on the FITINDEX statement.

- Case-level residual diagnostics

  The section "Case-Level Residuals, Outliers, Leverage Observations, and Residual Diagnostics" on page 1837 describes details about how the residual diagnostics at the individual data level are accomplished in general structural equation modeling, and how they lead to the graphical techniques for detecting outliers and leverage observations, studying residual distributions, and examining linear relationships and heteroscedasticity of error variances.

- Control and customization of path diagrams

  The section "Path Diagrams: Layout Algorithms, Default Settings, and Customization" on page 1784 discusses the path diagram layout algorithms that the CALIS procedure uses. It also illustrates useful options that control and customize path diagrams.

- Effect partitioning

  The section "Total, Direct, and Indirect Effects" on page 1844 discusses the total, direct, and indirect effects and their computations. The stability coefficient of reciprocal causation is also defined.

  To customize the effect analysis, you can use the EFFPART statement.

- Counting and adjusting degrees of freedom

  The section "Counting the Degrees of Freedom" on page 1822 describes how PROC CALIS computes model fit degrees of freedom and how you can use some options on the PROC CALIS statement to make degrees-of-freedom adjustments.

  To adjust the model fit degrees of freedom, you can use the DFREDUCE= and NOADJDF options in the PROC CALIS statement.

- Standardized solutions

  Standardization schemes used in PROC CALIS are described and discussed in the section "Standardized Solutions" on page 1847.

  Standardized solutions are displayed by default. You can turn them off by using the NOSTAND option of the PROC CALIS statement.

- Model modifications

  In the section "Modification Indices" on page 1848, modification indices such as Lagrange multiplier test indices and Wald statistics are defined and discussed. These indices can be used either to enhance your model fit or to make your model more precise.

  To limit the modification process only to those parameters of interest, you can use the LMTESTS statement to customize the sets of LM tests conducted on potential parameters.

- A Priori Parametric Function Testing

  You can use the TESTFUNC statement to test a priori hypotheses individually. You can use the SIMTESTS statement to test a priori hypotheses simultaneously.

Advanced Output Control Tools

To be more effective in presenting your analysis results, you need to be more sophisticated in controlling your output. Some customization tools have been discussed in the previous section "Advanced Data Analysis Tools" on page 1518 and might have been mentioned in the examples included in the basic and the intermediate levels. In the following topics, these output control tools are presented in a more organized way so that you can have a systematic study scheme of these tools.

- Global output control tools in PROC CALIS

  You can control output displays in PROC CALIS either by the global display options or by the individual output printing options. Each global display option typically controls more than one output display, while each individual output display controls only one output display. The global display options can both enable and suppress output displays, and they can also alter the format of the output.

  See the PALL, PRINT, PSHORT, PSUMMARY, and NOPRINT options for ways to control the appearances of the output. See the section "Global Display Options" on page 1540 for details about the global display options and their relationships with the individual output display options. Also see the ORDERALL, ORDERGROUPS, ORDERMODELS, ORDERSPEC, PARMNAME, PRIMAT, NOORDERSPEC, NOPARMNAME, NOSTAND, and NOSTDERR options which control the output formats.

- Customized analysis tools in PROC CALIS

  Many individual output displays in PROC CALIS can be customized via specific options or statements. If you do not use these customization tools, the default output will usually contain a large number of displays or displays with very large dimensions. These customized analysis tools are as follows:

  – The ON=, OFF=, ON(ONLY)= options in the FITINDEX statement enable you to select individual or groups of model fit indices or modeling information to display. You can still save the information of *all* fit indices in an external file by using the OUTFIT= option.

  - The EFFPART statement enables you to customize the effect analysis. You display only those effects of substantive interest.
  - The LMTESTS statement enables you to customize the sets of LM tests of interest. You test only those potential parameters that are theoretically and substantively possible.

- Output selection and destinations by the ODS system

  This kind of output control is used not only for PROC CALIS, but is used for all procedures that support the ODS system. The most common uses include output selection and output destinations assignment. You use the ODS SELECT statement together with the ODS table names or graph names to select particular output displays. See the section "ODS Table Names" on page 1870 for these names in PROC CALIS.

  The default output destination of PROC CALIS is the listing destination. You can add or change the destinations by using statements such as **`ods html`** (for html output), **`ods rtf`** (for rich text output), and so on. For details, see Chapter 23, "Using the Output Delivery System."

Reference Topics

Some topics in the "Details: CALIS Procedure" section are intended primarily for references—you consult them only when you encounter specific problems in the PROC CALIS modeling or when you need to know the very fine technical details in certain special situations. Many of these reference topics in the "Details: CALIS Procedure" section are not required for practical applications of structural equation modeling. The following technical topics are discussed:

- Measures of multivariate kurtosis and skewness

  This is covered in the section "Measures of Multivariate Kurtosis" on page 1851.

- Estimation criteria and the mathematical functions for estimation

  The section "Estimation Criteria" on page 1805 presents formulas for various estimation criteria. The relationships among these criteria are shown in the section "Relationships among Estimation Criteria" on page 1816. To optimize an estimation criterion, you usually need its gradient and Hessian functions. These functions are detailed in the section "Gradient, Hessian, Information Matrix, and Approximate Standard Errors" on page 1818, where you can also find information about the computation of the standard error estimates in PROC CALIS. Unlike other estimation methods, the robust estimation methods do not optimize a discrepancy function themselves. The robust estimation methods that are implemented in PROC CALIS use the iteratively reweighted least squares (IRLS) method to obtain parameter convergence. The robust estimation technique is detailed in the section "Robust Estimation" on page 1812.

- Initial estimation

  Initial estimates are necessary for all kinds of iterative optimization techniques. They are described in section "Initial Estimates" on page 1853.

- Use of optimization techniques

  Optimization techniques are covered in section "Use of Optimization Techniques" on page 1854. See this section if you need to fine-tune the optimization.

- Output displays and control

  The output displays in PROC CALIS are listed in the section "Displayed Output" on page 1865. General requirements for the displays are also shown.

  With the ODS system, each table and graph has a name, which can be used on the ODS OUTPUT or ODS SELECT statement. See the section "ODS Table Names" on page 1870 for the ODS table and graph names.

- Input and output files

  PROC CALIS supports several input and output data files for data, model information, weight matrices, estimates, fit indices, and estimation and descriptive statistics. The uses and the structures of these input and output data files are described in the sections "Input Data Sets" on page 1711 and "Output Data Sets" on page 1715.

Getting Started: CALIS Procedure

A Structural Equation Example

This example from Wheaton et al. (1977) illustrates the basic uses of the CALIS procedure and the relationships among the LINEQS, LISMOD, PATH, and RAM modeling languages. Different structural models for these data are analyzed in Jöreskog and Sörbom (1985) and in (Bentler 1995, p. 28). The data contain the following six (manifest) variables collected from 932 people in rural regions of Illinois:

Anomie67:	Anomie 1967
Powerless67:	Powerlessness 1967
Anomie71:	Anomie 1971
Powerless71:	Powerlessness 1971
Education:	Education level (years of schooling)
SEI:	Duncan's socioeconomic index (SEI)

The covariance matrix of these six variables is stored in the data set named Wheaton.

It is assumed that anomie and powerlessness are indicators of an alienation factor and that education and SEI are indicators for a socioeconomic status (SES) factor. Hence, the analysis contains three latent variables (factors):

Alien67:	Alienation 1967
Alien71:	Alienation 1971
SES:	Socioeconomic status (SES)

The path diagram in Figure 33.1 shows the structural model used in Bentler (1985, p. 29) and slightly modified in Jöreskog and Sörbom (1985, p. 56).

Figure 33.1 Path Diagram of Stability and Alienation Example

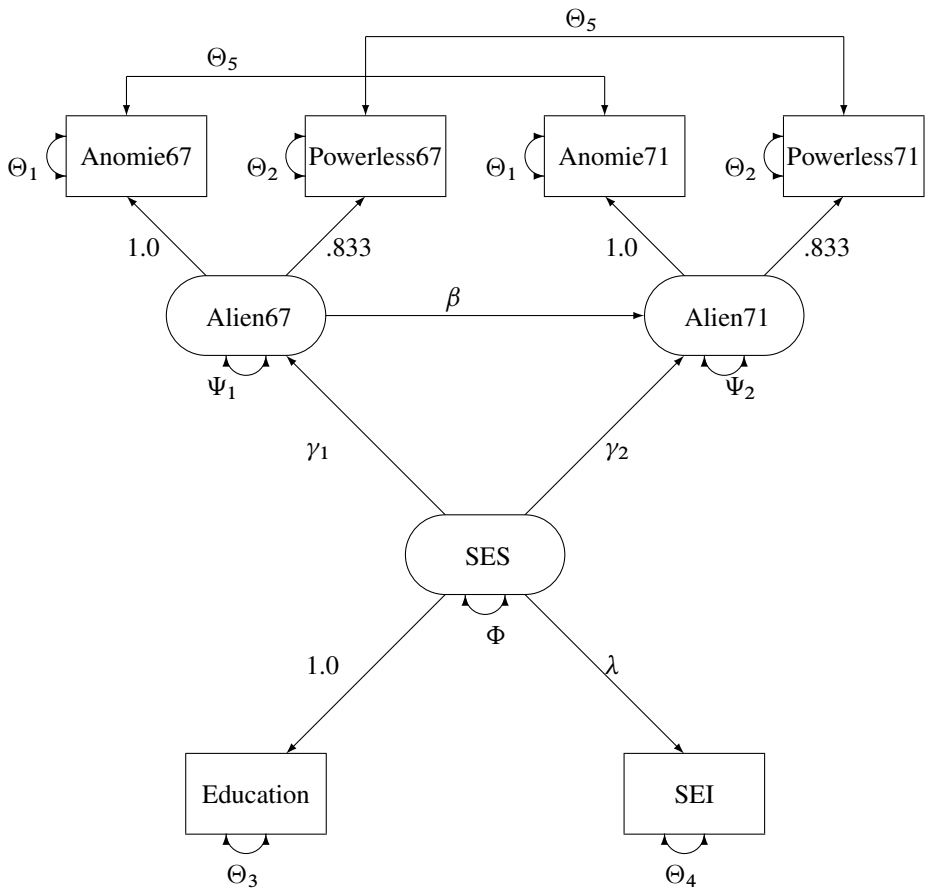

In the path diagram shown in Figure 33.1, regressions of variables are represented by one-headed arrows. Regression coefficients are indicated along these one-headed arrows. Variances and covariances among the variables are represented by two-headed arrows. Error variances and covariances are also represented by two-headed arrows. This scheme of representing paths, variances and covariances, and error variances and covariances (McArdle 1988; McDonald 1985) is helpful in translating the path diagram to the PATH or RAM model input in the CALIS procedure.

PATH Model

Specification by using the PATH modeling language is direct and intuitive in PROC CALIS once a path diagram is drawn. The following statements specify the path diagram almost intuitively:

```
proc calis nobs=932 data=Wheaton;
   path
      Anomie67     <=== Alien67   = 1.0,
      Powerless67  <=== Alien67   = 0.833,
      Anomie71     <=== Alien71   = 1.0,
      Powerless71  <=== Alien71   = 0.833,
```

```
      Education    <=== SES       = 1.0,
      SEI          <=== SES       = lambda,
      Alien67      <=== SES       = gamma1,
      Alien71      <=== SES       = gamma2,
      Alien71      <=== Alien67   = beta;
   pvar
      Anomie67     = theta1,
      Powerless67  = theta2,
      Anomie71     = theta1,
      Powerless71  = theta2,
      Education    = theta3,
      SEI          = theta4,
      Alien67      = psi1,
      Alien71      = psi2,
      SES          = phi;
   pcov
      Anomie67    Anomie71    = theta5,
      Powerless67 Powerless71 = theta5;
run;
```

In the PROC CALIS statement, you specify Wheaton as the input data set, which contains the covariance matrix of the variables.

In the PATH model specification, all the one-headed arrows in the path diagram are represented as path entries in the PATH statement, with entries separated by commas. In each path entry, you specify a pair of variables and the direction of the path (either `<===` or `===>`), followed by a path coefficient, which is either a fixed constant or a parameter with a name in the specification.

All the two-headed arrows each with the same source and destination are represented as entries in the PVAR statement, with entries separated by commas. In the PVAR statement, you specify the variance or error (or partial) variance parameters. In each entry, you specify a variable and then a parameter name or a fixed parameter value. If the variable involved is exogenous in the model (serves only as a predictor; never being pointed at by one-headed arrows), you are specifying a variance parameter for an exogenous variable in the PVAR statement. Otherwise, you are specifying an error variance (or a partial variance) parameter for an endogenous variable.

All other two-headed arrows are represented as entries in the PCOV statement, with entries separated by commas. In the PCOV statement, you specify the covariance or error (or partial) covariance parameters. In each entry, you specify a pair of variables and then a parameter name or a fixed parameter value. If both variables involved in an entry are exogenous, you are specifying a covariance parameter. If both variables involved in an entry are endogenous, you are specifying an error (or partial) covariance parameter. When one variable is exogenous and the other is endogenous in an entry, you are specifying a partial covariance parameter that can be interpreted as the covariance between the exogenous variable and the error of the endogenous variable.

See Example 33.17 for the results of the current PATH model analysis. For more information about the PATH modeling language, see the section "The PATH Model" on page 1763 and the PATH statement on page 1665.

RAM Model

The PATH modeling language is not the only specification method that you can use to represent the path diagram. You can also use the RAM, LINEQS or LISMOD modeling language to represent the diagram equivalently.

The RAM model specification in PROC CALIS resembles that of the PATH model, as shown in the following statements:

```
proc calis nobs=932 data=Wheaton;
   ram
      var =  Anomie67     /* 1 */
             Powerless67  /* 2 */
             Anomie71     /* 3 */
             Powerless71  /* 4 */
             Education    /* 5 */
             SEI          /* 6 */
             Alien67      /* 7 */
             Alien71      /* 8 */
             SES,         /* 9 */
      _A_    1   7   1.0,
      _A_    2   7   0.833,
      _A_    3   8   1.0,
      _A_    4   8   0.833,
      _A_    5   9   1.0,
      _A_    6   9   lambda,
      _A_    7   9   gamma1,
      _A_    8   9   gamma2,
      _A_    8   7   beta,
      _P_    1   1   theta1,
      _P_    2   2   theta2,
      _P_    3   3   theta1,
      _P_    4   4   theta2,
      _P_    5   5   theta3,
      _P_    6   6   theta4,
      _P_    7   7   psi1,
      _P_    8   8   psi2,
      _P_    9   9   phi,
      _P_    1   3   theta5,
      _P_    2   4   theta5;
run;
```

In the RAM statement, you specify a list of entries for parameters, with entries separated by commas. In each entry, you specify the type of parameter (PATH, PVAR, or PCOV in the code), the associated variable or pair of variables and the path direction if applicable, and then a parameter name or a fixed parameter value. The types of parameters you specify in this RAM model are for path coefficients, variances or partial variances, and covariances or partial covariances. They bear the same meanings as those in the PATH model specified previously. The RAM model specification is therefore quite similar to the PATH model specification—except that in the RAM model you put all parameter specification in the same list under the RAM statement, whereas you specify different types of parameters separately under different statements in the PATH model.

See Example 33.23 for partial results of the current RAM model analysis. For more information about the RAM modeling language, see the section "The RAM Model" on page 1769 and the RAM statement on page 1692.

LINEQS Model

The LINEQS modeling language uses equations to specify functional relationships among variables, as shown in the following statements:

```
proc calis nobs=932 data=Wheaton;
   lineqs
      Anomie67     = 1.0    * f_Alien67 + E1,
      Powerless67  = 0.833  * f_Alien67 + E2,
      Anomie71     = 1.0    * f_Alien71 + E3,
      Powerless71  = 0.833  * f_Alien71 + E4,
      Education    = 1.0    * f_SES     + E5,
      SEI          = lambda * f_SES     + E6,
      f_Alien67    = gamma1 * f_SES     + D1,
      f_Alien71    = gamma2 * f_SES     + beta * f_Alien67 + D2;
   std
      E1           = theta1,
      E2           = theta2,
      E3           = theta1,
      E4           = theta2,
      E5           = theta3,
      E6           = theta4,
      D1           = psi1,
      D2           = psi2,
      f_SES        = phi;
   cov
      E1  E3       = theta5,
      E2  E4       = theta5;
run;
```

In the LINEQS statement, equations are separated by commas. In each equation, you specify an endogenous variable on the left-hand side, and then predictors and path coefficients on the right-hand side of the equal side. The set of equations specified in this LINEQS model is equivalent to the system of paths specified in the preceding PATH (or RAM) model. However, there are some notable differences between the LINEQS and the PATH specifications.

First, in the LINEQS modeling language you must specify the error terms explicitly as exogenous variables. For example, E1, E2, and D1 are error terms in the specification. In the PATH (or RAM) modeling language, you do not need to specify error terms explicitly.

Second, equations specified in the LINEQS modeling language are oriented by the endogenous variables. Each endogenous variable can appear on the left-hand side of an equation only *once* in the LINEQS statement. All the corresponding predictor variables must then be specified on the right-hand side of the equation. For example, f_Alien71 is predicted from f_Alien67 and f_SES in the last equation of the LINEQS statement. In the PATH or RAM modeling language, however, you would specify the same functional relationships in two separate paths.

Third, you must follow some naming conventions for latent variables when using the LINEQS modeling language. The names of latent variables that are not errors or disturbances must start with an ‘f’ or ‘F’. Also, the names of the error variables must start with ‘e’ or ‘E’ and the names of the disturbance variables must start with ‘d’ or ‘D’. For example, variables Alien67, Alien71, and SES serve as latent factors in the previous PATH or RAM model specification. To comply with the naming conventions, these variables are named with an extra prefix ‘f_’ in the LINEQS model specification—that is, f_Alien67, f_Alien71, and f_SES,

respectively. In addition, because of the naming conventions of the LINEQS modeling language, E1–E6 serve as error terms and D1–D1 serve as disturbances in the specification.

A consequence of explicit specification of error terms in the LINEQS statement is that the partial variance and partial covariance concepts used in the PATH and RAM modeling languages are no longer needed. They are replaced by the variances or covariances of the error terms or disturbances. Errors and disturbances are exogenous variables by nature. Hence, in terms of variance and covariance specification, they are treated exactly the same way as other non-error exogenous variables in the LINEQS modeling language. That is, variance parameters for all exogenous variables, including errors and disturbances, are specified in the VARIANCE statement, and covariance parameters among exogenous variables, including errors and disturbances, are specified in COV statement.

See Example 33.23 for partial results of the current LINEQS model analysis. For more information about the LINEQS modeling language, see the section "The LINEQS Model" on page 1745 and the LINEQS statement on page 1618.

LISMOD Model

The LISMOD language is quite different from the LINEQS, PATH, and RAM modeling languages. In the LISMOD specification, you define parameters as entries in model matrices, as shown in the following statements:

```
proc calis nobs=932 data=Wheaton;
   lismod
      yvar = Anomie67 Powerless67 Anomie71 Powerless71,
      xvar = Education SEI,
      etav = Alien67 Alien71,
      xiv  = SES;
   matrix _LAMBDAY_ [1,1] = 1.0,
                    [2,1] = 0.833,
                    [3,2] = 1.0,
                    [4,2] = 0.833;
   matrix _LAMBDAX_ [1,1] = 1.0,
                    [2,1] = lambda;
   matrix _GAMMA_   [1,1] = gamma1,
                    [2,1] = gamma2;
   matrix _BETA_    [2,1] = beta;
   matrix _THETAY_  [1,1] = theta1,
                    [2,2] = theta2,
                    [3,3] = theta1,
                    [4,4] = theta2,
                    [3,1] = theta5,
                    [4,2] = theta5;
   matrix _THETAX_  [1,1] = theta3,
                    [2,2] = theta4;
   matrix _PSI_     [1,1] = psi1,
                    [2,2] = psi2;
   matrix _PHI_     [1,1] = phi;
run;
```

In the LISMOD statement, you specify the lists of variables in the model. In the MATRIX statements, you specify the parameters in the LISMOD model matrices. Each MATRIX statement contains the matrix name of interest and then locations of the parameters, followed by the parameter names or fixed parameter values. It

would be difficult to explain the LISMOD specification here without better knowledge about the formulation of the mathematical model. For this purpose, see the section "The LISMOD Model and Submodels" on page 1752 and the LISMOD statement on page 1624. See also Example 33.23 for partial results of the current LISMOD model analysis.

COSAN Model

The COSAN model specification is even more abstract than all of the modeling languages considered. Like the LISMOD model specification, to specify a COSAN model you need to define parameters as entries in model matrices. In addition, you must also provide the definitions of the model matrices and the matrix formula for the covariance structures in the COSAN model specification. Therefore, the COSAN model specification requires sophisticated knowledge about the formulation of the mathematical model. For this reason, the COSAN model specification of the preceding path model is not discussed here (but see Example 33.29). For more details about the COSAN model specification, see the section "The COSAN Model" on page 1734 and the COSAN statement on page 1583.

A Factor Model Example

In addition to the general modeling languages such as PATH, RAM, LINEQS, and LISMOD, the CALIS procedure provides a specialized language for factor analysis. In the FACTOR modeling language, you can specify either exploratory or confirmatory factor models. For exploratory factor models, you can specify the number of factors, factor extraction method, and rotation algorithm, among many other options. For confirmatory factor models, you can specify the variable-factor relationships, factor variances and covariances, and the error variances.

For example, the following is an exploratory factor model fitted to the Wheaton et al. (1977) data by using PROC CALIS:

```
proc calis nobs=932 data=Wheaton corr;
   factor n=2 rotate=varimax;
run;
```

In this model, you want to get the varimax-rotated solution with two factors by analyzing the correlation matrix (with the CORR option). By default, the factor extraction method is maximum likelihood (METHOD=ML). Maximum likelihood exploratory factor analysis by PROC CALIS can also be done equivalently by the FACTOR procedure, as shown in the following statements for the Wheaton et al. (1977) data:

```
proc factor nobs=932 data=Wheaton n=2 rotate=varimax method=ml;
run;
```

Note that METHOD=ML is necessary because maximum likelihood is not the default method in PROC FACTOR.

Whereas you can use either the CALIS or FACTOR procedure to fit certain exploratory factor models for correlations, you can only use the CALIS procedure to fit confirmatory factor models. In a confirmatory factor model, you are assumed to have some prior knowledge about the variable-factor relations. For example, in your substantive theory, some observed variables are not related to certain factors in the model. The following statements illustrate the specification of a confirmatory factor model for Wheaton et al. (1977) data:

```
proc calis nobs=932 data=Wheaton;
   factor
      Alien67 ===> Anomie67 Powerless67    = 1.0 load1,
      Alien71 ===> Anomie71 Powerless71    = 1.0 load2,
      SES     ===> Education SEI           = 1.0 load3;
   pvar
      Alien67      = phi11,
      Alien71      = phi22,
      SES          = phi33,
      Anomie67     = theta1,
      Powerless67  = theta2,
      Anomie71     = theta3,
      Powerless71  = theta4,
      Education    = theta5,
      SEI          = theta6;
   cov
      Alien71 Alien67 = phi21,
      SES     Alien67 = phi31,
      SES     Alien71 = phi32;
run;
```

Unlike the model fitted by the PATH, RAM, LINEQS, or LISMOD modeling language in previous sections, the confirmatory factor model considered here is purely a measurement model—that is, there are no functional relationships among factors in the model (beyond the covariances among factors) and hence it is a different model. In the FACTOR statement, you specify factors on the left-hand side of the entries, followed by arrows and the manifest variables that are related to the factors. On the right-hand side of the entries, you specify either parameter names or fixed parameter values for the corresponding factor loadings. In this example, there are three factors with three loadings to estimate. In the PVAR statement, you specify the parameters for factor variances and error variances of manifest variables. In the COV statement, you specify the factor covariances. As compared with the PATH, RAM, LINEQS, or LISMOD, the factor modeling language has more restrictions on parameters. These restrictions are listed as follows:

- factor-factor paths and variable-to-factor paths are not allowed
- error covariances and factor-error covariances are not allowed

For more information about exploratory and confirmatory factor models and the FACTOR modeling language, see the section "The FACTOR Model" on page 1738 or the FACTOR statement on page 1600.

Direct Covariance Structures Analysis

Previous examples are concerned with the implied covariance structures from the functional relationships among manifest and latent variables. In some cases, direct modeling of the covariance structures is not only possible, but indeed more convenient. The MSTRUCT modeling language in PROC CALIS is designed for this purpose. Consider the following four variables from the Wheaton et al. (1977) data:

Anomie67:	Anomie 1967
Powerless67:	Powerlessness 1967
Anomie71:	Anomie 1971
Powerless71:	Powerlessness 1971

The covariance structures are hypothesized as follows:

$$\mathbf{\Sigma} = \begin{pmatrix} \phi_1 & \theta_1 & \theta_2 & \theta_1 \\ \theta_1 & \phi_2 & \theta_1 & \theta_3 \\ \theta_2 & \theta_1 & \phi_1 & \theta_1 \\ \theta_1 & \theta_3 & \theta_1 & \phi_2 \end{pmatrix}$$

where:

ϕ_1:	Variance of Anomie
ϕ_2:	Variance of Powerlessness
θ_1:	Covariance between Anomie and Powerlessness
θ_2:	Covariance between Anomie measures
θ_3:	Covariance between Powerlessness measures

In the hypothesized covariance structures, the variances of Anomie and Powerlessness measures are assumed to stay constant over the two time points. Their covariances are also independent of the time of measurements. To test the tenability of this covariance structure model, you can use the following statements of the MSTRUCT modeling language:

```
proc calis nobs=932 data=Wheaton;
   mstruct
      var = Anomie67 Powerless67 Anomie71 Powerless71;
   matrix _COV_ [1,1] = phi1,
                [2,2] = phi2,
                [3,3] = phi1,
                [4,4] = phi2,
                [2,1] = theta1,
                [3,1] = theta2,
                [3,2] = theta1,
                [4,1] = theta1,
                [4,2] = theta3,
                [4,3] = theta1;
run;
```

In the MSTRUCT statement, you specify the list of variables of interest with the VAR= option. The order of the variables in the list will be the order in the hypothesized covariance matrix. Next, you use the MATRIX _COV_ statement to specify the parameters in the covariance matrix. The specification is a direct translation from the hypothesized covariance matrix. For example, the `[1,1]` element of the covariance matrix is fitted by the free parameter phi1. Depending on the hypothesized model, you can also specify fixed constants for the elements in the covariance matrix. If an element in the covariance matrix is not specified by either a parameter name or a constant, it is assumed to be a fixed zero.

The analysis of this model is carried out in Example 33.19.

The MSTRUCT modeling language appears to be more restrictive than any of the other modeling languages discussed, in regard to the following limitations:

- It does not explicitly support latent variables in modeling.
- It does not explicitly support modeling of linear functional relations among variables (for example, paths).

However, these limitations are more apparent than real. In PROC CALIS, the parameters defined in models can be dependent. These dependent parameters can be defined further as functions of other parameters in the PARAMETERS and the SAS programming statements. With these capabilities, it is possible to fit structural models with latent variables and with linear functional relations by using the MSTRUCT modeling language. However, this requires a certain level of sophistication in statistical knowledge and in programming. Therefore, it is recommended that the MSTRUCT modeling language be used only when the covariance and mean structures are modeled directly.

For more information about the MSTRUCT modeling language, see the section "The MSTRUCT Model" on page 1761 and the MSTRUCT statement on page 1656.

Which Modeling Language?

Various modeling languages are supported in PROC CALIS because researchers are trained in or adhere to different schools of modeling. Different modeling languages reflect different modeling terminology and philosophies. The statistical and mathematical consequences by using these various modeling languages, however, might indeed be the same. In other words, you can use more than one modeling language for certain types of models without affecting the statistical analysis. Given the choices, which modeling language is preferred? There are two guidelines for this:

- Use the modeling language that you are most familiar with.
- Use the most specialized modeling language whenever it is possible.

The first guideline calls for researchers' knowledge about a particular modeling language. Use the language you know the best. For example, some researchers might find equation input language like LINEQS the most suitable, while others might feel more comfortable using matrix input language like LISMOD.

The second guideline depends on the nature of the model at hand. For example, to specify a factor analysis model in the CALIS procedure, the specialized FACTOR language, instead of the LISMOD language,

is recommended. It is less error-prone to use a more specialized modeling language. In addition, using a specialized language like FACTOR in this case amounts to giving the CALIS procedure additional information about the specific mathematical properties of the model. This additional information is used to enhance computational efficiency and to provide more specialized results. Another example is fitting an equi-covariance model. You can simply use the MSTRUCT model specification, in which you specify the same parameter for all off-diagonal elements of the covariance elements. This is direct and intuitive. Alternatively, you could tweak a LINEQS model that would predict the same covariance for all variables. However, this is indirect and error-prone, especially for novice modelers.

In PROC CALIS, the FACTOR and MSTRUCT modeling languages are considered more specialized, while other languages are more general in applications. Whenever possible, you should use the more specialized languages. However, if your model involves some novel covariance or mean structures that are not covered by the more specialized modeling languages, you can consider the more generalized modeling languages. See Example 33.33 for an application of the generalized COSAN model.

Syntax: CALIS Procedure

The following statements are available in the CALIS procedure:

PROC CALIS < *options* > ;
- **AUXILIARY** *variables* ;
- **BOUNDS** *boundary-constraints* ;
- **BY** *variables* ;
- **COSAN** *model-specifications* ;
- **COV** *covariance-parameters* ;
- **DETERM** *variables* ;
- **EFFPART** *effects* ;
- **FACTOR** *model-specifications* ;
- **FITINDEX** *fit-options* ;
- **FREQ** *variable* ;
- **GROUP** *group-number* < / *options* > ;
- **LINCON** *linear-constraints* ;
- **LINEQS** *equations* ;
- **LISMOD** *variable-lists* ;
- **LMTESTS** *test-options* ;
- **MATRIX** *parameter-specifications* ;
- **MEAN** *mean-parameters* ;
- **MODEL** *model-number* < / *options* > ;
- **MSTRUCT** *variable-list* ;
- **NLINCON** *nonlinear-constraints* ;
- **NLOPTIONS** *optimization-options* ;
- **OUTFILES** *file-options* ;
- **PARAMETERS** *parameter-specifications* ;
- **PARTIAL** *variables* ;
- **PATH** *paths* ;
- **PATHDIAGRAM** < *options* > ;
- **PCOV** *covariance-parameters* ;
- **PVAR** *variance-parameters* ;
- **RAM** *model-specifications* ;
- **REFMODEL** *model-number* < / *options* > ;
- **RENAMEPARM** *parameter-assignments* ;
- **SIMTESTS** *simultaneous-tests* ;
- **STD** *variance-parameters* ;
- **STRUCTEQ** *variables* < *label* > ;
- **TESTFUNC** *parametric-functions* ;
- **VAR** *variables* ;
- **VARIANCE** *variance-parameters* ;
- **VARNAMES** *name-assignments* ;
- **WEIGHT** *variable* ;
- **Programming statements** ;

Classes of Statements in PROC CALIS

To better understand the syntax of PROC CALIS, it is useful to classify the statements into classes. These classes of statements are described in the following sections.

PROC CALIS Statement

The PROC CALIS statement is the main statement that invokes the CALIS procedure. You can specify options for input and output data sets, printing, statistical analysis, and computations in this statement. The options specified in the PROC CALIS statement will propagate to all groups and models, but are superseded by the options specified in the individual GROUP or MODEL statements.

GROUP Statement

The GROUP statement signifies the beginning of a group specification. A group in the CALIS procedure is an independent sample of observations. You can specify options for input and output data sets, printing, and statistical computations in this statement. Some of these group options in the GROUP statement can also be specified in the MODEL or PROC CALIS statement, but the options specified in the GROUP statement supersede those specified in the MODEL or PROC CALIS statement for the group designated in the GROUP statement. For group options that are available in both of the GROUP and PROC CALIS statements, see the section "Options Available in the GROUP and PROC CALIS Statements" on page 1615. For group options that are available in the GROUP, MODEL, and PROC CALIS statements, see the section "Options Available in GROUP, MODEL, and PROC CALIS Statements" on page 1616. If no GROUP statement is used, a single-group analysis is assumed. The group options for a single-group analysis are specified in the PROC CALIS statement.

The GROUP statement can be followed by subsidiary group specification statements, which specify further data processing procedures for the group designated in the GROUP statement.

Subsidiary Group Specification Statements

Subsidiary group specification statements are for specifying additional data processing attributes for the input data. These statements are summarized in the following table:

Statement	Description
AUXILIARY on page 1581	Specifies the auxiliary variables
FREQ on page 1614	Specifies the frequency variable for the input observations
PARTIAL on page 1665	Specifies the partial variables
VAR on page 1703	Specifies the set of variables in analysis
WEIGHT on page 1711	Specifies the weight variable for the input observations

These statements can be used after the PROC CALIS statement or each GROUP statement. Again, the specifications within the scope of the GROUP statement supersede those specified after the PROC CALIS statement for the group designated in the GROUP statement.

MODEL Statement

The MODEL statement signifies the beginning of a model specification. In the MODEL statement, you can specify the fitted groups, input and output data sets for model specification or estimates, printing options, statistical analysis, and computational options. Some of the options in the MODEL statement can also be specified in the PROC CALIS statement. These options are called model options. Model options specified in the MODEL statement supersede those specified in the PROC CALIS statement. For model options that are available in both of the MODEL and PROC CALIS statements, see the section "Options Available in the MODEL and PROC CALIS Statements" on page 1655. If no MODEL statement is used, a single model is assumed and the model options are specified in the PROC CALIS statement.

Some of the options in the MODEL statement can also be specified in the GROUP statement. These options are called group options. The group options in the MODEL statement are transferred to the groups being fitted, but they are superseded by the group options specified in the associated GROUP statement. For group options that are available in the GROUP and the MODEL statements, see the section "Options Available in GROUP, MODEL, and PROC CALIS Statements" on page 1616.

The MODEL statement itself does not define the model being fitted to the data; the main and subsidiary model specification statements that follow immediately after the MODEL statement do. These statements are described in the next two sections.

Main Model Specification Statements

Main model specification statements are for specifying the type of the modeling language and the main features of the model. These statements are summarized in the following table:

Statement	Description
COSAN on page 1583	Specifies general mean and covariance structures in matrix terms
FACTOR on page 1600	Specifies confirmatory or exploratory factor models
LINEQS on page 1618	Specifies models by using linear equations
LISMOD on page 1624	Specifies models in terms of LISREL-like model matrices
MSTRUCT on page 1656	Specifies parameters directly in the mean and covariance matrices
PATH on page 1665	Specifies models by using the causal paths of variables
RAM on page 1692	Specifies models by using RAM-like lists of parameters
REFMODEL on page 1698	Specifies a base model from which the target model is modified

You can use one of these statements for specifying one model. Each statement in the list represents a particular type of modeling language. After the main model specification statement, you might need to add subsidiary model specification statements, as described in the following section, to complete the model specification.

Subsidiary Model Specification Statements

Subsidiary model specification statements are used to supplement the model specification. They are specific to the types of the modeling languages invoked by the main model specification statements, as shown in the following table:

Statement	Specification	Modeling Languages
COV on page 1593	Covariance parameters	FACTOR, LINEQS
MATRIX on page 1638	Parameters in matrices	COSAN, LISMOD, MSTRUCT
MEAN on page 1651	Mean or intercept parameters	FACTOR, LINEQS, PATH
PCOV on page 1687	(Partial) covariance parameters	PATH
PVAR on page 1690	(Partial) variance parameters	FACTOR, PATH
RENAMEPARM on page 1700	New parameters by renaming	REFMODEL
VARIANCE on page 1706	Variance parameters	LINEQS

Notice that the RAM modeling language does not have any subsidiary model specification statements, because all model specification can be done in the RAM statement.

Model Analysis Statements

Model analysis statements are used to request specific statistical analysis, as shown in the following table:

Statement	Analysis
DETERM on page 1598	Sets variable groups for computing the determination coefficients; same as the STRUCTEQ statement
EFFPART on page 1554	Displays and partitions the effects in the model
FITINDEX on page 1610	Controls the fit summary output
LMTESTS on page 1628	Defines the Lagrange multiplier test regions
SIMTESTS on page 1701	Defines simultaneous parametric function tests
STRUCTEQ on page 1598	Sets variable groups for computing the determination coefficients; same as the DETERM statement
TESTFUNC on page 1702	Tests individual parametric functions

Notice that the DETERM and the STRUCTEQ statements function exactly the same way.

Optimization Statements

Optimization statements are used to define additional parameters and parameter constraints, to fine-tune the optimization techniques, or to set the printing options in optimization, as shown in the following table:

Statement	Description
BOUNDS on page 1582	Defines the bounds of parameters
LINCON on page 1617	Defines the linear constraints of parameters
NLINCON on page 1659	Defines the nonlinear constraints of parameters
NLOPTIONS on page 1659	Sets the optimization techniques and printing options

Other Statements

Other statements that are not listed in preceding sections are summarized in the following table:

Statement	Description
BY on page 1583	Fits a model to different groups separately
OUTFILES on page 1660	Controls multiple output data sets
PARAMETERS on page 1662	Defines additional parameters or superparameters
SAS programming statements on page 1701	Define parameters or functions

Note that SAS programming statements include the ARRAY statement and the mathematical statements for defining parameter interdependence.

Single-Group Analysis Syntax

```
PROC CALIS <options> ;
    subsidiary group specification statements ;
    main model specification statement ;
        subsidiary model specification statements ;
    model analysis statements ;
    optimization statements ;
    other statements ;
```

In a single-group analysis, there is only one group and one model. Because all model or group specifications are unambiguous, the MODEL and GROUP statements are not necessary. The order of the statements is not important for parsing purposes, although you might still like to order them in a particular way to aid understanding. Notice that the OUTFILES statement is not necessary in single-group analyses, as it is designed for multiple-group situations. Output file options in a single-group analysis can be specified in the PROC CALIS statement.

Multiple-Group Multiple-Model Analysis Syntax

```
PROC CALIS <options> ;
    subsidiary group specification statements ;
    model analysis statements ;
    GROUP 1 </ group options> ;
        subsidiary group specification statements ;
    GROUP 2 </ group options> ;
        subsidiary group specification statements ;
    MODEL 1 </ model options> ;
        main model specification statement ;
        subsidiary model specification statements ;
        model analysis statements ;
    MODEL 2 </ model options> ;
        main model specification statement ;
        subsidiary model specification statements ;
        model analysis statements ;
    optimization statements ;
    other statements ;
```

The multiple uses of the GROUP and the MODEL statements characterize the multiple-group multiple-model analysis. Unlike the single-group analysis, the order of some statements in a multiple-group multiple-model syntax is important for parsing purposes.

A GROUP statement signifies the beginning of a group specification block and designates a group number for the group. The scope of a GROUP statement extends to the subsequent subsidiary group specification statements until another MODEL or GROUP statement is encountered. In the preceding syntax, GROUP

1 and GROUP 2 have separate blocks of subsidiary group specification statements. By using additional GROUP statements, you can add as many groups as your situation calls for. Subsidiary group specification statements declared before the first GROUP statement are in the scope of the PROC CALIS statement. This means that these subsidiary group specification statements are applied globally to all groups unless they are respecified locally within the scopes of individual GROUP statements.

A MODEL statement signifies the beginning of a model specification block and designates a model number for the model. The scope of a MODEL statement extends to the subsequent main and subsidiary model specification statements until another MODEL or GROUP statement is encountered. In the preceding syntax, MODEL 1 and MODEL 2 have separate blocks of main and subsidiary model specification statements. By using additional MODEL statements, you can add as many models as your situation calls for. If you use at least one MODEL statement, any main and subsidiary model specification statements declared before the first MODEL statement are ignored.

Some model analysis statements are also bounded by the scope of the MODEL statements. These statements are: DETERM, EFFPART, LMTESTS, and STRUCTEQ. These statements are applied only locally to the model in which they belong. To apply these statements globally to all models, put these statements before the first MODEL statement.

Other model analysis statements are not bounded by the scope of the MODEL statements. These statements are: FITINDEX, SIMTESTS, and TESTFUNC. Because these statements are not model-specific, you can put these statements anywhere in a PROC CALIS run.

Optimization and other statements are not bounded by the scope of either the GROUP or MODEL statements. You can specify them anywhere between the PROC CALIS and the run statements without affecting the parsing of the models and the groups. For clarity of presentation, they are shown as last statement block in the syntax. Notice that the BY statement is not supported in a multiple-group setting.

PROC CALIS Statement

PROC CALIS < *options* > **;**

The PROC CALIS statement invokes the CALIS procedure. There are many *options* in the PROC CALIS statement. These *options*, together with brief descriptions, are classified into different categories in the next few sections. An alphabetical listing of these *options* with more details then follows.

Data Set Options

You can use the following *options* to specify input and output data sets:

Option	Description
BASEFIT=	Inputs the fit information of the customized baseline model
DATA=	Inputs the data
INEST=	Inputs the initial values and constraints
INMODEL=	Inputs the model specifications
INWGT=	Inputs the weight matrix
OUTEST=	Outputs the estimates and their covariance matrix
OUTFIT=	Outputs the fit indices
OUTMODEL=	Outputs the model specifications
OUTSTAT=	Outputs the statistical results
OUTWGT=	Outputs the weight matrix
READADDPARM	Inputs the generated default parameters in the INMODEL= data set

Model and Estimation Options

You can use these *options* to specify details about estimation, models, and computations:

Option	Description
CORR	Analyzes correlation matrix
COV	Analyzes covariance matrix
COVPATTERN=	Specifies one of the built-in covariance structures
DEMPHAS=	Emphasizes the diagonal entries
EDF=	Defines number of observations by the number of error degrees of freedom
INFORMATION=	Specifies the method of computing the information matrix
INWGTINV	Specifies that the INWGT= data set contains the inverse of the weight matrix
MEANPATTERN=	Specifies one of the built-in mean patterns
MEANSTR	Analyzes the mean structures
METHOD=	Specifies the estimation method
NOBS=	Defines the number of observations
NOMEANSTR	Deactivates the inherited MEANSTR option
RANDOM=	Specifies the seed for randomly generated initial values
RDF=	Defines nobs by the number of regression *df*
RIDGE=	Specifies the ridge factor for the covariance matrix
ROBPHI=	Specifies the tuning parameter for robust methods
ROBUST=	Specifies the type of robust method
SBNTW=	Specifies the covariance matrix used in the weight matrix for METHOD=MLSB
STDERR=	Computes the standard errors and specifies the computational method
START=	Specifies a constant for initial values
VARDEF=	Specifies the variance divisor
WPENALTY=	Specifies the penalty weight to fit correlations
WRIDGE=	Specifies the ridge factor for the weight matrix

Options for Fit Statistics

You can use these *options* to modify the default behavior of fit index computations, to control the display of fit indices, and specify output file for fit indices:

Option	Description
ALPHAECV=	Specifies the α level for computing the confidence interval of ECV (Browne and Cudeck 1993)
ALPHARMS=	Specifies the α level for computing the confidence interval of RMSEA (Steiger and Lind 1980)
BASEFUNC=	Specifies the function value and the degrees of freedom of the customized baseline model
CHICORRECT=	Specifies the chi-square correction factor
CLOSEFIT=	Defines the close fit value
DFREDUCE=	Reduces the degrees of freedom for model fit chi-square test
NOADJDF	Requests no degrees-of-freedom adjustment be made for active constraints
NOINDEXTYPE	Suppresses the printing of fit index types
OUTFIT=	Specifies the output data set for storing fit indices

These *options* can also be specified in the FITINDEX statement. However, to control the display of individual fit indices, you must use the ON= and OFF= options of the FITINDEX statement.

Options for Statistical Analysis

You can use these *options* to request specific statistical analysis and display and to set the parameters for statistical analysis:

Option	Description
ALPHA=	Specifies the α-level for confidence intervals
ALPHALEV=	Specifies the α-level criterion for detecting leverage points
ALPHAOUT=	Specifies the α-level criterion for detecting outliers
ASYCOV=	Specifies the formula for computing asymptotic covariances
BIASKUR	Computes the skewness and kurtosis without bias corrections
CI	Prints the confidence limits
COVDIAG	Counts the occurrences of problematic variance and covariance estimates
EFFPART | TOTEFF	Displays total, direct, and indirect effects
EXTENDPATH	Displays the extended path estimates
G4=	Specifies the algorithm for computing standard errors
KURTOSIS	Computes and displays kurtosis
MAXLEVERAGE=	Specifies the maximum number of leverage observations to display
MAXMISSPAT=	Specifies the maximum number of missing patterns to display
MAXOUTLIER=	Specifies the maximum number of outliers to display
MODIFICATION	Computes modification indices
NOMISSPAT	Suppresses the display of missing pattern analysis
NOMOD	Suppresses modification indices
NOSTAND	Suppresses the standardized output
NOSTDERR	Suppresses standard error computations
PCORR	Displays analyzed and estimated moment matrix
PCOVES	Displays the covariance matrix of estimates
PDETERM	Computes the determination coefficients
PESTIM	Prints parameter estimates
PINITIAL	Prints initial pattern and values
PLATCOV	Computes the latent variable covariances and score coefficients
PLOTS=	Specifies ODS Graphics selection
PWEIGHT	Displays the weight matrix
RESIDUAL=	Specifies the type of residuals being computed
SIMPLE	Prints univariate statistics
SLMW=	Specifies the probability limit for Wald tests
STDERR	Computes the standard errors
TMISSPAT=	Specifies the data proportion threshold for displaying the missing patterns

Global Display Options

There are two different kinds of global display options: one is for selecting output; the other is for controlling the format or order of output.

You can use the following *options* to select printed output:

Option	Description
NOPRINT	Suppresses the displayed output
PALL	Displays all displayed output (ALL)
PRINT	Adds default displayed output
PSHORT	Reduces default output (SHORT)
PSUMMARY	Displays fit summary only (SUMMARY)

In contrast to individual output printing options described in the section "Options for Statistical Analysis" on page 1539, the global display options typically control more than one output or analysis. The relations between these two types of *options* are summarized in the following table:

Options	**PALL**	**PRINT**	**default**	**PSHORT**	**PSUMMARY**
fit indices	*	*	*	*	*
linear dependencies	*	*	*	*	*
PESTIM	*	*	*	*	
iteration history	*	*	*	*	
PINITIAL	*	*	*		
SIMPLE	*	*	*		
STDERR	*	*	*		
EFFPART	*	*			
KURTOSIS	*	*			
PLATCOV	*	*			
RESIDUAL	*	*			
COVDIAG	*				
MODIFICATION	*				
PCORR	*				
PWEIGHT	*				
PCOVES					
PDETERM					
PRIMAT					

Each column in the table represents a global display option. An "*" in the column means that the individual output or analysis option listed in the corresponding row turns on when the global display option in the corresponding column is specified.

Note that the column labeled with "default" is for default printing. If the NOPRINT option is not specified, a default set of output is displayed. The PRINT and PALL options add to the default output, while the PSHORT and PSUMMARY options reduce from the default output.

Note also that the PCOVES, PDETERM, and PRIMAT options cannot be turned on by any global display options. They must be specified individually.

The following global display options are for controlling formats and order of the output:

Option	Description
NOORDERSPEC	Displays model specifications and results according to the input order
NOPARMNAME	Suppresses the printing of parameter names in results
ORDERALL	Orders all output displays according to the model numbers, group numbers, and parameter types
ORDERGROUPS	Orders the group output displays according to the group numbers
ORDERMODELS	Orders the model output displays according to the model numbers
ORDERSPEC	Orders the model output displays according to the parameter types within each model
PARMNAME	Displays parameter names in model specifications and results
PRIMAT	Displays estimation results in matrix form

Optimization Options

You can use the following *options* to control the behavior of the optimization. Most of these *options* are also available in the NLOPTIONS statement.

Option	Description
ASINGULAR=	Specifies the absolute singularity criterion for inverting the information matrix
COVSING=	Specifies the singularity tolerance of the information matrix
FCONV=	Specifies the relative function convergence criterion
GCONV=	Specifies the gradient convergence criterion
INSTEP=	Specifies the initial step length (RADIUS=, SALPHA=)
LINESEARCH=	Specifies the line-search method
LSPRECISION=	Specifies the line-search precision (SPRECISION=)
MAXFUNC=	Specifies the maximum number of function calls
MAXITER=	Specifies the maximum number of iterations
MSINGULAR=	Specifies the relative M singularity of the information matrix
OMETHOD=	Specifies the minimization method
ROBITER=	Specifies the maximum number of iterations for estimating robust covariance and mean matrices
SINGULAR=	Specifies the singularity criterion for matrix inversion
UPDATE=	Specifies the update method for some optimization techniques
VSINGULAR=	Specifies the relative V singularity of information matrix
XCONV=	Specifies the relative parameter convergence criterion

PROC CALIS Statement Options

ALPHA=α

specifies that interval estimation of parameters be done at the $(1-\alpha)100\%$ confidence level. The smaller the α value, the higher the confidence level. By default, $\alpha = 0.05$, which corresponds to a 95% confidence interval.

ALPHAECV=α

specifies a $(1-\alpha)100\%$ confidence interval ($0 \leq \alpha \leq 1$) for the Browne and Cudeck (1993) expected cross-validation index (ECVI). By default, $\alpha = 0.1$, which corresponds to a 90% confidence interval for the ECVI.

ALPHALEV=α

ALPHALEVERAGE=α

specifies the α-level criterion for detecting leverage observations (or leverage points) in case-level (observation-level) residual diagnostics. The default ALPHALEV= value is 0.01. An observation is a leverage observation if the p-value of its squared Mahalanobis distance (M-distance) for its predictor variables (including observed and latent variables) is smaller than the specified α-level, where the p-value is computed according to an appropriate theoretical chi-square distribution. The larger the ALPHALEV= value, the more liberal the criterion for detecting leverage observations.

In addition to displaying the leverage observations as defined by the ALPHALEV= criterion, PROC CALIS also displays the next 5 observations with the largest leverage M-distances for reference. However, the total number of observations in the displayed output cannot exceed 30 or the number of original observations, whichever is smaller.

This option is relevant only when residual analysis is requested with the RESIDUAL option and with raw data input.

ALPHAOUT=α

ALPHAOUTLIER=α

specifies the α-level criterion for detecting outliers in case-level (observation-level) residual diagnostics. The default ALPHAOUT= value is 0.01. An observation is an outlier if the p-value of its squared residual M-distance is smaller than the specified α-level, where the p-value is computed according to an appropriate theoretical chi-square distribution. The larger the ALPHAOUT= value, the more liberal the criterion for detecting outliers.

In addition to displaying the outliers as defined by the ALPHAOUT= criterion, PROC CALIS also displays the next 5 observations with the largest residual M-distances for reference. However, the total number of observations in the displayed output cannot exceed 30 or the number of original observations, whichever is smaller.

This option is relevant only when residual analysis is requested with the RESIDUAL option and with raw data input.

ALPHARMS=α

ALPHARMSEA=α

specifies a $(1-\alpha)100\%$ confidence interval ($0 \leq \alpha \leq 1$) for the Steiger and Lind (1980) root mean square error of approximation (RMSEA) coefficient (see Browne and Du Toit 1992). The default value is $\alpha = 0.1$, which corresponds to a 90% confidence interval for the RMSEA.

ASINGULAR=*r*

ASING=*r*

specifies an absolute singularity criterion r ($r > 0$), for the inversion of the information matrix, which is needed to compute the covariance matrix. The default value for r or ASING= is the square root of the smallest positive double precision value.

When inverting the information matrix, the following singularity criterion is used for the diagonal pivot $d_{j,j}$ of the matrix:

$$|d_{j,j}| \leq \max(ASING, VSING * |H_{j,j}|, MSING * \max(|H_{1,1}|, \ldots, |H_{n,n}|))$$

where *VSING* and *MSING* are the specified values in the VSINGULAR= and MSINGULAR= options, respectively, and $H_{j,j}$ is the *j*th diagonal element of the information matrix. Note that in many cases a normalized matrix $\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}$ is decomposed (where $\mathbf{D}^2 = \mathrm{diag}(\mathbf{H})$), and the singularity criteria are modified correspondingly.

ASYCOV=*name*

ASC=*name*

specifies the formula for asymptotic covariances used in the weight matrix **W** for WLS and DWLS estimation. The ASYCOV option is effective only if METHOD= WLS or METHOD=DWLS and no INWGT= input data set is specified. The following formulas are implemented:

BIASED: Browne (1984) formula (3.4)
biased asymptotic covariance estimates; the resulting weight matrix is at least positive semidefinite. This is the default for analyzing a covariance matrix.

UNBIASED: Browne (1984) formula (3.8)
asymptotic covariance estimates corrected for bias; the resulting weight matrix can be indefinite (that is, can have negative eigenvalues), especially for small *N*.

CORR: Browne and Shapiro (1986) formula (3.2)
(identical to De Leeuw (1983) formulas (2,3,4)) the asymptotic variances of the diagonal elements are set to the reciprocal of the value *r* specified by the WPENALTY= option (default: *r*=100). This formula is the default for analyzing a correlation matrix.

By default, AYSCOV=BIASED is used for covariance analyses and ASYCOV=CORR is used for correlation analyses. Therefore, in almost all cases you do not need to set the ASYCOV= option once you specify the covariance or correlation analysis by the COV or CORR option.

BASEFIT=*SAS-data-set*

INBASEFIT=*SAS-data-set*

inputs the *SAS-data-set* that contains the fit information of the baseline model of your choice. This customized baseline model replaces the default uncorrelatedness model for computing several fit indices of your target model. Typically, you create the BASEFIT= data set by using the OUTFIT= option in a previous PROC CALIS fitting of your customized baseline model. Using the BASEFIT= option assumes that you fit your customized baseline model and your target model with the same data, number of groups (for multiple-group analysis), and estimation method. Typically, your baseline model should be more restrictive (or have fewer parameters) than your target model.

For example, the following statements use the compound symmetry model (COVPATTERN=COMPSYM) as the customized baseline model for the subsequent factor model with two factors:

```
proc calis data=abc outfit=outf method=gls covpattern=compsym;
   var=x1-x10;
run;
```

```
proc calis data=abc method=gls basefit=outf;
   factor n=2;
   var=x1-x10;
run;
```

The fit information of the customized baseline model is saved as an OUTFIT= data set called outf, which is then read as a BASEFIT= data set in the next PROC CALIS run for fitting the target factor model. Notice that in this example the baseline model and the target factor model use the same data set, abc, and the same GLS estimation method.

Alternatively, you can use the BASEFUNC= option to input the function value and the degrees of freedom of your customized baseline model. See the BASEFUNC= option for details. The BASEFIT= option is ignored if you also specify the BASEFUNC= option.

Notice that the fit information in the BASEFIT= data set does not affect the computation of all fit indices. Mainly, it affects the incremental fit indices, because these indices are defined as the incremental fit of a target model over a baseline model. Among all absolute and parsimonious fit indices, only the parsimonious goodness-of-fit (PGFI) index (Mulaik et al. 1989) is affected by the input fit information provided in the BASEFIT= data set.

If you specify METHOD=LSDWLS, LSFIML, LSGLS, LSML, or LSMLSB for your target model, the fit information in the BASEFIT= data set is assumed to have been obtained from the DWLS, FIML, GLS, ML, or MLSB estimation of your customized baseline model. Hence, the fit information in the BASEFIT= data set applies only to the second estimation of your target model. The unweighted least squares (ULS) estimation of the target model still uses the uncorrelatedness model as the baseline model for computing fit indices.

If you specify METHOD=MLSB or LSMLSB for your target model, the fit information in the BASEFIT= data set is assumed to have been obtained from an ML estimation with the Satorra-Bentler scale correction on the model fit chi-square (METHOD=MLSB). The BASEFIT= data set should contain the information about the unadjusted chi-square and the SB-scaled chi-square so that the computation of the fit indices can take the scale correction into account.

If you use a BASEFIT= data set to input the fit information of your customized baseline model in a multiple-group analysis, then the baseline model function values, chi-squares, and degrees of freedom for individual groups are not known and hence not displayed in the multiple-group fit summary table. The Bentler-Bonett normed fit index (NFI) is also not displayed in the multiple-group fit summary table, because the baseline model function values for individual groups are not saved in the BASEFIT= data set.

BASEFUNC=*r*(< DF= >*i*)

BASEFUNC(< DF= >*i*)=*r*

inputs the fit function value *r* and the degrees of freedom *i* of the baseline model of your choice. This customized baseline model replaces the default uncorrelatedness model for computing several fit indices of your target model. To use this option, you must first fit your customized baseline model and then use this option to input the baseline model fit information when you fit your target model.

Using the BASEFUNC= option assumes that you fit your customized baseline model and your target model with the same data, number of groups (for multiple-group analysis), and estimation method. Typically, your baseline model should be more restrictive (or have fewer parameters) than your target model.

For example, assume that after fitting a customized baseline model you find that the function value of the baseline model is 20.54 and the model degrees of freedom are 15. The following code inputs the function value and the degrees of freedom of the customized baseline model by using the BASEFUNC= option:

```
proc calis data=abc basefunc(df=15)=20.54;
   path
     f1 ===> x1-x5   = 1.,
     f2 ===> x6-x10  = 1.,
     f1 ===> f2;
run;
```

You can use the following equivalent syntax to provide the same baseline model fit information:

```
basefunc(df=15)=20.54
basefunc(15)=20.54
basefunc=20.54(df=15)
basefunc=20.54(15)
```

It is emphasized here that you should input the fit function value, but not the model fit chi-square value, of the baseline model in the BASEFUNC= option. For all estimation methods except the full information maximum likelihood (FIML) method in PROC CALIS, the model fit chi-square values are some multiples of the fit function values. See the section "Estimation Criteria" on page 1805 for the definitions of the various fit functions that are assumed by the BASEFUNC= option.

Alternatively, it might be easier to use the BASEFIT= option to specify the SAS data set that contains the baseline model fit information. Such a SAS data set is typically created by using the OUTFIT= option in the PROC CALIS fitting of your customized baseline model. See the BASEFIT= option for details. However, the BASEFIT= option is ignored if you also specify the BASEFUNC= option.

Notice that the specified values in the BASEFUNC= option do not affect the computation of all fit indices. Mainly, they affect the incremental fit indices, because these indices are defined as the incremental fit of a target model over a baseline model. Among all absolute and parsimonious fit indices, only the parsimonious goodness-of-fit (PGFI) index (Mulaik et al. 1989) is affected by the values provided in the BASEFUNC= option.

If you specify METHOD=LSDWLS, LSFIML, LSGLS, LSML, or LSMLSB for your target model, the fit information that the BASEFUNC= option provides is assumed to have been obtained from the DWLS, FIML, GLS, ML, or MLSB estimation of your customized baseline model. Hence, the fit information that the BASEFUNC= option provides applies only to the second estimation of your target model. The unweighted least squares (ULS) estimation of the target model still uses the uncorrelatedness model as the baseline model for computing fit indices.

If you specify METHOD=MLSB or LSMLSB for your target model, you must be careful to input an appropriate BASEFUNC= value. Because the BASEFUNC= value is used directly to compute fit indices, the Satorra-Bentler scaling constant for the baseline model is unavailable to PROC CALIS for computing various fit indices. An appropriate step is to rescale the original baseline model fit function value before inputting it as the BASEFUNC= value. For example, suppose that you fitted a baseline model by using METHOD=MLSB. The function value was f, the unadjusted chi-square was u, and the Satorra-Bentler scaled chi-square was s. Instead of using f, you should use the value of $f \times s/u$ as the

input BASEFUNC= value. This way the fit indices that are computed for the target model would be based on the scaled chi-squares of the baseline and target models.

If you use the BASEFUNC= option to input the fit information of your customized baseline model in a multiple-group analysis, then the baseline model function values, chi-squares, and degrees of freedom for individual groups are not known and hence not displayed in the multiple-group fit summary table. The Bentler-Bonett NFI is also not displayed in the multiple-group fit summary table, because the baseline model function values for individual groups are not provided with the BASEFUNC= option.

BIASKUR

computes univariate skewness and kurtosis by formulas uncorrected for bias.

See the section "Measures of Multivariate Kurtosis" on page 1851 for more information.

CHICORRECT= *name* | *c*

CHICORR= *name* | *c*

specifies a correction factor *c* for the chi-square statistics for model fit. You can specify a *name* for a built-in correction factor or a value between 0 and 1 as the CHICORRECT= value. The model fit chi-square statistic is computed as:

$$\chi^2 = (1 - c)(N - k)F$$

where N is the total number of observations, k is the number of independent groups, and F is the optimized function value. Application of these correction factors requires appropriate specification of the covariance structural model suitable for the chi-square correction. For example, using CHICORRECT=UNCORR assumes that you are fitting a covariance structure with free parameters on the diagonal elements and fixed zeros off-diagonal elements of the covariance matrix. Because all the built-in correction factors assume multivariate normality in their derivations, the appropriateness of applying these built-in chi-square corrections to estimation methods other than METHOD= is not known. The CHICORRECT= option is ignored when you specify METHOD=MLSB, which has its own correction factor on the model fit chi-square statistics.

Valid *names* for the CHICORRECT= value are as follows:

COMPSYM | EQVARCOV specifies the correction factor due to Box (1949) for testing equal variances and equal covariances in a covariance matrix. The correction factor is:

$$c = \frac{p(p+1)^2(2p-3)}{6n(p-1)(p^2+p-4)}$$

where p ($p > 1$) represents the number of variables and $n = (N - 1)$, with N denoting the number of observations in a single group analysis. This option is not applied when you also analyze the mean structures or when you fit multiple-group models.

EQCOVMAT specifies the correction factor due to Box (1949) for testing equality of covariance matrices. The correction factor is:

$$c = \frac{2p^2+3p-1}{6(p+1)(k-1)}\left(\sum_{i=1}^{k}\frac{1}{n_i} - \frac{1}{\sum_{i=1}^{k} n_i}\right)$$

where p represents the number of variables, k ($k > 1$) represents the number of groups, and $n_i = (N_i - 1)$, with N_i denoting the number of observations in the

*i*th group. This option is not applied when you also analyze the mean structures or when you fit single-group models.

FIXCOV specifies the correction factor due to Bartlett (1954) for testing a covariance matrix against a hypothetical fixed covariance matrix. The correction factor is:

$$c = \frac{1}{6n}(2p + 1 - \frac{2}{p+1})$$

where p represents the number of variables and $n = (N - 1)$, with N denoting the number of observations in a single group analysis. This option is not applied when you also analyze the mean structures or when you fit multiple-group models.

SPHERICITY specifies the correction factor due to Box (1949) for testing a spherical covariance matrix (Mauchly 1940). The correction factor is:

$$c = \frac{2p^2 + p + 2}{6np}$$

where p represents the number of variables and $n = (N - 1)$, with N denoting the number of observations in a single group analysis. This option is not applied when you also analyze the mean structures or when you fit multiple-group models.

TYPEH specifies the correction factor for testing the H pattern (Huynh and Feldt 1970) directly. The correction factor is:

$$c = \frac{2p^2 - 3p + 3}{6n(p - 1)}$$

where p ($p > 1$) represents the number of variables and $n = (N - 1)$, with N denoting the number of observations in a single group analysis. This option is not applied when you also analyze the mean structures or when you fit multiple-group models.

This correction factor is derived by substituting p with $p - 1$ in the correction formula applied to Mauchly's sphericity test. The reason is that testing the H pattern of p variables is equivalent to testing the sphericity of the $(p - 1)$ orthogonal contrasts of the same set of variables (Huynh and Feldt 1970). See pp. 295–296 of Morrison (1990) for more details.

UNCORR specifies the correction factor due to Bartlett (1950) and Box (1949) for testing a diagonal pattern of a covariance matrix, while the diagonal elements (variances) are unconstrained. This test is sometimes called Bartlett's test of sphericity—not to be confused with the sphericity test dues to Mauchly (1940), which requires all variances in the covariance matrix to be equal. The correction factor is:

$$c = \frac{2p + 5}{6n}$$

where p represents the number of variables and $n = (N - 1)$, with N denoting the number of observations in a single group analysis. This option is not applied when you also analyze the mean structures or when you fit multiple-group models.

CI

CL

prints the confidence intervals or limits for parameters. By default, PROC CALIS estimates the lower and upper confidence limits at the 95% confidence level, which corresponds to the default value of the ALPHA= option (0.05). You can change the default level of confidence by specifying the ALPHA= option.

CLOSEFIT=*p*

defines the criterion value *p* for indicating a close fit. The smaller the better fit. The default value for close fit is .05.

CORR

CORRELATION

analyzes the correlation matrix, instead of the default covariance matrix. See the COV option for more details.

COV

COVARIANCE

analyzes the covariance matrix. Because this is also the default analysis in PROC CALIS, you can simply omit this option when you analyze covariance rather than correlation matrices. If the DATA= input data set is a TYPE=CORR data set (containing a correlation matrix and standard deviations), the default COV option means that the covariance matrix is computed and analyzed.

Unlike many other SAS/STAT procedures (for example, the FACTOR procedure) that analyze correlation matrices by default, PROC CALIS uses a different default because statistical theories of structural equation modeling or covariance structure analysis are mostly developed for covariance matrices. You must use the CORR option if correlation matrices are analyzed.

COVDIAG< (ALL) >

displays tables that count the occurrences of problematic variance and covariance estimates, which include the following cases:

- zero variance estimates
- negative variance estimates
- zero eigenvalues in predicted covariance matrices
- negative eigenvalues in predicted covariance matrices

PROC CALIS examines the following four predicted covariance matrices for problematic cases:

- covariance matrix of observed variables only
- covariance matrix of latent factors only
- covariance matrix of observed variables and latent factors
- covariance matrix of error variables (variance estimates only)

Although PROC CALIS examines both the variance estimates and the eigenvalues of the first three covariance matrices, it examines only the variance estimates (but not the eigenvalues) of the error covariance matrix. In addition, PROC CALIS excludes those fixed zero or negative variances in models when counting problematic cases.

The COVDIAG and COVDIAG(ALL) options are the same for single-group (single-model) analysis. They both count the occurrences of problematic cases in the only model. When there are multiple models, the two options display an overall table that aggregates the problematic cases in all models. However, only the COVDIAG(ALL) option displays the frequency counts of problematic cases for individual models.

COVPATTERN=*name*

COVPAT=*name*

specifies one of the built-in covariance structures for the data. The purpose of this option is to fit some commonly-used direct covariance structures efficiently without the explicit use of the MSTRUCT model specifications. With this option, the covariance structures are defined internally in PROC CALIS. The following *names* for the built-in covariance structures are supported:

COMPSYM | EQVARCOV specifies the compound symmetry pattern for the covariance matrix. That is, a covariance matrix with equal variances for all variables and equal covariance between any pairs of variables (EQVARCOV). For example, if there are four variables in the analysis, the covariance pattern generated by PROC CALIS has the following form:

$$\mathbf{\Sigma} = \begin{pmatrix} v & c & c & c \\ c & v & c & c \\ c & c & v & c \\ c & c & c & v \end{pmatrix}$$

PROC CALIS denotes the common variance parameter, v, by _varparm, and the common covariance parameter, v, by _covparm.

If you request a single-group maximum likelihood (METHOD=ML) covariance structure analysis by specifying the COVPATTERN=COMPSYM or COVPATTERN=EQVARCOV option and the mean structures are not modeled, the chi-square correction due to Box (1949) is applied automatically when the number of variables is greater than or equal to 2. See the CHICORRECT=COMPSYM option for the definition of the correction factor.

EQCOVMAT specifies the equality of covariance matrices between multiple groups. That is, this option tests the null hypothesis that

$$H_0 : \mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \ldots = \mathbf{\Sigma}_k = \mathbf{\Sigma}$$

where $\mathbf{\Sigma}$ is a common covariance matrix for the k $\mathbf{\Sigma}_j$'s ($j = 1, \ldots, k; k > 1$). The elements of $\mathbf{\Sigma}$ are named _cov_xx_yy automatically by PROC CALIS, where xx and yy represents the row and column numbers such that xx is at least as large as yy. For example, if there are four variables in the analysis, the (1,1) element of $\mathbf{\Sigma}$ is denoted by _cov_1_1, the (4,3) or (3,4) element of $\mathbf{\Sigma}$ is denoted by _cov_4_3, and so on.

If you request a multiple-group maximum likelihood (METHOD=ML) covariance structure analysis by specifying the COVPATTERN=EQCOVMAT and the mean structures are not modeled, the chi-square correction due to Box (1949) is applied automatically. See the CHICORRECT=EQCOVMAT option for the definition of the correction factor.

SATURATED specifies a saturated covariance structure model. This is the default option when you specify the MEANPATTERN= option without using the COVPATTERN= option. The elements of $\mathbf{\Sigma}$ are named _cov_xx_yy automatically by PROC CALIS, where xx represents the row number and yy represents the column number. For example, if there are three variables in the analysis, the (1,1) element in $\mathbf{\Sigma}$ is denoted by _cov_1_1, the (3,2) or (2,3) element in $\mathbf{\Sigma}$ is denoted by _cov_3_2, and so on.

SPHERICITY | SIGSQI specifies the spheric pattern of the covariance matrix (Mauchly 1940). That is, this option tests the null hypothesis that

$$H_0 : \mathbf{\Sigma} = \sigma^2 \mathbf{I}$$

where σ^2 is a common variance parameter and $\mathbf{I}$ is an identity matrix. For example, if there are three variables in the analysis, the covariance pattern generated by PROC CALIS is:

$$\mathbf{\Sigma} = \begin{pmatrix} v & 0 & 0 \\ 0 & v & 0 \\ 0 & 0 & v \end{pmatrix}$$

PROC CALIS denotes the common variance parameter, v, by _varparm.

If you request a single-group maximum likelihood (METHOD=ML) covariance structure analysis by specifying the COVPATTERN=SPHERICITY or COVPATTERN=SIGSQI option and the mean structures are not modeled, the chi-square correction due to Box (1949) is applied automatically. See the CHICORRECT=SPHERICITY option for the definition of the correction factor.

UNCORR | DIAG specifies the diagonal pattern of the covariance matrix. That is, this option tests the null hypothesis of uncorrelatedness—all correlations (or covariances) between variables are zero and the variances are unconstrained. For example, if there are three variables in the analysis, the covariance pattern generated by PROC CALIS is:

$$\mathbf{\Sigma} = \begin{pmatrix} v_1 & 0 & 0 \\ 0 & v_2 & 0 \\ 0 & 0 & v_3 \end{pmatrix}$$

PROC CALIS denotes the variance parameters v_1, v_2, and v_3 by _varparm_1, _varparm_2, and _varparm_3, respectively.

If you request a single-group maximum likelihood (METHOD=ML) covariance structure analysis by specifying the COVPATTERN=UNCORR or COVPATTERN=DIAG option and the mean structures are not modeled, the chi-square correction due to Bartlett (1950) is applied automatically. See the CHICORRECT=UNCORR option for the definition of the correction factor. Under the multivariate normal assumption, COVPATTERN=UNCORR is also a test of independence of the variables in the analysis.

When you specify the covariance structure model by means of the COVPATTERN= option, you can define the set of variables in the analysis by the VAR statement (either within the scope of the PROC CALIS statement or the GROUP statements). If the VAR statement is not used, PROC CALIS uses all numerical variables in the data sets.

Except for the EQCOVMAT pattern, all other built-in covariance patterns are primarily designed for single-group analysis. However, you can still use these covariance pattern options for multiple-group situations. For example, consider the following three-group analysis:

```
proc calis covpattern=compsym;
   group 1 / data=set1;
   group 2 / data=set2;
   group 3 / data=set3;
run;
```

In this specification, all three groups are fitted by the compound symmetry pattern. However, there would be no constraints across these groups. PROC CALIS generates two distinct parameters for each group: _varparm_mdl1 and _covparm_mdl1 for Group 1, _varparm_mdl2 and _covparm_mdl2 for Group 2, and _varparm_mdl3 and _covparm_mdl3 for Group 3. Similarly, the _mdlxx suffix, where xx represents the model number, is applied to the parameters defined by the SATURATED, SPHERICITY (or SIGSQI), and UNCORR (or DIAG) covariance patterns in multiple-group situations. However, chi-square correction, whenever it is applicable to single-group analysis, is not applied to such multiple-group analyses.

You can also apply the COVPATTERN= option partially to the groups in the analysis. For example, the following statements apply the spheric pattern to Group 1 and Group 2 only:

```
proc calis covpattern=sphericity;
   group 1 / data=set1;
   group 2 / data=set2;
   group 3 / data=set3;
   model 3 / group=3;
     path    x1 ===> y3;
run;
```

Group 3 is fitted by Model 3, which is specified explicitly by a PATH model with distinct covariance structures.

If the EQCOVMAT pattern is specified instead, as shown in the following statements, the equality of covariance matrices still holds for Groups 1 and 2:

```
proc calis covpattern=eqcovmat;
   group 1 / data=set1;
   group 2 / data=set2;
   group 3 / data=set3;
   model 3 / group=3;
     path    x1 ===> y3;
run;
```

However, Group 3 has it own covariances structures as specified in Model 3. In this case, the chi-square correction due to Box (1949) is not applied because the null hypothesis is no longer testing the equality of covariance matrices among the groups in the analysis.

Use the MEANPATTERN= option if you also want to analyze some built-in mean structures along with the covariance structures.

COVSING=*r*

specifies a nonnegative threshold *r*, which determines whether the eigenvalues of the information matrix are considered to be zero. If the inverse of the information matrix is found to be singular (depending on the VSINGULAR=, MSINGULAR=, ASINGULAR=, or SINGULAR= option), a generalized inverse is computed using the eigenvalue decomposition of the singular matrix. Those eigenvalues smaller than *r* are considered to be zero. If a generalized inverse is computed and you do not specify the NOPRINT option, the distribution of eigenvalues is displayed.

DATA=*SAS-data-set*

specifies an input data set that can be an ordinary SAS data set or a specially structured TYPE=CORR, TYPE=COV, TYPE=UCORR, TYPE=UCOV, TYPE=SSCP, or TYPE=FACTOR SAS data set, as described in the section "Input Data Sets" on page 1711. If the DATA= option is omitted, the most recently created SAS data set is used.

DEMPHAS=*r*

DE=*r*

changes the initial values of all variance parameters by the relationship:

$$s_{new} = r(|s_{old}| + 1)$$

where s_{new} is the new initial value and s_{old} is the original initial value. The value of *r* must be positive. If you specify an *r* value less than 1E–8, it is replaced with 1E–8.

The initial values of all variance parameters should always be nonnegative to generate positive definite predicted model matrices in the first iteration. By using values of $r > 1$, for example, $r = 2$, $r = 10$, and so on, you can increase these initial values to produce predicted model matrices with high positive eigenvalues in the first iteration. The DEMPHAS= option is effective independent of the way the initial values are set; that is, it changes the initial values set in the model specification as well as those set by an INMODEL= data set and those automatically generated for the FACTOR, LINEQS, LISMOD, PATH, or RAM models. It also affects the initial values set by the START= option, which uses, by default, DEMPHAS=100 if a covariance matrix is analyzed and DEMPHAS=10 for a correlation matrix.

DFREDUCE=*i*

DFRED=*i*

reduces the degrees of freedom of the model fit χ^2 test by *i*. In general, the number of degrees of freedom is the total number of nonredundant elements in all moment matrices minus the number of parameters, *t*. Because negative values of *i* are allowed, you can also increase the number of degrees of freedom by using this option.

EDF=*n*

DFE=*n*

makes the effective number of observations $n + 1$. You can also use the NOBS= option to specify the number of observations.

EFFPART

TOTEFF

TE

computes and displays total, direct, and indirect effects for the unstandardized and standardized estimation results. Standard errors for the effects are also computed. Note that this displayed output is not automatically included in the output generated by the PALL option.

Note also that in some situations computations of total effects and their partitioning are not appropriate. While total and indirect effects must converge in recursive models (models with no cyclic paths among variables), they do not always converge in nonrecursive models. When total or indirect effects do not converge, it is not appropriate to partition the effects. Therefore, before partitioning the total effects, the convergence criterion must be met. To check the convergence of the effects, PROC CALIS computes and displays the "stability coefficient of reciprocal causation"— that is, the largest modulus of the eigenvalues of the $\boldsymbol{\beta}$ matrix, which is the square matrix that contains the path coefficients of all endogenous variables in the model. Stability coefficients less than one provide a necessary and sufficient condition for the convergence of the total and the indirect effects. Otherwise, PROC CALIS does not show results for total effects and their partitioning. See the section "Stability Coefficient of Reciprocal Causation" on page 1846 for more information about the computation of the stability coefficient.

EXTENDPATH

GENPATH

displays the extended path estimates such as the variances, covariances, means, and intercepts in the table that contains the ordinary path effect (coefficient) estimates. This option applies to the PATH model only.

FCONV=*r*

FTOL=*r*

specifies the relative function convergence criterion. The optimization process is terminated when the relative difference of the function values of two consecutive iterations is smaller than the specified value of *r*; that is,

$$\frac{|f(x^{(k)}) - f(x^{(k-1)})|}{\max(|f(x^{(k-1)})|, FSIZE)} \leq r$$

where *FSIZE* can be defined by the FSIZE= option in the NLOPTIONS statement. The default value is $r = 10^{-FDIGITS}$, where *FDIGITS* either can be specified in the NLOPTIONS statement or is set by default to $- < \log_{10}(\epsilon)$, where ϵ is the machine precision.

G4=*i*

instructs that the algorithm to compute the approximate covariance matrix of parameter estimates used for computing the approximate standard errors and modification indices when the information matrix is singular. If the number of parameters *t* used in the model you analyze is smaller than the value of *i*, the time-expensive Moore-Penrose (G4) inverse of the singular information matrix is computed by eigenvalue decomposition. Otherwise, an inexpensive pseudo (G1) inverse is computed by sweeping. By default, *i* = 60.

See the section "Estimation Criteria" on page 1805 for more details.

GCONV=*r*

GTOL=*r*

specifies the relative gradient convergence criterion.

Termination of all techniques (except the CONGRA technique) requires the following normalized predicted function reduction to be smaller than *r*. That is,

$$\frac{[g(x^{(k)})]'[\mathbf{G}^{(k)}]^{-1}g(x^{(k)})}{\max(|f(x^{(k)})|, FSIZE)} \leq r$$

where *FSIZE* can be defined by the FSIZE= option in the NLOPTIONS statement. For the CONGRA technique (where a reliable Hessian estimate **G** is not available),

$$\frac{\| g(x^{(k)}) \|_2^2 \quad \| s(x^{(k)}) \|_2}{\| g(x^{(k)}) - g(x^{(k-1)}) \|_2 \max(|f(x^{(k)})|, FSIZE)} \leq r$$

is used. The default value is $r = 10^{-8}$.

INEST=*SAS-data-set*

INVAR=*SAS-data-set*

ESTDATA=*SAS-data-set*

specifies an input data set that contains initial estimates for the parameters used in the optimization process and can also contain boundary and general linear constraints on the parameters. Typical applications of this option are to specify an OUTEST= data set from a previous PROC CALIS analysis. The initial estimates are taken from the values of the PARMS observation in the INEST= data set.

INFORMATION=*name*

INFORM=*name*

specifies the type of information matrix from which the standard errors are computed. You can specify the following *names*:

EXPECTED | EXP requests that the expected information be used.

OBSERVED | OBS requests that the observed information be used.

By default, METHOD=FIML uses INFORMATION=OBSERVED, whereas METHOD=ML, MLSB, or GLS uses INFORMATION=EXPECTED. You overwrite the default by using this option. However, for METHOD=WLS, you can specify only INFORMATION=EXPECTED.

INMODEL=*SAS-data-set*

INRAM=*SAS-data-set*

specifies an input data set that contains information about the analysis model. A typical use of the INMODEL= option is when you run an analysis with its model specifications saved as an OUTMODEL= data set from a previous PROC CALIS run. Instead of specifying the main or subsidiary model specification statements in the new run, you use the INMODEL= option to input the model specification saved from the previous run.

Sometimes, you might create an INMODEL= data set from modifying an existing OUTMODEL= data set. However, editing and modifying OUTMODEL= data sets requires good understanding of the formats and contents of the OUTMODEL= data sets. This process could be error-prone for novice

users. For details about the format of INMODEL= or OUTMODEL= data sets, see the section"Input Data Sets" on page 1711.

It is important to realize that INMODEL= or OUTMODEL= data sets contain only the information about the specification of the model. These data sets do not store any information about the bounds on parameters, linear and nonlinear parametric constraints, and programming statements for computing dependent parameters. If required, these types of information must be provided in the corresponding statement specifications (for example, BOUNDS, LINCON, and so on) in addition to the INMODEL = data set.

An OUTMODEL= data set might also contain default parameters added automatically by PROC CALIS from a previous run (for example, observations with _TYPE_=ADDPCOV, ADDMEAN, or ADDPVAR). When reading the OUTMODEL= model specification as an INMODEL= data set in a new run, PROC CALIS ignores these added parameters so that the model being read is exactly like the previous PROC CALIS specification (that is, before default parameters were added automatically). After interpreting the specification in the INMODEL= data set, PROC CALIS will then add default parameters appropriate to the new run. The purpose of doing this is to avoid inadvertent parameter constraints in the new run, where another set of automatic default parameters might have the same generated names as those of the generated parameter names in the INMODEL= data set.

If you want the default parameters in the INMODEL= data set to be read as a part of model specification, you must also specify the READADDPARM option. However, using the READADDPARM option should be rare.

INSTEP=*r*

For highly nonlinear objective functions, such as the EXP function, the default initial radius of the trust-region algorithms (TRUREG, DBLDOG, and LEVMAR) or the default step length of the line-search algorithms can produce arithmetic overflows. If an arithmetic overflow occurs, specify decreasing values of $0 < r < 1$ such as INSTEP=1E–1, INSTEP=1E–2, INSTEP=1E–4, and so on, until the iteration starts successfully.

- For trust-region algorithms (TRUREG, DBLDOG, and LEVMAR), the INSTEP option specifies a positive factor for the initial radius of the trust region. The default initial trust-region radius is the length of the scaled gradient, and it corresponds to the default radius factor of $r = 1$.
- For line-search algorithms (NEWRAP, CONGRA, and QUANEW), INSTEP specifies an upper bound for the initial step length for the line search during the first five iterations. The default initial step length is $r = 1$.

For more details, see the section "Computational Problems" on page 1862.

INWGT< (INV) >=*SAS-data-set*

INWEIGHT< (INV) >=*SAS-data-set*

specifies an input data set that contains the weight matrix **W** used in generalized least squares (GLS), weighted least squares (WLS, ADF), or diagonally weighted least squares (DWLS) estimation, if you do not specify the INV option at the same time. The weight matrix must be positive definite because its inverse must be defined in the computation of the objective function. If the weight matrix **W** defined by an INWGT= data set is not positive definite, it can be ridged using the WRIDGE= option. See the section "Estimation Criteria" on page 1805 for more information. If you specify the INWGT(INV)= option, the INWGT= data set contains the inverse of the weight matrix, rather than the weight matrix itself. Specifying the INWGT(INV)= option is equivalent to specifying the INWGT= and INWGTINV

options simultaneously. With the INWGT(INV)= specification, the input matrix is not required to be positive definite. See the INWGTINV option for more details. If no INWGT= data set is specified, default settings for the weight matrices are used in the estimation process. The INWGT= data set is described in the section "Input Data Sets" on page 1711. Typically, this input data set is an OUTWGT= data set from a previous PROC CALIS analysis.

INWGTINV

specifies that the INWGT= data set contains the inverse of the weight matrix, rather than the weight matrix itself. This option is effective only with an input weight matrix specified in the INWGT= data set and with the generalized least squares (GLS), weighted least squares (WLS or ADF), or diagonally weighted least squares (DWLS) estimation. With this option, the input matrix provided in the INWGT= data set is not required to be positive definite. Also, the ridging requested by the WRIDGE= option is ignored when you specify the INWGTINV option.

KURTOSIS

KU

computes and displays univariate kurtosis and skewness, various coefficients of multivariate kurtosis, and the numbers of observations that contribute most to the normalized multivariate kurtosis. See the section "Measures of Multivariate Kurtosis" on page 1851 for more information. Using the KURTOSIS option implies the SIMPLE display option. This information is computed only if the DATA= data set is a raw data set, and it is displayed by default if the PRINT option is specified. The multivariate least squares kappa and the multivariate mean kappa are displayed only if you specify METHOD=WLS and the weight matrix is computed from an input raw data set. All measures of skewness and kurtosis are corrected for the mean. Using the BIASKUR option displays the biased values of univariate skewness and kurtosis.

LINESEARCH=*i*

LIS=*i*

SMETHOD=*i*

SM=*i*

specifies the line-search method for the CONGRA, QUANEW, and NEWRAP optimization techniques. See Fletcher (1980) for an introduction to line-search techniques. The value of *i* can be any integer between 1 and 8, inclusively; the default is *i*=2.

1 specifies a line-search method that needs the same number of function and gradient calls for cubic interpolation and cubic extrapolation; this method is similar to one used by the Harwell subroutine library.

2 specifies a line-search method that needs more function calls than gradient calls for quadratic and cubic interpolation and cubic extrapolation; this method is implemented as shown in Fletcher (1987) and can be modified to an exact line search by using the LSPRECISION= option.

3 specifies a line-search method that needs the same number of function and gradient calls for cubic interpolation and cubic extrapolation; this method is implemented as shown in Fletcher (1987) and can be modified to an exact line search by using the LSPRECISION= option.

4 specifies a line-search method that needs the same number of function and gradient calls for stepwise extrapolation and cubic interpolation.

5 specifies a line-search method that is a modified version of LIS=4.

LSPRECISION=*r*
LSP=*r*
SPRECISION=*r*
SP=*r*

specifies the degree of accuracy that should be obtained by the line-search algorithms LIS=2 and LIS=3. Usually an imprecise line search is inexpensive and successful. For more difficult optimization problems, a more precise and more expensive line search might be necessary (Fletcher 1980, p. 22). The second (default for NEWRAP, QUANEW, and CONGRA) and third line-search methods approach exact line search for small LSPRECISION= values. If you have numerical problems, you should decrease the LSPRECISION= value to obtain a more precise line search. The default LSPRECISION= values are displayed in the following table.

OMETHOD=	UPDATE=	LSP default
QUANEW	DBFGS, BFGS	$r = 0.4$
QUANEW	DDFP, DFP	$r = 0.06$
CONGRA	all	$r = 0.1$
NEWRAP	no update	$r = 0.9$

For more details, see Fletcher (1980, pp. 25–29).

MAXFUNC=*i*
MAXFU=*i*

specifies the maximum number *i* of function calls in the optimization process. The default values are displayed in the following table.

OMETHOD=	MAXFUNC default
LEVMAR, NEWRAP, NRRIDG, TRUREG	$i = 125$
DBLDOG, QUANEW	$i = 500$
CONGRA	$i = 1000$

The default is used if you specify MAXFUNC=0. The optimization can be terminated only after completing a full iteration. Therefore, the number of function calls that is actually performed can exceed the number that is specified by the MAXFUNC= option.

MAXITER=*i* < *n* >
MAXIT=*i* < *n* >

specifies the maximum number *i* of iterations in the optimization process. Except for the iteratively reweighted least squares (IRLS) algorithm for the robust estimation of model parameters, the default values are displayed in the following table.

OMETHOD=	MAXITER default
LEVMAR, NEWRAP, NRRIDG, TRUREG	$i = 50$
DBLDOG, QUANEW	$i = 200$
CONGRA	$i = 400$

The default maximum number of iterations for IRLS is 5000. The default value is used if you specify MAXITER=0 or if you omit the MAXITER option.

The optional second value *n* is valid only for OMETHOD=QUANEW with nonlinear constraints. It specifies an upper bound *n* for the number of iterations of an algorithm and reduces the violation of nonlinear constraints at a starting point. The default is $n = 20$. For example, specifying

```
maxiter= . 0
```

means that you do not want to exceed the default number of iterations during the main optimization process and that you want to suppress the feasible point algorithm for nonlinear constraints.

MAXLEVERAGE=*n*

MAXLEV=*n*

specifies the maximum number of leverage observations to display in the output, where *n* is between 1 and 9,999. The default MAXLEVERAGE= value is 30. The actual numbers of leverage observations in the output could be smaller than the maximum. In general, PROC CALIS finds the number leverage points *m* and then adds the next 5 most leveraged observations in the output. The actual number of leverage observations shown in the output is either *m*+5, the MAXLEVERAGE= value, or the number of observations in the data set, whichever is smaller.

MAXMISSPAT=*n*

specifies the maximum number of missing patterns to display in the output, where *n* is between 1 and 9,999. The default MAXMISSPAT= value is 10 or the number of missing patterns in the data, whichever is smaller. The number of missing patterns to display cannot exceed this MAXMISSPAT= value. This option is relevant only when there are incomplete observations (with some missing values in the analysis variables) in the input raw data set and when you use METHOD=FIML or METHOD=LSFIML for estimation.

Because the number of missing patterns could be quite large, PROC CALIS displays a limited number of the most frequent missing patterns in the output. The MAXMISSPAT= and the TMISSPAT= options are used in determining the number of missing patterns to display. The missing patterns are ordered according to the data proportions they account for, from the largest to the smallest. PROC CALIS displays a minimum number of the highest-frequency missing patterns. This minimum number is the smallest among five, the actual number of missing patterns, and the MAXMISSPAT= value. Then, PROC CALIS displays the subsequent high-frequency missing patterns if the data proportion accounted for by each of these patterns is at least as large as the proportion threshold set by the TMISSPAT= value (default at 0.05) until the total number of missing patterns displayed reaches the maximum set by the MAXMISSPAT= option.

MAXOUTLIER=*n*

MAXOUT=*n*

specifies the maximum number of outliers to display in the output, where *n* is between 1 and 9,999. The default MAXOUTLIER= value is 30. The actual numbers of outliers displayed in the output could be smaller than the maximum. In general, PROC CALIS finds the number outliers *m* and then adds the next 5 most outlying observations in the output. The actual number of outliers shown in the output is either *m*+5, the MAXOUTLIER= value, or the number of observations in the data set, whichever is smaller.

MEANPATTERN=*name*

MEANPAT=*name*

specifies one of the built-in mean structures for the data. The purpose of this option is to fit some commonly-used direct mean structures efficiently without the explicit use of the MSTRUCT model specifications. With this option, the mean structures are defined internally in PROC CALIS. The following *names* for the built-in mean structures are supported:

EQMEANVEC specifies the equality of mean vectors between multiple groups. That is, this option tests the null hypothesis that

$$H_0 : \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \ldots = \boldsymbol{\mu}_k = \boldsymbol{\mu}$$

where $\boldsymbol{\mu}$ is a common mean vector for the k $\boldsymbol{\mu}_j$'s ($j = 1, \ldots, k$). For example, if there are four variables in the analysis, the common $\boldsymbol{\mu}$ is defined as

$$\boldsymbol{\mu} = \begin{pmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \end{pmatrix}$$

PROC CALIS denotes m_1, m_2, m_3, and m_4 by _mean_1, _mean_2, _mean_3, and _mean_4, respectively.

If you use the COVPATTERN=EQCOVMAT and MEANPATTERN= EQMEANVEC together in a maximum likelihood (METHOD=ML) analysis, you are testing a null hypothesis of the same multivariate normal distribution for the groups.

If you use the MEANPATTERN=EQMEANVEC option for a single-group analysis, the parameters for the single group are still created accordingly. However, the mean model for the single group contains only unconstrained parameters that would result in saturated mean structures for the model.

SATURATED specifies a saturated mean structure model. This is the default mean structure pattern when the covariance structures are specified by the COVPATTERN= pattern and the mean structure analysis is invoked by MEANSTR option. For example, if there are three variables in the analysis, $\boldsymbol{\mu}$ is defined as

$$\boldsymbol{\mu} = \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix}$$

PROC CALIS denotes m_1, m_2, and m_3 by _mean_1, _mean_2, and _mean_3, respectively.

UNIFORM specifies a mean vector with a uniform mean parameter. For example, if there are three variables in the analysis, the mean pattern is:

$$\boldsymbol{\mu} = \begin{pmatrix} m \\ m \\ m \end{pmatrix}$$

PROC CALIS denotes the common mean parameter by _meanparm.

ZERO specifies a zero vector for the mean structures. For example, if there are four variables in the analysis, the mean pattern generated by PROC CALIS is:

$$\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

When you specify the mean structure model by means of the MEANPATTERN= option, you can define the set of variables in the analysis by the VAR statement (either within the scope of the PROC CALIS statement or the GROUP statements). If the VAR statement is not used, PROC CALIS uses all numerical variables in the data sets.

Except for the EQMEANVEC pattern, all other built-in mean patterns are primarily designed for single-group analysis. However, you can still use these mean pattern options for multiple-group situations. For example, consider the following three-group analysis:

```
proc calis meanpattern=uniform;
   group 1 / data=set1;
   group 2 / data=set2;
   group 3 / data=set3;
run;
```

In this specification, all three groups are fitted by the uniform mean pattern. However, there would be no constraints across these groups. PROC CALIS generates a distinct mean parameter for each group: _meanparm_mdl1 for Group 1, _meanparm_mdl2 for Group 2, and _meanparm_mdl3 for Group 3. Similarly, the _mdlxx suffix, where xx represents the model number, is applied to the parameters defined by the SATURATED mean pattern in multiple-group situations.

You can also apply the MEANPATTERN= option partially to the groups in the analysis. For example, the following statements apply the ZERO mean pattern to Group 1 and Group 2 only:

```
proc calis meanpattern=zero;
   group 1 / data=set1;
   group 2 / data=set2;
   group 3 / data=set3;
   model 3 / group=3;
     path    x1 ===> y3;
     means x1 = mean_x1;
run;
```

Group 3 is fitted by Model 3, which is specified explicitly by a PATH model with a distinct mean parameter mean_x1.

If the EQMEANVEC pattern is specified instead, as shown in the following statements, the equality of mean vectors still holds for Groups 1 and 2:

```
proc calis meanpattern=eqmeanvec;
   group 1 / data=set1;
   group 2 / data=set2;
```

```
    group 3 / data=set3;
    model 3 / group=3;
      path    x1 ===> y3;
      means x1 = mean_x1;
run;
```

However, Group 3 has it own mean structures as specified in Model 3.

Use the COVPATTERN= option if you also want to analyze some built-in covariance structures along with the mean structures. If you use the MEANPATTERN= option but do not specify the COVPATTERN= option, a saturated covariance structure model (that is, COVPATTERN=SATURATED) is assumed by default.

MEANSTR

invokes the analysis of mean structures. By default, no mean structures are analyzed. You can specify the MEANSTR option in both the PROC CALIS and the MODEL statements. When this option is specified in the PROC CALIS statement, it propagates to all models. When this option is specified in the MODEL statement, it applies only to the local model. Except for the COSAN model, the MEANSTR option adds default mean parameters to the model. For the COSAN model, the MEANSTR option adds null mean vectors to the model. Instead of using the MEANSTR option to analyze the mean structures, you can specify the mean and the intercept parameters explicitly in the model by some model specification statements. That is, you can specify the intercepts in the LINEQS statement, the intercepts and means in the PATH or the MEAN statement, the _MEAN_ matrix in the MATRIX statement, or the mean structure formula in the COSAN statement. The explicit mean structure parameter specifications are useful when you need to constrain the mean parameters or to create your own references of the parameters.

METHOD=*name*

MET=*name*

M=*name*

specifies the method of parameter estimation. The default is METHOD=ML. You can specify the following *names*.

ML | M | MAX

performs normal-theory maximum likelihood parameter estimation. The ML method requires a nonsingular covariance or correlation matrix. For options that provide more control of the standard error computation for METHOD=ML, see the INFORMATION= and STDERR= options.

MLSB | ML(SB) | MLM

performs normal-theory maximum likelihood parameter estimation and the Satorra-Bentler scale corrections on the chi-squares of the baseline and target models. Model fit statistics are then computed based on these scaled chi-squares. This method also adjusts the computation of standard error estimates by using the sandwich formula proposed by Satorra and Bentler (1994). In effect, the standard error estimates for METHOD=MLSB are the same as those obtained from METHOD=ML when you use the STDERR=SBSW option.

Because the chi-square statistics and standard error estimates for the MLSB method have been shown to have some desirable statistical behavior even under the violation of multivariate normality assumption, the

MLSB method is often referred to as a robust maximum likelihood method. However, in the CALIS procedure, METHOD=MLSB is not the same as METHOD=ML with the specification of the ROBUST option. The latter maximum likelihood method achieves the robustness by case-level weighting of the observations during iterative estimation. However, the MLSB method does not use case-level weighting at all. Instead, it can be viewed as a kind of postestimation adjustment of the chi-squares and standard error estimates.

The MLSB method requires raw input data sets. If you do not provide raw data, the estimation results of MLSB are the same as those obtained from METHOD=ML. For options that provide more control of the chi-square correction and standard error computation for METHOD=MLSB, see the INFORMATION=, SBNTW=, and STDERR= options.

FIML

performs full information maximum likelihood (FIML) or direct maximum likelihood parameter estimation for data that have missing values. This method assumes raw input data sets. When there are no missing values in the analysis, the FIML method yields the same estimates as those from using the regular maximum likelihood (METHOD=ML) method with VARDEF=N.

For METHOD=FIML, the observed information matrix, rather than the expected information matrix, is the default for computing standard error estimates. The use of the observed information matrix has been proved to produce better standard error estimates under the missing at random condition. However, you can specify INFORMATION=EXPECTED to overwrite the default.

Because the FIML method recomputes the mean estimates iteratively during estimation, it must intrinsically analyze the mean structures of models. If you do not specify the MEANSTR option or any mean parameters for your models (which is not required for using the FIML method), PROC CALIS assumes saturated mean structures for models. However, when computing fit statistics, these saturated mean structures would be ignored as if they were never modeled. If you do specify the MEANSTR option or any mean parameters for your models, these mean structures would be taken into account when computing fit statistics.

GLS | G

performs generalized least squares parameter estimation. If no INWGT= data set is specified, the GLS method uses the inverse sample covariance or correlation matrix as the weight matrix $\mathbf{W}$. Therefore, METHOD=GLS requires a nonsingular covariance or correlation matrix. For more control of the standard error computation for METHOD=GLS, see the INFORMATION= option.

WLS | W | ADF

performs weighted least squares parameter estimation. If no INWGT= data set is specified, the WLS method uses the inverse matrix of estimated asymptotic covariances of the sample covariance or correlation matrix as the weight matrix $\mathbf{W}$. In this case, the WLS estimation method is equivalent to Browne's asymptotically distribution-free estimation (Browne 1982, 1984). The WLS method requires a nonsingular weight matrix.

DWLS | D performs diagonally weighted least squares parameter estimation. If no INWGT= data set is specified, the DWLS method uses the inverse diagonal matrix of asymptotic variances of the input sample covariance or correlation matrix as the weight matrix **W**. The DWLS method requires a nonsingular diagonal weight matrix.

ULS | LS | U performs unweighted least squares parameter estimation.

LSML | LSM | LSMAX performs unweighted least squares followed by normal-theory maximum-likelihood parameter estimation.

LSMLSB | LSML(SB) | LSMLM performs unweighted least squares followed by normal-theory maximum likelihood parameter estimation. It also computes the Satorra-Bentler scaled chi-squares of the baseline and target models under maximum likelihood estimation. The standard errors of maximum likelihood estimates are based on the sandwich formula proposed by Satorra and Bentler (1994).

LSFIML performs unweighted least squares followed by full information maximum-likelihood parameter estimation.

LSGLS | LSG performs unweighted least squares followed by generalized least squares parameter estimation.

LSWLS | LSW | LSADF performs unweighted least squares followed by weighted least squares parameter estimation.

LSDWLS | LSD performs unweighted least squares followed by diagonally weighted least squares parameter estimation.

NONE | NO uses no estimation method. This option is suitable for checking the validity of the input information and for displaying the model matrices and initial values.

MODIFICATION

MOD

computes and displays Lagrange multiplier (LM) test indices for constant parameter constraints, equality parameter constraints, and active boundary constraints, as well as univariate and multivariate Wald test indices. The modification indices are not computed in the case of unweighted or diagonally weighted least squares estimation.

The Lagrange multiplier test (Bentler 1986; Lee 1985; Buse 1982) provides an estimate of the χ^2 reduction that results from dropping the constraint. For constant parameter constraints and active boundary constraints, the approximate change of the parameter value is displayed also. You can use this value to obtain an initial value if the parameter is allowed to vary in a modified model. See the section "Modification Indices" on page 1848 for more information.

Relying solely on the LM tests to modify your model can lead to unreliable models that capitalize purely on sampling errors. See MacCallum, Roznowski, and Necowitz (1992) for the use of LM tests.

MSINGULAR=*r*

MSING=*r*

specifies a relative singularity criterion r ($r > 0$) for the inversion of the information matrix, which is needed to compute the covariance matrix. If you do not specify the SINGULAR= option, the default value for r or MSING= is 1E–12; otherwise, the default value is 1E–4 $\times$ *SING*, where *SING* is the specified SINGULAR= value.

When inverting the information matrix, the following singularity criterion is used for the diagonal pivot $d_{j,j}$ of the matrix:

$$|d_{j,j}| \leq \max(ASING, VSING * |H_{j,j}|, MSING * \max(|H_{1,1}|, \ldots, |H_{n,n}|))$$

where *ASING* and *VSING* are the specified values of the ASINGULAR= and VSINGULAR= options, respectively, and $H_{j,j}$ is the jth diagonal element of the information matrix. Note that in many cases a normalized matrix $\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}$ is decomposed (where $\mathbf{D}^2 = \mathrm{diag}(\mathbf{H})$), and the singularity criteria are modified correspondingly.

NOADJDF

turns off the automatic adjustment of degrees of freedom when there are active constraints in the analysis. When the adjustment is in effect, most fit statistics and the associated probability levels will be affected. This option should be used when you believe that the active constraints observed in the current sample will have little chance to occur in repeated sampling. See the section "Adjustment of Degrees of Freedom" on page 1823 for more discussion on the issue.

NOBS=*nobs*

specifies the number of observations. If the DATA= input data set is a raw data set, *nobs* is defined by default to be the number of observations in the raw data set. The NOBS= and EDF= options override this default definition. You can use the RDF= option to modify the *nobs* specification. If the DATA= input data set contains a covariance, correlation, or scalar product matrix, you can specify the number of observations either by using the NOBS=, EDF=, and RDF= options in the PROC CALIS statement or by including a _TYPE_='N' observation in the DATA= input data set.

NOINDEXTYPE

disables the display of index types in the fit summary table.

NOMEANSTR

deactivates the inherited MEANSTR option for the analysis of mean structures. You can specify the NOMEANSTR option in both the PROC CALIS and the MODEL statements. When this option is specified in the PROC CALIS statement, it does not have any apparent effect because by default the mean structures are not analyzed. When this option is specified in the MODEL statement, it deactivates the inherited MEANSTR option from the PROC CALIS statement. In other words, this option is mainly used for resetting the default behavior in the local model that is specified within the scope of a particular MODEL statement. If you specify both the MEANSTR and NOMEANSTR options in the same statement, the NOMEANSTR option is ignored.

CAUTION: This option does not remove the mean structure specifications from the model. It only deactivates the MEANSTR option inherited from the PROC CALIS statement. The mean structures of the model are analyzed as long as there are mean structure specifications in the model (for example, when you specify the means or intercepts in any of the main or subsidiary model specification statements).

NOMISSPAT

suppresses the display of the analytic results of the missing patterns. This option is relevant only when there are incomplete observations (with some missing values in the analysis variables) in the input raw data set and when you use METHOD=FIML or METHOD=LSFIML for estimation.

NOMOD

suppresses the computation of modification indices. The NOMOD option is useful in connection with the PALL option because it saves computing time.

NOORDERSPEC

prints the model results in the order they appear in the input specifications. This is the default printing behavior. In contrast, the ORDERSPEC option arranges the model results by the types of parameters. You can specify the NOORDERSPEC option in both the PROC CALIS and the MODEL statements. When this option is specified in the PROC CALIS statement, it does not have any apparent effect because by default the model results display in the same order as that in the input specifications. When this option is specified in the MODEL statement, it deactivates the inherited ORDERSPEC option from the PROC CALIS statement. In other words, this option is mainly used for resetting the default behavior in the local model that is specified within the scope of a particular MODEL statement. If you specify both the ORDERSPEC and NOORDERSPEC options in the same statement, the NOORDERSPEC option is ignored.

NOPARMNAME

suppresses the printing of parameter names in the model results. The default is to print the parameter names. You can specify the NOPARMNAME option in both the PROC CALIS and the MODEL statements. When this option is specified in the PROC CALIS statement, it propagates to all models. When this option is specified in the MODEL statement, it applies only to the local model.

NOPRINT

NOP

suppresses the displayed output. Note that this option temporarily disables the Output Delivery System (ODS). See Chapter 23, "Using the Output Delivery System," for more information.

NOSTAND

suppresses the printing of standardized results. The default is to print the standardized results.

NOSTDERR

NOSE

suppresses the printing of the standard error estimates. Standard errors are not computed for unweighted least squares (ULS) or diagonally weighted least squares (DWLS) estimation. In general, standard errors are computed even if the STDERR display option is not used (for file output). You can specify the NOSTDERR option in both the PROC CALIS and the MODEL statements. When this option is specified in the PROC CALIS statement, it propagates to all models. When this option is specified in the MODEL statement, it applies only to the local model.

OMETHOD=*name*

OM=*name*

TECHNIQUE=*name*

TECH=*name*

specifies the optimization method or technique. Because there is no single nonlinear optimization

algorithm available that is clearly superior (in terms of stability, speed, and memory) for all applications, different types of optimization methods or techniques are provided in the CALIS procedure. The optimization method or technique is specified by using one of the following *names* in the OMETHOD= option:

CONGRA | CG chooses one of four different conjugate-gradient optimization algorithms, which can be more precisely defined with the UPDATE= option and modified with the LINESEARCH= option. The conjugate-gradient techniques need only $O(t)$ memory compared to the $O(t^2)$ memory for the other three techniques, where t is the number of parameters. On the other hand, the conjugate-gradient techniques are significantly slower than other optimization techniques and should be used only when memory is insufficient for more efficient techniques. When you choose this option, UPDATE=PB by default. This is the default optimization technique if there are more than 999 parameters to estimate.

DBLDOG | DD performs a version of double dogleg optimization, which uses the gradient to update an approximation of the Cholesky factor of the Hessian. This technique is, in many aspects, very similar to the dual quasi-Newton method, but it does not use line search. The implementation is based on Dennis and Mei (1979) and (Gay 1983).

LEVMAR | LM | MARQUARDT performs a highly stable (but for large problems, memory- and time-consuming) Levenberg-Marquardt optimization technique, a slightly improved variant of the (Moré 1978) implementation. This is the default optimization technique for estimation methods other than the FIML if there are fewer than 500 parameters to estimate.

NEWRAP | NRA performs a usually stable (but for large problems, memory- and time-consuming) Newton-Raphson optimization technique. The algorithm combines a line-search algorithm with ridging, and it can be modified with the LINESEARCH= option.

NRRIDG | NRR | NR | NEWTON performs a usually stable (but for large problems, memory- and time-consuming) Newton-Raphson optimization technique. This algorithm does not perform a line search. Since OMETHOD=NRRIDG uses an orthogonal decomposition of the approximate Hessian, each iteration of OMETHOD=NRRIDG can be slower than that of OMETHOD=NEWRAP, which works with Cholesky decomposition. However, usually OMETHOD=NRRIDG needs fewer iterations than OMETHOD=NEWRAP. The NRRIDG technique is the default optimization for the FIML estimation if there are fewer than 500 parameters to estimate.

QUANEW | QN chooses one of four different quasi-Newton optimization algorithms that can be more precisely defined with the UPDATE= option and modified with the LINESEARCH= option. If boundary constraints are used, these techniques sometimes converge slowly. When you choose this option, UPDATE=DBFGS by default. If nonlinear constraints are specified in the NLINCON statement, a modification of Powell's VMCWD algorithm (Powell 1982a, b) is used, which is a sequential quadratic programming (SQP) method. This algorithm can be modified by specifying VERSION=1, which replaces the update of the Lagrange multiplier estimate vector $\boldsymbol{\mu}$ to the original update of Powell (1978b, a) that is used in the VF02AD algorithm. This can be helpful for applications with

linearly dependent active constraints. The QUANEW technique is the default optimization technique if there are nonlinear constraints specified or if there are more than 499 and fewer than 1,000 parameters to estimate. The QUANEW algorithm uses only first-order derivatives of the objective function and, if available, of the nonlinear constraint functions.

TRUREG | TR performs a usually very stable (but for large problems, memory- and time-consuming) trust-region optimization technique. The algorithm is implemented similar to Gay (1983) and Moré and Sorensen (1983).

NONE | NO does not perform any optimization. This option is similar to METHOD=NONE, but OMETHOD=NONE also computes and displays residuals and goodness-of-fit statistics. If you specify METHOD=ML, METHOD=LSML, METHOD=GLS, METHOD=LSGLS, METHOD=WLS, or METHOD=LSWLS, this option enables computing and displaying (if the display options are specified) of the standard error estimates and modification indices corresponding to the input parameter estimates.

For fewer than 500 parameters ($t < 500$), OMETHOD=NRRIDG (Newton-Raphson Ridge) is the default optimization technique for the FIML estimation, and OMETHOD=LEVMAR (Levenberg-Marquardt) is the default optimization technique for the all other estimation methods. For $500 \leq t < 1{,}000$, OMETHOD=QUANEW (quasi-Newton) is the default method, and for $t \geq 1{,}000$, OMETHOD=CONGRA (conjugate gradient) is the default method. Each optimization method or technique can be modified in various ways. See the section "Use of Optimization Techniques" on page 1854 for more details.

ORDERALL

prints the model and group results in the order of the model or group numbers, starting from the smallest number. It also arrange some model results by the parameter types. In effect, this option turns on the ORDERGROUPS, ORDERMODELS, and ORDERSPEC options. The ORDERALL is not a default option. By default, the printing of the results follow the order of the input specifications.

ORDERGROUPS

ORDERG

prints the group results in the order of the group numbers, starting from the smallest number. The default behavior, however, is to print the group results in the order they appear in the input specifications.

ORDERMODELS

ORDERMO

prints the model results in the order of the model numbers, starting from the smallest number. The default behavior, however, is to print the model results in the order they appear in the input specifications.

ORDERSPEC

arranges some model results by the types of parameters. The default behavior, however, is to print the results in the order they appear in the input specifications. You can specify the ORDERSPEC option in both the PROC CALIS and the MODEL statements. When this option is specified in the PROC CALIS statement, it propagates to all models. When this option is specified in the MODEL statement, it applies only to the local model.

OUTEST=*SAS-data-set*

creates an output data set that contains the parameter estimates, their gradient, Hessian matrix, and boundary and linear constraints. For METHOD=ML, METHOD=GLS, and METHOD=WLS, the OUTEST= data set also contains the information matrix, the approximate covariance matrix of the parameter estimates ((generalized) inverse of information matrix), and approximate standard errors. If linear or nonlinear equality or active inequality constraints are present, the Lagrange multiplier estimates of the active constraints, the projected Hessian, and the Hessian of the Lagrange function are written to the data set.

See the section "OUTEST= Data Set" on page 1715 for a description of the OUTEST= data set. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTFIT=*SAS-data-set*

creates an output data set that contains the values of the fit indices. See the section "OUTFIT= Data Set" on page 1729 for details.

OUTMODEL=*SAS-data-set*

OUTRAM=*SAS-data-set*

creates an output data set that contains the model information for the analysis, the parameter estimates, and their standard errors. An OUTMODEL= data set can be used as an input INMODEL= data set in a subsequent analysis by PROC CALIS. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTSTAT=*SAS-data-set*

creates an output data set that contains the BY group variables, the analyzed covariance or correlation matrices, and the predicted and residual covariance or correlation matrices of the analysis. You can specify the correlation or covariance matrix in an OUTSTAT= data set as an input DATA= data set in a subsequent analysis by PROC CALIS. See the section "OUTSTAT= Data Set" on page 1725 for a description of the OUTSTAT= data set. If the model contains latent variables, this data set also contains the predicted covariances between latent and manifest variables and the latent variable score regression coefficients (see the PLATCOV option on page 1571). If the FACTOR statement is used, the OUTSTAT= data set also contains the rotated and unrotated factor loadings, the unique variances, the matrix of factor correlations, the transformation matrix of the rotation, and the matrix of standardized factor loadings.

You can use the latent variable score regression coefficients with PROC SCORE to compute factor scores.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTWGT=*SAS-data-set*

OUTWEIGHT=*SAS-data-set*

creates an output data set that contains the elements of the weight matrix $\mathbf{W}$ or the its inverse $\mathbf{W}^{-1}$ used in the estimation process. The inverse of the weight matrix is output only when you specify an INWGT= data set with the INWGT= and INWGTINV options (or the INWGT(INV)= option alone) in the same analysis. As a result, the entries in the INWGT= and OUTWGT= data sets are

consistent. In other situations where the weight matrix is computed by the procedure or obtained from the OUTWGT= data set without the INWGTINV option, the weight matrix is output in the OUTWGT= data set. Furthermore, if the weight matrix is computed by the procedure, the OUTWGT= data set contains the elements of the weight matrix on which the WRIDGE= and the WPENALTY= options are applied.

You cannot create an OUTWGT= data set with an unweighted least squares or maximum likelihood estimation. The weight matrix is defined only in the GLS, WLS (ADF), or DWLS fit function. An OUTWGT= data set can be used as an input INWGT= data set in a subsequent analysis by PROC CALIS. See the section "OUTWGT= Data Set" on page 1729 for the description of the OUTWGT= data set. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PALL

ALL

displays all optional output except the output generated by the PCOVES and PDETERM options.

CAUTION: The PALL option includes the very expensive computation of the modification indices. If you do not really need modification indices, you can save computing time by specifying the NOMOD option in addition to the PALL option.

PARMNAME

prints the parameter names in the model results. This is the default printing behavior. In contrast, the NOPARMNAME option suppresses the printing of the parameter names in the model results. You can specify the PARMNAME option in both the PROC CALIS and the MODEL statements. When this option is specified in the PROC CALIS statement, it does not have any apparent effect because by default model results show the parameter names. When this option is specified in the MODEL statement, it deactivates the inherited NOPARMNAME option from the PROC CALIS statement. In other words, this option is mainly used for resetting the default behavior in the local model that is specified within the scope of a particular MODEL statement. If you specify both the PARMNAME and NOPARMNAME options in the same statement, the PARMNAME option is ignored.

PCORR

CORR

displays the covariance or correlation matrix that is analyzed and the predicted model covariance or correlation matrix.

PCOVES

PCE

displays the following:

- the information matrix
- the approximate covariance matrix of the parameter estimates (generalized inverse of the information matrix)
- the approximate correlation matrix of the parameter estimates

The covariance matrix of the parameter estimates is not computed for estimation methods ULS and DWLS. This displayed output is not included in the output generated by the PALL option.

PDETERM

PDE

displays three coefficients of determination: the determination of all equations (DETAE), the determination of the structural equations (DETSE), and the determination of the manifest variable equations (DETMV). These determination coefficients are intended to be global means of the squared multiple correlations for different subsets of model equations and variables. The coefficients are displayed only when you specify a FACTOR, LINEQS, LISMOD, PATH, or RAM model, but they are displayed for all five estimation methods: ULS, GLS, ML, WLS, and DWLS.

You can use the STRUCTEQ statement to define which equations are structural equations. If you do not use the STRUCTEQ statement, PROC CALIS uses its own default definition to identify structural equations.

The term "structural equation" is not defined in a unique way. The LISREL program defines the structural equations by the user-defined BETA matrix. In PROC CALIS, the default definition of a structural equation is an equation that has a dependent left-side variable that appears at least once on the right side of another equation, or an equation that has at least one right-side variable that appears at the left side of another equation. Therefore, PROC CALIS sometimes identifies more equations as structural equations than the LISREL program does.

PESTIM

PES

displays the parameter estimates. In some cases, this includes displaying the standard errors and *t* values.

PINITIAL

PIN

displays the model specification with initial estimates and the vector of initial values.

PLATCOV

PLATMOM

PLC

displays the following:

- the estimates of the covariances among the latent variables
- the estimates of the covariances between latent and manifest variables
- the estimates of the latent variable means for mean structure analysis
- the latent variable score regression coefficients

The estimated covariances between latent and manifest variables and the latent variable score regression coefficients are written to the OUTSTAT= data set. You can use the score coefficients with PROC SCORE to compute factor scores.

PLOTS< **=***plot-request* >

PLOT< **=(***plot-request* < **...** *plot-request* >**)** >

specifies the ODS Graphics plots. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. For example:

PLOTS=ALL
PLOTS=RESIDUALS
PLOTS=(PP RESBYPRED QQ)

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc calis plots;
   path y <=== x,
        y <=== z;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *plot-requests*:

ALL displays all plots.

CASERESID | CASERESIDUAL | CASERESIDUALS displays all the case-level ODS Graphics plots enabled by the following *plot-requests*: CRESHIST, PP, QQ, RESBYLEV, and RESBYPRED. This option requires raw data input.

CRESHIST | CRESIDUALHISTOGRAM produces the ODS Graphics plot CaseResidualHistogram, which displays the distribution of the case-level (observation-level) residuals in the form of a histogram, where residuals are measured in terms of M-distances. This option requires raw data input.

NONE suppresses ODS Graphics plots.

PATHDIAGRAM produces the ODS Graphics plot PathDiagram, which display the path diagram for the unstandardized solution. For options that control and customize path diagrams, see the PATHDIAGRAM statement.

PP | PPPLOT produces the ODS Graphics plot ResPercentileByExpPercentile, which plots the observed percentiles of the residual M-distances against the theoretical percentiles. This plot is useful for showing departures from the theoretical distribution in terms of percentiles, and it is especially sensitive to departures in the middle region of the distribution. This option requires raw data input.

QQ | QQPLOT produces the ODS Graphics plot ResidualByQuantile, which plots the residual M-distances (observed quantiles) against the theoretical quantiles. This plot is useful for showing departures from the theoretical distribution in terms of quantiles, and it is especially sensitive to departures at the upper tail of the distribution. This option requires raw data input.

RESBYLEV | RESIDUALBYLEVERAGE produces the ODS Graphics plot ResidualByLeverage, which plots the residual M-distances against the leverage M-distances. This plot is useful for showing outliers and leverage observations graphically. See the ALPHAOUT= and ALPHALEV= options for detection criteria of outliers and leverage observations. This option requires raw data input.

RESBYPRED | RESONFIT | RESIDUALBYPREDICTED | RESIDUALONFIT<(VAR= *var-list***)>**
produces the ODS Graphics plot ResidualByPredicted, which plots the residuals against the predicted values of the dependent observed variables in the model. You can restrict the set of dependent variables to display by specifying *var-list* in the VAR= option. If *var-list* is not specified in the VAR= option, plots for all dependent observed variables are displayed. The residual on fit plots are useful for detecting nonlinear relationships in the model. If the relationships are linear and the residual variance is homoscedastic, the residuals should not show systematic pattern with the predicted values. This option requires raw data input.

RESIDUAL | RESIDUALS produces the ODS Graphics plot for the histogram of residuals in covariances and means rather than the case-level residuals. With this ODS Graphics plot, the nongraphical (legacy) output for the bar chart of residual tallies is redundant and therefore is suppressed. To display this bar chart together with the ODS Graphics for residual histogram, you must use the RESIDUAL(TALLY) option in the PROC CALIS statement. This option does not require raw data input.

PRIMAT
PMAT
displays parameter estimates, approximate standard errors, and t values in matrix form if you specify the analysis model using the RAM or LINEQS statement.

PRINT
PRI
adds the options KURTOSIS, RESIDUAL, PLATCOV, and EFFPART to the default output.

PSHORT
SHORT
PSH
excludes the output produced by the PINITIAL, SIMPLE, and STDERR options from the default output.

PSUMMARY
SUMMARY
PSUM
displays the fit assessment table only.

PWEIGHT
PW
displays the weight matrix **W** used in the estimation. The weight matrix is displayed after the WRIDGE= and the WPENALTY= options are applied to it. However, if you specify an INWGT= data set by the INWGT= and INWGTINV options (or the INWGT(INV)= option alone) in the same analysis, this option displays the elements of the inverse of the weight matrix.

RADIUS=*r*

is an alias for the INSTEP= option for Levenberg-Marquardt minimization.

RANDOM=*i*

specifies a positive integer as a seed value for the pseudo-random number generator to generate initial values for the parameter estimates for which no other initial value assignments in the model definitions are made. Except for the parameters in the diagonal locations of the central matrices in the model, the initial values are set to random numbers in the range $0 \leq r \leq 1$. The values for parameters in the diagonals of the central matrices are random numbers multiplied by 10 or 100. See the section "Initial Estimates" on page 1853 for more information.

RDF=*n*

DFR=*n*

makes the effective number of observations the actual number of observations minus the RDF= value. The degree of freedom for the intercept should not be included in the RDF= option. If you use PROC CALIS to compute a regression model, you can specify RDF= *number-of-regressor-variables* to get approximate standard errors equal to those computed by PROC REG.

READADDPARM

READADD

inputs the generated default parameters (for example, observations with _TYPE_=ADDPCOV, ADDMEAN, or ADDPVAR) in the INMODEL= data set as if they were part of the original model specification. Typically, these default parameters in the INMODEL= data set were generated automatically by PROC CALIS in a previous analysis and stored in an OUTMODEL= data set, which is then used as the INMODEL= data set in a new run of PROC CALIS. By default, PROC CALIS does not input the observations for default parameters in the INMODEL= data set. In most applications, you do not need to specify this option because PROC CALIS is able to generate a new set of default parameters that are appropriate to the new situation after it reads in the INMODEL= data set. Undistinguished uses of the READADDPARM option might lead to unintended constraints on the default parameters.

RESIDUAL < (TALLY | TALLIES) > < = NORM | VARSTAND | ASYSTAND >

RES < (TALLY | TALLIES) > < = NORM | VARSTAND | ASYSTAND >

displays the raw and normalized residual covariance matrix, the rank order of the largest residuals, and a bar chart of the residual tallies. If mean structures are modeled, mean residuals are also displayed and ranked.

For raw data input, this option also displays tables for case-level (observation-level) residual analysis, including outlier and leverage detections and departures of residuals from the theoretical residual distributions. To set the criterion for detecting outliers, use the ALPHAOUT= option. To set the criterion for leverage observations, use the ALPHALEV= option. Case-level residual analysis is not available when you specify METHOD=FIML.

For the covariance and mean residuals, three types of normalized or standardized residual matrices can be chosen with the RESIDUAL= specification.

NORM	normalized residuals
VARSTAND	variance standardized residuals
ASYSTAND	asymptotically standardized residuals

When ODS Graphics plots of covariance and mean residuals are also requested, the bar charts of residual tallies are suppressed. They are replaced with high quality graphical histograms showing residual distributions. If you still want to display the bar charts in this situation, use the RESIDUAL(TALLY) or RESIDUAL(TALLY)= option.

The RESIDUAL option is also enabled by the PRINT option. See the section "Assessment of Fit" on page 1824 for more details about the definitions of residuals.

RIDGE< =*r* >

defines a ridge factor *r* for the diagonal of the covariance or correlation matrix **S** that is analyzed. The matrix **S** is transformed to:

$$\mathbf{S} \longrightarrow \tilde{\mathbf{S}} = \mathbf{S} + r(\mathrm{diag}(\mathbf{S}))$$

If you do not specify *r* in the RIDGE option, PROC CALIS tries to ridge the covariance or correlation matrix **S** so that the smallest eigenvalue is about 10^{-3}. Because the weight matrix in the GLS method is the same as the observed covariance or correlation matrix, the RIDGE= option also applies to the weight matrix for the GLS estimation, unless you input the weight matrix by the INWGT= option.

CAUTION: The covariance or correlation matrix in the OUTSTAT= output data set does not contain the ridged diagonal.

ROBITER=*i*

ROBUSTITER=*i*

specifies the maximum number *i* of iterations for the iteratively reweighted least squares (IRLS) method to compute the robust mean and covariance matrices with the two-stage robust estimation. This option is relevant only with the use of the ROBUST= option and with raw data input. The default value is 5,000.

You can also specify this option in the GROUP statement so that different groups can use different ROBITER= values. Notice that the ROBITER= option does not specify the maximum number of iterations for the IRLS algorithm used in the direct robust estimation or in the second stage of the two-stage robust estimation. You can specify the MAXITER= option for this purpose.

ROBPHI=*r*

ROBUSTPHI=*r*

sets the tuning parameter *r* ($0 < r < 1$) for the robust estimation method that you specify using the ROBUST= option. The ROBPHI= value controls the criterion for downweighting observations. This value indicates approximately the proportion of observations that would receive weights less than 1 (that is, would be downweighted) according to certain theoretical distributions. The larger the ROBPHI= value, the more observations are downweighted (that is, with weights less than 1). The default value is 0.05.

You can also specify this option in the GROUP statement so that different groups can use different ROBPHI= values for the tuning parameters.

ROBUST <=*name*>

ROB <=*name*>

invokes the robust estimation method that downweights the outliers in estimation. You can use the ROBUST option only in conjunction with the ML method (METHOD=ML). More accurately, the robust estimation is done by using the iteratively reweighted least squares (IRLS) method under the normal distribution assumption. The model fit of robust estimation is evaluated with the ML discrepancy function.

You must provide raw data input for the robust estimation method to work. With the robust method, the Huber weights are applied to the observations so that outliers are downweighted during estimation. See the section "Robust Estimation" on page 1812 for details.

You can request the three different types of robust methods by using one of the following *names*:

RESIDUAL | DIRECT | RESID | RES <**(E)**> specifies a direct robust method that downweights observations with large residuals during the iterative estimation of the model. This method treats the disturbances (the error terms of endogenous latent factors) as errors or residuals (hence the keyword **E**) in the associated factor model when computing residual M-distances and factor scores during the robust estimation. The **(E)**specification is irrelevant if there are no endogenous latent factors in the model. This is the default robust method.

RESIDUAL | DIRECT | RESID | RES (F) specifies a direct robust method that downweights observations with large estimated residuals during the iterative estimation of the model. Unlike the **(E)**method, this method treats the disturbances (the error terms of endogenous latent factors) as factors (hence the keyword **F**) in the associated factor model when computing residual M-distances and factor scores during the robust estimation. The **(F)**specification is irrelevant if there are no endogenous latent factors in the model.

SAT | TWOSTAGE | UNSTRUCT | UNS specifies a two-stage robust method that downweights the observations with large M-distances in all observed variable dimensions when computing the covariance matrix and mean vector from the input raw data. As a results, this option produces a robust covariance matrix and a mean vector for a subsequent model estimation where no reweighting would be applied at the observational level. Hence, this is a two-stage method that applies weights only in the first stage for computing the robust covariance and mean matrices. This is in contrast with the **RES(E)** or **RES(F)** option, where weighting and reweighting of observations are applied directly during model estimation.

For details about these robust methods, see the section "Robust Estimation" on page 1812.

To control the proportion of the observations that are downweighted during the robust estimation, you can specify the value of the tuning parameter φ, which is between 0 and 1, by using the ROBPHI= option. Approximately, $\varphi \times 100\%$ of observations would receive weights less than 1 according to certain theoretical distributions. By default, the value of the tuning parameter φ is set to 0.05 for all robust methods in PROC CALIS.

By default, the robust method uses a maximum of 5,000 iterations to obtain parameter convergence through the IRLS algorithm. You can override this default maximum number of iterations by specifying the ROBITER= option. The default relative parameter convergence criterion for the robust method is

1E–8. See the XCONV= option for the mathematical definition of this criterion and for information about overriding the default convergence criterion.

Because all robust methods reweight the observations iteratively, the observed variable means are always implicitly updated with the robust weights. Therefore, in a sense all robust methods intrinsically analyze the mean structures of models. If you do not specify the MEANSTR option or any mean parameters for your models, PROC CALIS assumes appropriate saturated mean structures for the models. However, when you are computing fit statistics, these saturated mean structures are ignored as if they were never modeled. If you do specify the MEANSTR option or any mean parameters for your models, these mean structures are taken into account in computing fit statistics.

Robust estimation with the IRLS method is not supported when you specify the BOUNDS, LINCON, or NLINCON statement. However, you can still set parameter constraints by using the same parameter names or by specifying the PARAMETERS statement and the SAS programming statements. See the section "Setting Constraints on Parameters" on page 1778 for techniques to set up implicit parameter constraints by using the PARAMETERS statement and SAS programming statements.

SALPHA=*r*

is an alias for the INSTEP= option for line-search algorithms.

SBNTW=*name*

SBNTWGT=*name*

specifies the covariance matrix on which the normal-theory weight matrix is based when you use the Satorra-Bentler sandwich formula to compute standard errors. You can specify the following *names*:

OBS | OBSERVED specifies that the observed covariance matrix be used.

PRED | PREDICTED specifies that the model-predicted covariance matrix be used.

By default, SBNTW=PRED. The SBNTW= option is applicable only when you specify the STDERR=SBSW option or when you use METHOD=MLSB. For more information, see the section "Satorra-Bentler Sandwich Formula for Standard Errors" on page 1819.

SIMPLE

S

displays means, standard deviations, skewness, and univariate kurtosis if available. This information is displayed when you specify the PRINT option. If the KURTOSIS option is specified, the SIMPLE option is set by default.

SINGULAR=*r*

SING=*r*

specifies the singularity criterion r ($0 < r < 1$) used, for example, for matrix inversion. The default value is the square root of the relative machine precision or, equivalently, the square root of the largest double precision value that, when added to 1, results in 1.

SLMW=*r*

specifies the probability limit used for computing the stepwise multivariate Wald test. The process stops when the univariate probability is smaller than r. The default value is r=0.05.

SPRECISION=*r*

SP=*r*

is an alias for the LSPRECISION= option.

START=*r*

specifies initial estimates for parameters as multiples of the *r* value. In all CALIS models, you can supply initial estimates individually as parenthesized values after each parameter name. Unspecified initial estimates are usually computed by various reasonable initial estimation methods in PROC CALIS. If none of the initialization methods is able to compute all the unspecified initial estimates, then the remaining unspecified initial estimates are set to r, $10\,|r\,|$, or $100\,|r\,|$. For variance parameters, $100\,|r\,|$ is used for covariance structure analyses and $10\,|r\,|$ is used for correlation structure analyses. For other types of parameters, r is used. The default value is $r = 0.5$. If the DEMPHAS= option is used, the initial values of the variance parameters are multiplied by the value specified in the DEMPHAS= option. See the section "Initial Estimates" on page 1853 for more information.

STDERR < =SBSW | UNADJ >

SE < =SBSW | UNADJ >

displays standard error estimates if you use estimation methods other than unweighted least squares (ULS) or diagonally weighted least squares (DWLS) and you do not specify the NOSTDERR option. In contrast, the NOSTDERR option suppresses the printing of the standard error estimates. If you specify neither the STDERR nor NOSTDERR option, the standard errors are computed for the OUTMODEL= data set. This information is displayed by default when you specify the PRINT option.

Optionally, you can specify the method for computing standard errors. For METHOD=ML or GLS, the default is STDERR=UNADJ, where the standard errors are computed by inverting an (unadjusted) approximate Hessian or information matrix. For METHOD=MLSB, the default is STDERR=SBSW, where the sandwich formula proposed by Satorra and Bentler (1994) is used to adjust the computation of standard error estimates. Thus, the STDERR= option enables you to overwrite the default standard error method for ML, MLSB, or GLS estimation. For METHOD=FIML or WLS, STDERR=UNADJ is assumed and cannot be overwritten.

You can specify the STDERR option in both the PROC CALIS and MODEL statements. When you use this option in the MODEL statement, you can enable the standard error computation, but you cannot specify the standard error method. The latter is possible only in the PROC CALIS statement. If you specify both the STDERR and NOSTDERR options in the same statement, the STDERR option is ignored.

TMISSPAT=*n*

THRESHOLDMISSPAT=*n*

THRESMISSPAT=*n*

specifies the proportion threshold for the missing patterns to display in the output, where n is between 0 and 1. The default TMISSPAT= value is 0.05. This option is relevant only when there are incomplete observations (with some missing values in the analysis variables) in the input raw data set and when you use METHOD=FIML or METHOD=LSFIML for estimation.

Because the number of missing patterns could be quite large, PROC CALIS displays a limited number of the most frequent missing patterns in the output. Together with the MAXMISSPAT= option, this option controls the number of missing patterns to display in the output. See the MAXMISSPAT= option for a detailed description about how the number of missing patterns to display is determined.

UPDATE=*name*
UPD=*name*

specifies the update method for the quasi-Newton or conjugate-gradient optimization technique.

For OMETHOD=CONGRA, the following updates can be used:

PB performs the automatic restart update method of Powell (1977) and Beale (1972). This is the default.

FR performs the Fletcher-Reeves update (Fletcher 1980, p. 63).

PR performs the Polak-Ribiere update (Fletcher 1980, p. 66).

CD performs a conjugate-descent update of Fletcher (1987).

For OMETHOD=DBLDOG, the following updates (Fletcher 1987) can be used:

DBFGS performs the dual Broyden, Fletcher, Goldfarb, and Shanno (BFGS) update of the Cholesky factor of the Hessian matrix. This is the default.

DDFP performs the dual Davidon, Fletcher, and Powell (DFP) update of the Cholesky factor of the Hessian matrix.

For OMETHOD=QUANEW, the following updates (Fletcher 1987) can be used:

BFGS performs original BFGS update of the inverse Hessian matrix.

DFP performs the original DFP update of the inverse Hessian matrix.

DBFGS performs the dual BFGS update of the Cholesky factor of the Hessian matrix. This is the default.

DDFP performs the dual DFP update of the Cholesky factor of the Hessian matrix.

VARDEF= DF | N | WDF | WEIGHT | WGT

specifies the divisor used in the calculation of covariances and standard deviations. The default value is VARDEF=N for the METHOD=FIML, and VARDEF=DF for other estimation methods. The values and associated divisors are displayed in the following table, where k is the number of partial variables specified in the PARTIAL statement. When a WEIGHT statement is used, w_j is the value of the WEIGHT variable in the jth observation, and the summation is performed only over observations with positive weight.

Value	Description	Divisor
DF	Degrees of freedom	$N - k - 1$
N	Number of observations	N
WDF	Sum of weights DF	$\sum_j^N w_j - k - 1$
WEIGHT | WGT	Sum of weights	$\sum_j^N w_j$

VSINGULAR=*r*

VSING=*r*

specifies a relative singularity criterion *r* ($r > 0$) for the inversion of the information matrix, which is needed to compute the covariance matrix. If you do not specify the SINGULAR= option, the default value for *r* or VSING= is 1E–8; otherwise, the default value is *SING*, which is the specified SINGULAR= value.

When inverting the information matrix, the following singularity criterion is used for the diagonal pivot $d_{j,j}$ of the matrix:

$$|d_{j,j}| \leq \max(ASING, VSING * |H_{j,j}|, MSING * \max(|H_{1,1}|, \ldots, |H_{n,n}|))$$

where *ASING* and *MSING* are the specified values of the ASINGULAR= and MSINGULAR= options, respectively, and $H_{j,j}$ is the *j*th diagonal element of the information matrix. Note that in many cases a normalized matrix $\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}$ is decomposed (where $\mathbf{D}^2 = \mathrm{diag}(\mathbf{H})$), and the singularity criteria are modified correspondingly.

WPENALTY=*r*

WPEN=*r*

specifies the penalty weight $r \geq 0$ for the WLS and DWLS fit of the diagonal elements of a correlation matrix (constant 1s). The criterion for weighted least squares estimation of a correlation structure is

$$\mathbf{F}_{WLS} = \sum_{i=2}^{n}\sum_{j=1}^{i-1}\sum_{k=2}^{n}\sum_{l=1}^{k-1} w^{ij,kl}(s_{ij} - c_{ij})(s_{kl} - c_{kl}) + r\sum_{i}^{n}(s_{ii} - c_{ii})^2$$

where *r* is the penalty weight specified by the WPENALTY=*r* option and the $w^{ij,kl}$ are the elements of the inverse of the reduced $(n(n-1)/2) \times (n(n-1)/2)$ weight matrix that contains only the nonzero rows and columns of the full weight matrix $\mathbf{W}$. The second term is a penalty term to fit the diagonal elements of the correlation matrix. The default value is 100. The reciprocal of this value replaces the asymptotic variance corresponding to the diagonal elements of a correlation matrix in the weight matrix $\mathbf{W}$, and it is effective only with the ASYCOV=CORR option, which is the default for correlation analyses. The often used value $r = 1$ seems to be too small in many cases to fit the diagonal elements of a correlation matrix properly. The default WPENALTY= value emphasizes the importance of the fit of the diagonal elements in the correlation matrix. You can decrease or increase the value of *r* if you want to decrease or increase the importance of the diagonal elements fit. This option is effective only with the WLS or DWLS estimation method and the analysis of a correlation matrix.

See the section "Estimation Criteria" on page 1805 for more details.

CAUTION: If you input the weight matrix by the INWGT= option, the WPENALTY= option is ignored.

WRIDGE=*r*

defines a ridge factor *r* for the diagonal of the weight matrix $\mathbf{W}$ used in GLS, WLS, or DWLS estimation. The weight matrix $\mathbf{W}$ is transformed to

$$\mathbf{W} \longrightarrow \tilde{\mathbf{W}} = \mathbf{W} + r(\mathrm{diag}(\mathbf{W}))$$

The WRIDGE= option is applied on the weight matrix before the following actions occur:

- the WPENALTY= option is applied on it

- the weight matrix is written to the OUTWGT= data set
- the weight matrix is displayed

CAUTION: If you input the weight matrix by the INWGT= option, the OUTWGT= data set will contain the same weight matrix without the ridging requested by the WRIDGE= option. This ensures that the entries in the INWGT= and OUTWGT= data sets are consistent. The WRIDGE= option is ignored if you input the inverse of the weight matrix by the INWGT= and INWGTINV options (or the INWGT(INV)= option alone).

XCONV=*r*

XTOL=*r*

specifies the relative parameter convergence criterion. Termination requires a small relative parameter (x) change in subsequent iterations, that is,

$$\frac{\max_j |x_j^{(k)} - x_j^{(k-1)}|}{\max(|x_j^{(k)}|, |x_j^{(k-1)}|, \mathit{XSIZE})} \leq r$$

The default value for r is 1E–8 for robust estimation (see the ROBUST option) with the iteratively reweighted least squares method, and it is 0 for other estimation methods. The default value for *XSIZE* is 0. You can change this default value by specifying the XSIZE= option in the NLOPTIONS statement.

AUXILIARY Statement

AUXILIARY *variables* **;**

You can use the AUXILIARY statement to define a set of auxiliary variables to be included in the model. The primary purpose of including auxiliary variables is to help obtain unbiased estimation when you use the full information maximum likelihood (FIML) method to analyze incomplete data. Another purpose is to make the missing at random (MAR) assumption more plausible for the use of FIML. In fact, the auxiliary variables themselves can also have missing values when you use the FIML method.

For example, the following statements specify the addition of auxiliary variables w1–w3 in the model that is specified by the PATH statement:

```
proc calis;
   path
      y1 <=== x1 x2,
      y2 <=== x3 x4,
      y1 <=== y2;
   auxiliary w1-w3;
run
```

Auxiliary variables are observed variables that are not of primary interest in your target model. They are included in the model for the sole purpose of improving estimation. PROC CALIS implements the "saturated correlates" method of Graham (2003). The auxiliary variables are treated as exogenous observed variables in the model. They generate the following additional parameters in the model:

- the means of the auxiliary variables

- the variances and covariances among the auxiliary variables
- the covariances between the auxiliary variables and other exogenous observed variables in the model
- the covariances between the auxiliary variables and the error variables for the endogenous observed variables in the model

The addition of these parameters ensures that the degrees of freedom of the original model does not change with the inclusion of the auxiliary variables.

Currently, PROC CALIS supports the use of the AUXILIARY statement only when you analyze a single sample and with METHOD=FIML. In addition, only the following types of models support the use of auxiliary variables:

- LINEQS
- MSTRUCT
- PATH
- RAM

BOUNDS Statement

BOUNDS *constraint* < **,** *constraint . . .* > **;**

where *constraint* represents

< *number operator* > *parameter-list* < *operator number* >

You can use the BOUNDS statement to define boundary constraints for any independent parameter that has its name specified in the main or subsidiary model specification statements, the PARAMETERS statement, or the INMODEL= data set. You cannot define boundary constraints for dependent parameters created in SAS programming statements or elsewhere.

Valid operators are <=, <, >=, >, and = (or, equivalently, LE, LT, GE, GT, and EQ). The following is an example of the BOUNDS statement:

```
bounds        0.   <= a1-a9 x    <= 1. ,
             -1.   <= c2-c5            ,
                      b1-b10 y   >= 0. ;
```

You must separate boundary constraints with a comma, and you can specify more than one BOUNDS statement. The feasible region for a parameter is the intersection of all boundary constraints specified for that parameter; if a parameter has a maximum lower boundary constraint greater than its minimum upper bound, the parameter is set equal to the minimum of the upper bounds.

The active set strategies made available in PROC CALIS treat strict inequality constraints < or > as if they were just inequality constraints <= or >=. For example, if you require x be strictly greater than zero so as to prevent an undefined value for $y = \log(x)$, specifying the following statement is insufficient:

```
BOUNDS x > 0;
```

Specify the following statement instead:

```
BOUNDS x > 1E-8;
```

If the CALIS procedure encounters negative variance estimates during the minimization process, serious convergence problems can occur. You can use the BOUNDS statement to constrain these parameters to nonnegative values. Although allowing negative values for variances might lead to a better model fit with smaller χ^2 value, it adds difficulties in interpretation.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC CALIS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the CALIS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

The BY statement is not supported if you define more than one group by using the GROUP statements.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

COSAN Statement

COSAN < **VAR=***variable-list,* > *term* < + *term* . . . > **;**

where *variable-list* is a list of observed variables and *term* represents either one of the following forms:

matrix-definition < * *matrix-definition* . . . > < *mean-definition* >

or

mean-definition

where *matrix-definition* is of the following form:

matrix-name < (*number-of-columns* < , *matrix-type* < ,*transformation* >>) >

and *mean-definition* is one of the following forms:

[/ *mean-vector***]**

or

[MEAN=*mean-vector***]**

where *mean-vector* is a vector name.

COSAN stands for covariance structure analysis (McDonald 1978, 1980). The COSAN model in PROC CALIS is a generalized version of the original COSAN model. See the section "The COSAN Model" on page 1734 for details of the generalized COSAN model. You can analyze a very wide class of mean and covariance structures with the COSAN modeling language, which consists of the COSAN statement as the main model specification statement and the MATRIX statement as the subsidiary model specification statement. Use the following syntax to specify a COSAN model:

```
COSAN < VAR=variable-list, > term < + term ... > ;
    MATRIX matrix-name parameters-in-matrix ;
    /* Repeat the MATRIX statement as needed */  ;
    VARNAMES name-assignments ;
```

The PROC CALIS statement invokes the COSAN modeling language. You can specify at most one COSAN statement in a model within the scope of either the PROC CALIS statement or a MODEL statement. To complete the COSAN model specification, you might need to add as many MATRIX statements as needed. Optionally, you can provide the variable names for the COSAN model matrices in the VARNAMES statement.

In the COSAN statement, you specify the list of observed variables for analysis in the VAR= option and the formulas for covariance and mean structures in the *terms*. If specified at all, the VAR= option must be specified at the very beginning of the COSAN statement. The order of the variables in the VAR= option is important. It is the same order assumed for the row and column variables in the mean and covariance structures defined in the *terms*. If you do not specify the VAR= option, PROC CALIS selects all the numerical variables in the associated groups for analysis. To avoid confusion about the variables being analyzed in the model, it is recommended that you set the VAR= list explicitly in the COSAN statement.

To define the matrix formulas for the covariance and mean structures, you specify the *terms*, *matrix-definitions*, and *mean-vector* in the COSAN statement. The forms of the covariance and mean structures that are supported in PROC CALIS are mentioned in the section "The COSAN Model" on page 1734. In each *term*, you specify the covariance structures by listing the matrices in the *matrix-definitions*. These matrices must be in the proper order such that their matrix product produces the intended covariance structures. If you want to analyze the corresponding mean structures, specify the trailing *mean-vectors* in the *terms* whenever needed.

To illustrate the COSAN statement syntax, consider a factor-analytic model with six variables (var1–var6) and two factors. The covariance structures of the six variables are described by the matrix formula

$$\mathbf{\Sigma} = \mathbf{FPF}' + \mathbf{U}$$

where $\mathbf{\Sigma}$ is a 6×6 symmetric matrix for the covariance matrix, $\mathbf{F}$ is a 6×2 factor loading matrix, $\mathbf{P}$ is a 2×2 (symmetric) factor covariance matrix, and $\mathbf{U}$ is a 6×6 diagonal matrix of unique variances. You can use the following COSAN statement to specify the covariance structures of this factor model:

```
cosan var = var1-var6,
      F(2,GEN) * P(2,SYM) + U(6,DIA);
```

In the VAR= option of the COSAN statement, you define a list of six observed variables in the covariance structures. The order of the variables in the VAR= list determines the order of the row variables in the first matrix of each term in the model. That is, both matrices **F** and **U** have these six observed variables as their row variables, which are ordered the same way as in the VAR= list.

Next, you define the formula for the covariance structures by listing the matrices in the desired order *up to the central covariance matrix in each term*. In the first *term* of this example, you need to specify only **FP** instead of the complete covariance structure formula $\mathbf{FPF}'$. The reason is that the latter part of the term (that is, after the central covariance matrix) contains only the transpose of the matrices that have already been defined. Hence, PROC CALIS can easily generate the complete term with the nonredundant information given.

In each of the *matrix-definitions*, you can provide the number of columns in the first argument (that is, the *number-of-columns* field) inside a pair of parentheses. You do not need to provide the number of rows because this information can be deduced from the given covariance structure formula. By using some keywords, you can optionally provide the matrix type in the second argument (that is, the *matrix-type* field) and the matrix transformation in the third argument (that is, the *transformation* field).

In the current example, **F(2,GEN)** represents a general rectangular (GEN) matrix **F** with two columns. Implicitly, it has six rows because it is the first matrix of the first term in the covariance structure formula. **P(2,SYM)** represents a symmetric (SYM) matrix **P** with two columns. Implicitly, it has two rows because it is premultiplied with **F**, which has two columns. In the second term, **U(6,DIA)** represents a diagonal (DIA) matrix **U** with six rows and six columns. Because you do not specify the third argument in these *matrix-definitions*, no transformation is applied to any of the matrices in the covariance structure formula.

PROC CALIS supports the following keywords for *matrix-type*:

IDE specifies an identity matrix. If the matrix is not square, this specification describes an identity submatrix followed by a rectangular zero submatrix.

ZID specifies an identity matrix. If the matrix is not square, this specification describes a rectangular zero submatrix followed by an identity submatrix.

DIA specifies a diagonal matrix. If the matrix is not square, this specification describes a diagonal submatrix followed by a rectangular zero submatrix.

ZDI specifies a diagonal matrix. If the matrix is not square, this specification describes a rectangular zero submatrix followed by a diagonal submatrix.

LOW specifies a lower triangular matrix. The matrix can be rectangular.

UPP specifies an upper triangular matrix. The matrix can be rectangular.

SYM specifies a symmetric matrix. The matrix cannot be rectangular.

GEN specifies a general rectangular matrix (default).

If you omit the *matrix-type* argument, PROC CALIS sets the type of matrix by default. For central covariance matrices, the default for *matrix-type* is SYM. For all other matrices, the default for *matrix-type* is GEN. For example, if **A** is not a central covariance matrix in the covariance structure formula, the following specifications are equivalent for a general matrix **A** with three columns:

```
A(3,GEN)
A(3)
A(3,)
A(3, ,)
```

PROC CALIS supports the following two keywords for *transformation*:

INV uses the inverse of the matrix.

IMI uses the inverse of the difference between the identity and the matrix. For example, **A(3,GEN,IMI)** represents $(\mathbf{I} - \mathbf{A})^{-1}$.

Both INV or IMI require square (but not necessarily symmetric) matrices to transform. If you omit the *transformation* argument, no transformation is applied.

CAUTION: You can specify the same matrix by using the same *matrix-name* in different locations of the matrix formula in the COSAN statement. The *number-of-columns* and the *matrix-type* fields for matrices with identical *matrix-names* must be consistent. This consistency can be maintained easily by specifying each of these two fields only once in any of the locations of the same matrix. However, there is no restriction on the transformation for the same matrix in different locations. For example, while $\mathbf{R}$ must be the same 3×3 symmetric matrix throughout the formula in the following specification, the INV transformation of $\mathbf{R}$ applies only to the $\mathbf{R}$ matrix in the second term, but not to the same $\mathbf{R}$ matrix in the first term:

```
cosan var = var1-var6,
      B(3,GEN) * R(3,SYM) + H(3,DIA) * R(3,SYM,INV);
```

Mean and Covariance Structures

Suppose now you want to analyze the mean structures in addition to the covariance structures of the preceding factor model. The mean structure formula for $\boldsymbol{\mu}$ of the observed variables is

$$\boldsymbol{\mu} = \mathbf{F}\mathbf{v} + \mathbf{a}$$

where $\boldsymbol{\mu}$ is a 6×1 vector for the observed variable means, $\mathbf{v}$ is a 2×1 vector for the factor means, and $\mathbf{a}$ is a 6×1 vector for the intercepts of the observed variables. To include the mean structures in the COSAN model, you need to specify the mean vector at the end of the *terms*, as shown in the following statement:

```
cosan var = var1-var6,
      F(2,GEN) * P(2,SYM) [/ v] + U(6,DIA) [/ a];
```

If you take the mean vectors within the brackets away from each of the *terms*, the formula for the covariance structures is generated as

$$\boldsymbol{\Sigma} = \mathbf{F}\mathbf{P}\mathbf{F}' + \mathbf{U}$$

which is exactly the same covariance structure as described in a preceding example. Now, with the mean vectors specified at the end of each *term*, you analyze the corresponding mean structures simultaneously with the covariance structures.

To generate the mean structure formula, PROC CALIS replaces the central covariance matrices with the mean vectors in the *terms*. In the current example the mean structure formula is formed by replacing $\mathbf{P}$ and $\mathbf{U}$ with $\mathbf{v}$ and $\mathbf{a}$, respectively. Hence, the first term of the mean structure formula is $\mathbf{F} * \mathbf{v}$, and the second

term of the mean structure formula is simply $\mathbf{a}$. Adding these two terms yields the desired mean structure formula for the model.

To make the mean vector specification more explicit, you can use the following equivalent syntax with the MEAN= option:

```
cosan var = var1-var6,
      F(2,GEN) * P(2,SYM) [mean=v] + U(6,DIA) [mean=a];
```

If a *term* in the specification does not have a mean vector (covariance matrix) specification, a zero mean vector (null covariance matrix) is assumed. For example, the following specification generates the same mean and covariance structures as the preceding example:

```
cosan var = var1-var6,
      F(2,GEN) * P(2,SYM) [/ v] + U(6,DIA) + [/ a];
```

The covariance structure formula for this specification is

$$\mathbf{\Sigma} = \mathbf{FPF}' + \mathbf{U} + \mathbf{0}$$

where $\mathbf{0}$ in the last term represents a null matrix. The corresponding mean structure formula is

$$\boldsymbol{\mu} = \mathbf{Fv} + \mathbf{0} + \mathbf{a}$$

where $\mathbf{0}$ in the second term represents a zero vector.

Specifying Models with No Explicit Central Covariance Matrices

In some situations, the central covariance matrices in the covariance structure formula are not defined explicitly. For example, the covariance structure formula for an orthogonal factor model is:

$$\mathbf{\Sigma} = \mathbf{FF}' + \mathbf{U}$$

Again, assuming that $\mathbf{F}$ is a 6×2 factor loading matrix and $\mathbf{U}$ is a 6×6 diagonal matrix for unique variances, you can specify the covariance structure formula as in the following COSAN statement:

```
cosan var = var1-var6,
      F(2,GEN) + U(6,DIA);
```

In determining the proper formula for the covariance structures, PROC CALIS detects whether the last matrix specified in each *term* is symmetric. If you specify this last matrix explicitly with the SYM, IDE (with the same number of rows and columns), or DIA type, it is certainly a symmetric matrix. If you specify this last matrix without an explicit *matrix-type* and it has the same number of rows and columns, it is also treated as a symmetric matrix for the central covariance matrix of the *term*. Otherwise, this last matrix is not symmetric and PROC CALIS treats the *term* as if an identity matrix has been inserted for the central covariance matrix. For example, for the orthogonal factor model specified in the preceding statement, PROC CALIS correctly generates the first term as $\mathbf{FF}' = \mathbf{FIF}'$ and the second term as $\mathbf{U}$.

Certainly, you might also specify your own central covariance matrix explicitly for the orthogonal factor model. That is, you add an identity matrix into the COSAN model specification as shown in the following statement:

```
cosan var = var1-var6,
      F(2,GEN) * I(2,IDE) + U(6,DIA);
```

Specifying Mean Structures for Models with No Central Covariance Matrices

When you specify covariance structures with central covariance matrices explicitly defined in the *terms*, the corresponding mean structure formula is formed by replacing the central covariance matrices with the *mean-vectors* that are specified in the brackets. However, when there is no central covariance matrix explicitly specified in a *term*, the last matrix of the *term* in the covariance structure formula is replaced with the *mean-vector* to generate the mean structure formula. Consider the following specification where there is no central covariance matrix defined explicitly in the first *term* of the COSAN model:

```
cosan var = var1-var6,
      A(6,GEN) [ / v];
```

The generated formulas for the covariance and mean structures are

$$\begin{aligned} \boldsymbol{\Sigma} &= \mathbf{A}\mathbf{A}' \\ \boldsymbol{\mu} &= \mathbf{v} \end{aligned}$$

If, instead, you intend to fit the following covariance and mean structures

$$\begin{aligned} \boldsymbol{\Sigma} &= \mathbf{A}\mathbf{A}' \\ \boldsymbol{\mu} &= \mathbf{A}\mathbf{v} \end{aligned}$$

you must put an explicit identity matrix for the central covariance matrix in the first *term*. That is, you can use the following specification:

```
cosan var = var1-var6,
      A(6,GEN) * I(6,IDE) [ / v];
```

Specifying Parameters in Matrices

By specifying the COSAN statement, you define the covariance and mean structures in matrix formulas for the observed variables. To specify the parameters in the model matrices, you need to use the MATRIX statements.

For example, for an orthogonal factor model with six variables (var1–var6) and two factors, the 6×2 factor loading matrix $\mathbf{F}$ might take the following form:

$$\mathbf{F} = \begin{pmatrix} x & 0 \\ x & 0 \\ x & 0 \\ 0 & x \\ 0 & x \\ 0 & x \end{pmatrix}$$

The 6×6 unique variance matrix $\mathbf{U}$ might take the following form:

$$\mathbf{U} = \begin{pmatrix} x & 0 & 0 & 0 & 0 & 0 \\ 0 & x & 0 & 0 & 0 & 0 \\ 0 & 0 & x & 0 & 0 & 0 \\ 0 & 0 & 0 & x & 0 & 0 \\ 0 & 0 & 0 & 0 & x & 0 \\ 0 & 0 & 0 & 0 & 0 & x \end{pmatrix}$$

where each x in the matrices represents a free parameter to estimate and 0 represents a fixed zero value. The covariance structures for the observed variables are described by the following formula:

$$\mathbf{\Sigma} = \mathbf{F}\mathbf{F}' + \mathbf{U}$$

To specify the entire model, you use the following statements to define the covariance structure formula and the free parameters in the model matrices:

```
cosan var = var1-var6,
      F(2,GEN) + U(6,DIA);
matrix F [1 to 3,@1],[4 to 6,@2];
matrix U [1,1],[2,2],[3,3],[4,4],[5,5],[6,6];
```

In the MATRIX statements, you specify the free parameters in the matrices. For the factor loading matrix $\mathbf{F}$, you specify that rows 1, 2, and 3 in column 1 and rows 4, 5, and 6 in column 2 are free parameters. For the unique variance matrix $\mathbf{U}$, you specify that all diagonal elements are free parameters. All other unspecified entries in the matrices are fixed zeros by default. Certainly, you can also specify fixed zeros explicitly. For the current example, you can specify matrix $\mathbf{F}$ equivalently as:

```
matrix F [1 to 3,@1],[4 to 6,@2],
         [4 to 6,@1] = 0. 0. 0.,
         [1 to 3,@2] = 0. 0. 0.;
```

See the MATRIX statement on page 1638 for various ways to specify the parameters in matrices.

Matrix Names versus Parameter Names

Although parameter names and matrix names in PROC CALIS are both arbitrary SAS names for denoting mathematical entities in the model, their usages are very different in one aspect. That is, parameter names are globally defined in the procedure, while matrix names are only locally defined in models.

Consider the following two-group analysis example:

```
proc calis;
   group 1 / data=g1;
   group 2 / data=g2;
   model 1 / group=1;
      cosan var = var1-var6,
            F(2,GEN) * I(2,IDE) + U(6,DIA);
      matrix F [1 to 3,@1],[4 to 6,@2];
      matrix U [1,1] = u1-u6;
   model 2 / group=2;
      cosan var = var1-var6,
            F(1,GEN) * I(1,IDE) + D(6,DIA);
      matrix F [1 to 6,@1];
```

```
      matrix D [1,1] = u1-u6;
run;
```

In this example, you fit Model 1 to Group 1 and Model 2 to Group 2. You specify a matrix called **F** in each of the models. However, the two models are not constrained by this "same" matrix **F**. In fact, matrix **F** in Model 1 is a 6×2 matrix but matrix **F** in Model 2 is a 6×1 matrix. In addition, none of the parameters in the **F** matrices are constrained by the parameter names (simply because no parameter names are used). This illustrates that matrix names in PROC CALIS are defined only locally within models.

In contrast, in this example you use different matrix names for the second terms of the two models. In Model 1, you define a 6×6 diagonal matrix **U** for the second term; and in Model 2, you define a 6×6 diagonal matrix **D** for the second term. Are these two matrices necessarily different? The answer depends on how you define the parameters in these matrices. In the MATRIX statement for **U**, all diagonal elements of **U** are specified as free parameters u1–u6. Similarly, in the MATRIX statement for **D**, all diagonal elements of **D** as also specified free parameters u1–u6. Because you use the same sets of parameter names in both of these MATRIX statements, matrices **U** and **D** are essentially constrained to be the same even though their names are different. This illustrates that parameter names are defined globally in PROC CALIS.

The following points summarize how PROC CALIS treats matrix and parameter names differently:

- Matrices with the same name in the same model are treated as identical.
- Matrices with the same name in different models are not treated as identical.
- Parameters with the same name are identical throughout the entire PROC CALIS specification.
- Cross-model constraints on matrix elements are set by using the same parameter names, but not the same matrix names.

Row and Column Variable Names for Matrices

You can use the VARNAMES statement to define the column variable names for the model matrices of a COSAN model. However, you do not specify the row variable names for the model matrices directly because they are determined by the column variable names of the related matrices in the covariance and mean structure formulas. For example, the following specification names the column variables of matrices **F** and **I**:

```
cosan var = var1-var6,
      F(2,GEN) * I(2,IDE) + U(6,DIA);
varnames
      F = [Factor1 Factor2],
      I = F;
```

The column names for matrix **F** are Factor1 and Factor2. The row names of matrix **F** are var1–var6 because it is the first matrix in the first term. Matrix **I** has the same column variable names as those for matrix **F**, as specified in the last specification of the VARNAMES statement. Because matrix **I** is a central covariance matrix, its row variable names are the same as its column variable names: Factor1 and Factor2. You do not specify the column variables names for matrix **U** in the VARNAMES statement. However, because it is the first matrix in the second term, its row variable names are the same as that of the VAR= list in the COSAN statement. Because matrix **U** is also the central covariance matrix in the second term, its column variable names are the same its row variable names, which has been determined to be var1–var6. See the VARNAMES statement for more details.

Default Parameters

Unlike other modeling languages in PROC CALIS, the COSAN modeling language does not set any default free parameters for the model matrices. There is only one type of default parameters in the COSAN model: fixed values for matrix elements. These fixed values can be 0 or 1. For matrices with the IDE or ZID type, all elements are predetermined with either 0 or 1. They are fixed matrices in the sense that you cannot override these default fixed values. For all other matrix types, PROC CALIS sets their elements to fixed zeros by default. You can override these default zeros by specifying them explicitly in the MATRIX statements.

Modifying a COSAN Model from a Reference Model

In this section, it is assumed that you use a REFMODEL statement within the scope of a MODEL statement and the reference model (or base model) is also a COSAN model. The reference model is referred to as the old model, while the model that makes reference to this old model is referred to as the new model. If the new model is not intended to be an exact copy of the old model, you can use the following extended COSAN modeling language to make modifications within the scope of the MODEL statement for the new model. The syntax is similar to, but not exactly the same as, the ordinary COSAN modeling language. (See the section "COSAN Statement" on page 1583.) The respecification syntax for a COSAN model is as follows:

```
COSAN ;
    MATRIX matrix-name parameters-in-matrix ;
    /* Repeat the MATRIX statement as needed */ ;
    VARNAMES name-assignments ;
```

In the respecification, the COSAN statement is optional. In fact, the purpose of using the COSAN statement at all is to remind yourself that a COSAN model is used in the model definition. If you use the COSAN statement, you cannot specify the VAR= option or the covariance and mean structure formula. This means that the model form and the observed variable references of the new model must be the same as the old (reference) model. The reason for enforcing these model structures is to ensure that the MATRIX statement respecifications are consistently interpreted.

You can optionally use the VARNAMES statement in the respecification. If the variable names for a COSAN matrix are defined in the old model but not redefined the new model, all variable names for that matrix are duplicated in the new model. However, specification of variable names for a COSAN matrix in the new model overrides the corresponding specification in the old model.

You can respecify or modify the elements of the COSAN model matrices by using the MATRIX *matrix-name statements*. The syntax of the MATRIX statements for respecifications is the same as that in the ordinary COSAN modeling language, but with one more feature. In the respecification syntax, you can use the missing value '.' to drop a parameter specification from the old model.

The new model is formed by integrating with the old model in the following ways:

Duplication: If you do not specify in the new model a parameter location that exists in the old model, the old parameter specification is duplicated in the new model.

Addition: If you specify in the new model a parameter location that does not exist in the old model, the new parameter specification is used in the new model.

Deletion: If you specify in the new model a parameter location that also exists in the old model and the new parameter is denoted by the missing value '.', the old parameter specification is not copied into the new model.

Replacement: If you specify in the new model a parameter location that also exists in the old model and the new parameter is not denoted by the missing value '.', the new parameter specification replaces the old one in the new model.

For example, the following two-group analysis specifies Model 2 by referring to Model 1 in the REFMODEL statement:

```
proc calis;
   group 1 / data=d1;
   group 2 / data=d2;
   model 1 / group=1;
      cosan
         var = x1-x6,
         F(2,GEN) * PHI(2,SYM)  + PSI(6,SYM);
      matrix F           [1,1] = 1.,
                         [2,1] = load2,
                         [3,1] = load3,
                         [4,2] = 1.,
                         [5,2] = load5,
                         [6,2] = load6;
      matrix PHI         [1,1] = phi1,
                         [2,2] = phi2,
                         [2,1] = phi21;
      matrix PSI         [1,1] = psi1,
                         [2,2] = psi2,
                         [3,3] = psi3,
                         [4,4] = psi4,
                         [5,5] = psi5,
                         [6,6] = psi6;
      varnames F   = [Factor1 Factor2],
               PHI = F;
   model 2 / group=2;
      refmodel 1;
      matrix F           [3,1] = load2;    /* replacement */
      matrix PHI         [2,1] = .;        /* deletion */
      matrix PSI         [3,1] = psi31;    /* addition */
      varnames F   = [FF1 FF2],
run;
```

In this example, Model 2 is the new model which refers to the old model, Model 1. It illustrates the four types of model integration by using the MATRIX statements:

- Duplication: Except for the **F**[3, 1] and **PHI**[2, 1] elements, all parameter specifications in the old model are duplicated in the new model.
- Addition: The **PSI**[3, 1] element is added with a new parameter psi31 in the new model. This indicates the presence of a correlated error in Model 2, but not in Model 1.
- Deletion: The **PHI**[2, 1] element is no longer a free parameter in the new model. This means that the two latent factors are correlated in Model 1, but not in Model 2.
- Replacement: The **F**[3, 1] element defined in Model 2 replaces the definition in the old model. This element is now a free parameter named load2. Because the **F**[2, 1] element (via duplication from the

old model) is also a free parameter with this same name, **F**[3, 1] and **F**[2, 1] are constrained to be the same in Model 2, but not in Model 1.

With the VARNAMES statement specification in Model 1, the two columns of matrix **F** are labeled with Factor1 and Factor2, respectively. In addition, because PHI=F is specified in the VARNAMES statement of Model 1, the row and column of matrix **PHI** in Model 1 also contain Factor1 and Factor2 as the variable names. In Model 2, with the explicit VARNAMES specifications the two columns of matrix **F** are labeled with FF1 and FF2, respectively. These names are not the same as those for matrix **F** in the old (reference) model. However, because PHI=F is *not* specified in the VARNAMES statement of Model 2, the row and column of matrix **PHI** in Model 2 contain Factor1 and Factor2 as the variable names, which are duplicated from the old (reference) model.

COSAN Models and Other Models

Because the COSAN model is a more general model than any other model considered in PROC CALIS, you can virtually fit any other type of model in PROC CALIS by using the COSAN modeling language. See the section "Special Cases of the Generalized COSAN Model" on page 1736, Example 33.29, and Example 33.30 for illustrations and discussions.

In general, it is recommended that you use the more specific modeling languages such as FACTOR, LINEQS, LISMOD, MSTRUCT, PATH, and RAM. Because the COSAN model is very general in its formulation, PROC CALIS cannot exploit the specific model structures to generate reasonable initial estimates the way it does with other specific models such as FACTOR and PATH. If you do not provide initial estimates for a COSAN model, PROC CALIS uses some default starting values such as 0.5. See the START= option for controlling the starting value. See the RANDOM= option for setting random starting values. There are other reasons for preferring specific modeling languages whenever possible. The section "Which Modeling Language?" on page 1530 discusses these various reasons. However, when the covariance structures are complicated and are difficult to specify otherwise, the COSAN modeling language is a very useful tool. See Example 33.31 and Example 33.33 for illustrations.

COV Statement

COV *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents

var-list < ∗ *var-list2* > < **=** *parameter-spec* >

The COV statement is a subsidiary model specification statement for the confirmatory FACTOR and LINEQS models. In the LINEQS model, the COV statement defines the covariances among the exogenous variables, including errors and disturbances. In the confirmatory FACTOR model, the COV statement defines the factor covariances. In each *assignment* of the COV statement, you specify variables in the *var-list* and the *var-list2* lists, followed by the covariance parameter specification in the *parameter-spec* list. The latter two specifications are optional.

You can specify the following five types of the parameters for the covariances:

- an unnamed free parameter

- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

Consider a LINEQS model with exogenous variables V1, V2, V3, and V4. The following COV statement shows the five types of specifications in five *assignments*:

```
cov V2 V1 ,
    V3 V1 = (0.3),
    V3 V2 = 1.0,
    V4 V1 = phi1,
    V4 V2 = phi2(0.2);
```

In this statement, **cov(V2,V1)** is specified as an unnamed free parameter. For this covariance, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1). **cov(V3,V1)** is an unnamed free parameter but with an initial value of 0.3. PROC CALIS also generates a parameter name for this covariance. **cov(V3,V2)** is a fixed value of 1.0. This value stays the same in the estimation. **cov(V4,V1)** is a free parameter named phi1. **cov(V4,V2)** is a free parameter named phi2 with an initial value of 0.2.

Note that the *var-list* and *var-list2* lists on the left-hand side of the equal sign of the COV statement should contain only names of *exogenous* variables. Hence, the COV statement is different from the PCOV statement in which you can list both exogenous and endogenous variables, although the COV and PCOV statements share the same syntax.

You can use the COV statement for specifying covariance parameters in the FACTOR and LINEQS models. In the FACTOR model, the COV statement specifies the covariances among latent factors. In the LINEQS model, the COV statement specifies the covariances among all observed or latent exogenous variables, including error and disturbance terms.

If you specify only the *var-list* list, then you are specifying the so-called within-list covariances. If you specify both of the *var-list* and *var-list2* lists, then you are specifying the so-called between-list covariances. An asterisk is used to separate the two variable lists. You can use one of these two alternatives to specify the covariance parameters. Figure 33.2 illustrates the within-list and between-list covariance specifications.

Figure 33.2 Within-List and Between-List Covariances

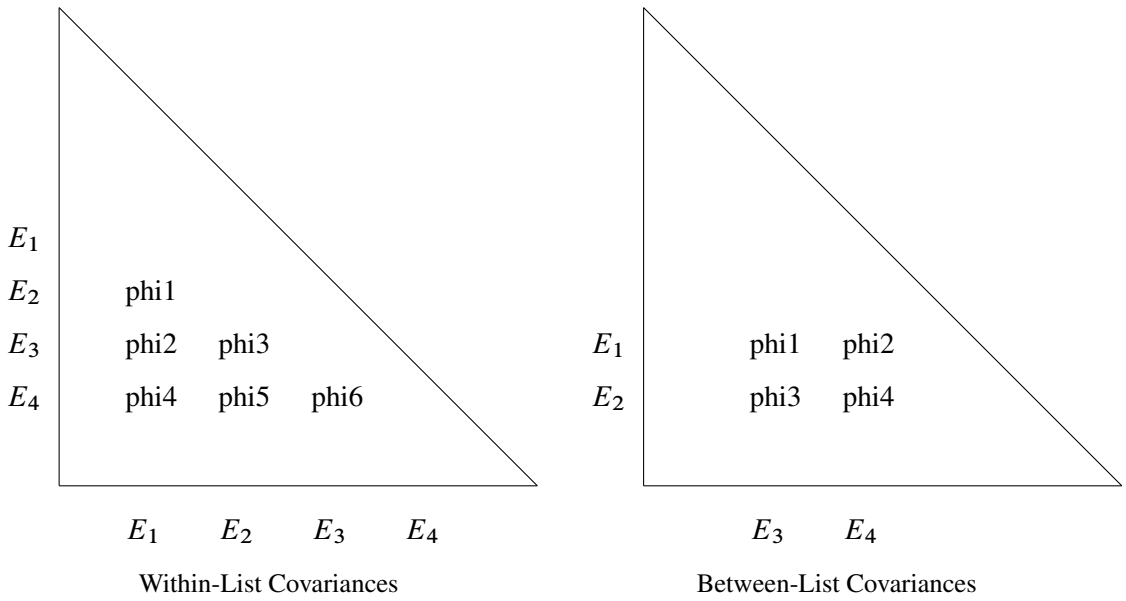

Within-List Covariances

The left panel of the figure shows that the same set of four variables are used in both the rows and columns. This yields six nonredundant covariances to specify. In general, with a *var-list* list with k variables in the COV statement, there are $k(k-1)/2$ distinct covariance parameters you can specify. The variable order of the *var-list* list is important. For example, the left panel of Figure 33.2 corresponds to the following COV statement specification:

```
cov E1-E4 = phi1-phi6;
```

This specification is equivalent to the following specification:

```
cov E2 E1 = phi1,
    E3 E1 = phi2, E3 E2 = phi3,
    E4 E1 = phi4, E4 E2 = phi5, E4 E3 = phi6;
```

Another way to assign distinct parameter names with the same prefix is to use the so-called prefix-name. For example, the following COV statement specification is exactly the same as the preceding specification:

```
cov E1-E4 = 6*phi__; /* phi with two trailing underscores */
```

In the COV statement, phi_ _ is a prefix-name with the root phi. The notation `6*` means this prefix-name is applied six times, resulting in a generation of the six parameter names phi1, phi2, . . . , phi6 for the six covariance parameters.

The root of the prefix-name should have few characters so that the generated parameter name is not longer than 32 characters. To avoid unintentional equality constraints, the prefix-names should not coincide with other parameter names.

You can also specify the within-list covariances as unnamed free parameters, as shown in the following statement:

```
cov E1-E4;
```

This specification is equivalent to the following specification:

```
cov E2 E1,
    E3 E1, E3 E2,
    E4 E1, E4 E2, E4 E3;
```

Between-List Covariances

The right panel of Figure 33.2 illustrates the application of the between-list covariance specification. The set of row variables is different from the set of column variables. You intend to specify the cross covariances of the two sets of variables. There are four of these covariances in the figure. In general, with k_1 and k_2 variable names in the two variable lists (separated by an asterisk) in a COV statement, there are $k_1 \times k_2$ distinct covariances to specify. Again, variable order is very important. For example, the right panel of Figure 33.2 corresponds to the following between-list covariance specification:

```
cov E1 E2 * E3 E4 = phi1-phi4;
```

This is equivalent to the following specification:

```
cov  E1 E3 = phi1, E1 E4 = phi2,
     E2 E3 = phi3, E2 E4 = phi4;
```

You can also use the prefix-name specification for the same specification, as shown in the following statement:

```
cov  E1 E2 * E3 E4 = 4*phi__ ; /* phi with two trailing underscores */
```

Mixed Parameter Lists

You can specify different types of parameters for the list of covariances. For example, you use a list of parameters with mixed types in the following statement:

```
cov E1-E4 = phi1(0.1)  0.2  phi3  phi4(0.4) (0.5) phi6;
```

This specification is equivalent to the following specification:

```
cov E2 E1 = phi1(0.1) ,
    E3 E1 = 0.2       , E3 E2 = phi3,
    E4 E1 = phi4(0.4) , E4 E2 = (0.5), E4 E3 = phi6;
```

Notice that an initial value that follows a parameter name is associated with the free parameter. Therefore, in the original mixed list specification, 0.1 is interpreted as the initial value for the parameter phi1, but not as the initial estimate for the covariance between E3 and E1. Similarly, 0.4 is the initial value for the parameter phi4, but not the initial estimate for the covariance between E4 and E2.

However, if you indeed want to specify that phi1 is a free parameter *without* an initial value and 0.1 is an initial estimate for the covariance between E3 and E1 (while keeping all other things the same), you can use a null initial value specification for the parameter phi1, as shown in the following statement:

```
cov E1-E4 = phi1() (0.1)  phi3  phi4(0.4) (0.5) phi6;
```

This way 0.1 becomes the initial estimate for the covariance between E3 and E1. Because a parameter list with mixed types might be confusing, you can break down the specifications into separate *assignments* to remove ambiguities. For example, you can use the following equivalent specification:

```
cov E2 E1 = phi1 ,
    E3 E1 = (0.1)     , E3 E2 = phi3,
    E4 E1 = phi4(0.4) , E4 E2 = (0.5), E4 E3 = phi6;
```

Shorter and Longer Parameter Lists

If you provide fewer parameters than the number of covariances in the variable lists, all the remaining parameters are treated as unnamed free parameters. For example, the following specification assigns a fixed value to **cov(E1,E3)** while treating all the other three covariances as unnamed free parameters:

```
cov  E1 E2 * E3 E4 = 1.0;
```

This specification is equivalent to the following specification:

```
cov  E1 E3 = 1.0, E1 E4, E2 E3, E2 E4;
```

If you intend to fill up all values by the last parameter specification in the list, you can use the continuation syntax **[...]**, **[..]**, or **[.]**, as shown in the following example:

```
cov  E1 E2 * E3 E4 = 1.0 phi [...];
```

This means that **cov(E1,E3)** is a fixed value of 1 and all the remaining three covariances are free parameter named phi. The last three covariances are thus constrained to be equal by using the same parameter name.

However, you must be careful not to provide too many parameters. For example, the following specification results in an error:

```
cov  E1 E2 * E3 E4 = 1.0 phi2(2.0) phi3 phi4 phi5 phi6;
```

The parameters after phi4 are excessive.

Default Covariance Parameters

In the confirmatory FACTOR model, by default all factor covariances are free parameters. In the LINEQS model, by default all covariances among exogenous manifest and latent variables (excluding error or disturbance variables) are also free parameters. For these default free parameters, PROC CALIS generate the parameter names with the _Add prefix and appended with unique integer suffixes. You can also use the COV statement specification to override these default covariance parameters in situations where you want to set parameter constraints, provide initial or fixed values, or make parameter references.

Another type of default covariances are fixed zeros. In the LINEQS model, covariances among errors or disturbances are all fixed zeros by default. Again, you can override the default fixed values by providing explicit specification of these covariances in the COV statement.

Modifying a Covariance Parameter Specification from a Reference Model

If you define a new model by using a reference (old) model in the REFMODEL statement, you might want to modify some parameter specifications from the COV statement of the reference model before transferring the specifications to the new model. To change a particular covariance specification from the reference model, you can simply respecify the same covariance with the desired parameter specification in the COV statement of the new model. To delete a particular covariance parameter from the reference model, you can specify the desired covariance with a missing value specification in the COV statement of the new model.

For example, suppose that the covariance between variables V1 and V2 is specified in the reference model but you do not want this covariance specification be transferred to the new model. You can use the following COV statement specification in the new model:

```
cov  V1 V2 = .;
```

Note that the missing value syntax is valid only when you use it with the REFMODEL statement. See the section "Modifying a LINEQS Model from a Reference Model" on page 1622 for a more detailed example of the LINEQS model respecification with the REFMODEL statement. See the section "Modifying a FACTOR Model from a Reference Model" on page 1608 for a more detailed example of the FACTOR model respecification with the REFMODEL statement.

As discussed in a preceding section, PROC CALIS generates some default free covariance parameters for the LINEQS and FACTOR models if you do not specify them explicitly in the COV statement. When you use the REFMODEL statement for defining a reference model, these default free covariance parameters in the old (reference) model are not transferred to the new model. Instead, the new model generates its own set of default free covariance parameters *after* it is resolved from the reference model, the REFMODEL statement options, the RENAMEPARM statement, and the COV statement specifications in the new model. This also implies that if you want any of the covariance parameters to be constrained across the models by means of the REFMODEL specification, you must specify them explicitly in the COV statement of the reference model so that the same covariance specification is transferred to the new model.

DETERM Statement

DETERM | STRUCTEQ *variables* < / *option* > **;**

where *option* represents:

LABEL | NAME= *name*

The DETERM statement is used to compute the determination coefficient of the listed dependent *variables* in the model. The precursor of the DETERM statement is the STRUCTEQ statement, which enables you to define the list of the dependent variables of the structural equations. Because the term *structural equation* is not defined in a unique way, a more generic concept of determination coefficients is revealed by the DETERM statement.

You can specify the DETERM statement as many times as you want for computing determination coefficients for the sets of dependent *variables* of interest. You can label each set of dependent variables by using the LABEL= option. Note that you cannot use the DETERM statement in an MSTRUCT model because there are no dependent variables in this type of model.

EFFPART Statement

EFFPART *effect* < **,** *effect . . .* > **;**

where *effect* represents:

var-list < *direction var-list2* >

and *direction* is the direction of the effect, as indicated by one of the following:
===>, **--->**, **==>**, **-->**, **=>**, **->**, **>**, **<===**, **<---**, **<==**, **<--**, **<=**, **<-**, or **<**.

In the EFFPART statement, you select those effects you want to analyze by partitioning the total effects into direct and indirect effects, with estimated standard errors. The EFFPART or TOTEFF option of the PROC CALIS statement also enables you to analyze effects. The difference is that the EFFPART or TOTEFF option displays effects on *all* endogenous variables, while the EFFPART statement shows only the effects of interest. In addition, the EFFPART statement enables you to arrange the effects in any way you like. Hence, the EFFPART statement offers a more precise and organized way to present various results of effects.

The EFFPART statement supports the following three types of effect specifications:

- **>**, **=>**, **->**, **==>**, **-->**, **===>**, or **--->** direction

  Example:

  ```
  effpart X1 X3-X5 ===> Y1 Y2;
  ```

  This will display *four* separate tables, respectively for the effects of X1, X3, X4, and X5 on Y1 and Y2. Each table contains the total, direct, and indirect effects of an X-variable on the two Y-variables.

- **<**, **<=**, **<-**, **<==**, **<--**, **<===**, or **<---** direction

  Example:

  ```
  effpart Y1 Y2 <=== X1 X3-X5;
  ```

  This will display *two* separate tables, respectively for the effects on Y1 and Y2, by X1, X3, X4, and X5. Each table contains the total, direct, and indirect effects of the four X-variables on a Y-variable. Certainly, the results produced from this statement are essentially the same as the previous statement. The difference is about the organization of the effects in the tables.

- no direction

  Example:

  ```
  effpart Y1 Y2 X1-X3;
  ```

  In this case, variables on the list are analyzed one by one to determine the nature of the effects. If a variable has nonzero effects on any other variables in the model, a table of the total, direct, and indirect effects of the variable on those variables is displayed. If a variable is endogenous, a table of total, direct, and indirect effects of those variables that have nonzero effects on the variable is displayed.

Note that an endogenous variable in a model might also have effects on other endogenous variables. Therefore, the two cases mentioned are not mutually exclusive—a variable listed in the EFFPART statement might yield two tables for effect analysis.

FACTOR Statement

FACTOR < *EFA-options* | *CFA-spec* > **;**

where *EFA-options* are options for the exploratory factor analysis that are described in the section "Exploratory Factor Analysis" on page 1600 and *CFA-spec* is a specification of confirmatory factor analysis that is described in the section "Confirmatory Factor Analysis" on page 1604.

In the FACTOR statement, you can specify either *EFA-options*, *CFA-spec*, or neither of these. However, you cannot specify both *EFA-options* and *CFA-spec* at the same time. If no option is specified or there is at least one *EFA-option* (exploratory factor analysis option) specified in the FACTOR statement, an exploratory factor analysis is conducted. Otherwise, a confirmatory factor analysis is conducted with the *CFA-spec*. These two types of models are discussed in the next two sections.

Exploratory Factor Analysis

FACTOR < *EFA-options* > **;**

For the exploratory factor model with orthogonal factors, PROC CALIS assumes the following model structures for the population covariance or correlation matrix $\mathbf{\Sigma}$:

$$\mathbf{\Sigma} = \mathbf{F}\mathbf{F}' + \mathbf{U}$$

where $\mathbf{F}$ is the factor loading matrix and $\mathbf{U}$ is a diagonal matrix of error variances. In this section, p denotes the number of manifest variables corresponding to the rows and columns of matrix $\mathbf{\Sigma}$, and n denotes the number of factors (or components, if the COMPONENT option is specified in the FACTOR statement) corresponding to the columns of the factor loading matrix $\mathbf{F}$. While the number of manifest variables is set automatically by the number of variables in the VAR statement or in the input data set, the number of factors can be set by the N= option in the FACTOR statement.

The unrestricted exploratory factor model is not identified because any orthogonal rotated factor loading matrix $\tilde{\mathbf{F}} = \mathbf{F}\mathbf{\Theta}$ satisfies the same model structures as $\mathbf{F}$ does, where $\mathbf{\Theta}$ is any orthogonal matrix so that $\mathbf{\Theta}'\mathbf{\Theta} = \mathbf{\Theta}\mathbf{\Theta}' = \mathbf{I}$. Mathematically, the covariance or correlation structures can be expressed as:

$$\mathbf{\Sigma} = \tilde{\mathbf{F}}\tilde{\mathbf{F}}' + \mathbf{U} = \mathbf{F}\mathbf{\Theta}\mathbf{\Theta}'\mathbf{F}' + \mathbf{U} = \mathbf{F}\mathbf{F}' + \mathbf{U}$$

To obtain an identified orthogonal factor solution as a starting point, the $n(n-1)/2$ elements in the upper triangle of $\mathbf{F}$ are constrained to zeros in PROC CALIS. Initial estimates for factor loadings and unique variances are computed by an algebraic method of approximate factor analysis. Given the initial estimates, final estimates are obtained through the iterative optimization of an objective function, which depends on the estimation method specified in the METHOD= option (default with ML—maximum likelihood) of the PROC CALIS statement.

To make the factor solution more interpretable, you can use the ROTATE= option in the FACTOR statement to obtain a rotated factor loading matrix with a "simple" pattern. Rotation can be orthogonal or oblique.

The rotated factors remain uncorrelated after an orthogonal rotation but would be correlated after an oblique rotation. The model structures of an oblique solution are expressed in the following equation:

$$\mathbf{\Sigma} = \tilde{\mathbf{F}}\mathbf{P}\tilde{\mathbf{F}}' + \mathbf{U}$$

where $\tilde{\mathbf{F}}$ is the rotated factor loading matrix and $\mathbf{P}$ is a symmetric matrix for factor correlations. See the sections "The FACTOR Model" on page 1738 and "Exploratory Factor Analysis Models" on page 1740 for more details about exploratory factor models.

You can also do exploratory factor analysis by the more dedicated FACTOR procedure. Even though extensive comparisons of the factor analysis capabilities between the FACTOR and CALIS procedures are not attempted here, some general points can be made here. In general, the FACTOR procedure provides more factor analysis options than the CALIS procedure does, although both procedures have some unique factor analysis features that are not shared by the other. PROC CALIS requires more computing time and memory than PROC FACTOR because it is designed for more general structural estimation problems and is not able to exploit all the special properties of the unconstrained factor analysis model. For maximum likelihood analysis, you can use either PROC FACTOR (with METHOD=ML, which is not the default method in PROC FACTOR) or PROC CALIS. Because the initial unrotated factor solution obtained by PROC FACTOR uses a different set of identification constraints than that of PROC CALIS, you would observe different initial ML factor solutions for the procedures. Nonetheless, the initial solutions by both procedures are statistically equivalent.

The following *EFA-options* are available in the FACTOR statement:

COMPONENT
COMP

computes a component analysis instead of a factor analysis (the diagonal matrix $\mathbf{U}$ in the model is set to 0). Note that the rank of $\mathbf{FF}'$ is equal to the number n of components in $\mathbf{F}$. If n is smaller than the number of variables in the moment matrix $\mathbf{\Sigma}$, the matrix of predicted model values is singular and maximum likelihood estimates for $\mathbf{F}$ cannot be computed. You should compute ULS estimates in this case.

HEYWOOD
HEY

constrains the diagonal elements of $\mathbf{U}$ to be nonnegative. Equivalently, you can constrain these elements to positive values by the BOUNDS statement.

GAMMA=*p*

specifies the orthomax weight used with the option ROTATE=ORTHOMAX. Alternatively, you can use ROTATE=ORTHOMAX(*p*) with *p* representing the orthomax weight. There is no restriction on valid values for the orthomax weight, although the most common values are between 0 and the number of variables. The default GAMMA= value is one, resulting in the varimax rotation.

N=*n*

specifies the number of first-order factors or components. The number of factors (n) should not exceed the number of manifest variables (p) in the analysis. For the saturated model with $n = p$, the COMP option should generally be specified for $\mathbf{U} = 0$; otherwise, $df < 0$. For $n = 0$ no factor loadings are estimated, and the model is $\mathbf{\Sigma} = \mathbf{U}$, with a diagonal $\mathbf{U}$ matrix. By default, $n = 1$.

NORM< = KAISER | NONE >

Kaiser-normalizes the rows of the factor pattern for rotation. NORM=KAISER, which is the default, has exactly the same effect as NORM. You can turn off the normalization by NORM=NONE.

RCONVERGE=*p*

RCONV=*p*

specifies the convergence criterion for rotation cycles. Rotation stops when the scaled change of the simplicity function value is less than the RCONVERGE= value. The default convergence criterion is:

$$|f_{new} - f_{old}|/K < \epsilon$$

where f_{new} and f_{old} are simplicity function values of the current cycle and the previous cycle, respectively, $K = \max(1, |f_{old}|)$ is a scaling factor, and ϵ is 1E–9 by default and is modified by the RCONVERGE= value.

RITER=*i*

specifies the maximum number of cycles *i* for factor rotation. The default *i* is the greater of 10 times the number of variables and 100.

ROTATE=*name*

R=*name*

specifies an orthogonal or oblique rotation of the initial factor solution. Although ROTATE=PRINCIPAL is actually not a rotation method, it is put here for convenience. By default, ROTATE=NONE.

Valid *names* for orthogonal rotations are as follows:

BIQUARTIMAX | BIQMAX specifies orthogonal biquartimax rotation. This corresponds to the specification ROTATE=ORTHOMAX(0.5).

EQUAMAX | E specifies orthogonal equamax rotation. This corresponds to the specification ROTATE=ORTHOMAX with GAMMA=*n*/2.

FACTORPARSIMAX | FPA specifies orthogonal factor parsimax rotation. This corresponds to the specification ROTATE=ORTHOMAX with GAMMA=*n*.

NONE | N specifies that no rotation be performed, leaving the original orthogonal solution.

ORTHCF(*p1***,***p2***) | ORCF(***p1***,***p2***)** specifies the orthogonal Crawford-Ferguson rotation (Crawford and Ferguson 1970) with the weights *p1* and *p2* for variable parsimony and factor parsimony, respectively. See the definitions of weights in Chapter 44, "The FACTOR Procedure."

ORTHGENCF(*p1***,***p2***,***p3***,***p4***) | ORGENCF(***p1***,***p2***,***p3***,***p4***)** specifies the orthogonal generalized Crawford-Ferguson rotation (Jennrich 1973), with the four weights *p1*, *p2*, *p3*, and *p4*. For the definitions of these weights, see the section "Simplicity Functions for Rotations" on page 2906 in Chapter 44, "The FACTOR Procedure."

ORTHOMAX< (*p1***) > | ORMAX< (***p1***) >** specifies the orthomax rotation (see Harman 1976) with orthomax weight *p1*. If ROTATE=ORTHOMAX is used, the default *p1* value is 1 unless specified otherwise in the GAMMA= option. Alternatively, ROTATE=ORTHOMAX(*p1*) specifies *p1* as the orthomax weight or the GAMMA= value. For the definitions of the orthomax weight, see the section "Simplicity Functions for Rotations" on page 2906 in Chapter 44, "The FACTOR Procedure."

PARSIMAX | PA specifies orthogonal parsimax rotation. This corresponds to the specification ROTATE=ORTHOMAX with

$$\text{GAMMA} = \frac{p \times (n-1)}{p+n-2}$$

PRINCIPAL | PC specifies a principal axis rotation. If ROTATE=PRINCIPAL is used with a factor rather than a component model, the following rotation is performed:

$$\mathbf{F}_{new} = \mathbf{F}_{old}\mathbf{T}, \quad \text{with} \quad \mathbf{F}'_{old}\mathbf{F}_{old} = \mathbf{T}\mathbf{\Lambda}\mathbf{T}'$$

where the columns of matrix $\mathbf{T}$ contain the eigenvectors of $\mathbf{F}'_{old}\mathbf{F}_{old}$.

QUARTIMAX | QMAX | Q specifies orthogonal quartimax rotation. This corresponds to the specification ROTATE=ORTHOMAX(0).

VARIMAX | V specifies orthogonal varimax rotation. This corresponds to the specification ROTATE=ORTHOMAX with GAMMA=1.

Valid *names* for oblique rotations are as follows:

BIQUARTIMIN | BIQMIN specifies biquartimin rotation. It corresponds to the specification ROTATE=OBLIMIN(.5) or ROTATE=OBLIMIN with TAU=.5.

COVARIMIN | CVMIN specifies covarimin rotation. It corresponds to the specification ROTATE=OBLIMIN(1) or ROTATE=OBLIMIN with TAU=1.

OBBIQUARTIMAX | OBIQMAX specifies oblique biquartimax rotation.

OBEQUAMAX | OE specifies oblique equamax rotation.

OBFACTORPARSIMAX | OFPA specifies oblique factor parsimax rotation.

OBLICF(*p1*,*p2*) | OBCF(*p1*,*p2*) specifies the oblique Crawford-Ferguson rotation (Crawford and Ferguson 1970) with the weights *p1* and *p2* for variable parsimony and factor parsimony, respectively. For the definitions of these weights, see the section "Simplicity Functions for Rotations" on page 2906 in Chapter 44, "The FACTOR Procedure."

OBLIGENCF(*p1*,*p2*,*p3*,*p4*) | OBGENCF(*p1*,*p2*,*p3*,*p4*) specifies the oblique generalized Crawford-Ferguson rotation (Jennrich 1973) with the four weights *p1*, *p2*, *p3*, and *p4*. For the definitions of these weights, see the section "Simplicity Functions for Rotations" on page 2906 in Chapter 44, "The FACTOR Procedure."

OBLIMIN<(*p1*)> | OBMIN<(*p1*)> specifies the oblimin rotation with oblimin weight *p1*. If ROTATE=OBLIMIN is used, the default *p1* value is zero unless specified otherwise in the TAU= option. Alternatively, ROTATE=OBLIMIN(*p1*) specifies *p1* as the oblimin weight or the TAU= value. For the definitions of the oblimin weight, see the section "Simplicity Functions for Rotations" on page 2906 in Chapter 44, "The FACTOR Procedure."

OBPARSIMAX | OPA specifies oblique parsimax rotation.

OBQUARTIMAX | OQMAX specifies oblique quartimax rotation. This is the same as the QUARTIMIN method.

OBVARIMAX | OV specifies oblique varimax rotation.

QUARTIMIN | QMIN specifies quartimin rotation. It is the same as the oblique quartimax method. It also corresponds to the specification ROTATE=OBLIMIN(0) or ROTATE=OBLIMIN with TAU=0.

TAU=*p*

specifies the oblimin weight used with the option ROTATE=OBLIMIN. Alternatively, you can use ROTATE=OBLIMIN(*p*) with *p* representing the oblimin weight. There is no restriction on valid values for the oblimin weight, although for practical purposes a negative or zero value is recommended. The default TAU= value is 0, resulting in the quartimin rotation.

Confirmatory Factor Analysis

FACTOR *factor-variables-relation* < **,** *factor-variables-relation . . .* > **;**

where each *factor-variables-relation* is defined as

factor right-arrow var-list < **=** *parameter-spec* >

where *right-arrow* is one of the following: **===>**, **--->**, **==>**, **-->**, **=>**, **->**, or **>**.
To complete the specification of a confirmatory factor model, you might need to use the PVAR, COV, and MEAN statements to specify the variance, partial variance, covariance, and mean parameters in the model, as shown in the following syntax:

FACTOR *factor-variable-relation* < **,** *factor-variables-relation . . .* > **;**
PVAR *partial-variance-parameters* **;**
COV *covariance-parameters* **;**
MEAN *mean-parameters* **;**

The model structures for the covariance matrix $\mathbf{\Sigma}$ of the confirmatory factor model are described in the equation

$$\mathbf{\Sigma} = \mathbf{FPF}' + \mathbf{U}$$

where $\mathbf{F}$ is the factor loading matrix, $\mathbf{P}$ is a symmetric matrix for factor correlations, and $\mathbf{U}$ is a diagonal matrix of error variances.

If the mean structures are also analyzed, the model structures for the mean vector $\boldsymbol{\mu}$ of the confirmatory factor model are described in the equation

$$\boldsymbol{\mu} = \boldsymbol{\alpha} + \mathbf{F}\boldsymbol{\nu}$$

where $\boldsymbol{\alpha}$ is the intercept vector for the observed variables and $\boldsymbol{\nu}$ is the vector for factor means. See the sections "The FACTOR Model" on page 1738 and "Confirmatory Factor Analysis Models" on page 1741 for more details about confirmatory factor models.

The FACTOR statement is the main model specification statement for the confirmatory factor model. The specifications in the FACTOR statement concern the factor loading pattern in the $\mathbf{F}$ matrix. More details follow after a brief description of the subsidiary model specification statements: PVAR, COV, and MEAN.

By default, the factor variance parameters in the diagonal of matrix $\mathbf{P}$ and the error variances in the diagonal of matrix $\mathbf{U}$ are free parameters in the confirmatory factor model. However, you can override these default parameters by specifying them explicitly in the PVAR statement. For example, in some confirmatory factor

models, you might want to set some of these variances to fixed constants, or you might want to set equality constraints by using the same parameter name at different parameter locations in your model.

By default, factor covariances, which are the off-diagonal elements of matrix $\mathbf{P}$, are free parameters in the confirmatory factor model. However, you can override these default covariance parameters by specifying them explicitly in the COV statement. Note that you cannot use the COV statement to specify the error covariances—they are always fixed zeros in the confirmatory factor analysis model.

By default, all factor means are fixed zeros and all intercepts are free parameters if the mean structures are analyzed. You can override these defaults by explicitly specifying the means of the factors in vector $\boldsymbol{\nu}$ and the intercepts of the manifest variables in vector $\boldsymbol{\alpha}$ in the MEAN statement.

Because the default parameterization of the confirmatory FACTOR model already covers most commonly used parameters in matrices $\mathbf{P}$, $\mathbf{U}$, $\boldsymbol{\alpha}$, and $\boldsymbol{\nu}$, the specifications in the PVAR, COV, and MEAN statements are secondary to the specifications in the FACTOR statement, which specifies the factor pattern of the $\mathbf{F}$ matrix. The following example statement introduces the syntax of the confirmatory FACTOR statement. Suppose that there are nine manifest variables V1–V9 in your sample and you want to fit a model with four factors, as shown in the following FACTOR statement:

```
factor
   g_factor   ===>   V1-V9 ,
   factor_a   ===>   V1-V3 ,
   factor_b   ===>   V4-V6 ,
   factor_c   ===>   V7-V9 ;
```

In this factor model, you assume a general factor g_factor and three group-factors: factor_a, factor_b, and factor_c. The general factor g_factor is related to all manifest variables in the sample, while each group-factor is related only to three manifest variables. This example fits the following pattern of factor pattern of $\mathbf{F}$:

	g_factor	factor_a	factor_b	factor_c
V1	x	x		
V2	x	x		
V3	x	x		
V4	x		x	
V5	x		x	
V6	x		x	
V7	x			x
V8	x			x
V9	x			x

where an x represents an unnamed free parameter and all other cells that are blank are fixed zeros. For each of these unnamed parameters, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1, _Parm2 and so on).

An unnamed free parameter is only one of the following five types of parameters (*parameter-spec*) you can specify at the end of each *factor-variables-relation*:

- an unnamed free parameter
- an initial value

- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

To illustrate these different types of parameter specifications, consider the following factor pattern for **F**:

```
          g_factor   factor_a   factor_b   factor_c

V1         g_load1    1.
V2         g_load2    x
V3         g_load3    x
V4         g_load4               1.
V5         g_load5               load_a
V6         g_load6               load_b
V7         g_load7                          1.
V8         g_load8                          load_c
V9         g_load9                          load_c
```

where an x represents an unnamed free parameter, a constant 1 represents a fixed value, and each name in a cell represents a name for a free parameter. You can specify this factor pattern by using the following FACTOR statement:

```
factor
   g_factor   ===>   V1-V9    = g_load1-g_load9 (9*0.6),
   factor_a   ===>   V1-V3    = 1. (.7  .8),
   factor_b   ===>   V4-V6    = 1. load_a (.9) load_b,
   factor_c   ===>   V7-V9    = 1. 2*load_c ;
```

In the first entry of the FACTOR statement, you specify that the loadings of V1–V9 on g_factor are free parameters g_load1–g_load9 with all given an initial estimate of 0.6. The syntax `9*0.6` means that `0.6` is repeated nine times. Because they are enclosed in a pair parentheses, all these values are treated as initial estimates, but not fixed values.

The second entry of the FACTOR statement can be split into the following specification:

```
factor_a   ===>   V1    = 1.  ,
factor_a   ===>   V2    = (.7),
factor_a   ===>   V3    = (.8),
```

This means that the first loading is a fixed value of 1, while the other loadings are unnamed free parameters with initial estimates 0.7 and 0.8, respectively. For each of these unnamed parameters with initial values, PROC CALIS also generates a parameter name with the _Parm prefix and appended with a unique integer.

The third entry of the FACTOR statement can be split into the following specification:

```
factor_b   ===>   V4    = 1.  ,
factor_b   ===>   V5    = load_a (.9),
factor_b   ===>   V6    = load_b,
```

This means that the first loading is a fixed value of 1, the second loading is a free parameter named load_a with an initial estimate of 0.9, and the third loading is a free parameter named load_b without an initial estimate. PROC CALIS generates the initial value for this free parameter.

The fourth entry of the FACTOR statement states that the first loading is a fixed 1 and the remaining two loadings are free parameters named load_c. No initial estimate is given. But because the two loadings have the same parameter name, they are constrained to be equal in the estimation.

Notice that an initial value that follows after a parameter name is associated with the free parameter. For example, in the third entry of the FACTOR statement, the specification `(.9)` after load_a is interpreted as the initial value for the parameter load_a, but not as the initial estimate for the next loading for V6.

However, if you indeed want to specify that load_a is a free parameter *without* an initial value and `(0.9)` is an initial estimate for the loading for V6, you can use a null initial value specification for the parameter load_a, as shown in the following specification:

```
factor_b   ===>   V4-V6    = 1. load_a() (.9),
```

This way 0.9 becomes the initial estimate of the loading for V6. Because a parameter list with mixed parameter types might be confusing, you can split the specification into separate entries to remove ambiguities. For example, you can use the following equivalent specification:

```
factor_b   ===>   V4    = 1.,
factor_b   ===>   V5    = load_a,
factor_b   ===>   V6    = (.9),
```

Shorter and Longer Parameter Lists

If you provide fewer parameters than the number of loadings that are specified in the corresponding *factor-variable-relation*, all the remaining parameters are treated as unnamed free parameters. For example, the following specification assigns a fixed value of 1.0 to the first loading, while treating the remaining two loadings as unnamed free parameters:

```
factor
   factor_a   ===>   V1-V3    = 1.;
```

This specification is equivalent to the following specification:

```
factor
   factor_a   ===>   V1     = 1.,
   factor_a   ===>   V2 V3      ;
```

If you intend to fill up all values with the last parameter specification in the list, you can use the continuation syntax `[...]`, `[..]`, or `[.]`, as shown in the following example:

```
factor
   g_factor   ===>   V1-V30    = 1.  (.5) [...];
```

This means that the loading of V1 on g_factor is a fixed value of 1.0, while the remaining 29 loadings are unnamed free parameters with all given an initial estimate of 0.5.

However, you must be careful not to provide too many parameters. For example, the following specification results in an error:

```
factor
   g_factor   ===>   V1-V3    = load1-load6;
```

The parameter list has six parameters for three loadings. Parameters after load3 are excessive.

Default Parameters

It is important to understand the default parameters in the FACTOR model. First, if you know which parameters are default free parameters, you can make your specification more efficient by omitting the specifications of those parameters that can be set by default. For example, because all error variances in the confirmatory FACTOR model are free parameters by default, you do not need to specify them with the PVAR statement if these error variances are not constrained. Second, if you know which parameters are default free parameters, you can specify your model accurately. For example, because all factor variance and covariances in the confirmatory FACTOR model are free parameters by default, you must use the COV statement to restrict the covariances among the factors if you want to fit an orthogonal factor model. See the section "Default Parameters in the FACTOR Model" on page 1745 for details about the default parameters of the FACTOR model.

Modifying a FACTOR Model from a Reference Model

This section assumes that you use a REFMODEL statement within the scope of a MODEL statement and that the reference model (or base model) is a factor model, either exploratory or confirmatory. The reference model is called the old model, and the model that refers to the old model is called the new model. If the new model is not intended to be an exact copy of the old FACTOR model, you can use the extended FACTOR modeling language described in this section to make modifications from the old model before transferring the specifications to the new model.

Using the REFMODEL statement for defining new factor models is not recommended in the following cases:

- If your old model is an exploratory factor analysis model, then specification by using the FACTOR modeling language in the new model replaces the old model completely. In this case, the use of the REFMODEL statement is superfluous and should be avoided.
- If your old model is a confirmatory factor analysis model, then specification of an exploratory factor model by using the FACTOR statement in the new model also replaces the old model completely. Again, the use of the REFMODEL statement is superfluous and should be avoided.

The nontrivial case where you might find the REFMODEL statement useful is when you modify an old confirmatory factor model to form a new confirmatory factor model. This nontrivial case is the focus of discussion in the remaining of the section.

The extended FACTOR modeling language for modifying model specification bears the same syntax as that of the ordinary FACTOR modeling language (see the section "Confirmatory Factor Analysis" on page 1604). The syntax is:

FACTOR *factor-variable-relation* **;**
  PVAR *partial-variance-parameters* **;**
  COV *covariance-parameters* **;**
  MEAN *mean-parameters* **;**

The new model is formed by integrating with the old model in the following ways:

Duplication: If you do not specify in the new model a parameter location that exists in the old model, the old parameter specification is duplicated in the new model.

Addition: If you specify in the new model a parameter location that does not exist in the old model, the new parameter specification is added in the new model.

Deletion: If you specify in the new model a parameter location that also exists in the old model and the new parameter is denoted by the missing value '.', the old parameter specification is not copied into the new model.

Replacement: If you specify in the new model a parameter location that also exists in the old model and the new parameter is not denoted by the missing value '.', the new parameter specification replaces the old one in the new model.

For example, consider the following two-group analysis:

```
proc calis;
   group 1 / data=d1;
   group 2 / data=d2;
   model 1 / group=1;
      factor
         F1 ===> V1-V3    = 1. load1 load2,
         F2 ===> V4-V6    = 1. load3 load4,
         F3 ===> V7-V9    = 1. load5 load6;
      cov
         F1 F2 = c12,
         F2 F3 = c23;
      pvar
         F1-F3 = c1-c3,
         V1-V9 = ev1-ev9;
   model 2 / group=2;
      refmodel 1;
      factor
         F1 ===> V1    = loada,
         F2 ===> V4    = loadb,
         F3 ===> V7    = loadc;
      cov
         F1 F2 = .,
         F1 F3 = c13;
run;
```

In this specification, you specify Model 2 by referring to Model 1 in the REFMODEL statement; Model 2 is the new model which refers to the old model, Model 1. Because the PVAR statement is not used in new model, all variance and partial variance parameter specifications in the PVAR statement of the old model are duplicated in the new model. The covariance parameter c23 for covariance between F2 and F3 in the COV statement of the old model is also duplicated in the new model. Similarly, loading parameters load1–load6 for some specific factor matrix locations are duplicated from the old model to the new model.

The new model has an additional parameter specification that the old model does not have. In the COV statement of the new model, covariance parameter c13 for the covariance between F1 and F3 is added.

In the same statement, the covariance between F1 and F2 is denoted by the missing value '.'. The missing value indicates that this parameter location in the old model should not be included in the new model. The consequence of this deletion from the old model is that the covariance between F1 and F2 is a fixed zero in the new model.

Finally, the three new loading specifications in the FACTOR statement of the new model replace the fixed ones in the old model. They are now free parameters loada, loadb, and loadc in the new model.

FITINDEX Statement

FITINDEX *option* < *option* . . . > **;**

You can use the FITINDEX statement to set the options for computing and displaying the fit indices, or to output the fit indices. All but the OFF= and ON= options of the FITINDEX statement are also available in the PROC CALIS statement. The options set in the FITINDEX statement will overwrite those set in the PROC CALIS statement.

For the listing of fit indices and their definitions, see the section "Overall Model Fit Indices" on page 1827. Note that not all fit indices are available with all estimation methods, which is specified by the METHOD= option of the PROC CALIS statement. See the section "Fit Indices and Estimation Methods" on page 1834 for more details.

The *options* of the FITINDEX statement are as follows:

ALPHAECV=α

specifies a $(1-\alpha)100\%$ confidence interval ($0 \leq \alpha \leq 1$) for the Browne and Cudeck (1993) expected cross-validation index (ECVI). See the ALPHAECV= option of the PROC CALIS statement on page 1543.

ALPHARMS=α

specifies a $(1-\alpha)100\%$ confidence interval ($0 \leq \alpha \leq 1$) for the Steiger and Lind (1980) root mean square error of approximation (RMSEA) coefficient. See the ALPHARMS= option of the PROC CALIS statement on page 1543.

BASEFIT=*SAS-data-set*

INBASEFIT=*SAS-data-set*

inputs the *SAS-data-set* that contains the fit information of the baseline model of your choice. See the BASEFIT= option of the PROC CALIS statement on page 1544.

BASEFUNC=*r***(< DF= >***i***)**

BASEFUNC(< DF= >*i***)=***r*

inputs the fit function value *r* and the degrees of freedom *i* of the baseline model of your choice. See the BASEFUNC= option of the PROC CALIS statement on page 1545.

CHICORRECT=*name* | *c*

CHICORR=*name* | *c*

specifies a correction factor *c* for the chi-square statistics for model fit. See the CHICORRECT= option of the PROC CALIS statement on page 1547.

CLOSEFIT=*p*

defines the criterion value *p* for indicating a close fit. See the CLOSEFIT= option of the PROC CALIS statement on page 1549.

DFREDUCE=*i*

reduces the degrees of freedom of the χ^2 test by *i*. See the DFREDUCE= option of the PROC CALIS statement on page 1553.

NOADJDF

turns off the automatic adjustment of degrees of freedom when there are active constraints in the analysis. See the NOADJDF option of the PROC CALIS statement on page 1565.

NOINDEXTYPE

disables the display of index types in the fit summary table. See the NOINDEXTYPE option of the PROC CALIS statement on page 1565.

OFF= [*names***] | {***names***}**

OFFLIST= [*names***] | {***names***}**

turns off the printing of one or more fit indices or modeling information as indicated by *names*, where a *name* represents a fit index, a group of fit indices, or modeling information. *Names* must be specified inside a pair of parentheses and separated by spaces. By default, all fit indices are printed. See the ON= option for the value of *names*.

ON < (ONLY) > = [*names***] | {***names***}**

ONLIST < (ONLY) > = [*names***] | {***names***}**

turns on the printing of one or more fit indices or modeling information as indicated by *names*, where a *name* represents a fit index, a group of fit indices, or modeling information. *Names* must be specified inside a pair of parentheses and separated by spaces. Because all fit indices and modeling information are printed by default, using an ON= list alone is redundant. When both ON= and OFF= lists are specified, the ON= list will override the OFF= list for those fit indices or modeling information that appear on both lists. If an ON(ONLY)= list is used, only those fit indices or modeling information specified in the list will be printed. Effectively, an ON(ONLY)= list is the same as the specification with an ON= list with the same selections and an OFF=ALL list in the FITINDEX statement.

Output Control of Fit Index Groups and Modeling Information Group

You can use the following *names* to refer to the groups of fit indices or modeling information available in PROC CALIS:

ABSOLUTE	Absolute or stand-alone fit indices that measures the model fit without using a baseline model.
ALL	All fit indices available in PROC CALIS.
INCREMENTAL	Incremental fit indices that measure model fit by comparing with a baseline model.
MODELINFO	General modeling information including sample size, number of variables, number of variables, and so on.
PARSIMONY	Fit indices that take model parsimony into account.

Output Control of Modeling Information

You can use the following *names* to refer to the individual modeling information available in PROC CALIS:

BASECHISQ Chi-square statistic for the baseline model.

BASEDF Degrees of freedom of the chi-square statistic for the baseline model.

BASEFUNC Baseline model function value.

BASELOGLIKE Baseline model –2 log-likelihood function value for METHOD=FIML.

BASEPROBCHI | BASEPROBCHISQ *P*-value of the chi-square statistic for the baseline model fit.

BASEPROBSBCHI | BASEPROBSBCHISQ *P*-value of the Satorra-Bentler scaled chi-square statistic for the baseline model fit.

BASESBCHISQ Satorra-Bentler scaled chi-square statistic for the baseline model.

BASESTATUS Status of the baseline model fitting for METHOD=FIML.

NACTCON Number of active constraints.

NIOBS Number of incomplete observations for METHOD=FIML.

NMOMENTS Number of elements in the moment matrices being modeled.

NOBS Number of observations assumed in the analysis.

NPARM | NPARMS Number of independent parameters.

NVAR Number of variables.

SATFUNC Saturated model function value for METHOD=FIML.

SATLOGLIKE Saturated model –2 log-likelihood function value for METHOD=FIML.

SATSTATUS Status of the saturated model fitting for METHOD=FIML.

Output Control of Absolute Fit Indices

You can use the following *names* to refer to the individual absolute fit indices available in PROC CALIS:

CHISQ Chi-square statistic for model fit.

CN | CRITICAL_N Hoelter's critical N.

CONTLIKE Percentage contribution to the log-likelihood function value of each group in multiple-group analyses with METHOD=FIML.

CONTRIBUTION | CONTCHI Percentage contribution to the chi-square value for multiple-group analyses.

DF Degrees of freedom for the chi-square test for model fit.

ELLIPTIC Elliptical chi-square statistic for ML and GLS methods in single-group analyses without mean structures. This index is computed only when you input the raw data with the KURTOSIS option specified.

FUNCVAL Optimized function value.

GFI Goodness-of-fit index by Jöreskog and Sörbom.

LOGLIKE Fitted model –2 log-likelihood function value for METHOD=FIML.

PROBCHI | PROBCHISQ *P*-value of the chi-square statistic for model fit.

PROBELLIPTIC *P*-value of the elliptical chi-square statistic.

PROBSBCHI | PROBSBCHISQ *P*-value of the Satorra-Bentler scaled chi-square statistic (Satorra and Bentler 1994) for model fit.

RMR Root mean square residual.

SBCHISQ Satorra-Bentler scaled chi-square statistic (Satorra and Bentler 1994) for model fit.

SRMR Standardized root mean square residual.

ZTEST Z-test of Wilson and Hilferty.

Output Control of Parsimonious Fit Indices

You can use the following *names* to refer to the individual parsimonious fit indices available in PROC CALIS:

AGFI Adjusted GFI.

AIC Akaike information criterion.

CAIC Bozdogan corrected AIC.

CENTRALITY McDonald centrality measure.

ECVI Expected cross-validation index.

ECVI_LL | LL_ECVI Lower confidence limit for ECVI.

ECVI_UL | UL_ECVI Upper confidence limit for ECVI.

PGFI Parsimonious GFI.

PROBCLFIT Probability of close fit.

RMSEA Root mean square error of approximation.

RMSEA_LL | LL_RMSEA Lower confidence limit for RMSEA.

RMSEA_UL | UL_RMSEA Upper confidence limit for RMSEA.

SBC Schwarz Bayesian criterion.

Output Control of Incremental Fit Indices

You can use the following *names* to refer to the individual incremental fit indices available in PROC CALIS:

BENTLERCFI | CFI Bentler comparative fit index.

BENTLERNFI Bentler-Bonett normed fit index.

BENTLERNNFI Bentler-Bonett nonnormed fit index.

BOLLENNFI Bollen normed fit index (Rho1).

BOLLENNNFI Bollen nonnormed fit index (Delta2).

PNFI James et al. parsimonious normed fit index.

OUTFIT=*SAS-data-set*
creates an output data set containing the values of the fit indices. This is the same as the OUTFIT= option of the PROC CALIS statement on page 1569. See the section "OUTFIT= Data Set" on page 1729 for details.

FREQ Statement

FREQ *variable* **;**

If one variable in your data set represents the frequency of occurrence for the other values in the observation, specify the variable's name in a FREQ statement. PROC CALIS then treats the data set as if each observation appears n_i times, where n_i is the value of the FREQ variable for observation i. Only the integer portion of the value is used. If the value of the FREQ variable is less than 1 or is missing, that observation is not included in the analysis. The total number of observations is considered to be the sum of the FREQ values. You can use only one FREQ statement within the scope of each GROUP or the PROC CALIS statement.

GROUP Statement

GROUP *i* < / *options* > **;**

where *i* is an assigned group number between 1 and 9999, inclusively.

The GROUP statement signifies the beginning of a group specification block and designates a group number for the group. All subsidiary group specification statements after a GROUP statement belong in that group until another MODEL or GROUP statement is used. The subsidiary group specification statements refer to one of the following four statements:

- FREQ statement on page 1614
- PARTIAL statement on page 1665
- VAR statement on page 1703
- WEIGHT statement on page 1711

For example, consider the following statements:

```
proc calis;
   var X1-X4;
   group 1 / label='Women' data=women_data;
      freq  Z;
   group 2 / label='Men' data=men_data;
      partial P;
   model 1 / group = 1-2;
      factor N=1; /* One factor exploratory factor analysis */
run;
```

In the GROUP statements, two groups are defined. Group 1, labeled as 'Women', refers to the data set women_data. Group 2, labeled as 'Men', refers to the data set men_data. Both groups are fitted by an exploratory factor model defined in Model 1, as indicated in the GROUP= option of the MODEL statement. While the frequency variable Z defined in the FREQ statement is applicable only to Group 1, the partial variable P defined in the PARTIAL statement is applicable only to Group 2. However, the VAR statement, which appears before the definitions of both groups, applies globally to both Group 1 and Group 2. Therefore, variables X1–X4 are the analysis variables in the two groups.

You can set group-specific *options* in each GROUP statement. All but one (that is, the LABEL= option) of these *options* are also available in the MODEL and PROC CALIS statements. If you set these group-specific *options* in the PROC CALIS statement, they will apply to all groups unless you respecify them in the GROUP statement. If you set these group-specific *options* in the MODEL statement, they will apply to all groups that are fitted by the associated model. In general, the group-specific options are transferred from the PROC CALIS statement to the MODEL statements (if present) and then to the fitted groups. In the transferring process, options are overwritten by the newer ones. If you want to apply some group-specific options to a particular group only, you should set those options in the GROUP statement corresponding to that group.

Option Available in the GROUP Statement Only

LABEL=*name*

NAME=*name*

specifies a label for the current group. You can use any valid SAS names up to 256 characters for labels. You can also use quote strings for the labels. This option can be specified only in the GROUP statement, not the PROC CALIS statement.

Options Available in the GROUP and PROC CALIS Statements

These *options* are available in the GROUP and PROC CALIS statements:

Option	Description
DATA= on page 1553	Specifies the input data set
INWGT= on page 1556	Specifies the data set that contains the weight matrix
OUTSTAT= on page 1569	Specifies the data set for storing the statistical results
OUTWGT= on page 1569	Specifies the data set for storing the weight matrix
ROBITER= on page 1575	Specifies the maximum number of iterations for estimating robust covariance and mean matrices
ROBPHI= on page 1575	Specifies the tuning parameter for robust methods

See the section "PROC CALIS Statement Options" on page 1542 for more details about these *options*. If you specify these *options* in the PROC CALIS statement, they are transferred to *all* GROUP statements. They might be overwritten by the respecifications in the individual GROUP statements.

Options Available in GROUP, MODEL, and PROC CALIS Statements

These *options* are available in the GROUP, MODEL, and PROC CALIS statements:

Option	Description
ALPHALEV= on page 1543	Specifies the α-level criterion for detecting leverage points
ALPHAOUT= on page 1543	Specifies the α-level criterion for detecting outliers
BIASKUR on page 1547	Computes the skewness and kurtosis without bias corrections
EDF= on page 1553	Defines nobs by the number of error *df*
INWGTINV on page 1557	Specifies that the INWGT= data set contains the inverse of the weight matrix
KURTOSIS on page 1557	Computes and displays kurtosis
MAXMISSPAT= on page 1559	Specifies the maximum number of missing patterns to display
NOBS= on page 1565	Defines the number of observations (nobs)
NOMISSPAT on page 1566	Suppresses the display of missing pattern analysis
PCORR on page 1570	Displays analyzed and estimated moment matrix
PLOTS= on page 1571	Specifies ODS Graphics selection
PWEIGHT on page 1711	Displays the weight matrix
RDF | DFR= on page 1574	Defines nobs by the number of regression *df*
RESIDUAL | RES on page 1574	Computes the default residuals
RESIDUAL | RES= on page 1574	Specifies the type of residuals
RIDGE on page 1575	Specifies the ridge factor for covariance matrix
SIMPLE on page 1577	Prints univariate statistics
TMISSPAT= on page 1578	Specifies the data proportion threshold for displaying the missing patterns
VARDEF= on page 1579	Specifies variance divisor
WPENALTY= on page 1580	Specifies the penalty weight to fit correlations
WRIDGE= on page 1580	Specifies the ridge factor for the weight matrix

If you specify these *options* in the PROC CALIS statement, they are transferred to all MODEL statements. These *options* are overwritten by the respecifications in the individual MODEL statements. After these *options* are resolved in a given MODEL statement, they are transferred further to the GROUP statements of which the associated groups are fitted by the model. Again, these *options* might be overwritten by the respecifications in the individual GROUP statements.

LINCON Statement

LINCON *constraint* < **,** *constraint* . . . > **;**

where *constraint* represents one of the following:

- *number operator linear-term*
- *linear-term operator number*

and *linear-term* is

< +|- > < *coefficient* * > *parameter* < < +|- > < *coefficient* * > *parameter* . . . >

The LINCON statement specifies a set of linear equality or inequality constraints of the following form:

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m$$

The constraints must be separated by commas. Each linear constraint i in the statement consists of a linear combination $\sum_j a_{ij} x_j$ of a subset of the n parameters x_j, $j = 1, \ldots, n$, and a constant value b_i separated by a comparison operator. Valid operators are <=, <, >=, >, and = or, equivalently, LE, LT, GE, GT, and EQ. PROC CALIS cannot enforce the strict inequalities < or >. Note that the coefficients a_{ij} in the linear combination must be constant numbers and must be followed by an asterisk and the name of a parameter (that is, listed in the PARMS, main, or subsidiary model specification statements). The following is an example of the LINCON statement that sets a linear constraint on parameters x1 and x2:

```
lincon       x1 + 3 * x2 <= 1;
```

Although you can express boundary constraints of individual parameters in LINCON statements, it is much more convenient to specify these boundary constraints with the BOUNDS statement. For example, the following LINCON statement essentially specifies boundary constraints on four parameters:

```
lincon       x1 <= 1,
             x2 <= 1,
             v1 > 0,
             v2 > 0;
```

Instead of using the LINCON statement, you can specify the following BOUNDS statement:

```
bounds       x1 x2 <= 1,
             v1 v2 >  0;
```

Another advantage of using the BOUNDS statement for setting bounds for individual parameters is that when a boundary constraint becomes active in the solution (that is, when the final estimate is on the boundary), a Lagrange multiplier test on releasing the boundary constraint would be conducted when you also specify the MOD option. However, no Lagrange multiplier test would be conducted if you specified the boundary constraint with the LINCON statement.

LINEQS Statement

LINEQS < *equation* <, *equation* . . . >> ;

where *equation* represents:

dependent = *term* < + *term* . . . >

and each *term* represents one of the following:

- *coefficient-name* < (*number*) >< * > *variable-name*
- *prefix-name* < (*number*) >< * > *variable-name*
- < *number* >< * > *variable-name*

The LINEQS statement is a main model specification statement that invokes the LINEQS modeling language. You can specify at most one LINEQS statement in a model, within the scope of either the PROC CALIS statement or a MODEL statement. To completely specify a LINEQS model, you might need to add some subsidiary model specification statements such as the VARIANCE, COV, and MEAN statements. The syntax for the LINEQS modeling language is as follows:

LINEQS < *equation* <, *equation* . . . >> ;
  VARIANCE *partial-variance-parameters* ;
  COV *covariance-parameters* ;
  MEAN *mean-parameters* ;

In the LINEQS statement, you use equations to specify the linear functional relations among manifest and latent variables. Equations in the LINEQS statement are separated by commas.

In the VARIANCE statement, you specify the variance parameters. In the COV statement, you specify the covariance parameters. In the MEAN statement, you specify the mean parameters. For details of these subsidiary model specification statements, see the syntax of these statements.

In the LINEQS statement, in addition to the functional relations among variables, you specify the coefficient parameters of interest in the equations. There are five types of parameters you can specify in equations, as shown in the following example:

```
lineqs
   V1 =          * F1 + E1,
   V2 = (.5)     * F1 + E2,
   V3 = 1.       * F1 + E3,
   V4 = b4       * F1 + E4;
   V5 = b5 (.4)  * F1 + E5;
```

In this example, you have manifest variables V1–V5, which are related to a latent factor, denoted by F1, as specified in the equations. In each equation, you have one outcome variable (V-variable), one predictor variable (F1, which is assumed to be a latent factor, the so-called F-variable), and one error variable (E-variable). The following four types of parameters have been specified:

- an unnamed free parameter

  The effect of F1 on V1 in the first equation is an unnamed free parameter. Although you specify nothing before the asterisk sign, the effect parameter is effectively specified. For an unnamed free

parameter, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1, _Parm2, and so on).

- an initial value

  The effect of F1 on V2 in the second equation is an unnamed free parameter with an initial estimate of 0.5. PROC CALIS also generates a parameter name for this specification. Notice that you must use a pair of parentheses for the initial value specification because it is interpreted as a fixed value otherwise, as described in the next case.

- a fixed value

  The effect of F1 on V3 in the third equation is an unnamed free parameter with a fixed value of 0.5. A fixed value remains the same in the estimation. There is no parameter name for a fixed constant in the model.

- a free parameter with a name

  The effect of F1 on V4 in the fourth equation is a free parameter named b4. You do not provide an initial estimate for this free parameter.

- a free parameter with a name and an initial estimate

  The effect of F1 on V5 in the fifth equation is a free parameter named b5 with an initial estimate of 0.4. Parameters with no starting values are initialized by various heuristic and effective methods in PROC CALIS. See the section "Initial Estimates" on page 1853 for details.

Notice that there must be an error term in each equation. The error terms in equation must start with the prefix 'E', 'e', 'D', or 'd'. See the section "Representing Latent Variables in the LINEQS Model" on page 1620 for details about naming the factors and error terms. The effect or the path coefficient attached to an error term must be 1.0. This is implicitly specified as in the preceding example. For example, there is no parameter specification nor an asterisk sign before the error term E1 in the first equation, as shown in the following:

```
V1 =          * F1 + E1,
```

This specification is the same as the following explicit specification with a fixed constant 1.0 for the effect of the error term E1:

```
V1 =          * F1 + 1. * E1,
```

The equivalence shown here implies that you can also specify the third equation in the following equivalent way:

```
V3 =  F1 + E3,
```

This implicitly specifies a constant 1.0 for the effect of F1 on V3. You must be very careful about the distinction between this specification and the following one with an asterisk before F1:

```
V3 =  * F1 + E3,
```

With an asterisk sign, the effect of F1 on V3 becomes an unnamed free parameter in the current specification. This interpretation is very different from the preceding one without an asterisk sign before F1, which assumes a fixed constant of 1.0.

Except for the unnamed free parameter specification, you can omit the asterisk signs in all other types of parameter specifications. That is, you can use the following equivalent statement for the preceding LINEQS specification:

```
lineqs
   V1 =          * F1 + E1,
   V2 = (.5)       F1 + E2,
   V3 = 1.         F1 + E3,
   V4 = b4         F1 + E4;
   V5 = b5 (.4)    F1 + E5;
```

Again, you cannot omit the asterisk in the first equation because it is intended to denote an unnamed free parameter.

If your model contains many unconstrained parameters and it is too cumbersome to find different parameter names, you can specify all those parameters by the same prefix-name. A prefix name is a short name called "root" followed by two underscores __. Whenever a prefix-name is encountered, the CALIS procedure generates a parameter name by appending a unique integer to the root. Hence, the prefix-name should have few characters so that the generated parameter name is not longer than thirty-two characters. To avoid unintentional equality constraints, the prefix names should not coincide with explicitly defined parameter names. The following statement illustrates the uses prefix-names:

```
lineqs
   V1 = 1.   * F1 + E1,
   V2 = b__  * F1 + E2,
   V3 = b__  * F1 + E3,
   V4 = b__  * F1 + E4;
   V5 = b__  * F1 + E5;
```

In the five equations, only the first equation has a fixed constant 1.0 for the effect of F1 on V1. For all the remaining equations, the effects of F1 on the variables are all free parameters with the prefix b. The generated parameter names for these effects have unique integers appended to this prefix. For example, b1, b2, b3, and b4 are the parameter names for these effects.

Representing Latent Variables in the LINEQS Model

Because latent variables are widely used in structural equation modeling, PROC CALIS needs a way to identify different types of latent variables that are specified in the LINEQS model. This is accomplished by following some naming conventions for the latent variables. See the section "Naming Variables in the LINEQS Model" on page 1747 for details about these naming rules. Essentially, latent factors (systematic sources) must start with the letter 'F' or 'f'. Error terms must start with the letter 'E', 'e', 'D', or 'd'. Prefix 'E' or 'e' represents the error term of an endogenous *manifest* variable. Prefix 'D' or 'd' represents the disturbance (or error) term of an endogenous *latent* variable. Although D- and E- variables are conceptually different, for modeling purposes 'D' and 'E' prefixes are interchangeable in the LINEQS modeling language. Essentially, only the distinction between latent factors (systematic sources) and errors or disturbances (unsystematic sources) is critical in specifying a proper LINEQS model. Manifest variables in the LINEQS model do not need to follow additional naming rules beyond those required by the general SAS System—they are recognized by PROC CALIS by referring to the variables in the input data sets.

Types of Variables and Semantic Rules of Equations

Depending on their roles in the system of equations, variables in a LINEQS model can be classified into endogenous or exogenous. An endogenous variable is a variable that serves as an outcome variable (left-hand side of an equation) in one of the equations. All other variables are exogenous variables, including those manifest variables that do not appear in any equations but are included in the model because they are specified in the VAR statement for the analysis.

Merely following the syntactic rules described so far is not sufficient to define a proper system of equations that PROC CALIS can analyze. You also need to observe the following semantic rules:

- Only manifest or latent variables can be endogenous. This means that you cannot specify any error or disturbances variables on the left-hand side of the equations. This also means that error and disturbance variables are always exogenous in the LINEQS model.
- An endogenous variable that appears on the left-hand side of an equation cannot appear on the left-hand side of another equation. In other words, you have to specify all the predictors for an endogenous variable in a single equation.
- An endogenous variable that appears on the left-hand side of an equation cannot appear on the right-hand side of the same equation. This prevents a variable to have a direct effect on itself (but indirect effect on itself is possible).
- Each equation must contain one and only one *unique* error term, be it an E-variable for a manifest outcome variable or a D-variable for a latent outcome variable. If, indeed, you want to specify an equation without an error term, you can equivalently set the variance of the error term to a fixed zero in the VARIANCE statement.

Mean Structures in Equations

To fit a LINEQS model with mean structures, you can specify the MEANSTR option in the PROC CALIS or the associated MODEL statement. This generates the default mean and intercept parameters for the model (see the section "Default Parameters" on page 1622). Alternatively, you can specify the intercept parameters with the Intercept variable in the equations or the mean parameters in the MEAN statement. The Intercept variable in the LINEQS model is a special "variable" that contains the value 1 for each observation. You do not need to have this variable in your input data set, nor do you need to generate it in the DATA step. It serves as a notational convenience in the LINEQS modeling language. The actual intercept parameter is expressed as a coefficient parameter with the intercept variable. For example, consider the following LINEQS model specification:

```
lineqs
   V1 = a1 (10) * Intercept +  1.0     * F1 + E1,
   V2 =         * Intercept +          * F1 + E2,
   V3 =                     +  b2      * F1 + E3,
   V4 = a2      * Intercept +  b2      * F1 + E4,
   V5 = a2      * Intercept +  b4 (.4) * F1 + E5;
```

In the first equation, a1, with a starting value at 10, is the intercept parameter of V1. In the second equation, the intercept parameter of V2 is an unnamed free parameter. In the third equation, although you do not specify the Intercept variable, the intercept parameter of manifest variable V3 is assumed to be a free parameter by

default. See the section "Default Parameters" on page 1622 for more details about default parameters. In the fourth and the fifth equations, the intercept parameters are both named a2. This means that these intercepts are constrained to be the same in the estimation.

In some cases, you might need to set the intercepts to fixed constants such as zeros. You can use the following syntax:

```
lineqs
   V1 = 0 * Intercept +  F_intercept + a2 * F_slope + E1;
```

This sets the intercept parameter of V1 to a fixed zero. An example of this application is the analysis of latent growth curve model in which you define the intercept as a random variable represented by a latent factor (for example, F_intercept in the specification). See Example 33.25 for a detailed example.

To complete the specification of the mean structures in the LINEQS model, you might want to use the MEAN statement to specify the mean parameters. For example, the following statements specify the means of F_intercept and F_slope as unnamed free parameters in the LINEQS model:

```
lineqs
   V1 = 0 * Intercept +  F_intercept + 1 * F_slope + E1;
mean
   F_intercept F_slope;
```

See the MEAN statement for details.

Default Parameters

It is important to understand the default parameters in the LINEQS model. First, if you know which parameters are default free parameters, you can make your specification more efficient by omitting the specifications of those parameters that can be set by default. For example, because all variances and covariances among exogenous variables (excluding error terms) are free parameters by default, you do not need to specify them with the COV and VARIANCE statements if these variances and covariances are not constrained. Second, if you know which parameters are default fixed zero parameters, you can specify your model accurately. For example, because all error covariances in the LINEQS model are fixed zeros by default, you must use the COV statement to specify the covariances among the errors if you want to fit a model with correlated errors. See the section "Default Parameters in the LINEQS Model" on page 1752 for details about the default parameters of the LINEQS model.

Modifying a LINEQS Model from a Reference Model

This section assumes that you use a REFMODEL statement within the scope of a MODEL statement and that the reference model (or base model) is a LINEQS model. The reference model is called the old model, and the model being defined is called the new model. If the new model is not intended to be an exact copy of the old model, you can use the extended LINEQS modeling language described in this section to make modifications within the scope of the MODEL statement for the new model.

The syntax of the extended LINEQS modeling language is the same as that of the ordinary LINEQS modeling language (see the section "LINEQS Statement" on page 1618):

LINEQS < *equation* < **,** *equation* . . . > > **;**
  VARIANCE *partial-variance-parameters* **;**
  COV *covariance-parameters* **;**
  MEAN *mean-parameters* **;**

The new model is formed by integrating with the old model in the following ways:

Duplication: If you do not specify in the new model an equation with an outcome variable (that is, a variable on the left side of the equal sign) that exists in the old model, the equation with that outcome variable in the old model is duplicated in the new model. For specifications other than the LINEQS statement, if you do not specify in the new model a parameter location that exists in the old model, the old parameter specification is duplicated in the new model.

Addition: If you specify in the new model an equation with an outcome variable that does not exist as an outcome variable in the equations of the old model, the equation is added in the new model. For specifications other than the LINEQS statement, if you specify in the new model a parameter location that does not exist in the old model, the new parameter specification is added in the new model.

Deletion: If you specify in the new model an equation with an outcome variable that also exists as an outcome variable in an equation of the old model and you specify the missing value '.' as the only term on the right-hand side of the equation in the new model, the equation with the same outcome variable in the old model is not copied into the new model. For specifications other than the LINEQS statement, if you specify in the new model a parameter location that also exists in the old model and the new parameter is denoted by the missing value '.', the old parameter specification is not copied into the new model.

Replacement: If you specify in the new model an equation with an outcome variable that also exists as an outcome variable in an equation of the model and the right-hand side of the equation in the new model is not denoted by the missing value '.', the new equation replaces the old equation with the same outcome variable in the new model. For specifications other than the LINEQS statement, if you specify in the new model a parameter location that also exists in the old model and the new parameter is not denoted by the missing value '.', the new parameter specification replaces the old one in the new model.

For example, the following two-group analysis specifies Model 2 by referring to Model 1 in the REFMODEL statement:

```
proc calis;
   group 1 / data=d1;
   group 2 / data=d2;
   model 1 / group=1;
      lineqs
         V1   =      1 * F1   + E1,
         V2   =  load1 * F1   + E2,
         V3   =  load2 * F1   + E3,
         F1   =     b1 * V4   + b2 * V5 + b3 * V6 + D1;
      variance
         E1-E3  = ve1-ve3,
         D1     = vd1,
         V4-V6  = phi4-phi6;
      cov
         E1 E2 = cve12;
   model 2 / group=2;
      refmodel 1;
```

```
      lineqs
         V3   = load1 * F1 + E3;
      cov
         E1 E2 = .,
         E2 E3 = cve23;
run;
```

Model 2 is the new model which refers to the old model, Model 1. This example illustrates the four types of model integration:

- Duplication: All equations, except the one with outcome variable V3, in the old model are duplicated in the new model. All specifications in the VARIANCE and COV statements, except the covariance between E1 and E2, in the old model are also duplicated in the new model.
- Addition: The parameter cve23 for the covariance between E2 and E3 is added in the new model.
- Deletion: The specification of covariance between E1 and E2 in the old model is not copied into the new model, as indicated by the missing value '.' specified in the new model.
- Replacement: The equation with V3 as the outcome variable in the old model is replaced with a new equation in the model. The new equation uses parameter load1 so that it is now constrained to be the same as the regression coefficient in the equation with V2 as the outcome variable.

LISMOD Statement

LISMOD < *var-lists* > **;**

where *var-lists* represent one or more of the following:

- **YVAR** | **YV** | **Y** = *var-list*
- **XVAR** | **XV** | **X** = *var-list*
- **ETAVAR** | **ETAV** | **ETA** = *var-list*
- **XIVAR** | **XIV** | **XI** | **KSIVAR** | **KSIV** | **KSI** = *var-list*

LISMOD stands for LISREL modeling, where LISREL is the program developed by Jöreskog and Sörbom (1988). Like the original implementation of LISREL, LISMOD uses a matrix specification interface. To complete the LISMOD specification, you might need to add as many MATRIX statements as needed, as shown in the following statement structure for the LISMOD model:

LISMOD *var-lists* **;**
  MATRIX *matrix-name parameters-in-matrix* **;**
  Repeat the MATRIX statement as needed **;**

The *matrix-name* in the MATRIX statement should be one of the twelve model matrices in LISMOD, as listed in the following:

- Matrices in the structural model: _ALPHA_, _KAPPA_, _BETA_, _GAMMA_, _PHI_, or _PSI_

- Matrices in the measurement model for **y**-variables: _NUY_, _LAMBDAY_, or _THETAY_
- Matrices in the measurement model for **x**-variables: _NUX_, _LAMBDAX_, or _THETAX_

See the section "Model Matrices in the LISMOD Model" on page 1754 for definitions of these matrices and their roles in the LISMOD modeling language. See the MATRIX statement on page 1638 for the details of parameter specification.

In the LISMOD statement, you can specify the following four lists of variables:

- **YVAR=** list is for manifest variables **y** that are directly related to the endogenous latent variables $\boldsymbol{\eta}$ (eta). Variables in the list are called **y**-variables.
- **XVAR=** list is for manifest variables **x** that are directly related to the exogenous latent variables $\boldsymbol{\xi}$ (xi or ksi). Variables in the list are called **x**-variables.
- **ETAVAR=** list is for endogenous latent variables $\boldsymbol{\eta}$. Variables in the list are called $\boldsymbol{\eta}$-variables.
- **XIVAR=** list is for exogenous latent variables $\boldsymbol{\xi}$. Variables in the list are called $\boldsymbol{\xi}$-variables.

The order of variables in the lists of the LISMOD statement is used to define the variable order in rows and columns of the LISMOD model matrices.

Depending on the model of interest, you might not need to specify all the lists of variables. When some variable lists are not specified, the full model reduces to specialized submodels. However, to be a proper submodel in the LISMOD modeling language, it is necessary (but not sufficient) that at least one of the YVAR= or XVAR= lists is defined. See the section "LISMOD Submodels" on page 1757 for the details about LISMOD submodels that PROC CALIS can handle.

An example of a LISMOD model specification is shown as follows:

```
proc calis;
   lismod xvar=x1-x3, yvar=y1-y6, xivar=xi, etavar=eta1-eta2;
   matrix _LAMBDAY_   [,1]  = 1. load3 load4,
                      [,2]  = 0. 0. 0. 1. load5 load6;
   matrix _THETAY_    [1,1] = ey1-ey3,
                      [2,1] = cey;
   matrix _LAMBDAX_   [,]   = 1. load1 load2;
   matrix _THETAX_    [1,1] = 3*ex;
   matrix _GAMMA_     [,1]  = beta1 beta2;
   matrix _PHI_       [1,1] = phi;
   matrix _PSI_       [1,1] = psi1-psi2;
run;
```

In this example, you have three **x**-variables x1–x3, six **y**-variables y1–y6, one $\boldsymbol{\xi}$-variable xi, and two $\boldsymbol{\eta}$-variables eta1–eta2. The numbers of variables in these lists define the dimensions of the LISMOD model matrices. For example, matrix _LAMBDAY_ is 6×2, with y1–y6 as the row variables and eta1–eta2 as the column variables. Matrix _THETAX_ is 3×3, with x1–x3 as the row and column variables. In the MATRIX statements, you specify parameters in the elements of the matrices. After the matrix name, you specify in square brackets '[' and ']' the starting row and column numbers of the first element to be parameterized. After the equal sign, you specify fixed or free parameters for the matrix elements.

Depending on how you specify the starting row and column numbers, the parameter specification might proceed differently. See the MATRIX statement on page 1638 for a detailed description. In this example, the first specification of the parameters in the _LAMBDAY_ matrix starts from [,1]—meaning that it starts from the first column and proceeds downwards. As a result, the [1,1] element is a fixed constant 1.0, the [2,1] element is a free parameter called load3, and the [3,1] element is a free parameter called load4. Similarly, in the second specification in the _LAMBDAY_ matrix, the [1,2], [2,2], [3,2], and [4,2] elements take constant values 0, 0, 0, and 1, respectively, and the [5,2] and [6,2] elements are free parameters load5 and load6, respectively.

You can also use similar notation to specify the parameters of a row. For example, with the notation [2,] for the starting row and column numbers, specification proceeds to the left with the same second row in the matrix.

If you have specified both starting row and column numbers, such as those in the first specification in matrix _THETAY_, the parameter specification starts from [1,1] and proceeds to the next row and column numbers—that is [2,2], [3,3], and so on. This results in specifying the diagonal elements of matrix _THETAY_ as free parameters ey1, ey2, and ey3.

With the notation [,], no starting row and column numbers are specified. Specification starts from the first valid element in the matrix and proceeds row-wise for all valid elements in the matrix. For example, in the matrix _LAMBDAX_ statement, the [1,1] element of matrix _LAMBDAX_ is a fixed constant 1, and the [1,2] and [1,3] elements are free parameters load1 and load2, respectively.

Default Parameters

It is important to understand the default parameters in the LISMOD model. First, if you know which parameters are default free parameters, you can make your specification more efficient by omitting some specifications. For example, because all variances and covariances among the exogenous $\boldsymbol{\xi}$-variables (excluding error terms) are free parameters by default, you do not need to specify them with MATRIX statement if these variances and covariances are not constrained. Second, if you know which parameters are default fixed zero parameters, you can specify your model accurately. For example, because all measurement errors in the LISMOD model are fixed zeros by default, you must use the MATRIX statement to specify the covariances among the errors in the $\boldsymbol{\Theta}_x$ (_THETAX_) or $\boldsymbol{\Theta}_y$ (_THETAY_) matrices if you want to fit a model with some correlated measurement errors. See the section "Default Parameters in the LISMOD Model" on page 1760 for details about the default parameters of the LISMOD model.

Modifying a LISMOD Model from a Reference Model

This section assumes that you use a REFMODEL statement within the scope of a MODEL statement and that the reference model (or base model) is also a LISMOD model. The reference model is called the old model, and the model that refers to this old model is called the new model. If the new model is not intended to be an exact copy of the old model, you can use the extended LISMOD modeling language described in this section to make modifications within the scope of the MODEL statement for the new model. The syntax is similar to, but not exactly the same as, the ordinary LISMOD modeling language (see the section "LISMOD Statement" on page 1624). The respecification syntax for a LISMOD model is shown as follows:

LISMOD ;
  MATRIX *matrix-name parameters-in-matrix* ;
  Repeat the MATRIX statement as needed ;

First, in the respecification you should not put any variable lists in the LISMOD statement. The reason is that the parameter respecifications in the new model refer to the variable lists of the old model. Therefore, the variable lists in the new model are implicitly assumed to be exactly the same as those in the old model. Because of this, the LISMOD statement is entirely optional for the respecification in the new model.

Second, you can use MATRIX *matrix-name statements* to modify the old model by using the same syntax as in the LISMOD modeling language. The *matrix-name* can be one of the twelve possible LISMOD matrices. In addition, in the respecification syntax you can use the missing value '.' to drop a parameter specification from the old model.

The new model is formed by integrating with the old model in the following ways:

Duplication: If you do not specify in the new model a parameter location that exists in the old model, the old parameter specification is duplicated in the new model.

Addition: If you specify in the new model a parameter location that does not exist in the old model, the new parameter specification is used in the new model.

Deletion: If you specify in the new model a parameter location that also exists in the old model and the new parameter is denoted by the missing value '.', the old parameter specification is not copied into the new model.

Replacement: If you specify in the new model a parameter location that also exists in the old model and the new parameter is not denoted by the missing value '.', the new parameter specification replaces the old one in the new model.

For example, the following two-group analysis specifies Model 2 by referring to Model 1 in the REFMODEL statement:

```
proc calis;
   group 1 / data=d1;
   group 2 / data=d2;
   model 1 / group=1;
      lismod xvar=X1-X3, yvar=Y1-Y6, xivar=xi, etavar=eta1-eta2;
      matrix _LAMBDAY_   [,1]  = 1. load3 load4,
                         [,2]  = 0. 0. 0. 1. load5 load6;
      matrix _THETAY_    [1,1] = ey1-ey3,
                         [2,1] = cey;
      matrix _LAMBDAX_   [,]   = 1. load1 load2;
      matrix _THETAX_    [1,1] = 3*ex;
      matrix _GAMMA_     [,1]  = beta1 beta2;
      matrix _PHI_       [1,1] = phi;
      matrix _PSI_       [1,1] = psi1-psi2;
   model 2 / group=2;
      refmodel 1;
      matrix _THETAY_    [2,1] = .;
      matrix _THETAX_    [1,1] = ex1-ex3;
      matrix _BETA_      [2,1] = beta;
run;
```

In this example, Model 2 is the new model which refers to the old model, Model 1. It illustrates the four types of model integration:

- Duplication: All parameter locations and specifications in the old model are duplicated in the new model, except for the [2,1] element in matrix _THETAY_ and the diagonal of matrix _THETAX_, which are modified in the new model.
- Addition: The _BETA_[2,1] parameter location is added with a new parameter beta in the new model. This indicates that *eta1* is a predictor variable of *eta2* in the new model, but not in the old model.
- Deletion: Because the missing value '.' is used for the parameter value, the _THETAY_[2,1] parameter location is no longer defined as a free parameter in the new model. In the old model, the same location is defined by the free parameter cey.
- Replacement: The diagonal elements of the _THETAX_ matrix in the new model are now defined by three distinct parameters ex1–ex3. This replaces the old specification where a single constrained parameter ex is applied to all the diagonal elements in the _THETAX_ matrix.

LMTESTS Statement

LMTESTS | LMTEST *option* < *option . . .* > **;**

where *option* represents one of the following:

- *display-option*
- *test-set*

and *test-set* represents one of the following:

- *set-name* = [*regions*]
- *set-name* = { *regions* }

where *set-name* is the name of the set of Lagrange multiplier (LM) tests defined by the *regions* that follow after the equal sign and *regions* are keywords denoting specific sets of parameters in the model.

You can use the LMTESTS statement to set *display-options* or to customize the *test-sets* for the LM tests. The LMTESTS statement is one of the model analysis statements. It can be used within the scope of the CALIS statement so that the options will apply to all models. It can also be used within the scope of each MODEL statement so that the options will apply only locally. Therefore, different models within a CALIS run can have very different LMTESTS *options*.

The LM Tests Display Options

The following are the *display-options* for the LM tests:

DEFAULT

conducts the default sets of LM tests for freeing fixed parameters in the model. This option is used when you need to reset the default sets of LM tests in the local model. For example, you might have turned off the default LM tests by using the NODEFAULT option in the LMTESTS statement within the scope of PROC CALIS statement. However, for the model under the scope of a particular MODEL statement, you can use this DEFAULT option in the local LMTESTS statement to turn on the default LM tests again.

MAXRANK=
sets the maximum number of rankings within a set of LM tests. The actual number of test rankings might be smaller because the number of possible LM tests within a set might be smaller than the maximum number requested.

NODEFAULT
turns off the default sets of LM tests for freeing fixed parameters in the model. As a result, only the customized LM tests defined in the *test-sets* of the LMTESTS statement are conducted and displayed. Note that the LM tests for equality and active boundary constraints are not turned off by this option. If you specify this option in the LMTESTS statement within the scope of the PROC CALIS statement, it will propagate to all models.

NORANK
turns off the ranking of the LM tests. Ranking of the LM tests is done automatically when the model modification indices are requested. The NORANK option is ignored if you also set the MAXRANK= option.

LMMAT
prints the sets of LM tests in matrix form, in addition to the normal LM test results.

The Customized Sets of LM Tests: Syntax of the Test-sets

In addition to the *display-options*, you can define customized sets of LM tests as *test-sets* in the LMTESTS statement. You can define as many *test-sets* as you like. Ranking of the LM tests will be done individually for each *test-set*. For example, the following LMTESTS statement requests that the default sets of LM tests not be conducted by the NODEFAULT option. Instead, two customized *test-sets* are defined.

```
lmtests nodefault MyFirstSet=[ALL] MySecondSet=[COVEXOG COVERR];
```

The first customized set MyFirstSet pulls all possible parameter locations together for the LM test ranking (ALL keyword). The second customized set MySecondSet pulls only the covariances among exogenous variables (COVEXOG keyword) and among errors (COVERR keyword) together for the LM test ranking.

Two different kinds of *regions* for LM tests are supported in PROC CALIS: matrix-based or non-matrix-based.

The matrix-based *regions* can be used if you are familiar with the matrix representations of various types of models. Note that defining *test-sets* by using matrix-based *regions* does not mean that LM tests are printed in matrix format. It means only that the parameter locations within the specified matrices are included into the specific *test-sets* for LM test ranking. For matrix output of LM tests, use the LMMAT option in the LMTESTS statement.

Non-matrix-based *regions* do not assume the knowledge of the model matrices. They are easier to use in most situations. In addition, non-matrix-based *regions* can cover special subsets of parameter locations that cannot be defined by model matrices and submatrices. For example, because of the compartmentalization according to independent and dependent variables in the LINEQS model matrices, the sets of LM tests defined by the LINEQS matrix-based *regions* are limited. For example, you cannot use any matrix-based *regions* to request LM tests for new paths to existing independent variables in the LINEQS model. Such a matrix does not exist in the original specification. However, you can use the non-matrix based *region* NEWENDO to refer to these new paths.

The *regions* for parameter locations are specified by *keywords* in the LMTESTS statement. Because the *regions* are specific to the types of models, they are described separately for each model type in the following.

The LM Test Regions for COSAN Models

ALLMAT
specifies all parameter locations in all matrices.

CENTRAL
specifies all parameter locations in the central covariance matrices in all terms.

MATRIX= [*set-of-matrices***] | {***set-of-matrices***}**

MAT= [*set-of-matrices***] | {***set-of-matrices***}**

MATSET= [*set-of-matrices***] | {***set-of-matrices***}**
specifies the parameter locations in the matrices specified in *set-of-matrices*.

MEANVEC
specifies all parameter locations in the central mean vectors in all terms.

OUTER
specifies all parameter locations in all matrices except for the central covariance matrices and central mean vectors in all terms.

The LM Test Regions for FACTOR Models

The *keywords* for the matrix-based regions are associated with the FACTOR model matrices. See the section "Summary of Matrices in the FACTOR Model" on page 1743 for the definitions and properties of these matrices.

Keywords for Matrix-Based Regions

FACTERRV | FACTERRV
specifies the error variances.

FACTFCOV | FACTFCOV
specifies the covariances among factors.

FACTINTE | FACTINTE
specifies the intercepts.

FACTLOAD | FACTLOAD
specifies the factor loadings.

FACTMEAN | FACTMEAN
specifies the factor means.

Keywords for Non-Matrix-Based Regions

ALL
specifies all parameter locations.

COV
specifies the covariances among factors.

COVERR
specifies the covariances among errors.

COVFACT | COVLV
specifies the covariances among factors.

FIRSTMOMENTS
specifies the means of factors and the intercepts.

INTERCEPTS
specifies the intercepts.

LOADINGS
specifies the factor loadings.

MEANS | MEAN
specifies the means of factors.

The LM Test Regions for LINEQS Models

Keywords for Matrix-Based Regions

The *keywords* for the matrix-based regions are associated with the LINEQS model matrices. See the section "Matrix Representation of the LINEQS Model" on page 1747 for definitions of these model matrices and see the section "Summary of Matrices and Submatrices in the LINEQS Model" on page 1750 for the names and properties and the model matrices and submatrices.

EQSALPHA | EQSALPHA
specifies the intercepts of dependent variables.

EQSBETA | EQSBETA
specifies effects of dependent variables on dependent variables.

EQSGAMMA | _EQSGAMMA_SUB_ | EQSGAMMA | EQSGAMMASUB
specifies the effects of independent variables (excluding errors) on dependent variables. Because effects of errors on dependent variables are restricted to ones in the LINEQS model, LM tests on _EQSGAMMA_ and _EQSGAMMA_SUB_ (submatrix of _EQSGAMMA_) are the same.

EQSNU | _EQSNU_SUB_ | EQSNU | EQSNUSUB
specifies the means of independent variables (excluding errors). Because means of errors are restricted to zero in the LINEQS model, LM tests on _EQSNU_ and _EQSNU_SUB_ (submatrix of _EQSNU_) are the same.

EQSPHI | EQSPHI
specifies variances and covariances among all independent variables, including errors.

EQSPHI11 | EQSPHI11
specifies variances and covariances among independent variables, excluding errors.

EQSPHI21 | EQSPHI21
specifies covariances between errors and disturbances with other independent variables.

EQSPHI22 | EQSPHI22
specifies variances and covariances among errors and disturbances.

Keywords for Non-Matrix-Based Regions

ALL
specifies all possible parameter locations.

COV
specifies all covariances among independent variables, including errors and disturbances.

COVERR
specifies covariances among errors or disturbances.

COVEXOG
specifies covariances among independent variables, excluding errors and disturbances.

COVEXOGERR
specifies covariances of errors and disturbances with other independent variables.

COVLV | COVFACT
specifies covariances among latent variables (excluding errors and disturbances).

COVMV | COVOV
specifies covariance among independent manifest variables.

EQUATION | EQUATIONS
specifies all possible linear relationships among variables.

FIRSTMOMENTS
specifies means and intercepts.

INTERCEPTS | INTERCEPT
specifies intercepts of dependent variables.

LV==->LV | LV–>LV
specifies all possible effects of latent factors on latent factors.

LV==>MV | LV–>MV | MV<==LV | MV<–LV
specifies all possible effects of latent factors on manifest variables.

LV<==MV | LV<–MV | MV==>LV | MV–>LV
specifies all possible effects of manifest variables on latent factors.

MEANS | MEAN
specifies the means of independent factors.

MV==>MV | MV–>MV
specifies all possible effects of manifest variables on manifest variables.

NEWDEP | NEWENDO
specifies effects of other variables on the independent variables in the original model.

PATHS | PATH
specifies all possible linear relationships among variables.

The LM Test Regions for LISMOD Models

The *keywords* for the matrix-based regions are associated with the LISMOD model matrices. See the section "Model Matrices in the LISMOD Model" on page 1754 for the definitions and properties of these matrices.

Keywords for Matrix-Based Regions

ALPHA | ALPHA
specifies the _ALPHA_ matrix.

BETA | BETA
specifies the _BETA_ matrix.

GAMMA | GAMMA
specifies the _GAMMA_ matrix.

KAPPA | KAPPA
specifies the _KAPPA_ matrix.

LAMBDA | LAMBDA
specifies the _LAMBDAX_ and _LAMBDAY_ matrices.

LAMBDAX | LAMBDAX
specifies the _LAMBDAX_ matrix.

LAMBDAY | LAMBDAY
specifies the _LAMBDAY_ matrix.

_NU_ | NU
specifies the _NUX_ and _NUY_ matrices.

NUX | NUX
specifies the _NUX_ matrix.

NUY | NUY
specifies the _NUY_ matrix.

PHI | PHI
specifies the _PHI_ matrix.

PSI | PSI
specifies the _PSI_ matrix.

THETA | THETA
specifies the _THETAX_ and _THETAY_ matrices.

THETAX | THETAX
specifies the _THETAX_ matrix.

THETAY | THETAY
specifies the _THETAY_ matrix.

Keywords for Non-Matrix-Based Regions

ALL
specifies all model matrices.

COV
specifies all covariance parameters in _THETAY_, _THETAX_, _PHI_, and _PSI_.

COVERR
specifies all covariances for errors or disturbances in _THETAY_, _THETAX_, and _PSI_.

COVFACT | COVLV
specifies all covariances among latent factors in _PHI_ when the ξ-variables exist, and in _PSI_ when the η-variables exist without the presence of the ξ-variables.

FIRSTMOMENTS
specifies all intercepts and means in _NUY_, _NUX_, _ALPHA_, and _KAPPA_.

INTERCEPTS | INTERCEPT
specifies all intercepts in _NUY_, _NUX_, and _ALPHA_.

LOADING | LOADINGS
specifies the coefficients in _LAMBDAY_ and _LAMBDAX_.

LV==>LV | LV–>LV
specifies the effects of latent variables on latent variables. Depending on the type of LISMOD model, the _BETA_ and _GAMMA_ might be involved.

LV==>MV | LV–>MV | MV<==LV | MV<–LV
specifies the effects of latent variables on manifest variables. Depending on the type of LISMOD model, the _LAMBDAY_, _LAMBDAX_, and _GAMMA_ matrices might be involved.

MEANS | MEAN
specifies the mean parameters. Depending on the type of LISMOD model, the _ALPHA_ and _KAPPA_ matrices might be involved.

MV==>MV | MV–>MV
specifies effects of manifest variables on manifest variables. Depending on the type of LISMOD model, the _BETA_ and _GAMMA_ matrices might be involved.

PATHS | PATH
specifies all path coefficients. Depending on the type of LISMOD model, the _LAMBDAY_, _LAMBDAX_, _BETA_, and _GAMMA_ matrices might be involved.

The LM Test Regions for MSTRUCT Models

The *keywords* for the matrix-based regions are associated with the MSTRUCT model matrices. See the section "Model Matrices in the MSTRUCT Model" on page 1761 for the definitions and properties of these matrices.

Keywords for Matrix-Based Regions

MSTRUCTCOV | _COV_ | MSTRUCTCOV
specifies the _MSTRUCTCOV_ or _COV_ matrix.

MSTRUCTMEAN | _MEAN_ | MSTRUCTMEAN
specifies the _MSTRUCTMEAN_ or _MEAN_ vector.

Keywords for Non-Matrix-Based Regions

ALL
specifies the _MSTRUCTCOV_ (or _COV_) and _MSTRUCTMEAN_ (or _MEAN_) matrices.

COV
specifies the _MSTRUCTCOV_ or _COV_ matrix.

MEANS | MEAN
specifies the _MSTRUCTMEAN_ or _MEAN_ matrix.

The LM Test Regions for PATH and RAM Models

The *keywords* for the matrix-based regions are associated with the submatrices of the RAM model matrices. See the section "Partitions of the RAM Model Matrices and Some Restrictions" on page 1771 for the definitions of these submatrices and the section "Summary of Matrices and Submatrices in the RAM Model" on page 1773 for the summary of the names and properties of these submatrices.

Keywords for Matrix-Based Regions

RAMA | _A_ | RAMA
specifies the _RAMA_ matrix.

_RAMA_LEFT_ | _A_LEFT_ | RAMALEFT
specifies the left portion of the _RAMA_ matrix.

_RAMA_LL_ | _A_LL_ | RAMALL
specifies the lower left portion of the _RAMA_ matrix.

_RAMA_LR_ | _A_LR_ | RAMALR
specifies the lower right portion of the _RAMA_ matrix.

_RAMA_LOWER_ | _A_LOWER_ | RAMALOWER
specifies the lower portion of the _RAMA_ matrix. This is equivalent to the region specified by the NEWENDO keyword.

_RAMA_RIGHT_ | _A_RIGHT_ | RAMARIGHT
specifies the right portion of the _RAMA_ matrix.

_RAMA_UPPER_ | _A_UPPER_ | RAMAUPPER
specifies the upper portion of the _RAMA_ matrix.

RAMALPHA | RAMALPHA
specifies the _RAMALPHA_ matrix.

RAMBETA | RAMBETA
specifies the _RAMBETA_ matrix.

RAMGAMMA | RAMGAMMA
specifies the _RAMGAMMA_ matrix.

RAMNU | RAMNU
specifies the _RAMNU_ matrix.

RAMP | _P_ | RAMP
specifies the _RAMP_ matrix.

RAMP11 | RAMP11
specifies the _RAMP11_ matrix.

RAMP21 | RAMP21
specifies the _RAMP21_ matrix.

RAMP22 | RAMP22
specifies the _RAMP22_ matrix.

RAMW | _W_ | RAMW
specifies the _RAMW_ vector.

Keywords for Non-Matrix-Based Regions

ALL
specifies all possible parameter locations.

ARROWS | ARROW
specifies all possible paths (that is, the entries in the _RAMA_ matrix).

COV
specifies all covariances and partial covariances (that is, the entries in the _RAMP_ matrix).

COVERR
specifies partial covariances among endogenous variables (that is, the entries in the _RAMP11_ matrix).

COVEXOG
specifies covariances among exogenous variables (that is, the entries in the _RAMP22_ matrix).

COVEXOGERR
specifies partial covariances of endogenous variables with exogenous variables (that is, the entries in the _RAM21_ matrix).

COVLV | COVFACT
specifies covariance among latent factors (that is, entries in _RAMP11_ pertaining to latent variables).

COVMV | COVOV
specifies covariance among manifest variables (that is, entries in _RAMP11_ pertaining to manifest variables).

FIRSTMOMENTS
specifies means or intercepts (that is, entries in _RAMW_ vector).

INTERCEPTS | INTERCEPT
specifies intercepts for endogenous variables (that is, entries in _RAMALPHA_ vector).

LV==>LV | LV–>LV
specifies effects of latent variables on latent variables.

LV==>MV | LV–>MV | MV<==LV | MV<–LV
specifies effects of latent variables on manifest variables.

LV<==MV | LV<–MV | MV==>LV | MV–>LV
specifies effects of manifest variables on latent variables.

MEANS | MEAN
specifies the means of exogenous variables (that is, entries in the _RAMNU_ vector).

MV==>MV | MV–>MV
specifies effects of manifest variables on manifest variables.

NEWENDO
specifies new paths to the exogenous variables in the original model.

PATHS | PATH
specifies all possible paths (that is, the entries in the _RAMA_ matrix).

MATRIX Statement

MATRIX *matrix-name* < *location* < **=** *parameter-spec* > < **,** *location* < **=** *parameter-spec* . . . > > > **;**

MATRIX statement specifies the matrix elements (locations) and their parameters. Parameters can be fixed or free, with or without initial estimates. The *matrix-name* indicates the matrix to specify in the MATRIX statement. The *location* indicates the starting row and column numbers of the matrix being specified and the *parameter-spec* is a list of free or fixed parameters for the elements that are indicated by the *location*.

The MATRIX statement is a subsidiary model specification statement of the COSAN, LISMOD, and MSTRUCT modeling languages. You might need to use the MATRIX statements as many times as needed for specifying your model. However, you can use the MATRIX statement at most once for each distinct model matrix.

Valid Matrix Names for the COSAN Model

The valid *matrix-names* depend on the your specification in the COSAN statement in which you define the COSAN model matrices and their properties. Except for those fixed matrices with the IDE or ZID type, you can use the MATRIX statement to specify any COSAN model matrices you define in the COSAN statement.

Valid Matrix Names for the LISMOD Model

There are 12 model matrices in the LISMOD model, and they correspond to the following valid *matrix-names*:

- matrices and their types in the measurement model for the $\mathbf{y}$-variables

  LAMBDAY the matrix of regression coefficients of the $\mathbf{y}$-variables on the $\boldsymbol{\eta}$-variables (general, GEN)

  NUY the vector of intercept terms of the $\mathbf{y}$-variables (general, GEN)

  THETAY the error covariance matrix for the $\mathbf{y}$-variables (symmetric, SYM)

- matrices and their types in the measurement model for the $\mathbf{x}$-variables

  LAMBDAX the matrix of regression coefficients of the $\mathbf{x}$-variables on the $\boldsymbol{\xi}$-variables (general, GEN)

  NUX the vector of intercept terms of the $\mathbf{x}$-variables (general, GEN)

  THETAX the error covariance matrix for the $\mathbf{x}$-variables (symmetric, SYM)

- matrices and their types in the structural model

  ALPHA the vector of intercept terms of the $\boldsymbol{\eta}$-variables (general, GEN)

  BETA the matrix of regression coefficients of the $\boldsymbol{\eta}$-variables on the $\boldsymbol{\eta}$-variables (general, GEN)

  GAMMA the matrix of regression coefficients of the $\boldsymbol{\eta}$-variables on the $\boldsymbol{\xi}$-variables (general, GEN)

  KAPPA the mean vector for the $\boldsymbol{\xi}$-variables (general, GEN)

  PHI the covariance matrix for the $\boldsymbol{\xi}$-variables (symmetric, SYM)

PSI the error covariance matrix for the $\boldsymbol{\eta}$-variables (symmetric, SYM)

Valid Matrix Names for the MSTRUCT Modeling Language

The following *matrix-names* are valid for the MSTRUCT modeling language:

COV the covariance matrix (symmetric, SYM)

MEAN the mean vector (general, GEN)

Specifying Locations in Model Matrices

The five main types of matrix *locations* (elements) specification in the MATRIX statement are briefly described in the following:

- Unspecified *location*: Blank or `[ , ]`
  Use this notation to specify the `[1,1]` element of the matrix, and to specify the remaining *valid* elements of the matrix in a prescribed order until all the parameters in the *parameter-spec* list are assigned.

- Row-and-column *location*: `[i,j]`, `[@i,j]`, `[i,@j]`, or `[@i,@j]`
  Use this notation to specify the `[i,j]` element of a matrix, and to specify the remaining elements of the matrix in the order indicated by the *location* notation until all the parameters in the *parameter-spec* list are assigned.

- Row *location* only: `[i,]`, `[@i,]`, or `[`*iset*`,]`
  Use this notation to specify the first *valid* matrix element in the `[i]`th row (for the first two notations) or the `[i1]`th row (for the `[`*iset*`,]` notation, where *iset*=(`i1`, `i2`, ...) is a set of row numbers), and to specify the remaining elements of the matrix in the order indicated by the *location* notation until all the parameters in the *parameter-spec* list are assigned.

- Column *location* only: `[,j]`, `[,@j]`, or `[,`*jset*`]`
  Use this notation to specify the first *valid* matrix element in the `[j]`th column (for the first two notations) or the `[j1]`th column (for the `[,`*jset*`]` notation, where *jset*=(`j1`, `j2`, ...) is a set of columns), and to specify the remaining elements of the matrix in the order indicated by the *location* notation until all the parameters in the *parameter-spec* list are assigned.

- Row-and-column-sets *location*: `[`*iset*`,`*jset*`]`, `[`*iset*`,j]`, or `[i,`*jset*`]`
  Use this notation to specify the `[i1,j1]` element of the matrix, where `i1` is either the same as `i` or the first row number specified in *iset*, and `j1` is either the same as `j` or the first column number specified in *jset*, and to specify the remaining elements of the matrix in the order indicated by the *location* notation until all the parameters in the *parameter-spec* list are assigned.

Consider the following points about the *location* specifications:

- In the description of the various *location* specifications, the starting matrix element for parameter assignment is relatively well-defined. However, if the *parameter-spec* list has more than one parameter, there are more matrix elements to assign with the parameters in the *parameter-spec* list. If there is

no *parameter-spec* list, a set of matrix elements are specified as unnamed free parameters. Hence, the actual number of elements specified by these *location* specifications depends on the length of the *parameter-spec* list.

- Because more than one matrix element could be specified in any of these *location* specifications, it is important to understand the order that PROC CALIS uses to assign the matrix elements.
- In some of the *location* specifications, either the row or column is unspecified and the assignment of the matrix element starts with the first *valid* element given the column or the row number. This first valid element depends on the type of the matrix in question.

The next few sections describe the parameter assignments in more detail for each of these *location* specifications in the MATRIX statement.

Unspecified Location: Blank or `[ , ]`

This notation means that all valid elements started with the `[1,1]` element of the matrix specified in the model. If no *parameter-spec* list is specified, all valid elements in the matrix are unnamed free parameters. For these elements, PROC CALIS generates parameter names with the _Parm prefix followed by a unique integer (for example, _Parm1, _Parm2, and so on). If a *parameter-spec* list is specified, the assignment of parameters starts with the `[1,1]` element and proceeds to the next *valid* elements in the same row. If the entire row of valid elements is assigned with parameters, it proceeds to the next row and so on, until all the parameters in the *parameter-spec* list are assigned. The valid element given the row or column number depends on the type of matrix in question. The following examples illustrate the usage of the unspecified *location* notation.

Suppose that _GAMMA_ is a general 3×3 matrix. The following statement specifies four elements of this matrix:

```
matrix _GAMMA_ [,] = gg1-gg4;
```

Equivalently, you can use the following blank *location* specification:

```
matrix _GAMMA_ = gg1-gg4;
```

Both specifications are equivalent to the following elementwise specification:

```
matrix _GAMMA_ [1,1] = gg1,
               [1,2] = gg2,
               [1,3] = gg3,
               [2,1] = gg4;
```

With the unspecified *location* for the matrix _GAMMA_, the first row is filled up with the parameters first. Then it proceeds to the next row and so on until all parameters in the *parameter-spec* list are assigned. Because there are four parameters and _GAMMA_ has three columns, the parameter gg4 is assigned to the _GAMMA_[2, 1] element.

However, if the preceding specification is for a 3×3 matrix symmetric matrix _PHI_, the parameters are assigned differently. That is, the following specification has different matrix elements assigned with the parameters:

```
matrix _PHI_ = gg1-gg4;
```

Because symmetric matrices contain redundant elements, parameters are assigned only to the lower triangular elements (including the diagonal elements). As a result, the following elementwise specification reflects the preceding specification of matrix _PHI_:

```
matrix _PHI_ [1,1] = gg1,
             [2,1] = gg2,
             [2,2] = gg3,
             [3,1] = gg4;
```

The case for lower triangular matrices is the same as the case for symmetric matrices. That is, only the lower triangular elements are valid elements for the parameter assignments.

For upper triangular matrices, only the upper triangular elements (including the diagonal elements) are valid for the parameter assignments. For example, consider the following specification of a 3×3 upper triangular matrix UPP:

```
matrix UPP = gg1-gg4;
```

The matrix elements assigned with the parameters are the same as the following elementwise specification:

```
matrix UPP [1,1] = gg1,
           [1,2] = gg2,
           [1,3] = gg3,
           [2,2] = gg4;
```

If a 4×4 diagonal matrix is specified by the preceding MATRIX statement, the parameters are assigned to the following elements: `[1,1]`, `[2,2]`, `[3,3]`, and `[4,4]`.

Lastly, if there is no *parameter-spec* list for the unspecified *location* notation, all valid parameters in the matrix being specified are unnamed free parameters. For example, if $\mathbf{A}$ is a 4×4 general rectangular matrix, the following specification assigns 16 unnamed free parameters to all of the elements in $\mathbf{A}$:

```
matrix A [,];
```

PROC CALIS generates parameters _Parm1, _Parm2, . . . , _Parm16 to the elements `[1,1]`, `[1,2]`, `[1,3]`, . . . , `[4,3]`, `[4,4]`, respectively.

However, if $\mathbf{S}$ is a 4×4 *symmetric* matrix, the following specification assigns only 10 unnamed free parameters to the lower triangular elements of $\mathbf{S}$:

```
matrix S;
```

PROC CALIS generates parameters _Parm1, _Parm2, . . . , _Parm10 to the elements `[1,1]`, `[2,1]`, `[2,2]`, . . . , `[4,3]`, `[4,4]`, respectively.

Row-and-Column Location: `[i,j]`, `[@i,j]`, `[i,@j]`, *or* `[@i,@j]`

All these notations provide the starting row (`i`) and column (`j`) numbers for the assignment of the parameters in the *parameter-spec* list. The notations are different in the way they proceed to the next element in the matrix. If no *parameter-spec* list is specified, only the single element `[i,j]` is an unnamed free parameter. For this `[i,j]` element, PROC CALIS generates a parameter name with the _Parm prefix followed by a unique integer (for example, _Parm1). If a *parameter-spec* list is specified, the assignment of parameters starts with the `[i,j]` element and proceeds to next element until all the parameters in the *parameter-spec*

list are assigned. The following summarizes how the assignment of parameter proceeds, depending on the uses of the `@` sign before the starting row or column number:

- `[i,j]` specifies the `[i,j]` element, and proceeds to `[i+1,j+1]`, `[i+2,j+2]`, and so on.
- `[@i,j]` specifies the `[i,j]` element, and proceeds to `[i,j+1]`, `[i,j+2]`, `[i,j+3]`, and so on.
- `[i,@j]` specifies the `[i,j]` element, and proceeds to `[i+1,j]`, `[i+2,j]`, `[i+3,j]`, and so on.
- `[@i,@j]` specifies the `[i,j]` element only.

The following examples illustrates the usage of the row-and-column *location* notation.

The simplest case is the specification of a single element as an unnamed free parameter. For example, the following statement specifies that `[1,4]` in matrix $\mathbf{A}$ is an unnamed free parameter:

```
matrix A [1,4];
```

PROC CALIS generates a parameter name with the _Parm prefix for this element. In this case, using the `@` sign before the row or column number is optional. That is, the following statements are all the same specification:

```
matrix A [1,4];
matrix A [@1,4];
matrix A [1,@4];
matrix A [@1,@4];
```

You can specify more than one unnamed free parameter by using multiple *location* specifications, as shown in the following example:

```
matrix A [1,4],[3,5];
```

Elements `[1,4]` and `[3,5]` of matrix $\mathbf{A}$ are both unnamed free parameters. However, when a *parameter-spec* list is specified after the *location*, more than one parameters might be specified. The use of the `@` determines how the elements in the matrix are assigned with the parameters in the *parameter-spec* list. The following examples illustrate this under various situations.

For example, consider the following specification of a 4×4 matrix general matrix $\mathbf{A}$:

```
matrix A
   [1,1] = a b c;
```

The three parameters a, b, and c, are assigned to the matrix elements `[1,1]`, `[2,2]`, and `[3,3]`, respectively. That is, this specification is equivalent to the following elementwise specification:

```
matrix A
   [1,1] = a ,
   [2,2] = b ,
   [3,3] = c ;
```

However, with the `@` sign, the assignment is different. For example, consider the `@` sign attached to the row number in the following specification:

```
matrix A
   [@1,1] = a b c;
```

The `@` sign fixes the row number to 1. As a result, this specification is equivalent to the following elementwise specification:

```
matrix A
   [1,1] = a ,
   [1,2] = b ,
   [1,3] = c ;
```

Using the `@` sign before the column number fixes the column number. For example, consider the following specification of matrix `A`:

```
matrix A
   [2,@2] = a b c;
```

The `@` sign fixes the column number to 2. As a result, this specification is equivalent to the following elementwise specification:

```
matrix A
   [2,2] = a ,
   [3,2] = b ,
   [4,2] = c ;
```

If you put the `@` sign in both of the row and column numbers, only one element is intended to be assigned. For example, the following specification means that only $\mathbf{A}[2, 3]$ is assigned with the parameter a:

```
matrix A
   [@2,@3] = a;
```

But you could specify this simply as the statement without the `@` sign:

```
matrix A
   [2,3] = a;
```

Notice that the matrix type does not play a role in determining the elements for the parameter assignments in the row-and-column *location* specification. You have to make sure that the parameters are assigned in the valid elements of the matrix. For example, suppose that S is a 4×4 symmetric matrix and you specify the following statement for its elements:

```
matrix A
   [@3,2] = a b c;
```

The elements to be assigned with the parameters a, b, and c, are `[3,2]`, `[3,3]`, and `[3,4]`, respectively. However, because S is symmetric, you can specify only the nonredundant elements in the lower triangular of S. Hence, the specification of the `[3,4]` element is not valid and it generates an error.

Row Location Only: `[i,]`, `[@i,]`, *or* `[`*iset*`,]`

All these notations provide the starting row `[i1,]` for the assignment of the parameters in *parameter-spec*, where `i1` is `i` for the first two *location* notations or `i1` is the first row specified in *iset*, where *iset* = (`i1`, `i2`, . . .) is a set of row numbers. Because no column location is specified, the starting element is the first valid element in the `i1`th row of the matrix.

If no *parameter-spec* list is specified, all the valid elements in the entire **i1**th row of the matrix are unnamed free parameters. If a set of row numbers is specified in *iset*, all the valid elements in the all the rows specified in *iset* are unnamed free parameters.

If a *parameter-spec* list is specified, the assignment of parameters starts with the first valid elements of the **i1**th row. The assignment proceeds to next valid elements in the same row. The **[i,]** specification proceeds row by row for parameter assignment while the **[@i,]** specification stays at the same **i**th row. The **[***iset***,]** specification indicates and limits the sequence of rows to be assigned with the parameter in the *parameter-spec* list. The assignment stops when all the parameters in the *parameter-spec* list are assigned. The following summarizes how the assignment of parameters proceeds in more precise terms:

- **[i,]** specifies the first valid element in row **i** and proceeds to the valid elements in rows **i**, **i+1**, **i+2**, . . . , until all parameters in the *parameter-spec* list are assigned.
- **[@i,]** specifies the first valid elements in row **i** and proceeds to the valid elements in the *same* row until all parameters in the *parameter-spec* list are assigned.
- **[***iset***,]** specifies the first valid elements in row **i1**, where **i1**, **i2**, . . . are the rows specified in *iset*. It proceeds to the valid elements in rows **i1**, **i2**, . . . , until all parameters in the *parameter-spec* list are assigned.

The following examples illustrates the usage of the row *locations*.

The simplest case is the specification of all valid elements of a single row as unnamed free parameters. For example, the following specification of a 3×3 rectangular matrix **A** assigns unnamed free parameters to all the elements in the second row of matrix **A**:

```
matrix A [2,];
```

PROC CALIS generates parameter names with the _Parm prefix for these elements. For example, the **[2,1]**, **[2,2]**, and **[2,3]** elements are named with _Parm1, _Parm2, and _Parm3, respectively.

Using the **@** sign before the row number in this case is optional. That is, the following statement is the same specification:

```
matrix A [@2,];
```

If you specify a set of row numbers without the *parameter-spec* list, all valid elements of the specified rows are unnamed free parameters. For example, consider the following specification of a 6×6 symmetric matrix **S**:

```
matrix S [1 3 5,];
```

This specification specifies unnamed free parameters for the lower triangular elements in the first, third, and fifth rows of matrix **S**. It is equivalent to the following specification:

```
matrix S [1,],
         [3,],
         [5,];
```

As a result, this means that the following elements in matrix **S** are free parameters: **[1,1]**, **[3,1]**, **[3,2]**, **[3,3]**, **[5,1]**, **[5,2]**, **[5,3]**, **[5,4]**, and **[5,5]**. Notice that only the elements in the lower triangular of

those specified rows in **S** are free parameters. This shows that parameter assignment with the row *location* notation depends on the matrix type.

With the use of the *parameter-spec* list, the parameter assignment with the row *location* notation stops when all the parameters are assigned. For example, consider the following specification of a 4×4 *general* (rectangular) matrix **A**:

```
matrix A
   [2,] = a b c;
```

The three parameters a, b, and c, are assigned to the matrix elements `[2,1]`, `[2,2]`, and `[2,3]`, respectively. However, a different assignment of the parameters applies if you use the same specification for a 4×4 *symmetric* matrix **S**, as shown in the following statement:

```
matrix S
   [2,] = a b c;
```

Because there are redundant elements in a symmetric matrix, you can specify only the lower triangular elements. Therefore, the row *location* specification is equivalent to the following elementwise specification:

```
matrix S
   [2,1] = a ,
   [2,2] = b ,
   [3,1] = c ;
```

When all the valid row elements are assigned with the parameters, the assignment proceeds to the next row. This is why the last parameter assignment is for $\mathbf{S}[3, 1]$. The same assignment sequence applies to matrices with the lower triangular type (LOW).

For matrices with the upper triangular matrix type (UPP), only the elements in the upper triangular are assigned. For example, consider a 4×4 upper triangular matrix **U** with the following row *location* specification:

```
matrix U
   [2,] = a b c d;
```

The assignment of parameters is the same as the following elementwise specification:

```
matrix U
   [2,2] = a ,
   [2,3] = b ,
   [2,4] = c ,
   [3,3] = d;
```

The first valid element in the second row of the **U** matrix is $\mathbf{U}[2, 2]$. Because all the valid elements in the second row are assigned with parameters, the last element has to go to the valid element in the next row. Hence, the parameter d is assigned to $\mathbf{U}[3, 3]$.

For matrices with the diagonal matrix type (DIA), only the diagonal elements are assigned. For example, consider a 4×4 upper diagonal matrix **D** with the following row *location* specification:

```
matrix D
   [2,] = a b c;
```

The assignment of parameters is the same as the following elementwise specification:

```
matrix D
   [2,2] = a ,
   [3,3] = b ,
   [4,4] = c ;
```

If you use an @ sign before the row number in the row *location* specification, the row number cannot move—it cannot proceed to the next row even if the valid elements in that row are already filled with the parameters in *parameter-spec*. All other behavior of the **[@i,]** specification is the same as that of the **[i,]** specification. For example, consider the following specification of a 4×4 *general* (rectangular) matrix **A**:

```
matrix A
   [@2,] = a b c d;
```

The four parameters a, b, c, and d, are assigned to the matrix elements **[2,1]**, **[2,2]**, **[2,3]**, and **[2,4]**, respectively. This is exactly the same result as the following specification without the @ sign:

```
matrix A
   [2,] = a b c d;
```

Here, all the elements of the second row of matrix **A** are assigned with elements. However, if one more parameter is specified in the *parameter-spec* list, the behavior for the two types of row *location* specifications are different. The following specification without the @ sign proceeds to the next row for the last parameter:

```
matrix A
   [2,] = a b c d e;
```

That is, the parameter e is assigned to the $\mathbf{A}[3, 1]$ element. However, the following specification *with* the @ sign results in an out-of-bound error:

```
matrix A
   [@2,] = a b c d e;
```

The out-of-bound error is due to the fact that the row number must be fixed so that the parameter e is forced to be assigned to $\mathbf{A}[2, 5]$, which does not exist.

However, the distinction between the row *location* specifications with and without the @ sign is not very important in common practice because in most cases you do not want the parameter assignment to proceed row after row automatically with a long list of parameters. For example, consider the following specification of a 4×4 *symmetric* matrix **S**:

```
matrix S
   [2,] = s21 s22 s31 s32 s33 s41 s42 s43;
```

This specification is equivalent to the following specification:

```
matrix S
   [2,] = s21 s22,
   [3,] = s31 s32 s33,
   [4,] = s41 s42 s43;
```

Although this specification is not as concise as the preceding one, it specifies more clearly about how parameters are assigned to each of the three rows of the **S** matrix. In this specification, you make sure that each of the three row *location* specifications has just enough parameters for the given row without proceeding to the next row for additional parameter assignments. With this kind of "careful" row *location* specifications, you do not need to use the @ sign before the row numbers at all.

The last type of row *location* specification is the `[`*iset*`,]` notation, where *iset* means a set of row numbers. This specification type provides the set of row numbers for the assignment of the parameters in the *parameter-spec* list. For example, consider the following specification of a 4×4 *general* matrix **A**:

```
matrix A
   [2 4,] = a21 a22 a23 a24 a41 a42 a43 a44;
```

This specification is equivalent to the following statement with two row *location* specifications:

```
matrix A
   [2,] = a21 a22 a23 a24,
   [4,] = a41 a42 a43 a44;
```

In other words, the assignment of parameters follows the order of rows provided in the *iset*. Notice that the *iset* notation merely provides the order of rows to be assigned with the parameters in the *parameter-spec* list; it is not an error if you provide a shorter parameter list than that of the total number of elements in the rows. For example, the following specification of a 4×4 *general* matrix **A** is valid:

```
matrix A
   [2 4,] = a21 a22 a23 a24;
```

This specification has the same results as the following statement with one row *location*:

```
matrix A
   [2,] = a21 a22 a23 a24;
```

However, a valid specification does not mean it is a good representation of the problem. Providing more rows in the *iset* specification than intended is simply not a good practice.

Although a shorter *parameter-spec* list is acceptable, a longer list results in an error. For example, the following specification of a 4×4 *symmetric* matrix **S** results in an error:

```
matrix S
   [2 to 3,] = s21 s22 s31 s32 s33 extra1 extra2;
```

The `[2 to 3,]` not only gives the sequence of the rows for the parameter assignment, it also limits the set of rows to assign. Because matrix **S** is symmetric and because only the second and the third rows are supposed to be assigned with the *iset* specification, the parameters extra1 and extra2 are excessive.

Column Location Only: `[,j]`, `[,@j]`, or `[,`*jset*`]`

These notations mirror that of the row *location* notations. Instead of the rows being specified, the columns are specified by these notations. Therefore, you can understand the column *location* notations the same way as the row *location* notations.

All these column *location* notations provide the starting column `[,j1]` for the assignment of the parameters in *parameter-spec*, where `j1` is `j` for the first two *location* notations or `j1` is the first column specified in *jset*, where *jset* = (`j1`, `j2`, . . .) is a set of column numbers. Because no row location is specified, the starting element is the first valid element in the `j1`th column of the matrix.

If no *parameter-spec* list is specified, all the valid elements in the entire `j1`th column of the matrix are unnamed free parameters. If a set of column numbers is specified in *jset*, all the valid elements in the all the columns specified in *jset* are unnamed free parameters.

If a *parameter-spec* list is specified, the assignment of parameters starts with the first valid elements of the `j1`th column. The assignment proceeds to next valid elements in the same column. The `[,j]` specification

proceeds column by column for parameter assignment while the `[,@j]` specification stays at the same `j`th column. The `[,jset]` specification indicates and limits the sequence of columns to be assigned with the parameter in the *parameter-spec* list. The assignment stops when all the parameters in the *parameter-spec* list are assigned. The following list summarizes how the assignment of parameters proceeds:

- `[,j]` specifies the first valid element in column `j` and proceeds to the valid elements in columns `j`, `j+1`, `j+2`, . . . , until all parameters in the *parameter-spec* list are assigned.
- `[,@j]` specifies the first valid elements in column `j` and proceeds to the valid elements in the *same* column until all parameters in the *parameter-spec* list are assigned.
- `[,`*jset*`]` specifies the first valid elements in column `j1`, where `j1`, `j2`, . . . are the columns specified in *jset*. It proceeds to the valid elements in columns `j1`, `j2`, . . . , until all parameters in the *parameter-spec* list are assigned.

See the section "Row Location Only: `[i,]`, `[@i,]`, or `[`*iset*`,]`" on page 1643 for examples, which are applicable to the usage of the column *locations*.

Row-and-Column-Sets Location: `[iset,jset]`, `[iset,j]`, *or* `[i,jset]`

These notations specify the sets of row and column elements for the assignment of the parameters in the *parameter-spec* list. In the first notation, you specify the set of row numbers in *iset* = (`i1`, `i2`, . . .), and the set of column numbers in *jset* = (`j1`, `j2`, . . .). The last two notations are special cases of the first notation. The `[iset,j]` notation specifies only one column with *jset* = `j1` = `j`. The `[i,jset]` notation specifies only one row with *iset* = `i1` = `i`. For the last two notations, adding the `@` sign before `j` or `i` is optional. In general, the row-and-column-sets *locations* specify the matrix elements in the following order:

```
[i1,j1], [i1,j2], ...,
[i2,j1], [i2,j2], ...,
[i3,j1], [i3,j2], ...,
..., ..., ...,
[ir,j1], [ir,j2], ..., [ir,js]
```

where `r` represents the number of rows in the *iset* and `s` represents the number of columns in the *jset*. Note that this ordering of elements does not necessarily mean that all these elements are specified. The number of elements specified depends on the length of the *parameter-spec* list.

If no *parameter-spec* list is specified after the *location* notation, all the `r` × `s` elements specified in the *iset* and *jset* are unnamed free parameters. PROC CALIS generates parameter names with the _Parm prefix for these elements.

If a *parameter-spec* list is specified after the *location* notation, the total number of matrix elements that are assigned with the parameters is the same as the number of parameter specifications in the *parameter-spec* list.

The following examples illustrates the usage of the row-and-column-sets *locations*.

The simplest case is the specification of all elements in the *iset* and *jset* as free parameters, as shown in the following statement:

```
matrix _Gamma_ [2 3,4 1];
```

This means that _Gamma_[2, 4], _Gamma_[2, 1], _Gamma_[3, 4], and _Gamma_[3, 1] are all free parameters in the matrix. For these elements, PROC CALIS generates parameter names with the _Parm prefix followed by a unique integer (for example, _Parm1, _Parm2, . . .). This row-and-column-sets *location* specification is the same as the following specification:

```
matrix _Gamma_ [2,4 1],[3,4 1];
```

It is also equivalent to the following elementwise specification:

```
matrix _Gamma_ [2,4],[2,1],[3,4],[3,1];
```

If you provide a *parameter-spec* list after the row-and-column-sets *location*, the parameters in the list are assigned to the matrix elements. For example, consider the following specification:

```
matrix _Gamma_ [2 3,4 1] = gamma1-gamma4;
```

This specification is equivalent to the following elementwise specification:

```
matrix _Gamma_ [2,4] = gamma1,
               [2,1] = gamma2,
               [3,4] = gamma3,
               [3,1] = gamma4;
```

It is not necessary for all the elements specified in the row-and-column-sets *location* to be assigned with the parameters in the *parameter-spec* list. For example, the following *iset* and *jset* specify a maximum of six elements, but only five parameters are assigned as a result of a shorter *parameter-spec* list:

```
matrix _Gamma_ [2 to 4,1 5] = gamma1-gamma4;
```

This specification is equivalent to the following elementwise specification:

```
matrix _Gamma_ [2,1] = gamma1,
               [2,5] = gamma2,
               [3,1] = gamma3,
               [3,5] = gamma4,
               [4,1] = gamma5;
```

In this case, _Gamma_[4, 5] is not specified and is fixed at zero by default.

With the row-and-column-sets *location* specifications, you need to be aware of the matrix type being specified. For example, the following specification of the symmetric matrix **S** results in an out-of-bounds error:

```
matrix S [1 2,1 2] = s1-s4;
```

This specification is equivalent to the following elementwise specification:

```
matrix S [1,1] = s1,
         [1,2] = s2,
         [2,1] = s3,
         [2,2] = s4;
```

The specification of the **S**[1, 2] element is not valid because you can specify only the lower triangular elements of a symmetric matrix in PROC CALIS. The upper triangular elements are redundant and are taken into account by PROC CALIS during computations.

Specifying Fixed and Free Parameters in Model Matrices

For clarity in describing various *location* notations, the *parameter-spec* list contains only free parameters in the examples. In general, you can specify fixed values, free parameters, and initial values in the *parameter-spec* list. The syntax for the *parameter-spec* list is the same as the *parameter-spec* list for the VARIANCE statement. You can specify the following five types of the parameters in the MATRIX statement:

- an unnamed free parameter
- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

The following example demonstrates these five types of specifications:

```
matrix A  [1,2],
          [1,3] = (.2),
          [1,4] = .3,
          [2,3] = a1,
          [2,4] = a2(.5);
```

In this statement, $\mathbf{A}[1,2]$ is an unnamed free parameter. For this element, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1). $\mathbf{A}[1,3]$ is an unnamed free parameter with an initial value of 0.2. PROC CALIS also generates a parameter name for this element. $\mathbf{A}[1,4]$ is fixed at 0.3. This value does not change in estimation. $\mathbf{A}[2,3]$ is a free parameter named a1. No initial value is given for this element. $\mathbf{A}[2,4]$ is a free parameter named a2 with an initial value of 0.5.

You can also specify different types of parameters in the *parameter-spec* list. The preceding specification is equivalent to the following specification:

```
matrix A  [1,2],
          [1 2,3 4] = (.2) .3  a1-a2 (.5);
```

Notice that 0.5 is the initial value for a2 but not for a1 because this specification is the same as:

```
matrix A  [1,2],
          [1 2,3 4] = (.2) .3  a1 a2(.5);
```

When you use *parameter-spec* lists with mixed parameters, you must be careful about how the initial value syntax is interpreted with and without a parameter name before it. With a parameter before the initial value, the initial value is for the parameter, as shown in the following statement:

```
matrix S  [1,1] = s1 s2 (.2);
```

This specification is the same as the following elementwise specification:

```
matrix S  [1,1] = s1,
          [2,2] = s2(.2);
```

This means that 0.2 is the initial value of parameter s2, but not interpreted as an unnamed free parameter for S[3, 3]. If you do intend to set the free parameter s2 for S[2, 2] without an initial value and set the initial value 0.2 for S[3, 3], you can use a null initial value for the s2 parameter, as shown in the following:

```
matrix S  [1,1] = s1 s2() (.2);
```

This specification is the same as the following elementwise specification:

```
matrix S  [1,1] = s1,
          [2,2] = s2,
          [3,3] = (.2);
```

Modifying a Parameter Specification from a Reference Model

If you define a new COSAN, LISMOD, or MSTRUCT model by using a reference (old) model in the REFMODEL statement, you might want to modify some parameter specifications from the MATRIX statement of the reference model before transferring the specifications to the new model. To change a particular matrix element specification from the reference model, you can simply respecify the same matrix element with the desired parameter specification in the MATRIX statement of the new model. To delete a particular matrix parameter from the reference model, you can specify the desired matrix element with a missing value specification in the MATRIX statement of the new model.

For example, suppose that _PHI_[1, 2] is a free parameter in the reference model but you do not want this matrix element be a free parameter in the new model, you can use the following specification in the new model:

```
matrix _PHI_  [1,2] = .;
```

Notice that the missing value syntax is valid only when you use the REFMODEL statement. See the section "Modifying a COSAN Model from a Reference Model" on page 1591 for a more detailed example of COSAN model respecification. See the section "Modifying a LISMOD Model from a Reference Model" on page 1626 for a more detailed example of LISMOD model respecification. See the section "Modifying an MSTRUCT Model from a Reference Model" on page 1657 for a more detailed example of MSTRUCT model respecification.

MEAN Statement

MEAN *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents:

var-list < **=** *parameter-spec* >

The MEAN statement specifies the mean or intercept parameters in connection with the FACTOR, LINEQS, and PATH modeling languages. With the MEAN statement specification, PROC CALIS analyzes the mean structures in addition to the covariance structures.

In each *assignment* of the MEAN statement, you list the *var-list* that you want to specify for their means or intercepts. Optionally, you can provide a list of parameter specifications in a *parameter-spec* after an equal sign for each *var-list*. The syntax of the MEAN statement is exactly the same as that of the VARIANCE statement. See the VARIANCE statement on page 1706 for details about the syntax.

For the confirmatory FACTOR or PATH model, the variables in a *var-list* can be exogenous or endogenous. You specify the mean of a variable if the variable is exogenous. You specify the intercept of a variable if the variable is endogenous. However, for the LINEQS model, you can specify only the means of exogenous variables whose type is not error (that is, not the E- or D- variables) in the MEAN statement. You cannot specify the intercept parameters in the MEAN statement for the LINEQS model. Instead, you must specify the intercepts in the equations of the LINEQS statement. For the exploratory FACTOR model, the MEAN statement is used merely to include the mean structures into analysis. In this case, the parameter specification by the *assignment* syntax is not interpreted. Instead, the intercept parameters for the exploratory factor model are automatically generated by PROC CALIS.

You can specify the following five types of the parameters for the means or intercepts in the MEAN statement:

- an unnamed free parameter
- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

For example, consider a PATH model with exogenous variables x1, x2, and x3 and endogenous variables y4 and y5. The following MEAN statement illustrates the five types of specifications in five *assignments*:

```
mean x1 ,
     x2 = (3.0),
     x3 = 1.5,
     y4 = intercept1,
     y5 = intercept2(0.6);
```

In this statement, the mean of x1 is specified as an unnamed free parameter. For this mean, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1). The mean of x2 is an unnamed free parameter with an initial value of 3.0. PROC CALIS also generates a parameter name for this mean. The mean of x3 is a fixed value of 1.5. This value stays the same during the estimation. The intercept of endogenous variable y4 is a free parameter named intercept1. The intercept of endogenous variable y5 is a free parameter named intercept2 with an initial value of 0.6.

The syntax of the MEAN statement is the same as the syntax of the VARIANCE statement. See the VARIANCE statement for more illustrations about the usage.

Default Mean and Intercept Parameters

If the mean structures are analyzed, either the means or intercepts of the *manifest* variables in the FACTOR, LINEQS, or PATH model are free parameters by default. If a manifest variable is exogenous, then its mean is a free parameter by default. If a manifest variable is endogenous, then its intercept is a free parameter by default. Except for the exploratory FACTOR model, PROC CALIS generates parameter names for these default free mean or intercept parameters that include the _Add prefix and are appended with unique integer suffixes. The default intercepts of the exploratory FACTOR model use the _a prefix instead. For the confirmatory FACTOR model and the PATH model, you can use the MEAN statement to override these default mean or intercept parameters in situations where you want to set parameter constraints, provide initial or fixed values, or make parameter references. However, you cannot override any default intercept parameters for the exploratory FACTOR model. For the LINEQS model, you can use the MEAN statement specification to override only the default mean parameters. The intercept parameters of the LINEQS model must be specified in the equations of the LINEQS statement.

Fixed zero is another type of default mean or intercept parameters for the FACTOR, LINEQS, or PATH model. All the intercepts and means of the *latent* variables in these models are fixed zeros by default. For the confirmatory FACTOR and PATH models, you can override these default fixed zeros by using the MEAN statement specifications. However, you cannot override the fixed zero factor means for the exploratory FACTOR model, For the LINEQS model, you can override only the default fixed zeros of latent variables whose type is not error. That is, you can use the MEAN statement to override the default zero mean for the exogenous latent factors (excluding the error or disturbance variables) or use the LINEQS statement to override the default zero intercept for the endogenous latent factors. The fixed zero means for the error or disturbance variables in the LINEQS model reflects model restrictions—there is no way you can override these default zero means.

Modifying a Mean or Intercept Parameter Specification from a Reference Model

If you define a new FACTOR (confirmatory factor model only), LINEQS, or PATH model by using a reference (old) model in the REFMODEL statement, you might want to modify some parameter specifications from the MEAN statement of the reference model before transferring the specifications to the new model. To change a particular mean or intercept specification from the reference model, you can simply respecify the same mean or intercept with the desired parameter specification in the MEAN statement of the new model. To delete a particular mean or intercept parameter from the reference model, you can specify the desired mean or intercept with a missing value specification in the MEAN statement of the new model.

For example, suppose that the mean of F1 is specified in the reference model, but you do not want this mean specification be transferred to the new model. You can use the following MEAN statement specification in the new model:

```
mean F1 = .;
```

Note that the missing value syntax is valid only when you use with the REFMODEL statement. See the section "Modifying a FACTOR Model from a Reference Model" on page 1608 for a more detailed example of FACTOR model respecification. See the section "Modifying a LINEQS Model from a Reference Model" on page 1622 for a more detailed example of LINEQS model respecification. See the section "Modifying a PATH Model from a Reference Model" on page 1673 for a more detailed example of PATH model respecification.

As discussed in a preceding section, PROC CALIS generates default free mean or intercept parameters for manifest variables in the FACTOR, LINEQS, or PATH model if you do not specify them explicitly in the MEAN statement (and the LINEQS statement for the LINEQS model). When you use the REFMODEL

statement for defining a reference model, these default free mean or intercept parameters in the old (reference) model are not transferred to the new model. Instead, the new model generates its own set of default free mean or intercept parameters *after* the new model is resolved from the reference model, the REFMODEL statement options, the RENAMEPARM statement, and the MEAN statement (and the LINEQS statement for the LINEQS model) specifications in the new model. This also implies that if you want any of the mean or intercept parameters to be constrained across the models by means of the REFMODEL specification, you must specify them explicitly in the MEAN statement (or the LINEQS statement for the LINEQS model) of the reference model so that the same mean or intercept specification is transferred to the new model.

MODEL Statement

MODEL *i* < / *options* > **;**

where *i* is an assigned model number between 1 and 9999, inclusively.

A MODEL statement signifies the beginning of a model specification block and designates a model number for the model. All main and subsidiary model specification statements after a MODEL statement belong in that model until another MODEL or GROUP statement is encountered.

The MODEL statement itself does not serve the purpose of model specification, which is actually done by the main and subsidiary model specification statements that follow it. The MODEL statement serves as a "place-holder" of specification of a single model. It also makes the reference to a model easier with an assigned model number. For example, consider the following statements:

```
proc calis;
   group 1 / data=women_data;
   group 2 / data=men_data;
   model 1 / group=1 label='Women Model';
      {model 1 specification here}
   model 2 / group=2 label='Men Model';
      {model 2 specification here}
run;
```

This example illustrates a two-group analysis with two models. One is **model 1** labeled as 'Women Model' in a MODEL statement. Another is **model 2** labeled as 'Men Model' in another MODEL statement. The two groups, **group 1** and **group 2**, as defined in two separate GROUP statements, are fitted by **model 1** and **model 2**, respectively, as indicated by the GROUP= option of the MODEL statements. Within the scope of **model 1**, you provide model specification statements by using the main and subsidiary model specification statements. Usually, one of the following main model specification statements is used: FACTOR, LINEQS, LISMOD, MSTRUCT, PATH, RAM, or REFMODEL. Similarly, you provide another set of model specification statements within the scope of **model 2**.

Hence, for an analysis with a single group, the use of the MODEL statement is not necessary because the model that fits the group is unambiguous.

You can set model-specific *options* in each MODEL statement. All but two of these *options* are also available in the PROC CALIS statement. If you set these *options* in the PROC CALIS statement, they apply to all models, unless you respecify them in the local MODEL statements. If you want to apply some *options* only to a particular model, specify these *options* in the MODEL statement that corresponds to that model.

You can also set group-specific *options* in the MODEL statement. These group *options* apply to the groups that are specified in GROUP= option of the MODEL statement. See the section "Options Available in the GROUP and PROC CALIS Statements" on page 1615 for a detailed descriptions of these group *options*.

Options Available Only in the MODEL Statement

LABEL=*name*

NAME=*name*

specifies a label for the model. You can use any valid SAS names up to 256 characters for labels. You can also use quoted strings for labels.

GROUP= [*int-list***] | {***int-list***}**

GROUPS= [*int-list***] | {***int-list***}**

specifies a list of integers which represent the groups to be fitted by the model.

Options Available in the MODEL and PROC CALIS Statements

The following *options* are available in the MODEL and PROC CALIS statements. If you specify these *options* in the PROC CALIS statement, they are transferred to all MODEL statements. These *options* might be overwritten by the respecifications in the local MODEL statements.

Option	Description
DEMPHAS= on page 1553	Emphasizes the diagonal entries
EFFPART | TOTEFF on page 1554	Displays total, direct, and indirect effects
EXTENDPATH | GENPATH on page 1554	Displays the extended path estimates that include variances and covariances
INEST= on page 1555	Specifies the data set that contains the initial values and constraints
INMODEL | INRAM= on page 1555	Specifies the data set that contains the model specifications
MEANSTR on page 1562	Analyzes the mean structures
MODIFICATION on page 1564	Computes modification indices
NOMEANSTR on page 1565	Deactivates the inherited MEANSTR option
NOMOD on page 1566	Suppresses modification indices
NOORDERSPEC on page 1566	Displays model specifications and results according to the input order
NOPARMNAME on page 1566	Suppresses the printing of parameter names in results
NOSTAND on page 1566	Suppresses the standardized output
NOSTDERR on page 1566	Suppresses standard error computations
ORDERSPEC on page 1568	Orders the model output displays according to the parameter types within each model
OUTEST= on page 1569	Specifies the data set that outputs the estimates and their covariance matrix
OUTMODEL | OUTRAM= on page 1569	Specifies the output data set for storing the model specification and results
PARMNAME on page 1570	Displays parameter names in model specifications and results

Option	Description
PDETERM on page 1571	Computes the determination coefficients
PESTIM on page 1571	Prints parameter estimates
PINITIAL on page 1571	Prints initial pattern and values
PLATCOV on page 1571	Computes the latent variable covariances and scoring coefficients
PRIMAT on page 1573	Displays results in matrix forms
READADDPARM on page 1574	Instructs generated default parameters be read in the INMODEL= data set
STDERR on page 1578	Computes the standard errors

Options Available in MODEL, GROUP, and PROC CALIS Statements

Some options in the GROUP statement can also be specified in the MODEL statements. Group options that are specified the MODEL statements are transferred to the GROUP statements that define the groups that are fitted by the associated models in the MODEL statements. This is a little more convenient than setting the common group options individually in the GROUP statements for all fitted groups by a model. See the section "Options Available in GROUP, MODEL, and PROC CALIS Statements" on page 1616 for a reference of these *options*.

MSTRUCT Statement

MSTRUCT < **VAR=***var-list* > **;**

MSTRUCT stands for matrix structures. As opposed to other modeling languages, in which the mean and covariance structures are implied from paths, equations, or complicated model matrix computations, the MSTRUCT language is for direct structured mean and covariance models.

In the MSTRUCT statement, you define the list of variables. You can use MATRIX statements to specify the parameters in the mean and covariance structures:

```
MSTRUCT <VAR=var-list> ;
    MATRIX _COV_ parameters-in-matrix ;
    MATRIX _MEAN_ parameters-in-matrix ;
```

You use the MATRIX _COV_ statement to specify the covariance and variance parameters in the structured covariance matrix. When applicable, you use the MATRIX _MEAN_ statement to specify the parameters in the structured mean vector. Each of these matrices can be specified no more than once within a model. See the MATRIX statement on page 1638 for details. If you do not use any MATRIX statement for specifying parameters, a saturated model is assumed. This means that all elements in the covariance and mean (if modeled) matrices are free parameters in the model.

The order of variables in the *var-list* of the MSTRUCT statement is important; it is used to refer to the row and column variables of the _COV_ and the _MEAN_ matrices. The variables specified in the list should be present in the input data set that is intended for the MSTRUCT model. With direct mean and covariance structures on the observed variables, no latent variables are explicitly involved in the MSTRUCT modeling language. However, this does not mean that the MSTRUCT modeling language cannot handle latent variable models. With additional specifications in the PARAMETERS and the SAS programming statements, it

is possible to fit certain latent variable models by using the MSTRUCT modeling language. Despite this, the code might get too complicated and error-prone. Hence, using the MSTRUCT modeling language for latent variable modeling is not recommended for novice users. The LINEQS, LISMOD, PATH, or RAM modeling language should be considered first for latent variable modeling. For applications of the MSTRUCT modeling, see Yung, Browne, and Zhang (2015).

Default Parameters

It is important to understand the default parameters in the MSTRUCT model. If you know which parameters are default free parameters, you can make your specification more efficient by omitting the specifications of those parameters that can be set by default. For example, you do not need to specify any elements of the _COV_ matrix if all elements are supposed to free parameters. See the section "Default Parameters in the MSTRUCT Model" on page 1762 for details about the default parameters of the FACTOR model.

Modifying an MSTRUCT Model from a Reference Model

This section assumes that you use a REFMODEL statement within the scope of a MODEL statement and that the reference model (or base model) is also an MSTRUCT model. The reference model is called the old model, and the model that refers to the old model is called the new model. If the new model is not intended to be an exact copy of the old model, you can use the following extended MSTRUCT modeling language to make modifications within the scope of the MODEL statement for the new model. The syntax is similar to, but not exactly the same as, the ordinary MSTRUCT modeling language, as described in the section "MSTRUCT Statement" on page 1656. The syntax for respecifying or modifying an MSTRUCT model takes the following form:

```
MSTRUCT ;
    MATRIX _COV_ parameters-in-matrix ;
    MATRIX _MEAN_ parameters-in-matrix ;
```

In the respecification, you should not put any VAR= list in the MSTRUCT statement, as you would do for specifying the original base model. The reason is that parameter respecifications in the new model refer to the variables in the VAR= list of the old model. Therefore, the VAR= list in the new model is implicitly assumed to be exactly the same as that in the old model. This renders the specification of a VAR= list of the MSTRUCT statement of the new model unnecessary. Because the VAR= option is the only possible option in the MSTRUCT statement, it also implies that the entire MSTRUCT statement is optional for the new model.

You can use the MATRIX _COV_ and MATRIX _MEAN_ statements to modify from the old model by using the same syntax as in ordinary MSTRUCT modeling language. In addition, in the respecification syntax, you can use the missing value '.' to drop a parameter location from the old model.

The new model is formed by integrating with the old model in the following ways:

Duplication: If you do not specify in the new model a parameter location that exists in the old model, the old parameter specification is duplicated in the new model.

Addition: If you specify in the new model a parameter location that does not exist in the old model, the new parameter specification is used in the new model.

Deletion: If you specify in the new model a parameter location that also exists in the old model and the new parameter is denoted by the missing value '.', the old parameter specification is not copied into the new model.

Replacement: If you specify in the new model a parameter location that also exists in the old model and the new parameter is not denoted by the missing value '.', the new parameter specification replaces the old one in the new model.

For example, consider the following statements for a two-group analysis:

```
proc calis;
   group 1 / data=d1;
   group 2 / data=d2;
   model 1 / group=1;
      mstruct var=V1-V6;
      matrix _COV_ [1,1] = 6*vparm (8.),
                   [2,]  = cv21,
                   [3,]  = cv31,
                   [4,]  = cv41 cv42 cv43,
                   [5,]  = cv51 cv52 cv53 cv54,
                   [6,]  = cv61 cv62 cv63 cv64 cv65;
   model 2 / group=2;
      refmodel 1;
      matrix _COV_ [2,]  = 3.,
                   [3,2] = cv32,
                   [4,]  = . . . ,
                   [5,]  = . . . ,
                   [6,]  = . . . ;
run;
```

In these statements, you specify Model 2 by referring to Model 1 in the REFMODEL statement. Hence, Model 2 is called the new model that refers to the old model, Model 1. Because they are not respecified in the new model, all parameters on the diagonal of the covariance matrix are duplicated from the old model for the new model. Similarly, parameter locations associated with the cv54, cv64, and cv65 parameters are also duplicated in the new model.

An added parameter in the new model is cv32 for the covariance between V3 and V2. This parameter location is not specified in the old model.

In the new model, parameters for the covariances between the variable sets {V1 V2 V3} and {V4 V5 V6} are all deleted from the old model. The corresponding parameter locations for these covariances are given missing values '.' in the new model, indicating that they are no longer free parameters as in the old model. Deleting these parameters amounts to setting the corresponding covariances to fixed zeros in the new model.

Finally, covariance between V2 and V1 is changed from a free parameter cv21 in the old model to a fixed constant 3 in the new model. This illustrates the replacement rule of the respecification syntax.

NLINCON Statement

NLINCON | NLC *constraint* < **,** *constraint . . .* > **;**

where *constraint* represents one of the following:

- *number operator variable-list number operator*
- *variable-list number operator*
- *number operator variable-list*

You can specify nonlinear equality and inequality constraints with the NLINCON or NLC statement. The QUANEW optimization subroutine is used when you specify nonlinear constraints by using the NLINCON statement.

The syntax of the NLINCON statement is similar to that of the BOUNDS statement, except that the NLINCON statement must contain the names of variables that are defined in the program statements and are defined as continuous functions of parameters in the model. They must not be confused with the variables in the data set.

As with the BOUNDS statement, one- or two-sided constraints are allowed in the NLINCON statement; equality constraints must be one sided. Valid operators are <=, <, >=, >, and = (or, equivalently, LE, LT, GE, GT, and EQ).

PROC CALIS cannot enforce the strict inequalities < or > but instead treats them as <= and >=, respectively. The listed nonlinear constraints must be separated by commas. The following is an example of the NLINCON statement that constrains the nonlinear parametric function $x_1 * x_1 + u_1$ to a fixed value of 1:

```
nlincon    xx = 1;
xx = x1 * x1 + u1;
```

Note that x1 and u1 are parameters defined in the model. The following three NLINCON statements, which require xx1, xx2, and xx3 to be between zero and ten, are equivalent:

```
nlincon  0. <= xx1-xx3,
               xx1-xx3 <= 10;
nlincon 0. <= xx1-xx3 <= 10.;
nlincon 10. >= xx1-xx3 >= 0.;
```

NLOPTIONS Statement

NLOPTIONS *options* **;**

Many options that are available in SAS/OR PROC NLP can be specified for the optimization subroutines in PROC CALIS by using the NLOPTIONS statement. The NLOPTIONS statement provides more displayed and file output control on the results of the optimization process, and it permits the same set of termination criteria as in PROC NLP. These are more technical options that you might not need to specify in most cases.

Several statistical procedures support the use of NLOPTIONS statement. The syntax of NLOPTIONS statement is common to all these procedures and can be found in the section "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics."

See the section "Use of Optimization Techniques" on page 1854 for more information about the use of optimization techniques in PROC CALIS.

OUTFILES Statement

OUTFILES | **OUTFILE** *file-option* < *file-option . . .* > **;**

where *file-option* represents one of the following:

- **OUTMODEL** | **OUTRAM**= *file-name* [**MODEL**= *int-list* < , *int-list* >]
- **OUTSTAT**= *file-name* [**GROUP**= *int-list* < , *int-list* >]
- **OUTWGT**= *file-name* [**GROUP**= *int-list* < , *int-list* >]

with *file-name* representing an output file name and *int-list* representing list of model or group numbers

Use the OUTFILES statement when you need to output multiple-group or multiple-model information to output files in a complex way. In each OUTFILES statement, each possible *file-option* should appear no more than once. However, as needed, you can use the OUTFILES statement more than once. For example, suppose you want to create two OUTWGT= files for different sets of groups. You can specify the OUTFILES statement twice, as shown in the following specification:

```
outfiles outwgt=file1 [group=1,2];
outfiles outwgt=file2 [group=3,4];
```

In the first OUTFILES statement, the weights for groups 1 and 2 are output to the file file1. In the second OUTFILES statement, the weights for groups 3 and 4 are output to the file file2.

When the OUTMODEL=, OUTSTAT=, or OUTWGT= option is intended for *all* groups or models, you can simply specify the option in the PROC CALIS statement. Only when you need to output the group (model) information from *more than one* group (model), *but not all* groups (models), to a single output file does the use the OUTFILES statement become necessary. For example, consider the following specification:

```
proc calis method=gls;
   outfiles outmodel=outmodel [model=1,3]
            outwgt=outwgt [group=1,2]
            outstat=outstat [group=2,3];
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3 outwgt=outwgt3;
   model 1 / group=1;
      factor N=3;
   model 2 / group=2;
      factor N=2;
   model 3 / group=3;
      factor N=3;
run;
```

You fit three different factor models to three groups: Model 1 for Group 1, Model 2 for Group 2, and Model 3 for group 3. In the OUTFILES statement, you output model information from models 1 and 3 to an output

file named outmodel, weight matrices from groups 1 and 2 to outwgt, and statistics from groups 2 and 3 to outstat. In each of these output files, you have information from more than one (but not all) groups or models. In the GROUP statement for group 3, you have another OUTWGT= file named outwgt3 for group 3 alone.

Note that you cannot specify the preceding output file organization by using the following statements:

```
proc calis method=gls;
   group 1 / data=g1 outwgt=outwgt;
   group 2 / data=g2 outwgt=outwgt outstat=outstat;
   group 3 / data=g3 outwgt=outwgt3 outstat=outstat;
   model 1 / group=1 outmodel=outmodel;
      factor N=3;
   model 2 / group=2;
      factor N=2;
   model 3 / group=3 outmodel=outmodel;
      factor N=3;
run;
```

This specification will not work because SAS forbids the repeated specification of the same output file in the same PROC CALIS run. That is, you cannot specify OUTWGT=outwgt, OUTSTAT=outstat, or OUTMODEL=outmodel more than once in the PROC CALIS run without causing file invocation problems (however, multiple specification of the same input file is not a problem).

If you specify any of the output files for a group (or a model) in both of the OUTFILES and the GROUP (or MODEL) statements, the destination specified in the more specific GROUP (or MODEL) statement will be used. For example, for the following specification PROC CALIS will save the Model 2 information in the OUTMODEL=outmodel2 data set, but not in the OUTMODEL=outfile1 data set:

```
proc calis method=gls;
   outfiles outmodel=outfile1 [model=1,2];
   group 1 / data=g1;
   group 2 / data=g2;
   model 1 / group=1;
      factor N=3;
   model 2 / group=2 outmodel=outmodel2;
      factor N=2;
run;
```

The OUTFILES statement is intended for arranging output files in a complex way. The use of the OUTFILES statement is unnecessary in the following situations:

- If you have a single-sample analysis, you do not need to use the GROUP statement. As a result, you can simply use the OUTSTAT= or OUTWGT= options in the PROC CALIS statement for specifying the output destinations. Therefore, the OUTFILES statement is not needed.
- If you have a single model in your analysis, you do not need to use the MODEL statement. As a result, you can simply use the OUTMODEL= options in the PROC CALIS statement for specifying the output destination. Therefore, the OUTFILES statement is not needed.
- If you have multiple groups or multiple models in your analysis and information for all groups or models is output to the same file, you do not need to use the OUTFILES statement. You can simply use the OUTSTAT=, OUTWGT=, or OUTMODEL= options in the PROC CALIS statement because the output file information is automatically propagated from the PROC CALIS statement to the groups or models.

- If you have multiple groups or multiple models in your analysis and each group or model has a unique output data file destination (including cases where some groups or models might not have any output files), you do not need to use the OUTFILES statement. You can simply specify the OUTSTAT=, OUTWGT=, or OUTMODEL= options in the GROUP or MODEL statements.

PARAMETERS Statement

PARAMETERS | PARMS *parameters* <<**=**> *numbers* | (*numbers*)>
<<**,**> *parameters* <<**=**> *numbers* | (*numbers*)> . . . > **;**

The PARAMETERS statement defines parameters that are useful in the modeling. You can specify more than one PARAMETERS statement.

The three main uses of the PARAMETERS statement are as follows:

- Define independent parameters that are not specified in your model. In some modeling situations, it is more convenient (practically or conceptually) to define the model parameters as functions of these independent parameters. PROC CALIS computes the estimates and their standard errors for these independent parameters.
- Define dependent parameters of interest. These dependent parameters are then defined as functions of the model parameters in the SAS programming statements. PROC CALIS computes the estimates and their standard errors for these dependent parameters.
- Provide initial values for the free parameters in the model if they have not been specified in the model specification statements.

For example, the following statements illustrate the three main uses of the PARAMETERS statement:

```
proc calis;
   path
      V1    <===  V2    = b1,
      V1    <===  V3    = b2,
      V2    <===  V4    = b3 (0.9),
      V3    <===  V5    = b4;
   parameters  a1 (0.1)  a2 (0.2)  b3 (0.3)  b4 (0.4) b5;
   b1  =  -a1 / a2;
   b2  =  1 / a2;
   b5  =  b3 - b4;
run;
```

In the PARAMETERS statement, you specify five parameters that take different roles in the modeling. Parameters a1 and a2 are independent parameters that are not specified in the PATH statement, which specifies four paths with four parameters for the path coefficients b1, b2, b3, and b4. The two SAS programming statements immediately after the PARAMETERS statement define parameters b1 and b2 as functions of parameters a1 and a2. Because a1 and a2 appear only on the right side of the programming statements, they are independent parameters of the model. In addition, b1 and b2 are dependent parameters because they appear on the left side of the first two programming statements. Independent parameters a1 and a2 are provided with initial values 0.1 and 0.2, respectively, in the PARAMETERS statement. Because the

initial values these two independent parameters are used, dependent parameters b1 and b2 are also initialized with values –0.5 and 5, respectively.

Parameters b3 and b4 appear in both the PATH and PARAMETERS statements. Because these two parameters are already specified in the PATH statement, their initial values might have been defined in the model. For example, parameter b3 in the PATH statement has an initial value of 0.9. This initial value is *not* replaced by the initial value of 0.3 specified in the PARAMETERS statement. However, because you do not specify an initial value for parameter b4 in the PATH statement, the initial value of 0.4 specified in the PARAMETERS statement is used. In general, the initial values that are specified in the PARAMETERS statement do not replace the initial values that have already been specified for the same parameters in the model.

Parameter b5 in the PARAMETERS statement is a dependent parameter, which is defined as the difference between the model parameters b3 and b4 in the last SAS programming statement. No initial value for this dependent parameter is provided (nor is it needed). By definition, dependent parameters are functions of other parameters, so their initial values are computed. In this example, parameter b5 takes an initial value of 0.5, the difference between the initial values of b3 and b4.

It is not necessary to provide initial values for the parameters in the PARAMETERS statement. For the independent parameters without initial values specified, PROC CALIS generates the initial values from the START= value in the PROC CALIS statement. The default START= value is 0.5. If you specify the RANDOM= option in the PROC CALIS statement, random numbers are generated for the initial values of the independent parameters. For the dependent parameters, PROC CALIS computes the initial values by using the SAS programming statements.

In general, the number of parameters and the number of initial values do not have to match. When you specify fewer initial values than parameter names, PROC CALIS either generates or computes the initial values. When you specify more values than parameter names, the extra values are ignored.

For example, consider the following statements:

```
parameters  c1 c2 c3 c4 (0.2 0.3 0.4);
parameters  d1 d2 d3 d4 (0.1 0.2 0.3 0.4 0.5);
run;
```

The first PARAMETERS statement has a shorter initial value list than the parameter list. The initial values 0.2, 0.3, and 0.4 are assigned to c2, c3, and c4, respectively. PROC CALIS generates the initial value for c1. The second PARAMETERS statement has a longer initial value list than the parameter list. The initial values 0.1, 0.2, 0.3, and 0.4 are assigned to d1, d2, d3, and d4, respectively. The extra initial value 0.5 is ignored.

When the lengths of the parameter list and the initial value list match, you can use an equal sign between the lists freely without affecting the assignments of the initial values. For example, each of the following PARAMETERS statements assigns the initial values alpha=0.5 and beta=–0.5:

```
parameters  alfa   0.5  beta   -0.5;
parameters  alfa = 0.5  beta = -0.5;
parameters  alfa   (0.5) beta   (-0.5);
parameters  alfa = (0.5) beta = (-0.5);
parameters  alfa beta   0.5 -0.5;
parameters  alfa beta = 0.5 -0.5;
parameters  alfa beta   (0.5 -0.5);
parameters  alfa beta = (0.5 -0.5);
```

However, when the parameter list is longer than the initial value list, the equal sign affects the assignments of the initial values. For example, the following statement assigns the initial values to the last three parameters, c2, c3, and c4:

```
parameters  c1 c2 c3 c4 (0.2 0.3 0.4);
```

But with an equal sign between the lists, the following statement assigns the initial values to the first three parameters, c1, c2, and c3:

```
parameters  c1 c2 c3 c4 = (0.2 0.3 0.4);
```

To make initial value assignments less ambiguous, you can use commas to separate parameter lists. For example, the following statement clearly assigns the initial values to c2, c3, and c4:

```
parameters  c1, c2 c3 c4 (0.2 0.3 0.4);
```

This specification is equivalent to the same specification without the comma. But the specification is certainly less ambiguous with the comma.

Proper grouping of the parameter and initial value lists with the use of commas helps make the specification in the PARAMETERS statement clear. For example, consider the following specification:

```
parameters  c1 c2 c3 c4 = (0.1 0.2 0.3) c5 c6 (0.6);
```

Initial values 0.1, 0.2, 0.3, and 0.6 are assigned to parameters c1, c2, c3, and c6, respectively. Initial values for other parameters are either generated or computed by PROC CALIS. A much better practice is to use the following equivalent specification:

```
parameters  c1 c2 c3 (0.1 0.2 0.3), c4 c5, c6 (0.6);
```

Matching the number of parameters and the number of initial values (if provided) in entries separated by commas in the PARAMETERS statement is highly recommended. It reduces ambiguities and makes your code more interpretable.

It is important to notice that PROC CALIS does not have better alternatives to generate initial values (such as those for setting the initial values of the model parameters) for the independent parameters specified in the PARAMETERS statement other than using either the START= value or random values generated by using the RANDOM= option. These ad hoc initial values are arbitrary and might not always lead to converged solutions. Therefore, you should try to provide good initial values for those independent parameters that are defined only in the PARAMETERS statement.

Do not confuse the PARAMETERS statement with the VAR statement. While you specify the parameters of the model in the PARAMETERS statement, you specify analysis variables in the VAR statement. See the VAR statement on page 1703 for more details.

CAUTION: The OUTMODEL= or OUTRAM= data sets do not contain any information about the PARAMETERS statement or the SAS programming statements.

PARTIAL Statement

PARTIAL *variables* **;**

If you want the analysis to be based on a partial correlation or covariance matrix, use the PARTIAL statement to list the variables used to partial out the variables in the analysis. You can specify only one PARTIAL statement within the scope of each GROUP or PROC CALIS statement.

PATH Statement

PATH *path* < **,** *path . . .* > **;**

where *path* represents any of the following specifications:

- single-headed path for defining functional relationship
- double-headed path for specifying variances or covariances
- 1-path for specifying means or intercepts

For example, the following PATH statement contains only the single-headed paths:

```
path
   v1    <===  v2,
   v2    <===  v4 v5,     /* same as: v2 <=== v4 and v2 <=== v5 */
   v3    ===>  v5,        /* same as: v5 <=== v3 */
   v4 v5 <===  v6 v7;     /* same as: v4 <=== v6, v4 <=== v7,
                             v5 <=== v6, and v5 <=== v7 */
```

Although the most common definition of paths refer to these single-headed paths, PROC CALIS extends the definition of paths to include the so-called "variance-paths," "covariance-paths," and "1-paths" that refer to the variance, covariance, and the mean or intercept parameters, respectively. Corresponding to these extended path definitions, PROC CALIS provides the double-headed path and 1-path syntax. For example, the following PATH statement contains single-headed paths for specifying functional relationships and double-headed paths for specifying variances and covariances:

```
path
   v1    <===  v3-v5,    /* same as: v1 <=== v3, v1 <=== v4, and v1 <=== v5 */
   v2    <===  v4 v5,
   v3    <===  v5,
   v1    <==>  v1,       /* error variance of v1 */
   <==> v2 v3,           /* error variances of v2 and v3 */
   v2    <==>  v3,       /* error covariance between v2 and v3 */
   <==> [v4 v5];         /* variances and covariance for v4 and v5 */
```

The following PATH statement contains single-headed paths for specifying functional relationships and 1-paths for specifying means and intercepts:

```
path
   v1    <===  v3-v5,
   v2    <===  v4 v5,
   v3    <===  v5,
   1     ===>  v1,       /* intercepts for v1 */
   1     ===>  v2-v3,    /* intercepts for v2 and v3 */
   1     ===>  v4 v5;    /* means of v4 and v5 */
```

Details about the syntax of these three different types of *paths* are described later. Instead of using double-headed paths and 1-paths, you can also specify these parameters by the subsidiary model specification statements such as the PVAR, PCOV, and the MEAN statements, as shown in the following syntactic structure of the PATH modeling language:

PATH *path* < **,** *path* . . . > **;**
  PVAR *partial-variance-parameters* **;**
  PCOV *partial-covariance-parameters* **;**
  MEAN *mean-parameters* **;**

Typically, in this syntactic structure the *paths* contains only single-headed paths for representing the functional relationships among variables, which could be observed or latent. The *paths* are separated by commas. You can specify at most one PATH statement in a model within the scope of either the PROC CALIS statement or a MODEL statement.

Next, the PVAR statement specifies the parameters for the variances or error (partial) variances. The PCOV statement specifies the parameters for the covariances or error (partial) covariances. The MEAN statement specifies the parameters for the means or intercepts. For details about these subsidiary model specification statements, see the syntax of the individual statements.

A natural question now arises. For the specification of variances, covariances, intercepts, and means, should you use the extended path syntax that includes double-headed paths and 1-paths or the subsidiary model specification statements such as the PVAR, PCOV, and MEAN statements? If you want to specify all parameters in a single statement and hence output and view all the parameter estimates in a single output table, then the extended path syntax would be your choice. If you want to use more common language for specifying and viewing the parameters or the estimates of variances, covariances, means, and intercepts, then the subsidiary model specification statements serve the purpose better.

You are not restricted to using extended path syntax or the subsidiary model statements exclusively in a PATH model specification. For example, you might specify the variance of v1 by using the double-headed path syntax and the variance of v2 by using the PVAR statement. The only restriction is that you cannot specify the same parameter twice. In addition, even if you specify your PATH model without using double-headed paths or 1-paths, you can include the estimation results associated with these extended paths in the same output table for the single-headed paths by using the EXTENDPATH or GENPATH option. This way all the estimates of the PATH model can be shown in a single output table.

The Single-Headed Path Syntax for Specifying Functional Relationships

var-list arrow var-list2 < = *parameter-spec* >

where *var-list* and *var-list2* are lists of variables, *parameter-spec* is an optional specification of parameters, and *arrow* represents either a *left-arrow* is one of the following forms:

<===, **<---**, **<==**, **<--**, **<=**, **<-**, or **<**

or a *right-arrow* is one of the following forms:

===>, **--->**, **==>**, **-->**, **=>**, **->**, or **>**

In each single-headed path, you specify two lists of variables: *var-list* and *var-list2*. Depending on the direction of the *arrow* specification, one group of variables contains the outcome variables and the other group contains the predictor variables. Optionally, you can specify the *parameter-spec* at the end of each path entry. You can specify the following five types of the parameters for the path entries:

- unnamed free parameters
- initial values
- fixed values
- free parameters with names provided
- free parameters with names and initial values provided

For example, in the following statement you specify a model with five paths:

```
path
   v1  <===  f1   ,
   v2  <===  f1   = (0.5),
   v3  <===  f1   =  1.,
   v4  <===  f1   =  b1,
   v5  <===  f1   =  b2 (.4);
```

The first path entry specifies a path from f1 to v1. The effect of f1 (or the path coefficient) on v1 is an unnamed free parameter. For this path effect parameter, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1). The second path entry specifies a path from f1 to v2. The effect of f1 is also an unnamed free parameter with an initial estimate of 0.5. PROC CALIS also generates a parameter name for effect parameter. The third path entry specifies a path from f1 to v3. The effect of f1 is also a fixed value of 1.0. This value stays the same in the model estimation. The fourth path entry specifies a path from f1 to v4. The effect of f1 is a free parameter named b1. The fifth path entry specifies a path from f1 to v5. The effect of f1 is a free parameter named b2, with an initial value of 0.4.

You can specify multiple variables in the *var-list* and *var-list2* lists. For example, the following statement specifies five paths from f1 to v1–v5:

```
path
   f1 ===> v1-v5;
```

All the five effects of f1 on the five variables are unnamed free parameters. If both *var-list* and *var-list2* lists contain multiple variables, you must be careful about the order of the variables when you also specify parameters at the end of the path entry. For example, the following statement specifies the paths from the predictor variables x1–x2 to the outcome variables y1–y3:

```
path
   y1-y3 <=== x1-x2   = a1-a6;
```

The PATH statement specifies six paths in the path entry. These six paths have effect parameters a1–a6. This specification is equivalent to the following specification:

```
path
   y1 <=== x1   = a1;
   y1 <=== x2   = a2;
   y2 <=== x1   = a3;
   y2 <=== x2   = a4;
   y3 <=== x1   = a5;
   y3 <=== x2   = a6;
```

The following statement shows another example of multiple-path specification:

```
path
   x1-x2 ===> y1-y3   = b1-b6;
```

This specification is equivalent to the following specification with separate path specifications:

```
path
   x1 ===> y1   = b1;
   x1 ===> y2   = b2;
   x2 ===> y3   = b3;
   x2 ===> y1   = b4;
   x2 ===> y2   = b5;
   x2 ===> y3   = b6;
```

You can also specify parameter with mixed types in any path entry, as shown in the following specification:

```
path
   f1 ===> y1-y3    = 1.  b1(.5) (.3),
   f2 ===> y4-y6    = 1.  b2  b3(.7);
```

This specification is equivalent to the following expanded version:

```
path
   f1 ===> y1    = 1.,
   f1 ===> y2    = b1(.5),
   f1 ===> y3    = (.3),
   f2 ===> y4    = 1.,
   f2 ===> y5    = b2,
   f2 ===> y6    = b3(.7);
```

Notice that in the original specification with multiple-path entries, 0.5 is interpreted as the initial value for the parameter b1, but not as the initial estimate for the path from f1 to y3. In general, an initial value that follows a parameter name is associated with the free parameter.

If you indeed want to specify that b1 is a free parameter *without* an initial estimate and 0.5 is the initial estimate for the path from f1 to y3 (while keeping all other specification the same), you can use a null initial value specification, as shown in the following statement:

```
path
   f1 ===> y1-y3    = 1.  b1() (.5) ,
   f2 ===> y4-y6    = 1.  b2  b3(.7);
```

This way 0.5 becomes the initial value for the path from f1 to y3. Because a parameter list with mixed types might be confusing, you can break down the specifications into separate path entries to remove ambiguities. For example, you can use the following specification equivalently:

```
path
   f1 ===> y1        = 1.,
   f1 ===> y2        = b1,
   f1 ===> y3        = (.5) ,
   f2 ===> y4-y6     = 1.  b2  b3(.7);
```

The equal signs in the path entries are optional when the parameter lists do not start with a parameter name. For example, the preceding specification is the same as the following specification:

```
path
   f1 ===> y1          1.,
   f1 ===> y2        = b1,
   f1 ===> y3          (.5) ,
   f2 ===> y4-y6       1.  b2  b3(.7);
```

Notice that in the second path entry, you must retain the equal sign because b1 is a parameter name. Omitting the equal sign makes the specification erroneous because b1 is treated as a variable. This might cause serious estimation problems. Omitting the equal signs might be cosmetically appealing in specifying fixed values or initial values (for example, the first and the third path entries). However, the gain of doing that is not much as compared to the clarity of specification that results from using the equal signs consistently.

NOTE: You do not need to specify single-headed paths from the errors or disturbances (that is, error terms) in the PATH model specification, even though the functional relationships between variables are not assumed to be perfect. Essentially, the roles of error terms in the PATH model are in effect represented by the associated default error variances of the endogenous variables, making it unnecessary to specify any single-headed paths from error or disturbance variables.

The Double-Headed Path Syntax That Uses Two Variable Lists for Specifying Variances and Covariances

var-list two-headed-arrow var-list2 < = parameter-spec >

where a *two-headed-arrow* is one of the following forms:

<==>, **<-->**, **<=>**, **<->**, or **<>**

This syntax enables you to specify covariances between the variables in *var-list* and the variables in *var-list2*. Consider the following example:

```
path
   v1     <==>  v2,
   v3 v4  <==>  v5 v6 v7 = cv1-cv6;
```

The first double-headed path specifies the covariance between v1 and v2 as an unnamed free parameter. PROC CALIS generates a name for this unnamed free parameter. The second double-headed path specifies six covariances with parameters named cv1–cv6. This multiple-covariance specification is equivalent to the following elementwise covariance specification:

```
path
   v3  <==>  v5    = cv1,
   v3  <==>  v6    = cv2,
   v3  <==>  v7    = cv3,
   v4  <==>  v5    = cv4,
   v4  <==>  v6    = cv5,
```

```
   v4  <==>  v7    = cv6;
```

Note that the order of variables in the list is important for determining the assignment of the parameters in the *parameter-spec* list.

If the same variable appears in both of the *var-list* and *var-list2* lists, the "covariance" specification becomes a variance specification for that variable. For example, the following statement specifies two variances:

```
path
   v1     <==>  v1    = 1.0,
   v2     <==>  v2 v3 = sigma2  cv23;
```

The first double-headed path entry specifies the variance of v1 as a fixed value of 1.0. The second double-headed path entry specifies the variance of v2 as a free parameter named sigma2, and then the covariance between v2 and v3 as a free parameter named cv23.

It results in an error if you attempt to use this syntax to specify the variance and covariances among a set of variables. For example, suppose you intend to specify the variances and covariances among v1–v3 as unnamed free parameters by the following statement:

```
path
   v1-v3  <==>  v1-v3 ;
```

This specification expands to the following elementwise specification:

```
path
   v1  <==>  v1 ,
   v1  <==>  v2 ,
   v1  <==>  v3 ,
   v2  <==>  v1 ,
   v2  <==>  v2 ,
   v2  <==>  v3 ,
   v3  <==>  v1 ,
   v3  <==>  v2 ,
   v3  <==>  v3 ;
```

There are nine variance or covariance specifications, but all of the covariances are specified twice. This is treated as a duplication error. The correct way is to specify only the nonredundant covariances, as shown in the following elementwise specification:

```
path
   v1  <==>  v1 ,
   v2  <==>  v1 ,
   v2  <==>  v2 ,
   v3  <==>  v1 ,
   v3  <==>  v2 ,
   v3  <==>  v3 ;
```

However, the elementwise specification is quite tedious when the number of variables is large. Fortunately, there is another syntax for double-headed paths to deal with this situation. This syntax is discussed next.

The Double-Headed Path Syntax That Uses a Single Variable List for Specifying Variances

two-headed-arrow var-list < = parameter-spec >

This syntax enables you to specify variances among the variables in *var-list*. Consider the following example:

```
path
   <==>  v1    = (0.8),
   <==>  v2-v4 ;
```

The first double-headed path entry specifies the variance of v1 as an unnamed free parameter with an initial estimate of 0.8. The second double-headed path entry specifies the variances of v2–v4 as unnamed free parameters. No initial values are given for these three variances. PROC CALIS generates names for all these variance parameters. You can specify these variances equivalently by the elementwise covariance specification syntax, as shown in the following, but former syntax is much more efficient.

```
path
   v1 <==>  v1   = (0.8),
   v2 <==>  v2 ,
   v3 <==>  v3 ,
   v4 <==>  v4 ;
```

The Double-Headed Path Syntax That Uses a Single Variable List for Specifying Variances and Covariances

two-headed-arrow **[***var-list***]** *< = parameter-spec >*

This syntax enables you to specify all the variances and covariances among the variables in *var-list*. For example,the following statement specifies all the variances and covariances among v2–v4:

```
path
    <==> [v2-v4]  = 1.0  cv32  cv33(0.5) cv42 .7 cv44;
```

This specification is more efficient as compared with the following equivalent specification with elementwise variance or covariance definitions:

```
path
    v2 <==> v2    = 1.0,
    v3 <==> v2    = cv32 ,
    v3 <==> v3    = cv33(0.5),
    v4 <==> v2    = cv42,
    v4 <==> v3    = .7,
    v4 <==> v2    = cv44;
```

The Double-Headed Path Syntax for Specifying Nonredundant Covariances

two-headed-arrow **(***var-list***)***< = parameter-spec >*

This syntax enables you to specify all the nonredundant covariances among the variables in *var-list*. For example, the following statement specifies all the nonredundant covariances between v2–v4:

```
path
    <==> (v2-v5)  = cv1-cv6;
```

This specification is equivalent to the following elementwise specification:

```
path
    v3 <==> v2    = cv1 ,
    v4 <==> v2    = cv2 ,
    v4 <==> v3    = cv3 ,
    v5 <==> v2    = cv4 ,
    v5 <==> v3    = cv5 ,
    v5 <==> v4    = cv6 ;
```

The 1-Path Syntax for Specifying Means and Intercepts

1 *right-arrow var-list* < = *parameter-spec* >

where a *right-arrow* is one of the following forms:

===>, **--->**, **==>**, **-->**, **=>**, **->**, or **>**

This syntax enables you to specify the means or intercepts of the variables in *var-list* as paths from the constant **1**. Consider the following example:

```
path
   v1 <===  v2-v4,
   1  ===>  v1     =  alpha,
   1  ===>  v2-v4  =  3*kappa;
```

The first single-headed path specifies that v1 is predicted by variables v2, v3, and v4. Next, the first 1-path entry specifies either the intercept of v1 as a free parameter named alpha. It is the intercept, rather than the mean, of v1 because endogenous in the PATH model. The second 1-path entry specifies the means of v2–v4 as constrained parameters. All these means or intercepts are named kappa so that they have the same estimate.

Therefore, whether the parameter is a mean or an intercept specified with the 1-path syntax depends on whether the associated variable is endogenous or exogenous in the model. The intercept is specified if the variable is endogenous. Otherwise, the mean of the variable is specified. Fortunately, any variable in the model can have either a mean or intercept (but not both) to specify. Therefore, the 1-path syntax is applicable to either the mean or intercept specification without causing conflicts.

Shorter and Longer Parameter Lists

If you provide fewer parameters in *parameter-spec* than the number of paths in a *path* entry, all the remaining parameters are treated as unnamed free parameters. For example, the following specification specifies the free parameter beta to the first path and assigns unnamed free parameters to the remaining four paths:

```
path
   f1 ===> y1 z1 z2 z3 z4  = beta;
```

This specification is equivalent to the following specification:

```
path
   f1 ===> y1  = beta,
   f1 ===> z1 z2 z3 z4;
```

If you intend to fill up all values with the last parameter specification in the list, you can use the continuation syntax **[...]**, **[..]**, or **[.]**, as shown in the following example:

```
path
   f1 ===> y1 z1 z2 z3 z4  = beta gamma [...];
```

This specification is equivalent to the following specification:

```
path
   f1 ===> y1 z1 z2 z3 z4  = beta 4*gamma;
```

The repetition factor **4*** means that gamma repeats 4 times.

However, you must be careful not to provide too many parameters. For example, the following specification results in an error:

```
path
   SES_Factor ===> y1 z1 z2 z3 z4  = beta gamma1-gamma6;
```

Because there are only five paths in the specification, parameters gamma5 and gamma6 are excessive.

Default Parameters

It is important to understand the default parameters in the PATH model. First, knowing which parameters are default free parameters makes your specification more efficient by omitting the specifications of those parameters that can be set by default. For example, because all variances and covariances among exogenous variables (excluding error terms) are free parameters by default, you do not need to specify them in the PATH model if these variances and covariances are not constrained. Second, knowing which parameters are default fixed zero parameters enables you to specify your model accurately. For example, because all error covariances in the PATH model are fixed zeros by default, you must use the PCOV statement or the double-headed path syntax to specify the partial (error) covariances among the endogenous variables if you want to fit a model with correlated errors. See the section "Default Parameters in the PATH Model" on page 1768 for details about the default parameters of the PATH model.

Modifying a PATH Model from a Reference Model

If you define a new model by using a reference (old) model in the REFMODEL statement, you might want to modify some path specifications from the PATH statement of the reference model before transferring the specifications to the new model. To change a particular path specification from the reference model, you can simply respecify the same path with the desired parameter specification in the PATH statement of the new model. To delete a particular path and its associated parameter from the reference model, you can specify the desired path with a missing value specification in the PATH statement of the new model.

PATH *path* < **,** *path* . . . > **;**
  PVAR *partial-variance-parameters* **;**
  PCOV *partial-covariance-parameters* **;**
  MEAN *mean-parameters* **;**

The new model is formed by integrating with the old model in the following ways:

Duplication: If you do not specify in the new model a parameter location that exists in the old model, the old parameter specification is duplicated in the new model.

Addition: If you specify in the new model a parameter location that does not exist in the old model, the new parameter specification is used in the new model.

Deletion: If you specify in the new model a parameter location that also exists in the old model and the new parameter is denoted by the missing value '.', the old parameter specification is not copied into the new model.

Replacement: If you specify in the new model a parameter location that also exists in the old model and the new parameter is not denoted by the missing value '.', the new parameter specification replaces the old one in the new model.

For example, consider the following specification of a two-group analysis:

```
proc calis;
   group 1 / data=d1;
   group 2 / data=d2;
   model 1 / group=1;
      path
         v1 <=== f1   = 1.,
         v2 <=== f1   = load1,
         v3 <=== f1   = load2,
         f1 <=== v4   = b1,
         f1 <=== v5   = b2,
         f1 <=== v6   = b3;
      pvar
         E1-E3    = ve1-ve3,
         f1       = vd1,
         v5-v6    = phi4-phi6;
      pcov
         v1 v2    = cve12;
   model 2 / group=2;
      refmodel 1;
      path
         v3 <=== f1   = load1,
      pcov
         v1 v2    =  .,
         v2 v3    = cve23;
run;
```

You specify Model 2 by referring to Model 1 in the REFMODEL statement. Model 2 is the new model that refers to the old model, Model 1. This example illustrates the four types of model integration rules for the new model:

- Duplication: All parameter specifications, except for the partial covariance between v1 and v2 and the **v3 <=== f1** path in the old model, are duplicated in the new model.
- Addition: The parameter cve23 for the partial covariance between v2 and v3 is added in the new model because there is no corresponding specification in the old model.
- Deletion: The specification of partial covariance between v1 and v2 in the old model is not copied into the new model, as indicated by the missing value '.' specified in the new model.
- Replacement: The new path **v3 <=== f1** replaces the same path in the old model with parameter load1 for the path coefficient. Thus, in the new model paths **v3 <=== f1** and **v2 <=== f1** are now constrained to have the same path coefficient parameter load1.

PATHDIAGRAM Statement

PATHDIAGRAM < *options* > **;**

You can use the PATHDIAGRAM statement to select particular path diagrams for output, to specify and modify the layout algorithm, to select variables and paths in path diagrams, to control the formatting of parameter labels, and to fine-tune many graphical and nongraphical features of path diagrams. You can use multiple PATHDIAGRAM statements to control different path diagrams or to produce path diagrams from the same model that have different styles and graphical features.

If you specify a PATHDIAGRAM statement without any *options*, this has the same effect as specifying the PLOTS=PATHDIAGRAM option in the PROC CALIS statement. Both produce a path diagram for the unstandardized solution if you do not specify the METHOD=NONE option in the PROC CALIS statement. By default, the path diagram for the unstandardized solution shows the paths, variables, unstandardized estimates and their significance, and fit summary table. For more information about these default settings, see the section "Default Path Diagram Settings" on page 1796. If you specify METHOD=NONE in the PROC CALIS statement, PROC CALIS produces a path diagram for the initial specifications, which shows the paths, the variables, and the input fixed values and parameter names. For introductory examples of creating and editing path diagrams, see Yung (2014).

The *options* of the PATHDIAGRAM statement can be classified into five categories. The following five tables summarize these *options*. An alphabetical listing of these *options* that includes more details follows the tables.

Options for Selecting Different Types of Path Diagrams

You can use the *options* in the following table to request various types of path diagrams. If you use more than one of these *options*, PROC CALIS outputs a combination of the specified types of path diagrams.

Option	Description
DIAGRAM=	Specifies the solution types for the output path diagrams
MODEL=	Specifies the models for the output path diagrams
STRUCTURAL	Requests the path diagrams for the structural components of models

Options for Controlling the Layout of Paths and Variables

You can use the following *options* to specify the path diagram layout algorithm, to define paths and variables that you want to include in or exclude from path diagrams, and to provide useful information that might improve the graphical presentation of the output path diagrams.

Option	Description
ARRANGE=	Specifies the graphical algorithm for laying out variables
DESTROYER=	Specifies paths that should not be considered in laying out variables
EMPHSTRUCT	Emphasizes the structural components in path diagrams
ERRORSIZE=	Specifies the size of error variables relative to observed variables
FACTORSIZE=	Specifies the size of latent factors relative to observed variables
MEANPARM=	Specifies mean parameters to be displayed as either paths or labels
OMITPATH=	Omits specified paths from the output path diagram
SCALE=	Specifies the scale factor for the node size
STRUCTADD=	Adds observed variables to the definition of the structural component
USEERROR	Requests the use of explicit error variables
VARPARM=	Specifies variance parameters to be displayed as either paths or labels

Options for Selecting the Parameters to Display in the Output Path Diagram

You can use the following *options* to select the parameters to display in output path diagrams.

Option	Description
EXOGCOV	Enables the display of covariances between exogenous non-error variables
NOCOV	Disables the display of covariance parameters
NOERRCOV	Disables the display of covariances between error variables
NOERRVAR	Disables the display of variances of error variables
NOEXOGCOV	Disables the display of covariances between exogenous non-error variables
NOEXOGVAR	Disables the display of variances between exogenous non-error variables
NOINITPARM	Disables the display of parameters in the path diagram for the initial specification
NOMEAN	Disables the display of means or intercepts of all variables
NOVARIANCE	Disables the display of variances of all variables

Options for Formatting the Parameters in the Output Path Diagram

You can use the following *options* to control the formats of the parameters that are displayed in path diagrams.

Option	Description
DECP=	Specifies number of the decimal places in the estimates
NOESTIM	Disables the display of numerical estimates
NOFLAG	Disables the display of flags for statistical significance
PARMNAMES	Enables the display of parameter names or labels

Miscellaneous Options for Controlling the Output Path Diagram

You can use the following *options* to control the displays of the title, the fit table, and the labels of the variables.

Option	Description
DECPFIT=	Specifies the number of decimal places in the fit statistics
DESIGNHEIGHT=	Sets the height, in pixels, of the output path diagram
DESIGNWIDTH=	Sets the width, in pixels, of the output path diagram
DIAGRAMLABEL=	Specifies the label of the diagram in the ODS
FITINDEX=	Selects fit information to display in the fit summary table
LABEL=	Specifies labels of the variables shown in the path diagram
NOFITTABLE	Suppresses the display of fit statistics
NOTITLE	Suppresses the display of the title
TEXTSIZEMIN=	Specifies the minimum text size in the output path diagram
TITLE=	Specifies the title to display in the output path diagram

In order to customize your path diagram output effectively by using these *options*, it is useful to know the default settings of the graphical and nongraphical elements in the output path diagram. For information about these default settings and the corresponding overriding or modifying *options*, see the section "Default Path Diagram Settings" on page 1796.

PATHDIAGRAM Statement Options

ARRANGE=*name*

ARRANGEMENT=*name*

METHOD=*name*

specifies the algorithm for laying out the variables in the path diagram. You can specify the following *names*:

AUTOMATIC requests the "best" algorithm. PROC CALIS analyzes the interrelationships among variables (excluding error variables) in the model and selects the most appropriate algorithm from those that you can explicitly request by specifying the FLOW, GRIP, and GROUPEDFLOW options. PROC CALIS first checks whether the ideal conditions for the process-flow algorithm are met; if so, the process-flow algorithm is used. If not, PROC CALIS checks whether the ideal conditions for the grouped-flow algorithm are met; if so, the grouped-flow algorithm is used. Otherwise, the more general GRIP algorithm is used.

FLOW requests the process-flow algorithm, which is most appropriate when all variables (not including the error variables) exhibit hierarchical functional relationships in the model. That is, all variables in the model can be ordered hierarchically such that functional relationships (or directional paths) can occur only between variables at adjacent levels (that is, no cross-level, within-level, or reciprocal functional relationships). For example, the process-flow algorithm is ideal for confirmatory factor models and higher-order factor models in their pure forms (no cross-level or within-level paths in the model).

GRIP requests the GRIP (Graph dRawing with Intelligent Placement) algorithm. This algorithm is more general than the process-flow and grouped-flow algorithms. Hence, the GRIP algorithm should be used when the ideal conditions for the process-flow and grouped-flow algorithms are not met.

GROUPEDFLOW requests the grouped-flow algorithm, which is most appropriate when all latent factors exhibit hierarchical functional relationships in the model. That is, all latent

factors can be ordered hierarchically such that their functional relationships (or directional paths) occur only between factors at adjacent levels (no cross-level, within-level, or reciprocal functional relationships). The ideal conditions for the grouped-flow algorithm are the same as those for the process-flow algorithm, except that the grouped-flow algorithm considers only the latent factors that have an ideal process-flow pattern. Because each latent factor is usually clustered with a group of measured variables, this pattern can be described as an ideal "grouped" process-flow pattern (hence the name grouped-flow algorithm).

By default, ARRANGE=AUTOMATIC.

For more information and for illustrations of these methods, see the section "The Process-Flow, Grouped-Flow, and GRIP Layout Algorithms" on page 1784.

DECP=*i*

sets the decimal places of the estimates that are displayed in the path diagram, where *i* is between 0 and 4. The default value is 2. The displayed estimates are at most seven digits long, including the decimal point for the nonzero value of *i*.

DECPFIT=*i*

sets the decimal places of the fit statistics or information that is shown in the fit summary table of the path diagram, where *i* is between 0 and 4. The default value is 2. The displayed numerical values are at most 10 digits long, including the decimal point for the nonzero value of *i*.

DESIGNHEIGHT=*i*

DH=*i*

sets the height of the path diagram, in number of pixels, where *i* is between 100 and 10,922. The default height is 600 pixels. Typically, you might want to set a larger design height and width when your path diagram contains more nodes or variables.

DESIGNWIDTH=*i*

D=*i*

sets the width of the path diagram, in number of pixels, where *i* is between 100 and 10,922. The default width is 720 with the fit summary table or 600 without the fit summary table. Typically, you might want to set a larger design width and height when your path diagram contains more nodes or variables.

DESTROYER=[*path* < **,** *path* . . . >**]** | **{***path* < **,** *path* . . . >**}**

DESTROYERPATH=[*path* < **,** *path* . . . >**]** | **{***path* < **,** *path* . . . >**}**

specifies a list of paths that are considered to be "destroyer" paths to the layout algorithm that is used (or specified using the ARRANGE= option), where *path* represents

var-list direction var-list2

and *direction* is the direction of the path, as indicated by one of the following:
===>, **--->**, **==>**, **-->**, **=>**, **->**, **>**, **<===**, **<---**, **<==**, **<--**, **<=**, **<-**, **<**, **<==>**, **<-->**, **<=>**, **<->**, or **<>**

For example:

```
pathdiagram destroyer=[x1 ===> x2, x2 <=== x5];
pathdiagram destroyer=[x1 x5 ===> x2];
```

Note that the two preceding statements specify the same set of destroyer paths: "**x1 ===> x2**" and "**x5 ===> x2**."

Destroyer paths are shown in the path diagram, but they are not used in determining the layout of variables. Destroyer paths are paths that violate the ideal conditions for a particular layout algorithm so that the placement of variables in the path diagram cannot take advantage of that algorithm (especially for the process-flow or grouped-flow algorithm). To counter the violations, PROC CALIS ignores the destroyer paths when laying out the variables. After determining the locations of all variables, PROC CALIS adds the destroyer paths back to the path diagram. If you can identify these destroyer paths and you have only a few of them (for example, fewer than five), specifying these destroyer paths in the DESTROYER= option can significantly improve the path diagram. However, if you have too many destroyer paths, this option might not be effective.

For more information and for illustrations, see the section "Handling Destroyers in Path Diagrams" on page 1792.

NOTE: If a path in the model serves as the only directed path that connects to a particular variable, it is generally not advisable to apply the DESTROYER= option to that path. During the layout process, the DESTROYER= option disconnects this variable from the rest of the variables in the model, so the location of this disconnected variable in the path diagram is arbitrary. This might lead to undesirable graphical results when PROC CALIS adds the destroyer paths back to the path diagram.

DIAGRAM=*name* | **[***names***]** | **{***names***}**
SOLUTION=*name* | **[***names***]** | **{***names***}**

specifies the solution types for the path diagram output. You can specify the following *names*:

ALL requests separate path diagrams for the initial, unstandardized, and standardized solutions.

INITIAL | **INIT** requests a path diagram for the initial solution. This diagram displays the fixed values and the parameters that you specify for the model. However, it does not display generated parameter names or initial estimates. To produce a "bare-bones" path diagram that shows only the variables and their interrelationships, use the NOINITPARM option, which suppresses the display of fixed values and the parameters.

STANDARD | **STAND** requests a path diagram for the standardized solution. This diagram displays the standardized parameter estimates and their significance in paths or as labels of variables. By default, it also displays the fit summary table.

UNSTANDARD | **UNSTAND** requests a path diagram for the unstandardized solution. This diagram displays the unstandardized parameter estimates and their significance in paths or as labels of variables. By default, it also displays the fit summary table.

For example, to display only the path diagram for the standardized solutions, you can use the following statement:

```
pathdiagram diagram=standard;
```

To display the path diagrams for the initial and unstandardized solution, you can use the following statement:

```
pathdiagram diagram=[initial unstandard];
```

DIAGRAMLABEL=*name*

DLABEL=*name*

specifies the label of the path diagram. You can use any valid SAS names or quoted strings up to 256 characters for *name*. However, only up to 40 characters of the label are used by ODS. If you do not specify this option, PROC CALIS uses the *name* provided in the TITLE= option. If you specify neither the DIAGRAMLABEL= nor TITLE= option, PROC CALIS generates a label for the path diagram. The generated label reflects the model number (if provided in the MODEL statement), the solution type (initial, unstandardized, or standardized), and whether the structural model is being shown. For example:

```
pathdiagram diagramlabel=MySpecialModel;
pathdiagram diagramlabel="The best Model";
```

Note that if you specify multiple path diagrams in the same PATHDIAGRAM statement, PROC CALIS applies the same label to all requested path diagrams. If unique labels are preferred, you can use separate PATHDIAGRAM statements to specify labels for different path diagrams.

EMPHSTRUCT< =*i* >

requests that the structural component of the model be emphasized in the path diagram, where *i* is any number between 0.2 and 5. The variables in the structural component are called structural variables. Usually, only latent factors are considered as structural variables. For this option, the structural variables have relatively larger sizes than other variables (approximately four times as large as the observed variables in each dimension). You can control the relative size by providing a suitable value for *i*, which is 4 by default. The corresponding path diagram displays and labels only the structural variables. Nonstructural variables are displayed but not labeled. In addition, the diagram displays and labels paths among structural variables. The diagram displays but does not label paths among structural and nonstructural variables. Finally, the diagram does not display or label paths among nonstructural variables.

If you consider some observed variables as structural variables in your model, use the STRUCTADD= option to include these observed variables in the structural component. This option is not applicable to the path diagram for the structural model that you request by specifying the STRUCTURAL option, which displays only the structural component of the model. In contrast, the EMPHSTRUCT option produces the complete model but emphasizes the structural component.

For more information and for illustrations, see the section "Showing or Emphasizing the Structural Components" on page 1798.

ERRORSIZE=*size*

ERRSIZE=*size*

specifies the size of error variables relative to that of observed variables, where *size* is between 0.2 and 5. The default value is 0.5, meaning that the size of error variables is about half that of observed variables.

EXOGCOV

EXOGCOVARIANCE

requests that the path diagram show the double-headed paths that represent the covariances among exogenous non-error variables. By default, these double-headed paths are displayed only for exploratory or confirmatory factor models, which you specify by using the FACTOR statement for other types of models.

FACTORSIZE=*size*

FACTSIZE=*size*

specifies the size of latent factors relative to that of observed variables, where *size* is between 0.2 and 5. The default value is 1.5, meaning that the size ratio of factors to observed variables is about 3 to 2.

FITINDEX=[*names***] | {***names***}**

defines fit statistics or information in *names* to display in the fit summary table, which is shown along with the path diagrams for unstandardized and standardized solutions. PROC CALIS uses the order of the fit statistics or information specified in the FITINDEX= option to display the information in the fit summary table.

For example:

```
pathdiagram fitindex=[chisq df probchi srmr rmsea];
pathdiagram fitindex=all;
pathdiagram fitindex=[default aic sbc caic];
```

For the default list of fit statistics and information, see the FITINDEX=DEFAULT option.

You can use the following *names* to refer to all or individual fit statistics or information available in the fit summary table.

ALL displays all available fit statistics or information. If you specify the ALL option along with other specific options for individual fit statistics or information, PROC CALIS displays the specific fit statistics or information in the fit summary table first, followed by the remaining available fit statistics or information.

DEFAULT displays a default set of fit statistics or information, which is the same as specifying the following in the FITINDEX= option: AGFI, CFI, CHISQ, DF, LL_RMSEA, LOGLIKE, PROBCHI, PROBCLFIT, RMSEA, SRMR, and UL_RMSEA.

If you specify the DEFAULT option along with other specific options for individual fit statistics or information, PROC CALIS displays the specific fit statistics or information in the fit summary table first, followed by the remaining available default fit statistics or information.

AGFI displays the adjusted GFI.

AIC displays the Akaike information criterion.

CAIC displays Bozdogan's corrected AIC.

CFI | BENTLERCFI displays Bentler's comparative fit index.

CHISQ displays the chi-square statistic for model fit.

CN | CRITICAL_N displays Hoelter's critical N.

DF displays the degrees of freedom for the chi-square test for model fit.

ECVI displays the expected cross-validation index.

GFI displays the goodness-of-fit index by Jöreskog and Sörbom.

LL_ECVI | ECVI_LL displays the lower confidence limit for RMSEA.

LL_RMSEA | RMSEA_LL displays the lower confidence limit for RMSEA.

LOGLIKE displays the fitted model –2 log-likelihood function value for METHOD=FIML only.

NIOBS displays the number of incomplete observations for METHOD=FIML.

NOBS displays the number of observations that are used in the analysis.

NPARMS | NPARM displays the number of independent parameters.

PGFI displays the parsimonious GFI.

PROBCHI | PROBCHISQ displays the *p*-value of the chi-square statistic for model fit.

PROBCLFIT displays the probability of close fit.

PROBSBCHI | PROBSBCHISQ displays the *p*-value of the Satorra-Bentler scaled chi-square for model fit.

RMR displays the root mean square residual.

RMSEA displays the root mean square error of approximation.

SBC displays the Schwarz Bayesian criterion.

SBCHISQ displays the Satorra-Bentler scaled chi-square for model fit.

SRMR displays the standardized root mean square residual.

UL_ECVI | ECVI_UL displays the upper confidence limit for ECVI.

UL_RMSEA | RMSEA_UL displays the upper confidence limit for RMSEA.

LABEL= [*varlabel* < **,** *varlabel . . .* >**] | {***varlabel* < **,** *varlabel . . .* >**}**
specifies the labels of variables to be displayed in path diagrams, where each *varlabel* is in the form

variable **=** *label*

You can use any valid SAS names or quoted strings up to 256 characters for *labels*. The *labels* are used to label the corresponding variables in output path diagrams. If you do not specify labels, the original variable names are used as labels.

For example, instead of using `x1` and `y1` to label the variables in the path diagram, the following statement specifies more meaningful labels:

```
pathdiagram label=[x1="Start Use" y1="Spending"];
```

Note that PROC CALIS does not currently use the variable labels from the LABEL statement for the path diagram.

MEANPARM=PATH | LABEL

MEAN=PATH | LABEL

specifies whether mean parameters are displayed as paths (PATH) or as labels that are attached to variables (LABEL). The default MEANPARM= value is LABEL when you model mean structures. This option does not apply when you model only covariance structures.

MODEL=[*int-list***] | {***int-list***}**

MODELS=[*int-list***] | {***int-list***}**

requests path diagrams for a list of models, which are specified by their associated model numbers. By default, the output shows path diagrams of all models in the analysis. This option is useful if you want to restrict the path diagram output to a particular set of models. For example:

```
pathdiagram model=[1 to 3];
pathdiagram model=[2 4 5];
```

The first PATHDIAGRAM statement requests path diagrams for models 1, 2, and 3. The second PATHDIAGRAM statement requests path diagrams for models 2, 4, and 5.

NOCOV

NOCOVARIANCE

suppresses the display of covariances between variables.

NOERRCOV

NOERRORCOVARIANCE

suppresses the default display of covariances among error variables.

NOERRVAR

NOERRORVARIANCE

suppresses the default display of error variances, which are represented as either double-headed paths or labels that are attached to error variables.

NOESTIM

NOEST

suppresses the default display of all numerical estimates (including fixed estimates) in path diagrams for unstandardized and standardized solutions.

NOEXOGCOV

NOEXOGCOVARIANCE

suppresses the display of covariances between exogenous non-error variables. By default, only the exploratory or confirmatory factor models, which you specify using the FACTOR statement, show the covariances between exogenous non-error variables. For other models, NOEXOGCOV is the default.

NOEXOGVAR

NOEXOGVARIANCE

suppresses the default display of variances of exogenous non-error variables. This applies to variance parameters that are represented as either double-headed paths or labels that are attached to the exogenous variables.

NOFITTABLE

NOFIT

suppresses the default display of fit summary tables in path diagrams for standardized or unstandardized solutions.

NOFLAG

suppresses the default flagging of significant estimates in path diagrams. By default, estimates that are significant at the 0.05 α-level are flagged with "*", and estimates that are also significant at the 0.01 α-level are flagged with "**". Fixed estimates are marked with "(fixed)".

NOINITPARM

suppresses the default display of user-specified parameter names and fixed values in path diagrams for initial specifications, which you request by specifying the DIAGRAM=INITIAL option in the PATHDIAGRAM statement. This option is not applicable to path diagrams for unstandardized or standardized solutions.

NOMEAN

suppresses the default display of the mean or intercept parameters and their estimates in models that contain mean structures. These mean parameters and estimates can be in the form of either paths or labels that are attached to variables.

NOTITLE

suppresses the display of the default title. You can use the TITLE= option to provide your own title.

NOVARIANCE

suppresses the default display of all variances. This applies to variance parameters that are represented as either double-headed paths or labels that are attached to nodes.

OMITPATH=[*path* < **,** *path . . .* > **] | {***path* < **,** *path . . .* > **}**

OMIT=[*path* < **,** *path . . .* > **] | {***path* < **,** *path . . .* > **}**

specifies a list of *paths* to be omitted from the output path diagram, where *path* represents

var-list direction var-list2

and *direction* is the direction of the path, as indicated by one of the following:
===>, **--->**, **==>**, **-->**, **=>**, **->**, **>**, **<===**, **<---**, **<==**, **<--**, **<=**, **<-**, **<**, **<==>**, **<-->**, **<=>**, **<->**, or **<>**

For example:

```
pathdiagram omitpath=[y1 ===> y4, y4 <=== y3];
pathdiagram omitpath=[y3 y1 ===> y4];
```

Note that the two preceding statements specify the same set of paths that are omitted: "**`y1 ===> y4`**" and "**`y3 ===> y4`**."

The omitted paths are not shown in the path diagram, nor are they used in determining the layout of the variables. The OMITPATH= option is useful when you want to see how a particular set of paths affects the display of a path diagram. If omitting a certain set of paths improves the display, the omitted paths can be considered as destroyer paths. You might then specify these paths in the DESTROYER= option to get an improved diagram.

NOTE: If a path in the model serves as the only directed path that connects to a particular variable, it is generally not advisable to apply the OMITPATH= option to that path. The OMITPATH= option disconnects this variable from the rest of the variables in the model, so the location of the disconnected variable in the path diagram is arbitrary. This might lead to undesirable graphical results.

PARMNAMES
PARM

requests the display of parameter names or labels. By default, path diagrams for unstandardized or standardized solutions do not show any parameter names or labels, whereas path diagrams for initial specifications show only user-specified parameter names or labels (but not the generated parameter names or labels) and fixed values.

SCALE=*n*
DIAGRAMSCALE=*n*

specifies the scaling factor, *n*, for the node size relative to the dimensions of the path diagram. Valid values of *n* are between 0 and 6. This option applies to the ARRANGE=GRIP layout only.

PROC CALIS uses certain default pixel dimensions for the nodes in path diagrams that have default design dimensions (see the DESIGNHEIGHT= and DESIGNWIDTH= options for the default design dimensions). The ratio of this node dimension to the design dimension defines the point at which SCALE=1. SCALE= option values greater than 1 enlarge the nodes (relative to the design dimensions). SCALE= option values less than 1 shrink the nodes (relative to the design dimensions).

If you change the default design dimensions by using the DESIGNHEIGHT= and DESIGNWIDTH= options, PROC CALIS automatically adjusts the scaling factor so that it is inversely proportional to the design dimension of the resultant path diagram. That is, if you set a larger (smaller) design dimension for the path diagram, PROC CALIS automatically decreases (increases) the scaling factor. This automatic scaling ensures that the nodes in different design dimensions have approximately the same resolutions (in terms of pixels). It also enables you to accommodate more nodes in path diagrams that have larger design dimensions. If you enforce SCALE=1 for path diagrams that have larger design dimensions, the resulting path diagrams are just like a simple magnification of the path diagram that has the default design dimensions. Such a simple magnification is not desirable if your purpose is to accommodate more nodes in path diagrams.

STRUCTADD=[*variables***] | {***variables***}**

specifies a list of observed *variables* that should be added to the structural component of a model. Traditionally, the structural component of a complete model includes only the latent factors and their interrelationships. However, this definition might be too restrictive in many applications. For the purpose of showing structural components in path diagrams, you can use the STRUCTADD= option to add observed variables to the definition of the structural component. The resulting path diagram for the structural component contains all latent factors, the additional observed variables, and their functional relationships.

For example, the following statement adds the observed variables x1 and x2 to the path diagram for displaying the structural component:

```
pathdiagram struct structadd=[x1 x2];
```

The following statement adds the observed variables x3 and x5 to the structural component for displaying the path diagram that emphasizes the structural component:

```
pathdiagram emphstruct structadd=[x3 x5];
```

For more information and for illustrations, see the section "Expanding the Definition of the Structural Variables" on page 1802.

STRUCTURAL < (ONLY) >

STRUCT < (ONLY) >

requests the path diagram for the so-called structural component of the model, or simply the component model. By default, the output shows this path diagram for the entire model, not just the structural component.

Traditionally, all structural equation models are considered to have two main components. One component is the so-called structural model, which involves latent factors and their functional relationships only. The other is the so-called measurement model, which involves observed variables and latent variables and the functional relationships that connect the observed variables to the latent factors. By specifying the STRUCTURAL option, you request the path diagram for the structural model in addition to the path diagram for the complete model. To display a path diagram for the structural model only, use the STRUCTURAL(ONLY) option.

For example, the following statement produces two path diagrams, one for the complete model and one for the structural component:

```
pathdiagram struct;
```

However, the following statement produces one path diagram for the structural component:

```
pathdiagram struct(only);
```

If you consider some observed variables as structural variables in your model, you can use the STRUCTADD= option to include these observed variables in the structural component.

For more information and for illustrations, see the section "Showing or Emphasizing the Structural Components" on page 1798.

TEXTSIZEMIN=*i*

specifies the minimum text size, *i*, in the output path diagram. Valid values of *i* are between 4 and 40. The default is 10, for the 10-point font size.

Because this value controls only the minimum text size, the text size actually used might be larger, depending on other parameters such as the node scale value (SCALE= option), design height

(DESIGNHEIGHT= option), and design width (DESIGNWIDTH= option). Practically, you use this option when you want to adjust to a larger text size of a default path diagram output. However, if the design space available is limited, the text size might increase only relatively to the node size, which means that the nodes might shrink as a side effect. If this side effect is not desirable, consider setting a higher TEXTSIZEMIN= value simultaneously with higher values in the DESIGNHEIGHT= and DESIGNWIDTH= options.

TITLE=*name*

specifies the title of the path diagram. You can use any valid SAS name or a quoted string of up to 256 characters for *name*. If you do not specify this option, PROC CALIS generates titles for path diagrams. The generated title reflects the model number (if provided in the MODEL statement), the model label (if provided in the MODEL statement), the solution type (initial, unstandardized, or standardized), and whether the structural model is being shown. For example:

```
pathdiagram title=ThisTitleDoesNotUseQuotations;
pathdiagram title="Title looks better with the use of quoted strings";
```

Note that if you specify multiple path diagrams in the same PATHDIAGRAM statement, PROC CALIS applies the same title to all requested path diagrams. If unique titles are preferred, you can use separate PATHDIAGRAM statements to specify titles for different path diagrams.

USEERROR

USEERR

requests that error variables be displayed in the path diagram. By default, PROC CALIS does not display error variables in path diagrams, because showing errors in path diagrams usually creates more clutter. If you prefer to show the error variables explicitly in path diagrams, specify this option.

VARPARM=PATH | LABEL

VARIANCE=PATH | LABEL

specifies whether the variance parameters are displayed as paths (PATH) or as labels that are attached to variables (LABEL). The default VARPARM= value is PATH if your model does not fit mean structures. When you fit a model that contains mean structures, VARPARM= value is set to be the same as the MEANPARM= value.

PCOV Statement

PCOV *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents:

var-list < ∗ *var-list2* > < = *parameter-spec* >

The PCOV statement is a subsidiary model specification statement for the PATH model. You can use the PCOV statement only with the PATH modeling language. The PCOV statement specifies the covariances of exogenous variables, or the error covariances of endogenous variables in the PATH model. It can also specify the covariance between an exogenous variable and the error term of an endogenous variables, although this usage is rare in practice.

In each *assignment* of the COV statement, you specify variables in the *var-list* and the *var-list2* lists, followed by the covariance parameter specification in the *parameter-spec* list. The latter two specifications are optional. The syntax of the PCOV statement is the same as that of the COV statement. See the COV statement on page 1593 for details about specifying within- and between-list (partial) covariances.

The concept behind the PCOV statement is broader than that of the COV statement. The PCOV statement supports the partial covariance parameter specification in addition to the covariance parameter specification, which is the only type of parameter that the COV statement supports. This difference is also reflected from the sets of *var-list* and *var-list2* that you can use in the PCOV statement. In the COV statement, variables on the left-hand side of an *assignment* must be exogenous. However, in the PCOV statement, you can specify both exogenous and endogenous variables. If a pair of variables are both exogenous in a specification, you are defining a covariance parameter between the variables. If a pair of variables are both endogenous in a specification, you are defining a partial covariance parameter between of the variables. This partial covariance is usually interpreted as the error covariance between the two endogenous variables. If one variable is exogenous while the other is endogenous, you are defining a covariance parameter between the exogenous variable and the error term for the endogenous variable.

You can specify the following five types of the parameters for the partial covariances in the PCOV statement:

- an unnamed free parameter
- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

For example, consider a PATH model with exogenous variables x1, x2, and x3 and endogenous variables y4, y5 and y6. The following PCOV statement shows the five types of specifications in five *assignments*:

```
pcov x1 x2 ,
     x1 x3 = (0.5),
     x2 x3 = 2.0,
     y4 y5 = psi1,
     y5 y6 = psi2(0.4);
```

In this statement, the covariance between x1 and x2 is specified as an unnamed free parameter. For this covariance, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1). The covariance between x1 and x3 is an unnamed free parameter with an initial value of 0.5. PROC CALIS also generates a parameter name for this covariance. The covariance between x2 and x3 is a fixed value of 2.0. This value stays the same during the estimation. The error covariance between endogenous variables y4 and y5 is a free parameter named psi1. The error covariance between endogenous variables y5 and y6 is a free parameter named psi2 with an initial value of 0.4.

The syntax of the PCOV statement is the same as the syntax of the COV statement. See the COV statement for more illustrations about the usage.

Default Covariance Parameters

Although the PCOV statement specification is conceptually broader than the COV statement specification, their related default set of covariance parameters is the same—that is, all covariances among *exogenous* manifest and latent variables (excluding error or disturbance variables) are free parameters. Because the PCOV statement applies only to the PATH model, it is easy to understand why the covariances do not apply to the error or disturbance terms. The PATH model, as implemented in PROC CALIS, simply does not use any explicit error or disturbance terms. For the default free covariance parameters, PROC CALIS generate the parameter names with the _Add prefix and appended with unique integer suffixes. You can also use the PCOV statement specification to override these default covariance parameters in situations where you want to set parameter constraints, provide initial or fixed values, or make parameter references.

Another type of default partial covariances are fixed zeros. This default applies to the partial (error) covariances among all *endogenous* variables, and to the partial covariances between all *exogenous* variables and all *endogenous* variables in the path model. Again, you can override the default fixed values by providing explicit specification of these partial or error covariances in the PCOV statement.

Modifying a Covariance or Partial Covariance Parameter Specification from a Reference Model

If you define a new PATH model by using a reference (old) model in the REFMODEL statement, you might want to modify some parameter specifications from the PCOV statement of the reference model before transferring the specifications to the new model. To change a particular partial covariance specification from the reference model, you can simply respecify the same covariance with the desired parameter specification in the PCOV statement of the new model. To delete a particular partial covariance parameter from the reference model, you can specify the desired partial covariance with a missing value specification in the PCOV statement of the new model.

For example, suppose that you are defining a new PATH model by using the REFMODEL statement and that the covariance between variables f1 and v2 is defined as a fixed or free parameter in the reference model. If you do not want this fixed parameter specification to be copied into your new model, you can use the following specification in the new model:

```
pcov f1 v2 = .;
```

Note that the missing value syntax is valid only when you use it with the REFMODEL statement. See the section "Modifying a PATH Model from a Reference Model" on page 1673 for a more detailed example of the PATH model respecification.

As discussed in the section "Default Covariance Parameters" on page 1689, PROC CALIS generates some default free covariance parameters for the PATH model if you do not specify them explicitly in the PCOV statement. When you use the REFMODEL statement for defining a reference model, these default free covariance parameters in the old (reference) model are not transferred to the new model. Instead, the new model generates its own set of default free covariance parameters *after* the new model is resolved from the reference model, the REFMODEL statement options, the RENAMEPARM statement, and the PCOV statement specifications in the new model. This also implies that if you want any of the (partial) covariance parameters to be constrained across the models by means of the REFMODEL specification, you must specify them explicitly in the PCOV statement of the reference model so that the same (partial) covariance specification is transferred to the new model.

PVAR Statement

PVAR *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents:

var-list < **=** *parameter-spec* >

The PVAR statement specifies the variance or error (partial) variance parameters in connection with the confirmatory FACTOR and PATH models.

In each *assignment* of the PVAR statement, you list the *var-list* that you want to specify for their variances or error (partial) variances. Optionally, you can provide a list of parameter specifications (*parameter-spec*) after an equal sign for each *var-list* list. The syntax of the PVAR statement is exactly the same as that of the VARIANCE statement. See the VARIANCE statement on page 1706 for details about the syntax.

The concept behind the PVAR statement is broader than that of the VARIANCE statement. The PVAR statement supports the partial variance parameter specification in addition to the variance parameter specification, which is the only type of parameters that the VARIANCE statement supports. This difference is reflected from the set of *var-list* you can use in the PVAR statement. You can specify both exogenous variables and endogenous variables in the *var-list* list of the PVAR statement, but you can specify only exogenous variables in the *var-list* list of the VARIANCE statement. This conceptualization of the PVAR statement is needed in the FACTOR and PATH modeling languages because error variables are not explicitly defined in these models. You specify the variance of a variable if the variable in the *var-list* list of the PVAR statement is an exogenous (independent) variable in the FACTOR or PATH model. You specify the error (partial) variance of a variable if the variable in the *var-list* list of the PVAR statement is an endogenous (dependent) variable in the FACTOR or PATH model.

You can specify the following five types of the parameters for the partial variances in the PVAR statement:

- an unnamed free parameter
- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

For example, consider a PATH model with exogenous variables x1, x2, and x3 and endogenous variables y4 and y5. The following PVAR statement illustrates the five types of specifications in five *assignments*:

```
pvar x1 ,
     x2 = (2.0),
     x3 = 1.0,
     y4 = psi1,
     y5 = psi2(0.6);
```

In this statement, the variance of x1 is specified as an unnamed free parameter. For this variance, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1). The variance of x2 is an unnamed free parameter with an initial value of 2.0. PROC CALIS also generates a parameter name for this variance. The variance of x3 is a fixed value of 1.0. This value stays the same during the estimation. The error variance of endogenous variable y4 is a free parameter named psi1. The error variance of endogenous variable y5 is a free parameter named psi2 with an initial value of 0.6.

The syntax of the PVAR statement is the same as the syntax of the VARIANCE statement. See the VARIANCE statement for more illustrations about the usage.

Default Partial Variance Parameters

By default, all variances of the *exogenous* manifest and latent variables and all error (partial) variances of the *endogenous* manifest and latent variables are free parameters in the FACTOR or PATH model. For these default free variance parameters, PROC CALIS generates the parameter names with the _Add prefix and appended with unique integer suffixes. You can also use the PVAR statement specification to override these default variance parameters in situations where you want to specify parameter constraints, provide initial or fixed values, or make parameter references.

In the FACTOR or PATH model, a variable can either be exogenous or endogenous. Therefore, the default free parameters covers all the possible variance or partial variance parameters in the model. There are no default fixed zeros for any variances or partial variances in the model.

Modifying a Variance or Partial Variance Parameter Specification from a Reference Model

If you define a new FACTOR or PATH model by using a reference (old) model in the REFMODEL statement, you might want to modify some parameter specifications from the PVAR statement of the reference model before transferring the specifications to the new model. To change a particular variance or partial variance specification from the reference model, you can simply respecify the same variance or partial variance with the desired parameter specification in the PVAR statement of the new model. To delete a particular variance parameter from the reference model, you can specify the desired variance or partial variance with a missing value specification in the PVAR statement of the new model.

For example, suppose that the variance of V1 is specified in the reference PATH model but you do not want this variance specification to be transferred to the new model. You can use the following PVAR statement specification in the new model:

```
pvar
   v2 = .;
```

Note that the missing value syntax is valid only when you use the REFMODEL statement. See the section "Modifying a FACTOR Model from a Reference Model" on page 1608 for a more detailed example of FACTOR model respecification. See the section "Modifying a PATH Model from a Reference Model" on page 1673 for a more detailed example of PATH model respecification.

As discussed the section "Default Partial Variance Parameters" on page 1691, PROC CALIS generates default free variance parameters for the exogenous variables and default free error variance parameters for the endogenous variables in the confirmatory FACTOR or PATH model. When you use the REFMODEL statement for defining a reference model, these default free variance parameters in the old (reference) model are not transferred to the new model. Instead, the new model generates its own set of default free variance parameters *after* the new model is resolved from the reference model, the REFMODEL statement options,

the RENAMEPARM statement, and the PVAR statement specifications in the new model. If you want any of the variance or error (partial) variance parameters to be constrained across the models by means of the REFMODEL specification, you must specify them explicitly in the PVAR statement of the reference model so that the same variance or error (partial) variance specification is transferred to the new model.

RAM Statement

RAM < **VAR=***variable-list* | **[** *variable-list***=***number-list* < **,** *variable-list***=***number-list . . .* > **]**, > < *ram-entry* < **,** *ram-entry . . .* > > **;**

where *variable-list* is a list of variables for the rows and columns of the _A_ and _P_ matrices and the rows of the _W_ vector of the RAM model, *number-list* is a list of positive integers that denote the order of the specified variables, and *ram-entry* is a parameter specification for an element in one of the three RAM model matrices. You can specify latent variables in addition to observed variables in the VAR= option.

RAM stands for the reticular action model developed by McArdle (1980). The RAM model implemented in PROC CALIS extends the original RAM model with the specification of the mean vector in the _W_ vector. See the section "The RAM Model" on page 1769 for details about the model.

The RAM statement specification consists of the list of the variables in the model and the parameters and their locations the RAM model matrices. For example, consider the following simple RAM model specification:

```
ram  var= x1-x2 y3,
     _A_ 3 1,
     _A_ 3 2;
```

In this statement,variables x1, x2, and y3 are specified in the VAR= option. The variable order in the VAR= option is important. The same variable order applies to the rows and columns of the _A_ matrix. Next, there are two *ram-entries*. The first *ram-entry* specifies that the third variable (y3) has a path from the first variable (x1). Similarly, the second *ram-entry* specifies that y3 has a path from x2.

Specifying the VAR= Option

In the VAR= option, you specify the list of observed and latent variables in the RAM model. There are two ways to specify the VAR= list. The first way is a simple listing of variables. For example, you specify a total of 18 variables in the RAM model in the following statement:

```
ram  var= a b c x1-x5 y1-y10;
```

The order of the variables in this VAR= list is important. The same variable order applies to the rows and columns of the _A_ and _P_ matrices and the rows of the _W_ matrices. Although it is not required to arrange the variables according to whether they are observed or latent in the VAR= list, you can do so for your own convenience. PROC CALIS checks each variable in the VAR= list against the associated data sets to determine whether the variable is observed or latent.

When you specify the parameters in the *ram-entries*, you represent variables by the variable numbers that refer to the VAR= list. Therefore, it is important to make correct association of the variables and their order on the VAR= list. To this end, you can add some comments in your VAR= list to make the variable numbers explicit. For example,

```
ram  var= a      /* 1 */
          b      /* 2 */
          c      /* 3 */
          x1-x5  /* 4-8 */
          y1-y10 /* 9-18 */;
```

Another way to specify VAR= list is to provide the *variable-lists* together with explicit ordering indicated in the *number-lists*. For example, in the following statement you specify exactly the same variable list as that in the preceding example:

```
ram  var= [x1-x5 = 4 to 8, c = 3,  y1-y10 = 9 to 18, a = 1, b = 2];
```

Apart from showing how you can construct the VAR= list in a very general way with the *number-lists*, there is no particular reason why the *variable-lists* in the preceding specification are not in either an ascending or a descending order. Perhaps a more natural and useful way to use this type of explicit ordering specification is to place variables in exactly the same order as intended. For example, the following VAR= specification serves as a "key" of the variable numbers in the subsequent *ram-entries*:

```
ram  var= [x1 = 1, x2 = 2, y1 = 3, y2 = 4, y3 = 5],
     _A_   1   2 ,
     _P_   2   2 ;
```

With reference to the explicit variable numbers in the VAR= list, you can interpret the **_A_[1,2]** specification immediately as the effect from x2 to x1, and the **_P_[2,2]** specification as the variance of x2.

If the VAR= option is not specified in the RAM statement, the *n* observed variables in the VAR statement are used as the first *n* variables in the VAR= list. If you specify neither the VAR= option in the RAM statement nor the VAR statement, all *n* numerical variables in the associated data sets serve as the first *n* variables in the RAM model matrices. If there are more than *n* variables used in the *ram-entries*, the extra variables are all treated as latent variables in the RAM model.

Latent variables generated by PROC CALIS for the RAM model are named in two different ways, depending on whether your RAM model is specified under a MODEL statement. If you do not use the MODEL statement (for example, in situations with single-group analyses), latent variables are named _Factor1, _Factor2, and so on. If your RAM model is define within the scope of a MODEL statement, latent variables are named _Mdlk_F1, _Mdlk_F2, and so on, where k is substituted with the model number that is specified in the MODEL statement. For example, _Mdl2_F1 is a latent factor that is specified under a RAM model within the scope of the MODEL statement with 2 as its model number.

Because data sets might contain nonnumerical variables, implicit variable ordering deduced from the data sets is sometimes obscured. Therefore, it is highly recommended that you use the VAR= option to list all the variables in the RAM model.

Specifying a *ram-entry*

matrix-name | *matrix-number* *row-number* *column-number* < *parameter-spec* >

A *ram-entry* is a parameter specification of a matrix element of the RAM model. In each *ram-entry*, you first specify the matrix by using either the *matrix-name* or the *matrix-number*. Then you specify the *row-number* and the *column-number* of the element of the matrix. At the end of the *ram-entry*, you can optionally specify various kinds of parameters in *parameter-spec*. You can specify as many *ram-entries* as needed in your RAM model. *Ram-entries* are separated by commas. For example, consider the following specification:

```
ram  var= x1-x2 y3,
     _A_ 3 1  1.,
     _A_ 3 2;
```

You specify three variables in the VAR= option of the RAM statement. In the first *ram-entry*, variable y3 has a path from variable x1 with a fixed path coefficient 1. In the second *ram-entry*, variable y3 has a path from variable x2. Because the *parameter-spec* is blank, the corresponding path coefficient (or the effect from x2 on y3) is a free parameter by default.

Specifying the matrix-name or matrix-number

The three model matrices in the RAM model are: _A_, _P_, and _W_. See the section "The RAM Model" on page 1769 for the mathematical formulation of the RAM model. The *matrix-name* or *matrix-number* specifications in the *ram-entries* refer to these model matrices. You can use the following keywords for *matrix-name* or *matrix-number*:

_A_, _RAMA_, or 1	for the elements in the **A** matrix, which is for path coefficients or effects
_P_, _RAMP_, or 2	for the elements in the **P** matrix, which is for variances and covariances
_W_, _RAMW_, or 3	for the elements in the **W** vector, which is for intercepts and means

Specifying the row-number and column-number

After you specify the *matrix-name* or *matrix-number* in a *ram-entry*, you need to specify the *row-number* and *column-number* that correspond to the intended element of the matrix being specified.

Specifying the parameter-spec

You can specify three types of parameters in *parameter-spec*:

- A free parameter without an initial estimate: blank or *parameter-name*

  You can specify a free parameter for the matrix element in a *ram-entry* by either omitting the *parameter-spec* (that is, leaving it blank) or specifying a *parameter-name*. For example, both of the following *ram-entries* specify that **_A_[3,1]** is a free parameter in the RAM model:

  ```
  _A_  3   1
  ```

  and

  ```
  _A_  3   1   beta
  ```

  The difference is that in the latter you name the effect (path coefficient) for the **_A_[3,1]** element as beta, while in the former PROC CALIS generates a free parameter name (prefixed with _Parm and followed by a unique parameter number) for the specified element. Leaving the *parameter-spec* blank is handy if you do not need to refer to this parameter in your code. But when you need to specify parameter constraints by referring to parameter names, the *parameter-name* syntax becomes necessary. For example, the following specification constrains the **_A_[3,1]** and **_A_[3,2]** paths to have equal effects (path coefficients) because they have the same *parameter-name* beta:

```
ram  var= x1-x2 y3,
     _A_ 3 1  beta,
     _A_ 3 2  beta;
```

- A free parameter with an initial estimate: (*number*) or *parameter-name* (*number*)

  You can specify a free parameter with an initial estimate in a *ram-entry* by either specifying the initial estimate within parentheses or specifying a *parameter-name* followed by the parenthesized initial estimate. For example, both of the following *ram-entries* specify that **_A_[3,1]** is a free parameter with an initial estimate of 0.3 in the RAM model:

  ```
  _A_  3   1   (0.3)
  ```

  and

  ```
  _A_  3   1   beta  (0.3)
  ```

  In the latter you name the effect (path coefficient) for the **_A_[3,1]** element as beta, while in the former PROC CALIS generates a free parameter name (prefixed with _Parm and followed by a unique parameter number). The latter syntax is necessary when you need to specify parameter constraints by referring to the parameter name beta. The former syntax is more convenient when you do not need to refer to this parameter in other specifications.

  For the latter syntax with a *parameter-name* specified, you can omit the pair of parentheses or exchange the position of *parameter-name* and *number* (or both) without changing the nature of the parameter. That is, you can use the following equivalent specifications for a named free parameter with initial values:

  ```
  _A_  3   1   beta  0.3
  ```

  and

  ```
  _A_  3   1   .3  beta
  ```

- A fixed parameter value: *number*

  You can specify a fixed value by simply providing it as the *parameter-spec* in a *ram-entry*. For example, in the following syntax you specify that **_A_[3,1]** is a fixed value of 0.3:

  ```
  _A_  3   1   0.3
  ```

  The fixed value for **_A_[3,1]** does not change during the estimation. To distinguish this syntax from the initial value specification, notice that you do not put **0.3** inside parentheses, nor do you put a *parameter-name* before or after the provided value.

Notes and Cautions about Specifying *ram-entries*

- Fixed values such as 1.0 must be specified explicitly.
- The *row-number* and *column-number* in the *ram-entries* refer to the VAR= variable list of the RAM statement. An exception is for the _W_ vector, of which the *column-number* should always be 1 and does not refer to any particular variable.
- When a *row-number* or *column-number* in a *ram-entry* (except for the *column-number* of _W_) does not have any reference in the VAR= variable list (or is greater than the number of default observed variables when the VAR= option is not specified), PROC CALIS treats the corresponding row or column variable as a latent variable and generates variable names for it.
- The largest row or column number used in any *ram-entry* should not exceed the sum of observed and latent variables intended in the RAM model. Otherwise, some extraneous latent variables might be created.

Default Parameters

It is important to understand the default parameters in the RAM model. First, if you know which parameters are default free parameters, you can make your specification more efficient by omitting the specifications of those parameters that can be set by default. For example, because all exogenous variances and error variances in the RAM model are free parameters by default, you do not need to specify the diagonal elements of the _P_ matrix if they are not constrained in the model. Second, if you know which parameters are default free parameters, you can specify your model accurately. For example, because all the error covariances in the RAM model are fixed zeros by default, you must specify the corresponding off-diagonal elements of the _P_ matrix in the *ram-entries*. See the section "Default Parameters in the RAM Model" on page 1776 for details about the default parameters of the RAM model.

Modifying a RAM Model from a Reference Model

This section assumes that you use a REFMODEL statement within the scope of a MODEL statement and that the reference model (or base model) is also a RAM model. The reference model is called the old model, and the model that refers to this old model is called the new model. If the new model is not intended to be an exact copy of the old model, you can use the following extended RAM modeling language to make modifications on the model specification. The syntax for modifications is very much the same as the ordinary RAM modeling language (see the section "RAM Statement" on page 1692), except that you cannot specify the VAR= option in the RAM statement. The reason is that the VAR= variable list in the new RAM model should be exactly the same as the old model; otherwise, the *row-number* and *column-number* in the *ram-entries* would not have the same references and thus would make model referencing meaningless. Hence, the syntax for respecifying (modifying) the RAM model contains only the *ram-entries*:

RAM *ram-entry* < **,** *ram-entry* . . . > **;**

The syntax of the *ram-entry* is the same as that of the original RAM statement, with an addition of the missing value specification for the *parameter-spec*, which denotes the deletion of a parameter location.

The new model is formed by integrating with the old model in the following ways:

Duplication: If you do not specify in the new model a parameter location (matrix element) that exists in the old model, the old parameter specification is duplicated in the new model.

Addition: If you specify in the new model a parameter location (matrix element) that does not exist in the old model, the new parameter specification is added to the new model.

Deletion: If you specify in the new model a parameter location (matrix element) that also exists in the old model and the new *parameter-spec* is denoted by the missing value '.', the old parameter specification is not copied into the new model.

Replacement: If you specify in the new model a parameter location (matrix element) that also exists in the old model and the new parameter is not denoted by the missing value '.', the new parameter specification replaces the old one in the new model.

For example, consider the following two-group analysis:

```
proc calis;
   group 1 / data=d1;
   group 2 / data=d2;
   model 1 / group=1;
      ram
         var = [V1-V6 = 1 to 6, F1 = 7],
         _A_ 1 7  1.,
         _A_ 2 7  load1,
         _A_ 3 7  load2,
         _A_ 7 4  ,
         _A_ 7 5  ,
         _A_ 7 6  ,
         _P_ 1 1  ,
         _P_ 2 2  ,
         _P_ 3 3  ,
         _P_ 7 7  ,
         _P_ 4 4  ,
         _P_ 5 5  ,
         _P_ 6 6  ,
         _P_ 1 2  cve12;
   model 2 / group=2;
      refmodel 1;
      ram
         _A_ 3 7  load1,
         _P_ 1 2      .,
         _P_ 2 3  cve23;
run;
```

In this example, you specify Model 2 by referring to Model 1 in the REFMODEL statement. Model 2 is the new model which refers to the old model, Model 1. This example illustrates the four types of model integration process by PROC CALIS:

- Duplication: All parameter specifications, except for `_A_[3,7]` and `_P_[1,2]`, in the old model are duplicated in the new model.
- Addition: The new parameter cve23 is added for the matrix element `_P_[2,3]` in the new model.
- Deletion: The parameter location `_P_[1,2]` and associated parameter cve12 are not copied into the new model, as indicated by the missing value '.' in the new model specification.

- Replacement: The **_A_[3,7]** path in the new model replaces the same path in the old model with another parameter for the path coefficient. As a results, in the new model paths specified by **_A_[3,7]** and **_A_[2,7]** are constrained to have the same path coefficient parameter load1.

PROC CALIS might have generated some default parameters (named with the '_Add' prefix) for the old (reference) model. These default parameters in the old (reference) model do *not* transfer to the new model. Only after the new model is resolved from the reference model, the REFMODEL statement options, the RENAMEPARM statement, and the model respecification are the default parameters of the new RAM model generated. In this way, the generated parameters in the new model are not constrained to be the same as the corresponding parameters in the old (reference) model. If you want any of these default parameters to be constrained across the models, you must specify them explicitly in the *ram-entries* of the RAM statement of the reference model so that these specifications are duplicated to the new model via the REFMODEL statement.

REFMODEL Statement

REFMODEL *model-number* < / *options* > **;**

The REFMODEL statement is not a modeling language itself. It is a tool for referencing and modifying models. It is classified into one of the modeling languages because its role is similar to other modeling languages.

REFMODEL *model-number* < / *options* > **;**
RENAMEPARM *parameter renaming* **;**
main model specification statement **;**
subsidiary model specification statements **;**

In the REFMODEL statement, you specify the *model-number* (between 1 and 9,999, inclusive) of the model you are making reference to. The reference model must be well-defined in the same PROC CALIS run. In the *options*, you can rename all the parameters in the reference model by adding a prefix or suffix so that the current model has a new set of parameters. The RENAMEPARM statement renames individual parameters in the reference model to new names. In the main model specification statement and the subsidiary model specification statements, you can respecify or modify the specific parts of the reference model. The specification of these statements must be compatible with the model type of the reference model.

NOTE: The REFMODEL statement does *not* simply copy model specifications from a reference model. If you do not change any of the parameter names of the reference model by any of the REFMODEL statement options, the REFMODEL statement copies only the *explicit* specifications from the reference model to the new model. However, the REFMODEL statement does not copy the default parameters from the reference model to the new model. For example, consider the following statements:

```
proc calis;
   group 1 / data=a1;
   group 2 / data=a2;
   model 1 / group=1;
      path x1 ===> x2;
   model 2 / group=2;
      refmodel 1;
run;
```

In this example, Model 2 makes reference to Model 1. This means that the path relationship between x1 and x2 as specified in Model 1 is exactly the same path relationship you want Model 2 to have. The path coefficients in these two models are constrained to be the same. However, the variance parameter of x1 and the error variance parameter for x2 are not constrained in these models. Rather, these two parameters are set by default in these models separately. If you intend to constrain all parameters in the two models, you can specify all the parameters in Model 1 explicitly and use the REFMODEL statement for Model 2, as shown in the following statements:

```
proc calis;
   group 1 / data=a1;
   group 2 / data=a2;
   model 1 / group=1;
      path x1 ===> x2;
      pvar x1 x2;
   model 2 / group=2;
      refmodel 1;
run;
```

This way Model 2 makes reference to all the explicitly specified parameters in Model 1. Hence the two models are completely constrained. However, a simpler way to fit exactly the same model to two groups is to use a single model definition, as shown in the following statements:

```
proc calis;
   group 1 / data=a1;
   group 2 / data=a2;
   model 1 / group=1,2;
      path x1 ===> x2;
run;
```

This specification has the same estimation results as those for the preceding specification.

When you also use one of the REFMODEL statement options, the REFMODEL statement is no longer a simple copy of explicit parameter specifications from the reference model. All parameters are renamed in the new model in the model referencing process. The following *options* are available in the REFMODEL statement:

ALLNEWPARMS

appends to the parameter names in the reference model with _mdl and then an integer suffix denoting the model number of the current model. For example, if qq is a parameter in the reference model for a current model with model number 3, then this option creates qq_mdl3 as a new parameter name.

PARM_PREFIX=*prefix*

inserts to all parameter names in the reference model with the *prefix* provided. For example, if qq is a parameter in the reference model for a current model, then **PARM_PREFIX=pre_** creates pre_qq as a new parameter name.

PARM_SUFFIX=*suffix*

appends to all parameter names in the reference model with the *suffix* provided. For example, if qq is a parameter in the reference model for a current model, then **PARM_SUFFIX=_suf** creates qq_suf as a new parameter name.

Instead of renaming all parameters, you can also rename parameters individually by using the RENAMEPARM statement within the scope of the REFMODEL statement.

You can also add the main and subsidiary model specification statements to modify a particular part from the reference model. For example, you might like to add or delete some equations or paths, or to change a fixed parameter to a free parameter or vice versa in the new model. All can be done in the respecification in the main and subsidiary model specification statements within the scope of the MODEL statement to which the REFMODEL statement belongs. Naturally, the modeling language used in respecification must be the same as that of the reference model. See the individual statements for modeling languages for the syntax of respecification. Note that when you respecify models by using the main and subsidiary model specification statements together with the RENAMEPARM statement or the REFMODEL options for changing parameter names, the parameter name changes occur after respecifications.

RENAMEPARM Statement

RENAMEPARM *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents:

old_parameters = *parameter-spec*

You can use the RENAMEPARM statement to rename parameters or to change the types of parameters of a reference model so that new parameters are transferred to the new model in question. The RENAMEPARM statement is a subsidiary model specification statement that should be used together with the REFMODEL statement. The syntax of the RENAMEPARM statement is similar to that of the VARIANCE statement—except that in the RENAMEPARM statement, you put parameter names on the left-hand side of equal signs, whereas you put variable names on the left-hand side in the VARIANCE statement. You can use no more than one RENAMEPARM statement within the scope of each REFMODEL statement.

In the REFMODEL statement, you transfer all the model specification information from a base model to the new model being specified. The RENAMEPARM statement enables you to modify the parameter names or types in the base model before transferring them to the new model. For example, in the following example, you define Model 2, which is a new model, by referring it to Model 1, the base model, in the REFMODEL statement.

```
model 1;
   lineqs
      V1 =    F1 + E1,
      V2 = b2 F1 + E2,
      V3 = b3 F1 + E3,
      V4 = b4 F1 + E4;
   variance F1 = vF1,
      E1-E4 = ve1-ve4;
model 2;
   refmodel 1;
   renameparm ve1-ve4=new1-new4, b2=newb2(.2), b4=.3;
```

Basically, the LINEQS model specification in Model 1 is transferred to Model 2. In addition, you redefine some parameters in the base model by using the RENAMEPARM statement. This example illustrates two kinds of modifications that the RENAMEPARM statement can do:

- creating new parameters in the new model

The error variances for E1–E4 in Model 2 are different from those defined in Model 1 because new parameters new1–new4 are now used. Parameter b2 is renamed as newb2 with a starting value at 0.2 in Model 2. So the two models have distinct path coefficients for the F1-to-V2 path.

- changing free parameters into fixed constants

  By using the specification **b4=.3** in the RENAMEPARM statement, b4 is no longer a free parameter in Model 2. The path coefficient for the F1-to-V4 path in Model 2 is now fixed at 0.3.

The RENAMEPARM statement is handy when you have just few parameters to change in the reference model defined by the REFMODEL statement. However, when there are a lot of parameters to modify, the RENAMEPARM statement might not be very efficient. For example, to make all parameters unique to the current model, you might consider using the ALLNEWPARMS, PARM_PREFIX=, or PARM_SUFFIX= option in the REFMODEL statement.

SAS Programming Statements

You can use SAS programming statements to define dependent parameters, parametric functions, and equality constraints among parameters.

Several statistical procedures support the use of SAS programming statements. The syntax of SAS programming statements are common to all these procedures and can be found in the section "Programming Statements" on page 558 in Chapter 20, "Shared Concepts and Topics."

SIMTESTS Statement

SIMTESTS | **SIMTEST** *sim-test* < *sim-test . . .* > **;**

where *sim-test* represents one of the following:

- *test-name* = **[** *functions* **]**
- *test-name* = **{** *functions* **}**

and *functions* are either parameters in the model or parametric functions computed in the SAS programming statements.

When the estimates in a model are asymptotically multivariate-normal, continuous and differentiable functions of the estimates are also multivariate-normally distributed. In the SIMTESTS statement, you can test these parametric functions simultaneously. The null hypothesis for the simultaneous tests is assumed to have the following form:

$H_0 : h_1(\theta) = 0, h_2(\theta) = 0, \ldots$

where θ is the set of model parameters (independent or dependent) in the analysis and each $h_i()$ is a continuous and differentiable function of the model parameters.

To test parametric functions simultaneously in the SIMTESTS statement, you first assign a name for the simultaneously test in *test-name*. Then you put the parametric functions for the simultaneous test inside a pair of parentheses: either the '{' and '}' pair, or the '[' and ']' pair. For example, if θ_1, θ_2, θ_3, and θ_4

are parameters in the model and you want to test the equality of θ_1 and θ_2 and the equality of θ_3 and θ_4 simultaneously, you can use the following statements:

```
simtests
   Equality_test = [t1_t2_diff t3_t4_diff];
t1_t2_diff   = theta1 - theta2;
t3_t4_diff   = theta3 - theta4;
```

In the SIMTESTS statement, you test two functions t1_t2_diff and t3_t4_diff simultaneously in the test named Equality_test. The two parametric functions t1_t2_diff and t3_t4_diff are computed in the SAS programming statements as differences of some parameters in the model.

See also the TESTFUNC statement on page 1702 for testing parametric functions individually.

STD Statement

STD *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents:

var-list = *parameter-spec*

The STD statement functions exactly the same as the VARIANCE statement. The STD statement is obsolete and might not be supported in future versions of PROC CALIS. Use the VARIANCE statement instead.

STRUCTEQ Statement

STRUCTEQ *variables* < / *label* > **;**

where *label* represents:

LABEL | **NAME** = *name*

The STRUCTEQ statement functions exactly the same as the DETERM statement.

TESTFUNC Statement

TESTFUNC *functions* **;**

where *functions* are either parameters in the model or parametric functions computed in the SAS programming statements.

When the estimates in a model are asymptotically multivariate-normal, any continuous and differentiable function of the estimates is also normally distributed. In the TESTFUNC statement, you can test these parametric functions using z-tests. The form of the null hypothesis is as follows:

$H_0 : h(\theta) = 0$

where θ is the set of model parameters (independent or dependent) in the analysis and $h()$ is a continuous and differentiable function of the model parameters.

For example, if θ_1, θ_2, and θ_3 are parameters in the model, and you want to test whether θ_1 and θ_2 are the same and whether θ_3 is the same as the average of θ_1 and θ_2, you can use the following statements:

```
testfunc   t1_t2_diff t3_t1t2_diff;
t1_t2_diff   = theta1 - theta2;
t3_t1t2_diff = theta3 - (theta1 + theta2)/2;
```

In the TESTFUNC statement, you test two functions: t1_t2_diff and t3_t1t2_diff. These two functions are defined in the SAS programming statements that follow after the TESTFUNC statement. Thus, t1_t2_diff represents the difference between θ_1 and θ_2, and t3_t1t2_diff represents the difference between θ_3 and the average of θ_1 and θ_2.

See the SIMTESTS statement if you want to test several null hypotheses simultaneously.

VAR Statement

VAR *variables* **;**

The VAR statement defines and limits the set of observed variables that are available for the corresponding model analysis. It is one of the subsidiary group specification statements. You can use the VAR statement no more than once within the scope of each GROUP or the PROC CALIS statement. The set of variables in the VAR statement must be present in the data set specified in the associated GROUP or the PROC CALIS statement.

The following example shows the specification of 16 variables (x1, x2, . . . , x9, x10, y, z1,z2, . . . , z5) in the VAR statement:

```
var x1-x10 y z1-z5;
```

The VAR statement in PROC CALIS does not support the specification of variable lists with syntax that refers to consecutive variables in the data set. For example, if QQ, RR, SS, and TT are consecutive variables in a data set, the following specification of these four variables in the VAR statement is not supported by PROC CALIS:

```
var QQ-TT;
```

The VAR statement should not be confused with the PARAMETERS statement. In the PARAMETERS statement, you specify additional *parameters* in the model. Parameters are population quantities that characterize the functional relationships, variations, or covariation among variables. Unfortunately, parameters are sometimes referred to as *var-list* in the optimization context. You have to make sure that all variables specified in the VAR statement refer to the variables in the input data set, while the parameters specified in the PARAMETERS statement are population quantities that characterize distributions of the variables and their relationships.

In some modeling languages of PROC CALIS, you can also specify the observed variables either directly (for example, through the VAR= or similar option in some main model specification statements) or indirectly (for example, through the specification of functional relationships between observed variables). How does the VAR statement specifications interplay with the observed variables specified in the model? This depends on the types of models specified. Four different cases are considered in the following.

Case 1. Exploratory Factor Models With No VAR= option in the FACTOR statement. For exploratory factor models specified using the FACTOR statement, it is important for you to use the VAR statement to select and

limit the set of the observed variables for analysis. The reason is simply that there is no other options in the FACTOR statement that will serve the same purpose. For example, you analyze only v1–v3 in the following exploratory factor model even though there might be more observed variables available in the data set:

```
proc calis;
   var v1-v3;
   factor  n=1;
```

If you do not specify the VAR statement, PROC CALIS simply selects all numerical variables for analysis. However, to avoid confusions it is a good practice to specify the observed variables explicitly in the VAR statement.

Case 2. Models With a VAR= or Similar Option for Defining the Set of Observed Variables for Analysis. The classes of models considered here are: COSAN, LISMOD, MSTRUCT, and RAM. Except for the LISMOD models, in all other three classes of models you can specify the observed variables in the model by using the a VAR= option in the respective main model specification statement. For the LISMOD models, you can specify all observed variables that should be included in the model in the XVAR= and YVAR= options of the LISMOD statement. Therefore, the use of the VAR statement for these models might become unnecessary. For example, the following MSTRUCT statement specifies the observed variables v1–v6 in the VAR= option:

```
proc calis;
   mstruct  var=v1-v6;
```

It would have been redundant to use a VAR statement to specify v1–v6 additionally. The same conclusion applies to the COSAN and the RAM models.

Another example is when you specify a LISMOD model. In the following LISMOD specification, variables v1–v8 would be the set of observed variables for analysis:

```
proc calis;
   var v1-v8;
   lismod  xvar = v1-v4,
           yvar = v5-v8,
           eta  = factor1,
           xi   = factor2;
```

Again, there is no need to add a VAR statement merely repeating the specification of variables v1–v8.

If you do specify the VAR statement in addition to the specification of variable lists in these models, PROC CALIS will check the consistency between the lists. Conflicts arise if the two lists do not match.

For example, the following statements will generate an error in model specification because v6 specified in the MSTRUCT model is not defined as an observed variable available for analysis in the VAR statement (even if v6 might indeed be present in the data set):

```
proc calis;
   var v1-v5;
   mstruct  var=v1-v6;
```

So it is an error when you specify fewer observed variables in the VAR statement than in the VAR= option in the model. How about if you specify more variables in the VAR statement? PROC CALIS will also general an error because the extra variables in VAR statement will not be well-undefined in the model. For example, v7–v10 specified in the VAR statement are supposed to be included into the model, but they not listed on either the XVAR= or YVAR= list in the following LISMOD statement:

```
proc calis;
   var v1-v10;
   lismod  xvar = v1-v3,
           yvar = v4-v6,
           eta  = factor1,
           xi   = factor2;
```

Therefore, if you must specify the VAR statement for these models, the specifications of the observed variables must be consistent in the VAR statement and in the relevant model options. However, to avoid potential conflicts in these situations, you are recommended to specify the observed variables in the VAR=, XVAR=, or YVAR= lists only.

When the VAR= option is not specified in the COSAN, MSTRUCT, or RAM statement, the VAR statement specification will be used as the list of observed variables in the model. If both of the VAR= option and VAR statement specification are lacking, then all numerical variables in the associated data set will be used in the model. However, to avoid confusions the preferred method is to specify the list of observed variables explicitly on the VAR=, XVAR=, or YVAR= option of the main model specification statements.

Case 3. Models With Indirect Ways to Include the Set of Observed Variables for Analysis. Two types of models are considered here: LINEQS and PATH. For these models, the main use of the VAR statement is to include those observed variables that are not mentioned in model specifications.

For example, in the following statements for a LINEQS model variable v3 is not mentioned in the LINEQS statement:

```
proc calis;
   var v1-v3;
   lineqs   v1 = a1 * v2 + e1;
```

With the specification in the VAR statement, however, variable v3 is included into the model as an exogenous manifest variable. Similarly, the same applies to the following PATH model specification:

```
proc calis;
   var v1-v3;
   path   v1 <=== v2;
```

Again, variable v3 is included into the PATH model because it is specified in the VAR statement.

The two preceding examples also suggest that you do not need to use the VAR statement when your already mentions all observed variables in the model specification. For example, if your target set of observed variables are v1–v3, the use of the VAR statement in the following specification is *unnecessary*:

```
proc calis;
   var v1-v3;
   path   v1 <=== v2;
      pvar v3;
```

For the two types of models considered here, you can also use the VAR statement to define and limit the set of observed variables for analysis. For example, you might have v1, v2, v3 in your data set as observed variables for analysis; but somehow in your model v2 should be treated as a latent variable. You might use the following code to exclude v2 as an observed variable in the model:

```
proc calis;
   var v1  v3;
   path   v1 <=== v2;
      pvar v3;
```

The role of the VAR statement here is to define and limit the set of observed variables available for the model. Hence, only variables v1 and v3 are supposed to be observed variables in the model while variable v2 in the PATH model is treated as latent.

In sum, in the current situation the use of the VAR statement should depend on whether a variable should or should not be included as an observed variable in your theoretical model.

Case 4. Confirmatory Factor Model With the FACTOR statement. In this case, the VAR statement still limits the set of observed variables being analyzed in the confirmatory factor model. However, because all observed variables in a confirmatory factor analysis must be loaded on (or related to) some factors through the specification of *factor-variable-relations* in the FACTOR statement, all observed variables in the model should have been specified (or mentioned) in the FACTOR statement already, making it redundant to use the VAR statement for the same purpose.

VARIANCE Statement

VARIANCE *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents:

var-list < **=***parameter-spec* >

The VARIANCE statement specifies the variance parameters in connection with the LINEQS model. Notice that the VARIANCE statement is different from the VAR statement, which specifies variables for analysis. In previous versions of PROC CALIS, the STD statement name was used instead of the VARIANCE statement name. Although these two names result in the same functionalities, the VARIANCE statement name reflects the intended usages better.

In the LINEQS model, variance parameters are defined only for *exogenous* manifest and latent variables (including error and disturbance variables) in the model. Therefore, you cannot list any *endogenous* variables in the *var-list* list of the VARIANCE statement. You can specify no more than one VARIANCE statement for each LINEQS model.

In each *assignment* of the VARIANCE statement, you list the *var-list* whose variances you want to specify. Optionally, you can provide a list of parameter specifications (*parameter-spec*) after an equal sign for each *var-list* list.

You can specify the following five types of the parameters for the variances of the exogenous variables in the VARIANCE statement:

- an unnamed free parameter
- an initial value
- a fixed value

- a free parameter with a name provided
- a free parameter with a name and initial value provided

Consider a LINEQS model with exogenous variables V1, V2, F1, D2, and E3. The following VARIANCE statement illustrates the five types of parameter specifications in five *assignments*:

```
variance
   V1 ,
   V2 = (.5),
   F1 = 1.0,
   D2 = dvar,
   E3 = evar(0.7);
```

In this statement, the variance of V1 is specified as an unnamed free parameter. For this variance, PROC CALIS generates a parameter name with the _Parm prefix and appended with a unique integer (for example, _Parm1). The variance of V2 is an unnamed free parameter with an initial value of 0.5. PROC CALIS also generates a parameter name for this variance. The variance of F1 is a fixed value of 1.0. This value stays the same during the estimation. The variance of D2 is a free parameter named dvar. The variance of E3 is a free parameter named evar with an initial value of 0.7.

When you need to specify a long parameter name list, you can consider using the prefix-name specification for the parameter list. For example, the following statement specifies 100 unique parameter names for the variances of E1–E100:

```
variance
   E1-E100 = 100 * evar__; /* evar with two trailing underscores */
```

In the VARIANCE statement, evar_ _ is a prefix-name with the root evar. The notation `100*` means this prefix-name is applied 100 times, resulting in a generation of the 100 unique parameter names evar001, evar002, . . . , evar100.

The root of the prefix-name should have few characters so that the generated parameter name is not longer than 32 characters. To avoid unintentional equality constraints, the prefix-names should not coincide with other parameter names.

Mixed Parameter Lists

You can specify different types of parameters for the list of variances. For example, the following statement uses a list of parameters with mixed types:

```
variance
   E1-E6 = vp1 vp2(2.0) vp3  4. (.3) vp6(.4);
```

This is equivalent to the following specification:

```
variance
   E1 = vp1
   E2 = vp2(2.0),
   E3 = vp3,
   E4 = 4. ,
   E5 = (.3),
   E6 = vp6(.4);
```

Notice that an initial value followed after a parameter name is associated with the free parameter. For example, in the original mixed list specification, the specification `(2.0)` after vp2 is interpreted as the initial value for the parameter vp2, but not as the initial estimate for the variance of E3.

However, if you indeed want to specify that vp2 is a free parameter *without* an initial value and 2.0 is an initial estimate for the variance of E3 (while keeping all other things the same), you can use a null initial value specification for the parameter vp2, as shown in the following statement:

```
variance
   E1-E6 = vp1 vp2() (2.0)  4. (.3) vp6(.4);
```

This way 2.0 becomes the initial estimate for the variance of E3. Because a parameter list with mixed types might be confusing, you can break down the specifications into separate *assignments* to remove ambiguities. For example, you can use the following equivalent specification:

```
variance
   E1 = vp1
   E2 = vp2,
   E3 = (2.),
   E4 = 4. ,
   E5 = (.3),
   E6 = vp6(.4);
```

Shorter and Longer Parameter Lists

If you provide fewer parameters than the number of variances in the *var-list* list, all the remaining parameters are treated as unnamed free parameters. For example, the following specification assigns a fixed value of 1.0 to the variance of F1 while treating the other three variances as unnamed free parameters:

```
variance
   F1-F4 = 1.0;
```

This specification is equivalent to the following specification:

```
variance
   F1 = 1.0, F2-F4;
```

If you intend to fill up all values with the last parameter specification in the list, you can use the continuation syntax `[...]`, `[..]`, or `[.]`, as shown in the following example:

```
variance
   E1-E100 = 1.0 psi [...];
```

This means that the variance of E1 is fixed at 1.0, while the variances of E1–E100 are all free parameter named psi. All variances except that for E1 are thus constrained to be equal by using the same parameter name.

However, you must be careful not to provide too many parameters. For example, the following specification results in an error:

```
variance
   E1-E6 = 1.0 psi2-psi6 extra;
```

The parameters after psi6 are excessive.

Default Variance Parameters

In the LINEQS model, by default all variances of exogenous manifest and latent variables (including error and disturbance variables) are free parameters. For these default free parameters, PROC CALIS generates the parameter names with the _Add prefix and appended with unique integer suffixes. You can also use the VARIANCE statement specification to override these default variance parameters in situations where you want to specify parameter constraints, provide initial or fixed values, or make parameter references.

Because only exogenous variables can have variance parameters in the LINEQS model and all these exogenous variances are free parameters by default, there are no default fixed zeros for any variances in the LINEQS model.

Modifying a Variance Parameter Specification from a Reference Model

If you define a new LINEQS model by using a reference (old) model in the REFMODEL statement, you might want to modify some parameter specifications from the VARIANCE statement of the reference model before transferring the specifications to the new model. To change a particular variance specification from the reference model, you can simply respecify the same variance with the desired parameter specification in the VARIANCE statement of the new model. To delete a particular variance parameter from the reference model, you can specify the desired variance with a missing value specification in the VARIANCE statement of the new model.

For example, suppose that the variance of V1 is specified in the reference model but you do not want this variance specification to be transferred to the new model, you can use the following VARIANCE statement specification in the new model:

```
variance V1 = .;
```

Note that the missing value syntax is valid only when you use the REFMODEL statement. See the section "Modifying a LINEQS Model from a Reference Model" on page 1622 for a more detailed example of the LINEQS model respecification.

As discussed in a preceding section, PROC CALIS generates default free variance parameters for the LINEQS model if you do not specify them explicitly in the VARIANCE statement. When you use the REFMODEL statement for defining a reference model, these default free variance parameters in the old (reference) model are not transferred to the new model. Instead, the new model generates its own set of default free variance parameters *after* the new model is resolved from the reference model, the REFMODEL statement options, the RENAMEPARM statement, and the VARIANCE statement specifications in the new model. This also implies that if you want any of the variance parameters to be constrained across the models by means of the REFMODEL specification, you must specify them explicitly in the VARIANCE statement of the reference model so that the same variance specification is transferred to the new model.

VARNAMES Statement

VARNAMES *name-assignment* < **,** *name-assignment . . .* > **;**

VARNAME *name-assignment* < **,** *name-assignment . . .* > **;**

VNAMES *name-assignment* < **,** *name-assignment . . .* > **;**

where *name-assignment* represents one of the following forms:

matrix-name variable-names
matrix-name = [*variable-names*]
matrix-name = *matrix-name*

You can use the VARNAMES statement in connection with the COSAN modeling language to assign variable names for matrices. The *matrix-name* refers to any matrix you define in the COSAN statement. The *variable-names* that follow the *matrix-name* are assigned to the column variables of the matrix of interest. This applies to the first two types of VARNAMES specifications. For example,

```
varnames  F   f1-f3;
```

is exactly the same as

```
varnames  F = [ f1-f3 ];
```

Both of these assign f1, f2, and f3 as the names for the first three column variables of matrix F.

You can also use another kind of *name-assignment* in connection with a COSAN statement. Two matrix names equated by an equal sign assign the column names of the matrix on the right-hand side to the column names of the matrix on the left-hand side. This assignment assumes that the column names of at least one of the two matrices are already defined. For example, assuming that **J** and **A** are model matrices defined in a COSAN statement, the following VARNAMES statement specification specifies that both **J** and **A** have the same set of column variable names V1–V6 and F1–F3:

```
varnames  J = [ V1-V6 F1-F3 ] ,
          A = J ;
```

This is the same as the following specification:

```
varnames  J = [ V1-V6 F1-F3 ] ,
          A = [ V1-V6 F1-F3 ] ;
```

The VARNAMES statement appears to enable you to specify only the column variable names for matrices. However, PROC CALIS also uses these column variable names to assign row variable names of the related matrices in the covariance and mean structure formulas for the COSAN model. PROC CALIS uses the following rules to determine the row variable names of a matrix in the model:

- If a matrix is the first matrix of any term in the covariance or mean structure formula, the row variable names are the names of the manifest variables.
- If a matrix is the central covariance matrix of any term in the covariance structure formula, the row variable names are the same as the column variable names.

- For any other matrices, the row variable names are the same as the column variable names of the preceding matrix in the multiplicative formula for the covariance or mean structures.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement specifies the weight variable for the observations. It is one of the subsidiary group specification statements. You can use the WEIGHT statement no more than once within the scope of each GROUP statement or the PROC CALIS statement.

Weighting is often done when the error variance associated with each observation is different and the values of the weight variable are proportional to the reciprocals of the variances. The WEIGHT and FREQ statements have a similar effect, except the WEIGHT statement does not alter the number of observations unless VARDEF=WGT or VARDEF=WDF. An observation is used in the analysis only if the WEIGHT variable is greater than 0 and is not missing.

Details: CALIS Procedure

Input Data Sets

You can use four different kinds of input data sets in the CALIS procedure, and you can use them simultaneously. The DATA= data set contains the data to be analyzed, and it can be an ordinary SAS data set containing raw data or a special TYPE=COV, TYPE=UCOV, TYPE=CORR, TYPE=UCORR, TYPE=SSCP, or TYPE=FACTOR data set containing previously computed statistics. The INEST= data set specifies an input data set that contains initial estimates for the parameters used in the optimization process, and it can also contain boundary and general linear constraints on the parameters. If the model does not change too much, you can use an OUTEST= data set from a previous PROC CALIS analysis; the initial estimates are taken from the values of the _TYPE_=PARMS observation. The INMODEL= or INRAM= data set contains information of the analysis models (except for user-written programming statements). Often the INMODEL= data set is created as the OUTMODEL= data set from a previous PROC CALIS analysis. See the section "OUTMODEL= or OUTRAM= Data Set" on page 1719 for the structure of both OUTMODEL= and INMODEL= data sets. Using the INWGT= data set enables you to read in the weight matrix **W** that can be used in generalized least squares, weighted least squares, or diagonally weighted least squares estimation.

BASEFIT= or INBASEFIT= Data Set

The BASEFIT= or INBASEFIT= data set saves the fit function value and the degrees of freedom of a baseline model for computing various fit indices, especially the incremental fit indices. Typically, the BASEFIT= data set is created as an OUTFIT= data set from a previous PROC CALIS fitting of a customized baseline model. See the section "OUTFIT= Data Set" on page 1729 for details about the format of the OUTFIT= and BASEFIT= data sets.

DATA= Data Set

A TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR data set can be created by the CORR procedure or various other procedures. It contains means, standard deviations, the sample size, the covariance or correlation matrix, and possibly other statistics depending on which procedure is used.

If your data set has many observations and you plan to run PROC CALIS several times, you can save computer time by first creating a TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR data set and using it as input to PROC CALIS.

For example, assuming that PROC CALIS is first run with an OUTMODEL=MODEL option, you can run the following statements in subsequent analyses with the same model in the first run:

```
/* create TYPE=COV data set */
proc corr cov nocorr data=raw outp=cov(type=cov);
run;
/* analysis using correlations */
proc calis corr data=cov inmodel=model;
run;
/* analysis using covariances */
proc calis data=cov inmodel=model;
run;
```

Most procedures automatically set the TYPE= option of an output data set appropriately. However, the CORR procedure sets TYPE=CORR unless an explicit TYPE= option is used. Thus, (TYPE=COV) is needed in the preceding PROC CORR request, since the output data set is a covariance matrix. If you use a DATA step with a SET statement to modify this data set, you must declare the TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR attribute in the new data set.

You can use a VAR statement with PROC CALIS when reading a TYPE=COV, TYPE=UCOV, TYPE=CORR, TYPE=UCORR, or TYPE=SSCP data set to select a subset of the variables or change the order of the variables.

CAUTION: Problems can arise from using the CORR procedure when there are missing data. By default, PROC CORR computes each covariance or correlation from all observations that have values present for the pair of variables involved ("pairwise deletion"). The resulting covariance or correlation matrix can have negative eigenvalues. A correlation or covariance matrix with negative eigenvalues is recognized as a singular matrix in PROC CALIS, and you cannot compute (default) generalized least squares or maximum likelihood estimates. You can specify the RIDGE option to ridge the diagonal of such a matrix to obtain a positive definite data matrix. If the NOMISS option is used with the CORR procedure, observations with any missing values are completely omitted from the calculations ("listwise deletion"), and there is no possibility of negative eigenvalues (but there is still a chance for a singular matrix).

PROC CALIS can also create a TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR data set that includes all the information needed for repeated analyses.

If the data set DATA=RAW does not contain missing values, the following statements should give the same PROC CALIS results as the previous example:

```
/* using correlations */
proc calis corr data=raw outstat=cov inmodel=model;
run;
/* using covariances */
proc calis data=cov inmodel=model;
```

```
run;
```

You can create a TYPE=COV, TYPE=UCOV, TYPE=CORR, TYPE=UCORR, or TYPE=SSCP data set in a DATA step. Be sure to specify the TYPE= option in parentheses after the data set name in the DATA statement and include the _TYPE_ and _NAME_ variables. If you want to analyze the covariance matrix but your DATA= data set is a TYPE=CORR or TYPE=UCORR data set, you should include an observation with _TYPE_=STD giving the standard deviation of each variable. By default, PROC CALIS analyzes the recomputed covariance matrix even when a TYPE=CORR data set is provided, as shown in the following statements:

```
data correl(type=corr);
   input _type_ $ _name_ $ X1-X3;
   datalines;
std   .   4.  2.  8.
corr  X1  1.0  .   .
corr  X2   .7 1.0  .
corr  X3   .5  .4 1.0
;
proc calis inmodel=model;
run;
```

INEST= Data Set

You can use the INEST= (or INVAR=) input data set to specify the initial values of the parameters used in the optimization and to specify boundary constraints and the more general linear constraints that can be imposed on these parameters.

The variables of the INEST= data set must correspond to the following:

- a character variable _TYPE_ that indicates the type of the observation
- *n* numeric variables with the parameter names used in the specified PROC CALIS model
- the BY variables that are used in a DATA= input data set
- a numeric variable _RHS_ (right-hand side); needed only if linear constraints are used
- additional variables with names corresponding to constants used in the programming statements

The content of the _TYPE_ variable defines the meaning of the observation of the INEST= data set. PROC CALIS recognizes observations with the following _TYPE_ specifications.

PARMS
: specifies initial values for parameters that are defined in the model statements of PROC CALIS. The _RHS_ variable is not used. Additional variables can contain the values of constants that are referred to in programming statements. At the beginning of each run of PROC CALIS, the values of the constants are read from the PARMS observation for initializing the constants in the SAS programming statements.

UPPERBD | UB
: specifies upper bounds with nonmissing values. The use of a missing value indicates that no upper bound is specified for the parameter. The _RHS_ variable is not used.

LOWERBD | LB
: specifies lower bounds with nonmissing values. The use of a missing value indicates that no lower bound is specified for the parameter. The _RHS_ variable is not used.

LE | <= | <
: specifies the linear constraint $\sum_j a_{ij} x_j \leq b_i$. The n parameter values contain the coefficients a_{ij}, and the _RHS_ variable contains the right-hand-side b_i. The use of a missing value indicates a zero coefficient a_{ij}.

GE | >= | >
: specifies the linear constraint $\sum_j a_{ij} x_j \geq b_i$. The n parameter values contain the coefficients a_{ij}, and the _RHS_ variable contains the right-hand-side b_i. The use of a missing value indicates a zero coefficient a_{ij}.

EQ | =
: specifies the linear constraint $\sum_j a_{ij} x_j = b_i$. The n parameter values contain the coefficients a_{ij}, and the _RHS_ variable contains the right-hand-side b_i. The use of a missing value indicates a zero coefficient a_{ij}.

The constraints specified in the INEST=, INVAR=, or ESTDATA= data set are added to the constraints specified in BOUNDS and LINCON statements.

You can use an OUTEST= data set from a PROC CALIS run as an INEST= data set in a new run. However, be aware that the OUTEST= data set also contains the boundary and general linear constraints specified in the previous run of PROC CALIS. When you are using this OUTEST= data set without changes as an INEST= data set, PROC CALIS adds the constraints from the data set to the constraints specified by a BOUNDS and LINCON statement. Although PROC CALIS automatically eliminates multiple identical constraints, you should avoid specifying the same constraint a second time.

INMODEL= or INRAM= Data Set

This data set is usually created in a previous run of PROC CALIS. It is useful if you want to reanalyze a problem in a different way such as using a different estimation method. You can alter an existing OUTMODEL= data set in the DATA step to create the INMODEL= data set that describes a modified model. See the section "OUTMODEL= or OUTRAM= Data Set" on page 1719 for more details about the INMODEL= data set.

INWGT= Data Set

This data set enables you to specify a weight matrix other than the default matrix for the generalized, weighted, and diagonally weighted least squares estimation methods. If you also specify the INWGTINV option (or use the INWGT(INV)=option), the INWGT= data set is assumed to contain the inverse of the weight matrix, rather than the weight matrix itself. The specification of any INWGT= data set for unweighted least squares or maximum likelihood estimation is ignored. For generalized and diagonally weighted least squares estimation, the INWGT= data set must contain a _TYPE_ and a _NAME_ variable as well as the manifest variables used in the analysis. The value of the _NAME_ variable indicates the row index i of the weight w_{ij}. For weighted least squares, the INWGT= data set must contain _TYPE_, _NAME_, _NAM2_, and _NAM3_ variables as well as the manifest variables used in the analysis. The values of the _NAME_, _NAM2_, and _NAM3_ variables indicate the three indices i, j, k of the weight $w_{ij,kl}$. You can store information other than the weight matrix in the INWGT= data set, but only observations with _TYPE_=WEIGHT are used to specify the weight matrix $\mathbf{W}$. This property enables you to store more than one weight matrix in the INWGT= data set. You can then run PROC CALIS with each of the weight matrices by changing only the _TYPE_ observation in the INWGT= data set with an intermediate DATA step.

See the section "OUTWGT= Data Set" on page 1729 for more details about the INWGT= data set.

Output Data Sets

OUTEST= Data Set

The OUTEST= (or OUTVAR=) data set is of TYPE=EST and contains the final parameter estimates, the gradient, the Hessian, and boundary and linear constraints. For METHOD=ML (with or without the ROBUST option), METHOD=FIML, METHOD=GLS, and METHOD=WLS, the OUTEST= data set also contains the approximate standard errors, the information matrix (crossproduct Jacobian), and the approximate covariance matrix of the parameter estimates ((generalized) inverse of the information matrix). If there are linear or nonlinear equality or active inequality constraints at the solution, the OUTEST= data set also contains Lagrange multipliers, the projected Hessian matrix, and the Hessian matrix of the Lagrange function.

The OUTEST= data set can be used to save the results of an optimization by PROC CALIS for another analysis with either PROC CALIS or another SAS procedure. Saving results to an OUTEST= data set is advised for expensive applications that cannot be repeated without considerable effort.

The OUTEST= data set contains the BY variables, two character variables _TYPE_ and _NAME_, t numeric variables corresponding to the parameters used in the model, a numeric variable _RHS_ (right-hand side) that is used for the right-hand-side value b_i of a linear constraint or for the value $f = f(x)$ of the objective function at the final point x^* of the parameter space, and a numeric variable _ITER_ that is set to zero for initial values, set to the iteration number for the OUTITER output, and set to missing for the result output.

The _TYPE_ observations in Table 33.3 are available in the OUTEST= data set, depending on the request.

Table 33.3 _TYPE_ Observations in the OUTEST= Data Set

TYPE	Description
ACTBC	If there are active boundary constraints at the solution x^*, three observations indicate which of the parameters are actively constrained, as follows:
COV	Contains the approximate covariance matrix of the parameter estimates; used in computing the approximate standard errors.
COVRANK	contains the rank of the covariance matrix of the parameter estimates.
CRPJ_LF	Contains the Hessian matrix of the Lagrange function (based on CRPJAC).
CRPJAC	Contains the approximate Hessian matrix used in the optimization process. This is the inverse of the information matrix.

NAME	Description
GE	indicates the active lower bounds
LE	indicates the active upper bounds
EQ	indicates the active masks

Table 33.3 *continued*

TYPE	Description
EQ	If linear constraints are used, this observation contains the ith linear constraint $\sum_j a_{ij} x_j = b_i$. The parameter variables contain the coefficients a_{ij}, $j = 1, \ldots, n$, the _RHS_ variable contains b_i, and _NAME_=ACTLC or _NAME_=LDACTLC.
GE	If linear constraints are used, this observation contains the ith linear constraint $\sum_j a_{ij} x_j \geq b_i$. The parameter variables contain the coefficients a_{ij}, $j = 1, \ldots, n$, and the _RHS_ variable contains b_i. If the constraint i is active at the solution x^*, then _NAME_=ACTLC or _NAME_=LDACTLC.
GRAD	Contains the gradient of the estimates.
GRAD_LF	Contains the gradient of the Lagrange function. The _RHS_ variable contains the value of the Lagrange function.
HESSIAN	Contains the Hessian matrix.
HESS_LF	Contains the Hessian matrix of the Lagrange function (based on HESSIAN).
INFORMAT	Contains the information matrix of the parameter estimates (only for METHOD=ML, METHOD=GLS, or METHOD=WLS).
INITGRAD	Contains the gradient of the starting estimates.
INITIAL	Contains the starting values of the parameter estimates.
JACNLC	Contains the Jacobian of the nonlinear constraints evaluated at the final estimates.
LAGM BC	Contains Lagrange multipliers for masks and active boundary constraints.

NAME	Description
GE	Indicates the active lower bounds
LE	Indicates the active upper bounds
EQ	Indicates the active masks

Table 33.3 *continued*

TYPE	Description
LAGM LC	Contains Lagrange multipliers for linear equality and active inequality constraints in pairs of observations containing the constraint number and the value of the Lagrange multiplier.

NAME	Description
LEC_NUM	Number of the linear equality constraint
LEC_VAL	Corresponding Lagrange multiplier value
LIC_NUM	Number of the linear inequality constraint
LIC_VAL	Corresponding Lagrange multiplier value

LAGM NLC	contains Lagrange multipliers for nonlinear equality and active inequality constraints in pairs of observations that contain the constraint number and the value of the Lagrange multiplier.

NAME	Description
NLEC_NUM	Number of the nonlinear equality constraint
NLEC_VAL	Corresponding Lagrange multiplier value
NLIC_NUM	Number of the linear inequality constraint
NLIC_VAL	Corresponding Lagrange multiplier value

LE	If linear constraints are used, this observation contains the ith linear constraint $\sum_j a_{ij} x_j \leq b_i$. The parameter variables contain the coefficients a_{ij}, $j = 1, \ldots, n$, and the _RHS_ variable contains b_i. If the constraint i is active at the solution x^*, then _NAME_=ACTLC or _NAME_=LDACTLC.
LOWERBD | LB	If boundary constraints are used, this observation contains the lower bounds. Those parameters not subjected to lower bounds contain missing values. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
NACTBC	All parameter variables contain the number n_{abc} of active boundary constraints at the solution x^*. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
NACTLC	All parameter variables contain the number n_{alc} of active linear constraints at the solution x^* that are recognized as linearly independent. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.

Table 33.3 *continued*

TYPE	Description
NLC_EQ NLC_GE NLC_LE	Contains values and residuals of nonlinear constraints. The _NAME_ variable is described as follows: **_NAME_** — **Description** NLC — Inactive nonlinear constraint NLCACT — Linear independent active nonlinear constraint NLCACTLD — Linear dependent active nonlinear constraint
NLDACTBC	Contains the number of active boundary constraints at the solution x^* that are recognized as linearly dependent. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
NLDACTLC	Contains the number of active linear constraints at the solution x^* that are recognized as linearly dependent. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
NOBS	Contains the number of observations.
PARMS	Contains the final parameter estimates. The _RHS_ variable contains the value of the objective function.
PCRPJ_LF	Contains the projected Hessian matrix of the Lagrange function (based on CRPJAC).
PHESS_LF	Contains the projected Hessian matrix of the Lagrange function (based on HESSIAN).
PROJCRPJ	Contains the projected Hessian matrix (based on CRPJAC).
PROJGRAD	If linear constraints are used in the estimation, this observation contains the $n - n_{act}$ values of the projected gradient $\mathbf{g}_z = \mathbf{Z}'\mathbf{g}$ in the variables corresponding to the first $n - n_{act}$ parameters. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.
PROJHESS	Contains the projected Hessian matrix (based on HESSIAN).
STDERR	Contains approximate standard errors (only for METHOD=ML, METHOD=GLS, or METHOD=WLS).
TERMINAT	The _NAME_ variable contains the name of the termination criterion.
UPPERBD | UB	If boundary constraints are used, this observation contains the upper bounds. Those parameters not subjected to upper bounds contain missing values. The _RHS_ variable contains a missing value, and the _NAME_ variable is blank.

If the technique specified by the OMETHOD= option cannot be performed (for example, no feasible initial values can be computed or the function value or derivatives cannot be evaluated at the starting point), the OUTEST= data set can contain only some of the observations (usually only the PARMS and GRAD observations).

OUTMODEL= or OUTRAM= Data Set

The OUTMODEL= (or OUTRAM=) data set is of TYPE=CALISMDL and contains the model specification, the computed parameter estimates, and the standard error estimates. This data set is intended to be reused as an INMODEL= data set to specify good initial values in a subsequent analysis by PROC CALIS.

The OUTMODEL= data set contains the following variables:

- the BY variables, if any
- an _MDLNUM_ variable for model numbers, if used
- a character variable _TYPE_, which takes various values that indicate the type of model specification
- a character variable _NAME_, which indicates the model type, parameter name, or variable name
- a character variable _MATNR_, which indicates the matrix number (COSAN models only)
- a character variable _VAR1_, which is the name or number of the first variable in the specification
- a character variable _VAR2_, which is the name or number of the second variable in the specification
- a numerical variable _ESTIM_ for the final estimate of the parameter location
- a numerical variable _STDERR_ for the standard error estimate of the parameter location
- a numerical variable _SDEST_ for the final standardized estimate of the parameter location
- a numerical variable _SDSE_ for the standard error of the standardized estimate of the parameter location

Although the _SDEST_ and _SDSE_ variables are created for COSAN models, the values for these two variables are always missing because there are no rules to carry out the standardization of COSAN models.

Each observation (record) of the OUTMODEL= data set contains a piece of information regarding the model specification. Depending on the type of the specification indicated by the value of the _TYPE_ variable, the meanings of _NAME_, _VAR1_, and _VAR2_ differ. The following tables summarize the meanings of the _NAME_, _MATNR_ (COSAN models only), _VAR1_, and _VAR2_ variables for each value of the _TYPE_ variable, given the type of the model.

COSAN Models

TYPE=	Description	_NAME_	_MATNR_	_VAR1_	_VAR2_
MDLTYPE	Model type	COSAN			
VAR	Variable	Variable name	Matrix number	Column location	
MATRIX	Matrix	Matrix name	Matrix number	Number of rows	Number of columns
MODEL	Model formula	COV or MEAN	Matrix number	Term number	Location in term
ESTIM	Parameters	Parameter name	Matrix number	Row number	Column number

The value of the _NAME_ variable is COSAN for the _TYPE_=MDLTYPE observation.

The _TYPE_=VAR observations store the information about the column variables in matrices. The _NAME_ variable stores the variable names. The value of _VAR1_ indicates the column location of the variable in the matrix with the matrix number stored in _MATNR_.

The _TYPE_=MATRIX observations store the information about the model matrices. The _NAME_ variable stores the matrix names. The value of _MATNR_ indicates the corresponding matrix number. The values of_VAR1_ and _VAR2_ indicates the numbers of rows and columns, respectively, of the matrix.

The _TYPE_=MODEL observations store the covariance and mean structure formulas. The _NAME_ variable indicates whether the mean (MEAN) or covariance (COV) structure information is stored. The value of _MATNR_ indicates the matrix number in the mean or covariance structure formula. The _VAR1_ variable indicates the term number, and the _VAR2_ variable indicates the location of the matrix in the term.

The _TYPE_=ESTIM observations store the information about the parameters and their estimates. The _NAME_ variable stores the parameter names. The value of _MATNR_ indicates the matrix number. The values of _VAR1_ and _VAR2_ indicate the associated row and column numbers, respectively, of the parameter.

FACTOR Models

TYPE=	Description	_NAME_	_VAR1_	_VAR2_
MDLTYPE	Model type	Model type		
FACTVAR	Variable	Variable name	Variable number	Variable type
LOADING	Factor loading	Parameter name	Manifest variable	Factor variable
COV	Covariance	Parameter name	First variable	Second variable
PVAR	(Partial) variance	Parameter name	Variable	
MEAN	Mean or intercept	Parameter name	Variable	
ADDCOV	Added covariance	Parameter name	First variable	Second variable
ADDPVAR	Added (partial) variance	Parameter name	Variable	
ADDMEAN	Added mean or intercept	Parameter name	Variable	

For factor models, the value of the _NAME_ variable is either EFACTOR (exploratory factor model) or CFACTOR (confirmatory factor model) for the _TYPE_=MDLTYPE observation.

The _TYPE_=FACTVAR observations store the information about the variables in the model. The _NAME_ variable stores the variable names. The value of _VAR1_ indicates the variable number. The value of _VAR2_ indicates the type of the variable: either DEPV for dependent observed variables or INDF for latent factors.

Other observations specify the parameters and their estimates in the model. The _NAME_ values for these observations are the parameter names. Observation with _TYPE_=LOADING, _TYPE_=COV, or _TYPE_=ADDCOV are for parameters that are associated with two variables. The _VAR1_ and _VAR2_ values of these two types of observations indicate the variables involved.

Observations with _TYPE_=PVAR, _TYPE_=MEAN, _TYPE_=ADDPVAR, or _TYPE_=ADDMEAN are for parameters that are associated with a single variable. The value of _VAR1_ indicates the variable involved.

LINEQS Models

TYPE=	Description	_NAME_	_VAR1_	_VAR2_
MDLTYPE	Model type	LINEQS		
EQSVAR	Variable	Variable name	Variable number	Variable type
EQUATION	Path coefficient	Parameter	Outcome variable	Predictor variable
COV	Covariance	Parameter	First variable	Second variable
VARIANCE	Variance	Parameter	Variable	
MEAN	Mean	Parameter	Variable	
ADDCOV	Added covariance	Parameter	First variable	Second variable
ADDVARIA	Added variance	Parameter	Variable	
ADDINTE	Added intercept	Parameter	Variable	
ADDMEAN	Added mean	Parameter	Variable	

The value of the _NAME_ variable is LINEQS for the _TYPE_=MDLTYPE observation.

The _TYPE_=EQSVAR observations store the information about the variables in the model. The _NAME_ variable stores the variable names. The value of _VAR1_ indicates the variable number. The value of _VAR2_ indicates the type of the variable. There are six types of variables in the LINEQS model:

- DEPV for dependent observed variables
- INDV for independent observed variables
- DEPF for dependent latent factors
- INDF for independent latent factors
- INDD for independent error terms
- INDE for independent disturbance terms

Other observations specify the parameters and their estimates in the model. The _NAME_ values for these observations are the parameter names. Observation with _TYPE_=EQUATION, _TYPE_=COV, or _TYPE_=ADDCOV are for parameters that are associated with two variables. The _VAR1_ and _VAR2_ values of these two types of observations indicate the variables involved.

Observations with _TYPE_=VARIANCE, _TYPE_=MEAN, _TYPE_=ADDVARIA, _TYPE_=ADDINTE, or _TYPE_=ADDMEAN are for parameters associated with a single variable. The value of _VAR1_ indicates the variable involved.

LISMOD Models

TYPE=	Description	_NAME_	_VAR1_	_VAR2_
MDLTYPE	model type	LISMOD		
XVAR	$\mathbf{x}$-variable	Variable	Variable number	
YVAR	$\mathbf{y}$-variable	Variable	Variable number	
ETAVAR	$\boldsymbol{\eta}$-variable	Variable	Variable number	
XIVAR	$\boldsymbol{\xi}$-variable	Variable	Variable number	
ALPHA	_ALPHA_ entry	Parameter	Row number	
BETA	_BETA_ entry	Parameter	Row number	Column number
GAMMA	_BETA_ entry	Parameter	Row number	Column number
KAPPA	_KAPPA_ entry	Parameter	Row number	
LAMBDAX	_LAMBDAX_ entry	Parameter	Row number	Column number
LAMBDAY	_LAMBDAY_ entry	Parameter	Row number	Column number
NUX	_NUX_ entry	Parameter	Row number	
NUY	_NUY_ entry	Parameter	Row number	
PHI	_PHI_ entry	Parameter	Row number	Column number
PSI	_PSI_ entry	Parameter	Row number	Column number
THETAX	_THETAX_ entry	Parameter	Row number	Column number
THETAY	_THETAY_ entry	Parameter	Row number	Column number
ADDALPHA	Added _ALPHA_ entry	Parameter	Row number	
ADDKAPPA	Added _KAPPA_ entry	Parameter	Row number	
ADDNUX	Added _NUX_ entry	Parameter	Row number	
ADDNUY	Added _NUY_ entry	Parameter	Row number	
ADDPHI	Added _PHI_ entry	Parameter	Row number	Column number
ADDPSI	Added _PSI_ entry	Parameter	Row number	Column number
ADTHETAX	Added _THETAX_ entry	Parameter	Row number	Column number
ADTHETAY	Added _THETAY_ entry	Parameter	Row number	Column number

The value of the _NAME_ variable is LISMOD for the _TYPE_=MDLTYPE observation. Other observations specify either the variables or the parameters in the model.

Observations with _TYPE_ values equal to XVAR, YVAR, ETAVAR, and XIVAR indicate the variables in the respective lists in the model. The _NAME_ variable of these observations stores the names of the variables, and the _VAR1_ variable stores the variable numbers in the respective list. The variable numbers in this data set are not arbitrary—that is, they define the variable orders in the rows and columns of the LISMOD model matrices. The _VAR2_ variable of these observations is not used.

All other observations in this data set specify the parameters in the model. The _NAME_ values of these observations are the parameter names. The corresponding _VAR1_ and _VAR2_ values of these observations indicate the row and column locations of the parameters in the LISMOD model matrices that are specified in the _TYPE_ variable. For example, when the value of _TYPE_ is ADDPHI or PHI, the parameter specified is located in the _PHI_ matrix, with its row and column numbers indicated by the _VAR1_ and _VAR2_ values, respectively. Some observations for specifying parameters do not have values in the _VAR2_ variable. This means that the associated LISMOD matrices are vectors so that the column numbers are always 1 for these observations.

MSTRUCT Models

TYPE=	Description	_NAME_	_VAR1_	_VAR2_
MDLTYPE	Model type	MSTRUCT		
VAR	Variable	Variable	Variable number	
COVMAT	Covariance	Parameter	Row number	Column number
MEANVEC	Mean	Parameter	Row number	
ADCOVMAT	Added covariance	Parameter	Row number	Column number
AMEANVEC	Added mean	Parameter	Row number	

The value of the _NAME_ variable is MSTRUCT for the _TYPE_=MDLTYPE observation. Other observations specify either the variables or the parameters in the model.

Observations with _TYPE_ values equal to VAR indicate the variables in the model. The _NAME_ variable of these observations stores the names of the variables, and the _VAR1_ variable stores the variable numbers in the variable list. The variable numbers in this data set are not arbitrary—that is, they define the variable orders in the rows and columns of the mean and covariance matrices. The _VAR2_ variable of these observations is not used.

All other observations in this data set specify the parameters in the model. The _NAME_ values of these observations are the parameter names. The corresponding _VAR1_ and _VAR2_ values of these observations indicate the row and column locations of the parameters in the mean or covariance matrix, as specified in the _TYPE_ model. For example, when _TYPE_=COVMAT, the parameter specified is located in the covariance matrix, with its row and column numbers indicated by the _VAR1_ and _VAR2_ values, respectively. For observations with _TYPE_=MEANVEC, the _VAR2_ variable is not used because the column numbers are always 1 for parameters in the mean vector.

PATH Models

TYPE=	Description	_NAME_	_VAR1_	_VAR2_
MDLTYPE	Model type	PATH		
PATHVAR	Variable	Variable name	Variable number	Variable type
LEFT	Path coefficient	Parameter	Outcome variable	Predictor variable
RIGHT	Path coefficient	Parameter	Predictor variable	Outcome variable
PCOV	(Partial) covariance	Parameter	First variable	Second variable
PCOVPATH	(Partial) covariance path	Parameter	First variable	Second variable
PVAR	(Partial) variance	Parameter	Variable	
PVARPATH	(Partial) variance path	Parameter	Variable	Variable
MEAN	Mean or intercept	Parameter	Variable	
ONEPATH	Mean or intercept path	Parameter	_ONE_	Variable
ADDPCOV	Added (partial) covariance	Parameter	First variable	Second variable
ADDPVAR	Added (partial) variance	Parameter	Variable	
ADDMEAN	Added mean	Parameter	Variable	

The value of the _NAME_ variable is PATH for the _TYPE_=MDLTYPE observation.

The _TYPE_=PATHVAR observations store the information about the variables in the model. The _NAME_ variable stores the variable names. The value of _VAR1_ indicates the variable number. The value of _VAR2_ indicates the type of the variable. There are four types of variables in the PATH model:

- DEPV for dependent observed variables

- INDV for independent observed variables
- DEPF for dependent latent factors
- INDF for independent latent factors

Other observations specify the parameters in the model. The _NAME_ values for these observations are the parameter names. Observation with _TYPE_=LEFT, _TYPE_=RIGHT, _TYPE_=PCOV, or _TYPE_=ADDPCOV are for parameters that are associated with two variables. The _VAR1_ and _VAR2_ values of these two types of observations indicate the variables involved.

Observations with _TYPE_=PVAR, _TYPE_=MEAN, _TYPE_=ADDPVAR, or _TYPE_=ADDMEAN are for parameters that are associated with a single variable. The value of _VAR1_ indicates the variable involved.

RAM Models

TYPE=	Description	_NAME_	_VAR1_	_VAR2_
MDLTYPE	Model type	RAM		
RAMVAR	Variable name	Variable	Variable number	Variable type
_A_	_A_ entry	Parameter	Row number	Column number
_P_	_P_ entry	Parameter	Row number	Column number
_W_	_W_ entry	Parameter	Row number	Column number
ADD_P_	Added _P_ entry	Parameter	Row number	Column number
ADD_W_	Added _W_ entry	Parameter	Row number	Column number

The value of the _NAME_ variable is RAM for the _TYPE_=MDLTYPE observation.

For the _TYPE_=RAMVAR observations, the _NAME_ variable stores the variable names, the _VAR1_ variable stores the variable number, and the _VAR2_ variable stores the variable type. There are four types of variables in the PATH model:

- DEPV for dependent observed variables
- INDV for independent observed variables
- DEPF for dependent latent factors
- INDF for independent latent factors

Other observations specify the parameters in the model. The _NAME_ variable stores the parameter name. The _TYPE_ variable indicates the associated matrix with the row number indicated in _VAR1_ and column number indicated in _VAR2_.

Reading an OUTMODEL= Data Set As an INMODEL= Data Set in Subsequent Analyses

When the OUTMODEL= data set is treated as an INMODEL= data set in subsequent analyses, you need to pay attention to observations with _TYPE_ values prefixed by “ADD”, “AD”, or “A” (for example, ADDCOV, ADTHETAY, or AMEANVEC). These observations represent default parameter locations that are generated by PROC CALIS in a previous run. Because the context of the new analyses might be different, these observations for added parameter locations might no longer be suitable in the new runs. Hence, these observations are *not* read as input model information. Fortunately, after reading the INMODEL= specification

in the new analyses, CALIS analyzes the new model specification again. It then adds an appropriate set of parameters in the new context when necessary. If you are certain that the added parameter locations in the INMODEL= data set are applicable, you can force the input of these observations by using the READADDPARM option in the PROC CALIS statement. However, you must be very careful about using the READADDPARM option. The added parameters from the INMODEL= data set might have the same parameter names as those for the generated parameters in the new run. This might lead to unnecessary constraints in the model.

OUTSTAT= Data Set

The OUTSTAT= data set is similar to the TYPE=COV, TYPE=UCOV, TYPE=CORR, or TYPE=UCORR data set produced by the CORR procedure. The OUTSTAT= data set contains the following variables:

- the BY variables, if any
- the _GPNUM_ variable for groups numbers, if used in the analysis
- two character variables, _TYPE_ and _NAME_
- the manifest and the latent variables analyzed

The OUTSTAT= data set contains the following information (when available) in the observations:

- the mean and standard deviation
- the skewness and kurtosis (if the DATA= data set is a raw data set and the KURTOSIS option is specified)
- the number of observations
- if the WEIGHT statement is used, sum of the weights
- the correlation or covariance matrix to be analyzed
- the robust covariances, standard deviations, and means for robust estimation
- the predicted correlation or covariance matrix
- the standardized or normalized residual correlation or covariance matrix
- if the model contains latent variables, the predicted covariances between latent and manifest variables and the latent variable (or factor) score regression coefficients (see the PLATCOV option on page 1571)

In addition, for FACTOR models the OUTSTAT= data set contains:

- the unrotated factor loadings, the error variances, and the matrix of factor correlations
- the standardized factor loadings and factor correlations
- the rotation matrix, rotated factor loadings, and factor correlations

- standardized rotated factor loadings and factor correlations

If effects are analyzed, the OUTSTAT= data set also contains:

- direct, indirect, and total effects and their standard error estimates
- standardized direct, indirect, and total effects and their standard error estimates

Each observation in the OUTSTAT= data set contains some type of statistic as indicated by the _TYPE_ variable. The values of the _TYPE_ variable are shown in the following tables:

Basic Descriptive Statistics

Value of _TYPE_	Contents
ADJCOV	Adjusted covariances
ADJSTD	Adjusted standard deviations
CORR	Correlations
COV	Covariances
KURTOSIS	Univariate kurtosis
MEAN	Means
N	Sample size
NPARTIAL	Number of partial variables
PARTCOV	Covariances after partialling
PARTCORR	Correlations after partialling
PARTMEAN	Means after partialling
PARTSTD	Standard deviations after partialling
ROBCOV	Robust covariances
ROBMEAN	Robust means
ROBSTD	Robust standard deviations
SKEWNESS	Univariate skewness
STD	Standard deviations
SUMWGT	Sum of weights (if the WEIGHT statement is used)
VARDIV	Variance divisor
VARDIVAJ	Variance divisor adjustment

For the _TYPE_=CORR or COV observations, the _NAME_ variable contains the name of the manifest variable that corresponds to each row for the covariance or correlation. For other observations, _NAME_ is blank.

Predicted Moments and Residuals

Value of _TYPE_	Contents
METHOD=DWLS	
DWLSPRED	DWLS predicted moments
DWLSRES	DWLS raw residuals
DWLSSRES	DWLS variance standardized residuals
METHOD=GLS	
GLSASRES	GLS asymptotically standardized residuals
GLSNRES	GLS normalized residuals
GLSPRED	GLS predicted moments
GLSRES	GLS raw residuals
GLSSRES	GLS variance standardized residuals
METHOD=ML or FIML	
MAXASRES	ML asymptotically standardized residuals
MAXNRES	ML normalized residuals
MAXPRED	ML predicted moments
MAXRES	ML raw residuals
MAXSRES	ML variance standardized residuals
METHOD=ULS	
ULSPRED	ULS predicted moments
ULSRES	ULS raw residuals
ULSSRES	ULS variance standardized residuals
METHOD=WLS	
WLSASRES	WLS asymptotically standardized residuals
WLSNRES	WLS normalized residuals
WLSPRED	WLS predicted moments
WLSRES	WLS raw residuals
WLSSRES	WLS variance standardized residuals

For residuals or predicted moments of means, the _NAME_ variable is a fixed value denoted by _Mean_. For residuals or predicted moments for covariances or correlations, the _NAME_ variable is used for names of variables.

Effects and Latent Variable Scores Regression Coefficients

Value of _TYPE_	Contents
Unstandardized Effects	
DEFFECT	Direct effects
DEFF_SE	Standard error estimates for direct effects
IEFFECT	Indirect effects
IEFF_SE	Standard error estimates for indirect effects
TEFFECT	Total effects
TEFF_SE	Standard error estimates for total effects
Standardized Effects	
SDEFF	Standardized direct effects
SDEFF_SE	Standard error estimates for standardized direct effects
SIEFF	Standardized indirect effects
SIEFF_SE	Standard error estimates for standardized indirect effects
STEFF	Standardized total effects
STEFF_SE	Standard error estimates for standardized total effects
Latent Variable Scores Coefficients	
LSSCORE	Latent variable (or factor) scores regression coefficients for ULS method
SCORE	Latent variable (or factor) scores regression coefficients other than ULS method

For latent variable or factor scores coefficients, the _NAME_ variable contains factor or latent variables in the observations. For other observations, the _NAME_ variable contains manifest or latent variable names.

You can use the latent variable score regression coefficients with PROC SCORE to compute factor scores. If the analyzed matrix is a covariance rather than a correlation matrix, the _TYPE_=STD observation is not included in the OUTSTAT= data set. In this case, the standard deviations can be obtained from the diagonal elements of the covariance matrix. Dropping the _TYPE_=STD observation prevents PROC SCORE from standardizing the observations before computing the factor scores.

Factor Analysis Results

Value of _TYPE_	Contents
ERRVAR	Error variances
FCOV	Factor correlations or covariances
LOADINGS	Unrotated factor loadings
RFCOV	Rotated factor correlations or covariances
RLOADING	Rotated factor loadings
ROTMAT	Rotation matrix
STDERVAR	Error variances in standardized solutions
STDFCOV	Standardized factor correlations
STDLOAD	Standardized factor loadings
STDRFCOV	Standardized rotated factor correlations or covariances
STDRLOAD	Standardized rotated factor loadings

For the _TYPE_=ERRVAR observation, the _NAME_ variable is blank. For all other observations, the _NAME_ variable contains factor names.

OUTWGT= Data Set

You can create an OUTWGT= data set that is of TYPE=WEIGHT and contains the weight matrix used in generalized, weighted, or diagonally weighted least squares estimation. The OUTWGT= data set contains the weight matrix on which the WRIDGE= and the WPENALTY= options are applied. However, if you input the inverse of the weight matrix with the INWGT= and INWGTINV options (or the INWGT(INV)= option alone) in the same analysis, the OUTWGT= data set contains the same elements of the inverse of the weight matrix. For unweighted least squares or maximum likelihood estimation, no OUTWGT= data set can be written. The weight matrix used in maximum likelihood estimation is dynamically updated during optimization. When the ML solution converges, the final weight matrix is the same as the predicted covariance or correlation matrix, which is included in the OUTSTAT= data set (observations with _TYPE_ =MAXPRED).

For generalized and diagonally weighted least squares estimation, the weight matrices $\mathbf{W}$ of the OUTWGT= data set contain all elements w_{ij}, where the indices i and j correspond to all manifest variables used in the analysis. Let $varnam_i$ be the name of the ith variable in the analysis. In this case, the OUTWGT= data set contains n observations with the variables shown in the following table:

Variable	Contents
TYPE	WEIGHT (character)
NAME	Name of variable $varnam_i$ (character)
$varnam_1$	Weight w_{i1} for variable $varnam_1$ (numeric)
$\vdots$	$\vdots$
$varnam_n$	Weight w_{in} for variable $varnam_n$ (numeric)

For weighted least squares estimation, the weight matrix $\mathbf{W}$ of the OUTWGT= data set contains only the nonredundant elements $w_{ij,kl}$. In this case, the OUTWGT= data set contains $n(n+1)(2n+1)/6$ observations with the variables shown in the following table:

Variable	Contents
TYPE	WEIGHT (character)
NAME	Name of variable $varnam_i$ (character)
NAM2	Name of variable $varnam_j$ (character)
NAM3	Name of variable $varnam_k$ (character)
$varnam_1$	Weight $w_{ij,k1}$ for variable $varnam_1$ (numeric)
$\vdots$	$\vdots$
$varnam_n$	Weight $w_{ij,kn}$ for variable $varnam_n$ (numeric)

Symmetric redundant elements are set to missing values.

OUTFIT= Data Set

You can create an OUTFIT= data set that is of TYPE=CALISFIT and that contains the values of the fit indices of your analysis. If you use two estimation methods such as LSML or LSWLS, the fit indices are for the second analysis. An OUTFIT=data set contains the following variables:

- a character variable _TYPE_ for the types of fit indices
- a numerical variable IndexCode for the codes of the fit indices

- a character variable FitIndex for the names of the fit indices
- a numerical variable FitValue for the numerical values of the fit indices
- a character variable PrintChar for the character-formatted fit index values.

The possible values of _TYPE_ are:

ModelInfo:	basic modeling statistics and information
Absolute:	stand-alone fit indices
Parsimony:	fit indices that take model parsimony into account
Incremental:	fit indices that are based on comparison with a baseline model

Possible Values of FitIndex When _TYPE_=ModelInfo

Value of FitIndex	Description
Number of Observations	Number of observations used in the analysis
Number of Complete Observations	Number of complete observations (METHOD=FIML)
Number of Incomplete Observations	Number of incomplete observations (METHOD=FIML)
Number of Variables	Number of variables
Number of Moments	Number of mean or covariance elements
Number of Parameters	Number of parameters
Number of Active Constraints	Number of active constraints in the solution
Saturated Model Estimation	Estimation status of the saturated model (METHOD=FIML)
Saturated Model Function Value	Saturated model function value (METHOD=FIML)
Saturated Model -2 Log-Likelihood	Saturated model –2 log-likelihood function value (METHOD=FIML)
Baseline Model Estimation	Estimation status of the baseline model (METHOD=FIML)
Baseline Model Function Value	Baseline model function value
Baseline Model -2 Log-Likelihood	Baseline model –2 log-likelihood function value (METHOD=FIML)
Baseline Model Chi-Square	Baseline model chi-square value
Baseline Model Chi-Square DF	Baseline model chi-square degrees of freedom
Baseline Model DF	Baseline model degrees of freedom (METHOD=ULS or METHOD=DWLS)
Pr > Baseline Model Chi-Square	*p*-value of the baseline model chi-square
SB-Scaled Base Model Chi-Square	Satorra-Bentler scaled chi-square for the baseline model
Pr > SB-Scaled Base Model Chi-Square	*p*-value of the Satorra-Bentler scaled chi-square for the baseline model

Possible Values of FitIndex When _TYPE_=Absolute

Value of FitIndex	Description
Fit Function	Fit function value
-2 Log-Likelihood	–2 log-likelihood function value for the model (METHOD=FIML)
Chi-Square	Model chi-square value
Chi-Square DF	Degrees of freedom for the model chi-square test
Model DF	Degrees of freedom for model (METHOD=ULS or METHOD=DWLS)
Pr > Chi-Square	Probability of obtaining a larger chi-square than the observed value
SB-Scaled Model Chi-Square	Satorra-Bentler scaled chi-square for the model
Pr > SB-Scaled Model Chi-Square	Probability of obtaining a larger chi-square than the observed Satorra-Bentler scaled chi-square value
Elliptic Corrected Chi-Square	Elliptic-corrected chi-square value
Pr > Elliptic Corr. Chi-Square	Probability of obtaining a larger elliptic-corrected chi-square value
Z-test of Wilson and Hilferty	Z-test of Wilson and Hilferty
Hoelter Critical N	N value that makes a significant chi-square when multiplied to the fit function value
Root Mean Square Residual (RMR)	Root mean square residual
Standardized RMR (SRMR)	Standardized root mean square residual
Goodness of Fit Index (GFI)	Jöreskog and Sörbom goodness-of-fit index

Possible Values of FitIndex When _TYPE_=Parsimony

Value of FitIndex	Description
Adjusted GFI (AGFI)	Goodness-of-fit index adjusted for the degrees of freedom of the model
Parsimonious GFI	Mulaik et al. (1989) modification of the GFI
RMSEA Estimate	Steiger and Lind (1980) root mean square error of approximation
RMSEA Lower **r%** Confidence Limit	Lower **r%**[1] confidence limit for RMSEA
RMSEA Upper **r%** Confidence Limit	Upper **r%**[1] confidence limit for RMSEA
Probability of Close Fit	Browne and Cudeck (1993) test of close fit
ECVI Estimate	Expected cross-validation index
ECVI Lower **r%** Confidence Limit	Lower **r%**[2] confidence limit for ECVI
ECVI Upper **r%** Confidence Limit	Upper **r%**[2] confidence limit for ECVI
Akaike Information Criterion	Akaike information criterion
Bozdogan CAIC	Bozdogan (1987) consistent AIC
Schwarz Bayesian Criterion	Schwarz (1978) Bayesian criterion
McDonald Centrality	McDonald and Marsh (1988) measure of centrality

1. The value of **r** is one minus the ALPHARMS= value. By default, **r**=90.
2. The value of **r** is one minus the ALPHAECV= value. By default, **r**=90.

Possible Values of FitIndex When _TYPE_=Incremental

Value of FitIndex	Description
Bentler Comparative Fit Index	Bentler (1985) comparative fit index
Bentler-Bonett NFI	Bentler and Bonett (1980) normed fit index
Bentler-Bonett Non-normed Index	Bentler and Bonett (1980) nonnormed fit index
Bollen Normed Index Rho1	Bollen normed ρ_1
Bollen Non-normed Index Delta2	Bollen nonnormed δ_2
James et al. Parsimonious NFI	James, Mulaik, and Brett (1982) parsimonious normed fit index

Default Analysis Type and Default Parameterization

This section describes the default analysis type and default parameterization of PROC CALIS. Because various types of models supported by PROC CALIS have their own specific features, the following table outlines only those major default settings of PROC CALIS:

Moment Structures Analyzed	
Covariance structures	Covariance structures are analyzed by default (COV option).
Mean structures	Not analyzed by default. Use the MEANSTR option, the 'Intercept' variable in the LINEQS statement, the MEAN statement, or specific MATRIX statements to specify mean or intercept parameters.
Default Parameterization	
Variance parameters	Free variance parameters for independent latent factors, observed variables, and errors.
Covariance parameters	Free covariance parameters for all pairs of independent latent factors and observed variables (except for the latent factors in exploratory factor models). Fixed zero covariances for all pairs of error variables, all pairs between errors and other exogenous variables, and all pairs of latent factors in exploratory factor models.
Mean parameters	Free mean parameters for all exogenous observed variables. Fixed zero means for all exogenous latent factors and error terms.
Intercept parameters	Free intercept parameters for all endogenous observed variables; Fixed zero intercepts for all endogenous latent factors.
Role of the VAR Statement in Model Specification	
Inclusion of Variables	If the VAR statement is used, observed variables that are listed in the VAR statement but not mentioned in corresponding model specification are included as exogenous variables in the model. Default variance, covariance, and mean parameterization apply to these exogenous variables.

Exclusion of Variables	If the VAR statement is used, observed variables that are not listed in the VAR statement are not recognized as observed variables in model specification. They might be treated as latent variables instead.

In general, the default settings of PROC CALIS are consistent with conventional practice of structural equation modeling. For example, because statistical theory of structural equation modeling is based mainly on the analysis of covariance structures, the default analysis type of PROC CALIS is for covariance structures. This means that the CORR option must be specified if you want to analyze correlation matrices. Mean structures in PROC CALIS are parameterized naturally so that users can make statistical inferences directly on the mean and intercept estimates. Default variance and covariance parameters for all independent factors and observed variables are implemented to reflect the common belief that no exogenous variables are absolutely uncorrelated (except for the unrotated factors in the initial solution of exploratory FACTOR models).

For details about the default parameters in specific types of model, see the FACTOR, LINEQS, LISMOD, PATH, and RAM statements. Because there are no explicit latent variables and the exogenous/endogenous variable distinction is not used in the COSAN and MSTRUCT modeling languages, the default parameterization outlined in the preceding table does not apply to these two types of models. See the COSAN and the MSTRUCT statements for details. See the following sections for details about model parameterization in various types of models:

- "The COSAN Model" on page 1734
- "The FACTOR Model" on page 1738
- "The LINEQS Model" on page 1745
- "The LISMOD Model and Submodels" on page 1752
- "The PATH Model" on page 1763
- "The RAM Model" on page 1769
- "The MSTRUCT Model" on page 1761

By default, PROC CALIS treats observed variables specified in the VAR statement as exogenous variables in the FACTOR, PATH, LINEQS, and RAM models. This minimizes the routine specifications in the COV, MEAN, PCOV, PVAR, or VARIANCE for those "standalone" exogenous observed variables— observed variables that are not functionally related to any other variables in the model. Including these standalone exogenous observed variables into the model is useful during the model modification process (by the MODIFICATION option) where you can do Lagrange multiplier tests to see whether relating these variables to other variables can improve the model fit.

Notice that the use of the VAR statement specification is optional in PROC CALIS. If you have specified all the observed variables that you want in the model specification statements, it is not necessary to use the VAR statement specifications for the purpose of specifying the set of observed variable for analysis.

The COSAN Model

The original COSAN (covariance structure analysis) model is proposed by McDonald (1978, 1980) for analyzing general covariance structure models. PROC CALIS enables you to analyze a generalized form of the original COSAN model. The generalized COSAN model extends the original COSAN model with the inclusion of addition terms in the covariance structure formula and the associated mean structure formula.

The covariance structure formula of the generalized COSAN model is

$$\mathbf{\Sigma} = \mathbf{F}_1\mathbf{P}_1\mathbf{F}_1' + \cdots + \mathbf{F}_m\mathbf{P}_m\mathbf{F}_m'$$

and the corresponding mean structure formula of the generalized COSAN model is

$$\boldsymbol{\mu} = \mathbf{F}_1\mathbf{v}_1 + \cdots + \mathbf{F}_m\mathbf{v}_m$$

where $\mathbf{\Sigma}$ is a symmetric correlation or covariance matrix for the observed variables, $\boldsymbol{\mu}$ is a vector for the observed variable means, each $\mathbf{P}_k$ is a symmetric matrix, each $\mathbf{v}_k$ is a mean vector, and each $\mathbf{F}_k$ ($k = 1, \ldots, m$,) is the product of $n(k)$ matrices $\mathbf{F}_{k_1}, \ldots, \mathbf{F}_{k_{n(k)}}$; that is,

$$\mathbf{F}_k = \mathbf{F}_{k_1} \cdots \mathbf{F}_{k_{n(k)}}, \quad k = 1, \ldots, m$$

The matrices $\mathbf{F}_{k_j}$ and $\mathbf{P}_k$ in the model can be one of the forms

$$\mathbf{F}_{k_j} = \begin{cases} \mathbf{G}_{k_j} \\ \mathbf{G}_{k_j}^{-1} \\ (\mathbf{I} - \mathbf{G}_{k_j})^{-1} \end{cases} \quad j = 1, \ldots, n(k) \qquad \text{and} \qquad \mathbf{P}_k = \begin{cases} \mathbf{Q}_k \\ \mathbf{Q}_k^{-1} \end{cases}$$

where $\mathbf{G}_{k_j}$ and $\mathbf{Q}_k$ are basic model matrices that are not expressed as functions of other matrices.

The COSAN model matrices and vectors are $\mathbf{G}_{k_j}$, $\mathbf{Q}_k$, and $\mathbf{v}_k$ (when the mean structures are analyzed). The elements of these model matrices and vectors are either parameters (free or constrained) or fixed values. Matrix $\mathbf{P}_k$ is referred to as the central covariance matrix for the kth term in the covariance structure formula.

Essentially, the COSAN modeling language enables you to define the covariance and mean structure formulas of the generalized COSAN model, the basic COSAN model matrices $\mathbf{G}_{k_j}$, $\mathbf{Q}_k$, and $\mathbf{v}_k$, and the parameters and fixed values in the model matrices.

You can also specify a generalized COSAN model without using an explicit central covariance matrix in any term. For example, you can define the kth term in the covariance structure formula as

$$\mathbf{F}_k\mathbf{F}_k' = \mathbf{F}_{k_1} \ldots \mathbf{F}_{k_{n-1}}\mathbf{F}_{k_n}\mathbf{F}_{k_n}'\mathbf{F}_{k_{n-1}}' \ldots \mathbf{F}_{k_1}'$$

The corresponding term for the mean structure becomes

$$\mathbf{F}_{k_1} \ldots \mathbf{F}_{k_{n-1}}\mathbf{v}_m$$

In the covariance structure formula, $\mathbf{F}_{k_n}\mathbf{F}_{k_n}'$ serves as an implicit central covariance matrix in this term of the covariance structure formula. Because of this, $\mathbf{F}_{k_n}$ does not appear in the corresponding mean structure formula.

To take advantage of the modeling flexibility of the COSAN model specifications, you are required to provide the correct covariance and mean structure formulas for the analysis problem. If you are not familiar with the mathematical formulations of structural equation models, you can consider using simpler modeling languages such as PATH or LINEQS.

An Example: Specifying a Second-Order Factor Model

This example illustrates how to specify the covariance structures in the COSAN statement. Consider a second-order factor analysis model with the following formula for the covariance structures of observed variables v1–v9

$$\mathbf{\Sigma} = \mathbf{F}_1(\mathbf{F}_2\mathbf{P}_2\mathbf{F}_2' + \mathbf{U}_2)\mathbf{F}_1' + \mathbf{U}_1$$

where $\mathbf{F}_1$ is a 9×3 first-order factor matrix, $\mathbf{F}_2$ is a 3×2 second-order factor matrix, $\mathbf{P}_2$ is a 2×2 covariance matrix for the second-order factors, $\mathbf{U}_2$ is a 3×3 diagonal matrix for the unique variances of the first-order factors, and $\mathbf{U}_1$ is a 9×9 diagonal matrix for the unique variances of the observed variables.

To fit this covariance structure model, you first rewrite the covariance structure formula in the form of the generalized COSAN model as

$$\mathbf{\Sigma} = \mathbf{F}_1\mathbf{F}_2\mathbf{P}_2\mathbf{F}_2'\mathbf{F}_1' + \mathbf{F}_1\mathbf{U}_2\mathbf{F}_1' + \mathbf{U}_1$$

You can specify the list of observed variables and the three terms for the covariance structure formula in the following COSAN statement:

```
cosan var= v1-v9,
      F1(3) * F2(2) * P2(2,SYM) + F1(3) * U2(3,DIA) + U1(9,DIA);
```

The VAR= option specifies the nine observed variables in the model. Next, the three terms of the covariance structure formula are specified. Because each term in the covariance structure formula is a symmetric product, you only need to specify each term up to the central covariance matrix. For example, although the first term in the covariance structure formula is $\mathbf{F}_1\mathbf{F}_2\mathbf{P}_2\mathbf{F}_2'\mathbf{F}_1'$, you only need to specify **F1(3) * F2(2) * P2(2,SYM)**. PROC CALIS generates the redundant information for the term. Similarly, you specify the other two terms of the covariance structure formula.

In each matrix specification of the COSAN statement, you can specify the following three matrix properties as the arguments in the trailing parentheses: the number of columns, the matrix type, and the transformation of the matrix. For example, **F1(3)** means that the number of columns of **F1** is 3 (while the number of rows is 9 because this number has to match the number of observed variables specified in the VAR= option), **F2(2)** means that the number of columns of **F2** is 2 (while the number of rows is 3 because the number has to match the number of columns of the preceding matrix, **F1**). You can specify the type of the matrix in the second argument. For example, **P2(2,SYM)** means that **P2** is a symmetric (SYM) matrix and **U2(2,DIA)** means that **U2** is a diagonal (DIA) matrix. You can also specify the transformation of the matrix in the third argument. Because there is no transformation needed in the current second-order factor model, this argument is omitted in the specification. See the COSAN statement for details about the matrix types and transformation that are supported by the COSAN modeling language.

Suppose now you also want to analyze the mean structures of the second-order factor model. The corresponding mean structure formula is

$$\boldsymbol{\mu} = \mathbf{F}_1\mathbf{F}_2\mathbf{v} + \mathbf{u}$$

where $\mathbf{v}$ is a 2×1 mean vector for the second-order factors and $\mathbf{u}$ is a 6×1 vector for the intercepts of the observed variables. To analyze the mean and covariance structures simultaneously, you can use the following COSAN statement:

```
cosan var= v1-v9,
      F1(3) * F2(2) * P2(2,SYM) [mean = v] + F1(3) * U2(3,DIA)
      + U1(9,DIA) [mean = u];
```

In addition to the covariance structure specified, you now add the trailing MEAN= options in the first and the third terms. PROC CALIS then generates the mean structure formula by the following steps:

- Remove the last matrix (that is, the central covariance matrix) in each term of the covariance structure formula.
- Append to each term the vector that is specified in the MEAN= option of the term, or if no MEAN= option is specified in a term, that term becomes a zero vector in the mean structure formula.

Following these steps, the mean structure formula generated for the second-order factor model is

$$\boldsymbol{\mu} = \mathbf{F}_1\mathbf{F}_2\mathbf{v} + 0 + \mathbf{u}$$

which is what you expect for the mean structures of the second-order factor model. To complete the COSAN model specification, you can use MATRIX statements to specify the parameters and fixed values in the COSAN model matrices. See Example 33.29 for a complete example.

Special Cases of the Generalized COSAN Model

It is illustrative to see how you can view different types of models as a special case of the generalized COSAN model. This section describes two such special cases.

The Original COSAN Model

The original COSAN (covariance structure analysis) model (McDonald 1978, 1980) specifies the following covariance structures:

$$\boldsymbol{\Sigma} = \mathbf{F}_1 \cdots \mathbf{F}_n \mathbf{P} \mathbf{F}_n' \cdots \mathbf{F}_1'$$

This is the generalized COSAN with only one term for the covariance structure model formula. Hence, using the COSAN statement to specify the original COSAN model is straightforward.

Reticular Action Model

The RAM (McArdle 1980; McArdle and McDonald 1984) model fits the covariance structures

$$\boldsymbol{\Sigma}_a = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{P}(\mathbf{I} - \mathbf{A})^{-1\prime}$$

where $\boldsymbol{\Sigma}_a$ is the symmetric covariance for all latent and observed variables in the RAM model, $\mathbf{A}$ is a square matrix for path coefficients, $\mathbf{I}$ is an identity matrix with the same dimensions as $\mathbf{A}$, and $\mathbf{P}$ is a symmetric covariance matrix. For details about the RAM model, see the section "The RAM Model" on page 1769.

Correspondingly, the RAM model fits the mean structure formula

$$\boldsymbol{\mu}_a = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{w}$$

where $\boldsymbol{\mu}_a$ is the mean vector for all latent and observed variables in the RAM model and $\mathbf{w}$ is a vector for mean or intercepts of the variables.

To extract the covariance and mean structures for the observed variables, a selection matrix $\mathbf{G}$ is used. The selection matrix $\mathbf{G}$ contains zeros and ones as its elements. Each row of $\mathbf{G}$ has exactly one nonzero element at the position that corresponds to the location of a manifest row variable in $\boldsymbol{\Sigma}_a$ or $\boldsymbol{\mu}_a$. The covariance structure formula for the observed variables in the RAM model becomes

$$\boldsymbol{\Sigma} = \mathbf{G}(\mathbf{I} - \mathbf{A})^{-1}\mathbf{P}(\mathbf{I} - \mathbf{A})^{-1\prime}\mathbf{G}'$$

The mean structure formula for the observed variables in the RAM model becomes

$$\boldsymbol{\mu} = \mathbf{G}(\mathbf{I} - \mathbf{A})^{-1}\mathbf{w}$$

These formulas suggest that the RAM model is special case of the generalized COSAN model with one term. For example, suppose that there are 10 observed variables (var1–var10) and 3 latent variables in a RAM model. The following COSAN statement represents the RAM model:

```
cosan var= v1-v10,
      G(13,GEN) * A(13,GEN,IMI) * P(13,SYM) [Mean = w];
```

In the COSAN statement, you define the 10 variables in the VAR= option. Next, you provide the formulas for the mean and covariance structures. $\mathbf{G}$ is 10×13 general matrix (GEN), $\mathbf{A}$ is a 13×13 general matrix with the IMI transformation (that is, $(\mathbf{I} - \mathbf{A})^{-1}$), $\mathbf{P}$ is a 13×13 symmetric matrix (SYM), and $\mathbf{w}$ is a 13×1 vector. With these COSAN statement specifications, your mean and covariance structure formulas represent exactly those of the RAM model. To complete the entire model specification, your next step is to use the MATRIX statements to specify the parameters and fixed values in the model matrices $\mathbf{G}$, $\mathbf{A}$, $\mathbf{P}$, and $\mathbf{w}$.

Similarly, it is possible to use the COSAN modeling language to represent any other model types such as models defined by the FACTOR, LINEQS, LISMOD, MSTRUCT, PATH, and RAM statements. But this is not an automatic recommendation of using the COSAN modeling languages in all situations. When an analysis can be specified by either the COSAN or a more specific modeling language (for example, PATH), you should consider using the specific modeling language because the specific modeling language can exploit specific model features so that it does the following:

- enables more supplemental analysis (effect analysis, standardized solutions, and so on), which COSAN has no general way to display
- supports better initial estimation methods (the COSAN model can only set initial estimates to certain default or random values)
- leads to more efficient computations due to the availability of more specific formulas and algorithms

Certainly, the COSAN modeling language is still very useful when you fit some nonstandard model structures that cannot be handled otherwise by the more specific modeling languages.

Naming Variables in the COSAN Model

Although you can define the list of observed (manifest) variables in the VAR= option of the COSAN statement, the COSAN modeling language does not support a direct specification of the latent or error variables in the model. In the COSAN statement, you can define the model matrices and how they multiply together to form the covariance and mean structures. However, except for the row variables of the first matrix in each term, you do not need to *identify* the row and column variables in all other matrices. However, you can use

the VARNAMES statement to *label* the column variables of the matrices. The names in the VARNAMES statement follow the general naming rules required by the general SAS system. They should not contain special characters and cannot be longer than 32 characters. Also, they do not need to use certain prefixes like what the LINEQS modeling language requires. It is important to realize that the VARNAME statement only *labels*, but does not *identify*, the column variables (and the row variables, by propagation). This means that while keeping all other things equal, changing the names in the VARNAMES statements does not change the mathematical model or the estimation of the model. For example, you can label all columns of a COSAN matrix with the same name but it does not mean that these columns refer to the same variable in the model. See the section "Naming Variables and Parameters" on page 1777 for the general rules about naming variables and parameters.

Default Parameters in the COSAN Model

The default parameters of the COSAN model matrices depend on the types of the matrices. Each element of the IDE or ZID matrix (identity matrix with or without an additional zero matrix) is either a fixed one or a fixed zero. You cannot override the default parameter values of these fixed matrices. For COSAN model matrices with types other than IDE or ZID, all elements are fixed zeros by default. You can override these default zeros by specifying them explicitly in the MATRIX statements.

The FACTOR Model

The FACTOR modeling language is used for specifying exploratory and confirmatory factor analysis models. You can use other general modeling languages such as LINEQS, LISMOD, PATH, and RAM to specify a factor model. But the FACTOR modeling language is more convenient for specifying factor models and is more specialized in displaying factor-analytic results. For convenience, models specified by the FACTOR modeling language are called FACTOR models.

Types of Variables in the FACTOR Model

Each variable in the FACTOR model is either manifest or latent. Manifest variables are those variables that are measured in the research. They must be present in the input data set. Latent variables are not directly observed. Each latent variable in the FACTOR model can be either a factor or an error term.

Factors are unmeasured hypothetical constructs for explaining the covariances among manifest variables, while errors are the unique parts of the manifest variables that are not explained by the (common) factors.

In the FACTOR model, all manifest variables are endogenous, which means that they are predicted from the latent variables. In contrast, all latent variables in the FACTOR model are exogenous, which means that they serve as predictors only.

Naming Variables in the FACTOR Model

Manifest variables in the FACTOR model are referenced in the input data set. In the FACTOR model specification, you use their names as they appear in the input data set. Manifest variable names must not be longer than 32 characters. There are no further restrictions on these names beyond those required by the SAS System.

Error variables in the FACTOR model are not named explicitly, although they are assumed in the model. You can name latent factors only in confirmatory FACTOR models. Factor names must not be longer than 32

characters and must be distinguishable from the manifest variable names in the same analysis. You do not need to name factors in exploratory FACTOR models, however. Latent factors named Factor1, Factor2, and so on are generated automatically in exploratory FACTOR models.

Model Matrices in the FACTOR Model

Suppose in the FACTOR model that there are p manifest variables and n factors. The FACTOR model matrices are described in the following subsections.

Matrix $\mathbf{F}$ *(*$p \times n$*) : Factor Loading Matrix*

The rows of $\mathbf{F}$ represent the p manifest variables, while the columns represent the n factors. Each row of $\mathbf{F}$ contains the factor loadings of a variable on all factors in the model.

Matrix $\mathbf{P}$ *(*$n \times n$*) : Factor Covariance Matrix*

The $\mathbf{P}$ matrix is a symmetric matrix for the variances of and covariances among the n factors.

Matrix $\mathbf{U}$ *(*$p \times p$*) : Error Covariance Matrix*

The $\mathbf{U}$ matrix represents a $p \times p$ diagonal matrix for the error variances for the manifest variables. Elements in this matrix are the parts of variances of the manifest variables that are not explained by the common factors. Note that all off-diagonal elements of $\mathbf{U}$ are fixed zeros in the FACTOR model.

Vector $\mathbf{a}$ *(*$p \times 1$*) : Intercepts*

If the mean structures are analyzed, vector $\mathbf{a}$ represents the intercepts of the manifest variables.

Vector $\mathbf{v}$ *(*$n \times 1$*) : Factor Means*

If the mean structures are analyzed, vector $\mathbf{v}$ represents the means of the factors.

Matrix Representation of the FACTOR Model

Let $\mathbf{y}$ be a $p \times 1$ vector of manifest variables, $\boldsymbol{\xi}$ be an $n \times 1$ vector of latent factors, and $\mathbf{e}$ be a $p \times 1$ vector of errors. The factor model is written as

$$\mathbf{y} = \mathbf{a} + \mathbf{F}\boldsymbol{\xi} + \mathbf{e}$$

With the model matrix definitions in the previous section, the covariance matrix $\boldsymbol{\Sigma}$ $(p \times p)$ of manifest variables is structured as

$$\boldsymbol{\Sigma} = \mathbf{FPF}' + \mathbf{U}$$

The mean vector $\boldsymbol{\mu}$ $(p \times p)$ of manifest variables is structured as

$$\boldsymbol{\mu} = \mathbf{a} + \mathbf{Fv}$$

Exploratory Factor Analysis Models

Traditionally, exploratory factor analysis is applied when the relationships of manifest variables with factors have not been well-established in research. All manifest variables are allowed to have nonzero loadings on the factors in the model. First, factors are extracted and an initial solution is obtained. Then, for ease of interpretation a final factor solution is usually derived by rotating the factor space. Factor-variable relationships are determined by interpreting the final factor solution. This is different from the confirmatory factor analysis in which the factor-variable relationships are prescribed and to be confirmed.

So far, confirmatory and exploratory models are not distinguished in deriving the covariance and mean structures. These two types of models are now distinguished in terms of the required structures or restrictions in model matrices.

In PROC CALIS, the initial exploratory factor solution is obtained from a specific confirmatory factor model with restricted model matrices, which are described as follows:

- The factor loading matrix $\mathbf{F}$ has $n \times (n-1)/2$ fixed zeros at the upper triangle portion of the matrix.
- The factor covariance matrix $\mathbf{P}$ is an identity matrix, which means that factors are not correlated.
- The error covariance matrix $\mathbf{U}$ is a diagonal matrix.
- Except for METHOD=FIML or METHOD=LSFIML and robust methods with the ROBUST option, the mean structures are not modeled by default—that is, the intercept vector $\mathbf{a}$ or the factor mean vector $\mathbf{v}$ are not parameterized in the model.
- With METHOD=FIML or METHOD=LSFIML, the application of the robust methods with the ROBUST option, the use of the MEAN statement, or the specification of the MEANSTR option, the mean structures are modeled. The intercept vector $\mathbf{a}$ contains p free parameters, and the factor mean vector $\mathbf{v}$ is a zero vector.

The intercept vector $\mathbf{a}$ is parameterized in the FIML method because the first-order moments (that is, the variable means) of the data have to be analyzed with the FIML treatment of the incomplete observations. For robust methods, the observations are reweighted during estimation, so the intercept vector $\mathbf{a}$ must also be parameterized to accommodate the use of robust means in computing robust covariances. Estimation that is done using methods other than FIML or without the robust methods usually ignores the analysis of the mean structures because they are saturated and do not affect the fitting of covariance structures.

With the exploratory factor specification, you do not need to specify the patterns of the model matrices. PROC CALIS automatically sets up the correct patterns for the model matrices. For example, for an analysis with nine variables and three factors, the relevant model matrices of an exploratory FACTOR model have the following patterns, where * denotes free parameters in the model matrices:

$$\mathbf{F} = \begin{pmatrix} * & 0 & 0 \\ * & * & 0 \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

$$\mathbf{P} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and

$$\mathbf{U} = \begin{pmatrix} * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & * & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & * & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & * & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & * & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \end{pmatrix}$$

If METHOD=FIML or METHOD=LSFIML, the elements of the intercept vector **a** are all free parameters, as shown in the following:

$$\mathbf{a} = \begin{pmatrix} * \\ * \\ * \\ * \\ * \\ * \\ * \\ * \\ * \end{pmatrix}$$

The factor mean vector **v** is a fixed zero vector.

If an initial factor solution is rotated afterward, some of these matrix patterns are changed. In general, rotating a factor solution eliminates the fixed zero pattern in the upper triangle of the factor loading matrix **F**. If you apply an orthogonal rotation, the factor covariance matrix **P** does not change. It is an identity matrix before and after rotation. However, if you apply an oblique rotation, in general the rotated factor covariance matrix **P** is not an identity matrix and the off-diagonal elements are not zeros.

The error covariance matrix **U** remains unchanged after rotation. That is, it would still be a diagonal matrix. For the FIML estimation, the rotation does not affect the estimation of the intercept vector **a** and the fixed factor mean vector **v**.

Confirmatory Factor Analysis Models

In confirmatory FACTOR models, there are no imposed patterns on the **F**, **P**, **a**, and **v** model matrices. All elements in these model matrices can be specified. However, for model identification, you might need to specify some factor loadings or factor variances as constants.

The only model restriction in confirmatory FACTOR models is placed on **U**, which must be a diagonal matrix, as in exploratory FACTOR models too.

For example, for a confirmatory factor analysis with nine variables and three factors, you might specify the following patterns for the model matrices, where * denotes free parameters in the model matrices:

$$\mathbf{F} = \begin{pmatrix} 1 & 0 & 0 \\ * & 0 & 0 \\ * & 0 & 0 \\ 0 & 1 & 0 \\ 0 & * & 0 \\ 0 & * & 0 \\ 0 & 0 & 1 \\ 0 & 0 & * \\ 0 & 0 & * \end{pmatrix}$$

$$\mathbf{P} = \begin{pmatrix} * & * & * \\ * & * & * \\ * & * & * \end{pmatrix}$$

and

$$\mathbf{U} = \begin{pmatrix} * & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & * & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & * & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & * & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & * & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \end{pmatrix}$$

In this confirmatory factor model, mean structures are not modeled. In addition, there are some distinctive features that underscore the differences between confirmatory and exploratory models:

- Factor loading matrix **F** contains mostly zero elements and few nonzero free parameters, a pattern which is seen in most confirmatory factor models. In contrast, in exploratory factor models most elements in the **F** matrix are nonzero parameters.
- Factor loading matrix **F** contains fixed values of ones. These fixed values are used for model identification purposes (that is, identifying the scales of the latent variables). In general, you always have to make sure that your confirmatory factor models are identified by putting fixed values in appropriate parameter locations in the model matrices. However, this is not a concern in exploratory FACTOR models because identification has been ensured by imposing certain patterns on the model matrices.
- The nonzero off-diagonal parameters in the factor covariance matrix **P** indicate that correlated factors are hypothesized in the confirmatory factor model. This cannot be the case with the initial model of exploratory FACTOR models, where the **P** matrix must be an identity matrix before rotation.

Summary of Matrices in the FACTOR Model

Let p be the number of manifest variables and n be the number of factors in the FACTOR model. The names, roles, and dimensions of the FACTOR model matrices are shown in the following table.

Matrix	Name	Description	Dimensions
F	_FACTLOAD_	Factor loading matrix	$p \times n$
P	_FACTFCOV_	Factor covariance matrix	$n \times n$
U	_FACTERRV_	Error covariance matrix	$p \times p$
a	_FACTINTE_	Intercepts	$p \times 1$
v	_FACTMEAN_	Factor means	$n \times 1$

Specification of the Exploratory Factor Model

Because all initial model matrices of exploratory FACTOR models are predefined in PROC CALIS, you do not need to specify any other parameters in the model matrices. To obtain desired factor solutions, you can use various options for exploratory factor analysis in the FACTOR statement. These options are the *EFA-options* in the FACTOR statement. Two main types of *EFA-options* are shown as follows:

- options for factor extraction: COMPONENT, HEYWOOD, and N=.
- options for factor rotation: GAMMA=, NORM=, RCONVERGE=, RITER=, ROTATE=, and TAU=.

For example, the following statement requests that three factors be extracted, followed by a varimax rotation of the initial factor solution:

```
factor n=3 rotate=varimax;
```

See the FACTOR statement on page 1600 for details about the *EFA-options*.

Specification of the Confirmatory Factor Model

To specify a confirmatory FACTOR model, you specify the factor-variable relationships in the FACTOR statement, the factor variances and error variances in the PVAR statement, the factor covariances in the COV statement, and the means and intercepts in the MEAN statement.

Specification of Factor-Variable Relationships

The *CFA-spec* in the FACTOR statement is for specifying the factor-variables relationships. For example, in the following statement you specify three factors F1, F2, and F3 that are related to different clusters of observed variables V1–V9:

```
factor
   F1  ===> V1-V3  = 1. parm1 (.4) parm2 (.4),
   F2  ===> V4-V6  = 1. parm3 parm4,
   F3  ===> V7-V9  = 1. parm5 parm6 (.3);
```

In the specification, variable V1 has a fixed loading of 1.0 on F1. Variables V2 and V3 have loadings on F1 also. These two loadings are free parameters named parm1 and parm2, respectively. Initial estimates can be set in parentheses after the free parameters. For example, both parm1 and parm2 have initial values at 0.4. Similarly, relationships of factor F2 with V4–V6 and of factor F3 with V7–V9 are defined in the

same FACTOR statement. Providing initial estimates for parameters is optional. In this example, parm3, parm4, and parm5 are all free parameters without initial values provided. PROC CALIS can determine appropriate initial estimates for these parameters. See the descriptions of *CFA-spec* in the FACTOR statement on page 1600 for more details about the syntax.

Specification of Factor Variances and Error Variances

You can specify the factor variances and error variances in the PVAR statement. For example, consider the following statement:

```
pvar F1-F3  = fvar1-fvar3,
     V1-V9  = evar1-evar9 (9*10.);
```

In the PVAR statement, you specify the variances of factors F1, F2, and F3 as free parameters fvar1, fvar2, and fvar3, respectively, and the error variances for manifest variables V1–V9 as free parameters evar1–evar9, respectively. Each of the error variance parameters is given a starting value at 10. See the PVAR statement on page 1690 for more details about the syntax.

Specification of Factor Covariances

You can specify the factor covariances in the COV statement. For example, you specify the covariances among factors F1, F2, and F3 in the following statement:

```
cov F1 F2  = cov12,
    F1 F3  = cov13,
    F2 F3  = cov23;
```

The covariance parameters are named cov12, cov13, and cov23, respectively. They represent the lower triangular elements of the factor covariance matrix $\mathbf{P}$. See the COV statement on page 1593 for more details about the syntax.

Specification of Means and Intercepts

If mean structures are of interest, you can also specify the factor means and the intercepts for the manifest variables in the MEAN statement. For example, consider the following statement:

```
mean F1-F3 = fmean1-fmean3,
     V1-V9 = 9*12.;
```

In this statement, you specify the factor means of F1, F2, and F3 as free parameters fmean1, fmean2, and fmean3, respectively, and the intercepts for variables V1–V9 as fixed parameters at 12. See the MEAN statement on page 1651 for more details about the syntax.

Naming the Factors

For the exploratory FACTOR model, PROC CALIS generates the names for the factors automatically. For the confirmatory FACTOR model, you can specify the names for the factors. Unlike the LINEQS model, in the confirmatory FACTOR model you do not need to use the ‘F’ or ‘f’ prefix to denote factors in the model. You can use any valid SAS variable names for the factors, especially those names that reflect the nature of the factors. To avoid confusions with other names in the model, some general rules are recommended. See the section “Naming Variables and Parameters” on page 1777 for these general rules about naming variables and parameters.

Default Parameters in the FACTOR Model

Default parameters in the FACTOR model are different for exploratory and confirmatory factor models.

For the exploratory FACTOR model, all fixed and free parameters of the model are prescribed. These prescribed parameters include a fixed pattern for the factor loading matrix **F**, a diagonal pattern for the error variance matrix **U**, and an identity matrix for factor covariance matrix **P**. This means that factors are uncorrelated in the estimation. However, if you specify an oblique rotation after the estimation of the factor solution, the factors could become correlated. See the section "Exploratory Factor Analysis Models" on page 1740 for more details about the patterns of the exploratory FACTOR model. Because all these patterns are prescribed, you cannot override any of these parameters for the exploratory FACTOR model.

For the confirmatory FACTOR model, the set of default free parameters of the confirmatory FACTOR model includes the following:

- the error variances of the observed variables; these correspond to the diagonal elements of the uniqueness matrix **U**
- the variances and covariances among the factors; these correspond to all elements of the factor covariance matrix **P**
- the intercepts of the observed variables if the mean structures are modeled; these correspond to all elements of the intercept vector **a**

PROC CALIS names the default free parameters with the _Add prefix (or the _a prefix for the case of default intercept parameters in **a** for exploratory factor models), followed by a unique integers for each parameter. Except for the exploratory factor model, you can override the default free parameters by explicitly specifying them as free, constrained, or fixed parameters in the COV, MEAN, or PVAR statement.

In addition to default free parameters, another type of default parameter is the fixed zeros applied to the unspecified parameters in the loading matrix **F** and the factor means in the $\boldsymbol{\nu}$ vector. Certainly, you use the FACTOR and MEAN specifications to override those default zero loadings or factor means and set them to free, constrained, or fixed parameters. Notice that the uniqueness matrix **U** in the confirmatory factor model is a diagonal element. You cannot specify any of its off-diagonal elements—they are always fixed zeros by the model restriction.

The LINEQS Model

The LINEQS modeling language is adapted from the EQS (equations) program by Bentler (1995). The statistical models that LINEQS or EQS analyzes are essentially the same as other general modeling languages such as LISMOD, RAM, and PATH. However, the terminology and approach of the LINEQS or EQS modeling language are different from other languages. They are based on the theoretical model developed by Bentler and Weeks (1980). For convenience, models that are analyzed using the LINEQS modeling language are called LINEQS models. Note that these so-called LINEQS models can also be analyzed by other general modeling languages in PROC CALIS.

In the LINEQS (or the original EQS) model, relationships among variables are represented by a system of equations. For example:

$$Y_1 = a_0 + a_1 X_1 + a_2 X_2 + E_1$$

$$Y_2 = b_0 + b_1 X_1 + b_2 Y_1 + E_2$$

On the left-hand side of each equation, an outcome variable is hypothesized to be a linear function of one or more predictor variables and an error, which are all specified on the right-hand side of the equation. The parameters specified in an equation are the effects (or regression coefficients) of the predictor variables. For example, in the preceding equations, Y_1 and Y_2 are outcome variables; E_1 and E_2 are error variables; a_1, a_2, b_1, and b_2 are effect parameters (or regression coefficients); and a_0 and b_0 are intercept parameters. Variables X_1 and X_2 serve as predictors in the first equation, while variables X_1 and Y_1 serve as predictors in the second equation.

This is almost the same representation as in multiple regression models. However, the LINEQS model entails more. It supports a system of equations that can also include latent variables, measurement errors, and correlated errors.

Types of Variables in the LINEQS Model

The distinction between dependent and independent variables is important in the LINEQS model.

A variable is dependent if it appears on the left-hand side of an equation in the model. A dependent variable might be observed (manifest) or latent. It might or might not appear on the right-hand side of other equations, but it cannot appear on the left-hand sides of two or more equations. Error variables cannot be dependent in the LINEQS model.

A variable in the LINEQS model is independent if it is not dependent. Independent variables can be observed (manifest) or latent. All error variables must be independent in the LINEQS model.

Dependent variables are also referred to as endogenous variables; these names are interchangeable. Similarly, independent variables are interchangeable with exogenous variables.

Whereas an outcome variable in any equation must be a dependent variable, a predictor variable in an equation is not necessarily an independent variable in the entire LINEQS model. For example, Y_1 is a predictor variable in the second equation of the preceding example, but it is a dependent variable in the LINEQS model. In summary, the predictor-outcome nature of a variable is determined within a single equation, while the exogenous-endogenous (independent-dependent) nature of variable is determined within the entire system of equations.

In addition to the dependent-independent variable distinction, variables in the LINEQS model are distinguished according to whether they are observed in the data. Variables that are observed in research are called observed or manifest variables. Hypothetical variables that are not observed in the LINEQS model are latent variables.

Two types of latent variables should be distinguished: one is error variables; the other is non-error variables. An error variable is unique to an equation. It serves as the unsystematic source of effect for the outcome variable in an equation. If the outcome variable in the equation is latent, the corresponding error variable is also called disturbance. In contrast, non-error or systematic latent variables are called factors. Factors are unmeasured hypothetical constructs in your model. They are systematic sources that explain or describe functional relationships in your model.

Both manifest variables and latent factors can be dependent or independent. However, error or disturbance terms must be independent (or exogenous) variables in your model.

Naming Variables in the LINEQS Model

Whether a variable in each equation is an outcome or a predictor variable is prescribed by the modeler. Whether a variable is independent or dependent can be determined by analyzing the entire system of equations in the model. Whether a variable is observed or latent can be determined if it is referenced in your data set. However, whether a latent variable serves as a factor or an error can be determined only if you provide the specific information.

To distinguish latent factors from errors and both from manifest variables, the following rules for naming variables in the LINEQS model are followed:

- Manifest variables are referenced in the input data set. You use their names in the LINEQS model specification directly. There is no additional naming rule for the manifest variables in the LINEQS model beyond those required by the SAS System.
- Latent factor variables must start with letter F or f (for factor).
- Error variables must start with letter E or e (for error), or D or d (for disturbance). Although you might enforce the use of D- (or d-) variables for disturbances, it is not required. For flexibility, disturbance variables can also start with letter E or e in the LINEQS model.
- The names of latent variables, errors, and disturbances (F-, E-, and D-variables) should not coincide with the names of manifest variables.
- You should not use Intercept as a name for any variable. This name is reserved for the intercept specification in LINEQS model equations.

See the section "Naming Variables and Parameters" on page 1777 for the general rules about naming variables and parameters.

Matrix Representation of the LINEQS Model

As a programming language, the LINEQS model uses equations to describes relationships among variables. But as a mathematical model, the LINEQS model is more conveniently described by matrix terms. In this section, the LINEQS matrix model is described.

Suppose in a LINEQS model that there are n_i independent variables and n_d dependent variables. The vector of the independent variables is denoted by $\boldsymbol{\xi}$, in the order of manifest variables, latent factors, and error variables. The vector of dependent variables is denoted by $\boldsymbol{\eta}$, in the order of manifest variables and latent factors. The LINEQS model matrices are defined as follows:

$\boldsymbol{\alpha}$ $(n_d \times 1)$:	intercepts of dependent variables
$\boldsymbol{\beta}$ $(n_d \times n_d)$:	effects of dependent variables (in columns) on dependent variables (in rows)
$\boldsymbol{\gamma}$ $(n_d \times n_i)$:	effects of independent variables (in columns) on dependent variables (in rows)
$\boldsymbol{\Phi}$ $(n_i \times n_i)$:	covariance matrix of independent variables
$\boldsymbol{\nu}$ $(n_i \times 1)$:	means of independent variables

The model equation of the LINEQS model is

$$\boldsymbol{\eta} = \boldsymbol{\alpha} + \boldsymbol{\beta}\boldsymbol{\eta} + \boldsymbol{\gamma}\boldsymbol{\xi}$$

Assuming that $(\mathbf{I}-\boldsymbol{\beta})$ is invertible, under the model the covariance matrix of all variables $(\boldsymbol{\eta}', \boldsymbol{\xi}')'$ is structured as

$$\boldsymbol{\Sigma}_a = \begin{pmatrix} (\mathbf{I}-\boldsymbol{\beta})^{-1}\boldsymbol{\gamma}\boldsymbol{\Phi}\boldsymbol{\gamma}'(\mathbf{I}-\boldsymbol{\beta})^{-1\prime} & (\mathbf{I}-\boldsymbol{\beta})^{-1}\boldsymbol{\gamma}\boldsymbol{\Phi} \\ \boldsymbol{\Phi}\boldsymbol{\gamma}'(\mathbf{I}-\boldsymbol{\beta})^{-1\prime} & \boldsymbol{\Phi} \end{pmatrix}$$

The mean vector of all variables $(\boldsymbol{\eta}', \boldsymbol{\xi}')'$ is structured as

$$\boldsymbol{\mu}_a = \begin{pmatrix} (\mathbf{I}-\boldsymbol{\beta})^{-1}(\boldsymbol{\alpha} + \boldsymbol{\gamma}\boldsymbol{\nu}) \\ \boldsymbol{\nu} \end{pmatrix}$$

As is shown in the structured covariance and mean matrices, the means $\mathbf{G}$ and covariances of independent variables are direct model parameters in $\boldsymbol{\nu}$ and $\boldsymbol{\Phi}$; whereas the means and covariances of dependent variables are functions of various model matrices and hence functions of model parameters.

The covariance and mean structures of all observed variables are obtained by selecting the elements in $\boldsymbol{\Sigma}_a$ and $\boldsymbol{\mu}_a$. Mathematically, define a selection matrix $\mathbf{G}$ of dimensions $n \times (n_d + n_i)$, where n is the number of observed variables in the model. The selection matrix $\mathbf{G}$ contains zeros and ones as its elements. Each row of $\mathbf{G}$ has exactly one nonzero element at the position that corresponds to the location of an observed row variable in $\boldsymbol{\Sigma}_a$ or $\boldsymbol{\mu}_a$. With each row of $\mathbf{G}$ selecting a distinct observed variable, the structured covariance matrix of all observed variables is represented by

$$\boldsymbol{\Sigma} = \mathbf{G}\boldsymbol{\Sigma}_a\mathbf{G}'$$

The structured mean vector of all observed variables is represented by

$$\boldsymbol{\mu} = \mathbf{G}\boldsymbol{\mu}_a$$

Partitions of Some LINEQS Model Matrices and Their Restrictions

There are some restrictions in some of the LINEQS model matrices. Although these restrictions do not affect the derivation of the covariance and mean structures, they are enforced in the LINEQS model specification.

Model Restrictions on the β Matrix

The diagonal of the $\boldsymbol{\beta}$ matrix must be zeros. This prevents the direct regression of dependent variables on themselves. Hence, in the LINEQS statement you cannot specify the same variable on both the left-hand and the right-hand sides of the same equation.

Partitions of the γ Matrix and the Associated Model Restrictions

The columns of the $\boldsymbol{\gamma}$ matrix refer to the variables in $\boldsymbol{\xi}$, in the order of manifest variables, latent factors, and error variables. In the LINEQS model, the following partition of the $\boldsymbol{\gamma}$ matrix is assumed:

$$\boldsymbol{\gamma} = \begin{pmatrix} \boldsymbol{\gamma}_0 & \mathbf{E} \end{pmatrix}$$

where $\boldsymbol{\gamma}_0$ is an $n_d \times (n_i - n_d)$ matrix for the effects of independent manifest variables and latent factors on the dependent variables and $\mathbf{E}$ is an $n_d \times n_d$ permutation matrix for the effects of errors on the dependent variables.

The dimension of submatrix $\mathbf{E}$ is $n_d \times n_d$ because in the LINEQS model each dependent variable signifies an equation with an error term. In addition, because $\mathbf{E}$ is a permutation matrix (which is formed by exchanging rows of an identity matrix of the same order), the partition of the $\boldsymbol{\gamma}$ matrix ensures that each dependent variable is associated with a *unique* error term and that the effect of each error term on its associated dependent variable is 1.

As a result of the error term restriction, in the LINEQS statement you must specify a unique error term in each equation. The coefficient associated with the error term can only be a fixed value at one, either explicitly (with 1.0 inserted immediately before the error term) or implicitly (with no coefficient specified).

Partitions of the ν Vector and the Associated Model Restrictions

The $\boldsymbol{\nu}$ vector contains the means of independent variables, in the order of the manifest, latent factor, and error variables. In the LINEQS model, the following partition of the $\boldsymbol{\nu}$ vector is assumed:

$$\boldsymbol{\nu} = \begin{pmatrix} \boldsymbol{\nu}_0 \\ 0 \end{pmatrix}$$

where $\boldsymbol{\nu}_0$ is an $(n_i - n_d) \times 1$ vector for the means of independent manifest variables and latent factors and 0 is a null vector of dimension n_d for the means of errors or disturbances. Again, the dimension of the null vector is n_d because each dependent variable is associated uniquely with an error term. This partition restricts the means of errors or disturbances to zeros.

Hence, when specifying a LINEQS model, you cannot specify the means of errors (or disturbances) as free parameter or fixed values other than zero in the MEAN statement.

Partitions of the Φ Matrix

The $\boldsymbol{\Phi}$ matrix is for the covariances of the independent variables, in the order of the manifest, latent factor, and error variables. The following partition of the $\boldsymbol{\Phi}$ matrix is assumed:

$$\boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\Phi}_{11} & \boldsymbol{\Phi}'_{21} \\ \boldsymbol{\Phi}_{21} & \boldsymbol{\Phi}_{22} \end{pmatrix}$$

where $\boldsymbol{\Phi}_{11}$ is an $(n_i - n_d) \times (n_i - n_d)$ covariance matrix for the independent manifest variables and latent factors, $\boldsymbol{\Phi}_{22}$ is an $n_d \times n_d$ covariance matrix for the errors, and $\boldsymbol{\Phi}_{21}$ is an $n_d \times (n_i - n_d)$ covariance matrix for the errors with other independent variables in the LINEQS model. Because $\boldsymbol{\Phi}$ is symmetric, $\boldsymbol{\Phi}_{11}$ and $\boldsymbol{\Phi}_{22}$ are also symmetric.

There are actually no model restrictions placed on the submatrices of the partition. However, in most statistical applications, errors represent unsystematic sources of effects and therefore they are not to be correlated with other systematic sources. This implies that submatrix $\boldsymbol{\Phi}_{21}$ is a null matrix. However, $\boldsymbol{\Phi}_{21}$ being null is not enforced in the LINEQS model specification. If you ever specify a covariance between an error variable and a non-error independent variable in the COV statement, as a workaround trick or otherwise, you should provide your own theoretical justifications.

Summary of Matrices and Submatrices in the LINEQS Model

Let n_d be the number of dependent variables and n_i be the number of independent variables. The names, roles, and dimensions of the LINEQS model matrices and submatrices are summarized in the following table.

Matrix	Name	Description	Dimensions
Model Matrices			
$\boldsymbol{\alpha}$	_EQSALPHA_	Intercepts of dependent variables	$n_d \times 1$
$\boldsymbol{\beta}$	_EQSBETA_	Effects of dependent (column) variables on dependent (row) variables	$n_d \times n_d$
$\boldsymbol{\gamma}$	_EQSGAMMA_	Effects of independent (column) variables on dependent (row) variables	$n_d \times n_i$
$\boldsymbol{\nu}$	_EQSNU_	Means of independent variables	$n_i \times 1$
$\boldsymbol{\Phi}$	_EQSPHI_	Covariance matrix of independent variables	$n_i \times n_i$
Submatrices			
$\boldsymbol{\gamma}_0$	_EQSGAMMA_SUB_	Effects of independent variables, excluding errors, on dependent variables	$n_d \times (n_i - n_d)$
$\boldsymbol{\nu}_0$	_EQSNU_SUB_	Means of independent variables, excluding errors	$(n_i - n_d) \times 1$
$\boldsymbol{\Phi}_{11}$	_EQSPHI11_	Covariance matrix of independent variables, excluding errors	$(n_i - n_d) \times (n_i - n_d)$
$\boldsymbol{\Phi}_{21}$	_EQSPHI21_	Covariances of errors with other independent variables	$n_d \times (n_i - n_d)$
$\boldsymbol{\Phi}_{22}$	_EQSPHI22_	Covariance matrix of errors	$n_d \times n_d$

Specification of the LINEQS Model

Specification in Equations

In the LINEQS statement, you specify intercepts and effect parameters (or regression coefficients) along with the variable relationships in equations. In terms of model matrices, you specify the $\boldsymbol{\alpha}$ vector and the $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ matrices in the LINEQS statement without using any matrix language.

For example:

$$Y = b_0 + b_1 * X_1 + b_2 * F_2 + E_1$$

In this equation, you specify Y as an outcome variable, X_1 and F_2 as predictor variables, and E_1 as an error variable. The parameters in the equation are the intercept b_0 and the path coefficients (or effects) b_1 and b_2.

This kind of model equation is specified in the LINEQS statement. For example, the previous equation translates into the following LINEQS statement specification:

```
lineqs Y = b0 * Intercept + b1 * X1 + b2 * F2 + E1;
```

If the mean structures of the model are not of interest, the intercept term can be omitted. The specification becomes:

```
lineqs Y =  b1 * X1 + b2 * F2 + E1;
```

See the LINEQS statement on page 1618 for the details about the syntax.

Because of the LINEQS model restrictions (see the section "Partitions of Some LINEQS Model Matrices and Their Restrictions" on page 1748), you must also follow these rules when specifying LINEQS model equations:

- A dependent variable can appear only on the left-hand side of an equation once. In other words, you must put all predictor variables for a dependent variable in one equation. This is different from some econometric models where a dependent variable can appear on the left-hand sides of two equations to represent an equilibrium point. However, this limitation can be resolved by reparameterization in some cases. See Example 33.18.
- A dependent variable that appears on the left-hand side of an equation cannot appear on the right-hand side of the same equation. If you measure the same characteristic at different time points and the previous measurement serves as a predictor of the next measurement, you should use different variable names for the measurements so as to comply with this rule.
- An error term must be specified in each equation and must be unique. The same error name cannot appear in two or more equations. When an equation is truly intended to have no error term, it should be represented equivalently in the LINEQS equation by introducing an error term with zero variance (specified in the VARIANCE statement).
- The regression coefficient (effect) that is associated with an error term must be fixed at one (1.0). This is done automatically by omitting any fixed constants or parameters that are associated with the error terms. Inserting a parameter or a fixed value other than 1 immediately before an error term is not allowed.

Mean, Variance, and Covariance Parameter Specification

In addition to the intercept and effect parameters that are specified in equations, the means, variances, and covariances among all independent variables are parameters in the LINEQS model. An exception is that the means of all error variables are restricted to fixed zeros in the LINEQS model. To specify the mean, variance, and covariance parameters, you use the MEAN, VARIANCE, and the COV statements, respectively.

The means, variances, and covariances among dependent variables are not parameters themselves in the model. Rather, they are complex functions of the model parameters. See the section "Matrix Representation of the LINEQS Model" on page 1747 for mathematical details.

Default Parameters in the LINEQS Model

There are two types of default parameters of the LINEQS model, as implemented in PROC CALIS. One is the free parameters; the other is the fixed constants.

The following sets of parameters are free parameters by default:

- the variances of all exogenous (independent) observed or latent variables (*including* error and disturbance variables)
- the covariances among all exogenous (independent) manifest or latent variables (*excluding* error and disturbance variances)
- the means of all exogenous (independent) observed variables if the mean structures are modeled
- the intercepts of all endogenous (dependent) manifest variables if the mean structures are modeled

PROC CALIS names the default free parameters with the _Add prefix and a unique integer suffix. You can override the default free parameters by explicitly specifying them as free, constrained, or fixed parameters in the COV, LINEQS, MEAN, or VARIANCE statement.

Parameters that are not default free parameters in the LINEQS model are fixed constants by default. You can override almost all of the default fixed constants of the LINEQS model by using the COV, LINEQS, MEAN, or VARIANCE statement. You cannot override the following two sets of fixed constants:

- fixed zero parameters for the direct effects (path coefficients) of variables on their own. You cannot have an equation in the LINEQS statement that has the same variable specified on the left-hand and the right-hand sides.
- fixed one effects from the error or disturbance variables. You cannot set the path coefficient (effect) of the error or disturbance term to any value other than 1 in the LINEQS statement.

These two sets of fixed parameters reflect the LINEQS model restrictions so that they cannot be modified. Other than these two sets of default fixed parameters, all other default fixed parameters are zeros. You can override these default zeros by explicitly specifying them as free, constrained, or fixed parameters in the COV, LINEQS, MEAN, or VARIANCE statement.

The LISMOD Model and Submodels

As a statistical model, the LISMOD modeling language is derived from the LISREL model proposed by Jöreskog and others (see Keesling 1972; Wiley 1973; Jöreskog 1973). But as a computer language, the LISMOD modeling language is quite different from the LISREL program. To maintain the consistence of specification syntax within the CALIS procedure, the LISMOD modeling language departs from the original LISREL programming language. In addition, to make the programming a little easier, some terminological changes from LISREL are made in LISMOD.

For brevity, models specified by the LISMOD modeling language are called LISMOD models, although you can also specify these LISMOD models by other general modeling languages that are supported in PROC CALIS.

The following descriptions of LISMOD models are basically the same as those of the original LISREL models. The main modifications are the names for the model matrices.

The LISMOD model is described by three component models. The first one is the structural equation model that describes the relationships among latent constructs or factors. The other two are measurement models that relate latent factors to manifest variables.

Structural Equation Model

The structural equation model for latent factors is

$$\boldsymbol{\eta} = \boldsymbol{\alpha} + \boldsymbol{\beta}\boldsymbol{\eta} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}$$

where:
$\boldsymbol{\eta}$ is a random vector of n_{η} endogenous latent factors
$\boldsymbol{\xi}$ is a random vector of $n_{\boldsymbol{\xi}}$ exogenous latent factors
$\boldsymbol{\zeta}$ is a random vector of errors
$\boldsymbol{\alpha}$ is a vector of intercepts
$\boldsymbol{\beta}$ is a matrix of regression coefficients of $\boldsymbol{\eta}$ variables on other $\boldsymbol{\eta}$ variables
$\boldsymbol{\Gamma}$ is a matrix of regression coefficients of $\boldsymbol{\eta}$ on $\boldsymbol{\xi}$

There are some assumptions in the structural equation model. To prevent a random variable in $\boldsymbol{\eta}$ from regressing directly on itself, the diagonal elements of $\boldsymbol{\beta}$ are assumed to be zeros. Also, $(I - \boldsymbol{\beta})^{-1}$ is assumed to be nonsingular, and $\boldsymbol{\zeta}$ is uncorrelated with $\boldsymbol{\xi}$.

The covariance matrix of $\boldsymbol{\zeta}$ is denoted by $\boldsymbol{\Psi}$ and its expected value is a null vector. The covariance matrix of $\boldsymbol{\xi}$ is denoted by $\boldsymbol{\Phi}$ and its expected value is denoted by $\boldsymbol{\kappa}$.

Because variables in the structural equation model are not observed, to analyze the model these latent variables must somehow relate to the manifest variables. The measurement models, which are discussed in the subsequent sections, provide such relations.

Measurement Model for y

$$\mathbf{y} = \boldsymbol{\nu}_y + \boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon}$$

where:
$\mathbf{y}$ is a random vector of n_y manifest variables
$\boldsymbol{\epsilon}$ is a random vector of errors for $\mathbf{y}$
$\boldsymbol{\nu}_y$ is a vector of intercepts for $\mathbf{y}$
$\boldsymbol{\Lambda}_y$ is a matrix of regression coefficients of $\mathbf{y}$ on $\boldsymbol{\eta}$

It is assumed that $\boldsymbol{\epsilon}$ is uncorrelated with either $\boldsymbol{\eta}$ or $\boldsymbol{\zeta}$. The covariance matrix of $\boldsymbol{\epsilon}$ is denoted by $\boldsymbol{\Theta}_y$ and its expected value is the null vector.

Measurement Model for x

$$\mathbf{x} = \boldsymbol{\nu}_x + \boldsymbol{\Lambda}_x \boldsymbol{\xi} + \boldsymbol{\delta}$$

where:
$\mathbf{x}$ is a random vector of n_x manifest variables
$\boldsymbol{\delta}$ is a random vector of errors for $\mathbf{x}$
$\boldsymbol{\nu}_x$ is a vector of intercepts for $\mathbf{x}$
$\boldsymbol{\Lambda}_x$ is a matrix of regression coefficients of $\mathbf{x}$ on $\boldsymbol{\xi}$

It is assumed that $\boldsymbol{\delta}$ is uncorrelated with $\boldsymbol{\xi}$, $\boldsymbol{\epsilon}$, or $\boldsymbol{\zeta}$. The covariance matrix of $\boldsymbol{\delta}$ is denoted by $\boldsymbol{\Theta}_x$ and its expected value is a null vector.

Covariance and Mean Structures

Under the structural and measurement equations and the model assumptions, the covariance structures of the manifest variables $(\mathbf{y}', \mathbf{x}')'$ are expressed as

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Lambda}_y (I - \boldsymbol{\beta})^{-1} (\boldsymbol{\Gamma}\boldsymbol{\Phi}\boldsymbol{\Gamma}' + \boldsymbol{\Psi})(I - \boldsymbol{\beta}')^{-1} \boldsymbol{\Lambda}_y' + \boldsymbol{\Theta}_y & \boldsymbol{\Lambda}_y (I - \boldsymbol{\beta})^{-1} \boldsymbol{\Gamma}\boldsymbol{\Phi}\boldsymbol{\Lambda}_x' \\ \boldsymbol{\Lambda}_x \boldsymbol{\Phi}\boldsymbol{\Gamma}' (I - \boldsymbol{\beta}')^{-1} \boldsymbol{\Lambda}_y' & \boldsymbol{\Lambda}_x \boldsymbol{\Phi}\boldsymbol{\Lambda}_x' + \boldsymbol{\Theta}_x \end{pmatrix}$$

The mean structures of the manifest variables $(\mathbf{y}', \mathbf{x}')'$ are expressed as

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\nu}_y + \boldsymbol{\Lambda}_y (I - \boldsymbol{\beta})^{-1} (\boldsymbol{\alpha} + \boldsymbol{\Gamma}\boldsymbol{\kappa}) \\ \boldsymbol{\nu}_x + \boldsymbol{\Lambda}_x \boldsymbol{\kappa} \end{pmatrix}$$

Model Matrices in the LISMOD Model

The parameters of the LISMOD model are elements in the model matrices, which are summarized as follows.

Matrix	Name	Description	Dimensions	Row Variables	Column Variables
$\boldsymbol{\alpha}$	_ALPHA_	Intercepts for $\boldsymbol{\eta}$	$n_\eta \times 1$	$\boldsymbol{\eta}$ (ETAVAR=)	N/A
$\boldsymbol{\beta}$	_BETA_	Effects of $\boldsymbol{\eta}$ on $\boldsymbol{\eta}$	$n_\eta \times n_\eta$	$\boldsymbol{\eta}$ (ETAVAR=)	$\boldsymbol{\eta}$ (ETAVAR=)
$\boldsymbol{\Gamma}$	_GAMMA_	Effects of $\boldsymbol{\xi}$ on $\boldsymbol{\eta}$	$n_\eta \times n_\xi$	$\boldsymbol{\eta}$ (ETAVAR=)	$\boldsymbol{\xi}$ (XIVAR=)
$\boldsymbol{\Psi}$	_PSI_	Error covariance matrix for $\boldsymbol{\eta}$	$n_\eta \times n_\eta$	$\boldsymbol{\eta}$ (ETAVAR=)	$\boldsymbol{\eta}$ (ETAVAR=)
$\boldsymbol{\Phi}$	_PHI_	Covariance matrix for $\boldsymbol{\xi}$	$n_\xi \times n_\xi$	$\boldsymbol{\xi}$ (XIVAR=)	$\boldsymbol{\xi}$ (XIVAR=)
$\boldsymbol{\kappa}$	_KAPPA_	Mean vector for $\boldsymbol{\xi}$	$n_\xi \times 1$	$\boldsymbol{\xi}$ (XIVAR=)	N/A
$\boldsymbol{\nu}_y$	_NUY_	Intercepts for $\mathbf{y}$	$n_y \times 1$	$\mathbf{y}$ (YVAR=)	N/A
$\boldsymbol{\Lambda}_y$	_LAMBDAY_	Effects of $\boldsymbol{\eta}$ on $\mathbf{y}$	$n_y \times n_\eta$	$\mathbf{y}$ (YVAR=)	$\boldsymbol{\eta}$ (ETAVAR=)
$\boldsymbol{\Theta}_y$	_THETAY_	Error covariance matrix for $\mathbf{y}$	$n_y \times n_y$	$\mathbf{y}$ (YVAR=)	$\mathbf{y}$ (YVAR=)
$\boldsymbol{\nu}_x$	_NUX_	Intercepts for $\mathbf{x}$	$n_x \times 1$	$\mathbf{x}$ (XVAR=)	N/A
$\boldsymbol{\Lambda}_x$	_LAMBDAX_	Effects of $\boldsymbol{\xi}$ on $\mathbf{x}$	$n_x \times n_\xi$	$\mathbf{x}$ (XVAR=)	$\boldsymbol{\xi}$ (XIVAR=)
$\boldsymbol{\Theta}_x$	_THETAX_	Error covariance matrix for $\mathbf{x}$	$n_x \times n_x$	$\mathbf{x}$ (XVAR=)	$\mathbf{x}$ (XVAR=)

There are twelve model matrices in the LISMOD model. Not all of them are used in all situations. See the section "LISMOD Submodels" on page 1757 for details. In the preceding table, each model matrix is given a name in the column Name, followed by a brief description of the parameters in the matrix, the dimensions, and the row and column variables being referred to. In the second column of the table, the LISMOD matrix names are used in the MATRIX statements when specifying the LISMOD model. In the last two columns of the table, following the row or column variables is the variable list (for example, ETAVAR=, YVAR=, and so on) in parentheses. These lists are used in the LISMOD statement for defining variables.

Specification of the LISMOD Model

The LISMOD specification consists of two tasks. The first task is to define the variables in the model. The second task is to specify the parameters in the LISMOD model matrices.

Specifying Variables

The first task is accomplished in the LISMOD statement. In the LISMOD statement, you define the lists of variables of interest: YVAR=, XVAR=, ETAVAR=, and XIVAR= lists, respectively for the $\mathbf{y}$-variables, $\mathbf{x}$-variables, $\boldsymbol{\eta}$-variables, and the $\boldsymbol{\xi}$-variables. While you provide the names of variables in these lists, you also define implicitly the numbers of four types of variables: n_y, n_x, n_η, and n_ξ. The variables in the YVAR= and XVAR= lists are manifest variables and therefore must be present in the analyzed data set. The variables in the ETAVAR= and XIVAR= lists are latent factors, the names of which are assigned by the researcher to represent their roles in the substantive theory. After these lists are defined, the dimensions of the model

matrices are also defined by the number of variables on various lists. In addition, the variable orders in the lists are referred to by the row and column variables of the model matrices.

Unlike the LINEQS model, in the LISMOD model you do not need to use the 'F' or 'f' prefix to denote factors in the ETAVAR= or XIVAR= list. You can use any valid SAS variable names for the factors, especially those names that reflect the nature of the factors. To avoid confusion with other names in the model, some general rules are recommended. See the section "Naming Variables and Parameters" on page 1777 for these general rules about naming variables and parameters.

Specifying Parameters in Model Matrices

The second task is accomplished by the MATRIX statements. In each MATRIX statement, you specify the model matrix by using the matrix names described in the previous table. Then you specify the parameters (free or fixed) in the locations of the model matrix. You can use as many MATRIX statements as needed for defining your model. But each model matrix can be specified only in one MATRIX statement, and each MATRIX statement is used for specifying only one model matrix.

An Example

In the section "LISMOD Model" on page 1526, the LISMOD modeling language is used to specify the model described in the section "A Structural Equation Example" on page 1521. In the LISMOD statement, you define four lists of variables, as shown in the following statement:

```
lismod
   yvar  = Anomie67 Powerless67 Anomie71 Powerless71,
   xvar  = Education SEI,
   etav  = Alien67 Alien71,
   xivar = SES;
```

Endogenous latent factors are specified in the ETAVAR= list. Exogenous latent factors are specified in the XIVAR= list. In this case, Alien67 and Alien71 are the $\boldsymbol{\eta}$-variables, and SES is the only $\boldsymbol{\xi}$-variable in the model. Manifest variables that are indicators of endogenous latent factors in $\boldsymbol{\eta}$ are specified in the YVAR= list. In this case, they are the Anomie and Powerless variables, measured at two different time points. Manifest variables that are indicators of exogenous latent factors in $\boldsymbol{\xi}$ are specified in the XVAR= list. In this case, they are the Education and the SEI variables. Implicitly, the dimensions of the model matrices are defined by these lists already; that is, $n_y = 4$, $n_x = 2$, $n_\eta = 2$, and $n_\xi = 1$.

The MATRIX statements are used to specify parameters in the model matrices. For example, in the following statement you define the _LAMBDAX_ ($\boldsymbol{\Lambda}_x$) matrix with two nonzero entries:

```
matrix _LAMBDAX_ [1,1] = 1.0,
                 [2,1] = lambda;
```

The first parameter location is for `[1,1]`, which is the effect of SES (the first variable in the XIVAR= list) on Education (the first element in the XVAR= list). A fixed value of 1.0 is specified there. The second parameter location is for `[2,1]`, which is the effect of SES (the first variable in the XIVAR= list) on SEI (the second variable in the XVAR= list). A parameter named lambda without an initial value is specified there.

Another example is shown as follows:

```
matrix _THETAY_  [1,1] = theta1,
                 [2,2] = theta2,
                 [3,3] = theta1,
                 [4,4] = theta2,
```

```
[3,1] = theta5,
[4,2] = theta5;
```

In this MATRIX statement, the error variances and covariances (that is, the $\boldsymbol{\Theta}_y$ matrix) for the $\mathbf{y}$-variables are specified. The diagonal elements of the _THETAY_ matrix are specified by parameters theta1, theta2, theta1, and theta2, respectively, for the four $\mathbf{y}$-variables Anomie67, Powerless67, Anomie71, and Powerless71. By using the same parameter name theta1, the error variances for Anomie67 and Anomie71 are implicitly constrained. Similarly, the error variances for Powerless67 and Powerless71 are also implicitly constrained. Two more parameter locations are specified. The error covariance between Anomie67 and Anomie71 and the error covariance between Powerless67 and Powerless71 are both represented by the parameter theta5. Again, this is an implicit constraint on the covariances. All other unspecified elements in the _THETAY_ matrix are treated as fixed zeros.

In this example, no parameters are specified for matrices _ALPHA_, _KAPPA_, _NUY_, or _NUX_. Therefore, mean structures are not modeled.

LISMOD Submodels

It is not necessary to specify all four lists of variables in the LISMOD statement. When some lists are unspecified in the LISMOD statement, PROC CALIS analyzes submodels derived logically from the specified lists of variables. For example, if only $\mathbf{y}$- and $\mathbf{x}$-variable lists are specified, the submodel being analyzed would be a multivariate regression model with manifest variables only. Not all combinations of lists lead to meaningful submodels, however. To determine whether and how a submodel (which is formed by a certain combination of variable lists) can be analyzed, the following three principles in the LISMOD modeling language are applied:

- Submodels with at least one of the YVAR= and XVAR= lists are required.
- Submodels that have an ETAVAR= list but no YVAR= list cannot be analyzed.
- When a submodel has a YVAR= (or an XVAR=) list but without an ETAVAR= (or a XIVAR=) list, it is assumed that the set of $\mathbf{y}$-variables ($\mathbf{x}$-variables) is equivalent to the $\boldsymbol{\eta}$-variables ($\boldsymbol{\xi}$-variables). Hereafter, this principle is referred to as an equivalence interpretation.

Apparently, the third principle is the same as the situation where the latent factors $\boldsymbol{\eta}$ (or $\boldsymbol{\xi}$) are perfectly measured by the manifest variables $\mathbf{y}$ (or $\mathbf{x}$). That is, in such a perfect measurement model, $\boldsymbol{\Lambda}_y$ ($\boldsymbol{\Lambda}_x$) is an identity matrix and $\boldsymbol{\Theta}_y$ ($\boldsymbol{\Theta}_x$) and $\boldsymbol{\nu}_y$ ($\boldsymbol{\nu}_x$) are both null. This can be referred to as a perfect measurement interpretation. However, the equivalence interpretation stated in the last principle presumes that there are actually no measurement equations at all. This is important because under the equivalence interpretation, matrices $\boldsymbol{\Lambda}_y$ ($\boldsymbol{\Lambda}_x$), $\boldsymbol{\Theta}_y$ ($\boldsymbol{\Theta}_x$) and $\boldsymbol{\nu}_y$ ($\boldsymbol{\nu}_x$) are nonexistent rather than fixed quantities, which is assumed under the perfect measurement interpretation. Hence, the $\mathbf{x}$-variables are treated as *exogenous* variables with the equivalence interpretation, but they are still treated as *endogenous* with the perfect measurement interpretation. Ultimately, whether $\mathbf{x}$-variables are treated as exogenous or endogenous affects the default or automatic parameterization. See the section "Default Parameters" on page 1626 for more details.

By using these three principles, the models and submodels that PROC CALIS analyzes are summarized in the following table, followed by detailed descriptions of these models and submodels.

Presence of Lists		Description	Model Equations	Nonfixed Model Matrices
Presence of Both x- and y-variables				
1	YVAR=, ETAVAR=, XVAR=, XIVAR=	Full model	$\mathbf{y} = \boldsymbol{\nu}_y + \boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon}$ $\mathbf{x} = \boldsymbol{\nu}_x + \boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta}$ $\boldsymbol{\eta} = \boldsymbol{\alpha} + \boldsymbol{\beta}\boldsymbol{\eta} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}$	$\boldsymbol{\nu}_y,\boldsymbol{\Lambda}_y,\boldsymbol{\Theta}_y$ $\boldsymbol{\nu}_x,\boldsymbol{\Lambda}_x,\boldsymbol{\Theta}_x,\boldsymbol{\kappa},\boldsymbol{\Phi}$ $\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Gamma},\boldsymbol{\Psi}$
2	YVAR=, XVAR=, XIVAR=	Full model with $\mathbf{y} \equiv \boldsymbol{\eta}$	$\mathbf{x} = \boldsymbol{\nu}_x + \boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta}$ $\mathbf{y} = \boldsymbol{\alpha} + \boldsymbol{\beta}y + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}$	$\boldsymbol{\nu}_x,\boldsymbol{\Lambda}_x,\boldsymbol{\Theta}_x,\boldsymbol{\kappa},\boldsymbol{\Phi}$ $\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Gamma},\boldsymbol{\Psi}$
3	YVAR=, ETAVAR=, XVAR=	Full model with $\mathbf{x} \equiv \boldsymbol{\xi}$	$\mathbf{y} = \boldsymbol{\nu}_y + \boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon}$ $\boldsymbol{\eta} = \boldsymbol{\alpha} + \boldsymbol{\beta}\boldsymbol{\eta} + \boldsymbol{\Gamma}x + \boldsymbol{\zeta}$	$\boldsymbol{\nu}_y,\boldsymbol{\Lambda}_y,\boldsymbol{\Theta}_y$ $\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Gamma},\boldsymbol{\Psi},\boldsymbol{\kappa},\boldsymbol{\Phi}$
4	YVAR=, XVAR=	Regression $(\mathbf{y} \equiv \boldsymbol{\eta})$ $(\mathbf{x} \equiv \boldsymbol{\xi})$	$\mathbf{y} = \boldsymbol{\alpha} + \boldsymbol{\beta}y + \boldsymbol{\Gamma}x + \boldsymbol{\zeta}$, or $(\mathbf{I} - \boldsymbol{\beta})^{-1}y = \boldsymbol{\alpha} + \boldsymbol{\Gamma}x + \boldsymbol{\zeta}$	$\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Gamma},\boldsymbol{\Psi},\boldsymbol{\kappa},\boldsymbol{\Phi}$
Presence of x-variables and Absence of y-variables				
5	XVAR=, XIVAR=	Factor model for x	$\mathbf{x} = \boldsymbol{\nu}_x + \boldsymbol{\Lambda}_x\boldsymbol{\xi} + \boldsymbol{\delta}$	$\boldsymbol{\nu}_x,\boldsymbol{\Lambda}_x,\boldsymbol{\Theta}_x,\boldsymbol{\kappa},\boldsymbol{\Phi}$
6	XVAR=	x-structures $(\mathbf{x} \equiv \boldsymbol{\xi})$		$\boldsymbol{\kappa},\boldsymbol{\Phi}$
Presence of y-variables and Absence of x-variables				
7	YVAR=, ETAVAR=	Factor model for y	$\mathbf{y} = \boldsymbol{\nu}_y + \boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon}$ $\boldsymbol{\eta} = \boldsymbol{\alpha} + \boldsymbol{\beta}\boldsymbol{\eta} + \boldsymbol{\zeta}$	$\boldsymbol{\nu}_y,\boldsymbol{\Lambda}_y,\boldsymbol{\Theta}_y$ $\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Psi}$
8	YVAR=	y-structures $(\mathbf{y} \equiv \boldsymbol{\eta})$	$\mathbf{y} = \boldsymbol{\alpha} + \boldsymbol{\beta}y + \boldsymbol{\zeta}$, or $(\mathbf{I} - \boldsymbol{\beta})^{-1}y = \boldsymbol{\alpha} + \boldsymbol{\zeta}$	$\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Psi}$
9	YVAR=, ETAVAR=, XIVAR=	Second-order factor model	$\mathbf{y} = \boldsymbol{\nu}_y + \boldsymbol{\Lambda}_y\boldsymbol{\eta} + \boldsymbol{\epsilon}$ $\boldsymbol{\eta} = \boldsymbol{\alpha} + \boldsymbol{\beta}\boldsymbol{\eta} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}$	$\boldsymbol{\nu}_y,\boldsymbol{\Lambda}_y,\boldsymbol{\Theta}_y$ $\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Gamma},\boldsymbol{\Psi},\boldsymbol{\kappa},\boldsymbol{\Phi}$
10	YVAR=, XIVAR=	Factor model $(\mathbf{y} \equiv \boldsymbol{\eta})$	$\mathbf{y} = \boldsymbol{\alpha} + \boldsymbol{\beta}y + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}$, or $(\mathbf{I} - \boldsymbol{\beta})^{-1}y = \boldsymbol{\alpha} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\zeta}$	$\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\Gamma},\boldsymbol{\Psi},\boldsymbol{\kappa},\boldsymbol{\Phi}$

Models 1, 2, 3, and 4—Presence of Both x- and y-Variables

Submodels 1, 2, 3, and 4 are characterized by the presence of both **x**- and **y**-variables in the model. In fact, Model 1 is the full model with the presence of all four types of variables. All twelve model matrices are parameter matrices in this model.

Depending on the absence of the latent factor lists, manifest variables can replace the role of the latent factors in Models 2–4. For example, the absence of the ETAVAR= list in Model 2 means **y** is equivalent to $\boldsymbol{\eta}$ ($\mathbf{y} \equiv \boldsymbol{\eta}$). Consequently, you cannot, nor do you need to, use the MATRIX statement to specify parameters in the _LAMBDAY_, _THETAY_, or _NUY_ matrices under this model. Similarly, because **x** is equivalent to $\boldsymbol{\xi}$ ($\mathbf{x} \equiv \boldsymbol{\xi}$) in Model 3, you cannot, nor do you need to, use the MATRIX statement to specify the parameters in the _LAMBDAX_, _THETAX_, or _NUX_ matrices. In Model 4, **y** is equivalent to $\boldsymbol{\eta}$ ($\mathbf{y} \equiv \boldsymbol{\eta}$) and **x** is equivalent to $\boldsymbol{\xi}$ ($\mathbf{x} \equiv \boldsymbol{\xi}$). None of the six model matrices in the measurement equations are defined in the

model. Matrices in which you can specify parameters by using the MATRIX statement are listed in the last column of the table.

Describing Model 4 as a regression model is a simplification. Because $\mathbf{y}$ can regress on itself in the model equation, the regression description is not totally accurate for Model 4. Nonetheless, if $\boldsymbol{\beta}$ is a null matrix, the equation describes a multivariate regression model with outcome variables $\mathbf{y}$ and predictor variables $\mathbf{x}$. This model is the TYPE 2A model in LISREL VI (Jöreskog and Sörbom 1985).

You should also be aware of the changes in meaning of the model matrices when there is an equivalence between latent factors and manifest variables. For example, in Model 4 the $\boldsymbol{\Phi}$ and $\boldsymbol{\kappa}$ are now the covariance matrix and mean vector, respectively, of manifest variables $\mathbf{x}$, while in Model 1 (the complete model) these matrices are of the latent factors $\boldsymbol{\xi}$.

Models 5 and 6 — Presence of x*-Variables and Absence of* y*-Variables*

Models 5 and 6 are characterized by the presence of the $\mathbf{x}$-variables and the absence of $\mathbf{y}$-variables.

Model 5 is simply a factor model for measured variables $\mathbf{x}$, with $\boldsymbol{\Lambda}_x$ representing the factor loading matrix, $\boldsymbol{\Theta}_x$ the error covariance matrix, and $\boldsymbol{\Phi}$ the factor covariance matrix. If mean structures are modeled, $\boldsymbol{\kappa}$ represents the factor means and $\boldsymbol{\nu}_x$ is the intercept vector. This is the TYPE 1 submodel in LISREL VI (Jöreskog and Sörbom 1985).

Model 6 is a special case where there is no model equation. You specify the mean and covariance structures (in $\boldsymbol{\kappa}$ and $\boldsymbol{\Phi}$, respectively) for the manifest variables $\mathbf{x}$ directly. The $\mathbf{x}$-variables are treated as exogenous variables in this case. Because this submodel uses direct mean and covariance structures for measured variables, it can also be handled more easily by the MSTRUCT modeling language. See the MSTRUCT statement and the section "The MSTRUCT Model" on page 1761 for more details.

Note that because $\boldsymbol{\eta}$-variables cannot exist in the absence of $\mathbf{y}$-variables (see one of the three aforementioned principles for deriving submodels), adding the ETAVAR= list alone to these two submodels does not generate new submodels that can be analyzed by PROC CALIS.

Models 7, 8, 9 and 10—Presence of y*-Variables and Absence of* x*-Variables*

Models 7–10 are characterized by the presence of the $\mathbf{y}$-variables and the absence of $\mathbf{x}$-variables.

Model 7 is a factor model for $\mathbf{y}$-variables (TYPE 3B submodel in LISREL VI). It is similar to Model 5, but with regressions among latent factors allowed. When $\boldsymbol{\beta}$ is null, Model 7 functions the same as Model 5. It becomes a factor model for $\mathbf{y}$-variables, with $\boldsymbol{\Lambda}_y$ representing the factor loading matrix, $\boldsymbol{\Theta}_y$ the error covariance matrix, $\boldsymbol{\Psi}$ the factor covariance matrix, $\boldsymbol{\alpha}$ the factor means, and $\boldsymbol{\nu}_y$ the intercept vector.

Model 8 (TYPE 2B submodel in LISREL VI) is a model for studying the mean and covariance structures of $\mathbf{y}$-variables, with regression among $\mathbf{y}$-variables allowed. When $\boldsymbol{\beta}$ is null, the mean structures of $\mathbf{y}$ are specified in $\boldsymbol{\alpha}$ and the covariance structures are specified in $\boldsymbol{\Psi}$. This is similar to Model 6. However, there is an important distinction. In Model 6, the $\mathbf{x}$-variables are treated as exogenous (no model equation at all). But the $\mathbf{y}$-variables are treated as endogenous in Model 8 (with or without $\boldsymbol{\beta} = 0$). Consequently, the default parameterization would be different for these two submodels. See the section "Default Parameters" on page 1626 for details about the default parameterization.

Model 9 represents a modified version of the second-order factor model for $\mathbf{y}$. It would be a standard second-order factor model when $\boldsymbol{\beta}$ is null. This is the TYPE 3A submodel in LISREL VI. With $\boldsymbol{\beta}$ being null, $\boldsymbol{\eta}$ represents the first-order factors and $\boldsymbol{\xi}$ represents the second-order factors. The first- and second-order factor loading matrices are $\boldsymbol{\Lambda}_y$ and $\boldsymbol{\Gamma}$, respectively.

Model 10 is another form of factor model when $\boldsymbol{\beta}$ is null, with factors represented by $\boldsymbol{\xi}$ and manifest variables represented by $\mathbf{y}$. However, if $\boldsymbol{\beta}$ is indeed a null matrix in applications, you might want to use Model 5, in which the factor model specification is more direct and intuitive.

Default Parameters in the LISMOD Model

When a model matrix is defined in a LISMOD model, you can specify fixed values or free parameters for the elements of the matrix by the MATRIX statement. All other unspecified elements in the matrix are set by default. There are two types of default parameters for the LISMOD model matrices: one is free parameters; the other is fixed zeros.

The following sets of parameters are free parameters by default:

- the diagonal elements of the _THETAX_, _THETAY_, and _PSI_ matrices; these elements represent the error variances in the model
- all elements of the _PHI_ matrix; these elements represent the variances and covariance among exogenous variables in the model
- all elements in the _NUX_ and _NUY_ vectors if the mean structures are modeled; these elements represent the intercepts of the observed variables
- all elements in the _ALPHA_ vector if a YVAR= list is specified but an ETAVAR= list is not specified and the mean structures are modeled; these elements represent the intercepts of the $\mathbf{y}$-variables
- all elements in the _KAPPA_ vector if an XVAR= list is specified but an XIVAR= list is not specified and the mean structures are modeled; these elements represent the means of the $\mathbf{x}$-variables

PROC CALIS names the default free parameters with the _Add prefix and a unique integer suffix. You can override the default free parameters by explicitly specifying them as free, constrained, or fixed parameter in the MATRIX statements for the matrices.

Parameters that are not default free parameters in the LISMOD model are fixed zeros by default. You can override almost all of these default fixed zeros of the LISMOD model by using the MATRIX statements for the matrices. The only set of default fixed zeros that you cannot override is the set of the diagonal elements of the _BETA_ matrix. These fixed zeros reflect a model restriction that precludes variables from having direct effects on themselves.

The MSTRUCT Model

In contrast to other modeling languages where the mean and covariance structures are implied from the specification of equations, paths, variable-factor relations, mean parameters, variance parameters, or covariance parameters, the MSTRUCT modeling language is supported in PROC CALIS for modeling mean and covariance structures directly.

A simple example for using the MSTRUCT modeling language is the testing of a covariance model with equal variances and covariances. Suppose that a variable was measured five times in an experiment. The covariance matrix of these five measurements is hypothesized to have the structure

$$\mathbf{\Sigma} = \mathbf{\Sigma}(\theta)$$

where

$$\theta = (\phi, \tau)$$

and

$$\mathbf{\Sigma}(\theta) = \begin{pmatrix} \phi & \tau & \tau & \tau & \tau \\ \tau & \phi & \tau & \tau & \tau \\ \tau & \tau & \phi & \tau & \tau \\ \tau & \tau & \tau & \phi & \tau \\ \tau & \tau & \tau & \tau & \phi \end{pmatrix}$$

For model structures that are hypothesized directly in the covariance matrix, the MSTRUCT modeling language is the most convenient to use. You can also use other general modeling languages such as LINEQS, PATH, or RAM to fit the same model structures, but the specification is less straightforward and more error-prone. For convenience, models that are specified using the MSTRUCT modeling language are called MSTRUCT models. For applications of MSTRUCT modeling, see Yung, Browne, and Zhang (2015).

Model Matrices in the MSTRUCT Model

Suppose that there are p observed variables. The two model matrices, their names, their roles, and their dimensions are summarized in the following table.

Matrix	Name	Description	Dimensions
$\mathbf{\Sigma}$	_COV_ or _MSTRUCTCOV_	Structured covariance matrix	$p \times p$
$\boldsymbol{\mu}$	_MEAN_ or _MSTRUCTMEAN_	Structured mean vector	$p \times 1$

Specification of the MSTRUCT Model

Specifying Variables

In the MSTRUCT statement, you specify the list of p manifest variables of interest in the VAR= list. For example, you specify v1–v5 as the variables to be analyzed in your MSTRUCT model by this statement:

```
mstruct VAR= v1 v2 v3 v4 v5;
```

See the MSTRUCT statement on page 1656 for details about the syntax.

The manifest variables in the VAR= list must be referenced in the input set. The number of variables in the VAR= list determines the dimensions of the _COV_ and the _MEAN_ matrices in the model. In addition, the order of variables determines the order of row and column variables in the model matrices.

Specifying Parameters in Model Matrices

Denote the parameter vector in the MSTRUCT model as θ. The dimension of θ depends on your hypothesized model. In the preceding example, θ contains two parameters in ϕ and τ. You can use the MATRIX statement to specify these parameters in the _COV_ matrix:

```
matrix _COV_ [1,1] = 5*phi,  /* phi for all diagonal elements */
             [2, ] = tau,    /* tau for all off-diagonal elements */
             [3, ] = 2*tau,
             [4, ] = 3*tau,
             [5, ] = 4*tau;
```

In this MATRIX statement, the five diagonal elements, starting from the `[1,1]` element of the covariance matrix, are fitted by the phi parameter. The specification `5*phi` is a shorthand for specifying phi five times, once for each of the five diagonal elements in the covariance matrix. All other lower triangular elements are fitted by the tau parameter, as shown in the MATRIX statement. For example, with `[3,]` the elements starting from the first element of the third row of the _COV_ matrix are parameterized by the tau parameter. The specification `2*tau` repeats the specification two times, meaning that the `[3,1]` and `[3,2]` elements are both fitted by the same parameter tau. Similarly, all lower triangular elements (not including the diagonal elements) of the _COV_ matrix are fitted by the tau parameter. The specification of the upper triangular elements (diagonal excluded) of the _COV_ matrix is not needed because the _COV_ matrix is symmetric. The specification in the lower triangular elements is transferred automatically to the upper triangular elements. See the MATRIX statement on page 1638 for details about the syntax.

Default Parameters in the MSTRUCT Model

By using the MATRIX statements, you can specify either fixed values or free parameters (with or without initial values) for the elements in the _COV_ and _MEAN_ model matrices. If some or all elements are not specified, default parameters are applied to the MSTRUCT. There are two types of default parameters: one is free parameters; the other is fixed zeros. They are applied in different situations.

If you do not specify *any* elements of the _COV_ matrix with the MATRIX statement, all elements of the _COV_ matrix are free parameters by default. PROC CALIS names the default free parameters with the _Add prefix and a unique integer suffix. However, if you specify *at least one* fixed or free parameter of the _COV_ matrix with the MATRIX statement, then all other unspecified elements of the _COV_ matrix are fixed zeros by default.

If the mean structures are modeled, the same treatment applies to the _MEAN_ vector. That is, if you do not specify any elements of the _MEAN_ vector with the MATRIX statement, all elements of the _MEAN_ vector are free parameters by default. However, if you specify *at least one* fixed or free parameter of the _MEAN_ vector with the MATRIX statement, then all other unspecified elements of the _MEAN_ vector are fixed zeros by default.

How and Why the Default Parameters Are Treated Differently in the MSTRUCT Model

Notice that the default parameter treatment of the MSTRUCT model is quite different from other types of models such as FACTOR, LINEQS, LISMOD, RAM, or PATH. For these models, unspecified variances and covariances among exogenous variables are all free parameters by default. However, for the MSTRUCT model, either default free parameters or fixed zeros are generated depending on whether at least one element of the covariance matrix is specified. The reason for this different treatment is that you fit the covariance structure directly by using the MSTRUCT modeling language. Hence, in an MSTRUCT model there is no information regarding the functional relationships among the variables that indicates whether the variables

are exogenous or endogenous in the model. Hence, PROC CALIS cannot assume default free parameters based on the exogenous or endogenous variable types.

Because of this special default parameter treatment, when fitting an MSTRUCT model you must make sure that each diagonal element in your _COV_ matrix is set as a free, constrained, or fixed parameter, in accordance with your theoretical model. If you specify some elements in the model matrix but omit the specification of other diagonal elements, the default fixed zero variances would lead to a nonpositive definite _COV_ model matrix, making the model fitting problematic.

The PATH Model

The PATH modeling language is supported in PROC CALIS as a more intuitive modeling tool. It is designed so that specification by using the PATH modeling language translates effortlessly from the path diagram. For example, consider the following simple path diagram:

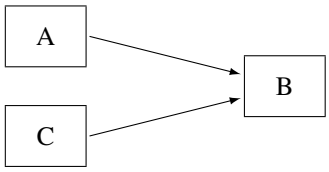

You can use the following PATH statement to specify the paths easily:

```
path    A ===> B ,
        C ===> B ;
```

There are two path entries in the PATH statement: one is for the path **A ===> B**, and the other is for the path **C ===> B**. Sometimes you might want to name the effect parameters in the path diagram, as shown in the following:

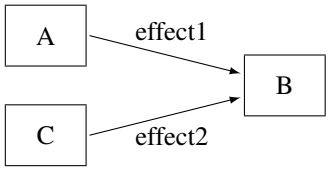

You can specify the paths and the parameters together in the following statement:

```
path    A ===> B   = effect1,
        C ===> B   = effect2;
```

In the first entry of the PATH statement, the path **A ===> B** is specified together with the path coefficient (effect) effect1. Similarly, in the second entry, the **C ===> B** path is specified together with the effect parameter effect2. In addition to the path coefficients (effects) in the path diagram, you can also specify other types of parameters by using the PVAR and PCOV statements. See the section "A Structural Equation Example" on page 1521 for a more detailed example of the PATH model specification.

Despite its simple representation of the path diagram, the PATH modeling language is general enough to handle a wide class of structural models that can also be handled by other general modeling languages such as LINEQS, LISMOD, or RAM. For brevity, models specified by the PATH modeling language are called PATH models.

Types of Variables in the PATH Model

When you specify the paths in the PATH model, you typically use arrows (such as **<===** or **===>**) to denote "causal" paths. For example, in the preceding path diagram or the PATH statement, you specify that B is an outcome variable with predictors A and C, respectively, in two paths. An outcome variable is the variable being pointed to in a path specification, while the predictor variable is the one where the arrow starts from.

Whereas the outcome–predictor relationship describes the roles of variables in each single path, the endogenous–exogenous relationship describes the roles of variables in the entire system of paths. In a system of path specification, a variable is endogenous if it is pointed to by at least one single-headed arrow or it serves as an outcome variable in at least one path. Otherwise, it is exogenous. In the preceding path diagram, for example, variable B is endogenous and both variables A and C are exogenous. Note that although any variable that serves as an outcome variable at least in one path must be endogenous, it does not mean that all endogenous variables must serve only as outcome variables in all paths. An endogenous variable in a model might also serve as a predictor variable in a path. For example, variable B in the following PATH statement is an endogenous variable, and it serves as an outcome variable in the first path but as a predictor variable in the second path.

```
path    A ===> B   = effect1,
        B ===> C   = effect2;
```

A variable is a manifest or observed variable in the PATH model if it is measured and exists in the input data set. Otherwise, it is a latent variable. Because error variables are not explicitly defined in the PATH modeling language, all latent variables that are named in the PATH model are *factors*, which are considered to be the systematic source of effects in the model. Each manifest variable in the PATH model can be endogenous or exogenous. The same is true for any latent factor in the PATH model.

Because you do not name error variables in the PATH model, you do not need to specify paths from errors to any endogenous variables. Error terms are implicitly assumed for all endogenous variables in the PATH model. Although error variables are not named in the PATH model, the error variances are expressed equivalently as partial variances of the associated endogenous variables. These partial variances are set by default in the PATH modeling language. Therefore, you do not need to specify error variance parameters explicitly unless constraints on these parameters are desirable in the model. You can use the PVAR statement to specify the error variance or partial variance parameters explicitly.

Naming Variables in the PATH Model

Manifest variables in the PATH model are referenced in the input data set. Their names must not be longer than 32 characters. There are no further restrictions beyond those required by the SAS System. You use the names of manifest variables directly in the PATH model specification.

Because you do not name error variables in the PATH model, all latent variables named in the PATH model specification are factors (non-errors). Factor names in the PATH model must not be longer than 32 characters, and they should be different from the manifest variables. Unlike the LINEQS model, you do not need to use 'F' or 'f' prefix to denote latent factors in the PATH model. As a general naming convention, you should not use Intercept as either a manifest or latent variable name. See the section "Naming Variables and Parameters" on page 1777 for these general rules about naming variables and parameters.

Specification of the PATH Model

(1) Specification of Effects or Paths

You specify the "causal" paths or linear functional relationships among variables in the PATH statement. For example, if there is a path from v2 to v1 in your model and the effect parameter is named parm1 with a starting value at 0.5, you can use either of these specifications:

```
path     v1 <===  v2    = parm1(0.5);

path     v2 ===>  v1    = parm1(0.5);
```

If you have more than one path in your model, path specifications should be separated by commas, as shown in the following PATH statement:

```
path
   v1 <===  v2   = parm1(0.5),
   v2 <===  v3   = parm2(0.3);
```

Because the PATH statement can be used only once in each model specification, all paths in the model must be specified together in a single PATH statement. See the PATH statement on page 1665 for more details about the syntax.

(2) Specification of Variances and Partial (Error) Variances

If v2 is an exogenous variable in the PATH model and you want to specify its variance as a parameter named parm2 with a starting value at 10, you can use the following PVAR statement specification:

```
pvar     v2  = parm2(10.);
```

If v1 is an endogenous variable in the PATH model and you want to specify its partial variance or error variance as a parameter named parm3 with a starting value at 5.0, you can also use the following PVAR statement specification:

```
pvar     v1 = parm3(5.0);
```

Therefore, the PVAR statement can be used for both exogenous and endogenous variables. When a variable in the statement is exogenous (which can be automatically determined by PROC CALIS), you are specifying the variance parameter of the variable. Otherwise, you are specifying the partial or error variance for an endogenous variable.

You do not need to supply the parameter names for the variances or partial variances if these parameters are not constrained. For example, the following statement specifies the unnamed free parameters for variances or partial variances of v1 and v2:

```
pvar     v1 v2;
```

If you have more than one variance or partial variance parameter to specify in your model, you can put a variable list on the left-hand side of the equal sign, and a parameter list on the right-hand side, as shown in the following PVAR statement specification:

```
pvar
   v1 v2 v3 = parm1(0.5) parm2 parm3;
```

In the specification, variance or partial variance parameters for variables v1–v3 are parm1, parm2, and parm3, respectively. Only parm1 is given an initial value at 0.5. The initial values for other parameters are generated by PROC CALIS.

You can also separate the specifications into several entries in the PVAR statement. Entries should be separated by commas. For example, the preceding specification is equivalent to the following specification:

```
pvar
   v1 = parm1 (0.5),
   v2 = parm2,
   v3 = parm3;
```

Because the PVAR statement can be used only once in each model specification, all variance and partial variance parameters in the model must be specified together in a single PVAR statement. See the PVAR statement on page 1690 for more details about the syntax.

(3) Specification of Covariances and Partial Covariances

If you want to specify the (partial) covariance between two variables v3 and v4 as a parameter named parm4 with a starting value at 3, you can use the following PCOV statement specification:

```
pcov  v3  v4 = parm4 (5.);
```

Whether parm4 is a covariance or partial covariance parameter depends on the variable types of v3 and v4. If both v3 and v4 are exogenous variables (manifest or latent), parm4 is a covariance parameter between v3 and v4. If both v3 and v4 are endogenous variables (manifest or latent), parm4 is a parameter for the covariance between the errors for v3 and v4. In other words, it is a partial covariance or error covariance parameter for v3 and v4.

A less common case is when one of the variables is exogenous and the other is endogenous. In this case, parm4 is a parameter for the partial covariance between the endogenous variable and the exogenous variable, or the covariance between the error for the endogenous variable and the exogenous variable. Fortunately, such covariances are relatively uncommon in statistical modeling. Their uses confuse the roles of systematic and unsystematic sources in the model and lead to difficulties in interpretations. Therefore, you should almost always avoid this kind of partial covariance.

Like the syntax of the PVAR statement, you can specify a list of (partial) covariance parameters in the PCOV statement. For example, consider the following statement:

```
pcov
   v1 v2 = parm4,
   v1 v3 = parm5,
   v2 v3 = parm6;
```

In the specification, three (partial) covariance parameters parm4, parm5, and parm6 are specified, respectively, for the variable pairs (v1,v2), (v1,v3), and (v2,v3). Entries for (partial) covariance specification are separated by commas.

Again, if all these covariances are not constrained, you can omit the names for the parameters. For example, the preceding specification can be specified as the following statement when the three covariances are free parameters in the model:

```
pcov
   v1 v2,
   v1 v3,
   v2 v3;
```

Or, you can simply use the following within-list covariance specification:

```
pcov
   v1 v2 v3;
```

Three covariance parameters are generated by this specification.

Because the PCOV statement can be used only once in each model specification, all covariance and partial covariance parameters in the model must be specified together in a single PCOV statement. See the PCOV statement on page 1687 for more details about the syntax.

(4) Specification of Means and Intercepts

Means and intercepts are specified when the mean structures of the model are of interest. You can specify mean and intercept parameters in the MEAN statement. For example, consider the following statement:

```
mean     V5 = parm5(11.);
```

If V5 is an exogenous variable (which is determined by PROC CALIS automatically), you are specifying parm5 as the mean parameter of V5. If V5 is an endogenous variable, you are specifying parm5 as the intercept parameter for V5.

Because each named variable in the PATH model is either exogenous or endogenous (exclusively), each variable in the PATH model would have either a mean or an intercept parameter (but not both) to specify in the MEAN statement. Like the syntax of the PVAR statement, you can specify a list of mean or intercept parameters in the MEAN statement. For example, in the following statement you specify a list of mean or intercept parameters for variables v1–v4:

```
mean
   v1-v4 = parm6-parm9;
```

This specification is equivalent to the following specification with four entries of parameter specifications:

```
mean
   v1 = parm6,
   v2 = parm7,
   v3 = parm8,
   v4 = parm9;
```

Again, entries in the MEAN statement must be separated by commas, as shown in the preceding statement.

Because the MEAN statement can be used only once in each model specification, all mean and intercept parameters in the model must be specified together in a single MEAN statement. See the MEAN statement on page 1651 for more details about the syntax.

Specifying Parameters without Initial Values

If you do not have any knowledge about the initial value for a parameter, you can omit the initial value specification and let PROC CALIS compute it. For example, you can provide just the parameter locations and parameter names as in the following specification:

```
path    v1 <=== v2   = parm1;
   pvar v2 = parm2,
        v1 = parm3;
```

Specifying Fixed Parameter Values

If you want to specify a fixed parameter value, you do not need to provide a parameter name. Instead, you provide the fixed value (without parentheses) in the specification.

For example, in the following statement the path coefficient for the path is fixed at 1.0 and the (partial) variance of F1 is also fixed at 1.0:

```
path    v1 <=== F1  = 1.;
   pvar
        F1 = 1.;
```

A Complete PATH Model Specification Example

The following specification shows a more complete PATH model specification:

```
path     v1 <===  v2 ,
         v1 <===  v3 ;
   pvar  v1,
         v2 = parm3,
         v3 = parm3;
   pcov  v3 v2 =  parm5(5.);
```

The two paths specified in the PATH statement have unnamed free effect parameters. These parameters are named by PROC CALIS with the _Parm prefix and unique integer suffixes. The error variance of v1 is an unnamed parameter, while the variances of v2 and v3 are constrained by using the same parameter parm3. The covariance between v2 and v3 is a free parameter named parm5, with a starting value of 5.0.

Default Parameters in the PATH Model

There are two types of default parameters of the PATH model. One is the free parameters; the other is the fixed constants.

The following sets of parameters are free parameters by default:

- the variances or partial (or error) variances of all variables, manifest or latent
- the covariances among all exogenous (independent) manifest or latent variables
- the means of all exogenous (independent) manifest variables if the mean structures are modeled
- the intercepts of all endogenous (dependent) manifest variables if the mean structures are modeled

For each of the default free parameters, PROC CALIS generates a parameter name with the _Add prefix and a unique integer suffix. Parameters that are not default free parameters in the PATH model are fixed zeros by default. You can override almost all of the default zeros of the PATH model by using the MEAN, PATH, PCOV, and MEAN statements. The only exception is the single-headed path that has the same variable on both sides. That is, the following specification is not accepted by PROC CALIS:

```
path     v1  <===  v1    = parm;
```

This path should always has a zero coefficient, which is treated as a model restriction that prevents a variable from having a direct effect on itself.

Relating the PATH Model to the RAM Model

Mathematically, the PATH model is essentially the RAM model. You can consider the PATH model to share exactly the same set of model matrices as in the RAM model. See the section "Model Matrices in the RAM Model" on page 1770 and the section "Summary of Matrices and Submatrices in the RAM Model" on page 1773 for details about the RAM model matrices. In the RAM model, the **A** matrix contains effects or path coefficients for describing relationships among variables. In the PATH model, you specify these effect or coefficient parameters in the PATH statement. The **P** matrix in the RAM model contains (partial) variance and (partial) covariance parameters. In the PATH model, you use the PVAR and PCOV statements to specify these parameters. The **W** vector in the RAM model contains the mean and intercept parameters, while in the PATH model you use the MEAN statement to specify these parameters. By using these model matrices in the PATH model, the covariance and mean structures are derived in the same way as they are derived in the RAM model. See the section "The RAM Model" on page 1769 for derivations of the model structures.

The RAM Model

The RAM modeling language is adapted from the basic RAM model developed by McArdle (1980). For brevity, models specified by the RAM modeling language are called RAM models. You can also specify these so-called RAM models by other general modeling languages that are supported in PROC CALIS.

Types of Variables in the RAM Model

A variable in the RAM model is manifest if it is observed and is defined in the input data set. A variable in the RAM model is latent if it is not manifest. Because error variables are not explicitly named in the RAM model, all latent variables in the RAM model are considered as factors (non-error-type latent variables).

A variable in the RAM model is endogenous if it ever serves as an outcome variable in the RAM model. That is, an endogenous variable has at least one path (or an effect) from another variable in the model. A variable is exogenous if it is not endogenous. Endogenous variables are also referred to as dependent variables, while exogenous variables are also referred to as independent variables.

In the RAM model, distinctions between exogenous and endogenous and between latent and manifest for variables are not essential to the definitions of model matrices, although they are useful for conceptual understanding when the model matrices are partitioned.

Naming Variables in the RAM Model

Manifest variables in the RAM model are referenced in the input data set. Their names must not be longer than 32 characters. There are no further restrictions beyond those required by the SAS System.

Latent variables in the RAM model are those not being referenced in the input data set. Their names must not be longer than 32 characters. Unlike the LINEQS model, you do not need to use any specific prefix (for example, 'F' or 'f') for the latent factor names. The reason is that error or disturbance variables in the RAM model are not named explicitly in the RAM model. Thus, any variable names that are not referenced in the input data set are for latent factors.

As a general naming convention, you should not use Intercept as either a manifest or latent variable name.

Model Matrices in the RAM Model

In terms of the number of model matrices involved, the RAM model is the simplest among all the general structural equations models that are supported by PROC CALIS. Essentially, there are only three model matrices in the RAM model: one for the interrelationships among variables, one for the variances and covariances, and one for the means and intercepts. These matrices are discussed in the following subsections.

Matrix $\mathbf{A}$ *(*$n_a \times n_a$*) : Effects of Column Variables on Row Variables*

The row and column variables of matrix **A** are the set of manifest and latent variables in the RAM model. Unlike the LINEQS model, the set of latent variables in the RAM model matrix does not include the error or disturbance variables. Each entry or element in the **A** matrix represents an effect of the associated column variable on the associated row variable or a path coefficient from the associated column variable to the associated row variable. A zero entry means an absence of a path or an effect.

The pattern of matrix **A** determines whether a variable is endogenous or exogenous. A variable in the RAM model is endogenous if its associated row in the **A** matrix has at least one nonzero entry. Any other variable in the RAM model is exogenous.

Mathematically, you do not need to arrange the set of variables for matrix **A** in a particular order, as long as the order of variables is the same for the rows and the columns. However, arranging the variables according to whether they are endogenous or exogenous is useful for showing the partitions of the model matrices and certain mathematical properties. See the section "Partitions of the RAM Model Matrices and Some Restrictions" on page 1771 for details.

Matrix $\mathbf{P}$ *(*$n_a \times n_a$*): Variances, Covariances, Partial Variances, and Partial Covariances*

The row and column variables of matrix **P** refer to the same set of manifest and latent variables that are defined in the RAM model matrix **A**. The diagonal entries of **P** contain variances or partial variances of variables. If a variable is exogenous, then the corresponding diagonal element in the **P** matrix represents its variance. Otherwise, the corresponding diagonal element in the **P** matrix represents its partial variance. This partial variance is an unsystematic source of variance that is not explained by the interrelationships of variables in the model. In most cases, you can interpret a partial variance as the error variance for an endogenous variable.

The off-diagonal elements of **P** contain covariances or partial covariances among variables. An off-diagonal element in **P** that is associated with exogenous row and column variables represents covariance between the two exogenous variables. An off-diagonal element in **P** that is associated with endogenous row and column variables represents *partial* covariance between the two variables. This partial covariance is unsystematic, in the sense that it is not explained by the interrelationships of variables in the model. In most cases, you can interpret a partial covariance as the error covariance between the two endogenous variables involved. An off-diagonal element in **P** that is associated with one exogenous variable and one endogenous variable in the row and column represents the covariance between the exogenous variable and the error of the endogenous variable. While this interpretation sounds a little awkward and inelegant, this kind of covariance, fortunately, is rare in most applications.

Vector $\mathbf{W}$ *(*$n_a \times 1$*): Intercepts and Means*

The row variables of vector **W** refer to the same set of manifest and latent variables that are defined in the RAM model matrix **A**. Elements in **W** represent either intercepts or means. An element in **W** that is associated with an exogenous row variable represents the mean of the variable. An element in **W** that is associated with an endogenous row variable represents the intercept term for the variable.

Covariance and Mean Structures

Assuming that $(\mathbf{I} - \mathbf{A})$ is invertible, where $\mathbf{I}$ is an identity matrix of the same dimension as $\mathbf{A}$, the structured covariance matrix of all variables (including latent variables) in the RAM model is shown as follows:

$$\boldsymbol{\Sigma}_a = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{P}(\mathbf{I} - \mathbf{A})^{-1\prime}$$

The structured mean vector of all variables is shown as follows:

$$\boldsymbol{\mu}_a = (\mathbf{I} - \mathbf{A})^{-1}\mathbf{W}$$

The covariance and mean structures of all manifest variables are obtained by selecting the elements in $\boldsymbol{\Sigma}_a$ and $\boldsymbol{\mu}_a$. This can be achieved by defining a selection matrix $\mathbf{G}$ of dimensions $n \times n_a$, where n is the number of manifest variables in the model. The selection matrix $\mathbf{G}$ contains zeros and ones as its elements. Each row of $\mathbf{G}$ has exactly one nonzero element at the position that corresponds to the location of a manifest row variable in $\boldsymbol{\Sigma}_a$ or $\boldsymbol{\mu}_a$. With each row of $\mathbf{G}$ selecting a distinct manifest variable, the structured covariance matrix of all manifest variables is expressed as the following:

$$\boldsymbol{\Sigma} = \mathbf{G}\boldsymbol{\Sigma}_a\mathbf{G}'$$

The structured mean vector of all observed variables is expressed as the following:

$$\boldsymbol{\mu} = \mathbf{G}\boldsymbol{\mu}_a$$

Partitions of the RAM Model Matrices and Some Restrictions

There are some model restrictions in the RAM model matrices. Although these restrictions do not affect the derivation of the covariance and mean structures, they are enforced in the RAM model specification.

For convenience, it is useful to assume that n_a variables are arranged in the order of n_d endogenous (or dependent) variables and the n_i exogenous (independent) variables in the rows and columns of the model matrices.

Model Restrictions on the $\mathbf{A}$ *Matrix*

The $\mathbf{A}$ matrix is partitioned as

$$\mathbf{A} = \begin{pmatrix} \boldsymbol{\beta} & \boldsymbol{\gamma} \\ 0 & 0 \end{pmatrix}$$

where $\boldsymbol{\beta}$ is an $n_d \times n_d$ matrix for paths or effects from (column) endogenous variables to (row) endogenous variables and $\boldsymbol{\gamma}$ is an $n_d \times n_i$ matrix for paths (effects) from (column) exogenous variables to (row) endogenous variables.

As shown in the matrix partitions, there are four submatrices. The two submatrices at the lower parts are seemingly structured to zeros. However, this should not be interpreted as restrictions imposed by the model. The zero submatrices are artifacts created by the exogenous-endogenous arrangement of the row and column variables. The only restriction on the $\mathbf{A}$ matrix is that the diagonal elements must all be zeros. This implies that the diagonal elements of the submatrix $\boldsymbol{\beta}$ are also zeros. This restriction prevents a direct path from any endogenous variable to itself. There are no restrictions on the pattern of $\boldsymbol{\gamma}$.

It is useful to denote the lower partitions of the $\mathbf{A}$ matrix by $\mathbf{A}_{LL}$ (lower left) and $\mathbf{A}_{LR}$ (lower right) so that

$$\mathbf{A} = \begin{pmatrix} \boldsymbol{\beta} & \boldsymbol{\gamma} \\ \mathbf{A}_{LL} & \mathbf{A}_{LR} \end{pmatrix}$$

Although they are zero matrices in the initial model specification, their entries could become non-zero (paths) in an improved model when you modify your model by using the Lagrange multiplier statistics (see the section "Modification Indices" on page 1848 or the MODIFICATION option). Hence, you might need to reference these two submatrices when you apply the customized LM tests on them during the model modification process (see the LMTESTS statement).

For the purposes of defining specific parameter regions in customized LM tests, you might also partition the $\mathbf{A}$ matrix in other ways. First, you can partition $\mathbf{A}$ into the left and right portions,

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{Left} & \mathbf{A}_{Right} \end{pmatrix}$$

where $\mathbf{A}_{Left}$ is top-down concatenation of the $\boldsymbol{\beta}$ and $\mathbf{A}_{LL}$ matrices and $\mathbf{A}_{Right}$ is the top-down concatenation of the $\boldsymbol{\gamma}$ and $\mathbf{A}_{LR}$ matrices. Second, you can partition $\mathbf{A}$ into the upper and lower portions,

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{Upper} \\ \mathbf{A}_{Lower} \end{pmatrix}$$

where $\mathbf{A}_{Upper}$ is the side-by-side concatenation of the $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ matrices and $\mathbf{A}_{Lower}$ is the side-by-side concatenation of the $\mathbf{A}_{LL}$ and $\mathbf{A}_{LR}$ matrices.

In your initial model, because of the arrangement of the endogenous and exogenous variables $\mathbf{A}_{Lower}$ is a null matrix. But if you improve your model by applying the LM tests on the entries in $\mathbf{A}_{Lower}$, some of these entries might become free paths in your improved model. Hence, some exogenous variables in your initial model now become endogenous variables in your improved model. For this reason, $\mathbf{A}_{Lower}$ is also designated as a parameter region for *new* endogenous variables, which is exactly what the NEWENDO region means in the LMTESTS statement.

Partition of the $\mathbf{P}$ Matrix

The $\mathbf{P}$ matrix is partitioned as

$$\mathbf{P} = \begin{pmatrix} \mathbf{P}_{11} & \mathbf{P}'_{21} \\ \mathbf{P}_{21} & \mathbf{P}_{22} \end{pmatrix}$$

where $\mathbf{P}_{11}$ is an $n_d \times n_d$ partial covariance matrix for the endogenous variables, $\mathbf{P}_{22}$ is an $n_i \times n_i$ covariance matrix for the exogenous variables, and $\mathbf{P}_{21}$ is an $n_i \times n_d$ covariance matrix between the exogenous variables and the error terms for the endogenous variables. Because $\mathbf{P}$ is symmetric, $\mathbf{P}_{11}$ and $\mathbf{P}_{22}$ are also symmetric.

There are virtually no model restrictions placed on these submatrices. However, in most statistical applications, errors for endogenous variables represent unsystematic sources of effects and therefore they are not to be correlated with other systematic sources such as the exogenous variables in the RAM model. This means that in most practical applications $\mathbf{P}_{21}$ would be a null matrix, although this is not enforced in PROC CALIS.

Partition of the $\mathbf{W}$ Vector

The $\mathbf{W}$ vector is partitioned as

$$\mathbf{W} = \begin{pmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\nu} \end{pmatrix}$$

where $\boldsymbol{\alpha}$ is an $n_d \times 1$ vector for intercepts of the endogenous variables and $\boldsymbol{\nu}$ is an $n_i \times 1$ vector for the means of the exogenous variables. There is no model restriction on these subvectors.

Summary of Matrices and Submatrices in the RAM Model

Let n_a be the total number of manifest and latent variables in the RAM model. Of these n_a variables, n_d are endogenous and n_i are exogenous. Suppose that the rows and columns of the RAM model matrices $\mathbf{A}$ and $\mathbf{P}$ and the rows of $\mathbf{W}$ are arranged in the order of n_d endogenous variables and then n_i exogenous variables. The names, roles, and dimensions of the RAM model matrices and submatrices are summarized in the following table.

Matrix	Name	Description	Dimensions
Model Matrices			
$\mathbf{A}$	_A_ or _RAMA_	Effects of column variables on row variables, or paths from the column variables to the row variables	$n_a \times n_a$
$\mathbf{P}$	_P_ or _RAMP_	(Partial) variances and covariances	$n_a \times n_a$
$\mathbf{W}$	_W_ or _RAMW_	Intercepts and means	$n_a \times 1$
Submatrices			
$\boldsymbol{\beta}$	_RAMBETA_	Effects of endogenous variables on endogenous variables	$n_d \times n_d$
$\boldsymbol{\gamma}$	_RAMGAMMA_	Effects of exogenous variables on endogenous variables	$n_d \times n_i$
$\mathbf{A}_{LL}$	_RAMA_LL_	The null submatrix at the lower left portion of _A_	$n_i \times n_d$
$\mathbf{A}_{LR}$	_RAMA_LR_	The null submatrix at the lower right portion of _A_	$n_i \times n_i$
$\mathbf{A}_{Left}$	_RAMA_LEFT_	The left portion of _A_, including $\boldsymbol{\beta}$ and $\mathbf{A}_{LL}$	$n_a \times n_d$
$\mathbf{A}_{Right}$	_RAMA_RIGHT_	The right portion of _A_, including $\boldsymbol{\gamma}$ and $\mathbf{A}_{LR}$	$n_a \times n_i$
$\mathbf{A}_{Upper}$	_RAMA_UPPER_	The upper portion of _A_, including $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$	$n_d \times n_a$
$\mathbf{A}_{Lower}$	_RAMA_LOWER_	The lower portion of _A_, including $\mathbf{A}_{LL}$ and $\mathbf{A}_{LR}$	$n_i \times n_a$
$\mathbf{P}_{11}$	_RAMP11_	Error variances and covariances for endogenous variables	$n_d \times n_d$
$\mathbf{P}_{21}$	_RAMP21_	Covariances between exogenous variables and error terms for endogenous variables	$n_d \times n_i$
$\mathbf{P}_{22}$	_RAMP22_	Variances and covariances for exogenous variables	$n_i \times n_i$
$\boldsymbol{\alpha}$	_RAMALPHA_	Intercepts for endogenous variables	$n_d \times 1$
$\boldsymbol{\nu}$	_RAMNU_	Means for exogenous variables	$n_i \times 1$

Specification of the RAM Model

In PROC CALIS, the RAM model specification is a matrix-oriented modeling language. That is, you have to define the row and column variables for the model matrices and specify the parameters in terms of matrix entries. The VAR= option specifies the variables (including manifest and latent) in the model. For example, the following statement specifies five variables in the model:

```
RAM
   var= v1 v2 v3;
```

The order of variables in the VAR= option is important. The same order is used for the row and column variables in the model matrices. After you specify the variables in the model, you can specify three types of parameters, which correspond to the elements in the three model matrices. The three types of RAM entries are described in the following.

(1) Specification of Effects or Paths in Model Matrix A

If there is a path from V2 to V1 in your model and the associated effect parameter is named parm1 with 0.5 as the starting value, you can use the following RAM statement:

```
RAM
   var= v1 v2 v3,
   _A_   1   2  parm1(0.5);
```

The *ram-entry* that starts with _A_ means that an element of the ram matrix **A** is being specified. The row number and the column number of this element are 1 and 2, respectively. With reference to the VAR= list, the row number 1 refers to variable v1, and the column number 2 refers to variable v2. Therefore, the effect of V2 on V1 is a parameter named parm1, with an initial value of 0.5.

You can specify fixed values in the *ram-entries* too. Suppose the effect of v3 on v1 is fixed at 1.0. You can use the following specification:

```
RAM
   var= v1 v2 v3,
   _A_   1   2  parm1(0.5),
   _A_   1   3  1.0;
```

(2) Specification of the Latent Factors in the Model

In the RAM model, you specify the list of variables in VAR= list of the RAM statement. The list of variables can include the latent variables in the model. Because observed variables have references in the input data sets, those variables that do not have references in the data sets are treated as latent factors automatically. Unlike the LINEQS model, you do not need to use ‘F’ or ‘f’ prefix to denote latent factors in the RAM model. It is recommended that you use meaningful names for the latent factors. See the section “Naming Variables and Parameters” on page 1777 for the general rules about naming variables and parameters.

For example, suppose that SES_Factor and Education_Factor are names that are not used as variable names in the input data set. These two names represent two latent factors in the model, as shown in the following specification:

```
RAM
   var= v1 v2 v3 SES_FACTOR Education_Factor,
   _A_   1    4    b1,
   _A_   2    5    b2,
   _A_   3    5    1.0;
```

This specification shows that the effect of SES_Factor on v1 is a free parameter named b1, and the effects of Education_Factor on v2 and v3 are a free parameter named b2 and a fixed value of 1.0, respectively.

However, naming latent factors is not compulsory. The preceding specification is equivalent to the following specification:

```
RAM
   var= v1 v2 v3,
   _A_   1    4    b1,
   _A_   2    5    b2,
   _A_   3    5    1.0;
```

Although you do not name the fourth and the fifth variables in the VAR= list, PROC CALIS generates the names for these two latent variables. In this case, the fourth variable is named _Factor1 and the fifth variable is named _Factor2.

(3) Specification of (Partial) Variances and (Partial) Covariances in Model Matrix P

Suppose now you want to specify the variance of v2 as a free parameter named parm2. You can add a new *ram-entry* for this variance parameter, as shown in the following statement:

```
RAM
   var= v1 v2 v3,
   _A_   1   2  parm1(0.5),
   _A_   1   3  1.0,
   _P_   2   2  parm2;
```

The *ram-entry* that starts with _P_ means that an element of the RAM matrix **P** is being specified. The (2,2) element of **P**, which is the variance of v2, is a parameter named parm2. You do not specify an initial value for this parameter.

You can also specify the error variance of v1 similarly, as shown in the following statement:

```
RAM
   var= v1 v2 v3,
   _A_   1   2  parm1(0.5),
   _A_   1   3  1.0,
   _P_   2   2  parm2,
   _P_   1   1;
```

In the last *ram-entry*, the (1,1) element of **P**, which is the error variance of v1, is an unnamed free parameter.

Covariance parameters are specified in the same manner. For example, the following specification adds a *ram-entry* for the covariance parameter between v2 and v3:

```
RAM
   var= v1 v2 v3,
   _A_   1   2  parm1(0.5),
   _A_   1   3  1.0,
   _P_   2   2  parm2,
   _P_   1   1,
   _P_   2   3  (.5);
```

The covariance between v2 and v3 is an unnamed parameter with an initial value of 0.5.

(4) Specification of Means and Intercepts in Model Matrix _W_

To specifying means or intercepts, you need to start the *ram-entries* with the _W_ keyword. For example, the last two entries of following statement specify the intercept of v1 and the mean of v2, respectively:

```
RAM
   var= v1 v2 v3,
   _A_   1   2  parm1(0.5),
   _A_   1   3  1.0,
   _P_   2   2  parm2,
   _P_   1   1 ,
   _P_   2   3  (.5),
   _W_   1   1  int_v1,
   _W_   2   1  mean_v2;
```

The intercept of v1 is a free parameter named int_v1, and the mean of v2 is a free parameter named mean_v2.

Default Parameters in the RAM Model

There are two types of default of parameters of the RAM model in PROC CALIS. One is the free parameters; the other is the fixed zeros.

By default, certain sets of model matrix elements in the RAM model are free parameters. These parameters are set automatically by PROC CALIS, although you can also specify them explicitly in the *ram-entries*. In general, default free parameters enable you to specify only what are absolutely necessary for defining your model. PROC CALIS automatically sets those commonly assumed free parameters so that you do not need to specify them routinely. The sets of default free parameters of the RAM model are as follows:

- Diagonal elements of the _P_ matrix—this includes the variance of exogenous variables (latent or observed) and error variances of all endogenous variables (latent or observed)
- The off-diagonal elements that pertain to the exogenous variables of the _P_ matrix—this includes all the covariances among exogenous variables, latent or observed
- If the mean structures are modeled, the elements that pertain to the observed variables (but not the latent variables) in the _W_ vector— this includes all the means of exogenous observed variables and the intercepts of all endogenous observed variables

For example, suppose you are fitting a RAM model with three observed variables x1, x2, and y3, you specify a simple multiple-regression model with x1 and x2 predicting y3 by the following statements:

```
proc calis meanstr;
   ram  var= x1-x2 y3,
        _A_ 3 1 ,
        _A_ 3 2 ;
```

In the RAM statement, you specify that path coefficients represented by **_A_[3,1]** and **_A_[3,2]** are free parameters in the model. In addition to these free parameters, PROC CALIS sets several other free parameters by default. **_P_[1,1]**, **_P_[2,2]**, and **_P_[3,3]** are set as free parameters for the variance of x1, the variance of x2, and the error variance of x3, respectively. **_P_[2,1]** (and hence **_P_[1,2]**) is set as a free parameter for the covariance between the exogenous variables x1 and x2. Because the mean structures are also analyzed by the MEANSTR option in the PROC CALIS statement, **_W_[1,1]**, **_W_[2,1]**, and **_W_[3,1]** are also set as free parameters for the mean of x1, the mean of x2, and the intercept of x3, respectively. In the current situation, this default parameterization is consistent with using PROC REG for multiple regression analysis, where you only need to specify the functional relationships among variables.

If a matrix element is not a default free parameter in the RAM model, then it is a fixed zero by default. You can override almost all default fixed zeros in the RAM model matrices by specifying the *ram-entries*. The diagonal elements of the _A_ matrix are exceptions. These elements are always fixed zeros. You cannot set these elements to free parameters or other fixed values—this reflects a model restriction that prevents a variable from having a direct effect on itself.

Naming Variables and Parameters

Follow these rules when you name your variables:

- Use the usual naming conventions of the SAS System.
- Variable names are not more than 32 characters.
- When you create latent variable names, make sure that they are not used in the input data set that is being analyzed.
- For the LINEQS model, error or disturbance variables must start with ‘E’, ‘e’, ‘D’, or ‘d’.
- For the LINEQS model, non-error-type latent variables (that is, factors) must start with ‘F’ or ‘f’.
- For modeling languages other than LINEQS, names for errors or disturbances are not needed. As a result, you do not need to distinguish latent factors from errors or disturbances by using particular prefixes. Variable names that are not referenced in the analyzed data set are supposed to be latent factors.
- You should not use Intercept (case-insensitive) as a variable name in your data set or as a latent variable name in your model.

Follow these rules when you name your parameters:

- Use the usual naming conventions of the SAS System.
- Parameter names are not more than 32 characters.
- Use a prefix-name when you want to generate new parameter names automatically. A prefix-name contains a short string of characters called a “root,” followed by double underscores ‘__’. Each occurrence of such a prefix-name generates a new parameter name by replacing the two trailing underscores with a unique integer. For example, occurrences of Gen__ generate Gen1, Gen2, and so on.
- A special prefix-name is the one without a root—that is, it contains only double underscores ‘__’. Occurrences of ‘__’ generate _Parm1, _Parm2, and so on.
- PROC CALIS generates parameter names for default parameters to safeguard ill-defined models. These generated parameter names start with the _Add prefix and unique integer suffixes. For example, _Add1, _Add2, and so on.

- Avoid using parameter names that start with either _, _Add, or _Parm. These names might get confused with the names generated by PROC CALIS. The confusion might lead to unintended constraints to your model if the parameter names that you use match those generated by PROC CALIS.
- Avoid using parameter names that are roots of prefix-names. For example, you should not use Gen as a parameter name if Gen__ is also used in the same model specification. Although violation of this rule might not distort the model specification, it might cause ambiguities and confusion.

Finally, parameter names and variable names in PROC CALIS are not distinguished by explicit declarations. That is, a valid SAS name can be used as a parameter name or a variable name in any model that is supported by PROC CALIS. Whether a name in a model specification is for a parameter or a variable is determined by the syntactic structure. For example, consider the following path specification:

```
proc calis;
   path
      a ===> b    = c;
run;
```

PROC CALIS parses the path specification according to the syntactic structure of the PATH statement and determines that a and b are variable names and c is a parameter name. Consider another specification as follows:

```
proc calis;
   path
      a ===> b    = b;
run;
```

This is a syntactically correct specification. Variables a and b are defined in a path relationship with a path coefficient parameter also named b. While such a name conflict between parameter and variable names would not confuse PROC CALIS in terms of model specification and fitting, it would create unnecessary confusion in programming and result interpretations. Hence, using parameter names that match variable names exactly is a bad practice and should be avoided.

Setting Constraints on Parameters

The CALIS procedure offers a very flexible way to constrain parameters. There are two main methods for specifying constraints. One is explicit specification by using specially designed statements for constraints. The other is implicit specification by using the SAS programming statements.

Explicit Specification of Constraints

Explicit constraints can be set in the following ways:

- specifying boundary constraints on independent parameters in the BOUNDS statement
- specifying general linear equality and inequality constraints on independent parameters in the LINCON statement
- specifying general nonlinear equality and inequality constraints on parametric functions in the NLINCON statement

BOUNDS Statement

You can specify one-sided or two-sided boundaries on independent parameters in the BOUNDS statement. For example, in the following statement you constrain parameter var1 to be nonnegative and parameter effect to be between 0 and 5.

```
bounds   var1 >= 0,
         0. <= effect <= 5.;
```

Note that if your upper and lower bounds are the same for a parameter, it effectively sets a fixed value for that parameter. As a result, PROC CALIS will reduce the number of independent parameters by one automatically. Note also that only independent parameters are allowed to be bounded in the BOUNDS statement.

LINCON Statement

You can specify equality or inequality linear constraints on independent parameters in the LINCON statement. For example, in the following statement you specify a linear inequality constraint on parameters beta1, beta2, and beta3 and an equality constraint on parameters gamma1 and gamma2.

```
lincon   beta1 - .5 * beta2 - .5 * beta3  >= 0.,
         gamma1 - gamma2 = 0.;
```

In the inequality constraint, beta1 is set to be at least as large as the average of beta2 and beta3. In the equality constraint, gamma1 and gamma2 are set to be equal. Note that in PROC CALIS a nonredundant linear equality constraint on independent parameters effectively reduces the number of parameters by one.

NLINCON Statement

You can specify equality or inequality nonlinear constraints for parameters in the NLINCON statement. While you can only constrain the independent parameters in the BOUNDS and the LINCON statements, you can constrain any of the following in the NLINCON statement:

- independent parameters
- dependent parameters
- parametric functions computed by the SAS programming statements

For example, consider the following statements:

```
nlincon
    IndParm >= 0,           /* constraint 1 */
    0 <= DepParm <= 10,     /* constraint 2 */
    ParmFunc1 >= 3,         /* constraint 3 */
    0 <=  ParmFunc2 <= 8;   /* constraint 4 */

/* SAS Programming statements in the following */
DepParm   = Indparm1 + IndParm5;
ParmFunc1 = IndParm1 - .5 * IndParm2 - .5 * IndParm3;
ParmFunc2 = (IndParm1 - 7.)**2 + SQRT(DepParm * IndParm4) * ParmFunc1;
```

You specify four nonlinear constraints by using the NLINCON statement. Labeled in a comment as “constraint 1” is a one-sided boundary constraint for independent parameter IndParm. Labeled in a comment as “constraint 2” is a two-sided boundary constraint for dependent parameter DepParm. Labeled in a

comment as "constraint 3" is a one-sided inequality constraint on parametric function named ParmFunc1. Finally, labeled in a comment as "constraint 4" is a two-sided inequality constraint on parametric function named ParmFunc2. Parametric functions ParmFunc1 and ParmFunc2 are defined and computed in the SAS programming statements after the NLINCON statement specification.

Constraint 1 could have been set in the BOUNDS statement because it is just a simple boundary constraint on an independent parameter. Constraint 3 could have been set in the LINCON statement because the definition of ParmFunc1 in a SAS programming statement shows that it is a linear function of independent parameters. The purpose of including these special cases of "nonlinear constraints" in this example is to show the flexibility of the NLINCON statement. However, whenever possible, the BOUNDS or the LINCON statement specification should be considered first because computationally they are more efficient than the equivalent specification in the NLINCON statement.

Specification in the NLINCON statement becomes necessary when you want to constrain dependent parameters or nonlinear parametric functions. For example, constraint 2 is a two-sided boundary constraint on the dependent parameter DepParm, which is defined as a linear function of independent parameters in a SAS programming statement. Constraints on dependent parameters are not allowed in the BOUNDS statement. Constraint 4 is a two-sided inequality constraint on the nonlinear parametric function ParmFunc2, which is defined as a nonlinear function of other parametric functions and parameters in the SAS programming statements. Again, you cannot use the LINCON statement to specify nonlinear constraints.

Implicit Constraint Specification

An implicit way to specify constraints is to use your own SAS programming statements together with the PARAMETERS statement to express special properties of the parameter estimates. This kind of reparameterization tool is also present in McDonald's COSAN implementation (McDonald 1978) but is considerably easier to use in the CALIS procedure. PROC CALIS is able to compute analytic first- and second-order derivatives that you would have to specify using the COSAN program.

Some traditional ways to enforce parameter constraints by using reparameterization or parameter transformation (McDonald 1980) are considered in the following:

- **one-sided boundary constraints of the form:**

  $$q \geq a \qquad \text{or} \qquad q \leq b$$

  where the parameter of interest is q, which should be at least as large as (or at most as small as) a given constant value a (or b). This inequality constraint can be expressed as an equality constraint:

  $$q = a + x^2 \qquad \text{or} \qquad q = b - x^2$$

  in which the new parameter x is unconstrained.

  For example, inequality constraint $q \geq 7$ can be accomplished by the following statements:

  ```
  parameters  x (0.);
  q = 7 + x * x;
  ```

  In this specification, you essentially redefine q as a parametric function of x, which is not constrained and has a starting value at 0.

- **two-sided boundary constraints of the form:**

  $$a \leq q \leq b$$

  where the parameter of interest is q, which should be located between two given constant values a and b, with $a < b$. This inequality constraint can be expressed as the following equality constraint:

  $$q = a + (b - a)\frac{\exp(x)}{1 + \exp(x)}$$

  where the new parameter x is unconstrained.

  For example, to implement $1 \leq q \leq 5$ in PROC CALIS, you can use the following statements:

  ```
  parameters x (0.);
  u = exp(x);
  q = 1 + 4 * u / (1 + u);
  ```

  In this specification, q becomes a dependent parameter which is nonlinearly related to independent parameter x, which is an independent parameter defined in the PARAMETERS statement with a starting value of 0.

- **one-sided order constraints of the form:**

  $$q_1 \leq q_2, \quad q_1 \leq q_3, \quad \ldots, \quad q_1 \leq q_k$$

  where $q_1, \ldots, q_k$ are the parameters of interest. These inequality constraints can be expressed as the following set of equality constraints:

  $$q_1 = x_1, \quad q_2 = x_1 + x_2^2, \quad \ldots, \quad q_k = x_1 + x_k^2$$

  where the new parameters $x_1, \ldots, x_k$ are unconstrained.

  For example, to implement $q_1 \leq q_2$, $q_1 \leq q_3$, and $q_1 \leq q_4$ simultaneously, you can use the following statements:

  ```
  parameters x1-x4 (4*0.);
  q1 = x1;
  q2 = x1 + x2 * x2;
  q3 = x1 + x3 * x3;
  q4 = x1 + x4 * x4;
  ```

  In this specification, you essentially redefine $q_1 - q_4$ as dependent parameters that are functions of $x_1 - x_4$, which are defined as independent parameters in the PARAMETERS statement with starting values of zeros. No constraints on x_i's are needed. The way that q_i's are computed in the SAS programming statements guarantees the required order constraints on q_i's are satisfied.

- **two-sided order constraints of the form:**

  $$q_1 \leq q_2 \leq q_3 \leq \cdots \leq q_k$$

  These inequality constraints can be expressed as the following set of equality constraints:

  $$q_1 = x_1, \quad q_2 = q_1 + x_2^2, \quad \ldots \quad q_k = q_{k-1} + x_k^2$$

  where the new parameters $x_1, \ldots, x_k$ are unconstrained.

  For example, to implement $q_1 \leq q_2 \leq q_3 \leq q_4$ simultaneously, you can use the following statements:

```
parameters x1-x4 (4*0.);
q1 = x1;
q2 = q1 + x2 * x2;
q3 = q2 + x3 * x3;
q4 = q3 + x4 * x4;
```

In this specification, you redefine $q_1 - q_4$ as dependent parameters that are functions of $x_1 - x_4$, which are defined as independent parameters in the PARAMETERS statement. Each x_i has a starting value of zero without being constrained in estimation. The order relation of q_i's are satisfied by the way they are computed in the SAS programming statements.

- **linear equation constraints of the form:**

$$\sum_{i}^{k} b_i q_i = a$$

where q_i's are the parameters of interest, b_i's are constant coefficients, a is a constant, and k is an integer greater than one. This linear equation can be expressed as the following system of equations with unconstrained new parameters $x_1, x_2, \ldots, x_k$:

$$\begin{aligned} q_i &= x_i / b_i \qquad (i < k) \\ q_k &= (a - \sum_{j}^{k-1} x_j)/b_k \end{aligned}$$

For example, consider the following linear constraint on independent parameters $q_1 - q_3$:

$$3q_1 + 2q_2 - 5q_3 = 8$$

You can use the following statements to implement the linear constraint:

```
parameters x1-x2 (2*0.);
q1 = x1 / 3;
q2 = x2 / 2;
q3 = -(8 - x1 - x2) / 5;
```

In this specification, $q_1 - q_3$ become dependent parameters that are functions of *x1* and *x2*. The linear constraint on q_1 and q_3 are satisfied by the way they are computed. In addition, after reparameterization the number of independent parameters drops to two.

See McDonald (1980) and Browne (1982) for further notes on reparameterization techniques.

Explicit or Implicit Specification of Constraints?

Explicit and implicit constraint techniques differ in their specifications and lead to different computational steps in optimizing a solution. The explicit constraint specification that uses the supported statements incurs additional computational routines within the optimization steps. In contrast, the implicit reparameterization method does not incur additional routines for evaluating constraints during the optimization. Rather, it changes the constrained problem to a non-constrained one. This costs more in computing function derivatives and in storing parameters.

If the optimization problem is small enough to apply the Levenberg-Marquardt or Newton-Raphson algorithm, use the BOUNDS and the LINCON statements to set explicit boundary and linear constraints. If the problem is so large that you must use a quasi-Newton or conjugate gradient algorithm, reparameterization techniques might be more efficient than the BOUNDS and LINCON statements.

Automatic Variable Selection

When you specify your model, you use the main and subsidiary model statements to define variable relationships and parameters. PROC CALIS checks the variables mentioned in these statements against the variable list of the input data set. If a variable in your model is also found in your data set, PROC CALIS knows that it is a manifest variable. Otherwise, it is either a latent variable or an invalid variable.

To save computational resources, only manifest variables defined in your model are automatically selected for analysis. For example, even if you have 100 variables in your input data set, only a covariance matrix of 10 manifest variables is computed for the analysis of the model if only 10 variables are selected for analysis.

In some special circumstances, the automatic variable selection performed for the analysis might be a drawback. For example, if you are interested in modification indices connected to some of the variables that are not used in the model, automatic variable selection in the specification stage will exclude those variables from consideration in computing modification indices. Fortunately, a little trick can be done. You can use the VAR statement to include as many exogenous manifest variables as needed. Any variables in the VAR statement that are defined in the input data set but are not used in the main and subsidiary model specification statements are included in the model as exogenous manifest variables.

For example, the first three steps in a stepwise regression analysis of the Werner Blood Chemistry data (Jöreskog and Sörbom 1988, p. 111) can be performed as follows:

```
proc calis data=dixon method=gls nobs=180 print mod;
   var    x1-x7;
   lineqs y = e;
   variance    e = var;
run;
proc calis data=dixon method=gls nobs=180 print mod;
   var    x1-x7;
   lineqs y = g1 x1 + e;
   variance    e = var;
run;
proc calis data=dixon method=gls nobs=180 print mod;
   var    x1-x7;
   lineqs y = g1 x1 + g6 x6 + e;
   variance    e = var;
run;
```

In the first analysis, no independent manifest variables are included in the regression equation for dependent variable y. However, x1–x7 are specified in the VAR statement so that in computing the Lagrange multiplier tests these variables would be treated as potential predictors in the regression equation for dependent variable y. Similarly, in the next analysis, x1 is already a predictor in the regression equation, while x2–x7 are treated as potential predictors in the LM tests. In the last analysis, x1 and x6 are predictors in the regression equation, while other x-variables are treated as potential predictors in the LM tests.

Path Diagrams: Layout Algorithms, Default Settings, and Customization

Path diagrams provide visually appealing representations of the interrelationships among variables in structural equation models. However, it is no easy task to determine the "best" layout of the variables in a path diagram. The CALIS procedure provide three algorithms for laying out good-quality path diagrams. Understanding the underlying principles of these algorithms helps you select the right algorithm and fine-tune the features of a path diagram.

The section "The Process-Flow, Grouped-Flow, and GRIP Layout Algorithms" on page 1784 describes the principles of the algorithms. The sections "Handling Destroyers in Path Diagrams" on page 1792 shows how you can improve the quality of path diagrams. The section "Default Path Diagram Settings" on page 1796 details the default settings for path diagram elements and the corresponding options. By using these options, you can customize path diagrams to meet your own requirements. Finally, the sections "Expanding the Definition of the Structural Variables" on page 1802 and "Showing or Emphasizing the Structural Components" on page 1798 show examples of customizations that you can make to the structural components of models.

The Process-Flow, Grouped-Flow, and GRIP Layout Algorithms

The most important factor in choosing an appropriate algorithm is the nature of the interrelationships among variables in the model. Depending on the pattern of interrelationships, the CALIS procedure provides three basic algorithms: the process-flow algorithm, grouped-flow algorithm, and GRIP algorithm. You can request these algorithms by specifying the ARRANGE= option in the PATHDIAGRAM statement. This section describes these algorithms and the situations in which they work well. The section also describes how the CALIS procedure selects the "best" algorithm by default (by specifying the ARRANGE=AUTOMATIC option).

The Process-Flow Algorithm

The process-flow algorithm (which you select by specifying ARRANGE=FLOW) works well when the interrelationships among the variables in the model (not including the error variables) follow an ideal process-flow pattern. In such a pattern, each variable can be placed at a unique level so that all paths between variables exist only between adjacent levels and are aligned in the same direction. Figure 33.3 shows an example of a linear ordering of observed variables based on the directions of the paths. This model clearly exhibits an ideal process-flow pattern: variables can be uniquely assigned to five levels, and the paths can occur only between adjacent levels and in the same direction.

Figure 33.3 Linear Ordering of Observed Variables

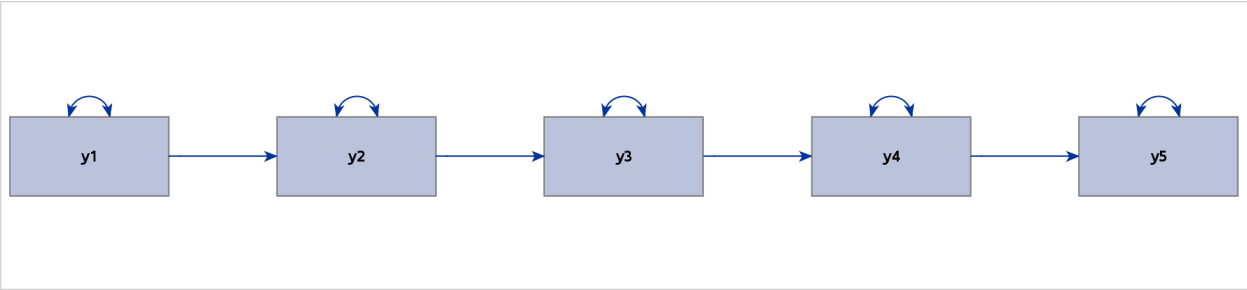

Figure 33.4 shows a hierarchical ordering of the observed variables. The pattern of interrelationships of variables in this model is much like an organizational chart. Clearly, this path diagram also depicts an ideal process-flow pattern.

Figure 33.4 Hierarchical Ordering of Observed Variables

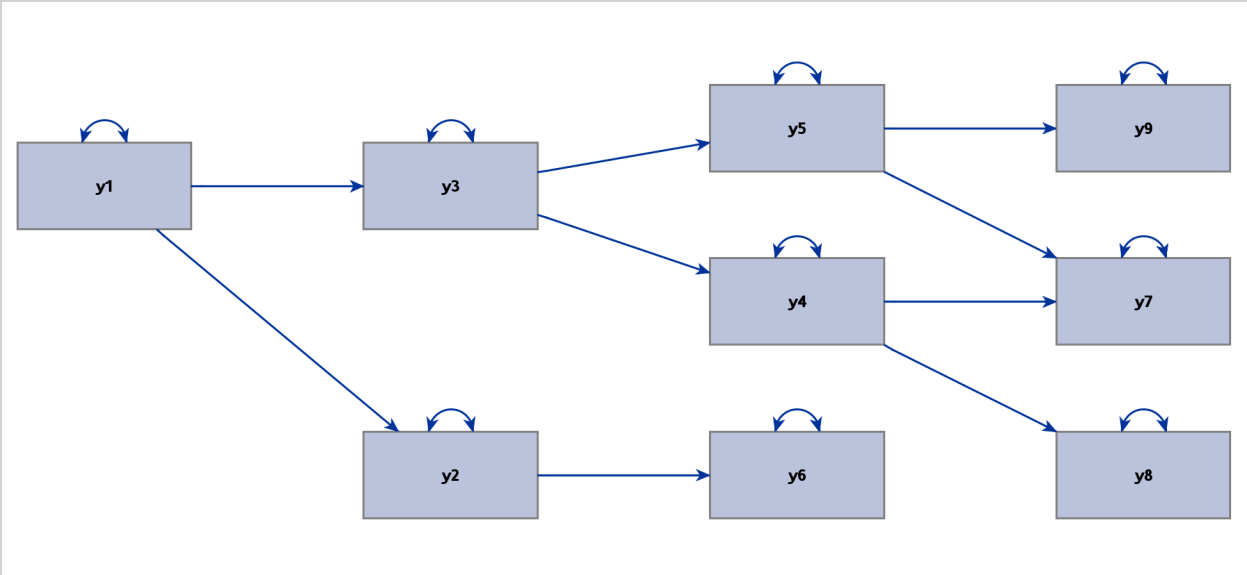

Latent variables are often used in structural equation modeling. Some classes of latent variable models have ideal process-flow patterns. For example, for confirmatory factor models in their "pure" form, the process-flow algorithm can be used to show the hierarchical relationships of factors and variables. Figure 33.5 shows an example of a confirmatory factor model.

Figure 33.5 Confirmatory Factor Model That Exhibits an Ideal Process-Flow Pattern

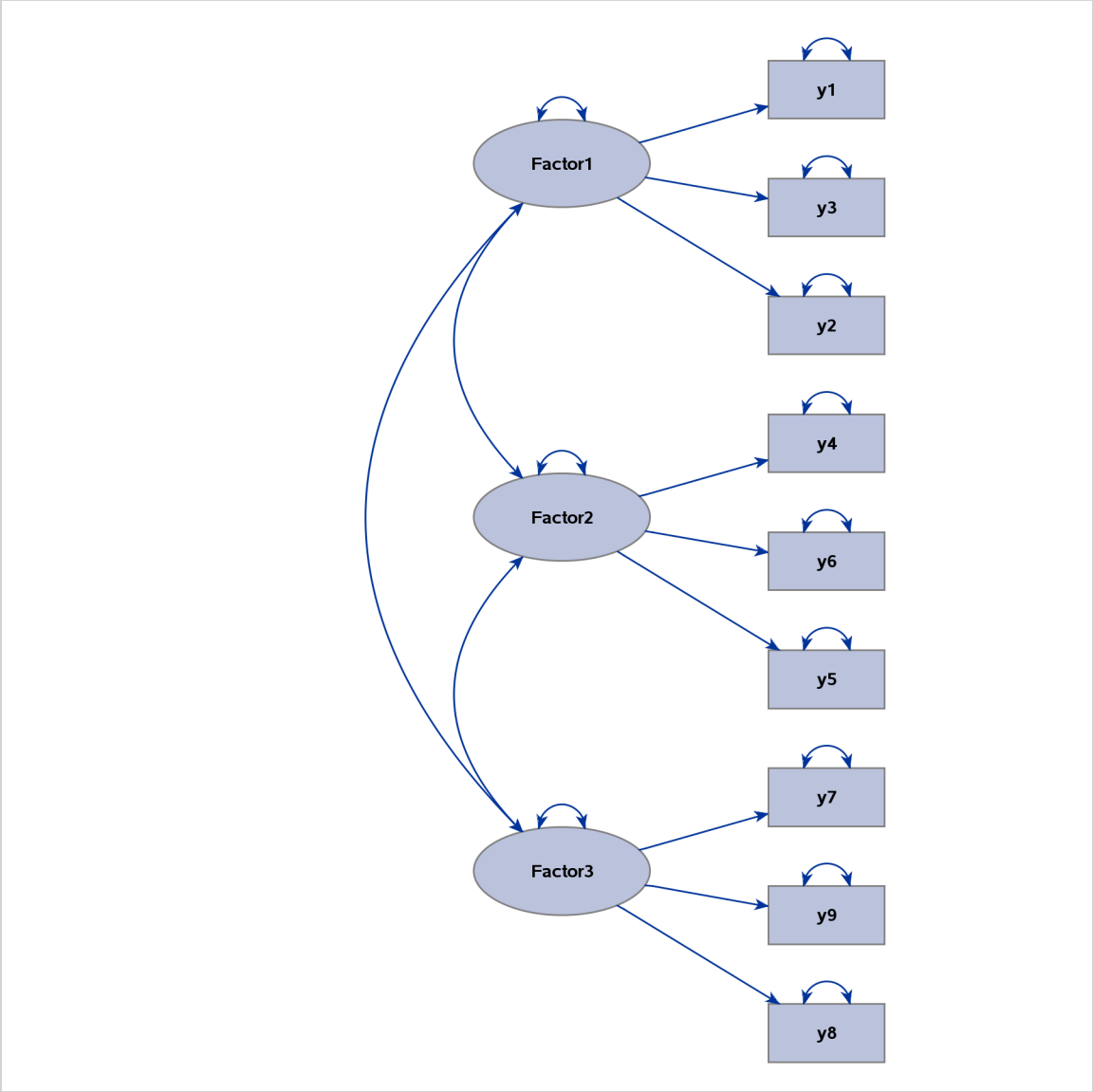

Other examples are higher-order factor models and hierarchical factor models, as shown in Figure 33.6 and Figure 33.7, respectively.

Figure 33.6 Linearly Ordered Factors with an Ideal Process-Flow Pattern

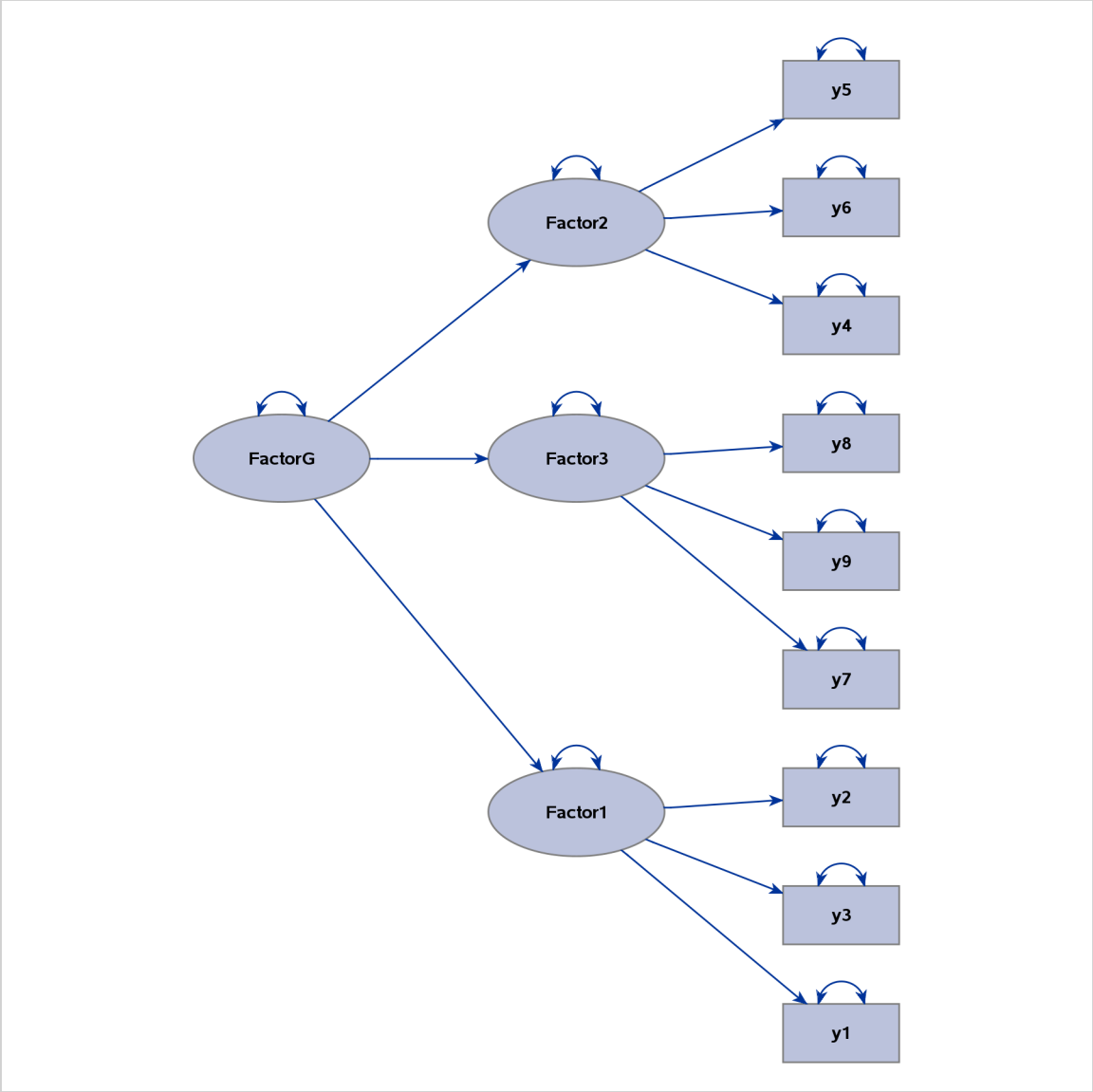

Figure 33.7 Hierarchical Factor Model That Exhibits an Ideal Process-Flow Pattern

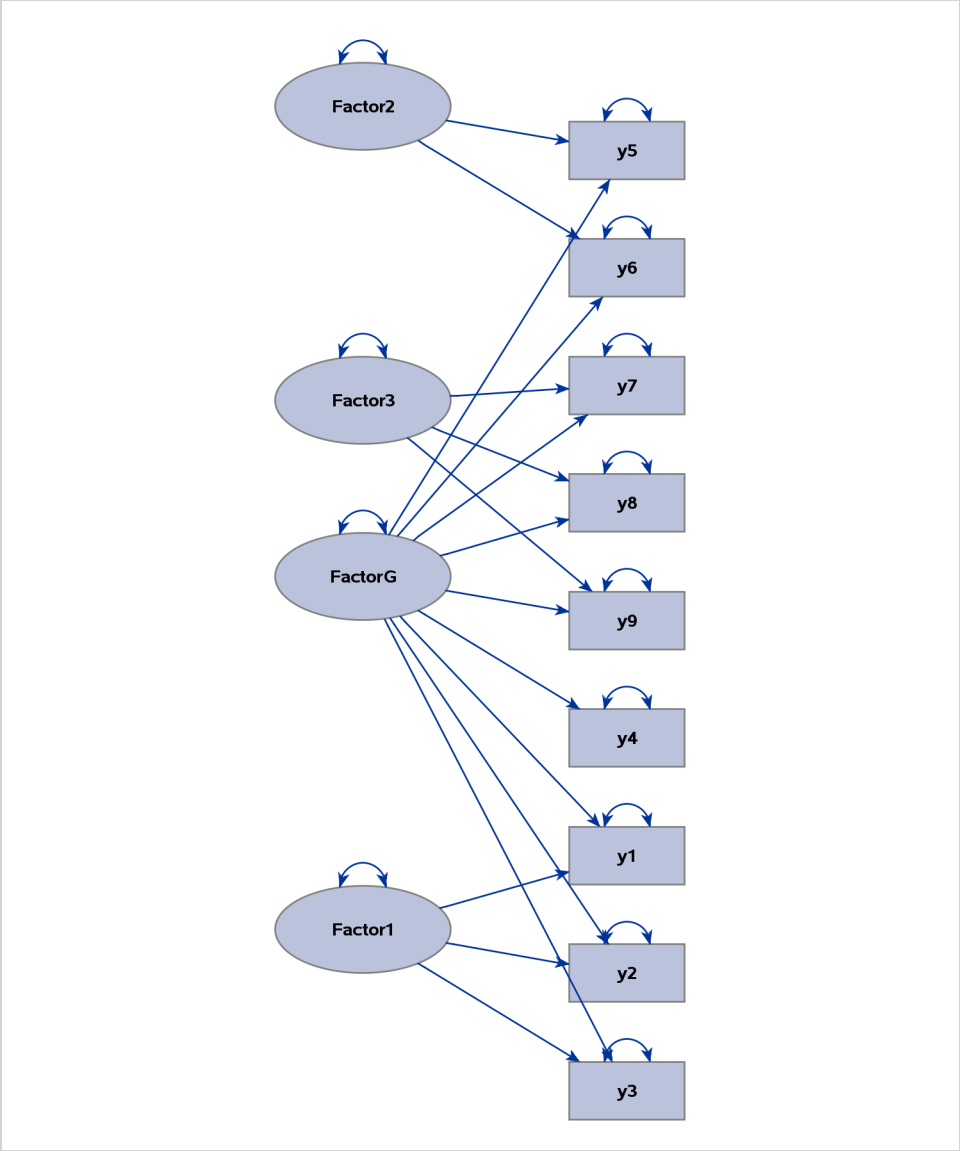

The Grouped-Flow Algorithm

The grouped-flow algorithm works well when the interrelationships of the latent factors follow an ideal process-flow pattern. Such a pattern requires that each latent factor in the model be placed at a unique level so that all paths between the latent factors exist only between adjacent levels and in the same direction. This pattern is called an ideal *grouped*-flow pattern because each latent factor in the model is associated with a cluster of observed variables. In other words, each cluster of observed variables and a latent factor forms a group. You can specify the ARRANGE=GROUPEDFLOW option in the PATHDIAGRAM statement to request the grouped-flow algorithm.

Figure 33.8 illustrates a linear ordering of latent factors, and Figure 33.9 illustrates a hierarchical ordering of latent factors. Both path diagrams illustrate ideal grouped-flow patterns.

Figure 33.8 Linearly Ordered Factors with an Ideal Grouped-Flow Pattern

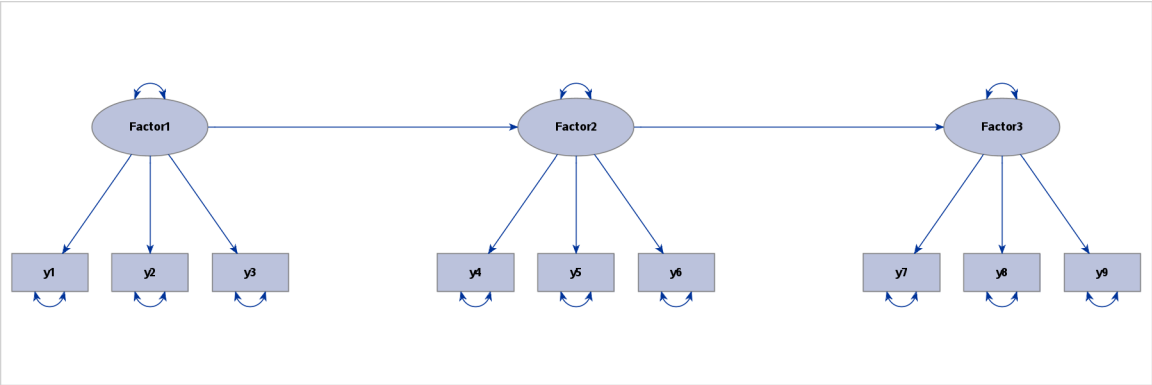

Figure 33.9 Hierarchically Ordered Factors with an Ideal Grouped-Flow Pattern

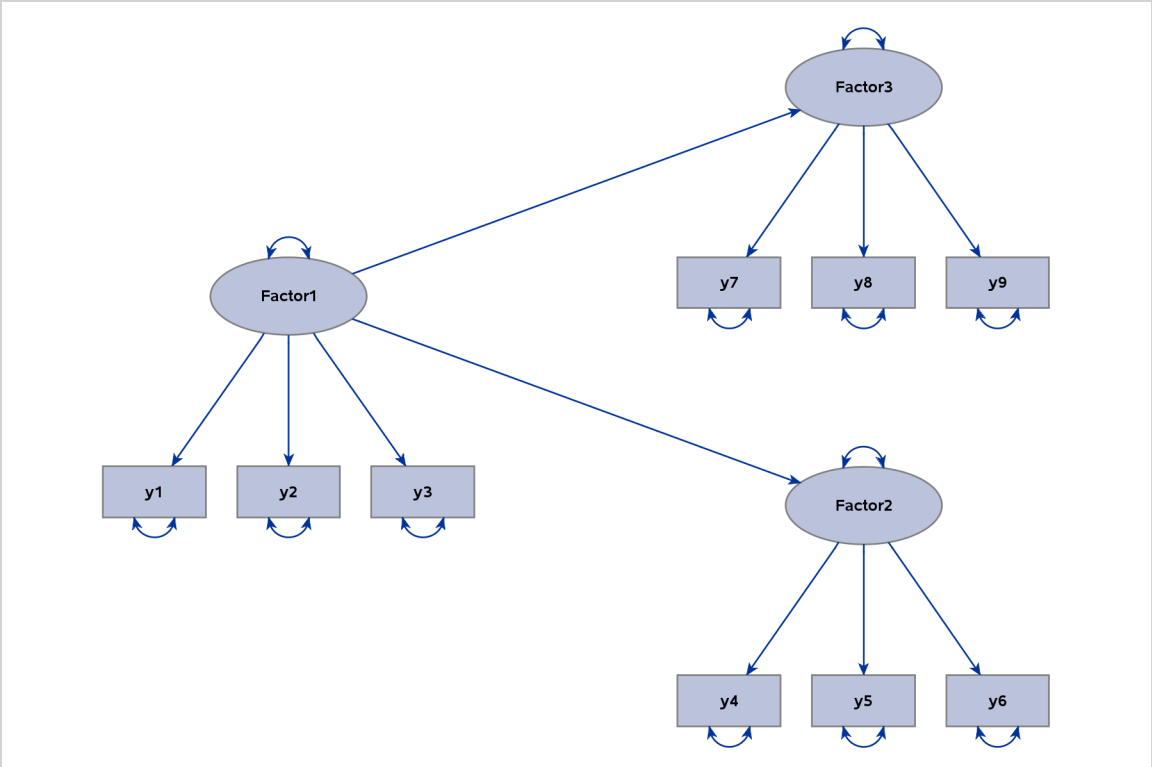

A model that exhibits an ideal grouped-flow pattern for factors can also exhibit an ideal process-flow pattern for non-error variables. The question is then which algorithm is better and under what situations. For example, you can characterize the interrelationships among the variables that are shown in Figure 33.8 by an ideal process-flow pattern of all non-error variables in the model. Figure 33.10 is the diagram that results if you use the ARRANGE=FLOW option for this model.

Figure 33.10 Applying the Process-Flow Algorithm to an Ideal Grouped-Flow Pattern

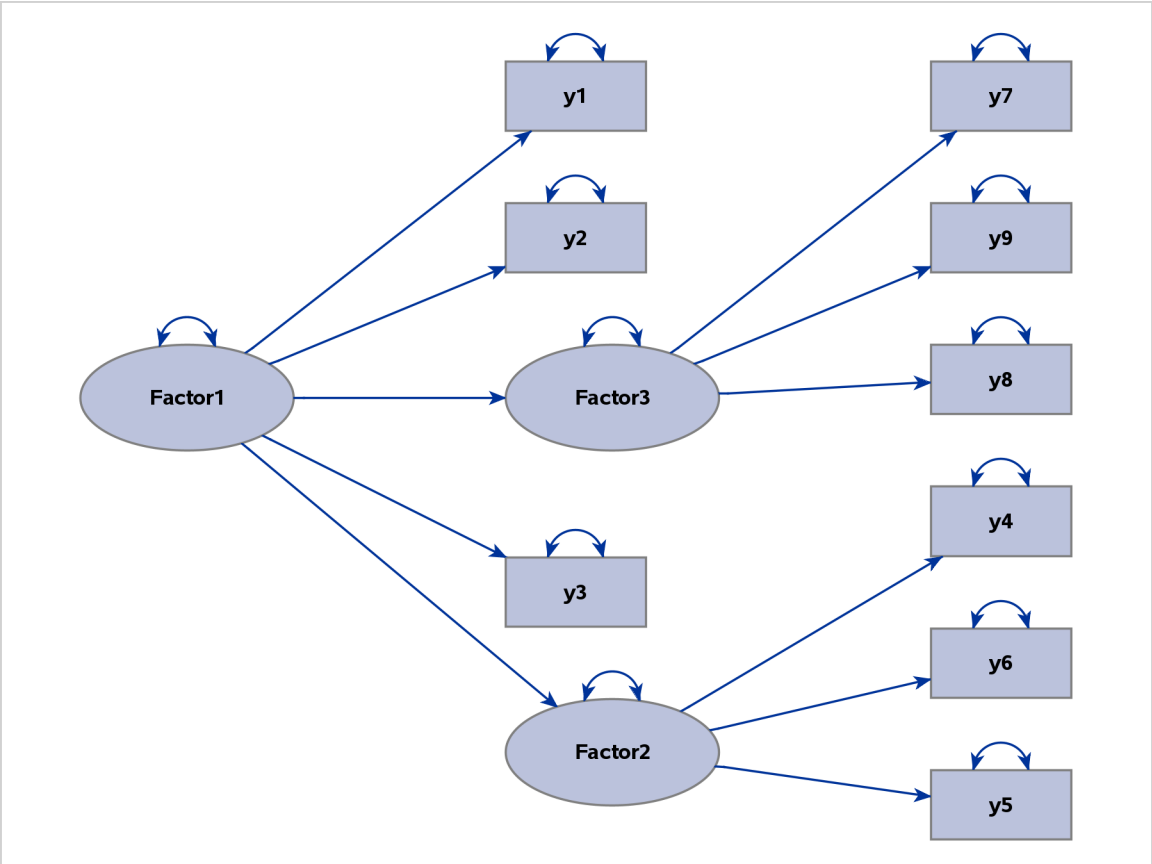

Unfortunately, the latent variables and observed variables in Figure 33.10 are now mixed within the two middle levels, burying the important clusters of observed variables and latent factors that are clearly shown in the grouped-flow representation in Figure 33.8. Hence, the grouped-flow algorithm is more preferable here. For this reason, PROC CALIS adds an extra condition for applying the process-flow algorithm when the default automatic layout method is specified. That is, in order for the process-flow algorithm to be used, variables within each level must also be of the same type (observed or latent), and an ideal process-flow pattern is required in the non-error variables.

The GRIP Algorithm

The GRIP (Graph dRrawing with Intelligent Placement) algorithm is a general layout method for graphs that display nodes and links (or variables and paths in the context of path diagrams). Unlike the process-flow and grouped-flow algorithms, the GRIP algorithm is not designed to display specific patterns of variable relationships. Rather, the GRIP algorithm uses a graph-theoretic approach to determine an intelligent initial placement of the nodes (variables). Then the algorithm goes through refinement stages in rendering the final path diagram. Essentially, the GRIP algorithm provides a general layout algorithm when variables or factors in the model cannot be ordered uniquely according to the path directions. The algorithm balances the placement of the variables and groups related variables together.

PROC CALIS uses a modified version of the GRIP algorithm that is more suitable for structural equation models. Basically, error variables are ignored in the layout process (but are displayed in path diagrams if requested), and the lengths of the paths are adjusted according to the types of variables that are being

connected by the paths. These modifications ensure that clusters of observed variables and factors are distinct from each other, as illustrated in Figure 33.11.

Figure 33.11 Path Diagram That Uses the GRIP Layout Algorithm

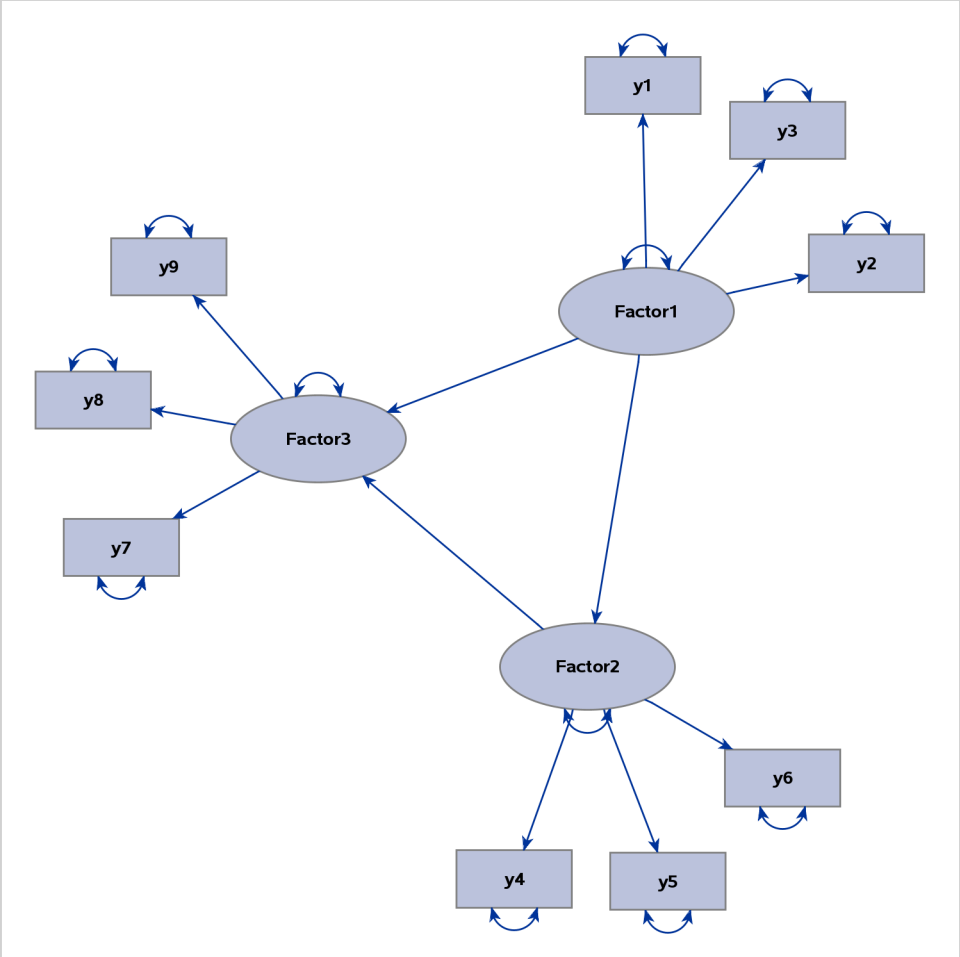

In Figure 33.11, Factor1 is the only exogenous variable in the model. This means that no directional path points to Factor1, which is considered to be a level-1 variable. At the next level are the variables that have directional paths from Factor1. Therefore, y1, y2, y3, Factor2, and Factor3 are considered to be at level 2. However, because Factor2 has a directional path to Factor3, Factor3 can also be considered to be at level 3. Hence, this model does not show either an ideal process-flow pattern or an ideal grouped-flow pattern. However, the more general GRIP algorithm can still show three recognizable clusters of variables.

What does the ARRANGE=AUTOMATIC option actually do?

By default, the CALIS procedure uses an automatic method to determine the layout algorithm for path diagrams. This automatic method is equivalent to specifying the ARRANGE=AUTOMATIC option. Actually, the AUTOMATIC option is not associated with a specific layout algorithm. What it does is determine automatically which algorithm—the process-flow, grouped-flow, or GRIP algorithm—is the best for a given model.

The following steps describe how PROC CALIS automatically determines the best layout algorithm:

1. PROC CALIS checks whether the interrelationships among all non-error variables in the model exhibit an ideal process-flow pattern. In this ideal pattern, all non-error variables in the model can be uniquely ordered according to the directional paths. PROC CALIS then checks whether variables at each level are of the same type (observed variables or latent factors). If they are, PROC CALIS uses the process-flow algorithm as if the ARRANGE=FLOW option were specified. Otherwise, it continues to the next step.

2. PROC CALIS checks whether the interrelationships among all factors in the model exhibit an ideal process-flow pattern. In this ideal pattern, all latent factors in the model can be uniquely ordered according to the directional paths. If they are, PROC CALIS uses the grouped-flow algorithm as if the ARRANGE=GROUPEDFLOW option were specified. Otherwise, it continues to the next step.

3. PROC CALIS uses the GRIP algorithm as if the ARRANGE=GRIP option were specified.

In summary, when the default ARRANGE=AUTOMATIC option is used, PROC CALIS chooses a suitable layout algorithm by examining the interrelationships of variables. The procedure detects ideal process-flow and grouped-flow patterns when they exist in models.

Handling Destroyers in Path Diagrams

No layout algorithm for path diagrams works perfectly in all situations. Chances are that some paths in your model would violate the ideal pattern that each of the basic layout algorithms assumes. In some cases, even if the violations are minor—for example, they occur in just one or two paths—they are sufficient to throw off the layout of the diagram.

There is a way to alleviate the problems that are caused by these minor violations. If you can proactively identify the paths in your model that violate the ideal pattern that the intended layout algorithm assumes, you can attempt to ignore these minor violations during the layout process so that the layout algorithm can work well for the majority of patterns in the path diagram. To accomplish this, the CALIS procedure provides the DESTROYER= option for you to specify these minor violations. This section presents examples to illustrate the use of this option.

To illustrate the use of the DESTROYER= option, consider a higher-order factor model that is specified by the following statements:

```
proc calis;
   path
      FactorG ===> Factor1-Factor3,
      Factor1 ===> y1-y3  ,
      Factor2 ===> y4-y6  ,
      Factor3 ===> y7-y9  ,
      FactorG ===> y2 y7  ;
run;
```

In this model, Factor1, Factor2, Factor3, and FactorG are all latent factors. All other variables are observed variables. Which would be the best layout algorithm to draw the path diagram for this model?

Does this model exhibit an ideal process-flow pattern? In this model, FactorG is the only exogenous non-error variable and is considered to be at level 1. Factor1, Factor2, Factor3, y2, and y7 all have directional paths from FactorG, and therefore they are considered to be at level 2. However, because y2 and y7 also have directional paths from Factor1 and Factor3, respectively, they can also be considered to be at level 3. Hence, this model does not exhibit an ideal process-flow pattern.

Does this model exhibit an ideal grouped-flow pattern? If you consider only the factors in the model, FactorG is at level 1 and all other latent factors are at level 2 without ambiguities. Hence, this model exhibits an ideal process-flow pattern in factors or an ideal grouped-flow pattern in all non-error variables. You expect the grouped-flow algorithm to be used to draw the path diagram for this model if you specify the following statements:

```
proc calis;
   path
      FactorG ===> Factor1-Factor3,
      Factor1 ===> y1-y3  ,
      Factor2 ===> y4-y6  ,
      Factor3 ===> y7-y9  ,
      FactorG ===> y2 y7  = destroyer1 destroyer2;
   pathdiagram diagram=initial notitle;
run;
```

Figure 33.12 A Higher-Order Factor Model With Two Destroyer Paths

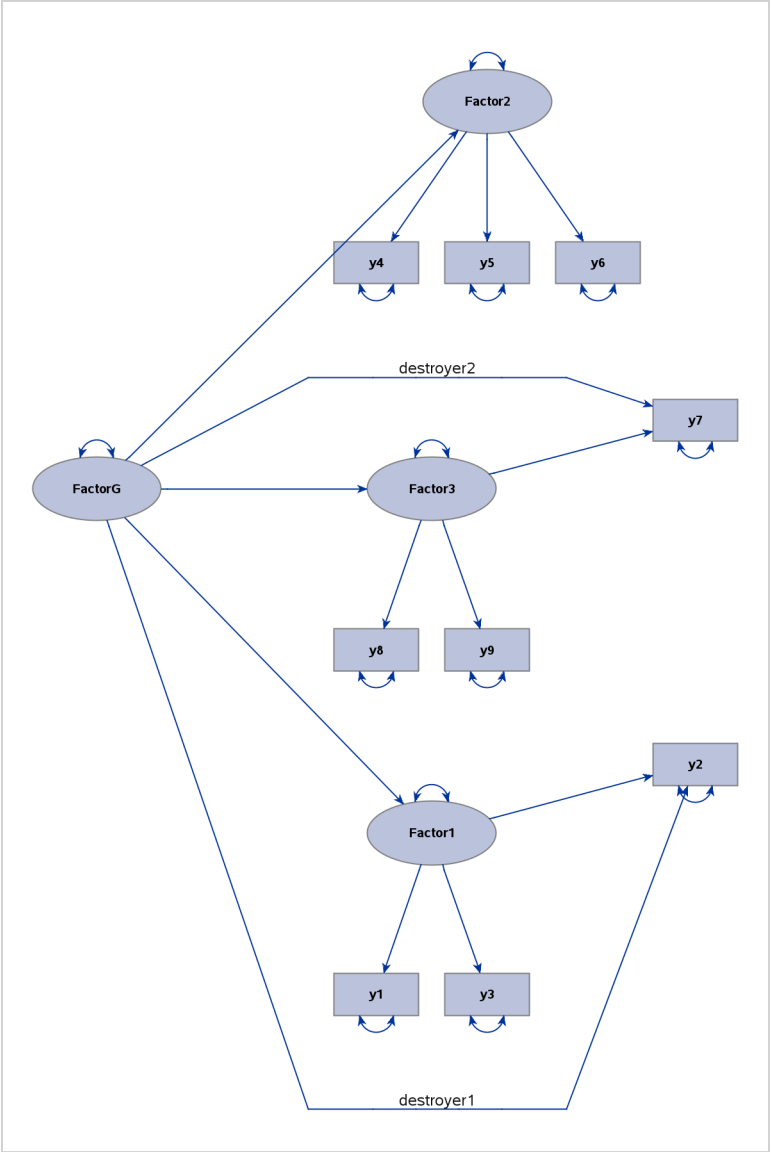

Figure 33.12 shows the output diagram. Although the path diagram still shows the hierarchical ordering of the latent factors, the two paths, "FactorG ===> y2" and "FactorG ===> y7," weaken the display of the factor-variable clusters for Factor1 and Factor3. Therefore, these two paths can be called destroyer paths, or simply destroyers. For this reason, the effect parameters of these two paths are labeled as "destroyer1" and "destroyer2" in the PATH statement to show their disruptive characteristics.

What if you ignore these destroyer paths in the layout process of the diagram? To do this, you can use the DESTROYER= option in the PATHDIAGRAM statement, as follows:

```
pathdiagram diagram=initial notitle destroyer=[FactorG ===> y2 y7];
```

Because the two destroyer paths are ignored when PROC CALIS determines the layout algorithm, an ideal process-flow pattern is recognized, and the process-flow algorithm is used to lay out all variables in the model. Then, all paths, including the destroyer paths, are simply put back into the picture to produce the final path diagram, as shown in Figure 33.13.

Figure 33.13 Higher-Order Factor Model with Two Destroyer Paths Identified

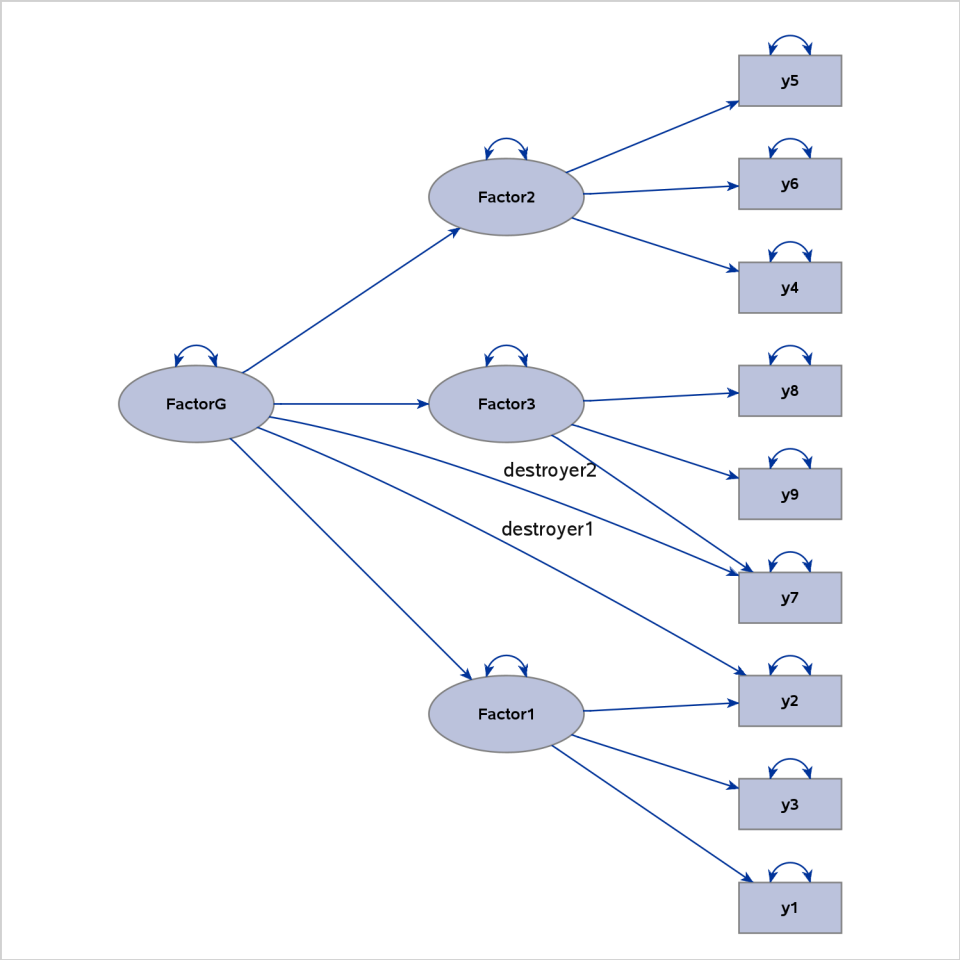

The diagram in Figure 33.13 is preferable to the one in Figure 33.12 because it clearly shows the three levels of variables in the higher-order factor model.

Destroyer paths do not necessarily occur only in ideal process-flow or grouped-flow patterns. Even when

the GRIP algorithm is used for a general pattern of variable interrelationships, you might be able to identify potential destroyers in the path diagram. Consider the model that is specified by the following statements:

```
proc calis;
   path
      Factor1 ===> y1 y2,
      Factor2 ===> y3 y4,
      Factor3 ===> y5 y6,
      Factor1 ===> Factor2 Factor3,
      Factor2 ===> Factor3,
      y2      ===> y5    = destroyer;
   pathdiagram diagram=initial notitle;
run;
```

This model exhibits neither an ideal process-flow pattern nor an ideal grouped-flow pattern. As a result, PROC CALIS uses the GRIP algorithm automatically to draw the path diagram. Figure 33.14 shows the output diagram.

Figure 33.14 Path Diagram Whose Destroyer Path Is Not Handled by the GRIP Algorithm

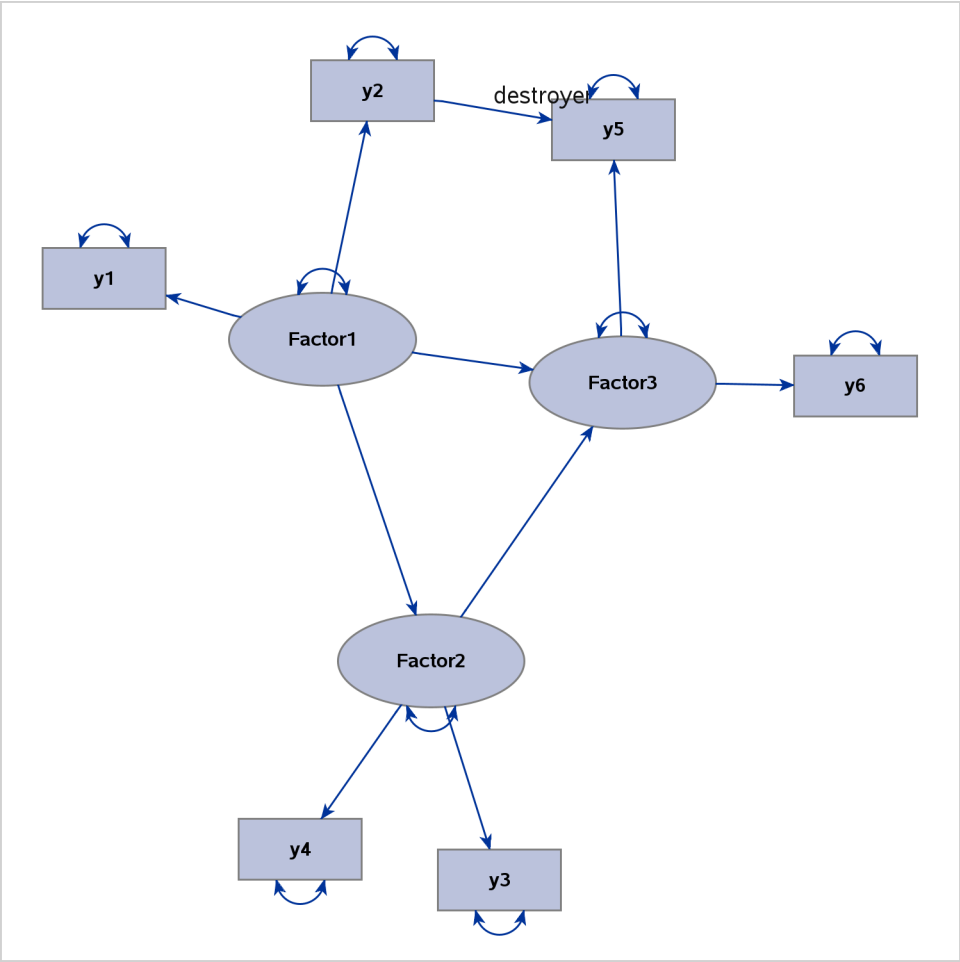

Because of the presence of the destroyer path "y2 ===> y5," the two factor clusters for Factor1 and Factor3 are drawn closer to each other in the path diagram, thus destroying their distinctive identities that are

associated with the related measured variables.

You can use the DESTROYER= option to fix this problem, as shown in the following modified PATHDIAGRAM statement:

```
pathdiagram diagram=initial notitle destroyer=[y2 ===> y5];
```

Figure 33.15 Path Diagram Whose Destroyer Path Is Handled by the GRIP Algorithm

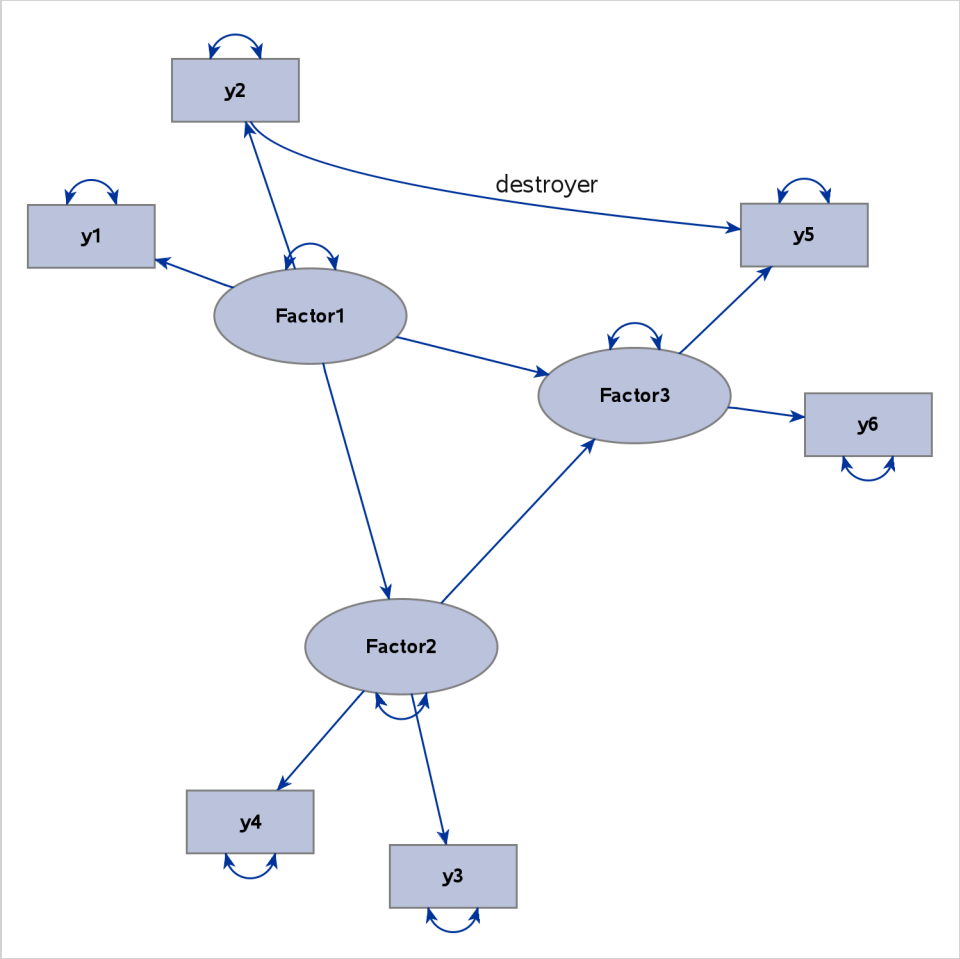

Figure 33.15 now shows a more preferable path diagram. The factor clusters in this diagram are more distinctive than those in Figure 33.14, in which the algorithm does not handle the destroyer path.

Default Path Diagram Settings

Taking the conventions and clarity of graphical representations into consideration, the CALIS procedure employs a set of default settings for producing path diagrams. These default settings define the style or scheme for representing the path diagram elements graphically. For example, PROC CALIS displays the error variances as double-headed paths that are attached to the associated variables. In this way, there is no need to display the error variables so that more space is available to present other important graphical elements in path diagrams.

However, researchers do not always agree on the best style or scheme for representing the path diagram

elements. Even the same researcher might want to use different representation style or scheme for his or her path diagrams in different situations. In this regard, you can override most of the default settings to customize path diagrams to meet your own requirements. Table 33.5 summarizes the default graphical and nongraphical settings of the path diagram elements. Related options are listed next to these default settings.

Table 33.5 Default Path Diagram Settings and the Overriding or Modifying Options

Graphical Element or Property	Default Settings	Options
General Graphical Properties		
Layout method	Automatically determined	ARRANGE=
Models with path diagram output	All models	MODEL=
Path diagrams for structural models	Not shown	STRUCTADD= STRUCTURAL
Paths that affect the layout	All paths	DESTROYER= OMITPATH=
Solution type of path diagrams	Unstandardized solution	DIAGRAM=
Structural components	Not emphasized	EMPHSTRUCT STRUCTADD=
Paths		
Covariances between error variables	Shown as double-headed paths	NOCOV NOERRCOV
Covariances between non-error and error variables	Shown as double-headed paths	NOCOV
Covariances between non-error variables	Shown as double-headed paths in models specified using the FACTOR statement; not shown in other types of models	EXOGCOV NOCOV NOEXOGVAR
Directional paths	Shown as single-headed paths	OMITPATH=
Error variances	Shown as double-headed paths in covariance models; shown as labels in mean and covariance models	NOERRVAR NOVARIANCE VARPARM=
Means and intercepts	Not shown as paths; shown as labels in mean and covariance models	MEANPARM= NOMEAN
Variances of non-error variables	Shown as double-headed paths in covariance models; shown as labels in mean and covariance models	NOVARIANCE VARPARM=
Variables		
Error variables	Not shown	USEERROR
Relative sizes	Observed variables:Factors:Errors = 1:1.5:0.5	ERRORSIZE= FACTORSIZE=
Variable labels	Original variable names used	LABEL=

Table 33.5 *continued*

Graphical Element or Property	Default Settings	Options
Parameter Estimates		
Covariances between error variables	Shown in all types of models	NOCOV NOERRCOV
Covariances between non-error variables	Shown in models specified using the FACTOR statement; not shown in other types of models	EXOGCOV NOCOV NOEXOGCOV
Initial parameter specifications	Shown with fixed values and user-specified parameter names	NOINITPARM
Means and intercepts	Shown in models with mean structures	NOMEAN
Variances of error variables	Shown in all types of models	NOERRVAR NOVARIANCE
Variances of non-error variables	Shown in all types of models	NOEXOGVAR NOVARIANCE
Formats of Parameter Estimates		
Decimal places	Two decimal places used	DECP=
Numerical values	Shown in unstandardized and standardized solutions	NOESTIM
Parameter names	Not shown	PARMNAMES
Significance flags	Shown in unstandardized and standardized solutions	NOFLAG
Graph Title and Fit Summary		
Decimal places in fit summary	Two decimal places used	DECPFIT=
Fit table	Shown in unstandardized and standardized solutions	NOFITTABLE
Fit table contents	A default set of available information and statistics	FITINDEX=
Title	Default title that reflects the identity and solution type of the model	NOTITLE TITLE=

Showing or Emphasizing the Structural Components

One often-used customization in structural equation modeling is to display only the structural component (or structural model) of the full model, which consists of the measurement and the structural components. Traditionally, the structural component refers to the latent factors and their interrelationships, whereas the measurement component refers to observed indicator variables and their relationships with the latent factors.

For example, consider the full model that is displayed as a path diagram in Figure 33.16. The latent factors are Factor1, Factor2, and Factor3. The remaining variables, y1–y9, are observed variables.

Figure 33.16 Path Diagram for the Complete Model

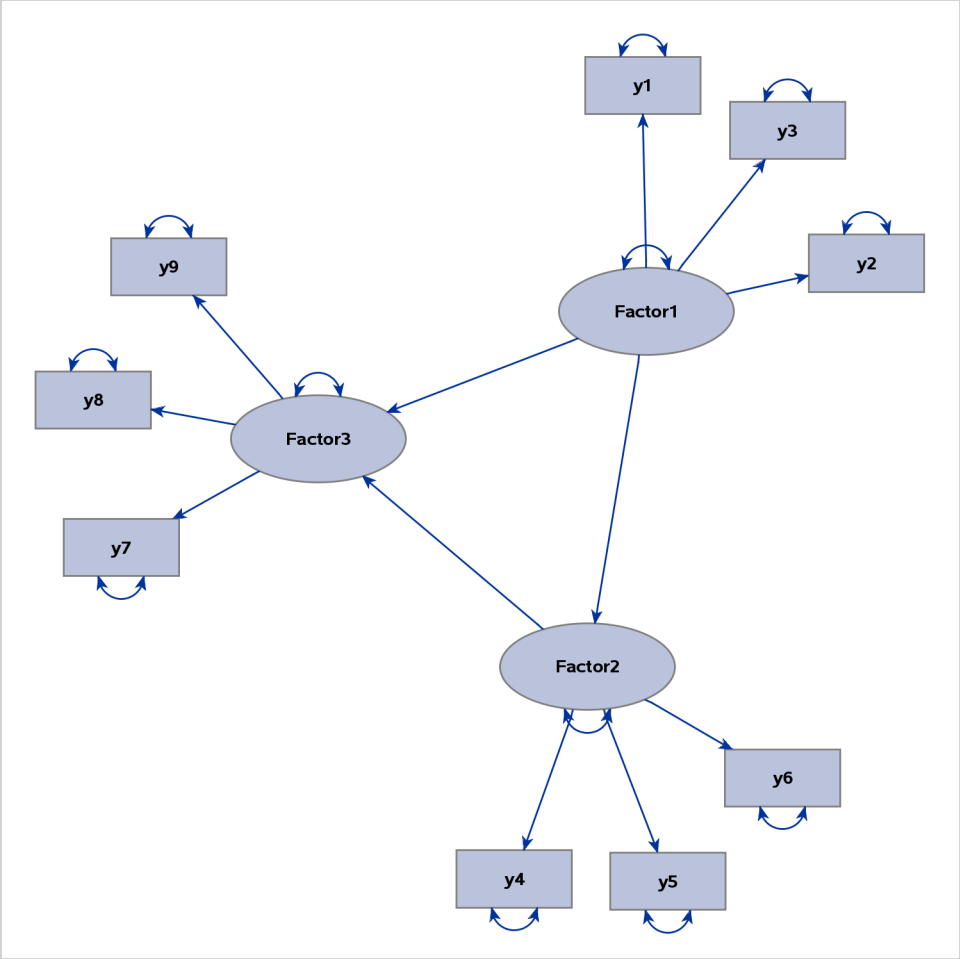

The structural component of this full model includes the three latent factors, their directional relationships (represented by the three directed paths), and their variance parameters (represented by the three double-headed arrows that are attached to the factors).

Showing only the structural component is useful when your research is focused mainly on the interrelationships among the latent factors. The measurement component, which usually contains a relatively large number of observed variables for reflecting or defining the latent factors, is of secondary interest. Eliminating the measurement component leads to a clearer path diagram to represent your model.

To request a path diagram that shows the structural component, you can specify the STRUCTURAL option in the PATHDIAGRAM statement, as in the following statements:

```
proc calis;
   path
      Factor1 ===> y1-y3 ,
      Factor2 ===> y4-y6 ,
      Factor3 ===> y7-y9 ,
      Factor1 ===> Factor2 Factor3,
      Factor2 ===> Factor3;
   pathdiagram diagram=initial structural;
```

```
run;
```

Figure 33.17 Path Diagram for the Structural Component

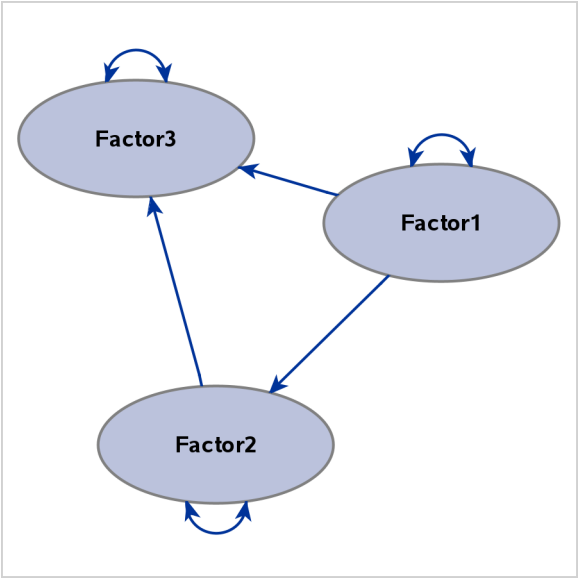

Figure 33.17 shows the path diagram for the structural model. For simplicity in presentation, the DIAGRAM=INITIAL option is used to show the initial model specifications only.

The preceding specification produces the path diagram for the structural component, in addition to that for the full model. To output the path diagram for the structural component only, use the STRUCTURAL(ONLY) option in the PATHDIAGRAM statement, as follows:

```
pathdiagram diagram=initial structural(only);
```

Instead of eliminating the measurement component from the path diagram, PROC CALIS provides the EMPHSTRUCT option to highlight the structural component in the full model. For example, the following PATHDIAGRAM statement produces the path diagram in Figure 33.17:

```
pathdiagram diagram=initial emphstruct;
```

Figure 33.18 Emphasizing the Structural Component in the Path Diagram

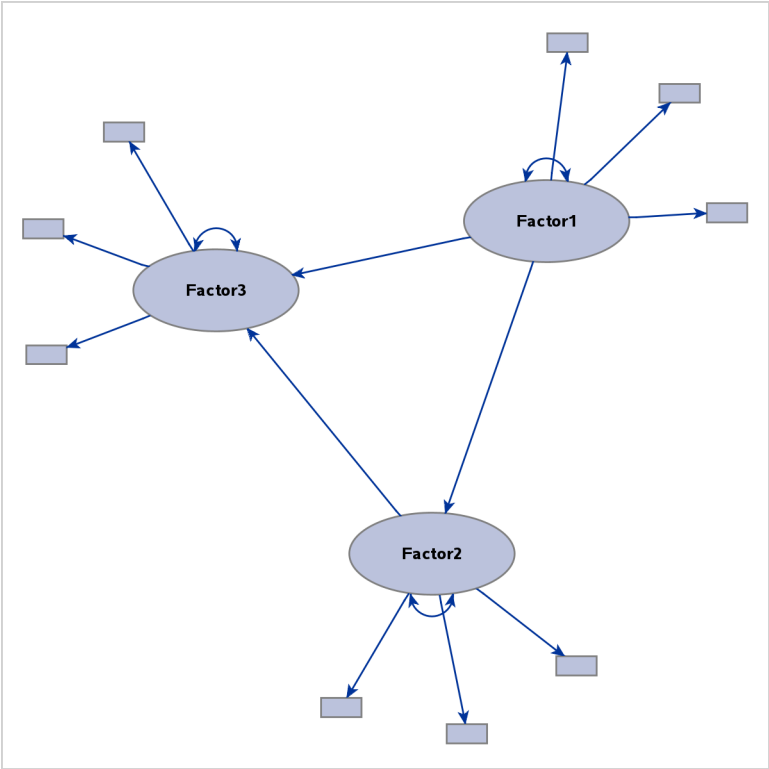

Figure 33.18 shows a complete model, with emphasis on the structural component. The path diagram labels only the latent factors. Observed variables are represented by small rectangles and are not labeled.

Certainly, the EMPHSTRUCT option is not limited to the initial path diagram, such as the one shown in Figure 33.18. For the unstandardized and standardized solutions, the EMPHSTRUCT option puts similar emphasis on the structural variables. In addition, the diagram displays only the parameter estimates of the structural components. Figure 33.19 shows the path diagram of an unstandardized solution that emphasizes the structural components.

Figure 33.19 Emphasizing the Structural Component in the Path Diagram for the Unstandardized Solution

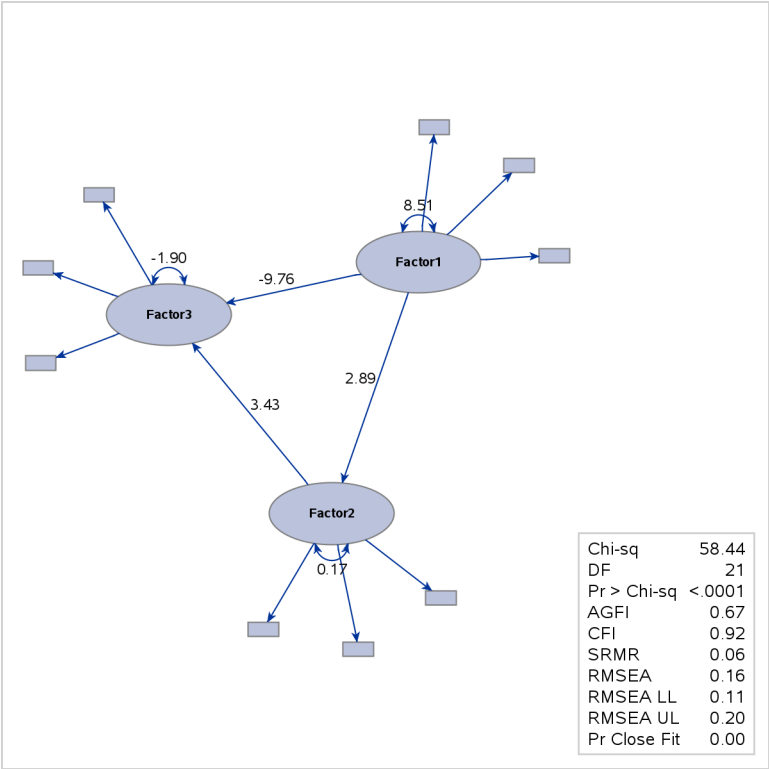

Expanding the Definition of the Structural Variables

The traditional definition of "structural model" or "structural component" might be too restrictive, in the sense that only latent factors and their interrelationships are included. In contemporary structural equation modeling, sometimes observed variables can take the role of "factors" in the structural component. For example, observed variables that have negligible measurement errors and serve as exogenous or predictor variables in a system of functional equations naturally belong to the structural component. For this reason or other reasons, researchers might want to expand the definition of "structural component" to include some designated observed variables. The STRUCTADD= option enables you to do that.

For example, the following statements specify a model that has two latent factors, Factor1 and Factor2:

```
proc calis;
   path
      Factor1 ===> y1-y3,
      Factor2 ===> y4-y6,
      y7      ===> Factor1 Factor2;
   pathdiagram diagram=initial notitle structadd=[y7]
               label=[y7="y7 is Exogenous"] ;
   pathdiagram diagram=initial notitle structadd=[y7] struct(only);
               label=[y7="y7 is Exogenous"] ;
   pathdiagram diagram=initial notitle structadd=[y7] emphstruct;
               label=[y7="y7 is Exogenous"] ;
run;
```

By default, only the latent factors Factor1 and Factor2 are treated as structural variables. However, because the observed variable y7 is exogenous and is a predictor of the two latent variables, it is not unreasonable to also treat it as a structural variable when you are producing path diagrams. Hence, all three PATHDIAGRAM statements use the STRUCTADD= option to include y7 in the set of structural variables. For illustration purposes, variable y7 is labeled as "y7 is Exogenous" in the path diagrams. Although all three PATHDIAGRAM statements produce diagrams for the initial solution, they each display the structural component in a different way.

The first PATHDIAGRAM statement produces a diagram for the full model in Figure 33.20. The structural component is not specifically emphasized. Because y7 is treated as a structural variable, it is larger than the other observed variables, and its size is comparable to that of the default structural variables Factor1 and Factor2.

Figure 33.20 Including an Observed Variable as a Structural Variable: Full Model

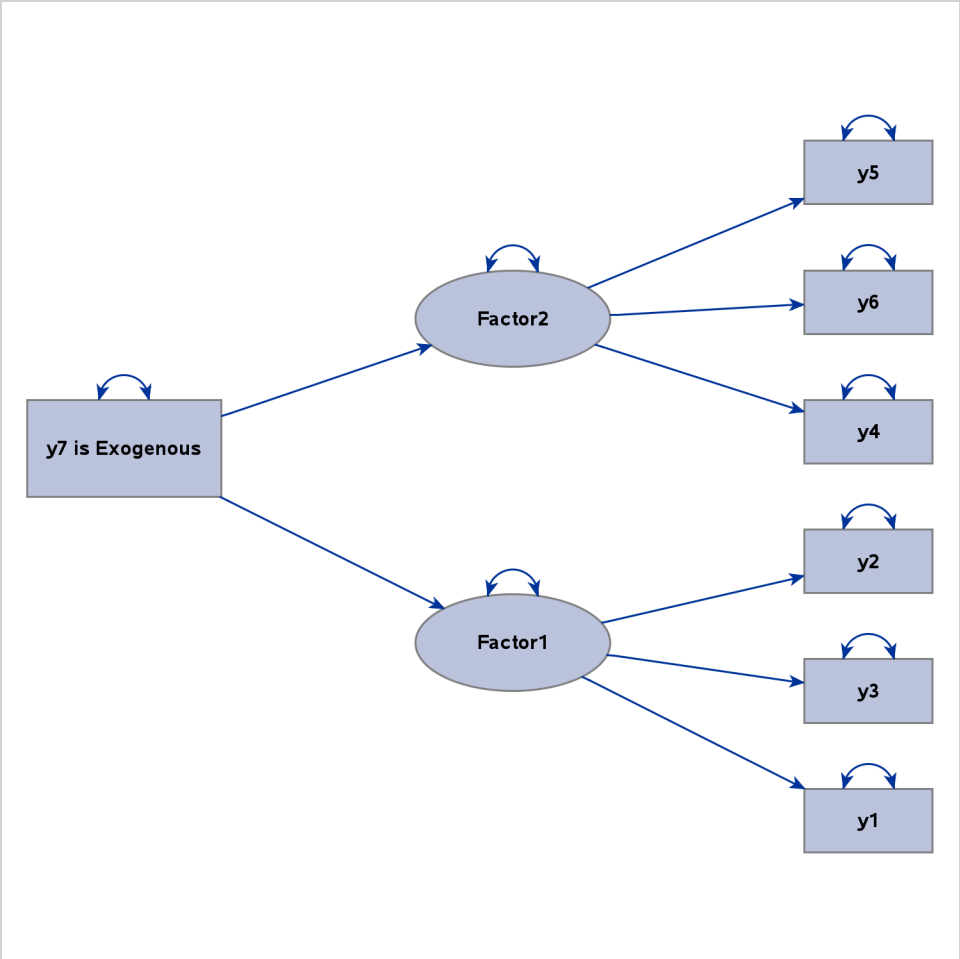

The second PATHDIAGRAM statement produces a diagram that emphasizes the structural component of the full model in Figure 33.21. Again, because y7 is treated as a structural variable, it is much larger than the other observed variables, and it is emphasized in the same way as the default structural variables Factor1 and Factor2.

Figure 33.21 Including an Observed Variable as a Structural Variable: Emphasizing the Structural Component

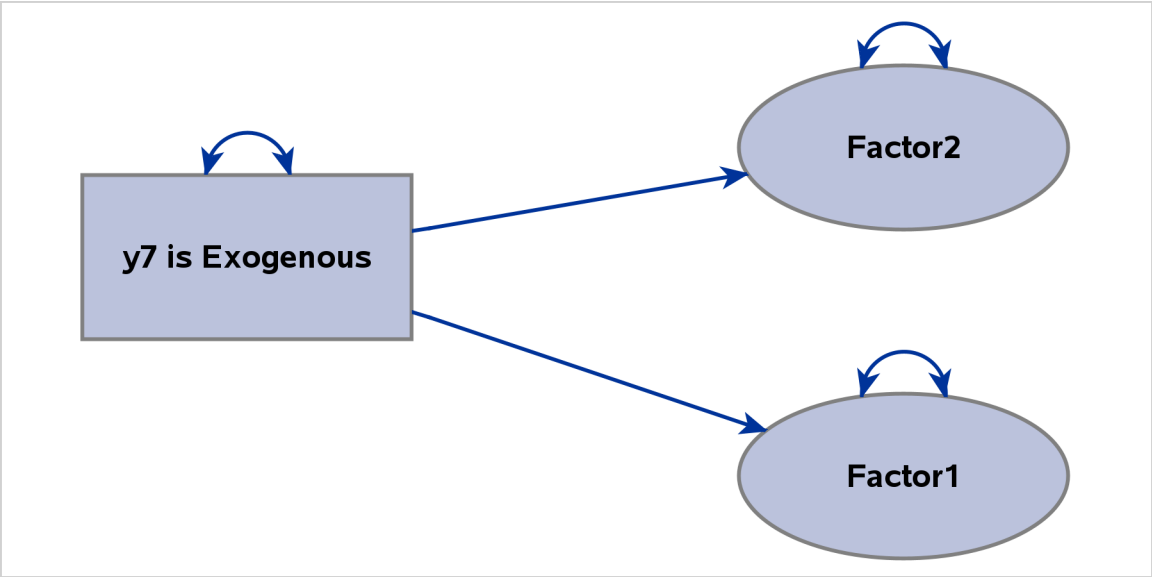

The third PATHDIAGRAM statement produces a diagram for the structural component in Figure 33.22. Because y7 is treated as a structural variable, it is also shown in this path diagram, and its size is comparable to that of Factor1 and Factor2.

Figure 33.22 Including an Observed Variable as a Structural Variable: Structural Component

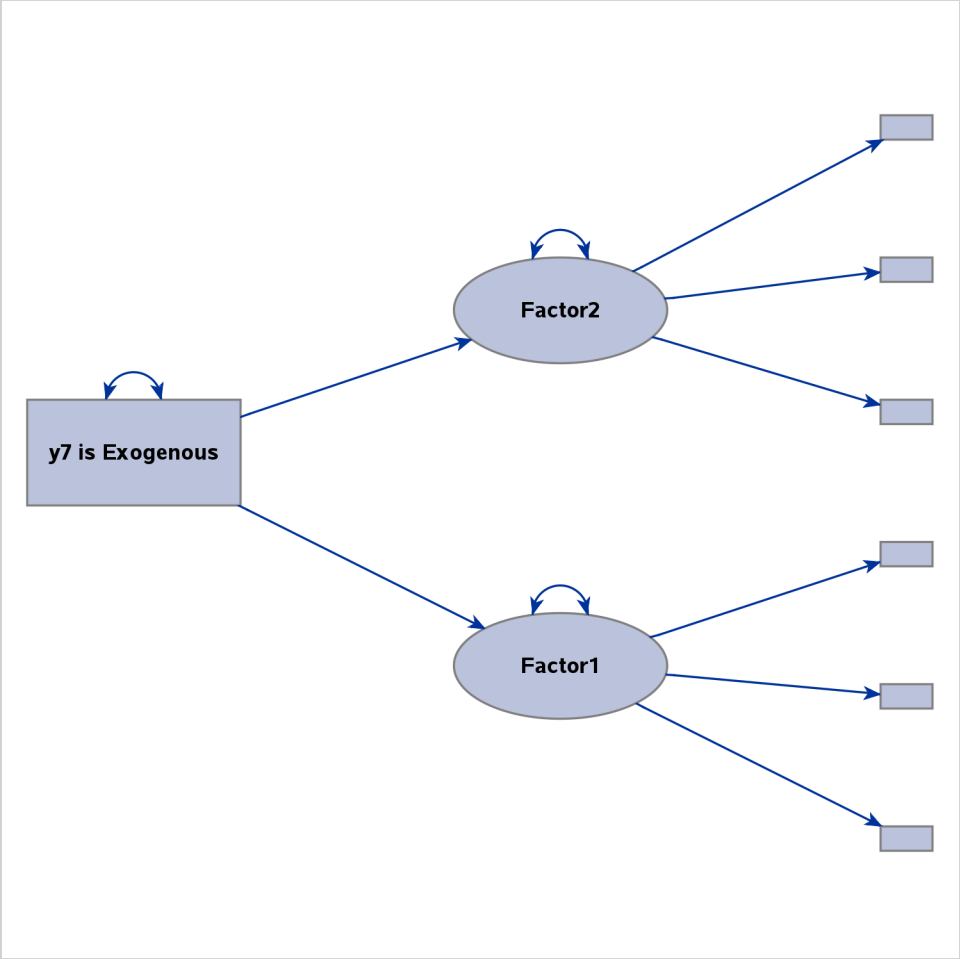

If you do not include y7 as a structural variable in this PATHDIAGRAM statement, Figure 33.22 would display Factor1 and Factor2 as isolated variables that do not have any directional paths that connect them, making it an uninteresting path diagram display.

Estimation Criteria

The following eight estimation methods are available in PROC CALIS:

- unweighted least squares (ULS)
- full information maximum likelihood (FIML)
- generalized least squares (GLS)
- normal-theory maximum likelihood (ML)
- normal-theory maximum likelihood with Satorra-Bentler adjustments (MLSB)

- weighted least squares (WLS, ADF)
- diagonally weighted least squares (DWLS)
- robust estimation (ROBUST) under the normal theory (ML)

Default weight matrices $\mathbf{W}$ are computed for GLS, WLS, and DWLS estimation. You can also provide your own weight matrices by using an INWGT= data set. The weight matrices in these estimation methods provide weights for the moment matrices. In contrast, weights that are applied to individual observations are computed in robust estimation. These observation weights are updated during iteration steps of robust estimation. The ULS, GLS, ML, WLS, ADF, and DWLS methods can analyze sample moment matrices as well as raw data, while the FIML and robust methods must analyze raw data.

PROC CALIS does not implement all estimation methods in the field. As mentioned in the section "Overview: CALIS Procedure" on page 1506, partial least squares (PLS) is not implemented. The PLS method is developed under less restrictive statistical assumptions. It circumvents some computational and theoretical problems encountered by the existing estimation methods in PROC CALIS; however, PLS estimates are less efficient in general. When the statistical assumptions of PROC CALIS are tenable (for example, large sample size, correct distributional assumptions, and so on), ML, GLS, or WLS methods yield better estimates than the PLS method. Note that there is a SAS/STAT procedure called PROC PLS that employs the partial least squares technique, but for a different class of models than those of PROC CALIS. For example, in a PROC CALIS model each latent variable is typically associated with only a subset of manifest variables (predictor or outcome variables). However, in PROC PLS latent variables are not prescribed with subsets of manifest variables. Rather, they are extracted from linear combinations of all manifest predictor variables. Therefore, for general path analysis with latent variables you should use PROC CALIS.

ULS, GLS, ML, and MLSB Discrepancy Functions

In each estimation method, the parameter vector is estimated iteratively by a nonlinear optimization algorithm that minimizes a discrepancy function F, which is also known as the fit function in the literature. With p denoting the number of manifest variables, $\mathbf{S}$ the sample $p \times p$ covariance matrix for a sample with size N, $\bar{\mathbf{x}}$ the $p \times 1$ vector of sample means, $\boldsymbol{\Sigma}$ the fitted covariance matrix, and $\boldsymbol{\mu}$ the vector of fitted means, the discrepancy function for unweighted least squares (ULS) estimation is:

$$F_{ULS} = 0.5\mathrm{Tr}[(\mathbf{S} - \boldsymbol{\Sigma})^2] + (\bar{\mathbf{x}} - \boldsymbol{\mu})'(\bar{\mathbf{x}} - \boldsymbol{\mu})$$

The discrepancy function for generalized least squares estimation (GLS) is:

$$F_{GLS} = 0.5\mathrm{Tr}[(\mathbf{W}^{-1}(\mathbf{S} - \boldsymbol{\Sigma}))^2] + (\bar{\mathbf{x}} - \boldsymbol{\mu})'\mathbf{W}^{-1}(\bar{\mathbf{x}} - \boldsymbol{\mu})$$

By default, $\mathbf{W} = \mathbf{S}$ is assumed so that F_{GLS} is the normal theory generalized least squares discrepancy function.

The discrepancy function for normal-theory maximum likelihood estimation (ML) is:

$$F_{ML} = \mathrm{Tr}(\mathbf{S}\boldsymbol{\Sigma}^{-1}) - p + \ln(|\boldsymbol{\Sigma}|) - \ln(|\mathbf{S}|) + (\bar{\mathbf{x}} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\bar{\mathbf{x}} - \boldsymbol{\mu})$$

The discrepancy function for MLSB is exactly the same as that for ML. Hence, METHOD=MLSB and METHOD=ML produce the same estimates. The critical difference is that METHOD=MLSB computes the Satorra-Bentler scaled chi-squares for the baseline and target models and uses these scaled chi-squares

to compute various fit indices. It also uses a sandwich formula (Satorra and Bentler 1994) to compute the standard error estimates (for more information, see the section "Satorra-Bentler Sandwich Formula for Standard Errors" on page 1819). The regular ML method makes neither of these adjustments.

In each of the discrepancy functions, $\mathbf{S}$ and $\bar{\mathbf{x}}$ are considered to be given and $\boldsymbol{\Sigma}$ and $\boldsymbol{\mu}$ are functions of model parameter vector $\boldsymbol{\Theta}$. That is:

$$F = F(\boldsymbol{\Sigma}(\boldsymbol{\Theta}), \boldsymbol{\mu}(\boldsymbol{\Theta}); \mathbf{S}, \bar{\mathbf{x}})$$

Estimating $\boldsymbol{\Theta}$ by using a particular estimation method amounts to choosing a vector $\boldsymbol{\theta}$ that minimizes the corresponding discrepancy function F.

When the mean structures are not modeled or when the mean model is saturated by parameters, the last term of each fit function vanishes. That is, they become:

$$F_{ULS} = 0.5\mathrm{Tr}[(\mathbf{S} - \boldsymbol{\Sigma})^2]$$

$$F_{GLS} = 0.5\mathrm{Tr}[(\mathbf{W}^{-1}(\mathbf{S} - \boldsymbol{\Sigma}))^2]$$

$$F_{ML} = \mathrm{Tr}(\mathbf{S}\boldsymbol{\Sigma}^{-1}) - p + \ln(|\boldsymbol{\Sigma}|) - \ln(|\mathbf{S}|)$$

Again, the MLSB fit function is exactly the same as that of ML.

If, instead of being a covariance matrix, $\mathbf{S}$ is a correlation matrix in the discrepancy functions, $\boldsymbol{\Sigma}$ would naturally be interpreted as the fitted correlation matrix. Although whether $\mathbf{S}$ is a covariance or correlation matrix makes no difference in minimizing the discrepancy functions, correlational analyses that use these functions are problematic because of the following issues:

- The diagonal of the fitted correlation matrix $\boldsymbol{\Sigma}$ might contain values other than ones, which violates the requirement of being a correlation matrix.
- Whenever available, standard errors computed for correlation analysis in PROC CALIS are straightforward generalizations of those of covariance analysis. In very limited cases these standard errors are good approximations. However, in general they are not even asymptotically correct.
- The model fit chi-square statistic for correlation analysis might not follow the theoretical distribution, thus making model fit testing difficult.

Despite these issues in correlation analysis, if your primary interest is to obtain the estimates in the correlation models, you might still find PROC CALIS results for correlation analysis useful.

The statistical techniques used in PROC CALIS are primarily developed for the analysis of covariance structures, and hence COV is the default option. Depending on the nature of your research, you can add the mean structures in the analysis by specifying mean and intercept parameters in your models. However, you cannot analyze mean structures simultaneously with correlation structures (see the CORR option) in PROC CALIS.

FIML Discrepancy Function

The full information maximum likelihood method (FIML) assumes multivariate normality of the data. Suppose that you analyze a model that contains p observed variables. The discrepancy function for FIML is

$$F_{FIML} = \frac{1}{N}\sum_{j=1}^{N}(\ln(|\boldsymbol{\Sigma}_j|) + (\mathbf{x}_j - \boldsymbol{\mu}_j)'\boldsymbol{\Sigma}_j^{-1}(\mathbf{x}_j - \boldsymbol{\mu}_j) + K_j)$$

where $\mathbf{x}_j$ is a data vector for observation j, and K_j is a constant term (to be defined explicitly later) independent of the model parameters $\boldsymbol{\Theta}$. In the current formulation, $\mathbf{x}_j$'s are not required to have the same dimensions. For example, $\mathbf{x}_1$ could be a complete vector with all p variables present while $\mathbf{x}_2$ is a $(p-1)\times 1$ vector with one missing value that has been excluded from the original $p \times 1$ data vector. As a consequence, subscript j is also used in $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ to denote the submatrices that are extracted from the entire $p \times 1$ structured mean vector $\boldsymbol{\mu}$ ($\boldsymbol{\mu} = \boldsymbol{\mu}(\boldsymbol{\Theta})$) and $p \times p$ covariance matrix $\boldsymbol{\Sigma}$ ($\boldsymbol{\Sigma} = \boldsymbol{\Sigma}(\boldsymbol{\Theta})$). In other words, in the current formulation $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ do not mean that each observation is fitted by distinct mean and covariance structures (although theoretically it is possible to formulate FIML in such a way). The notation simply signifies that the dimensions of $\mathbf{x}_j$ and of the associated mean and covariance structures could vary from observation to observation.

Let p_j be the number of variables without missing values for observation j. Then $\mathbf{x}_j$ denotes a $p_j \times 1$ data vector, $\boldsymbol{\mu}_j$ denotes a $p_j \times 1$ vector of means (structured with model parameters), $\boldsymbol{\Sigma}_j$ is a $p_j \times p_j$ matrix for variances and covariances (also structured with model parameters), and K_j is defined by the following formula, which is a constant term independent of model parameters:

$$K_j = \ln(2\pi) * p_j$$

As a general estimation method, the FIML method is based on the same statistical principle as the ordinary maximum likelihood (ML) method for multivariate normal data—that is, both methods maximize the normal theory likelihood function given the data. In fact, F_{FIML} used in PROC CALIS is related to the log-likelihood function L by the following formula:

$$F_{FIML} = \frac{-2L}{N}$$

Because the FIML method can deal with observations with various levels of information available, it is primarily developed as an estimation method that could deal with data with random missing values. See the section "Relationships among Estimation Criteria" on page 1816 for more details about the relationship between FIML and ML methods.

Whenever you use the FIML method, the mean structures are automatically assumed in the analysis. This is due to fact that there is no closed-form formula to obtain the saturated mean vector in the FIML discrepancy function if missing values are present in the data. You can certainly provide explicit specification of the mean parameters in the model by specifying intercepts in the LINEQS statement or means and intercepts in the MEAN or MATRIX statement. However, usually you do not need to do the explicit specification if all you need to achieve is to saturate the mean structures with p parameters (that is, the same number as the number of observed variables in the model). With METHOD=FIML, PROC CALIS uses certain default parameterizations for the mean structures automatically. For example, all intercepts of endogenous observed variables and all means of exogenous observed variables are default parameters in the model, making the explicit specification of these mean structure parameters unnecessary.

WLS and ADF Discrepancy Functions

Another important discrepancy function to consider is the weighted least squares (WLS) function. Let $\mathbf{u} = (\mathbf{s}, \bar{\mathbf{x}})$ be a $p(p+3)/2$ vector containing all nonredundant elements in the sample covariance matrix $\mathbf{S}$ and sample mean vector $\bar{\mathbf{x}}$, with $\mathbf{s} = \mathrm{vecs}(\mathbf{S})$ representing the vector of the $p(p+1)/2$ lower triangle elements of the symmetric matrix $\mathbf{S}$, stacking row by row. Similarly, let $\boldsymbol{\eta} = (\boldsymbol{\sigma}, \boldsymbol{\mu})$ be a $p(p+3)/2$ vector containing all nonredundant elements in the fitted covariance matrix $\boldsymbol{\Sigma}$ and the fitted mean vector $\boldsymbol{\mu}$, with $\boldsymbol{\sigma} = \mathrm{vecs}(\boldsymbol{\Sigma})$ representing the vector of the $p(p+1)/2$ lower triangle elements of the symmetric matrix $\boldsymbol{\Sigma}$.

The WLS discrepancy function is:

$$F_{WLS} = (\mathbf{u} - \boldsymbol{\eta})'\mathbf{W}^{-1}(\mathbf{u} - \boldsymbol{\eta})$$

where $\mathbf{W}$ is a positive definite symmetric weight matrix with $(p(p+3)/2)$ rows and columns. Because $\boldsymbol{\eta}$ is a function of model parameter vector $\boldsymbol{\Theta}$ under the structural model, you can write the WLS function as:

$$F_{WLS} = (\mathbf{u} - \boldsymbol{\eta}(\boldsymbol{\Theta}))'\mathbf{W}^{-1}(\mathbf{u} - \boldsymbol{\eta}(\boldsymbol{\Theta}))$$

Suppose that $\mathbf{u}$ converges to $\boldsymbol{\eta}_o = (\boldsymbol{\sigma}_o, \boldsymbol{\mu}_o)$ with increasing sample size, where $\boldsymbol{\sigma}_o$ and $\boldsymbol{\mu}_o$ denote the population covariance matrix and mean vector, respectively. By default, the WLS weight matrix $\mathbf{W}$ in PROC CALIS is computed from the raw data as a consistent estimate of the asymptotic covariance matrix $\boldsymbol{\Gamma}$ of $\sqrt{N}(\mathbf{u} - \boldsymbol{\eta}_o)$, with $\boldsymbol{\Gamma}$ partitioned as

$$\Gamma = \begin{pmatrix} \boldsymbol{\Gamma}_{ss} & \boldsymbol{\Gamma}'_{\bar{x}s} \\ \boldsymbol{\Gamma}_{\bar{x}s} & \boldsymbol{\Gamma}_{\bar{x}\bar{x}} \end{pmatrix}$$

where $\boldsymbol{\Gamma}_{ss}$ denotes the $(p(p+1)/2) \times (p(p+1)/2)$ asymptotic covariance matrix for $\sqrt{N}(\mathbf{s} - \boldsymbol{\sigma}_o)$, $\boldsymbol{\Gamma}_{\bar{x}\bar{x}}$ denotes the $p \times p$ asymptotic covariance matrix for $\sqrt{N}(\bar{\mathbf{x}} - \boldsymbol{\mu}_o)$, and $\boldsymbol{\Gamma}_{\bar{x}s}$ denotes the $p \times (p(p+1)/2)$ asymptotic covariance matrix between $\sqrt{N}(\bar{\mathbf{x}} - \boldsymbol{\mu}_o)$ and $\sqrt{N}(\mathbf{s} - \boldsymbol{\sigma}_o)$.

To compute the default weight matrix $\mathbf{W}$ as a consistent estimate of $\boldsymbol{\Gamma}$, define a similar partition of the weight matrix $\mathbf{W}$ as:

$$\mathbf{W} = \begin{pmatrix} \mathbf{W}_{ss} & \mathbf{W}'_{\bar{x}s} \\ \mathbf{W}_{\bar{x}s} & \mathbf{W}_{\bar{x}\bar{x}} \end{pmatrix}$$

Each of the submatrices in the partition can now be computed from the raw data. First, define the biased sample covariance for variables i and j as:

$$\mathbf{t}_{ij} = \frac{1}{N}\sum_{r=1}^{N}(x_{ri} - \bar{x}_i)(x_{rj} - \bar{x}_j)$$

and the sample fourth-order central moment for variables i, j, k, and l as:

$$\mathbf{t}_{ij,kl} = \frac{1}{N}\sum_{r=1}^{N}(x_{ri} - \bar{x}_i)(x_{rj} - \bar{x}_j)(x_{rk} - \bar{x}_k)(x_{rl} - \bar{x}_l)$$

The submatrices in $\mathbf{W}$ are computed by:

$$[\mathbf{W}_{ss}]_{ij,kl} = \mathbf{t}_{ij,kl} - \mathbf{t}_{ij}\mathbf{t}_{kl}$$

$$[\mathbf{W}_{\bar{x}s}]_{i,kl} = \frac{1}{N}\sum_{r=1}^{N}(x_{ri}-\bar{x}_i)(x_{rk}-\bar{x}_k)(x_{rl}-\bar{x}_l)$$

$$[\mathbf{W}_{\bar{x}\bar{x}}]_{ij} = \mathbf{t}_{ij}$$

Assuming the existence of finite eighth-order moments, this default weight matrix $\mathbf{W}$ is a consistent but biased estimator of the asymptotic covariance matrix $\boldsymbol{\Gamma}$.

By using the ASYCOV= option, you can use Browne's unbiased estimator (Browne 1984, formula (3.8)) of $\boldsymbol{\Gamma}_{ss}$ as:

$$\begin{aligned}[\mathbf{W}_{ss}]_{ij,kl} &= \frac{N(N-1)}{(N-2)(N-3)}(\mathbf{t}_{ij,kl}-\mathbf{t}_{ij}\mathbf{t}_{kl}) \\ &\quad -\frac{N}{(N-2)(N-3)}(\mathbf{t}_{ik}\mathbf{t}_{jl}+\mathbf{t}_{il}\mathbf{t}_{jk}-\frac{2}{N-1}\mathbf{t}_{ij}\mathbf{t}_{kl})\end{aligned}$$

There is no guarantee that $\mathbf{W}_{ss}$ computed this way is positive semidefinite. However, the second part is of order $O(N^{-1})$ and does not destroy the positive semidefinite first part for sufficiently large N. For a large number of independent observations, default settings of the weight matrix $\mathbf{W}$ result in asymptotically distribution-free parameter estimates with unbiased standard errors and a correct χ^2 test statistic (Browne 1982, 1984).

With the default weight matrix $\mathbf{W}$ computed by PROC CALIS, the WLS estimation is also called as the asymptotically distribution-free (ADF) method. In fact, as options in PROC CALIS, METHOD=WLS and METHOD=ADF are totally equivalent, even though WLS in general might include cases with special weight matrices other than the default weight matrix.

When the mean structures are not modeled, the WLS discrepancy function is still the same quadratic form statistic. However, with only the elements in covariance matrix being modeled, the dimensions of $\mathbf{u}$ and $\boldsymbol{\eta}$ are both reduced to $p(p+1)/2\times 1$, and the dimension of the weight matrix is now $(p(p+1)/2)\times(p(p+1)/2)$. That is, the WLS discrepancy function for covariance structure models is:

$$F_{WLS} = (\mathbf{s}-\boldsymbol{\sigma})'\mathbf{W}_{ss}^{-1}(\mathbf{s}-\boldsymbol{\sigma})$$

If $\mathbf{S}$ is a correlation rather than a covariance matrix, the default setting of the $\mathbf{W}_{ss}$ is a consistent estimator of the asymptotic covariance matrix $\boldsymbol{\Gamma}_{ss}$ of $\sqrt{N}(\mathbf{s}-\boldsymbol{\sigma}_o)$ (Browne and Shapiro 1986; De Leeuw 1983), with $\mathbf{s}$ and $\boldsymbol{\sigma}_o$ representing vectors of sample and population correlations, respectively. Elementwise, $\mathbf{W}_{ss}$ is expressed as:

$$\begin{aligned}[\mathbf{W}_{ss}]_{ij,kl} &= r_{ij,kl}-\frac{1}{2}r_{ij}(r_{ii,kl}+r_{jj,kl})-\frac{1}{2}r_{kl}(r_{kk,ij}+r_{ll,ij}) \\ &\quad +\frac{1}{4}r_{ij}r_{kl}(r_{ii,kk}+r_{ii,ll}+r_{jj,kk}+r_{jj,ll})\end{aligned}$$

where

$$r_{ij} = \frac{\mathbf{t}_{ij}}{\sqrt{\mathbf{t}_{ii}\mathbf{t}_{jj}}}$$

and

$$r_{ij,kl} = \frac{\mathbf{t}_{ij,kl}}{\sqrt{\mathbf{t}_{ii}\mathbf{t}_{jj}\mathbf{t}_{kk}\mathbf{t}_{ll}}}$$

The asymptotic variances of the diagonal elements of a correlation matrix are 0. That is,

$$[\mathbf{W}_{ss}]_{ii,ii} = 0$$

for all i. Therefore, the weight matrix computed this way is always singular. In this case, the discrepancy function for weighted least squares estimation is modified to:

$$\begin{aligned} F_{WLS} &= \sum_{i=2}^{p}\sum_{j=1}^{i-1}\sum_{k=2}^{p}\sum_{l=1}^{k-1}[\mathbf{W}_{ss}]^{ij,kl}([\mathbf{S}]_{ij} - [\mathbf{\Sigma}]_{ij})([\mathbf{S}]_{kl} - [\mathbf{\Sigma}]_{kl}) \\ &\quad + r\sum_{i}^{p}([\mathbf{S}]_{ii} - [\mathbf{\Sigma}]_{ii})^2 \end{aligned}$$

where r is the penalty weight specified by the WPENALTY=r option and the $[\mathbf{W}_{ss}]^{ij,kl}$ are the elements of the inverse of the reduced $(p(p-1)/2) \times (p(p-1)/2)$ weight matrix that contains only the nonzero rows and columns of the full weight matrix $\mathbf{W}_{ss}$.

The second term is a penalty term to fit the diagonal elements of the correlation matrix $\mathbf{S}$. The default value of $r = 100$ can be decreased or increased by the WPENALTY= option. The often used value of $r = 1$ seems to be too small in many cases to fit the diagonal elements of a correlation matrix properly.

Note that when you model correlation structures, no mean structures can be modeled simultaneously in the same model.

DWLS Discrepancy Functions

Storing and inverting the huge weight matrix $\mathbf{W}$ in WLS estimation requires considerable computer resources. A compromise is found by implementing the diagonally weighted least squares (DWLS) method that uses only the diagonal of the weight matrix $\mathbf{W}$ from the WLS estimation in the following discrepancy function:

$$\begin{aligned} F_{DWLS} &= (\mathbf{u} - \boldsymbol{\eta})'[\mathrm{diag}(\mathbf{W})]^{-1}(\mathbf{u} - \boldsymbol{\eta}) \\ &= \sum_{i=1}^{p}\sum_{j=1}^{i}[\mathbf{W}_{ss}]^{-1}_{ij,ij}([\mathbf{S}]_{ij} - [\mathbf{\Sigma}]_{ij})^2 + \sum_{i=1}^{p}[\mathbf{W}_{\bar{x}\bar{x}}]^{-1}_{ii}(\bar{\mathbf{x}}_i - \boldsymbol{\mu}_i)^2 \end{aligned}$$

When only the covariance structures are modeled, the discrepancy function becomes:

$$F_{DWLS} = \sum_{i=1}^{p}\sum_{j=1}^{i}[\mathbf{W}_{ss}]^{-1}_{ij,ij}([\mathbf{S}]_{ij} - [\mathbf{\Sigma}]_{ij})^2$$

For correlation models, the discrepancy function is:

$$F_{DWLS} = \sum_{i=2}^{p}\sum_{j=1}^{i-1}[\mathbf{W}_{ss}]^{-1}_{ij,ij}([\mathbf{S}]_{ij} - [\mathbf{\Sigma}]_{ij})^2 + r\sum_{i=1}^{p}([\mathbf{S}]_{ii} - [\mathbf{\Sigma}]_{ii})^2$$

where r is the penalty weight specified by the WPENALTY=r option. Note that no mean structures can be modeled simultaneously with correlation structures when using the DWLS method.

As the statistical properties of DWLS estimates are still not known, standard errors for estimates are not computed for the DWLS method.

Input Weight Matrices

In GLS, WLS, or DWLS estimation you can change from the default settings of weight matrices $\mathbf{W}$ by using an INWGT= data set. The CALIS procedure requires a positive definite weight matrix that has positive diagonal elements.

Multiple-Group Discrepancy Function

Suppose that there are k independent groups in the analysis and $N_1, N_2, \ldots, N_k$ are the sample sizes for the groups. The overall discrepancy function $F(\mathbf{\Theta})$ is expressed as a weighted sum of individual discrepancy functions F_i's for the groups:

$$F(\mathbf{\Theta}) = \sum_{i=1}^{k} t_i F_i(\mathbf{\Theta})$$

where

$$t_i = \frac{N_i - 1}{N - k}$$

is the weight of the discrepancy function for group i, and

$$N = \sum_{i=1}^{k} N_i$$

is the total number of observations in all groups. In PROC CALIS, all discrepancy function F_i's in the overall discrepancy function must belong to the same estimation method. You cannot specify different estimation methods for the groups in a multiple-group analysis. In addition, the same analysis type must be applied to all groups—that is, you can analyze either covariance structures, covariance and mean structures, and correlation structures for all groups.

Robust Estimation

Two robust estimation methods that are proposed by Yuan and Zhong (2008) and Yuan and Hayashi (2010) are implemented in PROC CALIS. The first method is the two-stage robust method, which estimates robust covariance and mean matrices in the first stage and then feeds the robust covariance and mean matrices (in place of the ordinary sample covariance and mean matrices) for ML estimation in the second stage. Weighting of the observations is done only in the first stage. The ROBUST=SAT option invokes the two-stage robust estimation. The second method is the direct robust method, which iteratively estimates model parameters with simultaneous weightings of the observations. The ROBUST, ROBUST=RES(E), or ROBUST=RES(F) option invokes the direct robust estimation method.

The procedural difference between the two robust methods results in differential treatments of model outliers and leverage observations (or leverage points). In producing the robust covariance and mean matrices in the first stage, the two-stage robust method downweights outlying observations in all variable dimensions without

regard to the model structure. This method downweights potential model outliers and leverage observations (which are not necessarily model outliers) in essentially the same way before the ML estimation in the second stage.

However, the direct robust method downweights the model outliers only. "Good" leverage observations (those that are not outliers at the same time) are not downweighted for model estimation. Therefore, it could be argued that the direct robust method is more desirable if you can be sure that the model is a reasonable one. The reason is that the direct robust method can retain the information from the "good" leverage observations for estimation, while the two-stage robust method downweights all leverage observations indiscriminately during its first stage. However, if the model is itself uncertain, the two-stage robust estimation method might be more foolproof.

Both robust methods employ weights on the observations. Weights are functions of the Mahalanobis distances (M-distances) of the observations and are computed differently for the two robust methods. The following two sections describe the weighting scheme and the estimation procedure of the two robust methods in more detail.

Two-Stage Robust Method

For the two-stage robust method, the following conventional M-distance d_s for an observed random vector $\mathbf{x}$ is computed as

$$d_s = \sqrt{(\mathbf{x} - \boldsymbol{\mu})' \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})}$$

where $\boldsymbol{\mu}$ and Σ are the unstructured mean and covariance matrices, respectively.

Two sets of weights are computed as functions of the M-distances of the observations. The weighting functions are essentially the same form as that of Huber (see, for example, Huber 1981). Let d_i be the M-distance of observation i, computed from the d_s formula. The first set of weights $w_1(d_i)$ applies to the first moments of the data and is defined as

$$\begin{aligned} w_1(d_i) &= 1 \qquad (\text{if } d_i \le \rho) \\ &= \rho/d_i \quad (\text{if } d_i > \rho) \end{aligned}$$

where ρ is the critical value corresponding to the $(1-\varphi) \times 100$ quantile of the $\chi_r = \sqrt{\chi_r{}^2}$ distribution, with r being the degrees of freedom. For the two-stage robust method, r is simply the number of observed variables in the analysis. The tuning parameter φ controls the approximate proportion of observations to be downweighted (that is, with $w_1(d_i)$ less than 1). The default φ value is set to 0.05. You can override this value by using the ROBPHI= option.

The second set of weights $w_2(d_i)$ applies to the second moments of the data and is defined as

$$w_2(d_i) = (w_1(d_i))^2/\kappa$$

where κ is a constant that adjusts the sum $\sum_{i=1}^{N} w_2(d_i)$ to 1 approximately. After the tuning parameter φ is determined, the critical value ρ and the adjustment κ are computed automatically by PROC CALIS.

With these two sets of weights, the two-stage robust method (ROBUST=SAT) estimates the mean and covariance by the so-called iteratively reweighted least squares (IRLS) algorithm. Specifically, the updating

formulas at the j+1 iteration are

$$\begin{aligned}
\boldsymbol{\mu}^{(j+1)} &= \frac{\sum_{i=1}^{N} w_1({d_i}^{(j)})\mathbf{x}_i}{\sum_{i=1}^{n} w_1({d_i}^{(j)})} \\
\boldsymbol{\Sigma}^{(j+1)} &= \frac{1}{N}\sum_{i=1}^{N} w_2({d_i}^{(j)})(\mathbf{x}_i - \boldsymbol{\mu}^{(j)})(\mathbf{x}_i - \boldsymbol{\mu}^{(j)})'
\end{aligned}$$

where ${d_i}^{(j)}$ is the M-distance evaluated at $\boldsymbol{\mu}^{(j)}$ and $\Sigma^{(j)}$ obtained in the jth iteration. Carry out the iterations until $\boldsymbol{\mu}$ and Σ converge. The final iteration yields the robust estimates of mean and covariance matrices. PROC CALIS uses the relative parameter convergence criterion for the IRLS algorithm. The default criterion is 1E–8. See the XCONV= option for the definition of the relative parameter convergence criterion. After the IRLS algorithm converges in the first stage, the two-stage robust method proceeds to treat the robust mean and covariance estimates as if they were sample mean and covariance matrices for a maximum likelihood estimation (METHOD=ML) of the model.

Direct Robust Method

The direct robust method computes the following residual M-distance d_r for an observation with residual random vector $\hat{\mathbf{e}}$ (say, of dimension $h \times 1$, where h is the number of dependent observed variables $\mathbf{y}$):

$$d_r = \sqrt{(\mathbf{L}\hat{\mathbf{e}})'(\mathbf{L}\Omega_{\hat{\mathbf{e}}}\mathbf{L}')^{-1}(\mathbf{L}\hat{\mathbf{e}})}$$

where $\mathbf{L}$ ($(h-q) \times h$) is a loading matrix that reduces $\hat{\mathbf{e}}$ to $(h-q)$ independent components and q is the number of independent factors to be estimated from the dependent observed variables $\mathbf{y}$. The reduction of the residual vector into independent components is necessary when the number of factors q is not zero in the model. For $q > 0$, the residual covariance matrix $\Omega_{\hat{\mathbf{e}}}$ is not invertible and cannot be used in computing the residual M-distances. Hence, the covariance matrix $\mathbf{L}\Omega_{\hat{\mathbf{e}}}\mathbf{L}'$ of independent components $\mathbf{L}\hat{\mathbf{e}}$ is used instead. See Yuan and Hayashi (2010) for details about the computation of the residuals in the context of structural equation modeling.

The direct robust method also computes two sets of weights as functions of the residual M-distances. Let d_i be the M-distance of observation i, computed from the d_r formula. The first set of weights $w_1(d_i)$ applies to parameters in the first moments and is defined as

$$\begin{aligned}
w_1(d_i) &= 1 \qquad (\text{if } d_i \le \rho) \\
&= \rho/d_i \quad (\text{if } d_i > \rho)
\end{aligned}$$

The second set of weights $w_2(d_i)$ applies to the parameters in the second moments and is defined as

$$w_2(d_i) = (w_1(d_i))^2/\kappa$$

These are essentially the same Huber-type weighting functions as those for the two-stage robust method. The only difference is that the d_r, instead of the d_s, formula is used in the weighting functions for the direct robust method. The definition of ρ is also the same as that in the two-stage robust method, but it is now based on a different theoretical chi-square distribution. That is, in the direct robust method, ρ is the critical value that corresponds to the $(1-\varphi) \times 100$ quantile of the $\chi_r = \sqrt{{\chi_r}^2}$ distribution, with $r = (h-q)$ being the degrees of freedom. Again, φ is a tuning parameter and is set to 0.05 by default. You can override this value by using the ROBPHI= option. The calculation of the number of "independent factors" q depends on the

variants of the direct robust estimation that you choose. With the ROBUST=RES(E) option, q is the same as the number of exogenous factors specified in the model. With the ROBUST=RES(F) option, the disturbances of the endogenous factors in the model are also treated as "independent factors," so q is the total number of latent factors specified in the model.

The direct robust method (ROBUST=RES(E) or ROBUST=RES(F)) employs the IRLS algorithm in model estimation. Let the following expression be a vector of nonredundant first and second moments structured in terms of model parameters $\boldsymbol{\theta}$:

$$\mathbf{m}(\boldsymbol{\theta}) = \begin{pmatrix} \boldsymbol{\mu}(\boldsymbol{\theta}) \\ \mathrm{vech}(\boldsymbol{\Sigma}(\boldsymbol{\theta})) \end{pmatrix}$$

where vech() extracts the lower-triangular nonredundant elements in $\boldsymbol{\Sigma}(\boldsymbol{\theta})$. The updating formulas for $\boldsymbol{\theta}$ at the j+1 iteration is

$$\boldsymbol{\theta}^{(j+1)} = \boldsymbol{\theta}^{(j)} + \Delta\boldsymbol{\theta}^{(j)}$$

where $\boldsymbol{\theta}^{(j)}$ is the parameter values at the jth iteration and $\Delta\boldsymbol{\theta}^{(j)}$ is defined by the following formula:

$$\Delta\boldsymbol{\theta}^{(j)} = (\dot{\mathbf{m}}'(\boldsymbol{\theta}^{(j)})\mathbf{W}(\boldsymbol{\theta}^{(j)})\dot{\mathbf{m}}(\boldsymbol{\theta}^{(j)}))^{-1}\dot{\mathbf{m}}'(\boldsymbol{\theta}^{(j)})\mathbf{W}(\boldsymbol{\theta}^{(j)})\mathbf{g}_j$$

where $\dot{\mathbf{m}}(\boldsymbol{\theta}) = \frac{\partial \mathbf{m}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}'}$ is the model Jacobian, $\mathbf{W}(\boldsymbol{\theta})$ is the (normal theory) weight matrix for the moments (see Yuan and Zhong 2008 for the formula of the weight matrix), and $\mathbf{g}_j$ is defined as

$$\mathbf{g}_j = \begin{pmatrix} \frac{1}{\sum_{i=1}^{N} w_1(d_i)} \sum_{i=1}^{N} w_1(d_i)\mathbf{x}_i - \boldsymbol{\mu}(\boldsymbol{\theta}^{(j)}) \\ \frac{1}{N}\sum_{i=1}^{N} w_2(d_i)\mathrm{vech}[(\mathbf{x}_i - \boldsymbol{\mu}(\boldsymbol{\theta}^{(j)}))(\mathbf{x}_i - \boldsymbol{\mu}(\boldsymbol{\theta}^{(j)}))'] - \Sigma(\boldsymbol{\theta}^{(j)}) \end{pmatrix}$$

Starting with some reasonable initial estimates for $\boldsymbol{\theta}$, PROC CALIS iterates the updating formulas until the relative parameter convergence criterion of the IRLS algorithm is satisfied. The default criterion value is 1E–8. This essentially means that the IRLS algorithm converges when $\Delta\boldsymbol{\theta}$ is sufficiently small. See the XCONV= option for the definition of the relative parameter convergence criterion.

Although the iterative formulas and the IRLS steps for robust estimation have been presented for single-group analysis, they are easily generalizable to multiple-group analysis. For the two-stage robust method, you only need to repeat the robust estimation of the means and covariances for the groups and then apply the obtained robust moments as if they were sample moments for regular maximum likelihood estimation (METHOD=ML). For the direct robust method, you need to expand the dimensions of the model Jacobian matrix $\dot{\mathbf{m}}(\boldsymbol{\theta})$, the weight matrix $\mathbf{W}(\boldsymbol{\theta})$, and the vector $\mathbf{g}$ to include moments from several groups/models. Therefore, the multiple-group formulas are conceptually quite simple but tedious to present. For this reason, they are omitted here.

Relationships among Estimation Criteria

There is always some arbitrariness to classify the estimation methods according to certain mathematical or numerical properties. The discussion in this section is not meant to be a thorough classification of the estimation methods available in PROC CALIS. Rather, classification is done here with the purpose of clarifying the uses of different estimation methods and the theoretical relationships of estimation criteria.

Assumption of Multivariate Normality

GLS, ML, and FIML assume multivariate normality of the data, while ULS, WLS, and DWLS do not. Although the ML method with covariance structure analysis alone can also be based on the Wishart distribution of the sample covariance matrix, for convenience GLS, ML, and FIML are usually classified as normal-theory based methods, while ULS, WLS, and DWLS are usually classified as distribution-free methods.

An intuitive or even naive notion is usually that methods without distributional assumptions such as WLS and DWLS are preferred to normal theory methods such as ML and GLS in practical situations where multivariate normality is doubt. This notion might need some qualifications because there are simply more factors to consider in judging the quality of estimation methods in practice. For example, the WLS method might need a very large sample size to enjoy its purported asymptotic properties, while the ML might be robust against the violation of multi-normality assumption under certain circumstances. No recommendations regarding which estimation criterion should be used are attempted here, but you should make your choice based more than the assumption of multivariate normality.

Contribution of the Off-Diagonal Elements to the Estimation of Covariance or Correlation Structures

If only the covariance or correlation structures are considered, the six estimation functions, F_{ULS}, F_{GLS}, F_{ML}, F_{FIML}, F_{WLS}, and F_{DWLS}, belong to the following two groups:

- The functions F_{ULS}, F_{GLS}, F_{ML}, and F_{FIML} take into account all n^2 elements of the symmetric residual matrix $\mathbf{S} - \mathbf{\Sigma}$. This means that the off-diagonal residuals contribute twice to the discrepancy function F, as lower and as upper triangle elements.
- The functions F_{WLS} and F_{DWLS} take into account only the $n(n+1)/2$ lower triangular elements of the symmetric residual matrix $\mathbf{S} - \mathbf{\Sigma}$. This means that the off-diagonal residuals contribute to the discrepancy function F only once.

The F_{DWLS} function used in PROC CALIS differs from that used by the LISREL 7 program. Formula (1.25) of the LISREL 7 manual (Jöreskog and Sörbom 1985, p. 23) shows that LISREL groups the F_{DWLS} function in the first group by taking into account all n^2 elements of the symmetric residual matrix $\mathbf{S} - \mathbf{\Sigma}$.

- Relationship between DWLS and WLS:
  PROC CALIS: The F_{DWLS} and F_{WLS} discrepancy functions deliver the same results for the special case that the weight matrix $\mathbf{W} = \mathbf{W}_{ss}$ used by WLS estimation is a diagonal matrix.
  LISREL 7: This is not the case.
- Relationship between DWLS and ULS:
  LISREL 7: The F_{DWLS} and F_{ULS} estimation functions deliver the same results for the special case

that the diagonal weight matrix $\mathbf{W} = \mathbf{W}_{ss}$ used by DWLS estimation is an identity matrix.
PROC CALIS: To obtain the same results with F_{DWLS} and F_{ULS} estimation, set the diagonal weight matrix $\mathbf{W} = \mathbf{W}_{ss}$ used in DWLS estimation to:

$$[\mathbf{W}_{ss}]_{ik,ik} = \begin{cases} 1. & \text{if } i = k \\ 0.5 & \text{otherwise} \qquad (k \le i) \end{cases}$$

Because the reciprocal elements of the weight matrix are used in the discrepancy function, the off-diagonal residuals are weighted by a factor of 2.

ML and FIML Methods

Both the ML and FIML methods can be derived from the log-likelihood function for multivariate normal data. The preceding section "Estimation Criteria" on page 1805 mentions that F_{FIML} is essentially the same as $\frac{-2L}{N}$, where L is the log-likelihood function for multivariate normal data. For the ML estimation, you can also consider $\frac{-2L}{N}$ as a part of the F_{ML} discrepancy function that contains the information regarding the model parameters (while the rest the F_{ML} function contains some constant terms given the data). That is, with some algebraic manipulations and assuming that there is no missing value in the analysis (so that all $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ are the same as $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, respectively), it can shown that

$$\begin{aligned} F_{FIML} &= \frac{-2L}{N} \\ &= \frac{1}{N}\sum_{j=1}^{n}(\ln(|\boldsymbol{\Sigma}|) + (\mathbf{x}_j - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\mathbf{x}_j - \boldsymbol{\mu}) + K) \\ &= \ln(|\boldsymbol{\Sigma}|) + \mathrm{Tr}(\mathbf{S_N}\boldsymbol{\Sigma}^{-1}) + (\bar{\mathbf{x}} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\bar{\mathbf{x}} - \boldsymbol{\mu}) + K \end{aligned}$$

where $\bar{\mathbf{x}}$ is the sample mean and $\mathbf{S_N}$ is the biased sample covariance matrix. Compare this FIML function with the ML function shown in the following expression, which shows that both functions are very similar:

$$F_{ML} = \ln(|\boldsymbol{\Sigma}|) + \mathrm{Tr}(\mathbf{S}\boldsymbol{\Sigma}^{-1}) + (\bar{\mathbf{x}} - \boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\bar{\mathbf{x}} - \boldsymbol{\mu}) - p - \ln(|\mathbf{S}|)$$

The two expressions differ only in the constant terms, which are independent of the model parameters, and in the formulas for computing the sample covariance matrix. While the FIML method assumes the biased formula (with N as the divisor, by default) for the sample covariance matrix, the ML method (as implemented in PROC CALIS) uses the unbiased formula (with $N - 1$ as the divisor, by default).

The similarity (or dissimilarity) of the ML and FIML discrepancy functions leads to some useful conclusions here:

- Because the constant terms in the discrepancy functions play no part in parameter estimation (except for shifting the function values), overriding the default ML method with VARDEF=N (that is, using N as the divisor in the covariance matrix formula) leads to the same estimation results as that of the FIML method, given that there are no missing values in the analysis.
- Because the FIML function is evaluated at the level of individual observations, it is much more expensive to compute than the ML function. As compared with ML estimation, FIML estimation takes longer and uses more computing resources. Hence, for data without missing values, the ML method should always be chosen over the FIML method.

- The advantage of the FIML method lies solely in its ability to handle data with random missing values. While the FIML method uses the information maximally from each observation, the ML method (as implemented in PROC CALIS) simply throws away any observations with at least one missing value. If it is important to use the information from observations with random missing values, the FIML method should be given consideration over the ML method.

See Example 33.15 for an application of the FIML method and Example 33.16 for an empirical comparison of the ML and FIML methods. For more examples and details about the FIML method employed by PROC CALIS, see Yung and Zhang (2011); Zhang and Yung (2011).

Gradient, Hessian, Information Matrix, and Approximate Standard Errors

For a single-sample setting with a discrepancy function $F = F(\mathbf{\Sigma}(\mathbf{\Theta}), \boldsymbol{\mu}(\mathbf{\Theta}); \mathbf{S}, \bar{\mathbf{x}})$, the gradient is defined as the first partial derivatives of the discrepancy function with respect to the model parameters $\mathbf{\Theta}$:

$$g(\mathbf{\Theta}) = \frac{\partial}{\partial \mathbf{\Theta}} F(\mathbf{\Theta})$$

The Hessian is defined as the second partial derivatives of the discrepancy function with respect to the model parameters $\mathbf{\Theta}$:

$$H(\mathbf{\Theta}) = \frac{\partial^2}{\partial \mathbf{\Theta} \partial \mathbf{\Theta}'} F(\mathbf{\Theta})$$

Suppose that the mean and covariance structures fit perfectly with $\mathbf{\Theta} = \mathbf{\Theta}_o$ in the population. The expected information matrix is defined as

$$I(\mathbf{\Theta}_o) = \frac{1}{2} \mathcal{E}(H(\mathbf{\Theta}_o))$$

where the expectation $\mathcal{E}(\cdot)$ is taken over the sampling space of $\mathbf{S}$ and $\bar{\mathbf{x}}$. Hence, the expected information matrix $I(\mathbf{\Theta}_o)$ does not contain any sample values.

The expected information matrix plays a significant role in statistical theory. Under certain regularity conditions, the inverse of the information matrix $I^{-1}(\mathbf{\Theta}_o)$ is the asymptotic covariance matrix for $\sqrt{N}(\hat{\mathbf{\Theta}} - \mathbf{\Theta}_o)$, where N denotes the sample size and $\hat{\mathbf{\Theta}}$ is an estimator.

In practice, $\mathbf{\Theta}_o$ is never known and can only be estimated. The information matrix is therefore evaluating at the sample estimate $\hat{\mathbf{\Theta}}$ and is denoted as

$$I(\hat{\mathbf{\Theta}})$$

This is the information matrix that PROC CALIS displays in the output.

For a sample of size N, PROC CALIS computes the estimated covariance matrix of $\hat{\mathbf{\Theta}}$ by

$$((N-1) I(\hat{\mathbf{\Theta}}))^{-1}$$

It then computes approximate standard errors for $\hat{\mathbf{\Theta}}$ as the square roots of the diagonal elements of this estimated covariance matrix. This formula is based on the expected information and is the default standard error method (INFORMATION=EXP) for the ML, MLSB, GLS, and WLS estimation methods.

In contrast, by default the FIML estimation method computes standard error estimates based on the so-called observed information matrix (INFORMATION=OBS), which is defined as

$$I_{obs}(\boldsymbol{\Theta}_o) = \frac{1}{2} H(\boldsymbol{\Theta}_o)$$

The critical difference between $I(\boldsymbol{\Theta}_o)$ and $I_{obs}(\boldsymbol{\Theta}_o)$ is that the latter does not take the expectation of $H(\boldsymbol{\Theta}_o)$ over the distribution of sample statistics. Kenward and Molenberghs (1998) show that the use of the expected information leads to biased standard errors when the missing data mechanism satisfies only the missing at random (MAR; see Rubin 1976) condition but not the missing completely at random (MCAR) condition. Under the MAR condition, the observed information matrix is the correct choice. Because the FIML estimation is mostly applied when the data contain missing values, using the observed information by default is quite reasonable.

In practice, the observed information is computed by

$$I_{obs}(\hat{\boldsymbol{\Theta}})$$

and the estimated covariance matrix of $\hat{\boldsymbol{\Theta}}$ is given by

$$((N-1) I_{obs}(\hat{\boldsymbol{\Theta}}))^{-1}$$

However, PROC CALIS does not compute $I_{obs}(\hat{\boldsymbol{\Theta}})$ analytically. It computes $I_{obs}(\hat{\boldsymbol{\Theta}})$ by the finite-difference method based on the analytic formulas of the first-order partial derivatives of F.

Finally, PROC CALIS does not compute standard errors when you use the ULS and DWLS estimation methods.

If a particular information matrix is singular, PROC CALIS offers two ways to compute a generalized inverse of the matrix and, therefore, two ways to compute approximate standard errors of implicitly constrained parameter estimates, t values, and modification indices. Depending on the G4= specification, either a Moore-Penrose inverse or a G2 inverse is computed. The computationally expensive Moore-Penrose inverse calculates an estimate of the null space by using an eigenvalue decomposition. The computationally cheaper G2 inverse is produced by sweeping the linearly independent rows and columns and zeroing out the dependent ones.

Satorra-Bentler Sandwich Formula for Standard Errors

In addition to the scaled chi-square statistics, Satorra and Bentler (1994) propose the so-called sandwich formula for computing standard errors. For ML estimation, let $C(\hat{\boldsymbol{\Theta}})$ be the estimated covariance matrix of the parameter estimates, obtained through either the expected or observed information matrix formula. The Satorra-Bentler sandwich formula for the estimated covariance matrix is of the form

$$C_{SB}(\hat{\boldsymbol{\Theta}}) = C(\hat{\boldsymbol{\Theta}}) \boldsymbol{\Upsilon}(\hat{\boldsymbol{\Sigma}}) C(\hat{\boldsymbol{\Theta}})$$

where $\Upsilon(\hat{\boldsymbol{\Sigma}})$ depends on the model Jacobian, the weight matrix under the normal distribution theory, and the weight matrix under general distribution theory—all evaluated at the sample estimates or the sample data values. See Satorra and Bentler (1994) for detailed formulas.

If you specify METHOD=MLSB, PROC CALIS uses the Satorra-Bentler sandwich formula to compute standard error estimates. For all other estimation methods that can produce standard error estimates, it uses the unadjusted formula by default. To use the unadjusted formula for METHOD=MLSB, you can specify the

STDERR=UNADJ option. To use the Satorra-Bentler sandwich formula for regular ML estimation, you can specify the STDERR=SBSW option.

Theoretically, if the population is truly multivariate normal, the weight matrix under normal distribution theory is correctly specified. Asymptotically, the first two terms in the formula for $C_{SB}(\hat{\boldsymbol{\Theta}})$ cancel out, so that

$$C_{SB}(\hat{\boldsymbol{\Theta}}) = C(\hat{\boldsymbol{\Theta}})$$

That is, you can use the unadjusted covariance formula to compute standard error estimates if the multivariate normality assumption is satisfied.

If the multivariate normal assumption is not true, then the full sandwich formula has to be involved. Specifically, the middle term, $\boldsymbol{\Upsilon}(\hat{\boldsymbol{\Sigma}})$, in the sandwich formula needs to compute the normal-theory weight matrix and other quantities. Because the normal-theory weight matrix is a function of $\hat{\boldsymbol{\Sigma}}$, evaluation of $\boldsymbol{\Upsilon}(\hat{\boldsymbol{\Sigma}})$ depends on the choice of $\hat{\boldsymbol{\Sigma}}$. PROC CALIS uses the model-predicted covariance matrix by default (SBNTW=PRED). You can also use the sample covariance matrix by specifying the SBNTW=OBS option.

Multiple-Group Extensions

In the section "Multiple-Group Discrepancy Function" on page 1812, the overall discrepancy function for multiple-group analysis is defined. The same notation is applied here. To begin with, the overall discrepancy function $F(\boldsymbol{\Theta})$ is expressed as a weighted sum of individual discrepancy functions F_i's for the groups as follows:

$$F(\boldsymbol{\Theta}) = \sum_{i=1}^{k} t_i F_i(\boldsymbol{\Theta})$$

where

$$t_i = \frac{N_i - 1}{N - k}$$

is the weight for group i,

$$N = \sum_{i=1}^{k} N_i$$

is the total sample size, and N_i is the sample size for group i.

The gradient $g(\boldsymbol{\Theta})$ and the Hessian $H(\boldsymbol{\Theta})$ are now defined as weighted sum of individual functions. That is,

$$g(\boldsymbol{\Theta}) = \sum_{i=1}^{k} t_i g_i(\boldsymbol{\Theta}) = \sum_{i=1}^{k} t_i \frac{\partial}{\partial \boldsymbol{\Theta}} F_i(\boldsymbol{\Theta})$$

and

$$H(\boldsymbol{\Theta}) = \sum_{i=1}^{k} t_i H_i(\boldsymbol{\Theta}) = \sum_{i=1}^{k} t_i \frac{\partial^2}{\partial \boldsymbol{\Theta} \partial \boldsymbol{\Theta}'} F_i(\boldsymbol{\Theta})$$

Suppose that the mean and covariance structures fit perfectly with $\mathbf{\Theta} = \mathbf{\Theta}_o$ in the population. If each t_i converges to a fixed constant τ_i ($\tau_i > 0$) with increasing total sample size, the expected information matrix can be written as:

$$I(\mathbf{\Theta}_o) = \frac{1}{2}\sum_{i=1}^{k} \tau_i \mathcal{E}(H_i(\mathbf{\Theta}_o))$$

To compute the expected information empirically, $\hat{\mathbf{\Theta}}$ replaces $\mathbf{\Theta}_o$ in the formula.

PROC CALIS computes the estimated covariance matrix of $\hat{\mathbf{\Theta}}$ by:

$$((N-k)I(\hat{\mathbf{\Theta}}))^{-1}$$

Approximate standard errors for $\hat{\mathbf{\Theta}}$ are then computed as the square roots of the diagonal elements of this estimated covariance matrix.

Again, by default the ML, MLSB, GLS, and WLS estimation use such an expected-information based method in the multiple-group setting. For the FIML estimation, the default standard error method is based on the observed information, which is defined as:

$$I_{obs}(\mathbf{\Theta}_o) = \frac{1}{2}\sum_{i=1}^{k} \tau_i H_i(\mathbf{\Theta}_o)$$

Similar to the single-group analysis, standard errors are not computed with the ULS and DWLS estimation methods in the multiple-group setting.

Testing Rank Deficiency in the Approximate Covariance Matrix for Parameter Estimates

When computing the approximate covariance matrix and hence the standard errors for the parameter estimates, inversion of the scaled information matrix or Hessian matrix is involved. The numerical condition of the information matrix can be very poor in many practical applications, especially for the analysis of unscaled covariance data. The following four-step strategy is used for the inversion of the information matrix.

1. The inversion (usually of a normalized matrix $\mathbf{D}^{-1}\mathbf{I}\mathbf{D}^{-1}$) is tried using a modified form of the Bunch and Kaufman (1977) algorithm, which allows the specification of a different singularity criterion for each pivot. The following three criteria for the detection of rank loss in the information matrix are used to specify thresholds:

   - *ASING* specifies absolute singularity.
   - *MSING* specifies relative singularity depending on the whole matrix norm.
   - *VSING* specifies relative singularity depending on the column matrix norm.

   If no rank loss is detected, the inverse of the information matrix is used for the covariance matrix of parameter estimates, and the next two steps are skipped.

2. The linear dependencies among the parameter subsets are displayed based on the singularity criteria.

3. If the number of parameters t is smaller than the value specified by the G4= option (the default value is 60), the Moore-Penrose inverse is computed based on the eigenvalue decomposition of the information matrix. If you do not specify the NOPRINT option, the distribution of eigenvalues is displayed, and those eigenvalues that are set to zero in the Moore-Penrose inverse are indicated. You should inspect this eigenvalue distribution carefully.

4. If PROC CALIS did not set the right subset of eigenvalues to zero, you can specify the COVSING= option to set a larger or smaller subset of eigenvalues to zero in a further run of PROC CALIS.

Counting the Degrees of Freedom

When fitting covariance and mean structure models, the population moments are hypothesized to be functions of model parameters $\mathbf{\Theta}$. The population moments refer to the first-order moments (means) and the second-order central moments (variances of and covariances among the variables). Usually, the number of nonredundant population moments is greater than the number of model parameters for a structural model. The difference between the two is the degrees of freedom (df) of your model.

Formally, define a multiple-group situation where you have k independent groups in your model. The set of variables in each group might be different so that you have $p_1, p_2, \ldots, p_k$ manifest or observed variables for the k groups. It is assumed that the primary interest is to study the covariance structures. The inclusion of mean structures is optional for each of these groups. Define $\delta_1, \delta_2, \ldots, \delta_k$ as zero-one indicators of the mean structures for the groups. If δ_i takes the value of one, it means that the mean structures of group i is modeled. The total number of nonredundant elements in the moment matrices is thus computed by:

$$q = \sum_{i=1}^{k} (p_i(p_i + 1)/2 + \delta_i p_i)$$

The first term in the summation represents the number of lower triangular elements in the covariance or correlation matrix, while the second term represents the number of elements in the mean matrix. Let t be the total number of independent parameters *in the model*. The degrees of freedom is:

$$df = q - (t - c)$$

where c represents the number of linear equality constraints imposed on the independent parameters in the model. In effect, the $(t - c)$ expression means that each nonredundant linear equality constraint reduces one independent parameter.

Counting the Number of Independent Parameters

To count the number of independent parameters in the model, first you have to distinguish them from the dependent parameters. Dependent parameters are expressed as functions of other parameters in the SAS programming statements. That is, a parameter is dependent if it appears at the left-hand side of the equal sign in a SAS programming statement.

A parameter is independent if it is not dependent. An independent parameter can be specified in the main or subsidiary model specification statements or the PARAMETERS statement, or it is generated automatically by PROC CALIS as additional parameters. Quite intuitively, all independent parameter specified in the

main or subsidiary model specification statements are independent parameters *in the model*. All automatic parameters added by PROC CALIS are also independent parameters *in the model*.

Intentionally or not, some independent parameters specified in the PARMS statement might not be counted as independent parameters in the model. Independent parameters in the PARMS statement belong in the model only when they are used to define at least one dependent parameter specified in the main or subsidiary model specification statements. This restriction eliminates the counting of superfluous independent parameters which have no bearing of model specification.

Note that when counting the number of independent parameters, you are counting the number of distinct independent parameter names but not the number of distinct parameter locations for independent parameters. For example, consider the following statement for defining the error variances in a LINEQS model:

```
variance    E1-E3 = vare1 vare2 vare3;
```

You define three variance parameter locations with three independent parameters vare1, vare2, and vare3. However, in the following specification:

```
variance    E1-E3 = vare vare vare;
```

you still have three variance parameter locations to define, but the number of independent parameter is only one, which is the parameter named vare.

Counting the Number of Linear Equality Constraints

The linear equality constraints refer to those specified in the BOUNDS or LINCON statement. For example, consider the following specification:

```
bounds  3 <= parm01 <= 3;
lincon  3 * parm02 + 2 * parm03 = 12;
```

In the BOUNDS statement, parm01 is constrained to a fixed number 3, and in the LINCON statement, parm02 and parm03 are constrained linearly. In effect, these two statements reduce two independent parameters from the model. In the degrees of freedom formula, the value of c is 2 for this example.

Adjustment of Degrees of Freedom

In some cases, computing degrees of freedom for model fit is not so straightforward. Two important cases are considered in the following.

The first case is when you set linear inequality or boundary constraints in your model, and these inequality or boundary constraints become "active" in your final solution. For example, you might have set inequality boundary and linear constraints as:

```
bounds  0 <= var01;
lincon  3 * beta1 + 2 * beta2 >= 7;
```

The optimal solution occurs at the boundary point so that you observe in the final solution the following two equalities:

```
var01 = 0,
3 * beta1 + 2 * beta2 = 7
```

These two active constraints reduce the number of independent parameters of your original model. As a result, PROC CALIS will automatically increase the degrees of freedom by the number of active linear constraints. Adjusting degrees of freedom not only affects the significance of the model fit chi-square statistic, but it also affects the computation of many fit statistics and indices. See Dijkstra (1992) for a discussion of the validity of statistical inferences with active boundary constraints.

Automatically adjusting *df* in such a situation might not be totally justified in all cases. Statistical estimation is subject to sampling fluctuation. Active constraints might not occur when fitting the same model in new samples. If the researcher believes that those linear inequality and boundary constraints have a small chance of becoming active in repeated sampling, it might be more suitable to turn off the automatic adjustment by using the NOADJDF option in the PROC CALIS statement.

Another case where you need to pay attention to the computation of degrees of freedom is when you fit correlation models. The degrees-of-freedom calculation in PROC CALIS applies mainly to models with covariance structures with or without mean structures. When you model correlation structures, the degrees of freedom calculation in PROC CALIS is a straightforward generalization of the covariance structures. It does not take the fixed ones at the diagonal elements of the sample correlation matrix into account. Some might argue that with correlation structures, the degrees of freedom should be reduced by the total number of diagonal elements in the correlation matrices in the model. While PROC CALIS does not do this automatically, you can use the DFREDUCE=*i* option to specify the adjustment, where *i* can be any positive or negative integer. The *df* value is reduced by the DFREDUCE= value.

A Different Type of Degrees of Freedom

The degrees of freedom for model fitting has to be distinguished from another type of degrees of freedom. In a regression problem, the number of degrees of freedom for the error variance estimate is the number of observations in the data set minus the number of parameters. The NOBS=, RDF= (or DFR=), and EDF= (or DFE=) options refer to degrees of freedom in this sense. However, these values are not related to the degrees of freedom for the model fit statistic. The NOBS=, RDF=, and EDF= options should be used in PROC CALIS to specify the effective number of observations in the input data set only.

Assessment of Fit

In PROC CALIS, there are three main tools for assessing model fit:

- residuals for the fitted means or covariances
- overall model fit indices
- squared multiple correlations and determination coefficients

This section contains a collection of formulas for these assessment tools. The following notation is used:

- N for the total sample size
- k for the total number of independent groups in analysis
- p for the number of manifest variables

- t for the number of parameters to estimate
- $\boldsymbol{\Theta}$ for the t-vector of parameters, $\hat{\boldsymbol{\Theta}}$ for the estimated parameters
- $\mathbf{S} = (s_{ij})$ for the $p \times p$ input covariance or correlation matrix
- $\bar{\mathbf{x}} = (\bar{x}_i)$ for the p-vector of sample means
- $\hat{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}(\hat{\boldsymbol{\Theta}}) = (\hat{\sigma}_{ij})$ for the predicted covariance or correlation matrix
- $\hat{\boldsymbol{\mu}} = (\hat{\mu}_i)$ for the predicted mean vector
- δ for indicating the modeling of the mean structures
- $\mathbf{W}$ for the weight matrix
- f_{min} for the minimized function value of the fitted model
- d_{min} for the degrees of freedom of the fitted model

In multiple-group analyses, subscripts are used to distinguish independent groups or samples. For example, $N_1, N_2, \ldots, N_r, \ldots, N_k$ denote the sample sizes for k groups. Similarly, notation such as p_r, $\mathbf{S}_r$, $\bar{\mathbf{x}}_r$, $\hat{\boldsymbol{\Sigma}}_r$, $\hat{\boldsymbol{\mu}}_r$, δ_r, and $\mathbf{W}_r$ is used for multiple-group situations.

Residuals in the Moment Matrices

Residuals indicate how well each entry or element in the mean or covariance matrix is fitted. Large residuals indicate bad fit.

PROC CALIS computes four types of residuals and writes them to the OUTSTAT= data set when requested.

- **raw residuals**

  $$s_{ij} - \hat{\sigma}_{ij}, \quad \bar{x}_i - \hat{\mu}_i$$

  for the covariance and mean residuals, respectively. The raw residuals are displayed whenever the PALL, PRINT, or RESIDUAL option is specified.

- **variance standardized residuals**

  $$\frac{s_{ij} - \hat{\sigma}_{ij}}{\sqrt{s_{ii}s_{jj}}}, \quad \frac{\bar{x}_i - \hat{\mu}_i}{\sqrt{s_{ii}}}$$

  for the covariance and mean residuals, respectively. The variance standardized residuals are displayed when you specify one of the following:

  - the PALL, PRINT, or RESIDUAL option and METHOD=NONE, METHOD=ULS, or METHOD=DWLS
  - RESIDUAL=VARSTAND

  The variance standardized residuals are equal to those computed by the EQS 3 program (Bentler 1995).

- **asymptotically standardized residuals**

$$\frac{s_{ij} - \hat{\sigma}_{ij}}{\sqrt{v_{ij,ij}}}, \quad \frac{\bar{x}_i - \hat{\mu}_i}{\sqrt{u_{ii}}}$$

for the covariance and mean residuals, respectively; with

$$v_{ij,ij} = (\hat{\mathbf{\Gamma}}_1 - \mathbf{J}_1 \widehat{\mathrm{Cov}}(\hat{\mathbf{\Theta}}) \mathbf{J}_1')_{ij,ij}$$

$$u_{ii} = (\hat{\mathbf{\Gamma}}_2 - \mathbf{J}_2 \widehat{\mathrm{Cov}}(\hat{\mathbf{\Theta}}) \mathbf{J}_2')_{ii}$$

where $\hat{\mathbf{\Gamma}}_1$ is the $p^2 \times p^2$ estimated asymptotic covariance matrix of sample covariances, $\hat{\mathbf{\Gamma}}_2$ is the $p \times p$ estimated asymptotic covariance matrix of sample means, $\mathbf{J}_1$ is the $p^2 \times t$ Jacobian matrix $d\mathbf{\Sigma}/d\mathbf{\Theta}$, $\mathbf{J}_2$ is the $p \times t$ Jacobian matrix $d\boldsymbol{\mu}/d\mathbf{\Theta}$, and $\widehat{\mathrm{Cov}}(\hat{\mathbf{\Theta}})$ is the $t \times t$ estimated covariance matrix of parameter estimates, all evaluated at the sample moments and estimated parameter values. See the next section for the definitions of $\hat{\mathbf{\Gamma}}_1$ and $\hat{\mathbf{\Gamma}}_2$. Asymptotically standardized residuals are displayed when one of the following conditions is met:

  - The PALL, the PRINT, or the RESIDUAL option is specified, and METHOD=ML, METHOD=GLS, or METHOD=WLS, and the expensive information and Jacobian matrices are computed for some other reason.
  - RESIDUAL= ASYSTAND is specified.

The asymptotically standardized residuals are equal to those computed by the LISREL 7 program (Jöreskog and Sörbom 1988) except for the denominator in the definition of matrix $\hat{\mathbf{\Gamma}}_1$.

- **normalized residuals**

$$\frac{s_{ij} - \hat{\sigma}_{ij}}{\sqrt{(\hat{\mathbf{\Gamma}}_1)_{ij,ij}}}, \quad \frac{\bar{x}_i - \hat{\mu}_i}{\sqrt{(\hat{\mathbf{\Gamma}}_2)_{ii}}}$$

for the covariance and mean residuals, respectively; with $\hat{\mathbf{\Gamma}}_1$ as the $p^2 \times p^2$ estimated asymptotic covariance matrix of sample covariances; and $\hat{\mathbf{\Gamma}}_2$ as the $p \times p$ estimated asymptotic covariance matrix of sample means.

Diagonal elements of $\hat{\mathbf{\Gamma}}_1$ and $\hat{\mathbf{\Gamma}}_2$ are defined for the following methods:

  - **GLS**: $(\hat{\mathbf{\Gamma}}_1)_{ij,ij} = \frac{1}{(N-1)}(s_{ii}s_{jj} + s_{ij}^2)$ and $(\hat{\mathbf{\Gamma}}_2)_{ii} = \frac{1}{(N-1)s_{ii}}$
  - **ML**: $(\hat{\mathbf{\Gamma}}_1)_{ij,ij} = \frac{1}{(N-1)}(\hat{\sigma}_{ii}\hat{\sigma}_{jj} + \hat{\sigma}_{ij}^2)$ and $(\hat{\mathbf{\Gamma}}_2)_{ii} = \frac{1}{(N-1)}\hat{\sigma}_{ii}$
  - **WLS**: $(\hat{\mathbf{\Gamma}}_1)_{ij,ij} = \frac{1}{(N-1)}W_{ij,ij}$ and $(\hat{\mathbf{\Gamma}}_2)_{ii} = \frac{1}{(N-1)s_{ii}}$

where $\mathbf{W}$ in the **WLS** method is the weight matrix for the second-order moments.

Normalized residuals are displayed when one of the following conditions is met:

  - The PALL, PRINT, or RESIDUAL option is specified, and METHOD=ML, METHOD=GLS, or METHOD=WLS, and the expensive information and Jacobian matrices are **not** computed for some other reasons.
  - RESIDUAL=NORM is specified.

The normalized residuals are equal to those computed by the LISREL VI program (Jöreskog and Sörbom 1985) except for the definition of the denominator in computing matrix $\hat{\mathbf{\Gamma}}_1$.

For estimation methods that are not "best" generalized least squares estimators (Browne 1982, 1984), such as METHOD=NONE, METHOD=ULS, or METHOD=DWLS, the assumption of an asymptotic covariance matrix $\mathbf{\Gamma}_1$ of sample covariances does not seem to be appropriate. In this case, the normalized residuals should be replaced by the more relaxed variance standardized residuals. Computation of asymptotically standardized residuals requires computing the Jacobian and information matrices. This is computationally very expensive and is done only if the Jacobian matrix has to be computed for some other reasons—that is, if at least one of the following items is true:

- The default, PRINT, or PALL displayed output is requested, and neither the NOMOD nor NOSTDERR option is specified.
- Either the MODIFICATION (included in PALL), PCOVES, or STDERR (included in default, PRINT, and PALL output) option is requested or RESIDUAL=ASYSTAND is specified.
- The LEVMAR or NEWRAP optimization technique is used.
- An OUTMODEL= data set is specified without using the NOSTDERR option.
- An OUTEST= data set is specified without using the NOSTDERR option.

Since normalized residuals use an overestimate of the asymptotic covariance matrix of residuals (the diagonals of $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$), the normalized residuals cannot be greater than the asymptotically standardized residuals (which use the diagonal of the form $\mathbf{\Gamma} - \mathbf{J}\widehat{\mathrm{Cov}}(\hat{\mathbf{\Theta}})\mathbf{J}'$).

Together with the residual matrices, the values of the average residual, the average off-diagonal residual, and the rank order of the largest values are displayed. The distributions of the normalized and standardized residuals are displayed also.

Overall Model Fit Indices

Instead of assessing the model fit by looking at a number of residuals of the fitted moments, an overall model fit index measures model fit by a single number. Although an overall model fit index is precise and easy to use, there are indeed many choices of overall fit indices. Unfortunately, researchers do not always have a consensus on the best set of indices to use in all occasions.

PROC CALIS produces a large number of overall model fit indices in the fit summary table. If you prefer to display only a subset of these fit indices, you can use the ONLIST(ONLY)= option of the FITINDEX statement to customize the fit summary table.

Fit indices are classified into three classes in the fit summary table of PROC CALIS:

- absolute or standalone Indices
- parsimony indices
- incremental indices

Absolute or Stand-Alone Indices

These indices are constructed so that they measure model fit without comparing with a baseline model and without taking the model complexity into account. They measure the absolute fit of the model.

- **fit function or discrepancy function**
  The fit function or discrepancy function F is minimized during the optimization. See the section "Estimation Criteria" on page 1805 for definitions of various discrepancy functions available in PROC CALIS. For a multiple-group analysis, the fit function can be written as a weighted average of discrepancy functions for k independent groups as:

  $$F = \sum_{r=1}^{k} a_r F_r$$

  where $a_r = \frac{(N_j - 1)}{(N-k)}$ and F_r are the group weight and the discrepancy function for the rth group, respectively. Notice that although the groups are assumed to be independent in the model, in general F_r's are not independent when F is being minimized. The reason is that F_r's might have shared parameters in $\mathbf{\Theta}$ during estimation.

  The minimized function value of F will be denoted as f_{min}, which is always positive, with small values indicating good fit.

- **χ^2 test statistic**
  For ML, GLS, and WLS estimation, the overall χ^2 measure for testing model fit is

  $$\chi^2 = (N - k) * f_{min}$$

  where f_{min} is the function value at the minimum, N is the total sample size, and k is the number of independent groups. The associated degrees of freedom is denoted by d_{min}.

  For ML estimation, this gives the likelihood ratio test statistic of the specified structural model in the null hypothesis against an unconstrained saturated model in the alternative hypothesis. The χ^2 test is valid only if the observations are independent and identically distributed, the analysis is based on the unstandardized sample covariance matrix $\mathbf{S}$, and the sample size N is sufficiently large (Browne 1982; Bollen 1989b; Jöreskog and Sörbom 1985). For ML and GLS estimates, the variables must also have an approximately multivariate normal distribution.

  In the output fit summary table of PROC CALIS, the notation "Prob > Chi-Square" means "the probability of obtaining a greater χ^2 value than the observed value under the null hypothesis." This probability is also known as the p-value of the chi-square test statistic.

- **Satorra-Bentler scaled χ^2 value (Satorra and Bentler 1994)**
  For MLSB estimation, the baseline and target model χ^2 is adjusted by the formula by

  $$\chi^2_{SB} = \frac{\chi^2}{\tau/d}$$

  where d is the degrees of freedom of the baseline or target model and τ is a quantity that must be estimated in practice. Raw data are necessary for computing the estimate of τ. Both d and τ are usually different for the baseline and target models. See Satorra and Bentler (1994) for detailed formulas.

  When you specify METHOD=MLSB, PROC CALIS displays the scaled chi-squares for the baseline and target models. In addition, various fit indices are computed based on the scaled chi-squares instead of the regular versions. If the formulas for the fit indices involve the fit function values of the baseline and target models, the scaled versions of these function values are used instead.

- **adjusted χ^2 value (Browne 1982)**
  If the variables are p-variate elliptic rather than normal and have significant amounts of multivariate kurtosis (leptokurtic or platykurtic), the χ^2 value can be adjusted to

  $$\chi^2_{\text{ell}} = \frac{\chi^2}{\eta_2}$$

  where η_2 is the multivariate relative kurtosis coefficient.

- **Z-test (Wilson and Hilferty 1931)**
  The Z-test of Wilson and Hilferty assumes a p-variate normal distribution,

  $$Z = \frac{\sqrt[3]{\frac{\chi^2}{d}} - (1 - \frac{2}{9d})}{\sqrt{\frac{2}{9d}}}$$

  where d is the degrees of freedom of the model. See McArdle (1988) and Bishop, Fienberg, and Holland (1975, p. 527) for an application of the Z-test.

- **critical N index (Hoelter 1983)**
  The critical N (Hoelter 1983) is defined as

  $$\text{CN} = \text{int}(\frac{\chi^2_{crit}}{f_{min}})$$

  where χ^2_{crit} is the critical chi-square value for the given d degrees of freedom and probability $\alpha = 0.05$, and int() takes the integer part of the expression. See Bollen (1989b, p. 277). Conceptually, the CN value is the largest number of observations that could still make the chi-square model fit statistic insignificant if it were to apply to the actual sample fit function value f_{min}. Hoelter (1983) suggests that CN should be at least 200; however, Bollen (1989b) notes that the CN value might lead to an overly pessimistic assessment of fit for small samples.

  Note that when you have a perfect model fit for your data (that is, $f_{min} = 0$) or a zero degree of freedom for your model (that is, $d = 0$), CN is not computable.

- **root mean square residual (RMR)**
  For a single-group analysis, the RMR is the root of the mean squared residuals:

  $$\text{RMR} = \sqrt{\frac{1}{b}[\sum_i^p \sum_j^i (s_{ij} - \hat{\sigma}_{ij})^2 + \delta \sum_i^p (\bar{x}_i - \hat{\mu}_i)^2]}$$

  where

  $$b = \frac{p(p + 1 + 2\delta)}{2}$$

  is the number of distinct elements in the covariance matrix and in the mean vector (if modeled).

  For multiple-group analysis, PROC CALIS uses the following formula for the overall RMR:

  $$\text{overall RMR} = \sqrt{\sum_{r=1}^k \frac{w_r}{\sum_{r=1}^k w_r}[\sum_i^p \sum_j^i (s_{ij} - \hat{\sigma}_{ij})^2 + \delta \sum_i^p (\bar{x}_i - \hat{\mu}_i)^2]}$$

where

$$w_r = \frac{N_r - 1}{N - k} b_r$$

is the weight for the squared residuals of the rth group. Hence, the weight w_r is the product of group size weight $\frac{N_r-1}{N-k}$ and the number of distinct moments b_r in the rth group.

- **standardized root mean square residual (SRMR)**

  For a single-group analysis, the SRMR is the root of the mean of the standardized squared residuals:

$$\text{SRMR} = \sqrt{\frac{1}{b}[\sum_i^p \sum_j^i \frac{(s_{ij} - \hat{\sigma}_{ij})^2}{s_{ii} s_{jj}} + \delta \sum_i^p \frac{(\bar{x}_i - \hat{\mu}_i)^2}{s_{ii}}]}$$

  where b is the number of distinct elements in the covariance matrix and in the mean vector (if modeled). The formula for b is defined exactly the same way as it appears in the formula for RMR.

  Similar to the calculation of the overall RMR, an overall measure of SRMR in a multiple-group analysis is a weighted average of the standardized squared residuals of the groups. That is,

$$\text{overall SRMR} = \sqrt{\sum_{r=1}^k \frac{w_r}{\sum_{r=1}^k w_r}[\sum_i^p \sum_j^i \frac{(s_{ij} - \hat{\sigma}_{ij})^2}{s_{ii} s_{jj}} + \delta \sum_i^p \frac{(\bar{x}_i - \hat{\mu}_i)^2}{s_{ii}}]}$$

  where w_r is the weight for the squared residuals of the rth group. The formula for w_r is defined exactly the same way as it appears in the formula for SRMR.

- **goodness-of-fit index (GFI)**
  For a single-group analysis, the goodness-of-fit index for the ULS, GLS, and ML estimation methods is:

$$\text{GFI} = 1 - \frac{\text{Tr}((\mathbf{W}^{-1}(\mathbf{S} - \hat{\mathbf{\Sigma}}))^2) + \delta(\bar{\mathbf{x}} - \hat{\boldsymbol{\mu}})'\mathbf{W}^{-1}(\bar{\mathbf{x}} - \hat{\boldsymbol{\mu}})}{\text{Tr}((\mathbf{W}^{-1}\mathbf{S})^2) + \delta\bar{\mathbf{x}}'\mathbf{W}^{-1}\bar{\mathbf{x}}}$$

  with $\mathbf{W} = I$ for ULS, $\mathbf{W} = \mathbf{S}$ for GLS, and $\mathbf{W} = \hat{\mathbf{\Sigma}}$. For WLS and DWLS estimation,

$$\text{GFI} = 1 - \frac{(\mathbf{u} - \hat{\boldsymbol{\eta}})'\mathbf{W}^{-1}(\mathbf{u} - \hat{\boldsymbol{\eta}})}{\mathbf{u}'\mathbf{W}^{-1}\mathbf{u}}$$

  where $\mathbf{u}$ is the vector of observed moments and $\hat{\boldsymbol{\eta}}$ is the vector of fitted moments. When the mean structures are modeled, vectors $\mathbf{u}$ and $\hat{\boldsymbol{\eta}}$ contains all the nonredundant elements $\text{vecs}(\mathbf{S})$ in the covariance matrix and all the means. That is,

$$\mathbf{u} = (\text{vecs}'(\mathbf{S}), \bar{\mathbf{x}}')', \quad \hat{\boldsymbol{\eta}} = (\text{vecs}'(\hat{\mathbf{\Sigma}}), \hat{\boldsymbol{\mu}}')'$$

  and the symmetric weight matrix $\mathbf{W}$ is of dimension $p \times (p + 3)/2$. When the mean structures are not modeled, vectors $\mathbf{u}$ and $\hat{\boldsymbol{\eta}}$ contains all the nonredundant elements $\text{vecs}(\mathbf{S})$ in the covariance matrix only. That is,

$$\mathbf{u} = \text{vecs}(\mathbf{S}), \quad \hat{\boldsymbol{\eta}} = \text{vecs}(\hat{\mathbf{\Sigma}})$$

and the symmetric weight matrix **W** is of dimension $p \times (p+1)/2$. In addition, for the DWLS estimation, **W** is a diagonal matrix.

For a constant weight matrix **W**, the goodness-of-fit index is 1 minus the ratio of the minimum function value and the function value before any model has been fitted. The GFI should be between 0 and 1. The data probably do not fit the model if the GFI is negative or much greater than 1.

For a multiple-group analysis, individual GFI_r's are computed for groups. The overall measure is a weighted average of individual GFI_r's, using weight $a_r = \frac{N_r - 1}{N-k}$. That is,

$$\text{overall GFI} = \sum_{r=1}^{k} a_r \text{GFI}_r$$

Parsimony Indices

These indices are constructed so that the model complexity is taken into account when assessing model fit. In general, models with more parameters (fewer degrees of freedom) are penalized.

- **adjusted goodness-of-fit index (AGFI)**
  The AGFI is the GFI adjusted for the degrees of freedom d of the model,

  $$\text{AGFI} = 1 - \frac{c}{d}(1 - \text{GFI})$$

  where

  $$c = \sum_{r=1}^{k} \frac{p_k(p_k + 1 + 2\delta_k)}{2}$$

  computes the total number of elements in the covariance matrices and mean vectors for modeling. For single-group analyses, the AGFI corresponds to the GFI in replacing the total sum of squares by the mean sum of squares.

  CAUTION:

  - Large p and small d can result in a negative AGFI. For example, GFI=0.90, p=19, and d=2 result in an AGFI of –8.5.
  - AGFI is not defined for a saturated model, due to division by $d = 0$.
  - AGFI is not sensitive to losses in d.

  The AGFI should be between 0 and 1. The data probably do not fit the model if the AGFI is negative or much greater than 1. For more information, see Mulaik et al. (1989).

- **parsimonious goodness-of-fit index (PGFI)**
  The PGFI (Mulaik et al. 1989) is a modification of the GFI that takes the parsimony of the model into account:

  $$\text{PGFI} = \frac{d_{min}}{d_0}\text{GFI}$$

  where d_{min} is the model degrees of freedom and d_0 is the degrees of freedom for the independence model. See the section "Incremental Indices" on page 1833 for the definition of independence model. The PGFI uses the same parsimonious factor as the parsimonious normed Bentler-Bonett index (James, Mulaik, and Brett 1982).

- **RMSEA index (Steiger and Lind 1980; Steiger 1998)**
  The root mean square error of approximation (RMSEA) coefficient is:

  $$\epsilon = \sqrt{k}\sqrt{\max(\frac{f_{min}}{d_{min}} - \frac{1}{(N-k)}, 0)}$$

  The lower and upper limits of the $(1-\alpha)\%$-confidence interval are computed using the cumulative distribution function of the noncentral chi-square distribution $\Phi(x|\lambda, d)$. With $x = (N-k)f_{min}$, λ_L satisfying $\Phi(x|\lambda_L, d_{min}) = 1 - \frac{\alpha}{2}$, and λ_U satisfying $\Phi(x|\lambda_U, d_{min}) = \frac{\alpha}{2}$:

  $$(\epsilon_{\alpha_L}; \epsilon_{\alpha_U}) = (\sqrt{k}\sqrt{\frac{\lambda_L}{(N-k)d_{min}}}; \sqrt{k}\sqrt{\frac{\lambda_U}{(N-k)d_{min}}})$$

  See Browne and Du Toit (1992) for more details. The size of the confidence interval can be set by the option ALPHARMS=α, $0 \le \alpha \le 1$. The default is $\alpha = 0.1$, which corresponds to the 90% confidence interval for the RMSEA.

- **probability for test of close fit (Browne and Cudeck 1993)**
  The traditional exact χ^2 test hypothesis $H_0: \epsilon = 0$ is replaced by the null hypothesis of close fit $H_0: \epsilon \le 0.05$ and the exceedance probability P is computed as:

  $$P = 1 - \Phi(x|\lambda^*, d_{min})$$

  where $x = (N-k)f_{min}$ and $\lambda^* = 0.05^2(N-k)d_{min}/k$. The null hypothesis of close fit is rejected if P is smaller than a prespecified level (for example, $P < 0.05$).

- **ECVI: expected cross-validation index (Browne and Cudeck 1993)**
  The following formulas for ECVI are limited to the case of single-sample analysis without mean structures and with either the GLS, ML, or WLS estimation method. For other cases, ECVI is not defined in PROC CALIS. For GLS and WLS, the estimator c of the ECVI is linearly related to AIC, Akaike's Information Criterion (Akaike 1974, 1987):

  $$c = f_{min} + \frac{2t}{N-1}$$

  For ML estimation, c_{ML} is used:

  $$c_{ML} = f_{min} + \frac{2t}{N-p-2}$$

  For GLS and WLS, the confidence interval $(c_L; c_U)$ for ECVI is computed using the cumulative distribution function $\Phi(x|\lambda, d_{min})$ of the noncentral chi-square distribution,

  $$(c_L; c_U) = (\frac{\lambda_L + p(p+1)/2 + t}{(N-1)}; \frac{\lambda_U + p(p+1)/2 + t}{(N-1)})$$

  with $x = (N-1)f_{min}$, $\Phi(x|\lambda_U, d_{min}) = \frac{\alpha}{2}$, and $\Phi(x|\lambda_L, d_{min}) = 1 - \frac{\alpha}{2}$.
  For ML, the confidence interval $(c_L^*; c_U^*)$ for ECVI is:

  $$(c_L^*; c_U^*) = (\frac{\lambda_L^* + p(p+1)/2 + t}{N-p-2}; \frac{\lambda_U^* + p(p+1)/2 + t}{N-p-2})$$

  where $x = (N-p-2)f_{min}$, $\Phi(x|\lambda_U^*, d_{min}) = \frac{\alpha}{2}$ and $\Phi(x|\lambda_L^*, d_{min}) = 1 - \frac{\alpha}{2}$. See Browne and Cudeck (1993). The size of the confidence interval can be set by the option ALPHAECV=α, $0 \le \alpha \le 1$. The default is $\alpha = 0.1$, which corresponds to the 90% confidence interval for the ECVI.

- **Akaike's information criterion (AIC) (Akaike 1974, 1987)**
  This is a criterion for selecting the best model among a number of candidate models. The model that yields the smallest value of AIC is considered the best.

  $$\text{AIC} = h + 2t$$

  where h is the –2 times the likelihood function value for the FIML method or the χ^2 value for other estimation methods.

- **consistent Akaike's information criterion (CAIC) (Bozdogan 1987)**
  This is another criterion, similar to AIC, for selecting the best model among alternatives. The model that yields the smallest value of CAIC is considered the best. CAIC is preferred by some people to AIC or the χ^2 test.

  $$\text{CAIC} = h + (\ln(N) + 1)t$$

  where h is the –2 times the likelihood function value for the FIML method or the χ^2 value for other estimation methods. Notice that N includes the number of incomplete observations for the FIML method while it includes only the complete observations for other estimation methods.

- **Schwarz's Bayesian criterion (SBC) (Schwarz 1978; Sclove 1987)**
  This is another criterion, similar to AIC, for selecting the best model. The model that yields the smallest value of SBC is considered the best. SBC is preferred by some people to AIC or the χ^2 test.

  $$\text{SBC} = h + \ln(N)t$$

  where h is the –2 times the likelihood function value for the FIML method or the χ^2 value for other estimation methods. Notice that N includes the number of incomplete observations for the FIML method while it includes only the complete observations for other estimation methods.

- **McDonald's measure of centrality (McDonald and Marsh 1988)**

  $$\text{CENT} = \exp(-\frac{(\chi^2 - d_{min})}{2N})$$

Incremental Indices

These indices are constructed so that the model fit is assessed through the comparison with a baseline model. The baseline model is usually the independence model where all covariances among manifest variables are assumed to be zeros. The only parameters in the independence model are the diagonals of covariance matrix. If modeled, the mean structures are saturated in the independence model. For multiple-group analysis, the overall independence model consists of component independence models for each group.

In the following, let f_0 and d_0 denote the minimized discrepancy function value and the associated degrees of freedom, respectively, for the independence model; and f_{min} and d_{min} denote the minimized discrepancy function value and the associated degrees of freedom, respectively, for the model being fitted in the null hypothesis.

- **Bentler comparative fit index (Bentler 1995)**

  $$\text{CFI} = 1 - \frac{\max((N-k)f_{min} - d_{min}, 0)}{\max((N-k)f_{min} - d_{min}, \max((N-k)f_0 - d_0, 0)}$$

- **Bentler-Bonett normed fit index (NFI) (Bentler and Bonett 1980)**

  $$\Delta = \frac{f_0 - f_{min}}{f_0}$$

  Mulaik et al. (1989) recommend the parsimonious weighted form called parsimonious normed fit index (PNFI) (James, Mulaik, and Brett 1982).

- **Bentler-Bonett nonnormed coefficient (Bentler and Bonett 1980)**

  $$\rho = \frac{f_0/d_0 - f_{min}/d_{min}}{f_0/d_0 - 1/(N-k)}$$

  See Tucker and Lewis (1973).

- **normed index ρ_1 (Bollen 1986)**

  $$\rho_1 = \frac{f_0/d_0 - f_{min}/d_{min}}{f_0/d_0}$$

  ρ_1 is always less than or equal to 1; $\rho_1 < 0$ is unlikely in practice. See the discussion in Bollen (1989a).

- **nonnormed index Δ_2 (Bollen 1989a)**

  $$\Delta_2 = \frac{f_0 - f_{min}}{f_0 - \frac{d_{min}}{(N-k)}}$$

  is a modification of Bentler and Bonett's Δ that uses d and "lessens the dependence" on N. See the discussion in (Bollen 1989b). Δ_2 is identical to the IFI2 index of Mulaik et al. (1989).

- **parsimonious normed fit index (James, Mulaik, and Brett 1982)**
  The PNFI is a modification of Bentler-Bonett's normed fit index that takes parsimony of the model into account,

  $$\text{PNFI} = \frac{d_{min}}{d_0}\frac{(f_0 - f_{min})}{f_0}$$

  The PNFI uses the same parsimonious factor as the parsimonious GFI of Mulaik et al. (1989).

Fit Indices and Estimation Methods

Note that not all fit indices are reasonable or appropriate for all estimation methods set by the METHOD= option of the PROC CALIS statement. The availability of fit indices is summarized as follows:

- Adjusted (elliptic) chi-square and its probability are available only for METHOD=ML or GLS and with the presence of raw data input.

- For METHOD=ULS or DWLS, probability of the chi-square value, RMSEA and its confidence intervals, probability of close fit, ECVI and its confidence intervals, critical N index, Z-test, AIC, CAIC, SBC, and measure of centrality are not appropriate and therefore not displayed.

Individual Fit Indices for Multiple Groups

When you compare the fits of individual groups in a multiple-group analysis, you can examine the residuals of the groups to gauge which group is fitted better than the others. While examining residuals is good for knowing specific locations with inadequate fit, summary measures like fit indices for individual groups would be more convenient for overall comparisons among groups.

Although the overall fit function is a weighted sum of individual fit functions for groups, these individual functions are not statistically independent. Therefore, in general you cannot partition the degrees of freedom or χ^2 value according to the groups. This eliminates the possibility of breaking down those fit indices that are functions of degrees of freedom or χ^2 for group comparison purposes. Bearing this fact in mind, PROC CALIS computes only a limited number of descriptive fit indices for individual groups.

- **fit function**
  The overall fit function is:

  $$F = \sum_{r=1}^{k} a_r F_r$$

  where $a_r = \frac{(N_j - 1)}{(N-k)}$ and F_r are the group weight and the discrepancy function for group *r*, respectively. The value of unweighted fit function F_r for the *r*th group is denoted by:

  $$f_r$$

  This f_r value provides a measure of fit in the *r*th group without taking the sample size into account. The large the f_r, the worse the fit for the group.

- **percentage contribution to the chi-square**
  The percentage contribution of group *r* to the chi-square is:

  $$\text{percentage contribution} = a_r f_r / f_{min} \times 100\%$$

  where f_r is the value of F_r with F minimized at the value f_{min}. This percentage value provides a descriptive measure of fit of the moments in group *r*, weighted by its sample size. The group with the largest percentage contribution accounts for the most lack of fit in the overall model.

- **root mean square residual (RMR)**
  For the *r*th group, the total number of moments being modeled is:

  $$g = \frac{p_r(p_r + 1 + 2\delta_r)}{2}$$

  where p_r is the number of variables and δ_r is the indicator variable of the mean structures in the *r*th group. The root mean square residual for the *r*th group is:

  $$\text{RMR}_r = \sqrt{\frac{1}{g}[\sum_{i}^{p_r}\sum_{j}^{i}([\mathbf{S}_r]_{ij} - [\hat{\boldsymbol{\Sigma}}_r]_{ij})^2 + \delta_r \sum_{i}^{p_r}([\bar{\mathbf{x}}_r]_i - [\hat{\boldsymbol{\mu}}_r]_i)^2]}$$

- **standardized root mean square residual (SRMR)**
  For the *r*th group, the standardized root mean square residual is:

  $$\text{SRMR} = \sqrt{\frac{1}{g}[\sum_{i}^{p_r}\sum_{j}^{i}\frac{([\mathbf{S}_r]_{ij} - [\hat{\boldsymbol{\Sigma}}_r]_{ij})^2}{[\mathbf{S}_r]_{ii}[\mathbf{S}_r]_{jj}} + \delta_r \sum_{i}^{p_r}\frac{([\bar{\mathbf{x}}_r]_i - [\hat{\boldsymbol{\mu}}_r]_i)^2}{[\mathbf{S}_r]_{ii}}]}$$

- **goodness-of-fit index (GFI)**
  For the ULS, GLS, and ML estimation, the goodness-of-fit index (GFI) for the rth group is:

$$\mathrm{GFI} = 1 - \frac{\mathrm{Tr}((\mathbf{W}_r^{-1}(\mathbf{S}_r - \hat{\mathbf{\Sigma}}_r))^2) + \delta_r(\bar{\mathbf{x}}_r - \hat{\mathbf{u}}_r)'\mathbf{W}_r^{-1}(\bar{\mathbf{x}}_r - \hat{\mathbf{u}}_r)}{\mathrm{Tr}((\mathbf{W}_r^{-1}\mathbf{S}_r)^2) + \delta_r\bar{\mathbf{x}}_r'\mathbf{W}_r^{-1}\bar{\mathbf{x}}_r}$$

  with $\mathbf{W}_r = I$ for ULS, $\mathbf{W}_r = \mathbf{S}_r$ for GLS, and $\mathbf{W}_r = \hat{\mathbf{\Sigma}}_r$. For the WLS and DWLS estimation,

$$\mathrm{GFI} = 1 - \frac{(\mathbf{u}_r - \hat{\boldsymbol{\eta}}_r)'\mathbf{W}_r^{-1}(\mathbf{u}_r - \hat{\boldsymbol{\eta}}_r)}{\mathbf{u}_r'\mathbf{W}_r^{-1}\mathbf{u}_r}$$

  where $\mathbf{u}_r$ is the vector of observed moments and $\hat{\boldsymbol{\eta}}_r$ is the vector of fitted moments for the rth group ($r = 1, \ldots, k$).

  When the mean structures are modeled, vectors $\mathbf{u}_r$ and $\hat{\boldsymbol{\eta}}_r$ contain all the nonredundant elements $\mathrm{vecs}(\mathbf{S}_r)$ in the covariance matrix and all the means, and $\mathbf{W}_r$ is the weight matrix for covariances and means. When the mean structures are not modeled, $\mathbf{u}_r$, $\hat{\boldsymbol{\eta}}_r$, and $\mathbf{W}_r$ contain elements pertaining to the covariance elements only. Basically, formulas presented here are the same as the case for a single-group GFI. The only thing added here is the subscript r to denote individual group measures.

- **Bentler-Bonett normed fit index (NFI)**
  For the rth group, the Bentler-Bonett NFI is:

$$\Delta_r = \frac{f_{0r} - f_r}{f_{0r}}$$

  where f_{0r} is the function value for fitting the independence model to the rth group. The larger the value of Δ_r, the better is the fit for the group. Basically, the formula here is the same as the overall Bentler-Bonett NFI. The only difference is that the subscript r is added to denote individual group measures.

Squared Multiple Correlations and Determination Coefficients

In the section, squared multiple correlations for endogenous variables are defined. Squared multiple correlation is computed for all of these five estimation methods: ULS, GLS, ML, WLS, and DWLS. These coefficients are also computed as in the LISREL VI program of Jöreskog and Sörbom (1985). The DETAE, DETSE, and DETMV determination coefficients are intended to be multivariate generalizations of the squared multiple correlations for different subsets of variables. These coefficients are displayed only when you specify the PDETERM option.

- **$\mathbf{R}^2$ values corresponding to endogenous variables**

$$R^2 = 1 - \frac{\widehat{\mathrm{Evar}}(y)}{\widehat{\mathrm{Var}}(y)}$$

  where y denotes an endogenous variable, $\widehat{\mathrm{Var}}(y)$ denotes its variance, and $\widehat{\mathrm{Evar}}(y)$ denotes its error (or unsystematic) variance. The variance and error variance are estimated under the model.

- **total determination of all equations**

$$\text{DETAE} = 1 - \frac{|\widehat{\text{Ecov}}(\mathbf{y}, \boldsymbol{\eta})|}{|\widehat{\text{Cov}}(\mathbf{y}, \boldsymbol{\eta})|}$$

  where the $\mathbf{y}$ vector denotes all manifest dependent variables, the $\boldsymbol{\eta}$ vector denotes all latent dependent variables, $\widehat{\text{Cov}}(\mathbf{y}, \boldsymbol{\eta})$ denotes the covariance matrix of $\mathbf{y}$ and $\boldsymbol{\eta}$, and $\widehat{\text{Ecov}}(\mathbf{y}, \boldsymbol{\eta})$ denotes the error covariance matrix of $\mathbf{y}$ and $\boldsymbol{\eta}$. The covariance matrices are estimated under the model.

- **total determination of latent equations**

$$\text{DETSE} = 1 - \frac{|\widehat{\text{Ecov}}(\boldsymbol{\eta})|}{|\widehat{\text{Cov}}(\boldsymbol{\eta})|}$$

  where the $\boldsymbol{\eta}$ vector denotes all latent dependent variables, $\widehat{\text{Cov}}(\boldsymbol{\eta})$ denotes the covariance matrix of $\boldsymbol{\eta}$, and $\widehat{\text{Ecov}}(\boldsymbol{\eta})$ denotes the error covariance matrix of $\boldsymbol{\eta}$. The covariance matrices are estimated under the model.

- **total determination of the manifest equations**

$$\text{DETMV} = 1 - \frac{|\widehat{\text{Ecov}}(\mathbf{y})|}{|\widehat{\text{Cov}}(\mathbf{y})|}$$

  where the $\mathbf{y}$ vector denotes all manifest dependent variables, $\widehat{\text{Cov}}(\mathbf{y})$ denotes the covariance matrix of $\mathbf{y}$, $\widehat{\text{Ecov}}(\mathbf{y})$ denotes the error covariance matrix of $\mathbf{y}$, and $|\mathbf{A}|$ denotes the determinant of matrix $\mathbf{A}$. All the covariance matrices in the formula are estimated under the model.

You can also use the DETERM statement to request the computations of determination coefficients for any subsets of dependent variables.

Case-Level Residuals, Outliers, Leverage Observations, and Residual Diagnostics

Residual M-Distances and Outlier Detection

In structural equation modeling, residual analysis has been developed traditionally for the elements in the moment matrices such as mean vector and covariance matrix. See the section "Residuals in the Moment Matrices" on page 1825 for a detailed description of this type of residual. In fact, almost all fit indices in the field were developed more or less based on measuring the overall magnitude of the residuals in the moment matrices. See the section "Assessment of Fit" on page 1824 for examples of fit indices such as standardized root mean square residual (SRMR), adjusted goodness-of-fit index (AGFI), and so on.

However, recent research advances make it possible to study case-level or observational-level residuals. Although case-level residuals have not yet been used to assess overall model fit, they are quite useful as model diagnostic tools. This section describes how case-level residuals are computed and applied to the detection of outliers, the identification of leverage observations, and various residual diagnostics.

The main difficulty of case-level residual analysis in general structural equation modeling is the presence of latent variables in the model. Another difficulty is the involvement of multivariate responses. Both are

nonissues in multiple regression analysis because there is no latent variable and there is only a single response variable. Hence, a good starting point for this section is to explain how PROC CALIS handles these two issues in general structural equation modeling. The residual diagnostic techniques described in the following draw heavily from Yuan and Hayashi (2010).

Latent variables in models are not observed, and their values must be predicted from the data. Although the indeterminacy of factor scores is a well-known issue, it does not mean that factor scores cannot be reasonably derived or predicted from the observed data. One of the factor score estimation methods is Bartlett's method. Suppose for a factor model that Λ represents a $p \times m$ factor matrix and Ψ represents a $p \times p$ error covariance matrix, where p is the number of manifest variables and m is the number of factors. Bartlett's factor scores are defined by the formula

$$\mathbf{f} = (\Lambda'\Psi^{-1}\Lambda)^{-1}\Lambda'\Psi^{-1}\mathbf{y}$$

where $\mathbf{f}$ represents an $m \times 1$ random vector of factors and $\mathbf{y}$ represents a $p \times 1$ random vector of manifest variables. Yuan and Hayashi (2010) generalize Bartlett's formulas to a certain class of structural equation models. Essentially, they provide formulas that the quantities Λ and Ψ are computable functions from the parameter estimates of the structural equation models. PROC CALIS adopts their formulas with an extension due to Yung and Yuan (2013). With the estimation of factor scores, residuals in structural equation models can be computed as if all values of factors are present. Hence, case-level residual analysis becomes possible.

With the possibility of a multivariate residual vector in structural equation models, a summary measure for the overall magnitude of the residuals in observations is needed. Yuan and Hayashi (2010) consider the Mahalanobis distance (M-distance) of case-level residuals. For observation i with residual vector $\hat{e}_i$ of dimension $h \times 1$, the residual M-distance of observation i is defined as

$$d_{\mathbf{r_i}} = \sqrt{(\mathbf{L}\hat{\mathbf{e}}_\mathbf{i})'(\mathbf{L}\Omega_{\hat{\mathbf{e}}}\mathbf{L}')^{-1}(\mathbf{L}\hat{\mathbf{e}}_\mathbf{i})}$$

where $\mathbf{L}$ ($(h-q) \times h$) is a matrix that reduces $\hat{\mathbf{e}}$ to $(h-q)$ independent components, q is the number of independent factors to be estimated in the structural equation model, and $\Omega_{\hat{\mathbf{e}}}$ is the covariance matrix of $\hat{\mathbf{e}}$. The reduction of the residual vector into independent components is necessary when the number of factors q is not zero in the model. For $q > 0$, $\Omega_{\hat{\mathbf{e}}}$ is not invertible and cannot be used in computing the M-distances for residuals. Hence, the covariance matrix $\mathbf{L}\Omega_{\hat{\mathbf{e}}}\mathbf{L}'$ of independent components $\mathbf{L}\hat{\mathbf{e}}$ is used instead. In practice, $\hat{\mathbf{e}}_\mathbf{i}$ and $\Omega_{\hat{\mathbf{e}}}$ are evaluated at the estimated parameter values. $\mathbf{L}$ can be computed after $\Omega_{\hat{\mathbf{e}}}$ is estimated.

Because the residual M-distance $d_{\mathbf{r_i}}$ is nonnegative and is a distance measure, the interpretation of $d_{\mathbf{r_i}}$ is straightforward—the larger it is, the farther away observation i is from the origin of the residuals. Therefore, the residual M-distance $d_{\mathbf{r_i}}$ can be used to detect outliers. The criterion for defining outliers is based on a selection of an α-level for the residual M-distance distribution. Under the multivariate normality distribution of the residuals, Yuan and Hayashi (2010) find that $d_{\mathbf{r_i}}$ is distributed as a χ_r variate (that is, the square root of the corresponding χ^2_r variate), where $r = h - q$ is the degrees of freedom of the distribution. Therefore, a probability-based criterion for outlier detection can be used. An observation is an outlier according to the α-level criterion if its residual M-distance $d_{\mathbf{r_i}}$ is located within the upper $\alpha \times 100\%$ region of the χ_r distribution. Mathematically, an observation with $d_{\mathbf{r_i}}$ satisfying the following criterion is an outlier if

$$\text{prob}(\chi_r \geq d_{\mathbf{r_i}}) < \alpha$$

The larger the α, the more liberal the criterion for detecting outliers. By default, PROC CALIS sets α to 0.01. You can override this value by using the ALPHAOUT= option.

Leverage Observations

Leverage observations are those with large M-distances in the exogenous variables of the model. For example, in the context of a confirmatory factor model, Yuan and Hayashi (2010) define the M-distance of exogenous latent factors $d_{\mathbf{f_i}}$ by the formula

$$d_{\mathbf{f_i}} = \sqrt{\mathbf{f_i}'(\Omega_\mathbf{f})^{-1}\mathbf{f_i}}$$

where $\mathbf{f_i}$ is the Bartlett's factor scores for observation i and $\Omega_\mathbf{f}$ is the covariance matrix of the factor scores. See Yuan and Hayashi (2010) for the formulas for $\mathbf{f_i}$ and $\Omega_\mathbf{f}$. PROC CALIS uses a more general formula to compute the M-distance of exogenous observed variables $\mathbf{x_i}$ and latent variables $\mathbf{f_i}$ in structural equation models. That is, with exogenous variables $\mathbf{v_i} = (\mathbf{x_i}\ \mathbf{f_i})'$, the leverage M-distance $d_{\mathbf{v_i}}$ is computed as

$$d_{\mathbf{v_i}} = \sqrt{\mathbf{v_i}'(\Omega_\mathbf{v})^{-1}\mathbf{v_i}}$$

where $\Omega_\mathbf{v}$ is the covariance matrix of the exogenous variables. Leverage observations are those judged to have large leverage M-distances.

As with outlier detection, you need to set a criterion for declaring leverage observations. Specifically, an observation with $d_{\mathbf{v_i}}$ satisfying the following criterion is a leverage observation (or leverage point):

$$\text{prob}(\chi_c \geq d_{\mathbf{v_i}}) < \alpha$$

where χ_c is the square root of the χ^2_c variate, with c being the dimension of $\mathbf{v_i}$. The larger the α, the more liberal the criterion for declaring leverage observations. The default α value is 0.01. You can override this value by using the ALPHALEV= option.

Unlike the outliers that are usually judged as undesirable in models, leverage observations might or might not be bad for model estimation. Simply stated, a leverage observation is bad if it is also an outlier according to its residual M-distance value $d_{\mathbf{r_i}}$. Otherwise, it is a "good" leverage observation and should not be removed in model estimation.

Residual Diagnostics

Case-level residuals analysis can be very useful in detecting outliers, anomalies in residual distributions, and nonlinear functional relationships. All these techniques are graphical in nature and can be found in most textbooks in regression analysis (see, for example, Belsley, Kuh, and Welsch 1980). PROC CALIS provides these graphical analyses in the context of structural equation modeling. However, because PROC CALIS deals with multivariate responses and predictors in general, in some residual analysis the residuals are measured in terms of M-distances, which are always positive. PROC CALIS can produce the following graphical plots (with their ODS graph names in parentheses) for residual diagnostics:

- Residual histogram (CaseResidualHistogram): the distribution of residual M-distances is shown as a histogram. The residual M-distances are theoretically distributed as a χ-variate. Use the PLOTS=CRESHIST option to request this plot.
- Residual by leverage plot (ResidualByLeverage): the observations are plotted in a two-dimensional space according to their M-distances for residuals and leverages. Criteria for classifying outliers and leverage observations are shown in the plot so that outliers and leverage observations are located in well-defined regions of the two-dimensional space. Use the PLOTS=RESBYLEV option to request this plot.

- Residual by predicted values (ResidualByPredicted): the residuals (not the M-distances) of a dependent observed variable are plotted against the predicted values. This is also called the residual on fit plot. If the relationship of the dependent variable to the predictors is truly linear in the model, the residuals should be randomly dispersed in the plot. If the residuals show some systematic pattern with the predicted values, unexplained nonlinear relationships of the dependent observed variable with other variables or heteroscedasticity of error variance might exist. Use the PLOTS=RESBYPRED option to request this plot.
- Residual by quantile plot or Q-Q plot (ResidualByQuantile): the residual M-distances are plotted against their theoretical quantiles. Observations that depart significantly from the theoretical line (with slope=1) are problematic observations. Use the PLOTS=QQPLOT option to request this plot.
- Residual percentile by theoretical percentile plot or P-P plot (ResPercentileByExpPercentile): the observed residual M-distance percentiles are plotted against the theoretical percentiles. Observations that depart significantly from the theoretical line (with slope=1) are problematic observations. Use the PLOTS=PPPLOT option to request this plot.

Residual Diagnostics with Robust Estimation

With the presence of multiple outliers in a data set, estimation might suffer from the so-called masking effect. This problem refers to the fact that some prominent outliers in the data might have biased the estimation so that usual residual diagnostic tools might fail to detect some less prominent outliers. As a result, subsequent analyses that are based on the data set with outliers removed might actually not be free from the effects of the outliers.

Robust estimation deals with the outliers differently. Instead of relying on diagnostic tools for detection and removal of outliers, robust methods downweight the outliers during the estimation so that the undesirable outlier effect on estimation is minimized effectively. No outliers need to be removed during or after the robust estimation. Hence, the masking effect is not an issue. In addition, with the use of robust estimation, residual diagnostics for the purpose of outlier removal become a redundant concept. In such a scenario, outlier diagnostics are more important for the purpose of identification than for removal. You can use the ROBUST=RES option (or the default ROBUST option) to downweight the outliers during model estimation.

Similarly, detecting leverage observations might also suffer from the masking effect. That is, some leverage observations might not be detected if the diagnostic results are based on model estimation that includes the leverage observations themselves. Unfortunately, the ROBUST=RES option downweights only the residual outliers and does not downweight the leverage observations during estimation. However, the ROBUST=SAT option does downweight outlying observations in all variable dimensions, and hence it could be used if the masking effect of leverage observations is a concern.

Following the recommendation by Yuan and Hayashi (2010), PROC CALIS uses an eclectic approach to compute the residual and leverage M-distances. That is, when a robust estimation is requested (using either the ROBUST=SAT or ROBUST=RES option) with case-level residual analysis (using the RESIDUAL option), the residual M-distances are computed from the robust estimation results that downweight the model outliers only (that is, the procedure invoked by the ROBUST=RES(E) or RES(F) option); but the leverage M-distances are computed from the robust estimation results that downweight the outlying observations in all variable dimensions (that is, the procedure invoked by the ROBUST=SAT option). Consequently, both the residual M-distances and the leverage M-distances calculated from this eclectic approach are free from the respective masking effects—and this is true for any robust estimation methods implemented in PROC CALIS.

It is important to note that this eclectic approach applies only to the computations of the residual and leverage M-distances. The parameter estimates, fit statistics, and the case-level residuals that are displayed in the residual on fit plots are still obtained from the specific robust method requested. Table 33.6 summarizes the robust methods that PROC CALIS uses to compute the parameter estimates, fit statistics, case-level residuals, and residual and leverage M-distances with different robust options.

Table 33.6 Robust Methods for the Computations of Various Results

	Robust Option Specified		
	ROBUST=SAT	**ROBUST=RES(E)**	**ROBUST=RES(F)**
Estimates	ROBUST=SAT	ROBUST=RES(E)	ROBUST=RES(F)
Model fit	ROBUST=SAT	ROBUST=RES(E)	ROBUST=RES(F)
Case-level residuals	ROBUST=SAT	ROBUST=RES(E)	ROBUST=RES(F)
Residual M-distances	ROBUST=RES(E)	ROBUST=RES(E)	ROBUST=RES(F)
Leverage M-distances	ROBUST=SAT	ROBUST=SAT	ROBUST=SAT

One important implication from this table is that ROBUST=SAT and ROBUST=RES(E) lead to the same residual and leverage diagnostic results, which are all free from masking effects.

In conclusion, even though outlier removal is unnecessary with robust estimation, case-level residual diagnostics are still useful for identifying outliers and leverage observations. Needless to say, other residual diagnostics such as residual M-distance distribution, Q-Q plots, P-P plots, and residual on fit plots are useful whether you use robust estimation or not.

Latent Variable Scores

Analysis of Covariance Structures Only

Although latent variable or factor scores are unobserved, you can estimate them after you use the CALIS procedure to fit your model. If your model contains covariance structures only, then latent variable or factor scores are estimated as linear combinations of observed variables, weighted by the latent variable (factor) score regression coefficients. This section covers the case of covariance structures. The next section covers the case of mean and covariance structures.

You can use the PLATCOV option to display the latent variable (factor) score regression coefficients. You can also save these coefficients in an output data set by using the OUTSTAT= option. You can then provide these coefficients to the SCORE procedure to compute the latent variable (factor) scores in a data set.

To summarize, follow these steps to compute latent variable (factor) scores for each observation:

- Create an output data set by using the OUTSTAT= option in the PROC CALIS statement.
- Run the SCORE procedure, using both the raw data and the OUTSTAT= data set.

For example, you can use the following statements to compute the latent variable (factor) scores, which are stored in the OUTSTAT= data set named ostat:

```
proc calis data=raw outstat=ostat;
   lineqs
      v1 = a1 f1 + e1,
      v2 = a2 f1 + e2,
      v3 = a3 f1 + e3;
   std
      f1 = 1.,
      e1-e3 = evar1-evar3;
run;
```

Then, in the PROC SCORE statement, you specify the raw data set in the DATA= option and the ostat data set in the SCORE= option:

```
proc score data=raw score=ostat out=scores;
   var v1-v3;
run;
```

The data set that you specify in the OUT= option stores the latent variable (factor) scores.

Although you can get the latent variable (factor) score regression coefficients by analyzing either raw data or covariance (correlation) matrices in PROC CALIS, you must provide the raw data to the SCORE procedure in order to compute latent variable (factor) scores. For a more detailed example, see Example 109.1 in Chapter 109, "The SCORE Procedure." Although that example uses PROC FACTOR, the scoring procedure that is demonstrated in the example is also applicable to PROC CALIS. For the conceptual differences of factor scores between the FACTOR and CALIS procedures, see the section "Factor Scores in PROC FACTOR and PROC CALIS" on page 1844.

Analysis of Mean and Covariance Structures

When you model the mean and covariance structures simultaneously, the computation of latent variable (factor) scores is not as straightforward as the case where you analyze covariance structures only. However, if the mean structures in your model are saturated, you can still use the steps that are described in the preceding section. For example, if you specify the MEANSTR option and do not specify any other mean parameters or constraints on means, then the mean structures in your model are saturated with default parameters. Another example is when you specify METHOD=FIML but do not specify any mean parameters or mean constraints. In this case, PROC CALIS adds default mean parameters to your model so that the mean structures are also saturated.

If the mean structures in your models are not saturated, you need to adjust the OUTSTAT= data set that contains the scoring coefficients before submitting it to the SCORE procedure to compute latent variable (factor) scores. For example, your OUTSTAT= data set for a mean and covariance structure model might look like the following:

OBS	_TYPE_	_NAME_	x1	x2	x3	f1
1	OBSERVED		1.0000	1.0000	1.0000	0.00000
2	MEAN		0.8674	0.9787	1.0114	.
3	SKEWNESS		-0.1240	-0.1181	-0.8996	.
4	KURTOSIS		-0.2892	0.4518	1.1604	.
5	N		50.0000	50.0000	50.0000	.
6	SUMWGT		50.0000	50.0000	50.0000	.
7	VARDIV		49.0000	49.0000	49.0000	.

8	COV	x1	1.4845	1.1501	1.3554	.
9	COV	x2	1.1501	1.5138	1.4084	.
10	COV	x3	1.3554	1.4084	1.9416	.
11	MAXRES	_Mean_	-0.0314	0.0384	-0.0118	.
12	MAXRES	x1	0.0318	-0.0039	0.0319	.
13	MAXRES	x2	-0.0039	-0.0448	-0.0204	.
14	MAXRES	x3	0.0319	-0.0204	0.0151	.
15	MAXASRES	_Mean_	-0.7066	0.7065	-0.7065	.
16	MAXASRES	x1	0.6853	-0.5405	0.6984	.
17	MAXASRES	x2	-0.5405	-0.7306	-0.6912	.
18	MAXASRES	x3	0.6984	-0.6912	0.7000	.
19	MAXPRED	_Mean_	0.8987	0.9403	1.0232	0.50558
20	MAXPRED	x1	1.4527	1.1540	1.3236	1.03392
21	MAXPRED	x2	1.1540	1.5586	1.4288	1.11613
22	MAXPRED	x3	1.3236	1.4288	1.9265	1.28014
23	MAXPRED	f1	1.0339	1.1161	1.2801	1.00000
24	SCORE	f1	0.2001	0.2650	0.3305	.

Observation 24 in this OUTSTAT= data set contains the scoring coefficients for computing the factor scores of f1. Observation 2 contains the sample means of the observed variables x1–x3. These are the means that the SCORE procedure uses to compute the deviation scores of the observed variables (before the deviation scores are multiplied by the scoring coefficients). However, when your model contains mean structures, you should use the predicted variable means instead of the observed variable means to compute the deviation scores. In the OUTSTAT= data set, the predicted means are stored in the observation that has _TYPE_=MAXPRED and _NAME_=_Mean_ (observation 19). In order to "trick" the SCORE procedure into using the predicted means, you can use the following statements:

```
data ostat2;
   set ostat;
   if _TYPE_='MEAN' then _TYPE_='OBSMEAN';
   if _TYPE_='MAXPRED' & _NAME_='_Mean_' then _TYPE_='MEAN';
run;

proc score data=raw score=ostat2 out=Scores2;
   var v1-v3;
run;
```

The DATA step creates the data set ostat2 by copying the original OUTSTAT= data set, ostat, but it also change the value of the _TYPE_ variable in observation 19 so that it becomes

19	MEAN	_Mean_	0.8987	0.9403	1.0232	0.50558

The SCORE procedure then uses the means in this observation to compute the deviation scores and hence the latent variable (factor) scores. It saves the scores in a new OUT= data set, Scores2. If the means of the factors are not important in subsequent analyses of the factor scores (for example, exploratory factor analysis), you can use the latent variable (factor) scores in the Scores2 data set without further processing.

However, if the factor means are important, you need to do perform one more step, as shown in the following statements:

```
data Scores3;
   set Scores2;
   f1  =  f1 + 0.50558;
run;
```

The data set Scores3 is essentially a copy of the Scores2 data set, but it adds the constant 0.50558 to the factor variable f1. This constant value is the predicted factor mean of f1, as you can see from observation 19 in the original OUTSTAT= data set. Before this constant is added, the expected value of f1 is 0. However, the new score data set, Scores3, changes this expected value to 0.50558, which is consistent with the original model results. If you have more latent variable (factors) in your model, you need to add the predicted mean for each of the latent variables (factors) when you create the target data set for the latent variable (factor) scores.

Factor Scores in PROC FACTOR and PROC CALIS

Conceptually, the scoring coefficients that you obtain from PROC FACTOR and PROC CALIS are different in their scoring applications. The scoring coefficients that PROC FACTOR produces are applied to observed variables in the standardized form (with mean 0 and standard deviation 1), whereas the scoring coefficients that PROC CALIS produces are applied to observed variables in the deviation form (with mean 0). By default, PROC SCORE uses the standardized variables, so it is compatible with the scoring coefficients that are obtained from PROC FACTOR. However, because the CALIS procedure does not contain the _TYPE_=STD observation in the OUTSTAT= data set (unless you specify the CORR option), PROC SCORE does not scale the observed variables by the standard deviations. Hence, in effect, PROC SCORE is able to use the correct deviation form for the observed variables "automatically" in computing latent variable (factor) scores.

When you specify the CORR option in the PROC CALIS statement, the estimation results are based on the analysis of correlation structures. The factor scoring coefficients are now applied to observed variables in the standardized form rather than the default deviation form. If you also specify the OUTSTAT= option, the OUTSTAT= data set for the correlation structure analysis contains the _TYPE_=STD observation. Consequently, when you use the SCORE procedure to compute latent variable (factor) scores, the standard deviations of the variables are available through the _TYPE_=STD observation in the OUTSTAT= data set. Therefore, even if you use PROC CALIS to perform correlational analysis (instead of the default covariance analysis), PROC SCORE is still able to use the correct standardized form for the observed variables "automatically" in computing latent variable (factor) scores.

Total, Direct, and Indirect Effects

Most structural equation models involve the specification of the effects of variables on each other. Whenever you specify equations in the LINEQS model, paths in the PATH model, path coefficient parameters in the RAM model, variable-factor relations in the FACTOR model, or regression coefficients in model matrices of the LISMOD model, you are specifying direct effects of predictor variables on outcome variables. All direct effects are represented by the associated regression coefficients, either fixed or free, in the specifications. You can examine the direct effect estimates easily in the output for model estimation.

However, direct effects are not the only effects that are important. In some cases, the indirect effects or total effects are of interest too. For example, suppose Self-Esteem is an important factor of Job Performance in your theory. Although it does not have a direct effect on Job Performance , it affects Job Performance through

its influences on Motivation and Endurance. Also, Motivation has a direct effect on Endurance in your theory. The following path diagram summarizes such a theory:

Figure 33.23 Direct and Indirect Effects of Self-Esteem on Job Performance

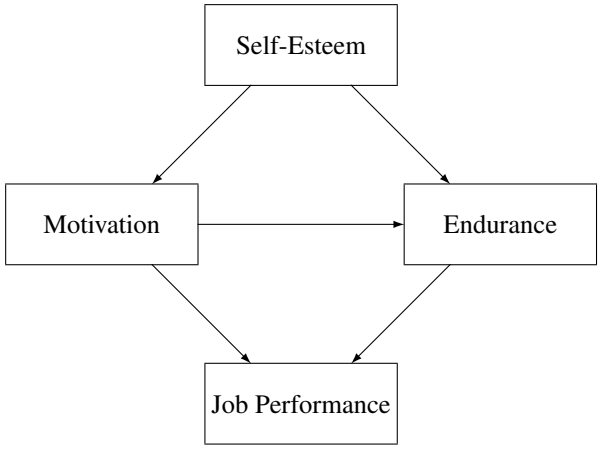

Clearly, each path in the diagram represents a direct effect of a predictor variable on an outcome variable. Less apparent are the total and indirect effects implied by the same path diagram. Despite this, interesting theoretical questions regarding the total and indirect effects can be raised in such a model. For example, even though there is no direct effect of Self-Esteem on Job Performance, what is its indirect effect on Job Performance? In addition to its direct effect on Job Performance, Motivation also has an indirect effect on Job Performance via its effect on Endurance. So, what is the total effect of Motivation on Job Performance and what portion of this total effect is indirect? The EFFPART option of the CALIS statement and the EFFPART statement are designed to address these questions. By using the EFFPART option or the EFFPART statement, PROC CALIS can compute the total, direct, and indirect effects of any sets of predictor variables on any sets of outcome variables. In this section, formulas for computing these effects are presented.

Formulas for Computing Total, Direct and Indirect Effects

No matter which modeling language is used, variables in a model can be classified into three groups. The first group is the so-called dependent variables, which serve as outcome variables at least once in the model. The other two groups consist of the remaining independent variables, which never serve as outcome variables in the model. The second group consists of independent variables that are unsystematic sources such as error and disturbance variables. The third group consists of independent variables that are systematic sources only.

Any variable, no matter which group it falls into, can have effects on the first group of variables. By definition, however, effects of variables in the first group on the other two groups do not exist. Because the effects of unsystematic sources in the second group are treated as residual effects on the first group of dependent variables, these effects are trivial in the sense that they always serve as direct effects only. That is, the effects from the second group of unsystematic sources partition trivially—total effects are always the same as the direct effects for this group. Therefore, for the purpose of effect analysis or partitioning, only the first group (dependent variables) and the third group (systematic independent variables) are considered.

Define $\mathbf{u}$ to be the set of n_u dependent variables in the first group and w to be the set of n_w systematic independent variables in the third group. Variables in both groups can be manifest or latent. All variables in the effect analysis is thus represented by the vector $(\mathbf{u}', \mathbf{w}')'$.

The $(n_u + n_w) \times (n_u + n_w)$ matrix $\mathbf{D}$ of *direct* effects refers to the path coefficients from all column variables to the row variables. This matrix is represented by:

$$\mathbf{D} = \begin{pmatrix} \boldsymbol{\beta} & \boldsymbol{\gamma} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$$

where $\boldsymbol{\beta}$ is an $(n_u \times n_u)$ matrix for direct effects of dependent variables on dependent variables and γ is an $(n_u \times n_w)$ matrix for direct effects of systematic independent variables on dependent variables. By definition, there should not be any direct effects on independent variables, and therefore the lower submatrices of $\mathbf{D}$ are null. In addition, by model restrictions the diagonal elements of matrix $\boldsymbol{\beta}$ must be zeros.

Correspondingly, the $(n_u + n_w) \times (n_u + n_w)$ matrix $\mathbf{T}$ of *total* effects of column variables on the row variables is computed by:

$$\mathbf{T} = \begin{pmatrix} (\mathbf{I} - \boldsymbol{\beta})^{-1} - \mathbf{I} & (\mathbf{I} - \boldsymbol{\beta})^{-1}\boldsymbol{\gamma} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$$

Finally, the $(n_u + n_w) \times (n_u + n_w)$ matrix $\boldsymbol{\mu}$ of *indirect* effects of column variables on the row variables is computed by the difference between $\mathbf{T}$ and $\mathbf{D}$ as:

$$\boldsymbol{\mu} = \begin{pmatrix} (\mathbf{I} - \boldsymbol{\beta})^{-1} - \mathbf{I} - \boldsymbol{\beta} & (\mathbf{I} - \boldsymbol{\beta})^{-1}\boldsymbol{\gamma} - \boldsymbol{\gamma} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}$$

In PROC CALIS, any subsets of $\mathbf{D}$, $\mathbf{T}$, and $\boldsymbol{\mu}$ can be requested via the specification in the EFFPART statement. All you need to do is to specify the sets of column variables (variables that have effects on others) and row variables (variables that receive the effects, direct or indirect). Specifications of the column and row variables are done conveniently by specifying variable names—no matrix terminology is needed. This feature is very handy if you have some focused subsets of effects that you want to analyze a priori. See the EFFPART statement on page 1599 for details about specifications.

Stability Coefficient of Reciprocal Causation

For recursive models (that is, models without cyclical paths of effects), using the preceding formulas for computing the total effect and the indirect effect is appropriate without further restrictions. However, for non-recursive models (that is, models with reciprocal effects or cyclical effects) the appropriateness of the preceding formulas for effect computations is restricted to situations with the convergence of the total effects.

A necessary and sufficient condition for the convergence of total effects (with or without cyclical paths) is when all eigenvalues, complex or real, of the $\boldsymbol{\beta}$ matrix fall into a unit circle (see Bentler and Freeman 1983). Equivalently, define the stability coefficient of reciprocal causation as the largest length (modulus) of the eigenvalues of the $\boldsymbol{\beta}$ matrix. A stability coefficient less than one would ensure that all eigenvalues, complex or real, of the $\boldsymbol{\beta}$ matrix fall into a unit circle. Hence, stability coefficient that is less than one is a necessary and sufficient condition for the convergence of the total effects, which in turn substantiates the appropriateness of total and indirect effect computations. Whenever effect analysis or partitioning is requested, PROC CALIS will check the appropriateness of effect computations by evaluating the stability coefficient of reciprocal causation. If the stability coefficient is greater than one, computations of the total and indirect effects will not be done.

Standardized Solutions

Standardized solutions are useful when you want to compare parameter values that are measured on quite different scales. PROC CALIS provides standardized solutions routinely. In standardizing a solution, parameters are classified into five groups:

- **path coefficients, regression coefficients, or direct effects**
  With each parameter α in this group, there is an associated outcome variable and a predictor variable. Denote the predicted variance of the outcome variable by σ_o^2 and the variance of the predictor variable by σ_p^2, the standardized parameter α^* is:

  $$\alpha^* = \alpha \frac{\sigma_p}{\sigma_o}$$

- **fixed ones for the path coefficients attached to error or disturbance terms**
  These fixed values are unchanged in standardization.

- **variances and covariances among exogenous variables, excluding errors and disturbances**
  Let σ_{ij} be the covariance between variables i and j. In this notation, σ_{ii} is the variance of variable i. The standardized covariance σ_{ij}^* is:

  $$\sigma_{ij}^* = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}}$$

  When $i = j$, σ_{ii}^* takes the value of 1 for all i. Also, σ_{ij}^* is the correlation between the ith and jth variables.

- **variances and covariances among errors or disturbances**
  Denote the error covariance parameter as θ_{ij} so that θ_{ii} represents the variance parameter of error variable i. Associated with each error or disturbance variable i is a unique outcome variable. Let the variance of such an outcome variable be σ_{ii}. In the standardized solution, the error covariance θ_{ij} is rescaled as:

  $$\theta_{ij}^* = \frac{\theta_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}}$$

  Notice that when $i = j$, θ_{ii}^* is not standardized to 1 in general. In fact, the error (disturbance) variance is simply rescaled by the reciprocal of the variance of the associated dependent variable. As a result, the rescaled error (disturbance) variance represents the proportion of variation of the dependent variable due to the unsystematic source. By the same token, θ_{ij}^* does not represent the correlation between errors i and j. It is a rescaled covariance of the errors involved.

- **intercepts and means of variables**
  These parameters are fixed zeros in the standardized solution.

While formulas for the standardized solution are useful in computing the parameter values in the standardized solution, it is conceptually more useful to explain how variables are being transformed in the standardization process. The following provides a summary of the transformation process:

- Observed and latent variables, excluding errors or disturbances, are centered and then divided by their corresponding standard deviations. Therefore, in the standardized solution, all these variables will have variance equal to 1. In other words, these variables are truly standardized.
- Errors or disturbances are divided by the standard deviations of the corresponding outcome variables. In the standardized solution, these variables will not have variance equal to 1 in general. However, the rescaled error variances represent the proportion of unexplained or unsystematic variance of the corresponding outcome variables. Therefore, errors or disturbances in the standardized solution are simply rescaled but not standardized.

Standardized total, direct, and indirect effects are computed using formulas presented in the section "Total, Direct, and Indirect Effects" on page 1844, but with the standardized parameter values substituted into the formulas.

Although parameter values associated with different scales are made more comparable in the standardized solution, a precaution should be mentioned. In the standardized solution, the original constraints on parameters in the unstandardized solution are usually lost. These constraints, however, might underscore some important theoretical position that needs to be maintained in the model. Destroying these constraints in the standardized solution means that interpretations or comparisons of parameter values in the standardized solution are made without maintaining the original theoretical position. You must judge whether such a departure from the original constraints poses conceptual difficulties for interpreting the standardized solution.

Modification Indices

While fitting structural equation models is mostly a confirmatory analytic procedure, it does not prevent you from exploring what might have been a better model given the data. After fitting your theoretical structural equation model, you might want to modify the original model in order to do one of the following:

- add free parameters to improve the model fit significantly
- reduce the number of parameters without affecting the model fit too much

The first kind of model modification can be achieved by using the Lagrange multiplier (LM) test indices. Parameters that have the largest LM indices would increase the model fit the most. In general, adding more parameters to your model improves the overall model fit, as measured by those absolute or standalone fit indices (see the section "Overall Model Fit Indices" on page 1827 for more details). However, adding parameters liberally makes your model more prone to sampling errors. It also makes your model more complex and less interpretable in most cases. A disciplined use of LM test indices is highly recommended. In addition to the model fit improvement indicated by the LM test indices, you should also consider the theoretical significance when adding particular parameters. See Example 33.28 for an illustration of the use of LM test indices for improving model fit.

The second kind of model modification can be achieved by using the Wald statistics. Parameters that are not significant in your model may be removed from the model without affecting the model fit too much. In general, removing parameters from your model decreases the model fit, as measured by those absolute or standalone fit indices (see the section "Overall Model Fit Indices" on page 1827 for more details). However, for just a little sacrifice in model fit, removing non-significant parameters increases the simplicity and precision of your model, which is the virtue that any modeler should look for.

Whether adding parameters by using the LM test indices or removing unnecessary parameters by the Wald statistics, you should not treat your modified model as if it were your original hypothesized model. That is, you should not publish your modified model as if it were hypothesized a priori. It is perfectly fine to use modification indices to gain additional insights for future research. But if you want to publish your modified model together with your original model, you should report the modification process that leads to your modified model. Theoretical justifications of the modified model should be supplemented if you want to make strong statements to support your modified model. Whenever possible, the best practice is to show reasonable model fit of the modified model with new data.

To modify your model either by LM test indices or Wald statistics, you can use the MODIFICATION or MOD option in the PROC CALIS statement. To customize the LM tests by setting specific regions of parameters, you can use the LMTESTS statements. PROC CALIS computes and displays the following default set of modification indices:

- **univariate Lagrange multiplier (LM) test indices for parameters in the model**
  These are second-order approximations of the decrease in the χ^2 value that would result from allowing the *fixed parameter values* in the model to be freed to estimate. LM test indices are ranked within their own parameter regions in the model. The ones that suggest greatest model improvements (that is, greatest χ^2 drop) are ranked first. Depending on the type of your model, the set of possible parameter regions varies. For example, in a RAM model, modification indices are ranked in three different parameter regions for the covariance structures: path coefficients, variances of and covariances among exogenous variables, and the error variances and covariances. In addition to the value of the Lagrange multiplier, the corresponding p-value ($df = 1$) and the approximate change of the parameter value are displayed.

  If you use the LMMAT option in the LMTESTS statement, LM test indices are shown as elements in model matrices. Not all elements in a particular model matrix will have LM test indices. Elements that are already free parameters in the model do not have LM test indices. Instead, the parameter names are shown. Elements that are model restricted values (for example, direct path from a variable to itself must be zero) are labeled Excluded in the matrix output. When you customize your own regions of LM tests, some elements might also be excluded from a custom set of LM tests. These elements are also labeled as Excluded in the matrix output. If an LM test for freeing a parameter would result in a singular information matrix, the corresponding element in the matrix is labeled as Singular.

- **univariate Lagrange multiplier test indices for releasing equality constraints**
  These are second-order approximations of the decrease in the χ^2 value that would result from the release of *equality constraints*. Multiple equality constraints containing $n > 2$ parameters are tested successively in n steps, each assuming the release of one of the equality-constrained parameters. The expected change of the parameter values of the separated parameter and the remaining parameter cluster are displayed, too.

- **univariate Lagrange multiplier test indices for releasing active boundary constraints**
  These are second-order approximations of the decrease in the χ^2 value that would result from the release of the *active boundary constraints* specified in the BOUNDS statement.

- **stepwise multivariate Wald statistics for constraining free parameters to 0**
  These are second-order approximations of the increases in χ^2 value that would result from constraining free parameters to zero in a stepwise fashion. In each step, the parameter that would lead to the smallest increase in the multivariate χ^2 value is set to 0. Besides the multivariate χ^2 value and its p-value, the

univariate increments are also displayed. The process stops when the univariate p-value is smaller than the specified value in the SLMW= option, of which the default value is 0.05.

All of the preceding tests are approximations. You can often obtain more accurate tests by actually fitting different models and computing likelihood ratio tests. For more details about the Wald and the Lagrange multiplier test, see MacCallum (1986), Buse (1982), Bentler (1986), or Lee (1985). Note that relying solely on the LM tests to modify your model can lead to unreliable models that capitalize purely on sampling errors. See MacCallum, Roznowski, and Necowitz (1992) for the use of LM tests.

For large model matrices, the computation time for the default modification indices can considerably exceed the time needed for the minimization process.

The modification indices are not computed for unweighted least squares or diagonally weighted least squares estimation.

Missing Values and the Analysis of Missing Patterns

If the DATA= data set contains raw data (rather than a covariance or correlation matrix), in general observations with missing values for any variables in the analysis are omitted from the computations. The only exception is with METHOD=FIML. Incomplete observations with at least one nonmissing variables in the analysis are also used for the estimation.

If a covariance or correlation matrix is read, missing values are allowed as long as every pair of variables has at least one nonmissing value. Unlike the raw data input, METHOD=FIML does not allow missing values in the covariance or correlation matrix.

When you use METHOD=FIML, PROC CALIS provide several analyses on the missing patterns of the raw input data sets. First, PROC CALIS shows the coverage results for the means and covariances. The coverage results refer to the proportions of data present for computing the means and the covariances. Because distinct missing patterns in the data sets are possible, the coverage proportions for the individual means and covariances could vary. Average coverage proportions of the means and covariances give you an overall idea about the missingness (or the lack of). In order to help locate the problematic means and covariances that have the low coverage, PROC CALIS shows the rank orders of the smallest coverages of mean and covariance elements. The number of smallest coverages shown for the means is equal to half of the total number of variables. The number of smallest coverages shown for the covariances is equal to half of the total number of the distinct elements in the lower triangular of the covariance matrix. However, in both cases at most 10 smallest coverages would be shown.

Second, PROC CALIS ranks the most frequent missing patterns in the data set (the nonmissing pattern is excluded in the ranking). Because the number of missing patterns could be quite large, PROC CALIS displays only a limited number of most frequent missing patterns in the output. You can use the MAXMISSPAT= and the TMISSPAT= options to control the number of missing patterns to display. See these options for details.

Third, PROC CALIS shows the means of the most frequent missing patterns, along with the means for the nonmissing pattern for comparison.

See Example 33.15 for an illustration of the use of the full information maximum likelihood method and the analysis of missing patterns. For examples and details about the FIML method and its missing data treatment in PROC CALIS, see Yung and Zhang (2011); Zhang and Yung (2011).

Measures of Multivariate Kurtosis

In many applications, the manifest variables are not even approximately multivariate normal. If this happens to be the case with your data set, the default generalized least squares and maximum likelihood estimation methods are not appropriate, and you should compute the parameter estimates and their standard errors by an asymptotically distribution-free method, such as the WLS estimation method. If your manifest variables are multivariate normal, then they have a zero relative multivariate kurtosis, and all marginal distributions have zero kurtosis (Browne 1982). If your DATA= data set contains raw data, PROC CALIS computes univariate skewness and kurtosis and a set of multivariate kurtosis values. By default, the values of univariate skewness and kurtosis are corrected for bias (as in PROC UNIVARIATE), but using the BIASKUR option enables you to compute the uncorrected values also. The values are displayed when you specify the PROC CALIS statement option KURTOSIS.

In the following formulas, N denotes the sample size and p denotes the number of variables.

- **corrected variance for variable** z_j

$$\sigma_j^2 = \frac{1}{N-1}\sum_i^N (z_{ij} - \bar{z}_j)^2$$

- **uncorrected univariate skewness for variable** z_j

$$\gamma_{1(j)} = \frac{N\sum_i^N (z_{ij} - \bar{z}_j)^3}{\sqrt{N[\sum_i^N (z_{ij} - \bar{z}_j)^2]^3}}$$

- **corrected univariate skewness for variable** z_j

$$\gamma_{1(j)} = \frac{N}{(N-1)(N-2)}\frac{\sum_i^N (z_{ij} - \bar{z}_j)^3}{\sigma_j^3}$$

- **uncorrected univariate kurtosis for variable** z_j

$$\gamma_{2(j)} = \frac{N\sum_i^N (z_{ij} - \bar{z}_j)^4}{[\sum_i^N (z_{ij} - \bar{z}_j)^2]^2} - 3$$

- **corrected univariate kurtosis for variable** z_j

$$\gamma_{2(j)} = \frac{N(N+1)}{(N-1)(N-2)(N-3)}\frac{\sum_i^N (z_{ij} - \bar{z}_j)^4}{\sigma_j^4} - \frac{3(N-1)^2}{(N-2)(N-3)}$$

- **Mardia's multivariate kurtosis**

$$\gamma_2 = \frac{1}{N}\sum_i^N [(z_i - \bar{z})'\mathbf{S}^{-1}(z_i - \bar{z})]^2 - p(p+2)$$

where $\mathbf{S}$ is the biased sample covariance matrix with N as the divisor.

- **relative multivariate kurtosis**

$$\eta_2 = \frac{\gamma_2 + p(p+2)}{p(p+2)}$$

- **normalized multivariate kurtosis**

$$\kappa_0 = \frac{\gamma_2}{\sqrt{8p(p+2)/N}}$$

- **Mardia based kappa**

$$\kappa_1 = \frac{\gamma_2}{p(p+2)}$$

- **mean scaled univariate kurtosis**

$$\kappa_2 = \frac{1}{3p}\sum_{j}^{p}\gamma_{2(j)}$$

- **adjusted mean scaled univariate kurtosis**

$$\kappa_3 = \frac{1}{3p}\sum_{j}^{p}\gamma^*_{2(j)}$$

with

$$\gamma^*_{2(j)} = \begin{cases} \gamma_{2(j)} & , \quad \text{if} \quad \gamma_{2(j)} > \frac{-6}{p+2} \\ \frac{-6}{p+2} & , \quad \text{otherwise} \end{cases}$$

If variable Z_j is normally distributed, the uncorrected univariate kurtosis $\gamma_{2(j)}$ is equal to 0. If Z has an p-variate normal distribution, Mardia's multivariate kurtosis γ_2 is equal to 0. A variable Z_j is called *leptokurtic* if it has a positive value of $\gamma_{2(j)}$ and is called *platykurtic* if it has a negative value of $\gamma_{2(j)}$. The values of κ_1, κ_2, and κ_3 should not be smaller than the following lower bound (Bentler 1985):

$$\hat{\kappa} \geq \frac{-2}{p+2}$$

PROC CALIS displays a message if κ_1, κ_2, or κ_3 falls below the lower bound.

If weighted least squares estimates (METHOD=WLS or METHOD=ADF) are specified and the weight matrix is computed from an input raw data set, the CALIS procedure computes two more measures of multivariate kurtosis.

- **multivariate mean kappa**

$$\kappa_4 = \frac{1}{m}\sum_{i}^{p}\sum_{j}^{i}\sum_{k}^{j}\sum_{l}^{k}\hat{\kappa}_{ij,kl} - 1$$

where

$$\hat{\kappa}_{ij,kl} = \frac{s_{ij,kl}}{s_{ij}s_{kl} + s_{ik}s_{jl} + s_{il}s_{jk}}$$

and $m = p(p+1)(p+2)(p+3)/24$ is the number of elements in the vector $s_{ij,kl}$ (Bentler 1985).

- **multivariate least squares kappa**

$$\kappa_5 = \frac{s_4' s_2}{s_2' s_2} - 1$$

where s_2 is the vector of the elements in the denominator of $\hat{\kappa}$ (Bentler 1985) and s_4 is the vector of the $s_{ij,kl}$, which is defined as

$$s_{ij,kl} = \frac{1}{N}\sum_{r=1}^{N}(z_{ri} - \bar{z}_i)(z_{rj} - \bar{z}_j)(z_{rk} - \bar{z}_k)(z_{rl} - \bar{z}_l)$$

The occurrence of significant nonzero values of Mardia's multivariate kurtosis γ_2 and significant amounts of some of the univariate kurtosis values $\gamma_{2(j)}$ indicate that your variables are not multivariate normal distributed. Violating the multivariate normality assumption in (default) generalized least squares and maximum likelihood estimation usually leads to the wrong approximate standard errors and incorrect fit statistics based on the χ^2 value. In general, the parameter estimates are more stable against violation of the normal distribution assumption. For more details, see Browne (1974, 1982, 1984).

Initial Estimates

Each optimization technique requires a set of initial values for the parameters. To avoid local optima, the initial values should be as close as possible to the globally optimal solution. You can check for local optima by running the analysis with several different sets of initial values; the RANDOM= option in the PROC CALIS statement is useful in this regard.

Except for the case of exploratory FACTOR model, you can specify initial estimates manually for all different types of models. If you do not specify some of the initial estimates and the RANDOM= option is not used, PROC CALIS will use a combination of good strategic methods to compute initial estimates for your model.

These initial estimation methods are used in PROC CALIS:

- two-stage least squares estimation
- instrumental variable method (Hägglund 1982; Jennrich 1987)
- approximate factor analysis method
- ordinary least squares estimation
- estimation method of McDonald (McDonald and Hartmann 1992)
- observed moments of manifest exogenous variables

The choice of initial estimation methods is dependent on the data and on the model. In general, it is difficult to tell in advance which initial estimation methods will be used for a given analysis. However, PROC CALIS displays the methods used to obtain initial estimates in the output. Notice that none of these initial estimation methods can be applied to the COSAN model because of the general formulation of the COSAN model. If you do not provide initial parameter estimates for the COSAN model, the default values or random values are used (see the START= and the RANDOM= options).

Poor initial values can cause convergence problems, especially with maximum likelihood estimation. Sufficiently large positive initial values for variance estimates (as compared with the covariance estimates) might help prevent a nonnegative definite initial predicted covariance model matrix from happening. If maximum likelihood estimation fails to converge, it might help to use METHOD=LSML, which uses the final estimates from an unweighted least squares analysis as initial estimates for maximum likelihood. Or you can fit a slightly different but better-behaved model and produce an OUTMODEL= data set, which can then be modified in accordance with the original model and used as an INMODEL= data set to provide initial values for another analysis.

If you are analyzing a covariance or scalar product matrix, be sure to take into account the scales of the variables. The default initial values might be inappropriate when some variables have extremely large or small variances.

Use of Optimization Techniques

No algorithm for optimizing general nonlinear functions exists that can always find the global optimum for a general nonlinear minimization problem in a reasonable amount of time. Since no single optimization technique is invariably superior to others, PROC CALIS provides a variety of optimization techniques that work well in various circumstances. However, you can devise problems for which none of the techniques in PROC CALIS can find the correct solution. All optimization techniques in PROC CALIS use $O(n^2)$ memory except the conjugate gradient methods, which use only $O(n)$ of memory and are designed to optimize problems with many parameters.

The PROC CALIS statement NLOPTIONS can be especially helpful for tuning applications with nonlinear equality and inequality constraints on the parameter estimates. Some of the options available in NLOPTIONS can also be invoked as PROC CALIS options. The NLOPTIONS statement can specify almost the same options as the SAS/OR NLP procedure.

Nonlinear optimization requires the repeated computation of the following:

- the function value (optimization criterion)
- the gradient vector (first-order partial derivatives)
- for some techniques, the (approximate) Hessian matrix (second-order partial derivatives)
- values of linear and nonlinear constraints
- the first-order partial derivatives (Jacobian) of nonlinear constraints

For the criteria used by PROC CALIS, computing the gradient takes more computer time than computing the function value, and computing the Hessian takes *much* more computer time and memory than computing the

gradient, especially when there are many parameters to estimate. Unfortunately, optimization techniques that do not use the Hessian usually require many more iterations than techniques that do use the (approximate) Hessian, and so they are often slower. Techniques that do not use the Hessian also tend to be less reliable (for example, they might terminate at local rather than global optima).

The available optimization techniques are displayed in the following table and can be chosen by the OMETHOD=*name* option.

OMETHOD=	**Optimization Technique**
LEVMAR	Levenberg-Marquardt method
TRUREG	Trust-region method
NEWRAP	Newton-Raphson method with line search
NRRIDG	Newton-Raphson method with ridging
QUANEW	Quasi-Newton methods (DBFGS, DDFP, BFGS, DFP)
DBLDOG	Double-dogleg method (DBFGS, DDFP)
CONGRA	Conjugate gradient methods (PB, FR, PR, CD)

The following table shows, for each optimization technique, which derivatives are needed (first-order or second-order) and what kind of constraints (boundary, linear, or nonlinear) can be imposed on the parameters.

	Derivatives		**Constraints**		
OMETHOD=	**First Order**	**Second Order**	**Boundary**	**Linear**	**Nonlinear**
LEVMAR	x	x	x	x	-
TRUREG	x	x	x	x	-
NEWRAP	x	x	x	x	-
NRRIDG	x	x	x	x	-
QUANEW	x	-	x	x	x
DBLDOG	x	-	x	x	-
CONGRA	x	-	x	x	-

The Levenberg-Marquardt, trust-region, and Newton-Raphson techniques are usually the most reliable, work well with boundary and general linear constraints, and generally converge after a few iterations to a precise solution. However, these techniques need to compute a Hessian matrix in each iteration. Computing the approximate Hessian in each iteration can be very time- and memory-consuming, especially for large problems (more than 200 parameters, depending on the computer used). For large problems, a quasi-Newton technique, especially with the BFGS update, can be far more efficient.

For a poor choice of initial values, the Levenberg-Marquardt method seems to be more reliable.

If memory problems occur, you can use one of the conjugate gradient techniques, but they are generally slower and less reliable than the methods that use second-order information.

There are several options to control the optimization process. You can specify various termination criteria. You can specify the GCONV= option to specify a relative gradient termination criterion. If there are active boundary constraints, only those gradient components that correspond to inactive constraints contribute to the criterion. When you want very precise parameter estimates, the GCONV= option is useful. Other criteria that use relative changes in function values or parameter estimates in consecutive iterations can lead to early termination when active constraints cause small steps to occur. The small default value for the FCONV= option helps prevent early termination. Using the MAXITER= and MAXFUNC= options enables you to specify the maximum number of iterations and function calls in the optimization process. These limits

are especially useful in combination with the INMODEL= and OUTMODEL= options; you can run a few iterations at a time, inspect the results, and decide whether to continue iterating.

Nonlinearly Constrained QN Optimization

The algorithm used for nonlinearly constrained quasi-Newton optimization is an efficient modification of Powell's Variable Metric Constrained WatchDog (VMCWD) algorithm (Powell 1978a, b, 1982a, b) A similar but older algorithm (VF02AD) is part of the Harwell library. Both VMCWD and VF02AD use Fletcher's VE02AD algorithm (also part of the Harwell library) for positive definite quadratic programming. The PROC CALIS QUANEW implementation uses a quadratic programming subroutine that updates and downdates the approximation of the Cholesky factor when the active set changes. The nonlinear QUANEW algorithm is not a feasible point algorithm, and the value of the objective function might not necessarily decrease (minimization) or increase (maximization) monotonically. Instead, the algorithm tries to reduce a linear combination of the objective function and constraint violations, called the *merit function*.

The following are similarities and differences between this algorithm and VMCWD:

- A modification of this algorithm can be performed by specifying VERSION=1, which replaces the update of the Lagrange vector μ with the original update of Powell (1978a, b), which is used in VF02AD. This can be helpful for some applications with linearly dependent active constraints.
- If the VERSION= option is not specified or VERSION=2 is specified, the evaluation of the Lagrange vector μ is performed in the same way as Powell (1982a, b) describes.
- Instead of updating an approximate Hessian matrix, this algorithm uses the dual BFGS (or DFP) update that updates the Cholesky factor of an approximate Hessian. If the condition of the updated matrix gets too bad, a restart is done with a positive diagonal matrix. At the end of the first iteration after each restart, the Cholesky factor is scaled.
- The Cholesky factor is loaded into the quadratic programming subroutine, automatically ensuring positive definiteness of the problem. During the quadratic programming step, the Cholesky factor of the projected Hessian matrix $\mathbf{Z}_k'\mathbf{G}\mathbf{Z}_k$ and the QT decomposition are updated simultaneously when the active set changes. See Gill et al. (1984) for more information.
- The line-search strategy is very similar to that of Powell (1982a, b). However, this algorithm does not call for derivatives during the line search; hence, it generally needs fewer derivative calls than function calls. The VMCWD algorithm always requires the same number of derivative and function calls. It was also found in several applications of VMCWD that Powell's line-search method sometimes uses steps that are too long during the first iterations. In those cases, you can use the INSTEP= option specification to restrict the step length α of the first iterations.
- The watchdog strategy is similar to that of Powell (1982a, b). However, this algorithm does not return automatically after a fixed number of iterations to a former better point. A return here is further delayed if the observed function reduction is close to the expected function reduction of the quadratic model.
- Although Powell's termination criterion still is used (as FCONV2), the QUANEW implementation uses two additional termination criteria (GCONV and ABSGCONV).

This algorithm is automatically invoked when you specify the NLINCON statement. The nonlinear QUANEW algorithm needs the Jacobian matrix of the first-order derivatives (constraints normals) of the constraints:

$$(\nabla c_i) = (\frac{\partial c_i}{\partial x_j}), \quad i = 1, \ldots, nc, j = 1, \ldots, n$$

where nc is the number of nonlinear constraints for a given point x.

You can specify two update formulas with the UPDATE= option:

- UPDATE=DBFGS performs the dual BFGS update of the Cholesky factor of the Hessian matrix. This is the default.
- UPDATE=DDFP performs the dual DFP update of the Cholesky factor of the Hessian matrix.

This algorithm uses its own line-search technique. All options and parameters (except the INSTEP= option) controlling the line search in the other algorithms do not apply here. In several applications, large steps in the first iterations are troublesome. You can specify the INSTEP= option to impose an upper bound for the step size α during the first five iterations. The values of the LCSINGULAR=, LCEPSILON=, and LCDEACT= options (which control the processing of linear and boundary constraints) are valid only for the quadratic programming subroutine used in each iteration of the nonlinear constraints QUANEW algorithm.

Optimization and Iteration History

The optimization and iteration histories are displayed by default because it is important to check for possible convergence problems. The optimization history includes the following summary of information about the initial state of the optimization:

- the number of constraints that are active at the starting point, or more precisely, the number of constraints that are currently members of the working set. If this number is followed by a plus sign, there are more active constraints, of which at least one is temporarily released from the working set due to negative Lagrange multipliers.
- the value of the objective function at the starting point
- if the (projected) gradient is available, the value of the largest absolute (projected) gradient element
- for the TRUREG and LEVMAR subroutines, the initial radius of the trust region around the starting point

The optimization history ends with some information concerning the optimization result:

- the number of constraints that are active at the final point, or more precisely, the number of constraints that are currently members of the working set. If this number is followed by a plus sign, there are more active constraints, of which at least one is temporarily released from the working set due to negative Lagrange multipliers.
- the value of the objective function at the final point
- if the (projected) gradient is available, the value of the largest absolute (projected) gradient element

- other information specific to the optimization technique

The iteration history generally consists of one line of displayed output containing the most important information for each iteration.

The iteration history always includes the following:

- the iteration number
- the number of iteration restarts
- the number of function calls
- the number of active constraints
- the value of the optimization criterion
- the difference between adjacent function values
- the maximum of the absolute gradient components that correspond to inactive boundary constraints

An apostrophe trailing the number of active constraints indicates that at least one of the active constraints is released from the active set due to a significant Lagrange multiplier.

For the Levenberg-Marquardt technique (LEVMAR), the iteration history also includes the following information:

- an asterisk trailing the iteration number when the computed Hessian approximation is singular and consequently ridged with a positive lambda value. If all or the last several iterations show a singular Hessian approximation, the problem is not sufficiently identified. Thus, there are other locally optimal solutions that lead to the same optimum function value for different parameter values. This implies that standard errors for the parameter estimates are not computable without the addition of further constraints.
- the value of the Lagrange multiplier (lambda). This value is 0 when the optimum of the quadratic function approximation is inside the trust region (a trust-region-scaled Newton step can be performed) and is greater than 0 when the optimum of the quadratic function approximation is located at the boundary of the trust region (the scaled Newton step is too long to fit in the trust region and a quadratic constraint optimization is performed). Large values indicate optimization difficulties. For a nonsingular Hessian matrix, the value of lambda should go to 0 during the last iterations, indicating that the objective function can be well approximated by a quadratic function in a small neighborhood of the optimum point. An increasing lambda value often indicates problems in the optimization process.
- the value of the ratio ρ (rho) between the actually achieved difference in function values and the predicted difference in the function values on the basis of the quadratic function approximation. Values much less than 1 indicate optimization difficulties. The value of the ratio ρ indicates the goodness of the quadratic function approximation. In other words, $\rho << 1$ means that the radius of the trust region has to be reduced; a fairly large value of ρ means that the radius of the trust region does not need to be changed. And a value close to or greater than 1 means that the radius can be increased, indicating a good quadratic function approximation.

For the Newton-Raphson technique (NRRIDG), the iteration history also includes the following information:

- the value of the ridge parameter. This value is 0 when a Newton step can be performed, and it is greater than 0 when either the Hessian approximation is singular or a Newton step fails to reduce the optimization criterion. Large values indicate optimization difficulties.
- the value of the ratio ρ (rho) between the actually achieved difference in function values and the predicted difference in the function values on the basis of the quadratic function approximation. Values much less than 1.0 indicate optimization difficulties.

For the Newton-Raphson with line-search technique (NEWRAP), the iteration history also includes the following information:

- the step size α (alpha) computed with one of the line-search algorithms
- the slope of the search direction at the current parameter iterate. For minimization, this value should be significantly negative. Otherwise, the line-search algorithm has difficulty reducing the function value sufficiently.

For the trust-region technique (TRUREG), the iteration history also includes the following information:

- an asterisk after the iteration number when the computed Hessian approximation is singular and consequently ridged with a positive lambda value.
- the value of the Lagrange multiplier (lambda). This value is zero when the optimum of the quadratic function approximation is inside the trust region (a trust-region-scaled Newton step can be performed) and is greater than zero when the optimum of the quadratic function approximation is located at the boundary of the trust region (the scaled Newton step is too long to fit in the trust region and a quadratically constrained optimization is performed). Large values indicate optimization difficulties. As in Gay (1983), a negative lambda value indicates the special case of an indefinite Hessian matrix (the smallest eigenvalue is negative in minimization).
- the value of the radius Δ of the trust region. Small trust-region radius values combined with large lambda values in subsequent iterations indicate optimization problems.

For the quasi-Newton (QUANEW) and conjugate gradient (CONGRA) techniques, the iteration history also includes the following information:

- the step size (alpha) computed with one of the line-search algorithms
- the descent of the search direction at the current parameter iterate. This value should be significantly smaller than 0. Otherwise, the line-search algorithm has difficulty reducing the function value sufficiently.

Frequent update restarts (rest) of a quasi-Newton algorithm often indicate numerical problems related to required properties of the approximate Hessian update, and they decrease the speed of convergence. This can happen particularly if the ABSGCONV= termination criterion is too small—that is, when the requested precision cannot be obtained by quasi-Newton optimization. Generally, the number of automatic restarts used by conjugate gradient methods are much higher.

For the nonlinearly constrained quasi-Newton technique, the iteration history also includes the following information:

- the maximum value of all constraint violations,

$$\text{conmax} = \max(|c_i(x)| : c_i(x) < 0)$$

- the value of the predicted function reduction used with the GCONV and FCONV2 termination criteria,

$$\text{pred} = |g(x^{(k)})s(x^{(k)})| + \sum_{i=1}^{m} |\lambda_i c_i(x^{(k)})|$$

- the step size α of the quasi-Newton step. Note that this algorithm works with a special line-search algorithm.
- the maximum element of the gradient of the Lagrange function,

$$\begin{aligned} \text{lfgmax} &= \nabla_x L(x^{(k)}, \lambda^{(k)}) \\ &= \nabla_x f(x^{(k)}) - \sum_{i=1}^{m} \lambda_i^{(k)} \nabla_x c_i(x^{(k)}) \end{aligned}$$

For the double dogleg technique, the iteration history also includes the following information:

- the parameter λ of the double-dogleg step. A value $\lambda = 0$ corresponds to the full (quasi) Newton step.
- the slope of the search direction at the current parameter iterate. For minimization, this value should be significantly negative.

Line-Search Methods

In each iteration k, the (dual) quasi-Newton, hybrid quasi-Newton, conjugate gradient, and Newton-Raphson minimization techniques use iterative line-search algorithms that try to optimize a linear, quadratic, or cubic approximation of the nonlinear objective function f of n parameters x along a feasible descent search direction $s^{(k)}$ as follows:

$$f(x^{(k+1)}) = f(x^{(k)} + \alpha^{(k)} s^{(k)})$$

by computing an approximately optimal scalar $\alpha^{(k)} > 0$. Since the outside iteration process is based only on the approximation of the objective function, the inside iteration of the line-search algorithm does not have to be perfect. Usually, it is satisfactory that the choice of α significantly reduces (in a minimization) the objective function. Criteria often used for termination of line-search algorithms are the Goldstein conditions (Fletcher 1987).

Various line-search algorithms can be selected by using the LIS= option on page 1557. The line-search methods LIS=1, LIS=2, and LIS=3 satisfy the left-hand-side and right-hand-side Goldstein conditions (Fletcher 1987).

The line-search method LIS=2 seems to be superior when function evaluation consumes significantly less computation time than gradient evaluation. Therefore, LIS=2 is the default value for Newton-Raphson, (dual) quasi-Newton, and conjugate gradient optimizations.

Restricting the Step Length

Almost all line-search algorithms use iterative extrapolation techniques that can easily lead to feasible points where the objective function f is no longer defined (resulting in indefinite matrices for ML estimation) or is difficult to compute (resulting in floating point overflows). Therefore, PROC CALIS provides options that restrict the step length or trust region radius, especially during the first main iterations.

The inner product $\mathbf{g}'\mathbf{s}$ of the gradient $\mathbf{g}$ and the search direction $\mathbf{s}$ is the slope of $f(\alpha) = f(\mathbf{x} + \alpha\mathbf{s})$ along the search direction $\mathbf{s}$ with step length α. The default starting value $\alpha^{(0)} = \alpha^{(k,0)}$ in each line-search algorithm ($\min_{\alpha>0} f(\mathbf{x} + \alpha\mathbf{s})$) during the main iteration k is computed in three steps:

1. Use either the difference $df = |f^{(k)} - f^{(k-1)}|$ of the function values during the last two consecutive iterations or the final stepsize value α^- of the previous iteration $k - 1$ to compute a first value $\alpha_1^{(0)}$.

   - Using the DAMPSTEP<r> option:

     $$\alpha_1^{(0)} = \min(1, r\alpha^-)$$

     The initial value for the new step length can be no greater than r times the final step length α^- of the previous iteration. The default is $r = 2$.

   - Not using the DAMPSTEP option:

     $$\alpha_1^{(0)} = \begin{cases} step & \text{if } 0.1 \le step \le 10 \\ 10 & \text{if } step > 10 \\ 0.1 & \text{if } step < 0.1 \end{cases}$$

     with

     $$step = \begin{cases} df/|\mathbf{g}'\mathbf{s}| & \text{if } |\mathbf{g}'\mathbf{s}| \ge \epsilon \max(100df, 1) \\ 1 & \text{otherwise} \end{cases}$$

     This value of $\alpha_1^{(0)}$ can be too large and can lead to a difficult or impossible function evaluation, especially for highly nonlinear functions such as the EXP function.

2. During the first five iterations, the second step enables you to reduce $\alpha_1^{(0)}$ to a smaller starting value $\alpha_2^{(0)}$ using the INSTEP=r option:

   $$\alpha_2^{(0)} = \min(\alpha_1^{(0)}, r)$$

   After more than five iterations, $\alpha_2^{(0)}$ is set to $\alpha_1^{(0)}$.

3. The third step can further reduce the step length by

$$\alpha_3^{(0)} = \min(\alpha_2^{(0)}, \min(10, u))$$

where u is the maximum length of a step inside the feasible region.

The INSTEP=*r* option lets you specify a smaller or larger radius of the trust region used in the first iteration by the trust-region, double-dogleg, and Levenberg-Marquardt algorithms. The default initial trust region radius is the length of the scaled gradient (Moré 1978). This default length for the initial trust region radius corresponds to the default radius factor of $r = 1$. This choice is successful in most practical applications of the TRUREG, DBLDOG, and LEVMAR algorithms. However, for bad initial values used in the analysis of a covariance matrix with high variances or for highly nonlinear constraints (such as using the EXP function) in your SAS programming statements, the default start radius can result in arithmetic overflows. If this happens, you can try decreasing values of INSTEP=*r* ($0 < r < 1$), until the iteration starts successfully. A small factor r also affects the trust region radius of the next steps because the radius is changed in each iteration by a factor $0 < c \leq 4$ depending on the ρ ratio. Reducing the radius corresponds to increasing the ridge parameter λ that produces smaller steps directed closer toward the gradient direction.

Computational Problems

First Iteration Overflows

Analyzing a covariance matrix that includes high variances in the diagonal and using bad initial estimates for the parameters can easily lead to arithmetic overflows in the first iterations of the minimization algorithm. The line-search algorithms that work with cubic extrapolation are especially sensitive to arithmetic overflows. If this occurs with quasi-Newton or conjugate gradient minimization, you can specify the INSTEP= option to reduce the length of the first step. If an arithmetic overflow occurs in the first iteration of the Levenberg-Marquardt algorithm, you can specify the INSTEP= option to reduce the trust region radius of the first iteration. You also can change the minimization technique or the line-search method. If none of these help, you can consider doing the following:

- scaling the covariance matrix
- providing better initial values
- changing the model

No Convergence of Minimization Process

If convergence does not occur during the minimization process, perform the following tasks:

- If there are *negative variance estimates*, you can do either of the following:
  - Specify the BOUNDS statement to obtain nonnegative variance estimates.
  - Specify the HEYWOOD option, if the FACTOR statement is specified.

- Change the estimation method to obtain a better set of initial estimates. For example, if you use METHOD=ML, you can do either of the following:
  - Change to METHOD=LSML.
  - Run some iterations with METHOD=DWLS or METHOD=GLS, write the results in an OUTMODEL= data set, and use the results as initial values specified by an INMODEL= data set in a second run with METHOD=ML.
- Change the optimization technique. For example, if you use the default OMETHOD=LEVMAR, you can do either of the following:
  - Change to OMETHOD=QUANEW or to OMETHOD=NEWRAP.
  - Run some iterations with OMETHOD=CONGRA, write the results in an OUTMODEL= data set, and use the results as initial values specified by an INMODEL= data set in a second run with a different OMETHOD= technique.
- Change or modify the update technique or the line-search algorithm or both when using OMETHOD=QUANEW or OMETHOD=CONGRA. For example, if you use the default update formula and the default line-search algorithm, you can do any or all of the following:
  - Change the update formula with the UPDATE= option.
  - Change the line-search algorithm with the LIS= option.
  - Specify a more precise line search with the LSPRECISION= option, if you use LIS=2 or LIS=3.
- Add more iterations and function calls by using the MAXIT= and MAXFU= options.
- Change the initial values. For many categories of model specifications, PROC CALIS computes an appropriate set of initial values automatically. However, for some of the model specifications (for example, structural equations with latent variables on the left-hand side and manifest variables on the right-hand side), PROC CALIS might generate very obscure initial values. In these cases, you have to set the initial values yourself.
  - Increase the initial values of the variance parameters by one of the following ways:
    - Set the variance parameter values in the model specification manually.
    - Use the DEMPHAS= option to increase all initial variance parameter values.
  - Use a slightly different, but more stable, model to obtain preliminary estimates.
  - Use additional information to specify initial values, for example, by using other SAS software like the FACTOR, REG, SYSLIN, and MODEL (SYSNLIN) procedures for the modified, unrestricted model case.

Unidentified Model

The parameter vector $\boldsymbol{\Theta}$ in the structural model

$$\boldsymbol{\Sigma} = \boldsymbol{\Sigma}(\boldsymbol{\Theta})$$

is said to be identified in a parameter space G, if

$$\boldsymbol{\Sigma}(\boldsymbol{\Theta}) = \boldsymbol{\Sigma}(\tilde{\boldsymbol{\Theta}}), \quad \tilde{\boldsymbol{\Theta}} \in G$$

implies $\boldsymbol{\Theta} = \tilde{\boldsymbol{\Theta}}$. The parameter estimates that result from an unidentified model can be very far from the parameter estimates of a very similar but identified model. They are usually machine dependent. Do not use parameter estimates of an unidentified model as initial values for another run of PROC CALIS.

Singular Predicted Covariance Model Matrix

Sometimes you might inadvertently specify models with singular predicted covariance model matrices (for example, by fixing diagonal elements to zero). In such cases, you cannot compute maximum likelihood estimates (the ML function value F is not defined). Since singular predicted covariance model matrices can also occur temporarily in the minimization process, PROC CALIS tries in such cases to change the parameter estimates so that the predicted covariance model matrix becomes positive definite. This process does not always work well, especially if there are fixed instead of free diagonal elements in the predicted covariance model matrices. A famous example where you cannot compute ML estimates is a component analysis with fewer components than given manifest variables. See the section "FACTOR Statement" on page 1600 for more details. If you continue to obtain a singular predicted covariance model matrix after changing initial values and optimization techniques, then your model might be specified so that ML estimates cannot be computed.

Saving Computing Time

For large models, the most computing time is needed to compute the modification indices. If you do not really need the Lagrange multipliers or multiple Wald test indices (the univariate Wald test indices are the same as the t values), using the NOMOD option can save a considerable amount of computing time.

Predicted Covariance Matrices with Negative Eigenvalues

A covariance matrix cannot have negative eigenvalues, since a negative eigenvalue means that some linear combination of the variables has negative variance. PROC CALIS displays a warning if the predicted covariance matrix has negative eigenvalues but does not actually compute the eigenvalues. Sometimes this warning can be triggered by 0 or very small positive eigenvalues that appear negative because of numerical error. If you want to be sure that the predicted covariance matrix you are fitting can be considered to be a variance-covariance matrix, you can use the SAS/IML command *VAL=EIGVAL(U)* to compute the vector *VAL* of eigenvalues of matrix $\mathbf{U}$.

Negative R^2 Values

The estimated squared multiple correlations R^2 of the endogenous variables are computed using the estimated error variances:

$$R_i^2 = 1 - \frac{\widehat{var(\zeta_i)}}{\widehat{var(\eta_i)}}$$

When $\widehat{var(\zeta_i)} > \widehat{var(\eta_i)}$, R_i^2 is negative. This might indicate poor model fit or R square is an inappropriate measure for the model. For the latter case, for example, negative R square might be due to cyclical (nonrecursive) paths in the model so that the R square interpretation is not appropriate.

Displayed Output

The output of PROC CALIS includes the following:

- a list of basic modeling information such as: the data set, the number of records read and used in the raw data set, the number of observations assumed by the statistical analysis, and the model type. When a multiple-group analysis is specified, the groups and their corresponding models are listed. This output assumes at least the PSHORT option.
- a list of all variables in the models. This output is displayed by default or by the PINITIAL option. It will not be displayed when you use the PSHORT or the PSUMMARY option.

  Depending on the modeling language, the variable lists vary, as shown in the following:

  - COSAN: a list of the observed variables
  - FACTOR: a list of the variables and the factors
  - LINEQS, PATH, and RAM: a list of the endogenous and exogenous variables specified in the model
  - LISMOD: a list of x-, y-, $\boldsymbol{\xi}$-, and $\boldsymbol{\eta}$- variables specified in the model
  - MSTRUCT: a list of the manifest variables specified in the model
- initial model specification. This output is displayed by default or by the PINITIAL option. It will not be displayed when you use the PSHORT or the PSUMMARY option.

  Depending on the modeling language, the sets of output vary, as shown in the following:

  - COSAN: matrix equations for the covariance and mean structures, dimensions of the model matrices, and types and transformations of the model matrices. The initial values for free parameters, the fixed values, and the parameter names are also displayed in the matrices.
  - FACTOR: factor loading matrix, factor covariance matrix, intercepts, factor means, and error variances as specified initially in the model. The initial values for free parameters, the fixed values, and the parameter names are also displayed.

  - LINEQS: linear equations, variance and covariance parameters, and mean parameters as specified initially in the model. The initial values for free parameters, the fixed values, and the parameter names are also displayed.
  - LISMOD: all model matrices as specified initially in the model. The initial values for free parameters, the fixed values, and the parameter names are also displayed.
  - MSTRUCT: initial covariance matrix and mean vectors, with parameter names and initial values displayed.
  - PATH: the path list, variance and covariance parameters, intercept and mean parameters as specified initially in the model. The initial values for free parameters, the fixed values, and the parameter names are also displayed.
  - RAM: a list of parameters, their types, names, and initial values.

- mean and standard deviation of each manifest variable if you specify the SIMPLE option, as well as skewness and kurtosis if the DATA= data set is a raw data set and you specify the KURTOSIS option.

- various coefficients of multivariate kurtosis and the numbers of observations that contribute most to the normalized multivariate kurtosis if the DATA= data set is a raw data set and the KURTOSIS option is used or you specify at least the PRINT option. See the section "Measures of Multivariate Kurtosis" on page 1851 for more information.

- covariance coverage, variable coverage, average coverage of covariances and means, rank orders of the variable (mean) and covariance coverage, most frequent missing patterns in the input data set, and the means of the missing patterns when there are incomplete observations (with some missing values in the analysis variables) in the input raw data set and when you use METHOD=FIML or METHOD=LSFIML for estimation.

- covariance or correlation matrix to be analyzed and the value of its determinant if you specify the output option PCORR or PALL. A zero determinant indicates a singular data matrix. In this case, the generalized least squares estimates with default weight matrix $\mathbf{S}$ and maximum likelihood estimates cannot be computed.

- robust covariance matrix and its determinant and robust mean vector if you specify the output option PCORR or PALL for robust estimation with the ROBUST option. Raw data input is required.

- the weight matrix $\mathbf{W}$ or its inverse is displayed if GLS, WLS, or DWLS estimation is used and you specify the PWEIGHT or PALL option.

- initial estimation methods for generating initial estimates. This output is displayed by default. It will not be displayed when you use the PSHORT or the PSUMMARY option.

- vector of parameter names and initial values and gradients. This output is displayed by default, unless you specify the PSUMMARY or NOPRINT option.

- special features of the optimization technique chosen if you specify at least the PSHORT option.

- optimization history if at least the PSHORT option is specified. For more details, see the section "Use of Optimization Techniques" on page 1854.

- specific output requested by options in the NLOPTIONS statement; for example, parameter estimates, gradient, constraints, projected gradient, Hessian, projected Hessian, Jacobian of nonlinear constraints, estimated covariance matrix of parameter estimates, and information matrix. Note that the estimated covariance of parameter estimates and the information matrix are not printed for the ULS and DWLS estimation methods.

- tables that count the occurrences of problematic variance and covariance estimates in various predicted covariance matrices if you specify the COVDIAG or PALL option.

- fit summary table with various model fit test statistics or fit indices, and some basic modeling information. For the listing of fit indices and their definitions, see the section "Overall Model Fit Indices" on page 1827. Note that for ULS and DWLS estimation methods, many of those fit indices that are based on model fit χ^2 are not displayed. See the section "Overall Model Fit Indices" on page 1827 for details. This output can be suppressed by the NOPRINT option.

- fit comparison for multiple-group analysis. See the section "Individual Fit Indices for Multiple Groups" on page 1835 for the fit indices for group comparison. This output can be suppressed by the NOPRINT option.

- the predicted covariance matrix and its determinant and mean vector, if you specify the output option PCORR or PALL.

- outlier, leverage, and case-level residual analysis if you specify the RESIDUAL option (or at least the PRINT option) with raw data input.

- leverage and outlier plot, quantile and percentile plots of residual M-distances, distribution of residual M-distances, and residual on fit plots if you request ODS Graphics by using the relevant PLOTS= option.

- residual and normalized residual matrix if you specify the RESIDUAL option or at least the PRINT option. The variance standardized or asymptotically standardized residual matrix can be displayed also. The average residual and the average off-diagonal residual are also displayed. Note that normalized or asymptotically standardized residuals are not applicable for the ULS and DWLS estimation methods.

  See the section "Residuals in the Moment Matrices" on page 1825 for more details.

- rank order of the largest normalized residuals if you specify the RESIDUAL option or at least the PRINT option.

- bar chart of the normalized residuals if you specify the RESIDUAL option or at least the PRINT option.

- plotting of smoothed density functions of residuals if you request ODS Graphics by the relevant PLOTS= option.

- equations of linear dependencies among the parameters used in the model specification if the information matrix is recognized as singular at the final solution.

- the (unstandardized) estimation results for all types of models and the standardized results for all but the COSAN models. Except for ULS or DWLS estimates, the approximate standard errors, t values, and p values are also displayed. This output is displayed by default or if you specify the PESTIM option or at least the PSHORT option. If you specify the CI option, confidence intervals are also displayed.

Depending on the modeling language, the sets of output vary, as shown in the following:

- COSAN: all model matrices in the model.
- FACTOR: factor loading matrix, rotation matrix, rotated factor loading matrix (if rotation requested), factor covariance matrix, intercepts, factor means, and error variances in the model. Factor rotation matrix is printed for the unstandardized solution.
- LINEQS: linear equations, effects in equations, variance and covariance parameters, and mean parameters in the model.
- LISMOD: all model matrices in the model.
- MSTRUCT: covariance matrix and mean vectors.
- PATH: the path list, variance and covariance parameters, intercept and mean parameters.
- RAM: a list of parameters, their types, names, and initial values.

- squared multiple correlations table which displays the error variance, total variance, and the squared multiple correlation of each endogenous variable in the model. The total variances are the diagonal elements of the predicted covariance matrix. This output is displayed if you specify the PESTIM option or at least the PSHORT option.

- the total determination of all equations, the total determination of the latent equations, and the total determination of the manifest equations if you specify the PDETERM or the PALL option. See the section "Assessment of Fit" on page 1824 for more details. If you specify subsets of variables in the DETERM statements, the corresponding determination coefficients will also be shown. If one of the determinants in the formula for computing the determination coefficient is zero, the corresponding coefficient is displayed as the missing value '.'.

- the matrix of estimated covariances among the latent variables in the model if you specify the PLATCOV option or at least the PRINT option.

- the matrix of estimated covariances between latent and manifest variables in the model if you specify the PLATCOV option or at least the PRINT option.

- the vector of estimated means for the latent and manifest variables in the model if you specify the PLATCOV option or at least the PRINT option.

- the matrix **FSR** of latent variable scores regression coefficients if you specify the PLATCOV option or at least the PRINT option. The **FSR** matrix is a generalization of Lawley and Maxwell (1971, p. 109) factor scores regression matrix,

  $$\mathbf{FSR} = \hat{\mathbf{\Sigma}}_{yx}\hat{\mathbf{\Sigma}}_{xx}^{-1}$$

  where $\hat{\mathbf{\Sigma}}_{xx}$ is the $p \times p$ predicted covariance matrix among manifest variables and $\hat{\mathbf{\Sigma}}_{yx}$ is the $m \times p$ matrix of the predicted covariances between latent and manifest variables, with p being the number of manifest variables, and m being the number of latent variables. You can multiply the observed values by this matrix to estimate the scores of the latent variables used in your model.

- stability coefficient of reciprocal causation if you request the effect analysis by using the EFFPART or EFFPART option, and you must not use the NOPRINT option.

- the matrices for the total, direct, and indirect effects if you specify the EFFPART or EFFPART option or at least the PRINT option, and you must not use the NOPRINT option. Unstandardized and standardized effects are printed in separate tables. Standard errors for the all estimated effects are also included in the output. Additional tables for effects are available if you request specialized effect analysis in the EFFPART statements.

- the matrix of rotated factor loadings and the orthogonal transformation matrix if you specify the ROTATE= and PESTIM options or at least the PSHORT options. This output is available for the FACTOR models.

- factor scores regression matrix, if you specify the PESTIM option or at least the PSHORT option. The determination of manifest variables is displayed only if you specify the PDETERM option.

- univariate Lagrange multiplier indices if you specify the MODIFICATION (or MOD) or the PALL option. The value of a Lagrange multiplier (LM) index indicates the approximate drop in χ^2 when the corresponding fixed parameter in the original model is freely estimated. The corresponding probability (with $df = 1$) and the estimated change of the parameter value are printed. Ranking of the LM indices is automatically done for prescribed parameter subsets of the original model. The LM indices with greatest improvement of χ^2 model fit appear in the beginning of the ranking list. Note that LM indices are not applicable to the ULS and the DWLS estimation methods. See the section "Modification Indices" on page 1848 for more detail.

- matrices of univariate Lagrange multiplier (LM) indices if you specify the MODIFICATION (or MOD) or the PALL option, and the LMMAT option in the LMTESTS statement. These matrices are predefined in PROC CALIS, or you can specify them in the LMTESTS statements. If releasing a fixed parameter in the matrix would result in a singular information matrix, the string 'Singular' is displayed instead of the Lagrange multiplier index. If a fixed entry in the matrix is restricted by the model (for example, fixed ones for coefficients associated with error terms) or being excluded in the specified subsets in the LMTESTS statement, the string 'Excluded' is displayed. Note that matrices for LM indices are not printed for the ULS and the DWLS estimation methods. See the section "Modification Indices" on page 1848 for more detail.

- univariate Lagrange multiplier test indices for releasing equality constraints if you specify the MODIFICATION (or MOD) or the PALL option. Note that this output is not applicable to the ULS and the DWLS estimation methods. See the section "Modification Indices" on page 1848 for more detail.

- univariate Lagrange multiplier test indices for releasing active boundary constraints specified by the BOUNDS statement if you specify the MODIFICATION (or MOD) or the PALL option. Note that this output is not applicable to the ULS and the DWLS estimation methods. See the section "Modification Indices" on page 1848 for more detail.

- the stepwise multivariate Wald test for constraining estimated parameters to zero constants if the MODIFICATION (or MOD) or the PALL option is specified and the univariate probability is greater than the value specified in the PMW= option (default PMW=0.05). Note that this output is not applicable to the ULS and the DWLS estimation methods. See the section "Modification Indices" on page 1848 for more details.

- path diagrams if you specify the PLOTS=PATHDIAGRAM option or the PATHDIAGRAM statement.

ODS Table Names

PROC CALIS assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

There are numerous ODS tables in the CALIS procedure. The conditions for these ODS tables to display vary a lot. For convenience in presentation, the ODS tables for the PROC CALIS procedure are organized in the following categories:

- ODS tables for descriptive statistics, missing patterns, and residual analysis
- ODS tables for model specification and results
- ODS tables for supplementary model analysis
- ODS tables for modification indices
- ODS tables for optimization control and results

Many ODS tables are displayed when you set either a specialized option in a certain statement or a global display option in the PROC CALIS statement. Rather than requesting displays by setting specialized options separately, you can request a group of displays by using a global display option.

There are five global display levels, represented by five options: PALL (highest), PRINT, *default*, PSHORT, and PSUMMARY. The higher the level, the more output requested. The *default* printing level is in effect when you do not specify any other global printing options in the PROC CALIS statement. See the section "Global Display Options" on page 1540 for details.

In the following description of ODS tables whenever applicable, the lowest level of global printing options for an ODS table to print is listed. It is understood that global printing options at higher levels can also be used. For example, if PSHORT is the global display option to print an ODS table, you can also use PALL, PRINT, or *default*.

ODS Tables for Descriptive Statistics, Missing Patterns, and Residual Analysis

These ODS tables are group-oriented, meaning that each group has its own set of tables in the output. To display these tables in your output, you can set a specialized option in either the PROC CALIS or GROUP statement. If the specialized option is set in the PROC CALIS statement, it will apply to all groups. If the option is set in the GROUP statement, it will apply to the associated group only. Alternatively, you can set a global printing option in the PROC CALIS statement to print these tables. Either a specialized or a global printing option is sufficient to print the tables.

Table Names for Descriptive Statistics

ODS Table Name	Description	Specialized Option	Global Display Option
ContKurtosis	Contributions to kurtosis from observations	KURTOSIS	PRINT
CovRobust	Robust covariance matrix, with raw data input	PCORR[1]	PRINT
CovDetRobust	Determinant of the robust covariance matrix, with raw data input	PCORR[1]	PRINT
InCorr	Input correlation matrix	PCORR	PALL
InCorrDet	Determinant of the input correlation matrix	PCORR	PALL
InCov	Input covariance matrix	PCORR	PALL
InCovDet	Determinant of the input covariance matrix	PCORR	PALL
InMean	Input mean vector	PCORR	PALL
Kurtosis	Kurtosis, with raw data input	KURTOSIS	PRINT
MeanRobust	Robust mean vector, with raw data input	PCORR[1]	PRINT
PredCorr	Predicted correlation matrix	PCORR	PALL
PredCorrDet	Determinant of the predicted correlation matrix	PCORR	PALL
PredCov	Predicted covariance matrix	PCORR	PALL
PredCovDet	Determinant of the predicted covariance matrix	PCORR	PALL
PredCovDiagnostics	Counting problematic cases of variance and covariance estimates	COVDIAG	PALL
PredMean	Predicted mean vector	PCORR	PALL
RidgedCovRobust	Ridged robust covariance matrix, with raw data input	PCORR[1]	PRINT
RidgedCovDetRobust	Determinant of the ridged robust covariance matrix, with raw data input	PCORR[1]	PRINT
RidgedInCorr	Ridged input correlation matrix	PCORR	PALL
RidgedInCorrDet	Determinant of the ridged input correlation matrix	PCORR	PALL
RidgedInCov	Ridged input covariance matrix	PCORR	PALL
RidgedInCovDet	Determinant of the ridged input covariance matrix	PCORR	PALL
SimpleStatistics	Simple statistics, with raw data input	SIMPLE	Default
Weights	Weight matrix	PWEIGHT	PALL
WeightsDet	Determinant of the weight matrix	PWEIGHT	PALL

1. Robust estimation with the ROBUST option is required.

Table Names for Missing Pattern Analysis

ODS Table Name	Description	Specialized Option	Global Display Option
AveCoverage	Average proportion coverages of means (variances) and covariances	SIMPLE or PCORR[1]	Default[2]
MeanCovCoverage	Proportions of data present for means (variances) and covariances	SIMPLE or PCORR[1]	Default[2]
MissPatternsMeans	Means of the nonmissing and the most frequent missing patterns	SIMPLE or PCORR[1]	Default[2]
RankCovCoverage	Rank order of the covariance coverages	SIMPLE or PCORR[1]	Default[2]
RankMissPatterns	Rank order of the most frequent missing patterns	SIMPLE or PCORR[1]	Default[2]
RankVariableCoverage	Rank order of the proportion coverages of the variables	SIMPLE or PCORR[1]	Default[2]

1. You can use the NOMISSPAT option in the PROC CALIS statement to suppress the analytic output of the missing patterns. If you use the NOMISSPAT option in the GROUP statements, only the output of the missing pattern analysis for the corresponding groups are suppressed.
2. PROC CALIS outputs these tables by default only when there are incomplete observations in the data sets and you use METHOD=FIML or METHOD=LSFIML for estimation.

Table Names for Residual Displays

ODS Table Name	Description	Specialized Option	Global Display Option
Residuals in the Moment Matrices			
AsymStdRes	Asymptotically standardized residual matrix	RESIDUAL=ASYSTAND[1]	PALL
AveAsymStdRes	Average of absolute asymptotically standardized residual values	RESIDUAL=ASYSTAND[1]	PALL
AveNormRes	Average of absolute normalized residual values	RESIDUAL=NORM[1]	PALL
AveRawRes	Average of absolute raw residual values	RESIDUAL	PALL
AveVarStdRes	Average of absolute variance standardized residual values	RESIDUAL=VARSTAND[1]	PALL
DistAsymStdRes	Distribution of asymptotically standardized residuals	RESIDUAL=ASYSTAND[1]	PALL
DistNormRes	Distribution of normalized residuals	RESIDUAL=NORM[1]	PALL
DistRawRes	Distribution of raw residuals	RESIDUAL	PALL
DistVarStdRes	Distribution of variance standardized residuals	RESIDUAL=VARSTAND[1]	PALL

ODS Table Name	Description	Specialized Option	Global Display Option
NormRes	Normalized residual matrix	RESIDUAL=NORM[1]	PALL
RawRes	Raw residual matrix	RESIDUAL[2]	PALL
RankAsymStdRes	Rank order of asymptotically standardized residuals	RESIDUAL=ASYSTAND[1]	PALL
RankNormRes	Rank order of normalized residuals	RESIDUAL=NORM[1]	PALL
RankRawRes	Rank order of raw residuals	RESIDUAL	PALL
RankVarStdRes	Rank order of variance standardized residuals	RESIDUAL=VARSTAND[1]	PALL
VarStdRes	Variance standardized residual matrix	RESIDUAL=VARSTAND[1]	PALL
Case-Level Residuals[3]			
DepartResidualDist	Departures from the theoretical distribution for the residual M-distances	RESIDUAL	PALL
Leverage	Observations with largest leverage M-distances	RESIDUAL	PALL
Outlier	Observations with largest residual M-distances	RESIDUAL	PALL

1. The RESIDUAL= option specifies the RESIDUAL option and the type of residuals. For example, RESIDUAL=ASYSTAND requests asymptotically standardized residuals, in addition to the tables enabled by the RESIDUAL option. In some cases, the requested types of residuals are not available due to the specific estimation method or the data type used. When this occurs, PROC CALIS will determine the appropriate types of normalized or standardized residuals to display.
2. Raw residuals are also printed for correlation analysis even if RESIDUAL or PALL is not specified.
3. Raw data input is required.

ODS Tables for Model Specification and Results

Some ODS tables of this group are model-oriented. Others are not. Model-oriented ODS tables are printed for each model, while others are printed no more than once no matter how many models you have.

Non-Model-Oriented ODS Tables

The ODS tables that are not model-oriented are listed in the following table:

ODS Table Name	Description	Global Display Option	Additional Specification Required
AddParms	Estimates for additional parameters	PSHORT	PARAMETERS statement
AddParmsInit	Initial values for additional parameters	PSHORT	PARAMETERS statement
Fit	Fit summary	PSUMMARY	
GroupFit	Fit comparison among groups	PSUMMARY	Multiple groups
ModelingInfo	General modeling information	PSHORT	
ModelSummary	Summary of models and their labels and types	PSHORT	Multiple models[1]
ParmFunc	Parametric function testing	PSHORT	TESTFUNC statement
SimTest	Simultaneous tests of parametric functions	PSHORT	SIMTESTS statement

1. This table is displayed when you have multiple models that have labels specified by the LABEL= option, or when you define a model with more than a single level of reference by using the REFMODEL option. Otherwise, the ModelingInfo table contains all pertinent information regarding the models in the analysis.

Model-Oriented ODS Tables

These ODS tables are model-oriented, meaning that each model has its own set of ODS tables in the output. There are three types of model specification and results printing in PROC CALIS: initial specification, (unstandardized) estimated model results, and standardized model results. To distinguish these three types of ODS tables, different suffixes for the ODS table names are used. An "Init" suffix indicates initial specification, while a "Std" suffix indicates standardized solutions. All other tables are for unstandardized solutions.

These ODS tables require some specialized options to print. If you set the specialized option in the PROC CALIS statement, it applies to all models. If you set the specialized option in the MODEL statement, it applies to the associated model only. Alternatively, to print *all* these ODS tables, you can use the PSHORT or any higher level global display options in the PROC CALIS statement. Either a specialized or a global printing option is sufficient to print these ODS tables. The following is a summary of the specialized and global printing options for these three types of ODS tables:

Type of ODS Tables	Table Name Suffix	Specialized Option	Global Display Option
Initial specification	Init	PINITIAL	PSHORT
Unstandardized solutions	(none)	PESTIM	PSHORT
Standardized solutions	Std	PESTIM, and NOSTAND not used	PSHORT

In the following list of ODS tables, the prefixes of the ODS table names indicate the modeling language required for the ODS tables to print. The last column of the list indicates whether the PRIMAT option is needed to print the corresponding ODS tables in matrix formats. You can use the PRIMAT option either in the PROC CALIS or MODEL statement. If you want matrix output for all models, set this option in the

PROC CALIS statement. If you want matrix output for a specific model, set this option in the associated MODEL statement only.

ODS Table Name	Description	Additional Option
COSANVariables	Variables in the COSAN model	
COSANModel	Mean and covariance structure formulas	
COSANMatrixSummary	Summary of COSAN model matrices	
COSANMatrix	Estimated model matrix	
COSANMatrixInit	Initial model matrix	
FACTORCov	Estimated factor covariances	
FACTORCovInit	Initial factor covariances	
FACTORCovStd	Factor correlations	
FACTORErrVar	Estimated error variances	
FACTORErrVarInit	Initial error variances	
FACTORErrVarStd	Standardized results for error variances	
FACTORIntercepts	Estimated intercepts	
FACTORInterceptsInit	Initial intercepts	
FACTORLoadings	Estimated factor loadings	
FACTORLoadingsInit	Initial factor loadings	
FACTORLoadingsStd	Standardized factor loadings	
FACTORMeans	Estimated factor means	
FACTORMeansInit	Initial factor means	
FACTORRotCov	Estimated rotated factor covariances	
FACTORRotCovStd	Rotated factor correlations	
FACTORRotErrVar	Error variances in rotated solution	
FACTORRotErrVarStd	Standardized results for error variances in rotated solution	
FACTORRotLoadings	Rotated factor loadings	
FACTORRotLoadingsStd	Standardized rotated factor loadings	
FACTORRotMat	Rotation matrix	
FACTORScoresRegCoef	Factor scores regression coefficients	
FACTORVariables	Variables in the analysis	
LINEQSAlpha	Estimated intercept vector	PRIMAT
LINEQSAlphaInit	Initial intercept vector	PRIMAT
LINEQSBeta	Estimated _EQSBETA_ matrix	PRIMAT
LINEQSBetaInit	Initial _EQSBETA_ matrix	PRIMAT
LINEQSBetaStd	Standardized results for _EQSBETA_ matrix	PRIMAT
LINEQSCovExog	Estimated covariances among exogenous variables	
LINEQSCovExogInit	Initial covariances among exogenous variables	
LINEQSCovExogStd	Standardized results for covariances among exogenous variables	
LINEQSEffects	Estimated effects in equations	
LINEQSEffectsStd	Estimated standardized effects in equations	
LINEQSEq	Estimated equations	
LINEQSEqInit	Initial equations	
LINEQSEqStd	Standardized equations	

ODS Table Name	Description	Additional Option
LINEQSGamma	Estimated _EQSGAMMA_ matrix	PRIMAT
LINEQSGammaInit	Initial _EQSGAMMA_ matrix	PRIMAT
LINEQSGammaStd	Standardized results for _EQSGAMMA_ matrix	PRIMAT
LINEQSMeans	Estimated means for exogenous variables	
LINEQSMeansInit	Initial means for exogenous variables	
LINEQSNu	Estimated mean vector	PRIMAT
LINEQSNuInit	Initial mean vector	PRIMAT
LINEQSPhi	Estimated _EQSPHI_ matrix	PRIMAT
LINEQSPhiInit	Initial _EQSPHI_ matrix	PRIMAT
LINEQSPhiStd	Standardized results for _EQSPHI_ matrix	PRIMAT
LINEQSVarExog	Estimated variances of exogenous variables	
LINEQSVarExogInit	Initial variances of exogenous variables	
LINEQSVarExogStd	Standardized results for variances of exogenous variables	
LINEQSVariables	Exogenous and endogenous variables	
LISMODAlpha	Estimated _ALPHA_ vector	
LISMODAlphaInit	Initial _ALPHA_ vector	
LISMODBeta	Estimated _BETA_ matrix	
LISMODBetaInit	Initial _BETA_ matrix	
LISMODBetaStd	Standardized _BETA_ matrix	
LISMODGamma	Estimated _GAMMA_ matrix	
LISMODGammaInit	Initial _GAMMA_ matrix	
LISMODGammaStd	Standardized _GAMMA_ matrix	
LISMODKappa	Estimated _KAPPA_ vector	
LISMODKappaInit	Initial _KAPPA_ vector	
LISMODLambdaX	Estimated _LAMBDAX_ matrix	
LISMODLambdaXInit	Initial _LAMBDAX_ matrix	
LISMODLambdaXStd	Standardized _LAMBDAX_ matrix	
LISMODLambdaY	Estimated _LAMBDAY_ matrix	
LISMODLambdaYInit	Initial _LAMBDAY_ matrix	
LISMODLambdaYStd	Standardized _LAMBDAY_ matrix	
LISMODNuX	Estimated _NUX_ vector	
LISMODNuXInit	Initial _NUX_ vector	
LISMODNuY	Estimated _NUY_ vector	
LISMODNuYInit	Initial _NUY_ vector	
LISMODPhi	Estimated _PHI_ matrix	
LISMODPhiInit	Initial _PHI_ matrix	
LISMODPhiStd	Standardized _PHI_ matrix	
LISMODPsi	Estimated _PSI_ matrix	
LISMODPsiInit	Initial _PSI_ matrix	
LISMODPsiStd	Standardized _PSI_ matrix	
LISMODThetaX	Estimated _THETAX_ matrix	
LISMODThetaXInit	Initial _THETAX_ matrix	
LISMODThetaXStd	Standardized _THETAX_ matrix	
LISMODThetaY	Estimated _THETAY_ matrix	

ODS Table Name	Description	Additional Option
LISMODThetaYInit	Initial _THETAY_ matrix	
LISMODThetaYStd	Standardized _THETAY_ matrix	
LISMODVariables	Variables in the model	
MSTRUCTCov	Estimated _COV_ matrix	
MSTRUCTCovInit	Initial _COV_ matrix	
MSTRUCTCovStd	Standardized _COV_ matrix	
MSTRUCTMean	Estimated _MEAN_ vector	
MSTRUCTMeanInit	Initial _MEAN_ vector	
MSTRUCTVariables	Variables in the model	
PATHCovErrors	Estimated error covariances	
PATHCovErrorsInit	Initial error covariances	
PATHCovErrorsStd	Standardized error covariances	
PATHCovVarErr	Estimated covariances between exogenous variables and errors	
PATHCovVarErrInit	Initial covariances between exogenous variables and errors	
PATHCovVarErrStd	Standardized results for covariances between exogenous variables and errors	
PATHCovVars	Estimated covariances among exogenous variables	
PATHCovVarsInit	Initial covariances among exogenous variables	
PATHCovVarsStd	Standardized results for covariances among exogenous variables	
PATHList	Estimated path list	
PATHListInit	Initial path list	
PATHListStd	Standardized path list	
PATHMeansIntercepts	Estimated intercepts	
PATHMeansInterceptsInit	Initial intercepts	
PATHVariables	Exogenous and endogenous variables	
PATHVarParms	Estimated variances or error variances	
PATHVarParmsInit	Initial variances or error variances	
PATHVarParmsStd	Standardized results for variances or error variances	
RAMAMat	Estimated _A_ matrix	PRIMAT
RAMAMatInit	Initial _A_ matrix	PRIMAT
RAMAMatStd	Standardized results of _A_ matrix	PRIMAT
RAMList	List of RAM estimates	
RAMListInit	List of initial RAM estimates	
RAMListStd	Standardized results for RAM estimates	
RAMPMat	Estimated _P_ matrix	PRIMAT
RAMPMatInit	Initial _P_ matrix	PRIMAT
RAMPMatStd	Standardized results of _P_ matrix	PRIMAT
RAMVariables	Exogenous and endogenous variables	
RAMWVec	Estimated mean and intercept vector	PRIMAT
RAMWVecInit	Initial mean and intercept vector	PRIMAT

ODS Tables for Supplementary Model Analysis

These ODS tables are model-oriented. They are printed for each model in your analysis. To display these ODS tables, you can set some specialized *options* in either the PROC CALIS or MODEL statement. If the specialized options are used in the PROC CALIS statement, they apply to all models. If the specialized options are used in the MODEL statement, they apply to the associated model only. For some of these ODS tables, certain specialized *statements* for the model might also enable the printing. Alternatively, you can use the global printing options in the PROC CALIS statement to print these ODS tables. Either a specialized option (or statement) or a global printing option is sufficient to print a particular ODS table.

ODS Table Name	Description	Specialized Option or Statement	Global Display Option
Determination	Coefficients of determination	PDETERM DETERM[4]	Default
DirectEffects	Direct effects	EFFPART[1]	PRINT
DirectEffectsStd	Standardized direct effects	EFFPART[1,3]	PRINT
EffectsOf	Effects of the listed variables	EFFPART[2]	PRINT
EffectsOn	Effects on the listed variables	EFFPART[2]	PRINT
IndirectEffects	Indirect effects	EFFPART[1]	PRINT
IndirectEffectsStd	Standardized indirect effects	EFFPART[1,3]	PRINT
LatentScoresRegCoef	Latent variable scores regression coefficients	PLATCOV	PRINT
PredCovLatent	Predicted covariances among latent variables	PLATCOV	PRINT
PredCovLatMan	Predicted covariances between latent and manifest variables	PLATCOV	PRINT
PredMeanLatent	Predicted means of latent variables	PLATCOV	PRINT
SqMultCorr	Squared multiple correlations	PESTIM	PSHORT
Stability	Stability coefficient of reciprocal causation	PDETERM, DETERM[4]	Default
StdEffectsOf	Standardized effects of the listed variables	EFFPART[2,3]	PSHORT
StdEffectsOn	Standardized effects on the listed variables	EFFPART[2,3]	PSHORT
TotalEffects	Total effects	EFFPART[1]	PRINT
TotalEffectsStd	Standardized total effects	EFFPART[1,3]	PRINT

1. This refers to the EFFPART or TOTEFF option in the PROC CALIS or MODEL statement.
2. This refers to the EFFPART statement specifications.
3. NOSTAND option must not be specified in the MODEL or PROC CALIS statement.
4. PDETERM is an option specified in the PROC CALIS or MODEL statement, while DETERM is a statement name.

ODS Tables for Model Modification Indices

To print the ODS tables for model modification indices, you can use the MODIFICATION option in either the PROC CALIS or MODEL statement. When this option is set in the PROC CALIS statement, it applies to all models. When this option is set in the MODEL statement, it applies to the associated model only. Alternatively, you can also use the PALL option in the PROC CALIS statement to print these ODS tables.

If the NOMOD option is set in the PROC CALIS statement, these ODS tables are not printed for all models, unless the MODIFICATION is respecified in the individual MODEL statements. If the NOMOD option is set in the MODEL statement, then the ODS tables for modification do not print for the associated model.

For convenience in presentation, three different classes of ODS tables for model modifications are described in the following. First, ODS tables for ranking of LM indices are the default printing when the MODIFICATION option is specified. Second, ODS tables for LM indices in matrix forms require an additional option to print. Last, ODS tables for other modification indices, including the Wald test indices, require specific data-analytic conditions to print. While the first two classes of ODS tables are model-oriented (that is, each model has its own sets of output), the third one is not.

ODS Table Names for Ranking of LM Indices

Rankings of the LM statistics in different regions of parameter space are the default printing format when you specify the MODIFICATION option in the PROC CALIS or MODEL statement. You can also turn off these default printing by the NODEFAULT option in the LMTESTS statement for models. If you want to print matrices of LM test statistics rather than the rankings of LM test statistics, you can use the NORANK or MAXRANK=0 option in the LMTESTS statement.

These ODS tables for ranking LM statistics are specific to the types of modeling languages used. This is noted in the last column of the following table.

ODS Table Name	Description	Model
LMRankCosanMatrix	Any COSAN model matrix	COSAN
LMRankCov	Covariances among variables	MSTRUCT
LMRankCovErr	Covariances among errors	LINEQS
LMRankCovErrofVar	Covariances among errors of variables	PATH
LMRankCovExog	Covariances among existing exogenous variables	LINEQS or PATH
LMRankCovFactors	Covariance among factors	FACTOR
LMRankCustomSet	Customized sets of parameters defined in LMTESTS statements	any model
LMRankErrorVar	Error variances	FACTOR
LMRankFactMeans	Factor means	FACTOR
LMRankIntercepts	Intercepts	FACTOR, LINEQS, or PATH
LMRankLisAlpha	LISMOD _ALPHA_	LISMOD
LMRankLisBeta	LISMOD _BETA_	LISMOD
LMRankLisGamma	LISMOD _GAMMA_	LISMOD
LMRankLisKappa	LISMOD _KAPPA_	LISMOD
LMRankLisLambdaX	LISMOD _LAMBDAX_	LISMOD
LMRankLisLambdaY	LISMOD _LAMBDAY_	LISMOD
LMRankLisNuX	LISMOD _NUX_	LISMOD

ODS Table Name	Description	Model
LMRankLisNuY	LISMOD _NUY_	LISMOD
LMRankLisPhi	LISMOD _PHI_	LISMOD
LMRankLisPsi	LISMOD _PSI_	LISMOD
LMRankLisThetaX	LISMOD _THETAX_	LISMOD
LMRankLisThetaY	LISMOD _THETAY_	LISMOD
LMRankLoadings	Factor loadings	FACTOR
LMRankMeans	Means of existing variables	LINEQS, MSTRUCT, or PATH
LMRankPaths	All possible paths in the model	PATH
LMRankPathsFromEndo	Paths from existing endogenous variables	LINEQS
LMRankPathsFromExog	Paths from existing exogenous variables	LINEQS
LMRankPathsNewEndo	Paths to existing exogenous variables	LINEQS
LMRankRamA	_RAMA_ matrix	RAM
LMRankRamAlpha	_RAMALPHA_ matrix	RAM
LMRankRamNu	_RAMNU_ matrix	RAM
LMRankRamP11	_RAMP11_ matrix	RAM
LMRankRamP22	_RAMP22_ matrix	RAM

ODS Table Names for Lagrange Multiplier Tests in Matrix Form

To print matrices of LM test indices for a model, you must also use the LMMAT option in the LMTESTS statement for the model. Some of these matrices are printed by default, while others are printed only when certain regions of parameter are specified in the LM test sets. In the following tables, the ODS table names for LM test statistics in matrix form are listed for each model type.

The COSAN Model

ODS Table Name	Description	Selected Region in Test Sets
LMCosanMatrix	Any COSAN model matrix	(default)

The FACTOR Model

ODS Table Name	Description	Selected Region in Test Sets
LMFactErrv	Vector of error variances	FACTERRV (default)
LMFactFcov	Factor covariance matrix	FACTFCOV (default)
LMFactInte	Intercept vector	FACTINTE (default)
LMFactLoad	Factor loading matrix	FACTLOAD (default)
LMFactMean	Factor mean vector	FACTMEAN (default)

The LINEQS Model

ODS Table Name	Description	Selected Regions in Test Sets
LMEqsAlpha	_EQSALPHA_ vector	_EQSALPHA_ (default)

ODS Table Name	Description	Selected Regions in Test Sets
LMEqsBeta	_EQSBETA_ matrix	_EQSBETA_ (default)
LMEqsGammaSub	_EQSGAMMA_ matrix, excluding entries with error variables in columns	_EQSGAMMA_ (default)
LMEqsNewDep	New rows for expanding _EQSBETA_ and _EQSGAMMA_ matrices	NEWDEP
LMEqsNuSub	_EQSNU_ vector, excluding fixed zero means for error variables	_EQSNU_ (default)
LMEqsPhi	_EQSPHI_ matrix	_EQSPHI_ alone or _EQSPHI11_, _EQSPHI21_ and _EQSPHI22_ together
LMEqsPhi11	Upper left portion (exogenous variances and covariances) of the _EQSPHI_ matrix	_EQSPHI11_ (default)
LMEqsPhi21	Lower left portion (error variances and covariances) of the _EQSPHI_ matrix	_EQSPHI21_
LMEqsPhi22	Lower right portion (error variances and covariances) of the _EQSPHI_ matrix	_EQSPHI22_ (default)

The LISMOD Model

ODS Table Name	Description	Selected Regions in Test Sets
LMLisAlpha	LISMOD _ALPHA_ vector	_ALPHA_ (default)
LMLisBeta	LISMOD _BETA_ matrix	_BETA_ (default)
LMLisGamma	LISMOD _GAMMA_ matrix	_GAMMA_ (default)
LMLisKappa	LISMOD _KAPPA_ vector	_KAPPA_ (default)
LMLisLambdaX	LISMOD _LAMBDAX_ matrix	_LAMBDAX_ (default) or _LAMBDA_
LMLisLambdaY	LISMOD _LAMBDAY_ matrix	_LAMBDAY_ (default) or _LAMBDA_
LMLisNuX	LISMOD _NUX_ vector	_NUX_ (default) or _NU_
LMLisNuY	LISMOD _NUY_ vector	_NUY_ (default) or _NU_
LMLisPhi	LISMOD _PHI_ matrix	_PHI_ (default)
LMLisPsi	LISMOD _PSI_ matrix	_PSI_ (default)
LMLisThetaX	LISMOD _THETAX_ matrix	_THETAX_ (default) or _THETA_
LMLisThetaY	LISMOD _THETAY_ matrix	_THETAY_ (default) or _THETA_

The MSTRUCT Model

ODS Table Name	Description	Selected Regions in Test Sets
LMMstructCov	Covariance matrix	MSTRUCTCOV (default) or _COV_
LMMstructMean	Mean vector	MSTRUCTMEAN (default) or _MEAN_

The PATH Model

ODS Table Name	Description	Selected Regions in Test Sets
LMRamA	_RAMA_ matrix	ARROWS or _RAMA_ (default)
LMRamALeft	Left portion of the _RAMA_ matrix	_RAMA_LEFT_ alone or _RAMBETA_ and _RAMA_LL_ together
LMRamALL	Lower left portion of the _RAMA_ matrix	_RAMA_LL_
LMRamALower	Lower portion of the _RAMA_ matrix	NEWENDO or _RAMA_LOWER_ or _RAMA_LL_ and _RAMA_LR_ together
LMRamALR	Lower right portion of the _RAMA_ matrix	_RAMA_LR_
LMRamARight	Right portion of the _RAMA_ matrix	_RAMA_RIGHT_
LMRamAUpper	Upper portion of the _RAMA_ matrix	_RAMA_UPPER_ alone or _RAMBETA_ and _RAMGAMMA_ together
LMRamAlpha	_RAMALPHA_ matrix	INTERCEPTS (default)
LMRamBeta	Upper left portion of the _RAMA_ matrix	_RAMBETA_
LMRamGamma	Upper right portion of the _RAMA_ matrix	_RAMGAMMA_
LMRamNu	_RAMNU_ matrix	MEANS (default)
LMRamP	_RAMP_ matrix	_RAMP_ alone or _RAMP11_, _RAMP21_, and _RAMP22_ together
LMRamP11	Upper left portion of the _RAMP_ matrix	COVERR (default)
LMRamP21	Lower left portion of the _RAMP_ matrix	COVEXOGERR
LMRamP22	Lower right portion of the _RAMP_ matrix	COVEXOG (default)
LMRamW	_RAMW_ matrix	FIRSTMOMENTS alone or MEANS and INTERCEPTS together

The RAM Model

ODS Table Name	Description	Selected Regions in Test Sets
LMRamA	_RAMA_ matrix	_RAMA_ (default)
LMRamALeft	Left portion of the _RAMA_ matrix	_RAMA_LEFT_ alone or _RAMBETA_ and _RAMA_LL_ together
LMRamALL	Lower left portion of the _RAMA_ matrix	_RAMA_LL_
LMRamALower	Lower portion of the _RAMA_ matrix	NEWENDO or _RAMA_LOWER_ or _RAMA_LL_ and _RAMA_LR_ together
LMRamALR	Lower right portion of the _RAMA_ matrix	_RAMA_LR_
LMRamARight	Right portion of the _RAMA_ matrix	_RAMA_RIGHT_
LMRamAUpper	Upper portion of the _RAMA_ matrix	_RAMBETA_ and _RAMGAMMA_

ODS Table Name	Description	Selected Regions in Test Sets
LMRamAlpha	_RAMALPHA_ matrix	_RAMALPHA_ (default)
LMRamBeta	Upper left portion of the _RAMA_ matrix	_RAMBETA_
LMRamGamma	Upper right portion of the _RAMA_ matrix	_RAMGAMMA_
LMRamNu	_RAMNU_ matrix	_RAMNU_ (default)
LMRamP	_RAMP_ matrix	_RAMP_ alone or _RAMP11_, _RAMP21_, and _RAMP22_ together
LMRamP11	Upper left portion of the _RAMP_ matrix	_RAMP11_ (default)
LMRamP21	Lower left portion of the _RAMP_ matrix	_RAMP21_
LMRamP22	Lower right portion of the _RAMP_ matrix	_RAMP22_ (default)
LMRamW	_RAMW_ matrix	_RAMW_

ODS Table Names for Other Modification Indices

The following table shows the ODS tables for the remaining modification indices.

ODS Table Name	Description	Additional Requirement
LagrangeBoundary	LM tests for active boundary constraints	Presence of active boundary constraints
LagrangeDepParmEquality	LM tests for equality constraints in dependent parameters	Presence of equality constraints in dependent parameters
LagrangeEquality	LM tests for equality constraints	Presence of equality constraints in independent parameters
WaldTest	Wald tests for testing existing parameters equaling zeros	At least one insignificant parameter value

ODS Table for Optimization Control and Results

To display the ODS tables for optimization control and results, you must specify any of the following global display options in the PROC CALIS statement: PRINT, PALL, or default (that is, NOPRINT is not specified). Also, you must not use the NOPRINT option in the NLOPTIONS statement. For some of these tables, you must also specify additional options, either in the PROC CALIS or the NLOPTIONS statement. Some restrictions might apply. Additional options and restrictions are noted in the last column.

ODS Table Name	Description	Additional Option Required or Restriction
CovParm	Covariances of parameters	PCOVES[1] or PALL[2], restriction[3]
ConvergenceStatus	Convergence status	
DependParmsResults	Final dependent parameter estimates	Restriction[4]
DependParmsStart	Initial dependent parameter estimates	Restriction[4]
Information	Information matrix	PCOVES[1] or PALL[2], restriction[3]
InitEstMethods	Initial estimation methods	
InputOptions	Optimization options	PALL[2]
IterHist	Iteration history	
IterStart	Iteration start	
IterStop	Iteration stop	
Lagrange	First and second order Lagrange multipliers	PALL[2]
LinCon	Linear constraints	PALL[2], restriction[5]
LinConDel	Deleted constraints	PALL[2], restriction[5]
LinConSol	Linear constraints evaluated at solution	PALL[2], restriction[5]
LinDep	Linear dependencies of parameter estimates	Restriction[6]
ParameterEstimatesResults	Final estimates	
ParameterEstimatesStart	Initial estimates	
ProblemDescription	Problem description	
ProjGrad	Projected gradient	PALL[2]

1. PCOVES option is specified in the PROC CALIS statement.
2. PALL option is specified in the NLOPTIONS statement.
3. Estimation method must not be ULS or DWLS.
4. Existence of dependent parameters.
5. Linear equality or boundary constraints are imposed.
6. Existence of parameter dependencies during optimization, but not due to model specification.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

In the following table, ODS graph names and the options to display the graphs are listed.

Table 33.17 Graphs Produced by PROC CALIS

ODS Graph Name	Plot Description	Option
Distribution of Residuals in the Moment Matrices		
AsymStdResidualHistogram	Asymptotically standardized residuals	PLOTS=RESIDUALS and RESIDUAL=ASYMSTD, METHOD= is not ULS or DWLS
NormResidualHistogram	Normalized residuals	PLOTS=RESIDUALS and RESIDUAL=NORM
RawResidualHistogram	Raw residuals	PLOTS=RESIDUALS
VarStdResidualHistogram	Variance standardized residuals	PLOTS=RESIDUALS and RESIDUAL=VARSTD
Case-Level Residual Diagnostics		
CaseResidualHistogram	Distribution of the residual M-distances	PLOTS=CRESHIST or CASERESIDUAL
ResidualByLeverage	Residual M-distances against the leverage M-distances	PLOTS=RESBYLEV or CASERESIDUAL
ResidualByPredicted	Residuals against the predicted values of dependent variables	PLOTS=RESBYPRED or CASERESIDUAL
ResidualByQuantile	Residual M-distances against the theoretical quantiles	PLOTS=QQ or CASERESIDUAL
ResPercentileByExpPercentile	Percentiles of residual M-distances against the theoretical percentiles	PLOTS=PP or CASERESIDUAL
Path Diagrams		
PathDiagramInit	Path diagram for initial model specifications	DIAGRAM=INITIAL in the PATHDIAGRAM statement
PathDiagram	Path diagram for the unstandardized solution	PLOTS=PATHDIAGRAM or PATHDIAGRAM statement
PathDiagramStand	Path diagram for the standardized solution	DIAGRAM=STANDARD in the PATHDIAGRAM statement
PathDiagramStructInit	Path diagram for initial structural model specifications	DIAGRAM=INITIAL and STRUCTURAL in the PATHDIAGRAM statement
PathDiagramStruct	Path diagram for the unstandardized solution of the structural model	STRUCTURAL in the PATHDIAGRAM statement
PathDiagramStructStand	Path diagram for the standardized solution of the structural model	DIAGRAM=STANDARD and STRUCTURAL in the PATHDIAGRAM statement

Examples: CALIS Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 33.1: Estimating Covariances and Correlations

(View the complete code for this example (calex101.sas) in the example repository.)

This example shows how you can use PROC CALIS to estimate the covariances and correlations of the variables in your data set. Estimating the covariances introduces you to the most basic form of covariance structures—a saturated model with all variances and covariances as parameters in the model. To fit such a saturated model when there is no need to specify the functional relationships among the variables, you can use the MSTRUCT modeling language of PROC CALIS.

The following data set contains four variables q1–q4 for the quarterly sales (in millions) of a company. The 14 observations represent 14 retail locations in the country. The input data set is shown in the following DATA step:

```
data sales;
   input q1 q2 q3 q4;
   datalines;
1.03   1.54   1.11   2.22
1.23   1.43   1.65   2.12
3.24   2.21   2.31   5.15
1.23   2.35   2.21   7.17
 .98   2.13   1.76   2.38
1.02   2.05   3.15   4.28
1.54   1.99   1.77   2.00
1.76   1.79   2.28   3.18
1.11   3.41   2.20   3.21
1.32   2.32   4.32   4.78
1.22   1.81   1.51   3.15
1.11   2.15   2.45   6.17
1.01   2.12   1.96   2.08
1.34   1.74   2.16   3.28
;
```

Use the following PROC CALIS specification to estimate a saturated covariance structure model with all variances and covariances as parameters:

```
proc calis data=sales pcorr;
   mstruct var=q1-q4;
run;
```

In the PROC CALIS statement, specify the data set with the DATA= option. Use the PCORR option to display the observed and predicted covariance matrix. Next, use the MSTRUCT statement to fit a covariance matrix of the variables that are provided in the VAR= option. Without further specifications such as the MATRIX statement, PROC CALIS assumes all elements in the covariance matrix are model parameters. Hence, this is a saturated model.

Output 33.1.1 shows the modeling information. Information about the model is displayed: the name and location of the data set, the number of data records read and used, and the number of observations in the analysis. The number of data records read is the actual number of records (or observations) that PROC CALIS processes from the data set. The number of data records used might or might not be the same as the actual number of records read from the data set. For example, records with missing values are read but not used in the analysis for the default maximum likelihood (ML) method. The number of observations refers to the N used for testing statistical significance and model fit. This number might or might not be the same as the number of records used for at least two reasons. First, if you use a frequency variable in the FREQ statement, the number of observations used is a weighted sum of the number of records, with the frequency variable being the weight. Second, if you use the NOBS= option in the PROC CALIS statement, you can override the number of observations that are used in the analysis. Because the current data set does not have any missing data and there are no frequency variables or an NOBS= option specified, these three numbers are all 14.

The model type is MSTRUCT because you use the MSTRUCT statement to define your model. The analysis type is covariances, which is the default. Output 33.1.1 then shows the four variables in the covariance structure model.

Output 33.1.1 Modeling Information of the Saturated Covariance Structure Model for the Sales Data

Estimating the Covariance Matrix by the MSTRUCT Modeling Language

The CALIS Procedure
Covariance Structure Analysis: Model and Initial Values

Modeling Information	
Maximum Likelihood Estimation	
Data Set	WORK.SALES
N Records Read	14
N Records Used	14
N Obs	14
Model Type	MSTRUCT
Analysis	Covariances

Variables in the Model
q1 q2 q3 q4
Number of Variables = 4

Output 33.1.2 shows the initial covariance structure model for these four variables. All lower triangular elements (including the diagonal elements) of the covariance matrix are parameters in the model. PROC CALIS generates the names for these parameters: _Add01–_Add10. Because the covariance matrix is symmetric, all upper triangular elements of the matrix are redundant. The initial estimates for covariance are denoted by missing values no initial values were specified.

Output 33.1.2 Initial Saturated Covariance Structure Model for the Sales Data

Initial MSTRUCT _COV_ Matrix				
	q1	q2	q3	q4
q1	. [_Add01]	. [_Add02]	. [_Add04]	. [_Add07]
q2	. [_Add02]	. [_Add03]	. [_Add05]	. [_Add08]
q3	. [_Add04]	. [_Add05]	. [_Add06]	. [_Add09]
q4	. [_Add07]	. [_Add08]	. [_Add09]	. [_Add10]

The PCORR option in the PROC CALIS statement displays the sample covariance matrix in Output 33.1.3. By default, PROC CALIS computes the unbiased sample covariance matrix (with variance divisor equal to $N - 1$) and uses it for the covariance structure analysis.

Output 33.1.3 Sample Covariance Matrix for the Sales Data

Covariance Matrix (DF = 13)				
	q1	q2	q3	q4
q1	0.33830	0.00020	0.03610	0.22137
q2	0.00020	0.22466	0.12653	0.24425
q3	0.03610	0.12653	0.60633	0.63012
q4	0.22137	0.24425	0.63012	2.66552

The fit summary and the fitted covariance matrix are shown in Output 33.1.4 and Output 33.1.5, respectively.

Output 33.1.4 Fit Summary of the Saturated Covariance Structure Model for the Sales Data

Fit Summary	
Chi-Square	0.0000
Chi-Square DF	0
Pr > Chi-Square	.

Output 33.1.5 Fitted Covariance Matrix for the Sales Data

MSTRUCT _COV_ Matrix: Estimate/StdErr/t-value/p-value				
	q1	**q2**	**q3**	**q4**
q1	0.3383	0.000198	0.0361	0.2214
	0.1327	0.0765	0.1260	0.2704
	2.5495	0.002587	0.2865	0.8186
	0.0108	0.9979	0.7745	0.4130
q2	0.000198	0.2247	0.1265	0.2443
	0.0765	0.0881	0.1082	0.2251
	0.002587	2.5495	1.1693	1.0853
	0.9979	0.0108	0.2423	0.2778
q3	0.0361	0.1265	0.6063	0.6301
	0.1260	0.1082	0.2378	0.3935
	0.2865	1.1693	2.5495	1.6012
	0.7745	0.2423	0.0108	0.1093
q4	0.2214	0.2443	0.6301	2.6655
	0.2704	0.2251	0.3935	1.0455
	0.8186	1.0853	1.6012	2.5495
	0.4130	0.2778	0.1093	0.0108

In Output 33.1.4, the model fit chi-square is 0 (df = 0). The p-value cannot be computed because the degrees of freedom is zero. This fit is perfect because the model is saturated.

Output 33.1.5 shows the fitted covariance matrix, along with standard error estimates and t values in each cell. The variance and covariance estimates match exactly those of the sample covariance matrix shown in Output 33.1.3.

A common practice for determining statistical significance for estimates in structural equation modeling is to require the absolute value of t to be greater than 1.96, which is the critical value of a standard normal variate at α=0.05. While all diagonal elements in Output 33.1.5 show statistical significance, all off-diagonal elements are not significantly different from zero. The t values for these elements range from 0.002 to 1.601.

Output 33.1.6 shows the standardized estimates of the variance and covariance elements. This is also the correlation matrix under the MSTRUCT model. Standard error estimates and t values are computed with the correlation estimates. Note that because the diagonal element values are fixed at 1, no standard errors or t values are shown.

Output 33.1.6 Standardized Covariance Matrix for the Sales Data

Standardized MSTRUCT _COV_ Matrix: Estimate/StdErr/t-value/p-value				
	q1	**q2**	**q3**	**q4**
q1	1.0000	0.000717	0.0797	0.2331
		0.2773	0.2756	0.2623
		0.002587	0.2892	0.8888
		0.9979	0.7724	0.3741
q2	0.000717	1.0000	0.3428	0.3156
	0.2773		0.2448	0.2497
	0.002587		1.4008	1.2640
	0.9979		0.1613	0.2062
q3	0.0797	0.3428	1.0000	0.4957
	0.2756	0.2448		0.2092
	0.2892	1.4008		2.3692
	0.7724	0.1613		0.0178
q4	0.2331	0.3156	0.4957	1.0000
	0.2623	0.2497	0.2092	
	0.8888	1.2640	2.3692	
	0.3741	0.2062	0.0178	

Sometimes researchers do not need to estimate the standard errors that are in their models. You can suppress the standard error and *t* value computations by using the NOSTDERR option in the PROC CALIS statement:

```
proc calis data=sales nose;
   mstruct var=q1-q4;
run;
```

Output 33.1.7 shows the fitted covariance matrix with the NOSE option. These values are exactly the same as in the sample covariance matrix shown in Output 33.1.3.

Output 33.1.7 Fitted Covariance Matrix without Standard Error Estimates for the Sales Data

MSTRUCT _COV_ Matrix				
	q1	**q2**	**q3**	**q4**
q1	0.3383	0.000198	0.0361	0.2214
q2	0.000198	0.2247	0.1265	0.2443
q3	0.0361	0.1265	0.6063	0.6301
q4	0.2214	0.2443	0.6301	2.6655

This example shows a very simple application of PROC CALIS: estimating the covariance matrix with standard error estimates. The covariance structure model is saturated. Several extensions of this very simple model are possible. To estimate the means and covariances simultaneously, see Example 33.2. To fit nonsaturated covariance structure models with certain hypothesized patterns, see Example 33.3 and Example 33.4. To fit structural models with implied covariance structures that are based on specified functional relationships among variables, see Example 33.6.

Example 33.2: Estimating Covariances and Means Simultaneously

(View the complete code for this example (calex102.sas) in the example repository.)

This example uses the same data set that is used in Example 33.1 and estimates the means and covariances. Use the MSTRUCT model specification as shown in the following statements:

```
proc calis data=sales meanstr nostand;
   mstruct var=q1-q4;
run;
```

In the PROC CALIS statement, specify the MEANSTR option to request the mean structure analysis in addition to the default covariance structure analysis. If you are not interested in the standardized solution, specify the NOSTAND option in the PROC CALIS statement to suppress computation of the standardized estimates. Without further model specification (such as the MATRIX statement), PROC CALIS assumes a saturated structural model with all means, variances, and covariances as model parameters.

Output 33.2.1 shows the modeling information. With the MEANSTR option specified in the PROC CALIS statement, the current analysis type is Means and Covariances, instead of the default Covariances in Example 33.1.

Output 33.2.1 Modeling Information of the Saturated Mean and Covariance Structure Model for the Sales Data

Saturated Means and Covariance Structures Using MSTRUCT

The CALIS Procedure
Mean and Covariance Structures: Model and Initial Values

Modeling Information	
Maximum Likelihood Estimation	
Data Set	WORK.SALES
N Records Read	14
N Records Used	14
N Obs	14
Model Type	MSTRUCT
Analysis	Means and Covariances

Variables in the Model
q1 q2 q3 q4
Number of Variables = 4

Output 33.2.2 shows the fit summary of the current model. Again, this is a perfect model fit with 0 chi-square value and 0 degrees of freedom.

Output 33.2.2 Fit Summary of the Saturated Mean and Covariance Structure Model for the Sales Data

Fit Summary	
Chi-Square	0.0000
Chi-Square DF	0
Pr > Chi-Square	.

Output 33.2.3 shows the estimates of the means, together with the standard error estimates and the *t* values. These estimated means are exactly the same as the sample means, which are not shown here.

Output 33.2.3 Mean Estimates for the Sales Data

MSTRUCT _Mean_ Vector				
Variable	Estimate	Standard Error	t Value	Pr > |t|
q1	1.36714	0.16132	8.4749	<.0001
q2	2.07429	0.13146	15.7790	<.0001
q3	2.20286	0.21596	10.2001	<.0001
q4	3.65500	0.45281	8.0718	<.0001

Output 33.2.4 shows the variance and covariance estimates. These estimates are exactly the same as the elements in the sample covariance matrix. In addition, these estimates match the estimates in Output 33.1.5 of Example 33.1, where only the covariance structures are analyzed.

Output 33.2.4 Variance and Covariance Estimates for the Sales Data

MSTRUCT _COV_ Matrix: Estimate/StdErr/t-value/p-value				
	q1	**q2**	**q3**	**q4**
q1	0.3383	0.000198	0.0361	0.2214
	0.1327	0.0765	0.1260	0.2704
	2.5495	0.002587	0.2865	0.8186
	0.0108	0.9979	0.7745	0.4130
q2	0.000198	0.2247	0.1265	0.2443
	0.0765	0.0881	0.1082	0.2251
	0.002587	2.5495	1.1693	1.0853
	0.9979	0.0108	0.2423	0.2778
q3	0.0361	0.1265	0.6063	0.6301
	0.1260	0.1082	0.2378	0.3935
	0.2865	1.1693	2.5495	1.6012
	0.7745	0.2423	0.0108	0.1093
q4	0.2214	0.2443	0.6301	2.6655
	0.2704	0.2251	0.3935	1.0455
	0.8186	1.0853	1.6012	2.5495
	0.4130	0.2778	0.1093	0.0108

These estimates are essentially the same as the sample means, variances, and covariances. This kind of analysis is much easier using PROC CORR with the NOMISS option. However, the main purpose of Example 33.1 and Example 33.2 is to introduce the MSTRUCT modeling language and some basic but important options in PROC CALIS. You can apply the MSTRUCT modeling language to more sophisticated situations that are beyond the saturated mean and covariance structure models. Example 33.3 and Example 33.4 fit some patterned covariance models that are nonsaturated. Also, options such as NOSTDERR, NOSTAND, and MEANSTR are useful for all modeling languages in PROC CALIS.

Example 33.3: Testing Uncorrelatedness of Variables

(View the complete code for this example (calex104.sas) in the example repository.)

This example uses the sales data in Example 33.1 and tests the uncorrelatedness of the variables in the model by using the MSTRUCT model specification. With the multivariate normality assumption, this is also the test of independence of the variables. The MATRIX statement defines the parameters in the model.

The uncorrelatedness model assumes that the correlations or covariances among the four variables are zero. Therefore, only the four diagonal elements of the covariance matrix, which represent the variances of the variables, are free parameters in the covariance structure model. To specify these parameters, use the MATRIX statement with the MSTRUCT model specification:

```
proc calis data=sales;
   mstruct var=q1-q4;
   matrix _cov_ [1,1], [2,2], [3,3], [4,4];
run;
```

Example 33.1 specifies exactly the same MSTRUCT statement for the four variables. The difference here is the addition of the MATRIX statement. Without a MATRIX statement, the MSTRUCT model assumes that all nonredundant elements in the covariance matrix are model parameters. This assumption is not the case in the current specification. The MATRIX statement specification for the covariance matrix (denoted by the _cov_ keyword) specifies four free parameters on the diagonal of the covariance matrix: `[1,1]`, `[2,2]`, `[3,3]`, and `[4,4]`. All other unspecified elements in the covariance matrix are fixed zeros by default.

The uncorrelatedness model is displayed in the output for the initial model specification. Output 33.3.1 shows that all off-diagonal elements of the covariance matrix are fixed zeros while the diagonal elements are missing and labeled with _Parm1–_Parm4. PROC CALIS generates these parameter names automatically and estimates these four parameters in the analysis.

Output 33.3.1 Initial Uncorrelatedness Model for the Sales Data

Initial MSTRUCT _COV_ Matrix				
	q1	**q2**	**q3**	**q4**
q1	. [_Parm1]	0	0	0
q2	0	. [_Parm2]	0	0
q3	0	0	. [_Parm3]	0
q4	0	0	0	. [_Parm4]

Output 33.3.2 shows the model fit chi-square test of the uncorrelatedness model. The chi-square is 6.528 ($df = 6$, $p = 0.3667$), which is not significant. This means that you fail to reject the uncorrelatedness model. In other words, the data is consistent with the uncorrelatedness model (zero covariances or correlations among the quarterly sales).

Output 33.3.2 Fit Summary of the Uncorrelatedness Model for the Sales Data

Fit Summary	
Chi-Square	6.5280
Chi-Square DF	6
Pr > Chi-Square	0.3667

Output 33.3.3 shows the estimates of the covariance matrix under the uncorrelatedness model, together with standard error estimates and t values. All off-diagonal elements are fixed zeros in the estimation results.

Output 33.3.3 Estimates of Variance under the Uncorrelatedness Model for the Sales Data

MSTRUCT _COV_ Matrix: Estimate/StdErr/t-value/p-value				
	q1	**q2**	**q3**	**q4**
q1	0.3383 0.1327 2.5495 0.0108 [_Parm1]	0	0	0
q2	0	0.2247 0.0881 2.5495 0.0108 [_Parm2]	0	0
q3	0	0	0.6063 0.2378 2.5495 0.0108 [_Parm3]	0
q4	0	0	0	2.6655 1.0455 2.5495 0.0108 [_Parm4]

This example shows how to specify free parameters in the MSTRUCT model by using the MATRIX statement. To specify the covariance matrix, use the _COV_ keyword in the MATRIX statement. To specify the parameters in the mean structures, you need use an additional MATRIX statement with the _MEAN_ keyword.

Two important notes regarding the MSTRUCT model specification are now in order:

- When you use the MSTRUCT statement without any MATRIX statements, *all* elements in the covariance matrix are *free parameters* in the model (for example, see Example 33.1). However, if the MATRIX statement includes at least one free or fixed parameter in the covariance matrix, PROC CALIS assumes that all other unspecified elements in the covariance matrix are *fixed zeros* (such as the current example).

- Using parameter names in the MATRIX statement specification is optional. In the context of the current example, naming the parameters is optional because there is no need to refer to them anywhere in the specification. PROC CALIS automatically generates unique names for these parameters. Alternatively, you can specify your own parameter names in the MATRIX statement. Naming parameters is not

only useful for references, but is also indispensable when you need to constrain model parameters by referring to their names. See Example 33.4 to use parameter names to define a covariance pattern.

Example 33.4: Testing Covariance Patterns

(View the complete code for this example (calex105.sas) in the example repository.)

In the test for sphericity, a covariance matrix is hypothesized to be a constant multiple of an identity matrix. That is, the null hypothesis for the population covariance matrix is

$$\mathbf{\Sigma} = \sigma^2 \mathbf{I}$$

where σ^2 is an unknown positive constant and $\mathbf{I}$ is an identity matrix. When this covariance pattern is applied to the sales data in Example 33.1, this hypothesis states that all four variables have the same variance σ^2 and are uncorrelated with each other. This model is more restricted than the uncorrelatedness model in Example 33.3, which requires uncorrelatedness but does not require equal variances. Use the following specification to conduct a sphericity test for the sales data:

```
proc calis data=sales;
   mstruct var=q1-q4;
   matrix _cov_ [1,1] = 4*sigma_sq;
run;
```

This specification is similar to that of Example 33.3. The major difference is the MATRIX statement specification. The current example uses a parameter name sigma_sq to represent the unknown variance parameter σ^2, whereas Example 33.3 specifies only the locations of the four free variance parameters.

The current MATRIX statement specification uses a shorthand notation. On the left-hand side of the equal sign, `[1,1]` indicates the starting location of the covariance matrix. The matrix entries automatically proceed to `[2,2]`, `[3,3]` and so on, depending on the length of the parameter list specified on the right-hand sign of the equal sign. For example, if there is just one parameter on the right-hand side, the matrix specification contains only `[1,1]`. In the current example, the specification `4*sigma_sq` means that sigma_sq appears four times in the specification. As a result, the preceding MATRIX statement specification is equivalent to the following statement:

```
matrix _cov_ [1,1] = sigma_sq,
             [2,2] = sigma_sq,
             [3,3] = sigma_sq,
             [4,4] = sigma_sq;
```

This matrix is what is required by the sphericity test. Use either the expanded notation or the shorthand notation for specifying the covariance pattern. For details about various types of shorthand notation for parameter specifications, see the MATRIX statement.

Output 33.4.1 shows the initial model specification under the test of sphericity. All the diagonal elements are labeled with the same name sigma_sq, indicating that they are the same parameter.

Output 33.4.1 Covariance Model under Sphericity for the Sales Data

Initial MSTRUCT _COV_ Matrix				
	q1	q2	q3	q4
q1	. [sigma_sq]	0	0	0
q2	0	. [sigma_sq]	0	0
q3	0	0	. [sigma_sq]	0
q4	0	0	0	. [sigma_sq]

Output 33.4.2 shows that the model fit chi-square is 31.5951 ($df = 9$, $p = 0.0002$). This means that the covariance pattern under the sphericity hypothesis is not supported.

Output 33.4.2 Fit Summary of the Sphericity Test for the Sales Data

Fit Summary	
Chi-Square	31.5951
Chi-Square DF	9
Pr > Chi-Square	0.0002

Output 33.4.3 shows the estimated covariance matrix under the sphericity hypothesis. The variance estimate for all four diagonal elements is 0.9587 (standard error=0.1880).

Output 33.4.3 Fitted Covariance Matrix under the Sphericity Hypothesis for the Sales Data

MSTRUCT _COV_ Matrix: Estimate/StdErr/t-value/p-value				
	q1	q2	q3	q4
q1	0.9587 0.1880 5.0990 <.0001 [sigma_sq]	0	0	0
q2	0	0.9587 0.1880 5.0990 <.0001 [sigma_sq]	0	0
q3	0	0	0.9587 0.1880 5.0990 <.0001 [sigma_sq]	0
q4	0	0	0	0.9587 0.1880 5.0990 <.0001 [sigma_sq]

This example shows how you can specify a simple covariance pattern by using the MATRIX statement. Use the same parameter names to constrain variance parameters that are supposed to be the same under the model. Constraining parameters by using the same parameter names is applicable not only to the MSTRUCT

models, but also to more complicated covariance structure models, such as multiple-group modeling (see Example 33.19 and Example 33.21).

The MSTRUCT modeling language is handy when you can directly specify the covariance pattern or structures in your model. However, in most applications of structural equation modeling, it is difficult to specify such direct covariance structures. Instead, the covariance structures are usually implied from the functional relationships among the variables in the model. Using the MSTRUCT modeling language in such a situation is not easy. Fortunately, PROC CALIS supports other modeling languages that enable you to specify the functional relationships among variables. The functional relationships can be in the form of a set of path-like descriptions, a system of linear equations, or parameter specifications in matrices. See Example 33.6 for an introduction to using the PATH modeling language for specifying path models.

Example 33.5: Testing Some Standard Covariance Pattern Hypotheses

(View the complete code for this example (calex106.sas) in the example repository.)

In Example 33.3, you test the uncorrelatedness of variables by using the MSTRUCT model specification. In Example 33.4, you test the sphericity of the covariance matrix by using the same model specification technique. In both examples, you need to specify the parameters in the covariance structure model explicitly by using the MATRIX statements.

Some covariance patterns are well-known in multivariate statistics, including the two tests in Example 33.3 and Example 33.4. To facilitate the tests of these "standard" covariance patterns, PROC CALIS provides the COVPATTERN= option to specify those standard covariance patterns more efficiently. With the COVPATTERN=option, you do not need to use the MSTRUCT and MATRIX statements to specify the covariance patterns explicitly. See the COVPATTERN= option for the supported covariance patterns. This example illustrates the use of the COVPATTERN= option.

In Example 33.3, you conduct a test of uncorrelatedness for the four variables in the sales data (see Example 33.1 for the data set). That is, the variables are hypothesized to be uncorrelated and only the four variances on the diagonal of the covariance matrix are population parameters of interest. The null hypothesis for the population covariance matrix is

$$\mathbf{\Sigma} = \begin{pmatrix} \mathrm{x} & 0 & 0 & 0 \\ 0 & \mathrm{x} & 0 & 0 \\ 0 & 0 & \mathrm{x} & 0 \\ 0 & 0 & 0 & \mathrm{x} \end{pmatrix}$$

where each x represents a distinct parameter (that is, the diagonal elements are not constrained with each other). You can test the diagonal covariance pattern easily by the following specification:

```
proc calis data=sales covpattern=diag;
run;
```

The COVPATTERN=DIAG option specifies the required diagonal covariance pattern for the test. PROC CALIS then sets up the covariance structures automatically. Output 33.5.1 shows the initial specification of the covariance pattern. As required, only the diagonal elements are parameters in the test and the other elements are fixed to zero. PROC CALIS names the variance parameters automatically—that is, _varparm_1–_varparm_4 are the four parameters for the variances. This is the same pattern as shown in Output 33.3.1 of Example 33.3, although the parameter names are different.

Output 33.5.1 Initial Diagonal Pattern for the Covariance Matrix of the Sales Data

Initial MSTRUCT _COV_ Matrix				
	q1	**q2**	**q3**	**q4**
q1	. [_varparm_1]	0	0	0
q2	0	. [_varparm_2]	0	0
q3	0	0	. [_varparm_3]	0
q4	0	0	0	. [_varparm_4]

Output 33.5.2 shows the results of the chi-square test of the diagonal covariance pattern. The chi-square is 5.44 ($df = 6$, $p = 0.4887$), which is not significant. You fail to reject the null hypothesis of the diagonal covariance pattern in the population.

Output 33.5.2 Fit Summary of the Diagonal Covariance Pattern Test for the Sales Data

Fit Summary	
Chi-Square	5.4400
Chi-Square DF	6
Pr > Chi-Square	0.4887

The numerical results shown in Output 33.5.2 are different from those of the same test by using the MSTRUCT model specification, which is shown in Output 33.3.2 of Example 33.3, although you do not reject the null hypothesis in both cases. The reason is that with the use of COVPATTERN= option, PROC CALIS applies the appropriate chi-square correction to the test statistic automatically. In the current example, the chi-square correction due to Bartlett (1950) has been applied. Test results with chi-square corrections are theoretically more accurate.

To obtain the same numerical results as those in Output 33.3.2, you can turn off the chi-square correction by using the CHICORRECT=0 option, as shown in the following specification:

```
proc calis data=sales covpattern=diag chicorrect=0;
run;
```

Output 33.5.3 shows the fit summary results without any chi-square correction. The numerical results match exactly to those shown in Output 33.3.2 of Example 33.3.

Output 33.5.3 Fit Summary of the Diagonal Covariance Pattern Test for the Sales Data: No Chi-Square Correction

Fit Summary	
Chi-Square	6.5280
Chi-Square DF	6
Pr > Chi-Square	0.3667

Example 33.4 tests the sphericity of the covariance matrix of the same data set. The null hypothesis for the population covariance matrix is

$$\mathbf{\Sigma} = \sigma^2 \mathbf{I}$$

where σ^2 is an unknown positive constant and $\mathbf{I}$ is an identity matrix. You can use the following specification to test this hypothesis easily:

```
proc calis data=sales covpattern=sigsqi;
run;
```

Output 33.5.4 shows the initial specification of the covariance pattern. As required, the diagonal elements are all the same parameter named _varparm, and all the off-diagonal elements are fixed to zero. This is the same pattern as shown in Output 33.4.1 of Example 33.4.

Output 33.5.4 Initial Covariance Pattern for the Sphericity Test on the Sales Data

Initial MSTRUCT _COV_ Matrix				
	q1	**q2**	**q3**	**q4**
q1	. [_varparm]	0	0	0
q2	0	. [_varparm]	0	0
q3	0	0	. [_varparm]	0
q4	0	0	0	. [_varparm]

Output 33.5.5 shows the fit summary of the sphericity test. The chi-square is 27.747 ($df = 9$, $p = 0.0011$), which is statistically significant. You reject the sphericity hypothesis for the population covariance matrix.

Output 33.5.5 Fit Summary of the Sphericity Test on the Sales Data

Fit Summary	
Chi-Square	27.7470
Chi-Square DF	9
Pr > Chi-Square	0.0011

Again, the numerical results in Output 33.5.5 are different from those shown in Output 33.4.2 of Example 33.4. This is because with the COVPATTERN=SIGSQI option, the chi-square correction due to Box (1949) has been applied in the current example. To turn off the automatic chi-square correction, you can use the following specification:

```
proc calis data=sales covpattern=sigsqi chicorrect=0;
run;
```

As expected, the numerical results in Output 33.5.6 match exactly to those of in Output 33.4.2 of Example 33.4.

Output 33.5.6 Fit Summary of the Sphericity Test on the Sales Data: No Chi-Square Correction

Fit Summary	
Chi-Square	31.5951
Chi-Square DF	9
Pr > Chi-Square	0.0002

This example shows that for the tests of some standard covariance patterns, you can use the COVPATTERN=

option directly. As compared with the use of the explicit MSTRUCT model specifications, which are shown in Example 33.3 and Example 33.4, the use of COVPATTERN= option is more efficient and less error-prone in coding. In addition, it can apply chi-square corrections in appropriate situations.

PROC CALIS also provides the test of some "standard" mean patterns by the MEANPATTERN= option. You can use the COVPATTERN= and MEANPATTERN= options together to define the desired combinations of covariance and mean patterns. See these two options for details. See Example 33.22 for a multiple-group analysis with the simultaneous use of the COVPATTERN= and MEANPATTERN= options. Certainly, the COVPATTERN= and MEANPATTERN= options are limited to the standard covariance and mean patterns provided by PROC CALIS. When you need to fit some specific (nonstandard) covariance or mean patterns, the MSTRUCT model specification would be indispensable. See Example 33.19 and Example 33.22 for applications.

Example 33.6: Linear Regression Model

(View the complete code for this example (calex111.sas) in the example repository.)

This example shows how you can use PROC CALIS to fit the basic regression models. Unlike the preceding examples (Example 33.1, Example 33.2, Example 33.3, and Example 33.4) where you specify the covariance structures directly, in this example the covariance structures being analyzed are implied by the functional relationships specified in the model. The PATH modeling language introduced in the current example requires you to specify only the functional or path relationships among variables. PROC CALIS analyzes the implied covariance structures that are derived from the specified functional or path relationships.

Consider the same sales data as in Example 33.1. This example demonstrates a simple linear regression that uses q1 (the sales in the first quarter) to predict q4 (the sales in the fourth quarter).

In covariance structural analysis, or in general structural equation modeling, relationships among variables are usually represented by the so-called path diagram. For example, you can represent the linear regression of q4 on q1 by the following simple path diagram:

In the path diagram, q1 is an exogenous (or independent) variable and q4 is an endogenous (or dependent) variable. Formally, a variable in a path diagram is endogenous if there is at least one single-headed arrow pointing to it. Otherwise, the variable is exogenous. In some situations, researchers apply "causal" interpretations among variables in the path diagram, with the single-headed arrows indicating the causal directions. However, causal interpretations are not a requirement for using covariance structure analysis or structural equation modeling.

It is easy to transcribe the preceding path diagram into the PATH model specification in PROC CALIS, as shown in the following statements:

```
proc calis data=sales;
   path   q1  ===>  q4;
run;
```

Output 33.6.1 shows the modeling information of the linear regression model. It shows that all 14 observations are used and the model type is PATH. PROC CALIS analyzes the (implied) covariance structure model for the data. In the next table of Output 33.6.1, PROC CALIS shows the nature of the variables in the model: q4 is an endogenous manifest variable and q1 is an exogenous manifest variable. There is no latent variable in this simple path model.

Output 33.6.1 Modeling Information of the Linear Regression Model for the Sales Data

Simple Linear Regression Model by the PATH Modeling Language

The CALIS Procedure
Covariance Structure Analysis: Model and Initial Values

Modeling Information	
Maximum Likelihood Estimation	
Data Set	WORK.SALES
N Records Read	14
N Records Used	14
N Obs	14
Model Type	PATH
Analysis	Covariances

Variables in the Model		
Endogenous	**Manifest**	q4
	Latent	
Exogenous	**Manifest**	q1
	Latent	

Number of Endogenous Variables = 1
Number of Exogenous Variables = 1

Output 33.6.2 shows the initial model specification. The path is in the first table. A parameter name is attached to the path. The name _Parm1, which is generated automatically by PROC CALIS, denotes the effect parameter of q1 on q4. In the context of linear regression, _Parm1 also denotes the regression coefficient.

Output 33.6.2 Initial Specification of the Linear Regression Model for the Sales Data

Initial Estimates for PATH List

Path	Parameter	Estimate
q1 ===> q4	_Parm1	.

Initial Estimates for Variance Parameters

Variance Type	Variable	Parameter	Estimate
Exogenous	**q1**	_Add1	.
Error	**q4**	_Add2	.

NOTE: Parameters with prefix '_Add' are added by PROC CALIS.

Next, Output 33.6.2 shows the variance parameters in the model. You do not need to specify any of these parameters in the preceding PATH model specification—because PROC CALIS adds these parameters by

default. _Add1 denotes the variance parameter for the exogenous variable q1. _Add2 denotes the error variance parameter for the endogenous variable q4.

In the PATH model of PROC CALIS, all variances of exogenous variables and all error variances of endogenous variables are free parameters by default. In most practical applications, these parameters are usually free parameters in models and it would be laborious to specify them each time when you fit a covariance structure model. Therefore, to make the PATH model specification more efficient and easier, PROC CALIS sets these free parameters by default. In fact, with these default parameters in the PATH model, PROC CALIS produces essentially the same regression analysis results as those produced by common linear regression procedures such as PROC REG. This consistency is shown in the subsequent estimation results for the current example.

You can also explicitly specify those otherwise default parameters of the PATH model in PROC CALIS. Depending on the modeling situation, you can set any parameter in the PATH model as a free, fixed, or constrained parameter. You can also provide names for the parameters. Naming parameters is very useful for parameter referencing and for setting up parameter constraints. See Example 33.4. For details, see the PATH statement and the section "The PATH Model" on page 1763.

Output 33.6.3 shows some fit statistics from the linear regression model. The model fit chi-square is 0 with 0 degrees of freedom. This is a perfect model fit. The fit is perfect because the covariance model contains three distinct elements (variance of q1, variance of q4, and covariance between q1 and q4) that are fitted perfectly by three parameters: _Parm1 for the effect of q1 on q4, _Add1 for the variance of variable q1, and _Add2 for the error variance of variable q4. Thus, the unconstrained linear regression model estimates are simply a transformation of the covariance elements. Hence, the model is saturated with a perfect fit and zero degrees of freedom.

Output 33.6.3 Model Fit of the Linear Regression Model for the Sales Data

Fit Summary	
Chi-Square	0.0000
Chi-Square DF	0
Pr > Chi-Square	.
Standardized RMR (SRMR)	0.0000
RMSEA Estimate	.

Output 33.6.4 shows the estimates of the model. The effect of q1 on q4 is 0.6544 (standard error=0.7571). The associated t value is 0.86433, which is not significantly different from zero. The estimated variance of q1 is 0.3383 and the estimated error variance for q4 is 2.5207. Both estimates are significant.

Output 33.6.4 Parameter Estimates of the Linear Regression Model for the Sales Data

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q4	_Parm1	0.65436	0.75707	0.8643	0.3874

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	**q1**	_Add1	0.33830	0.13269	2.5495	0.0108
Error	**q4**	_Add2	2.52066	0.98869	2.5495	0.0108

For a simple linear regression such as this one, you could have used PROC REG. You get essentially the same estimates by specifying the following statements:

```
proc reg data=sales;
   model q4 = q1;
run;
```

Output 33.6.5 shows the parameter estimates from PROC REG. The intercept estimate is 2.7604 (standard error=1.1643) and the regression coefficient is 0.6544 (standard error=0.7880). The regression coefficient estimate matches PROC CALIS. However, the corresponding standard error estimate in PROC CALIS is 0.7571, which is slightly different from PROC REG. This difference is due to the different variance divisors that are used in calculating the standard error estimates. PROC CALIS uses $(N - 1)$ as the divisor (by default) while PROC REG uses $(N - q - 1)$, where N is the number of observations and q is the number of regression coefficients. In the current example, q is 1 so that the variance divisor in PROC REG is 1 less than the divisor in PROC CALIS. If you have at least a moderate sample size and the number of regression parameters is relatively small compared to the sample size, the discrepancy due to using different variance divisors is of little consequence.

Output 33.6.5 Parameter Estimates from PROC REG for the Sales Data

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	2.76040	1.16430	2.37	0.0353
q1	1	0.65436	0.78798	0.83	0.4225

By default, PROC CALIS analyzes only the covariance structures, which are properties of the second-order moments of the data. PROC CALIS does not automatically produce intercept estimates, which are properties of the first-order moments of the data.

In order to produce the intercept estimate in the linear regression context, you can add the MEANSTR (mean structures) option in the PROC CALIS statement, as shown in the following statements:

```
proc calis data=sales meanstr;
   path   q1   ===>   q4;
run;
```

Output 33.6.6 shows the parameter estimates of the model with the MEANSTR option added. Compared with Output 33.6.4, Output 33.6.6 produces one more table: estimates of the mean and intercept. The intercept estimate for q4 is 2.7604, which matches the intercept estimate from PROC REG. The estimated mean of q1 is 1.3671. All other estimates are the same for the analyses with and without the MEANSTR option.

Output 33.6.6 Parameter Estimates of the Linear Regression Model with the MEANSTR option for the Sales Data

PATH List					
Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q4	_Parm1	0.65436	0.75707	0.8643	0.3874

Output 33.6.6 *continued*

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	**q1**	_Add1	0.33830	0.13269	2.5495	0.0108
Error	**q4**	_Add2	2.52066	0.98869	2.5495	0.0108

Means and Intercepts						
Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	**q4**	_Add3	2.76040	1.12480	2.4541	0.0141
Mean	**q1**	_Add4	1.36714	0.16132	8.4749	<.0001

Linear regression estimates from PROC CALIS are comparable to those obtained from PROC REG, although the two procedures have different default treatments of the variance divisor in calculating the standard error estimates. With the MEANSTR option in the PROC CALIS statement, you can analyze the mean and covariance structures simultaneously. PROC CALIS prints the estimates of the intercepts and means when you model the mean structures.

This example shows how you can fit the linear regression model as a PATH model in PROC CALIS. You need to specify only path relationships among the variables in the PATH statement, because the implied covariance structures are generated and analyzed by PROC CALIS. To make model specification more efficient, PROC CALIS sets default variance parameters for exogenous variables and default error variance parameters for endogenous variables. You can also overwrite these default parameters by explicit specifications. See Example 33.7 for some sophisticated regression models that you can specify with PROC CALIS. See Example 33.17 for a more elaborate path model specification.

Example 33.7: Multivariate Regression Models

(View the complete code for this example (calex201.sas) in the example repository.)

This example shows how to analyze different types of multivariate regression models with PROC CALIS. Example 33.6 fits a simple linear regression model to the sales data that are described in Example 33.1. The simple linear regression model predicts the fourth quarter sales (q4) from the first quarter sales (q1). There is only one dependent (outcome) variable (q4) and one independent (predictor) variable (q1) in the analysis. Also, there are no constraints on the parameters. This example fits more sophisticated regression models. The models include more than one predictor. Some variables can serve as outcome variables and predictor variables at the same time. This example also illustrates the use of parameter constraints in model specifications and the use of the model fit statistics to search for a "best" model for the sales data.

Multiple Regression Model for the Sales Data

Consider a multiple regression model for q4. Instead of using just q1 as the predictor in the model as in Example 33.6, use all previous sales q1–q3 to predict the fourth-quarter sale (q4). The path model representation is shown in the following path diagram:

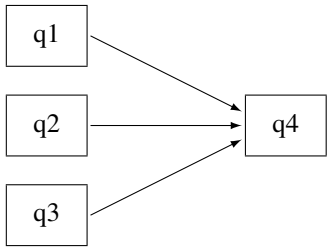

You can transcribe this path diagram into the following PATH model specification:

```
proc calis data=sales;
   path   q1 q2 q3 ===>  q4;
run;
```

In the path statement, the shorthand path specification

```
   path   q1 q2 q3 ===>  q4;
```

is equivalent to the following specification:

```
   path   q1  ===>  q4,
          q2  ===>  q4,
          q3  ===>  q4;
```

The shorthand notation provides a more convenient way to specify the path model. Some of the model fit statistics are shown in Output 33.7.1. This is a saturated model with perfect fit and zero degrees of freedom. Because the chi-square statistic is always smallest in a saturated model (with a zero chi-square value), it does not makes much sense to judge the model quality solely by looking at the chi-square value. However, a saturated model is useful for serving as a baseline model with which other nonsaturated competing models are compared.

Output 33.7.1 Model Fit of the Multiple Regression Model for the Sales Data

Fit Summary	
Chi-Square	0.0000
Chi-Square DF	0
Pr > Chi-Square	.
Standardized RMR (SRMR)	0.0000
RMSEA Estimate	.
Akaike Information Criterion	20.0000
Bozdogan CAIC	36.3906
Schwarz Bayesian Criterion	26.3906

In addition to the model fit chi-square statistic, Output 33.7.1 also shows Akaike's information criterion (AIC), Bozdogan's CAIC, and Schwarz's Bayesian criterion (SBC) of the saturated model. The AIC, CAIC, and SBC are derived from information theory and henceforth they are referred to as the information-theoretic fit indices. These information-theoretic fit indices measure the model quality by taking the model parsimony into account. The root mean square error of approximation (RMSEA) also takes the model parsimony into account, but it is not an information-theoretic fit index. The values of these information-theoretic fit indices

themselves do not indicate the quality of the model. However, when you fit several different models to the same data, you can order the models by these fit indices. The better the model, the smaller the fit index values. Unlike the chi-square statistic, these fit indices do not always favor a saturated model because a saturated model lacks model parsimony (the saturated model uses the most parameters to explain the data). The subsequent discussion uses these fit indices to select the "best" model for the sales data.

Output 33.7.2 shows the parameter estimates of the multiple regression model. In the first table, all path effect estimates are not statistically significant—that is, all t values are less than 1.96. The next table in Output 33.7.2 shows the variance estimates of q1–q3 and the error variance estimate for q4. All of these estimates are significant. The last table in Output 33.7.2 shows the covariances among the exogenous variables q1–q3. These covariance estimates are small and are not statistically significant.

Output 33.7.2 Parameter Estimates of the Multiple Regression Model for the Sales Data

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q4	_Parm1	0.55980	0.64938	0.8621	0.3887
q2 ===> q4	_Parm2	0.58946	0.84558	0.6971	0.4857
q3 ===> q4	_Parm3	0.88290	0.51635	1.7099	0.0873

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	**q1**	_Add1	0.33830	0.13269	2.5495	0.0108
	q2	_Add2	0.22466	0.08812	2.5495	0.0108
	q3	_Add3	0.60633	0.23782	2.5495	0.0108
Error	**q4**	_Add4	1.84128	0.72221	2.5495	0.0108

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q2	**q1**	_Add5	0.0001978	0.07646	0.00259	0.9979
q3	**q1**	_Add6	0.03610	0.12601	0.2865	0.7745
q3	**q2**	_Add7	0.12653	0.10821	1.1693	0.2423

In Output 33.7.2, the total number of parameter estimates is 10 (_Parm1–_Parm3 and _Add1–_Add7). Under the covariance structure model, these 10 parameters explain the 10 nonredundant elements in the covariance matrix for the sales data. That is why the model has a perfect fit with zero degrees of freedom.

In Output 33.7.2, notice that some parameters have the prefix '_Parm', while others have the prefix '_Add'. Both types of parameter names are generated by PROC CALIS. The parameters named with the '_Parm' prefix are those that were specified in the model, but were not named. In the current example, the parameters specified but not named are the path coefficients (effects) for the three paths in the PATH statement. The parameters named with the '_Add' prefix are default parameters added by PROC CALIS. In the current multiple regression example, the variances and covariances among the predictors (q1–q3) and the error variance for the outcome variable (q4) are default parameters in the model. In general, variances and covariances among exogenous variables and error variances of endogenous variables are default parameters in the PATH model. Avoid using parameter names with the '_Parm' and '_Add' prefixes to avoid confusion with parameters that are generated by PROC CALIS.

Direct and Indirect Effects Model for the Sales Data

In the multiple regression model, q1–q3 are all predictors that have direct effects on q4. This example considers the possibility of adding indirect effects into the multiple regression model. Because of the time ordering, it is reasonable to assume that there is a causal sequence q1 ===> q2 ===> q3. To implement this idea into the model, put two more paths into the preceding path diagram to form the following new path diagram:

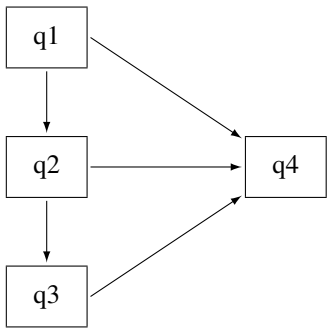

With the q1 ===> q2 and q2 ===> q3 paths, q2 and q3 are no longer exogenous in the model. They become endogenous. The only exogenous variable in the model is q1, which has a direct effect in addition to indirect effects on q4. The direct effect is indicated by the q1 ===> q4 path. The indirect effects are indicated by the following two causal chains: q1 ===> q2 ===> q4 and q1 ===> q2 ===> q3 ===> q4. Similarly, q2 has a direct and an indirect effect on q4. However, q3 has only a direct effect on q4. You can use the following statements to specify this *direct and indirect effects* model:

```
proc calis data=sales;
   path   q1       ===>  q2,
          q2       ===>  q3,
          q1 q2 q3 ===>  q4;
run;
```

Although the *direct and indirect effects* model has two more paths in the PATH statement than does the preceding multiple regression model, the current model is more precise because it has one fewer parameter. By introducing the causal paths q1 ===> q2 and q2===> q3, the six variances and covariances among q1–q3 are explained by: the two causal effects, the exogenous variance of q1, and the error variances for q2 and q3 (that is, five parameters in the model). Hence, the current *direct and indirect effects* model has one fewer parameter than the preceding multiple regression model.

Output 33.7.3 shows some model fit indices of the direct and indirect effects model. The model fit chi-square is 0.0934 with one degree of freedom. It is not significant. Therefore, you cannot reject the model on statistical grounds. The standardized root mean squares of residuals (SRMR) is 0.028 and the root mean square error of approximation (RMSEA) is close to zero. Both indices point to a very good model fit. The AIC, CAIC, and SBC are all smaller than those of the saturated model, as shown in Output 33.7.1. This suggests that the *direct and indirect effects* model is better than the saturated model.

Output 33.7.3 Model Fit of the Direct and Indirect Effects Model for the Sales Data

Fit Summary	
Chi-Square	0.0934
Chi-Square DF	1
Pr > Chi-Square	0.7600
Standardized RMR (SRMR)	0.0280
RMSEA Estimate	0.0000
Akaike Information Criterion	18.0934
Bozdogan CAIC	32.8449
Schwarz Bayesian Criterion	23.8449

Output 33.7.4 shows the parameter estimates of the *direct and indirect effects* model. All the path effects are not significant, while all the variance or error variance estimates are significant. Unlike the saturated model where you have covariance estimates among several exogenous variables (as shown in Output 33.7.2), in the *direct and indirect effects* model there is only one exogenous variable (q1) and hence there is no covariance estimate in the results.

Output 33.7.4 Parameter Estimates of the Direct and Indirect Effects Model for the Sales Data

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q2	_Parm1	0.0005847	0.22602	0.00259	0.9979
q2 ===> q3	_Parm2	0.56323	0.42803	1.3159	0.1882
q1 ===> q4	_Parm3	0.55980	0.64705	0.8652	0.3870
q2 ===> q4	_Parm4	0.58946	0.84524	0.6974	0.4856
q3 ===> q4	_Parm5	0.88290	0.51450	1.7160	0.0862

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	q1	_Add1	0.33830	0.13269	2.5495	0.0108
Error	q2	_Add2	0.22466	0.08812	2.5495	0.0108
	q3	_Add3	0.53506	0.20987	2.5495	0.0108
	q4	_Add4	1.84128	0.72221	2.5495	0.0108

Although the current *direct and indirect effects* model is better than the saturated model and both the SRMR and RMSEA indicate a good model fit, the nonsignificant path effect estimates are unsettling. You continue to explore alternative models for the data.

Indirect Effects Model for the Sales Data

The saturated model includes only the direct effects of q1–q3 on q4, while the *direct and indirect effects* model includes both the direct and indirect effects of q1 and q2 on q4. An alternative model with only the indirect effects of q1 and q2 on q4, but without their direct effects, is possible. Such an *indirect effects* model is represented by the following path diagram:

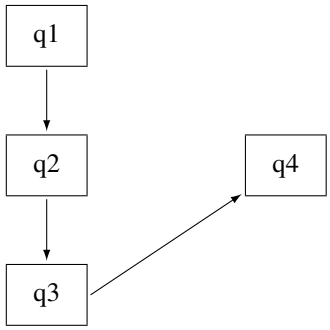

You can easily transcribe this path diagram into the following PATH model specification:

```
proc calis data=sales;
   path   q1   ===>  q2,
          q2   ===>  q3,
          q3   ===>  q4;
run;
```

Output 33.7.5 shows some model fit indices for the *indirect effects* model. The chi-square model fit statistic is not statistically significant, so the model is not rejected. The standardized RMR is 0.0905, which is a bit higher than the conventional value of 0.05 for an acceptable good model fit. However, the RMSEA is close to zero, which shows a very good model fit. The AIC, CAIC and SBC are all smaller than the *direct and indirect effects* model. These information-theoretic fit indices suggest that the *indirect effects* model is better.

Output 33.7.5 Model Fit of the Indirect Effects Model for the Sales Data

Fit Summary	
Chi-Square	1.2374
Chi-Square DF	3
Pr > Chi-Square	0.7440
Standardized RMR (SRMR)	0.0905
RMSEA Estimate	0.0000
Akaike Information Criterion	15.2374
Bozdogan CAIC	26.7108
Schwarz Bayesian Criterion	19.7108

Output 33.7.6 shows the parameter estimates of the *indirect effects* model. All the variance and error variance estimates are statistically significant. However, only the path effect of q3 on q4 is statistically significant, and all other path effects are not. Having significant variances with nonsignificant paths raises some concerns about accepting the current model even though the AIC, CAIC, and SBC values suggest that it is the best model so far.

Output 33.7.6 Parameter Estimates of the Indirect Effects Model for the Sales Data

PATH List						
Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1	===>	q2	_Parm1	0.0005847	0.22602	0.00259	0.9979
q2	===>	q3	_Parm2	0.56323	0.42803	1.3159	0.1882
q3	===>	q4	_Parm3	1.03924	0.50506	2.0577	0.0396

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	q1	_Add1	0.33830	0.13269	2.5495	0.0108
Error	q2	_Add2	0.22466	0.08812	2.5495	0.0108
	q3	_Add3	0.53506	0.20987	2.5495	0.0108
	q4	_Add4	2.01067	0.78865	2.5495	0.0108

Constrained Indirect Effects Model for the Sales Data

In the preceding *indirect effects* model, some path effects are not significant. In the current model, all the path effects are constrained to be equal. The following path diagram represents the *constrained indirect effects* model:

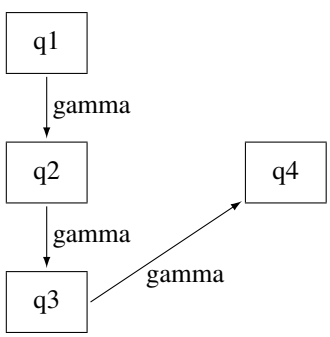

Except for one notable difference, this path diagram is the same as the path diagram for the preceding *indirect effects* model. The current path diagram labels all the paths with the same name (gamma) to signify that they are the same parameter. You can specify this *constrained indirect effects* model with this chosen constraint on the path effects by the using following statements:

```
proc calis data=sales;
   path   q1   ===>  q2      = gamma,
          q2   ===>  q3      = gamma,
          q3   ===>  q4      = gamma;
run;
```

In the PATH statement, append an equal sign and a parameter name gamma in each of the path entries. This specification means that all the associated path effects are the same parameter named gamma.

Output 33.7.7 shows some fit indices for the *constrained indirect effects* model. Again, the model fit chi-square statistic is not significant. However, the SRMR is 0.2115, which is too large to accept as a good model. The RMSEA is 0.0499, which still indicates a good model fit. The AIC, CAIC, and SBC values

are a bit smaller than those of the preceding unconstrained *indirect effects* model. Therefore, it seems that constraining the path effects leads to a slightly better model.

Output 33.7.7 Model Fit of the Constrained Indirect Effects Model for the Sales Data

Fit Summary	
Chi-Square	5.1619
Chi-Square DF	5
Pr > Chi-Square	0.3964
Standardized RMR (SRMR)	0.2115
RMSEA Estimate	0.0499
Akaike Information Criterion	15.1619
Bozdogan CAIC	23.3572
Schwarz Bayesian Criterion	18.3572

Output 33.7.8 shows the parameter estimates of the *constrained indirect effects* model. Again, all variance and error variance estimates are significant, and all path effects are not significant. The effect estimate is 0.24 (standard error=0.19, $t = 1.25$).

Output 33.7.8 Parameter Estimates of the Constrained Indirect Effects Model for the Sales Data

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q2	gamma	0.24014	0.19152	1.2539	0.2099
q2 ===> q3	gamma	0.24014	0.19152	1.2539	0.2099
q3 ===> q4	gamma	0.24014	0.19152	1.2539	0.2099

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	**q1**	_Add1	0.33830	0.13269	2.5495	0.0108
Error	**q2**	_Add2	0.24407	0.09573	2.5495	0.0108
	q3	_Add3	0.55851	0.21907	2.5495	0.0108
	q4	_Add4	2.39783	0.94051	2.5495	0.0108

Constrained Indirect Effects and Error Variances Model for the Sales Data

In addition to constraining all the path effects in the preceding model, the current model constrains all the error variances. Before using a path diagram to represent the current constrained indirect effects and constrained error variances, it is important to realize that you have not manually defined variances and covariances in the path diagrams for all of the preceding models. The default parameterization in PROC CALIS defined those parameters.

Represent the variances and covariances in a path diagram with double-headed arrows. When a double-headed arrow points to a single variable, it represents the variance parameter. When a double-headed arrow points to two distinct variables, it represents the covariance between the two variables. Consider the unconstrained *indirect effects* model for the sales data as an example. A more complete path diagram representation is as follows:

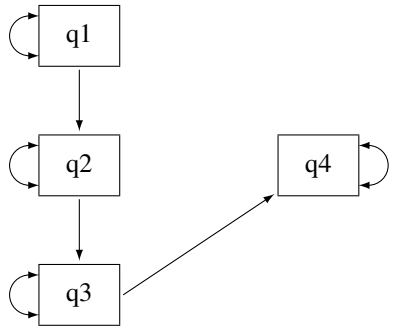

In this path diagram, a double-headed arrow on each variable represents variance or error variance. For q1, the double-headed arrow represents the variance parameter of q1. For other variables, the double-headed arrows represent error variances because those variables are endogenous (that is, they are predicted from other variables) in the model.

In order to represent the equality-constrained parameters in the model, you can put parameter names in the respective parameter locations in the path diagram. For the current *constrained indirect effects and error variances* model, you can represent the model by the following path diagram:

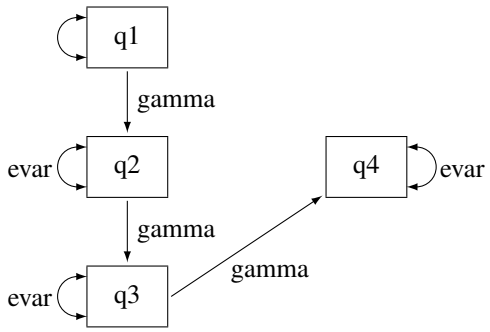

In the path diagram, label all the path effects by the parameter gamma and all error variances by the parameter evar. The double-headed arrow attached to q1 is not labeled by any name. This means that it is an unnamed free parameter in the model.

You can transcribe the path diagram into the following statements:

```
proc calis data=sales;
   path   q1   ===>  q2      = gamma,
          q2   ===>  q3      = gamma,
          q3   ===>  q4      = gamma;
   pvar   q2 q3 q4 = 3 * evar;
run;
```

The specification in the PATH statement is the same as the preceding PATH model specification for the *constrained indirect effects* model. The new specification here is the PVAR statement. You use the PVAR statement to specify partial variances, which include the (total) variances of exogenous variables and the error variances of the endogenous variables. In the PVAR statement, you specify the variables for which you intend to define variances. If you do not specify anything after the list of variables, the variances of these variables

are unnamed free parameters. If you put an equal sign after the variable lists, you can specify parameter names, initial values, or fixed parameters for the variances of the variables. See the PVAR statement for details. In the current model, **3*evar** means that you want to specify evar three times (for the error variance parameters of q2, q3, and q4).

Note that you did not specify the variance of q1 in the PVAR statement. This variance is a default parameter in the model, and therefore you do not need to specify it in the PVAR statement. Alternatively, you can specify it explicitly in the PVAR statement by giving it a parameter name. For example, you can specify the following:

```
pvar   q2 q3 q4 = 3 * evar,
       q1       = MyOwnName;
```

Or, you can specify it explicitly without giving it a parameter name, as shown in following statement:

```
pvar   q2 q3 q4 = 3 * evar,
       q1 ;
```

All these specifications lead to the same estimation results. The difference between the two specifications is the explicit parameter name for the variance of q1. Without putting q1 in the PVAR statement, the variance parameter is named with the prefix _Add, which is generated as a default parameter by PROC CALIS. With the explicit specification of q1, the variance parameter is named MyOwnName. With the explicit specification of q1, but without giving it a parameter name in the PVAR statement, the variance parameter is named with the prefix _Parm, which PROC CALIS generates for unnamed free parameters.

Output 33.7.9 shows some fit indices for the *constrained indirect effects and error variances* model. The model fit chi-square is 19.7843, which is significant at the 0.05 α-level. In practice, the model fit chi-square statistic is not the only criterion for judging model fit. In fact, it might not even be the most commonly used criterion for measuring model fit. Other criteria such as the SRMR and RMSEA are more popular or important. Unfortunately, the values of these two fit indices do not support the current constrained model either. The SRMR is 1.5037 and the RMSEA is 0.3748. Both are much greater than the commonly accepted 0.05 criterion.

Output 33.7.9 Model Fit of the Constrained Indirect Effects and Error Variances Model for the Sales Data

Fit Summary	
Chi-Square	19.7843
Chi-Square DF	7
Pr > Chi-Square	0.0061
Standardized RMR (SRMR)	1.5037
RMSEA Estimate	0.3748
Akaike Information Criterion	25.7843
Bozdogan CAIC	30.7015
Schwarz Bayesian Criterion	27.7015

The AIC, CAIC, and SBC values are all much greater than those of the preceding *constrained indirect effects* model. Therefore, constraining the error variances in addition to the constrained indirect effects does not lead to a better model.

Output 33.7.10 shows the parameter estimates of the *constrained indirect effects and error variances* model. All estimates are significant in the model, which is often desirable. However, because of the bad model fit, this model is not acceptable.

Output 33.7.10 Parameter Estimates of the Constrained Indirect Effects and Error Variances Model for the Sales Data

PATH List					
Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q2	gamma	0.64733	0.16128	4.0137	<.0001
q2 ===> q3	gamma	0.64733	0.16128	4.0137	<.0001
q3 ===> q4	gamma	0.64733	0.16128	4.0137	<.0001

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**q2**	evar	1.00220	0.22695	4.4159	<.0001
	q3	evar	1.00220	0.22695	4.4159	<.0001
	q4	evar	1.00220	0.22695	4.4159	<.0001
Exogenous	**q1**	_Add1	0.33830	0.13269	2.5495	0.0108

Partially Constrained Model for the Sales Data

In the preceding model, constraining all error variances to be same shows that the model fit is unacceptable, even though all parameter estimates are significant. Relaxing those constraints a little might improve the model. The following path diagram represents such a *partially constrained* model:

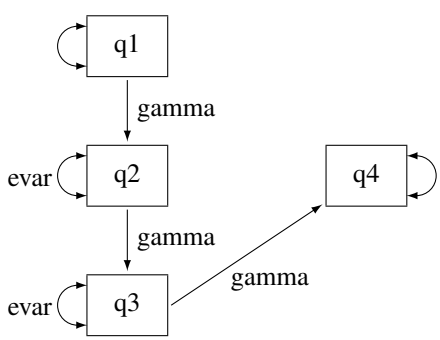

The only difference between the current *partially constrained* model and the preceding *constrained indirect effects and error variances* model is that the error variance for q4 is no longer constrained to be equal to the error variances of q2 and q3. In the path diagram, evar is no longer attached to the double-headed arrow that is associated with the error variance of q4. You can transcribe this path diagram representation into the following PATH model specification:

```
proc calis data=sales;
   path   q1   ===>  q2       = gamma,
          q2   ===>  q3       = gamma,
          q3   ===>  q4       = gamma;
   pvar   q2 q3 = 2 * evar,
          q4 q1;
run;
```

Now, the PVAR statement has only the error variances of q2 and q3 constrained to be equal. The error variance of q4 and the variance of q1 are free parameters without constraints.

Output 33.7.11 shows some fit indices for the *partially constrained* model. The chi-square model fit test statistic is not significant. The SRMR is 0.3877 and the RMSEA is 0.1164. These are far from the conventional acceptance level of 0.05. However, the AIC, CAIC, and SBC values are all slightly smaller than the *constrained indirect effects* model, as shown in Output 33.7.7. In fact, these information-theoretic fit indices suggest that the *partially constrained* model is the best model among all models that have been considered.

Output 33.7.11 Model Fit of the Partially Constrained Model for the Sales Data

Fit Summary	
Chi-Square	7.0575
Chi-Square DF	6
Pr > Chi-Square	0.3156
Standardized RMR (SRMR)	0.3877
RMSEA Estimate	0.1164
Akaike Information Criterion	15.0575
Bozdogan CAIC	21.6138
Schwarz Bayesian Criterion	17.6138

Output 33.7.12 shows the parameter estimates of the *partially constrained* model. Again, all variance and error variance parameters are statistically significant. However, the path effects are only marginally significant.

Output 33.7.12 Parameter Estimates of the Partially Constrained Model for the Sales Data

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q2	gamma	0.35546	0.18958	1.8750	0.0608
q2 ===> q3	gamma	0.35546	0.18958	1.8750	0.0608
q3 ===> q4	gamma	0.35546	0.18958	1.8750	0.0608

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**q2**	evar	0.40601	0.11261	3.6056	0.0003
	q3	evar	0.40601	0.11261	3.6056	0.0003
	q4	_Parm1	2.29415	0.89984	2.5495	0.0108
Exogenous	**q1**	_Parm2	0.33830	0.13269	2.5495	0.0108

Which Model Should You Choose?

You fit various models in this example for the sales data. The fit summary of the models is shown in the following table:

	1	2	3	4	5	6
	Saturated	**Direct and Indirect Effects**	**Indirect Effects**	**Constrained Indirect Effects**	**Constrained Indirect Effects and Error Variances**	**Partially Constrained**
df	0	1	3	5	7	6
p-value	.	0.76	0.74	0.40	0.01	0.32
SRMR	0	0.03	0.09	0.21	1.50	0.39
RMSEA	.	0.00	0.00	0.05	0.37	0.12
AIC	20.00	18.09	15.24	15.16	25.78	15.06
CAIC	36.39	32.84	26.71	23.36	30.70	21.61
SBC	26.39	23.84	19.71	18.36	27.70	17.61

As discussed previously, the model fit chi-square test statistic always favors models with a lot of parameters. It does not take model parsimony into account. In particular, a saturated model (Model 1) always has a perfect fit. However, it does not explain the data in a concise way. Therefore, the model fit chi-square statistic is not used here for comparing the competing models.

The standardized root mean square residual (SRMR) also does not take the model parsimony into account. It tells you how the fitted covariance matrix is different from the observed covariance matrix in a certain standardized way. Again, it always favors models with a lot of parameters. As shown in the preceding table, the more parameters (the fewer degrees of freedom) the model has, the smaller the SRMR is. A conventional criterion is to accept a model with SRMR less than 0.05. Applying this criterion, only the saturated model (Model 1) and the *direct and indirect effects* (Model 2) models are acceptable. The *indirect effects* model (Model 3) is marginally acceptable.

The root mean square error of approximation (RMSEA) fit index does take model parsimony into account. With the 'RMSEA less than 0.05 criterion', the *constrained indirect effects and error variances* model (Model 5) and the *partially constrained* model (Model 6) are not acceptable.

The information-theoretic fit indices such as the AIC, CAIC, and SBC also take model parsimony into account. All of these indices point to the *partially constrained* model (Model 6) as the best model among the competing models. However, because this model has a relatively bad absolute fit, as indicated by the large SRMR value (0.39), accepting this model is questionable. In addition, the information-theoretic fit indices of the *indirect effects* model (Model 3) and of the *constrained indirect effects* model (Model 4) are not too different from those of the *partially constrained* model (Model 6). The indirect effects model is especially promising because it has relatively small SRMR and RMSEA values. The drawback is that some path effect estimates in the indirect effects model are not significant. Perhaps collecting and analyzing more data might confirm these promising models with significant path effects.

You might not be able to draw a unanimous conclusion about the best model for the sales data of this example. Different fit indices in structural equation modeling do not always point to the same conclusions. The analyses in the current example show some of the complexity of structural equation modeling. Some interesting questions about model selections are:

- Do you choose a model based on a single fit criterion? Or, do you consider a set of model fit criteria to weigh competing models?
- Which fit index criterion is the most important for judging model fit?
- In selecting your "best" model, how do you take "chance" into account?

- How would you use your substantive theory to guide your model search?

The answers to these interesting research questions might depend on the context. Nonetheless, PROC CALIS can help you in the model selection process by computing various kinds of fit indices. (Only a few of these fit indices are shown in the output of this example. See the FITINDEX statement for a wide variety of fit indices that you can obtain from PROC CALIS.)

Alternative PATH Model Specifications for Variances and Covariances

The PATH modeling language of PROC CALIS is designed to map the path diagram representation into the PATH statement syntax efficiently. For any path that is denoted by a single-headed arrow in the path diagram, you can specify a path entry in the PATH statement. You can also specify double-headed arrows in the PATH statement.

Consider the preceding path diagram for the *partially constrained* model for the sales data. You use double-headed arrows to denote variances or error variances of the variables. The path diagram is shown in the following:

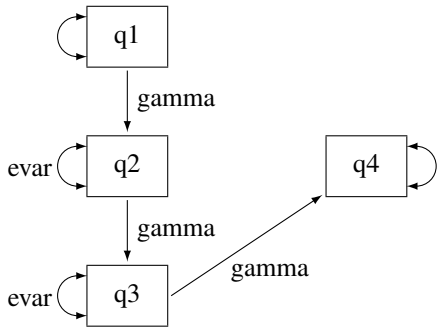

As discussed previously, you can use the PVAR statement to specify these variances or error variances as in following syntax:

```
pvar   q2 q3 = 2 * evar,
       q4 q1;
```

Alternatively, you can specify these double-headed arrows directly as paths in the PATH statement, as shown in the following statements:

```
proc calis data=sales;
   path   q1   ===>  q2      = gamma,
          q2   ===>  q3      = gamma,
          q3   ===>  q4      = gamma,
          <==>    q2 q3      = 2 * evar,
          <==>    q4 q1;
run;
```

To specify the double-headed paths pointing to individual variables, you begin with the double-headed arrow notation `<==>`, followed by the list of variables. For example, in the preceding specification, the error variance of q4 and the variance of q1 are specified in the last path entry of the PATH statement. If you want to define the parameter names for the variances, you can add a parameter list after an equal sign in the path entries.

For example, the error variances of q2 and q3 are denoted by the free parameter evar in a path entry in the PATH statement.

Alternatively, you can specify the double-headed arrow paths literally in a PATH statement, as shown in the following equivalent specification:

```
proc calis data=sales;
   path   q1   ===>  q2      = gamma,
          q2   ===>  q3      = gamma,
          q3   ===>  q4      = gamma,
          q2   <==>  q2      = evar,
          q3   <==>  q3      = evar,
          q4   <==>  q4,
          q1   <==>  q1;
run;
```

For example, the path entry `q1 <==> q1` specifies the variance of q1. It is an unnamed free parameter in the model.

Output 33.7.13 show the parameter estimates for this alternative specification method. All these estimates match exactly those with the PVAR statement specification, as shown in Output 33.7.12. The only difference is that all estimation results are now presented under one PATH List, as shown in Output 33.7.13, instead of as two tables as shown in Output 33.7.12.

Output 33.7.13 Path Estimates of the Partially Constrained Model for the Sales Data

PATH List					
Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q2	gamma	0.35546	0.18958	1.8750	0.0608
q2 ===> q3	gamma	0.35546	0.18958	1.8750	0.0608
q3 ===> q4	gamma	0.35546	0.18958	1.8750	0.0608
q2 <==> q2	evar	0.40601	0.11261	3.6056	0.0003
q3 <==> q3	evar	0.40601	0.11261	3.6056	0.0003
q4 <==> q4	_Parm1	2.29415	0.89984	2.5495	0.0108
q1 <==> q1	_Parm2	0.33830	0.13269	2.5495	0.0108

The double-headed arrow path syntax applies to covariance specification as well. For example, the following PATH statement specifies the covariances among variables x1–x3:

```
path    x2 <==> x1,
        x3 <==> x1,
        x3 <==> x2;
```

In the beginning of the current example, you use the following path diagram to represent the multiple regression model for the sales data:

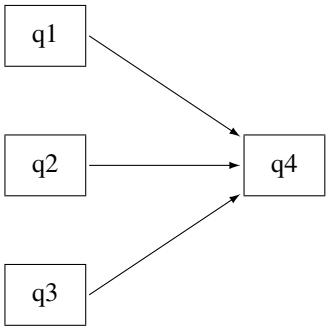

The following statements specify the multiple regression model:

```
proc calis data=sales;
   path   q1 q2 q3 ===>  q4;
run;
```

You do not represent the covariances and variances among the exogenous variables explicitly in the path diagram, nor in the PATH statement specification. However, PROC CALIS generates them as free parameters by default. Some researchers might prefer to represent the exogenous variances and covariances explicitly in the path diagram, as shown in the following path diagram:

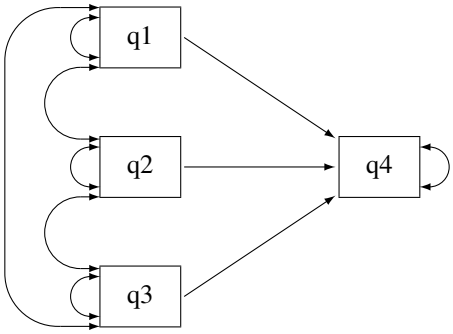

In the path diagram, there are three single-head arrows and seven double-headed arrows. These 10 paths represent the 10 parameters in the covariance structure model. To represent all these parameters in the PATH model specification, you can use the following statements:

```
proc calis data=sales;
   path   q1   ===>  q4  ,
          q2   ===>  q4  ,
          q3   ===>  q4  ,
          q1   <==>  q1  ,
          q2   <==>  q2  ,
          q3   <==>  q3  ,
          q1   <==>  q2  ,
          q2   <==>  q3  ,
          q1   <==>  q3  ,
          q4   <==>  q4  ;
run;
```

The first three path entries in the PATH statement reflect the single-headed paths in the path diagram. The next six path entries in the PATH statement reflect the double-headed paths among the exogenous variables q1–q3 in the path diagram. The last path entry in the PATH statement reflects the double-headed path attached to the endogenous variable q4 in the path diagram. With this specification, the parameter estimates for the multiple regression model are all shown in Output 33.7.14.

Output 33.7.14 Path Estimates of the Multiple Regression Model for the Sales Data

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
q1 ===> q4	_Parm01	0.55980	0.64938	0.8621	0.3887
q2 ===> q4	_Parm02	0.58946	0.84558	0.6971	0.4857
q3 ===> q4	_Parm03	0.88290	0.51635	1.7099	0.0873
q1 <==> q1	_Parm04	0.33830	0.13269	2.5495	0.0108
q2 <==> q2	_Parm05	0.22466	0.08812	2.5495	0.0108
q3 <==> q3	_Parm06	0.60633	0.23782	2.5495	0.0108
q1 <==> q2	_Parm07	0.0001978	0.07646	0.00259	0.9979
q2 <==> q3	_Parm08	0.12653	0.10821	1.1693	0.2423
q1 <==> q3	_Parm09	0.03610	0.12601	0.2865	0.7745
q4 <==> q4	_Parm10	1.84128	0.72221	2.5495	0.0108

These estimates are the same as those in Output 33.7.2, where the estimates are shown in three different tables, instead of in one table for all paths as in Output 33.7.14.

Sometimes, specification of some single-headed and double-headed paths can become very laborious. Fortunately, PROC CALIS provides shorthand notation for the PATH statement to make the specification more efficient. For example, a more concise way to specify the preceding multiple regression model is shown in the following statements:

```
proc calis data=sales;
   path   q1 q2 q3  ===>  q4  ,
          <==>  [q1-q3]  ,
          <==>  q4  ;
run;
```

The first path entry `q1 q2 q3 ===> q4` in the PATH statement represents the three single-headed arrows in the path diagram. The second path entry `<==> [q1-q3]` generates the variances and covariances for the set of variables specified in the rectangular brackets. The last path entry represents the error variance of q4. Consequently, expanding the preceding shorthand specification generates the following specification:

```
proc calis data=sales;
   path   q1   ===>  q4  ,
          q2   ===>  q4  ,
          q3   ===>  q4  ,
          q1   <==>  q1  ,
          q2   <==>  q1  ,
          q2   <==>  q2  ,
          q3   <==>  q1  ,
          q3   <==>  q2  ,
          q3   <==>  q3  ,
          q4   <==>  q4  ;
```

```
run;
```

Notice that the third through ninth path entries correspond to the lower triangular elements of the covariance matrix for q1–q3.

CAUTION: The double-headed path specification does *not* represent a reciprocal relationship. That is, the following statement specifies the covariance between x2 and x1:

```
path    x2 <==> x1,
```

But the following statement specifies that x2 and x1 have reciprocal causal effects:

```
path    x2 <=== x1,
        x1 ===> x2;
```

The reciprocal causal effects specification reflects the following path diagram:

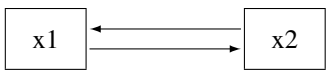

Example 33.8: Measurement Error Models

(View the complete code for this example (calex202.sas) in the example repository.)

In this example, you use PROC CALIS to fit some measurement error models. You use latent variables to define "true" scores variables that are measured without errors. You constrain parameters by using parameter names or fixed values in the PATH model specification.

Consider a simple linear regression model with dependent variable y and predictor variable x. The path diagram for this simple linear regression model is depicted as follows:

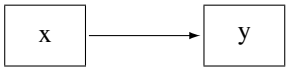

Suppose you have the following SAS data set for the regression analysis of y on x:

```
data measures;
   input x y @@;
   datalines;
 7.91736    13.8673    6.10807    11.7966    6.94139    12.2174
 7.61290    12.9761    6.77190    11.6356    6.33328    11.7732
 7.60608    12.8040    6.65642    12.8866    6.26643    11.9382
 7.32266    13.2590    5.76977    10.7654    5.62881    11.5041
 7.57418    13.2502    7.17305    13.3416    8.23123    13.9876
 7.17199    13.1750    8.04604    14.5968    5.77692    11.5077
 5.72741    11.3299    6.66033    12.5159    7.14944    12.4988
 7.51832    12.3588    5.48877    11.2211    7.50323    13.3735
 7.15814    13.1556    7.35485    13.8457    8.91648    14.4929
 5.37445     9.6366    6.00419    11.7654    6.89546    13.1493
```

```
;
```

This data set contains 30 observations for the x and y variables. You can fit the simple linear regression model to the measures data by the PATH model specification of PROC CALIS, as shown in the following statements:

```
proc calis data=measures;
   path
      x ===> y;
run;
```

Output 33.8.1 shows that the regression coefficient estimate (denoted as _Parm1 in the PATH List) is 1.1511 (standard error = 0.1002).

Output 33.8.1 Estimates of the Linear Regression Model for the Measures Data

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x ===> y	_Parm1	1.15112	0.10016	11.4924	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	x	_Add1	0.79962	0.20999	3.8079	0.0001
Error	y	_Add2	0.23265	0.06110	3.8079	0.0001

You can also do the simple linear regression by PROC REG by the following statement:

```
proc reg data=measures;
   model y = x;
run;
```

Output 33.8.2 shows that PROC REG essentially gives the same regression coefficient estimate with a similar standard error estimate. The discrepancy in the standard error estimates produced by the two procedures is due to the different variance divisors in computing standard errors in the two procedures. But the discrepancy is negligible when the sample size becomes large.

Output 33.8.2 PROC REG Estimates of the Linear Regression Model for the Measures Data

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	4.62455	0.70790	6.53	<.0001
x	1	1.15112	0.10194	11.29	<.0001

There are two main differences between PROC CALIS and PROC REG regarding the parameter estimation results. First, PROC CALIS does not give the estimate of the intercept because by default PROC CALIS analyzes only the covariance structures. Therefore, it does not estimate the intercept. To obtain the intercept estimate, you can add the MEANSTR option in the PROC CALIS statement, as is shown in Example 33.9. Second, in Output 33.8.1 of PROC CALIS, the variance estimate of x and the error variance estimate of y are shown. The corresponding results are not shown as parameter estimates in the PROC REG results. In PROC

CALIS, these two variances are model parameters in covariance structure analysis. PROC CALIS adds these variances as default parameters. You can also represent these two variance parameters by double-headed arrows in the path diagram, as shown in the following:

The two double headed-arrows attached to x and y represent the variances. Although it is not necessary to specify these default parameters, you can use the PVAR statement to specify them explicitly, as shown in the following statements:

```
proc calis data=measures meanstr;
   path
      x ===> y;
   pvar
      x y;
run;
```

In the PROC CALIS statement, you specify the MEANSTR option to request the analysis of mean structures together with covariance structures. Output 33.8.3 shows the estimation results.

Output 33.8.3 Estimates of the Measurement Error Model with Error in x

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x ===> y	_Parm1	1.15112	0.10016	11.4924	<.0001

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Exogenous	x	_Parm2	0.79962	0.20999	3.8079	0.0001
Error	y	_Parm3	0.23265	0.06110	3.8079	0.0001

Means and Intercepts

Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	y	_Add1	4.62455	0.69578	6.6466	<.0001
Mean	x	_Add2	6.88865	0.16605	41.4850	<.0001

The regression coefficient estimate and the variance estimates are the same as those in Output 33.8.1. However, in Output 33.8.3, there is an additional table for the mean and intercept estimates. The intercept estimate for y is 4.6246 (standard error=0.6958), which match closely to the results obtained from PROC REG, as shown in Output 33.8.2.

Measurement Error in x

PROC CALIS can also handle more complicated regression situations where the variables are measured with errors. This is beyond the application of PROC REG.

Suppose that the predictor variable x is measured with error and from prior studies you know that the size of the measurement error variance is about 0.019. You can use PROC CALIS to incorporate this information into the model. First, think of the measured variable x as composed of two components: one component is the "true" score measure Fx and the other is the measurement error e1. Both of these components are not observed (that is, latent) but they sum up to yield x. That is,

$$x = Fx + e1$$

Because x is contaminated with measurement error, what you are interested in knowing is the regression effect of the true score Fx on x. The following path diagram represents this regression scenario:

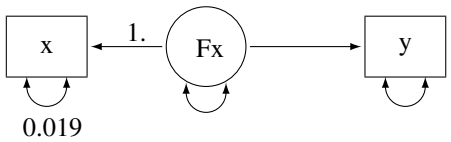

In path diagrams, latent variables are usually represented by circles or ovals, while observed variables are represented by rectangles. In the current path diagram, Fx is a latent variable and is represented by a circle. The other two variables are observed variables and are represented by rectangles. There are five arrows in the path diagram. Two of them are single-headed arrows that represent functional relationships, while the other three are double-headed arrows that represent variances or error variances. Two paths are labeled with fixed values. The path effect from Fx to x is fixed at 1, as assumed in the measurement error model. The error variance for measuring x is fixed at 0.019 due to the prior knowledge about the measurement error variance. The remaining three arrows represent free parameters in the model: the regression coefficient of y on Fx, the variance of Fx, and the error variance of y. The following statements specify the model for this path diagram:

```
proc calis data=measures;
   path
      x  <=== Fx   = 1.,
      Fx ===> y;
   pvar
      x  = 0.019,
      Fx, y;
run;
```

You specify all the single-headed paths in the PATH statement and all the double-headed arrows in the PVAR statement. For paths with fixed values, you put the equality at the back of the specifications to tell PROC CALIS about the fixed values. For example, the path coefficient in the path **x <=== Fx** is fixed at 1 and the error variance for x is fixed at 0.019. All other specifications represent unnamed free parameters in the model.

Output 33.8.4 shows the estimation results. The effect of Fx on y is 1.1791 (standard error=0.1029). This effect is slightly greater than the corresponding effect (1.1511) of x on y in the preceding model where the measurement error of x has not been taken into account, as shown in Output 33.8.3.

Output 33.8.4 Estimates of the Measurement Error Model with Error in x

PATH List						
Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
x	<===	Fx		1.00000			
Fx	===>	y	_Parm1	1.17914	0.10288	11.4615	<.0001

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	x		0.01900			
Exogenous	Fx	_Parm2	0.78062	0.20999	3.7174	0.0002
Error	y	_Parm3	0.20686	0.06126	3.3767	0.0007

Measurement Errors in x and y

Measurement error can occur in the y variable too. Suppose that both x and y are measured with errors. From prior studies, the measurement error variance of x is known to be 0.019 (as in the preceding modeling scenario) and the measurement error variance of y is known to be 0.022. The following path diagram represents the current modeling scenario:

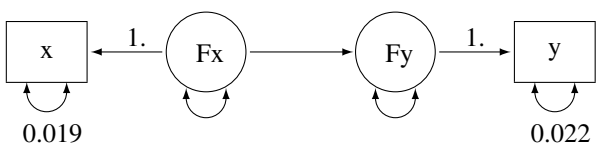

In the current path diagram the true score variable Fy and its measurement indicator y have the same kind of relationship as the relationship between the true score variable Fx and its measurement indicator x in the previous description. The error variance for measuring y is treated as a known constant 0.022. You can transcribe this path diagram easily to the following PROC CALIS specification:

```
proc calis data=measures;
   path
      x  <=== Fx  = 1.,
      Fx ===> Fy   ,
      Fy ===> y   = 1.;
   pvar
      x  = 0.019,
      y  = 0.022,
      Fx Fy;
run;
```

Again, you specify all the single-headed paths in the PATH statement and the double-headed paths in the PVAR statement. You provide the fixed parameter values by appending the required equalities after the individual specifications.

Output 33.8.5 shows the estimation results of the model with measurement errors in both x and y. The effect of Fx on Fy is 1.1791 (standard error=0.1029). This is essentially the same effect of Fx on y as in the preceding measurement model in which no measurement error in y is assumed.

Output 33.8.5 Estimates of the Measurement Error Model with Errors in x and y

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x <=== Fx		1.00000			
Fx ===> Fy	_Parm1	1.17914	0.10288	11.4615	<.0001
Fy ===> y		1.00000			

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	x		0.01900			
	y		0.02200			
Exogenous	Fx	_Parm2	0.78062	0.20999	3.7174	0.0002
Error	Fy	_Parm3	0.18486	0.06126	3.0176	0.0025

The estimated error variance for Fy in the current model is 0.1849 and the measurement error variance of y is fixed at 0.022, as shown in the last table of Output 33.8.5. The sum is 0.2069, which is the same amount of error variance for y in the preceding model with measurement error assumed only in x. Hence, the assumption of the measurement error in y does not change the structural effect of Fx on y (same amount of effect Fx on Fy, which is 1.1791). It only changes the variance components of y. In the preceding model with measurement error assumed only in x, the total error variance in y is 0.2069. In the current model, this total error variance is partitioned into the measurement error variance (which is fixed at 0.022) and the error variance in the regression on Fx (which is estimated at 0.1849).

Linear Regression Model as a Special Case of Structural Equation Model

By using the current measurement error model as an illustration, it is easy to see that the structural equation model is a more general model that includes the linear regression model as a special case. If you restrict the measurement error variances in x and y to zero, the measurement error model (which represents the structural equation model in this example) reduces to the linear regression model. That is, the path diagram becomes:

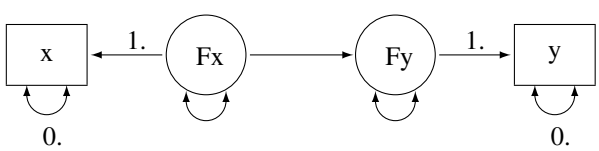

You can then specify the PATH model by the following statements:

```
proc calis data=measures;
   path
      x  <=== Fx  = 1.,
      Fx ===> Fy   ,
      Fy ===> y   = 1.;
   pvar
      x  = 0.,
      y  = 0.,
      Fx Fy;
run;
```

Output 33.8.6 shows the estimation results of this measurement error model with zero measurement errors. The estimate of the regression coefficient is 1.1511, which is essentially the same result as in Output 33.8.2 by using PROC REG.

Output 33.8.6 Estimates of the Measurement Error Model with Zero Measurement Errors

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x <=== Fx		1.00000			
Fx ===> Fy	_Parm1	1.15112	0.10016	11.4924	<.0001
Fy ===> y		1.00000			

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	x		0			
	y		0			
Exogenous	Fx	_Parm2	0.79962	0.20999	3.8079	0.0001
Error	Fy	_Parm3	0.23265	0.06110	3.8079	0.0001

This example shows that you can apply PROC CALIS to fit measurement error models. You treat true scores variables as latent variables in the structural equation model. The linear regression model is a special case of the structural equation model (or measurement error model) where measurement error variances are assumed to be zero. Structural equation modeling by PROC CALIS is not limited to this simple modeling scenario. PROC CALIS can treat more complicated measurement error models. In Example 33.9 and Example 33.10, you fit measurement error models with parameter constraints and with more than one predictor. You can also fit measurement error models with correlated errors.

Example 33.9: Testing Specific Measurement Error Models

(View the complete code for this example (calex203.sas) in the example repository.)

In Example 33.8, you used the PATH modeling language of PROC CALIS to fit some basic measurement error models. In this example, you continue to fit the same kind of measurement error models but you restrict some model parameters to test some specific hypotheses.

This example uses the same data set as is used in Example 33.8. This data set contains 30 observations for the x and y variables. The general measurement error model with measurement errors in both x and y is shown in the following path diagram:

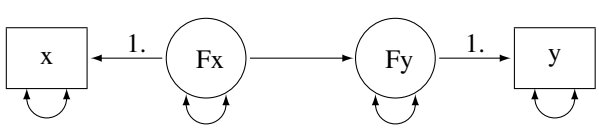

In the path diagram, two paths are fixed with a path coefficient of 1. They are required in the model for representing the relationships between true scores (latent) and measured indicators (observed). In Example 33.8, you consider several different modeling scenarios, all of which require you to make some

parameter restrictions to estimate the models. You fix the measurement error variances to certain values that are based on prior knowledge or studies. Without those fixed error variances, those models would have been overparameterized and the parameters would not have been estimable.

For example, in the current path diagram, five of the single- or double-headed paths are not labeled with fixed numbers. Each of these paths represents a free parameter in the model. However, in the covariance structure model analysis, you fit these free parameters to the three nonredundant elements of the sample covariance matrix, which is a 2×2 symmetric matrix. Hence, to have an identified model, you can at most have three free parameters in your covariance structure model. However, the path diagram shows that you have five free parameters in the model. You must introduce additional parameter constraints to make the model identified.

If you do not have prior knowledge about the measurement error variances (as those described in Example 33.8), then you might need to make some educated guesses about how to restrict the overparameterized model. For example, if x and y are of the same kind of measurements, perhaps you can assume that they have an equal amount of measurement error variance. Furthermore, if the measurement errors have been taken into account, in some physical science studies you might be able to assume that the relationship between the true scores Fx and Fy is almost deterministic, resulting in a near zero error variance of Fy.

The assumptions here are not comparable to prior knowledge or studies about the measurement error variances. If you suppose they are reasonable enough in a particular field, you can use these assumptions to give you an identified model to work with (at least as an exploratory study) when the required prior knowledge is lacking. The following path diagram incorporates these two assumptions in the measurement error model:

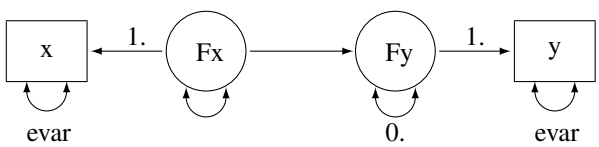

In the path diagram, you use evar to denote the error variances of x and y. This implicitly constrains the two error variances to be equal. The error variance of Fy is labeled zero, indicating a fixed parameter value and a deterministic relationship between x and y. You can transcribe this path diagram into the following PATH modeling specification:

```
proc calis data=measures;
   path
      x  <=== Fx  = 1.,
      Fx ===> Fy   ,
      Fy ===> y   = 1.;
   pvar
      x  = evar,
      y  = evar,
      Fy = 0.,
      Fx;
run;
```

In the PVAR statement, you specify the same parameter name evar for the error variances of x and y. This way their estimates are constrained to be the same in the estimation. In addition, the error variance for Fy is fixed at zero, which reflects the "near-deterministic" assumption about the relationship between Fx and Fy. These two assumptions effectively reduce the overparameterized model by two parameters so that the new model is just-identified and estimable.

Output 33.9.1 shows the estimation results. The estimated effect of Fx on Fy is 1.3028 (standard error = 0.1134). The measurement error variances for x and y are both estimated at 0.0931 (standard error = 0.0244).

Output 33.9.1 Estimates of the Measurement Error Model with Equal Measurement Error Variances

PATH List						
Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x <=== Fx		1.00000			
Fx ===> Fy	_Parm1	1.30275	0.11336	11.4924	<.0001
Fy ===> y		1.00000			

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	x	evar	0.09307	0.02444	3.8079	0.0001
	y	evar	0.09307	0.02444	3.8079	0.0001
	Fy		0			
Exogenous	Fx	_Parm2	0.70655	0.20962	3.3706	0.0008

Testing the Effect of Fx on Fy

Suppose you are interested in testing the hypothesis that the effect of Fx on Fy (that is, the regression slope) is 1. The following path diagram represents the model under the hypothesis:

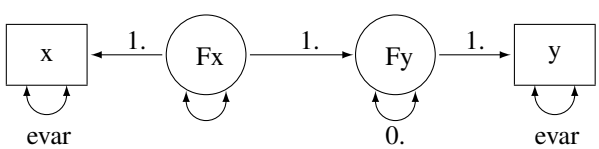

Now you label the path from Fx to Fy with a fixed constant 1, which reflects the hypothesis you want to test. You can transcribe the current path diagram easily into the following PROC CALIS specification:

```
proc calis data=measures;
   path
      x  <=== Fx  = 1.,
      Fx ===> Fy  = 1.,  /* Testing a fixed constant effect */
      Fy ===> y   = 1.;
   pvar
      x  = evar,
      y  = evar,
      Fy = 0.,
      Fx;
run;
```

Output 33.9.2 shows the model fit chi-square statistic. The model fit chi-square test here essentially is a test of the null hypothesis of the constant effect at 1 because the alternative hypothesis is a saturated model. The chi-square value is 8.1844 ($df = 1$, $p = .0042$), which is statistically significant. This means that the hypothesis of constant effect at 1 is rejected.

Output 33.9.2 Fit Summary for Testing Constant Effect

Fit Summary	
Chi-Square	8.1844
Chi-Square DF	1
Pr > Chi-Square	0.0042

Output 33.9.3 shows the estimates under this restricted model. In the first table of Output 33.9.3, all path effects or coefficients are fixed at 1. In the second tale of Output 33.9.3, estimates of the error variances are 0.1255 (standard error = 0.0330) for both x and y. The error variance of Fy is a fixed zero, as required in the hypothesis. The variance estimate of Fx is 0.9205 (standard error = 0.2587).

Output 33.9.3 Estimates of Constant Effect Measurement Error Model

PATH List

Path	Estimate	Standard Error	t Value	Pr > |t|
x <=== Fx	1.00000			
Fx ===> Fy	1.00000			
Fy ===> y	1.00000			

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	x	evar	0.12545	0.03295	3.8079	0.0001
	y	evar	0.12545	0.03295	3.8079	0.0001
	Fy		0			
Exogenous	Fx	_Parm1	0.92046	0.25872	3.5577	0.0004

Testing a Zero Intercept

Suppose you are interested in testing the hypothesis that the intercept for the regression of Fy on Fx is zero, while the regression effect is freely estimated. Because the intercept parameter belongs to the mean structures, you need to specify this parameter in PROC CALIS to test the hypothesis.

There are two ways to include the mean structure analysis. First, you can include the MEANSTR option in the PROC CALIS statement. Alternatively, you can use the MEAN statement to specify the means and intercepts in the model. The following statements specify the model under the zero intercept hypothesis:

```
proc calis data=measures;
   path
      x  <=== Fx  = 1.,
      Fx ===> Fy  ,       /* regression effect is freely estimated */
      Fy ===> y   = 1.;
   pvar
      x  = evar,
      y  = evar,
      Fy = 0.,
      Fx;
   mean
      x y = 0. 0., /* Intercepts are zero in the measurement error model */
```

```
        Fy  = 0.,     /* Fixed to zero under the hypothesis */
        Fx;           /* Mean of Fx is freely estimated */
  run;
```

In the PATH statement, the regression effect of Fx on Fy is freely estimated. In the MEAN statement, you specify the means or intercepts of the variables. Each variable in your measurement error model has either a mean or an intercept (but not both) to specify. If a variable is exogenous (independent), you can specify its mean in the MEAN statement. Otherwise, you can specify its intercept in the MEAN statement. Variables x and y in the measurement error model are both endogenous. They are measured indicators of their corresponding true scores Fx and Fy. Under the measurement error model, their intercepts are fixed zeros. The intercept for Fy is zero under the current hypothesized model. The mean of Fx is freely estimated under the model. This parameter is specified in the MEAN statement but is not named.

Output 33.9.4 shows the model fit chi-square statistic. The chi-square value is 10.5397 ($df = 1$, $p = .0012$), which is statistically significant. This means that the zero intercept hypothesis for the regression of Fy on Fx is rejected.

Output 33.9.4 Fit Summary for Testing Zero Intercept

Fit Summary	
Chi-Square	10.5397
Chi-Square DF	1
Pr > Chi-Square	0.0012

Output 33.9.5 shows the estimates under the hypothesized model. The effect of Fx on Fy is 1.8169 (standard error = 0.0206). In the last table of Output 33.9.5, the estimate of the mean of Fx is 6.9048 (standard error = 0.1388). The intercepts for all other variables are fixed at zero under the hypothesized model.

Output 33.9.5 Estimates of the Zero Intercept Measurement Error Model

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x <=== Fx		1.00000			
Fx ===> Fy	_Parm1	1.81689	0.02055	88.4247	<.0001
Fy ===> y		1.00000			

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**x**	evar	0.13684	0.03594	3.8079	0.0001
	y	evar	0.13684	0.03594	3.8079	0.0001
	Fy		0			
Exogenous	**Fx**	_Parm2	0.42280	0.11990	3.5261	0.0004

Means and Intercepts

Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	**x**		0			
	y		0			
	Fy		0			
Mean	**Fx**	_Parm3	6.90483	0.13881	49.7431	<.0001

Measurement Model with Means and Intercepts Freely Estimated

In the preceding model, you fit a restricted regression model with a zero intercept. You reject the null hypothesis and conclude that this intercept is significantly different from zero. The alternative hypothesis is a saturated model with the intercept freely estimated. The model under the alternative hypothesis is specified in the following statements:

```
proc calis data=measures;
   path
      x  <=== Fx  = 1.,
      Fx ===> Fy  ,
      Fy ===> y   = 1.;
   pvar
      x  = evar,
      y  = evar,
      Fy = 0.,
      Fx;
   mean
      x y = 0. 0.,
      Fy Fx;
run;
```

Output 33.9.6 shows that model fit chi-square statistic is zero. This is expected because you are fitting a measurement error model with saturated mean and covariance structures.

Output 33.9.6 Fit Summary of the Saturated Measurement Model with Mean Structures

Fit Summary	
Chi-Square	0.0000
Chi-Square DF	0
Pr > Chi-Square	.

Output 33.9.7 shows the estimates under the measurement model with saturated mean and covariance structures. The effect of Fx on Fy is 1.3028 (standard error=0.1134), which is considerably smaller than the corresponding estimate in the restricted model with zero intercept, as shown in Output 33.9.5. The intercept estimate of Fy is 3.5800 (standard error = 0.7864), with a significant t value of 4.55.

Output 33.9.7 Estimates of the Saturated Measurement Model with Mean Structures

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x <=== Fx		1.00000			
Fx ===> Fy	_Parm1	1.30275	0.11336	11.4924	<.0001
Fy ===> y		1.00000			

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	x	evar	0.09307	0.02444	3.8079	0.0001
	y	evar	0.09307	0.02444	3.8079	0.0001
	Fy		0			
Exogenous	Fx	_Parm2	0.70655	0.20962	3.3706	0.0008

Output 33.9.7 *continued*

Means and Intercepts						
Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	x		0			
	y		0			
	Fy	_Parm3	3.57998	0.78641	4.5523	<.0001
Mean	Fx	_Parm4	6.88865	0.16605	41.4850	<.0001

In this example, you fit some measurement error models with some parameter constraints that reflect the hypothesized models of interest. You can set equality constraints by simply providing the same parameter names in the PATH model specification of PROC CALIS. You can also fix parameters to constants. In the MEAN statement, you can specify the intercepts and means of the variables in the measurement error models. You can apply all these techniques to more complicated measurement error models with multiple predictors, as shown in Example 33.10, where you also fit measurement error models with correlated errors.

Example 33.10: Measurement Error Models with Multiple Predictors

(View the complete code for this example (calex204.sas) in the example repository.)

In Example 33.8 and Example 33.9, you fit various measurement error models with only one predictor. This example illustrates the case in which you have more than one predictor, all measured with errors. The measurement errors might also be correlated.

The data from 37 observations are summarized in a covariance matrix as shown in the following SAS DATA step:

```
data multiple(type=cov);
   input _type_ $ 1-4  _name_ $ 6-8 @10 y x1 x2 x3;
   datalines;
mean     0.93   1.33   1.34   4.11
cov  y   1.31    .      .      .
cov  x1  1.24   1.42    .      .
cov  x2  0.21   0.18   1.15    .
cov  x3  3.91   4.21   0.58  14.11
;
```

In this data set, four variables are measured. Variables x1–x3 are predictors of y. Instead of the raw data, you can input the sample covariance matrix in the form of a SAS data set for PROC CALIS to analyze.

You assume all of these variables in the data set are measured with errors. From prior studies, you establish the knowledge about the measurement errors of these variables. You create the true score counterparts for each of these variables in the same manner as you do in Example 33.8 and Example 33.9. The following path diagram represents your measurement error model for the data:

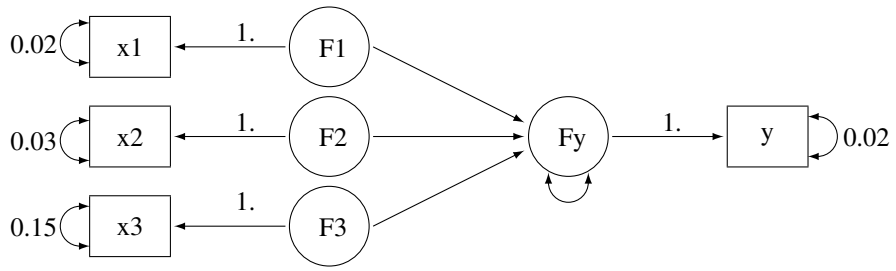

In the path diagram, variables F1–F3 and Fy are latent variables that represent the true score for the measured indicators x1–x3 and y, respectively. All paths from the true scores to the corresponding measured indicators are labeled with the fixed constant 1, as required by the measurement model. Each measured indicator is attached with a double-headed arrow that indicates the error variance. Because you have knowledge about these measurement error variances, you put fixed constant values adjacent to these double-headed arrows. For example, the measurement error variance of y is 0.02 and the measurement error variance of x3 is 0.15. The path diagram also indicates that the paths from F1–F3 to Fy and the error variance for Fy are free parameters to estimate in the model.

Notice that for brevity the variances and covariances among the three exogenous true score variables F1–F3 are not represented in the path diagram. These six variance and covariance parameters could have been represented by double-headed arrows in the path diagram. However, because PROC CALIS always assumes the exogenous variances and covariances as default model parameters, this information is not represented to reduce clutter in the path diagram.

You can transcribe the path diagram easily to the following PATH model specification:

```
proc calis data=multiple nobs=37;
   path
      Fy  <===   F1 F2 F3,
      F1  ===>   x1   = 1.,
      F2  ===>   x2   = 1.,
      F3  ===>   x3   = 1.,
      Fy  ===>   y    = 1.;
   pvar
      x1 x2 x3 y = .02 .03 .15 .02,
      Fy;
run;
```

In the first entry of the PATH statement, you specify that F1–F3 predicts Fy. In the next four path entries you specify the measurement model for the true scores and how they are related to the observed variables. In the PVAR statement, you specify all the known measurement error variances for the observed variables. They are all fixed constants in the model. In the last entry in the PVAR statement, you specify the error variance of Fy as a free (unnamed) parameter. You could have omitted this entry because error variances for all endogenous variables in the PATH model are free parameters by default. Setting these default parameters explicitly as free parameters would not affect model fitting.

Output 33.10.1 shows the parameter estimates of the model. The path coefficient or effect from F2 to Fy is not significant, while the other two path coefficients are at least marginally significant.

Output 33.10.1 Parameter Estimates of the Measurement Model with Multiple Predictors

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Fy <=== F1	_Parm1	0.46507	0.22682	2.0503	0.0403
Fy <=== F2	_Parm2	0.04123	0.07069	0.5832	0.5597
Fy <=== F3	_Parm3	0.13812	0.07175	1.9249	0.0542
F1 ===> x1		1.00000			
F2 ===> x2		1.00000			
F3 ===> x3		1.00000			
Fy ===> y		1.00000			

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**x1**		0.02000			
	x2		0.03000			
	x3		0.15000			
	y		0.02000			
	Fy	_Parm4	0.16461	0.04522	3.6403	0.0003
Exogenous	**F1**	_Add1	1.40000	0.33470	4.1829	<.0001
	F2	_Add2	1.12000	0.27106	4.1320	<.0001
	F3	_Add3	13.96000	3.32576	4.1975	<.0001

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
F2	**F1**	_Add4	0.18000	0.21508	0.8369	0.4027
F3	**F1**	_Add5	4.21000	1.02416	4.1107	<.0001
F3	**F2**	_Add6	0.58000	0.67829	0.8551	0.3925

The second table of Output 33.10.1 shows the variance estimates. As specified in the model, all measurement error variances for the observed variables are fixed constants. The error variance of Fy is 0.1646 (standard error =0.0452). Although you do not specify them in the PATH model specification, variances of F1–F3 are free parameters in the model. The second table of Output 33.10.1 shows their estimates. The last table of Output 33.10.2 shows the covariances among the latent true scores. Only the covariance between F3 and F1 is significant.

PROC CALIS not only can handle measurement error variance with multiple true score predictors, but it also can handle correlated errors. Suppose that the measurement errors for variables x1 and x2 are correlated. From prior studies, you determine that their covariance is 0.01. The path diagram with this new piece of information added is shown in the following:

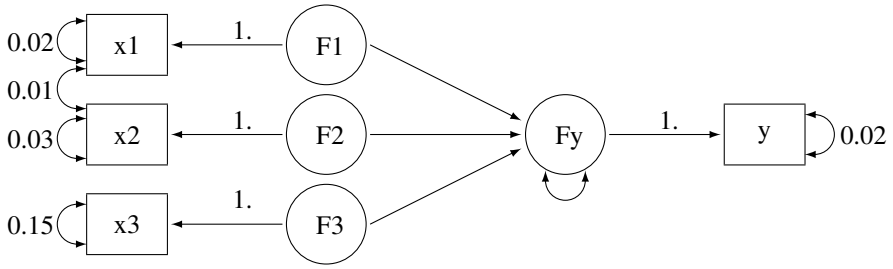

In the path diagram, the double-headed arrow that connects x1 and x2 represents the covariance between the error terms for the two variables. The value attached to this double-headed arrow is 0.01, which represents a fixed constant in the model. The PATH model specification is similar to the preceding specification, with one more entry added in the PCOV statement, as shown in the following statements:

```
proc calis data=multiple nobs=37;
   path
      Fy  <===   F1 F2 F3,
      F1  ===>   x1   = 1.,
      F2  ===>   x2   = 1.,
      F3  ===>   x3   = 1.,
      Fy  ===>   y    = 1.;
   pvar
      x1 x2 x3 y = .02 .03 .15 .02,
      Fy;
   pcov
      x1 x2 = 0.01;
run;
```

Except for the PCOV statement specification, everything else is the same as in the preceding specification. In the PCOV statement, you can specify covariance or error covariances between exogenous or endogenous variables. In the current model, because both x1 and x2 are endogenous in the model, the specification is for their error covariance, which is fixed at 0.01 as required.

Output 33.10.2 shows the parameter estimates of the measurement model with correlated errors. The estimates do not change much from the preceding analysis in which correlated errors is not assumed. Perhaps the correlation between the errors in the current model is so small that it is ignorable. The last table in Output 33.10.2 shows the covariance estimates among errors. This table is unique to the current model. It shows that the measurement errors for x1 and x2 have a covariance of 0.01, which is treated as a fixed constant in the current model.

Output 33.10.2 Parameter Estimates of the Measurement Model with Multiple Predictors: Correlated Errors

PATH List

Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Fy <=== F1	_Parm1	0.46839	0.22695	2.0639	0.0390
Fy <=== F2	_Parm2	0.04549	0.07074	0.6431	0.5202
Fy <=== F3	_Parm3	0.13694	0.07194	1.9035	0.0570
F1 ===> x1		1.00000			
F2 ===> x2		1.00000			
F3 ===> x3		1.00000			
Fy ===> y		1.00000			

Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**x1**		0.02000			
	x2		0.03000			
	x3		0.15000			
	y		0.02000			
	Fy	_Parm4	0.16421	0.04523	3.6305	0.0003
Exogenous	**F1**	_Add1	1.40000	0.33470	4.1829	<.0001
	F2	_Add2	1.12000	0.27106	4.1320	<.0001
	F3	_Add3	13.96000	3.32576	4.1975	<.0001

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
F2	**F1**	_Add4	0.17000	0.21508	0.7904	0.4293
F3	**F1**	_Add5	4.21000	1.02416	4.1107	<.0001
F3	**F2**	_Add6	0.58000	0.67829	0.8551	0.3925

Covariances Among Errors

Error of	Error of	Estimate	Standard Error	t Value	Pr > |t|
x1	**x2**	0.01000			

This example shows how you can use PROC CALIS to fit measurement error models with multiple true score predictors. You can also fit models with correlated errors. The model specification tool is the PATH modeling language, which ties closely to the path diagram representations.

However, some researchers might prefer to use linear equations to represent the measurement error models. PROC CALIS provides the LINEQS modeling language for specifying the measurement error models, or mean and covariance structure models in general. Example 33.11 illustrates the LINEQS model specification of the measurement error models.

Example 33.11: Measurement Error Models Specified as Linear Equations

(View the complete code for this example (calex205.sas) in the example repository.)

In Example 33.8, you fit a simple measurement error model with errors in both of the predictor variable x and the outcome variable y. From prior studies, the measurement error variance of x is 0.019 and the measurement error variance of y is 0.022. You use the following path diagram to represent the model:

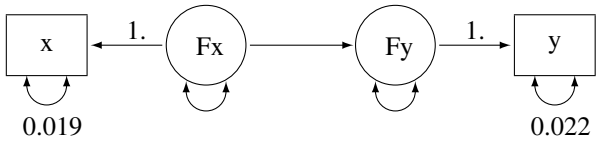

With this path diagram, you use the PATH modeling language of PROC CALIS to specify the model, as shown in the following:

```
proc calis data=measures;
   path
      x  <=== Fx  = 1.,
      Fx ===> Fy   ,
      Fy ===> y   = 1.;
   pvar
      x  = 0.019,
      y  = 0.022,
      Fx Fy;
run;
```

In the path diagram and in the PATH model specification, there are no explicit representations of the error terms in the model. You express the error variances of x, y, and Fy as partial variances of the endogenous variables. In the path diagram, you represent these partial variances by the double-headed arrows. Correspondingly, in the PATH statement of PROC CALIS, you specify these partial variances in the PVAR statement.

In practice, some researchers might prefer to express the error terms in the model explicitly. For example, with the error terms added to the preceding measurement error model, the new path diagram becomes:

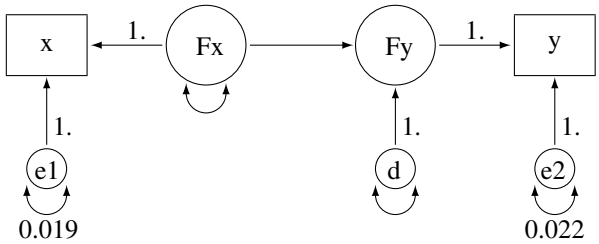

In the path diagram, you add paths from error variables e1, e2, and d to the endogenous variables x, y, and Fy, respectively. All these paths from the error terms have a fixed path coefficient of 1. The error variances are represented by double-headed arrows directly attached to them. For example, the variance of e1 is fixed at 0.019, and the variance of e2 is fixed at 0.022. The variance of d, which is sometime called the disturbance, is a free unnamed parameter in the path diagram. Similarly, the variance of Fx is a free unnamed parameter in the model.

Corresponding to this new path diagram, you can use the LINEQS modeling language for specifying your model in PROC CALIS, as shown in the following statements:

```
proc calis data=measures;
   lineqs
      x   = 1. * Fx  +  e1,
      y   = 1. * Fy  +  e2,
      Fy  =    * Fx  +   d;
   variance
      e1  = 0.019,
      e2  = 0.022,
      Fx d;
run;
```

The LINEQS model specification in PROC CALIS emphasizes the linear equation input. In each of the linear equations in the LINEQS statement, you specify an endogenous variable and how it is related to other variables. An endogenous variable in the path diagram is a variable that has at least one single-headed arrow pointing to it. You need to list all endogenous variables on the left-hand side of the linear equations of the LINEQS statement. In the current model, variables x, y, and Fy are endogenous, and therefore you specify three linear equations in the LINEQS statement. The first two equations represent the measurement model for the observed variables, while the third equation represents the structural equation of the model. Notice that in the third equation, you do not specify the path coefficient that is attached to Fx. PROC CALIS treats unspecified path coefficients as free parameters. The effect of Fx on Fy is freely estimated, as required in the path diagram representation.

In the VARIANCE statement, you specify the variances of the exogenous variables in the model. The specifications in the VARIANCE statement of the LINEQS model are very similar to those in the PVAR statement of the PATH model. The main difference is the use of error variable names in the VARIANCE statement. With the LINEQS model specification, you can only specify exogenous variables in the VARIANCE statement. Hence, you must specify the error variables e1, e2, and d in the VARIANCE statement of the LINEQS model, instead of the corresponding endogenous variables x, y, and Fy in the PVAR statement of the PATH model.

Output 33.11.1 shows the parameter estimates of the LINEQS model.

Output 33.11.1 LINEQS Parameter Estimates of the Measurement Model for the Measures Data

Linear Equations

x	=	1.0000		Fx	+	1.0000	e1
y	=	1.0000		Fy	+	1.0000	e2
Fy	=	1.1791	(**)	Fx	+	1.0000	d

Effects in Linear Equations

Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x	Fx		1.00000			
y	Fy		1.00000			
Fy	Fx	_Parm1	1.17914	0.10288	11.4615	<.0001

Output 33.11.1 *continued*

Estimates for Variances of Exogenous Variables						
Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	e1		0.01900			
	e2		0.02200			
Latent	Fx	_Parm2	0.78062	0.20999	3.7174	0.0002
Disturbance	d	_Parm3	0.18486	0.06126	3.0176	0.0025

All these estimates are essentially the same as those obtained from the PATH model specification, as shown in Output 33.11.2.

Output 33.11.2 PATH Parameter Estimates of the Measurement Model for the Measures Data

PATH List						
Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
x	<===	Fx		1.00000			
Fx	===>	Fy	_Parm1	1.17914	0.10288	11.4615	<.0001
Fy	===>	y		1.00000			

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	x		0.01900			
	y		0.02200			
Exogenous	Fx	_Parm2	0.78062	0.20999	3.7174	0.0002
Error	Fy	_Parm3	0.18486	0.06126	3.0176	0.0025

You can use either the LINEQS or PATH model specification in PROC CALIS for your analysis problems. They give you the same estimation results.

So far the measurement error model is concerned with one predictor. With more predictors in the model, you might also want to model the correlated measurement errors in the x variables. You can analyze this kind of model by using the PATH model specification, as shown in Example 33.10. With measurement error terms explicitly assumed, you can also use the LINEQS model specification. This example illustrates how you can do that by using the same data set and the measurement error model with correlated errors in Example 33.10.

In the data set, you have four observed variables: x1–x3 and y. All are measured with errors, as represented by the following path diagram:

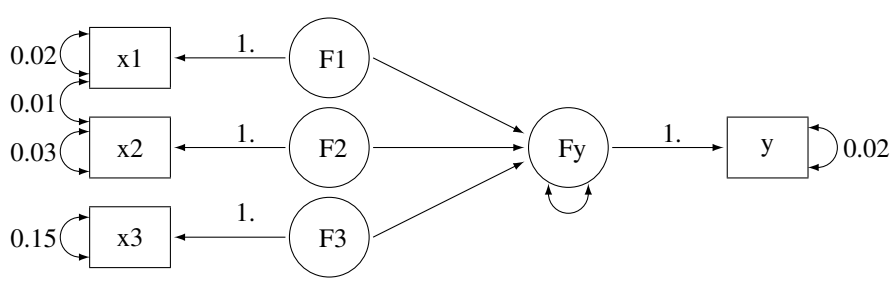

In the path diagram, F1–F3 and Fy represent true scores for the measurement indicators x1–x3 and y, respectively. You predict Fy by F1–F3, which represents the structural relationships in the model. Measurement error variances of the observed variables are treated as known and are represented by the double-headed arrows attached to the observed variables. For example, the error variance of x3 is 0.15. In addition, the error covariance between x1 and x2 is treated as known. The double-headed arrow that connects x1 and x2 represents the error covariance, and this covariance is fixed at 0.01 in the model.

You transcribe this path diagram representation into the following PATH model specification:

```
proc calis data=multiple nobs=37;
   path
      Fy  <===   F1 F2 F3,
      F1  ===>   x1   = 1.,
      F2  ===>   x2   = 1.,
      F3  ===>   x3   = 1.,
      Fy  ===>   y    = 1.;
   pvar
      x1 x2 x3 y = .02 .03 .15 .02,
      Fy;
   pcov
      x1 x2 = 0.01;
run;
```

To represent the error terms explicitly, you can add the error terms to the path diagram with some modifications, as shown in the following:

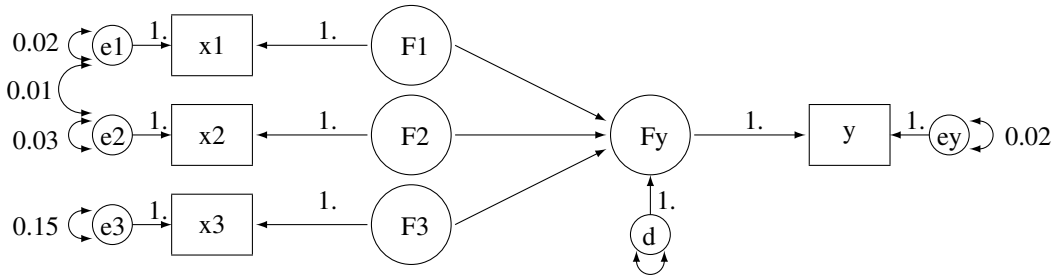

In the path diagram, you attach error variables e1–e3, ey, and d to the associated endogenous variables in the model. The error variances and covariances, which are attached to the endogenous variables directly, are now attached to the corresponding error variables. With this new path diagram, you can use the following LINEQS model specification for the model:

```
proc calis data=multiple nobs=37;
   lineqs
      Fy =     * F1 +  * F2 +  * F3 + d,
      x1 =  1. * F1 + e1,
      x2 =  1. * F2 + e2,
      x3 =  1. * F3 + e3,
       y =  1. * Fy + ey;
    variance
       e1-e3 ey = .02 .03 .15 .02,
       d;
    cov
       e1 e2 = 0.01;
```

```
run;
```

Again, in each linear equation of the LINEQS statement, you specify the functional relationship of an endogenous variable with other variables, including the error variable. The first equation is the structural equation in the model. You want to estimate the effects of F1, F2, and F3 on Fy. The error or disturbance variable is d. In the next four equations, you relate the observed variables with their true scores counterparts.

In the VARIANCE statement, you specify the error variances with reference to the error variables in the path diagram. Four of the error variances are fixed constants, as required in the model. The last specification represents a free parameter for the variance of d. The specifications in the VARIANCE statement of the LINEQS model are similar to those in the PVAR statement of the PATH model specification. The difference is that in the PATH model specification the reference variables are the endogenous variables in the PATH model, while in the LINEQS model specification the reference variables are the associated error variables.

In the COV statement, you specify the covariance between the error variables e1 and e2. Again, this is similar to the corresponding specification of the PATH model, where the same error covariance is specified as the partial covariance between x1 and x2 in the PCOV statement.

Output 33.11.3 shows the parameter estimates that result from using the LINEQS model specification. Estimates in the equations, variances, and covariances are shown respectively.

Output 33.11.3 Parameter Estimates of the Measurement Model with Multiple Predictors: LINEQS Model

Linear Equations

Fy =	0.4684	(**)	F1 +	0.0455	(ns)	F2 +	0.1369	(ns)	F3 +	1.0000 d
x1 =	1.0000		F1 +	1.0000		e1				
x2 =	1.0000		F2 +	1.0000		e2				
x3 =	1.0000		F3 +	1.0000		e3				
y =	1.0000		Fy +	1.0000		ey				

Effects in Linear Equations

Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Fy	F1	_Parm1	0.46839	0.22695	2.0639	0.0390
Fy	F2	_Parm2	0.04549	0.07074	0.6431	0.5202
Fy	F3	_Parm3	0.13694	0.07194	1.9035	0.0570
x1	F1		1.00000			
x2	F2		1.00000			
x3	F3		1.00000			
y	Fy		1.00000			

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**e1**		0.02000			
	e2		0.03000			
	e3		0.15000			
	ey		0.02000			
Disturbance	**d**	_Parm4	0.16421	0.04523	3.6305	0.0003
Latent	**F1**	_Add1	1.40000	0.33470	4.1829	<.0001
	F2	_Add2	1.12000	0.27106	4.1320	<.0001
	F3	_Add3	13.96000	3.32576	4.1975	<.0001

Output 33.11.3 *continued*

Covariances Among Exogenous Variables						
Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
e1	e2		0.01000			
F2	F1	_Add4	0.17000	0.21508	0.7904	0.4293
F3	F1	_Add5	4.21000	1.02416	4.1107	<.0001
F3	F2	_Add6	0.58000	0.67829	0.8551	0.3925

Output 33.11.4 shows the parameter estimates that result from using the PATH model specification. The estimates in the path list shown in Output 33.11.4 correspond to those of the equation output in Output 33.11.3. The variance estimates in Output 33.11.4 correspond to those variance estimates of the exogenous variables of the LINEQS model, as shown in Output 33.11.3. Finally, the last two tables in Output 33.11.4 correspond to the covariance estimates among the exogenous variables of the LINEQS model, as shown in Output 33.11.3. Again, the LINEQS and PATH model specification give you exactly the same estimation results, but in different output formats.

Output 33.11.4 Parameter Estimates of the Measurement Model with Multiple Predictors: PATH Model

PATH List						
Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
Fy	<===	F1	_Parm1	0.46839	0.22695	2.0639	0.0390
Fy	<===	F2	_Parm2	0.04549	0.07074	0.6431	0.5202
Fy	<===	F3	_Parm3	0.13694	0.07194	1.9035	0.0570
F1	===>	x1		1.00000			
F2	===>	x2		1.00000			
F3	===>	x3		1.00000			
Fy	===>	y		1.00000			

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	x1		0.02000			
	x2		0.03000			
	x3		0.15000			
	y		0.02000			
	Fy	_Parm4	0.16421	0.04523	3.6305	0.0003
Exogenous	F1	_Add1	1.40000	0.33470	4.1829	<.0001
	F2	_Add2	1.12000	0.27106	4.1320	<.0001
	F3	_Add3	13.96000	3.32576	4.1975	<.0001

Covariances Among Exogenous Variables						
Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
F2	F1	_Add4	0.17000	0.21508	0.7904	0.4293
F3	F1	_Add5	4.21000	1.02416	4.1107	<.0001
F3	F2	_Add6	0.58000	0.67829	0.8551	0.3925

Output 33.11.4 *continued*

Covariances Among Errors					
Error of	Error of	Estimate	Standard Error	t Value	Pr > |t|
x1	x2	0.01000			

In this example, you fit measurement error models by using the LINEQS and PATH model specifications of PROC CALIS. The two different model specification languages give you essentially the same estimation results. The measurement models can have multiple true scores predictors and correlated errors. The measurement error models considered so far have only one measured indicator for each true score latent variable. This is usually not the case in many psychometric or sociological applications where latent factors usually have several observed indicators. The confirmatory factor model is a typical example of this kind of applications. See Example 33.12 for an application of PROC CALIS to fit confirmatory factor models. See Example 33.17 for an application of PROC CALIS to fit a general structural equation model where latent variables have more than one measured indicators.

Example 33.12: Confirmatory Factor Models

(View the complete code for this example (calex112.sas) in the example repository.)

This example shows how you can fit a confirmatory factor analysis model by the FACTOR modeling language. Thirty-two students take tests of their verbal and math abilities. Six tests are administered separately. Tests x1–x3 test their verbal skills and tests y1–y3 test their math skills.

The data are shown in the following DATA step:

```
data scores;
   input x1 x2 x3 y1 y2 y3;
   datalines;
 23  17  16  15  14  16
 29  26  23  22  18  19
 14  21  17  15  16  18
 20  18  17  18  21  19
 25  26  22  26  21  26
 26  19  15  16  17  17
 14  17  19   4   6   7
 12  17  18  14  16  13
 25  19  22  22  20  20
  7  12  15  10  11   8
 29  24  30  14  13  16
 28  24  29  19  19  21
 12   9  10  18  19  18
 11   8  12  15  16  16
 20  14  15  24  23  16
 26  25  21  24  23  24
 20  16  19  22  21  20
 14  19  15  17  19  23
 14  20  13  24  26  25
 29  24  24  21  20  18
 26  28  26  28  26  23
```

```
 20  23  24  22  23  22
 23  24  20  23  22  18
 14  18  17  13  16  14
 28  34  27  25  21  21
 17  12  10  14  12  16
  8   1  13  14  15  14
 22  19  19  13  11  14
 18  21  18  15  18  19
 12  12  10  13  13  16
 22  14  20  20  18  19
 29  21  22  13  17  12
;
```

Because of the unambiguous nature of the tests, you hypothesize that this is a confirmatory factor model with two factors: one is the verbal ability factor and the other is the math ability factor. You can represent such a confirmatory factor model by the following path diagram:

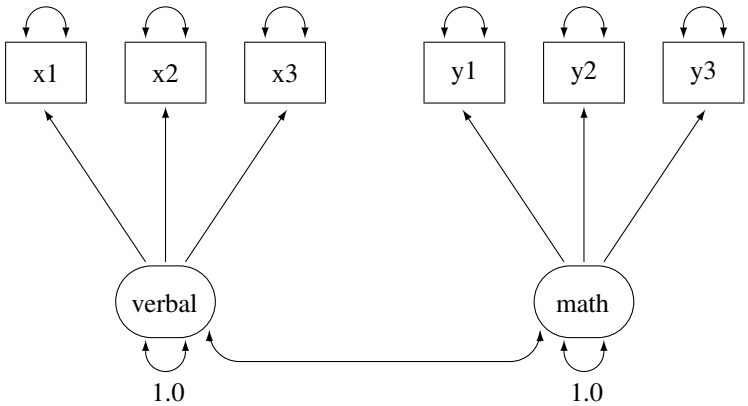

In the path diagram, there are two clusters of variables. One cluster is for the verbal factor and the other is for the math factor. The single-headed arrows in the path diagram represent functional relationships between factors and the observed variables. The double-headed arrows that point to single variables represent variances of the factors or error variances of the observed variables. The double-headed arrow that connect the two factors represents their covariance. All but two of these arrows are not labeled with numbers. Each of the unlabeled arrows represents a free parameter in the confirmatory factor model. You label the double-headed arrows that attach to the two factors with the constant 1. This means that the variances of the factors are fixed at 1.0 in the model.

You can specify the confirmatory factor model by the FACTOR model language of PROC CALIS, as shown in the following statements:

```
proc calis data=scores;
   factor
      verbal ===> x1-x3,
      math   ===> y1-y3;
   pvar
      verbal = 1.,
      math   = 1.;
run;
```

In each of the entry of the FACTOR statement, you specify a latent factor, followed by a list of observed variables that are functionally related to the latent factor. For example, in the first entry, the verbal factor is related to variables x1–x3, as shown by the single-headed arrows in the path diagram. In fact, all single-headed arrows in the path diagram are specified in the FACTOR statement. Notice that each entry of the FACTOR statement must take the format of

```
factor_name ===> variable_list
```

You cannot reverse the arrow specification as in the following:

```
variable_list <=== factor_name
```

Nor you can have a specification such as the following:

```
variable_list ===> factor_name
```

However, you can specify the functional relationships between factors and variables in different entries. For example, you can specify the same confirmatory factor model by the following statements:

```
title "Basic Confirmatory Factor Model: Separate Path Entries";
title2 "FACTOR Model Specification";
proc calis data=scores;
   factor
      verbal ===> x1,
      verbal ===> x2,
      verbal ===> x3,
      math   ===> y1,
      math   ===> y2,
      math   ===> y3;
   pvar
      verbal = 1.,
      math   = 1.;
   fitindex  noindextype on(only)=[chisq df probchi rmsea  srmr bentlercfi];
run;
```

In the PVAR statement, which is for the specification of variances or error variances, you fix the variances of the latent factors to 1. This completes the model specification of the confirmatory factor model, although you do not specify other arrows in the path diagram as free parameters in these statements. The reason is that in the FACTOR modeling language, the variances and covariances among factors and the error variances of the observed variables are default parameters in the confirmatory factor model. It is not necessary to specify these parameters (or the corresponding arrows in the path diagram) explicitly if they are free parameters in the model. You can also specify these free parameters explicitly without affecting the estimation. However, if these parameters (or the corresponding double-headed arrows in the path diagram) are intended to be constrained parameters or fixed values, you must specify them explicitly. For example, in the current confirmatory factor model, you must provide explicit specifications for the variances of the verbal and the math factors because these parameters are fixed at 1.

Output 33.12.1 shows the modeling information and the variables in the confirmatory factor model.

Output 33.12.1 Modeling Information and Variables of the CFA Model: Scores Data

Simple Confirmatory Factor Model
FACTOR Model Specification

The CALIS Procedure
Covariance Structure Analysis: Model and Initial Values

Modeling Information	
Maximum Likelihood Estimation	
Data Set	WORK.SCORES
N Records Read	32
N Records Used	32
N Obs	32
Model Type	FACTOR
Analysis	Covariances

Variables in the Model	
Variables	x1 x2 x3 y1 y2 y3
Factors	verbal math
Number of Variables = 6	
Number of Factors = 2	

In the beginning of the output, PROC CALIS shows the data set, the number of observations, the model type, and the analysis type. The default analysis type in PROC CALIS is covariances (that is, covariance structures). If you want to analyze the correlation structures instead, you can use the CORR option in the PROC CALIS statement. Next, PROC CALIS shows the list of variables and factors in the model. As expected, the number of variables is 6 and the number of factors is 2.

Output 33.12.2 shows the initial model specifications of the confirmatory factor model.

Output 33.12.2 Initial Specification of the CFA Model: Scores Data

Initial Factor Loading Matrix	verbal	math
x1	. [_Parm1]	0
x2	. [_Parm2]	0
x3	. [_Parm3]	0
y1	0	. [_Parm4]
y2	0	. [_Parm5]
y3	0	. [_Parm6]

Output 33.12.2 *continued*

Initial Factor Covariance Matrix		
	verbal	**math**
verbal	1.0000	. [_Add1]
math	. [_Add1]	1.0000

Initial Error Variances		
Variable	**Parameter**	**Estimate**
x1	_Add2	.
x2	_Add3	.
x3	_Add4	.
y1	_Add5	.
y2	_Add6	.
y3	_Add7	.

NOTE:
Parameters with prefix
'_Add'
are added by PROC CALIS.

The first table of Output 33.12.2 shows the pattern of factor loadings of the variables on the two latent factors. As expected, x1–x3 have nonzero loadings only on the verbal factor, while y1–y3 have nonzero loadings on the math factor. PROC CALIS names these free parameters automatically with the "_Parm" prefix and unique numerical suffixes. There are six parameters in the factor loading matrix with six different parameter names.

The next table of Output 33.12.2 shows the covariance matrix of the factors. The variances of the factors are fixed at one, as shown on the diagonal of the covariance matrix. The covariance between the two factors is a free parameter named _Add1. You did not specify this covariance parameter explicitly in the factor model specification. By default, PROC CALIS assumes that latent factors are correlated. Default free parameters added by PROC CALIS have the _Add prefix for their names. If you do not want to assume the covariances among the factors, you must specify zero covariances in the COV statement. For example, the following statement specifies that the math and verbal factors have zero covariance:

```
COV
  math verbal = 0.;
```

The last table of Output 33.12.2 shows the error variance parameters of the observed variables. By default PROC CALIS assumes these error variances are free parameters in the confirmatory factor model. These added parameters are named with the _Add prefix. However, as all other default parameters that are assumed by PROC CALIS, you can overwrite the default by using explicit specifications. You can specify the error variances of a confirmatory factor model explicitly in the PVAR statement. See specifications in Example 33.13.

Output 33.12.3 shows the fit summary of the confirmatory factor model for the scores data.

Output 33.12.3 Fit Summary of the CFA Model: Scores Data

Fit Summary	
Chi-Square	9.8052
Chi-Square DF	8
Pr > Chi-Square	0.2790
Standardized RMR (SRMR)	0.0571
RMSEA Estimate	0.0853
Bentler Comparative Fit Index	0.9887

The model fit chi-square is 9.805 (df = 8, p = 0.279). This shows that statistically you cannot reject the confirmatory factor model for the test scores. However, the root mean square error of approximation (RMSEA) estimate is 0.0853, which is greater than the conventional 0.05 value for a good model fit. The standardized root mean square residual (SRMR) is 0.0571, which is close to the conventional 0.05 value for a good model fit. Bentler's comparative fit index is 0.9887, which indicates a very good model fit. Overall, the model seems to be quite reasonable for the data.

Output 33.12.4 shows the loading and factor covariance estimates of the confirmatory factor model for the scores data. The first table shows the loading estimates, together with the standard error estimates and the t values. In structural equation modeling, the significance of the parameter estimates is usually inferred by comparing the t values with the critical value of a standardized normal variate (that is, the z-table). Therefore, estimates with associated (absolute) t values greater than 1.96 are significant at α=.05. In Output 33.12.4, all the t values for the loading estimates are greater than 2. This indicates that the prescribed relationships between the variables and the factors are significant.

Output 33.12.4 Loading and Factor Covariance Estimates of the CFA Model: Scores Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value	verbal	math
x1	5.8406 0.9962 5.8629 <.0001 [_Parm1]	0
x2	5.8182 0.9537 6.1004 <.0001 [_Parm2]	0
x3	4.6619 0.7814 5.9662 <.0001 [_Parm3]	0
y1	0	5.2804 0.6998 7.5455 <.0001 [_Parm4]
y2	0	4.2003 0.6220 6.7532 <.0001 [_Parm5]
y3	0	3.7596 0.6341 5.9289 <.0001 [_Parm6]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value	verbal	math
verbal	1.0000	0.5175 0.1429 3.6221 0.000292 [_Add1]
math	0.5175 0.1429 3.6221 0.000292 [_Add1]	1.0000

The second table of Output 33.12.4 shows the covariance matrix of the verbal and the math factors. Because the factor variances are fixed at one, the covariance estimate is also the correlation between the two factors. Output 33.12.4 shows that the two factors are moderately correlated with a correlation estimate of 0.5175, which is statistically significant.

Output 33.12.5 shows the estimates of the error variances. All but the error variance of y1 are significant. This suggests that y1 might have an almost perfect relationship with the math factor.

Output 33.12.5 Error Variance Estimates of the CFA Model: Scores Data

Error Variances					
Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add2	11.52376	4.26398	2.7026	0.0069
x2	_Add3	9.14503	3.83219	2.3864	0.0170
x3	_Add4	6.68169	2.59770	2.5722	0.0101
y1	_Add5	0.78580	1.29440	0.6071	0.5438
y2	_Add6	2.88069	1.09395	2.6333	0.0085
y3	_Add7	5.15573	1.46854	3.5108	0.0004

Output 33.12.6 echoes this same fact. The R-squares in this table shows the percentages of variance of the variables that are overlapped with the factors. While all these percentages (0.74 – 0.97) are quite high for all variables, the percentage is especially high for y1. It shares 97% of the variance with the math factor. So, it appears that the observed variable y1 is almost a perfect indicator of the math factor.

Output 33.12.6 Squared Multiple Correlations of the CFA Model: Scores Data

Squared Multiple Correlations			
Variable	Error Variance	Total Variance	R-Square
x1	11.52376	45.63609	0.7475
x2	9.14503	42.99597	0.7873
x3	6.68169	28.41532	0.7649
y1	0.78580	28.66835	0.9726
y2	2.88069	20.52319	0.8596
y3	5.15573	19.29032	0.7327

Alternative Identification Constraints

Setting the variances of the latent factors to 1 in the preceding FACTOR model specification makes the model identified. This is necessary because the scales of the latent factors are arbitrary and the constraints imposed on the factor variances fix the scales of the factors.

In practice, there is another way to fix the scales of the factors. For each factor, you can fix the loading of one of its measured indicators to a constant. This fixed loading value is usually set at 1. For example, you can represent the confirmatory factor model for the scores data by the following alternative path diagram:

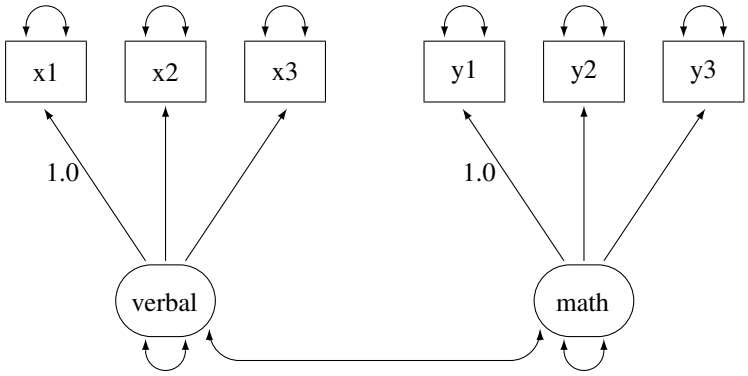

This path diagram is essentially the same as the preceding one. However, the fixed constants adjacent to the double-headed arrows that attach to the two factors in the preceding path diagram are now moved to two of the single-headed paths in the current path diagram.

You can specify this path diagram by the following FACTOR model specification of PROC CALIS:

```
ods graphics on;
proc calis data=scores plots=pathdiagram;
   factor
      verbal ===> x1-x3   = 1. ,
      math   ===> y1-y3   = 1. ;
run;
ods graphics off;
```

The PLOTS=PATHDIAGRAM option requests the path diagram output. In the FACTOR statement, you assign a fixed constant to each of the path entries. In the first entry, the constant 1 is assigned to the loading of x1 on the verbal factor, while all other loadings in this entry are (unnamed) free parameters. Similarly, in the second entry, the fixed constant 1 is assigned to the loading of y1 on the math factor, while all other loadings in this entry are (unnamed) free parameters. This completes the specification of the confirmatory factor model because all the double-headed arrows in the path diagram correspond to default free parameters in the FACTOR modeling language of PROC CALIS.

Output 33.12.7 shows some fit indices for the current confirmatory factor model for the scores data.

Output 33.12.7 Fit Summary of the CFA Model with Alternative Identification Constraints: Scores Data

Fit Summary	
Chi-Square	9.8052
Chi-Square DF	8
Pr > Chi-Square	0.2790
Standardized RMR (SRMR)	0.0571
RMSEA Estimate	0.0853
Bentler Comparative Fit Index	0.9887

The model fit chi-square is 9.805 ($df = 8$, $p = 0.279$). This is the same model fit chi-square as that for the preceding CFA model specification with factor variances constrained to 1. In fact, all fit information in

Output 33.12.7 are identical to Output 33.12.3.

Figure 33.12.8 shows the path diagram of the confirmatory factor model. The path diagram indicates significant estimates by attaching asterisks to the numerical values. Estimates that are flagged with one asterisk are significant at 0.05 α-level. Estimates that are flagged with two asterisks are significant at 0.01 α-level. The path diagram also shows a summary of fit statistics. For more information about specifying path diagram output, see the section "Path Diagrams: Layout Algorithms, Default Settings, and Customization" on page 1784.

Output 33.12.8 Path Diagram and Fit Summary: Scores Data

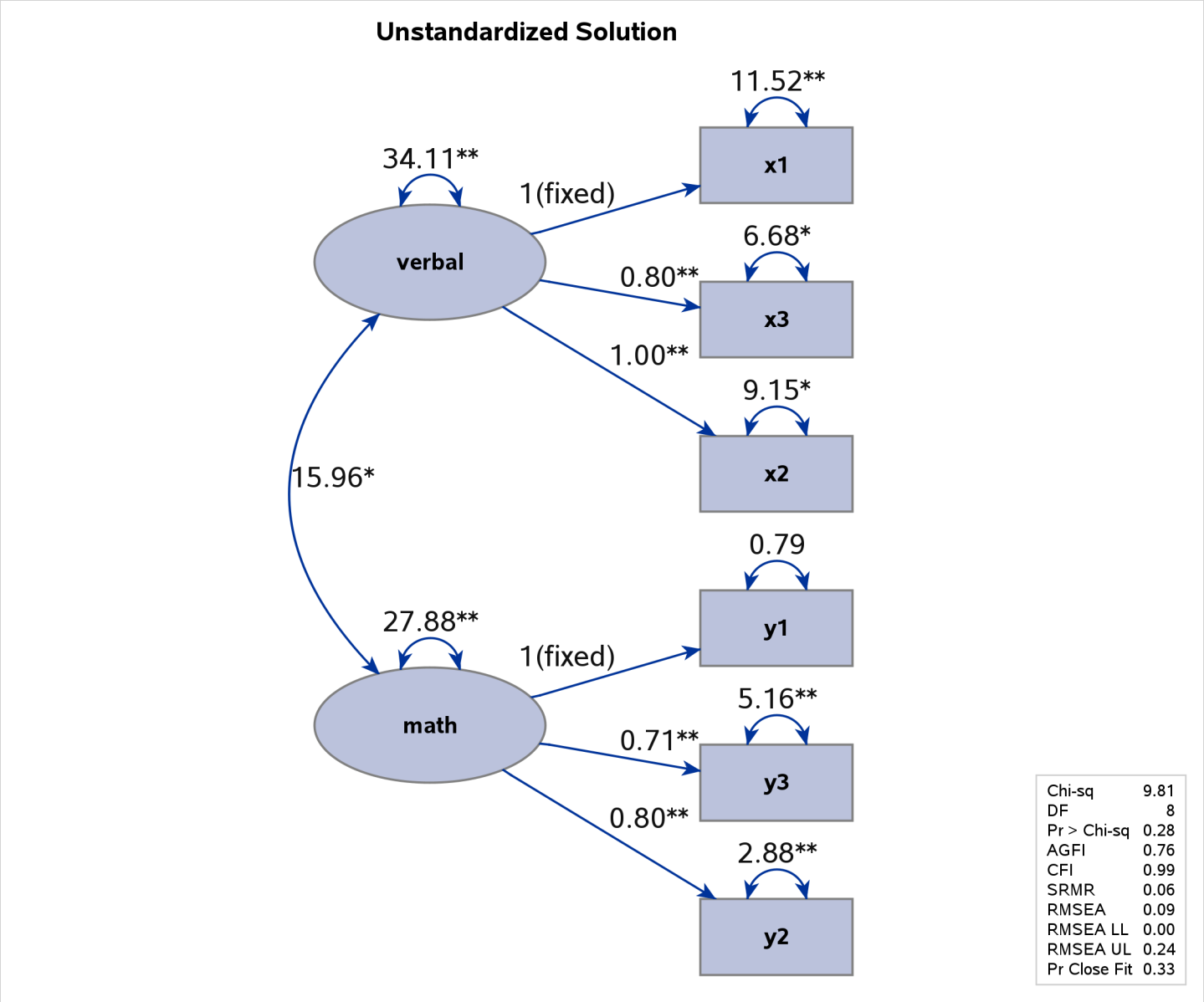

Output 33.12.9 shows the parameter estimates under the current model specification. The loading of x1 on the verbal factor is a fixed at 1, as required for the identification of the scale of the verbal factor. Similarly, the loading of y1 on the math factor is a fixed at 1 for the identification of the scale of the math factor. All

other loading estimates in Output 33.12.9 are not the same as those in the preceding model specification, as shown in Output 33.12.4. The reason is that the scales of the factors (as measured by the estimated standard deviations of the factors) in the two specifications are not the same. In the current model specification, the verbal factor has an estimated variance of 34.1123 and the math factor has an estimated variance of 27.8825, as shown in the second table of Output 33.12.9. Hence, the estimated standard deviations of these two factors are 5.8406 and 5.2804, respectively. But the standard deviations of the factors in the preceding confirmatory factor model specification are fixed at 1.

Output 33.12.9 Loading and Factor Covariance Estimates of the CFA Model with Alternative Identification Constraints: Scores Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
x1	1.0000	0
x2	0.9962 0.1576 6.3194 <.0001 [_Parm1]	0
x3	0.7982 0.1286 6.2083 <.0001 [_Parm2]	0
y1	0	1.0000
y2	0	0.7955 0.0718 11.0820 <.0001 [_Parm3]
y3	0	0.7120 0.0858 8.3027 <.0001 [_Parm4]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
verbal	34.1123 11.6366 2.9315 0.003374 [_Add1]	15.9585 6.7270 2.3723 0.0177 [_Add3]
math	15.9585 6.7270 2.3723 0.0177 [_Add3]	27.8825 7.3905 3.7727 0.000161 [_Add2]

However, if you multiply the loading estimates in Output 33.12.9 by the corresponding estimated factor standard deviation, you get the same set of loading estimates as in Output 33.12.4. For example, the loading of x1 on the verbal factor is 1.0 in Output 33.12.9. Multiplying this loading by the estimated standard deviation 5.8406 of the verbal factor gives you the same corresponding loading as in Output 33.12.4. Another example is the loading of y3 on the math factor. This loading is 0.7120 in Output 33.12.9. Multiplying this estimate by the estimated standard deviation 5.2804 of the verbal factor gives an estimate of 3.7596, which matches the corresponding loading estimate in Output 33.12.4. Therefore, the discrepancies in the loading estimates are due to different factor scales in the two specifications. The loading estimates in Output 33.12.9 are simply rescaled version of the loading estimates in Output 33.12.4.

However, the scales of the factors do not affect the estimates of the error variances, as shown in Output 33.12.10. These estimates are the same as those for the preceding model specification, as shown in the Output 33.12.5.

Output 33.12.10 Error Variance Estimates of the CFA Model with Alternative Identification Constraints: Scores Data

Error Variances					
Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add4	11.52376	4.26398	2.7026	0.0069
x2	_Add5	9.14503	3.83219	2.3864	0.0170
x3	_Add6	6.68169	2.59770	2.5722	0.0101
y1	_Add7	0.78580	1.29440	0.6071	0.5438
y2	_Add8	2.88069	1.09395	2.6333	0.0085
y3	_Add9	5.15573	1.46854	3.5108	0.0004

This example shows how you can fit a basic confirmatory factor model by the FACTOR modeling language of PROC CALIS. You can set the identification constraints and get statistically equivalent estimation results in two different ways. By setting up additional parameter constraints, you can also fit some variations of the basic confirmatory factor model. See Example 33.13 for illustrations of some restricted confirmatory factor models for the scores data.

When your data have missing values, with the default ML estimation method PROC CALIS deletes all observations with missing values for the analysis. This might result in a serious loss of information. Example 33.15 considers a hypothetical situation where some observations in the scores data have missing values in the observed variables. Only 16 observations have complete data. By using the full information maximum likelihood (FIML) method for treating the missing data, Example 33.15 shows how you can fully use the information from the scores data set with missing values.

Example 33.13: Confirmatory Factor Models: Some Variations

(View the complete code for this example (calex115.sas) in the example repository.)

This example shows how you can fit some variations of the basic confirmatory factor analysis model by the FACTOR modeling language. You apply these models to the scores data set that is described in Example 33.12. The data set contains six test scores of verbal and math abilities. Thirty-two students take the tests. Tests x1–x3 test their verbal skills and tests y1–y3 test their math skills.

The Parallel Tests Model

In classical measurement theory, test items for a latent factor are parallel if they have the same loadings on the factor and the same error variances (or reliability). Suppose for the scores data, the items within each of the verbal and the math factors are parallel. You can use the following path diagram to represent such a parallel tests model:

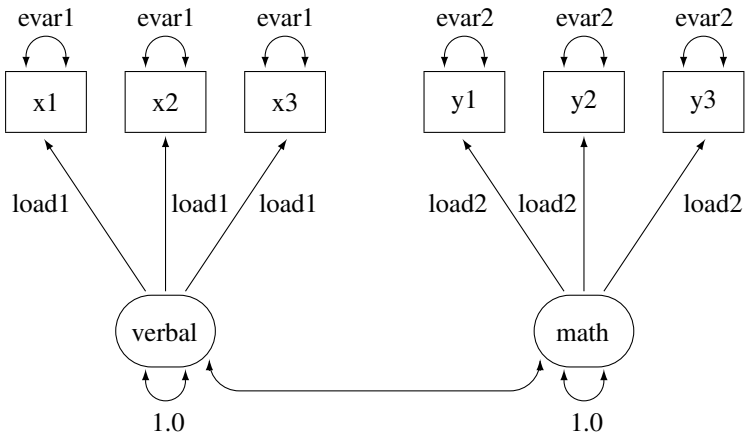

In the path diagram, the variances of the verbal and the math are both fixed at 1, as indicated by the constants 1.0 adjacent to the double-headed arrows that are attached to factors. You label all the single-headed paths in the path diagram by parameter names. For the three paths (loadings) from the verbal factor, you use the same parameter name load1. This means that these loadings are the same parameter. You also label the double-headed arrows that are attached to x1–x3 by the parameter name evar1. This means that the corresponding error variances for these three observed variables are exactly the same. Hence, x1–x3 are parallel tests for the verbal factor, as required by the current confirmatory factor model.

Similarly, you define parallel tests y1–y3 for the math factor by using load2 as the common factor loading parameter and evar2 as the common error variances for the observed variables.

Corresponding to this path diagram, you can specify the model by the following FACTOR model specification of PROC CALIS:

```
proc calis data=scores;
   factor
      verbal ===> x1-x3   = load1 load1 load1,
      math   ===> y1-y3   = load2 load2 load2;
   pvar
```

```
        verbal = 1.,
        math   = 1.,
        x1-x3  = 3*evar1,
        y1-y3  = 3*evar2;
run;
```

In each entry of the FACTOR statement, you specify the factor-variables relationships, followed by a list of parameters. For example, the three loading parameters of x1–x3 on the verbal factor are all named load1. This effectively constrains the corresponding loading estimates to be the same. Similarly, in the next entry you set equality constraints on the loading estimates y1–y3 on the math factor by using the same parameter name load2.

To make the tests parallel, you also need to constrain the error variances for each variable cluster. In the PVAR statement, in addition to setting the factor variances to 1 for identification, you set all the error variances of x1–x3 to be the same by using the same parameter name evar1. The notation **3*evar1** means that you want to specify evar1 three times, one time each for the error variances for the three observed variables in the variable list of the entry. Similarly, you set the equality of the error variances of y1–y3 by using the same parameter name evar2.

Output 33.13.1 shows some fit indices of the parallel tests model for the scores data. The model fit chi-square is 26.128 ($df = 16$, $p = 0.0522$). The SRMR value is 0.1537 and the RMSEA value is 0.1429. All these indices show that the model does not fit very well. However, Bentler's CFI is 0.9366, which shows a good model fit.

Output 33.13.1 Model Fit of the Parallel Tests Model: Scores Data

Fit Summary	
Chi-Square	26.1283
Chi-Square DF	16
Pr > Chi-Square	0.0522
Standardized RMR (SRMR)	0.1537
RMSEA Estimate	0.1429
Bentler Comparative Fit Index	0.9366

Output 33.13.2 shows the parameter estimates of the parallel tests model. The first table of Output 33.13.2 shows the required factor pattern for parallel tests. Variables x1–x3 all have the same loading estimates on the verbal factor, and variables y1–y3 all have the same loading estimates on the math factor. All loading estimates are statistically significant.

Output 33.13.2 Parameter Estimates of the Parallel Tests Model: Scores Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
x1	5.4226 0.7655 7.0833 <.0001 [load1]	0
x2	5.4226 0.7655 7.0833 <.0001 [load1]	0
x3	5.4226 0.7655 7.0833 <.0001 [load1]	0
y1	0	4.4001 0.5926 7.4246 <.0001 [load2]
y2	0	4.4001 0.5926 7.4246 <.0001 [load2]
y3	0	4.4001 0.5926 7.4246 <.0001 [load2]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
verbal	1.0000	0.5024 0.1497 3.3569 0.000788 [_Add1]
math	0.5024 0.1497 3.3569 0.000788 [_Add1]	1.0000

Error Variances

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	evar1	9.61122	1.72623	5.5678	<.0001
x2	evar1	9.61122	1.72623	5.5678	<.0001
x3	evar1	9.61122	1.72623	5.5678	<.0001
y1	evar2	3.46673	0.62264	5.5678	<.0001
y2	evar2	3.46673	0.62264	5.5678	<.0001
y3	evar2	3.46673	0.62264	5.5678	<.0001

In the second table of Output 33.13.2, the factor covariance (or correlation) estimate is 0.5024, showing moderate relationship between the verbal and the math factors. The last table of Output 33.13.2 shows the error variances of the variables. As required by the parallel tests model, the error variance estimates of x1–x3 are all 9.6112, and the error variance estimates of y1–y3 are all 3.4667.

The Tau-Equivalent Tests Model

Because the parallel tests model does not fit well, you are looking for a less constrained model for the scores data. The tau-equivalent tests model is such a model. It requires only the equality of factor loadings but not the equality of error variances within each factor. The following path diagram represents the tau-equivalent tests model for the scores data:

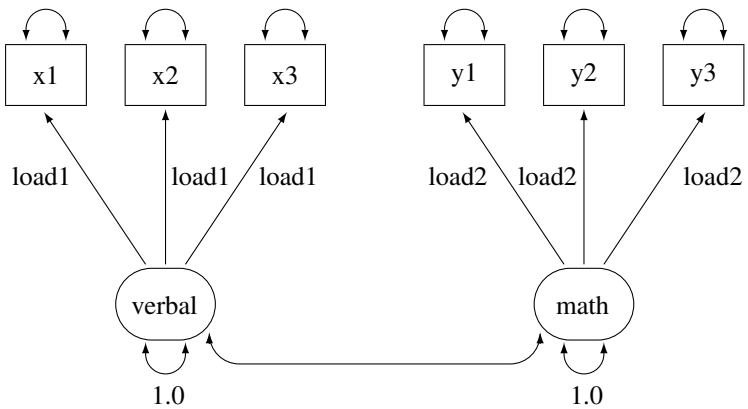

This path diagram is much the same as that for the parallel tests model except that now you do not use parameter names to label the double-headed arrows that are attached to the observed variables. This means that you allow the corresponding error variances to be free parameters in the tau-equivalent tests model. You can use the following FACTOR model specification of PROC CALIS to specify the tau-equivalent tests model for the scores data:

```
proc calis data=scores;
   factor
      verbal ===> x1-x3   = load1 load1 load1,
      math   ===> y1-y3   = load2 load2 load2;
   pvar
      verbal = 1.,
      math   = 1.;
run;
```

This specification is the same as that for the parallel tests model except that you remove the specifications about the error variances in the PVAR statement in the current tau-equivalent model. This effectively allows the error variances of the observed variables to be (default) free parameters in the model.

Output 33.13.3 shows some model fit indices of the tau-equivalent tests model for the scores data. The chi-square is 22.0468 (df = 12, p = 0.037). The SRMR is 0.1398 and the RMSEA is 0.1643. The comparative fit index (CFI) is 0.9371. Except for the CFI value, all other values do not support a good model fit. This model has a degrees of freedom of 12, which is less restrictive (has more parameters) than the parallel tests model, which has a degrees of freedom of 16, as shown in Output 33.13.1. However, it seems that the tau-equivalent tests model is still too restrictive for the data.

Output 33.13.3 Model Fit of the Tau-Equivalent Tests Model: Scores Data

Fit Summary	
Chi-Square	22.0468
Chi-Square DF	12
Pr > Chi-Square	0.0370
Standardized RMR (SRMR)	0.1398
RMSEA Estimate	0.1643
Bentler Comparative Fit Index	0.9371

Output 33.13.4 shows the parameter estimates. The first table of Output 33.13.4 shows the required pattern of factor loadings under the tau-equivalent tests model. The third table of Output 33.13.4 shows the error variance estimates. The error variance parameters are no longer constrained under the tau-equivalent tests model. Each has a unique estimate.

Output 33.13.4 Parameter Estimates of the Tau-Equivalent Tests Model: Scores Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
x1	5.2418 0.7374 7.1085 <.0001 [load1]	0
x2	5.2418 0.7374 7.1085 <.0001 [load1]	0
x3	5.2418 0.7374 7.1085 <.0001 [load1]	0
y1	0	4.4462 0.5932 7.4953 <.0001 [load2]
y2	0	4.4462 0.5932 7.4953 <.0001 [load2]
y3	0	4.4462 0.5932 7.4953 <.0001 [load2]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
verbal	1.0000	0.4514 0.1569 2.8772 0.004012 [_Add1]
math	0.4514 0.1569 2.8772 0.004012 [_Add1]	1.0000

Error Variances

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add2	13.05681	4.19549	3.1121	0.0019
x2	_Add3	10.80421	3.70322	2.9175	0.0035
x3	_Add4	5.43527	2.72147	1.9972	0.0458
y1	_Add5	3.29858	1.24673	2.6458	0.0082
y2	_Add6	1.90435	1.02393	1.8598	0.0629
y3	_Add7	5.09724	1.61477	3.1566	0.0016

The Partially Constrained Parallel Tests Model

Because both the parallel tests and tau-equivalent tests models do not fit the data well, you can explore an alternative model for the scores data. Suppose that for each factor only two (but not all) of their measured variables (tests) are parallel. For example, suppose you know that tests x1 and x2 are very similar to each other (for example, both are speeded tests with forced-choice answers), while x3 is a little different in the way it is administered (for example, open-ended questions). Although all tests are designed for measuring the verbal factor, only x1 and x2 are parallel tests while x3 is congeneric to the verbal factor. Similarly, suppose you can argue that y2 and y3 are parallel tests while y1 is only congeneric to the math factor.

The current modeling idea is represented by the following path diagram:

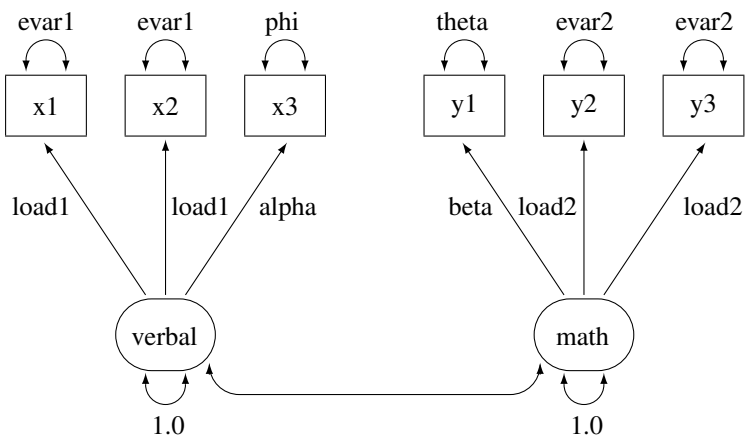

In the path diagram, x1 and x2 have the same parameter load1 for the paths from the verbal factor. Their error variances are also the same, as labeled with the evar1 parameter adjacent to the double-headed arrows that are attached to the variables. The test x3 has distinct parameter names for its associated path and the attached double-headed arrow. The corresponding loading and error variance parameters are alpha and phi, respectively. Similarly, with the use of specific parameter names, you define y2 and y3 as parallel tests for the math factor, while y1 is congeneric to the same factor but with distinct loading and error variance parameters. Lastly, you fix the variances of the factors to 1.0 for identification of the factor scales.

You can specify such a partially constrained parallel tests model by the following FACTOR model specification of PROC CALIS:

```
proc calis data=scores;
   factor
      verbal ===> x1-x3   = load1 load1 alpha,
      math   ===> y1-y3   = beta  load2 load2;
   pvar
      verbal = 1.,
      math   = 1.,
      x1-x3  = evar1  evar1  phi,
      y1-y3  = theta  evar2  evar2;
run;
```

First, in the FACTOR statement, you name the loading parameters that reflect the parallel tests constraints. For example, the loading parameters of x1 and x2 on the verbal factor are both named load1. This means that they are the same. However, the loading parameter of x3 on the verbal factor is named alpha, which means

that it is a separate parameter. Similarly, you apply the load2 parameter name to the loading parameters of y2 and y3 on the math factor, but the loading parameter of y1 on the math factor is a distinct parameter named beta.

In the PVAR statement, the two factor variances are set to a constant 1 for the identification of latent factor scales. Next, you use the same naming techniques as in the FACTOR statement to constrain some parts of the error variances. As a result, together with the specifications in the FACTOR statement, x1 and x2 are parallel tests for the verbal factor and y2 and y3 are parallel tests for the math factor, while x3 and y1 are only congeneric tests for their respective factors.

Output 33.13.5 shows some fit indices of the partially constrained parallel tests model. The model fit chi-square is 12.6784 ($df = 12$, $p = 0.3928$). The SRMR is 0.0585 and the RMSEA is close to 0.0427. The comparative fit index (CFI) is 0.9958. All these fit indices point to a quite reasonable model fit for the scores data.

Output 33.13.5 Model Fit of the Partially Constrained Parallel Tests Model: Scores Data

Fit Summary	
Chi-Square	12.6784
Chi-Square DF	12
Pr > Chi-Square	0.3928
Standardized RMR (SRMR)	0.0585
RMSEA Estimate	0.0427
Bentler Comparative Fit Index	0.9958

Notice that the current model actually has the same degrees of freedom as that of the tau-equivalent tests model, as shown in Output 33.13.3. Both models have nine parameters. But the current partially constrained parallel tests model is definitely a better model for the data. This shows that sometimes you do not have to add more parameters to improve the model fit. Structurally different models might explain the data quite differently, even though they might use the same number of parameters.

Output 33.13.6 show the parameter estimates of the partially constrained parallel tests model for the scores data. The estimates in the factor loading matrix and error variances table confirm the prescribed nature of the tests—that is, x1 and x2 are parallel tests for the verbal factor and y2 and y3 are parallel tests for the math factor.

Output 33.13.6 Parameter Estimates of the Partially Constrained Parallel Tests Model: Scores Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value	verbal	math
x1	5.8306 0.8593 6.7853 <.0001 [load1]	0
x2	5.8306 0.8593 6.7853 <.0001 [load1]	0
x3	4.6623 0.7814 5.9664 <.0001 [alpha]	0
y1	0	5.2784 0.7010 7.5294 <.0001 [beta]
y2	0	3.9789 0.5732 6.9419 <.0001 [load2]
y3	0	3.9789 0.5732 6.9419 <.0001 [load2]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value	verbal	math
verbal	1.0000	0.5203 0.1425 3.6497 0.000263 [_Add1]
math	0.5203 0.1425 3.6497 0.000263 [_Add1]	1.0000

Error Variances

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	evar1	10.31998	2.57827	4.0027	<.0001
x2	evar1	10.31998	2.57827	4.0027	<.0001
x3	phi	6.67832	2.59902	2.5696	0.0102
y1	theta	0.80714	1.35247	0.5968	0.5506
y2	evar2	4.07534	1.00371	4.0603	<.0001
y3	evar2	4.07534	1.00371	4.0603	<.0001

Example 33.14: Residual Diagnostics and Robust Estimation

(View the complete code for this example (calex206.sas) in the example repository.)

This example illustrates case-level residual diagnostics and robust estimation with PROC CALIS. The data set is available in Mardia, Kent, and Bibby (1979). It contains 88 responses on five subjects: mechanics, vector, algebra, analysis, and statistics. Yuan and Hayashi (2010) use the same data set to illustrate the uses of residual diagnostics in the context of robust estimation. Inspired by their illustrations, this example mirrors their one-factor model, with some modifications.

Case-Level Residual Diagnostics

The following DATA step specifies and inputs five observed variables for the data set mardia:

```
data mardia;
   input mechanics vector algebra analysis statistics;
   datalines;
77.000     82.000     67.000     67.000     81.000
63.000     78.000     80.000     70.000     81.000
75.000     73.000     71.000     66.000     81.000
  .          .          .          .          .
  .          .          .          .          .
     { More Data }
  .          .          .          .          .
  .          .          .          .          .
  .          .          .          .          .
  .          .          .          .          .
;
```

You specify a one-factor model for the five variables by using the following statements:

```
ods graphics on;

proc calis data=mardia residual plots=caseresid;
   path
      fact1 ===>  mechanics vector algebra analysis statistics = 1. ;
run;

ods graphics off;
```

In the PATH statement, all observed variables are indicators of the latent factor fact1. The first factor loading for mechanics is fixed at 1 for identification of the latent variable scale, while the remaining four loadings are free parameters in the model. The RESIDUAL option that is specified in the PROC CALIS statement requests residual analysis. Because raw data are input, case-level residual analysis is done. To request all the available ODS Graphics output for case-level residual diagnostics, you specify the PLOTS=CASERESID option. To enable ODS Graphics, you specify the ODS GRAPHICS ON statement before running the CALIS procedure.

Output 33.14.1 shows some of the fit statistics of the one-factor model. The model fit chi-square is 8.9782 ($df = 5$, $p = 0.1099$). Therefore, you fail to reject the one-factor model for the data. The SRMR is 0.0411, which shows a good fit. But the RMSEA is 0.0956, which does not show that the one-factor model is a very

good approximating model for the data. Output 33.14.2 shows the loading estimates. All are statistically significant.

Output 33.14.1 Model Fit of Mardia Data

Fit Summary	
Chi-Square	8.9782
Chi-Square DF	5
Pr > Chi-Square	0.1099
Standardized RMR (SRMR)	0.0475
RMSEA Estimate	0.0956

Output 33.14.2 Estimates of Loadings

PATH List						
Path	**Parameter**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
fact1 ===> mechanics		1.00000			
fact1 ===> vector	_Parm1	0.83770	0.16598	5.0469	<.0001
fact1 ===> algebra	_Parm2	0.93064	0.15369	6.0552	<.0001
fact1 ===> analysis	_Parm3	1.09480	0.19611	5.5826	<.0001
fact1 ===> statistics	_Parm4	1.19229	0.22309	5.3444	<.0001

Output 33.14.3 shows the classifications of outlier and leverage observations in a two-dimensional space. Four regions are clearly shown.

Output 33.14.3 Outliers and Leverage Observations

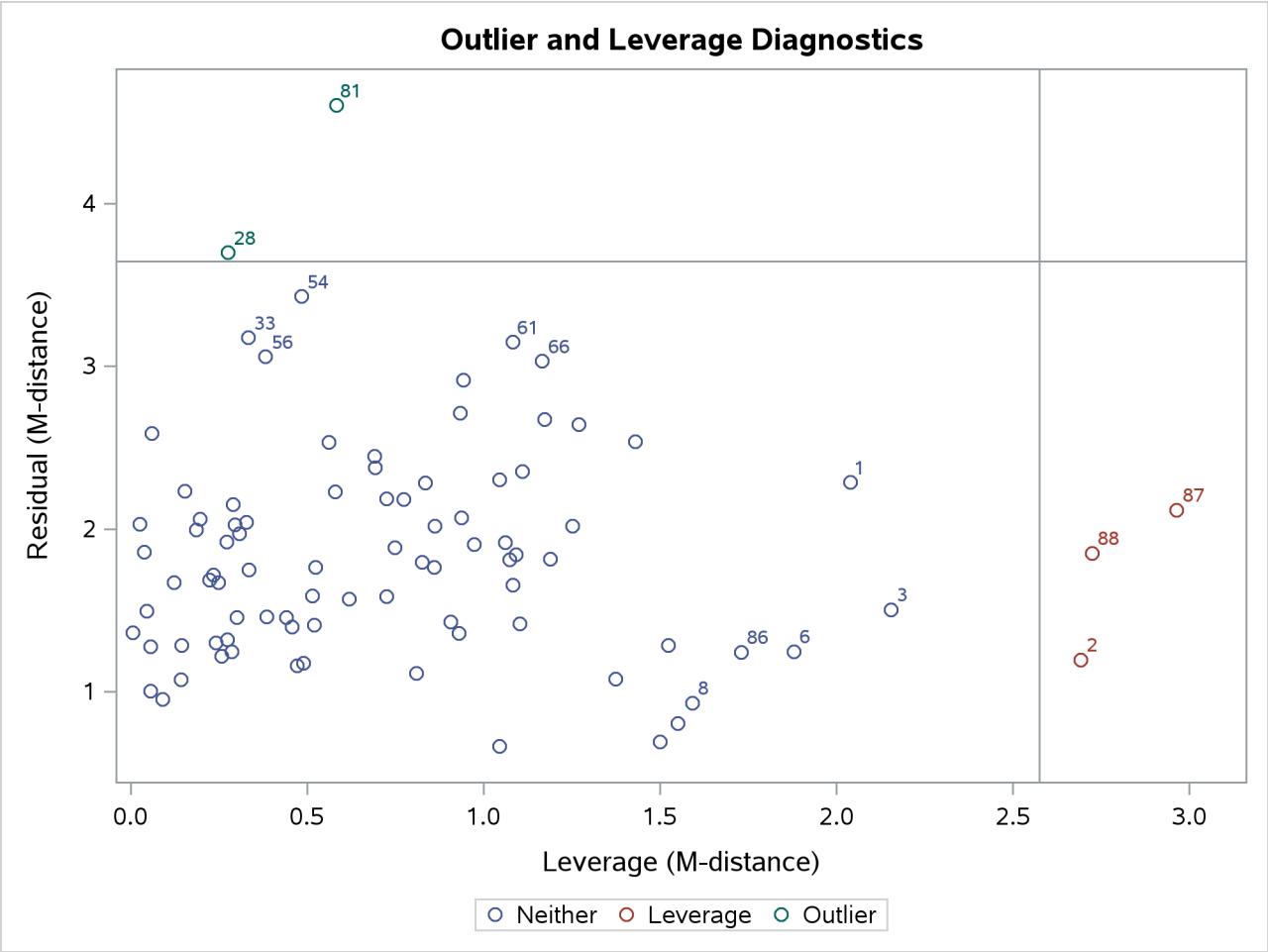

The upper and the lower regions are separated by a horizontal line that represents the criterion for outlier detection. This criterion is set by a certain α-level that has a probability meaning. Loosely speaking, if the model is true and the distribution of the residuals is multivariate normal, you would expect that about $\alpha \times 100\%$ of observations will have their residual M-distances above the criterion. The larger the α value, the more liberal the outlier detection criterion. This α-level is set to 0.01 by default. You can change this default by using the ALPHAOUT= option.

Notice that all residuals and the criterion are measured in terms of M-distances. Hence, these measures are always positive. Output 33.14.3 classifies observations 81 and 28 as outliers (in the residual dimension). In addition to these two outlying observations, PROC CALIS also labels the next five observations with the largest residual M-distances. Output 33.14.3 shows that observations 33, 54, 56, 61, and 66 are the closest to being labeled as outliers.

The right and left regions are separated by a vertical line that represents the criterion for leverage observations. Again, this criterion is set by a certain α-level that has a probability meaning. If the model is true and the distribution of the predictor variables (in this case, the factor variable fact1) is multivariate normal, you would expect that about $\alpha \times 100\%$ of observations will have their leverage M-distances above the criterion. The larger the α value, the more liberal the leverage observation detection criterion. This α-level is also set to 0.01 by default. You can change this default by using the ALPHALEV= option.

Output 33.14.3 classifies observations 2, 87, and 88 as leverage points or observations. In addition to these three observations, PROC CALIS also labels the next five observations with the largest leverage M-distances. Output 33.14.3 shows that observations 1, 3, 6, 8, and 86 are the closest to being labeled as leverage observations.

The lower left region in Output 33.14.3 contains observations that are neither outliers nor leverage observations. This region contains the majority of the observations. The upper right region in Output 33.14.3 contains observations that are classified as both outliers and leverage observations. For the current model, no observations are classified as such.

To supplement the graphical depiction of outliers and leverage observations, PROC CALIS shows two tables that contain more numerical information. Output 33.14.4 shows the seven observations that have the largest residual M-distances. The table reflects the outlier classification in Output 33.14.3 numerically. Two outliers are flagged in the table. However, none of the seven observations are leverage observations. Output 33.14.5 shows the eight observations that have the largest leverage M-distances. Again, this table reflects the classification of leverage observations in Output 33.14.3 numerically. Three leverage observations are flagged, but none of these observations are residual outliers at the same time.

Output 33.14.4 Largest Residual Outliers

Diagnostics of the 7 Largest Case-Level Residuals (alpha=0.01)			
		Diagnostics	
Case Number	Residual (M-Distance)	Outlier	Leverage
81	4.60511	*	
28	3.69953	*	
54	3.43068		
33	3.17659		
61	3.15104		
56	3.06094		
66	3.03262		

Output 33.14.5 Largest Leverage Observations

Diagnostics of the 8 Largest Case-Level Leverages (alpha=0.01)			
		Diagnostics	
Case Number	Leverage (M-Distance)	Leverage	Outlier
87	2.96438	*	
88	2.72441	*	
2	2.69293	*	
3	2.15519		
1	2.04014		
6	1.87894		
86	1.72950		
8	1.59242		

Output 33.14.6 shows the histogram and its kernel density function of the residual M-distances. Theoretically,

the density should look like a χ-distribution (that is, the distribution of the square root of the corresponding χ^2 variate). While this histogram provides an overall impression of the empirical distribution of the residual M-distances, it is not very useful in diagnosing the observations that do not conform to the theoretical distribution of the residual M-distances. To this end, Output 33.14.7 and Output 33.14.8 are more useful.

Output 33.14.6 Distribution of Case-Level Residuals

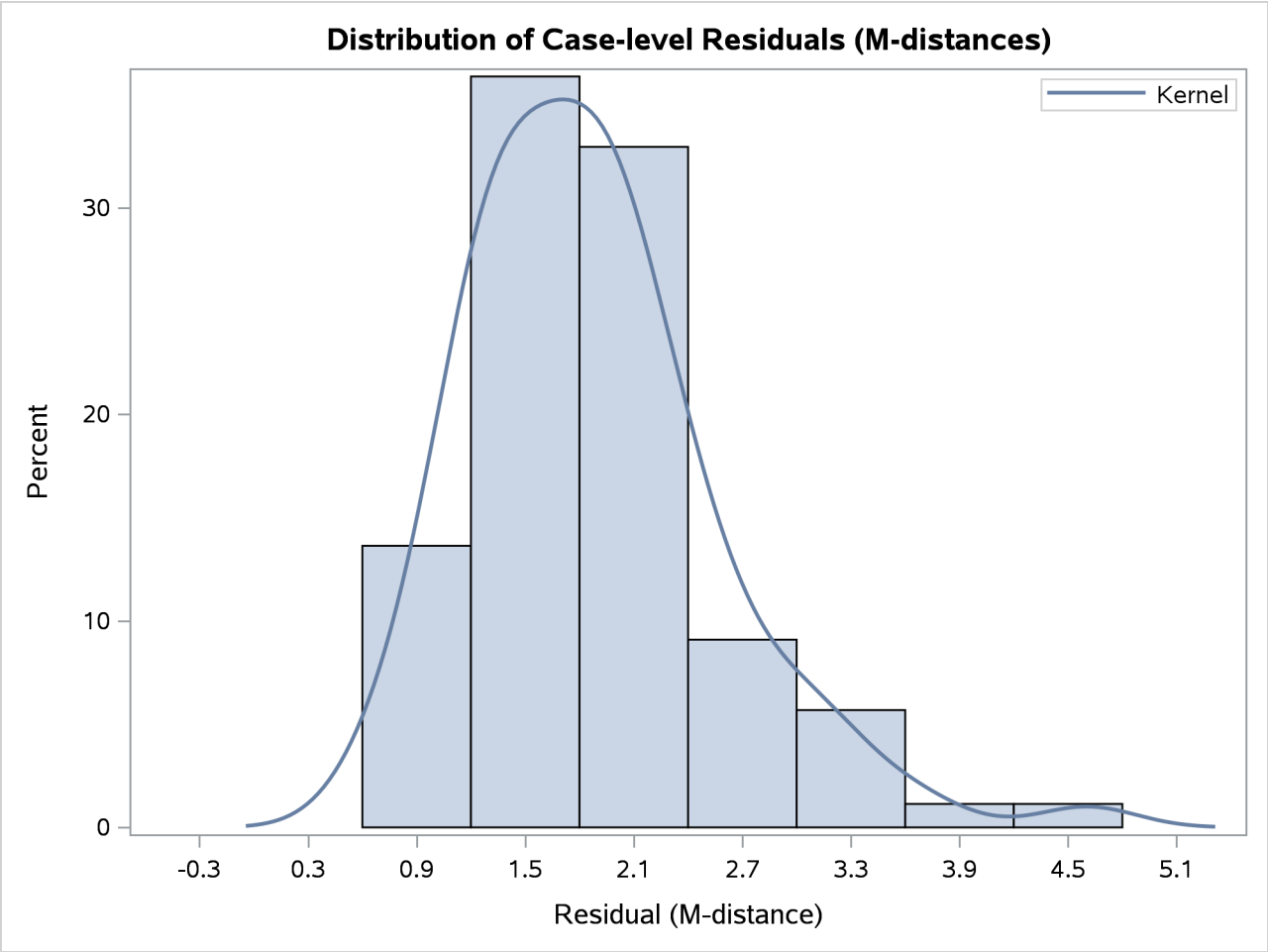

Output 33.14.7 shows the so-called Q-Q plot for the residual M-distances. The Y coordinates represent the observed quantiles, and the X coordinates represent the theoretical quantiles. Output 33.14.8 shows the so-called P-P plots for the residual M-distances. The Y coordinates represent the observed percentiles, and the X coordinates represent the theoretical percentiles. If the observed residual M-distances distribute exactly like the theoretical χ-distribution, all observations should lie on the straight lines with a slope of 1 in both plots. In Output 33.14.7 and Output 33.14.8, observations with the largest departures from the straight lines are labeled.

Output 33.14.7 Q-Q Plot of Residual M-Distances

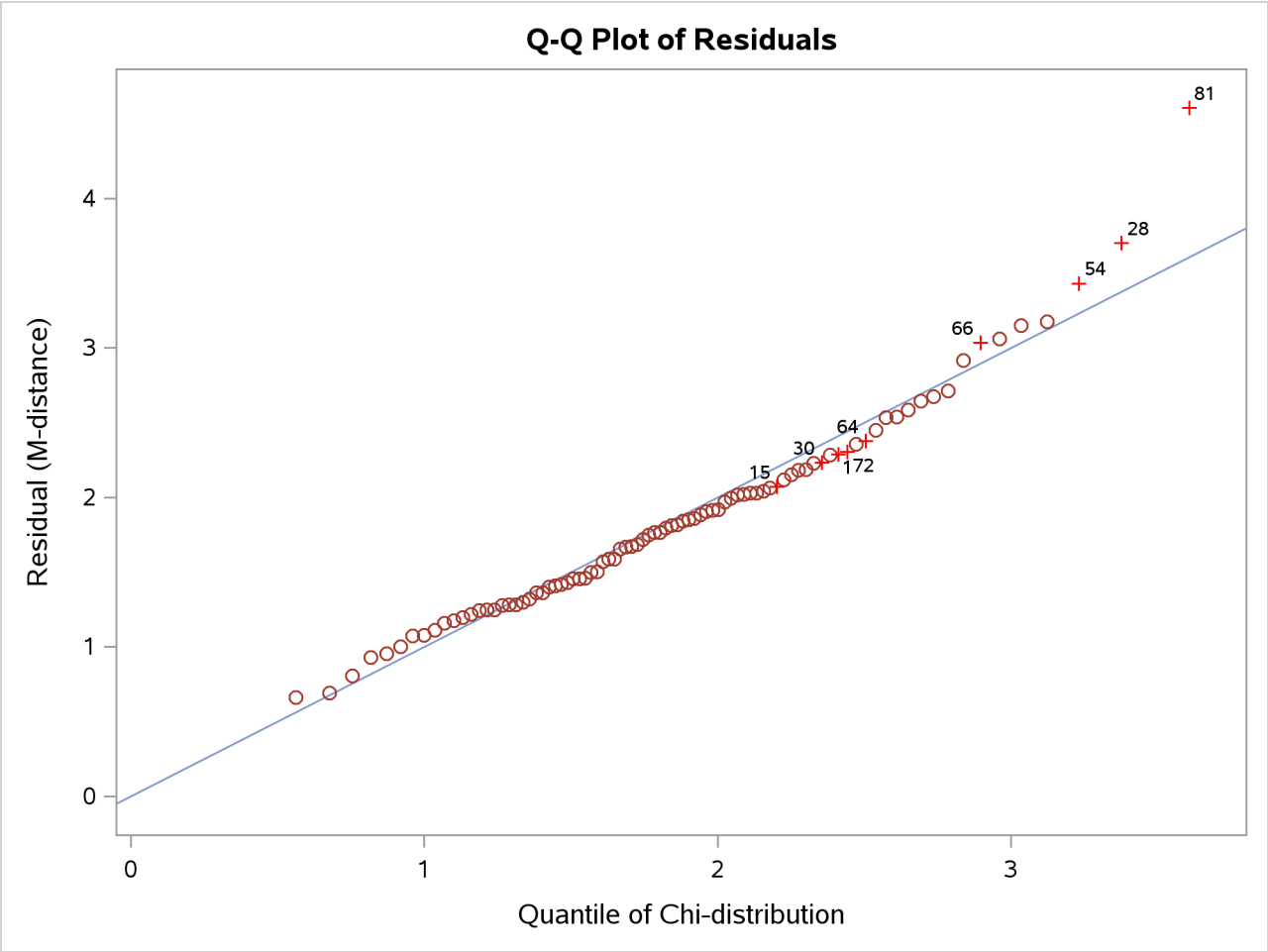

In terms of quantiles, the Q-Q plot in Output 33.14.7 shows that observations 81, 28, and 54 have the largest departures from the theoretical distribution for the residual M-distances. Notice that observations 81 and 28 were also identified as outliers previously.

Output 33.14.8 P-P Plot of Residual M-Distances

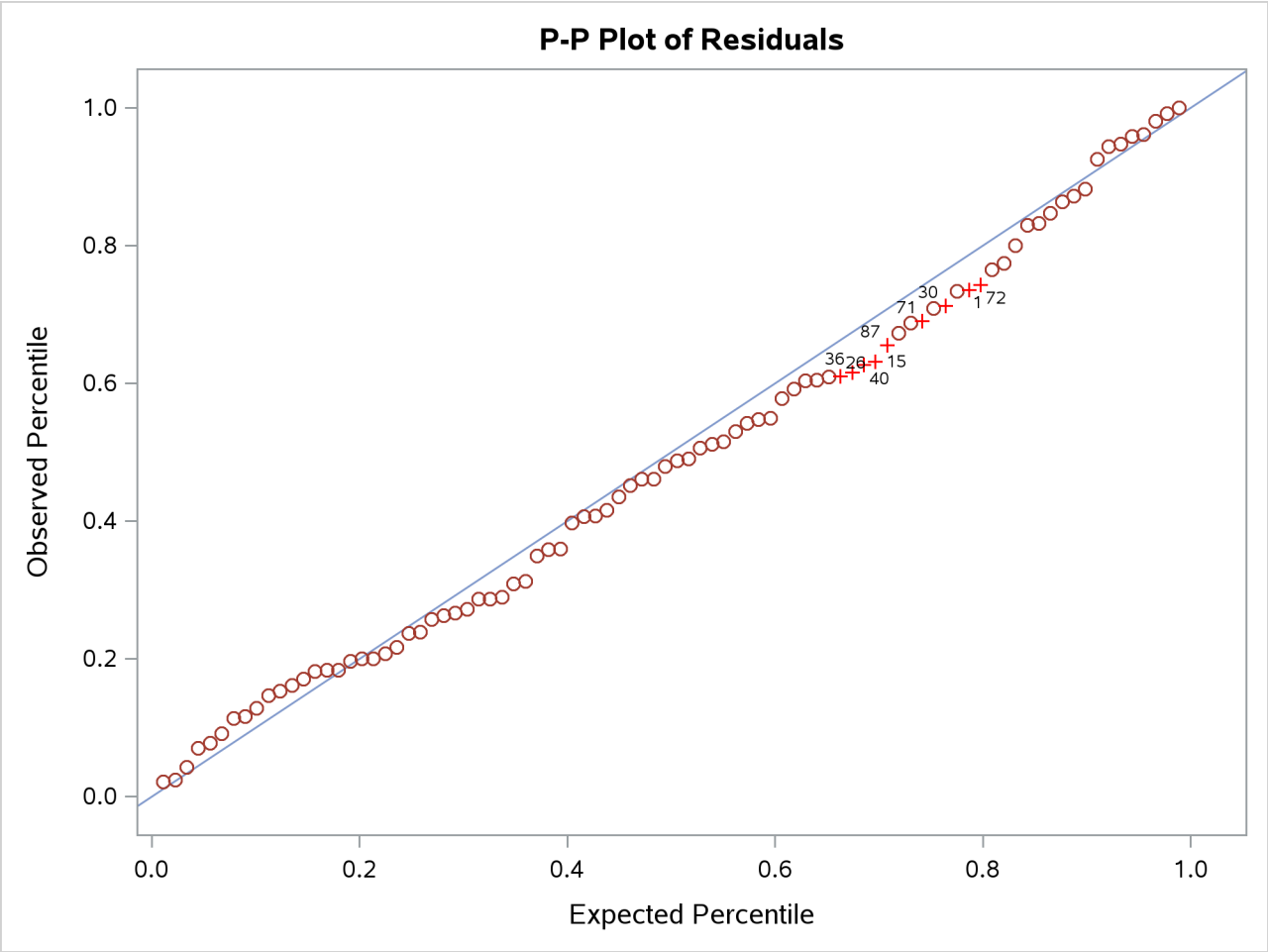

In terms of percentiles, the P-P plot in Output 33.14.8 shows that several observations in the middle of the distribution have the largest departures from the theoretical distribution. However, it is not clear from the plot which observations have the largest departures.

To supplement the Q-Q and the P-P plots with numerical information, PROC CALIS outputs the table in Output 33.14.9, which shows the observations with the largest departures from the theoretical residual M-distance distribution in terms of quantiles and percentiles. The observations in the table are arranged by their percentile regions. Approximately 10% of the observations with the largest departures in terms of quantile and in terms of percentile, respectively, are shown in this table. However, the final percentage of observations shown in the table might not add up to 20%, because some observations could have large departures in terms of both quantile and percentile. Output 33.14.9 shows that observations 1, 15, 30, and 72 are this kind of observation. However, none of these are residual outliers.

Output 33.14.9 Largest Departures from the Theoretical Residual Distribution

Largest Departures From the Theoretical Residual Distribution			
Percentile Region	**Case Number**	**Quantile Deviation**	**Percentile Deviation**
(.65,.90)	**26**		-0.05302
	36		-0.05836
	40		-0.05863
	15	-0.13087	-0.06509
	87		-0.05257
	71		-0.05145
	30	-0.11973	-0.05198
	1	-0.12344	-0.05065
	72	-0.13739	-0.05498
	64	-0.12550	
(.90,.95)	**66**	0.13803	
>.95	**54**	0.20039	
	28	0.32398	
	81	0.99843	

Output 33.14.10 shows the residual on fit plot for the variable algebra. Notice that unlike the preceding residual diagnostic plots where the M-distances (nonnegative) for residuals are plotted, the residuals (which can be positive or negative) themselves are plotted against the predicted values for algebra in the current residual on fit (or residual on predicted) plot. If the linear relationship as described by the one-factor model is reasonable, then the residuals should show no systematic trend with the predicted values. That is, the residuals should be randomly scattered around the plot in the range of the predicted values. The pattern of residuals in Output 33.14.10 looks sufficiently "random," and therefore it gives support to the prescribed linear relationship between the variable algebra and the factor fact1. PROC CALIS also outputs the residual on fit plots for other dependent observed variables. To conserve space, they are not shown here. You can also control the amount of graphical output by specifying the desirable set of dependent observed variables for the residual on fit plots. For example, the following requests the residual on fit plots for the variables mechanics and analysis only, even though there are five dependent observed variables in the model:

```
PLOTS=RESBYPRED(VAR=mechanics analysis)
```

See the PLOTS= option for details.

Output 33.14.10 Residual on Fit Plot for the Variable Algebra

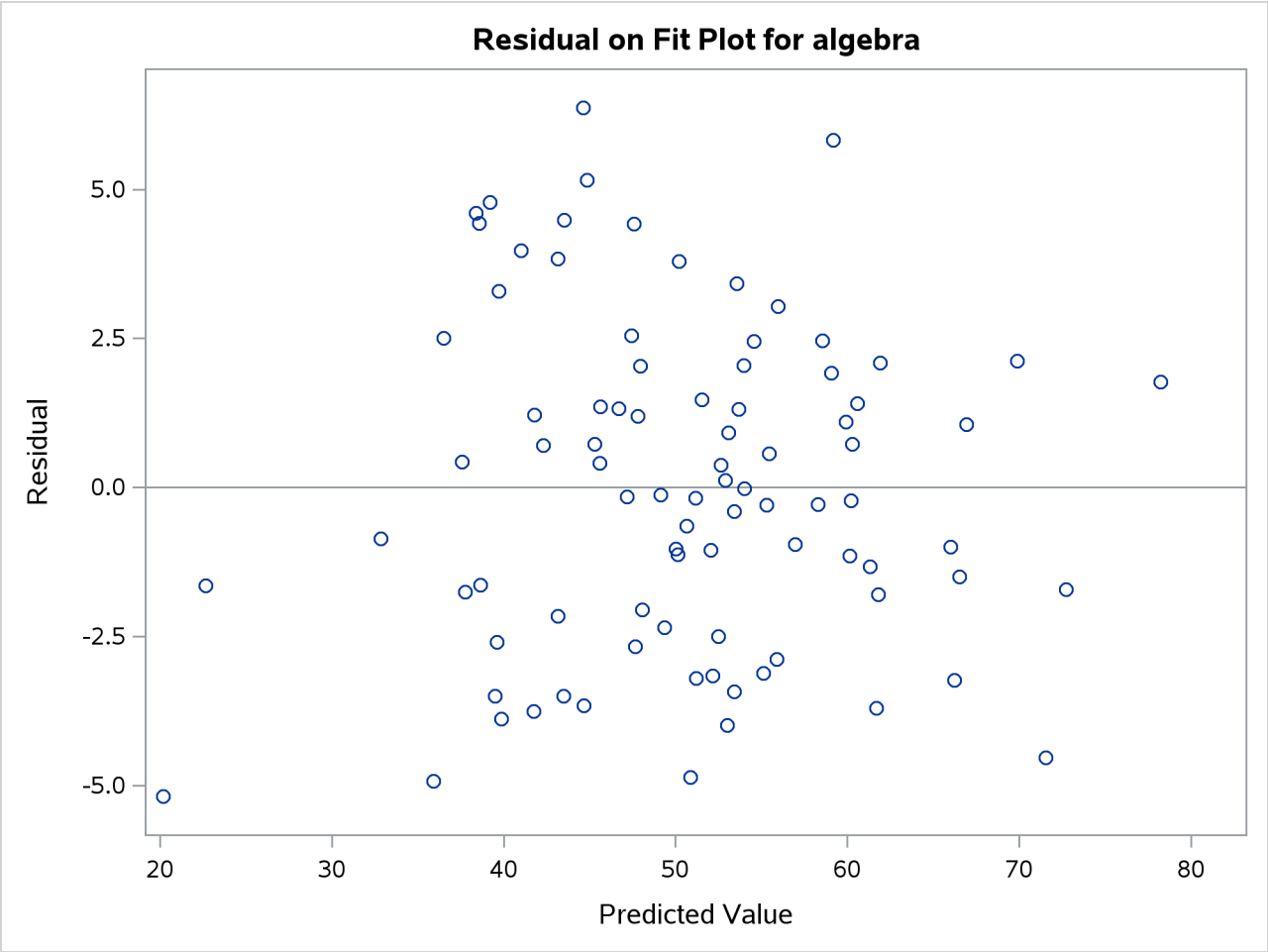

Direct Robust Estimation

Usually, the outlier detection process in residual diagnostics implies some treatments after their detections. For example, in the preceding analysis, observations 28 and 81 were detected as outliers. You would naturally think that the next step is to do an additional model fitting with these two outlying observations removed. While this thinking is intuitively appealing, the outlier issues are usually more complicated than can be resolved by such a "pick-and-remove" strategy. One issue is the so-called masking effect of outliers. In the preceding analysis, outliers are detected based on a criterion that is a function of the chosen criterion level (for example, the ALPHAOUT= value set by the researcher) and the parameter estimates. However, because the outliers have contributed to the estimation of the parameters, you might wonder whether the outlier detection process has been contaminated by the outliers themselves. If such a masking effect is present and an additional analysis is done after employing the pick-and-remove strategy for the outliers, you would be likely to discover more outliers. As a result, you might wonder whether the outlier removal process should be carried out indefinitely.

Robust estimation provides an alternative way to resolve the issue of masking effect. In robust estimation, outlying observations are downweighted simultaneously with the estimation. Iterative steps are used to reweight the observations according to the updated parameter estimates. Hence, with robust estimation you

do not need to make a decision about removing outliers because they are already downweighted during the estimation. The issue of masking effect is avoided.

PROC CALIS provides two major methods of robust estimation—direct and two-stage. This section illustrates the direct robust estimation method, in which the model is estimated simultaneously with the (re)weighting of the observations. The next section illustrates the two-stage estimation method, in which the robust mean and covariance matrices obtained in the first stage are analyzed by the maximum likelihood (ML) estimation in the second stage. Both methods are based on the methodology proposed by Yuan and Zhong (2008) and Yuan and Hayashi (2010).

The following statements request direct robust estimation of the model for the mardia data set:

```
ods graphics on;

proc calis data=mardia residual robust plots=caseresid pcorr;
   path
      fact1 ===>  mechanics vector algebra analysis statistics = 1. ;
run;

ods graphics off;
```

The ROBUST option invokes the direct robust estimation method. The direct robust method reweights the observations according to the magnitudes of the residuals during iterative steps. For this reason, you can also specify ROBUST=RES to signify direct robust estimation with residual (M-distance) weighting. Following Yuan and Hayashi (2010), there are two variants of the direct robust estimation, One treats the disturbances (that is, error terms for endogenous latent factors) as factors to estimate (see the ROBUST=RES(F) option), and the other treats the disturbances as true error terms (see the ROBUST=RES(E) option). See the ROBUST= option and the section "Robust Estimation" on page 1812 for details. These variants would not make any difference for the current one-factor model estimation, and the ROBUST option that is used in the specification is the same as ROBUST=RES(E).

You also request case-level residual analysis by using the RESIDUAL option, even though the intention now is to identify rather than to remove the outliers.

Output 33.14.11 shows the fit summary of the direct robust estimation. The model fit chi-square is 6.0031 ($df = 5$, $p = 0.3059$). Again, you fail to reject the one-factor model for the data. Unlike the regular ML estimation in the preceding analysis, both SRMR (0.0321) and RMSEA (0.0480) show good model fit. This result can be attributed to the downweighting of the outlying observations in the direct robust estimation. Output 33.14.12 shows that all loading estimates are statistically significant.

Output 33.14.11 Model Fit of Mardia Data with Direct Robust Estimation

Fit Summary	
Chi-Square	6.0031
Chi-Square DF	5
Pr > Chi-Square	0.3059
Standardized RMR (SRMR)	0.0370
RMSEA Estimate	0.0480

NOTE: Saturated mean structure parameters are excluded from the computations of fit indices.

Output 33.14.12 Estimates of Loadings with Direct Robust Estimation

PATH List					
Path	Parameter	Estimate	Standard Error	t Value	Pr > |t|
fact1 ===> mechanics		1.00000			
fact1 ===> vector	_Parm1	0.83279	0.15680	5.3111	<.0001
fact1 ===> algebra	_Parm2	0.92854	0.14763	6.2895	<.0001
fact1 ===> analysis	_Parm3	1.08665	0.18855	5.7632	<.0001
fact1 ===> statistics	_Parm4	1.18799	0.21487	5.5289	<.0001

Output 33.14.13 shows the outlier and leverage classifications by the direct robust estimation. This plot has a similar pattern to the one obtained from the preceding regular ML estimation (see Output 33.14.3). Both plots show that observations 28 and 81 are the only outliers. Thus, the results from the direct robust estimation eliminate the concern of the masking effect.

Output 33.14.13 Outliers and Leverage Observations with Direct Robust Estimation

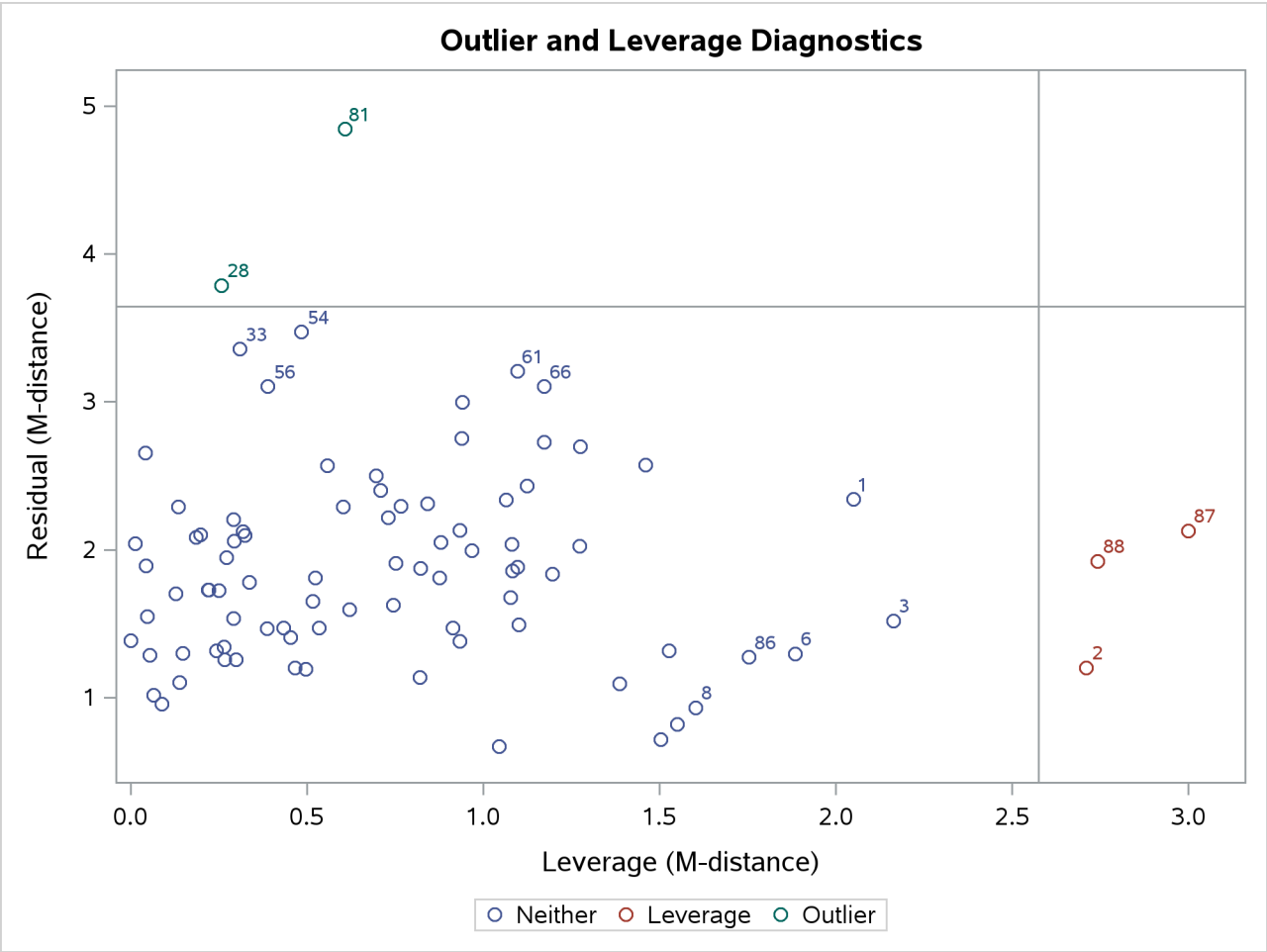

As a by-product of the direct robust estimation, robust (unstructured) covariance and mean matrices are computed with the PCORR option. Output 33.14.14 shows the robust covariance and mean matrices obtained for the current direct robust estimation for the mardia data set. These matrices are essentially computed by

the usual formulas for covariance and mean matrices, but with weights obtained from the final estimates of the direct robust estimation of the model. For the current analysis, outlying observations (28 and 81) receive weights of less than 1 whereas all other observations receive weights of 1 in the computation.

Output 33.14.14 Robust Covariance and Mean Matrices with Direct Robust Estimation

Robust Covariance Matrix					
	mechanics	vector	algebra	analysis	statistics
mechanics	299.05121	118.73107	104.04139	111.35028	122.12239
vector	118.73107	162.60829	86.88667	97.18075	102.10393
algebra	104.04139	86.88667	114.69031	113.43289	124.09414
analysis	111.35028	97.18075	113.43289	220.34769	156.62425
statistics	122.12239	102.10393	124.09414	156.62425	296.42019

Determinant	33723890038	**Ln**	24.241472

Robust Means				
mechanics	vector	algebra	analysis	statistics
39.11015	50.80549	50.61153	46.70196	42.21135

Two-Stage Robust Estimation

In addition to the direct robust estimation method, PROC CALIS implements another robust estimation method for structural equation models. The two-stage robust estimation method estimates the robust means and covariance matrices in the first stage. It then fits the model to the robust mean and covariance matrix by using the ML method in the second stage. During the first stage, the robust method downweights the outlying observations in all variable dimensions. The robust properties are then propagated to the second stage of estimation.

In PROC CALIS, you can use the ROBUST=SAT option to invoke the two-stage robust estimation method, as shown in the following statements for fitting the one-factor model to the mardia data set:

```
proc calis data=mardia robust=sat pcorr;
   path
      fact1 ===>  mechanics vector algebra analysis statistics = 1. ;
run;
```

With the PCORR option, the (unstructured) robust mean and covariance matrices are displayed in Output 33.14.15. Although the robust covariance and mean matrices obtained here are numerically similar to those obtained from the direct robust estimation (see Output 33.14.14), they are obtained with totally different methods. The robust mean and covariance matrices in the direct robust estimation are the by-products of the model estimation, while the robust mean and covariance matrices in the two-stage robust estimation are the initial inputs for model estimation.

Output 33.14.15 Robust Covariance and Mean Matrices with Two-Stage Robust Estimation

Robust Covariance Matrix					
	mechanics	**vector**	**algebra**	**analysis**	**statistics**
mechanics	298.83513	119.28292	99.65134	105.47098	116.68242
vector	119.28292	165.11323	84.25310	94.28855	98.70260
algebra	99.65134	84.25310	110.93012	110.46759	121.18480
analysis	105.47098	94.28855	110.46759	219.56424	155.24377
statistics	116.68242	98.70260	121.18480	155.24377	298.80146

Determinant	36001000868	**Ln**	24.306813

Robust Means				
mechanics	**vector**	**algebra**	**analysis**	**statistics**
39.10571	50.74232	50.65677	46.73882	42.33179

Output 33.14.16 shows the fit summary of the two-stage robust estimation. The model fit chi-square is 7.2941 ($df = 5$, $p = 0.1997$). As in direct robust estimation, you fail to reject the one-factor model for the data. Unlike the case with direct robust estimation, only SRMR (0.0366) shows a good model fit. RMSEA is 0.0726 and is not very close to a good model fit.

Output 33.14.16 Model Fit of Mardia Data with Two-Stage Robust Estimation

Fit Summary	
Chi-Square	7.2941
Chi-Square DF	5
Pr > Chi-Square	0.1997
Standardized RMR (SRMR)	0.0422
RMSEA Estimate	0.0726

NOTE: Saturated mean structure parameters are excluded from the computations of fit indices.

Output 33.14.16 shows the estimates of loadings with the two-stage robust estimation. All the estimates are statistically significant. All the numerical results are similar to that of the direct robust estimation in Output 33.14.12.

Output 33.14.17 Estimates of Loadings with Two-Stage Robust Estimation

PATH List						
Path			**Parameter**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
fact1	**===>**	**mechanics**		1.00000			
fact1	**===>**	**vector**	_Parm1	0.84322	0.16538	5.0986	<.0001
fact1	**===>**	**algebra**	_Parm2	0.93683	0.15485	6.0500	<.0001
fact1	**===>**	**analysis**	_Parm3	1.10397	0.19849	5.5617	<.0001
fact1	**===>**	**statistics**	_Parm4	1.21080	0.22686	5.3371	<.0001

The outlier and leverage observation detection in two-stage robust estimation is essentially the same method as that in direct robust estimation. Therefore, the corresponding results for the two-stage robust estimation are not shown here. See the section "Residual Diagnostics with Robust Estimation" on page 1840 for details

about case-level residual analysis with robust estimation.

Example 33.15: The Full Information Maximum Likelihood Method

(View the complete code for this example (calex113.sas) in the example repository.)

This example shows how you can fully utilize all available information from the data when there is a high proportion of observations with random missing value. You use the full information maximum likelihood method for model estimation.

In Example 33.12, 32 students take six tests. These six tests are indicator measures of two ability factors: verbal and math. You conduct a confirmatory factor analysis in Example 33.12 based on a data set without any missing values. The path diagram for the confirmatory factor model is shown the following:

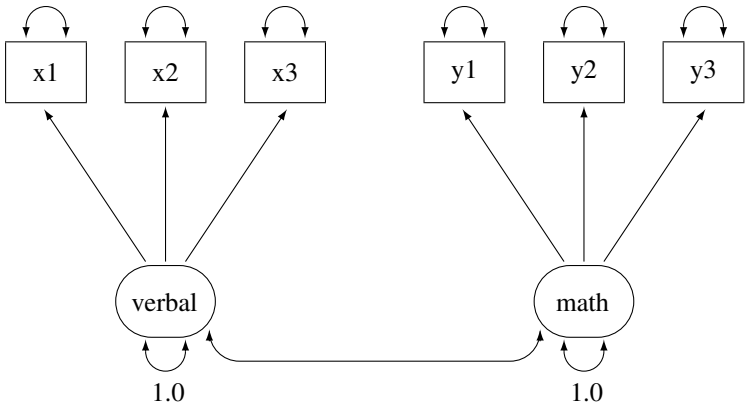

Suppose now due to sickness or unexpected events, some students cannot take part in one of these tests. Now, the data test contains missing values at various locations, as indicated by the following DATA step:

```
data missing;
   input x1 x2 x3 y1 y2 y3;
   datalines;
 23   .  16  15  14  16
 29  26  23  22  18  19
 14  21   .  15  16  18
 20  18  17  18  21  19
 25  26  22   .  21  26
 26  19  15  16  17  17
  .  17  19   4   6   7
 12  17  18  14  16   .
 25  19  22  22  20  20
  7  12  15  10  11   8
 29  24   .  14  13  16
 28  24  29  19  19  21
 12   9  10  18  19   .
 11   .  12  15  16  16
 20  14  15  24  23  16
```

```
 26  25   .  24  23  24
 20  16  19  22  21  20
 14   .  15  17  19  23
 14  20  13  24   .   .
 29  24  24  21  20  18
 26   .  26  28  26  23
 20  23  24  22  23  22
 23  24  20  23  22  18
 14   .  17   .  16  14
 28  34  27  25  21  21
 17  12  10  14  12  16
  .   1  13  14  15  14
 22  19  19  13  11  14
 18  21   .  15  18  19
 12  12  10  13  13  16
 22  14  20  20  18  19
 29  21  22  13  17   .
;
```

This data set is similar to the scores data set used in Example 33.12, except that some values are replaced at random with missing values. You can still fit the same confirmatory factor analysis model described in Example 33.12 to this data set by the default maximum likelihood (ML) method, as shown in the following statement:

```
proc calis data=missing;
   factor
      verbal ===> x1-x3,
      math   ===> y1-y3;
   pvar
      verbal = 1.,
      math   = 1.;
run;
```

The data set, the number of observations, the model type, and analysis type are shown in the first table of Output 33.15.1. Although PROC CALIS reads all 32 records in the data set, only 16 of these records are used. The remaining 16 records contain at least one missing value in the tests. They are discarded from the analysis. Therefore, the maximum likelihood method only uses those 16 observations without missing values.

Output 33.15.1 Modeling Information of the CFA Model: Missing Data

Confirmatory Factor Model With \Dataset{Missing} Data: ML
FACTOR Model Specification

The CALIS Procedure
Covariance Structure Analysis: Model and Initial Values

Modeling Information	
Maximum Likelihood Estimation	
Data Set	WORK.MISSING
N Records Read	32
N Records Used	16
N Obs	16
Model Type	FACTOR
Analysis	Covariances

Output 33.15.2 shows the parameter estimates.

Output 33.15.2 Parameter Estimates of the CFA Model: Missing Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
x1	5.1110 1.3110 3.8984 <.0001 [_Parm1]	0
x2	5.6261 1.2561 4.4790 <.0001 [_Parm2]	0
x3	4.8739 1.1410 4.2717 <.0001 [_Parm3]	0
y1	0	4.4529 0.8530 5.2205 <.0001 [_Parm4]
y2	0	3.8562 0.8303 4.6444 <.0001 [_Parm5]
y3	0	2.6338 0.7416 3.5513 0.000383 [_Parm6]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
verbal	1.0000	0.7050 0.1464 4.8165 <.0001 [_Add1]
math	0.7050 0.1464 4.8165 <.0001 [_Add1]	1.0000

Output 33.15.2 *continued*

Error Variances					
Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add2	11.27773	5.19739	2.1699	0.0300
x2	_Add3	6.33003	4.25356	1.4882	0.1367
x3	_Add4	6.47402	3.61040	1.7932	0.0729
y1	_Add5	0.57143	1.51781	0.3765	0.7066
y2	_Add6	2.57992	1.47618	1.7477	0.0805
y3	_Add7	4.59651	1.77777	2.5855	0.0097

Most of the factor loading estimates shown in Output 33.15.2 are similar to those estimated from the data set without missing values, as shown in Output 33.12.4. The loading estimate of y3 on the math factor shows the largest discrepancy. With only half of the data used in the current estimation, this loading estimate is 2.6338 in the current analysis, while it is 3.7596 if no data were missing, as shown in Output 33.12.4. Another obvious difference between the two sets of results is that the standard error estimates for the loadings are consistently larger in the current analysis than in the analysis in Example 33.12 where there are no missing data. This is expected because you have only half of the data set available in the current analysis.

Similarly, the estimates for the factor covariance and error variances are mostly similar to those in the analysis with complete data, but the standard error estimates in the current analysis are consistently higher.

The maximum likelihood method, as implemented in PROC CALIS, deletes all observations with at least one missing value in the estimation. In a sense, the partially available information of these deleted observations is wasted. This greatly reduces the efficiency of the estimation, which results in higher standard error estimates.

To fully utilize all available information from the data set with the presence of missing values, you can use the full information maximum likelihood (FIML) method in PROC CALIS, as shown in the following statements:

```
proc calis method=fiml data=missing;
   factor
      verbal ===> x1-x3,
      math   ===> y1-y3;
   pvar
      verbal = 1.,
      math   = 1.;
run;
```

In the PROC CALIS statement, you use METHOD=FIML to request the full information maximum likelihood method. Instead of deleting observations with missing values, the full information maximum likelihood method uses all available information in all observations. Output 33.15.3 shows some modeling information of the FIML estimation of the confirmatory factor model on the missing data.

Output 33.15.3 Modeling Information of the CFA Model with FIML: Missing Data

Confirmatory Factor Model With Missing Data: FIML
FACTOR Model Specification

The CALIS Procedure
Mean and Covariance Structures: Model and Initial Values

Modeling Information	
Full Information Maximum Likelihood Estimation	
Data Set	WORK.MISSING
N Records Read	32
N Complete Records	16
N Incomplete Records	16
N Complete Obs	16
N Incomplete Obs	16
Model Type	FACTOR
Analysis	Means and Covariances

PROC CALIS shows you that the number of complete observations is 16 and the number of incomplete observations is 16 in the data set. All these observations are included in the estimation. The analysis type is 'Means and Covariances' because with full information maximum likelihood, the sample means have to be analyzed during the estimation.

For the full information maximum likelihood estimation, PROC CALIS outputs several tables to summarize the missing data patterns and statistics. Output 33.15.4 shows the proportions of data that are present for the variables, individually or jointly by pairs.

Output 33.15.4 Proportions of Data Present for the Variables: Missing Data

Proportions of Data Present for Means (Diagonal) and Covariances (Off-Diagonal)						
	x1	**x2**	**x3**	**y1**	**y2**	**y3**
x1	0.9375					
x2	0.7813	0.8438				
x3	0.8125	0.7188	0.8750			
y1	0.8750	0.8125	0.8125	0.9375		
y2	0.9063	0.8125	0.8438	0.9063	0.9688	
y3	0.8125	0.7188	0.7500	0.8125	0.8750	0.8750

Average Proportion Coverage of Means	0.906250
Average Proportion Coverage of Covariances	0.816667

The diagonal elements of the table in Output 33.15.4 show the proportions of data coverage by each of the variables. The off-diagonal elements shows the proportions of joint data coverage by all possible pairs of variables. For example, the first diagonal element of the table shows that about 94% of the observations have x1 values that are not missing. This percentage value is referred to as the proportion coverage for x1 or the proportion coverage for computing the means of x1. The off-diagonal element for x1 and x2 shows that about 78% of the observations have nonmissing values for both their x1 and x2 values. This percentage value is referred to as the joint proportion coverage of x1 and x2 or the proportion coverage for computing

the covariance between x1 and x2. The larger the coverage proportions this table shows, the more relative information the data contain for estimating the corresponding moments.

To summarize the proportion coverage, Output 33.15.4 shows that on average about 91% of the data are nonmissing for computing the means, and about 82% of the data are nonmissing for computing the covariances.

Output 33.15.5 shows the lowest coverage proportions of the means and the covariances.

Output 33.15.5 Ranking the Lowest Coverage Proportions: Missing Data

Rank Order of the 3 Smallest Variable (Mean) Coverages	
Variable	**Coverage**
x2	0.8438
x3	0.8750
y3	0.8750

Rank Order of the 7 Smallest Covariance Coverages		
Var1	**Var2**	**Coverage**
x3	**x2**	0.7188
y3	**x2**	0.7188
y3	**x3**	0.7500
x2	**x1**	0.7813
x3	**x1**	0.8125
y1	**x2**	0.8125
y1	**x3**	0.8125

The first table of Output 33.15.5 shows that x2 has the lowest proportion coverage at about 84%, and x3 and y3 are the next at about 88%. The second table of Output 33.15.5 shows that the joint proportion coverage by the x3–x2 pair and the y3–x2 pair are the lowest at about 72%, followed by the y3–x3 pair at 75%. These two tables are useful to diagnose which variables most lack the information for estimation. For this data set, these tables show that estimation related to the moments of x2, x3, and y3 suffers the missing data problem the most. However, because the worst proportion coverage is still higher than 70%, the missingness problem does not seem to be very serious based on percentage.

In Output 33.15.6, PROC CALIS outputs two tables that show an overall picture of the missing patterns in the data set.

Output 33.15.6 The Most Frequent Missing Patterns and Their Mean Profiles: Missing Data

Rank Order of the 5 Most Frequent Missing Patterns Total Number of Distinct Patterns with Missing Values = 7					
	Pattern	NVar Miss	Freq	Proportion	Cumulative
1	x.xxxx	1	4	0.1250	0.1250
2	xx.xxx	1	4	0.1250	0.2500
3	xxxxx.	1	3	0.0938	0.3438
4	.xxxxx	1	2	0.0625	0.4063
5	xxxx..	2	1	0.0313	0.4375
NOTE: Nonmissing Pattern Proportion = 0.5000 (N=16)					

Means of the Nonmissing and the Most Frequent Missing Patterns						
		Missing Pattern				
Variable	Nonmissing (N=16)	1 (N=4)	2 (N=4)	3 (N=3)	4 (N=2)	5 (N=1)
x1	21.75000	18.50000	21.75000	17.66667	.	14.00000
x2	19.37500	.	22.75000	15.66667	9.00000	20.00000
x3	19.31250	17.25000	.	16.66667	16.00000	13.00000
y1	19.00000	18.75000	17.00000	15.00000	9.00000	24.00000
y2	18.12500	18.75000	17.50000	17.33333	10.50000	.
y3	17.75000	19.50000	19.25000	.	10.50000	.

The first table of Output 33.15.6 shows that "x.xxxx" and "xx.xxx" are the two most frequent missing patterns in the data set. Each has a frequency of 4. An "x" in the missing pattern denotes a nonmissing value, while a "." denotes a missing value. Hence, the first pattern has all missing values for the second variable, and the second pattern has all missing values for the third variable. Each of these two missing patterns accounts for 12.5% of the total observations. Together, the five missing patterns shown in Output 33.15.6 account for about 43.8% of the total observations. The note after this table shows that 50% of the total observations do not have any missing values.

To determine exactly which variables are missing in the missing patterns, it is useful to consult the second table in Output 33.15.6. In this table, the variable means of the most frequent missing patterns are shown, together with the variable means of the nonmissing pattern for comparisons. Missing means in this table show that the corresponding variables are not present in the missing patterns. For example, the column labeled "Nonmissing" is for the group of 16 observations that do not have any missing values. Each of the variable means is computed based on 16 observations. The next column labeled "1" is the first missing pattern that has four observations. The variable mean for x2 is missing for this missing pattern group, while each of the other variable means is computed based on four observations. Comparing these means with those in the nonmissing group, it shows that the means for x1, x3, and y1 in the first missing pattern are smaller than those in the nonmissing group, while the means for y2 and y3 are greater. This comparison does not seem to suggest any systematic bias in the means of the first missing pattern group.

However, the nonmissing means in the third missing pattern (the column labeled "3" do show a consistent downward bias, as compared with the means in the nonmissing group. This might mean that respondents with low scores in x1–x3, y1, and y2 tend not to respond to y3 for some reason. Similarly, the fourth missing pattern shows a consistent downward bias in x2, x3, and y1–y3. Whether these patterns suggest a systematic (or nonrandom) pattern of missingness must be judged in the substantive context. Nonetheless, the numerical results if Output 33.15.6 provide some insight on this matter.

The tables shown in Output 33.15.6 do not show all the missing patterns. In general, PROC CALIS shows only the most frequent or dominant missing patterns so that the output results are more focused. By default, if the total number of missing patterns in a data set is below six, then PROC CALIS shows all the missing patterns. If the total number of missing patterns is at least six, PROC CALIS shows up to 10 missing patterns provided that each of these missing patterns accounts for at least 5% of the total observations. The 10 missing patterns is the default maximum number of missing patterns to show, and the 5% is the default proportion threshold for a missing pattern to display. You can override the default maximum number of missing patterns by the MAXMISSPAT= option and the proportion threshold by the TMISSPAT= option.

Output 33.15.7 shows the parameter estimates by the FIML estimation.

Output 33.15.7 Parameter Estimates of the CFA Model with FIML: Missing Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	verbal	**math**
x1	5.5003 1.0345 5.3166 <.0001 [_Parm1]	0
x2	5.7134 1.0153 5.6273 <.0001 [_Parm2]	0
x3	4.4417 0.7596 5.8473 <.0001 [_Parm3]	0
y1	0	4.9277 0.6860 7.1836 <.0001 [_Parm4]
y2	0	4.1215 0.5743 7.1766 <.0001 [_Parm5]
y3	0	3.3834 0.6038 5.6031 <.0001 [_Parm6]

Output 33.15.7 *continued*

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value	verbal	math
verbal	1.0000	0.5014 0.1515 3.3099 0.000933 [_Add01]
math	0.5014 0.1515 3.3099 0.000933 [_Add01]	1.0000

Error Variances

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add08	12.72770	4.77932	2.6631	0.0077
x2	_Add09	9.35994	4.95096	1.8905	0.0587
x3	_Add10	5.67393	2.79751	2.0282	0.0425
y1	_Add11	1.86768	1.47469	1.2665	0.2053
y2	_Add12	1.49942	1.06245	1.4113	0.1582
y3	_Add13	5.24973	1.56780	3.3485	0.0008

First, you can compare the current FIML results with the results in Example 33.12, where maximum likelihood method is used with the complete data set. Overall, the estimates of loadings, factor covariance, and error variances are similar in the two analyses. Next, you compare the current FIML results with the results in Output 33.15.2, where the default ML method is applied to the same data set with missing values. Except for the standard error estimate of the factor covariance, which are very similar with ML and FIML, the standard error estimates with FIML are consistently smaller than those with ML in Output 33.15.2. This means that with FIML, you improve the estimation efficiency by including the partial information in those observations with missing values.

When you have a data set with no missing values, the ML and FIML methods, as implemented in PROC CALIS, are theoretically the same. Both are equally efficient and produce similar estimates (see Example 33.16). FIML and ML are the same estimation technique that maximizes the likelihood function under the multivariate normal distribution. However, in PROC CALIS, the distinction between of ML and FIML concerns different treatments of the missing values. With METHOD=ML, all observations with one or more missing values are discarded from the analysis. With METHOD=FIML, all observations with at least one nonmissing value are included in the analysis.

Example 33.16: Comparing the ML and FIML Estimation

(View the complete code for this example (calex114.sas) in the example repository.)

This example uses the complete data set from Example 33.12 to illustrate how the maximum likelihood (ML) and full information maximum likelihood (FIML) methods are theoretically equivalent when you apply them to data set without missing values. In Example 33.15, you apply a confirmatory factor model to a data set with missing values. You find that with METHOD=FIML, you can get more stable estimates than with METHOD=ML (which is the default estimation method). Near the end of Example 33.15, you learn that ML and FIML are theoretically equivalent estimation methods when you apply them to data sets *without* missing values.

However, the ML and FIML methods have two major computational differences in their implementations in PROC CALIS. First, with METHOD=FIML the first-order properties (that is, the means of the variables) of the data are automatically included in the analysis. However, by default you analyze only the second-order properties (that is, the covariances of the variables) with METHOD=ML. Second, the biased sample covariance formula (with N as the variance divisor) is used with METHOD=FIML, while the unbiased sample covariance formula (with DF=N – 1 as the variance divisor) is used with METHOD=ML. See the section "Relationships among Estimation Criteria" on page 1816 for more details about the similarities and differences between the ML and FIML methods.

If you take care of these two differences between ML and FIML in PROC CALIS, you can obtain exactly the same results with these two methods when you apply them to data sets without missing values.

For example, with the complete data set scores from Example 33.12, you specify the FIML estimation in the following statements:

```
proc calis method=fiml data=scores;
   factor
      verbal ===> x1-x3,
      math   ===> y1-y3;
   pvar
      verbal = 1.,
      math   = 1.;
run;
```

An equivalent specification with the ML method is shown in the following statements:

```
proc calis method=ml meanstr vardef=n data=scores;
   factor
      verbal ===> x1-x3,
      math   ===> y1-y3;
   pvar
      verbal = 1.,
      math   = 1.;
run;
```

In the PROC CALIS statement, you specify two options to make the ML estimation exactly equivalent to the FIML estimation in PROC CALIS. First, the MEANSTR option requests the first-order properties (the mean structures) to be analyzed with the covariance structures. Second, the VARDEF=N option defines the variance divisor to N, instead of the default DF, which is the same as N–1. These two options make the ML estimation equivalent to the FIML estimation.

Output 33.16.1 and Output 33.16.2 show some fit summary statistics under the FIML and ML methods, respectively.

Output 33.16.1 Model Fitting by the FIML Method: Scores Data

Fit Summary	
Fit Function	31.7837
Chi-Square	10.1215
Chi-Square DF	8
Pr > Chi-Square	0.2566
Standardized RMR (SRMR)	0.0571
RMSEA Estimate	0.0910
Bentler Comparative Fit Index	0.9872
NOTE: Saturated mean structure parameters are excluded from the computations of fit indices.	

Output 33.16.2 Model Fitting by the ML Method: Scores Data

Fit Summary	
Fit Function	0.3163
Chi-Square	10.1215
Chi-Square DF	8
Pr > Chi-Square	0.2566
Standardized RMR (SRMR)	0.0504
RMSEA Estimate	0.0910
Bentler Comparative Fit Index	0.9872

Except for the fit function values, both FIML and ML methods produce the same set of fit statistics. The difference in the fit function values is expected because the FIML function has a constant term which is derived from the likelihood function. This constant term does not depend on the model parameters. Hence, the FIML and ML discrepancy functions that are used in PROC CALIS are equivalent when VARDEF=N is used in the ML method for analyzing mean and covariance structures.

The parameter estimates are shown in Output 33.16.3 and Output 33.16.4 for the FIML and ML methods, respectively. Except for very tiny numerical differences in some estimates, the FIML and ML estimates match.

Output 33.16.3 Parameter Estimates by the FIML Method: Scores Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value	verbal	math
x1	5.7486 0.9692 5.9315 <.0001 [_Parm1]	0
x2	5.7265 0.9278 6.1720 <.0001 [_Parm2]	0
x3	4.5886 0.7562 6.0676 <.0001 [_Parm3]	0
y1	0	5.1972 0.6796 7.6477 <.0001 [_Parm4]
y2	0	4.1342 0.6036 6.8489 <.0001 [_Parm5]
y3	0	3.7004 0.6177 5.9904 <.0001 [_Parm6]

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value	verbal	math
verbal	1.0000	0.5175 0.1433 3.6113 0.000305 [_Add01]
math	0.5175 0.1433 3.6113 0.000305 [_Add01]	1.0000

Intercepts

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add02	19.90625	1.17540	16.9357	<.0001
x2	_Add03	18.81250	1.14089	16.4893	<.0001
x3	_Add04	18.68750	0.92749	20.1486	<.0001
y1	_Add05	17.90625	0.93161	19.2208	<.0001
y2	_Add06	17.84375	0.78823	22.6377	<.0001
y3	_Add07	17.75000	0.76419	23.2272	<.0001

Output 33.16.3 *continued*

Error Variances					
Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add08	11.16406	4.19243	2.6629	0.0077
x2	_Add09	8.85978	3.78083	2.3433	0.0191
x3	_Add10	6.47248	2.45789	2.6334	0.0085
y1	_Add11	0.76135	1.32719	0.5737	0.5662
y2	_Add12	2.79060	1.08465	2.5728	0.0101
y3	_Add13	4.99466	1.48010	3.3745	0.0007

Output 33.16.4 Parameter Estimates by the ML Method: Scores Data

Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	verbal	math
x1	5.7486 0.9651 5.9567 <.0001 [_Parm1]	0
x2	5.7265 0.9239 6.1981 <.0001 [_Parm2]	0
x3	4.5885 0.7570 6.0617 <.0001 [_Parm3]	0
y1	0	5.1972 0.6779 7.6662 <.0001 [_Parm4]
y2	0	4.1341 0.6025 6.8612 <.0001 [_Parm5]
y3	0	3.7004 0.6143 6.0238 <.0001 [_Parm6]

Output 33.16.4 *continued*

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value	verbal	math
verbal	1.0000	0.5175 0.1406 3.6800 0.000233 [_Add01]
math	0.5175 0.1406 3.6800 0.000233 [_Add01]	1.0000

Intercepts

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add02	19.90625	1.17540	16.9357	<.0001
x2	_Add03	18.81250	1.14089	16.4893	<.0001
x3	_Add04	18.68750	0.92749	20.1486	<.0001
y1	_Add05	17.90625	0.93161	19.2208	<.0001
y2	_Add06	17.84375	0.78823	22.6377	<.0001
y3	_Add07	17.75000	0.76419	23.2272	<.0001

Error Variances

Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
x1	_Add08	11.16365	4.06567	2.7458	0.0060
x2	_Add09	8.85925	3.65397	2.4246	0.0153
x3	_Add10	6.47288	2.47689	2.6133	0.0090
y1	_Add11	0.76124	1.23420	0.6168	0.5374
y2	_Add12	2.79066	1.04307	2.6754	0.0075
y3	_Add13	4.99461	1.40024	3.5670	0.0004

The equivalence between METHOD=ML and METHOD=FIML implies that if you do not have any missing data in your data, you can just use METHOD=ML because it is computationally more efficient than the FIML method.

While the equivalence between ML and FIML is established here with the use of the VARDEF= and MEANSTR options (for data without missing values), it is not necessary in practice to use these options with METHOD=ML. The VARDEF= option is used in this example only to demonstrate the theoretical equivalence between METHOD=ML and METHOD=FIML. The VARDEF= option has very little effect if you have at least a moderate sample size (for example, 30 or more observations).

Merely adding the MEANSTR option to an analysis for data without missing values amounts to adding a saturated mean structure to a covariance structure analysis. In this case, the MEANSTR option only gives you more estimates that pertain to the mean structures, but the parameter estimates that pertain to the covariance structures do not change. Therefore, use the MEANSTR option only when you need to estimate certain mean structure parameters or when you fit models with nonsaturated mean structures.

However, use METHOD=FIML when there are missing values in your data and you need to use every bit of information from the incomplete observations with random missing values.

Example 33.17: Path Analysis: Stability of Alienation

(View the complete code for this example (calex01.sas) in the example repository.)

The following covariance matrix from Wheaton et al. (1977) has served to illustrate the performance of several implementations for the analysis of structural equation models. Two different models have been analyzed by an early implementation of LISREL and are mentioned in Jöreskog (1978). You can also find a more detailed discussion of these models in the LISREL VI manual (Jöreskog and Sörbom 1985). A slightly modified model for this covariance matrix is included in the EQS 2.0 manual (Bentler 1985, p. 28). However, for the analysis with the EQS implementation, the SEI variable is rescaled by a factor of 0.1 to make the matrix less ill-conditioned. Since the Levenberg-Marquardt or Newton-Raphson optimization techniques are used with PROC CALIS, rescaling the data matrix is not necessary and, therefore, is not done here. The results reported here reflect the estimates based on the original covariance matrix.

The path diagram of this model is displayed in Figure 33.1 and is reproduced in the following:

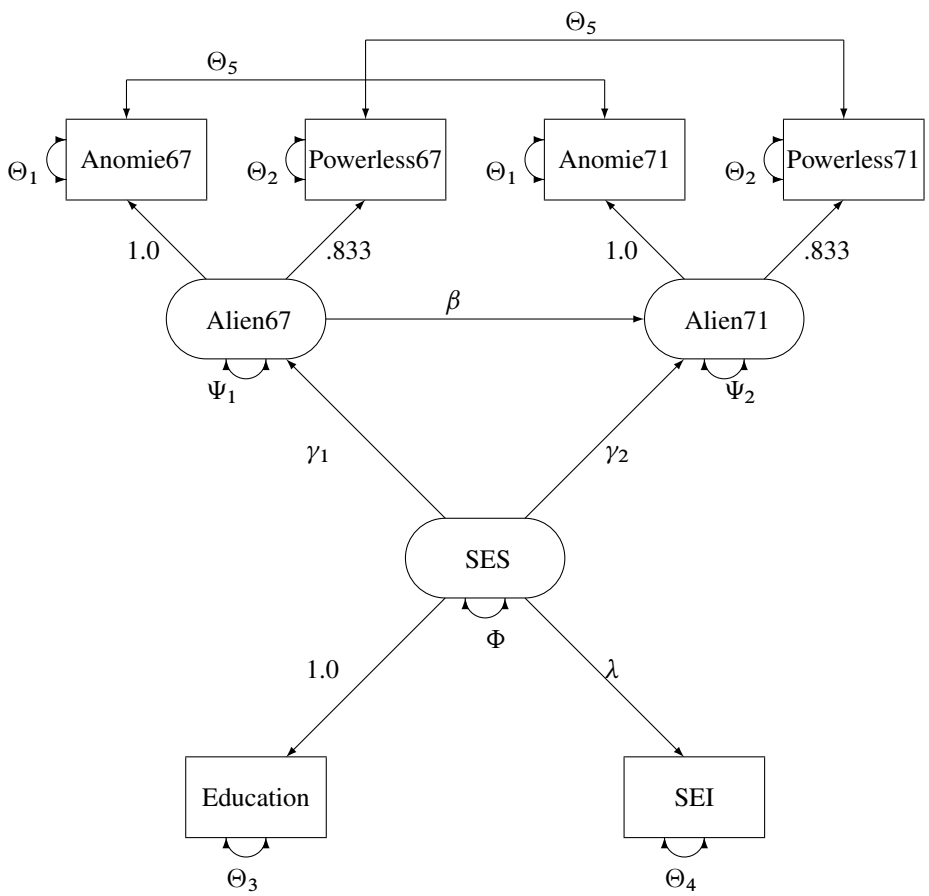

You use the PATH modeling language of PROC CALIS to specify this path model, as shown in the following statements:

```
title "Stability of Alienation";
title2 "Data Matrix of WHEATON, MUTHEN, ALWIN & SUMMERS (1977)";
data Wheaton(TYPE=COV);
   _type_ = 'cov';
   input _name_ $ 1-11 Anomie67 Powerless67 Anomie71 Powerless71
                       Education SEI;
   label Anomie67='Anomie (1967)' Powerless67='Powerlessness (1967)'
         Anomie71='Anomie (1971)' Powerless71='Powerlessness (1971)'
         Education='Education'    SEI='Occupational Status Index';
   datalines;
Anomie67       11.834     .        .        .       .        .
Powerless67     6.947    9.364     .        .       .        .
Anomie71        6.819    5.091   12.532     .       .        .
Powerless71     4.783    5.028    7.495    9.986    .        .
Education      -3.839   -3.889   -3.841   -3.625   9.610     .
SEI           -21.899  -18.831  -21.748  -18.775  35.522  450.288
;

ods graphics on;

proc calis nobs=932 data=Wheaton plots=residuals;
   path
      Anomie67   Powerless67  <===  Alien67   = 1.0  0.833,
      Anomie71   Powerless71  <===  Alien71   = 1.0  0.833,
      Education  SEI          <===  SES       = 1.0  lambda,
      Alien67    Alien71      <===  SES       = gamma1 gamma2,
      Alien71                 <===  Alien67   = beta;
   pvar
      Anomie67     = theta1,
      Powerless67  = theta2,
      Anomie71     = theta1,
      Powerless71  = theta2,
      Education    = theta3,
      SEI          = theta4,
      Alien67      = psi1,
      Alien71      = psi2,
      SES          = phi;
   pcov
      Anomie67    Anomie71    = theta5,
      Powerless67 Powerless71 = theta5;
   pathdiagram title='Stability of Alienation';
run;

ods graphics off;
```

Since no METHOD= option is used in the PROC CALIS statement, maximum likelihood estimates are computed by default.

In the PATH statement, you specify the functional relationships of the variables in the model. These functional relationships are represented as single-headed paths in the path diagram. There are five entries in the PATH statement. You specify the relationships between the latent constructs and the observed variables in the first three path entries. For example, the first entry states that Anomie and Powerless67 are measured indicators of the latent variable Alien67. The path effects or coefficients from the latent factor to these measured indicators

are fixed at 1.0 and 0.833, respectively. Similarly, in the next two path entries, you define the relationships between the latent factors Alien71 and SES and their measured indicators. The last two path entries in the PATH statement represent the functional relationships among the latent variables in the model. SES has effects on Alien67 and Alien71. These effect parameters are labeled or named with gamma1 and gamma2, respectively. Alien67 also has an effect on Alien71, with the effect parameter named beta.

In the PVAR statement, you specify the variance or error variance parameters in the model. These parameters correspond to the double-headed arrows pointing to the individual variables in the path diagram. In the first six entries of the PVAR statement, you specify the error variance parameters of the observed variables. You also give names to these parameters that correspond to the notation in the path diagram. Although you can choose any names for the parameters, it is important to remember that parameters with the same name are identical and will have the same estimates. For example, the error variances of Anomie67 and Anomie71 are the same parameter named theta1. Similarly, you constrain the error variances of Powerless67 and Powerless71. However, the error variance parameters of Education and SEI are unique. They are not constrained with other parameters in the model because they have unique parameter names. Next, you specify the error variance parameters of Alien67 and Alien71. They also have unique parameter names and therefore they are not constrained with any other parameters in the model. Lastly, you specify the variance parameter phi of SES.

In the PCOV statement, you specify the covariances or error covariances among variables in the model. These parameters correspond to the double-headed arrows pointing to distinct pairs of variables in the path diagram. Observed variables Anomie67 and Anomie71 have correlated errors and you specify this error covariance parameter as theta5. Similarly, observed variables Powerless67 and Powerless71 have correlated errors and you also specify this error covariance parameter as theta5. This way, the two error covariances are constrained to be equal.

PROC CALIS can produce a high-quality residual histogram that is useful for showing the distribution of residuals. Before you request the residual histogram, ODS Graphics must be enabled. For example, you can specify the ODS GRAPHICS ON statement, as shown in the preceding statements before the PROC CALIS statement. Then, the residual histogram is requested by the **`plots=residuals`** option in the PROC CALIS statement. PROC CALIS can also produce a high-quality path diagram for the model. You can use the PATHDIAGRAM statement to request the path diagram and to specify related options.

Output 33.17.1 displays the modeling information and variables in the analysis.

Output 33.17.1 Model Specification and Variables

PATH Model Specification

The CALIS Procedure
Covariance Structure Analysis: Model and Initial Values

Modeling Information	
Maximum Likelihood Estimation	
Data Set	WORK.WHEATON
N Obs	932
Model Type	PATH
Analysis	Covariances

Output 33.17.1 *continued*

Variables in the Model		
Endogenous	**Manifest**	Anomie67 Anomie71 Education Powerless67 Powerless71 SEI
	Latent	Alien67 Alien71
Exogenous	**Manifest**	
	Latent	SES

Number of Endogenous Variables = 8
Number of Exogenous Variables = 1

Output 33.17.1 shows that the data set Wheaton was used with 932 observations. The model is specified with the PATH modeling language. Variables in the model are classified into different categories according to their roles. All manifest variables are endogenous in the model. Also, three latent variables are hypothesized in the model: Alien67, Alien71, and SES. While Alien67 and Alien71 are endogenous, SES is exogenous in the model.

Output 33.17.2 echoes the initial specification of the PATH model.

Output 33.17.2 Initial Estimates

Initial Estimates for PATH List

Path			Parameter	Estimate
Anomie67	**<===**	**Alien67**		1.00000
Powerless67	**<===**	**Alien67**		0.83300
Anomie71	**<===**	**Alien71**		1.00000
Powerless71	**<===**	**Alien71**		0.83300
Education	**<===**	**SES**		1.00000
SEI	**<===**	**SES**	lambda	.
Alien67	**<===**	**SES**	gamma1	.
Alien71	**<===**	**SES**	gamma2	.
Alien71	**<===**	**Alien67**	beta	.

Initial Estimates for Variance Parameters

Variance Type	Variable	Parameter	Estimate
Error	**Anomie67**	theta1	.
	Powerless67	theta2	.
	Anomie71	theta1	.
	Powerless71	theta2	.
	Education	theta3	.
	SEI	theta4	.
	Alien67	psi1	.
	Alien71	psi2	.
Exogenous	**SES**	phi	.

Initial Estimates for Covariances Among Errors

Error of	Error of	Parameter	Estimate
Anomie67	**Anomie71**	theta5	.
Powerless67	**Powerless71**	theta5	.

The numerical values for estimates in Output 33.17.2 are the initial values that you input in the model

specification. A blank value for the associated parameter name for a numerical estimate indicates that the estimate is a fixed value, which would not be changed in the estimation. For example, the first five paths have fixed path coefficients, and their fixed values are shown in the Estimate column. For numerical estimates that have specified parameter names, the numerical values serve as initial values, which would be changed during the estimation. Output 33.17.2 does not actually have this type of specification. Missing values '.' are specified as initial values for all the free parameters that are specified in the model. For example, lambda, gamma1, theta1, and psi1, among others, are free parameters that do not have specified initial values. PROC CALIS automatically generates the initial values of these parameters.

You can examine this output to ensure that the desired model is being analyzed. PROC CALIS outputs the initial specifications or the estimation results in the order you specify in the model, unless you use reordering options such as ORDERSPEC and ORDERALL. Therefore, the input order of specifications is important—it determines how your output would look.

Simple descriptive statistics are displayed in Output 33.17.3.

Output 33.17.3 Descriptive Statistics

Simple Statistics			
	Variable	Mean	Std Dev
Anomie67	Anomie (1967)	0	3.44006
Powerless67	Powerlessness (1967)	0	3.06007
Anomie71	Anomie (1971)	0	3.54006
Powerless71	Powerlessness (1971)	0	3.16006
Education	Education	0	3.10000
SEI	Occupational Status Index	0	21.21999

Because the input data set contains only the covariance matrix, the means of the manifest variables are assumed to be zero. Note that this has no impact on the estimation, unless a mean structure model is being analyzed.

Initial estimates are necessary in all kinds of optimization problems. You can provide these initial estimates or let PROC CALIS to generate them automatically. As shown in Output 33.17.2, you did not provide any initial estimates for the parameters. PROC CALIS uses a combination of well-behaved mathematical methods to complete the initial estimation. The initial estimation methods for the current analysis are shown in Output 33.17.4.

Output 33.17.4 Optimization Starting Point

Initial Estimation Methods	
1	Instrumental Variables Method
2	McDonald Method
3	Two-Stage Least Squares

Output 33.17.4 *continued*

Optimization Start Parameter Estimates			
N	**Parameter**	**Estimate**	**Gradient**
1	**lambda**	4.99508	-0.00206
2	**gamma1**	-0.62322	-0.04069
3	**gamma2**	-0.20437	-0.03816
4	**beta**	0.66589	0.03789
5	**theta1**	3.51433	-0.00409
6	**theta2**	3.65991	0.01182
7	**theta3**	2.49860	-0.00578
8	**theta4**	272.85274	0.0000194
9	**psi1**	5.57764	-0.00217
10	**psi2**	3.79636	-0.00935
11	**phi**	7.11140	0.00108
12	**theta5**	0.45298	-0.06463
Value of Objective Function = 0.0365979443			

In this example, the instrumental variable Method, the McDonald and Hartmann method, and the two-stage least squares method have been used for initial estimation. In the same output, the vector of initial parameter estimates and their gradients are also shown. The initial objective function value is 0.0366.

Output 33.17.5 displays the optimization information, including technical details, iteration history and convergence status.

Output 33.17.5 Optimization

Optimization Start			
Active Constraints	0	**Objective Function**	0.0365979443
Max Abs Gradient Element	0.0646338767	**Radius**	1

Iteration	Restarts	Function Calls	Active Constraints	Objective Function	Objective Function Change	Max Abs Gradient Element	Lambda	Ratio Between Actual and Predicted Change
1	0	4	0	0.01453	0.0221	0.00142	0	1.013
2	0	6	0	0.01448	0.000046	0.000249	0	1.001
3	0	8	0	0.01448	1.007E-7	4.717E-6	0	1.006

Optimization Results			
Iterations	3	**Function Calls**	11
Jacobian Calls	5	**Active Constraints**	0
Objective Function	0.0144844814	**Max Abs Gradient Element**	4.7172823E-6
Lambda	0	**Actual Over Pred Change**	1.0060911681
Radius	0.001390392		

Convergence criterion (ABSGCONV=0.00001) satisfied.

The convergence status is important for the validity of your solution. In most cases, you should interpret your results only when the solution is converged. In this example, you obtain a converged solution, as shown in the message at the bottom of the table. The final objective function value is 0.01448, which is the minimized function value during the optimization. If problematic solutions such as nonconvergence are encountered, PROC CALIS issues an error message.

The fit summary statistics are displayed in Output 33.17.6. By default, PROC CALIS displays all available fit indices and modeling information.

Output 33.17.6 Fit Summary

Fit Summary		
Modeling Info	**Number of Observations**	932
	Number of Variables	6
	Number of Moments	21
	Number of Parameters	12
	Number of Active Constraints	0
	Baseline Model Function Value	2.2894
	Baseline Model Chi-Square	2131.4327
	Baseline Model Chi-Square DF	15
	Pr > Baseline Model Chi-Square	<.0001
Absolute Index	**Fit Function**	0.0145
	Chi-Square	13.4851
	Chi-Square DF	9
	Pr > Chi-Square	0.1419
	Z-Test of Wilson & Hilferty	1.0754
	Hoelter Critical N	1169
	Root Mean Square Residual (RMR)	0.2281
	Standardized RMR (SRMR)	0.0150
	Goodness of Fit Index (GFI)	0.9953
Parsimony Index	**Adjusted GFI (AGFI)**	0.9890
	Parsimonious GFI	0.5972
	RMSEA Estimate	0.0231
	RMSEA Lower 90% Confidence Limit	0.0000
	RMSEA Upper 90% Confidence Limit	0.0470
	Probability of Close Fit	0.9705
	ECVI Estimate	0.0405
	ECVI Lower 90% Confidence Limit	0.0357
	ECVI Upper 90% Confidence Limit	0.0556
	Akaike Information Criterion	37.4851
	Bozdogan CAIC	107.5330
	Schwarz Bayesian Criterion	95.5330
	McDonald Centrality	0.9976
Incremental Index	**Bentler Comparative Fit Index**	0.9979
	Bentler-Bonett NFI	0.9937
	Bentler-Bonett Non-normed Index	0.9965
	Bollen Normed Index Rho1	0.9895
	Bollen Non-normed Index Delta2	0.9979
	James et al. Parsimonious NFI	0.5962

First, the fit summary table starts with some basic modeling information, as shown in Output 33.17.6. You can check the number of observations, number of variables, number of moments being fitted, number of parameters, number of active constraints in the solution, and the independent model chi-square and its degrees of freedom in this modeling information category. Next, three types of fit indices are shown: absolute, parsimony, and incremental.

The absolute indices are fit measures that you interpret them without referring to any baseline model. These indices do not adjust for model parsimony. They always favor models with a large number of parameters. The chi-square test statistic is the best-known absolute index in this category. In this example, the p-value of the chi-square is 0.1419, which is greater than the conventional 0.05 value. From the statistical hypothesis testing point of view, you cannot reject this model. The Z-test of Wilson and Hilferty is also insignificant at $\alpha = .05$, which echoes the result of the chi-square test. You can consult other absolute indices as well. Although it seems that there are no clear conventional levels for these absolute indices to indicate an acceptable model fit, you can always use these indices to compare the relative fit among competing models.

Next, the parsimony fit indices take the model parsimony into account. These indices adjust the model fit by the degrees of freedom (or the number of the parameters) of the model in certain ways. The advantage of these indices is that merely increasing the number of parameters in the model might not necessarily lead better model fit measures. These fit indices penalize models with large numbers of parameters. There is no universal way to interpret all these indices. However, for the relatively well-known RMSEA estimate, by convention values under 0.05 indicate good model fit. The RMSEA value for this example is 0.0231, and so this is a very good model fit. For interpretations of other parsimony indices, you can consult the original articles for these indices.

Last, the incremental fit indices are computed based on comparing the target model fit against the fit of a baseline model, which is usually the so-called uncorrelatedness model where all manifest variables are assumed to be uncorrelated. This is the baseline model that PROC CALIS uses. The baseline model fit statistic is shown under the 'Modeling Info' category of the same fit summary table. In this example, the model fit chi-square of the baseline model is 2131.43, with 15 degrees of freedom. The incremental indices show how well the hypothesized model improves over the baseline model for the data. Various incremental fit indices have been proposed. In the fit summary table, there are six of such fit indices. Large values for these indices are desired. It has been suggested that values greater than .9 for these indices indicate acceptable model fit. In this example, all incremental indices but James et al. parsimonious NFI show that the hypothesized model fits well.

There is no consensus as to which fit index is the best to judge model fit. Probably, with artificial data and model, all fit indices can be shown defective in some aspects of measuring model fit. Conventional wisdom is to look at all fit indices and determine whether the majority of them are close to the desirable ranges of values. In this example, almost all fit indices are good, and so it is safe to conclude that the model fits well.

Nowadays, most researchers pay less attention to the model fit chi-square statistic because it tends to reject all meaningful models with minimum departures from the truth. Although the model fit chi-square test statistic is an impeccable statistical inference tool when the underlying statistical assumptions are satisfied, for practical purposes it is just too powerful to accept any useful and reasonable models with only tiny imperfections. Some fit indices are more popular than others. Standardized RMR, RMSEA estimate, adjusted AGFI, and Bentler's comparative fit index are frequently reported in empirical research for judging model fit. In this example, all these measures show good model fit of the hypothesized model. While there are certainly legitimate reasons why these fit indices are more popular than others, they are out of the current scope of discussion.

Figure 33.17.7 shows the path diagram of the unstandardized solution. The path diagram indicates significant

estimates by attaching asterisks to the numerical values. Estimates that are flagged with one asterisk are significant at 0.05 α-level. Estimates that are flagged with two asterisks are significant at 0.01 α-level. The path diagram also shows a summary of fit statistics. For more information about specifying path diagram output, see the section "Path Diagrams: Layout Algorithms, Default Settings, and Customization" on page 1784.

Output 33.17.7 Path Diagram and Fit Summary

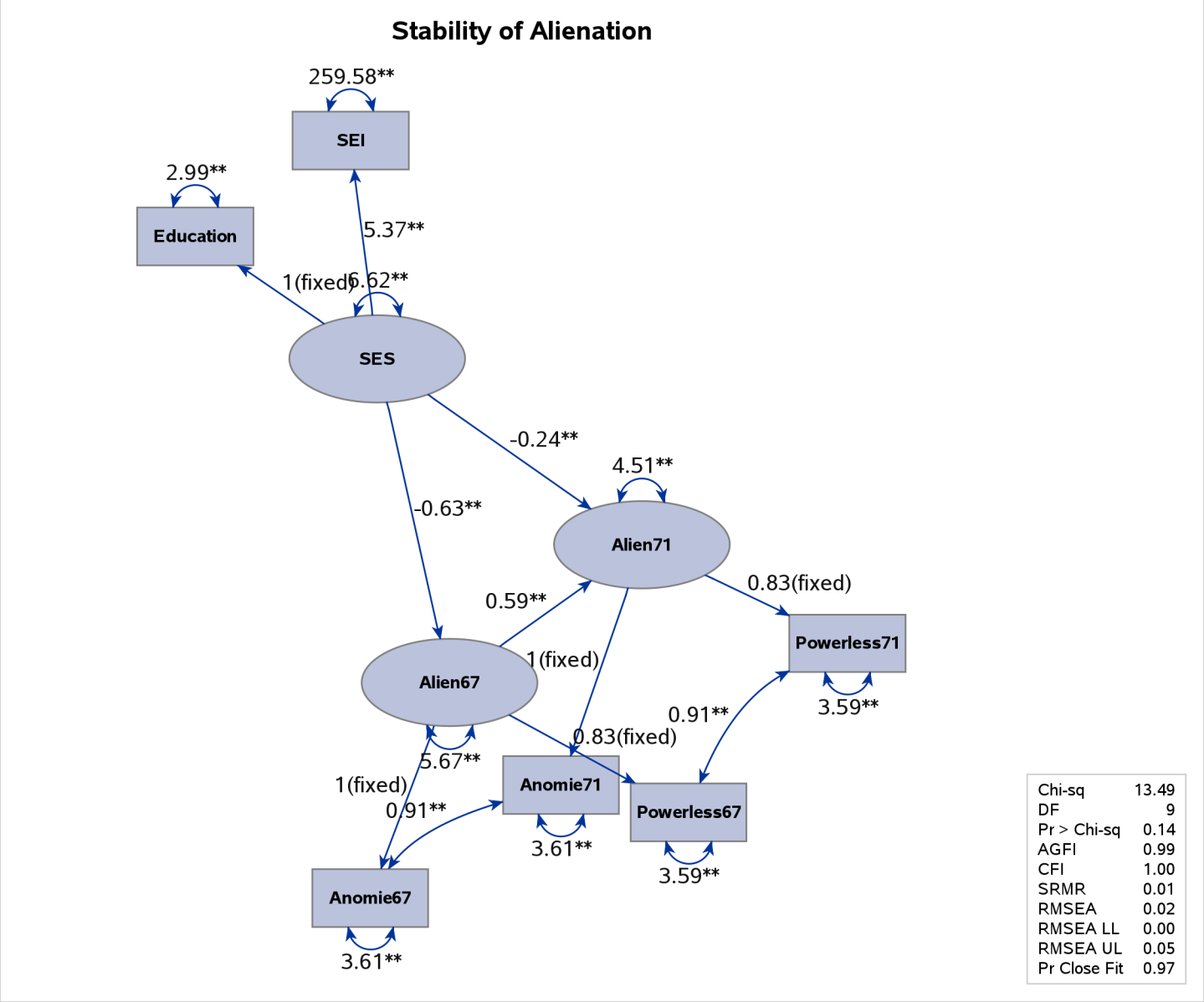

PROC CALIS can perform a detailed residual analysis. Large residuals might indicate misspecification of the model. In Output 33.17.8, raw residuals are reported and ranked.

Output 33.17.8 Raw Residuals and Ranking

Raw Residual Matrix							
		Anomie67	**Powerless67**	**Anomie71**	**Powerless71**	**Education**	**SEI**
Anomie67	Anomie (1967)	-0.06997	0.03642	-0.01116	-0.15200	0.32892	0.47786
Powerless67	Powerlessness (1967)	0.03642	0.01261	0.15600	0.01135	-0.41712	-0.19108
Anomie71	Anomie (1971)	-0.01116	0.15600	-0.08381	-0.00854	0.22464	0.07976
Powerless71	Powerlessness (1971)	-0.15200	0.01135	-0.00854	0.14067	-0.23832	-0.59248
Education	Education	0.32892	-0.41712	0.22464	-0.23832	0.00000	0.00000
SEI	Occupational Status Index	0.47786	-0.19108	0.07976	-0.59248	0.00000	0.00002

Average Absolute Residual	0.153940
Average Off-diagonal Absolute Residual	0.195044

Rank Order of the 10 Largest Raw Residuals		
Var1	**Var2**	**Residual**
SEI	**Powerless71**	-0.59248
SEI	**Anomie67**	0.47786
Education	**Powerless67**	-0.41712
Education	**Anomie67**	0.32892
Education	**Powerless71**	-0.23832
Education	**Anomie71**	0.22464
SEI	**Powerless67**	-0.19108
Anomie71	**Powerless67**	0.15600
Powerless71	**Anomie67**	-0.15200
Powerless71	**Powerless71**	0.14067

Because of the differential scaling of the variables, it is usually more useful to examine the standardized residuals instead. In Output 33.17.9, for example, the table for the 10 largest asymptotically standardized residuals is displayed.

Output 33.17.9 Asymptotically Standardized Residuals and Ranking

Asymptotically Standardized Residual Matrix							
		Anomie67	**Powerless67**	**Anomie71**	**Powerless71**	**Education**	**SEI**
Anomie67	Anomie (1967)	-0.30882	0.52686	-0.05619	-0.86507	2.55338	0.46484
Powerless67	Powerlessness (1967)	0.52686	0.05464	0.87613	0.05735	-2.76371	-0.17015
Anomie71	Anomie (1971)	-0.05619	0.87613	-0.35460	-0.12169	1.69781	0.07009
Powerless71	Powerlessness (1971)	-0.86507	0.05735	-0.12169	0.58521	-1.55750	-0.49608
Education	Education	2.55338	-2.76371	1.69781	-1.55750	0.00000	0.00000
SEI	Occupational Status Index	0.46484	-0.17015	0.07009	-0.49608	0.00000	0.00000

Average Standardized Residual	0.646672
Average Off-diagonal Standardized Residual	0.818456

Output 33.17.9 *continued*

Rank Order of the 10 Largest Asymptotically Standardized Residuals		
Var1	**Var2**	**Residual**
Education	**Powerless67**	-2.76371
Education	**Anomie67**	2.55338
Education	**Anomie71**	1.69781
Education	**Powerless71**	-1.55750
Anomie71	**Powerless67**	0.87613
Powerless71	**Anomie67**	-0.86507
Powerless71	**Powerless71**	0.58521
Powerless67	**Anomie67**	0.52686
SEI	**Powerless71**	-0.49608
SEI	**Anomie67**	0.46484

The model performs the poorest concerning the covariances of Education with all measures of Powerless and Anomie. This might suggest a misspecification of the functional relationships of Education with other variables in the model. However, because the model fit is quite good, such a possible misspecification should not be a serious concern in the analysis.

The histogram of the asymptotically standardized residuals is displayed in Output 33.17.10, which also shows the normal and kernel approximations.

Output 33.17.10 Distribution of Asymptotically Standardized Residuals

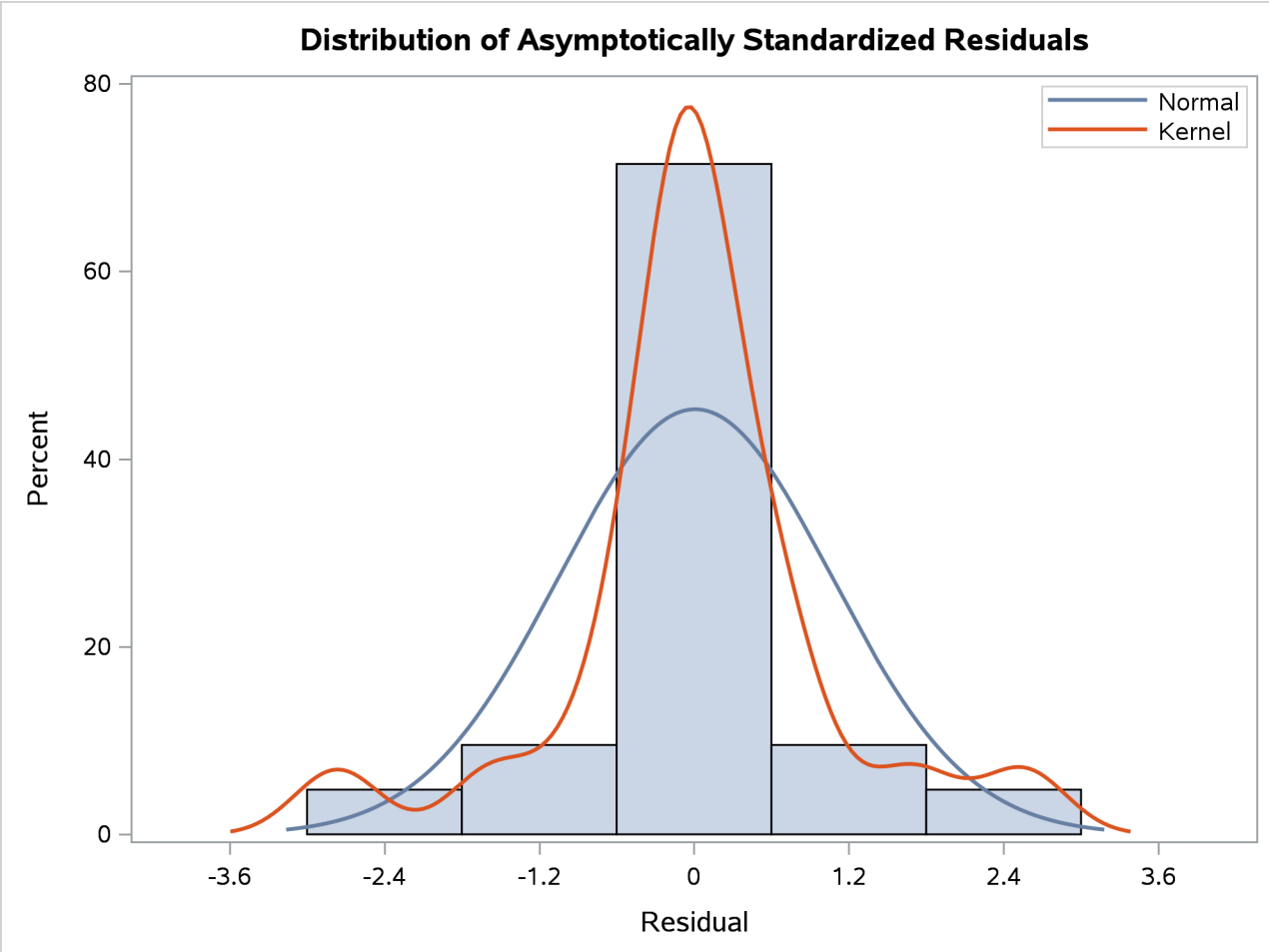

The residual distribution looks quite symmetrical. It shows a small to medium departure from the normal distribution, as evidenced by the discrepancies between the kernel and the normal distribution curves.

Output 33.17.11 shows the estimation results.

Output 33.17.11 Estimation Results

PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
Anomie67	**<===**	**Alien67**		1.00000			
Powerless67	**<===**	**Alien67**		0.83300			
Anomie71	**<===**	**Alien71**		1.00000			
Powerless71	**<===**	**Alien71**		0.83300			
Education	**<===**	**SES**		1.00000			
SEI	**<===**	**SES**	lambda	5.36883	0.43371	12.3788	<.0001
Alien67	**<===**	**SES**	gamma1	-0.62994	0.05634	-11.1809	<.0001
Alien71	**<===**	**SES**	gamma2	-0.24086	0.05489	-4.3884	<.0001
Alien71	**<===**	**Alien67**	beta	0.59312	0.04678	12.6788	<.0001

Output 33.17.11 *continued*

Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	Anomie67	theta1	3.60796	0.20092	17.9572	<.0001
	Powerless67	theta2	3.59488	0.16448	21.8556	<.0001
	Anomie71	theta1	3.60796	0.20092	17.9572	<.0001
	Powerless71	theta2	3.59488	0.16448	21.8556	<.0001
	Education	theta3	2.99366	0.49861	6.0040	<.0001
	SEI	theta4	259.57639	18.31151	14.1756	<.0001
	Alien67	psi1	5.67046	0.42301	13.4050	<.0001
	Alien71	psi2	4.51479	0.33532	13.4639	<.0001
Exogenous	SES	phi	6.61634	0.63914	10.3519	<.0001

Covariances Among Errors						
Error of	Error of	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Anomie67	Anomie71	theta5	0.90580	0.12167	7.4447	<.0001
Powerless67	Powerless71	theta5	0.90580	0.12167	7.4447	<.0001

The paths, variances and partial (or error) variances, and covariances and partial covariances are shown. When you have fixed parameters such as the first five path coefficients in the output, the standard errors and t values are all blanks. For free or constrained estimates, standard errors and t values are computed. Researchers in structural equation modeling usually use the value 2 as an approximate critical value for the observed t values. The reason is that the estimates are asymptotically normal, and so the two-sided critical point with $\alpha = 0.05$ is 1.96, which is close to 2. Using this criterion, all estimates shown in Output 33.17.11 are significantly different from zero, supporting the presence of these parameters in the model.

Squared multiple correlations are shown in Output 33.17.12.

Output 33.17.12 Squared Multiple Correlations

Squared Multiple Correlations			
Variable	Error Variance	Total Variance	R-Square
Anomie67	3.60796	11.90397	0.6969
Anomie71	3.60796	12.61581	0.7140
Education	2.99366	9.61000	0.6885
Powerless67	3.59488	9.35139	0.6156
Powerless71	3.59488	9.84533	0.6349
SEI	259.57639	450.28798	0.4235
Alien67	5.67046	8.29601	0.3165
Alien71	4.51479	9.00786	0.4988

For each endogenous variable in the model, the corresponding squared multiple correlation is computed by:

$$1 - \frac{\text{error variance}}{\text{total variance}}$$

In regression analysis, this is the percentage of explained variance of the endogenous variable by the predictors. However, this interpretation is complicated or even uninterpretable when your structural equation model

has correlated errors or reciprocal casual relations. In these situations, it is not uncommon to see negative R-squares. Negative R-squares do not necessarily mean that your model is wrong or the model prediction is weak. Rather, the R-square interpretation is questionable in these situations.

When your variables are measured on different scales, comparison of path coefficients cannot be made directly. For example, in Output 33.17.11, the path coefficient for path Education <=== SES is fixed at one, while the path coefficient for path SEI <=== SES is 5.369. It would be simple-minded to conclude that the effect of SES on SEI is greater than that SES on Education. Because SEI and Education are measured on different scales, direct comparison of the corresponding path coefficients is simply inappropriate.

In alleviating this problem, some might resort to the standardized solution for a better comparison. In a standardized solution, because the variances of manifest variables and systematic predictors are all standardized to ones, you hope the path coefficients are more comparable. In this example, PROC CALIS standardizes your results in Output 33.17.13.

Output 33.17.13 Standardized Results

Standardized Results for PATH List

Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
Anomie67	**<===**	**Alien67**		0.83481	0.01093	76.3531	<.0001
Powerless67	**<===**	**Alien67**		0.78459	0.01163	67.4776	<.0001
Anomie71	**<===**	**Alien71**		0.84499	0.01031	81.9796	<.0001
Powerless71	**<===**	**Alien71**		0.79678	0.01107	71.9626	<.0001
Education	**<===**	**SES**		0.82975	0.03172	26.1599	<.0001
SEI	**<===**	**SES**	lambda	0.65079	0.03019	21.5533	<.0001
Alien67	**<===**	**SES**	gamma1	-0.56257	0.03456	-16.2796	<.0001
Alien71	**<===**	**SES**	gamma2	-0.20642	0.04483	-4.6043	<.0001
Alien71	**<===**	**Alien67**	beta	0.56920	0.04066	14.0000	<.0001

Standardized Results for Variance Parameters

Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**Anomie67**	theta1	0.30309	0.01825	16.6031	<.0001
	Powerless67	theta2	0.38442	0.01825	21.0695	<.0001
	Anomie71	theta1	0.28599	0.01742	16.4178	<.0001
	Powerless71	theta2	0.36514	0.01764	20.6942	<.0001
	Education	theta3	0.31152	0.05264	5.9182	<.0001
	SEI	theta4	0.57647	0.03930	14.6680	<.0001
	Alien67	psi1	0.68352	0.03888	17.5797	<.0001
	Alien71	psi2	0.50121	0.03321	15.0897	<.0001
Exogenous	**SES**	phi	1.00000			

Standardized Results for Covariances Among Errors

Error of	Error of	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Anomie67	**Anomie71**	theta5	0.07391	0.01013	7.2957	<.0001
Powerless67	**Powerless71**	theta5	0.09440	0.01274	7.4092	<.0001

Now, the standardized path coefficient for path Education <=== SES is 0.830, while the standardized path coefficient for path SEI <=== SES is 0.651. So the standardized effect of SES on SEI is actually smaller

than that of SES on Education.

Furthermore, in PROC CALIS the standardized estimates are computed with standard error estimates and t values so that you can make statistical inferences on the standardized estimates as well.

PROC CALIS might differ from other software in its standardization scheme. Unlike other software that might standardize the path coefficients that attach to the error terms (unsystematic sources), PROC CALIS keeps these path coefficients at ones (not shown in the output). Unlike other software that might also standardize the corresponding error variances to ones, the error variances in the standardized solution of PROC CALIS are rescaled so as to keep the mathematical consistency of the model.

Essentially, in PROC CALIS only variances of manifest and non-error-type latent variables are standardized to ones. The error variances are rescaled, but not standardized. For example, in the standardized solution shown in Output 33.17.13, the error variances for all endogenous variables are not ones (see the middle portion of the output). Only the variance for the latent variable SES is standardized to one. See the section "Standardized Solutions" on page 1847 for the logic of the standardization scheme adopted by PROC CALIS.

In appearance, the standardized solution is like a correlational analysis on the standardized manifest variables with standardized exogenous latent factors. Unfortunately, this statement is over-simplified, if not totally inappropriate. In standardizing a solution, the implicit equality constraints are likely destroyed. In this example, the unstandardized error variances for Anomie67 and Anomie71 are both 3.608, represented by a common parameter theta1. However, after standardization, these error variances have different values at 0.303 and 0.286, respectively. In addition, fixed parameter values are no longer fixed in a standardized solution (for example, the first five paths in the current example). The issue of standardization is common to all other SEM software and beyond the current discussion. PROC CALIS provides the standardized solution so that users can interpret the standardized estimates whenever they find them appropriate.

Example 33.18: Simultaneous Equations with Mean Structures and Reciprocal Paths

(View the complete code for this example (calex02.sas) in the example repository.)

The supply-and-demand food example of Kmenta (1971, pp. 565, 582) is used to illustrate PROC CALIS for the estimation of intercepts and coefficients of simultaneous equations in econometrics. The model is specified by two simultaneous equations containing two endogenous variables Q and P, and three exogenous variables D, F, and Y:

$$Q_t(demand) = \alpha_1 + \beta_1 P_t + \gamma_1 D_t$$

$$Q_t(supply) = \alpha_2 + \beta_2 P_t + \gamma_2 F_t + \gamma_3 Y_t$$

for $t = 1, \ldots, 20$.

To analyze this model in PROC CALIS, the second equation needs to be written in another form. For instance, in the LINEQS model each endogenous variable must appear on the left-hand side of exactly one equation. To satisfy this requirement, you can rewrite the second equation as an equation for P_t as:

$$P_t = -\frac{\alpha_2}{\beta_2} + \frac{1}{\beta_2} Q_t - \frac{\gamma_2}{\beta_2} F_t - \frac{\gamma_3}{\beta_2} Y_t$$

or, equivalently reparameterized as:

$$P_t = \theta_1 + \theta_2 Q_t + \theta_3 F_t + \theta_4 Y_t$$

where

$$\theta_1 = -\frac{\alpha_2}{\beta_2}, \quad \theta_2 = \frac{1}{\beta_2}, \quad \theta_3 = -\frac{\gamma_2}{\beta_2}, \quad \theta_4 = -\frac{\gamma_3}{\beta_2}$$

This new equation for P_t together with the first equation for Q_t suggest the following LINEQS model specification in PROC CALIS:

```
title 'Food example of KMENTA(1971, p.565 & 582)';
data food;
   input Q P D F Y;
   label  Q='Food Consumption per Head'
          P='Ratio of Food Prices to General Price'
          D='Disposable Income in Constant Prices'
          F='Ratio of Preceding Years Prices'
          Y='Time in Years 1922-1941';
   datalines;
  98.485  100.323   87.4   98.0   1
  99.187  104.264   97.6   99.1   2
 102.163  103.435   96.7   99.1   3
 101.504  104.506   98.2   98.1   4
 104.240   98.001   99.8  110.8   5
 103.243   99.456  100.5  108.2   6
 103.993  101.066  103.2  105.6   7
  99.900  104.763  107.8  109.8   8
 100.350   96.446   96.6  108.7   9
 102.820   91.228   88.9  100.6  10
  95.435   93.085   75.1   81.0  11
  92.424   98.801   76.9   68.6  12
  94.535  102.908   84.6   70.9  13
  98.757   98.756   90.6   81.4  14
 105.797   95.119  103.1  102.3  15
 100.225   98.451  105.1  105.0  16
 103.522   86.498   96.4  110.5  17
  99.929  104.016  104.4   92.5  18
 105.223  105.769  110.7   89.3  19
 106.232  113.490  127.1   93.0  20
;

proc calis data=food pshort nostand;
   lineqs
      Q = alpha1 * Intercept + beta1  * P  + gamma1 * D + E1,
      P = theta1 * Intercept + theta2 * Q  + theta3 * F + theta4 * Y + E2;
   variance
      E1-E2 = eps1-eps2;
   cov
      E1-E2 = eps3;
   bounds
      eps1-eps2 >= 0. ;
run;
```

The LINEQS modeling language is used in this example because its specification is similar to the original equations. In the LINEQS statement, you essentially input the two model equations for Q and P. Parameters

for intercepts and regression coefficients are also specified in the equations. Note that Intercept in the two equations is treated as a special variable that contains ones for all observations. Intercept is not a variable in the data set, nor do you need to create such a variable in your data set. Hence, the variable Intercept does not represent the intercept parameter itself. Instead, the intercept parameters for the two equations are the coefficients attached to Intercept. In this example, the intercept parameters are alpha1 and theta1, respectively, in the two equations. As required, error terms E1 and E2 are added to complete the equation specification.

In the VARIANCE statement, you specify eps1 and eps2, respectively, for the variance parameters of the error terms. In the COV, you specify eps3 for the covariance parameter between the error terms. In the BOUNDS statement, you set lower bounds for the error variances so that estimates of eps1 and eps2 would be nonnegative.

In this example, the PSHORT and the NOSTAND options are used in the PROC CALIS statement. The PSHORT option suppresses a large amount of the output. For example, initial estimates are not printed and simple descriptive statistics and standard errors are not computed. The NOSTAND option suppresses the printing of the standardized results. Because the default printing in PROC CALIS might produce a large amount of output, using these printing options make your output more concise and readable. Whenever appropriate, you may consider using these printing options.

The estimated equations are shown in Output 33.18.1.

Output 33.18.1 Linear Equations

Linear Equations

```
Q =   93.6193  (**)  Intercept  +  -0.2295  (**)  P  +   0.3100  (**)  D  +   1.0000        E1
P =    -218.9  (ns)  Intercept  +   4.2140  (**)  Q  +  -0.9305  (**)  F  +  -1.5579  (**)  Y   +  1.0000    E2
```

The estimates of intercepts and regression coefficients are shown directly in the equations. Any number in an equation followed by an asterisk is an estimate. For the estimates in equations, the parameter names are shown underneath the associated variables. Any number in an equation not followed by an asterisk is a fixed value. For example, the value 1.0000 attached to the error term in each of the output equation is fixed. Also, for fixed coefficients there are no parameter names underneath the associated variables.

All but the intercept estimates in the equation for predicting P are statistically significant at $\alpha = 0.05$ (when using an approximate critical value of 2). The t ratio for theta1 is –1.590, which implies that this intercept might have been zero in the population. However, because you have reparameterized the original model to use the LINEQS model specification, transformed parameters like theta1 in this model might not be of primary interest. Therefore, you might not need to pay any attention to the significance of the theta1 estimate. There is a way to use the original econometric parameters to specify the LINEQS model. It is discussed in the later part of this example.

Estimates for variance, covariance, and mean parameters are shown in Output 33.18.2.

Output 33.18.2 Variance, Covariance, and Mean Parameters

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	E1	eps1	3.51274	1.20204	2.9223	0.0035
	E2	eps2	105.06749	83.89446	1.2524	0.2104
Observed	D	_Add1	139.96029	45.40911	3.0822	0.0021
	F	_Add2	161.51355	52.40192	3.0822	0.0021
	Y	_Add3	35.00000	11.35550	3.0822	0.0021

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
E1	E2	eps3	-18.87270	8.77951	-2.1496	0.0316
F	D	_Add4	74.02539	38.44699	1.9254	0.0542
Y	D	_Add5	22.99211	16.90102	1.3604	0.1737
Y	F	_Add6	-21.58158	17.94544	-1.2026	0.2291

Mean Parameters

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Observed	D	_Add7	97.53500	2.71410	35.9364	<.0001
	F	_Add8	96.62500	2.91560	33.1407	<.0001
	Y	_Add9	10.50000	1.35724	7.7363	<.0001

Parameters with a name prefix _Add are added automatically by PROC CALIS. These parameters are added as free parameters to complete the model specification. In PROC CALIS, variances and covariances among the set of exogenous manifest variables must be parameters. You either specify them explicitly or let the CALIS procedure to add them. If you need to constrain or to fix these parameters, then you must specify them explicitly. When your model also fits the mean structures, the same principle applies to the means of the exogenous manifest variables. In this example, because variables D, F, and Y are all exogenous manifest variables, their associated means, variances and covariances must be parameters in the model.

The squared multiple correlations for the equations are shown in Output 33.18.3.

Output 33.18.3 Squared Multiple Correlations

Squared Multiple Correlations

Variable	Error Variance	Total Variance	R-Square
Q	3.51274	14.11128	0.7511
P	105.06749	35.11850	-1.9918

For endogenous variable P, the R-square is –1.9918, which is obviously an invalid value. In fact, because there are correlated errors (between E1 and E2) and reciprocal paths (paths to and from Q and P), the model departs from the regular assumptions of multiple regression analysis. As a result, you should not interpret the R-squares for this example.

Specifying the LINEQS with the Original Econometric Parameters

If you are interested in estimating the parameters in the original econometric model (that is, α_2, β_2, γ_2, and γ_3), the previous reparameterized LINEQS model does not serve your purpose well enough. However, using the relations between these original parameters with the θ parameters in the reparameterized LINEQS model, you can set up some "super-parameters" in the LINEQS model, as shown in the following statements:

```
proc calis data=Food pshort nostand;
   lineqs
      Q = alpha1 * Intercept + beta1  * P  + gamma1 * D + E1,
      P = theta1 * Intercept + theta2 * Q  + theta3 * F + theta4 * Y + E2;
   variance
      E1-E2 = eps1-eps2;
   cov
      E1-E2 = eps3;
   bounds
      eps1-eps2 >= 0. ;
   parameters alpha2 (50.) beta2 gamma2 gamma3 (3*.25);
      theta1  = -alpha2 / beta2;
      theta2  = 1 / beta2;
      theta3  = -gamma2 / beta2;
      theta4  = -gamma3 / beta2;
run;
```

In this new specification, only the PARAMETERS statement and the SAS programming statements following it are new. In the PARAMETERS statement, you define super-parameters alpha2, beta2, gamma2, and gamma3, and put initial values for them in parentheses. These parameters are the original econometric parameters of interest. The SAS programming statements that follow the PARAMETERS statement are used to define the functional relationships of the super-parameters with the parameters in the LINEQS model. Consequently, in this new specification, theta1, theta2, theta3, and theta4 are no longer independent parameters in the model, as they are in the previous reparameterized model. Instead, alpha2, beta2, gamma2, and gamma3 are independent parameters in this new specification. By fitting this new model, you get the same set of estimates as those in the previous LINEQS model. In addition, you get estimates of the super-parameters, as shown in Output 33.18.4.

Output 33.18.4 Additional Parameters

Additional Parameters

Type	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Independent	alpha2	51.94452	11.70002	4.4397	<.0001
	beta2	0.23731	0.09877	2.4026	0.0163
	gamma2	0.22082	0.04161	5.3070	<.0001
	gamma3	0.36971	0.07060	5.2365	<.0001

You can now interpret the results in terms of the original econometric parameterization. As shown in Output 33.18.4, all these estimates are significant, despite the fact that one of the transformed parameter estimates in the linear equations of the LINEQS model is not. You can obtain almost equivalent results by applying the SAS/ETS procedure SYSLIN on this problem.

Alternative Ways to Specify Your LINEQS Model

In specifying the linear equations in the LINEQS model, it might become cumbersome when you need to name a lot of parameters into the equations. If the parameters in your model are unconstrained, you need to very careful to use unique parameter names to distinguish the free parameters because parameters with the same name are identical and will have the same estimate. To make model specification easier and to avoid accidental constraints, PROC CALIS provides an efficient way to specify these free parameters. That is, you can simply omit the parameter names in the specification. For example, in the first specification of the current example, except for the boundary constraints on the error variance parameters, all other parameters in the model are not constrained, as shown in the following statements:

```
proc calis data=food pshort nostand;
   lineqs
      Q = alpha1 * Intercept + beta1  * P  + gamma1 * D + E1,
      P = theta1 * Intercept + theta2 * Q  + theta3 * F + theta4 * Y + E2;
   variance
      E1-E2 = eps1-eps2;
   cov
      E1-E2 = eps3;
   bounds
      eps1-eps2 >= 0. ;
run;
```

Parameters such as alpha1, beta1, and so on are unique parameter names in the specific locations of the model. They are free parameters. Hence, you can use the following equivalent specification:

```
proc calis data=food pshort nostand;
   lineqs
      Q =  * Intercept +  *  P  +  * D        + E1,
      P =  * Intercept +  *  Q  +  * F +  * Y + E2;
   variance
      E1-E2 = eps1-eps2;
   cov
      E1 E2;
   bounds
      eps1-eps2 >= 0. ;
run;
```

Only the parameters eps1 and eps2 remain in this equivalent specification. You omit the specification of all other parameter names. But the estimation results are the same, as shown in Output 33.18.5.

Output 33.18.5 Estimation Results With Generated Parameter Names

Linear Equations

Q =	93.6193	(**)	Intercept	+	-0.2295	(**)	P	+	0.3100	(**)	D	+	1.0000		E1		
P =	-218.9	(ns)	Intercept	+	4.2140	(**)	Q	+	-0.9305	(**)	F	+	-1.5579	(**)	Y	+	1.0000	E2

Output 33.18.5 *continued*

Effects in Linear Equations

Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Q	Intercept	_Parm1	93.61928	7.57485	12.3592	<.0001
Q	P	_Parm2	-0.22954	0.09235	-2.4856	0.0129
Q	D	_Parm3	0.31001	0.04481	6.9186	<.0001
P	Intercept	_Parm4	-218.89312	137.69794	-1.5897	0.1119
P	Q	_Parm5	4.21398	1.75396	2.4025	0.0163
P	F	_Parm6	-0.93053	0.39597	-2.3500	0.0188
P	Y	_Parm7	-1.55795	0.66497	-2.3429	0.0191

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**E1**	eps1	3.51274	1.20204	2.9223	0.0035
	E2	eps2	105.06749	83.89446	1.2524	0.2104
Observed	**D**	_Add1	139.96029	45.40911	3.0822	0.0021
	F	_Add2	161.51355	52.40192	3.0822	0.0021
	Y	_Add3	35.00000	11.35550	3.0822	0.0021

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
E1	**E2**	_Parm8	-18.87270	8.77951	-2.1496	0.0316
F	**D**	_Add4	74.02539	38.44699	1.9254	0.0542
Y	**D**	_Add5	22.99211	16.90102	1.3604	0.1737
Y	**F**	_Add6	-21.58158	17.94544	-1.2026	0.2291

Mean Parameters

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Observed	**D**	_Add7	97.53500	2.71410	35.9364	<.0001
	F	_Add8	96.62500	2.91560	33.1407	<.0001
	Y	_Add9	10.50000	1.35724	7.7363	<.0001

The estimation results in Output 33.18.5 are the same as those in Output 33.18.2 and Output 33.18.3 with the original LINEQS model specification, only now PROC CALIS generates the parameter names with the _Parm in the results, as shown in Output 33.18.5. Note that you retain the parameter names eps1 and eps2 because you need to refer to them in the BOUNDS statement.

Example 33.19: Fitting Direct Covariance Structures

(View the complete code for this example (calex03.sas) in the example repository.)

In the section "Direct Covariance Structures Analysis" on page 1529, the MSTRUCT modeling language is used to specify a model with direct covariance structures. In the model, four variables from the data set of Wheaton et al. (1977) are used. The analysis is carried out in this example to investigate the tenability of the hypothesized covariance structures.

The four variables used are: Anomie67, Powerless67, Anomie71, and Powerless71. The hypothesized covariance matrix is structured as:

$$\mathbf{\Sigma} = \begin{pmatrix} \phi_1 & \theta_1 & \theta_2 & \theta_1 \\ \theta_1 & \phi_2 & \theta_l & \theta_3 \\ \theta_2 & \theta_1 & \phi_1 & \theta_1 \\ \theta_1 & \theta_3 & \theta_1 & \phi_2 \end{pmatrix}$$

where:

ϕ_1:	variance of anomie
ϕ_2:	variance of powerlessness
θ_1:	covariance between anomie and powerlessness
θ_2:	covariance between anomie measures
θ_3:	covariance between powerlessness measures

In this example, you hypothesize the covariance structures directly, as opposed to those models with implied covariance structures from path models (see Example 33.17), structural equations (see Example 33.18), or other types of models. The basic assumption of the direct covariance structures in this example is that Anomie and Powerless were invariant over the measurement periods employed. This implies that the time of measurement did not change the variances and covariances of the measures. Therefore, both Anomie67 and Anomie71 have the same variance parameter ϕ_1, and both Powerless67 and Powerless71 have the same variance parameter ϕ_2. These two parameters, ϕ_1 and ϕ_2, are hypothesized on the diagonal of the covariance matrix $\mathbf{\Sigma}$. In the same structured covariance matrix, θ_1 represents the covariance between Anomie and Powerless, without regard to the time of measurement. The θ_2 parameter represents the covariance between the Anomie measures, or the reliability of the Anomie measure. Similarly, the θ_2 parameter represents the covariance between the Powerless measures, or the reliability of the Anomie measure.

As explained in the section "Direct Covariance Structures Analysis" on page 1529, you can use the MSTRUCT modeling language to specify the hypothesized covariance structures directly, as shown in the following statements:

```
proc calis nobs=932 data=Wheaton psummary;
   fitindex on(only)=[chisq df probchi] outfit=savefit;
   mstruct
      var = Anomie67 Powerless67 Anomie71 Powerless71;
   matrix _COV_ [1,1] = phi1,
                [2,2] = phi2,
                [3,3] = phi1,
                [4,4] = phi2,
```

```
                    [2,1] = theta1,
                    [3,1] = theta2,
                    [3,2] = theta1,
                    [4,1] = theta1,
                    [4,2] = theta3,
                    [4,3] = theta1;
run;
```

In the MSTRUCT statement you specify the variables in the VAR= list. The order of variables in this VAR= list is assumed to be the same as that in the row and column of the hypothesized covariance matrix. Next, in the MATRIX statement you specify parameters as entries in the hypothesized covariance matrix _COV_. Only the lower diagonal elements need to be specified because covariance matrices, by nature, are symmetric. Redundant specification of the upper triangular elements are unnecessary as PROC CALIS has the information accounted for. You can also set initial estimates by putting parenthesized numbers after the parameter names. But in this example you let PROC CALIS determine all the initial estimates.

In the PROC CALIS statement, the PSUMMARY option is used. As a global display option, this option suppresses a lot of displayed output and requests only the fit summary table be printed. This way you can eliminate quite a lot of displayed output that is not of your primary interest. In this example, the specification of the covariance structures is straightforward, and you do not need any output regarding the initial estimation or standardized solution. Suppose that you are not even concerned with the estimates of the parameters because you are not yet sure if this model is good enough for the data. All you want to know at this stage is whether the hypothesized covariance structures fit the data well. Therefore, the PSUMMARY option would serve your purpose well in this example.

In fact, even the fit summary table can be trimmed down quite a bit if you only want to look at certain specific fit indices. In the FITINDEX statement of this example, the ON(ONLY)= option turns on the printing of the model fit chi-square, its *df*, and *p*-value only. This does not mean that you must lose the information of all other fit indices. In addition to the printed output, you can save all fit indices in an output data set. To this end, you can use the OUTFIT= option in the FITINDEX statement. In this example, you save the results of all fit indices in a SAS data set called savefit.

Output 33.19.1 shows the entire printed output.

Output 33.19.1 Testing Direct Covariance Structures

Fit Summary	
Chi-Square	221.5798
Chi-Square DF	5
Pr > Chi-Square	<.0001

The displayed output is very concise. It contains only a fit summary table with three statistics. The *p*-value for the model fit chi-square test indicates that the hypothesized structures should be rejected at $\alpha = 0.05$. Therefore, this rather restrictive direct covariance structure model does not fit the data well. A less restrictive covariance structure model is needed to explain the variances and covariances.

All fit indices are saved in the savefit data set. To view it, you can use the following statement:

```
proc print data=savefit;
run;
```

Output 33.19.2 shows all indices, their types and values of all fit indices and information.

Output 33.19.2 Saved Fit Indices

Analysis of Direct Covariance Structures
Testing Model by the MSTRUCT Language

Obs	_TYPE_	IndexCode	FitIndex	FitValue	PrintChar
1	ModelInfo	101	Number of Observations	932.00	932
2	ModelInfo	103	Number of Variables	4.00	4
3	ModelInfo	104	Number of Moments	10.00	10
4	ModelInfo	105	Number of Parameters	5.00	5
5	ModelInfo	106	Number of Active Constraints	0.00	0
6	ModelInfo	111	Baseline Model Function Value	1.68	1.6799
7	ModelInfo	113	Baseline Model Chi-Square	1563.94	1563.9442
8	ModelInfo	114	Baseline Model Chi-Square DF	6.00	6
9	ModelInfo	115	Pr > Baseline Model Chi-Square	0.00	<.0001
10	Absolute	201	Fit Function	0.24	0.2380
11	Absolute	203	Chi-Square	221.58	221.5798
12	Absolute	204	Chi-Square DF	5.00	5
13	Absolute	205	Pr > Chi-Square	0.00	<.0001
14	Absolute	211	Z-Test of Wilson & Hilferty	12.25	12.2533
15	Absolute	212	Hoelter Critical N	47.00	47
16	Absolute	213	Root Mean Square Residual (RMR)	0.76	0.7649
17	Absolute	214	Standardized RMR (SRMR)	0.07	0.0701
18	Absolute	215	Goodness of Fit Index (GFI)	0.90	0.9036
19	Parsimony	301	Adjusted GFI (AGFI)	0.81	0.8071
20	Parsimony	302	Parsimonious GFI	0.75	0.7530
21	Parsimony	303	RMSEA Estimate	0.22	0.2157
22	Parsimony	304	RMSEA Lower 90% Confidence Limit	0.19	0.1920
23	Parsimony	305	RMSEA Upper 90% Confidence Limit	0.24	0.2404
24	Parsimony	306	Probability of Close Fit	0.00	<.0001
25	Parsimony	307	ECVI Estimate	0.25	0.2488
26	Parsimony	308	ECVI Lower 90% Confidence Limit	0.20	0.2003
27	Parsimony	309	ECVI Upper 90% Confidence Limit	0.31	0.3053
28	Parsimony	310	Akaike Information Criterion	231.58	231.5798
29	Parsimony	311	Bozdogan CAIC	260.77	260.7665
30	Parsimony	312	Schwarz Bayesian Criterion	255.77	255.7665
31	Parsimony	313	McDonald Centrality	0.89	0.8903
32	Incremental	401	Bentler Comparative Fit Index	0.86	0.8610
33	Incremental	403	Bentler-Bonett NFI	0.86	0.8583
34	Incremental	402	Bentler-Bonett Non-normed Index	0.83	0.8332
35	Incremental	405	Bollen Normed Index Rho1	0.83	0.8300
36	Incremental	406	Bollen Non-normed Index Delta2	0.86	0.8611
37	Incremental	404	James et al. Parsimonious NFI	0.72	0.7153

The results of various fit indices from this output data set confirm that the hypothesized model does not fit the data well.

As an aside, it is noted with some shorthand notation, the specification of the MSTRUCT model parameters that use the MATRIX statements can be made a little more precise for the current example. This is shown as follows:

```
proc calis nobs=932 data=Wheaton psummary;
   mstruct
      var = Anomie67 Powerless67 Anomie71 Powerless71;
   matrix _COV_ [1,1] = phi1 phi2 phi1 phi2,
                [2, ] = theta1,
                [3, ] = theta2 theta1,
                [4, ] = theta1 theta3 theta1;
   fitindex on(only)=[chisq df probchi] outfit=savefit;
run;
```

In the first entry of the MATRIX statement, the notation [1,1] represents that the parameter list specified after the equal sign starts with the [1,1] element of the _COV_ matrix and proceeds down the diagonal. In the next three entries, the notations [2,], [3,], and [4,] represent that parameter lists start with the first elements of the second, third, and fourth rows, respectively, and proceed to the next (right) elements on the same rows. See the syntax of the MATRIX statement on page 1638 for more details about this kind of shorthand notation.

This example shows how you can use the MSTRUCT modeling language to test specific covariance patterns. You need to define the parameters of the covariance patterns explicitly by the MATRIX statements. See Example 33.4 and Example 33.21 for more applications.

However, some commonly-used covariance and mean patterns are built into PROC CALIS. For these covariance and mean patterns, you can simply use the COVPATTERN= and the MEANPATTERN= options without the need to specify the parameters in the MATRIX statements. See the COVPATTERN= and the MEANPATTERN= options for the supported covariance and mean patterns. See Example 33.5 and Example 33.22 for applications.

Example 33.20: Confirmatory Factor Analysis: Cognitive Abilities

(View the complete code for this example (calex04.sas) in the example repository.)

In this example, cognitive abilities of 64 students from a middle school were measured. The fictitious data contain nine cognitive test scores. Three of the scores were for reading skills, three others were for math skills, and the remaining three were for writing skills. The covariance matrix for the nine variables was obtained. A confirmatory factor analysis with three factors was conducted. The following is the input data set:

```
title "Confirmatory Factor Analysis Using the FACTOR Modeling Language";
title2 "Cognitive Data";
data cognitive1(type=cov);
   _type_='cov';
   input _name_ $ reading1 reading2 reading3 math1 math2 math3
         writing1 writing2 writing3;
   datalines;
reading1 83.024    .      .      .      .      .      .      .      .
reading2 50.924 108.243   .      .      .      .      .      .      .
reading3 62.205  72.050 99.341   .      .      .      .      .      .
math1    22.522  22.474 25.731 82.214   .      .      .      .      .
math2    14.157  22.487 18.334 64.423 96.125   .      .      .      .
math3    22.252  20.645 23.214 49.287 58.177 88.625   .      .      .
writing1 33.433  42.474 41.731 25.318 14.254 27.370 90.734   .      .
writing2 24.147  20.487 18.034 22.106 26.105 22.346 53.891 96.543   .
```

```
writing3 13.340  20.645 23.314 19.387 28.177 38.635 55.347 52.999 98.445
;
```

Confirmatory Factor Model with Uncorrelated Factors

You first fit a confirmatory factor model with uncorrelated factors to the data, as shown in the following statements:

```
proc calis data=cognitive1 nobs=64 modification;
   factor
      Read_Factor   ===> reading1-reading3 ,
      Math_Factor   ===> math1-math3       ,
      Write_Factor  ===> writing1-writing3 ;
   pvar
      Read_Factor Math_Factor Write_Factor = 3 * 1.;
   cov
      Read_Factor Math_Factor Write_Factor = 3 * 0.;
run;
```

In the PROC CALIS statement, the number of observations is specified with the NOBS= option. With the MODIFICATION in the PROC CALIS statement, LM (Lagrange Multiplier) tests are conducted. The results of LM tests can suggest the inclusion of additional parameters for a better model fit.

The FACTOR modeling language is most handy when you specify confirmatory factor models. You use the FACTOR statement to invoke the FACTOR modeling language. Entries in the FACTOR statement are for specifying factor-variables relationships and are separated by commas. In each entry, you first specify a latent factor, followed by the right arrow sign `===>` (you can use `>`, `=>`, `==>`, or `===>`). Then you specify the observed variables that have nonzero loadings on the factor. For example, in the first entry of FACTOR statement, you specify that latent factor Read_Factor has nonzero loadings (free parameters) on variables reading1–reading3. Optionally, you can specify the parameter list after you specify the factor-variable relationships. For example, you can name the loading parameters as in the following specification:

```
factor
   Read_Factor   ===> reading1-reading3    = load1-load3;
```

This way, you name the factor loadings with parameter names load1, load2, and load3, respectively. However, in the current example, because the loading parameters are all unconstrained, you can just let PROC CALIS to generate the parameter names for you. In this example, there are three factors: Read_Factor, Math_Factor, and Write_Factor. These factors have simple cluster structures with the nine observed variables. Each observed variable has only one loading on exactly one factor.

In the PVAR statement, you can specify the variances of the factors and the error variances of the observed variables. The factor variances in this model are all fixed at 1.0 for identification purposes. You do not need to specify the error variances of the observed variables in the current model because PROC CALIS assumes these are free parameters by default.

In the COV statement, you specify that the covariances among the factors are fixed zeros. There are three covariances among the three latent factors and therefore you put `3 * 0.` for their fixed values. This means that the factors in the current model are uncorrelated. Note that you must specify uncorrelated factors explicitly in the COV statement because all latent factors are correlated by default.

In Output 33.20.1, the initial model specification is echoed in matrix form. The observed variables and factors are also displayed.

Output 33.20.1 Uncorrelated Factor Model Specification

Variables in the Model	
Variables	reading1 reading2 reading3 math1 math2 math3 writing1 writing2 writing3
Factors	Read_Factor Math_Factor Write_Factor

Number of Variables = 9
Number of Factors = 3

Initial Factor Loading Matrix	Read_Factor	Math_Factor	Write_Factor
reading1	. [_Parm1]	0	0
reading2	. [_Parm2]	0	0
reading3	. [_Parm3]	0	0
math1	0	. [_Parm4]	0
math2	0	. [_Parm5]	0
math3	0	. [_Parm6]	0
writing1	0	0	. [_Parm7]
writing2	0	0	. [_Parm8]
writing3	0	0	. [_Parm9]

Initial Factor Covariance Matrix	Read_Factor	Math_Factor	Write_Factor
Read_Factor	1.0000	0	0
Math_Factor	0	1.0000	0
Write_Factor	0	0	1.0000

Initial Error Variances		
Variable	**Parameter**	**Estimate**
reading1	_Add1	.
reading2	_Add2	.
reading3	_Add3	.
math1	_Add4	.
math2	_Add5	.
math3	_Add6	.
writing1	_Add7	.
writing2	_Add8	.
writing3	_Add9	.

NOTE:
Parameters with prefix '_Add' are added by PROC CALIS.

In the table for initial factor loading matrix, the nine loading parameters are shown to have simple cluster relations with the factors. In the table for initial factor covariance matrix, the diagonal matrix shows that the factors are not correlated. The diagonal elements are fixed at ones so that this matrix is also a correlation

matrix for the factors. In the table for initial error variances, the nine variance parameters are shown. As described previously, these error variances are generated by PROC CALIS as default parameters.

In Output 33.20.2, initial estimates are generated by the instrumental variable method and the McDonald method.

Output 33.20.2 Optimization of the Uncorrelated Factor Model: Initial Estimates

Initial Estimation Methods	
1	Instrumental Variables Method
2	McDonald Method

Optimization Start Parameter Estimates			
N	Parameter	Estimate	Gradient
1	_Parm1	7.15372	0.00851
2	_Parm2	7.80225	-0.00170
3	_Parm3	8.70856	-0.00602
4	_Parm4	7.68637	0.00272
5	_Parm5	8.01765	-0.01096
6	_Parm6	7.05012	0.00932
7	_Parm7	8.76776	-0.0009955
8	_Parm8	5.96161	-0.01335
9	_Parm9	7.23168	0.01665
10	_Add1	31.84831	-0.00179
11	_Add2	47.36790	0.0003461
12	_Add3	23.50199	0.00257
13	_Add4	23.13374	-0.0008384
14	_Add5	31.84224	0.00280
15	_Add6	38.92075	-0.00167
16	_Add7	13.86035	-0.00579
17	_Add8	61.00217	0.00115
18	_Add9	46.14784	-0.00300
Value of Objective Function = 0.9103815918			

These initial estimates turn out to be pretty good, in the sense that only three more iterations are needed to converge to the maximum likelihood estimates and the final function value 0.784 does not change much from the initial function value 0.910, as shown in Output 33.20.3.

Output 33.20.3 Optimization of the Uncorrelated Factor Model: Iteration Summary

Iteration	Restarts	Function Calls	Active Constraints	Objective Function	Objective Function Change	Max Abs Gradient Element	Lambda	Ratio Between Actual and Predicted Change
1	0	4	0	0.78792	0.1225	0.00175	0	0.932
2	0	6	0	0.78373	0.00419	0.000037	0	1.051
3	0	8	0	0.78373	5.087E-7	3.715E-9	0	1.001

Output 33.20.3 *continued*

Optimization Results			
Iterations	3	**Function Calls**	11
Jacobian Calls	5	**Active Constraints**	0
Objective Function	0.783733415	**Max Abs Gradient Element**	3.7146571E-9
Lambda	0	**Actual Over Pred Change**	1.000666095
Radius	0.0025042942		

Convergence criterion (ABSGCONV=0.00001) satisfied.

The fit summary is shown in Output 33.20.4.

Output 33.20.4 Fit of the Uncorrelated Factor Model

Fit Summary		
Modeling Info	**Number of Observations**	64
	Number of Variables	9
	Number of Moments	45
	Number of Parameters	18
	Number of Active Constraints	0
	Baseline Model Function Value	4.3182
	Baseline Model Chi-Square	272.0467
	Baseline Model Chi-Square DF	36
	Pr > Baseline Model Chi-Square	<.0001
Absolute Index	**Fit Function**	0.7837
	Chi-Square	49.3752
	Chi-Square DF	27
	Pr > Chi-Square	0.0054
	Z-Test of Wilson & Hilferty	2.5474
	Hoelter Critical N	52
	Root Mean Square Residual (RMR)	19.5739
	Standardized RMR (SRMR)	0.2098
	Goodness of Fit Index (GFI)	0.8555
Parsimony Index	**Adjusted GFI (AGFI)**	0.7592
	Parsimonious GFI	0.6416
	RMSEA Estimate	0.1147
	RMSEA Lower 90% Confidence Limit	0.0617
	RMSEA Upper 90% Confidence Limit	0.1646
	Probability of Close Fit	0.0271
	ECVI Estimate	1.4630
	ECVI Lower 90% Confidence Limit	1.2069
	ECVI Upper 90% Confidence Limit	1.8687
	Akaike Information Criterion	85.3752
	Bozdogan CAIC	142.2351
	Schwarz Bayesian Criterion	124.2351
	McDonald Centrality	0.8396
Incremental Index	**Bentler Comparative Fit Index**	0.9052
	Bentler-Bonett NFI	0.8185
	Bentler-Bonett Non-normed Index	0.8736
	Bollen Normed Index Rho1	0.7580
	Bollen Non-normed Index Delta2	0.9087
	James et al. Parsimonious NFI	0.6139

Using the chi-square model test criterion, the uncorrelated factor model should be rejected at $\alpha = 0.05$. The RMSEA estimate is 0.1147, which is not indicative of a good fit according to Browne and Cudeck (1993) Other indices might suggest only a marginal good fit. For example, Bentler's comparative fit index and Bollen nonnormed index delta2 are both above 0.90. However, many other do not attain this 0.90 level. For example, adjusted GFI is only 0.759. It is thus safe to conclude that there could be some improvements on the model fit.

The MODIFICATION option in the PROC CALIS statement has been used to request for computing the LM test indices for model modifications. The results are shown in Output 33.20.5.

Output 33.20.5 Lagrange Multiplier Tests

Rank Order of the 10 Largest LM Stat for Factor Loadings

Variable	Factor	LM Stat	Pr > ChiSq	Parm Change
writing1	Read_Factor	9.76596	0.0018	2.95010
math3	Write_Factor	3.58077	0.0585	1.89703
math1	Read_Factor	2.15312	0.1423	1.17976
writing3	Math_Factor	1.87637	0.1707	1.41298
math3	Read_Factor	1.02954	0.3103	0.95427
reading2	Write_Factor	0.91230	0.3395	0.99933
writing2	Math_Factor	0.86221	0.3531	0.95672
reading1	Write_Factor	0.63403	0.4259	0.73916
math1	Write_Factor	0.55602	0.4559	0.63906
reading2	Math_Factor	0.55362	0.4568	0.74628

Rank Order of the 3 Largest LM Stat for Covariances of Factors

Var1	Var2	LM Stat	Pr > ChiSq	Parm Change
Write_Factor	Read_Factor	8.95268	0.0028	0.44165
Write_Factor	Math_Factor	7.07904	0.0078	0.40132
Math_Factor	Read_Factor	4.61896	0.0316	0.30411

Rank Order of the 10 Largest LM Stat for Error Variances and Covariances

Error of	Error of	LM Stat	Pr > ChiSq	Parm Change
writing1	math2	5.45986	0.0195	-13.16822
writing1	math1	5.05573	0.0245	12.32431
writing3	math3	3.93014	0.0474	13.59149
writing3	math1	2.83209	0.0924	-9.86342
writing2	reading1	2.56677	0.1091	10.15901
writing2	math2	1.94879	0.1627	8.40273
writing2	reading3	1.75181	0.1856	-7.82777
writing3	reading1	1.57978	0.2088	-7.97915
writing1	reading2	1.34894	0.2455	7.77158
writing2	math3	1.11704	0.2906	-7.23762

Three different tables for ranking the LM test results are shown. In the first table, the new loading parameters that would improve the model fit the most are shown first. For example, in the first row a new factor loading of writing1 on the Read_Factor is suggested to improve the model fit the most. The 'LM Stat' value is 9.77. This is an approximation of the chi-square drop if this parameter was included in the model. The 'Pr > ChiSq' value of 0.0018 indicates a significant improvement of model fit at $\alpha = 0.05$. Nine more new loading parameters are suggested in the table, with less and less statistical significance in the change of model fit chi-square. Note that these approximate chi-squares are one-at-a-time chi-square changes. That means that the overall chi-square drop is not a simple sum of individual chi-square changes when you include two or more new parameters in the modified model.

The other two tables in Output 33.20.5 shows the new parameters in factor covariances, error variances, or error covariances that would result in a better model fit. The table for the new parameters of the factor

covariance matrix indicates that adding each of the covariances among factors might lead to a statistically significant improvement in model fit. The largest 'LM Stat' value in this table is 8.95, which is smaller than that of the largest 'LM Stat' for the factor loading parameters. Despite this, it is more reasonable to add the covariance parameters among factors first to determine whether that improves the model fit.

Confirmatory Factor Model with Correlated Factors

To fit the corresponding confirmatory factor model with correlated factors, you can remove the fixed zeros from the COV statement in the preceding specification, as shown in the following statements:

```
proc calis data=cognitive1 nobs=64 modification;
   factor
      Read_Factor   ===> reading1-reading3 ,
      Math_Factor   ===> math1-math3       ,
      Write_Factor  ===> writing1-writing3 ;
   pvar
      Read_Factor Math_Factor Write_Factor = 3 * 1.;
   cov
      Read_Factor Math_Factor Write_Factor /* = 3 * 0. */;
run;
```

In the COV statement, you comment out the fixed zeros so that the covariances among the latent factors are now free parameters. An alternative way is to delete the entire COV statement so that the covariances among factors are free parameters by the FACTOR model default.

The fit summary of the correlated factor model is shown in Output 33.20.6.

Output 33.20.6 Fit of the Correlated Factor Model

Fit Summary		
Modeling Info	**Number of Observations**	64
	Number of Variables	9
	Number of Moments	45
	Number of Parameters	21
	Number of Active Constraints	0
	Baseline Model Function Value	4.3182
	Baseline Model Chi-Square	272.0467
	Baseline Model Chi-Square DF	36
	Pr > Baseline Model Chi-Square	<.0001
Absolute Index	**Fit Function**	0.4677
	Chi-Square	29.4667
	Chi-Square DF	24
	Pr > Chi-Square	0.2031
	Z-Test of Wilson & Hilferty	0.8320
	Hoelter Critical N	78
	Root Mean Square Residual (RMR)	5.7038
	Standardized RMR (SRMR)	0.0607
	Goodness of Fit Index (GFI)	0.9109
Parsimony Index	**Adjusted GFI (AGFI)**	0.8330
	Parsimonious GFI	0.6073
	RMSEA Estimate	0.0601
	RMSEA Lower 90% Confidence Limit	0.0000
	RMSEA Upper 90% Confidence Limit	0.1244
	Probability of Close Fit	0.3814
	ECVI Estimate	1.2602
	ECVI Lower 90% Confidence Limit	1.2453
	ECVI Upper 90% Confidence Limit	1.5637
	Akaike Information Criterion	71.4667
	Bozdogan CAIC	137.8032
	Schwarz Bayesian Criterion	116.8032
	McDonald Centrality	0.9582
Incremental Index	**Bentler Comparative Fit Index**	0.9768
	Bentler-Bonett NFI	0.8917
	Bentler-Bonett Non-normed Index	0.9653
	Bollen Normed Index Rho1	0.8375
	Bollen Non-normed Index Delta2	0.9780
	James et al. Parsimonious NFI	0.5945

The model fit chi-square value is 29.47, which is about 20 less than the model with uncorrelated factors. The p-value is 0.20, indicating a satisfactory model fit. The RMSEA value is 0.06, which is close to 0.05, a value recommended as an indication of good model fit by Browne and Cudeck (1993) More fit indices that do not attain the 0.9 level with the uncorrelated factor model now have values close to or above 0.9. These include the goodness-of-fit index (GFI), McDonald centrality, Bentler-Bonnet NFI, and Bentler-Bonnet nonnormed index. By all counts, the correlated factor model is a much better fit than the uncorrelated factor model.

In Output 33.20.7, the estimation results for factor loadings are shown. All these loadings are statistically significant, indicating non-chance relationships with the factors.

Output 33.20.7 Estimation of the Factor Loading Matrix

	Factor Loading Matrix: Estimate/StdErr/t-value/p-value		
	Read_Factor	**Math_Factor**	**Write_Factor**
reading1	6.7657 1.0459 6.4689 <.0001 [_Parm01]	0	0
reading2	7.8579 1.1890 6.6090 <.0001 [_Parm02]	0	0
reading3	9.1344 1.0712 8.5269 <.0001 [_Parm03]	0	0
math1	0	7.5488 1.0128 7.4536 <.0001 [_Parm04]	0
math2	0	8.4401 1.0838 7.7874 <.0001 [_Parm05]	0
math3	0	6.8194 1.0910 6.2506 <.0001 [_Parm06]	0
writing1	0	0	7.9677 1.1254 7.0797 <.0001 [_Parm07]
writing2	0	0	6.8742 1.1986 5.7350 <.0001 [_Parm08]
writing3	0	0	7.0949 1.2057 5.8844 <.0001 [_Parm09]

In Output 33.20.8, the factor covariance matrix is shown. Because the diagonal elements are all ones, the off-diagonal elements are correlations among factors. The correlations range from 0.30–0.5. These factors are moderately correlated.

Output 33.20.8 Estimation of the Correlations of Factors

Factor Covariance Matrix: Estimate/StdErr/t-value/p-value			
	Read_Factor	Math_Factor	Write_Factor
Read_Factor	1.0000	0.3272	0.4810
		0.1311	0.1208
		2.4955	3.9813
		0.0126	<.0001
		[_Parm10]	[_Parm11]
Math_Factor	0.3272	1.0000	0.3992
	0.1311		0.1313
	2.4955		3.0417
	0.0126		0.002352
	[_Parm10]		[_Parm12]
Write_Factor	0.4810	0.3992	1.0000
	0.1208	0.1313	
	3.9813	3.0417	
	<.0001	0.002352	
	[_Parm11]	[_Parm12]	

In Output 33.20.9, the error variances for variables are shown.

Output 33.20.9 Estimation of the Error Variances

Error Variances					
Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
reading1	_Add1	37.24939	8.33997	4.4664	<.0001
reading2	_Add2	46.49695	10.69869	4.3460	<.0001
reading3	_Add3	15.90447	9.26097	1.7174	0.0859
math1	_Add4	25.22889	7.72269	3.2669	0.0011
math2	_Add5	24.89032	8.98327	2.7707	0.0056
math3	_Add6	42.12110	9.20362	4.5766	<.0001
writing1	_Add7	27.24965	10.36489	2.6290	0.0086
writing2	_Add8	49.28881	11.39812	4.3243	<.0001
writing3	_Add9	48.10684	11.48868	4.1873	<.0001

All t values except the one for reading3 are greater than 2, a value close to a critical t value at $\alpha = 0.05$. This means that the error variance for reading3 could have been zero in the population, or it could have been nonzero but the current sample just has this insignificant value by chance (that is, a Type 2 error). Further research is needed to confirm either way.

In addition to the parameter estimation results, PROC CALIS also outputs supplementary results that could be useful for interpretations. In Output 33.20.10, the squared multiple correlations and the factor scores regression coefficients are shown.

Output 33.20.10 Supplementary Estimation Results

Squared Multiple Correlations			
Variable	Error Variance	Total Variance	R-Square
reading1	37.24939	83.02400	0.5513
reading2	46.49695	108.24300	0.5704
reading3	15.90447	99.34100	0.8399
math1	25.22889	82.21400	0.6931
math2	24.89032	96.12500	0.7411
math3	42.12110	88.62500	0.5247
writing1	27.24965	90.73400	0.6997
writing2	49.28881	96.54300	0.4895
writing3	48.10684	98.44500	0.5113

Factor Scores Regression Coefficients			
	Read_Factor	Math_Factor	Write_Factor
reading1	0.0200	0.000681	0.001985
reading2	0.0186	0.000633	0.001847
reading3	0.0633	0.002152	0.006275
math1	0.001121	0.0403	0.002808
math2	0.001271	0.0457	0.003183
math3	0.000607	0.0218	0.001520
writing1	0.003195	0.002744	0.0513
writing2	0.001524	0.001309	0.0245
writing3	0.001611	0.001384	0.0259

The percentages of variance for the observed variables that can be explained by the factors are shown in the 'R-Square' column of the table for squared multiple correlations (R-squares). These R-squares can be interpreted meaningfully because there is no reciprocal relationships among variables or correlated errors in the model. All estimates of R-squares are bounded between 0 and 1.

In the table for factor scores regression coefficients, entries are coefficients for the variables you can use to create the factor scores. The larger the coefficient, the more influence of the corresponding variable for creating the factor scores. It makes intuitive sense to see the cluster pattern of these coefficients—the reading measures are more important to create the latent variable scores of Read_Factor and so on.

Example 33.21: Testing Equality of Two Covariance Matrices Using a Multiple-Group Analysis

(View the complete code for this example (calex05.sas) in the example repository.)

You can use PROC CALIS to do multiple-group or multiple-sample analysis. The groups in the analysis must be independent. In this example, a relatively simple multiple-group analysis is carried out. The covariance matrices of two independent groups are tested for equality. Hence, individual covariance matrices are actually not structured. Rather, they are constrained to be the same under the null hypothesis. That is, you want to test the following null hypothesis:

$$H_0 : \mathbf{\Sigma}_1 = \mathbf{\Sigma}_2$$

where $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$ represent the population covariance matrices of the two independent groups in question.

In PROC CALIS, you can use two different approaches to test the equality of covariance matrices. The first approach is to define an MSTRUCT model explicitly and to fit this model to the independent groups. The second approach is to use the COVPATTERN= option to invoke the required covariance structure model for the independent groups. Some standard covariance structures or patterns with the MSTRUCT modeling language are built into PROC CALIS internally. With appropriate keywords for the COVPATTERN= option, you can invoke the target built-in covariance patterns without defining the MSTRUCT model explicitly. This example considers these two approaches successively.

This example is concerned with a reaction time experiment that was conducted on two groups of individuals. One group ($N = 20$) was considered to be an expert group with prior training related to the tasks of the experiment. Another group ($N = 18$) was a control group without prior training. Three tasks of dexterity were administered to all individuals. These tasks differed by their required complexity levels of body skills. They were labeled as high, medium, and low complexities.

Apparently, the differential performance of the two groups under different task complexities was the primary research objective. In this example, however, you are interested in testing whether the groups have the same covariance matrix for the tasks. Equality of covariance matrices might be an essential assumption in some statistical tests for comparing group means. In this example, the sample covariance matrices for the two groups are stored in the data sets Expert and Novice, as shown in the following:

```
data expert(type=cov);
   input _type_ $ _name_ $ high medium low;
   datalines;
COV   high    5.88     .      .
COV   medium  2.88    7.16    .
COV   low     3.12    4.44   8.14
;

data novice(type=cov);
   input _type_ $ _name_ $ high medium low;
   datalines;
COV   high    6.42     .      .
COV   medium  1.24    8.25    .
COV   low     4.26    2.75   7.99
;
```

These data sets are read into the analysis through the GROUP statements in the following PROC CALIS specification:

```
proc calis;
   group 1 / data=expert nobs=20 label="Expert";
   group 2 / data=novice nobs=18 label="Novice";
   model 1 / groups=1,2;
      mstruct
         var=high medium low;
   fitindex NoIndexType On(only)=[chisq df probchi]
            chicorrect=eqcovmat;
   ods select ModelingInfo MSTRUCTVariables MSTRUCTCovInit Fit;
run;
```

The first GROUP statement defines `group 1` for the expert group. The second GROUP statement defines `group 2` for the novice group. You use the NOBS= option in both statements to provide the number of

observations of these groups. You use the LABEL= option in these statements to provide meaningful group labels.

The MODEL statement defines **MODEL 1**. In the analysis, this model fits to both groups 1 and 2, as indicated by the GROUPS= option of the statement. This is done to test the null hypothesis of equality of covariance matrices in the two groups. An MSTRUCT model for **MODEL 1** is defined immediately afterward. Three variables, high, medium, and low, are specified in the VAR= option of the MSTRUCT statement.

Without further specification about the MSTRUCT model, PROC CALIS assumes all non redundant elements in the covariance matrix are free parameters. This is what is required under the null hypothesis of the equality of covariance matrices in the two groups—the groups have the same covariance matrix, but the covariance matrix itself is unconstrained. Your model under the null hypothesis is now well-defined and ready to run. In addition, you use FITINDEX and ODS SELECT statements to customize or fine tune the analysis.

By using the options in the FITINDEX statement, you can customize the fit summary table and control some analytic options. In the current example, you use the NOINDEXTYPE option to suppress the printing of the index types in the fit summary table. Then, you use the ON(ONLY)= option to specify the fit indices printed in the fit summary table. In this example, you request only the model fit chi-square statistic, degrees of freedom, and the probability value of the chi-square be printed. Finally, you use the CHICORRECT=EQCOVMAT option to request a chi-square correction for the test of equality of covariance matrices. This correction is due to Box (1949) and is implemented in PROC CALIS as a built-in chi-square correction option.

In addition, because you are not interested in all displayed output for the current hypothesized model, you use the ODS SELECT statement to display only those output (or ODS tables) of interest. In this example, you request only the modeling information, the variables involved, the initial covariance matrix specification, and the fit summary table be printed. All output in PROC CALIS are named as an ODS table. To locate a particular output in PROC CALIS, you must know the corresponding ODS table name. See the section "ODS Table Names" on page 1870 for a listing of ODS tables produced by PROC CALIS.

Output 33.21.1 displays some information regarding the basic model setup.

Output 33.21.1 Modeling Information and Initial Specification

Modeling Information

Maximum Likelihood Estimation

Group	Label	Data Set	N Obs	Model	Type	Analysis
1	**Expert**	WORK.EXPERT	20	Model 1	MSTRUCT	Covariances
2	**Novice**	WORK.NOVICE	18	Model 1	MSTRUCT	Covariances

Model 1. Variables in the Model
high medium low
Number of Variables = 3

Model 1. Initial MSTRUCT _COV_ Matrix

	high	medium	low
high	.	.	.
	[_Add1]	[_Add2]	[_Add4]
medium	.	.	.
	[_Add2]	[_Add3]	[_Add5]
low	.	.	.
	[_Add4]	[_Add5]	[_Add6]

The modeling information table summarizes some basic information about the two groups. Both of them are fitted by Model 1. The next table shows the variables involved: high, medium, and low. The order of variables in this table is the same as that of the row and column variables of the covariance model matrix, which is shown next in Output 33.21.1. The parameters for the entries in the covariance matrix are shown. The names of parameters are displayed in parentheses. All these parameters are set by default and their names have the prefix _Add. No initial estimates are given as input, as indicated by the missing value '.'.

Output 33.21.2 shows the customized fit summary table, which has been much simplified for the current example due to the uses of some options in the FITINDEX statement.

Output 33.21.2 Model Fit

Fit Summary	
Chi-Square	2.4924
Chi-Square DF	6
Pr > Chi-Square	0.8693

As shown in Output 33.21.2, the chi-square test statistic is 2.4924. With six degrees of freedom, the test statistic is not significant at $\alpha = 0.01$. Therefore, the hypothesized model is supported, which means that the equality of the covariance matrices of the groups is supported.

Instead of using the MSTRUCT modeling language explicitly for defining the hypothesized covariance patterns (or structures), you can also invoke the same covariance patterns by using the COVPATTERN= option, as shown in the following statements:

```
proc calis covpattern=eqcovmat;
   var high medium low;
   group 1 / data=expert nobs=20 label="Expert";
   group 2 / data=novice nobs=18 label="Novice";
   fitindex NoIndexType On(only)=[chisq df probchi];
run;
```

The COVPATTERN=EQCOVMAT option in the PROC CALIS statement hypothesizes that the two population covariance matrices for the groups are the same. Next, you specify the set of variables in the covariance matrices in the VAR statement, followed by the specification of the data for the two groups. You use the FITINDEX statement to select a subset of fit indices to display in the output.

Output 33.21.3 shows the data sets and the corresponding MSTRUCT models that are generated by the COVPATTERN=EQCOVMAT option.

Output 33.21.3 Modeling Information with the COVPATTERN=EQCOVMAT Option

Modeling Information						
Maximum Likelihood Estimation						
Group	**Label**	**Data Set**	**N Obs**	**Model**	**Type**	**Analysis**
1	**Expert**	WORK.EXPERT	20	Model 1	MSTRUCT	Covariances
2	**Novice**	WORK.NOVICE	18	Model 2	MSTRUCT	Covariances

PROC CALIS generates Model 1 for the expert group and Model 2 for the novice group. Output 33.21.4 and Output 33.21.5 show the covariance matrices of these two models.

Output 33.21.4 Initial Specification of Model 1 for the Expert Group

Model 1. Variables in the Model
high medium low
Number of Variables = 3

Model 1. Initial MSTRUCT _COV_ Matrix			
	high	**medium**	**low**
high	. [_cov_1_1]	. [_cov_2_1]	. [_cov_3_1]
medium	. [_cov_2_1]	. [_cov_2_2]	. [_cov_3_2]
low	. [_cov_3_1]	. [_cov_3_2]	. [_cov_3_3]

Output 33.21.5 Initial Specification of Model 2 for the Novice Group

Model 2. Variables in the Model
high medium low
Number of Variables = 3

Model 2. Initial MSTRUCT _COV_ Matrix			
	high	**medium**	**low**
high	. [_cov_1_1]	. [_cov_2_1]	. [_cov_3_1]
medium	. [_cov_2_1]	. [_cov_2_2]	. [_cov_3_2]
low	. [_cov_3_1]	. [_cov_3_2]	. [_cov_3_3]

In Output 33.21.4, the covariance matrix for the expert group has three variables: high, medium, and low. The second table of Output 33.21.4 shows the parameters for the corresponding covariance matrix. PROC CALIS generates the parameter names for the elements in this covariance matrix: _cov_1_1, _cov_2_1, . . . , _cov_3_3. In Output 33.21.5, the covariance matrix for the novice group has exactly the same set of three variables: high, medium, and low. The second table of Output 33.21.5 shows the parameters for the corresponding covariance matrix. These variance and covariance parameters are exactly the same as those in Output 33.21.4, as required by the testing of equality of covariance matrices.

Output 33.21.6 shows the fit summary of the test. The test results are exactly the same as those in Output 33.21.2, as expected. The chi-square value is 2.4924. With six degrees of freedom, the test statistic is not significant at $\alpha = 0.01$. The hypothesis about the equality of the covariance matrices between the groups is supported.

Output 33.21.6 Model Fit with the COVPATTERN=EQCOVMAT Option

Fit Summary	
Chi-Square	2.4924
Chi-Square DF	6
Pr > Chi-Square	0.8693

One advantage of using the built-in covariance patterns such as the current COVPATTERN=EQCOVMAT option is that it is more efficient and less error-prone than if you specify the covariance patterns manually by using the MSTRUCT and MATRIX statements. With the COVPATTERN= option, PROC CALIS generates the correct model specification internally. Another advantage is that when applicable, PROC CALIS applies the appropriate chi-square correction to the chi-square test statistic. For the current example, PROC CALIS displays the following message in the output:

```
NOTE: The chi-square correction due to Box for testing equality of
      covariance matrices was applied. Use the CHICORRECT=0 option
      if this correction is not desirable.
```

This shows that when you use the COVPATTERN=EQCOVMAT option, an appropriate chi-square correction is applied automatically to the chi-square test statistic. To turn off this automatic chi-square, you can use the CHICORRECT=0 in the PROC CALIS statement (although this should be a rare practice with the COVPATTERN= options).

To extend the test of the equality of covariance matrices to the test of the equality of mean vectors, see Example 33.4. To extend the multiple-group analysis of covariance patterns to the multiple-group analysis of a general structural equation model, see Example 33.28.

Example 33.22: Testing Equality of Covariance and Mean Matrices between Independent Groups

(View the complete code for this example (calex13.sas) in the example repository.)

To make the specification of some standard MSTRUCT models for covariance and mean patterns more efficient, PROC CALIS defines these standard models internally. You can use two options to invoke these built-in covariance and mean patterns easily. For example, with the COVPATTERN= option, you can define the compound symmetry (COMPSYM) pattern for the covariance matrix or the equality of covariance matrices between groups (EQCOVMAT). With the MEANPATTERN= option, you can define uniform means (UNIFORM) for the mean vector or the equality of mean vectors between groups (EQMEANVEC). See the COVPATTERN= and the MEANPATTERN= options for details about the supported built-in covariance and mean patterns.

In Example 33.21, you test of the equality of covariance matrices between two groups. This example extends the application to the test of equality of mean vectors between three independent groups by using the COVPATTERN= and MEANPATTERN= options together. The “best” fit model for the data is explored. The following DATA steps define the covariance and mean matrices for the three independent groups, respectively:

```
data g1(type=corr);
   Input _type_ $ 1-8 _name_ $ 9-11 x1-x9;
   datalines;
corr    x1  1.     .       .       .       .       .       .       .       .
corr    x2 .721    1.      .       .       .       .       .       .       .
corr    x3 .676   .379     1.      .       .       .       .       .       .
corr    x4 .149   .403    .450    1.      .       .       .       .       .
corr    x5 .422   .384    .445   .411    1.      .       .       .       .
corr    x6 .343   .456    .243   .308   .531    1.      .       .       .
corr    x7 .115   .225    .201   .481   .373   .198   1.       .       .
```

```
corr    x8 .213   .237    .434   .503   .267   .333   .355   1.       .
corr    x9 .236   .257    .159   .246   .126   .235   .601   .512    1.
mean     . 21.3   22.3    17.2   23.4   22.1   15.6   18.7   20.1  19.7
std      .  1.2    1.4    .87    1.33    2.2    1.4    2.3    2.1   1.8
n        .   21     21      21     21     21     21     21     21    21
;

data g2(type=corr);
   Input _type_ $ 1-8 _name_ $ 9-11 x1-x9;
   datalines;
corr    x1  1.     .       .      .      .      .      .      .       .
corr    x2 .733    1.      .      .      .      .      .      .       .
corr    x3 .576   .388     1.     .      .      .      .      .       .
corr    x4 .209   .414    .425    1.     .      .      .      .       .
corr    x5 .412   .286    .461   .398    1.     .      .      .       .
corr    x6 .323   .399    .212   .302   .522    1.     .      .       .
corr    x7 .215   .295    .188   .467   .334   .232   1.      .       .
corr    x8 .204   .257    .462   .522   .298   .355  .372    1.       .
corr    x9 .245   .272    .177   .301   .156   .246  .578   .422     1.
mean     . 22.1   19.8    16.9   23.3   21.9   17.3   17.9   19.1  19.8
std      .  1.3    1.3    .99    1.25    2.1    1.3    2.2    2.0   1.5
n        .   22     22      22     22     22     22     22     22    22
;

data g3(type=corr);
   Input _type_ $ 1-8 _name_ $ 9-11 x1-x9;
   datalines;
corr    x1  1.     .       .      .      .      .      .      .       .
corr    x2 .699    1.      .      .      .      .      .      .       .
corr    x3 .488   .328     1.     .      .      .      .      .       .
corr    x4 .235   .398    .413    1.     .      .      .      .       .
corr    x5 .377   .265    .471   .376    1.     .      .      .       .
corr    x6 .335   .412    .265   .314   .503    1.     .      .       .
corr    x7 .243   .216    .192   .423   .369   .212   1.      .       .
corr    x8 .217   .292    .423   .525   .219   .317  .376    1.       .
corr    x9 .211   .283    .152   .285   .147   .135  .633   .579     1.
mean     . 22.2   20.9    15.4   25.1   22.6   16.3   19.3   20.2  19.5
std      .  1.5    1.0    1.04    1.5    1.9    1.6    2.4    2.2   1.6
n        .   20     20      20     20     20     20     20     20    20
;
```

Each of these data sets contains the information about the correlations, means, standard deviations, and sample sizes. Even though these data sets contain correlations, by default PROC CALIS analyzes the covariances and means.

The first hypothesis to test is the equality of covariance matrices and mean vectors:

$$H_0 : \mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \mathbf{\Sigma}_3 \text{ and } \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \boldsymbol{\mu}_3$$

where $\mathbf{\Sigma}_1$, $\mathbf{\Sigma}_2$, and $\mathbf{\Sigma}_3$ are the population covariance matrices for the three independent groups, respectively, and $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, and $\boldsymbol{\mu}_3$ are the population mean vectors for the three independent groups, respectively.

The following statements specify this test:

```
proc calis covpattern=eqcovmat meanpattern=eqmeanvec;
   var x1-x9;
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3;
   fitindex NoIndexType On(only)=[chisq df probchi rmsea aic caic sbc];
run;
```

In the PROC CALIS statement, the COVPATTERN=EQCOVMAT option specifies the same covariance matrix for the three groups and the MEANPATTERN=EQMEANVEC option specifies the same mean vector for the three groups. The VAR statement specifies that x1–9 are the variables in the hypothesis test. Next, the GROUP statements specify the data sets for the three independent groups. You use the FITINDEX statement to limit the amount of output fit statistics to the quantities specified: the chi-square test (CHISQ), the degrees of freedom (DF), the significance value of the test statistic (PROBCHI), the root mean square error of approximation (RMSEA), Akaike's information criterion (AIC), consistent Akaike's information criterion (CAIC), and Schwarz's Bayesian criterion (SBC). The first three quantities are useful for the chi-square model fit test, while the rest of the fit indices are useful for comparing competing models for the data. Because there are not many quantities in this customized fit summary table, the NOINDEXTYPE option is used to suppress the printing of the fit index types.

Output 33.22.1 shows the general modeling information, including the sample sizes, the models for the groups, the model types, and the analysis types.

Output 33.22.1 Modeling Information for Testing Equality of Covariance and Mean Matrices

Modeling Information

Maximum Likelihood Estimation

Group	Data Set	N Obs	Model	Type	Analysis
1	WORK.G1	21	Model 1	MSTRUCT	Means and Covariances
2	WORK.G2	22	Model 2	MSTRUCT	Means and Covariances
3	WORK.G3	20	Model 3	MSTRUCT	Means and Covariances

Output 33.22.2 shows the initial mean vector and the initial covariance matrix specifications for Model 1, which fits to Group 1. PROC CALIS generates the mean parameter names _mean_1, _mean_2, . . . , and _mean_9 for the nine elements in the mean vector. It also generates the covariance parameter names _cov_1_1, _cov_2_1, . . . , and _cov_9_9 for the 45 nonredundant elements in the covariance matrix.

Output 33.22.2 Initial Mean Vector and Covariance Matrix for Model 1

Model 1. Initial MSTRUCT _MEAN_ Vector		
Variable	**Parameter**	**Estimate**
x1	_mean_1	.
x2	_mean_2	.
x3	_mean_3	.
x4	_mean_4	.
x5	_mean_5	.
x6	_mean_6	.
x7	_mean_7	.
x8	_mean_8	.
x9	_mean_9	.

Model 1. Initial MSTRUCT _COV_ Matrix									
	x1	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**	**x8**	**x9**
x1	.	.	.	.	.	.	.	.	.
	[_cov_1_1]	[_cov_2_1]	[_cov_3_1]	[_cov_4_1]	[_cov_5_1]	[_cov_6_1]	[_cov_7_1]	[_cov_8_1]	[_cov_9_1]
x2	.	.	.	.	.	.	.	.	.
	[_cov_2_1]	[_cov_2_2]	[_cov_3_2]	[_cov_4_2]	[_cov_5_2]	[_cov_6_2]	[_cov_7_2]	[_cov_8_2]	[_cov_9_2]
x3	.	.	.	.	.	.	.	.	.
	[_cov_3_1]	[_cov_3_2]	[_cov_3_3]	[_cov_4_3]	[_cov_5_3]	[_cov_6_3]	[_cov_7_3]	[_cov_8_3]	[_cov_9_3]
x4	.	.	.	.	.	.	.	.	.
	[_cov_4_1]	[_cov_4_2]	[_cov_4_3]	[_cov_4_4]	[_cov_5_4]	[_cov_6_4]	[_cov_7_4]	[_cov_8_4]	[_cov_9_4]
x5	.	.	.	.	.	.	.	.	.
	[_cov_5_1]	[_cov_5_2]	[_cov_5_3]	[_cov_5_4]	[_cov_5_5]	[_cov_6_5]	[_cov_7_5]	[_cov_8_5]	[_cov_9_5]
x6	.	.	.	.	.	.	.	.	.
	[_cov_6_1]	[_cov_6_2]	[_cov_6_3]	[_cov_6_4]	[_cov_6_5]	[_cov_6_6]	[_cov_7_6]	[_cov_8_6]	[_cov_9_6]
x7	.	.	.	.	.	.	.	.	.
	[_cov_7_1]	[_cov_7_2]	[_cov_7_3]	[_cov_7_4]	[_cov_7_5]	[_cov_7_6]	[_cov_7_7]	[_cov_8_7]	[_cov_9_7]
x8	.	.	.	.	.	.	.	.	.
	[_cov_8_1]	[_cov_8_2]	[_cov_8_3]	[_cov_8_4]	[_cov_8_5]	[_cov_8_6]	[_cov_8_7]	[_cov_8_8]	[_cov_9_8]
x9	.	.	.	.	.	.	.	.	.
	[_cov_9_1]	[_cov_9_2]	[_cov_9_3]	[_cov_9_4]	[_cov_9_5]	[_cov_9_6]	[_cov_9_7]	[_cov_9_8]	[_cov_9_9]

Although not shown here, the initial mean vector and covariance matrices for Models 2 and 3 are exactly the same as those shown in Output 33.22.2, as required by the equality of covariance and mean matrices in the null hypothesis H_0.

Output 33.22.3 shows the customized fit summary table. The chi-square test statistic is 203.2605. The degrees of freedom is 108 and the p-value is less than 0.0001. Therefore, the hypothesis H_0 of equality in covariance and mean matrices is rejected for the three independent groups. The RMSEA index is much greater than 0.05, which does not indicate a good model fit. Other fit indices such as AIC, CAIC, and SBC are not interpreted for the fit of the model itself, but are useful for comparing competing models in the later discussion.

Output 33.22.3 Fit Summary for Testing H_0: Equality of Covariance and Mean Matrices

Fit Summary	
Chi-Square	203.2605
Chi-Square DF	108
Pr > Chi-Square	<.0001
RMSEA Estimate	0.2100
Akaike Information Criterion	311.2605
Bozdogan CAIC	480.9897
Schwarz Bayesian Criterion	426.9897

A less restrictive hypothesis is now considered. This hypothesis states the equality of covariance matrices only:

$$H_1 : \boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}_3 (\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \text{ and } \boldsymbol{\mu}_3 \text{unconstrained})$$

H_1 differs from H_0 in that the population means in H_1 are not constrained. To test this hypothesis, you need to change the MEANPATTERN= option to use the SATURATED keyword, as shown in the following statements:

```
proc calis covpattern=eqcovmat meanpattern=saturated;
   var x1-x9;
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3;
   fitindex NoIndexType On(only)=[chisq df probchi rmsea aic caic sbc];
run;
```

Output 33.22.4 shows the results of the testing H_1.

Output 33.22.4 Fit Summary for Testing H_1: Equality of Covariance Matrices but Unconstrained Means

Fit Summary	
Chi-Square	26.7897
Chi-Square DF	90
Pr > Chi-Square	1.0000
RMSEA Estimate	0.0000
Akaike Information Criterion	170.7897
Bozdogan CAIC	397.0954
Schwarz Bayesian Criterion	325.0954

The chi-square test statistic is 26.7897 (df = 90, p = 1.000). You cannot reject this null hypothesis about the equality of the population covariance matrices. The RMSEA value is virtually zero, which indicates a perfect fit. Comparing the models under H_0 and H_1, it is clear that the three groups are significantly different with regard to their mean vectors. By relaxing all the equality constraints on the means in H_0, H_1 is derived and is supported by the chi-square test. In addition, the RMSEA value for the model under H_1 is perfect. Because lower values of AIC, CAIC, and SBC values indicate better model fit (with the model complexity taken into account), these indices in Output 33.22.3 and Output 33.22.4 support that the model under H_1 is better than H_0.

However, in getting a superior model fit, H_1 might have relaxed more constraints than absolutely necessary

for an optimal fit. That is, it might be possible to impose equality constraints on only some (but not all, as in H_1) of the means to reach the same or even better model fit (by the RMSEA, AIC, CAIC, or SBC criterion) than the model under H_1. But how can you determine this set of constrained means?

To answer this question, you conduct an exploratory analysis of the data by using some model modification techniques. Models established from exploratory analysis should be validated by external data in the future. However, this example demonstrates the exploratory part only.

Beginning with the model under H_0, you can manually take away some particular constraints on the means and explore whether the revised model improves the fit. If the revised model fits better, you can repeat the process until you cannot improve more. Ultimately, you might be able to find the "best" model between the models specified under H_0 and H_1. Such an exploratory analysis is laborious, considering the vast possibilities of constraints on the nine variable means in three independent groups that you could attempt to release. Fortunately, PROC CALIS provides some model modification statistics, called the LM (Lagrange multiplier) statistics, to assist this kind of exploratory analysis.

The following statements specify the model under H_0, but now with the MODIFICATION option added to the PROC CALIS statement:

```
proc calis covpattern=eqcovmat meanpattern=eqmeanvec modification;
   var x1-x9;
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3;
   fitindex NoIndexType On(only)=[chisq df probchi rmsea aic caic sbc];
run;
```

The MODIFICATION option requests the so-called LM (Lagrange multiplier) statistics for releasing the parameter constraints. These constraints include the cross-group or within-group constraints and the fixed values in the model. For the model under H_0, the covariances and the means are all constrained across groups. These are the equality constraints that you would like to release to obtain a better model fit. Output 33.22.5 shows the results of the LM statistics for releasing these equality constraints in variances, covariances, and means.

Output 33.22.5 Lagrange Multiplier Statistics for Releasing the Equality Constraints

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
_cov_1_1	1	COV	x1	x1	0.01137	0.9151	0.0178	-0.0355
	2	COV	x1	x1	1.00150	0.3169	0.1729	-0.3212
	3	COV	x1	x1	1.28632	0.2567	-0.1818	0.3923
_cov_2_1	1	COV	x2	x1	2.19353	0.1386	0.2038	-0.4076
	2	COV	x2	x1	0.77014	0.3802	-0.1253	0.2327
	3	COV	x2	x1	0.36128	0.5478	-0.0796	0.1718
_cov_2_2	1	COV	x2	x2	3.12065	0.0773	-0.4344	0.8687
	2	COV	x2	x2	0.05704	0.8112	-0.0609	0.1132
	3	COV	x2	x2	4.14151	0.0418	0.4817	-1.0395
_cov_3_1	1	COV	x3	x1	0.00672	0.9347	0.00888	-0.0178
	2	COV	x3	x1	2.23758	0.1347	-0.1681	0.3122
	3	COV	x3	x1	2.10455	0.1469	0.1512	-0.3264
_cov_3_2	1	COV	x3	x2	2.18538	0.1393	-0.1940	0.3881
	2	COV	x3	x2	3.14532	0.0761	0.2416	-0.4487
	3	COV	x3	x2	0.10264	0.7487	-0.0405	0.0874
_cov_3_3	1	COV	x3	x3	1.56813	0.2105	0.1815	-0.3630
	2	COV	x3	x3	0.66118	0.4161	0.1223	-0.2272
	3	COV	x3	x3	4.42160	0.0355	-0.2934	0.6332
_cov_4_1	1	COV	x4	x1	0.31691	0.5735	-0.0667	0.1333
	2	COV	x4	x1	0.32615	0.5679	0.0702	-0.1304
	3	COV	x4	x1	0.0002277	0.9880	-0.00172	0.00371
_cov_4_2	1	COV	x4	x2	0.73377	0.3917	0.1242	-0.2484
	2	COV	x4	x2	0.53196	0.4658	-0.1097	0.2038
	3	COV	x4	x2	0.01445	0.9043	-0.0168	0.0362
_cov_4_3	1	COV	x4	x3	0.0000258	0.9959	0.000547	-0.00109
	2	COV	x4	x3	0.24892	0.6178	-0.0558	0.1036
	3	COV	x4	x3	0.25646	0.6126	0.0525	-0.1134
_cov_4_4	1	COV	x4	x4	0.04412	0.8336	0.0361	-0.0722
	2	COV	x4	x4	0.52198	0.4700	0.1288	-0.2392
	3	COV	x4	x4	0.90948	0.3403	-0.1577	0.3403
_cov_5_1	1	COV	x5	x1	0.0008607	0.9766	-0.00477	0.00953
	2	COV	x5	x1	0.01238	0.9114	0.0188	-0.0348
	3	COV	x5	x1	0.00712	0.9328	-0.0132	0.0285
_cov_5_2	1	COV	x5	x2	0.10637	0.7443	-0.0649	0.1297
	2	COV	x5	x2	0.00631	0.9367	-0.0164	0.0304
	3	COV	x5	x2	0.16971	0.6804	0.0789	-0.1702
_cov_5_3	1	COV	x5	x3	0.06645	0.7966	-0.0385	0.0771
	2	COV	x5	x3	0.0008275	0.9771	0.00446	-0.00829
	3	COV	x5	x3	0.05370	0.8167	0.0334	-0.0720
_cov_5_4	1	COV	x5	x4	0.24212	0.6227	0.0809	-0.1617
	2	COV	x5	x4	0.04459	0.8328	-0.0360	0.0669
	3	COV	x5	x4	0.07959	0.7779	-0.0446	0.0963
_cov_5_5	1	COV	x5	x5	0.01778	0.8939	-0.0431	0.0862
	2	COV	x5	x5	0.08223	0.7743	-0.0962	0.1787
	3	COV	x5	x5	0.18417	0.6678	0.1336	-0.2883
_cov_6_1	1	COV	x6	x1	0.29558	0.5867	-0.0721	0.1442

Output 33.22.5 *continued*

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
	2	COV	x6	x1	0.26589	0.6061	-0.0710	0.1318
	3	COV	x6	x1	1.16570	0.2803	0.1378	-0.2974
_cov_6_2	1	COV	x6	x2	0.00228	0.9619	-0.00780	0.0156
	2	COV	x6	x2	1.00319	0.3165	0.1697	-0.3152
	3	COV	x6	x2	0.95767	0.3278	-0.1538	0.3320
_cov_6_3	1	COV	x6	x3	1.39116	0.2382	0.1513	-0.3027
	2	COV	x6	x3	0.08741	0.7675	-0.0394	0.0731
	3	COV	x6	x3	0.79586	0.3723	-0.1102	0.2378
_cov_6_4	1	COV	x6	x4	0.46031	0.4975	-0.0947	0.1894
	2	COV	x6	x4	0.04254	0.8366	0.0299	-0.0555
	3	COV	x6	x4	0.22665	0.6340	0.0640	-0.1381
_cov_6_5	1	COV	x6	x5	0.14991	0.6986	-0.0700	0.1399
	2	COV	x6	x5	0.04723	0.8280	0.0408	-0.0757
	3	COV	x6	x5	0.02874	0.8654	0.0295	-0.0636
_cov_6_6	1	COV	x6	x6	0.22550	0.6349	0.1079	-0.2158
	2	COV	x6	x6	0.04390	0.8340	0.0494	-0.0918
	3	COV	x6	x6	0.48451	0.4864	-0.1523	0.3286
_cov_7_1	1	COV	x7	x1	0.50774	0.4761	0.1203	-0.2406
	2	COV	x7	x1	0.01246	0.9111	-0.0196	0.0363
	3	COV	x7	x1	0.36926	0.5434	-0.0988	0.2131
_cov_7_2	1	COV	x7	x2	0.01235	0.9115	0.0228	-0.0455
	2	COV	x7	x2	0.16400	0.6855	-0.0861	0.1598
	3	COV	x7	x2	0.09159	0.7622	0.0597	-0.1288
_cov_7_3	1	COV	x7	x3	0.16844	0.6815	-0.0644	0.1288
	2	COV	x7	x3	0.15095	0.6976	0.0633	-0.1175
	3	COV	x7	x3	0.0003079	0.9860	0.00265	-0.00572
_cov_7_4	1	COV	x7	x4	0.22542	0.6349	-0.0776	0.1551
	2	COV	x7	x4	0.00754	0.9308	0.0147	-0.0273
	3	COV	x7	x4	0.15376	0.6950	0.0617	-0.1331
_cov_7_5	1	COV	x7	x5	0.07831	0.7796	-0.0631	0.1262
	2	COV	x7	x5	0.07552	0.7835	0.0643	-0.1195
	3	COV	x7	x5	3.293E-6	0.9986	0.000394	-0.00085
_cov_7_6	1	COV	x7	x6	0.13810	0.7102	0.0726	-0.1452
	2	COV	x7	x6	0.0001086	0.9917	0.00211	-0.00392
	3	COV	x7	x6	0.14999	0.6985	-0.0729	0.1572
_cov_7_7	1	COV	x7	x7	0.09334	0.7600	0.1051	-0.2101
	2	COV	x7	x7	0.00128	0.9714	0.0128	-0.0237
	3	COV	x7	x7	0.11994	0.7291	-0.1147	0.2474
_cov_8_1	1	COV	x8	x1	0.04800	0.8266	0.0353	-0.0706
	2	COV	x8	x1	0.19725	0.6569	0.0743	-0.1379
	3	COV	x8	x1	0.45888	0.4981	-0.1051	0.2268
_cov_8_2	1	COV	x8	x2	0.13689	0.7114	0.0727	-0.1453
	2	COV	x8	x2	0.31671	0.5736	-0.1147	0.2130
	3	COV	x8	x2	0.04084	0.8398	0.0382	-0.0825
_cov_8_3	1	COV	x8	x3	0.37615	0.5397	-0.0904	0.1808
	2	COV	x8	x3	0.00452	0.9464	-0.0103	0.0191

Output 33.22.5 *continued*

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
	3	COV	x8	x3	0.47678	0.4899	0.0980	-0.2114
_cov_8_4	1	COV	x8	x4	0.00989	0.9208	0.0150	-0.0300
	2	COV	x8	x4	0.01001	0.9203	0.0157	-0.0291
	3	COV	x8	x4	0.04138	0.8388	-0.0296	0.0638
_cov_8_5	1	COV	x8	x5	0.01378	0.9066	-0.0267	0.0533
	2	COV	x8	x5	0.03154	0.8590	-0.0419	0.0778
	3	COV	x8	x5	0.09063	0.7634	0.0659	-0.1421
_cov_8_6	1	COV	x8	x6	0.0007193	0.9786	0.00510	-0.0102
	2	COV	x8	x6	0.01293	0.9095	0.0224	-0.0417
	3	COV	x8	x6	0.02067	0.8857	-0.0263	0.0568
_cov_8_7	1	COV	x8	x7	0.16543	0.6842	0.0952	-0.1904
	2	COV	x8	x7	0.29902	0.5845	-0.1328	0.2467
	3	COV	x8	x7	0.02206	0.8819	0.0335	-0.0722
_cov_8_8	1	COV	x8	x8	0.00581	0.9392	0.0244	-0.0487
	2	COV	x8	x8	0.00694	0.9336	-0.0276	0.0513
	3	COV	x8	x8	0.0000660	0.9935	0.00250	-0.00539
_cov_9_1	1	COV	x9	x1	0.19272	0.6607	-0.0532	0.1063
	2	COV	x9	x1	0.01910	0.8901	-0.0174	0.0323
	3	COV	x9	x1	0.34408	0.5575	0.0684	-0.1476
_cov_9_2	1	COV	x9	x2	0.09017	0.7640	-0.0446	0.0892
	2	COV	x9	x2	0.26496	0.6067	0.0794	-0.1474
	3	COV	x9	x2	0.04994	0.8232	-0.0320	0.0690
_cov_9_3	1	COV	x9	x3	0.44236	0.5060	0.0758	-0.1516
	2	COV	x9	x3	0.12761	0.7209	-0.0422	0.0784
	3	COV	x9	x3	0.09470	0.7583	-0.0338	0.0728
_cov_9_4	1	COV	x9	x4	0.04619	0.8298	0.0260	-0.0520
	2	COV	x9	x4	0.22996	0.6316	-0.0602	0.1117
	3	COV	x9	x4	0.07502	0.7842	0.0319	-0.0688
_cov_9_5	1	COV	x9	x5	0.02807	0.8669	0.0279	-0.0557
	2	COV	x9	x5	0.0006585	0.9795	-0.00443	0.00823
	3	COV	x9	x5	0.02058	0.8859	-0.0230	0.0496
_cov_9_6	1	COV	x9	x6	0.03989	0.8417	-0.0282	0.0563
	2	COV	x9	x6	0.15069	0.6979	-0.0568	0.1055
	3	COV	x9	x6	0.36051	0.5482	0.0815	-0.1759
_cov_9_7	1	COV	x9	x7	0.03398	0.8537	-0.0284	0.0567
	2	COV	x9	x7	0.05802	0.8097	0.0385	-0.0714
	3	COV	x9	x7	0.00362	0.9520	-0.00891	0.0192
_cov_9_8	1	COV	x9	x8	0.06050	0.8057	-0.0391	0.0781
	2	COV	x9	x8	0.56151	0.4537	0.1235	-0.2294
	3	COV	x9	x8	0.26945	0.6037	-0.0794	0.1713
_cov_9_9	1	COV	x9	x9	0.13296	0.7154	-0.0655	0.1310
	2	COV	x9	x9	0.00130	0.9712	-0.00673	0.0125
	3	COV	x9	x9	0.16526	0.6844	0.0703	-0.1517
_mean_1	1	MEAN	x1		11.09173	0.0009	0.3453	-0.6906
	2	MEAN	x1		1.21196	0.2709	-0.1184	0.2200
	3	MEAN	x1		5.04550	0.0247	-0.2242	0.4838

Output 33.22.5 *continued*

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
_mean_2	1	MEAN	x2		21.46921	<.0001	-0.5837	1.1675
	2	MEAN	x2		15.27776	<.0001	0.5110	-0.9490
	3	MEAN	x2		0.47301	0.4916	0.0834	-0.1800
_mean_3	1	MEAN	x3		4.41967	0.0355	-0.2034	0.4067
	2	MEAN	x3		6.37770	0.0116	-0.2535	0.4708
	3	MEAN	x3		22.27732	<.0001	0.4395	-0.9485
_mean_4	1	MEAN	x4		3.26860	0.0706	0.1904	-0.3807
	2	MEAN	x4		0.03260	0.8567	0.0197	-0.0366
	3	MEAN	x4		4.06935	0.0437	-0.2045	0.4413
_mean_5	1	MEAN	x5		0.22210	0.6374	-0.0681	0.1362
	2	MEAN	x5		1.50172	0.2204	0.1837	-0.3412
	3	MEAN	x5		0.60673	0.4360	-0.1083	0.2338
_mean_6	1	MEAN	x6		1.61486	0.2038	0.1539	-0.3078
	2	MEAN	x6		6.72912	0.0095	-0.3260	0.6055
	3	MEAN	x6		1.88248	0.1701	0.1600	-0.3452
_mean_7	1	MEAN	x7		0.14035	0.7079	-0.0558	0.1116
	2	MEAN	x7		0.11034	0.7398	0.0514	-0.0954
	3	MEAN	x7		0.00153	0.9688	0.00560	-0.0121
_mean_8	1	MEAN	x8		0.12603	0.7226	-0.0510	0.1019
	2	MEAN	x8		1.96607	0.1609	0.2089	-0.3880
	3	MEAN	x8		1.16200	0.2811	-0.1490	0.3215
_mean_9	1	MEAN	x9		0.05301	0.8179	0.0248	-0.0496
	2	MEAN	x9		0.97965	0.3223	-0.1106	0.2054
	3	MEAN	x9		0.61083	0.4345	0.0810	-0.1748

To use the results of this table, you look for parameters that have large LM statistics (in the LM Stat column). Equivalently, you can look for parameters that have small p-values (in the Pr > ChiSq column). Loosely speaking, an LM statistic estimates the reduction of model fit chi-square statistic if you release the constraint on the corresponding parameter. The p-value indicates whether the improvement would be significant. Therefore, releasing those parameters with a high LM statistic and small p-value would be the key to model improvements. Bear in mind that the LM statistics are linear approximations and they might not be very accurate as estimates of the actual model improvement, which could only be accessed when you refit the model with the particular constraint released. Nonetheless, the LM statistics could still be very useful because they show which constraints could potentially improve the model the most.

Output 33.22.5 shows the results from releasing the constraints on the variances and covariances first. Each constrained element of the covariance matrix has three rows, respectively, for the three models (or groups). For example, the first parameter is _cov_1_1, which is the same variance parameter for x1 in the three models. The first row shows that if you release the variance of x1 in Model 1 from the constraint (while keeping the variances of x1 being constrained between Models 2 and 3), the LM statistic is 0.01127, and the corresponding p-value is 0.9155. This means that the model fit improvement would be very small and so you do not expect a significant model fit improvement by releasing this constraint. The columns entitled "Changes" show the estimated parameter changes in the original parameters (that is, _cov_1_1 for Models 2 and 3) and in the released parameter (that is, the new parameter for the variance of x1 in Model 1) if you

release the corresponding equality constraint. These two "Changes" columns are not very useful for the present purpose.

Looking through the results for the variance and covariance constraints, you can see that almost all the associated *p*-values are large (that is, as compared with the conventional 0.05 level for significance). Therefore, all these constraints on variances and covariances would not improve the model fit significantly. In contrast, the constraints on the means show that several of them could be released for a sizable model fit improvement. The largest LM statistic in the table is the one for _mean_3 in Model 3. The LM statistic is 22.27678 and its corresponding *p*-value is less than 0.0001. This means that if the mean of x3 in Model 3 were not constrained with the means of x3 in Models 1 and 2, you would have expected a reduction in the model fit chi-square statistic that is estimated at 22.27678. Other notable LM statistics are those for _mean_1 in Model 1, _mean_2 in Model 1 or 2, and _mean_6 in Model 2.

Two important points are noted about the use of the LM statistics. First, the LM statistics are not additive. You cannot expect that the total reduction in model fit chi-square for releasing a particular set of parameter constraints is the sum of the corresponding LM statistics. Second, once you release a particular constraint and refit the model, the LM statistics in the revised model might not follow the same pattern as those LM statistics in the original model. Basically, these are due to the nonlinearity of the fit function and the dependence of the parameter estimates. Therefore, in order to find the best model for the data, it would be more sensible to adopt a one-at-a-time approach to release the constraints. That is, you release one constraint at a time and refit the model to see if you can release more constraints to improve the model fit.

According to the results of LM statistics in Output 33.22.5, you first release the constraint on the _mean_3 parameter, which is for the mean of x3 in Model 3. The following statements fit such a model:

```
proc calis modification;
   var x1-x9;
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3;
   model 1 / group = 1;
      mstruct;
      matrix _cov_  = cov01-cov45;
      matrix _mean_ = mean1-mean9;
   model 2 / group = 2;
      refmodel 1;
   model 3 / group = 3;
      refmodel 1;
      renameparm mean3=mean3_mdl3;
   fitindex NoIndexType On(only)=[chisq df probchi rmsea aic caic sbc];
run;
```

Because the revised model is no longer a supported built-in MSTRUCT model, you cannot use the MEANPATTERN= or the COVPATTERN= options any more. Instead, you now use the MSTRUCT modeling language to specify the covariance and mean patterns. Model 1, which fits to Group 1, is an MSTRUCT model with variance and covariance parameters cov01–cov45 and mean parameters mean1–mean9. Model 2, which fits to Group 2, refers to the specifications of Model 1, as indicated in a REFMODEL statement. Hence, Model 1 and Model 2 are completely constrained in variances, covariances, and means. Model 3, which fits to Group 3, also refers to the specifications of Model 1, as indicated in another REFMODEL statement. However, the RENAMEPARM statement renames the parameter mean3 in the reference model (that is, Model 1) to a new name mean3_mdl3. As a results, all variance, covariance, and mean parameters except one in Model 3 are constrained to be the same as those in Model 1. The mean of x3 in Model 3 is the

only parameter that is not constrained with any other parameters. This forms the first revised model from H_0. The MODIFICATION option is specified again to determine whether a further model fit improvement is possible.

Output 33.22.6 shows the modeling information of the first revised model. It shows that Models 2 and 3 make references to Model 1. Therefore, parameters between models are constrained by referencing.

Output 33.22.6 Modeling Information for The First Revised Model

Modeling Information
Maximum Likelihood Estimation

Group	Data Set	N Obs	Model	Type	Base Model	Analysis
1	WORK.G1	21	Model 1	MSTRUCT		Means and Covariances
2	WORK.G2	22	Model 2	MSTRUCT	Model 1	Means and Covariances
3	WORK.G3	20	Model 3	MSTRUCT	Model 1	Means and Covariances

Output 33.22.7 shows the initial specifications of the means, variances, and covariances in Model 1.

Output 33.22.7 Initial Mean Vector and Covariance Matrix for Model 1 in the First Revised Model

Model 1.
Initial MSTRUCT _MEAN_
Vector

Variable	Parameter	Estimate
x1	mean1	.
x2	mean2	.
x3	mean3	.
x4	mean4	.
x5	mean5	.
x6	mean6	.
x7	mean7	.
x8	mean8	.
x9	mean9	.

Model 1. Initial MSTRUCT _COV_ Matrix

	x1	x2	x3	x4	x5	x6	x7	x8	x9
x1	. [cov01]	. [cov02]	. [cov04]	. [cov07]	. [cov11]	. [cov16]	. [cov22]	. [cov29]	. [cov37]
x2	. [cov02]	. [cov03]	. [cov05]	. [cov08]	. [cov12]	. [cov17]	. [cov23]	. [cov30]	. [cov38]
x3	. [cov04]	. [cov05]	. [cov06]	. [cov09]	. [cov13]	. [cov18]	. [cov24]	. [cov31]	. [cov39]
x4	. [cov07]	. [cov08]	. [cov09]	. [cov10]	. [cov14]	. [cov19]	. [cov25]	. [cov32]	. [cov40]
x5	. [cov11]	. [cov12]	. [cov13]	. [cov14]	. [cov15]	. [cov20]	. [cov26]	. [cov33]	. [cov41]
x6	. [cov16]	. [cov17]	. [cov18]	. [cov19]	. [cov20]	. [cov21]	. [cov27]	. [cov34]	. [cov42]
x7	. [cov22]	. [cov23]	. [cov24]	. [cov25]	. [cov26]	. [cov27]	. [cov28]	. [cov35]	. [cov43]
x8	. [cov29]	. [cov30]	. [cov31]	. [cov32]	. [cov33]	. [cov34]	. [cov35]	. [cov36]	. [cov44]
x9	. [cov37]	. [cov38]	. [cov39]	. [cov40]	. [cov41]	. [cov42]	. [cov43]	. [cov44]	. [cov45]

Output 33.22.8 shows the initial specifications of the means in Model 2. The mean parameters in Model 2 are exactly the same as those in Model 1, as shown in Output 33.22.7. The variance and covariance parameters in Model 2 are also exactly the same as those in Model 1, but are not shown here to conserve space.

Output 33.22.8 Initial Mean Vector for Model 2 in the First Revised Model

Model 2. Initial MSTRUCT _MEAN_ Vector		
Variable	**Parameter**	**Estimate**
x1	mean1	.
x2	mean2	.
x3	mean3	.
x4	mean4	.
x5	mean5	.
x6	mean6	.
x7	mean7	.
x8	mean8	.
x9	mean9	.

Output 33.22.9 shows the initial specifications of the means in Model 3. All but one mean parameter in Model 3 are exactly the same as those in Models 1 and 2, as shown in Output 33.22.7 and Output 33.22.8, respectively. The mean for x3 in Model 3 is mean3_mdl3, which is now a distinct parameter, and therefore it is not constrained with any other parameters in the first or the second models for Groups 1 or 2. However, the variance and covariance parameters in Model 3 are exactly the same as those in Model 1. They are not shown here to conserve space.

Output 33.22.9 Initial Mean Vector for Model 3 in the First Revised Model

Model 3. Initial MSTRUCT _MEAN_ Vector		
Variable	**Parameter**	**Estimate**
x1	mean1	.
x2	mean2	.
x3	mean3_mdl3	.
x4	mean4	.
x5	mean5	.
x6	mean6	.
x7	mean7	.
x8	mean8	.
x9	mean9	.

Output 33.22.10 shows the fit summary of the first revised model. The model fit chi-square is 148.8865, which drops quite a bit from the original model under H_0. The p-value of the model fit chi-square is 0.0046, which is statistically significant. The RMSEA value is 0.1399, which is also a sizable improvement. All the AIC, CAIC, and SBC values are reduced, indicating better model fit than the model under H_0.

Output 33.22.10 Fit Summary for the First Revised Model

Fit Summary	
Chi-Square	148.8865
Chi-Square DF	107
Pr > Chi-Square	0.0046
RMSEA Estimate	0.1399
Akaike Information Criterion	258.8865
Bozdogan CAIC	431.7589
Schwarz Bayesian Criterion	376.7589

Output 33.22.11 shows the LM statistics for releasing the equality constraints in the first revised model. Almost all of the results for the variance and covariance constraints are omitted because their LM statistics are not significant. However, Output 33.22.11 shows all the LM statistics for releasing the constraints in means. The mean of x2 in Model 2 has the largest LM statistic at 26.25044.

Output 33.22.11 LM Statistics for Releasing the Equality Constraints in the First Revised Model

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
cov01	1	COV	x1	x1	0.64999	0.4201	0.1050	-0.2100
	2	COV	x1	x1	0.41758	0.5181	0.0874	-0.1622
	3	COV	x1	x1	2.18923	0.1390	-0.1855	0.4004
					.			
					.			
					.			
mean1	1	MEAN	x1		9.26674	0.0023	0.2872	-0.5745
	2	MEAN	x1		3.00599	0.0830	-0.1702	0.3160
	3	MEAN	x1		2.13787	0.1437	-0.1481	0.3196
mean2	1	MEAN	x2		26.25115	<.0001	-0.6568	1.3135
	2	MEAN	x2		12.34638	0.0004	0.4674	-0.8680
	3	MEAN	x2		2.52683	0.1119	0.1962	-0.4234
mean3	1	MEAN	x3		0.58891	0.4428	-0.0787	0.0827
	2	MEAN	x3		0.58891	0.4428	0.0827	-0.0787
mean4	1	MEAN	x4		6.59009	0.0103	0.2746	-0.5493
	2	MEAN	x4		0.51343	0.4737	0.0796	-0.1478
	3	MEAN	x4		11.61610	0.0007	-0.3586	0.7739
mean5	1	MEAN	x5		0.52967	0.4667	-0.1042	0.2084
	2	MEAN	x5		0.22294	0.6368	0.0702	-0.1304
	3	MEAN	x5		0.06889	0.7930	0.0374	-0.0807
mean6	1	MEAN	x6		1.16656	0.2801	0.1270	-0.2540
	2	MEAN	x6		5.29599	0.0214	-0.2810	0.5218
	3	MEAN	x6		1.69412	0.1931	0.1518	-0.3275
mean7	1	MEAN	x7		0.03791	0.8456	-0.0291	0.0582
	2	MEAN	x7		0.44510	0.5047	0.1036	-0.1923
	3	MEAN	x7		0.23804	0.6256	-0.0704	0.1520
mean8	1	MEAN	x8		0.39420	0.5301	-0.0883	0.1765
	2	MEAN	x8		0.24231	0.6225	0.0719	-0.1335
	3	MEAN	x8		0.01951	0.8889	0.0200	-0.0431
mean9	1	MEAN	x9		0.00156	0.9685	0.00423	-0.00846
	2	MEAN	x9		1.06866	0.3012	-0.1150	0.2136
	3	MEAN	x9		1.05210	0.3050	0.1065	-0.2297

You now modify the preceding statements to specify the second revised model, as shown in the following statements:

```
proc calis modification;
   var x1-x9;
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3;
   model 1 / group = 1;
      mstruct;
      matrix _cov_  = cov01-cov45;
      matrix _mean_ = mean1-mean9;
```

```
   model 2 / group = 2;
      refmodel 1;
      renameparm mean2=mean2_new;    /* constraint a */
   model 3 / group = 3;
      refmodel 1;
      renameparm mean2=mean2_new,    /* constraint a */
                 mean3=mean3_mdl3;
   fitindex NoIndexType On(only)=[chisq df probchi rmsea aic caic sbc];
run;
```

This second revised model must not constrain the mean of x2 in Model 1 with any parameters. A straightforward way to do this is to rename the mean2 parameter to a unique name in Model 1. However, for the current specification it is more convenient to rename the mean2 parameter in Models 2 and 3 to another name. In the specification of the second revised model, Models 2 and 3 still make references to Model 1. However, in the respective RENAMEPARM statements, both Model 2 and 3 rename the mean2 parameter that is referenced from Model 1 to the new name mean2_new. This way the mean for x2 in Model 1 is not constrained with the means of x2 in Models 2 and 3. But the means for x2 in Models 2 and 3 are still constrained to be equal by the same parameter mean2_new. Output 33.22.12 shows the fit summary of the second revised model.

Output 33.22.12 Fit Summary for the Second Revised Model

Fit Summary	
Chi-Square	86.3927
Chi-Square DF	106
Pr > Chi-Square	0.9183
RMSEA Estimate	0.0000
Akaike Information Criterion	198.3927
Bozdogan CAIC	374.4083
Schwarz Bayesian Criterion	318.4083

Again, a sizable improvement over the first revised model is shown in the second revised model. The model fit chi-square statistic is no longer significant ($p = 0.9183$), and the RMSEA value is perfect at 0. Large drops in the AIC, CAIC, and SBC values are also observed.

Output 33.22.13 suggests that the mean of x6 in Model 2 (which has the largest LM statistic at 11.41243) could be released from the equality constraints to achieve the largest model improvement over the current model.

Output 33.22.13 LM Statistics for Releasing the Equality Constraints in the Second Revised Model

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
cov01	1	COV	x1	x1	2.77024	0.0960	0.1384	-0.2770
	2	COV	x1	x1	0.28728	0.5920	0.0462	-0.0860
	3	COV	x1	x1	5.00087	0.0253	-0.1791	0.3864
					.			
					.			
					.			
mean1	1	MEAN	x1		2.75437	0.0970	0.1646	-0.3292
	2	MEAN	x1		3.21093	0.0731	-0.1511	0.2806
	3	MEAN	x1		0.24923	0.6176	0.0424	-0.0915
mean3	1	MEAN	x3		0.74338	0.3886	-0.0877	0.0934
	2	MEAN	x3		0.74338	0.3886	0.0934	-0.0877
mean4	1	MEAN	x4		6.17449	0.0130	0.2672	-0.5343
	2	MEAN	x4		0.02087	0.8851	-0.0146	0.0272
	3	MEAN	x4		4.71344	0.0299	-0.2072	0.4470
mean5	1	MEAN	x5		1.65517	0.1983	-0.1853	0.3706
	2	MEAN	x5		1.16118	0.2812	0.1606	-0.2982
	3	MEAN	x5		0.04040	0.8407	0.0287	-0.0618
mean6	1	MEAN	x6		5.03834	0.0248	0.2712	-0.5423
	2	MEAN	x6		11.41247	0.0007	-0.4217	0.7831
	3	MEAN	x6		1.51175	0.2189	0.1460	-0.3150
mean7	1	MEAN	x7		0.32382	0.5693	-0.0853	0.1706
	2	MEAN	x7		0.82184	0.3646	0.1410	-0.2619
	3	MEAN	x7		0.12513	0.7235	-0.0512	0.1104
mean8	1	MEAN	x8		2.39207	0.1220	-0.2210	0.4420
	2	MEAN	x8		1.58292	0.2083	0.1867	-0.3467
	3	MEAN	x8		0.08641	0.7688	0.0427	-0.0922
mean9	1	MEAN	x9		0.00682	0.9342	0.00886	-0.0177
	2	MEAN	x9		1.20949	0.2714	-0.1225	0.2274
	3	MEAN	x9		1.10016	0.2942	0.1089	-0.2349
mean2_new	2	MEAN	x2		4.47814	0.0343	0.2983	-0.2661
	3	MEAN	x2		4.47814	0.0343	-0.2661	0.2983

The process of model refitting should now become familiar. You modify the previous model to release the constraint on the mean of x6 in Model 2. As a result, the third revised model is specified by the following statements:

```
proc calis modification;
   var x1-x9;
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3;
   model 1 / group = 1;
      mstruct;
      matrix _cov_  = cov01-cov45;
      matrix _mean_ = mean1-mean9;
```

```
   model 2 / group = 2;
      refmodel 1;
      renameparm mean2=mean2_new,     /* constraint a */
                 mean6=mean6_mdl2;
   model 3 / group = 3;
      refmodel 1;
      renameparm mean2=mean2_new,     /* constraint a */
                 mean3=mean3_mdl3;
   fitindex NoIndexType On(only)=[chisq df probchi rmsea aic caic sbc];
run;
```

The only modification from the previous specification is to rename mean6 to mean6_mdl2 in the RENAMEPARM statement of Model 2. Output 33.22.14 shows the model fit summary of the third revised model.

Output 33.22.14 Fit Summary for the Third Revised Model

Fit Summary	
Chi-Square	68.7869
Chi-Square DF	105
Pr > Chi-Square	0.9976
RMSEA Estimate	0.0000
Akaike Information Criterion	182.7869
Bozdogan CAIC	361.9456
Schwarz Bayesian Criterion	304.9456

The model improvement over the second revised model is still notable in the third revised model. The chi-square value drops about 20 points in the third revised model. The AIC, CAIC, and the SBC values are reduced notably, though not as impressively as with the previous improvements.

Output 33.22.15 suggests that the mean of x4 in Model 1 (which has the largest LM statistic at 7.01946) could be released from the equality constraint to improve model fit further.

Output 33.22.15 LM Statistics for Releasing the Equality Constraints in the Third Revised Model

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
cov01	1	COV	x1	x1	2.43374	0.1187	0.1342	-0.2684
	2	COV	x1	x1	0.19037	0.6626	0.0390	-0.0723
	3	COV	x1	x1	4.11402	0.0425	-0.1679	0.3625
					.			
					.			
					.			
mean1	1	MEAN	x1		6.15722	0.0131	0.2550	-0.5101
	2	MEAN	x1		6.05791	0.0138	-0.2109	0.3917
	3	MEAN	x1		0.29302	0.5883	0.0463	-0.0999
mean3	1	MEAN	x3		2.89780	0.0887	-0.1796	0.1889
	2	MEAN	x3		2.89780	0.0887	0.1889	-0.1796
mean4	1	MEAN	x4		7.01915	0.0081	0.2850	-0.5701
	2	MEAN	x4		0.04916	0.8245	-0.0226	0.0419
	3	MEAN	x4		5.05102	0.0246	-0.2148	0.4635
mean5	1	MEAN	x5		0.21231	0.6450	-0.0672	0.1345
	2	MEAN	x5		0.07502	0.7842	-0.0443	0.0822
	3	MEAN	x5		0.55031	0.4582	0.1059	-0.2285
mean6	1	MEAN	x6		0.07013	0.7911	0.0503	-0.0486
	3	MEAN	x6		0.07013	0.7911	-0.0486	0.0503
mean7	1	MEAN	x7		0.98902	0.3200	-0.1513	0.3025
	2	MEAN	x7		2.42355	0.1195	0.2463	-0.4575
	3	MEAN	x7		0.34231	0.5585	-0.0858	0.1850
mean8	1	MEAN	x8		1.58481	0.2081	-0.1786	0.3572
	2	MEAN	x8		0.81634	0.3663	0.1347	-0.2502
	3	MEAN	x8		0.14503	0.7033	0.0549	-0.1184
mean9	1	MEAN	x9		0.13504	0.7133	0.0399	-0.0797
	2	MEAN	x9		2.54369	0.1107	-0.1796	0.3335
	3	MEAN	x9		1.61681	0.2035	0.1337	-0.2885
mean2_new	2	MEAN	x2		3.21203	0.0731	0.2484	-0.2280
	3	MEAN	x2		3.21203	0.0731	-0.2280	0.2484

To make the mean parameter for x4 in Model 1 unique, the mean parameters for x4 in Models 2 and 3 are renamed from mean4 to mean4_new, as shown in the following statements:

```
proc calis modification;
   var x1-x9;
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3;
   model 1 / group = 1;
      mstruct;
      matrix _cov_  = cov01-cov45;
      matrix _mean_ = mean1-mean9;
   model 2 / group = 2;
      refmodel 1;
```

```
      renameparm mean2=mean2_new,     /* constraint a */
                 mean4=mean4_new,     /* constraint b */
                 mean6=mean6_mdl2;
   model 3 / group = 3;
      refmodel 1;
      renameparm mean2=mean2_new,     /* constraint a */
                 mean3=mean3_mdl3,
                 mean4=mean4_new;     /* constraint b */
   fitindex NoIndexType On(only)=[chisq df probchi rmsea aic caic sbc];
run;
```

This forms the fourth revised model. Output 33.22.16 shows the fit summary of this revised model. Again, the chi-square, AIC, CAIC, and SBC values all show improvements, as compared with the third revised model. However, the improvements do seem to slow down. For example, the CAIC value drops from 361.95 to the current value at 358.43—a mere 3 points reduction. The SBC value drops from 304.95 to the current value at 300.43—a mere 4 points reduction. These small reductions indicate that you might soon reach the point that no more model fit improvement would be possible with additional release of parameter constraints.

Output 33.22.16 Fit Summary for the Fourth Revised Model

Fit Summary	
Chi-Square	60.1265
Chi-Square DF	104
Pr > Chi-Square	0.9998
RMSEA Estimate	0.0000
Akaike Information Criterion	176.1265
Bozdogan CAIC	358.4283
Schwarz Bayesian Criterion	300.4283

Output 33.22.17 suggests that the mean of x1 in Model 1 (which has the largest LM statistic at 6.45785) could be released from the equality constraint to achieve the largest model improvement over the current model.

Output 33.22.17 LM Statistics for Releasing the Equality Constraints in the Fourth Revised Model

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
cov01	1	COV	x1	x1	2.60531	0.1065	0.1376	-0.2751
	2	COV	x1	x1	0.28122	0.5959	0.0469	-0.0871
	3	COV	x1	x1	4.75001	0.0293	-0.1788	0.3859
					.			
					.			
					.			
mean1	1	MEAN	x1		6.45761	0.0110	0.2616	-0.5232
	2	MEAN	x1		5.00991	0.0252	-0.1921	0.3568
	3	MEAN	x1		0.05931	0.8076	0.0209	-0.0452
mean3	1	MEAN	x3		1.53300	0.2157	-0.1298	0.1406
	2	MEAN	x3		1.53300	0.2157	0.1406	-0.1298
mean5	1	MEAN	x5		0.09749	0.7549	-0.0457	0.0913
	2	MEAN	x5		0.19688	0.6572	-0.0716	0.1330
	3	MEAN	x5		0.56583	0.4519	0.1071	-0.2310
mean6	1	MEAN	x6		0.35800	0.5496	0.1141	-0.1113
	3	MEAN	x6		0.35800	0.5496	-0.1113	0.1141
mean7	1	MEAN	x7		4.53367E-6	0.9983	0.000350	-0.00070
	2	MEAN	x7		0.96363	0.3263	0.1572	-0.2920
	3	MEAN	x7		1.00890	0.3152	-0.1486	0.3208
mean8	1	MEAN	x8		0.20289	0.6524	-0.0676	0.1351
	2	MEAN	x8		0.12445	0.7243	0.0525	-0.0974
	3	MEAN	x8		0.00590	0.9388	0.0110	-0.0237
mean9	1	MEAN	x9		0.05893	0.8082	-0.0271	0.0542
	2	MEAN	x9		1.63723	0.2007	-0.1448	0.2689
	3	MEAN	x9		2.44241	0.1181	0.1652	-0.3565
mean2_new	2	MEAN	x2		3.05068	0.0807	0.2396	-0.2246
	3	MEAN	x2		3.05068	0.0807	-0.2246	0.2396
mean4_new	2	MEAN	x4		1.81983	0.1773	0.2306	-0.2003
	3	MEAN	x4		1.81983	0.1773	-0.2003	0.2306

To make the mean parameter for x1 in Model 1 unique, the mean parameters for x1 in Models 2 and 3 are renamed from mean1 to mean1_new, as shown in the following statements:

```
proc calis modification;
   var x1-x9;
   group 1 / data=g1;
   group 2 / data=g2;
   group 3 / data=g3;
   model 1 / group = 1;
      mstruct;
      matrix _cov_  = cov01-cov45;
      matrix _mean_ = mean1-mean9;
   model 2 / group = 2;
      refmodel 1;
      renameparm mean1=mean1_new,     /* constraint c */
```

```
                 mean2=mean2_new,     /* constraint a */
                 mean4=mean4_new,     /* constraint b */
                 mean6=mean6_mdl2;
   model 3 / group = 3;
      refmodel 1;
      renameparm mean1=mean1_new,    /* constraint c */
                 mean2=mean2_new,    /* constraint a */
                 mean3=mean3_mdl3,
                 mean4=mean4_new;    /* constraint b */
   fitindex NoIndexType On(only)=[chisq df probchi rmsea aic caic sbc];
run;
```

This forms the fifth revised model. Output 33.22.18 shows the fit summary of the fifth revised model. Again, the chi-square, AIC, CAIC, and SBC values all show improvements, as compared with the fourth revised model. However, the improvements slow down even more. For example, the CAIC value drops from 358.43 to the current value at 356.32. The SBC value drops from 300.43 to the current value at 297.32. Because the model fit does not improve much, this is the point where you would cease to release more equality constraints for improving the model fit.

Output 33.22.18 Fit Summary for the Fifth Revised Model

Fit Summary	
Chi-Square	52.8821
Chi-Square DF	103
Pr > Chi-Square	1.0000
RMSEA Estimate	0.0000
Akaike Information Criterion	170.8821
Bozdogan CAIC	356.3270
Schwarz Bayesian Criterion	297.3270

Output 33.22.19 does not suggest the release of any equality constraints on the means, because all the *p*-values for the LM statistics are not significant (that is, all are greater than 0.05). Therefore, the same suggestion from examining the model fit improvements of the fifth revised model echoes here: this is the point that the "best" model for the data is found.

Output 33.22.19 LM Statistics for Releasing the Equality Constraints in the Fifth Revised Model

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
cov01	1	COV	x1	x1	4.06279	0.0438	0.1590	-0.3180
	2	COV	x1	x1	0.48735	0.4851	0.0571	-0.1061
	3	COV	x1	x1	7.60892	0.0058	-0.2095	0.4520
					.			
					.			
					.			
mean3	1	MEAN	x3		0.08363	0.7724	-0.0312	0.0382
	2	MEAN	x3		0.08363	0.7724	0.0382	-0.0312
mean5	1	MEAN	x5		0.02394	0.8770	0.0229	-0.0458
	2	MEAN	x5		0.47076	0.4926	-0.1113	0.2067
	3	MEAN	x5		0.26015	0.6100	0.0728	-0.1571
mean6	1	MEAN	x6		0.97521	0.3234	0.1893	-0.1892
	3	MEAN	x6		0.97521	0.3234	-0.1892	0.1893
mean7	1	MEAN	x7		0.03746	0.8465	-0.0319	0.0638
	2	MEAN	x7		1.10428	0.2933	0.1683	-0.3126
	3	MEAN	x7		0.79474	0.3727	-0.1321	0.2851
mean8	1	MEAN	x8		0.86792	0.3515	-0.1426	0.2852
	2	MEAN	x8		0.47493	0.4907	0.1038	-0.1928
	3	MEAN	x8		0.03722	0.8470	0.0276	-0.0595
mean9	1	MEAN	x9		0.12190	0.7270	0.0401	-0.0801
	2	MEAN	x9		2.66768	0.1024	-0.1869	0.3472
	3	MEAN	x9		1.78114	0.1820	0.1417	-0.3058
mean1_new	2	MEAN	x1		1.28034	0.2578	-0.1794	0.1359
	3	MEAN	x1		1.28034	0.2578	0.1359	-0.1794
mean2_new	2	MEAN	x2		2.53131	0.1116	0.2117	-0.2112
	3	MEAN	x2		2.53131	0.1116	-0.2112	0.2117
mean4_new	2	MEAN	x4		2.25832	0.1329	0.2558	-0.2253
	3	MEAN	x4		2.25832	0.1329	-0.2253	0.2558

To see where the fifth revised model (equality in the covariance matrix and partial equality in the means) stands between the models under H_0 (equality in the covariance and mean matrices) and H_1 (equality in the covariance matrix only), the following table shows the fit statistics of these three models:

	H_0	**"Fifth"**	H_1
Chi-square	203.2605	52.8821	26.7897
Chi-square DF	108	103	90
Pr > chi-square	<0.0001	1.0000	1.0000
RMSEA estimate	0.2100	0.0000	0.0000
Akaike information criterion	311.2605	170.8821	170.7897
Bozdogan CAIC	480.9898	356.3270	397.0954
Schwarz Bayesian criterion	426.9898	297.3270	325.0954

The fifth revised model is labeled "Fifth" in the table. Compared with the model under H_0, the fifth revised model is clearly superior. It uses only five more parameters (or five fewer degrees of freedom), but the improvement in the model fit chi-square and the RMSEA value are huge. The AIC, CAIC, and SBC are also much better.

Compared with the model under H_1, the fifth revised model appears to be inferior in only the chi-square model fit statistic, although both models already have the highest possible p-value at 1.000 and smallest possible RMSEA value at 0. However, the model under H_1 uses 13 more parameters (or it has 13 fewer degrees of freedom), and hence it is more complex. In fact, because the model fit chi-square value does not take model complexity into account, it is often criticized as the basis for choosing competing models for the data. In contrast, the AIC, CAIC, and SBC measures take model complexity into account, and they are more reasonable as the basis for choosing competing models. Although the AIC values for the fifth revised model and the model under H_1 are very close, the CAIC and SBC values clearly favor the fifth revised model. Therefore, according to the CAIC and SBC criteria, the fifth revised model, which is a model with partial equality constraints on the means, is actually better than the model with all the means being unconstrained (that is, under H_1) for the current data with three independent groups.

Example 33.23: Illustrating Various General Modeling Languages

(View the complete code for this example (calex06.sas) in the example repository.)

In PROC CALIS, you can use many different modeling languages to specify the same model. The choice of modeling language depends on personal preferences and the purposes of the analysis. See the section "Which Modeling Language?" on page 1530 for guidance. In this example, the data and the model in Example 33.17 are used to illustrate how a particular model can be specified by various general modeling languages.

RAM Model Specification

In Example 33.17, you use the PATH modeling language to specify the model because of its close resemblance to the path diagram. In this example, you consider another modeling language of PROC CALIS that is also closely related to the path diagram representation of structural equation models. The so-called RAM model language has syntax that represents the single- and double-headed paths (or arrows) in the path diagram. However, unlike the PATH modeling language, the RAM modeling language is matrix-based. The following statements show how you can specify the same path model with the RAM model specification for the data in Example 33.17:

```
proc calis nobs=932 data=Wheaton;
   ram
      var =  Anomie67     /* 1 */
             Powerless67  /* 2 */
             Anomie71     /* 3 */
             Powerless71  /* 4 */
             Education    /* 5 */
             SEI          /* 6 */
             Alien67      /* 7 */
             Alien71      /* 8 */
             SES,         /* 9 */
      _A_    1   7   1.0,
      _A_    2   7   0.833,
```

```
      _A_    3   8   1.0,
      _A_    4   8   0.833,
      _A_    5   9   1.0,
      _A_    6   9   lambda,
      _A_    7   9   gamma1,
      _A_    8   9   gamma2,
      _A_    8   7   beta,
      _P_    1   1   theta1,
      _P_    2   2   theta2,
      _P_    3   3   theta1,
      _P_    4   4   theta2,
      _P_    5   5   theta3,
      _P_    6   6   theta4,
      _P_    7   7   psi1,
      _P_    8   8   psi2,
      _P_    9   9   phi,
      _P_    1   3   theta5,
      _P_    2   4   theta5;
run;
```

In the RAM model for covariance structure analysis, you have two important matrices to specify. The first one is the _A_ matrix, which is for the specification of the single-headed paths (arrows) in the path diagram. The second one is the _P_ matrix, which is for the specification of the double-headed paths (arrows) in the path diagram. Hence, to specify the RAM model is much like mapping the path diagram arrows into the parameter of the RAM model matrices.

In the RAM statement, you can specify the variables in the model in the VAR= option. The VAR= list contains all observed and latent variables in your path diagram (without the use of error terms). Although you can specify the variables in the VAR= list in any order you like, the variable order in the list is also the order of variables in the RAM model matrices. In VAR= list of the RAM statement, you put comments to note the order of the variables.

After you specify the variable list, you can specify the model parameter locations in the RAM statement entries. In the first nine entries, you specify the single-headed paths by mapping them into the elements of the _A_ matrix of the RAM model. For example, the first entry represents the single-headed path of variable 1 (Anomie67) from variable 7 (Alien67). The corresponding path effect or coefficient is fixed at 1, which is also the value for _A_[1,7]. Another example is the ninth path entry. You specify a single-headed path of variable 8 (Alien71) from variable 7 (Alien67). The corresponding path effect or coefficient is a free parameter named beta, which is also the parameter for _A_[8,7]. Hence, you can specify all single-headed paths in the path diagram as elements in the _A_ matrix of the RAM model.

To facilitate the comparisons between the RAN and PATH modeling languages, the PATH model specification in Example 33.17 for the same data is reproduced in the following:

```
proc calis nobs=932 data=Wheaton plots=residuals;
   path
      Anomie67   Powerless67  <===  Alien67   = 1.0  0.833,
      Anomie71   Powerless71  <===  Alien71   = 1.0  0.833,
      Education  SEI          <===  SES       = 1.0  lambda,
      Alien67    Alien71      <===  SES       = gamma1 gamma2,
      Alien71                 <===  Alien67   = beta;
   pvar
      Anomie67     = theta1,
```

```
      Powerless67  = theta2,
      Anomie71     = theta1,
      Powerless71  = theta2,
      Education    = theta3,
      SEI          = theta4,
      Alien67      = psi1,
      Alien71      = psi2,
      SES          = phi;
   pcov
      Anomie67    Anomie71    = theta5,
      Powerless67 Powerless71 = theta5;
run;
```

It is clear that each of the path entries specified in the PATH statement corresponds to an matrix element entry of the _A_ matrix in the RAM statement. How about the specifications of the double-headed arrows in the path diagram? Do the RAM and PATH model specifications correspond to each other?

The answer is yes. In the PATH modeling language, you specify all double-headed arrows in the path diagram as entries either in the PVAR or PCOV statement. In the RAM modeling language, you specify the corresponding entries as matrix element entries of the _P_ matrix in the RAM statement. For example, the error variance of Anomie67 is a parameter called _Variabletheta1 in the PVAR statement of the PATH model. You specify the same parameter for the _P_[1,1] element in an entry of the RAM statement. Another example is the error covariance between Powerless67 and Powerless71. You specify this a parameter called theta5 in the last entry of the PCOV statement in the PATH model. You specify the same parameter for the _P_[2,4] element in the last entry of the RAM statement. Therefore, it is not difficult to find that the specifications in the PATH and the RAM model have some kind of one-to-one correspondence.

Output 33.23.1 shows the RAM model estimates for the Wheaton data. These RAM model estimates match the set of estimates using the PATH model specification, as shown in Output 33.17.11.

Output 33.23.1 RAM Model Estimates

RAM Pattern and Estimates

Matrix	Row		Column		Parameter	Estimate	Standard Error	t Value	Pr > |t|
_A_ (1)	Anomie67	1	Alien67	7		1.00000			
	Powerless67	2	Alien67	7		0.83300			
	Anomie71	3	Alien71	8		1.00000			
	Powerless71	4	Alien71	8		0.83300			
	Education	5	SES	9		1.00000			
	SEI	6	SES	9	lambda	5.36883	0.43371	12.3788	<.0001
	Alien67	7	SES	9	gamma1	-0.62994	0.05634	-11.1809	<.0001
	Alien71	8	SES	9	gamma2	-0.24086	0.05489	-4.3884	<.0001
	Alien71	8	Alien67	7	beta	0.59312	0.04678	12.6788	<.0001
_P_ (2)	Anomie67	1	Anomie67	1	theta1	3.60796	0.20092	17.9572	<.0001
	Powerless67	2	Powerless67	2	theta2	3.59488	0.16448	21.8556	<.0001
	Anomie71	3	Anomie71	3	theta1	3.60796	0.20092	17.9572	<.0001
	Powerless71	4	Powerless71	4	theta2	3.59488	0.16448	21.8556	<.0001
	Education	5	Education	5	theta3	2.99366	0.49861	6.0040	<.0001
	SEI	6	SEI	6	theta4	259.57639	18.31151	14.1756	<.0001
	Alien67	7	Alien67	7	psi1	5.67046	0.42301	13.4050	<.0001
	Alien71	8	Alien71	8	psi2	4.51479	0.33532	13.4639	<.0001
	SES	9	SES	9	phi	6.61634	0.63914	10.3519	<.0001
	Anomie67	1	Anomie71	3	theta5	0.90580	0.12167	7.4447	<.0001
	Powerless67	2	Powerless71	4	theta5	0.90580	0.12167	7.4447	<.0001

LINEQS Model Specification

Another way to specify the model in Example 33.17 is to use the LINEQS modeling language, which is shown in the following:

```
proc calis nobs=932 data=Wheaton;
   lineqs
      Anomie67     = 1.0    * f_Alien67 + e1,
      Powerless67  = 0.833  * f_Alien67 + e2,
      Anomie71     = 1.0    * f_Alien71 + e3,
      Powerless71  = 0.833  * f_Alien71 + e4,
      Education    = 1.0    * f_SES     + e5,
      SEI          = lambda * f_SES     + e6,
      f_Alien67    = gamma1 * f_SES     + d1,
      f_Alien71    = gamma2 * f_SES     + beta * f_Alien67 + d2;
   variance
      E1           = theta1,
      E2           = theta2,
      E3           = theta1,
      E4           = theta2,
      E5           = theta3,
      E6           = theta4,
      D1           = psi1,
      D2           = psi2,
      f_SES        = phi;
```

```
    cov
       E1  E3        = theta5,
       E2  E4        = theta5;
run;
```

As compared with the PATH and RAM modeling languages, the most distinct feature of the LINEQS modeling language is the explicit use of error terms in equation specifications. In the LINEQS statement, you specify exactly one equation for each endogenous variable. In each equation, you list an endogenous variable on the left-hand-side of the equation and all its predictors on the right-hand-side of the equation. You must also include an error term in each equation. Because each endogenous variable in the LINEQS statement can only be specified in exactly one equation, the number of equations in the LINEQS model and the number of paths in the corresponding path diagram do not match necessarily. In this example, there are eight equations in the LINEQS statement, but there are nine paths in the corresponding path diagram.

In addition, in the LINEQS model, you need to follow a convention of naming latent variables. For latent variables that are neither errors nor disturbances, you must use either the 'F' or 'f' prefix. For error terms, you must use either the 'E' or 'e' prefix. For disturbances, you must use either the 'D' or 'd' prefix. However, in the PATH or RAM model specification, no such convention is imposed. For example, f_Alien67, f_Alien71, and f_SES are latent factors in the LINEQS model. They are not error terms, and so they must start with the 'f' prefix. However, this prefix is not needed in the PATH or RAM model. Furthermore, there are no explicit error terms that need to be specified in the PATH or RAM model, let alone specific prefixes for the error terms.

The PVAR statement in the PATH model is replaced with the VARIANCE statement in the LINEQS model, and the PCOV statement with the COV statement. The PVAR and PCOV statements in the PATH model are for the partial variance and partial covariance specifications. The partial variance or covariance concepts are used in the PATH or RAM model specification because error terms are not named explicitly. Specification of error variances in the PATH and RAM model is conceptualized as the specification of the partial variances of the corresponding variables. But in the LINEQS model, because errors or disturbances are named explicitly as *exogenous* variables, the partial variance or covariance concepts are no longer necessary. Instead, you specify the variances of the error terms directly, which reflects the conceptualization behind the VARIANCE statement of the LINEQS modeling language. Similarly, you use the COV, but not PCOV, statement in the LINEQS modeling language because you can specify the covariances among variables or error terms without using the partial covariance conceptualization.

In this example, the variances of the errors ("E"-variables) and disturbances ("D"-variables) specified in the VARIANCE statement of the LINEQS model correspond to the partial variances of the endogenous variables specified in the PVAR statement of the PATH model. Similarly, covariances of errors specified in the COV statement of the LINEQS model correspond to the partial covariances of endogenous variables specified in the PCOV statement of the PATH model. The estimation results of the LINEQS model are shown in Output 33.23.2. Again, they are essentially the same estimates obtained from the PATH model specified in Example 33.17, as shown in Output 33.17.11.

Output 33.23.2 LINEQS Model Estimates

Linear Equations

Anomie67	=	1.0000		f_Alien67	+	1.0000		e1			
Powerless67	=	0.8330		f_Alien67	+	1.0000		e2			
Anomie71	=	1.0000		f_Alien71	+	1.0000		e3			
Powerless71	=	0.8330		f_Alien71	+	1.0000		e4			
Education	=	1.0000		f_SES	+	1.0000		e5			
SEI	=	5.3688	(**)	f_SES	+	1.0000		e6			
f_Alien67	=	-0.6299	(**)	f_SES	+	1.0000		d1			
f_Alien71	=	-0.2409	(**)	f_SES	+	0.5931	(**)	f_Alien67	+	1.0000	d2

Effects in Linear Equations

Variable	Predictor	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Anomie67	f_Alien67		1.00000			
Powerless67	f_Alien67		0.83300			
Anomie71	f_Alien71		1.00000			
Powerless71	f_Alien71		0.83300			
Education	f_SES		1.00000			
SEI	f_SES	lambda	5.36883	0.43371	12.3788	<.0001
f_Alien67	f_SES	gamma1	-0.62994	0.05634	-11.1809	<.0001
f_Alien71	f_SES	gamma2	-0.24086	0.05489	-4.3884	<.0001
f_Alien71	f_Alien67	beta	0.59312	0.04678	12.6788	<.0001

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	**e1**	theta1	3.60796	0.20092	17.9572	<.0001
	e2	theta2	3.59488	0.16448	21.8556	<.0001
	e3	theta1	3.60796	0.20092	17.9572	<.0001
	e4	theta2	3.59488	0.16448	21.8556	<.0001
	e5	theta3	2.99366	0.49861	6.0040	<.0001
	e6	theta4	259.57639	18.31151	14.1756	<.0001
Disturbance	**d1**	psi1	5.67046	0.42301	13.4050	<.0001
	d2	psi2	4.51479	0.33532	13.4639	<.0001
Latent	**f_SES**	phi	6.61634	0.63914	10.3519	<.0001

Covariances Among Exogenous Variables

Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
e1	**e3**	theta5	0.90580	0.12167	7.4447	<.0001
e2	**e4**	theta5	0.90580	0.12167	7.4447	<.0001

LISMOD Specification

You can also specify general structural models by using the LISMOD modeling language. See the section "The LISMOD Model and Submodels" on page 1752 for details.

To use the LISMOD modeling language, you must recognize four types of variables in the model. The $\boldsymbol{\eta}$-variables (eta-variables) are latent factors that are endogenous, or predicted by other latent factors. The $\boldsymbol{\xi}$-variables (xi-variables) are exogenous latent variables that are not predicted by any other variables. The $\mathbf{y}$-variables are manifest variables that are indicators of the $\boldsymbol{\eta}$-variables, and the $\mathbf{x}$-variables are manifest variables that are indicators of the $\boldsymbol{\xi}$-variables. In this example, Alien67 and Alien71 are the $\boldsymbol{\eta}$-variables, and SES is the $\boldsymbol{\xi}$-variable in the model. Manifest indicators for Alien67 and Alien71 include Anomie67, Powerless67, Anomie71, and Powerless71, which are the $\mathbf{y}$-variables. Manifest indicators for SES include Education and SEI, which are the $\mathbf{x}$-variables.

After defining these four types of variables, the parameters of the model are defined as entries in the model matrices. The _LAMBDAY_, _LAMBDAX_, _GAMMA_, and _BETA_ are matrices for the path coefficients or effects. The _THETAY, _THETAX_, _PSI_, and _PHI_ are matrices for the variances and covariances.

The following is the LISMOD specification for the model in Example 33.17:

```
proc calis nobs=932 data=Wheaton;
   lismod
      yvar   = Anomie67 Powerless67 Anomie71 Powerless71,
      xvar   = Education SEI,
      etavar = Alien67  Alien71,
      xivar  = SES;
   matrix _LAMBDAY_
      [1,1]  = 1,
      [2,1]  = 0.833,
      [3,2]  = 1,
      [4,2]  = 0.833;
   matrix _LAMBDAX_
      [1,1]  = 1,
      [2,1]  = lambda;
   matrix _GAMMA_
      [1,1]  = gamma1,
      [2,1]  = gamma2;
   matrix _BETA_
      [2,1]  = beta;
   matrix _THETAY_
      [1,1]  = theta1-theta2 theta1-theta2,
      [3,1]  = theta5,
      [4,2]  = theta5;
   matrix _THETAX_
      [1,1]  = theta3-theta4;
   matrix _PSI_
      [1,1]  = psi1-psi2;
   matrix _PHI_
      [1,1]  = phi;
run;
```

In the LISMOD statement, you specify the four lists of variables in the model. The orders of the variables in these lists define the order of the row and column variables in the model matrices, of which the parameter locations are specified in the MATRIX statements.

The estimated model is divided into three conceptual parts. The first part is the measurement model that relates the $\boldsymbol{\eta}$-variables with the $\mathbf{y}$-variables, as shown in Output 33.23.3:

Output 33.23.3 LISMOD Model Measurement Model for the $\boldsymbol{\eta}$-Variables

LAMBDAY Matrix: Estimate/StdErr/t-value/p-value	Alien67	Alien71
Anomie67	1.0000	0
Powerless67	0.8330	0
Anomie71	0	1.0000
Powerless71	0	0.8330

THETAY Matrix: Estimate/StdErr/t-value/p-value	Anomie67	Powerless67	Anomie71	Powerless71
Anomie67	3.6080 0.2009 17.9572 <.0001 [theta1]	0	0.9058 0.1217 7.4447 <.0001 [theta5]	0
Powerless67	0	3.5949 0.1645 21.8556 <.0001 [theta2]	0	0.9058 0.1217 7.4447 <.0001 [theta5]
Anomie71	0.9058 0.1217 7.4447 <.0001 [theta5]	0	3.6080 0.2009 17.9572 <.0001 [theta1]	0
Powerless71	0	0.9058 0.1217 7.4447 <.0001 [theta5]	0	3.5949 0.1645 21.8556 <.0001 [theta2]

The _LAMBDAY_ matrix contains the coefficients or effects of the $\boldsymbol{\eta}$-variables on the $\mathbf{y}$-variables. All these estimates are fixed constants as specified. The _THETAY_ matrix contains the error variances and covariances for the $\mathbf{y}$-variables. Three free parameters are located in this matrix: theta1, theta2, and theta5.

The second part of the estimated model is the measurement model that relates the $\boldsymbol{\xi}$-variable with the $\mathbf{x}$-variables, as shown in Output 33.23.4:

Output 33.23.4 LISMOD Model Measurement Model for the ξ-Variables

LAMBDAX Matrix: Estimate/StdErr/t-value/p-value	SES
Education	1.0000
SEI	5.3688 0.4337 12.3788 <.0001 [lambda]

THETAX Matrix: Estimate/StdErr/t-value/p-value	Education	SEI
Education	2.9937 0.4986 6.0040 <.0001 [theta3]	0
SEI	0	259.5764 18.3115 14.1756 <.0001 [theta4]

The _LAMBDAX_ matrix contains the coefficients or effects of the ξ-variable SES on the $\mathbf{x}$-variables. The effect of SES on Education is fixed at one. The effect of SES on SEI is represented by the free parameter lambda, which is estimated at 5.3688. The _THETAX_ matrix contains the error variances and covariances for the $\mathbf{x}$-variables. Two free parameters are located in this matrix: theta3 and theta4.

The last part of the estimated model is the structural model that relates the latent variables $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$, as shown in Output 33.23.5:

Output 33.23.5 LISMOD Structural Model for the Latent Variables

BETA Matrix: Estimate/StdErr/t-value/p-value	Alien67	Alien71
Alien67	0	0
Alien71	0.5931 0.0468 12.6788 <.0001 [beta]	0

Output 33.23.5 *continued*

GAMMA Matrix: Estimate/StdErr/t-value/p-value	SES
Alien67	-0.6299 0.0563 -11.1809 <.0001 [gamma1]
Alien71	-0.2409 0.0549 -4.3884 <.0001 [gamma2]

PSI Matrix: Estimate/StdErr/t-value/p-value	Alien67	Alien71
Alien67	5.6705 0.4230 13.4050 <.0001 [psi1]	0
Alien71	0	4.5148 0.3353 13.4639 <.0001 [psi2]

PHI Matrix: Estimate/StdErr/t-value/p-value	SES
SES	6.6163 0.6391 10.3519 <.0001 [phi]

The _BETA_ matrix contains effects of η-variables on themselves. In the current example, there is only one such effect. The effect of Alien67 on Alien71 is represented by the free parameter beta. The _GAMMA_ matrix contains effects of the ξ-variable, which is SES in this example, on the η-variables Alien67 on Alien71. These effects are represented by the free parameters gamma1 and gamma2. The _PSI_ matrix contains the error variances and covariances in the structural model. In this example, psi1 and psi2 are two free parameters for the error variances. Finally, the _PHI_ matrix is the covariance matrix for the ξ-variables. In this example, there is only one ξ-variable so that this matrix contains only the estimated variance of SES. This variance is represented by the parameter phi.

The estimates obtained from fitting the LISMOD model are the same as those from fitting the equivalent PATH, RAM, or LINEQS model. To some researchers the LISMOD modeling language might be more familiar, while for others modeling languages such as PATH, RAM, or LINEQS are more convenient to use.

Example 33.24: Testing Competing Path Models for the Career Aspiration Data

(View the complete code for this example (calex12.sas) in the example repository.)

This example uses some well-known data from Haller and Butterworth (1960). The section "A Combined Measurement-Structural Model" on page 349 in Chapter 18, "Introduction to Structural Equation Modeling with Latent Variables," analyzes some models for these data. Inspired by the examples given in Loehlin (1987), this example shows additional applications to the same data set, but with a focus on testing nested models. By manipulating the OUTMODEL= data set, this example shows how you can specify new models in an efficient way. Various models and analyses of these data are also given by Duncan, Haller, and Portes (1968), Jöreskog and Sörbom (1988), and Loehlin (1987).

The study is concerned with the career aspirations of high school students and how these aspirations are affected by close friends. The data are collected from 442 seventeen-year-old boys in Michigan. There are 329 boys in the sample who named another boy in the sample as a best friend. The data from these 329 boys paired with the data from their best friends are analyzed.

Because of the dependency of the data, the effective sample size assumed in the example is 329, which you can specify in the NOBS= option in the PROC CALIS statements. See the section "A Combined Measurement-Structural Model" on page 349 in Chapter 18, "Introduction to Structural Equation Modeling with Latent Variables," for the justification of the use of this effective sample size.

The correlation matrix, taken from Jöreskog and Sörbom (1988), is shown in the following DATA step:

```
title 'Peer Influences on Aspiration: Haller & Butterworth (1960)';
data aspire(type=corr);
   _type_='corr';
   input _name_ $ riq rpa rses roa rea fiq fpa fses foa fea;
   label riq='Respondent: Intelligence'
         rpa='Respondent: Parental Aspiration'
         rses='Respondent: Family SES'
         roa='Respondent: Occupational Aspiration'
         rea='Respondent: Educational Aspiration'
         fiq='Friend: Intelligence'
         fpa='Friend: Parental Aspiration'
         fses='Friend: Family SES'
         foa='Friend: Occupational Aspiration'
         fea='Friend: Educational Aspiration';
   datalines;
riq   1.      .      .      .      .      .       .      .      .      .
rpa   .1839  1.      .      .      .      .       .      .      .      .
rses  .2220  .0489  1.      .      .      .       .      .      .      .
roa   .4105  .2137  .3240  1.      .      .       .      .      .      .
rea   .4043  .2742  .4047  .6247  1.      .       .      .      .      .
fiq   .3355  .0782  .2302  .2995  .2863  1.       .      .      .      .
fpa   .1021  .1147  .0931  .0760  .0702  .2087   1.      .      .      .
fses  .1861  .0186  .2707  .2930  .2407  .2950  -.0438  1.      .      .
foa   .2598  .0839  .2786  .4216  .3275  .5007   .1988  .3607  1.      .
fea   .2903  .1124  .3054  .3269  .3669  .5191   .2784  .4105  .6404  1.
;
```

For illustration purposes, this correlation matrix is treated here as if it were a covariance matrix for PROC CALIS to analyze. The reason is that the chi-square tests shown in this example are valid only with covariance structure analysis. See Example 33.27 for an illustration of covariance structure analysis on correlations.

Model 1: The Full Model

Loehlin (1987) analyzes the path model in Figure 33.24 for the data.

Figure 33.24 Path Diagram for Career Aspiration: Model 1

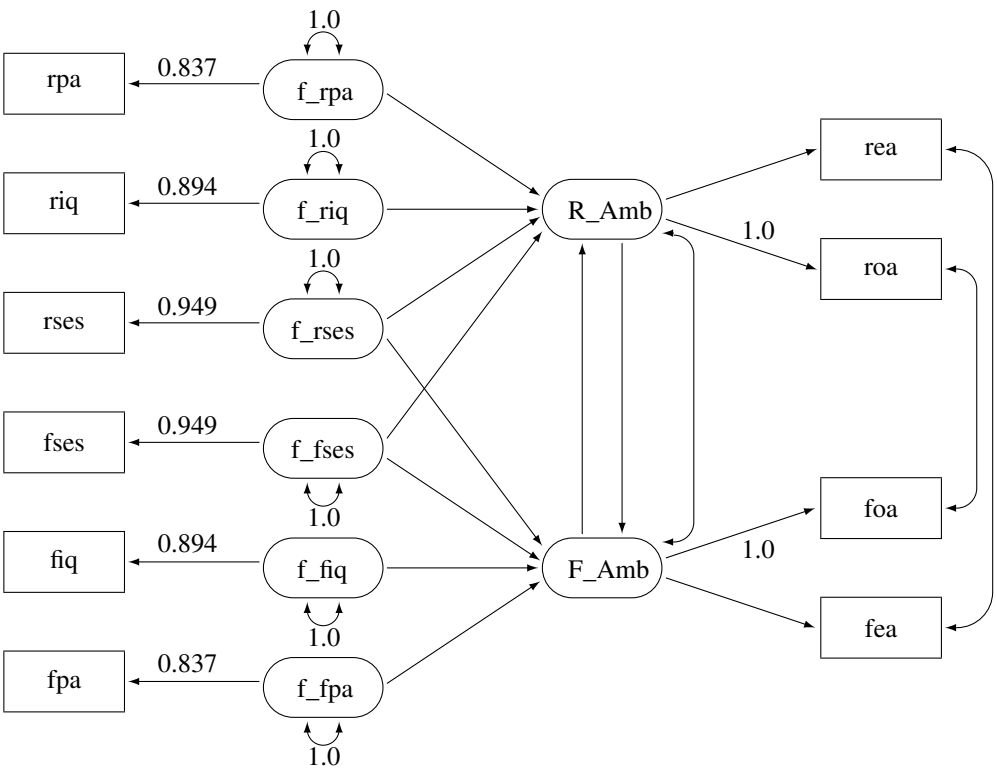

In Figure 33.24, the observed variables rpa, riq, rses, fses, fiq, and fpa are measured with errors. Their true scores counterparts f_rpa, f_riq, f_rses, f_fses, f_fiq, and f_fpa are latent variables in the model. Path coefficients from these latent variables to the observed variables are fixed coefficients, indicating the square roots of the theoretical reliabilities in the model. These latent variables, rather than the observed counterparts, serve as predictors of the ambition factors R_Amb and F_Amb. The error terms for these two latent factors are correlated, as indicated by a double-headed path (arrow) that connects the two factors. Correlated errors for the occupational aspiration variables (roa and foa) and the educational aspiration variables (rea and fea) are also shown in Figure 33.24. These correlated errors are also represented by two double-headed paths (arrows) in the path diagram.

Notice that the covariances among the six exogenous latent variables (f_rpa, f_riq, f_rses, f_fses, f_fiq, and f_fpa) are not represented in the path diagram for two reasons. First, there are 15 of these covariances and hence you need 15 double-headed arrows to represent them in the path diagram. Apparently, because of the space limitations, it would be difficult to put all these double-headed arrows in the path diagram without cluttering it. Second, covariances among exogenous latent variables are free parameters by default in

PROC CALIS, and therefore omitting these double-headed arrows in the path diagram is compatible with the default model specification in PROC CALIS. Similarly, double-headed arrows for the error variances of the endogenous variables (rpa, riq, rses, fses, fiq, fpa, R_Amb, and F_Amb) in the path diagram are omitted because they are unconstrained free parameters and are set automatically by default in PROC CALIS .

The model represented by the path diagram in Figure 33.24 is considered to be the full model for the data, in the sense that it has the largest number of parameters among the competing models considered this example. The same model is analyzed in the section "A Combined Measurement-Structural Model" on page 349 in Chapter 18, "Introduction to Structural Equation Modeling with Latent Variables," with the following specification:

```
proc calis data=aspire nobs=329;
   path
      /* measurement model for intelligence and environment */
      rpa     <===  f_rpa    = 0.837,
      riq     <===  f_riq    = 0.894,
      rses    <===  f_rses   = 0.949,
      fses    <===  f_fses   = 0.949,
      fiq     <===  f_fiq    = 0.894,
      fpa     <===  f_fpa    = 0.837,

      /* structural model of influences: 5 equality constraints */
      f_rpa   ===>  R_Amb ,
      f_riq   ===>  R_Amb ,
      f_rses  ===>  R_Amb ,
      f_fses  ===>  R_Amb ,
      f_rses  ===>  F_Amb ,
      f_fses  ===>  F_Amb ,
      f_fiq   ===>  F_Amb ,
      f_fpa   ===>  F_Amb ,
      F_Amb   ===>  R_Amb ,
      R_Amb   ===>  F_Amb ,

      /* measurement model for aspiration: 1 equality constraint */
      R_Amb   ===>  rea  ,
      R_Amb   ===>  roa      = 1.,
      F_Amb   ===>  foa      = 1.,
      F_Amb   ===>  fea  ;
   pvar
      f_rpa f_riq f_rses f_fpa f_fiq f_fses = 6 * 1.0;
   pcov
      R_Amb F_Amb            ,
      rea  fea               ,
      roa  foa               ;
run;
```

The PATH model specification represents each arrow (single-headed and double-headed) in the path diagram. You transcribe each arrow in Figure 33.24 into an entry in the PATH model. The PATH statement specifies all the single-headed arrows in the path diagram. The PVAR statement specifies all the double-headed arrows that point to individual variables (that is, the fixed error variances of the exogenous latent variables) in the path diagram. The PCOV statement specifies all the double-headed arrows that connect paired variables (that is, the error covariances) in the path diagram.

Output 33.24.1 shows the fit summary of Model 1.

Output 33.24.1 Career Aspiration Data: Fit Summary of Model 1

Fit Summary	
Chi-Square	12.0132
Chi-Square DF	13
Pr > Chi-Square	0.5266
Standardized RMR (SRMR)	0.0149
RMSEA Estimate	0.0000
Akaike Information Criterion	96.0132
Bozdogan CAIC	297.4476
Schwarz Bayesian Criterion	255.4476

Since the p-value for the chi-square test is 0.5266, this model clearly cannot be rejected. Both standardized RMR and RMSEA are very small. All these point to an excellent model fit. Three information-theoretic fit indices are also shown: Akaike's information criterion (AIC), Bozdogan's CAIC, and Schwarz's Bayesian Criterion (SBC). These indices are useful when you need to compare competing models for the data.

Model 2: The Model with Equality Constraints

You now consider a much more restrictive model with equality constraints in the model. The path diagram for this constrained model is shown in Figure 33.25.

Figure 33.25 Path Diagram for Career Aspiration: Model 2

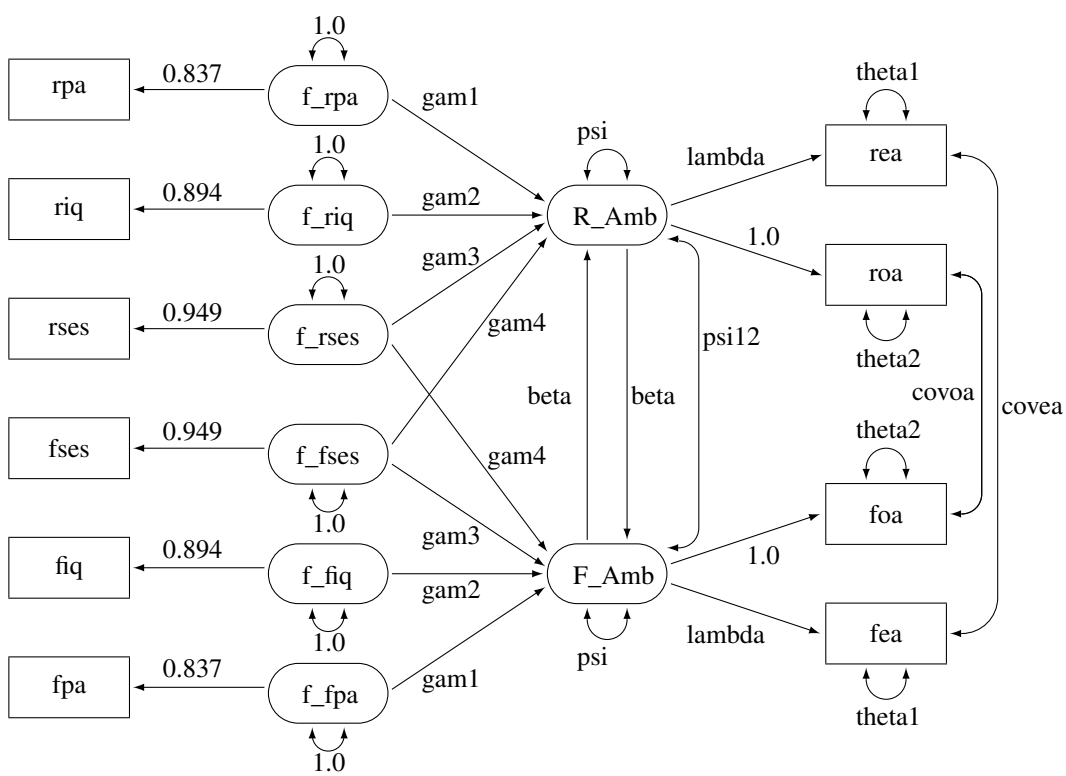

The main idea about setting the equality constraints in this model is that there is some symmetry in the model components that correspond to the respondent and his friend. In particular, the corresponding coefficients

or parameters should be equal. For example, the path **f_rpa===>R_Amb** for the respondent has the same effect as that of **f_fpa===>F_Amb**. In the path diagram, they are both labeled by the same parameter gam1. Generalizing the same idea to other pairs of paths, Output 33.25 shows nine pairs of these equality constraints, which are all represented by the same parameter names for distinct (single-headed or double-headed) paths.

However, because of the space limitation, there are six more equality constraints that are not shown in the path diagram. These six constraints concern the covariance structures of the exogenous latent factors f_rpa, f_riq, f_rses, f_fses, f_fiq, and f_fpa. The first three factors are for the respondent, and the last three are for his friend. Using the same symmetry argument, the covariance structures imposed on these exogenous latent factors are shown in the following:

```
         f_rpa  f_riq  f_rses   f_fpa f_fiq f_fses
f_rpa      1.
f_riq      c1     1.
f_rses     c2     c3     1.

f_fpa      c4     c5     c6       1.
f_fiq      c5     c7     c8       c1    1.
f_fses     c6     c8     c9       c2    c3    1.
```

In this pattern of covariance structures, the covariance matrix (upper left portion) for the latent factors of the respondent is the same as that (lower right portion) for the latent factors of his friend. The cross-covariances among the factors between the friends (lower left portion) also display a symmetry pattern. There are six pairs of equality constraints in the covariance structures. Imposing these six pairs of equality constraints and the nine pairs of equality constraints in the path diagram lead to Model 2 of Loehlin (1987).

You can specify the current constrained model by the following PATH modeling language of PROC CALIS:

```
proc calis data=aspire nobs=329 outmodel=model2;
   path
      /* measurement model for intelligence and environment */
      rpa     <===  f_rpa    = 0.837,
      riq     <===  f_riq    = 0.894,
      rses    <===  f_rses   = 0.949,
      fses    <===  f_fses   = 0.949,
      fiq     <===  f_fiq    = 0.894,
      fpa     <===  f_fpa    = 0.837,

      /* structural model of influences: 5 equality constraints */
      f_rpa   ===>  R_Amb    = gam1,
      f_riq   ===>  R_Amb    = gam2,
      f_rses  ===>  R_Amb    = gam3,
      f_fses  ===>  R_Amb    = gam4,
      f_rses  ===>  F_Amb    = gam4,
      f_fses  ===>  F_Amb    = gam3,
      f_fiq   ===>  F_Amb    = gam2,
      f_fpa   ===>  F_Amb    = gam1,
      F_Amb   ===>  R_Amb    = beta,
      R_Amb   ===>  F_Amb    = beta,

      /* measurement model for aspiration: 1 equality constraint */
      R_Amb   ===>  rea      = lambda,
```

```
      R_Amb   ===>  roa      = 1.,
      F_Amb   ===>  foa      = 1.,
      F_Amb   ===>  fea      = lambda;
   pvar
      f_rpa f_riq f_rses f_fpa f_fiq f_fses = 6 * 1.0,
      R_Amb F_Amb             = 2 * psi,        /* 1 ec */
      rea fea                 = 2 * theta1,     /* 1 ec */
      roa foa                 = 2 * theta2;     /* 1 ec */
   pcov
      R_Amb F_Amb             = psi12,
      rea  fea                = covea,
      roa  foa                = covoa,
      f_rpa f_riq f_rses      = cov1-cov3,       /* 3 ec */
      f_fpa f_fiq f_fses      = cov1-cov3,
      f_rpa f_riq f_rses * f_fpa f_fiq f_fses =  /* 3 ec */
          cov4 cov5 cov6  cov5 cov7 cov8  cov6 cov8 cov9;
run;
```

In the current PATH model specification, you specify the same set of paths as in Model 1. In addition, to set the required constraints in this path model, you use parameter names to label the related paths, variances, or covariances. Same parameter names mean equality constraints. The 15 equality constraints are labeled with comments in the specification. In the PROC CALIS statement, you use the OUTMODEL= option to output the model estimation results into the output data set model2, which is used for subsequent hypotheses tests.

Output 33.24.2 shows the fit summary of Model 2.

Output 33.24.2 Career Aspiration Data: Fit Summary of Model 2 in Loehlin (1987)

Fit Summary	
Chi-Square	19.0697
Chi-Square DF	28
Pr > Chi-Square	0.8960
Standardized RMR (SRMR)	0.0276
RMSEA Estimate	0.0000
Akaike Information Criterion	73.0697
Bozdogan CAIC	202.5632
Schwarz Bayesian Criterion	175.5632

The test of Model 2 against Model 1 (Loehlin 1987) yields a chi-square of 19.0697 – 12.0132 = 7.0565 with 15 degrees of freedom, which is clearly not significant. This indicates that the restricted Model 2 fits at least as well as Model 1. Schwarz's Bayesian criterion (SBC) is also much lower for Model 2 (175.5623) than for Model 1 (255.4476). Hence, Model 2 seems preferable on both substantive and statistical grounds.

Model 3: No SES Paths

A question of substantive interest is whether the friend's socioeconomic status (SES) has a significant direct influence on a boy's ambition. This can be addressed by omitting the paths from f_fses to R_Amb and from f_rses to F_Amb designated by the parameter name gam4, yielding Model 3 of Loehlin (1987). The corresponding path diagram is shown in Figure 33.26.

Figure 33.26 Path Diagram for Career Aspiration: Model 3

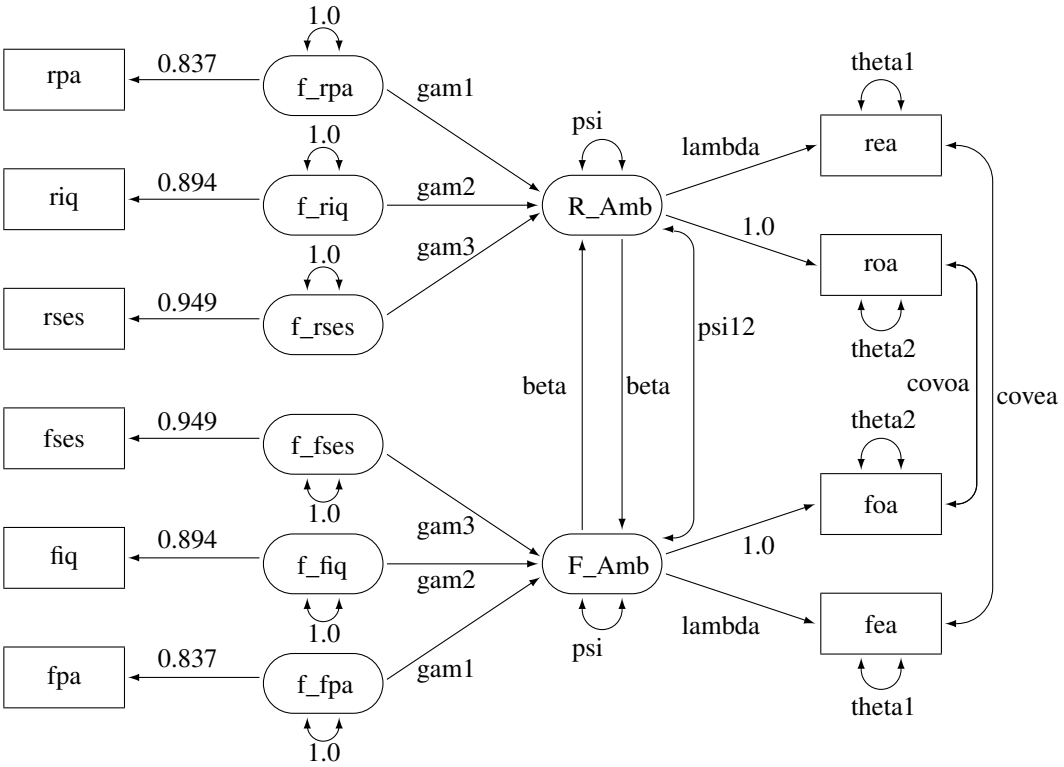

In Figure 33.26, you drop the paths **f_rses===>F_Amb** and **f_fses===>R_Amb** from the previous model. Using the path diagram in Figure 33.26, you can specify the current model the same way you do for Model 2. However, because you have the estimation results from Model 2 in the SAS data set model2, you can modify this SAS data set to reflect the current model specification and then input the modified SAS data set as an INMODEL= file for PROC CALIS to analyze.

First, you create a new SAS data set model3 by the following DATA step:

```
data model3(type=calismdl);
   set model2;
   if _name_='gam4' then
      do;
         _name_=' ';
         _estim_=0;
      end;
run;
```

Essentially, by blanking out the parameter name for the target paths, you are stating that these paths are no longer associated with the free parameter gam4 in the new model. Instead, you put a fixed zero to these paths. This way you eliminate the paths **f_rses===>F_Amb** and **f_fses===>R_Amb** for Model 3, of which the model specification is now saved in the model3 data set.

Next, you input model3 as the INMODEL= data set for PROC CALIS to analyze, as shown in the following statements:

```
proc calis data=aspire nobs=329 inmodel=model3;
run;
```

PROC CALIS can now use the previous estimation results for fitting the required model. Output 33.24.3 shows the fit summary of Model 3.

Output 33.24.3 Career Aspiration Data: Fit Summary of Model 3 in Loehlin (1987)

Fit Summary	
Chi-Square	23.0365
Chi-Square DF	29
Pr > Chi-Square	0.7749
Standardized RMR (SRMR)	0.0304
RMSEA Estimate	0.0000
Akaike Information Criterion	75.0365
Bozdogan CAIC	199.7340
Schwarz Bayesian Criterion	173.7340

The chi-square value for testing Model 3 versus Model 2 is 23.0365 – 19.0697 = 3.9668 with one degree of freedom and a p-value of 0.0464. The chi-square test shows a marginal significance, which means that the paths might be needed in the model. However, the SBC (173.7340) indicates that Model 3 is slightly preferable to Model 2, which has an SBC value of 175.5632.

Model 4: No Reciprocal Influence between the Ambition Factors

Another important question is whether the reciprocal influences between the respondent's and friend's ambitions are needed in the model. To test whether these paths are zero, you can set the parameter beta for the paths linking R_Amb and F_Amb to zero to obtain Model 4 of Loehlin (1987).

Similar to Model 3, you can modify the model2 data set to form the new model data set model4 for PROC CALIS to analyze, as shown in the following statements:

```
data model4(type=calismdl);
   set model2;
   if _name_='beta' then
      do;
         _name_=' ';
         _estim_=0;
      end;
run;

proc calis data=aspire nobs=329 inmodel=model4;
run;
```

Output 33.24.4 shows the fit summary of Model 4.

Output 33.24.4 Career Aspiration Data: Fit Summary of Model 4 in Loehlin (1987)

Fit Summary	
Chi-Square	20.9981
Chi-Square DF	29
Pr > Chi-Square	0.8592
Standardized RMR (SRMR)	0.0304
RMSEA Estimate	0.0000
Akaike Information Criterion	72.9981
Bozdogan CAIC	197.6956
Schwarz Bayesian Criterion	171.6956

The chi-square value for testing Model 4 versus Model 2 is 20.9981 – 19.0697 = 1.9284 with one degree of freedom and a p-value of 0.1649. Hence, there is little evidence of reciprocal influence.

Model 5: No Disturbance Correlation between the Ambition Factors

Model 2 of Loehlin (1987) has the direct paths connecting the latent ambition factors R_Amb and F_Amb and a covariance between the disturbance or error terms (that is, a double-headed arrow connecting the two factors in the path diagram shown in Figure 33.25). The presence of this disturbance correlation serves as a "wastebasket" that enables other omitted variables to have joint influences on the respondent's and his friend's ambition factors. To test the hypothesis that this disturbance correlation is zero, you use the following statements to set the parameter psi12 to zero in the model5 data set and fit the new model by PROC CALIS:

```
data model5(type=calismdl);
   set model2;
   if _name_='psi12' then
      do;
         _name_=' ';
         _estim_=0;
      end;
run;

proc calis data=aspire nobs=329 inmodel=model5;
run;
```

Output 33.24.5 displays the fit summary of Model 5.

Output 33.24.5 Career Aspiration Data: Fit Summary of Model 5 in Loehlin (1987)

Fit Summary	
Chi-Square	19.0745
Chi-Square DF	29
Pr > Chi-Square	0.9194
Standardized RMR (SRMR)	0.0276
RMSEA Estimate	0.0000
Akaike Information Criterion	71.0745
Bozdogan CAIC	195.7721
Schwarz Bayesian Criterion	169.7721

The chi-square value for testing Model 5 versus Model 2 is 19.0745 – 19.0697 = 0.0048 with one degree of

freedom. This test statistic is insignificant. Therefore, omitting the covariance between the disturbance terms causes hardly any deterioration in the fit of the model.

Model 7: No Reciprocal Influence and No Disturbance Correlation between the Ambition Factors

The test in Model 4 fails to provide evidence of a direct reciprocal influence between the respondent's and friend's ambitions, and the test in Model 5 fails to provide evidence of a covariance or correlation between the disturbance terms for the ambition factors. Because you consider these two tests separately, you cannot establish evidence to eliminate the reciprocal influence and the disturbance correlation jointly. Instead, to make such a joint inference, it is important to test both hypotheses together by setting both beta and psi12 to zero as in Model 7 of Loehlin (1987). The following statements show how you can do that by modifying the model2 data set to form a new INMODEL= data set model7 for PROC CALIS to analyze:

```
data model7(type=calismdl);
   set model2;
   if _name_='psi12'|_name_='beta' then
      do;
         _name_=' ';
         _estim_=0;
      end;
run;

proc calis data=aspire nobs=329 inmodel=model7;
run;
```

Output 33.24.6 shows the fit summary of Model 7.

Output 33.24.6 Career Aspiration Data: Fit Summary of Model 7 in Loehlin (1987)

Fit Summary	
Chi-Square	25.3466
Chi-Square DF	30
Pr > Chi-Square	0.7080
Standardized RMR (SRMR)	0.0363
RMSEA Estimate	0.0000
Akaike Information Criterion	75.3466
Bozdogan CAIC	195.2480
Schwarz Bayesian Criterion	170.2480

When Model 7 is tested against Models 2, 4, and 5, the p-values are respectively 0.0433, 0.0370, and 0.0123, indicating that the combined effect of the reciprocal influence and the covariance of the disturbance terms is statistically significant. Thus, the hypothesis tests indicate that it is acceptable to omit either the reciprocal influences or the covariance of the disturbances, but not both.

Model 6: No Error Correlations between the Friend's Educational and Occupational Aspiration

It is also of interest to test the covariances (covea and covoa) between the error terms for educational aspiration (that is, between rea and fea) and occupational aspiration (that is, between roa and foa), because these terms are omitted from Jöreskog and Sörbom (1988) models. Constraining covea and covoa to zero produces Model 6 of Loehlin (1987). You can use the following statements to fit this model:

```
data model6(type=calismdl);
   set model2;
   if _name_='covea'|_name_='covoa' then
      do;
         _name_=' ';
         _estim_=0;
      end;
run;

proc calis data=aspire nobs=329 inmodel=model6;
run;
```

Output 33.24.7 shows the fit summary of Model 6.

Output 33.24.7 Career Aspiration Data: Model 6 of Loehlin (1987)

Fit Summary	
Chi-Square	33.4475
Chi-Square DF	30
Pr > Chi-Square	0.3035
Standardized RMR (SRMR)	0.0306
RMSEA Estimate	0.0187
Akaike Information Criterion	83.4475
Bozdogan CAIC	203.3489
Schwarz Bayesian Criterion	178.3489

The chi-square value for testing Model 6 versus Model 2 is 33.4476 – 19.0697 = 14.3779 with two degrees of freedom and a p-value of 0.0008, indicating that there is considerable evidence of correlation between the error terms.

Summary of Competing Models

The following table summarizes the results from the seven models described in Loehlin (1987).

Model	χ^2	*df*	*p*-value	SBC
1. Full model	12.0132	13	0.5266	255.4476
2. Equality constraints	19.0697	28	0.8960	175.5632
3. No SES path	23.0365	29	0.7749	173.7340
4. No reciprocal influence	20.9981	29	0.8592	171.6956
5. No disturbance correlation	19.0745	29	0.9194	169.7721
6. No error correlation	33.4475	30	0.3035	178.3489
7. Constraints from both 4 and 5	25.3466	30	0.7080	170.2480

For comparing models, you can use a DATA step to compute the differences of the chi-square statistics and *p*-values, as shown in the following statements:

```
data _null_;
   array achisq[7] _temporary_
      (12.0132 19.0697 23.0365 20.9981 19.0745 33.4475 25.3466);
   array adf[7] _temporary_
      (13 28 29 29 29 30 30);
   retain indent 16;
   file print;
   input ho ha @@;
   chisq = achisq[ho] - achisq[ha];
   df = adf[ho] - adf[ha];
   p = 1 - probchi( chisq, df);
   if _n_ = 1 then put
      / +indent 'model comparison   chi**2   df  p-value'
      / +indent '--------------------------------------';
   put +indent +3 ho ' versus ' ha @18 +indent chisq 8.4 df 5. p 9.4;
   datalines;
2 1    3 2    4 2    5 2    7 2    7 4    7 5    6 2
;
```

The DATA step displays the table in Output 33.24.8.

Output 33.24.8 Career Aspiration Data: Model Comparisons

model	comparison	chi**2	df	p-value
2	versus 1	7.0565	15	0.9561
3	versus 2	3.9668	1	0.0464
4	versus 2	1.9284	1	0.1649
5	versus 2	0.0048	1	0.9448
7	versus 2	6.2769	2	0.0433
7	versus 4	4.3485	1	0.0370
7	versus 5	6.2721	1	0.0123
6	versus 2	14.3778	2	0.0008

Although none of the seven models can be rejected when tested against the alternative of an unrestricted covariance matrix, the model comparisons make it clear that there are important differences among the models. Schwarz's Bayesian criterion indicates Model 5 as the model of choice. The constraints added to Model 5 in Model 7 can be rejected ($p = 0.0123$), while Model 5 cannot be rejected when tested against the less constrained Model 2 ($p = 0.9448$). Hence, among the small number of models considered, Model 5 has strong statistical support. However, as Loehlin (1987, p.106) points out, many other models for these data could be constructed. Further analysis should consider, in addition to simple modifications of the models, the possibility that more than one friend could influence a boy's aspirations, and that a boy's ambition might have some effect on his choice of friends. Pursuing such theories would be statistically challenging.

Example 33.25: Fitting a Latent Growth Curve Model

(View the complete code for this example (calex07.sas) in the example repository.)

Latent factors in structural equation modeling are constructed to represent important unobserved hypothetical constructs. However, with some manipulations latent factors can also represent random effects in models. In this example, a simple latent growth curve model is considered. You use latent factors to represent the random intercepts and slopes in the latent growth curve model.

Sixteen individuals were invited to a training program that was designed to boost self-confidence. During the training, the individuals' confidence levels were measured at five time points: initially and four more times separated by equal intervals. The data are stored in the following SAS data set:

```
data growth;
   input y1 y2 y3 y4 y5;
   datalines;
17.6  21.4  25.6  32.1  37.7
13.2  14.3  18.9  20.3  25.4
11.6  13.5  17.4  22.1  39.6
10.7  11.1  13.2  18.2  21.4
18.7  23.7  28.6  31.5  34.0
18.3  19.2  20.5  23.2  25.9
 9.2  13.5  17.8  19.2  21.1
18.3  23.5  27.9  30.2  34.6
11.2  15.6  20.8  22.7  30.4
17.0  22.9  26.9  31.9  35.6
10.4  13.6  18.0  25.6  29.3
17.7  19.0  22.5  28.5  30.7
14.5  19.4  21.1  28.8  31.5
20.0  21.4  28.9  30.2  35.6
14.6  19.3  21.7  28.5  32.0
11.7  15.2  19.1  23.7  28.7
;
```

First, consider a simple linear regression model for the confidence levels at time t due to training. That is,

$$y_t = \alpha + \beta T_t + e_t$$

where y_t represents the confidence level at time t ($t = 1, 2, \ldots, 5$), α represents the intercept, β represents the slope or the effect of training, T_t represents the fixed time point at t ($T_1 = 0$ and $T_i = T_{i-1} + 1$), and e_t is the error term at time t.

This simple linear regression assumes that the effect of training (slope) and the intercept are constants for the individuals. However, individual differences are rules rather than exceptions. It is thus more reasonable to argue that an index i for individuals should be added to the intercept and slope in the model. As a result, the following individualized regression model is derived:

$$y_{it} = \alpha_i + \beta_i T_t + e_t$$

where $i = 1, 2, \ldots, 16$. In this model, individuals are assumed to have different intercepts and slopes (regression coefficients). Note that theoretically e_t could also be "individualized" as e_{ti} in the model. But this is not done because such a model would be unnecessarily complicated without gaining additional insights in return.

Unfortunately, this individualized model with individual intercepts and slopes cannot be estimated directly. If you treat each α_i and β_i as fixed parameters, you are going to have too many parameters for the model to be identified or estimable. A workable solution is to treat α and β in the original linear regression model as random variables instead. That is, the latent growth curve model of interest is as follows:

$$y_t = \alpha + \beta T_t + e_t$$

where (α, β) is bivariate normal with unknown means, variances, and covariance. Therefore, instead of having 16 intercepts and 16 slopes to estimate in the individualized regression model, the final latent growth curve model has to estimate only two means, two variances and one covariance in the bivariate distribution of (α, β).

To use PROC CALIS to fit this latent growth curve model, the random intercept and effect are treated as if they were covarying latent factors. To make them stand out more as latent variables, the random intercept and slope are renamed as f_α and f_β in the following structural equation:

$$y_t = f_\alpha + T_t f_\beta + e_t$$

where f_α and f_β are bivariate-normal latent variables. This model assumes that the error distribution is time dependent (with the index t). A simpler version is to make this error term invariant over time, which is then represented by the following model with constrained error variances:

$$y_t = f_\alpha + T_t f_\beta + e$$

This constrained model is considered first. The LINEQS modeling language is used to specify this constrained model, as shown in the following statements.

```
proc calis method=ml data=growth nostand noparmname;
   lineqs
      y1 = 0. * Intercept + f_alpha                 + e1,
      y2 = 0. * Intercept + f_alpha  +  1 * f_beta + e2,
      y3 = 0. * Intercept + f_alpha  +  2 * f_beta + e3,
      y4 = 0. * Intercept + f_alpha  +  3 * f_beta + e4,
      y5 = 0. * Intercept + f_alpha  +  4 * f_beta + e5;
   variance
      f_alpha f_beta,
      e1-e5 = 5 * evar;
   mean
      f_alpha f_beta;
   cov
      f_alpha f_beta;
   fitindex on(only)=[chisq df probchi];
run;
```

In the LINEQS model specification, f_alpha and f_beta are treated as latent factors representing the random intercept and random slope, respectively. The f_ prefix for latent factors is required as a convention in the LINEQS modeling language. See the sections "Naming Variables in the LINEQS Model" on page 1747 and "Naming Variables and Parameters" on page 1777 for details.

Notice that you need to set the ordinary (non-random) intercepts for endogenous variables to zero by the `0.*Intercept` specification because non-random intercepts for observed endogenous variables are default parameters in the LINEQS model. Because you have already used f_alpha as the random intercept, you must

turn off the default non-random intercept term for the observed endogenous variables y1–y5. Otherwise, your latent growth curve model might be over-parameterized.

At $T_1 = 0$, y_1 represents the initial confidence measurement so that it is not subject to the random effect f_beta. The next four measurements y_2, y_3, y_4, and y_5 are measured at time points T_2, T_3, T_4, and T_5, respectively. These are fixed time points with constant values 1, 2, 3, and 4, respectively, in the equations of the LINEQS statement.

The means, variances and covariances of f_alpha and f_beta are parameters in the model. The variances of these two latent variables are specified in the VARIANCE statement, while their covariance is specified in the COV statement. The means of f_alpha and f_beta are specified in the MEAN statement. Unlike the specification for the variances of e1–e5. All these parameters for the latent factors are unnamed because you do not need to constrain them by references.

The error variances for e1–e5 are also specified in the VARIANCE statement. Using the shorthand notation **`5 * evar`**, the parameter name evar is repeated five times for the five error variances. This constrains the error variances for e1–e5 to be equal.

You also use some special printing options in this example. In the PROC CALIS statement, the NOSTAND option is specified because standardized solution is not of interest. The reason is that y1–y5 were already measured on comparable scales, making standardization unnecessary for interpretations. Another printing option specified is the NOPARMNAME option in the PROC CALIS statement. This option suppresses the printing of parameter names in the output for estimation. This makes the output look more concise when you do not need to make references to the parameter names. Still another printing option used is the ON(ONLY)= option of the FITINDEX statement. This option trims down the display of fit indices to include only those listed in the option. See the FITINDEX statement on page 1610 for details.

Output 33.25.1 shows the fit summary table.

Output 33.25.1 Random Intercepts and Effects with Constrained Error Variances: Model Fit

Fit Summary	
Chi-Square	31.4310
Chi-Square DF	14
Pr > Chi-Square	0.0048

In Output 33.25.1, the chi-square value in the fit summary table is 31.431 ($df = 14$, $p < 0.01$), which is a statistically significant result that might indicate a poor model fit. Despite that, it is illustrative to continue to look at the main estimation results, which are shown in the following table.

Output 33.25.2 Estimation of Random Intercepts and Effects with Constrained Error Variances

Estimates for Variances of Exogenous Variables					
Variable Type	**Variable**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Latent	**f_alpha**	13.89140	5.81540	2.3887	0.0169
	f_beta	0.80742	0.42198	1.9134	0.0557
Error	**e1**	3.32185	0.70031	4.7434	<.0001
	e2	3.32185	0.70031	4.7434	<.0001
	e3	3.32185	0.70031	4.7434	<.0001
	e4	3.32185	0.70031	4.7434	<.0001
	e5	3.32185	0.70031	4.7434	<.0001

Output 33.25.2 *continued*

Covariances Among Exogenous Variables					
Var1	Var2	Estimate	Standard Error	t Value	Pr > |t|
f_alpha	f_beta	-0.35281	1.13815	-0.3100	0.7566

Mean Parameters					
Variable Type	Variable	Estimate	Standard Error	t Value	Pr > |t|
Latent	f_alpha	14.15875	1.02906	13.7589	<.0001
	f_beta	4.04813	0.27563	14.6867	<.0001

In Output 33.25.2, the estimated variance of the random intercept α, which is represented by the variance estimate of the latent factor f_alpha, is 13.891 ($t = 2.389$). In the next row of the same table, the variance estimate of the random effect β, which is represented by the variance estimate of the latent factor f_beta, is 0.807 ($t = 1.913$).

The covariance of the random intercept and the random effect is shown in the next table for "Covariances Among Exogenous Variables." A negative estimate of –0.353 is shown. This means that the initial self-confidence level and the boosting effect of training are negatively correlated. The higher the initial self-confidence level, the smaller the training effect.

In the last table for the "Mean Parameters," the estimated mean of the random intercept is 14.159, which is an estimate of the averaged initial self-confidence level. The estimated mean of random effect is 4.048, which is an estimate of the averaged training effect. They are both significantly different from zero.

Given that the model does not fit that well, perhaps you should not take the interpretations of these estimates so seriously. Knowing that the distribution of the errors might have been time-dependent, you now try to improve the fit of the model by relaxing the constraint about common error variances. You can use the following specifications:

```
proc calis method=ml data=growth nostand noparmname;
   lineqs
      y1 = 0. * Intercept + f_alpha                + e1,
      y2 = 0. * Intercept + f_alpha  +  1 * f_beta + e2,
      y3 = 0. * Intercept + f_alpha  +  2 * f_beta + e3,
      y4 = 0. * Intercept + f_alpha  +  3 * f_beta + e4,
      y5 = 0. * Intercept + f_alpha  +  4 * f_beta + e5;
   variance
      f_alpha f_beta,
      e1-e5;
   mean
      f_alpha f_beta;
   cov
      f_alpha f_beta;
   fitindex on(only)=[chisq df probchi];
run;
```

In this new specification, there is only one change in the VARIANCE statement from the previous specification. That is, you now specify only the error variables without putting parameter names for them. This makes the variances of e1–e5 free (unconstrained) parameters in the model.

Output 33.25.3 shows the model fit summary.

Output 33.25.3 Random Intercepts and Effects with Unconstrained Error Variances: Model Fit

Fit Summary	
Chi-Square	11.6250
Chi-Square DF	10
Pr > Chi-Square	0.3109

The chi-square for the unconstrained model is 11.625 (df = 10, p > .10). This indicates an acceptable model fit. The chi-square difference test can also be conducted for testing the previous constrained model against this new model. The chi-square difference is 19.81 = 31.431 – 11.625. With df = 4, this chi-square difference value is statistically significant at α=0.01, indicating a significant improvement of model fit by using the unconstrained model.

Output 33.25.4 shows the estimation results.

Output 33.25.4 Estimation of Random Intercepts and Effects with Unconstrained Error Variances

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Estimate	Standard Error	t Value	Pr > |t|
Latent	**f_alpha**	14.70066	5.66942	2.5930	0.0095
	f_beta	0.45059	0.29867	1.5087	0.1314
Error	**e1**	2.81709	1.35331	2.0816	0.0374
	e2	0.32214	0.46118	0.6985	0.4849
	e3	1.94428	0.86824	2.2393	0.0251
	e4	1.88569	1.21307	1.5545	0.1201
	e5	14.65202	5.99357	2.4446	0.0145

Covariances Among Exogenous Variables

Var1	Var2	Estimate	Standard Error	t Value	Pr > |t|
f_alpha	**f_beta**	0.35292	0.90366	0.3906	0.6961

Mean Parameters

Variable Type	Variable	Estimate	Standard Error	t Value	Pr > |t|
Latent	**f_alpha**	14.03047	1.01534	13.8185	<.0001
	f_beta	3.96793	0.22612	17.5478	<.0001

The estimation results for the unconstrained model present a slightly different picture than the constrained model. While the estimates for the means and variances of the random intercept and the random training effect look similar in both models, estimates of the covariance between the random intercept and the random training effect are quite different in the two models. The covariance estimate is negative (–0.353) in the constrained model, but it is positive (0.353) in the unconstrained model. However, because the covariance estimates are not statistically significant in both models (t = –0.310 and 0.391, respectively), you wonder whether the current data are showing strong evidence that supports one way or another. To get a clearer picture, perhaps you need to collect more data and fit the models again to examine the significance of the covariance between the random intercept and slope.

Example 33.26: Higher-Order and Hierarchical Factor Models

(View the complete code for this example (calex08.sas) in the example repository.)

In this example, confirmatory higher-order and hierarchical factor models are fitted by PROC CALIS.

In higher-order factor models, factors are at different levels. The higher-order factors explain the relationships among factors at the next lower level, in the same way that the first-order factors explain the relationships among manifest variables. For example, in a two-level higher order factor model you have nine manifest variables V1–V9 with three first-order factors F1–F3. The first-order factor pattern of the model might appear like the following:

```
     F1   F2   F3
V1    x
V2    x
V3    x
V4         x
V5         x
V6         x
V7              x
V8              x
V9              x
```

where each "x" marks a nonzero factor loading and all other unmarked entries are fixed zeros in the model. To explain the correlations among the first-order factors, a second-order factor F4 is hypothesized with the following second-order factor pattern:

```
     F4
F1    x
F2    x
F3    x
```

If substantiated by your theory, you might have higher-order factor models with more than two levels.

In hierarchical factor models, all factors are at the same (first-order) level but are different in their clusters of manifest variables related. Using the terminology of Yung, Thissen, and McLeod (1999), factors in hierarchical factor models are classified into "layers." The factors in the first layer partition the manifest variables into clusters so that each factor has a distinct cluster of related manifest variables. This part of the factor pattern of the hierarchical factor model is similar to that of the first-order factor model for manifest variables. The next layer of factors in the hierarchical factor model again partitions the manifest variables into clusters. However, this time each cluster contains at least two clusters of manifest variables that are formed in the previous layer. For example, the following is a factor pattern of a confirmatory hierarchical factor model with two layers:

```
      First Layer   |  Second Layer
     F1   F2   F3   |     F4
V1    x             |      x
V2    x             |      x
V3    x             |      x
```

```
V4        x       |      x
V5        x       |      x
V6        x       |      x
V7             x  |      x
V8             x  |      x
V9             x  |      x
```

F1–F3 are first-layer factors and F4 is the only second-layer factor. This special kind of two-layer hierarchical pattern is also known as the bifactor solution. In a bifactor solution, there are two classes of factors—group factors and a general factor. For example, in the preceding hierarchical factor pattern F1–F3 are group factors for different abilities and F4 is a general factor such as "intelligence" (see, for example, Holzinger and Swineford 1937). See Mulaik and Quartetti (1997) for more examples and distinctions among various types of hierarchical factor models. Certainly, if substantiated by your theory, hierarchical factor models with more than two layers are possible.

In this example, you use PROC CALIS to fit these two types of confirmatory factor models. First, you fit a second-order factor model to a real data set. Then you fit a bifactor model to the same data set. In the final section of this example, an informal account of the relationship between the higher-order and hierarchical factor models is attempted. Techniques for constraining parameters using PROC CALIS are also shown. This final section might be too technical in the first reading. Interested readers are referred to articles by Mulaik and Quartetti (1997), Schmid and Leiman (1957), and Yung, Thissen, and McLeod (1999) for more details.

A Second-Order Factor Analysis Model

In this section, a second-order confirmatory factor analysis model is applied to a correlation matrix of Thurstone reported by McDonald (1985). The correlation matrix is read into a SAS data set in the following statements:

```
data Thurst(type=corr);
title "Example of THURSTONE resp. McDONALD (1985, p.57, p.105)";
   _type_ = 'corr'; input _name_ $ V1-V9;
   label V1='Sentences' V2='Vocabulary' V3='Sentence Completion'
         V4='First Letters' V5='Four-letter Words' V6='Suffices'
         V7='Letter series' V8='Pedigrees' V9='Letter Grouping';
   datalines;
V1  1.       .      .      .      .      .      .      .      .
V2   .828   1.      .      .      .      .      .      .      .
V3   .776   .779   1.      .      .      .      .      .      .
V4   .439   .493    .460  1.      .      .      .      .      .
V5   .432   .464    .425   .674  1.      .      .      .      .
V6   .447   .489    .443   .590   .541  1.      .      .      .
V7   .447   .432    .401   .381   .402   .288  1.      .      .
V8   .541   .537    .534   .350   .367   .320   .555  1.      .
V9   .380   .358    .359   .424   .446   .325   .598   .452  1.
;
```

Variables in this data set are measures of cognitive abilities. Three factors are assumed for these nine variable V1–V9. These three factors are the first-order factors in the analysis. A second-order factor is also assumed to explain the correlations among the three first-order factors.

The following statements define a second-order factor model by using the LINEQS modeling language.

```
proc calis corr data=Thurst method=max nobs=213 nose nostand;
   lineqs
      V1      = X11 * Factor1                                  + E1,
      V2      = X21 * Factor1                                  + E2,
      V3      = X31 * Factor1                                  + E3,
      V4      =             X42 * Factor2                      + E4,
      V5      =             X52 * Factor2                      + E5,
      V6      =             X62 * Factor2                      + E6,
      V7      =                         X73 * Factor3          + E7,
      V8      =                         X83 * Factor3          + E8,
      V9      =                         X93 * Factor3          + E9,
      Factor1 =                                     L1g * FactorG + E10,
      Factor2 =                                     L2g * FactorG + E11,
      Factor3 =                                     L3g * FactorG + E12;
   variance
      FactorG   = 1. ,
      E1-E12    = U1-U9 W1-W3;
   bounds
      0. <= U1-U9;
   fitindex ON(ONLY)=[chisq df probchi];
   /* SAS Programming Statements: Dependent parameter definitions */
      W1  = 1. - L1g * L1g;
      W2  = 1. - L2g * L2g;
      W3  = 1. - L3g * L3g;
run;
```

In the first nine equations of the LINEQS statement, variables V1–V3 are manifest indicators of latent factor Factor1, variables V4–V6 are manifest indicators of latent factor Factor2, and variables V7–V9 are manifest indicators of latent factor Factor3. In the last three equations of the LINEQS statement, the three first-order factors Factor1–Factor3 are explained by a common source: FactorG. Hence, Factor1–Factor3 are correlated due to the common source FactorG in the model.

An error term is added to each equation in the LINEQS statement. These error terms E1–E12 are needed because the factors are not assumed to be perfect predictors of the corresponding outcome variables.

In the VARIANCE statement, you specify variance parameters for all independent or exogenous variables in the model: FactorG, and E1–E12. The variance of FactorG is fixed at one for identification. Variances for E1–E9 are given parameter names U1–U9, respectively. Variances for E10–E12 are given parameter names W1–W3, respectively. Note that for model identification purposes, W1–W3 are defined as dependent parameters in the SAS programming statements. That is,

$$W_i = 1. - L_{ig}^2 \quad (i = 1, 2, 3)$$

These dependent parameter definitions ensure that the variances for Factor1–Factor3 are fixed at ones for identification.

In the BOUNDS statement, you specify that variance parameters U1–U9 must be positive in the solution.

In addition to the statements for model specification, you specify some output control options in the PROC CALIS statement. You use the NOSTDERR and NOSTAND options suppress the display of standard errors and standardized results. In the FITINDEX statement, the ON(ONLY)= option requests only the model fit chi-square and its associated degrees of freedom and p-value be shown in the fit summary table. Using printing options in PROC CALIS to reduce the amount the of printout is a good practice. It makes your output more focused, as you output only what you need in a particular situation.

In Output 33.26.1, parameters and their initial values, gradients, and bounds are shown.

Output 33.26.1 Parameters in the Model

Optimization Start Parameter Estimates					
N	Parameter	Estimate	Gradient	Lower Bound	Upper Bound
1	X11	1.00000	0.13476	.	.
2	X21	1.01408	0.17327	.	.
3	X31	0.95518	0.12174	.	.
4	X42	1.00000	0.22548	.	.
5	X52	0.96603	0.21304	.	.
6	X62	0.88305	0.19782	.	.
7	X73	1.00000	0.21041	.	.
8	X83	1.03403	0.39324	.	.
9	X93	0.91752	0.19880	.	.
10	L1g	0.75060	-0.57492	.	.
11	L2g	0.64268	-0.50975	.	.
12	L3g	0.60919	-0.56538	.	.
13	U1	0.18879	0.14837	0	.
14	U2	0.16579	0.08989	0	.
15	U3	0.25988	-0.03231	0	.
16	U4	0.33068	0.20120	0	.
17	U5	0.37538	0.09124	0	.
18	U6	0.47808	-0.03595	0	.
19	U7	0.44813	0.20918	0	.
20	U8	0.40994	-0.12469	0	.
21	U9	0.53541	0.05959	0	.
Value of Objective Function = 0.5693888709					

The Number of Dependent Parameters is 3		
N	Parameter	Estimate
22	W1	0.43660
23	W2	0.58697
24	W3	0.62889

The first table contains all the independent parameters. There are twenty-one in total. Parameters W1–W3, which are defined in the SAS programming statements, are shown in the next table for dependent parameters. Their initial values are computed as functions of the independent parameters.

Output 33.26.2 shows the information about optimization—iteration history and the convergence status.

Output 33.26.2 Optimization

Optimization Start			
Active Constraints	0	Objective Function	0.5693888709
Max Abs Gradient Element	0.5749163348	Radius	1.8533033852

Output 33.26.2 *continued*

Iteration	Restarts	Function Calls	Active Constraints	Objective Function	Objective Function Change	Max Abs Gradient Element	Lambda	Ratio Between Actual and Predicted Change
1	0	5	0	0.38684	0.1825	0.5158	3.214	1.174
2	0	9	0	0.18706	0.1998	0.1003	0	1.181
3	0	11	0	0.18039	0.00667	0.0273	0	0.987
4	0	13	0	0.18020	0.000192	0.00581	0	0.881
5	0	15	0	0.18017	0.000023	0.00295	0	0.967
6	0	17	0	0.18017	3.08E-6	0.000686	0	1.083
7	0	19	0	0.18017	4.606E-7	0.000379	0	1.195
8	0	21	0	0.18017	7.365E-8	0.000096	0	1.283
9	0	23	0	0.18017	1.228E-8	0.000054	0	1.342
10	0	25	0	0.18017	2.098E-9	0.000018	0	1.377
11	0	27	0	0.18017	3.63E-10	8.561E-6	0	1.397

Optimization Results			
Iterations	11	**Function Calls**	30
Jacobian Calls	13	**Active Constraints**	0
Objective Function	0.1801712146	**Max Abs Gradient Element**	8.5605681E-6
Lambda	0	**Actual Over Pred Change**	1.3969188993
Radius	0.0000572561		

Convergence criterion (GCONV=1E-8) satisfied.

First, there are 21 independent parameters in the optimization by using 45 “Functions (Observations).” The so-called functions refer to the moments in the model that are structured with parameters. Nine lower bounds, which are specified for the error variance parameters, are specified in the optimization. The next table for iteration history shows that the optimization stops in 11 iterations. The notes at the bottom of table show that the solution converges without problems.

Output 33.26.3 shows the fit summary table. The chi-square model fit value is 38.196, with $df = 24$, and $p = 0.033$. This indicates a satisfactory model fit.

Output 33.26.3 Fit Summary

Fit Summary	
Chi-Square	38.1963
Chi-Square DF	24
Pr > Chi-Square	0.0331

Output 33.26.4 shows the fitted equations with final estimates. Interpretations of these loadings are not done here. The last table in this output shows various variance estimates. These estimates are classified by whether they are for the latent variables, error variables, or disturbance variables.

Output 33.26.4 Estimation Results

Linear Equations

V1	=	0.9047	Factor1	+	1.0000	E1
V2	=	0.9138	Factor1	+	1.0000	E2
V3	=	0.8561	Factor1	+	1.0000	E3
V4	=	0.8358	Factor2	+	1.0000	E4
V5	=	0.7972	Factor2	+	1.0000	E5
V6	=	0.7026	Factor2	+	1.0000	E6
V7	=	0.7808	Factor3	+	1.0000	E7
V8	=	0.7202	Factor3	+	1.0000	E8
V9	=	0.7035	Factor3	+	1.0000	E9
Factor1	=	0.8221	FactorG	+	1.0000	E10
Factor2	=	0.7818	FactorG	+	1.0000	E11
Factor3	=	0.8150	FactorG	+	1.0000	E12

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate
Latent	**FactorG**		1.00000
Error	**E1**	U1	0.18150
	E2	U2	0.16493
	E3	U3	0.26713
	E4	U4	0.30150
	E5	U5	0.36450
	E6	U6	0.50642
	E7	U7	0.39033
	E8	U8	0.48137
	E9	U9	0.50510
Disturbance	**E10**	W1	0.32420
	E11	W2	0.38879
	E12	W3	0.33576

For illustration purposes, you might check whether the model constraints put on the variances of Factor1–Factor3 are honored (although this should have been taken care of in the optimization). For example, the variance of Factor1 should be:

$$1 = (\text{Loading on FactorG})^2 + \text{Variance of E10}$$

Extracting the estimates from the output, you indeed verify the required equality, as shown in the following:

$$1.0000 = (0.8221)^2 + 0.32420$$

A Bifactor Model

A bifactor model (or a hierarchical factor model with two layers) for the same data set is now considered. In this model, the same set of factors as in the preceding higher-order factor model are used. The most notable difference is that the second-order factor FactorG in the higher-order factor model is no longer a factor of the first-order factors Factor1–Factor3. Instead, FactorG, like Factor1–Factor3, now also serves as a factor of the observed variable V1–V9. Unlike Factor1–Factor3, FactorG is considered to be a *general* factor in the sense that *all* observed variables have direct functional relationships with it. In contrast, Factor1–Factor3 are *group* factors in the sense that each of them has a direct functional relationship with only one group of *observed* variables. Because of the coexistence of a general factor and group factors at the same factor level, such a hierarchical model is also called a bifactor model.

The bifactor model is specified in the following statements:

```
proc calis corr data=Thurst method=max nobs=213 nose nostand;
   lineqs
      V1 = X11 * Factor1                      + X1g * FactorG + E1,
      V2 = X21 * Factor1                      + X2g * FactorG + E2,
      V3 = X31 * Factor1                      + X3g * FactorG + E3,
      V4 =            X42 * Factor2           + X4g * FactorG + E4,
      V5 =            X52 * Factor2           + X5g * FactorG + E5,
      V6 =            X62 * Factor2           + X6g * FactorG + E6,
      V7 =                      X73 * Factor3 + X7g * FactorG + E7,
      V8 =                      X83 * Factor3 + X8g * FactorG + E8,
      V9 =                      X93 * Factor3 + X9g * FactorG + E9;
   variance
      Factor1-Factor3 = 3 * 1.,
      FactorG         = 1. ,
      E1-E9           = U1-U9;
   cov
      Factor1-Factor3 FactorG = 6 * 0.;
   bounds
      0. <= U1-U9;
   fitindex ON(ONLY)=[chisq df probchi];
run;
```

In the LINEQS statement, there are only nine equations for the manifest variables in the model. Unlike the second-order factor model fitted previously, Factor1–Factor3 are no longer functionally related to FactorG and therefore there are no equations with Factor1–Factor3 as outcome variables.

The factor variances are all fixed at 1 in the VARIANCE statement. The variance parameters for E1–E9 are named U1–U9, respectively. The BOUNDS statement, again, is specified so that only positive estimates are accepted for error variance estimates.

All factors in the bifactor model are uncorrelated. In the COV statement, you specify that the six covariances among Factor1–Factor3 and FactorG are all zero. This specification is necessary because by default exogenous variables (excluding error terms) in the LINEQS model are correlated.

Like the previous PROC CALIS run, options are specified in the PROC CALIS and the FITINDEX statements to reduce the amount of default output.

There are more parameters in this model than in the preceding higher-order factor model, as shown in Output 33.26.5, which shows the optimization information.

Output 33.26.5 Optimization

Optimization Start			
Active Constraints	0	**Objective Function**	0.8380304146
Max Abs Gradient Element	2.4076251809	**Radius**	20.596787596

Iteration	Restarts	Function Calls	Active Constraints	Objective Function	Objective Function Change	Max Abs Gradient Element	Lambda	Ratio Between Actual and Predicted Change
1	0	5	0	0.70566	0.1324	0.4851	0.00140	0.148
2	0	7	0	0.30090	0.4048	0.3269	0	1.292
3	0	9	0	0.17403	0.1269	0.2947	0	0.985
4	0	11	0	0.11759	0.0564	0.0677	0	1.190
5	0	13	0	0.11455	0.00304	0.0267	0	1.043
6	0	15	0	0.11426	0.000285	0.00242	0	1.153
7	0	17	0	0.11423	0.000027	0.00168	0	1.394
8	0	19	0	0.11423	5.552E-6	0.000478	0	1.413
9	0	21	0	0.11423	1.154E-6	0.000335	0	1.420
10	0	23	0	0.11423	2.405E-7	0.000105	0	1.427
11	0	25	0	0.11423	5.016E-8	0.000068	0	1.432
12	0	27	0	0.11423	1.047E-8	0.000023	0	1.436
13	0	29	0	0.11423	2.184E-9	0.000014	0	1.439
14	0	31	0	0.11423	4.56E-10	4.909E-6	0	1.442

Optimization Results			
Iterations	14	**Function Calls**	34
Jacobian Calls	16	**Active Constraints**	0
Objective Function	0.1142278162	**Max Abs Gradient Element**	4.9090342E-6
Lambda	0	**Actual Over Pred Change**	1.4423519301
Radius	0.0002294218		

Convergence criterion (GCONV=1E-8) satisfied.

There are 27 parameters in the bifactor model: nine for the loadings on the group factors Factor1–Factor3, nine for the loadings on the general factor FactorG, and nine for the variances of errors E1–E9. The optimization converges in 14 iterations without problems.

A fit summary table is shown in Output 33.26.6

Output 33.26.6 Fit Summary

Fit Summary	
Chi-Square	24.2163
Chi-Square DF	18
Pr > Chi-Square	0.1481

The fit of this model is quite good. The chi-square value is 24.216, with $df = 18$ and $p = 0.148$. This is expected because the bifactor model has more parameters than the second-order factor model, which already

has a good fit with fewer parameters.

Estimation results are shown in Output 33.26.7. Estimates are left uninterpreted because they are not the main interest of this example.

Output 33.26.7 Estimation Results

Linear Equations

V1 =	-0.4879	Factor1	+	0.7679	FactorG	+	1.0000	E1
V2 =	-0.4523	Factor1	+	0.7909	FactorG	+	1.0000	E2
V3 =	-0.4045	Factor1	+	0.7536	FactorG	+	1.0000	E3
V4 =	0.6140	Factor2	+	0.6084	FactorG	+	1.0000	E4
V5 =	0.5058	Factor2	+	0.5973	FactorG	+	1.0000	E5
V6 =	0.3943	Factor2	+	0.5718	FactorG	+	1.0000	E6
V7 =	-0.7273	Factor3	+	0.5669	FactorG	+	1.0000	E7
V8 =	-0.2468	Factor3	+	0.6623	FactorG	+	1.0000	E8
V9 =	-0.4091	Factor3	+	0.5300	FactorG	+	1.0000	E9

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate
Latent	**Factor1**		1.00000
	Factor2		1.00000
	Factor3		1.00000
	FactorG		1.00000
Error	**E1**	U1	0.17236
	E2	U2	0.16984
	E3	U3	0.26848
	E4	U4	0.25281
	E5	U5	0.38735
	E6	U6	0.51757
	E7	U7	0.14966
	E8	U8	0.50039
	E9	U9	0.55175

One might ask whether this bifactor (hierarchical) model provides a significantly better fit than the previous second-order model. Can one use a chi-square difference test for nested models to answer this question? The answer is yes.

Although it is not obvious that the previous second-order factor model is nested within the current bifactor model, a general nested relationship between the higher-order factor and the hierarchical factor model is formally proved by Yung, Thissen, and McLeod (1999). Therefore, a chi-square difference test can be conducted using the following DATA step:

```
data _null_;
   df0 = 24; chi0 = 38.1963;
   df1 = 18; chi1 = 24.2163;
   diff = chi0-chi1;
   p = 1.-probchi(chi0-chi1,df0-df1);
   put 'Chi-square difference = ' diff;
   put 'p-value = ' p;
run;
```

The results are shown in the following:

Output 33.26.8 Chi-square Difference Test

```
Chi-square difference = 13.98
p-value = 0.0298603746
```

The chi-square difference is 13.98, with $df = 6$ and $p = 0.02986$. If α-level is set at 0.05, the bifactor model indicates a significantly better fit. But if α-level is set at 0.01, statistically the two models fit equally well to the data.

In the next section, it is demonstrated that the second-order factor model is indeed nested within the bifactor model, and hence the chi-square test conducted in the previous section is justified. In addition, through the demonstration of the nested relationship between the two classes of models, you can see how some parameter constraints in structural equation model can be set up in PROC CALIS.

For some practical researchers, the technical details involved in the next section might not be of interest and therefore could be skipped.

A Constrained Bifactor Model and Its Equivalence to the Second-Order Factor Model

To demonstrate that the second-order factor model is indeed nested within the bifactor model, a constrained bifactor model is fitted in this section. This constrained bifactor model is essentially the same as the preceding bifactor model, but with additional constraints on the factor loadings. Hence, the constrained bifactor model is nested within the unconstrained bifactor model.

Furthermore, if it can be shown that the constrained bifactor model is equivalent to the previous second-order factor, then the second-order factor model must also be nested within the unconstrained bifactor model. As a result, it justifies the chi-square difference test conducted in the previous section.

The construction of such a constrained bifactor model is based on Yung, Thissen, and McLeod (1999). In the following statements, a constrained bifactor model is specified.

```
proc calis corr data=Thurst method=max nobs=213 nose nostand;
   lineqs
      V1 = X11 * Factor1                          + X1g * FactorG + E1,
      V2 = X21 * Factor1                          + X2g * FactorG + E2,
      V3 = X31 * Factor1                          + X3g * FactorG + E3,
      V4 =              X42 * Factor2             + X4g * FactorG + E4,
      V5 =              X52 * Factor2             + X5g * FactorG + E5,
      V6 =              X62 * Factor2             + X6g * FactorG + E6,
      V7 =                          X73 * Factor3 + X7g * FactorG + E7,
      V8 =                          X83 * Factor3 + X8g * FactorG + E8,
      V9 =                          X93 * Factor3 + X9g * FactorG + E9;
   variance
      Factor1-Factor3 = 3 * 1.,
      FactorG         = 1. ,
      E1-E9           = U1-U9;
   cov
      Factor1-Factor3 FactorG = 6 * 0.;
   bounds
      0. <= U1-U9;
   fitindex ON(ONLY)=[chisq df probchi];
```

```
    parameters p1 (.5) p2 (.5) p3 (.5);
    /* Proportionality constraints */
    X1g = p1 * X11;
    X2g = p1 * X21;
    X3g = p1 * X31;
    X4g = p2 * X42;
    X5g = p2 * X52;
    X6g = p2 * X62;
    X7g = p3 * X73;
    X8g = p3 * X83;
    X9g = p3 * X93;
  run;
```

In this constrained model, you add a PARAMETERS statement and nine SAS programming statements to the previous bifactor model. In the PARAMETERS statement, three new independent parameters are added: p1, p2, and p3. These parameters represent the proportions that constrain the factor loadings of the observed variables on the group factors Factor1–Factor3 and the general factor FactorG. They are all free parameters and have initial values at 0.5. The next nine SAS programming statements represent the proportionality constraints imposed. For example, X1g–X3g are now dependent parameters expressed as functions of p1, X11, X21, and X31. Adding three new parameters (in the PARAMETERS statement) and redefining nine original parameters as dependent (in the SAS programming statements) is equivalent to adding six (=9-3) constraints to the original bifactor model. Mathematically, the additional statements in specifying the constrained bifactor model realizes the following six constraints:

$$\frac{\text{X1g}}{\text{X11}} = \frac{\text{X2g}}{\text{X21}} = \frac{\text{X3g}}{\text{X31}}$$

$$\frac{\text{X4g}}{\text{X42}} = \frac{\text{X5g}}{\text{X52}} = \frac{\text{X6g}}{\text{X62}}$$

$$\frac{\text{X7g}}{\text{X73}} = \frac{\text{X8g}}{\text{X83}} = \frac{\text{X9g}}{\text{X93}}$$

Obviously, with these six constraints the current constrained bifactor model is nested within the unconstrained version. What remains to be shown is that this constrained bifactor model is indeed equivalent to the previous second-order factor model. If so, the second-order factor model is also nested within the unconstrained bifactor model. One evidence for the equivalence of the current constrained bifactor model and the second-order factor model is from the fit summary table shown in Output 33.26.10. But first, it is also useful to consider the optimization information of the constrained bifactor model, which is shown in Output 33.26.9.

Output 33.26.9 Optimization

Optimization Start			
Active Constraints	0	**Objective Function**	6.8290849536
Max Abs Gradient Element	8.9903038299	**Radius**	46.912034073

Output 33.26.9 *continued*

Iteration	Restarts	Function Calls	Active Constraints	Objective Function	Objective Function Change	Max Abs Gradient Element	Lambda	Ratio Between Actual and Predicted Change
1 *	0	4	0	4.42090	2.4082	1.5979	0.195	0.121
2 *	0	6	0	2.42951	1.9914	2.4449	0.195	1.700
3 *	0	8	0	1.52546	0.9040	2.1675	0.781	1.657
4 *	0	10	0	0.86547	0.6600	1.2570	0.781	1.552
5 *	0	12	0	0.50425	0.3612	0.4075	0.781	1.351
6 *	0	14	0	0.41864	0.0856	1.1698	0.195	0.216
7 *	0	16	0	0.28298	0.1357	0.2355	0.0488	0.983
8	0	19	0	0.27010	0.0129	0.9602	0.00098	0.327
9	0	21	0	0.21798	0.0521	0.2433	0	0.751
10 *	0	23	0	0.20781	0.0102	0.0373	0.0488	0.943
11 *	0	26	0	0.20251	0.00530	0.0273	0.0918	1.064
12	0	28	0	0.19176	0.0108	0.0718	0.0515	1.008
13	0	30	0	0.18575	0.00601	0.1407	0.00393	0.562
14	0	32	0	0.18024	0.00551	0.0224	0	0.921
15	0	34	0	0.18017	0.000064	0.00161	0	1.067
16	0	36	0	0.18017	2.357E-6	0.000538	0	1.344
17	0	38	0	0.18017	3.389E-7	0.000216	0	1.380
18	0	40	0	0.18017	5.146E-8	0.000075	0	1.389
19	0	42	0	0.18017	7.886E-9	0.000031	0	1.391
20	0	44	0	0.18017	1.211E-9	0.000012	0	1.392

Optimization Results			
Iterations	20	**Function Calls**	47
Jacobian Calls	22	**Active Constraints**	0
Objective Function	0.1801712147	**Max Abs Gradient Element**	0.0000117927
Lambda	0	**Actual Over Pred Change**	1.3915929548
Radius	0.0002502182		

Convergence criterion (GCONV=1E-8) satisfied.

As shown Output 33.26.9, there are 21 independent parameters in the constrained bifactor model for the 45 “Functions (Observations).” These numbers match those of the second-order factor model exactly. The optimization shows some problems in initial iterations. The iteration numbers with asterisks indicate that the Hessian matrix is not positive definite in those iterations. But as long as the final converged iteration is not marked with an asterisk, the problems exhibited in early iterations do not raise any concern, as in the current case. Next, the fit summary is shown in Output 33.26.10.

Output 33.26.10 Model Fit

Fit Summary	
Chi-Square	38.1963
Chi-Square DF	24
Pr > Chi-Square	0.0331

In Output 33.26.10, the chi-square value in the fit summary table is 38.196, with $df = 24$, and $p = 0.033$. Again, these numbers match those of the second-order factor model exactly. These matches (same model fit with the same number of parameters) are necessary (but not sufficient) to show that the constrained bifactor model is equivalent to the second-order model. Stronger evidence is now presented.

In Output 33.26.11, estimation results of the constrained bifactor model are shown.

Output 33.26.11 Estimation Results

Linear Equations

V1 =	-0.5151	Factor1	+	0.7437	FactorG	+	1.0000	E1
V2 =	-0.5203	Factor1	+	0.7512	FactorG	+	1.0000	E2
V3 =	-0.4874	Factor1	+	0.7038	FactorG	+	1.0000	E3
V4 =	0.5211	Factor2	+	0.6534	FactorG	+	1.0000	E4
V5 =	0.4971	Factor2	+	0.6232	FactorG	+	1.0000	E5
V6 =	0.4381	Factor2	+	0.5493	FactorG	+	1.0000	E6
V7 =	0.4524	Factor3	+	0.6364	FactorG	+	1.0000	E7
V8 =	0.4173	Factor3	+	0.5869	FactorG	+	1.0000	E8
V9 =	0.4076	Factor3	+	0.5734	FactorG	+	1.0000	E9

Estimates for Variances of Exogenous Variables

Variable Type	Variable	Parameter	Estimate
Latent	**Factor1**		1.00000
	Factor2		1.00000
	Factor3		1.00000
	FactorG		1.00000
Error	**E1**	U1	0.18150
	E2	U2	0.16493
	E3	U3	0.26713
	E4	U4	0.30150
	E5	U5	0.36450
	E6	U6	0.50641
	E7	U7	0.39034
	E8	U8	0.48136
	E9	U9	0.50511

According to Yung, Thissen, and McLeod (1999), two models are equivalent if there is a one-to-one correspondence of the parameters in the models. This fact is illustrated for the constrained bifactor model and the second-order factor model.

First, the error variances for E1–E9 in the second-order factor model are transformed directly (using an identity map) to that of the bifactor models. The nine error variances in Output 33.26.4 for the second-order factor model match those of the constrained bifactor model exactly in Output 33.26.11. In addition, the variances of factors are fixed at one in both models. The error variances and the factor loadings at both factor levels in Output 33.26.4 for the second-order factor model are now transformed to yield the loading estimates in the constrained bifactor model.

Denote $\mathbf{P}_1$ as the first-order factor loading matrix, $\mathbf{P}_2$ as the second-order factor loading matrix, and $\mathbf{U}_1^2$ be

the matrix of variances for disturbances. That is, for the second-order factor model,

$$\mathbf{P}_1 = \begin{pmatrix} 0.9047 & 0 & 0 \\ 0.9138 & 0 & 0 \\ 0.8561 & 0 & 0 \\ 0 & 0.8358 & 0 \\ 0 & 0.7972 & 0 \\ 0 & 0.7026 & 0 \\ 0 & 0 & 0.7808 \\ 0 & 0 & 0.7202 \\ 0 & 0 & 0.7035 \end{pmatrix}$$

$$\mathbf{P}_2 = \begin{pmatrix} 0.8221 \\ 0.7818 \\ 0.8150 \end{pmatrix}$$

$$\mathbf{U}_1^2 = \begin{pmatrix} 0.3242 & 0 & 0 \\ 0 & 0.3888 & 0 \\ 0 & 0 & 0.3358 \end{pmatrix}$$

According to Yung, Thissen, and McLeod (1999), the transformation to obtain the estimates in the equivalent constrained bifactor model is:

$$\begin{aligned} \mathbf{L}_1 &= \mathbf{P}_1\mathbf{U}_1 \\ \mathbf{L}_2 &= \mathbf{P}_1\mathbf{P}_2 \end{aligned}$$

where $\mathbf{L}_1$ is the matrix of the first-layer factor loadings (that is, loadings on group factors Factor1–Factor3), and $\mathbf{L}_2$ is the matrix of the second-layer factor loadings (that is, loadings on FactorG) in the constrained bifactor model. Carrying out the matrix calculations for $\mathbf{L}_1$ and $\mathbf{L}_2$ shows that:

$$\mathbf{L}_1 = \begin{pmatrix} 0.5151 & 0 & 0 \\ 0.5203 & 0 & 0 \\ 0.4875 & 0 & 0 \\ 0 & 0.5212 & 0 \\ 0 & 0.4971 & 0 \\ 0 & 0.4381 & 0 \\ 0 & 0 & 0.4525 \\ 0 & 0 & 0.4173 \\ 0 & 0 & 0.4077 \end{pmatrix}$$

$$\mathbf{L}_2 = \begin{pmatrix} 0.7438 \\ 0.7512 \\ 0.7038 \\ 0.6534 \\ 0.6232 \\ 0.5493 \\ 0.6364 \\ 0.5870 \\ 0.5734 \end{pmatrix}$$

With very minor numerical differences and ignorable sign changes, these transformation results match the estimated loadings observed in Output 33.26.11 for the constrained bifactor model. Therefore, the second-order factor model is shown to be equivalent to the constrained bifactor model, and hence is nested within the unconstrained bifactor model.

Example 33.27: Linear Relations among Factor Loadings

(View the complete code for this example (calex09.sas) in the example repository.)

In this example, you use the FACTOR modeling language of PROC CALIS to specify a confirmatory factor analysis model with linear constraints on loadings. You use SAS programming statements to set the constraints. This example also discusses the differences between fitting covariance structures and correlation structures in the current modeling context.

The correlation matrix of six variables from Kinzer and Kinzer (N=326) is used by Guttman (1957) as an example that yields an approximate simplex. McDonald (1980) uses this data set as an example of factor analysis where he assumes that the loadings on the second factor are linear functions of the loadings on the first factor. Let $\mathbf{B}$ be the factor loading matrix containing the two factors and six variables so that:

$$\mathbf{B} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \\ b_{41} & b_{42} \\ b_{51} & b_{52} \\ b_{61} & b_{62} \end{pmatrix}$$

and

$$b_{j2} = \alpha + \beta b_{j1}, \quad j = 1, \ldots, 6$$

The correlation structures are represented by:

$$\mathbf{P} = \mathbf{B}\mathbf{B}' + \mathbf{\Psi}$$

where $\mathbf{\Psi} = \mathrm{diag}(\psi_{11}, \psi_{22}, \psi_{33}, \psi_{44}, \psi_{55}, \psi_{66})$ represents the diagonal matrix of unique variances for the variables.

With parameters α and β being unconstrained, McDonald (1980) has fitted an underidentified model with seven degrees of freedom. Browne (1982) imposes the following identification condition:

$$\beta = -1$$

In this example, Browne's identification condition is imposed. The following is the specification of the confirmatory factor model using the FACTOR modeling language.

```
data kinzer(type=corr);
title "Data Matrix of Kinzer & Kinzer, see GUTTMAN (1957)";
   _type_ = 'corr';
   input _name_ $ var1-var6;
   datalines;
var1  1.00   .     .     .     .     .
```

```
var2   .51  1.00   .     .     .     .
var3   .46   .51  1.00   .     .     .
var4   .46   .47   .54  1.00   .     .
var5   .40   .39   .49   .57  1.00   .
var6   .33   .39   .47   .45   .56  1.00
;

proc calis data=kinzer nobs=326 nose;
   factor
      factor1 ===> var1-var6   = b11 b21 b31 b41 b51 b61 (6 *.6),
      factor2 ===> var1-var6   = b12 b22 b32 b42 b52 b62;
   pvar
      factor1-factor2 = 2 * 1.,
      var1-var6       = psi1-psi6 (6 *.3);
   cov
      factor1 factor2 = 0.;
   parameters alpha (1.);
   /* SAS Programming Statements to define dependent parameters */
   b12 = alpha - b11;
   b22 = alpha - b21;
   b32 = alpha - b31;
   b42 = alpha - b41;
   b52 = alpha - b51;
   b62 = alpha - b61;
   fitindex on(only)=[chisq df probchi];
run;
```

In the FACTOR statement, you specify two factors, named factor1 and factor2, for the variables. In this model, all manifest variables have nonzero loadings on the two factors. These loading parameters are specified after the equal signs and are named with the prefix 'b.' You specify the initial estimates in the parentheses for the parameters in the first entry of the FACTOR statement. The loadings in the first entry are all free parameters with initial estimates of .6. In the second entry of the FACTOR statement, you specify the Loadings of var1–var6 on factor2. However, these parameters are dependent, as shown in the SAS programming statements. Initial values for these dependent parameters are thus unnecessary.

In the PVAR statement, the factor variances are fixed at ones, while the error variances of the variables are free parameters named psi1–psi6. Again, you provide initial estimates for these error variance parameters. All have the initial value of 0.3.

An additional parameter alpha is specified in the PARAMETERS statement with an initial value of 1. Then, you use six SAS programming statements to define the loadings on the second factor as functions of the loadings on the first factor. Lastly, the FITINDEX statement is used to trim the results in the fit summary table.

In the specification, there are twelve loadings in the FACTOR statement and six error variances in the PVAR statement. Adding the parameter alpha in the list, there are 19 parameters in total. However, the loading parameters are not all independent of each other. As defined in the SAS programming statements, six loadings are dependent. This reduces the number of free parameters to 13. Hence the degrees of freedom for the model is 8 = 21 – 13. Notice that the factor variances are fixed at 1, as specified in the PVAR statement, and covariance among the two factors is fixed at zero, as specified in the COV statement.

Output 33.27.1 shows a concise fit summary table. The chi-square test statistic of model fit is 10.337 with *df* = 8 (p = 0.242). This indicates a good model fit.

Output 33.27.1 Fit of the Correlation Structures

Fit Summary	
Chi-Square	10.3374
Chi-Square DF	8
Pr > Chi-Square	0.2421

The estimated factor loading matrix is presented in Output 33.27.2, and the estimated error variances and the estimate for alpha are presented in Output 33.27.3.

Output 33.27.2 Loading Estimates

Factor Loading Matrix		
	factor1	**factor2**
var1	0.3609 [b11]	0.6174 [b12]
var2	0.3212 [b21]	0.6571 [b22]
var3	0.4859 [b31]	0.4923 [b32]
var4	0.5745 [b41]	0.4038 [b42]
var5	0.7985 [b51]	0.1797 [b52]
var6	0.6736 [b61]	0.3046 [b62]

Output 33.27.3 Unique Variances and the Additional Parameter

Error Variances		
Variable	**Parameter**	**Estimate**
var1	psi1	0.53035
var2	psi2	0.44986
var3	psi3	0.48756
var4	psi4	0.47278
var5	psi5	0.31124
var6	psi6	0.53815

Additional Parameters		
Type	**Parameter**	**Estimate**
Independent	alpha	0.97825

All these estimates are essentially the same as those reported in Browne (1982). Notice that there are no standard error estimates in the output, as requested by the NOSTDERR option in the PROC CALIS statement. Standard error estimates are not of interest in this example.

In fitting the preceding factor model, wrong covariance structures rather than the intended correlation structures have been specified. As pointed out by Browne (1982), fitting such covariance structures directly is not entirely appropriate for analyzing correlations. For example, when fitting the correlation structures, the diagonal elements of **P** must always be fixed ones. This fact has never been enforced in the preceding

specification. A simple check of the estimates will illustrate the problem. In Output 33.27.2, the loading estimates of VAR1 on the two factors are 0.3609 and 0.6174, respectively. In Output 33.27.3, the error variance estimate for VAR1 is 0.53036. The fitted variance of VAR1 can therefore be computed by the following equation:

$$\text{fitted variance} = 0.3609^2 + 0.6174^2 + 0.53036 = 1.0418$$

This fitted value is quite a bit off from 1.00, as required for the standardized variance of VAR1.

Fortunately, even though the wrong covariance structure model has been analyzed, the preceding analysis is not completely useless. For the current confirmatory factor model, according to Browne (1982) the estimates obtained from fitting the wrong covariance structure model are still consistent (as if they were estimating the population parameters in the correlation structures). However, the chi-square test statistic as reported previously is not correct.

Note that using the CORR option in the PROC CALIS statement will not solve the problem. By specifying the CORR option you merely request PROC CALIS to use the correlation matrix directly as a covariance matrix in the objective function for model fitting. It still would not constrain the fitting of the diagonal elements to 1 during estimation.

In the next section, a solution to the correlation analysis problem is suggested. It is not claimed that this is the only solution or the best solution. Alternative treatments of the problem are possible.

Fitting the Correct Correlation Structures

This main idea of this solution is to embed the intended correlation structures (with correct constraints on the diagonal elements of the correlation matrix) into a covariance structure model so that the estimation methods of PROC CALIS can be applied legitimately to the specially constructed covariance structures.

First, the issue of the fixed ones on the diagonal of the correlation structure model is addressed. That is, the diagonal elements of the correlation structures represented by $(\mathbf{BB}' + \mathbf{\Psi})$ must be fitted by ones. This can be accomplished by constraining the error variances as dependent parameters of the loadings, as shown in the following:

$$\mathbf{\Psi}_{jj} = 1. - b_{j1}^2 - b_{j2}^2, \quad j = 1, \ldots, 6$$

Other constraints might also serve the purpose, but the proposed constraints here are the most convenient and intuitive.

Now, due to the fact that discrepancy functions used in PROC CALIS are derived for covariance matrices rather than correlation matrices, PROC CALIS is essentially set up for analyzing covariance structures (with or without mean structures), but not correlation structures. Hence, the statistical theory behind PROC CALIS applies to covariance structure analysis, but it might not generalize to correlation structure analysis in all situations. Despite that, with some manipulations PROC CALIS can fit the correct correlation structures to the current data without compromising the statistical theory. These manipulations are now discussed. Recall that the correlation structures are represented by:

$$\mathbf{P} = \mathbf{BB}' + \mathbf{\Psi}$$

As before, in the $\mathbf{B}$ matrix, there are six linear constraints on the factor loadings. In addition, the diagonal elements of $(\mathbf{BB}' + \mathbf{\Psi})$ are constrained to ones, as done by defining the error variances as dependent parameters of the loadings in the preceding equation. To analyze the correlation structures by using PROC

CALIS, a covariance structure model with such correlation structures embedded is now specified. That is, the covariance structure to be fitted by PROC CALIS is as follows:

$$\mathbf{\Sigma} = \mathbf{DPD}' = \mathbf{D}(\mathbf{BB}' + \mathbf{\Psi})\mathbf{D}'$$

where $\mathbf{D}$ is a 6 x 6 diagonal matrix containing the population standard deviations for the manifest variables. Theoretically, it is legitimate that you analyze this covariance structure model for studying the embedded correlation structures. In addition, it does not matter whether your input matrix is a correlation or covariance matrix, or any rescaled covariance matrix (by multiplying any variables by any positive constants). You would get correct results if you could somehow specify these covariance structures correctly in PROC CALIS. However, there seems to be nowhere in PROC CALIS that you can specify the diagonal matrix $\mathbf{D}$ for the population standard deviations. So what can one do with this formulation? The answer is to rewrite the covariance structure model in a form similar to the usual confirmatory factor model, as presented in the following.

Let $\mathbf{T} = \mathbf{DB}$ and $\mathbf{K} = \mathbf{D\Psi D}'$. The covariance structure model of interest can now be rewritten as:

$$\mathbf{\Sigma} = \mathbf{TT}' + \mathbf{K}$$

This form of covariance structures implies a confirmatory factor model with factor loading matrix $\mathbf{T}$ and error covariance matrix $\mathbf{K}$. This confirmatory factor model can certainly be specified using the FACTOR modeling language, in much the same way you specify a confirmatory factor model in the preceding section. However, because you are actually more interested in estimating the basic set of parameters in matrices $\mathbf{B}$ and $\mathbf{\Psi}$ of the embedded correlation structures, you would define the model parameters as functions of this basic set of parameters of interest. This can be accomplished by using the PARAMETERS and the SAS programming statements.

All in all, you can use the following statements to set up such a confirmatory factor model with the desired correlation structures embedded.

```
proc calis data=Kinzer nobs=326 nose;
   factor
      factor1 ===> var1-var6   = t11 t21 t31 t41 t51 t61,
      factor2 ===> var1-var6   = t12 t22 t32 t42 t52 t62;
   pvar
      factor1-factor2 = 2 * 1.,
      var1-var6       = k1-k6;
   cov
      factor1 factor2 = 0.;
   parameters alpha (1.) d1-d6 (6 * 1.)
              b11 b21 b31 b41 b51 b61 (6 *.6),
              b12 b22 b32 b42 b52 b62
              psi1-psi6;
   /* SAS Programming Statements */
   /* 12 Constraints on Correlation structures */
   b12  = alpha - b11;
   b22  = alpha - b21;
   b32  = alpha - b31;
   b42  = alpha - b41;
   b52  = alpha - b51;
   b62  = alpha - b61;
   psi1 = 1. - b11 * b11 - b12 * b12;
   psi2 = 1. - b21 * b21 - b22 * b22;
```

```
    psi3 = 1. - b31 * b31 - b32 * b32;
    psi4 = 1. - b41 * b41 - b42 * b42;
    psi5 = 1. - b51 * b51 - b52 * b52;
    psi6 = 1. - b61 * b61 - b62 * b62;
    /* Defining Covariance Structure Parameters */
    t11  = d1 * b11;
    t21  = d2 * b21;
    t31  = d3 * b31;
    t41  = d4 * b41;
    t51  = d5 * b51;
    t61  = d6 * b61;
    t12  = d1 * b12;
    t22  = d2 * b22;
    t32  = d3 * b32;
    t42  = d4 * b42;
    t52  = d5 * b52;
    t62  = d6 * b62;
    k1   = d1 * d1 * psi1;
    k2   = d2 * d2 * psi2;
    k3   = d3 * d3 * psi3;
    k4   = d4 * d4 * psi4;
    k5   = d5 * d5 * psi5;
    k6   = d6 * d6 * psi6;
    fitindex on(only)=[chisq df probchi];
  run;
```

First, you notice that specifications in the FACTOR and the PVAR statements are essentially unchanged from the previous specification, except that the parameters are named differently here to reflect different model matrices. In the current specification, the factor loading parameters in matrix $\mathbf{T}$ are named with prefix 't,' and the error variance parameters in matrix $\mathbf{K}$ are named with prefix 'k.' Specification of these parameters reflects the covariance structures. As you see in the last block of the SAS programming statements statements, all these parameters are functions of the correlation structure parameters in $\mathbf{B}$, $\mathbf{\Psi}$, and $\mathbf{D}$.

Next, in the PARAMETERS statement, all correlation structure parameters are defined with initial values provided. These are the parameters of interest: alpha is used to define dependencies among loadings, d's are the population standard deviations, b's are the loading parameters, and psi's are the error variance parameters. There are 25 parameters specified in this statement, but not all of them are free or independent.

In the first block of SAS programming statements, parameter dependencies or constraints on the correlation structures are specified. The first six statements realize the required linear relations among the factor loadings:

$$b_{j2} = \alpha - b_{j1}, \quad j = 1, \ldots, 6$$

The next six statements constrain the error variances so as to ensure that an embedded correlation structure model is being fitted. That is, each error variance is dependent on the corresponding loadings, as prescribed by the following equation:

$$\mathbf{\Psi}_{jj} = 1. - b_{j1}^2 - b_{j2}^2, \quad j = 1, \ldots, 6$$

These twelve constraints reduce the number of independent parameters to 13, as expected.

The next block of SAS programming statements are essentially for relating the correlation structure parameters to the covariance structures that are specified in the FACTOR and the PVAR statements. These SAS programming statements realize the required relations: $\mathbf{T} = \mathbf{DB}$ and $\mathbf{K} = \mathbf{D\Psi D}'$, but in non-matrix forms:

$$t_{ji} = d_j b_{ji} \quad (j = 1, \ldots, 6; \quad i = 1, 2)$$

$$k_{jj} = d_j d_j \mathbf{\Psi}_{jj} \quad (j = 1, \ldots, 6)$$

where d_j denotes the jth diagonal element of $\mathbf{D}$.

The fit summary is presented in Output 33.27.4. The chi-square test statistic is 14.63 with $df = 8$ ($p = 0.067$). This shows that the previous chi-square test based on fitting a wrong covariance structure model is indeed questionable.

Output 33.27.4 Model Fit of the Correlation Structures

Fit Summary	
Chi-Square	14.6269
Chi-Square DF	8
Pr > Chi-Square	0.0668

Estimates of the loadings and error variances are presented in Output 33.27.5. These estimates are for the covariance structure model with loading matrix $\mathbf{T}$ and error covariance matrix $\mathbf{K}$. They are rescaled versions of the correlation structure parameters and are not of primary interest themselves.

Output 33.27.5 Estimates of Loadings and Error Variances

Factor Loading Matrix		
	factor1	**factor2**
var1	0.3448	0.6367
	[t11]	[t12]
var2	0.3200	0.6512
	[t21]	[t22]
var3	0.4873	0.4778
	[t31]	[t32]
var4	0.5703	0.3948
	[t41]	[t42]
var5	0.7741	0.1964
	[t51]	[t52]
var6	0.6778	0.3126
	[t61]	[t62]

Factor Covariance Matrix		
	factor1	**factor2**
factor1	1.0000	0
factor2	0	1.0000

Error Variances		
Variable	**Parameter**	**Estimate**
var1	k1	0.49119
var2	k2	0.46780
var3	k3	0.51597
var4	k4	0.50070
var5	k5	0.35505
var6	k6	0.47685

The parameter estimates of the embedded correlation structures are shown in Output 33.27.6 as “additional”

parameters.

Output 33.27.6 Estimates of Correlation Structure Parameters

Additional Parameters		
Type	**Parameter**	**Estimate**
Independent	alpha	0.97400
	d1	1.00771
	d2	0.99712
	d3	0.99078
	d4	0.99085
	d5	0.99640
	d6	1.01687
	b11	0.34217
	b21	0.32095
	b31	0.49179
	b41	0.57553
	b51	0.77686
	b61	0.66659
Dependent	b12	0.63183
	b22	0.65305
	b32	0.48222
	b42	0.39848
	b52	0.19714
	b62	0.30742
	psi1	0.48371
	psi2	0.47051
	psi3	0.52561
	psi4	0.50998
	psi5	0.35762
	psi6	0.46116

Except for the population standard deviation parameter d's, all other parameters estimated in the current model can be compared with those from the previous fitting of an incorrect covariance structure model. Although estimates in the current model do not differ very much from those in the previous specification, it is at least reassuring that they are obtained from fitting a correctly specified covariance structure model with the intended correlation structures embedded.

Example 33.28: Multiple-Group Model for Purchasing Behavior

(View the complete code for this example (calex10.sas) in the example repository.)

In this example, data were collected from customers who made purchases from a retail company during years 2002 and 2003. A two-group structural equation model is fitted to the data.

The variables are:

Spend02:	total purchase amount in 2002
Spend03:	total purchase amount in 2003
Courtesy:	rating of the courtesy of the customer service
Responsive:	rating of the responsiveness of the customer service
Helpful:	rating of the helpfulness of the customer service
Delivery:	rating of the timeliness of the delivery
Pricing:	rating of the product pricing
Availability:	rating of the product availability
Quality:	rating of the product quality

For the ratings scales, nine-point scales were used. Customers could respond 1 to 9 on these scales, with 1 representing "extremely unsatisfied" and 9 representing "extremely satisfied." Data were collected from two different regions, which are labeled as Region 1 ($N = 378$) and Region 2 ($N = 423$), respectively. The ratings were collected at the end of year 2002 so that they represent customers' purchasing experience in year 2002.

The central questions of the study are:

- How does the overall customer service affect the current purchases and predict future purchases?
- How does the overall product quality affect the current purchases and predict future purchases?
- Do current purchases predict future purchases?
- Do the two regions have different structural models for predicting the purchases?

In stating these research questions, you use several constructs that might or might not correspond to objective measurements. Current and future purchases are certainly measurable directly by the spending of the customers. That is, because customer service and product satisfaction and quality were surveyed between 2002 and 2003, Spend02 represents current purchases and Spend03 represents future purchases in the study. Both variables Spend02 and Spend03 are objective measurements without measurement errors. All you need to do is to extract the information from the transaction records. But how about hypothetical constructs such as customer service quality and product quality? How would you measure them in the model?

In measuring these hypothetical constructs, you might ask customers' perception about the service or product quality directly in a single question. A simple survey with two questions about the customer service and product qualities could then be what you need. These questions are called indicators (or indicator variables) of the underlying constructs. However, using just one indicator (question) for each of these hypothetical

constructs would be quite unreliable—that is, measurement errors might dominate in the data collection process. Therefore, multiple indicators are usually recommended for measuring such hypothetical constructs.

There are two main advantages of using multiple indicators for hypothetical constructs. The first one is conceptual and the other is statistical and mathematical.

First, hypothetical constructs might conceptually be multifaceted themselves. Measuring a hypothetical construct by a single indicator does not capture the full meaning of the construct. For example, the product quality might refer to the durability of the product, the outlook of the product, the pricing of the product, and the availability of product, among others. The customer service quality might refer to the politeness of the customer service, the timeliness of the delivery, and the responsiveness of customer services, among others. Therefore, multiple indicators for a single hypothetical construct might be necessary if you want to cover the multifaceted aspects of a given hypothetical construct.

Second, from a statistical point of view, the reliability would be higher if you combine correlated indicators for a construct than if you use a single indicator only. Therefore, combining correlated indicators would lead to more accurate and reliable results.

One way to combine correlated indicators is to use a simple sum of them to represent the underlying hypothetical construct. However, a better way is to use the structural equation modeling technique that represents each indicator (variable) as a function of the underlying hypothetical construct plus an error term. In structural equation modeling, hypothetical constructs are constructed as latent factors, which are unobserved systematic (that is, non-error) variables. Theoretically, latent factors are free from measurement errors, and so the estimation through the structural equation modeling technique is more accurate than if you just use simple sums of indicators to represent hypothetical constructs. Therefore, a structural equation modeling approach is the method of the choice in the current analysis.

In practice, you must also make sure that there are enough indicators for the identification of the underlying latent factor, and hence the identification of the entire model. Necessary and sufficient rules for identification are very complicated to describe and are out of the scope of the current discussion (however, see Bollen 1989b for discussions of identification rules for various classes of structural equation models). Some simple rules of thumb might be useful as a guidance. For example, for unconstrained situations, you should at least have three indicators (variables) measured for a latent factor. Unfortunately, these rules of thumb do not guarantee identification in every case.

In this example, Service and Product are latent factors in the structural equation model which represent service and product qualities, respectively. In the model, these two latent factors are reflected by the ratings of the customers. Ratings on the Courtesy, Responsive, Helpful, and Delivery scales are indicators of Service. Ratings on the Pricing, Availability, and Quality scales are indicators of Product (that is, product quality).

A Path Diagram

A path diagram shown in Figure 33.27 represents the structural equation model for the purchase behavior. Observed or manifest variables are represented by rectangles, and latent variables are represented by ovals. As mentioned, two latent variables (factors), Service and Product, are created as overall measures of customer service and product qualities, respectively.

Figure 33.27 Path Diagram of Purchasing Behavior

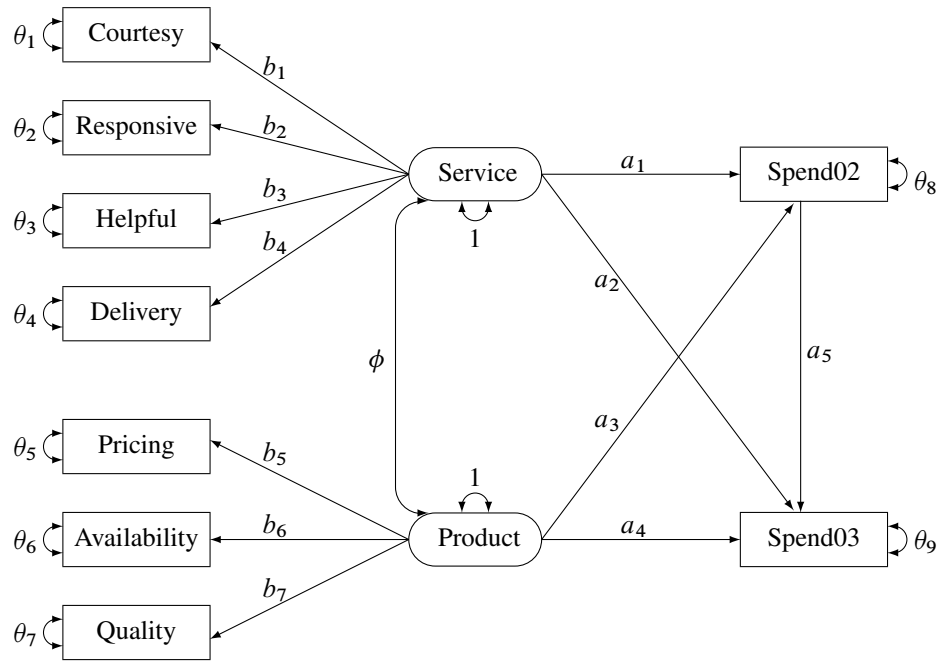

The left part of the diagram represents the measurement model of the latent variables. The Service factor has four indicators: Courtesy, Responsive, Helpful, and Delivery. The path coefficients to these observed variables from the Service factor are b_1, b_2, b_3, and b_4, respectively. Similarly, the Product variable has three indicators: Pricing, Availability, and Quality, with path coefficients b_5, b_6, and b_7, respectively.

The two latent factors are predictors of the purchase amounts Spend02 and Spend03. In addition, Spend02 also serves as a predictor of Spend03. Path coefficients (effects) for this part of functional relationships are represented by a1–a5 in the diagram.

Each variable in the path diagram has a variance parameter. For endogenous or dependent variables, which serve as outcome variables in the model, the variance parameters are the error variances that are not accounted for by the predictors. For example, in the current model all observed variables are endogenous variables. The double-headed arrows that are attached to these variables represent error variances. In the diagram, θ_1 to θ_9 are the names of these error variance parameters. For exogenous or independent variables, which never serve as outcome variables in the model, the variance parameters are the (total) variances of these variables. For example, in the diagram the double-headed arrows that are attached to Service and Product represent the variances of these two latent variables. In the current model, both of these variances are fixed at one.

When the double-headed arrows point to two variables, they represent covariances in the path diagram. For example, in Figure 33.27 the covariance between Service and Product is represented by the parameter ϕ.

The Basic Path Model Specification

For the moment, it is hypothesized that both 'Region 1' and 'Region 2' data are fitted by the same model as shown in Figure 33.27. Once the path diagram is drawn, it is readily translated into the PATH modeling language. See the PATH statement on page 1665 for details about how to use the PATH modeling language to specify structural equation models.

To represent all the features in the path diagram in the PATH model language, you can use the following specification:

```
path
   Service ===> Spend02      = a1,
   Service ===> Spend03      = a1,
   Product ===> Spend02      = a3,
   Product ===> Spend03      = a4,
   Spend02 ===> Spend03      = a5,
   Service ===> Courtesy     = b1,
   Service ===> Responsive   = b2,
   Service ===> Helpful      = b3,
   Service ===> Delivery     = b4,
   Product ===> Pricing      = b5,
   Product ===> Availability = b6,
   Product ===> Quality      = b7;
pvar
   Courtesy Responsive Helpful
   Delivery Pricing
   Availability Quality = theta01-theta07,
   Spend02 = theta08,
   Spend03 = theta09,
   Service Product = 2 * 1.;
pcov
   Service Product = phi;
```

The PATH statement captures all the path coefficient specifications and the direction of the paths (single-headed arrows) in the path diagram. The first five paths define how Spend02 and Spend03 are predicted from the latent variables Service, Product, and Spend02. The next seven paths define the measurement model, which shows how the latent variables in the model relate to the observed indicator variables.

The PVAR statement captures the specification of the error variances and the variances of exogenous variables (that is, the double-headed arrows in the path diagram). The PCOV statement captures the specification of covariance between the two latent variables in the model (which is represented by the double-headed arrow that connects Service and Product in the path diagram).

You can also use the following simpler version of the PATH model specification for the path diagram:

```
path
   Service ===> Spend02  Spend03      ,
   Product ===> Spend02  Spend03      ,
   Spend02 ===> Spend03               ,
   Service ===> Courtesy Responsive
                Helpful Delivery      ,
   Product ===> Pricing  Availability
                Quality               ;
pvar
```

```
   Courtesy Responsive Helpful Delivery Pricing
   Availability Quality Spend02 Spend03,
   Service Product = 2 * 1.;
pcov
   Service Product;
```

There are two simplifications in this PATH model specification. First, you do not need to specify the parameter names if they are unconstrained in the model. For example, parameter a1 in the model is unique to the path effect from Service to Spend02. You do not need to name this effect because it is not constrained to be the same as any other parameter in the model. Similar, all the path coefficients (effects), error variances, and covariances in the path diagram are not constrained. Therefore, you can omit all the corresponding parameter name specifications in the PATH model specification. The only exceptions are the variances of Service and Product. Both are fixed constants 1 in the path diagram, and so you must specify them explicitly in the PVAR statement.

Second, you use a condensed way to specify the paths. In the first three path entries of the PATH statement, you specify how Spend02 and Spend03 are predicted from the latent variables Service, Product, and Spend02. Notice that in each path entry, you can define more than one path (single-headed arrow relationship). For example, in the first path entry, you specify two paths: one is **Service===>Spend02** and the other is **Service===>Spend03**. In the last two path entries of the PATH statement, you define the relationships between the two latent constructs Spend and Service and their measured indicators. Each of these path entries specifies multiple paths (single-headed arrow relationships).

You use this simplified PATH specifications in the subsequent analysis.

A Restrictive Model with Invariant Mean and Covariance Structures

In this section, you fit a mean and covariance structure model to the data from two regions, as shows in the following DATA steps:

```
data region1(type=cov);
   input _type_ $6. _name_ $12. Spend02 Spend03 Courtesy Responsive
         Helpful Delivery Pricing Availability Quality;
   datalines;
COV   Spend02      14.428  2.206  0.439 0.520 0.459 0.498 0.635 0.642 0.769
COV   Spend03       2.206 14.178  0.540 0.665 0.560 0.622 0.535 0.588 0.715
COV   Courtesy      0.439  0.540  1.642 0.541 0.473 0.506 0.109 0.120 0.126
COV   Responsive    0.520  0.665  0.541 2.977 0.582 0.629 0.119 0.253 0.184
COV   Helpful       0.459  0.560  0.473 0.582 2.801 0.546 0.113 0.121 0.139
COV   Delivery      0.498  0.622  0.506 0.629 0.546 3.830 0.120 0.132 0.145
COV   Pricing       0.635  0.535  0.109 0.119 0.113 0.120 2.152 0.491 0.538
COV   Availability  0.642  0.588  0.120 0.253 0.121 0.132 0.491 2.372 0.589
COV   Quality       0.769  0.715  0.126 0.184 0.139 0.145 0.538 0.589 2.753
MEAN     .        183.500 301.921 4.312 4.724 3.921 4.357 6.144 4.994 5.971
;

data region2(type=cov);
   input _type_ $6. _name_ $12. Spend02 Spend03 Courtesy Responsive
         Helpful Delivery Pricing Availability Quality;
   datalines;
COV   Spend02       14.489   2.193 0.442 0.541 0.469 0.508 0.637 0.675 0.769
COV   Spend03        2.193  14.168 0.542 0.663 0.574 0.623 0.607 0.642 0.732
COV   Courtesy       0.442   0.542 3.282 0.883 0.477 0.120 0.248 0.283 0.387
```

```
COV   Responsive     0.541   0.663 0.883 2.717 0.477 0.601 0.421 0.104 0.105
COV   Helpful        0.469   0.574 0.477 0.477 2.018 0.507 0.187 0.162 0.205
COV   Delivery       0.508   0.623 0.120 0.601 0.507 2.999 0.179 0.334 0.099
COV   Pricing        0.637   0.607 0.248 0.421 0.187 0.179 2.512 0.477 0.423
COV   Availability   0.675   0.642 0.283 0.104 0.162 0.334 0.477 2.085 0.675
COV   Quality        0.769   0.732 0.387 0.105 0.205 0.099 0.423 0.675 2.698
MEAN     .         156.250 313.670 2.412 2.727 5.224 6.376 7.147 3.233 5.119
;
```

To include the analysis of the mean structures, you need to introduce the mean and intercept parameters in the model. Although various researchers propose some representation schemes that include the mean parameters in the path diagram, the mean parameters are not depicted in Figure 33.27. The reason is that representing the mean and intercept parameters in the path diagram would usually obscure the "causal" paths, which are of primary interest. In addition, it is a simple matter to specify the mean and intercept parameters in the MEAN statement without the help of a path diagram when you follow these principles:

- Each variable in the path diagram has a mean parameter that can be specified in the MEAN statement. For an exogenous variable, the mean parameter refers to the variable mean. For an endogenous variable, the mean parameter refers to the intercept of the variable.
- The means of exogenous observed variables are free parameters by default. The means of exogenous latent variables are fixed zeros by default.
- The intercepts of endogenous observed variables are free parameters by default. The intercepts of endogenous latent variables are fixed zeros by default.
- The total number of mean parameters should not exceed the number of observed variables.

Because all nine observed variables are endogenous (each has at least one single-headed arrow pointing to it) in the path diagram, you can specify these nine intercepts in the MEAN statement, as shown in the following specification:

```
mean
   Courtesy Responsive Helpful Delivery Pricing
   Availability Quality Spend02 Spend03;
```

However, the intercepts of endogenous observed variables are already free parameters by default and this MEAN statement specification is not necessary for the current situation. For the means of the latent variables Service and Product, you do not have any theoretical reasons to set them other than the default fixed zero. Hence, you do not need to set these mean parameters explicitly either. Consequently, to include the analysis of the mean structures with these default mean parameters, all you need to specify the MEANSTR option in the PROC CALIS statement, as shown in the following specification of the fitting of a constrained two-group model for the purchase data:

```
proc calis meanstr;
   group 1 / data=region1 label="Region 1" nobs=378;
   group 2 / data=region2 label="Region 2" nobs=423;
   model 1 / group=1,2;
      path
         Service ===> Spend02  Spend03      ,
         Product ===> Spend02  Spend03      ,
         Spend02 ===> Spend03               ,
```

```
            Service ===> Courtesy Responsive
                         Helpful Delivery      ,
            Product ===> Pricing  Availability
                         Quality               ;
         pvar
            Courtesy Responsive Helpful Delivery Pricing
            Availability Quality Spend02 Spend03,
            Service Product = 2 * 1.;
         pcov
            Service Product;
   run;
```

You use the GROUP statements to specify the data for the two regions. Using the DATA= options in the GROUP statements, you assign the 'Region 1' data to Group 1 and the 'Region 2' data to Group 2. You label the two groups by the LABEL= options. Because the number of observations is not defined in the data sets, you use the NOBS= options in the GROUP statements to provide this information.

In the MODEL statement, you specify in the GROUP= option that both Groups 1 and 2 are fitted by the same model—**`model 1`**. Next, the path model is specified. As discussed before, you do not need to specify the default mean parameters by using the MEAN statement because the MEANSTR option in the PROC CALIS statement already indicates the analysis of mean structures.

Output 33.28.1 presents a summary of modeling information. Each group is listed with its associated data set, number of observations, and its corresponding model and the model type. In the current analysis, the same model is fitted to both groups. Next, a table for the types of variables is presented. As intended, all nine observed (manifest) variables are endogenous, and all latent variables are exogenous in the model.

Output 33.28.1 Modeling Information and Variables

Modeling Information

Maximum Likelihood Estimation

Group	Label	Data Set	N Obs	Model	Type	Analysis
1	Region 1	WORK.REGION1	378	Model 1	PATH	Means and Covariances
2	Region 2	WORK.REGION2	423	Model 1	PATH	Means and Covariances

Model 1. Variables in the Model

Endogenous	Manifest	Availability Courtesy Delivery Helpful Pricing Quality Responsive Spend02 Spend03
	Latent	
Exogenous	Manifest	
	Latent	Product Service

Number of Endogenous Variables = 9
Number of Exogenous Variables = 2

The optimization converges. The fit summary table is presented in Output 33.28.2.

Output 33.28.2 Fit Summary

Fit Summary		
Modeling Info	**Number of Observations**	801
	Number of Variables	9
	Number of Moments	108
	Number of Parameters	31
	Number of Active Constraints	0
	Baseline Model Function Value	0.5003
	Baseline Model Chi-Square	399.7468
	Baseline Model Chi-Square DF	72
	Pr > Baseline Model Chi-Square	<.0001
Absolute Index	**Fit Function**	3.5297
	Chi-Square	2820.2504
	Chi-Square DF	77
	Pr > Chi-Square	<.0001
	Z-Test of Wilson & Hilferty	43.2575
	Hoelter Critical N	29
	Root Mean Square Residual (RMR)	28.2208
	Standardized RMR (SRMR)	2.1367
	Goodness of Fit Index (GFI)	0.9996
Parsimony Index	**Adjusted GFI (AGFI)**	0.9995
	Parsimonious GFI	1.0690
	RMSEA Estimate	0.2986
	RMSEA Lower 90% Confidence Limit	0.2892
	RMSEA Upper 90% Confidence Limit	0.3081
	Probability of Close Fit	<.0001
	Akaike Information Criterion	2882.2504
	Bozdogan CAIC	3058.5121
	Schwarz Bayesian Criterion	3027.5121
	McDonald Centrality	0.1804
Incremental Index	**Bentler Comparative Fit Index**	0.0000
	Bentler-Bonett NFI	-6.0551
	Bentler-Bonett Non-normed Index	-6.8265
	Bollen Normed Index Rho1	-5.5970
	Bollen Non-normed Index Delta2	-7.4997
	James et al. Parsimonious NFI	-6.4756

The model chi-square statistic is 2820.25. With $df = 77$ and $p < .0001$, the null hypothesis for the mean and covariance structures is rejected. All incremental fit indices are negative. These negative indices indicate a bad model fit, as compared with the independence model. The same fact can be deduced by comparing the chi-square values of the baseline model and the fitted model. The baseline model has five degrees of freedom less (five parameters more) than the structural model but the chi-square value is only 399.747, much less than the model fit chi-square value of 2820.25. Because variables in social and behavioral sciences are almost always expected to correlate with each other, a structural model that explains relationships even worse than the baseline model is deemed inappropriate for the data. The RMSEA for the structural model is 0.2986, which also indicates a bad model fit. However, the GFI, AGFI, and parsimonious GFI indicate good model fit, which is a little surprising given the fact that all other indices indicate the opposite and the overall model is pretty restrictive in the first place.

There are some warnings in the output:

```
WARNING: Model 1. The estimated error variance for variable Spend02 is
         negative.
WARNING: Model 1. Although all predicted variances for the observed and
         latent variables are positive, the corresponding predicted
         covariance matrix is not positive definite. It has one negative
         eigenvalue.
```

PROC CALIS routinely checks the properties of the estimated variances and the predicted covariance matrix. It issues warnings when there are problems. In this case, the error variance estimate of Spend02 is negative, and the predicted covariance matrix for the observed and latent variables is not positive definite and has one negative eigenvalue. You can inspect Output 33.28.3, which shows the variance parameter estimates of the variables.

Output 33.28.3 Variance Estimates

Model 1. Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	Courtesy	_Parm13	2.59181	0.13600	19.0574	<.0001
	Responsive	_Parm14	2.92423	0.15325	19.0821	<.0001
	Helpful	_Parm15	2.44625	0.12320	19.8566	<.0001
	Delivery	_Parm16	3.53408	0.18169	19.4510	<.0001
	Pricing	_Parm17	2.52948	0.12784	19.7869	<.0001
	Availability	_Parm18	1.57410	0.16884	9.3230	<.0001
	Quality	_Parm19	2.41658	0.13230	18.2661	<.0001
	Spend02	_Parm20	-14.40124	16.92863	-0.8507	0.3949
	Spend03	_Parm21	22.79309	5.75120	3.9632	<.0001
Exogenous	Service		1.00000			
	Product		1.00000			

The error variance estimate for Spend02 is –14.40, which is negative and might have led to the negative eigenvalue problem in the predicted covariance matrix for the observed and latent variables.

A Model with Unconstrained Parameters for the Two Regions

With all the bad model fit indications and the problematic predicted covariance matrix for the latent variables, you might conclude that an overly restricted model has been fit. Region 1 and Region 2 might not share exactly the same set of parameters. How about fitting a model at the other extreme with all parameters unconstrained for the two groups (regions)? Such a model can be easily specified, as shown in the following statements:

```
proc calis meanstr;
   group 1 / data=region1 label="Region 1" nobs=378;
   group 2 / data=region2 label="Region 2" nobs=423;
   model 1 / group=1;
      path
         Service ===> Spend02  Spend03      ,
         Product ===> Spend02  Spend03      ,
```

```
         Spend02 ===> Spend03               ,
         Service ===> Courtesy Responsive Helpful Delivery  ,
         Product ===> Pricing  Availability Quality ;
      pvar
         Courtesy Responsive Helpful Delivery Pricing
         Availability Quality Spend02 Spend03,
         Service Product = 2 * 1.;
      pcov
         Service Product;
   model 2 / group=2;
      refmodel 1/ allnewparms;
run;
```

Unlike the previous specification, in the current specification Group 2 is now fitted by a new model labeled as Model 2. This model is based on Model 1, as specified in REFMODEL statement. The ALLNEWPARMS option in the REFMODEL statement request that all parameters specified in Model 1 be renamed so that they become new parameters in Model 2. As a result, this specification gives different sets of estimates for Model 1 and Model 2, although both models have the same path structures and a comparable set of parameters.

The optimization converges without problems. The fit summary table is displayed in Output 33.28.4. The chi-square statistic is 29.613 ($df = 46$, $p = .97$). The theoretical model is not rejected. Many other measures of fit also indicate very good model fit. For example, the GFI, AGFI, Bentler CFI, Bentler-Bonett NFI, and Bollen nonnormed index delta2 are all close to one, and the RMSEA is close to zero.

Output 33.28.4 Fit Summary

Fit Summary		
Modeling Info	**Number of Observations**	801
	Number of Variables	9
	Number of Moments	108
	Number of Parameters	62
	Number of Active Constraints	0
	Baseline Model Function Value	0.5003
	Baseline Model Chi-Square	399.7468
	Baseline Model Chi-Square DF	72
	Pr > Baseline Model Chi-Square	<.0001
Absolute Index	**Fit Function**	0.0371
	Chi-Square	29.6131
	Chi-Square DF	46
	Pr > Chi-Square	0.9710
	Z-Test of Wilson & Hilferty	-1.8950
	Hoelter Critical N	1697
	Root Mean Square Residual (RMR)	0.0670
	Standardized RMR (SRMR)	0.0220
	Goodness of Fit Index (GFI)	1.0000
Parsimony Index	**Adjusted GFI (AGFI)**	1.0000
	Parsimonious GFI	0.6389
	RMSEA Estimate	0.0000
	RMSEA Lower 90% Confidence Limit	0.0000
	RMSEA Upper 90% Confidence Limit	0.0000
	Probability of Close Fit	1.0000
	Akaike Information Criterion	153.6131
	Bozdogan CAIC	506.1365
	Schwarz Bayesian Criterion	444.1365
	McDonald Centrality	1.0103
Incremental Index	**Bentler Comparative Fit Index**	1.0000
	Bentler-Bonett NFI	0.9259
	Bentler-Bonett Non-normed Index	1.0783
	Bollen Normed Index Rho1	0.8840
	Bollen Non-normed Index Delta2	1.0463
	James et al. Parsimonious NFI	0.5916

Notice that because there are no constraints between the two models for the groups, you might have fit the two sets of data by the respective models separately and gotten exactly the same results as in the current analysis. For example, you get two model fit chi-square values from separate analyses. Adding up these two chi-squares gives you the same overall chi-square as in Output 33.28.4.

PROC CALIS also provides a table for comparing the relative model fit of the groups. In Output 33.28.5, basic modeling information and some measures of fit for the two groups are shown along with the corresponding overall measures.

Output 33.28.5 Fit Comparison among Groups

Fit Comparison Among Groups				
		Overall	**Region 1**	**Region 2**
Modeling Info	**Number of Observations**	801	378	423
	Number of Variables	9	9	9
	Number of Moments	108	54	54
	Number of Parameters	62	31	31
	Number of Active Constraints	0	0	0
	Baseline Model Function Value	0.5003	0.4601	0.5363
	Baseline Model Chi-Square	399.7468	173.4482	226.2986
	Baseline Model Chi-Square DF	72	36	36
Fit Index	**Fit Function**	0.0371	0.0023	0.0681
	Percent Contribution to Chi-Square	100	3	97
	Root Mean Square Residual (RMR)	0.0670	0.0172	0.0907
	Standardized RMR (SRMR)	0.0220	0.0057	0.0298
	Goodness of Fit Index (GFI)	1.0000	1.0000	1.0000
	Bentler-Bonett NFI	0.9259	0.9950	0.8730

When you examine the results of this table, the first thing you have to realize is that in general the group statistics are not independent. For example, although the overall chi-square statistic can be written as the weighted sum of fit functions of the groups, in general it does not imply that the individual terms are statistically independent. In the current two-group analysis, the overall chi-square is written as

$$T = (N_1 - 1) f_1 + (N_2 - 1) f_2$$

where N_1 and N_2 are sample sizes for the groups and f_1 and f_2 are the discrepancy functions for the groups. Even though T is chi-square distributed under the null hypothesis, in general the individual terms $(N_1 - 1) f_1$ and $(N_2 - 1) f_2$ are not chi-square distributed under the same null hypothesis. So when you compare the group fits by using the statistics in Output 33.28.5, you should treat those as descriptive measures only.

The current model is a special case where f_1 and f_2 are actually independent of each other. The reason is that there are no constrained parameters for the models fitted to the two groups. This would imply that the individual terms $(N_1 - 1) f_1$ and $(N_2 - 1) f_2$ are chi-square distributed under the null hypothesis. Nonetheless, this fact is not important to the group comparison of the descriptive statistics in Output 33.28.5. The values of f_1 and f_2 are shown in the row labeled "Fit Function." Group 1 (Region 1) is fitted better by its model ($f_1 = 0.0023$) than is Group 2 (Region 2) by its model ($f_2 = 0.0681$). Next, the percentage contributions to the overall chi-square statistic for the two groups are shown. Group 1 contributes only 3% ($= (N_1 - 1) f_1 / T \times 100\%$) while Group 2 contributes 97%. Other measures like RMR, SRMR, and Bentler-Bonett NFI show that Group 1 data are fitted better. The GFI's show equal fits for the two groups, however.

Despite a very good fit, the current model is not intended to be the final model. It was fitted mainly for illustration purposes. The next section considers a partially constrained model for the two groups of data.

A Model with Partially Constrained Parameters for the Two Regions

For multiple-group analysis, cross-group constraints are of primary interest and should be explored whenever appropriate. The first fitting with all model parameters constrained for the groups has been shown to be too restrictive, while the current model with no cross-group constraints fits very well—so well that it might have overfit unnecessarily. A multiple-group model between these extremes is now explored. The following statements specify such a partially constrained model:

```
proc calis meanstr modification;
   group 1 / data=region1 label="Region 1" nobs=378;
   group 2 / data=region2 label="Region 2" nobs=423;
   model 3 / label="Model for References Only";
      path
         Service ===> Spend02  Spend03      ,
         Product ===> Spend02  Spend03      ,
         Spend02 ===> Spend03               ,
         Service ===> Courtesy Responsive
                      Helpful Delivery      ,
         Product ===> Pricing  Availability
                      Quality               ;
      pvar
         Courtesy Responsive Helpful Delivery Pricing
         Availability Quality Spend02 Spend03,
         Service Product = 2 * 1.;
      pcov
         Service Product;
   model 1 / groups=1;
      refmodel 3;
      mean
         Spend02 Spend03 = G1_InterSpend02 G1_InterSpend03,
         Courtesy Responsive Helpful
         Delivery Pricing Availability
         Quality = G1_intercept01-G1_intercept07;
   model 2 / groups=2;
      refmodel 3;
      mean
         Spend02 Spend03 = G2_InterSpend02 G2_InterSpend03,
         Courtesy Responsive Helpful
         Delivery Pricing Availability
         Quality = G2_intercept01-G2_intercept07;
      simtests
         SpendDiff       = (Spend02Diff Spend03Diff)
         MeasurementDiff = (CourtesyDiff ResponsiveDiff
                            HelpfulDiff DeliveryDiff
                            PricingDiff AvailabilityDiff
                            QualityDiff);
      Spend02Diff      = G2_InterSpend02 - G1_InterSpend02;
      Spend03Diff      = G2_InterSpend03 - G1_InterSpend03;
      CourtesyDiff     = G2_intercept01  - G1_intercept01;
      ResponsiveDiff   = G2_intercept02  - G1_intercept02;
      HelpfulDiff      = G2_intercept03  - G1_intercept03;
      DeliveryDiff     = G2_intercept04  - G1_intercept04;
      PricingDiff      = G2_intercept05  - G1_intercept05;
      AvailabilityDiff = G2_intercept06  - G1_intercept06;
```

```
        QualityDiff      = G2_intercept07  - G1_intercept07;
run;
```

In this specification, you use a special model definition. Model 3 serves as a reference model. You are not going to fit this model directly to any data set, but the specifications of other two models makes reference to it. Model 3 is no different from the basic path model specification used in preceding examples. The PATH model specification reflects the path diagram in Figure 33.27.

Region 1 is fitted by Model 1, which makes reference to Model 3 by using the REFMODEL statement. In addition, you add the MEAN statement specification. You now specify the intercept parameters explicitly by using the parameter names G1_intercept01–G1_intercept07, G1_InterSpend02, and G1_InterSpend03. In previous examples, these intercept parameters are set by default by PROC CALIS. This explicit parameter naming serves the purpose of distinguishing these parameters from those for Model 2.

Region 2 is fitted by Model 2, which also refers to Model 3 by using the REFMODEL statement. You also specify a MEAN statement for this model with explicit specifications of the intercept parameters. You name these intercepts G2_intercept01–G2_intercept07, G2_InterSpend02, and G2_InterSpend03. The G2 prefix distinguishes these parameters from the corresponding intercept parameters in the parent model. All in all, this means that both Models 1 and 2 refers to Model 3, except that Model 2 uses a different set of intercept parameters. In other words, in this multiple-group model the covariance structures for the two regions are constrained to be the same, while the means structures are allowed to be unconstrained.

You request additional statistics or tests in the current PROC CALIS analysis. The MODIFICATION option in the PROC CALIS statement requests that the Lagrange multiplier tests and Wald tests be conducted. The Lagrange multiplier tests provide information about which constrained or fixed parameters could be freed or added so as to improve the overall model fit. The Wald tests provide information about which existing parameters could be fixed at zeros (eliminated) without significantly affecting the overall model fit. These tests are discussed in more detail when the results are presented.

In the SIMTESTS statement, two simultaneous tests are requested. The first simultaneous test is named SpendDiff, which includes two parametric functions Spend02Diff and Spend03Diff. The second simultaneous test is named MeasurementDiff, which includes seven parametric functions: CourtesyDiff, ResponsiveDiff, HelpfulDiff, DeliveryDiff, PricingDiff, AvailabilityDiff, and QualityDiff. The null hypothesis of these simultaneous tests is of the form

$$H_0 : t_i = 0 \quad (i = 1 \ldots k)$$

where k is the number of parametric functions within the simultaneous test. In the current analysis, the component parametric functions are defined in the SAS programming statements, which are shown in the last block of the specification. Essentially, all these parametric functions represent the differences of the mean or intercept parameters between the two models for groups. The first simultaneous test is intended to test whether the mean or intercept parameters in the structural models are the same, while the second simultaneous test is intended to test whether the mean parameters in the measurement models are the same.

The fit summary table is shown in Output 33.28.6.

Output 33.28.6 Fit Summary

Fit Summary		
Modeling Info	**Number of Observations**	801
	Number of Variables	9
	Number of Moments	108
	Number of Parameters	40
	Number of Active Constraints	0
	Baseline Model Function Value	0.5003
	Baseline Model Chi-Square	399.7468
	Baseline Model Chi-Square DF	72
	Pr > Baseline Model Chi-Square	<.0001
Absolute Index	**Fit Function**	0.1346
	Chi-Square	107.5461
	Chi-Square DF	68
	Pr > Chi-Square	0.0016
	Z-Test of Wilson & Hilferty	2.9452
	Hoelter Critical N	657
	Root Mean Square Residual (RMR)	0.1577
	Standardized RMR (SRMR)	0.0678
	Goodness of Fit Index (GFI)	1.0000
Parsimony Index	**Adjusted GFI (AGFI)**	0.9999
	Parsimonious GFI	0.9444
	RMSEA Estimate	0.0382
	RMSEA Lower 90% Confidence Limit	0.0237
	RMSEA Upper 90% Confidence Limit	0.0514
	Probability of Close Fit	0.9275
	Akaike Information Criterion	187.5461
	Bozdogan CAIC	414.9806
	Schwarz Bayesian Criterion	374.9806
	McDonald Centrality	0.9756
Incremental Index	**Bentler Comparative Fit Index**	0.8793
	Bentler-Bonett NFI	0.7310
	Bentler-Bonett Non-normed Index	0.8722
	Bollen Normed Index Rho1	0.7151
	Bollen Non-normed Index Delta2	0.8808
	James et al. Parsimonious NFI	0.6904

The chi-square value is 107.55 ($df = 68$, $p = 0.0016$), which is statistically significant. The null hypothesis of the mean and covariance structures is rejected if an α-level at 0.01 or larger is chosen. However, in practical structural equation modeling, the chi-square test is not the only criterion, or even an important criterion, for evaluating model fit. The RMSEA estimate for the current model is 0.0382, which indicates a good fit. The probability level of close fit is 0.9275, indicating that a good population fit hypothesis (that is, population RMSEA < 0.05) cannot be rejected. The GFI, AGFI, and parsimonious GFI all indicate good fit. However, the incremental indices show only a respectable model fit.

Comparison of the model fit to the groups is shown in Output 33.28.7.

Output 33.28.7 Fit Comparison among Groups

Fit Comparison Among Groups				
		Overall	Region 1	Region 2
Modeling Info	Number of Observations	801	378	423
	Number of Variables	9	9	9
	Number of Moments	108	54	54
	Number of Parameters	40	31	31
	Number of Active Constraints	0	0	0
	Baseline Model Function Value	0.5003	0.4601	0.5363
	Baseline Model Chi-Square	399.7468	173.4482	226.2986
	Baseline Model Chi-Square DF	72	36	36
Fit Index	Fit Function	0.1346	0.1261	0.1422
	Percent Contribution to Chi-Square	100	44	56
	Root Mean Square Residual (RMR)	0.1577	0.1552	0.1599
	Standardized RMR (SRMR)	0.0678	0.0792	0.0557
	Goodness of Fit Index (GFI)	1.0000	1.0000	1.0000
	Bentler-Bonett NFI	0.7310	0.7260	0.7348

Looking at the percentage contribution to the chi-square, the Region 2 fitting shows a worse fit. However, this might be due to the larger sample size in Region 2. When comparing the fit of the two regions by using RMR, which does not take the sample size into account, the fitting of two groups are about the same. The standardized RMR even shows that Region 2 is fitted better. So, it seems to be safe to conclude that the models fit almost equally well (or badly) for the two regions.

The constrained parameter estimates for the two regions are shown in Output 33.28.8.

Output 33.28.8 Estimates of Path Coefficients and Other Covariance Parameters

Model 1. PATH List							
Path			Parameter	Estimate	Standard Error	t Value	Pr > |t|
Service	===>	Spend02	_Parm01	0.37475	0.21318	1.7579	0.0788
Service	===>	Spend03	_Parm02	0.53851	0.20840	2.5840	0.0098
Product	===>	Spend02	_Parm03	0.80372	0.21939	3.6635	0.0002
Product	===>	Spend03	_Parm04	0.59879	0.22144	2.7041	0.0068
Spend02	===>	Spend03	_Parm05	0.08952	0.03694	2.4233	0.0154
Service	===>	Courtesy	_Parm06	0.72418	0.07989	9.0648	<.0001
Service	===>	Responsive	_Parm07	0.90452	0.08886	10.1797	<.0001
Service	===>	Helpful	_Parm08	0.64969	0.07683	8.4557	<.0001
Service	===>	Delivery	_Parm09	0.64473	0.09021	7.1468	<.0001
Product	===>	Pricing	_Parm10	0.63452	0.07916	8.0160	<.0001
Product	===>	Availability	_Parm11	0.76737	0.08265	9.2852	<.0001
Product	===>	Quality	_Parm12	0.79716	0.08922	8.9347	<.0001

Output 33.28.8 *continued*

Model 1. Variance Parameters						
Variance Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Error	Courtesy	_Parm13	1.98374	0.13169	15.0638	<.0001
	Responsive	_Parm14	2.02152	0.16159	12.5101	<.0001
	Helpful	_Parm15	1.96535	0.12263	16.0273	<.0001
	Delivery	_Parm16	2.97542	0.17049	17.4518	<.0001
	Pricing	_Parm17	1.93952	0.12326	15.7358	<.0001
	Availability	_Parm18	1.63156	0.13067	12.4865	<.0001
	Quality	_Parm19	2.08849	0.15329	13.6246	<.0001
	Spend02	_Parm20	13.47066	0.71842	18.7505	<.0001
	Spend03	_Parm21	13.02883	0.68682	18.9697	<.0001
Exogenous	Service		1.00000			
	Product		1.00000			

Model 1. Covariances Among Exogenous Variables						
Var1	Var2	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Service	Product	_Parm22	0.33725	0.07061	4.7760	<.0001

All parameter estimates but one are statistically significant at $\alpha = 0.05$. The parameter _Parm01, which represents the path coefficient from Service to Spend02, has a t value of 1.76. This is only marginally significant. Although all these results bear the title of Model 1, these estimates are the same for Model 2, of which the corresponding results are not shown here.

The mean and intercept parameters for the two models (regions) are shown in Output 33.28.9.

Output 33.28.9 Estimates of Means and Intercepts

Model 1. Means and Intercepts						
Type	Variable	Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	Spend02	G1_InterSpend02	183.50000	0.19585	937.0	<.0001
	Spend03	G1_InterSpend03	285.49480	6.78127	42.1005	<.0001
	Courtesy	G1_intercept01	4.31200	0.08157	52.8652	<.0001
	Responsive	G1_intercept02	4.72400	0.08679	54.4310	<.0001
	Helpful	G1_intercept03	3.92100	0.07958	49.2720	<.0001
	Delivery	G1_intercept04	4.35700	0.09484	45.9397	<.0001
	Pricing	G1_intercept05	6.14400	0.07882	77.9499	<.0001
	Availability	G1_intercept06	4.99400	0.07674	65.0732	<.0001
	Quality	G1_intercept07	5.97100	0.08500	70.2454	<.0001

Output 33.28.9 *continued*

Model 2. Means and Intercepts						
Type	**Variable**	**Parameter**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	**Spend02**	G2_InterSpend02	156.25000	0.18511	844.1	<.0001
	Spend03	G2_InterSpend03	299.68311	5.77478	51.8952	<.0001
	Courtesy	G2_intercept01	2.41200	0.07709	31.2863	<.0001
	Responsive	G2_intercept02	2.72700	0.08203	33.2435	<.0001
	Helpful	G2_intercept03	5.22400	0.07522	69.4532	<.0001
	Delivery	G2_intercept04	6.37600	0.08964	71.1270	<.0001
	Pricing	G2_intercept05	7.14700	0.07450	95.9343	<.0001
	Availability	G2_intercept06	3.23300	0.07254	44.5702	<.0001
	Quality	G2_intercept07	5.11900	0.08034	63.7150	<.0001

All the mean and intercept estimates are statistically significant at $\alpha = 0.01$. Except for the fixed zero means for Service and Product, a quick glimpse of these mean and intercepts estimates shows a quite different pattern for the two models. Do these estimates truly differ beyond chance? The simultaneous tests of these parameter estimates shown in Output 33.28.10 can confirm this.

Output 33.28.10 shows two simultaneous tests, as requested in the original statements.

Output 33.28.10 Simultaneous Tests

Simultaneous Tests					
Simultaneous Test	**Parametric Function**	**Function Value**	**DF**	**Chi-Square**	**p Value**
SpendDiff			2	10458	<.0001
	Spend02Diff	-27.25000	1	10225	<.0001
	Spend03Diff	14.18831	1	185.86725	<.0001
MeasurementDiff			7	1610	<.0001
	CourtesyDiff	-1.90000	1	286.58605	<.0001
	ResponsiveDiff	-1.99700	1	279.63659	<.0001
	HelpfulDiff	1.30300	1	141.59942	<.0001
	DeliveryDiff	2.01900	1	239.35318	<.0001
	PricingDiff	1.00300	1	85.52567	<.0001
	AvailabilityDiff	-1.76100	1	278.09360	<.0001
	QualityDiff	-0.85200	1	53.06240	<.0001

The first one is SpendDiff, which tests simultaneously the following hypotheses:

H_0 : G2_InterSpend02 = G1_InterSpend02

H_0 : G2_InterSpend03 = G1_InterSpend03

The exceedingly large chi-square value 10,460 suggests the composite null hypothesis is false. Individual tests for these hypotheses suggest that each of these hypotheses should be rejected. The chi-square values for individual tests are 10,227 and 185.84, respectively.

Similarly, the simultaneous and individual tests of the intercepts in the measurement model suggest that the two models (groups) differ significantly in the means of the measured variables. Region 2 has significantly

higher means in variables Helpful, Delivery, and Pricing, but significantly lower means in variables Courtesy, Responsive, Availability, and Quality.

Now you are ready to answer the main research questions. The overall customer service (Service) does affect the future purchase (Spend03), but not the current purchase (Spend02), because the corresponding path coefficient (_Parm01) is only marginally significant. Perhaps this is an artifact because the rating was done after the purchases in 2002. That is, purchases in 2002 had been done before the impression about customer service was fully formed. However, this argument cannot explain why overall customer service (Service) also shows a strong and significant relationship with purchases in 2002 (Spend02). Nonetheless, customer service and product quality do affect the future purchases (Spend03) in an expected way, even after partialling out the effect of the previous purchase amount (Spend02). Apart from the mean differences of the variables, the common measurement and prediction (or structural) models fit the two regions very well.

Because the current model fits well and most parts of fitting meet your expectations, you might accept this model without looking for further improvement. Nonetheless, for illustration purposes, it would be useful to consider the LM test results. In Output 33.28.11, ranked LM statistics for the path coefficients in Model 1 and Model 2 are shown.

Output 33.28.11 LM Tests for Path Coefficients

Model 1.
Rank Order of the 10 Largest LM Stat for Path Relations

To	From	LM Stat	Pr > ChiSq	Parm Change
Service	Courtesy	11.15249	0.0008	-0.17145
Service	Helpful	3.09038	0.0788	0.09431
Service	Delivery	2.59511	0.1072	0.07504
Courtesy	Responsive	1.75943	0.1847	-0.07730
Delivery	Courtesy	1.66721	0.1966	0.08669
Helpful	Courtesy	1.62005	0.2031	0.07277
Courtesy	Product	1.48928	0.2223	-0.14815
Service	Product	0.83498	0.3608	-0.12327
Responsive	Helpful	0.76664	0.3813	-0.05625
Product	Helpful	0.53020	0.4665	-0.03831

Model 2.
Rank Order of the 10 Largest LM Stat for Path Relations

To	From	LM Stat	Pr > ChiSq	Parm Change
Delivery	Courtesy	16.91167	<.0001	-0.26641
Service	Courtesy	9.11235	0.0025	0.15430
Courtesy	Delivery	8.12091	0.0044	-0.12989
Courtesy	Responsive	8.03954	0.0046	0.16215
Pricing	Responsive	5.48406	0.0192	0.10424
Courtesy	Product	4.39347	0.0361	0.24412
Courtesy	Quality	3.52147	0.0606	0.08672
Service	Delivery	3.20160	0.0736	-0.08281
Service	Helpful	2.97015	0.0848	-0.09198
Responsive	Pricing	2.91498	0.0878	0.08943

Path coefficients that lead to better improvement (larger chi-square decrease) are shown first in the tables. For example, the first path coefficient that is suggested to be freed in Model 1 is the Service <=== Courtesy path. The associated p-value is 0.0008 and the estimated change of parameter value is –0.171. The second path coefficient is for the Service <=== Helpful path, but it is not significant at the 0.05 level. So, is it good to add the Service <=== Courtesy path to Model 1, based on the LM test results? The answer is that it depends on your application and the theoretical and practical implications. For example, the Service ===> Courtesy path, which is a part of the measurement model, is already specified in Model 1. Even though the LM test statistic shows a significant decrease of model fit chi-square, adding the Service <=== Courtesy path might destroy the measurement model and lead to problematic interpretations. In this case, it is wise not to add the Service <=== Courtesy path, which is suggested by the LM test results.

LM tests for the path coefficients in Model 2 are shown at the bottom of Output 33.28.11. Quite a few of these tests suggest significant improvements in model fit. Again, you are cautioned against adding these paths blindly.

LM tests for the error variances and covariances are shown in Output 33.28.12.

Output 33.28.12 LM Tests for Error Covariances

Model 1.
Rank Order of the 10 Largest LM Stat for Error Variances and Covariances

Error of	Error of	LM Stat	Pr > ChiSq	Parm Change
Responsive	Helpful	1.26589	0.2605	-0.15774
Delivery	Courtesy	0.70230	0.4020	0.12577
Helpful	Courtesy	0.50167	0.4788	0.09103
Quality	Availability	0.47993	0.4885	-0.09739
Quality	Pricing	0.45925	0.4980	0.09449
Responsive	Availability	0.25734	0.6120	0.05965
Helpful	Availability	0.24811	0.6184	-0.05413
Responsive	Pricing	0.23748	0.6260	-0.05911
Spend02	Availability	0.19634	0.6577	-0.13200
Responsive	Courtesy	0.18212	0.6696	0.06201

Model 2.
Rank Order of the 10 Largest LM Stat for Error Variances and Covariances

Error of	Error of	LM Stat	Pr > ChiSq	Parm Change
Delivery	Courtesy	16.00996	<.0001	-0.57408
Responsive	Pricing	4.89190	0.0270	0.25403
Helpful	Delivery	3.33480	0.0678	0.25299
Delivery	Availability	2.79513	0.0946	0.20656
Responsive	Availability	2.16944	0.1408	-0.16421
Quality	Courtesy	2.14952	0.1426	0.17094
Responsive	Courtesy	2.12832	0.1446	0.20604
Quality	Pricing	2.00978	0.1563	-0.19154
Quality	Availability	1.99477	0.1578	0.19459
Responsive	Quality	1.88736	0.1695	-0.16963

Using $\alpha = 0.05$, you might consider adding two pairs of correlated errors in Model 2. The first pair is for

Delivery and Courtesy, which has a p-value less than 0.0001. The second pair is Pricing and Responsive, which has a p-value of 0.027. Again, adding correlated errors (in the PCOV statement) should not be a pure statistical consideration. You should also consider theoretical and practical implications.

LM tests for other subsets of parameters are also conducted. Some subsets do not have parameters that can be freed, and so they are not shown here. Other subsets are not shown here simply for conserving space.

PROC CALIS ranks and outputs the LM test results for some default subsets of parameters. You have seen the subsets for path coefficients and correlated errors in the two previous outputs. Some other LM test results are not shown. With this kind of default LM output, there could be a huge amount of modification indices to look at. Fortunately, you can limit the LM test results to any subsets of potential parameters that you might be interested in. With your substantive knowledge, you can define such meaningful subsets of potential parameters by using the LMTESTS statement. The LM test indices and rankings are then done for each predefined subset of potential parameters. With these customized LM results, you can limit your attention to consider only those meaningful parameters to be added. See the LMTESTS statement on page 1628 for details.

The next group of LM tests is for releasing implicit equality constraints in your model, as shown in Output 33.28.13.

Output 33.28.13 LM Tests for Equality Constraints

Lagrange Multiplier Statistics for Releasing Equality Constraints								
Released Parameter							Changes	
Parm	Model	Type	Var1	Var2	LM Stat	Pr > ChiSq	Original Parm	Released Parm
_Parm01	1	DV_IV	Spend02	Service	0.01554	0.9008	-0.0213	0.0238
	2	DV_IV	Spend02	Service	0.01554	0.9008	0.0238	-0.0213
_Parm02	1	DV_IV	Spend03	Service	0.01763	0.8944	-0.0222	0.0248
	2	DV_IV	Spend03	Service	0.01763	0.8944	0.0248	-0.0222
_Parm03	1	DV_IV	Spend02	Product	0.0003403	0.9853	-0.00321	0.00355
	2	DV_IV	Spend02	Product	0.0003403	0.9853	0.00355	-0.00321
_Parm04	1	DV_IV	Spend03	Product	0.00176	0.9665	0.00714	-0.00802
	2	DV_IV	Spend03	Product	0.00176	0.9665	-0.00802	0.00714
_Parm05	1	DV_DV	Spend03	Spend02	0.0009698	0.9752	-0.00100	0.00112
	2	DV_DV	Spend03	Spend02	0.0009698	0.9752	0.00112	-0.00100
_Parm06	1	DV_IV	Courtesy	Service	19.17225	<.0001	0.2851	-0.3191
	2	DV_IV	Courtesy	Service	19.17225	<.0001	-0.3191	0.2851
_Parm07	1	DV_IV	Responsive	Service	0.21266	0.6447	-0.0304	0.0341
	2	DV_IV	Responsive	Service	0.21266	0.6447	0.0341	-0.0304
_Parm08	1	DV_IV	Helpful	Service	4.60629	0.0319	-0.1389	0.1555
	2	DV_IV	Helpful	Service	4.60629	0.0319	0.1555	-0.1389
_Parm09	1	DV_IV	Delivery	Service	3.59763	0.0579	-0.1508	0.1687
	2	DV_IV	Delivery	Service	3.59763	0.0579	0.1687	-0.1508
_Parm10	1	DV_IV	Pricing	Product	0.50974	0.4753	0.0468	-0.0524
	2	DV_IV	Pricing	Product	0.50974	0.4753	-0.0524	0.0468
_Parm11	1	DV_IV	Availability	Product	0.57701	0.4475	-0.0457	0.0512
	2	DV_IV	Availability	Product	0.57701	0.4475	0.0512	-0.0457
_Parm12	1	DV_IV	Quality	Product	0.00566	0.9400	-0.00511	0.00574
	2	DV_IV	Quality	Product	0.00566	0.9400	0.00574	-0.00511
_Parm13	1	COVERR	Courtesy	Courtesy	45.24725	<.0001	0.7204	-0.8064
	2	COVERR	Courtesy	Courtesy	45.24725	<.0001	-0.8064	0.7204
_Parm14	1	COVERR	Responsive	Responsive	1.73499	0.1878	-0.1555	0.1740
	2	COVERR	Responsive	Responsive	1.73499	0.1878	0.1740	-0.1555
_Parm15	1	COVERR	Helpful	Helpful	11.13266	0.0008	-0.3448	0.3860
	2	COVERR	Helpful	Helpful	11.13266	0.0008	0.3860	-0.3448
_Parm16	1	COVERR	Delivery	Delivery	4.99097	0.0255	-0.3364	0.3766
	2	COVERR	Delivery	Delivery	4.99097	0.0255	0.3766	-0.3364
_Parm17	1	COVERR	Pricing	Pricing	2.86428	0.0906	0.1729	-0.1936
	2	COVERR	Pricing	Pricing	2.86428	0.0906	-0.1936	0.1729
_Parm18	1	COVERR	Availability	Availability	2.53147	0.1116	-0.1494	0.1672
	2	COVERR	Availability	Availability	2.53147	0.1116	0.1672	-0.1494
_Parm19	1	COVERR	Quality	Quality	0.07328	0.7866	-0.0315	0.0352
	2	COVERR	Quality	Quality	0.07328	0.7866	0.0352	-0.0315
_Parm20	1	COVERR	Spend02	Spend02	0.00214	0.9631	0.0304	-0.0340
	2	COVERR	Spend02	Spend02	0.00214	0.9631	-0.0340	0.0304
_Parm21	1	COVERR	Spend03	Spend03	0.0001773	0.9894	-0.00842	0.00946
	2	COVERR	Spend03	Spend03	0.0001773	0.9894	0.00946	-0.00842
_Parm22	1	COVEXOG	Service	Product	0.87147	0.3505	0.0605	-0.0678
	2	COVEXOG	Service	Product	0.87147	0.3505	-0.0678	0.0605

Recall that the measurement and the prediction models for the two regions are constrained to be the same by model referencing (that is, the REFMODEL statement). Output 33.28.13 shows you which parameter can be unconstrained so that your overall model fit might improve. For example, if you unconstrain the first parameter _Parm01, which is for the path effect of Spend02 <=== Service, for the two models, the expected chi-square decrease (LM Stat) is about 0.0158, which is not significant (p = .9001). The associated parameter changes are small too. However, if you consider unconstraining parameter _Parm06, which is for the path effect of Courtesy <=== Service, the expected decrease of chi-square is 19.22 ($p < 0.0001$). There are two rows for this parameter. Each row represents a parameter location to be released from the equality constraint. Consider the first row first. If you rename the coefficient for the Courtesy <=== Service path in Model 1 to a new parameter, say "new" (while keeping _Parm06 as the parameter for the Courtesy <=== Service path in Model 2) and fit the model again, the new estimate of _Parm06 is 0.2852 greater than the previous _Parm06 estimate. The estimate of "new" is 0.3196 less than the previous _Parm06 estimate. The second row for the _Parm06 parameter shows similar but reflected results. It is for renaming the parameter location in Model 2. For this example each equality constraint has exactly two locations, one for Model 1 and one for Model 2. That is the reason why you always observe reflected results for freeing the locations successively. Reflected results are not the case if you have equality constraints with more than two parameter locations.

Another example of a large expected improvement of model fit is the result of freeing the constrained variances of Courtesy among the two models. The corresponding row to look at is the row with parameter _Parm13, where the parameter type is labeled "COVERR" and the values for Var1 and Var2 are both "Courtesy." The LM statistic is 45.255, which is a significant chi-square decrease if you free either parameter location. If you choose to rename the error variance for Courtesy in Model 1, the new _Parm13 estimate is 0.8052 smaller than the original _Parm13 estimate. The new estimate of the error variance for Courtesy in Model 2 is 0.7211 greater than the previous _Parm13 estimate. Finally, the constrained parameter _Parm15, which is the error variance parameter for Helpful in both models, is also a potential constraint that can be released with a significant model fit improvement.

In addition to the LM statistics for suggesting ways to improve model fit, PROC CALIS also computes the Wald tests to show which parameters can be constrained to zero without jeopardizing the model fit significantly. The Wald test results are shown in Output 33.28.14.

Output 33.28.14 Wald Tests

Stepwise Multivariate Wald Test					
	Cumulative Statistics			Univariate Increment	
Parm	Chi-Square	DF	Pr > ChiSq	Chi-Square	Pr > ChiSq
_Parm01	3.09039	1	0.0788	3.09039	0.0788

In Output 33.28.14, you see that _Parm01, which is for the path effect of Spend02 <=== Service, is suggested to be a fixed zero parameter (eliminated from the model) by the Wald test. Fixing this parameter to zero (or dropping the Spend02 <=== Service path from the model) is expected to increase the model fit chi-square by 3.085 (p = .079), which is only marginally significant at $\alpha = 0.05$.

As is the case for the LM test statistics, you should not automatically adhere to the suggestions by the Wald statistics. Substantive and theoretical considerations should always be considered when determining whether a parameter should be added or dropped.

Example 33.29: Fitting the RAM and EQS Models by the COSAN Modeling Language

(View the complete code for this example (calex401.sas) in the example repository.)

The COSAN modeling language in PROC CALIS enables you to specify the direct or implied mean and covariance structures for the data in terms of matrix formulas. It is a very general modeling language, and all other modeling languages in PROC CALIS are special cases of the COSAN modeling language. This example shows how you can apply the COSAN modeling language to situations where you might usually use the "easier" modeling languages. Therefore, the purpose of this example is not to recommend the use of the COSAN modeling specification to the specific application. Rather, through its connections with other more well-known model types, this example intends to help you understand the basics of the COSAN modeling language.

In Example 33.17, you fit a path model to the Wheaton data (Wheaton et al. 1977) by using the PATH modeling language. The mathematical basis of the PATH modeling language is the RAM model. In Example 33.23, you use the RAM and LINEQS statements to specify the same path model. In all these different types of specifications, you specify the functional relationships of the variables and the variance and covariance parameters in the model. PROC CALIS then generates the implied covariance structures for analysis internally. The COSAN modeling language is quite different. In the COSAN statement, you specify the covariance structures directly as a matrix formula. This example shows how you can do that in two different ways. One specification emulates the RAM model (McDonald 1978, 1980) covariance structures and the other emulates the EQS model (Bentler 1995) covariance structures.

Emulating the RAM Model by the COSAN Modeling Language

In the RAM model, you specify all information regarding the path effects or coefficients (that is, single-headed arrows in the path diagram) in the so-called **A** (_A_) matrix. You specify all the information regarding the variances and covariances (that is, the double-headed arrows in the path diagram) in the **P** (_P_) matrix. See the section "The RAM Model" on page 1769 for more details about the mathematical model for RAM. Once you define these two matrices, the implied covariance structures for the observed variables are derived by the formula

$$\mathbf{\Sigma} = \mathbf{J} * (\mathbf{I} - \mathbf{A})^{-1} * \mathbf{P} * (\mathbf{I} - \mathbf{A})^{-1\prime} * \mathbf{J}'$$

where **I** is an identity matrix and **J** is a selection matrix that contains 0 or 1 as its elements for selecting the covariance structures elements for the observed variables.

For example, in the RAM model specification in Example 33.23, you essentially use the following RAM model specification:

```
proc calis nobs=932 data=Wheaton primat nose;
   ram
      var =  Anomie67     /* 1 */
             Powerless67  /* 2 */
             Anomie71     /* 3 */
             Powerless71  /* 4 */
             Education    /* 5 */
             SEI          /* 6 */
             Alien67      /* 7 */
```

```
            Alien71      /* 8 */
            SES,         /* 9 */
      _A_   1   7   1.0,
      _A_   2   7   0.833,
      _A_   3   8   1.0,
      _A_   4   8   0.833,
      _A_   5   9   1.0,
      _A_   6   9   lambda,
      _A_   7   9   gamma1,
      _A_   8   9   gamma2,
      _A_   8   7   beta,
      _P_   1   1   theta1,
      _P_   2   2   theta2,
      _P_   3   3   theta1,
      _P_   4   4   theta2,
      _P_   5   5   theta3,
      _P_   6   6   theta4,
      _P_   7   7   psi1,
      _P_   8   8   psi2,
      _P_   9   9   phi,
      _P_   1   3   theta5,
      _P_   2   4   theta5;
run;
```

In the RAM statement, you specify all the parameters in the _A_ and _P_ matrices, and PROC CALIS generates the corresponding covariance structures for analysis. However, with the COSAN modeling language, in addition to the parameter in the model matrices, you need to supply the matrix formula for the covariance structures, as shown in the preceding formula for $\mathbf{\Sigma}$.

Before discussing how you can specify the COSAN model that corresponds to this RAM model specification, it is useful to look at the initial model matrices that are generated by the preceding RAM model specification. To do this, you use the PRIMAT option in the PROC CALIS statement.

Output 33.29.1 and Output 33.29.2 show the initial _A_ and _P_ matrices, respectively, for the RAM model.

Output 33.29.1 Initial _A_ Matrix of the RAM Model

Initial RAM _A_ Matrix									
	Anomie67	Powerless67	Anomie71	Powerless71	Education	SEI	Alien67	Alien71	SES
Anomie67	0	0	0	0	0	0	1.0000	0	0
Powerless67	0	0	0	0	0	0	0.8330	0	0
Anomie71	0	0	0	0	0	0	0	1.0000	0
Powerless71	0	0	0	0	0	0	0	0.8330	0
Education	0	0	0	0	0	0	0	0	1.0000
SEI	0	0	0	0	0	0	0	0	. [lambda]
Alien67	0	0	0	0	0	0	0	0	. [gamma1]
Alien71	0	0	0	0	0	0	. [beta]	0	. [gamma2]
SES	0	0	0	0	0	0	0	0	0

Output 33.29.2 Initial _P_ Matrix of the RAM Model

Initial RAM _P_ Matrix									
	Anomie67	Powerless67	Anomie71	Powerless71	Education	SEI	Alien67	Alien71	SES
Anomie67	. [theta1]	0	. [theta5]	0	0	0	0	0	0
Powerless67	0	. [theta2]	0	. [theta5]	0	0	0	0	0
Anomie71	. [theta5]	0	. [theta1]	0	0	0	0	0	0
Powerless71	0	. [theta5]	0	. [theta2]	0	0	0	0	0
Education	0	0	0	0	. [theta3]	0	0	0	0
SEI	0	0	0	0	0	. [theta4]	0	0	0
Alien67	0	0	0	0	0	0	. [psi1]	0	0
Alien71	0	0	0	0	0	0	0	. [psi2]	0
SES	0	0	0	0	0	0	0	0	. [phi]

Essentially, to specify the same model by the COSAN modeling language, you need to provide the same information in these two initial model matrices and the covariance structure formula for $\mathbf{\Sigma}$ in the COSAN model specification, which is shown in the following statements:

```
proc calis data=Wheaton nobs=932 nose;
   cosan
      var= Anomie67 Powerless67 Anomie71 Powerless71 Education SEI,
      J(9, IDE) * A(9, GEN, IMI) * P(9, SYM);
```

```
    matrix  A
       [1 2 8   , 7] = 1.0  0.833  beta,
       [3 4     , 8] = 1.0  0.833 ,
       [5 6 7 8 , 9] = 1.   lambda  gamma1  gamma2;
    matrix P
       [1,1] = theta1-theta2 theta1-theta4 ,
       [7,7] = psi1 psi2 phi,
       [3,1] = theta5 ,
       [4,2] = theta5 ;
    vnames
       J = [Anomie67 Powerless67 Anomie71 Powerless71
            Education SEI Alien67 Alien71 SES],
       A = J,
       P = A;
run;
```

In the PROC CALIS statement, you provide the data set in the DATA= option and the number of observations in the NOBS= option. You use the NOSE option to turn off the computation of the standard error estimates.

In the VAR= option of the COSAN statement, you provide the list of observed variables for the analysis. You do not specify the latent variables in the VAR= option in the COSAN statement as you do in the VAR= option in the RAM statement. Then, you specify the formula for the covariance structures for the set of variables in the VAR= list. Because the covariance structure formula is symmetric, you only need to specify "half" of it. That is, the specification **J(9,IDE)*A(9,GEN,IMI)*P(9,SYM)** in the COSAN statement automatically expands to

$$\mathbf{J} * (\mathbf{I} - \mathbf{A})^{-1} * \mathbf{P} * (\mathbf{I} - \mathbf{A})^{-1\prime} * \mathbf{J}'$$

which is the required covariance structures. The arguments in the matrices represent the number of columns, the matrix type, and the transformation type (optional), respectively. For example, the notation **A(9, GEN, IMI)** means that matrix **A** has nine columns and it is a general (GEN) rectangular or square matrix. You do not specify the number of rows for matrix **A** explicitly, but PROC CALIS can deduce that because matrix **A** follows matrix **J** in the multiplication. To make matrix multiplication conformable, the number of rows for matrix **A** must be the same as the number of columns for matrix **J**, which is nine. The IMI notation means the identity-minus-inverse transformation, which results in putting $(\mathbf{I} - \mathbf{A})^{-1}$ in the expression. Matrix **P** in the covariance structure formula is a 9×9 symmetric matrix. It does not have any transformation in the formula. Matrix **J** in the covariance structure formula is a so-called generalized identity matrix (IDE), which has six rows and nine columns. Basically, you use this matrix to select the observed variables in the covariance structure formula. The exact form of this matrix will become clear when the PROC CALIS output is shown.

Next, you use two MATRIX statements to specify the parameters in the model matrices **A** and **P**, for RAM model matrices _A_ and _P_, respectively. For example, in the first entry of the MATRIX statement for the **A** matrix, you specify the elements **[1,7]**, **[2,7]**, and **[8,7]** by 1.0, 0.833, and beta, respectively. The first two elements are fixed constants, while the last one is a free parameter named beta. Similarly, you specify all the fixed or free parameters in matrix **A**, which reflects the same pattern you specify for the _A_ matrix of the RAM model, as shown in Output 33.29.1.

For the **P** matrix, you specify the parameters in the same fashion. Because **P** is defined as a symmetric matrix, you need to specify only the lower triangular elements. In the first entry of the MATRIX statement for the **P** matrix, you specify the **[1,1]** element, but the trailing parameter list has six parameters. The **[1,1]** notation here is interpreted as the starting location of the matrix. It proceeds to **[2,2]**, **[3,3]**, **[4,4]** and so on. The length of the trailing parameter list determines the number of elements being specified. Therefore,

the last parameter in this entry is for $\mathbf{P}[6, 6]$, which is a free parameter theta4. Similarly, you define all other parameters in the **P** matrix, which reflects the same pattern you specify for the _P_ matrix of the RAM model, as shown in Output 33.29.2.

In the VNAMES statement, you can specify the column variable names for the model matrices. You provide a set of nine variable names for the column of matrix **J** in the pairs of brackets. The first six names are those of the observed variables in the COSAN model, while the last six names are for latent factors. How about the row variable names for matrix **J**? Because matrix **J** is the first matrix in the covariance structure formula, its row names are automatically the same as the names of the observed variables in the VAR= list of the COSAN statement. Next, you specify the column variable names of matrix **A**. You equate that to matrix **J**, meaning that the column variable names in matrix **A** are the same those for matrix **J**. How about the row variable names for matrix **A**? Because matrix **A** follows matrix **J** in the covariance structure formula, its row names are automatically same as the column names for matrix **J**. Lastly, you define that the column names for matrix **P** are the same as those for matrix **A**.

Notice that column names serve only as labels. PROC CALIS does not know the identities of the row and column variables. For example, the first column of matrix **A** is Anomie67, which is also a name for an observed variable in the COSAN model. Keeping other specifications intact, you could name this column by any other name without affecting the model estimation. It is recommended that you use sensible names that help you remember the identities of the row and column variables, such as this example shows.

Output 33.29.3 shows the modeling information and the observed variables in the COSAN model. PROC CALIS analyzed the covariance structures of the six observed variables listed in Output 33.29.3.

Output 33.29.3 Modeling Information of the COSAN Model for the Wheaton Data: RAM Emulation

Modeling Information	
Maximum Likelihood Estimation	
Data Set	WORK.WHEATON
N Obs	932
Model Type	COSAN
Analysis	Covariances

Observed Variables (N = 6) in the Model					
Anomie67	Powerless67	Anomie71	Powerless71	Education	SEI

Output 33.29.4 shows the covariance structures and some properties of the model matrices. The covariance structure formula for Sigma is defined as required. You can also check the matrix properties in this output to see if they are what you intend them to be.

Output 33.29.4 The Covariance Structures and Model Matrices of the COSAN Model for the Wheaton Data: RAM Emulation

COSAN Model Structures
Sigma = J*inv(_I_-A)*P*(inv(_I_-A))`*J`

Summary of Model Matrices			
Matrix	**N Row**	**N Col**	**Matrix Type**
A	9	9	GEN: Square
J	6	9	IDE: (I || 0)
P	9	9	SYM: Symmetric

Output 33.29.4 shows that $\mathbf{J}$ is a 6×9 "identity" matrix $(\mathbf{I}||\mathbf{0})$. Essentially, $\mathbf{J}$ is a selection matrix that contains either 0 or 1 as its elements. The role of matrix $\mathbf{J}$ in the covariance structure formula is to extract first six rows and columns in the inner covariance structures $(\mathbf{I}-\mathbf{A})^{-1} * \mathbf{P} * (\mathbf{I}-\mathbf{A})^{-1\prime}$ (which is 9×9) to form the covariance structures only for the observed variables (which is 6×6). But how can this identity matrix have more columns (9) than rows (6)? In common mathematical notation, an identity matrix must always be a square matrix. However, for convenience in notation, PROC CALIS generalizes it to the IDE type. An IDE matrix that has the same numbers of columns and rows is a square identity matrix. If an IDE matrix has more columns than rows, it denotes an identity matrix concatenated (to the right) by a null matrix (that is, the $(\mathbf{I}||\mathbf{0})$ notation). If an IDE matrix has more rows than columns, it denotes an identity matrix appended (to the bottom) by a null matrix (that is, the $(\mathbf{I}//\mathbf{0})$ notation). The generalized definition for the IDE matrix offers an efficient way to define selection matrix, such as the $\mathbf{J}$ matrix shown in this example.

Output 33.29.5 shows the model fit chi-square of the COSAN model. This is the same model fit as in Output 33.17.6 of Example 33.17, as expected.

Output 33.29.5 Model Fit of the COSAN Model for the Wheaton Data: RAM Emulation

Fit Summary	
Chi-Square	13.4851
Chi-Square DF	9
Pr > Chi-Square	0.1419

Output 33.29.6 shows the estimates in the $\mathbf{A}$ matrix.

Output 33.29.6 Estimate of the $\mathbf{A}$ Matrix by the COSAN Model Specification

Model Matrix A
(9 x 9 General Square Matrix)

	Anomie67	Powerless67	Anomie71	Powerless71	Education	SEI	Alien67	Alien71	SES
Anomie67	0	0	0	0	0	0	1.0000	0	0
Powerless67	0	0	0	0	0	0	0.8330	0	0
Anomie71	0	0	0	0	0	0	0	1.0000	0
Powerless71	0	0	0	0	0	0	0	0.8330	0
Education	0	0	0	0	0	0	0	0	1.0000
SEI	0	0	0	0	0	0	0	0	5.3689 [lambda]
Alien67	0	0	0	0	0	0	0	0	-0.6299 [gamma1]
Alien71	0	0	0	0	0	0	0.5931 [beta]	0	-0.2409 [gamma2]
SES	0	0	0	0	0	0	0	0	0

The estimates in Output 33.29.6 from the COSAN model specification are essentially the same as those from the RAM model specification, as shown in the matrix form in Output 33.29.7.

Output 33.29.7 Estimate of the $\mathbf{A}$ Matrix by the RAM Model Specification

RAM _A_ Matrix									
	Anomie67	Powerless67	Anomie71	Powerless71	Education	SEI	Alien67	Alien71	SES
Anomie67	0	0	0	0	0	0	1.0000	0	0
Powerless67	0	0	0	0	0	0	0.8330	0	0
Anomie71	0	0	0	0	0	0	0	1.0000	0
Powerless71	0	0	0	0	0	0	0	0.8330	0
Education	0	0	0	0	0	0	0	0	1.0000
SEI	0	0	0	0	0	0	0	0	5.3688 [lambda]
Alien67	0	0	0	0	0	0	0	0	-0.6299 [gamma1]
Alien71	0	0	0	0	0	0	0.5931 [beta]	0	-0.2409 [gamma2]
SES	0	0	0	0	0	0	0	0	0

Output 33.29.8 shows the estimates in the $\mathbf{P}$ matrix.

Output 33.29.8 Estimate of the $\mathbf{P}$ Matrix by the COSAN Model Specification

Model Matrix P (9 x 9 Symmetric Matrix)									
	Anomie67	Powerless67	Anomie71	Powerless71	Education	SEI	Alien67	Alien71	SES
Anomie67	3.6078 [theta1]	0	0.9058 [theta5]	0	0	0	0	0	0
Powerless67	0	3.5950 [theta2]	0	0.9058 [theta5]	0	0	0	0	0
Anomie71	0.9058 [theta5]	0	3.6078 [theta1]	0	0	0	0	0	0
Powerless71	0	0.9058 [theta5]	0	3.5950 [theta2]	0	0	0	0	0
Education	0	0	0	0	2.9938 [theta3]	0	0	0	0
SEI	0	0	0	0	0	259.5738 [theta4]	0	0	0
Alien67	0	0	0	0	0	0	5.6705 [psi1]	0	0
Alien71	0	0	0	0	0	0	0	4.5148 [psi2]	0
SES	0	0	0	0	0	0	0	0	6.6162 [phi]

Again, aside from very minor numerical differences, the estimates shown in Output 33.29.8 from the COSAN model specification are essentially the same as those from the RAM model specification, as shown in the matrix form in Output 33.29.9.

Output 33.29.9 Estimate of the $\mathbf{P}$ Matrix by the RAM Model Specification

RAM _P_ Matrix									
	Anomie67	Powerless67	Anomie71	Powerless71	Education	SEI	Alien67	Alien71	SES
Anomie67	3.6080 [theta1]	0	0.9058 [theta5]	0	0	0	0	0	0
Powerless67	0	3.5949 [theta2]	0	0.9058 [theta5]	0	0	0	0	0
Anomie71	0.9058 [theta5]	0	3.6080 [theta1]	0	0	0	0	0	0
Powerless71	0	0.9058 [theta5]	0	3.5949 [theta2]	0	0	0	0	0
Education	0	0	0	0	2.9937 [theta3]	0	0	0	0
SEI	0	0	0	0	0	259.5764 [theta4]	0	0	0
Alien67	0	0	0	0	0	0	5.6705 [psi1]	0	0
Alien71	0	0	0	0	0	0	0	4.5148 [psi2]	0
SES	0	0	0	0	0	0	0	0	6.6163 [phi]

Emulating the EQS Model by the COSAN Modeling Language

The LINEQS modeling language in PROC CALIS enables you to specify the functional relationships among variables by using the equation input, much the same way that you can do with the EQS software (Bentler 1995). The covariance structure formula for the observed variables in the EQS model is

$$\mathbf{\Sigma} = \mathbf{J} * (\mathbf{I} - \mathbf{Beta})^{-1} * \mathbf{Gamma} * \mathbf{Phi} * \mathbf{Gamma}' * (\mathbf{I} - \mathbf{Beta})^{-1\prime} * \mathbf{J}'$$

where $\mathbf{I}$ is an identity matrix, $\mathbf{J}$ is a selection matrix that contains 0 or 1 as its elements for selecting the covariance structures elements for the observed variables, $\mathbf{Beta}$ is a square matrix for specifying relationships among the endogenous variables, Gamma is a matrix for specifying relationships between the endogenous variables and the exogenous variables, and $\mathbf{Phi}$ is a matrix for specifying the variances and covariances of the exogenous variables. Notice that in the EQS model, error or disturbance variables are counted as exogenous variables in the model.

In Example 33.23, you use the following LINEQS specification for the Wheaton data:

```
proc calis nobs=932 data=Wheaton primat nose;
   lineqs
      Anomie67     = 1.0    * f_Alien67 + e1,
      Powerless67  = 0.833  * f_Alien67 + e2,
      Anomie71     = 1.0    * f_Alien71 + e3,
      Powerless71  = 0.833  * f_Alien71 + e4,
      Education    = 1.0    * f_SES     + e5,
      SEI          = lambda * f_SES     + e6,
      f_Alien67    = gamma1 * f_SES     + d1,
      f_Alien71    = gamma2 * f_SES     + beta * f_Alien67 + d2;
   variance
      E1           = theta1,
      E2           = theta2,
```

```
      E3            = theta1,
      E4            = theta2,
      E5            = theta3,
      E6            = theta4,
      D1            = psi1,
      D2            = psi2,
      f_SES         = phi;
   cov
      E1  E3        = theta5,
      E2  E4        = theta5;
run;
```

In the LINEQS statement, you specify all the functional relationships among variables. In the VARIANCE and COV statements, you specify all the variance and covariance parameters in the model. None of the parameters is specified as a matrix element in the LINEQS model. The default output by PROC CALIS does not print the EQS model matrices. To print these model matrices, you use the PRIMAT option in the PROC CALIS statement. Output 33.29.10, Output 33.29.11, and Output 33.29.12 show the initial specification of these model matrices:

Output 33.29.10 The Initial _EQSBETA_ Matrix by the LINEQS Model Specification

Initial _EQSBETA_ Matrix								
	Anomie67	Anomie71	Education	Powerless67	Powerless71	SEI	f_Alien67	f_Alien71
Anomie67	0	0	0	0	0	0	1.0000	0
Anomie71	0	0	0	0	0	0	0	1.0000
Education	0	0	0	0	0	0	0	0
Powerless67	0	0	0	0	0	0	0.8330	0
Powerless71	0	0	0	0	0	0	0	0.8330
SEI	0	0	0	0	0	0	0	0
f_Alien67	0	0	0	0	0	0	0	0
f_Alien71	0	0	0	0	0	0	. [beta]	0

Output 33.29.11 The Initial _EQSGAMMA_ Matrix by the LINEQS Model Specification

Initial _EQSGAMMA_ Matrix									
	f_SES	e1	e3	e5	e2	e4	e6	d1	d2
Anomie67	0	1.0000	0	0	0	0	0	0	0
Anomie71	0	0	1.0000	0	0	0	0	0	0
Education	1.0000	0	0	1.0000	0	0	0	0	0
Powerless67	0	0	0	0	1.0000	0	0	0	0
Powerless71	0	0	0	0	0	1.0000	0	0	0
SEI	. [lambda]	0	0	0	0	0	1.0000	0	0
f_Alien67	. [gamma1]	0	0	0	0	0	0	1.0000	0
f_Alien71	. [gamma2]	0	0	0	0	0	0	0	1.0000

Output 33.29.12 The Initial _EQSPHI_ Matrix by the LINEQS Model Specification

Initial _EQSPHI_ Matrix									
	f_SES	e1	e3	e5	e2	e4	e6	d1	d2
f_SES	. [phi]	0	0	0	0	0	0	0	0
e1	0	. [theta1]	. [theta5]	0	0	0	0	0	0
e3	0	. [theta5]	. [theta1]	0	0	0	0	0	0
e5	0	0	0	. [theta3]	0	0	0	0	0
e2	0	0	0	0	. [theta2]	. [theta5]	0	0	0
e4	0	0	0	0	. [theta5]	. [theta2]	0	0	0
e6	0	0	0	0	0	0	. [theta4]	0	0
d1	0	0	0	0	0	0	0	. [psi1]	0
d2	0	0	0	0	0	0	0	0	. [psi2]

In the COSAN modeling language, you need to provide the three initial model matrices and the covariance structure formula for Σ, which is shown in the following statements:

```
proc calis cov data=Wheaton nobs=932 nose;
   cosan
      var = Anomie67 Anomie71 Education Powerless67 Powerless71 SEI,
      J(8, IDE) * Beta(8, GEN, IMI) * Gamma(9, GEN) * Phi(9, SYM);
   matrix Beta
           [1 4 8    , 7] = 1.0  0.833  beta,
           [2 5      , 8] = 1.0  0.833 ;
```

```
    matrix Gamma
            [3 6 7 8 , 1] = 1.0  lambda gamma1 gamma2,
            [1,2]         = 8 *  1.0;
    matrix Phi
            [1,1] = phi 2*theta1 theta3 2*theta2 theta4 psi1 psi2,
            [3,2] = theta5 ,
            [6,5] = theta5 ;
    vnames J     = [Anomie67 Anomie71 Education Powerless67 Powerless71 SEI
                    f_Alien67 f_Alien71],
           Beta  = J,
           Gamma = [f_SES e1 e3 e5 e2 e4 e6 d1 d2],
           Phi   = Gamma;
run;
```

In the PROC CALIS statement, you provide the data set in the DATA= option and the number of observations in the NOBS= option. You use the NOSE option to turn off the computation of the standard error estimates.

In the VAR= option of the COSAN statement, you provide the list of observed variables for the analysis. You arrange the observed variables in such a way that they are in the same order as in Output 33.29.10, Output 33.29.11, and Output 33.29.12. This is useful for comparing the results from the LINEQS and COSAN model specifications. After the specification of the observed variables, you specify the covariance structure model in the COSAN statement. Again, you only need to specify "half" of it. That is, the specification `J(8,IDE)*Beta(8,GEN,IMI)*Gamma(9,GEN)*Phi(9,SYM)` in the COSAN statement automatically expands to

$$\mathbf{\Sigma} = \mathbf{J} * (\mathbf{I} - \mathbf{Beta})^{-1} * \mathbf{Gamma} * \mathbf{Phi} * \mathbf{Gamma}' * (\mathbf{I} - \mathbf{Beta})^{-1\prime} * \mathbf{J}'$$

which is the required covariance structures. Matrix properties and transformation types are defined in the arguments for the matrices.

Next, you use three matrix statements to specify the parameters in the matrix elements. The specifications here reflect exactly the initial specifications for the LINEQS model matrices as shown in Output 33.29.10, Output 33.29.11, and Output 33.29.12.

In the VNAMES statement, you specify the column variable names for the matrices. The column variable names of the **J** matrix include all the observed variable names and the names of the intended endogenous latent factors f_Alien67 and f_Alien71. The column variable names for the **Beta** matrix are the same as those for matrix **J**. The column variables for the **Gamma** matrix include the intended latent factor f_SES and error variable names e1–e6 and d1–d2, which are arranged in such a way that they match the order of the error variables in the LINEQS output shown in Output 33.29.12.

Output 33.29.13 shows the covariance structures and some properties of the model matrices. The covariance structure formula for **Sigma** is defined as required. You can also check the matrix properties in this output to see if they are what you intend them to be.

Output 33.29.13 The Covariance Structures and Model Matrices of the COSAN Model for the Wheaton Data: EQS Emulation

COSAN Model Structures
Sigma = J*inv(_I_-Beta)*Gamma*Phi*Gamma`*(inv(_I_-Beta))`*J`

Output 33.29.13 *continued*

Summary of Model Matrices			
Matrix	**N Row**	**N Col**	**Matrix Type**
Beta	8	8	GEN: Square
Gamma	8	9	GEN: Rectangular
J	6	8	IDE: (I || 0)
Phi	9	9	SYM: Symmetric

Output 33.29.14 shows the model fit chi-square of the current COSAN model. As expected, this is the same model fit as in Output 33.17.6 of Example 33.17 and in Output 33.29.5.

Output 33.29.14 Model Fit of the COSAN Model for the Wheaton Data: EQS Emulation

Fit Summary	
Chi-Square	13.4851
Chi-Square DF	9
Pr > Chi-Square	0.1419

Output 33.29.15 shows the estimates of the **Beta** matrix by the COSAN model specification. These estimates are essentially the same as the estimates of the _EQSBETA_ matrix obtained from the LINEQS model specification, as shown in Output 33.29.16.

Output 33.29.15 Estimate of the **Beta** Matrix by the COSAN Model Specification

Model Matrix Beta (8 x 8 General Square Matrix)								
	Anomie67	**Anomie71**	**Education**	**Powerless67**	**Powerless71**	**SEI**	**f_Alien67**	**f_Alien71**
Anomie67	0	0	0	0	0	0	1.0000	0
Anomie71	0	0	0	0	0	0	0	1.0000
Education	0	0	0	0	0	0	0	0
Powerless67	0	0	0	0	0	0	0.8330	0
Powerless71	0	0	0	0	0	0	0	0.8330
SEI	0	0	0	0	0	0	0	0
f_Alien67	0	0	0	0	0	0	0	0
f_Alien71	0	0	0	0	0	0	0.5931 [beta]	0

Output 33.29.16 Estimate of the _EQSBETA_ Matrix by the LINEQS Model Specification

EQSBETA Matrix								
	Anomie67	Anomie71	Education	Powerless67	Powerless71	SEI	f_Alien67	f_Alien71
Anomie67	0	0	0	0	0	0	1.0000	0
Anomie71	0	0	0	0	0	0	0	1.0000
Education	0	0	0	0	0	0	0	0
Powerless67	0	0	0	0	0	0	0.8330	0
Powerless71	0	0	0	0	0	0	0	0.8330
SEI	0	0	0	0	0	0	0	0
f_Alien67	0	0	0	0	0	0	0	0
f_Alien71	0	0	0	0	0	0	0.5931 [beta]	0

Output 33.29.17 shows the estimates of the **Gamma** matrix by the COSAN model specification. Again, these estimates are essentially the same as the estimates of the _EQSGAMMA_ matrix obtained from the LINEQS model specification, as shown in Output 33.29.18.

Output 33.29.17 Estimate of the Gamma Matrix by the COSAN Model Specification

Model Matrix Gamma (8 x 9 General Rectangular Matrix)									
	f_SES	e1	e3	e5	e2	e4	e6	d1	d2
Anomie67	0	1.0000	0	0	0	0	0	0	0
Anomie71	0	0	1.0000	0	0	0	0	0	0
Education	1.0000	0	0	1.0000	0	0	0	0	0
Powerless67	0	0	0	0	1.0000	0	0	0	0
Powerless71	0	0	0	0	0	1.0000	0	0	0
SEI	5.3689 [lambda]	0	0	0	0	0	1.0000	0	0
f_Alien67	-0.6299 [gamma1]	0	0	0	0	0	0	1.0000	0
f_Alien71	-0.2409 [gamma2]	0	0	0	0	0	0	0	1.0000

Output 33.29.18 Estimate of the _EQSGAMMA_ Matrix by the LINEQS Model Specification

EQSGAMMA Matrix									
	f_SES	e1	e3	e5	e2	e4	e6	d1	d2
Anomie67	0	1.0000	0	0	0	0	0	0	0
Anomie71	0	0	1.0000	0	0	0	0	0	0
Education	1.0000	0	0	1.0000	0	0	0	0	0
Powerless67	0	0	0	0	1.0000	0	0	0	0
Powerless71	0	0	0	0	0	1.0000	0	0	0
SEI	5.3688 [lambda]	0	0	0	0	0	1.0000	0	0
f_Alien67	-0.6299 [gamma1]	0	0	0	0	0	0	1.0000	0
f_Alien71	-0.2409 [gamma2]	0	0	0	0	0	0	0	1.0000

Finally, Output 33.29.19 shows the estimates of the **Phi** matrix by the COSAN model specification. These estimates are essentially the same as the estimates of the _EQSPHI_ matrix obtained from the LINEQS model specification, as shown in Output 33.29.20.

Output 33.29.19 Estimate of the **Phi** Matrix by the COSAN Model Specification

Model Matrix Phi (9 x 9 Symmetric Matrix)									
	f_SES	e1	e3	e5	e2	e4	e6	d1	d2
f_SES	6.6162 [phi]	0	0	0	0	0	0	0	0
e1	0	3.6078 [theta1]	0.9058 [theta5]	0	0	0	0	0	0
e3	0	0.9058 [theta5]	3.6078 [theta1]	0	0	0	0	0	0
e5	0	0	0	2.9938 [theta3]	0	0	0	0	0
e2	0	0	0	0	3.5950 [theta2]	0.9058 [theta5]	0	0	0
e4	0	0	0	0	0.9058 [theta5]	3.5950 [theta2]	0	0	0
e6	0	0	0	0	0	0	259.5738 [theta4]	0	0
d1	0	0	0	0	0	0	0	5.6705 [psi1]	0
d2	0	0	0	0	0	0	0	0	4.5148 [psi2]

Output 33.29.20 Estimate of the _EQSPHI_ Matrix by the LINEQS Model Specification

EQSPHI Matrix									
	f_SES	**e1**	**e3**	**e5**	**e2**	**e4**	**e6**	**d1**	**d2**
f_SES	6.6163 [phi]	0	0	0	0	0	0	0	0
e1	0	3.6080 [theta1]	0.9058 [theta5]	0	0	0	0	0	0
e3	0	0.9058 [theta5]	3.6080 [theta1]	0	0	0	0	0	0
e5	0	0	0	2.9937 [theta3]	0	0	0	0	0
e2	0	0	0	0	3.5949 [theta2]	0.9058 [theta5]	0	0	0
e4	0	0	0	0	0.9058 [theta5]	3.5949 [theta2]	0	0	0
e6	0	0	0	0	0	0	259.5764 [theta4]	0	0
d1	0	0	0	0	0	0	0	5.6705 [psi1]	0
d2	0	0	0	0	0	0	0	0	4.5148 [psi2]

Example 33.30: Second-Order Confirmatory Factor Analysis

(View the complete code for this example (calex402.sas) in the example repository.)

A second-order confirmatory factor analysis model is applied to a correlation matrix of Thurstone reported by McDonald (1985). The data set is shown in the following DATA step:

```
data Thurst(TYPE=CORR);
title "Example of THURSTONE resp. McDONALD (1985, p.57, p.105)";
   _TYPE_ = 'CORR'; Input _NAME_ $ Obs1-Obs9;
   label obs1='Sentences' obs2='Vocabulary' obs3='Sentence Completion'
         obs4='First Letters' obs5='Four-letter Words' obs6='Suffices'
         obs7='Letter series' obs8='Pedigrees' obs9='Letter Grouping';
   datalines;
obs1  1.      .      .      .      .      .      .      .
obs2   .828   1.      .      .      .      .      .      .
obs3   .776   .779   1.      .      .      .      .      .
obs4   .439   .493    .460  1.      .      .      .      .
obs5   .432   .464    .425   .674  1.      .      .      .
obs6   .447   .489    .443   .590   .541  1.      .      .
obs7   .447   .432    .401   .381   .402   .288  1.      .
obs8   .541   .537    .534   .350   .367   .320   .555  1.      .
obs9   .380   .358    .359   .424   .446   .325   .598   .452  1.
;
```

Using the LINEQS modeling language, you specify the three-term second-order factor analysis model in the following statements:

```
proc calis data=Thurst nobs=213 corr nose;
lineqs
   obs1 =  x1 * f1 + e1,
   obs2 =  x2 * f1 + e2,
   obs3 =  x3 * f1 + e3,
   obs4 =  x4 * f2 + e4,
   obs5 =  x5 * f2 + e5,
   obs6 =  x6 * f2 + e6,
   obs7 =  x7 * f3 + e7,
   obs8 =  x8 * f3 + e8,
   obs9 =  x9 * f3 + e9,
   f1   = x10 * f4 + e10,
   f2   = x11 * f4 + e11,
   f3   = x12 * f4 + e12;
variance
   f4      = 1.,
   e1-e9   = u1-u9,
   e10-e12 = 3 * 1.;
bounds
   0. <= u1-u9;
run;
```

In the PROC CALIS statement, you specify the data set in the DATA= option and the number of observations in the NOBS= option. With the CORR option, you request the correlations be analyzed. You use the NOSE option to suppress the computation of standard error estimates.

In the LINEQS statement, the first-order loadings for the three factors, f1, f2, and f3, each refer to three variables, X1–X3, X4–X6, and X7–X9, respectively. One second-order factor, f4, reflects the correlations among the three first-order factors, f1, f2, and f3.

In the VARIANCE statement, you fix the variance of f4 to 1.0 for identification. The variances of error terms e1–e9 are free parameters u1–u9. The error variances for the three first-order factors are also fixed at 1.0 for identification purposes.

You also specify the boundary constraints for the error variance parameters u1–u9. You require them to be positive in the estimation.

Output 33.30.1 shows the estimation results.

Output 33.30.1 Estimation Results of the Second-Order Factor Model for Thurstone Data: LINEQS Model

Linear Equations

Obs1	=	0.5151	f1	+	1.0000	e1
Obs2	=	0.5203	f1	+	1.0000	e2
Obs3	=	0.4874	f1	+	1.0000	e3
Obs4	=	0.5211	f2	+	1.0000	e4
Obs5	=	0.4971	f2	+	1.0000	e5
Obs6	=	0.4381	f2	+	1.0000	e6
Obs7	=	0.4524	f3	+	1.0000	e7
Obs8	=	0.4173	f3	+	1.0000	e8
Obs9	=	0.4076	f3	+	1.0000	e9
f1	=	1.4438	f4	+	1.0000	e10
f2	=	1.2538	f4	+	1.0000	e11
f3	=	1.4065	f4	+	1.0000	e12

Output 33.30.1 *continued*

Estimates for Variances of Exogenous Variables			
Variable Type	Variable	Parameter	Estimate
Latent	f4		1.00000
Error	e1	u1	0.18150
	e2	u2	0.16493
	e3	u3	0.26713
	e4	u4	0.30150
	e5	u5	0.36450
	e6	u6	0.50642
	e7	u7	0.39033
	e8	u8	0.48138
	e9	u9	0.50509
Disturbance	e10		1.00000
	e11		1.00000
	e12		1.00000

Alternatively, you can use the COSAN model specification for analyzing the same data set. First, under the second-order factor model, the covariance structures of the observed variables can be derived as

$$\mathbf{\Sigma} = \mathbf{F1} * \mathbf{F2} * \mathbf{P} * \mathbf{F2}' * \mathbf{F1}' + \mathbf{F1} * \mathbf{U2} * \mathbf{F1}' + \mathbf{U1}$$

where $\mathbf{F1}$ is the 9×3 first-order loading matrix for the observed variables, $\mathbf{F2}$ is the 3×1 second-order loading matrix for the first-order factors, $\mathbf{P}$ is the 1×1 covariance matrix for the second-order factor f4, $\mathbf{U2}$ is the 3×3 error covariance matrix of the first-order factors f1–f3 (or the covariance matrix of the error terms e10–12), and $\mathbf{U1}$ is the 9×9 error covariance matrix for the observed variables (or the covariance matrix of the error terms e1–9).

Matrix $\mathbf{F1}$ contains the loading parameters x1–x9 and matrix $\mathbf{F2}$ contains the loading parameters x10–x12. Because there is only one second-order factor f4 in the model, matrix $\mathbf{P}$ is a scalar, which is a fixed constant 1 in the LINEQS model. Matrix $\mathbf{U2}$ is an identity matrix because all error variances are fixed at 1 and they are not correlated. Matrix $\mathbf{U2}$ is a diagonal matrix that contains the parameters u1–u9. Given this information, you can use the following statements to specify the second-order factor model as a COSAN model:

```
proc calis data=Thurst nobs=213 corr nose;
   cosan
      var = obs1-obs9,
      F1(3) * F2(1) * P(1,IDE) + F1(3) * U2(3,IDE) + U1(9,DIA);
   matrix F1
      [1 , @1] = x1-x3,
      [4 , @2] = x4-X6,
      [7 , @3] = x7-x9;
   matrix F2
      [ ,1]    = x10-x12;
   matrix U1
      [1,1]    = u1-u9;
   bounds
      0. <= u1-u9;
   vnames
```

```
        F1 = [f1 f2 f3],
        F2 = [f4],
        U1 = [e1-e9];
run;
```

In the PROC CALIS statement, you specify the observed variables in the VAR= option and the covariance structures for the observed variables. In the terms of the covariance structure formula, you need to specify the expressions only up the central symmetric matrices. The latter parts of these expressions are redundant and can be generated automatically by PROC CALIS, as shown in Output 33.30.2.

Output 33.30.2 The Covariance Structures and Model Matrices of the Second-Order Factor Model: COSAN Model

COSAN Model Structures
Sigma = F1*F2*P*F2`*F1` + F1*U2*F1` + U1

Summary of Model Matrices			
Matrix	N Row	N Col	Matrix Type
F1	9	3	GEN: Rectangular
F2	3	1	GEN: Vector
P	1	1	IDE: Identity
U1	9	9	DIA: Diagonal
U2	3	3	IDE: Identity

Output 33.30.2 shows that the intended covariance structures for the observed variables are being analyzed. The matrix types are shown next. Matrix **F1** is a rectangular matrix and matrix **F2** is a vector, although they have the default general (GEN) matrix type. Matrices **P** and **U2** are fixed identity (IDE) matrices in the model. For these two matrices, you do not need to specify any of their elements by using the MATRIX statement because they are already well-defined with the IDE type. Lastly, matrix **U1** is a diagonal (DIA) matrix in the model.

Output 33.30.3 shows the estimates of the first-order factor loading matrix **F1**.

Output 33.30.3 Estimation of the $\mathbf{F1}$ Matrix of the Second-Order Factor Model: COSAN Model

Model Matrix F1 (9 x 3 General Rectangular Matrix)			
	f1	**f2**	**f3**
Obs1	0.5151 [x1]	0	0
Obs2	0.5203 [x2]	0	0
Obs3	0.4874 [x3]	0	0
Obs4	0	0.5211 [x4]	0
Obs5	0	0.4971 [x5]	0
Obs6	0	0.4381 [x6]	0
Obs7	0	0	0.4524 [x7]
Obs8	0	0	0.4173 [x8]
Obs9	0	0	0.4076 [x9]

In the MATRIX statement for $\mathbf{F1}$, you specify the pattern of the loadings. In the first entry of the MATRIX statement, you specify the loadings in the following elements: `[1,1]`, `[2,1]`, and `[3,1]`. They are free parameters x1–x3, respectively. Notice that the `@` sign is necessary in the first entry because the elements being defined would have been `[1,1]`, `[2,2]`, and `[3,3]` otherwise. The `@` sign fixes the column number to 1. See the MATRIX statement for more details about the notation. Similarly, you define the other clusters of loading in the second and third entries in the MATRIX statement for $\mathbf{F1}$. This explains the pattern of factor loadings in Output 33.30.3. These loading estimates x1–x9 match those by the LINEQS model specification, as shown in Output 33.30.1.

Output 33.30.3 shows the estimates of the second-order factor loading matrix $\mathbf{F2}$.

Output 33.30.4 Estimation of the $\mathbf{F2}$ Matrix of the Second-Order Factor Model: COSAN Model

Model Matrix F2 (3 x 1 Column Vector)	
	f4
f1	1.4438 [x10]
f2	1.2538 [x11]
f3	1.4066 [x12]

In the MATRIX statement for $\mathbf{F2}$, you do not specify the row numbers in the `[ ,1]` specification. PROC CALIS interprets this as stating that all the valid elements in the first column are being specified in the parameter list. In the current example, this means that elements $\mathbf{F2}[1, 1]$, $\mathbf{F2}[2, 1]$, and $\mathbf{F2}[3, 1]$ are filled with the free parameters x10, x11, and x12, respectively. Output 33.30.3 shows these specification and the corresponding estimates, which match those of the LINEQS model specification, as shown in Output 33.30.1.

Output 33.30.5 shows the estimates of the error covariance matrix **U1**.

Output 33.30.5 Estimation of the **U1** Matrix of the Second-Order Factor Model: COSAN Model

Model Matrix U1 (9 x 9 Diagonal Matrix)									
	e1	e2	e3	e4	e5	e6	e7	e8	e9
e1	0.1815 [u1]	0	0	0	0	0	0	0	0
e2	0	0.1649 [u2]	0	0	0	0	0	0	0
e3	0	0	0.2671 [u3]	0	0	0	0	0	0
e4	0	0	0	0.3015 [u4]	0	0	0	0	0
e5	0	0	0	0	0.3645 [u5]	0	0	0	0
e6	0	0	0	0	0	0.5064 [u6]	0	0	0
e7	0	0	0	0	0	0	0.3903 [u7]	0	0
e8	0	0	0	0	0	0	0	0.4814 [u8]	0
e9	0	0	0	0	0	0	0	0	0.5051 [u9]

In the MATRIX statement for **U1**, you specify the diagonal elements of the matrix by using the starting element at `[1,1]`. The parameter assignment proceeds to `[2,2]`, `[3,3]` and so on such that all the trailing parameters u1–u9 are filled. This means that the last element $\mathbf{U1}[9,9]$ is a free parameter named u9. Output 33.30.5 confirms this intended pattern. Again, all these error variance estimates match those by the LINEQS model specification, as shown in Output 33.30.1.

Example 33.31: Linear Relations among Factor Loadings: COSAN Model Specification

(View the complete code for this example (calex403.sas) in the example repository.)

This example reanalyzes the models in Example 33.27 by using the COSAN modeling language. The correlation matrix of six variables from Kinzer and Kinzer (N=326) is used (see Guttman 1957). McDonald (1980) uses this data set to demonstrate the fitting of a factor analysis model with linear constraints on factor loadings. Two factors are assumed for the data. The factor loading matrix **B** is shown in the following:

$$\mathbf{B} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \\ b_{41} & b_{42} \\ b_{51} & b_{52} \\ b_{61} & b_{62} \end{pmatrix}$$

The loadings on the second factor are linearly related to the loadings on the first factor, as described by the following formula:

$$b_{j2} = \alpha - b_{j1}, \quad j = 1, \ldots, 6$$

The correlation structures are represented by

$$\mathbf{P} = \mathbf{BB}' + \mathbf{\Psi}$$

where $\Psi = \mathrm{diag}(\psi_{11}, \psi_{22}, \psi_{33}, \psi_{44}, \psi_{55}, \psi_{66})$ represents the diagonal matrix of unique variances for the variables. Because matrix $\mathbf{P}$ is a correlation matrix, its diagonal elements are fixed constants 1. This means that the diagonal elements of the correlation structures must also satisfy the following condition:

$$\mathbf{\Psi}_{jj} = 1. - b_{j1}^2 - b_{j2}^2, \quad j = 1, \ldots, 6$$

To analyze the correlation structures by using PROC CALIS, you formulate a covariance structure model with such correlation structures embedded in the model. That is, you want to fit the following covariance structure model to the Kinzer data:

$$\mathbf{\Sigma} = \mathbf{DPD}' = \mathbf{D}(\mathbf{BB}' + \mathbf{\Psi})\mathbf{D}' = \mathbf{DBB}'\mathbf{D}' + \mathbf{D\Psi D}'$$

where $\mathbf{D}$ is a 6 x 6 diagonal matrix that contains the population standard deviations of the observed variables.

The following statements use the COSAN modeling language to specify this covariance structure model:

```
proc calis data=Kinzer nobs=326 nose;
   cosan
      var= var1-var6,
      D(6,DIA) * B(2,GEN) + D(6,DIA) * Psi(6,DIA);
   matrix B
      [ ,1] = b11 b21 b31 b41 b51 b61,
      [ ,2] = b12 b22 b32 b42 b52 b62;
   matrix Psi
      [1,1] = psi1-psi6;
   matrix D
      [1,1] = d1-d6;
   parameters alpha (1.);

   /* SAS Programming Statements to Define Dependent Parameters*/
   /* 6 constraints on the factor loadings */
   b12  = alpha - b11;
   b22  = alpha - b21;
   b32  = alpha - b31;
   b42  = alpha - b41;
   b52  = alpha - b51;
   b62  = alpha - b61;

   /* 6 Constraints on Correlation structures */
   psi1 = 1. - b11 * b11 - b12 * b12;
   psi2 = 1. - b21 * b21 - b22 * b22;
   psi3 = 1. - b31 * b31 - b32 * b32;
   psi4 = 1. - b41 * b41 - b42 * b42;
   psi5 = 1. - b51 * b51 - b52 * b52;
   psi6 = 1. - b61 * b61 - b62 * b62;
```

```
    vnames
       D   = [var1-var6],
       B   = [factor1 factor2],
       Psi = D;
run;
```

In the PROC CALIS statement, you specify the data set by the DATA= option and the number of observations by the NOBS= option. You also use the NOSE option to suppress the printing of the standard error estimates.

In the COSAN statement, you specify the variables for the covariance structure analysis in the VAR= option. Next, you specify the covariance structure formula for the variables. When generating the covariance structure expressions for the terms, PROC CALIS examines the matrix type of the last matrix in each term to determine how the expression is generated. If the last matrix in a term is not a symmetric matrix (including diagonal or identity matrix), the transpose of the last matrix would be included in the expression. This ensures that a symmetric matrix expression is formed for the covariance structures. For example, the first term in the current covariance structure formula is `D(6,DIA)*B(2,GEN)`. Because **B** is not a symmetric matrix, the expression generated by PROC CALIS is

$$\mathbf{D} * \mathbf{B} * \mathbf{B}' * \mathbf{D}'$$

However, for the second term `D(6,DIA)* Psi(6,DIA)`, matrix **Psi** is a symmetric matrix so that the expression generated by PROC CALIS is

$$\mathbf{D} * \mathbf{Psi} * \mathbf{D}'$$

Output 33.31.1 shows the covariance structure model and the model matrices. With **Psi** representing the unique variance matrix $\mathbf{\Psi}$, the printed covariance structure formula for **Sigma** is clearly what you intend to specify.

Output 33.31.1 The Covariance Structures and Model Matrices: Linearly Constrained Loadings

COSAN Model Structures
Sigma = D*B*B`*D` + D*Psi*D`

Summary of Model Matrices			
Matrix	N Row	N Col	Matrix Type
B	6	2	GEN: Rectangular
D	6	6	DIA: Diagonal
Psi	6	6	DIA: Diagonal

In the MATRIX statements, you specify the parameters in the model matrices. You use parameters with the b prefix to name the two columns of loadings of the factor matrix **B**. You use free parameters psi1–psi6 for the diagonal elements of the **Psi** matrix, and free parameters d1–d6 for the diagonal elements of the **D** matrix. Next, you use a PARAMETERS statement to define an independent parameter alpha in the model. This parameter takes an initial value of 1.0. Using this independent parameter and six SAS programming statements, you define the loadings in the second column of matrix **B** as functions of the loadings in the first column of the same matrix.

You use six more SAS programming statements to define the unique variance parameters psi1–psi6 as dependent parameters of the factor loadings. These constraints ensure that the embedded correlation structures have diagonal elements fixed at 1.0.

Lastly, you use the VNAMES statement to label the column names of the model matrices. The column names of the diagonal matrix **D** are the same as the observed variables. The column names of matrix **B** are for the factor names.

As compared with the covariance structure specification (that is, the second specification) by the FACTOR model in Example 33.27, the current COSAN specification seems to be more direct and concise in specifying the parameter constraints. Because of the direct references to the matrix elements in the COSAN modeling language, you can set the required 12 constraints in a very straightforward way as the 12 SAS programming statements in the preceding specification. However, with the FACTOR model specification language in Example 33.27, you need 18 more SAS programming statements to define the correct constraints for the same covariance structure model.

Output 33.31.2 shows the fit summary table. The chi-square test statistic is 14.63 with $df = 8$ ($p = 0.067$). These are the same model fitting results as using the FACTOR model specification, as shown in Output 33.27.4 of Example 33.27.

Output 33.31.2 Model Fit: Linearly Constrained Loadings with Embedded Correlation Structures

Fit Summary	
Chi-Square	14.6269
Chi-Square DF	8
Pr > Chi-Square	0.0668

Output 33.31.3 shows the estimation of the loading matrix **B**. These estimates of factor loadings are essentially the same as those obtained from the FACTOR model specification, as shown in Output 33.27.6, except that the two columns of the loading matrix **B** are switched. The column switching is not a concern because the factor labels are arbitrary.

Output 33.31.3 Estimation of the **B** Matrix by the COSAN Model Specification

Model Matrix B (6 x 2 General Rectangular Matrix)		
	factor1	**factor2**
var1	0.6318 [b11]	0.3422 [b12]
var2	0.6531 [b21]	0.3210 [b22]
var3	0.4822 [b31]	0.4918 [b32]
var4	0.3985 [b41]	0.5755 [b42]
var5	0.1971 [b51]	0.7769 [b52]
var6	0.3074 [b61]	0.6666 [b62]

Output 33.31.4 shows the estimation of the scaling matrix **D**. All these standard deviation estimates for the observed variables match those obtained from the FACTOR model specification, as shown in Output 33.27.6.

Output 33.31.4 Estimation of the **D** Matrix by the COSAN Model Specification

Model Matrix D (6 x 6 Diagonal Matrix)						
	var1	**var2**	**var3**	**var4**	**var5**	**var6**
var1	1.0077 [d1]	0	0	0	0	0
var2	0	0.9971 [d2]	0	0	0	0
var3	0	0	0.9908 [d3]	0	0	0
var4	0	0	0	0.9909 [d4]	0	0
var5	0	0	0	0	0.9964 [d5]	0
var6	0	0	0	0	0	1.0169 [d6]

Output 33.31.5 shows the estimation of the unique covariance matrix **Psi**. All these unique variance parameter estimates match those obtained from the FACTOR model specification, as shown in Output 33.27.6.

Output 33.31.5 Estimation of the **Psi** Matrix by the COSAN Model Specification

Model Matrix Psi (6 x 6 Diagonal Matrix)						
	var1	**var2**	**var3**	**var4**	**var5**	**var6**
var1	0.4837 [psi1]	0	0	0	0	0
var2	0	0.4705 [psi2]	0	0	0	0
var3	0	0	0.5256 [psi3]	0	0	0
var4	0	0	0	0.5100 [psi4]	0	0
var5	0	0	0	0	0.3576 [psi5]	0
var6	0	0	0	0	0	0.4612 [psi6]

Finally, Output 33.31.6 shows the estimation of the independent parameter alpha. The same estimate of alpha is shown in Output 33.27.6.

Output 33.31.6 Estimation of the Independent Parameter alpha by the COSAN Model Specification

Additional Parameters		
Type	**Parameter**	**Estimate**
Independent	alpha	0.97400

Example 33.32: Ordinal Relations among Factor Loadings

(View the complete code for this example (calex405.sas) in the example repository.)

The same data set as in Example 33.31 is used in McDonald (1980) for analysis with ordinally constrained factor loadings. In Example 33.27, the results of the linearly constrained factor analysis show that the loadings of the two factors are ordered as 2, 1, 3, 4, 6, 5. McDonald (1980) then tests the hypothesis that the factor loadings are all nonnegative and can be ordered in the following manner:

$$b_{11} \geq b_{21} \geq b_{31} \geq b_{41} \geq b_{51} \geq b_{61}$$

$$b_{12} \leq b_{22} \leq b_{32} \leq b_{42} \leq b_{52} \leq b_{62}$$

In this example, you implement these ordinal relationships by using the LINCON statement in the following COSAN model specification:

```
proc calis data=Kinzer nobs=326 nose;
   cosan
      var= var1-var6,
      D(6,DIA) * B(2,GEN) + D(6,DIA) * Psi(6,DIA);
   matrix B
      [ ,1]= b11 b21 b31 b41 b51 b61,
      [ ,2]= 0.  b22 b32 b42 b52 b62;
   matrix Psi
      [1,1]= psi1-psi6;
   matrix D
      [1,1]= d1-d6 ;
   lincon
      b61  <= b51,
      b51  <= b41,
      b41  <= b31,
      b31  <= b21,
      b21  <= b11,
       0.  <= b22,
      b22  <= b32,
      b32  <= b42,
      b42  <= b52,
      b52  <= b62;

   /* SAS Programming Statements */
   /* 6 Constraints on Correlation structures */
   psi1 = 1. - b11 * b11;
   psi2 = 1. - b21 * b21 - b22 * b22;
   psi3 = 1. - b31 * b31 - b32 * b32;
   psi4 = 1. - b41 * b41 - b42 * b42;
   psi5 = 1. - b51 * b51 - b52 * b52;
   psi6 = 1. - b61 * b61 - b62 * b62;
   vnames
       B   = [factor1 factor2],
       Psi = [var1-var6],
       D   = Psi;
run;
```

As in Example 33.31, correlation structures are analyzed in the current example so that the unique variance parameters psi1–psi6 are defined as functions of the loadings in the SAS programming statements. However, the loading parameters are no longer not constrained in the current model. Instead, you impose ordinal constraints on the loading parameters. First, b21 is fixed at 0 for identification purposes. Then, you use the LINCON statement to specify the ordinal relations of the factor loadings.

As shown in Output 33.32.1, the solution converges in 12 iterations. In the fit summary table, the chi-square test statistic is 8.48 ($df = 6$, $p = 0.20$). This indicates a good fit. However, in the model there are 11 loading parameters (the b's) and 6 population standard deviation parameters (the d's). The degrees of freedom should have been 4 = 21 – 11 – 6, but why is this number 6 in the fit summary table?

Output 33.32.1 Final Iteration Status and Fit

Optimization Results			
Iterations	12	**Function Calls**	29
Jacobian Calls	14	**Active Constraints**	2
Objective Function	0.0260990149	**Max Abs Gradient Element**	2.7626747E-6
Lambda	0	**Actual Over Pred Change**	1.1572127502
Radius	0.0000851592		

Convergence criterion (ABSGCONV=0.00001) satisfied.

Fit Summary	
Chi-Square	8.4822
Chi-Square DF	6
Pr > Chi-Square	0.2049

The reason is that there are two active constraints in the solution, resulting in two free parameters fewer in the final solution than originally specified. Active constraints are those inequality constraints that are fulfilled on the boundary equalities. As shown in the "Optimization Results" table, the number of active constraints for the current fitting is two. The default treatment in PROC CALIS is to treat these active constraints as if they were going to happen for all possible repeated sampling. This might as well be seen as fitting the active equality constraints on every possible repeated sample. This results in an increase of the degrees of freedom for model fit, as adjusted in the current fit summary table in Output 33.32.1. To warn you about the degrees-of-freedom adjustment, the following messages are also printed with the output:

```
WARNING: There are 2 active boundary or linear inequality constraints at
         the solution. The standard errors and chi-square test statistic
         assume that the solution is located in the interior of the
         parameter space; hence, they do not apply if it is likely that
         some different set of inequality a constraints could be active.

NOTE: The degrees of freedom are increased by the number of active
      constraints.  The number of parameters in calculating fit indices
      is decreased by the number of active constraints. To turn off the
      adjustment, use the NOADJDF option.
```

When active constraints are encountered, you need to be cautious about two implications. First, the estimates fall on the boundary of the parameter space originally specified. As shown in Output 33.32.2, estimates for b11 and b21 are the same, and so are the pair of estimates for b52 and b62. These pairs of parameters were

originally constrained by inequalities in the model. For example, b62 was constrained to be at least as large as b52. The fact that this constraint is honored only on the bound means that a better model fit might exist with b62 being smaller than b52. Similarly, a better model fit might result without requiring b11 to be at least as large as b21. Therefore, solutions with active boundary constraints might imply that the original strict inequality constraints are not appropriate for the data.

Output 33.32.2 Estimation of the Factor Loading Matrix B

Model Matrix B (6 x 2 General Rectangular Matrix)	factor1	factor2
var1	0.7100 [b11]	0
var2	0.7100 [b21]	0.0393 [b22]
var3	0.6799 [b31]	0.2463 [b32]
var4	0.6561 [b41]	0.3295 [b42]
var5	0.5541 [b51]	0.5432 [b52]
var6	0.4733 [b61]	0.5432 [b62]

Output 33.32.3 Estimation of the Scaling Matrix D and Unique Covariance Matrix Psi

Model Matrix D (6 x 6 Diagonal Matrix)	var1	var2	var3	var4	var5	var6
var1	1.0022 [d1]	0	0	0	0	0
var2	0	0.9985 [d2]	0	0	0	0
var3	0	0	1.0004 [d3]	0	0	0
var4	0	0	0	1.0004 [d4]	0	0
var5	0	0	0	0	0.9990 [d5]	0
var6	0	0	0	0	0	1.0021 [d6]

Output 33.32.3 *continued*

Model Matrix Psi (6 x 6 Diagonal Matrix)						
	var1	var2	var3	var4	var5	var6
var1	0.4959 [psi1]	0	0	0	0	0
var2	0	0.4944 [psi2]	0	0	0	0
var3	0	0	0.4771 [psi3]	0	0	0
var4	0	0	0	0.4610 [psi4]	0	0
var5	0	0	0	0	0.3979 [psi5]	0
var6	0	0	0	0	0	0.4809 [psi6]

The second implication for the presence of active constraints is that the chi-square test statistic and the standard error estimates are computed as if repeated samples were fitted by the model with the presence of the active equality constraints. The degrees-of-freedom adjustment by PROC CALIS is based on this assumption. However, if the particular active constraints reflect only a rare sampling event, the degrees-of-freedom adjustment (or even the computation of the chi-square statistic and standard error estimates) might not be justified. Unfortunately, whether the active constraints are reflecting the truth of the model or pure sampling fluctuation is usually difficult to determine.

Example 33.33: Longitudinal Factor Analysis

(View the complete code for this example (calex406.sas) in the example repository.)

The following example (McDonald 1980) illustrates both the ability of PROC CALIS to formulate complex covariance structure analysis problems by the generalized COSAN matrix model and the use of programming statements to impose nonlinear constraints on the parameters. The example is a longitudinal factor analysis that uses the Swaminathan (1974) model. For $m = 3$ tests, $k = 3$ occasions, and $r = 2$ factors, the covariance structure model is formulated as follows:

$$\mathbf{\Sigma} = \mathbf{F}_1\mathbf{F}_2\mathbf{F}_3\mathbf{L}\mathbf{F}_3^{-1}\mathbf{F}_2^{-1}\mathbf{P}(\mathbf{F}_2^{-1})'(\mathbf{F}_3^{-1})'\mathbf{L}'\mathbf{F}_3'\mathbf{F}_2'\mathbf{F}_1' + \mathbf{U}^2$$

$$\mathbf{F}_1 = \begin{pmatrix} \mathbf{B}_1 & & \\ & \mathbf{B}_2 & \\ & & \mathbf{B}_3 \end{pmatrix}, \quad \mathbf{F}_2 = \begin{pmatrix} \mathbf{I}_2 & & \\ & \mathbf{D}_2 & \\ & & \mathbf{D}_2 \end{pmatrix}, \quad \mathbf{F}_3 = \begin{pmatrix} \mathbf{I}_2 & & \\ & \mathbf{I}_2 & \\ & & \mathbf{D}_3 \end{pmatrix}$$

$$\mathbf{L} = \begin{pmatrix} \mathbf{I}_2 & 0 & 0 \\ \mathbf{I}_2 & \mathbf{I}_2 & 0 \\ \mathbf{I}_2 & \mathbf{I}_2 & \mathbf{I}_2 \end{pmatrix}, \quad \mathbf{P} = \begin{pmatrix} \mathbf{I}_2 & & \\ & \mathbf{S}_2 & \\ & & \mathbf{S}_3 \end{pmatrix}, \quad \mathbf{U} = \begin{pmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} & \mathbf{U}_{13} \\ \mathbf{U}_{21} & \mathbf{U}_{22} & \mathbf{U}_{23} \\ \mathbf{U}_{31} & \mathbf{U}_{32} & \mathbf{U}_{33} \end{pmatrix}$$

$$\mathbf{S}_2 = \mathbf{I}_2 - \mathbf{D}_2^2, \qquad \mathbf{S}_3 = \mathbf{I}_2 - \mathbf{D}_3^2$$

The Swaminathan longitudinal factor model assumes that the factor scores for each (m) common factor change from occasion to occasion (k) according to a first-order autoregressive scheme. The matrix $\mathbf{F}_1$

contains the k factor loading matrices $\mathbf{B}_1$, $\mathbf{B}_2$, and $\mathbf{B}_3$ (each is $n \times m$). The matrices $\mathbf{D}_2, \mathbf{D}_3, \mathbf{S}_2, \mathbf{S}_3$ and $\mathbf{U}_{ij}, i, j = 1, \ldots, k$, are diagonal, and the matrices $\mathbf{D}_i$ and $\mathbf{S}_i, i = 2, \ldots, k$, are subjected to the constraint

$$\mathbf{S}_i + \mathbf{D}_i^2 = \mathbf{I}$$

Although the covariance structure model looks pretty complicated, it poses no problem for the COSAN model specifications. Since the constructed correlation matrix given by McDonald (1980) is singular, only unweighted least squares (METHOD=LS) estimates can be computed. The following statements specify the COSAN model for the correlation structures.

```
Title "Swaminathan's Longitudinal Factor Model, Data: McDONALD(1980)";
Title2 "Constructed Singular Correlation Matrix, GLS & ML not possible";
data Mcdon(TYPE=CORR);
   _TYPE_ = 'CORR'; INPUT _NAME_ $ obs1-obs9;
   datalines;
obs1  1.000    .      .      .      .      .      .      .      .
obs2   .100  1.000    .      .      .      .      .      .      .
obs3   .250   .400  1.000    .      .      .      .      .      .
obs4   .720   .108   .270  1.000    .      .      .      .      .
obs5   .135   .740   .380   .180  1.000    .      .      .      .
obs6   .270   .318   .800   .360   .530  1.000    .      .      .
obs7   .650   .054   .135   .730   .090   .180  1.000    .      .
obs8   .108   .690   .196   .144   .700   .269   .200  1.000    .
obs9   .189   .202   .710   .252   .336   .760   .350   .580  1.000
;

proc calis data=Mcdon method=ls nobs=100 corr;
   cosan
      var = obs1-obs9,
      F1(6,GEN) * F2(6,DIA) * F3(6,DIA) * L(6,LOW) * F3(6,DIA,INV)
                * F2(6,DIA,INV) * P(6,DIA) + U(9,SYM);
   matrix F1
            [1 , @1] = x1-x3,
            [2 , @2] = x4-x5,
            [4 , @3] = x6-x8,
            [5 , @4] = x9-x10,
            [7 , @5] = x11-x13,
            [8 , @6] = x14-x15;
   matrix F2
            [1,1]= 2 * 1. x16 x17 x16 x17;
   matrix F3
            [1,1]= 4 * 1. x18 x19;
   matrix L
            [1,1]= 6 * 1.,
            [3,1]= 4 * 1.,
            [5,1]= 2 * 1.;
   matrix P
            [1,1]= 2 * 1. x20-x23;
   matrix U
            [1,1]= x24-x32,
            [4,1]= x33-x38,
            [7,1]= x39-x41;
   bounds 0. <= x24-x32,
```

```
        -1. <= x16-x19 <= 1.;
    /* SAS programming statements for dependent parameters */
    x20 = 1. - x16 * x16;
    x21 = 1. - x17 * x17;
    x22 = 1. - x18 * x18;
    x23 = 1. - x19 * x19;
run;
```

In the PROC CALIS statement, you use the NOBS= option to specify the number of observations. The CORR option requests the analysis of the correlation matrix.

In the COSAN statement, you list the observed variables for the analysis in the VAR= option. Then you specify the formula for the covariance structures. Notice that in the covariance structure formula, some matrices are specified twice. That is, matrix **F2** and **F3** appear in two different places. Matrices with the same name means that they are identical—which certainly makes sense. In addition, you can apply different transformations to the same matrix in different locations of the matrix formula. For example, you do not transform matrix **F2** in the first location, but the same matrix is inverted (INV) later in the expression. Similarly for matrix **F3**.

Next, you define the parameters in the six distinct model matrices by six MATRIX statements. Each matrix has some specific patterns under the covariance structure model. For the **F1** matrix, it has the following pattern for the free parameters in the model:

	col1	col2	col3	col4	col5	col6
row1	x					
row2	x	x				
row3	x	x				
row4			x			
row5			x	x		
row6			x	x		
row7					x	
row8					x	x
row9					x	x

To specify these parameters, you can use some shorthand notation in the MATRIX statement. For example, in the first entry of the MATRIX statement for matrix **F1**, you use the notation `[1,@1]`. This means that the parameter specification starts with the `[1,1]` element and proceeds to the next element while fixing the column number at 1. Hence, parameters x1–x3 are specified for the **F1**[1, 1], **F1**[2, 1], and **F1**[3, 1] elements, respectively. Similarly, you specify other parameters in the **F1** matrix in a column by column fashion.

If you do not use the `@` sign in the specification, the parameters are assigned differently. For example, in the specification of the **L** matrix, the first entry in the corresponding MATRIX statement also starts with the `[1,1]` element. But it proceeds down to `[2,2]`, `[3,3]`, and so on because the `@` sign is not used to fix any column or row number. As a result, the MATRIX statement for **L** specifies the following pattern:

	col1	col2	col3	col4	col5	col6
row1	1					
row2		1				
row3	1		1			
row4		1		1		
row5	1		1		1	
row6		1		1		1

The unspecified elements are fixed zeros in the model.

Similarly, you specify the diagonal matrices **F2**, **F3**, and **P**, and the symmetric matrix **U**.

You also set bounds for some parameters in the BOUNDS statement and some dependent parameters in the SAS programming statements.

Output 33.33.1 shows the correlation structures and the model matrices in the analysis. All appear to be intended.

Output 33.33.1 The Correlation Structures and Model Matrices of the Longitudinal Factor Model

COSAN Model Structures
Sigma = F1*F2*F3*L*inv(F3)*inv(F2)*P*(inv(F2))`*(inv(F3))`*L`*F3`*F2`*F1` + U

Summary of Model Matrices			
Matrix	**N Row**	**N Col**	**Matrix Type**
F1	9	6	GEN: Rectangular
F2	6	6	DIA: Diagonal
F3	6	6	DIA: Diagonal
L	6	6	LOW: L Triangular
P	6	6	DIA: Diagonal
U	9	9	SYM: Symmetric

PROC CALIS finds a converged solution for the estimation problem. Output 33.33.2, Output 33.33.3, and Output 33.33.4 show the estimation results of the **F1**, **F2**, and **F3** matrices, respectively.

Output 33.33.2 Estimation of the **F1** Matrix of the Longitudinal Factor Model

Model Matrix F1 (9 x 6 General Rectangular Matrix)						
	Col1	**Col2**	**Col3**	**Col4**	**Col5**	**Col6**
obs1	0.3515 [x1]	0	0	0	0	0
obs2	0.2871 [x2]	0.9528 [x4]	0	0	0	0
obs3	0.7101 [x3]	0.2059 [x5]	0	0	0	0
obs4	0	0	0.4204 [x6]	0	0	0
obs5	0	0	0.4303 [x7]	0.9027 [x9]	0	0
obs6	0	0	0.8591 [x8]	0.1772 [x10]	0	0
obs7	0	0	0	0	0.3487 [x11]	0
obs8	0	0	0	0	0.5924 [x12]	-0.1971 [x14]
obs9	0	0	0	0	0.9987 [x13]	0.0871 [x15]

Output 33.33.3 Estimation of the $\mathbf{F2}$ Matrix of the Longitudinal Factor Model

Model Matrix F2 (6 x 6 Diagonal Matrix)						
	Col1	Col2	Col3	Col4	Col5	Col6
Row1	1.0000	0	0	0	0	0
Row2	0	1.0000	0	0	0	0
Row3	0	0	0.8939 [x16]	0	0	0
Row4	0	0	0	0.5806 [x17]	0	0
Row5	0	0	0	0	0.8939 [x16]	0
Row6	0	0	0	0	0	0.5806 [x17]

Output 33.33.4 Estimation of the $\mathbf{F3}$ Matrix of the Longitudinal Factor Model

Model Matrix F3 (6 x 6 Diagonal Matrix)						
	Col1	Col2	Col3	Col4	Col5	Col6
Row1	1.0000	0	0	0	0	0
Row2	0	1.0000	0	0	0	0
Row3	0	0	1.0000	0	0	0
Row4	0	0	0	1.0000	0	0
Row5	0	0	0	0	0.5963 [x18]	0
Row6	0	0	0	0	0	1.0000 [x19]

Output 33.33.5 shows the estimation results of the $\mathbf{L}$ matrix, which is a fixed matrix that contains only 0 or 1 for its elements.

Output 33.33.5 Estimation of the $\mathbf{L}$ Matrix of the Longitudinal Factor Model

Model Matrix L (6 x 6 Lower Triangular Matrix)						
	Col1	Col2	Col3	Col4	Col5	Col6
Row1	1.0000	0	0	0	0	0
Row2	0	1.0000	0	0	0	0
Row3	1.0000	0	1.0000	0	0	0
Row4	0	1.0000	0	1.0000	0	0
Row5	1.0000	0	1.0000	0	1.0000	0
Row6	0	1.0000	0	1.0000	0	1.0000

Output 33.33.6 shows the estimation results of the $\mathbf{P}$ matrix. Notice that parameter estimate x23 falls on the

lower boundary at zero.

Output 33.33.6 Estimation of the **P** Matrix of the Longitudinal Factor Model

Model Matrix P (6 x 6 Diagonal Matrix)						
	Col1	Col2	Col3	Col4	Col5	Col6
Row1	1.0000	0	0	0	0	0
Row2	0	1.0000	0	0	0	0
Row3	0	0	0.2010 [x20]	0	0	0
Row4	0	0	0	0.6629 [x21]	0	0
Row5	0	0	0	0	0.6444 [x22]	0
Row6	0	0	0	0	0	0 [x23]

In fact, PROC CALIS routinely checks for zero values for the estimates on the diagonal of the central symmetric matrices. In this case, you get the following messages regarding the estimation of matrix **P**:

```
WARNING: Although all predicted variances for the observed variables are
         positive, the corresponding predicted covariance matrix is not
         positive definite. It has one negative eigenvalue.
WARNING: The estimated variance of variable 6 is essentially zero in the
         central matrix P of term 1 of the COSAN model.
WARNING: The central matrix P of term 1 of the COSAN model is not positive
         definite. It has one zero eigenvalue.
```

Output 33.33.7 shows the estimation results of the **U** matrix. Parameter estimates x28 and x32 fall on the lower boundary at zero. PROC CALIS issues the following messages regarding the estimation of matrix **U**:

```
WARNING: The estimated variance of obs5 is essentially zero in the central
         matrix U of term 2 of the COSAN model.
WARNING: The estimated variance of obs9 is essentially zero in the central
         matrix U of term 2 of the COSAN model.
WARNING: The central matrix U of term 2 of the COSAN model is not positive
         definite. It has 3 negative eigenvalues.
```

Output 33.33.7 Estimation of the **U** Matrix of the Longitudinal Factor Model

Model Matrix U (9 x 9 Symmetric Matrix)									
	obs1	**obs2**	**obs3**	**obs4**	**obs5**	**obs6**	**obs7**	**obs8**	**obs9**
obs1	0.8764 [x24]	0	0	0.5879 [x33]	0	0	0.5847 [x39]	0	0
obs2	0	0.009683 [x25]	0	0	0.1302 [x34]	0	0	0.7084 [x40]	0
obs3	0	0	0.4533 [x26]	0	0	0.2335 [x35]	0	0	0.3215 [x41]
obs4	0.5879 [x33]	0	0	0.8233 [x27]	0	0	0.6426 [x36]	0	0
obs5	0	0.1302 [x34]	0	0	0 [x28]	0	0	0.7259 [x37]	0
obs6	0	0	0.2335 [x35]	0	0	0.2305 [x29]	0	0	0.2329 [x38]
obs7	0.5847 [x39]	0	0	0.6426 [x36]	0	0	0.8784 [x30]	0	0
obs8	0	0.7084 [x40]	0	0	0.7259 [x37]	0	0	0.6102 [x31]	0
obs9	0	0	0.3215 [x41]	0	0	0.2329 [x38]	0	0	0 [x32]

Because this formulation of Swaminathan's model in general leads to an unidentified problem, the results given here are different from those reported by McDonald (1980). The displayed output of PROC CALIS also indicates that the fitted central model matrices **P** and **U** are not positive-definite. The BOUNDS statement constrains the diagonals of the matrices **P** and **U** to be nonnegative, but this cannot prevent **U** from having three negative eigenvalues. The fact that many of the published results for more complex models in covariance structure analysis are connected to unidentified problems implies that more theoretical work should be done to study the general features of such models.

References

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Akaike, H. (1987). "Factor Analysis and AIC." *Psychometrika* 52:317–332.

Bartlett, M. S. (1950). "Tests of Significance in Factor Analysis." *British Journal of Psychology* 3:77–85.

Bartlett, M. S. (1954). "A Note on Multiplying Factors for Various Chi-Squared Approximations." *Journal of the Royal Statistical Society, Series B* 16:296–298.

Beale, E. M. L. (1972). "A Derivation of Conjugate Gradients." In *Numerical Methods for Nonlinear Optimization*, edited by F. A. Lootsma, 39–43. London: Academic Press.

Belsley, D. A., Kuh, E., and Welsch, R. E. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. New York: John Wiley & Sons.

Bentler, P. M. (1985). *Theory and Implementation of EQS: A Structural Equations Program.* Manual for Program Version 2.0. Los Angeles: BMDP Statistical Software.

Bentler, P. M. (1986). *Lagrange Multiplier and Wald Tests for EQS and EQS/PC*. Los Angeles: BMDP Statistical Software.

Bentler, P. M. (1995). *EQS: Structural Equations Program Manual.* Program Version 5.0. Encino, CA: Multivariate Software.

Bentler, P. M., and Bonett, D. G. (1980). "Significance Tests and Goodness of Fit in the Analysis of Covariance Structures." *Psychological Bulletin* 88:588–606.

Bentler, P. M., and Freeman, E. H. (1983). "Test for Stability in Linear Structural Equation Systems." *Psychometrika* 48:143–145.

Bentler, P. M., and Weeks, D. G. (1980). "Linear Structural Equations with Latent Variables." *Psychometrika* 45:289–308.

Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1975). *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, MA: MIT Press.

Bollen, K. A. (1986). "Sample Size and Bentler and Bonett's Nonnormed Fit Index." *Psychometrika* 51:375–377.

Bollen, K. A. (1989a). "A New Incremental Fit Index for General Structural Equation Models." *Sociological Methods and Research* 17:303–316.

Bollen, K. A. (1989b). *Structural Equations with Latent Variables*. New York: John Wiley & Sons.

Box, G. E. P. (1949). "A General Distribution Theory for a Class of Likelihood Criteria." *Biometrika* 36:317–346.

Bozdogan, H. (1987). "Model Selection and Akaike's Information Criterion (AIC): The General Theory and Its Analytical Extensions." *Psychometrika* 52:345–370.

Browne, M. W. (1974). "Generalized Least Squares Estimators in the Analysis of Covariance Structures." *South African Statistical Journal* 8:1–24.

Browne, M. W. (1982). "Covariance Structures." In *Topics in Applied Multivariate Analysis*, edited by D. M. Hawkins, 72–141. Cambridge: Cambridge University Press.

Browne, M. W. (1984). "Asymptotically Distribution-Free Methods for the Analysis of Covariance Structures." *British Journal of Mathematical and Statistical Psychology* 37:62–83.

Browne, M. W., and Cudeck, R. (1993). "Alternative Ways of Assessing Model Fit." In *Testing Structural Equation Models*, edited by K. A. Bollen and J. S. Long, 136–162. Newbury Park, CA: Sage Publications.

Browne, M. W., and Du Toit, S. H. C. (1992). "Automated Fitting of Nonstandard Models." *Multivariate Behavioral Research* 27:269–300.

Browne, M. W., and Shapiro, A. (1986). "The Asymptotic Covariance Matrix of Sample Correlation Coefficients under General Conditions." *Linear Algebra and Its Applications* 82:169–176.

Bunch, J. R., and Kaufman, K. (1977). "Some Stable Methods for Calculating Inertia and Solving Symmetric Linear Systems." *Mathematics of Computation* 31:162–179.

Buse, A. (1982). "The Likelihood Ratio, Wald, and Lagrange Multiplier Tests: An Expository Note." *American Statistician* 36:153–157.

Crawford, C. B., and Ferguson, G. A. (1970). "A General Rotation Criterion and Its Use in Orthogonal Rotation." *Psychometrika* 35:321–332.

De Leeuw, J. (1983). "Models and Methods for the Analysis of Correlation Coefficients." *Journal of Econometrics* 22:113–137.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Dijkstra, T. K. (1992). "On Statistical Inference with Parameter Estimates on the Boundary of the Parameter Space." *British Journal of Mathematical and Statistical Psychology* 45:289–309.

Duncan, O. D., Haller, A. O., and Portes, A. (1968). "Peer Influences on Aspirations: A Reinterpretation." *American Journal of Sociology* 74:119–137.

Everitt, B. S. (1984). *An Introduction to Latent Variable Methods*. London: Chapman & Hall.

Fletcher, R. (1980). *Unconstrained Optimization.* Vol. 1 of Practical Methods of Optimization. Chichester, UK: John Wiley & Sons.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Fuller, W. A. (1987). *Measurement Error Models*. New York: John Wiley & Sons.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Gill, P. E., Murray, W., Saunders, M. A., and Wright, M. H. (1984). "Procedures for Optimization Problems with a Mixture of Bounds and General Linear Constraints." *ACM Transactions on Mathematical Software* 10:282–298.

Graham, J. W. (2003). "Adding Missing-Data-Relevant Variables to FIML-Based Structural Equation Models." *Structural Equation Modeling* 10:80–100.

Guttman, L. (1957). "Empirical Verification of the Radex Structure of Mental Abilities and Personality Traits." *Educational and Psychological Measurement* 17:391–407.

Hägglund, G. (1982). "Factor Analysis by Instrumental Variable Methods." *Psychometrika* 47:209–222.

Haller, A. O., and Butterworth, C. E. (1960). "Peer Influences on Levels of Occupational and Educational Aspiration." *Social Forces* 38:289–295.

Harman, H. H. (1976). *Modern Factor Analysis*. 3rd ed. Chicago: University of Chicago Press.

Hoelter, J. W. (1983). "The Analysis of Covariance Structures: Goodness-of-Fit Indices." *Sociological Methods and Research* 11:325–344.

Holzinger, K. J., and Swineford, F. (1937). "The Bi-Factor Method." *Psychometrika* 2:41–54.

Huber, P. J. (1981). *Robust Statistics*. New York: John Wiley & Sons.

Huynh, H., and Feldt, L. S. (1970). "Conditions Under Which Mean Square Ratios in Repeated Measurements Designs Have Exact F-Distributions." *Journal of the American Statistical Association* 65:1582–1589.

James, L. R., Mulaik, S. A., and Brett, J. M. (1982). *Causal Analysis*. Beverly Hills, CA: Sage Publications.

Jennrich, R. I. (1973). "Standard Errors for Obliquely Rotated Factor Loadings." *Psychometrika* 38:593–604.

Jennrich, R. I. (1987). "Tableau Algorithms for Factor Analysis by Instrumental Variable Methods." *Psychometrika* 52:469–476.

Jöreskog, K. G. (1973). "A General Method for Estimating a Linear Structural Equation System." In *Structural Equation Models in the Social Sciences*, edited by A. S. Goldberger and O. D. Duncan, 85–112. New York: Academic Press.

Jöreskog, K. G. (1978). "Structural Analysis of Covariance and Correlation Matrices." *Psychometrika* 43:443–477.

Jöreskog, K. G., and Sörbom, D. (1985). *LISREL VI: Analysis of Linear Structural Relationships by Maximum Likelihood, Instrumental Variables, and Least Squares*. Uppsala, Sweden: University of Uppsala.

Jöreskog, K. G., and Sörbom, D. (1988). *LISREL 7: A Guide to the Program and Applications*. Chicago: SPSS.

Keesling, J. W. (1972). "Maximum Likelihood Approaches to Causal Analysis." Ph.D. diss., University of Chicago.

Kenward, M. G., and Molenberghs, G. (1998). "Likelihood Based Frequentist Inference When Data Are Missing at Random." *Statistical Science* 236–247.

Kmenta, J. (1971). *Elements of Econometrics*. New York: Macmillan.

Lawley, D. N., and Maxwell, A. E. (1971). *Factor Analysis as a Statistical Method*. New York: Macmillan.

Lee, S. Y. (1985). "On Testing Functional Constraints in Structural Equation Models." *Biometrika* 72:125–131.

Loehlin, J. C. (1987). *Latent Variable Models: An Introduction to Factor, Path, and Structural Analysis*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Loehlin, J. C. (2004). *Latent Variable Models: An Introduction to Factor, Path, and Structural Analysis*. 4th ed. Mahwah, NJ: Lawrence Erlbaum Associates.

Long, J. S. (1983). *Covariance Structure Models: An Introduction to LISREL*. Beverly Hills, CA: Sage Publications.

MacCallum, R. C. (1986). "Specification Searches in Covariance Structure Modeling." *Psychological Bulletin* 100:107–120.

MacCallum, R. C., Roznowski, M., and Necowitz, L. B. (1992). "Model Modification in Covariance Structure Analysis: The Problem of Capitalization on Chance." *Psychological Bulletin* 111:490–504.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Mauchly, J. W. (1940). "Significance Test for Sphericity of a Normal N-Variate Distribution." *Annals of Mathematical Statistics* 11:204–209.

McArdle, J. J. (1980). "Causal Modeling Applied to Psychonomic Systems Simulation." *Behavior Research Methods and Instrumentation* 12:193–209.

McArdle, J. J. (1988). "Dynamic but Structural Equation Modeling of Repeated Measures Data." In *The Handbook of Multivariate Experimental Psychology,* vol. 2, edited by J. R. Nesselroade and R. B. Cattell, 561–614. New York: Plenum.

McArdle, J. J., and McDonald, R. P. (1984). "Some Algebraic Properties of the Reticular Action Model for Moment Structures." *British Journal of Mathematical and Statistical Psychology* 37:234–251.

McDonald, R. P. (1978). "A Simple Comprehensive Model for the Analysis of Covariance Structures." *British Journal of Mathematical and Statistical Psychology* 31:59–72.

McDonald, R. P. (1980). "A Simple Comprehensive Model for the Analysis of Covariance Structures: Some Remarks on Applications." *British Journal of Mathematical and Statistical Psychology* 33:161–183.

McDonald, R. P. (1985). *Factor Analysis and Related Methods*. Hillsdale, NJ: Lawrence Erlbaum Associates.

McDonald, R. P., and Hartmann, W. M. (1992). "A Procedure for Obtaining Initial Values of Parameters in the RAM Model." *Multivariate Behavioral Research* 27:57–76.

McDonald, R. P., and Marsh, H. W. (1988). "Choosing a Multivariate Model: Noncentrality and Goodness of Fit." Distributed paper.

Moré, J. J. (1978). "The Levenberg-Marquardt Algorithm: Implementation and Theory." In *Lecture Notes in Mathematics*, vol. 30, edited by G. A. Watson, 105–116. Berlin: Springer-Verlag.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Morrison, D. F. (1990). *Multivariate Statistical Methods*. 3rd ed. New York: McGraw-Hill.

Mulaik, S. A., James, L. R., Van Alstine, J., Bennett, N., Lind, S., and Stilwell, C. D. (1989). "Evaluation of Goodness-of-Fit Indices for Structural Equation Models." *Psychological Bulletin* 105:430–445.

Mulaik, S. A., and Quartetti, D. A. (1997). "First Order or Higher Order General Factor." *Structural Equation Modeling* 4:193–211.

Powell, M. J. D. (1977). "Restart Procedures for the Conjugate Gradient Method." *Mathematical Programming* 12:241–254.

Powell, M. J. D. (1978a). "Algorithms for Nonlinear Constraints That Use Lagrangian Functions." *Mathematical Programming* 14:224–248.

Powell, M. J. D. (1978b). "A Fast Algorithm for Nonlinearly Constrained Optimization Calculations." In *Lecture Notes in Mathematics*, vol. 630, edited by G. A. Watson, 144–175. Berlin: Springer-Verlag.

Powell, M. J. D. (1982a). "Extensions to Subroutine VF02AD." In *Systems Modeling and Optimization, Lecture Notes in Control and Information Sciences*, vol. 38, edited by R. F. Drenick and F. Kozin, 529–538. Berlin: Springer-Verlag.

Powell, M. J. D. (1982b). *VMCWD: A Fortran Subroutine for Constrained Optimization*. Technical Report DAMTP 1982/NA4, Department of Applied Mathematics and Theoretical Physics, University of Cambridge.

Rubin, D. B. (1976). "Inference and Missing Data." *Biometrika* 63:581–592.

Satorra, A., and Bentler, P. M. (1994). "Corrections to Test Statistics and Standard Errors in Covariance Structure Analysis." In *Latent Variables Analysis: Applications for Developmental Research*, edited by A. von Eye and C. C. Clogg, 399–419. Thousand Oaks, CA: Sage Publications.

Schmid, J., and Leiman, J. M. (1957). "The Development of Hierarchical Factor Solutions." *Psychometrika* 22:53–61.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Sclove, S. L. (1987). "Application of Model-Selection Criteria to Some Problems in Multivariate Analysis." *Psychometrika* 52:333–343.

Steiger, J. H. (1998). "A Note on Multiple Sample Extensions of the RMSEA Fit Index." *Structural Equation Modeling* 5:411–419.

Steiger, J. H., and Lind, J. C. (1980). "Statistically Based Tests for the Number of Common Factors." Paper presented at the annual meeting of the Psychometric Society, Iowa City, IA.

Swaminathan, H. (1974). *A General Factor Model for the Description of Change*. Technical Report LR-74-9, Laboratory of Psychometric and Evaluative Research, University of Massachusetts.

Tucker, L. R., and Lewis, C. (1973). "A Reliability Coefficient for Maximum Likelihood Factor Analysis." *Psychometrika* 38:1–10.

Wheaton, B., Muthén, B. O., Alwin, D. F., and Summers, G. F. (1977). "Assessing Reliability and Stability in Panel Models." In *Sociological Methodology*, edited by D. R. Heise, 84–136. San Francisco: Jossey-Bass.

Wiley, D. E. (1973). "The Identification Problem for Structural Equation Models with Unmeasured Variables." In *Structural Equation Models in the Social Sciences*, edited by A. S. Goldberger and O. D. Duncan, 69–83. New York: Academic Press.

Wilson, E. B., and Hilferty, M. M. (1931). "The Distribution of Chi-Square." *Proceedings of the National Academy of Sciences* 17:684–688.

Yuan, K.-H., and Hayashi, K. (2010). "Fitting Data to Model: Structural Equation Modeling Diagnosis Using Two Scatter Plots." *Psychological Methods* 15:335–351.

Yuan, K.-H., and Zhong, X. (2008). "Outliers, Leverage Observations, and Influential Cases in Factor Analysis: Using Robust Procedures to Minimize Their Effect." *Sociological Methodology* 38:329–368.

Yung, Y.-F. (2014). "Creating Path Diagrams That Impress: A New Graphical Capability of the CALIS Procedure." SAS Institute Inc., Cary, NC. http://support.sas.com/rnd/app/stat/papers/2014/yungpd2014.pdf.

Yung, Y. F., Browne, M., and Zhang, W. (2015). "Fitting Direct Covariance Structures by the MSTRUCT Modeling Language of the CALIS Procedure." *British Journal of Mathematical and Statistical Psychology* 68:178–193.

Yung, Y.-F., Thissen, D., and McLeod, L. D. (1999). "On the Relationship between the Higher-Order Factor Model and the Hierarchical Factor Model." *Psychometrika* 64:113–128.

Yung, Y.-F., and Yuan, K.-H. (2013). "Bartlett Factor Scores: General Formulas and Applications to Structural Equation Models." In *New Developments in Quantitative Psychology: Presentations from the Seventy-Seventh Annual Psychometric Society Meeting*, edited by R. E. Millsap, L. A. van der Ark, D. M. Bolt, and C. M. Woods, 385–401. New York: Springer.

Yung, Y.-F., and Zhang, W. (2011). "Making Use of Incomplete Observations in the Analysis of Structural Equation Models: The CALIS Procedure's Full Information Maximum Likelihood Method in SAS/STAT 9.3." In *Proceedings of the SAS Global Forum 2011 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings11/333-2011.pdf.

Zhang, W., and Yung, Y.-F. (2011). "A Tutorial on Structural Equation Modeling with Incomplete Observations: Multiple Imputation and FIML Methods Using SAS." SAS Institute Inc., Cary, NC. http://support.sas.com/rnd/app/stat/papers/imps2011_FIML.pdf.

Chapter 34
The CANCORR Procedure

Contents

Overview: CANCORR Procedure **2168**
- Background 2168

Getting Started: CANCORR Procedure **2169**

Syntax: CANCORR Procedure **2174**
- PROC CANCORR Statement 2174
- BY Statement 2180
- FREQ Statement 2180
- PARTIAL Statement 2181
- VAR Statement 2181
- WEIGHT Statement 2181
- WITH Statement 2181

Details: CANCORR Procedure **2182**
- Missing Values 2182
- Formulas 2182
- Output Data Sets 2183
- Computational Resources 2185
- Displayed Output 2186
- ODS Table Names 2188

Example: CANCORR Procedure **2190**
- Example 34.1: Canonical Correlation Analysis of Fitness Club Data 2190

References **2195**

Overview: CANCORR Procedure

The CANCORR procedure performs canonical correlation, partial canonical correlation, and canonical redundancy analysis.

Canonical correlation is a generalization of multiple correlation for analyzing the relationship between two sets of variables. In multiple correlation, you examine the relationship between a linear combination of a set of explanatory variables, **X**, and a *single* response variable, **Y**. In canonical correlation, you examine the relationship between linear combinations of the set of **X** variables and linear combinations of a *set* of **Y** variables. These linear combinations are called *canonical variables* or *canonical variates*. Either set of variables can be considered explanatory or response variables, since the statistical model is symmetric in the two sets of variables. Simple and multiple correlation are special cases of canonical correlation in which one or both sets contain a single variable.

The CANCORR procedure tests a series of hypotheses that each canonical correlation and all smaller canonical correlations are zero in the population. PROC CANCORR uses an F approximation (Rao 1973; Kshirsagar 1972) that gives better small sample results than the usual χ^2 approximation. At least one of the two sets of variables should have an approximate multivariate normal distribution in order for the probability levels to be valid.

Both standardized and unstandardized canonical coefficients are computed, as well as the four *canonical structure* matrices showing correlations between the two sets of canonical variables and the two sets of original variables. A canonical redundancy analysis (Stewart and Love 1968; Cooley and Lohnes 1971) can also be done. PROC CANCORR provides multiple regression analysis options to aid in interpreting the canonical correlation analysis. You can examine the linear regression of each variable on the opposite set of variables.

PROC CANCORR can produce a data set containing the scores of each observation on each canonical variable, and you can use the PRINT procedure to list these values. A plot of each canonical variable against its counterpart in the other group is often useful, and you can use PROC SGPLOT with the output data set to produce these plots. A second output data set contains the canonical correlations, coefficients, and most other statistics computed by the procedure.

Background

Canonical correlation was developed by Hotelling (1935, 1936).

The application of canonical correlation is discussed by Cooley and Lohnes (1971); Tatsuoka (1971); Mardia, Kent, and Bibby (1979). One of the best theoretical treatments is given by Kshirsagar (1972).

Given a set of p **X** variables and q **Y** variables, the CANCORR procedure finds the linear combinations

$$w_1 = a_{11}x_1 + a_{21}x_2 + \cdots + a_{p1}x_p$$
$$v_1 = b_{11}y_1 + b_{21}y_2 + \cdots + b_{q1}y_q$$

such that the two canonical variables, w_1 and v_1, have the largest possible correlation. This maximized correlation between the two canonical variables is the first canonical correlation. The coefficients of the linear

combinations are canonical coefficients or canonical weights. It is customary to normalize the canonical coefficients so that each canonical variable has a variance of 1.

PROC CANCORR continues by finding a second set of canonical variables, uncorrelated with the first pair, that produces the second-highest correlation coefficient. That is, the second pair of canonical variables is

$$w_2 = a_{12}x_1 + a_{22}x_2 + \cdots + a_{p2}x_p$$
$$v_2 = b_{12}y_1 + b_{22}y_2 + \cdots + b_{q2}y_q$$

such that w_2 is uncorrelated with w_1 and v_1, v_2 is uncorrelated with w_1 and v_1, and w_2 and v_2 have the largest possible correlation subject to these constraints. The process of constructing canonical variables continues until the number of pairs of canonical variables is $\min(p, q)$, the number of variables in the smaller group.

Each canonical variable is uncorrelated with all the other canonical variables of either set except for the one corresponding canonical variable in the opposite set. The canonical coefficients are not generally orthogonal, however, so the canonical variables do not represent jointly perpendicular directions through the space of the original variables.

The first canonical correlation is at least as large as the multiple correlation between any variable and the opposite set of variables. It is possible for the first canonical correlation to be very large while all the multiple correlations for predicting one of the original variables from the opposite set of canonical variables are small. Canonical redundancy analysis (Stewart and Love 1968; Cooley and Lohnes 1971; Van den Wollenberg 1977) examines how well the original variables can be predicted from the canonical variables.

PROC CANCORR can also perform partial canonical correlation, which is a multivariate generalization of ordinary partial correlation (Cooley and Lohnes 1971; Timm 1975). Most commonly used parametric statistical methods, ranging from *t* tests to multivariate analysis of covariance, are special cases of partial canonical correlation.

Getting Started: CANCORR Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/cancogs.sas`.)

The following example demonstrates how you can use the CANCORR procedure to calculate and test canonical correlations between two sets of variables.

Suppose you want to determine the degree of correspondence between a set of job characteristics and measures of employee satisfaction. Using a survey instrument for employees, you calculate three measures of job satisfaction. With another instrument designed for supervisors, you calculate the corresponding job characteristics profile.

Your three variables associated with job satisfaction are as follows:

- career track satisfaction: employee satisfaction with career direction and the possibility of future advancement, expressed as a percent
- management and supervisor satisfaction: employee satisfaction with supervisor's communication and management style, expressed as a percent

- financial satisfaction: employee satisfaction with salary and other benefits, using a scale measurement from 1 to 10 (1=unsatisfied, 10=satisfied)

The three variables associated with job characteristics are as follows:

- task variety: degree of variety involved in tasks, expressed as a percent
- feedback: degree of feedback required in job tasks, expressed as a percent
- autonomy: degree of autonomy required in job tasks, expressed as a percent

The following statements create the SAS data set Jobs and request a canonical correlation analysis:

```
data Jobs;
   input Career Supervisor Finance Variety Feedback Autonomy;
   label Career    ='Career Satisfaction' Variety ='Task Variety'
         Supervisor='Supervisor Satisfaction' Feedback='Amount of Feedback'
         Finance   ='Financial Satisfaction' Autonomy='Degree of Autonomy';
   datalines;
72  26  9          10  11  70
63  76  7          85  22  93
96  31  7          83  63  73
96  98  6          82  75  97
84  94  6          36  77  97
66  10  5          28  24  75
31  40  9          64  23  75
45  14  2          19  15  50
42  18  6          33  13  70
79  74  4          23  14  90
39  12  2          37  13  70
54  35  3          23  74  53
60  75  5          45  58  83
63  45  5          22  67  53
;

proc cancorr data=Jobs
             vprefix=Satisfaction wprefix=Characteristics
             vname='Satisfaction Areas' wname='Job Characteristics';
   var  Career Supervisor Finance;
   with Variety Feedback Autonomy;
run;
```

The DATA= option in the PROC CANCORR statement specifies Jobs as the SAS data set to be analyzed. The VPREFIX and WPREFIX options specify the prefixes for naming the canonical variables from the VAR statement and the WITH statement, respectively. The VNAME option specifies 'Satisfaction Areas' to refer to the set of variables from the VAR statement. Similarly, the WNAME option specifies 'Job Characteristics' to refer to the set of variables from the WITH statement.

The VAR statement defines the first of the two sets of variables to be analyzed as Career, Supervisor, and Finance. The WITH statement defines the second set of variables to be Variety, Feedback, and Autonomy. The results of this analysis are displayed in Figure 34.1 to Figure 34.4.

Figure 34.1 displays the canonical correlation, adjusted canonical correlation, approximate standard error, and squared canonical correlation for each pair of canonical variables. The first canonical correlation (the correlation between the first pair of canonical variables) is 0.9194. This value represents the highest possible

correlation between any linear combination of the job satisfaction variables and any linear combination of the job characteristics variables.

Figure 34.1 also lists the likelihood ratio and associated statistics for testing the hypothesis that the canonical correlations in the current row and all that follow are zero.

Figure 34.1 Canonical Correlations, Eigenvalues, and Likelihood Tests

The CANCORR Procedure

Canonical Correlation Analysis

	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq) Eigenvalue	Difference	Proportion	Cumulative
1	0.919412	0.898444	0.042901	0.845318	5.4649	5.2524	0.9604	0.9604
2	0.418649	0.276633	0.228740	0.175267	0.2125	0.1995	0.0373	0.9977
3	0.113366	.	0.273786	0.012852	0.0130		0.0023	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero		Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
	1	0.12593148	2.93	9	19.621	0.0223
	2	0.81413359	0.49	4	18	0.7450
	3	0.98714819	0.13	1	10	0.7257

The first approximate F value of 2.93 corresponds to the test that all three canonical correlations are zero. Since the p-value is small (0.0223), you would reject the null hypothesis at the 0.05 level. The second approximate F value of 0.49 corresponds to the test that both the second and the third canonical correlations are zero. Since the p-value is large (0.7450), you would fail to reject the hypothesis and conclude that only the first canonical correlation is significant.

Figure 34.2 lists several multivariate statistics and tests that use approximations based on the F distribution for the null hypothesis that all canonical correlations are zero. Alternatively, you can specify MSTAT=EXACT to compute exact p-values for three of the four tests (Wilks' Lambda, the Hotelling-Lawley Trace, and Roy's greatest root) and an improved F approximation for the fourth (Pillai's Trace). These statistics are described in the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures."

Figure 34.2 Multivariate Statistics and F Approximations

Multivariate Statistics and F Approximations

S=3 M=-0.5 N=3

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.12593148	2.93	9	19.621	0.0223
Pillai's Trace	1.03343732	1.75	9	30	0.1204
Hotelling-Lawley Trace	5.69042615	4.76	9	9.8113	0.0119
Roy's Greatest Root	5.46489324	18.22	3	10	0.0002

NOTE: F Statistic for Roy's Greatest Root is an upper bound.

The small p-values for these tests (< 0.05), except for Pillai's trace, suggest rejecting the null hypothesis that

all canonical correlations are zero in the population, confirming the results of the preceding likelihood ratio test (Figure 34.1). With only one of the tests resulting in a p-value larger than 0.05, you can assume that the first canonical correlation is significant. The next step is to interpret or identify the two canonical variables corresponding to this significant correlation.

Even though canonical variables are artificial, they can often be "identified" in terms of the original variables. This is done primarily by inspecting the standardized coefficients of the canonical variables and the correlations between the canonical variables and their original variables. Since only the first canonical correlation is significant, only the first pair of canonical variables (Satisfaction1 and Characteristics1) need to be identified.

PROC CANCORR calculates and displays the raw canonical coefficients for the job satisfaction variables and the job characteristic variables. However, since the original variables do not necessarily have equal variance and are not measured in the same units, the raw coefficients must be standardized to allow interpretation. The coefficients are standardized by multiplying the raw coefficients with the standard deviation of the associated variable.

The standardized canonical coefficients in Figure 34.3 show that the first canonical variable for the Satisfaction group is a weighted sum of the variables Supervisor (0.7854) and Career (0.3028), with the emphasis on Supervisor. The coefficient for the variable Finance is near 0. Thus, a person satisfied with his or her supervisor and with a large degree of career satisfaction would score high on the canonical variable Satisfaction1.

Figure 34.3 Standardized Canonical Coefficients from the CANCORR Procedure

Standardized Canonical Coefficients for the Satisfaction Areas				
		Satisfaction1	**Satisfaction2**	**Satisfaction3**
Career	Career Satisfaction	0.3028	-0.5416	1.0408
Supervisor	Supervisor Satisfaction	0.7854	0.1305	-0.9085
Finance	Financial Satisfaction	0.0538	0.9754	0.3329

Standardized Canonical Coefficients for the Job Characteristics				
		Characteristics1	**Characteristics2**	**Characteristics3**
Variety	Task Variety	-0.1108	0.8095	0.9071
Feedback	Amount of Feedback	0.5520	-0.7722	0.4194
Autonomy	Degree of Autonomy	0.8403	0.1020	-0.8297

The coefficients for the job characteristics variables show that degree of autonomy (Autonomy) and amount of feedback (Feedback) contribute heavily to the Characteristics1 canonical variable (0.8403 and 0.5520, respectively).

Figure 34.4 shows the table of correlations between the canonical variables and the original variables.

Figure 34.4 Canonical Structure Correlations from the CANCORR Procedure

The CANCORR Procedure

Canonical Structure

Correlations Between the Satisfaction Areas and Their Canonical Variables				
		Satisfaction1	**Satisfaction2**	**Satisfaction3**
Career	Career Satisfaction	0.7499	-0.2503	0.6123
Supervisor	Supervisor Satisfaction	0.9644	0.0362	-0.2618
Finance	Financial Satisfaction	0.2873	0.8814	0.3750

Correlations Between the Job Characteristics and Their Canonical Variables				
		Characteristics1	**Characteristics2**	**Characteristics3**
Variety	Task Variety	0.4863	0.6592	0.5736
Feedback	Amount of Feedback	0.6216	-0.5452	0.5625
Autonomy	Degree of Autonomy	0.8459	0.4451	-0.2938

Correlations Between the Satisfaction Areas and the Canonical Variables of the Job Characteristics				
		Characteristics1	**Characteristics2**	**Characteristics3**
Career	Career Satisfaction	0.6895	-0.1048	0.0694
Supervisor	Supervisor Satisfaction	0.8867	0.0152	-0.0297
Finance	Financial Satisfaction	0.2642	0.3690	0.0425

Correlations Between the Job Characteristics and the Canonical Variables of the Satisfaction Areas				
		Satisfaction1	**Satisfaction2**	**Satisfaction3**
Variety	Task Variety	0.4471	0.2760	0.0650
Feedback	Amount of Feedback	0.5715	-0.2283	0.0638
Autonomy	Degree of Autonomy	0.7777	0.1863	-0.0333

Although these univariate correlations must be interpreted with caution since they do not indicate how the original variables contribute *jointly* to the canonical analysis, they are often useful in the identification of the canonical variables.

Figure 34.4 shows that the supervisor satisfaction variable Supervisor is strongly associated with the Satisfaction1 canonical variable, with a correlation of 0.9644. Slightly less influential is the variable Career, which has a correlation with the canonical variable of 0.7499. Thus, the canonical variable Satisfaction1 seems to represent satisfaction with supervisor and career track.

The correlations for the job characteristics variables show that the canonical variable Characteristics1 seems to represent all three measured variables, with degree of autonomy variable (Autonomy) being the most influential (0.8459).

Hence, you can interpret these results to mean that job characteristics and job satisfaction are related—jobs that possess a high degree of autonomy and level of feedback are associated with workers who are more satisfied with their supervisor and their career. While financial satisfaction is a factor in job satisfaction, it is not as important as the other measured satisfaction-related variables.

Syntax: CANCORR Procedure

The following statements are available in the CANCORR procedure:

```
PROC CANCORR <options> ;
   WITH variables ;
   BY variables ;
   FREQ variable ;
   PARTIAL variables ;
   VAR variables ;
   WEIGHT variable ;
```

The PROC CANCORR statement and the WITH statement are required. The rest of this section provides detailed syntax information for each of the preceding statements, beginning with the PROC CANCORR statement. The remaining statements are covered in alphabetical order.

PROC CANCORR Statement

```
PROC CANCORR <options> ;
```

The PROC CANCORR statement invokes the CANCORR procedure. Optionally, it also identifies input and output data sets, specifies the analyses performed, and controls displayed output. Table 34.1 summarizes the options available in the PROC CANCORR statement.

Table 34.1 Summary of PROC CANCORR Statement Options

Option	Description
Specify Computational Details	
EDF=	Specifies error degrees of freedom if input observations are regression residuals
MSTAT=	Specifies the method of evaluating the multivariate test statistics
NOINT	Omits intercept from canonical correlation and regression models
RDF=	Specifies regression degrees of freedom if input observations are regression residuals
SINGULAR=	Specifies the singularity criterion
Specify Input and Output Data Sets	
DATA=	Specifies input data set name
OUT=	Specifies output data set name
OUTSTAT=	Specifies output data set name containing various statistics
Specify Labeling Options	
PARPREFIX=	Specifies a prefix for naming residual variables
VNAME=	Specifies a name to refer to VAR statement variables
VPREFIX=	Specifies a prefix for naming VAR statement canonical variables
WNAME=	Specifies a name to refer to WITH statement variables
WPREFIX=	Specifies a prefix for naming WITH statement canonical variables

Table 34.1 *continued*

Option	Description
Control Amount of Output	
ALL	Produces simple statistics, input variable correlations, and canonical redundancy analysis
CORR	Produces input variable correlations
NCAN=	Specifies number of canonical variables for which full output is desired
NOPRINT	Suppresses all displayed output
REDUNDANCY	Produces canonical redundancy analysis
SHORT	Suppresses default output from canonical analysis
SIMPLE	Produces means and standard deviations
Request Regression Analyses	
VDEP	Requests multiple regression analyses with the VAR variables as dependents and the WITH variables as regressors
VREG	Requests multiple regression analyses with the VAR variables as regressors and the WITH variables as dependents
WDEP	Same as VREG
WREG	Same as VDEP
Specify Regression Statistics	
ALL	Produces all regression statistics and includes these statistics in the OUTSTAT= data set
B	Produces raw regression coefficients
CLB	Produces 95% confidence interval limits for the regression coefficients
CORRB	Produces correlations among regression coefficients
INT	Requests statistics for the intercept when you specify the B, CLB, SEB, T, or PROBT option
PCORR	Displays partial correlations between regressors and dependents
PROBT	Displays probability levels for *t* statistics
SEB	Displays standard errors of regression coefficients
SMC	Displays squared multiple correlations and *F* tests
SPCORR	Displays semipartial correlations between regressors and dependents
SQPCORR	Displays squared partial correlations between regressors and dependents
SQSPCORR	Displays squared semipartial correlations between regressors and dependents
STB	Displays standardized regression coefficients
T	Displays *t* statistics for regression coefficients

Following are explanations of the options that can be used in the PROC CANCORR statement (in alphabetic order).

ALL

displays simple statistics, correlations among the input variables, the confidence limits for the regression coefficients, and the canonical redundancy analysis. If you specify the VDEP or WDEP option, the ALL option displays all related regression statistics (unless the NOPRINT option is specified) and includes these statistics in the OUTSTAT= data set.

B

produces raw regression coefficients from the regression analyses.

CLB

produces the 95% confidence limits for the regression coefficients from the regression analyses.

CORR
C

produces correlations among the original variables. If you include a PARTIAL statement, the CORR option produces a correlation matrix for all variables in the analysis, the regression statistics (R square, RMSE), the standardized regression coefficients for both the VAR and WITH variables as predicted from the PARTIAL statement variables, and partial correlation matrices.

CORRB

produces correlations among the regression coefficient estimates.

DATA=*SAS-data-set*

names the SAS data set to be analyzed by PROC CANCORR. It can be an ordinary SAS data set or a TYPE=CORR, COV, FACTOR, SSCP, UCORR, or UCOV data set. By default, the procedure uses the most recently created SAS data set.

EDF=*error-df*

specifies the error degrees of freedom if the input observations are residuals from a regression analysis. The effective number of observations is the EDF= value plus one. If you have 100 observations, then specifying EDF=99 has the same effect as omitting the EDF= option.

INT

requests that statistics for the intercept be included when B, CLB, SEB, T, or PROBT is specified for the regression analyses.

MSTAT=FAPPROX | EXACT

specifies the method of evaluating the multivariate test statistics. The default is MSTAT=FAPPROX, which specifies that the multivariate tests are evaluated using the usual approximations based on the F distribution, as discussed in the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures." Alternatively, you can specify MSTAT=EXACT to compute exact p-values for three of the four tests (Wilks' lambda, the Hotelling-Lawley trace, and Roy's greatest root) and an improved F approximation for the fourth (Pillai's trace). While MSTAT=EXACT provides better control of the significance probability for the tests, especially for Roy's greatest root, computations for the exact p-values can be appreciably more demanding, and are in fact infeasible for large problems (many dependent variables). Thus, although MSTAT=EXACT is more accurate for most data, it is not the default method.

NCAN=*number*

specifies the number of canonical variables for which full output is desired. The *number* must be less than or equal to the number of canonical variables in the analysis.

The value of the NCAN= option specifies the number of canonical variables for which canonical coefficients and canonical redundancy statistics are displayed, and the number of variables shown in the canonical structure matrices. The NCAN= option does not affect the number of displayed canonical correlations.

If an OUTSTAT= data set is requested, the NCAN= option controls the number of canonical variables for which statistics are output. If an OUT= data set is requested, the NCAN= option controls the number of canonical variables for which scores are output.

NOINT

omits the intercept from the canonical correlation and regression models. Standard deviations, variances, covariances, and correlations are not corrected for the mean. If you use a TYPE=SSCP data set as input to the CANCORR procedure and list the variable Intercept in the VAR or WITH statement, the procedure runs as if you also specified the NOINT option. If you use NOINT and also create an OUTSTAT= data set, the data set is TYPE=UCORR.

NOPRINT

suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*

creates an output SAS data set to contain all the original data plus scores on the canonical variables. The OUT= option cannot be used when the DATA= data set is TYPE=CORR, COV, FACTOR, SSCP, UCORR, or UCOV. For details about OUT= data sets, see the section "Output Data Sets" on page 2183. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTSTAT=*SAS-data-set*

creates an output SAS data set containing various statistics, including the canonical correlations and coefficients and the multiple regression statistics you request. For details about OUTSTAT= data sets, see the section "Output Data Sets" on page 2183. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PCORR

produces partial correlations between regressors and dependent variables, removing from each dependent variable and regressor the effects of all other regressors.

PROBT

produces probability levels for the t statistics in the regression analyses.

RDF=*regression-df*

specifies the regression degrees of freedom if the input observations are residuals from a regression analysis. The effective number of observations is the actual number minus the RDF= value. The degrees of freedom for the intercept should not be included in the RDF= option.

REDUNDANCY

RED

produces canonical redundancy statistics.

PARPREFIX=*name*

PPREFIX=*name*

specifies a prefix for naming the residual variables in the OUT= data set and the OUTSTAT= data set. By default, the prefix is R_. The number of characters in the prefix plus the maximum length of the variable names should not exceed the current name length defined by the VALIDVARNAME= system option.

SEB

produces standard errors of the regression coefficients.

SHORT

suppresses all default output from the canonical analysis except the tables of canonical correlations and multivariate statistics.

SIMPLE

S

produces means and standard deviations.

SINGULAR=*p*

SING=*p*

specifies the singularity criterion, where $0 < p < 1$. If a variable in the PARTIAL statement has an R square as large as $1 - p$ (where p is the value of the SINGULAR= option) when predicted from the variables listed before it in the statement, the variable is assigned a standardized regression coefficient of 0, and the SAS log generates a linear dependency warning message. By default, SINGULAR=1E–8.

SMC

produces squared multiple correlations and F tests for the regression analyses.

SPCORR

produces semipartial correlations between regressors and dependent variables, removing from each regressor the effects of all other regressors.

SQPCORR

produces squared partial correlations between regressors and dependent variables, removing from each dependent variable and regressor the effects of all other regressors.

SQSPCORR

produces squared semipartial correlations between regressors and dependent variables, removing from each regressor the effects of all other regressors.

STB

produces standardized regression coefficients.

T

produces *t* statistics for the regression coefficients.

VDEP

WREG

requests multiple regression analyses with the VAR variables as dependent variables and the WITH Variables as regressors.

VNAME=*label*

VN=*label*

specifies a character constant to refer to variables from the VAR statement in the output. Enclose the constant in single or double quotes. If you omit the VNAME= option, these variables are referred to as the VAR variables. The number of characters in the label should not exceed the label length defined by the VALIDVARNAME= system option. For more information about the VALIDVARNAME= system option, see *SAS System Options: Reference*.

VPREFIX=*name*

VP=*name*

specifies a prefix for naming canonical variables from the VAR statement. By default, these canonical variables are given the names V1, V2, and so on. If you specify VPREFIX=ABC, the names are ABC1, ABC2, and so on. The number of characters in the prefix plus the number of digits required to designate the variables should not exceed the name length defined by the VALIDVARNAME= system option. For more information about the VALIDVARNAME= system option, see *SAS System Options: Reference*.

WDEP

VREG

requests multiple regression analyses with the WITH variables as dependent variables and the VAR variables as regressors.

WNAME=*label*

WN=*label*

specifies a character constant to refer to variables in the WITH statement in the output. Enclose the constant in single or double quotes. If you omit the WNAME= option, these variables are referred to as the WITH variables. The number of characters in the label should not exceed the label length defined by the VALIDVARNAME= system option. For more information about the VALIDVARNAME= system option, see *SAS System Options: Reference*.

WPREFIX=*name*

WP=*name*

specifies a prefix for naming canonical variables from the WITH statement. By default, these canonical variables are given the names W1, W2, and so on. If you specify WPREFIX=XYZ, the names are XYZ1, XYZ2, and so on. The number of characters in the prefix plus the number of digits required to designate

the variables should not exceed the label length defined by the VALIDVARNAME= system option. For more information about the VALIDVARNAME= system option, see *SAS System Options: Reference*.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC CANCORR to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the CANCORR procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If one variable in your input data set represents the frequency of occurrence for other values in the observation, specify the variable's name in a FREQ statement. PROC CANCORR then treats the data set as if each observation appeared n times, where n is the value of the FREQ variable for the observation. If the value of the FREQ variable is less than one, the observation is not used in the analysis. Only the integer portion of the value is used. The total number of observations is considered to be equal to the sum of the FREQ variable when PROC CANCORR calculates significance probabilities.

PARTIAL Statement

PARTIAL *variables* **;**

You can use the PARTIAL statement to base the canonical analysis on partial correlations. The variables in the PARTIAL statement are partialed out of the VAR and WITH variables. If you request an OUT= or OUTSTAT= data set, the residual variables are named by prefixing the characters R_ by default or the string specified in the RPREFIX= option to the VAR variables.

VAR Statement

VAR *variables* **;**

The VAR statement lists the variables in the first of the two sets of variables to be analyzed. The variables must be numeric. If you omit the VAR statement, all numeric variables not mentioned in other statements make up the first set of variables. If, however, the DATA= data set is TYPE=SSCP, the default set of variables used as VAR variables does not include the variable Intercept.

WEIGHT Statement

WEIGHT *variable* **;**

If you want to compute weighted product-moment correlation coefficients, specify the name of the weighting variable in a WEIGHT statement. The WEIGHT and FREQ statements have a similar effect, except the WEIGHT statement does not alter the degrees of freedom or number of observations. An observation is used in the analysis only if the WEIGHT variable is greater than zero.

WITH Statement

WITH *variables* **;**

The WITH statement lists the variables in the second set of variables to be analyzed. The variables must be numeric. The WITH statement is required.

Details: CANCORR Procedure

Missing Values

If an observation has a missing value for any of the variables in the analysis, that observation is omitted from the analysis.

Formulas

Assume without loss of generality that the two sets of variables, $\mathbf{X}$ with p variables and $\mathbf{Y}$ with q variables, have means of zero. Let n be the number of observations, and let m be $n-1$.

Note that the scales of eigenvectors and canonical coefficients are arbitrary. PROC CANCORR follows the usual procedure of rescaling the canonical coefficients so that each canonical variable has a variance of one.

There are several different sets of formulas that can be used to compute the canonical correlations, ρ_i, $i = 1, \ldots, \min(p, q)$, and unscaled canonical coefficients:

1. Let $\mathbf{S}_{XX} = \mathbf{X}'\mathbf{X}/m$ be the covariance matrix of $\mathbf{X}$, $\mathbf{S}_{YY} = \mathbf{Y}'\mathbf{Y}/m$ be the covariance matrix of $\mathbf{Y}$, and $\mathbf{S}_{XY} = \mathbf{X}'\mathbf{Y}/m$ be the covariance matrix between $\mathbf{X}$ and $\mathbf{Y}$. Then the eigenvalues of $\mathbf{S}_{YY}^{-1}\mathbf{S}_{XY}'\mathbf{S}_{XX}^{-1}\mathbf{S}_{XY}$ are the squared canonical correlations, and the right eigenvectors are raw canonical coefficients for the $\mathbf{Y}$ variables. The eigenvalues of $\mathbf{S}_{XX}^{-1}\mathbf{S}_{XY}\mathbf{S}_{YY}^{-1}\mathbf{S}_{XY}'$ are the squared canonical correlations, and the right eigenvectors are raw canonical coefficients for the $\mathbf{X}$ variables.

2. Let $\mathbf{T} = \mathbf{Y}'\mathbf{Y}$ and $\mathbf{H} = \mathbf{Y}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$. The eigenvalues ξ_i of $\mathbf{T}^{-1}\mathbf{H}$ are the squared canonical correlations, ρ_i^2, and the right eigenvectors are raw canonical coefficients for the $\mathbf{Y}$ variables. Interchange $\mathbf{X}$ and $\mathbf{Y}$ in the preceding formulas, and the eigenvalues remain the same, but the right eigenvectors are raw canonical coefficients for the $\mathbf{X}$ variables.

3. Let $\mathbf{E} = \mathbf{T} - \mathbf{H}$. The eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$ are $\lambda_i = \rho_i^2/(1-\rho_i^2)$. The right eigenvectors of $\mathbf{E}^{-1}\mathbf{H}$ are the same as the right eigenvectors of $\mathbf{T}^{-1}\mathbf{H}$.

4. Canonical correlation can be viewed as a principal component analysis of the predicted values of one set of variables from a regression on the other set of variables, in the metric of the error covariance matrix. For example, regress the $\mathbf{Y}$ variables on the $\mathbf{X}$ variables. Call the predicted values $\mathbf{P} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$ and the residuals $\mathbf{R} = \mathbf{Y} - \mathbf{P} = (\mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}')\mathbf{Y}$. The error covariance matrix is $\mathbf{R}'\mathbf{R}/m$. Choose a transformation $\mathbf{Q}$ that converts the error covariance matrix to an identity—that is, $(\mathbf{RQ})'(\mathbf{RQ}) = \mathbf{Q}'\mathbf{R}'\mathbf{RQ} = m\mathbf{I}$. Apply the same transformation to the predicted values to yield, say, $\mathbf{Z} = \mathbf{PQ}$. Now do a principal component analysis on the covariance matrix of $\mathbf{Z}$, and you get the eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$. Repeat with $\mathbf{X}$ and $\mathbf{Y}$ variables interchanged, and you get the same eigenvalues.

   To show this relationship between canonical correlation and principal components, note that $\mathbf{P}'\mathbf{P} = \mathbf{H}$, $\mathbf{R}'\mathbf{R} = \mathbf{E}$, and $\mathbf{QQ}' = m\mathbf{E}^{-1}$. Let the covariance matrix of $\mathbf{Z}$ be $\mathbf{G}$. Then $\mathbf{G} = \mathbf{Z}'\mathbf{Z}/m = (\mathbf{PQ})'\mathbf{PQ}/m = \mathbf{Q}'\mathbf{P}'\mathbf{PQ}/m = \mathbf{Q}'\mathbf{HQ}/m$. Let $\mathbf{u}$ be an eigenvector of $\mathbf{G}$ and κ be the corresponding eigenvalue. Then by definition, $\mathbf{Gu} = \kappa\mathbf{u}$; hence $\mathbf{Q}'\mathbf{HQu}/m = \kappa\mathbf{u}$. Premultiplying both sides by $\mathbf{Q}$ yields $\mathbf{QQ}'\mathbf{HQu}/m = \kappa\mathbf{Qu}$ and thus $\mathbf{E}^{-1}\mathbf{HQu} = \kappa\mathbf{Qu}$. Hence $\mathbf{Qu}$ is an eigenvector of $\mathbf{E}^{-1}\mathbf{H}$ and κ is also an eigenvalue of $\mathbf{E}^{-1}\mathbf{H}$.

5. If the covariance matrices are replaced by correlation matrices, the preceding formulas yield standardized canonical coefficients instead of raw canonical coefficients.

The formulas for multivariate test statistics are shown in the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures." Formulas for linear regression are provided in other sections of that chapter.

Output Data Sets

OUT= Data Set

The OUT= data set contains all the variables in the original data set plus new variables containing the canonical variable scores. The number of new variables is twice that specified by the NCAN= option. The names of the new variables are formed by concatenating the values given by the VPREFIX= and WPREFIX= options (the defaults are V and W) with the numbers 1, 2, 3, and so on. The new variables have mean 0 and variance equal to 1. An OUT= data set cannot be created if the DATA= data set is TYPE=CORR, COV, FACTOR, SSCP, UCORR, or UCOV.

If you use a PARTIAL statement, the OUT= data set also contains the residuals from predicting the VAR variables from the PARTIAL variables. The names of the residual variables are formed by concatenating the values given by the PARPREFIX= option (the default is R_) with the numbers 1, 2, 3, and so on.

OUTSTAT= Data Set

The OUTSTAT= data set is similar to the TYPE=CORR or TYPE=UCORR data set produced by the CORR procedure, but it contains several results in addition to those produced by PROC CORR.

The new data set contains the following variables:

- the BY variables, if any
- two new character variables, _TYPE_ and _NAME_
- Intercept, if the INT option is used
- the variables analyzed (those in the VAR statement and the WITH statement)

Each observation in the new data set contains some type of statistic as indicated by the _TYPE_ variable. The values of the _TYPE_ variable are as follows.

TYPE	**Contents**
MEAN	means
STD	standard deviations
USTD	uncorrected standard deviations. When you specify the NOINT option in the PROC CANCORR statement, the OUTSTAT= data set contains standard deviations not corrected for the mean (_TYPE_='USTD').
N	number of observations on which the analysis is based. This value is the same for each variable.

SUMWGT	sum of the weights if a WEIGHT statement is used. This value is the same for each variable.
CORR	correlations. The _NAME_ variable contains the name of the variable corresponding to each row of the correlation matrix.
UCORR	uncorrected correlation matrix. When you specify the NOINT option in the PROC CANCORR statement, the OUTSTAT= data set contains a matrix of correlations not corrected for the means.
CORRB	correlations among the regression coefficient estimates
STB	standardized regression coefficients. The _NAME_ variable contains the name of the dependent variable.
B	raw regression coefficients
SEB	standard errors of the regression coefficients
LCLB	95% lower confidence limits for the regression coefficients
UCLB	95% upper confidence limits for the regression coefficients
T	*t* statistics for the regression coefficients
PROBT	probability levels for the *t* statistics
SPCORR	semipartial correlations between regressors and dependent variables
SQSPCORR	squared semipartial correlations between regressors and dependent variables
PCORR	partial correlations between regressors and dependent variables
SQPCORR	squared partial correlations between regressors and dependent variables
RSQUARED	R squares for the multiple regression analyses
ADJRSQ	adjusted R squares
LCLRSQ	approximate 95% lower confidence limits for the R squares
UCLRSQ	approximate 95% upper confidence limits for the R squares
F	*F* statistics for the multiple regression analyses
PROBF	probability levels for the *F* statistics
CANCORR	canonical correlations
SCORE	standardized canonical coefficients. The _NAME_ variable contains the name of the canonical variable. To obtain the canonical variable scores, these coefficients should be multiplied by the standardized data, using means obtained from the observation with _TYPE_='MEAN' and standard deviations obtained from the observation with _TYPE_='STD'.
RAWSCORE	raw canonical coefficients. To obtain the canonical variable scores, these coefficients should be multiplied by the raw data centered by means obtained from the observation with _TYPE_='MEAN'.
USCORE	scoring coefficients to be applied without subtracting the mean from the raw variables. These are standardized canonical coefficients computed under a NOINT model. To obtain the canonical variable scores, these coefficients should be multiplied by the data that are standardized by the uncorrected standard deviations obtained from the observation with _TYPE_='USTD'.

STRUCTUR canonical structure.

Computational Resources

Notation

Let

$$
\begin{aligned}
n &= \text{number of observations} \\
v &= \text{number of variables} \\
w &= \text{number of WITH variables} \\
p &= \max(v, w) \\
q &= \min(v, w) \\
b &= v + w \\
t &= \text{total number of variables (VAR, WITH, and PARTIAL)}
\end{aligned}
$$

Time Requirements

The time required to compute the correlation matrix is roughly proportional to

$$n(p+q)^2$$

The time required for the canonical analysis is roughly proportional to

$$\frac{1}{6}p^3 + p^2q + \frac{3}{2}pq^2 + 5q^3$$

but the coefficient for q^3 varies depending on the number of QR iterations in the singular value decomposition.

Memory Requirements

The minimum memory required is approximately

$$4(v^2 + w^2 + t^2)$$

bytes. Additional memory is required if you request the VDEP or WDEP option.

Displayed Output

If the SIMPLE option is specified, PROC CANCORR produces means and standard deviations for each input variable. If the CORR option is specified, PROC CANCORR produces correlations among the input variables. Unless the NOPRINT option is specified, PROC CANCORR displays a table of canonical correlations containing the following:

- Canonical Correlations. These are always nonnegative.
- Adjusted Canonical Correlations (Lawley 1959), which are asymptotically less biased than the raw correlations and can be negative. The adjusted canonical correlations might not be computable, and they are displayed as missing values if two canonical correlations are nearly equal or if some are close to zero. A missing value is also displayed if an adjusted canonical correlation is larger than a previous adjusted canonical correlation.
- Approx Standard Errors, which are the approximate standard errors of the canonical correlations
- Squared Canonical Correlations
- Eigenvalues of INV(E)*H, which are equal to CanRsq/(1–CanRsq), where CanRsq is the corresponding squared canonical correlation. Also displayed for each eigenvalue is the Difference from the next eigenvalue, the Proportion of the sum of the eigenvalues, and the Cumulative proportion.
- Likelihood Ratio for the hypothesis that the current canonical correlation and all smaller ones are zero in the population. The likelihood ratio for all canonical correlations equals Wilks' lambda.
- Approx F statistic based on Rao's approximation to the distribution of the likelihood ratio (Rao 1973, p. 556; Kshirsagar 1972, p. 326)
- Num DF and Den DF (numerator and denominator degrees of freedom) and Pr $> F$ (probability level) associated with the F statistic

Unless you specify the NOPRINT option, PROC CANCORR produces a table of multivariate statistics for the null hypothesis that all canonical correlations are zero in the population. These statistics, as described in the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures," are as follows:

- Wilks' lambda
- Pillai's trace
- Hotelling-Lawley trace
- Roy's greatest root

For each of the preceding statistics, PROC CANCORR displays the following, depending on the specification of the MSTAT= option.

If you specify MSTAT=FAPPROX (also the default value), the following statistics are displayed:

- an F approximation or upper bound
- Num DF, the numerator degrees of freedom

- Den DF, the denominator degrees of freedom
- $\Pr > F$, the probability level

If you specify MSTAT=EXACT, the following statistic is displayed:

- a t value

Unless you specify the SHORT or NOPRINT option, PROC CANCORR displays the following:

- both Raw (unstandardized) and Standardized Canonical Coefficients normalized to give canonical variables with unit variance. Standardized coefficients can be used to compute canonical variable scores from the standardized (zero mean and unit variance) input variables. Raw coefficients can be used to compute canonical variable scores from the input variables without standardizing them.
- all four Canonical Structure matrices, giving Correlations Between the canonical variables and the original variables

If you specify the REDUNDANCY option, PROC CANCORR displays the following:

- the Canonical Redundancy Analysis (Stewart and Love 1968; Cooley and Lohnes 1971), including Raw (unstandardized) and Standardized Variance and Cumulative Proportion of the Variance of each set of variables Explained by Their Own Canonical Variables and Explained by The Opposite Canonical Variables
- the Squared Multiple Correlations of each variable with the first m canonical variables of the opposite set, where m varies from 1 to the number of canonical correlations

If you specify the VDEP option, PROC CANCORR performs multiple regression analyses with the VAR variables as dependent variables and the WITH variables as regressors. If you specify the WDEP option, PROC CANCORR performs multiple regression analyses with the WITH variables as dependent variables and the VAR variables as regressors. If you specify the VDEP or WDEP option and also specify the ALL option, PROC CANCORR displays the following items. You can also specify individual options to request a subset of the output generated by the ALL option; or you can suppress the output by specifying the NOPRINT option.

SMC	Squared Multiple Correlations and F Tests. For each regression model, identified by its dependent variable name, PROC CANCORR displays the R square, Adjusted R square (Wherry 1931), F Statistic, and $\Pr > F$. Also for each regression model, PROC CANCORR displays an Approximate 95% Confidence Interval for the population R square Helland (1987). These confidence limits are valid only when the regressors are random and when the regressors and dependent variables are approximately distributed according to a multivariate normal distribution. The average R squares for the models considered, unweighted and weighted by variance, are also given.
CORRB	Correlations Among the Regression Coefficient Estimates
STB	Standardized Regression Coefficients

B	Raw Regression Coefficients
SEB	Standard Errors of the Regression Coefficients
CLB	95% confidence limits for the regression coefficients
T	T Statistics for the Regression Coefficients
PROBT	Probability > |T| for the Regression Coefficients
SPCORR	Semipartial Correlations between regressors and dependent variables, Removing from Each Regressor the Effects of All Other Regressors
SQSPCORR	Squared Semipartial Correlations between regressors and dependent variables, Removing from Each Regressor the Effects of All Other Regressors
PCORR	Partial Correlations between regressors and dependent variables, Removing the Effects of All Other Regressors from Both Regressor and Criterion
SQPCORR	Squared Partial Correlations between regressors and dependent variables, Removing the Effects of All Other Regressors from Both Regressor and Criterion

ODS Table Names

PROC CANCORR assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 34.2.

For more information about ODS, see Chapter 23, "Using the Output Delivery System."

All of the tables in Table 34.2 are created with the specification of the PROC CANCORR statement; a few tables need an additional PARTIAL statement.

Table 34.2 ODS Tables Produced by PROC CANCORR

ODS Table Name	Description	Statement and Option
AvgRSquare	Average R squares (weighted and unweighted)	VDEP, WDEP, SMC, or ALL
CanCorr	Canonical correlations	Default
CanStructureVCan	Correlations between the VAR canonical variables and the VAR and WITH variables	Default (if SHORT is not specified)
CanStructureWCan	Correlations between the WITH canonical variables and the WITH and VAR variables	Default (if SHORT is not specified)
ConfidenceLimits	95% confidence limits for the regression coefficients	VDEP, WDEP, CLB, or ALL
Corr	Correlations among the original variables	CORR or ALL
CorrOnPartial	Partial correlations	PARTIAL statement with CORR or ALL

Table 34.2 *continued*

ODS Table Name	Description	Statement and Option
CorrRegCoefEst	Correlations among the regression coefficient estimates	VDEP, WDEP, CORRB, or ALL
MultStat	Multivariate statistics	Default
NObsNVar	Number of observations and variables	SIMPLE or ALL
ParCorr	Partial correlations	VDEP, WDEP, PCORR, or ALL
ProbtRegCoef	Prob > |t| for the regression coefficients	VDEP, WDEP, PROBT, or ALL
RawCanCoefV	Raw canonical coefficients for the VAR variables	Default (if SHORT is not specified)
RawCanCoefW	Raw canonical coefficients for the WITH variables	Default (if SHORT is not specified)
RawRegCoef	Raw regression coefficients	VDEP, WDEP, B, or ALL
Redundancy	Canonical redundancy analysis	REDUNDANCY or ALL
Regression	Squared multiple correlations and *F* tests	VDEP, WDEP, SMC, or ALL
RSquareRMSEOnPartial	R squares and RMSEs on PARTIAL variables	PARTIAL statement with CORR or ALL
SemiParCorr	Semipartial correlations	VDEP, WDEP, SPCORR, or ALL
SimpleStatistics	Simple statistics	SIMPLE or ALL
SqMultCorr	Canonical redundancy analysis: squared multiple correlations	REDUNDANCY or ALL
SqParCorr	Squared partial correlations	VDEP, WDEP, SQPCORR, or ALL
SqSemiParCorr	Squared semipartial correlations	VDEP, WDEP, SQSPCORR, or ALL
StdCanCoefV	Standardized canonical coefficients for the VAR variables	Default (if SHORT is not specified)
StdCanCoefW	Standardized canonical coefficients for the WITH variables	Default (if SHORT is not specified)
StdErrRawRegCoef	Standard errors of the raw regression coefficients	VDEP, WDEP, SEB, or ALL
StdRegCoef	Standardized regression coefficients	VDEP, WDEP, STB, or ALL
StdRegCoefOnPartial	Standardized regression coefficients on PARTIAL variables	PARTIAL statement with CORR or ALL
tValueRegCoef	*t* values for the regression coefficients	VDEP, WDEP, T, or ALL

Example: CANCORR Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 34.1: Canonical Correlation Analysis of Fitness Club Data

(View the complete code for this example (cancoex1.sas) in the example repository.)

Three physiological and three exercise variables are measured on 20 middle-aged men in a fitness club. You can use the CANCORR procedure to determine whether the physiological variables are related in any way to the exercise variables. The following statements create the SAS data set Fit and produce Output 34.1.1 through Output 34.1.5:

```
data Fit;
   input Weight Waist Pulse Chins Situps Jumps;
   datalines;
191  36  50   5  162   60
189  37  52   2  110   60
193  38  58  12  101  101
162  35  62  12  105   37
189  35  46  13  155   58
182  36  56   4  101   42
211  38  56   8  101   38
167  34  60   6  125   40
176  31  74  15  200   40
154  33  56  17  251  250
169  34  50  17  120   38
166  33  52  13  210  115
154  34  64  14  215  105
247  46  50   1   50   50
193  36  46   6   70   31
202  37  62  12  210  120
176  37  54   4   60   25
157  32  52  11  230   80
156  33  54  15  225   73
138  33  68   2  110   43
;

proc cancorr data=Fit all
     vprefix=Physiological vname='Physiological Measurements'
     wprefix=Exercises wname='Exercises';
   var Weight Waist Pulse;
   with Chins Situps Jumps;
   title 'Middle-Aged Men in a Health Fitness Club';
   title2 'Data Courtesy of Dr. A. C. Linnerud, NC State Univ';
run;
```

Output 34.1.1 Correlations among the Original Variables

Middle-Aged Men in a Health Fitness Club
Data Courtesy of Dr. A. C. Linnerud, NC State Univ

The CANCORR Procedure

Correlations Among the Original Variables

Correlations Among the Physiological Measurements	Weight	Waist	Pulse
Weight	1.0000	0.8702	-0.3658
Waist	0.8702	1.0000	-0.3529
Pulse	-0.3658	-0.3529	1.0000

Correlations Among the Exercises	Chins	Situps	Jumps
Chins	1.0000	0.6957	0.4958
Situps	0.6957	1.0000	0.6692
Jumps	0.4958	0.6692	1.0000

Correlations Between the Physiological Measurements and the Exercises	Chins	Situps	Jumps
Weight	-0.3897	-0.4931	-0.2263
Waist	-0.5522	-0.6456	-0.1915
Pulse	0.1506	0.2250	0.0349

Output 34.1.1 displays the correlations among the original variables. The correlations between the physiological and exercise variables are moderate, the largest being –0.6456 between Waist and Situps. There are larger within-set correlations: 0.8702 between Weight and Waist, 0.6957 between Chins and Situps, and 0.6692 between Situps and Jumps.

Output 34.1.2 Canonical Correlations and Multivariate Statistics

Middle-Aged Men in a Health Fitness Club
Data Courtesy of Dr. A. C. Linnerud, NC State Univ

The CANCORR Procedure

Canonical Correlation Analysis

	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq) Eigenvalue	Difference	Proportion	Cumulative
1	0.795608	0.754056	0.084197	0.632992	1.7247	1.6828	0.9734	0.9734
2	0.200556	-.076399	0.220188	0.040223	0.0419	0.0366	0.0237	0.9970
3	0.072570	.	0.228208	0.005266	0.0053		0.0030	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero

	Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
1	0.35039053	2.05	9	34.223	0.0635
2	0.95472266	0.18	4	30	0.9491
3	0.99473355	0.08	1	16	0.7748

Multivariate Statistics and F Approximations

S=3 M=-0.5 N=6

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.35039053	2.05	9	34.223	0.0635
Pillai's Trace	0.67848151	1.56	9	48	0.1551
Hotelling-Lawley Trace	1.77194146	2.64	9	19.053	0.0357
Roy's Greatest Root	1.72473874	9.20	3	16	0.0009

NOTE: F Statistic for Roy's Greatest Root is an upper bound.

As Output 34.1.2 shows, the first canonical correlation is 0.7956, which would appear to be substantially larger than any of the between-set correlations. The probability level for the null hypothesis that all the canonical correlations are zero in the population is only 0.0635, so no firm conclusions can be drawn. The remaining canonical correlations are not worthy of consideration, as can be seen from the probability levels and especially from the negative adjusted canonical correlations.

Because the variables are not measured in the same units, the standardized coefficients rather than the raw coefficients should be interpreted. The correlations given in the canonical structure matrices should also be examined.

Output 34.1.3 Raw and Standardized Canonical Coefficients

Raw Canonical Coefficients for the Physiological Measurements

	Physiological1	Physiological2	Physiological3
Weight	-0.031404688	-0.076319506	-0.007735047
Waist	0.4932416756	0.3687229894	0.1580336471
Pulse	-0.008199315	-0.032051994	0.1457322421

Output 34.1.3 *continued*

Raw Canonical Coefficients for the Exercises			
	Exercises1	**Exercises2**	**Exercises3**
Chins	-0.066113986	-0.071041211	-0.245275347
Situps	-0.016846231	0.0019737454	0.0197676373
Jumps	0.0139715689	0.0207141063	-0.008167472

Standardized Canonical Coefficients for the Physiological Measurements			
	Physiological1	**Physiological2**	**Physiological3**
Weight	-0.7754	-1.8844	-0.1910
Waist	1.5793	1.1806	0.5060
Pulse	-0.0591	-0.2311	1.0508

Standardized Canonical Coefficients for the Exercises			
	Exercises1	**Exercises2**	**Exercises3**
Chins	-0.3495	-0.3755	-1.2966
Situps	-1.0540	0.1235	1.2368
Jumps	0.7164	1.0622	-0.4188

The first canonical variable for the physiological variables, displayed in Output 34.1.3, is a weighted difference of Waist (1.5793) and Weight (–0.7754), with more emphasis on Waist. The coefficient for Pulse is near 0. The correlations between Waist and Weight and the first canonical variable are both positive, 0.9254 for Waist and 0.6206 for Weight. Weight is therefore a suppressor variable, meaning that its coefficient and its correlation have opposite signs.

The first canonical variable for the exercise variables also shows a mixture of signs, subtracting Situps (–1.0540) and Chins (–0.3495) from Jumps (0.7164), with the most weight on Situps. All the correlations are negative, indicating that Jumps is also a suppressor variable.

It might seem contradictory that a variable should have a coefficient of opposite sign from that of its correlation with the canonical variable. In order to understand how this can happen, consider a simplified situation: predicting Situps from Waist and Weight by multiple regression. In informal terms, it seems plausible that obese people should do fewer sit-ups than skinny people. Assume that the men in the sample do not vary much in height, so there is a strong correlation between Waist and Weight (0.8702). Examine the relationships between obesity and the independent variables:

- People with large waists tend to be more obese than people with small waists. Hence, the correlation between Waist and Situps should be negative.
- People with high weights tend to be more obese than people with low weights. Therefore, Weight should correlate negatively with Situps.
- For a fixed value of Weight, people with large waists tend to be shorter and more obese. Thus, the multiple regression coefficient for Waist should be negative.
- For a fixed value of Waist, people with higher weights tend to be taller and skinnier. The multiple regression coefficient for Weight should therefore be positive, of opposite sign from the correlation between Weight and Situps.

Therefore, the general interpretation of the first canonical correlation is that Weight and Jumps act as suppressor variables to enhance the correlation between Waist and Situps. This canonical correlation might be strong enough to be of practical interest, but the sample size is not large enough to draw definite conclusions.

The canonical redundancy analysis (Output 34.1.4) shows that neither of the first pair of canonical variables is a good overall predictor of the opposite set of variables, the proportions of variance explained being 0.2854 and 0.2584. The second and third canonical variables add virtually nothing, with cumulative proportions for all three canonical variables being 0.2969 and 0.2767.

Output 34.1.4 Canonical Redundancy Analysis

Middle-Aged Men in a Health Fitness Club
Data Courtesy of Dr. A. C. Linnerud, NC State Univ

The CANCORR Procedure

Canonical Redundancy Analysis

Standardized Variance of the Physiological Measurements Explained by					
	Their Own Canonical Variables			The Opposite Canonical Variables	
Canonical Variable Number	Proportion	Cumulative Proportion	Canonical R-Square	Proportion	Cumulative Proportion
1	0.4508	0.4508	0.6330	0.2854	0.2854
2	0.2470	0.6978	0.0402	0.0099	0.2953
3	0.3022	1.0000	0.0053	0.0016	0.2969

Standardized Variance of the Exercises Explained by					
	Their Own Canonical Variables			The Opposite Canonical Variables	
Canonical Variable Number	Proportion	Cumulative Proportion	Canonical R-Square	Proportion	Cumulative Proportion
1	0.4081	0.4081	0.6330	0.2584	0.2584
2	0.4345	0.8426	0.0402	0.0175	0.2758
3	0.1574	1.0000	0.0053	0.0008	0.2767

The squared multiple correlations (Output 34.1.5) indicate that the first canonical variable of the physiological measurements has some predictive power for Chins (0.3351) and Situps (0.4233) but almost none for Jumps (0.0167). The first canonical variable of the exercises is a fairly good predictor of Waist (0.5421), a poorer predictor of Weight (0.2438), and nearly useless for predicting Pulse (0.0701).

Output 34.1.5 Canonical Redundancy Analysis

Squared Multiple Correlations Between the Physiological Measurements and the First M Canonical Variables of the Exercises			
M	**1**	**2**	**3**
Weight	0.2438	0.2678	0.2679
Waist	0.5421	0.5478	0.5478
Pulse	0.0701	0.0702	0.0749

Squared Multiple Correlations Between the Exercises and the First M Canonical Variables of the Physiological Measurements			
M	**1**	**2**	**3**
Chins	0.3351	0.3374	0.3396
Situps	0.4233	0.4365	0.4365
Jumps	0.0167	0.0536	0.0539

References

Cooley, W. W., and Lohnes, P. R. (1971). *Multivariate Data Analysis*. New York: John Wiley & Sons.

Helland, I. S. (1987). "On the Interpretation and Use of R^2 in Regression Analysis." *Biometrics* 43:61–69.

Hotelling, H. (1935). "The Most Predictable Criterion." *Journal of Educational Psychology* 26:139–142.

Hotelling, H. (1936). "Relations between Two Sets of Variables." *Biometrika* 28:321–377.

Kshirsagar, A. M. (1972). *Multivariate Analysis*. New York: Marcel Dekker.

Lawley, D. N. (1959). "Tests of Significance in Canonical Analysis." *Biometrika* 46:59–66.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Rao, C. R. (1973). *Linear Statistical Inference and Its Applications*. 2nd ed. New York: John Wiley & Sons.

Stewart, D. K., and Love, W. A. (1968). "A General Canonical Correlation Index." *Psychological Bulletin* 70:160–163.

Tatsuoka, M. M. (1971). *Multivariate Analysis*. New York: John Wiley & Sons.

Timm, N. H. (1975). *Multivariate Analysis with Applications in Education and Psychology*. Monterey, CA: Brooks/Cole.

Van den Wollenberg, A. L. (1977). "Redundancy Analysis: An Alternative to Canonical Correlation Analysis." *Psychometrika* 42:207–219.

Wherry, R. J. (1931). "A New Formula for Predicting the Shrinkage of the Coefficient of Multiple Correlation." *Annals of Mathematical Statistics* 2:440–457.

Chapter 35
The CANDISC Procedure

Contents

Overview: CANDISC Procedure **2198**
Getting Started: CANDISC Procedure **2199**
Syntax: CANDISC Procedure **2203**
- PROC CANDISC Statement 2204
- BY Statement 2207
- CLASS Statement 2208
- FREQ Statement 2208
- VAR Statement 2208
- WEIGHT Statement 2208

Details: CANDISC Procedure **2209**
- Missing Values 2209
- Computational Details 2209
- Input Data Set 2210
- Output Data Sets 2211
- Computational Resources 2213
- Displayed Output 2214
- ODS Table Names 2216

Example: CANDISC Procedure **2217**
- Example 35.1: Analyzing Iris Data by Using PROC CANDISC 2217

References **2222**

Overview: CANDISC Procedure

Canonical discriminant analysis is a dimension-reduction technique related to principal component analysis and canonical correlation. The methodology that is used in deriving the canonical coefficients parallels that of a one-way multivariate analysis of variance (MANOVA). MANOVA tests for equality of the mean vector across class levels. Canonical discriminant analysis finds linear combinations of the quantitative variables that provide maximal separation between classes or groups. Given a classification variable and several quantitative variables, the CANDISC procedure derives *canonical variables*, which are linear combinations of the quantitative variables that summarize between-class variation in much the same way that principal components summarize total variation.

The CANDISC procedure performs a canonical discriminant analysis, computes squared Mahalanobis distances between class means, and performs both univariate and multivariate one-way analyses of variance. Two output data sets can be produced: one that contains the canonical coefficients and another that contains, among other things, scored canonical variables. You can rotate the canonical coefficients by using the FACTOR procedure. It is customary to standardize the canonical coefficients so that the canonical variables have means that are equal to 0 and pooled within-class variances that are equal to 1. PROC CANDISC displays both standardized and unstandardized canonical coefficients. Correlations between the canonical variables and the original variables in addition to the class means for the canonical variables are also displayed; these correlations, sometimes known as loadings, are called canonical structures. To aid the visual interpretation of group differences, you can use ODS Graphics to display graphs of pairs of canonical variables from the scored canonical variables output data set.

When you have two or more groups of observations that have measurements on several quantitative variables, canonical discriminant analysis derives a linear combination of the variables that has the highest possible multiple correlation with the groups. This maximal multiple correlation is called the *first canonical correlation*. The coefficients of the linear combination are the *canonical coefficients* or *canonical weights*. The variable that is defined by the linear combination is the *first canonical variable* or *canonical component*. The second canonical correlation is obtained by finding the linear combination uncorrelated with the first canonical variable that has the highest possible multiple correlation with the groups. The process of extracting canonical variables can be repeated until the number of canonical variables equals the number of original variables or the number of classes minus one, whichever is smaller.

The first canonical correlation is at least as large as the multiple correlation between the groups and any of the original variables. If the original variables have high within-group correlations, the first canonical correlation can be large even if all the multiple correlations are small. In other words, the first canonical variable can show substantial differences between the classes, even if none of the original variables do. Canonical variables are sometimes called *discriminant functions*, but this usage is ambiguous because the DISCRIM procedure produces very different functions for classification that are also called discriminant functions.

For each canonical correlation, PROC CANDISC tests the hypothesis that it and all smaller canonical correlations are zero in the population. An F approximation (Rao 1973; Kshirsagar 1972) is used that gives better small-sample results than the usual chi-square approximation. The variables should have an approximate multivariate normal distribution within each class, with a common covariance matrix in order for the probability levels to be valid.

Canonical discriminant analysis is equivalent to canonical correlation analysis between the quantitative variables and a set of dummy variables coded from the CLASS variable. Performing canonical discriminant

analysis is also equivalent to performing the following steps:

1. Transform the variables so that the pooled within-class covariance matrix is an identity matrix.
2. Compute class means on the transformed variables.
3. Perform a principal component analysis on the means, weighting each mean by the number of observations in the class. The eigenvalues are equal to the ratio of between-class variation to within-class variation in the direction of each principal component.
4. Back-transform the principal components into the space of the original variables to obtain the canonical variables.

An interesting property of the canonical variables is that they are uncorrelated whether the correlation is calculated from the total sample or from the pooled within-class correlations. However, the canonical coefficients are not orthogonal, so the canonical variables do not represent perpendicular directions through the space of the original variables.

Getting Started: CANDISC Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/candgs.sas.)

The data in this example are measurements of 159 fish caught in Finland's Lake Laengelmaevesi; this data set is available from the Puranen. For each of the seven species (bream, roach, whitefish, parkki, perch, pike, and smelt), the weight, length, height, and width of each fish are tallied. Three different length measurements are recorded: from the nose of the fish to the beginning of its tail, from the nose to the notch of its tail, and from the nose to the end of its tail. The height and width are recorded as percentages of the third length variable. The fish data set is available from the Sashelp library.

The following step uses PROC CANDISC to find the three canonical variables that best separate the species of fish in the Sashelp.Fish data and create the output data set outcan. When the NCAN=3 option is specified, only the first three canonical variables are displayed. The ODS EXCLUDE statement excludes the canonical structure tables and most of the canonical coefficient tables in order to obtain a more compact set of results. The TEMPLATE and SGRENDER procedures create a plot of the first two canonical variables. The following statements produce Figure 35.1 through Figure 35.6:

```
title 'Fish Measurement Data';

proc candisc data=sashelp.fish ncan=3 out=outcan;
   ods exclude tstruc bstruc pstruc tcoef pcoef;
   class Species;
   var Weight Length1 Length2 Length3 Height Width;
run;

proc template;
   define statgraph scatter;
      begingraph / attrpriority=none;
         entrytitle 'Fish Measurement Data';
         layout overlayequated / equatetype=fit
            xaxisopts=(label='Canonical Variable 1')
```

```
            yaxisopts=(label='Canonical Variable 2');
            scatterplot x=Can1 y=Can2 / group=species name='fish'
                                        markerattrs=(size=3px);
            layout gridded / autoalign=(topright);
               discretelegend 'fish' / border=false opaque=false;
            endlayout;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=outcan template=scatter;
run;
```

PROC CANDISC begins by displaying summary information about the variables in the analysis. This information includes the number of observations, the number of quantitative variables in the analysis (specified with the VAR statement), and the number of classes in the classification variable (specified with the CLASS statement). The frequency of each class is also displayed.

Figure 35.1 Summary Information

Fish Measurement Data

The CANDISC Procedure

Total Sample Size	158	**DF Total**	157
Variables	6	**DF Within Classes**	151
Classes	7	**DF Between Classes**	6

Number of Observations Read	159
Number of Observations Used	158

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion
Bream	Bream	34	34.0000	0.215190
Parkki	Parkki	11	11.0000	0.069620
Perch	Perch	56	56.0000	0.354430
Pike	Pike	17	17.0000	0.107595
Roach	Roach	20	20.0000	0.126582
Smelt	Smelt	14	14.0000	0.088608
Whitefish	Whitefish	6	6.0000	0.037975

PROC CANDISC performs a multivariate one-way analysis of variance (one-way MANOVA) and provides four multivariate tests of the hypothesis that the class mean vectors are equal. These tests, shown in Figure 35.2, indicate that not all the mean vectors are equal ($p < 0.0001$).

Figure 35.2 MANOVA and Multivariate Tests

Fish Measurement Data

The CANDISC Procedure

Multivariate Statistics and F Approximations					
	S=6	M=-0.5	N=72		
Statistic	**Value**	**F Value**	**Num DF**	**Den DF**	**Pr > F**
Wilks' Lambda	0.00036325	90.71	36	643.89	<.0001
Pillai's Trace	3.10465132	26.99	36	906	<.0001
Hotelling-Lawley Trace	52.05799676	209.24	36	413.64	<.0001
Roy's Greatest Root	39.13499776	984.90	6	151	<.0001
NOTE: F Statistic for Roy's Greatest Root is an upper bound.					

The first canonical correlation is the greatest possible multiple correlation with the classes that can be achieved by using a linear combination of the quantitative variables. The first canonical correlation, displayed in Figure 35.3, is 0.987463. Figure 35.3 also displays a likelihood ratio test of the hypothesis that the current canonical correlation and all smaller ones are zero. The first line is equivalent to Wilks' lambda multivariate test.

Figure 35.3 Canonical Correlations

Fish Measurement Data

The CANDISC Procedure

					Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq)			
	Canonical Correlation	**Adjusted Canonical Correlation**	**Approximate Standard Error**	**Squared Canonical Correlation**	**Eigenvalue**	**Difference**	**Proportion**	**Cumulative**
1	0.987463	0.986671	0.001989	0.975084	39.1350	29.3859	0.7518	0.7518
2	0.952349	0.950095	0.007425	0.906969	9.7491	7.3786	0.1873	0.9390
3	0.838637	0.832518	0.023678	0.703313	2.3706	1.7016	0.0455	0.9846
4	0.633094	0.623649	0.047821	0.400809	0.6689	0.5346	0.0128	0.9974
5	0.344157	0.334170	0.070356	0.118444	0.1344	0.1343	0.0026	1.0000
6	0.005701	.	0.079806	0.000033	0.0000		0.0000	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero					
	Likelihood Ratio	**Approximate F Value**	**Num DF**	**Den DF**	**Pr > F**
1	0.00036325	90.71	36	643.89	<.0001
2	0.01457896	46.46	25	547.58	<.0001
3	0.15671134	23.61	16	452.79	<.0001
4	0.52820347	12.09	9	362.78	<.0001
5	0.88152702	4.88	4	300	0.0008
6	0.99996749	0.00	1	151	0.9442

The first canonical variable, Can1, shows that the linear combination of the centered variables Can1 = $-0.0006 \times$ Weight $- 0.33 \times$ Length1 $2.49 \times$ Length2 $+ 2.60 \times$ Length3 $+ 1.12 \times$ Height $- 1.45 \times$ Width separates the species most effectively (see Figure 35.4).

Figure 35.4 Raw Canonical Coefficients

Fish Measurement Data

The CANDISC Procedure

Raw Canonical Coefficients			
Variable	**Can1**	**Can2**	**Can3**
Weight	-0.000648508	-0.005231659	-0.005596192
Length1	-0.329435762	-0.626598051	-2.934324102
Length2	-2.486133674	-0.690253987	4.045038893
Length3	2.595648437	1.803175454	-1.139264914
Height	1.121983854	-0.714749340	0.283202557
Width	-1.446386704	-0.907025481	0.741486686

PROC CANDISC computes the means of the canonical variables for each class. The first canonical variable is the linear combination of the variables Weight, Length1, Length2, Length3, Height, and Width that provides the greatest difference (in terms of a univariate F test) between the class means. The second canonical variable provides the greatest difference between class means while being uncorrelated with the first canonical variable.

Figure 35.5 Class Means for Canonical Variables

Class Means on Canonical Variables			
Species	**Can1**	**Can2**	**Can3**
Bream	10.94142464	0.52078394	0.23496708
Parkki	2.58903743	-2.54722416	-0.49326158
Perch	-4.47181389	-1.70822715	1.29281314
Pike	-4.89689441	8.22140791	-0.16469132
Roach	-0.35837149	0.08733611	-1.10056438
Smelt	-4.09136653	-2.35805841	-4.03836098
Whitefish	-0.39541755	-0.42071778	1.06459242

Figure 35.6 displays a plot of the first two canonical variables, which shows that Can1 discriminates among three groups: (1) bream; (2) whitefish, roach, and parkki; and (3) smelt, pike, and perch. Can2 best discriminates between pike and the other species.

Figure 35.6 Plot of First Two Canonical Variables

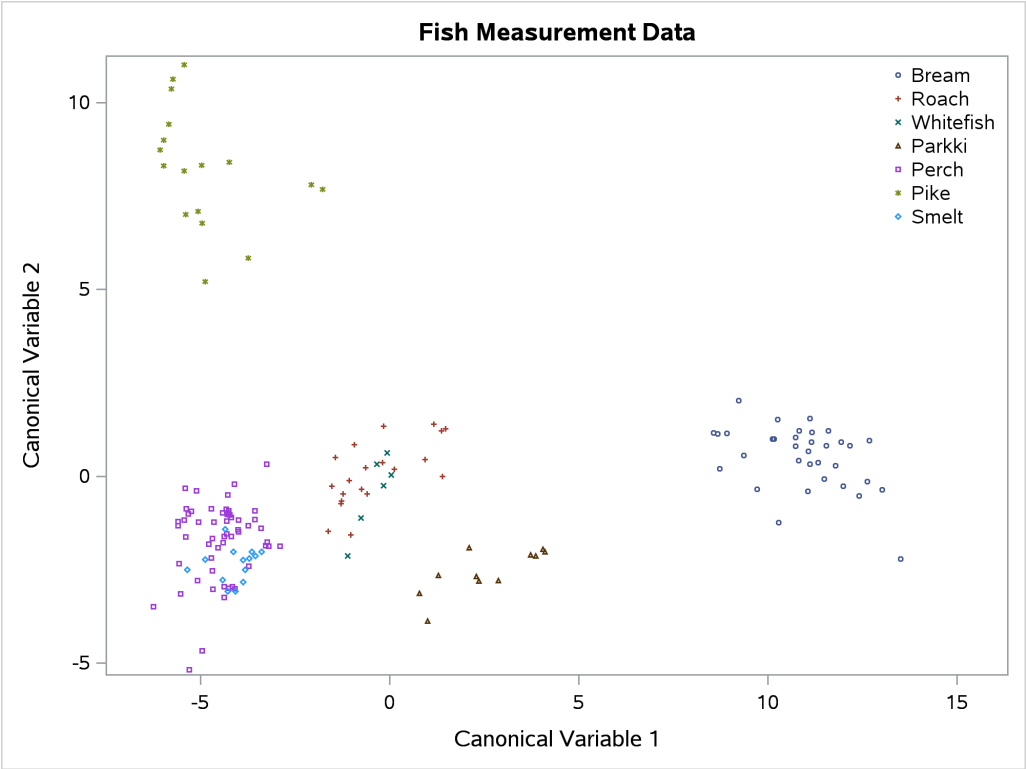

Syntax: CANDISC Procedure

The following statements are available in the CANDISC procedure:

PROC CANDISC < *options* > **;**
  CLASS *variable* **;**
  BY *variables* **;**
  FREQ *variable* **;**
  VAR *variables* **;**
  WEIGHT *variable* **;**

The BY, CLASS, FREQ, VAR, and WEIGHT statements are described in alphabetical order after the PROC CANDISC statement.

PROC CANDISC Statement

PROC CANDISC < *options* > **;**

The PROC CANDISC statement invokes the CANDISC procedure. Table 35.1 summarizes the options available in the PROC CANDISC statement.

Table 35.1 CANDISC Procedure Options

Option	Description
Input Data Set	
DATA=	Specifies the input SAS data set
Output Data Sets	
OUT=	Specifies the output data set that contains the canonical scores
OUTSTAT=	Specifies the output statistics data set
Method Details	
NCAN=	Specifies the number of canonical variables
PREFIX=	Specifies a prefix for naming the canonical variables
SINGULAR=	Specifies the singularity criterion
Control Displayed Output	
ALL	Displays all output
ANOVA	Displays univariate statistics
BCORR	Displays between correlations
BCOV	Displays between covariances
BSSCP	Displays between SSCPs
DISTANCE	Displays squared Mahalanobis distances
NOPRINT	Suppresses all displayed output
PCORR	Displays pooled correlations
PCOV	Displays pooled covariances
PSSCP	Displays pooled SSCPs
SHORT	Suppresses some displayed output
SIMPLE	Displays simple descriptive statistics
STDMEAN	Displays standardized class means
TCORR	Displays total correlations
TCOV	Displays total covariances
TSSCP	Displays total SSCPs
WCORR	Displays within correlations
WCOV	Displays within covariances
WSSCP	Displays within SSCPs

ALL

activates all the display options.

ANOVA

displays univariate statistics for testing the hypothesis that the class means are equal in the population for each variable.

BCORR

displays between-class correlations.

BCOV

displays between-class covariances. The between-class covariance matrix equals the between-class SSCP matrix divided by $n(c-1)/c$, where n is the number of observations and c is the number of classes. The between-class covariances should be interpreted in comparison with the total-sample and within-class covariances, not as formal estimates of population parameters.

BSSCP

displays the between-class SSCP matrix.

DATA=*SAS-data-set*

specifies the data set to be analyzed. The data set can be an ordinary SAS data set or one of several specially structured data sets created by SAS statistical procedures. These specially structured data sets include TYPE=CORR, TYPE=COV, TYPE=CSSCP, and TYPE=SSCP. If you omit the DATA= option, PROC CANDISC uses the most recently created SAS data set.

DISTANCE

MAHALANOBIS

displays squared Mahalanobis distances between the group means, the F statistics, and the corresponding probabilities of greater squared Mahalanobis distances between the group means.

NCAN=*n*

specifies the number of canonical variables to be computed. The value of n must be less than or equal to the number of variables. If you specify NCAN=0, the procedure displays the canonical correlations but not the canonical coefficients, structures, or means. A negative value suppresses the canonical analysis entirely. Let v be the number of variables in the VAR statement, and let c be the number of classes. If you omit the NCAN= option, only $\min(v, c-1)$ canonical variables are generated; if you also specify an OUT= output data set, v canonical variables are generated, and the last $v-(c-1)$ canonical variables have missing values.

NOPRINT

suppresses the normal display of results. This option temporarily disables the Output Delivery System (ODS). For more information about ODS, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*

creates an output SAS data set to contain the original data and the canonical variable scores. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTSTAT=*SAS-data-set*

creates a TYPE=CORR output SAS data set to contain various statistics, including class means, standard deviations, correlations, canonical correlations, canonical structures, canonical coefficients, and means of canonical variables for each class. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PCORR

displays pooled within-class correlations (partial correlations based on the pooled within-class covariances).

PCOV

displays pooled within-class covariances.

PREFIX=*name*

specifies a prefix for naming the canonical variables. By default the names are Can1, Can2, Can3, and so on. If you specify PREFIX=Abc, the components are named Abc1, Abc2, and so on. The number of characters in the prefix plus the number of digits required to designate the canonical variables should not exceed the length defined by the VALIDVARNAME= system option (for example, 32 for VALIDVARNAME=V7). The prefix is truncated if the combined length exceeds the maximum.

PSSCP

displays the pooled within-class corrected SSCP matrix.

SHORT

suppresses the display of canonical structures, canonical coefficients, and class means on canonical variables; only tables of canonical correlations and multivariate test statistics are displayed.

SIMPLE

displays simple descriptive statistics for the total sample and within each class.

SINGULAR=*p*

specifies the criterion for determining the singularity of the total-sample correlation matrix and the pooled within-class covariance matrix, where $0 < p < 1$. The default is SINGULAR=1E–8.

Let **S** be the total-sample correlation matrix. If the R square for predicting a quantitative variable in the VAR statement from the variables that precede it exceeds $1 - p$, then **S** is considered singular. If **S** is singular, the probability levels for the multivariate test statistics and canonical correlations are adjusted for the number of variables whose R square exceeds $1 - p$.

If **S** is considered singular and the inverse of **S** (squared Mahalanobis distances) is required, a quasi inverse is used instead. For more information, see the section "Quasi-inverse" on page 2745 in Chapter 42, "The DISCRIM Procedure."

STDMEAN

displays total-sample and pooled within-class standardized class means.

TCORR
displays total-sample correlations.

TCOV
displays total-sample covariances.

TSSCP
displays the total-sample corrected SSCP matrix.

WCORR
displays within-class correlations for each class level.

WCOV
displays within-class covariances for each class level.

WSSCP
displays the within-class corrected SSCP matrix for each class level.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC CANDISC to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the CANDISC procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* **;**

The values of the CLASS variable define the groups for analysis. Class levels are determined by the formatted values of the CLASS variable. The CLASS variable can be numeric or character. A CLASS statement is required.

FREQ Statement

FREQ *variable* **;**

If a variable in the data set represents the frequency of occurrence of the other values in the observation, include the name of the variable in a FREQ statement. The procedure then treats the data set as if each observation appears *n* times, where *n* is the value of the FREQ variable for the observation. The total number of observations is considered to be equal to the sum of the FREQ variable when the procedure determines degrees of freedom for significance probabilities.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, the value is truncated to an integer.

VAR Statement

VAR *variables* **;**

You specify the quantitative variables to include in the analysis by using a VAR statement. If you do not use a VAR statement, the analysis includes all numeric variables not listed in other statements.

WEIGHT Statement

WEIGHT *variable* **;**

To use relative weights for each observation in the input data set, place the weights in a variable in the data set and specify the name in a WEIGHT statement. This is often done when the variance associated with each observation is different and the values of the WEIGHT variable are proportional to the reciprocals of the variances. If the value of the WEIGHT variable is missing or is less than 0, then a value of 0 for the weight is assumed.

The WEIGHT and FREQ statements have a similar effect except that the WEIGHT statement does not alter the degrees of freedom.

Details: CANDISC Procedure

Missing Values

If an observation has a missing value for any of the quantitative variables, it is omitted from the analysis. If an observation has a missing CLASS value but is otherwise complete, it is not used in computing the canonical correlations and coefficients; however, canonical variable scores are computed for that observation for the OUT= data set.

Computational Details

General Formulas

Canonical discriminant analysis is equivalent to canonical correlation analysis between the quantitative variables and a set of dummy variables coded from the CLASS variable. In the following notation, the dummy variables are denoted by $\mathbf{y}$ and the quantitative variables are denoted by $\mathbf{x}$. The total sample covariance matrix for the $\mathbf{x}$ and $\mathbf{y}$ variables is

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_{xx} & \mathbf{S}_{xy} \\ \mathbf{S}_{yx} & \mathbf{S}_{yy} \end{bmatrix}$$

When c is the number of groups, n_t is the number of observations in group t, and $\mathbf{S}_t$ is the sample covariance matrix for the $\mathbf{x}$ variables in group t, the within-class pooled covariance matrix for the $\mathbf{x}$ variables is

$$\mathbf{S}_p = \frac{1}{\sum n_t - c} \sum (n_t - 1)\mathbf{S}_t$$

The canonical correlations, ρ_i, are the square roots of the eigenvalues, λ_i, of the following matrix. The corresponding eigenvectors are $\mathbf{v}_i$.

$$\mathbf{S}_p{}^{-1/2}\mathbf{S}_{xy}\mathbf{S}_{yy}{}^{-1}\mathbf{S}_{yx}\mathbf{S}_p{}^{-1/2}$$

Let $\mathbf{V}$ be the matrix that contains the eigenvectors $\mathbf{v}_i$ that correspond to nonzero eigenvalues as columns. The raw canonical coefficients are calculated as follows:

$$\mathbf{R} = \mathbf{S}_p{}^{-1/2}\mathbf{V}$$

The pooled within-class standardized canonical coefficients are

$$\mathbf{P} = \mathrm{diag}(\mathbf{S}_p)^{1/2}\mathbf{R}$$

The total sample standardized canonical coefficients are

$$\mathbf{T} = \mathrm{diag}(\mathbf{S}_{xx})^{1/2}\mathbf{R}$$

Let $\mathbf{X}_c$ be the matrix that contains the centered $\mathbf{x}$ variables as columns. The canonical scores can be calculated by any of the following:

$$\mathbf{X}_c\,\mathbf{R}$$

$$\mathbf{X}_c \operatorname{diag}(\mathbf{S}_p)^{-1/2}\mathbf{P}$$

$$\mathbf{X}_c \operatorname{diag}(\mathbf{S}_{xx})^{-1/2}\mathbf{T}$$

For the multivariate tests based on $\mathbf{E}^{-1}\mathbf{H}$,

$$\mathbf{E} = (n-1)(\mathbf{S}_{yy} - \mathbf{S}_{yx}\mathbf{S}_{xx}^{-1}\mathbf{S}_{xy})$$

$$\mathbf{H} = (n-1)\mathbf{S}_{yx}\mathbf{S}_{xx}^{-1}\mathbf{S}_{xy}$$

where n is the total number of observations.

Input Data Set

The input DATA= data set can be an ordinary SAS data set or one of several specially structured data sets created by statistical procedures available in SAS/STAT software. For more information about special types of data sets, see Appendix A, "Special SAS Data Sets." The BY variable in these data sets becomes the CLASS variable in PROC CANDISC. These specially structured data sets include the following:

- TYPE=CORR data sets created by PROC CORR by using a BY statement
- TYPE=COV data sets created by PROC PRINCOMP by using both the COV option and a BY statement
- TYPE=CSSCP data sets created by PROC CORR by using the CSSCP option and a BY statement, where the OUT= data set is assigned TYPE=CSSCP by using the TYPE= data set option
- TYPE=SSCP data sets created by PROC REG by using both the OUTSSCP= option and a BY statement

When the input data set is TYPE=CORR, TYPE=COV, or TYPE=CSSCP, then PROC CANDISC reads the number of observations for each class from the observations for which _TYPE_='N' and the variable means in each class from the observations for which _TYPE_='MEAN'. The CANDISC procedure then reads the within-class correlations from the observations for which _TYPE_='CORR', the standard deviations from the observations for which _TYPE_='STD' (data set TYPE=CORR), the within-class covariances from the observations for which _TYPE_='COV' (data set TYPE=COV), or the within-class corrected sums of squares and crossproducts from the observations for which _TYPE_='CSSCP' (data set TYPE=CSSCP).

When the data set does not include any observations for which _TYPE_='CORR' (data set TYPE=CORR), _TYPE_='COV' (data set TYPE=COV), or _TYPE_='CSSCP' (data set TYPE=CSSCP) for each class, PROC CANDISC reads the pooled within-class information from the data set. In this case, PROC CANDISC reads the pooled within-class correlations from the observations for which _TYPE_='PCORR', the pooled within-class standard deviations from the observations for which _TYPE_='PSTD' (data set TYPE=CORR), the pooled within-class covariances from the observations for which _TYPE_='PCOV' (data set TYPE=COV), or the pooled within-class corrected SSCP matrix from the observations for which _TYPE_='PSSCP' (data set TYPE=CSSCP).

When the input data set is TYPE=SSCP, then PROC CANDISC reads the number of observations for each class from the observations for which _TYPE_='N', the sum of weights of observations from the variable Intercept in observations for which _TYPE_='SSCP' and _NAME_='Intercept', the variable sums from the analysis variables in observations for which _TYPE_='SSCP' and _NAME_='Intercept', and the uncorrected sums of squares and crossproducts from the analysis variables in observations for which _TYPE_='SSCP' and _NAME_=*variable-name*.

Output Data Sets

OUT= Data Set

The OUT= data set contains all the variables in the original data set plus new variables that contain the canonical variable scores. You determine the number of new variables by using the NCAN= option. The names of the new variables are formed as they are for the PREFIX= option. The new variables have means equal to 0 and pooled within-class variances equal to 1. An OUT= data set cannot be created if the DATA= data set is not an ordinary SAS data set.

OUTSTAT= Data Set

The OUTSTAT= data set is similar to the TYPE=CORR data set that the CORR procedure produces but contains many results in addition to those produced by the CORR procedure.

The OUTSTAT= data set is TYPE=CORR, and it contains the following variables:

- the BY variables, if any
- the CLASS variable
- _TYPE_, a character variable of length 8 that identifies the type of statistic
- _NAME_, a character variable that identifies the row of the matrix or the name of the canonical variable
- the quantitative variables (those in the VAR statement, or if there is no VAR statement, all numeric variables not listed in any other statement)

The observations, as identified by the variable _TYPE_, have the following _TYPE_ values:

TYPE	Contents
N	number of observations both for the total sample (CLASS variable missing) and within each class (CLASS variable present)
SUMWGT	sum of weights both for the total sample (CLASS variable missing) and within each class (CLASS variable present) if a WEIGHT statement is specified
MEAN	means both for the total sample (CLASS variable missing) and within each class (CLASS variable present)
STDMEAN	total-standardized class means
PSTDMEAN	pooled within-class standardized class means
STD	standard deviations both for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PSTD	pooled within-class standard deviations
BSTD	between-class standard deviations
RSQUARED	univariate R squares

The following kinds of observations are identified by the combination of the variables _TYPE_ and _NAME_. When the _TYPE_ variable has one of the following values, the _NAME_ variable identifies the row of the matrix:

TYPE	Contents
CSSCP	corrected SSCP matrix for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PSSCP	pooled within-class corrected SSCP matrix
BSSCP	between-class SSCP matrix
COV	covariance matrix for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PCOV	pooled within-class covariance matrix
BCOV	between-class covariance matrix
CORR	correlation matrix for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PCORR	pooled within-class correlation matrix
BCORR	between-class correlation matrix

When the _TYPE_ variable has one of the following values, the _NAME_ variable identifies the canonical variable:

TYPE	Contents
CANCORR	canonical correlations
STRUCTUR	canonical structure
BSTRUCT	between canonical structure
PSTRUCT	pooled within-class canonical structure
SCORE	total sample standardized canonical coefficients
PSCORE	pooled within-class standardized canonical coefficients
RAWSCORE	raw canonical coefficients
CANMEAN	means of the canonical variables for each class

You can use this data set in PROC SCORE to get scores on the canonical variables for new data by using one of the following forms:

```
* The CLASS variable C is numeric;
proc score data=NewData score=Coef(where=(c = .  )) out=Scores;
run;

* The CLASS variable C is character;
proc score data=NewData score=Coef(where=(c = ' ')) out=Scores;
run;
```

The WHERE clause is used to exclude the within-class means and standard deviations. PROC SCORE standardizes the new data by subtracting the original variable means that are stored in the _TYPE_='MEAN' observations and dividing by the original variable standard deviations from the _TYPE_='STD' observations. Then PROC SCORE multiplies the standardized variables by the coefficients from the _TYPE_='SCORE' observations to get the canonical scores.

Computational Resources

In the following discussion, let

$$
\begin{aligned}
n &= \text{number of observations} \\
c &= \text{number of class levels} \\
v &= \text{number of variables in the VAR list} \\
l &= \text{length of the CLASS variable}
\end{aligned}
$$

Memory Requirements

The amount of memory in bytes for temporary storage needed to process the data is

$$c(4v^2 + 28v + 4l + 68) + 16v^2 + 96v + 4l$$

For the ANOVA option, the temporary storage must be increased by $16v$ bytes. The DISTANCE option requires an additional temporary storage of $4v^2 + 4v$ bytes.

Time Requirements

The following factors determine the time requirements of the CANDISC procedure:

- The time needed for reading the data and computing covariance matrices is proportional to nv^2. PROC CANDISC must also look up each class level in the list. This is faster if the data are sorted by the CLASS variable. The time for looking up class levels is proportional to a value that ranges from n to $n \log(c)$.
- The time for inverting a covariance matrix is proportional to v^3.
- The time required for the canonical discriminant analysis is proportional to v^3.

Each of the preceding factors has a different constant of proportionality.

Displayed Output

The displayed output from PROC CANDISC includes the class level information table. For each level of the classification variable, the following information is provided: the output data set variable name, frequency sum, weight sum, and the proportion of the total sample.

The optional output from PROC CANDISC includes the following:

- Within-class SSCP matrices for each group
- Pooled within-class SSCP matrix
- Between-class SSCP matrix
- Total-sample SSCP matrix
- Within-class covariance matrices for each group
- Pooled within-class covariance matrix
- Between-class covariance matrix, equal to the between-class SSCP matrix divided by $n(c-1)/c$, where n is the number of observations and c is the number of classes
- Total-sample covariance matrix
- Within-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the within-class population correlation coefficients are zero
- Pooled within-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the partial population correlation coefficients are zero
- Between-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the between-class population correlation coefficients are zero
- Total-sample correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the total population correlation coefficients are zero
- Simple statistics, including N (the number of observations), sum, mean, variance, and standard deviation both for the total sample and within each class
- Total-sample standardized class means, obtained by subtracting the grand mean from each class mean and dividing by the total sample standard deviation
- Pooled within-class standardized class means, obtained by subtracting the grand mean from each class mean and dividing by the pooled within-class standard deviation
- Pairwise squared distances between groups
- Univariate test statistics, including total-sample standard deviations, pooled within-class standard deviations, between-class standard deviations, R square, $R^2/(1-R^2)$, F, and $\mathrm{Pr} > F$ (univariate F values and probability levels for one-way analyses of variance)

By default, PROC CANDISC displays these statistics:

- Multivariate statistics and F approximations, including Wilks' lambda, Pillai's trace, Hotelling-Lawley trace, and Roy's greatest root with F approximations, numerator and denominator degrees of freedom (Num DF and Den DF), and probability values ($\Pr > F$). Each of these four multivariate statistics tests the hypothesis that the class means are equal in the population. For more information, see the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures."

- Canonical correlations

- Adjusted canonical correlations (Lawley 1959). These are asymptotically less biased than the raw correlations and can be negative. The adjusted canonical correlations might not be computable and are displayed as missing values if two canonical correlations are nearly equal or if some are close to zero. A missing value is also displayed if an adjusted canonical correlation is larger than a previous adjusted canonical correlation.

- Approximate standard error of the canonical correlations

- Squared canonical correlations

- Eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$. Each eigenvalue is equal to $\rho^2/(1-\rho^2)$, where ρ^2 is the corresponding squared canonical correlation and can be interpreted as the ratio of between-class variation to pooled within-class variation for the corresponding canonical variable. The table includes eigenvalues, differences between successive eigenvalues, the proportion of the sum of the eigenvalues, and the cumulative proportion.

- Likelihood ratio for the hypothesis that the current canonical correlation and all smaller ones are zero in the population. The likelihood ratio for the hypothesis that all canonical correlations equal zero is Wilks' lambda.

- Approximate F statistic based on Rao's approximation to the distribution of the likelihood ratio (Rao 1973, p. 556; Kshirsagar 1972, p. 326)

- Numerator degrees of freedom (Num DF), denominator degrees of freedom (Den DF), and $\Pr > F$, the probability level associated with the F statistic

You can suppress the following statistics by specifying the SHORT option:

- Total canonical structure, giving total-sample correlations between the canonical variables and the original variables

- Between canonical structure, giving between-class correlations between the canonical variables and the original variables

- Pooled within canonical structure, giving pooled within-class correlations between the canonical variables and the original variables

- Total-sample standardized canonical coefficients, standardized to give canonical variables that have zero mean and unit pooled within-class variance when applied to the total-sample standardized variables

- Pooled within-class standardized canonical coefficients, standardized to give canonical variables that have zero mean and unit pooled within-class variance when applied to the pooled within-class standardized variables

- Raw canonical coefficients, standardized to give canonical variables that have zero mean and unit pooled within-class variance when applied to the centered variables
- Class means on the canonical variables

ODS Table Names

PROC CANDISC assigns a name to each table that it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 35.2. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 35.2 ODS Tables Produced by PROC CANDISC

ODS Table Name	Description	PROC CANDISC Option
ANOVA	Univariate statistics	ANOVA
AveRSquare	Average R square	ANOVA
BCorr	Between-class correlations	BCORR
BCov	Between-class covariances	BCOV
BSSCP	Between-class SSCP matrix	BSSCP
BStruc	Between canonical structure	Default
CanCorr	Canonical correlations	Default
CanonicalMeans	Class means on canonical variables	Default
Counts	Number of observations, variables, classes, degrees of freedom	Default
CovDF	Degrees of freedom for covariance matrices, not printed	Any *COV option
Dist	Squared distances	DISTANCE
DistFValues	F statistics based on squared distances	DISTANCE
DistProb	Probabilities for F statistics from squared distances	DISTANCE
Levels	Class level information	Default
MultStat	MANOVA	Default
NObs	Number of observations	Default
PCoef	Pooled standard canonical coefficients	Default
PCorr	Pooled within-class correlations	PCORR
PCov	Pooled within-class covariances	PCOV
PSSCP	Pooled within-class SSCP matrix	PSSCP
PStdMeans	Pooled standardized class means	STDMEAN
PStruc	Pooled within canonical structure	Default
RCoef	Raw canonical coefficients	Default
SimpleStatistics	Simple statistics	SIMPLE
TCoef	Total-sample standard canonical coefficients	Default
TCorr	Total-sample correlations	TCORR
TCov	Total-sample covariances	TCOV

Table 35.2 *continued*

ODS Table Name	Description	PROC CANDISC Option
TSSCP	Total-sample SSCP matrix	TSSCP
TStdMeans	Total standardized class means	STDMEAN
TStruc	Total canonical structure	Default
WCorr	Within-class correlations	WCORR
WCov	Within-class covariances	WCOV
WSSCP	Within-class SSCP matrices	WSSCP

Example: CANDISC Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 35.1: Analyzing Iris Data by Using PROC CANDISC

(View the complete code for this example (candex1.sas) in the example repository.)

The iris data that were published by Fisher (1936) have been widely used for examples in discriminant analysis and cluster analysis. The sepal length, sepal width, petal length, and petal width are measured in millimeters in 50 iris specimens from each of three species: *Iris setosa*, *I. versicolor*, and *I. virginica*. The iris data set is available from the Sashelp library.

This example is a canonical discriminant analysis that creates an output data set that contains scores on the canonical variables and plots the canonical variables.

The following statements produce Output 35.1.1 through Output 35.1.6:

```
title 'Fisher (1936) Iris Data';

proc candisc data=sashelp.iris out=outcan distance anova;
   class Species;
   var SepalLength SepalWidth PetalLength PetalWidth;
run;
```

PROC CANDISC first displays information about the observations and the classes in the data set in Output 35.1.1.

Output 35.1.1 Iris Data: Summary Information

Fisher (1936) Iris Data

The CANDISC Procedure

Total Sample Size	150	**DF Total**	149
Variables	4	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion
Setosa	Setosa	50	50.0000	0.333333
Versicolor	Versicolor	50	50.0000	0.333333
Virginica	Virginica	50	50.0000	0.333333

The DISTANCE option in the PROC CANDISC statement displays squared Mahalanobis distances between class means. Results from the DISTANCE option are shown in Output 35.1.2.

Output 35.1.2 Iris Data: Squared Mahalanobis Distances and Distance Statistics

Fisher (1936) Iris Data

The CANDISC Procedure

Squared Distance to Species

From Species	Setosa	Versicolor	Virginica
Setosa	0	89.86419	179.38471
Versicolor	89.86419	0	17.20107
Virginica	179.38471	17.20107	0

F Statistics, NDF=4, DDF=144 for Squared Distance to Species

From Species	Setosa	Versicolor	Virginica
Setosa	0	550.18889	1098
Versicolor	550.18889	0	105.31265
Virginica	1098	105.31265	0

Prob > Mahalanobis Distance for Squared Distance to Species

From Species	Setosa	Versicolor	Virginica
Setosa	1.0000	<.0001	<.0001
Versicolor	<.0001	1.0000	<.0001
Virginica	<.0001	<.0001	1.0000

Output 35.1.3 displays univariate and multivariate statistics. The ANOVA option uses univariate statistics

to test the hypothesis that the class means are equal. The resulting R-square values range from 0.4008 for SepalWidth to 0.9414 for PetalLength, and each variable is significant at the 0.0001 level. The multivariate test for differences between the classes (which is displayed by default) is also significant at the 0.0001 level; you would expect this from the highly significant univariate test results.

Output 35.1.3 Iris Data: Univariate and Multivariate Statistics

Fisher (1936) Iris Data

The CANDISC Procedure

Univariate Test Statistics

F Statistics, Num DF=2, Den DF=147

Variable	Label	Total Standard Deviation	Pooled Standard Deviation	Between Standard Deviation	R-Square	R-Square / (1-RSq)	F Value	Pr > F
SepalLength	Sepal Length (mm)	8.2807	5.1479	7.9506	0.6187	1.6226	119.26	<.0001
SepalWidth	Sepal Width (mm)	4.3587	3.3969	3.3682	0.4008	0.6688	49.16	<.0001
PetalLength	Petal Length (mm)	17.6530	4.3033	20.9070	0.9414	16.0566	1180.16	<.0001
PetalWidth	Petal Width (mm)	7.6224	2.0465	8.9673	0.9289	13.0613	960.01	<.0001

Average R-Square	
Unweighted	0.7224358
Weighted by Variance	0.8689444

Multivariate Statistics and F Approximations

S=2 M=0.5 N=71

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.02343863	199.15	8	288	<.0001
Pillai's Trace	1.19189883	53.47	8	290	<.0001
Hotelling-Lawley Trace	32.47732024	582.20	8	203.4	<.0001
Roy's Greatest Root	32.19192920	1166.96	4	145	<.0001

NOTE: F Statistic for Roy's Greatest Root is an upper bound.

NOTE: F Statistic for Wilks' Lambda is exact.

Output 35.1.4 displays canonical correlations and eigenvalues. The R square between Can1 and the CLASS variable, 0.969872, is much larger than the corresponding R square for Can2, 0.222027.

Output 35.1.4 Iris Data: Canonical Correlations and Eigenvalues

Fisher (1936) Iris Data

The CANDISC Procedure

	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq) Eigenvalue	Difference	Proportion	Cumulative
1	0.984821	0.984508	0.002468	0.969872	32.1919	31.9065	0.9912	0.9912
2	0.471197	0.461445	0.063734	0.222027	0.2854		0.0088	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero					
	Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
1	0.02343863	199.15	8	288	<.0001
2	0.77797337	13.79	3	145	<.0001

Output 35.1.5 displays correlations between canonical and original variables.

Output 35.1.5 Iris Data: Correlations between Canonical and Original Variables

Fisher (1936) Iris Data

The CANDISC Procedure

Total Canonical Structure			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	0.791888	0.217593
SepalWidth	Sepal Width (mm)	-0.530759	0.757989
PetalLength	Petal Length (mm)	0.984951	0.046037
PetalWidth	Petal Width (mm)	0.972812	0.222902

Between Canonical Structure			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	0.991468	0.130348
SepalWidth	Sepal Width (mm)	-0.825658	0.564171
PetalLength	Petal Length (mm)	0.999750	0.022358
PetalWidth	Petal Width (mm)	0.994044	0.108977

Pooled Within Canonical Structure			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	0.222596	0.310812
SepalWidth	Sepal Width (mm)	-0.119012	0.863681
PetalLength	Petal Length (mm)	0.706065	0.167701
PetalWidth	Petal Width (mm)	0.633178	0.737242

Output 35.1.6 displays canonical coefficients. The raw canonical coefficients for the first canonical variable, Can1, show that the classes differ most widely on the linear combination of the centered variables: $-0.0829378 \times \text{SepalLength} - 0.153447 \times \text{SepalWidth} + 0.220121 \times \text{PetalLength} + 0.281046 \times \text{PetalWidth}$.

Output 35.1.6 Iris Data: Canonical Coefficients

Fisher (1936) Iris Data

The CANDISC Procedure

Total-Sample Standardized Canonical Coefficients			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	-0.686779533	0.019958173
SepalWidth	Sepal Width (mm)	-0.668825075	0.943441829
PetalLength	Petal Length (mm)	3.885795047	-1.645118866
PetalWidth	Petal Width (mm)	2.142238715	2.164135931

Pooled Within-Class Standardized Canonical Coefficients			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	-.4269548486	0.0124075316
SepalWidth	Sepal Width (mm)	-.5212416758	0.7352613085
PetalLength	Petal Length (mm)	0.9472572487	-.4010378190
PetalWidth	Petal Width (mm)	0.5751607719	0.5810398645

Raw Canonical Coefficients			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	-.0829377642	0.0024102149
SepalWidth	Sepal Width (mm)	-.1534473068	0.2164521235
PetalLength	Petal Length (mm)	0.2201211656	-.0931921210
PetalWidth	Petal Width (mm)	0.2810460309	0.2839187853

Output 35.1.7 displays class level means on canonical variables.

Output 35.1.7 Iris Data: Canonical Means

Class Means on Canonical Variables		
Species	**Can1**	**Can2**
Setosa	-7.607599927	0.215133017
Versicolor	1.825049490	-0.727899622
Virginica	5.782550437	0.512766605

The TEMPLATE and SGRENDER procedures are used to create a plot of the first two canonical variables. The following statements produce Output 35.1.8:

```
proc template;
   define statgraph scatter;
      begingraph / attrpriority=none;
         entrytitle 'Fisher (1936) Iris Data';
         layout overlayequated / equatetype=fit
            xaxisopts=(label='Canonical Variable 1')
            yaxisopts=(label='Canonical Variable 2');
            scatterplot x=Can1 y=Can2 / group=species name='iris'
                                        markerattrs=(size=3px);
            layout gridded / autoalign=(topright topleft);
               discretelegend 'iris' / border=false opaque=false;
```

```
            endlayout;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=outcan template=scatter;
run;
```

Output 35.1.8 Iris Data: Plot of First Two Canonical Variables

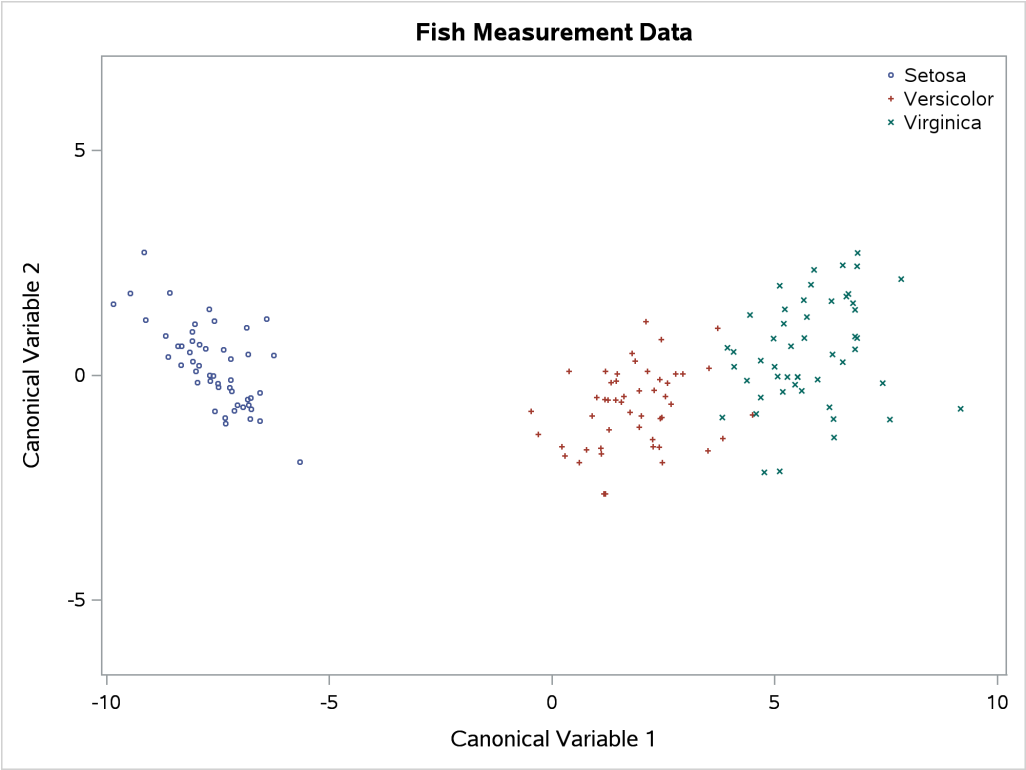

The plot of canonical variables in Output 35.1.8 shows that of the two canonical variables, Can1 has more discriminatory power.

References

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Kshirsagar, A. M. (1972). *Multivariate Analysis*. New York: Marcel Dekker.

Lawley, D. N. (1959). "Tests of Significance in Canonical Analysis." *Biometrika* 46:59–66.

Puranen, J. (1917). "Fish Catch data set (1917)." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Rao, C. R. (1973). *Linear Statistical Inference and Its Applications*. 2nd ed. New York: John Wiley & Sons.

Chapter 36
The CATMOD Procedure

Contents

Overview: CATMOD Procedure . . . **2226**
- Types of Input Data . . . 2227
- Types of Statistical Analyses . . . 2227
- Background: The Underlying Model . . . 2229
- Linear Models Contrasted with Log-Linear Models . . . 2230
- Using PROC CATMOD Interactively . . . 2231

Getting Started: CATMOD Procedure . . . **2231**
- Weighted Least Squares Analysis of Mean Response . . . 2232
- Generalized Logits Model . . . 2236

Syntax: CATMOD Procedure . . . **2240**
- PROC CATMOD Statement . . . 2241
- BY Statement . . . 2242
- CONTRAST Statement . . . 2243
- DIRECT Statement . . . 2246
- FACTORS Statement . . . 2247
- LOGLIN Statement . . . 2250
- MODEL Statement . . . 2251
- POPULATION Statement . . . 2258
- REPEATED Statement . . . 2261
- RESPONSE Statement . . . 2262
- RESTRICT Statement . . . 2269
- WEIGHT Statement . . . 2270

Details: CATMOD Procedure . . . **2270**
- Missing Values . . . 2270
- Input Data Sets . . . 2270
- Ordering of Populations and Responses . . . 2273
- Specification of Effects . . . 2274
- Output Data Sets . . . 2276
- Logistic Analysis . . . 2277
- Log-Linear Model Analysis . . . 2279
- Repeated Measures Analysis . . . 2282
- Generation of the Design Matrix . . . 2284
- Cautions . . . 2294
- Computational Method . . . 2297
- Computational Formulas . . . 2298
- Memory and Time Requirements . . . 2303

Displayed Output 2303
ODS Table Names 2306
Examples: CATMOD Procedure **2307**
Example 36.1: Linear Response Function, r=2 Responses 2307
Example 36.2: Mean Score Response Function, r=3 Responses 2312
Example 36.3: Logistic Regression, Standard Response Function 2316
Example 36.4: Log-Linear Model, Three Dependent Variables 2321
Example 36.5: Log-Linear Model, Structural and Sampling Zeros 2323
Example 36.6: Repeated Measures, 2 Response Levels, 3 Populations 2329
Example 36.7: Repeated Measures, 4 Response Levels, 1 Population 2332
Example 36.8: Repeated Measures, Logistic Analysis of Growth Curve 2335
Example 36.9: Repeated Measures, Two Repeated Measurement Factors 2339
Example 36.10: Direct Input of Response Functions and Covariance Matrix 2344
Example 36.11: Predicted Probabilities 2348
References **2350**

Overview: CATMOD Procedure

The CATMOD procedure performs categorical data modeling of data that can be represented by a contingency table. PROC CATMOD fits linear models to functions of response frequencies, and it can be used for linear modeling, log-linear modeling, logistic regression, and repeated measurement analysis. PROC CATMOD uses the following estimation methods:

- weighted least squares (WLS) estimation of parameters for a wide range of general linear models
- maximum likelihood (ML) estimation of parameters for log-linear models and the analysis of generalized logits

The CATMOD procedure provides a wide variety of categorical data analyses, many of which are generalizations of continuous data analysis methods. For example, analysis of variance, in the traditional sense, refers to the analysis of means and the partitioning of variation among the means into various sources. Here, the term *analysis of variance* is used in a generalized sense to denote the analysis of response functions and the partitioning of variation among those functions into various sources. The response functions might be mean scores if the dependent variables are ordinally scaled. But they can also be marginal probabilities, cumulative logits, or other functions that incorporate the essential information from the dependent variables.

NOTE: PROC CATMOD specializes in WLS modeling and analysis of a wide range of models on contingency tables. For ML modeling on standard models, especially with continuous predictors, it might be more appropriate to use a procedure such as PROC GENMOD or PROC LOGISTIC; see Chapter 51, "The GENMOD Procedure," and Chapter 79, "The LOGISTIC Procedure," for more information.

Types of Input Data

The data that PROC CATMOD analyzes are usually supplied in one of two ways. First, you can supply raw data, where each observation is a subject. Second, you can supply cell count data, where each observation is a cell in a contingency table. (A third way, which uses direct input of the covariance matrix, is also available; details are given in the section "Inputting Response Functions and Covariances Directly" on page 2272.)

Suppose detergent brand preference is related to three other categorical variables: water softness, water temperature, and previous use of a brand of detergent. In the raw data case, each observation in the input data set identifies a given respondent in the study and contains information about all four variables. The data set contains the same number of observations as the survey had respondents. In the cell count case, each observation identifies a given cell in the four-way table of water softness, water temperature, previous use of brand, and brand preference. A fifth variable contains the number of respondents in the cell. In the analysis, this fifth variable is identified in a WEIGHT statement. The data set contains the same number of observations as the number of cross-classifications formed by the four categorical variables. For more about this particular example, see Example 36.1. For additional details, see the section "Input Data Sets" on page 2270.

Most of the examples in this chapter use cell counts as input and use a WEIGHT statement.

Types of Statistical Analyses

This section illustrates, by example, the wide variety of categorical data analyses that PROC CATMOD provides. For each type of analysis, a brief description of the statistical problem and the SAS statements to provide the analysis are given. For each analysis, assume that the input data set consists of a set of cell counts from a contingency table. The variable specified in the WEIGHT statement contains these counts. In all these analyses, both the dependent and independent variables are categorical.

Linear Model Analysis

Suppose you want to analyze the relationship between the dependent variables (r1, r2) and the independent variables (a, b). Analyze the marginal probabilities of the dependent variables, and use a main-effects model:

```
proc catmod;
   weight wt;
   response marginals;
   model r1*r2=a b;
quit;
```

Log-Linear Model Analysis

Suppose you want to analyze the nominal dependent variables (r1, r2, r3) with a log-linear model. Use maximum likelihood analysis, include the main effects and the r1*r2 interaction in the model, and obtain the predicted cell frequencies:

```
proc catmod;
   weight wt;
   model r1*r2*r3=_response_ / pred=freq;
   loglin r1|r2 r3;
```

```
quit;
```

Logistic Regression

Suppose you want to analyze the relationship between the nominal dependent variable (r) and the independent variables (x1, x2) with a logistic regression analysis. Use maximum likelihood estimation:

```
proc catmod;
   weight wt;
   direct x1 x2;
   model r=x1 x2;
quit;
```

If x1 and x2 are continuous so that each observation has a unique value of these two variables, then it might be more appropriate to use the LOGISTIC or GENMOD procedure. (See the section "Logistic Regression" on page 2278.)

Repeated Measures Analysis

Suppose the dependent variables (r1, r2, r3) represent the same type of measurement taken at three different times. Analyze the relationship among the dependent variables, the repeated measurement factor (time), and the independent variable (a):

```
proc catmod;
   weight wt;
   response marginals;
   model r1*r2*r3=_response_|a;
   repeated time 3 / _response_=time;
quit;
```

Analysis of Variance

Suppose you want to investigate the relationship between the dependent variable (r) and the independent variables (a, b). Analyze the mean of the dependent variable, and include all main effects and interactions in the model:

```
proc catmod;
   weight wt;
   response mean;
   model r=a|b;
quit;
```

Linear Regression

PROC CATMOD can analyze the relationship between the dependent variables (r1, r2) and the independent variables (x1, x2). Use a linear regression analysis to analyze the marginal probabilities of the dependent variables:

```
proc catmod;
   weight wt;
   direct x1 x2;
   response marginals;
   model r1*r2=x1 x2;
```

```
quit;
```

Logistic Analysis of Ordinal Data

Suppose you want to analyze the relationship between the ordinally scaled dependent variable (r) and the independent variable (a). Use cumulative logits to take into account the ordinal nature of the dependent variable, and use weighted least squares estimation:

```
proc catmod;
   weight wt;
   response clogits;
   model r=_response_ a;
quit;
```

Sample Survey Analysis

Suppose the data set contains estimates of a vector of four functions and their covariance matrix, estimated in such a way as to correspond to the sampling process that is used. Analyze the functions with respect to the independent variables (a, b), and use a main-effects model:

```
proc catmod;
   response read b1-b10;
   model _f_=_response_;
   factors  a 2 , b 5 / _response_=a b;
quit;
```

Background: The Underlying Model

The CATMOD procedure analyzes data that can be represented by a two-dimensional contingency table. The rows of the table correspond to populations (or samples) formed on the basis of one or more independent variables. The columns of the table correspond to observed responses formed on the basis of one or more dependent variables. The frequency in the (i, j) cell is the number of subjects in the ith population that have the jth response. The frequencies in the table are assumed to follow a product multinomial distribution, corresponding to a sampling design in which a simple random sample is taken for each population. The contingency table can be represented as shown in Table 36.1.

Table 36.1 Contingency Table Representation

	Response				
Sample	**1**	**2**	$\cdots$	r	**Total**
1	n_{11}	n_{12}	$\cdots$	n_{1r}	n_1
2	n_{21}	n_{22}	$\cdots$	n_{2r}	n_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
s	n_{s1}	n_{s2}	$\cdots$	n_{sr}	n_s

For each sample i, the probability of the jth response (π_{ij}) is estimated by the sample proportion, $p_{ij} = n_{ij}/n_i$. The vector ($\mathbf{p}$) of all such proportions is then transformed into a vector of functions, denoted by

$\mathbf{F} = \mathbf{F}(\mathbf{p})$. If $\boldsymbol{\pi}$ denotes the vector of true probabilities for the entire table, then the functions of the true probabilities, denoted by $\mathbf{F}(\boldsymbol{\pi})$, are assumed to follow a linear model

$$\mathbf{E_A}(\mathbf{F}) = \mathbf{F}(\boldsymbol{\pi}) = \mathbf{X}\boldsymbol{\beta}$$

where $\mathbf{E_A}$ denotes asymptotic expectation, $\mathbf{X}$ is the design matrix containing fixed constants, and $\boldsymbol{\beta}$ is a vector of parameters to be estimated.

PROC CATMOD provides two estimation methods:

- The weighted least squares method minimizes the weighted residual sum of squares for the model. The weights are contained in the inverse covariance matrix of the functions $\mathbf{F}(\mathbf{p})$. According to central limit theory, if the sample sizes within populations are sufficiently large, the elements of $\mathbf{F}$ and $\mathbf{b}$ (the estimate of $\boldsymbol{\beta}$) are distributed approximately as multivariate normal. This allows the computation of statistics for testing the goodness of fit of the model and the significance of other sources of variation. For details of the theory, see Grizzle, Starmer, and Koch (1969) or Koch et al. (1977, Appendix 1). Weighted least squares estimation is available for all types of response functions.

- The maximum likelihood method estimates the parameters of the linear model so as to maximize the value of the joint multinomial likelihood function of the responses. Maximum likelihood estimation is available only for the standard response functions, logits and generalized logits, which are used for logistic regression analysis and log-linear model analysis. Two methods of maximization are available: Newton-Raphson and iterative proportional fitting. For details of the theory, see Bishop, Fienberg, and Holland (1975).

Following parameter estimation, hypotheses about linear combinations of the parameters can be tested. For that purpose, PROC CATMOD computes generalized Wald (1943) statistics, which are approximately chi-square distributed if the sample sizes are sufficiently large and the null hypotheses are true.

Linear Models Contrasted with Log-Linear Models

Linear model methods typified by the Grizzle, Starmer, and Koch (1969) approach make a very clear distinction between independent and dependent variables. The emphasis of these methods is estimation and hypothesis testing of the model parameters. Therefore, it is easy to test for differences among probabilities, perform repeated measures analysis, and test for marginal homogeneity, but it is difficult to test for independence and generalized independence. These methods are a natural extension of the usual ANOVA approach for continuous data.

In contrast, log-linear model methods typified by the Bishop, Fienberg, and Holland (1975) approach do not make an a priori distinction between independent and dependent variables, although model specifications that allow for the distinction can be made. The emphasis of these methods is on model building, goodness-of-fit tests, and estimation of cell frequencies or probabilities for the underlying contingency table. With these methods, it is easy to test independence and generalized independence, but it is difficult to test for differences among probabilities, do repeated measures analysis, and test for marginal homogeneity.

Using PROC CATMOD Interactively

You can use the CATMOD procedure interactively. After specifying a model with a MODEL statement and running PROC CATMOD with a RUN statement, you can execute any statement without reinvoking PROC CATMOD. You can execute the statements singly or in groups by following the single statement or group of statements with a RUN statement. Note that you can use more than one MODEL statement; this is an important difference from the GLM procedure.

If you use PROC CATMOD interactively, you can end the CATMOD procedure with a DATA step, another PROC step, an ENDSAS statement, or a QUIT statement. The syntax of the QUIT statement is as follows:

```
quit;
```

When you are using PROC CATMOD interactively, additional RUN statements do not end the procedure run but tell the procedure to execute additional statements.

When the CATMOD procedure detects a BY statement, it disables interactive processing; that is, once the BY statement and the next RUN statement are encountered, processing proceeds for each BY group in the data set, and no additional statements are accepted by the procedure. For example, the following statements perform three analyses: one for the entire data set, one for males, and one for females:

```
proc catmod;
   weight wt;
   response marginals;
   model r1*r2=a|b;
run;
   by sex;
run;
```

Note that the BY statement can appear after the first RUN statement; this is an important difference from PROC GLM, which requires that the BY statement appear before the first RUN statement.

Getting Started: CATMOD Procedure

The CATMOD procedure is a general modeling procedure for categorical data analysis, and it can be used for sophisticated analyses that require matrix specification of the response function and the design matrix. It can also be used to perform basic analysis-of-variance-type analyses that require only a few statements. The following is a basic example.

Weighted Least Squares Analysis of Mean Response

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/catgs1.sas.)

Consider the data in Table 36.2 (Stokes, Davis, and Koch 2000).

Table 36.2 Colds in Children

		Periods with Colds			
Sex	**Residence**	**0**	**1**	**2**	**Total**
Female	Rural	45	64	71	180
Female	Urban	80	104	116	300
Male	Rural	84	124	82	290
Male	Urban	106	117	87	310

For male and female children in rural and urban counties, the number of periods (of two) in which subjects report cold symptoms are recorded. So 45 subjects who are female and in rural counties report no cold symptoms, and 71 subjects who are female and from rural counties report colds in both periods.

The question of interest is whether the mean number of periods with colds reported is associated with gender or type of county. There is no reason to believe that the mean number of periods with colds is normally distributed, so a weighted least squares analysis of these data is performed with PROC CATMOD instead of an analysis of variance with PROC ANOVA or PROC GLM.

The input data for categorical data are often recorded in frequency form, with the counts for each particular profile being the input values. For the colds data, the input SAS data set colds is created with the following statements. The variable count contains the frequency of observations that have the particular profile described by the values of the other variables in that input line.

```
data colds;
   input sex $ residence $ periods count @@;
   datalines;
female rural 0  45  female rural 1  64  female rural 2  71
female urban 0  80  female urban 1 104  female urban 2 116
male   rural 0  84  male   rural 1 124  male   rural 2  82
male   urban 0 106  male   urban 1 117  male   urban 2  87
;
```

In order to fit a model to the mean number of periods with colds, you have to specify the response function in PROC CATMOD. The default response function is the logit if the response variable has two values, and it is generalized logits if the response variable has more than two values. If you want a different response function, then you specify that function in the RESPONSE statement. To request the mean number of periods with colds, you specify the MEANS option in the RESPONSE statement.

You can request a model consisting of the main effects and interaction of the variables sex and residence just as you would in the GLM procedure. Unlike the GLM procedure, PROC CATMOD does not require you to use a CLASS statement to treat a variable as a classification variable. In the CATMOD procedure, all variables in the MODEL statement are treated as classification variables unless you specify otherwise with a

DIRECT statement. To verify that your model is specified correctly, you can specify the DESIGN option in the MODEL statement to display the design matrix.

The PROC CATMOD statements needed to model mean periods of colds with a main-effects and interaction model are as follows:

```
proc catmod data=colds;
   weight count;
   response means;
   model periods = sex residence sex*residence / design;
run;
```

The results of this analysis are shown in Figure 36.1 through Figure 36.3.

In Figure 36.1, the CATMOD procedure first displays a summary of the contingency table you are analyzing. The "Population Profiles" table lists the values of the explanatory variables that define each population, or row of the underlying contingency table, and labels each group with a sample number. The number of observations in each population is also displayed. The "Response Profiles" table lists the variable levels that define the response, or columns of the underlying contingency table.

Figure 36.1 Model Information and Profile Tables

The CATMOD Procedure

Data Summary			
Response	periods	**Response Levels**	3
Weight Variable	count	**Populations**	4
Data Set	COLDS	**Total Frequency**	1080
Frequency Missing	0	**Observations**	12

Population Profiles			
Sample	**sex**	**residence**	**Sample Size**
1	female	rural	180
2	female	urban	300
3	male	rural	290
4	male	urban	310

Response Profiles	
Response	**periods**
1	0
2	1
3	2

The "Response Functions and Design Matrix" table in Figure 36.2 contains the observed response functions—in this case, the mean number of periods with colds for each of the populations—and the design matrix. The first column of the design matrix contains the coefficients for the intercept parameter. The second column contains the coefficients for the sex parameter. (Note that the sum-to-zero constraint of the default full-rank parameterization PARAM=EFFECT implies that the coefficient for males is the negative of that for females; the parameter is called the *differential effect* for females.) The third column is similarly set up for residence, and the last column is for the interaction.

Figure 36.2 Observed Response Functions and Design Matrix

Response Functions and Design Matrix					
		Design Matrix			
Sample	Response Function	1	2	3	4
1	1.14444	1	1	1	1
2	1.12000	1	1	-1	-1
3	0.99310	1	-1	1	-1
4	0.93871	1	-1	-1	1

The model-fitting results are displayed in the "Analysis of Variance" table (Figure 36.3), which is similar to an ANOVA table. The effects from the right side of the MODEL statement are listed in the Source column.

Figure 36.3 ANOVA Table for the Saturated Model

Analysis of Variance			
Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	1841.13	<.0001
sex	1	11.57	0.0007
residence	1	0.65	0.4202
sex*residence	1	0.09	0.7594
Residual	0	.	.

You can see in Figure 36.3 that the interaction effect is nonsignificant, so the data are reanalyzed using a main-effects model. Since PROC CATMOD is an interactive procedure, you can analyze the main-effects model by simply submitting the new MODEL statement as follows. The resulting tables are displayed in Figure 36.4 and Figure 36.5.

```
    model periods = sex residence / design;
run;
```

From the ANOVA table in Figure 36.4, you can see that the goodness-of-fit chi-square statistic is 0.09 with one degree of freedom and a p-value of 0.7594; hence, the model fits the data. Note that the chi-square tests in Figure 36.4 check whether all the parameters for a given effect are zero. In this model, each effect has only one parameter and therefore only one degree of freedom.

Figure 36.4 Main-Effects Model

The CATMOD Procedure

Data Summary			
Response	periods	Response Levels	3
Weight Variable	count	Populations	4
Data Set	COLDS	Total Frequency	1080
Frequency Missing	0	Observations	12

Figure 36.4 *continued*

Population Profiles			
Sample	**sex**	**residence**	**Sample Size**
1	female	rural	180
2	female	urban	300
3	male	rural	290
4	male	urban	310

Response Profiles	
Response	**periods**
1	0
2	1
3	2

Response Functions and Design Matrix				
		Design Matrix		
Sample	**Response Function**	**1**	**2**	**3**
1	1.14444	1	1	1
2	1.12000	1	1	-1
3	0.99310	1	-1	1
4	0.93871	1	-1	-1

Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	1	1882.77	<.0001
sex	1	12.08	0.0005
residence	1	0.76	0.3839
Residual	1	0.09	0.7594

The "Analysis of Weighted Least Squares Estimates" table in Figure 36.5 lists the parameters and their estimates for the model, as well as the standard errors, Wald statistics, and p-values. These chi-square tests are one-degree-of-freedom tests that the individual parameter is equal to zero. They are equal to the tests shown in Figure 36.4 since each effect is composed of exactly one parameter.

Figure 36.5 Parameter Estimates for the Main-Effects Model

Analysis of Weighted Least Squares Estimates					
Parameter		**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Intercept		1.0501	0.0242	1882.77	<.0001
sex	**female**	0.0842	0.0242	12.08	0.0005
residence	**rural**	0.0210	0.0241	0.76	0.3839

You can compute the mean number of periods with colds for the first population (Sample 1, females in rural residences) from Table 36.2 as follows:

$$\text{mean colds} = 0 \times \frac{45}{180} + 1 \times \frac{64}{180} + 2 \times \frac{71}{180} = 1.1444$$

This is the same value reported in the Response Function column for Sample 1 in the "Response Functions and Design Matrix" table displayed in Figure 36.4.

PROC CATMOD is fitting a model to the mean number of colds in each population as follows:

$$\begin{bmatrix} \text{Expected number of colds for rural females} \\ \text{urban females} \\ \text{rural males} \\ \text{urban males} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

where the design matrix is the same one displayed in Figure 36.4, β_0 is the mean number of colds averaged over all the populations, β_1 is the differential effect for females, and β_2 is the differential effect for rural residences. The parameter estimates are shown in Figure 36.5; the expected number of periods with colds for rural females from this model is computed as

$$1 \times 1.0501 + 1 \times 0.0842 + 1 \times 0.0210 = 1.1553$$

and the expected number for rural males from this model is

$$1 \times 1.0501 - 1 \times 0.0842 + 1 \times 0.0210 = 0.9869$$

Notice also, in Figure 36.5, that the differential effect for residence is nonsignificant ($p = 0.3839$). If you continue the analysis by fitting a single-effect model (sex), you need to include a POPULATION statement to maintain the same underlying contingency table:

```
   population sex residence;
   model periods = sex;
run;
```

Generalized Logits Model

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/catgs2.sas.)

Over the course of one school year, third-graders from three different schools are exposed to three different styles of mathematics instruction: a self-paced computer-learning style, a team approach, and a traditional class approach. The students are asked which style they prefer, and their responses, classified by the type of program they are in (a regular school day versus a regular school day supplemented with an afternoon school program), are displayed in Table 36.3. The data set is from Stokes, Davis, and Koch (2000), and it is also analyzed in the section "Example 79.4: Nominal Response Data: Generalized Logits Model" on page 6123 in Chapter 79, "The LOGISTIC Procedure."

Table 36.3 School Program Data

		Learning Style Preference		
School	**Program**	**Self**	**Team**	**Class**
1	Regular	10	17	26
1	Afternoon	5	12	50
2	Regular	21	17	26
2	Afternoon	16	12	36
3	Regular	15	15	16
3	Afternoon	12	12	20

The levels of the response variable (self, team, and class) have no essential ordering, so a logistic regression is performed on the generalized logits. The model to be fit is

$$\log\left(\frac{\pi_{hij}}{\pi_{hir}}\right) = \alpha_j + \mathbf{x}'_{hi}\boldsymbol{\beta}_j$$

where π_{hij} is the probability that a student in school h and program i prefers teaching style j, $j \neq r$, and style r is the class style. There are separate sets of intercept parameters α_j and regression parameters β_j for each logit, and the matrix $\mathbf{x}_{hi}$ is the set of explanatory variables for the hi population. Thus, two logits are modeled for each school and program combination (population): the logit comparing self to class and the logit comparing team to class.

The following statements create the data set school and request the analysis. Generalized logits are the default response functions, and maximum likelihood estimation is the default method for analyzing generalized logits, so only the WEIGHT and MODEL statements are required. The option ORDER=DATA means that the response variable levels are ordered as they exist in the data set: self, team, and class; the logits are formed by comparing self to class and by comparing team to class. The results of this analysis are shown in Figure 36.6 and Figure 36.7.

```
data school;
   length Program $ 9;
   input School Program $ Style $ Count @@;
   datalines;
1 regular   self 10  1 regular   team 17  1 regular   class 26
1 afternoon self  5  1 afternoon team 12  1 afternoon class 50
2 regular   self 21  2 regular   team 17  2 regular   class 26
2 afternoon self 16  2 afternoon team 12  2 afternoon class 36
3 regular   self 15  3 regular   team 15  3 regular   class 16
3 afternoon self 12  3 afternoon team 12  3 afternoon class 20
;

proc catmod order=data;
   weight Count;
   model Style=School Program School*Program;
run;
```

A summary of the data set is displayed in Figure 36.6; the variable levels that form the three responses and six populations are listed in the “Response Profiles” and “Population Profiles” tables, respectively.

Figure 36.6 Model Information and Profile Tables

The CATMOD Procedure

Data Summary			
Response	Style	**Response Levels**	3
Weight Variable	Count	**Populations**	6
Data Set	SCHOOL	**Total Frequency**	338
Frequency Missing	0	**Observations**	18

Population Profiles			
Sample	**School**	**Program**	**Sample Size**
1	1	regular	53
2	1	afternoon	67
3	2	regular	64
4	2	afternoon	64
5	3	regular	46
6	3	afternoon	44

Response Profiles	
Response	**Style**
1	self
2	team
3	class

The analysis of variance table is displayed in Figure 36.7. Since this is a saturated model, there are no degrees of freedom remaining for a likelihood ratio test, and missing values are displayed in the table. The interaction effect is clearly nonsignificant, so a main-effects model is fit.

Figure 36.7 Saturated Model: ANOVA Table

Maximum Likelihood Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	2	40.05	<.0001
School	4	14.55	0.0057
Program	2	10.48	0.0053
School*Program	4	1.74	0.7827
Likelihood Ratio	0	.	.

Since PROC CATMOD is an interactive procedure, you can analyze the main-effects model by simply submitting the new MODEL statement as follows:

```
    model Style=School Program;
  run;
```

You can check the population and response profiles (not shown) to confirm that they are the same as those in Figure 36.6. The analysis of variance table is shown in Figure 36.8. The likelihood ratio chi-square statistic is 1.78 with a p-value of 0.7766, indicating a good fit; the Wald chi-square tests for the school and program effects are also significant. Since School has three levels, two parameters are estimated for each of the two

logits they modeled, for a total of four degrees of freedom. Since Program has two levels, one parameter is estimated for each of the two logits, for a total of two degrees of freedom.

Figure 36.8 Main-Effects Model: ANOVA Table

Maximum Likelihood Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	2	39.88	<.0001
School	4	14.84	0.0050
Program	2	10.92	0.0043
Likelihood Ratio	4	1.78	0.7766

The parameter estimates and tests for individual parameters are displayed in Figure 36.9. The order of the parameters corresponds to the order of the population and response variables as shown in the profile tables (see Figure 36.6), with the levels of the response variables varying most rapidly. The first response function is the logit that compares self to class, and the corresponding parameters have Function Number=1. The second logit (Function Number=2) compares team to class. The School=1 parameters are the differential effects versus School=3 for their respective logits, and the School=2 parameters are likewise differential effects versus School=3. The Program parameters are the differential effects of 'regular' versus 'afternoon' for the two response functions.

Figure 36.9 Parameter Estimates

Analysis of Maximum Likelihood Estimates						
Parameter		**Function Number**	**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Intercept		**1**	-0.7979	0.1465	29.65	<.0001
		2	-0.6589	0.1367	23.23	<.0001
School	**1**	**1**	-0.7992	0.2198	13.22	0.0003
	1	**2**	-0.2786	0.1867	2.23	0.1356
	2	**1**	0.2836	0.1899	2.23	0.1352
	2	**2**	-0.0985	0.1892	0.27	0.6028
Program	**regular**	**1**	0.3737	0.1410	7.03	0.0080
	regular	**2**	0.3713	0.1353	7.53	0.0061

The Program variable has nearly the same effect on both logits, while School=1 has the largest effect of the schools.

Syntax: CATMOD Procedure

The following statements are available in the CATMOD procedure:

PROC CATMOD < *options* > ;
DIRECT < *variables* > ;
MODEL *response-effect* = *design-effects* < / *options* > ;
CONTRAST '*label*' *row-description* < , . . . , *row-description* > < / *options* > ;
BY *variables* ;
FACTORS *factor-description* < , . . . , *factor-description* > < / *options* > ;
LOGLIN *effects*< / *option* > ;
POPULATION *variables* ;
REPEATED *factor-description* < , . . . , *factor-description* > < / *options* > ;
RESPONSE < *function* > < / *options* > ;
RESTRICT *parameter*=*value* < . . . *parameter*=*value* > ;
WEIGHT *variable* ;

You can use all of the statements in PROC CATMOD interactively. The first RUN statement executes all of the previous statements. Any subsequent RUN statement executes only those statements that appear between the previous RUN statement and the current one. However, if you specify a BY statement, interactive processing is disabled. That is, all statements through the following RUN statement are processed for each BY group in the data set, but no additional statements are accepted by the procedure.

If more than one CONTRAST statement appears between two RUN statements, all the CONTRAST statements are processed. If more than one RESPONSE statement appears between two RUN statements, then analyses associated with each RESPONSE statement are produced. For all other statements, there can be only one occurrence of the statement between any two RUN statements. For example, if there are two LOGLIN statements between two RUN statements, the first LOGLIN statement is ignored.

The PROC CATMOD and MODEL statements are required. If specified, the DIRECT statement must precede the MODEL statement. As a result, if you use the DIRECT statement interactively, you need to specify a MODEL statement in the same RUN group. See the section "DIRECT Statement" on page 2246 for an example.

The CONTRAST statements, if any, must follow the MODEL statement.

You can specify only one of the LOGLIN, REPEATED, and FACTORS statements between any two RUN statements, because they all specify the same information: how to partition the variation among the response functions within a population.

A QUIT statement executes any statements that have not been processed and then ends the CATMOD procedure run.

The purpose of each statement, other than the PROC CATMOD statement, is summarized in the following list:

BY	determines groups in which data are to be processed separately.
CONTRAST	specifies a hypothesis to test.
DIRECT	specifies independent variables that are to be treated quantitatively (like continuous variables) rather than qualitatively (like classification or discrete variables). These variables

also help to determine the rows of the contingency table and distinguish response functions in one population from those in other populations.

FACTORS specifies (1) the factors that distinguish response functions from others in the same population and (2) model effects, based on these factors, which help to determine the design matrix.

LOGLIN specifies log-linear model effects.

MODEL specifies (1) dependent variables, which determine the columns of the contingency table, (2) independent variables, which distinguish response functions in one population from those in other populations, and (3) model effects, which determine the design matrix and the way in which total variation among the response functions is partitioned.

POPULATION specifies variables that determine the rows of the contingency table and distinguish response functions in one population from those in other populations.

REPEATED specifies (1) the repeated measurement factors that distinguish response functions from others in the same population and (2) model effects, based on these factors, which help to determine the design matrix.

RESPONSE determines the response functions that are to be modeled.

RESTRICT restricts values of parameters to the values you specify.

WEIGHT specifies a variable containing frequency counts.

PROC CATMOD Statement

PROC CATMOD < *options* > **;**

The PROC CATMOD statement invokes the CATMOD procedure. Table 36.4 summarizes the *options* available in the PROC CATMOD statement.

Table 36.4 PROC CATMOD Statement Options

Option	Description
DATA=	Names the input SAS data set
NAMELEN=	Specifies the length of effect names
NOPRINT	Suppresses the normal display of results
ORDER=	Specifies the sort order for the levels of classification variables

You can specify the following *options*.

DATA=*SAS-data-set*

names the SAS data set containing the data to be analyzed. By default, the CATMOD procedure uses the most recently created SAS data set. For details, see the section "Input Data Sets" on page 2270.

NAMELEN=*n*
specifies the length of effect names in tables and output data sets to be *n* characters long, where *n* is a value between 24 and 200. The default length is 24 characters.

NOPRINT
suppresses the normal display of results. The NOPRINT option is useful when you only want to create output data sets with the OUT= or OUTEST= option in the RESPONSE statement. A NOPRINT option is also available in the MODEL statement. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of classification variables. This affects the ordering of the populations, responses, and parameters, as well as the definitions of the parameters. The default, ORDER=INTERNAL, orders the variable levels by their unformatted values (for example, numeric order or alphabetical order).

The following table shows how PROC CATMOD interprets values of the ORDER= option.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=INTERNAL. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine dependent. See the section "Ordering of Populations and Responses" on page 2273 for more information and examples. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC CATMOD to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the CATMOD procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged

in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.

- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

You can specify one or more *variables* in the input data set on the BY statement.

When you specify a BY statement with PROC CATMOD, no further interactive processing is possible. In other words, once the BY statement appears, all statements up to the associated RUN statement are executed for each BY group in the data set. After the RUN statement, no further statements are accepted by the procedure.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CONTRAST Statement

CONTRAST '*label*' *row-description* <, . . . , *row-description*> </ *options*> ;

where a *row-description* is defined as follows:

<@*n*> *effect values* <. . . <@*n*> *effect values* >

The CONTRAST statement constructs and tests linear functions of the parameters in the MODEL statement or effects listed in the LOGLIN statement. Each set of effects (separated by commas) specifies one row or set of rows of the matrix $\mathbf{C}$ that PROC CATMOD uses to test the hypothesis $\mathbf{C}\boldsymbol{\beta} = \mathbf{0}$.

CONTRAST statements must be preceded by the MODEL statement, and by the LOGLIN statement, if one is used. You can specify the following terms in the CONTRAST statement.

'*label*' specifies up to 256 characters of identifying information displayed with the test. The '*label*' is required.

effect is one of the effects specified in the MODEL or LOGLIN statement, INTERCEPT (for the intercept parameter), or ALL_PARMS (for the complete set of parameters).

The ALL_PARMS option is regarded as an effect with the same number of parameters as the number of columns in the design matrix. This is particularly useful when the design matrix is input directly, as in the following example:

```
model y=(1 0 0 0,
         1 0 1 0,
         1 1 0 0,
         1 1 1 1);
contrast 'Main Effect of B' all_parms 0 1 0 0;
contrast 'Main Effect of C' all_parms 0 0 1 0;
contrast 'B*C Interaction ' all_parms 0 0 0 1;
```

values are numbers that form the coefficients of the parameters associated with the given effect. If there are fewer values than parameters for an effect, the remaining coefficients become zero. For example, if you specify two values and the effect actually has five parameters, the final three are set to zero.

@*n* points to the parameters in the *n*th set when the model has a separate set of parameters for each of the response functions. The @*n* notation is seldom needed. It enables you to test the variation among response functions in the same population. However, it is usually easier to model and test such variation by using the _RESPONSE_ effect in the MODEL statement or by using the ALL_PARMS designation. Usually, contrasts are performed with respect to all of the response functions, and this is what the CONTRAST statement does by default (in this case, do not use the @*n* notation).

For example, if there are three response functions per population, then the following contrast results in a three-degree-of-freedom test comparing the first two levels of A simultaneously on the three response functions.

```
contrast 'Level 1 vs. Level 2' A 1 -1 0;
```

If, however, you want to specify a contrast with respect to the parameters in the *n*th set only, then use a single @*n* in a *row-description*. For example, the following statement tests that the first parameter of A and the first parameter of B are zero in the third response function:

```
contrast 'A=0, B=0, Function 3'  @3  A 1  B 1;
```

To specify a contrast with respect to parameters in two or more different sets of effects, use @*n* with each effect. For example:

```
contrast 'Average over Functions' @1 A 1 0 -1
                                  @2 A 1 1 -2;
```

When the model does not have a separate set of parameters for each of the response functions, the @*n* notation is invalid. This type of model is called AVERAGED. For details, see the description of the AVERAGED option and the section "Generation of the Design Matrix" on page 2284.

You can specify the following *options* in the CONTRAST statement after a slash.

ALPHA=*value*

specifies the significance level of the confidence interval for each contrast when the ESTIMATE= option is specified. The default is ALPHA=0.05, resulting in a 95% confidence interval for each contrast.

ESTIMATE=*keyword*

EST=*keyword*

requests that each individual contrast (that is, each row, $c_i\boldsymbol{\beta}$, of $\mathbf{C}\boldsymbol{\beta}$) or exponentiated contrast ($\exp(c_i\boldsymbol{\beta})$) be estimated and tested. PROC CATMOD displays the point estimate, its standard error, a Wald confidence interval, and a Wald chi-square test for each contrast. The significance level of the confidence interval is controlled by the ALPHA= option.

You can estimate the contrast or the exponentiated contrast, or both, by specifying one of the following *keywords*:

PARM specifies that the contrast itself be estimated.

EXP specifies that the exponentiated contrast be estimated.

BOTH specifies that both the contrast and the exponentiated contrast be estimated.

Specifying Contrasts

PROC CATMOD is parameterized differently than PROC GLM, so you must be careful not to use the same contrasts that you would with PROC GLM. Since PROC CATMOD uses full-rank parameterizations, all estimable parameters are directly estimable without involving other parameters.

For example, suppose a classification variable A has four levels and uses the default parameterization (PARAM=EFFECT). Then there are four parameters ($\alpha_1, \alpha_2, \alpha_3, \alpha_4$), of which PROC CATMOD uses only the first three. The fourth parameter is related to the others by the equation

$$\alpha_4 = -\alpha_1 - \alpha_2 - \alpha_3$$

To test the first versus the fourth level of A, you would test $\alpha_1 = \alpha_4$, which is

$$\alpha_1 = -\alpha_1 - \alpha_2 - \alpha_3$$

or, equivalently,

$$2\alpha_1 + \alpha_2 + \alpha_3 = 0$$

Therefore, you would use the following CONTRAST statement:

```
contrast '1 vs. 4' A 2 1 1;
```

To contrast the third level with the average of the first two levels, you would test

$$\frac{\alpha_1 + \alpha_2}{2} = \alpha_3$$

or, equivalently,

$$\alpha_1 + \alpha_2 - 2\alpha_3 = 0$$

Therefore, you would use the following CONTRAST statement:

```
contrast '1&2 vs. 3' A 1 1 -2;
```

Other CONTRAST statements are constructed similarly. For example:

```
contrast '1 vs. 2    '  A  1 -1  0;
contrast '1&2 vs. 4  '  A  3  3  2;
contrast '1&2 vs. 3&4'  A  2  2  0;
contrast 'Main Effect'  A  1  0  0,
                        A  0  1  0,
                        A  0  0  1;
```

The actual form of the **C** matrix depends on the effects in the model. The remaining examples in this section assume a single response function for each population.

Recall that the statements to test the first versus the fourth level of A are as follows:

```
proc catmod;
   model y=a;
   contrast '1 vs. 4' A 2 1 1;
run;
```

Since the first parameter corresponds to the intercept, the **C** matrix for the preceding statements is

$$\mathbf{C} = [\,0\ 2\ 1\ 1\,]$$

But suppose you have a variable B with three levels and you use the following statements:

```
proc catmod;
   model y=b a;
   contrast '1 vs. 4' A 2 1 1;
run;
```

Then the CONTRAST statement produces the **C** matrix

$$\mathbf{C} = [\,0\;0\;0\;2\;1\;1\,]$$

since the first parameter corresponds to the intercept and the next two correspond to the B main effect.

You can also use the CONTRAST statement to test the joint effect of two or more effects in the MODEL statement. For example, the joint effect of A and B in the previous model has five degrees of freedom and is obtained by specifying the following:

```
contrast 'Joint Effect of A&B' A 1 0 0,
                               A 0 1 0,
                               A 0 0 1,
                               B 1 0,
                               B 0 1;
```

The ordering of variable levels is determined by the ORDER= option in the PROC CATMOD statement. Whenever you specify a contrast that depends on the order of the variable levels, you should verify the order from the "Population Profiles" table, the "Response Profiles" table, or the "One-Way Frequencies" table.

DIRECT Statement

DIRECT *variables* **;**

The DIRECT statement lists numeric independent variables to be treated in a quantitative, rather than qualitative, way. The DIRECT statement is useful for logistic regression, which is described in the section "Logistic Regression" on page 2278. For limitations of models involving continuous variables, see the section "Continuous Variables" on page 2279.

CAUTION: If a DIRECT variable is formatted, then the unformatted (internal) values are used in the analysis and the formatted values are displayed. If you use a format to group the internal values into one formatted value, then the first internal value is used in the analysis. If specified, the DIRECT statement must precede the MODEL statement. For example:

```
proc catmod;
   direct X;
   model Y=X;
run;
```

Suppose X has five levels. Then the main effect X adds only one column to the design matrix rather than four. The values inserted into the design matrix are the actual values of X.

You can interactively change the variables declared as DIRECT variables by using the statement without listing any variables. The following statements are valid:

```
proc catmod;
   direct X;
   model Y=X;
   weight wt;
run;
   direct;
   model Y=X;
run;
```

The first MODEL statement uses the actual values of X, and the second MODEL statement uses the four variables created when PROC CATMOD generates the design matrix. Note that the preceding statements can be run without a WEIGHT statement if the input data are raw data rather than cell counts.

For more details, see the discussions of main and direct effects in the section "Generation of the Design Matrix" on page 2284.

FACTORS Statement

FACTORS *factor-description* < **,** . . . **,** *factor-description* > < / *options* > **;**

where a *factor-description* is defined as follows:

factor-name < *$* > < *levels* >

and *factor-description*s are separated from each other by a comma. The $ is required for character-valued factors. The value of *levels* provides the number of levels of the factor identified by a given *factor-name*. For only one factor, *levels* is optional; for two or more factors, it is required.

The FACTORS statement identifies factors that distinguish response functions from others in the same population. It also specifies how those factors are incorporated into the model. You can use the FACTORS statement whenever there is more than one response function per population and the keyword _RESPONSE_ is specified in the MODEL statement. You can specify the name, type, and number of levels of each factor and the identification of each level.

The FACTORS statement is most useful when the response functions and their covariance matrix are read directly from the input data set. In this case, PROC CATMOD reads the response functions as though they are from one population (this poses no problem in the multiple-population case because the appropriately constructed covariance matrix is also read directly). Thus, you can use the FACTORS statement to partition the variation among the response functions into appropriate sources, even when the functions actually represent separate populations.

The format of the FACTORS statement is identical to that of the REPEATED statement. In fact, repeated measurement factors are simply special cases of factors in which some of the response functions correspond to multiple dependent variables that are measurements on the same experimental (or sampling) units.

You cannot specify the FACTORS statement for an analysis that also contains the REPEATED or LOGLIN statement since all of them specify the same information: how to partition the variation among the response functions within a population.

You can specify the following terms in the FACTORS statement:

factor-name names a factor that corresponds to two or more response functions. This name must be a valid SAS variable name, and it should not be the same as the name of a variable that already exists in the data set being analyzed.

$ indicates that the factor is character-valued. If the $ is omitted, then the CATMOD procedure assumes that the factor is numeric. The type of the factor is relevant only when you use the PROFILE= option or when the _RESPONSE_= option (described later in this section) specifies nested-by-value effects.

levels specifies the number of levels of the corresponding factor. If there is only one such factor, and the number is omitted, then PROC CATMOD assumes that the number of levels is equal to the number of response functions per population (q). Unless you specify the PROFILE= option, the number q must either be equal to or be a multiple of the product of the number of levels of all the factors.

You can specify the following *options* in the FACTORS statement after a slash.

PROFILE=(*matrix*)

specifies the values assumed by the factors for each response function. There should be one column for each factor, and the values in a given column (character or numeric) should match the type of the corresponding factor. Character values are restricted to 16 characters or less. If there are q response functions per population, then the matrix must have i rows, where q must either be equal to or be a multiple of i. Adjacent rows of the matrix should be separated by a comma.

The values in the PROFILE matrix are useful for specifying models in those situations where the study design is not a full factorial with respect to the factors. They can also be used to specify nested-by-value effects in the _RESPONSE_= option. If you specify character values in both places (the PROFILE= option and the _RESPONSE_= option), then the values must match with respect to whether or not they are enclosed in quotes (that is, enclosed in quotes in both places or in neither place).

For an example of using the PROFILE= option, see Example 36.10.

RESPONSE=*effects*

specifies design effects. The variables named in the effects must be *factor-names* that appear in the FACTORS statement. If the _RESPONSE_= option is omitted, then PROC CATMOD builds a full factorial _RESPONSE_ effect with respect to the factors.

TITLE=*'title'*

displays the *title* at the top of certain pages of output that correspond to the current FACTORS statement.

For an example of how the FACTORS statement is useful, consider the case where the response functions and their covariance matrix are read directly from the input data set. The TYPE=EST data set might be created in the following manner:

```
data direct(type=est);
   input b1-b4 _type_ $ _name_ $8.;
   datalines;
0.590463   0.384720   0.273269   0.136458   parms     .
0.001690   0.000911   0.000474   0.000432   cov       b1
0.000911   0.001823   0.000031   0.000102   cov       b2
0.000474   0.000031   0.001056   0.000477   cov       b3
0.000432   0.000102   0.000477   0.000396   cov       b4
```

```
;
```

Suppose the response functions correspond to four populations that represent the cross-classification of age (two groups) by sex. You can use the FACTORS statement to identify these two factors and to name the effects in the model. The statements required to fit a main-effects model to these data are as follows:

```
proc catmod data=direct;
   response read b1-b4;
   model _f_=_response_;
   factors age 2, sex 2 / _response_=age sex;
run;
```

If you want to specify some nested-by-value effects, you can change the FACTORS statement to the following:

```
factors age $ 2, sex $ 2 /
       _response_=age sex(age='under 30') sex(age='30 & over')
        profile=('under 30'   male,
                 'under 30'   female,
                 '30 & over'  male,
                 '30 & over'  female);
```

If, by design or by chance, the study contains no male subjects under 30 years of age, then there are only three response functions, and you can specify a main-effects model as follows:

```
proc catmod data=direct;
   response read b2-b4;
   model _f_=_response_;
   factors age $ 2, sex $ 2 / _response_=age sex
          profile=('under 30'   female,
                   '30 & over'  male,
                   '30 & over'  female);
run;
```

When you specify two or more factors and omit the PROFILE= option, PROC CATMOD presumes that the response functions are ordered so that the levels of the rightmost factor change most rapidly. For the preceding example, the order implied by the FACTORS statement is as follows:

Response Function	Dependent Variable	Age	Sex
1	b1	1	1
2	b2	1	2
3	b3	2	1
4	b4	2	2

For additional examples of how to use the FACTORS statement, see the section "Repeated Measures Analysis" on page 2282. All of the examples in that section are applicable, with the REPEATED statement replaced by the FACTORS statement.

LOGLIN Statement

LOGLIN *effects* < / *option* > **;**

The LOGLIN statement is used to define log-linear model effects. It can be used whenever the default response functions (generalized logits) are used.

In the LOGLIN statement, *effects* are design effects that contain dependent variables in the MODEL statement, including interaction, nested, and nested-by-value effects. You can use the bar (|) and at (@) operators as well. The following lists of effects are equivalent:

```
a b c a*b a*c b*c
```

and

```
a|b|c @2
```

When you use the LOGLIN statement, the keyword _RESPONSE_ should be specified in the MODEL statement. For further information about log-linear model analysis, see the section "Log-Linear Model Analysis" on page 2279.

You cannot specify the LOGLIN statement for an analysis that also contains the REPEATED or FACTORS statement since all of them specify the same information: how to partition the variation among the response functions within a population.

You can specify the following *option* in the LOGLIN statement after a slash.

TITLE=*'title'*

displays the *title* at the top of certain pages of output that correspond to this LOGLIN statement.

The following statements give an example of how to use the LOGLIN statement:

```
proc catmod;
   model a*b*c=_response_;
   loglin a|b|c @ 2;
run;
```

These statements yield a log-linear model analysis that contains all main effects and two-variable interactions. For more examples of log-linear model analysis, see the section "Log-Linear Model Analysis" on page 2279.

MODEL Statement

MODEL *response-effect* **=** < *design-effects* > < / *options* > **;**

PROC CATMOD requires a MODEL statement. You can specify the following in a MODEL statement:

response-effect can be either a single variable, a crossed effect with two or more variables joined by asterisks, or _F_. The _F_ specification indicates that the response functions and their estimated covariance matrix are to be read directly into the procedure (see the section "Inputting Response Functions and Covariances Directly" on page 2272 for details). The *response-effect* indicates the dependent variables that determine the response categories (the columns of the underlying contingency table).

design-effects specify potential sources of variation (such as main effects and interactions) in the model. These effects determine the number of model parameters, as well as the interpretation of such parameters. In addition, if there is no POPULATION statement, PROC CATMOD uses these variables to determine the populations (the rows of the underlying contingency table). When fitting the model, PROC CATMOD adjusts the independent effects in the model for all other independent effects in the model.

Design-effects can be any of those described in the section "Specification of Effects" on page 2274, or they can be defined by specifying the actual design matrix, enclosed in parentheses (see the section "Specifying the Design Matrix Directly" on page 2257). In addition, you can use the keyword _RESPONSE_ alone or as part of an effect. Effects cannot be nested within _RESPONSE_, so effects of the form A(_RESPONSE_) are invalid.

For more information, see the section "Log-Linear Model Analysis" on page 2279 and the section "Repeated Measures Analysis" on page 2282.

Some example MODEL statements are shown in the following table:

Example	Result
`model r=a b;`	Main effects only
`model r=a b a*b;`	Main effects with interaction
`model r=a b(a);`	Nested effect
`model r=a|b;`	Complete factorial
`model r=a b(a=1) b(a=2);`	Nested-by-value effects
`model r*s=_response_;`	Log-linear model
`model r*s=a _response_(a);`	Nested repeated measurement factor
`model _f_=_response_;`	Direct input of the response functions

The relationship between these specifications and the structure of the design matrix **X** is described in the section "Generation of the Design Matrix" on page 2284.

Table 36.5 summarizes the *options* available in the MODEL statement.

Table 36.5 MODEL Statement Options

Options	Task
Specify Details of Computation	
ML=	Generates the maximum likelihood estimates
GLS	Generates the weighted least squares estimates
WLS	
NOINT	Omits the intercept term from the model
PARAM=	Specifies the parameterization of classification variables
ADDCELL=	Adds a number to each cell frequency
AVERAGED	Averages the main effects across response functions
EPSILON=	Specifies the convergence criterion for maximum likelihood
MAXITER=	Specifies the number of iterations for maximum likelihood
MISSING=	Specifies how missing cells are treated
ZERO=	Specifies how zero cells are treated
Request Additional Computation and Tables	
ALPHA=	Specifies the significance level of confidence intervals
CLPARM	Displays the Wald confidence intervals of estimates
CORRB	Displays the estimated correlation matrix of estimates
COV	Displays the covariance matrix of response functions
COVB	Displays the estimated covariance matrix of estimates
DESIGN	Displays the design and _RESPONSE_ matrix
FREQ	Displays the two-way frequency tables
ITPRINT	Displays the iterations for maximum likelihood
ONEWAY	Displays the one-way frequency tables
PRED=	Displays the predicted values
PREDICT	
PROB	Displays the probability estimates
PROFILE	Displays the population profiles
XPX	Displays the crossproducts matrix
TITLE=	Specifies the title
Suppress Output	
NODESIGN	Suppresses the design matrix
NOPARM	Suppresses the parameter estimates
NOPREDVAR	Suppresses the variable levels
NOPROFILE	Suppresses the population and response profiles
NORESPONSE	Suppresses the _RESPONSE_ matrix

The following list describes these *options* in alphabetical order.

ADDCELL=*number*

adds *number* to the frequency count in each cell, where *number* is any positive number. This option has no effect on maximum likelihood analysis; it is used only for weighted least squares analysis.

ALPHA=*number*

sets the significance level for the Wald confidence intervals for parameter estimates. The value must be between 0 and 1. The default value of 0.05 results in the calculation of a 95% confidence interval. This option has no effect unless the CLPARM option is also specified.

AVERAGED

specifies that dependent variable effects can be modeled and that independent variable main effects are averaged across the response functions in a population. For further information about the effect of using (or not using) the AVERAGED option, see the section "Generation of the Design Matrix" on page 2284. Direct input of the design matrix or specification of the _RESPONSE_ keyword in the MODEL statement automatically uses an AVERAGED model type.

CLPARM

produces Wald confidence limits for the parameter estimates. The confidence coefficient can be specified with the ALPHA= option.

CORRB

displays the estimated correlation matrix of the parameter estimates.

COV

displays $\mathbf{S}_i$, which is the covariance matrix of the response functions for each population.

COVB

displays the estimated covariance matrix of the parameter estimates.

DESIGN

displays the design matrix $\mathbf{X}$ for WLS and ML analyses, and also displays the _RESPONSE_ matrix for log-linear models. For further information, see the section "Generation of the Design Matrix" on page 2284.

EPSILON=*number*

specifies the convergence criterion for the maximum likelihood estimation of the parameters. The iterative estimation process stops when the proportional change in the log likelihood is less than *number*, or after the number of iterations specified by the MAXITER= option, whichever comes first. By default, EPSILON=1E–8.

FREQ

produces the two-way frequency table for the cross-classification of populations by responses.

ITPRINT

displays parameter estimates and other information at each iteration of a maximum likelihood analysis.

MAXITER=*number*

specifies the maximum number of iterations used for the maximum likelihood estimation of the parameters. By default, MAXITER=20.

ML < = NR | IPF< (*ipf-options*) > >

computes maximum likelihood estimates (MLE) by using either a Newton-Raphson algorithm (NR) or an iterative proportional fitting algorithm (IPF).

The option ML=NR (or simply ML) is available when you use generalized logits, and also when you perform binary logistic regression with logits, cumulative logits, or adjacent category logits. For generalized logits (the default response functions), ML=NR is the default estimation method.

The option ML=IPF is available for fitting a hierarchical log-linear model with one population (no independent variables and no population variables). The use of bar notation to express the log-linear effects guarantees that the model is hierarchical (the presence of any interaction term in the model requires the presence of all its lower-order terms). If your table is *incomplete* (that is, your table has a zero or missing entry in at least one cell), then all missing cells and all cells with zero weight are treated as structural zeros by default; this behavior can be modified with the ZERO= and MISSING= options in the MODEL statement.

You can control the convergence of the two algorithms with the EPSILON= and MAXITER= options in the MODEL statement. You can select the convergence criterion for the IPF algorithm with the CONVCRIT= option.

NOTE: The RESTRICT statement is not available with the ML=IPF option.

You can specify the following *ipf-options* within parentheses after the ML=IPF option.

CONVCRIT=*keyword*

specifies the method that determines when convergence of the IPF algorithm occurs. You can specify one of the following *keywords*:

CELL termination requires the maximum absolute difference between consecutive cell estimates to be less than 0.001 (or the value of the EPSILON= option, if specified).

LOGL termination requires the relative difference between consecutive estimates of the log likelihood to be less than 1E–8 (or the value of the EPSILON= option, if specified). This is the default.

MARGIN termination requires the maximum absolute difference between consecutive margin estimates to be less than 0.001 (or the value of the EPSILON= option, if specified).

DF=*keyword*

specifies the method used to compute the degrees of freedom for the goodness-of-fit G^2 test (labeled "Likelihood Ratio" in the "Estimates" table).

For a *complete* table (a table having nonzero entries in every cell), the degrees of freedom are calculated as the number of cells in the table (n_c) minus the number of independent parameters specified in the model (n_p). For incomplete tables, these degrees of freedom can be adjusted by the number of fitted zeros (n_z, which includes the number of structural zeros) and the number of nonestimable parameters due to the zeros (n_n). If you are analyzing an incomplete table, you should verify that the degrees of freedom are correct.

You can specify one of the following *keywords*:

UNADJ computes the unadjusted degrees of freedom as $n_c - n_p$. These are the same degrees of freedom you would get if all cells in the table were positive.

ADJ computes the degrees of freedom as $(n_c - n_p) - (n_z - n_n)$ (Bishop, Fienberg, and Holland 1975), which adjusts for fitted zeros and nonestimable parameters. This is the default, and for complete tables it gives the same results as the UNADJ option.

ADJEST computes the degrees of freedom as $(n_c - n_p) - n_z$, which adjusts for fitted zeros only. This gives a lower bound on the true degrees of freedom.

PARM

computes parameter estimates, generates the "ANOVA," "Parameter Estimates," and "Predicted Values of Response Functions" tables, and includes the predicted standard errors in the "Predicted Values of Frequencies and Probabilities" tables.

When you specify the PARM option, the algorithm used to obtain the maximum likelihood parameter estimates is weighted least squares on the IPF-predicted frequencies. This algorithm can be much faster than the Newton-Raphson algorithm that is used if you specify the ML=NR option. In the resulting ANOVA table, the likelihood ratio is computed from the initial IPF fit while the degrees of freedom are generated from the WLS analysis; the DF= option can override this. Also, the initial response function, which the WLS method usually computes from the raw data, is computed from the IPF-predicted frequencies.

If there are any zero marginals in the configurations that define the model, then there are predicted cell frequencies of zero and WLS cannot be used to compute the estimates. In this case, PROC CATMOD automatically changes the algorithm from ML=IPF to ML=NR and prints a note in the log.

MISSING=*keyword*

MISS=*keyword*

specifies whether a missing cell is treated as a sampling or structural zero.

Structural zero cells are removed from the analysis since their expected values are zero, while sampling zero cells can have nonzero expected value and might be estimable. For a single population, the missing cells are treated as structural zeros by default. For multiple populations, as long as some population has a nonzero count for a given population and response profile, the missing values are treated as sampling zeros by default.

The following table displays the available *keywords* and summarizes how PROC CATMOD treats missing values for one or more populations:

MISSING=	One Population	Multiple Populations
STRUCTURAL (default)	Structural zeros	Sampling zeros
SAMP | **SAMPLING**	Sampling zeros	Sampling zeros
value	Sets missing weights and cells to *value*	Sets missing weights and cells to *value*

NODESIGN

suppresses the display of the design matrix **X** when the DESIGN option is also specified. This enables you to display only the _RESPONSE_ matrix for log-linear models.

NOINT

suppresses the intercept term in the model.

NOPARM

suppresses the display of the estimated parameters and the statistics for testing that each parameter is zero.

NOPREDVAR

suppresses the display of the variable levels in tables requested with the PRED= option and in the "Estimates" table. Population profiles are replaced with the sample number, classification variable levels are suppressed, and response profiles are replaced with a function number.

NOPRINT

suppresses the normal display of results. The NOPRINT option is useful when you only want to create output data sets with the OUT= or OUTEST= option in the RESPONSE statement. A NOPRINT option is also available in the PROC CATMOD statement. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

NOPROFILE

suppresses the display of the population profiles and the response profiles.

NORESPONSE

suppresses the display of the _RESPONSE_ matrix for log-linear models when the DESIGN option is also specified. This enables you to display only the design matrix for log-linear models.

ONEWAY

produces a one-way table of frequencies for each variable used in the analysis. This table is useful in determining the order of the observed levels for each variable.

PARAM=EFFECT | REFERENCE

specifies the parameterization method for the classification variable or variables. The default is PARAM=EFFECT. Both the effect and reference parameterizations are full rank. See the section "Generation of the Design Matrix" on page 2284 for further details.

PREDICT

PRED=FREQ | PROB

displays the observed and predicted values of the response functions for each population, together with their standard errors and the residuals (observed minus predicted). In addition, if the response functions are the standard ones (generalized logits), then the PRED=FREQ option specifies the computation and display of predicted cell frequencies, while PRED=PROB (or just PREDICT) specifies the computation and display of predicted cell probabilities.

The OUT= data set always contains the predicted probabilities. If the response functions are the generalized logits, the predicted cell probabilities are output unless the option PRED=FREQ is specified, in which case the predicted cell frequencies are output.

PROB
produces the two-way table of probability estimates for the cross-classification of populations by responses. These estimates sum to one across the response categories for each population.

PROFILE
displays all of the population profiles. If you have more than 60 populations, then by default only the first 40 profiles are displayed; the PROFILE option overrides this default behavior.

TITLE='*title***'**
displays the *title* at the top of certain pages of output that correspond to this MODEL statement.

WLS
GLS
computes weighted least squares estimates. This type of estimation is also called generalized least squares estimation. For response functions other than the default (of generalized logits), WLS is the default estimation method.

XPX
displays $\mathbf{X}'\mathbf{S}^{-1}\mathbf{X}$, the crossproducts matrix for the normal equations.

ZERO=*keyword*
specifies whether a nonmissing cell with zero weight in the data set is treated as a sampling or structural zero.

Structural zero cells are removed from the analysis since their expected values are zero, while sampling zero cells have nonzero expected value and might be estimable. For a single population, the zero cells are treated as structural zeros by default; with multiple populations, as long as some population has a nonzero count for a given population and response profile, the zeros are treated as sampling zeros by default.

The following table displays the available *keywords* and summarizes how PROC CATMOD treats zeros for one or more populations:

ZERO=	One Population	Multiple Populations
STRUCTURAL (default)	Structural zeros	Sampling zeros
SAMP | **SAMPLING**	Sampling zeros	Sampling zeros
value	Sets zero weights to *value*	Sets zero weights to *value*

Specifying the Design Matrix Directly

If you specify the design matrix directly, adjacent rows of the matrix must be separated by a comma, and the matrix must have $q \times s$ rows, where s is the number of populations and q is the number of response functions per population. The first q rows correspond to the response functions for the first population, the second set of q rows corresponds to the functions for the second population, and so forth. The following is an example of using direct specification of the design matrix.

```
proc catmod;
   model R=(1 0,
            1 1,
            1 2,
            1 3);
```

```
run;
```

These statements are appropriate for the case of one population and for R with five levels (generating four response functions), so that $4 \times 1 = 4$. These statements are also appropriate for a situation with two populations and two response functions per population, giving $2 \times 2 = 4$ rows of the design matrix. (To accommodate more than one population, the POPULATION statement is needed.)

When you input the design matrix directly, you also have the option of specifying that any subsets of the parameters be tested for equality to zero. Indicate each subset by specifying the appropriate column numbers of the design matrix, followed by an equal sign and a label (24 characters or less, in quotes) that describes the subset. Adjacent subsets are separated by a comma, and the entire specification is enclosed in parentheses and placed after the design matrix. For example:

```
proc catmod;
   population Group Time;
   model R=(1  1  0  0,
            1  1  0  1,
            1  1  0  2,
            1  0  1  0,
            1  0  1  1,
            1  0  1  2,
            1 -1 -1  0,
            1 -1 -1  1,
            1 -1 -1  2) (1  ='Intercept',
                         2 3='Group main effect',
                         4  ='Linear effect of Time');
run;
```

The preceding statements are appropriate when Group and Time each have three levels and R is dichotomous. The POPULATION statement produces nine populations, and $q = 1$ (since R is dichotomous), so $q \times s = 1 \times 9 = 9$.

If you input the design matrix directly but do not specify any subsets of the parameters to be tested, then PROC CATMOD tests the effect of MODEL | MEAN, which represents the significance of the model beyond what is explained by an overall mean. For the previous example, the MODEL | MEAN effect is the same as that obtained by specifying the following at the end of the MODEL statement:

```
(2 3 4='model|mean');
```

POPULATION Statement

POPULATION *variables* **;**

The POPULATION statement specifies that populations are to be based only on cross-classifications of the specified *variables*. If you do not specify the POPULATION statement, then populations are based only on cross-classifications of the independent variables in the MODEL statement.

The POPULATION statement has two major uses:

- When you enter the design matrix directly, there are no independent variables in the MODEL statement; therefore, the POPULATION statement is the only way to produce more than one population.

- When you fit a reduced model, the POPULATION statement might be necessary if you want to form the same number of populations as there are for the saturated model.

To illustrate the first use, suppose you specify the following statements:

```
data one;
   input A $ B $ wt @@;
   datalines;
yes yes 23   yes no 31   no yes 47   no no 50
;

proc catmod;
   weight wt;
   population B;
   model A=(1 0,
            1 1);
run;
```

Since the dependent variable A has two levels, there is one response function per population. Since the variable B has two levels, there are two populations. The MODEL statement is valid since the number of rows in the design matrix (2) is the same as the total number of response functions. If the POPULATION statement is omitted, there would be only one population and one response function, and the MODEL statement would be invalid.

To illustrate the second use, suppose you specify the following statements:

```
data two;
   input A $ B $ Y wt @@;
   datalines;
yes  yes  1  23       yes  yes  2  63
yes  no   1  31       yes  no   2  70
no   yes  1  47       no   yes  2  80
no   no   1  50       no   no   2  84
;

proc catmod;
   weight wt;
   model Y=A B A*B / wls;
run;
```

These statements form four populations and produce the following design matrix and analysis of variance table:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	48.10	<.0001
A	1	3.47	0.0625
B	1	0.25	0.6186
A*B	1	0.19	0.6638
Residual	0		

Since the B and A*B effects are nonsignificant ($p > 0.10$), fit the reduced model that contains only the A effect:

```
proc catmod;
   weight wt;
   model Y=A / wls;
run;
```

Now only two populations are formed, and the design matrix and the analysis of variance table are as follows:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	47.94	<.0001
A	1	3.33	0.0678
Residual	0		

However, you can form four populations by adding the POPULATION statement to the analysis:

```
proc catmod;
   weight wt;
   population A B;
   model Y=A / wls;
run;
```

The design matrix and the analysis of variance table resulting from these statements are as follows:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & -1 \\ 1 & -1 \end{bmatrix}$$

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	47.76	<.0001
A	1	3.30	0.0694
Residual	2	0.35	0.8374

The advantage of the latter analysis is that it retains four populations for the reduced model, thereby creating a built-in goodness-of-fit test: the residual chi-square. Such a test is important because the cumulative (or joint) effect of deleting two or more effects from the model can be significant, even if the individual effects are not.

The resulting differences between the two analyses are due to the fact that the latter analysis uses pure weighted least squares estimates with respect to the four populations that are actually sampled. The former analysis pools populations and therefore uses parameter estimates that can be regarded as weighted least squares estimates of maximum likelihood predicted cell frequencies. In any case, the estimation methods are asymptotically equivalent; therefore, the results are very similar. If you specify the ML option (instead of the WLS option) in the preceding MODEL statements, then the parameter estimates are identical for the two analyses.

CAUTION: If your model has different covariate profiles within any population, then the first profile is used in the analysis.

REPEATED Statement

REPEATED *factor-description* < **,** . . . **,** *factor-description* > < / *options* > **;**

where a *factor-description* is defined as follows:

factor-name < *$* > < *levels* >

and *factor-description*s are separated from each other by a comma. The $ is required for character-valued factors. The value of *levels* provides the number of levels of the repeated measurement factor identified by a given *factor-name*. For only one repeated measurement factor, *levels* is optional; for two or more repeated measurement factors, it is required. The REPEATED statement incorporates repeated measurement factors into the model. You can use this statement whenever there is more than one dependent variable and the keyword _RESPONSE_ is specified in the MODEL statement. If the dependent variables correspond to one or more repeated measurement factors, you can use the REPEATED statement to define _RESPONSE_ in terms of those factors. You can specify the name, type, and number of levels of each factor, as well as the identification of each level.

You cannot specify the REPEATED statement for an analysis that also contains the FACTORS or LOGLIN statement since all of them specify the same information: how to partition the variation among the response functions within a population.

You can specify the following terms in the REPEATED statement:

factor-name names a repeated measurement factor that corresponds to two or more response functions. This name must be a valid SAS variable name, and it should not be the same as the name of a variable that already exists in the data set being analyzed.

$ indicates that the factor is character-valued. If the $ is omitted, then the CATMOD procedure assumes that the factor is numeric. The type of the factor is relevant only when you use the PROFILE= option or when the _RESPONSE_= option specifies nested-by-value effects.

levels specifies the number of levels of the corresponding repeated measurement factor. If there is only one such factor and the number is omitted, then PROC CATMOD assumes that the number of levels is equal to the number of response functions per population (q). Unless you specify the PROFILE= option, the number q must either be equal to or be a multiple of the product of the number of levels of all the factors.

You can specify the following *options* in the REPEATED statement after a slash.

PROFILE=(*matrix***)**

specifies the values assumed by the factors for each response function. There should be one column for each factor, and the values in a given column should match the type (character or numeric) of the corresponding factor. Character values are restricted to 16 characters or less. If there are q response functions per population, then the matrix must have i rows, where q must either be equal to or be a multiple of i. Adjacent rows of the matrix should be separated by a comma.

The values in the PROFILE matrix are useful for specifying models in those situations where the study design is not a full factorial with respect to the factors. They can also be used to specify nested-with-value effects in the _RESPONSE_= option. If you specify character values in both the PROFILE= option and the _RESPONSE_= option, then the values must match with respect to whether or not they are enclosed in quotes (that is, they must be enclosed in quotes in both places or in neither place).

RESPONSE=*effects*

specifies design effects. The variables named in the effects must be *factor-names* that appear in the REPEATED statement. If the _RESPONSE_= option is omitted, then PROC CATMOD builds a full factorial _RESPONSE_ effect with respect to the repeated measurement factors. For example, the following two statements are equivalent in that they produce the same parameter estimates:

```
repeated Time 2, Treatment 2;
repeated Time 2, Treatment 2 / _response_=Time|Treatment;
```

However, the second statement produces tests of the Time, Treatment, and Time*Treatment effects in the "Analysis of Variance" table, whereas the first statement produces a single test for the combined effects in _RESPONSE_.

TITLE=*'title'*

displays the *title* at the top of certain pages of output that correspond to this REPEATED statement.

For further information and numerous examples of the REPEATED statement, see the section "Repeated Measures Analysis" on page 2282.

RESPONSE Statement

RESPONSE < *function* > < / *options* > **;**

The RESPONSE statement specifies functions of the response probabilities. The procedure models these response functions as linear combinations of the parameters.

By default, PROC CATMOD uses the standard response functions (generalized logits, which are explained in detail in the section "Understanding the Standard Response Functions" on page 2269). With these standard response functions, the default estimation method is maximum likelihood, but you can use the WLS option in the MODEL statement to request weighted least squares estimation. With other response functions (specified in the RESPONSE statement), the default (and only) estimation method is weighted least squares.

You can specify more than one RESPONSE statement, in which case each RESPONSE statement produces a separate analysis. If the computed response functions for any population are linearly dependent (yielding a singular covariance matrix), then PROC CATMOD displays an error message and stops processing. See the section "Cautions" on page 2294 for methods of dealing with this.

The *function* specification can be any of the items in the following list. For an example of response functions generated and formulas for q (the number of response functions), see the section "More on Response Functions" on page 2264.

Table 36.6 summarizes the *options* available in the RESPONSE statement.

Table 36.6 RESPONSE Statement Options

Option	Description
ALOGIT	Specifies response functions as adjacent-category logits
CLOGIT	Specifies that the response functions are cumulative logits
JOINT	Specifies that the response functions are the joint response probabilities

Table 36.6 *continued*

Option	Description
LOGIT	Specifies that the response functions are generalized logits
MARGINAL	Specifies that the response functions are marginal probabilities
MEAN	Specifies that the response functions are the means dependent variables
READ	Directly reads the response functions and their covariance matrix from the input data set
OUT=	Produces a SAS data set that contains predicted values, standard errors and residuals
OUTEST=	Produces a SAS data set that contains the estimated parameter vector and its estimated covariance matrix
TITLE=	Displays the *title*

ALOGIT

ALOGITS

specifies response functions as adjacent-category logits of the marginal probabilities for each of the dependent variables. For each dependent variable, the response functions are a set of linearly independent adjacent-category logits, obtained by taking the logarithms of the ratios of two probabilities. The denominator of the kth ratio is the marginal probability corresponding to the kth level of the variable, and the numerator is the marginal probability corresponding to the $(k + 1)$ level. If a dependent variable has two levels, then the adjacent-category logit is the negative of the generalized logit.

CLOGIT

CLOGITS

specifies that the response functions are cumulative logits of the marginal probabilities for each of the dependent variables. For each dependent variable, the response functions are a set of linearly independent cumulative logits, obtained by taking the logarithms of the ratios of two probabilities. The denominator of the kth ratio is the cumulative probability, c_k, corresponding to the kth level of the variable, and the numerator is $1 - c_k$ (Agresti 1984, 113–114). If a dependent variable has two levels, then PROC CATMOD computes its cumulative logit as the negative of its generalized logit. You should use cumulative logits only when the dependent variables are ordinally scaled.

JOINT

specifies that the response functions are the joint response probabilities. A linearly independent set is created by deleting the last response probability. For the case of one dependent variable, the JOINT and MARGINALS specifications are equivalent.

LOGIT

LOGITS

specifies that the response functions are generalized logits of the marginal probabilities for each of the dependent variables. For each dependent variable, the response functions are a set of linearly independent generalized logits, obtained by taking the logarithms of the ratios of two probabilities. The denominator of each ratio is the marginal probability corresponding to the last observed level of the variable, and the numerators are the marginal probabilities corresponding to each of the other levels.

If there is one dependent variable, then specifying LOGIT is equivalent to using the standard response functions.

MARGINAL
MARGINALS
specifies that the response functions are marginal probabilities for each of the dependent variables in the MODEL statement. For each dependent variable, the response functions are a set of linearly independent marginals, obtained by deleting the marginal probability corresponding to the last level.

MEAN
MEANS
specifies that the response functions are the means of the dependent variables in the MODEL statement. This specification requires that all of the dependent variables be numeric.

READ *variables*
specifies that the response functions and their covariance matrix are to be read directly from the input data set with one response function for each variable named. See the section "Inputting Response Functions and Covariances Directly" on page 2272 for more information.

transformation
specifies response functions that can be expressed by using successive applications of the four operations: **LOG**, **EXP**, $*$ matrix literal, or $+$ matrix literal. The operations are described in detail in the section "Using a Transformation to Specify Response Functions" on page 2267.

You can specify the following *options* in the RESPONSE statement after a slash.

OUT=*SAS-data-set*
produces a SAS data set that contains, for each population, the observed and predicted values of the response functions, their standard errors, and the residuals. Moreover, if you use the standard response functions, the data set also includes observed and predicted values of the cell frequencies or the cell probabilities. For further information, see the section "Output Data Sets" on page 2276.

OUTEST=*SAS-data-set*
produces a SAS data set that contains the estimated parameter vector and its estimated covariance matrix. For further information, see the section "Output Data Sets" on page 2276.

TITLE=*'title'*
displays the *title* at the top of certain pages of output that correspond to this RESPONSE statement.

More on Response Functions

Suppose the dependent variable A has three levels and is the only *response-effect* in the MODEL statement. The following table shows the proportions upon which the response functions are defined:

Value of A:	1	2	3
Proportions:	p_1	p_2	p_3

Note that $\sum_j p_j = 1$. The following table shows the response functions generated for each population:

Function Specification	Value of q	Response Function
none*	2	$\ln\left(\frac{p_1}{p_3}\right),\ \ln\left(\frac{p_2}{p_3}\right)$
ALOGITS	2	$\ln\left(\frac{p_2}{p_1}\right),\ \ln\left(\frac{p_3}{p_2}\right)$
CLOGITS	2	$\ln\left(\frac{1-p_1}{p_1}\right),\ \ln\left(\frac{1-(p_1+p_2)}{p_1+p_2}\right)$
JOINT	2	$p_1,\ p_2$
LOGITS	2	$\ln\left(\frac{p_1}{p_3}\right),\ \ln\left(\frac{p_2}{p_3}\right)$
MARGINAL	2	$p_1,\ p_2$
MEAN	1	$1p_1 + 2p_2 + 3p_3$

*Without a function specification, the default response functions are generalized logits.

Now, suppose the dependent variables A and B each have three levels (valued 1, 2, and 3 each) and the *response-effect* in the MODEL statement is A*B. The following table shows the proportions upon which the response functions are defined:

Value of A:	1	1	1	2	2	2	3	3	3
Value of B:	1	2	3	1	2	3	1	2	3
Proportions:	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9

The marginal totals for the preceding table are defined as follows:

$$
\begin{aligned}
p_{1\cdot} &= p_1 + p_2 + p_3 & \qquad p_{\cdot 1} &= p_1 + p_4 + p_7 \\
p_{2\cdot} &= p_4 + p_5 + p_6 & \qquad p_{\cdot 2} &= p_2 + p_5 + p_8 \\
p_{3\cdot} &= p_7 + p_8 + p_9 & \qquad p_{\cdot 3} &= p_3 + p_6 + p_9
\end{aligned}
$$

where $\sum_j p_j = 1$. The following table shows the response functions generated for each population:

Function Specification	Value of q	Response Function
none*	8	$\ln\left(\frac{p_1}{p_9}\right),\ \ln\left(\frac{p_2}{p_9}\right),\ \ln\left(\frac{p_3}{p_9}\right), \ldots,\ \ln\left(\frac{p_8}{p_9}\right)$
ALOGITS	4	$\ln\left(\frac{p_{2\cdot}}{p_{1\cdot}}\right),\ \ln\left(\frac{p_{3\cdot}}{p_{2\cdot}}\right),\ \ln\left(\frac{p_{\cdot 2}}{p_{\cdot 1}}\right),\ \ln\left(\frac{p_{\cdot 3}}{p_{\cdot 2}}\right)$
CLOGITS	4	$\ln\left(\frac{1-p_{1\cdot}}{p_{1\cdot}}\right),\ \ln\left(\frac{1-(p_{1\cdot}+p_{2\cdot})}{p_{1\cdot}+p_{2\cdot}}\right),\ \ln\left(\frac{1-p_{\cdot 1}}{p_{\cdot 1}}\right),\ \ln\left(\frac{1-(p_{\cdot 1}+p_{\cdot 2})}{p_{\cdot 1}+p_{\cdot 2}}\right)$
JOINT	8	$p_1,\ p_2,\ p_3,\ p_4,\ p_5,\ p_6,\ p_7,\ p_8$
LOGITS	4	$\ln\left(\frac{p_{1\cdot}}{p_{3\cdot}}\right),\ \ln\left(\frac{p_{2\cdot}}{p_{3\cdot}}\right),\ \ln\left(\frac{p_{\cdot 1}}{p_{\cdot 3}}\right),\ \ln\left(\frac{p_{\cdot 2}}{p_{\cdot 3}}\right)$
MARGINAL	4	$p_{1\cdot},\ p_{2\cdot},\ p_{\cdot 1},\ p_{\cdot 2}$
MEAN	2	$1p_{1\cdot} + 2p_{2\cdot} + 3p_{3\cdot},\ 1p_{\cdot 1} + 2p_{\cdot 2} + 3p_{\cdot 3}$

* Without a function specification, the default response functions are generalized logits.

The READ and *transformation* function specifications are not shown in the preceding table. For these two situations, there is not a general response function; the response functions that are generated depend on what you specify.

Another important aspect of the function specification is the number of response functions generated per population, q. Let m_i represent the number of levels for the ith dependent variable in the MODEL statement, and let d represent the number of dependent variables in the MODEL statement. Then, if the function specification is ALOGITS, CLOGITS, LOGITS, or MARGINALS, the number of response functions is

$$q = \sum_{i=1}^{d} (m_i - 1)$$

If the function specification is JOINT or the default (generalized logits), the number of response functions per population is

$$q = r - 1$$

where r is the number of response profiles. If every possible cross-classification of the dependent variables is observed in the samples, then

$$r = \prod_{i=1}^{d} m_i$$

Otherwise, r is the number of cross-classifications actually observed.

If the function specification is MEANS, the number of response functions per population is $q = d$.

Response Statement Examples

Some example response statements are shown in the following table:

Example	Result
`response marginals;`	Marginals for each dependent variable
`response means;`	The mean of each dependent variable
`response logits;`	Generalized logits of the marginal probabilities
`response clogits;`	Cumulative logits of the marginal probabilities
`response alogits;`	Adjacent-category logits of the marginal probabilities
`response joint;`	The joint probabilities
`response 1 -1 log;`	The logit
`response;`	Generalized logits
`response 1 2 3;`	The mean score, with scores of 1, 2, and 3 corresponding to the three response levels
`response read b1-b4;`	Four response functions and their covariance matrix, read directly from the input data set

Using a Transformation to Specify Response Functions

If you specify a *transformation*, it is applied to the vector that contains the sample proportions in each population. The *transformation* can be any combination of the following four operations:

Operation	Specification
Linear combination	$*$ Matrix literal
Linear combination	Matrix literal
Logarithm	**LOG**
Exponential	**EXP**
Adding constant	$+$ Matrix literal

If more than one operation is specified, then PROC CATMOD applies the operations consecutively from right to left.

A matrix literal is a matrix of numbers with each row of the matrix separated from the next by a comma. If you specify a linear combination, in most cases the $*$ is not needed. The following statement defines the response function $p_1 + 1$. The $*$ is needed to separate the two matrix literals '1' and '1 0'.

```
response + 1 * 1 0;
```

The **LOG** of a vector transforms each element of the vector into its natural logarithm; the **EXP** of a vector transforms each element into its exponential function (antilogarithm).

In order to specify a linear response function for data that have $r = 3$ response categories, you can specify either of the following RESPONSE statements:

```
response  * 1 0 0 , 0 1 0;
response    1 0 0 , 0 1 0;
```

The matrix literal in the preceding statements specifies a 2×3 matrix, which is applied to each population as follows:

$$\begin{bmatrix} F_1 \\ F_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} * \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

where p_1, p_2, and p_3 are sample proportions for the three response categories in a population, and F_1 and F_2 are the two response functions computed for that population. Therefore, this response function sets $F1 = p_1$ and $F2 = p_2$ in each population.

As another example of the linear response function, suppose you have two dependent variables corresponding to two observers who evaluate the same subjects. If the observers grade on the same three-point scale and if all nine possible responses are observed, then the following RESPONSE statement would compute the probability that the observers agree on their assessments:

```
response 1 0 0 0 1 0 0 0 1;
```

This response function is then computed as

$$F = p_{11} + p_{22} + p_{33} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ p_{21} \\ p_{22} \\ p_{23} \\ p_{31} \\ p_{32} \\ p_{33} \end{bmatrix}$$

where p_{ij} denotes the probability that a subject gets a grade of i from the first observer and j from the second observer.

If the function is a compound function, requiring more than one operation to specify it, then the operations should be listed so that the first operation to be applied is on the right and the last operation to be applied is on the left. For example, if there are two response levels, you can have the following response function:

```
response 1 -1 log;
```

This is equivalent to the matrix expression

$$F = \begin{bmatrix} 1 & -1 \end{bmatrix} * \begin{bmatrix} \log(p_1) \\ \log(p_2) \end{bmatrix} = \log(p_1) - \log(p_2) = \log\left(\frac{p_1}{p_2}\right)$$

which is the logit response function since $p_2 = 1 - p_1$ when there are only two response levels.

The following statement specifies another example of a compound response function:

```
response exp 1 -1 * 1 0 0 1, 0 1 1 0 log;
```

This is equivalent to the matrix expression

$$F = \mathbf{EXP}(\mathbf{A} * \mathbf{B} * \mathbf{LOG}(\mathbf{P}))$$

where $\mathbf{P}$ is the vector of sample proportions for some population,

$$\mathbf{A} = \begin{bmatrix} 1 & -1 \end{bmatrix} \text{ and } \mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

If the four responses are based on two dependent variables, each with two levels, then the function can also be written as

$$F = \frac{p_{11}p_{22}}{p_{12}p_{21}}$$

which is the odds (crossproduct) ratio for a 2×2 table.

Understanding the Standard Response Functions

If no RESPONSE statement is specified, PROC CATMOD computes the standard response functions, which contrast the log of each response probability with the log of the probability for the last response category. If there are r response categories, then there are $r - 1$ standard response functions. For example, if there are four response categories, using no RESPONSE statement is equivalent to specifying the following:

```
response  1 0 0 -1,
          0 1 0 -1,
          0 0 1 -1  log;
```

This results in three response functions:

$$F = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} \log(p_1/p_4) \\ \log(p_2/p_4) \\ \log(p_3/p_4) \end{bmatrix}$$

If there are only two response levels, the resulting response function would be a logit, which is why the standard response functions are called generalized logits. They are useful in dealing with the log-linear model:

$$\boldsymbol{\pi} = \mathbf{EXP}(\mathbf{X}\boldsymbol{\beta})$$

If $\mathbf{C}$ denotes the matrix in the preceding RESPONSE statement, then because of the restriction that the probabilities sum to 1, it follows that an equivalent model is

$$\mathbf{C} * \mathbf{LOG}(\boldsymbol{\pi}) = (\mathbf{CX})\boldsymbol{\beta}$$

But $\mathbf{C} * \mathbf{LOG}(\mathbf{P})$ is simply the vector of standard response functions. Thus, fitting a log-linear model on the cell probabilities is equivalent to fitting a linear model on the generalized logits.

RESTRICT Statement

RESTRICT *parameter*=*value* < . . . *parameter*=*value* > **;**

where *parameter* is the letter B followed by a number; for example, B3 specifies the third parameter in the model. The *value* is the value to which the parameter is restricted. The RESTRICT statement restricts values of parameters to the values you specify, so that the estimation of the remaining parameters is subject to these restrictions. Consider the following statement:

```
restrict b1=1 b4=0 b6=0;
```

This restricts the values of three parameters. The first parameter is set to 1, and the fourth and sixth parameters are set to zero.

The RESTRICT statement is interactive. A new RESTRICT statement replaces any previous ones. In addition, if you submit two or more MODEL, LOGLIN, FACTORS, or REPEATED statements, then the subsequent occurrences of these statements also delete the previous RESTRICT statement.

WEIGHT Statement

WEIGHT *variable* **;**

You can use a WEIGHT statement to refer to a variable containing the cell frequencies, which do not need to be integers. The WEIGHT statement lets you use summary data sets containing a count variable. See the section "Input Data Sets" on page 2270 for further information about the WEIGHT statement.

Details: CATMOD Procedure

Missing Values

Observations with missing values for any variable listed in the MODEL or POPULATION statement are omitted from the analysis.

If the WEIGHT variable for an observation has a missing value, the observation is by default omitted from the analysis. You can modify this behavior by specifying the MISSING= option in the MODEL statement. The option MISSING=*value* sets all missing weights to *value* and all missing cells to *value*. The option MISSING=SAMPLING causes all missing cells in a contingency table to be treated as sampling zeros.

Any observation with nonpositive weight is also, by default, omitted from the analysis. If it has zero weight, then you can specify the ZERO= option in the MODEL statement.

Input Data Sets

Data to be analyzed by PROC CATMOD must be in a SAS data set containing one of the following:

- raw data values (variable values for every subject)
- frequency counts and the corresponding variable values
- response function values and their covariance matrix

If you specify a WEIGHT statement, then PROC CATMOD uses the values of the WEIGHT variable as the frequency counts. If the READ function is specified in the RESPONSE statement, then the procedure expects the input data set to contain the values of response functions and their covariance matrix. Otherwise, PROC CATMOD assumes that the SAS data set contains raw data values.

Raw Data Values

If you use raw data, PROC CATMOD first counts the number of observations having each combination of values for all variables specified in the MODEL or POPULATION statement. For example, suppose the variables A and B each take on the values 1 and 2, and their frequencies can be represented as follows:

		A	
		1	**2**
B	**1**	2	1
	2	3	1

The SAS data set Raw containing the raw data might be as follows:

Observation	A	B
1	1	1
2	1	1
3	1	2
4	1	2
5	1	2
6	2	1
7	2	2

And the statements for PROC CATMOD are as follows:

```
proc catmod data=Raw;
   model A=B;
run;
```

For discussions of how to handle structural and random zeros with raw data as input data, see the section "Zero Frequencies" on page 2295 and Example 36.5.

Frequency Counts

If your data set contains frequency counts, then use the WEIGHT statement to specify the variable containing the frequencies. For example, you could create and analyze the Summary data set as follows:

```
data Summary;
   input A B Count;
   datalines;
1 1 2
1 2 3
2 1 1
2 2 1
;

proc catmod data=Summary;
   weight Count;
   model A=B;
run;
```

The data set Summary can also be created from the data set Raw by using the FREQ procedure:

```
proc freq data=Raw;
   tables A*B / out=Summary;
run;
```

Inputting Response Functions and Covariances Directly

If you want to read in the response functions and their covariance matrix, rather than have PROC CATMOD compute them, create a TYPE=EST data set. In addition to having one variable name for each function, the data set should have two additional variables: _TYPE_ and _NAME_, both character variables of length 8. The variable _TYPE_ should have the value 'PARMS' when the observation contains the response functions; it should have the value 'COV' when the observation contains elements of the covariance matrix of the response functions. The variable _NAME_ is used only when _TYPE_=COV, in which case it should contain the name of the variable that has its covariance elements stored in that observation. In the following data set, for example, the covariance between the second and fourth response functions is 0.000102:

```
data direct(type=est);
   input b1-b4 _type_ $ _name_ $8.;
   datalines;
0.590463   0.384720   0.273269   0.136458   PARMS     .
0.001690   0.000911   0.000474   0.000432   COV       B1
0.000911   0.001823   0.000031   0.000102   COV       B2
0.000474   0.000031   0.001056   0.000477   COV       B3
0.000432   0.000102   0.000477   0.000396   COV       B4
;
```

In order to tell PROC CATMOD that the input data set contains the values of response functions and their covariance matrix, do the following:

- specify the READ function in the RESPONSE statement
- specify _F_ as the dependent variable in the MODEL statement

For example, suppose the response functions correspond to four populations that represent the cross-classification of two age groups by two race groups. You can use the FACTORS statement to identify these two factors and to name the effects in the model. The following statements are required to fit a main-effects model to these data:

```
proc catmod data=direct;
   response read b1-b4;
   model _f_=_response_;
   factors age 2, race 2 / _response_=age race;
run;
```

Ordering of Populations and Responses

By default, populations and responses are sorted in standard SAS order as follows:

- alphabetical order for character variables
- increasing numeric order for numeric variables

Suppose you specify the following statements:

```
data one;
   length A B $ 6;
   input A $ B $ wt @@;
   datalines;
low       low  23  low    medium  31 low    high  38
medium    low  40  medium medium  42 medium high  50
high      low  52  high   medium  54 high   high  61
;

proc catmod;
   weight wt;
   model A=B / oneway;
run;
```

The ordering of populations and responses corresponds to the alphabetical order of the levels of the character variables. You can specify the ONEWAY option to display the ordering of the variables, while the "Population Profiles" and "Response Profiles" tables display the ordering of the populations and the responses, respectively.

Population Profiles		Response Profiles	
Sample	**B**	**Response**	**A**
1	high	1	high
2	low	2	low
3	medium	3	medium

In this example, if you want to have the levels ordered in the natural order of 'low,' 'medium,' 'high,' you can specify the ORDER=DATA option:

```
proc catmod order=data;
   weight wt;
   model a=b / oneway;
run;
```

The resulting ordering of populations and responses is as follows:

Population Profiles		Response Profiles	
Sample	**B**	**Response**	**A**
1	low	1	low
2	medium	2	medium
3	high	3	high

You can use the ORDER=DATA option to ensure that populations and responses are ordered in a specific way. But since this also affects the definitions and the ordering of the parameters, you must exercise caution when using the _RESPONSE_ effect, the CONTRAST statement, or direct input of the design matrix.

An alternative method of ensuring that populations and responses are ordered in a specific way is to assign a format to your variables and specify the ORDER=FORMATTED option. The levels are then ordered according to their formatted values.

Another method is to replace any character variables with numeric variables and to assign formatted values such as 'yes' and 'no' to the numeric levels. Since ORDER=INTERNAL is the default ordering, PROC CATMOD orders the populations and responses according to the numeric values but displays the formatted values.

Specification of Effects

By default, the CATMOD procedure treats all variables as classification variables. As a result, there is no CLASS statement in PROC CATMOD. The values of a classification variable can be numeric or character. PROC CATMOD builds a set of effects-coded variables to represent the levels of the classification variable and then uses these to fit the model (for details, see the section "Generation of the Design Matrix" on page 2284). You can modify the default by using the DIRECT statement to treat numeric independent continuous variables as continuous variables. The classification variables, combinations of classification variables, and continuous variables are then used in fitting linear models to data.

The parameters of a linear model are generally divided into subsets that correspond to meaningful sources of variation in the response functions. These sources, called *effects*, can be specified in the MODEL, LOGLIN, FACTORS, REPEATED, and CONTRAST statements. Effects can be specified in any of the following ways:

- A main effect is a single classification variable (that is, it produces class levels): A B C.
- A crossed effect (or interaction) is two or more classification variables joined by asterisks—for example: A*B A*B*C.
- A nested effect is a main effect or an interaction, followed by a parenthetical field containing a main effect or an interaction. Multiple variables within the parentheses are assumed to form a crossed effect even when the asterisk is absent. In the following list, the last two effects are identical: B(A) C(A*B) A*B(C*D) A*B(C D).
- A nested-by-value effect is the same as a nested effect except that any variable in the parentheses can be followed by an equal sign and a value: B(A=1) C(A B=1) C*D(A=1 B=1) A(C='low').
- A direct effect is a variable specified in a DIRECT statement: X Y.
- Direct effects can be crossed with other effects: X*Y X*X*X X*A*B(C D=1).

The variables for crossed and nested effects remain in the order in which they are first encountered. For example, in the following model, the effect A*B is reported as B*A since B appears before A in the statement:

```
model R=B A A*B C(A B);
```

Also, C(A B) is interpreted as C(A*B) and is therefore reported as C(B*A).

Bar Notation

You can shorten the specification of multiple effects by using bar notation. For example, the following statements illustrate two methods of writing a full three-way factorial model:

```
proc catmod;
   model y=a b c a*b a*c b*c a*b*c;
run;

proc catmod;
   model y=a|b|c;
run;
```

When you use the bar (|) notation, the right and left sides become effects, and the interaction between them becomes an effect. Multiple bars are permitted. The expressions are expanded from left to right, using rules 1 through 4 given in Searle (1971, p. 390):

- Multiple bars are evaluated left to right. For example, A|B|C is evaluated as follows:
  A | B | C → {A | B} | C
  → {A B A*B} | C
  → A B A*B C A*C B*C A*B*C
- Crossed and nested groups of variables are combined. For example, A(B) | C(D) generates A*C(B D), among other terms.
- Duplicate variables are removed. For example, A(C) | B(C) generates A*B(C C), among other terms, and the extra C is removed.
- Effects are discarded if a variable occurs on both the crossed and nested sides of an effect. For instance, A(B) | B(D E) generates A*B(B D E), but this effect is deleted.

You can also specify the maximum number of variables involved in any effect that results from bar evaluation by specifying that maximum number, preceded by an @ sign, at the end of the bar effect. For example, the specification A | B | C @ 2 would result in only those effects that contain two or fewer variables; in this case, the effects A, B, A*B, C, A*C, and B*C are generated.

Other examples of the bar notation follow:

A | C(B)	is equivalent to	A C(B) A*C(B)
A(B) | C(B)	is equivalent to	A(B) C(B) A*C(B)
A(B) | B(D E)	is equivalent to	A(B) B(D E)
A | B(A) | C	is equivalent to	A B(A) C A*C B*C(A)
A | B(A) | C@2	is equivalent to	A B(A) C A*C
A | B | C | D@2	is equivalent to	A B A*B C A*C B*C D A*D B*D C*D

For details about how the effects specified lead to a design matrix, see the section "Generation of the Design Matrix" on page 2284.

Output Data Sets

OUT= Data Set

For each population, the OUT= data set contains the observed and predicted values of the response functions, their standard errors, the residuals, and variables that describe the population and response profiles. In addition, if you use the standard response functions, the data set includes observed and predicted values for the cell frequencies or the cell probabilities, together with their standard errors and residuals.

Number of Observations

For the standard response functions, there are $s \times (2q - 1)$ observations in the data set for each BY group, where s is the number of populations and q is the number of response functions per population. Otherwise, there are $s \times q$ observations in the data set for each BY group.

Variables in the OUT= Data Set

The data set contains the following variables:

BY variables	If you use a BY statement, the BY variables are included in the OUT= data set.
dependent variables	If the response functions are the default ones (generalized logits), then the dependent variables, which describe the response profiles, are included in the OUT= data set. When _TYPE_=FUNCTION, the values of these variables are missing.
independent variables	The independent variables, which describe the population profiles, are included in the OUT= data set.
NUMBER	the sequence number of the response function or the cell probability or the cell frequency
OBS	the observed value
PRED	the predicted value
RESID	the residual (observed minus predicted)
SAMPLE	the population number. This matches the sample number in the "Population Profile" section of the output.
SEOBS	the standard error of the observed value
SEPRED	the standard error of the predicted value
TYPE	specifies a character variable with three possible values. When _TYPE_=FUNCTION, the observed and predicted values are values of the response functions. When _TYPE_=PROB, they are values of the cell probabilities. When _TYPE_=FREQ, they are values of the cell frequencies. Cell probabilities or frequencies are provided only when the default response functions are modeled. In this case, cell probabilities are provided by default, and cell frequencies are provided if you specify the option PRED=FREQ.

OUTEST= Data Set

This TYPE=EST output data set contains the estimated parameter vector and its estimated covariance matrix. If you specify both the ML and WLS options in the MODEL statement, the OUTEST= data set contains both sets of estimates. For each BY group, there are $p + 1$ observations in the data set for each estimation method, where p is the number of estimated parameters. The data set contains the following variables:

B1, B2, and so on	variables for the estimated parameters. The OUTEST= data set contains one variable for each estimated parameter.
BY variables	If you use a BY statement, the BY variables are included in the OUT= data set.
METHOD	the method used to obtain parameter estimates. For weighted least squares estimation, _METHOD_=WLS, and for maximum likelihood estimation, _METHOD_=ML.
NAME	identifies parameter names. When _TYPE_=PARMS, _NAME_ is blank, but when _TYPE_=COV, _NAME_ has one of the values B1, B2, and so on, corresponding to the parameter names.
STATUS	indicates whether the estimates have converged
TYPE	identifies the statistics contained in the variables for parameter estimates (B1, B2, and so on). When _TYPE_=PARMS, the variables contain parameter estimates; when _TYPE_=COV, they contain covariance estimates.

The variables _METHOD_, _NAME_, and _TYPE_ are character variables; the BY variables can be either character or numeric; and the variables for estimated parameters are numeric.

See Appendix A, "Special SAS Data Sets," for more information about special SAS data sets.

Logistic Analysis

In a logistic analysis, the response functions are the logits of the dependent variable.

PROC CATMOD can compute the three following types of logits with the use of *keywords* in the RESPONSE statement. Note that other types of response functions can be generated by specifying appropriate transformations in the RESPONSE statement.

- Generalized logits are used primarily for nominally scaled dependent variables, but they can also be used for ordinal data modeling. Maximum likelihood estimation is available for the analysis of these logits.
- Cumulative logits are used for ordinally scaled dependent variables. Except for dependent variables with two response levels, only weighted least squares estimation is available for the analysis of these logits.
- Adjacent-category logits are equivalent to generalized logits, but they have some advantages for ordinal data analysis because they automatically incorporate integer scores for the levels of the dependent variable. Except for dependent variables with two response levels, only weighted least squares estimation is available for the analysis of these logits.

If the dependent variable has only two responses, then the cumulative logit and the adjacent-category logit are the negative of the generalized logit, as computed by PROC CATMOD. Consequently, parameter estimates obtained by using these logits are the negative of those obtained from using generalized logits. A simple logistic analysis of variance uses statements like the following:

```
proc catmod;
   model r=a|b;
run;
```

Logistic Regression

If the independent variables are treated quantitatively (like continuous variables), then a logistic analysis is known as a *logistic regression*. If you want PROC CATMOD to treat the independent variables as quantitative variables, specify them in both the DIRECT and MODEL statements, as follows:

```
proc catmod;
   direct x1 x2 x3;
   model r=x1 x2 x3;
run;
```

Since the preceding statements do not include a RESPONSE statement, generalized logits are computed. See Example 36.3 for another example.

The parameter estimates from the CATMOD procedure are the same as those from a logistic regression program such as PROC LOGISTIC (see Chapter 79, "The LOGISTIC Procedure"). The chi-square statistics and the predicted values are also identical. In the binary response case, PROC CATMOD can be made to model the probability of the maximum value by either (1) organizing the input data so that the maximum value occurs first and specifying ORDER=DATA in the PROC CATMOD statement or (2) specifying cumulative logits (CLOGITS) in the RESPONSE statement.

CAUTION: Computational difficulties might occur if you use a continuous variable with a large number of unique values in a DIRECT statement. See the section "Continuous Variables" on page 2279 for more details.

Cumulative Logits

If your dependent variable is ordinally scaled, you can specify the analysis of cumulative logits that take into account the ordinal nature of the dependent variable:

```
proc catmod;
   response clogits;
   direct x;
   model r=a x;
run;
```

The preceding statements correspond to a simple analysis that addresses the question of existence of an association between the independent variables and the ordinal dependent variable. However, there are some commonly used models for the analysis of ordinal data (Agresti 1984) that address the structure of association (in terms of odds ratios), as well as its existence.

If the independent variables are classification variables, a typical analysis for such a model uses the following statements:

```
proc catmod;
   weight wt;
   response clogits;
   model r=_response_ a b;
run;
```

On the other hand, if the independent variables are ordinally scaled, you might specify numeric scores in variables x1 and x2, and use the following statements:

```
proc catmod;
   weight wt;
   direct x1 x2;
   response clogits;
   model r=_response_ x1 x2;
run;
```

See Agresti (1984) for additional details of estimation, testing, and interpretation.

Continuous Variables

Computational difficulties might occur if you have a continuous variable with a large number of unique values and you use this variable in a DIRECT statement, since an observation often represents a separate population of size one. At this extreme of sparseness, the weighted least squares method is inappropriate since there are too many zero frequencies. Therefore, you should use the maximum likelihood method. PROC CATMOD is not designed optimally for continuous variables; therefore, it might be less efficient and unable to allocate sufficient memory to handle this problem, as compared with a procedure designed specifically to handle continuous data. In these situations, consider using the LOGISTIC or GENMOD procedure to analyze your data.

Log-Linear Model Analysis

When the response functions are the default generalized logits, then inclusion of the keyword _RESPONSE_ in every effect in the right side of the MODEL statement fits a log-linear model. The keyword _RESPONSE_ tells PROC CATMOD that you want to model the variation among the dependent variables. You then specify the actual model in the LOGLIN statement.

When you perform log-linear model analysis, you can request weighted least squares estimates, maximum likelihood estimates, or both. By default, PROC CATMOD calculates maximum likelihood estimates when the default response functions are used. The following table provides appropriate MODEL statements for the combinations of types of estimates:

Estimation Desired	MODEL Statement
Maximum likelihood (Newton-Raphson)	`model a*b=_response_;`
Maximum likelihood (Iterative Proportional Fitting)	`model a*b=_response_ / ml=ipf;`
Weighted least squares	`model a*b=_response_ / wls;`
Maximum likelihood and weighted least squares	`model a*b=_response_ / wls ml;`

CAUTION: Sampling zeros in the input data set should be specified with the ZERO= option to ensure that these sampling zeros are not treated as structural zeros. Alternatively, you can replace cell counts for sampling zeros with some positive number close to zero (such as 1E–20) in a DATA step. Data containing sampling zeros should be analyzed with maximum likelihood estimation. See the section "Cautions" on page 2294 and Example 36.5 for further information and an illustration that uses both cell count data and raw data.

One Population

The usual log-linear model analysis has one population, which means that all of the variables are dependent variables. For example, the following statements yield a maximum likelihood analysis of a saturated log-linear model for the dependent variables r1 and r2:

```
proc catmod;
   weight wt;
   model r1*r2=_response_;
   loglin r1|r2;
run;
```

If you want to fit a reduced model with respect to the dependent variables (for example, a model of independence or conditional independence), specify the reduced model in the LOGLIN statement. For example, the following statements yield a main-effects log-linear model analysis of the factors r1 and r2:

```
proc catmod;
   weight wt;
   model r1*r2=_response_ / pred;
   loglin r1 r2;
run;
```

The output includes Wald statistics for the individual effects r1 and r2, as well as predicted cell probabilities. Moreover, the goodness-of-fit statistic is the likelihood ratio test for the hypothesis of independence between r1 and r2 or, equivalently, a test of r1*r2.

Multiple Populations

You can do log-linear model analysis with multiple populations by using a POPULATION statement or by including effects on the right side of the MODEL statement that contain independent variables. Each effect must include the _RESPONSE_ keyword.

For example, suppose the dependent variables r1 and r2 are dichotomous, and the independent variable group has three levels. Then the following statements specify a saturated model (three degrees of freedom for _RESPONSE_ and six degrees of freedom for the interaction between _RESPONSE_ and group):

```
proc catmod;
   weight wt;
   model r1*r2=_response_ group*_response_;
   loglin r1|r2;
run;
```

From another point of view, _RESPONSE_*group can be regarded as a main effect for group with respect to the three response functions, while _RESPONSE_ can be regarded as an intercept effect with respect to the functions. In other words, the following statements give essentially the same results as the logistic analysis:

```
proc catmod;
   weight wt;
   model r1*r2=group;
run;
```

The ability to model the interaction between the independent and the dependent variables becomes particularly useful when a reduced model is specified for the dependent variables. For example, the following statements specify a model with two degrees of freedom for _RESPONSE_ (one for r1 and one for r2) and four degrees of freedom for the interaction of _RESPONSE_*group:

```
proc catmod;
   weight wt;
   model r1*r2=_response_ group*_response_;
   loglin r1 r2;
run;
```

The likelihood ratio goodness-of-fit statistic (three degrees of freedom) tests the hypothesis that r1 and r2 are independent in each of the three groups.

Iterative Proportional Fitting

You can use the iterative proportional fitting (IPF) algorithm to fit a hierarchical log-linear model with no independent variables and no population variables.

The advantage of IPF over the Newton-Raphson (NR) algorithm and over the weighted least squares (WLS) method is that, when the contingency table has several dimensions and the parameter vector is large, you can obtain the log likelihood, the goodness-of-fit G^2, and the predicted frequencies or probabilities without performing potentially expensive parameter estimation and covariance matrix calculations. This enables you to do the following:

- compare two models by computing the likelihood ratio statistics to test the significance of the contribution of the variables in one model that are not in the other model
- compute predicted values of the cell probabilities or frequencies for the final model

Each iteration of the IPF algorithm is generally faster than an iteration of the NR algorithm; however, the IPF algorithm converges to the MLEs more slowly than the NR algorithm. Both NR and WLS are more general methods that are able to perform more complex analyses than IPF can.

Repeated Measures Analysis

If there are multiple dependent variables and the variables represent repeated measurements of the same observational unit, then the variation among the dependent variables can be attributed to one or more repeated measurement factors. The factors can be included in the model by specifying _RESPONSE_ on the right side of the MODEL statement and by using a REPEATED statement to identify the factors.

To perform a repeated measures analysis, you also need to specify a RESPONSE statement, since the standard response functions (generalized logits) cannot be used. Typically, the MEANS or MARGINALS response functions are specified in a repeated measures analysis, but other response functions can also be reasonable.

One Population

Consider an experiment in which each subject is measured at three times, and the response functions are marginal probabilities for each of the dependent variables. If the dependent variables each have k levels, then PROC CATMOD computes $k-1$ response functions for each time. Differences among the response functions with respect to these times could be attributed to the repeated measurement factor Time. To incorporate the Time variation into the model, specify the following statements:

```
proc catmod;
   response marginals;
   model t1*t2*t3=_response_;
   repeated Time 3 / _response_=Time;
run;
```

These statements produce a Time effect that has $2(k-1)$ degrees of freedom since there are $k-1$ response functions at each time point. For a dichotomous variable, the Time effect has two degrees of freedom.

Now suppose that at each time point, each subject has X-rays taken, and the X-rays are read by two different radiologists. This creates six dependent variables that represent the 3×2 cross-classification of the repeated measurement factors Time and Reader. A saturated model with respect to these factors can be obtained by specifying the following statements:

```
proc catmod;
   response marginals;
   model r11*r12*r21*r22*r31*r32=_response_;
   repeated Time 3, Reader 2
      / _response_=Time Reader Time*Reader;
run;
```

If you want to fit a main-effects model with respect to Time and Reader, then change the REPEATED statement to the following:

```
   repeated Time 3, Reader 2 / _response_=Time Reader;
```

If you want to fit a main-effects model for Time but for only one of the readers, the REPEATED statement might look like the following:

```
   repeated Time $ 3, Reader $ 2
            /_response_=Time(Reader=Smith)
              profile  =('1'  Smith,
                         '1'  Jones,
                         '2'  Smith,
```

```
                          '2'  Jones,
                          '3'  Smith,
                          '3'  Jones);
```

If Jones had been unavailable for a reading at time 3, then there would be only $5(k-1)$ response functions, even though PROC CATMOD would be expecting some multiple of 6 ($= 3 \times 2$). In that case, the PROFILE= option would be necessary to indicate which repeated measurement profiles were actually represented:

```
repeated Time $ 3, Reader $ 2
         /_response_=Time(Reader=Smith)
           profile  =('1'  Smith,
                      '1'  Jones,
                      '2'  Smith,
                      '2'  Jones,
                      '3'  Smith);
```

When two or more repeated measurement factors are specified, PROC CATMOD presumes that the response functions are ordered so that the levels of the rightmost factor change most rapidly. This means that the dependent variables should be specified in the same order. For this example, the order implied by the REPEATED statement is as follows, where the variable r_{ij} corresponds to Time i and Reader j:

Response Function	Dependent Variable	Time	Reader
1	r_{11}	1	1
2	r_{12}	1	2
3	r_{21}	2	1
4	r_{22}	2	2
5	r_{31}	3	1
6	r_{32}	3	2

The order of dependent variables in the MODEL statement must agree with the order implied by the REPEATED statement.

Multiple Populations

When there are variables specified in the POPULATION statement or on the right side of the MODEL statement, these variables produce multiple populations. PROC CATMOD can then model these independent variables, the repeated measurement factors, and the interactions between the two.

For example, suppose that there are five groups of subjects, that each subject in the study is measured at three different times, and that the dichotomous dependent variables are labeled t1, t2, and t3. The following statements compute three response functions for each population:

```
proc catmod;
   weight wt;
   population Group;
   response marginals;
   model t1*t2*t3=_response_;
   repeated Time / _response_=Time;
run;
```

PROC CATMOD then regards _RESPONSE_ as a variable with three levels corresponding to the three response functions in each population and forms an effect with two degrees of freedom. The MODEL and REPEATED statements tell PROC CATMOD to fit the main effect of Time.

In general, the MODEL statement tells PROC CATMOD how to integrate the independent variables and the repeated measurement factors into the model. For example, again suppose that there are five groups of subjects, that each subject is measured at three times, and that the dichotomous independent variables are labeled t1, t2, and t3. If you use the same WEIGHT, POPULATION, RESPONSE, and REPEATED statements as in the preceding program, the following MODEL statements result in the indicated analyses:

`model t1*t2*t3=Group / averaged;`	Specifies the Group main effect (with 4 degrees of freedom)
`model t1*t2*t3=_response_;`	Specifies the Time main effect (with 2 degrees of freedom)
`model t1*t2*t3=_response_*Group;`	Specifies the interaction between Time and Group (with 8 degrees of freedom)
`model t1*t2*t3=_response_|Group;`	Specifies both main effects, and the interaction between Time and Group (with a total of 14 degrees of freedom)
`model t1*t2*t3=_response_(Group);`	Specifies a Time main effect within each Group (with 10 degrees of freedom)

However, the following MODEL statement is invalid since effects cannot be nested within _RESPONSE_:

```
model t1*t2*t3=Group(_response_);
```

Generation of the Design Matrix

Each row of the design matrix (corresponding to a population) is generated by a unique combination of independent variable values. Each column of the design matrix corresponds to a model parameter. The columns are produced from the effect specifications in the MODEL, LOGLIN, FACTORS, and REPEATED statements. For details about effect specifications, see the section "Specification of Effects" on page 2274.

This section is divided into three parts:

- one response function per population
- two or more response functions per population (excluding log-linear models), beginning on page 2287
- log-linear models, beginning on page 2291

This section assumes that the default effect parameterization is used. Specifying the reference parameterization replaces the "–1"s with zeros in the design matrix for the main effects of classification variables, and makes appropriate changes to interaction terms.

You can display the design matrix by specifying the DESIGN option in the MODEL statement.

One Response Function per Population

Intercept

When there is one response function per population, all design matrices start with a column of 1s for the intercept unless the NOINT option is specified or the design matrix is input directly.

Main Effects

If a classification variable A has k levels, then its main effect has $k-1$ degrees of freedom, and the design matrix has $k-1$ columns that correspond to the first $k-1$ levels of A. The ith column contains a 1 in the ith row, a –1 in the last row, and 0s everywhere else. If α_i denotes the parameter that corresponds to the ith level of variable A, then the $k-1$ columns yield estimates of the independent parameters, $\alpha_1, \alpha_i, \ldots, \alpha_{k-1}$. The last parameter is not needed because PROC CATMOD constrains the k parameters to sum to zero. In other words, PROC CATMOD uses a full-rank center-point parameterization to build design matrices. Here are two examples:

Variable	Data Levels	Effect Parameterization Design Matrix	
A	1	1	0
	2	0	1
	3	–1	–1
B	1	–1	
	2	–1	

For an effect with three levels, such as A, PROC CATMOD produces two parameter estimates for each response function. By default, the first (corresponding to the first row in the design columns) estimates the effect of level 1 of A compared to the average effect of the three levels of A. The second (corresponding to the second row in the design columns) estimates the effect of level 2 of A compared to the average effect of the three levels of A. The sum-to-zero constraint requires the effect of level 3 of A to be the negative of the sum of the level 1 and 2 effects (as shown by the third row in the design columns).

Crossed Effects (Interactions)

Crossed effects (such as A*B) are formed by the horizontal direct products of main effects, as illustrated in the following table:

Data Levels		Design Matrix				
A	**B**	**A**		**B**	**A*B**	
1	1	1	0	1	1	0
1	2	1	0	–1	–1	0
2	1	0	1	1	0	1
2	2	0	1	–1	0	–1
3	1	–1	–1	1	–1	–1
3	2	–1	–1	–1	1	1

The number of degrees of freedom for a crossed effect (that is, the number of design matrix columns) is equal to the product of the numbers of degrees of freedom for the separate effects.

Nested Effects

The effect A(B) is read "A within B" and is the same as specifying an A main effect for every value of B. If n_a and n_b are the number of levels in A and B, respectively, then the number of columns for A(B) is $(n_a - 1)n_b$ when every combination of levels exists in the data. The following table gives an example:

Data Levels		Design Matrix			
B	A	A(B)			
1	1	1	0	0	0
1	2	0	1	0	0
1	3	–1	–1	0	0
2	1	0	0	1	0
2	2	0	0	0	1
2	3	0	0	–1	–1

CAUTION: PROC CATMOD actually allocates a column for all possible combinations of values even though some combinations are not present in the data. This can be of particular concern if the data are not balanced with respect to the nested levels.

Nested-by-Value Effects

Instead of nesting an effect within all values of the main effect, you can nest an effect within specified values of the nested variable (A(B=1), for example). The four degrees of freedom for the A(B) effect shown in the preceding section can also be obtained by specifying the two separate nested effects with values, as the following table shows:

Data Levels		Design Matrix			
B	A	A(B=1)		A(B=2)	
1	1	1	0	0	0
1	2	0	1	0	0
1	3	–1	–1	0	0
2	1	0	0	1	0
2	2	0	0	0	1
2	3	0	0	–1	–1

Each effect has $n_a - 1$ degrees of freedom, assuming a complete combination, so each effect in this example has two degrees of freedom.

The procedure compares nested values to data values on the basis of formatted values. If a format is not specified for the variable, the procedure formats internal data values to BEST16, left-justified. The nested values specified in nested-by-value effects are also converted to a BEST16 formatted value, left-justified.

For example, if the numeric variable B has internal data values 1 and 2, then A(B=1), A(B=1.0), and A(B=1E0) are all valid nested-by-value effects. However, if the data value 1 is formatted as 'one', then A(B='one') is a valid effect, but A(B=1) is not since the formatted nested value (1) does not match the formatted data value (one).

To ensure correct nested-by-value effects, look at the tables of population and response profiles. These are displayed by default, and they contain the formatted data values. In addition, the population and response profiles are displayed when you specify the ONEWAY option in the MODEL statement.

Direct Effects

To request that the actual values of a variable be inserted into the design matrix, declare the variable in a DIRECT statement, and specify the effect by the variable name. For example, specifying the effects X1 and X2 in both the MODEL and DIRECT statements results in the following:

Data Levels		Design Columns	
X1	X2	X1	X2
1	1	1	1
2	4	2	4
3	9	3	9

Unless there is a POPULATION statement that excludes the direct variables, the direct variables help to define the sample populations. In general, the variables should not be continuous in the sense that every subject has a different value because this would create a separate population for each subject (note, however, that such a strategy is used purposely for logistic regression).

If there is a POPULATION statement that omits mention of the direct variables, then the values of the direct variables must be identical for all subjects in a given population since there can be only one independent variable profile for each population.

Two or More Response Functions per Population

When there is more than one response function per population, the structure of the design matrix depends on whether or not the model type is AVERAGED (see the AVERAGED option in the MODEL statement). The model type is AVERAGED if independent variable effects are averaged over the multiple responses within a population rather than being nested in them.

The following subsections illustrate the effect of specifying (or not specifying) an AVERAGED model type. This section does not apply to log-linear models; for these models, see the section "Log-Linear Model Design Matrices" on page 2291.

Model Type Not AVERAGED

Suppose the variable A has two levels, and you specify the following statements:

```
proc catmod;
   model Y=A / design;
run;
```

If the variable Y has two levels, then there is only one response function per population, and the design matrix is as follows:

	Design Matrix	
Sample	Intercept	A
1	1	1
2	1	–1

But if the variable Y has three levels, then there are two response functions per population, and the preceding design matrix is assumed to hold for each of the two response functions. The response functions are always ordered so that the multiple response functions within a population are grouped together. For this example, the design matrix would be as follows:

Sample	Response Function Number	Design Matrix			
		Intercept		A	
1	1	1	0	1	0
1	2	0	1	0	1
2	1	1	0	–1	0
2	2	0	1	0	–1

Since the same submatrix applies to each of the multiple response functions, PROC CATMOD displays only the submatrix (that is, the one it would create if there were only one response function per population) rather than the entire design matrix. PROC CATMOD displays

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Ordering of Parameters

This grouping of multiple response functions within populations also has an effect in the table of parameter estimates displayed by PROC CATMOD. The following table shows some parameter estimates, where the four rows of the table correspond to the four columns in the preceding design matrix:

Effect	Parameter	Estimate
Intercept	1	1.4979
	2	0.8404
A	3	0.1116
	4	–0.3296

Notice that the intercept and the A effect each have two parameter estimates associated with them. The first estimate in each pair is associated with the first response function, and the second in each pair is associated with the second response function. Consequently, 0.1116 is the effect of the first level of A on the first response function. In any table of parameter estimates displayed by PROC CATMOD, as you read down the column of estimates, the response function level changes before levels of the variables making up the effect.

Model Type AVERAGED

When the model type is AVERAGED (for example, when the AVERAGED option is specified in the MODEL statement, when _RESPONSE_ is used in the MODEL statement, or when the design matrix is input directly in the MODEL statement), PROC CATMOD does not assume that the same submatrix applies to each of the q response functions per population. Rather, it averages any independent variable effects across the functions, and it enables you to study variation among the q functions. The first column of the design matrix is always a column of 1s corresponding to the intercept, unless the NOINT option is specified in the MODEL statement or the design matrix is input directly. Also, since the design matrix does not have any special submatrix structure, PROC CATMOD displays the entire matrix.

For example, suppose the dependent variable Y has three levels, the independent variable A has two levels, and you specify the following statements:

```
proc catmod;
   response marginals;
   model y=a / averaged design;
run;
```

Then there are two response functions per population, and the response functions are always ordered so that the multiple response functions within a population are grouped together. For this example, the design matrix would be as follows:

Sample	Response Function Number	Design Matrix Intercept	A
1	1	1	1
1	2	1	1
2	1	1	–1
2	2	1	–1

Note that the model now has only two degrees of freedom. The remaining two degrees of freedom in the residual correspond to variation among the three levels of the dependent variable. Generally, that variation tends to be statistically significant and therefore should not be left out of the model. You can include it in the model by including the two effects, _RESPONSE_ and _RESPONSE_*A, but if the study is not a repeated measures study, those sources of variation tend to be uninteresting. The usual solution for this type of study (one dependent variable) is to exclude the AVERAGED option from the MODEL statement.

An AVERAGED model type is automatically used whenever you use the _RESPONSE_ keyword in the MODEL statement. The _RESPONSE_ effect models variation among the q response functions per population. If there is no REPEATED, FACTORS, or LOGLIN statement, then PROC CATMOD builds a main effect with $q - 1$ degrees of freedom. For example, three response functions would produce the following design columns:

Response Function Number	Design Columns _RESPONSE_	
1	1	0
2	0	1
3	–1	–1

If there is more than one population, then the _RESPONSE_ effect is averaged over the populations. Also, the _RESPONSE_ effect can be crossed with any other effect, or it can be nested within an effect.

If there is a REPEATED statement that contains only one repeated measurement factor, then PROC CATMOD builds the design columns for _RESPONSE_ in the same way, except that the output labels the main effect with the factor name rather than with the word _RESPONSE_. For example, suppose an independent variable A has two levels, and the input statements are as follows:

```
proc catmod;
   response marginals;
   model Time1*Time2=A _response_ A*_response_ / design;
   repeated Time 2 / _response_=Time;
run;
```

If Time1 and Time2 each have two levels (so that they each have one independent marginal probability), then the RESPONSE statement makes PROC CATMOD compute two response functions per population. The design matrix is as follows:

Sample	Response Function Number	Design Matrix			
		Intercept	A	Time	A*Time
1	1	1	1	1	1
1	2	1	1	–1	–1
2	1	1	–1	1	–1
2	2	1	–1	–1	1

However, if Time1 and Time2 each have three levels (so that they each have two independent marginal probabilities), then the RESPONSE statement causes PROC CATMOD to compute four response functions per population. In that case, since Time has two levels, PROC CATMOD groups the functions into sets of 2 $(= 4/2)$ and constructs the preceding submatrix for each function in the set. This results in the following design matrix, which is obtained from the previous one by multiplying each element by an identity matrix of order two:

Sample	Response Function	Design Matrix							
		Intercept		A		Time		A*Time	
1	P(Time1=1)	1	0	1	0	1	0	1	0
1	P(Time1=2)	0	1	0	1	0	1	0	1
1	P(Time2=1)	1	0	1	0	–1	0	–1	0
1	P(Time2=2)	0	1	0	1	0	–1	0	–1
2	P(Time1=1)	1	0	–1	0	1	0	–1	0
2	P(Time1=2)	0	1	0	–1	0	1	0	–1
2	P(Time2=1)	1	0	–1	0	–1	0	1	0
2	P(Time2=2)	0	1	0	–1	0	–1	0	1

If there is a REPEATED statement that contains two or more repeated measurement factors, then PROC CATMOD builds the design columns for _RESPONSE_ according to the definition of _RESPONSE_ in the REPEATED statement. For example, suppose you specify the following statements:

```
proc catmod;
   response marginals;
   model R11*R12*R21*R22=_response_ / design;
   repeated Time 2, Place 2 / _response_=Time Place;
run;
```

If each of the dependent variables has two levels, then PROC CATMOD builds four response functions. The _RESPONSE_ effect generates a main-effects model with respect to Time and Place, and the design matrix is as follows:

Response Function Number	Variable	Time	Place	Design Matrix Intercept	_RESPONSE_	
1	R11	1	1	1	1	1
2	R12	1	2	1	1	–1
3	R21	2	1	1	–1	1
4	R22	2	2	1	–1	–1

Log-Linear Model Design Matrices

When the response functions are the standard ones (generalized logits), then inclusion of the keyword _RESPONSE_ in every design effect fits a log-linear model. The design matrix for a log-linear model looks different from a standard design matrix because the standard one is transformed by the same linear transformation that converts the r response probabilities to $r-1$ generalized logits. For example, suppose the dependent variables X and Y each have two levels, and you specify a saturated log-linear model analysis:

```
proc catmod;
   model X*Y=_response_ / design;
   loglin X Y X*Y;
run;
```

Then the cross-classification of X and Y yields four response probabilities, p_{11}, p_{12}, p_{21}, and p_{22}, which are then reduced to three generalized logit response functions, $F_1 = \log(p_{11}/p_{22})$, $F_2 = \log(p_{12}/p_{22})$, and $F_3 = \log(p_{21}/p_{22})$.

Since the saturated log-linear model implies that

$$\begin{bmatrix} \log(p_{11}) \\ \log(p_{12}) \\ \log(p_{21}) \\ \log(p_{22}) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \boldsymbol{\gamma} - \lambda \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix} \boldsymbol{\beta} - \delta \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

where $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are parameter vectors, and λ and δ are normalizing constants required by the restriction that

the probabilities sum to 1, it follows that the MODEL statement yields

$$\begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \times \begin{bmatrix} \log(p_{11}) \\ \log(p_{12}) \\ \log(p_{21}) \\ \log(p_{22}) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & -1 \\ -1 & 1 & -1 \\ -1 & -1 & 1 \end{bmatrix} \boldsymbol{\beta}$$

$$= \begin{bmatrix} 2 & 2 & 0 \\ 2 & 0 & -2 \\ 0 & 2 & -2 \end{bmatrix} \boldsymbol{\beta}$$

The design matrix is as follows:

Sample	Response Function Number	Design Matrix X	Y	X*Y
1	1	2	2	0
1	2	2	0	–2
1	3	0	2	–2

Design matrices for reduced models are constructed similarly. For example, suppose you request a main-effects log-linear model analysis of the factors X and Y:

```
proc catmod;
   model X*Y=_response_ / design;
   loglin X Y;
run;
```

Since the main-effects log-linear model implies that

$$\begin{bmatrix} \log(p_{11}) \\ \log(p_{12}) \\ \log(p_{21}) \\ \log(p_{22}) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & 1 \\ 1 & -1 & -1 \end{bmatrix} \boldsymbol{\gamma} - \lambda \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & -1 \end{bmatrix} \boldsymbol{\beta} - \delta \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

it follows that the MODEL statement yields

$$
\begin{aligned}
\begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \times \begin{bmatrix} \log(p_{11}) \\ \log(p_{12}) \\ \log(p_{21}) \\ \log(p_{22}) \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & -1 \end{bmatrix} \boldsymbol{\beta} \\
&= \begin{bmatrix} 2 & 2 \\ 2 & 0 \\ 0 & 2 \end{bmatrix} \boldsymbol{\beta}
\end{aligned}
$$

Therefore, the corresponding design matrix is as follows:

Sample	Response Function Number	Design Matrix X	Design Matrix Y
1	1	2	2
1	2	2	0
1	3	0	2

Since it is difficult to tell from the final design matrix whether PROC CATMOD used the parameterization that you intended, the procedure displays the untransformed _RESPONSE_ matrix for log-linear models. For example, specifying the main-effects model in the preceding example displays the following matrix:

Response Function Number	_RESPONSE_ Matrix 1	_RESPONSE_ Matrix 2
1	1	1
2	1	–1
3	–1	1
4	–1	–1

You can suppress the display of this matrix by specifying the NORESPONSE option in the MODEL statement.

Cautions

Effective Sample Size

Since the method depends on asymptotic approximations, you need to be careful that the sample sizes are sufficiently large to support the asymptotic normal distributions of the response functions. A general guideline is that you would like to have an effective sample size of at least 25 to 30 for each response function that is being analyzed. For example, if you have one dependent variable and $r = 4$ response levels, and you use the standard response functions to compute three generalized logits for each population, then you would like the sample size of each population to be at least 75. Moreover, the subjects should be dispersed throughout the table so that less than 20 percent of the response functions have an effective sample size less than 5. For example, if each population had less than 5 subjects in the first response category, then it would be wiser to pool this category with another category rather than to assume the asymptotic normality of the first response function. Or, if the dependent variable is ordinally scaled, an alternative is to request the mean score response function rather than three generalized logits.

If there is more than one dependent variable, and you specify RESPONSE MEANS, then the effective sample size for each response function is the same as the actual sample size. Thus, a sample size of 30 could be sufficient to support four response functions, provided that the functions are the means of four dependent variables.

A Singular Covariance Matrix

If there is a singular (noninvertible) covariance matrix for the response functions in any population, then PROC CATMOD writes an error message and stops processing. You have several options available to correct this problem:

- You can reduce the number of response functions according to how many can be supported by the populations with the smallest sample sizes.
- If there are three or more levels for any independent variable, you can pool the levels into a fewer number of categories, thereby reducing the number of populations. However, your interpretation of results must be done more cautiously since such pooling implies a different sampling scheme and masks any differences that existed among the pooled categories.
- If there are two or more independent variables, you can delete at least one of them from the model. However, this is just another form of pooling, and the same cautions that apply to the previous option also apply here.
- If there is one independent variable, then, in some situations, you might simply eliminate the populations that are causing the covariance matrices to be singular.
- You can use the ADDCELL= option in the MODEL statement to add a small amount (for example, 0.5) to every cell frequency, but this can seriously bias the results if the cell frequencies are small.

Zero Frequencies

There are two types of zero cells in a contingency table: structural and sampling. A structural zero cell has an expected value of zero, while a sampling zero cell can have nonzero expected value and can be estimable.

If you use the standard response functions and there are zero frequencies, you should use maximum likelihood estimation (the default is ML=NR) rather than weighted least squares to analyze the data. For weighted least squares analysis, the CATMOD procedure always computes the observed response functions and might need to take the logarithm of a zero proportion. In this case, PROC CATMOD issues a warning and then takes the log of a small value ($0.5/n_i$ for the probability) in order to continue, but this can produce invalid results if the cells contain too few observations. Maximum likelihood analysis, on the other hand, does not require computation of the observed response functions and therefore yields valid results for the parameter estimates and all of the predicted values.

For a log-linear model analysis with WLS or ML=NR, PROC CATMOD creates response profiles only for the observed profiles. For any log-linear model analysis with one population (the usual case), the contingency table does not contain zeros, which means that all zero frequencies are treated as structural zeros. If there is more than one population, then a zero in the body of the contingency table is treated as a sampling zero (as long as some population has a nonzero count for that profile). If you fit the log-linear model by using ML=IPF, the contingency table is incomplete and the zeros are treated like structural zeros. If you want zero frequencies that PROC CATMOD would normally treat as structural zeros to be interpreted as sampling zeros, you can specify the ZERO=SAMPLING and MISSING=SAMPLING options in the MODEL statement. Alternatively, you can specify ZERO=1E–20 and MISSING=1E–20.

See Bishop, Fienberg, and Holland (1975) for a discussion of the issues and Example 36.5 for an illustration of a log-linear model analysis of data that contain both structural and sampling zeros.

If you perform a weighted least squares analysis on a contingency table that contains zero cell frequencies, then avoid using the LOG transformation as the first transformation on the observed proportions. In general, it is better to change the response functions or to pool some of the response categories than to settle for the 0.5 correction or to use the ADDCELL= option.

Testing the Wrong Hypothesis

If you use the keyword _RESPONSE_ in the MODEL statement, and you specify MARGINALS, LOGITS, ALOGITS, or CLOGITS in your RESPONSE statement, you might receive the following warning message:

```
Warning: The _RESPONSE_ effect may be testing the wrong
         hypothesis since the marginal levels of the
         dependent variables do not coincide. Consult the
         response profiles and the CATMOD documentation.
```

The following examples illustrate situations in which the _RESPONSE_ effect tests the wrong hypothesis.

Zeros in the Marginal Frequencies

Suppose you specify the following statements:

```
data A1;
   input Time1 Time2 @@;
   datalines;
1 2    2 3    1 3
```

```
;

proc catmod;
   response marginals;
   model Time1*Time2=_response_;
   repeated Time 2 / _response_=Time;
run;
```

One marginal probability is computed for each dependent variable, resulting in two response functions. The model is a saturated one: one degree of freedom for the intercept and one for the main effect of Time. Except for the warning message, PROC CATMOD produces an analysis with no apparent errors, but the "Response Profiles" table displayed by PROC CATMOD is as follows:

Response Profiles		
Response	**Time1**	**Time2**
1	1	2
2	1	3
3	2	3

Since RESPONSE MARGINALS yields marginal probabilities for every level but the last, the two response functions being analyzed are Prob(Time1=1) and Prob(Time2=2). The Time effect is testing the hypothesis that Prob(Time1=1)=Prob(Time2=2). What it *should* be testing is the hypothesis that

```
Prob(Time1=1) = Prob(Time2=1)
Prob(Time1=2) = Prob(Time2=2)
Prob(Time1=3) = Prob(Time2=3)
```

but there are not enough data to support the test (assuming that none of the probabilities are structural zeros by the design of the study).

The ORDER=DATA Option

Suppose you specify the following statements:

```
data a1;
   input Time1 Time2 @@;
   datalines;
2 1    2 2    1 1    1 2    2 1
;

proc catmod order=data;
   response marginals;
   model Time1*Time2=_response_;
   repeated Time 2 / _response_=Time;
run;
```

As in the preceding example, one marginal probability is computed for each dependent variable, resulting in two response functions. The model is also the same: one degree of freedom for the intercept and one for the main effect of Time. PROC CATMOD issues the warning message and displays the following "Response Profiles" table:

Response Profiles		
Response	Time1	Time2
1	2	1
2	2	2
3	1	1
4	1	2

Although the marginal levels are the same for the two dependent variables, they are not in the same order because the ORDER=DATA option specified that they be ordered according to their appearance in the input stream. Since RESPONSE MARGINALS yields marginal probabilities for every level except the last, the two response functions being analyzed are Prob(Time1=2) and Prob(Time2=1). The Time effect is testing the hypothesis that Prob(Time1=2)=Prob(Time2=1). What it *should* be testing is the hypothesis that

```
Prob(Time1=1) = Prob(Time2=1)
Prob(Time1=2) = Prob(Time2=2)
```

Whenever the warning message appears, look at the "Response Profiles" table or the "One-Way Frequencies" table to determine what hypothesis is actually being tested. For the latter example, a correct analysis can be obtained by deleting the ORDER=DATA option or by reordering the data so that the (1,1) observation is first.

Computational Method

The notation used in PROC CATMOD differs slightly from that used in the literature. The following table provides a summary of the basic dimensions and the notation for a contingency table. See the section "Computational Formulas" on page 2298 for a complete description.

Summary of Basic Dimensions

s = number of populations or samples (= number of rows in the underlying contingency table)

r = number of response categories (= number of columns in the underlying contingency table)

q = number of response functions computed for each population

d = number of parameters

Notation

$\mathbf{j}$	Denotes a column vector of 1s.
$\mathbf{J}$	Denotes a square matrix of 1s.
$\sum_k$	Denotes the sum over all the possible values of k.
n_i	Denotes the row sum $\sum_j n_{ij}$.
$\mathbf{DIAG}_n(\mathbf{p})$	Denotes the diagonal matrix formed from the first n elements of the vector $\mathbf{p}$.
$\mathbf{DIAG}_n^{-1}(\mathbf{p})$	Denotes the inverse of $\mathbf{DIAG}_n(\mathbf{p})$.
$\mathbf{DIAG}(\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_k)$	Denotes a block diagonal matrix with the $\mathbf{A}$ matrices on the main diagonal.

Input data can be represented by a contingency table, as shown in Table 36.7.

Table 36.7 Input Data Represented by a Contingency Table

	Response				
Population	**1**	**2**	$\cdots$	r	**Total**
1	n_{11}	n_{12}	$\cdots$	n_{1r}	n_1
2	n_{21}	n_{22}	$\cdots$	n_{2r}	n_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
s	n_{s1}	n_{s2}	$\cdots$	n_{sr}	n_s

Computational Formulas

The following formulas are shown for each population and for all populations combined.

Source	Formula	Dimension
Probability Estimates		
jth response	$p_{ij} = \dfrac{n_{ij}}{n_i}$	1×1
ith population	$\mathbf{p}_i = \begin{bmatrix} p_{i1} \\ p_{i2} \\ \vdots \\ p_{ir} \end{bmatrix}$	$r \times 1$
all populations	$\mathbf{p} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_s \end{bmatrix}$	$sr \times 1$
Variance of Probability Estimates		
ith population	$\mathbf{V}_i = \dfrac{1}{n_i}(\mathbf{DIAG}(\mathbf{p}_i) - \mathbf{p}_i\mathbf{p}_i{}')$	$r \times r$
all populations	$\mathbf{V} = \mathbf{DIAG}(\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_s)$	$sr \times sr$
Response Functions		
ith population	$\mathbf{F}_i = \mathbf{F}(\mathbf{p}_i)$	$q \times 1$
all populations	$\mathbf{F} = \begin{bmatrix} \mathbf{F}_1 \\ \mathbf{F}_2 \\ \vdots \\ \mathbf{F}_s \end{bmatrix}$	$sq \times 1$

Source	Formula	Dimension
Derivative of Function with Respect to Probability Estimates		
ith population	$\mathbf{H}_i = \dfrac{\partial \mathbf{F}(\mathbf{p}_i)}{\partial \mathbf{p}_i}$	$q \times r$
all populations	$\mathbf{H} = \mathbf{DIAG}(\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_s)$	$sq \times sr$
Variance of Functions		
ith population	$\mathbf{S}_i = \mathbf{H}_i \mathbf{V}_i \mathbf{H}_i{}'$	$q \times q$
all populations	$\mathbf{S} = \mathbf{DIAG}(\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_s)$	$sq \times sq$
Inverse Variance of Functions		
ith population	$\mathbf{S}^i = (\mathbf{S}_i)^{-1}$	$q \times q$
all populations	$\mathbf{S}^{-1} = \mathbf{DIAG}(\mathbf{S}^1, \mathbf{S}^2, \ldots, \mathbf{S}^s)$	$sq \times sq$

Derivative Table for Compound Functions: Y=F(G(p))

In the following table, let $\mathbf{G}(\mathbf{p})$ be a vector of functions of $\mathbf{p}$, and let $\mathbf{D}$ denote $\partial\mathbf{G}/\partial\mathbf{p}$, which is the first derivative matrix of $\mathbf{G}$ with respect to $\mathbf{p}$:

Function	$\mathbf{Y} = \mathbf{F}(\mathbf{G})$	**Derivative** $(\partial\mathbf{Y}/\partial\mathbf{p})$
Multiply matrix	$\mathbf{Y} = \mathbf{A} * \mathbf{G}$	$\mathbf{A} * \mathbf{D}$
Logarithm	$\mathbf{Y} = \mathbf{LOG}(\mathbf{G})$	$\mathbf{DIAG}^{-1}(\mathbf{G}) * \mathbf{D}$
Exponential	$\mathbf{Y} = \mathbf{EXP}(\mathbf{G})$	$\mathbf{DIAG}(\mathbf{Y}) * \mathbf{D}$
Add constant	$\mathbf{Y} = \mathbf{G} + \mathbf{A}$	$\mathbf{D}$

Default Response Functions: Generalized Logits

In the following table, subscripts i for the population are suppressed. Also denote $f_j = \log\left(\dfrac{p_j}{p_r}\right)$ for $j = 1, \ldots, r-1$ for each population $i = 1, \ldots, s$.

Formula
Inverse of Response Functions for a Population
$p_j = \dfrac{\exp(f_j)}{1 + \sum_k \exp(f_k)}$ for $j = 1, \ldots, r-1$
$p_r = \dfrac{1}{1 + \sum_k \exp(f_k)}$

Formula
Form of F and Derivative for a Population
$\mathbf{F} = \mathbf{KLOG(p)} = (\mathbf{I}_{r-1}, -\mathbf{j})\ \mathbf{LOG(p)}$
$\mathbf{H} = \frac{\partial \mathbf{F}}{\partial \mathbf{p}} = \left(\mathbf{DIAG}_{r-1}^{-1}(\mathbf{p}), \frac{-1}{p_r}\mathbf{j}\right)$
Covariance Results for a Population
$\mathbf{S} = \mathbf{HVH}'$
$= \frac{1}{n}\left(\mathbf{DIAG}_{r-1}^{-1}(\mathbf{p}) + \frac{1}{p_r}\mathbf{J}_{r-1}\right)$
where $\mathbf{V}, \mathbf{H},$ and $\mathbf{J}$ are as previously defined.
$\mathbf{S}^{-1} = n(\mathbf{DIAG}_{r-1}(\mathbf{p}) - \mathbf{qq}')$, where $\mathbf{q} = \mathbf{DIAG}_{r-1}(\mathbf{p})\ \mathbf{j}$
$\mathbf{S}^{-1}\mathbf{F} = n\mathbf{DIAG}_{r-1}(\mathbf{p})\mathbf{F} - (n\sum_j p_j f_j)\ \mathbf{q}$
$\mathbf{F}'\mathbf{S}^{-1}\mathbf{F} = n\sum_j p_j f_j^2 - n(\sum_j p_j f_j)^2$

The following calculations are shown for each population and then for all populations combined:

Source	Formula	Dimension
Design Matrix		
ith population	$\mathbf{X}_i$	$q \times d$
all populations	$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \\ \vdots \\ \mathbf{X}_s \end{bmatrix}$	$sq \times d$
Crossproduct of Design Matrix		
ith population	$\mathbf{C}_i = \mathbf{X}_i{}'\mathbf{S}^i\mathbf{X}_i$	$d \times d$
all populations	$\mathbf{C} = \mathbf{X}'\mathbf{S}^{-1}\mathbf{X} = \sum_i \mathbf{C}_i$	$d \times d$

In the following table, z_p is the 100pth percentile of the standard normal distribution:

Formula	Dimension
Crossproduct of Design Matrix with Function	
$\mathbf{R} = \mathbf{X}'\mathbf{S}^{-1}\mathbf{F} = \sum_i \mathbf{X}_i{}'\mathbf{S}^i\mathbf{F}_i$	$d \times 1$

Formula	Dimension
Weighted Least Squares Estimates	
$\mathbf{b} = \mathbf{C}^{-1}\mathbf{R} = (\mathbf{X}'\mathbf{S}^{-1}\mathbf{X})^{-1}(\mathbf{X}'\mathbf{S}^{-1}\mathbf{F})$	$d \times 1$
Covariance of Weighted Least Squares Estimates	
$\mathbf{COV}(\mathbf{b}) = \mathbf{C}^{-1}$	$d \times d$
Wald Confidence Limits for Parameter Estimates	
$b_k \pm z_{1-\alpha/2}\mathbf{C}^{-1}_{kk}$	$k = 1, \ldots, d$
Predicted Response Functions	
$\hat{\mathbf{F}} = \mathbf{X}\mathbf{b}$	$sq \times 1$
Covariance of Predicted Response Functions	
$\mathbf{V}_{\hat{\mathbf{F}}} = \mathbf{X}\mathbf{C}^{-1}\mathbf{X}'$	$sq \times sq$
Residual Chi-Square	
$\mathrm{RSS} = \mathbf{F}'\mathbf{S}^{-1}\mathbf{F} - \hat{\mathbf{F}}'\mathbf{S}^{-1}\hat{\mathbf{F}}$	1×1
Chi-Square for H_0: $\mathbf{L}\boldsymbol{\beta} = \mathbf{0}$	
$\mathrm{Q} = (\mathbf{L}\mathbf{b})'(\mathbf{L}\mathbf{C}^{-1}\mathbf{L}')^{-1}(\mathbf{L}\mathbf{b})$	1×1

Maximum Likelihood Method

Let $\mathbf{C}$ be the Hessian matrix and $\mathbf{G}$ be the gradient of the log-likelihood function (both functions of $\boldsymbol{\pi}$ and the parameters $\boldsymbol{\beta}$). Let $\mathbf{p}_i^*$ denote the vector containing the first $r - 1$ sample proportions from population i, and let $\boldsymbol{\pi}_i^*$ denote the corresponding vector of probability estimates from the current iteration. Starting with the least squares estimates $\mathbf{b}_0$ of $\boldsymbol{\beta}$ (if you use the ML and WLS options; with the ML option alone, the procedure starts with $\mathbf{0}$), the probabilities $\boldsymbol{\pi}(\mathbf{b})$ are computed, and $\mathbf{b}$ is calculated iteratively by the Newton-Raphson method until it converges (see the EPSILON= option). The factor λ is a step-halving factor that equals one at the start of each iteration. For any iteration in which the likelihood decreases, PROC CATMOD uses a series of subiterations in which λ is iteratively divided by two. The subiterations continue until the likelihood is greater than that of the previous iteration. If the likelihood has not reached that point after 10 subiterations, then convergence is assumed, and a warning message is displayed.

Sometimes, infinite parameters are present in the model, either because of the presence of one or more zero frequencies or because of a poorly specified model with collinearity among the estimates. If an estimate is tending toward infinity, then PROC CATMOD flags the parameter as infinite and holds the estimate fixed in subsequent iterations. PROC CATMOD regards a parameter to be infinite when two conditions apply:

- The absolute value of its estimate exceeds five divided by the range of the corresponding variable.

- The standard error of its estimate is at least three times greater than the estimate itself.

The estimator of the asymptotic covariance matrix of the maximum likelihood predicted probabilities is given by Imrey, Koch, and Stokes (1981, eq. 2.18).

The following equations summarize the method:

$$\mathbf{b}_{k+1} = \mathbf{b}_k - \lambda \mathbf{C}^{-1}\mathbf{G}$$

where

$$\begin{aligned}
\mathbf{C} &= \mathbf{X}'\mathbf{S}^{-1}(\boldsymbol{\pi})\mathbf{X} \\
\mathbf{N} &= \begin{bmatrix} n_1(\mathbf{p}_1^* - \boldsymbol{\pi}_1^*) \\ \vdots \\ n_s(\mathbf{p}_s^* - \boldsymbol{\pi}_s^*) \end{bmatrix} \\
\mathbf{G} &= \mathbf{X}'\mathbf{N}
\end{aligned}$$

Iterative Proportional Fitting

The algorithm used by PROC CATMOD for iterative proportional fitting is described in Bishop, Fienberg, and Holland (1975); Haberman (1972); Agresti (2002). To illustrate the method, consider the observed three-dimensional table $\{n_{ijk}\}$ for the variables X, Y, and Z, and the following hierarchical model:

$$\log(m_{ijk}) = \mu + \lambda_i^X + \lambda_j^Y + \lambda_k^Z + \lambda_{ij}^{XY} + \lambda_{ik}^{XZ} + \lambda_{jk}^{YZ}$$

The following statements request that PROC CATMOD use IPF to fit the preceding model:

```
model X*Y*Z = _response_ / ml=ipf;
loglin X|Y|Z@2;
```

Begin with a table of initial cell estimates $\{\hat{m}_{ijk}^{(0)}\}$. PROC CATMOD produces the initial estimates by setting the n_{sz} structural zero cells to 0 and all other cells to $n/(n_c - n_{sz})$, where n is the total weight of the table and n_c is the total number of cells in the table. Iteratively adjust the estimates at step $s-1$ to the observed marginal tables specified in the model by cycling through the following three-stage process to produce the estimates at step s:

$$\begin{aligned}
\hat{m}_{ijk}^{(s_1)} &= \hat{m}_{ijk}^{(s-1)} \frac{n_{ij\cdot}}{\hat{m}_{ij\cdot}^{(s-1)}} \\
\hat{m}_{ijk}^{(s_2)} &= \hat{m}_{ijk}^{(s_1)} \frac{n_{i\cdot k}}{\hat{m}_{i\cdot k}^{(s_1)}} \\
\hat{m}_{ijk}^{(s)} &= \hat{m}_{ijk}^{(s_2)} \frac{n_{\cdot jk}}{\hat{m}_{\cdot jk}^{(s_2)}}
\end{aligned}$$

The subscript "·" indicates summation over the missing subscript. The log-likelihood l_s is estimated at each step s by

$$l_s = \sum_{i,j,k} n_{ijk} \log \left(\frac{\hat{m}_{ijk}^{(s)}}{n} \right)$$

When the function $|(l_{s-1} - l_s)/l_{s-1}|$ is less than 10^{-8}, the iterations terminate. You can change the comparison value with the EPSILON= option, and you can change the convergence criterion with the CONVCRIT= option. The option CONVCRIT=CELL uses the maximum cell difference

$$\max_{i,j,k} |\hat{m}_{ijk}^{(s-1)} - \hat{m}_{ijk}^{(s)}|$$

as the criterion while the option CONVCRIT=MARGIN computes the maximum difference of the margins

$$\text{Maximum of} \left\{ \max_{i,j} |\hat{m}_{ij\cdot}^{(s-1)} - \hat{m}_{ij\cdot}^{(s)}|, \max_{i,k} |\hat{m}_{i\cdot k}^{(s-1)} - \hat{m}_{i\cdot k}^{(s)}|, \max_{j,k} |\hat{m}_{\cdot jk}^{(s-1)} - \hat{m}_{\cdot jk}^{(s)}| \right\}$$

Memory and Time Requirements

The memory and time required by PROC CATMOD are proportional to the number of parameters in the model.

Displayed Output

PROC CATMOD displays the following information in the "Data Summary" table:

- the response effect
- the weight variable, if one is specified
- the data set name
- the number of response levels
- the number of samples or populations
- the total frequency, which is the total sample size
- the number of observations from the data set (the number of data records)
- the frequency of missing observations, labeled as "Frequency Missing"

Except for the analysis of variance table, all of the following items can be displayed or suppressed, depending on your specification of statements and *options*.

- The ONEWAY option produces the "One-Way Frequencies" table, which displays the frequencies of each variable value used in the analysis.

- The populations (or samples) are defined in a table labeled "Population Profiles." The sample size and the values of the defining variables are displayed for each sample. This table is suppressed if the NOPROFILE option is specified.

- The observed responses are defined in a table labeled "Response Profiles." The values of the defining variables are displayed for each response. This table is suppressed if the NOPROFILE option is specified.

- If the FREQ option is specified, then the "Response Frequencies" table is displayed, which shows the frequency of each response for each population.

- If the PROB option is specified, then the "Response Probabilities" table is produced. This table displays the probability of each response for each population.

- If the COV option is specified, the "Response Functions, Covariance Matrix" table, which shows the covariance matrix of the response functions for each sample, is displayed.

- If the DESIGN option is specified, the response functions are displayed in the "Response Functions, Design Matrix" table. If the COV option is also specified, the response functions are displayed in the "Response Functions, Covariance Matrix" table.

- If the DESIGN option is specified, the design matrix is displayed in the "Response Functions, Design Matrix" table, and if a log-linear model is being fit, the _RESPONSE_ matrix is displayed in the "_Response_ Matrix" table. If the model type is AVERAGED, then the design matrix is displayed with $q * s$ rows, assuming q response functions for each of s populations. Otherwise, the design matrix is displayed with only s rows since the model is the same for each of the q response functions.

- The "X′*Inv(S)*X" matrix is displayed for weighted least squares analyses if the XPX option is specified.

- The "Analysis of Variance" table for the weighted least squares analysis reports the results of significance tests for each of the *design-effects* on the right side of the MODEL statement. If _RESPONSE_ is a *design-effect* and is defined explicitly in the LOGLIN, FACTORS, or REPEATED statement, then the table contains test statistics for the individual effects constituting the _RESPONSE_ effect. If the design matrix is input directly, then the content of the displayed output depends on whether you specify any subsets of the parameters to be tested. If you specify one or more subsets, then the table contains one test for each subset. Otherwise, the table contains one test for the effect MODEL | MEAN. In every case, the table also contains the residual goodness-of-fit test. Produced for each test of significance are the source of variation, the number of degrees of freedom (DF), the Wald chi-square value, and the significance probability (Pr > ChiSq).

- The "Analysis of Weighted Least Squares Estimates" table lists, for each parameter in the model, the least squares estimate, the estimated standard error of the parameter estimate, the Wald chi-square value (calculated as $((\text{parameter estimate})/(\text{standard error}))^2$) for testing that the parameter is zero, and the significance probability (Pr > ChiSq) of the test. If the CLPARM option is specified, then 95% Wald confidence intervals are displayed.

  Each row in the table is labeled with the parameter (the model effect and the class levels) and the response function number; however, if the NOPREDVAR option or a REPEATED or FACTORS statement is specified or if the design matrix is directly input, the rows are labeled by the effect in the model for which parameters are formed and the parameter number.

- The "Covariance Matrix of the Parameter Estimates" table for the weighted least squares analysis displays the estimated covariance matrix of the least squares estimates of the parameters, provided that the COVB option is specified.

- The "Correlation Matrix of the Parameter Estimates" table for the weighted least squares analysis displays the estimated correlation matrix of the least squares estimates of the parameters, provided that the CORRB option is specified.

- The "Maximum Likelihood Analysis" table is produced when the ML and ITPRINT options are specified for the standard response functions. It displays the iteration number, the number of step-halving sub-iterations, –2 log likelihood for that iteration, the convergence criterion, and the parameter estimates for each iteration.

- The "Maximum Likelihood Analysis of Variance" table, displayed when the ML option is specified for the standard response functions, is similar to the table produced for the least squares analysis. The Wald chi-square test for each effect is based on the information matrix from the likelihood calculations. The likelihood ratio statistic compares the specified model with the unrestricted (saturated) model and is an appropriate goodness-of-fit test for the model.

- The "Analysis of Maximum Likelihood Estimates" table, displayed when the ML option is specified for the standard response functions, is similar to the one produced for the least squares analysis. The table includes the maximum likelihood estimates, the estimated standard errors based on the information matrix, and the Wald chi-square statistics based on estimated standard errors.

- The "Covariance Matrix of the Maximum Likelihood Estimates" table displays the estimated covariance matrix of the maximum likelihood estimates of the parameters, provided that the COVB and ML options are specified for the standard response functions.

- The "Correlation Matrix of the Maximum Likelihood Estimates" table displays the estimated correlation matrix of the maximum likelihood estimates of the parameters, provided that the CORRB and ML options are specified for the standard response functions.

- For each source of variation specified in a CONTRAST statement, the "Contrasts" table lists the label for the source (Contrast), the number of degrees of freedom (DF), the Wald chi-square value, and the significance probability (Pr > ChiSq). If the ESTIMATE= option is specified, the "Analysis of Contrasts" table displays, for each row of the contrast, the label (Contrast), the type (PARM or EXP), the row of the contrast, the estimate and its standard error, a Wald confidence interval, the Wald chi-square, and the *p*-value (Pr > ChiSq) for 1 degree of freedom.

- Specification of the PREDICT option in the MODEL statement has the following effect. Produced for each response function within each population are the observed and predicted function values, their standard errors, and the residual (observed minus predicted). If the response functions are the default ones (generalized logits), additional information displayed for each response within each population includes the observed and predicted cell probabilities, their standard errors, and the residual. However, specifying PRED=FREQ in the MODEL statement results in the display of the predicted cell frequencies rather than the predicted cell probabilities. The displayed output includes the population profiles and, for the response function table, the function number, while the probability and frequency tables display the response profiles. If the NOPREDVAR option is specified in the MODEL statement, the population profiles are replaced with the sample numbers, and the response profiles are replaced with the labels "P*n*" for the *n*th cell probability, and "F*n*" for the *n*th cell frequency.

- When there are multiple RESPONSE statements, the output for each statement starts on a new page. For each RESPONSE statement, the corresponding title, if specified, is displayed at the top of each page.
- If the ADDCELL= option is specified in the MODEL statement, and if there is a weighted least squares analysis specified, the adjusted sample size for each population (with number added to each cell) is labeled "Adjusted Sample Size" in the "Population Profiles" table. Similarly, the adjusted response frequencies and probabilities are displayed in the "Adjusted Response Frequencies" and "Adjusted Response Probabilities" tables, respectively.
- If _RESPONSE_ is defined explicitly in the LOGLIN, FACTORS, or REPEATED statement, then the definition is displayed as a note whenever _RESPONSE_ appears in the output.

ODS Table Names

PROC CATMOD assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 36.12 ODS Tables Produced by PROC CATMOD

ODS Table Name	Description	Statement	Option
ANOVA	Analysis of variance	MODEL	Default
Contrasts	Contrasts	CONTRAST	Default
ContrastEstimates	Analysis of contrasts	CONTRAST	ESTIMATE=
ConvergenceStatus	Convergence status	MODEL	ML
CorrB	Correlation matrix of the estimates	MODEL	CORRB
CovB	Covariance matrix of the estimates	MODEL	COVB
DataSummary	Data summary	PROC	Default
Estimates	Analysis of estimates	MODEL	Default, unless NOPARM
MaxLikelihood	Maximum likelihood analysis	MODEL	ML and ITPRINT
OneWayFreqs	One-way frequencies	MODEL	ONEWAY
PopProfiles	Population profiles	MODEL	Default, unless NOPROFILE
PredictedFreqs	Predicted frequencies	MODEL	PRED=FREQ
PredictedProbs	Predicted probabilities	MODEL	PREDICT or PRED=PROB
PredictedValues	Predicted values	MODEL	PREDICT or PRED=
ResponseCov	Response functions, covariance matrix	MODEL	COV
ResponseDesign	Response functions, design matrix	MODEL	DESIGN, unless NODESIGN
ResponseFreqs	Response frequencies	MODEL	FREQ

Table 36.12 *continued*

ODS Table Name	Description	Statement	Option
ResponseMatrix	_RESPONSE_ matrix	MODEL and LOGLIN	DESIGN, unless NORESPONSE
ResponseProbs	Response probabilities	MODEL	PROB
ResponseProfiles	Response profiles	MODEL	Default, unless NOPROFILE
XPX	**X**'*Inv(**S**)*X matrix	MODEL	XPX, for WLS*

* WLS estimation is the default for response functions other than the default (generalized logits).

Examples: CATMOD Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 36.1: Linear Response Function, r=2 Responses

(View the complete code for this example (catex1.sas) in the example repository.)

In an example from Ries and Smith (1963), the choice of detergent brand (Brand = M or X) is related to three other categorical variables: the softness of the laundry water (Softness = soft, medium, or hard), the temperature of the water (Temperature = high or low), and whether the subject was a previous user of Brand M (Previous = yes or no). The linear response function, which could also be specified as RESPONSE MARGINALS, yields one probability, Pr(brand preference=M), as the response function to be analyzed. Two models are fit in this example: the first model is a saturated one, containing all of the main effects and interactions, while the second is a reduced model containing only the main effects. The following statements produce Output 36.1.1 through Output 36.1.4:

```
data detergent;
   input Softness $ Brand $ Previous $ Temperature $ Count @@;
   datalines;
soft X yes high 19   soft X yes low 57
soft X no  high 29   soft X no  low 63
soft M yes high 29   soft M yes low 49
soft M no  high 27   soft M no  low 53
med  X yes high 23   med  X yes low 47
med  X no  high 33   med  X no  low 66
med  M yes high 47   med  M yes low 55
med  M no  high 23   med  M no  low 50
hard X yes high 24   hard X yes low 37
hard X no  high 42   hard X no  low 68
hard M yes high 43   hard M yes low 52
hard M no  high 30   hard M no  low 42
;
```

```
title 'Detergent Preference Study';
proc catmod data=detergent;
   response 1 0;
   weight Count;
   model Brand=Softness|Previous|Temperature / freq prob;
   title2 'Saturated Model';
run;
```

The "Data Summary" table (Output 36.1.1) indicates that you have two response levels and twelve populations.

Output 36.1.1 Detergent Preference Study: Linear Model Analysis

Detergent Preference Study
Saturated Model

The CATMOD Procedure

Data Summary			
Response	Brand	**Response Levels**	2
Weight Variable	Count	**Populations**	12
Data Set	DETERGENT	**Total Frequency**	1008
Frequency Missing	0	**Observations**	24

The "Population Profiles" table in Output 36.1.2 displays the ordering of independent variable levels as used in the table of parameter estimates.

Output 36.1.2 Population Profiles

Population Profiles				
Sample	**Softness**	**Previous**	**Temperature**	**Sample Size**
1	hard	no	high	72
2	hard	no	low	110
3	hard	yes	high	67
4	hard	yes	low	89
5	med	no	high	56
6	med	no	low	116
7	med	yes	high	70
8	med	yes	low	102
9	soft	no	high	56
10	soft	no	low	116
11	soft	yes	high	48
12	soft	yes	low	106

Since Brand M is the first level in the "Response Profiles" table (Output 36.1.3), the RESPONSE statement causes Pr(Brand=M) to be the single response function modeled.

Output 36.1.3 Response Profiles, Frequencies, and Probabilities

Response Profiles

Response	Brand
1	M
2	X

Response Frequencies

	Response Number	
Sample	**1**	**2**
1	30	42
2	42	68
3	43	24
4	52	37
5	23	33
6	50	66
7	47	23
8	55	47
9	27	29
10	53	63
11	29	19
12	49	57

Response Probabilities

	Response Number	
Sample	**1**	**2**
1	0.41667	0.58333
2	0.38182	0.61818
3	0.64179	0.35821
4	0.58427	0.41573
5	0.41071	0.58929
6	0.43103	0.56897
7	0.67143	0.32857
8	0.53922	0.46078
9	0.48214	0.51786
10	0.45690	0.54310
11	0.60417	0.39583
12	0.46226	0.53774

The “Analysis of Variance” table in Output 36.1.4 shows that all of the interactions are nonsignificant.

Output 36.1.4 Analysis of Variance

Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	1	983.13	<.0001
Softness	2	0.09	0.9575
Previous	1	22.68	<.0001
Softness*Previous	2	3.85	0.1457
Temperature	1	3.67	0.0555
Softness*Temperature	2	0.23	0.8914
Previous*Temperature	1	2.26	0.1324
Softnes*Previou*Temperat	2	0.76	0.6850
Residual	0	.	.

Therefore, a main-effects model is fit with the following statements:

```
    model Brand=Softness Previous Temperature
        / clparm noprofile design;
    title2 'Main-Effects Model';
run;
quit;
```

The PROC CATMOD statement is not required due to the interactive capability of the CATMOD procedure. The NOPROFILE option suppresses the redisplay of the "Response Profiles" table. The CLPARM option produces 95% confidence limits for the parameter estimates. Output 36.1.5 through Output 36.1.7 are produced.

The design matrix in Output 36.1.5 displays the results of the differential-effects modeling used in PROC CATMOD.

Output 36.1.5 Main-Effects Design Matrix

Detergent Preference Study
Main-Effects Model

The CATMOD Procedure

Data Summary			
Response	Brand	**Response Levels**	2
Weight Variable	Count	**Populations**	12
Data Set	DETERGENT	**Total Frequency**	1008
Frequency Missing	0	**Observations**	24

Output 36.1.5 *continued*

Response Functions and Design Matrix						
		Design Matrix				
Sample	Response Function	1	2	3	4	5
1	0.41667	1	1	0	1	1
2	0.38182	1	1	0	1	-1
3	0.64179	1	1	0	-1	1
4	0.58427	1	1	0	-1	-1
5	0.41071	1	0	1	1	1
6	0.43103	1	0	1	1	-1
7	0.67143	1	0	1	-1	1
8	0.53922	1	0	1	-1	-1
9	0.48214	1	-1	-1	1	1
10	0.45690	1	-1	-1	1	-1
11	0.60417	1	-1	-1	-1	1
12	0.46226	1	-1	-1	-1	-1

The analysis of variance table in Output 36.1.6 shows that previous use of Brand M, together with the temperature of the laundry water, is a significant factor in whether a subject prefers Brand M laundry detergent. The table also shows that the additive model fits since the goodness-of-fit statistic (the residual chi-square) is nonsignificant.

Output 36.1.6 ANOVA Table for the Main-Effects Model

Analysis of Variance			
Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	1004.93	<.0001
Softness	2	0.24	0.8859
Previous	1	20.96	<.0001
Temperature	1	3.95	0.0468
Residual	7	8.26	0.3100

The chi-square test in Output 36.1.7 shows that the Softness parameters are not significantly different from zero; as expected, the Wald confidence limits for these two estimates contain zero. So softness of the water is not a factor in choosing Brand M.

Output 36.1.7 WLS Estimates for the Main-Effects Model

Analysis of Weighted Least Squares Estimates							
Parameter		Estimate	Standard Error	Chi-Square	Pr > ChiSq	95% Confidence Limits	
Intercept		0.5080	0.0160	1004.93	<.0001	0.4766	0.5394
Softness	hard	-0.00256	0.0218	0.01	0.9066	-0.0454	0.0402
	med	0.0104	0.0218	0.23	0.6342	-0.0323	0.0530
Previous	no	-0.0711	0.0155	20.96	<.0001	-0.1015	-0.0407
Temperature	high	0.0319	0.0161	3.95	0.0468	0.000446	0.0634

The negative coefficient for Previous (–0.0711) indicates that the first level of Previous (which is shown to be 'no') is associated with a smaller probability of preferring Brand M than the second level of Previous (with coefficient constrained to be 0.0711 since the parameter estimates for a given effect must sum to zero). In other words, previous users of Brand M are much more likely to prefer it than those who have never used it before.

Similarly, the positive coefficient for Temperature indicates that the first level of Temperature (which, from the "Population Profiles" table, is 'high') has a larger probability of preferring Brand M than the second level of Temperature. In other words, those who do their laundry in hot water are more likely to prefer Brand M than those who do their laundry in cold water.

Example 36.2: Mean Score Response Function, r=3 Responses

(View the complete code for this example (catex2.sas) in the example repository.)

Four surgical operations for duodenal ulcers are compared in a clinical trial at four hospitals. The operations performed are as follows: Treatment = a, drainage and vagotomy; Treatment = b, 25% resection and vagotomy; Treatment = c, 50% resection and vagotomy; and Treatment = d, 75% resection. The response is severity of an undesirable complication called "dumping syndrome." The data in the following statements are from Grizzle, Starmer, and Koch (1969, pp. 489–504).

```
data operate;
   input Hospital Treatment $ Severity $ wt @@;
   datalines;
1 a none 23    1 a slight  7    1 a moderate 2
1 b none 23    1 b slight 10    1 b moderate 5
1 c none 20    1 c slight 13    1 c moderate 5
1 d none 24    1 d slight 10    1 d moderate 6
2 a none 18    2 a slight  6    2 a moderate 1
2 b none 18    2 b slight  6    2 b moderate 2
2 c none 13    2 c slight 13    2 c moderate 2
2 d none  9    2 d slight 15    2 d moderate 2
3 a none  8    3 a slight  6    3 a moderate 3
3 b none 12    3 b slight  4    3 b moderate 4
3 c none 11    3 c slight  6    3 c moderate 2
3 d none  7    3 d slight  7    3 d moderate 4
4 a none 12    4 a slight  9    4 a moderate 1
4 b none 15    4 b slight  3    4 b moderate 2
4 c none 14    4 c slight  8    4 c moderate 3
4 d none 13    4 d slight  6    4 d moderate 4
;
```

The response variable (Severity) is ordinally scaled with three levels, so assignment of scores is appropriate (0 = none, 0.5 = slight, 1 = moderate). For these scores, the response function yields the mean score. The following statements produce Output 36.2.1 through Output 36.2.3:

```
title 'Dumping Syndrome Data';
proc catmod data=operate order=data ;
   weight wt;
   response 0  0.5  1;
   model Severity=Treatment Hospital / freq oneway design;
```

```
   title2 'Main-Effects Model';
quit;
```

The ORDER= option is specified so that the levels of the response variable remain in the correct order. A main-effects model is fit. The ONEWAY option produces a table of the number of subjects within each variable level, and the FREQ option displays the frequency of each response within each sample (Output 36.2.1).

Output 36.2.1 Surgical Data: Analysis of Mean Scores

Dumping Syndrome Data
Main-Effects Model

The CATMOD Procedure

Data Summary			
Response	Severity	**Response Levels**	3
Weight Variable	wt	**Populations**	16
Data Set	OPERATE	**Total Frequency**	417
Frequency Missing	0	**Observations**	48

One-Way Frequencies		
Variable	**Value**	**Frequency**
Severity	none	240
	slight	129
	moderate	48
Treatment	a	96
	b	104
	c	110
	d	107
Hospital	1	148
	2	105
	3	74
	4	90

Output 36.2.1 *continued*

Population Profiles			
Sample	**Treatment**	**Hospital**	**Sample Size**
1	a	1	32
2	a	2	25
3	a	3	17
4	a	4	22
5	b	1	38
6	b	2	26
7	b	3	20
8	b	4	20
9	c	1	38
10	c	2	28
11	c	3	19
12	c	4	25
13	d	1	40
14	d	2	26
15	d	3	18
16	d	4	23

Response Profiles	
Response	**Severity**
1	none
2	slight
3	moderate

Response Frequencies			
	Response Number		
Sample	**1**	**2**	**3**
1	23	7	2
2	18	6	1
3	8	6	3
4	12	9	1
5	23	10	5
6	18	6	2
7	12	4	4
8	15	3	2
9	20	13	5
10	13	13	2
11	11	6	2
12	14	8	3
13	24	10	6
14	9	15	2
15	7	7	4
16	13	6	4

You can use the one-way frequencies and the response profiles from Output 36.2.1 to verify that the response

levels are in the desired order (none, slight, moderate) so that the response scores (0, 0.5, 1.0) are applied appropriately. If the ORDER=DATA option had not been used, the levels would have been in a different order.

The analysis of variance table (Output 36.2.2) shows that the additive model fits (since the residual chi-square is not significant), that the Treatment effect is significant, and that the Hospital effect is not significant.

Output 36.2.2 Surgical Data: Analysis of Mean Scores

Response Functions and Design Matrix

Sample	Response Function	Design Matrix 1	2	3	4	5	6	7
1	0.17188	1	1	0	0	1	0	0
2	0.16000	1	1	0	0	0	1	0
3	0.35294	1	1	0	0	0	0	1
4	0.25000	1	1	0	0	-1	-1	-1
5	0.26316	1	0	1	0	1	0	0
6	0.19231	1	0	1	0	0	1	0
7	0.30000	1	0	1	0	0	0	1
8	0.17500	1	0	1	0	-1	-1	-1
9	0.30263	1	0	0	1	1	0	0
10	0.30357	1	0	0	1	0	1	0
11	0.26316	1	0	0	1	0	0	1
12	0.28000	1	0	0	1	-1	-1	-1
13	0.27500	1	-1	-1	-1	1	0	0
14	0.36538	1	-1	-1	-1	0	1	0
15	0.41667	1	-1	-1	-1	0	0	1
16	0.30435	1	-1	-1	-1	-1	-1	-1

Analysis of Variance

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	248.77	<.0001
Treatment	3	8.90	0.0307
Hospital	3	2.33	0.5065
Residual	9	6.33	0.7069

The coefficients of Treatment in Output 36.2.3 show that the first two treatments (with negative coefficients) have lower mean scores than the last two treatments (the fourth coefficient, not shown, must be positive since the four coefficients must sum to zero). In other words, the less severe treatments (the first two) cause significantly less severe dumping syndrome complications.

Output 36.2.3 Surgical Data: Analysis of Mean Scores

Analysis of Weighted Least Squares Estimates					
Parameter		Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept		0.2724	0.0173	248.77	<.0001
Treatment	a	-0.0552	0.0270	4.17	0.0411
	b	-0.0365	0.0289	1.59	0.2073
	c	0.0248	0.0280	0.78	0.3757
Hospital	1	-0.0204	0.0264	0.60	0.4388
	2	-0.0178	0.0268	0.44	0.5055
	3	0.0531	0.0352	2.28	0.1312

Example 36.3: Logistic Regression, Standard Response Function

(View the complete code for this example (catex3.sas) in the example repository.)

In this data set, from Cox and Snell (1989), ingots are prepared with different heating and soaking times and tested for their readiness to be rolled. The following DATA step creates a response variable Y with value 1 for ingots that are not ready and value 0 otherwise. The explanatory variables are Heat and Soak.

```
data ingots;
   input Heat Soak nready ntotal @@;
   Count=nready;
   Y=1;
   output;
   Count=ntotal-nready;
   Y=0;
   output;
   drop nready ntotal;
   datalines;
7 1.0 0 10   14 1.0 0 31   27 1.0 1 56   51 1.0 3 13
7 1.7 0 17   14 1.7 0 43   27 1.7 4 44   51 1.7 0  1
7 2.2 0  7   14 2.2 2 33   27 2.2 0 21   51 2.2 0  1
7 2.8 0 12   14 2.8 0 31   27 2.8 1 22   51 4.0 0  1
7 4.0 0  9   14 4.0 0 19   27 4.0 1 16
;
```

Logistic regression analysis is often used to investigate the relationship between discrete response variables and continuous explanatory variables. For logistic regression, the continuous *design-effects* are declared in a DIRECT statement. The following statements produce Output 36.3.1 through Output 36.3.6:

```
title 'Maximum Likelihood Logistic Regression';
proc catmod data=ingots;
   weight Count;
   direct Heat Soak;
   model Y=Heat Soak / freq covb corrb itprint design;
quit;
```

You can verify that the populations are defined as you intended by looking at the “Population Profiles” table in Output 36.3.1.

Output 36.3.1 Maximum Likelihood Logistic Regression

Maximum Likelihood Logistic Regression

The CATMOD Procedure

Data Summary			
Response	Y	**Response Levels**	2
Weight Variable	Count	**Populations**	19
Data Set	INGOTS	**Total Frequency**	387
Frequency Missing	0	**Observations**	25

Population Profiles			
Sample	**Heat**	**Soak**	**Sample Size**
1	7	1	10
2	7	1.7	17
3	7	2.2	7
4	7	2.8	12
5	7	4	9
6	14	1	31
7	14	1.7	43
8	14	2.2	33
9	14	2.8	31
10	14	4	19
11	27	1	56
12	27	1.7	44
13	27	2.2	21
14	27	2.8	22
15	27	4	16
16	51	1	13
17	51	1.7	1
18	51	2.2	1
19	51	4	1

Since the "Response Profiles" table in Output 36.3.2 shows the response level ordering as 0, 1, the default response function, the logit, is defined as $\log\left(\frac{p_{Y=0}}{p_{Y=1}}\right)$.

Output 36.3.2 Response Summaries

Response Profiles	
Response	**Y**
1	0
2	1

Output 36.3.2 *continued*

Response Frequencies		
	Response Number	
Sample	**1**	**2**
1	10	0
2	17	0
3	7	0
4	12	0
5	9	0
6	31	0
7	43	0
8	31	2
9	31	0
10	19	0
11	55	1
12	40	4
13	21	0
14	21	1
15	15	1
16	10	3
17	1	0
18	1	0
19	1	0

The values of the continuous variable are inserted into the design matrix (Output 36.3.3).

Output 36.3.3 Design Matrix

Response Functions and Design Matrix				
		Design Matrix		
Sample	Response Function	1	2	3
1	2.99573	1	7	1
2	3.52636	1	7	1.7
3	2.63906	1	7	2.2
4	3.17805	1	7	2.8
5	2.89037	1	7	4
6	4.12713	1	14	1
7	4.45435	1	14	1.7
8	2.74084	1	14	2.2
9	4.12713	1	14	2.8
10	3.63759	1	14	4
11	4.00733	1	27	1
12	2.30259	1	27	1.7
13	3.73767	1	27	2.2
14	3.04452	1	27	2.8
15	2.70805	1	27	4
16	1.20397	1	51	1
17	0.69315	1	51	1.7
18	0.69315	1	51	2.2
19	0.69315	1	51	4

Seven Newton-Raphson iterations are required to find the maximum likelihood estimates (Output 36.3.4).

Output 36.3.4 Iteration History

Maximum Likelihood Analysis						
				Parameter Estimates		
Iteration	Sub Iteration	-2 Log Likelihood	Convergence Criterion	1	2	3
0	0	536.49592	1.0000	0	0	0
1	0	152.58961	0.7156	2.1594	-0.0139	-0.003733
2	0	106.76066	0.3003	3.5334	-0.0363	-0.0120
3	0	96.692171	0.0943	4.7489	-0.0640	-0.0299
4	0	95.383825	0.0135	5.4138	-0.0790	-0.0498
5	0	95.345659	0.000400	5.5539	-0.0819	-0.0564
6	0	95.345613	4.8289E-7	5.5592	-0.0820	-0.0568
7	0	95.345613	7.731E-13	5.5592	-0.0820	-0.0568

Maximum likelihood computations converged.

The analysis of variance table (Output 36.3.5) shows that the model fits since the likelihood ratio goodness-of-fit test is nonsignificant. It also shows that the length of heating time is a significant factor with respect to readiness but that length of soaking time is not.

Output 36.3.5 Analysis of Variance Table

Maximum Likelihood Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	1	24.65	<.0001
Heat	1	11.95	0.0005
Soak	1	0.03	0.8639
Likelihood Ratio	16	13.75	0.6171

From the table of maximum likelihood estimates in Output 36.3.6, the fitted model is

$$\mathrm{E}(\mathrm{logit}(p)) = 5.559 - 0.082(\mathrm{Heat}) - 0.057(\mathrm{Soak})$$

For example, for Sample 1 with Heat = 7 and Soak = 1, the estimate is

$$\mathrm{E}(\mathrm{logit}(p)) = 5.559 - 0.082(7) - 0.057(1) = 4.9284$$

Output 36.3.6 Maximum Likelihood Estimates, Covariances, and Correlations

Analysis of Maximum Likelihood Estimates				
Parameter	**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Intercept	5.5592	1.1197	24.65	<.0001
Heat	-0.0820	0.0237	11.95	0.0005
Soak	-0.0568	0.3312	0.03	0.8639

Covariance Matrix of the Maximum Likelihood Estimates				
Row	**Parameter**	**Col1**	**Col2**	**Col3**
1	Intercept	1.2537133	-0.0215664	-0.2817648
2	Heat	-0.0215664	0.0005633	0.0026243
3	Soak	-0.2817648	0.0026243	0.1097020

Correlation Matrix of the Maximum Likelihood Estimates				
Row	**Parameter**	**Col1**	**Col2**	**Col3**
1	Intercept	1.00000	-0.81152	-0.75977
2	Heat	-0.81152	1.00000	0.33383
3	Soak	-0.75977	0.33383	1.00000

Predicted values of the logits, as well as the probabilities of readiness, could be obtained by specifying PRED=PROB in the MODEL statement. For the example of Sample 1 with Heat = 7 and Soak = 1, PRED=PROB would give an estimate of the probability of readiness equal to 0.9928 since

$$4.9284 = \log\left(\frac{\hat{p}}{1-\hat{p}}\right)$$

implies that

$$\hat{p} = \frac{e^{4.9284}}{1+e^{4.9284}} = 0.9928$$

As another consideration, since soaking time is nonsignificant, you could fit another model that deleted the variable Soak.

Example 36.4: Log-Linear Model, Three Dependent Variables

(View the complete code for this example (catex4.sas) in the example repository.)

This analysis reproduces the predicted cell frequencies for Bartlett's data by using a log-linear model of no three-variable interaction (Bishop, Fienberg, and Holland 1975, p. 89). Cuttings of two different lengths (Length=short or long) are planted at one of two time points (Time=now or spring), and their survival status (Status=dead or alive) is recorded.

As in the text, the variable levels are simply labeled 1 and 2. The following statements produce Output 36.4.1 through Output 36.4.3:

```
data bartlett;
   input Length Time Status wt @@;
   datalines;
1 1 1 156     1 1 2  84     1 2 1 84     1 2 2 156
2 1 1 107     2 1 2 133     2 2 1 31     2 2 2 209
;

title 'Bartlett''s Data';
proc catmod data=bartlett;
   weight wt;
   model Length*Time*Status=_response_
         / noparm pred=freq;
   loglin Length|Time|Status @ 2;
   title2 'Model with No 3-Variable Interaction';
quit;
```

Output 36.4.1 Analysis of Bartlett's Data: Log-Linear Model

Bartlett's Data
Model with No 3-Variable Interaction

The CATMOD Procedure

Data Summary			
Response	Length*Time*Status	**Response Levels**	8
Weight Variable	wt	**Populations**	1
Data Set	BARTLETT	**Total Frequency**	960
Frequency Missing	0	**Observations**	8

Population Profiles	
Sample	**Sample Size**
1	960

Output 36.4.1 *continued*

Response Profiles			
Response	**Length**	**Time**	**Status**
1	1	1	1
2	1	1	2
3	1	2	1
4	1	2	2
5	2	1	1
6	2	1	2
7	2	2	1
8	2	2	2

Maximum Likelihood Analysis
Maximum likelihood computations converged.

Maximum Likelihood Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Length	1	2.64	0.1041
Time	1	5.25	0.0220
Length*Time	1	5.25	0.0220
Status	1	48.94	<.0001
Length*Status	1	48.94	<.0001
Time*Status	1	95.01	<.0001
Likelihood Ratio	1	2.29	0.1299

The analysis of variance table shows that the model fits since the likelihood ratio test for the three-variable interaction is nonsignificant. All of the two-variable interactions, however, are significant; this shows that there is mutual dependence among all three variables.

The predicted values table (Output 36.4.2) displays observed and predicted values for the generalized logits.

Output 36.4.2 Response Function Predicted Values

Maximum Likelihood Predicted Values for Response Functions					
	Observed		**Predicted**		
Function Number	**Function**	**Standard Error**	**Function**	**Standard Error**	**Residual**
1	-0.29248	0.105806	-0.23565	0.098486	-0.05683
2	-0.91152	0.129188	-0.94942	0.129948	0.037901
3	-0.91152	0.129188	-0.94942	0.129948	0.037901
4	-0.29248	0.105806	-0.23565	0.098486	-0.05683
5	-0.66951	0.118872	-0.69362	0.120172	0.024113
6	-0.45199	0.110921	-0.3897	0.102267	-0.06229
7	-1.90835	0.192465	-1.73146	0.142969	-0.17688

The predicted frequencies table (Output 36.4.3) displays observed and predicted cell frequencies, their standard errors, and residuals.

Output 36.4.3 Predicted Frequencies

Maximum Likelihood Predicted Values for Frequencies							
			Observed		Predicted		
Length	Time	Status	Frequency	Standard Error	Frequency	Standard Error	Residual
1	1	1	156	11.43022	161.0961	11.07379	-5.09614
1	1	2	84	8.754999	78.90386	7.808613	5.096139
1	2	1	84	8.754999	78.90386	7.808613	5.096139
1	2	2	156	11.43022	161.0961	11.07379	-5.09614
2	1	1	107	9.750588	101.9039	8.924304	5.096139
2	1	2	133	10.70392	138.0961	10.33434	-5.09614
2	2	1	31	5.47713	36.09614	4.826315	-5.09614
2	2	2	209	12.78667	203.9039	12.21285	5.09614

Example 36.5: Log-Linear Model, Structural and Sampling Zeros

(View the complete code for this example (catex5.sas) in the example repository.)

This example illustrates a log-linear model of independence, by using data that contain structural zero frequencies as well as sampling (random) zero frequencies.

In a population of six squirrel monkeys, the joint distribution of genital display with respect to active or passive role was observed. The data are from Fienberg (1980, Table 8-2). Since a monkey cannot have both the active and passive roles in the same interaction, the diagonal cells of the table are structural zeros. See Agresti (2002) for more information about the quasi-independence model.

The DATA step replaces the structural zeros with missing values, and the MISSING=STRUCTURAL option is specified in the MODEL statement to remove these zeros from the analysis. The ZERO=SAMPLING option treats the off-diagonal zeros as sampling zeros. Also, the row for Monkey ‘t’ is deleted since it contains all zeros; therefore, the cell frequencies predicted by a model of independence are also zero. In addition, the CONTRAST statement compares the behavior of the two monkeys labeled ‘u’ and ‘v’. See the section “Structural and Sampling Zeros with Raw Data” on page 2328 for information about how to perform this analysis when you have raw data. The following statements produce Output 36.5.1 through Output 36.5.8:

```
data Display;
   input Active $ Passive $ wt @@;
   if Active ne 't';
   if Active eq Passive then wt=.;
   datalines;
r r  0   r s  1   r t  5   r u  8   r v  9   r w  0
s r 29   s s  0   s t 14   s u 46   s v  4   s w  0
t r  0   t s  0   t t  0   t u  0   t v  0   t w  0
u r  2   u s  3   u t  1   u u  0   u v 38   u w  2
v r  0   v s  0   v t  0   v u  0   v v  0   v w  1
w r  9   w s 25   w t  4   w u  6   w v 13   w w  0
;
```

```
title 'Behavior of Squirrel Monkeys';
proc catmod data=Display;
   weight wt;
   model Active*Passive=_response_ /
         missing=structural zero=sampling
         freq pred=freq noparm oneway;
   loglin Active Passive;
   contrast 'Passive, U vs. V' Passive 0 0 0 1 -1;
   contrast 'Active,  U vs. V' Active  0 0 1 -1;
   title2 'Test Quasi-Independence for the Incomplete Table';
quit;
```

Output 36.5.1 Log-Linear Model Analysis with Zero Frequencies

Behavior of Squirrel Monkeys
Test Quasi-Independence for the Incomplete Table

The CATMOD Procedure

Data Summary			
Response	Active*Passive	**Response Levels**	25
Weight Variable	wt	**Populations**	1
Data Set	DISPLAY	**Total Frequency**	220
Frequency Missing	0	**Observations**	25

The results of the ONEWAY option are shown in Output 36.5.2. Monkey ‘t’ does not show up as a value for the Active variable since that row was removed.

Output 36.5.2 Output from the ONEWAY option

One-Way Frequencies		
Variable	**Value**	**Frequency**
Active	r	23
	s	93
	u	46
	v	1
	w	57
Passive	r	40
	s	29
	t	24
	u	60
	v	64
	w	3

Sampling zeros are displayed as 0 in Output 36.5.4. The Response Number column corresponds to the value displayed in the “Response Profiles” table in Output 36.5.3.

Output 36.5.3 Profiles

Population Profiles	
Sample	**Sample Size**
1	220

Response Profiles		
Response	**Active**	**Passive**
1	r	s
2	r	t
3	r	u
4	r	v
5	r	w
6	s	r
7	s	t
8	s	u
9	s	v
10	s	w
11	u	r
12	u	s
13	u	t
14	u	v
15	u	w
16	v	r
17	v	s
18	v	t
19	v	u
20	v	w
21	w	r
22	w	s
23	w	t
24	w	u
25	w	v

Output 36.5.4 Frequency of Response by Response Number

Response Frequencies																									
	Response Number																								
Sample	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**
1	1	5	8	9	0	29	14	46	4	0	2	3	1	38	2	0	0	0	0	1	9	25	4	6	13

The analysis of variance table (Output 36.5.5) shows that the model of independence does not fit since the likelihood ratio test for the interaction is significant. In other words, active and passive behaviors of the squirrel monkeys are dependent behavior roles.

Output 36.5.5 Analysis of Variance Table

Maximum Likelihood Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Active	4	56.58	<.0001
Passive	5	47.94	<.0001
Likelihood Ratio	15	135.17	<.0001

If the model fit these data, then the contrasts in Output 36.5.6 show that monkeys ‘u’ and ‘v’ appear to have similar passive behavior patterns but very different active behavior patterns.

Output 36.5.6 Contrasts between Monkeys ‘u’ and ‘v’

Contrasts of Maximum Likelihood Estimates			
Contrast	**DF**	**Chi-Square**	**Pr > ChiSq**
Passive, U vs. V	1	1.31	0.2524
Active, U vs. V	1	14.87	0.0001

Output 36.5.7 displays the predicted response functions and Output 36.5.8 displays predicted cell frequencies (from the PRED=FREQ option), but since the model does not fit, these should be ignored. Note that, since the response function is the generalized logit with the 25th response as the baseline, the observed response functions for the sampling zeros are missing.

Output 36.5.7 Response Function Predicted Values

Maximum Likelihood Predicted Values for Response Functions

	Observed		Predicted		
Function Number	Function	Standard Error	Function	Standard Error	Residual
1	-2.56495	1.037749	-0.97355	0.339019	-1.5914
2	-0.95551	0.526235	-1.72504	0.345438	0.769529
3	-0.48551	0.449359	-0.52751	0.309254	0.042007
4	-0.36772	0.433629	-0.73927	0.249006	0.371543
5	.	.	-3.56052	0.634104	.
6	0.802346	0.333775	0.320589	0.26629	0.481758
7	0.074108	0.385164	-0.29934	0.295634	0.37345
8	1.263692	0.314105	0.898184	0.250857	0.365508
9	-1.17865	0.571772	0.686431	0.173396	-1.86509
10	.	.	-2.13482	0.608071	.
11	-1.8718	0.759555	-0.2415	0.287218	-1.63031
12	-1.46634	0.640513	-0.10994	0.303568	-1.3564
13	-2.56495	1.037749	-0.86143	0.314794	-1.70352
14	1.072637	0.321308	0.124346	0.204345	0.94829
15	-1.8718	0.759555	-2.6969	0.617433	0.8251
16	.	.	-4.14787	1.024508	.
17	.	.	-4.01632	1.030062	.
18	.	.	-4.76781	1.032457	.
19	.	.	-3.57028	1.020794	.
20	-2.56495	1.037749	-6.60328	1.161289	4.038332
21	-0.36772	0.433629	-0.36584	0.202959	-0.00188
22	0.653926	0.34194	-0.23429	0.232794	0.888212
23	-1.17865	0.571772	-0.98577	0.239408	-0.19288
24	-0.77319	0.493548	0.211754	0.185007	-0.98494

Output 36.5.8 Predicted Frequencies

Maximum Likelihood Predicted Values for Frequencies						
		Observed		Predicted		
Active	Passive	Frequency	Standard Error	Frequency	Standard Error	Residual
r	s	1	0.997725	5.259508	1.36156	-4.25951
r	t	5	2.210512	2.480726	0.691066	2.519274
r	u	8	2.776525	8.215948	1.855146	-0.21595
r	v	9	2.937996	6.648049	1.50932	2.351951
r	w	0	0	0.395769	0.240268	-0.39577
s	r	29	5.017696	19.18599	3.147915	9.814007
s	t	14	3.620648	10.32172	2.169599	3.678284
s	u	46	6.031734	34.18463	4.428706	11.81537
s	v	4	1.981735	27.66096	3.722788	-23.661
s	w	0	0	1.6467	0.952712	-1.6467
u	r	2	1.407771	10.9364	2.12322	-8.9364
u	s	3	1.720201	12.47407	2.554336	-9.47407
u	t	1	0.997725	5.883583	1.380655	-4.88358
u	v	38	5.606814	15.7673	2.684692	22.2327
u	w	2	1.407771	0.938652	0.551645	1.061348
v	r	0	0	0.219966	0.221779	-0.21997
v	s	0	0	0.250893	0.253706	-0.25089
v	t	0	0	0.118338	0.120314	-0.11834
v	u	0	0	0.391924	0.393255	-0.39192
v	w	1	0.997725	0.018879	0.021728	0.981121
w	r	9	2.937996	9.657645	1.808656	-0.65765
w	s	25	4.707344	11.01553	2.275019	13.98447
w	t	4	1.981735	5.195638	1.184452	-1.19564
w	u	6	2.415857	17.2075	2.772098	-11.2075
w	v	13	3.497402	13.92369	2.24158	-0.92369

Structural and Sampling Zeros with Raw Data

The preceding PROC CATMOD step uses cell count data as input. Prior to invoking the CATMOD procedure, structural and sampling zeros are easily identified and manipulated in a single DATA step. For the situation where structural or sampling zeros (or both) exist and the input data set is raw data, use the following steps:

1. Run PROC FREQ on the raw data (see Chapter 47, "The FREQ Procedure"). In the TABLES statement, list all dependent and independent variables, separated by asterisks, and use the SPARSE option and the OUT= option. This creates an output data set that contains all possible zero frequencies. Since the tabled output can be huge, you should also specify the NOPRINT option in the TABLES statement.

2. Use a DATA step to change the zero frequencies associated with either sampling zeros or structural zeros to missing.

3. Use the resulting data set as input to PROC CATMOD, specify the statement WEIGHT COUNT to use adjusted frequencies, and specify the ZERO= and MISSING= options to define your sampling and structural zeros.

For example, suppose the data set RawDisplay contains the raw data for the squirrel monkey data. The following statements show how to obtain the same analysis as shown previously:

```
proc freq data=RawDisplay;
   tables Active*Passive / sparse out=Combos noprint;
run;

data Combos2;
   set Combos;
   if Active ne 't';
   if Active eq Passive then count=.;
run;

proc catmod data=Combos2;
   weight count;
   model Active*Passive=_response_ /
         zero=sampling missing=structural
         freq pred=freq noparm noresponse;
   loglin Active Passive;
quit;
```

The first IF statement in the DATA step is needed only for this particular example; since observations for Monkey 't' were deleted from the Display data set, they also need to be deleted from Combos2.

Example 36.6: Repeated Measures, 2 Response Levels, 3 Populations

(View the complete code for this example (catex6.sas) in the example repository.)

In this multiple-population repeated measures example, from Guthrie (1981), subjects from three groups have their responses (0 or 1) recorded in each of four trials. The analysis of the marginal probabilities is directed at assessing the main effects of the repeated measurement factor (Trial) and the independent variable (Group), as well as their interaction. Although the contingency table is incomplete (only 13 of the 16 possible responses are observed), this poses no problem in the computation of the marginal probabilities. The following statements produce Output 36.6.1:

```
data group;
   input a b c d Group wt @@;
   datalines;
1 1 1 1 2 2     0 0 0 0 2 2     0 0 1 0 1 2     0 0 1 0 2 2
0 0 0 1 1 4     0 0 0 1 2 1     0 0 0 1 3 3     1 0 0 1 2 1
0 0 1 1 1 1     0 0 1 1 2 2     0 0 1 1 3 5     0 1 0 0 1 4
0 1 0 0 2 1     0 1 0 1 2 1     0 1 0 1 3 2     0 1 1 0 3 1
1 0 0 0 1 3     1 0 0 0 2 1     0 1 1 1 2 1     0 1 1 1 3 2
1 0 1 0 1 1     1 0 1 1 2 1     1 0 1 1 3 2
;

title 'Multiple-Population Repeated Measures';
proc catmod data=group;
   weight wt;
   response marginals;
   model a*b*c*d=Group _response_ Group*_response_
         / freq;
```

```
   repeated Trial 4;
   title2 'Saturated Model';
run;
```

Output 36.6.1 Analysis of Multiple-Population Repeated Measures

Multiple-Population Repeated Measures
Saturated Model

The CATMOD Procedure

Data Summary			
Response	a*b*c*d	**Response Levels**	13
Weight Variable	wt	**Populations**	3
Data Set	GROUP	**Total Frequency**	45
Frequency Missing	0	**Observations**	23

Population Profiles		
Sample	**Group**	**Sample Size**
1	1	15
2	2	15
3	3	15

Response Profiles				
Response	**a**	**b**	**c**	**d**
1	0	0	0	0
2	0	0	0	1
3	0	0	1	0
4	0	0	1	1
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	0	1	1	1
9	1	0	0	0
10	1	0	0	1
11	1	0	1	0
12	1	0	1	1
13	1	1	1	1

Response Frequencies													
	Response Number												
Sample	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	0	4	2	1	4	0	0	0	3	0	1	0	0
2	2	1	2	2	1	1	0	1	1	1	0	1	2
3	0	3	0	5	0	2	1	2	0	0	0	2	0

Output 36.6.1 *continued*

Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	1	354.88	<.0001
Group	2	24.79	<.0001
Trial	3	21.45	<.0001
Group*Trial	6	18.71	0.0047
Residual	0	.	.

Analysis of Weighted Least Squares Estimates					
Effect	**Parameter**	**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Intercept	**1**	0.5833	0.0310	354.88	<.0001
Group	**2**	0.1333	0.0335	15.88	<.0001
	3	-0.0333	0.0551	0.37	0.5450
Trial	**4**	0.1722	0.0557	9.57	0.0020
	5	0.1056	0.0647	2.66	0.1028
	6	-0.0722	0.0577	1.57	0.2107
Group*Trial	**7**	-0.1556	0.0852	3.33	0.0679
	8	-0.0556	0.0800	0.48	0.4877
	9	-0.0889	0.0953	0.87	0.3511
	10	0.0111	0.0866	0.02	0.8979
	11	0.0889	0.0822	1.17	0.2793
	12	-0.0111	0.0824	0.02	0.8927

The analysis of variance table in Output 36.6.1 shows that there is a significant interaction between the independent variable Group and the repeated measurement factor Trial. An intermediate model (not shown) is fit in which the effects Trial and Group* Trial are replaced by Trial(Group=1), Trial(Group=2), and Trial(Group=3). Of these three effects, only the last is significant, so it is retained in the final model. The following statements produce Output 36.6.2 and Output 36.6.3:

```
    model a*b*c*d=Group _response_(Group=3)
          / noprofile noparm design;
    title2 'Trial Nested within Group 3';
  quit;
```

Output 36.6.2 displays the design matrix resulting from retaining the nested effect.

Output 36.6.2 Final Model: Design Matrix

Multi-Population Repeated Measures
Trial Nested within Group 3

The CATMOD Procedure

Data Summary			
Response	a*b*c*d	**Response Levels**	13
Weight Variable	wt	**Populations**	3
Data Set	GROUP	**Total Frequency**	45
Frequency Missing	0	**Observations**	23

Output 36.6.2 *continued*

Response Functions and Design Matrix								
			Design Matrix					
Sample	Function Number	Response Function	1	2	3	4	5	6
1	1	0.73333	1	1	0	0	0	0
	2	0.73333	1	1	0	0	0	0
	3	0.73333	1	1	0	0	0	0
	4	0.66667	1	1	0	0	0	0
2	1	0.66667	1	0	1	0	0	0
	2	0.66667	1	0	1	0	0	0
	3	0.46667	1	0	1	0	0	0
	4	0.40000	1	0	1	0	0	0
3	1	0.86667	1	-1	-1	1	0	0
	2	0.66667	1	-1	-1	0	1	0
	3	0.33333	1	-1	-1	0	0	1
	4	0.06667	1	-1	-1	-1	-1	-1

The residual goodness-of-fit statistic tests the joint effect of Trial(Group=1) and Trial(Group=2). The analysis of variance table in Output 36.6.3 shows that the final model fits, that there is a significant Group effect, and that there is a significant Trial effect in Group 3.

Output 36.6.3 ANOVA Table

Analysis of Variance			
Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	386.94	<.0001
Group	2	25.42	<.0001
Trial(Group=3)	3	75.07	<.0001
Residual	6	5.09	0.5319

Example 36.7: Repeated Measures, 4 Response Levels, 1 Population

(View the complete code for this example (catex7.sas) in the example repository.)

This example illustrates a repeated measures analysis in which there are more than two levels of response. In this study, from Grizzle, Starmer, and Koch (1969, p. 493), 7,477 women aged 30–39 are tested for vision in both right and left eyes. Since there are four response levels for each dependent variable, the RESPONSE statement computes three marginal probabilities for each dependent variable, resulting in six response functions for analysis. Since the model contains a repeated measurement factor (Side) with two levels (Right, Left), PROC CATMOD groups the functions into sets of three (=6/2). Therefore, the Side effect has three degrees of freedom (one for each marginal probability), and it is the appropriate test of marginal homogeneity. The following statements produce Output 36.7.1:

```
title 'Vision Symmetry';
data vision;
   input Right Left count @@;
   datalines;
1 1 1520    1 2  266    1 3  124    1 4  66
2 1  234    2 2 1512    2 3  432    2 4  78
3 1  117    3 2  362    3 3 1772    3 4 205
4 1   36    4 2   82    4 3  179    4 4 492
;

proc catmod data=vision;
   weight count;
   response marginals;
   model Right*Left=_response_ / freq design;
   repeated Side 2;
   title2 'Test of Marginal Homogeneity';
quit;
```

Output 36.7.1 Vision Study: Analysis of Marginal Homogeneity

Vision Symmetry
Test of Marginal Homogeneity

The CATMOD Procedure

Data Summary			
Response	Right*Left	**Response Levels**	16
Weight Variable	count	**Populations**	1
Data Set	VISION	**Total Frequency**	7477
Frequency Missing	0	**Observations**	16

Population Profiles	
Sample	**Sample Size**
1	7477

Output 36.7.1 *continued*

Response Profiles		
Response	**Right**	**Left**
1	1	1
2	1	2
3	1	3
4	1	4
5	2	1
6	2	2
7	2	3
8	2	4
9	3	1
10	3	2
11	3	3
12	3	4
13	4	1
14	4	2
15	4	3
16	4	4

Response Frequencies

	Response Number															
Sample	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
1	1520	266	124	66	234	1512	432	78	117	362	1772	205	36	82	179	492

Response Functions and Design Matrix

			Design Matrix					
Sample	**Function Number**	**Response Function**	**1**	**2**	**3**	**4**	**5**	**6**
1	**1**	0.26428	1	0	0	1	0	0
	2	0.30173	0	1	0	0	1	0
	3	0.32847	0	0	1	0	0	1
	4	0.25505	1	0	0	-1	0	0
	5	0.29718	0	1	0	0	-1	0
	6	0.33529	0	0	1	0	0	-1

Analysis of Variance

Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	3	78744.17	<.0001
Side	3	11.98	0.0075
Residual	0	.	.

Output 36.7.1 *continued*

Analysis of Weighted Least Squares Estimates					
Effect	Parameter	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	0.2597	0.00468	3073.03	<.0001
	2	0.2995	0.00464	4160.17	<.0001
	3	0.3319	0.00483	4725.25	<.0001
Side	4	0.00461	0.00194	5.65	0.0174
	5	0.00227	0.00255	0.80	0.3726
	6	-0.00341	0.00252	1.83	0.1757

The analysis of variance table in Output 36.7.1 shows that the Side effect is significant, so there is not marginal homogeneity between left-eye vision and right-eye vision. In other words, the distribution of the quality of right-eye vision differs significantly from the distribution of the quality of left-eye vision in the same subjects. The test of the Side effect is equivalent to Bhapkar's test (Agresti 1990) .

Example 36.8: Repeated Measures, Logistic Analysis of Growth Curve

(View the complete code for this example (catex8.sas) in the example repository.)

The following data, from a longitudinal study reported in Koch et al. (1977), are from patients in four populations (2 diagnostic groups × 2 treatments) who are measured at three times to assess their response (n=normal or a=abnormal) to treatment:

```
title 'Growth Curve Analysis';
data growth2;
   input Diagnosis $ Treatment $ week1 $ week2 $ week4 $ count @@;
   datalines;
mild std n n n 16    severe std n n n  2
mild std n n a 13    severe std n n a  2
mild std n a n  9    severe std n a n  8
mild std n a a  3    severe std n a a  9
mild std a n n 14    severe std a n n  9
mild std a n a  4    severe std a n a 15
mild std a a n 15    severe std a a n 27
mild std a a a  6    severe std a a a 28
mild new n n n 31    severe new n n n  7
mild new n n a  0    severe new n n a  2
mild new n a n  6    severe new n a n  5
mild new n a a  0    severe new n a a  2
mild new a n n 22    severe new a n n 31
mild new a n a  2    severe new a n a  5
mild new a a n  9    severe new a a n 32
mild new a a a  0    severe new a a a  6
;
```

The analysis is directed at assessing the effect of the repeated measurement factor, Time, as well as the independent variables, Diagnosis (mild or severe) and Treatment (std or new). The RESPONSE statement is used to compute the logits of the marginal probabilities. The times used in the design matrix (0, 1,

2) correspond to the logarithms (base 2) of the actual times (1, 2, 4). The following statements produce Output 36.8.1 through Output 36.8.4:

```
proc catmod data=growth2 order=data;
   title2 'Reduced Logistic Model';
   weight count;
   population Diagnosis Treatment;
   response logit;
   model week1*week2*week4=(1 0 0 0,  /* mild, std */
                            1 0 1 0,
                            1 0 2 0,

                            1 0 0 0,  /* mild, new */
                            1 0 0 1,
                            1 0 0 2,

                            0 1 0 0,  /* severe, std */
                            0 1 1 0,
                            0 1 2 0,

                            0 1 0 0,  /* severe, new */
                            0 1 0 1,
                            0 1 0 2)
          (1='Mild diagnosis, week 1',
           2='Severe diagnosis, week 1',
           3='Time effect for std trt',
           4='Time effect for new trt')
           / freq design;
   contrast 'Diagnosis effect, week 1' all_parms 1 -1 0 0;
   contrast 'Equal time effects' all_parms 0 0 1 -1;
quit;
```

The samples and the response numbers are defined in Output 36.8.1, and the frequency distribution of the response numbers within the samples is displayed.

Output 36.8.1 Logistic Analysis of Growth Curve

Growth Curve Analysis
Reduced Logistic Model

The CATMOD Procedure

Data Summary			
Response	week1*week2*week4	**Response Levels**	8
Weight Variable	count	**Populations**	4
Data Set	GROWTH2	**Total Frequency**	340
Frequency Missing	0	**Observations**	29

Output 36.8.1 *continued*

Population Profiles			
Sample	**Diagnosis**	**Treatment**	**Sample Size**
1	mild	std	80
2	mild	new	70
3	severe	std	100
4	severe	new	90

Response Profiles			
Response	**week1**	**week2**	**week4**
1	n	n	n
2	n	n	a
3	n	a	n
4	n	a	a
5	a	n	n
6	a	n	a
7	a	a	n
8	a	a	a

Output 36.8.2 displays the design matrix specified in the MODEL statement, and the observed logits of the marginal probabilities are displayed in the Response Function column.

Output 36.8.2 Response Frequencies

Response Frequencies								
	Response Number							
Sample	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	16	13	9	3	14	4	15	6
2	31	0	6	0	22	2	9	0
3	2	2	8	9	9	15	27	28
4	7	2	5	2	31	5	32	6

Response Functions and Design Matrix						
			Design Matrix			
Sample	**Function Number**	**Response Function**	**1**	**2**	**3**	**4**
1	**1**	0.05001	1	0	0	0
	2	0.35364	1	0	1	0
	3	0.73089	1	0	2	0
2	**1**	0.11441	1	0	0	0
	2	1.29928	1	0	0	1
	3	3.52636	1	0	0	2
3	**1**	-1.32493	0	1	0	0
	2	-0.94446	0	1	1	0
	3	-0.16034	0	1	2	0
4	**1**	-1.53148	0	1	0	0
	2	0.00000	0	1	0	1
	3	1.60944	0	1	0	2

The analysis of variance table in Output 36.8.3 shows that the data can be adequately modeled by two parameters that represent diagnosis effects at week 1 and two log-linear time effects (one for each treatment). Both of the time effects are significant.

Since the estimate of the logit for the severe diagnosis effect (parameter 2) is more negative than it is for the mild diagnosis effect (parameter 1), there is a smaller predicted probability of the first response (normal) for the severe diagnosis group.

Output 36.8.3 ANOVA and Parameter Estimates

Analysis of Variance

Source	DF	Chi-Square	Pr > ChiSq
Mild diagnosis, week 1	1	0.28	0.5955
Severe diagnosis, week 1	1	100.48	<.0001
Time effect for std trt	1	26.35	<.0001
Time effect for new trt	1	125.09	<.0001
Residual	8	4.20	0.8387

Analysis of Weighted Least Squares Estimates

Effect	Parameter	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Model	**1**	-0.0716	0.1348	0.28	0.5955
	2	-1.3529	0.1350	100.48	<.0001
	3	0.4944	0.0963	26.35	<.0001
	4	1.4552	0.1301	125.09	<.0001

The analysis of contrasts (Output 36.8.4) shows that the diagnosis effect at week 1 is highly significant. In other words, those subjects with a severe diagnosis have a significantly higher probability of abnormal response at week 1 than those subjects with a mild diagnosis.

Output 36.8.4 Contrasts

Analysis of Contrasts

Contrast	DF	Chi-Square	Pr > ChiSq
Diagnosis effect, week 1	1	77.02	<.0001
Equal time effects	1	59.12	<.0001

The analysis of contrasts (Output 36.8.4) also shows that the time effect for the standard treatment is significantly different from the one for the new treatment. The table of parameter estimates (Output 36.8.3) shows that the time effect for the new treatment (parameter 4) is stronger than it is for the standard treatment (parameter 3).

Example 36.9: Repeated Measures, Two Repeated Measurement Factors

(View the complete code for this example (catex9.sas) in the example repository.)

This example, from MacMillan et al. (1981), illustrates a repeated measures analysis in which there are two repeated measurement factors. Two diagnostic procedures (standard and test) are performed on each subject, and the results of both are evaluated at each of two times as being positive or negative. In the following DATA step, std1 and std2 are the two measurements of the standard procedure, and test1 and test2 are the two measurements of the test procedure:

```
data a;
   input std1 $ test1 $ std2 $ test2 $ wt @@;
   datalines;
neg neg neg neg 509  neg neg neg pos  4  neg neg pos neg  17
neg neg pos pos   3  neg pos neg neg 13  neg pos neg pos   8
neg pos pos pos   8  pos neg neg neg 14  pos neg neg pos   1
pos neg pos neg  17  pos neg pos pos  9  pos pos neg neg   7
pos pos neg pos   4  pos pos pos neg  9  pos pos pos pos 170
;
```

For the initial model, the response functions are marginal probabilities, and the repeated measurement factors are Time and Treatment. The model is a saturated one, containing effects for Time, Treatment, and Time*Treatment. The following statements produce Output 36.9.1:

```
proc catmod data=a;
   title2 'Marginal Symmetry, Saturated Model';
   weight wt;
   response marginals;
   model std1*test1*std2*test2=_response_ / freq design noparm;
   repeated Time 2, Treatment 2 / _response_=Time Treatment
            Time*Treatment;
run;
```

The analysis of variance table in Output 36.9.1 shows that there is no significant effect of Time, either by itself or in its interaction with Treatment. The second model includes only the Treatment effect. Again, the response functions are marginal probabilities, and the repeated measurement factors are Time and Treatment.

Output 36.9.1 Diagnosis Data: Two Repeated Measurement Factors

Diagnostic Procedure Comparison
Marginal Symmetry, Saturated Model

The CATMOD Procedure

Data Summary			
Response	std1*test1*std2*test2	**Response Levels**	15
Weight Variable	wt	**Populations**	1
Data Set	A	**Total Frequency**	793
Frequency Missing	0	**Observations**	15

Population Profiles	
Sample	**Sample Size**
1	793

Output 36.9.1 *continued*

Response Profiles				
Response	std1	test1	std2	test2
1	neg	neg	neg	neg
2	neg	neg	neg	pos
3	neg	neg	pos	neg
4	neg	neg	pos	pos
5	neg	pos	neg	neg
6	neg	pos	neg	pos
7	neg	pos	pos	pos
8	pos	neg	neg	neg
9	pos	neg	neg	pos
10	pos	neg	pos	neg
11	pos	neg	pos	pos
12	pos	pos	neg	neg
13	pos	pos	neg	pos
14	pos	pos	pos	neg
15	pos	pos	pos	pos

Response Frequencies														
	Response Number													
Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	509	4	17	3	13	8	8	14	1	17	9	7	4	9	170

Response Functions and Design Matrix						
			Design Matrix			
Sample	Function Number	Response Function	1	2	3	4
1	1	0.70870	1	1	1	1
	2	0.72383	1	1	-1	-1
	3	0.70618	1	-1	1	-1
	4	0.73897	1	-1	-1	1

Analysis of Variance			
Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	2385.34	<.0001
Time	1	0.85	0.3570
Treatment	1	8.20	0.0042
Time*Treatment	1	2.40	0.1215
Residual	0	.	.

A main effect model with respect to Treatment is fit. The following statements produces Output 36.9.2:

```
    title2 'Marginal Symmetry, Reduced Model';
    model std1*test1*std2*test2=_response_ / corrb design noprofile;
    repeated Time 2, Treatment 2 / _response_=Treatment;
run;
```

The analysis of variance table for the reduced model (Output 36.9.2) shows that the model fits (since the

residual chi-square is nonsignificant) and that the treatment effect is significant. The negative parameter estimate for Treatment shows that the first level of treatment (std) has a smaller probability of the first response level (neg) than the second level of treatment (test). In other words, the standard diagnostic procedure gives a significantly higher probability of a positive response than the test diagnostic procedure.

Output 36.9.2 Diagnosis Data: Reduced Model

Diagnostic Procedure Comparison
Marginal Symmetry, Reduced Model

The CATMOD Procedure

Data Summary			
Response	std1*test1*std2*test2	**Response Levels**	15
Weight Variable	wt	**Populations**	1
Data Set	A	**Total Frequency**	793
Frequency Missing	0	**Observations**	15

Response Functions and Design Matrix

Sample	Function Number	Response Function	Design Matrix 1	Design Matrix 2
1	**1**	0.70870	1	1
	2	0.72383	1	-1
	3	0.70618	1	1
	4	0.73897	1	-1

Analysis of Variance

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	2386.97	<.0001
Treatment	1	9.55	0.0020
Residual	2	3.51	0.1731

Analysis of Weighted Least Squares Estimates

Effect	Parameter	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	**1**	0.7196	0.0147	2386.97	<.0001
Treatment	**2**	-0.0128	0.00416	9.55	0.0020

Correlation Matrix of the Parameter Estimates

Row	Col1	Col2
1	1.00000	0.04194
2	0.04194	1.00000

The next example illustrates a RESPONSE statement that, at each time, computes the sensitivity and specificity of the test diagnostic procedure with respect to the standard procedure. Since these are measures of the relative accuracy of the two diagnostic procedures, the repeated measurement factors in this case are labeled Time and Accuracy. Only 15 of the 16 possible responses are observed, so additional care must be

taken in formulating the RESPONSE statement for computation of sensitivity and specificity.

The following statements produce Output 36.9.3 and Output 36.9.4:

```
    title2 'Sensitivity and Specificity Analysis, '
           'Main-Effects Model';
    model std1*test1*std2*test2=_response_ / covb design noprofile;
    repeated Time 2, Accuracy 2 / _response_=Time Accuracy;
    response exp 1 -1  0  0  0  0  0  0,
                 0  0  1 -1  0  0  0  0,
                 0  0  0  0  1 -1  0  0,
                 0  0  0  0  0  0  1 -1

             log 0 0 0 0   0 0  0   0 0 0 0   1 1 1 1,
                 0 0 0 0   0 0  0   1 1 1 1   1 1 1 1,
                 1 1 1 1   0 0  0   0 0 0 0   0 0 0 0,
                 1 1 1 1   1 1  1   0 0 0 0   0 0 0 0,
                 0 0 0 1   0 0  1   0 0 0 1   0 0 0 1,
                 0 0 1 1   0 0  1   0 0 1 1   0 0 1 1,
                 1 0 0 0   1 0  0   1 0 0 0   1 0 0 0,
                 1 1 0 0   1 1  0   1 1 0 0   1 1 0 0;
quit;
```

For the sensitivity and specificity analysis, the four response functions displayed next to the design matrix (Output 36.9.3) represent the following:

1. sensitivity, time 1
2. specificity, time 1
3. sensitivity, time 2
4. specificity, time 2

The sensitivities and specificities are for the test diagnostic procedure relative to the standard procedure.

Output 36.9.3 Diagnosis Data: Sensitivity and Specificity Analysis

Diagnostic Procedure Comparison
Sensitivity and Specificity Analysis, Main-Effects Model

The CATMOD Procedure

Data Summary			
Response	std1*test1*std2*test2	**Response Levels**	15
Weight Variable	wt	**Populations**	1
Data Set	A	**Total Frequency**	793
Frequency Missing	0	**Observations**	15

Output 36.9.3 *continued*

Response Functions and Design Matrix

Sample	Function Number	Response Function	Design Matrix 1	2	3
1	1	0.82251	1	1	1
	2	0.94840	1	1	-1
	3	0.81545	1	-1	1
	4	0.96964	1	-1	-1

Analysis of Variance

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	6448.79	<.0001
Time	1	4.10	0.0428
Accuracy	1	38.81	<.0001
Residual	1	1.00	0.3178

The ANOVA table in Output 36.9.3 shows that an additive model fits, that there is a significant effect of time, and that the sensitivity is significantly different from the specificity.

Output 36.9.4 shows that the predicted sensitivities and specificities are lower for time 1 (since parameter 2 is negative). It also shows that the sensitivity is significantly less than the specificity.

Output 36.9.4 Parameter Estimates

Analysis of Weighted Least Squares Estimates

Effect	Parameter	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	0.8892	0.0111	6448.79	<.0001
Time	2	-0.00932	0.00460	4.10	0.0428
Accuracy	3	-0.0702	0.0113	38.81	<.0001

Covariance Matrix of the Parameter Estimates

Row	Col1	Col2	Col3
1	0.00012260	0.00000229	0.00010137
2	0.00000229	0.00002116	-.00000587
3	0.00010137	-.00000587	0.00012697

Example 36.10: Direct Input of Response Functions and Covariance Matrix

(View the complete code for this example (catex10.sas) in the example repository.)

This example illustrates the ability of PROC CATMOD to operate on an existing vector of functions and the corresponding covariance matrix. The estimates under investigation are composite indices summarizing the responses to 18 psychological questions pertaining to general well-being. These estimates are computed for domains corresponding to an age-by-sex cross-classification, and the covariance matrix is calculated using the method of balanced repeated replications. The analysis is directed at obtaining a description of the variation among these domain estimates. The data are from Koch and Stokes (1979).

In the following statements, the first row of the fbeing data set contains the response functions for the variables b1–b10, while the remaining rows contain the covariance matrix. From the PROC CATMOD statements, the READ option in the RESPONSE statement says that you are inputting the response functions and their covariance matrix, while the PROFILE= option in the FACTORS statement tells you that the variables b1–b5 correspond to the effects for sex='male' at the five different age groupings, and b6–b10 likewise correspond to the effects for sex='female'. See the section "Inputting Response Functions and Covariances Directly" on page 2272 for more information about using the READ option.

```
data fbeing(type=est);
   input   b1-b5   _type_ $  _name_ $  b6-b10 #2;
   datalines;
 7.93726   7.92509   7.82815   7.73696   8.16791  parms    .
 7.24978   7.18991   7.35960   7.31937   7.55184
 0.00739   0.00019   0.00146  -0.00082   0.00076  cov      b1
 0.00189   0.00118   0.00140  -0.00140   0.00039
 0.00019   0.01172   0.00183   0.00029   0.00083  cov      b2
-0.00123  -0.00629  -0.00088  -0.00232   0.00034
 0.00146   0.00183   0.01050  -0.00173   0.00011  cov      b3
 0.00434  -0.00059  -0.00055   0.00023  -0.00013
-0.00082   0.00029  -0.00173   0.01335   0.00140  cov      b4
 0.00158   0.00212   0.00211   0.00066   0.00240
 0.00076   0.00083   0.00011   0.00140   0.01430  cov      b5
-0.00050  -0.00098   0.00239  -0.00010   0.00213
 0.00189  -0.00123   0.00434   0.00158  -0.00050  cov      b6
 0.01110   0.00101   0.00177  -0.00018  -0.00082
 0.00118  -0.00629  -0.00059   0.00212  -0.00098  cov      b7
 0.00101   0.02342   0.00144   0.00369   0.00253
 0.00140  -0.00088  -0.00055   0.00211   0.00239  cov      b8
 0.00177   0.00144   0.01060   0.00157   0.00226
-0.00140  -0.00232   0.00023   0.00066  -0.00010  cov      b9
-0.00018   0.00369   0.00157   0.02298   0.00918
 0.00039   0.00034  -0.00013   0.00240   0.00213  cov     b10
-0.00082   0.00253   0.00226   0.00918   0.01921
;
```

The following statements produce Output 36.10.1:

```
proc catmod data=fbeing;
   title 'Complex Sample Survey Analysis';
   response read b1-b10;
   factors sex $ 2, age $ 5 / _response_=sex age
```

```
                                profile=(male     '25-34',
                                         male     '35-44',
                                         male     '45-54',
                                         male     '55-64',
                                         male     '65-74',
                                         female   '25-34',
                                         female   '35-44',
                                         female   '45-54',
                                         female   '55-64',
                                         female   '65-74');
   model _f_=_response_
         / design title='Main Effects for Sex and Age';
run;
```

Output 36.10.1 Health Survey Data: Using Direct Input

Complex Sample Survey Analysis

Main Effects for Sex and Age

The CATMOD Procedure

Response Functions and Design Matrix

Sample	Function Number	Response Function	Design Matrix 1	2	3	4	5	6
1	1	7.93726	1	1	1	0	0	0
	2	7.92509	1	1	0	1	0	0
	3	7.82815	1	1	0	0	1	0
	4	7.73696	1	1	0	0	0	1
	5	8.16791	1	1	-1	-1	-1	-1
	6	7.24978	1	-1	1	0	0	0
	7	7.18991	1	-1	0	1	0	0
	8	7.35960	1	-1	0	0	1	0
	9	7.31937	1	-1	0	0	0	1
	10	7.55184	1	-1	-1	-1	-1	-1

Analysis of Variance

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	28089.07	<.0001
sex	1	65.84	<.0001
age	4	9.21	0.0561
Residual	4	2.92	0.5713

Analysis of Weighted Least Squares Estimates

Effect	Parameter	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	7.6319	0.0455	28089.07	<.0001
sex	2	0.2900	0.0357	65.84	<.0001
age	3	-0.00780	0.0645	0.01	0.9037
	4	-0.0465	0.0636	0.54	0.4642
	5	-0.0343	0.0557	0.38	0.5387
	6	-0.1098	0.0764	2.07	0.1506

The analysis of variance table in Output 36.10.1 shows that the additive model fits and that there is a significant effect of both sex and age. The following statements produce Output 36.10.2:

```
    contrast 'No Age Effect for Age<65' all_parms 0 0 1 0 0 -1,
                                       all_parms 0 0 0 1 0 -1,
                                       all_parms 0 0 0 0 1 -1;
run;
```

The analysis of the contrast shows that there is no significant difference among the four age groups that are under age 65.

Output 36.10.2 Health Survey Data: Age<65 Contrast

Complex Sample Survey Analysis

Main Effects for Sex and Age

The CATMOD Procedure

Analysis of Contrasts			
Contrast	DF	Chi-Square	Pr > ChiSq
No Age Effect for Age<65	3	0.72	0.8678

The next model contains a binary age effect (under 65 versus 65 and over). The following statements produce Output 36.10.3:

```
    model _f_=(1  1  1,
               1  1  1,
               1  1  1,
               1  1  1,
               1  1 -1,
               1 -1  1,
               1 -1  1,
               1 -1  1,
               1 -1  1,
               1 -1 -1)
                       (1='Intercept' ,
                        2='Sex'       ,
                        3='Age (25-64 vs. 65-74)')
          / design title='Binary Age Effect (25-64 vs. 65-74)' ;
run;
quit;
```

Output 36.10.3 Health Survey Data: Age<65 Model

Complex Sample Survey Analysis

Binary Age Effect (25-64 vs. 65-74)

The CATMOD Procedure

Response Functions and Design Matrix

Sample	Function Number	Response Function	Design Matrix 1	2	3
1	1	7.93726	1	1	1
	2	7.92509	1	1	1
	3	7.82815	1	1	1
	4	7.73696	1	1	1
	5	8.16791	1	1	-1
	6	7.24978	1	-1	1
	7	7.18991	1	-1	1
	8	7.35960	1	-1	1
	9	7.31937	1	-1	1
	10	7.55184	1	-1	-1

Analysis of Variance

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	19087.16	<.0001
Sex	1	72.64	<.0001
Age (25-64 vs. 65-74)	1	8.49	0.0036
Residual	7	3.64	0.8198

Analysis of Weighted Least Squares Estimates

Effect	Parameter	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Model	1	7.7183	0.0559	19087.16	<.0001
	2	0.2800	0.0329	72.64	<.0001
	3	-0.1304	0.0448	8.49	0.0036

The analysis of variance table in Output 36.10.3 shows that the model fits (note that the goodness-of-fit statistic is the sum of the previous one (Output 36.10.1) plus the chi-square for the contrast matrix in Output 36.10.2). The age and sex effects are significant. Since the second parameter in the table of estimates is positive, males (the first level for the sex variable) have a higher predicted index of well-being than females. Since the third parameter estimate is negative, those younger than age 65 (the first level of age) have a lower predicted index of well-being than those 65 and older.

Example 36.11: Predicted Probabilities

(View the complete code for this example (catex11.sas) in the example repository.)

Suppose you have collected marketing research data to examine the relationship between a prospect's likelihood of buying your product and the person's education and income. Specifically, the variables are as follows:

Variable	Levels	Interpretation
Education	high, low	Prospect's education level
Income	high, low	Prospect's income level
Purchase	yes, no	Did prospect purchase product?

The following statements first create a data set, loan, that contains the marketing research data. Then the CATMOD procedure fits a model, obtains the parameter estimates, and obtains the predicted probabilities of interest. These statements produce Output 36.11.1 and Output 36.11.2.

```
data loan;
   input Education $ Income $ Purchase $ wt;
   datalines;
high  high  yes    54
high  high  no     23
high  low   yes    41
high  low   no     12
low   high  yes    35
low   high  no     42
low   low   yes    19
low   low   no      8
;

ods output PredictedValues=Predicted (keep=Education Income PredFunction);
proc catmod data=loan order=data;
   weight wt;
   response marginals;
   model Purchase=Education Income / pred design;
run;

proc sort data=Predicted;
   by descending PredFunction;
run;
proc print data=Predicted;
run;
```

Notice that the preceding statements use the Output Delivery System (ODS) to output the parameter estimates instead of the OUT= option, though either can be used.

Output 36.11.1 Marketing Research Data: Obtaining Predicted Probabilities

The CATMOD Procedure

Data Summary			
Response	Purchase	**Response Levels**	2
Weight Variable	wt	**Populations**	4
Data Set	LOAN	**Total Frequency**	234
Frequency Missing	0	**Observations**	8

Population Profiles

Sample	Education	Income	Sample Size
1	high	high	77
2	high	low	53
3	low	high	77
4	low	low	27

Response Profiles

Response	Purchase
1	yes
2	no

Response Functions and Design Matrix

Sample	Response Function	Design Matrix 1	2	3
1	0.70130	1	1	1
2	0.77358	1	1	-1
3	0.45455	1	-1	1
4	0.70370	1	-1	-1

Analysis of Variance

Source	DF	Chi-Square	Pr > ChiSq
Intercept	1	418.36	<.0001
Education	1	8.85	0.0029
Income	1	4.70	0.0302
Residual	1	1.84	0.1745

Analysis of Weighted Least Squares Estimates

Parameter		Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept		0.6481	0.0317	418.36	<.0001
Education	**high**	0.0924	0.0311	8.85	0.0029
Income	**high**	-0.0675	0.0312	4.70	0.0302

Output 36.11.1 *continued*

Predicted Values for Response Functions							
			Observed		Predicted		
Education	Income	Function Number	Function	Standard Error	Function	Standard Error	Residual
high	high	1	0.701299	0.052158	0.67294	0.047794	0.028359
high	low	1	0.773585	0.057487	0.808034	0.051586	-0.03445
low	high	1	0.454545	0.056744	0.48811	0.051077	-0.03356
low	low	1	0.703704	0.087877	0.623204	0.064867	0.080499

Output 36.11.2 Predicted Probabilities Data Set

Obs	Education	Income	PredFunction
1	high	low	0.808034
2	high	high	0.67294
3	low	low	0.623204
4	low	high	0.48811

You can use the predicted values (values of PredFunction in Output 36.11.2) as scores representing the likelihood that a randomly chosen subject from one of these populations will purchase the product. Notice that the "Response Profiles" table in Output 36.11.1 shows you that the first sorted level of Purchase is 'yes', indicating that the predicted probabilities are for Pr(Purchase='yes'). For example, someone with high education and low income has an estimated probability of purchase of 0.808. Like any response function estimate given by PROC CATMOD, this estimate can be obtained by cross-multiplying the row from the design matrix corresponding to the sample (sample number 2 in this case) with the vector of parameter estimates: $(1 * 0.6481) + (1 * 0.0924) + (-1 * (-0.0675))$.

This ranking of scores can help in decision making (for example, with respect to allocation of advertising dollars, choice of advertising media, choice of print media, and so on).

References

Agresti, A. (1984). *Analysis of Ordinal Categorical Data*. New York: John Wiley & Sons.

Agresti, A. (1990). *Categorical Data Analysis*. New York: John Wiley & Sons.

Agresti, A. (1996). *An Introduction to Categorical Data Analysis*. New York: John Wiley & Sons.

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1975). *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, MA: MIT Press.

Christensen, R. (1997). *Log-Linear Models and Logistic Regression*. 2nd ed. New York: Springer-Verlag.

Cox, D. R., and Snell, E. J. (1989). *The Analysis of Binary Data*. 2nd ed. London: Chapman & Hall.

Fienberg, S. E. (1980). *The Analysis of Cross-Classified Categorical Data*. 2nd ed. Cambridge, MA: MIT Press.

Forthofer, R. N., and Koch, G. G. (1973). "An Analysis of Compounded Functions of Categorical Data." *Biometrics* 29:143–157.

Forthofer, R. N., and Lehnen, R. G. (1981). *Public Program Analysis: A New Categorical Data Approach*. Belmont, CA: Wadsworth.

Freeman, D. H., Jr. (1987). *Applied Categorical Data Analysis*. New York: Marcel Dekker.

Grizzle, J. E., Starmer, C. F., and Koch, G. G. (1969). "Analysis of Categorical Data by Linear Models." *Biometrics* 25:489–504.

Guthrie, D. (1981). "Analysis of Dichotomous Variables in Repeated Measures Experiments." *Psychological Bulletin* 90:189–195.

Haberman, S. J. (1972). "Algorithm AS 51: Log-Linear Fit for Contingency Tables." *Journal of the Royal Statistical Society, Series C* 21:218–225.

Haslett, S. (1990). "Degrees of Freedom and Parameter Estimability in Hierarchical Models for Sparse Complete Contingency Tables." *Computational Statistics and Data Analysis* 9:179–195.

Imrey, P. B., Koch, G. G., and Stokes, M. E. (1981). "Categorical Data Analysis: Some Reflections on the Log Linear Model and Logistic Regression, Part I: Historical and Methodological Overview." *International Statistical Review* 49:265–283.

Koch, G. G., Landis, J. R., Freeman, J. L., Freeman, D. H., and Lehnen, R. G. (1977). "A General Methodology for the Analysis of Experiments with Repeated Measurement of Categorical Data." *Biometrics* 33:133–158.

Koch, G. G., and Stokes, M. E. (1979). *Annotated Computer Applications of Weighted Least Squares Methods for Illustrative Analyses of Examples Involving Health Survey Data*. Technical report, prepared for the U.S. National Center for Health Studies.

Landis, J. R., Stanish, W. M., Freeman, J. L., and Koch, G. G. (1976). "A Computer Program for the Generalized Chi-Square Analysis of Categorical Data Using Weighted Least Squares (GENCAT)." *Computer Programs in Biomedicine* 6:196–231.

MacMillan, J., Becker, C., Koch, G. G., Stokes, M. E., and Vandiviere, H. M. (1981). "An Application of Weighted Least Squares Methods to the Analysis of Measurement Process Components of Variability in an Observational Study." In *Proceedings of Survey Research Methods*, 680–685. Alexandria, VA: American Statistical Association.

Ries, P. N., and Smith, H. (1963). "The Use of Chi-Square for Preference Testing in Multidimensional Problems." *Chemical Engineering Progress* 59:39–43.

Searle, S. R. (1971). "Topics in Variance Component Estimation." *Biometrics* 26:1–76.

Stanish, W. M., and Koch, G. G. (1984). "The Use of CATMOD for Repeated Measurement Analysis of Categorical Data." In *Proceedings of the Ninth Annual SAS Users Group International Conference*, 761–770. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI84/Sugi-84-142%20Stanish%20Koch.pdf.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2000). *Categorical Data Analysis Using the SAS System*. 2nd ed. Cary, NC: SAS Institute Inc.

Wald, A. (1943). "Tests of Statistical Hypotheses Concerning Several Parameters When the Number of Observations Is Large." *Transactions of the American Mathematical Society* 54:426–482.

Chapter 37
The CAUSALGRAPH Procedure

Contents

Overview: CAUSALGRAPH Procedure **2354**
- Features of the CAUSALGRAPH Procedure 2355

Getting Started: CAUSALGRAPH Procedure **2357**

Syntax: CAUSALGRAPH Procedure **2360**
- PROC CAUSALGRAPH Statement 2360
- IDENTIFY Statement 2366
- MODEL Statement 2367
- PATHDIAGRAM Statement 2372
- TESTID Statement 2375
- UNMEASURED Statement 2377

Details: CAUSALGRAPH Procedure **2377**
- Statistical and Causal Concepts 2378
- Causal Graph Theory 2378
- Causal Graphs and Potential Outcomes 2384
- Choosing an Identification Strategy 2385
- Arrow or Edge Specification 2386
- INMODEL= and OUTMODEL= Data Sets 2386
- Limitations of the CAUSALGRAPH Procedure 2387
- Time Requirements 2388
- ODS Table Names 2389

Examples: CAUSALGRAPH Procedure **2390**
- Example 37.1: Constructing Adjustment Sets 2390
- Example 37.2: Searching Efficiently for an Adjustment Set 2394
- Example 37.3: Testing Adjustments and Enumerating Paths 2396
- Example 37.4: Finding Adjustment Sets Common to Multiple Models 2401
- Example 37.5: Identifying a Causal Effect by Using Instrumental Variables 2404
- Example 37.6: Distinguishing Models with Data 2406
- Example 37.7: Applying an Adjustment Set to Estimate a Causal Effect from Data . . 2410

References **2417**

Overview: CAUSALGRAPH Procedure

The CAUSALGRAPH procedure examines the structure of graphical causal models and suggests statistical strategies or steps that enable researchers to estimate causal effects that have valid causal interpretations. Causal models are encoded in the form of directed acyclic graphs (Pearl 2009a, b), which are the primary input for the procedure. Henceforth, the input causal graph or diagram for the procedure is assumed to be a directed acyclic graph.

A causal graph depicts the causal relationships among variables in the context of a specific data generating process, which includes observational and experimental situations. Either the data have been collected previously or they will be collected in the future. The focus of a causal graph is usually on the relationship between the treatment variables and the outcome variables. Specifically, one of the primary goals of causal graph analysis is to determine how to estimate the causal effects of the treatment variables on the outcome variables.

Although the causal effect that is defined in PROC CAUSALGRAPH is called a treatment effect, it is not confined to effects that result from controlled treatments (such as effects in an experiment). Rather, the treatment might represent an intervention (such as smoking cessation versus control), an exposure to a condition (such as an infectious agent), or an existing characteristic of subjects (such as high versus low socioeconomic status). For example, the analysis in the section "Getting Started: CAUSALGRAPH Procedure" on page 2357 considers the effect of maternal exposure to persistent perfluoroalkyl substances on the duration of breastfeeding, and the analysis in Example 37.3 considers the link between an individual's measured serum urate and risk of cardiovascular disease.

Given the input causal graph and the treatment and outcome variables of interest, you can use the CAUSALGRAPH procedure to explore some formal properties of causal graphs, including the following:

- the causal and noncausal (associative) proper paths between the treatment and outcome variables
- the adjustment sets of variables that can be used to remove or block the spurious or confounding associations between the treatment and outcome variables
- the set of variables that can be used as instruments to estimate the causal treatment effect of interest

Essentially, because of the possible presence of spurious associations and unmeasured confounding, you cannot determine the causal effect of the treatment directly. Identifying and removing spurious association are especially important in observational studies where the treatment conditions are not randomly assigned to individuals. PROC CAUSALGRAPH uses the formal properties of a causal graph to suggest statistical strategies that can lead to unbiased estimation of causal treatment effects. For example, the list of adjustment covariates that the procedure produces can be used as input for an appropriate statistical procedure, such as PROC CAUSALTRT or PROC PSMATCH, to estimate the magnitude of a causal effect. For more information, see Chapter 39, "The CAUSALTRT Procedure," or Chapter 101, "The PSMATCH Procedure." The estimation process is also illustrated by the analysis in Example 37.7.

The CAUSALGRAPH procedure suggests statistical strategies that are based on the assumptions of a causal model that specifies the relationships between variables of interest (Elwert 2013). This causal model supplements the available data and cannot be discerned from those data (Pearl 2009b). Therefore, specifying a causal graph that accurately reflects the data generating process is essential in causal graph analysis.

Domain-specific knowledge and familiarity with the data collection process can help you produce reasonably accurate causal graphs.

Using PROC CAUSALGRAPH to examine the identifiability of causal effects in causal graphs requires an understanding of the terminology, concepts, and assumptions of graphical causal models. For more information about these technical aspects, see the section "Details: CAUSALGRAPH Procedure" on page 2377.

Features of the CAUSALGRAPH Procedure

PROC CAUSALGRAPH provides several criteria for identifying causal treatment effects. In particular, you can use the METHOD= option in the PROC CAUSALGRAPH statement to specify any one of the following identification criteria:

- constructive backdoor criterion (Van der Zander, Liśkiewicz, and Textor 2014)
- backdoor criterion (Pearl 2009b)
- parents-of-treatment criterion (Elwert 2013)
- parents-of-outcome criterion (Elwert 2013)
- joint ancestor criterion (Elwert 2013)
- instrumental variables (Van der Zander, Textor, and Liśkiewicz 2015)

The constructive backdoor criterion (METHOD=ADJUSTMENT), also called the adjustment criterion, finds all valid adjustment sets that consist of observed variables only. The backdoor criterion (METHOD=BACKDOOR) similarly finds adjustment sets that consist of observed variables, but with a slightly stronger criterion. The backdoor criterion is computationally more efficient than the adjustment criterion, but it might not find every possible valid adjustment set. The appeal of the backdoor criterion is that it has an intuitive interpretation and provides a fast method of constructing valid adjustment sets (Elwert 2013). The parents-of-treatment (METHOD=PTREATMENT), parents-of-outcome (METHOD=POUTCOME), and joint ancestor (METHOD=ANCESTOR) criteria are similar to the backdoor criterion in that they place stronger restrictions on the possible adjustment sets in order to achieve greater computational efficiency. The instrumental variables method (METHOD=IV) finds instrumental variables to deal with the presence of unmeasured confounding between the treatment and outcome variables. This is useful because unmeasured confounding is a situation in which the other criteria might fail.

To identify the sets of adjustment covariates or instrumental variables, PROC CAUSALGRAPH has two primary modes of operation:

- The LIST option in the PROC CAUSALGRAPH statement enables you to enumerate the criteria under which it is possible to estimate a causal effect.
- The TESTID statement specification enables you to test whether a user-specified criterion is valid for estimating a causal effect.

You can use both of these modes in a single run of the procedure. Various options are available to fine-tune the output listing of the requested criteria. You can use these options to limit the number of identification strategies that are listed, sort the listed strategies, improve the searching and listing efficiency, and so on.

In the CAUSALGRAPH procedure, every causal model must be a directed acyclic graph (DAG). Each edge in a causal path represents a direct causal effect of one variable on another variable. For more information about the causal interpretation of directed graphs, see the section "Causal Graph Theory" on page 2378.

You can input causal graphs or models by using the MODEL statement. The MODEL statement supports a pathlike syntax to input causal relationships among variables. For example, to specify the causal path $X \rightarrow Y$, you can use either the **`X ==> Y`** or **`Y <== X`** syntax in the MODEL statement. You can also specify multiple causal relationships as a chain of causal paths: for example, **`X ==> Y ==> Z`**, **`Z <== X ==> Y <== W`**, and so on. You can also use the INMODEL= option to load one or more models from a data set that was created by a prior invocation of PROC CAUSALGRAPH.

PROC CAUSALGRAPH performs the following semantic validation checks for every model that you specify:

- The model should be weakly connected. That is, there should be a path between any pair of variables when all edges are treated as undirected.
- The model cannot contain any directed cycles.

The procedure also supports the specification of bidirected edges (or paths). A bidirected edge syntax, such as **`X <==> Y`** (for $X \leftrightarrow Y$), is interpreted as unmeasured confounding between the two variables, so that the graph is still a DAG. That is, **`X <==> Y`** is equivalent to **`X <== L ==> Y`** (for $X \leftarrow L \rightarrow Y$), where the node L represents some unmeasured variable, which you specify in the UNMEASURED statement.

It is important to distinguish between measured and unmeasured variables in a graphical model analysis. Variables that you list in the UNMEASURED statement are treated as unmeasured or unobserved. All other variables are treated as measured or observed. In order to make causal effect estimation meaningful, your treatment and outcome variables must always be measured. All other variables can either be measured or unmeasured. Unmeasured variables in a causal model cannot be included in a statistical analysis, and thus you cannot use them in any identification criterion for causal treatment effects.

In practice, there are several reasons why you might want to specify a variable to be unmeasured:

- The variable corresponds to a latent construct that cannot be measured.
- The variable was not measured or will not be measured in the data set for the causal effect estimation.
- The data that are collected for the variable are not considered reliable enough to include in the causal effect estimation.

Although PROC CAUSALGRAPH distinguishes between measured and unmeasured variables, it does not distinguish between classification and continuous variables in an analysis.

You can specify multiple causal models in a single run of the procedure by using multiple MODEL statements. This enables you to compare the identification criteria that are produced for alternative models. Or, as in Example 37.4, you can search for identification criteria that are simultaneously valid for all specified models.

The CAUSALGRAPH procedure also includes the following important features:

- identification of joint causal effects from multiple treatments on multiple outcomes. For more information about the interpretation of joint effects, see the section "Identifying Joint Treatment Effects" on page 2387.
- enumeration of all the observationally testable assumptions that are encoded by a causal model
- listing of causal and noncausal treatment-to-outcome proper paths that are blocked or nonblocked under specified adjustment sets of variables

- import and export of saved models
- ODS graphics to visualize causal models

Getting Started: CAUSALGRAPH Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/cagrgs1.sas.)

This example demonstrates how you can use the CAUSALGRAPH procedure to determine which covariates in a causal model you must control in order to estimate a treatment effect that has a valid causal interpretation.

The causal model shown in Figure 37.1 has been adapted from Timmermann et al. (2017) and examines the relationship between maternal exposure to persistent perfluoroalkyl substances (PFAS) and breastfeeding duration (Duration) among residents of the Faroe Islands. The model includes the following variables:

- PFAS: the treatment variable
- Duration: the outcome variable
- Age: age of the mother at the child's birth
- Education: indicator of whether the mother had any postprimary education
- Employment: a categorical variable that describes the employment condition of the mother (employed, unemployed, homemaker, and so on)
- Parity: indicator of whether this was the mother's first childbirth
- Alcohol: indicator of whether the mother consumed alcohol during the pregnancy
- Smoking: indicator of whether the mother smoked cigarettes during the pregnancy
- BMI: prepregnancy body mass index of the mother
- PrevBF: indicator of prior breastfeeding experience

The treatment (PFAS) and outcome (Duration) variables are shaded in Figure 37.1. For this example, it is assumed that the variables Alcohol and Smoking are not observed (for example, because the data are considered to be unreliable).

Figure 37.1 Causal Model of the Effect of Persistent Perfluoroalkyl Substances on Breastfeeding Duration

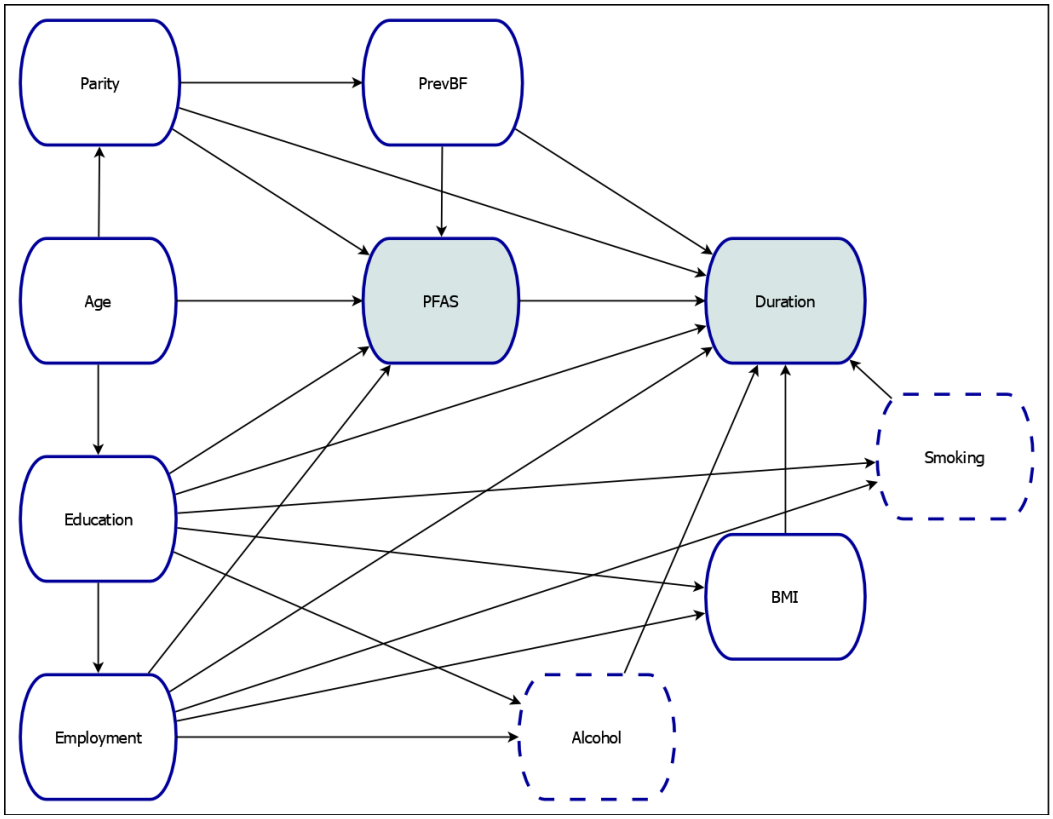

The statistical association between the variables PFAS and Duration that would be measured in an observational study reflects a combination of true causal association and additional spurious or noncausal association. In order to isolate the true causal association between PFAS and Duration, you must devise a strategy to eliminate the noncausal association. One way to do this is to find an adjustment set. You can use the CAUSALGRAPH procedure to construct all possible adjustment sets that can be used to identify the causal effect of PFAS on Duration, subject to the assumptions that are encoded in the causal model in Figure 37.1. If at least one such adjustment set exists, then it is possible to estimate the causal effect by using observational data. For more information about adjustment sets and identifying causal effects, see the section "Identification by Adjustment" on page 2382.

The following statements invoke PROC CAUSALGRAPH to define and analyze the causal model and construct the adjustment sets:

```
proc causalgraph;
   model "Timm17TwoLatent"
      Age ==> Parity PFAS Education,
      Parity ==> PrevBF Duration PFAS,
      PrevBF ==> PFAS Duration,
      PFAS ==> Duration,
      Education ==> Duration Employment PFAS BMI Alcohol Smoking,
      Employment ==> Duration PFAS BMI Alcohol Smoking,
      BMI Alcohol Smoking ==> Duration;
   identify PFAS ==> Duration;
   unmeasured Alcohol Smoking;
```

```
run;
```

In an analysis that uses PROC CAUSALGRAPH, you must specify at least one causal model in a MODEL statement. You can also specify multiple models. Each MODEL statement must begin with a quoted string that provides a unique name for the model. This example labels the model as **Timm17TwoLatent**, a reference to its original publication. The remainder of the MODEL statement specifies the variables and their causal relationships (as indicated by directed edges). In this example, the MODEL statement encodes the model shown in Figure 37.1.

In the IDENTIFY statement, you specify the causal effect of interest. You can use this statement to specify one or more treatment variables and one or more outcome variables. The treatment and outcome variables are separated by a single right arrow, **==>**. This example studies the causal effect of the variable PFAS on the variable Duration.

The UNMEASURED statement specifies variables that are not observed and thus cannot be included in any adjustment set. In this example, the variables Alcohol and Smoking are treated as unmeasured.

The output in Figure 37.2 summarizes the variables and edges in the causal model that is specified in the MODEL statement. You can use this information as a qualitative check of the model specification.

Figure 37.2 Input Summary Tables for the Causal Model in Figure 37.1

The CAUSALGRAPH Procedure

Variables in Model		
	N	**Variables**
Measured	8	Age BMI Duration Education Employment Parity PFAS PrevBF
Unmeasured	2	Alcohol Smoking

Graphical Model Summary						
Model	**Nodes**	**Edges**	**Treatments**	**Outcomes**	**Measured**	**Unmeasured**
Timm17TwoLatent	10	23	1	1	8	2

In this example, the CAUSALGRAPH procedure uses the constructive backdoor criterion (METHOD=ADJUSTMENT; see Van der Zander, Liśkiewicz, and Textor 2014) to construct all valid adjustments. You can change the default criterion by specifying the METHOD= option in the PROC CAUSALGRAPH statement.

The adjustment sets are displayed in Figure 37.3. For the model in Figure 37.1, there are four valid adjustment sets. Each row of Figure 37.3 contains an adjustment set, and the variables in each set are indicated in the table by an asterisk. Assuming that the causal model is accurate, you can estimate the causal effect of PFAS on Duration by using any one of these adjustment sets.

Figure 37.3 Adjustment Sets for the Causal Model in Figure 37.1

Covariate Adjustment Sets for Timm17TwoLatent Causal Effect of PFAS on Duration							
				Covariates			
	Size	**Minimal**	**Age**	**BMI**	**Education**	**Employment**	**Parity**	**PrevBF**
1	4	Yes			*	*	*	*
2	5	No	*		*	*	*	*
3	5	No		*	*	*	*	*
4	6	No	*	*	*	*	*	*

The table also indicates the size of each set and whether or not the set is minimal. An adjustment set is minimal if no proper subset of the set is also a valid adjustment set. In this example, there is one minimal set that contains four covariates that you must adjust for in order to estimate the specified causal effect. You can use one of these covariate adjustment sets as input for an appropriate statistical procedure, such as PROC PSMATCH or PROC CAUSALTRT, to estimate the magnitude of the specified causal effect. For an illustration of how you can use an adjustment set to estimate a causal effect, see Example 37.7.

By default, PROC CAUSALGRAPH constructs every possible adjustment set for the specified causal effect. You can use the MAXLIST=, MAXSIZE=, and MINIMAL options in the PROC CAUSALGRAPH statement to refine the adjustment sets that are computed. You can modify the displayed output by using the LIST, NOLIST, NOPRINT, or PSUMMARY option in the PROC CAUSALGRAPH statement.

Syntax: CAUSALGRAPH Procedure

The following statements are available in the CAUSALGRAPH procedure:

PROC CAUSALGRAPH < *options* > **;**
  MODEL '*label*' *path* < , *path* . . . > **;**
  IDENTIFY *effect-specification* **;**
  UNMEASURED *variables* **;**
  TESTID < '*label*' > *variables* < / *options* > **;**
  PATHDIAGRAM < *options* > **;**

You can specify only one UNMEASURED statement and only one IDENTIFY statement for each PROC CAUSALGRAPH statement. The following sections describe the PROC CAUSALGRAPH statement and then describe the other statements in alphabetical order.

PROC CAUSALGRAPH Statement

PROC CAUSALGRAPH < *options* > **;**

The PROC CAUSALGRAPH statement invokes the procedure. The *options* listed in Table 37.1 are available in the PROC CAUSALGRAPH statement.

Table 37.1 Options Available in the PROC CAUSALGRAPH Statement

Option	Description
Analysis Options	
COMMON	Requests adjustment sets common to all models
IMAP	Requests conditional independence assumptions
INMODEL=	Specifies a data set that contains one or more causal models
LIST	Requests possible identification criteria
METHOD=	Specifies the method to use for identification
MINIMAL	Requests only minimal adjustment sets
NOLIST	Excludes the construction of possible identification criteria

Table 37.1 *continued*

Option	Description
Output Options	
COMPACT	Suppresses the display of unused variables
CYCLES	Displays directed cycles
DISCONNECTED	Suppresses warnings for models that contain disjoint DAGs
MAXLIST=	Specifies the maximum number of identification criteria to print
MAXSIZE=	Specifies the maximum size of constructed adjustment sets
NOPRINT	Suppresses display of all output
NOSORT	Requests an unsorted list of adjustment sets
NTHREADS=	Specifies the maximum number of processing threads
ORDERMODELS	Orders the model output displays according to the model labels
OUTMODEL=	Creates an output data set that contains the causal models
PSUMMARY	Displays only the identification analysis summary

COMMON < (ONLY) >

requests adjustment sets common to all models. If you specify the COMMON option, the adjustment sets common to all models are computed in addition to the adjustment sets specific to each model. You can specify the COMMON(ONLY) option to compute only the adjustment sets common to all models. The COMMON option is ignored if you specify METHOD=IV.

COMPACT

suppresses the display of unused variables in a table. By default, the procedure prints one column for every variable when adjustment sets or conditional instrumental variables are displayed. Specify this option if you want to print only the columns corresponding to variables that appear in at least one adjustment set or conditioning set.

CYCLES < =*n* | =ALL >

displays directed cycles in each model that you specify by using a MODEL statement. By default, the procedure issues an error if any model that you specify in a MODEL statement contains a directed cycle. If you specify this option, the procedure does not issue an error. This is useful if you need to investigate and revise a large causal model that contains directed cycles.

By default, the procedure displays only one directed cycle (if such a cycle exists) for each model. You can change the number of cycles that are printed for each model by specifying the optional argument. For a positive integer *n*, CYCLES=*n* prints at most *n* directed cycles. To print all cycles, specify CYCLES=ALL.

This option has no effect for a model that contains no directed cycles.

DISCONNECTED

DISCONNECT

suppresses warnings for models that contain disjoint directed acyclic graphs (DAGs). By default, the procedure issues a warning if a model that you specify in a MODEL statement is not weakly connected (that is, if the model consists of two or more disjoint DAGs). Although the procedure can still continue to analyze disconnected models, this warning provides a safeguard against accidental misspecification. You can use this option to turn off the warning. This option does not have any effect for a model that is connected.

IMAP < =GLOBAL | LOCAL >

requests a list of the conditional independence properties (an independence map) that are encoded by each causal model. For more information about conditional independence properties in causal models, see the section "Statistical Properties of Causal Models" on page 2381.

GLOBAL produces a list of global Markov properties. For a causal model that is encoded in a DAG, the global Markov property corresponds to d-separation (Koller and Friedman 2009).

LOCAL produces a list of local Markov properties. For a causal model that is encoded in a DAG, the local Markov property is the set of nodes that are jointly independent of a given node, conditional on the parents of that node (Koller and Friedman 2009).

By default, IMAP=LOCAL.

When a conditional independence property includes an unmeasured variable, the implications of that property (for example, zero partial correlation) cannot be tested in a data set. Such properties are still included in the independence map as an implication of a causal model.

INMODEL=*SAS-data-set*

specifies a data set that contains one or more causal models to be analyzed. A typical use of the INMODEL= option is when you want to reuse one or more causal models that you saved by specifying the OUTMODEL= option in a previous invocation of PROC CAUSALGRAPH. Instead of using a MODEL statement to specify the model, you use the INMODEL= option to input the model specification that was saved from the previous run.

Sometimes, you might create an INMODEL= data set by modifying an existing OUTMODEL= data set. However, editing and modifying OUTMODEL= data sets require a good understanding of the formats and contents of the OUTMODEL= data sets. This process could be error-prone for novice users. For more information about the format of INMODEL= or OUTMODEL= data sets, see the section "INMODEL= and OUTMODEL= Data Sets" on page 2386.

You can combine the INMODEL= option with one or more MODEL statements. When you specify the INMODEL= option, every model that is defined in the specified *SAS-data-set* is loaded into PROC CAUSALGRAPH before any MODEL statements are processed. This means that you can use the REFMODEL= option in the MODEL statement to refer to models in the *SAS-data-set*.

LIST

requests the possible identification criteria for each model. The type of the identification criteria that are constructed depends on the value that you specify in the METHOD= option. If METHOD=ADJUSTMENT or BACKDOOR, a list of adjustment sets is produced for each model. If METHOD=IV, a list of instrumental variables is produced for each model.

The LIST option is used by default if you specify an IDENTIFY statement but no TESTID statement. Otherwise, the NOLIST option is used.

You cannot specify both the LIST and NOLIST options.

MAXLIST=*n* | ALL

specifies the maximum number of identification criteria to be printed when you specify the LIST option. By default, the procedure prints up to 100 entries. To change the maximum number of entries to be printed, you can specify a positive integer *n*. To print all entries, specify MAXLIST=ALL.

MAXSIZE=*n* | MIN

specifies the maximum size of the adjustment sets to be constructed when you specify the LIST option. By default, the procedure prints all valid adjustment sets without regard to the size of the set. You can specify the MAXSIZE= option with a positive integer *n* to change the maximum size of the sets to be printed. To see only those adjustment sets with the smallest size, specify MAXSIZE=MIN.

The MAXSIZE=MIN option is not the same as the MINIMAL option. Every adjustment set that has the smallest size is minimal, but not every minimal adjustment set necessarily has the smallest size.

The MAXSIZE= option is ignored if you specify METHOD=IV.

METHOD=ADJUSTMENT | ANCESTOR | BACKDOOR | POUTCOME | PTREATMENT | IV

specifies the method to use for identifying a causal effect. For more information about identifying causal effects, see the section "Causal Effect Identification" on page 2382. You can specify the following values:

ADJUSTMENT specifies the constructive backdoor criterion (Van der Zander, Liśkiewicz, and Textor 2014). The constructive backdoor criterion is equivalent to the adjustment criterion (Shpitser, VanderWeele, and Robins 2010). You can use this criterion to find all valid adjustment sets that consist of observed variables only. This method contains the backdoor (METHOD=BACKDOOR), parents-of-treatment (METHOD=PTREATMENT), parents-of-outcome (METHOD=POUTCOME), and joint ancestor (METHOD=ANCESTOR) criteria as special cases.

ANCESTOR specifies the joint ancestor criterion. When you have a single treatment variable and a single outcome variable, the set of all common ancestors of those two variables is a valid adjustment set (Elwert 2013). You can use the joint ancestor criterion in PROC CAUSALGRAPH with any number of treatment and outcome variables to find all valid adjustment sets that consist exclusively of nodes that are ancestors of at least one treatment and at least one outcome variable.

This method is a special case of the constructive backdoor criterion. This means that every adjustment set that satisfies the joint ancestor criterion also satisfies the constructive backdoor criterion.

BACKDOOR specifies the backdoor adjustment criterion (Pearl 2009b). The backdoor adjustment criterion is similar to the constructive backdoor criterion (METHOD=ADJUSTMENT) in that it finds valid adjustment sets that consist of observed variables only. The backdoor criterion is stronger than the constructive backdoor criterion, so it is computationally more efficient than the constructive backdoor criterion, although it might not find every valid adjustment set. The appeal of the backdoor criterion is that it has an intuitive interpretation and is more widely known (Pearl 2009b).

This method is a special case of the constructive backdoor criterion. This means that every adjustment set that satisfies the backdoor adjustment criterion also satisfies the constructive backdoor criterion.

POUTCOME specifies the parents-of-outcome criterion. This criterion is similar to the parents-of-treatment (METHOD=PTREATMENT) criterion except that it uses the parents of the outcome variables. When you have a single treatment variable and a single outcome variable, there are certain conditions under which you can form a valid adjustment set by including all nodes that are parents of the outcome but that do not lie on a causal path (Elwert 2013). You can use the parents-of-outcome criterion in PROC CAUSALGRAPH with any number of treatment and outcome variables to find all valid adjustment sets that consist exclusively of parents of outcome variables.

This method is a special case of the constructive backdoor criterion. This means that every adjustment set that satisfies the parents-of-outcome criterion also satisfies the constructive backdoor criterion.

PTREATMENT specifies the parents-of-treatment criterion (Elwert 2013). The set that contains all parents of all treatment variables is a valid adjustment set. (In the case of multiple treatment variables, a parent of treatment that is itself a treatment variable is not considered to be part of the adjustment set.) The parents-of-treatment criterion in PROC CAUSALGRAPH uses this sufficient condition as a starting point and finds all valid adjustment sets that consist exclusively of parents of treatment variables.

This method is a special case of the constructive backdoor criterion. This means that every adjustment set that satisfies the parents-of-treatment criterion also satisfies the constructive backdoor criterion.

IV specifies the ancestral instrument criterion (Van der Zander, Textor, and Liśkiewicz 2015). This method finds all traditional and ancestral instruments for a causal model. If you use this criterion with the LIST option, the procedure first checks to see whether each observed variable can be used as a classical instrument (that is, the variable is a valid instrument without conditioning on any other variables). If the observed variable is not a classical instrument, then the procedure searches for a conditioning set that instrumentalizes the observed variable.

In order to use the ancestral instrument criterion, every model that you specify by using a MODEL statement must contain a single treatment variable and a single outcome variable that are directly connected by an edge.

By default, METHOD=ADJUSTMENT.

MINIMAL
MIN

requests a list of only minimal adjustment sets when you specify the LIST option. A minimal adjustment set is a set for which no proper subset is also a valid adjustment set. If you specify prescribed adjustment variables in the IDENTIFY statement, then the minimal condition is checked only among those sets that contain the prescribed variables.

The MINIMAL option is not the same as the MAXSIZE=MIN option. Every adjustment set that has the smallest size is minimal, but not every minimal adjustment set necessarily has the smallest size.

The MINIMAL option is ignored if you specify METHOD=IV.

NOLIST

excludes the construction of possible identification criteria. If you specify this option, the procedure does not construct any identification criteria for any model. This option is used by default if you specify a TESTID statement or if you do not specify an IDENTIFY statement.

You cannot specify both the LIST and NOLIST options.

NOPRINT

suppresses all displayed output.

NOSORT

requests an unsorted list of adjustment sets when METHOD=ADJUSTMENT or BACKDOOR. By default, the procedure prints adjustment sets in order of increasing set size. The NOSORT option enables more efficient output because the sets do not need to be sorted. This can be especially useful for large models when you have a larger number of adjustment sets. For more information about options that improve the performance of PROC CAUSALGRAPH, see the section "Time Requirements" on page 2388.

The NOSORT option is ignored if you specify METHOD=IV.

NTHREADS=*n*

specifies the maximum number of simultaneous computational threads available to the procedure. By default, the procedure uses the values of the THREADS and CPUCOUNT system variables to determine the number of computational threads. Multithreading is available when you specify more than one MODEL statement. There is no performance benefit in using a value of *n* greater than the number of models. To disable multithreading within the procedure, specify NTHREADS=1.

ORDERMODELS

prints the model results in alphabetical order by model label. However, the default behavior is to print the model results in the order in which the models appear in the input specifications.

OUTMODEL=*SAS-data-set*

creates an output data set that contains the causal model specifications. An OUTMODEL= data set can be used as an input INMODEL= data set in a subsequent analysis by PROC CAUSALGRAPH.

PSUMMARY

displays only the summary of the causal effect identification analysis. Graphics are also produced if you explicitly specified them in a PATHDIAGRAM statement.

IDENTIFY Statement

IDENTIFY *effect-specification* **;**

The IDENTIFY statement specifies the causal effects to be analyzed. The *effect-specification* has the following form:

treatment-variables right-arrow outcome-variables < | *adjustment-variables* >

Each of the *treatment-variables*, *outcome-variables*, and *adjustment-variables* contains a list of variables in any of the following forms:

- *variables*
- {*variables*}
- [*variables*]
- (*variables*)

The use of braces, brackets, or parentheses for grouping variables is optional but highly recommended because it clearly distinguishes the roles of different types of variables within the *effect-specification*.

The *right-arrow* that indicates the causal direction can be of different forms. For information about specifying arrows in the CAUSALGRAPH procedure, see the section "Arrow or Edge Specification" on page 2386.

The following is an example for analyzing the effect of a single treatment variable on a single outcome variable without any prescribed adjustment variables:

```
identify x ==> y;
```

The following examples are equivalent for specifying a joint causal effect analysis that includes a prescribed set of adjustment variables:

```
identify {x1-x3} ==> {y1 y5} | {c1-c4 z};
identify {x1 x2 x3} ==> y1 y5 | {c1-c4 z};
identify x1 x2 x3 ==> y1 y5 | c1 c2 c3 c4 z;
```

You must use an IDENTIFY statement if you use a TESTID statement or if you use the LIST option in the PROC CAUSALGRAPH statement. You cannot specify more than one IDENTIFY statement.

Specifying Causal Effects

A causal effect consists of one or more treatment variables and one or more outcome variables. The treatment variables and the outcome variables are each specified as a list of one or more valid SAS variable names. The two variable lists are separated by a single right arrow. Every treatment variable and every outcome variable must be measured or observed. This means that you cannot include any treatment or outcome variable in the UNMEASURED statement.

If you specify more than one treatment variable or more than one outcome variable, PROC CAUSALGRAPH attempts to identify the joint effect. For more information about the interpretation of joint treatment effects, see the section "Identifying Joint Treatment Effects" on page 2387. However, you cannot specify multiple treatment or multiple outcome variables if you specify the METHOD=IV option in the PROC CAUSALGRAPH statement.

You cannot use the same variable as a treatment variable and an outcome variable. Every treatment variable and every outcome variable must appear in at least one MODEL statement, or else the effect is ill-defined. In that case, the missing variable is ignored. In addition, if you use an IDENTIFY statement, then every model must contain at least one treatment variable and at least one outcome variable.

Prescribing Adjustment Variables

If you want to prescribe a set of adjustment variables in the *effect-specification*, use a vertical bar (|) after the list of outcome variables and then specify the list of adjustment variables. This set of prescribed adjustment variables is included in every adjustment set that is tested or constructed by the CAUSALGRAPH procedure. The set of prescribed adjustment variables is ignored if you specify the METHOD=IV in the PROC CAUSALGRAPH statement.

An adjustment variable cannot also be a treatment variable or an outcome variable. Every adjustment variable must appear in at least one MODEL statement, or else it is not defined and is ignored. Every adjustment variable must be measured or observed. This means that you cannot include any prescribed adjustment variable in the UNMEASURED statement.

MODEL Statement

MODEL '*label*' *path* <, *path* . . . > < / *options* > **;**

where *label* represents a name that you assign to the model and *path* represents either of the following specifications:

- a *directed-path*
- a *covariance-path*

Details about the syntax and interpretation of these different types of *paths* are described later in this section. Here are some examples:

```
model 'Example1'
   X1 ==> X2,
   X3 <== X2,       /* same as: X2 ==> X3 */
   X3 ==> X4 X5,    /* same as: X3 ==> X4, X3 ==> X5 */
   <==> {X2 X5 X6}; /* latent confounding between X2, X5, and X6 */

model 'Example2'
   X1 ==> X2 ==> X3 <== X4 <==> X5;
```

You must specify at least one MODEL statement in an analysis.

The *label* is enclosed within quotation marks and can be any string of characters. Every model that you specify using a MODEL statement must have a unique *label*. The *labels* for models are not case-sensitive.

A MODEL statement specifies a causal model in the form of a directed acyclic graph (DAG). A DAG consists of nodes that represent variables in the model and edges that represent causal relationships between pairs of variables. For more information about how to use a DAG to represent a causal model, see the section "Causal Graph Theory" on page 2378. You specify the causal relationships in a DAG in accordance with the *path* syntax in the MODEL statement.

The following subsections explain the *path* syntax. Each *path* is either a *directed-path* or a *covariance-path*.

Directed-paths

A *directed-path* has the following form:

variables arrow variables < arrow variables . . . >

The *directed-path* continues alternating between *arrows* and *variables* and terminates with either a comma (which ends the path) or a semicolon (which ends the MODEL statement). Each *variables* argument contains a list of variable names. Optionally, you can enclose this list of names in curly braces, square brackets, or parentheses. This means that all the following forms are equivalent:

- *variables*
- {*variables*}
- [*variables*]
- (*variables*)

The use of braces, brackets, or parentheses for grouping variables is optional but highly recommended because it clearly identifies the edges that are associated with each variable.

Each *arrow* in a *directed-path* defines a set of edges in a DAG. Each variable in the list preceding the *arrow* is linked by an edge to each variable in the list following the *arrow*. The direction of the edge is given by the direction of the *arrow*. An *arrow* in a *directed-path* can be a right arrow (**==>**), a left arrow (**<==**), or a bidirected arrow (**<==>**). For more information about representing arrows in the CAUSALGRAPH procedure, see the section "Arrow or Edge Specification" on page 2386.

The procedure does not allow multiple edges of the same type between two variables. If the same edge is specified more than once in a MODEL statement, the repeated specifications are ignored. Variable names are not case-sensitive in the procedure.

Here are some examples of specifying *directed-paths*:

```
model 'M1'
   Y ==> Z,
   U <== W,
   X U V ==> Y,
   W ==> Z <== M N ==> Y;
```

The following MODEL statement specifications of "M2" are equivalent:

```
model 'M2' V1 V2 ==> X1-X3 A <== B C <==> D;
model 'M2' {V1 V2} ==> {X1-X3 A} <== {B C} <==> D;
model 'M2' V1 V2 ==> X1-X3 A,
           D <==> {B C} ==> X1-X3 A;
model 'M2' V1 ==> X1, V1 ==> X2, V1 ==> X3, V1 ==> A,
   V2 ==> X1, V2 ==> X2, V2 ==> X3, V2 ==> A,
   X1 <== B, X2 <== B, X3 <== B, A <== B,
   X1 <== C, X2 <== C, X3 <== C, A <== C,
   B <==> D, C <==> D;
```

Covariance-paths

A *covariance-path* has the following form:

<==> *variables*

The **<==>** syntax represents a bidirected arrow. For more information about representing arrows in the CAUSALGRAPH procedure, see the section "Arrow or Edge Specification" on page 2386. The *variables* argument contains a list of variable names. Optionally, you can enclose this list of names in curly braces, square brackets, or parentheses. This means that all the following forms are equivalent:

- *variables*
- {*variables*}
- [*variables*]
- (*variables*)

The use of braces, brackets, or parentheses for grouping variables is optional but highly recommended because it clearly identifies the edges that are associated with each variable.

You use a *covariance-path* to specify covariances between pairs of variables that are not explained by the causal paths in the model. Essentially, these covariances are equivalent to assuming latent confounding between each pair of variables that you specify in the *covariance-path*. That is, one bidirected edge is added to the model for each pair of unique variables in the *covariance-path*. Thus the following specifications are equivalent:

```
model 'MyModel' <==> {X1 X2};
model 'MyModel' X1 <==> X2;
```

Furthermore, because a bidirected edge represents latent confounding, this is also equivalent to the following specification:

```
model 'MyModel' X1 <== L ==> X2;
unmeasured L;
```

For more information about the interpretation of bidirected edges in the CAUSALGRAPH procedure, see the section "Causal Graph Theory" on page 2378.

Here are some examples of specifying *covariance-paths*:

```
model 'M3'
   <==> {X1 X2},
   <==> {X3-X5 X8};
```

The following MODEL statement specifications of "M4" are equivalent:

```
model 'M4' <==> {X1-X3 Y Z};
model 'M4' <==> {X1 X2 X3 Y Z};
model 'M4'
   <==> {X1 X2},
   <==> {X1 X3},
   <==> {X1 Y},
   <==> {X1 Z},
   <==> {X2 X3},
```

```
      <==> {X2 Y},
      <==> {X2 Z},
      <==> {X3 Y},
      <==> {X3 Z},
      <==> {Y Z},
   model 'M4'
      X1 <==> X2 X3 Y Z,
      X2 <==> X3 Y Z,
      X3 <==> Y Z,
      Y <==> Z;
```

Options

You can specify the following *options* in the MODEL statement:

NOANALYSIS

excludes the model from the identification analysis. A typical use case for this option is when you have two or more models that you want to analyze and those models have very similar structures. You can specify the common structure in a base model and use the NOANALYSIS option to prevent the base model from being analyzed. You can then use two additional MODEL statements with the REFMODEL= option to specify the causal models that you want to analyze.

If you use the OUTMODEL= option in the PROC CAUSALGRAPH statement to save your models, the NOANALYSIS option excludes the current model from that data set.

REFMODEL='*label*'

specifies a reference model to use as a starting point for the current model. The *label* is a quoted string that specifies the label of the reference model. The reference model must already exist before you can refer to it in a REFMODEL= option. This means that the reference model must be defined in a prior MODEL statement or must be defined in the data set that you specify in the INMODEL= option. You can specify at most one reference model for each MODEL statement.

When you use the REFMODEL= option, all nodes and edges in the reference model are included in the current model. You can use the path specification in the current MODEL statement to add more nodes and edges to the model, or you can use the REMOVE= or REMOVENODES= option to remove elements from the reference model. For example, the models “M1” through “M4” are identical in the following analysis:

```
proc causalgraph;
   model 'Base' X -> M -> Y / noanalysis;
   model 'Full' X -> M -> Y, {U C} -> {X Y} / noanalysis;
   model 'M1' X -> M -> Y, C -> X Y;
   model 'M2' C -> X Y / refmodel='Base';
   model 'M3' / refmodel='Full' remove=(U -> X Y);
   model 'M4' / refmodel='Full' removenode=U;
   identify X -> Y;
run;
```

It is important to understand how PROC CAUSALGRAPH interprets the model specification when you specify the REFMODEL=, REMOVE=, and REMOVENODES= options together. The model specification options in the MODEL statement are processed in the following order:

- All nodes and edges in the reference model (specified in the REFMODEL= option) are added to the current model.
- Nodes that you specify in the REMOVENODES= option are removed, along with all adjacent edges.
- All edges that you specify in the REMOVE= option are removed, along with any nodes that are left unconnected.
- Nodes and edges in the MODEL path specification are added.

REMOVE=(*path <, path . . . >***)**

specifies paths to remove after the reference model has been processed. You must specify the REFMODEL= option in order to use the REMOVE= option.

A *path* represents either of the following specifications:

- a directed path
- a covariance path

For more information about these path specifications, see the sections on directed paths and covariance paths in the main MODEL statement documentation. Note that the path specification for the REMOVE= option must be enclosed in parentheses. The REMOVE= option does not change the reference model.

It is important to understand how PROC CAUSALGRAPH interprets the model specification when you specify the REFMODEL=, REMOVE=, and REMOVENODES= options together. For more information, see the REFMODEL= option.

REMOVENODES={*variables***}**

REMOVENODES=[*variables***]**

REMOVENODES=(*variables***)**

REMOVENODE=*variable*

specifies one or more nodes to remove after PROC CAUSALGRAPH processes the reference model. You must specify the REFMODEL= option in order to use the REMOVENODES= option.

The *variables* or *variable* is a list of nodes to remove from the current model. When these nodes are removed, all edges that are connected to these nodes (regardless of direction) are also removed from the model.

It is important to understand how PROC CAUSALGRAPH interprets the model specification when you specify the REFMODEL=, REMOVE=, and REMOVENODES= options together. For more information, see the REFMODEL= option.

PATHDIAGRAM Statement

PATHDIAGRAM < *options* > **;**

The PATHDIAGRAM statement requests a path diagram, which is a visual representation of a graphical causal model. You can use this statement to specify and modify the layout algorithm and to fine-tune many graphical and nongraphical features of path diagrams. You can use multiple PATHDIAGRAM statements to produce path diagrams that have different styles and graphical features.

You can specify a PATHDIAGRAM statement without any *options* to produce a default path diagram for every model in an analysis by PROC CAUSALGRAPH. Table 37.2 shows the *options* that you can use to alter this default behavior.

Table 37.2 Options Available in the PATHDIAGRAM Statement

Option	Description
ARRANGE=	Specifies the algorithm for laying out variables in the path diagram
DESIGNHEIGHT=	Sets the height, in pixels, of the output path diagram
DESIGNWIDTH=	Sets the width, in pixels, of the output path diagram
DIAGRAMLABEL=	Specifies the label of the path diagram in the ODS
LABEL=	Specifies labels of the nodes that are shown in the path diagram
MODEL=	Requests path diagrams for a list of models
NOTITLE	Suppresses the display of the title
SCALE=	Specifies the scale factor for the node size
TEXTSIZEMIN=	Specifies the minimum text size in the output path diagram
TITLE=	Specifies the title to display in the output path diagram

ARRANGE=*name*

ARRANGEMENT=*name*

METHOD=*name*

specifies the algorithm for laying out the variables in the path diagram. You can specify the following *names*:

FLOW specifies the process-flow algorithm.

GRIP specifies the graph drawing with intelligent placement (GRIP) algorithm.

GROUPEDFLOW specifies the grouped-flow algorithm.

By default, ARRANGE=GRIP. For more information and for general uses of these layout algorithms, see the section "The Process-Flow, Grouped-Flow, and GRIP Layout Algorithms" on page 1784 in Chapter 33, "The CALIS Procedure."

DESIGNHEIGHT=*h*

DH=*h*

sets the height of the path diagram, in number of pixels, where *h* is between 100 and 10,922. The default height is 600 pixels. Typically, you might want to set a larger design height and width when your path diagram contains more nodes or variables.

DESIGNWIDTH=*w*

DW=*w*

sets the width of the path diagram, in number of pixels, where *w* is between 100 and 10,922. The default width is 600 pixels. Typically, you might want to set a larger design width and height when your path diagram contains more nodes or variables.

DIAGRAMLABEL=*name*

DLABEL=*name*

specifies the label of the path diagram. You can use any valid SAS name or quoted string of up to 256 characters for *name*. However, ODS uses only up to 40 characters of the label. The following statements show two example label specifications:

```
pathdiagram diagramlabel=MyCausalModel;
pathdiagram diagramlabel="Model of the Data Generating Process";
```

If you do not specify this option, PROC CAUSALGRAPH uses the *name* that the TITLE= option provides. If you specify neither the DIAGRAMLABEL= option nor the TITLE= option, PROC CAUSALGRAPH uses "Path Diagram: *model-label*" for the path diagram label, where *model-label* is the label for the model that is depicted in the diagram.

LABEL= [*varlabel* < **,** *varlabel . . .* > **] | {***varlabel* < **,** *varlabel . . .* > **}**

specifies the labels of variables to be displayed in path diagrams, where each *varlabel* has the following form:

*variable***=***label*

You can use any valid SAS names or quoted strings of up to 256 characters for *labels*. The *labels* provide alternative text for the corresponding variables in the causal model. For example, instead of using original variable names such as x1 in the path diagram, the following statement specifies the use of more meaningful labels:

```
pathdiagram label=[x1="Math Ability"];
```

If you do not specify a label for a variable, the name of the variable is used in the output path diagram.

MODEL={*label* < **,** *label . . .* > **}**

MODEL=[*label* < **,** *label . . .* > **]**

MODEL=(*label* < **,** *label . . .* > **)**

MODEL=*label*

requests path diagrams for a list of models, which are specified by their associated labels. By default, each PATHDIAGRAM statement produces a path diagram for all models in the analysis. This option is useful if you want to restrict the path diagram output to a particular set of models.

NOTITLE

suppresses the display of the default title. You can use the TITLE= option to provide your own title.

SCALE=*n*

DIAGRAMSCALE=*n*

specifies the scaling factor, *n*, for the node size relative to the dimensions of the path diagram. Valid values of *n* are between 0 and 6, inclusive. This option applies to the ARRANGE=GRIP layout only.

PROC CAUSALGRAPH uses certain default pixel dimensions for the nodes in path diagrams that have default design dimensions (see the DESIGNHEIGHT= and DESIGNWIDTH= options for the default design dimensions). The ratio of this node dimension to the design dimension defines the point at which SCALE=1. Values of *n* greater than 1 enlarge the nodes (relative to the design dimensions), and values of *n* less than 1 shrink the nodes (relative to the design dimensions). Hence, you can accommodate more nodes (variables) in your path diagram by specifying smaller SCALE= option values.

If you use the GRIP layout algorithm, PROC CAUSALGRAPH automatically adjusts the SCALE= option value according to the number of nodes in the path diagram, as shown in the following table:

Number of Nodes	SCALE=
14 or less	1.00
15–19	0.95
20–24	0.90
25–29	0.85
30–34	0.80
35–39	0.75
40–44	0.70
45–49	0.65
50–59	0.60
60–69	0.55
70 or more	0.50

These default values are also applied if you specify DIAGRAMSCALE=0. Although these values yield reasonable relative node sizes in most situations, you can always adjust the relative node size by setting the SCALE= option value manually. For example, if your path diagram contains 33 nodes and some nodes appear to be overlapping, then you can consider setting a SCALE= option value of less than 0.8. When you are satisfied with the SCALE= option value, you can improve the resolution of the path diagram by using the DESIGNHEIGHT= and DESIGNWIDTH= options.

TEXTSIZEMIN=*i*

specifies the minimum text size, *i*, in the output path diagram. Valid values of *i* are between 4 and 40. The default is 10, for the 10-point font size.

TITLE=*name*

specifies the title of the path diagram. You can use any valid SAS name or a quoted string of up to 256 characters for *name*. If you do not specify this option, PROC CAUSALGRAPH uses "Path Diagram: *model-label*" for the diagram title, where *model-label* is the label for the model that is depicted in the diagram.

TESTID Statement

TESTID <'*label*'> *variable-list* <*/ options*> **;**

The TESTID statement tests whether a criterion that you specify is valid for identifying a causal effect. The form of the *variable-list* depends on which method you specify in the METHOD= option in the PROC CAUSALGRAPH statement and is described later in this section. The causal effect to be identified is specified by the IDENTIFY statement. You must specify the IDENTIFY statement in order to use a TESTID statement.

For example, the following code tests two specified adjustment sets to determine whether each set is sufficient for the unbiased estimation of the causal effect of X on Y:

```
proc causalgraph method=adjustment;
   model 'TESTID Demo'
      Z ==> X ==> Y,
      U => X Y,
      W => Z U;
   identify X => Y;
   testid U;
   testid U W;
run;
```

The following example tests whether you can use the variable Z as an instrument for the causal effect of X on Y when U is unmeasured:

```
proc causalgraph method=iv;
   model 'TESTID Demo'
      Z ==> X ==> Y,
      U => X Y,
      W => Z U;
   identify X => Y;
   testid Z;
   unmeasured U;
run;
```

For more information about when an estimation criterion is valid, see the section "Causal Effect Identification" on page 2382.

When METHOD=ADJUSTMENT, ANCESTOR, BACKDOOR, POUTCOME, or PTREATMENT, or when you omit the METHOD= option, *variable-list* is a list of adjustment variables. This list of variables is tested to see whether it forms a valid adjustment set according to the criterion that you specify in the METHOD= option. For these adjustment methods, the list of variables can be empty. If you specify a set of prescribed adjustment variables in the IDENTIFY statement, then these prescribed adjustment variables are added to the list of variables that you specify in each TESTID statement. If *variable-list* is empty and there are no prescribed adjustment variables in the IDENTIFY statement, then the TESTID statement tests whether the causal effect is identified without any adjustment.

When METHOD=IV, *variable-list* consists of a single variable name. This variable name is required and cannot be empty. The single variable is tested to see whether it is an instrumental variable that can be used to estimate the causal effect that you specify in the IDENTIFY statement. If you specify a set of prescribed adjustment variables in the IDENTIFY statement, these variables are ignored.

Regardless of the value of the METHOD= option, every variable in a TESTID statement must be observed. This means that any variable name in *variables* cannot also be specified in the UNMEASURED statement.

You can specify multiple TESTID statements in each run of the CAUSALGRAPH procedure. Each proposed criterion that you specify in a TESTID statement is tested using each model that you specify in a MODEL statement unless you specify the NOANALYSIS option in the MODEL statement.

The *label* is optional and can be any string of characters enclosed in quotation marks. If you specify a *label*, it must be the first item in the TESTID statement. If you do not specify a *label*, then a test name is automatically generated. The autogenerated name has the form "Test*i*", where *i* is the smallest positive integer such that each test name is unique. It is recommended that you use your own labels so that you can identify the tests easily in the output results.

You can specify the following *options* in the TESTID statement:

CONDITIONAL={*variables***}**

CONDITIONAL=[*variables***]**

CONDITIONAL=(*variables***)**

CONDITIONAL=*variable*

performs a test for a conditional instrumental variable, where the *variables* or *variable* is a list of conditioning variables. You can specify this option only if you specify METHOD=IV in the PROC CAUSALGRAPH statement. If you use this option, the procedure determines whether the instrumental variable that you specify in the TESTID statement is a conditional instrument associated with the specified conditioning set of *variables*.

PATHS <=(*path-types)***>**

PATHS <=*path-type***>**

creates an enumeration of the proper paths from the treatment variables to the outcome variables in each model. You cannot use this option if you specify METHOD=IV in the PROC CAUSALGRAPH statement.

A proper path is a path that begins with a treatment variable and does not contain any other treatment variables. The procedure determines whether each path is causal or noncausal. It also applies the adjustment set that is indicated by the TESTID statement and then determines whether each path is blocked or unblocked.

You can use the *path-types* argument to change the paths that are displayed. You can specify the following *path-type* values:

ALL	displays all paths.
BLOCKED	displays blocked paths.
CAUSAL	displays causal paths.
NONBLOCKED	displays nonblocked paths.
NONCAUSAL	displays noncausal paths.

If you do not specify *path-types*, then PATHS=(ALL) is used by default. You can specify multiple values for *path-types* at the same time. For example, you can specify PATHS=(CAUSAL BLOCKED) to display all paths that are both causal and blocked.

UNMEASURED Statement

UNMEASURED *variables* **;**

The UNMEASURED statement specifies a list of variables that are treated as unmeasured. You might want to specify a variable as unmeasured in the following situations:

- The variable corresponds to a latent construct that cannot be measured.
- The variable was not measured or will not be measured in the data set for the causal effect estimation.
- The data that are collected for the variable are not considered reliable enough to include in the causal effect estimation.

You can specify only one UNMEASURED statement.

If you specify multiple models by using multiple MODEL statements, each variable that you specify in the UNMEASURED statement is treated as unmeasured in every model. In addition, any variable that you specify in the IDENTIFY statement or in a TESTID statement cannot also be specified in the UNMEASURED statement. That is, the treatment and outcome variables, the adjustment variables, and the instrumental variables are all assumed to be measured or observed.

Details: CAUSALGRAPH Procedure

In a simple randomized controlled study, experimental units are randomly assigned to either a treatment group or a control (that is, untreated or unexposed) group. As a result of this randomization step, the two groups have the same distribution of covariates. Thus you can estimate the causal effect of the treatment by a direct comparison of the outcome variables in the two groups.

In a nonrandomized study, such as an observational study, the units of interest are not randomized to the treatment and control groups. As a result, some covariates can be correlated with both the group assignment and the outcome variables. In this case, the values of the outcome variables are determined both by the causal effect of treatment and by spurious association with covariates. Because the covariates confound the causal effect, you cannot determine the causal effect of the treatment without some form of adjustment (for example, matching or stratification) to remove the spurious association.

In fact, spurious association that is induced by confounding covariates can also be present in imperfect randomized controlled studies. For example, because of idiosyncratic background factors, subjects might not fully comply with the treatment protocol.

You can use the CAUSALGRAPH procedure to determine when, and under what circumstances, it is possible to estimate a causal effect that has a valid causal interpretation. To do so, you define causal models in the form of directed acyclic graphs (DAGs). The procedure accepts the input DAG and outputs implications of the models that are useful for causal analysis and effect estimation. The sections that follow describe the link between causal models and DAGs and discuss important properties of DAGs that you can use to identify causal effects.

Statistical and Causal Concepts

A statistical model is a mathematical rule for computing quantities of interest from the joint distribution function for a set of observed variables. For example, you can use a statistical model to answer queries such as the probability of a specific event and how that probability would change as different variables are observed (Pearl 1993). However, such models and data are purely associative. Because associations typically contain a mix of causal and noncausal components, statistical approaches alone are not sufficient to answer causal queries (Elwert 2013). This is because a causal query seeks to predict the effect of an action that intervenes in and changes the data generating process (Pearl 2009b). Such queries are essential for estimating the effects of treatments or assessing the impact of public policies (Pearl 1993). In order to answer a causal query by using data from a nonrandomized experiment, you must supplement the joint distribution function with a set of causal assumptions that, together, form a causal model.

Causal Graph Theory

A causal model represents beliefs about the data generating process that is being studied. That is, it defines the causal relationships that determine how the value of each variable is determined. These beliefs reflect an existing state of knowledge, including expert opinion and past experience. Constructing a causal model requires not only expertise in the subject matter being studied but also knowledge of the measurement process so that the factors affecting each variable can be accurately reflected in the model.

DAGs provide a formal semantics for defining and manipulating a causal model. The theoretical developments that link DAGs with causal analysis are most closely associated with the work of Pearl and his colleagues. See Pearl (2009b) for a detailed overview or Pearl (2010) for an abbreviated review. Lauritzen (1996) provides a technical treatment of the probabilistic properties of graphical models (including DAGs). Koller and Friedman (2009) and Spirtes, Glymour, and Scheines (2001) provide extensive treatments of the computational tools available for analyzing data by using graphical models. For an accessible summary of using DAGs to identify causal effects, see Elwert (2013) and Elwert and Winship (2014). After you decide on a valid identification strategy, Schafer and Kang (2008) provide a useful and accessible summary of computational tools for estimating the average causal effect.

Components of a Causal Graph

A DAG consists of three components (Elwert 2013):

- nodes
- edges
- missing edges

Each node in the DAG represents a variable that is assumed to play a causal role in the process being studied. Each variable can have any distribution. It is not necessary for every node in the DAG to correspond to a variable that has been (or could be) measured. For example, some variables in the DAG might correspond to latent constructs that are assumed to play a causal role in the process being modeled but that cannot possibly be observed directly. By convention, error random variables (independent error terms) are not represented in a DAG (Elwert 2013) unless the error random variable is a common cause of two or more variables that the model already includes (Spirtes, Glymour, and Scheines 2001).

All edges in a DAG are directed. That is, an edge consists of an arrow that points from one node into another node. An edge is a graphical representation of a causal assumption. Specifically, an edge in a DAG represents an assumed *possible* direct causal effect of one variable on another (Elwert 2013). These causal effects are assumed to be deterministic (Pearl 1993), but they are fully nonparametric in the sense that each edge can have any functional form (Elwert and Winship 2014). There can be at most one edge between a pair of nodes in a DAG.

Because each edge in a DAG is given a causal interpretation, each edge is associated with a temporal ordering of a pair of nodes. For this reason, the DAG cannot contain a directed cycle. The CAUSALGRAPH procedure performs a semantic validation of every model to verify that it does not contain a directed cycle.

PROC CAUSALGRAPH allows bidirected edges to be specified. A bidirected edge is interpreted as unmeasured confounding between two variables, and thus the graph is still a DAG. For example, the edge

$$X \leftrightarrow Y$$

is interpreted as the pair of edges

$$X \leftarrow L \rightarrow Y$$

where the node L represents some unmeasured variable.

Missing edges in a DAG (that is, where two nodes are *not* directly connected by an edge) indicate an assumption of exactly zero direct causal effect. Thus a missing edge in a DAG represents a much stronger assumption than an edge. This is the *strong null hypothesis* for graphical models; it is known in the econometrics literature as an *exclusion restriction* (Elwert 2013). Missing edges have implications for the statistical properties that are implied by a causal model. For more information about these properties, see the section "Statistical Properties of Causal Models" on page 2381.

Together the nodes, edges, and missing edges in a DAG form a causal model that encodes researchers' assumptions about a data generating process. Starting with these data generating assumptions, you can use a set of graphical rules that operate on the DAG to derive statements of statistical association. For more information about sources of statistical association in DAGs, see the section "Sources of Association and Bias" on page 2380.

Terminology

Two variables in a DAG are *adjacent* if they are directly connected by a single edge.

A *path* is an ordered list of variables in which no variable appears more than once and consecutive variables in the list are adjacent in the graph. The edges that connect consecutive nodes in a path can point in either direction.

A path is *causal* if, for every consecutive pair of variables on the path, the arrow that connects the two variables points toward the latter variable. A path that is not causal is called *noncausal*. A path is *proper* if it begins with a treatment variable and does not contain any other treatment variables (Shpitser, VanderWeele, and Robins 2010). A path can also be *blocked* or *nonblocked*. For information about blocked and nonblocked paths, see the section "Statistical Properties of Causal Models" on page 2381.

A DAG encodes specific relationships between the variables in a causal model. It is standard practice to describe these relationships by using familial adjectives; for example, see Koller and Friedman (2009); Pearl (2009b); Elwert and Winship (2014); Van der Zander, Liśkiewicz, and Textor (2014). For an edge

$$P \rightarrow Q$$

P is the *parent* of *Q* and *Q* is the *child* of *P*. If there is a causal path from a variable *S* to a variable *T*, then T is a *descendant* of *S* and *S* is an *ancestor* of *T*. Thus the set of descendants of a variable *S* is the set of all variables that are caused (either directly or indirectly) by *S*. Similarly, the set of ancestors of a variable *T* is the set of all direct or indirect causes of *T*.

For a variable *V* on a path, *V* is a *collider* on the path if it has two arrows (one on each side) that point to it. A variable that is not a collider is a *noncollider*. The definition of a collider is path-specific. A variable can be a collider on one path but a noncollider on another path.

Statistical association between sets of variables can be divided into two components: a causal component and a noncausal or spurious component. If all spurious association can be removed, the causal effect is said to be *identified*. *Identification analysis* is the process of determining whether a causal effect can be identified and, if so, how to identify that effect. For more information about identification analysis, see the section "Causal Effect Identification" on page 2382.

Spurious association is typically removed through some form of statistical *adjustment* or *conditioning*. For example, you could compute an adjustment by including a variable as a regressor in a regression model or by stratifying the analysis by levels of the variable. You can use the causal model to determine which variables must be included in an adjustment set.

Sources of Association and Bias

A causal model that is represented by a DAG has unambiguous implications for the manner in which information can flow in the underlying data generating process. This flow of information is encapsulated by three graphical constructs that can be used to assemble every path in a DAG (Elwert 2013). The three constructs, which correspond to the three fundamental sources of association in a causal model, are as follows:

- causation
- confounding
- endogenous selection

These constructs are summarized graphically in Figure 37.4.

Figure 37.4 Three Fundamental Sources of Association

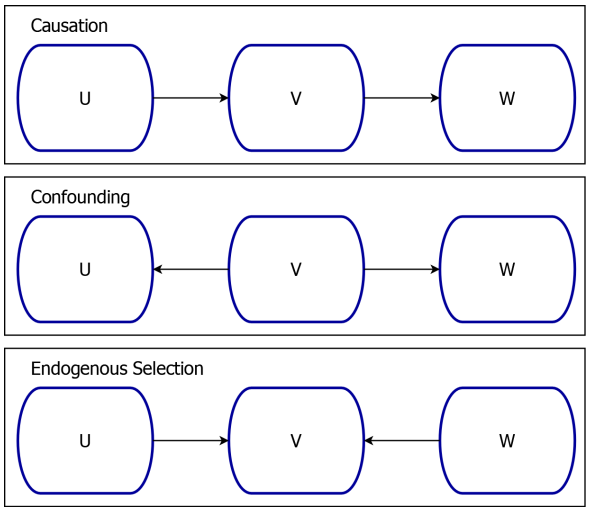

In the causal structure

$$U \rightarrow V \rightarrow W$$

the variables U and W are associated, and this association is the result of the causal chain. If you were to condition on the mediating variable V, then this would block the flow of information such that U and W would no longer be associated.

In the confounding structure

$$U \leftarrow V \rightarrow W$$

there is no causal path that relates U and W. However, U and W are still associated. This association is induced by the confounding variable V, the common parent of U and W. If you were to condition on the common cause V, then this would block the flow of information such that U and W would no longer be associated.

In the endogenous selection structure

$$U \rightarrow V \leftarrow W$$

the variables U and W jointly determine the value of their common child V, but U and W are not associated. However, if you were to condition on the common outcome V, then this would create a flow of information such that U and W would then be associated. For examples of endogenous selection, see Elwert and Winship (2014).

Loosely speaking, if you have a treatment variable (such as U) and an outcome variable (such as W) in a causal analysis, the goal is to eliminate noncausal association between U and W and leave the causal association unchanged. Thus the three fundamental graphical constructs correspond not only to the three fundamental sources of association but also to the three fundamental sources of bias. Generally, when you have a set of treatment and outcome variables, if you control for a variable on a causal path, you block the flow of information along that causal path. This is called overcontrol bias. Similarly, if you fail to control for a confounding common cause, some of the association between the treatment and outcome variables is the result of this confounding. This is called confounding bias. Finally, if you control for a common outcome, you create association between the treatment and outcome variables that is not causal. This is called endogenous selection bias. For an accessible discussion of the three fundamental sources of association and bias, see Elwert (2013) and Elwert and Winship (2014).

Statistical Properties of Causal Models

There are two ways to interpret the assumptions that are encoded within a DAG:

- A DAG is a "formal language for organizing claims about external interventions and their interactions" (Pearl 1993). For more information about this interpretation, see the section "Components of a Causal Graph" on page 2378.
- A DAG is a set of structures that define the flow of information between a set of variables. For more information about this interpretation, see the section "Sources of Association and Bias" on page 2380.

These two interpretations are equivalent under two additional assumptions (Elwert 2013):

- The variables in a DAG satisfy the *local Markov property*.

- The DAG satisfies the *weak faithfulness* property.

The local Markov property states that every variable in the DAG is statistically independent, conditional on its parents, of its set of nondescendants. In other words, the joint distribution function that is defined by the data generating process *factorizes* over the DAG (Koller and Friedman 2009).

The weak faithfulness property is discussed after the definition of d-separation.

A path in a DAG is said to be *d-separated* by a set of variables $\mathbf{Z}$ if either of the following conditions holds:

- The path contains a chain $U \rightarrow V \rightarrow W$ or a fork $U \leftarrow V \rightarrow W$ such that $V \in \mathbf{Z}$.
- The path contains a collider $U \rightarrow V \leftarrow W$ such that $V \notin \mathbf{Z}$ and such that no descendant of V is in $\mathbf{Z}$.

A path that is d-separated is said to be *blocked*; otherwise it is *nonblocked*. A set of variables $\boldsymbol{X}$ is d-separated from a set of variables $\boldsymbol{Y}$ by a set of variables $\boldsymbol{Z}$ if every path between a node in $\boldsymbol{X}$ and a node in $\boldsymbol{Y}$ is blocked.

The blocked/nonblocked terminology reflects the flow of information in a causal model. If a path is blocked, then information does not flow through that path. If the path is nonblocked, then information might flow through that path. The link between d-separation and information flow is embodied in the assumption of *weak faithfulness*. Weak faithfulness states that if two variables, X and Y, are not d-separated in a DAG, then the two variables are dependent in at least one distribution that factorizes over the DAG. The practical importance of faithfulness is that it does not permit the exact cancellation of the effects in a path (Elwert 2013). The use of weak faithfulness (rather than faithfulness) is consistent with the interpretation of edges as possible, rather than certain, effects (Spirtes, Glymour, and Scheines 2001; Pearl 2009b; Elwert 2013).

By interpreting a causal model as a DAG that represents the flow of association between variables, you can transform the causal assumptions that underlie a DAG into conditional independence statements. Specifically, if two variables are d-separated in a DAG by a set $\boldsymbol{Z}$, then those two variables must be statistically independent conditional on $\boldsymbol{Z}$. In other words, d-separation is a *global Markov property*. If a conditional independence statement contains only observed variables, then you can perform a statistical test by using the observed data to see whether the independence statement holds. Thus, the d-separation criterion determines the set of observationally testable implications of a causal model (Elwert 2013).

In fact, the global Markov property for DAGs (d-separation) and the local Markov property for DAGs are logically equivalent (Koller and Friedman 2009). If you have a complete list of either the local or global Markov properties, you can derive the other list by using the semigraphoid axioms (Pearl and Verma 1987; Geiger and Pearl 1988). In the CAUSALGRAPH procedure, you can use the IMAP option in the PROC CAUSALGRAPH statement to request a list of these properties.

Causal Effect Identification

The statistical association between a pair of variables can be divided into two components: a causal component and a noncausal or spurious component. If all spurious association can be removed, the causal effect is identified.

Identification by Adjustment

One possible approach to identification is *identification by adjustment*, which is the basis of causal effect identification in regression and matching (Elwert and Winship 2014). When you use identification by adjustment, you seek an *adjustment set*, a set of variables that, when controlled for in an analysis, blocks all the noncausal paths in a DAG without blocking any causal paths in that same DAG. The causal property of a

path is inherited from the direction of the edges in a model. That is, the causal property is a property of the causal model and does not change during an analysis. However, whether a path is blocked depends not only on the structure of the DAG that represents the causal model but also on the set of variables that are included in the adjustment set. Thus, you must carefully choose an adjustment set so as to remove all confounding bias without introducing any overcontrol or endogenous selection bias.

The following criterion is necessary and sufficient for a valid adjustment set (Shpitser, VanderWeele, and Robins 2010; Perković et al. 2018). For a set of treatment variables $\boldsymbol{X}$ and a set of outcome variables $\boldsymbol{Y}$, a set of observed variables $\boldsymbol{Z}$ is a valid adjustment set if all the following conditions are present:

- $\boldsymbol{Z}$ blocks all noncausal paths between $\boldsymbol{X}$ and $\boldsymbol{Y}$.
- No variable in $\boldsymbol{Z}$ lies on a causal path or descends from a causal path from $\boldsymbol{X}$ to $\boldsymbol{Y}$.
- No variable in $\boldsymbol{Z}$ is a descendant of any variable on a causal path (except possibly the variables in $\boldsymbol{X}$).

This criterion is identical to the constructive backdoor criterion (Van der Zander, Liśkiewicz, and Textor 2014). In the CAUSALGRAPH procedure, you can specify the METHOD=ADJUSTMENT option in the PROC CAUSALGRAPH statement to list all adjustment sets that satisfy the constructive backdoor criterion.

Pearl's backdoor criterion (Pearl 2009b) is a stronger criterion and thus can be used to produce a smaller list of adjustment sets. In PROC CAUSALGRAPH, you can specify the METHOD=BACKDOOR option to list all adjustment sets that satisfy the backdoor criterion.

Other adjustment criteria are designed to produce adjustment sets that satisfy special properties. For example, the parents-of-treatment criterion (Elwert 2013) produces adjustment sets that consist only of variables that are parents of a treatment variable. In PROC CAUSALGRAPH, you can specify the METHOD=PTREATMENT option to list all adjustment sets that satisfy the parents-of-treatment criterion. The parents-of-outcome (METHOD=POUTCOME) criterion (Elwert 2013) produces adjustment sets that consist only of variables that are parents of an outcome variable. The joint ancestor (METHOD=ANCESTOR) criterion (Elwert 2013) produces adjustment sets that consist only of variables that are ancestors of both a treatment variable and an outcome variable.

When a valid adjustment set $\boldsymbol{Z}$ has been identified, you can estimate the total causal effect of $\boldsymbol{X}$ on $\boldsymbol{Y}$ by using the stratification estimator (Shpitser, VanderWeele, and Robins 2010; Elwert 2013). For discrete data, this estimator has the form

$$P(\mathbf{Y} = \mathbf{y}|\mathrm{do}(\mathbf{X} = \mathbf{x})) = \sum_{z} P(\mathbf{Y} = \mathbf{y}|\mathbf{X} = \mathbf{x}, \mathbf{Z} = \mathbf{z}) P(\mathbf{Z} = \mathbf{z})$$

where the do-operator is intended to emphasize the interpretation of a causal effect as the result of an action or intervention (Pearl 2009b). In the language of potential outcomes, the preceding expression describes the distribution of potential outcomes under the assumption that the potential outcome and the treatment are independent after you condition on $\boldsymbol{Z}$. For more information about the relationship to potential outcomes, see the section "Causal Graphs and Potential Outcomes" on page 2384.

Although the stratification estimator is fully nonparametric, it is rarely used in practice. Instead, the distribution functions in the estimator are replaced by parametric functions. This task requires considerable care and expertise. Causal effect identification is a nonparametric concept. A poorly specified parametric model can produce biased or incorrect results. For a useful summary of computational tools that you can use to estimate an average causal effect, see Schafer and Kang (2008).

Identification by Instruments

The existence of an adjustment set is sufficient, but not necessary, to determine that a causal effect is identified. When an adjustment set does not exist, it might still be possible to estimate a causal effect by using a different method. For example, it is sometimes possible to estimate a causal effect by using an instrumental variable approach even when there is unobserved confounding between a treatment and an outcome. You can use the METHOD=IV option in the PROC CAUSALGRAPH statement to see whether a causal effect can be identified by using an instrumental variable.

Causal Graphs and Potential Outcomes

The CAUSALGRAPH procedure emphasizes the graphical or structural model framework of causality as developed by Pearl, among others (Pearl 2009b). This is in contrast with the Neyman-Rubin potential outcomes framework (Neyman, Dabrowska, and Speed 1990; Rubin 1980, 1990). Although the notation differs, these two frameworks are equivalent in the sense that any theorem that can be proved in one framework can also be proved in the other framework (Galles and Pearl 1998; Elwert 2013). For an extended discussion, see Pearl (2009b, chap. 7) and Pearl (2012).

For a single outcome variable *Y* and a single treatment variable *X*, the potential outcome *Y(x)* is a random variable that describes the possible values of the outcome variable for an experimental unit that is associated with the treatment *X*=*x*. The identity

$$P(Y(x) = y) = P(Y = y|\mathrm{do}(X = x))$$

establishes a map between the potential outcomes framework and the structural model framework (Pearl 2009b).

The do-operator is meant to emphasize the interpretation of a causal effect as the effect of an action or intervention (Pearl 2009b). That is, the quantity $P(Y = y|\mathrm{do}(X = x))$ reflects the distribution of the outcome variable that would result from the hypothetical act of intervening and imposing the condition *X*=*x* (Elwert 2013).

The correspondence between potential outcomes and the do-operator, together with the stratification estimator, indicates another important link between the potential outcomes framework and the structural model framework (Elwert 2013). In particular, the conditional ignorability of treatment assignment (in the potential outcomes framework) is exactly equivalent to the criterion for the existence of an adjustment set. That is, $Y(x) \perp X|\mathbf{Z}$ if and only if $\boldsymbol{Z}$ satisfies the adjustment criterion (Shpitser, VanderWeele, and Robins 2010). For more information about the adjustment criterion, see the section "Identification by Adjustment" on page 2382. Significantly, the adjustment criterion involves only observed quantities, whereas conditional ignorability requires reasoning about counterfactuals that might be unobserved (Elwert 2013).

Choosing an Identification Strategy

As discussed in the sections "Identification by Adjustment" on page 2382 and "Identification by Instruments" on page 2384, PROC CAUSALGRAPH offers several different methods that you can use to identify a causal effect. You choose one of these methods by specifying the METHOD= option in the PROC CAUSALGRAPH statement.

Most of the popular methods of estimating a causal effect (matching, outcome regression, methods based on propensity scores) are consistent with identification by adjustment. That is, you use one of the adjustment methods (METHOD=ADJUSTMENT, ANCESTOR, BACKDOOR, POUTCOME, or PTREATMENT) to find a valid adjustment set. You then use the variables in this adjustment set as covariates in your preferred analysis.

The choice of adjustment method depends on your goals and preferences. The constructive backdoor criterion (METHOD=ADJUSTMENT) is a necessary and sufficient condition for adjustment (Shpitser, VanderWeele, and Robins 2010). This means that this criterion can find every possible adjustment set within a causal model (when used with the LIST option) and can always determine whether or not a proposed adjustment set is valid (when used with the TESTID statement).

The remaining adjustment criteria are sufficient but not necessary. This means that, when used with the LIST option, these criteria might find some valid adjustment sets, but they are not guaranteed to find all adjustment sets. When used with a TESTID statement, an adjustment set that meets one of these criteria is guaranteed to be valid, but an adjustment set that does not meet one of these criteria is not necessarily invalid. In spite of these limitations, these sufficient criteria have several advantages:

- They are more computationally efficient than the constructive backdoor criterion (METHOD= ADJUSTMENT).
- They are simpler than the adjustment criterion and are better established within certain research communities.
- The adjustment sets that satisfy one of these criteria have special properties that might make them more useful in the context of a specific analysis. For example, if you intend to build a propensity score model, it might be more intuitive to use covariates that are parents of the treatment variable(s). If you intend to use outcome regression, it might be more intuitive to use covariates that are parents of the outcome variable(s).

Not all causal effects can be identified by adjustment. In particular, if your hypothesized model exhibits unmeasured confounding between a treatment variable and an outcome variable, the adjustment methods will not be successful. In this case, you should look for an instrumental variable (METHOD=IV). The section "Dealing with Nonidentified Causal Effects" on page 2388 contains information about steps that you can take when your causal effect of interest is not identified.

Arrow or Edge Specification

In the CAUSALGRAPH procedure, edges in a causal model are specified in the MODEL statement as either right, left, or bidirected arrows. In addition, the treatment and outcome variables are separated by a right arrow in the IDENTIFY statement.

The following table summarizes the edges and the syntax that you can specify in PROC CAUSALGRAPH:

Edge Type	Syntax	Statements
Right arrow	`>`, `->`, `->`, `-->`, `=>`, `==>`, `===>`	MODEL, IDENTIFY
Left arrow	`<`, `<-`, `<-`, `<--`, `<=`, `<==`, `<===`	MODEL
Bidirected arrow	`<>`, `<->`, `<->`, `<-->`, `<=>`, `<==>`, `<===>`	MODEL

INMODEL= and OUTMODEL= Data Sets

The OUTMODEL= option creates an output data set that contains the model specifications for the current analysis by PROC CAUSALGRAPH. An OUTMODEL= data set can be used as an input INMODEL= data set in a subsequent analysis by the procedure. Similarly, the INMODEL= data set is usually created in a previous run of PROC CAUSALGRAPH in which the OUTMODEL= option was specified. Together, the INMODEL= and OUTMODEL= options are useful if you want to reuse a causal diagram for multiple analyses. For example, you might want to analyze a causal model when different sets of variables are unmeasured, or you might want to store a common base model for use with the REFMODEL= option.

It is important to realize that INMODEL= and OUTMODEL= data sets contain only the information about the specification of the causal model(s). These data sets do not store any information about treatment variables, outcome variables, or unmeasured variables.

You can alter an existing OUTMODEL= data set in the DATA step to create the INMODEL= data set that describes a modified model. However, editing and modifying OUTMODEL= data sets require a good understanding of the format and contents of the OUTMODEL= data sets and can be error-prone.

The OUTMODEL= data set is of TYPE=CALISMDL. For more information about this type of data set, see the section "OUTMODEL= or OUTRAM= Data Set" on page 1719 in Chapter 33, "The CALIS Procedure." PROC CAUSALGRAPH uses the following variables to represent the causal model(s):

- the numeric variable _MDLNUM_ for model numbers
- the character variable _MDLNAM_ for model labels
- the character variable _TYPE_, which takes various values that indicate the type of model specification
- the character variable _NAME_, which indicates the model type
- the character variable _VAR1_, which is the name of the first variable in the specification
- the character variable _VAR2_, which is the name of the second variable in the specification

Each observation (record) of the OUTMODEL= data set contains a piece of information about the model specification. Depending on the type of the specification that is indicated by the value of the _TYPE_ variable, the meanings of _NAME_, _VAR1_, and _VAR2_ differ. The following table summarizes the meanings of the _NAME_, _VAR1_, and _VAR2_ fields for different values of the _TYPE_ variable:

TYPE=	Description	_NAME_	_VAR1_	_VAR2_
MDLTYPE	Model type	PATH		
LEFT	Path coefficient	*(empty)*	Child variable	Parent variable
RIGHT	Path coefficient	*(empty)*	Parent variable	Child variable
PCOV	(Partial) covariance	*(empty)*	First variable	Second variable

For data sets that PROC CAUSALGRAPH uses, the value of the _NAME_ variable is always PATH for an observation for which _TYPE_=MDLTYPE.

Limitations of the CAUSALGRAPH Procedure

Causal Models with Directed Cycles

PROC CAUSALGRAPH analyzes DAGs that represent a causal model. These DAGs cannot contain a directed cycle. This can lead to difficulties in situations where two variables seem to cause (directly or indirectly) each other. In such situations, a common approach is to introduce additional variables so as to describe the data generating process at a more refined temporal scale (Greenland, Pearl, and Robins 1999; Elwert 2013).

Identifying Joint Treatment Effects

The CAUSALGRAPH procedure enables you to specify multiple treatment variables and multiple outcome variables in an identification analysis.

When you specify multiple treatment variables, the causal effect is interpreted as a *joint causal effect*. That is, the causal effect is interpreted as the hypothetical result of imposing specific values on all treatment variables simultaneously (Elwert 2013). In the language of the do-operator, the joint causal effect is treated as a conjunction so that $\mathrm{do}(X_1 = x_1, X_2 = x_2)$ is interpreted as $\mathrm{do}(X_1 = x_1)$ and $\mathrm{do}(X_2 = x_2)$.

You can also interpret multiple treatment variables as sequential treatment actions, provided that the treatment sequence is predetermined (Elwert 2013). However, you cannot use PROC CAUSALGRAPH to assess the identifiability of a dynamic treatment regime.

When you specify multiple outcome variables, each outcome is interpreted separately as a unique causal effect. Although the interpretation is separate, PROC CAUSALGRAPH constructs only those adjustment sets that are valid for every outcome variable. In some situations, there might not be any such adjustment sets, even though it is possible to identify the causal effect on each outcome separately. For example, if the causal effect of X on Y_1 can be identified only with an adjustment set $\mathbf{Z}_1$ and the causal effect of X on Y_2 can be identified only with an adjustment set $\mathbf{Z}_2$ for disjoint sets $\mathbf{Z}_1$ and $\mathbf{Z}_2$, then there is no adjustment set that is valid for both outcome variables simultaneously.

Causal Effect Identification Is a Population Concept

A causal effect that you estimate from observational data cannot have a valid causal interpretation unless those data are supplemented by a set of causal assumptions in the form of a causal model (Pearl 2009b). However, the causal model represents assumed relationships between variables at the population level and not at the level of an individual subject. Therefore, the theory that describes causal effect identification by using DAGs does not consider sampling variability. The conditions for identification are valid in the asymptotic limit (as the number of observations increases) (Elwert 2013). For this reason, a successful identification strategy (using either an adjustment set or a conditional instrumental variable) is a necessary first step to estimate a causal effect by using data from a nonrandomized experiment (Elwert and Winship 2014). You should carefully consider the role of sampling variability when estimating a causal effect and when examining the testable implications of a model.

Causal Effect Identification Is a Nonparametric Concept

The identifiability of a causal effect is a fully nonparametric concept in the sense that it does not depend on distributional or functional forms for the variables and edges in a causal model. However, an identification strategy as well as any estimate that is computed by that strategy should be understood to be conditional on the validity of the assumed causal model (Elwert 2013). In addition, when a causal effect is shown to be identified (for example, using an adjustment set), this does not mean that you can freely choose a parametric estimator in order to quantify the effect. The suitability of a parametric estimator is contingent on parametric assumptions. These assumptions are separate from the assumptions of the causal model and must be justified for each specific situation (Elwert 2013).

Dealing with Nonidentified Causal Effects

When a causal effect cannot be identified in a particular causal model, there are a couple of actions that you can take. First, you can revise the assumptions of the causal model to see whether the data generating process might be equally well described by an alternative model. In some cases, this might involve testing the observable implications of a model against existing data. For more information about the testable implications of a model, see the section "Statistical Properties of Causal Models" on page 2381. Second, you can consider observing additional variables. This might take the form of adding observations for a previously unmeasured variable or adding new variables and edges to an existing model (Pearl 2009b). However, adding edges to an existing set of variables never helps, and might harm, identification (Pearl 2009b; Elwert and Winship 2014).

Time Requirements

The following table summarizes the approximate computational cost to perform various tasks by using the CAUSALGRAPH procedure. The computational costs are computed for a DAG that contains N nodes (variables) and M edges.

For certain tasks, such as listing all adjustment sets, the output might grow exponentially large with the size of the graph. For these tasks, the computational cost is given in terms of delay complexity. The delay complexity is the computational cost that is associated with producing each element of the output set (Takata 2010).

The time required to...	Is roughly proportional to
Test an adjustment	N^2
Find one adjustment	N^2
List all adjustments	$N(N+M)$ delay
List all minimal adjustments	N^3 delay
Test an instrument	$N+M$
List all instruments	NM delay
List local Markov properties	$N(N+M)$
List global Markov properties	$N(N+M)$ delay

In general, the algorithms to test or find a single adjustment set have a computational cost proportional to $(N+M)$ (Van der Zander, Liśkiewicz, and Textor 2014). However, PROC CAUSALGRAPH always checks whether an adjustment set is minimal, and this operation is associated with a computational cost proportional to N^2.

By default, PROC CAUSALGRAPH sorts adjustment sets so that the smallest sets are printed first. You can use the MAXLIST= option in the PROC CAUSALGRAPH statement to limit the printed output, but this does not change the computational cost unless you also specify the NOSORT option in the same statement.

In certain situations, you might simply want to ascertain whether any adjustment set exists, regardless of which variables that adjustment set includes. To do this efficiently, you can request a single adjustment set by combining the NOSORT option with the MAXLIST=1 option. This enables the procedure to quickly find and display a single adjustment set. In the special case where you specify METHOD=ADJUSTMENT together with the NOSORT and MAXLIST=1 options in the PROC CAUSALGRAPH statement, the procedure switches to an even more efficient algorithm (Van der Zander, Liśkiewicz, and Textor 2014). For an illustration, see Example 37.2.

ODS Table Names

PROC CAUSALGRAPH assigns a name to each table that it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 37.3. The specific statements and options that produce these output tables are shown in the last two columns. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 37.3 ODS Tables Produced by the CAUSALGRAPH Procedure

ODS Table Name	Description	Statement	Option
AdjustList	List of adjustment sets	PROC CAUSALGRAPH	LIST
AdjustTest	Test results of adjustment sets	TESTID	Default
ConnectedComponents	List of connected components	PROC CAUSALGRAPH	Default
DirectedCycles	List of directed cycles	PROC CAUSALGRAPH	Default
GraphicalModelSummary	Summary of the causal models		Default

Table 37.3 *continued*

ODS Table Name	Description	Statement	Option
Imap	List of derived conditional independence assumptions	PROC CAUSALGRAPH	IMAP
InstrumentList	List of instrumental variables	PROC CAUSALGRAPH	LIST
InstrumentTest	Test results of instrumental variables	TESTID	Default
Paths	Summary of proper paths in a model	TESTID	PATHS
VariableInfo	Summary of variables in the causal models		Default

Examples: CAUSALGRAPH Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 37.1: Constructing Adjustment Sets

(View the complete code for this example (cagrex1.sas) in the example repository.)

This example illustrates how you can use the CAUSALGRAPH procedure to construct adjustment sets from a causal model. An adjustment set is a set of variables that can be used to remove noncausal association between the treatment and outcome variables in a causal model. If an adjustment set exists, then the causal effect of the treatment variables on the outcome variables is identified. The identification implies that it is possible to estimate the specified causal effects from data. For more information about identifying causal effects in a causal model, see the section "Causal Effect Identification" on page 2382.

The causal model shown in Figure 37.5, which has been adapted from Timmermann et al. (2017), examines the relationship between maternal exposure to persistent perfluoroalkyl substances (PFAS) and breastfeeding duration among residents of the Faroe Islands. For a summary of the variables in the model, see the example in the section "Getting Started: CAUSALGRAPH Procedure" on page 2357. For this example, it is assumed that all variables are observed.

Figure 37.5 Causal Model of the Effect of Persistent Perfluoroalkyl Substances on Breastfeeding Duration

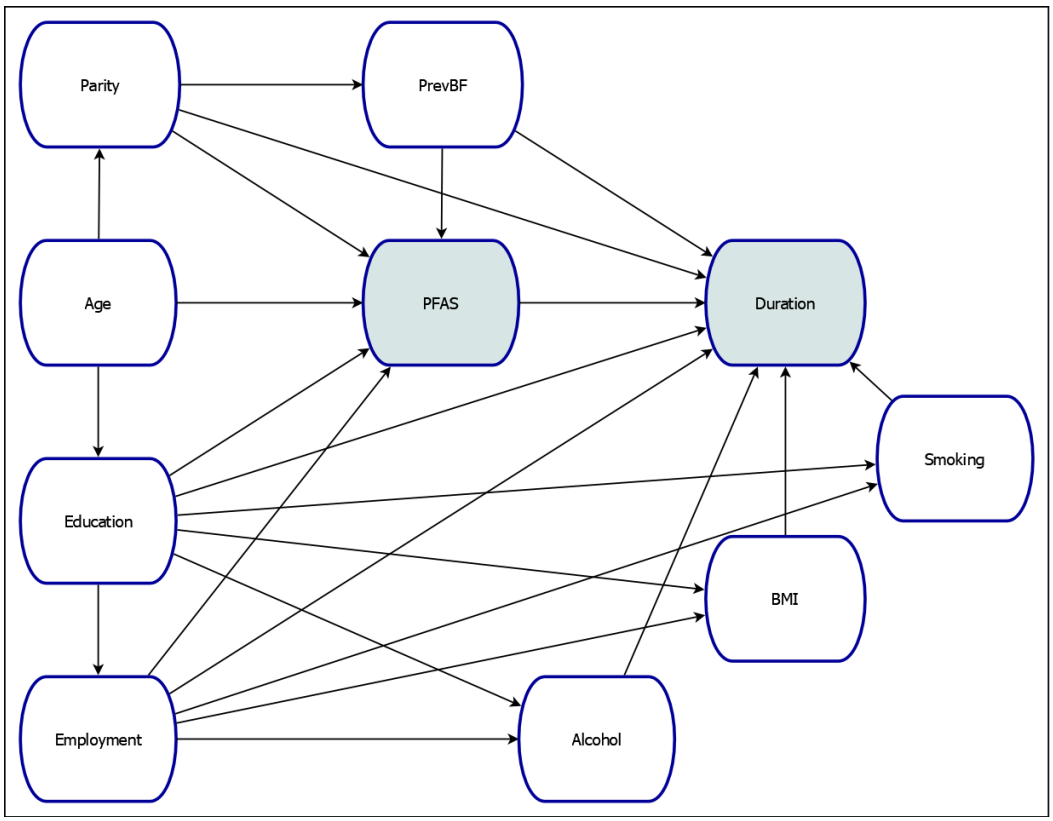

In a naive approach to identify the causal effect of the variable PFAS on the variable Duration, you might consider an adjustment set that includes all observed covariates. The following statements invoke the CAUSALGRAPH procedure to test whether such an adjustment set is valid:

```
proc causalgraph;
   model "Timm17AllObs"
      Age ==> Parity PFAS Education,
      Parity ==> PrevBF Duration PFAS,
      PrevBF ==> PFAS Duration,
      PFAS ==> Duration,
      Education ==> Duration Employment PFAS BMI Alcohol Smoking,
      Employment ==> Duration PFAS BMI Alcohol Smoking,
      BMI Alcohol Smoking ==> Duration;
   identify PFAS ==> Duration;
   testid "All Covariates" Age Education Employment Parity Alcohol
      Smoking BMI PrevBF;
run;
```

In the MODEL statement, you specify the causal model to be analyzed. The quoted string in the statement labels the model. The remainder of the MODEL statement specifies all the variables and edges in the model. These variables and edges reflect the hypothesized data generating process shown in Figure 37.5.

In the IDENTIFY statement, you specify the causal effect of interest. You can use this statement to specify one or more treatment variables and one or more outcome variables. The treatment and outcome variables

are separated by a single right arrow, ==>. In this example, you are interested in testing the identification of the causal effect of the variable PFAS on the variable Duration.

Because the METHOD= option is not specified in the PROC CAUSALGRAPH statement, the procedure uses the constructive backdoor criterion (METHOD=ADJUSTMENT) by default to test the identification of the causal effect by the adjustment set that you specify in the TESTID statement.

The "Variables in Model" and "Graphical Model Summary" tables (Output 37.1.1) summarize the variables and edges in the causal model. You can use this information as a qualitative check of the model specification.

Output 37.1.1 Input Summary Tables for the Causal Model in Figure 37.5

The CAUSALGRAPH Procedure

Variables in Model

	N	Variables
Measured	10	Age Alcohol BMI Duration Education Employment Parity PFAS PrevBF Smoking
Unmeasured	0	

Graphical Model Summary

Model	Nodes	Edges	Treatments	Outcomes	Measured	Unmeasured
Timm17AllObs	10	23	1	1	10	0

Output 37.1.2 displays the results of the test for adjustment. Although the proposed adjustment is sufficient to identify the causal effect, it is not a minimal adjustment. If you use this adjustment set, the estimation of the causal effect might be computationally inefficient. In addition, you have to collect data for all these variables in order to estimate the causal effect.

Output 37.1.2 Adjustment Set Test Output for the Causal Model in Figure 37.5

Covariate Adjustment Test: All Covariates

Causal Effect of PFAS on Duration

				Covariates							
Model	**Size**	**Valid**	**Minimal**	**Age**	**Alcohol**	**BMI**	**Education**	**Employment**	**Parity**	**PrevBF**	**Smoking**
Timm17AllObs	8	Yes	No	*	*	*	*	*	*	*	*

You can use the CAUSALGRAPH procedure to see whether there are any smaller adjustment sets that can be used to identify the causal effect shown in Figure 37.5. The following statements produce a list of all possible adjustment sets that can be used to estimate the causal effect of PFAS on Duration in the model:

```
proc causalgraph;
   model "Timm17AllObs"
      Age ==> Parity PFAS Education,
      Parity ==> PrevBF Duration PFAS,
      PrevBF ==> PFAS Duration,
      PFAS ==> Duration,
      Education ==> Duration Employment PFAS BMI Alcohol Smoking,
      Employment ==> Duration PFAS BMI Alcohol Smoking,
      BMI Alcohol Smoking ==> Duration;
   identify PFAS ==> Duration;
run;
```

As before, the METHOD= option is not specified here, so the constructive backdoor criterion (METHOD=ADJUSTMENT) is used by default. In this case, because no TESTID statement is specified, all adjustment sets are listed. Alternatively, when you specify a TESTID statement, you can use the LIST option in the PROC CAUSALGRAPH statement to list all adjustment sets.

As shown in Output 37.1.3, the causal model contains 16 valid adjustment sets that can be used to identify the causal effect of PFAS on Duration. These 16 sets differ in size. The last set contains eight covariates that you have tested previously by using the TESTID statement. If data collection cost is a concern, then you can consider other adjustment sets with smaller sizes. As shown in Output 37.1.3, the first 15 sets all have smaller sizes, ranging from four to seven. The first set is marked as a minimal because the identification of the specified causal effect would not hold if any one of these four variables were removed from the adjustment set.

Output 37.1.3 Adjustment Set List Output for the Causal Model in Figure 37.5

Covariate Adjustment Sets for Timm17AllObs
Causal Effect of PFAS on Duration

			Covariates							
	Size	Minimal	Age	Alcohol	BMI	Education	Employment	Parity	PrevBF	Smoking
1	4	Yes				*	*	*	*	
2	5	No	*			*	*	*	*	
3	5	No		*		*	*	*	*	
4	5	No			*	*	*	*	*	
5	5	No				*	*	*	*	*
6	6	No	*	*		*	*	*	*	
7	6	No	*		*	*	*	*	*	
8	6	No	*			*	*	*	*	*
9	6	No		*	*	*	*	*	*	
10	6	No		*		*	*	*	*	*
11	6	No			*	*	*	*	*	*
12	7	No	*	*	*	*	*	*	*	
13	7	No	*	*		*	*	*	*	*
14	7	No	*		*	*	*	*	*	*
15	7	No		*	*	*	*	*	*	*
16	8	No	*	*	*	*	*	*	*	*

You can use the MAXLIST=, MAXSIZE=, or MINIMAL option in the PROC CAUSALGRAPH statement to reduce the number of adjustment sets that are displayed. For example, the following statements use the MINIMAL option to display only minimal adjustment sets:

```
proc causalgraph minimal;
   model "Timm17AllObs"
      Age ==> Parity PFAS Education,
      Parity ==> PrevBF Duration PFAS,
      PrevBF ==> PFAS Duration,
      PFAS ==> Duration,
      Education ==> Duration Employment PFAS BMI Alcohol Smoking,
      Employment ==> Duration PFAS BMI Alcohol Smoking,
      BMI Alcohol Smoking ==> Duration;
   identify PFAS ==> Duration;
run;
```

The table in Output 37.1.4 shows all the minimal adjustment sets for the causal model. In this example, there is only a single minimal adjustment set.

Output 37.1.4 Minimal Adjustment Sets for the Causal Model in Figure 37.5

Covariate Adjustment Sets for Timm17AllObs Causal Effect of PFAS on Duration									
			Covariates						
	Size	Minimal	Age	Alcohol	BMI	Education	Employment	Parity	PrevBF	Smoking
1	4	Yes				*	*	*	*	

Example 37.2: Searching Efficiently for an Adjustment Set

(View the complete code for this example (cagrex2.sas) in the example repository.)

This example illustrates how you can use the CAUSALGRAPH procedure to quickly determine whether an adjustment set exists in a particular causal model. If an adjustment set exists, then the causal effect of the treatment variables on the outcome variables is identified. The identification implies that it is possible to estimate the specified causal effects from the data. For this reason, you might want to determine whether such a set exists, regardless of what that set might be. For more information about identifying causal effects in a causal model, see the section "Causal Effect Identification" on page 2382.

The causal model shown in Figure 37.6, which has been adapted from Timmermann et al. (2017), examines the relationship between maternal exposure to persistent perfluoroalkyl substances (PFAS) and breastfeeding duration among residents of the Faroe Islands. For a summary of the variables in the model, see the section "Getting Started: CAUSALGRAPH Procedure" on page 2357. For this example, the causal model includes an additional variable, HealthBehavior, which is assumed to be a latent construct that represents the degree to which an individual's behavior is considered to be healthful. The variables HealthBehavior and PrevBF are assumed to be unobserved, as in Timmermann et al. (2017).

Figure 37.6 Causal Model of the Effect of Persistent Perfluoroalkyl Substances on Breastfeeding Duration

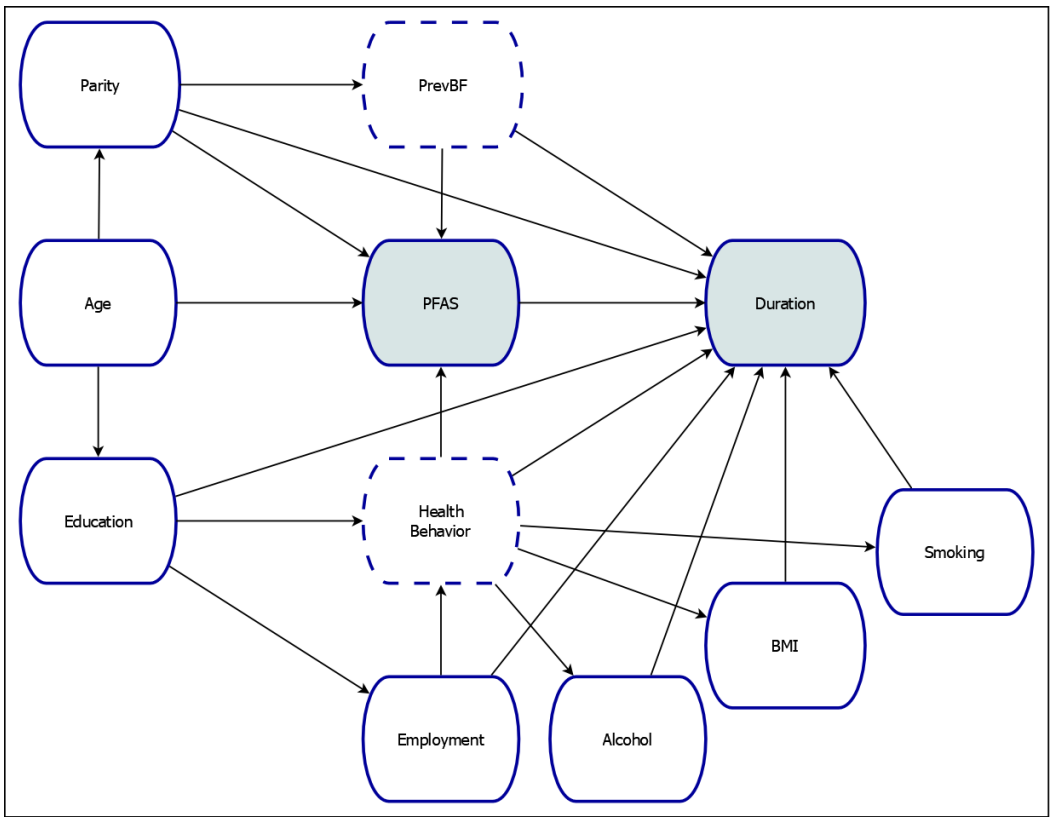

The following statements invoke the CAUSALGRAPH procedure to determine whether it is possible to find an adjustment set that can be used to estimate the causal effect:

```
proc causalgraph method=adjustment maxlist=1 nosort;
   model "Timm17HealthBehavior"
      Age ==> Parity PFAS Education,
      Parity ==> PrevBF Duration PFAS,
      PrevBF ==> PFAS Duration,
      PFAS ==> Duration,
      Education ==> Duration HealthBehavior Employment,
      HealthBehavior ==> PFAS Duration BMI Alcohol Smoking,
      Employment ==> HealthBehavior Duration,
      BMI Alcohol Smoking ==> Duration;
   identify PFAS ==> Duration;
   unmeasured PrevBF HealthBehavior;
run;
```

In the MODEL statement, you specify the causal model to be analyzed. The quoted string in the statement labels the model. The remainder of the MODEL statement specifies all the variables and edges in the model. These variables and edges reflect the hypothesized data generating process shown in Figure 37.6.

In the IDENTIFY statement, you specify the causal effect of interest. You can use this statement to specify one or more treatment variables and one or more outcome variables. The treatment and outcome variables are separated by a single right arrow, `==>`. In this example, you are interested in testing the identification of the causal effect of the variable PFAS on the variable Duration.

In the UNMEASURED statement, you specify variables that are not observed and thus cannot be included in an adjustment set. In this example, the variables PrevBF and HealthBehavior are unmeasured.

The METHOD=ADJUSTMENT uses the constructive backdoor criterion for identifying causal effects. The MAXLIST=1 prints only a single adjustment set. The NOSORT option specifies that it is not necessary to print the smallest adjustment sets first. These options enable the procedure to use an efficient algorithm to quickly find an adjustment set, if at least one adjustment set exists. For more information about the computational complexity of the algorithms that PROC CAUSALGRAPH uses, see the section "Time Requirements" on page 2388.

As shown in Output 37.2.1, for the causal model in Figure 37.6, it is not possible to use an adjustment set to identify the effect of PFAS on Duration.

Output 37.2.1 Adjustment Set List Summary Note

NOTE: There are no adjustment sets satisfying the specified criteria for Timm17HealthBehavior.

Although you cannot use an adjustment set to estimate the causal effect in Figure 37.6, it is still possible to estimate the causal effect if you are willing to make additional parametric assumptions in the model. See Example 37.5 for an example.

Example 37.3: Testing Adjustments and Enumerating Paths

(View the complete code for this example (cagrex3.sas) in the example repository.)

This example illustrates how you can use the CAUSALGRAPH procedure to determine whether a proposed adjustment set is valid for estimating a causal effect in a particular causal model. It also illustrates how you can enumerate the paths that connect the treatment and outcome variables in the model. If a causal effect cannot be identified in a causal model, the enumeration of paths can provide information that might help you modify the study design in order to enable identification of the causal effect.

The causal model shown in Figure 37.7 is used by Thornley et al. (2013) to examine the relationship between an individual's serum urate and risk of cardiovascular disease. The model includes the following variables:

- Urate: the treatment variable
- CVD: the outcome variable
- AntiHypertensiveUse: indicator of antihypertensive drug use
- Creatinine: measured serum creatinine level
- Diabetes: indicator of diabetes diagnosis
- Ethnicity: classification variable for ethnicity
- Gender: indicator for biological male
- Gout: indicator of gout diagnosis
- HbA1c: measured glycated hemoglobin
- MedicationPropensity: latent construct that reflects an individual's propensity to take prescribed medication
- Nutrition: latent construct that reflects diet or nutrition
- Obesity: indicator of body mass index ≥ 30
- CurrentBP: measured blood pressure
- CurrentHDL: measured HDL cholesterol

- PreviousBP: previous (prior to study) blood pressure
- PreviousHDL: previous (prior to study) HDL cholesterol
- Smoking: indicator of current smoking status
- StatinUse: indicator of statin drug use

The variables MedicationPropensity and Nutrition correspond to latent constructs and thus cannot be observed. It is also assumed that the variables PreviousBP, PreviousHDL, and Obesity are not observed.

Figure 37.7 Causal Model of the Effect of Serum Urate on Risk of Cardiovascular Disease

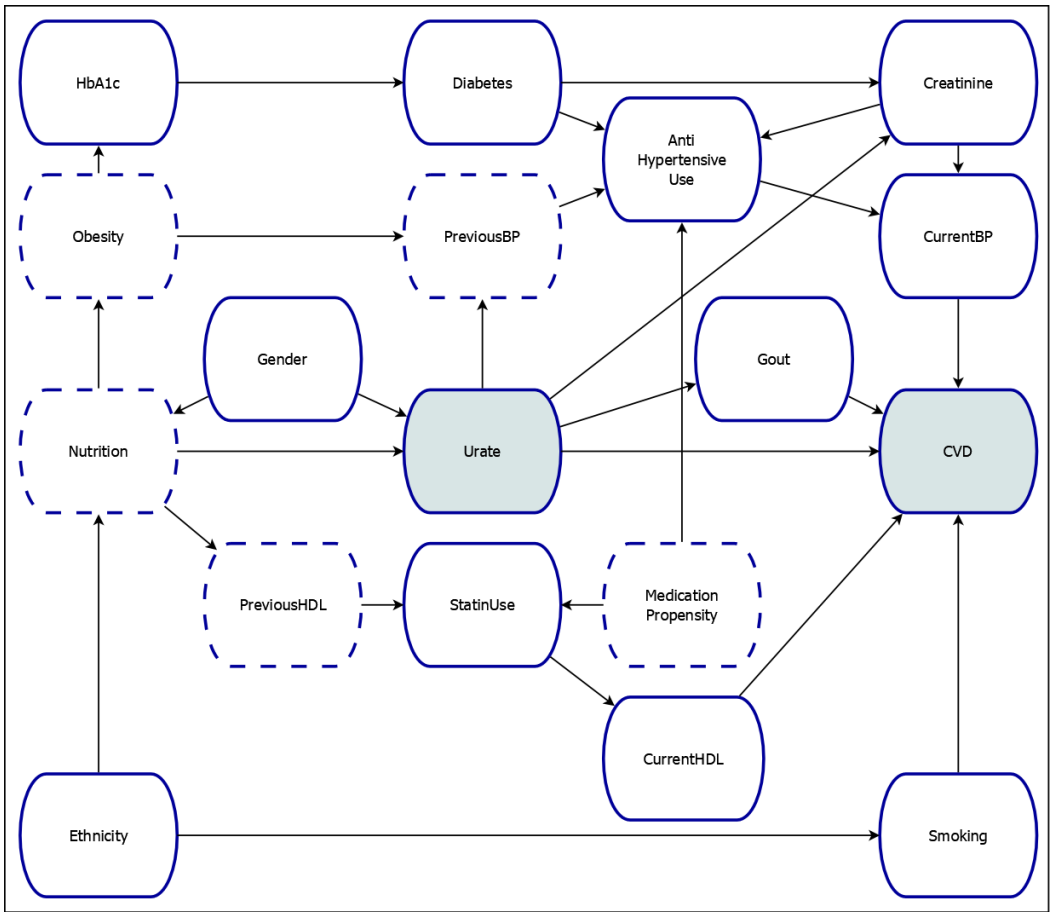

According to this causal model, the statistical association between the variables Urate and CVD reflects a combination of true causal association and additional spurious or noncausal association. In order to isolate the true causal association between these two variables, Thornley et al. (2013) consider adjustment for CurrentHDL, Ethnicity, Gender, HbA1c, and Smoking. The following code invokes the CAUSALGRAPH procedure to test whether this adjustment set can be used to estimate the causal effect of Urate on CVD according to the causal model:

```
proc causalgraph compact;
   model "Thor12"
      AntiHypertensiveUse ==> CurrentBP,
      Creatinine ==> AntiHypertensiveUse CurrentBP,
      CurrentBP ==> CVD,
      CurrentHDL ==> CVD,
```

```
      Diabetes ==> AntiHypertensiveUse Creatinine,
      Ethnicity ==> Nutrition Smoking,
      Gender ==> Nutrition Urate,
      Gout ==> CVD,
      HbA1c ==> Diabetes,
      MedicationPropensity ==> AntiHypertensiveUse StatinUse,
      Nutrition ==> PreviousHDL Urate Obesity,
      Obesity ==> PreviousBP HbA1c,
      PreviousBP ==> AntiHypertensiveUse,
      PreviousHDL ==> StatinUse,
      Smoking ==> CVD,
      StatinUse ==> CurrentHDL,
      Urate ==> PreviousBP Creatinine CVD Gout;
   identify Urate ==> CVD;
   unmeasured Nutrition Obesity PreviousBP MedicationPropensity PreviousHDL;
   testid CurrentHDL Ethnicity Gender HbA1c Smoking;
run;
```

In the MODEL statement, you specify the causal model to be analyzed. The quoted string in the statement labels the model. The remainder of the MODEL statement specifies all the variables and edges in the model. These variables and edges reflect the hypothesized data generating process shown in Figure 37.7.

In the IDENTIFY statement, you specify the causal effect of interest. You can use this statement to specify one or more treatment variables and one or more outcome variables. The treatment and outcome variables are separated by a single right arrow, `==>`. In this example, you are interested in testing the identification of the causal effect of Urate on CVD.

In the UNMEASURED statement, you specify variables that are not observed and thus cannot be included in an adjustment set. In this example, five variables are specified as unmeasured.

Because the METHOD= option is not specified in the PROC CAUSALGRAPH statement, the procedure uses the constructive backdoor criterion (METHOD=ADJUSTMENT) by default to test the identification of the causal effect by the adjustment set that you specify in the TESTID statement.

The COMPACT option in the PROC CAUSALGRAPH statement displays the output table in a compact manner. For this example, this means that a column is added for a covariate in the adjustment test output table only if that variable actually appears in the test.

As shown in Output 37.3.1, the proposed adjustment set is marked “No” in the Valid column, so it is not sufficient to estimate the causal effect of Urate on CVD.

Output 37.3.1 Adjustment Set Test for the Model in Figure 37.7

Covariate Adjustment Test: Test1

Causal Effect of Urate on CVD

				Covariates				
Model	**Size**	**Valid**	**Minimal**	**CurrentHDL**	**Ethnicity**	**Gender**	**HbA1c**	**Smoking**
Thor12	5	No	No	*	*	*	*	*

To see why the proposed adjustment set is not valid, you can request an enumeration of the proper paths that connect the treatment to the outcome in the model. You can also use the procedure to search for a valid adjustment set. The following code invokes the CAUSALGRAPH procedure to perform these two tasks:

```
proc causalgraph compact list;
   model "Thor12"
      AntiHypertensiveUse ==> CurrentBP,
      Creatinine ==> AntiHypertensiveUse CurrentBP,
      CurrentBP ==> CVD,
      CurrentHDL ==> CVD,
      Diabetes ==> AntiHypertensiveUse Creatinine,
      Ethnicity ==> Nutrition Smoking,
      Gender ==> Nutrition Urate,
      Gout ==> CVD,
      HbA1c ==> Diabetes,
      MedicationPropensity ==> AntiHypertensiveUse StatinUse,
      Nutrition ==> PreviousHDL Urate Obesity,
      Obesity ==> PreviousBP HbA1c,
      PreviousBP ==> AntiHypertensiveUse,
      PreviousHDL ==> StatinUse,
      Smoking ==> CVD,
      StatinUse ==> CurrentHDL,
      Urate ==> PreviousBP Creatinine CVD Gout;
   identify Urate ==> CVD;
   unmeasured Nutrition Obesity PreviousBP MedicationPropensity PreviousHDL;
   testid Gender HbA1c Ethnicity Smoking
      CurrentHDL / paths=(noncausal nonblocked);
run;
```

The PATHS option in the TESTID statement requests an analysis of the proper paths in the model when the variables in the TESTID statement are being adjusted for identifying the causal effect. By default, the procedure prints all proper paths when you specify the PATHS option. You can change this behavior by specifying additional suboptions. In this example, the NONCAUSAL and NONBLOCKED suboptions display only noncausal paths that are not blocked. For more information about the terminology of paths in DAGs, see the section "Terminology" on page 2379.

Output 37.3.2 shows the proper paths in the model that are not causal and are not blocked. There are two such paths. If you were to use the proposed adjustment set, some of the association between Urate and CVD would be attributable to these two noncausal paths, and the causal effect would not be estimated correctly. The first path is not blocked because the variable StatinUse is a collider on the path and one of its descendants, the variable CurrentHDL, appears in the adjustment set. The second path does not contain any colliders but is not blocked because it does not contain any element of the proposed adjustment set. For more details about the flow of information in DAGs, see the section "Statistical Properties of Causal Models" on page 2381.

Output 37.3.2 Analysis of Proper Paths

Treatment-to-Outcome Paths

Test Test1 for Model Thor12

Causal Effect of Urate on CVD Adjusted for {CurrentHDL Ethnicity Gender HbA1c Smoking}

	Causal	Blocked	Paths
1*	No	No	Urate <== Nutrition ==> PreviousHDL ==> StatinUse <== MedicationPropensity ==> AntiHypertensiveUse ==> CurrentBP ==> CVD
2*	No	No	Urate <== Nutrition ==> Obesity ==> PreviousBP ==> AntiHypertensiveUse ==> CurrentBP ==> CVD

*** indicates a biasing path**

Because the adjustment set in the TESTID statement leads to two nonblocked, noncausal paths, it is not valid

for identifying the specified causal effect. The next question to ask is whether there are any valid adjustment sets. You can use the LIST option in the PROC CAUSALGRAPH statement to address this question. The LIST option lists all valid adjustment sets.

The note in Output 37.3.3 summarizes the results of the search for any adjustment set that can be used to identify the effect of Urate on CVD in the model in Figure 37.7. For this causal model, it is not possible to use an adjustment set to estimate the causal effect.

Output 37.3.3 Adjustment Set List Summary Note

NOTE: There are no adjustment sets satisfying the specified criteria for Thor12.

In order to obtain identifiability, you might consider collecting additional data. For example, if you were to collect data for the variables Obesity and PreviousHDL so that these two variables are no longer unmeasured, then you could block the two noncausal paths in Output 37.3.2. The following code demonstrates this:

```
proc causalgraph compact;
   model "Thor12"
      AntiHypertensiveUse ==> CurrentBP,
      Creatinine ==> AntiHypertensiveUse CurrentBP,
      CurrentBP ==> CVD,
      CurrentHDL ==> CVD,
      Diabetes ==> AntiHypertensiveUse Creatinine,
      Ethnicity ==> Nutrition Smoking,
      Gender ==> Nutrition Urate,
      Gout ==> CVD,
      HbA1c ==> Diabetes,
      MedicationPropensity ==> AntiHypertensiveUse StatinUse,
      Nutrition ==> PreviousHDL Urate Obesity,
      Obesity ==> PreviousBP HbA1c,
      PreviousBP ==> AntiHypertensiveUse,
      PreviousHDL ==> StatinUse,
      Smoking ==> CVD,
      StatinUse ==> CurrentHDL,
      Urate ==> PreviousBP Creatinine CVD Gout;
   identify Urate ==> CVD;
   unmeasured Nutrition PreviousBP MedicationPropensity;
   testid Gender HbA1c Ethnicity Smoking
      CurrentHDL PreviousHDL Obesity;
run;
```

As shown in Output 37.3.4, the proposed adjustment set is marked "Yes" in the Valid column, so it is sufficient to estimate the causal effect of Urate on CVD. For more information about possible approaches to obtain identifiability, see the section "Dealing with Nonidentified Causal Effects" on page 2388.

Output 37.3.4 Adjustment Set Test

Covariate Adjustment Test: Test1

Causal Effect of Urate on CVD

				Covariates						
Model	**Size**	**Valid**	**Minimal**	**CurrentHDL**	**Ethnicity**	**Gender**	**HbA1c**	**Obesity**	**PreviousHDL**	**Smoking**
Thor12	7	Yes	No	*	*	*	*	*	*	*

Example 37.4: Finding Adjustment Sets Common to Multiple Models

(View the complete code for this example (cagrex4.sas) in the example repository.)

This example illustrates how to use the CAUSALGRAPH procedure to find adjustment sets that you can use to estimate a causal effect when you are uncertain of the exact structure of the causal model or when alternative causal models might be possible for the data. For instance, you might not be sure whether to include an edge in a causal model, what direction an edge should have, or what covariates to include in the model. You can use multiple MODEL statements in the procedure to specify every plausible causal model and then use the COMMON option in the PROC CAUSALGRAPH statement to search for adjustment sets that are valid for all the models. When the set of candidate models share a similar structure, you can use the REFMODEL= option in the PROC CAUSALGRAPH statement to specify the models concisely.

For a single causal model, if an adjustment set exists, then you can use this adjustment set to estimate the causal effect. If multiple causal models share a common adjustment set, then you can use this set to estimate the causal effect, no matter which causal model reflects the true data generating process. For more information about identifying causal effects in a causal model, see the section "Causal Effect Identification" on page 2382.

The models in this example are derived from a larger model that was developed by Thornley et al. (2013). These models examine the relationship between an individual's serum urate and risk of cardiovascular disease. See Example 37.3 for a summary of the variables in the models. Figure 37.8 depicts two plausible causal assumptions for the effect of the variable StatinUse on the variable CVD. One model assumes that all the causal influence of StatinUse on CVD is strictly mediated by the variable CurrentHDL. The other model includes a direct effect of StatinUse on CVD, in addition to the mediated effect. These two models differ by a single edge, which is indicated by the broken arrow in Figure 37.8. In both models in this example, the variables Nutrition and PreviousHDL are assumed to be unmeasured and thus are represented by nodes with broken outlines.

Figure 37.8 Two Possible Causal Models of the Effect of Serum Urate on Risk of Cardiovascular Disease

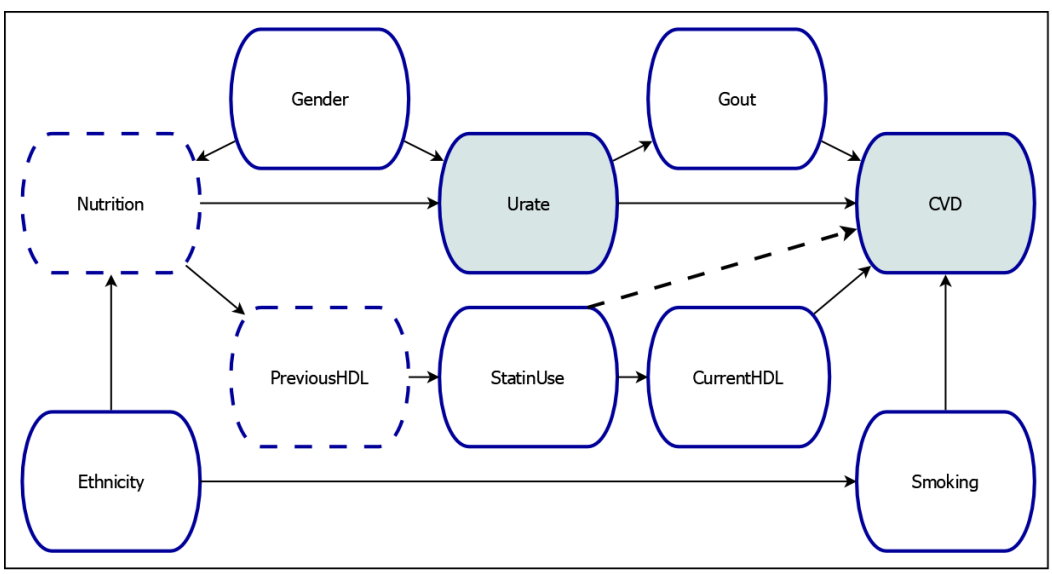

The following statements invoke PROC CAUSALGRAPH to construct adjustment sets for the two models in

Figure 37.8:

```
proc causalgraph common;
   model "Thor12SimpleHDL"
      Ethnicity ==> Nutrition Smoking,
      Gender ==> Nutrition Urate,
      Gout ==> CVD,
      Nutrition ==> PreviousHDL Urate,
      CurrentHDL ==> CVD,
      PreviousHDL ==> StatinUse,
      Smoking ==> CVD,
      StatinUse ==> CurrentHDL,
      Urate ==> CVD Gout;
   model "Thor12AltHDL"
      StatinUse ==> CVD / refmodel="Thor12SimpleHDL";
   identify Urate ==> CVD;
   unmeasured Nutrition PreviousHDL;
run;
```

When you analyze multiple causal models, you must use a separate MODEL statement for each causal model. Each MODEL statement must begin with a quoted string that provides a unique name for the model. The remainder of the MODEL statement specifies all the variables and edges in the model. The model **Thor12SimpleHDL** assumes that the causal association between StatinUse and CVD is strictly mediated by the variable CurrentHDL. The model **Thor12AltHDL** is specified as a single edge from StatinUse to CVD, together with the REFMODEL= option. This means that the model **Thor12AltHDL** contains all the edges in the model **Thor12SimpleHDL**, as well as the additional edge that indicates a direct effect between StatinUse and CVD.

In the IDENTIFY statement, you specify the causal effect of interest. You can use this statement to specify one or more treatment variables and one or more outcome variables. The treatment and outcome variables are separated by a single right arrow, **==>**. In this example, you are interested in testing the identification of the causal effect of the variable Urate on the variable CVD.

In the UNMEASURED statement, you specify variables that are not observed and thus cannot be included in an adjustment set. In this example, the variables Nutrition and PreviousHDL are specified as unmeasured.

The COMMON option in the PROC CAUSALGRAPH statement requests a list of all adjustment sets that are common to all the models that you specify in the MODEL statements. In this example, the procedure produces three tables of adjustment sets: one for each model, and one that displays common adjustments. You can use the COMMON(ONLY) option to produce only the table of common adjustments without also producing separate tables of adjustment sets for each model.

The adjustment sets for the models **Thor12SimpleHDL** and **Thor12AltHDL** are shown in Output 37.4.1 and Output 37.4.2, respectively. Although both models contain the same number of variables, **Thor12SimpleHDL** has one less edge and a larger number of valid adjustment sets. This is usually, but not always, expected. Adding edges to a model does not generally improve the identifiability of a causal effect, and it is often harmful to identification. For more information about obtaining identifiability in a model, see the section "Dealing with Nonidentified Causal Effects" on page 2388.

Output 37.4.1 Adjustment Sets for `Thor12SimpleHDL`

The CAUSALGRAPH Procedure

Covariate Adjustment Sets for Thor12SimpleHDL
Causal Effect of Urate on CVD

			Covariates					
	Size	Minimal	CurrentHDL	Ethnicity	Gender	Gout	Smoking	StatinUse
1	2	Yes	*	*				
2	2	Yes	*				*	
3	2	Yes		*				*
4	2	Yes					*	*
5	3	No	*	*	*			
6	3	No	*	*			*	
7	3	No	*	*				*
8	3	No	*		*		*	
9	3	No	*				*	*
10	3	No		*	*			*
11	3	No		*			*	*
12	3	No			*		*	*
13	4	No	*	*	*		*	
14	4	No	*	*	*			*
15	4	No	*	*			*	*
16	4	No	*		*		*	*
17	4	No		*	*		*	*
18	5	No	*	*	*		*	*

Output 37.4.2 Adjustment Sets for `Thor12AltHDL`

The CAUSALGRAPH Procedure

Covariate Adjustment Sets for Thor12AltHDL
Causal Effect of Urate on CVD

			Covariates					
	Size	Minimal	CurrentHDL	Ethnicity	Gender	Gout	Smoking	StatinUse
1	2	Yes		*				*
2	2	Yes					*	*
3	3	No	*	*				*
4	3	No	*				*	*
5	3	No		*	*			*
6	3	No		*			*	*
7	3	No			*		*	*
8	4	No	*	*	*			*
9	4	No	*	*			*	*
10	4	No	*		*		*	*
11	4	No		*	*		*	*
12	5	No	*	*	*		*	*

Output 37.4.3 shows the adjustment sets that are common to both models. Coincidentally, the adjustment sets that are valid in both models are the same as the adjustment sets that are valid in the model that has the extra

edge. Because at least one such common adjustment set exists, this analysis suggests that it is possible to estimate the causal effect of Urate on CVD by using any adjustment set from Output 37.4.3, regardless of whether there is a direct effect of StatinUse on CVD.

Output 37.4.3 Common Adjustment Sets

The CAUSALGRAPH Procedure

Covariate Adjustment Sets Common to All Models								
Causal Effect of Urate on CVD								
			Covariates					
	Size	Minimal	CurrentHDL	Ethnicity	Gender	Gout	Smoking	StatinUse
1	2	Yes		*				*
2	2	Yes					*	*
3	3	No	*	*				*
4	3	No	*				*	*
5	3	No		*	*			*
6	3	No		*			*	*
7	3	No			*		*	*
8	4	No	*	*	*			*
9	4	No	*	*			*	*
10	4	No	*		*		*	*
11	4	No		*	*		*	*
12	5	No	*	*	*		*	*

Example 37.5: Identifying a Causal Effect by Using Instrumental Variables

(View the complete code for this example (cagrex5.sas) in the example repository.)

This example illustrates how you can use the CAUSALGRAPH procedure to find an instrumental variable in a causal model. By using an instrumental variable, you can identify a causal effect even when there is unobserved confounding between a treatment variable and an outcome variable, which is a situation where the adjustment criterion might fail.

In Example 37.2, a causal model, adapted from Timmermann et al. (2017), is used to examine the relationship between maternal exposure to persistent perfluoroalkyl substances (PFAS) and breastfeeding duration among residents of the Faroe Islands. That example shows that you cannot construct an adjustment set to estimate the causal effect of the treatment variable PFAS on the outcome variable Duration. This is because there is confounding bias between the treatment and outcome variables that results from the unobserved variables HealthBehavior and PrevBF.

In many cases where there is unmeasured confounding, you can still estimate the causal effect if you are willing to assume that certain edges in the causal model have a particular parametric form. Then it might be possible to use an instrumental variable (Angrist, Imbens, and Rubin 1996; Imbens 2014). The following statements invoke PROC CAUSALGRAPH to list possible instrumental variables that you can use to estimate the causal effect:

```
proc causalgraph method=iv;
   model "Timm17HealthBehavior"
      Age ==> Parity PFAS Education,
      Parity ==> PrevBF Duration PFAS,
      PrevBF ==> PFAS Duration,
      PFAS ==> Duration,
      Education ==> Duration HealthBehavior Employment,
      HealthBehavior ==> PFAS Duration BMI Alcohol Smoking,
      Employment ==> HealthBehavior Duration,
      BMI Alcohol Smoking ==> Duration;
   identify PFAS ==> Duration;
   unmeasured PrevBF HealthBehavior;
run;
```

In the MODEL statement, you specify the same causal model as in Example 37.2. In the IDENTIFY statement, you specify that the causal effect of the variable PFAS on the variable Duration is of interest. By specifying the METHOD=IV option in the PROC CAUSALGRAPH statement, you use an instrumental variable (or a conditional instrumental variable) to identify the causal effect. The UNMEASURED statement specifies that the variables PrevBF and HealthBehavior are not observed and thus cannot be included as an instrument or in any conditional set.

As shown in the table in Output 37.5.1, there is one variable that you can use as an instrument to identify the causal effect of PFAS on Duration. The variable Age is a conditional instrument. That is, Age becomes an instrument after its association with the outcome variable Duration is blocked by conditioning on the five variables shown in Output 37.5.1. If an instrumental variable is a classical instrument (that is, you can use the variable as an instrument without conditioning on any other variables), then the procedure prints an empty conditioning set for that variable.

Output 37.5.1 Instrumental Variables for the Causal Model

Instrumental Variables for Timm17HealthBehavior
Causal Effect of PFAS on Duration

		Conditionals					
	Instrument	Age	Alcohol	BMI	Education	Employment	Parity	Smoking
1	Age		*	*	*	*	*	

The conditional sets that are produced in constructing an instrument might not be minimal. For instance, the following test shows that you can use Age as a conditional instrument if you adjustment for only the variables Education and Parity:

```
proc causalgraph method=iv;
   model "Timm17HealthBehavior"
      Age ==> Parity PFAS Education,
      Parity ==> PrevBF Duration PFAS,
      PrevBF ==> PFAS Duration,
      PFAS ==> Duration,
      Education ==> Duration HealthBehavior Employment,
      HealthBehavior ==> PFAS Duration BMI Alcohol Smoking,
      Employment ==> HealthBehavior Duration,
      BMI Alcohol Smoking ==> Duration;
   identify PFAS ==> Duration;
```

```
    unmeasured PrevBF HealthBehavior;
    testid "Minimal CIV" Age / conditional = (Education Parity);
  run;
```

The model specification, the causal effect of interest, and the identification criterion are the same as those in the previous CAUSALGRAPH analysis in this example. What is new here is the inclusion of the TESTID statement. You want to investigate whether Age can serve as a conditional instrumental variable for the causal effect, where the conditioning variables are Education and Parity. This set of conditional variables is a proper subset of the proposed conditional variables in the previous analysis. As shown in Figure 37.5.2, the value for the Valid column is "Yes," providing an affirmative answer to this investigation.

Output 37.5.2 Instrumental Variable Test Output

Instrumental Variable Test: Minimal CIV
Causal Effect of PFAS on Duration

			Conditionals						
Model	**Instrument**	**Valid**	**Age**	**Alcohol**	**BMI**	**Education**	**Employment**	**Parity**	**Smoking**
Timm17HealthBehavior	Age	Yes				*		*	

Example 37.6: Distinguishing Models with Data

(View the complete code for this example (cagrex6.sas) in the example repository.)

This example illustrates how you can use the CAUSALGRAPH procedure to examine the statistical implications of the assumptions that are encoded in a causal model. When you have multiple plausible models that might describe the same data set, you can use these statistical implications to decide which model best represents the data generating process.

Two plausible causal models for describing an individual's serum urate and risk of cardiovascular disease are shown in Figure 37.9. The model are derived from a larger model that was developed by Thornley et al. (2013). See Example 37.3 for a summary of the variables in the models. In one of the two models, blood pressure and antihypertensive medication use are assumed to mediate the effect of the variable Urate on the variable CVD. In the other model, the causal direction is reversed, and the variable AntiHypertensiveUse is assumed to exert a causal effect on Urate. Differences between the two models are highlighted in red. The treatment and outcome variables are shaded in both models. For this example, the variable Nutrition corresponds to a latent construct and thus is not measured or observed. It is also assumed that the variable PreviousBP is not measured.

Figure 37.9 Two Possible Causal Models of the Effect of Serum Urate on Risk of Cardiovascular Disease

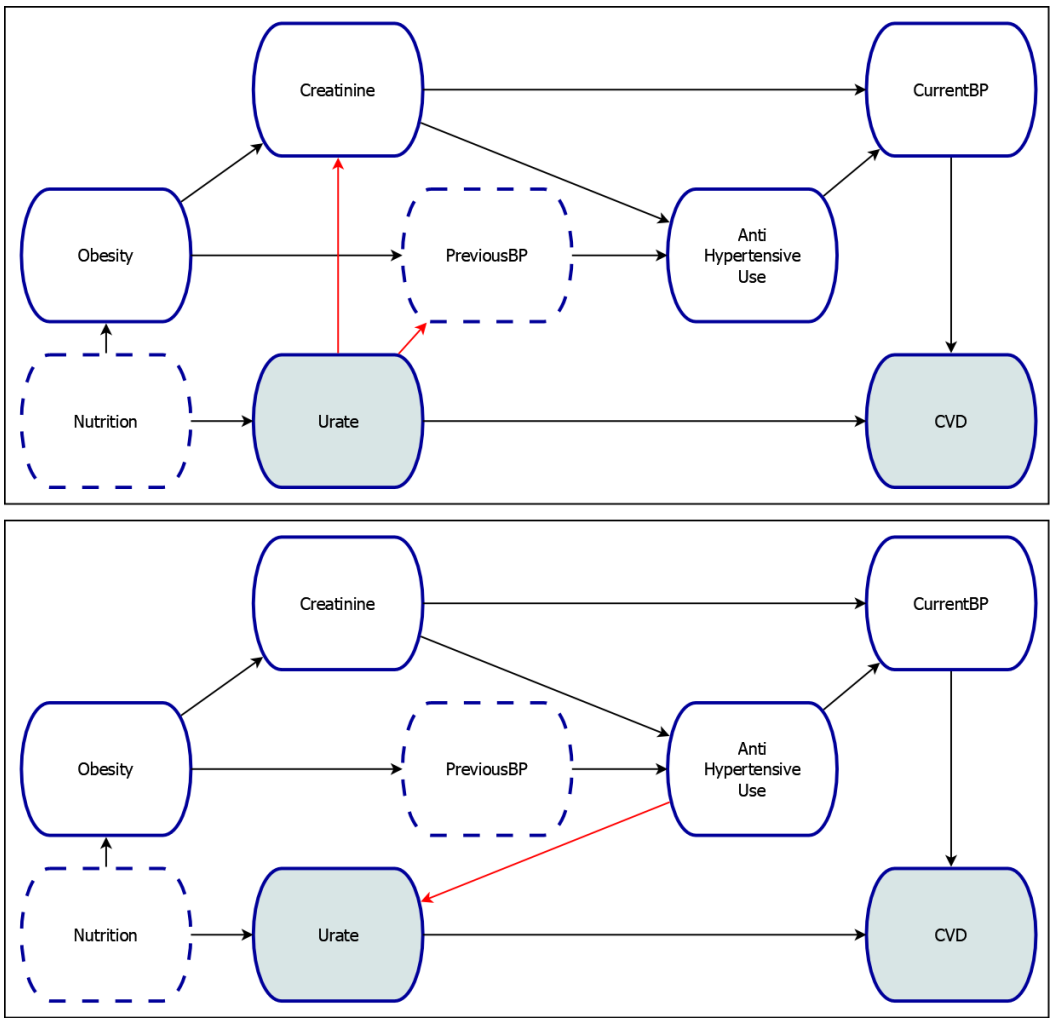

When you have multiple possible causal models, sometimes there is a common adjustment set that is valid for all models. In this case, you can use the adjustment technique to estimate the causal effect from the data, no matter which model best represents the true data generating process. See Example 37.4 for an illustration. The following statements invoke PROC CAUSALGRAPH to construct a common adjustment set:

```
proc causalgraph common(only) outmodel=Thor12;
   model "Thor12Base"
      AntiHypertensiveUse ==> CurrentBP,
      Creatinine ==> AntiHypertensiveUse CurrentBP,
      Nutrition ==> Urate Obesity,
      Obesity ==> PreviousBP Creatinine,
      CurrentBP ==> CVD,
      PreviousBP ==> AntiHypertensiveUse,
      Urate ==> CVD / noanalysis;
   model "Thor12SimpleBP"
      Urate ==> PreviousBP Creatinine / refmodel="Thor12Base";
   model "Thor12AltBP"
      AntiHypertensiveUse ==> Urate / refmodel="Thor12Base";
   identify Urate ==> CVD;
```

```
   unmeasured Nutrition PreviousBP;
run;
```

The OUTMODEL= option in the PROC CAUSALGRAPH statement creates a data set that contains the models to be analyzed. When you analyze multiple causal models, you must use a separate MODEL statement for each causal model. Each MODEL statement must begin with a quoted string that provides a unique name for the model. In this example, you first specify the nodes and edges that are common to both models to be analyzed. You use the NOANALYSIS option to indicate that this model specification is not intended for identification analysis. Next, you use two additional MODEL statements, labeled **Thor12SimpleBP** and **Thor12AltBP**, to specify the models to be analyzed. In both cases, you use the REFMODEL= option to indicate the reference model.

In the IDENTIFY statement, you specify the causal effect of interest. In this example, you are investigating the causal effect of Urate on CVD.

The COMMON(ONLY) option in the PROC CAUSALGRAPH statement lists all adjustment sets that are common to all the models that you specify using MODEL statements. In this example, the procedure produces output only for the common adjustments. You can use the COMMON option without the (ONLY) suboption to produce separate adjustment sets for each model, in addition to the common adjustment sets.

The note in Output 37.6.1 shows that, in this example, it is not possible to find a common adjustment set. Thus, either you must find separate adjustment sets, then estimate the causal effect separately by using each model, or you must determine which model best represents the data generating process. The IMAP option provides analyses of model properties that might be useful for the latter action.

Output 37.6.1 Common Adjustment Note

NOTE: There are no adjustment sets common to all models that satisfy the specified criteria.

The causal assumptions that are encoded in a graphical model have implications for the statistical properties of the data generating process. You can test the implied statistical properties against the available data (subject to the usual limitations of sampling error and hypothesis testing). If the property does not hold in the data, then you should consider revising or discarding the model. For more information about the statistical implications of a graphical model, see the section "Statistical Properties of Causal Models" on page 2381. If you have multiple models, then you can compare the statistical implications of the models to find a property that is implied in one model but not the others. You can then test this property in the data and use the corresponding test results to determine which model best represents the true data generating process.

The following statements invoke the procedure to enumerate these statistical properties for both models in this example:

```
proc causalgraph inmodel=Thor12 nolist imap=global;
   identify Urate ==> CVD;
   unmeasured Nutrition PreviousBP;
   ods output Imap = Thor12Indep;
run;
```

The INMODEL= statement loads the models that you saved by specifying the OUTMODEL= option in the previous invocation of the procedure. The NOLIST turns off the search for valid adjustment sets in this analysis.

The IMAP=GLOBAL option produces a table of global Markov properties for every model in the analysis. A global Markov property consists of two variables that are statistically independent, conditional on another

(possibly empty; in this case, the independence is unconditional) set of variables. For more information about the global Markov properties that are associated with a model, see the section "Statistical Properties of Causal Models" on page 2381. In this example, the table of conditional independence properties that the procedure produces is saved to a data set by the ODS OUTPUT statement. This table indicates which model gives rise to each property.

If every variable in a Markov property is observed, then you can perform statistical tests (for example, you can test for zero partial correlation) by using the available data to see whether the property can be falsified. Independence properties that involve one or more unmeasured variables cannot be tested.

The following code prints the first 10 observed conditional independence properties for each model. This example focuses on the first 10 statements for the sake of brevity. In practice, you perform the following analysis on the complete sets of independence properties.

```
proc print data=Thor12Indep(obs=10);
   var Set1 Set2 CondSet;
   where Model="Thor12SimpleBP" and Observable = 1;
run;

proc print data=Thor12Indep(obs=10);
   var Set1 Set2 CondSet;
   where Model="Thor12AltBP" and Observable = 1;
run;
```

The first 10 observed conditional independence properties for the two models are displayed in Output 37.6.2 and Output 37.6.3. You can now compare the two models. For both models, there are four conditioning sets such that the variable AntiHypertensiveUse is conditionally independent of the variable CVD, and these four conditioning sets are the same for both models. Thus, you cannot distinguish between the two models by testing for conditional independence between AntiHypertensiveUse and CVD. Next, you compare the conditional independence properties for the variables Creatinine and CVD. There are four such properties for the model Thor12SimpleBP, but there are five such properties for the model **Thor12AltBP**. The model **Thor12AltBP** encodes the statistical implication that Creatinine and CVD are independent conditional on the set (AntiHypertensiveUse, CurrentBP, Obesity), but this implication is not encoded in **Thor12SimpleBP**.

This means that if you were to find a nonzero partial correlation between Creatinine and CVD (after partialing out AntiHypertensiveUse, CurrentBP, and Obesity), you would have evidence to reject **Thor12AltBP**. You can continue this analysis for every independence property that is unique to one of the two models. The best model is the one whose conditional independence properties most closely match the zero partial correlations in the available data.

Output 37.6.2 Conditional Independence Properties of **Thor12SimpleBP**

Obs	Set1	Set2	CondSet
1	AntiHypertensiveUse	CVD	CurrentBP Urate
3	AntiHypertensiveUse	CVD	CurrentBP Obesity Urate
9	AntiHypertensiveUse	CVD	Creatinine CurrentBP Urate
12	AntiHypertensiveUse	CVD	Creatinine CurrentBP Obesity Urate
73	Creatinine	CVD	CurrentBP Urate
75	Creatinine	CVD	CurrentBP Obesity Urate
81	Creatinine	CVD	AntiHypertensiveUse CurrentBP Urate
83	Creatinine	CVD	AntiHypertensiveUse CurrentBP Obesity Urate
141	CurrentBP	Obesity	AntiHypertensiveUse Creatinine
142	CurrentBP	Obesity	AntiHypertensiveUse Creatinine Urate

Output 37.6.3 Conditional Independence Properties of `Thor12AltBP`

Obs	Set1	Set2	CondSet
263	AntiHypertensiveUse	CVD	CurrentBP Urate
265	AntiHypertensiveUse	CVD	CurrentBP Obesity Urate
271	AntiHypertensiveUse	CVD	Creatinine CurrentBP Urate
273	AntiHypertensiveUse	CVD	Creatinine CurrentBP Obesity Urate
299	Creatinine	CVD	CurrentBP Urate
301	Creatinine	CVD	CurrentBP Obesity Urate
307	Creatinine	CVD	AntiHypertensiveUse CurrentBP Urate
309	Creatinine	CVD	AntiHypertensiveUse CurrentBP Obesity
310	Creatinine	CVD	AntiHypertensiveUse CurrentBP Obesity Urate
341	Creatinine	Urate	AntiHypertensiveUse Obesity

Example 37.7: Applying an Adjustment Set to Estimate a Causal Effect from Data

(View the complete code for this example (cagrex7.sas) in the example repository.)

This example illustrates how you can use the CAUSALGRAPH procedure to estimate the magnitude of a causal effect that has a valid causal interpretation. To compute such an estimate from a data set, you can use the following general approach:

1. Carefully consider the data generating process, and create a list of causal assumptions that accurately represents that process. Encode these assumptions in a graphical causal model. For more information about causal assumptions and graphical models, see the section "Causal Graph Theory" on page 2378.
2. Use this graphical model to find a valid identification strategy, such as an adjustment set.
3. Use the identification results to construct an estimator, such as the stratification estimator.

In most practical situations, the true data generating process is not known exactly. In such situations, you must define a causal model that is assumed to represent the data generating process. To construct this causal model, you might rely on expert opinion, established scientific theory, prior experience, or any other source of substantive knowledge. This example uses simulated data, so that the data generating process is treated as known and the impact of identification and adjustment on causal effect estimation can be illustrated. For more information about identification strategies and adjustment sets, see the section "Identification by Adjustment" on page 2382.

A causal model that relates an individual's serum urate to the risk of cardiovascular disease is shown in Figure 37.10. This model is derived from a larger model that was developed by Thornley et al. (2013). See Example 37.3 for a summary of the variables in the figure. The treatment and outcome variables are shaded in the model. For this example, the variable Nutrition corresponds to a latent construct and thus is not measured or observed. It is also assumed that the variable PreviousHDL is not measured.

Figure 37.10 Causal Model of the Effect of Serum Urate on Risk of Cardiovascular Disease

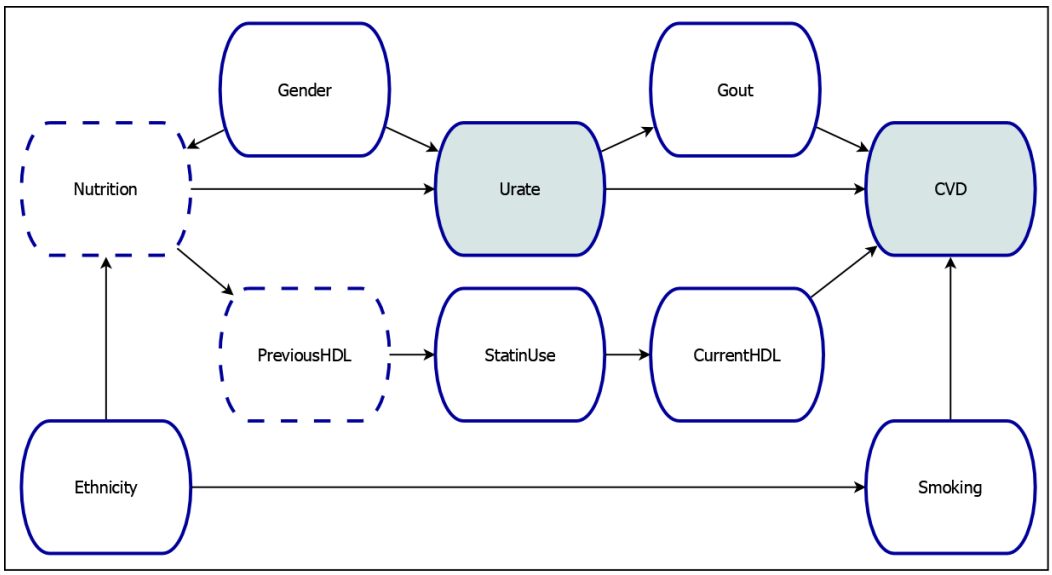

The following DATA step creates a simulated data set that is consistent with the model in Figure 37.10. Thus, it also defines the true data generating process.

```
data CVDdata;
   drop ii Nutrition PreviousHDL;
   call streaminit(1000);
   array EthProb[6] _temporary_ (0.60, 0.18, 0.13, 0.05, 0.01, 0.03);
   array SmokeRates[6] _temporary_ (0.17, 0.10, 0.17, 0.07, 0.22, 0.15);
   array EthNut[6] _temporary_ (0.20, 0.18, 0.08, 0.03, 0.11, 0.04);
   do ii = 1 to 79000;
      Gender = rand("Bernoulli" , .5);
      Ethnicity = rand("Table" , of EthProb[*]);
      Smoking = rand("Bernoulli", SmokeRates[Ethnicity]);
      Nutrition = 0.5 - Gender + 10.0*rand("Normal", 0, EthNut[Ethnicity]);
      PreviousHDL = 55 + 4.0*Nutrition;
      if      PreviousHDL<40 then StatinUse = rand("Bernoulli", 0.90);
      else if PreviousHDL<60 then StatinUse = rand("Bernoulli", 0.65);
      else                        StatinUse = rand("Bernoulli", 0.05);
      CurrentHDL = 55 + rand("Normal",0.0,7) + StatinUse*rand("Normal",4.0,0.5);
      Urate = 6.0 + 0.4*Nutrition + 1.5*Gender;
      Gout = rand("Bernoulli", logistic(-8.0 + 0.90*Urate));
      CVD = rand("Bernoulli", logistic(-1.2 - 0.04*CurrentHDL + 0.2*Gout +
         0.65*Smoking + 0.1*Urate));
      output;
   end;
run;
```

The first 10 lines of the simulated data set are shown in Output 37.7.1.

Output 37.7.1 First 10 Lines of the Simulated Data Set

Obs	Gender	Ethnicity	Smoking	StatinUse	CurrentHDL	Urate	Gout	CVD
1	Female	Hispanic	No	0	45.9241	6.41547	1	0
2	Male	Hispanic	No	1	66.9528	7.43840	0	0
3	Female	NativeAmer	No	1	54.5367	6.42775	0	0
4	Female	Hispanic	No	1	67.7733	5.32636	0	0
5	Female	WhiteNonHisp	No	0	64.2067	6.67543	0	0
6	Male	WhiteNonHisp	No	1	53.0462	7.22069	1	0
7	Female	AfricanAmer	Yes	1	49.4850	5.15609	0	0
8	Male	Hispanic	Yes	0	57.8065	6.47476	0	0
9	Female	WhiteNonHisp	No	0	57.7600	6.52005	0	1
10	Male	AfricanAmer	No	1	71.0460	7.30762	0	0

The following code uses the simulated data to create a table of summary statistics for the variable Urate:

```
proc means data=CVDdata;
   var Urate;
   ods output Summary=SampleMeansOutput;
run;
```

The summary statistics are shown in Output 37.7.2. You use the ODS OUTPUT statement to store the summary statistics for Urate in an output data set. You use this information later in the analysis to define the treatment and control levels for the causal effect of interest.

Output 37.7.2 Summary Statistics for Urate

Analysis Variable : Urate				
N	Mean	Std Dev	Minimum	Maximum
79000	6.7460150	0.8936312	3.0665877	10.3763707

In this example, the treatment or exposure variable Urate is continuous. Moreover, the effect of this variable on the mediator variable Gout and the outcome variable (CVD) is nonlinear. Because there are no natural treatment and control levels for Urate, you must somehow define the causal effect of interest. A common causal effect measure is the average treatment effect (ATE) or the expected risk difference, which is the difference in the expected potential outcome values between well-defined treatment and control conditions or levels. For more information about the definitions of causal effect measures in the potential outcome framework, see the section "Causal Effects: Definitions, Assumptions, and Identification" on page 2533 in Chapter 39, "The CAUSALTRT Procedure."

In this example, you consider the causal effect of interest to be the expected risk difference in CVD that is associated with a change in Urate from a control condition to a treatment condition. Two possibilities for defining the control and treatment conditions are considered here. In this way, you can explore how the magnitude of the causal effect depends on the values of the treatment variable that are being considered.

First, you consider the causal effect of a unit change in Urate, centered around the population mean. Then, in the potential outcomes notation, the causal effect of interest is the expected risk difference

$$\text{UnitEff} = E[\text{CVD}(\text{Urate} = \mu + 0.5)] - E[\text{CVD}(\text{Urate} = \mu - 0.5)]$$

where μ is the population mean of Urate. In this causal effect definition, the control condition is defined as a

half unit below the population mean of Urate, and the treatment condition is defined as a half unit above the population mean of Urate.

Second, you consider the causal effect of a change of one standard deviation in Urate, also centered around the population mean. The causal effect is now defined as the expected risk difference

$$\text{StdEff} = E[\text{CVD}(\text{Urate} = \mu + 0.5\sigma)] - E[\text{CVD}(\text{Urate} = \mu - 0.5\sigma)]$$

where σ is the population standard deviation of Urate.

For demonstration purposes, the two population causal effects are computed by generating a large number of potential outcomes (1,000,000,000 replications) according to the true data generating process mentioned earlier. By this method, the population effect, UnitEff, is found to be 0.0076, and the standardized population effect, StdEff, is found to be 0.0068. These values are the target causal effects that you are estimating from the random sample. Now the most interesting questions are whether and how the CAUSALGRAPH procedure can help find a statistical strategy that provides accurate estimates of these target causal effects.

Given the causal model for the data, you can use the procedure to analyze the identifiability of the causal effect of Urate on CVD. The following code uses the procedure to list valid adjustment sets that can be used to identify this causal effect. For brevity, you can use the MAXSIZE=2 option to construct only those adjustment sets that have no more than two elements.

```
proc causalgraph maxsize=2;
   model "Thor12SimpleHDL"
      Ethnicity ==> Nutrition Smoking,
      Gender ==> Nutrition Urate,
      Gout ==> CVD,
      Nutrition ==> PreviousHDL Urate,
      CurrentHDL ==> CVD,
      PreviousHDL ==> StatinUse,
      Smoking ==> CVD,
      StatinUse ==> CurrentHDL,
      Urate ==> CVD Gout;
   identify Urate ==> CVD;
   unmeasured Nutrition PreviousHDL;
run;
```

In the MODEL statement, you specify the causal model to be analyzed. The quoted string in the statement labels the model. The remainder of the MODEL statement specifies all the variables and edges in the model. These variables and edges reflect the data generating process shown in Figure 37.10.

In the IDENTIFY statement, you specify that the causal effect of interest is the effect of Urate on CVD. In the UNMEASURED statement, you specify that the variables Nutrition and PreviousHDL are not measured or observed.

The list of adjustment sets that the procedure produces is shown in Output 37.7.3. Notice that the empty set does not appear in this list. This means that the marginal association between Urate and CVD cannot be used to estimate a causal effect that has a valid causal interpretation. Rather, you must use an alternative estimation strategy, such as estimation by adjustment that uses one of the adjustment sets in Output 37.7.3. As is shown later in this example, the failure to perform such an adjustment results in biased estimation of the causal effects.

Output 37.7.3 Possible Adjustment Sets for the Model in Figure 37.10

Covariate Adjustment Sets for Thor12SimpleHDL								
Causal Effect of Urate on CVD								
			Covariates					
	Size	Minimal	CurrentHDL	Ethnicity	Gender	Gout	Smoking	StatinUse
1	2	Yes	*	*				
2	2	Yes	*				*	
3	2	Yes		*				*
4	2	Yes					*	*

You can use any of the adjustments sets in Output 37.7.3 to obtain an estimate for the effect of Urate on CVD that has a valid causal interpretation. In particular, the set {Smoking, StatinUse} is a valid adjustment set. This set also has the useful property that both variables in the set are binary classification variables. Thus one possible way to estimate the causal effect is to stratify the analysis by the levels of these variables.

As mentioned earlier, two causal effects are being estimated. One is the unstandardized unit effect of Urate on CVD, denoted as UnitEff. The other is the standardized unit effect of Urate on CVD, denoted as StdEff. Both of these causal effects are defined in terms of the differences in expected CVD potential outcome values. These potential outcomes are evaluated at some Urate treatment and control levels that are defined in terms of population parameters. Because these population parameters and hence the treatment and control levels are not known, you need to estimate them from the sample. The following code computes two sets of sample values for the treatment and control levels of Urate from the table of summary statistics that you created earlier in this example. These computed values are stored in the data set ScoreData that you will use to estimate the two causal effects of interest.

```
data _null_;
   set SampleMeansOutput;
   call symputx("UrateMean",Urate_Mean);
   call symputx("UrateStd", Urate_StdDev);
   call symputx("UrateUnit1", Urate_Mean + 0.5);
   call symputx("UrateUnit0", Urate_Mean - 0.5);
   call symputx("UrateStd1", Urate_Mean + 0.5*Urate_StdDev);
   call symputx("UrateStd0", Urate_Mean - 0.5*Urate_StdDev);
run;

data ScoreData;
   set SampleMeansOutput;
   keep Urate Test;

   Test  = "UnitTreat  ";
   Urate = &UrateUnit1;
   output;

   Test  = "UnitControl";
   Urate = &UrateUnit0;
   output;

   Test  = "StdTreat   ";
   Urate = &UrateStd1;
   output;
```

```
   Test  = "StdControl ";
   Urate = &UrateStd0;
   output;
run;
```

Now, the following code performs logistic regression analyses that are stratified by levels of the two adjustment variables that are suggested by the results of PROC CAUSALGRAPH:

```
proc sort data=CVDdata;
   by Smoking StatinUse;
run;
proc logistic data=CVDdata noprint;
   by Smoking StatinUse;
   model CVD(event='1') = Urate;
   score data=ScoreData out=ProbStrat;
run;
```

The MODEL statement specifies the outcome variable CVD and the treatment variable Urate. The effect of Urate on CVD is estimated within each of the four strata separately. However, to estimate the aforementioned population causal effects of interest, you need to apply the logistic regression results to estimate the expected CVD values under the specific Urate treatment and control levels of interest. You can accomplish this by using the SCORE statement.

The SCORE statement estimates the probability of a CVD event (that is, the probability that CVD=1), which is the expected value of CVD, for each value of Urate that is specified in the data set ScoreData. In this example, these values of Urate correspond to the treatment and control levels for defining the causal effects of interest. Hence, the SCORE statement estimates the expected CVD values under the required treatment and control levels in each stratum. The OUT= option in the SCORE statement saves the expected CVD values in the data set ProbStrat. These expected CVD values are shown in the column P_1 in Output 37.7.4.

Output 37.7.4 Posterior Probabilities for Each Strata

Obs	Smoking	StatinUse	Test	Urate	P_1
1	0	0	UnitTreat	7.24602	0.06929
2	0	0	UnitControl	6.24602	0.05994
3	0	0	StdTreat	7.19283	0.06876
4	0	0	StdControl	6.29920	0.06041
5	0	1	UnitTreat	7.24602	0.05829
6	0	1	UnitControl	6.24602	0.05489
7	0	1	StdTreat	7.19283	0.05811
8	0	1	StdControl	6.29920	0.05506
9	1	0	UnitTreat	7.24602	0.12451
10	1	0	UnitControl	6.24602	0.10476
11	1	0	StdTreat	7.19283	0.12338
12	1	0	StdControl	6.29920	0.10574
13	1	1	UnitTreat	7.24602	0.11226
14	1	1	UnitControl	6.24602	0.09918
15	1	1	StdTreat	7.19283	0.11153
16	1	1	StdControl	6.29920	0.09984

The two binary adjustment variables result in four strata for analysis. Within each stratum, you can compute the unstandardized unit effect by the difference in P_1 between UnitTreat and UnitControl, and you can

compute the standardized effect by the difference in P_1 between StdTreat and StdControl. However, none of these effects within strata are the causal effect estimates themselves. The estimate of the causal effect UnitEff must be computed by using the weighted average of the difference in P_1 between UnitTreat and UnitControl in the strata, where the weights are the sample sizes of the strata. Similarly, the estimate of the causal effect StdEff must be computed by using the weighted average of the difference in P_1 between StdTreat and StdControl in the strata. These estimates of the causal effects are shown in the column Stratified Estimation in Output 37.7.6.

As discussed previously, if you attempt to estimate the effect of Urate on CVD by using the marginal association between the two variables (that is, no adjustment), confounding covariates bias the estimation results. Strictly for the purpose of demonstrating such a biased result, the following PROC LOGISTIC code performs a logistic regression that is not stratified by any covariate:

```
proc logistic data=CVDdata noprint;
   model CVD(event='1') = Urate;
   score data=ScoreData out=ProbNaive;
run;
```

Just as before, the SCORE statement in PROC LOGISTIC estimates the expected CVD values under the required Urate treatment and control levels. The corresponding estimates of the expected CVD values are shown in Output 37.7.5. As in the stratified estimation, the target causal effects are estimated by computing the related differences in P_1, except that in this case no weighted average is computed for the differences.

Output 37.7.5 Unadjusted Posterior Probabilities

Obs	Test	Urate	P_1
1	UnitTreat	7.24602	0.073217
2	UnitControl	6.24602	0.064062
3	StdTreat	7.19283	0.072702
4	StdControl	6.29920	0.064521

Now, you have two sets of estimation results. One set of results is computed by using a stratified estimator that is based on an adjustment strategy. The other set of results is computed by using the naive marginal association between the treatment and outcome variables. The estimates of the causal effects UnitEff and StdEff that are computed using these two estimators are shown in Output 37.7.6. For comparison, the target true values of the causal effects are also shown.

Output 37.7.6 Causal Effect Estimation Summary

Obs	Effect	True Effect	Stratified Estimation	Unadjusted Estimation
1	UnitEff	0.007620	0.007766	0.009155
2	StdEff	0.006789	0.006940	0.008181

The estimates that are computed by using the stratified estimator very closely approximate the true values. This is expected, because the set {Smoking,StatinUse} is a valid adjustment set for the data generating process shown in Figure 37.10. However, the estimates that are computed by using the naive unadjusted logistic regression results do not agree with the true values. This is also expected, because the PROC CAUSALGRAPH analysis in Output 37.7.3 shows that the empty set is not a valid adjustment. Thus, this example demonstrates the usefulness of causal graph theory to identify causal effects in confounding situations.

By devising a stratified estimator of the causal effects, this example also demonstrates how to implement a good statistical estimation strategy based on the identification results from PROC CAUSALGRAPH.

References

Angrist, J. D., Imbens, G. W., and Rubin, D. B. (1996). "Identification of Causal Effects Using Instrumental Variables." *Journal of the American Statistical Association* 91:444–455.

Elwert, F. (2013). "Graphical Causal Models." In *Handbook of Causal Analysis for Social Research*, edited by S. L. Morgan, 245–273. Dordrecht: Springer.

Elwert, F., and Winship, C. (2014). "Endogenous Selection Bias: The Problem of Conditioning on a Collider Variable." *Annual Review of Sociology* 40:31–53.

Galles, D., and Pearl, J. (1998). "An Axiomatic Characterization of Causal Counterfactuals." *Foundations of Science* 3:151–182.

Geiger, D., and Pearl, J. (1988). "On the Logic of Causal Models." In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence*, edited by R. D. Shacter, T. S. Levitt, L. N. Kanal, and J. F. Lemmer, 136–147. Amsterdam: North-Holland.

Greenland, S., Pearl, J., and Robins, J. M. (1999). "Causal Diagrams for Epidemiologic Research." *Epidemiology* 10:37–48.

Imbens, G. W. (2014). "Instrumental Variables: An Econometrician's Perspective." *Statistical Science* 29:323–358.

Koller, D., and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA: MIT Press.

Lauritzen, S. L. (1996). *Graphical Models*. Oxford: Clarendon Press.

Neyman, J., Dabrowska, D. M., and Speed, T. P. (1990). "On the Application of Probability Theory to Agricultural Experiments: Essay on Principles, Section 9." *Statistical Science* 5:465–472. Translated and edited by Dabrowska and Speed from the Polish original by Neyman (1923).

Pearl, J. (1993). "Comment: Graphical Models, Causality and Intervention." *Statistical Science* 8:266–269.

Pearl, J. (2009a). "Causal Inference in Statistics: An Overview." *Statistics Surveys* 3:96–146.

Pearl, J. (2009b). *Causality: Models, Reasoning, and Inference*. 2nd ed. Cambridge: Cambridge University Press.

Pearl, J. (2010). "An Introduction to Causal Inference." *International Journal of Biostatistics* 6:1–62.

Pearl, J. (2012). "The Causal Foundations of Structural Equation Modeling." In *Handbook of Structural Equation Modeling*, edited by R. H. Hoyle, 68–91. New York: Guilford Press.

Pearl, J., and Verma, T. (1987). "The Logic of Representing Dependencies by Directed Graphs." In *Proceedings of the Sixth National Conference on Artificial Intelligence*, 374–379. AAAI Press.

Perković, E., Textor, J., Kalisch, M., and Maathuis, M. (2018). "Complete Graphical Characterization and Construction of Adjustment Sets in Markov Equivalence Classes of Ancestral Graphs." *Journal of Machine Learning Research* 18:1–62.

Rubin, D. B. (1980). "Comment on D. Basu, 'Randomization Analysis of Experimental Data: The Fisher Randomization Test'." *Journal of the American Statistical Association* 75:591–593.

Rubin, D. B. (1990). "Comment: Neyman (1923) and Causal Inference in Experiments and Observational Studies." *Statistical Science* 5:472–480.

Schafer, J. L., and Kang, J. (2008). "Average Causal Effects from Nonrandomized Studies: A Practical Guide and Simulated Example." *Psychological Methods* 13:279–313.

Shpitser, I., VanderWeele, T., and Robins, J. M. (2010). "On the Validity of Covariate Adjustment for Estimating Causal Effects." In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, edited by P. Grünwald and P. Spirtes, 527–536. Corvallis, OR: AUAI Press.

Spirtes, P., Glymour, C., and Scheines, R. (2001). *Causation, Prediction, and Search*. 2nd ed. Cambridge, MA: MIT Press.

Takata, K. (2010). "Space-Optimal, Backtracking Algorithms to List the Minimal Vertex Separators of a Graph." *Discrete Applied Mathematics* 158:1660–1667.

Thornley, S., Marshall, R. J., Jackson, R., Gentles, D., Dalbeth, N., Crengle, S., Kerr, A., and Wells, S. (2013). "Is Serum Urate Causally Associated with Incident Cardiovascular Disease?" *Rheumatology* 52:135–142.

Timmermann, C. A. G., Budtz-Jørgensen, E., Petersen, M. S., Weihe, P., Steuerwald, U., Nielsen, F., Jensen, T. K., and Grandjean, P. (2017). "Shorter Duration of Breastfeeding at Elevated Exposures to Perfluoroalkyl Substances." *Reproductive Toxicology* 68:164–170.

Van der Zander, B., Liśkiewicz, M., and Textor, J. (2014). "Constructing Separators and Adjustment Sets in Ancestral Graphs." In *Proceedings of the Thirtieth Conference on Causal Inference: Learning and Prediction*, edited by N. L. Zhang and J. Tian, 11–24. Corvallis, OR: AUAI Press.

Van der Zander, B., Textor, J., and Liśkiewicz, M. (2015). "Efficiently Finding Conditional Instruments for Causal Inference." In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, edited by Q. Yang and M. Wooldridge, 3242–3249. Palo Alto, CA: AAAI Press.

Chapter 38
The CAUSALMED Procedure

Contents

Overview: CAUSALMED Procedure **2420**
- Features of the CAUSALMED Procedure 2422

Getting Started: CAUSALMED Procedure **2424**

Syntax: CAUSALMED Procedure **2429**
- PROC CAUSALMED Statement 2429
- BOOTSTRAP Statement 2436
- BY Statement 2439
- CLASS Statement 2439
- COVAR Statement 2441
- EVALUATE Statement 2441
- FREQ Statement 2445
- MEDIATOR Statement 2445
- MODEL Statement 2449
- STD Statement 2453
- WEIGHT Statement 2454

Details: CAUSALMED Procedure **2455**
- Causal Mediation Effects: Theory, Definitions, and Effect Decompositions 2455
- Causal Mediation Effects: Assumptions, Identification, and Estimation 2462
- Evaluating Causal Mediation Effects 2466
- Bootstrap Methods 2472
- ODS Table Names 2473

Examples: CAUSALMED Procedure **2475**
- Example 38.1: Mediation Analysis with Interaction Effects and Four-Way Decomposition 2475
- Example 38.2: Evaluating Controlled Direct Effects and Conditional Mediation Effects 2478
- Example 38.3: Smoking Effect on Infant Mortality 2483
- Example 38.4: Mediation Analysis by Linear Structural Equation Modeling 2490
- Example 38.5: Causal Mediation Analysis of a Time-to-Event Outcome 2493

References **2500**

Overview: CAUSALMED Procedure

The CAUSALMED procedure estimates causal mediation effects from observational data. In causal mediation analysis, there are four main types of variables:

- an outcome variable Y
- a treatment variable T that is hypothesized to potentially have direct and indirect causal effects on the outcome variable Y (in epidemiology, a treatment variable is also known as an exposure, sometimes denoted by A or E)
- a mediator variable M that is hypothesized to be causally affected by the treatment variable T and that itself has an effect on the outcome variable Y
- a set of pretreatment or background covariates that might confound the observed relationships among Y, T, and M

The relationships among the first three types of variables represent the primary causal mediation effects of interest. The set of covariates represent background or pretreatment characteristics that confound the observed relationships among Y, T, and M. To focus on describing the causal mediation and related effects of interest, the roles of covariates are omitted for the present discussion and will be described later.

Figure 38.1 represents the first three main variables in a causal diagram (see Pearl 2009 for a general theory of causal diagrams).

Figure 38.1 Mediation Model

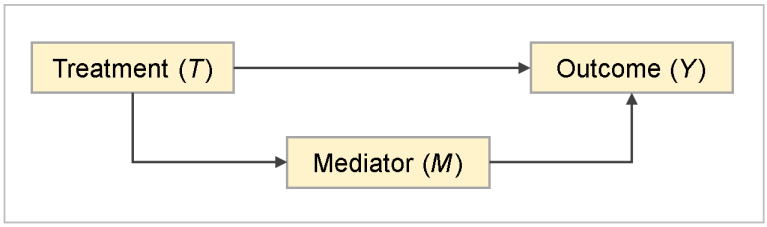

In the terminology of causal diagrams, there are two causal pathways of effects from the treatment variable T:

- A direct pathway: $T \rightarrow Y$
- A mediated or indirect pathway: $T \rightarrow M \rightarrow Y$

For example, the analyses in "Getting Started: CAUSALMED Procedure" on page 2424 and Example 38.1 examine whether parental encouragement (T) affects children's cognitive development (Y) and whether the effect of parental encouragement is mediated by children's learning motivation (M). The analysis in Example 38.3 examines the effect of smoking (T) on infant mortality (Y) and its mediation effect through lowering birth weights (M). The analysis in Example 38.5 investigates the effect of red dye in food (T) on the survival time (Y) of mice and its mediation effect through the development of reticuloendothelial tumors (M).

The mediated or indirect pathway, $T \rightarrow M \rightarrow Y$, provides an explanation of how the cause T leads to the outcome Y—that is, through a mechanism by which T first affects a mediator M and then affects Y (see, for example, VanderWeele 2015). In structural equation modeling (see, for example, Bollen 1989), this pathway simply represents an indirect effect of T on Y. Whereas the former interpretation emphasizes the role of the mediator in the mechanism, the latter interpretation emphasizes the root cause in treatment T. Correspondingly, there are two ways to interpret the direct pathway, $T \rightarrow Y$: as the causal effect of T on Y that is not through the mechanism involving M, or as the direct causal effect of T on Y. As an analysis tool, the CAUSALMED procedure does not distinguish these alternative interpretations.

In traditional structural equation modeling, the diagram representation in Figure 38.1 is often expressed in the form of a linear model that excludes the interaction effect between T and M on Y, which is usually assumed to be continuous. As a result, the definitions and computation of mediation effects do not apply to situations in which interaction effects or nonlinear models are present.

However, under the counterfactual framework for causal mediation analysis (Robins and Greenland 1992; Pearl 2001), the same diagram representation in Figure 38.1 excludes neither the interaction nor nonlinear models. Basing on this counterfactual framework, the CAUSALMED procedure models the following variables in causal mediation analysis:

- outcome variable Y: binary, continuous (including time-to-event responses), or count
- treatment variable T: binary or continuous
- mediator variable M: binary or continuous
- covariates: categorical or continuous

Instead of linear models, PROC CAUSALMED supports a limited set of generalized linear models and survival models for describing the relationships among the Y, T, and M variables. For survival models, right-censored outcomes, which are indicated by a censoring variable, can also be modeled.

In practice, one of the biggest issues is the presence of confounding covariates in observational data. Confounding covariates are those pretreatment or background characteristics that are associated with the Y, T, and M variables before the treatment and mediation take place. They complicate the observed relationships among Y, T, and M by introducing extraneous associations that are not due to the direct and indirect pathways in the causal diagram for mediation analysis.

Therefore, for observational studies, statistical analysis that is based solely on specifying the causal diagram in Figure 38.1 does not lead to causal interpretation of the mediation and related effects. Various approaches have been proposed for dealing with confounding, including the weighting approach of Hong (2015) and the Monte Carlo simulation approach of Imai et al. (2010). The CAUSALMED procedure implements the regression approach of VanderWeele (2014). The procedure conducts causal mediation analysis that involves a single outcome variable, a single treatment variable, a single mediator variable, and multiple covariates. You can analyze multiple outcomes by running the procedure separately for each outcome.

The regression approach relies on correct specification of covariate effects in the outcome and mediator models for covariate effect adjustment. The CAUSALMED procedure enables you to specify these covariate effects in a causal mediation analysis.

Using the CAUSALMED procedure to establish causal interpretations of mediation and related effects requires an understanding of the terminology, concepts, and assumptions of causal mediation analysis. These

technical aspects are explained in the sections "Causal Mediation Effects: Theory, Definitions, and Effect Decompositions" on page 2455 and "Causal Mediation Effects: Assumptions, Identification, and Estimation" on page 2462.

Features of the CAUSALMED Procedure

The CAUSALMED procedure provides the following statements for specifying the regression models in a causal mediation analysis:

- The MODEL statement enables you to specify a generalized linear model or a survival model for the outcome variable.
- The MEDIATOR statement enables you to specify a generalized linear model for the mediator variable.
- The COVAR statement enables you to specify the covariate effects.

The CAUSALMED procedure fits generalized linear models that have binary, negative binomial, Poisson, or normal distributions for the outcome and that have binary or normal distributions for the mediator. For time-to-event outcomes, PROC CAUSALMED fits Cox proportional hazards or accelerated failure time models. For accelerated failure time models, you can specify an exponential, gamma, logistic, normal, or Weibull distribution. You can include interaction effects between the treatment and mediator variables and among the covariates.

If you specify covariate effects in the COVAR statement, the procedure incorporates them when it fits the outcome and mediator models. The model estimates are then used to compute various causal mediation effects.

By default, PROC CAUSALMED computes the following main causal mediation effects:

- total effect (TE)
- controlled direct effect (CDE)
- natural direct effect (NDE)
- natural indirect effect (NIE)

For continuous outcomes and time-to-event outcomes that are fitted by the accelerated failure time model without the log transformation, these effects are computed as mean differences on the original scale. For other time-to-event outcomes, these effects are computed on the hazard ratio scale for the proportional hazards model, and on the mean ratio scale for the accelerated failure time model when the log transformation is applied to the outcome variable. For binary outcomes, these effects are computed on the odds ratio scale. For definitions of these and other effects, see the sections "Estimands on Difference Scale" on page 2460 and "Estimands on Ratio Scale" on page 2461, and the references therein.

In the literature of causal mediation analysis, more detailed decompositions of mediation effects (that is, the three-way or four-way decompositions that are described later) have been developed. These decompositions distinguish various mediation and interaction effects. You can request these detailed decompositions by options.

In addition to computing the default estimated effects, PROC CAUSALMED also computes the following percentage measures relative to the total effect automatically:

- percentage mediated
- percentage due to interaction
- percentage eliminated

PROC CAUSALMED can compute various two-way, three-way, and four-way decompositions of the total effect and their percentages based on formulas of VanderWeele (2014). These decompositions provide more detailed analyses of the mediation and interaction effects.

PROC CAUSALMED computes a four-way decomposition that includes the following effects:

- controlled direct effect (CDE): the component that is due neither to interaction nor mediation
- reference interaction (IRF or $\mathrm{INT}_{\mathrm{ref}}$): the component that is due to interaction but not mediation
- mediated interaction (IMD or $\mathrm{INT}_{\mathrm{med}}$): the component that is due to both interaction and mediation
- pure indirect effect (PIE): the component that is due to mediation but not interaction

PROC CAUSALMED computes the following three-way decompositions:

- NDE + PIE + IMD: natural direct effect, pure indirect effect, and mediated interaction
- CDE + PIE + PAI: controlled direct effect, pure indirect effect, and portion attributed to interaction

PROC CAUSALMED computes the following two-way decompositions:

- NDE + NIE: natural direct effect and natural indirect effect
- CDE + PE: controlled direct effect and portion eliminated
- TDE + PIE: total direct effect and pure indirect effect

For more information about estimation of various causal mediation effects and total effect decompositions, see the section "Causal Mediation Effects: Theory, Definitions, and Effect Decompositions" on page 2455.

Because causal mediation effects are generally defined conditionally on values of covariates, it is important that you interpret mediation effects at suitable levels of the covariates. You can specify the levels in the EVALUATE statement. If you do not specify the covariate levels, PROC CAUSALMED sets them at some default "averaged" levels, which are the same as in the default method that was implemented by Valeri and VanderWeele (2013).

Likewise, you can specify the treatment, control, and mediator levels in the EVALUATE statement to evaluate causal mediation effects that are dependent on the levels of these variables. For example, the controlled direct effect is always evaluated at a controlled level of the mediator variable. For more information about how to

use the EVALUATE statement, see the section "Evaluating Causal Mediation Effects" on page 2466. For an illustration, see Example 38.2.

PROC CAUSALMED supports the computation of standard errors and confidence intervals for the effects by the following methods:

- analytic methods that are based on the asymptotic theory for maximum likelihood estimates
- bootstrap methods that are based on resampling

The default is the analytic method. You can request the bootstrap methods by specifying the BOOTSTRAP statement.

For ratio-type effects, PROC CAUSALMED produces two types of analytic confidence intervals. One is based on the original (nontransformed) scale, and the other is based on the back-transformation from the log scale. For more information, see the CIRATIO= option.

For more information about the maximum likelihood analytic approach, see the section "Causal Mediation Effects: Assumptions, Identification, and Estimation" on page 2462. For more information about bootstrapping, see the BOOTSTRAP statement and the section "Bootstrap Methods" on page 2472.

Getting Started: CAUSALMED Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/cmedgs.sas`.)

This section illustrates basic features of the CAUSALMED procedure for estimating total, direct, and indirect effects and their corresponding percentages.

The example presented in this section is patterned after the theoretical educational models that are discussed by Marjoribanks (1974). However, the data in this example are simulated, and neither the analysis nor the interpretation of procedure output mirrors that of Marjoribanks (1974).

A study is conducted to understand whether an encouraging environment provided by parents has an effect on the cognitive development of children. A key question is whether the effect of parental encouragement is due in part to its enhancement of children's motivation to learn. Two pathways of parental encouragement effect are possible:

- a direct pathway, which can be denoted as Encourage → CogPerform
- a mediated or indirect pathway, which can be denoted as Encourage → Motivation → CogPerform

In these pathways, the variable Encourage represents parental encouragement, the variable Motivation represents the learning motivation of the children, and the variable CogPerform represents the cognitive performance of the children. In the terminology of mediation analysis, Encourage is a treatment or an exposure, Motivation is a mediator, and CogPerform is an outcome.

A simulated sample of 300 observations is saved in a data set named Cognitive. Each observation has six variable values, as shown in Figure 38.2.

Figure 38.2 First 10 Observations of the Input Data Set

Obs	SubjectID	FamSize	SocStatus	Encourage	Motivation	CogPerform
1	1	7	31	36	40	103
2	2	3	27	36	40	103
3	3	0	25	35	40	99
4	4	6	29	36	40	103
5	5	4	22	33	37	79
6	6	2	23	34	38	87
7	7	0	29	37	41	112
8	8	4	23	34	38	87
9	9	3	20	32	36	71
10	10	3	28	36	40	103

The variables are defined as follows:

- CogPerform: the child's score on a cognitive test (outcome)
- Encourage: the sum score of the ratings of three items about parents' encouraging behavior in a questionnaire (treatment)
- FamSize: the size of the child's family
- Motivation: the sum score of the child's levels of motivation as evaluated by the child, the teacher, and the primary caretaker (mediator)
- SocStatus: the child's social status, which is an aggregate measure of household income, parents' occupations, and parents' educational levels
- StudentID: the child's identifier

Variables FamSize and SocStatus are background or pretreatment characteristics that you would like to control for when observing various causal effects—either total, direct, or mediated.

First, consider an analysis in which the pretreatment characteristics are omitted. The following statements invoke PROC CAUSALMED to estimate various effects without controlling for background confounding variables:

```
proc causalmed data=Cognitive all;
   model    CogPerform  = Encourage Motivation;
   mediator Motivation  = Encourage;
run;
```

The ALL option in the PROC CAUSALMED statement displays all available output. The MODEL statement specifies the outcome model for CogPerform, which is affected by Encourage and Motivation. The MEDIATOR statement specifies the mediator model for Motivation, which is affected only by Encourage.

The output produced by PROC CAUSALMED is displayed in Figure 38.3 through Figure 38.6.

Figure 38.3 echoes the modeling information and displays the number of observations read and used in the analysis; it also identifies the outcome, treatment, and mediator variables. By default, PROC CAUSALMED assumes normal distributions and identity links for the response variables in the outcome and mediator models because they are continuous.

Figure 38.3 Model Information

Model Information	
Data Set	WORK.COGNITIVE
Outcome Variable	CogPerform
Treatment Variable	Encourage
Mediator Variable	Motivation
Outcome Modeling	Generalized Linear Model
Outcome Distribution	Normal
Outcome Link Function	Identity
Mediator Modeling	Generalized Linear Model
Mediator Distribution	Normal
Mediator Link Function	Identity

Number of Observations Read	300
Number of Observations Used	300

Figure 38.4 presents the estimated effects. All effect estimates and percentage estimates are significant. The total effect estimate is 8.04, which is decomposed into the natural direct effect (NDE=4.28) and natural indirect effect (NIE=3.76). The estimated controlled direct effect (CDE) is 4.28, which is evaluated at the mean value of the mediator variable Motivation by default. In the current model, CDE is the same as NDE. The 'Percentage Mediated' is 46.74%. This means that slightly less than half of the parental encouragement effect on children's cognitive development can be attributed to the enhancement of children's learning motivation.

Figure 38.4 Summary of Total, Direct, and Mediated Effects

Summary of Effects						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	8.0423	0.03200	7.9796	8.1050	251.30	<.0001
Controlled Direct Effect (CDE)	4.2835	0.1062	4.0754	4.4917	40.33	<.0001
Natural Direct Effect (NDE)	4.2835	0.1062	4.0754	4.4917	40.33	<.0001
Natural Indirect Effect (NIE)	3.7588	0.1091	3.5449	3.9727	34.44	<.0001
Percentage Mediated	46.7377	1.3254	44.1400	49.3353	35.26	<.0001
Percentage Due to Interaction	0	.	.	.	.	.
Percentage Eliminated	46.7377	1.3254	44.1400	49.3353	35.26	<.0001

The tables in Figure 38.5 and Figure 38.6 are useful for confirming the direction of the effects. Figure 38.5 shows the estimates of the outcome model for CogPerform.

Figure 38.5 Estimates of the Outcome Model

Outcome Model Estimates						
Parameter	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	-201.21	0.6426	-202.47	-199.95	98053.6157	<.0001
Encourage	4.2835	0.1062	4.0754	4.4917	1626.7935	<.0001
Motivation	3.7576	0.1052	3.5514	3.9639	1274.6903	<.0001
Scale	0.4605	0.01880	0.4251	0.4989		

Figure 38.6 shows the estimates of the mediator model for Motivation. The estimates of the direct effects from Encourage and Motivation are both positive and significant, thus confirming the positive effect of parental encouragement on children's learning motivation.

Figure 38.6 Estimates of the Mediator Model

Mediator Model Estimates						
Parameter	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	4.0428	0.2641	3.5251	4.5605	234.2732	<.0001
Encourage	1.0003	0.007663	0.9853	1.0153	17040.9178	<.0001
Scale	0.2526	0.01031	0.2332	0.2737		

Although the preceding analysis is interpretable, it does not take full advantage of the causal analytic techniques that are available in the CAUSALMED procedure. In order to draw valid causal interpretations from observational data, you must statistically control for all important confounding background characteristics.

Assume that FamSize and SocStatus are the only important confounding background characteristics that need to be controlled for. You can specify these variables as covariates in the COVAR statement and use PROC CAUSALMED as follows to fit an appropriate causal mediation model:

```
proc causalmed data=Cognitive;
   model    CogPerform  = Encourage Motivation;
   mediator Motivation  = Encourage;
   covar FamSize SocStatus;
run;
```

When the confounding covariates FamSize and SocStatus are included, the procedure adjusts the estimates of the causal effects, leading to a new set of results which are summarized in Figure 38.7.

Figure 38.7 Summary of Causal Effects

Summary of Effects						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8435	0.1525	6.5446	7.1424	44.88	<.0001
Controlled Direct Effect (CDE)	4.2962	0.1098	4.0811	4.5114	39.14	<.0001
Natural Direct Effect (NDE)	4.2962	0.1098	4.0811	4.5114	39.14	<.0001
Natural Indirect Effect (NIE)	2.5473	0.1563	2.2410	2.8536	16.30	<.0001
Percentage Mediated	37.2219	1.7523	33.7874	40.6564	21.24	<.0001
Percentage Due to Interaction	0	.	.	.	.	.
Percentage Eliminated	37.2219	1.7523	33.7874	40.6564	21.24	<.0001

The total effect of Encourage on CogPerform is now 6.84, which is about 1 point lower than the total effect that is obtained without including the confounding covariates in the analysis (see Figure 38.4). This discrepancy suggests that parts of the observed association between Encourage and CogPerform are indeed due to their associations with the confounding background covariates. Failure to adjust for the confounding covariates led to inflated estimates of the total causal effect in Figure 38.4.

The natural direct effect (NDE) in the current analysis is 4.30, which is not much different from that of the preceding analysis. However, the natural indirect effect (NIE) is now 2.55, which is more than 1 point lower

than the NIE in Figure 38.4. Finally, the 'Percentage Mediated' is now only 37%, which is almost 10% lower than the 'Percentage Mediated' (47%) that Figure 38.4 shows.

These results demonstrate that you must carefully consider the set of confounding covariates when conducting a causal mediation analysis. First and foremost, the no unmeasured confounding assumption must be reasonably satisfied. That is, to enable causal interpretations of the effect estimates, the baseline covariates for which adjustment is made must suffice to control for treatment-outcome, mediator-outcome, and treatment-mediator confounding. Second, causal analysis from observational data might involve many other assumptions that require serious attention. For instance, in the current example you could consider the following questions:

1. Why should the analysis assume that the variables Encourage and Motivation do not have an interaction effect on CogPerform? Is there any justification for this assumption?
2. If there is an interaction effect between the treatment and the mediator, what is the amount of this effect?
3. What justifies treating Encourage as the cause and CogPerform as the effect?
4. Is the causal sequence among the variables Encourage, Motivation, and CogPerform properly captured in the data?

You can address Questions 1 and 2 by fitting a more general model that includes the interaction term to determine whether the interaction effect is ignorable. PROC CAUSALMED supports outcome models that have interaction effects, as illustrated in Example 38.1, which presents a continuation of the current analysis.

Question 3 does not have a definite statistical answer. Substantive knowledge or existing evidence of the relationships is required to support these justifications.

An answer to Question 4 must also be justified by using substantive knowledge about the system of interest. In many systems there are temporal conditions that the data must satisfy so that the effects of the treatment on the outcome, the treatment on the mediator, and the mediator on the outcome can be observed.

Some researchers use longitudinal studies to establish the causal sequence. For instance, you can collect data in stages to ensure a proper temporal ordering of the causal, mediation, and outcome events. In this example, you could collect data for CogPerform several months after collecting data for Motivation, which were collected several years after you obtained the information about Encourage and any pretreatment confounders.

If you collect all the data at the same time point, you would need to justify that the parental encouragement pattern has long been established and that its effect on children's learning motivation has been stabilized well before the children took the cognitive performance test. In addition, you would also need to justify that the background or pretreatment characteristics had been stabilized well before the measurements of the treatment, mediator, and the outcome. Substantive knowledge is required to support these justifications.

The role of CAUSALMED procedure is to estimate causal mediation effects given that all related assumptions are satisfied. The procedure can only serve as a tool to refute the presence of causal effects (when estimates are close to zero) given the model. The procedure cannot be used to establish causal interpretations of effects if the necessary methodological and statistical assumptions are not satisfied. For more information about assumptions of causal mediation analysis, see the sections "Causal Mediation Effects: Theory, Definitions, and Effect Decompositions" on page 2455 and "Causal Mediation Effects: Assumptions, Identification, and Estimation" on page 2462.

Syntax: CAUSALMED Procedure

The following statements are available in the CAUSALMED procedure:

```
PROC CAUSALMED < options > ;
    CLASS variables < (options) > . . . < variable< (options) > > < / global-options > ;
    MODEL outcome < *censor(list) > < (outcome-options) > =effects < / model-options > ;
    MEDIATOR mediator < (med-options) > =treatment < (treat-options) > ;
    COVAR effects ;
    BOOTSTRAP < options > ;
    BY variables ;
    EVALUATE 'label' assignment < assignment . . . > < / options > ;
    FREQ variable ;
    STD variable=value < variable=value . . . > ;
    WEIGHT variable ;
```

Together with the PROC CAUSALMED statement, the MODEL and MEDIATOR statements are essential to causal mediation analysis. The MODEL statement provides the model for the outcome variable; you use this statement to specify the effects of the treatment, the mediator, and possibly their interactions on the outcome variable. The MEDIATOR statement provides the model for the mediator variable; you use this statement to specify the effect of the treatment on the mediator variable.

In addition to the MODEL and MEDIATOR statements, you use the COVAR statement to specify the effects of confounding covariates. Because the assumption of no unmeasured confounding covariates is critical to the validity of causal mediation analysis for observational data, it is important that you specify all important covariate effects in this statement.

The CLASS statement, if provided, must precede the MODEL, MEDIATOR, and COVAR statements. The CLASS statement names the classification variables to be used in the analysis.

The following sections describe the PROC CAUSALMED statement and then describe the other statements in alphabetical order.

PROC CAUSALMED Statement

```
PROC CAUSALMED < options > ;
```

The PROC CAUSALMED statement invokes the CAUSALMED procedure. Table 38.1 summarizes the *options* available in the PROC CAUSALMED statement.

Table 38.1 Options Available in the PROC CAUSALMED Statement

Option	Description
Data Set and Variable Options	
DATA=	Specifies the input SAS data set
DESCENDING	Reverses the order of levels of binary outcome and mediator variables

Table 38.1 *continued*

Option	Description
NAMELENGTH=	Specifies the length of effect names
ORDER=	Specifies the ordering method for the levels of the classification variable
RORDER=	Specifies the ordering method for the levels of the outcome variable
Estimation and Analysis	
ALPHA=	Specifies the level for confidence intervals
CASECONTROL	Requests an analysis for a case-control study
CIRATIO=	Specifies the scale for confidence interval construction of ratio-type effects
DECOMP	Requests various decompositions of the total effect
VARDEF=	Specifies the divisor to use in calculating variances or standard deviations
Displayed Output	
NOPRINT	Suppresses display of all output
PALL	Displays all output
PMEDMOD	Displays mediator model parameter estimates
POUTCOMEMOD	Displays outcome model parameter estimates
PSHORT	Displays only the basic modeling information and effects summary
PSUMMARY	Displays only the effects summary
Technical Details	
NLOPTIONS	Specifies the optimization options for model fitting
SINGULAR=	Specifies the singularity criterion
THREADS=	Specifies the number of threads to use

You can specify the following *options*:

ALPHA=*p*

specifies the level $1 - p$ for constructing confidence intervals. By default, $p = 0.05$, which corresponds to $1 - p = 95\%$ confidence intervals. If p is greater than 1, it is interpreted as a percentage and divided by 100. When multiple confidence intervals are constructed, this level is applied to each interval one at a time. This will not control the coverage probability of the intervals simultaneously. To control familywise coverage probability, you might consider supplying a value of p that is precomputed based on a method such as Bonferroni adjustment.

CASECONTROL

requests an analysis for a case-control study. When you specify this option, PROC CAUSALMED fits a mediator model by using only observations for subjects in the control group (VanderWeele and Vansteelandt 2010).

In case-control studies, a group of subjects is identified with a target outcome condition (for example, a disease). This group is called the case group. A second group, known as the control group, is formed

by identifying subjects who are known to be absent of the target outcome, but whose background characteristics are the same as those of the case group. The values of the hypothesized exposure or treatment variable of the case and control groups are then compared to see whether the outcome can be attributed to the exposure or treatment variable.

CIRATIO=ALL | LOG | NONTRANS
CIRATIO=(ALL | LOG | NONTRANS)

computes the confidence intervals (CIs) for ratio-type causal mediation effects on the log scale, original (nontransformed) scale, or both. These ratio-type effects include the odds ratio, mean ratio, and hazard ratio effects, which are all nonnegative and have null (no-effect) values at 1. The z-value and the p-value for testing null hypothesis of no effect are also affected by the choice of these scales. By default, CIRATIO=ALL.

By using the parentheses, you can specify more than one keyword for the CIRATIO= option. Otherwise, you can specify only one of the following keywords for the scale:

ALL computes the CIs, z-values, and p-values of the ratio-type effects that are based on the log scale and the original effect scale, separately.

LOG computes the Wald-type CIs of ratio-type effects on the log scale and then back-transforms to the confidence limits on the original effect scale by exponentiation. Hence, the constructed CIs for effects are generally asymmetrical around the corresponding point estimates. The z-value and p-value for testing the null effect hypothesis of no effect on the log scale are also computed.

NONTRANS computes the Wald-type CIs of ratio-type effects on the original scale. Hence, the constructed CIs for effects are always symmetrical around the corresponding point estimates. The z-value and p-value for testing the null effect hypothesis of no effect on the original (nontransformed) scale are also computed.

The computations of confidence intervals, z-values, and p-values of estimates for the difference-type effects, excess ratios, or percentages are always based on the original (nontransformed) scale and would not be affected by this option.

DATA=*SAS-data-set*

specifies an input data set that contains the raw data. If the DATA= option is omitted, the most recently created SAS data set is used.

DECOMP<=*i***>**

requests various decompositions of the total effect. By default, several two- and three-way decompositions and a four-way decomposition are computed. When you specify 2, 3, or 4 for i, decompositions up to an i-way decomposition are computed.

For continuous outcomes, the decomposition of the total effect is on the original continuous scale. For binary responses, the decomposition of the total effect is on the excess relative risk scale (VanderWeele 2014). In addition, PROC CAUSALMED displays the corresponding decompositions as percentages.

The four-way decomposition is described in the section "Causal Mediation Effects: Theory, Definitions, and Effect Decompositions" on page 2455. It contains the following four components:

- CDE (controlled direct effect): the component effect that is not due to interaction or mediation

- IRF (reference interaction): the component effect that is due to interaction but not mediation (IRF is denoted as $\mathrm{INT}_{\mathrm{ref}}$ in VanderWeele (2014))
- IMD (mediated interaction): the component effect that is due to both interaction and mediation (IMD is denoted as $\mathrm{INT}_{\mathrm{med}}$ in VanderWeele (2014))
- PIE (pure indirect effect): the component effect that is due to mediation but not interaction

PROC CAUSALMED computes the following three-way decompositions:

- NDE + PIE + IMD: natural direct effect, pure indirect effect, and mediated interaction
- CDE + PIE + PAI: controlled direct effect, pure indirect effect, and portion attributed to interaction

PROC CAUSALMED computes the following two-way decompositions:

- NDE + NIE: natural direct effect and natural indirect effect
- CDE + PE: controlled direct effect and portion eliminated
- TDE + PIE: total direct effect and pure indirect effect

For more information about the logic and interpretations of these decompositions, see VanderWeele (2014) and VanderWeele (2015).

DESCENDING
DESCEND
DESC

sorts the levels of the binary outcome and the binary mediator variables in reverse of the specified order.

NAMELENGTH=*n*

specifies the maximum length of effect names in tables to be *n* characters, where *n* is a value between 20 and 128. By default, NAMELEN=20.

NLOPTIONS(*nlo-options***)**

specifies options for the nonlinear optimization methods that are used for fitting the specified models. You can specify one or more of the following *nlo-options* separated by spaces:

ABSCONV=*r*
ABSTOL=*r*

specifies an absolute function convergence criterion by which minimization stops when $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of *r* is the negative square root of the largest double-precision value.

ABSFCONV=*r*
ABSFTOL=*r*

specifies an absolute function difference convergence criterion. Termination requires a small change of the function value in successive iterations,

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. By default, ABSFCONV=0.

ABSGCONV=*r*

ABSGTOL=*r*

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small,

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the *j*th parameter. By default, ABSGCONV=1E–7.

FCONV=*r*

FTOL=*r*

specifies a relative function convergence criterion. Termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. By default, FCONV=$10^{-\mathrm{FDIGITS}}$, where by default FDIGITS is $-\log_{10}\{\epsilon\}$, where ϵ is the machine precision.

GCONV=*r*

GTOL=*r*

specifies a relative gradient convergence criterion. For all values of the TECHNIQUE= suboption except CONGRA, termination requires the normalized predicted function reduction to be small,

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. When TECHNIQUE=CONGRA (for which a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \; |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

By default, GCONV=1E–8.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values depend on the value of the TECHNIQUE= suboption as follows:

- TRUREG, NRRIDG, and NEWRAP: 125
- QUANEW and DBLDOG: 500
- CONGRA: 1,000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed *n*.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values depend on the value of the TECHNIQUE= suboption as follows:

- TRUREG, NRRIDG, and NEWRAP: 50
- QUANEW and DBLDOG: 200
- CONGRA: 400

These default values also apply when *n* is specified as a missing value.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. Because the time is checked only at the end of each iteration, the actual run time might be longer than *r*. By default, CPU time is not limited.

TECHNIQUE=CONGRA | DBLDOG | NEWRAP | NRRIDG | QUANEW | TRUREG

specifies the optimization technique to obtain maximum likelihood estimates. You can specify the following values:

CONGRA performs a conjugate-gradient optimization.

DBLDOG performs a version of double-dogleg optimization.

NEWRAP performs a Newton-Raphson optimization that combines a line-search algorithm with ridging.

NRRIDG performs a Newton-Raphson optimization with ridging.

QUANEW performs a dual quasi-Newton optimization.

TRUREG performs a trust-region optimization.

By default, TECHNIQUE=NRRIDG.

For more information about these optimization methods, see the section "Choosing an Optimization Algorithm" on page 545 in Chapter 20, "Shared Concepts and Topics."

NOPRINT

suppresses all displayed output. For more information about the options for controlling output display, see the section "ODS Table Names" on page 2473.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of CLASS *variables*. This ordering determines which parameters in the model correspond to each level in the data.

You can specify the following values:

DATA sorts the levels in their order of appearance in the input data set.

FORMATTED sorts the levels by external formatted values, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) values. The sort order is machine-dependent.

FREQ sorts the levels by descending frequency count. Levels that have more observations come earlier in the order.

INTERNAL sorts the levels by an unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PALL

ALL

displays all output tables. For more information about the options for controlling output display, see the section "ODS Table Names" on page 2473.

PMEDMOD

displays parameter estimates for the mediator model. For more information about the options for controlling output display, see the section "ODS Table Names" on page 2473.

POUTCOMEMOD

displays parameter estimates for the outcome model. For more information about the options for controlling output display, see the section "ODS Table Names" on page 2473.

PSHORT

displays only the basic modeling information and the summary of effects. When you specify this option, you can also display the effect decomposition table by specifying the DECOMP option. For more information about the options for controlling output display, see the section "ODS Table Names" on page 2473.

PSUMMARY

displays only a summary of effects. When you specify this option, you can also display the effect decomposition table by specifying the DECOMP option. For more information about the options for controlling output display, see the section "ODS Table Names" on page 2473.

RORDER=DATA | FORMATTED | FREQ | INTERNAL

RESPORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the outcome variable. In order for this option to apply, either the outcome variable must be specified in the CLASS statement or the DIST=BIN option must be specified in the MODEL statement. The following table shows how PROC CAUSALMED interprets values of the RORDER= option.

Value of RORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.
FREQ	Descending frequency count. Levels that have the most observations come first in the order.
INTERNAL	Unformatted value. The sort order is machine-dependent.

By default, RORDER=FORMATTED. The DESCENDING option in the PROC CAUSALMED statement causes the response variable to be sorted in reverse of the order displayed in the previous table. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide*.

SINGULAR=*tolerance*
specifies the *tolerance* for testing the singularity of a matrix, where *tolerance* must be between 0 and 1. The default *tolerance* is 1E7 times the machine precision.

THREADS=*n*
NTHREADS=*n*
specifies the number of threads (*n*) for analytic computations and overrides the SAS system option THREADS | NOTHREADS. If you do not specify the THREADS= option or if you specify THREADS=0, the number of threads is determined from the number of CPUs in the host on which the analytic computations execute.

VARDEF=DF | N | WDF | WEIGHT | WGT
specifies the divisor to use in calculating the variance and standard deviation. By default, VARDEF=DF. With n denoting the total number of observations and w_i denoting the weight for observation i, the values and associated divisors are displayed in the following table.

Value	Description	Divisor
DF	Degrees of freedom	$n - 1$
N	Number of observations	n
WDF	Sum of weights DF	$\sum_i w_i - 1$
WEIGHT | WGT	Sum of weights	$\sum_i w_i$

You can use the WEIGHT statement to specify the variable that contains weights. If the WEIGHT statement is not used, each w_i has a value of 1.

BOOTSTRAP Statement

BOOTSTRAP < *options* > **;**

The BOOTSTRAP statement requests bootstrap estimates of standard errors and bootstrap confidence intervals for various effects and percentages of total effects.

Table 38.2 summarizes the *options* available in the BOOTSTRAP statement.

Table 38.2 Summary of Options in BOOTSTRAP Statement

Option	Description
BOOTCI	Produces bootstrap confidence intervals for effects and percentages
IGNORECC	Ignores the case-control design and performs a regular bootstrap sampling
MINSAMP=	Specifies the minimum number of converged bootstrap samples required for bootstrap estimation
NBOOT=	Specifies the number of bootstrap sample data sets (replicates)
NOSKIP	Includes bootstrap samples that have inconsistent class levels
SEED=	Specifies the seed that initializes the random number stream

You can specify the following *options*:

BOOTCI <(BC | NORMAL | PERC | ALL)>

CI <(BC | NORMAL | PERC | ALL)>

computes bootstrap-based confidence intervals for the effects and percentages of effects, and displays them in the "Summary of Effects" table. This table includes a column that indicates the number of bootstrap samples used to compute the confidence intervals; this column is not displayed but is available if you save the table as an output data set by using the ODS OUTPUT statement. You can also display this column by modifying the corresponding template.

You can specify one or more of the following types of bootstrap confidence intervals separated by spaces:

ALL produces all three confidence intervals, which are described in the following types.

BC produces bias-corrected confidence intervals. You must specify a value of 1,000 or more for the NBOOT= option, but the confidence intervals are not computed if fewer than 900 bootstrap replicates produce bootstrap estimates.

NORMAL produces confidence intervals that are based on the assumption that bootstrap estimates follow a normal distribution. You must specify a value of 50 or more for the value NBOOT= option, but the corresponding standard errors and confidence intervals are not computed if fewer than 30 bootstrap replicates produce bootstrap estimates.

PERC produces percentile-based confidence intervals. You must specify a value of 1,000 or more for the NBOOT= option, but the confidence intervals are not computed if fewer than 900 bootstrap replicates produce bootstrap estimates.

The ALPHA= option in the PROC CAUSALMED statement sets the confidence level for constructing bootstrap intervals. For more information about how bootstrap-based confidence intervals are computed, see the section "Bootstrap Methods" on page 2472. By default, PROC CAUSALMED produces bias-corrected confidence intervals (BOOTCI(BC)) based on 1,000 bootstrap samples.

IGNORECC

ignores the case-control design and performs a regular bootstrap sampling that draws observations from the entire sample with replacement. When you specify a case-control design by using the CASECONTROL option and omit the IGNORECC option, the default bootstrapping draws samples separately from the case and control groups. As a result, the bootstrap samples maintain the same numbers of case and control observations as those of the original sample. However, when you specify the IGNORECC option, it overrides the default bootstrap sampling method for case-control studies, and the numbers of case and control observations generally vary from (bootstrap) sample to sample.

If you do not specify the CASECONTROL option, the procedure performs regular bootstrapping automatically and the IGNORECC option is irrelevant.

MINSAMP=*n*

specifies the minimum number of converged bootstrap samples that is required for conducting any bootstrap estimation, where *n* is between 30 and 10,000. By default, *n*=30.

NBOOT=*n*

NSAMPLE=*n*

NSAMPLES=*n*

specifies the number of bootstrap sample data sets (replicates), where *n* is between 50 and 10,000. By default, *n*=1000.

NOSKIP

includes in the bootstrap estimation bootstrap samples that have inconsistent class levels in either the mediator or outcome model. By default, for any classification variable in a bootstrap sample data set that does not contain all levels that have been used in fitting either the mediator or outcome model for the original input data set, PROC CAUSALMED treats the corresponding bootstrap sample estimates as nonconvergent and skips the estimation of the mediation effects. This option overrides the default and continues the estimation of the mediation effects for bootstrap samples that have inconsistent class levels.

SEED=*n*

provides the seed that initializes the random number stream for generating the bootstrap sample data sets (replicates). If you do not specify this option or if you specify a value for *n* that is less than or equal to 0, the seed is generated from reading the time of day from the computer's clock. The largest possible value for the seed is $2^{31} - 1$.

You can use the SYSRANDOM and SYSRANEND macro variables after a PROC CAUSALMED step to query the initial and final seed values. However, using the final seed value as the starting seed for a subsequent analysis does not continue the random number stream where the previous analysis ended. The SYSRANEND macro variable provides a mechanism to pass on seed values to ensure that the sequence of random numbers is the same every time you run an entire program. To reproduce the random number stream that was used to generate bootstrap estimates, you must specify the same value for the SEED= option and the same value for the THREADS= option in the PROC CAUSALMED statement.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC CAUSALMED to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the CAUSALMED procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *global-options* > **;**

The CLASS statement names one or more classification *variables* to be used as explanatory variables in the analysis.

The CLASS statement must precede the COVAR, MEDIATOR, and MODEL statements. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables specified in the CLASS statement. However, individual CLASS variable *options* override the *global-options*. Unless otherwise indicated, you can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

uses at most the first *n* characters of a CLASS *variable* name in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING

DESC

reverses the sort order of the CLASS *variables*. If both the DESCENDING and ORDER= options are specified, PROC CAUSALMED orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

uses at most the first *n* characters of a CLASS *variable* name in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (blanks for character variables and ., ._, .A, . . . , .Z for numeric variables) as valid values for the CLASS *variables*.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of CLASS *variables*. This ordering determines which parameters in the model correspond to each level in the data.

You can specify the following values:

DATA sorts the levels in their order of appearance in the input data set.

FORMATTED sorts the levels by external formatted values, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) values. The sort order is machine-dependent.

FREQ sorts the levels by descending frequency count. Levels that have more observations come earlier in the order.

INTERNAL sorts the levels by an unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

REF='*level***' | FIRST | LAST**

specifies a level of the CLASS *variable* to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates that have a singular parameterization.

You can specify the following values:

'*level*' specifies the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. You cannot specify '*level*' as a *global-option*.

FIRST designates the first ordered level as the reference level.

LAST designates the last ordered level as the reference level.

By default, REF=LAST.

TRUNCATE< =*n* >

specifies the length (*n*) of variable values to use in determining the CLASS *variable* levels. The default is to use the full formatted length of the CLASS *variable*. If you specify this option without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a *global-option*.

COVAR Statement

COVAR *effects* **;**

The COVAR statement specifies the effects of covariates in a causal mediation analysis. These covariates represent important confounders in the causal model. You do not need to distinguish confounders for the treatment-outcome, treatment-mediator, or mediator-outcome relationships. You simply enter all confounding covariate effects in this statement. PROC CAUSALMED appends these effects into the design matrices of the outcome and mediator models that you specify in the MODEL and MEDIATOR statements, respectively. The causal mediation and other related effects are thus estimated with adjustment for confounding by including covariate effects in outcome and mediator modeling.

The simplest form of *effects* is a list of confounding covariates. For example, the following statement specifies that C1, C2, and C3 are confounding covariates in the causal model:

```
covar C1 C2 C3;
```

You can also include interaction terms in the specification. For example, the following statement adds the interaction of C1 and C2 as a confounding effect to the preceding specification:

```
covar C1 C2 C3 C1*C2;
```

Alternatively, you can use the following equivalent specification:

```
covar C1|C2 C3;
```

If a confounding covariate represents nominal (classification) data, you must also include the covariate in the CLASS statement. For more information about specifying *effects*, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

EVALUATE Statement

EVALUATE '*label*' *assignment* < *assignment . . .* > < / *options* > **;**

You use the EVALUATE statement to specify variable levels or values for evaluating various effects. In the *assignments*, you can specify one or more of the following variable levels:

- the control and treatment levels for computing all effects
- the mediator level for computing the controlled direct effect
- the covariate levels for evaluating various conditional causal effects

Each *assignment* is of the form

var-key=value-key

where *var-key* specifies a variable and *value-key* is either its numerical value, its character value, or a keyword (such as MEAN) that generates a value from the variable.

Because of interaction effects and nonlinear models, computation of causal mediation effects usually depend on the values or levels of the treatment, control, mediator, or covariate levels. PROC CAUSALMED assigns values or levels to these variables automatically when it performs a default mediation effect analysis. By setting these default variable levels, you obtain "overall" measures of causal mediation and related effects. For more information about how PROC CAUSALMED sets the default values of levels, see the section "Evaluating Causal Mediation Effects" on page 2466.

However, to address your particular research questions more directly, you can provide EVALUATE statements with specific variable levels to evaluate mediation and related effects. Specifying covariate levels, the treatment level (of the treatment variable), or the control level (of the treatment variable) changes the estimates of all mediation effects and decompositions. Specifying the controlled level of the mediator variable does not change the estimates of the total effect (TE), the natural direct effect (NDE), or the natural indirect effect (NIE). But it does change the estimates of the controlled direct effect (CDE) and the reference interaction (IRF).

You can provide as many EVALUATE statements as you want. Each statement specifies an assignment scheme that defines the mediation effects and produces a summary of effects, decompositions of effects (if requested), and percentage decompositions of effects (if requested).

To distinguish the results that are produced by different EVALUATE statements, you can specify a distinct *label* in each EVALUATE statement. A maximum of 256 characters is allowed for each *label*. This label is displayed in the output tables.

For example, suppose that C1 and C2 are continuous covariates in the mediation model and you want to evaluate the mediation effects at C1=5 and C2=10. You can request that by providing the following statement:

```
evaluate 'Set C1=5 C2=10' C1=5 C2=10;
```

In this statement, the quoted string, 'Set C1=5 C2=10', labels the set of assignments for evaluating the mediation effects and is followed by the *assignments*, `C1=5` and `C2=10`.

If you want to evaluate the mediation effects conditioned on a different set of covariate values, you can add another EVALUATE statement. For example,

```
evaluate 'Scheme 1 -- C1=5  C2=10' C1=5  C2=10;
evaluate 'Scheme 2 -- C1=10 C2=5'  C1=10 C2=5;
```

Meaningful labels for the EVALUATE statements are highly recommended in practice.

If you use '_Default' as the *label*, PROC CAUSALMED overrides the default variable levels for evaluating mediation effects. For example, the following statement generates only one set of mediation effect output tables, which replace the default tables:

```
evaluate '_Default'  C1=5 C2=10;   /* Overrides the default assignment scheme */
```

In addition to the use of fixed value assignments (such as `C1=5` in the preceding examples), PROC CAUSALMED provides several ways to specify the *var-key* and the *value-key* in an *assignment*.

You can use the following *var-keys* in an *assignment*:

_CONTROL | _A0 | _T0

varname(**CONTROL**)

specifies the control level of the treatment variable, where *varname* represents the actual treatment variable name.

covariate-name

specifies a covariate by using its actual variable name for *covariate-name*.

_MEDIATOR | _MSTAR

varname

specifies the controlled level of the mediator variable, where *varname* represents the actual mediator variable name.

_TREATMENT | _A1 | _T1

varname(**TREATMENT**)

specifies the treatment level of the treatment variable, where *varname* represents the actual treatment variable name.

In all the preceding examples, actual variable names have served as *var-keys* and numerical values have served as *value-keys*. The following statements show examples of *assignments* that specify keywords for *var-keys*:

```
evaluate 'Scheme 3' _treatment=max _control=mean _mediator=last C1=mode;
evaluate 'Scheme 4' _A1=0 _A0=1 _mstar=10 C1='Boys';
evaluate 'Scheme 5' _A1=.5 _A0=-.5 _mediator=0 C1=2;
```

You can use the following *value-keys* in an *assignment*:

'*level*'

assigns the *level* of the corresponding classification variable that is specified in the *var-key*, where *level* represents an actual character level of the variable.

FIRST

assigns the first level of the corresponding classification variable that is specified in the *var-key*.

LAST

assigns the last level of the corresponding classification variable that is specified in the *var-key*.

MAX

assigns the maximum variable value (denoted as *max*) of the corresponding numerical variable that is specified in the *var-key*. If you assign this *value-key* to both the treatment and control levels of the treatment variable, then the treatment level is *max*+0.5 and the control level is *max*–0.5. If you assign this *value-key* to the treatment level but not the control level, then the treatment level is *max* and the control level is *max*–1. If you assign this *value-key* to the control level but not the treatment level, then the control level is *max* and the treatment level is *max*+1.

MEAN

assigns the mean variable value (denoted as *mean*) of the corresponding numerical variable that is specified in the *var-key*. If you assign this *value-key* to both the treatment and control levels of the treatment variable, then the treatment level is *mean*+0.5 and the control level is *mean*–0.5. If you assign this *value-key* to the treatment level but not the control level, then the treatment level is *mean* and the control level is *mean*–1. If you assign this *value-key* to the control level but not the treatment level, then the control level is *mean* and the treatment level is *mean*+1.

MIN

assigns the minimum variable value (denoted as *min*) of the corresponding numerical variable that is specified in the *var-key*. If you assign this *value-key* to both the treatment and control levels of the treatment variable, then the treatment level is *min*+0.5 and the control level is *min*–0.5. If you assign this *value-key* to the treatment level but not the control level, then the treatment level is *min* and the control level is *min*–1. If you assign this *value-key* to the control level but not the treatment level, then the control level is *min* and the treatment level is *min*+1.

MODE

assigns the modal level of the corresponding CLASS variable that is specified in the *var-key*. In multimodal situations, the modal classes are averaged in a particular way. For more information about the averaging process of modal classes, see the section "Evaluating Causal Mediation Effects" on page 2466.

value<**(SD)**>

assigns the numerical *value* in the *assignment*, where *value* represents a fixed number. If you use the SD option, the measurement scale of the numerical value refers to the measurement scale of the standardized variable. Hence, with the SD option the actual assigned value is

$$m + \mathit{value} * s$$

where m and s are the sample mean and standard deviation, respectively, of the corresponding variable that is specified in the *var-key* of the *assignment*.

The following statements show examples of *assignments* that use different types of keywords for *value-keys*:

```
evaluate 'Evaluation 6' _treatment=.5(SD) _control=-0.5(SD) _mediator=min
                        C1=mode;
evaluate 'Evaluation 7' _A1=first _A0=last _mediator=mean C1='Boys';
```

After specifying the *assignments* in an EVALUATE statement, you can use one or more of the following *options* to control the displays of mediation effects and decompositions that are generated by the EVALUATE statement:

CLABEL='*clabel*'

CLABEL=*clabel*

specifies a short content-label for the mediation effects that are generated by the EVALUATE statement. The *clabel* is used to label the corresponding set of tables in the output. Only the first 20 characters of *clabel* is used. If you do not specify this option, the first 20 characters of the *label*, which you specify in the beginning of the EVALUATE statement, is used as *clabel*.

DECOMP< =*i* >

specifies the type of decompositions requested, where *i* is between 2 and 4, representing two-, three-, or four-way decompositions, respectively. If you specify the DECOMP= options in the PROC CAUSALMED statement and in an EVALUATE statement, the DECOMP option in the EVALUATE statement is used for evaluating the requested effects.

NODECOMP

suppresses the display of all decomposition results for the specified evaluation scheme of mediation effects. Only the summary table of effects is shown. This option also overrides the DECOMP= option in the same EVALUATE statement.

For more information about how variable levels are related to the interpretation of causal mediation effects, see the section "Evaluating Causal Mediation Effects" on page 2466. For illustrations of the use of the EVALUATE statement, see Example 38.2 and Example 38.3.

FREQ Statement

FREQ *variable* **;**

If a variable in the data set represents the frequency of occurrence for the other values in the observation, include the variable's name in a FREQ statement. The procedure then treats the data set as if each observation appears *n* times, where *n* is the value of the FREQ *variable* for the observation. The total number of observations is considered to be equal to the sum of the FREQ *variable* values when the procedure determines degrees of freedom for significance probabilities.

If the value of the FREQ *variable* is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, the value is truncated to an integer.

MEDIATOR Statement

MEDIATOR *mediator* < *(med-options)* > **=***treatment* < *(treat-options)* > **;**

The MEDIATOR statement is required for specifying the mediator model. You provide *mediator* (the name of the mediator variable) to the left of the equal sign and *treatment* (the name of the treatment variable) to the right. You cannot specify more than one MEDIATOR statement in an analysis.

For example, the following statement specifies M as the mediator variable and T as the treatment variable in the analysis:

```
mediator M = T;
```

Together, the COVAR, MEDIATOR, and MODEL statements specify the relationships of all variables in the mediation analysis. The mediator and treatment variables that you specify in the MEDIATOR statement must be consistent with those that you specify in the MODEL statement. If there are covariates in the analysis, do not specify them or their effects in the MEDIATOR statement even though covariate effects on the mediator variable are being modeled. Instead, use the COVAR statement to specify covariate effects.

Mediator and treatment variables can be either binary or continuous. You can specify that a mediator or treatment variable is a binary variable by listing it in the CLASS statement. Otherwise, it is assumed to be continuous.

PROC CAUSALMED assumes a normal distribution and the identity link function for a continuous mediator, and a Bernoulli distribution (binomial distribution of a single trial) and the logit link function for a binary mediator.

For binary mediators, it is important to indicate which level or category represents the events being modeled and which level or category represents the controlled mediator level. You can use the following *med-options* to specify the attributes of the binary levels:

DESCENDING
DESC

reverses the sort order of a binary mediator variable. If both the DESCENDING and ORDER= options are specified, PROC CAUSALMED orders the mediator categories according to the ORDER= option and then reverses that order.

EVENT='*level*' | FIRST | LAST

specifies the event category or level for the binary mediator. The same event level is also assumed for the mediator variable in the outcome model that you specify in the MODEL statement. PROC CAUSALMED models the probability of the event level of the mediator.

The category that is not specified in the EVENT= option is then automatically treated as the reference level (see the REF= option). This reference level is the controlled mediator value for computing controlled direct effects and reference interactions. You cannot specify both the EVENT= and REF= options for the mediator variable.

You can specify one of the following keywords for the EVENT= option:

'*level*' specifies the *level* in quotation marks to be used as the event. Specify the formatted value of the variable if a format is assigned.

FIRST designates the first ordered level as the event.

LAST designates the last ordered level as the event.

By default, EVENT=FIRST.

Consider a situation where the mediator variable M takes the values 1 and 0 for event and nonevent, respectively, and T is the treatment variable. To specify the value 1 as the event category of the mediator variable, use the following MEDIATOR statement:

```
mediator M(event='1') = T;
```

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the mediator variable. The following table displays the available ORDER= options.

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.
FREQ	Descending frequency count. Levels that have the most observations come first in the order.
INTERNAL	Unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

REFERENCE='*level***' | FIRST | LAST**

REF='*level***' | FIRST | LAST**

specifies the reference level for the binary mediator variable. This reference level is also the controlled mediator value for computing controlled direct effects. The same reference level is also assumed for the mediator variable in the outcome model that you specify in the MODEL statement.

The category that is not specified in the REF= option is then automatically treated as the event level (see the EVENT= option). You cannot specify both the EVENT= and REF= options for the mediator variable.

You can specify one of the following keywords for the REFERENCE= option:

'*level***'** specifies the *level* in quotation marks to be used as the reference level. Specify the formatted value of the variable if a format is assigned.

FIRST designates the first ordered level as the reference level.

LAST designates the last ordered level as the reference level.

By default, REF=LAST.

For binary treatment variables, it is important to indicate which level or category represents the treatment or control level. You can use the following *treat-options* to specify the attributes of the binary treatment levels:

CONTROL='*level***' | FIRST | LAST**

REF='*level***' | FIRST | LAST**

specifies the control level for the binary treatment variable. The same control level is also assumed for the treatment variable in the outcome model that you specify in the MODEL statement.

The category that is not specified in the CONTROL= option is then automatically treated as the treatment level of the treatment variable (see the TREAT= option). You cannot specify both the CONTROL= and TREAT= options.

You can specify one of the following keywords for the CONTROL= option:

'*level*' specifies the *level* in quotation marks to be used as the control level. Specify the formatted value of the variable if a format is assigned.

FIRST designates the first ordered level as the control level.

LAST designates the last ordered level as the control level.

By default, CONTROL=LAST.

DESCENDING
DESC

reverses the sort order of a binary treatment variable. If both the DESCENDING and ORDER= options are specified, PROC CAUSALMED orders the treatment categories according to the ORDER= option and then reverses that order.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the treatment variable. The following table displays the available ORDER= options.

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.
FREQ	Descending frequency count. Levels that have the most observations come first in the order.
INTERNAL	Unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

TREAT='*level*' | FIRST | LAST

specifies the treatment level of the treatment variable. The same treatment level is also assumed for the treatment variable in the outcome model that you specify in the MODEL statement.

The category that is not specified in the TREAT= option is then automatically treated as the control level (see the CONTROL= option). You cannot specify both the CONTROL= and TREAT= options for the treatment variable.

You can specify one of the following keywords for the TREAT= option:

'*level*' specifies the *level* in quotation marks to be used as the treatment level. Specify the formatted value of the variable if a format is assigned.

FIRST designates the first ordered level as the treatment level.

LAST designates the last ordered level as the treatment level.

By default, TREAT=FIRST.

Consider a situation where the treatment variable T takes the values 1 and 0 for treatment and control, respectively, and M is the mediator variable. To specify the value 1 as the treatment level of the treatment variable, use the following MEDIATOR statement:

```
mediator M = T(treat='1');
```

MODEL Statement

MODEL *outcome* < **censor(list)* > < *(outcome-options)* > **=***effects* < / *model-options* > **;**

The MODEL statement is required for specifying the outcome model. You provide the *outcome* (the name of the outcome variable) to the left of the equal sign and *effects* (the treatment and mediator effects) to the right. If your outcome variable is a time-to-event response, you can specify the right-censoring variable in *censor*. The values in the *list* for the censoring variable indicate right-censored responses. Optionally, you can specify outcome attributes in *outcome-options* and modeling attributes in *model-options*.

Together, the COVAR, MEDIATOR, and MODEL statements specify the relationships of all variables in the mediation analysis. The treatment and mediator variables that you specify in the MODEL statement must be consistent with those that are you specify in the MEDIATOR statement. If there are covariates in the analysis, do not specify them or their effects in the MODEL statement even though the covariate effects on the outcome variable are being modeled. Instead, use the COVAR statement to specify the covariate effects.

Outcome variables can be binary, continuous, count, or time-to-event (or failure/survival time) variables. PROC CAUSALMED does not support outcome variables that are nominal or ordinal and have more than two levels.

Suppose that the outcome variable is Y, the treatment variable is T, and the mediator variable is M. The three possibilities for the syntax of *effects* are as follows:

```
model Y = T M;
model Y = T M T*M;
model Y = T | M;
```

The first statement specifies the effects of T and M but no interaction effect between the two. The second and third statements are equivalent. Both specify the effects of T, M, and their interaction. The order of T and M is not important.

If outcome variable Y is a count response, you can specify either a Poisson (DIST=POISSON) or negative binomial (DIST=NB) distribution in the *model-options*. For example:

```
model Y = T | M  / dist=poisson;
```

If outcome variable Y is a time-to-event response, you can specify either an accelerated failure time model (Kalbfleisch and Prentice 1980) by using the AFT option or a Cox proportional hazards model (Cox 1972) by using the COXPH option. For example:

```
model Y = T | M  / AFT;
```

If outcome variable Y is a time-to-event response and Q is a variable that indicates right-censoring of the response by values 1 or 2, you can use the following statement to specify such a right-censored outcome response:

```
model Y*Q(1 2) = T | M / AFT;
```

If outcome variable Y is a binary variable, you can list it in the CLASS statement. Alternatively, you can specify DIST=BIN in *model-options*. For example, specifying

```
class Y;
model Y = T | M;
```

is equivalent to specifying

```
model Y = T | M / dist=bin;
```

Both specifications model Y as a binary response with a Bernoulli distribution (that is, a binomial distribution with a single trial).

For binary outcomes, it is important to indicate which level or category represents the events being modeled. You can use the following *outcome-options* to specify the attributes of the binary levels:

DESCENDING
DESC

reverses the sort order of a binary outcome variable. If both the DESCENDING and ORDER= options are specified, PROC CAUSALMED orders the outcome categories according to the ORDER= option and then reverses that order.

EVENT='*level***' | FIRST | LAST**

specifies the event category or level for the binary outcome. PROC CAUSALMED models the probability of the event of the outcome. The category that is not specified in the EVENT= option is then automatically treated as the reference level (see the REF= option). You cannot specify both the EVENT= and REF= options for the outcome variable.

You can specify one of the following keywords for the EVENT= option:

'*level***'** specifies the *level* in quotation marks to be used as the event. Specify the formatted value of the variable if a format is assigned.

FIRST designates the first ordered level as the event.

LAST designates the last ordered level as the event.

By default, EVENT=FIRST.

One of the most common sets of outcome levels is {'No','Yes'}, where 'Yes' represents the event whose probability is modeled. To specify this event level for the outcome variable Y, use the following MODEL statement:

```
model Y(event='Yes') = T | M;
```

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the outcome variable. The following table displays the available ORDER= options.

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.
FREQ	Descending frequency count. Levels that have the most observations come first in the order.
INTERNAL	Unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

REFERENCE='*level*' | FIRST | LAST

REF='*level*' | FIRST | LAST

specifies the reference level for the binary outcome variable.

You can specify one of the following keywords for the REF= option:

'*level*' specifies the *level* in quotation marks to be used as the reference level. Specify the formatted value of the variable if a format is assigned.

FIRST designates the first ordered level as the reference level.

LAST designates the last ordered level as the reference level.

By default, REF=LAST.

PROC CAUSALMED supports two main classes of outcome models: generalized linear models and failure time (survival) models. You can specify the related modeling attributes by providing the following *model-options* after the slash (/):

AFT

fits an accelerated failure time model to time-to-event outcomes.

COXPH

fits a Cox proportional hazards model to time-to-event outcomes.

DIST=*keyword*

DISTRIBUTION=*keyword*

specifies the built-in probability distribution to use in the model.

For generalized linear models, the valid *keywords* for this option are described in Table 38.3. If you specify these distributions and you omit the LINK= option, a default link function is chosen as displayed in Table 38.3.

Table 38.3 Distributions and Default Link Functions

DIST=	Distribution	Default Link Function
BIN | **B**	Binary	Logit
NEGBIN | **NB**	Negative binomial	Log
NORMAL | **NOR** | **N**	Normal	Identity
POISSON | **POI** | **P**	Poisson	Log

If you specify neither the DIST= option nor the LINK= option for a generalized linear model, then the CAUSALMED procedure defaults to the binary distribution with logit link if the outcome variable is listed in the CLASS statement. If the outcome variable is not listed in the CLASS statement, then the CAUSALMED procedure defaults to the normal distribution with the identity link function.

For the Poisson and negative binomial distributions, responses must be nonnegative, but they can take noninteger values. Observations whose response values are outside of the distribution's support are not used to estimate the mediation effects.

If you specify an accelerated failure time model by using the AFT option in the *model-options*, you can specify the following *keywords* for the distribution of the time-to-event response:

EXP | **EXPONENTIAL** specifies an exponential distribution, which is treated as a restricted Weibull distribution.

GAMMA specifies a generalized gamma distribution (Lawless 2003, p. 240). The standard two-parameter gamma distribution is not available in PROC CAUSALMED.

LOGISTIC specifies a logistic distribution.

NORMAL specifies a normal distribution.

WEIBULL specifies a Weibull distribution. If the NOLOG option is also specified, PROC CAUSALMED fits a type 1 extreme-value distribution to the raw, untransformed data.

By default, DIST=WEIBULL for accelerated failure time models.

LINK=*keyword*

specifies the link function in the model. You can specify the *keywords* shown in Table 38.4.

Table 38.4 Built-In Link Functions of the CAUSALMED Procedure

LINK=	Link Link Function	$g(\mu) =$
IDENTITY | ID	Identity	μ
LOG	Log	$\log(\mu)$
LOGIT	Logit	$\log(\mu/(1-\mu))$

By default, the link function is chosen as shown in Table 38.3. This option does not apply to accelerated failure time and Cox proportional hazards models.

NOLOG

requests that no log transformation of the time-to-event outcome response be performed to fit the accelerated failure time model. This option is irrelevant if you do not specify the accelerated failure time model with the AFT option.

By default, PROC CAUSALMED transforms the time-to-event outcome response with the natural logarithm before fitting the accelerated failure time model if you specify DIST=EXP, GAMMA, or WEIBULL. The NOLOG option thus suppresses the log transformation for these three distributions. For DIST=NORMAL or LOGISTIC, log transformation is not applied whether you specify the NOLOG option or not.

If log transformation is applied to an accelerated failure time model, the causal mediation effects are expressed as ratios of the relevant expected event (failure) times. If log transformation is not applied to an accelerated failure time model, the causal mediation effects are expressed as differences between the relevant expected event (failure) times.

STD Statement

STD *variable*=*value* < *variable*=*value* . . . > **;**

You can use this statement to specify the standard deviations of the treatment variable, mediator variable, and covariates. The *values* for standard deviations must be nonnegative.

For example, the following statement specifies the standard deviations of the variables x1 and x2:

```
std  x1=2.3 x2=4.1;
```

The STD statement is not required. When the standard deviations of variables are used in evaluating specific causal mediation effects, PROC CAUSALMED can compute the standard deviations from the input raw data automatically. When you specify the STD statement, it overwrites those computed values of standard deviations. Respecifying standard deviations of variables can affect the evaluation of causal mediation effects only when both of the following conditions are true:

1. The treatment variable, mediator variable, or covariate that you specify in the STD statement is a continuous variable.

2. The standard deviation of the specified treatment variable, mediator variable, or covariate serves as the scale of the respective variable level that is used to evaluate causal mediation effects.

In other words, specifying the STD statement is effective only when you use the **SD** keyword to define the level of a continuous treatment variable, mediator variable, or covariate for computing specific sets of causal mediation effects that are defined in the EVALUATE statement.

A practical situation in which you might want to use the STD statement is when you input sampling weights by using the WEIGHT statement. PROC CAUSALMED computes the weighted standard deviation by a formula that is consistent with the formula used in PROC MEANS. However, this formula might not be appropriate in every instance. In such cases, you can compute your own standard deviations by using the appropriate formulas and input them into the procedure by using the STD statement.

For more information about the default weighted standard deviation formula that the CAUSALMED procedure uses, see the WEIGHT statement.

WEIGHT Statement

WEIGHT *variable* **;**

If you want to use relative weights for each observation in the input data set, then specify a variable that contains weights in a WEIGHT statement. This is often done when the variance that is associated with each observation is different and the values of the weight variable are proportional to the reciprocals of the variances.

If an observation has a negative or missing weight, it is excluded from the analysis. Otherwise, observations that have nonnegative weights are included in the analysis. The weights affect the computations in two ways. One way is in the computation of means and standard deviations of continuous variables. The other way is in the fitting of generalized linear models for the mediator and outcome variables.

The computation of weighted means and standard deviations by PROC CAUSALMED is consistent with that of PROC MEANS when nonpositive weights are excluded (that is, by using the EXCLNPWGT option in PROC MEANS). For nonnegative weight values w_i, the formulas for computing the weighted mean and standard deviation are

$$\begin{aligned} \bar{x}_w &= \sum_{i=1}^{n} w_i x_i \Big/ \sum_{i=1}^{n} w_i \\ \hat{\sigma}_w &= \sqrt{\sum_{i=1}^{n} w_i (x_i - \bar{x}_w)^2 / d} \end{aligned}$$

where x_i is a variable value, n is the total number of observations, and d is the variance divisor that is defined by the VARDEF= option. By default, VARDEF=DF, so d is $n - 1$.

The weights are also used in fitting generalized linear models. For more information about how the weights are used in model fitting, see the WEIGHT statement in Chapter 51, "The GENMOD Procedure."

The WEIGHT statement is ignored if you specify the COXPH option in the MODEL statement.

Details: CAUSALMED Procedure

Causal Mediation Effects: Theory, Definitions, and Effect Decompositions

This section describes the theoretical foundations of the CAUSALMED procedure. It defines the causal mediation and related effects that the procedure estimates, and it describes various types of effect decompositions. The section "Causal Mediation Effects: Assumptions, Identification, and Estimation" on page 2462 continues the discussion by laying out the assumptions for identifying and estimating causal mediation effects.

In any causal mediation analysis, there are four main types of variables of interest:

- an outcome variable Y
- a treatment variable T that is hypothesized to have direct and indirect causal effects on the outcome variable Y (in epidemiology, a treatment variable is also known as an exposure, denoted as A)
- a mediator variable M that is hypothesized to be causally affected by the treatment variable T and that itself has a direct effect on the outcome variable Y
- a set of pretreatment or background covariates that confound the observed relationships among Y, T, and M

Figure 38.8 represents the first three variables in a causal diagram. A causal diagram depicts the causal relationships of variables in an intuitive way. For a general theory of causal diagrams, see Pearl (2009). The role of the background covariates C is discussed after the causal diagram is interpreted.

Figure 38.8 A Causal Mediation Model

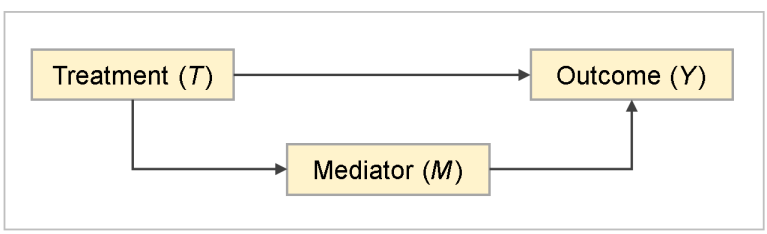

Figure 38.8 shows two causal pathways that represent the effect of T on Y:

- A direct pathway: $T \rightarrow Y$
- A mediated or indirect pathway: $T \rightarrow M \rightarrow Y$

The first causal pathway generates the *direct* effect of T on Y, and the second pathway generates the *indirect* effect of T on Y.

Suppose that Y, T, and M are all continuous variables. If you ignore the causal pathways and regress Y on T by using a linear model of the form

$$Y = \gamma_0 + \gamma_1 T + e$$

where e is an error term that has an expected value of 0 and γ_0 is an intercept, then γ_1 is referred to as the *total* effect of T on Y. This total effect is the overall effect of T on Y without referring to a particular pathway.

When you hypothesize a causal diagram such as Figure 38.8, the relationships among Y, T, and M are described by two linear equations,

$$\begin{aligned} M &= \beta_0 + \beta_1 T + \epsilon \\ Y &= \theta_0 + \theta_1 T + \theta_2 M + \delta \end{aligned}$$

where ϵ and δ are error terms that have expected values of 0, and the parameters of these two equations are as follows:

- β_0 is the intercept of the equation for predicting M.
- θ_0 is the intercept of the equation for predicting Y.
- β_1 is the effect of the $T \rightarrow M$ path.
- θ_1 is the effect of the $T \rightarrow Y$ path.
- θ_2 is the effect of the $M \rightarrow Y$ path.

Substituting the equation for predicting M into that for predicting Y, you have

$$Y = (\theta_0 + \theta_2\beta_0) + (\theta_1 + \theta_2\beta_1)T + (\theta_2\epsilon + \delta)$$

Comparing this equation with the regression equation that predicts Y by T ignoring the causal pathways, you have the equality

$$\gamma_1 = \theta_1 + \beta_1\theta_2$$

where the two terms on the right side of the equation represent additive components of the total effect γ_1, assuming that the causal diagram and the corresponding linear equations are true.

Because the first component θ_1 represents the direct effect of the $T \rightarrow Y$ path, the second component $\theta_2\beta_1$ thus represents the effect of T on Y that is not direct, or simply the indirect effect of T on Y. You can also interpret this indirect effect ($\beta_1\theta_2$) intuitively—it is the product of the two path effects along the indirect pathway $T \rightarrow M \rightarrow Y$.

Therefore, conceptually, the total effect decomposition can be written as follows:

$$\text{total effect} = \text{direct effect} + \text{indirect effect}$$

The direct and indirect effect components are also well defined by the parameters in linear models for continuous Y, T, and M. For an illustration, see Example 38.4.

However, the illustration of the total effect decomposition has been quite ad hoc in nature. It is based on comparing linear models for continuous variables without prior definitions of direct and indirect effects. Consequently, for nonlinear models or linear models that have interaction effects between T and M, the preceding strategy would not work. One reason is that there could be more than two terms in the decomposition so that the direct-indirect decomposition is ambiguous. Another reason is that the terms become much more complicated in nonlinear models, and how to obtain those direct-indirect components would not be clear.

In contrast, the counterfactual framework addresses this issue by offering clear definitions of direct and indirect effects that are applicable to linear and nonlinear models with or without interaction effects. The next section describes this framework.

Another limitation of the illustration that is based solely on the diagram in Figure 38.8 is that it does not deal adequately with pretreatment characteristics or covariates C in observational studies. Typically, covariates C functions like common causes among Y, T, and M in a causal diagram. In observational studies, the observed associations or relationships among Y, T, and M are attributed to two parts. One part is the actual causal effects among them (that is, the effects that are due to the previously mentioned direct and indirect causal pathways). The other part is their induced associations by C. This part of induced association is often called confounding associations or effects. To obtain unbiased estimates of causal mediation and related effects in observational studies, statistical methods must be able to "remove" the confounding associations. More specifically, the covariates C must suffice to control for confounding of the treatment-outcome, mediator-outcome, and treatment-mediator relationship.

Before a discussion of such statistical methods, a more fundamental issue needs to be addressed: Under what conditions can causal mediation effects be identified? Only after the identification conditions are satisfied can you then attempt to obtain unbiased estimation of causal mediation and related effects. The identification issue is addressed in the section "Identification of Causal Mediation Effects" on page 2463 after the counterfactual framework is described in the next section. Regression adjustment methods that are based on the identification conditions are then presented in the section "Regression Methods for Causal Mediation Analysis" on page 2464.

Counterfactual Framework for Defining Causal Mediation Effects

Mediation analysis has a relatively long history in the field of psychology. Almost all recent developments in the area of causal mediation analysis trace back to the psychological tradition of mediation analysis, as typified by Baron and Kenny (1986). The preceding section illustrates such a traditional approach.

However, as discussed in the preceding section, a problem of the traditional approach is that it lacks a general framework that offers clear definitions of causal mediation and related effects. As a result, the traditional approach cannot deal with interaction effects effectively and it cannot treat binary outcomes and binary mediators in a unified framework.

The counterfactual framework (Robins and Greenland 1992; Pearl 2001) offers a solution to this problem. Within this framework, direct and indirect effects are well defined in terms of counterfactual outcomes. Using these definitions, VanderWeele and Vansteelandt (2009) and VanderWeele and Vansteelandt (2010) derived analytic results for computing causal mediation effects under a wide class of parametric models for various types of treatment and outcome variables. Valeri and VanderWeele (2013) extended these results to binary mediators and count outcomes. VanderWeele (2011) and Valeri and VanderWeele (2015) derived analytic results for analyzing time-to-event (survival) data. This line of development provides the theoretical foundations of the CAUSALMED procedure.

A counterfactual outcome is the outcome that you would observe under a hypothetical intervention that you can set the treatment T to particular level t. Counterfactual outcomes, which are also called potential outcomes by some researchers, are therefore defined for scenarios that might be contrary to the factual outcomes. In the counterfactual framework for causal mediation analysis, interventions on the mediator level are also used in various hypothetical scenarios for defining mediation effects.

The following notation is used for counterfactual outcomes that depend on interventions:

- Y_t is the counterfactual outcome of Y for a subject when an intervention sets the treatment level to $T = t$.
- M_t is the counterfactual outcome of M for a subject when an intervention sets the treatment level to $T = t$.
- Y_{tm} is the counterfactual outcome of Y for a subject when an intervention sets the treatment level to $T = t$ and $M = m$.

This notation places no restriction on variable types. The variables Y, T, and M can be continuous or binary.

Suppose for the moment that the treatment is binary so that t is either 0 or 1, denoting the control (no treatment) and treatment conditions, respectively. The total effect (TE) for a subject is defined as the difference between the counterfactual outcomes at the treatment and control levels:

$$\text{TE} = Y_{1M_1} - Y_{0M_0}$$

In this equation, the first subscript in the counterfactual outcomes denotes the intervention of the treatment (either at 1 or 0), and the second subscript denotes the mediator value that would follow from the intervention of the treatment (either M_1 or M_0).

The controlled direct effect (CDE) for a subject is defined as the difference between the counterfactual outcomes at the two treatment levels when an intervention sets the mediator to a particular level $M = m$. That is,

$$\text{CDE}(m) = Y_{1m} - Y_{0m}$$

The natural direct effect (NDE) for a subject is defined as the difference between the counterfactual outcomes at the two treatment levels when an intervention sets the mediator value to $M = M_0$, which is the natural level of the mediator when there is no treatment. That is,

$$\text{NDE} = Y_{1M_0} - Y_{0M_0}$$

The natural indirect effect (NIE) for a subject is defined as the difference between the counterfactual outcomes at the two mediator levels at M_1 and M_0 when an intervention sets the treatment to $T = 1$. That is,

$$\text{NIE} = Y_{1M_1} - Y_{1M_0}$$

All the preceding definitions assume that the treatment variable T is binary. If the treatment variable is continuous, then the treatment levels must be defined according to the treatment and control levels of interest.

For example, if t_1 and t_0 are the treatment and control levels on a continuous scale and they represent the levels of substantive interest, they should replace the 1 and 0 values, respectively, for the treatment and control levels in the definitions. However, this more general notation is not used here because it would make the presentation unnecessarily complicated.

These definitions have two important properties. First, they lead to the following conventional two-way decomposition of the total effect (TE):

$$\text{TE} = \text{NDE} + \text{NIE}$$

Second, these definitions are independent of the models for the outcome or mediator. Hence, these definitions and the total effect decomposition are applicable to linear or nonlinear models, with or without an interaction effect between T and M.

The percentage of total effect that is mediated (PM) is computed as

$$\text{NIE}/\text{TE} * 100\%$$

VanderWeele (2014) took a step further and introduces the following four-way decomposition of the total effect:

$$\text{TE} = \text{CDE} + \text{IRF} + \text{IMD} + \text{PIE}$$

The component effects in this equation are called the controlled direct effect, the reference interaction, the mediated interaction, and the pure indirect effect, respectively. In VanderWeele (2014), IRF is denoted as INT_{ref} and IMD is denoted as INT_{med}. These four component effects are also defined in terms of counterfactual outcomes. For definitions, see VanderWeele (2014).

The significance of these components in causal mediation analysis is that they characterize interaction and mediation effects as follows:

- CDE (controlled direct effect) is the component effect that is not due to interaction or mediation.
- IRF (reference interaction) is the component effect that is due to interaction but not mediation.
- IMD (mediated interaction) is the component effect that is due to both interaction and mediation.
- PIE (pure indirect effect) is the component effect that is due to mediation but not interaction.

Dividing each of these component effects by the total effect yields the corresponding proportion contributions of these components. However, these contributions are not interpretable when the components effects have mixed signs.

Some important relationships between the two-way decomposition and the four-way decomposition are expressed by the following equations:

$$\text{NDE} = \text{CDE} + \text{IRF}$$
$$\text{NIE} = \text{PIE} + \text{IMD}$$

The first equation expresses the natural direct effect (NDE) as the composite component of the controlled direct effect and reference interaction. The second equation expresses the mediation effect or natural indirect effect (NIE) as the composite component of the pure indirect effect and mediated interaction.

Another useful composite component of the four-way decomposition is the "portion attributed to interaction," which is defined as

$$\text{PAI} = \text{IRF} + \text{IMD}$$

As its name suggests, this is the portion of the total effect that is due to the interaction between T and M. The percentage of total effect that is due to the interaction is therefore computed as

$$\text{PAI}/\text{TE} * 100\%$$

VanderWeele (2014) discusses various two-way and three-way decompositions and their relationships with the four-way decomposition. He also offers interesting interpretations and applications of these decompositions. For any causal mediation analysis, you can use the DECOMP option in the CAUSALMED procedure to obtain several two-way decompositions, several three-way decompositions, and the four-way decomposition. For more information about the decompositions, see the DECOMP option.

Causal Mediation Estimands

The section "Counterfactual Framework for Defining Causal Mediation Effects" on page 2457 defines causal mediation and related effects as differences between various counterfactual outcomes for an individual. The presentation is useful for explaining basic concepts. In practice, however, individual causal mediation effects are seldom the target estimands (or population parameters) of interest. This section provides details about the causal mediation estimands. The section "Estimands on Difference Scale" on page 2460 describes estimands on difference scales, and the section "Estimands on Ratio Scale" on page 2461 describes estimands on ratio scales.

Estimands on Difference Scale

This section provides details about causal estimands on difference scales. In general, for the regression approach, the total effect estimand can be expressed as the population mean difference between the following two population counterfactual outcomes:

$$\mathcal{E}[(Y_{1M_1} - Y_{0M_0}) \mid C] = \mathcal{E}[Y_{1M_1} \mid C] - \mathcal{E}[Y_{0M_0} \mid C]$$

where $\mathcal{E}[\cdot]$ is an expectation operation and $\mathcal{E}[\cdot \mid C]$ denotes an expectation conditional on the values of covariates C. Let $\mu_c = \mathcal{E}[C]$ be the population mean of C. In all default estimation, PROC CAUSALMED uses μ_c in place of C to define the causal mediation estimands. That is, the *overall* total effect estimand on the difference scale is denoted as

$$\mathrm{TE}_{\mathrm{pop}} = \mathcal{E}[(Y_{1M_1} - Y_{0M_0}) \mid \mu_c]$$

It is important to note that this overall total effect is generally not the same as the marginal total effect, which is defined as

$$\mathcal{E}[\mathcal{E}[(Y_{1M_1} - Y_{0M_0}) \mid C]] = \mathcal{E}[Y_{1M_1}] - \mathcal{E}[Y_{0M_0}]$$

However, if the outcome regression model for Y is linear in C, the expectation operator can "pass through" the formula so that the marginal and overall total effects are equivalent.

In summary, for causal mediation effects that are defined on the difference scale, PROC CAUSALMED has the following main default causal estimands:

$$\begin{aligned}
\mathrm{TE}_{\mathrm{pop}} &= \mathcal{E}[Y_{1M_1} \mid \mu_c] - \mathcal{E}[Y_{0M_0} \mid \mu_c] \\
\mathrm{CDE}_{\mathrm{pop}}(m) &= \mathcal{E}[Y_{1m} \mid \mu_c] - \mathcal{E}[Y_{0m} \mid \mu_c] \\
\mathrm{NDE}_{\mathrm{pop}} &= \mathcal{E}[Y_{1M_0} \mid \mu_c] - \mathcal{E}[Y_{0M_0} \mid \mu_c] \\
\mathrm{NIE}_{\mathrm{pop}} &= \mathcal{E}[Y_{1M_1} \mid \mu_c] - \mathcal{E}[Y_{1M_0} \mid \mu_c]
\end{aligned}$$

The total effect decomposition into direct and indirect effects would still have the same additive form as that of the individual total effect decomposition (see the section "Counterfactual Framework for Defining Causal Mediation Effects" on page 2457). That is,

$$\begin{aligned}
\mathrm{TE}_{\mathrm{pop}} &= \mathrm{NDE}_{\mathrm{pop}} + \mathrm{NIE}_{\mathrm{pop}} \\
\text{Percentage Mediated} &= \mathrm{NIE}_{\mathrm{pop}}/\mathrm{TE}_{\mathrm{pop}} * 100\%
\end{aligned}$$

Similarly, all formulas for other decompositions that are described in the section "Counterfactual Framework for Defining Causal Mediation Effects" on page 2457 for individual causal mediation effects apply to the causal estimands that are expressed on the difference scale in this section.

PROC CAUSALMED estimates mean difference causal mediation effects when Y is a continuous outcome that is fitted by a linear model, or when Y is a time-to-event outcome that is fitted by an accelerated failure time model that does not use the log transformation (see the NOLOG option for more information).

In addition to these default mean difference causal mediation estimands, you can estimate similar mean difference effects conditional on particular covariate values for C. For more information about specifying these covariate values and evaluating conditional causal mediation effects, see the EVALUATE statement and the section "Evaluating Causal Mediation Effects" on page 2466.

Estimands on Ratio Scale

For Y being fitted by nonlinear outcome models, causal mediation estimands are defined on the ratio scale. This section describes a variety of ratio scale estimands.

For binary outcomes that are fitted by the generalized linear model with a logit link, PROC CAUSALMED has the following main odds ratio estimands:

$$\begin{aligned}
\mathrm{TE}_{\mathrm{pop}} &= \frac{\mathcal{E}[Y_{1M_1} \mid \mu_c]/(1-\mathcal{E}[Y_{1M_1} \mid \mu_c])}{\mathcal{E}[Y_{0M_0} \mid \mu_c]/(1-\mathcal{E}[Y_{0M_0} \mid \mu_c])} \\
\mathrm{CDE}_{\mathrm{pop}}(m) &= \frac{\mathcal{E}[Y_{1m} \mid \mu_c]/(1-\mathcal{E}[Y_{1m} \mid \mu_c])}{\mathcal{E}[Y_{0m} \mid \mu_c]/(1-\mathcal{E}[Y_{0m} \mid \mu_c])} \\
\mathrm{NDE}_{\mathrm{pop}} &= \frac{\mathcal{E}[Y_{1M_0} \mid \mu_c]/(1-\mathcal{E}[Y_{1M_0} \mid \mu_c])}{\mathcal{E}[Y_{0M_0} \mid \mu_c]/(1-\mathcal{E}[Y_{0M_0} \mid \mu_c])} \\
\mathrm{NIE}_{\mathrm{pop}} &= \frac{\mathcal{E}[Y_{1M_1} \mid \mu_c]/(1-\mathcal{E}[Y_{1M_1} \mid \mu_c])}{\mathcal{E}[Y_{1M_0} \mid \mu_c]/(1-\mathcal{E}[Y_{1M_0} \mid \mu_c])}
\end{aligned}$$

where $\mathcal{E}[\cdot]$ is an expectation operation. With the rare outcome assumption (for example, see Valeri and VanderWeele 2013), these odds ratio effect are approximated by their corresponding risk ratio effects. Although these formulas are different from those for defining mean difference mediation effects (see the section "Estimands on Difference Scale" on page 2460), both sets of formulas involve the same set of comparisons among counterfactual outcomes conditional on the population mean of C.

For binary outcomes that are fitted by the generalized linear model with the log link, count outcomes that are fitted by the generalized linear model with a Poisson or negative binomial distribution, and time-to-event outcomes that are fitted by the accelerated failure time models, PROC CAUSALMED has the following main risk or mean ratio estimands:

$$\begin{aligned}
\mathrm{TE}_{\mathrm{pop}} &= \frac{\mathcal{E}[Y_{1M_1} \mid \mu_c]}{\mathcal{E}[Y_{0M_0} \mid \mu_c]} \\
\mathrm{CDE}_{\mathrm{pop}}(m) &= \frac{\mathcal{E}[Y_{1m} \mid \mu_c]}{\mathcal{E}[Y_{0m} \mid \mu_c]} \\
\mathrm{NDE}_{\mathrm{pop}} &= \frac{\mathcal{E}[Y_{1M_0} \mid \mu_c]}{\mathcal{E}[Y_{0M_0} \mid \mu_c]} \\
\mathrm{NIE}_{\mathrm{pop}} &= \frac{\mathcal{E}[Y_{1M_1} \mid \mu_c]}{\mathcal{E}[Y_{1M_0} \mid \mu_c]}
\end{aligned}$$

Finally, for time-to-event outcomes that are fitted by the Cox proportional hazards model, PROC

CAUSALMED has the following main hazard ratio estimands:

$$\mathrm{TE}_{\mathrm{pop}} = \frac{\lambda_{Y_1 M_1}(y \mid \mu_c)}{\lambda_{Y_0 M_0}(y \mid \mu_c)}$$

$$\mathrm{CDE}_{\mathrm{pop}}(m) = \frac{\lambda_{Y_1 m}(y \mid \mu_c)}{\lambda_{Y_0 m}(y \mid \mu_c)}$$

$$\mathrm{NDE}_{\mathrm{pop}} = \frac{\lambda_{Y_1 M_0}(y \mid \mu_c)}{\lambda_{Y_0 M_0}(y \mid \mu_c)}$$

$$\mathrm{NIE}_{\mathrm{pop}} = \frac{\lambda_{Y_1 M_1}(y \mid \mu_c)}{\lambda_{Y_1 M_0}(y \mid \mu_c)}$$

where $\lambda_Y(y \mid C)$ denotes the hazard conditional on covariates C at time y.

Again, although mean ratio and hazard ratio effects use different formulas from those for defining mean difference mediation effects (see the section "Estimands on Difference Scale" on page 2460), both sets of formulas involve the same set of comparisons among counterfactual outcomes conditional on the population mean of C.

Common to all these ratio scale estimands, the total effect decomposition into direct and indirect effects is not additive, but multiplicative. That is,

$$\mathrm{TE}_{\mathrm{pop}} = \mathrm{NDE}_{\mathrm{pop}} \times \mathrm{NIE}_{\mathrm{pop}}$$

The formulas for computing the corresponding percentage mediated, percentage due to interaction, and various decompositions of mean ratio effects are based on the components of the excess of total effect ratio (that is, $\mathrm{TE}_{\mathrm{pop}} - 1$) rather than those of the total effect ratio itself (that is, $\mathrm{TE}_{\mathrm{pop}}$). Formulas for different cases are quite complicated and therefore are not shown here (but see, for example, VanderWeele 2014).

In addition to these default odds/risk/mean ratio causal mediation estimands, you can estimate similar odds/risk/mean ratio effects conditional on particular covariate values for C. For more information about specifying these covariate values and evaluating conditional causal mediation effects, see the EVALUATE statement and the section "Evaluating Causal Mediation Effects" on page 2466.

Causal Mediation Effects: Assumptions, Identification, and Estimation

This section continues the discussion of the theoretical foundations of the CAUSALMED procedure. The section "Causal Mediation Effects: Theory, Definitions, and Effect Decompositions" on page 2455 defines the causal mediation effects and various total effect decompositions that the procedure estimates. This section describes the assumptions that underlie the identification and estimation of causal mediation effects. It also discusses the implications of these assumptions for valid applications of the procedure.

Identification of Causal Mediation Effects

This section lays out the identification conditions of causal mediation effects and their implications for applying statistical methods that aim to obtain unbiased estimation of the effects.

First, it is useful to distinguish the following three types of confounding covariates:

- C_1 represents a generic covariate that confounds the relationship between T and Y. This is a treatment-outcome confounder.
- C_2 represents a generic covariate that confounds the relationship between M and Y. This is a mediator-outcome confounder.
- C_3 represents a generic covariate that confounds the relationship between T and M. This is a treatment-mediator confounder.

As in preceding sections, let C denote all of the covariates C_1, C_2, and C_3. Thus, controlling for C in regression analysis means that all types of confounding covariates are being controlled for.

According to Valeri and VanderWeele (2013), the following four assumptions are required for the identification of causal mediation effects:

- no unmeasured treatment-outcome confounders given C
- no unmeasured mediator-outcome confounders given (C, T)
- no unmeasured treatment-mediator confounders given C
- no mediator-outcome confounder is affected by T (directly or indirectly) given C

The identification of the controlled direct effect (CDE) assumes the first two conditions, and the identification of the natural direct effect (NDE) and the natural indirect effect (NIE) assumes all four conditions. These four assumptions are collectively called the "no unmeasured confounding assumption." Formal statements for these identification conditions can be found in the appendix of Valeri and VanderWeele (2013) and VanderWeele (2015).

Essentially, in order to obtain unbiased estimation of causal mediation and related effects, the regression adjustment method that is discussed in the next section assumes that the identification conditions are satisfied.

In practice, the implication is that in order to have valid causal interpretations of the mediation effects, you must be able to measure all relevant confounding covariates C and include them in a causal mediation analysis. For example, the first identification condition states that there are no unmeasured treatment-outcome confounders given C. Practically, a simple interpretation of this condition is that if there are treatment-outcome confounders C_1 in the observational study, your set of C must have measured and included these confounders in the analysis in order to obtain unbiased estimation of causal effects. Similarly, other identification conditions require C_2 or C_3, if present, to be measured and included in the analysis.

Regression Methods for Causal Mediation Analysis

The CAUSALMED procedure implements regression methods for estimating causal mediation effects that assume the identification conditions of the preceding section along with correct specification of the following two models:

- the outcome model for Y given T, M, and C
- the mediator model for M given T and C

For a class of generalized linear models, VanderWeele and Vansteelandt (2009), VanderWeele and Vansteelandt (2010), VanderWeele (2011), and Valeri and VanderWeele (2013) derived analytic formulas for computing various causal mediation effects for different variable types, including combinations of the following cases:

- outcome variable Y, which can be binary, continuous (including time-to-event outcome), or count
- treatment variable T, which can be binary or continuous
- mediator variable M, which can be binary or continuous
- covariates C, which can be categorical or continuous

PROC CAUSALMED implements these analytic formulas.

In addition to the causal effect identification assumptions that are described in the section "Identification of Causal Mediation Effects" on page 2463, the validity of the formulas assumes that the outcomes are rare events (VanderWeele 2011, 2014) in the following two cases:

- when the outcome Y is binary
- when the outcome Y is a time-to-event variable and is modeled by the proportional hazards model

For the first case, if Y is not rare, then the formulas are still valid if Y is modeled by using the log link.

Let θ represent the vector that collects all parameters in the outcome and mediator models. Under the correct specification of regression models and the identification assumptions, the causal effects in a mediation analysis are functions of θ conditional on the covariate values. That is, a causal effect, which is denoted by *ef*, can be expressed as a function of θ given $C = c$,

$$g_{ef}(\theta \mid C = c),$$

where c represents some fixed values for covariates C. By default, the causal estimands that PROC CAUSALMED targets are of the form $g_{ef}(\theta \mid \mu_c)$, where μ_c is the expected value of C. Therefore, in general $g_{ef}(\theta \mid \mu_c)$ is interpreted as a conditional effect or as an "overall" effect, but not as a marginal effect. It would also be a marginal effect only when $g_{ef}(\theta \mid C)$ is linear in C.

The default estimand, $g_{ef}(\theta \mid \mu_c)$, is consistent with the treatment of the SAS macros that are implemented by Valeri and VanderWeele (2013) and Valeri and VanderWeele (2015). For more information about the definitions of various causal mediation effects, see the sections "Estimands on Difference Scale" on page 2460 and "Estimands on Ratio Scale" on page 2461.

For categorical covariates, μ_c is defined by the expectations of the dummy-coded 0 or 1 value for categorical levels. However, this does not mean that PROC CAUSALMED requires you to dummy-code the categorical covariates for analysis. The dummy coding is done internally in the procedure.

In addition to $g_{ef}(\theta \mid \mu_c)$, PROC CAUSALMED also supports the estimation of $g_{ef}(\theta \mid C = c_0)$, where c_0 represents specific fixed covariate levels that you want to use for evaluating the causal mediation effects—for example, the causal mediation effects for a particular interest group that is defined by some fixed values of c_0. For more information about specifying these covariate values and evaluating conditional causal mediation effects, see the EVALUATE statement and the section "Evaluating Causal Mediation Effects" on page 2466.

Maximum Likelihood Estimation

For random samples, PROC CAUSALMED estimates causal mediation effects by the maximum likelihood method. The maximum likelihood estimate $\hat{\theta}$ of θ is first estimated for the outcome and mediator models. Then the maximum likelihood estimates of various causal mediation effects are computed as

$$g_{ef}(\hat{\theta} \mid C = c_0),$$

where *ef* is the index for effects and $C = c_0$ is the average of the covariate values that are computed from the sample. For categorical covariates, this definition of c_0 assumes that the levels are dummy-coded as 0 and 1, which is done internally by PROC CAUSALMED. Therefore, $g_{ef}(\hat{\theta} \mid C = c_0)$ estimates $g_{ef}(\theta \mid \mu_c)$ in the default output of PROC CAUSALMED.

Given the estimated covariance matrix for $\hat{\theta}$, the delta method is used to estimate the standard errors for the causal effects $g_{ef}(\hat{\theta} \mid C = c_0)$. In the computation of these estimates, the covariate values c_0 are treated as fixed values. For more information about the delta method for computing standard errors in this context, see VanderWeele and Vansteelandt (2009) or VanderWeele (2015).

Alternatively, you can use bootstrap techniques to compute standard error estimates and confidence intervals. For more information about the bootstrap method, see the BOOTSTRAP statement and the section "Bootstrap Methods" on page 2472.

As explained in the preceding sections, the evaluation of causal mediation effects depends on the levels of covariates. In addition to the overall causal mediation effects that are evaluated at $C = c_0$, you can provide particular covariate levels, say $C = c_1$, that are particularly meaningful to your research by using the EVALUATE statement. The maximum likelihood estimate is then $g_{ef}(\hat{\theta} \mid C = c_1)$ and the standard error is computed similarly by the delta method. For more information about evaluating causal mediation effects, see the section "Evaluating Causal Mediation Effects" on page 2466. For an illustration, see Example 38.2.

Estimation of Various Total Effect Decompositions

Formulas for estimating the components of the four-way decomposition of the total effect (VanderWeele 2014) follow essentially the same logic that is described in the preceding sections.

For time-to-event outcomes, the components of the four-way decomposition are computed on the mean time ratio scale when the outcome is modeled by the accelerated failure time model, and they are computed on the hazard ratio scale when the outcome is modeled by the Cox proportional hazards model.

For other continuous outcomes, the components of the four-way decomposition are computed on the original scales. For binary outcomes, the components of the four-way decomposition are computed on the odds ratio scale and the excess relative risk scale. These formulas are quite involved and are not presented here. For more information, see VanderWeele (2014).

In addition to the four-way decomposition, PROC CAUSALMED estimates the component effects of several other two-way and three-way decompositions by using the same analytic technique as that of the four-way decomposition. You can use the DECOMP= option in the EVALUATE or PROC CAUSALMED statement to request these decompositions.

To compute standard error estimates for these component effects and their percentage contribution, PROC CAUSALMED uses the delta method with analytic derivatives. Bootstrap methods are also available for computing standard errors and confidence intervals. For more information about bootstrap estimation, see the BOOTSTRAP statement and the section "Bootstrap Methods" on page 2472.

Evaluating Causal Mediation Effects

In general, the CAUSALMED procedure computes causal mediation effects and decompositions that are conditioned on specific levels of covariates. In addition, some of the causal mediation effects are defined at specific levels (numerical or categorical) of treatment, control, and mediator variables. Therefore, it is important to understand how to set these variable levels for evaluating causal mediation effects that meet your research goals.

This section explains the roles of treatment, control, mediator, and covariate levels in defining and computing causal mediation effects. It shows how you can use the options in the EVALUATE statement to specify these levels, and it describes the default levels that the CAUSALMED procedure uses.

Suppose that T represents a treatment variable that has a causal effect on an outcome variable Y. Furthermore, suppose that M represents a mediator variable, which is affected by T and has a causal effect on Y, and that C represents a generic covariate that confounds the causal treatment and mediation effects.

The roles of the treatment, control, mediator, and covariate levels in defining causal mediation effects are as follows:

- The level t_1 of the treatment variable T is the level that you designate as the treatment condition for all the effects and decompositions that are computed. For example, if the variable T represents the dosage level of a drug, $t_1 = 10$ mg is the dosage level that defines the treatment condition. For a binary treatment variable, researchers usually define t_1 as 1 to represent the presence of the treatment.
- The level t_0 of the treatment variable T is the level that you designate as the reference or control condition for all the effects and decompositions that are computed. For example, if the variable T represents the dosage level of a drug, $t_0 = 5$ mg is the dosage level that defines the control condition. For a binary treatment variable, researchers usually define t_0 as 0 to represent the absence of treatment.
- The level m^* of the mediator variable M is the level that you designate to compute the controlled direct effect. For binary mediator variables, researchers usually define m^* as 0 so that they can evaluate the controlled direct effect by holding the mediator value at the "absence" level.
- The levels c of the covariates are the conditional covariate values in the formulas for computing causal mediation effects.

In general, specifying covariate levels c, the treatment level t_1 (of the treatment variable), or the control level t_0 (of the treatment variable) changes the estimates of all mediation effects and decompositions. Specifying the controlled level m^* of the mediator variable does not change the estimates of the total effect (TE), the

natural direct effect (NDE), or the natural indirect effect (NIE). But it does change the estimates of the controlled direct effect (CDE) and the reference interaction (IRF).

Default Settings of Treatment and Control Levels

For binary treatment variables, PROC CAUSALMED uses the first level of the variable as the default treatment level and the second (last) level of the variable as the default control level. In other words, the first level of a binary treatment variable takes the role of t_1 and the second level takes the role of t_0.

For continuous or ordinal treatment variables, researchers habitually set levels of t_1 and t_0 in such a way that their difference is 1. Such a habitual setting serves well for linear models, including linear regression analysis and linear structural equation modeling. The associated regression coefficient (or effect) is defined as the change in the outcome Y for a unit change in the predictor T. In linear models, the effect on Y depends only on the *difference* between t_1 and t_0 but not on the levels of t_1 and t_0 themselves.

However, with nonlinear models, binary responses, and interaction effects, the computation of causal mediation effects and decompositions does, in general, depend on the levels of t_1 and t_0. Using different sets of t_1 and t_0 (even if their difference remains constant) leads to numerically different estimates of causal mediation effects. By default, PROC CAUSALMED sets the treatment and control levels around the center of the distribution of the treatment variable. That is,

$$t_1 = \bar{t} + 0.5$$
$$t_0 = \bar{t} - 0.5$$

where $\bar{t}$ is the sample mean of the treatment variable. This sample mean value is treated as fixed when computing standard errors.

You can define your own treatment and control levels for evaluating causal mediation effects and decompositions. For example, instead of using a single unstandardized unit as the treatment amount, you can use one standard deviation,

$$t_1 = \bar{t} + 0.5 \times s_t$$
$$t_0 = \bar{t} - 0.5 \times s_t$$

where s_t is the sample standard deviation of the treatment variable. This sample standard deviation is treated as fixed when computing standard errors.

PROC CAUSALMED enables you to set the treatment and control levels either on an unstandardized scale or on a standardized scale. Table 38.5 presents more options for setting these levels.

Default Settings of the Controlled Mediator Level

For binary mediator variables, PROC CAUSALMED uses the second (last) level of the variable as the default controlled (baseline) level, m^*, of the mediator variable. This is consistent with the way that you specify the mediator model in the MEDIATOR statement. That is, by default, the procedure models the probability of the event indicated by the first level of the mediator variable.

For continuous or ordinal mediator variables, PROC CAUSALMED uses the sample mean of M as the default controlled mediator level, m^*, when evaluating causal mediation effects. Table 38.5 presents more options for setting this level.

Covariate Levels and Their Default Settings

When you specify the effects of confounder covariates in the COVAR statement, the CAUSALMED procedure computes mediation effects conditionally at specific levels of the covariates. You can provide one or more EVALUATE statements to request that these effects be computed at specified settings that are of interest in your study. However, whether or not you provide an EVALUATE statement, the CAUSALMED procedure uses the sample means of the covariates to compute "overall" measures of causal mediation effects, which are displayed in the "Summary of Effects" table. For an illustration, see Example 38.2.

Although the means of ordinal and continuous covariates are well defined, less apparent is how to define the mean levels of categorical covariates and any interaction terms that might be included in the model.

To illustrate this, suppose that C1 is a continuous covariate and C2 is a categorical covariate that has three levels: 1, 2, and 3. Also suppose that there are six observations for C1 and C2:

C1	C2
1	1
2	1
3	2
4	2
5	3
6	3

All other variables are not shown.

The following design matrix for the linear predictor contains one column for C1, three columns for C2, and three columns for the interaction of the two variables:

C1	C2			C1 x C2		
1	1	0	0	1	0	0
2	1	0	0	2	0	0
3	0	1	0	0	3	0
4	0	1	0	0	4	0
5	0	0	1	0	0	5
6	0	0	1	0	0	6

The parameterization shown here for C2 is represented internally in PROC CAUSALMED. You are not required to use this coding in the input.

The marginal means of the seven columns are 3.5, 1/3, 1/3, 1/3, 1.5, 3.5, and 5.5, respectively. By default, PROC CAUSALMED substitutes these means for covariate levels in the formulas for computing mediation effects and decompositions.

Substitution of marginal means makes intuitive sense when the models for Y and M are both linear. In this case, the computed causal mediated effects and decompositions can be interpreted as marginal effects. However, causal mediation effects that are computed in this way for nonlinear models (for example, binary responses with logit links) cannot be interpreted as marginal effects. Nonetheless, the default provides "overall" causal mediation effect estimates that are not entirely arbitrary. In a sense, the default method for categorical covariates provides an averaged categorical profile for evaluating causal mediation effects.

The default levels are not the only setting that you can consider. In this example, it would be interesting to conduct three causal mediation analyses, each of which is conditioned on a particular level of C2. You can request these analyses by specifying the following EVALUATE statements:

```
evaluate 'Conditional on Level 1 of C2' C1=mean C2='1';
evaluate 'Conditional on Level 2 of C2' C1=mean C2='2';
evaluate 'Conditional on Level 3 of C2' C1=mean C3='3';
```

Each EVALUATE statement generates a set of mediation analysis results.

In summary, you can use the EVALUATE statement to examine causal mediation effects that are conditional on the covariate levels that you specify. The CAUSALMED procedure displays these effects in the output together with overall effects that are conditioned on default settings; For illustrations, see Example 38.2 and Example 38.3. The next section describes the options for specifying treatment, control, mediator, and covariate levels.

Options for Setting Variables Levels

You use the EVALUATE statement to request the computation of causal mediation effects that are conditional on particular levels of variables. You can set the levels of variables by specifying an *assignment* of the following form:

var-key=value-key

Table 38.5 summarizes the options for *var-key* and *value-key*. The last two columns of Table 38.5 display the default *value-key*.

Table 38.5 EVALUATE Statement Options for Setting Variable Levels

Level	*var-key*	*value-key*		**Default** *value-key*	
		Class Variable	**Count or Continuous Variable**	**Class Variable**	**Count or Continuous Variable**
Treatment					
	_TREATMENT	FIRST	MAX	FIRST	*mean* + 0.5
	_A1	LAST	MEAN		
	_T1	'*level*'	MIN		
	vname(TREATMENT)		*value*		
			value(SD)		
Control					
	_CONTROL	FIRST	MAX	LAST	*mean* – 0.5
	_A0	LAST	MEAN		
	_T0	'*level*'	MIN		
	vname(CONTROL)		*value*		
			value(SD)		
Mediator					
	_MEDIATOR	FIRST	MAX	LAST	*mean*
	_MSTAR	LAST	MEAN		
	vname	'*level*'	MIN		
			value		
			value(SD)		

Table 38.5 *continued*

Variable	*var-key*	*value-key*		**Default** *value-key*	
		Class Variable	**Count or Continuous Variable**	**Class Variable**	**Count or Continuous Variable**
Covariate					
	vname	FIRST LAST MODE '*level*'	MAX MEAN MIN *value* *value*(SD)	*mean* or MODE	*mean*

In this table, *vname* represents an actual variable name, '*level*' represents an actual level of a classification variable, and *value* represents an actual value of a numeric variable. In the last two columns, *mean* represents the sample mean of a continuous variable or the sample mean of a categorical variable (in dummy coding).

To specify an *assignment*, first look for the correct *var-key* in the second column. Different *var-keys* are used for the treatment, control, mediator, and covariate levels. In all cases, you can use the actual variable name of the variable. Next, select one of the *value-keys* in the third or fourth column to specify the desired variable level.

Repeat as many *assignments* as you need to specify the levels of various variables.

For example, suppose that there is a continuous treatment variable Exposure and a binary mediator variable PerceivedPain in your analysis. You identify the roles of these variables by using the following statements:

```
proc causalmed;
  class PerceivedPain;
  mediator PerceivedPain = Exposure;
  model outcome = PerceivedPain | Exposure;
```

To set the treatment level at the maximum sample value, the control level at the mean value, and the mediator at the level encoded as “none,” you can use any of the following equivalent specifications:

```
evaluate 'Setting 1' _t1=max _t0=mean _mstar='none';
evaluate 'Setting 2' _treatment=max _control=mean _mediator='none';
evaluate 'Setting 3' Exposure(treatment)=max Exposure(control)=mean
                     PerceivedPain='none';
```

This example shows that you can specify a *var-key* either directly (by providing an actual variable name) or indirectly (by providing a keyword). Likewise, you can specify an *value-key* either directly (by providing an actual level) or indirectly (by providing a keyword). For a complete description of these options, see the EVALUATE statement.

Note that the default *value-key* for categorical covariates can be either the sample means (denoted as *mean* in the table) or MODE. If you do not assign any levels for categorical covariates in an EVALUATE statement, PROC CAUSALMED uses the sample means as the default levels for all unassigned categorical covariates that are specified in the COVAR statement. For example, the sample means of C1, C2, and C3 are the default levels used in the EVALUATE statement for the following specification:

```
proc causalmed;
  class C1 C2 C3;
  mediator M = T;
  model Y = T | M;
  covar C1 C2 C3 C4;
  evaluate 'Conditional on C4=max' C4=max M=mean;
```

If you assign the level of at least one categorical covariate in an EVALUATE statement, PROC CAUSALMED uses MODE as the default level for the unassigned categorical covariates that are specified in the COVAR statement. For example, the modal levels of C2 and C3 and the sample mean of C4 are the default levels used in the EVALUATE statement for the following specification:

```
proc causalmed;
  class C1 C2 C3;
  mediator M = T;
  model Y = T | M;
  covar C1 C2 C3 C4;
  evaluate 'Conditional on C1=1' C1='1' M=mean;
```

Multimodal Covariates

If you specify MODE as the *value-key* for a categorical covariate and it has multiple modes, an averaging process is used to compute the levels. To illustrate this, suppose that C1 is a continuous covariate and C2 and C3 are binary covariates. Also suppose that there are six observations with the following values for the three covariates:

C1	C2	C3
1	1	1
2	1	1
3	1	1
4	1	2
5	2	2
6	2	2

The design matrix for the linear predictor contains one column for C1 and two columns for each of C2 and C3:

C1	C2		C3	
1	1	0	1	0
2	1	0	1	0
3	1	0	1	0
4	1	0	0	1
5	0	1	0	1
6	0	1	0	1

Suppose you specify the following EVALUATE statement:

```
evaluate 'Setting A' C1=mean C2=mode C3=mode;
```

The mean of C1 is 3.5. The modal class of C2 is '1', and hence the coding '1 0' is used as the covariate level for C2. However, because C3 has two modal classes,'1 0' and '0 1', these two modal class codings are averaged out with other levels. The final coding vector for the covariate levels is then the average of the following two vectors:

```
3.5   1  0  1  0
3.5   1  0  0  1
```

As a result, the averaged levels 3.5, 1, 0, 0.5, and 0.5 are used in the formulas for evaluating causal mediation effects and decompositions.

If an interaction between C1 and C3 is also modeled, then the average of the following two vectors is used:

```
3.5   1  0  1  0  3.5    0
3.5   1  0  0  1    0  3.5
```

Here the last two columns represent the interaction terms. As a result, the averaged levels 3.5, 1, 0, 0.5, 0.5, 1.75, and 1.75 are used in the formulas for evaluating causal mediation effects and decompositions.

Bootstrap Methods

If you specify the BOOTSTRAP statement, PROC CAUSALMED uses bootstrap resampling to compute standard errors and confidence intervals for causal mediation effects and decompositions. The procedure samples as many bootstrap sample data sets (replicates) as you specify in the NBOOT= option and then estimates the effects and decompositions for each replication.

Bootstrap confidence intervals are computed only for the effects and their corresponding percentages. These intervals are not computed for the parameters in the outcome or mediator models. You can specify one or more of the following types of bootstrap confidence intervals by using the BOOTCI option in the BOOTSTRAP statement:

- The BOOTCI(NORMAL) option requests bootstrap confidence intervals that are based on the normal approximation method. The $(1-\alpha)100\%$ normal bootstrap confidence interval is given by

  $$\hat{\mu}_j \pm \sigma_{\mu_j^*} \times z_{(1-\alpha/2)}$$

  where $\hat{\mu}_j$ is the estimate of μ_j from the original sample, $\sigma_{\mu_j^*}$ is the standard deviation of the bootstrap parameter estimates, and $z_{(1-\alpha/2)}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution.

- The BOOTCI(PERC) option requests bootstrap confidence intervals that are based on the percentile method. The confidence limits are the $100(\alpha/2)$th and $100(1-\alpha/2)$th percentiles of the bootstrap parameter estimates, which are computed as follows. Let $\mu_{j,1}^*, \mu_{j,2}^*, \ldots, \mu_{j,B}^*$ represent the ordered values of the bootstrap estimates for the potential outcome mean μ_j. Let the kth weighted average percentile be q, set $p = \frac{k}{100}$, and let

  $$np = l + g$$

where l is the integer part of np and g is the fractional part of np. Then the kth percentile, q, is computed as follows, which corresponds to the default percentile definition used by the UNIVARIATE procedure:

$$q = \begin{cases} \frac{1}{2}\left(\mu^*_{j,l} + \mu^*_{j,l+1}\right) & \text{if} \quad g = 0 \\ \mu^*_{j,l+1} & \text{if} \quad g > 0 \end{cases}$$

- The BOOTCI(BC) option requests bias-corrected bootstrap confidence intervals, which use the cumulative distribution function (CDF), $G(\mu^*)$, of the bootstrap parameter estimates to determine the upper and lower endpoints of the confidence interval. The bias-corrected bootstrap confidence interval is given by

  $$G^{-1}\left(\Phi(2z_0 \pm z_{\alpha/2})\right)$$

  where Φ is the standard normal CDF, $z_{\alpha/2} = \Phi^{-1}(\alpha/2)$, and z_0 is a bias correction,

  $$z_0 = \Phi^{-1}\left(\frac{\mathrm{N}\left(\mu^*_j \le \hat{\mu}_j\right)}{\mathrm{B}}\right)$$

  where $\hat{\mu}_j$ is the original sample estimate of μ_j from the input data set, $\mathrm{N}(\mu^*_j \le \hat{\mu}_j)$ is the number of bootstrap estimates (μ^*_j) that are less than or equal to $\hat{\mu}_j$, and B is the number of bootstrap replicates for which an estimate for the treatment effect is obtained.

Bias-corrected bootstrap confidence intervals are the default.

PROC CAUSALMED requires at least 50 bootstrap samples for normal bootstrap confidence intervals and does not compute them if fewer than 40 of the samples produce usable estimates. The procedure requires at least 1,000 bootstrap samples for percentile and bias-corrected bootstrap confidence intervals and does not compute them if fewer than 900 of the samples produce usable estimates. If the number of samples *n* specified in the NBOOT=*n* option is less than 1,000 and percentile or bias-corrected bootstrap confidence intervals are requested, the value of *n* is ignored.

ODS Table Names

PROC CAUSALMED assigns a name to each table it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 38.6. The options or specifications of the specific statements that produce these output tables are shown in the last two columns. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 38.6 ODS Tables Produced by the CAUSALMED Procedure

ODS Table Name	Description	Statement	Option or Specification
BootstrapSamples	Information about bootstrap samples	BOOTSTRAP	Default
ClassLevels	Classification variable levels	CLASS	Classification variables
EffectDecomp	Decompositions of the total effect	PROC CAUSALMED	DECOMP
EffectSummary	Summary of the direct and mediated effects		Default
MediatorEstimates	Parameter estimates for the mediator model	PROC CAUSALMED	PMEDMOD
MediatorProfile	Frequency counts for a binary mediator variable	CLASS	Binary mediator
ModelInfo	Model information		Default
NObs	Number of observations		Default
OutcomeEstimates	Parameter estimates for the outcome model	PROC CAUSALMED	POUTCOMEMOD
PercentDecomp	Percentage decompositions of the total effect	PROC CAUSALMED	DECOMP
ResponseProfile	Frequency counts for a binary outcome variable	MODEL	DIST=BIN
TreatmentProfile	Frequency counts for a binary treatment variable	CLASS	Binary treatment

To control the display of multiple ODS tables, you can override the "Default" settings in Table 38.6 by specifying global display options in the PROC CAUSALMED statement. Table 38.7 shows these options. The ODS tables that are displayed by these options are marked by *. Notice that the NOPRINT option suppresses all ODS table output.

Table 38.7 Global Display Options of the CAUSALMED Procedure

Options	PALL	Default	PSHORT	PSUMMARY	NOPRINT
BootstrapSamples	*	*	*	*	
ClassLevels	*	*			
EffectDecomp	*				
EffectSummary	*	*	*	*	
MediatorEstimates	*				
MediatorProfile	*	*			
ModelInfo	*	*	*		
NObs	*	*	*		
OutcomeEstimates	*				
PercentDecomp	*				

Options	PALL	Default	PSHORT	PSUMMARY	NOPRINT
ResponseProfile	*	*			
TreatmentProfile	*				

Examples: CAUSALMED Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 38.1: Mediation Analysis with Interaction Effects and Four-Way Decomposition

(View the complete code for this example (cmedex1.sas) in the example repository.)

This example continues the example in the section "Getting Started: CAUSALMED Procedure" on page 2424 by including an interaction effect between the treatment and the mediator in the outcome model. It also shows how you can obtain a four-way decomposition of the effects.

The goals of the observational study on page 2424 are to determine whether an encouraging environment provided by parents (which is represented by the variable Encourage) has an effect on the cognitive development of children (which is represented by the variable CogPerform) and to estimate the amount of the total causal effect that is due to the mediation of learning motivation (which is represented by the variable Motivation).

In the example on page 2424, the following statements are used to request a mediation analysis in which the main effects of Encourage and Motivation are specified in the outcome model:

```
proc causalmed data=Cognitive;
   model    CogPerform  = Encourage Motivation;
   mediator Motivation  = Encourage;
   covar FamSize SocStatus;
run;
```

This analysis also specifies confounding covariates, and it produces the summary of effects in Output 38.1.1.

Output 38.1.1 Estimation of Causal Effects Adjusting for Confounding Covariates

Summary of Effects						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8435	0.1525	6.5446	7.1424	44.88	<.0001
Controlled Direct Effect (CDE)	4.2962	0.1098	4.0811	4.5114	39.14	<.0001
Natural Direct Effect (NDE)	4.2962	0.1098	4.0811	4.5114	39.14	<.0001
Natural Indirect Effect (NIE)	2.5473	0.1563	2.2410	2.8536	16.30	<.0001
Percentage Mediated	37.2219	1.7523	33.7874	40.6564	21.24	<.0001
Percentage Due to Interaction	0	.	.	.	.	.
Percentage Eliminated	37.2219	1.7523	33.7874	40.6564	21.24	<.0001

The following statements extend the analysis by including an interaction term between Encourage and Motivation in the outcome model:

```
proc causalmed data=Cognitive decomp;
   model    CogPerform  = Encourage | Motivation;
   mediator Motivation  = Encourage;
   covar    FamSize SocStatus;
run;
```

The specification **`Encourage | Motivation`** includes the main effects of Encourage and Motivation and their interaction. Equivalently, you could specify **`Encourage Motivation Encourage*Motivation`**, where the third term represents the interaction effect. The results of this analysis are shown in Output 38.1.2.

Output 38.1.2 Summary of Causal Effects with Interaction Effects

Summary of Effects						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8421	0.1430	6.5618	7.1224	47.84	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.1509	0.04706	4.0587	4.2432	88.21	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.3325	1.3704	36.6465	42.0184	28.70	<.0001
Percentage Due to Interaction	0.4197	0.02367	0.3733	0.4661	17.73	<.0001
Percentage Eliminated	38.9128	1.3574	36.2524	41.5733	28.67	<.0001

When the interaction term is included, the 'Percentage Mediated' changes slightly from 37% (for the model without this term) to 39%. Although the percentage due to interaction that is shown in Output 38.1.2 is significant, it is less than 1%. Therefore, the interpretation of the results is not drastically different from those of the analysis with no interaction.

When you specify the DECOMP option, the CAUSALMED procedure generates a table, shown in Output 38.1.3, that displays various decompositions of the total effect: several two-way and three-way decompositions and a four-way decomposition. For more information about various decompositions and their interpretations, see the section "Causal Mediation Effects: Theory, Definitions, and Effect Decompositions" on page 2455.

Output 38.1.3 Decompositions of the Total Effect

Decompositions of Total Effect							
Decomposition	Effect	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
NDE+NIE	Natural Direct	4.1509	0.04706	4.0587	4.2432	88.21	<.0001
	Natural Indirect	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
CDE+PE	Controlled Direct	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
	Portion Eliminated	2.6625	0.1438	2.3807	2.9443	18.52	<.0001
TDE+PIE	Total Direct	4.2084	0.04695	4.1163	4.3004	89.63	<.0001
	Pure Indirect	2.6338	0.1423	2.3548	2.9127	18.51	<.0001
NDE+PIE+IMD	Natural Direct	4.1509	0.04706	4.0587	4.2432	88.21	<.0001
	Pure Indirect	2.6338	0.1423	2.3548	2.9127	18.51	<.0001
	Mediated Interaction	0.05743	0.003391	0.05078	0.06407	16.93	<.0001
CDE+PIE+PAI	Controlled Direct	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
	Pure Indirect	2.6338	0.1423	2.3548	2.9127	18.51	<.0001
	Portion Due to Interaction	0.02871	0.002009	0.02478	0.03265	14.30	<.0001
Four-Way	Controlled Direct	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
	Reference Interaction	-0.02871	0.002009	-0.03265	-0.02478	-14.30	<.0001
	Mediated Interaction	0.05743	0.003391	0.05078	0.06407	16.93	<.0001
	Pure Indirect	2.6338	0.1423	2.3548	2.9127	18.51	<.0001
Total	Total Effect	6.8421	0.1430	6.5618	7.1224	47.84	<.0001
Note: NDE=CDE+IRF, NIE=PIE+IMD, PAI=IRF+IMD, PE=PAI+PIE, TDE=CDE+PAI.							

Important effects such as CDE, NDE, and NIE that contribute to the components of the decompositions are shown in the "Summary of Effects" table in Output 38.1.2.

The primary decomposition in the table in Output 38.1.3 is the four-way decomposition. All other component effects can be deduced by summing up particular subsets of the effects in the four-way decomposition. The note at the bottom of the table shows how component effects are related.

The DECOMP option also generates a table of percentage decompositions, which is shown in Output 38.1.4.

Output 38.1.4 Percentage Decompositions of the Total Effect

Percentage Decompositions of Total Effect							
Decomposition	Effect	Percent	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
NDE+NIE	Natural Direct	60.67	1.37	57.98	63.35	44.27	<.0001
	Natural Indirect	39.33	1.37	36.65	42.02	28.70	<.0001
CDE+PE	Controlled Direct	61.09	1.36	58.43	63.75	45.00	<.0001
	Portion Eliminated	38.91	1.36	36.25	41.57	28.67	<.0001
TDE+PIE	Total Direct	61.51	1.34	58.87	64.14	45.75	<.0001
	Pure Indirect	38.49	1.34	35.86	41.13	28.63	<.0001
NDE+PIE+IMD	Natural Direct	60.67	1.37	57.98	63.35	44.27	<.0001
	Pure Indirect	38.49	1.34	35.86	41.13	28.63	<.0001
	Mediated Interaction	0.84	0.04	0.77	0.91	23.65	<.0001
CDE+PIE+PAI	Controlled Direct	61.09	1.36	58.43	63.75	45.00	<.0001
	Pure Indirect	38.49	1.34	35.86	41.13	28.63	<.0001
	Portion Due to Interaction	0.42	0.02	0.37	0.47	17.73	<.0001
Four-Way	Controlled Direct	61.09	1.36	58.43	63.75	45.00	<.0001
	Reference Interaction	-0.42	0.02	-0.47	-0.37	-17.66	<.0001
	Mediated Interaction	0.84	0.04	0.77	0.91	23.65	<.0001
	Pure Indirect	38.49	1.34	35.86	41.13	28.63	<.0001
Note: NDE=CDE+IRF, NIE=PIE+IMD, PAI=IRF+IMD, PE=PAI+PIE, TDE=CDE+PAI.							

This table shows that the two components that involve the interaction make up a very small percentage of the total effect. The mediated interaction effect represents less than 1%, and the reference interaction is very small and negative.

Example 38.2: Evaluating Controlled Direct Effects and Conditional Mediation Effects

(View the complete code for this example (cmedex2.sas) in the example repository.)

This example continues the analysis of Example 38.1; it illustrates the use of the EVALUATE statement for computing controlled direct effects and mediation effects conditional on covariate values.

The following code includes three EVALUATE statements that assign different values for the mediator Motivation:

```
proc causalmed data=Cognitive;
   model    CogPerform  = Encourage | Motivation;
   mediator Motivation  = Encourage;
   covar    FamSize SocStatus;
   evaluate 'Default Mean Value of Mediator' Motivation=mean;
   evaluate 'High-Motivation Group' Motivation = 1(SD);
   evaluate 'Low-Motivation Group' Motivation = -1(SD);
run;
```

The labels, which are enclosed in quotation marks, distinguish the three EVALUATE statements and the output that they produce. The first EVALUATE statement specifies the level of the mediator Motivation as its mean. This happens to be the default level, so you should expect this statement to produce the same

evaluation of causal effects that PROC CAUSALMED produces by default. Output 38.2.1 displays the causal mediation effects that are evaluated by default, and Output 38.2.2 displays the causal mediation effects that are evaluated for the mediator level in the first EVALUATE statement. Clearly, these two tables are identical.

Output 38.2.1 Summary of Effects (Default)

Summary of Effects						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8421	0.1430	6.5618	7.1224	47.84	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.1509	0.04706	4.0587	4.2432	88.21	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.3325	1.3704	36.6465	42.0184	28.70	<.0001
Percentage Due to Interaction	0.4197	0.02367	0.3733	0.4661	17.73	<.0001
Percentage Eliminated	38.9128	1.3574	36.2524	41.5733	28.67	<.0001

Output 38.2.2 Replicating the Default Summary of Effects

Summary of Effects: Default Mean Value of Mediator						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8421	0.1430	6.5618	7.1224	47.84	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.1509	0.04706	4.0587	4.2432	88.21	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.3325	1.3704	36.6465	42.0184	28.70	<.0001
Percentage Due to Interaction	0.4197	0.02367	0.3733	0.4661	17.73	<.0001
Percentage Eliminated	38.9128	1.3574	36.2524	41.5733	28.67	<.0001

There might be situations in which you want to evaluate the causal effects at other mediator levels. The second and third EVALUATE statements set the mediator level at one standard deviation above and below the mean, respectively. PROC CAUSALMED computes the sample mean and standard deviation (SD) for Motivation and then computes the levels of motivation as

$$m^* = \text{mean} + \text{SD}$$

$$m^* = \text{mean} - \text{SD}$$

These values of m^* are then used to evaluate the various causal mediation effects, which are displayed in Output 38.2.3 and Output 38.2.4.

Output 38.2.3 Evaluation of Effects for the High-Motivation Group

Summary of Effects: High-Motivation Group						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8421	0.1430	6.5618	7.1224	47.84	<.0001
Controlled Direct Effect (CDE)	4.3403	0.04687	4.2484	4.4321	92.60	<.0001
Natural Direct Effect (NDE)	4.1509	0.04706	4.0587	4.2432	88.21	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.3325	1.3704	36.6465	42.0184	28.70	<.0001
Percentage Due to Interaction	-1.9278	0.08279	-2.0901	-1.7656	-23.29	<.0001
Percentage Eliminated	36.5653	1.3995	33.8224	39.3082	26.13	<.0001

Output 38.2.4 Evaluation of Effects for the Low-Motivation Group

Summary of Effects: Low-Motivation Group						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8421	0.1430	6.5618	7.1224	47.84	<.0001
Controlled Direct Effect (CDE)	4.0190	0.04746	3.9260	4.1121	84.68	<.0001
Natural Direct Effect (NDE)	4.1509	0.04706	4.0587	4.2432	88.21	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.3325	1.3704	36.6465	42.0184	28.70	<.0001
Percentage Due to Interaction	2.7671	0.08527	2.6000	2.9343	32.45	<.0001
Percentage Eliminated	41.2603	1.3189	38.6753	43.8454	31.28	<.0001

Output 38.2.3 and Output 38.2.4 show that the total effect remains the same in the two evaluations, as expected. Because the controlled direct effect is defined at a particular level of the mediator level (m^*), it is not surprising that the two evaluations lead to different estimates of the controlled direct effect.

At one standard deviation above the mean of Motivation, the controlled direct effect is 4.34. This is higher than the controlled direct effect at one standard deviation below the mean, which is 4.02. The percentages of the total effect that are due to interaction also differ for the two levels of Motivation. One percentage is –2% and the other is 3%, although both are small and negligible.

You can also use the EVALUATE statement to evaluate causal mediation effects for particular target groups. The following EVALUATE statements estimate causal mediation effects for small families (FamSize=3) and large families (FamSize=7):

```
proc causalmed data=Cognitive;
   model    CogPerform  = Encourage | Motivation;
   mediator Motivation  = Encourage;
   covar    FamSize SocStatus;
   evaluate 'Small Families' FamSize=3;
   evaluate 'Large Families' FamSize=7;
run;
```

Output 38.2.5 and Output 38.2.6 display the corresponding effect summaries.

Output 38.2.5 Mediation Effects Conditional on Small Families

Summary of Effects: Small Families						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8495	0.1423	6.5705	7.1285	48.12	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.1584	0.04707	4.0661	4.2506	88.34	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.2900	1.3732	36.5985	41.9815	28.61	<.0001
Percentage Due to Interaction	0.5273	0.02278	0.4826	0.5719	23.15	<.0001
Percentage Eliminated	38.9788	1.3492	36.3344	41.6233	28.89	<.0001

Output 38.2.6 Mediation Effects Conditional on Large Families

Summary of Effects: Large Families						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8127	0.1457	6.5271	7.0982	46.77	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.1215	0.04717	4.0290	4.2140	87.37	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.5025	1.3590	36.8389	42.1662	29.07	<.0001
Percentage Due to Interaction	-0.01090	0.07140	-0.1508	0.1290	-0.15	0.8787
Percentage Eliminated	38.6487	1.3905	35.9234	41.3740	27.80	<.0001

The patterns of all causal effects are similar for small families and large families. Small families appear to have a slightly higher total effect. For both groups, the percentage of the total effect that is due to the interaction between Encourage and Motivation is very small. For both groups, about 40% of the total effect is due to the mediation of Motivation.

The next set of EVALUATE statements estimate causal mediation effects for subjects whose social status (SocStatus) is high or low.

```
proc causalmed data=Cognitive;
   model    CogPerform  = Encourage | Motivation;
   mediator Motivation  = Encourage;
   covar    FamSize SocStatus;
   evaluate 'High Social Status' SocStatus=1(SD);
   evaluate 'Low Social Status'  SocStatus=-1(SD);
run;
```

Output 38.2.7 and Output 38.2.8 display the corresponding effect summaries.

Output 38.2.7 Mediation Effects Conditional on High Social Status

Summary of Effects: High Social Status						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8894	0.1378	6.6193	7.1595	50.00	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.1982	0.04746	4.1052	4.2912	88.47	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.0625	1.3935	36.3312	41.7938	28.03	<.0001
Percentage Due to Interaction	1.1031	0.08592	0.9347	1.2715	12.84	<.0001
Percentage Eliminated	39.3321	1.2980	36.7881	41.8760	30.30	<.0001

Output 38.2.8 Mediation Effects Conditional on Low Social Status

Summary of Effects: Low Social Status						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.7948	0.1482	6.5043	7.0854	45.84	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.1037	0.04733	4.0109	4.1964	86.70	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.6062	1.3467	36.9667	42.2457	29.41	<.0001
Percentage Due to Interaction	-0.2733	0.1094	-0.4877	-0.05890	-2.50	0.0125
Percentage Eliminated	38.4877	1.4190	35.7066	41.2688	27.12	<.0001

Again, the patterns of all causal effects are similar for both groups. The high social status group appears to have a slightly higher total effect.

You can also combine the specifications of covariates to evaluate specific causal mediation effects. In the following EVALUATE statements, subjects are defined by a combination of levels of FamSize and SocStatus:

```
proc causalmed data=Cognitive;
   model    CogPerform  = Encourage | Motivation;
   mediator Motivation  = Encourage;
   covar    FamSize SocStatus;
   evaluate 'Most Favorable Environment'  FamSize=-.5(SD) SocStatus=1(SD);
   evaluate 'Least Favorable Environment' FamSize=.5(SD) SocStatus=-1(SD);
run;
```

The effects labeled ‘Most Favorable Environment’ are defined by FamSize at 0.5 standard deviation below the mean family size and SocStatus at 1 standard deviation above the mean social status rating. The effects labeled ‘Least Favorable Environment’ are defined by FamSize at 0.5 standard deviation above the mean family size and SocStatus at 1 standard deviation below the mean social status rating.

Output 38.2.9 and Output 38.2.10 display the corresponding effect summaries.

Output 38.2.9 Mediation Effects Conditional on Most Favorable Environment

Summary of Effects: Most Favorable Environment						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8969	0.1371	6.6281	7.1656	50.29	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.2057	0.04755	4.1125	4.2989	88.45	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.0202	1.3963	36.2835	41.7570	27.94	<.0001
Percentage Due to Interaction	1.2102	0.09789	1.0183	1.4020	12.36	<.0001
Percentage Eliminated	39.3978	1.2900	36.8694	41.9261	30.54	<.0001

Output 38.2.10 Mediation Effects Conditional on Least Favorable Environment

Summary of Effects: Least Favorable Environment						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.7874	0.1489	6.4955	7.0793	45.58	<.0001
Controlled Direct Effect (CDE)	4.1797	0.04696	4.0876	4.2717	89.00	<.0001
Natural Direct Effect (NDE)	4.0962	0.04742	4.0033	4.1891	86.39	<.0001
Natural Indirect Effect (NIE)	2.6912	0.1453	2.4065	2.9759	18.53	<.0001
Percentage Mediated	39.6497	1.3438	37.0160	42.2835	29.51	<.0001
Percentage Due to Interaction	-0.3836	0.1222	-0.6231	-0.1441	-3.14	0.0017
Percentage Eliminated	38.4201	1.4276	35.6221	41.2180	26.91	<.0001

The patterns of all causal mediation effects are similar for the two groups. The total effect for 'Most Favorable Environment' is slightly larger than the total effect for 'Least Favorable Environment'.

Together, these evaluations show that about 40% of the effect of parental encouragement on cognitive development is mediated by children's learning motivation. The interaction effect of parental encouragement and children's learning motivation is small. And more importantly, these conclusions appear to hold for different family sizes and levels of social status.

For more information about setting the covariate levels, see the EVALUATE statement and the section "Evaluating Causal Mediation Effects" on page 2466.

Example 38.3: Smoking Effect on Infant Mortality

(View the complete code for this example (cmedex3.sas) in the example repository.)

This example demonstrates causal mediation analysis with treatment, outcome, and mediator variables that are all binary. The data contain information about infant mortality in 2003 and were obtained from the US National Center for Health Statistics. A random sample of 100,000 observations is used in this example. The analysis and its interpretation are purely illustrative; definitive conclusions should not be drawn from this example.

The following statements print the first 10 observations of the data set, which are shown in Output 38.3.1:

```
proc print data=sashelp.birthwgt(obs=10);
run;
```

Output 38.3.1 First 10 Observations of birthwgt Data Set

Obs	LowBirthWgt	Married	AgeGroup	Race	Drinking	Death	Smoking	SomeCollege
1	No	No	3	Asian	No	No	No	Yes
2	No	No	2	White	No	No	No	No
3	Yes	Yes	2	Native	No	Yes	No	No
4	No	No	2	White	No	No	No	No
5	No	No	2	White	No	No	No	Yes
6	No	No	2	White	No	No	No	
7	No	No	2	Asian	No	No	No	Yes
8	No	No	3	White	No	No	No	Yes
9	No	Yes	1	Black	No	No	No	No
10	No	No	2	Native	No	No	No	Yes

The main variables in the analysis are as follows:

- The treatment variable is Smoking. It is an indicator of maternal smoking behavior, with values `'Yes'` and `'No'`.
- The outcome variable is Death. It is an indicator of infant death within one year of birth, with values `'Yes'` and `'No'`.
- The mediator variable is LowBirthWgt. It is an indicator of low birth weight (less than 2,500 grams), with values `'Yes'` and `'No'`.

The analysis also includes five confounding covariates:

- AgeGroup represents maternal ages of less than 20, between 20 and 35, and greater than 35, with values 1, 2, and 3, respectively.
- Drinking is an indicator of maternal drinking during pregnancy, with values `'Yes'` and `'No'`.
- Married is an indicator of marital status, with values `'Yes'` and `'No'`.
- Race is an indicator of race, with values `'Asian'`, `'Black'`, `'Hispanic'`, `'Native'` (native American), and `'White'`.
- SomeCollege is an indicator of whether the mother has 12 or more years of education, with values `'Yes'` and `'No'`.

The following statements specify a causal mediation model:

```
proc causalmed data=sashelp.birthwgt decomp;
   class LowBirthWgt Smoking Death AgeGroup Married Race
         Drinking SomeCollege /descending;
   mediator LowBirthWgt = Smoking;
```

```
    model Death = LowBirthWgt | Smoking;
    covar AgeGroup Married Race Drinking SomeCollege;
    evaluate 'Low Birth-Weight' LowBirthWgt='Yes' / nodecomp;
    evaluate 'Normal Birth-Weight' LowBirthWgt='No' / nodecomp;
  run;
```

The DECOMP option requests various total effect decompositions. The MEDIATOR statement specifies the mediator model for the response LowBirthWgt. The MODEL statement specifies the outcome model for the response Death and assumes an interaction between LowBirthWgt and Smoking. The CLASS statement names the categorical variables in the analysis, and the DESCENDING option models the probability of the last level of both responses (Death=`'Yes'` and LowBirthWgt=`'Yes'`). The COVAR statement specifies the five covariates. Finally, the two EVALUATE statements specify the mediator levels for comparing their patterns of causal mediation effects.

Output 38.3.2 displays the model information, which includes the outcome, treatment, and mediator variables, the distributions, and the link functions of the response variables. Because observations that have missing values are not included, only 93,292 observations are used for analysis.

Output 38.3.2 Model Information

Model Information	
Data Set	SASHELP.BIRTHWGT
Outcome Variable	Death
Treatment Variable	Smoking
Mediator Variable	LowBirthWgt
Outcome Modeling	Generalized Linear Model
Outcome Distribution	Binomial
Outcome Link Function	Logit
Mediator Modeling	Generalized Linear Model
Mediator Distribution	Binomial
Mediator Link Function	Logit

Number of Observations Read	100000
Number of Observations Used	93292

Output 38.3.3 displays the levels of the categorical variables, including binary response variables.

Output 38.3.3 Class Levels

Class Level Information		
Class	**Levels**	**Values**
LowBirthWgt	2	Yes No
Smoking	2	Yes No
Death	2	Yes No
AgeGroup	3	3 2 1
Married	2	Yes No
Race	5	White Native Hispanic Black Asian
Drinking	2	Yes No
SomeCollege	2	Yes No

Output 38.3.4 displays frequency counts of the binary outcome, mediator, and treatment variables. It also

shows which levels of the response variables are being modeled.

Output 38.3.4 Profiles of Binary Outcome, Mediator, and Treatment Variables

Response Profile		
Ordered Value	Death	Total Frequency
1	Yes	527
2	No	92765

Outcome probability modeled is Death='Yes'.

Mediator Profile		
Ordered Value	LowBirthWgt	Total Frequency
1	Yes	7562
2	No	85730

Mediator probability modeled is LowBirthWgt='Yes'.

Treatment Profile		
Ordered Value	Smoking	Total Frequency
1	Yes	20984
2	No	72308

Output 38.3.5 displays the major decompositions of effects on infant mortality on both the odds ratio (OR) scale and the excess relative risk scale. Percentages of the total effect are displayed only on the excess relative risk scale.

Output 38.3.5 Summary of Effects on Infant Mortality

Summary of Effects										
	Nontransformed Scale						Back-Transformed from Log Scale			
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|	95% Confidence Limits		Z	Pr > |Z|
Odds Ratio Total Effect	1.7071	0.2215	1.2729	2.1412	3.19	0.0014	1.3237	2.2014	4.12	<.0001
Odds Ratio Controlled Direct Effect (CDE)	1.8940	0.3540	1.2002	2.5879	2.53	0.0116	1.3131	2.7320	3.42	0.0006
Odds Ratio Natural Direct Effect (NDE)	1.3626	0.1768	1.0160	1.7092	2.05	0.0403	1.0566	1.7572	2.38	0.0171
Odds Ratio Natural Indirect Effect (NIE)	1.2528	0.03432	1.1855	1.3201	7.37	<.0001	1.1873	1.3219	8.23	<.0001
Total Excess Relative Risk	0.7071	0.2215	0.2729	1.1412	3.19	0.0014				
Excess Relative Risk Due to CDE	0.3246	0.1207	0.08810	0.5611	2.69	0.0071				
Excess Relative Risk Due to NDE	0.3626	0.1768	0.01604	0.7092	2.05	0.0403				
Excess Relative Risk Due to NIE	0.3445	0.06119	0.2245	0.4644	5.63	<.0001				
Percentage Mediated	48.7165	9.8917	29.3291	68.1040	4.92	<.0001				
Percentage Due to Interaction	8.1202	19.8380	-30.7615	47.0020	0.41	0.6823				
Percentage Eliminated	54.0930	11.6647	31.2306	76.9554	4.64	<.0001				

The first four rows of the table in Output 38.3.5 summarize the effects on the odds ratio scale. The controlled direct effect (CDE) on this scale is 1.894. This is the CDE when the mediator variable LowBirthWgt is controlled at the level `'No'`. In other words, this is the CDE odds ratio for the group that has normal birth weights. The natural direct effect (NDE) and natural indirect effect (NIE) on the odds ratio scale are 1.363 and 1.253, respectively. Their product, rather than their sum, is the same as the total effect on the odds ratio scale, which is 1.707.

There are two sets of confidence limits, z-values, and p-values for the odds ratio effects in Output 38.3.5. One set is based on the normality of the original (nontransformed) scale, and the other is based on the normality of log scale with back-transformations. The log scale inferences are sometimes preferred because they are range-respecting. That is, unlike the nontransformed Wald confidence intervals that could include negative values for odds ratios, the confidence limits constructed from back-transformation of the log scale are always nonnegative. However, for the current example, the two sets of confidence limits, z-values, and p-values result in very much the same statistical inferences about the four causal mediation effects.

The next seven rows of the table in Output 38.3.5 summarize effects on the excess relative risk (ERR) scale. The natural direct effect (0.363) and natural indirect effect (0.345) have an additive property on this scale; they sum to the total excess relative risk, which is 0.707. Additivity makes it easier to use these values to deduce the 'Percentage Mediated', which is 48.72% (= 0.3445/0.7071×100%). Therefore, about 50% of the smoking effect on infant mortality is mediated through the lowering of babies' birth weights. However, the 95% confidence interval for the 'Percentage Mediated' is (29.3%, 68.1%), which is fairly wide. More data would yield a more precise interval estimate.

The percentage of total effect due to the interaction between smoking and low birth weights is about 8%, which is relatively small. Again, the corresponding 95% confidence interval, (–30.8%, 47.0%), is quite wide.

The DECOMP option requests various total effect decompositions, which are shown in Output 38.3.6. Following VanderWeele (2014), all these decompositions are computed on the excess relative risk scale.

Output 38.3.6 Decompositions of Smoking Effects on Infant Mortality

Decompositions of Total Excess Relative Risk

Decomposition	Excess Relative Risk	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
NDE+NIE	Natural Direct	0.3626	0.1768	0.01604	0.7092	2.05	0.0403
	Natural Indirect	0.3445	0.06119	0.2245	0.4644	5.63	<.0001
CDE+PE	Controlled Direct	0.3246	0.1207	0.08810	0.5611	2.69	0.0071
	Portion Eliminated	0.3825	0.1556	0.07752	0.6874	2.46	0.0140
TDE+PIE	Total Direct	0.3820	0.2188	-0.04688	0.8109	1.75	0.0809
	Pure Indirect	0.3251	0.03534	0.2558	0.3943	9.20	<.0001
NDE+PIE+IMD	Natural Direct	0.3626	0.1768	0.01604	0.7092	2.05	0.0403
	Pure Indirect	0.3251	0.03534	0.2558	0.3943	9.20	<.0001
	Mediated Interaction	0.01940	0.05229	-0.08309	0.1219	0.37	0.7106
CDE+PIE+PAI	Controlled Direct	0.3246	0.1207	0.08810	0.5611	2.69	0.0071
	Pure Indirect	0.3251	0.03534	0.2558	0.3943	9.20	<.0001
	Portion Due to Interaction	0.05742	0.1547	-0.2457	0.3606	0.37	0.7105
Four-Way	Controlled Direct	0.3246	0.1207	0.08810	0.5611	2.69	0.0071
	Reference Interaction	0.03801	0.1024	-0.1627	0.2387	0.37	0.7105
	Mediated Interaction	0.01940	0.05229	-0.08309	0.1219	0.37	0.7106
	Pure Indirect	0.3251	0.03534	0.2558	0.3943	9.20	<.0001
Total	Excess Relative Risk	0.7071	0.2215	0.2729	1.1412	3.19	0.0014

Note: NDE=CDE+IRF, NIE=PIE+IMD, PAI=IRF+IMD, PE=PAI+PIE, TDE=CDE+PAI.

As shown in Output 38.3.7, PROC CAUSALMED also displays the corresponding decompositions by their percentage contribution to the total effect on the excess relative risk scale.

Output 38.3.7 Percentage Decomposition of Smoking Effects on Infant Mortality

Percentage Decompositions of Total Excess Relative Risk							
Decomposition	Excess Relative Risk	Percent	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
NDE+NIE	Natural Direct	51.28	9.89	31.90	70.67	5.18	<.0001
	Natural Indirect	48.72	9.89	29.33	68.10	4.92	<.0001
CDE+PE	Controlled Direct	45.91	11.66	23.04	68.77	3.94	<.0001
	Portion Eliminated	54.09	11.66	31.23	76.96	4.64	<.0001
TDE+PIE	Total Direct	54.03	14.49	25.62	82.43	3.73	0.0002
	Pure Indirect	45.97	14.49	17.57	74.38	3.17	0.0015
NDE+PIE+IMD	Natural Direct	51.28	9.89	31.90	70.67	5.18	<.0001
	Pure Indirect	45.97	14.49	17.57	74.38	3.17	0.0015
	Mediated Interaction	2.74	6.70	-10.40	15.88	0.41	0.6823
CDE+PIE+PAI	Controlled Direct	45.91	11.66	23.04	68.77	3.94	<.0001
	Pure Indirect	45.97	14.49	17.57	74.38	3.17	0.0015
	Portion Due to Interaction	8.12	19.84	-30.76	47.00	0.41	0.6823
Four-Way	Controlled Direct	45.91	11.66	23.04	68.77	3.94	<.0001
	Reference Interaction	5.38	13.14	-20.37	31.13	0.41	0.6824
	Mediated Interaction	2.74	6.70	-10.40	15.88	0.41	0.6823
	Pure Indirect	45.97	14.49	17.57	74.38	3.17	0.0015
Note: NDE=CDE+IRF, NIE=PIE+IMD, PAI=IRF+IMD, PE=PAI+PIE, TDE=CDE+PAI.							

The entries for the four-way decomposition in Output 38.3.7 show that 46% of the total effect is attributed to neither interaction nor mediation (‘Controlled Direct’), 5% is attributed to interaction but not mediation (‘Reference Interaction’), 3% is attributed to both mediation and interaction (‘Mediated Interaction’), and 46% is attributed to mediation but not interaction (‘Pure Indirect’).

In the three-way decomposition labeled ‘CDE+PIE+PAI,’ the percentage of total effect that is attributed to the interaction (PAI or ‘Portion Due to Interaction’ in the table) is about 8%, which is not large but is also not ignorable.

Note that some of the confidence intervals in this table span from negative to positive values. This indicates that the corresponding point estimates might not be very accurate.

As requested by the first EVALUATE statement, the table in Output 38.3.8 displays the major effects and percentages when the mediator LowBirthWgt is set at the level `'Yes'`.

Output 38.3.8 Summary of Smoking Effects for the Low Birth-Weight Group

Summary of Effects: Low Birth-Weight										
			Nontransformed Scale				Back-Transformed from Log Scale			
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|	95% Confidence Limits		Z	Pr > |Z|
Odds Ratio Total Effect	1.7071	0.2215	1.2729	2.1412	3.19	0.0014	1.3237	2.2014	4.12	<.0001
Odds Ratio Controlled Direct Effect (CDE)	1.0917	0.1591	0.7799	1.4036	0.58	0.5643	0.8205	1.4527	0.60	0.5470
Odds Ratio Natural Direct Effect (NDE)	1.3626	0.1768	1.0160	1.7092	2.05	0.0403	1.0566	1.7572	2.38	0.0171
Odds Ratio Natural Indirect Effect (NIE)	1.2528	0.03432	1.1855	1.3201	7.37	<.0001	1.1873	1.3219	8.23	<.0001
Total Excess Relative Risk	0.7071	0.2215	0.2729	1.1412	3.19	0.0014				
Excess Relative Risk Due to CDE	0.8669	1.4959	-2.0649	3.7988	0.58	0.5622				
Excess Relative Risk Due to NDE	0.3626	0.1768	0.01604	0.7092	2.05	0.0403				
Excess Relative Risk Due to NIE	0.3445	0.06119	0.2245	0.4644	5.63	<.0001				
Percentage Mediated	48.7165	9.8917	29.3291	68.1040	4.92	<.0001				
Percentage Due to Interaction	-68.5853	167.58	-397.04	259.87	-0.41	0.6823				
Percentage Eliminated	-22.6126	179.59	-374.60	329.38	-0.13	0.8998				

The odds ratio CDE (which is evaluated for the low birth-weight group) is 1.09, with a corresponding 95% confidence interval of (0.82, 1.45) when the log scale is used for inferences.

As requested by the second EVALUATE statement, the table in Output 38.3.9 displays the major effects and percentages when the mediator LowBirthWgt is set at the level `'No'`.

Output 38.3.9 Summary of Smoking Effects for the Normal Birth-Weight Group

Summary of Effects: Normal Birth-Weight										
			Nontransformed Scale				Back-Transformed from Log Scale			
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|	95% Confidence Limits		Z	Pr > |Z|
Odds Ratio Total Effect	1.7071	0.2215	1.2729	2.1412	3.19	0.0014	1.3237	2.2014	4.12	<.0001
Odds Ratio Controlled Direct Effect (CDE)	1.8940	0.3540	1.2002	2.5879	2.53	0.0116	1.3131	2.7320	3.42	0.0006
Odds Ratio Natural Direct Effect (NDE)	1.3626	0.1768	1.0160	1.7092	2.05	0.0403	1.0566	1.7572	2.38	0.0171
Odds Ratio Natural Indirect Effect (NIE)	1.2528	0.03432	1.1855	1.3201	7.37	<.0001	1.1873	1.3219	8.23	<.0001
Total Excess Relative Risk	0.7071	0.2215	0.2729	1.1412	3.19	0.0014				
Excess Relative Risk Due to CDE	0.3246	0.1207	0.08810	0.5611	2.69	0.0071				
Excess Relative Risk Due to NDE	0.3626	0.1768	0.01604	0.7092	2.05	0.0403				
Excess Relative Risk Due to NIE	0.3445	0.06119	0.2245	0.4644	5.63	<.0001				
Percentage Mediated	48.7165	9.8917	29.3291	68.1040	4.92	<.0001				
Percentage Due to Interaction	8.1202	19.8380	-30.7615	47.0020	0.41	0.6823				
Percentage Eliminated	54.0930	11.6647	31.2306	76.9554	4.64	<.0001				

The odds ratio CDE (which is evaluated for the normal birth-weight group) is now 1.89, with a corresponding 95% confidence interval of (1.31, 2.73) when the log scale is used for inferences.

Note that the controlled level of the mediator requested by the second EVALUATE statement coincides the default setting that uses the last level of mediator as the controlled level. Hence, the results in Output 38.3.9 and Output 38.3.5 are identical.

Example 38.4: Mediation Analysis by Linear Structural Equation Modeling

(View the complete code for this example (cmedex4.sas) in the example repository.)

This example illustrates the use of linear structural equation modeling and the CALIS procedure for doing a limited form of mediation analysis. For this analysis, the CALIS procedure and the CAUSALMED procedure produce results that are very similar. However, the more general approach implemented in the CAUSALMED procedure is needed to define and compute the mediation effects in a broader context. Within this context, Example 38.1 illustrates how the general approach deals with interaction effects, and Example 38.3 illustrates how it treats binary outcomes and binary mediators in a unified fashion.

The scenario in this example is the observational study that is presented in the section "Getting Started: CAUSALMED Procedure" on page 2424. The goals of the study are to determine whether an encouraging environment provided by parents (which is represented by the variable Encourage) has an effect on the cognitive development of children (which is represented by the variable CogPerform) and to estimate the amount of the total causal effect that is due to the mediation of learning motivation (which is represented by the variable Motivation).

In the example in the section "Getting Started: CAUSALMED Procedure" on page 2424, PROC CAUSALMED is used to carry out two mediation analyses:

```
proc causalmed data=Cognitive;
   model    CogPerform  = Encourage Motivation;
   mediator Motivation  = Encourage;
run;

proc causalmed data=Cognitive;
   model    CogPerform  = Encourage Motivation;
   mediator Motivation  = Encourage;
   covar FamSize SocStatus;
run;
```

The first analysis does not specify any confounding covariates. It produces the summary of effects in Output 38.4.1, which shows that the 'Percentage Mediated' is about 47%.

Output 38.4.1 Estimation of Causal Effects without Adjusting for Confounding Covariates

Summary of Effects						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	8.0423	0.03200	7.9796	8.1050	251.30	<.0001
Controlled Direct Effect (CDE)	4.2835	0.1062	4.0754	4.4917	40.33	<.0001
Natural Direct Effect (NDE)	4.2835	0.1062	4.0754	4.4917	40.33	<.0001
Natural Indirect Effect (NIE)	3.7588	0.1091	3.5449	3.9727	34.44	<.0001
Percentage Mediated	46.7377	1.3254	44.1400	49.3353	35.26	<.0001
Percentage Due to Interaction	0	.	.	.	.	.
Percentage Eliminated	46.7377	1.3254	44.1400	49.3353	35.26	<.0001

The second analysis specifies confounding covariates. It produces the summary of effects in Output 38.4.2. These effects have more appropriate causal interpretations if FamSize and SocStatus are the only important confounding variables that must be controlled for. Controlling for covariates, Output 38.4.2 shows a more conservative 'Percentage Mediated' of 37%.

Output 38.4.2 Estimation of Causal Effects Adjusting for Confounding Covariates

Summary of Effects						
	Estimate	Standard Error	Wald 95% Confidence Limits		Z	Pr > |Z|
Total Effect	6.8435	0.1525	6.5446	7.1424	44.88	<.0001
Controlled Direct Effect (CDE)	4.2962	0.1098	4.0811	4.5114	39.14	<.0001
Natural Direct Effect (NDE)	4.2962	0.1098	4.0811	4.5114	39.14	<.0001
Natural Indirect Effect (NIE)	2.5473	0.1563	2.2410	2.8536	16.30	<.0001
Percentage Mediated	37.2219	1.7523	33.7874	40.6564	21.24	<.0001
Percentage Due to Interaction	0	.	.	.	.	.
Percentage Eliminated	37.2219	1.7523	33.7874	40.6564	21.24	<.0001

he second analysis decomposes the total effect of an encouraging environment on cognitive development into two percentages:

- 63% of the total effect is through the direct pathway Encourage→CogPerform
- 37% of the total effect is through the mediation pathway Encourage→Motivation→CogPerform

Statements such as these invite the use of structural equation modeling, which offers the same type of language for describing causal sequences. Indeed, mediation analysis has a relatively long history in the field of psychology, where structural equation modeling is quite popular.

By specifying the relevant causal pathways in structural equation models, you can use the CALIS procedure to obtain essentially the same mediation analyses as those obtained with the CAUSALMED procedure:

```
proc calis data=cognitive;
   path
      Encourage            ===> Motivation,
      Encourage Motivation ===> CogPerform;
   effpart  Encourage ===> CogPerform;
run;

proc calis data=cognitive;
   path
      Encourage            ===> Motivation,
      Encourage Motivation ===> CogPerform,
      FamSize    ===> Encourage Motivation CogPerform,
      SocStatus  ===> Encourage Motivation CogPerform;
   effpart  Encourage ===> CogPerform;
run;
```

The EFFPART statements request the total effect decompositions of Encourage on CogPerform in the two analyses. Output 38.4.3 shows the total effect decomposition when the covariates are ignored in the linear structural equation model. The total, direct, and indirect effects and their standard error estimates closely match those in Output 38.4.1.

Output 38.4.3 Summary of Causal Effects

Effects of Encourage			
Effect / Std Error / t Value / p Value			
	Total	Direct	Indirect
CogPerform	8.0423	4.2835	3.7588
	0.0321	0.1064	0.1093
	250.8761	40.2662	34.3806
	<.0001	<.0001	<.0001

Output 38.4.4 shows the total effect decomposition when the covariates are incorporated in the linear structural equation model. Again, the total, direct, and indirect effects and their standard error estimates closely match those in Output 38.4.2.

Output 38.4.4 Summary of Causal Effects

Effects of Encourage			
Effect / Std Error / t Value / p Value			
	Total	Direct	Indirect
CogPerform	6.8435	4.2962	2.5473
	0.1527	0.1099	0.1565
	44.8044	39.0749	16.2739
	<.0001	<.0001	<.0001

However, the similarity of the analyses obtained with the CALIS and CAUSALMED procedures does not extend to more general situations. The limitations of structural equation modeling include the following:

- It does not have a clear foundation for defining causal mediation effects.
- It does not deal with interaction effects effectively.
- It does not treat binary outcomes and binary mediators in a unified fashion.

The general mediation approach that is implemented in PROC CAUSALMED overcomes these limitations of traditional linear structural equation modeling. For more information about the theoretical foundation of the general mediation approach, see the sections "Causal Mediation Effects: Theory, Definitions, and Effect Decompositions" on page 2455 and "Causal Mediation Effects: Assumptions, Identification, and Estimation" on page 2462.

Example 38.5: Causal Mediation Analysis of a Time-to-Event Outcome

(View the complete code for this example (cmedex5.sas) in the example repository.)

This example illustrates causal mediation analyses of time-to-event data (in this case, the event is death). Lagakos and Mosteller (1981) presented a case study of the effects of Red 40, a monoazoaryl disodium disulfonate dye that is used as a color additive in foods and drugs, on the development of reticuloendothelial (RE) tumors and acceleration of death in mice.

The experiment took place in 1976. Four hundred mice, 200 females and 200 males, were allocated to four groups: a control group, and three dose-level groups of Red 40. There were 50 females and 50 males in each group. The original plan was to run the experiment for two years (104 weeks). If a mouse died before the experiment ended, the presence of RE tumors would be investigated. If a mouse survived to week 104, it was euthanized to investigate the presence of RE tumors. However, for some reason, 142 mice were unexpectedly euthanized at week 42. No RE tumors were found in these mice. Therefore, the time-to-death measure in the data can be censored at either week 42 or week 104.

Lagakos and Mosteller (1981) noticed that the evaluation of Red 40 effects could have been complicated by cage and litter effects. Unfortunately, these variables were not collected in the experiment. Despite that problem, the data are used here to demonstrate some simplified causal mediation analyses for time-to-event data. In any case, this example is not intended to provide definitive analyses of the data.

For simplicity, only the data for the control and the high-dose-level (treatment) groups are analyzed. The following DATA step inputs the observations for the control and treatment groups:

```
data RedDye;
   input RedDye Weeks Censor Female Tumor Freq;
   datalines;
0    29      0       0    0    1
0    30      0       0    0    1
0    38      0       0    0    1
0    48      0       0    0    1
0    53      0       0    0    1
0    56      0       0    0    1
0    62      0       0    0    1
0    70      0       0    0    1
0    71      0       0    0    1
0    74      0       0    0    1
0    74      0       0    0    1
0    76      0       0    0    1
0    85      0       0    1    1
0    86      0       0    0    1
0    86      0       0    0    1
0    92      0       0    0    1
0    97      0       0    1    1
0    99      0       0    1    1
0    101     0       0    0    1
0    102     0       0    0    1
0    103     0       0    0    1
0    104     1       0    1    0
0    104     1       0    0   12
0    42      1       0    0   17
0    70      0       1    1    1
0    77      0       1    0    1
0    83      0       1    1    1
0    87      0       1    0    1
0    92      0       1    0    1
0    92      0       1    0    1
0    93      0       1    1    1
0    96      0       1    1    1
0    100     0       1    1    1
0    102     0       1    0    1
0    102     0       1    0    1
0    103     0       1    1    1
0    104     1       1    1    2
0    104     1       1    0   16
0    42      1       1    0   20
3    2       0       0    0    1
3    16      0       0    0    1
3    23      0       0    0    1
3    34      0       0    0    1
3    35      0       0    1    1
3    35      0       0    1    1
3    67      0       0    1    1
3    77      0       0    0    1
3    77      0       0    0    1
3    79      0       0    1    1
3    84      0       0    0    1
```

```
3    89      0       0    0    1
3    92      0       0    0    1
3    96      0       0    0    1
3    97      0       0    0    1
3    98      0       0    0    1
3    99      0       0    0    1
3    104     1       0    1    0
3    104     1       0    0   19
3    42      1       0    0   14
3    34      0       1    1    1
3    36      0       1    1    1
3    48      0       1    1    1
3    48      0       1    0    1
3    65      0       1    1    1
3    91      0       1    1    1
3    91      0       1    0    1
3    98      0       1    0    1
3    102     0       1    0    1
3    102     0       1    0    1
3    103     0       1    1    1
3    104     1       1    1    3
3    104     1       1    0   18
3     42     1       1    0   18
;
```

The six variables in the data set are described as follows:

- RedDye indicates the absence (0, the control level) or high dose level (3, the treatment level) of Red 40. This is the treatment variable of interest.
- Weeks measures the time to death in weeks. This is the outcome variable of interest.
- Censor is an indicator of a censored outcome. A value of 1 indicates that Weeks was a censored measure.
- Female indicates the sex of the mice, 0 for male and 1 for female. This variable is a confounding covariate in the following causal mediation analyses.
- Tumor indicates the absence (0) or presence (1) of RE tumors. This variable is the mediator variable of interest.
- Freq indicates the frequency of an observation. As mentioned, the values of Weeks in some observations were censored at week 42 or week 104. The frequencies of these censored observations could be different from 1. In contrast, all noncensored observations have a frequency value of 1 in the data set.

Before doing a causal mediation analysis, it is useful to compare some simple statistics between the control and treatment groups. The following statements compute the means and other statistics of some variables:

```
proc means mean stderr min max data=RedDye;
   var Censor Female Tumor Weeks;
   class RedDye;
   freq Freq;
```

```
run;
```

Output 38.5.1 displays the observed means of Censor, Female, Tumor, and Weeks in the control and treatment groups. The treatment (high-dose) group has a slightly higher rate of RE tumors and a slightly longer average life than the control group. But these observed differences are not greater than their corresponding standard errors and thus are negligible.

Output 38.5.1 Comparing the Control and High-Dose Groups

RedDye	N Obs	Variable	Mean	Std Error	Minimum	Maximum
0	100	Censor	0.6700000	0.0472582	0	1.0000000
		Female	0.5000000	0.0502519	0	1.0000000
		Tumor	0.1100000	0.0314466	0	1.0000000
		Weeks	73.0300000	2.8561277	29.0000000	104.0000000
3	100	Censor	0.7200000	0.0451261	0	1.0000000
		Female	0.5000000	0.0502519	0	1.0000000
		Tumor	0.1300000	0.0337998	0	1.0000000
		Weeks	74.2200000	3.1066751	2.0000000	104.0000000

It is important to note that the means of Tumor and Weeks might have been plagued by the censored observations. The proportions of censored observations are high—67% and 72%, respectively, for the control and treatment groups. Therefore, the comparisons of these means between groups must be interpreted with caution.

Fortunately, you can analyze right-censored time-to-event outcomes together with noncensored outcomes in the CAUSALMED procedure. You perform a causal mediation analysis of the data by specifying the following statements:

```
proc causalmed data=RedDye ciratio=log poutcomemod pmedmod;
   class RedDye Female Tumor;
   model Weeks*Censor(1) = RedDye | Tumor / aft;
   mediator Tumor(event='1') = RedDye(treat='3');
   covar Female;
   freq Freq;
run;
```

In the PROC CAUSALMED statement, you use the CIRATIO=LOG option to base the computations of confidence intervals, z-values, and p-values of the mean ratio effects on the log scale. You use the POUTCOMEMOD and PMEDMOD options, respectively, to output the estimates of the outcome and mediator models.

In the CLASS statement, you specify the binary variables RedDye, Female, and Tumor.

In the MODEL statement, you use the AFT option to fit an accelerated failure time model for the outcome variable Weeks. You include the RedDye effect, the Tumor effect, their interaction effect, and the covariate effect of Female (which is specified in the COVAR statement) in this outcome model. Observations with a censored outcome are indicated by a value of '1' for the variable Censor.

In the MEDIATOR statement, you model the mediator variable Tumor by using a logistic regression model that includes the RedDye treatment effect and the covariate effect of Female, which is specified in the COVAR statement. The EVENT='1' option specifies that a value of '1' for Tumor indicates the presence of a tumor. The TREAT='3' option specifies that a value of '3' for RedDye indicates the treatment condition—that is, a high dose of Red 40.

In the FREQ statement, you specify that Freq indicates the frequency of an observation.

Output 38.5.2 displays information about various variables and their modeling and distributional types in the analysis. You can use this table to check whether your specifications have been set as intended.

Output 38.5.2 Model Information

Model Information	
Data Set	WORK.REDDYE
Outcome Variable	Log(Weeks)
Censoring Variable	Censor
Censoring Value(s)	1
Treatment Variable	RedDye
Mediator Variable	Tumor
Frequency Variable	Freq
Outcome Modeling	Accelerated Failure Time
Outcome Distribution	Weibull
Mediator Modeling	Generalized Linear Model
Mediator Distribution	Binomial
Mediator Link Function	Logit

Output 38.5.3 shows the number of observations and frequencies in different categories. Because the frequency variable is specified, the total number of mice in the study was 200. Of this total, 139 (69.5%) were right-censored.

Output 38.5.3 Number of Observations

Number of Observations Read	73
Number of Observations Used	71
Right-Censored Observations	10
Noncensored Observations	61
Sum of Frequencies Read	200
Sum of Frequencies Used	200
Right-Censored Frequencies	139
Noncensored Frequencies	61
Percentage Censored	69.5

Output 38.5.4 is useful for checking whether the treatment level and the mediator level for modeling were specified correctly. The information in these tables confirms the intended specifications.

Output 38.5.4 Class Levels, Mediator Profile, and Treatment Profile

Class Level Information		
Class	**Levels**	**Values**
RedDye	2	3 0
Female	2	0 1
Tumor	2	1 0

Output 38.5.4 *continued*

Mediator Profile		
Ordered Value	Tumor	Total Frequency
1	1	24
2	0	176

Mediator probability modeled is Tumor='1'.

Treatment Profile		
Ordered Value	RedDye	Total Frequency
1	3	100
2	0	100

Output 38.5.5 shows a summary of the causal mediation effects. The causal effects are expressed as mean ratios. A ratio of 1 means that the average mean time to death in the treatment group is the same as that in the control group, indicating no treatment effect. The mean ratio total effect is 1.14, which is not significantly different from the null value of 1. This indicates that there is no significant total causal treatment effect of Red 40 on the survival time.

The natural indirect effect (NIE) is 0.99. Although this indicates that a high dose of Red 40 might have a very small negative indirect effect on survival through its effect on the tumor, the evidence here is very weak because the NIE is not statistically significantly different from 1. The natural direct effect (NDE) is 1.15, which also is not significantly different from 1. For these mean ratio causal effects, you can verify that the product of NIE and NDE is the same as the total effect.

The total excess mean ratio measures how different the mean ratio total effect is from 1. In this case, it is 0.14 (=1.14–1). The percentage decomposition of effects in the summary table is based on the partitioning of the total excess mean ratio rather than that of the mean ratio total effect. For example, the percentage mediated is about –8%, which is computed by $-0.011/0.140\times100\%$. The 95% confidence interval for percentage mediated is very wide and includes the zero value. Therefore, this percentage is not statistically significant.

In fact, no effect or percentage estimates in Output 38.5.5 are statistically significant.

Output 38.5.5 Summary of Causal Mediation Effects

Summary of Effects						
	Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Mean Ratio Total Effect	1.1402	0.1180	0.9308	1.3967	1.27	0.2050
Mean Ratio Controlled Direct Effect (CDE)	1.1803	0.1286	0.9535	1.4612	1.52	0.1279
Mean Ratio Natural Direct Effect (NDE)	1.1515	0.1184	0.9414	1.4087	1.37	0.1700
Mean Ratio Natural Indirect Effect (NIE)	0.9902	0.02232	0.9474	1.0349	-0.44	0.6607
Total Excess Mean Ratio	0.1402	0.1180	-0.09115	0.3716	1.19	0.2349
Excess Mean Ratio Due to CDE	0.1851	0.1314	-0.07239	0.4426	1.41	0.1588
Excess Mean Ratio Due to NDE	0.1515	0.1184	-0.08054	0.3836	1.28	0.2006
Excess Mean Ratio Due to NIE	-0.01134	0.02582	-0.06195	0.03927	-0.44	0.6605
Percentage Mediated	-8.0882	20.2332	-47.7446	31.5682	-0.40	0.6893
Percentage Due to Interaction	-28.4718	20.4780	-68.6080	11.6644	-1.39	0.1644
Percentage Eliminated	-32.0379	27.1096	-85.1717	21.0959	-1.18	0.2373

To complete the interpretations of the results, you would like to examine the estimation of the outcome and mediator models. Output 38.5.6 shows the estimates for the outcome model.

Output 38.5.6 Outcome Model Estimation

Outcome Model Estimates							
Parameter		Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		5.0436	0.1110	4.8261	5.2611	2065.0523	<.0001
Female	0	-0.3366	0.1015	-0.5356	-0.1376	10.9921	0.0009
Female	1	0	.	.	.	.	.
RedDye	3	0.1658	0.1089	-0.04765	0.3793	2.3178	0.1279
RedDye	0	0	.	.	.	.	.
Tumor	1	-0.2952	0.1414	-0.5723	-0.01801	4.3569	0.0369
Tumor	0	0	.	.	.	.	.
RedDye*Tumor	3 1	-0.3804	0.1953	-0.7631	0.002341	3.7946	0.0514
RedDye*Tumor	3 0	0	.	.	.	.	.
RedDye*Tumor	0 1	0	.	.	.	.	.
RedDye*Tumor	0 0	0	.	.	.	.	.
Scale		0.3468	0.03761	0.2804	0.4290		
Weibull Shape		2.8833	0.3127	2.3312	3.5662		

As suggested by the effect estimate of 0.166, a high dose of Red 40 might have a positive effect on the survival time. But the corresponding p-value of 0.128 does not support the statistical significance of this effect.

The Tumor effect (p-value = 0.037) and the RedDye×Tumor effect (p-value = 0.051) are both negative. These negative effects suggest that the development of a tumor (which is the mediator in the analysis) and its combination with a high dose of Red 40 might accelerate the death of the mice. However, these negative effects do not lead to a statistically significant NIE (in Output 38.5.5) for the mediation analysis. The mediator model estimation results in Output 38.5.7 might provide a hint as to why this is the case.

Output 38.5.7 Mediator Model Estimation

Mediator Model Estimates							
Parameter		Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		-1.6859	0.3553	-2.3823	-0.9896	22.5196	<.0001
Female	0	-1.0020	0.4740	-1.9311	-0.07296	4.4685	0.0345
Female	1	0	.	.	.	.	.
RedDye	3	0.1944	0.4418	-0.6715	1.0603	0.1936	0.6599
RedDye	0	0	.	.	.	.	.

In Output 38.5.7, a high dose of RedDye does not have a significant effect on Tumor. Therefore, this weak linkage between RedDye and Tumor did not lead to a meaningful indirect effect (NIE) of Red 40 on the time to death via the tumor development.

Finally, instead of fitting the accelerated failure time model to the data, you can fit a Cox proportional hazards model for the time-to-event outcome to the same data. To do this, you use almost the same specification as in the preceding analysis, except that now you use the COXPH option in the MODEL statement, as follows:

```
proc causalmed data=RedDye ciratio=log poutcomemod pmedmod;
   class RedDye Female Tumor;
   model Weeks*Censor(1) = RedDye | Tumor / coxph;
   mediator Tumor(event='1') = RedDye(treat='3');
   covar Female;
   freq Freq;
run;
```

When you use the proportional hazards model, the causal effects in the output are expressed as hazard ratios. Output 38.5.8 shows the summary of the causal mediation analysis.

Output 38.5.8 Summary of Causal Mediation Effects Using Cox Proportional Hazards Model

Summary of Effects						
	Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Hazard Ratio Total Effect	0.9444	0.2808	0.5274	1.6913	-0.19	0.8474
Hazard Ratio Controlled Direct Effect (CDE)	0.6146	0.1921	0.3331	1.1340	-1.56	0.1193
Hazard Ratio Natural Direct Effect (NDE)	0.8777	0.2412	0.5122	1.5040	-0.47	0.6349
Hazard Ratio Natural Indirect Effect (NIE)	1.0761	0.1802	0.7750	1.4940	0.44	0.6615
Total Excess Hazard Ratio	-0.05559	0.2808	-0.6059	0.4947	-0.20	0.8430
Excess Hazard Ratio Due to CDE	-0.3287	0.1769	-0.6754	0.01800	-1.86	0.0631
Excess Hazard Ratio Due to NDE	-0.1223	0.2412	-0.5951	0.3504	-0.51	0.6120
Excess Hazard Ratio Due to NIE	0.06675	0.1561	-0.2393	0.3728	0.43	0.6690
Percentage Mediated	-120.07	788.41	-1665.32	1425.18	-0.15	0.8790
Percentage Due to Interaction	-441.30	2513.97	-5368.59	4485.99	-0.18	0.8607
Percentage Eliminated	-491.28	2811.86	-6002.43	5019.86	-0.17	0.8613

Because a higher hazard ratio implies a shorter expected survival time, the causal mediation effects that are expressed as hazard ratios (that is, in Output 38.5.8) would show the opposite directions as those that are expressed as mean ratios of survival time (that is, in Output 38.5.5). For example, whereas the hazard ratios for the total effect, CDE, and NDE in Output 38.5.8 are all less than 1, the corresponding mean ratios for the total effect, CDE, and NDE in Output 38.5.5 are all greater than 1. Similarly, whereas the excess hazard ratios for the total effect, CDE, and NDE in Output 38.5.8 are all less than 0, the corresponding excess mean ratios for the total effect, CDE, and NDE in Output 38.5.5 are all greater than 0. Therefore, these two sets of causal mediation results are in fact consistent.

References

Baron, R. M., and Kenny, D. A. (1986). "The Moderator-Mediator Variable Distinction in Social Psychological Research: Conceptual, Strategic, and Statistical Considerations." *Journal of Personality and Social Psychology* 51:1173–1182.

Bollen, K. A. (1989). *Structural Equations with Latent Variables*. New York: John Wiley & Sons.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220. With discussion.

Hong, G. (2015). *Causality in a Social World: Moderation, Mediation, and Spill-Over*. New York: John Wiley & Sons.

Imai, K., Keele, L., Tingley, D., and Yamamoto, T. (2010). "Causal Mediation Analysis Using R." In *Advances in Social Science Research Using R*, edited by H. D. Vinod, 129–154. New York: Springer.

Kalbfleisch, J. D., and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley & Sons.

Lagakos, S., and Mosteller, F. (1981). "A Case Study of Statistics in the Regulatory Process: The FD&C Red No. 40 Experiments." *Journal of the National Cancer Institute* 66:197–212.

Lawless, J. F. (2003). *Statistical Model and Methods for Lifetime Data*. 2nd ed. New York: John Wiley & Sons.

Marjoribanks, K., ed. (1974). *Environments for Learning*. London: National Foundation for Educational Research Publications.

Pearl, J. (2001). "Direct and Indirect Effects." In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, edited by J. Breese and D. Koller, 411–420. San Francisco: Morgan Kaufmann.

Pearl, J. (2009). *Causality: Models, Reasoning, and Inference*. 2nd ed. Cambridge: Cambridge University Press.

Robins, J. M., and Greenland, S. (1992). "Identifiability and Exchangeability for Direct and Indirect Effects." *Epidemiology* 3:143–155.

Valeri, L., and VanderWeele, T. J. (2013). "Mediation Analysis Allowing for Exposure-Mediator Interactions and Causal Interpretation: Theoretical Assumptions and Implementation with SAS and SPSS Macros." *Psychological Methods* 18:137–150.

Valeri, L., and VanderWeele, T. J. (2015). "SAS Macro for Causal Mediation Analysis with Survival Data." *Epidemiology* 26:e23–e24.

VanderWeele, T. J. (2011). "Causal Mediation Analysis with Survival Data." *Epidemiology* 22:582–585.

VanderWeele, T. J. (2014). "A Unification of Mediation and Interaction: A 4-Way Decomposition." *Epidemiology* 25:749–761.

VanderWeele, T. J. (2015). *Explanation in Causal Inference: Methods for Mediation and Interaction*. New York: Oxford University Press.

VanderWeele, T. J., and Vansteelandt, S. (2009). "Conceptual Issues Concerning Mediation, Interventions and Compositions." *Statistics and Its Interface* 2:457–468.

VanderWeele, T. J., and Vansteelandt, S. (2010). "Odds Ratios for Mediation Analysis for a Dichotomous Outcome." *American Journal of Epidemiology* 172:1339–1348.

Chapter 39
The CAUSALTRT Procedure

Contents

Overview: CAUSALTRT Procedure **2504**
- Features of the CAUSALTRT Procedure 2504
- Outline of Estimation Method Requirements 2505

Getting Started: CAUSALTRT Procedure **2507**

Syntax: CAUSALTRT Procedure **2510**
- PROC CAUSALTRT Statement 2511
- BOOTSTRAP Statement 2519
- BY Statement 2522
- CLASS Statement 2522
- FREQ Statement 2524
- MODEL Statement 2524
- OUTPUT Statement 2528
- PSMODEL Statement 2529

Details: CAUSALTRT Procedure **2533**
- Causal Effects: Definitions, Assumptions, and Identification 2533
- Estimating the Average Treatment Effect (ATE) 2536
- Estimating the Average Treatment Effect for the Treated (ATT) 2541
- Comparing Weights for Estimating the ATE and ATT 2543
- Standard Errors and Confidence Intervals 2543
- Missing Values 2547
- ODS Table Names 2547
- ODS Graphics 2548

Examples: CAUSALTRT Procedure **2549**
- Example 39.1: Estimating the ATE Using Inverse Probability Weights 2549
- Example 39.2: Estimating the ATE Using Augmented Inverse Probability Weights . . 2554
- Example 39.3: Estimating Treatment Effects Using Regression Adjustment 2556

References **2560**

Overview: CAUSALTRT Procedure

The CAUSALTRT procedure estimates the average causal effect of a binary treatment, T, on a continuous or discrete outcome, Y. Although the causal effect that is defined and estimated in PROC CAUSALTRT is called a treatment effect, it is not confined to effects that result from controllable treatments (such as effects in an experiment). Depending on the application, the binary treatment variable T can represent an intervention (such as smoking cessation versus control), an exposure to a condition (such as attending a private versus public school), or an existing characteristic of subjects (such as high versus low socioeconomic status). The CAUSALTRT procedure can estimate two types of causal effects: the average treatment effect (ATE) and the average treatment effect for the treated (ATT). For more information about the causal effects that the CAUSALTRT procedure can estimate, see the section "Causal Effects: Definitions, Assumptions, and Identification" on page 2533.

The CAUSALTRT procedure implements causal inference methods that are designed primarily for use with data from nonrandomized trials or observational studies. In an observational study, you observe the treatment T and the outcome Y without assigning subjects randomly to the treatment conditions. Instead, subjects "select" themselves into the treatment conditions according to their pretreatment characteristics. If these pretreatment characteristics are also associated with the outcome Y, they induce a specious relationship between T and Y and hence cloud the causal interpretation of T on Y. Therefore, estimating the causal effect of T in observational studies usually requires adjustments that remove or counter the specious effects that are induced by the confounding variables.

To adjust for the effects of the confounding variables, you can model either the treatment assignment T or the outcome Y, or both. Modeling the treatment leads to inverse probability weighting methods, and modeling the outcome leads to regression adjustment methods. Combined modeling of the treatment and outcome leads to doubly robust methods that can provide unbiased estimates for the treatment effect even if one of the models is misspecified. For more information about which model specifications are required for different estimation methods and how default estimation methods are determined, see the section "Outline of Estimation Method Requirements" on page 2505. For an introduction to causal inference from nonrandomized data, see Imbens and Rubin (2015); Morgan and Winship (2015); Berzuini, Dawid, and Bernardinelli (2012).

Features of the CAUSALTRT Procedure

The CAUSALTRT procedure provides the following methods to estimate causal effects:

- inverse probability weighting methods
- regression adjustment
- doubly robust methods

You can estimate two types of causal effects:

- average treatment effect (ATE), also sometimes called the average causal effect (ACE)
- average treatment effect for the treated (ATT or ATET)

For more information about the causal effects that the CAUSALTRT procedure can estimate, see the section "Causal Effects: Definitions, Assumptions, and Identification" on page 2533.

PROC CAUSALTRT computes standard errors and confidence intervals for the causal effects by the following methods:

- asymptotic methods
- bootstrap methods (which you request by specifying the BOOTSTRAP statement)

For more information about how the CAUSALTRT procedure computes standard errors and confidence limits, see the section "Standard Errors and Confidence Intervals" on page 2543.

PROC CAUSALTRT provides the following types of graphical output:

- diagnostic plots for the propensity score model, including various plots of propensity scores or weights
- histograms for bootstrap estimates

Finally, you can save the propensity scores, inverse probability weights, and the predicted potential outcomes in a SAS data set.

Outline of Estimation Method Requirements

The CAUSALTRT procedure uses various estimation methods to adjust for the effects of confounding variables by fitting models for the treatment assignment T or the outcome Y, or both. For the estimation methods, you specify the outcome variable Y in the MODEL statement and the treatment assignment T in the PSMODEL statement.

Depending on the estimation method that you specify in the METHOD= option in the PROC CAUSALTRT statement, you must also provide specific additional modeling information in the MODEL and PSMODEL statements, as shown in Table 39.1.

Table 39.1 Estimation Method Requirements

METHOD=	Additional Specification
IPW, IPWR, or IPWS	A model for the treatment assignment in the PSMODEL statement.
REGADJ	A model for the outcome in the MODEL statement. If you do not specify an outcome model, PROC CAUSALTRT fits an intercept-only model separately for each treatment condition.
AIPW or IPWREG	Both an outcome model in the MODEL statement and a model for the treatment assignment in the PSMODEL statement.

The model for the treatment assignment is also called the propensity score model.

If you do not specify the METHOD= option in the PROC CAUSALTRT statement, the CAUSALTRT procedure determines a default estimation method based on which models are specified in the MODEL and

PSMODEL statements, according to following criteria:

- When only a treatment model is specified (in the PSMODEL statement), the default estimation method is IPWR. For example:

```
proc causaltrt;
   model y;
   psmodel trt = x1 x2;
run;
```

- When only an outcome model is specified (in the MODEL statement), the default estimation method is REGADJ. For example:

```
proc causaltrt;
   model y = x1 x2;
   psmodel trt;
run;
```

- When both the treatment and outcome models are specified, the default estimation method is AIPW. For example:

```
proc causaltrt;
   model y = x1 x2;
   psmodel trt = x1 x2;
run;
```

- When neither an outcome nor a treatment model is specified, the default estimation method is REGADJ with an intercept-only outcome model. For example:

```
proc causaltrt;
   model y;
   psmodel trt;
run;
```

If you specify the ATT option in the PROC CAUSALTRT statement, then you are estimating the average treatment effect for the treated (ATT) instead of the default average treatment effect (ATE). In this case, estimation methods are limited to either IPWR or regression adjustment. Therefore, if models for both the treatment assignment and outcome variable are specified without the METHOD=IPWR or METHOD=REGADJ option, PROC CAUSALTRT issues an error.

The CAUSALTRT procedure fits only the models necessary for the estimation method used. For example, if a model for the treatment assignment is specified but the REGADJ estimation method is requested, then PROC CAUSALTRT does not fit the treatment model and therefore does not produce output or displays that involve the propensity scores or weights.

For more information about the estimation methods that the CAUSALTRT procedure implements, see the sections "Estimating the Average Treatment Effect (ATE)" on page 2536 and "Estimating the Average Treatment Effect for the Treated (ATT)" on page 2541.

Getting Started: CAUSALTRT Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/ctrtgs.sas.)

This section illustrates some of the basic features of the CAUSALTRT procedure. This example uses data from a hypothetical nonrandomized trial.

Suppose that 486 patients in a trial are at risk for developing type 2 diabetes and are allowed to choose which of two preventive drugs they want to receive. For this study, the outcome of interest is whether a patient develops type 2 diabetes within five years. The nonrandom assignment of patients to treatment conditions does not control for confounding variables that might explain any observed difference between the treatment conditions. The hypothetical data set Drugs contains the following variables:

- Age: age at the start of the trial
- BMI: body mass index at the start of the trial
- Diabetes2: indicator for whether a patient developed type 2 diabetes, with values Yes and No
- Drug: indicator of treatment assignment, with values Drug_A and Drug_X
- Gender: gender

The first 10 observations in the Drugs data set are listed in Figure 39.1.

Figure 39.1 Drug Trial Data

Obs	Gender	Age	BMI	Drug	Diabetes2
1	Male	29	22.02	Drug_A	Yes
2	Female	30	23.83	Drug_X	Yes
3	Female	42	23.64	Drug_X	Yes
4	Male	47	23.67	Drug_A	No
5	Female	42	22.27	Drug_A	Yes
6	Male	45	26.90	Drug_A	Yes
7	Female	46	24.06	Drug_A	Yes
8	Female	45	23.41	Drug_A	Yes
9	Female	28	24.65	Drug_X	No
10	Male	36	20.13	Drug_X	Yes

The following statements invoke the CAUSALTRT procedure and request the estimation of the average treatment effect (ATE) by using the inverse probability weighting method with ratio adjustment (METHOD=IPWR):

```
proc causaltrt data=drugs method=ipwr ppsmodel;
   class Gender;
   psmodel Drug(ref='Drug_A') = Age Gender BMI;
   model Diabetes2(ref='No') / dist = bin;
run;
```

The results of this analysis are shown in the following figures. In Figure 39.2, PROC CAUSALTRT displays information about the estimation method used, the outcome variable, and the treatment variable.

Figure 39.2 Model Information

The CAUSALTRT Procedure

Model Information	
Data Set	WORK.DRUGS
Distribution	Binomial
Estimation Method	IPWR
Treatment Variable	Drug
Outcome Variable	Diabetes2

The METHOD= option in the PROC CAUSALTRT statement specifies the estimation method to be used. The outcome variable, Diabetes2, is specified in the MODEL statement, and the treatment variable, Drug, is specified in the PSMODEL statement.

The DIST=BIN option in the MODEL statement identifies the outcome as a binary variable, and the **`ref='No'`** option identifies No as the reference level. In the PSMODEL statement, the **`ref='Drug_A'`** option identifies Drug_A as the reference (control) condition when the treatment assignment is modeled and the ATE is estimated. The frequencies for the outcome and treatment variable categories are listed in the "Response Profile" (Figure 39.3) and "Treatment Profile" (Figure 39.4) tables, respectively.

Figure 39.3 Response Profile

Response Profile		
Ordered Value	**Diabetes2**	**Total Frequency**
1	Yes	287
2	No	199

Figure 39.4 Treatment Profile

Treatment Profile		
Ordered Value	**Drug**	**Total Frequency**
1	Drug_X	149
2	Drug_A	337

The IPWR estimation method uses inverse probability weighting to estimate the potential outcome means and ATE. The inverse probability weights are estimated by inverting the predicted probability of receiving treatment (that is, the Drug_X condition), which is estimated from a logistic regression model that is specified in the PSMODEL statement. The probability of receiving treatment is called the propensity score, and hence the regression model fit by the PSMODEL statement is also called the propensity score model. In this example, the predictors of the propensity score model are Age, Gender, and BMI. The PPSMODEL option in the PROC CAUSALTRT statement displays the parameter estimates for the propensity score model, as shown in Figure 39.5.

Figure 39.5 Propensity Score Model Estimates

The CAUSALTRT Procedure

Propensity Score Model Estimates							
Parameter		Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		-0.6226	1.2738	-3.1193	1.8740	0.2389	0.6250
Age		-0.0888	0.0170	-0.1220	-0.0556	27.4228	<.0001
Gender	Female	-0.2103	0.2055	-0.6130	0.1924	1.0474	0.3061
Gender	Male	0	.	.	.	.	.
BMI		0.1434	0.0517	0.0422	0.2447	7.7115	0.0055

The estimates for the potential outcome means (POM) and ATE are displayed in the "Analysis of Causal Effect" table (Figure 39.6). The ATE estimate of –0.1787 indicates that Drug_X is more effective on average than Drug_A in preventing the development of type 2 diabetes. The ATE is significantly different from 0 at the 0.05 α-level, as indicated by the 95% confidence interval (–0.2771, –0.0803) or the p-value (0.0004).

Figure 39.6 IPWR Potential Outcome Means and ATE Estimates

The CAUSALTRT Procedure

Analysis of Causal Effect							
Parameter	Treatment Level	Estimate	Robust Std Err	Wald 95% Confidence Limits		Z	Pr > |Z|
POM	Drug_X	0.4681	0.0431	0.3835	0.5526	10.85	<.0001
POM	Drug_A	0.6468	0.0265	0.5948	0.6988	24.38	<.0001
ATE		-0.1787	0.0502	-0.2771	-0.08031	-3.56	0.0004

The ATE of Drug_X compared to Drug_A can also be estimated by using augmented inverse probability weights (AIPW). In addition to a propensity score model, the AIPW estimation method incorporates a model for the outcome variable, Diabetes2, into the estimation of the potential outcome means and ATE. The AIPW estimation method is doubly robust and provides unbiased estimates for the ATE even if one of the outcome or treatment models is misspecified.

The following statements invoke the CAUSALTRT procedure and use the AIPW estimation method to estimate the ATE:

```
proc causaltrt data=drugs method=aipw;
   class Gender;
   psmodel Drug(ref='Drug_A') = Age Gender BMI;
   model Diabetes2(ref='No') = Age Gender BMI / dist = bin;
run;
```

For this example, the same set of effects is specified in both the MODEL and PSMODEL statements, but in general this need not be the case. The use of the AIPW estimation method for this PROC CAUSALTRT step is reflected in the "Model Information" table (Figure 39.7).

Figure 39.7 Model Information

The CAUSALTRT Procedure

Model Information	
Data Set	WORK.DRUGS
Distribution	Binomial
Link Function	Logit
Estimation Method	AIPW
Treatment Variable	Drug
Outcome Variable	Diabetes2

As shown in Figure 39.8, the AIPW estimate for the ATE is –0.1709, which is similar to the IPWR estimate of –0.1787. Moreover, the AIPW standard error estimate for ATE is slightly smaller than that of the IPWR. The AIPW 95% confidence interval for ATE is also slightly narrower than that of the IPWR.

Figure 39.8 AIPW Potential Outcome Means and ATE Estimates

The CAUSALTRT Procedure

Analysis of Causal Effect							
Parameter	Treatment Level	Estimate	Robust Std Err	Wald 95% Confidence Limits		Z	Pr > |Z|
POM	Drug_X	0.4782	0.0422	0.3954	0.5609	11.32	<.0001
POM	Drug_A	0.6491	0.0267	0.5968	0.7013	24.33	<.0001
ATE		-0.1709	0.0495	-0.2679	-0.07390	-3.45	0.0006

Syntax: CAUSALTRT Procedure

The following statements are available in the CAUSALTRT procedure. Items within < > are optional.

```
PROC CAUSALTRT <options> ;
    BOOTSTRAP < options > ;
    BY variables ;
    CLASS variables <(options)> ... <variable<(options)>> </ global-options> ;
    FREQ variable ;
    MODEL outcome <(variable-options)><= <effects>> </ model-options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=name ... keyword=name> ;
    PSMODEL treatment <(variable-options)><= effects </ psmodel-options>> ;
```

The PROC CAUSALTRT, MODEL, and PSMODEL statements are required. The CLASS statement, if present, must precede the MODEL and PSMODEL statements. All statements can appear only once.

All the estimation methods implemented in PROC CAUSALTRT require the specification of both an outcome and a treatment variable. You specify the outcome variable in the MODEL statement and the treatment variable in the PSMODEL statement. In the MODEL and PSMODEL statements, you can use the same syntax to specify main effects and interaction terms for modeling the outcome and treatment assignment. This syntax is described in the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure." For more information about which model specifications are required for different estimation

methods and how default estimation methods are determined, see the section "Outline of Estimation Method Requirements" on page 2505. The following sections describe the PROC CAUSALTRT statement and then describe the other statements in alphabetical order.

PROC CAUSALTRT Statement

PROC CAUSALTRT < *options* > **;**

The PROC CAUSALTRT statement invokes the CAUSALTRT procedure. Table 39.2 summarizes the *options* available in the PROC CAUSALTRT statement.

Table 39.2 PROC CAUSALTRT Statement Options

Option	Description
Input	
DATA=	Specifies the input data set
Response and Classification Variable Options	
DESCENDING	Sorts the outcome variable in reverse of the default order
NAMELEN=	Specifies the length of effect names
ORDER=	Specifies the sort order of the classification variables
RORDER=	Specifies the sort order of the outcome variable
Estimation and Analysis	
ATT	Estimates the average treatment effect for the treated
COVDIFFPS	Computes and displays standardized mean differences for the effects in the propensity score model
METHOD=	Specifies the estimation method
Displayed Output	
ALPHA=	Specifies the level for confidence limits
NOEFFECT	Suppresses all displayed output that involves the outcome variable
NOPRINT	Suppresses all displayed output
PALL	Displays all optional output
PLOTS	Produces ODS Graphics displays
POUTCOMEMOD	Displays outcome model parameter estimates
PPSMODEL	Displays propensity score model parameter estimates
Technical Details	
NLOPTIONS	Specifies optimization parameters for fitting the specified models
SINGULAR=	Specifies the singularity tolerance
THREADS=	Specifies the number of threads for the computation

You can specify the following *options*.

ALPHA=*number*

specifies a *number* to be used as the α level for $100(1-\alpha)\%$ confidence limits in the "Analysis of Causal Effect" table. The *number* must be between 0 and 1. This *number* is also used as the default level for propensity score and outcome model confidence limits. For the propensity score and outcome models, you can override this default level by specifying the ALPHA= options in the MODEL and PSMODEL statements, respectively. By default, ALPHA=0.05, which results in 95% confidence intervals.

ATT

ATET

estimates the average treatment effect for the treated. When this option is applied, it replaces the default estimation of the average treatment effect (ATE). For more information about the estimation methods implemented in the CAUSALTRT procedure and for comparisons between the average treatment effect and average treatment effect for the treated, see the sections "Estimating the Average Treatment Effect for the Treated (ATT)" on page 2541 and "Causal Effects: Definitions, Assumptions, and Identification" on page 2533. This option can be used only when METHOD=IPWR or REGADJ.

COVDIFFPS

computes weighted and unweighted standardized mean differences (between treatment and control conditions) and variance ratios (treatment to control) for the covariates (effects) in the propensity score model. This option is supported only for estimation methods that fit a propensity score model.

The results are displayed in the "Covariate Differences for Propensity Score Model" table. This table also includes columns for the weighted and unweighted mean and variance for propensity score model effects within each treatment condition; these columns are not displayed but are accessible if you save the table as an output data set by specifying the ODS OUTPUT statement. You can display these columns by modifying the corresponding template.

DATA=*SAS-data-set*

names the SAS data set that contains the data to be analyzed. If you omit this option, the procedure uses the most recently created SAS data set.

DESCENDING

DESCEND

DESC

sorts the levels of the outcome variable for a binary model in reverse of the specified order.

METHOD= AIPW | IPW | IPWR | IPWS | REGADJ | IPWREG

specifies the method to use to estimate the potential outcomes and treatment effect. You can specify one of the following values:

AIPW performs a doubly robust estimation by using augmented inverse probability weighting. You must specify a model for the outcome variable in the MODEL statement and a model for the treatment assignment in the PSMODEL statement.

IPW uses a basic inverse probability weighting method. You must specify a model for the treatment assignment in the PSMODEL statement.

IPWR uses an inverse probability weighting method with ratio adjustment. You must specify a model for the treatment assignment in the PSMODEL statement.

IPWS uses an inverse probability weighting method with ratio and scale adjustments. You must specify a model for the treatment assignment in the PSMODEL statement.

IPWREG performs a doubly robust estimation by using inverse probability weighted regression adjustment. You must specify a model for the outcome variable in the MODEL statement and a model for the treatment assignment in the PSMODEL statement.

REGADJ uses regression adjustment. You must specify a model for the outcome variable in the MODEL statement.

For all estimation methods, you specify the outcome variable in the MODEL statement and the treatment variable in the PSMODEL statement. For more information about the estimation methods that the CAUSALTRT procedure implements, see the sections "Estimating the Average Treatment Effect (ATE)" on page 2536 and "Estimating the Average Treatment Effect for the Treated (ATT)" on page 2541.

NAMELEN=*n*

specifies the maximum length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 128. By default, NAMELEN=20.

NLOPTIONS(*nlo-options***)**

specifies options for the nonlinear optimization methods that are used for fitting the specified models. You can specify one or more of the following *nlo-options* separated by spaces:

ABSCONV=*r*

ABSTOL=*r*

specifies an absolute function convergence criterion by which minimization stops when $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of *r* is the negative square root of the largest double-precision value.

ABSFCONV=*r*

ABSFTOL=*r*

specifies an absolute function difference convergence criterion. Termination requires a small change of the function value in successive iterations,

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. By default, ABSFCONV=0.

ABSGCONV=*r*

ABSGTOL=*r*

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small,

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the *j*th parameter. By default, ABSGCONV=1E–7.

FCONV=*r*

FTOL=*r*

specifies a relative function convergence criterion. Termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. By default, FCONV=$10^{-\mathrm{FDIGITS}}$, where by default FDIGITS is $-\log_{10}\{\epsilon\}$, where ϵ is the machine precision.

GCONV=*r*

GTOL=*r*

specifies a relative gradient convergence criterion. For all values of the TECHNIQUE= suboption except CONGRA, termination requires the normalized predicted function reduction to be small,

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. When TECHNIQUE=CONGRA (for which a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \; |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

By default, GCONV = 1E–8.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values are as follows, depending on the value of the TECHNIQUE= suboption:

- TRUREG, NRRIDG, and NEWRAP: 125
- QUANEW and DBLDOG: 500
- CONGRA: 1,000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed *n*.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on the value of the TECHNIQUE= suboption:

- TRUREG, NRRIDG, and NEWRAP: 50
- QUANEW and DBLDOG: 200
- CONGRA: 400

These default values also apply when *n* is specified as a missing value.

MAXTIME=*r*
specifies an upper limit of *r* seconds of CPU time for the optimization process. Because the time is checked only at the end of each iteration, the actual run time might be longer than *r*. By default, CPU time is not limited.

TECHNIQUE=CONGRA | DBLDOG | NEWRAP | NRRIDG | QUANEW | TRUREG
specifies the optimization technique to obtain maximum likelihood estimates. You can specify from the following values:

CONGRA	performs a conjugate-gradient optimization.
DBLDOG	performs a version of double-dogleg optimization.
NEWRAP	performs a Newton-Raphson optimization that combines a line-search algorithm with ridging.
NRRIDG	performs a Newton-Raphson optimization with ridging.
QUANEW	performs a dual quasi-Newton optimization.
TRUREG	performs a trust-region optimization.

By default, TECHNIQUE=NEWRAP.

For more information about these optimization methods, see the section "Choosing an Optimization Algorithm" on page 545 in Chapter 20, "Shared Concepts and Topics."

NOEFFECT
suppresses the display of all output that involves the outcome variable. This option is useful for investigating the balance of covariates between treatment conditions by exploring different propensity score models before displaying estimates for the causal effect. This option is effective only when METHOD=IPW, IPWR, or IPWS.

NOPRINT
suppresses all displayed output. This option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PALL

ALL

displays all optional output.

PLOTS< (*global-plot-options*) >< =*plot-request* >

PLOTS < (*global-plot-options*) >=(*plot-request* < . . . *plot-request* >)

controls plots that are produced through ODS Graphics. For more information about controlling specific plots, see also the PLOTS options in the BOOTSTRAP and PSMODEL statements.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc causaltrt plots=all method=ipwr;
   model y;
   psmodel trt = x1 x2;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *plot-requests*:

ALL

produces all plots that are available for the specified estimation method.

BOOTHIST

produces histograms of the bootstrap estimates for the potential outcome means and treatment effect, which are displayed in a panel by default. This option is ignored if the BOOTSTRAP statement is not specified.

LOGITPSCORE

LPS

produces overlaid density plots for the logit of the propensity score within each treatment condition. These plots are not produced when METHOD=REGADJ.

NONE

suppresses all plots. If you specify this *plot-request*, then all plot requests that are specified in the BOOTSTRAP and PSMODEL statements are ignored.

OUTBYPSCORE

OUTBYPS

produces a scatter plot of the outcome variable by the propensity score for nonbinary outcomes. If the outcome is binary, box plots of the propensity scores within treatment conditions are produced for each outcome level. The whisker lengths for each box plot are determined by the maximum and minimum of the propensity scores. This plot is not produced when METHOD=REGADJ.

OUTBYWEIGHT

OUTBYWGT

produces a scatter plot of the outcome variable by weight for nonbinary outcomes. If the outcome is binary, box plots of the weights within treatment conditions are produced for each outcome level. The whisker lengths for each box plot are determined by the maximum and minimum of the weights. This plot is not produced when METHOD=REGADJ.

PSCLOUD

produces a point cloud of the propensity scores by jittering within the control and treatment conditions. This plot is not produced when METHOD=REGADJ.

PSCOVDEN

produces density plots for the covariates or continuous effects that are specified in the PSMODEL statement. Each plot displays the density of an effect for the treatment and control conditions. Two plots are produced for each effect: one plot displays unweighted densities, and the other plot displays densities that are weighted by inverse probability weights. By default, plots are produced for all continuous effects in the PSMODEL statement and are collected in panels. You can customize the density plots by using the PLOTS= option in the PSMODEL statement. This option is ignored when METHOD=REGADJ.

PSDIST

produces a box plot of the propensity score for each treatment condition. The whisker lengths for each box plot are determined by the maximum and minimum of the propensity scores. This plot is not produced when METHOD=REGADJ.

WEIGHTCLOUD

WCLOUD

produces a point cloud of the weights by jittering within the control and treatment conditions. This plot is not produced when METHOD=REGADJ.

WEIGHTDIST

WDIST

produces a box plot of the weights for each treatment condition. The whisker lengths for each box plot are determined by the maximum and minimum of the weights. This plot is not produced when METHOD=REGADJ.

You can specify the following *global-plot-options*:

UNPACK

UNPACKPANEL

suppresses paneling. By default, multiple plots can appear in the same output panel. You can use this option to display each plot separately.

POUTCOMEMOD

PREGADJ

displays parameter estimates for the outcome models for the control and treatment conditions.

PPSMODEL

PTREATMOD

displays parameter estimates for the propensity score model.

RORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the outcome variable. In order for this option to apply, either the outcome variable must be specified in the CLASS statement or the DIST=BIN option must be specified in the MODEL statement. The following table shows how PROC CAUSALTRT interprets values of the RORDER= option.

Value of RORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.
FREQ	Descending frequency count. Levels that have the most observations come first in the order.
INTERNAL	Unformatted value. The sort order is machine-dependent.

By default, RORDER=FORMATTED. The DESCENDING option in the PROC CAUSALTRT statement causes the response variable to be sorted in reverse of the order displayed in the previous table. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide*.

SINGULAR=*tolerance*
specifies the *tolerance* for testing the singularity of a matrix, where *tolerance* must be between 0 and 1. By default, *tolerance* is 1E7 times the machine epsilon.

THREADS=*n*
NTHREADS=*n*
specifies the number of threads for analytic computations and overrides the SAS system option THREADS | NOTHREADS. If you do not specify the THREADS= option or if you specify THREADS=0, the number of threads is determined from the number of CPUs in the host on which the analytic computations execute.

BOOTSTRAP Statement

BOOTSTRAP < *options* > **;**

A BOOTSTRAP statement requests bootstrap estimation of confidence intervals for the potential outcome means and treatment effect estimates. The bootstrap samples are taken from within the treatment conditions. To produce the plots that are controlled by the BOOTSTRAP statement, ODS Graphics must be enabled. PROC CAUSALTRT ignores the BOOTSTRAP statement if the initial estimation of the treatment effect results in an error or if the NOEFFECT option is specified in the PROC CAUSALTRT statement.

Table 39.3 summarizes the *options* available in the BOOTSTRAP statement.

Table 39.3 Summary of Options in BOOTSTRAP Statement

Option	Description
BOOTCI	Produces bootstrap confidence intervals of the estimates
BOOTDATA=	Specifies the bootstrap output data set
NBOOT=	Specifies the number of bootstrap sample data sets (replicates)
NOSKIP	Includes bootstrap samples that have inconsistent class levels
PLOTS=	Produces plots of the bootstrap estimates
SEED=	Specifies the seed that initializes the random number stream

You can specify the following *options*:

BOOTCI <(BC | NORMAL | PERC | ALL)>
computes bootstrap-based confidence intervals for the potential outcome means and the treatment effect estimates, and displays the values in the "Analysis of Causal Effect" table. This table includes a column that indicates the number of bootstrap samples used to compute the confidence intervals; this column is not displayed but is available if you save the table as an output data set by using the ODS OUTPUT statement. You can also display this column by modifying the corresponding template.

You can specify one or more of the following types of bootstrap confidence intervals separated by spaces:

BC produces bias-corrected confidence intervals. You must specify a value of 1,000 or more for the NBOOT= option, but the confidence intervals are not computed if

fewer than 900 bootstrap replicates produce bootstrap estimates.

NORMAL produces confidence intervals that are based on the assumption that bootstrap estimates follow a normal distribution. You must specify a value of 50 or more for the value NBOOT= option, but the corresponding standard errors and confidence intervals are not computed if fewer than 40 bootstrap replicates produce bootstrap estimates.

PERC produces percentile-based confidence intervals. You must specify a value of 1,000 or more for the NBOOT= option, but the confidence intervals are not computed if fewer than 900 bootstrap replicates produce bootstrap estimates.

ALL produces all three confidence intervals.

The ALPHA= option in the PROC CAUSALTRT statement sets the level of significance that is used to construct the bootstrap confidence intervals. For more information about how the bootstrap-based confidence intervals are computed, see the section "Bootstrap Methods" on page 2545. By default, BOOTCI=BC and PROC CAUSALTRT produces bias-corrected confidence intervals based on 1,000 bootstrap samples.

BOOTDATA=*SAS-data-set*

specifies the *SAS-data-set* that contains the estimates for the potential outcome means and treatment effect for each bootstrap sample that converges.

NBOOT=*n*

NSAMPLE=*n*

NSAMPLES=*n*

specifies the number of bootstrap sample data sets (replicates). A maximum of 10,000 bootstrap sample data sets can be requested. By default, NBOOT=1000.

NOSKIP

includes in the bootstrap estimation bootstrap samples that have inconsistent class levels in either the treatment or outcome model. By default, for any classification variable in a bootstrap sample data set that does not contain all levels that have been used in fitting either the treatment or outcome model for the original input data set, PROC CAUSALTRT treats the corresponding bootstrap sample estimates as nonconvergent and skips the estimation of the treatment effect. This option overrides the default and continues the estimation of the treatment effect for bootstrap samples that have inconsistent class levels.

PLOTS < (*global-plot-options*) >< =*plot-request*< (*options*) > >

produces ODS graphics of bootstrap estimates. By default, if you specify the PLOTS option, a panel is produced with the histograms of the bootstrap estimates for the potential outcome means and treatment effect. No graphics are produced if you specify the PLOTS=NONE option in the PROC CAUSALTRT statement.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc causaltrt method=ipwr;
   model y;
```

```
    psmodel trt = x1 x2;
    bootstrap plot;
  run;

  ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *plot-requests*:

HIST<(EFFECT | POMCNT | POMTRT)>

HISTOGRAM<(EFFECT | POMCNT | POMTRT)>

produces histograms of the bootstrap estimates for the potential outcome means and treatment effect, which are displayed in a panel by default. You can request output of specific histograms by specifying one or more the following additional suboptions:

EFFECT produces a histogram of the bootstrap estimates for the treatment effect.

POMCNT produces a histogram of the bootstrap estimates for the potential outcome mean of the control condition.

POMTRT produces a histogram of the bootstrap estimates for the potential outcome mean of the treatment condition.

If you specify the ATT option in the PROC CAUSALTRT statement, then the histogram produced by the EFFECT suboption is the average treatment effect for the treated, and the histograms produced by the POMCNT and POMTRT suboptions are the potential outcome means conditioned on having received treatment. For more information about the differences between the average treatment effect and average treatment effect for the treated, see the section "Causal Effects: Definitions, Assumptions, and Identification" on page 2533.

NONE

suppresses all plots that are usually produced by the BOOTSTRAP statement. You can use this option to suppress output if the PLOTS=ALL option is specified in the PROC CAUSALTRT statement.

You can specify the following *global-plot-option*:

UNPACK

UNPACKPANEL

suppresses paneling. By default, multiple histograms of bootstrap estimates appear in the same output panel. You can use this option to display each histogram separately.

SEED=*n*

provides the seed that initializes the random number stream for generating the bootstrap sample data sets (replicates). If you do not specify this option or if you specify a value for *n* that is less than or equal to 0, the seed is generated from reading the time of day from the computer's clock. The largest possible value for the seed is $2^{31} - 1$.

You can use the SYSRANDOM and SYSRANEND macro variables after a PROC CAUSALTRT step to query the initial and final seed values. However, using the final seed value as the starting seed for a

subsequent analysis does not continue the random number stream where the previous analysis ended. The SYSRANEND macro variable provides a mechanism to pass on seed values to ensure that the sequence of random numbers is the same every time you run an entire program. To reproduce the random number stream that was used to generate bootstrap estimates, you must use the same SEED= value and the same number of threads for the analytic computations. When bootstrap resampling is performed, a column that indicates the number of threads used is added to the "Analysis of Causal Effect" table; this column is not displayed but is available if you use the ODS OUTPUT statement to save the table as an output data set. You can also display this column by modifying the corresponding template.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC CAUSALTRT to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the CAUSALTRT procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis.

The CLASS statement must precede the MODEL and PSMODEL statements. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables specified in the CLASS statement. However, individual CLASS variable *options* override the *global-options*. Unless otherwise indicated, you can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

uses at most the first *n* characters of a CLASS variable name in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING

DESC

reverses the sort order of the classification variable. If both the DESCENDING and ORDER= options are specified, PROC CAUSALTRT orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

uses at most the first *n* characters of a CLASS variable label in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (blanks for character variables and ., ._, .A, . . . , .Z for numeric variables) as valid values for the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data.

The following table shows how PROC CAUSALTRT interprets values of the ORDER= option.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted values, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) values. The sort order is machine-dependent.
FREQ	Descending frequency count. Levels that have more observations come earlier in the order.
INTERNAL	Unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REF='*level***' | FIRST | LAST**

specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates that have a singular parameterization.

You can specify the following values:

'*level*' specifies the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. You cannot specify '*level*' as a *global-option*.

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n* >

specifies the length (*n*) of variable values to use in determining the CLASS *variable* levels. The default is to use the full formatted length of the CLASS variable. If you specify this option without the length *n*, the first 16 characters of the formatted values are used. When formatted values are longer than 16 characters, you can use this option to revert to the levels as determined in releases before SAS 9. The TRUNCATE option is available only as a *global-option*.

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a variable in the input data set that contains the frequency of occurrence of each observation. PROC CAUSALTRT treats each observation as if it appeared *n* times, where *n* is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If it is less than 1 or missing, the observation is not used.

MODEL Statement

MODEL *outcome* < *(variable-options)* >< **=** *effects* > < / *model-options* > **;**

For all the estimation methods that PROC CAUSALTRT implements, you must specify the *outcome* variable in the MODEL statement. For more information about the estimation methods implemented in PROC CAUSALTRT and the model specifications that are required, see the section "Outline of Estimation Method Requirements" on page 2505.

In addition to specifying the *outcome* variable, you can also specify the following:

effects

specify one or more explanatory variables or a combination of variables that are used to fit regression models for the *outcome* variable within each treatment condition. Explanatory variables that represent nominal (classification) data must be declared in the CLASS statement. For more information about specifying *effects*, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

variable-options

specify one or more of the following options within parentheses for a binary *outcome* variable immediately after the variable:

DESCENDING
DESC

reverses the order of the outcome categories. If both the DESCENDING and ORDER= options are specified, PROC CAUSALTRT orders the outcome categories according to the ORDER= option and then reverses that order.

EVENT='*category*' | FIRST | LAST

specifies the event category for the binary outcome model. PROC CAUSALTRT models the probability of the event category. You can specify one of the following:

'*category*' specifies the value (formatted if a format is applied) of the event category in quotation marks.

FIRST designates the first-ordered category as the event.

LAST designates the last-ordered category as the event.

By default, EVENT=FIRST.

One of the most common sets of outcome levels is {0,1}, where 1 represents the event for which the probability is to be modeled. Consider the following example, where Y takes the values 1 and 0 for event and nonevent, respectively, and X is the explanatory variable. To specify the value 1 as the event category, use the following MODEL statement:

```
model Y(event='1') = X;
```

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the outcome variable. The following table displays the available ORDER= options.

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.
FREQ	Descending frequency count. Levels that have the most observations come first in the order.
INTERNAL	Unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category***' | FIRST | LAST**

REF='*category***' | FIRST | LAST**

specifies the reference category for the binary outcome model. Specifying one outcome category as the reference is the same as specifying the other outcome category as the event category. You can specify one of the following:

'*category***'** specifies the value (formatted if a format is applied) of the reference category in quotation marks.

FIRST designates the first-ordered category as the reference.

LAST designates the last-ordered category as the reference.

By default, REF=LAST.

model-options

specify additional options for the outcome model after a slash (/). These *model-options* are summarized in Table 39.4 and described in detail after the table.

Table 39.4 *model-options* in the MODEL Statement

model-option	**Description**
ALPHA=	Specifies the level for confidence limits
DIST=	Specifies the probability distribution
LINK=	Specifies the link function
NOSCALE	Holds the scale parameter fixed
SCALE=	Specifies the value used for the scale

ALPHA=*number*

specifies a *number* to use as the α level for $100(1-\alpha)\%$ confidence limits that the MODEL statement computes. The value of *number* must be between 0 and 1. The default value of *number* is the set by the ALPHA= option in the PROC CAUSALTRT statement.

DIST=*keyword*

DISTRIBUTION=*keyword*

specifies the built-in probability distribution to use in the model. If you specify the DIST= option and you omit the LINK= option, a default link function is chosen as displayed in Table 39.5. If you specify neither the DIST= option nor the LINK= option, then the CAUSALTRT procedure defaults to the binomial distribution with logit link if the outcome variable is listed in the CLASS statement. If the outcome variable is not listed in the CLASS statement, then the CAUSALTRT procedure defaults to the normal distribution with the identity link function.

Table 39.5 Distributions and Default Link Functions

DIST=	Distribution	Default Link Function
BIN | **B**	Binary	Logit
GAMMA | **GAM** | **G**	Gamma	Reciprocal
IGAUSSIAN | **IG**	Inverse Gaussian	Reciprocal square
NORMAL | **NOR** | **N**	Normal	Identity
POISSON | **POI** | **P**	Poisson	Log

Responses for the inverse Gaussian and gamma distributions must be strictly positive. For the Poisson distribution, responses must be nonnegative, but they can take noninteger values. Observations whose response values are outside of the distribution's support are not used to estimate the causal effect, even if the estimation method does not fit an outcome model.

LINK=*keyword*

specifies the link function in the model. You can specify the *keywords* shown in Table 39.6.

Table 39.6 Built-In Link Functions of the CAUSALTRT Procedure

LINK=	Link Function	$g(\mu) = \eta =$
CLOGLOG | **CLL**	Complementary log-log	$\log(-\log(1-\mu))$
IDENTITY | **ID**	Identity	μ
INVERSE | **RECIPROCAL**	Reciprocal	$1/\mu$
LOG	Log	$\log(\mu)$
LOGIT	Logit	$\log(\mu/(1-\mu))$
POWERMINUS2	Power with exponent –2	$1/\mu^2$
PROBIT	Probit	$\Phi^{-1}(\mu)$

For the probit link, $\Phi^{-1}(\cdot)$ denotes the quantile function of the standard normal distribution. By default, the link function is chosen as shown in Table 39.5.

NOSCALE

holds the scale parameter fixed. If you omit the SCALE= option, the scale parameter is fixed at the value 1. If you do not specify the NOSCALE option, the scale parameter is estimated by maximum likelihood for the normal, inverse Gaussian, and gamma distributions.

SCALE=*number*

specifies the value used for the scale parameter when the NOSCALE option is specified. For the binomial and Poisson distributions, which have no free scale parameter, you can use this option to specify an overdispersed model. If the NOSCALE option is not specified, then *number* is used as an initial estimate of the scale parameter.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*> < *keyword*=*name* . . . *keyword*=*name*> **;**

The OUTPUT statement creates a new SAS data set that contains all the variables in the input data set and optionally contains predicted values that are used to estimate the treatment effect. If the estimation of the causal effect requires that a regression model for the outcome variable be fit, then you can request that the predicted potential outcomes be included in the output data set. You can request that the predicted propensity scores and inverse probability weights be included if the estimation of the causal effect fits a model for the treatment assignment.

The output data set contains estimated potential outcomes and propensity scores for all observations in which the explanatory variables for the model are not missing, regardless of whether the outcome is missing.

This behavior enables you to predict potential outcomes and propensity scores of "new" observations without affecting the original model fit. These new observations must have nonmissing explanatory variable values and treatment assignment, but their outcome values are missing in the input data set. This way, these new observations cannot be used for model fitting but their predicted values are computed in the OUT= data set.

You can specify the following options:

OUT=*SAS-data-set*

names the output data set. If you omit this option, an output data set is created and given a default name that uses the DATA*n* convention.

keyword=name

specifies a statistic to be included in the output data set and assigns the specified *name* to the new variable that contains the statistic. Specify one or more of the following *keywords* and *names* (connect each *keyword* and *name* by an equal sign):

IPW< (ATT) >=*name*

requests the predicted inverse probability weight. By default, the ATE weight is included. If you estimate the ATT, you can use the ATT option to specify the ATT weight. For more information about the ATE and ATT weights, see the section "Comparing Weights for Estimating the ATE and ATT" on page 2543.

POCNT=*name*

PREDCNT=*name*

requests the predicted potential outcome for the control condition.

POTRT=*name*

PREDTRT=*name*

requests the predicted potential outcome for the treatment condition.

PSCORE=*name*

PS=*name*

requests the predicted propensity score.

If you do not specify any *keyword=name* options, the output data set contains only the original variables.

PSMODEL Statement

PSMODEL *treatment* < *(variable-options)* >< **=** *effects* < / *psmodel-options* > > **;**

For all the estimation methods that PROC CAUSALTRT implements, you must specify a *treatment* variable in a PSMODEL statement. The regression model that is fit by the PSMODEL statement is called the propensity score model. The propensity score refers to the probability of receiving treatment conditional on the model *effects*. For more information about the estimation methods that PROC CAUSALTRT implements, see the section "Estimating the Average Treatment Effect (ATE)" on page 2536.

In addition to specifying the *treatment* variable, you can also specify the following:

effects

specify one or more explanatory variables or a combination of variables that are used to fit a regression model for the *treatment* variable. Explanatory variables that represent nominal (classification) data must be declared in the CLASS statement. For more information about specifying *effects*, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

variable-options

specify one or more of the following options within parentheses for after the *treatment* variable:

DESCENDING

DESC

reverses the order of the treatment categories. If both the DESCENDING and ORDER= options are specified, PROC CAUSALTRT orders the treatment categories according to the ORDER= option and then reverses that order.

EVENT='*category***' | FIRST | LAST**

specifies the event category for the binary treatment model. PROC CAUSALTRT models the probability of the event category. You can specify one of the following:

'*category***'**	specifies the value (formatted if a format is applied) of the event category in quotation marks.
FIRST	designates the first-ordered category as the event.
LAST	designates the last-ordered category as the event.

By default, EVENT=FIRST.

One of the most common sets of treatment levels is {0,1}, where 1 represents the event for which the probability is to be modeled. Consider the following example, where T takes the values 1 and 0 for event and nonevent, respectively, and X is the explanatory variable. To specify the value 1 as the event category, use the following PSMODEL statement:

```
psmodel T(event='1') = X;
```

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the treatment variable. The following table displays the available ORDER= options.

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.
FREQ	Descending frequency count. Levels that have the most observations come first in the order.
INTERNAL	Unformatted value. The sort order is machine-dependent.

By default, ORDER=FORMATTED.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | FIRST | LAST

REF='*category*' | FIRST | LAST

specifies the reference category for the binary treatment model. Specifying one treatment category as the reference is the same as specifying the other treatment category as the event category. You can specify one of the following:

'*category*' specifies the value (formatted if a format is applied) of the reference category in quotation marks.

FIRST designates the first-ordered category as the reference.

LAST designates the last-ordered category as the reference.

By default, REF=LAST.

psmodel-options

specify additional options for the propensity score model after a slash (/). These *psmodel-options* are summarized in Table 39.7 and described in detail after the table. These options are applied only if the estimation method you request requires fitting a propensity score model.

Table 39.7 *psmodel-options* in the PSMODEL Statement

Option	Description
ALPHA=	Specifies the level for confidence limits
PLOTS=	Produces graphics for the propensity scores and weights
WGTFLAG=	Specifies a value to be used to flag large weights

ALPHA=*number*

specifies a *number* to use as the α level for $100(1-\alpha)\%$ confidence limits that the PSMODEL statement computes. The value of *number* must be between 0 and 1. The default value of *number* is the set by the ALPHA= option in the PROC CAUSALTRT statement.

PLOTS < (*global-plot-options*) >=*plot-request*< (*options*) >

PLOTS < (*global-plot-options*) >=(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >)

produces ODS graphics by using the predicted propensity scores. No graphics are produced if you specify the PLOTS=NONE option in the PROC CAUSALTRT statement or if the estimation method used does not require fitting a propensity score model.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc causaltrt method=ipwr;
   model y;
   psmodel trt = x1 x2 /plots=all;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *plot-requests*:

ALL

produces all possible plots.

LOGITPSCORE

LPS

produces overlaid density plots for the logit of the propensity score within each treatment condition.

NONE

suppresses all plots that the PSMODEL statement could produce. This *plot-request* is used to suppress output if the PLOTS=ALL option is specified in the PROC CAUSALTRT statement.

OUTBYPSCORE

OUTBYPS

produces a scatter plot of the outcome variable by the propensity score for nonbinary outcomes. If the outcome is binary, box plots of the propensity scores within treatment conditions are produced for each outcome level. The whisker lengths for each box plot are determined by the maximum and minimum of the propensity scores.

OUTBYWEIGHT

OUTBYWGT

produces a scatter plot of the outcome variable by weight for nonbinary outcomes. If the outcome is binary, box plots of the weights within treatment conditions are produced for each outcome level. The whisker lengths for each box plot are determined by the maximum and minimum of the weights.

PSCLOUD

produces a point cloud of the propensity scores by jittering within the control and treatment conditions.

PSCOVDEN< *pscovden-options* >

produces density plots for the covariates or continuous effects that are specified in the PSMODEL statement. Each plot displays the density of effects for the treatment and control condition. Two plots are produced for each effect: one plot displays unweighted densities, and the other plot displays densities that are weighted by inverse probability weights. By default, plots are produced for all continuous effects that are specified in the PSMODEL statement, and the plots are collected in panels. You can specify the following *pscovden-options*:

EFFECTS(*effects***)**

specifies the effects for which density plots are produced. Plots are produced only for continuous effects that are specified in the PSMODEL statement. The *effects* consist of an explanatory variable or combination of variables. The syntax for specifying *effects* in this option is the same as the syntax for specifying *effects* in the MODEL and PSMODEL statements.

UNPACK

suppresses paneling. By default, multiple plots can appear in the same output panel. You can use this option to display each plot separately.

UNPACKEFFECTS

creates a separate panel for each effect. Each panel has a single row and two columns that contain the unweighted and weighted density plots. This option is ignored if you specify the UNPACK option in the PROC CAUSALTRT or PSMODEL statements.

PSDIST

produces a box plot of the propensity scores for each treatment condition. The whisker lengths for each box plot are determined by the maximum and minimum of the propensity scores.

WEIGHTCLOUD

WCLOUD

produces a point cloud of the weights by jittering within the control and treatment conditions.

WEIGHTDIST
WDIST
produces a box plot of the weights for each treatment condition. The whisker lengths for each box plot are determined by the maximum and minimum of the weights.

You can specify the following *global-plot-options*:

ONLY
suppresses plots that are not specified as a *plot-request*. This option applies only if the PLOTS= option is specified in the PROC CAUSALTRT statement.

UNPACK
UNPACKPANEL
suppresses paneling. By default, multiple plots can appear in the same output panel. You can specify UNPACK to display each plot separately.

WGTFLAG=*number*
specifies a *number* to use to flag large weights. The value of *number* must be greater than or equal to 1. If any observations have weights greater than *number*, a note is added to the SAS log. By default, WGTFLAG=50.

Details: CAUSALTRT Procedure

Causal Effects: Definitions, Assumptions, and Identification

The CAUSALTRT procedure estimates the causal effect of a binary treatment, T, on a continuous or discrete outcome, Y. This section defines causal effects and discusses the assumptions and conditions that enable researchers to establish causal interpretations.

Although the causal effects that PROC CAUSALTRT estimates are called treatment effects, the variable T might represent an intervention or exposure, depending on the problem nature. For example, consider that T is an intervention such as receiving subsidized access to a fitness center, and Y is measure of physical health. The researcher would like to assess the treatment or causal effect of the subsidized fitness program T on the physical health Y. But what exactly is the causal effect? This section uses Neyman-Rubin's potential outcomes framework (Rubin 1980, 1990) to define causal effects and to discuss their assumptions. Next, it explains the conditions that enable researchers to identify and establish causal effects.

The first important concept for defining a causal effect is potential outcomes. Potential outcomes describe an idealized source of data where you can observe a subject's response for all possible treatment assignments. For example, if you are estimating the causal effect of a binary variable for which $T = 0$ indicates assignment to the control and $T = 1$ assignment to treatment, then each subject has two potential outcomes $Y(0)$ and $Y(1)$. The subject-level (also called unit-level) treatment effect is defined as the difference in potential outcomes $Y(1) - Y(0)$.

To formally define the potential outcomes as unique and intrinsic values that are associated with a subject, Rubin (1980) states the stable unit treatment value assumption (SUTVA). See also Imbens and Rubin (2015). Two components of the SUTVA are the following:

- No interference. A subject's potential outcomes are not affected by the treatment assignments of the other subjects.
- No hidden variations of treatments. Subjects must receive the same form of treatment at each treatment level.

With a more refined statement, VanderWeele (2009) proposed a less restrictive version to the second component of SUTVA. Essentially, VanderWeele's version requires that the potential outcomes remain unchanged within certain degree of treatment variation for each treatment level. This version is also known as the treatment variation irrelevance assumption. See also Cole and Frangakis (2009).

Furthermore, Rubin (2005) points out an implicit but important assumption for constructing the potential outcome framework: the potential outcomes are not affected by the treatment assignment mechanism that the researchers use to learn about them. Whether in an observational study or a randomized experiment, potential outcomes remain unique to the subjects given the well-defined causal problem and SUTVA.

To summarize, the potential outcome framework with SUTVA ensures that the causal treatment effect is well defined. However, to actually attempt to estimate causal treatment effects from the data, the consistency assumption that relates an observed outcome to the potential outcome is needed:

$$Y = TY(1) + (1 - T)Y(0)$$

This equation states that the observed response Y is equal to the potential outcome with a treatment level that matches the actual assigned treatment level. In this setting, if j is an unobserved treatment condition, the unobserved potential outcome $Y(j)$ is defined using counterfactual conditionals as the outcome that would occur if, contrary to the observed assignment, the subject received treatment $T = j$. Because each subject can take part in only one treatment condition, at least half of the potential outcomes would not be observed in the data collection process. Therefore, the unit-level causal effect, as defined by $Y(1) - Y(0)$, is seldom the primary interest in research. Instead, average treatment effects in the population are the more common estimands.

The CAUSALTRT procedure estimates two types of treatment effects:

- The average treatment effect (ATE) for the entire population is given by

  $$\text{ATE} = \mu_1 - \mu_0 = E[Y(1)] - E[Y(0)]$$

  where $\mu_1 = E[Y(1)]$ and $\mu_0 = E[Y(0)]$ are potential outcome means (POMs) for the treatment and control conditions, respectively.

- The average treatment effect for the treated (ATT or ATET) is the average causal effect among only those individuals that receive treatment and is given by

  $$\text{ATT} = \mu_{1|T=1} - \mu_{0|T=1} = E[Y(1)|T = 1] - E[Y(0)|T = 1]$$

  where $\mu_{1|T=1} = E[Y(1)|T = 1]$ and $\mu_{0|T=1} = E[Y(0)|T = 1]$ are potential outcome means for the treatment and control conditions, respectively, conditional on receiving treatment.

It is important to notice that the preceding discussion of potential outcome framework, the associated assumptions, and the definitions of causal treatment effects apply to both experimental and observational

data (Rubin 1990). But for estimating causal treatment effects, statistical methodology for experimental and observational data would be differentiated.

The differentiation is caused by the following important condition (or assumption) for the identification of causal treatment effect: given the pretreatment covariates $\mathbf{X}$, the potential outcomes are independent of treatment assignment

This condition states that if all the covariates $\mathbf{X}$ have been included in the model appropriately (that is, there are no unmeasured covariates), the treatment effect of T on Y conditional on $\mathbf{X}$ would have a causal interpretation. Consequently, the causal treatment effects can be estimated as some functions of the observed conditional means.

Unfortunately, this condition cannot be 100% established in most applications; hence, this condition is usually referred to as an assumption in causal analysis literature. This assumption is called strong ignorability of treatment assignment or conditional exchangeability. This assumption sheds light on the different statistical methodology between randomized experiments and observational studies.

In experiments where randomization is used to assign subjects to treatment conditions, you can safely assume that the treatment assignment and potential outcomes are independent by design. Hence, for randomized experiments, traditional statistical methods such as t tests and ANOVA can be used without worrying about the confounding that can be caused by covariates $\mathbf{X}$.

However, in observational studies, subjects "select" the treatment conditions based on their pretreatment characteristics (covariates), $\mathbf{X}$, which could also be associated with the outcome variable. The treatment and outcome association would be confounded by the covariates. In this case, identification of causal effects requires that the strong ignorability or conditional exchangeability assumption be satisfied; that is, only when the estimation of the causal effects can successfully take into account all the confounding that is caused by the covariates $\mathbf{X}$.

The CAUSALTRT procedure implements several statistical techniques that enable you to estimate treatment effects when you specify confounding covariates $\mathbf{X}$. PROC CAUSALTRT uses the covariates $\mathbf{X}$ to fit generalized linear models for the treatment T, outcome Y, or both. Predicted values from these models are then incorporated into the estimation of the causal effects. In particular, you can estimate the ATE by the following methods:

- inverse probability weighting, where weights are predicted from the model for the treatment assignment T (also called the propensity score model)
- regression adjustment, where the model of the outcome Y is used to predict potential outcomes for each treatment assignment
- doubly robust estimation, which combines both inverse probability weighting and regression adjustment methods to estimate the causal effect; doubly robust estimation methods provide unbiased estimates even if one of the models is misspecified

Whenever your modeling involves a model for the treatment assignment T or propensity score (that is, when you use either the inverse probability weighting or doubly robust estimation), an additional assumption about positivity is needed: each subject should have a nonzero probability of receiving any treatment condition. This assumption is expressed as follows:

$$0 < \Pr(T = 1 \mid \mathbf{X} = \mathbf{x}) < 1$$

For more information about estimation of the ATE, see the section "Estimating the Average Treatment Effect (ATE)" on page 2536. For more information about the models that are required for the estimation methods that PROC CAUSALTRT implements and how default estimation methods are determined, see the section "Outline of Estimation Method Requirements" on page 2505. You can estimate the ATT by using either inverse probability weights or regression adjustment. For more information about estimation of the ATT, see the section "Estimating the Average Treatment Effect for the Treated (ATT)" on page 2541.

Estimating the Average Treatment Effect (ATE)

The CAUSALTRT procedure can estimate the average treatment effect (ATE) by using inverse probability weighting, regression adjustment, and doubly robust methods (doubly robust methods are combinations of the first two methods). The following sections describe each of these estimation methods.

Inverse Probability Weighting

The CAUSALTRT procedure estimates the potential outcome means and ATE by using the three inverse probability weighting methods that are described in Lunceford and Davidian (2004). All three methods compute weights using estimates for the conditional probability of receiving treatment,

$$e(\mathbf{x}) = \Pr(T = 1 \mid \mathbf{X} = \mathbf{x})$$

where $\mathbf{x}$ is a vector of the observed covariates. The conditional probability $e(\mathbf{x})$ is called the propensity score, and the model that is used to estimate $e(\mathbf{x})$ is called the propensity score model. The inverse probability weight for an observation is equal to

$$\frac{1}{\Pr\left(T = t \mid \mathbf{x}\right)} = \frac{t}{e(\mathbf{x})} + \frac{1-t}{1-e(\mathbf{x})}$$

In addition to the stable unit treatment value assumption (SUTVA), two more assumptions on the propensity scores are required for the use of inverse probability weighting methods. The positivity assumption, that $0 < e(\mathbf{x}) < 1$, ensures that each observation has a nonzero probability of receiving each treatment condition. The assumption of no unmeasured confounders, also known as strong ignorability, requires that the potential outcomes $Y(0)$ and $Y(1)$ be independent of the treatment variable T conditional on the covariates $\mathbf{x}$. In practice, this assumption means that the set of covariates $\mathbf{x}$ should have included all the important confounders in the propensity score model. If these conditions are satisfied and the propensity score model is correctly specified, then it follows that

$$E\left[\frac{TY}{e(\mathbf{x})}\right] - E\left[\frac{(1-T)Y}{1-e(\mathbf{x})}\right] = E\left[\frac{Y(1)}{e(\mathbf{x})} E\left[T \mid \mathbf{x}\right]\right] - E\left[\frac{Y(0)}{1-e(\mathbf{x})} E\left[(1-T) \mid \mathbf{x}\right]\right] = E\left[Y(1)\right] - E\left[Y(0)\right]$$

The CAUSALTRT procedure uses the effects that are specified in the PSMODEL statement to fit a logistic regression to the propensity score model. The inverse probability weights are then computed by taking the inverse of the estimated propensity scores.

Suppose you have observations $(y_i, t_i, \mathbf{x}_i)$, for $i = 1, \ldots, n$. The parameter estimates for the propensity score model is given by $\hat{\boldsymbol{\beta}}_{\mathrm{ps}}$, and the predicted values for the propensity score are given by

$$\hat{e}_i = \hat{e}(\mathbf{x}_i) = \frac{\exp\left(\mathbf{x}_i' \hat{\boldsymbol{\beta}}_{\mathrm{ps}}\right)}{1 + \exp\left(\mathbf{x}_i' \hat{\boldsymbol{\beta}}_{\mathrm{ps}}\right)}$$

The three weighting methods that PROC CAUSALTRT implements obtain unbiased estimates of the potential outcome means by solving the equations

$$\mathbf{S}_{\text{ipw}}(\boldsymbol{\mu}) = \sum_{i=1}^{n} \mathbf{S}_{\text{ipw},i} = \mathbf{0}$$

for $\boldsymbol{\mu} = (\mu_0, \mu_1)$, where

$$\mathbf{S}_{\text{ipw},i} = \begin{bmatrix} \frac{(1-t_i)(y_i-\mu_0)}{1-\hat{e}_i} - \eta_0 \left(\frac{t_i-\hat{e}_i}{1-\hat{e}_i}\right) \\ \frac{t_i(y_i-\mu_1)}{\hat{e}_i} + \eta_1 \left(\frac{t_i-\hat{e}_i}{\hat{e}_i}\right) \end{bmatrix}$$

These estimation methods differ in the values of (η_0, η_1). The choices of (η_0, η_1) and the corresponding potential outcome mean estimates for each method are as follows:

- IPW:

$$(\eta_0, \eta_1) = (\mu_0, \mu_1)$$

$$\hat{\mu}_0^{\text{ipw}} = n^{-1} \sum_{i=1}^{n} \frac{(1-t_i)y_i}{1-\hat{e}_i}$$

$$\hat{\mu}_1^{\text{ipw}} = n^{-1} \sum_{i=1}^{n} \frac{t_i y_i}{\hat{e}_i}$$

- IPWR:

$$(\eta_0, \eta_1) = (0, 0)$$

$$\hat{\mu}_0^{\text{ipwr}} = \left[\sum_{i=1}^{n} \frac{(1-t_i)}{1-\hat{e}_i}\right]^{-1} \sum_{i=1}^{n} \frac{(1-t_i)y_i}{1-\hat{e}_i}$$

$$\hat{\mu}_1^{\text{ipwr}} = \left[\sum_{i=1}^{n} \frac{t_i}{\hat{e}_i}\right]^{-1} \sum_{i=1}^{n} \frac{t_i y_i}{\hat{e}_i}$$

- IPWS:

$$(\eta_0, \eta_1) = \left(-\frac{E\left[(1-T)(Y-\mu_0)/(1-e(\mathbf{x}))^2\right]}{E\left[(T-e(\mathbf{x}))^2/(1-e(\mathbf{x}))^2\right]}, -\frac{E\left[T(Y-\mu_1)/e(\mathbf{x})^2\right]}{E\left[(T-e(\mathbf{x}))^2/e(\mathbf{x})^2\right]}\right)$$

$$\hat{\mu}_0^{\text{ipws}} = -\left[\sum_{i=1}^{n} \frac{(1-t_i)}{1-\hat{e}_i}\left(1-\frac{C_0}{1-\hat{e}_i}\right)\right]^{-1} \sum_{i=1}^{n} \frac{(1-t_i)y_i}{1-\hat{e}_i}\left(1-\frac{C_0}{1-\hat{e}_i}\right)$$

$$\hat{\mu}_1^{\text{ipws}} = \left[\sum_{i=1}^{n} \frac{t_i}{\hat{e}_i}\left(1-\frac{C_1}{\hat{e}_i}\right)\right]^{-1} \sum_{i=1}^{n} \frac{t_i y_i}{\hat{e}_i}\left(1-\frac{C_1}{\hat{e}_i}\right)$$

where

$$C_0 = \left[-\sum_{j=1}^{n} \frac{t_j - \hat{e}_j}{1 - \hat{e}_j} \right] \left[\sum_{j=1}^{n} \left(\frac{t_j - \hat{e}_j}{1 - \hat{e}_j} \right)^2 \right]^{-1}$$

$$C_1 = \left[\sum_{j=1}^{n} \frac{t_j - \hat{e}_j}{\hat{e}_j} \right] \left[\sum_{j=1}^{n} \left(\frac{t_j - \hat{e}_j}{\hat{e}_j} \right)^2 \right]^{-1}$$

If the values of (η_0, η_1) that motivate the IPWS method were known constants, they would produce estimates for the potential outcome means that have the smallest asymptotic variance among the class of estimators that is defined by $\mathbf{S}_{\mathrm{ipw}}(\boldsymbol{\mu})$. The IPWS method estimates $(\hat{\eta}_0, \hat{\eta}_1)$ by solving the estimating equations

$$\mathbf{S}(\boldsymbol{\eta}) = \sum_{i=1}^{n} \mathbf{S}_{\boldsymbol{\eta},i} = \mathbf{0}$$

for $\boldsymbol{\eta} = (\eta_0, \eta_1)$, where:

$$\mathbf{S}_{\boldsymbol{\eta},i} = \begin{bmatrix} \frac{(1-t_i)(y_i-\mu_0)}{(1-\hat{e}_i)^2} + \eta_0 \left(\frac{t_i-\hat{e}_i}{1-\hat{e}_i} \right)^2 \\ \frac{t_i(y_i-\mu_1)}{\hat{e}_i^2} + \eta_1 \left(\frac{t_i-\hat{e}_i}{\hat{e}_i} \right)^2 \end{bmatrix}$$

Although you do not need to specify an outcome model when you use the inverse probability weighting methods, you still need to specify the outcome variable in the MODEL statement. The IPW, IPWR, or IPWS estimation method can be requested using the METHOD= option in the PROC CAUSALTRT statement. For more information about which model specifications are required for different estimation methods and how default estimation methods are determined, see the section "Outline of Estimation Method Requirements" on page 2505.

The robust sandwich covariance estimate for $\hat{\boldsymbol{\mu}}$ is computed by stacking the equations $\mathbf{S}_{\mathrm{ipw}}(\boldsymbol{\mu})$ and the score function for the logistic regression model that is used to obtain $\hat{\boldsymbol{\beta}}_{\mathrm{ps}}$. By stacking the equations, PROC CAUSALTRT adjusts the covariance matrix for $\hat{\boldsymbol{\mu}}$ to account for the prediction of $\hat{\boldsymbol{\beta}}_{\mathrm{ps}}$ and the propensity scores $\hat{e}_i$. For more information about using the robust sandwich covariance estimate to compute standard errors, see the section "Asymptotic Formulas" on page 2543.

The finite sample properties of $\hat{\boldsymbol{\mu}}$ and its covariance matrix can be affected by observations that have large weights. By default, a note is added to the SAS log if any observations have weights greater than 50. You can specify the value that is used to flag large weights in the WGTFLAG= option in the PSMODEL statement. You can also inspect the values of the predicted weights by requesting graphics in the PLOTS= option in the PROC CAUSALTRT or PSMODEL statements.

As described in Rosenbaum and Rubin (1983), the propensity score is also a balancing score. To assess the balance that is produced by a propensity score model, you can request the display of weighted and unweighted standardized mean differences and variance ratios for propensity score model effects in the COVDIFFPS option in the PROC CAUSALTRT statement. You can also request weighted and unweighted densities for propensity score model effects by specifying PSCOVDEN in the PLOTS= option in the PROC CAUSALTRT or PSMODEL statements. To inspect the balance of a propensity score model without showing the causal effects, specify the NOEFFECT option in the PROC CAUSALTRT statement.

Based on the balancing score property, you can use the propensity scores to create matched samples or strata for the data. For more information about applying the propensity score in matching and stratification see Chapter 101, "The PSMATCH Procedure."

Regression Adjustment

Regression adjustment estimates the ATE by using predicted values for the potential outcomes. The CAUSALTRT procedure predicts the potential outcomes from generalized linear models that are fit to the data by maximum likelihood estimation. For more information about generalized linear models, see the section "Generalized Linear Models Theory" on page 3655 in Chapter 51, "The GENMOD Procedure."

The effects that you specify in the MODEL statement are used to fit models for the outcome within each treatment condition. The predicted potential outcomes are then given by

$$\hat{y}_{i0} = g^{-1}(\mathbf{x}_i'\hat{\boldsymbol{\beta}}_c)$$

$$\hat{y}_{i1} = g^{-1}(\mathbf{x}_i'\hat{\boldsymbol{\beta}}_t)$$

where g is the link function being used and $\hat{\boldsymbol{\beta}}_{\mathrm{c}}$ and $\hat{\boldsymbol{\beta}}_{\mathrm{t}}$ are the parameter estimates for the control and treatment outcome models, respectively. The potential outcome mean μ_j is then estimated by averaging the predicted $\hat{y}_{ij}$ for all observations $i = 1, \ldots, n$, regardless of the observed treatment assignment. Therefore, the regression adjustment estimates for the potential outcome means solve the estimating equations

$$\mathbf{S}_{\mathrm{reg}}(\boldsymbol{\mu}) = \sum_{i=1}^{n} \mathbf{S}_{\mathrm{reg},i} = \mathbf{0}$$

where

$$\mathbf{S}_{\mathrm{reg},i} = \begin{bmatrix} \hat{y}_{i0} - \mu_0 \\ \hat{y}_{i1} - \mu_1 \end{bmatrix}$$

The regression adjustment estimates for the potential outcome means are equal to

$$\hat{\mu}_0^{\mathrm{reg}} = n^{-1} \sum_{i=1}^{n} \hat{y}_{i0}$$

$$\hat{\mu}_1^{\mathrm{reg}} = n^{-1} \sum_{i=1}^{n} \hat{y}_{i1}$$

If the outcome model is correctly specified and the assumption of no unmeasured confounders is satisfied, then regression adjustment produces unbiased estimates for the potential outcome means. The outcome model effects should also have similar distributions between treatment conditions to avoid extrapolation when the potential outcomes $\hat{y}_{ij}$ are predicted.

Although you do not need to specify a propensity score model in order to estimate the potential outcome means by the regression adjustment method, you still need to specify the treatment variable in the PSMODEL statement. You can specify METHOD=REGADJ in the PROC CAUSALTRT statement to request the regression adjustment method. For more information about which model specifications are required for different estimation methods and how default estimation methods are determined, see the section "Outline of Estimation Method Requirements" on page 2505.

The robust sandwich covariance estimate for $\hat{\boldsymbol{\mu}}^{\text{reg}}$ is estimated by stacking the equations $\mathbf{S}_{\text{reg}}(\boldsymbol{\mu})$ and the score function for the outcome models that are used to estimate $\hat{\boldsymbol{\beta}}_{\text{c}}$ and $\hat{\boldsymbol{\beta}}_{\text{t}}$. By stacking the equations, PROC CAUSALTRT adjusts the covariance matrix for $\hat{\boldsymbol{\mu}}^{\text{reg}}$ to account for the prediction of the potential outcomes $\hat{y}_{ij}$. For more information about using the robust sandwich covariance estimate to compute standard errors, see the section "Asymptotic Formulas" on page 2543.

Doubly Robust Estimation

Doubly robust estimation methods fit models for both the outcome and treatment variables. They combine inverse probability weighting and regression adjustment to estimate the potential outcome means. The methods are said to be doubly robust because they provide unbiased estimates for $\boldsymbol{\mu}$ even if one of the models is misspecified (Bang and Robins 2005). In this sense, you have two opportunities to obtain unbiased estimation of causal effects by specifying either a correct treatment or a correct outcome model. The CAUSALTRT procedure implements two doubly robust estimation methods: the augmented inverse probability weighting (AIPW) method described in Lunceford and Davidian (2004) and the inverse probability weighted regression adjustment (IPWREG) method described in Wooldridge (2010).

The AIPW estimation method estimates propensity scores $\hat{e}_i$ and the associated parameter estimates $\hat{\boldsymbol{\beta}}_{ps}$ by fitting a logistic regression model for treatment assignment, as described in section "Inverse Probability Weighting" on page 2536. It also estimates the potential outcomes $\hat{y}_{ij}$ by using the maximum likelihood fitting of the generalized linear models, as described in section "Regression Adjustment" on page 2539. Note that the treatment assignment and outcome models do not need to be fit using the same set of model effects.

The AIPW estimates for potential outcome means solve the estimating equations

$$\mathbf{S}_{\text{aipw}}(\boldsymbol{\mu}) = \sum_{i=1}^{n} \mathbf{S}_{\text{aipw},i}(\boldsymbol{\mu}) = \mathbf{0}$$

for (μ_0, μ_1), where

$$\mathbf{S}_{\text{aipw},i}(\boldsymbol{\mu}) = \begin{bmatrix} \frac{(1-t_i)y_i}{1-\hat{e}_i} + \hat{y}_{i0}\left(\frac{t_i-\hat{e}_i}{1-\hat{e}_i}\right) - \mu_0 \\ \frac{t_i y_i}{\hat{e}_i} - \hat{y}_{i1}\left(\frac{t_i-\hat{e}_i}{\hat{e}_i}\right) - \mu_1 \end{bmatrix} = \begin{bmatrix} \hat{y}_{i0} - \left(\frac{1-t_i}{1-\hat{e}_i}\right)(\hat{y}_{i0} - y_i) - \mu_0 \\ \hat{y}_{i1} - \left(\frac{t_i}{\hat{e}_i}\right)(\hat{y}_{i1} - y_i) - \mu_1 \end{bmatrix}$$

The AIPW estimates for the potential outcome means are given by

$$\hat{\mu}_0^{\text{aipw}} = n^{-1} \sum_{i=1}^{n} \frac{(1-t_i)y_i}{1-\hat{e}_i} + \hat{y}_{i0}\left(\frac{t_i-\hat{e}_i}{1-\hat{e}_i}\right)$$

$$\hat{\mu}_1^{\text{aipw}} = n^{-1} \sum_{i=1}^{n} \frac{t_i y_i}{\hat{e}_i} - \hat{y}_{i1}\left(\frac{t_i-\hat{e}_i}{\hat{e}_i}\right)$$

If one of the models is misspecified, $\hat{\boldsymbol{\mu}}^{\text{aipw}}$ remains an unbiased estimate of $\boldsymbol{\mu}$. However, the efficiency of $\hat{\boldsymbol{\mu}}^{\text{aipw}}$ and the estimation of its covariance matrix are affected.

The IPWREG estimation method uses the same logistic model for predicting $\hat{e}_i$, but it fits weighted models for the outcome variable separately for each treatment condition. To fit the outcome models, each observation is weighted by the inverse probability weights as follows:

$$\frac{1}{\Pr(T = t_i \mid \mathbf{x}_i)} = \frac{t_i}{e(\mathbf{x}_i)} + \frac{1-t_i}{1-e(\mathbf{x}_i)}$$

Let $\hat{\boldsymbol{\beta}}_{\mathrm{c}}^{w}$ and $\hat{\boldsymbol{\beta}}_{\mathrm{t}}^{w}$ denote the parameter estimates from the weighted regression outcome models for the control and treatment conditions, respectively. The predicted potential outcomes are then given by

$$\hat{y}_{i0}^{w} = g^{-1}(\mathbf{x}_i'\hat{\boldsymbol{\beta}}_{\mathrm{c}}^{w})$$

$$\hat{y}_{i1}^{w} = g^{-1}(\mathbf{x}_i'\hat{\boldsymbol{\beta}}_{\mathrm{t}}^{w})$$

where g is the link function being used. For the IPWREG method, PROC CAUSALTRT enforces the use of the canonical link for the distribution of the outcome variable. You cannot override the canonical link by the LINK= option. For information about the canonical link for each distribution, see Table 39.5.

The IPWREG estimates for the potential outcome means are given by

$$\hat{\mu}_0^{\mathrm{wreg}} = n^{-1}\sum_{i=1}^{n}\hat{y}_{i0}^{w}$$

$$\hat{\mu}_1^{\mathrm{wreg}} = n^{-1}\sum_{i=1}^{n}\hat{y}_{i1}^{w}$$

These estimates solve the following estimating equations:

$$\mathbf{S}_{\mathrm{reg}}^{w}(\boldsymbol{\mu}) = \sum_{i=1}^{n}\mathbf{S}_{\mathrm{reg},i}^{w} = \sum_{i=1}^{n}\begin{bmatrix} \hat{y}_{i0}^{w} - \mu_0 \\ \hat{y}_{i1}^{w} - \mu_1 \end{bmatrix} = \mathbf{0}$$

Estimating the Average Treatment Effect for the Treated (ATT)

In some research settings or program evaluation studies, instead of the average treatment effect (ATE), researchers or policy makers might be more interested in the causal treatment effects only for those who choose to participate in the treatment condition. Hence, the average treatment effect for the treated (ATT or ATET) becomes the focus.

The CAUSALTRT procedure enables you to estimate the ATT and the corresponding conditional potential outcome means by using either of the following methods:

- Inverse probability weighting with ratio adjustment (IPWR). To estimate the ATT, the inverse probability weights that are described in the section "Inverse Probability Weighting" on page 2536 are multiplied by the predicted propensity scores. The ATT weights are therefore given by

  $$\frac{e(\mathbf{x}_i)}{\Pr(T = t_i \mid \mathbf{x}_i)} = t_i + \frac{(1-t_i)e(\mathbf{x}_i)}{1-e(\mathbf{x}_i)}$$

  The estimating equations that are solved by the IPWR estimates for the conditional potential outcome means $\boldsymbol{\mu}_{T=1}$ are

  $$\mathbf{S}_{\mathrm{ipw}}(\boldsymbol{\mu}_{T=1}) = \sum_{i=1}^{n}\mathbf{S}_{\mathrm{ipw},i|T=1} = \mathbf{0}$$

  where

  $$\mathbf{S}_{\mathrm{ipw},i|T=1} = \begin{bmatrix} (1-t_i)(y_i - \mu_{0|T=1})\frac{\hat{e}_i}{1-\hat{e}_i} \\ t_i(y_i - \mu_{1|T=1}) \end{bmatrix}$$

The IPWR estimates for the conditional potential outcome means $\boldsymbol{\mu}_{T=1}$ are given by

$$\hat{\mu}^{\text{ipwr}}_{0|T=1} = \left[\sum_{i=1}^{n}(1-t_i)\frac{\hat{e}_i}{1-\hat{e}_i}\right]^{-1}\sum_{i=1}^{n}(1-t_i)y_i\frac{\hat{e}_i}{1-\hat{e}_i}$$

$$\hat{\mu}^{\text{ipwr}}_{1|T=1} = \left[\sum_{i=1}^{n}t_i\right]^{-1}\sum_{i=1}^{n}t_i\,y_i$$

If the propensity score model is correctly specified and the stable unit treatment value assumption (SUTVA), positivity, and no unmeasured confounders assumptions are satisfied, then the predicted conditional means are unbiased estimates for $\mu_{j|T=1}$.

- Regression adjustment (REGADJ). For this method, PROC CAUSALTRT obtains predicted potential outcomes $\hat{y}_{ij}$ from the outcome models that are fitted separately for each treatment condition. This approach is described in section "Regression Adjustment" on page 2539. To estimate $\boldsymbol{\mu}_{T=1}$, the predicted values are averaged only for individuals who received treatment. The REGADJ estimates for the conditional potential outcome means $\boldsymbol{\mu}_{T=1}$ are given by

$$\hat{\mu}^{\text{reg}}_{0|T=1} = \left[\sum_{i=1}^{n}t_i\right]^{-1}\sum_{i=1}^{n}t_i\,\hat{y}_{i0}$$

$$\hat{\mu}^{\text{reg}}_{1|T=1} = \left[\sum_{i=1}^{n}t_i\right]^{-1}\sum_{i=1}^{n}t_i\,\hat{y}_{i1}$$

The estimates therefore solve the estimating equations

$$\mathbf{S}_{\text{reg}}(\boldsymbol{\mu}_{T=1}) = \sum_{i=1}^{n}\mathbf{S}_{\text{reg},i|T=1} = \mathbf{0}$$

where

$$\mathbf{S}_{\text{reg},i|T=1} = \begin{bmatrix} t_i\left(\hat{y}_{i0}-\mu_{0|T=1}\right) \\ t_i\left(\hat{y}_{i1}-\mu_{1|T=1}\right) \end{bmatrix}$$

To request the estimation of the conditional potential outcome means and ATT, specify the ATT option in the PROC CAUSALTRT statement.

Comparing Weights for Estimating the ATE and ATT

This section describes and distinguishes the weights that are used in different estimation methods and analysis situations.

When the average treatment effect (ATE) is estimated, the inverse probability weight for an observation is equal to

$$\frac{1}{\Pr(T = t_i \mid \mathbf{x}_i)} = \frac{t_i}{e(\mathbf{x}_i)} + \frac{1 - t_i}{1 - e(\mathbf{x}_i)}$$

These are the weights that are used by the IPW, IPWR, IPWS, AIPW, and IPWREG methods to estimate the potential outcome means and ATE.

However, when you estimate the average treatment effect for the treated (ATT), the IPWR method uses the weights

$$\frac{\Pr(T = 1 \mid \mathbf{x}_i)}{\Pr(T = t_i \mid \mathbf{x}_i)} = t_i + \frac{(1 - t_i)e(\mathbf{x}_i)}{1 - e(\mathbf{x}_i)}$$

The differential uses of weights for ATE and ATT also apply to the following analyses or plots:

- weights that are flagged by the WGTFLAG= value
- the OUTBYWEIGHT, WDIST, and WCLOUD plots

Regardless of whether the ATT or ATE is estimated, the following analyses or plots always use the inverse probability weights (the first formula in this section):

- the weighted standardized mean differences and variance ratios for the covariates (effects) in the propensity score model, which are requested by the COVDIFFPS option
- the weighted density plots for the continuous covariates (effects) in the propensity score model, which are requested by the PLOTS=PSCOVDEN option

Standard Errors and Confidence Intervals

Asymptotic Formulas

PROC CAUSALTRT computes standard errors for the potential outcome means and treatment effect by using the robust sandwich covariance formula that is based on asymptotic theory. In general, let $\boldsymbol{\theta}_0$ denote the vector of parameters of interest. PROC CAUSALTRT considers all its estimation as an M-estimation problem that solves the following estimating equations:

$$\mathbf{S}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \mathbf{S}_i = \mathbf{0}$$

When the function $S(\cdot)$ is evaluated at θ_0, it satisfies $E[S] = 0$. By the theory of M-estimation (Stefanski and Boos 2002), the robust sandwich covariance matrix, $\hat{V}(\hat{\boldsymbol{\theta}})$, for the estimates $\hat{\boldsymbol{\theta}}$ is given by

$$n^{-1}\mathbf{A}_n(\hat{\boldsymbol{\theta}})^{-1}\mathbf{B}_n(\hat{\boldsymbol{\theta}})\mathbf{A}_n(\hat{\boldsymbol{\theta}})^{-T}$$

where n is the sample size, $\mathbf{A}^{-T}$ denotes $\left(\mathbf{A}^T\right)^{-1}$, and

$$\mathbf{A}_n(\hat{\boldsymbol{\theta}}) = n^{-1}\sum_{i=1}^{n}\frac{-\partial S_i}{\partial \boldsymbol{\theta}^T}$$

$$\mathbf{B}_n(\hat{\boldsymbol{\theta}}) = n^{-1}\sum_{i=1}^{n} S_i S_i^T$$

PROC CAUSALTRT computes standard errors by taking the square roots of the diagonal elements of $\hat{V}(\hat{\boldsymbol{\theta}})$.

The form and dimension of $S(\boldsymbol{\theta})$, and hence those of the robust sandwich covariance matrix, depend on which estimation method is used. Whenever an outcome or treatment (propensity score) model is used in a particular estimation method, the estimating equations must also include the corresponding score vectors for that model.

Let $\boldsymbol{\beta}$ be a generic vector of parameters of a model, and let $\boldsymbol{\mu} = (\mu_0, \mu_1)$ be the vector of potential outcome means. Further, let $\mathbf{S}_{\mathrm{ps}}$, $\mathbf{S}_{\mathrm{out}}$, and $\mathbf{S}^w_{\mathrm{out}}$ denote the score functions from the maximum likelihood estimation of the propensity score, outcome, and inverse probability weighted outcome models, respectively. Then the composition of $\boldsymbol{\theta}_0$ and $\mathbf{S}_i$ for each estimation method is as follows:

- AIPW

$$\boldsymbol{\theta}_0 = \left(\boldsymbol{\beta}_{\mathrm{ps}}, \boldsymbol{\beta}_{\mathrm{c}}, \boldsymbol{\beta}_{\mathrm{t}}, \boldsymbol{\mu}\right)$$

$$\mathbf{S}_i = \left(\mathbf{S}_{\mathrm{ps},i}, \mathbf{S}_{\mathrm{out},i}, \mathbf{S}_{\mathrm{ipw},i}\right)$$

- IPW and IPWR

$$\boldsymbol{\theta}_0 = \left(\boldsymbol{\beta}_{\mathrm{ps}}, \boldsymbol{\mu}\right)$$

$$\mathbf{S}_i = \left(\mathbf{S}_{\mathrm{ps},i}, \mathbf{S}_{\mathrm{ipw},i}\right)$$

- IPWS

$$\boldsymbol{\theta}_0 = \left(\boldsymbol{\beta}_{\mathrm{ps}}, \boldsymbol{\eta}, \boldsymbol{\mu}\right)$$

$$\mathbf{S}_i = \left(\mathbf{S}_{\mathrm{ps},i}, \mathbf{S}_{\boldsymbol{\eta},i}, \mathbf{S}_{\mathrm{ipw},i}\right)$$

- IPWREG

$$\boldsymbol{\theta}_0 = \left(\boldsymbol{\beta}_{\mathrm{ps}}, \boldsymbol{\beta}^w_{\mathrm{c}}, \boldsymbol{\beta}^w_{\mathrm{t}}, \boldsymbol{\mu}\right)$$

$$\mathbf{S}_i = \left(\mathbf{S}_{\mathrm{ps},i}, \mathbf{S}^w_{\mathrm{out},i}, \mathbf{S}^w_{\mathrm{reg},i}\right)$$

- REGADJ

$$\boldsymbol{\theta}_0 = (\boldsymbol{\beta}_{\mathrm{c}}, \boldsymbol{\beta}_{\mathrm{t}}, \boldsymbol{\mu})$$

$$\mathbf{S}_i = \left(\mathbf{S}_{\mathrm{out},i}, \mathbf{S}_{\mathrm{reg},i}\right)$$

The $(1-\alpha)100\%$ Wald confidence interval for the potential outcome mean μ_i is given by

$$\hat{\mu}_i \pm \mathrm{z}_{1-\alpha/2}\hat{\sigma}_i$$

where z_p is the $100p$th percentile of the standard normal distribution and $\hat{\sigma}_i$ is the standard error estimate for the potential outcome estimate $\hat{\mu}_i$. Because the ATE is the difference in potential outcome means, its standard error is computed by applying standard variance rules to the estimates from the covariance matrix for the potential outcomes. The corresponding Wald confidence interval for ATE is then constructed by the standard formula as shown in the previous expression. For the estimates of conditional potential outcome means, the standard errors and confidence limits are computed using the same approach but with $\mathbf{S}_{\mathrm{ipw},i}$ and $\mathbf{S}_{\mathrm{reg},i}$ replaced by $\mathbf{S}_{\mathrm{ipw},i|T=1}$ and $\mathbf{S}_{\mathrm{reg},i|T=1}$.

To provide better theoretical and computational properties, PROC CAUSALTRT uses modified expressions for components of A_n and B_n when the covariance matrix $\hat{V}(\hat{\boldsymbol{\theta}})$ is computed. For descriptions of the modifications, their motivation, and theoretical justification, see Pierce (1982); Robins, Rotnitzky, and Zhao (1995); Lunceford and Davidian (2004); Wooldridge (2010).

Bootstrap Methods

If you specify the BOOTSTRAP statement, PROC CAUSALTRT uses bootstrap resampling to compute standard errors and confidence intervals for the potential outcome means and treatment effect. The procedure samples as many bootstrap sample data sets (replicates) as you specify in the NBOOT= option and then estimates the potential outcome means and treatment effect for each replication. The bootstrap samples are taken from within the treatment conditions. Each bootstrap replicate contains the same numbers of usable observations in the control and treatment conditions as the number of usable observations that are included in the input data set.

Bootstrap confidence intervals are computed only for the potential outcome means and treatment effect. You can specify one or more the following types of bootstrap confidence intervals in the BOOTCI option in the BOOTSTRAP statement:

- The BOOTCI(NORMAL) option uses the normal approximation method to compute the bootstrap confidence interval. That is, the $(1-\alpha)100\%$ normal bootstrap confidence interval is given by

  $$\hat{\mu}_j \pm \sigma_{\mu_j^*} \times \mathrm{z}_{(1-\alpha/2)}$$

  where $\hat{\mu}_j$ is the estimate of μ_j from the original sample, $\sigma_{\mu_j^*}$ is the standard deviation of the bootstrap parameter estimates, and $\mathrm{z}_{(1-\alpha/2)}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. PROC CAUSALTRT requires at least 50 bootstrap samples for normal bootstrap confidence intervals and does not compute them if 40 or fewer of the samples produce usable estimates.

- The BOOTCI(PERC) option requests the percentile method, which uses the $100(\alpha/2)$th and $100(1-\alpha/2)$th percentiles of the bootstrap parameter estimates as the confidence limits. These percentiles are

computed as follows. Let $\mu^*_{j,1}, \mu^*_{j,2}, \ldots, \mu^*_{j,B}$ represent the ordered values of the bootstrap estimates for the potential outcome mean μ_j. Let the kth weighted average percentile be q, set $p = \frac{k}{100}$, and let

$$np = l + g$$

where l is the integer part of np and g is the fractional part of np. Then the kth percentile, q, is computed as follows, which corresponds to the default percentile definition of the UNIVARIATE procedure:

$$q = \begin{cases} \frac{1}{2}\left(\mu^*_{j,l} + \mu^*_{j,l+1}\right) & \text{if} \quad g = 0 \\ \mu^*_{j,l+1} & \text{if} \quad g > 0 \end{cases}$$

- The BOOTCI(BC) option requests bias-corrected bootstrap confidence intervals, which use the cumulative distribution function (CDF), $G(\mu^*)$, of the bootstrap parameter estimates to determine the upper and lower endpoints of the confidence interval. The bias-corrected bootstrap confidence interval is given by

  $$G^{-1}\left(\Phi(2z_0 \pm z_{\alpha/2})\right)$$

  where Φ is the standard normal CDF, $z_{\alpha/2} = \Phi^{-1}(\alpha/2)$, and z_0 is a bias correction,

  $$z_0 = \Phi^{-1}\left(\frac{\mathrm{N}\left(\mu^*_j \le \hat{\mu}_j\right)}{\mathrm{B}}\right)$$

  where $\hat{\mu}_j$ is the original sample estimate of μ_j from the input data set; $\mathrm{N}(\mu^*_j \le \hat{\mu}_j)$ is the number of bootstrap estimates, μ^*_j, that are less than or equal to $\hat{\mu}_j$; and B is the number of bootstrap replicates for which an estimate for the treatment effect is obtained.

The bias-corrected bootstrap confidence intervals are the default intervals used if you do not override them by specifying the NORMAL or PERC suboptions in the BOOTCI option.

PROC CAUSALTRT requires at least 1,000 bootstrap samples for the percentile and bias-corrected bootstrap confidence intervals and does not compute them if fewer than 900 of the samples produce usable estimates. If the number of samples *n* specified in the NBOOT=*n* option is less than 1,000 and the percentile or bias-corrected bootstrap confidence intervals are requested, the value of *n* is ignored. Bootstrap confidence intervals and the bootstrap standard deviations are computed in the same manner for the estimates of the ATE and ATT.

Missing Values

When the CAUSALTRT procedure fits a model, it excludes observations that have missing values for the outcome variable, treatment variable, frequency variable, or predictor variables, which are specified in the MODEL and PSMODEL statements. It also excludes observations that have invalid response or frequency values. Invalid response values are determined by the probability distribution for the response as specified in the DIST= option in the MODEL statement. The exclusion of observations because of invalid responses applies to all estimation methods, even for methods that do not require an outcome model specification (such as IPW, IPWR, and IPWS). In particular, responses must be strictly positive for the inverse Gaussian and gamma distributions, and must be nonnegative for the Poisson distribution. In all cases, exclusion of observations almost certainly affects the estimation results.

ODS Table Names

PROC CAUSALTRT assigns a name to each table that it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 39.8. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 39.8 ODS Tables Produced by PROC CAUSALTRT

ODS Table Name	Description	Statement	Option
CausalEffects	Estimates for potential outcome means and treatment effect		Default
ClassLevels	Classification variable levels	CLASS	Default
ControlEstimates	Parameter estimates for the control condition outcome model	PROC CAUSALTRT	POUTCOMEMOD
ModelInfo	Model information		Default
NObs	Number of observations		Default
PSCovDiff	Standardized mean differences and variance ratios of propensity score model effects	PROC CAUSALTRT	COVDIFFPS
PSModelEstimates	Parameter estimates for the propensity score model	PROC CAUSALTRT	PPSMODEL
ResponseProfile	Frequency counts for a binary outcome variable	MODEL	DIST=BIN
TreatmentEstimates	Parameter estimates for the treatment condition outcome model	PROC CAUSALTRT	POUTCOMEMOD
TreatmentProfile	Frequency counts for the treatment variable		Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC CAUSALTRT assigns a name to each graph it creates using ODS. You can use these names to refer to the graphs when you use ODS. Table 39.9 lists the names.

To request these graphs, ODS Graphics must be enabled and you must specify the option indicated in Table 39.9.

Table 39.9 Graphs Produced by PROC CAUSALTRT

ODS Graph Name	Description	Statement	Option
BootHistPlot	Histogram of bootstrap sample estimates	BOOTSTRAP	PLOTS=HIST
LogitPSPlot	Kernel density plot for the logit of the propensity score	PSMODEL	PLOTS=LOGITPSCORE
OutcomeByPScore	Scatter plot of outcome by propensity score	PSMODEL	PLOTS=OUTBYPSCORE
OutcomeByWeight	Scatter plot of outcome by weight	PSMODEL	PLOTS=OUTBYWEIGHT
PSCloud	Point clouds of propensity scores	PSMODEL	PLOTS=PSCLOUD
PSCovDenPlot	Kernel density plots of PSMODEL effects	PSMODEL	PLOTS=PSCOVDEN
PSDistPlot	Box plot of propensity score	PSMODEL	PLOTS=PSDIST
WeightCloud	Point clouds of weights	PSMODEL	PLOTS=WEIGHTCLOUD
WeightDistPlot	Box plot of weights	PSMODEL	PLOTS=WEIGHTDIST

Examples: CAUSALTRT Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The following examples illustrate some of the capabilities of the CAUSALTRT procedure. These examples are not intended to represent definitive analyses of the data sets that are presented here.

Example 39.1: Estimating the ATE Using Inverse Probability Weights

(View the complete code for this example (ctrtex1.sas) in the example repository.)

This example illustrates how the CAUSALTRT procedure uses inverse probability weights to estimate the average treatment effect (ATE). The question of interest is the effect that quitting smoking has on an individual's weight. The data for this example are a subset of data from the NHANES I Epidemiologic Follow-Up Study (NHEFS) in Hernán and Robins (2018). For the study, medical and behavioral information were collected during an initial physical examination, and again at follow-up interviews approximately one decade later.

The input data set SmokingWeight is shown in the following DATA step:

```
data SmokingWeight;
   input
   Sex Age Race Education Exercise BaseWeight Weight Change Activity
   YearsSmoke PerDay Quit;
   datalines;
   0 42 1 1 2 79.04 68.95 -10.09 0 29 30 0
   0 36 0 2 0 58.63 61.23 2.60 0 24 20 0
   1 56 1 2 2 56.81 66.22 9.41 0 26 20 0
   0 68 1 1 2 59.42 64.41 4.99 1 53 3 0
   0 40 0 2 1 87.09 92.08 4.99 1 19 20 0

   ... more lines ...

   0 45 0 1 0 63.05 64.41 1.36 0 29 40 0
   1 47 0 1 0 57.72 61.23 3.51 0 31 20 0
   1 51 0 3 0 62.71 . . 0 30 40 0
   0 68 0 1 1 52.39 57.15 4.76 1 46 15 0
   0 26 0 . 0 86.75 87.54 0.79 0 9 20 0
   0 29 0 2 1 90.83 106.59 15.76 1 14 30 1
;
```

The variables in the data are as follows:

- Activity: level of daily activity, with values 0, 1, and 2
- Age: age in 1971
- BaseWeight: weight in kilograms in 1971
- Change: difference in weight at the follow-up and baseline interviews (measured in kilograms)

- Education: level of education, with values 0, 1, 2, 3, and 4
- Exercise: amount of regular recreational exercise, with values 0, 1, and 2
- PerDay: number of cigarettes smoked per day in 1971
- Quit: 1 if an individual quit smoking between the initial and follow-up interviews; 0 otherwise
- Race: 0 for white; 1 otherwise
- Sex: 0 for male; 1 for female
- Weight: weight in kilograms at the follow-up interview
- YearsSmoke: number of years an individual has smoked

The following statements invoke PROC CAUSALTRT to estimate the average causal effect that quitting smoking has on weight gain:

```
ods graphics on;
proc causaltrt data=smokingweight covdiffps;
   class Sex Race Education Exercise Activity Quit /desc;
   psmodel Quit = Sex Age Education Exercise Activity YearsSmoke PerDay
          /plots=(PSDist pscovden(effects(Age YearsSmoke)) );
   model Change;
run;
```

The MODEL and PSMODEL statements are both required. A model for the treatment variable, Quit, is listed in the PSMODEL statement, and the outcome variable, Change, is specified in the MODEL statement. For estimation, the inverse probability weighting method with ratio adjustment (IPWR) is used by default because only a model for the treatment variable is specified. The estimation method, outcome variable, and treatment variable are all listed in the "Model Information" table in Output 39.1.1.

Output 39.1.1 Model Information

The CAUSALTRT Procedure

Model Information	
Data Set	WORK.SMOKINGWEIGHT
Distribution	Normal
Estimation Method	IPWR
Treatment Variable	Quit
Outcome Variable	Change

To examine whether the weights obtained from the propensity score model balance the covariates between the treatment and control conditions, you use the COVDIFFPS option in the PROC CAUSALTRT statement to request the computation of weighted and unweighted standardized mean differences (between treatment and control conditions) and variance ratios (treatment to control) for the covariates that are specified in the PSMODEL statement. These values are displayed in the "Covariate Differences for Propensity Score Model" table in Output 39.1.2. An improvement in the covariate balance after weighting would be indicated by the following two observations when you compare the unweighted and weighted columns in the table:

- smaller standardized mean differences in the weighted column

- variance ratios closer to one in the weighted column

Establishing covariate balance under the propensity score model is an important diagnostic step. Failing to achieve covariate balance would directly challenge the assumption of strong ignorability or conditional exchangeability, which is essential for estimating the causal effect. For more information about the assumptions and conditions required for the causal estimation methods, see the section "Causal Effects: Definitions, Assumptions, and Identification" on page 2533.

For the current example, Output 39.1.2 clearly shows that all standardized differences in the weighted column are smaller in magnitude than their counterparts in the unweighted column. All but one of the weighted standardized differences are smaller than 0.03 in magnitude. In addition, all but one of the variance ratios in the weighted column are closer to 1 than their counterparts in the unweighted column.

Notice that rows with blank values are for reference levels of the categorical covariates. The blank values indicate that they were not computed because they can be expressed in terms of the remaining levels.

Output 39.1.2 Standardized Mean Differences

The CAUSALTRT Procedure

Covariate Differences for Propensity Score Model

		Standardized Difference		Variance Ratio	
Parameter		**Unweighted**	**Weighted**	**Unweighted**	**Weighted**
Sex	**1**	-0.1603	-0.0200	0.9962	1.0006
Sex	**0**				
Age		0.2820	0.0318	1.0731	0.9847
Education	**5**	0.1660	0.0111	1.4610	1.0268
Education	**4**	-0.0270	0.0196	0.9167	1.0624
Education	**3**	-0.0472	-0.0015	0.9811	0.9994
Education	**2**	-0.1116	-0.0034	0.8498	0.9953
Education	**1**				
Exercise	**2**	0.0568	-0.0029	1.0252	0.9986
Exercise	**1**	0.0398	0.0166	1.0119	1.0049
Exercise	**0**				
Activity	**2**	0.0740	-0.0074	1.2182	0.9796
Activity	**1**	0.0268	0.0196	1.0043	1.0029
Activity	**0**				
YearsSmoke		0.1589	0.0253	1.1846	1.0894
PerDay		-0.2167	0.0027	1.1679	1.3323

Further diagnostics for assessing the covariate balance are requested by using the PLOTS= option in the PSMODEL statement.

The EFFECTS(Age YearsSmoke) suboption in the PSCOVDEN option requests that weighted and unweighted kernel density estimates be computed for the covariates Age and YearsSmoke. Output 39.1.3 displays the related paneled plots. The densities for both covariates show improved balance after weighting, as shown by the closely matched density curves in the weighted column. This observation echoes the preceding conclusion that the weights obtained from fitting the propensity score model lead to good covariate balance in Age and YearsSmoke. These density plots are available only for continuous covariates.

Output 39.1.4 shows the box plots that are requested by specifying PSDIST in the PLOTS= option. These box plots of propensity scores by treatment level indicate a similar distribution of propensity scores between treatment conditions.

Output 39.1.3 Density Estimates

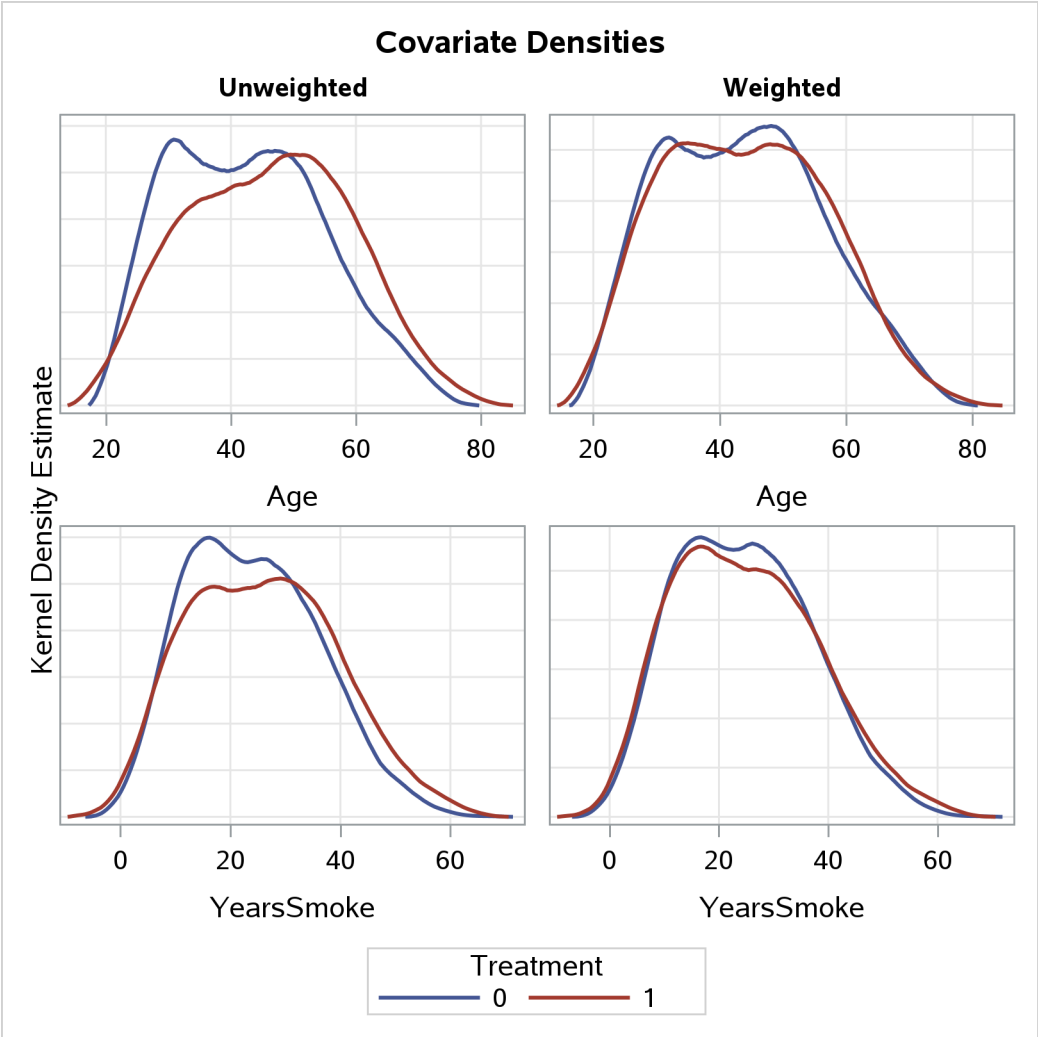

The estimates for the potential outcome means and ATE are displayed in Output 39.1.5. The estimate for the ATE of 3.1876 indicates that, on average, quitting smoking leads to an increase of about three kilograms in an individual's weight.

Output 39.1.4 Propensity Score Distributions

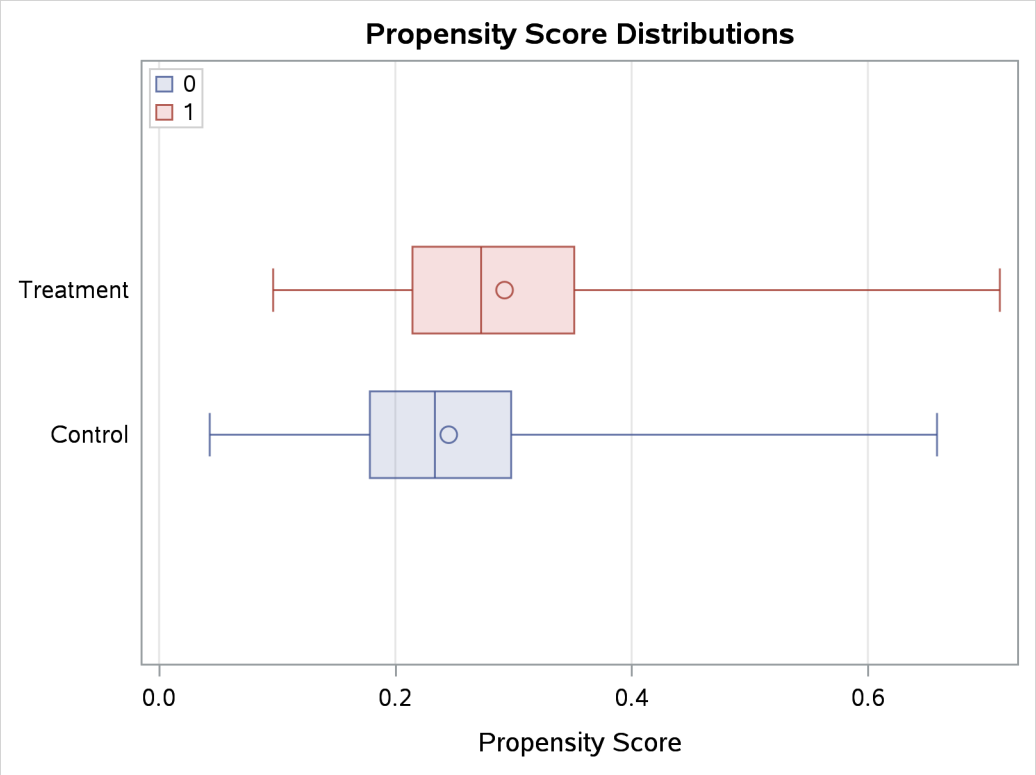

Output 39.1.5 IPWR Potential Outcome Means and ATE Estimates

The CAUSALTRT Procedure

Analysis of Causal Effect							
Parameter	Treatment Level	Estimate	Robust Std Err	Wald 95% Confidence Limits		Z	Pr > |Z|
POM	1	4.9824	0.4528	4.0949	5.8699	11.00	<.0001
POM	0	1.7948	0.2163	1.3709	2.2187	8.30	<.0001
ATE		3.1876	0.4972	2.2132	4.1621	6.41	<.0001

As an alternative to estimating the ATE by using the IPWR estimation method, you can estimate it by using inverse probability weights with either the basic inverse probability weighting method (METHOD=IPW) or the inverse probability weighting method with ratio and scale adjustments (METHOD=IPWS). The following statements use the METHOD=IPWS option to estimate the ATE:

```
proc causaltrt data=smokingweight method=IPWS;
   class Sex Race Education Exercise Activity Quit /desc;
   psmodel Quit = Sex Age Education Exercise Activity YearsSmoke PerDay;
   model Change;
run;
```

Because the effects that are specified in the PSMODEL statement have not changed from the preceding analysis, the same model is fit for the treatment assignment in the current example. Therefore, the predicted weights and measures of covariate balance do not change. What has changed by using the IPWS instead of

the IPWR estimation method is how the weights are incorporated into the estimation of the potential outcome means and ATE. For this example, there is little difference between the IPWS estimate of 3.1896 for the ATE, shown in Output 39.1.6, and the IPWR estimate of 3.1876, shown in Output 39.1.5.

Output 39.1.6 IPWS Potential Outcome Means and ATE Estimates

The CAUSALTRT Procedure

Analysis of Causal Effect							
Parameter	Treatment Level	Estimate	Robust Std Err	Wald 95% Confidence Limits		Z	Pr > |Z|
POM	1	4.9850	0.4530	4.0972	5.8728	11.01	<.0001
POM	0	1.7954	0.2163	1.3715	2.2193	8.30	<.0001
ATE		3.1896	0.4973	2.2149	4.1643	6.41	<.0001

Example 39.2: Estimating the ATE Using Augmented Inverse Probability Weights

(View the complete code for this example (ctrtex2.sas) in the example repository.)

This example illustrates how the CAUSALTRT procedure uses augmented inverse probability weights (AIPW) to perform doubly robust estimation of the ATE. This example uses the SmokingWeight data set that is described in Example 39.1 and estimates the same causal effect: how quitting smoking might affect an individual's change in weight.

The AIPW method combines modeling the treatment assignment and modeling the outcome variable to estimate the potential outcome means and the ATE. The following statements invoke PROC CAUSALTRT to estimate the causal effect of quitting smoking on weight gain by using the AIPW method:

```
ods graphics on;
proc causaltrt data=smokingweight method=AIPW nthreads=4;
   class Sex Race Education Exercise Activity Quit /desc;
   psmodel Quit = Sex Age Education Exercise Activity YearsSmoke PerDay ;
   model Change = Sex Age Exercise Activity BaseWeight;
   bootstrap bootci(all) plot=hist seed=1234;
run;
```

The model for the treatment assignment is specified in the PSMODEL statement, and the outcome model is specified in the MODEL statement. The AIPW method is a doubly robust estimation method and provides unbiased estimates for the ATE even when one of the models is misspecified. The model for the treatment assignment is the same model that is used in Example 39.1. Therefore, the estimates for the weights and measures of covariate balance (Output 39.1.2 and Output 39.1.3) are the same for this example.

Although the AIPW estimates for the ATE and potential outcome means are doubly robust, the empirical estimates for their standard errors (the default estimates of standard errors) are not. You can request bootstrap-based estimation of the standard errors and confidence limits by using the BOOTSTRAP statement. The BOOTCI(ALL) option requests the estimation of all three bootstrap-based confidence intervals that PROC CAUSALTRT implements: the bias-corrected percentile method (default), percentile method, and normal approximation method. The bootstrap-based confidence intervals are added to the "Analysis of Causal Effect" table. The standard Wald confidence intervals and the three bootstrap-based confidence intervals all provide

similar limits as shown in Output 39.2.1. The robust standard error estimate and the bootstrap standard error estimate are also very close. The AIPW estimate for the ATE of 3.3049 is larger than the IPWS estimate for the ATE of 3.1896 (as shown in Output 39.1.6) and also larger than the IPWR estimate of 3.1876 (as shown in Output 39.1.5).

Output 39.2.1 AIPW Potential Outcome Means and ATE Estimates

The CAUSALTRT Procedure

Analysis of Causal Effect

Parameter	Treatment Level	Estimate	Robust Std Err	Bootstrap Std Err	Wald 95% Confidence Limits		Bootstrap Wald 95% Confidence Limits		Bootstrap Percentile 95% Confidence Limits	
POM	1	5.0830	0.4495	0.4588	4.2019	5.9641	4.1839	5.9822	4.1907	6.0400
POM	0	1.7781	0.2156	0.2227	1.3556	2.2007	1.3416	2.2147	1.3175	2.2155
ATE		3.3049	0.4911	0.5016	2.3423	4.2675	2.3218	4.2879	2.3511	4.3029

Analysis of Causal Effect

Parameter	Bootstrap Bias Corrected 95% Confidence Limits		Z	Pr > |Z|
POM	4.1736	6.0079	11.31	<.0001
POM	1.3068	2.2127	8.25	<.0001
ATE	2.3502	4.2981	6.73	<.0001

To inspect the distribution of the bootstrap estimates, you request histograms of the estimates by specifying the PLOTS=HIST option in the BOOTSTRAP statement. In Output 39.2.2, the histograms are collected in a panel and show no extreme estimates. Normal densities are overlaid in the histograms. Output 39.2.2 shows that the distributions of the potential outcome means (POM) and ATE estimates are approximately normal. To reproduce the random number stream that was used to generate the bootstrap estimates, you must use the same SEED= value and the same number of threads for the analytic computations. When bootstrap resampling is performed, a column that indicates the number of threads used is added to the "Analysis of Causal Effect" table; this column is not displayed but is available if you use the ODS OUTPUT statement to save the table as an output data set. You can also display this column by modifying the corresponding template.

Output 39.2.2 Histograms of Bootstrap Estimates

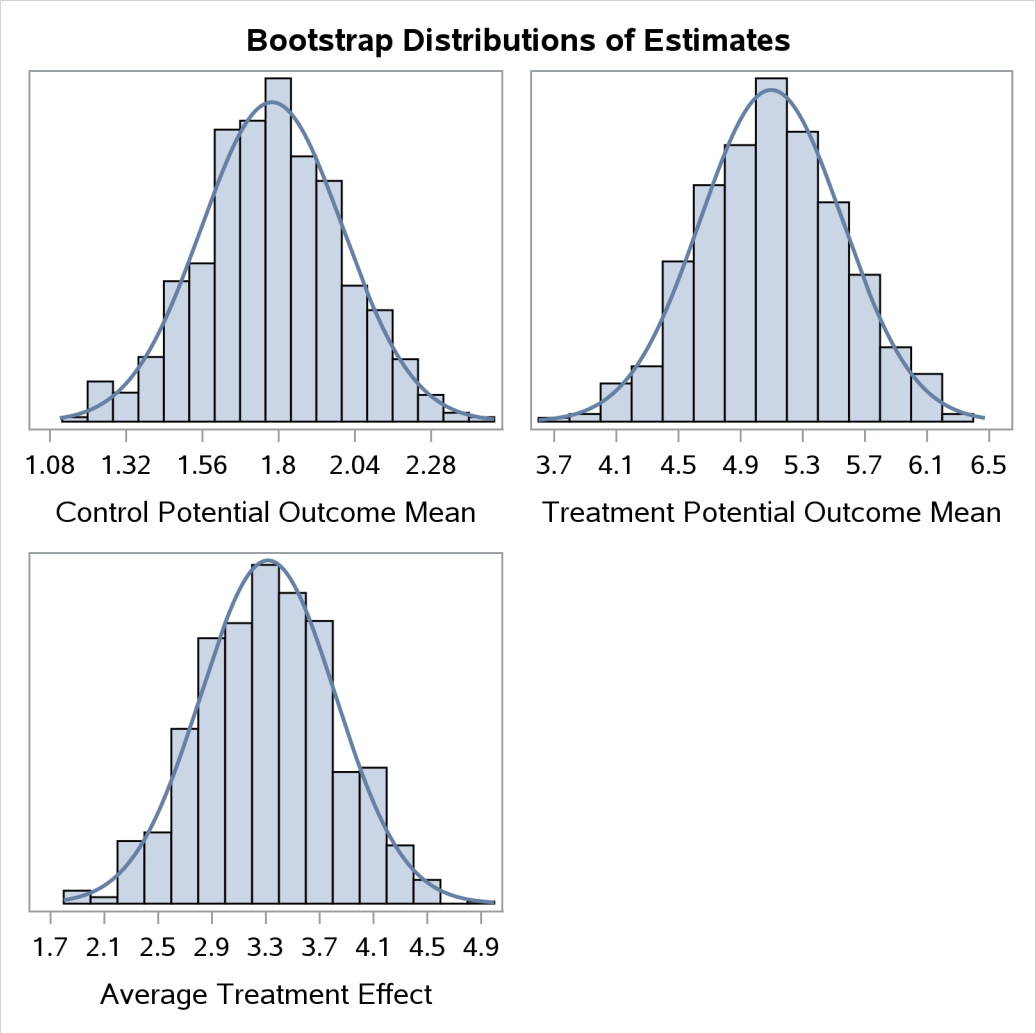

Example 39.3: Estimating Treatment Effects Using Regression Adjustment

(View the complete code for this example (ctrtex3.sas) in the example repository.)

This example demonstrates how the CAUSALTRT procedure can use regression adjustment to estimate both the ATE (average treatment effect) and the ATT (average treatment effect for the treated). The question of interest in this example is the effect that attending a Catholic high school has on a student's performance in mathematics. The data for this example are a subset of data from the National Educational Longitudinal Study of 1988 in Murnane and Willett (2011). Demographic information was collected from the students and their families along with standardized test scores in 1988 and subsequent follow-up years. The outcome for this example is the student's score on the mathematics portion of the standardized tests in 1992.

The input data set School is shown in the following DATA step:

```
data School;
   input
   Income $ FatherEd $ MotherEd $ Math BaseMath  Catholic $;
   datalines;
   Middle HighSchool HighSchool 49.77 50.27 Yes
   Middle Unknown Unknown 59.84 51.52 Yes
   Middle NoHighSchool Postsecondary 50.38 47.56 Yes
   Middle Unknown Unknown 45.03 46.60 Yes
   High HighSchool HighSchool 54.26 60.22 Yes
   Low NoHighSchool HighSchool 55.37 58.63 Yes

   ... more lines ...

   High SomeSecondary HighSchool 43.28 43.27 Yes
   Middle HighSchool HighSchool 41.69 48.29 Yes
   High College College 56.60 60.15 Yes
   High College SomeSecondary 56.29 59.53 Yes
   High College Postsecondary 58.16 57.06 Yes
   High College SomeSecondary 63.57 63.51 Yes
;
```

The data set School consists of records for 5,671 high school students whose 1988 total family income was at most $75,000. The variables in the data are as follows:

- BaseMath: student's score on the mathematics portion of a standardized test in 1988
- Catholic: Yes if a student attended a Catholic high school; No otherwise
- FatherEd: highest level of education completed by the student's father, with six levels
- Income: classification based on total family income, with values of Low, Middle, and High
- Math: student's score on the mathematics portion of a standardized test in 1992
- MotherEd: highest level of education completed by the student's mother, with six levels

The following statements invoke the PROC CAUSALTRT procedure to estimate the ATE of attending a Catholic high school on math scores:

```
proc causaltrt data=school method=regadj poutcomemod;
   class Income FatherEd MotherEd;
   psmodel Catholic(ref='No');
   model Math = BaseMath Income FatherEd MotherEd;
run;
```

The METHOD=REGADJ option requests that regression adjustment be used to estimate the ATE. For this estimation method, models are fit for the outcome variable separately for each treatment condition. The treatment variable, Catholic, is listed in PSMODEL statement. The **ref='No'** option identifies Catholic='No' as the reference or control level. The outcome variable Math is specified in the MODEL statement along with the effects used to fit the outcome model.

The POUTCOMEMOD option in the PROC CAUSALTRT statement requests the display of the parameter estimates for the outcome model fit within each treatment condition. Two tables are produced: one for the control condition (Output 39.3.1) and one for the treatment condition (Output 39.3.2).

Output 39.3.1 Control Group Estimates

The CAUSALTRT Procedure

Outcome Model Estimates for Control Group							
Parameter		Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		10.9061	0.4808	9.9638	11.8484	514.5891	<.0001
BaseMath		0.7774	0.0078	0.7622	0.7927	9950.9801	<.0001
Income	High	0.4244	0.1818	0.0681	0.7807	5.4516	0.0196
Income	Low	-0.7194	0.1944	-1.1004	-0.3385	13.7017	0.0002
Income	Middle	0	.	.	.	.	.
FatherEd	College	0.3857	0.3313	-0.2637	1.0351	1.3548	0.2444
FatherEd	HighScho	-0.3112	0.2903	-0.8803	0.2579	1.1488	0.2838
FatherEd	NoHighSc	-1.1478	0.3302	-1.7950	-0.5005	12.0806	0.0005
FatherEd	Postseco	0.5464	0.3827	-0.2037	1.2965	2.0381	0.1534
FatherEd	SomeSeco	0.1948	0.3066	-0.4061	0.7957	0.4036	0.5252
FatherEd	Unknown	0	.	.	.	.	.
MotherEd	College	0.6255	0.3747	-0.1089	1.3600	2.7863	0.0951
MotherEd	HighScho	-0.00768	0.3222	-0.6392	0.6238	0.0006	0.9810
MotherEd	NoHighSc	-0.5579	0.3610	-1.2654	0.1496	2.3890	0.1222
MotherEd	Postseco	0.1472	0.4370	-0.7093	1.0036	0.1134	0.7363
MotherEd	SomeSeco	0.4911	0.3346	-0.1647	1.1469	2.1540	0.1422
MotherEd	Unknown	0	.	.	.	.	.

Output 39.3.2 Treatment Group Estimates

The CAUSALTRT Procedure

Outcome Model Estimates for Treatment Group							
Parameter		Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		15.6156	1.7186	12.2471	18.9840	82.5561	<.0001
BaseMath		0.7274	0.0244	0.6795	0.7752	887.3010	<.0001
Income	High	-0.3734	0.5055	-1.3641	0.6173	0.5458	0.4600
Income	Low	-1.9732	0.6744	-3.2950	-0.6514	8.5605	0.0034
Income	Middle	0	.	.	.	.	.
FatherEd	College	2.3426	1.0035	0.3758	4.3095	5.4497	0.0196
FatherEd	HighScho	0.6428	1.0026	-1.3223	2.6080	0.4111	0.5214
FatherEd	NoHighSc	0.0547	1.0842	-2.0702	2.1796	0.0025	0.9598
FatherEd	Postseco	1.2191	1.0603	-0.8591	3.2973	1.3219	0.2503
FatherEd	SomeSeco	1.9739	0.9848	0.0436	3.9041	4.0171	0.0450
FatherEd	Unknown	0	.	.	.	.	.
MotherEd	College	-0.1707	1.0663	-2.2607	1.9193	0.0256	0.8728
MotherEd	HighScho	-1.3311	1.0506	-3.3903	0.7280	1.6053	0.2051
MotherEd	NoHighSc	-0.6978	1.3538	-3.3513	1.9557	0.2656	0.6063
MotherEd	Postseco	-0.8308	1.1261	-3.0379	1.3764	0.5442	0.4607
MotherEd	SomeSeco	-0.8607	1.0016	-2.8238	1.1025	0.7383	0.3902
MotherEd	Unknown	0	.	.	.	.	.

The potential outcome means and ATE are estimated by averaging predicted values for each student from each of the outcome models. The ATE estimate of 1.6920 in Output 39.3.3 indicates that attending a Catholic high school has a positive effect on math scores. This estimate for the treatment effect is much smaller than the estimate of 3.8949 that you would obtain by comparing the unadjusted average math scores for each treatment condition.

Output 39.3.3 Regression Adjustment Potential Outcome Means and ATE Estimates

The CAUSALTRT Procedure

Analysis of Causal Effect							
Parameter	Treatment Level	Estimate	Robust Std Err	Wald 95% Confidence Limits		Z	Pr > |Z|
POM	Yes	52.5790	0.2645	52.0605	53.0975	198.76	<.0001
POM	No	50.8871	0.1280	50.6363	51.1378	397.70	<.0001
ATE		1.6920	0.2536	1.1950	2.1889	6.67	<.0001

Conceptually, the ATE in this example is the average effect on the math scores for everyone in the population who would have attended a Catholic high school versus a non-Catholic high school. Causal effect analysis such as the current example enables you to estimate the ATE with causal interpretation, which is of general scientific interest.

However, for policy-making purposes, sometimes it is more interesting to examine the effectiveness of a program or treatment for those who did participate. The average treatment effect for the treated (ATT) is such a concept; it measures the treatment effect conditional on those who receive the treatment condition. PROC CAUSALTRT provides you two methods to estimate ATT: the inverse probability weighting method with ratio adjustment (METHOD=IPWR) or regression adjustment (METHOD=REGADJ).

The remainder of this section demonstrates the estimation of ATT for the Catholic high school data example. To estimate the average treatment effect for the treated, you specify the ATT option in the PROC CAUSALTRT statement. The following SAS statements estimate the ATT using the regression adjustment estimation method and the same outcome model as specified in the previous analysis for ATE:

```
proc causaltrt data=school method=regadj att;
   class Income FatherEd MotherEd;
   psmodel Catholic(ref='No');
   model Math = BaseMath Income FatherEd MotherEd;
run;
```

The "Analysis of Causal Effect" table now displays estimates for the ATT and the potential outcome means conditional on receiving the treatment condition. The estimate for the ATT is 1.5727, which is slightly smaller than the ATE estimate.

Output 39.3.4 Regression Adjustment ATT Estimates

The CAUSALTRT Procedure

Analysis of Causal Effect							
Parameter	Treatment Level	Estimate	Robust Std Err	Wald 95% Confidence Limits		Z	Pr > |Z|
POM	Yes	54.5395	0.3475	53.8583	55.2207	156.93	<.0001
POM	No	52.9668	0.3034	52.3722	53.5614	174.59	<.0001
ATT		1.5727	0.2268	1.1281	2.0173	6.93	<.0001

From a policy-making point of view, the average effect that attending a Catholic high school would have on students who are in non-Catholic high school would also be of interest. Such a mirror concept of ATT is also known as ATU, average treatment effect for the untreated. Although the CAUSALTRT procedure does not provide a direct option for estimating ATU, you can still achieve that purpose by using the ATT option with the reverse treatment designation. In the following PROC CAUSALTRT statements, the **ref='Yes'** option in the PSMODEL statement redefines the treatment condition to be attending a non-Catholic high school.

```
proc causaltrt data=school method=regadj att;
   class Income FatherEd MotherEd;
   psmodel Catholic(ref='Yes');
   model Math = BaseMath Income FatherEd MotherEd;
run;
```

The ATT estimate of –1.7059 in Output 39.3.5 is the average causal effect that attending non-Catholic high schools has on the math scores of students in non-Catholic high schools. Reversing the sign of this estimate provides an estimate for the ATU: the average treatment effect that attending a Catholic school would have for those students in non-Catholic high schools. Hence, the ATU in the current example is 1.7059.

In some research studies, researchers would like to compare ATT and ATU to investigate the so-called effect heterogeneity issue. In the current example, the ATU is only slightly larger (by about 0.13 points) than the ATT. Considering the fact that all potential outcome means are about 50, it thus appears to be safe to say that effect heterogeneity, as indicated by the 0.13 point difference, is not an issue in the current example.

Output 39.3.5 AIPW Potential Outcome Means and ATE Estimates

The CAUSALTRT Procedure

Analysis of Causal Effect

Parameter	Treatment Level	Estimate	Robust Std Err	Wald 95% Confidence Limits		Z	Pr > |Z|
POM	No	50.6447	0.1338	50.3825	50.9068	378.60	<.0001
POM	Yes	52.3505	0.2717	51.8179	52.8831	192.66	<.0001
ATT		-1.7059	0.2608	-2.2171	-1.1947	-6.54	<.0001

References

Bang, H., and Robins, J. M. (2005). "Doubly Robust Estimation in Missing Data and Causal Inference Models." *Biometrics* 61:962–973.

Berzuini, C., Dawid, P., and Bernardinelli, L., eds. (2012). *Causality: Statistical Perspectives and Applications*. Chichester, UK: John Wiley & Sons.

Cole, S. R., and Frangakis, C. E. (2009). "The Consistency Statement in Causal Inference: A Definition or an Assumption?" *Epidemiology* 20:3–5.

Hernán, M. A., and Robins, J. M. (2018). *Causal Inference*. Boca Raton, FL: Chapman & Hall/CRC. Forthcoming.

Imbens, G. W., and Rubin, D. B. (2015). *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. New York: Cambridge University Press.

Lunceford, J. K., and Davidian, M. (2004). "Stratification and Weighting via the Propensity Score in Estimation of Causal Treatment Effects: A Comparative Study." *Statistics in Medicine* 23:2937–2960.

Morgan, S. L., and Winship, C. (2015). *Counterfactuals and Causal Inference: Methods and Principles for Social Research*. 2nd ed. New York: Cambridge University Press.

Murnane, R. J., and Willett, J. B. (2011). *Methods Matter: Improving Causal Inference in Educational and Social Science Research*. New York: Oxford University Press.

Neyman, J., Dabrowska, D. M., and Speed, T. P. (1990). "On the Application of Probability Theory to Agricultural Experiments: Essay on Principles, Section 9." *Statistical Science* 5:465–472. Translated and edited by Dabrowska and Speed from the Polish original by Neyman (1923).

Pierce, D. A. (1982). "The Asymptotic Effect of Substituting Estimators for Parameters in Certain Types of Statistics." *Annals of Statistics* 10:475–478.

Robins, J. M., Rotnitzky, A., and Zhao, L. P. (1995). "Analysis of Semiparametric Regression Models for Repeated Outcomes in the Presence of Missing Data." *Journal of the American Statistical Association* 90:106–121.

Rosenbaum, P. R., and Rubin, D. B. (1983). "The Central Role of the Propensity Score in Observational Studies for Causal Effects." *Biometrika* 70:41–55.

Rubin, D. B. (1980). "Comment on D. Basu, 'Randomization Analysis of Experimental Data: The Fisher Randomization Test'." *Journal of the American Statistical Association* 75:591–593.

Rubin, D. B. (1990). "Comment: Neyman (1923) and Causal Inference in Experiments and Observational Studies." *Statistical Science* 5:472–480.

Rubin, D. B. (2005). "Causal Inference Using Potential Outcomes: Design, Modeling, Decisions." *Journal of the American Statistical Association* 100:322–331.

Stefanski, L. A., and Boos, D. D. (2002). "The Calculus of M-Estimation." *American Statistician* 56:29–38.

VanderWeele, T. J. (2009). "Concerning the Consistency Assumption in Causal Inference." *Epidemiology* 20:880–883.

Wooldridge, J. M. (2010). *Econometric Analysis of Cross Section and Panel Data*. 2nd ed. Cambridge, MA: MIT Press.

Chapter 40
The CLUSTER Procedure

Contents

Overview: CLUSTER Procedure **2564**
Getting Started: CLUSTER Procedure **2565**
Syntax: CLUSTER Procedure **2572**
- PROC CLUSTER Statement 2572
- BY Statement 2582
- COPY Statement 2583
- FREQ Statement 2583
- ID Statement 2583
- RMSSTD Statement 2584
- VAR Statement 2584

Details: CLUSTER Procedure **2585**
- Clustering Methods 2585
- Miscellaneous Formulas 2592
- Ultrametrics 2593
- Algorithms 2594
- Computational Resources 2594
- Missing Values 2595
- Ties 2595
- Size, Shape, and Correlation 2596
- Output Data Set 2597
- Displayed Output 2599
- ODS Table Names 2602
- ODS Graphics 2602

Examples: CLUSTER Procedure **2603**
- Example 40.1: Cluster Analysis of Flying Mileages between 10 American Cities . . . 2603
- Example 40.2: Crude Birth and Death Rates 2610
- Example 40.3: Cluster Analysis of Fisher's Iris Data 2622
- Example 40.4: Evaluating the Effects of Ties 2636

References **2646**

Overview: CLUSTER Procedure

The CLUSTER procedure hierarchically clusters the observations in a SAS data set by using one of 11 methods. The data can be coordinates or distances. If the data are coordinates, PROC CLUSTER computes (possibly squared) Euclidean distances. If you want non-Euclidean distances, use the DISTANCE procedure (see Chapter 43) to compute an appropriate distance data set that can then be used as input to PROC CLUSTER.

The clustering methods are: average linkage, the centroid method, complete linkage, density linkage (including Wong's hybrid and *k*th-nearest-neighbor methods), maximum likelihood for mixtures of spherical multivariate normal distributions with equal variances but possibly unequal mixing proportions, the flexible-beta method, McQuitty's similarity analysis, the median method, single linkage, two-stage density linkage, and Ward's minimum-variance method. Each method is described in the section "Clustering Methods" on page 2585.

All methods are based on the usual agglomerative hierarchical clustering procedure. Each observation begins in a cluster by itself. The two closest clusters are merged to form a new cluster that replaces the two old clusters. Merging of the two closest clusters is repeated until only one cluster is left. The various clustering methods differ in how the distance between two clusters is computed.

The CLUSTER procedure is not practical for very large data sets because the CPU time is roughly proportional to the square or cube of the number of observations. The FASTCLUS procedure (see Chapter 45) requires time proportional to the number of observations and thus can be used with much larger data sets than PROC CLUSTER. If you want to cluster a very large data set hierarchically, use PROC FASTCLUS for a preliminary cluster analysis to produce a large number of clusters. Then use PROC CLUSTER to cluster the preliminary clusters hierarchically. This method is illustrated in Example 40.3.

PROC CLUSTER displays a history of the clustering process, showing statistics useful for estimating the number of clusters in the population from which the data are sampled. It creates a dendrogram when ODS Graphics is enabled. PROC CLUSTER also creates an output data set that can be used by the TREE procedure to output the cluster membership at any desired level. For example, to obtain the six-cluster solution, you could first use PROC CLUSTER with the OUTTREE= option, and then use this output data set as the input data set to the TREE procedure. With PROC TREE, specify the NCLUSTERS=6 and the OUT= options to obtain the six-cluster solution. For an example, see Example 127.1 in Chapter 127, "The TREE Procedure."

For coordinate data, Euclidean distances are computed from differences between coordinate values. The use of differences has several important consequences:

- For differences to be valid, the variables must have an interval or stronger scale of measurement. Ordinal or ranked data are generally not appropriate for cluster analysis.
- For Euclidean distances to be comparable, equal differences should have equal practical importance. You might need to transform the variables linearly or nonlinearly to satisfy this condition. For example, if one variable is measured in dollars and one in euros, you might need to convert to the same currency. Or, if ratios are more meaningful than differences, take logarithms.
- Variables with large variances tend to have more effect on the resulting clusters than variables with small variances. If you consider all variables to be equally important, you can use the STD option in PROC CLUSTER to standardize the variables to mean 0 and standard deviation 1. However,

standardization is not always appropriate. See Milligan and Cooper (1987) for a Monte Carlo study on various methods of variable standardization. You should remove outliers before using PROC CLUSTER with the STD option unless you specify the TRIM= option. The STDIZE procedure (see Chapter 115) provides additional methods for standardizing variables and imputing missing values.

The ACECLUS procedure (see Chapter 27) is useful for linear transformations of the variables if any of the following conditions hold:

- You have no idea how the variables should be scaled.
- You want to detect natural clusters regardless of whether some variables have more influence than others.
- You want to use a clustering method designed for finding compact clusters, but you want to be able to detect elongated clusters.

Agglomerative hierarchical clustering is discussed in all standard references on cluster analysis, such as Anderberg (1973); Sneath and Sokal (1973); Hartigan (1975); Everitt (1980); Spath (1980). An especially good introduction is given by Massart and Kaufman (1983). Anyone considering doing a hierarchical cluster analysis should study the Monte Carlo results of Milligan (1980); Milligan and Cooper (1985); Cooper and Milligan (1988).

Other essential, though more advanced, references on hierarchical clustering include Hartigan (1977, pp. 60–68; 1981), Wong (1982); Wong and Schaack (1982); Wong and Lane (1983). For a discussion of the confusing terminology in hierarchical cluster analysis, see Blashfield and Aldenderfer (1978).

Getting Started: CLUSTER Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/clusgs.sas`.)

This example shows how you can use the CLUSTER procedure to compute hierarchical clusters of observations in a SAS data set.

Suppose you want to determine whether national figures for birth rates, death rates, and infant death rates can be used to categorize countries. Previous studies indicate that the clusters computed from this type of data can be elongated and elliptical. Thus, you need to perform a linear transformation on the raw data before the cluster analysis.

The following data[1] from Rouncefield (1995) are birth rates, death rates, and infant death rates for 97 countries. The DATA step creates the SAS data set Poverty:

[1] These data have been compiled from the *United Nations Demographic Yearbook 1990* (United Nations publications, Sales No. E/F.91.XII.1, copyright 1991, United Nations, New York) and are reproduced with the permission of the United Nations.

```
data Poverty;
   input Birth Death InfantDeath Country $20. @@;
   datalines;
24.7  5.7  30.8 Albania              12.5 11.9  14.4 Bulgaria
13.4 11.7  11.3 Czechoslovakia       12   12.4   7.6 Former E. Germany
11.6 13.4  14.8 Hungary              14.3 10.2    16 Poland
13.6 10.7  26.9 Romania                14    9  20.2 Yugoslavia

   ... more lines ...

41.7 10.3    66 Zimbabwe
;
```

The data set Poverty contains the character variable Country and the numeric variables Birth, Death, and InfantDeath, which represent the birth rate per thousand, death rate per thousand, and infant death rate per thousand. The `$20.` in the INPUT statement specifies that the variable Country is a character variable with a length of 20. The double trailing at sign (@@) in the INPUT statement holds the input line for further iterations of the DATA step, specifying that observations are input from each line until all values are read.

Because the variables in the data set do not have equal variance, you must perform some form of scaling or transformation. One method is to standardize the variables to mean zero and variance one. However, when you suspect that the data contain elliptical clusters, you can use the ACECLUS procedure to transform the data such that the resulting within-cluster covariance matrix is spherical. The procedure obtains approximate estimates of the pooled within-cluster covariance matrix and then computes canonical variables to be used in subsequent analyses.

The following statements perform the ACECLUS transformation by using the SAS data set Poverty. The OUT= option creates an output SAS data set called Ace that contains the canonical variable scores:

```
proc aceclus data=Poverty out=Ace p=.03 noprint;
   var Birth Death InfantDeath;
run;
```

The P= option specifies that approximately 3% of the pairs are included in the estimation of the within-cluster covariance matrix. The NOPRINT option suppresses the display of the output. The VAR statement specifies that the variables Birth, Death, and InfantDeath are used in computing the canonical variables.

The following statements invoke the CLUSTER procedure, using the SAS data set Ace created in the previous PROC ACECLUS run:

```
ods graphics on;

proc cluster data=Ace method=ward ccc pseudo print=15 out=tree
   plots=den(height=rsq);
   var can1-can3;
   id country;
run;

ods graphics off;
```

The ODS GRAPHICS ON statement enables ODS Graphics. Ward's minimum-variance clustering method is specified by the METHOD= option. The CCC option displays the cubic clustering criterion, and the PSEUDO option displays pseudo F and t^2 statistics. The PRINT=15 option displays only the last 15 generations of

the cluster history. By default, when ODS Graphics is enabled, a dendrogram displaying the semipartial R square is displayed on the X axis. The option PLOTS=DEN(HEIGHT=RSQ) requests a dendrogram with R square displayed instead.

The VAR statement specifies that the canonical variables computed in the ACECLUS procedure are used in the cluster analysis. The ID statement selects the variable Country as the Y axis variable in the dendrogram and also specifies that Country should be added to the Tree output data set.

PROC CLUSTER first displays the table of eigenvalues of the covariance matrix (Figure 40.1). These eigenvalues are used in the computation of the cubic clustering criterion. The first two columns list each eigenvalue and the difference between the eigenvalue and its successor. The last two columns display the individual and cumulative proportion of variation associated with each eigenvalue.

Figure 40.1 Table of Eigenvalues of the Covariance Matrix

The CLUSTER Procedure
Ward's Minimum Variance Cluster Analysis

Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	64.5500051	54.7313223	0.8091	0.8091
2	9.8186828	4.4038309	0.1231	0.9321
3	5.4148519		0.0679	1.0000

Root-Mean-Square Total-Sample Standard Deviation	5.156987

Root-Mean-Square Distance Between Observations	12.63199

Figure 40.2 displays the last 15 generations of the cluster history. First listed are the number of clusters and the names of the clusters joined. The observations are identified either by the ID value or by CL*n*, where *n* is the number of the cluster. Next, PROC CLUSTER displays the number of observations in the new cluster and the semipartial R square. The latter value represents the decrease in the proportion of variance accounted for by joining the two clusters.

Figure 40.2 Cluster History

Cluster History										
Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Tie
15	Oman	CL37	5	0.0039	.957	.933	6.03	132	12.1	
14	CL31	CL22	13	0.0040	.953	.928	5.81	131	9.7	
13	CL41	CL17	32	0.0041	.949	.922	5.70	131	13.1	
12	CL19	CL21	10	0.0045	.945	.916	5.65	132	6.4	
11	CL39	CL15	9	0.0052	.940	.909	5.60	134	6.3	
10	CL76	CL27	6	0.0075	.932	.900	5.25	133	18.1	
9	CL23	CL11	15	0.0130	.919	.890	4.20	125	12.4	
8	CL10	Afghanistan	7	0.0134	.906	.879	3.55	122	7.3	
7	CL9	CL25	17	0.0217	.884	.864	2.26	114	11.6	
6	CL8	CL20	14	0.0239	.860	.846	1.42	112	10.5	
5	CL14	CL13	45	0.0307	.829	.822	0.65	112	59.2	
4	CL16	CL7	28	0.0323	.797	.788	0.57	122	14.8	
3	CL12	CL6	24	0.0323	.765	.732	1.84	153	11.6	
2	CL3	CL4	52	0.1782	.587	.613	-.82	135	48.9	
1	CL5	CL2	97	0.5866	.000	.000	0.00	.	135	

Next listed is the squared multiple correlation, R square, which is the proportion of variance accounted for by the clusters. Figure 40.2 shows that, when the data are grouped into three clusters, the proportion of variance accounted for by the clusters (R square) is just under 77%. The approximate expected value of R square is given in the ERSq column. This expectation is approximated under the null hypothesis that the data have a uniform distribution instead of forming distinct clusters.

The next three columns display the values of the cubic clustering criterion (CCC), pseudo F (PSF), and t^2 (PST2) statistics. These statistics are useful for estimating the number of clusters in the data.

The final column in Figure 40.2 lists ties for minimum distance; a blank value indicates the absence of a tie. A tie means that the clusters are indeterminate and that changing the order of the observations might change the clusters. See Example 40.4 for ways to investigate the effects of ties.

Figure 40.3 plots the three statistics for estimating the number of clusters. Peaks in the plot of the cubic clustering criterion with values greater than 2 or 3 indicate good clusters; peaks with values between 0 and 2 indicate possible clusters. Large negative values of the CCC can indicate outliers. In Figure 40.3, there is a local peak of the CCC when the number of clusters is three. The CCC drops at four clusters and then steadily increases, leveling off at eleven clusters.

Another method of judging the number of clusters in a data set is to look at the pseudo F statistic (PSF). Relatively large values indicate good numbers of clusters. In Figure 40.3, the pseudo F statistic suggests three clusters or eleven clusters.

Figure 40.3 Plot of Statistics for Estimating the Number of Clusters

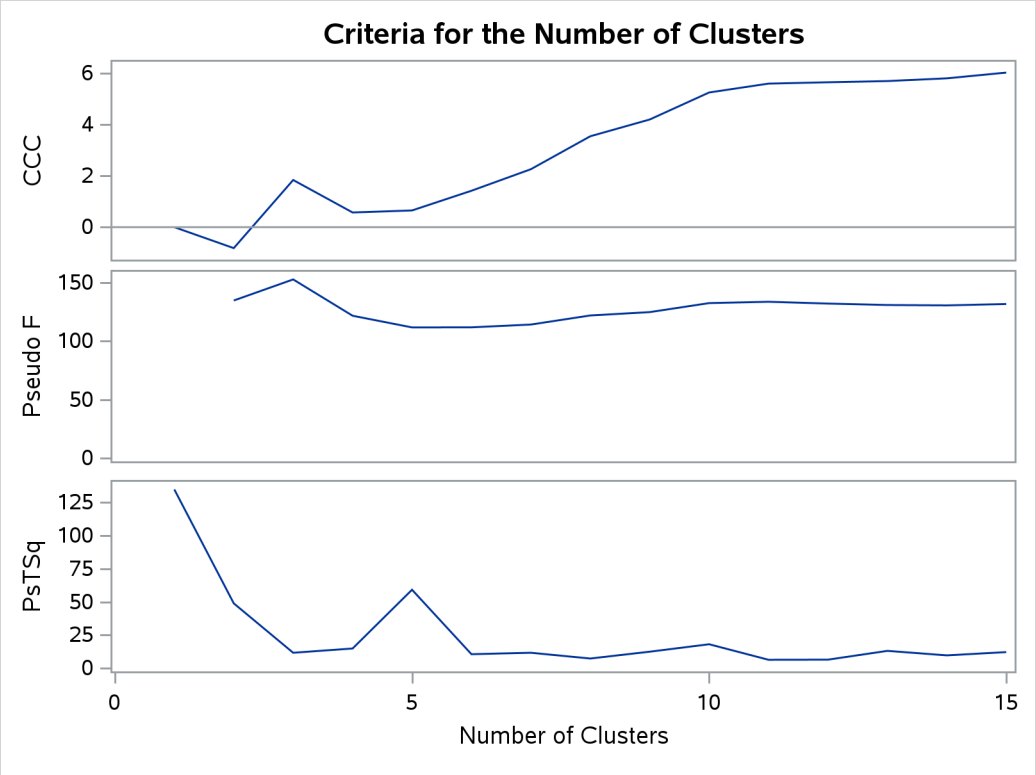

To interpret the values of the pseudo t^2 statistic, look down the column or look at the plot from right to left until you find the first value that is markedly larger than the previous value, then move back up the column or to the right in the plot by one step in the cluster history. In Figure 40.3, you can see possibly good clustering levels at eleven clusters, six clusters, three clusters, and two clusters.

Considered together, these statistics suggest that the data can be clustered into eleven clusters or three clusters. The following statements examine the results of clustering the data into three clusters.

Figure 40.4 displays the dendrogram. The figure provides a graphical view of the information in Figure 40.2. As the number of branches grows to the left from the root, the R square approaches 1; the first three clusters (branches of the tree) account for over half of the variation (about 77%, from Figure 40.4). In other words, only three clusters are necessary to explain over three-fourths of the variation.

Figure 40.4 Dendrogram of Clusters versus R-Square Values

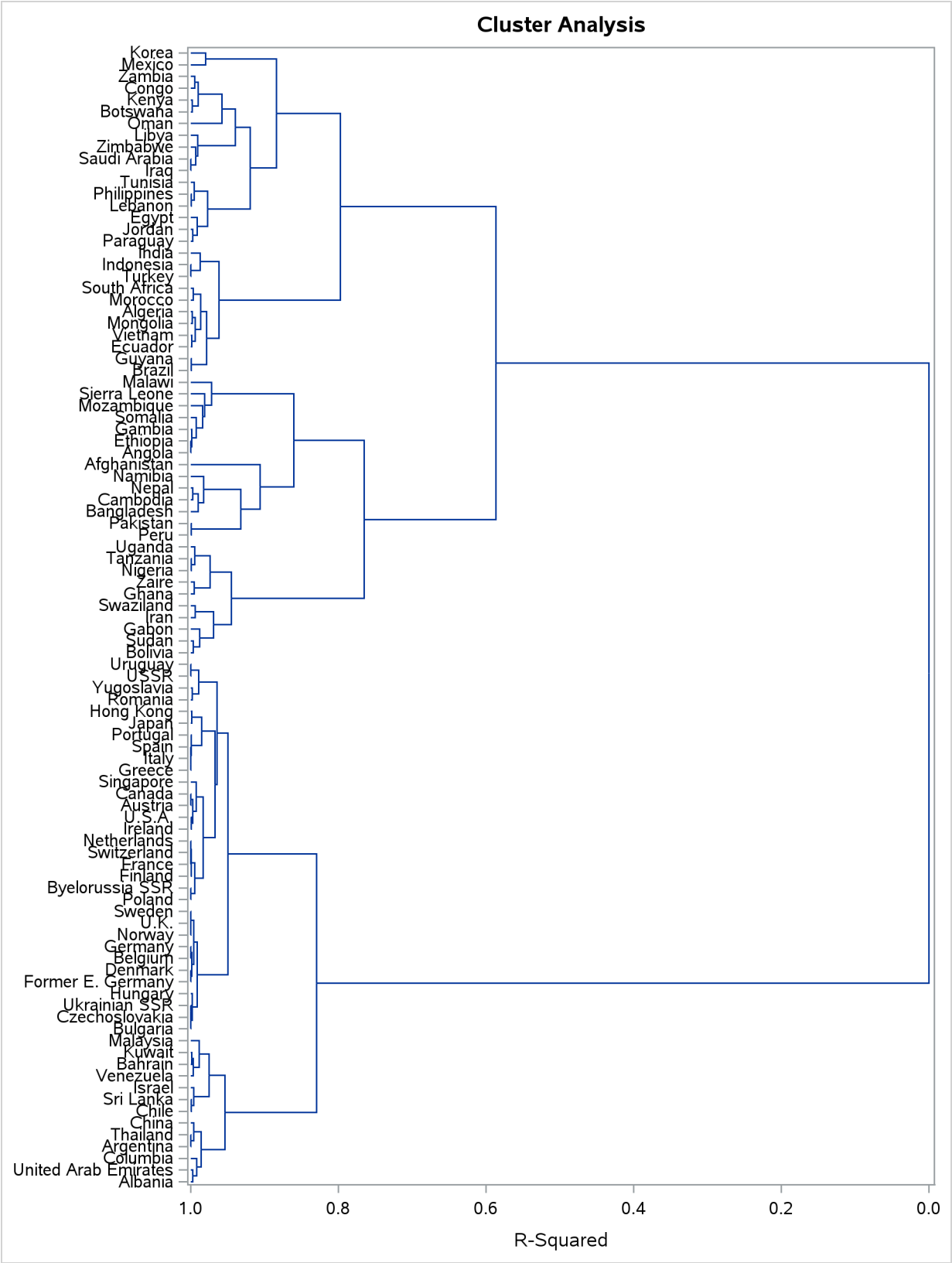

You can use PROC TREE and the output data set from PROC CLUSTER to create a new data set that contains information about cluster membership as follows:

```
proc tree data=Tree out=New nclusters=3 noprint;
   height _rsq_;
   copy can1 can2;
   id country;
run;
```

The SAS data set Tree is input. The OUT= option creates an output SAS data set named New that contains information about cluster membership. The NCLUSTERS= option specifies the number of clusters desired in the data set New. The results can be displayed in a scatter plot.

The following statements use the SGPLOT procedure to display the results that are in the SAS data set New:

```
proc sgplot data=New;
   scatter y=can2 x=can1 / group=cluster;
run;
```

The SCATTER statement requests a plot of the two canonical variables, using the value of the variable cluster, which is produced by PROC TREE as the identification variable. The results are displayed in Figure 40.5.

Figure 40.5 Plot of Canonical Variables and Cluster for Three Clusters

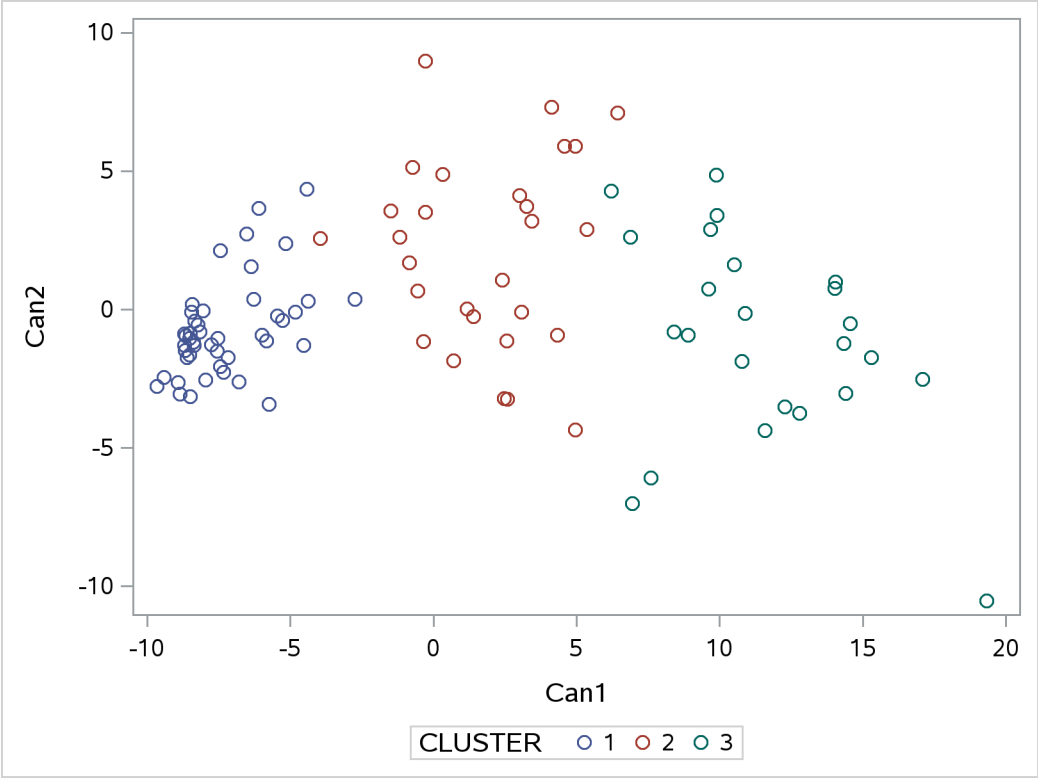

The statistics in Figure 40.2 and Figure 40.3, the dendrogram in Figure 40.4, and the plot of the canonical variables in Figure 40.5 assist in the estimation of the number of clusters in the data. There seems to be reasonable separation in the clusters. However, you must use this information, along with experience and knowledge of the field, to help in deciding the correct number of clusters.

Syntax: CLUSTER Procedure

The following statements are available in the CLUSTER procedure:

```
PROC CLUSTER METHOD=name < options > ;
    BY variables ;
    COPY variables ;
    FREQ variable ;
    ID variable ;
    RMSSTD variable ;
    VAR variables ;
```

Only the PROC CLUSTER statement is required, except that the FREQ statement is required when the RMSSTD statement is used; otherwise the FREQ statement is optional. Usually only the VAR statement and possibly the ID and COPY statements are needed in addition to the PROC CLUSTER statement. The rest of this section provides detailed syntax information for each of the preceding statements, beginning with the PROC CLUSTER statement. The remaining statements are covered in alphabetical order.

PROC CLUSTER Statement

```
PROC CLUSTER METHOD=name < options > ;
```

The PROC CLUSTER statement invokes the CLUSTER procedure. It also specifies a clustering method, and optionally specifies details for clustering methods, data sets, data processing, and displayed output.

Table 40.1 summarizes the *options* available in the PROC CLUSTER statement.

Table 40.1 PROC CLUSTER Statement Options

Option	Description
Specify Input and Output Data Sets	
DATA=	Specifies input data set
OUTTREE=	Creates output data set
Specify Clustering Methods	
BETA=	Specifies beta value for flexible beta method
HYBRID	Specifies Wong's hybrid clustering method
METHOD=	Specifies clustering method
MODE=	Specifies the minimum number of members for modal clusters
PENALTY=	Specifies the penalty coefficient for maximum likelihood
Control Data Processing Prior to Clustering	
NOEIGEN	Suppresses computation of eigenvalues
NONORM	Suppresses normalizing of distances
NOSQUARE	Suppresses squaring of distances
STANDARD	Standardizes variables
TRIM=	Omits points with low probability densities

Table 40.1 *continued*

Option	Description
Control Density Estimation	
K=	Specifies number of neighbors for *k*th-nearest-neighbor density estimation
R=	Specifies radius of sphere of support for uniform-kernel density estimation
Ties	
NOTIE	Suppresses checking for ties
Control Display of the Cluster History	
CCC	Displays cubic clustering criterion
NOID	Suppresses display of ID values
PRINT=	Specifies number of generations to display
PSEUDO	Displays pseudo F and t^2 statistics
RMSSTD	Displays root mean square standard deviation
RSQUARE	Displays R square and semipartial R square
Control Other Aspects of Output	
NOPRINT	Suppresses display of all output
PLOTS=	Specifies ODS graphics details
SIMPLE	Displays simple summary statistics

METHOD=*name*

The METHOD= specification determines the clustering method used by the procedure. Any one of the following 11 methods can be specified for *name*:

AVERAGE | AVE requests average linkage (group average, unweighted pair-group method using arithmetic averages, UPGMA). Distance data are squared unless you specify the NOSQUARE option.

CENTROID | CEN requests the centroid method (unweighted pair-group method using centroids, UPGMC, centroid sorting, weighted-group method). Distance data are squared unless you specify the NOSQUARE option.

COMPLETE | COM requests complete linkage (furthest neighbor, maximum method, diameter method, rank order typal analysis). To reduce distortion of clusters by outliers, the TRIM= option is recommended.

DENSITY | DEN requests density linkage, which is a class of clustering methods using nonparametric probability density estimation. You must also specify either the K=, R=, or HYBRID option to indicate the type of density estimation to be used. See also the MODE= and DIM= options in this section.

EML requests maximum-likelihood hierarchical clustering for mixtures of spherical multivariate normal distributions with equal variances but possibly unequal mixing proportions. Use METHOD=EML only with coordinate data. See the PENALTY= option for details. The NONORM option does not affect the reported likelihood values but does affect other unrelated criteria. The EML method is much slower than the other methods in the CLUSTER procedure.

FLEXIBLE | FLE requests the Lance-Williams flexible-beta method. See the BETA= option in this section.

MCQUITTY | MCQ requests McQuitty's similarity analysis (weighted average linkage, weighted pair-group method using arithmetic averages, WPGMA).

MEDIAN | MED requests Gower's median method (weighted pair-group method using centroids, WPGMC). Distance data are squared unless you specify the NOSQUARE option.

SINGLE | SIN requests single linkage (nearest neighbor, minimum method, connectedness method, elementary linkage analysis, or dendritic method). To reduce chaining, you can use the TRIM= option with METHOD=SINGLE.

TWOSTAGE | TWO requests two-stage density linkage. You must also specify the K=, R=, or HYBRID option to indicate the type of density estimation to be used. See also the MODE= and DIM= options in this section.

WARD | WAR requests Ward's minimum-variance method (error sum of squares, trace W). Distance data are squared unless you specify the NOSQUARE option. To reduce distortion by outliers, the TRIM= option is recommended. See the NONORM option.

The following list provides details about the other *options*.

BETA=*n*

specifies the beta parameter for METHOD=FLEXIBLE. The value of *n* should be less than 1, usually between 0 and –1. By default, BETA=–0.25. Milligan (1987) suggests a somewhat smaller value, perhaps –0.5, for data with many outliers.

CCC

displays the cubic clustering criterion and approximate expected R square under the uniform null hypothesis (Sarle 1983). The statistics associated with the RSQUARE option, R square and semipartial R square, are also displayed. The CCC option applies only to coordinate data. The CCC option is not appropriate with METHOD=SINGLE because of the method's tendency to chop off tails of distributions. Computation of the CCC requires the eigenvalues of the covariance matrix. If the number of variables is large, computing the eigenvalues requires much computer time and memory.

DATA=*SAS-data-set*

names the input data set that contains observations to be clustered. By default, the procedure uses the most recently created SAS data set. If the data set is TYPE=DISTANCE, the data are interpreted as a distance matrix; the number of variables must equal the number of observations in the data set or in each BY group. The distances are assumed to be Euclidean, but the procedure accepts other types of distances or dissimilarities. If the data set is not TYPE=DISTANCE, the data are interpreted as coordinates in a Euclidean space, and Euclidean distances are computed. For more about TYPE=DISTANCE data sets, see Chapter A, "Special SAS Data Sets."

Data set types (such as TYPE=DISTANCE) do not persist when you copy or modify a data set. You must specify the TYPE= data set option for the new data set, as in the following example:

```
data dist2(type=distance);
   set dist;
run;
```

If you do not specify the TYPE=DISTANCE data set option, the new data set is the default TYPE=DATA. If you use the new data set in a procedure that accepts both TYPE=DATA or TYPE=DISTANCE data sets (such as PROC CLUSTER or PROC MODECLUS), the results will be incorrect.

You cannot use a TYPE=CORR data set as input to PROC CLUSTER, since the procedure uses dissimilarity measures. Instead, you can use a DATA step or the IML procedure to extract the correlation matrix from a TYPE=CORR data set and transform the values to dissimilarities such as 1 – r or $1 - r^2$, where r is the correlation.

All methods produce the same results when used with coordinate data as when used with Euclidean distances computed from the coordinates. However, the DIM= option must be used with distance data if you specify METHOD=TWOSTAGE or METHOD=DENSITY or if you specify the TRIM= option.

Certain methods that are most naturally defined in terms of coordinates require *squared* Euclidean distances to be used in the combinatorial distance formulas (Lance and Williams 1967). For this reason, distance data are automatically squared when used with METHOD=AVERAGE, METHOD=CENTROID, METHOD=MEDIAN, or METHOD=WARD. If you want the combinatorial formulas to be applied to the (unsquared) distances with these methods, use the NOSQUARE option.

DIM=*n*

specifies the dimensionality used when computing density estimates with the TRIM= option, METHOD=DENSITY, or METHOD=TWOSTAGE. The values of n must be greater than or equal to 1. The default is the number of variables if the data are coordinates; the default is 1 if the data are distances.

HYBRID

requests the Wong (1982) hybrid clustering method in which density estimates are computed from a preliminary cluster analysis using the k-means method. The DATA= data set must contain means, frequencies, and root mean square standard deviations of the preliminary clusters (see the FREQ and RMSSTD statements). To use HYBRID, you must use either a FREQ statement or a DATA= data set that contains a _FREQ_ variable, and you must also use either an RMSSTD statement or a DATA= data set that contains an _RMSSTD_ variable.

The MEAN= data set produced by the FASTCLUS procedure is suitable for input to the CLUSTER procedure for hybrid clustering. Since this data set contains _FREQ_ and _RMSSTD_ variables, you can use it as input and then omit the FREQ and RMSSTD statements.

You must specify either METHOD=DENSITY or METHOD=TWOSTAGE with the HYBRID option. You cannot use this option in combination with the TRIM=, K=, or R= option.

K=*n*

specifies the number of neighbors to use for kth-nearest-neighbor density estimation (Silverman 1986, pp. 19–21 and 96–99). The number of neighbors (n) must be at least two but less than the number of observations. See the MODE= option, which follows.

Density estimation is used with the TRIM=, METHOD=DENSITY, and METHOD=TWOSTAGE options.

MODE=*n*

specifies that, when two clusters are joined, each must have at least *n* members in order for either cluster to be designated a modal cluster. If you specify MODE=1, each cluster must also have a maximum density greater than the fusion density in order for either cluster to be designated a modal cluster.

Use the MODE= option only with METHOD=DENSITY or METHOD=TWOSTAGE. With METHOD=TWOSTAGE, the MODE= option affects the number of modal clusters formed. With METHOD=DENSITY, the MODE= option does not affect the clustering process but does determine the number of modal clusters reported on the output and identified by the _MODE_ variable in the output data set.

If you specify the K= option, the default value of MODE= is the same as the value of K= because the use of *k*th-nearest-neighbor density estimation limits the resolution that can be obtained for clusters with fewer than *k* members. If you do not specify the K= option, the default is MODE=2.

If you specify MODE=0, the default value is used instead of 0.

If you specify a FREQ statement or if a _FREQ_ variable appears in the input data set, the MODE= value is compared with the number of actual observations in the clusters being joined, not with the sum of the frequencies in the clusters.

NOEIGEN

suppresses computation of the eigenvalues of the covariance matrix and substitutes the variances of the variables for the eigenvalues when computing the cubic clustering criterion. The NOEIGEN option saves time if the number of variables is large, but it should be used only if the variables are nearly uncorrelated. If you specify the NOEIGEN option and the variables are highly correlated, the cubic clustering criterion might be very liberal. The NOEIGEN option applies only to coordinate data.

NOID

suppresses the display of ID values for the clusters joined at each generation of the cluster history.

NONORM

prevents the distances from being normalized to unit mean or unit root mean square with most methods. With METHOD=WARD, the NONORM option prevents the between-cluster sum of squares from being normalized by the total sum of squares to yield a squared semipartial correlation. The NONORM option does not affect the reported likelihood values with METHOD=EML, but it does affect other unrelated criteria, such as the _DIST_ variable.

NOPRINT

suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

NOSQUARE

prevents input distances from being squared with METHOD=AVERAGE, METHOD=CENTROID, METHOD=MEDIAN, or METHOD=WARD.

If you specify the NOSQUARE option with distance data, the data are assumed to be squared Euclidean distances for computing R-square and related statistics defined in a Euclidean coordinate system.

If you specify the NOSQUARE option with coordinate data with METHOD=CENTROID, METHOD=MEDIAN, or METHOD=WARD, then the combinatorial formula is applied to unsquared

Euclidean distances. The resulting cluster distances do not have their usual Euclidean interpretation and are therefore labeled "False" in the output.

NOTIE

prevents PROC CLUSTER from checking for ties for minimum distance between clusters at each generation of the cluster history. If your data are measured with such precision that ties are unlikely, then you can specify the NOTIE option to reduce slightly the time and space required by the procedure. See the section "Ties" on page 2595 for more information.

OUTTREE=*SAS-data-set*

creates an output data set that can be used by the TREE procedure to draw a tree diagram. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. If you omit the OUTTREE= option, the data set is named by using the DATA*n* convention and is not permanently saved. If you do not want to create an output data set, use OUTTREE=_NULL_.

PENALTY=*p*

specifies the penalty coefficient used with METHOD=EML. See the section "Clustering Methods" on page 2585 for more information. Values for *p* must be greater than zero. By default, PENALTY=2.

PLOTS < (*global-plot-options***) > < = *plot-request* >**

PLOTS < (*global-plot-options***) > < = (*plot-request* < . . . *plot-request* >) >**

controls the plots produced through ODS Graphics.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc cluster method=ward plots=all;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

By default, PROC CLUSTER produces a dendrogram. PROC CLUSTER can also produce plots of the cubic clustering criterion, the pseudo F statistic, and the pseudo t^2 statistic from the cluster history table. These statistics are useful for estimating the number of clusters. Each statistic is plotted against the number of clusters. You can request that PROC CLUSTER create these graphs by specifying the CCC or PSEUDO options, or by specifying the statistics in a *plot-request* in the PLOT option. PROC CLUSTER might be unable to compute the statistics in some cases; for details, see the CCC and PSEUDO options. If a statistic cannot be computed, it cannot be plotted. PROC CLUSTER plots all of these statistics that are computed unless you tell it specifically what to plot using PLOTS=.

PROC CLUSTER has a CCC and PSEUDO option as well as CCC and PSEUDO *plot-requests*. All four options are illustrated in the following step:

```
ods graphics on;

proc cluster ccc pseudo plots=(ccc pseudo);
run;

ods graphics off;
```

The maximum number of clusters shown in all the plots is the minimum of the following quantities:

- the number of observations
- the value of the PRINT= option, if that option is specified
- the maximum number of clusters for which CCC is computed, if the CCC is plotted

The *global-plot-options* apply to all plots generated by the CLUSTER procedure. The *global-plot-options* are as follows:

MAXCLUS=*n*
right-truncates the CCC, PSF, and PST2 plots at the *n* value. This prevents these plots from losing resolution when a large number of clusters are plotted. The default is MAXCLUS=200.

MAXPOINTS=*n*
MAXPTS=*n*
suppresses the dendrogram when the number of clusters exceeds the *n* value. This prevents an unreadable plot from being produced. The default is MAXPOINTS=200.

UNPACKPANEL
UNPACK
breaks a plot that is otherwise paneled into separate plots for each statistic.

ONLY
suppresses the default plots. Only plots specifically requested are displayed.

The following *plot-requests* can be specified:

ALL
generates all possible plots. The CCC and PSEUDO options must be specified to obtain the CCC, PSF and PST2 plots in addition to the dendrogram.

CCC
implicitly specifies the CCC option and, if possible, plots the cubic clustering criterion against the number of clusters.

DENDROGRAM <(*dendrogram-options* **)>**
requests a dendrogram and specifies *dendrogram-options*. A dendrogram is created by default unless the ONLY *global-plot-option* is requested.

Unlike most graphs, the size of the dendrogram can vary as a function of the number of objects that appear in the dendrogram. You can specify the following *dendrogram-options* to control the size and appearance of the dendrogram:

COMPUTEHEIGHT=*a b*

CH=*a b*

specifies the constants for computing the height of the dendrogram. For n points being clustered, intercept a, and slope b, the height is based in part on $a + bn$. For a horizontal dendrogram, the default (given in pixels) is COMPUTEHEIGHT=100 12, the default height in pixels is max($100 + 12n$, 480), the default height in inches is max($1.04167 + 0.125n$, 5), and the default height in centimeters is max($2.64583 + 0.3175n$, 12.7). For a vertical dendrogram, the default height is 480 pixels. The default unit is pixels, and you can use the UNIT= *dendrogram-option* to change the unit to inches or centimeters for this option. Inches equals pixels divided by 96, and centimeters equals inches times 2.54.

COMPUTEWIDTH=*a b*

CW=*a b*

specifies the constants for computing the width of the dendrogram. For n points being clustered, intercept a, and slope b, the width is based in part on $a + bn$. For a vertical dendrogram, the default (given in pixels) is COMPUTEWIDTH=100 12, the default width in pixels is max($100 + 12n$, 640), the default width in inches is max($1.04167 + 0.125n$, 6.66667), and the default width in centimeters is max($2.64583 + 0.3175n$, 16.933). For a horizontal dendrogram, the default width is 640 pixels. The default unit is pixels, and you can use the UNIT= *dendrogram-option* to change the unit to inches or centimeters for this option. Inches equals pixels divided by 96, and centimeters equals inches times 2.54.

HEIGHT=HEIGHT | MODE | NCL | RSQ

H=H | M | N | R

specifies the method for drawing the height of the dendrogram. HEIGHT=HEIGHT is the default.

HEIGHT=HEIGHT specifies the distance or similarity between the last clusters joined, as defined in the section "Clustering Methods" on page 2585.

HEIGHT=MODE pertains to the modal clusters. With METHOD=DENSITY, the mode indicates the number of modal clusters contained by the current cluster. With METHOD=TWOSTAGE, the mode gives the maximum density in each modal cluster and the fusion density, d^*, for clusters containing two or more modal clusters; for clusters that contain no modal clusters, that value of the _MODE_ variable is missing.

HEIGHT=NCL specifies that the number of clusters is used.

HEIGHT=RSQ specifies that the squared multiple correlation is used.

HORIZONTAL | VERTICAL

specifies either a horizontal dendrogram with the objects on the vertical axis (HORIZONTAL) or a vertical dendrogram with the objects on the horizontal axis (VERTICAL). The default is HORIZONTAL.

SETHEIGHT=*height*

SH=*height*

specifies the height of the dendrogram. By default, the height is based on the COMPUTEHEIGHT= option. The default unit is pixels, and you can use the UNIT= *dendrogram-option* to change the unit to inches or centimeters for this *dendrogram-option*.

SETWIDTH=*width*

SW=*width*

specifies the width of the dendrogram. By default, the width is based on the COMPUTEWIDTH= option. The default unit is pixels, and you can use the UNIT= *dendrogram-option* to change the unit to inches or centimeters for this *dendrogram-option*.

UNIT=PX | IN | CM

specifies the unit (pixels, inches, or centimeters) for the SETHEIGHT=, SETWIDTH=, COMPUTEHEIGHT=, and COMPUTEWIDTH= *dendrogram-options*.

NONE

suppresses all plots.

PSEUDO

implicitly specifies the PSEUDO option and, if possible, plots the pseudo F statistic and the pseudo t^2 statistic against the number of clusters.

PSF

implicitly specifies the PSEUDO option and, if possible, plots the pseudo F statistic against the number of clusters.

PST2

implicitly specifies the PSEUDO option and, if possible, plots the pseudo t^2 statistic against the number of clusters.

You can specify one or more of the *plot-requests* in the same PLOT option. For example, all of the following are valid:

```
proc cluster plots=(ccc pst2);
proc cluster plots=(psf);
proc cluster plots=psf;
```

The first statement plots both the cubic clustering criterion and the pseudo t^2 statistic, while the second and third statements plot the pseudo F statistic only. When you specify only one plot request, you can omit the parentheses around the plot request. When you specify more than one *plot-request*, you must specify parentheses. Otherwise the second and subsequent *plot-requests* are *options*. Since CCC and PSEUDO are both *options* as well as *plot-requests*, the following three statements are valid, but they are not equivalent:

```
proc cluster plots(only)=ccc pseudo;
proc cluster plots(only)=pseudo ccc;
proc cluster plots(only)=(ccc pseudo);
```

The first two examples have one *plot-request* and one procedure *option*. The third example has two *plot-requests*.

The names of the graphs that PROC CLUSTER generates are listed in Table 40.5, along with the required statements and options.

PRINT=*n* | **P=***n*

specifies the number of generations of the cluster history to display. The PRINT= option displays the latest *n* generations; for example, PRINT=5 displays the cluster history from one cluster through five clusters. The value of PRINT= must be a nonnegative integer. The default is to display all generations. Specify PRINT=0 to suppress the cluster history.

PSEUDO

displays pseudo F and t^2 statistics. This option is effective only when the data are coordinates or when METHOD=AVERAGE, METHOD=CENTROID, or METHOD=WARD is specified. See the section "Miscellaneous Formulas" on page 2592 for more information. The PSEUDO option is not appropriate with METHOD=SINGLE because of the method's tendency to chop off tails of distributions.

R=*n*

specifies the radius of the sphere of support for uniform-kernel density estimation (Silverman 1986, pp. 11–13 and 75–94).

The value of R= must be greater than zero.

Density estimation is used with the TRIM=, METHOD=DENSITY, and METHOD=TWOSTAGE options.

RMSSTD

displays the root mean square standard deviation of each cluster. This option is effective only when the data are coordinates or when METHOD=AVERAGE, METHOD=CENTROID, or METHOD=WARD is specified.

See the section "Miscellaneous Formulas" on page 2592 for more information.

RSQUARE | RSQ

displays the R square and semipartial R square. This option is effective only when the data are coordinates or when METHOD=AVERAGE or METHOD=CENTROID is specified. The R square and semipartial R square statistics are always displayed with METHOD=WARD. See the section "Miscellaneous Formulas" on page 2592 for more information..

SIMPLE | S

displays means, standard deviations, skewness, kurtosis, and a coefficient of bimodality. The SIMPLE option applies only to coordinate data. See the section "Miscellaneous Formulas" on page 2592 for more information.

STANDARD | STD

standardizes the variables to mean 0 and standard deviation 1. The STANDARD option applies only to coordinate data.

TRIM=*p*

omits points with low estimated probability densities from the analysis. Valid values for the TRIM= option are $0 \leq p < 100$. If $p < 1$, then p is the proportion of observations omitted. If $p \geq 1$, then p is interpreted as a percentage. A specification of TRIM=10, which trims 10% of the points, is a reasonable value for many data sets. Densities are estimated by the *k*th-nearest-neighbor or uniform-kernel method. Trimmed points are indicated by a negative value of the _FREQ_ variable in the OUTTREE= data set.

You must use either the K= or R= option when you use TRIM=. You cannot use the HYBRID option in combination with TRIM=, so you might want to use the DIM= option instead. If you specify the

STANDARD option in combination with TRIM=, the variables are standardized both before and after trimming.

The TRIM= option is useful for removing outliers and reducing chaining. Trimming is highly recommended with METHOD=WARD or METHOD=COMPLETE because clusters from these methods can be severely distorted by outliers. Trimming is also valuable with METHOD=SINGLE since single linkage is the method most susceptible to chaining. Most other methods also benefit from trimming. However, trimming is unnecessary with METHOD=TWOSTAGE or METHOD=DENSITY when kth-nearest-neighbor density estimation is used.

Use of the TRIM= option can spuriously inflate the cubic clustering criterion and the pseudo F and t^2 statistics. Trimming only outliers improves the accuracy of the statistics, but trimming saddle regions between clusters yields excessively large values.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC CLUSTER to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the CLUSTER procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the “Grouping Data” section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

COPY Statement

COPY *variables* **;**

The variables in the COPY statement are copied from the input data set to the OUTTREE= data set. Observations in the OUTTREE= data set that represent clusters of more than one observation from the input data set have missing values for the COPY variables.

FREQ Statement

FREQ *variable* **;**

If one variable in the input data set represents the frequency of occurrence for other values in the observation, specify the variable's name in a FREQ statement. PROC CLUSTER then treats the data set as if each observation appeared *n* times, where *n* is the value of the FREQ variable for the observation. Noninteger values of the FREQ variable are truncated to the largest integer less than the FREQ value.

If you omit the FREQ statement but the DATA= data set contains a variable called _FREQ_, then frequencies are obtained from the _FREQ_ variable. If neither a FREQ statement nor an _FREQ_ variable is present, each observation is assumed to have a frequency of one.

If each observation in the DATA= data set represents a cluster (for example, clusters formed by PROC FASTCLUS), the variable specified in the FREQ statement should give the number of original observations in each cluster.

If you specify the RMSSTD statement, a FREQ statement is required. A FREQ statement or _FREQ_ variable is required when you specify the HYBRID option.

With most clustering methods, the same clusters are obtained from a data set with a FREQ variable as from a similar data set without a FREQ variable, if each observation is repeated as many times as the value of the FREQ variable in the first data set. The FLEXIBLE method can yield different results due to the nature of the combinatorial formula. The DENSITY and TWOSTAGE methods are also exceptions because two identical observations can be absorbed one at a time by a cluster with a higher density. If you are using a FREQ statement with either the DENSITY or TWOSTAGE method, see the MODE=option for details.

ID Statement

ID *variable* **;**

The values of the ID variable identify observations in the displayed cluster history and in the OUTTREE= data set. If the ID statement is omitted, each observation is denoted by OB*n*, where *n* is the observation number.

RMSSTD Statement

RMSSTD *variable* **;**

If the coordinates in the DATA= data set represent cluster means (for example, formed by the FASTCLUS procedure), you can obtain accurate statistics in the cluster histories for METHOD=AVERAGE, METHOD=CENTROID, or METHOD=WARD if the data set contains both of the following:

- a variable giving the number of original observations in each cluster (see the discussion of the FREQ statement earlier in this chapter)
- a variable giving the root mean squared standard deviation of each cluster

Specify the name of the variable containing root mean squared standard deviations in the RMSSTD statement. If you specify the RMSSTD statement, you must also specify a FREQ statement.

If you omit the RMSSTD statement but the DATA= data set contains a variable called _RMSSTD_, then the root mean squared standard deviations are obtained from the _RMSSTD_ variable.

An RMSSTD statement or _RMSSTD_ variable is required when you specify the HYBRID option.

A data set created by PROC FASTCLUS, using the MEAN= option, contains _FREQ_ and _RMSSTD_ variables, so you do not have to use FREQ and RMSSTD statements when using such a data set as input to the CLUSTER procedure.

VAR Statement

VAR *variables* **;**

The VAR statement lists numeric variables to be used in the cluster analysis. If you omit the VAR statement, all numeric variables not listed in other statements are used.

Details: CLUSTER Procedure

Clustering Methods

The following notation is used, with lowercase symbols generally pertaining to observations and uppercase symbols pertaining to clusters:

n	number of observations
v	number of variables if data are coordinates
G	number of clusters at any given level of the hierarchy
x_i or $\mathbf{x}_i$	ith observation (row vector if coordinate data)
C_K	Kth cluster, subset of $\{1, 2, \ldots, n\}$
N_K	number of observations in C_K
$\bar{\mathbf{x}}$	sample mean vector
$\bar{\mathbf{x}}_K$	mean vector for cluster C_K
$\lVert\mathbf{x}\rVert$	Euclidean length of the vector $\mathbf{x}$—that is, the square root of the sum of the squares of the elements of $\mathbf{x}$
T	$\sum_{i=1}^{n} \lVert\mathbf{x}_i - \bar{\mathbf{x}}\rVert^2$
W_K	$\sum_{i \in C_k} \lVert\mathbf{x}_i - \bar{\mathbf{x}}_K\rVert^2$
P_G	$\sum W_J$, where summation is over the G clusters at the Gth level of the hierarchy
B_{KL}	$W_M - W_K - W_L$ if $C_M = C_K \cup C_L$
$d(\mathbf{x}, \mathbf{y})$	any distance or dissimilarity measure between observations or vectors $\mathbf{x}$ and $\mathbf{y}$
D_{KL}	any distance or dissimilarity measure between clusters C_K and C_L

The distance between two clusters can be defined either directly or combinatorially (Lance and Williams 1967)—that is, by an equation for updating a distance matrix when two clusters are joined. In all of the following combinatorial formulas, it is assumed that clusters C_K and C_L are merged to form C_M, and the formula gives the distance between the new cluster C_M and any other cluster C_J.

For an introduction to most of the methods used in the CLUSTER procedure, see Massart and Kaufman (1983).

Average Linkage

The following method is obtained by specifying METHOD=AVERAGE. The distance between two clusters is defined by

$$D_{KL} = \frac{1}{N_K N_L} \sum_{i \in C_K} \sum_{j \in C_L} d(x_i, x_j)$$

If $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|^2$, then

$$D_{KL} = \|\bar{\mathbf{x}}_K - \bar{\mathbf{x}}_L\|^2 + \frac{W_K}{N_K} + \frac{W_L}{N_L}$$

The combinatorial formula is

$$D_{JM} = \frac{N_K D_{JK} + N_L D_{JL}}{N_M}$$

In average linkage the distance between two clusters is the average distance between pairs of observations, one in each cluster. Average linkage tends to join clusters with small variances, and it is slightly biased toward producing clusters with the same variance.

Average linkage was originated by Sokal and Michener (1958).

Centroid Method

The following method is obtained by specifying METHOD=CENTROID. The distance between two clusters is defined by

$$D_{KL} = \|\bar{\mathbf{x}}_K - \bar{\mathbf{x}}_L\|^2$$

If $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|^2$, then the combinatorial formula is

$$D_{JM} = \frac{N_K D_{JK} + N_L D_{JL}}{N_M} - \frac{N_K N_L D_{KL}}{N_M^2}$$

In the centroid method, the distance between two clusters is defined as the (squared) Euclidean distance between their centroids or means. The centroid method is more robust to outliers than most other hierarchical methods but in other respects might not perform as well as Ward's method or average linkage (Milligan 1980).

The centroid method was originated by Sokal and Michener (1958).

Complete Linkage

The following method is obtained by specifying METHOD=COMPLETE. The distance between two clusters is defined by

$$D_{KL} = \max_{i \in C_K} \max_{j \in C_L} d(x_i, x_j)$$

The combinatorial formula is

$$D_{JM} = \max(D_{JK}, D_{JL})$$

In complete linkage, the distance between two clusters is the maximum distance between an observation in one cluster and an observation in the other cluster. Complete linkage is strongly biased toward producing clusters with roughly equal diameters, and it can be severely distorted by moderate outliers (Milligan 1980).

Complete linkage was originated by Sørensen (1948).

Density Linkage

The phrase *density linkage* is used here to refer to a class of clustering methods that use nonparametric probability density estimates (for example, Hartigan 1975, pp. 205–212; Wong 1982; Wong and Lane 1983). Density linkage consists of two steps:

1. A new dissimilarity measure, d^*, based on density estimates and adjacencies is computed. If x_i and x_j are adjacent (the definition of *adjacency* depends on the method of density estimation), then $d^*(x_i, x_j)$ is the reciprocal of an estimate of the density midway between x_i and x_j; otherwise, $d^*(x_i, x_j)$ is infinite.

2. A single linkage cluster analysis is performed using d^*.

The CLUSTER procedure supports three types of density linkage: the *k*th-nearest-neighbor method, the uniform-kernel method, and Wong's hybrid method. These are obtained by using METHOD=DENSITY and the K=, R=, and HYBRID options, respectively.

kth-Nearest-Neighbor Method

The *k*th-nearest-neighbor method (Wong and Lane 1983) uses *k*th-nearest-neighbor density estimates. Let $r_k(x)$ be the distance from point x to the *k*th-nearest observation, where k is the value specified for the K= option. Consider a closed sphere centered at x with radius $r_k(x)$. The estimated density at x, $f(x)$, is the proportion of observations within the sphere divided by the volume of the sphere. The new dissimilarity measure is computed as

$$d^*(x_i, x_j) = \begin{cases} \frac{1}{2}\left(\frac{1}{f(x_i)} + \frac{1}{f(x_j)}\right) & \text{if } d(x_i, x_j) \le \max(r_k(x_i), r_k(x_j)) \\ \infty & \text{otherwise} \end{cases}$$

Wong and Lane (1983) show that *k*th-nearest-neighbor density linkage is strongly set consistent for high-density (density-contour) clusters if k is chosen such that $k/n \to 0$ and $k/\ln(n) \to \infty$ as $n \to \infty$. Wong and Schaack (1982) discuss methods for estimating the number of population clusters by using *k*th-nearest-neighbor clustering.

Uniform-Kernel Method

The uniform-kernel method uses uniform-kernel density estimates. Let r be the value specified for the R= option. Consider a closed sphere centered at point x with radius r. The estimated density at x, $f(x)$, is the proportion of observations within the sphere divided by the volume of the sphere. The new dissimilarity measure is computed as

$$d^*(x_i, x_j) = \begin{cases} \frac{1}{2}\left(\frac{1}{f(x_i)} + \frac{1}{f(x_j)}\right) & \text{if } d(x_i, x_j) \le r \\ \infty & \text{otherwise} \end{cases}$$

Wong's Hybrid Method

The Wong (1982) hybrid clustering method uses density estimates based on a preliminary cluster analysis by the *k*-means method. The preliminary clustering can be done by the FASTCLUS procedure, by using the MEAN= option to create a data set containing cluster means, frequencies, and root mean squared standard deviations. This data set is used as input to the CLUSTER procedure, and the HYBRID option is specified

with METHOD=DENSITY to request the hybrid analysis. The hybrid method is appropriate for very large data sets but should not be used with small data sets—say, than those with fewer than 100 observations in the original data. The term *preliminary cluster* refers to an observation in the DATA= data set.

For preliminary cluster C_K, N_K and W_K are obtained from the input data set, as are the cluster means or the distances between the cluster means. Preliminary clusters C_K and C_L are considered adjacent if the midpoint between $\bar{x}_K$ and $\bar{\mathbf{x}}_L$ is closer to either $\bar{\mathbf{x}}_K$ or $\bar{\mathbf{x}}_L$ than to any other preliminary cluster mean or, equivalently, if $d^2(\bar{\mathbf{x}}_K, \bar{\mathbf{x}}_L) < d^2(\bar{\mathbf{x}}_K, \bar{\mathbf{x}}_M) + d^2(\bar{\mathbf{x}}_L, \bar{\mathbf{x}}_M)$ for all other preliminary clusters C_M, $M \neq K$ or L. The new dissimilarity measure is computed as

$$d^*(\bar{\mathbf{x}}_K, \bar{\mathbf{x}}_L) = \begin{cases} \frac{\left(W_K + W_L + \frac{1}{4}(N_K + N_L) d^2(\bar{\mathbf{x}}_K, \bar{\mathbf{x}}_L)\right)^{\frac{v}{2}}}{(N_K + N_L)^{1+\frac{v}{2}}} & \text{if } C_K \text{ and } C_L \text{ are adjacent} \\ \infty & \text{otherwise} \end{cases}$$

Using the K= and R= Options

The values of the K= and R= options are called *smoothing parameters*. Small values of K= or R= produce jagged density estimates and, as a consequence, many modes. Large values of K= or R= produce smoother density estimates and fewer modes. In the hybrid method, the smoothing parameter is the number of clusters in the preliminary cluster analysis. The number of modes in the final analysis tends to increase as the number of clusters in the preliminary analysis increases. Wong (1982) suggests using $n^{0.3}$ preliminary clusters, where n is the number of observations in the original data set. There is no rule of thumb for selecting K= values. For all types of density linkage, you should repeat the analysis with several different values of the smoothing parameter (Wong and Schaack 1982).

There is no simple answer to the question of which smoothing parameter to use (Silverman 1986, pp. 43–61, 84–88, and 98–99). It is usually necessary to try several different smoothing parameters. A reasonable first guess for the R= option in many coordinate data sets is given by

$$\left[\frac{2^{v+2}(v+2)\Gamma(\frac{v}{2}+1)}{nv^2}\right]^{\frac{1}{v+4}} \sqrt{\sum_{l=1}^{v} s_l^2}$$

where s_l^2 is the standard deviation of the lth variable. The estimate for R= can be computed in a DATA step by using the GAMMA function for Γ. This formula is derived under the assumption that the data are sampled from a multivariate normal distribution and tends, therefore, to be too large (oversmooth) if the true distribution is multimodal. Robust estimates of the standard deviations can be preferable if there are outliers. If the data are distances, the factor $\sum s_l^2$ can be replaced by an average (mean, trimmed mean, median, root mean square, and so on) distance divided by $\sqrt{2}$. To prevent outliers from appearing as separate clusters, you can also specify K=2, or more generally K=m, $m \geq 2$, which in most cases forces clusters to have at least m members.

If the variables all have unit variance (for example, if the STANDARD option is used), Table 40.2 can be used to obtain an initial guess for the R= option.

Since infinite d^* values occur in density linkage, the final number of clusters can exceed one when there are wide gaps between the clusters or when the smoothing parameter results in little smoothing.

Density linkage applies no constraints to the shapes of the clusters and, unlike most other hierarchical clustering methods, is capable of recovering clusters with elongated or irregular shapes. Since density linkage uses less prior knowledge about the shape of the clusters than do methods restricted to compact clusters,

density linkage is less effective at recovering compact clusters from small samples than are methods that always recover compact clusters, regardless of the data.

Table 40.2 Reasonable First Guess for the R= Option for Standardized Data

Number of Observations	Number of Variables									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
20	1.01	1.36	1.77	2.23	2.73	3.25	3.81	4.38	4.98	5.60
35	0.91	1.24	1.64	2.08	2.56	3.08	3.62	4.18	4.77	5.38
50	0.84	1.17	1.56	1.99	2.46	2.97	3.50	4.06	4.64	5.24
75	0.78	1.09	1.47	1.89	2.35	2.85	3.38	3.93	4.50	5.09
100	0.73	1.04	1.41	1.82	2.28	2.77	3.29	3.83	4.40	4.99
150	0.68	0.97	1.33	1.73	2.18	2.66	3.17	3.71	4.27	4.85
200	0.64	0.93	1.28	1.67	2.11	2.58	3.09	3.62	4.17	4.75
350	0.57	0.85	1.18	1.56	1.98	2.44	2.93	3.45	4.00	4.56
500	0.53	0.80	1.12	1.49	1.91	2.36	2.84	3.35	3.89	4.45
750	0.49	0.74	1.06	1.42	1.82	2.26	2.74	3.24	3.77	4.32
1000	0.46	0.71	1.01	1.37	1.77	2.20	2.67	3.16	3.69	4.23
1500	0.43	0.66	0.96	1.30	1.69	2.11	2.57	3.06	3.57	4.11
2000	0.40	0.63	0.92	1.25	1.63	2.05	2.50	2.99	3.49	4.03

EML

The following method is obtained by specifying METHOD=EML. The distance between two clusters is given by

$$D_{KL} = nv \ln\left(1 + \frac{B_{KL}}{P_G}\right) - 2\left(N_M \ln(N_M) - N_K \ln(N_K) - N_L \ln(N_L)\right)$$

The EML method joins clusters to maximize the likelihood at each level of the hierarchy under the following assumptions:

- multivariate normal mixture
- equal spherical covariance matrices
- unequal sampling probabilities

The EML method is similar to Ward's minimum-variance method but removes the bias toward equal-sized clusters. Practical experience has indicated that EML is somewhat biased toward unequal-sized clusters. You can specify the PENALTY= option to adjust the degree of bias. If you specify PENALTY=*p*, the formula is modified to

$$D_{KL} = nv \ln\left(1 + \frac{B_{KL}}{P_G}\right) - p\left(N_M \ln(N_M) - N_K \ln(N_K) - N_L \ln(N_L)\right)$$

The EML method was derived by W. S. Sarle of SAS Institute from the maximum likelihood formula obtained by Symons (1981, p. 37, Equation 8) for disjoint clustering. There are currently no other published references on the EML method.

Flexible-Beta Method

The following method is obtained by specifying METHOD=FLEXIBLE. The combinatorial formula is

$$D_{JM} = (D_{JK} + D_{JL})\frac{1-b}{2} + D_{KL}b$$

where b is the value of the BETA= option, or –0.25 by default.

The flexible-beta method was developed by Lance and Williams (1967); see also Milligan (1987).

McQuitty's Similarity Analysis

The following method is obtained by specifying METHOD=MCQUITTY. The combinatorial formula is

$$D_{JM} = \frac{D_{JK} + D_{JL}}{2}$$

The method was independently developed by Sokal and Michener (1958) and McQuitty (1966).

Median Method

The following method is obtained by specifying METHOD=MEDIAN. If $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|^2$, then the combinatorial formula is

$$D_{JM} = \frac{D_{JK} + D_{JL}}{2} - \frac{D_{KL}}{4}$$

The median method was developed by Gower (1967).

Single Linkage

The following method is obtained by specifying METHOD=SINGLE. The distance between two clusters is defined by

$$D_{KL} = \min_{i \in C_K} \min_{j \in C_L} d(x_i, x_j)$$

The combinatorial formula is

$$D_{JM} = \min(D_{JK}, D_{JL})$$

In single linkage, the distance between two clusters is the minimum distance between an observation in one cluster and an observation in the other cluster. Single linkage has many desirable theoretical properties (Jardine and Sibson 1971; Fisher and Van Ness 1971; Hartigan 1981) but has fared poorly in Monte Carlo studies (for example, Milligan 1980). By imposing no constraints on the shape of clusters, single linkage sacrifices performance in the recovery of compact clusters in return for the ability to detect elongated and irregular clusters. You must also recognize that single linkage tends to chop off the tails of distributions before separating the main clusters (Hartigan 1981). The notorious chaining tendency of single linkage can be alleviated by specifying the TRIM= option (Wishart 1969, pp. 296–298).

Density linkage and two-stage density linkage retain most of the virtues of single linkage while performing better with compact clusters and possessing better asymptotic properties (Wong and Lane 1983).

Single linkage was originated by Florek et al. (1951b, a) and later reinvented by McQuitty (1957) and Sneath (1957).

Two-Stage Density Linkage

If you specify METHOD=DENSITY, the modal clusters often merge before all the points in the tails have clustered. The option METHOD=TWOSTAGE is a modification of density linkage that ensures that all points are assigned to modal clusters before the modal clusters are permitted to join. The CLUSTER procedure supports the same three varieties of two-stage density linkage as of ordinary density linkage: kth-nearest neighbor, uniform kernel, and hybrid.

In the first stage, disjoint modal clusters are formed. The algorithm is the same as the single linkage algorithm ordinarily used with density linkage, with one exception: two clusters are joined only if at least one of the two clusters has fewer members than the number specified by the MODE= option. At the end of the first stage, each point belongs to one modal cluster.

In the second stage, the modal clusters are hierarchically joined by single linkage. The final number of clusters can exceed one when there are wide gaps between the clusters or when the smoothing parameter is small.

Each stage forms a tree that can be plotted by the TREE procedure. By default, the TREE procedure plots the tree from the first stage. To obtain the tree for the second stage, use the option HEIGHT=MODE in the PROC TREE statement. You can also produce a single tree diagram containing both stages, with the number of clusters as the height axis, by using the option HEIGHT=N in the PROC TREE statement. To produce an output data set from PROC TREE containing the modal clusters, use _HEIGHT_ for the HEIGHT variable (the default) and specify LEVEL=0.

Two-stage density linkage was developed by W. S. Sarle of SAS Institute. There are currently no other published references on two-stage density linkage.

Ward's Minimum-Variance Method

The following method is obtained by specifying METHOD=WARD. The distance between two clusters is defined by

$$D_{KL} = B_{KL} = \frac{\|\bar{\mathbf{x}}_K - \bar{\mathbf{x}}_L\|^2}{\frac{1}{N_K} + \frac{1}{N_L}}$$

If $d(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2$, then the combinatorial formula is

$$D_{JM} = \frac{(N_J + N_K)D_{JK} + (N_J + N_L)D_{JL} - N_J D_{KL}}{N_J + N_M}$$

In Ward's minimum-variance method, the distance between two clusters is the ANOVA sum of squares between the two clusters added up over all the variables. At each generation, the within-cluster sum of squares is minimized over all partitions obtainable by merging two clusters from the previous generation. The sums of squares are easier to interpret when they are divided by the total sum of squares to give proportions of variance (squared semipartial correlations).

Ward's method joins clusters to maximize the likelihood at each level of the hierarchy under the following assumptions:

- multivariate normal mixture
- equal spherical covariance matrices
- equal sampling probabilities

Ward's method tends to join clusters with a small number of observations, and it is strongly biased toward producing clusters with roughly the same number of observations. It is also very sensitive to outliers (Milligan 1980).

Ward (1963) describes a class of hierarchical clustering methods including the minimum variance method.

Miscellaneous Formulas

The root mean squared standard deviation of a cluster C_K is

$$\text{RMSSTD} = \sqrt{\frac{W_K}{v(N_K - 1)}}$$

The R-square statistic for a given level of the hierarchy is

$$R^2 = 1 - \frac{P_G}{T}$$

The squared semipartial correlation for joining clusters C_K and C_L is

$$\text{semipartial } R^2 = \frac{B_{KL}}{T}$$

The bimodality coefficient is

$$b = \frac{m_3^2 + 1}{m_4 + \frac{3(n-1)^2}{(n-2)(n-3)}}$$

where m_3 is skewness and m_4 is kurtosis. Values of b greater than 0.555 (the value for a uniform population) can indicate bimodal or multimodal marginal distributions. The maximum of 1.0 (obtained for the Bernoulli distribution) is obtained for a population with only two distinct values. Very heavy-tailed distributions have small values of b regardless of the number of modes.

Formulas for the cubic-clustering criterion and approximate expected R square are given in Sarle (1983).

The pseudo F statistic for a given level is

$$\text{pseudo } F = \frac{\frac{T-P_G}{G-1}}{\frac{P_G}{n-G}}$$

The pseudo t^2 statistic for joining C_K and C_L is

$$\text{pseudo } t^2 = \frac{B_{KL}}{\frac{W_K+W_L}{N_K+N_L-2}}$$

The pseudo F and t^2 statistics can be useful indicators of the number of clusters, but they are *not* distributed as F and t^2 random variables. If the data are independently sampled from a multivariate normal distribution with a scalar covariance matrix and if the clustering method allocates observations to clusters randomly (which no clustering method actually does), then the pseudo F statistic is distributed as an F random variable with $v(G-1)$ and $v(n-G)$ degrees of freedom. Under the same assumptions, the pseudo t^2 statistic is distributed as an F random variable with v and $v(N_K + N_L - 2)$ degrees of freedom. The pseudo t^2 statistic differs computationally from Hotelling's T^2 in that the latter uses a general symmetric covariance matrix instead of a scalar covariance matrix. The pseudo F statistic was suggested by Caliński and Harabasz (1974). The pseudo t^2 statistic is related to the $J_e(2)/J_e(1)$ statistic of Duda and Hart (1973) by

$$\frac{J_e(2)}{J_e(1)} = \frac{W_K + W_L}{W_M} = \frac{1}{1 + \frac{t^2}{N_K + N_L - 2}}$$

See Milligan and Cooper (1985) and Cooper and Milligan (1988) regarding the performance of these statistics in estimating the number of population clusters. Conservative tests for the number of clusters using the pseudo F and t^2 statistics can be obtained by the Bonferroni approach (Hawkins, Muller, and ten Krooden 1982, pp. 337–340).

Ultrametrics

A dissimilarity measure $d(x, y)$ is called an *ultrametric* if it satisfies the following conditions:

- $d(x, x) = 0$ for all x
- $d(x, y) \geq 0$ for all x, y
- $d(x, y) = d(y, x)$ for all x, y
- $d(x, y) \leq \max(d(x, z), d(y, z))$ for all x, y, and z

Any hierarchical clustering method induces a dissimilarity measure on the observations—say, $h(x_i, x_j)$. Let C_M be the cluster with the fewest members that contains both x_i and x_j. Assume C_M was formed by joining C_K and C_L. Then define $h(x_i, x_j) = D_{KL}$.

If the fusion of C_K and C_L reduces the number of clusters from g to $g-1$, then define $D_{(g)} = D_{KL}$. Johnson (1967) shows that if

$$0 \leq D_{(n)} \leq D_{(n-1)} \leq \cdots \leq D_{(2)}$$

then $h(\cdot, \cdot)$ is an ultrametric. A method that always satisfies this condition is said to be a *monotonic* or *ultrametric clustering method*. All methods implemented in PROC CLUSTER except CENTROID, EML, and MEDIAN are ultrametric (Milligan 1979; Batagelj 1981).

Algorithms

Anderberg (1973) describes three algorithms for implementing agglomerative hierarchical clustering: stored data, stored distance, and sorted distance. The algorithms used by PROC CLUSTER for each method are indicated in Table 40.3. For METHOD=AVERAGE, METHOD=CENTROID, or METHOD=WARD, either the stored data or the stored distance algorithm can be used. For these methods, if the data are distances or if you specify the NOSQUARE option, the stored distance algorithm is used; otherwise, the stored data algorithm is used.

Table 40.3 Three Algorithms for Implementing Agglomerative Hierarchical Clustering

	Algorithm		
Clustering Method	**Stored Data**	**Stored Distance**	**Sorted Distance**
AVERAGE	x	x	
CENTROID	x	x	
COMPLETE		x	
DENSITY			x
EML	x		
FLEXIBLE		x	
MCQUITTY		x	
MEDIAN		x	
SINGLE		x	
TWOSTAGE			x
WARD	x	x	

Note: All of the hierarchical methods accept coordinate data. Methods that require stored or sorted distances automatically calculate distances from the coordinates.

Computational Resources

The CLUSTER procedure stores the data (including the COPY and ID variables) in memory or, if necessary, on disk. If eigenvalues are computed, the covariance matrix is stored in memory. If the stored distance or sorted distance algorithm is used, the distances are stored in memory or, if necessary, on disk.

With coordinate data, the increase in CPU time is roughly proportional to the number of variables. The VAR statement should list the variables in order of decreasing variance for greatest efficiency.

For both coordinate and distance data, the dominant factor determining CPU time is the number of observations. For density methods with coordinate data, the asymptotic time requirements are somewhere between $n \ln(n)$ and n^2, depending on how the smoothing parameter increases. For other methods except EML, time is roughly proportional to n^2. For the EML method, time is roughly proportional to n^3.

PROC CLUSTER runs much faster if the data can be stored in memory and, when the stored distance algorithm is used, if the distance matrix can be stored in memory as well. To estimate the bytes of memory needed for the data, use the following formula and round up to the nearest multiple of d.

$n(vd$	$+$	$8d + i$	
	$+$	i	if density estimation or the sorted distance algorithm is used
	$+$	$3d$	if stored data algorithm is used
	$+$	$3d$	if density estimation is used
	$+$	max(8, length of ID variable)	if ID variable is used
	$+$	length of ID variable	if ID variable is used
	$+$	sum of lengths of COPY variables)	if COPY variables is used

where

n is the number of observations

v is the number of variables

d is the size of a C variable of type *double*. For most computers, $d = 8$.

i is the size of a C variable of type *int*. For most computers, $i = 4$.

The number of bytes needed for the distance matrix is $dn(n + 1)/2$.

Missing Values

If the data are coordinates, observations with missing values are excluded from the analysis. If the data are distances, missing values are not permitted in the lower triangle of the distance matrix. The upper triangle is ignored. For more about TYPE=DISTANCE data sets, see Chapter A, "Special SAS Data Sets."

Ties

At each level of the clustering algorithm, PROC CLUSTER must identify the pair of clusters with the minimum distance. Sometimes, usually when the data are discrete, there can be two or more pairs with the same minimum distance. In such cases the tie must be broken in some arbitrary way. If there are ties, then the results of the cluster analysis depend on the order of the observations in the data set. The presence of ties is reported in the SAS log and in the column of the cluster history labeled "Tie" unless the NOTIE option is specified.

PROC CLUSTER breaks ties as follows. Each cluster is identified by the smallest observation number among its members. For each pair of clusters, there is a smaller identification number and a larger identification number. If two or more pairs of clusters are tied for minimum distance between clusters, the pair that has the minimum larger identification number is merged. If there is a tie for minimum larger identification number, the pair that has the minimum smaller identification number is merged.

A tie means that the level in the cluster history at which the tie occurred and possibly some of the subsequent levels are not uniquely determined. Ties that occur early in the cluster history usually have little effect on the later stages. Ties that occur in the middle part of the cluster history are cause for further investigation. Ties that occur late in the cluster history indicate important indeterminacies.

The importance of ties can be assessed by repeating the cluster analysis for several different random permutations of the observations. The discrepancies at a given level can be examined by crosstabulating the clusters obtained at that level for all of the permutations. See Example 40.4 for details.

Size, Shape, and Correlation

In some biological applications, the organisms that are being clustered can be at different stages of growth. Unless it is the growth process itself that is being studied, differences in size among such organisms are not of interest. Therefore, distances among organisms should be computed in such a way as to control for differences in size while retaining information about differences in shape.

If coordinate data are measured on an interval scale, you can control for size by subtracting a measure of the overall size of each observation from each data item. For example, if no other direct measure of size is available, you could subtract the mean of each row of the data matrix, producing a row-centered coordinate matrix. An easy way to subtract the mean of each row is to use PROC STANDARD on the transposed coordinate matrix:

```
proc transpose data= coordinate-datatype;
run;

proc standard m=0;
run;

proc transpose out=row-centered-coordinate-data;
run;
```

Another way to remove size effects from interval-scale coordinate data is to do a principal component analysis and discard the first component (Blackith and Reyment 1971).

If the data are measured on a ratio scale, you can control for size by dividing each observation by a measure of overall size; in this case, the geometric mean is a more natural measure of size than the arithmetic mean. However, it is often more meaningful to analyze the logarithms of ratio-scaled data, in which case you can subtract the arithmetic mean after taking logarithms. You must also consider the dimensions of measurement. For example, if you have measures of both length and weight, you might need to cube the measures of length or take the cube root of the weights. Various other complications can also arise in real applications, such as different growth rates for different parts of the body (Sneath and Sokal 1973).

Issues of size and shape are pertinent to many areas besides biology (for example, Hamer and Cunningham 1981). Suppose you have data consisting of subjective ratings made by several different raters. Some raters tend to give higher overall ratings than other raters. Some raters also tend to spread out their ratings over more of the scale than other raters. If it is impossible for you to adjust directly for rater differences, then distances should be computed in such a way as to control for differences both in size and variability. For example, if the data are considered to be measured on an interval scale, you can subtract the mean of each observation and divide by the standard deviation, producing a row-standardized coordinate matrix. With some clustering methods, analyzing squared Euclidean distances from a row-standardized coordinate matrix is equivalent to analyzing the matrix of correlations among rows, since squared Euclidean distance is an affine transformation of the correlation (Hartigan 1975, p. 64).

If you do an analysis of row-centered or row-standardized data, you need to consider whether the columns (variables) should be standardized before centering or standardizing the rows, after centering or standardizing

the rows, or both before and after. If you standardize the columns after standardizing the rows, then strictly speaking you are not analyzing shape because the profiles are distorted by standardizing the columns; however, this type of double standardization might be necessary in practice to get reasonable results. It is not clear whether iterating the standardization of rows and columns can be of any benefit.

The choice of distance or correlation measure should depend on the meaning of the data and the purpose of the analysis. Simulation studies that compare distance and correlation measures are useless unless the data are generated to mimic data from your field of application. Conclusions drawn from artificial data cannot be generalized, because it is possible to generate data such that distances that include size effects work better or such that correlations work better.

You can standardize the rows of a data set by using a DATA step or by using the TRANSPOSE and STANDARD procedures. You can also use PROC TRANSPOSE and then have PROC CORR create a TYPE=CORR data set containing a correlation matrix. If you want to analyze a TYPE=CORR data set with PROC CLUSTER, you must use a DATA step to perform the following steps:

1. Set the data set TYPE= to DISTANCE.
2. Convert the correlations to dissimilarities by computing $1-r$, $\sqrt{1-r}$, $1-r^2$, or some other decreasing function.
3. Delete observations for which the variable _TYPE_ does not have the value 'CORR'.

Output Data Set

The OUTTREE= data set contains one observation for each observation in the input data set, plus one observation for each cluster of two or more observations (that is, one observation for each node of the cluster tree). The total number of output observations is usually $2n-1$, where n is the number of input observations. The density methods can produce fewer output observations when the number of clusters cannot be reduced to one.

The label of the OUTTREE= data set identifies the type of cluster analysis performed and is automatically displayed when the TREE procedure is invoked.

The variables in the OUTTREE= data set are as follows:

- the BY variables, if you use a BY statement
- the ID variable, if you use an ID statement
- the COPY variables, if you use a COPY statement
- _NAME_, a character variable giving the name of the node. If the node is a cluster, the name is CLn, where n is the number of the cluster. If the node is an observation, the name is OBn, where n is the observation number. If the node is an observation and the ID statement is used, the name is the formatted value of the ID variable.
- _PARENT_, a character variable giving the value of _NAME_ of the parent of the node
- _NCL_, the number of clusters

- _FREQ_, the number of observations in the current cluster
- _HEIGHT_, the distance or similarity between the last clusters joined, as defined in the section "Clustering Methods" on page 2585. The variable _HEIGHT_ is used by the TREE procedure as the default height axis. The label of the _HEIGHT_ variable identifies the between-cluster distance measure. For METHOD=TWOSTAGE, the _HEIGHT_ variable contains the densities at which clusters joined in the first stage; for clusters formed in the second stage, _HEIGHT_ is a very small negative number.

If the input data set contains coordinates, the following variables appear in the output data set:

- the variables containing the coordinates used in the cluster analysis. For output observations that correspond to input observations, the values of the coordinates are the same in both data sets except for some slight numeric error possibly introduced by standardizing and unstandardizing if the STANDARD option is used. For output observations that correspond to clusters of more than one input observation, the values of the coordinates are the cluster means.
- _ERSQ_, the approximate expected value of R square under the uniform null hypothesis
- _RATIO_, equal to (1 – _ERSQ_) / (1 – _RSQ_)
- _LOGR_, natural logarithm of _RATIO_
- _CCC_, the cubic clustering criterion

The variables _ERSQ_, _RATIO_, _LOGR_, and _CCC_ have missing values when the number of clusters is greater than one-fifth the number of observations.

If the input data set contains coordinates and METHOD=AVERAGE, METHOD=CENTROID, or METHOD=WARD, then the following variables appear in the output data set:

- _DIST_, the Euclidean distance between the means of the last clusters joined
- _AVLINK_, the average distance between the last clusters joined

If the input data set contains coordinates or METHOD=AVERAGE, METHOD=CENTROID, or METHOD=WARD, then the following variables appear in the output data set:

- _RMSSTD_, the root mean squared standard deviation of the current cluster
- _SPRSQ_, the semipartial squared multiple correlation or the decrease in the proportion of variance accounted for due to joining two clusters to form the current cluster
- _RSQ_, the squared multiple correlation
- _PSF_, the pseudo F statistic
- _PST2_, the pseudo t^2 statistic

If METHOD=EML, then the following variable appears in the output data set:

- _LNLR_, the log-likelihood ratio

If METHOD=TWOSTAGE or METHOD=DENSITY, the following variable appears in the output data set:

- _MODE_, pertaining to the modal clusters. With METHOD=DENSITY, the _MODE_ variable indicates the number of modal clusters contained by the current cluster. With METHOD=TWOSTAGE, the _MODE_ variable gives the maximum density in each modal cluster and the fusion density, d^*, for clusters containing two or more modal clusters; for clusters containing no modal clusters, _MODE_ is missing.

If nonparametric density estimates are requested (when METHOD=DENSITY or METHOD=TWOSTAGE and the HYBRID option is not used; or when any of the TRIM=, K= or R= options are used), the output data set contains the following:

- _DENS_, the maximum density in the current cluster

Displayed Output

If you specify the SIMPLE option and the data are coordinates, PROC CLUSTER produces simple descriptive statistics for each variable:

- the Mean
- the standard deviation, Std Dev
- the Skewness
- the Kurtosis
- a coefficient of Bimodality

If the data are coordinates and you do not specify the NOEIGEN option, PROC CLUSTER displays the following:

- the Eigenvalues of the Correlation or Covariance Matrix
- the Difference between successive eigenvalues
- the Proportion of variance explained by each eigenvalue
- the Cumulative proportion of variance explained

If the data are coordinates, PROC CLUSTER displays the Root Mean Squared Total-Sample Standard Deviation of the variables

If the distances are normalized, PROC CLUSTER displays one of the following, depending on whether squared or unsquared distances are used:

- the Root Mean Squared Distance Between Observations
- the Mean Distance Between Observations

For the generations in the clustering process specified by the PRINT= option, PROC CLUSTER displays the following:

- the Number of Clusters or NCL
- the names of the Clusters Joined. The observations are identified by the formatted value of the ID variable, if any; otherwise, the observations are identified by OB*n*, where *n* is the observation number. The CLUSTER procedure displays the entire value of the ID variable in the cluster history instead of truncating at 16 characters. Long ID values might be split onto several lines. Clusters of two or more observations are identified as CL*n*, where *n* is the number of clusters existing after the cluster in question is formed.
- the number of observations in the new cluster, Frequency of New Cluster or FREQ

If you specify the RMSSTD option and the data are coordinates, or if you specify METHOD=AVERAGE, METHOD=CENTROID, or METHOD=WARD, then PROC CLUSTER displays the root mean squared standard deviation of the new cluster, RMS Std of New Cluster or RMS Std.

PROC CLUSTER displays the following items if you specify METHOD=WARD. It also displays them if you specify the RSQUARE option and either the data are coordinates or you specify METHOD=AVERAGE or METHOD=CENTROID.

- the decrease in the proportion of variance accounted for resulting from joining the two clusters, Semipartial R-Square or SPRSQ. This equals the between-cluster sum of squares divided by the corrected total sum of squares.
- the squared multiple correlation, R-Square or RSQ. R square is the proportion of variance accounted for by the clusters.

If you specify the CCC option and the data are coordinates, PROC CLUSTER displays the following:

- Approximate Expected R-Square or ERSQ, the approximate expected value of R square under the uniform null hypothesis
- the Cubic Clustering Criterion or CCC. The cubic clustering criterion and approximate expected R square are given missing values when the number of clusters is greater than one-fifth the number of observations.

If you specify the PSEUDO option and the data are coordinates, or if you specify METHOD=AVERAGE, METHOD=CENTROID, or METHOD=WARD, then PROC CLUSTER displays the following:

- Pseudo F or PSF, the pseudo F statistic measuring the separation among all the clusters at the current level

- Pseudo t^2 or PST2, the pseudo t^2 statistic measuring the separation between the two clusters most recently joined

If you specify the NOSQUARE option and METHOD=AVERAGE, PROC CLUSTER displays the (Normalized) Average Distance or (Norm) Aver Dist, the average distance between pairs of objects in the two clusters joined with one object from each cluster.

If you do not specify the NOSQUARE option and METHOD=AVERAGE, PROC CLUSTER displays the (Normalized) RMS Distance or (Norm) RMS Dist, the root mean squared distance between pairs of objects in the two clusters joined with one object from each cluster.

If METHOD=CENTROID, PROC CLUSTER displays the (Normalized) Centroid Distance or (Norm) Cent Dist, the distance between the two cluster centroids.

If METHOD=COMPLETE, PROC CLUSTER displays the (Normalized) Maximum Distance or (Norm) Max Dist, the maximum distance between the two clusters.

If METHOD=DENSITY or METHOD=TWOSTAGE, PROC CLUSTER displays the following:

- Normalized Fusion Density or Normalized Fusion Dens, the value of d^* as defined in the section "Clustering Methods" on page 2585
- the Normalized Maximum Density in Each Cluster joined, including the Lesser or Min, and the Greater or Max, of the two maximum density values

If METHOD=EML, PROC CLUSTER displays the following:

- Log Likelihood Ratio or LNLR
- Log Likelihood or LNLIKE

If METHOD=FLEXIBLE, PROC CLUSTER displays the (Normalized) Flexible Distance or (Norm) Flex Dist, the distance between the two clusters based on the Lance-Williams flexible formula.

If METHOD=MEDIAN, PROC CLUSTER displays the (Normalized) Median Distance or (Norm) Med Dist, the distance between the two clusters based on the median method.

If METHOD=MCQUITTY, PROC CLUSTER displays the (Normalized) McQuitty's Similarity or (Norm) MCQ, the distance between the two clusters based on McQuitty's similarity method.

If METHOD=SINGLE, PROC CLUSTER displays the (Normalized) Minimum Distance or (Norm) Min Dist, the minimum distance between the two clusters.

If you specify the NONORM option and METHOD=WARD, PROC CLUSTER displays the Between-Cluster Sum of Squares or BSS, the ANOVA sum of squares between the two clusters joined.

If you specify neither the NOTIE option nor METHOD=TWOSTAGE or METHOD=DENSITY, PROC CLUSTER displays Tie, where a T in the column indicates a tie for minimum distance and a blank indicates the absence of a tie.

After the cluster history, if METHOD=TWOSTAGE or METHOD=DENSITY, PROC CLUSTER displays the number of modal clusters.

ODS Table Names

PROC CLUSTER assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 40.4. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 40.4 ODS Tables Produced by PROC CLUSTER

ODS Table Name	Description	Statement	Option
ClusterHistory	Observation or clusters joined, frequencies and other cluster statistics	PROC CLUSTER	Default
SimpleStatistics	Simple statistics, before or after trimming	PROC CLUSTER	SIMPLE
EigenvalueTable	Eigenvalues of the CORR or COV matrix	PROC CLUSTER	Default
RMSStd	Root mean square total sample standard deviation	PROC CLUSTER	Default
AvDist	Root mean square distance between observations	PROC CLUSTER	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC CLUSTER can produce plots of the cubic clustering criterion, pseudo F, and pseudo t^2 statistics, and a dendrogram. To plot a statistic, you must ask for it to be computed via one or more of the CCC, PSEUDO, or PLOT options.

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC CLUSTER generates are listed in Table 40.5, along with the required statements and options.

Table 40.5 Graphs Produced by PROC CLUSTER

ODS Graph Name	Plot Description	Statement and Option
CubicClusCritPlot	Cubic clustering criterion for the number of clusters	PROC CLUSTER PLOTS=CCC
PseudoFPlot	Pseudo F criterion for the number of clusters	PROC CLUSTER PLOTS=PSF
PseudoTSqPlot	Pseudo t^2 criterion for the number of clusters	PROC CLUSTER PLOTS=PST2
CccAndPsTSqPlot	Cubic clustering criterion and pseudo t^2	PROC CLUSTER PLOTS=(CCC PST2)
CccAndPsfPlot	Cubic clustering criterion and pseudo F	PROC CLUSTER PLOTS=(CCC PSF)
CccPsfAndPsTSqPlot	Cubic clustering criterion, pseudo F, and pseudo t^2	PROC CLUSTER PLOTS=ALL
Dendrogram	Dendrogram (tree diagram)	PROC CLUSTER PLOTS=DENDROGRAM

Examples: CLUSTER Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 40.1: Cluster Analysis of Flying Mileages between 10 American Cities

(View the complete code for this example (clusex1.sas) in the example repository.)

This example clusters 10 American cities based on the flying mileages between them. Six clustering methods are shown with corresponding dendrograms. The EML method cannot be used because it requires coordinate data. The other omitted methods produce the same clusters, although not the same distances between clusters, as one of the illustrated methods: complete linkage and the flexible-beta method yield the same clusters as Ward's method, McQuitty's similarity analysis produces the same clusters as average linkage, and the median method corresponds to the centroid method.

All of the methods suggest a division of the cities into two clusters along the east-west dimension. There is disagreement, however, about which cluster Denver should belong to. Some of the methods indicate a possible third cluster that contains Denver and Houston.

The following step displays the city mileage SAS data set, which is available in the Sashelp library and is designated as a TYPE=DISTANCE data set when it is used by PROC CLUSTER:

```
proc print noobs data=sashelp.mileages;
run;
```

Output 40.1.1 City Mileage Data Set

Atlanta	Chicago	Denver	Houston	LosAngeles	Miami	NewYork	SanFrancisco	Seattle	WashingtonDC	City
0	.	.	.	.	.	.	.	.	.	Atlanta
587	0	.	.	.	.	.	.	.	.	Chicago
1212	920	0	.	.	.	.	.	.	.	Denver
701	940	879	0	.	.	.	.	.	.	Houston
1936	1745	831	1374	0	.	.	.	.	.	Los Angeles
604	1188	1726	968	2339	0	.	.	.	.	Miami
748	713	1631	1420	2451	1092	0	.	.	.	New York
2139	1858	949	1645	347	2594	2571	0	.	.	San Francisco
2182	1737	1021	1891	959	2734	2408	678	0	.	Seattle
543	597	1494	1220	2300	923	205	2442	2329	0	Washington D.C.

A partial listing from the following statements include Output 40.1.2 and Output 40.1.3:

```
title 'Cluster Analysis of Flying Mileages Between 10 American Cities';
ods graphics on;

title2 'Using METHOD=AVERAGE';
proc cluster data=sashelp.mileages(type=distance) method=average pseudo;
   id City;
run;
```

Output 40.1.2 Cluster History Using METHOD=AVERAGE

Cluster Analysis of Flying Mileages Between 10 American Cities
Using METHOD=AVERAGE

The CLUSTER Procedure
Average Linkage Cluster Analysis

Cluster History							
Number of Clusters	Clusters Joined		Freq	Pseudo F Statistic	Pseudo t-Squared	Norm RMS Distance	Tie
9	New York	Washington D.C.	2	66.7	.	0.1297	
8	Los Angeles	San Francisco	2	39.2	.	0.2196	
7	Atlanta	Chicago	2	21.7	.	0.3715	
6	CL7	CL9	4	14.5	3.4	0.4149	
5	CL8	Seattle	3	12.4	7.3	0.5255	
4	Denver	Houston	2	13.9	.	0.5562	
3	CL6	Miami	5	15.5	3.8	0.6185	
2	CL3	CL4	7	16.0	5.3	0.8005	
1	CL2	CL5	10	.	16.0	1.2967	

Output 40.1.3 Dendrogram Using METHOD=AVERAGE

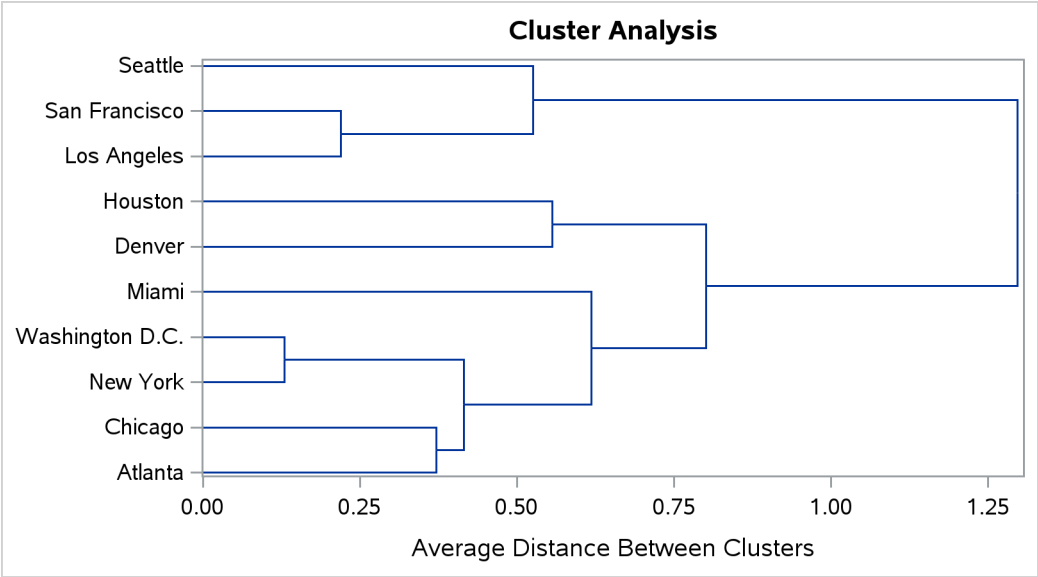

A partial listing from the following statements include Output 40.1.4 and Output 40.1.5:

```
title2 'Using METHOD=CENTROID';
proc cluster data=sashelp.mileages(type=distance) method=centroid pseudo;
   id City;
run;
```

Output 40.1.4 Cluster History Using METHOD=CENTROID

Cluster Analysis of Flying Mileages Between 10 American Cities Using METHOD=CENTROID

The CLUSTER Procedure
Centroid Hierarchical Cluster Analysis

Cluster History

Number of Clusters	Clusters Joined		Freq	Pseudo F Statistic	Pseudo t-Squared	Norm Centroid Distance	Tie
9	New York	Washington D.C.	2	66.7	.	0.1297	
8	Los Angeles	San Francisco	2	39.2	.	0.2196	
7	Atlanta	Chicago	2	21.7	.	0.3715	
6	CL7	CL9	4	14.5	3.4	0.3652	
5	CL8	Seattle	3	12.4	7.3	0.5139	
4	Denver	CL5	4	12.4	2.1	0.5337	
3	CL6	Miami	5	14.2	3.8	0.5743	
2	CL3	Houston	6	22.1	2.6	0.6091	
1	CL2	CL4	10	.	22.1	1.173	

Output 40.1.5 Dendrogram Using METHOD=CENTROID

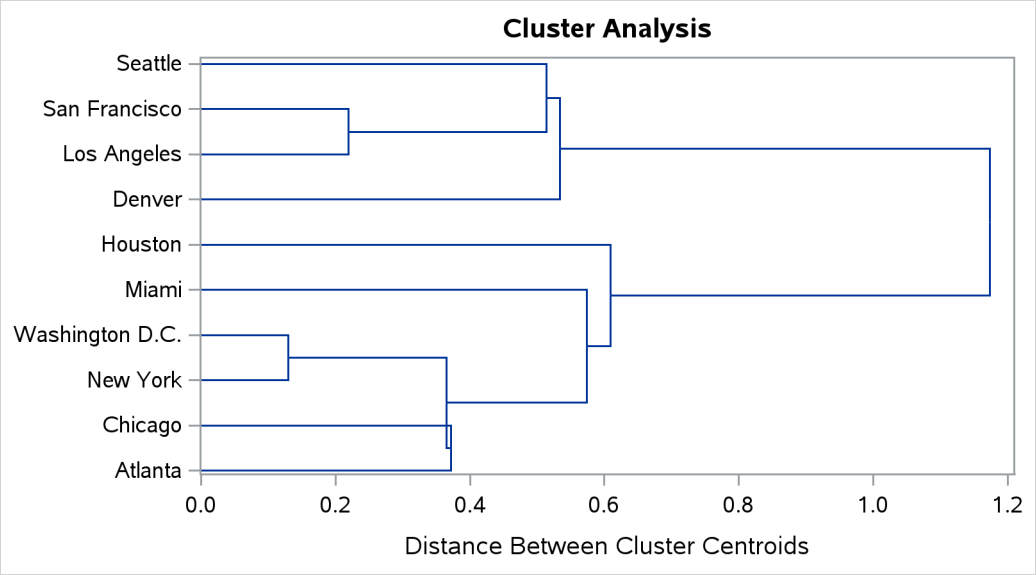

A partial listing from the following statements include Output 40.1.6 and Output 40.1.7:

```
title2 'Using METHOD=DENSITY K=3';
proc cluster data=sashelp.mileages(type=distance) method=density k=3;
   id City;
run;
```

Output 40.1.6 Cluster History Using METHOD=DENSITY K=3

Cluster Analysis of Flying Mileages Between 10 American Cities Using METHOD=DENSITY K=3

The CLUSTER Procedure
Density Linkage Cluster Analysis

Cluster History							
					Maximum Density in Each Cluster		
Number of Clusters	Clusters Joined		Freq	Normalized Fusion Density	Lesser	Greater	Tie
9	Atlanta	Washington D.C.	2	96.106	92.5043	100.0	
8	CL9	Chicago	3	95.263	90.9548	100.0	
7	CL8	New York	4	86.465	76.1571	100.0	
6	CL7	Houston	5	74.079	61.7747	100.0	T
5	CL6	Miami	6	74.079	58.8299	100.0	
4	Los Angeles	San Francisco	2	71.968	65.3430	80.0885	
3	CL4	Seattle	3	66.341	56.6215	80.0885	
2	CL3	Denver	4	63.509	61.7747	80.0885	
1	CL5	CL2	10	61.775 *	80.0885	100.0	

Output 40.1.7 Dendrogram Using METHOD=DENSITY K=3

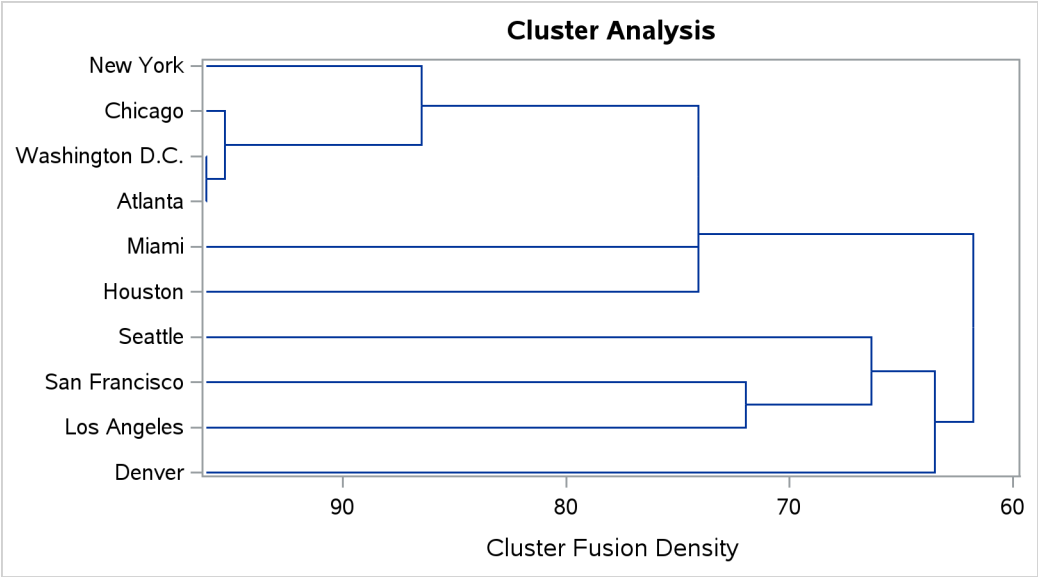

A partial listing from the following statements include Output 40.1.8 and Output 40.1.9:

```
title2 'Using METHOD=SINGLE';
proc cluster data=sashelp.mileages(type=distance) method=single;
   id City;
run;
```

Output 40.1.8 Cluster History Using METHOD=SINGLE

Cluster Analysis of Flying Mileages Between 10 American Cities Using METHOD=SINGLE

The CLUSTER Procedure
Single Linkage Cluster Analysis

Cluster History				
Number of Clusters	Clusters Joined		Freq	Norm Minimum Distance	Tie
9	New York	Washington D.C.	2	0.1447	
8	Los Angeles	San Francisco	2	0.2449	
7	Atlanta	CL9	3	0.3832	
6	CL7	Chicago	4	0.4142	
5	CL6	Miami	5	0.4262	
4	CL8	Seattle	3	0.4784	
3	CL5	Houston	6	0.4947	
2	Denver	CL4	4	0.5864	
1	CL3	CL2	10	0.6203	

Output 40.1.9 Dendrogram Using METHOD=SINGLE

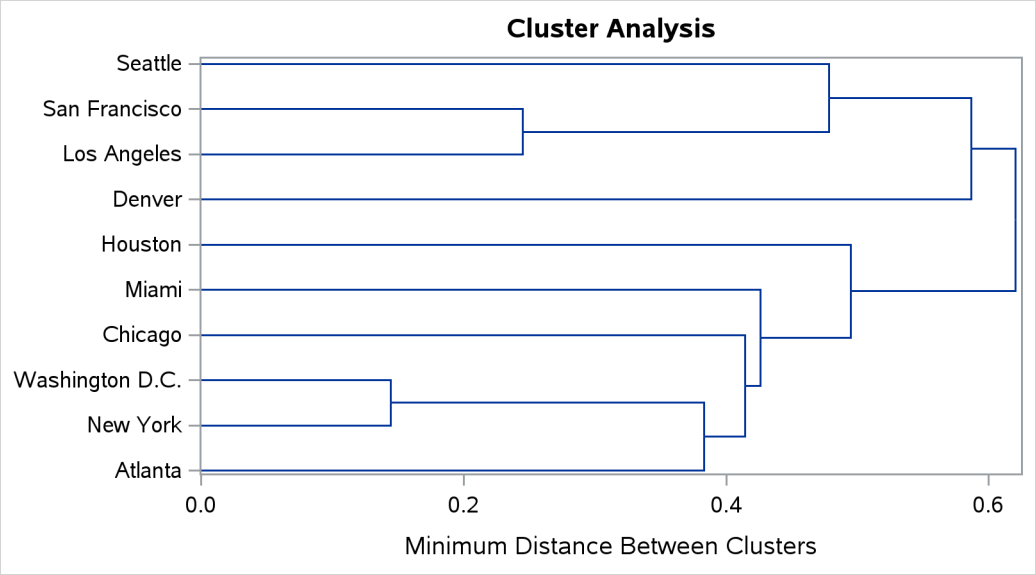

A partial listing from the following statements include Output 40.1.10 and Output 40.1.11:

```
title2 'Using METHOD=TWOSTAGE K=3';
proc cluster data=sashelp.mileages(type=distance) method=twostage k=3;
   id City;
run;
```

Output 40.1.10 Cluster History Using METHOD=TWOSTAGE K=3

Cluster Analysis of Flying Mileages Between 10 American Cities Using METHOD=TWOSTAGE K=3

The CLUSTER Procedure
Two-Stage Density Linkage Clustering

Cluster History							
					Maximum Density in Each Cluster		
Number of Clusters	Clusters Joined		Freq	Normalized Fusion Density	Lesser	Greater	Tie
9	Atlanta	Washington D.C.	2	96.106	92.5043	100.0	
8	CL9	Chicago	3	95.263	90.9548	100.0	
7	CL8	New York	4	86.465	76.1571	100.0	
6	CL7	Houston	5	74.079	61.7747	100.0	T
5	CL6	Miami	6	74.079	58.8299	100.0	
4	Los Angeles	San Francisco	2	71.968	65.3430	80.0885	
3	CL4	Seattle	3	66.341	56.6215	80.0885	
2	CL3	Denver	4	63.509	61.7747	80.0885	
1	CL5	CL2	10	61.775	80.0885	100.0	

Output 40.1.11 Dendrogram Using METHOD=TWOSTAGE K=3

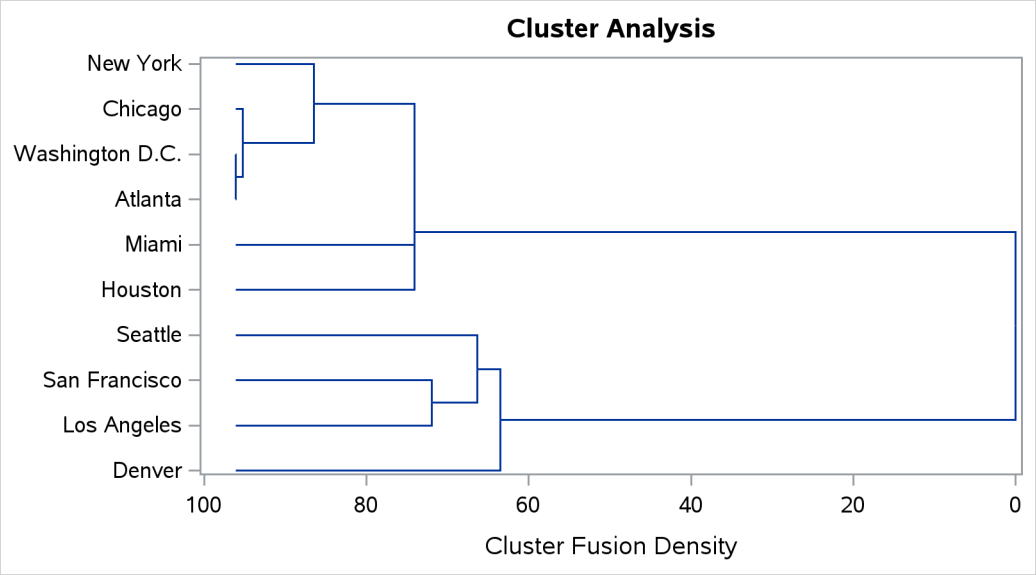

A partial listing from the following statements include Output 40.1.12 and Output 40.1.13:

```
title2 'Using METHOD=WARD';
proc cluster data=sashelp.mileages(type=distance) method=ward pseudo;
   id City;
run;
```

Output 40.1.12 Cluster History Using METHOD=WARD

Cluster Analysis of Flying Mileages Between 10 American Cities Using METHOD=WARD

The CLUSTER Procedure
Ward's Minimum Variance Cluster Analysis

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Pseudo F Statistic	Pseudo t-Squared	Tie
9	New York	Washington D.C.	2	0.0019	.998	66.7	.	
8	Los Angeles	San Francisco	2	0.0054	.993	39.2	.	
7	Atlanta	Chicago	2	0.0153	.977	21.7	.	
6	CL7	CL9	4	0.0296	.948	14.5	3.4	
5	Denver	Houston	2	0.0344	.913	13.2	.	
4	CL8	Seattle	3	0.0391	.874	13.9	7.3	
3	CL6	Miami	5	0.0586	.816	15.5	3.8	
2	CL3	CL5	7	0.1488	.667	16.0	5.3	
1	CL2	CL4	10	0.6669	.000	.	16.0	

Output 40.1.13 Dendrogram Using METHOD=WARD

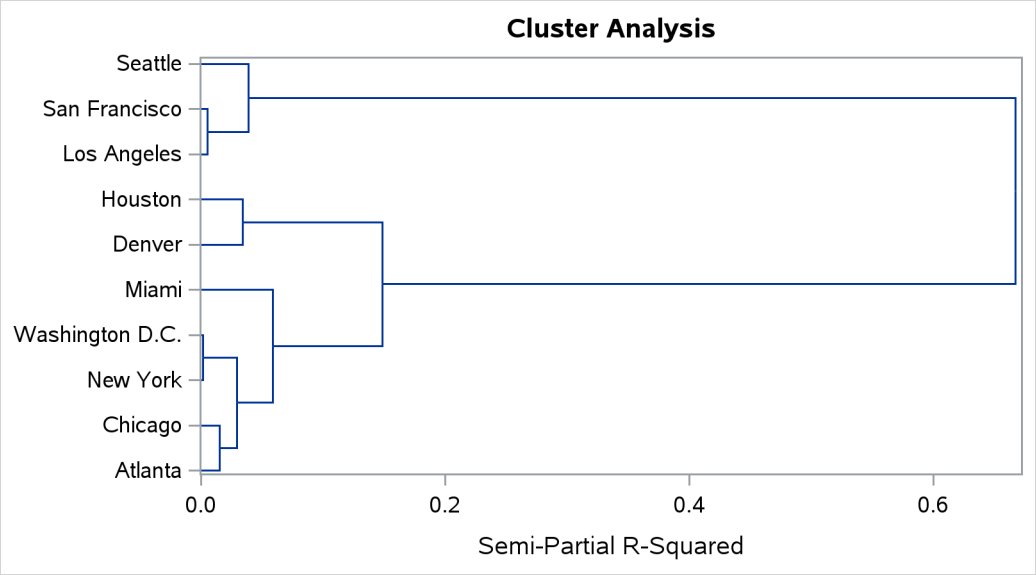

Example 40.2: Crude Birth and Death Rates

(View the complete code for this example (clusex2.sas) in the example repository.)

This example uses the SAS data set Poverty created in the section "Getting Started: CLUSTER Procedure" on page 2565. The data, from Rouncefield (1995), are birth rates, death rates, and infant death rates for 97 countries. Six cluster analyses are performed with eight methods. Scatter plots showing cluster membership at selected levels are produced instead of tree diagrams.

Each cluster analysis is performed by a macro called ANALYZE. The macro takes two arguments. The first, &METHOD, specifies the value of the METHOD= option to be used in the PROC CLUSTER statement. The second, &NCL, must be specified as a list of integers, separated by blanks, indicating the number of clusters desired in each scatter plot. For example, the first invocation of ANALYZE specifies the AVERAGE method and requests plots of three and eight clusters. When two-stage density linkage is used, the K= and R= options are specified as part of the first argument.

The ANALYZE macro first invokes the CLUSTER procedure with METHOD=&METHOD, where &METHOD represents the value of the first argument to ANALYZE. This part of the macro produces the PROC CLUSTER output shown.

The %DO loop processes &NCL, the list of numbers of clusters to plot. The macro variable &K is a counter that indexes the numbers within &NCL. The %SCAN function picks out the *k*th number in &NCL, which is then assigned to the macro variable &N. When &K exceeds the number of numbers in &NCL, %SCAN returns a null string. Thus, the %DO loop executes while &N is not equal to a null string. In the %WHILE condition, a null string is indicated by the absence of any nonblank characters between the comparison operator (NE) and the right parenthesis that terminates the condition.

Within the %DO loop, the TREE procedure creates an output data set containing &N clusters. The SGPLOT procedure then produces a scatter plot in which each observation is identified by the number of the cluster to which it belongs. The TITLE2 statement uses double quotes so that &N and &METHOD can be used within

the title. At the end of the loop, &K is incremented by 1, and the next number is extracted from &NCL by %SCAN.

```
title 'Cluster Analysis of Birth and Death Rates';
ods graphics on;

%macro analyze(method,ncl);
   proc cluster data=poverty outtree=tree method=&method print=15 ccc pseudo;
      var birth death;
      title2;
   run;

   %let k=1;
   %let n=%scan(&ncl,&k);
   %do %while(&n NE);

      proc tree data=tree noprint out=out ncl=&n;
         copy birth death;
      run;

      proc sgplot;
         scatter y=death x=birth / group=cluster;
         title2 "Plot of &n Clusters from METHOD=&METHOD";
      run;

      %let k=%eval(&k+1);
      %let n=%scan(&ncl,&k);
   %end;
%mend;
```

The following statement produces Output 40.2.1, Output 40.2.3, and Output 40.2.4:

```
%analyze(average, 3 8)
```

For average linkage, the CCC has peaks at three, eight, ten, and twelve clusters, but the three-cluster peak is lower than the eight-cluster peak. The pseudo F statistic has peaks at three, eight, and twelve clusters. The pseudo t^2 statistic drops sharply at three clusters, continues to fall at four clusters, and has a particularly low value at twelve clusters. However, there are not enough data to seriously consider as many as twelve clusters. Scatter plots are given for three and eight clusters. The results are shown in Output 40.2.1 through Output 40.2.4. In Output 40.2.4, the eighth cluster consists of the two outlying observations, Mexico and Korea.

Output 40.2.1 Cluster Analysis for Birth and Death Rates: METHOD=AVERAGE

Cluster Analysis of Birth and Death Rates

The CLUSTER Procedure
Average Linkage Cluster Analysis

Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	189.106588	173.101020	0.9220	0.9220
2	16.005568		0.0780	1.0000

Output 40.2.1 *continued*

Root-Mean-Square Total-Sample Standard Deviation	10.127

Root-Mean-Square Distance Between Observations	20.25399

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Norm RMS Distance	Tie
15	CL27	CL20	18	0.0035	.980	.975	2.61	292	18.6	0.2325	
14	CL23	CL17	28	0.0034	.977	.972	1.97	271	17.7	0.2358	
13	CL18	CL54	8	0.0015	.975	.969	2.35	279	7.1	0.2432	
12	CL21	CL26	8	0.0015	.974	.966	2.85	290	6.1	0.2493	
11	CL19	CL24	12	0.0033	.971	.962	2.78	285	14.8	0.2767	
10	CL22	CL16	12	0.0036	.967	.957	2.84	284	17.4	0.2858	
9	CL15	CL28	22	0.0061	.961	.951	2.45	271	17.5	0.3353	
8	OB23	OB61	2	0.0014	.960	.943	3.59	302	.	0.3703	
7	CL25	CL11	17	0.0098	.950	.933	3.01	284	23.3	0.4033	
6	CL7	CL12	25	0.0122	.938	.920	2.63	273	14.8	0.4132	
5	CL10	CL14	40	0.0303	.907	.902	0.59	225	82.7	0.4584	
4	CL13	CL6	33	0.0244	.883	.875	0.77	234	22.2	0.5194	
3	CL9	CL8	24	0.0182	.865	.827	2.13	300	27.7	0.735	
2	CL5	CL3	64	0.1836	.681	.697	-.55	203	148	0.8402	
1	CL2	CL4	97	0.6810	.000	.000	0.00	.	203	1.3348	

Output 40.2.2 Criteria for the Number of Clusters: METHOD=AVERAGE

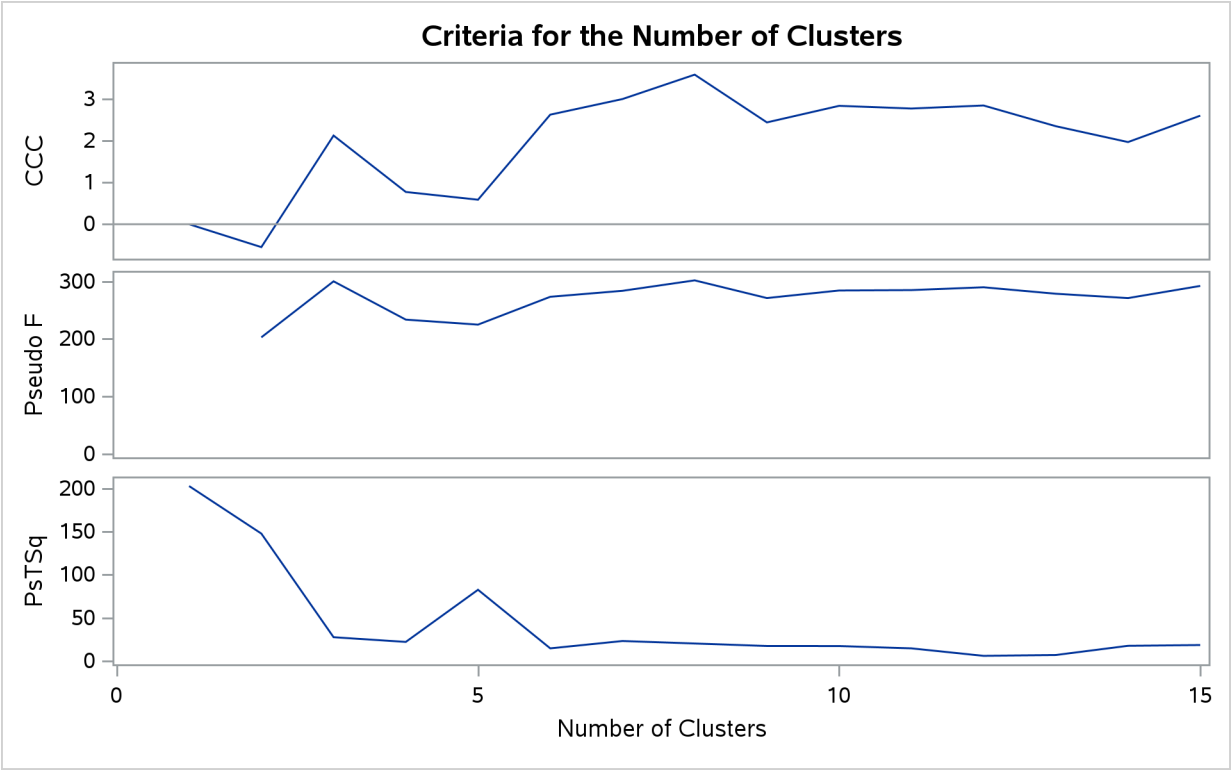

Output 40.2.3 Plot of Three Clusters: METHOD=AVERAGE

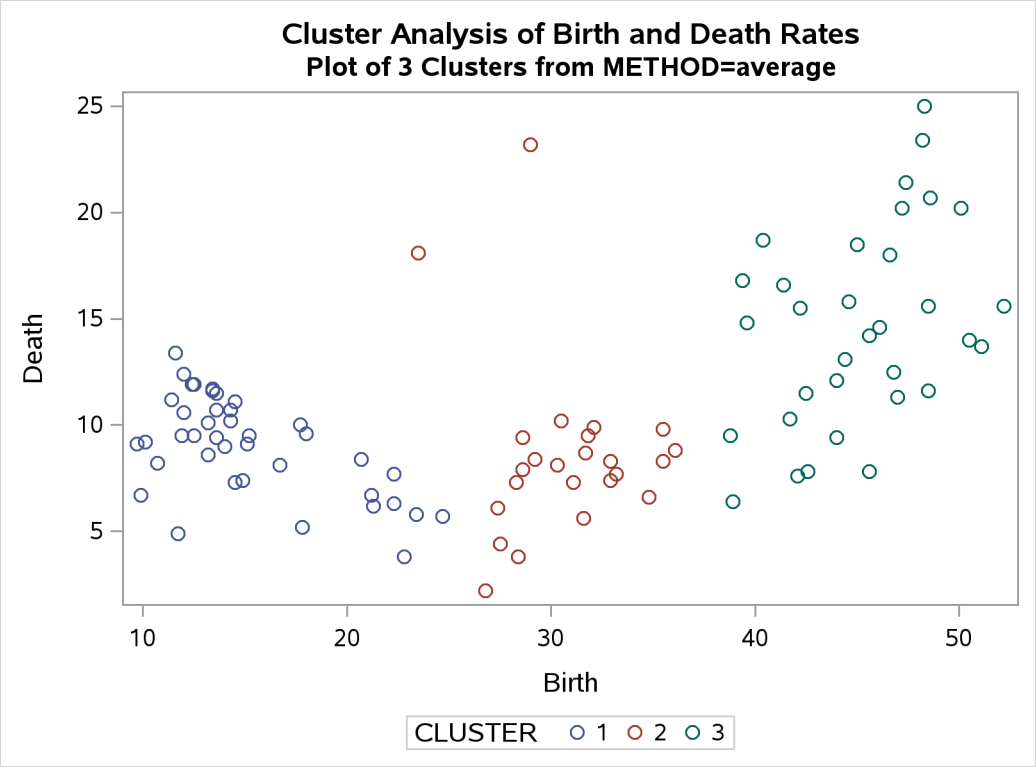

Output 40.2.4 Plot of Eight Clusters: METHOD=AVERAGE

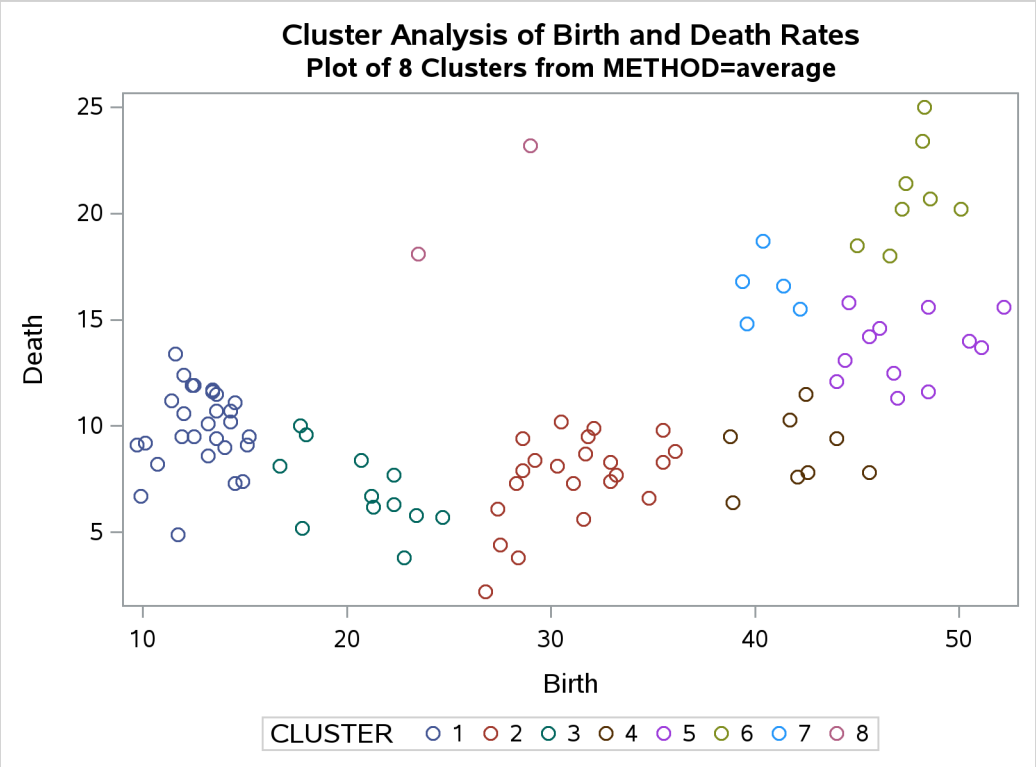

The following statement produces Output 40.2.5 and Output 40.2.7:

```
%analyze(complete, 3)
```

Complete linkage shows CCC peaks at three, eight and twelve clusters. The pseudo F statistic peaks at three and twelve clusters. The pseudo t^2 statistic indicates three clusters.

The scatter plot for three clusters is shown.

Output 40.2.5 Cluster History for Birth and Death Rates: METHOD=COMPLETE

Cluster Analysis of Birth and Death Rates

The CLUSTER Procedure
Complete Linkage Cluster Analysis

Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	189.106588	173.101020	0.9220	0.9220
2	16.005568		0.0780	1.0000

Root-Mean-Square Total-Sample Standard Deviation	10.127

Mean Distance Between Observations	17.13099

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Norm Maximum Distance	Tie
15	CL22	CL33	8	0.0015	.983	.975	3.80	329	6.1	0.4092	
14	CL56	CL18	8	0.0014	.981	.972	3.97	331	6.6	0.4255	
13	CL30	CL44	8	0.0019	.979	.969	4.04	330	19.0	0.4332	
12	OB23	OB61	2	0.0014	.978	.966	4.45	340	.	0.4378	
11	CL19	CL24	24	0.0034	.974	.962	4.17	327	24.1	0.4962	
10	CL17	CL28	12	0.0033	.971	.957	4.18	325	14.8	0.5204	
9	CL20	CL13	16	0.0067	.964	.951	3.38	297	25.2	0.5236	
8	CL11	CL21	32	0.0054	.959	.943	3.44	297	19.7	0.6001	
7	CL26	CL15	13	0.0096	.949	.933	2.93	282	28.9	0.7233	
6	CL14	CL10	20	0.0128	.937	.920	2.46	269	27.7	0.8033	
5	CL9	CL16	30	0.0237	.913	.902	1.29	241	47.1	0.8993	
4	CL6	CL7	33	0.0240	.889	.875	1.38	248	21.7	1.2165	
3	CL5	CL12	32	0.0178	.871	.827	2.56	317	13.6	1.2326	
2	CL3	CL8	64	0.1900	.681	.697	-.55	203	167	1.5412	
1	CL2	CL4	97	0.6810	.000	.000	0.00	.	203	2.5233	

Output 40.2.6 Criteria for the Number of Clusters: METHOD=COMPLETE

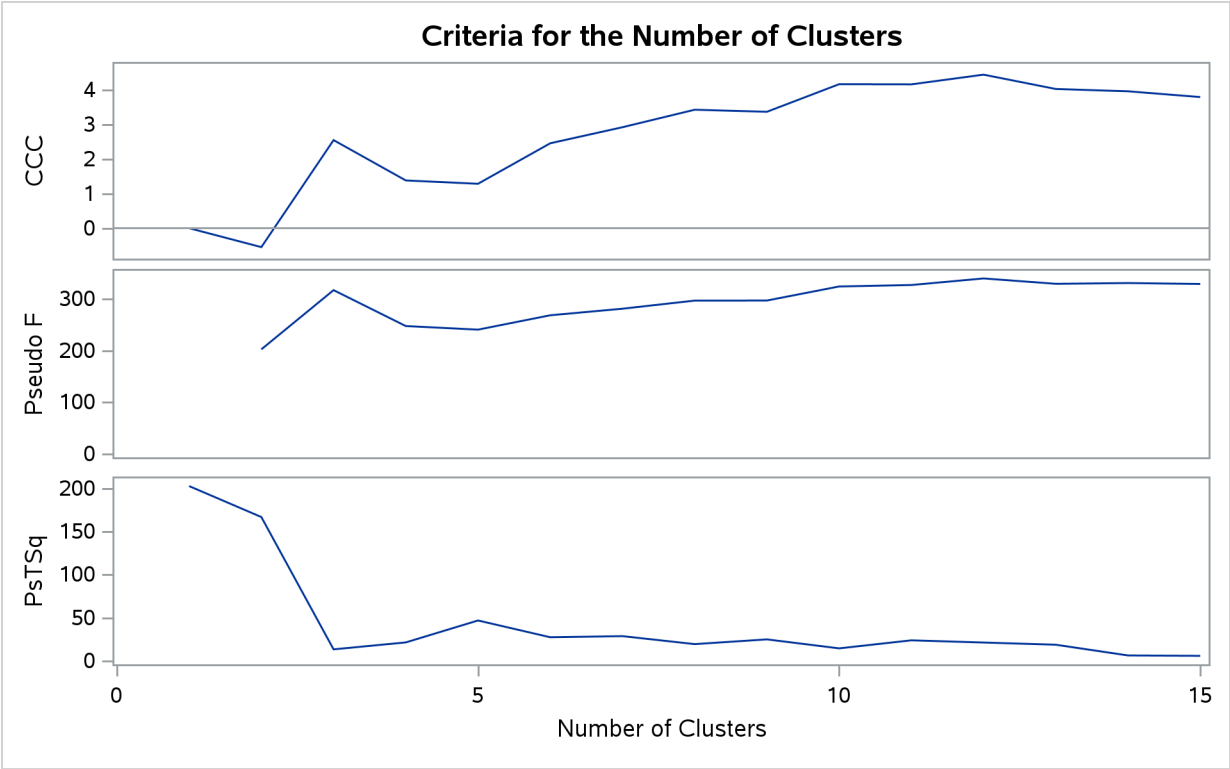

Output 40.2.7 Plot of Clusters for METHOD=COMPLETE

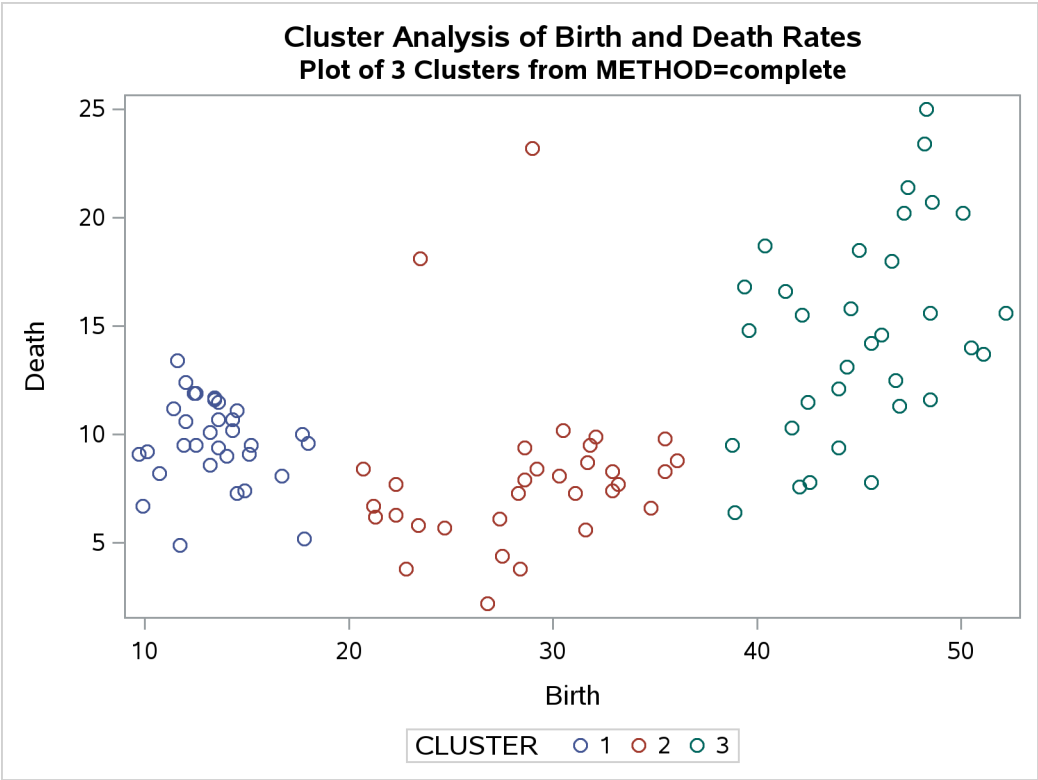

The following statement produces Output 40.2.8 and Output 40.2.10:

```
%analyze(single, 7 10)
```

The CCC and pseudo F statistics are not appropriate for use with single linkage because of the method's tendency to chop off tails of distributions. The pseudo t^2 statistic can be used by looking for *large* values and taking the number of clusters to be one greater than the level at which the large pseudo t^2 value is displayed. For these data, there are large values at levels 6 and 9, suggesting seven or ten clusters.

The scatter plots for seven and ten clusters are shown.

Output 40.2.8 Cluster History for Birth and Death Rates: METHOD=SINGLE

Cluster Analysis of Birth and Death Rates

The CLUSTER Procedure
Single Linkage Cluster Analysis

Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	189.106588	173.101020	0.9220	0.9220
2	16.005568		0.0780	1.0000

Root-Mean-Square Total-Sample Standard Deviation	10.127

Mean Distance Between Observations	17.13099

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Norm Minimum Distance	Tie
15	CL37	CL19	8	0.0014	.968	.975	-2.3	178	6.6	0.1331	
14	CL20	CL23	15	0.0059	.962	.972	-3.1	162	18.7	0.1412	
13	CL14	CL16	19	0.0054	.957	.969	-3.4	155	8.8	0.1442	
12	CL26	OB58	31	0.0014	.955	.966	-2.7	165	4.0	0.1486	
11	OB86	CL18	4	0.0003	.955	.962	-1.6	183	3.8	0.1495	
10	CL13	CL11	23	0.0088	.946	.957	-2.3	170	11.3	0.1518	
9	CL22	CL17	30	0.0235	.923	.951	-4.7	131	45.7	0.1593	T
8	CL15	CL10	31	0.0210	.902	.943	-5.8	117	21.8	0.1593	
7	CL9	OB75	31	0.0052	.897	.933	-4.7	130	4.0	0.1628	
6	CL7	CL12	62	0.2023	.694	.920	-15	41.3	223	0.1725	
5	CL6	CL8	93	0.6681	.026	.902	-26	0.6	199	0.1756	
4	CL5	OB48	94	0.0056	.021	.875	-24	0.7	0.5	0.1811	T
3	CL4	OB67	95	0.0083	.012	.827	-15	0.6	0.8	0.1811	
2	OB23	OB61	2	0.0014	.011	.697	-13	1.0	.	0.4378	
1	CL3	CL2	97	0.0109	.000	.000	0.00	.	1.0	0.5815	

Output 40.2.9 Criteria for the Number of Clusters: METHOD=SINGLE

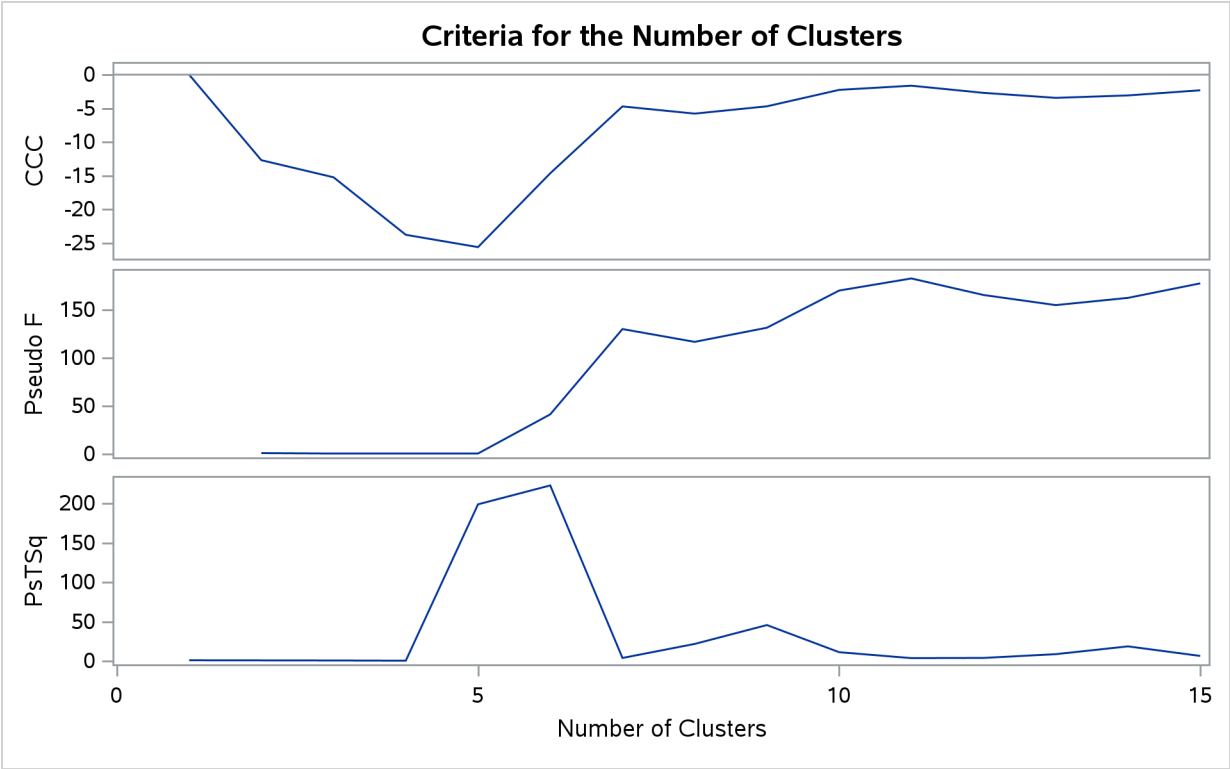

Output 40.2.10 Plot of Clusters for METHOD=SINGLE

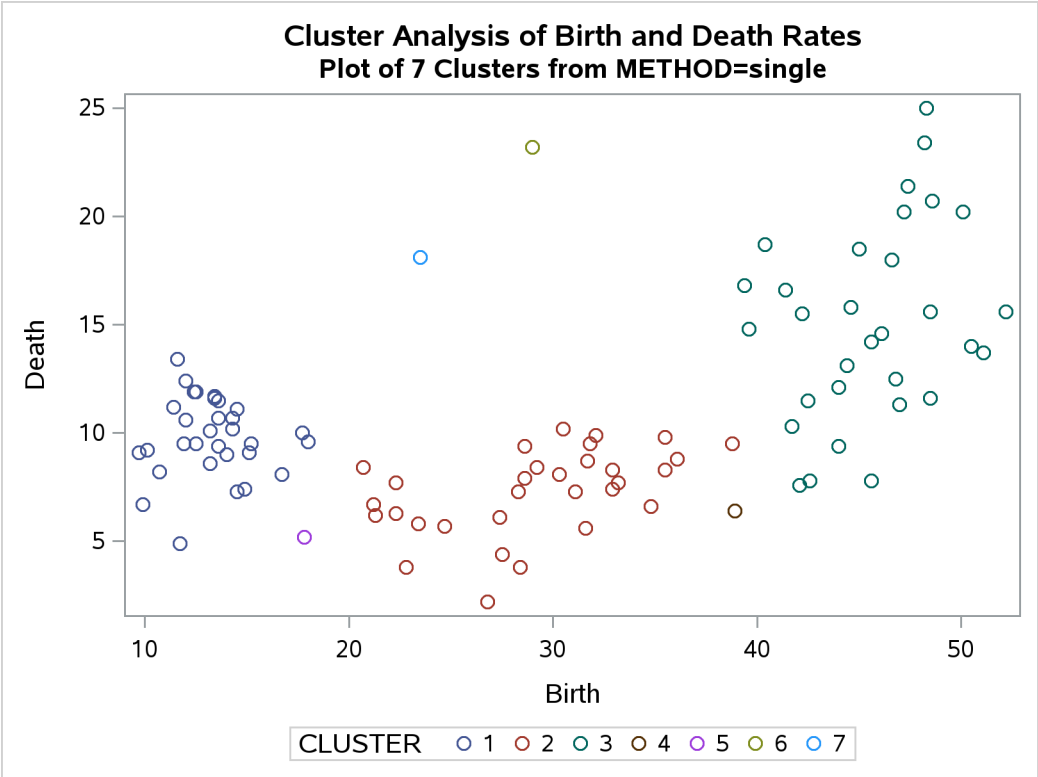

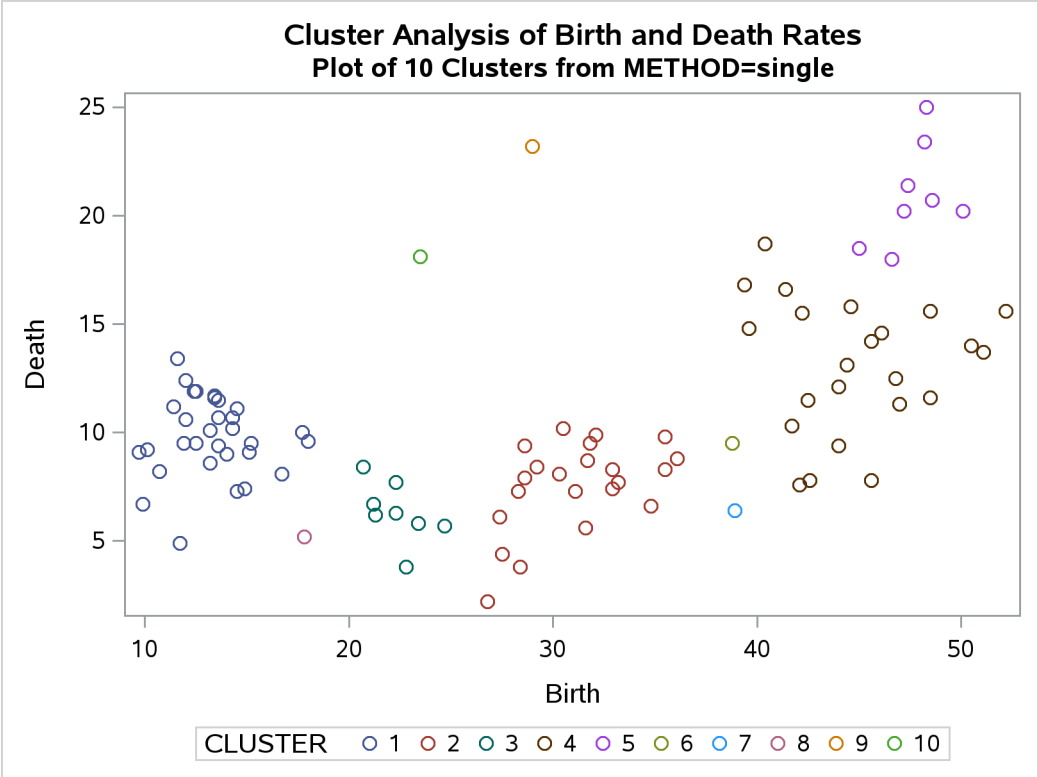

The following statements produce Output 40.2.11 through Output 40.2.14:

```
%analyze(two k=10, 3)

%analyze(two k=18, 2)
```

For *k*th-nearest-neighbor density linkage, the number of modes as a function of *k* is as follows (not all of these analyses are shown):

k	Modes
3	13
4	6
5-7	4
8-15	3
16-21	2
22+	1

Thus, there is strong evidence of three modes and an indication of the possibility of two modes. Uniform-kernel density linkage gives similar results. For K=10 (10th-nearest-neighbor density linkage), the scatter plot for three clusters is shown; and for K=18, the scatter plot for two clusters is shown.

Output 40.2.11 Cluster History for Birth and Death Rates: METHOD=TWOSTAGE K=10

Cluster Analysis of Birth and Death Rates

The CLUSTER Procedure
Two-Stage Density Linkage Clustering

Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	189.106588	173.101020	0.9220	0.9220
2	16.005568		0.0780	1.0000

K = 10

Root-Mean-Square Total-Sample Standard Deviation	10.127

Output 40.2.11 *continued*

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Normalized Fusion Density	Maximum Density in Each Cluster Lesser	Greater	Tie
15	CL16	OB94	22	0.0015	.921	.975	-11	68.4	1.4	9.2234	6.7927	15.3069	
14	CL19	OB49	28	0.0021	.919	.972	-11	72.4	1.8	8.7369	5.9334	33.4385	
13	CL15	OB52	23	0.0024	.917	.969	-10	76.9	2.3	8.5847	5.9651	15.3069	
12	CL13	OB96	24	0.0018	.915	.966	-9.3	83.0	1.6	7.9252	5.4724	15.3069	
11	CL12	OB93	25	0.0025	.912	.962	-8.5	89.5	2.2	7.8913	5.4401	15.3069	
10	CL11	OB78	26	0.0031	.909	.957	-7.7	96.9	2.5	7.787	5.4082	15.3069	
9	CL10	OB76	27	0.0026	.907	.951	-6.7	107	2.1	7.7133	5.4401	15.3069	
8	CL9	OB77	28	0.0023	.904	.943	-5.5	120	1.7	7.4256	4.9017	15.3069	
7	CL8	OB43	29	0.0022	.902	.933	-4.1	138	1.6	6.927	4.4764	15.3069	
6	CL7	OB87	30	0.0043	.898	.920	-2.7	160	3.1	4.932	2.9977	15.3069	
5	CL6	OB82	31	0.0055	.892	.902	-1.1	191	3.7	3.7331	2.1560	15.3069	
4	CL22	OB61	37	0.0079	.884	.875	0.93	237	10.6	3.1713	1.6308	100.0	
3	CL14	OB23	29	0.0126	.872	.827	2.60	320	10.4	2.0654	1.0744	33.4385	
2	CL4	CL3	66	0.2129	.659	.697	-1.3	183	172	12.409	33.4385	100.0	
1	CL2	CL5	97	0.6588	.000	.000	0.00	.	183	10.071	15.3069	100.0	

3 modal clusters have been formed.

Output 40.2.12 Cluster History for Birth and Death Rates: METHOD=TWOSTAGE K=18

Cluster Analysis of Birth and Death Rates

The CLUSTER Procedure
Two-Stage Density Linkage Clustering

Eigenvalues of the Covariance Matrix

	Eigenvalue	Difference	Proportion	Cumulative
1	189.106588	173.101020	0.9220	0.9220
2	16.005568		0.0780	1.0000

K = 18

Root-Mean-Square Total-Sample Standard Deviation	10.127

Output 40.2.12 *continued*

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Normalized Fusion Density	Maximum Density in Each Cluster Lesser	Maximum Density in Each Cluster Greater	Tie
15	CL16	OB72	46	0.0107	.799	.975	-21	23.3	3.0	10.118	7.7445	23.4457	
14	CL15	OB94	47	0.0098	.789	.972	-21	23.9	2.7	9.676	7.1257	23.4457	
13	CL14	OB51	48	0.0037	.786	.969	-20	25.6	1.0	9.409	6.8398	23.4457	T
12	CL13	OB96	49	0.0099	.776	.966	-19	26.7	2.6	9.409	6.8398	23.4457	
11	CL12	OB76	50	0.0114	.764	.962	-19	27.9	2.9	8.8136	6.3138	23.4457	
10	CL11	OB77	51	0.0021	.762	.957	-18	31.0	0.5	8.6593	6.0751	23.4457	
9	CL10	OB78	52	0.0103	.752	.951	-17	33.3	2.5	8.6007	6.0976	23.4457	
8	CL9	OB43	53	0.0034	.748	.943	-16	37.8	0.8	8.4964	5.9160	23.4457	
7	CL8	OB93	54	0.0109	.737	.933	-15	42.1	2.6	8.367	5.7913	23.4457	
6	CL7	OB88	55	0.0110	.726	.920	-13	48.3	2.6	7.916	5.3679	23.4457	
5	CL6	OB87	56	0.0120	.714	.902	-12	57.5	2.7	6.6917	4.3415	23.4457	
4	CL20	OB61	39	0.0077	.707	.875	-9.8	74.7	8.3	6.2578	3.2882	100.0	
3	CL5	OB82	57	0.0138	.693	.827	-5.0	106	3.0	5.3605	3.2834	23.4457	
2	CL3	OB23	58	0.0117	.681	.697	-.54	203	2.5	3.2687	1.7568	23.4457	
1	CL2	CL4	97	0.6812	.000	.000	0.00	.	203	13.764	23.4457	100.0	

2 modal clusters have been formed.

Output 40.2.13 Plot of Clusters for METHOD=TWOSTAGE K=10

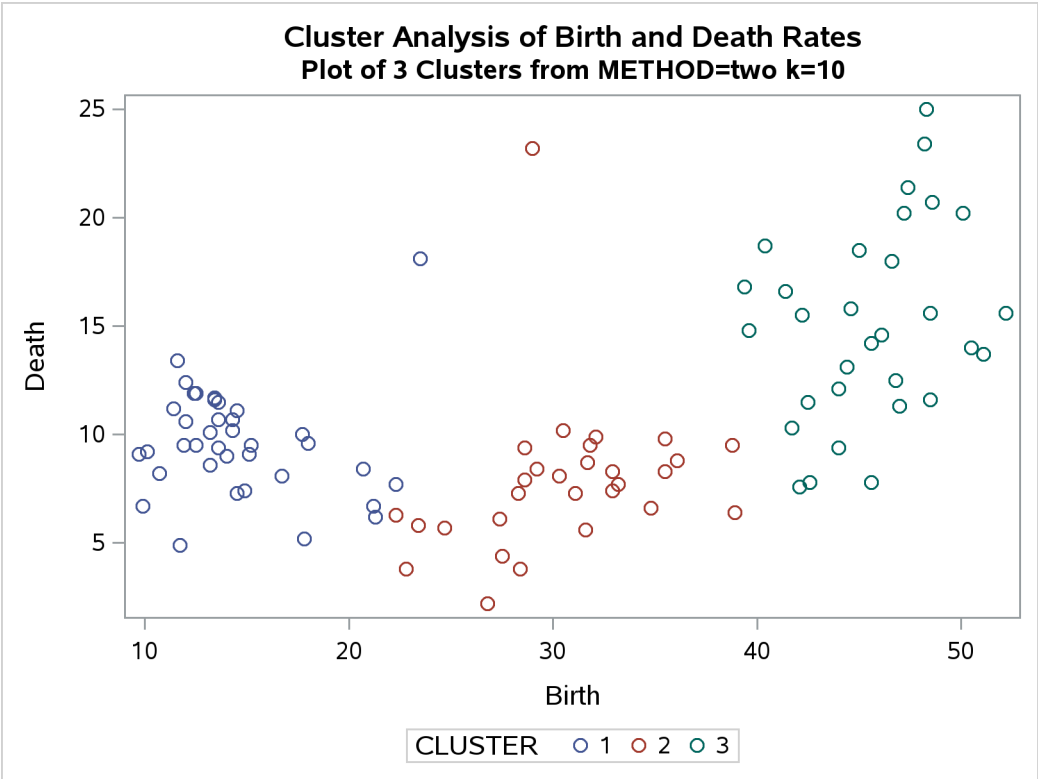

Output 40.2.14 Plot of Clusters for METHOD=TWOSTAGE K=18

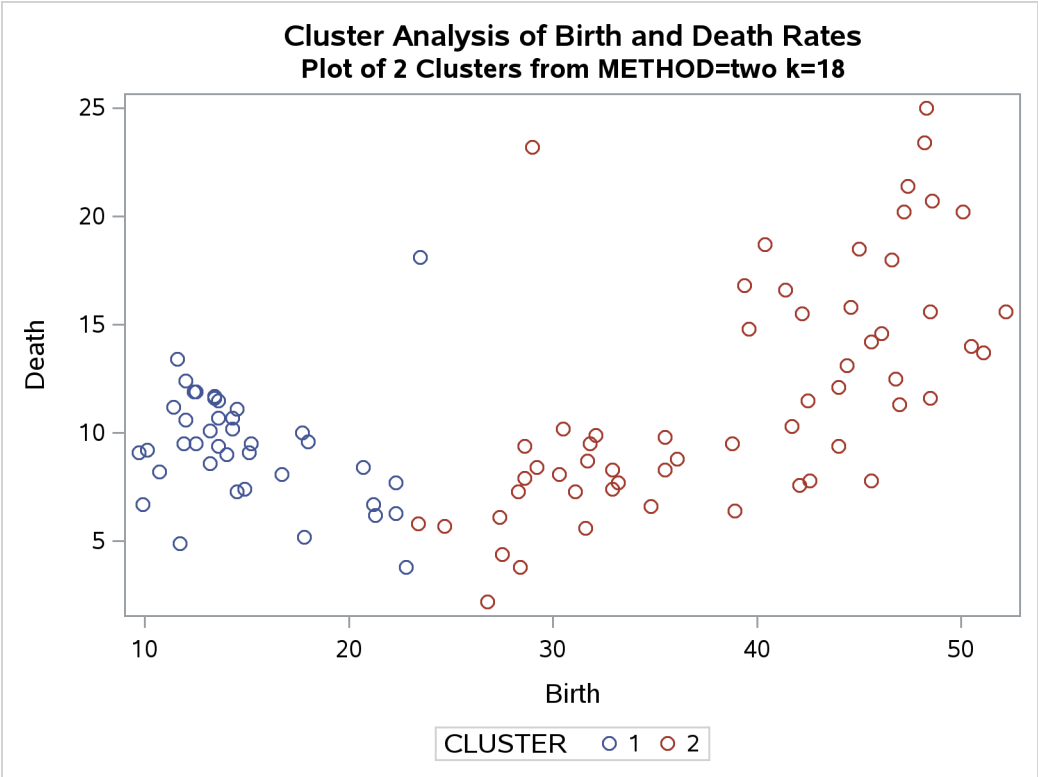

In summary, most of the clustering methods indicate three or eight clusters. Most methods agree at the three-cluster level, but at the other levels, there is considerable disagreement about the composition of the clusters. The presence of numerous ties also complicates the analysis; see Example 40.4.

Example 40.3: Cluster Analysis of Fisher's Iris Data

(View the complete code for this example (clusex3.sas) in the example repository.)

The iris data published by Fisher (1936) have been widely used for examples in discriminant analysis and cluster analysis. The sepal length, sepal width, petal length, and petal width are measured in millimeters on 50 iris specimens from each of three species, *Iris setosa, I. versicolor,* and *I. virginica*. Mezzich and Solomon (1980) discuss a variety of cluster analyses of the iris data. The following statements create the data set Iris:

```
proc format;
   value specname
      1='Setosa    '
      2='Versicolor'
      3='Virginica ';
run;

data iris;
   title 'Fisher (1936) Iris Data';
   input SepalLength SepalWidth PetalLength PetalWidth Species @@;
   format Species specname.;
```

```
   label SepalLength='Sepal Length in mm.'
         SepalWidth ='Sepal Width in mm.'
         PetalLength='Petal Length in mm.'
         PetalWidth ='Petal Width in mm.';
   datalines;
50 33 14 02 1 64 28 56 22 3 65 28 46 15 2 67 31 56 24 3
63 28 51 15 3 46 34 14 03 1 69 31 51 23 3 62 22 45 15 2
59 32 48 18 2 46 36 10 02 1 61 30 46 14 2 60 27 51 16 2
65 30 52 20 3 56 25 39 11 2 65 30 55 18 3 58 27 51 19 3
68 32 59 23 3 51 33 17 05 1 57 28 45 13 2 62 34 54 23 3
77 38 67 22 3 63 33 47 16 2 67 33 57 25 3 76 30 66 21 3
49 25 45 17 3 55 35 13 02 1 67 30 52 23 3 70 32 47 14 2

   ... more lines ...

55 23 40 13 2 66 30 44 14 2 68 28 48 14 2 54 34 17 02 1
51 37 15 04 1 52 35 15 02 1 58 28 51 24 3 67 30 50 17 2
63 33 60 25 3 53 37 15 02 1
;
```

This example analyzes the iris data by using Ward's method and two-stage density linkage and then illustrates how the FASTCLUS procedure can be used in combination with PROC CLUSTER to analyze large data sets.

The following macro, SHOW, is used in the subsequent analyses to display cluster results. It invokes the FREQ procedure to crosstabulate clusters and species. The CANDISC procedure computes canonical variables for discriminating among the clusters, and the first two canonical variables are plotted to show cluster membership. See Chapter 35, "The CANDISC Procedure," for a canonical discriminant analysis of the iris species.

```
/*--- Define macro show ---*/
%macro show;
   proc freq;
      tables cluster*species / nopercent norow nocol plot=none;
   run;

   proc candisc noprint out=can;
      class cluster;
      var petal: sepal:;
   run;

   proc sgplot data=can;
      scatter y=can2 x=can1 / group=cluster;
   run;
%mend;
```

The first analysis clusters the iris data by using Ward's method (see Output 40.3.1) and plots the CCC and pseudo F and t^2 statistics (see Output 40.3.2). The CCC has a local peak at three clusters but a higher peak at five clusters. The pseudo F statistic indicates three clusters, while the pseudo t^2 statistic suggests three or six clusters.

The TREE procedure creates an output data set containing the three-cluster partition for use by the SHOW macro. The FREQ procedure reveals 16 misclassifications. The results are shown in Output 40.3.3.

```
title2 'By Ward''s Method';
ods graphics on;

proc cluster data=iris method=ward print=15 ccc pseudo;
   var petal: sepal:;
   copy species;
run;

proc tree noprint ncl=3 out=out;
   copy petal: sepal: species;
run;

%show;
```

Output 40.3.1 Cluster Analysis of Fisher's Iris Data: PROC CLUSTER with METHOD=WARD

Fisher (1936) Iris Data
By Ward's Method

The CLUSTER Procedure
Ward's Minimum Variance Cluster Analysis

Eigenvalues of the Covariance Matrix

	Eigenvalue	Difference	Proportion	Cumulative
1	422.824171	398.557096	0.9246	0.9246
2	24.267075	16.446125	0.0531	0.9777
3	7.820950	5.437441	0.0171	0.9948
4	2.383509		0.0052	1.0000

Root-Mean-Square Total-Sample Standard Deviation	10.69224

Root-Mean-Square Distance Between Observations	30.24221

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Tie
15	CL24	CL28	15	0.0016	.971	.958	5.93	324	9.8	
14	CL21	CL53	7	0.0019	.969	.955	5.85	329	5.1	
13	CL18	CL48	15	0.0023	.967	.953	5.69	334	8.9	
12	CL16	CL23	24	0.0023	.965	.950	4.63	342	9.6	
11	CL14	CL43	12	0.0025	.962	.946	4.67	353	5.8	
10	CL26	CL20	22	0.0027	.959	.942	4.81	368	12.9	
9	CL27	CL17	31	0.0031	.956	.936	5.02	387	17.8	
8	CL35	CL15	23	0.0031	.953	.930	5.44	414	13.8	
7	CL10	CL47	26	0.0058	.947	.921	5.43	430	19.1	
6	CL8	CL13	38	0.0060	.941	.911	5.81	463	16.3	
5	CL9	CL19	50	0.0105	.931	.895	5.82	488	43.2	
4	CL12	CL11	36	0.0172	.914	.872	3.99	515	41.0	
3	CL6	CL7	64	0.0301	.884	.827	4.33	558	57.2	
2	CL4	CL3	100	0.1110	.773	.697	3.83	503	116	
1	CL5	CL2	150	0.7726	.000	.000	0.00	.	503	

Output 40.3.2 Criteria for the Number of Clusters with METHOD=WARD

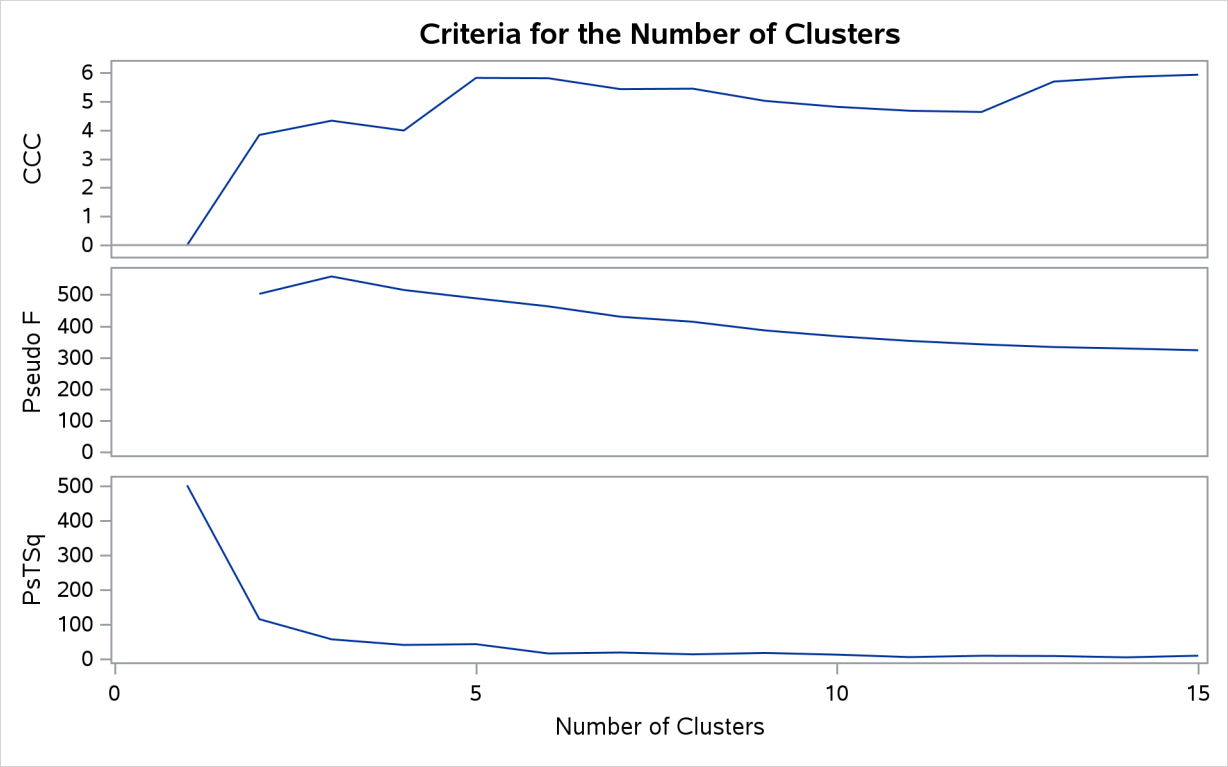

Output 40.3.3 Crosstabulation of Clusters for METHOD=WARD

Fisher (1936) Iris Data
By Ward's Method

The FREQ Procedure

Frequency

Table of CLUSTER by Species

CLUSTER	Species: Setosa	Versicolor	Virginica	Total
1	0	49	15	64
2	0	1	35	36
3	50	0	0	50
Total	50	50	50	150

Output 40.3.4 Scatter Plot of Clusters for METHOD=WARD

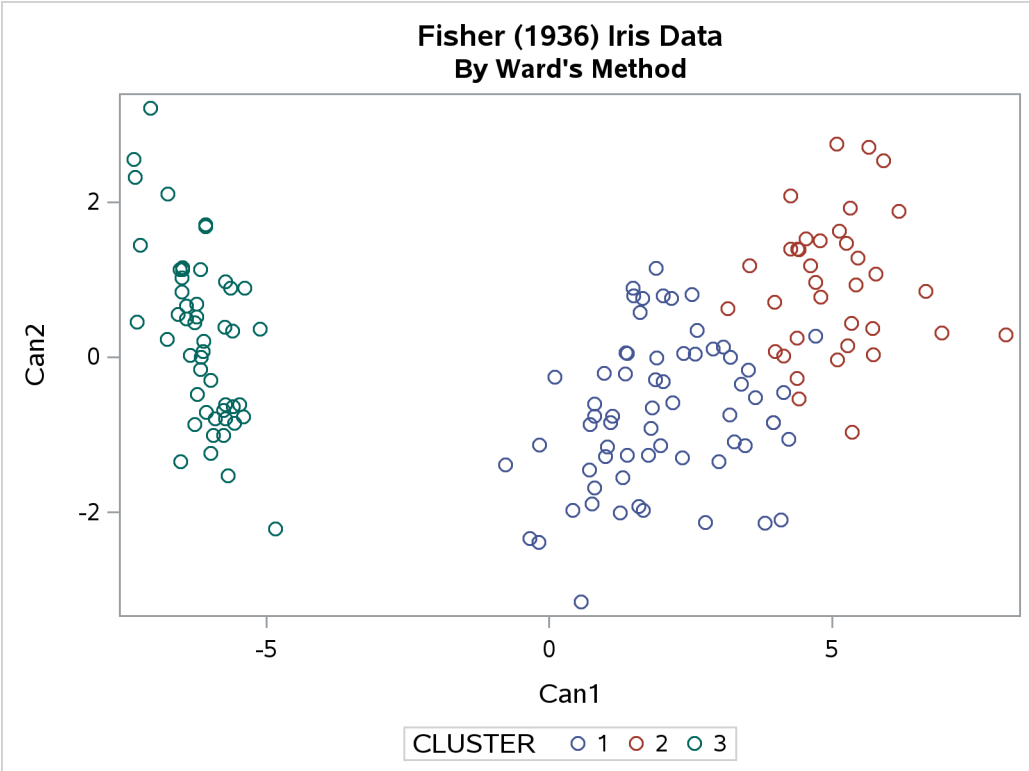

The second analysis uses two-stage density linkage. The raw data suggest two or six modes instead of three:

k	Modes
3	12
4-6	6
7	4
8	3
9-50	2
51+	1

The following analysis uses K=8 to produce three clusters for comparison with other analyses. There are only six misclassifications. The results are shown in Output 40.3.5 and Output 40.3.6.

```
title2 'By Two-Stage Density Linkage';

proc cluster data=iris method=twostage k=8 print=15 ccc pseudo;
   var petal: sepal:;
   copy species;
run;

proc tree noprint ncl=3 out=out;
   copy petal: sepal: species;
run;
```

```
%show;
```

Output 40.3.5 Cluster Analysis of Fisher's Iris Data: PROC CLUSTER with METHOD=TWOSTAGE

Fisher (1936) Iris Data
By Two-Stage Density Linkage

The CLUSTER Procedure
Two-Stage Density Linkage Clustering

Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	422.824171	398.557096	0.9246	0.9246
2	24.267075	16.446125	0.0531	0.9777
3	7.820950	5.437441	0.0171	0.9948
4	2.383509		0.0052	1.0000

K = 8

Root-Mean-Square Total-Sample Standard Deviation	10.69224

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Normalized Fusion Density	Maximum Density in Each Cluster Lesser	Greater	Tie
15	CL17	OB127	43	0.0024	.917	.958	-11	107	3.4	0.3903	0.2066	3.5156	
14	CL16	OB137	50	0.0023	.915	.955	-10	113	5.6	0.3637	0.1837	100.0	
13	CL15	OB74	44	0.0029	.912	.953	-9.8	119	3.8	0.3553	0.2130	3.5156	
12	CL22	OB49	47	0.0036	.909	.950	-7.7	125	5.2	0.3223	0.1736	8.3678	T
11	CL12	OB85	48	0.0036	.905	.946	-7.4	132	4.8	0.3223	0.1736	8.3678	
10	CL11	OB98	49	0.0033	.902	.942	-6.8	143	4.1	0.2879	0.1479	8.3678	
9	CL13	OB24	45	0.0036	.898	.936	-6.2	155	4.5	0.2802	0.2005	3.5156	
8	CL10	OB25	50	0.0019	.896	.930	-5.2	175	2.2	0.2699	0.1372	8.3678	
7	CL8	OB121	51	0.0035	.893	.921	-4.2	198	4.0	0.2586	0.1372	8.3678	
6	CL9	OB45	46	0.0041	.888	.911	-3.0	229	4.7	0.1412	0.0832	3.5156	
5	CL6	OB39	47	0.0048	.884	.895	-1.5	276	5.1	0.107	0.0605	3.5156	
4	CL5	OB21	48	0.0048	.879	.872	0.54	353	4.7	0.0969	0.0541	3.5156	
3	CL4	OB90	49	0.0046	.874	.827	3.49	511	4.2	0.0715	0.0370	3.5156	
2	CL7	CL3	100	0.1017	.773	.697	3.83	503	96.3	2.6277	3.5156	8.3678	

3 modal clusters have been formed.

Output 40.3.6 Criteria for the Number of Clusters with METHOD=TWOSTAGE

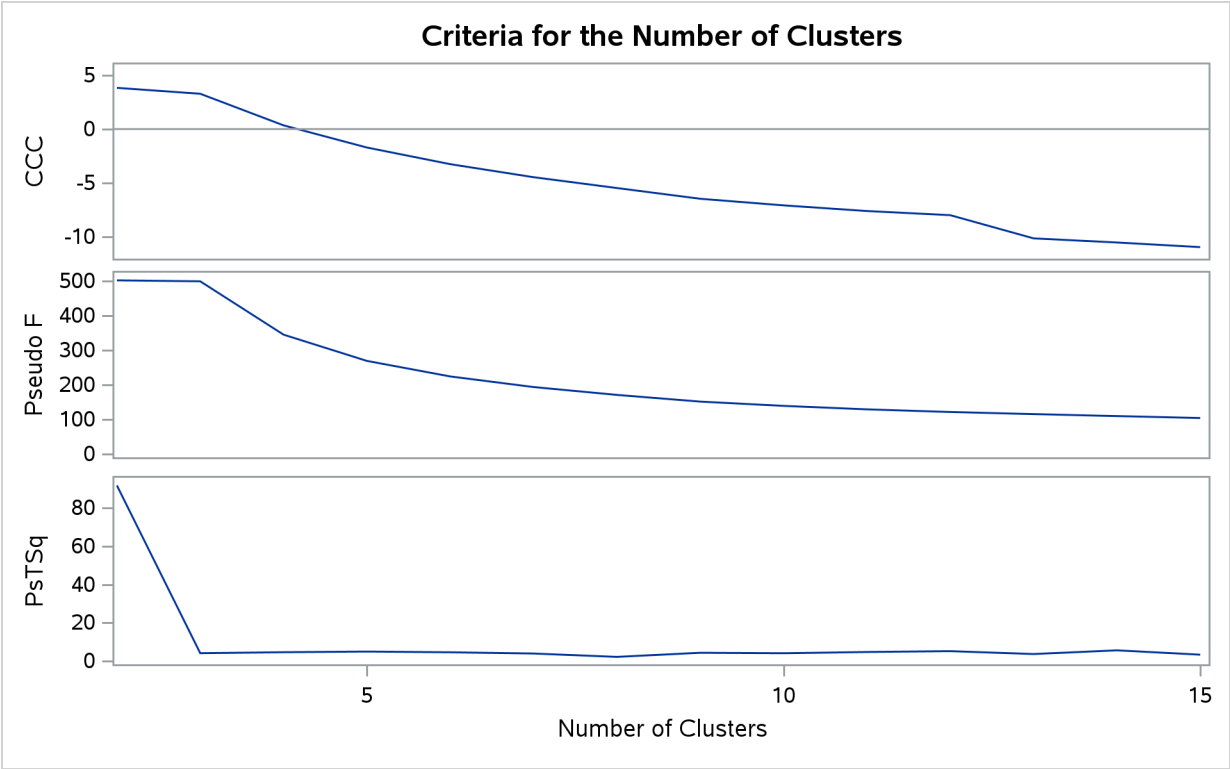

Output 40.3.7 Crosstabulation of Clusters for METHOD=TWOSTAGE

Fisher (1936) Iris Data
By Two-Stage Density Linkage

The FREQ Procedure

Frequency	Table of CLUSTER by Species			
		Species		
CLUSTER	Setosa	Versicolor	Virginica	Total
1	50	0	0	50
2	0	48	3	51
3	0	2	47	49
Total	50	50	50	150

Output 40.3.8 Scatter Plot of Clusters for METHOD=TWOSTAGE

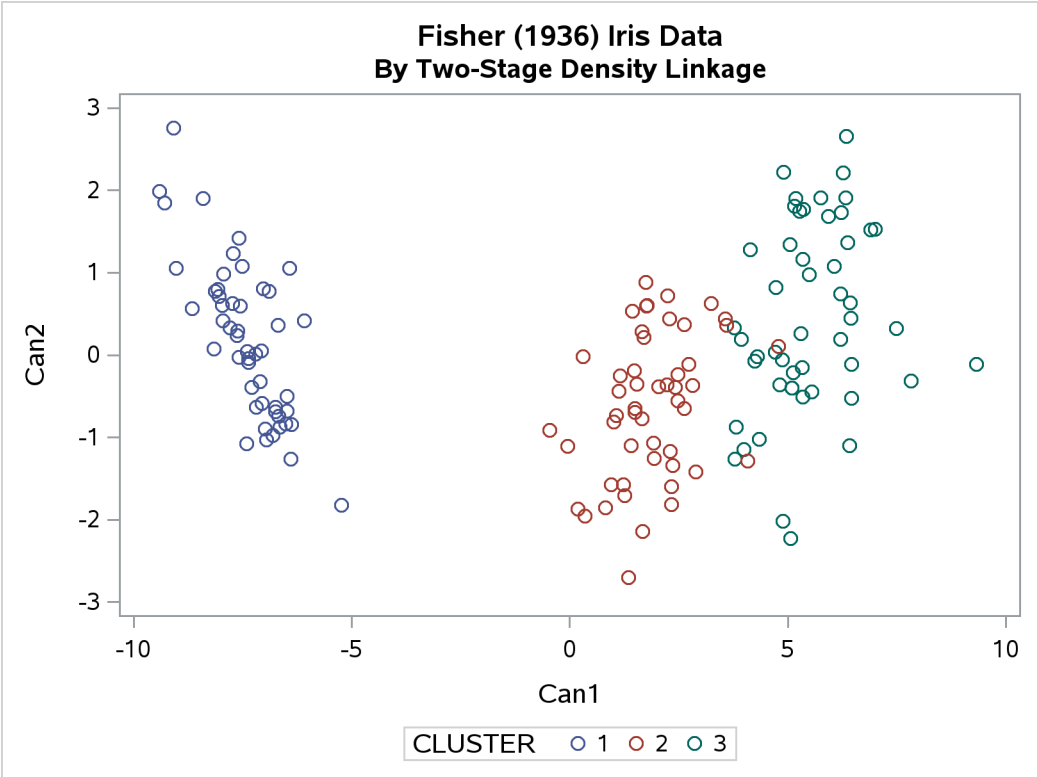

The CLUSTER procedure is not practical for very large data sets because, with most methods, the CPU time is roughly proportional to the square or cube of the number of observations. The FASTCLUS procedure requires time proportional to the number of observations and can therefore be used with much larger data sets than PROC CLUSTER. If you want to hierarchically cluster a very large data set, you can use PROC FASTCLUS for a preliminary cluster analysis to produce a large number of clusters and then use PROC CLUSTER to hierarchically cluster the preliminary clusters.

FASTCLUS automatically creates the variables _FREQ_ and _RMSSTD_ in the MEAN= output data set. These variables are then automatically used by PROC CLUSTER in the computation of various statistics.

The following SAS code uses the iris data to illustrate the process of clustering clusters. In the preliminary analysis, PROC FASTCLUS produces ten clusters, which are then crosstabulated with species. The data set containing the preliminary clusters is sorted in preparation for later merges. The results are shown in Output 40.3.9 and Output 40.3.10.

```
title2 'Preliminary Analysis by FASTCLUS';
proc fastclus data=iris summary maxc=10 maxiter=99 converge=0
              mean=mean out=prelim cluster=preclus;
   var petal: sepal:;
run;

proc freq;
   tables preclus*species / nopercent norow nocol plot=none;
run;

proc sort data=prelim;
```

```
   by preclus;
run;
```

Output 40.3.9 Preliminary Analysis of Fisher's Iris Data: FASTCLUS Procedure

Fisher (1936) Iris Data
Preliminary Analysis by FASTCLUS

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=10 Maxiter=99 Converge=0

Convergence criterion is satisfied.

Criterion Based on Final Seeds = 2.1389

Cluster Summary

Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	9	2.7067	8.2027		5	8.7362
2	19	2.2001	7.7340		4	6.2243
3	18	2.1496	6.2173		8	7.5049
4	4	2.5249	5.3268		2	6.2243
5	3	2.7234	5.8214		1	8.7362
6	7	2.2939	5.1508		2	9.3318
7	17	2.0274	6.9576		10	7.9503
8	18	2.2628	7.1135		3	7.5049
9	22	2.2666	7.5029		8	9.0090
10	33	2.0594	10.0033		7	7.9503

Pseudo F Statistic = 370.58

Observed Over-All R-Squared = 0.95971

Approximate Expected Over-All R-Squared = 0.82928

Cubic Clustering Criterion = 27.077

WARNING: The two values above are invalid for correlated variables.

Output 40.3.10 Crosstabulation of Species and Cluster From the FASTCLUS Procedure

Fisher (1936) Iris Data
Preliminary Analysis by FASTCLUS

The FREQ Procedure

Frequency — **Table of preclus by Species**

preclus(Cluster)	Species: Setosa	Versicolor	Virginica	Total
1	0	0	9	9
2	0	19	0	19
3	0	18	0	18
4	0	3	1	4
5	0	0	3	3
6	0	7	0	7
7	17	0	0	17
8	0	3	15	18
9	0	0	22	22
10	33	0	0	33
Total	50	50	50	150

The following macro, CLUS, clusters the preliminary clusters. There is one argument to choose the METHOD= specification to be used by PROC CLUSTER. The TREE procedure creates an output data set containing the three-cluster partition, which is sorted and merged with the OUT= data set from PROC FASTCLUS to determine which cluster each of the original 150 observations belongs to. The SHOW macro is then used to display the results. In this example, the CLUS macro is invoked using Ward's method, which produces 16 misclassifications, and Wong's hybrid method, which produces 22 misclassifications.

```
/*--- Define macro clus ---*/
%macro clus(method);
   proc cluster data=mean method=&method ccc pseudo;
      var petal: sepal:;
      copy preclus;
   run;

   proc tree noprint ncl=3 out=out;
      copy petal: sepal: preclus;
   run;

   proc sort data=out;
      by preclus;
   run;

   data clus;
      merge out prelim;
      by preclus;
   run;

   %show;
%mend;
```

The following statements produce Output 40.3.11 through Output 40.3.14.

```
title2 'Clustering Clusters by Ward''s Method';
%clus(ward);
```

Output 40.3.11 Clustering Clusters by Ward's Method

Fisher (1936) Iris Data
Clustering Clusters by Ward's Method

The CLUSTER Procedure
Ward's Minimum Variance Cluster Analysis

Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	416.976349	398.666421	0.9501	0.9501
2	18.309928	14.952922	0.0417	0.9918
3	3.357006	3.126943	0.0076	0.9995
4	0.230063		0.0005	1.0000

Root-Mean-Square Total-Sample Standard Deviation	10.69224

Root-Mean-Square Distance Between Observations	30.24221

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Tie
9	OB2	OB4	23	0.0019	.958	.932	6.26	400	6.3	
8	OB1	OB5	12	0.0025	.955	.926	6.75	434	5.8	
7	CL9	OB6	30	0.0069	.948	.918	6.28	438	19.5	
6	OB3	OB8	36	0.0074	.941	.907	6.21	459	26.0	
5	OB7	OB10	50	0.0104	.931	.892	6.15	485	42.2	
4	CL8	OB9	34	0.0162	.914	.870	4.28	519	39.3	
3	CL7	CL6	66	0.0318	.883	.824	4.39	552	59.7	
2	CL4	CL3	100	0.1099	.773	.695	3.94	503	113	
1	CL2	CL5	150	0.7726	.000	.000	0.00	.	503	

Output 40.3.12 Criteria for the Number of Clusters for Clustering Clusters from Ward's Method

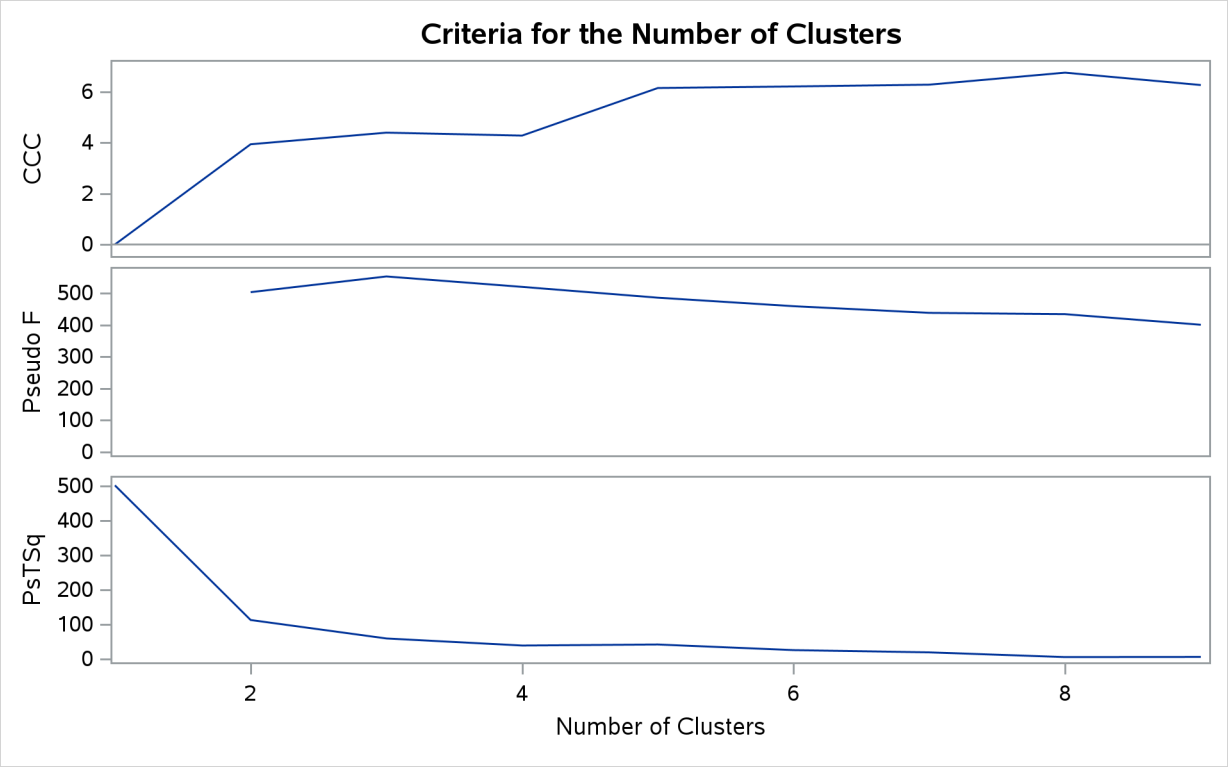

Output 40.3.13 Crosstabulation for Clustering Clusters from Ward's Method

Fisher (1936) Iris Data
Clustering Clusters by Ward's Method

The FREQ Procedure

Frequency	Table of CLUSTER by Species			
		Species		
CLUSTER	Setosa	Versicolor	Virginica	Total
1	0	50	16	66
2	0	0	34	34
3	50	0	0	50
Total	50	50	50	150

Output 40.3.14 Scatter Plot for Clustering Clusters using Ward's Method

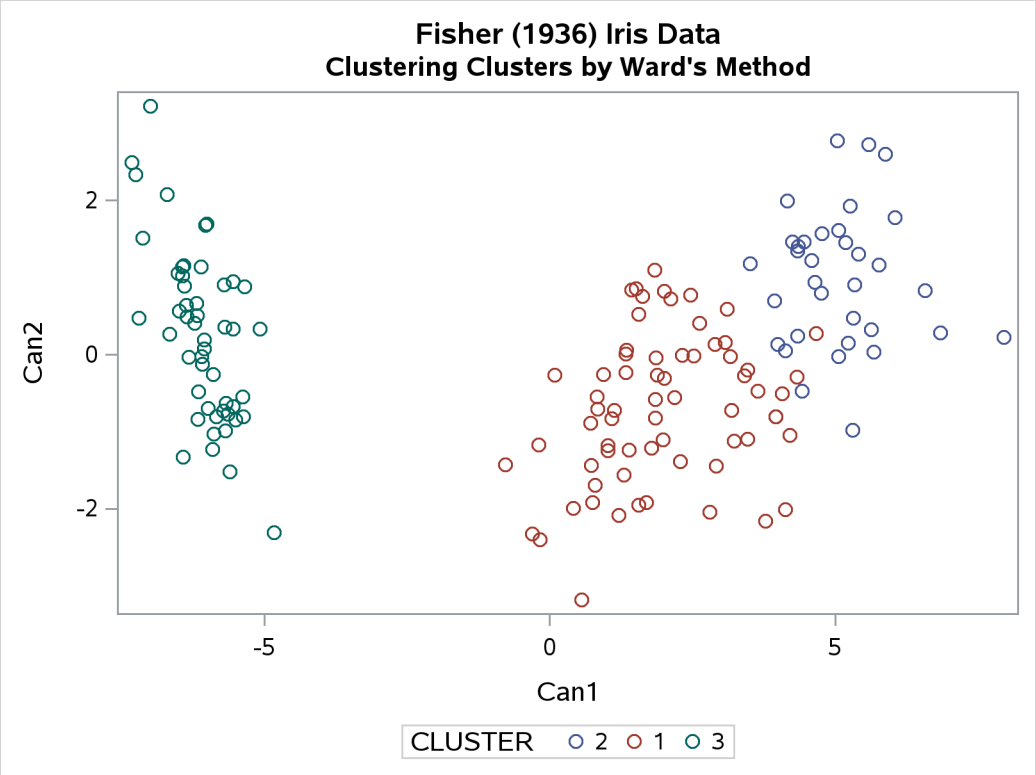

The following statements produce Output 40.3.15 through Output 40.3.17.

```
title2 "Clustering Clusters by Wong's Hybrid Method";
%clus(twostage hybrid);
```

Output 40.3.15 Clustering Clusters by Wong's Hybrid Method

Fisher (1936) Iris Data
Clustering Clusters by Wong's Hybrid Method

The CLUSTER Procedure
Two-Stage Density Linkage Clustering

Eigenvalues of the Covariance Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	416.976349	398.666421	0.9501	0.9501
2	18.309928	14.952922	0.0417	0.9918
3	3.357006	3.126943	0.0076	0.9995
4	0.230063		0.0005	1.0000

Root-Mean-Square Total-Sample Standard Deviation	10.69224

Output 40.3.15 *continued*

Cluster History

Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Normalized Fusion Density	Maximum Density in Each Cluster Lesser	Maximum Density in Each Cluster Greater	Tie
9	OB10	OB7	50	0.0104	.949	.932	3.81	330	42.2	40.24	58.2179	100.0	
8	OB3	OB8	36	0.0074	.942	.926	3.22	329	26.0	27.981	39.4511	48.4350	
7	OB2	OB4	23	0.0019	.940	.918	4.24	373	6.3	23.775	8.9675	46.3026	
6	CL8	OB9	58	0.0194	.921	.907	2.13	334	46.3	20.724	46.8846	48.4350	
5	CL7	OB6	30	0.0069	.914	.892	3.09	383	19.5	13.303	17.6360	46.3026	
4	CL6	OB1	67	0.0292	.884	.870	1.21	372	41.0	8.4137	10.8758	48.4350	
3	CL4	OB5	70	0.0138	.871	.824	3.33	494	12.3	5.1855	6.2890	48.4350	
2	CL3	CL5	100	0.0979	.773	.695	3.94	503	89.5	19.513	46.3026	48.4350	
1	CL2	CL9	150	0.7726	.000	.000	0.00	.	503	1.3337	48.4350	100.0	

3 modal clusters have been formed.

Output 40.3.16 Crosstabulation for Clustering Clusters from Wong's Hybrid Method

Fisher (1936) Iris Data
Clustering Clusters by Wong's Hybrid Method

The FREQ Procedure

Frequency

Table of CLUSTER by Species

CLUSTER	Species Setosa	Versicolor	Virginica	Total
1	50	0	0	50
2	0	21	49	70
3	0	29	1	30
Total	50	50	50	150

Output 40.3.17 Scatter Plot for Clustering Clusters using Wong's Hybrid Method

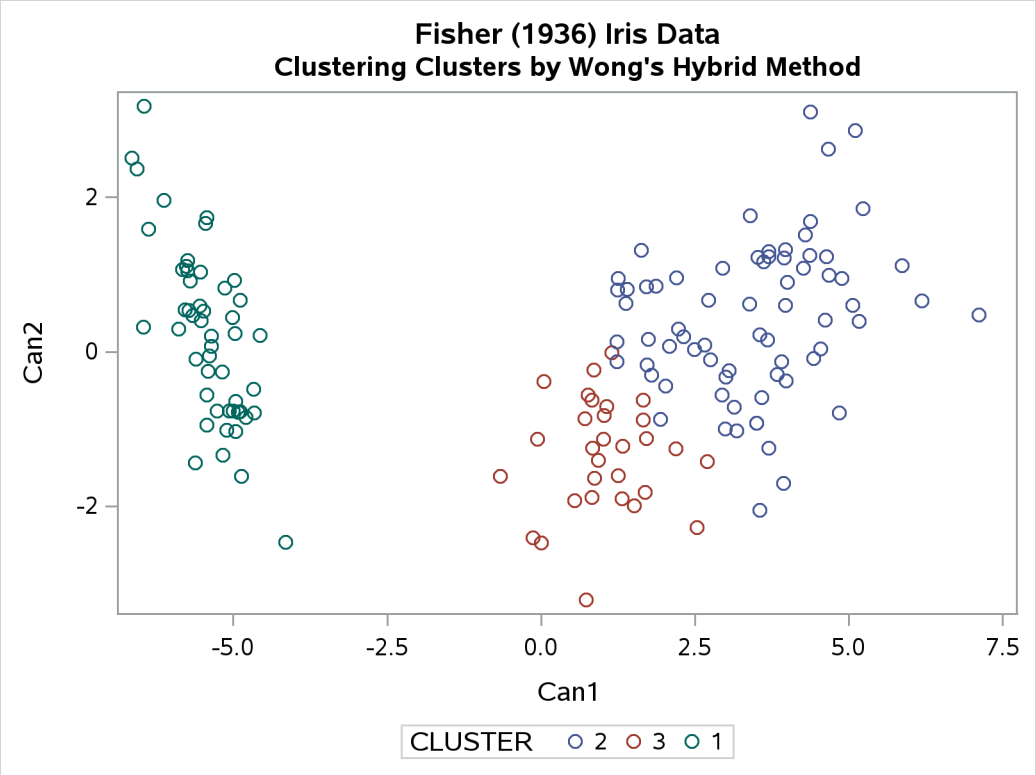

Example 40.4: Evaluating the Effects of Ties

(View the complete code for this example (clusex4.sas) in the example repository.)

If, at some level of the cluster history, there is a tie for minimum distance between clusters, then one or more levels of the sample cluster tree are not uniquely determined. This example shows how the degree of indeterminacy can be assessed.

Mammals have four kinds of teeth: incisors, canines, premolars, and molars. The following data set gives the number of teeth of each kind on one side of the top and bottom jaws for 32 mammals.

Since all eight variables are measured in the same units, it is not strictly necessary to rescale the data. However, the canines have much less variance than the other kinds of teeth and, therefore, have little effect on the analysis if the variables are not standardized. An average linkage cluster analysis is run with and without standardization to enable comparison of the results.

```
title 'Hierarchical Cluster Analysis of Mammals'' Teeth Data';
title2 'Evaluating the Effects of Ties';
data teeth;
   input Mammal & $16. v1-v8 @@;
   label v1='Top incisors'
         v2='Bottom incisors'
         v3='Top canines'
         v4='Bottom canines'
         v5='Top premolars'
```

```
         v6='Bottom premolars'
         v7='Top molars'
         v8='Bottom molars';
   datalines;
Brown Bat         2 3 1 1 3 3 3 3   Mole              3 2 1 0 3 3 3 3
Silver Hair Bat   2 3 1 1 2 3 3 3   Pigmy Bat         2 3 1 1 2 2 3 3
House Bat         2 3 1 1 1 2 3 3   Red Bat           1 3 1 1 2 2 3 3
Pika              2 1 0 0 2 2 3 3   Rabbit            2 1 0 0 3 2 3 3
Beaver            1 1 0 0 2 1 3 3   Groundhog         1 1 0 0 2 1 3 3
Gray Squirrel     1 1 0 0 1 1 3 3   House Mouse       1 1 0 0 0 0 3 3
Porcupine         1 1 0 0 1 1 3 3   Wolf              3 3 1 1 4 4 2 3
Bear              3 3 1 1 4 4 2 3   Raccoon           3 3 1 1 4 4 3 2
Marten            3 3 1 1 4 4 1 2   Weasel            3 3 1 1 3 3 1 2
Wolverine         3 3 1 1 4 4 1 2   Badger            3 3 1 1 3 3 1 2
River Otter       3 3 1 1 4 3 1 2   Sea Otter         3 2 1 1 3 3 1 2
Jaguar            3 3 1 1 3 2 1 1   Cougar            3 3 1 1 3 2 1 1
Fur Seal          3 2 1 1 4 4 1 1   Sea Lion          3 2 1 1 4 4 1 1
Grey Seal         3 2 1 1 3 3 2 2   Elephant Seal     2 1 1 1 4 4 1 1
Reindeer          0 4 1 0 3 3 3 3   Elk               0 4 1 0 3 3 3 3
Deer              0 4 0 0 3 3 3 3   Moose             0 4 0 0 3 3 3 3
;
```

The following statements produce Output 40.4.1:

```
title3 'Raw Data';
proc cluster data=teeth method=average nonorm noeigen;
   var v1-v8;
   id mammal;
run;
```

Output 40.4.1 Average Linkage Analysis of Mammals' Teeth Data: Raw Data

Hierarchical Cluster Analysis of Mammals' Teeth Data
Evaluating the Effects of Ties
Raw Data

The CLUSTER Procedure
Average Linkage Cluster Analysis

Root-Mean-Square Total-Sample Standard Deviation	0.898027

Output 40.4.1 *continued*

Cluster History					
Number of Clusters	Clusters Joined		Freq	RMS Distance	Tie
31	Beaver	Groundhog	2	0	T
30	Gray Squirrel	Porcupine	2	0	T
29	Wolf	Bear	2	0	T
28	Marten	Wolverine	2	0	T
27	Weasel	Badger	2	0	T
26	Jaguar	Cougar	2	0	T
25	Fur Seal	Sea Lion	2	0	T
24	Reindeer	Elk	2	0	T
23	Deer	Moose	2	0	
22	Brown Bat	Silver Hair Bat	2	1	T
21	Pigmy Bat	House Bat	2	1	T
20	Pika	Rabbit	2	1	T
19	CL31	CL30	4	1	T
18	CL28	River Otter	3	1	T
17	CL27	Sea Otter	3	1	T
16	CL24	CL23	4	1	
15	CL21	Red Bat	3	1.2247	
14	CL17	Grey Seal	4	1.291	
13	CL29	Raccoon	3	1.4142	T
12	CL25	Elephant Seal	3	1.4142	
11	CL18	CL14	7	1.5546	
10	CL22	CL15	5	1.5811	
9	CL20	CL19	6	1.8708	T
8	CL11	CL26	9	1.9272	
7	CL8	CL12	12	2.2278	
6	Mole	CL13	4	2.2361	
5	CL9	House Mouse	7	2.4833	
4	CL6	CL7	16	2.5658	
3	CL10	CL16	9	2.8107	
2	CL3	CL5	16	3.7054	
1	CL2	CL4	32	4.2939	

The following statements produce Output 40.4.2:

```
title3 'Standardized Data';
proc cluster data=teeth std method=average nonorm noeigen;
   var v1-v8;
   id mammal;
run;
```

Output 40.4.2 Average Linkage Analysis of Mammals' Teeth Data: Standardized Data

Hierarchical Cluster Analysis of Mammals' Teeth Data
Evaluating the Effects of Ties
Standardized Data

The CLUSTER Procedure
Average Linkage Cluster Analysis

The data have been standardized to mean 0 and variance 1

Root-Mean-Square Total-Sample Standard Deviation	1

Cluster History

Number of Clusters	Clusters Joined		Freq	RMS Distance	Tie
31	Beaver	Groundhog	2	0	T
30	Gray Squirrel	Porcupine	2	0	T
29	Wolf	Bear	2	0	T
28	Marten	Wolverine	2	0	T
27	Weasel	Badger	2	0	T
26	Jaguar	Cougar	2	0	T
25	Fur Seal	Sea Lion	2	0	T
24	Reindeer	Elk	2	0	T
23	Deer	Moose	2	0	
22	Pigmy Bat	Red Bat	2	0.9157	
21	CL28	River Otter	3	0.9169	
20	CL31	CL30	4	0.9428	T
19	Brown Bat	Silver Hair Bat	2	0.9428	T
18	Pika	Rabbit	2	0.9428	
17	CL27	Sea Otter	3	0.9847	
16	CL22	House Bat	3	1.1437	
15	CL21	CL17	6	1.3314	
14	CL25	Elephant Seal	3	1.3447	
13	CL19	CL16	5	1.4688	
12	CL15	Grey Seal	7	1.6314	
11	CL29	Raccoon	3	1.692	
10	CL18	CL20	6	1.7357	
9	CL12	CL26	9	2.0285	
8	CL24	CL23	4	2.1891	
7	CL9	CL14	12	2.2674	
6	CL10	House Mouse	7	2.317	
5	CL11	CL7	15	2.6484	
4	CL13	Mole	6	2.8624	
3	CL4	CL8	10	3.5194	
2	CL3	CL6	17	4.1265	
1	CL2	CL5	32	4.7753	

There are ties at 16 levels for the raw data but at only 10 levels for the standardized data. There are more ties for the raw data because the increments between successive values are the same for all of the raw variables

but different for the standardized variables.

One way to assess the importance of the ties in the analysis is to repeat the analysis on several random permutations of the observations and then to see to what extent the results are consistent at the interesting levels of the cluster history. Three macros are presented to facilitate this process, as follows.

```
/* ---------------------------------------------------------- */
/*                                                            */
/* The macro CLUSPERM randomly permutes observations and      */
/* does a cluster analysis for each permutation.              */
/* The arguments are as follows:                              */
/*                                                            */
/*    data    data set name                                   */
/*    var     list of variables to cluster                    */
/*    id      id variable for proc cluster                    */
/*    method  clustering method (and possibly other options) */
/*    nperm   number of random permutations.                  */
/*                                                            */
/* ---------------------------------------------------------- */
%macro CLUSPERM(data,var,id,method,nperm);

   /* ------CREATE TEMPORARY DATA SET WITH RANDOM NUMBERS------ */
   data _temp_(drop=i);
      set &data;
      array _random_ _ran_1-_ran_&nperm;
      do i = 1 to dim(_random_);
         _random_[i]=ranuni(835297461);
      end;
   run;

   /* ------PERMUTE AND CLUSTER THE DATA----------------------- */
   %do n=1 %to &nperm;
      proc sort data=_temp_(keep=_ran_&n &var &id) out=_perm_;
         by _ran_&n;
      run;

      proc cluster method=&method noprint outtree=_tree_&n;
         var &var;
         id &id;
      run;
   %end;
%mend;
/* ---------------------------------------------------------- */
/*                                                            */
/* The macro PLOTPERM plots various cluster statistics        */
/* against the number of clusters for each permutation.       */
/* The arguments are as follows:                              */
/*                                                            */
/*    nclus   maximum number of clusters to be plotted        */
/*    nperm   number of random permutations.                  */
/*                                                            */
/* ---------------------------------------------------------- */
%macro PLOTPERM(nclus,nperm);

   /* ---CONCATENATE TREE DATA SETS FOR 20 OR FEWER CLUSTERS--- */
```

```
   data _plot_;
      set %do n=1 %to &nperm; _tree_&n(in=_in_&n) %end;;
      if _ncl_<=&nclus;
      %do n=1 %to &nperm;
         if _in_&n then _perm_=&n;
      %end;
      label _perm_='permutation number';
      keep _ncl_ _psf_ _pst2_ _ccc_ _perm_;
   run;

   /* ---PLOT THE REQUESTED STATISTICS BY NUMBER OF CLUSTERS--- */
   proc sgscatter;
      compare y=(_ccc_ _psf_ _pst2_) x=_ncl_ /group=_perm_;
      label _ccc_ = 'CCC' _psf_ = 'Pseudo F' _pst2_ = 'Pseudo T-Squared';
   run;
%mend;

/* ---------------------------------------------------------- */
/*                                                            */
/* The macro TABPERM generates cluster-membership variables  */
/* for a specified number of clusters for each permutation.  */
/* PROC TABULATE gives the frequencies and means.            */
/* The arguments are as follows:                             */
/*                                                            */
/*    var     list of variables to cluster                   */
/*            (no "-" or ":" allowed)                        */
/*    id      id variable for proc cluster                   */
/*    meanfmt format for printing means in PROC TABULATE     */
/*    nclus   number of clusters desired                     */
/*    nperm   number of random permutations.                 */
/*                                                            */
/* ---------------------------------------------------------- */
%macro TABPERM(var,id,meanfmt,nclus,nperm);

   /* ------CREATE DATA SETS GIVING CLUSTER MEMBERSHIP--------- */
   %do n=1 %to &nperm;
      proc tree data=_tree_&n noprint n=&nclus
                out=_out_&n(drop=clusname
                            rename=(cluster=_clus_&n));
         copy &var;
         id &id;
      run;

      proc sort;
         by &id &var;
      run;
   %end;

   /* ------MERGE THE CLUSTER VARIABLES------------------------ */
   data _merge_;
      merge
         %do n=1 %to &nperm;
            _out_&n
         %end;;
```

```
      by &id &var;
      length all_clus $ %eval(3*&nperm);
      %do n=1 %to &nperm;
         substr( all_clus, %eval(1+(&n-1)*3), 3) =
            put( _clus_&n, 3.);
      %end;
   run;

   /* ------ TABULATE CLUSTER COMBINATIONS------------ */
   proc sort;
      by _clus_:;
   run;
   proc tabulate order=data formchar='           ';
      class all_clus;
      var &var;
      table all_clus, n='FREQ'*f=5. mean*f=&meanfmt*(&var) /
         rts=%eval(&nperm*3+1);
   run;
%mend;
```

To use these macros, it is first convenient to define a macro variable, VLIST, listing the teeth variables, since the forms V1-V8 or V: cannot be used with the TABULATE procedure in the TABPERM macro:

```
/* -TABULATE does not accept hyphens or colons in VAR lists- */
%let vlist=v1 v2 v3 v4 v5 v6 v7 v8;
```

The CLUSPERM macro is then called to analyze 10 random permutations. The PLOTPERM macro plots the pseudo F and t^2 statistics and the cubic clustering criterion. Since the data are discrete, the pseudo F statistic and the cubic clustering criterion can be expected to increase as the number of clusters increases, so local maxima or large jumps in these statistics are more relevant than the global maximum in determining the number of clusters. For the raw data, only the pseudo t^2 statistic indicates the possible presence of clusters, with the four-cluster level being suggested. Hence, the macros are used as follows to analyze the results at the four-cluster level:

```
title3 'Raw Data';

/* ------CLUSTER RAW DATA WITH AVERAGE LINKAGE-------------- */
%clusperm( teeth, &vlist, mammal, average, 10);
```

The following statements produce Output 40.4.3.

```
/* -----PLOT STATISTICS FOR THE LAST 20 LEVELS-------------- */
%plotperm(20, 10);
```

Output 40.4.3 Analysis of 10 Random Permutations of Raw Mammals' Teeth Data

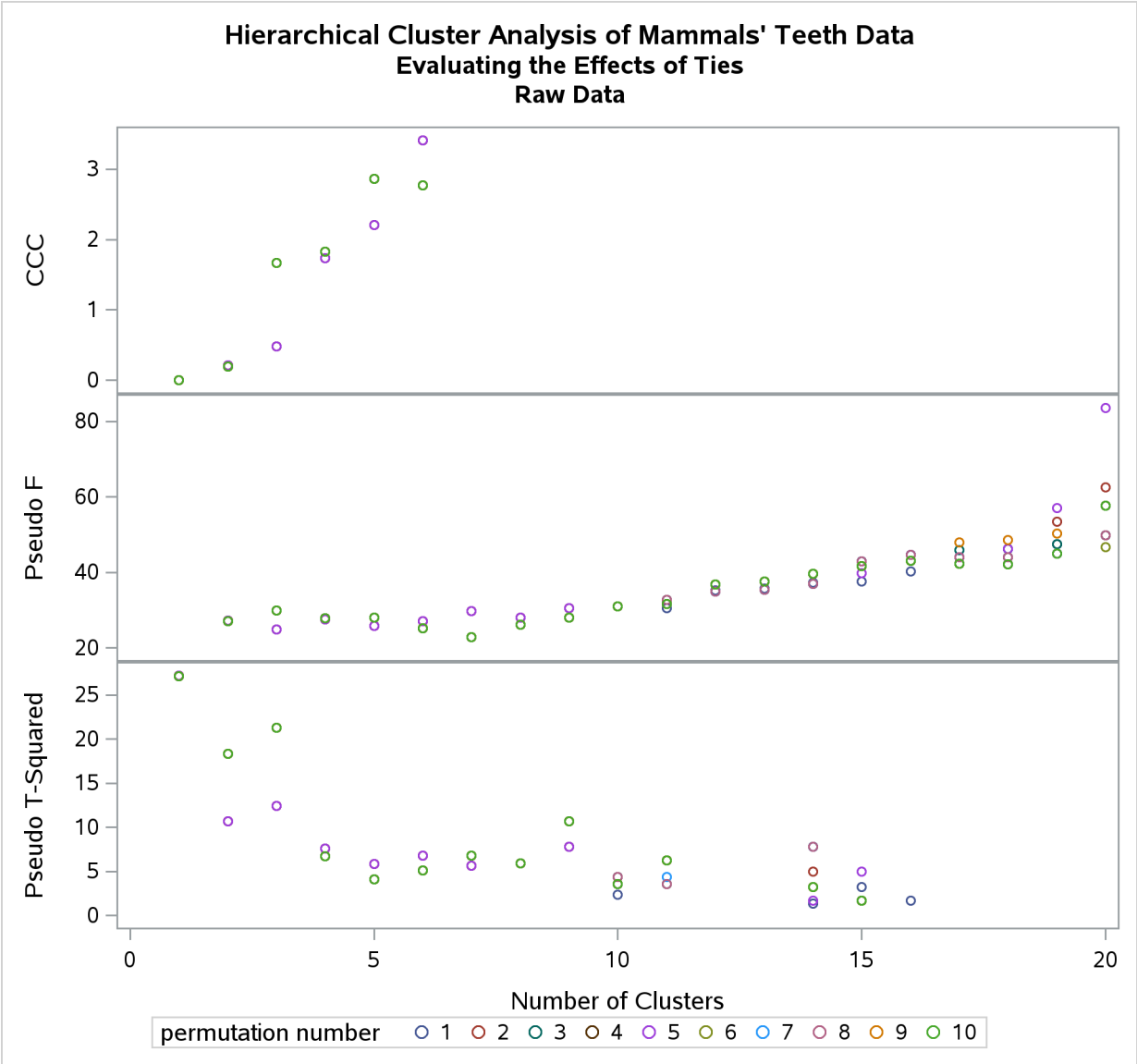

The following statements produce Output 40.4.4.

```
/* ------ANALYZE THE 4-CLUSTER LEVEL------------------------ */
%tabperm( &vlist, mammal, 9.1, 4, 10);
```

Output 40.4.4 Raw Mammals' Teeth Data: Indeterminacy at the Four-Cluster Level

Hierarchical Cluster Analysis of Mammals' Teeth Data
Evaluating the Effects of Ties
Raw Data

		Mean							
all_clus	FREQ	Top incisors	Bottom incisors	Top canines	Bottom canines	Top premolars	Bottom premolars	Top molars	Bottom molars
1 3 1 1 1 3 3 3 2 3	4	0.0	4.0	0.5	0.0	3.0	3.0	3.0	3.0
2 2 2 2 2 2 1 2 1 1	15	2.9	2.6	1.0	1.0	3.6	3.4	1.3	1.8
2 4 2 2 4 2 1 2 1 1	1	3.0	2.0	1.0	0.0	3.0	3.0	3.0	3.0
3 1 3 3 3 1 2 1 3 2	5	1.0	1.0	0.0	0.0	1.2	0.8	3.0	3.0
3 4 3 3 4 1 2 1 3 2	2	2.0	1.0	0.0	0.0	2.5	2.0	3.0	3.0
4 4 4 4 4 4 4 4 4 4	5	1.8	3.0	1.0	1.0	2.0	2.4	3.0	3.0

From the TABULATE output, you can see that two types of clustering are obtained. In one case, the mole is grouped with the carnivores, while the pika and rabbit are grouped with the rodents. In the other case, both the mole and the lagomorphs are grouped with the bats.

Next, the analysis is repeated with the standardized data as shown in the following statements. The pseudo F and t^2 statistics indicate three or four clusters, while the cubic clustering criterion shows a sharp rise up to four clusters and then levels off up to six clusters. So the TABPERM macro is used again at the four-cluster level. In this case, there is no indeterminacy, because the same four clusters are obtained with every permutation, although in different orders. It must be emphasized, however, that lack of indeterminacy in no way indicates validity.

```
title3 'Standardized Data';

/*------CLUSTER STANDARDIZED DATA WITH AVERAGE LINKAGE------*/
%clusperm( teeth, &vlist, mammal, average std, 10);
```

The following statements produce Output 40.4.5.

```
/* -----PLOT STATISTICS FOR THE LAST 20 LEVELS-------------- */
%plotperm(20, 10);
```

Output 40.4.5 Analysis of 10 Random Permutations of Standardized Mammals' Teeth Data

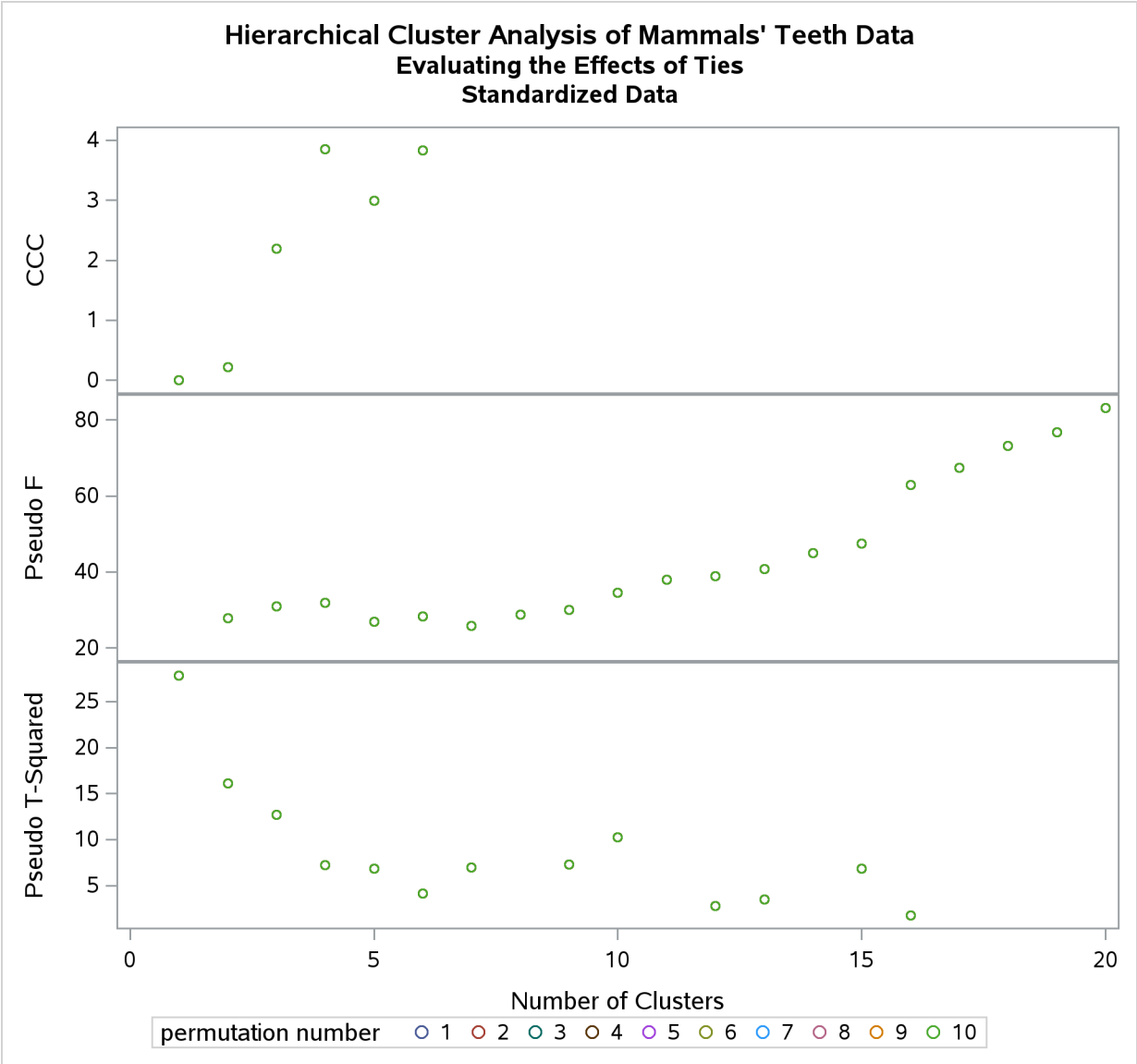

The following statements produce Output 40.4.6.

```
/* ------ANALYZE THE 4-CLUSTER LEVEL------------------------ */
%tabperm( &vlist, mammal, 9.1, 4, 10);
```

Output 40.4.6 Standardized Mammals' Teeth Data: No Indeterminacy at the Four-Cluster Level

Hierarchical Cluster Analysis of Mammals' Teeth Data
Evaluating the Effects of Ties
Standardized Data

		Mean							
all_clus	FREQ	Top incisors	Bottom incisors	Top canines	Bottom canines	Top premolars	Bottom premolars	Top molars	Bottom molars
1 3 1 1 1 3 3 3 2 3	4	0.0	4.0	0.5	0.0	3.0	3.0	3.0	3.0
2 2 2 2 2 2 1 2 1 1	15	2.9	2.6	1.0	1.0	3.6	3.4	1.3	1.8
3 1 3 3 3 1 2 1 3 2	7	1.3	1.0	0.0	0.0	1.6	1.1	3.0	3.0
4 4 4 4 4 4 4 4 4 4	6	2.0	2.8	1.0	0.8	2.2	2.5	3.0	3.0

References

Anderberg, M. R. (1973). *Cluster Analysis for Applications*. New York: Academic Press.

Batagelj, V. (1981). "Note on Ultrametric Hierarchical Clustering Algorithms." *Psychometrika* 46:351–352.

Blackith, R. E., and Reyment, R. A. (1971). *Multivariate Morphometrics*. London: Academic Press.

Blashfield, R. K., and Aldenderfer, M. S. (1978). "The Literature on Cluster Analysis." *Multivariate Behavioral Research* 13:271–295.

Caliński, T., and Harabasz, J. (1974). "A Dendrite Method for Cluster Analysis." *Communications in Statistics—Theory and Methods* 3:1–27.

Cooper, M. C., and Milligan, G. W. (1988). "The Effect of Error on Determining the Number of Clusters." In *Data, Expert Knowledge, and Decisions*, edited by W. Gaul and M. Schrader, 319–328. London: Springer-Verlag.

Duda, R. O., and Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. New York: John Wiley & Sons.

Everitt, B. S. (1980). *Cluster Analysis*. 2nd ed. London: Heineman Educational Books.

Fisher, L., and Van Ness, J. W. (1971). "Admissible Clustering Procedures." *Biometrika* 58:91–104.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Florek, K., Lukaszewicz, J., Perkal, J., and Zubrzycki, S. (1951a). "Sur la liaison et la division des points d'un ensemble fini [On the linking and dividing of points in a finite set]." *Colloquium Mathematicae* 2:282–285.

Florek, K., Lukaszewicz, J., Perkal, J., and Zubrzycki, S. (1951b). "Taksonomia Wroclawska [Wroclaw's taxonomy]." *Przeglad Antropologiczny* 17:193–211.

Gower, J. C. (1967). "A Comparison of Some Methods of Cluster Analysis." *Biometrics* 23:623–637.

Hamer, R. M., and Cunningham, J. W. (1981). "Cluster Analyzing Profile Data with Interrater Differences: A Comparison of Profile Association Measures." *Applied Psychological Measurement* 5:63–72.

Hartigan, J. A. (1975). *Clustering Algorithms*. New York: John Wiley & Sons.

Hartigan, J. A. (1977). "Distribution Problems in Clustering." In *Classification and Clustering*, edited by J. V. Ryzin, 45–72. New York: Academic Press.

Hartigan, J. A. (1981). "Consistency of Single Linkage for High-Density Clusters." *Journal of the American Statistical Association* 76:388–394.

Hawkins, D. M., Muller, M. W., and ten Krooden, J. A. (1982). "Cluster Analysis." In *Topics in Applied Multivariate Analysis*, edited by D. M. Hawkins, 303–356. Cambridge: Cambridge University Press.

Jardine, N., and Sibson, R. (1971). *Mathematical Taxonomy*. New York: John Wiley & Sons.

Johnson, S. C. (1967). "Hierarchical Clustering Schemes." *Psychometrika* 32:241–254.

Lance, G. N., and Williams, W. T. (1967). "A General Theory of Classificatory Sorting Strategies, Part I: Hierarchical Systems." *Computer Journal* 9:373–380.

Massart, D. L., and Kaufman, L. (1983). *The Interpretation of Analytical Chemical Data by the Use of Cluster Analysis*. New York: John Wiley & Sons.

McQuitty, L. L. (1957). "Elementary Linkage Analysis for Isolating Orthogonal and Oblique Types and Typal Relevancies." *Educational and Psychological Measurement* 17:207–229.

McQuitty, L. L. (1966). "Similarity Analysis by Reciprocal Pairs for Discrete and Continuous Data." *Educational and Psychological Measurement* 26:825–831.

Mezzich, J. E., and Solomon, H. (1980). *Taxonomy and Behavioral Science*. New York: Academic Press.

Milligan, G. W. (1979). "Ultrametric Hierarchical Clustering Algorithms." *Psychometrika* 44:343–346.

Milligan, G. W. (1980). "An Examination of the Effect of Six Types of Error Perturbation on Fifteen Clustering Algorithms." *Psychometrika* 45:325–342.

Milligan, G. W. (1987). "A Study of the Beta-Flexible Clustering Method." College of Administrative Science Working Paper Series, No. 87-61, Ohio State University.

Milligan, G. W., and Cooper, M. C. (1985). "An Examination of Procedures for Determining the Number of Clusters in a Data Set." *Psychometrika* 50:159–179.

Milligan, G. W., and Cooper, M. C. (1987). "A Study of Variable Standardization." College of Administrative Science Working Paper Series, No. 87-63, Ohio State University.

Rouncefield, M. (1995). "The Statistics of Poverty and Inequality." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Sarle, W. S. (1983). *Cubic Clustering Criterion*. Technical Report A-108, SAS Institute Inc., Cary, NC.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. New York: Chapman & Hall.

Sneath, P. H. A. (1957). "The Application of Computers to Taxonomy." *Journal of General Microbiology* 17:201–226.

Sneath, P. H. A., and Sokal, R. R. (1973). *Numerical Taxonomy*. San Francisco: W. H. Freeman.

Sokal, R. R., and Michener, C. D. (1958). "A Statistical Method for Evaluating Systematic Relationships." *University of Kansas Science Bulletin* 38:1409–1438.

Sørensen, T. (1948). "A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content and Its Application to Analyses of the Vegetation on Danish Commons." *Biologiske Skrifter* 5:1–34.

Spath, H. (1980). *Cluster Analysis Algorithms*. Chichester, UK: Ellis Horwood.

Symons, M. J. (1981). "Clustering Criteria and Multivariate Normal Mixtures." *Biometrics* 37:35–43.

Ward, J. H. (1963). "Hierarchical Grouping to Optimize an Objective Function." *Journal of the American Statistical Association* 58:236–244.

Wishart, D. (1969). "Mode Analysis: A Generalisation of Nearest Neighbour Which Reduces Chaining Effects." In *Numerical Taxonomy*, edited by A. J. Cole, 282–311. London: Academic Press.

Wong, M. A. (1982). "A Hybrid Clustering Method for Identifying High-Density Clusters." *Journal of the American Statistical Association* 77:841–847.

Wong, M. A., and Lane, T. (1983). "A *k*th Nearest Neighbor Clustering Procedure." *Journal of the Royal Statistical Society, Series B* 45:362–368.

Wong, M. A., and Schaack, C. (1982). "Using the *k*th Nearest Neighbor Clustering Procedure to Determine the Number of Subpopulations." *Proceedings of the Statistical Computing Section, American Statistical Association* 40–48.

Chapter 41
The CORRESP Procedure

Contents

Overview: CORRESP Procedure **2650**
- Background 2650

Getting Started: CORRESP Procedure **2651**

Syntax: CORRESP Procedure **2653**
- PROC CORRESP Statement 2653
- BY Statement 2663
- ID Statement 2664
- SUPPLEMENTARY Statement 2664
- TABLES Statement 2664
- VAR Statement 2665
- WEIGHT Statement 2665

Details: CORRESP Procedure **2666**
- Input Data Set 2666
- Using the TABLES Statement 2666
- Using the VAR Statement 2674
- Missing and Invalid Data 2675
- Coding, Fuzzy Coding, and Doubling 2675
- Creating a Data Set Containing the Crosstabulation 2678
- Output Data Sets 2680
- Computational Resources 2682
- Algorithm and Notation 2683
- Displayed Output 2691
- ODS Table Names 2693
- ODS Graphics 2695

Examples: CORRESP Procedure **2695**
- Example 41.1: Simple and Multiple Correspondence Analysis of Automobiles and Their Owners 2695
- Example 41.2: Simple Correspondence Analysis of U.S. Population 2707
- Example 41.3: MCA and Large Data Sets 2712

References **2718**

Overview: CORRESP Procedure

The CORRESP procedure performs simple correspondence analysis and multiple correspondence analysis (MCA). You can use correspondence analysis to find a low-dimensional graphical representation of the rows and columns of a crosstabulation or contingency table. Each row and column is represented by a point in a plot determined from the cell frequencies. PROC CORRESP can also compute coordinates for supplementary rows and columns.

PROC CORRESP can read two kinds of input: raw categorical responses on two or more classification variables or a two-way contingency table. The correspondence analysis plot is displayed with ODS Graphics. For more information about ODS Graphics, see the section "ODS Graphics" on page 2695.

Background

Correspondence analysis is a popular data analysis method in France and Japan. In France, correspondence analysis was developed under the influence of Jean-Paul Benzécri; in Japan, it was developed under Chikio Hayashi. The name *correspondence analysis* is a translation of the French *analyse des correspondances*. The technique apparently has many independent beginnings (for example, Richardson and Kuder 1933; Hirshfield 1935; Horst 1935; Fisher 1940; Guttman 1941; Burt 1950; Hayashi 1950). It has had many other names, including optimal scaling, reciprocal averaging, optimal scoring, and appropriate scoring in the United States; quantification method in Japan; homogeneity analysis in the Netherlands; dual scaling in Canada; and scalogram analysis in Israel.

Correspondence analysis is described in more detail in French in Benzécri (1973) and Lebart, Morineau, and Tabard (1977). In Japanese, the subject is described in Komazawa (1982); Nishisato (1982); Kobayashi (1981). In English, correspondence analysis is described in Lebart, Morineau, and Warwick (1984); Greenacre (1984); Nishisato (1980); Tenenhaus and Young (1985); Gifi (1990); Greenacre and Hastie (1987), and many other sources. Hoffman and Franke (1986) offer a short, introductory treatment that uses examples from the field of market research.

Getting Started: CORRESP Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/crspgs.sas.)

Data are available containing the numbers of Ph.D.'s awarded in the United States during the years 1973 through 1978 (US Bureau of the Census 1979). The table has six rows, one for each of six academic disciplines, and six columns for the six years.

The following DATA step reads the complete table into a SAS data set, and PROC CORRESP displays correspondence analysis results including the inertia decomposition and coordinates:

```
title "Number of Ph.D.'s Awarded from 1973 to 1978";

data PhD;
   input Science $ 1-19 y1973-y1978;
   label y1973 = '1973'
         y1974 = '1974'
         y1975 = '1975'
         y1976 = '1976'
         y1977 = '1977'
         y1978 = '1978';
   datalines;
Life Sciences       4489 4303 4402 4350 4266 4361
Physical Sciences   4101 3800 3749 3572 3410 3234
Social Sciences     3354 3286 3344 3278 3137 3008
Behavioral Sciences 2444 2587 2749 2878 2960 3049
Engineering         3338 3144 2959 2791 2641 2432
Mathematics         1222 1196 1149 1003  959  959
;

ods graphics on;

proc corresp data=PhD out=Results short chi2p;
   var y1973-y1978;
   id Science;
run;
```

The results are displayed in Figure 41.1 and Figure 41.2.

Figure 41.1 Inertia and Chi-Square Decomposition

Inertia and Chi-Square Decomposition

Singular Value	Principal Inertia	Chi-Square	Percent	Cumulative Percent
0.05845	0.00342	368.653	96.04	96.04
0.00861	0.00007	7.995	2.08	98.12
0.00694	0.00005	5.197	1.35	99.48
0.00414	0.00002	1.852	0.48	99.96
0.00122	0.00000	0.160	0.04	100.00
	0.00356	383.856	100.00	

Pr > Chisq < .0001 DF = 25

The concept of *inertia* in correspondence analysis is analogous to the concept of variance in principal component analysis, and it is proportional to the chi-square information.

In Figure 41.1, the total chi-square statistic, which is a measure of the association between the rows and columns in the full five dimensions of the (centered) table, is 383.856. The chi-square p-value, < 0.001, is displayed when you specify the CHI2P option. The maximum number of dimensions (or axes) is the minimum of the number of rows and columns, minus one. More than 96% of the total chi-square and inertia is explained by the first dimension, indicating that the association between the row and column categories is essentially one-dimensional. The plot in Figure 41.2 shows how the number of doctorates in the different disciplines changes over time. The plot shows that the number of doctorates in the behavioral sciences is associated with later years, and the number of doctorates in mathematics and engineering is associated with earlier years. This is consistent with the data that show that the number of doctorates in the behavioral sciences is increasing, the number of doctorates in every other discipline is decreasing, and the rate of decrease is greatest for mathematics and engineering.

Figure 41.2 Correspondence Analysis of Ph.D. Data

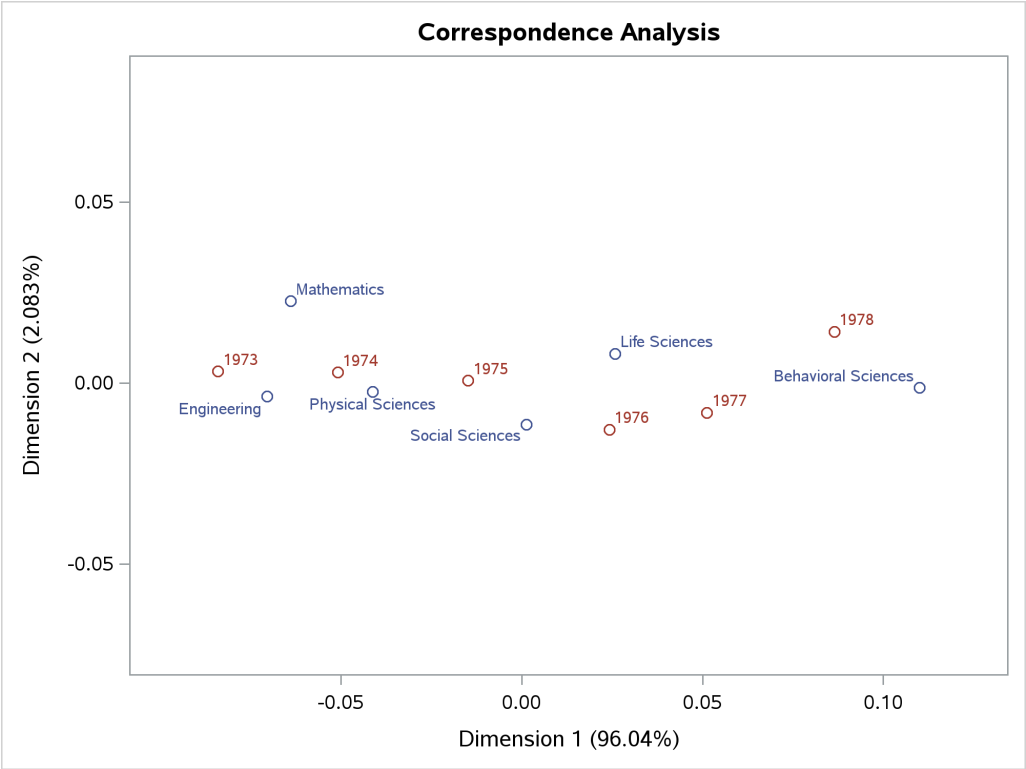

Syntax: CORRESP Procedure

The following statements are available in the CORRESP procedure.

PROC CORRESP < *options* > ;
  TABLES < *row-variables*, > *column-variables* ;
  VAR *variables* ;
  BY *variables* ;
  ID *variable* ;
  SUPPLEMENTARY *variables* ;
  WEIGHT *variable* ;

There are two separate forms of input to PROC CORRESP. One form is specified in the TABLES statement, the other in the VAR statement. You must specify either the TABLES or the VAR statement, but not both, each time you run PROC CORRESP.

Specify the TABLES statement if you are using raw, categorical data, the levels of which define the rows and columns of a table.

Specify the VAR statement if your data are already in tabular form. PROC CORRESP is generally more efficient with VAR statement input than with TABLES statement input.

The other statements are optional. All of the statements are explained, in alphabetical order, following the PROC CORRESP statement. All of the options in PROC CORRESP can be abbreviated to their first three letters, except for the OUTF= option. This is a special feature of PROC CORRESP and is not generally true of SAS/STAT procedures.

PROC CORRESP Statement

PROC CORRESP < *options* > ;

The PROC CORRESP statement invokes the CORRESP procedure. Table 41.1 summarizes the *options* available in the PROC CORRESP statement. These *options* are described following the table.

Table 41.1 Summary of PROC CORRESP Statement Options

Option	Description
Data Set Options	
DATA=	Specifies input SAS data set
OUTC=	Specifies output coordinate SAS data set
OUTF=	Specifies output frequency SAS data set
Row and Column Coordinates	
DIMENS=	Specifies the number of dimensions or axes
MCA	Performs multiple correspondence analysis
PROFILE=	Standardizes the row and column coordinates
Table Construction	
BINARY	Specifies binary table

Table 41.1 *continued*

Option	Description
BMIM	Specifies in-memory Burt matrix construction
CROSS=	Specifies cross levels of TABLES variables
FREQOUT	Specifies input data in PROC FREQ output
MISSING	Includes observations with missing values
Control Displayed Output	
ALL	Displays all output
BENZECRI	Displays inertias adjusted by Benzécri's method
CELLCHI2	Displays cell contributions to chi-square
CHI2P	Displays the chi-square *p*-value
CP	Displays column profile matrix
DEVIATION	Displays observed minus expected values
EXPECTED	Displays chi-square expected values
GREENACRE	Displays inertias adjusted by Greenacre's method
INERTIATABLE	Displays the inertia and chi-square decomposition in tabular form
NOCOLUMN=	Suppresses the display of column coordinates
NOPRINT	Suppresses the display of all output
NOROW=	Suppresses the display of row coordinates
OBSERVED	Displays contingency table of observed frequencies
PLOTS=	Specifies ODS Graphics details
PRINT=	Displays percentages or frequencies
RP	Displays row profile matrix
SHORT	Suppresses all point and coordinate statistics
UNADJUSTED	Displays unadjusted inertias
Other Options	
COLUMN=	Specifies esoteric column coordinate standardizations
MININERTIA=	Specifies minimum inertia
NVARS=	Specifies number of classification variables
ROW=	Specifies esoteric row coordinate standardizations
SINGULAR=	Specifies effective zero
SOURCE	Includes level source in the OUTC= data set

The display options control the amount of displayed output. The CELLCHI2, EXPECTED, DEVIATION, and CHI2P options display additional chi-square information. See the "Details: CORRESP Procedure" section for more information. The unit of the matrices displayed by the CELLCHI2, CP, DEVIATION, EXPECTED, OBSERVED, and RP options depends on the value of the PRINT= option. The table construction options control the construction of the contingency table; these options are valid only when you also specify a TABLES statement.

You can specify the following *options* in the PROC CORRESP statement. They are listed in alphabetical order.

ALL

is equivalent to specifying the OBSERVED, RP, CP, CELLCHI2, EXPECTED, and DEVIATION options. Specifying the ALL option does not affect the PRINT= option. Therefore, only frequencies (not percentages) for these options are displayed unless you specify otherwise with the PRINT= option.

BENZECRI

BEN

displays adjusted inertias when you are performing multiple correspondence analysis. By default, unadjusted inertias (the usual inertias from multiple correspondence analysis) are displayed. However, adjusted inertias that use a method proposed by Benzécri (1979) and described by Greenacre (1984, p. 145) can be displayed by specifying the BENZECRI option. Specify the UNADJUSTED option to output the usual table of unadjusted inertias as well. For more information, see the section "MCA Adjusted Inertias" on page 2687.

BINARY

enables you to create binary tables easily. When you specify the BINARY option, specify only column variables in the TABLES statement. Each input data set observation forms a single row in the constructed table.

BMIM

enables you to create a Burt matrix for MCA by using an in-memory algorithm and a single pass through the data. The default Burt matrix construction method might use a utility file, which can be slow for large data sets. The algorithm that this option uses is usually faster than the default algorithm, particularly for large data sets. However, it does not sort the levels (the Burt matrix rows and columns) within the categorical variables.

CELLCHI2

CEL

displays the contribution to the total chi-square test statistic for each cell. See also the descriptions of the DEVIATION, EXPECTED, and OBSERVED options.

CHI2P

CHI

displays the chi-square p-value in the inertia and chi-square decomposition table. The chi-square p-value is not displayed by default because in many cases the table being analyzed is not a true two-way contingency table.

COLUMN=B | BD | DB | DBD | DBD1/2 | DBID1/2

COL=B | BD | DB | DBD | DBD1/2 | DBID1/2

provides other standardizations of the column coordinates. The COLUMN= option is rarely needed. Typically, you should use the PROFILE= option instead (see the section "The PROFILE=, ROW=, and COLUMN= Options" on page 2684). By default, COLUMN=DBD.

CP

displays the column profile matrix. Column profiles contain the observed conditional probabilities of row membership given column membership. See also the RP option.

CROSS=BOTH | COLUMN | NONE | ROW

CRO=BOT | COL | NON | ROW

specifies the method of crossing (factorially combining) the levels of the TABLES variables. The default is CROSS=NONE.

NONE causes each level of every row variable to become a row label and each level of every column variable to become a column label.

ROW causes each combination of levels for all row variables to become a row label, whereas each level of every column variable becomes a column label.

COLUMN causes each combination of levels for all column variables to become a column label, whereas each level of every row variable becomes a row label.

BOTH causes each combination of levels for all row variables to become a row label and each combination of levels for all column variables to become a column label.

The section "TABLES Statement" on page 2664 provides a more detailed description of this option.

DATA=*SAS-data-set*

specifies the SAS data set to be used by PROC CORRESP. If you do not specify the DATA= option, PROC CORRESP uses the most recently created SAS data set.

DEVIATION

DEV

displays the matrix of deviations between the observed frequency matrix and the product of its row marginals and column marginals divided by its grand frequency. For ordinary two-way contingency tables, these are the observed minus expected frequencies under the hypothesis of row and column independence and are components of the chi-square test statistic. See also the CELLCHI2, EXPECTED, and OBSERVED options.

DIMENS=*n*

DIM=*n*

specifies the number of dimensions or axes to use. The default is DIMENS=2. The maximum value of the DIMENS= option in an $(n_r \times n_c)$ table is $n_r - 1$ or $n_c - 1$, whichever is smaller. For example, in a table with 4 rows and 5 columns, the maximum specification is DIMENS=3. If your table has 2 rows or 2 columns, specify DIMENS=1.

EXPECTED

EXP

displays the product of the row marginals and the column marginals divided by the grand frequency of the observed frequency table. For ordinary two-way contingency tables, these are the expected frequencies under the hypothesis of row and column independence and are components of the chi-square test statistic. In other situations, this interpretation is not strictly valid. See also the CELLCHI2, DEVIATION, and OBSERVED options.

FREQOUT

FRE

indicates that the PROC CORRESP input data set has the same form as an output data set from the FREQ procedure, even if it was not directly produced by PROC FREQ. The FREQOUT option enables PROC CORRESP to take shortcuts in constructing the contingency table.

When you specify the FREQOUT option, you must also specify a WEIGHT statement. The cell frequencies in a PROC FREQ output data set are contained in a variable called COUNT, so specify COUNT in a WEIGHT statement with PROC CORRESP. The FREQOUT option might produce unexpected results if the DATA= data set is structured incorrectly. Each of the two variable lists specified in the TABLES statement must consist of a single variable, and observations must be grouped by the levels of the row variable and then by the levels of the column variable. It is not required that the observations be sorted by the row variable and column variable, but they must be grouped consistently. There must be as many observations in the input data set (or BY group) as there are cells in the completed contingency table. Zero cells must be specified with zero weights. When you use PROC FREQ to create the PROC CORRESP input data set, you must specify the SPARSE option in the FREQ procedure's TABLES statement so that the zero cells are written to the output data set.

GREENACRE

GRE

displays adjusted inertias when you are performing multiple correspondence analysis. By default, unadjusted inertias (the usual inertias from multiple correspondence analysis) are displayed. However, adjusted inertias that use a method proposed by Greenacre (1984, p. 156) can be displayed by specifying the GREENACRE option. Specify the UNADJUSTED option to output the usual table of unadjusted inertias as well. For more information, see the section "MCA Adjusted Inertias" on page 2687.

INERTIATABLE

INE

displays the inertia and chi-square decomposition table in addition to the inertia and chi-square decomposition chart when ODS Graphics is enabled. This table is produced by default when ODS Graphics is not enabled or when the chart is not produced. When ODS Graphics is enabled:

- By default, the chart is produced and the table is not produced.
- Specify the PLOTS(ONLY)=CONFIGURATION option to produce the table but not the chart.
- Specify the INERTIATABLE option if you want to see the table in addition to the chart.[1]

MCA

requests a multiple correspondence analysis. This option requires that the input table be a Burt table, which is a symmetric matrix of crosstabulations among several categorical variables. If you specify the MCA option and a VAR statement, you must also specify the NVARS= option, which gives the number of categorical variables that were used to create the table. With raw categorical data, if you want results for the individuals as well as the categories, use the BINARY option instead.

[1] The INERTIATABLE option controls whether the inertia table is displayed with the inertia chart, whereas the UNADJUSTED option controls whether the unadjusted inertia table or chart is displayed with the BENZECRI or GREENACRE adjusted table or chart.

MININERTIA=*n*

MIN=*n*

specifies the minimum inertia ($0 \leq n \leq 1$) used to create the "best" tables—the indicator of which points best explain the inertia of each dimension. By default, MININERTIA=0.8. For more information, see the section "Algorithm and Notation" on page 2683.

MISSING

MIS

specifies that observations with missing values for the TABLES statement variables are included in the analysis. Missing values are treated as a distinct level of each categorical variable. By default, observations with missing values are excluded from the analysis.

NOCOLUMN < = BOTH | DATA | PRINT >

NOC < = BOT | DAT | PRI >

suppresses the display of the column coordinates and statistics and omits them from the output coordinate data set.

BOTH suppresses all column information from both the SAS listing and the output data set. The NOCOLUMN option is equivalent to the option NOCOLUMN=BOTH.

DATA suppresses all column information from the output data set.

PRINT suppresses all column information from the SAS listing.

NOPRINT

NOP

suppresses the display of all output. This option is useful when you need only an output data set. This option disables the Output Delivery System (ODS), including ODS Graphics, for the duration of the PROC. For more information, see Chapter 23, "Using the Output Delivery System."

NOROW < = BOTH | DATA | PRINT >

NOR < = BOT | DAT | PRI >

suppresses the display of the row coordinates and statistics and omits them from the output coordinate data set.

BOTH suppresses all row information from both the SAS listing and the output data set. The NOROW option is equivalent to the option NOROW=BOTH.

DATA suppresses all row information from the output data set.

PRINT suppresses all row information from the SAS listing.

The NOROW option can be useful when the rows of the contingency table are replications.

NVARS=*n*

NVA=*n*

specifies the number of classification variables that were used to create the Burt table. For example, suppose the Burt table was originally created with the following statement:

```
tables a b c;
```

You must specify NVARS=3 to read the table with a VAR statement.

The NVARS= option is required when you specify both the MCA option and a VAR statement. (See the section "VAR Statement" on page 2665 for an example.)

OBSERVED
OBS

displays the contingency table of observed frequencies and its row, column, and grand totals. If you do not specify the OBSERVED or ALL option, the contingency table is not displayed.

OUTC=*SAS-data-set*
OUT=*SAS-data-set*

creates an output coordinate SAS data set to contain the row, column, supplementary observation, and supplementary variable coordinates. This data set also contains the masses, squared cosines, quality of each point's representation in the DIMENS=*n* dimensional display, relative inertias, partial contributions to inertia, and best indicators.

OUTF=*SAS-data-set*

creates an output frequency SAS data set to contain the contingency table, row, and column profiles, the expected values, and the observed minus expected values and contributions to the chi-square statistic.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*options*) > >
PLOTS < (*global-plot-options*) > < =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

specifies options that control the details of the plots. When you specify only one plot request, you can omit the parentheses around the plot request.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc corresp;
   tables Marital, Origin;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

By default, for simple correspondence analysis, PROC CORRESP prints the configuration of points consisting of the row coordinates and column coordinates. With MCA, only column coordinates are printed. The default plots ($y * x$) are Dim2 * Dim1, Dim3 * Dim1, Dim3 * Dim2, and so on. When you specify PLOTS(FLIP), the plots are Dim1 * Dim2, Dim1 * Dim3, Dim2 * Dim3, and so on.

The *global-plot-options* are as follows:

FLIP

FLI

flips or interchanges the X-axis and Y-axis dimensions.

ONLY

ONL

suppresses the default plots. Only plots that are specifically requested are displayed.

SOURCE

SOU

displays the levels that correspond to each TABLES statement variable in the same color and shows the source of each group of levels. This option is most useful with multiple correspondence analysis. For example, if Sex and Age are TABLES statement variables, then when you specify SOURCE, `'Male'` and `'Female'` are displayed in one color, and `'Old'` and `'Young'` are displayed in a different color. By default, color groups correspond to rows, supplementary rows, columns, and supplementary columns.

The *plot-requests* include the following:

ALL

produces all appropriate plots.

CONFIGURATION < (*configuration-options*) >

CONFIG < (*configuration-options*) >

CON < (*configuration-options*) >

produces the configuration plot. This plot is produced when ODS Graphics is enabled, unless you specify PLOTS(ONLY)=INERTIA.

PROC CORRESP has procedure options that appear in the PROC statement (but not as sub-options of the PLOTS= option) and *configuration-options* that appear in parentheses after PLOTS=CONFIGURATION. If you specify the BENZECRI procedure option, then percentages based on the Benzécri inertia adjustment are displayed in the MCA configuration plot axis labels and in the inertia and chi-square decomposition table. If you specify the GREENACRE procedure option and not the BENZECRI procedure option, then percentages based on the GREENACRE inertia adjustment are displayed in the MCA configuration plot axis labels. If you want any other combination of inertia adjustments, then you need to specify one of the *configuration-options* (BENZECRI, GREENACRE, or UNADJUSTED) to override the default that comes from the procedure option. By default, a footnote is displayed below the plot when adjusted inertias are used to compute the percentages for the axis labels. You can use the NOFOOTNOTE *configuration-option* to suppress the footnote. You can specify the following *configuration-options*:

BENZECRI

BEN

displays percentages based on the Benzécri inertia adjustment in the MCA configuration plot axis labels. This *configuration-option* is the default when you specify the BENZECRI procedure option.

GREENACRE

GRE

displays percentages based on the Greenacre inertia adjustment in the MCA configuration plot axis labels. This *configuration-option* is the default when you specify the GREENACRE procedure option but not the BENZECRI procedure option.

NOFOOTNOTE

NOF

suppresses the footnote below the MCA configuration plot that states which inertia adjustment is used. By default, a footnote is displayed below the graph when percentages based on adjusted inertias are displayed in the axis labels.

UNADJUSTED

UNA

displays percentages based on unadjusted inertias in the MCA configuration plot axis labels. This *configuration-option* is the default when you specify neither the GREENACRE nor BENZECRI procedure option.

INERTIA < (*inertia-options*) >

INE< (*inertia-options*) >

requests an inertia decomposition chart and specifies *inertia-options*. An inertia decomposition chart is created when ODS Graphics is enabled unless you specify PLOTS(ONLY)=CONFIGURATION.

Unlike most graphs, the height of the inertia decomposition chart can vary as a function of the number of dimensions that appear in the chart. You can specify the following *inertia-options* to control the height of the inertia decomposition chart:

COMPUTEHEIGHT=*a b* < *max* >

CH=*a b* < *max* >

specifies the constants for computing the height of the inertia decomposition chart. For n dimensions, intercept a, slope b, and maximum height *max*, the height is $\min(a + b\,(n + 1), max)$. By default, COMPUTEHEIGHT=130 15 1200. Thus, the default height in pixels is $\min(130 + 15(n + 1), 1200)$. The default unit is pixels, and you can use the UNIT= *inertia-option* to change the unit to inches or centimeters.

SETHEIGHT=*height*

SH=*height*

specifies the height of the inertia decomposition chart. By default, the height is based on the COMPUTEHEIGHT= option. The default unit is pixels, and you can use the UNIT= *inertia-option* to change the unit to inches or centimeters.

UNIT=PX | IN | CM

specifies the unit (pixels, inches, or centimeters) for the SETHEIGHT= and COMPUTEHEIGHT= *inertia-options*. Inches equals pixels divided by 96, and centimeters equals inches times 2.54. By default, UNIT=PX.

NONE
NON
suppresses all plots.

PRINT=BOTH | FREQ | PERCENT
PRI=BOT | FRE | PER
affects the OBSERVED, RP, CP, CELLCHI2, EXPECTED, and DEVIATION options. The default is PRINT=FREQ.

- The PRINT=FREQ option displays output in the appropriate raw or natural units. (That is, PROC CORRESP displays raw frequencies for the OBSERVED option, relative frequencies with row marginals of 1.0 for the RP option, and so on.)
- The PRINT=PERCENT option scales results to percentages for the display of the output. (All elements in the OBSERVED matrix sum to 100.0, the row marginals are 100.0 for the RP option, and so on.)
- The PRINT=BOTH option displays both percentages and frequencies.

PROFILE=BOTH | COLUMN | NONE | ROW
PRO=BOT | COL | NON | ROW
specifies the standardization for the row and column coordinates. The default is PROFILE=BOTH.

BOTH specifies a standard correspondence analysis, which jointly displays the principal row and column coordinates. Row coordinates are computed from the row profile matrix, and column coordinates are computed from the column profile matrix.

ROW specifies a correspondence analysis of the row profile matrix. The row coordinates are weighted centroids of the column coordinates.

COLUMN specifies a correspondence analysis of the column profile matrix. The column coordinates are weighted centroids of the row coordinates.

NONE is rarely needed. Row and column coordinates are the generalized singular vectors, without the customary standardizations.

ROW=A | AD | DA | DAD | DAD1/2 | DAID1/2
provides other standardizations of the row coordinates. The ROW= option is rarely needed. Typically, you should use the PROFILE= option instead (see the section "The PROFILE=, ROW=, and COLUMN= Options" on page 2684). By default, ROW=DAD.

RP
displays the row profile matrix. Row profiles contain the observed conditional probabilities of column membership given row membership. See also the CP option.

SHORT
SHO
suppresses the display of all point and coordinate statistics except the coordinates. The following information is suppressed: each point's mass, relative contribution to the total inertia, and quality of representation in the DIMENS=*n* dimensional display; the squared cosines of the angles between each axis and a vector from the origin to the point; the partial contributions of each point to the inertia of each dimension; and the best indicators.

SINGULAR=*n*

SIN=*n*

specifies the largest value that is considered to be within rounding error of zero. The default value is 1E–8. This parameter is used in checking for zero rows and columns, in checking Burt table diagonal sums for equality, in checking denominators before dividing, and so on. Typically, you should not assign a value outside the range 1E–6 to 1E–12.

SOURCE

SOU

adds the variable _VAR_, which contains the name or label of the variable corresponding to the current level, to the OUTC= and OUTF= data sets.

UNADJUSTED

UNA

displays unadjusted inertias when you are performing multiple correspondence analysis. By default, unadjusted inertias (the usual inertias from multiple correspondence analysis) are displayed. However, if adjusted inertias are requested by either the GREENACRE option or the BENZECRI option, then the unadjusted inertia table is not displayed unless the UNADJUSTED option is specified.[2] For more information, see the section "MCA Adjusted Inertias" on page 2687.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC CORRESP to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the CORRESP procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

[2]The UNADJUSTED option controls whether the unadjusted inertia table or chart is displayed with the BENZECRI or GREENACRE adjusted table or chart, whereas the INERTIATABLE option controls whether the inertia table is displayed with the inertia chart.

ID Statement

ID *variable* **;**

You specify the ID statement only in conjunction with the VAR statement. You cannot specify the ID statement when you use the TABLES statement or the MCA option. When you specify an ID variable, PROC CORRESP labels the rows of the tables with the ID values and places the ID variable in the output data set.

SUPPLEMENTARY Statement

SUPPLEMENTARY *variables* **;**

SUP *variables* **;**

The SUPPLEMENTARY statement specifies variables that are to be represented as points in the joint row and column space but that are not used in determining the locations of the other, active row and column points of the contingency table. Supplementary observations on supplementary variables are ignored in simple correspondence analysis but are needed to compute the squared cosines for multiple correspondence analysis. Variables that are specified in the SUPPLEMENTARY statement must also be specified in the TABLES or VAR statement.

When you specify a VAR statement, each SUPPLEMENTARY variable indicates one supplementary column of the table. Supplementary variables must be numeric with VAR statement input.

When you specify a TABLES statement, each SUPPLEMENTARY variable indicates a set of rows or columns of the table that is supplementary. Supplementary variables can be either character or numeric with TABLES statement input.

TABLES Statement

TABLES < *row-variables*, > *column-variables* **;**

The TABLES statement instructs PROC CORRESP to create a contingency table, Burt table, or binary table from the values of two or more categorical variables. The TABLES statement specifies classification variables that are used to construct the rows and columns of the contingency table. The variables can be either numeric or character. The variable lists in the TABLES statement and the CROSS= option together determine the row and column labels of the contingency table.

You can specify both row variables and column variables separated by a comma, or you can specify only column variables and no comma. If you do not specify row variables (that is, if you list variables but do not use the comma as a delimiter), then you should specify either the MCA or the BINARY option. With the MCA option, PROC CORRESP creates a Burt table, which is a crosstabulation of each variable with itself and every other variable. The Burt table is symmetric. With the BINARY option, PROC CORRESP creates a binary table, which consists of one row for each input data set observation and one column for each category of each TABLES statement variable. If the binary matrix is $\mathbf{Z}$, then the Burt table is $\mathbf{Z}'\mathbf{Z}$. Specifying the BINARY option with the NOROWS option produces the same results as specifying the MCA option (except for the chi-square statistics).

See Figure 41.6 for an example or see the section "The MCA Option" on page 2686 for a detailed description of Burt tables.

You can use the WEIGHT statement with the TABLES statement to read category frequencies. Specify the SUPPLEMENTARY statement to name variables with categories that are supplementary rows or columns. You cannot specify the ID or VAR statement with the TABLES statement. See the section "Using the TABLES Statement" on page 2666 for an example.

VAR Statement

VAR *variables* **;**

You should specify the VAR statement when your data are in tabular form. The VAR variables must be numeric. The VAR statement instructs PROC CORRESP to read an existing contingency table, binary indicator matrix, fuzzy-coded indicator matrix, or Burt table, rather than raw data. See the section "Algorithm and Notation" on page 2683 for a description of a binary indicator matrix and a fuzzy-coded indicator matrix.

You can specify the WEIGHT statement with the VAR statement to read category frequencies and designate supplementary rows. Specify the SUPPLEMENTARY statement to name supplementary variables. You cannot specify the TABLES statement with the VAR statement.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement specifies weights for each observation and indicates supplementary observations for simple correspondence analyses with VAR statement input. You can include only one WEIGHT statement, and the weight variable must be numeric.

If you omit the WEIGHT statement, each observation contributes a value of 1 to the frequency count for its category. That is, each observation represents one subject. When you specify a WEIGHT statement, each observation contributes the value of the weighting variable for that observation. For example, a weight of 3 means that the observation represents three subjects. Weight values are not required to be integers.

You can specify the WEIGHT statement with a TABLES statement to indicate category frequencies, as in the following example:

```
proc freq;
   tables a*b / out=outfreq sparse;
run;

proc corresp freqout;
   tables a, b;
   weight count;
run;
```

If you specify a VAR statement, you can specify the WEIGHT statement to indicate supplementary observations and to weight some rows of the table more heavily than others. When the value of the WEIGHT variable is negative, the observation is treated as supplementary, and the absolute value of the weight is used as the weighting value.

You cannot specify a WEIGHT statement with a VAR statement and the MCA option, because the table must be symmetric. Supplementary variables are indicated with the SUPPLEMENTARY statement, so differential weighting of rows is inappropriate.

Details: CORRESP Procedure

Input Data Set

PROC CORRESP can read two kinds of input:

- raw category responses on two or more classification variables with the TABLES statement
- a two-way contingency table with the VAR statement

You can use output from PROC FREQ as input for PROC CORRESP.

The classification variables referred to by the TABLES statement can be either numeric or character variables. Normally, all observations for a given variable that have the same formatted value are placed in the same level, and observations with different values are placed in different levels.

The variables in the VAR statement must be numeric. The values of the observations specify the cell frequencies. These values are not required to be integers, but only those observations with all nonnegative, nonmissing values are used in the correspondence analysis. Observations with one or more negative values are removed from the analysis.

The WEIGHT variable must be numeric. Observations with negative weights are treated as supplementary observations. The absolute values of the weights are used to weight the observations.

Using the TABLES Statement

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/crsptab1.sas.)

This section explains some of the choices for the correspondence analysis input data table and illustrates some table-construction capabilities of PROC CORRESP. The SAS data set Neighbor, which follows, will be used throughout this section to illustrate various ways in which PROC CORRESP can read and process data. This data set consists of one observation for each resident in a fictitious neighborhood along with some personal information.

```
title 'PROC CORRESP Table Construction';

data Neighbor;
   input Name $ 1-10 Age $ 12-18 Sex $ 19-25
         Height $ 26-30 Hair $ 32-37;
   datalines;
Jones      Old    Male   Short White
```

```
Smith      Young  Female Tall  Brown
Kasavitz   Old    Male   Short Brown
Ernst      Old    Female Tall  White
Zannoria   Old    Female Short Brown
Spangel    Young  Male   Tall  Blond
Myers      Young  Male   Tall  Brown
Kasinski   Old    Male   Short Blond
Colman     Young  Female Short Blond
Delafave   Old    Male   Tall  Brown
Singer     Young  Male   Tall  Brown
Igor       Old           Short
;
```

This first step creates a simple contingency table or crosstabulation. In the TABLES statement, each variable list consists of a single variable. The following statements produce the table in Figure 41.3.

```
proc corresp data=Neighbor dimens=1 observed short;
   title2 'Simple Crosstabulation';
   ods select observed;
   tables Sex, Age;
run;
```

These statements create a contingency table with two rows (**Female** and **Male**) and two columns (**Old** and **Young**) and show the neighbors categorized by age and sex. The DIMENS=1 option specifies the number of dimensions in the correspondence analysis. Typically, you do not have to specify this option, because typically your tables will be larger than two by two. The default is DIMENS=2, which is too large for a table with a two-level factor. The OBSERVED option displays the contingency table. The SHORT option limits the displayed output. Because it contains missing values, the observation where Name='Igor' is omitted from the analysis. The table is shown in Figure 41.3.

Figure 41.3 Contingency Table for Sex, Age

PROC CORRESP Table Construction
Simple Crosstabulation

The CORRESP Procedure

Contingency Table			
	Old	**Young**	**Sum**
Female	2	2	4
Male	4	3	7
Sum	6	5	11

The preceding example showed how to make a two-way contingency table based on the levels of two categorical variables, which, if it were larger, would be a very typical form of data for a correspondence analysis. However, many other types of tables, **N**, can be used as input to a correspondence analysis, and all tables can be defined based on a binary matrix, **Z**. The BINARY option enables you to directly compute and display this matrix. The TABLES statement consists of a single list of all the categorical variables. The following statements produce Figure 41.4.

```
proc corresp data=neighbor observed short binary;
   title2 'Binary Coding';
   ods select binary;
   tables Hair Height Sex Age;
run;
```

Figure 41.4 Binary Table Using the BINARY Option

PROC CORRESP Table Construction
Binary Coding

The CORRESP Procedure

Binary Table									
	Blond	Brown	White	Short	Tall	Female	Male	Old	Young
1	0	0	1	1	0	0	1	1	0
2	0	1	0	0	1	1	0	0	1
3	0	1	0	1	0	0	1	1	0
4	0	0	1	0	1	1	0	1	0
5	0	1	0	1	0	1	0	1	0
6	1	0	0	0	1	0	1	0	1
7	0	1	0	0	1	0	1	0	1
8	1	0	0	1	0	0	1	1	0
9	1	0	0	1	0	1	0	0	1
10	0	1	0	0	1	0	1	1	0
11	0	1	0	0	1	0	1	0	1

In this case, $\mathbf{N} = \mathbf{Z}$ is directly analyzed. The binary matrix has one row for each individual or case and one column for each category. A binary table constructed from m categorical variables has m partitions. This binary table has four partitions, one for each of the four categorical variables. Each partition has a 1 in each row, and each row contains exactly four 1s because there are four categorical variables. More generally, the binary design matrix has exactly m 1s in each row. The 1s indicate the categories to which the observation applies. For example, the categorical variable Sex, with two levels (**Female** and **Male**), is coded using two indicator variables. For the variable Sex, a male would be coded Female=0 and Male=1, and a female would be coded Female=1 and Male=0. This is the same kind of coding that procedures like GLM and TRANSREG use for CLASS variables.

Implicitly, the binary table has an automatic row variable that is equal to the observation number. Alternatively, when there is a row ID variable, as there is in this case, you can use it as a row variable in the TABLES statement, and the resulting ordinary observed frequency table is the binary table. This example uses two variable lists: Name for the row variable, and Hair Height Sex Age for the column variables. Because two lists were provided, the BINARY option was not specified. The following statements produce Figure 41.5.

```
proc corresp data=neighbor observed short;
   title2 'Binary Coding';
   ods select observed;
   tables Name, Hair Height Sex Age;
run;
```

Figure 41.5 Binary Table Using a Row Variable

PROC CORRESP Table Construction
Binary Coding

The CORRESP Procedure

Contingency Table										
	Blond	Brown	White	Short	Tall	Female	Male	Old	Young	Sum
Colman	1	0	0	1	0	1	0	0	1	4
Delafave	0	1	0	0	1	0	1	1	0	4
Ernst	0	0	1	0	1	1	0	1	0	4
Jones	0	0	1	1	0	0	1	1	0	4
Kasavitz	0	1	0	1	0	0	1	1	0	4
Kasinski	1	0	0	1	0	0	1	1	0	4
Myers	0	1	0	0	1	0	1	0	1	4
Singer	0	1	0	0	1	0	1	0	1	4
Smith	0	1	0	0	1	1	0	0	1	4
Spangel	1	0	0	0	1	0	1	0	1	4
Zannoria	0	1	0	1	0	1	0	1	0	4
Sum	3	6	2	5	6	4	7	6	5	44

With the MCA option, the Burt table ($\mathbf{Z}'\mathbf{Z}$) is analyzed. A Burt table is a partitioned symmetric matrix containing all pairs of crosstabulations among a set of categorical variables. Each diagonal partition is a diagonal matrix containing marginal frequencies (a crosstabulation of a variable with itself). Each off-diagonal partition is an ordinary contingency table. The following statements produce Figure 41.6.

```
proc corresp data=neighbor observed short mca;
   title2 'MCA Burt Table';
   ods select burt;
   tables Hair Height Sex Age;
run;
```

Note that there is a single variable list in the TABLES statement, because the row and column variable lists are the same.

Figure 41.6 MCA Burt Table

PROC CORRESP Table Construction
MCA Burt Table

The CORRESP Procedure

Burt Table	Blond	Brown	White	Short	Tall	Female	Male	Old	Young
Blond	3	0	0	2	1	1	2	1	2
Brown	0	6	0	2	4	2	4	3	3
White	0	0	2	1	1	1	1	2	0
Short	2	2	1	5	0	2	3	4	1
Tall	1	4	1	0	6	2	4	2	4
Female	1	2	1	2	2	4	0	2	2
Male	2	4	1	3	4	0	7	4	3
Old	1	3	2	4	2	2	4	6	0
Young	2	3	0	1	4	2	3	0	5

This Burt table is composed of all pairs of crosstabulations among the variables Hair, Height, Sex, and Age. It is composed of sixteen individual subtables—the number of variables squared. Both the rows and the columns have the same nine categories (in this case Blond, Brown, White, Short, Tall, Female, Male, Old, and Young). Below the diagonal (from left to right, top to bottom) are the following crosstabulations: Height * Hair, Sex * Hair, Sex * Height, Age * Hair, Age * Height, and Age * Sex. Each crosstabulation below the diagonal has a transposed counterpart above the diagonal. The diagonal contains the crosstabulations: Hair * Hair, Height * Height, Sex * Sex, and Age * Age. The diagonal elements of the diagonal partitions contain marginal frequencies of the off-diagonal partitions. The table Hair * Height, for example, has three rows for Hair and two columns for Height. The values of the Hair * Height table, summed across rows, sum to the diagonal values of the Height * Height table, as displayed in the following results. The following statements produce Figure 41.7.

```
proc corresp data=neighbor observed short dimens=1;
   title2 'Part of the Burt Table';
   ods output observed=o;
   tables Hair Height, Height;
run;

proc print data=o(drop=sum) label noobs;
   where label ne 'Sum';
   label label = '00'x;
run;
```

Figure 41.7 Part of the Burt Table

PROC CORRESP Table Construction
Part of the Burt Table

	Short	Tall
Blond	2	1
Brown	2	4
White	1	1
Short	5	0
Tall	0	6

A simple crosstabulation of Hair × Height is $\mathbf{N} = \mathbf{Z}_{\text{Hair}}{}'\mathbf{Z}_{\text{Height}}$. Tables such as ($\mathbf{N} = \mathbf{Z}_{\text{Hair}}{}'\mathbf{Z}_{\text{Height,Sex}}$), made up of several crosstabulations, can also be analyzed in simple correspondence analysis. The following statements produce Figure 41.8.

```
proc corresp data=neighbor observed short dimens=1;
   title2 'Multiple Crosstabulations';
   ods select observed;
   tables Hair, Height Sex;
run;
```

Figure 41.8 Hair × (Height Sex) Crosstabulation

PROC CORRESP Table Construction
Multiple Crosstabulations

The CORRESP Procedure

Contingency Table					
	Short	Tall	Female	Male	Sum
Blond	2	1	1	2	6
Brown	2	4	2	4	12
White	1	1	1	1	4
Sum	5	6	4	7	22

The following statements create a table with six rows (**Blond*Short**, **Blond*Tall**, **Brown*Short**, **Brown*Tall**, **White*Short**, and **White*Tall**) and four columns (**Female**, **Male**, **Old**, and **Young**). The levels of the row variables are crossed by the CROSS=ROW option, forming mutually exclusive categories. Hence each individual fits into exactly one row category, but two column categories. The following statements produce Figure 41.9.

```
proc corresp data=Neighbor cross=row observed short;
   title2 'Multiple Crosstabulations with Crossed Rows';
   ods select observed;
   tables Hair Height, Sex Age;
run;
```

Figure 41.9 Contingency Table for Hair * Height, Sex Age

PROC CORRESP Table Construction
Multiple Crosstabulations with Crossed Rows

The CORRESP Procedure

Contingency Table					
	Female	Male	Old	Young	Sum
Blond * Short	1	1	1	1	4
Blond * Tall	0	1	0	1	2
Brown * Short	1	1	2	0	4
Brown * Tall	1	3	1	3	8
White * Short	0	1	1	0	2
White * Tall	1	0	1	0	2
Sum	4	7	6	5	22

You can enter supplementary variables with TABLES input by including a SUPPLEMENTARY statement. Variables named in the SUPPLEMENTARY statement indicate TABLES variables with categories that are supplementary. In other words, the categories of the variable Age are represented in the row and column space, but they are not used in determining the scores of the categories of the variables Hair, Height, and Sex. The variable used in the SUPPLEMENTARY statement must be listed in the TABLES statement as well. For example, the following statements create a Burt table with seven active rows and columns (**Blond**, **Brown**, **White**, **Short**, **Tall**, **Female**, **Male**) and two supplementary rows and columns (**Old** and **Young**). The following statements produce Figure 41.10.

```
proc corresp data=Neighbor observed short mca;
   title2 'MCA with Supplementary Variables';
   ods select burt supcols;
   tables Hair Height Sex Age;
   supplementary Age;
run;
```

Figure 41.10 Burt Table from PROC CORRESP with Supplementary Variables

PROC CORRESP Table Construction
MCA with Supplementary Variables

The CORRESP Procedure

Burt Table							
	Blond	Brown	White	Short	Tall	Female	Male
Blond	3	0	0	2	1	1	2
Brown	0	6	0	2	4	2	4
White	0	0	2	1	1	1	1
Short	2	2	1	5	0	2	3
Tall	1	4	1	0	6	2	4
Female	1	2	1	2	2	4	0
Male	2	4	1	3	4	0	7

Figure 41.10 *continued*

	Supplementary Columns	
	Old	**Young**
Blond	1	2
Brown	3	3
White	2	0
Short	4	1
Tall	2	4
Female	2	2
Male	4	3

The following statements create a binary table with 7 active columns (**Blond**, **Brown**, **White**, **Short**, **Tall**, **Female**, **Male**), 2 supplementary columns (**Old** and **Young**), and 11 rows for the 11 observations with nonmissing values. The following statements produce Figure 41.11.

```
proc corresp data=Neighbor observed short binary;
   title2 'Supplementary Binary Variables';
   ods select binary supcols;
   tables Hair Height Sex Age;
   supplementary Age;
run;
```

Figure 41.11 Binary Table from PROC CORRESP with Supplementary Variables

PROC CORRESP Table Construction
Supplementary Binary Variables

The CORRESP Procedure

	Binary Table						
	Blond	**Brown**	**White**	**Short**	**Tall**	**Female**	**Male**
1	0	0	1	1	0	0	1
2	0	1	0	0	1	1	0
3	0	1	0	1	0	0	1
4	0	0	1	0	1	1	0
5	0	1	0	1	0	1	0
6	1	0	0	0	1	0	1
7	0	1	0	0	1	0	1
8	1	0	0	1	0	0	1
9	1	0	0	1	0	1	0
10	0	1	0	0	1	0	1
11	0	1	0	0	1	0	1

Figure 41.11 *continued*

	Supplementary Columns	
	Old	Young
1	1	0
2	0	1
3	1	0
4	1	0
5	1	0
6	0	1
7	0	1
8	1	0
9	0	1
10	1	0
11	0	1

Using the VAR Statement

With VAR statement input, the rows of the contingency table correspond to the observations of the input data set, and the columns correspond to the VAR statement variables. The values of the variables typically contain the table frequencies. The table in Figure 41.3 could be created with VAR statement input by using the following statements:

```
data Ages;
   input Sex $ Old Young;
   datalines;
Female  2 2
Male    4 3
;

proc corresp data=Ages dimens=1 observed short;
   var Old Young;
   id Sex;
run;
```

Only nonnegative values are accepted. Negative values are treated as missing, causing the observation to be excluded from the analysis. The values are not required to be integers. Row labels for the table are specified with an ID variable. Column labels are constructed from the variable name or variable label if one is specified. When you specify multiple correspondence analysis (MCA), the row and column labels are the same and are constructed from the variable names or labels, so you cannot include an ID statement. With MCA, the VAR statement must list the variables in the order in which the rows occur. An example is the table in Figure 41.6, which was created with the following TABLES statement.

```
   tables Hair Height Sex Age;
```

This table could have been created with a VAR statement as follows:

```
proc corresp data=table nvars=4 mca;
   var Blond Brown White Short Tall Female Male Old Young;
run;
```

You must specify the NVARS= option in order to specify the number of original categorical variables with the MCA option. The option NVARS=*n* is needed to find boundaries between the subtables of the Burt table. If f is the sum of all elements in the Burt table $\mathbf{Z}'\mathbf{Z}$, then fn^{-2} is the number of rows in the binary matrix $\mathbf{Z}$. The sum of all elements in each diagonal subtable of the Burt table must be fn^{-2}.

To enter supplementary observations, include a WEIGHT statement with negative weights for those observations. Specify the SUPPLEMENTARY statement to include supplementary variables. You must list supplementary variables in both the VAR and SUPPLEMENTARY statements.

Missing and Invalid Data

With VAR statement input, observations with missing or negative frequencies are excluded from the analysis. Supplementary variables and supplementary observations with missing or negative frequencies are also excluded. Negative weights are valid with VAR statement input.

With TABLES statement input, observations with negative weights are excluded from the analysis. With this form of input, missing cell frequencies cannot occur. Observations with missing values on the categorical variables are excluded unless you specify the MISSING option. If you specify the MISSING option, ordinary missing values and special missing values are treated as additional levels of a categorical variable. In all cases, if any row or column of the constructed table contains only zeros, that row or column is excluded from the analysis.

Observations with missing weights are excluded from the analysis.

Coding, Fuzzy Coding, and Doubling

Sometimes, binary data such as Yes/No data are available—for example, 1 means “Yes, I have bought this brand in the last month” and 0 means “No, I have not bought this brand in the last month”. The following statements read a data set with Yes/No purchase data for three hypothetical brands.

```
title 'Doubling Yes/No Data';

proc format;
   value yn 0 = 'No '  1 = 'Yes';
run;

data BrandChoice;
   input a b c;
   label a = 'Brand A' b = 'Brand B' c = 'Brand B';
   format a b c yn.;
   datalines;
0 0 1
1 1 0
0 1 1
0 1 0
```

```
1 0 0
;
```

Data such as these cannot be analyzed directly because the raw data do not consist of partitions, each with one column per level and exactly one 1 in each row. (See the section "Using the TABLES Statement" on page 2666.) The data must be *doubled* so that both Yes and No are represented by a column in the data matrix. The TRANSREG procedure provides one way of doubling. In the following statements, the DESIGN option specifies that PROC TRANSREG is being used only for coding, not analysis. The option SEPARATORS=': ' specifies that labels for the coded columns are constructed from input variable labels, followed by a colon and space, followed by the formatted value. The variables are designated in the MODEL statement as CLASS variables, and the ZERO=NONE option creates binary variables for all levels. The OUTPUT statement specifies the output data set and drops the _NAME_, _TYPE_, and Intercept variables. PROC TRANSREG stores a list of coded variable names in a macro variable &_TRGIND, which in this case has the value "aNo aYes bNo bYes cNo cYes". This macro variable can be used directly in the VAR statement in PROC CORRESP. The following statements produce Figure 41.12. Only the input table is displayed.

```
proc transreg data=BrandChoice design separators=': ';
   model class(a b c / zero=none);
   output out=Doubled(drop=_: Intercept);
run;

proc print label;
run;

proc corresp data=Doubled norow short;
   var &_trgind;
run;
```

Figure 41.12 Doubling Yes/No Data

Doubling Yes/No Data

Obs	Brand A: No	Brand A: Yes	Brand B: No	Brand B: Yes	Brand B: No	Brand B: Yes	Brand A	Brand B	Brand B
1	1	0	1	0	0	1	No	No	Yes
2	0	1	0	1	1	0	Yes	Yes	No
3	1	0	0	1	0	1	No	Yes	Yes
4	1	0	0	1	1	0	No	Yes	No
5	0	1	1	0	1	0	Yes	No	No

A fuzzy-coded indicator also sums to 1.0 across levels of the categorical variable, but it is coded with fractions rather than with 0 and 1. The fractions represent the distribution of the attribute across several levels of the categorical variable.

Ordinal variables, such as survey responses of 1 to 3, can be represented as two fuzzy-coded variables, as shown in Table 41.2.

Table 41.2 Coding an Ordinal Variable

Ordinal Values	Coding	
1	0.25	0.75
2	0.50	0.50
3	0.75	0.25

The values of the coding sum to one across the two coded variables.

These next steps illustrate the use of binary and fuzzy-coded indicator variables. Fuzzy-coded indicators are used to represent missing data. Note that the missing values in the observation Igor are coded with equal proportions. The following statements produce Figure 41.13.

```
title 'Fuzzy Coding of Missing Values';

proc transreg data=Neighbor design cprefix=0;
   model class(Age Sex Height Hair / zero=none);
   output out=Neighbor2(drop=_: Intercept);
   id Name;
run;

data Neighbor3;
   set Neighbor2;
   if Sex = ' ' then do;
      Female = 0.5;
      Male   = 0.5;
   end;
   if Hair = ' ' then do;
      White = 1/3;
      Brown = 1/3;
      Blond = 1/3;
   end;
run;

proc print label noobs data=Neighbor3(drop=age--name);
   format _numeric_ best4.;
run;
```

Figure 41.13 Fuzzy Coding of Missing Values

Fuzzy Coding of Missing Values

Age Old	Age Young	Sex Female	Sex Male	Height Short	Height Tall	Hair Blond	Hair Brown	Hair White
1	0	0	1	1	0	0	0	1
0	1	1	0	0	1	0	1	0
1	0	0	1	1	0	0	1	0
1	0	1	0	0	1	0	0	1
1	0	1	0	1	0	0	1	0
0	1	0	1	0	1	1	0	0
0	1	0	1	0	1	0	1	0
1	0	0	1	1	0	1	0	0
0	1	1	0	1	0	1	0	0
1	0	0	1	0	1	0	1	0
0	1	0	1	0	1	0	1	0
1	0	0.5	0.5	1	0	0.33	0.33	0.33

There is one set of coded variables for each input categorical variable. If observation 12 is excluded, each set is a binary design matrix. Each design matrix has one column for each category and exactly one 1 in each row. Fuzzy coding is shown in the final observation, which corresponds to Igor. The observation for Igor has missing values for the variables Sex and Hair. The design matrix variables are coded with fractions that sum to one within each categorical variable.

An alternative way to represent missing data is to treat missing values as an additional level of the categorical variable. This alternative is available with the MISSING option in the PROC CORRESP statement. This approach yields coordinates for missing responses, allowing the comparison of "missing" along with the other levels of the categorical variables.

Greenacre and Hastie (1987) discuss additional coding schemes, including one for continuous variables. Continuous variables can be coded with PROC TRANSREG by specifying BSPLINE(*variables* / degree=1) in the MODEL statement.

Creating a Data Set Containing the Crosstabulation

The CORRESP procedure can read or create a contingency or Burt table. PROC CORRESP is generally more efficient with VAR statement input than with TABLES statement input. TABLES statement input requires that the table be created from raw categorical variables, whereas the VAR statement is used to read an existing table. For extremely large problems, if PROC CORRESP runs out of memory, it might be possible to use some other method to create the table and then use VAR statement input with PROC CORRESP.

The following example uses the CORRESP, FREQ, and TRANSPOSE procedures to create rectangular tables from a SAS data set WORK.A that contains the categorical variables V1–V5. The Burt table examples assume that no categorical variable has a value found in any of the other categorical variables (that is, that each row and column label is unique).

You can use PROC CORRESP and the ODS OUTPUT statement as follows to create a rectangular two-way contingency table from two categorical variables:

```
proc corresp data=a observed short;
   ods output Observed=Obs(drop=Sum where=(Label ne 'Sum'));
   tables v1, v2;
run;
```

You can use PROC FREQ and PROC TRANSPOSE to create a rectangular two-way contingency table from two categorical variables, as in the following statements:

```
proc freq data=a;
   tables v1 * v2 / sparse noprint out=freqs;
run;

proc transpose data=freqs out=rfreqs(drop=_:);
   id  v2;
   var count;
   by  v1;
run;
```

You can use PROC CORRESP and the ODS OUTPUT statement as follows to create a Burt table from five categorical variables:

```
proc corresp data=a observed short mca;
   ods output Burt=Obs;
   tables v1-v5;
run;
```

You can use a DATA step, PROC FREQ, and PROC TRANSPOSE to create a Burt table from five categorical variables, as in the following statements:

```
data b;
   set a;
   array v[5] $ v1-v5;
   do i = 1 to 5;
      row = v[i];
      do j = 1 to 5;
         column = v[j];
         output;
      end;
   end;
   keep row column;
run;

proc freq data=b;
   tables row * column / sparse  noprint out=freqs;
run;

proc transpose data=freqs out=rfreqs(drop=_:);
   id  column;
   var count;
   by  row;
run;
```

Output Data Sets

PROC CORRESP has two output data sets. The OUTC= data set contains coordinates and the results of the correspondence analysis. The OUTF= data set contains frequencies and other cross-tabulation results.

The OUTC= Data Set

The OUTC= data set contains two or three character variables and $4n + 4$ numeric variables, where n is the number of axes from DIMENS=n (two by default). The OUTC= data set contains one observation for each row, column, supplementary row, and supplementary column point, and one observation for inertias.

The first variable is named _TYPE_ and identifies the type of observation. The values of _TYPE_ are as follows:

- The 'INERTIA' observation contains the total inertia in the INERTIA variable, and each dimension's inertia in the Contr1–Contr$n$ variables.
- The 'OBS' observations contain the coordinates and statistics for the rows of the table.
- The 'SUPOBS' observations contain the coordinates and statistics for the supplementary rows of the table.
- The 'VAR' observations contain the coordinates and statistics for the columns of the table.
- The 'SUPVAR' observations contain the coordinates and statistics for the supplementary columns of the table.

If you specify the SOURCE option, then the data set also contains a variable _VAR_ containing the name or label of the input variable from which that row originates. The name of the next variable is either _NAME_ or (if you specify an ID statement) the name of the ID variable.

For observations with a value of 'OBS' or 'SUPOBS' for the _TYPE_ variable, the values of the second variable are constructed as follows:

- When you use a VAR statement without an ID statement, the values are 'Row1', 'Row2', and so on.
- When you specify a VAR statement with an ID statement, the values are set equal to the values of the ID variable.
- When you specify a TABLES statement, the _NAME_ variable has values formed from the appropriate row variable values.

For observations with a value of 'VAR' or 'SUPVAR' for the _TYPE_ variable, the values of the second variable are equal to the names or labels of the VAR (or SUPPLEMENTARY) variables. When you specify a TABLES statement, the values are formed from the appropriate column variable values.

The third and subsequent variables contain the numerical results of the correspondence analysis.

- Quality contains the quality of each point's representation in the DIMENS=n dimensional display, which is the sum of squared cosines over the first n dimensions.

- Mass contains the masses or marginal sums of the relative frequency matrix.
- Inertia contains each point's relative contribution to the total inertia.
- Dim1–Dim*n* contain the point coordinates.
- Contr1–Contr*n* contain the partial contributions to inertia.
- SqCos1–SqCos*n* contain the squared cosines.
- Best1–Best*n* and Best contain the summaries of the partial contributions to inertia.

The OUTF= Data Set

The OUTF= data set contains frequencies and percentages. It is similar to a PROC FREQ output data set. The OUTF= data set begins with a variable called _TYPE_, which contains the observation type. If the SOURCE option is specified, the data set contains two variables, _ROWVAR_ and _COLVAR_, that contain the names or labels of the row and column input variables from which each cell originates. The next two variables are classification variables that contain the row and column levels. If you use TABLES statement input and each variable list consists of a single variable, the names of the first two variables match the names of the input variables; otherwise, these variables are named Row and Column. The next two variables are Count and Percent, which contain frequencies and percentages.

The _TYPE_ variable can have the following values:

- 'OBSERVED' observations contain the contingency table.
- 'SUPOBS' observations contain the supplementary rows.
- 'SUPVAR' observations contain the supplementary columns.
- 'EXPECTED' observations contain the product of the row marginals and the column marginals divided by the grand frequency of the observed frequency table. For ordinary two-way contingency tables, these are the expected frequency matrix under the hypothesis of row and column independence.
- 'DEVIATION' observations contain the matrix of deviations between the observed frequency matrix and the product of its row marginals and column marginals divided by its grand frequency. For ordinary two-way contingency tables, these are the observed minus expected frequencies under the hypothesis of row and column independence.
- 'CELLCHI2' observations contain contributions to the total chi-square test statistic.
- 'RP' observations contain the row profiles.
- 'SUPRP' observations contain supplementary row profiles.
- 'CP' observations contain the column profiles.
- 'SUPCP' observations contain supplementary column profiles.

Computational Resources

Let

$$
\begin{aligned}
n_r &= \text{number of rows in the table} \\
n_c &= \text{number of columns in the table} \\
n &= \text{number of observations} \\
v &= \text{number of VAR statement variables} \\
t &= \text{number of TABLES statement variables} \\
c &= \max(n_r, n_c) \\
d &= \min(n_r, n_c)
\end{aligned}
$$

For TABLES statement input, more than

$$32(t+1) + 8(\max(2tn, (n_r+3)(n_c+3)))$$

bytes of array space are required.

For VAR statement input, more than

$$16(v+2) + 8(n_r+3)(n_c+3)$$

bytes of array space are required.

Memory

The computational resources formulas are underestimates of the amounts of memory needed to handle most problems. If you use a utility data set, and if memory could be used with perfect efficiency, then roughly the stated amount of memory would be needed. In reality, most problems require at least two or three times the minimum.

PROC CORRESP tries to store the raw data (TABLES input) and the contingency table in memory. If there is not enough memory, a utility data set is used, potentially resulting in a large increase in execution time.

Time

The time required to perform the generalized singular value decomposition is roughly proportional to $2cd^2 + 5d^3$. Overall computation time increases with table size at a rate roughly proportional to $(n_r n_c)^{\frac{3}{2}}$.

Algorithm and Notation

This section is primarily based on the theory of correspondence analysis found in Greenacre (1984). If you are interested in other references, see the section "Background" on page 2650.

Let $\mathbf{N}$ be the contingency table formed from those observations and variables that are not supplementary and from those observations that have no missing values and have a positive weight. This table is an $(n_r \times n_c)$ rank q matrix of nonnegative numbers with nonzero row and column sums. If $\mathbf{Z}_a$ is the binary coding for variable A, and $\mathbf{Z}_b$ is the binary coding for variable B, then $\mathbf{N} = \mathbf{Z}_a'\mathbf{Z}_b$ is a contingency table. Similarly, if $\mathbf{Z}_{b,c}$ contains the binary coding for both variables B and C, then $\mathbf{N} = \mathbf{Z}_a'\mathbf{Z}_{b,c}$ can also be input to a correspondence analysis. With the BINARY option, $\mathbf{N} = \mathbf{Z}$, and the analysis is based on a binary table. In multiple correspondence analysis, the analysis is based on a Burt table, $\mathbf{Z}'\mathbf{Z}$.

Let $\mathbf{1}$ be a vector of 1s of the appropriate order, let $\mathbf{I}$ be an identity matrix, and let diag(·) be a matrix-valued function that creates a diagonal matrix from a vector. Let

$$
\begin{aligned}
f &= \mathbf{1}'\mathbf{N}\mathbf{1} \\
\mathbf{P} &= \frac{1}{f}\mathbf{N} \\
\mathbf{r} &= \mathbf{P}\mathbf{1} \\
\mathbf{c} &= \mathbf{P}'\mathbf{1} \\
\mathbf{D_r} &= \mathrm{diag}(\mathbf{r}) \\
\mathbf{D_c} &= \mathrm{diag}(\mathbf{c}) \\
\mathbf{R} &= \mathbf{D_r}^{-1}\mathbf{P} \\
\mathbf{C}' &= \mathbf{D_c}^{-1}\mathbf{P}'
\end{aligned}
$$

The scalar f is the sum of all elements in $\mathbf{N}$. The matrix $\mathbf{P}$ is a matrix of relative frequencies. The vector $\mathbf{r}$ contains row marginal proportions or row "masses." The vector $\mathbf{c}$ contains column marginal proportions or column masses. The matrices $\mathbf{D_r}$ and $\mathbf{D_c}$ are diagonal matrices of marginals.

The rows of $\mathbf{R}$ contain the "row profiles." The elements of each row of $\mathbf{R}$ sum to one. Each (i, j) element of $\mathbf{R}$ contains the observed probability of being in column j given membership in row i. Similarly, the columns of $\mathbf{C}$ contain the column profiles. The coordinates in correspondence analysis are based on the generalized singular value decomposition of $\mathbf{P}$,

$$\mathbf{P} = \mathbf{A}\mathbf{D_u}\mathbf{B}'$$

where

$$\mathbf{A}'\mathbf{D_r}^{-1}\mathbf{A} = \mathbf{B}'\mathbf{D_c}^{-1}\mathbf{B} = \mathbf{I}$$

In multiple correspondence analysis,

$$\mathbf{P} = \mathbf{B}\mathbf{D_u}^2\mathbf{B}'$$

The matrix $\mathbf{A}$, which is the rectangular matrix of left generalized singular vectors, has n_r rows and q columns; the matrix $\mathbf{D_u}$, which is a diagonal matrix of singular values, has q rows and columns; and the matrix $\mathbf{B}$,

which is the rectangular matrix of right generalized singular vectors, has n_c rows and q columns. The columns of $\mathbf{A}$ and $\mathbf{B}$ define the principal axes of the column and row point clouds, respectively.

The generalized singular value decomposition of $\mathbf{P} - \mathbf{rc}'$, discarding the last singular value (which is zero) and the last left and right singular vectors, is exactly the same as a generalized singular value decomposition of $\mathbf{P}$, discarding the first singular value (which is one), the first left singular vector, $\mathbf{r}$, and the first right singular vector, $\mathbf{c}$. The first (trivial) column of $\mathbf{A}$ and $\mathbf{B}$ and the first singular value in $\mathbf{D_u}$ are discarded before any results are displayed. You can obtain the generalized singular value decomposition of $\mathbf{P} - \mathbf{rc}'$ from the ordinary singular value decomposition of $\mathbf{D_r^{-1/2}}(\mathbf{P} - \mathbf{rc}')\mathbf{D_c^{-1/2}}$:

$$\mathbf{D_r^{-1/2}}(\mathbf{P} - \mathbf{rc}')\mathbf{D_c^{-1/2}} = \mathbf{U}\mathbf{D_u}\mathbf{V}' = (\mathbf{D_r^{-1/2}}\mathbf{A})\mathbf{D_u}(\mathbf{D_c^{-1/2}}\mathbf{B})'$$

$$\mathbf{P} - \mathbf{rc}' = \mathbf{D_r^{1/2}}\mathbf{U}\mathbf{D_u}\mathbf{V}'\mathbf{D_c^{1/2}} = (\mathbf{D_r^{1/2}}\mathbf{U})\mathbf{D_u}(\mathbf{D_c^{1/2}}\mathbf{V})' = \mathbf{A}\mathbf{D_u}\mathbf{B}'$$

Hence, $\mathbf{A} = \mathbf{D_r^{1/2}}\mathbf{U}$ and $\mathbf{B} = \mathbf{D_c^{1/2}}\mathbf{V}$.

The default row coordinates are $\mathbf{D_r^{-1}}\mathbf{A}\mathbf{D_u}$, and the default column coordinates are $\mathbf{D_c^{-1}}\mathbf{B}\mathbf{D_u}$. Typically the first two columns of $\mathbf{D_r^{-1}}\mathbf{A}\mathbf{D_u}$ and $\mathbf{D_c^{-1}}\mathbf{B}\mathbf{D_u}$ are plotted to display graphically associations between the row and column categories. The plot consists of two overlaid plots, one for rows and one for columns. The row points are row profiles, and the column points are column profiles, both rescaled so that distances between profiles can be displayed as ordinary Euclidean distances, then orthogonally rotated to a principal axes orientation. Distances between row points and other row points have meaning, as do distances between column points and other column points. However, distances between column points and row points are not interpretable.

The PROFILE=, ROW=, and COLUMN= Options

The PROFILE=, ROW=, and COLUMN= options standardize the coordinates before they are displayed and placed in the output data set. The options PROFILE=BOTH, PROFILE=ROW, and PROFILE=COLUMN provide the standardizations that are typically used in correspondence analysis. There are six choices each for row and column coordinates (see Table 41.3). However, most of the combinations of the ROW= and COLUMN= options are not useful. The ROW= and COLUMN= options are provided for completeness, but they are not intended for general use.

Table 41.3 Coordinates

ROW=	**Matrix Formula**
A	$\mathbf{A}$
AD	$\mathbf{AD_u}$
DA	$\mathbf{D_r^{-1}A}$
DAD	$\mathbf{D_r^{-1}AD_u}$
DAD1/2	$\mathbf{D_r^{-1}AD_u^{1/2}}$
DAID1/2	$\mathbf{D_r^{-1}A(I+D_u)^{1/2}}$
COLUMN=	**Matrix Formula**
B	$\mathbf{B}$
BD	$\mathbf{BD_u}$
DB	$\mathbf{D_c^{-1}B}$
DBD	$\mathbf{D_c^{-1}BD_u}$
DBD1/2	$\mathbf{D_c^{-1}BD_u^{1/2}}$
DBID1/2	$\mathbf{D_c^{-1}B(I+D_u)^{1/2}}$

When PROFILE=ROW (ROW=DAD and COLUMN=DB), the row coordinates $\mathbf{D_r^{-1}AD_u}$ and column coordinates $\mathbf{D_c^{-1}B}$ provide a correspondence analysis based on the row profile matrix. The row profile (conditional probability) matrix is defined as $\mathbf{R} = \mathbf{D_r^{-1}P} = \mathbf{D_r^{-1}AD_uB'}$. The elements of each row of $\mathbf{R}$ sum to one. Each (i, j) element of $\mathbf{R}$ contains the observed probability of being in column j given membership in row i. The "principal" row coordinates $\mathbf{D_r^{-1}AD_u}$ and "standard" column coordinates $\mathbf{D_c^{-1}B}$ provide a decomposition of $\mathbf{D_r^{-1}AD_uB'D_c^{-1}} = \mathbf{D_r^{-1}PD_c^{-1}} = \mathbf{RD_c^{-1}}$. Because $\mathbf{D_r^{-1}AD_u} = \mathbf{RD_c^{-1}B}$, the row coordinates are weighted centroids of the column coordinates. Each column point, with coordinates scaled to standard coordinates, defines a vertex in $(n_c - 1)$-dimensional space. All of the principal row coordinates are located in the space defined by the standard column coordinates. Distances among row points have meaning, but distances among column points and distances between row and column points are not interpretable.

The option PROFILE=COLUMN can be described as applying the PROFILE=ROW formulas to the transpose of the contingency table. When PROFILE=COLUMN (ROW=DA and COLUMN=DBD), the principal column coordinates $\mathbf{D_c^{-1}BD_u}$ are weighted centroids of the standard row coordinates $\mathbf{D_r^{-1}A}$. Each row point, with coordinates scaled to standard coordinates, defines a vertex in $(n_r - 1)$-dimensional space. All of the principal column coordinates are located in the space defined by the standard row coordinates. Distances among column points have meaning, but distances among row points and distances between row and column points are not interpretable.

The usual sets of coordinates are given by the default PROFILE=BOTH (ROW=DAD and COLUMN=DBD). All of the summary statistics, such as the squared cosines and contributions to inertia, apply to these two sets of points. One advantage to using these coordinates is that both sets ($\mathbf{D_r^{-1}AD_u}$ and $\mathbf{D_c^{-1}BD_u}$) are postmultiplied by the diagonal matrix $\mathbf{D_u}$, which has diagonal values that are all less than or equal to one. When $\mathbf{D_u}$ is a part of the definition of only one set of coordinates, that set forms a tight cluster near the centroid, whereas the other set of points is more widely dispersed. Including $\mathbf{D_u}$ in both sets makes a better

graphical display. However, care must be taken in interpreting such a plot. No correct interpretation of distances between row points and column points can be made.

Another property of this choice of coordinates concerns the geometry of distances between points within each set. The default row coordinates can be decomposed into $\mathbf{D_r^{-1}AD_u} = \mathbf{D_r^{-1}AD_uB'D_c^{-1}B} = (\mathbf{D_r^{-1}P})(\mathbf{D_c^{-1/2}})(\mathbf{D_c^{-1/2}B})$. The row coordinates are row profiles ($\mathbf{D_r^{-1}P}$), rescaled by $\mathbf{D_c^{-1/2}}$ (rescaled so that distances between profiles are transformed from a chi-square metric to a Euclidean metric), then orthogonally rotated (with $\mathbf{D_c^{-1/2}B}$) to a principal axes orientation. Similarly, the column coordinates are column profiles rescaled to a Euclidean metric and orthogonally rotated to a principal axes orientation.

The rationale for computing distances between row profiles by using the non-Euclidean chi-square metric is as follows. Each row of the contingency table can be viewed as a realization of a multinomial distribution conditional on its row marginal frequency. The null hypothesis of row and column independence is equivalent to the hypothesis of homogeneity of the row profiles. A significant chi-square statistic is geometrically interpreted as a significant deviation of the row profiles from their centroid, $\mathbf{c}'$. The chi-square metric is the Mahalanobis metric between row profiles based on their estimated covariance matrix under the homogeneity assumption (Greenacre and Hastie 1987). A parallel argument can be made for the column profiles.

When ROW=DAD1/2 and COLUMN=DBD1/2 (Gifi 1990; Van der Heijden and de Leeuw 1985), the row coordinates $\mathbf{D_r^{-1}AD_u^{1/2}}$ and column coordinates $\mathbf{D_c^{-1}BD_u^{1/2}}$ are a decomposition of $\mathbf{D_r^{-1}PD_c^{-1}}$.

In all of the preceding pairs, distances between row and column points are not meaningful. This prompted Carroll, Green, and Schaffer (1986) to propose that row coordinates $\mathbf{D_r^{-1}A(I+D_u)^{1/2}}$ and column coordinates $\mathbf{D_c^{-1}B(I+D_u)^{1/2}}$ be used. These coordinates are (except for a constant scaling) the coordinates from a multiple correspondence analysis of a Burt table created from two categorical variables. This standardization is available with ROW=DAID1/2 and COLUMN=DBID1/2. However, this approach has been criticized on both theoretical and empirical grounds by Greenacre (1989). The Carroll, Green, and Schaffer standardization relies on the assumption that the chi-square metric is an appropriate metric for measuring the distance between the columns of a bivariate indicator matrix. See the section "Using the TABLES Statement" on page 2666 for a description of indicator matrices. Greenacre (1989) showed that this assumption cannot be justified.

The MCA Option

The MCA option performs a multiple correspondence analysis (MCA). This option requires a Burt table. You can specify the MCA option with a table created from a design matrix with fuzzy coding schemes as long as every row of every partition of the design matrix has the same marginal sum. For example, each row of each partition could contain the probabilities that the observation is a member of each level. Then the Burt table constructed from this matrix no longer contains all integers, and the diagonal partitions are no longer diagonal matrices, but MCA is still valid.

A TABLES statement with a single variable list creates a Burt table. Thus, you can always specify the MCA option with this type of input. If you use the MCA option when reading an existing table with a VAR statement, you must ensure that the table is a Burt table.

If you perform MCA on a table that is not a Burt table, the results of the analysis are invalid. If the table is not symmetric, or if the sums of all elements in each diagonal partition are not equal, PROC CORRESP displays an error message and quits.

A subset of the columns of a Burt table is not necessarily a Burt table, so in MCA it is not appropriate to designate arbitrary columns as supplementary. You can, however, designate all columns from one or more categorical variables as supplementary.

The results of a multiple correspondence analysis of a Burt table $\mathbf{Z}'\mathbf{Z}$ are the same as the column results from a simple correspondence analysis of the binary (or fuzzy) matrix $\mathbf{Z}$. Multiple correspondence analysis is not a simple correspondence analysis of the Burt table. It is not appropriate to perform a simple correspondence analysis of a Burt table. The MCA option is based on $\mathbf{P} = \mathbf{B}\mathbf{D}_\mathbf{u}^2\mathbf{B}'$, whereas a simple correspondence analysis of the Burt table would be based on $\mathbf{P} = \mathbf{B}\mathbf{D}_\mathbf{u}\mathbf{B}'$.

Because the rows and columns of the Burt table are the same, no row information is displayed or written to the output data sets. The resulting inertias and the default (COLUMN=DBD) column coordinates are the appropriate inertias and coordinates for an MCA. The supplementary column coordinates, cosines, and quality of representation formulas for MCA differ from the simple correspondence analysis formulas because the design matrix column profiles and left singular vectors are not available.

The following statements create a Burt table and perform a multiple correspondence analysis:

```
proc corresp data=Neighbor observed short mca;
   tables Hair Height Sex Age;
run;
```

Both the rows and the columns have the same nine categories (Blond, Brown, White, Short, Tall, Female, Male, Old, and Young).

MCA Adjusted Inertias

The usual principal inertias of a Burt table constructed from m categorical variables in MCA are the eigenvalues u_k from $\mathbf{D}_\mathbf{u}^2$. The problem with these inertias is that they provide a pessimistic indication of fit. Benzécri (1979) proposed the following inertia adjustment, which is also described by Greenacre (1984, p. 145):

$$\left(\frac{m}{m-1}\right)^2 \times \left(u_k - \frac{1}{m}\right)^2 \qquad \text{for } u_k > \frac{1}{m}$$

This adjustment computes the percent of adjusted inertia relative to the sum of the adjusted inertias for all inertias greater than $\frac{1}{m}$. The Benzécri adjustment is available with the BENZECRI option.

Greenacre (1994, p. 156) argues that the Benzécri adjustment overestimates the quality of fit. Greenacre proposes instead to compute the percentage of adjusted inertia relative to

$$\frac{m}{m-1}\left(\text{trace}(D_u^4) - \frac{n_c - m}{m^2}\right)$$

for all inertias greater than $\frac{1}{m}$, where $\text{trace}(D_u^4)$ is the sum of squared inertias. The Greenacre adjustment is available with the GREENACRE option.

Ordinary unadjusted inertias are printed by default with MCA when neither the BENZECRI nor the GREENACRE option is specified. However, the unadjusted inertias are not printed by default when either the BENZECRI or the GREENACRE option is specified. To display both adjusted and unadjusted inertias, specify the UNADJUSTED option in addition to the relevant adjusted inertia option (BENZECRI, GREENACRE, or both).

Supplementary Rows and Columns

Supplementary rows and columns are represented as points in the joint row and column space, but they are not used in determining the locations of the other active rows and columns of the table. The formulas that are used to compute coordinates for the supplementary rows and columns depend on the PROFILE= option or the ROW= and COLUMN= options. Let $\mathbf{S_o}$ be a matrix with rows that contain the supplementary observations, and let $\mathbf{S_v}$ be a matrix with rows that contain the supplementary variables. Note that $\mathbf{S_v}$ is defined to be the transpose of the supplementary variable partition of the table. Let $\mathbf{R_s} = \mathrm{diag}(\mathbf{S_o 1})^{-1}\mathbf{S_o}$ be the supplementary observation profile matrix, and let $\mathbf{C_s} = \mathrm{diag}(\mathbf{S_v 1})^{-1}\mathbf{S_v}$ be the supplementary variable profile matrix. Note that the notation $\mathrm{diag}(\cdot)^{-1}$ means to convert the vector to a diagonal matrix, then invert the diagonal matrix. The coordinates for the supplementary observations and variables are shown in Table 41.4.

Table 41.4 Coordinates for Supplementary Observations

ROW=	**Matrix Formula**
A	$\frac{1}{f}\mathbf{S_o D_c^{-1} B D_u^{-1}}$
AD	$\frac{1}{f}\mathbf{S_o D_c^{-1} B}$
DA	$\mathbf{R_s D_c^{-1} B D_u^{-1}}$
DAD	$\mathbf{R_s D_c^{-1} B}$
DAD1/2	$\mathbf{R_s D_c^{-1} B D_u^{-1/2}}$
DAID1/2	$\mathbf{R_s D_c^{-1} B D_u^{-1} (I + D_u)^{1/2}}$
COLUMN=	**Matrix Formula**
B	$\frac{1}{f}\mathbf{S_v D_r^{-1} A D_u^{-1}}$
BD	$\frac{1}{f}\mathbf{S_v D_r^{-1} A}$
DB	$\mathbf{C_s D_r^{-1} A D_u^{-1}}$
DBD	$\mathbf{C_s D_r^{-1} A}$
DBD1/2	$\mathbf{C_s D_r^{-1} A D_u^{-1/2}}$
DBID1/2	$\mathbf{C_s D_r^{-1} A D_u^{-1} (I + D_u)^{1/2}}$
MCA COLUMN=	**Matrix Formula**
B	Not allowed
BD	Not allowed
DB	$\mathbf{C_s D_r^{-1} B D_u^{-2}}$
DBD	$\mathbf{C_s D_r^{-1} B D_u^{-1}}$
DBD1/2	$\mathbf{C_s D_r^{-1} B D_u^{-3/2}}$
DBID1/2	$\mathbf{C_s D_r^{-1} B D_u^{-2} (I + D_u)^{1/2}}$

Statistics That Aid Interpretation

The partial contributions to inertia, squared cosines, quality of representation, inertia, and mass provide additional information about the coordinates. These statistics are displayed by default. Include the SHORT or NOPRINT option in the PROC CORRESP statement to avoid having these statistics displayed.

These statistics pertain to the default PROFILE=BOTH coordinates, no matter what values you specify for the ROW=, COLUMN=, or PROFILE= option. Let $\mathrm{sq}(\cdot)$ be a matrix-valued function denoting element-wise squaring of the argument matrix. Let t be the total inertia (the sum of the elements in $\mathbf{D}_\mathbf{u}^\mathbf{2}$).

In MCA, let $\mathbf{D}_\mathbf{s}$ be the Burt table partition containing the intersection of the supplementary columns and the supplementary rows. The matrix $\mathbf{D}_\mathbf{s}$ is a diagonal matrix of marginal frequencies of the supplemental columns of the binary matrix $\mathbf{Z}$. Let p be the number of rows in this design matrix. The statistics are defined in Table 41.5.

Table 41.5 Statistics That Aid Interpretation

Statistic	Matrix Formula
Row partial contributions to inertia	$\mathbf{D}_\mathbf{r}^{-\mathbf{1}}\mathrm{sq}(\mathbf{A})$
Column partial contributions to inertia	$\mathbf{D}_\mathbf{c}^{-\mathbf{1}}\mathrm{sq}(\mathbf{B})$
Row squared cosines	$\mathrm{diag}(\mathrm{sq}(\mathbf{AD}_\mathbf{u})\mathbf{1})^{-\mathbf{1}}\mathrm{sq}(\mathbf{AD}_\mathbf{u})$
Column squared cosines	$\mathrm{diag}(\mathrm{sq}(\mathbf{BD}_\mathbf{u})\mathbf{1})^{-\mathbf{1}}\mathrm{sq}(\mathbf{BD}_\mathbf{u})$
Row mass	$\mathbf{r}$
Column mass	$\mathbf{c}$
Row inertia	$\frac{1}{t}\mathbf{D}_\mathbf{r}^{-\mathbf{1}}\mathrm{sq}(\mathbf{AD}_\mathbf{u})\mathbf{1}$
Column inertia	$\frac{1}{t}\mathbf{D}_\mathbf{c}^{-\mathbf{1}}\mathrm{sq}(\mathbf{BD}_\mathbf{u})\mathbf{1}$
Supplementary row squared cosines	$\mathrm{diag}(\mathrm{sq}(\mathbf{R}_\mathbf{s} - \mathbf{1c}')\mathbf{D}_\mathbf{c}^{-\mathbf{1}}\mathbf{1})^{-\mathbf{1}}\mathrm{sq}(\mathbf{R}_\mathbf{s}\mathbf{D}_\mathbf{c}^{-\mathbf{1}}\mathbf{B})$
Supplementary column squared cosines	$\mathrm{diag}(\mathrm{sq}(\mathbf{C}_\mathbf{s} - \mathbf{1r}')\mathbf{D}_\mathbf{r}^{-\mathbf{1}}\mathbf{1})^{-\mathbf{1}}\mathrm{sq}(\mathbf{C}_\mathbf{s}\mathbf{D}_\mathbf{r}^{-\mathbf{1}}\mathbf{A})$
MCA supplementary column squared cosines	$\mathbf{D}_\mathbf{s}(p\mathbf{I} - \mathbf{D}_\mathbf{s})^{-\mathbf{1}}\mathrm{sq}(\mathbf{C}_\mathbf{s}\mathbf{D}_\mathbf{r}^{-\mathbf{1}}\mathbf{BD}_\mathbf{u}^{-\mathbf{1}})$

The quality of representation in the DIMENS=n dimensional display of any point is the sum of its squared cosines over only the n dimensions. Inertia and mass are not defined for supplementary points.

A table that summarizes the partial contributions to inertia table is also computed. The points that best explain the inertia of each dimension and the dimension to which each point contributes the most inertia are indicated. The output data set variable names for this table are Best1–Best$n$ (where DIMENS=n) and Best. The Best column contains the dimension number of the largest partial contribution to inertia for each point (the index of the maximum value in each row of $\mathbf{D}_\mathbf{r}^{-\mathbf{1}}\mathrm{sq}(\mathbf{A})$ or $\mathbf{D}_\mathbf{c}^{-\mathbf{1}}\mathrm{sq}(\mathbf{B})$).

For each row, the Best1–Best$n$ columns contain either the corresponding value of Best, if the point is one of the biggest contributors to the dimension's inertia, or 0 if it is not. Specifically, Best1 contains the value of

Best for the point with the largest contribution to dimension one's inertia. A cumulative proportion sum is initialized to this point's partial contribution to the inertia of dimension one. If this sum is less than the value for the MININERTIA= option, then Best1 contains the value of Best for the point with the second-largest contribution to dimension one's inertia. Otherwise, this point's Best1 is 0. This point's partial contribution to inertia is added to the sum. This process continues for the point with the third-largest partial contribution, and so on, until adding a point's contribution to the sum increases the sum beyond the value of the MININERTIA= option. This same algorithm is then used for Best2, and so on.

For example, the following table contains contributions to inertia and the corresponding Best variables. The contribution to inertia variables are proportions that sum to 1 within each column. The first point makes its greatest contribution to the inertia of dimension two, so Best for point one is set to 2, and Best1–Best3 for point one must all be 0 or 2. The second point also makes its greatest contribution to the inertia of dimension two, so Best for point two is set to 2, and Best1–Best3 for point two must all be 0 or 2, and so on.

Assume MININERTIA=0.8, the default. Table 41.6 shows some contributions to inertia. In dimension one, the largest contribution is 0.41302 for the fourth point, so Best1 is set to 1, the value of Best for the fourth point. Because this value is less than 0.8, the second-largest value (0.36456 for point five) is found and its Best1 is set to its Best's value of 1. Because $0.41302 + 0.36456 = 0.77758$ is less than 0.8, the third point (0.0882 at point eight) is found and Best1 is set to 3, because the contribution to dimension three for that point is greater than the contribution to dimension one. This increases the sum of the partial contributions to greater than 0.8, so the remaining Best1 values are all 0.

Table 41.6 Best Statistics

Contr1	Contr2	Contr3	Best1	Best2	Best3	Best
0.01593	0.32178	0.07565	0	2	2	2
0.03014	0.24826	0.07715	0	2	2	2
0.00592	0.02892	0.02698	0	0	0	2
0.41302	0.05191	0.05773	1	0	0	1
0.36456	0.00344	0.15565	1	0	1	1
0.03902	0.30966	0.11717	0	2	2	2
0.00019	0.01840	0.00734	0	0	0	2
0.08820	0.00527	0.16555	3	0	3	3
0.01447	0.00024	0.03851	0	0	0	3
0.02855	0.01213	0.27827	0	0	3	3

Displayed Output

The display options control the amount of displayed output. By default, the following information is displayed:

- an inertia and chi-square decomposition table including the total inertia, the principal inertias of each dimension (eigenvalues), the singular values (square roots of the eigenvalues), each dimension's percentage of inertia, a horizontal bar chart of the percentages, and the total chi-square with its degrees of freedom and decomposition. The chi-square statistics and degrees of freedom are valid only when the constructed table is an ordinary two-way contingency table.
- the coordinates of the rows and columns on the dimensions
- the mass, relative contribution to the total inertia, and quality of representation in the DIMENS=*n* dimensional display of each row and column
- the squared cosines of the angles between each axis and a vector from the origin to the point
- the partial contributions of each point to each dimension's inertia
- the table of indicators of which points best explain the inertia of each dimension

Specific display options and combinations of options display output as follows.

If you specify the OBSERVED or ALL option and you do not specify PRINT=PERCENT, PROC CORRESP displays the following:

- the contingency table, including the row and column marginal frequencies; or with BINARY, the binary table; or the Burt table in MCA
- the supplementary rows
- the supplementary columns

If you specify the OBSERVED or ALL option, with the PRINT=PERCENT or PRINT=BOTH option, PROC CORRESP displays the following:

- the contingency table or Burt table in MCA, scaled to percentages, including the row and column marginal percentages
- the supplementary rows, scaled to percentages
- the supplementary columns, scaled to percentages

If you specify the EXPECTED or ALL option and you do not specify PRINT=PERCENT, PROC CORRESP displays the product of the row marginals and the column marginals divided by the grand frequency of the observed frequency table. For ordinary two-way contingency tables, these are the expected frequencies under the hypothesis of row and column independence.

If you specify the EXPECTED or ALL option with the PRINT=PERCENT or PRINT=BOTH option, PROC CORRESP displays the product of the row marginals and the column marginals divided by the grand frequency of the observed percentages table. For ordinary two-way contingency tables, these are the expected percentages under the hypothesis of row and column independence.

If you specify the DEVIATION or ALL option and you do not specify PRINT=PERCENT, PROC CORRESP displays the observed minus expected frequencies. For ordinary two-way contingency tables, these are the expected frequencies under the hypothesis of row and column independence.

If you specify the DEVIATION or ALL option with the PRINT=PERCENT or PRINT=BOTH option, PROC CORRESP displays the observed minus expected percentages. For ordinary two-way contingency tables, these are the expected percentages under the hypothesis of row and column independence.

If you specify the CELLCHI2 or ALL option and you do not specify PRINT=PERCENT, PROC CORRESP displays contributions to the total chi-square test statistic, including the row and column marginals. The intersection of the marginals contains the total chi-square statistic.

If you specify the CELLCHI2 or ALL option with the PRINT=PERCENT or the PRINT=BOTH option, PROC CORRESP displays contributions to the total chi-square, scaled to percentages, including the row and column marginals.

If you specify the RP or ALL option and you do not specify PRINT=PERCENT, PROC CORRESP displays the row profiles and the supplementary row profiles.

If you specify the RP or ALL option with the PRINT=PERCENT or the PRINT=BOTH option, PROC CORRESP displays the row profiles (scaled to percentages) and the supplementary row profiles (scaled to percentages).

If you specify the CP or ALL option and you do not specify PRINT=PERCENT, PROC CORRESP displays the column profiles and the supplementary column profiles.

If you specify the CP or ALL option with the PRINT=PERCENT or PRINT=BOTH option, PROC CORRESP displays the column profiles (scaled to percentages) and the supplementary column profiles (scaled to percentages).

If you do not specify the NOPRINT option, PROC CORRESP displays the inertia and chi-square decomposition table. This includes the nonzero singular values of the contingency table (or, in MCA, the binary matrix $\mathbf{Z}$ used to create the Burt table), the nonzero principal inertias (or eigenvalues) for each dimension, the total inertia, the total chi-square, the decomposition of chi-square, the chi-square degrees of freedom (appropriate only when the table is an ordinary two-way contingency table), the percentage of the total chi-square and inertia for each dimension, and a bar chart of the percentages.

If you specify the MCA option and you do not specify the NOPRINT option, PROC CORRESP displays the adjusted inertias. This includes the nonzero adjusted inertias, percentages, cumulative percentages, and a bar chart of the percentages.

If you do not specify the NOROW, NOPRINT, or MCA option, PROC CORRESP displays the row coordinates and the supplementary row coordinates (displayed when there are supplementary row points).

If you do not specify the NOROW, NOPRINT, MCA, or SHORT option, PROC CORRESP displays the following:

- the summary statistics for the row points, including the quality of representation of the row points in the n-dimensional display, the mass, and the relative contributions to inertia

- the quality of representation of the supplementary row points in the *n*-dimensional display (displayed when there are supplementary row points)
- the partial contributions to inertia for the row points
- the table of indicators of which row points best explain the inertia of each dimension
- the squared cosines for the row points
- the squared cosines for the supplementary row points (displayed when there are supplementary row points)

If you do not specify the NOCOLUMN or NOPRINT option, PROC CORRESP displays the column coordinates and the supplementary column coordinates (displayed when there are supplementary column points).

If you do not specify the NOCOLUMN, NOPRINT, or SHORT option, PROC CORRESP displays the following:

- the summary statistics for the column points, including the quality of representation of the column points in the *n*-dimensional display, the mass, and the relative contributions to inertia for the supplementary column points
- the quality of representation of the supplementary column points in the *n*-dimensional display (displayed when there are supplementary column points)
- the partial contributions to inertia for the column points
- the table of indicators of which column points best explain the inertia of each dimension
- the squared cosines for the column points
- the squared cosines for the supplementary column points

ODS Table Names

PROC CORRESP assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 41.7 along with the PROC CORRESP statement options needed to produce the table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 41.7 ODS Tables Produced by PROC CORRESP

ODS Table Name	Description	Option
AdjInGreenacre	Greenacre Inertia Adjustment	GREENACRE
AdjInBenzecri	Benzécri Inertia Adjustment	BENZECRI
Binary	Binary table	OBSERVED, BINARY
BinaryPct	Binary table percentages	OBSERVED, BINARY*

Table 41.7 *continued*

ODS Table Name	Description	Option
Burt	Burt table	OBSERVED, MCA
BurtPct	Burt table percentages	OBSERVED, MCA*
CellChiSq	Contributions to chi-square	CELLCHI2
CellChiSqPct	Contributions, percentages	CELLCHI2*
ColBest	Col best indicators	Default
ColContr	Col contributions to inertia	Default
ColCoors	Col coordinates	Default
ColProfiles	Col profiles	CP
ColProfilesPct	Col profiles, percentages	CP*
ColQualMassIn	Col quality, mass, inertia	Default
ColSqCos	Col squared cosines	Default
DF	DF, chi-square (not displayed)	Default
Deviations	Observed - expected freqs	DEVIATIONS
DeviationsPct	Observed - expected percentages	DEVIATIONS*
Expected	Expected frequencies	EXPECTED
ExpectedPct	Expected percentages	EXPECTED*
Inertias	Inertia decomposition table	Default
Observed	Observed frequencies	OBSERVED
ObservedPct	Observed percentages	OBSERVED*
RowBest	Row best indicators	Default
RowContr	Row contributions to inertia	Default
RowCoors	Row coordinates	Default
RowProfiles	Row profiles	RP
RowProfilesPct	Row profiles, percentages	RP*
RowQualMassIn	Row quality, mass, inertia	Default
RowSqCos	Row squared cosines	Default
SupColCoors	Supp col coordinates	Default
SupColProfiles	Supp col profiles	CP
SupColProfilesPct	Supp col profiles, percentages	CP*
SupColQuality	Supp col quality	Default
SupCols	Supplementary col freq	OBSERVED
SupColsPct	Supplementary col percentages	OBSERVED*
SupColSqCos	Supp col squared cosines	Default
SupRows	Supplementary row freqs	OBSERVED
SupRowCoors	Supp row coordinates	Default
SupRowProfiles	Supp row profiles	RP
SupRowProfilesPct	Supp row profiles, percentages	RP*
SupRowQuality	Supp row quality	Default
SupRowsPct	Supplementary row percentages	OBSERVED*
SupRowSqCos	Supp row squared cosines	Default

*Percentages are displayed when you specify the PRINT=PERCENT or PRINT=BOTH option.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC CORRESP generates are listed in Table 41.8. Graphs are displayed by default when ODS graphics is enabled.

Table 41.8 Graphs Produced by PROC CORRESP

ODS Graph Name	Plot Description
ConfigPlot	Correspondence analysis plot
InertiaChart	Inertia and chi-square decomposition

Examples: CORRESP Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 41.1: Simple and Multiple Correspondence Analysis of Automobiles and Their Owners

(View the complete code for this example (crspex1.sas) in the example repository.)

In this example, PROC CORRESP creates a contingency table from categorical data and performs a simple correspondence analysis. The data are from a sample of individuals who were asked to provide information about themselves and their automobiles. The questions included origin of the automobile (American, Japanese, or European) and family status (single, married, single living with children, or married living with children).

The first steps read the input data and assign formats:

```
title1 'Automobile Owners and Auto Attributes';
title2 'Simple Correspondence Analysis';

proc format;
```

```
   value Origin  1 = 'American' 2 = 'Japanese' 3 = 'European';
   value Size    1 = 'Small'    2 = 'Medium'   3 = 'Large';
   value Type    1 = 'Family'   2 = 'Sporty'   3 = 'Work';
   value Home    1 = 'Own'      2 = 'Rent';
   value Sex     1 = 'Male'     2 = 'Female';
   value Income  1 = '1 Income' 2 = '2 Incomes';
   value Marital 1 = 'Single with Kids' 2 = 'Married with Kids'
                 3 = 'Single'           4 = 'Married';
run;

data Cars;
   missing a;
   input (Origin Size Type Home Income Marital Kids Sex) (1.) @@;
   * Check for End of Line;
   if n(of Origin -- Sex) eq 0 then do; input; return; end;
   marital = 2 * (kids le 0) + marital;
   format Origin Origin. Size Size. Type Type. Home Home.
          Sex Sex. Income Income. Marital Marital.;
   output;
   datalines;
1311122121211101211122011312110112112211221121211311221232112222122122 01
1211220231212212322111011221220221211101221121021311122111211101123111 01
2111121132112231211222022211221113111231312111023211222232212202212211 01
1221220221212202112122012211220211221101321122022131121113312261222211 01

   ... more lines ...

2121220112111221312211011212110222122202121211 01
;
```

PROC CORRESP is used to perform the simple correspondence analysis. The ALL option displays all tables, including the contingency table, chi-square information, profiles, and all results of the correspondence analysis. The CHI2P option displays the chi-square *p*-value. The TABLES statement specifies the row and column categorical variables. The results are displayed with ODS Graphics.

The following statements produce Output 41.1.1:

```
ods graphics on;

* Perform Simple Correspondence Analysis;
proc corresp data=Cars all chi2p;
   tables Marital, Origin;
run;
```

Correspondence analysis locates all the categories in a Euclidean space. The first two dimensions of this space are plotted to examine the associations among the categories.[3] Because the smallest dimension of this table is three, there is no loss of information when only two dimensions are plotted. The plot should be thought of as two different overlaid plots, one for each categorical variable. Distances between points within a variable have meaning, but distances between points from different variables do not.

[3]In this analysis, the chi-square statistic is not significantly different from 0. Hence, you would not reject the null hypothesis that the rows and columns are independent. If your goal is hypothesis testing, you might stop at this point and not proceed to interpret the graphical and tabular results. If your goal is exploratory data analysis, you might proceed and interpret the results. This example will proceed, because it is intended to be a small, simple teaching example.

Output 41.1.1 Simple Correspondence Analysis

Automobile Owners and Auto Attributes
Simple Correspondence Analysis

The CORRESP Procedure

Contingency Table	American	European	Japanese	Sum
Married	37	14	51	102
Married with Kids	52	15	44	111
Single	33	15	63	111
Single with Kids	6	1	8	15
Sum	128	45	166	339

Chi-Square Statistic Expected Values	American	European	Japanese
Married	38.5133	13.5398	49.9469
Married with Kids	41.9115	14.7345	54.3540
Single	41.9115	14.7345	54.3540
Single with Kids	5.6637	1.9912	7.3451

Observed Minus Expected Values	American	European	Japanese
Married	-1.5133	0.4602	1.0531
Married with Kids	10.0885	0.2655	-10.3540
Single	-8.9115	0.2655	8.6460
Single with Kids	0.3363	-0.9912	0.6549

Contributions to the Total Chi-Square Statistic	American	European	Japanese	Sum
Married	0.05946	0.01564	0.02220	0.09730
Married with Kids	2.42840	0.00478	1.97235	4.40553
Single	1.89482	0.00478	1.37531	3.27492
Single with Kids	0.01997	0.49337	0.05839	0.57173
Sum	4.40265	0.51858	3.42825	8.34947

Row Profiles	American	European	Japanese
Married	0.362745	0.137255	0.500000
Married with Kids	0.468468	0.135135	0.396396
Single	0.297297	0.135135	0.567568
Single with Kids	0.400000	0.066667	0.533333

Column Profiles	American	European	Japanese
Married	0.289063	0.311111	0.307229
Married with Kids	0.406250	0.333333	0.265060
Single	0.257813	0.333333	0.379518
Single with Kids	0.046875	0.022222	0.048193

Inertia and Chi-Square Decomposition

Singular Value	Principal Inertia	Chi-Square	Percent	Cumulative Percent	0 20 40 60 80
0.15122	0.02287	7.7516	92.84	92.84	
0.04200	0.00176	0.5979	7.16	100.00	
	0.02463	8.3495	100.00		Pr > Chisq = .2136 DF = 6

Row Coordinates

	Dim1	Dim2
Married	-0.0278	0.0134
Married with Kids	0.1991	0.0064
Single	-0.1716	0.0076
Single with Kids	-0.0144	-0.1947

Summary Statistics for the Row Points

	Quality	Mass	Inertia
Married	1.0000	0.3009	0.0117
Married with Kids	1.0000	0.3274	0.5276
Single	1.0000	0.3274	0.3922
Single with Kids	1.0000	0.0442	0.0685

Partial Contributions to Inertia for the Row Points

	Dim1	Dim2
Married	0.0102	0.0306
Married with Kids	0.5678	0.0076
Single	0.4217	0.0108
Single with Kids	0.0004	0.9511

Indices of the Coordinates That Contribute Most to Inertia for the Row Points

	Dim1	Dim2	Best
Married	0	0	2
Married with Kids	1	0	1
Single	1	0	1
Single with Kids	0	2	2

Squared Cosines for the Row Points

	Dim1	Dim2
Married	0.8121	0.1879
Married with Kids	0.9990	0.0010
Single	0.9980	0.0020
Single with Kids	0.0054	0.9946

Column Coordinates

	Dim1	Dim2
American	0.1847	-0.0166
European	0.0013	0.1073
Japanese	-0.1428	-0.0163

Summary Statistics for the Column Points

	Quality	Mass	Inertia
American	1.0000	0.3776	0.5273
European	1.0000	0.1327	0.0621
Japanese	1.0000	0.4897	0.4106

Partial Contributions to Inertia for the Column Points

	Dim1	Dim2
American	0.5634	0.0590
European	0.0000	0.8672
Japanese	0.4366	0.0737

Indices of the Coordinates That Contribute Most to Inertia for the Column Points

	Dim1	Dim2	Best
American	1	0	1
European	0	2	2
Japanese	1	0	1

Squared Cosines for the Column Points

	Dim1	Dim2
American	0.9920	0.0080
European	0.0001	0.9999
Japanese	0.9871	0.0129

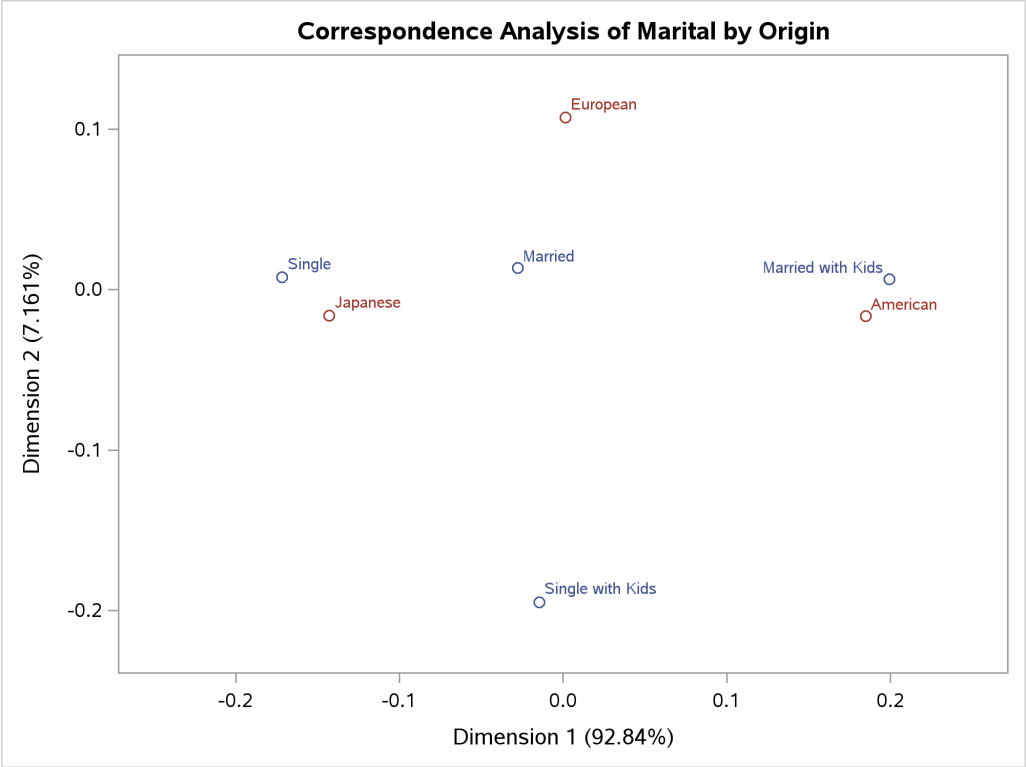

To interpret the plot, start by interpreting the row points separately from the column points. The European point is near and to the left of the centroid, so it makes a relatively small contribution to the chi-square statistic (because it is near the centroid), it contributes almost nothing to the inertia of dimension one (because its coordinate on dimension one has a small absolute value relative to the other column points), and it makes a relatively large contribution to the inertia of dimension two (because its coordinate on dimension two has a large absolute value relative to the other column points). Its squared cosines for dimension one and two, approximately 0 and 1, respectively, indicate that its position is almost completely determined by its location on dimension two. Its quality of display is 1.0, indicating perfect quality, because the table is two-dimensional after the centering. The American and Japanese points are far from the centroid, and they lie along dimension one. They make relatively large contributions to the chi-square statistic and the inertia of dimension one. The horizontal dimension seems to be largely determined by Japanese versus American automobile ownership.

In the row points, the Married point is near the centroid, and the Single with Kids point has a small coordinate on dimension one that is near zero. The horizontal dimension seems to be largely determined by the Single versus the Married with Kids points. The two interpretations of dimension one show the association with being Married with Kids and owning an American auto, and being single and owning a Japanese auto. The fact that the Married with Kids point is close to the American point and the fact that the Japanese point is near the Single point should be ignored. Distances between row and column points are not defined. The plot shows that more people who are married with kids than you would expect if the rows and columns were independent drive an American auto, and more people who are single than you would expect if the rows and columns were independent drive a Japanese auto.

In the second part of this example, PROC CORRESP creates a Burt table from categorical data and performs a multiple correspondence analysis. The variables used in this example are Origin, Size, Type, Income, Home, Marital, and Sex. MCA specifies multiple correspondence analysis, OBSERVED displays the Burt table. The TABLES statement with only a single variable list and no comma creates the Burt table.

The following statements produce Output 41.1.2:

```
title2 'Multiple Correspondence Analysis';

* Perform Multiple Correspondence Analysis;
proc corresp mca observed data=Cars;
   tables Origin Size Type Income Home Marital Sex;
run;
```

Output 41.1.2 Multiple Correspondence Analysis

Automobile Owners and Auto Attributes
Multiple Correspondence Analysis

The CORRESP Procedure

Burt Table

	American	European	Japanese	Large	Medium	Small	Family	Sporty	Work
American	125	0	0	36	60	29	81	24	20
European	0	44	0	4	20	20	17	23	4
Japanese	0	0	165	2	61	102	76	59	30
Large	36	4	2	42	0	0	30	1	11
Medium	60	20	61	0	141	0	89	39	13
Small	29	20	102	0	0	151	55	66	30
Family	81	17	76	30	89	55	174	0	0
Sporty	24	23	59	1	39	66	0	106	0
Work	20	4	30	11	13	30	0	0	54
1 Income	58	18	74	20	57	73	69	55	26
2 Incomes	67	26	91	22	84	78	105	51	28
Own	93	38	111	35	106	101	130	71	41
Rent	32	6	54	7	35	50	44	35	13
Married	37	13	51	9	42	50	50	35	16
Married with Kids	50	15	44	21	51	37	79	12	18
Single	32	15	62	11	40	58	35	57	17
Single with Kids	6	1	8	1	8	6	10	2	3
Female	58	21	70	17	70	62	83	44	22
Male	67	23	95	25	71	89	91	62	32

Output 41.1.2 *continued*

Automobile Owners and Auto Attributes
Multiple Correspondence Analysis

The CORRESP Procedure

Burt Table

	1 Income	2 Incomes	Own	Rent	Married	Married with Kids	Single	Single with Kids	Female	Male
American	58	67	93	32	37	50	32	6	58	67
European	18	26	38	6	13	15	15	1	21	23
Japanese	74	91	111	54	51	44	62	8	70	95
Large	20	22	35	7	9	21	11	1	17	25
Medium	57	84	106	35	42	51	40	8	70	71
Small	73	78	101	50	50	37	58	6	62	89
Family	69	105	130	44	50	79	35	10	83	91
Sporty	55	51	71	35	35	12	57	2	44	62
Work	26	28	41	13	16	18	17	3	22	32
1 Income	150	0	80	70	10	27	99	14	47	103
2 Incomes	0	184	162	22	91	82	10	1	102	82
Own	80	162	242	0	76	106	52	8	114	128
Rent	70	22	0	92	25	3	57	7	35	57
Married	10	91	76	25	101	0	0	0	53	48
Married with Kids	27	82	106	3	0	109	0	0	48	61
Single	99	10	52	57	0	0	109	0	35	74
Single with Kids	14	1	8	7	0	0	0	15	13	2
Female	47	102	114	35	53	48	35	13	149	0
Male	103	82	128	57	48	61	74	2	0	185

Inertia and Chi-Square Decomposition

Singular Value	Principal Inertia	Chi-Square	Percent	Cumulative Percent
0.56934	0.32415	970.77	18.91	18.91
0.48352	0.23380	700.17	13.64	32.55
0.42716	0.18247	546.45	10.64	43.19
0.41215	0.16987	508.73	9.91	53.10
0.38773	0.15033	450.22	8.77	61.87
0.38520	0.14838	444.35	8.66	70.52
0.34066	0.11605	347.55	6.77	77.29
0.32983	0.10879	325.79	6.35	83.64
0.31517	0.09933	297.47	5.79	89.43
0.28069	0.07879	235.95	4.60	94.03
0.26115	0.06820	204.24	3.98	98.01
0.18477	0.03414	102.24	1.99	100.00
	1.71429	5133.92	100.00	

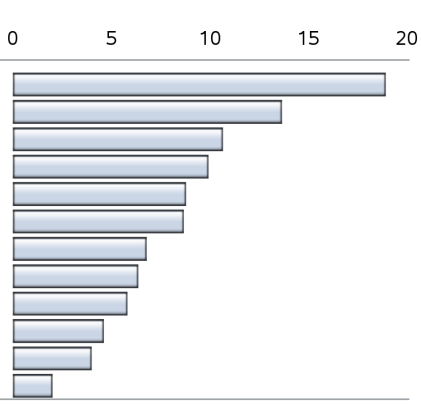

Degrees of Freedom = 324

Column Coordinates		
	Dim1	**Dim2**
American	-0.4035	0.8129
European	-0.0568	-0.5552
Japanese	0.3208	-0.4678
Large	-0.6949	1.5666
Medium	-0.2562	0.0965
Small	0.4326	-0.5258
Family	-0.4201	0.3602
Sporty	0.6604	-0.6696
Work	0.0575	0.1539
1 Income	0.8251	0.5472
2 Incomes	-0.6727	-0.4461
Own	-0.3887	-0.0943
Rent	1.0225	0.2480
Married	-0.4169	-0.7954
Married with Kids	-0.8200	0.3237
Single	1.1461	0.2930
Single with Kids	0.4373	0.8736
Female	-0.3365	-0.2057
Male	0.2710	0.1656

Summary Statistics for the Column Points			
	Quality	**Mass**	**Inertia**
American	0.4925	0.0535	0.0521
European	0.0473	0.0188	0.0724
Japanese	0.3141	0.0706	0.0422
Large	0.4224	0.0180	0.0729
Medium	0.0548	0.0603	0.0482
Small	0.3825	0.0646	0.0457
Family	0.3330	0.0744	0.0399
Sporty	0.4112	0.0453	0.0569
Work	0.0052	0.0231	0.0699
1 Income	0.7991	0.0642	0.0459
2 Incomes	0.7991	0.0787	0.0374
Own	0.4208	0.1035	0.0230
Rent	0.4208	0.0393	0.0604
Married	0.3496	0.0432	0.0581
Married with Kids	0.3765	0.0466	0.0561
Single	0.6780	0.0466	0.0561
Single with Kids	0.0449	0.0064	0.0796
Female	0.1253	0.0637	0.0462
Male	0.1253	0.0791	0.0372

Partial Contributions to Inertia for the Column Points	Dim1	Dim2
American	0.0268	0.1511
European	0.0002	0.0248
Japanese	0.0224	0.0660
Large	0.0268	0.1886
Medium	0.0122	0.0024
Small	0.0373	0.0764
Family	0.0405	0.0413
Sporty	0.0610	0.0870
Work	0.0002	0.0023
1 Income	0.1348	0.0822
2 Incomes	0.1099	0.0670
Own	0.0482	0.0039
Rent	0.1269	0.0103
Married	0.0232	0.1169
Married with Kids	0.0967	0.0209
Single	0.1889	0.0171
Single with Kids	0.0038	0.0209
Female	0.0223	0.0115
Male	0.0179	0.0093

Indices of the Coordinates That Contribute Most to Inertia for the Column Points	Dim1	Dim2	Best
American	0	2	2
European	0	0	2
Japanese	0	2	2
Large	0	2	2
Medium	0	0	1
Small	0	2	2
Family	2	0	2
Sporty	2	2	2
Work	0	0	2
1 Income	1	1	1
2 Incomes	1	1	1
Own	1	0	1
Rent	1	0	1
Married	0	2	2
Married with Kids	1	0	1
Single	1	0	1
Single with Kids	0	0	2
Female	0	0	1
Male	0	0	1

Squared Cosines for the Column Points		
	Dim1	**Dim2**
American	0.0974	0.3952
European	0.0005	0.0468
Japanese	0.1005	0.2136
Large	0.0695	0.3530
Medium	0.0480	0.0068
Small	0.1544	0.2281
Family	0.1919	0.1411
Sporty	0.2027	0.2085
Work	0.0006	0.0046
1 Income	0.5550	0.2441
2 Incomes	0.5550	0.2441
Own	0.3975	0.0234
Rent	0.3975	0.0234
Married	0.0753	0.2742
Married with Kids	0.3258	0.0508
Single	0.6364	0.0416
Single with Kids	0.0090	0.0359
Female	0.0912	0.0341
Male	0.0912	0.0341

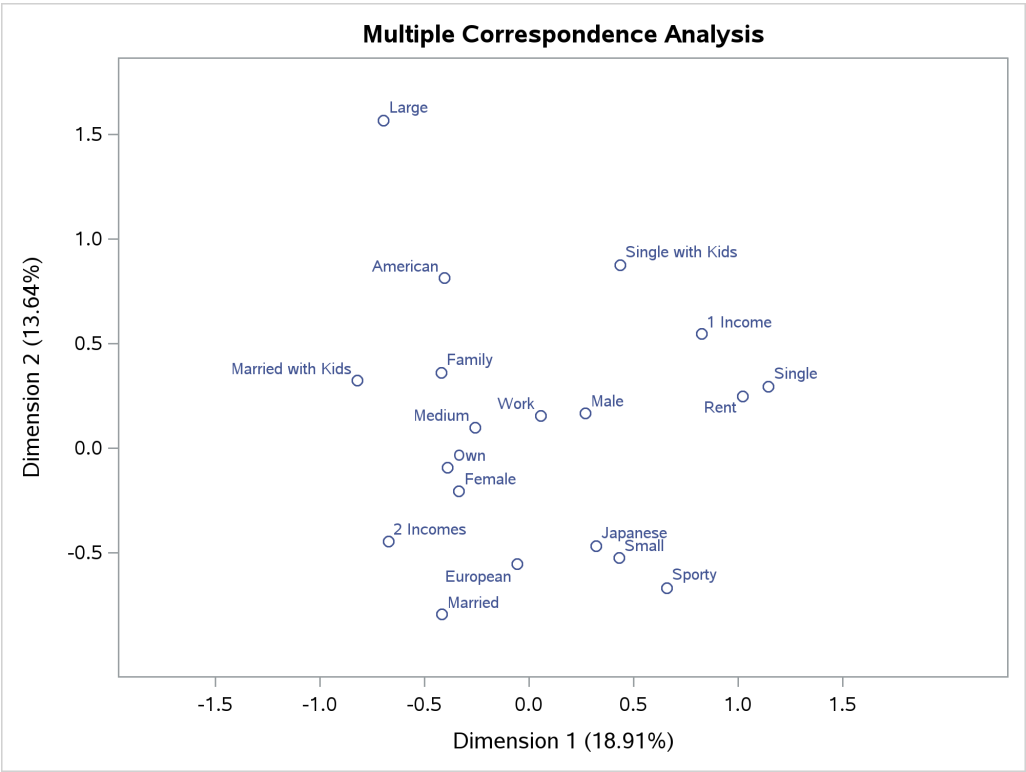

Multiple correspondence analysis locates all the categories in a Euclidean space. The first two dimensions of this space are plotted to examine the associations among the categories. The top-right quadrant of the plot

shows that the categories Single, Single with Kids, 1 Income, and Rent are associated. Proceeding clockwise, the categories Sporty, Small, and Japanese are associated. The bottom-left quadrant shows the association between being married, owning your own home, and having two incomes. Having children is associated with owning a large American family auto. Such information could be used in market research to identify target audiences for advertisements.

This interpretation is based on points found in approximately the same direction from the origin and in approximately the same region of the space. Distances between points do not have a straightforward interpretation in multiple correspondence analysis. The geometry of multiple correspondence analysis is not a simple generalization of the geometry of simple correspondence analysis (Greenacre and Hastie 1987; Greenacre 1988).

If you want to perform a multiple correspondence analysis and get scores for the individuals, you can specify the BINARY option to analyze the binary table, as in the following statements. In the interest of space, only the first 10 rows of coordinates are printed in Output 41.1.3.

```
title2 'Binary Table';

* Perform Multiple Correspondence Analysis;
proc corresp data=Cars binary;
   ods select RowCoors;
   tables Origin Size Type Income Home Marital Sex;
run;
```

Output 41.1.3 Correspondence Analysis of a Binary Table

Automobile Owners and Auto Attributes
Binary Table

The CORRESP Procedure

Row Coordinates

	Dim1	Dim2
1	-0.4093	1.0878
2	0.8198	-0.2221
3	-0.2193	-0.5328
4	0.4382	1.1799
5	-0.6750	0.3600
6	-0.1778	0.1441
7	-0.9375	0.6846
8	-0.7405	-0.1539
9	-0.3027	-0.2749
10	-0.7263	-0.0803

Example 41.2: Simple Correspondence Analysis of U.S. Population

(View the complete code for this example (crspex2.sas) in the example repository.)

In this example, PROC CORRESP reads an existing contingency table with supplementary observations and performs a simple correspondence analysis. The data are populations of the 50 U.S. states, grouped into regions, for each of the census years from 1920 to 1970 (US Bureau of the Census 1979). Alaska and Hawaii are treated as supplementary regions, because they were not states during this entire period and are not physically connected to the other 48 states. Consequently, it is reasonable to expect that population changes in these two states operate differently from population changes in the other states. The correspondence analysis is performed giving the supplementary points negative weight, and then the coordinates for the supplementary points are computed in the solution defined by the other points.

The initial DATA step reads the table, provides labels for the years, flags the supplementary rows with negative weights, and specifies absolute weights of 1000 for all observations because the data were originally reported in units of 1000 people.

In the PROC CORRESP statement, PRINT=PERCENT and the display options display the table of cell percentages (OBSERVED), cell contributions to the total chi-square scaled to sum to 100 (CELLCHI2), row profile rows that sum to 100 (RP), and column profile columns that sum to 100 (CP). The SHORT option specifies that the correspondence analysis summary statistics, contributions to inertia, and squared cosines should not be displayed. Because the data are already in table form, a VAR statement is used to read the table. Row labels are specified with the ID statement, and column labels come from the variable labels. The WEIGHT statement flags the supplementary observations and restores the table values to populations.

The following statements produce Output 41.2.1:

```
title 'United States Population, 1920-1970';

data USPop;

   * Regions:
   * New England     - ME, NH, VT, MA, RI, CT.
   * Great Lakes     - OH, IN, IL, MI, WI.
   * South Atlantic  - DE, MD, DC, VA, WV, NC, SC, GA, FL.
   * Mountain        - MT, ID, WY, CO, NM, AZ, UT, NV.
   * Pacific         - WA, OR, CA.
   *
   * Note: Multiply data values by 1000 to get populations.;

   input Region $14. y1920 y1930 y1940 y1950 y1960 y1970;

   label y1920 = '1920'    y1930 = '1930'    y1940 = '1940'
         y1950 = '1950'    y1960 = '1960'    y1970 = '1970';

   if region = 'Hawaii' or region = 'Alaska'
      then w = -1000;       /* Flag Supplementary Observations */
      else w =  1000;

   datalines;
New England        7401  8166  8437  9314 10509 11842
NY, NJ, PA        22261 26261 27539 30146 34168 37199
```

```
Great Lakes       21476 25297 26626 30399 36225 40252
Midwest           12544 13297 13517 14061 15394 16319
South Atlantic    13990 15794 17823 21182 25972 30671
KY, TN, AL, MS     8893  9887 10778 11447 12050 12803
AR, LA, OK, TX    10242 12177 13065 14538 16951 19321
Mountain           3336  3702  4150  5075  6855  8282
Pacific            5567  8195  9733 14486 20339 25454
Alaska               55    59    73   129   226   300
Hawaii              256   368   423   500   633   769
;

ods graphics on;

* Perform Simple Correspondence Analysis;
proc corresp data=uspop print=percent observed cellchi2 rp cp chi2p
     short plot(flip);
   var y1920 -- y1970;
   id Region;
   weight w;
run;
```

The contingency table shows that the population of all regions increased over this time period. The row profiles show that population increased at a different rate for the different regions. There was a small increase in population in the Midwest, for example, but the population more than quadrupled in the Pacific region over the same period. The column profiles show that in 1920, the U.S. population was concentrated in the NY, NJ, PA, Great Lakes, Midwest, and South Atlantic regions. With time, the population shifted more to the South Atlantic, Mountain, and Pacific regions. This is also clear from the correspondence analysis. The inertia and chi-square decomposition table shows that there are five nontrivial dimensions in the table, but the association between the rows and columns is almost entirely one-dimensional.

Output 41.2.1 United States Population, 1920–1970

United States Population, 1920-1970

The CORRESP Procedure

Contingency Table

Percents	1920	1930	1940	1950	1960	1970	Sum
New England	0.830	0.916	0.946	1.045	1.179	1.328	6.245
NY, NJ, PA	2.497	2.946	3.089	3.382	3.833	4.173	19.921
Great Lakes	2.409	2.838	2.987	3.410	4.064	4.516	20.224
Midwest	1.407	1.492	1.516	1.577	1.727	1.831	9.550
South Atlantic	1.569	1.772	1.999	2.376	2.914	3.441	14.071
KY, TN, AL, MS	0.998	1.109	1.209	1.284	1.352	1.436	7.388
AR, LA, OK, TX	1.149	1.366	1.466	1.631	1.902	2.167	9.681
Mountain	0.374	0.415	0.466	0.569	0.769	0.929	3.523
Pacific	0.625	0.919	1.092	1.625	2.282	2.855	9.398
Sum	11.859	13.773	14.771	16.900	20.020	22.677	100.000

Output 41.2.1 *continued*

Supplementary Rows						
Percents	**1920**	**1930**	**1940**	**1950**	**1960**	**1970**
Alaska	0.006170	0.006619	0.008189	0.014471	0.025353	0.033655
Hawaii	0.028719	0.041283	0.047453	0.056091	0.071011	0.086268

Contributions to the Total Chi-Square Statistic							
Percents	**1920**	**1930**	**1940**	**1950**	**1960**	**1970**	**Sum**
New England	0.937	0.314	0.054	0.009	0.352	0.469	2.135
NY, NJ, PA	0.665	1.287	0.633	0.006	0.521	2.265	5.378
Great Lakes	0.004	0.085	0.000	0.001	0.005	0.094	0.189
Midwest	5.749	2.039	0.684	0.072	1.546	4.472	14.563
South Atlantic	0.509	1.231	0.259	0.000	0.285	1.688	3.973
KY, TN, AL, MS	1.454	0.711	1.098	0.087	0.946	2.945	7.242
AR, LA, OK, TX	0.000	0.069	0.077	0.001	0.059	0.030	0.238
Mountain	0.391	0.868	0.497	0.098	0.498	1.834	4.187
Pacific	18.591	9.380	5.458	0.074	7.346	21.248	62.096
Sum	28.302	15.986	8.761	0.349	11.558	35.046	100.000

Row Profiles						
Percents	**1920**	**1930**	**1940**	**1950**	**1960**	**1970**
New England	13.2947	14.6688	15.1557	16.7310	18.8777	21.2722
NY, NJ, PA	12.5362	14.7888	15.5085	16.9766	19.2416	20.9484
Great Lakes	11.9129	14.0325	14.7697	16.8626	20.0943	22.3281
Midwest	14.7348	15.6193	15.8777	16.5167	18.0825	19.1691
South Atlantic	11.1535	12.5917	14.2093	16.8872	20.7060	24.4523
KY, TN, AL, MS	13.5033	15.0126	16.3655	17.3813	18.2969	19.4403
AR, LA, OK, TX	11.8687	14.1111	15.1401	16.8471	19.6433	22.3897
Mountain	10.6242	11.7898	13.2166	16.1624	21.8312	26.3758
Pacific	6.6453	9.7823	11.6182	17.2918	24.2784	30.3841

Supplementary Row Profiles						
Percents	**1920**	**1930**	**1940**	**1950**	**1960**	**1970**
Alaska	6.5321	7.0071	8.6698	15.3207	26.8409	35.6295
Hawaii	8.6809	12.4788	14.3438	16.9549	21.4649	26.0766

Column Profiles						
Percents	**1920**	**1930**	**1940**	**1950**	**1960**	**1970**
New England	7.0012	6.6511	6.4078	6.1826	5.8886	5.8582
NY, NJ, PA	21.0586	21.3894	20.9155	20.0109	19.1457	18.4023
Great Lakes	20.3160	20.6042	20.2221	20.1788	20.2983	19.9126
Midwest	11.8664	10.8303	10.2660	9.3337	8.6259	8.0730
South Atlantic	13.2343	12.8641	13.5363	14.0606	14.5532	15.1729
KY, TN, AL, MS	8.4126	8.0529	8.1857	7.5985	6.7521	6.3336
AR, LA, OK, TX	9.6888	9.9181	9.9227	9.6503	9.4983	9.5581
Mountain	3.1558	3.0152	3.1519	3.3688	3.8411	4.0971
Pacific	5.2663	6.6748	7.3921	9.6158	11.3968	12.5921

Inertia and Chi-Square Decomposition

Singular Value	Principal Inertia	Chi-Square	Percent	Cumulative Percent
0.10664	0.01137	10137829	98.16	98.16
0.01238	0.00015	136586	1.32	99.48
0.00658	0.00004	38540	0.37	99.85
0.00333	0.00001	9897	0.10	99.95
0.00244	0.00001	5310	0.05	100.00
	0.01159	10328162	100.00	

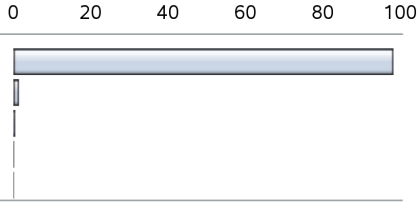

Pr > Chisq < .0001 DF = 40

Row Coordinates

	Dim1	Dim2
New England	0.0611	0.0132
NY, NJ, PA	0.0546	-0.0117
Great Lakes	0.0074	-0.0028
Midwest	0.1315	0.0186
South Atlantic	-0.0553	0.0105
KY, TN, AL, MS	0.1044	-0.0144
AR, LA, OK, TX	0.0131	-0.0067
Mountain	-0.1121	0.0338
Pacific	-0.2766	-0.0070

Supplementary Row Coordinates

	Dim1	Dim2
Alaska	-0.4152	0.0912
Hawaii	-0.1198	-0.0321

Column Coordinates

	Dim1	Dim2
1920	0.1642	0.0263
1930	0.1149	-0.0089
1940	0.0816	-0.0108
1950	-0.0046	-0.0125
1960	-0.0815	-0.0007
1970	-0.1335	0.0086

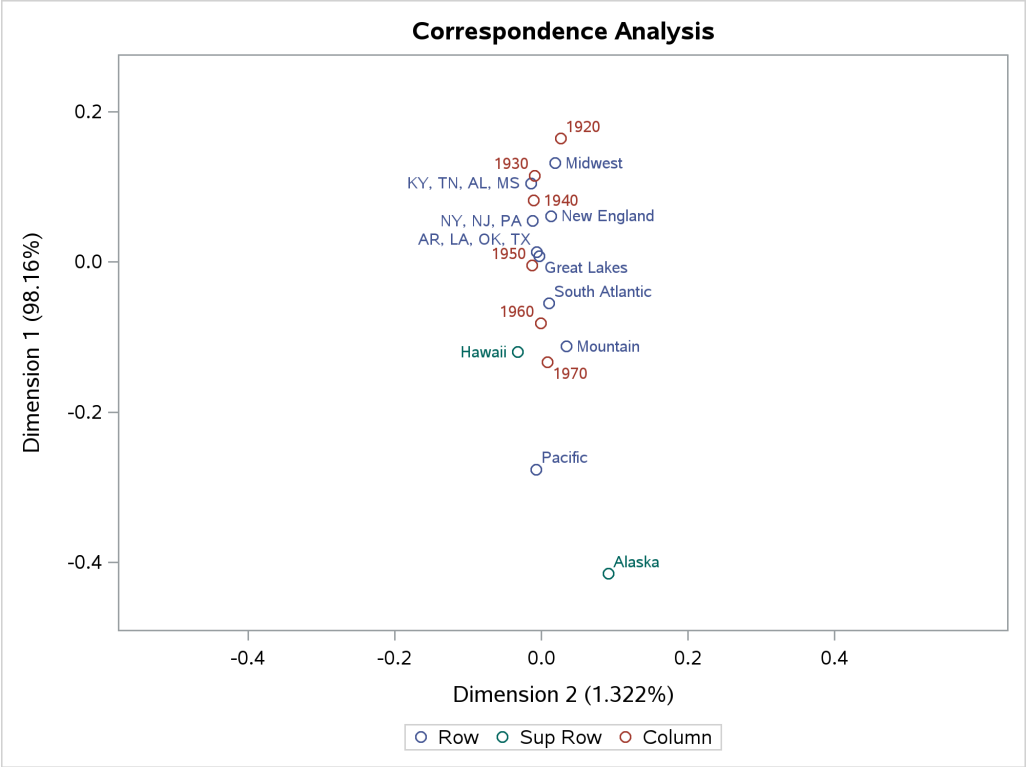

ODS Graphics is used to plot the results. The results are essentially one-dimensional. For results such as these, it is better to plot the first dimension vertically, as opposed to the default, which is horizontally. The vertical orientation has fewer opportunities for label collisions. Specifying PLOTS(FLIP) on the PROC CORRESP statement switches the vertical and horizontal axes to improve the graphical display.

The plot shows that the first dimension correctly orders the years. There is nothing in the correspondence analysis that forces this to happen; the analysis has no information about the inherent ordering of the column categories. The ordering of the regions and the ordering of the years reflect the shift over time of the U.S. population from the Northeast quadrant of the country to the South and to the West. The results show that the West and Southeast grew faster than the rest of the contiguous 48 states during this period.

The plot also shows that the growth pattern for Hawaii was similar to the growth pattern for the mountain states and that Alaska's growth was even more extreme than the Pacific states' growth. The row profiles confirm this interpretation.

The Pacific region is farther from the origin than all other active points. The Midwest is the extreme region in the other direction. The table of contributions to the total chi-square shows that 62% of the total chi-square statistic is contributed by the Pacific region, which is followed by the Midwest at over 14%. Similarly the two extreme years, 1920 and 1970, together contribute over 63% to the total chi-square, whereas the years nearer the origin of the plot contribute less.

Example 41.3: MCA and Large Data Sets

(View the complete code for this example (crspex3.sas) in the example repository.)

PROC CORRESP performs simple and multiple correspondence analysis (MCA). When you perform MCA on a large data set of categorical variables, you might run out of memory. This example shows ways to avoid the memory problem when you have large data sets.

PROC CORRESP accepts two forms of input:

- raw categorical data that you input by using a TABLES statement
- a crosstabulation that you input by using a VAR statement

You can create the crosstabulation (which in MCA is called a Burt table) by using PROC CORRESP and an ODS OUTPUT statement or by other means. The results (not shown) of the following steps show that PROC CORRESP performs the same MCA for both types of input:

```
proc corresp data=sashelp.heart short obs mca outc=c1;
   ods output burt=burt;
   tables DeathCause Sex Chol_Status BP_Status Weight_Status Smoking_Status;
run;

proc corresp data=burt short mca nvars=6 outc=c2;
   var _numeric_;
run;

data c2; length _NAME_ $ 26; set c2; run;

proc compare error note nosummary criterion=1e-10
   data=c1 compare=c2 method=relative(1);
run;
```

The only difference between the two output data sets is the length of the _NAME_ variable, and that difference is masked by the DATA step. PROC COMPARE reports that all values are exactly equal.

Although it is unlikely that you will ever run out of memory when you use VAR statement input, you might run out of memory when you have a large data set of categorical variables. As an alternative, you can create the Burt table in a more memory-efficient manner outside PROC CORRESP. The section "Using the %Burt and %Check Macros" on page 2712 illustrates how you can use the %Burt macro to solve the memory problem by creating the Burt table. It also demonstrates how you can use the %Check macro to confirm the accuracy of the Burt table generated by the %Burt macro. The section "The %Burt and %Check Macros" on page 2714 provide details about these two macros.

Using the %Burt and %Check Macros

To demonstrate how to use the %Burt and %Check macros, the following steps generate a Burt table from a small data set, sashelp.heart, and check the accuracy of the generated Burt table:

```
%let v = DeathCause Sex Chol_Status BP_Status Weight_Status Smoking_Status;
%burt(data=sashelp.heart, vars=&v)
%check(data=sashelp.heart, vars=&v)
```

The %Check macro checks the accuracy of the Burt table by comparing two sets of output data results:

- the Burt tables that are generated by the %Burt macro and PROC CORRESP, respectively
- the analysis results in the OUTC= data sets that are based on running PROC CORRESP with the input Burt table generated by using the %Burt macro and running PROC CORRESP directly with the raw data input, respectively

The %Check macro prints the following message twice, indicating that both the Burt tables and the MCA analysis results match:

```
NOTE: No unequal values were found. All values compared are exactly equal.
```

This message confirms that the %Burt macro successfully created the Burt table. But there is no reason to use the %Burt macro for a problem this small. Furthermore, in practice you never need the %Check macro. However, you can use the two macros together on small data sets or on a subset of the real data to assure yourself that the %Burt macro works correctly.

In general applications, you can perform MCA by running PROC CORRESP after calling the %Burt macro, as follows:

```
proc corresp data=burt short obs mca nvars=&_nvars;
   var _numeric_;
run;
```

When you use VAR statement input and the MCA option, you need to tell PROC CORRESP how many categorical variables went into constructing the Burt matrix. The %Burt macro stores that value in the global macro variable _NVARS.

To demonstrate the application with a large data set, the following steps use the %Burt macro to generate a Burt table from an artificial data set, x1, and check the generated Burt table by using the %Check macro:

```
%let nvars = 50;
data x1(drop=i j a);
   a = catx(' ', '1 2 3 4 5 6 7 8 9 0 abcd fgh jkl nop rst',
            'vwx zAB DEF HIJ LMN PQR TUV XYZ');
   length x1-x&nvars $ 36;
   array x[&nvars] $;
   do i = 1 to 1e5;
      do j = 1 to &nvars;
         x[j] = catx(' ', 'v', put(j, 2.),
                     substr(a, ceil(uniform(7) * (mod(j, 20) + 2))));
      end;
   output;
   end;
run;
```

```
%burt(data=x1, vars=x:)
%check(data=x1, vars=x:)
```

Again, the %Check macro shows that the %Burt macro works correctly here.

However, with the raw categorical data set input, when you run PROC CORRESP you might run out of memory. For example, you can set the number of variables to 100 and change the number of observations from 1E5 (100,000) to 1E6 (one million), and the %Burt macro followed by PROC CORRESP and the VAR statement will still run (depending on your system resources), whereas running PROC CORRESP and the TABLES statement input might cause you to run out of memory.

The %Burt and %Check Macros

The section provides details about the macros that are used in the section "Using the %Burt and %Check Macros" on page 2712.

As shown in the following code, the %Burt macro creates a Burt table by using a DATA step view for preprocessing, PROC FREQ for tabulation, and PROC TRANSPOSE and DATA steps for postprocessing.

```
                            /*---------------------------------------------*/
                            /* Create a Burt table.                        */
%macro burt(data=_last_,    /* Input SAS data set.                         */
            vars=,          /* Variable list.                              */
            options=nomiss, /* If you know that you have no missing        */
                            /* values, you might speed things up by        */
                            /* instead specifying options=.                */
            out=Burt,       /* Output SAS data set.                        */
           );;              /*---------------------------------------------*/

%let _t = %sysfunc(time());
%if &data eq _last_ %then %let data = &syslast;

proc contents data=&data(keep=&vars) noprint out=_cont(keep=length); run;

data _null_;
   set _cont end=eof;
   retain max 0;
   max = max(length, max);
   if eof then do;
      l = floor(log10(_n_)) + 1;
      call symputx('_opt', catx(' ', getoption('source'),
                   'varlenchk=', getoption('varlenchk')));
      call symputx('_f', l );
      call symputx('_lablen', l + max + 1, 'G');
      call symputx('_nvars', _n_, 'G');
      call symputx('_nomiss', lowcase(symget('options')) eq 'nomiss');
   end;
run;

options nosource varlenchk=nowarn;
data _r2(keep=_r1 _r2) / view=_r2;
   length _r1 _r2 $ &_lablen;
   set &data(keep=&vars);
   array _x[&_nvars] $ &vars;
```

```
   %if &_nomiss %then %do; if cmiss(of _x[*]) eq 0; %end;
   do _j = 1 to &_nvars;
      _r1 = cats('v', put(_j,z&_f..), _x[_j]); _r2 = _r1; output;
      do _k = _j+1 to &_nvars;
         _r2 = cats('v', put(_k,z&_f..), _x[_k]); output;
      end;
   end;
run;

proc freq data=_r2;
   tables _r1 *_r2 / noprint sparse out=_c2(drop=p:);
run;

proc transpose data=_c2 out=_t1(drop=_n: _l:);
   by _r1;
   var count;
   id _r2;
   idlabel _r2;
run;

proc transpose data=_t1 out=_t2(drop=_n:);
   var _numeric_;
   id _r1;
   idlabel _r1;
run;

data _t2(drop=_j);
   length _label_ $ &_lablen;
   set _t2 end=_eof;
   array _x[*] _numeric_;
   do _j = _n_ + 1 to dim(_x); _x[_j] = .; end;
   if _n_ eq 1 then call execute('%nrstr(%%)macro _lab;');
   call execute(catx(' ', 'label', vname(_x[_n_]), '=',
                     quote(trim(substr(_label_, &_f+2))), ';'));
   if _eof then call execute('%nrstr(%%)mend;');
run;

data &out(drop=_label_);
   label Label=' ';
   update _t1(rename=(_r1=_label_)) _t2;
   by _label_;
   Label = substr(_label_, &_f+2);
   %_lab
run;

proc datasets nolist;
   delete _cont _r2(memtype=view) _t1 _t2 _c2;
quit;

options &_opt;
%let _t = %sysfunc(round(%sysevalf((%sysfunc(time()) - &_t)/60), 0.1));
%put Burt macro time=&_t minutes.;
%mend;
```

The key to the increased memory efficiency is a DATA step view in the %Burt macro. It enables you to generate, use, and then discard the preprocessed input data without also storing them.

You can use the following macro to check the accuracy of the Burt table that is created from the %Burt macro:

```
%macro check(data=, vars=);
options varlenchk=nowarn nosource nonotes;
ods select none;

%let t1 = %sysfunc(time());
proc corresp mca data=&data short obs outc=c1;
   ods output burt=burt2;
   tables &vars;
run;
%let t1 = %sysfunc(round(%sysevalf((%sysfunc(time()) - &t1)/60), 0.1));
%put PROC CORRESP time=&t1 minutes.;
ods select all;

proc contents data=burt noprint
   out=_cont(keep=name varnum rename=(name=_n) where=(_n ne 'Label'));
run;

proc sort data=_cont out=_cont(drop=varnum); by varnum; run;

data _null_;
   merge _cont burt2 end=_eof;
   array _x[*] _numeric_;
   if _n_ eq 1 then call execute('%nrstr(%%)macro ren;');
   _l = vname(_x[_n_]);
   call execute(catx(' ', 'label', _l, '=', quote(trim(label)), ';'));
   call execute(catx(' ', 'rename', _l, '=', _n, ';'));
   if _eof then call execute('%nrstr(%%)mend;');
run;

data burt2;
   length label $ &_lablen;
   set burt2;
   array _x[*] _numeric_;
   format _numeric_;
   %ren
run;

proc corresp data=burt mca nvar=&_nvars noprint outc=c2;
   var _numeric_;
run;

data c1;
   length _NAME_ $ &_lablen;
   set c1;
run;

data c2;
   length _NAME_ $ &_lablen;
   set c2;
```

```
run;
options varlenchk=warn source notes;

proc compare error note nosummary data=burt compare=burt2;
run;

proc compare error note nosummary data=c1 compare=c2 method=relative(1);
run;
%mend;
```

To confirm the accuracy of the Burt table that is generated from the %Burt macro, the two PROC COMPARE steps at the end of the %Check macro should show no differences. For demonstrations of using these macros, see the section "Using the %Burt and %Check Macros" on page 2712.

Finally, the %Burt macro assumes that all variables are unformatted character variables; that is, the macro assumes that the variables are not numeric and that the values do not require formats. If you have numeric variables or formats, you can create a data set view that applies the formats and creates character variables, and then you can send that view to the macro as follows:

```
%let nvars = 10;
data x1(drop=i j);
   array x[&nvars];
   length x: 3;
   do i = 1 to 1e5;
      do j = 1 to &nvars;
         x[j] = ceil(uniform(7) * 3);
      end;
   output;
   end;
run;

proc format;
   value yn 1='Yes' 2='No' 3='Maybe';
run;

data z(drop=x:) / view=z;
   set x1;
   a = put(x1 , yn5.);
   b = put(x2 , yn5.);
   c = put(x3 , yn5.);
   d = put(x4 , yn5.);
   e = put(x5 , yn5.);
   f = put(x6 , yn5.);
   g = put(x7 , yn5.);
   h = put(x8 , yn5.);
   i = put(x9 , yn5.);
   j = put(x10, yn5.);
   length k $ 1;
   k = cats(mod(_n_, 5));
run;

%burt(data=z, vars=a b c d e f g h i j k)

%check(data=z, vars=a b c d e f g h i j k)
```

```
%burt(data=z, vars=a k b d c e g f h j i)

%check(data=z, vars=a k b d c e g f h j i)
```

If the formatted levels of the categorical variables are long, you can save memory by using formats to assign the levels. The first DATA step also saves space by using a LENGTH statement and a length of 3. This example would be more realistic if there were different formats for each variable so that a level from one categorical variable never appeared in another categorical variable. However, because the macro might encounter unrealistic data sets like this, it makes no attempt to create a Burt table that has the same variable names as the Burt table that comes from PROC CORRESP and the ODS OUTPUT statement. For an example like this, ODS encounters duplicate column names that it renames when it creates variable names. The %Burt macro instead incorporates the categorical variable number into the Burt table variable names to avoid duplicates. The variable labels are identical, and PROC CORRESP uses the labels, not the names. The %Check macro manages this difference by renaming variables so that only the values and labels are checked.

References

Benzécri, J. P. (1973). *L'analyse des données: T. 2, l'analyse des correspondances [Data analysis: T. 2, Correspondence analysis]*. Paris: Dunod.

Benzécri, J. P. (1979). "Sur le calcul des taux d'inertie dans l'analyse d'un questionnaire, addendum et erratum à [Bin. Mult.] [On the calculation of the rates of inertia in the analysis of a questionnaire, addendum and erratum in [Bin. Mult.]]." *Cahiers de l'Analyse des Données* 4:377–378.

Burt, C. (1950). "The Factorial Analysis of Qualitative Data." *British Journal of Psychology* 3:166–185.

Carroll, J. D., Green, P. E., and Schaffer, C. M. (1986). "Interpoint Distance Comparisons in Correspondence Analysis." *Journal of Marketing Research* 23:271–280.

Fisher, R. A. (1940). "The Precision of Discriminant Functions." *Annals of Eugenics* 10:422–429.

Gifi, A. (1990). *Nonlinear Multivariate Analysis*. New York: John Wiley & Sons.

Greenacre, M. J. (1984). *Theory and Applications of Correspondence Analysis*. London: Academic Press.

Greenacre, M. J. (1988). "Correspondence Analysis of Multivariate Categorical Data by Weighted Least-Squares." *Biometrika* 75:457–467.

Greenacre, M. J. (1989). "The Carroll-Green-Schaffer Scaling in Correspondence Analysis: A Theoretical and Empirical Appraisal." *Journal of Market Research* 26:358–365.

Greenacre, M. J. (1994). "Multiple and Joint Correspondence Analysis." In *Correspondence Analysis in the Social Sciences*, edited by M. J. Greenacre and J. Blasius, 141–161. London: Academic Press.

Greenacre, M. J., and Hastie, T. J. (1987). "The Geometric Interpretation of Correspondence Analysis." *Journal of the American Statistical Association* 82:437–447.

Guttman, L. (1941). "The Quantification of a Class of Attributes: A Theory and Method of Scale Construction." In *The Prediction of Personal Adjustment*, edited by P. Horst, P. Wallin, and L. Guttman, 319–348. New York: Social Science Research Council.

Hayashi, C. (1950). "On the Quantification of Qualitative Data from the Mathematico-statistical Point of View." *Annals of the Institute of Statistical Mathematics* 2:35–47.

Hirshfield, H. O. (1935). "A Connection between Correlation and Contingency." *Cambridge Philosophical Society Proceedings* 31:520–524.

Hoffman, D. L., and Franke, G. R. (1986). "Correspondence Analysis: Graphical Representation of Categorical Data in Marketing Research." *Journal of Marketing Research* 23:213–227.

Horst, P. (1935). "Measuring Complex Attitudes." *Journal of Social Psychology* 6:369–374.

Kobayashi, R. (1981). *An Introduction to Quantification Theory*. Tokyo: Japan Union of Scientists and Engineers.

Komazawa, T. (1982). *Quantification Theory and Data Processing*. Tokyo: Asakura-shoten.

Lebart, L., Morineau, A., and Tabard, N. (1977). *Techniques de la description statistique: Méthodes et logiciels pour l'analyse des grands tableaux [Techniques of statistical description: Methods and software for the analysis of large tables]*. Paris: Dunod.

Lebart, L., Morineau, A., and Warwick, K. M. (1984). *Multivariate Descriptive Statistical Analysis: Correspondence Analysis and Related Techniques for Large Matrices*. New York: John Wiley & Sons.

Nishisato, S. (1980). *Analysis of Categorical Data: Dual Scaling and Its Applications*. Toronto: University of Toronto Press.

Nishisato, S. (1982). *Quantification of Qualitative Data: Dual Scaling and Its Applications*. Tokyo: Asakura-shoten.

Richardson, M., and Kuder, G. F. (1933). "Making a Rating Scale That Measures." *Personnel Journal* 12:36–40.

Tenenhaus, M., and Young, F. W. (1985). "An Analysis and Synthesis of Multiple Correspondence Analysis, Optimal Scaling, Dual Scaling, Homogeneity Analysis, and Other Methods of Quantifying Categorical Multivariate Data." *Psychometrika* 50:91–119.

US Bureau of the Census (1979). *Statistical Abstract of the United States*. 100th ed. Washington, DC: US Government Printing Office.

Van der Heijden, P. G. M., and de Leeuw, J. (1985). "Correspondence Analysis Used Complementary to Loglinear Analysis." *Psychometrika* 50:429–447.

Chapter 42
The DISCRIM Procedure

Contents

Overview: DISCRIM Procedure **2722**
Getting Started: DISCRIM Procedure **2723**
Syntax: DISCRIM Procedure **2726**
- PROC DISCRIM Statement 2726
- BY Statement 2734
- CLASS Statement 2735
- FREQ Statement 2735
- ID Statement 2736
- PRIORS Statement 2736
- TESTCLASS Statement 2737
- TESTFREQ Statement 2737
- TESTID Statement 2737
- VAR Statement 2737
- WEIGHT Statement 2738

Details: DISCRIM Procedure **2738**
- Missing Values 2738
- Background 2738
- Posterior Probability Error-Rate Estimates 2746
- Saving and Using Calibration Information 2748
- Input Data Sets 2749
- Output Data Sets 2750
- Computational Resources 2754
- Displayed Output 2755
- ODS Table Names 2759

Examples: DISCRIM Procedure **2761**
- Example 42.1: Univariate Density Estimates and Posterior Probabilities 2761
- Example 42.2: Bivariate Density Estimates and Posterior Probabilities 2776
- Example 42.3: Normal-Theory Discriminant Analysis of Iris Data 2793
- Example 42.4: Linear Discriminant Analysis of Remote-Sensing Data on Crops . . . 2802

References **2811**

Overview: DISCRIM Procedure

For a set of observations containing one or more quantitative variables and a classification variable defining groups of observations, the DISCRIM procedure develops a discriminant criterion to classify each observation into one of the groups. The derived discriminant criterion from this data set can be applied to a second data set during the same execution of PROC DISCRIM. The data set that PROC DISCRIM uses to derive the discriminant criterion is called the *training* or *calibration* data set.

When the distribution within each group is assumed to be multivariate normal, a parametric method can be used to develop a discriminant function. The discriminant function, also known as a classification criterion, is determined by a measure of generalized squared distance (Rao 1973). The classification criterion can be based on either the individual within-group covariance matrices (yielding a quadratic function) or the pooled covariance matrix (yielding a linear function); it also takes into account the prior probabilities of the groups. The calibration information can be stored in a special SAS data set and applied to other data sets.

When no assumptions can be made about the distribution within each group, or when the distribution is assumed not to be multivariate normal, nonparametric methods can be used to estimate the group-specific densities. These methods include the kernel and k-nearest-neighbor methods (Rosenblatt 1956; Parzen 1962). The DISCRIM procedure uses uniform, normal, Epanechnikov, biweight, or triweight kernels for density estimation.

Either Mahalanobis or Euclidean distance can be used to determine proximity. Mahalanobis distance can be based on either the full covariance matrix or the diagonal matrix of variances. With a k-nearest-neighbor method, the pooled covariance matrix is used to calculate the Mahalanobis distances. With a kernel method, either the individual within-group covariance matrices or the pooled covariance matrix can be used to calculate the Mahalanobis distances. With the estimated group-specific densities and their associated prior probabilities, the posterior probability estimates of group membership for each class can be evaluated.

Canonical discriminant analysis is a dimension-reduction technique related to principal component analysis and canonical correlation. Given a classification variable and several quantitative variables, PROC DISCRIM derives canonical variables (linear combinations of the quantitative variables) that summarize between-class variation in much the same way that principal components summarize total variation. (See Chapter 35, "The CANDISC Procedure," for more information about canonical discriminant analysis.) A discriminant criterion is always derived in PROC DISCRIM. If you want canonical discriminant analysis without the use of a discriminant criterion, you should use the CANDISC procedure.

The DISCRIM procedure can produce an output data set containing various statistics such as means, standard deviations, and correlations. If a parametric method is used, the discriminant function is also stored in the data set to classify future observations. When canonical discriminant analysis is performed, the output data set includes canonical coefficients that can be rotated by the FACTOR procedure. PROC DISCRIM can also create a second type of output data set containing the classification results for each observation. When canonical discriminant analysis is performed, this output data set also includes canonical variable scores. A third type of output data set containing the group-specific density estimates at each observation can also be produced.

PROC DISCRIM evaluates the performance of a discriminant criterion by estimating error rates (probabilities of misclassification) in the classification of future observations. These error-rate estimates include error-count estimates and posterior probability error-rate estimates. When the input data set is an ordinary SAS data set, the error rate can also be estimated by cross validation.

Do not confuse discriminant analysis with cluster analysis. All varieties of discriminant analysis require prior knowledge of the classes, usually in the form of a sample from each class. In cluster analysis, the data do not include information about class membership; the purpose is to construct a classification.

See Chapter 11, "Introduction to Discriminant Procedures," for a discussion of discriminant analysis.

Getting Started: DISCRIM Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/discgs.sas.)

The data in this example are measurements of 159 fish caught in Finland's Lake Laengelmaevesi; this data set is available from Puranen (1917). For each of the seven species (bream, roach, whitefish, parkki, perch, pike, and smelt) the weight, length, height, and width of each fish are tallied. Three different length measurements are recorded: from the nose of the fish to the beginning of its tail, from the nose to the notch of its tail, and from the nose to the end of its tail. The height and width are recorded as percentages of the third length variable. The fish data set is available from the Sashelp library. The goal now is to find a discriminant function based on these six variables that best classifies the fish into species.

First, assume that the data are normally distributed within each group with equal covariances across groups. The following statements use PROC DISCRIM to analyze the Sashelp.Fish data and create Figure 42.1 through Figure 42.5:

```
title 'Fish Measurement Data';

proc discrim data=sashelp.fish;
   class Species;
run;
```

The DISCRIM procedure begins by displaying summary information about the variables in the analysis (see Figure 42.1). This information includes the number of observations, the number of quantitative variables in the analysis (specified with the VAR statement), and the number of classes in the classification variable (specified with the CLASS statement). The frequency of each class, its weight, the proportion of the total sample, and the prior probability are also displayed. Equal priors are assigned by default.

Figure 42.1 Summary Information

Fish Measurement Data

The DISCRIM Procedure

Total Sample Size	158	**DF Total**	157
Variables	6	**DF Within Classes**	151
Classes	7	**DF Between Classes**	6

Number of Observations Read	159
Number of Observations Used	158

Figure 42.1 *continued*

Class Level Information					
Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Bream	Bream	34	34.0000	0.215190	0.142857
Parkki	Parkki	11	11.0000	0.069620	0.142857
Perch	Perch	56	56.0000	0.354430	0.142857
Pike	Pike	17	17.0000	0.107595	0.142857
Roach	Roach	20	20.0000	0.126582	0.142857
Smelt	Smelt	14	14.0000	0.088608	0.142857
Whitefish	Whitefish	6	6.0000	0.037975	0.142857

The natural log of the determinant of the pooled covariance matrix is displayed in Figure 42.2.

Figure 42.2 Pooled Covariance Matrix Information

Pooled Covariance Matrix Information	
Covariance Matrix Rank	Natural Log of the Determinant of the Covariance Matrix
6	4.17613

The squared distances between the classes are shown in Figure 42.3.

Figure 42.3 Squared Distances

Fish Measurement Data

The DISCRIM Procedure

Generalized Squared Distance to Species							
From Species	Bream	Parkki	Perch	Pike	Roach	Smelt	Whitefish
Bream	0	83.32523	243.66688	310.52333	133.06721	252.75503	132.05820
Parkki	83.32523	0	57.09760	174.20918	27.00096	60.52076	26.54855
Perch	243.66688	57.09760	0	101.06791	29.21632	29.26806	20.43791
Pike	310.52333	174.20918	101.06791	0	92.40876	127.82177	99.90673
Roach	133.06721	27.00096	29.21632	92.40876	0	33.84280	6.31997
Smelt	252.75503	60.52076	29.26806	127.82177	33.84280	0	46.37326
Whitefish	132.05820	26.54855	20.43791	99.90673	6.31997	46.37326	0

The coefficients of the linear discriminant function are displayed (in Figure 42.4) with the default options METHOD=NORMAL and POOL=YES.

Figure 42.4 Linear Discriminant Function

Linear Discriminant Function for Species							
Variable	**Bream**	**Parkki**	**Perch**	**Pike**	**Roach**	**Smelt**	**Whitefish**
Constant	-185.91682	-64.92517	-48.68009	-148.06402	-62.65963	-19.70401	-67.44603
Weight	-0.10912	-0.09031	-0.09418	-0.13805	-0.09901	-0.05778	-0.09948
Length1	-23.02273	-13.64180	-19.45368	-20.92442	-14.63635	-4.09257	-22.57117
Length2	-26.70692	-5.38195	17.33061	6.19887	-7.47195	-3.63996	3.83450
Length3	50.55780	20.89531	5.25993	22.94989	25.00702	10.60171	21.12638
Height	13.91638	8.44567	-1.42833	-8.99687	-0.26083	-1.84569	0.64957
Width	-23.71895	-13.38592	1.32749	-9.13410	-3.74542	-3.43630	-2.52442

A summary of how the discriminant function classifies the data used to develop the function is displayed last. In Figure 42.5, you see that only three of the observations are misclassified. The error-count estimates give the proportion of misclassified observations in each group. Since you are classifying the same data that are used to derive the discriminant function, these error-count estimates are biased.

Figure 42.5 Resubstitution Misclassification Summary

Fish Measurement Data

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.FISH
Resubstitution Summary using Linear Discriminant Function

Number of Observations and Percent Classified into Species								
From Species	**Bream**	**Parkki**	**Perch**	**Pike**	**Roach**	**Smelt**	**Whitefish**	**Total**
Bream	34	0	0	0	0	0	0	34
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	100.00
Parkki	0	11	0	0	0	0	0	11
	0.00	100.00	0.00	0.00	0.00	0.00	0.00	100.00
Perch	0	0	53	0	0	3	0	56
	0.00	0.00	94.64	0.00	0.00	5.36	0.00	100.00
Pike	0	0	0	17	0	0	0	17
	0.00	0.00	0.00	100.00	0.00	0.00	0.00	100.00
Roach	0	0	0	0	20	0	0	20
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	100.00
Smelt	0	0	0	0	0	14	0	14
	0.00	0.00	0.00	0.00	0.00	100.00	0.00	100.00
Whitefish	0	0	0	0	0	0	6	6
	0.00	0.00	0.00	0.00	0.00	0.00	100.00	100.00
Total	34	11	53	17	20	17	6	158
	21.52	6.96	33.54	10.76	12.66	10.76	3.80	100.00
Priors	0.14286	0.14286	0.14286	0.14286	0.14286	0.14286	0.14286	

Error Count Estimates for Species								
	Bream	**Parkki**	**Perch**	**Pike**	**Roach**	**Smelt**	**Whitefish**	**Total**
Rate	0.0000	0.0000	0.0536	0.0000	0.0000	0.0000	0.0000	0.0077
Priors	0.1429	0.1429	0.1429	0.1429	0.1429	0.1429	0.1429	

One way to reduce the bias of the error-count estimates is to split your data into two sets. One set is used to

derive the discriminant function, and the other set is used to run validation tests. Example 42.4 shows how to analyze a test data set. Another method of reducing bias is to classify each observation by using a discriminant function computed from all of the other observations; this method is invoked with the CROSSVALIDATE option.

Syntax: DISCRIM Procedure

The following statements are available in the DISCRIM procedure:

```
PROC DISCRIM <options> ;
    CLASS variable ;
    BY variables ;
    FREQ variable ;
    ID variable ;
    PRIORS probabilities ;
    TESTCLASS variable ;
    TESTFREQ variable ;
    TESTID variable ;
    VAR variables ;
    WEIGHT variable ;
```

Only the PROC DISCRIM and CLASS statements are required.

The following sections describe the PROC DISCRIM statement and then describe the other statements in alphabetical order.

PROC DISCRIM Statement

```
PROC DISCRIM <options> ;
```

The PROC DISCRIM statement invokes the DISCRIM procedure. Table 42.1 summarizes the options available in the PROC DISCRIM statement.

Table 42.1 Options Available in the PROC DISCRIM Statement

Option	Description
Input Data Sets	
DATA=	Specifies input SAS data set
TESTDATA=	Specifies input SAS data set to classify
Output Data Sets	
OUTSTAT=	Specifies output statistics data set
OUT=	Specifies output data set with classification results
OUTCROSS=	Specifies output data set with cross validation results
OUTD=	Specifies output data set with densities
SCORES=	Outputs discriminant scores to the OUT= data set
TESTOUT=	Specifies output data set with TEST= results

Table 42.1 *continued*

Option	Description
TESTOUTD=	Specifies output data set with TEST= densities
Method Details	
METHOD=	Specifies parametric or nonparametric method
POOL=	Specifies whether to pool the covariance matrices
SINGULAR=	Specifies the singularity criterion
SLPOOL=	Specifies significance level homogeneity test
THRESHOLD=	Specifies the minimum threshold for classification
Nonparametric Methods	
K=	Specifies k value for k nearest neighbors
KPROP=	Specifies proportion, p, for computing k
R=	Specifies radius for kernel density estimation
KERNEL=	Specifies a kernel density to estimate
METRIC=	Specifies metric in for squared distances
Canonical Discriminant Analysis	
CANONICAL	Performs canonical discriminant analysis
CANPREFIX=	Specifies a prefix for naming the canonical variables
NCAN=	Specifies the number of canonical variables
Resubstitution Classification	
LIST	Displays the classification results
LISTERR	Displays the misclassified observations
NOCLASSIFY	Suppresses the classification
TESTLIST	Displays the classification results of TEST=
TESTLISTERR	Displays the misclassified observations of TEST=
Cross Validation Classification	
CROSSLIST	Displays the cross validation results
CROSSLISTERR	Displays the misclassified cross validation results
CROSSVALIDATE	Specifies cross validation
Control Displayed Output	
ALL	Displays all output
ANOVA	Displays univariate statistics
BCORR	Displays between correlations
BCOV	Displays between covariances
BSSCP	Displays between SSCPs
DISTANCE	Displays squared Mahalanobis distances
FORMULA	Displays formulas in destinations other than LISTING
MANOVA	Displays multivariate ANOVA results
NOPRINT	Suppresses all displayed output
PCORR	Displays pooled correlations
PCOV	Displays pooled covariances
POSTERR	Displays posterior probability error-rate estimates
PSSCP	Displays pooled SSCPs

Table 42.1 *continued*

Option	Description
SHORT	Suppresses some displayed output
SIMPLE	Displays simple descriptive statistics
STDMEAN	Displays standardized class means
TCORR	Displays total correlations
TCOV	Displays total covariances
TSSCP	Displays total SSCPs
WCORR	Displays within correlations
WCOV	Displays within covariances
WSSCP	Displays within SSCPs

ALL

activates all options that control displayed output. When the derived classification criterion is used to classify observations, the ALL option also activates the POSTERR option.

ANOVA

displays univariate statistics for testing the hypothesis that the class means are equal in the population for each variable.

BCORR

displays between-class correlations.

BCOV

displays between-class covariances. The between-class covariance matrix equals the between-class SSCP matrix divided by $n(c-1)/c$, where n is the number of observations and c is the number of classes. You should interpret the between-class covariances in comparison with the total-sample and within-class covariances, not as formal estimates of population parameters.

BSSCP

displays the between-class SSCP matrix.

CANONICAL

CAN

performs canonical discriminant analysis.

CANPREFIX=*name*

specifies a prefix for naming the canonical variables. By default, the names are Can1, Can2, ..., Can*n*. If you specify CANPREFIX=ABC, the components are named ABC1, ABC2, ABC3, and so on. The number of characters in the prefix plus the number of digits required to designate the canonical variables should not exceed the length defined by the VALIDVARNAME= system option (for example, 32 for VALIDVARNAME=V7). The prefix is truncated if the combined length exceeds the maximum.

The CANONICAL option is activated when you specify either the NCAN= or the CANPREFIX= option. A discriminant criterion is always derived in PROC DISCRIM. If you want canonical discriminant analysis without the use of discriminant criteria, you should use PROC CANDISC.

CROSSLIST

displays the cross validation classification results for each observation.

CROSSLISTERR

displays the cross validation classification results for misclassified observations only.

CROSSVALIDATE

specifies the cross validation classification of the input DATA= data set. When a parametric method is used, PROC DISCRIM classifies each observation in the DATA= data set by using a discriminant function computed from the other observations in the DATA= data set, excluding the observation being classified. When a nonparametric method is used, the covariance matrices used to compute the distances are based on all observations in the data set and do not exclude the observation being classified. However, the observation being classified is excluded from the nonparametric density estimation (if you specify the R= option) or the k nearest neighbors (if you specify the K= or KPROP= option) of that observation. The CROSSVALIDATE option is set when you specify the CROSSLIST, CROSSLISTERR, or OUTCROSS= option. With these options, cross validation information is displayed or output in addition to the usual resubstitution classification results. Cross validation classification results are written to the OUTCROSS= data set, and resubstitution classification results are written to the OUT= data set.

DATA=*SAS-data-set*

specifies the data set to be analyzed. The data set can be an ordinary SAS data set or one of several specially structured data sets created by SAS/STAT procedures. These specially structured data sets include TYPE=CORR, TYPE=COV, TYPE=CSSCP, TYPE=SSCP, TYPE=LINEAR, TYPE=QUAD, and TYPE=MIXED. The input data set must be an ordinary SAS data set if you specify METHOD=NPAR. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

DISTANCE

MAHALANOBIS

displays the squared Mahalanobis distances between the group means, F statistics, and the corresponding probabilities of greater Mahalanobis squared distances between the group means. The squared distances are based on the specification of the POOL= and METRIC= options.

FORMULA

displays formulas in destinations other than LISTING. See the section "Formulas" on page 2758 for more information.

K=*k*

specifies a k value for the k-nearest-neighbor rule. An observation $\mathbf{x}$ is classified into a group based on the information from the k nearest neighbors of $\mathbf{x}$. Do not specify the K= option with the KPROP= or R= option.

KPROP=*p*

specifies a proportion, p, for computing the k value for the k-nearest-neighbor rule: $k = \max(1, \mathrm{floor}(np))$, where n is the number of valid observations. When there is a FREQ statement, n is the sum of the FREQ variable for the observations used in the analysis (those without missing or invalid values). An observation $\mathbf{x}$ is classified into a group based on the information from the k nearest neighbors of $\mathbf{x}$. Do not specify the KPROP= option with the K= or R= option.

KERNEL=BIWEIGHT | BIW

KERNEL=EPANECHNIKOV | EPA

KERNEL=NORMAL | NOR

KERNEL=TRIWEIGHT | TRI

KERNEL=UNIFORM | UNI

specifies a kernel density to estimate the group-specific densities. You can specify the KERNEL= option only when the R= option is specified. The default is KERNEL=UNIFORM.

LIST

displays the resubstitution classification results for each observation. You can specify this option only when the input data set is an ordinary SAS data set.

LISTERR

displays the resubstitution classification results for misclassified observations only. You can specify this option only when the input data set is an ordinary SAS data set.

MANOVA

displays multivariate statistics for testing the hypothesis that the class means are equal in the population.

METHOD=NORMAL | NPAR

determines the method to use in deriving the classification criterion. When you specify METHOD=NORMAL, a parametric method based on a multivariate normal distribution within each class is used to derive a linear or quadratic discriminant function. The default is METHOD=NORMAL. When you specify METHOD=NPAR, a nonparametric method is used and you must also specify either the K= or R= option.

METRIC=DIAGONAL | FULL | IDENTITY

specifies the metric in which the computations of squared distances are performed. If you specify METRIC=FULL, then PROC DISCRIM uses either the pooled covariance matrix (POOL=YES) or individual within-group covariance matrices (POOL=NO) to compute the squared distances. If you specify METRIC=DIAGONAL, then PROC DISCRIM uses either the diagonal matrix of the pooled covariance matrix (POOL=YES) or diagonal matrices of individual within-group covariance matrices (POOL=NO) to compute the squared distances. If you specify METRIC=IDENTITY, then PROC DISCRIM uses Euclidean distance. The default is METRIC=FULL. When you specify METHOD=NORMAL, the option METRIC=FULL is used.

NCAN=*number*

specifies the number of canonical variables to compute. The value of *number* must be less than or equal to the number of variables. If you specify the option NCAN=0, the procedure displays the canonical correlations but not the canonical coefficients, structures, or means. Let v be the number of variables in the VAR statement, and let c be the number of classes. If you omit the NCAN= option, only $\min(v, c-1)$ canonical variables are generated. If you request an output data set (OUT=, OUTCROSS=, TESTOUT=), v canonical variables are generated. In this case, the last $v-(c-1)$ canonical variables have missing values.

The CANONICAL option is activated when you specify either the NCAN= or the CANPREFIX= option. A discriminant criterion is always derived in PROC DISCRIM. If you want canonical discriminant analysis without the use of discriminant criterion, you should use PROC CANDISC.

NOCLASSIFY

suppresses the resubstitution classification of the input DATA= data set. You can specify this option only when the input data set is an ordinary SAS data set.

NOPRINT

suppresses the normal display of results. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

OUT=*SAS-data-set*

creates an output SAS data set containing all the data from the DATA= data set, plus the posterior probabilities and the class into which each observation is classified by resubstitution. When you specify the CANONICAL option, the data set also contains new variables with canonical variable scores. See the section "OUT= Data Set" on page 2750 for more information.

OUTCROSS=*SAS-data-set*

creates an output SAS data set containing all the data from the DATA= data set, plus the posterior probabilities and the class into which each observation is classified by cross validation. When you specify the CANONICAL option, the data set also contains new variables with canonical variable scores. See the section "OUT= Data Set" on page 2750 for more information.

OUTD=*SAS-data-set*

creates an output SAS data set containing all the data from the DATA= data set, plus the group-specific density estimates for each observation. See the section "OUT= Data Set" on page 2750 for more information.

OUTSTAT=*SAS-data-set*

creates an output SAS data set containing various statistics such as means, standard deviations, and correlations. When the input data set is an ordinary SAS data set or when TYPE=CORR, TYPE=COV, TYPE=CSSCP, or TYPE=SSCP, this option can be used to generate discriminant statistics. When you specify the CANONICAL option, canonical correlations, canonical structures, canonical coefficients, and means of canonical variables for each class are included in the data set. If you specify METHOD=NORMAL, the output data set also includes coefficients of the discriminant functions, and the output data set is TYPE=LINEAR (POOL=YES), TYPE=QUAD (POOL=NO), or TYPE=MIXED (POOL=TEST). If you specify METHOD=NPAR, this output data set is TYPE=CORR. This data set also holds calibration information that can be used to classify new observations. See the sections "Saving and Using Calibration Information" on page 2748 and "OUT= Data Set" on page 2750 for more information.

PCORR

displays pooled within-class correlations.

PCOV

displays pooled within-class covariances.

POOL=NO | TEST | YES

determines whether the pooled or within-group covariance matrix is the basis of the measure of the squared distance. If you specify POOL=YES, then PROC DISCRIM uses the pooled covariance matrix in calculating the (generalized) squared distances. Linear discriminant functions are computed. If you specify POOL=NO, the procedure uses the individual within-group covariance matrices in calculating the distances. Quadratic discriminant functions are computed. The default is POOL=YES.

The k-nearest-neighbor method assumes the default of POOL=YES, and the POOL=TEST option cannot be used with the METHOD=NPAR option.

When you specify METHOD=NORMAL, the option POOL=TEST requests Bartlett's modification of the likelihood ratio test (Morrison 1976; Anderson 1984) of the homogeneity of the within-group covariance matrices. The test is unbiased (Perlman 1980). However, it is not robust to nonnormality. If the test statistic is significant at the level specified by the SLPOOL= option, the within-group covariance matrices are used. Otherwise, the pooled covariance matrix is used. The discriminant function coefficients are displayed only when the pooled covariance matrix is used.

POSTERR

displays the posterior probability error-rate estimates of the classification criterion based on the classification results.

PSSCP

displays the pooled within-class corrected SSCP matrix.

R=*r*

specifies a radius r value for kernel density estimation. With uniform, Epanechnikov, biweight, or triweight kernels, an observation $\mathbf{x}$ is classified into a group based on the information from observations $\mathbf{y}$ in the training set within the radius r of $\mathbf{x}$—that is, the group t observations $\mathbf{y}$ with squared distance $d_t^2(\mathbf{x}, \mathbf{y}) \leq r^2$. When a normal kernel is used, the classification of an observation $\mathbf{x}$ is based on the information of the estimated group-specific densities from all observations in the training set. The matrix $r^2\mathbf{V}_t$ is used as the group t covariance matrix in the normal-kernel density, where $\mathbf{V}_t$ is the matrix used in calculating the squared distances. Do not specify the K= or KPROP= option with the R= option. For more information about selecting r, see the section "Nonparametric Methods" on page 2740.

SCORES< = *prefix* >

computes and outputs discriminant scores to the OUT= and TESTOUT= data sets with the default options METHOD=NORMAL and POOL=YES (or with METHOD=NORMAL, POOL=TEST, and a nonsignificant chi-square test). Otherwise, or if no OUT= or TESTOUT= data set is specified, this option is ignored. The scores are computed by a matrix multiplication of an intercept term and the raw data or test data by the coefficients in the linear discriminant function. One score variable is created for each level of the CLASS variable. By default, the variables are named "Sc_" followed by the formatted class level. You can specify SCORES=prefix to use a prefix other than "Sc_". The specifications SCORES and SCORES=Sc_ are equivalent.

SHORT

suppresses the display of certain items in the default output. If you specify METHOD=NORMAL, then PROC DISCRIM suppresses the display of determinants, generalized squared distances between-class means, and discriminant function coefficients. When you specify the CANONICAL option, PROC DISCRIM suppresses the display of canonical structures, canonical coefficients, and class means on canonical variables; only tables of canonical correlations are displayed.

SIMPLE

displays simple descriptive statistics for the total sample and within each class.

SINGULAR=*p*

specifies the criterion for determining the singularity of a matrix, where $0 < p < 1$. The default is SINGULAR=1E–8.

Let $\mathbf{S}$ be the total-sample correlation matrix. If the R square for predicting a quantitative variable in the VAR statement from the variables preceding it exceeds $1 - p$, then $\mathbf{S}$ is considered singular. If $\mathbf{S}$ is singular, the probability levels for the multivariate test statistics and canonical correlations are adjusted for the number of variables with R square exceeding $1 - p$.

Let $\mathbf{S}_t$ be the group t covariance matrix, and let $\mathbf{S}_p$ be the pooled covariance matrix. In group t, if the R square for predicting a quantitative variable in the VAR statement from the variables preceding it exceeds $1 - p$, then $\mathbf{S}_t$ is considered singular. Similarly, if the partial R square for predicting a quantitative variable in the VAR statement from the variables preceding it, after controlling for the effect of the CLASS variable, exceeds $1 - p$, then $\mathbf{S}_p$ is considered singular.

If PROC DISCRIM needs to compute either the inverse or the determinant of a matrix that is considered singular, then it uses a quasi inverse or a quasi determinant. For details, see the section "Quasi-inverse" on page 2745.

SLPOOL=*p*

specifies the significance level for the test of homogeneity. You can specify the SLPOOL= option only when POOL=TEST is also specified. If you specify POOL= TEST but omit the SLPOOL= option, PROC DISCRIM uses 0.10 as the significance level for the test.

STDMEAN

displays total-sample and pooled within-class standardized class means.

TCORR

displays total-sample correlations.

TCOV

displays total-sample covariances.

TESTDATA=*SAS-data-set*

names an ordinary SAS data set with observations that are to be classified. The quantitative variable names in this data set must match those in the DATA= data set. When you specify the TESTDATA= option, you can also specify the TESTCLASS, TESTFREQ, and TESTID statements. When you specify the TESTDATA= option, you can use the TESTOUT= and TESTOUTD= options to generate classification results and group-specific density estimates for observations in the test data set. Note that if the CLASS variable is not present in the TESTDATA= data set, the output will not include misclassification statistics.

TESTLIST

lists classification results for all observations in the TESTDATA= data set.

TESTLISTERR
lists only misclassified observations in the TESTDATA= data set but only if a TESTCLASS statement is also used.

TESTOUT=*SAS-data-set*
creates an output SAS data set containing all the data from the TESTDATA= data set, plus the posterior probabilities and the class into which each observation is classified. When you specify the CANONICAL option, the data set also contains new variables with canonical variable scores. See the section "OUT= Data Set" on page 2750 for more information.

TESTOUTD=*SAS-data-set*
creates an output SAS data set containing all the data from the TESTDATA= data set, plus the group-specific density estimates for each observation. See the section "OUT= Data Set" on page 2750 for more information.

THRESHOLD=*p*
specifies the minimum acceptable posterior probability for classification, where $0 \leq p \leq 1$. If the largest posterior probability of group membership is less than the THRESHOLD value, the observation is labeled as `'Other'`. The default is THRESHOLD=0. In some cases, you might want to specify a THRESHOLD= value slightly smaller than the desired *p* so that observations with posterior probabilities within rounding error of *p* are classified. For example, you can specify **`threshold=%sysevalf(0.5 - 1e-8)`** instead of THRESHOLD=0.5 so that observations with posterior probabilities within 1E–8 of 0.5 and larger are classified.

TSSCP
displays the total-sample corrected SSCP matrix.

WCORR
displays within-class correlations for each class level.

WCOV
displays within-class covariances for each class level.

WSSCP
displays the within-class corrected SSCP matrix for each class level.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC DISCRIM to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the DISCRIM procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

If you specify the TESTDATA= option and the TESTDATA= data set does not contain any of the BY variables, then the entire TESTDATA= data set is classified according to the discriminant functions computed in each BY group in the DATA= data set.

If the TESTDATA= data set contains some but not all of the BY variables, or if some BY variables do not have the same type or length in the TESTDATA= data set as in the DATA= data set, then PROC DISCRIM displays an error message and stops.

If all BY variables appear in the TESTDATA= data set with the same type and length as in the DATA= data set, then each BY group in the TESTDATA= data set is classified by the discriminant function from the corresponding BY group in the DATA= data set. The BY groups in the TESTDATA= data set must be in the same order as in the DATA= data set. If you specify the NOTSORTED option in the BY statement, there must be exactly the same BY groups in the same order in both data sets. If you omit the NOTSORTED option, some BY groups can appear in one data set but not in the other. If some BY groups appear in the TESTDATA= data set but not in the DATA= data set, and you request an output test data set by using the TESTOUT= or TESTOUTD= option, these BY groups are not included in the output data set.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* **;**

The values of the classification variable define the groups for analysis. Class levels are determined by the formatted values of the CLASS variable. The specified variable can be numeric or character. A CLASS statement is required.

FREQ Statement

FREQ *variable* **;**

If a variable in the data set represents the frequency of occurrence for the other values in the observation, include the variable's name in a FREQ statement. The procedure then treats the data set as if each observation appears n times, where n is the value of the FREQ variable for the observation. The total number of observations is considered to be equal to the sum of the FREQ variable when the procedure determines degrees of freedom for significance probabilities.

If the value of the FREQ variable is missing or is less than one, the observation is not used in the analysis. If the value is not an integer, it is truncated to an integer.

ID Statement

ID *variable* **;**

The ID statement is effective only when you specify the LIST or LISTERR option in the PROC DISCRIM statement. When the DISCRIM procedure displays the classification results, the ID variable (rather than the observation number) is displayed for each observation.

PRIORS Statement

PRIORS EQUAL ;

PRIORS PROPORTIONAL | PROP ;

PRIORS *probabilities* **;**

The PRIORS statement specifies the prior probabilities of group membership. To set the prior probabilities equal, use the following statement:

```
priors equal;
```

To set the prior probabilities proportional to the sample sizes, use the following statement:

```
priors proportional;
```

For other than equal or proportional priors, specify the prior probability for each level of the classification variable. Each class level can be written as either a SAS name or a quoted string, and it must be followed by an equal sign and a numeric constant between zero and one. A SAS name begins with a letter or an underscore and can contain digits as well. Lowercase character values and data values with leading blanks must be enclosed in quotes. For example, to define prior probabilities for each level of Grade, where Grade's values are `'A'`, `'B'`, `'C'`, and `'D'`, the PRIORS statement can be specified as follows:

```
priors A=0.1 B=0.3 C=0.5 D=0.1;
```

If Grade's values are `'a'`, `'b'`, `'c'`, and `'d'`, each class level must be written as a quoted string as follows:

```
priors 'a'=0.1  'b'=0.3  'c'=0.5  'd'=0.1;
```

If Grade is numeric, with formatted values of `'1'`, `'2'`, and `'3'`, the PRIORS statement can be written as follows:

```
priors '1'=0.3  '2'=0.6  '3'=0.1;
```

The specified class levels must exactly match the formatted values of the CLASS variable. For example, if a CLASS variable C has the format 4.2 and a value 5, the PRIORS statement must specify `'5.00'`, not `'5.0'` or `'5'`. If the prior probabilities do not sum to one, these probabilities are scaled proportionally to have the sum equal to one. The default is PRIORS EQUAL.

TESTCLASS Statement

TESTCLASS *variable* **;**

The TESTCLASS statement names the variable in the TESTDATA= data set that is used to determine whether an observation in the TESTDATA= data set is misclassified. The TESTCLASS variable should have the same type (character or numeric) and length as the variable given in the CLASS statement. PROC DISCRIM considers an observation misclassified when the formatted value of the TESTCLASS variable does not match the group into which the TESTDATA= observation is classified. When the TESTCLASS statement is missing and the TESTDATA= data set contains the variable given in the CLASS statement, the CLASS variable is used as the TESTCLASS variable. Note that if the CLASS variable is not present in the TESTDATA= data set, the output will not include misclassification statistics.

TESTFREQ Statement

TESTFREQ *variable* **;**

If a variable in the TESTDATA= data set represents the frequency of occurrence of the other values in the observation, include the variable's name in a TESTFREQ statement. The procedure then treats the data set as if each observation appears *n* times, where *n* is the value of the TESTFREQ variable for the observation.

If the value of the TESTFREQ variable is missing or is less than one, the observation is not used in the analysis. If the value is not an integer, it is truncated to an integer.

TESTID Statement

TESTID *variable* **;**

The TESTID statement is effective only when you specify the TESTLIST or TESTLISTERR option in the PROC DISCRIM statement. When the DISCRIM procedure displays the classification results for the TESTDATA= data set, the TESTID variable (rather than the observation number) is displayed for each observation. The variable given in the TESTID statement must be in the TESTDATA= data set.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the quantitative variables to be included in the analysis. The default is all numeric variables not listed in other statements.

WEIGHT Statement

WEIGHT *variable* **;**

To use relative weights for each observation in the input data set, place the weights in a variable in the data set and specify the name in a WEIGHT statement. This is often done when the variance associated with each observation is different and the values of the weight variable are proportional to the reciprocals of the variances. If the value of the WEIGHT variable is missing or is less than zero, then a value of zero for the weight is used.

The WEIGHT and FREQ statements have a similar effect except that the WEIGHT statement does not alter the degrees of freedom.

Details: DISCRIM Procedure

Missing Values

Observations with missing values for variables in the analysis are excluded from the development of the classification criterion. When the values of the classification variable are missing, the observation is excluded from the development of the classification criterion, but if no other variables in the analysis have missing values for that observation, the observation is classified and displayed with the classification results.

Background

The following notation is used to describe the classification methods:

$\mathbf{x}$	a p-dimensional vector containing the quantitative variables of an observation
$\mathbf{S}_p$	the pooled covariance matrix
t	a subscript to distinguish the groups
n_t	the number of training set observations in group t
$\mathbf{m}_t$	the p-dimensional vector containing variable means in group t
$\mathbf{S}_t$	the covariance matrix within group t
$\lvert\mathbf{S}_t\rvert$	the determinant of $\mathbf{S}_t$
q_t	the prior probability of membership in group t
$p(t\lvert\mathbf{x})$	the posterior probability of an observation $\mathbf{x}$ belonging to group t
f_t	the probability density function for group t
$f_t(\mathbf{x})$	the group-specific density estimate at $\mathbf{x}$ from group t
$f(\mathbf{x})$	$\sum_t q_t f_t(\mathbf{x})$, the estimated unconditional density at $\mathbf{x}$
e_t	the classification error rate for group t

Bayes' Theorem

Assuming that the prior probabilities of group membership are known and that the group-specific densities at $\mathbf{x}$ can be estimated, PROC DISCRIM computes $p(t|\mathbf{x})$, the probability of $\mathbf{x}$ belonging to group t, by applying Bayes' theorem:

$$p(t|\mathbf{x}) = \frac{q_t f_t(\mathbf{x})}{f(\mathbf{x})}$$

PROC DISCRIM partitions a p-dimensional vector space into regions R_t, where the region R_t is the subspace containing all p-dimensional vectors $\mathbf{y}$ such that $p(t|\mathbf{y})$ is the largest among all groups. An observation is classified as coming from group t if it lies in region R_t.

Parametric Methods

Assuming that each group has a multivariate normal distribution, PROC DISCRIM develops a discriminant function or classification criterion by using a measure of generalized squared distance. The classification criterion is based on either the individual within-group covariance matrices or the pooled covariance matrix; it also takes into account the prior probabilities of the classes. Each observation is placed in the class from which it has the smallest generalized squared distance. PROC DISCRIM also computes the posterior probability of an observation belonging to each class.

The squared Mahalanobis distance from $\mathbf{x}$ to group t is

$$d_t^2(\mathbf{x}) = (\mathbf{x} - \mathbf{m}_t)'\mathbf{V}_t^{-1}(\mathbf{x} - \mathbf{m}_t)$$

where $\mathbf{V}_t = \mathbf{S}_t$ if the within-group covariance matrices are used, or $\mathbf{V}_t = \mathbf{S}_p$ if the pooled covariance matrix is used.

The group-specific density estimate at $\mathbf{x}$ from group t is then given by

$$f_t(\mathbf{x}) = (2\pi)^{-\frac{p}{2}}|\mathbf{V}_t|^{-\frac{1}{2}} \exp\left(-0.5 d_t^2(\mathbf{x})\right)$$

Using Bayes' theorem, the posterior probability of $\mathbf{x}$ belonging to group t is

$$p(t|\mathbf{x}) = \frac{q_t f_t(\mathbf{x})}{\sum_u q_u f_u(\mathbf{x})}$$

where the summation is over all groups.

The generalized squared distance from $\mathbf{x}$ to group t is defined as

$$D_t^2(\mathbf{x}) = d_t^2(\mathbf{x}) + g_1(t) + g_2(t)$$

where

$$g_1(t) = \begin{cases} \ln|\mathbf{S}_t| & \text{if the within-group covariance matrices are used} \\ 0 & \text{if the pooled covariance matrix is used} \end{cases}$$

and

$$g_2(t) = \begin{cases} -2\ln(q_t) & \text{if the prior probabilities are not all equal} \\ 0 & \text{if the prior probabilities are all equal} \end{cases}$$

The posterior probability of $\mathbf{x}$ belonging to group t is then equal to

$$p(t|\mathbf{x}) = \frac{\exp\left(-0.5D_t^2(\mathbf{x})\right)}{\sum_u \exp\left(-0.5D_u^2(\mathbf{x})\right)}$$

The discriminant scores are $-0.5D_u^2(\mathbf{x})$. An observation is classified into group u if setting $t = u$ produces the largest value of $p(t|\mathbf{x})$ or the smallest value of $D_t^2(\mathbf{x})$. If this largest posterior probability is less than the threshold specified, $\mathbf{x}$ is labeled as `'Other'`.

Nonparametric Methods

Nonparametric discriminant methods are based on nonparametric estimates of group-specific probability densities. Either a kernel method or the k-nearest-neighbor method can be used to generate a nonparametric density estimate in each group and to produce a classification criterion. The kernel method uses uniform, normal, Epanechnikov, biweight, or triweight kernels in the density estimation.

Either Mahalanobis distance or Euclidean distance can be used to determine proximity. When the k-nearest-neighbor method is used, the Mahalanobis distances are based on the pooled covariance matrix. When a kernel method is used, the Mahalanobis distances are based on either the individual within-group covariance matrices or the pooled covariance matrix. Either the full covariance matrix or the diagonal matrix of variances can be used to calculate the Mahalanobis distances.

The squared distance between two observation vectors, $\mathbf{x}$ and $\mathbf{y}$, in group t is given by

$$d_t^2(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})'\mathbf{V}_t^{-1}(\mathbf{x} - \mathbf{y})$$

where $\mathbf{V}_t$ has one of the following forms:

$$\mathbf{V}_t = \begin{cases} \mathbf{S}_p & \text{the pooled covariance matrix} \\ \text{diag}(\mathbf{S}_p) & \text{the diagonal matrix of the pooled covariance matrix} \\ \mathbf{S}_t & \text{the covariance matrix within group } t \\ \text{diag}(\mathbf{S}_t) & \text{the diagonal matrix of the covariance matrix within group } t \\ \mathbf{I} & \text{the identity matrix} \end{cases}$$

The classification of an observation vector $\mathbf{x}$ is based on the estimated group-specific densities from the training set. From these estimated densities, the posterior probabilities of group membership at $\mathbf{x}$ are evaluated. An observation $\mathbf{x}$ is classified into group u if setting $t = u$ produces the largest value of $p(t|\mathbf{x})$. If there is a tie for the largest probability or if this largest probability is less than the threshold specified, $\mathbf{x}$ is labeled as `'Other'`.

The kernel method uses a fixed radius, r, and a specified kernel, K_t, to estimate the group t density at each observation vector $\mathbf{x}$. Let $\mathbf{z}$ be a p-dimensional vector. Then the volume of a p-dimensional unit sphere bounded by $\mathbf{z}'\mathbf{z} = 1$ is

$$v_0 = \frac{\pi^{\frac{p}{2}}}{\Gamma\left(\frac{p}{2} + 1\right)}$$

where Γ represents the gamma function (see *SAS Functions and CALL Routines: Reference*).

Thus, in group t, the volume of a p-dimensional ellipsoid bounded by $\{\mathbf{z} \mid \mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z} = r^2\}$ is

$$v_r(t) = r^p|\mathbf{V}_t|^{\frac{1}{2}}v_0$$

The kernel method uses one of the following densities as the kernel density in group t:

Uniform Kernel

$$K_t(\mathbf{z}) = \begin{cases} \dfrac{1}{v_r(t)} & \text{if } \mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z} \le r^2 \\ 0 & \text{elsewhere} \end{cases}$$

Normal Kernel (with mean zero, variance $r^2\mathbf{V}_t$)

$$K_t(\mathbf{z}) = \frac{1}{c_0(t)} \exp\left(-\frac{1}{2r^2}\mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z}\right)$$

where $c_0(t) = (2\pi)^{\frac{p}{2}} r^p |\mathbf{V}_t|^{\frac{1}{2}}$.

Epanechnikov Kernel

$$K_t(\mathbf{z}) = \begin{cases} c_1(t)\left(1 - \dfrac{1}{r^2}\mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z}\right) & \text{if } \mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z} \le r^2 \\ 0 & \text{elsewhere} \end{cases}$$

where $c_1(t) = \dfrac{1}{v_r(t)}\left(1 + \dfrac{p}{2}\right)$.

Biweight Kernel

$$K_t(\mathbf{z}) = \begin{cases} c_2(t)\left(1 - \dfrac{1}{r^2}\mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z}\right)^2 & \text{if } \mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z} \le r^2 \\ 0 & \text{elsewhere} \end{cases}$$

where $c_2(t) = \left(1 + \dfrac{p}{4}\right) c_1(t)$.

Triweight Kernel

$$K_t(\mathbf{z}) = \begin{cases} c_3(t)\left(1 - \dfrac{1}{r^2}\mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z}\right)^3 & \text{if } \mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z} \le r^2 \\ 0 & \text{elsewhere} \end{cases}$$

where $c_3(t) = \left(1 + \dfrac{p}{6}\right) c_2(t)$.

The group t density at $\mathbf{x}$ is estimated by

$$f_t(\mathbf{x}) = \frac{1}{n_t} \sum_{\mathbf{y}} K_t(\mathbf{x} - \mathbf{y})$$

where the summation is over all observations $\mathbf{y}$ in group t, and K_t is the specified kernel function. The posterior probability of membership in group t is then given by

$$p(t|\mathbf{x}) = \frac{q_t f_t(\mathbf{x})}{f(\mathbf{x})}$$

where $f(\mathbf{x}) = \sum_u q_u f_u(\mathbf{x})$ is the estimated unconditional density. If $f(\mathbf{x})$ is zero, the observation $\mathbf{x}$ is labeled as `'Other'`.

The uniform-kernel method treats $K_t(\mathbf{z})$ as a multivariate uniform function with density uniformly distributed over $\mathbf{z}'\mathbf{V}_t^{-1}\mathbf{z} \le r^2$. Let k_t be the number of training set observations $\mathbf{y}$ from group t within the closed ellipsoid centered at $\mathbf{x}$ specified by $d_t^2(\mathbf{x}, \mathbf{y}) \le r^2$. Then the group t density at $\mathbf{x}$ is estimated by

$$f_t(\mathbf{x}) = \frac{k_t}{n_t v_r(t)}$$

When the identity matrix or the pooled within-group covariance matrix is used in calculating the squared distance, $v_r(t)$ is a constant, independent of group membership. The posterior probability of $\mathbf{x}$ belonging to group t is then given by

$$p(t|\mathbf{x}) = \frac{\frac{q_t k_t}{n_t}}{\sum_u \frac{q_u k_u}{n_u}}$$

If the closed ellipsoid centered at $\mathbf{x}$ does not include any training set observations, $f(\mathbf{x})$ is zero and $\mathbf{x}$ is labeled as `'Other'`. When the prior probabilities are equal, $p(t|\mathbf{x})$ is proportional to k_t/n_t and $\mathbf{x}$ is classified into the group that has the highest proportion of observations in the closed ellipsoid. When the prior probabilities are proportional to the group sizes, $p(t|\mathbf{x}) = k_t / \sum_u k_u$, $\mathbf{x}$ is classified into the group that has the largest number of observations in the closed ellipsoid.

The nearest-neighbor method fixes the number, k, of training set points for each observation $\mathbf{x}$. The method finds the radius $r_k(\mathbf{x})$ that is the distance from $\mathbf{x}$ to the kth-nearest training set point in the metric $\mathbf{V}_t^{-1}$. Consider a closed ellipsoid centered at $\mathbf{x}$ bounded by $\{\mathbf{z} \mid (\mathbf{z}-\mathbf{x})'\mathbf{V}_t^{-1}(\mathbf{z}-\mathbf{x}) = r_k^2(\mathbf{x})\}$; the nearest-neighbor method is equivalent to the uniform-kernel method with a location-dependent radius $r_k(\mathbf{x})$. Note that, with ties, more than k training set points might be in the ellipsoid.

Using the k-nearest-neighbor rule, the k_n (or more with ties) smallest distances are saved. Of these k distances, let k_t represent the number of distances that are associated with group t. Then, as in the uniform-kernel method, the estimated group t density at $\mathbf{x}$ is

$$f_t(\mathbf{x}) = \frac{k_t}{n_t v_k(\mathbf{x})}$$

where $v_k(\mathbf{x})$ is the volume of the ellipsoid bounded by $\{\mathbf{z} \mid (\mathbf{z}-\mathbf{x})'\mathbf{V}_t^{-1}(\mathbf{z}-\mathbf{x}) = r_k^2(\mathbf{x})\}$. Since the pooled within-group covariance matrix is used to calculate the distances used in the nearest-neighbor method, the volume $v_k(\mathbf{x})$ is a constant independent of group membership. When $k = 1$ is used in the nearest-neighbor rule, $\mathbf{x}$ is classified into the group associated with the $\mathbf{y}$ point that yields the smallest squared distance $d_t^2(\mathbf{x}, \mathbf{y})$. Prior probabilities affect nearest-neighbor results in the same way that they affect uniform-kernel results.

With a specified squared distance formula (METRIC=, POOL=), the values of r and k determine the degree of irregularity in the estimate of the density function, and they are called smoothing parameters. Small values

of r or k produce jagged density estimates, and large values of r or k produce smoother density estimates. Various methods for choosing the smoothing parameters have been suggested, and there is as yet no simple solution to this problem.

For a fixed kernel shape, one way to choose the smoothing parameter r is to plot estimated densities with different values of r and to choose the estimate that is most in accordance with the prior information about the density. For many applications, this approach is satisfactory.

Another way of selecting the smoothing parameter r is to choose a value that optimizes a given criterion. Different groups might have different sets of optimal values. Assume that the unknown density has bounded and continuous second derivatives and that the kernel is a symmetric probability density function. One criterion is to minimize an approximate mean integrated square error of the estimated density (Rosenblatt 1956). The resulting optimal value of r depends on the density function and the kernel. A reasonable choice for the smoothing parameter r is to optimize the criterion with the assumption that group t has a normal distribution with covariance matrix $\mathbf{V}_t$. Then, in group t, the resulting optimal value for r is given by

$$\left(\frac{A(K_t)}{n_t}\right)^{1/(p+4)}$$

where the optimal constant $A(K_t)$ depends on the kernel K_t (Epanechnikov 1969). For some useful kernels, the constants $A(K_t)$ are given by the following:

$$A(K_t) = \frac{1}{p}2^{p+1}(p+2)\Gamma\left(\frac{p}{2}\right) \qquad \text{with a uniform kernel}$$

$$A(K_t) = \frac{4}{2p+1} \qquad \text{with a normal kernel}$$

$$A(K_t) = \frac{2^{p+2}p^2(p+2)(p+4)}{2p+1}\Gamma\left(\frac{p}{2}\right) \qquad \text{with an Epanechnikov kernel}$$

These selections of $A(K_t)$ are derived under the assumption that the data in each group are from a multivariate normal distribution with covariance matrix $\mathbf{V}_t$. However, when the Euclidean distances are used in calculating the squared distance ($\mathbf{V}_t = I$), the smoothing constant should be multiplied by s, where s is an estimate of standard deviations for all variables. A reasonable choice for s is

$$s = \left(\frac{1}{p}\sum s_{jj}\right)^{\frac{1}{2}}$$

where s_{jj} are group t marginal variances.

The DISCRIM procedure uses only a single smoothing parameter for all groups. However, the selection of the matrix in the distance formula (from the METRIC= or POOL= option), enables individual groups and variables to have different scalings. When $\mathbf{V}_t$, the matrix used in calculating the squared distances, is an identity matrix, the kernel estimate at each data point is scaled equally for all variables in all groups. When $\mathbf{V}_t$ is the diagonal matrix of a covariance matrix, each variable in group t is scaled separately by its variance in the kernel estimation, where the variance can be the pooled variance ($\mathbf{V}_t = \mathbf{S}_p$) or an individual within-group variance ($\mathbf{V}_t = \mathbf{S}_t$). When $\mathbf{V}_t$ is a full covariance matrix, the variables in group t are scaled simultaneously by $\mathbf{V}_t$ in the kernel estimation.

In nearest-neighbor methods, the choice of k is usually relatively uncritical (Hand 1982). A practical approach is to try several different values of the smoothing parameters within the context of the particular application and to choose the one that gives the best cross validated estimate of the error rate.

Classification Error-Rate Estimates

A classification criterion can be evaluated by its performance in the classification of future observations. PROC DISCRIM uses two types of error-rate estimates to evaluate the derived classification criterion based on parameters estimated by the training sample:

- error-count estimates
- posterior probability error-rate estimates

The error-count estimate is calculated by applying the classification criterion derived from the training sample to a test set and then counting the number of misclassified observations. The group-specific error-count estimate is the proportion of misclassified observations in the group. When the test set is independent of the training sample, the estimate is unbiased. However, the estimate can have a large variance, especially if the test set is small.

When the input data set is an ordinary SAS data set and no independent test sets are available, the same data set can be used both to define and to evaluate the classification criterion. The resulting error-count estimate has an optimistic bias and is called an *apparent error rate*. To reduce the bias, you can split the data into two sets—one set for deriving the discriminant function and the other set for estimating the error rate. Such a split-sample method has the unfortunate effect of reducing the effective sample size.

Another way to reduce bias is cross validation (Lachenbruch and Mickey 1968). Cross validation treats $n - 1$ out of n training observations as a training set. It determines the discriminant functions based on these $n - 1$ observations and then applies them to classify the one observation left out. This is done for each of the n training observations. The misclassification rate for each group is the proportion of sample observations in that group that are misclassified. This method achieves a nearly unbiased estimate but with a relatively large variance.

To reduce the variance in an error-count estimate, smoothed error-rate estimates are suggested (Glick 1978). Instead of summing terms that are either zero or one as in the error-count estimator, the smoothed estimator uses a continuum of values between zero and one in the terms that are summed. The resulting estimator has a smaller variance than the error-count estimate. The posterior probability error-rate estimates provided by the POSTERR option in the PROC DISCRIM statement (see the section "Posterior Probability Error-Rate Estimates" on page 2746) are smoothed error-rate estimates. The posterior probability estimates for each group are based on the posterior probabilities of the observations classified into that same group. The posterior probability estimates provide good estimates of the error rate when the posterior probabilities are accurate. When a parametric classification criterion (linear or quadratic discriminant function) is derived from a nonnormal population, the resulting posterior probability error-rate estimators might not be appropriate.

The overall error rate is estimated through a weighted average of the individual group-specific error-rate estimates, where the prior probabilities are used as the weights.

To reduce both the bias and the variance of the estimator, Hora and Wilcox (1982) compute the posterior probability estimates based on cross validation. The resulting estimates are intended to have both low variance from using the posterior probability estimate and low bias from cross validation. They use Monte Carlo studies on two-group multivariate normal distributions to compare the cross validation posterior probability estimates with three other estimators: the apparent error rate, cross validation estimator, and posterior probability estimator. They conclude that the cross validation posterior probability estimator has a lower mean squared error in their simulations.

Quasi-inverse

Consider the plot shown in Figure 42.6 with two variables, X1 and X2, and two classes, A and B. The within-class covariance matrix is diagonal, with a positive value for X1 but zero for X2. Using a Moore-Penrose pseudo-inverse would effectively ignore X2 in doing the classification, and the two classes would have a zero generalized distance and could not be discriminated at all. The quasi inverse used by PROC DISCRIM replaces the zero variance for X2 with a small positive number to remove the singularity. This permits X2 to be used in the discrimination and results correctly in a large generalized distance between the two classes and a zero error rate. It also permits new observations, such as the one indicated by N, to be classified in a reasonable way. PROC CANDISC also uses a quasi inverse when the total-sample covariance matrix is considered to be singular and Mahalanobis distances are requested. This problem with singular within-class covariance matrices is discussed in Ripley (1996, p. 38). The use of the quasi inverse is an innovation introduced by SAS.

Figure 42.6 Plot of Data with Singular Within-Class Covariance Matrix

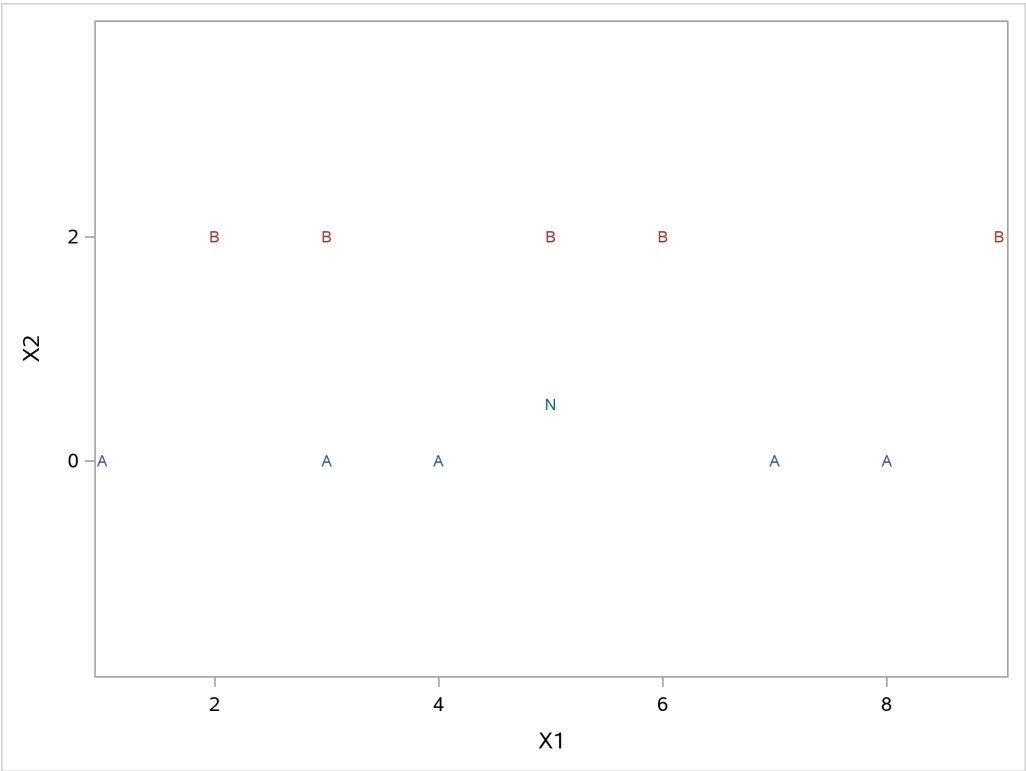

Let $\mathbf{S}$ be a singular covariance matrix. The matrix $\mathbf{S}$ can be either a within-group covariance matrix, a pooled covariance matrix, or a total-sample covariance matrix. Let v be the number of variables in the VAR statement, and let the nullity n be the number of variables among them with (partial) R square exceeding $1 - p$. If the determinant of $\mathbf{S}$ (Testing of Homogeneity of Within Covariance Matrices) or the inverse of $\mathbf{S}$ (Squared Distances and Generalized Squared Distances) is required, a quasi determinant or quasi inverse is used instead. With raw data input, PROC DISCRIM scales each variable to unit total-sample variance before calculating this quasi inverse. The calculation is based on the spectral decomposition $\mathbf{S} = \mathbf{\Gamma\Lambda\Gamma}'$, where $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues λ_j, $j = 1, \ldots, v$, where $\lambda_i \geq \lambda_j$ when $i < j$, and $\mathbf{\Gamma}$ is a matrix with the corresponding orthonormal eigenvectors of $\mathbf{S}$ as columns. When the nullity n is less than v, set $\lambda_j^0 = \lambda_j$ for

$j = 1, \ldots, v - n$, and $\lambda_j^0 = p\bar{\lambda}$ for $j = v - n + 1, \ldots, v$, where

$$\bar{\lambda} = \frac{1}{v - n} \sum_{k=1}^{v-n} \lambda_k$$

When the nullity n is equal to v, set $\lambda_j^0 = p$, for $j = 1, \ldots, v$. A quasi determinant is then defined as the product of λ_j^0, $j = 1, \ldots, v$. Similarly, a quasi inverse is then defined as $\mathbf{S}^* = \mathbf{\Gamma}\mathbf{\Lambda}^*\mathbf{\Gamma}'$, where $\mathbf{\Lambda}^*$ is a diagonal matrix of values $1/\lambda_j^0$, $j = 1, \ldots, v$.

Posterior Probability Error-Rate Estimates

The posterior probability error-rate estimates (Fukunaga and Kessel 1973; Glick 1978; Hora and Wilcox 1982) for each group are based on the posterior probabilities of the observations classified into that same group.

A sample of observations with classification results can be used to estimate the posterior error rates. The following notation is used to describe the sample:

$\mathcal{S}$	the set of observations in the (training) sample
n	the number of observations in $\mathcal{S}$
n_t	the number of observations in $\mathcal{S}$ in group t
$\mathcal{R}_t$	the set of observations such that the posterior probability belonging to group t is the largest
$\mathcal{R}_{ut}$	the set of observations from group u such that the posterior probability belonging to group t is the largest

The classification error rate for group t is defined as

$$e_t = 1 - \int_{\mathcal{R}_t} f_t(\mathbf{x}) d\mathbf{x}$$

The posterior probability of $\mathbf{x}$ for group t can be written as

$$p(t|\mathbf{x}) = \frac{q_t f_t(\mathbf{x})}{f(\mathbf{x})}$$

where $f(\mathbf{x}) = \sum_u q_u f_u(\mathbf{x})$ is the unconditional density of $\mathbf{x}$.

Thus, if you replace $f_t(\mathbf{x})$ with $p(t|\mathbf{x}) f(\mathbf{x})/q_t$, the error rate is

$$e_t = 1 - \frac{1}{q_t} \int_{\mathcal{R}_t} p(t|\mathbf{x}) f(\mathbf{x}) d\mathbf{x}$$

An estimator of e_t, unstratified over the groups from which the observations come, is then given by

$$\hat{e}_t \text{ (unstratified)} = 1 - \frac{1}{n q_t} \sum_{\mathcal{R}_t} p(t|\mathbf{x})$$

where $p(t|\mathbf{x})$ is estimated from the classification criterion, and the summation is over all sample observations of $\mathcal{S}$ classified into group t. The true group membership of each observation is not required in the estimation. The term nq_t is the number of observations that are expected to be classified into group t, given the priors. If more observations than expected are classified into group t, then $\hat{e}_t$ can be negative.

Further, if you replace $f(\mathbf{x})$ with $\sum_u q_u f_u(\mathbf{x})$, the error rate can be written as

$$e_t = 1 - \frac{1}{q_t} \sum_u q_u \int_{\mathcal{R}_{ut}} p(t|\mathbf{x}) f_u(\mathbf{x}) d\mathbf{x}$$

and an estimator stratified over the group from which the observations come is given by

$$\hat{e}_t \text{ (stratified)} = 1 - \frac{1}{q_t} \sum_u q_u \frac{1}{n_u} \left(\sum_{\mathcal{R}_{ut}} p(t|\mathbf{x}) \right)$$

The inner summation is over all sample observations of $\mathcal{S}$ coming from group u and classified into group t, and n_u is the number of observations originally from group u. The stratified estimate uses only the observations with known group membership. When the prior probabilities of the group membership are proportional to the group sizes, the stratified estimate is the same as the unstratified estimator.

The estimated group-specific error rates can be less than zero, usually due to a large discrepancy between prior probabilities of group membership and group sizes. To have a reliable estimate for group-specific error rate estimates, you should use group sizes that are at least approximately proportional to the prior probabilities of group membership.

A total error rate is defined as a weighted average of the individual group error rates

$$e = \sum_t q_t e_t$$

and can be estimated from

$$\hat{e} \text{ (unstratified)} = \sum_t q_t \hat{e}_t \text{ (unstratified)}$$

or

$$\hat{e} \text{ (stratified)} = \sum_t q_t \hat{e}_t \text{ (stratified)}$$

The total unstratified error-rate estimate can also be written as

$$\hat{e} \text{ (unstratified)} = 1 - \frac{1}{n} \sum_t \sum_{\mathcal{R}_t} p(t|\mathbf{x})$$

which is one minus the average value of the maximum posterior probabilities for each observation in the sample. The prior probabilities of group membership do not appear explicitly in this overall estimate.

Saving and Using Calibration Information

When you specify METHOD=NORMAL to derive a linear or quadratic discriminant function, you can save the calibration information developed by the DISCRIM procedure in a SAS data set by using the OUTSTAT= option in the procedure. PROC DISCRIM then creates a specially structured SAS data set of TYPE=LINEAR, TYPE=QUAD, or TYPE=MIXED that contains the calibration information. For more information about these data sets, see Appendix A, "Special SAS Data Sets." Calibration information cannot be saved when METHOD=NPAR, but you can classify a TESTDATA= data set in the same step. For an example of this, see Example 42.1.

To use this calibration information to classify observations in another data set, specify both of the following:

- the name of the calibration data set after the DATA= option in the PROC DISCRIM statement
- the name of the data set to be classified after the TESTDATA= option in the PROC DISCRIM statement

Here is an example:

```
data original;
   input position x1 x2;
   datalines;
...[data lines]
;

proc discrim outstat=info;
   class position;
run;

data check;
   input position x1 x2;
   datalines;
...[second set of data lines]
;

proc discrim data=info testdata=check testlist;
   class position;
run;
```

The first DATA step creates the SAS data set Original, which the DISCRIM procedure uses to develop a classification criterion. Specifying OUTSTAT=INFO in the PROC DISCRIM statement causes the DISCRIM procedure to store the calibration information in a new data set called Info. The next DATA step creates the data set Check. The second PROC DISCRIM statement specifies DATA=INFO and TESTDATA=CHECK so that the classification criterion developed earlier is applied to the Check data set. Note that if the CLASS variable is not present in the TESTDATA= data set, the output will not include misclassification statistics.

Input Data Sets

DATA= Data Set

When you specify METHOD=NPAR, an ordinary SAS data set is required as the input DATA= data set. When you specify METHOD=NORMAL, the DATA= data set can be an ordinary SAS data set or one of several specially structured data sets created by SAS/STAT procedures. These specially structured data sets include the following:

- TYPE=CORR data sets created by PROC CORR by using a BY statement
- TYPE=COV data sets created by PROC PRINCOMP by using both the COV option and a BY statement
- TYPE=CSSCP data sets created by PROC CORR by using the CSSCP option and a BY statement, where the OUT= data set is assigned TYPE=CSSCP with the TYPE= data set option
- TYPE=SSCP data sets created by PROC REG by using both the OUTSSCP= option and a BY statement
- TYPE=LINEAR, TYPE=QUAD, and TYPE=MIXED data sets produced by previous runs of PROC DISCRIM that used both METHOD=NORMAL and OUTSTAT= options

When the input data set is TYPE=CORR, TYPE=COV, TYPE=CSSCP, or TYPE=SSCP, the BY variable in these data sets becomes the CLASS variable in the DISCRIM procedure.

When the input data set is TYPE=CORR, TYPE=COV, or TYPE=CSSCP, then PROC DISCRIM reads the number of observations for each class from the observations with _TYPE_='N' and reads the variable means in each class from the observations with _TYPE_='MEAN'. Then PROC DISCRIM reads the within-class correlations from the observations with _TYPE_='CORR' and reads the standard deviations from the observations with _TYPE_='STD' (data set TYPE=CORR), the within-class covariances from the observations with _TYPE_='COV' (data set TYPE=COV), or the within-class corrected sums of squares and crossproducts from the observations with _TYPE_='CSSCP' (data set TYPE=CSSCP).

When you specify POOL=YES and the data set does not include any observations with _TYPE_='CSSCP' (data set TYPE=CSSCP), _TYPE_='COV' (data set TYPE=COV), or _TYPE_='CORR' (data set TYPE=CORR) for each class, PROC DISCRIM reads the pooled within-class information from the data set. In this case, PROC DISCRIM reads the pooled within-class covariances from the observations with _TYPE_='PCOV' (data set TYPE=COV) or reads the pooled within-class correlations from the observations with _TYPE_='PCORR' and the pooled within-class standard deviations from the observations with _TYPE_='PSTD' (data set TYPE=CORR) or the pooled within-class corrected SSCP matrix from the observations with _TYPE_='PSSCP' (data set TYPE=CSSCP).

When the input data set is TYPE=SSCP, the DISCRIM procedure reads the number of observations for each class from the observations with _TYPE_='N', the sum of weights of observations for each class from the variable INTERCEP in observations with _TYPE_='SSCP' and _NAME_='INTERCEPT', the variable sums from the analysis variables in observations with _TYPE_='SSCP' and _NAME_='INTERCEPT', and the uncorrected sums of squares and crossproducts from the analysis variables in observations with _TYPE_='SSCP' and _NAME_=*variable-name*.

When the input data set is TYPE=LINEAR, TYPE=QUAD, or TYPE=MIXED, then PROC DISCRIM reads the prior probabilities for each class from the observations with variable _TYPE_='PRIOR'.

When the input data set is TYPE=LINEAR, then PROC DISCRIM reads the coefficients of the linear discriminant functions from the observations with variable _TYPE_='LINEAR'.

When the input data set is TYPE=QUAD, then PROC DISCRIM reads the coefficients of the quadratic discriminant functions from the observations with variable _TYPE_='QUAD'.

When the input data set is TYPE=MIXED, then PROC DISCRIM reads the coefficients of the linear discriminant functions from the observations with variable _TYPE_='LINEAR'. If there are no observations with _TYPE_='LINEAR', then PROC DISCRIM reads the coefficients of the quadratic discriminant functions from the observations with variable _TYPE_='QUAD'.

TESTDATA= Data Set

The TESTDATA= data set is an ordinary SAS data set with observations that are to be classified. The quantitative variable names in this data set must match those in the DATA= data set. The TESTCLASS statement can be used to specify the variable containing group membership information of the TESTDATA= data set observations. When the TESTCLASS statement is missing and the TESTDATA= data set contains the variable given in the CLASS statement, this variable is used as the TESTCLASS variable. The TESTCLASS variable should have the same type (character or numeric) and length as the variable given in the CLASS statement. PROC DISCRIM considers an observation misclassified when the value of the TESTCLASS variable does not match the group into which the TESTDATA= observation is classified.

Output Data Sets

When an output data set includes variables containing the posterior probabilities of group membership (OUT=, OUTCROSS=, or TESTOUT= data sets) or group-specific density estimates (OUTD= or TESTOUTD= data sets), the names of these variables are constructed from the formatted values of the class levels converted to valid SAS variable names.

OUT= Data Set

The OUT= data set contains all the variables in the DATA= data set, plus new variables containing the posterior probabilities and the resubstitution classification results. The names of the new variables containing the posterior probabilities are constructed from the formatted values of the class levels converted to SAS names. A new variable, _INTO_, with the same attributes as the CLASS variable, specifies the class to which each observation is assigned. If an observation is labeled as 'Other', the variable _INTO_ has a missing value. When you specify the CANONICAL option, the data set also contains new variables with canonical variable scores. The NCAN= option determines the number of canonical variables. The names of the canonical variables are constructed as described in the CANPREFIX= option. The canonical variables have means equal to zero and pooled within-class variances equal to one.

An OUT= data set cannot be created if the DATA= data set is not an ordinary SAS data set.

OUTD= Data Set

The OUTD= data set contains all the variables in the DATA= data set, plus new variables containing the group-specific density estimates. The names of the new variables containing the density estimates are constructed from the formatted values of the class levels.

An OUTD= data set cannot be created if the DATA= data set is not an ordinary SAS data set.

OUTCROSS= Data Set

The OUTCROSS= data set contains all the variables in the DATA= data set, plus new variables containing the posterior probabilities and the classification results of cross validation. The names of the new variables containing the posterior probabilities are constructed from the formatted values of the class levels. A new variable, _INTO_, with the same attributes as the CLASS variable, specifies the class to which each observation is assigned. When an observation is labeled as `'Other'`, the variable _INTO_ has a missing value. When you specify the CANONICAL option, the data set also contains new variables with canonical variable scores. The NCAN= option determines the number of new variables. The names of the new variables are constructed as described in the CANPREFIX= option. The new variables have mean zero and pooled within-class variance equal to one.

An OUTCROSS= data set cannot be created if the DATA= data set is not an ordinary SAS data set.

TESTOUT= Data Set

The TESTOUT= data set contains all the variables in the TESTDATA= data set, plus new variables containing the posterior probabilities and the classification results. The names of the new variables containing the posterior probabilities are formed from the formatted values of the class levels. A new variable, _INTO_, with the same attributes as the CLASS variable, gives the class to which each observation is assigned. If an observation is labeled as `'Other'`, the variable _INTO_ has a missing value. When you specify the CANONICAL option, the data set also contains new variables with canonical variable scores. The NCAN= option determines the number of new variables. The names of the new variables are formed as described in the CANPREFIX= option.

TESTOUTD= Data Set

The TESTOUTD= data set contains all the variables in the TESTDATA= data set, plus new variables containing the group-specific density estimates. The names of the new variables containing the density estimates are formed from the formatted values of the class levels.

OUTSTAT= Data Set

The OUTSTAT= data set is similar to the TYPE=CORR data set produced by the CORR procedure. The data set contains various statistics such as means, standard deviations, and correlations. For an example of an OUTSTAT= data set, see Example 42.3. When you specify the CANONICAL option, canonical correlations, canonical structures, canonical coefficients, and means of canonical variables for each class are included in the data set.

If you specify METHOD=NORMAL, the output data set also includes coefficients of the discriminant functions, and the data set is TYPE=LINEAR (POOL=YES), TYPE=QUAD (POOL=NO), or TYPE=MIXED (POOL=TEST). If you specify METHOD=NPAR, this output data set is TYPE=CORR.

The OUTSTAT= data set contains the following variables:

- the BY variables, if any
- the CLASS variable
- _TYPE_, a character variable of length 8 that identifies the type of statistic
- _NAME_, a character variable that identifies the row of the matrix, the name of the canonical variable, or the type of the discriminant function coefficients
- the quantitative variables—that is, those in the VAR statement, or, if there is no VAR statement, all numeric variables not listed in any other statement

The observations, as identified by the variable _TYPE_, have the following values:

TYPE	**Contents**
N	number of observations both for the total sample (CLASS variable missing) and within each class (CLASS variable present)
SUMWGT	sum of weights both for the total sample (CLASS variable missing) and within each class (CLASS variable present), if a WEIGHT statement is specified
MEAN	means both for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PRIOR	prior probability for each class
STDMEAN	total-standardized class means
PSTDMEAN	pooled within-class standardized class means
STD	standard deviations both for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PSTD	pooled within-class standard deviations
BSTD	between-class standard deviations
RSQUARED	univariate R squares
LNDETERM	the natural log of the determinant or the natural log of the quasi determinant of the within-class covariance matrix either pooled (CLASS variable missing) or not pooled (CLASS variable present)

The following kinds of observations are identified by the combination of the variables _TYPE_ and _NAME_. When the _TYPE_ variable has one of the following values, the _NAME_ variable identifies the row of the matrix:

TYPE	**Contents**
CSSCP	corrected SSCP matrix both for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PSSCP	pooled within-class corrected SSCP matrix
BSSCP	between-class SSCP matrix
COV	covariance matrix both for the total sample (CLASS variable missing) and within each class (CLASS variable present)

PCOV	pooled within-class covariance matrix
BCOV	between-class covariance matrix
CORR	correlation matrix both for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PCORR	pooled within-class correlation matrix
BCORR	between-class correlation matrix

When you request canonical discriminant analysis, the _NAME_ variable identifies a canonical variable, and _TYPE_ variable can have one of the following values:

TYPE	**Contents**
CANCORR	canonical correlations
STRUCTUR	canonical structure
BSTRUCT	between canonical structure
PSTRUCT	pooled within-class canonical structure
SCORE	standardized canonical coefficients
RAWSCORE	raw canonical coefficients
CANMEAN	means of the canonical variables for each class

When you specify METHOD=NORMAL, the _NAME_ variable identifies different types of coefficients in the discriminant function, and the _TYPE_ variable can have one of the following values:

TYPE	**Contents**
LINEAR	coefficients of the linear discriminant functions
QUAD	coefficients of the quadratic discriminant functions

The values of the _NAME_ variable are as follows:

NAME	**Contents**
variable names	quadratic coefficients of the quadratic discriminant functions (a symmetric matrix for each class)
LINEAR	linear coefficients of the discriminant functions
CONST	constant coefficients of the discriminant functions

Computational Resources

In the following discussion, let

$$
\begin{aligned}
n &= \text{number of observations in the training data set} \\
v &= \text{number of variables} \\
c &= \text{number of class levels} \\
k &= \text{number of canonical variables} \\
l &= \text{length of the CLASS variable}
\end{aligned}
$$

Memory Requirements

The amount of temporary storage required depends on the discriminant method used and the options specified. The least amount of temporary storage in bytes needed to process the data is approximately

$$c(32v + 3l + 128) + 8v^2 + 104v + 4l$$

A parametric method (METHOD=NORMAL) requires an additional temporary memory of $12v^2 + 100v$ bytes. When you specify the CROSSVALIDATE option, this temporary storage must be increased by $4v^2 + 44v$ bytes. When a nonparametric method (METHOD=NPAR) is used, an additional temporary storage of $10v^2 + 94v$ bytes is needed if you specify METRIC=FULL to evaluate the distances.

With the MANOVA option, the temporary storage must be increased by $8v^2 + 96v$ bytes. The CANONICAL option requires a temporary storage of $2v^2 + 94v + 8k(v + c)$ bytes. The POSTERR option requires a temporary storage of $8c^2 + 64c + 96$ bytes. Additional temporary storage is also required for classification summary and for each output data set.

Consider the following statements:

```
proc discrim manova;
   class gp;
   var x1 x2 x3;
run;
```

If the CLASS variable gp has a length of 8 and the input data set contains two class levels, the procedure requires a temporary storage of 1992 bytes. This includes 1104 bytes for processing data, 480 bytes for using a parametric method, and 408 bytes for specifying the MANOVA option.

Time Requirements

The following factors determine the time requirements of discriminant analysis:

- The time needed for reading the data and computing covariance matrices is proportional to nv^2. PROC DISCRIM must also look up each class level in the list. This is faster if the data are sorted by the CLASS variable. The time for looking up class levels is proportional to a value ranging from n to $n \ln(c)$.
- The time for inverting a covariance matrix is proportional to v^3.
- With a parametric method, the time required to classify each observation is proportional to cv for a linear discriminant function and cv^2 for a quadratic discriminant function. When you specify the CROSSVALIDATE option, the discriminant function is updated for each observation in the classification. A substantial amount of time is required.
- With a nonparametric method, the data are stored in a tree structure (Friedman, Bentley, and Finkel 1977). The time required to organize the observations into the tree structure is proportional to $nv \ln(n)$. The time for performing each tree search is proportional to $\ln(n)$. When you specify the normal KERNEL= option, all observations in the training sample contribute to the density estimation and more computer time is needed.
- The time required for the canonical discriminant analysis is proportional to v^3.

Each of the preceding factors has a different machine-dependent constant of proportionality.

Displayed Output

The displayed output from PROC DISCRIM includes the class level information table. For each level of the classification variable, the following information is provided: the output data set variable name, frequency sum, weight sum, proportion of the total sample, and prior probability.

The optional output from PROC DISCRIM includes the following:

- Within-class SSCP matrices for each group
- Pooled within-class SSCP matrix
- Between-class SSCP matrix
- Total-sample SSCP matrix
- Within-class covariance matrices, $\mathbf{S}_t$, for each group
- Pooled within-class covariance matrix, $\mathbf{S}_p$
- Between-class covariance matrix, equal to the between-class SSCP matrix divided by $n(c-1)/c$, where n is the number of observations and c is the number of classes
- Total-sample covariance matrix

- Within-class correlation coefficients and $\Pr > |r|$ to test the hypothesis that the within-class population correlation coefficients are zero
- Pooled within-class correlation coefficients and $\Pr > |r|$ to test the hypothesis that the partial population correlation coefficients are zero
- Between-class correlation coefficients and $\Pr > |r|$ to test the hypothesis that the between-class population correlation coefficients are zero
- Total-sample correlation coefficients and $\Pr > |r|$ to test the hypothesis that the total population correlation coefficients are zero
- Simple statistics, including N (the number of observations), sum, mean, variance, and standard deviation both for the total sample and within each class
- Total-sample standardized class means, obtained by subtracting the grand mean from each class mean and dividing by the total-sample standard deviation
- Pooled within-class standardized class means, obtained by subtracting the grand mean from each class mean and dividing by the pooled within-class standard deviation
- Pairwise squared distances between groups
- Univariate test statistics, including total-sample standard deviations, pooled within-class standard deviations, between-class standard deviations, R square, $R^2/(1-R^2)$, F, and $\Pr > F$ (univariate F values and probability levels for one-way analyses of variance)
- Multivariate statistics and F approximations, including Wilks' lambda, Pillai's trace, Hotelling-Lawley trace, and Roy's greatest root with F approximations, numerator and denominator degrees of freedom (Num DF and Den DF), and probability values ($\Pr > F$). Each of these four multivariate statistics tests the hypothesis that the class means are equal in the population. See the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures," for more information.

If you specify METHOD=NORMAL, the following three statistics are displayed:

- Covariance matrix information, including covariance matrix rank and natural log of determinant of the covariance matrix for each group (POOL=TEST, POOL=NO) and for the pooled within-group (POOL=TEST, POOL=YES)
- Optionally, test of homogeneity of within covariance matrices (the results of a chi-square test of homogeneity of the within-group covariance matrices) (Morrison 1976; Kendall, Stuart, and Ord 1983; Anderson 1984)
- Pairwise generalized squared distances between groups

If the CANONICAL option is specified, the displayed output contains these statistics:

- Canonical correlations

- Adjusted canonical correlations (Lawley 1959). These are asymptotically less biased than the raw correlations and can be negative. The adjusted canonical correlations might not be computable and are displayed as missing values if two canonical correlations are nearly equal or if some are close to zero. A missing value is also displayed if an adjusted canonical correlation is larger than a previous adjusted canonical correlation.
- Approximate standard error of the canonical correlations
- Squared canonical correlations
- Eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$. Each eigenvalue is equal to $\rho^2/(1-\rho^2)$, where ρ^2 is the corresponding squared canonical correlation and can be interpreted as the ratio of between-class variation to within-class variation for the corresponding canonical variable. The table includes eigenvalues, differences between successive eigenvalues, proportion of the sum of the eigenvalues, and cumulative proportion.
- Likelihood ratio for the hypothesis that the current canonical correlation and all smaller ones are zero in the population. The likelihood ratio for all canonical correlations equals Wilks' lambda.
- Approximate F statistic based on Rao's approximation to the distribution of the likelihood ratio (Rao 1973, p. 556; Kshirsagar 1972, p. 326)
- Numerator degrees of freedom (Num DF), denominator degrees of freedom (Den DF), and $\Pr > F$, the probability level associated with the F statistic

The following statistic concerns the classification criterion:

- the linear discriminant function, but only if you specify METHOD=NORMAL and the pooled covariance matrix is used to calculate the (generalized) squared distances

When the input DATA= data set is an ordinary SAS data set, the displayed output includes the following:

- Optionally, the resubstitution results including the observation number (if an ID statement is included, the values of the ID variable are displayed instead of the observation number), the actual group for the observation, the group into which the developed criterion would classify it, and the posterior probability of membership in each group
- Resubstitution summary, a summary of the performance of the classification criterion based on resubstitution classification results
- Error count estimate of the resubstitution classification results
- Optionally, posterior probability error rate estimates of the resubstitution classification results

If you specify the CROSSVALIDATE option, the displayed output contains these statistics:

- Optionally, the cross validation results including the observation number (if an ID statement is included, the values of the ID variable are displayed instead of the observation number), the actual group for the observation, the group into which the developed criterion would classify it, and the posterior probability of membership in each group

- Cross validation summary, a summary of the performance of the classification criterion based on cross validation classification results
- Error count estimate of the cross validation classification results
- Optionally, posterior probability error rate estimates of the cross validation classification results

If you specify the TESTDATA= option, the displayed output contains these statistics:

- Optionally, the classification results including the observation number (if a TESTID statement is included, the values of the ID variable are displayed instead of the observation number), the actual group for the observation (if a TESTCLASS statement is included), the group into which the developed criterion would classify it, and the posterior probability of membership in each group
- Classification summary, a summary of the performance of the classification criterion
- Error count estimate of the test data classification results
- Optionally, posterior probability error rate estimates of the test data classification results

Formulas

By default, PROC DISCRIM displays formulas such as the following in the LISTING destination:

```
      2                 _   _                  -1   _   _
    D  (i|j)  =  (X  -  X  )'  DIAG(COV  )   (X  -  X  )
                   i     j              j      i     j

                  _                          _      2
                 |        1          1      |   2P  + 3P  - 1
  RHO   = 1.0 -  | SUM -----   -   ---     | -------------
                 |_      N(i)        N     _|   6(P+1)(K-1)

 2          _         -1    _
D  (X)  =  (X-X  )'  COV    (X-X  )  +  ln  |COV  |  -  2  ln  PRIOR
 j            j        j      j              j                  j
```

These formulas are properly displayed only with uniform fonts. Formulas are not displayed by default in destinations such as HTML, RTF, and PDF, because destinations other than LISTING typically use proportional fonts. You can use the FORMULA option to display formulas in destinations other than LISTING.

In the LISTING destination, formulas are displayed as notes. In other destinations, formulas are displayed with batch capture (which provides uniform fonts). Therefore, the names that are used to select and exclude formulas from the LISTING destination are different from the names that are used with other destinations.

By default, batch capture results are displayed in a box. You can remove the box and change the color and font of the formulas as follows:

```
proc template;
   define style styles.batch; parent=styles.htmlblue;
      style fonts from fonts / "BatchFixedFont" =
         ("SAS Monospace, <monospace>, Courier, monospace", 2, bold);
      style batch from batch / frame=void foreground=colors('notefg');
   end;
run;

ods listing;
ods html style=batch body='dis.html';
ods rtf  style=batch body='dis.rtf';
ods pdf  style=batch body='dis.pdf';

proc discrim data=sashelp.iris formula;
   class Species;
   var Peta:;
run;

ods _all_ close;
```

The results of these steps are not displayed.

ODS Table Names

PROC DISCRIM assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 42.2. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 42.2 ODS Tables Produced by PROC DISCRIM

ODS Table Name	Description	PROC DISCRIM Option
ANOVA	Univariate statistics	ANOVA
AvePostCrossVal	Average posterior probabilities, cross validation	POSTERR & CROSSVALIDATE
AvePostResub	Average posterior probabilities, resubstitution	POSTERR
AvePostTestClass	Average posterior probabilities, test classification	POSTERR & TEST=
AveRSquare	Average R-square	ANOVA
BCorr	Between-class correlations	BCORR
BCov	Between-class covariances	BCOV
BSSCP	Between-class SSCP matrix	BSSCP
BStruc	Between canonical structure	CANONICAL
CanCorr	Canonical correlations	CANONICAL
CanonicalMeans	Class means on canonical variables	CANONICAL
ChiSq	Chi-square information	POOL=TEST

Table 42.2 *continued*

ODS Table Name	Description	PROC DISCRIM Option
ClassifiedCrossVal	Number of observations and percent classified, cross validation	CROSSVALIDATE
ClassifiedResub	Number of observations and percent classified, resubstitution	Default
ClassifiedTestClass	Number of observations and percent classified, test classification	TEST=
Counts	Number of observations, variables, classes, df	Default
CovDF	DF for covariance matrices, not displayed	Any *COV option
Dist	Squared distances	DISTANCE
DistFValues	*F* values based on squared distances	DISTANCE
DistGeneralized	Generalized squared distances	Default
DistProb	Probabilities for *F* values from squared distances	DISTANCE
ErrorCrossVal	Error count estimates, cross validation	CROSSVALIDATE
ErrorResub	Error count estimates, resubstitution	Default
ErrorTestClass	Error count estimates, test classification	TEST=
Levels	Class level information	Default
LinearDiscFunc	Linear discriminant function	POOL=YES
LogDet	Log determinant of the covariance matrix	Default
MultStat	MANOVA	MANOVA
NObs	Number of observations	Default
PCoef	Pooled standard canonical coefficients	CANONICAL
PCorr	Pooled within-class correlations	PCORR
PCov	Pooled within-class covariances	PCOV
PSSCP	Pooled within-class SSCP matrix	PSSCP
PStdMeans	Pooled standardized class means	STDMEAN
PStruc	Pooled within canonical structure	CANONICAL
PostCrossVal	Posterior probabilities, cross validation	CROSSLIST or CROSSLISTERR
PostErrCrossVal	Posterior error estimates, cross validation	POSTERR & CROSSVALIDATE
PostErrResub	Posterior error estimates, resubstitution	POSTERR

Table 42.2 *continued*

ODS Table Name	Description	PROC DISCRIM Option
PostErrTestClass	Posterior error estimates, test classification	POSTERR & TEST=
PostResub	Posterior probabilities, resubstitution	LIST or LISTERR
PostTestClass	Posterior probabilities, test classification	TESTLIST or TESTLISTERR
RCoef	Raw canonical coefficients	CANONICAL
SimpleStatistics	Simple statistics	SIMPLE
TCoef	Total-sample standard canonical coefficients	CANONICAL
TCorr	Total-sample correlations	TCORR
TCov	Total-sample covariances	TCOV
TSSCP	Total-sample SSCP matrix	TSSCP
TStdMeans	Total standardized class means	STDMEAN
TStruc	Total canonical structure	CANONICAL
WCorr	Within-class correlations	WCORR
WCov	Within-class covariances	WCOV
WSSCP	Within-class SSCP matrices	WSSCP

Examples: DISCRIM Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

The iris data published by Fisher (1936) are widely used for examples in discriminant analysis and cluster analysis. The sepal length, sepal width, petal length, and petal width are measured in millimeters on 50 iris specimens from each of three species: *Iris setosa*, *I. versicolor*, and *I. virginica*. The iris data set is available from the Sashelp library.

Example 42.1: Univariate Density Estimates and Posterior Probabilities

(View the complete code for this example (discex1.sas) in the example repository.)

In this example, several discriminant analyses are run with a single quantitative variable, petal width, so that density estimates and posterior probabilities can be plotted easily. The example produces Output 42.1.1 through Output 42.1.5. ODS Graphics is used to display the sample distribution of petal width in the three species. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." Note the overlap between the species *I. versicolor* and *I. virginica* that the bar chart shows. The following statements produce Output 42.1.1:

```
title 'Discriminant Analysis of Fisher (1936) Iris Data';

proc freq data=sashelp.iris noprint;
   tables petalwidth * species / out=freqout;
run;

proc sgplot data=freqout;
   vbar petalwidth / response=count group=species;
   keylegend / location=inside position=ne noborder across=1;
run;
```

Output 42.1.1 Sample Distribution of Petal Width in Three Species

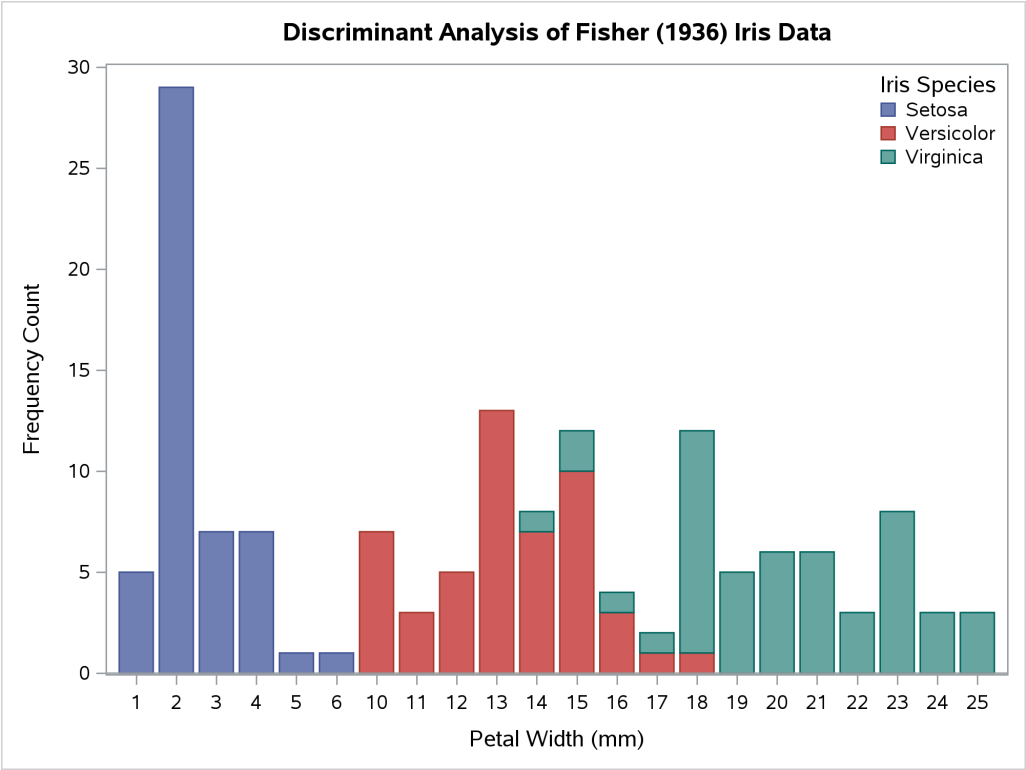

In order to plot the density estimates and posterior probabilities, a data set called plotdata is created containing equally spaced values from –5 to 30, covering the range of petal width with a little to spare on each end. The plotdata data set is used with the TESTDATA= option in PROC DISCRIM. The following statements make the data set:

```
data plotdata;
   do PetalWidth=-5 to 30 by 0.5;
      output;
   end;
run;
```

The same plots are produced after each discriminant analysis, so macros are used to reduce the amount of typing required. The macros use two data sets. The data set plotd, containing density estimates, is created by the TESTOUTD= option in PROC DISCRIM. The data set plotp, containing posterior probabilities, is

created by the TESTOUT= option. For each data set, the macros remove uninteresting values (near zero) and create an overlay plot showing all three species in a single plot.

The following statements create the macros:

```
%macro plotden;
   title3 'Plot of Estimated Densities';

   data plotd2;
      set plotd;
      if setosa     < .002 then setosa     = .;
      if versicolor < .002 then versicolor = .;
      if virginica  < .002 then virginica  = .;
      g = 'Setosa    '; Density = setosa;     output;
      g = 'Versicolor'; Density = versicolor; output;
      g = 'Virginica '; Density = virginica;  output;
      label PetalWidth='Petal Width in mm.';
   run;

   proc sgplot data=plotd2;
      series y=Density x=PetalWidth / group=g;
      discretelegend;
   run;
%mend;

%macro plotprob;
   title3 'Plot of Posterior Probabilities';

   data plotp2;
      set plotp;
      if setosa     < .01 then setosa     = .;
      if versicolor < .01 then versicolor = .;
      if virginica  < .01 then virginica  = .;
      g = 'Setosa    '; Probability = setosa;     output;
      g = 'Versicolor'; Probability = versicolor; output;
      g = 'Virginica '; Probability = virginica;  output;
      label PetalWidth='Petal Width in mm.';
   run;

   proc sgplot data=plotp2;
      series y=Probability x=PetalWidth / group=g;
      discretelegend;
   run;
%mend;
```

The first analysis uses normal-theory methods (METHOD=NORMAL) assuming equal variances (POOL=YES) in the three classes. The NOCLASSIFY option suppresses the resubstitution classification results of the input data set observations. The CROSSLISTERR option lists the observations that are misclassified under cross validation and displays cross validation error-rate estimates. The following statements produce Output 42.1.2:

```
title2 'Using Normal Density Estimates with Equal Variance';

proc discrim data=sashelp.iris method=normal pool=yes
   testdata=plotdata testout=plotp testoutd=plotd
   short noclassify crosslisterr;
   class Species;
   var PetalWidth;
run;

%plotden;
%plotprob;
```

Output 42.1.2 Normal Density Estimates with Equal Variance

Discriminant Analysis of Fisher (1936) Iris Data
Using Normal Density Estimates with Equal Variance

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	1	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Discriminant Analysis of Fisher (1936) Iris Data
Using Normal Density Estimates with Equal Variance

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Linear Discriminant Function

Posterior Probability of Membership in Species

Obs	From Species	Classified into Species		Setosa	Versicolor	Virginica
53	**Versicolor**	**Virginica**	*	0.0000	0.0952	0.9048
100	**Versicolor**	**Virginica**	*	0.0000	0.3828	0.6172
103	**Virginica**	**Versicolor**	*	0.0000	0.9610	0.0390
124	**Virginica**	**Versicolor**	*	0.0000	0.9940	0.0060
130	**Virginica**	**Versicolor**	*	0.0000	0.8009	0.1991
136	**Virginica**	**Versicolor**	*	0.0000	0.9610	0.0390

*** Misclassified observation**

Output 42.1.2 *continued*

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Equal Variance

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Linear Discriminant Function

Number of Observations and Percent Classified into Species

From Species	Setosa	Versicolor	Virginica	Total
Setosa	50	0	0	50
	100.00	0.00	0.00	100.00
Versicolor	0	48	2	50
	0.00	96.00	4.00	100.00
Virginica	0	4	46	50
	0.00	8.00	92.00	100.00
Total	50	52	48	150
	33.33	34.67	32.00	100.00
Priors	0.33333	0.33333	0.33333	

Error Count Estimates for Species

	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0400	0.0800	0.0400
Priors	0.3333	0.3333	0.3333	

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Equal Variance

The DISCRIM Procedure
Classification Summary for Test Data: WORK.PLOTDATA
Classification Summary using Linear Discriminant Function

Observation Profile for Test Data	
Number of Observations Read	71
Number of Observations Used	71

Number of Observations and Percent Classified into Species

	Setosa	Versicolor	Virginica	Total
Total	26	18	27	71
	36.62	25.35	38.03	100.00
Priors	0.33333	0.33333	0.33333	

Output 42.1.2 *continued*

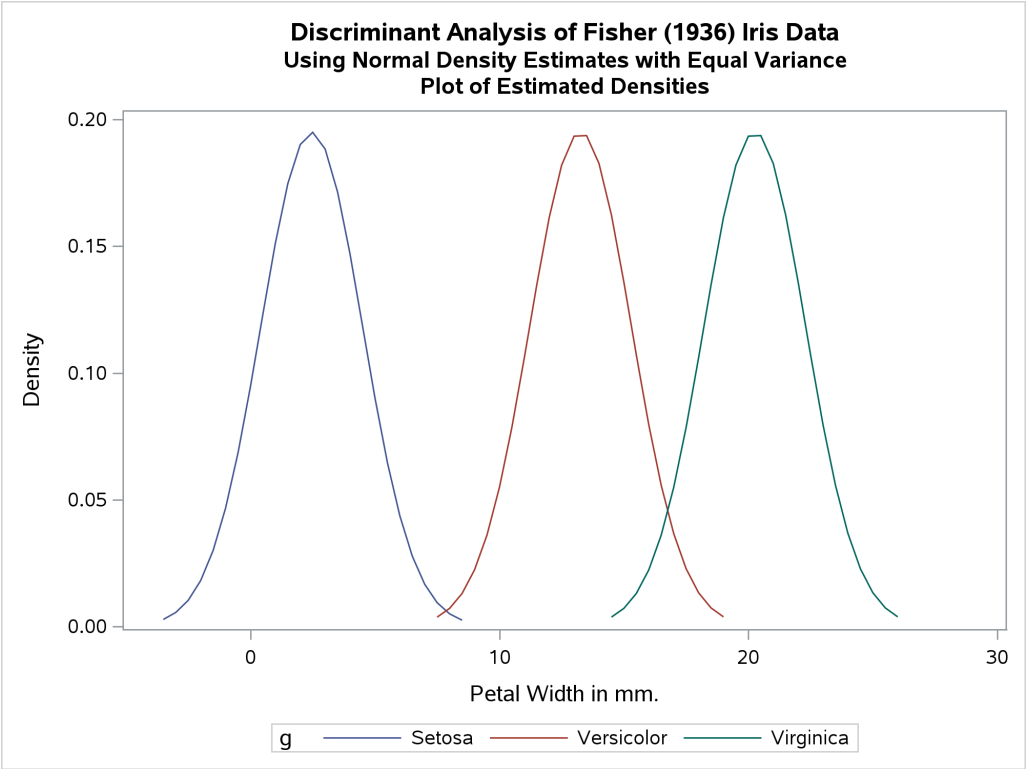

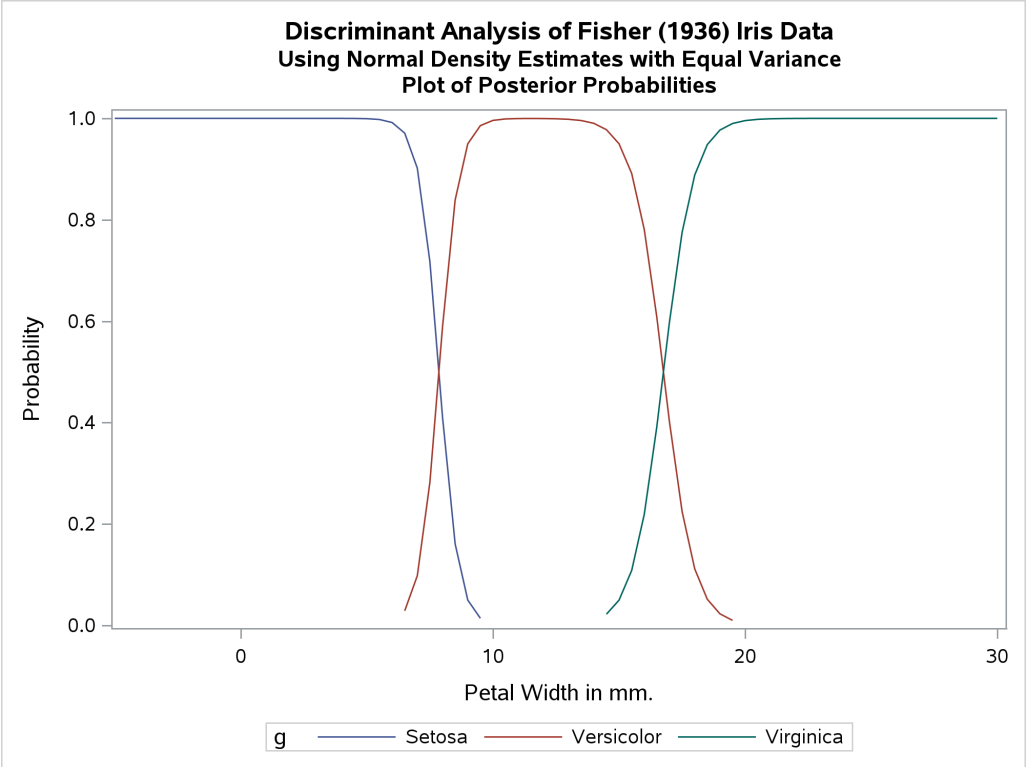

The next analysis uses normal-theory methods assuming unequal variances (POOL=NO) in the three classes.

The following statements produce Output 42.1.3:

```
title2 'Using Normal Density Estimates with Unequal Variance';

proc discrim data=sashelp.iris method=normal pool=no
   testdata=plotdata testout=plotp testoutd=plotd
   short noclassify crosslisterr;
   class Species;
   var PetalWidth;
run;

%plotden;
%plotprob;
```

Output 42.1.3 Normal Density Estimates with Unequal Variance

Discriminant Analysis of Fisher (1936) Iris Data
Using Normal Density Estimates with Unequal Variance

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	1	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Discriminant Analysis of Fisher (1936) Iris Data
Using Normal Density Estimates with Unequal Variance

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Quadratic Discriminant Function

Posterior Probability of Membership in Species

Obs	From Species	Classified into Species		Setosa	Versicolor	Virginica
10	**Setosa**	**Versicolor**	*	0.4923	0.5073	0.0004
53	**Versicolor**	**Virginica**	*	0.0000	0.0686	0.9314
100	**Versicolor**	**Virginica**	*	0.0000	0.2871	0.7129
103	**Virginica**	**Versicolor**	*	0.0000	0.8740	0.1260
124	**Virginica**	**Versicolor**	*	0.0000	0.9602	0.0398
130	**Virginica**	**Versicolor**	*	0.0000	0.6558	0.3442
136	**Virginica**	**Versicolor**	*	0.0000	0.8740	0.1260

*** Misclassified observation**

Output 42.1.3 *continued*

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Unequal Variance

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Quadratic Discriminant Function

Number of Observations and Percent Classified into Species				
From Species	Setosa	Versicolor	Virginica	Total
Setosa	49	1	0	50
	98.00	2.00	0.00	100.00
Versicolor	0	48	2	50
	0.00	96.00	4.00	100.00
Virginica	0	4	46	50
	0.00	8.00	92.00	100.00
Total	49	53	48	150
	32.67	35.33	32.00	100.00
Priors	0.33333	0.33333	0.33333	

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0200	0.0400	0.0800	0.0467
Priors	0.3333	0.3333	0.3333	

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Unequal Variance

The DISCRIM Procedure
Classification Summary for Test Data: WORK.PLOTDATA
Classification Summary using Quadratic Discriminant Function

Observation Profile for Test Data	
Number of Observations Read	71
Number of Observations Used	71

Number of Observations and Percent Classified into Species				
	Setosa	Versicolor	Virginica	Total
Total	23	20	28	71
	32.39	28.17	39.44	100.00
Priors	0.33333	0.33333	0.33333	

Output 42.1.3 *continued*

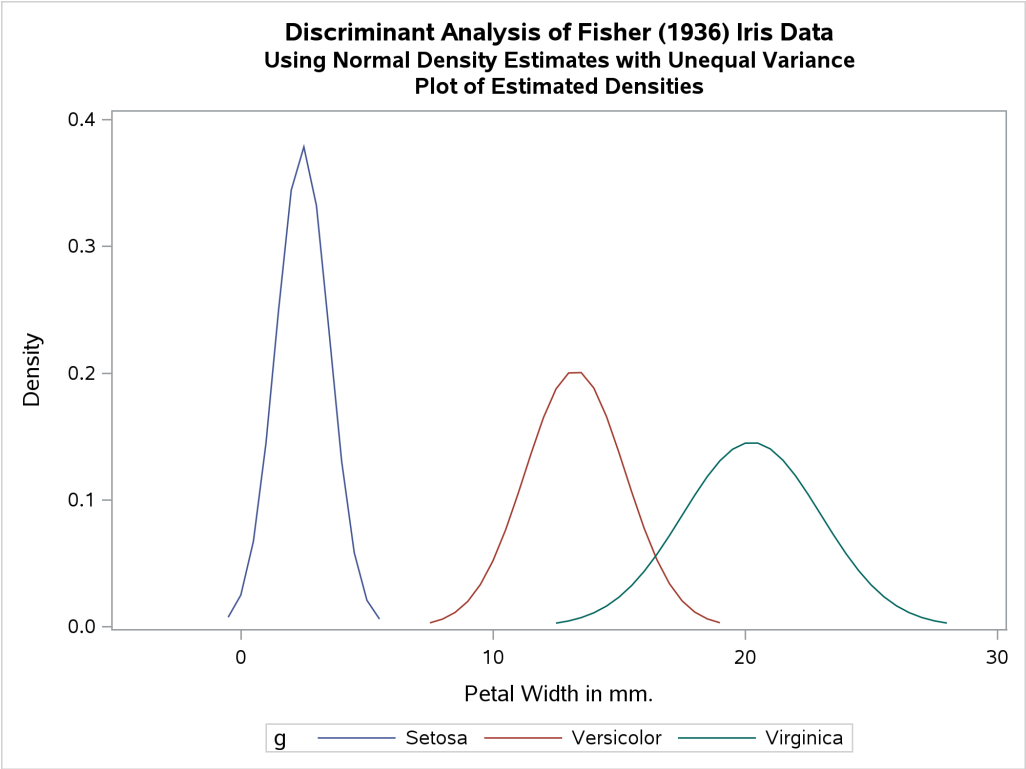

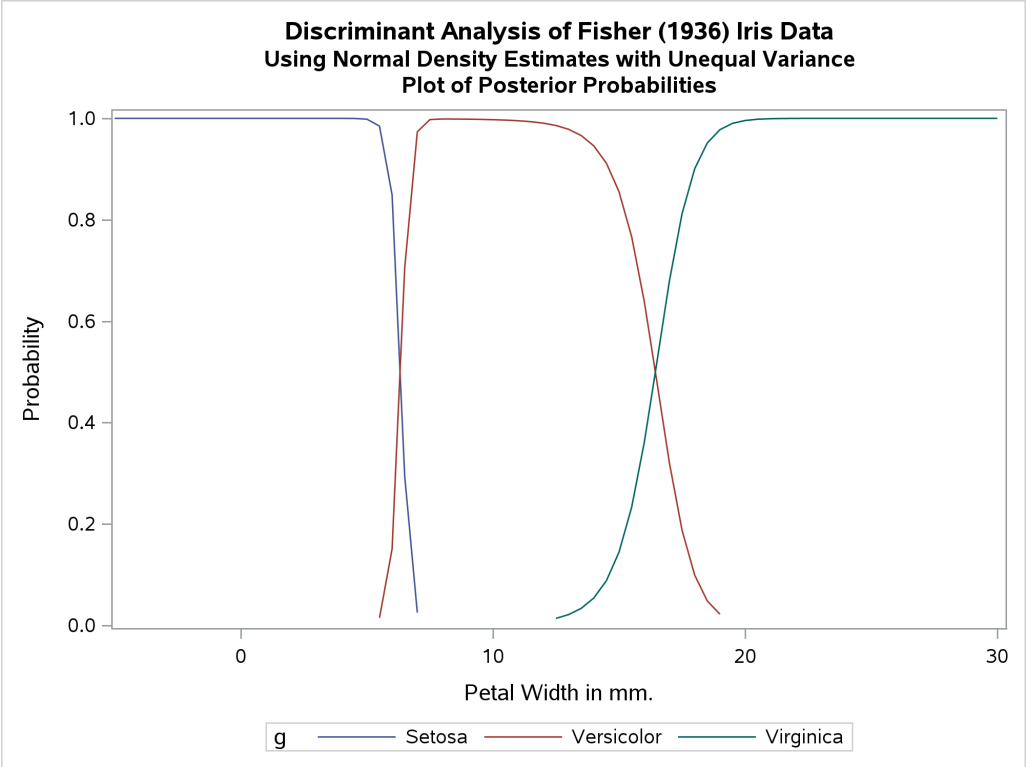

Two more analyses are run with nonparametric methods (METHOD=NPAR), specifically kernel density

estimates with normal kernels (KERNEL=NORMAL). The first of these uses equal bandwidths (smoothing parameters) (POOL=YES) in each class. The use of equal bandwidths does not constrain the density estimates to be of equal variance. The value of the radius parameter that, assuming normality, minimizes an approximate mean integrated square error is 0.48 (see the section "Nonparametric Methods" on page 2740). Choosing $r = 0.4$ gives a more detailed look at the irregularities in the data. The following statements produce Output 42.1.4:

```
title2 'Using Kernel Density Estimates with Equal Bandwidth';

proc discrim data=sashelp.iris method=npar kernel=normal
   r=.4 pool=yes testdata=plotdata testout=plotp
   testoutd=plotd short noclassify crosslisterr;
   class Species;
   var PetalWidth;
run;

%plotden;
%plotprob;
```

Output 42.1.4 Kernel Density Estimates with Equal Bandwidth

Discriminant Analysis of Fisher (1936) Iris Data
Using Kernel Density Estimates with Equal Bandwidth

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	1	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Output 42.1.4 *continued*

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Equal Bandwidth

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Normal Kernel Density

			Posterior Probability of Membership in Species		
Obs	From Species	Classified into Species	Setosa	Versicolor	Virginica
53	Versicolor	Virginica *	0.0000	0.0438	0.9562
100	Versicolor	Virginica *	0.0000	0.2586	0.7414
103	Virginica	Versicolor *	0.0000	0.8827	0.1173
124	Virginica	Versicolor *	0.0000	0.9472	0.0528
130	Virginica	Versicolor *	0.0000	0.8061	0.1939
136	Virginica	Versicolor *	0.0000	0.8827	0.1173

*** Misclassified observation**

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Equal Bandwidth

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Normal Kernel Density

Number of Observations and Percent Classified into Species				
From Species	Setosa	Versicolor	Virginica	Total
Setosa	50	0	0	50
	100.00	0.00	0.00	100.00
Versicolor	0	48	2	50
	0.00	96.00	4.00	100.00
Virginica	0	4	46	50
	0.00	8.00	92.00	100.00
Total	50	52	48	150
	33.33	34.67	32.00	100.00
Priors	0.33333	0.33333	0.33333	

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0400	0.0800	0.0400
Priors	0.3333	0.3333	0.3333	

Output 42.1.4 *continued*

Discriminant Analysis of Fisher (1936) Iris Data
Using Kernel Density Estimates with Equal Bandwidth

The DISCRIM Procedure
Classification Summary for Test Data: WORK.PLOTDATA
Classification Summary using Normal Kernel Density

Observation Profile for Test Data	
Number of Observations Read	71
Number of Observations Used	71

Number of Observations and Percent Classified into Species				
	Setosa	Versicolor	Virginica	Total
Total	26	18	27	71
	36.62	25.35	38.03	100.00
Priors	0.33333	0.33333	0.33333	

Output 42.1.4 *continued*

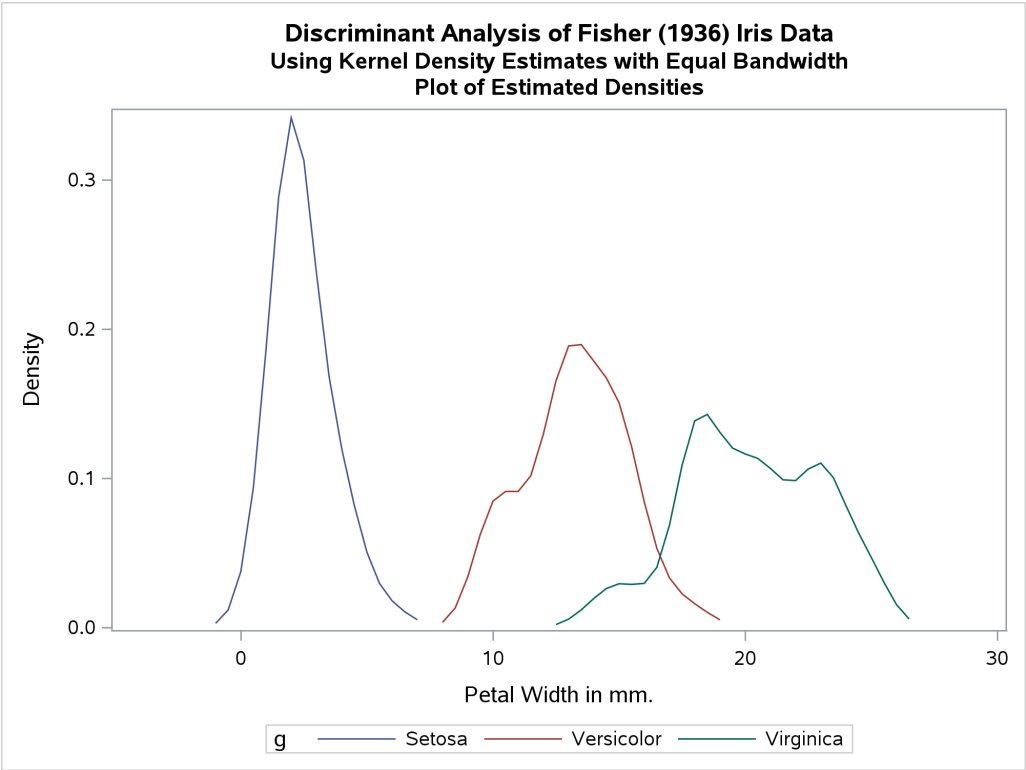

Output 42.1.4 *continued*

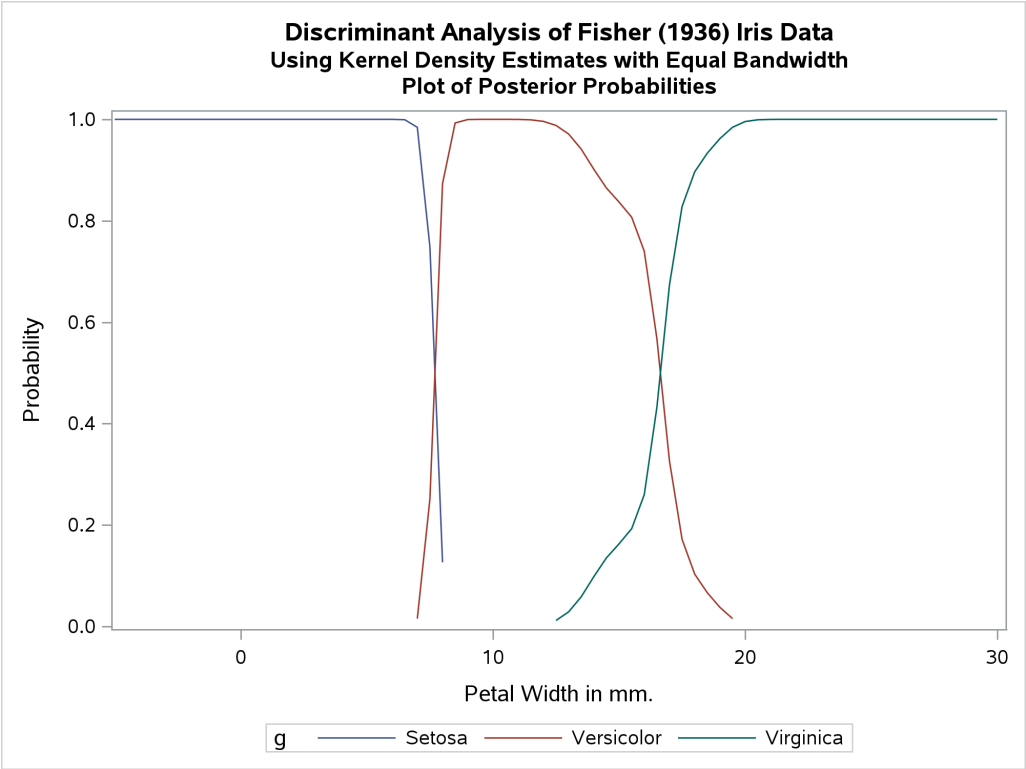

Another nonparametric analysis is run with unequal bandwidths (POOL=NO). The following statements produce Output 42.1.5:

```
title2 'Using Kernel Density Estimates with Unequal Bandwidth';

proc discrim data=sashelp.iris method=npar kernel=normal
   r=.4 pool=no testdata=plotdata testout=plotp
   testoutd=plotd short noclassify crosslisterr;
   class Species;
   var PetalWidth;
run;

%plotden;
%plotprob;
```

Output 42.1.5 Kernel Density Estimates with Unequal Bandwidth

Discriminant Analysis of Fisher (1936) Iris Data
Using Kernel Density Estimates with Unequal Bandwidth

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	1	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Output 42.1.5 *continued*

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Unequal Bandwidth

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Normal Kernel Density

Posterior Probability of Membership in Species

Obs	From Species	Classified into Species		Setosa	Versicolor	Virginica
53	**Versicolor**	**Virginica**	*	0.0000	0.0475	0.9525
100	**Versicolor**	**Virginica**	*	0.0000	0.2310	0.7690
103	**Virginica**	**Versicolor**	*	0.0000	0.8805	0.1195
124	**Virginica**	**Versicolor**	*	0.0000	0.9394	0.0606
130	**Virginica**	**Versicolor**	*	0.0000	0.7193	0.2807
136	**Virginica**	**Versicolor**	*	0.0000	0.8805	0.1195

*** Misclassified observation**

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Unequal Bandwidth

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Normal Kernel Density

Number of Observations and Percent Classified into Species

From Species	Setosa	Versicolor	Virginica	Total
Setosa	50 100.00	0 0.00	0 0.00	50 100.00
Versicolor	0 0.00	48 96.00	2 4.00	50 100.00
Virginica	0 0.00	4 8.00	46 92.00	50 100.00
Total	50 33.33	52 34.67	48 32.00	150 100.00
Priors	0.33333	0.33333	0.33333	

Output 42.1.5 *continued*

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0400	0.0800	0.0400
Priors	0.3333	0.3333	0.3333	

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Unequal Bandwidth

The DISCRIM Procedure
Classification Summary for Test Data: WORK.PLOTDATA
Classification Summary using Normal Kernel Density

Observation Profile for Test Data	
Number of Observations Read	71
Number of Observations Used	71

Number of Observations and Percent Classified into Species				
	Setosa	Versicolor	Virginica	Total
Total	25 35.21	18 25.35	28 39.44	71 100.00
Priors	0.33333	0.33333	0.33333	

Output 42.1.5 *continued*

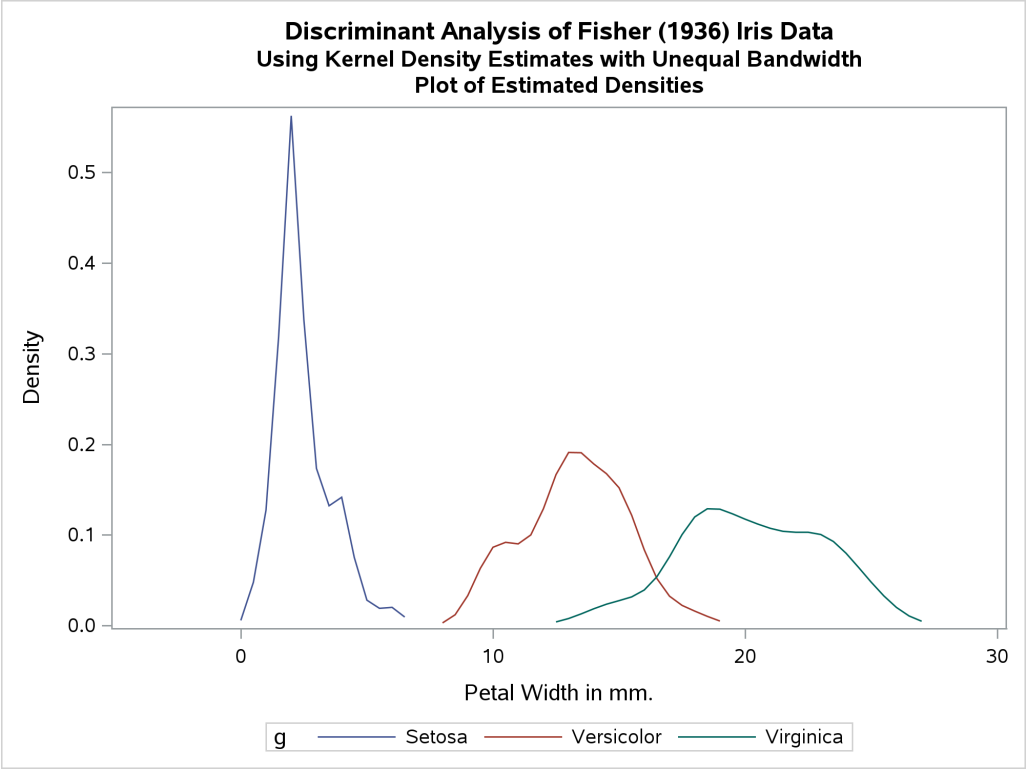

Output 42.1.5 *continued*

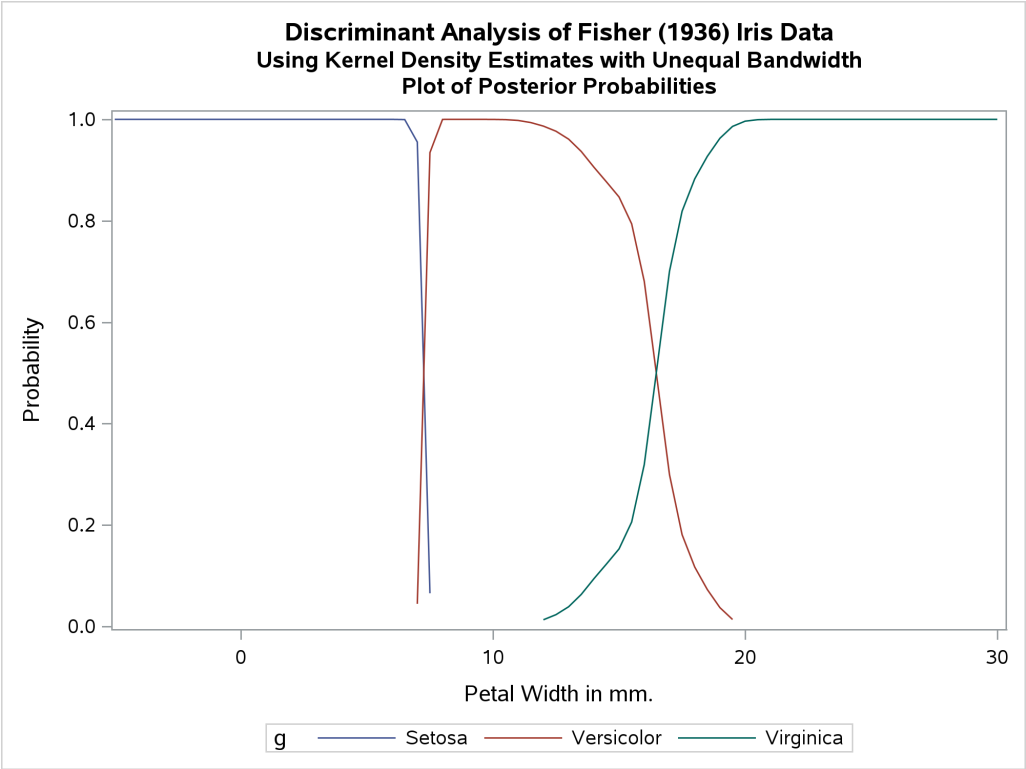

Example 42.2: Bivariate Density Estimates and Posterior Probabilities

(View the complete code for this example (discex2.sas) in the example repository.)

In this example, four more discriminant analyses of iris data are run with two quantitative variables: petal width and petal length. The following statements produce Output 42.2.1 through Output 42.2.5:

```
title 'Discriminant Analysis of Fisher (1936) Iris Data';
proc template;
   define statgraph scatter;
      begingraph;
         entrytitle 'Fisher (1936) Iris Data';
         layout overlayequated / equatetype=fit;
            scatterplot x=petallength y=petalwidth /
                        group=species name='iris';
            layout gridded / autoalign=(topleft);
               discretelegend 'iris' / border=false opaque=false;
            endlayout;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=sashelp.iris template=scatter;
run;
```

The scatter plot in Output 42.2.1 shows the joint sample distribution.

Output 42.2.1 Joint Sample Distribution of Petal Width and Petal Length in Three Species

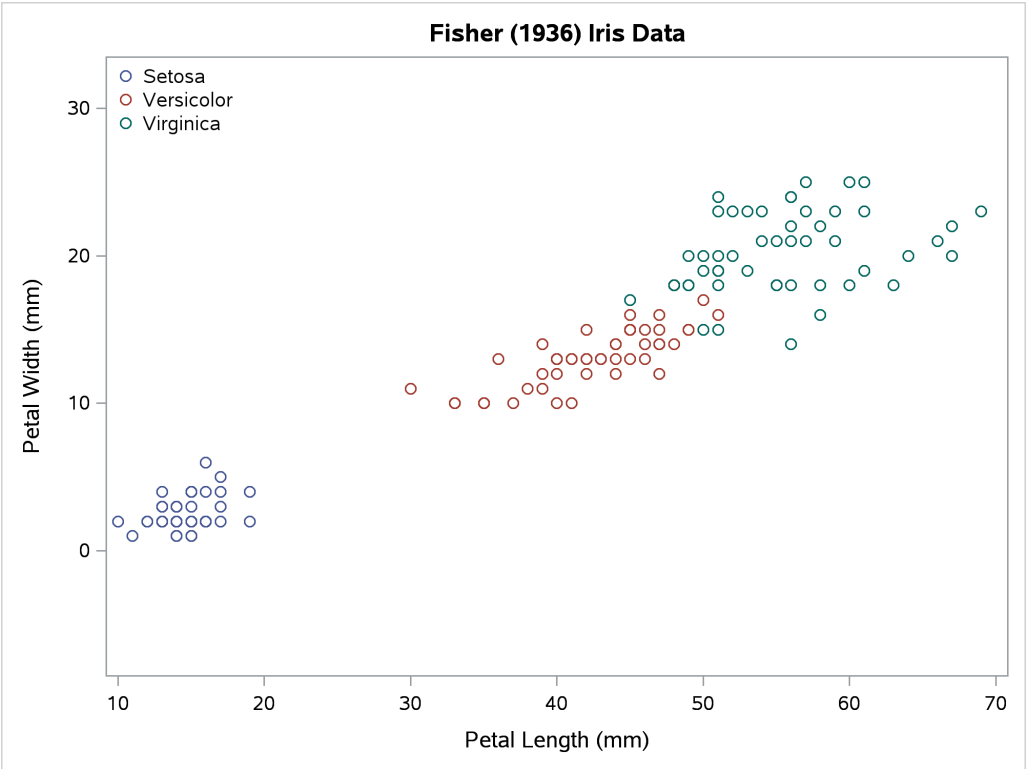

Another data set is created for plotting, containing a grid of points suitable for contour plots. The following statements create the data set:

```
data plotdata;
   do PetalLength = -2 to 72 by 0.5;
      do PetalWidth= - 5 to 32 by 0.5;
         output;
      end;
   end;
run;
```

Three macros are defined as follows to make contour plots of density estimates, posterior probabilities, and classification results:

```
%let close = thresholdmin=0 thresholdmax=0 offsetmin=0 offsetmax=0;
%let close = xaxisopts=(&close) yaxisopts=(&close);

proc template;
   define statgraph contour;
      begingraph;
         layout overlayequated / equatetype=equate &close;
            contourplotparm x=petallength y=petalwidth z=z /
                            contourtype=fill nhint=30;
            scatterplot x=pl y=pw / group=species name='iris'
```

```
                        includemissinggroup=false primary=true;
            layout gridded / autoalign=(topleft);
               discretelegend 'iris' / border=false opaque=false;
            endlayout;
         endlayout;
      endgraph;
   end;
run;

%macro contden;
   data contour(keep=PetalWidth PetalLength species z pl pw);
      merge plotd(in=d) sashelp.iris(keep=PetalWidth PetalLength species
                                     rename=(PetalWidth=pw PetalLength=pl));
      if d then z = max(setosa,versicolor,virginica);
   run;

   title3 'Plot of Estimated Densities';

   proc sgrender data=contour template=contour;
   run;
%mend;

%macro contprob;
   data posterior(keep=PetalWidth PetalLength species z pl pw into);
      merge plotp(in=d) sashelp.iris(keep=PetalWidth PetalLength species
                                     rename=(PetalWidth=pw PetalLength=pl));
      if d then z = max(setosa,versicolor,virginica);
      into = 1 * (_into_ =: 'Set') + 2 * (_into_ =: 'Ver') +
             3 * (_into_ =: 'Vir');
   run;

   title3 'Plot of Posterior Probabilities ';

   proc sgrender data=posterior template=contour;
   run;
%mend;

%macro contclass;
   title3 'Plot of Classification Results';

   proc sgrender data=posterior(drop=z rename=(into=z)) template=contour;
   run;
%mend;
```

A normal-theory analysis (METHOD=NORMAL) assuming equal covariance matrices (POOL=YES) illustrates the linearity of the classification boundaries. These statements produce Output 42.2.2:

```
title2 'Using Normal Density Estimates with Equal Variance';

proc discrim data=sashelp.iris method=normal pool=yes
   testdata=plotdata testout=plotp testoutd=plotd
   short noclassify crosslisterr;
   class Species;
   var Petal:;
```

```
run;

%contden
%contprob
%contclass
```

Output 42.2.2 Normal Density Estimates with Equal Variance

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Equal Variance

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	2	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Equal Variance

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Linear Discriminant Function

Posterior Probability of Membership in Species

Obs	From Species	Classified into Species		Setosa	Versicolor	Virginica
53	**Versicolor**	**Virginica**	*	0.0000	0.2130	0.7870
100	**Versicolor**	**Virginica**	*	0.0000	0.3118	0.6882
103	**Virginica**	**Versicolor**	*	0.0000	0.8453	0.1547
113	**Virginica**	**Versicolor**	*	0.0000	0.8322	0.1678
124	**Virginica**	**Versicolor**	*	0.0000	0.8057	0.1943
136	**Virginica**	**Versicolor**	*	0.0000	0.8903	0.1097

*** Misclassified observation**

Output 42.2.2 *continued*

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Equal Variance

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Linear Discriminant Function

Number of Observations and Percent Classified into Species				
From Species	Setosa	Versicolor	Virginica	Total
Setosa	50 100.00	0 0.00	0 0.00	50 100.00
Versicolor	0 0.00	48 96.00	2 4.00	50 100.00
Virginica	0 0.00	4 8.00	46 92.00	50 100.00
Total	50 33.33	52 34.67	48 32.00	150 100.00
Priors	0.33333	0.33333	0.33333	

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0400	0.0800	0.0400
Priors	0.3333	0.3333	0.3333	

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Equal Variance

The DISCRIM Procedure
Classification Summary for Test Data: WORK.PLOTDATA
Classification Summary using Linear Discriminant Function

Observation Profile for Test Data	
Number of Observations Read	11175
Number of Observations Used	11175

Number of Observations and Percent Classified into Species				
	Setosa	Versicolor	Virginica	Total
Total	3670 32.84	4243 37.97	3262 29.19	11175 100.00
Priors	0.33333	0.33333	0.33333	

Output 42.2.2 *continued*

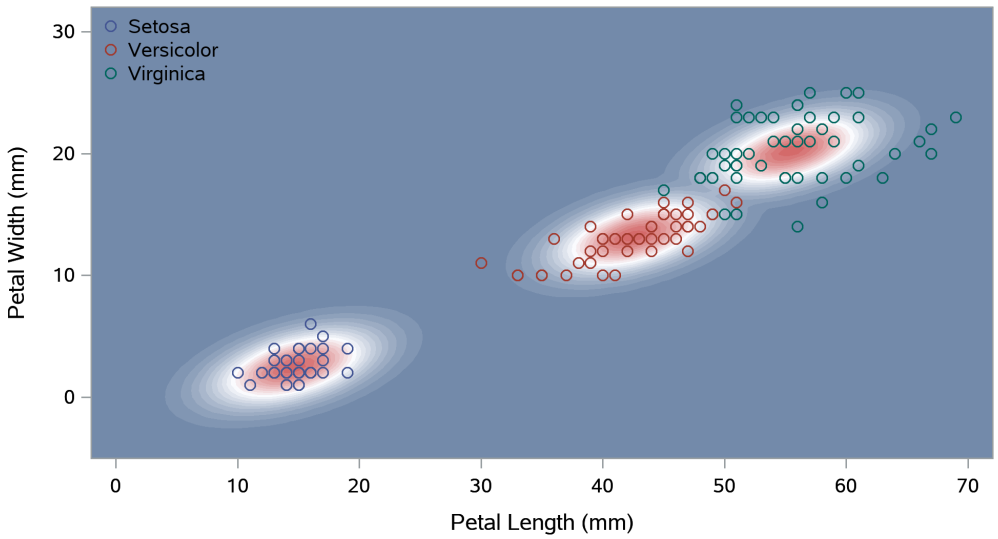

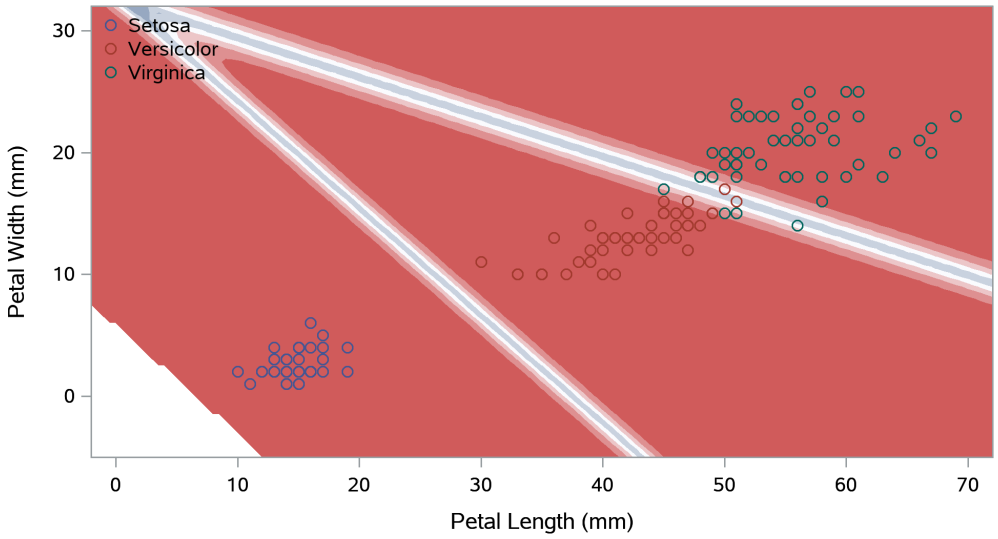

Output 42.2.2 *continued*

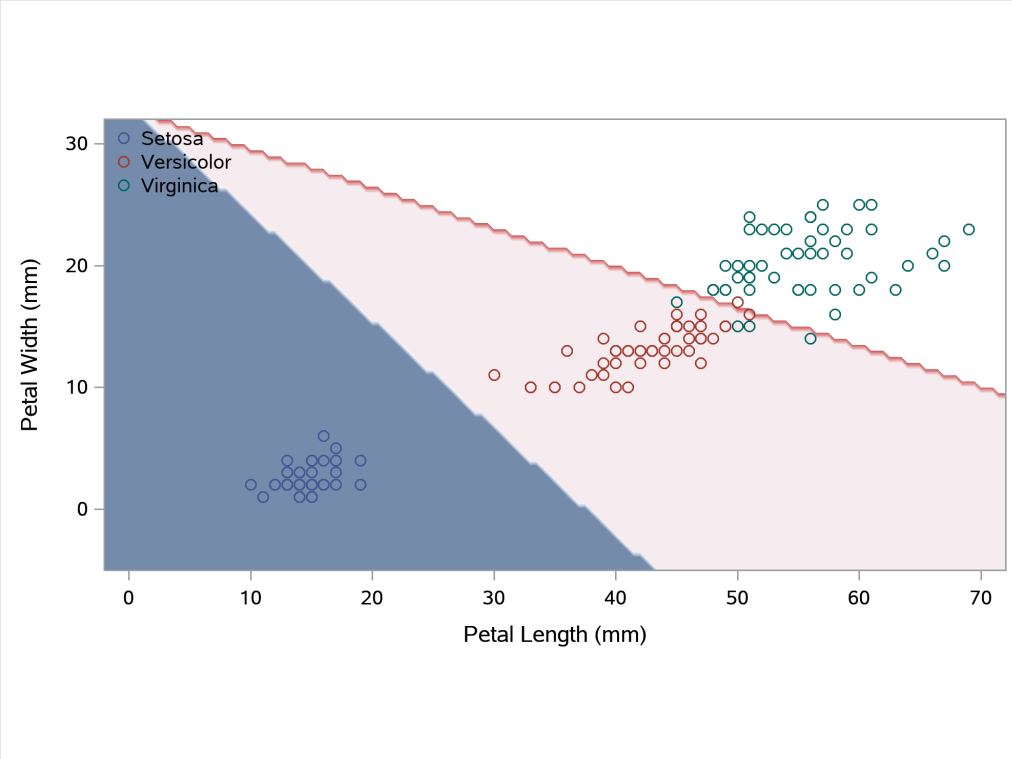

A normal-theory analysis assuming unequal covariance matrices (POOL=NO) illustrates quadratic classification boundaries. These statements produce Output 42.2.3:

```
title2 'Using Normal Density Estimates with Unequal Variance';

proc discrim data=sashelp.iris method=normal pool=no
   testdata=plotdata testout=plotp testoutd=plotd
   short noclassify crosslisterr;
   class Species;
   var Petal:;
run;

%contden
%contprob
%contclass
```

Output 42.2.3 Normal Density Estimates with Unequal Variance

Discriminant Analysis of Fisher (1936) Iris Data
Using Normal Density Estimates with Unequal Variance

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	2	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Output 42.2.3 *continued*

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Discriminant Analysis of Fisher (1936) Iris Data
Using Normal Density Estimates with Unequal Variance

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Quadratic Discriminant Function

Posterior Probability of Membership in Species

Obs	From Species	Classified into Species		Setosa	Versicolor	Virginica
53	**Versicolor**	**Virginica**	*	0.0000	0.0903	0.9097
100	**Versicolor**	**Virginica**	*	0.0000	0.4675	0.5325
103	**Virginica**	**Versicolor**	*	0.0000	0.7288	0.2712
113	**Virginica**	**Versicolor**	*	0.0000	0.5196	0.4804
136	**Virginica**	**Versicolor**	*	0.0000	0.8335	0.1665

*** Misclassified observation**

Discriminant Analysis of Fisher (1936) Iris Data
Using Normal Density Estimates with Unequal Variance

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Quadratic Discriminant Function

Number of Observations and Percent Classified into Species

From Species	Setosa	Versicolor	Virginica	Total
Setosa	50	0	0	50
	100.00	0.00	0.00	100.00
Versicolor	0	48	2	50
	0.00	96.00	4.00	100.00
Virginica	0	3	47	50
	0.00	6.00	94.00	100.00
Total	50	51	49	150
	33.33	34.00	32.67	100.00
Priors	0.33333	0.33333	0.33333	

Output 42.2.3 *continued*

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0400	0.0600	0.0333
Priors	0.3333	0.3333	0.3333	

Discriminant Analysis of Fisher (1936) Iris Data Using Normal Density Estimates with Unequal Variance

The DISCRIM Procedure
Classification Summary for Test Data: WORK.PLOTDATA
Classification Summary using Quadratic Discriminant Function

Observation Profile for Test Data	
Number of Observations Read	11175
Number of Observations Used	11175

Number of Observations and Percent Classified into Species				
	Setosa	Versicolor	Virginica	Total
Total	1382	1345	8448	11175
	12.37	12.04	75.60	100.00
Priors	0.33333	0.33333	0.33333	

Output 42.2.3 *continued*

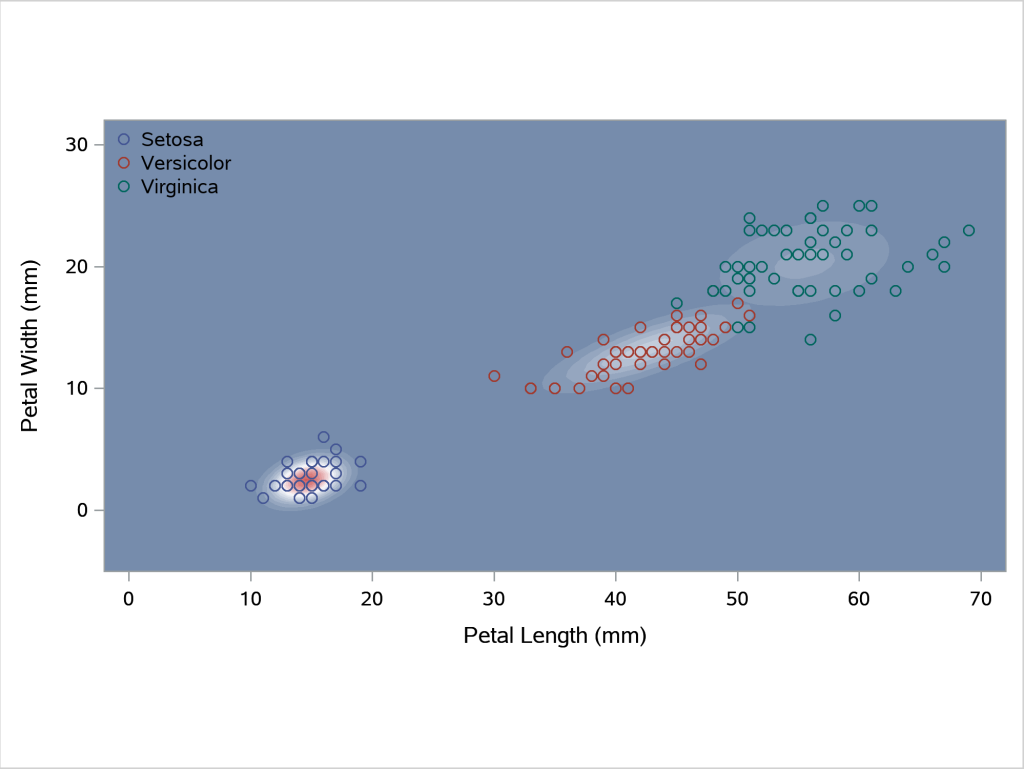

Output 42.2.3 *continued*

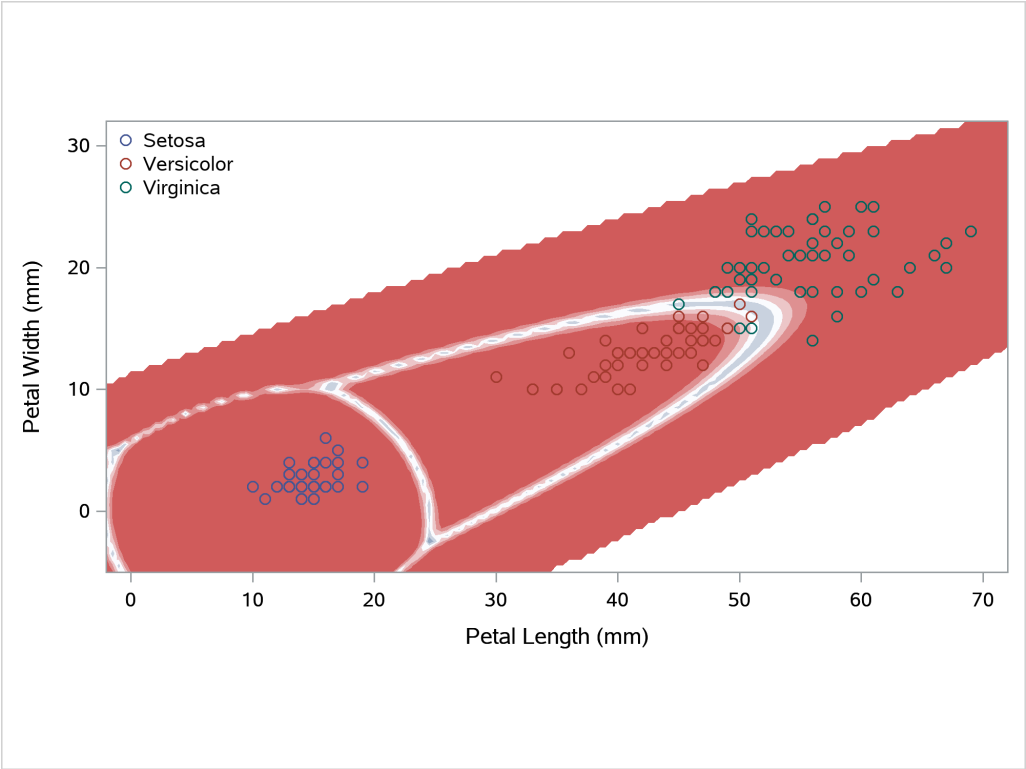

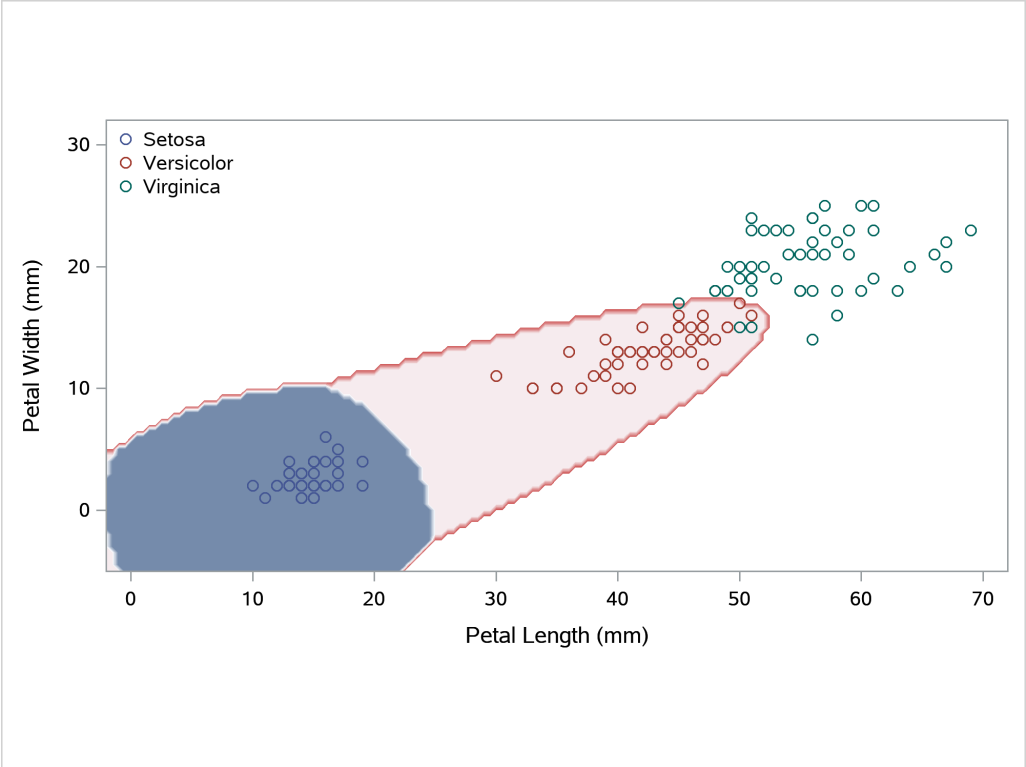

A nonparametric analysis (METHOD=NPAR) follows, using normal kernels (KERNEL=NORMAL) and

equal bandwidths (POOL=YES) in each class. The value of the radius parameter *r* that, assuming normality, minimizes an approximate mean integrated square error is 0.50 (see the section "Nonparametric Methods" on page 2740). These statements produce Output 42.2.4:

```
title2 'Using Kernel Density Estimates with Equal Bandwidth';

proc discrim data=sashelp.iris method=npar kernel=normal
   r=.5 pool=yes testoutd=plotd testdata=plotdata testout=plotp
   short noclassify crosslisterr;
   class Species;
   var Petal:;
run;

%contden
%contprob
%contclass
```

Output 42.2.4 Kernel Density Estimates with Equal Bandwidth

Discriminant Analysis of Fisher (1936) Iris Data
Using Kernel Density Estimates with Equal Bandwidth

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	2	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Output 42.2.4 *continued*

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Equal Bandwidth

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Normal Kernel Density

				Posterior Probability of Membership in Species		
Obs	From Species	Classified into Species		Setosa	Versicolor	Virginica
53	Versicolor	Virginica	*	0.0000	0.0800	0.9200
100	Versicolor	Virginica	*	0.0000	0.4123	0.5877
103	Virginica	Versicolor	*	0.0000	0.7474	0.2526
113	Virginica	Versicolor	*	0.0000	0.5863	0.4137
136	Virginica	Versicolor	*	0.0000	0.8358	0.1642

*** Misclassified observation**

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Equal Bandwidth

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Normal Kernel Density

Number of Observations and Percent Classified into Species				
From Species	Setosa	Versicolor	Virginica	Total
Setosa	50 100.00	0 0.00	0 0.00	50 100.00
Versicolor	0 0.00	48 96.00	2 4.00	50 100.00
Virginica	0 0.00	3 6.00	47 94.00	50 100.00
Total	50 33.33	51 34.00	49 32.67	150 100.00
Priors	0.33333	0.33333	0.33333	

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0400	0.0600	0.0333
Priors	0.3333	0.3333	0.3333	

Output 42.2.4 *continued*

Discriminant Analysis of Fisher (1936) Iris Data
Using Kernel Density Estimates with Equal Bandwidth

The DISCRIM Procedure
Classification Summary for Test Data: WORK.PLOTDATA
Classification Summary using Normal Kernel Density

Observation Profile for Test Data	
Number of Observations Read	11175
Number of Observations Used	11175

Number of Observations and Percent Classified into Species				
	Setosa	Versicolor	Virginica	Total
Total	3195	2492	5488	11175
	28.59	22.30	49.11	100.00
Priors	0.33333	0.33333	0.33333	

Output 42.2.4 *continued*

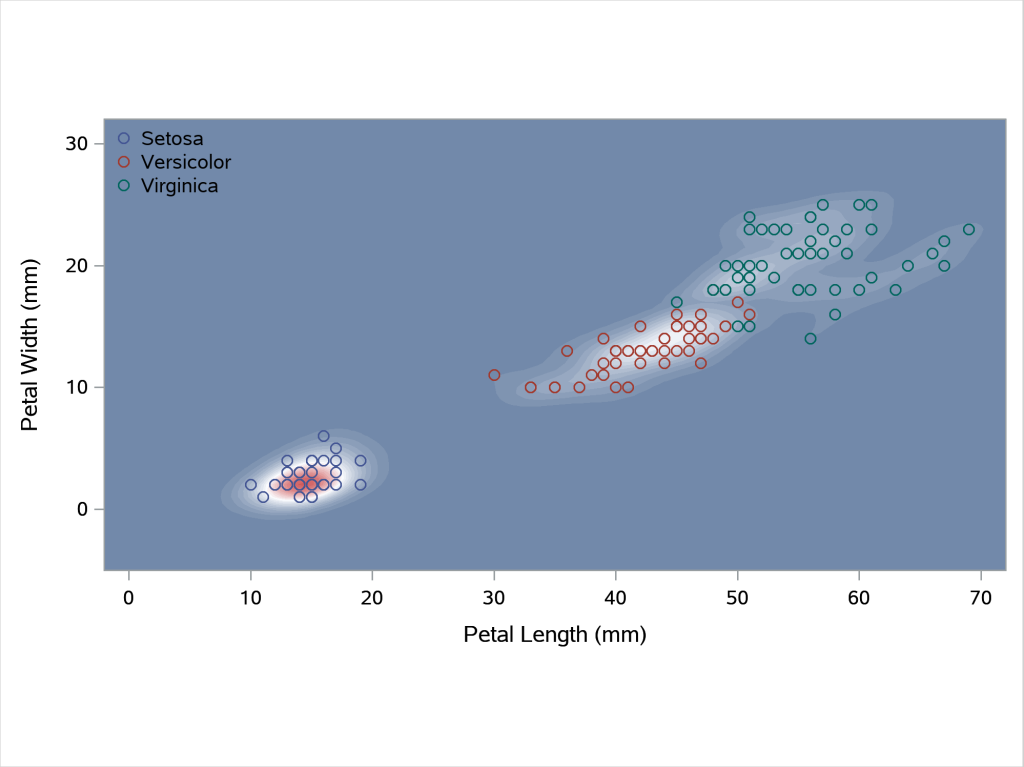

Output 42.2.4 *continued*

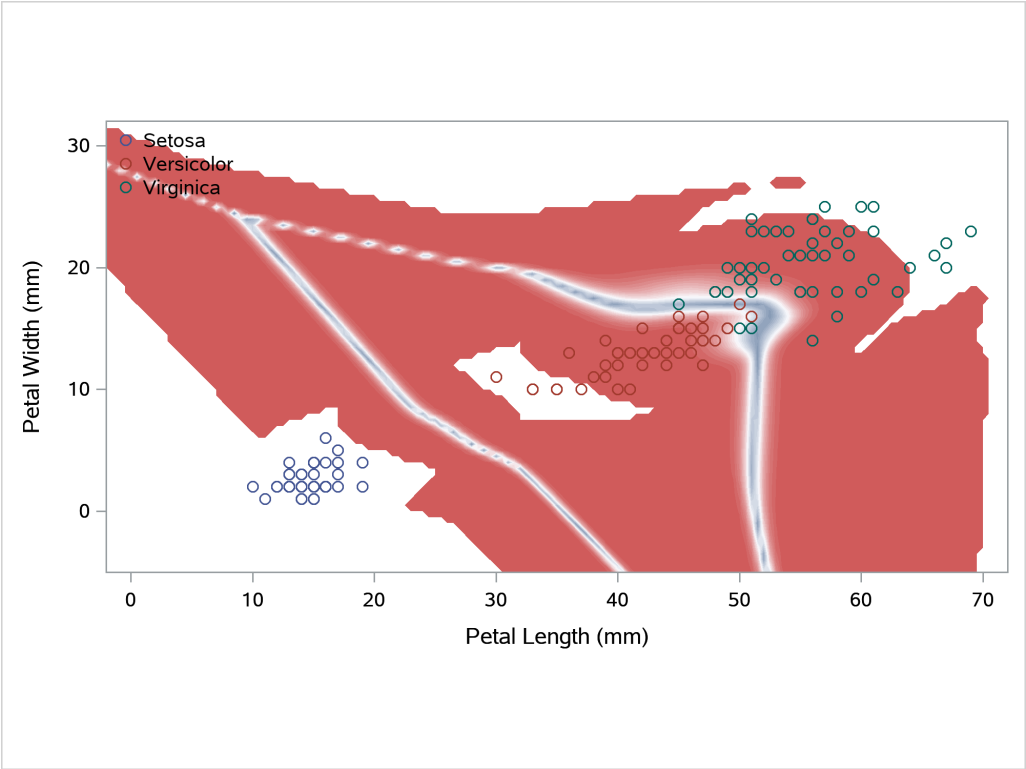

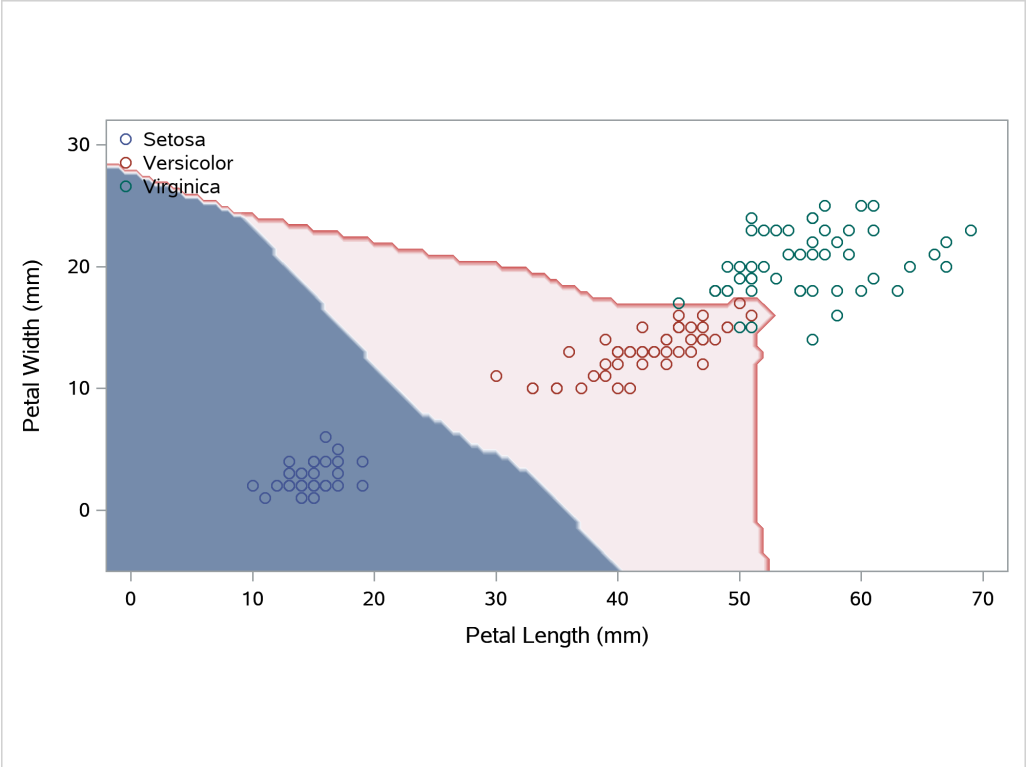

Another nonparametric analysis is run with unequal bandwidths (POOL=NO). These statements produce

Output 42.2.5:

```
title2 'Using Kernel Density Estimates with Unequal Bandwidth';

proc discrim data=sashelp.iris method=npar kernel=normal
   r=.5 pool=no testoutd=plotd testdata=plotdata testout=plotp
   short noclassify crosslisterr;
   class Species;
   var Petal:;
run;

%contden
%contprob
%contclass
```

Output 42.2.5 Kernel Density Estimates with Unequal Bandwidth

Discriminant Analysis of Fisher (1936) Iris Data
Using Kernel Density Estimates with Unequal Bandwidth

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	2	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Discriminant Analysis of Fisher (1936) Iris Data
Using Kernel Density Estimates with Unequal Bandwidth

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Normal Kernel Density

Posterior Probability of Membership in Species

Obs	From Species	Classified into Species		Setosa	Versicolor	Virginica
53	**Versicolor**	**Virginica**	*	0.0000	0.0516	0.9484
100	**Versicolor**	**Virginica**	*	0.0000	0.3773	0.6227
103	**Virginica**	**Versicolor**	*	0.0000	0.7826	0.2174
136	**Virginica**	**Versicolor**	*	0.0000	0.8802	0.1198

*** Misclassified observation**

Output 42.2.5 *continued*

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Unequal Bandwidth

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Normal Kernel Density

Number of Observations and Percent Classified into Species				
From Species	Setosa	Versicolor	Virginica	Total
Setosa	50 100.00	0 0.00	0 0.00	50 100.00
Versicolor	0 0.00	48 96.00	2 4.00	50 100.00
Virginica	0 0.00	2 4.00	48 96.00	50 100.00
Total	50 33.33	50 33.33	50 33.33	150 100.00
Priors	0.33333	0.33333	0.33333	

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0400	0.0400	0.0267
Priors	0.3333	0.3333	0.3333	

Discriminant Analysis of Fisher (1936) Iris Data Using Kernel Density Estimates with Unequal Bandwidth

The DISCRIM Procedure
Classification Summary for Test Data: WORK.PLOTDATA
Classification Summary using Normal Kernel Density

Observation Profile for Test Data	
Number of Observations Read	11175
Number of Observations Used	11175

Number of Observations and Percent Classified into Species				
	Setosa	Versicolor	Virginica	Total
Total	1370 12.26	1505 13.47	8300 74.27	11175 100.00
Priors	0.33333	0.33333	0.33333	

Output 42.2.5 *continued*

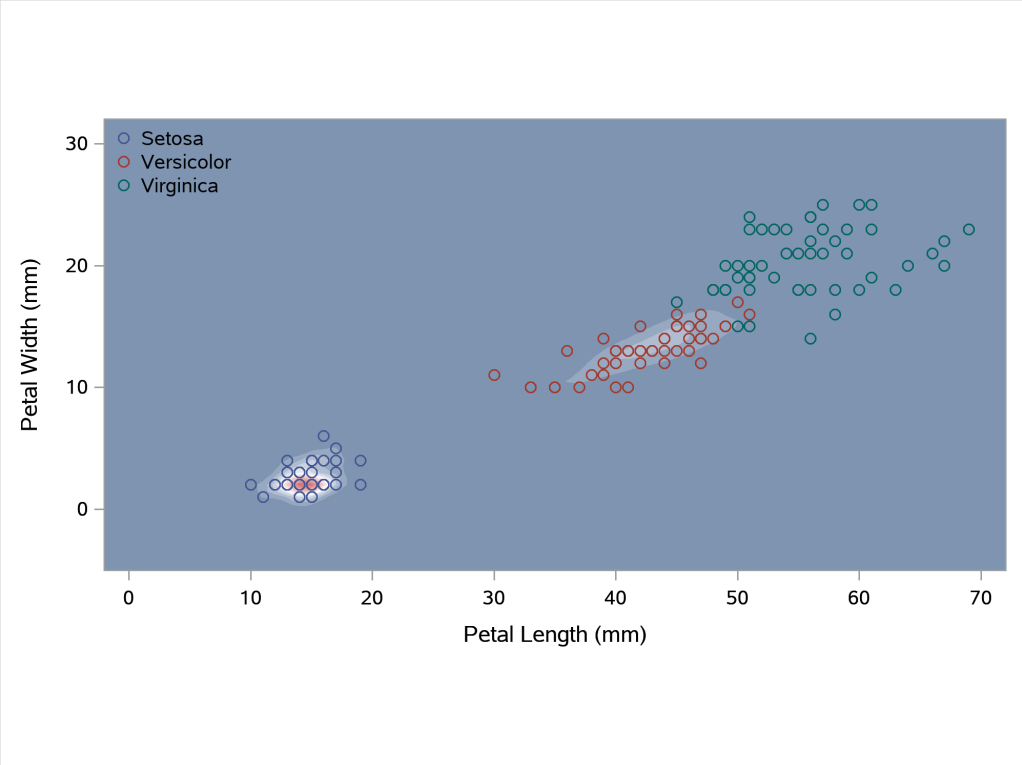

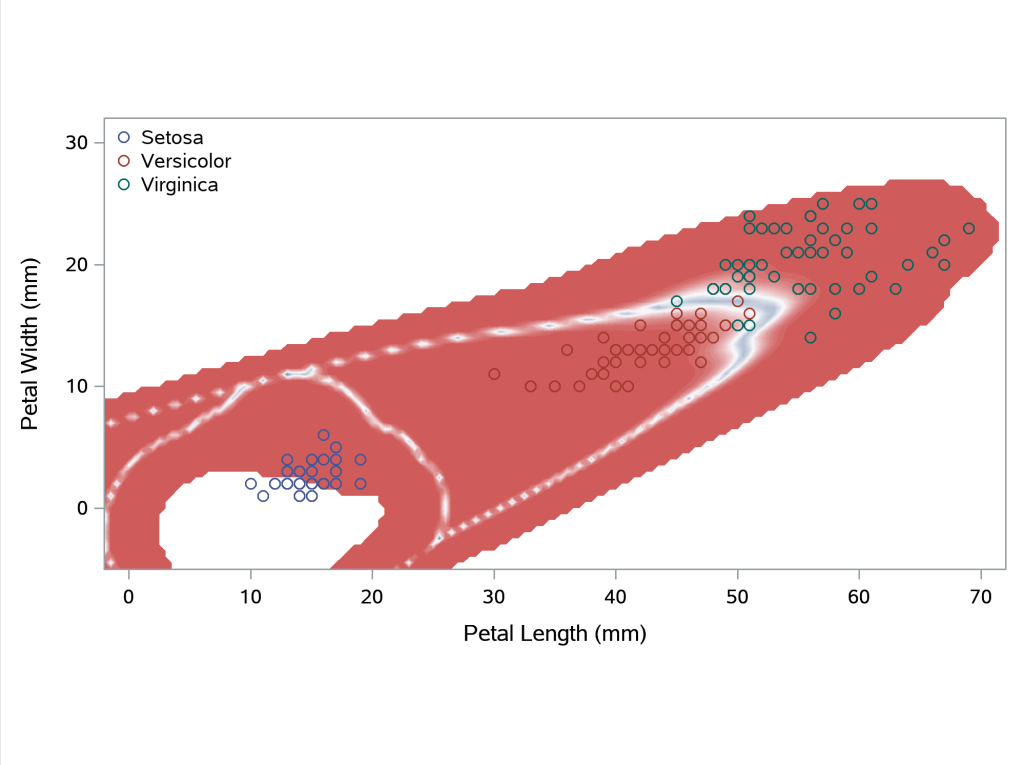

Output 42.2.5 *continued*

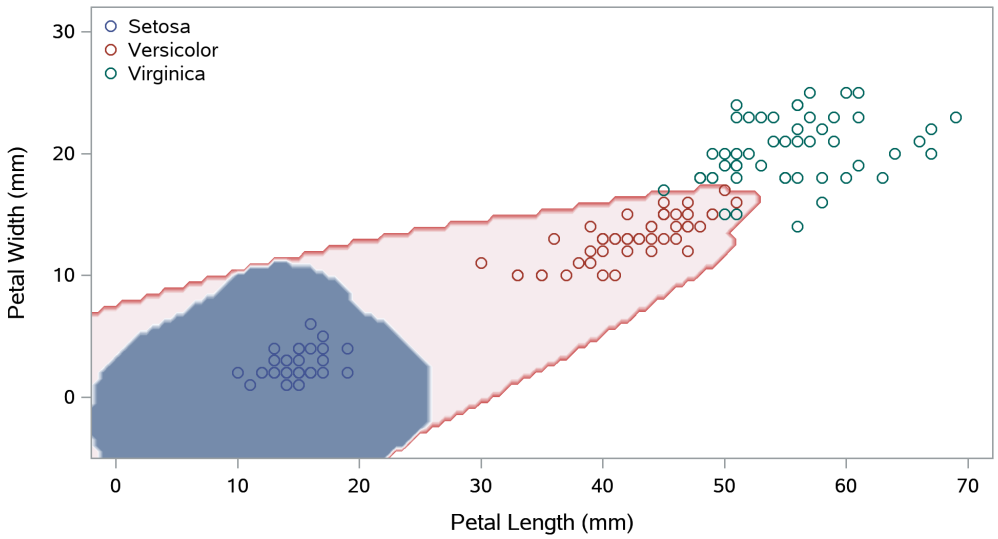

Example 42.3: Normal-Theory Discriminant Analysis of Iris Data

(View the complete code for this example (discex3.sas) in the example repository.)

In this example, PROC DISCRIM uses normal-theory methods to classify the iris data used in Example 42.1. The POOL=TEST option tests the homogeneity of the within-group covariance matrices (Output 42.3.3). Since the resulting test statistic is significant at the 0.10 level, the within-group covariance matrices are used to derive the quadratic discriminant criterion. The WCOV and PCOV options display the within-group covariance matrices and the pooled covariance matrix (Output 42.3.2). The DISTANCE option displays squared distances between classes (Output 42.3.4). The ANOVA and MANOVA options test the hypothesis that the class means are equal, by using univariate statistics and multivariate statistics; all statistics are significant at the 0.0001 level (Output 42.3.5). The LISTERR option lists the misclassified observations under resubstitution (Output 42.3.6). The CROSSLISTERR option lists the observations that are misclassified under cross validation and displays cross validation error-rate estimates (Output 42.3.7). The resubstitution error count estimate, 0.02, is not larger than the cross validation error count estimate, 0.0267, as would be expected because the resubstitution estimate is optimistically biased. The OUTSTAT= option generates a TYPE=MIXED (because POOL=TEST) output data set containing various statistics such as means, covariances, and coefficients of the discriminant function (Output 42.3.8).

The following statements produce Output 42.3.1 through Output 42.3.8:

```
title 'Discriminant Analysis of Fisher (1936) Iris Data';
title2 'Using Quadratic Discriminant Function';

proc discrim data=sashelp.iris outstat=irisstat
   wcov pcov method=normal pool=test
   distance anova manova listerr crosslisterr;
   class Species;
   var SepalLength SepalWidth PetalLength PetalWidth;
run;

proc print data=irisstat;
   title2 'Output Discriminant Statistics';
run;
```

Output 42.3.1 Quadratic Discriminant Analysis of Iris Data

Discriminant Analysis of Fisher (1936) Iris Data
Using Quadratic Discriminant Function

The DISCRIM Procedure

Total Sample Size	150	**DF Total**	149
Variables	4	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information

Species	Variable Name	Frequency	Weight	Proportion	Prior Probability
Setosa	Setosa	50	50.0000	0.333333	0.333333
Versicolor	Versicolor	50	50.0000	0.333333	0.333333
Virginica	Virginica	50	50.0000	0.333333	0.333333

Output 42.3.2 Covariance Matrices

Discriminant Analysis of Fisher (1936) Iris Data
Using Quadratic Discriminant Function

The DISCRIM Procedure
Within-Class Covariance Matrices

Species = Setosa, DF = 49

Variable	Label	SepalLength	SepalWidth	PetalLength	PetalWidth
SepalLength	Sepal Length (mm)	12.42489796	9.92163265	1.63551020	1.03306122
SepalWidth	Sepal Width (mm)	9.92163265	14.36897959	1.16979592	0.92979592
PetalLength	Petal Length (mm)	1.63551020	1.16979592	3.01591837	0.60693878
PetalWidth	Petal Width (mm)	1.03306122	0.92979592	0.60693878	1.11061224

Output 42.3.2 *continued*

Species = Versicolor, DF = 49					
Variable	**Label**	**SepalLength**	**SepalWidth**	**PetalLength**	**PetalWidth**
SepalLength	Sepal Length (mm)	26.64326531	8.51836735	18.28979592	5.57795918
SepalWidth	Sepal Width (mm)	8.51836735	9.84693878	8.26530612	4.12040816
PetalLength	Petal Length (mm)	18.28979592	8.26530612	22.08163265	7.31020408
PetalWidth	Petal Width (mm)	5.57795918	4.12040816	7.31020408	3.91061224

Species = Virginica, DF = 49					
Variable	**Label**	**SepalLength**	**SepalWidth**	**PetalLength**	**PetalWidth**
SepalLength	Sepal Length (mm)	40.43428571	9.37632653	30.32897959	4.90938776
SepalWidth	Sepal Width (mm)	9.37632653	10.40040816	7.13795918	4.76285714
PetalLength	Petal Length (mm)	30.32897959	7.13795918	30.45877551	4.88244898
PetalWidth	Petal Width (mm)	4.90938776	4.76285714	4.88244898	7.54326531

Discriminant Analysis of Fisher (1936) Iris Data Using Quadratic Discriminant Function

The DISCRIM Procedure

Pooled Within-Class Covariance Matrix, DF = 147					
Variable	**Label**	**SepalLength**	**SepalWidth**	**PetalLength**	**PetalWidth**
SepalLength	Sepal Length (mm)	26.50081633	9.27210884	16.75142857	3.84013605
SepalWidth	Sepal Width (mm)	9.27210884	11.53877551	5.52435374	3.27102041
PetalLength	Petal Length (mm)	16.75142857	5.52435374	18.51877551	4.26653061
PetalWidth	Petal Width (mm)	3.84013605	3.27102041	4.26653061	4.18816327

Output 42.3.2 *continued*

Within Covariance Matrix Information		
Species	**Covariance Matrix Rank**	**Natural Log of the Determinant of the Covariance Matrix**
Setosa	4	5.35332
Versicolor	4	7.54636
Virginica	4	9.49362
Pooled	4	8.46214

Output 42.3.3 Homogeneity Test

Discriminant Analysis of Fisher (1936) Iris Data Using Quadratic Discriminant Function

The DISCRIM Procedure
Test of Homogeneity of Within Covariance Matrices

Chi-Square	DF	Pr > ChiSq
140.943050	20	<.0001

Since the Chi-Square value is significant at the 0.1 level, the within covariance matrices will be used in the discriminant function.
Reference: Morrison, D.F. (1976) Multivariate Statistical Methods p252.

Output 42.3.4 Squared Distances

Discriminant Analysis of Fisher (1936) Iris Data Using Quadratic Discriminant Function

The DISCRIM Procedure

Squared Distance to Species			
From Species	Setosa	Versicolor	Virginica
Setosa	0	103.19382	168.76759
Versicolor	323.06203	0	13.83875
Virginica	706.08494	17.86670	0

Generalized Squared Distance to Species			
From Species	Setosa	Versicolor	Virginica
Setosa	5.35332	110.74017	178.26121
Versicolor	328.41535	7.54636	23.33238
Virginica	711.43826	25.41306	9.49362

Output 42.3.5 Tests of Equal Class Means

Discriminant Analysis of Fisher (1936) Iris Data Using Quadratic Discriminant Function

The DISCRIM Procedure

Univariate Test Statistics								
F Statistics, Num DF=2, Den DF=147								
Variable	Label	Total Standard Deviation	Pooled Standard Deviation	Between Standard Deviation	R-Square	R-Square / (1-RSq)	F Value	Pr > F
SepalLength	Sepal Length (mm)	8.2807	5.1479	7.9506	0.6187	1.6226	119.26	<.0001
SepalWidth	Sepal Width (mm)	4.3587	3.3969	3.3682	0.4008	0.6688	49.16	<.0001
PetalLength	Petal Length (mm)	17.6530	4.3033	20.9070	0.9414	16.0566	1180.16	<.0001
PetalWidth	Petal Width (mm)	7.6224	2.0465	8.9673	0.9289	13.0613	960.01	<.0001

Output 42.3.5 *continued*

Average R-Square	
Unweighted	0.7224358
Weighted by Variance	0.8689444

Multivariate Statistics and F Approximations

S=2 M=0.5 N=71

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.02343863	199.15	8	288	<.0001
Pillai's Trace	1.19189883	53.47	8	290	<.0001
Hotelling-Lawley Trace	32.47732024	582.20	8	203.4	<.0001
Roy's Greatest Root	32.19192920	1166.96	4	145	<.0001

NOTE: F Statistic for Roy's Greatest Root is an upper bound.

NOTE: F Statistic for Wilks' Lambda is exact.

Output 42.3.6 Misclassified Observations: Resubstitution

Discriminant Analysis of Fisher (1936) Iris Data Using Quadratic Discriminant Function

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Resubstitution Results using Quadratic Discriminant Function

				Posterior Probability of Membership in Species		
Obs	**From Species**	**Classified into Species**		**Setosa**	**Versicolor**	**Virginica**
53	**Versicolor**	**Virginica**	*	0.0000	0.3359	0.6641
55	**Versicolor**	**Virginica**	*	0.0000	0.1543	0.8457
103	**Virginica**	**Versicolor**	*	0.0000	0.6050	0.3950

*** Misclassified observation**

Output 42.3.6 *continued*

Discriminant Analysis of Fisher (1936) Iris Data Using Quadratic Discriminant Function

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Resubstitution Summary using Quadratic Discriminant Function

Number of Observations and Percent Classified into Species				
From Species	Setosa	Versicolor	Virginica	Total
Setosa	50	0	0	50
	100.00	0.00	0.00	100.00
Versicolor	0	48	2	50
	0.00	96.00	4.00	100.00
Virginica	0	1	49	50
	0.00	2.00	98.00	100.00
Total	50	49	51	150
	33.33	32.67	34.00	100.00
Priors	0.33333	0.33333	0.33333	

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0400	0.0200	0.0200
Priors	0.3333	0.3333	0.3333	

Output 42.3.7 Misclassified Observations: Cross Validation

Discriminant Analysis of Fisher (1936) Iris Data Using Quadratic Discriminant Function

The DISCRIM Procedure
Classification Results for Calibration Data: SASHELP.IRIS
Cross-validation Results using Quadratic Discriminant Function

Posterior Probability of Membership in Species						
Obs	From Species	Classified into Species		Setosa	Versicolor	Virginica
52	Versicolor	Virginica	*	0.0000	0.3134	0.6866
53	Versicolor	Virginica	*	0.0000	0.1616	0.8384
55	Versicolor	Virginica	*	0.0000	0.0713	0.9287
103	Virginica	Versicolor	*	0.0000	0.6632	0.3368

*** Misclassified observation**

Output 42.3.7 *continued*

Discriminant Analysis of Fisher (1936) Iris Data Using Quadratic Discriminant Function

The DISCRIM Procedure
Classification Summary for Calibration Data: SASHELP.IRIS
Cross-validation Summary using Quadratic Discriminant Function

Number of Observations and Percent Classified into Species				
From Species	Setosa	Versicolor	Virginica	Total
Setosa	50	0	0	50
	100.00	0.00	0.00	100.00
Versicolor	0	47	3	50
	0.00	94.00	6.00	100.00
Virginica	0	1	49	50
	0.00	2.00	98.00	100.00
Total	50	48	52	150
	33.33	32.00	34.67	100.00
Priors	0.33333	0.33333	0.33333	

Error Count Estimates for Species				
	Setosa	Versicolor	Virginica	Total
Rate	0.0000	0.0600	0.0200	0.0267
Priors	0.3333	0.3333	0.3333	

Output 42.3.8 Output Statistics from Iris Data

Discriminant Analysis of Fisher (1936) Iris Data
Output Discriminant Statistics

Obs	Species	_TYPE_	_NAME_	SepalLength	SepalWidth	PetalLength	PetalWidth
1		N		150.00	150.00	150.00	150.00
2	Setosa	N		50.00	50.00	50.00	50.00
3	Versicolor	N		50.00	50.00	50.00	50.00
4	Virginica	N		50.00	50.00	50.00	50.00
5		MEAN		58.43	30.57	37.58	11.99
6	Setosa	MEAN		50.06	34.28	14.62	2.46
7	Versicolor	MEAN		59.36	27.70	42.60	13.26
8	Virginica	MEAN		65.88	29.74	55.52	20.26
9	Setosa	PRIOR		0.33	0.33	0.33	0.33
10	Versicolor	PRIOR		0.33	0.33	0.33	0.33
11	Virginica	PRIOR		0.33	0.33	0.33	0.33
12	Setosa	CSSCP	SepalLength	608.82	486.16	80.14	50.62
13	Setosa	CSSCP	SepalWidth	486.16	704.08	57.32	45.56
14	Setosa	CSSCP	PetalLength	80.14	57.32	147.78	29.74
15	Setosa	CSSCP	PetalWidth	50.62	45.56	29.74	54.42
16	Versicolor	CSSCP	SepalLength	1305.52	417.40	896.20	273.32
17	Versicolor	CSSCP	SepalWidth	417.40	482.50	405.00	201.90
18	Versicolor	CSSCP	PetalLength	896.20	405.00	1082.00	358.20
19	Versicolor	CSSCP	PetalWidth	273.32	201.90	358.20	191.62
20	Virginica	CSSCP	SepalLength	1981.28	459.44	1486.12	240.56
21	Virginica	CSSCP	SepalWidth	459.44	509.62	349.76	233.38
22	Virginica	CSSCP	PetalLength	1486.12	349.76	1492.48	239.24
23	Virginica	CSSCP	PetalWidth	240.56	233.38	239.24	369.62
24		PSSCP	SepalLength	3895.62	1363.00	2462.46	564.50
25		PSSCP	SepalWidth	1363.00	1696.20	812.08	480.84
26		PSSCP	PetalLength	2462.46	812.08	2722.26	627.18
27		PSSCP	PetalWidth	564.50	480.84	627.18	615.66
28		BSSCP	SepalLength	6321.21	-1995.27	16524.84	7127.93
29		BSSCP	SepalWidth	-1995.27	1134.49	-5723.96	-2293.27
30		BSSCP	PetalLength	16524.84	-5723.96	43710.28	18677.40
31		BSSCP	PetalWidth	7127.93	-2293.27	18677.40	8041.33
32		CSSCP	SepalLength	10216.83	-632.27	18987.30	7692.43
33		CSSCP	SepalWidth	-632.27	2830.69	-4911.88	-1812.43
34		CSSCP	PetalLength	18987.30	-4911.88	46432.54	19304.58
35		CSSCP	PetalWidth	7692.43	-1812.43	19304.58	8656.99
36		RSQUARED		0.62	0.40	0.94	0.93
37	Setosa	COV	SepalLength	12.42	9.92	1.64	1.03
38	Setosa	COV	SepalWidth	9.92	14.37	1.17	0.93
39	Setosa	COV	PetalLength	1.64	1.17	3.02	0.61
40	Setosa	COV	PetalWidth	1.03	0.93	0.61	1.11
41	Versicolor	COV	SepalLength	26.64	8.52	18.29	5.58
42	Versicolor	COV	SepalWidth	8.52	9.85	8.27	4.12
43	Versicolor	COV	PetalLength	18.29	8.27	22.08	7.31
44	Versicolor	COV	PetalWidth	5.58	4.12	7.31	3.91
45	Virginica	COV	SepalLength	40.43	9.38	30.33	4.91
46	Virginica	COV	SepalWidth	9.38	10.40	7.14	4.76

Output 42.3.8 *continued*

Discriminant Analysis of Fisher (1936) Iris Data
Output Discriminant Statistics

Obs	Species	_TYPE_	_NAME_	SepalLength	SepalWidth	PetalLength	PetalWidth
47	Virginica	COV	PetalLength	30.33	7.14	30.46	4.88
48	Virginica	COV	PetalWidth	4.91	4.76	4.88	7.54
49		PCOV	SepalLength	26.50	9.27	16.75	3.84
50		PCOV	SepalWidth	9.27	11.54	5.52	3.27
51		PCOV	PetalLength	16.75	5.52	18.52	4.27
52		PCOV	PetalWidth	3.84	3.27	4.27	4.19
53		BCOV	SepalLength	63.21	-19.95	165.25	71.28
54		BCOV	SepalWidth	-19.95	11.34	-57.24	-22.93
55		BCOV	PetalLength	165.25	-57.24	437.10	186.77
56		BCOV	PetalWidth	71.28	-22.93	186.77	80.41
57		COV	SepalLength	68.57	-4.24	127.43	51.63
58		COV	SepalWidth	-4.24	19.00	-32.97	-12.16
59		COV	PetalLength	127.43	-32.97	311.63	129.56
60		COV	PetalWidth	51.63	-12.16	129.56	58.10
61	Setosa	STD		3.52	3.79	1.74	1.05
62	Versicolor	STD		5.16	3.14	4.70	1.98
63	Virginica	STD		6.36	3.22	5.52	2.75
64		PSTD		5.15	3.40	4.30	2.05
65		BSTD		7.95	3.37	20.91	8.97
66		STD		8.28	4.36	17.65	7.62
67	Setosa	CORR	SepalLength	1.00	0.74	0.27	0.28
68	Setosa	CORR	SepalWidth	0.74	1.00	0.18	0.23
69	Setosa	CORR	PetalLength	0.27	0.18	1.00	0.33
70	Setosa	CORR	PetalWidth	0.28	0.23	0.33	1.00
71	Versicolor	CORR	SepalLength	1.00	0.53	0.75	0.55
72	Versicolor	CORR	SepalWidth	0.53	1.00	0.56	0.66
73	Versicolor	CORR	PetalLength	0.75	0.56	1.00	0.79
74	Versicolor	CORR	PetalWidth	0.55	0.66	0.79	1.00
75	Virginica	CORR	SepalLength	1.00	0.46	0.86	0.28
76	Virginica	CORR	SepalWidth	0.46	1.00	0.40	0.54
77	Virginica	CORR	PetalLength	0.86	0.40	1.00	0.32
78	Virginica	CORR	PetalWidth	0.28	0.54	0.32	1.00
79		PCORR	SepalLength	1.00	0.53	0.76	0.36
80		PCORR	SepalWidth	0.53	1.00	0.38	0.47
81		PCORR	PetalLength	0.76	0.38	1.00	0.48
82		PCORR	PetalWidth	0.36	0.47	0.48	1.00
83		BCORR	SepalLength	1.00	-0.75	0.99	1.00
84		BCORR	SepalWidth	-0.75	1.00	-0.81	-0.76
85		BCORR	PetalLength	0.99	-0.81	1.00	1.00
86		BCORR	PetalWidth	1.00	-0.76	1.00	1.00
87		CORR	SepalLength	1.00	-0.12	0.87	0.82
88		CORR	SepalWidth	-0.12	1.00	-0.43	-0.37
89		CORR	PetalLength	0.87	-0.43	1.00	0.96
90		CORR	PetalWidth	0.82	-0.37	0.96	1.00
91	Setosa	STDMEAN		-1.01	0.85	-1.30	-1.25
92	Versicolor	STDMEAN		0.11	-0.66	0.28	0.17

Output 42.3.8 *continued*

Discriminant Analysis of Fisher (1936) Iris Data
Output Discriminant Statistics

Obs	Species	_TYPE_	_NAME_	SepalLength	SepalWidth	PetalLength	PetalWidth
93	Virginica	STDMEAN		0.90	-0.19	1.02	1.08
94	Setosa	PSTDMEAN		-1.63	1.09	-5.34	-4.66
95	Versicolor	PSTDMEAN		0.18	-0.85	1.17	0.62
96	Virginica	PSTDMEAN		1.45	-0.25	4.17	4.04
97		LNDETERM		8.46	8.46	8.46	8.46
98	Setosa	LNDETERM		5.35	5.35	5.35	5.35
99	Versicolor	LNDETERM		7.55	7.55	7.55	7.55
100	Virginica	LNDETERM		9.49	9.49	9.49	9.49
101	Setosa	QUAD	SepalLength	-0.09	0.06	0.02	0.02
102	Setosa	QUAD	SepalWidth	0.06	-0.08	-0.01	0.01
103	Setosa	QUAD	PetalLength	0.02	-0.01	-0.19	0.09
104	Setosa	QUAD	PetalWidth	0.02	0.01	0.09	-0.53
105	Setosa	QUAD	_LINEAR_	4.46	-0.76	3.36	-3.13
106	Setosa	QUAD	_CONST_	-121.83	-121.83	-121.83	-121.83
107	Versicolor	QUAD	SepalLength	-0.05	0.02	0.04	-0.03
108	Versicolor	QUAD	SepalWidth	0.02	-0.10	-0.01	0.10
109	Versicolor	QUAD	PetalLength	0.04	-0.01	-0.10	0.13
110	Versicolor	QUAD	PetalWidth	-0.03	0.10	0.13	-0.44
111	Versicolor	QUAD	_LINEAR_	1.80	1.60	0.33	-1.47
112	Versicolor	QUAD	_CONST_	-76.55	-76.55	-76.55	-76.55
113	Virginica	QUAD	SepalLength	-0.05	0.02	0.05	-0.01
114	Virginica	QUAD	SepalWidth	0.02	-0.08	-0.01	0.04
115	Virginica	QUAD	PetalLength	0.05	-0.01	-0.07	0.01
116	Virginica	QUAD	PetalWidth	-0.01	0.04	0.01	-0.10
117	Virginica	QUAD	_LINEAR_	0.74	1.32	0.62	0.97
118	Virginica	QUAD	_CONST_	-75.82	-75.82	-75.82	-75.82

Example 42.4: Linear Discriminant Analysis of Remote-Sensing Data on Crops

(View the complete code for this example (discex4.sas) in the example repository.)

In this example, the remote-sensing data are used. In this data set, the observations are grouped into five crops: clover, corn, cotton, soybeans, and sugar beets. Four measures called x1 through x4 make up the descriptive variables.

In the first PROC DISCRIM statement, the DISCRIM procedure uses normal-theory methods (METHOD=NORMAL) assuming equal variances (POOL=YES) in five crops. The PRIORS statement, PRIORS PROP, sets the prior probabilities proportional to the sample sizes. The LIST option lists the resubstitution classification results for each observation (Output 42.4.2). The CROSSVALIDATE option displays cross validation error-rate estimates (Output 42.4.3). The OUTSTAT= option stores the calibration information in a new data set to classify future observations. A second PROC DISCRIM statement uses this calibration information to classify a test data set. Note that the values of the identification variable,

xvalues, are obtained by rereading the x1 through x4 fields in the data lines as a single character variable. The following statements produce Output 42.4.1 through Output 42.4.3:

```
title 'Discriminant Analysis of Remote Sensing Data on Five Crops';

data crops;
   input Crop $ 1-10 x1-x4 xvalues $ 11-21;
   datalines;
Corn      16 27 31 33
Corn      15 23 30 30
Corn      16 27 27 26
Corn      18 20 25 23
Corn      15 15 31 32
Corn      15 32 32 15
Corn      12 15 16 73
Soybeans  20 23 23 25
Soybeans  24 24 25 32
Soybeans  21 25 23 24
Soybeans  27 45 24 12
Soybeans  12 13 15 42
Soybeans  22 32 31 43
Cotton    31 32 33 34
Cotton    29 24 26 28
Cotton    34 32 28 45
Cotton    26 25 23 24
Cotton    53 48 75 26
Cotton    34 35 25 78
Sugarbeets22 23 25 42
Sugarbeets25 25 24 26
Sugarbeets34 25 16 52
Sugarbeets54 23 21 54
Sugarbeets25 43 32 15
Sugarbeets26 54  2 54
Clover    12 45 32 54
Clover    24 58 25 34
Clover    87 54 61 21
Clover    51 31 31 16
Clover    96 48 54 62
Clover    31 31 11 11
Clover    56 13 13 71
Clover    32 13 27 32
Clover    36 26 54 32
Clover    53 08 06 54
Clover    32 32 62 16
;

title2 'Using the Linear Discriminant Function';

proc discrim data=crops outstat=cropstat method=normal pool=yes
   list crossvalidate;
   class Crop;
   priors prop;
   id xvalues;
   var x1-x4;
```

```
run;
```

Output 42.4.1 Linear Discriminant Function on Crop Data

Discriminant Analysis of Remote Sensing Data on Five Crops Using the Linear Discriminant Function

The DISCRIM Procedure

Total Sample Size	36	**DF Total**	35
Variables	4	**DF Within Classes**	31
Classes	5	**DF Between Classes**	4

Number of Observations Read	36
Number of Observations Used	36

Class Level Information

Crop	Variable Name	Frequency	Weight	Proportion	Prior Probability
Clover	Clover	11	11.0000	0.305556	0.305556
Corn	Corn	7	7.0000	0.194444	0.194444
Cotton	Cotton	6	6.0000	0.166667	0.166667
Soybeans	Soybeans	6	6.0000	0.166667	0.166667
Sugarbeets	Sugarbeets	6	6.0000	0.166667	0.166667

Pooled Covariance Matrix Information

Covariance Matrix Rank	Natural Log of the Determinant of the Covariance Matrix
4	21.30189

Discriminant Analysis of Remote Sensing Data on Five Crops Using the Linear Discriminant Function

The DISCRIM Procedure

Generalized Squared Distance to Crop

From Crop	Clover	Corn	Cotton	Soybeans	Sugarbeets
Clover	2.37125	7.52830	4.44969	6.16665	5.07262
Corn	6.62433	3.27522	5.46798	4.31383	6.47395
Cotton	3.23741	5.15968	3.58352	5.01819	4.87908
Soybeans	4.95438	4.00552	5.01819	3.58352	4.65998
Sugarbeets	3.86034	6.16564	4.87908	4.65998	3.58352

Linear Discriminant Function for Crop

Variable	Clover	Corn	Cotton	Soybeans	Sugarbeets
Constant	-10.98457	-7.72070	-11.46537	-7.28260	-9.80179
x1	0.08907	-0.04180	0.02462	0.0000369	0.04245
x2	0.17379	0.11970	0.17596	0.15896	0.20988
x3	0.11899	0.16511	0.15880	0.10622	0.06540
x4	0.15637	0.16768	0.18362	0.14133	0.16408

Output 42.4.2 Misclassified Observations: Resubstitution

Discriminant Analysis of Remote Sensing Data on Five Crops Using the Linear Discriminant Function

The DISCRIM Procedure
Classification Results for Calibration Data: WORK.CROPS
Resubstitution Results using Linear Discriminant Function

				Posterior Probability of Membership in Crop				
xvalues	From Crop	Classified into Crop		Clover	Corn	Cotton	Soybeans	Sugarbeets
16 27 31 33	Corn	Corn		0.0894	0.4054	0.1763	0.2392	0.0897
15 23 30 30	Corn	Corn		0.0769	0.4558	0.1421	0.2530	0.0722
16 27 27 26	Corn	Corn		0.0982	0.3422	0.1365	0.3073	0.1157
18 20 25 23	Corn	Corn		0.1052	0.3634	0.1078	0.3281	0.0955
15 15 31 32	Corn	Corn		0.0588	0.5754	0.1173	0.2087	0.0398
15 32 32 15	Corn	Soybeans	*	0.0972	0.3278	0.1318	0.3420	0.1011
12 15 16 73	Corn	Corn		0.0454	0.5238	0.1849	0.1376	0.1083
20 23 23 25	Soybeans	Soybeans		0.1330	0.2804	0.1176	0.3305	0.1385
24 24 25 32	Soybeans	Soybeans		0.1768	0.2483	0.1586	0.2660	0.1502
21 25 23 24	Soybeans	Soybeans		0.1481	0.2431	0.1200	0.3318	0.1570
27 45 24 12	Soybeans	Sugarbeets	*	0.2357	0.0547	0.1016	0.2721	0.3359
12 13 15 42	Soybeans	Corn	*	0.0549	0.4749	0.0920	0.2768	0.1013
22 32 31 43	Soybeans	Cotton	*	0.1474	0.2606	0.2624	0.1848	0.1448
31 32 33 34	Cotton	Clover	*	0.2815	0.1518	0.2377	0.1767	0.1523
29 24 26 28	Cotton	Soybeans	*	0.2521	0.1842	0.1529	0.2549	0.1559
34 32 28 45	Cotton	Clover	*	0.3125	0.1023	0.2404	0.1357	0.2091
26 25 23 24	Cotton	Soybeans	*	0.2121	0.1809	0.1245	0.3045	0.1780
53 48 75 26	Cotton	Clover	*	0.4837	0.0391	0.4384	0.0223	0.0166
34 35 25 78	Cotton	Cotton		0.2256	0.0794	0.3810	0.0592	0.2548
22 23 25 42	Sugarbeets	Corn	*	0.1421	0.3066	0.1901	0.2231	0.1381
25 25 24 26	Sugarbeets	Soybeans	*	0.1969	0.2050	0.1354	0.2960	0.1667
34 25 16 52	Sugarbeets	Sugarbeets		0.2928	0.0871	0.1665	0.1479	0.3056
54 23 21 54	Sugarbeets	Clover	*	0.6215	0.0194	0.1250	0.0496	0.1845
25 43 32 15	Sugarbeets	Soybeans	*	0.2258	0.1135	0.1646	0.2770	0.2191
26 54 2 54	Sugarbeets	Sugarbeets		0.0850	0.0081	0.0521	0.0661	0.7887
12 45 32 54	Clover	Cotton	*	0.0693	0.2663	0.3394	0.1460	0.1789
24 58 25 34	Clover	Sugarbeets	*	0.1647	0.0376	0.1680	0.1452	0.4845
87 54 61 21	Clover	Clover		0.9328	0.0003	0.0478	0.0025	0.0165
51 31 31 16	Clover	Clover		0.6642	0.0205	0.0872	0.0959	0.1322
96 48 54 62	Clover	Clover		0.9215	0.0002	0.0604	0.0007	0.0173
31 31 11 11	Clover	Sugarbeets	*	0.2525	0.0402	0.0473	0.3012	0.3588
56 13 13 71	Clover	Clover		0.6132	0.0212	0.1226	0.0408	0.2023
32 13 27 32	Clover	Clover		0.2669	0.2616	0.1512	0.2260	0.0943
36 26 54 32	Clover	Cotton	*	0.2650	0.2645	0.3495	0.0918	0.0292
53 08 06 54	Clover	Clover		0.5914	0.0237	0.0676	0.0781	0.2392
32 32 62 16	Clover	Cotton	*	0.2163	0.3180	0.3327	0.1125	0.0206

*** Misclassified observation**

Output 42.4.2 *continued*

Discriminant Analysis of Remote Sensing Data on Five Crops Using the Linear Discriminant Function

The DISCRIM Procedure
Classification Summary for Calibration Data: WORK.CROPS
Resubstitution Summary using Linear Discriminant Function

Number of Observations and Percent Classified into Crop						
From Crop	**Clover**	**Corn**	**Cotton**	**Soybeans**	**Sugarbeets**	**Total**
Clover	6	0	3	0	2	11
	54.55	0.00	27.27	0.00	18.18	100.00
Corn	0	6	0	1	0	7
	0.00	85.71	0.00	14.29	0.00	100.00
Cotton	3	0	1	2	0	6
	50.00	0.00	16.67	33.33	0.00	100.00
Soybeans	0	1	1	3	1	6
	0.00	16.67	16.67	50.00	16.67	100.00
Sugarbeets	1	1	0	2	2	6
	16.67	16.67	0.00	33.33	33.33	100.00
Total	10	8	5	8	5	36
	27.78	22.22	13.89	22.22	13.89	100.00
Priors	0.30556	0.19444	0.16667	0.16667	0.16667	

Error Count Estimates for Crop						
	Clover	**Corn**	**Cotton**	**Soybeans**	**Sugarbeets**	**Total**
Rate	0.4545	0.1429	0.8333	0.5000	0.6667	0.5000
Priors	0.3056	0.1944	0.1667	0.1667	0.1667	

Output 42.4.3 Misclassified Observations: Cross Validation

Discriminant Analysis of Remote Sensing Data on Five Crops Using the Linear Discriminant Function

The DISCRIM Procedure
Classification Summary for Calibration Data: WORK.CROPS
Cross-validation Summary using Linear Discriminant Function

Number of Observations and Percent Classified into Crop						
From Crop	Clover	Corn	Cotton	Soybeans	Sugarbeets	Total
Clover	4	3	1	0	3	11
	36.36	27.27	9.09	0.00	27.27	100.00
Corn	0	4	1	2	0	7
	0.00	57.14	14.29	28.57	0.00	100.00
Cotton	3	0	0	2	1	6
	50.00	0.00	0.00	33.33	16.67	100.00
Soybeans	0	1	1	3	1	6
	0.00	16.67	16.67	50.00	16.67	100.00
Sugarbeets	2	1	0	2	1	6
	33.33	16.67	0.00	33.33	16.67	100.00
Total	9	9	3	9	6	36
	25.00	25.00	8.33	25.00	16.67	100.00
Priors	0.30556	0.19444	0.16667	0.16667	0.16667	

Error Count Estimates for Crop						
	Clover	Corn	Cotton	Soybeans	Sugarbeets	Total
Rate	0.6364	0.4286	1.0000	0.5000	0.8333	0.6667
Priors	0.3056	0.1944	0.1667	0.1667	0.1667	

Next, you can use the calibration information stored in the Cropstat data set to classify a test data set. The TESTLIST option lists the classification results for each observation in the test data set. The following statements produce Output 42.4.4 and Output 42.4.5:

```
data test;
   input Crop $ 1-10 x1-x4 xvalues $ 11-21;
   datalines;
Corn      16 27 31 33
Soybeans  21 25 23 24
Cotton    29 24 26 28
Sugarbeets54 23 21 54
Clover    32 32 62 16
;

title2 'Classification of Test Data';

proc discrim data=cropstat testdata=test testout=tout testlist;
   class Crop;
   testid xvalues;
   var x1-x4;
run;

proc print data=tout;
```

```
   title 'Discriminant Analysis of Remote Sensing Data on Five Crops';
   title2 'Output Classification Results of Test Data';
run;
```

Output 42.4.4 Classification of Test Data

Discriminant Analysis of Remote Sensing Data on Five Crops
Classification of Test Data

The DISCRIM Procedure
Classification Results for Test Data: WORK.TEST
Classification Results using Linear Discriminant Function

			Posterior Probability of Membership in Crop				
xvalues	From Crop	Classified into Crop	Clover	Corn	Cotton	Soybeans	Sugarbeets
16 27 31 33	Corn	Corn	0.0894	0.4054	0.1763	0.2392	0.0897
21 25 23 24	Soybeans	Soybeans	0.1481	0.2431	0.1200	0.3318	0.1570
29 24 26 28	Cotton	Soybeans *	0.2521	0.1842	0.1529	0.2549	0.1559
54 23 21 54	Sugarbeets	Clover *	0.6215	0.0194	0.1250	0.0496	0.1845
32 32 62 16	Clover	Cotton *	0.2163	0.3180	0.3327	0.1125	0.0206

*** Misclassified observation**

Discriminant Analysis of Remote Sensing Data on Five Crops
Classification of Test Data

The DISCRIM Procedure
Classification Summary for Test Data: WORK.TEST
Classification Summary using Linear Discriminant Function

Observation Profile for Test Data	
Number of Observations Read	5
Number of Observations Used	5

Number of Observations and Percent Classified into Crop						
From Crop	Clover	Corn	Cotton	Soybeans	Sugarbeets	Total
Clover	0 0.00	0 0.00	1 100.00	0 0.00	0 0.00	1 100.00
Corn	0 0.00	1 100.00	0 0.00	0 0.00	0 0.00	1 100.00
Cotton	0 0.00	0 0.00	0 0.00	1 100.00	0 0.00	1 100.00
Soybeans	0 0.00	0 0.00	0 0.00	1 100.00	0 0.00	1 100.00
Sugarbeets	1 100.00	0 0.00	0 0.00	0 0.00	0 0.00	1 100.00
Total	1 20.00	1 20.00	1 20.00	2 40.00	0 0.00	5 100.00
Priors	0.30556	0.19444	0.16667	0.16667	0.16667	

Error Count Estimates for Crop						
	Clover	Corn	Cotton	Soybeans	Sugarbeets	Total
Rate	1.0000	0.0000	1.0000	0.0000	1.0000	0.6389
Priors	0.3056	0.1944	0.1667	0.1667	0.1667	

Output 42.4.5 Output Data Set of the Classification Results for Test Data

Discriminant Analysis of Remote Sensing Data on Five Crops
Output Classification Results of Test Data

Obs	Crop	x1	x2	x3	x4	xvalues	Clover	Corn	Cotton	Soybeans	Sugarbeets	_INTO_
1	Corn	16	27	31	33	16 27 31 33	0.08935	0.40543	0.17632	0.23918	0.08972	Corn
2	Soybeans	21	25	23	24	21 25 23 24	0.14811	0.24308	0.11999	0.33184	0.15698	Soybeans
3	Cotton	29	24	26	28	29 24 26 28	0.25213	0.18420	0.15294	0.25486	0.15588	Soybeans
4	Sugarbeets	54	23	21	54	54 23 21 54	0.62150	0.01937	0.12498	0.04962	0.18452	Clover
5	Clover	32	32	62	16	32 32 62 16	0.21633	0.31799	0.33266	0.11246	0.02056	Cotton

In this next example, PROC DISCRIM uses normal-theory methods (METHOD=NORMAL) assuming unequal variances (POOL=NO) for the remote-sensing data. The PRIORS statement, PRIORS PROP, sets the prior probabilities proportional to the sample sizes. The CROSSVALIDATE option displays cross validation error-rate estimates. Note that the total error count estimate by cross validation (0.5556) is much larger than the total error count estimate by resubstitution (0.1111). The following statements produce Output 42.4.6:

```
title2 'Using Quadratic Discriminant Function';

proc discrim data=crops method=normal pool=no crossvalidate;
   class Crop;
   priors prop;
   id xvalues;
   var x1-x4;
run;
```

Output 42.4.6 Quadratic Discriminant Function on Crop Data

Discriminant Analysis of Remote Sensing Data on Five Crops
Using Quadratic Discriminant Function

The DISCRIM Procedure

Total Sample Size	36	**DF Total**	35
Variables	4	**DF Within Classes**	31
Classes	5	**DF Between Classes**	4

Number of Observations Read	36
Number of Observations Used	36

Class Level Information

Crop	Variable Name	Frequency	Weight	Proportion	Prior Probability
Clover	Clover	11	11.0000	0.305556	0.305556
Corn	Corn	7	7.0000	0.194444	0.194444
Cotton	Cotton	6	6.0000	0.166667	0.166667
Soybeans	Soybeans	6	6.0000	0.166667	0.166667
Sugarbeets	Sugarbeets	6	6.0000	0.166667	0.166667

Output 42.4.6 *continued*

Within Covariance Matrix Information		
Crop	Covariance Matrix Rank	Natural Log of the Determinant of the Covariance Matrix
Clover	4	23.64618
Corn	4	11.13472
Cotton	4	13.23569
Soybeans	4	12.45263
Sugarbeets	4	17.76293

Discriminant Analysis of Remote Sensing Data on Five Crops Using Quadratic Discriminant Function

The DISCRIM Procedure

Generalized Squared Distance to Crop					
From Crop	Clover	Corn	Cotton	Soybeans	Sugarbeets
Clover	26.01743	1320	104.18297	194.10546	31.40816
Corn	27.73809	14.40994	150.50763	38.36252	25.55421
Cotton	26.38544	588.86232	16.81921	52.03266	37.15560
Soybeans	27.07134	46.42131	41.01631	16.03615	23.15920
Sugarbeets	26.80188	332.11563	43.98280	107.95676	21.34645

Discriminant Analysis of Remote Sensing Data on Five Crops Using Quadratic Discriminant Function

The DISCRIM Procedure
Classification Summary for Calibration Data: WORK.CROPS
Resubstitution Summary using Quadratic Discriminant Function

Number of Observations and Percent Classified into Crop						
From Crop	Clover	Corn	Cotton	Soybeans	Sugarbeets	Total
Clover	9 81.82	0 0.00	0 0.00	0 0.00	2 18.18	11 100.00
Corn	0 0.00	7 100.00	0 0.00	0 0.00	0 0.00	7 100.00
Cotton	0 0.00	0 0.00	6 100.00	0 0.00	0 0.00	6 100.00
Soybeans	0 0.00	0 0.00	0 0.00	6 100.00	0 0.00	6 100.00
Sugarbeets	0 0.00	0 0.00	1 16.67	1 16.67	4 66.67	6 100.00
Total	9 25.00	7 19.44	7 19.44	7 19.44	6 16.67	36 100.00
Priors	0.30556	0.19444	0.16667	0.16667	0.16667	

Error Count Estimates for Crop						
	Clover	Corn	Cotton	Soybeans	Sugarbeets	Total
Rate	0.1818	0.0000	0.0000	0.0000	0.3333	0.1111
Priors	0.3056	0.1944	0.1667	0.1667	0.1667	

Output 42.4.6 *continued*

Discriminant Analysis of Remote Sensing Data on Five Crops Using Quadratic Discriminant Function

The DISCRIM Procedure
Classification Summary for Calibration Data: WORK.CROPS
Cross-validation Summary using Quadratic Discriminant Function

Number of Observations and Percent Classified into Crop						
From Crop	**Clover**	**Corn**	**Cotton**	**Soybeans**	**Sugarbeets**	**Total**
Clover	9	0	0	0	2	11
	81.82	0.00	0.00	0.00	18.18	100.00
Corn	3	2	0	0	2	7
	42.86	28.57	0.00	0.00	28.57	100.00
Cotton	3	0	2	0	1	6
	50.00	0.00	33.33	0.00	16.67	100.00
Soybeans	3	0	0	2	1	6
	50.00	0.00	0.00	33.33	16.67	100.00
Sugarbeets	3	0	1	1	1	6
	50.00	0.00	16.67	16.67	16.67	100.00
Total	21	2	3	3	7	36
	58.33	5.56	8.33	8.33	19.44	100.00
Priors	0.30556	0.19444	0.16667	0.16667	0.16667	

Error Count Estimates for Crop						
	Clover	**Corn**	**Cotton**	**Soybeans**	**Sugarbeets**	**Total**
Rate	0.1818	0.7143	0.6667	0.6667	0.8333	0.5556
Priors	0.3056	0.1944	0.1667	0.1667	0.1667	

References

Anderson, T. W. (1984). *An Introduction to Multivariate Statistical Analysis*. 2nd ed. New York: John Wiley & Sons.

Epanechnikov, V. A. (1969). "Nonparametric Estimation of a Multivariate Probability Density." *Theory of Probability and Its Applications* 14:153–158.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Friedman, J. H., Bentley, J. L., and Finkel, R. A. (1977). "An Algorithm for Finding Best Matches in Logarithmic Expected Time." *ACM Transactions on Mathematical Software* 3:209–226.

Fukunaga, K., and Kessel, D. L. (1973). "Nonparametric Bayes Error Estimation Using Unclassified Samples." *IEEE Transactions on Information Theory* 19:434–440.

Glick, N. (1978). "Additive Estimators for Probabilities of Correct Classification." *Pattern Recognition* 10:211–222.

Hand, D. J. (1982). *Kernel Discriminant Analysis*. New York: Research Studies Press.

Hora, S. C., and Wilcox, J. B. (1982). "Estimation of Error Rates in Several-Population Discriminant Analysis." *Journal of Marketing Research* 19:57–61.

Kendall, M. G., Stuart, A., and Ord, J. K. (1983). *The Advanced Theory of Statistics.* 4th ed. Vol. 3. New York: Macmillan.

Kshirsagar, A. M. (1972). *Multivariate Analysis*. New York: Marcel Dekker.

Lachenbruch, P. A., and Mickey, M. R. (1968). "Estimation of Error Rates in Discriminant Analysis." *Technometrics* 10:1–10.

Lawley, D. N. (1959). "Tests of Significance in Canonical Analysis." *Biometrika* 46:59–66.

Morrison, D. F. (1976). *Multivariate Statistical Methods*. 2nd ed. New York: McGraw-Hill.

Parzen, E. (1962). "On Estimation of a Probability Density Function and Mode." *Annals of Mathematical Statistics* 33:1065–1076.

Perlman, M. D. (1980). "Unbiasedness of the Likelihood Ratio Tests for Equality of Several Covariance Matrices and Equality of Several Multivariate Normal Populations." *Annals of Statistics* 8:247–263.

Puranen, J. (1917). "Fish Catch data set (1917)." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Rao, C. R. (1973). *Linear Statistical Inference and Its Applications*. 2nd ed. New York: John Wiley & Sons.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge: Cambridge University Press.

Rosenblatt, M. (1956). "Remarks on Some Nonparametric Estimates of a Density Function." *Annals of Mathematical Statistics* 27:832–837.

Chapter 43
The DISTANCE Procedure

Contents

Overview: DISTANCE Procedure **2814**
- Levels of Measurement 2814
- Symmetric versus Asymmetric Nominal Variables 2815
- Standardization 2816

Getting Started: DISTANCE Procedure **2817**
- Creating a Distance Matrix as Input for a Subsequent Cluster Analysis 2817

Syntax: DISTANCE Procedure **2821**
- PROC DISTANCE Statement 2821
- VAR Statement 2828
- ID Statement 2833
- COPY Statement 2833
- BY Statement 2833
- FREQ Statement 2834
- WEIGHT Statement 2834

Details: DISTANCE Procedure **2835**
- Proximity Measures 2835
- Missing Values 2843
- Formatted versus Unformatted Values 2843
- Output Data Sets 2844

Examples: DISTANCE Procedure **2845**
- Example 43.1: Divorce Grounds—the Jaccard Coefficient 2845
- Example 43.2: Financial Data—Stock Dividends 2851

References **2857**

Overview: DISTANCE Procedure

The DISTANCE procedure computes various measures of distance, dissimilarity, or similarity between the observations (rows) of an input SAS data set, which can contain numeric or character variables, or both, depending on which proximity measure is used.

The proximity measures are stored as a lower triangular matrix or a square matrix (depending on the SHAPE= option) in an output data set that can then be used as input to the CLUSTER, MDS, and MODECLUS procedures.

The number of rows and columns in the output data set equals the number of observations in the input data set. If the input data set contains BY groups, an output matrix is computed for each BY group with the size determined by the maximum number of observations in any BY group.

The output data set is of type TYPE=DISTANCE or TYPE=SIMILAR, depending on the value of the METHOD= option. See the METHOD= option for more information about the association between the method and the output data set type.

Data set types do not persist when you copy or modify a data set. You must specify the TYPE= data set option for the new data set, as in the following example:

```
data dist2(type=distance);
   set dist;
run;
```

See the OUT= option for more information about data set type persistence.

PROC DISTANCE also provides various nonparametric and parametric methods for standardizing variables. Different variables can be standardized with different methods.

Distance matrices are used frequently in data mining, genomics, marketing, financial analysis, management science, education, chemistry, psychology, biology, and various other fields.

Levels of Measurement

Measurement of some attribute of a set of objects is the process of assigning numbers or other symbols to the objects in such a way that properties of the numbers or symbols reflect properties of the attribute being measured. There are different *levels* of measurement that involve different properties (relations and operations) of the numbers or symbols. Associated with each level of measurement is a set of transformations of the measurements that preserve the relevant properties; these transformations are called *permissible* transformations. A particular way of assigning numbers or symbols to measure something is called a *scale* of measurement.

The most commonly discussed levels of measurement are as follows:

Nominal
: Two objects are assigned the same symbol if they have the same value of the attribute. Permissible transformations are any one-to-one or many-to-one transformation, although a many-to-one transformation loses information.

Ordinal
: Objects are assigned numbers such that the order of the numbers reflects an order relation defined on the attribute. Two objects x and y with attribute values $a(x)$ and $a(y)$

are assigned numbers $m(x)$ and $m(y)$ such that if $m(x) > m(y)$, then $a(x) > a(y)$. Permissible transformations are any monotone increasing transformation, although a transformation that is not strictly increasing loses information.

Interval — Objects are assigned numbers such that differences between the numbers reflect differences of the attribute. If $m(x) - m(y) > m(u) - m(v)$, then $a(x) - a(y) > a(u) - a(v)$. Permissible transformations are any affine transformation $t(m) = c * m + d$, where c and d are constants; another way of saying this is that the origin and unit of measurement are arbitrary.

Log-interval — Objects are assigned numbers such that ratios between the numbers reflect ratios of the attribute. If $m(x)/m(y) > m(u)/m(v)$, then $a(x)/a(y) > a(u)/a(v)$. Permissible transformations are any power transformation $t(m) = c * m^d$, where c and d are constants.

Ratio — Objects are assigned numbers such that differences and ratios between the numbers reflect differences and ratios of the attribute. Permissible transformations are any linear (similarity) transformation $t(m) = c * m$, where c is a constant; another way of saying this is that the unit of measurement is arbitrary.

Absolute — Objects are assigned numbers such that all properties of the numbers reflect analogous properties of the attribute. The only permissible transformation is the identity transformation.

Proximity measures provided in the DISTANCE procedure accept four levels of measurement: nominal, ordinal, interval, and ratio. Ordinal variables are transformed to interval variables before processing. This is done by replacing the data with their rank scores, and by assuming that the classes of an ordinal variable are spaced equally along the interval scale. See the RANKSCORE= option in the section "PROC DISTANCE Statement" on page 2821 for choices on assigning scores to ordinal variables. There are also different approaches for how to transform an ordinal variable to an interval variable. See Anderberg (1973) for alternatives.

Symmetric versus Asymmetric Nominal Variables

A binary variable contains two possible outcomes: 1 (positive/present) or 0 (negative/absent). If there is no preference for which outcome should be coded as 0 and which as 1, the binary variable is called *symmetric*. For example, the binary variable "is evergreen?" for a plant has the possible states "loses leaves in winter" and "does not lose leaves in winter." Both are equally valuable and carry the same weight when a proximity measure is computed. Commonly used measures that accept symmetric binary variables include the Simple Matching, Hamann, Roger and Tanimoto, Sokal and Sneath 1, and Sokal and Sneath 3 coefficients.

If the outcomes of a binary variable are not equally important, the binary variable is called *asymmetric*. An example of such a variable is the presence or absence of a relatively rare attribute, such as "is color-blind" for a human being. While you say that two people who are color-blind have something in common, you cannot say that people who are not color-blind have something in common. The most important outcome is usually coded as 1 (present) and the other is coded as 0 (absent). The agreement of two 1's (a present-present match or a positive match) is more significant than the agreement of two 0's (an absent-absent match or a negative match). Usually, the negative match is treated as irrelevant. Commonly used measures that accept asymmetric binary variables include Jaccard, Dice, Russell and Rao, Binary Lance and Williams nonmetric, and Kulcynski coefficients.

When nominal variables are employed, the comparison of one data unit with another can only be in terms of whether the data units score the same or different on the variables. If a variable is defined as an asymmetric nominal variable and two data units score the same but fall into the absent category, the absent-absent match is excluded from the computation of the proximity measure.

Standardization

Since variables with large variances tend to have more effect on the proximity measure than those with small variances, it is recommended that you standardize the variables before the computation of the proximity measure. The DISTANCE procedure provides a convenient way to standardize each variable with its own method before the proximity measures are computed. You can also perform the standardization by using the STDIZE procedure, with the limitation that all variables must be standardized with the same method.

Mandatory Standardization

Variable standardization is not required if any of the following conditions is true:

- if there is only one level of measurement
- if only asymmetric nominal and nominal levels are specified
- if the NOSTD option is specified in the PROC DISTANCE statement
- if METHOD=JACCARD or METHOD=DJACCARD is specified

Otherwise, standardization is mandatory.

When standardization is mandatory and no standardization method is specified, a default method of standardization will be used. This default method is determined by the measurement level. In general, the default method is STD for interval variables except when METHOD=GOWER or METHOD=DGOWER is specified. See the STD= option in the section "VAR Statement" on page 2828 for the default methods for GOWER and DGOWER as well as methods available for standardizing variables.

When standardization is mandatory, PROC DISTANCE ignores the REPONLY option, if it is specified.

Getting Started: DISTANCE Procedure

Creating a Distance Matrix as Input for a Subsequent Cluster Analysis

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/distangs.sas.)

The following example demonstrates how you can use the DISTANCE procedure to obtain a distance matrix that will be used as input to a subsequent clustering procedure.

The following data, originated by A. Weber and cited in Hand et al. (1994, p. 297), measure the amount of protein consumed for nine food groups in 25 European countries. The nine food groups are red meat (RedMeat), white meat (WhiteMeat), eggs (Eggs), milk (Milk), fish (Fish), cereal (Cereal), starch (Starch), nuts (Nuts), and fruits and vegetables (FruitVeg). Suppose you want to determine whether national figures in protein consumption can be used to determine certain types or categories of countries; specifically, you want to perform a cluster analysis to determine whether these 25 countries can be formed into groups suggested by the data.

The following DATA step creates the SAS data set Protein:

```
data Protein;
   length Country $ 14;
   input Country &$ RedMeat WhiteMeat Eggs Milk
                     Fish Cereal Starch Nuts FruitVeg;
   datalines;
Albania            10.1  1.4  0.5   8.9  0.2  42.3  0.6  5.5  1.7
Austria             8.9 14.0  4.3  19.9  2.1  28.0  3.6  1.3  4.3
Belgium            13.5  9.3  4.1  17.5  4.5  26.6  5.7  2.1  4.0
Bulgaria            7.8  6.0  1.6   8.3  1.2  56.7  1.1  3.7  4.2
Czechoslovakia      9.7 11.4  2.8  12.5  2.0  34.3  5.0  1.1  4.0
Denmark            10.6 10.8  3.7  25.0  9.9  21.9  4.8  0.7  2.4
E Germany           8.4 11.6  3.7  11.1  5.4  24.6  6.5  0.8  3.6
Finland             9.5  4.9  2.7  33.7  5.8  26.3  5.1  1.0  1.4
France             18.0  9.9  3.3  19.5  5.7  28.1  4.8  2.4  6.5
Greece             10.2  3.0  2.8  17.6  5.9  41.7  2.2  7.8  6.5
Hungary             5.3 12.4  2.9   9.7  0.3  40.1  4.0  5.4  4.2
Ireland            13.9 10.0  4.7  25.8  2.2  24.0  6.2  1.6  2.9
Italy               9.0  5.1  2.9  13.7  3.4  36.8  2.1  4.3  6.7
Netherlands         9.5 13.6  3.6  23.4  2.5  22.4  4.2  1.8  3.7
Norway              9.4  4.7  2.7  23.3  9.7  23.0  4.6  1.6  2.7
Poland              6.9 10.2  2.7  19.3  3.0  36.1  5.9  2.0  6.6
Portugal            6.2  3.7  1.1   4.9 14.2  27.0  5.9  4.7  7.9
Romania             6.2  6.3  1.5  11.1  1.0  49.6  3.1  5.3  2.8
Spain               7.1  3.4  3.1   8.6  7.0  29.2  5.7  5.9  7.2
Sweden              9.9  7.8  3.5   4.7  7.5  19.5  3.7  1.4  2.0
Switzerland        13.1 10.1  3.1  23.8  2.3  25.6  2.8  2.4  4.9
UK                 17.4  5.7  4.7  20.6  4.3  24.3  4.7  3.4  3.3
USSR                9.3  4.6  2.1  16.6  3.0  43.6  6.4  3.4  2.9
W Germany          11.4 12.5  4.1  18.8  3.4  18.6  5.2  1.5  3.8
Yugoslavia          4.4  5.0  1.2   9.5  0.6  55.9  3.0  5.7  3.2
```

```
;
```

The data set Protein contains the character variable Country and the nine numeric variables representing the food groups. The $14. in the INPUT statement specifies that the variable Country has a length of 14.

The following statements create the distance matrix and display part of it:

```
title 'Protein Consumption in Europe';
proc distance data=Protein out=Dist method=Euclid;
   var interval(RedMeat--FruitVeg / std=Std);
   id Country;
run;

proc print data=Dist(obs=10);
   title2 'First 10 Observations in Output Data Set from PROC DISTANCE';
run;
title2;
```

An output SAS data set called Dist, which contains the distance matrix, is created through the OUT= option. The METHOD=EUCLID option requests that Euclidean distances (which is the default) should be computed and produces an output data set of TYPE=DISTANCE.[1]

The VAR statement lists the variables (RedMeat—FruitVeg) along with their measurement level to be used in the analysis. An interval level of measurement is assigned to those variables. Since variables with large variances tend to have more effect on the proximity measure than those with small variances, each variable is standardized by the STD method to have a mean of 0 and a standard deviation of 1. This is done by adding "/ STD=STD" at the end of the variables list.

The ID statement specifies that the variable Country should be copied to the OUT= data set and used to generate names for the distance variables. The distance variables in the output data set are named by the values in the ID variable, and the maximum length for the names of these variables is 14.

There are 25 observations in the input data set; therefore, the output data set Dist contains a 25-by-25 lower triangular matrix.

The PROC PRINT statement displays the first 10 observations in the output data set Dist as shown in Figure 43.1.

[1] Data set types do not persist when you copy or modify a data set. You must specify the TYPE= data set option for the new data set. See the METHOD= and OUT= options for more information about data set types.

Figure 43.1 First 10 Observations in the Output Data Set from PROC DISTANCE

Protein Consumption in Europe
First 10 Observations in Output Data Set from PROC DISTANCE

Obs	Country	Albania	Austria	Belgium	Bulgaria	Czechoslovakia	Denmark	E_Germany	Finland
1	Albania	0.00000	.	.	.	.	.	.	.
2	Austria	6.12388	0.00000	.	.	.	.	.	.
3	Belgium	5.94109	2.44987	0.00000	.	.	.	.	.
4	Bulgaria	2.76446	4.88331	5.22711	0.00000	.	.	.	.
5	Czechoslovakia	5.13959	2.11498	2.21330	3.94761	0.00000	.	.	.
6	Denmark	6.61002	3.01392	2.52541	6.00803	3.34049	0.00000	.	.
7	E Germany	6.39178	2.56341	2.10211	5.40824	1.87962	2.72112	0.00000	.
8	Finland	5.81458	4.04271	3.45779	5.74882	3.91378	2.61570	3.99426	0.00000
9	France	6.29601	3.58891	2.19329	5.54675	3.36011	3.65772	3.78184	4.56796
10	Greece	4.24495	5.16330	4.69515	3.74849	4.86684	5.59084	5.61496	5.47453

Obs	France	Greece	Hungary	Ireland	Italy	Netherlands	Norway	Poland	Portugal	Romania	Spain
1	.	.	.	.	.	.	.	.	.	.	.
2	.	.	.	.	.	.	.	.	.	.	.
3	.	.	.	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.	.	.	.	.
6	.	.	.	.	.	.	.	.	.	.	.
7	.	.	.	.	.	.	.	.	.	.	.
8	.	.	.	.	.	.	.	.	.	.	.
9	0.00000	.	.	.	.	.	.	.	.	.	.
10	4.54456	0	.	.	.	.	.	.	.	.	.

Obs	Sweden	Switzerland	UK	USSR	W_Germany	Yugoslavia
1	.	.	.	.	.	.
2	.	.	.	.	.	.
3	.	.	.	.	.	.
4	.	.	.	.	.	.
5	.	.	.	.	.	.
6	.	.	.	.	.	.
7	.	.	.	.	.	.
8	.	.	.	.	.	.
9	.	.	.	.	.	.
10	.	.	.	.	.	.

The following statements produce the dendrogram in Figure 43.2:

```
ods graphics on;

proc cluster data=Dist method=Ward plots=dendrogram(height=rsq);
   id Country;
run;
```

The CLUSTER procedure performs a Ward's minimum-variance cluster analysis based on the distance matrix created by PROC DISTANCE. PROC CLUSTER, along with ODS Graphics, produces the dendrogram

shown in Figure 43.2. The option PLOTS=DENDROGRAM(HEIGHT=RSQ) specifies the squared multiple correlation as the height variable in the dendrogram.

Figure 43.2 Dendrogram of R Square

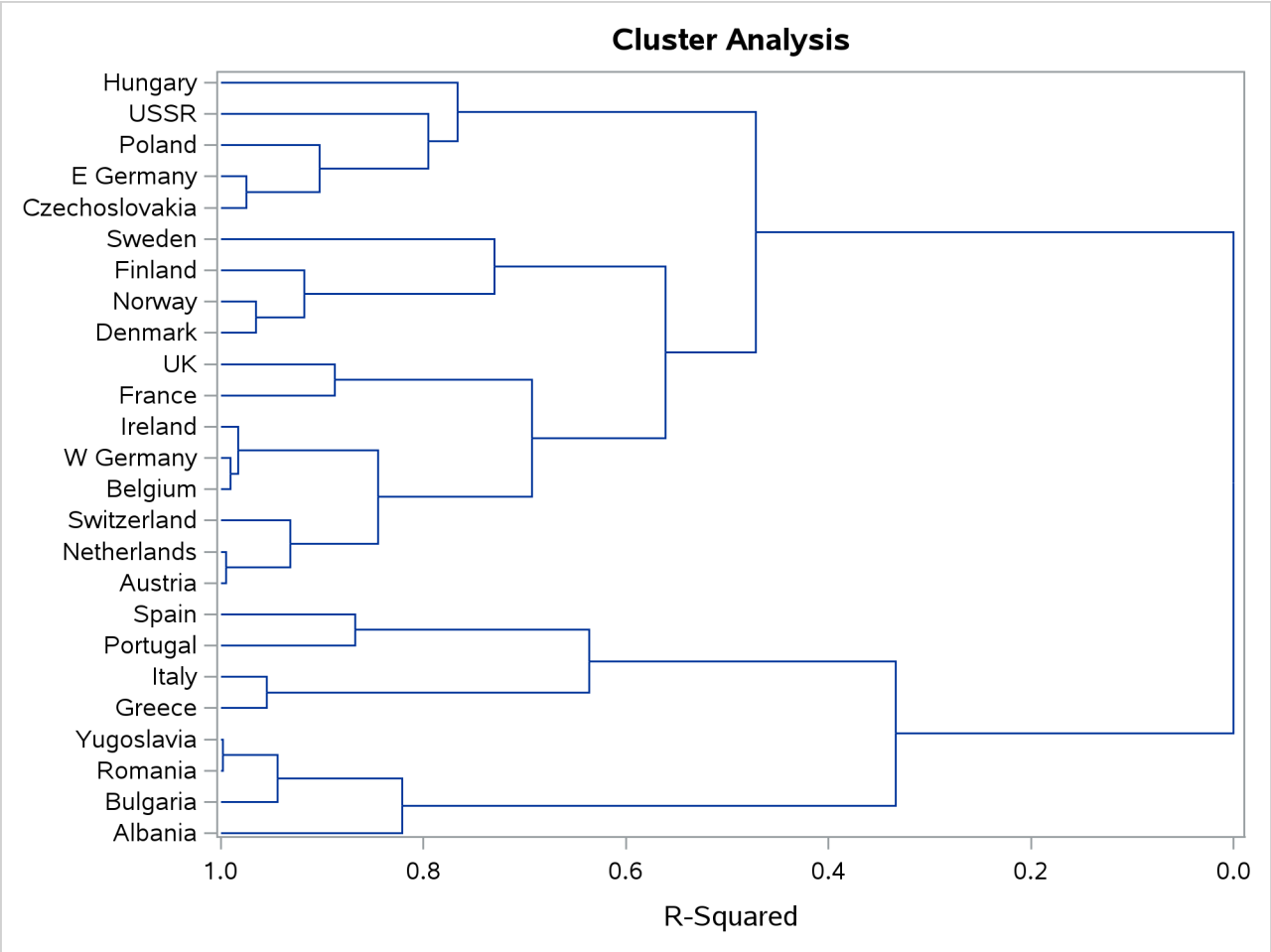

After inspecting the dendrogram in Figure 43.2, you will see that when the countries are grouped into six clusters, the proportion of variance accounted for by these clusters is slightly less than 70% (69.3%). The 25 countries are clustered as follows:

- Balkan countries: Albania, Bulgaria, Romania, and Yugoslavia
- Mediterranean countries: Greece and Italy
- Iberian countries: Portugal and Spain
- Western European countries: Austria, Netherlands, Switzerland, Belgium, former West Germany, Ireland, France, and U.K.
- Scandinavian countries: Denmark, Norway, Finland, and Sweden
- Eastern European countries: former Czechoslovakia, former East Germany, Poland, former U.S.S.R., and Hungary

Syntax: DISTANCE Procedure

The following statements are available in the DISTANCE procedure:

```
PROC DISTANCE <options> ;
    BY variables ;
    COPY variables ;
    FREQ variable ;
    ID variable ;
    VAR level (variables </ options>) ;
    WEIGHT variable ;
```

Both the PROC DISTANCE statement and the VAR statement are required.

PROC DISTANCE Statement

```
PROC DISTANCE <options> ;
```

The PROC DISTANCE statement invokes the DISTANCE procedure. Table 43.1 summarizes the *options* available in the PROC DISTANCE statement. These *options* are discussed in the following section.

Table 43.1 Summary of PROC DISTANCE Statement Options

Option	Description
Standardize Variables	
ADD=	Specifies the constant to add to each value after standardizing and multiplying by the value specified in the MULT= option
FUZZ=	Specifies the relative fuzz factor for writing the output
INITIAL=	Specifies the method for computing initial estimates for the A-estimates
MULT=	Specifies the constant to multiply each value by after standardizing
NORM	Normalizes the scale estimator to be consistent for the standard deviation of a normal distribution
NOSTD	Suppresses standardization
SNORM	Normalizes the scale estimator to have an expectation of approximately 1 for a standard normal distribution
STDONLY	Standardizes variables only (suppresses computation of the distance matrix)
VARDEF=	Specifies the variances divisor
Generate Distance Matrix	
ABSENT=	Specifies the value to be used as an absence value for all the asymmetric nominal variables
METHOD=	Specifies the method for computing proximity measures
PREFIX=	Specifies a prefix for naming the distance variables in the OUT= data set
RANKSCORE=	Specifies the method of assigning scores to ordinal variables
SHAPE=	Specifies the shape of the proximity matrix to be stored in the OUT= data set
UNDEF=	Specifies the numeric constant used to replace undefined distances

Table 43.1 continued

Option	Description
Replace Missing Values	
NOMISS	Omits observations with missing values from computation of the location and scale measures, if standardization applies; outputs missing values to the distance matrix for observations with missing values
REPLACE	Replaces missing data with zero in the standardized data
REPONLY	Replaces missing data with the location measure (does not standardize the data)
Specify Data Set Details	
DATA=	Specifies the input data set
OUT=	Specifies the output data set
OUTSDZ=	Specifies the output data set for standardized scores

These *options* and their abbreviations are described (in alphabetical order) in the remainder of this section.

ABSENT=*number* | *qs*

specifies the value to be used as an absence value in an irrelevant absent-absent match for *all* of the asymmetric nominal variables. If you want to specify a different absence value for a particular variable, use the ABSENT= option in the VAR statement. See the ABSENT= option in the section "VAR Statement" on page 2828 for details.

An absence value for a variable can be either a numeric value or a quoted string consisting of combinations of characters. For instance, ., -999, and "NA" are legal values for the ABSENT= option.

The default absence value for a character variable is "NONE" (notice that a blank value is considered a missing value), and the default absence value for a numeric variable is 0.

ADD=*c*

specifies a constant, *c*, to add to each value after standardizing and multiplying by the value you specify in the MULT= option. The default value is 0.

DATA=*SAS-data-set*

specifies the input data set containing observations from which the proximity is computed. If you omit the DATA= option, the most recently created SAS data set is used.

FUZZ=*c*

specifies the relative fuzz factor for computing the standardized scores. The default value is 1E–14. For the OUTSDZ= data set, the score is computed as follows:

$$\text{if } |\text{standardized scores}| < m \times c, \text{ then standardized scores} = 0$$

where m is the numeric constant specified in the MULT= option, or 1 if MULT= option is not specified.

INITIAL=*method*

specifies the method of computing initial estimates for the A-estimates (ABW, AWAVE, and AHUBER). The following methods are not allowed for the INITIAL= option: ABW, AHUBER, AWAVE, and IN.

The default value is INITIAL=MAD.

METHOD=*method*

specifies the method of computing proximity measures.

For use in PROC CLUSTER, you should choose distance or dissimilarity measures such as METHOD=EUCLID or METHOD=DGOWER.

The following six tables outline the proximity measures available for the METHOD= option. These tables are classified by levels of measurement accepted by each method. Each table contains four or five columns: the Method column shows the proximity measures, one or two Range columns show the upper and lower bounds, and the TYPE= column shows the type of proximity. The TYPE= column contains SIMILAR if a method generates similarity measures or DISTANCE if a method generates distance or dissimilarity measures. The output data set is of the type shown. For more information about the output data set, see the OUT= option.

For formulas and descriptions of these methods, see the section "Details: DISTANCE Procedure" on page 2835.

Table 43.2 shows the range and output matrix type of the GOWER and DGOWER methods. These two methods accept all measurement levels, including ratio, interval, ordinal, nominal, and asymmetric nominal. METHOD=GOWER or METHOD=DGOWER always implies standardization. Assuming that all the numeric (ordinal, interval, and ratio) variables are standardized by their corresponding default methods, the possible range values for both methods are from 0 to 1, inclusive. For more information about the default methods of standardization for METHOD=GOWER or METHOD=DGOWER, see the STD= option in the section "VAR Statement" on page 2828.

Table 43.2 Methods That Accept All Measurement Levels

Method	Description	Range	TYPE=
GOWER	Gower and Legendre (1986) similarity	0 to 1	**SIMILAR**
DGOWER	1 minus GOWER	0 to 1	**DISTANCE**

Table 43.3 shows *methods* that accept ratio, interval, and ordinal variables. Similar methods are grouped together in the table.

Table 43.3 Methods That Accept Ratio, Interval, and Ordinal Variables

Method	Description	Range	TYPE=
EUCLID	Euclidean distance	≥ 0	**DISTANCE**
SQEUCLID	Squared Euclidean distance	≥ 0	**DISTANCE**
SIZE	Size distance	≥ 0	**DISTANCE**
SHAPE	Shape distance	≥ 0	**DISTANCE**
COV	Covariance	≥ 0	**SIMILAR**

Table 43.3 *continued*

Method	Description	Range	TYPE=
CORR	Correlation	-1 to 1	**SIMILAR**
DCORR	Correlation transformed to Euclidean distance	0 to $\sqrt{2}$	**DISTANCE**
SQCORR	Squared correlation	0 to 1	**SIMILAR**
DSQCORR	One minus squared correlation	0 to 1	**DISTANCE**
L(p)	Minkowski (L_p) distance, where p is a positive numeric value. Typical values of p include 1 and 2. Very small or large values of p might cause floating-point overflow.	> 0	**DISTANCE**
CITYBLOCK	L_1, city-block, or Manhattan distance	≥ 0	**DISTANCE**
CHEBYCHEV	L_∞	≥ 0	**DISTANCE**
POWER(p, r)	Generalized Euclidean distance, where p is a positive numeric value and r is a nonnegative numeric value. The distance between two observations is the rth root of sum of the absolute differences to the pth power between the values for the observations.	≥ 0	**DISTANCE**

Table 43.4 shows *methods* that accept ratio variables. Notice that all possible range values are nonnegative, because ratio variables are assumed to be positive. Similar methods are grouped together in the table.

Table 43.4 Methods That Accept Ratio Variables

Method	Description	Range	TYPE=
SIMRATIO	Similarity ratio (if variables are binary, this is the Jaccard coefficient)	0 to 1	**SIMILAR**
DISRATIO	One minus similarity ratio	0 to 1	**DISTANCE**
NONMETRIC	Lance and Williams nonmetric coefficient	0 to 1	**DISTANCE**
CANBERRA	Canberra metric distance	≥ 0	**DISTANCE**
CANSCALED	Canberra metric distance, scaled	0 to 1	**DISTANCE**
CANADKINS	Canberra metric distance, Adkins form	0 to 1	**DISTANCE**
COSINE	Cosine coefficient	0 to 1	**SIMILAR**
DOT	Dot (inner) product coefficient	≥ 0	**SIMILAR**
OVERLAP	Overlap similarity	≥ 0	**SIMILAR**
DOVERLAP	Overlap dissimilarity	≥ 0	**DISTANCE**
CHISQ	Chi-square coefficient	≥ 0	**DISTANCE**
CHI	Square root of chi-square coefficient	≥ 0	**DISTANCE**
PHISQ	Phi-square coefficient	≥ 0	**DISTANCE**
PHI	Square root of phi-square coefficient	≥ 0	**DISTANCE**

Table 43.5 shows *methods* that accept nominal variables. Similar methods are grouped together in the table.

Table 43.5 Methods That Accept Nominal Variables

Method	Description	Range	TYPE=
HAMMING	Hamming distance	0 to v	**DISTANCE**
MATCH	Simple matching coefficient	0 to 1	**SIMILAR**
DMATCH	Simple matching coefficient transformed to Euclidean distance	0 to 1	**DISTANCE**
DSQMATCH	Simple matching coefficient transformed to squared Euclidean distance	0 to 1	**DISTANCE**
HAMANN	Hamann coefficient	–1 to 1	**SIMILAR**
RT	Roger and Tanimoto	0 to 1	**SIMILAR**
SS1	Sokal and Sneath 1	0 to 1	**SIMILAR**
SS3	Sokal and Sneath 3	0 to 1	**SIMILAR**

Note that v denotes the number of variables (dimensionality).

Table 43.6 shows *methods* that accept asymmetric nominal variables. Use the ABSENT= option to create a value to be considered absent.

Table 43.6 Methods That Accept Asymmetric Nominal Variables

Method	Description	Range	TYPE=
DICE	Dice coefficient or Czekanowski/Sorensen similarity coefficient	0 to 1	**SIMILAR**
RR	Russell and Rao	0 to 1	**SIMILAR**
BLWNM	Binary Lance and Williams nonmetric, or Bray-Curtis coefficient	0 to 1	**DISTANCE**
K1	Kulcynski 1	≥ 0	**SIMILAR**

Table 43.7 shows *methods* that accept asymmetric nominal and ratio variables. Use the ABSENT= option to create a value to be considered absent. The table contains five columns. The third column contains possible range values if only one level of measurement (either ratio or asymmetric nominal but not both) is specified; the fourth column contains possible range values if both levels are specified.

The JACCARD method is equivalent to the SIMRATIO method if there is no asymmetric nominal variable; if both ratio and asymmetric nominal variables are present, the coefficient is computed as the sum of the coefficient from the ratio variables and the coefficient from the asymmetric nominal variables. See "Proximity Measures" in the section "Details: DISTANCE Procedure" on page 2835 for the formula and descriptions of the JACCARD method.

Table 43.7 Methods That Accept Asymmetric Nominal and Ratio Variables

Method	Description	Range (One Level)	Range (Two Levels)	TYPE=
JACCARD	Jaccard similarity coefficient	0 to 1	0 to 2	**SIMILAR**
DJACCARD	Jaccard dissimilarity coefficient	0 to 1	0 to 2	**DISTANCE**

MULT=*c*

specifies a numeric constant, *c*, by which to multiply each value after standardizing. The default value is 1.

NOMISS

omits observations with missing values from computation of the location and scale measures when standardizing; generates undefined (missing) distances for observations with missing values when computing distances. Use the UNDEF= option to specify the undefined values.

If a distance matrix is created to be used as an input to PROC CLUSTER, the NOMISS option should not be used because PROC CLUSTER does not accept distance matrices with missing values.

NORM

normalizes the scale estimator to be consistent for the standard deviation of a normal distribution when you specify the option STD=AGK, STD=IQR, STD=MAD, or STD=SPACING in the VAR statement.

NOSTD

suppresses standardization of the variables. The NOSTD option should not be specified with the STDONLY option or with the REPLACE option.

OUT=*SAS-data-set*

specifies the name of the SAS data set created by PROC DISTANCE. The output data set contains the BY variables, the ID variable, computed distance variables, the COPY variables, the FREQ variable, and the WEIGHT variables.

If you omit the OUT= option, PROC DISTANCE creates an output data set named according to the DATA*n* convention.

The output data set is of type TYPE=DISTANCE or TYPE=SIMILAR. See the METHOD= option for more information about the association between the method and the output data set type. Data set types do not persist when you copy or modify a data set. You must specify the TYPE= data set option for the new data set, as in the following example:

```
data dist2(type=distance);
   set dist;
run;
```

If you do not specify the TYPE=DISTANCE data set option, the new data set is the default TYPE=DATA. If you use the new data set in a procedure that accepts both TYPE=DATA or TYPE=DISTANCE data sets (such as PROC CLUSTER or PROC MODECLUS), the results will be incorrect.

OUTSDZ=*SAS-data-set*

specifies the name of the SAS data set containing the standardized scores. The output data set contains a copy of the DATA= data set, except that the analyzed variables have been standardized. Analyzed variables are those listed in the VAR statement.

PREFIX=*name*

specifies a prefix for naming the distance variables in the OUT= data set. By default, the names are Dist1, Dist2, . . . , Dist*n*. If you specify PREFIX=ABC, the variables are named ABC1, ABC2, . . . , ABCn. If the ID statement is also specified, the variables are named by appending the value of the ID variable to the prefix.

RANKSCORE=MIDRANK | INDEX

specifies the method of assigning scores to ordinal variables. The available methods are listed as follows:

MIDRANK assigns consecutive integers to each category with consideration of the frequency value. This is the default method.

INDEX assigns consecutive integers to each category regardless of frequencies.

The following example explains how each method assigns the rank scores. Suppose the data contain an ordinal variable ABC with values A, B, C. There are two ways to assign numbers. One is to use midranks, which depend on the frequencies of each category. Another is to assign consecutive integers to each category, regardless of frequencies.

Table 43.8 Example of Assigning Rank Scores

ABC	MIDRANK	INDEX
A	1.5	1
A	1.5	1
B	4	2
B	4	2
B	4	2
C	6	3

REPLACE

replaces missing data with zero in the standardized data (to correspond to the location measure before standardizing). To replace missing data with something else, use the MISSING= option in the VAR statement. The REPLACE option implies standardization.

You cannot specify the following *options* together:

- both the REPLACE and the REPONLY options
- both the REPLACE and the NOSTD options

REPONLY
replaces missing data by the location measure that is specified by the MISSING= option or the STD= option (if the MISSING= option is not specified), but does *not* standardize the data. If the MISSING= option is not specified and METHOD=GOWER is specified, missing values are replaced by the location measure from the RANGE method (the minimum value), and the STD= option is.

You cannot specify both the REPLACE and the REPONLY options.

SHAPE=TRIANGLE | TRI | SQUARE | SQU | SQR
specifies the shape of the proximity matrix to be stored in the OUT= data set. SHAPE=TRIANGLE requests the matrix to be stored as a lower triangular matrix; SHAPE=SQUARE requests that the matrix be stored as a square matrix. Use SHAPE=SQUARE if the output data set is to be used as input to the MODECLUS procedures. The default is TRIANGLE.

SNORM
normalizes the scale estimator to have an expectation of approximately 1 for a standard normal distribution when the STD=SPACING option is specified.

STDONLY
standardizes variables only and computes no distance matrix. You must use the OUTSDZ= option to save the standardized scores. You cannot specify both the STDONLY option and the NOSTD option.

UNDEF=*n*
specifies the numeric constant used to replace undefined distances, such as when an observation has all missing values, or if a divisor is zero.

VARDEF=DF | N | WDF | WEIGHT | WGT
specifies the divisor to be used in the calculation of distance, dissimilarity, or similarity measures, and for standardizing variables whenever a variance or covariance is computed. By default, VARDEF=DF. The values and associated divisors are as follows:

Value	Divisor	Formula
DF	Degrees of freedom	$n - 1$
N	Number of observations	n
WDF	Sum of weights minus 1	$(\sum_i w_i) - 1$
WEIGHT | WGT	Sum of weights	$\sum_i w_i$

VAR Statement

VAR *level* **(***variables* < / *options* >**)**< *level* **(***variables* < / *options* >**)**. . . > **;**

where the syntax for the *options* is as follows:

ABSENT=*value*
MISSING=*miss-method* | *value*
ORDER=*order-option*
STD=*std-method*
WEIGHTS=*weight-list*

The VAR statement lists variables from which distances are to be computed. The VAR statement is required. The variables can be numeric or character depending on their measurement levels. A variable cannot appear more than once in either the same list or a different list.

level is required. It declares the levels of measurement for those variables specified within the parentheses. Available values for *level* are as follows:

ANOMINAL variables are asymmetric nominal and can be either numeric or character.

NOMINAL variables are symmetric nominal and can be either numeric or character.

ORDINAL variables are ordinal and can be either numeric or character. Values of ordinal variables are replaced by their corresponding rank scores. If standardization is required, the standardized rank scores are output to the data set specified in the OUTSDZ= option. See the RANKSCORE= option in the PROC DISTANCE statement for methods available for assigning rank scores to ordinal variables. After being replaced by scores, ordinal variables are considered interval.

INTERVAL variables are interval and numeric.

RATIO variables are ratio and numeric. Ratio variables should always contain positive measurements.

Each variable list can be followed by an *option* list. Use "/ " after the list of variables to start the option list. An *option* list contains options that are applied to the variables. The following *options* are available in the option list:

ABSENT= specifies the value to be used as an absence value in an irrelevant absent-absent match for asymmetric nominal variables.

MISSING= specifies the method (or numeric value) with which to replace missing data.

ORDER= selects the order for assigning scores to ordinal variables.

STD= selects the standardization method.

WEIGHTS= assigns weights to the variables in the list.

If an *option* is missing from the current attribute list, PROC DISTANCE provides default values for all the variables in the current list.

For example, in the VAR statement

```
var ratio(x1-x4/std= mad weights= .5 .5 .1 .5 missing= -99)
    interval(x5/std= range)
    ordinal(x6/order= desc);
```

the first option list defines x1–x4 as ratio variables to be standardized by the MAD method. Also, any missing values in x1–x4 should be replaced by –99. x1 is given a weight of 0.5, x2 is given a weight of 0.5, x3 is given a weight of 0.1, and x4 is given a weight of 0.5.

The second option list defines x5 as an interval variable to be standardized by the RANGE method. If the REPLACE option is specified in the PROC DISTANCE statement, missing values in x5 are replaced by the location estimate from the RANGE method. By default, x5 is given a weight of 1.

The last option list defines x6 as an ordinal variable. The scores are assigned from highest to lowest by its unformatted values. Although the STD= option is not specified, x6 is standardized by the default method (STD) because there is more than one level of measurements (ratio, interval, and ordinal) in the VAR statement. Again, if the REPLACE option is specified, missing values in x6 are replaced by the location estimate from the STD method. Finally, by default, x6 is given a weight of 1.

More details for the options are explained as follows.

STD=*std-method*

specifies the standardization method. Valid values for *std-method* are MEAN, MEDIAN, SUM, EUCLEN, USTD, STD, RANGE, MIDRANGE, MAXABS, IQR, MAD, ABW, AHUBER, AWAVE, AGK, SPACING, and L. Table 43.9 lists available methods of standardization as well as their corresponding location and scale measures.

Table 43.9 Available Standardization Methods

Method	Scale	Location
MEAN	1	Mean
MEDIAN	1	Median
SUM	Sum	0
EUCLEN	Euclidean length	0
USTD	Standard deviation about origin	0
STD	Standard deviation	Mean
RANGE	Range	Minimum
MIDRANGE	Range/2	Midrange
MAXABS	Maximum absolute value	0
IQR	Interval quartile range	Median
MAD	Median absolute deviation from median	Median
ABW(c)	Biweight A-estimate	Biweight 1-step M-estimate
AHUBER(c)	Huber A-estimate	Huber 1-step M-estimate
AWAVE(c)	Wave 1-step M-estimate	Wave A-estimate
AGK(p)	AGK estimate (ACECLUS)	Mean
SPACING(p)	Minimum spacing	Mid minimum-spacing
L(p)	L_p	L_p

These standardization methods are further documented in the section on the METHOD= option in the PROC STDIZE statement of the STDIZE procedure (see the section "Standardization Methods" on page 9652 in Chapter 115, "The STDIZE Procedure").

Standardization is not required if there is only one level of measurement, or if only asymmetric nominal and nominal levels are specified; otherwise, standardization is mandatory. When standardization is mandatory, a default method is provided when the STD= option is not specified. You can suppress the mandatory standardization by using the NOSTD option in the PROC DISTANCE statement. See the NOSTD option in the section "PROC DISTANCE Statement" on page 2821 and the section "Mandatory Standardization" on page 2816 for details.

The default method is STD for standardizing interval variables unless METHOD=GOWER or METHOD=DGOWER is specified. If METHOD=GOWER is specified, interval variables are standardized by the RANGE method, and whatever is specified in the STD= option is ignored; if METHOD=DGOWER is specified, the RANGE method is the default standardization method for interval variables. The MAXABS method is the default standardization method for ratio variables for both the GOWER and DGOWER methods.

Notice that a ratio variable should always be positive.

Table 43.10 lists standardization methods and the levels of measurement that can be accepted by each method. For example, the SUM method can be used to standardize ratio variables but not interval or ordinal variables. Also, the AGK and SPACING methods should not be used to standardize ordinal variables. If you apply AGK and SPACING to ranks, the results are degenerate because all the spacings of a given order are equal.

Table 43.10 Legitimate Levels of Measurements for Each Method

Standardization Method	Legitimate Levels of Measurement
MEAN	Ratio, interval, ordinal
MEDIAN	Ratio, interval, ordinal
SUM	Ratio
EUCLEN	Ratio
USTD	Ratio
STD	Ratio, interval, ordinal
RANGE	Ratio, interval, ordinal
MIDRANGE	Ratio, interval, ordinal
MAXABS	Ratio
IQR	Ratio, interval, ordinal
MAD	Ratio, interval, ordinal
ABW(c)	Ratio, interval, ordinal
AHUBER(c)	Ratio, interval, ordinal
AWAVE(c)	Ratio, interval, ordinal
AGK(p)	Ratio, interval
SPACING(p)	Ratio, interval
L(p)	Ratio, interval, ordinal

ABSENT=*number* | *qs*

specifies the value to be used as an absence value in an irrelevant absent-absent match for asymmetric nominal variables. The absence value specified here overwrites the absence value specified through the ABSENT= option in the PROC DISTANCE statement for those variables in the current variable list.

An absence value for a variable can be either a numeric value or a quoted string consisting of combinations of characters. For instance, ., –999, “NA” are legal values for the ABSENT= option.

The default for an absence value for a character variable is “NONE” (notice that a blank value is considered a missing value), and the default for an absence value for a numeric variable is 0.

MISSING=*miss-method* | *value*

specifies the method or a numeric value for replacing missing values. If you omit the MISSING= option, the REPLACE option replaces missing values with the location measure given by the STD= option. Specify the MISSING= option when you want to replace missing values with a different value. You can specify any method that is valid in the STD= option. The corresponding location measure is used to replace missing values.

If a numeric value is given, the value replaces missing values after standardizing the data. However, when standardization is not mandatory, you can specify the REPONLY option with the MISSING= option to suppress standardization for cases in which you want only to replace missing values.

If the NOSTD option is specified, there is no standardization, but missing values are replaced by the corresponding location measures or by the numeric value of the MISSING= option. See the section "Missing Values" on page 2843 for details about missing values replacement with and without standardization.

ORDER=ASCENDING | DESCENDING | ASCFORMATTED | DESFORMATTED | DSORDER
ORDER=ASC | DESC | ASCFMT | DESFMT | DATA

specifies the order for assigning score to ordinal variables. The value for the ORDER= option can be one of the following:

ASCENDING scores are assigned in lowest-to-highest order of unformatted values.

DESCENDING scores are assigned in highest-to-lowest order of unformatted values.

ASCFORMATTED scores are assigned in ascending order by their formatted values. This option can be applied to character variables only, since unformatted values are always used for numeric variables.

DESFORMATTED scores are assigned in descending order by their formatted values. This option can be applied to character variables only, since unformatted values are always used for numeric variables.

DSORDER scores are assigned according to the order of their appearance in the input data set.

The default value is ASCENDING.

WEIGHTS=*weight-list*

specifies a list of values for weighting individual variables while computing the proximity. Values in this list can be separated by blanks or commas. You can include one or more items of the form *start* TO *stop* BY *increment*. This list should contain at least one weight. The maximum number of weights you can list is equal to the number of variables. If the number of weights is less than the number of variables, the last value in the *weight-list* is used for the rest of the variables; conversely, if the number of weights is greater than the number of variables, the trailing weights are discarded.

The default value is 1.

ID Statement

ID *variable* **;**

The ID statement specifies a single variable to be copied to the OUT= data set and used to generate names for the distance variables. The ID variable must be character.

Typically, each ID value occurs only once in the input data set or, if you use a BY statement, only once within a BY group.

If you specify both the ID and BY statements, the ID variable must have the same values in the same order in each BY group.

COPY Statement

COPY *variables* **;**

The COPY statement specifies a list of additional variables to be copied to the OUT= data set.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC DISTANCE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the DISTANCE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the “Grouping Data” section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ | **FREQUENCY** *variable* **;**

The frequency variable is used both for standardizing variables and for assigning rank scores to ordinal variables. It has no direct effect on computing the distances.

For standardizing variables and assigning rank scores, PROC DISTANCE treats the data set as if each observation appeared n times, where n is the value of the FREQ variable for the observation. Nonintegral values of the FREQ variable are truncated to the largest integer less than the FREQ value. If the FREQ variable has a value that is less than 1 or is missing, the observation is not used in the analysis, and distances between that observation and other observations are set to missing

WEIGHT Statement

WGT | **WEIGHT** *variable* **;**

The WEIGHT statement specifies a numeric variable in the input data set with values that are used to weight each observation. This weight variable is used for standardizing variables rather than computing the distances. Only one variable can be specified.

The WEIGHT variable values can be nonintegers. An observation is used in the analysis only if the value of the WEIGHT variable is greater than zero. The WEIGHT variable applies to variables that are standardized by the following *options*: STD=MEAN, STD=SUM, STD=EUCLEN, STD=USTD, STD=STD, STD=AGK, or STD=L.

PROC DISTANCE uses the value of the WEIGHT variable w_i to compute the sample mean, uncorrected sample variances, and sample variances as follows:

$$\overline{x}_w = \sum_i w_i x_i / \sum_i w_i$$

$$u_w^2 = \sum_i w_i x_i^2 / d$$

$$s_w^2 = \sum_i w_i (x_i - \overline{x}_w)^2 / d$$

w_i is the weight value of the ith observation, x_i is the value of the ith observation, and d is the divisor controlled by the VARDEF= option (see the VARDEF= option in the PROC DISTANCE statement for details).

PROC DISTANCE uses the value of the WEIGHT variable to calculate the following statistics for standardization:

MEAN	the weighted mean, $\overline{x}_w$
SUM	the weighted sum, $\sum_i w_i x_i$
USTD	the weighted uncorrected standard deviation, $\sqrt{u_w^2}$
STD	the weighted standard deviation, $\sqrt{s_w^2}$
EUCLEN	the weighted Euclidean length, computed as the square root of the weighted uncorrected sum of squares: $$\sqrt{\sum_i w_i x_i^2}$$
AGK	the AGK estimate. This estimate is documented further in the ACECLUS procedure as the METHOD=COUNT option. See the discussion of the WEIGHT statement in Chapter 27, "The ACECLUS Procedure," for information about how the WEIGHT variable is applied to the AGK estimate.
L	the L_p estimate. This estimate is documented further in the FASTCLUS procedure as the LEAST= option. See the discussion of the WEIGHT statement in Chapter 45, "The FASTCLUS Procedure," for information about how the WEIGHT variable is used to compute weighted cluster means. Note that the number of clusters is always 1.

Details: DISTANCE Procedure

Proximity Measures

The following notation is used in this section:

v	the number of variables or the dimensionality
x_j	data for observation x and the jth variable, where $j = 1$ to v
y_j	data for observation y and the jth variable, where $j = 1$ to v
w_j	weight for the jth variable from the WEIGHTS= option in the VAR statement. The weight $w_j = 0$ when either x_j or y_j is missing.
W	the sum of total weights. Regardless of whether the observation is missing, its weight is added to this metric.
$\bar{x}$	mean for observation x $\bar{x} = \sum_{j=1}^{v} w_j x_j / \sum_{j=1}^{v} w_j$
$\bar{y}$	mean for observation y $\bar{y} = \sum_{j=1}^{v} w_j y_j / \sum_{j=1}^{v} w_j$
$d(x, y)$	the distance or dissimilarity between observations x and y
$s(x, y)$	the similarity between observations x and y

The factor $W / \sum_{j=1}^{v} w_j$ is used to adjust some of the proximity measures for missing values.

Methods That Accept All Measurement Levels

GOWER

Gower's similarity

$s_1(x, y) = \sum_{j=1}^{v} w_j \delta_{x,y}^j d_{x,y}^j / \sum_{j=1}^{v} w_j \delta_{x,y}^j$

You compute $\delta_{x,y}^j$ as follows:

For nominal, ordinal, interval, or ratio variables,

$$\delta_{x,y}^j = 1$$

For asymmetric nominal variables,

$$\delta_{x,y}^j = 1, \text{ if either } x_j \text{ or } y_j \text{ is present}$$
$$\delta_{x,y}^j = 0, \text{ if both } x_j \text{ and } y_j \text{ are absent}$$

For nominal or asymmetric nominal variables,

$$d_{x,y}^j = 1, \text{ if } x_j = y_j$$
$$d_{x,y}^j = 0, \text{ if } x_j \neq y_j$$

For ordinal, interval, or ratio variables,

$$d_{x,y}^j = 1 - |x_j - y_j|$$

DGOWER

1 minus Gower

$d_2(x, y) = 1 - s_1(x, y)$

Methods That Accept Ratio, Interval, and Ordinal Variables

Similar methods are presented together in the following list.

EUCLID

Euclidean distance

$d_3(x, y) = \sqrt{(\sum_{j=1}^{v} w_j (x_j - y_j)^2) W / (\sum_{j=1}^{v} w_j)}$

SQEUCLID

Squared Euclidean distance

$d_4(x, y) = (\sum_{j=1}^{v} w_j (x_j - y_j)^2) W / (\sum_{j=1}^{v} w_j)$

SIZE

size distance

$d_5(x, y) = |\sum_{j=1}^{v} w_j (x_j - y_j)| \sqrt{W} / (\sum_{j=1}^{v} w_j)$

SHAPE

shape distance

$d_6(x, y) = \sqrt{(\sum_{j=1}^{v} w_j [(x_j - \bar{x}) - (y_j - \bar{y})]^2) W / (\sum_{j=1}^{v} w_j)}$

NOTE: Squared shape distance plus squared size distance equals squared Euclidean distance.

COV — covariance similarity coefficient

$s_7(x, y) = \sum_{j=1}^{v} w_j (x_j - \bar{x})(y_j - \bar{y}) / vardiv$, where

$$\begin{aligned} vardiv &= v \text{ if VARDEF} = \text{N} \\ &= v - 1 \text{ if VARDEF} = \text{DF} \\ &= \textstyle\sum_{j=1}^{v} w_j \text{ if VARDEF} = \text{WEIGHT} \\ &= \textstyle\sum_{j=1}^{v} w_j - 1 \text{ if VARDEF} = \text{WDF} \end{aligned}$$

CORR — correlation similarity coefficient

$$s_8(x, y) = \frac{\sum_{j=1}^{v} w_j (x_j - \bar{x})(y_j - \bar{y})}{\sqrt{\sum_{j=1}^{v} w_j (x_j - \bar{x})^2 \sum_{j=1}^{v} w_j (y_j - \bar{y})^2}}$$

DCORR — correlation transformed to Euclidean distance as sqrt(1–CORR)

$d_9(x, y) = \sqrt{1 - s_8(x, y)}$

SQCORR — squared correlation

$$s_{10}(x, y) = \frac{[\sum_{j=1}^{v} w_j (x_j - \bar{x})(y_j - \bar{y})]^2}{\sum_{j=1}^{v} w_j (x_j - \bar{x})^2 \sum_{j=1}^{v} w_j (y_j - \bar{y})^2}$$

DSQCORR — squared correlation transformed to squared Euclidean distance as (1–SQCORR)

$d_{11}(x, y) = 1 - s_{10}(x, y)$

L(p) — Minkowski (L_p) distance, where p is a positive numeric value

$d_{12}(x, y) = [(\sum_{j=1}^{v} w_j |x_j - y_j|^p) W / (\sum_{j=1}^{v} w_j)]^{1/p}$

CITYBLOCK — L_1

$d_{13}(x, y) = (\sum_{j=1}^{v} w_j |x_j - y_j|) W / (\sum_{j=1}^{v} w_j)$

CHEBYCHEV — L_∞

$d_{14}(x, y) = \max_{j=1}^{v} w_j |x_j - y_j|$

POWER(p,r) — generalized Euclidean distance, where p is a nonnegative numeric value and r is a positive numeric value. The distance between two observations is the rth root of sum of the absolute differences to the pth power between the values for the observations:

$d_{15}(x, y) = [(\sum_{j=1}^{v} w_j |x_j - y_j|^p) W / (\sum_{j=1}^{v} w_j)]^{1/r}$

Methods That Accept Ratio Variables

Similar methods are presented together in the following list.

SIMRATIO similarity ratio

$$s_{16}(x, y) = \frac{\sum_j^v w_j(x_j y_j)}{\sum_{j=1}^v w_j(x_j y_j) + \sum_j^v w_j(x_j - y_j)^2}$$

DISRATIO one minus similarity ratio

$$d_{17}(x, y) = 1 - s_{16}(x, y)$$

NONMETRIC Lance-Williams nonmetric coefficient

$$d_{18}(x, y) = \frac{\sum_{j=1}^v w_j|x_j - y_j|}{\sum_{j=1}^v w_j(x_j + y_j)}$$

CANBERRA Canberra metric distance. See Sneath and Sokal (1973, pp. 125–126)

$$d_{19}(x, y) = \sum_{j=1}^v w_j \frac{|x_j - y_j|}{(x_j + y_j)}$$

NOTE: When both x_j and y_j are zeros, the corresponding component fraction $\frac{|x_j - y_j|}{(x_j + y_j)}$ is defined to be zero. Two variants of this distance measure, CANSCALED and CANADKINS, follow.

CANSCALED Canberra metric distance, scaled

$$d_{19s}(x, y) = \frac{1}{\sum_{j=1}^v w_j} \sum_{j=1}^v w_j \frac{|x_j - y_j|}{(x_j + y_j)}$$

NOTE: This measure is a scaled version of $d_{19}(x, y)$ so that the resulting range falls between 0 and 1.

CANADKINS Canberra metric distance, Adkins form (Lance and Williams 1967)

$$d_{19a}(x, y) = \frac{1}{\sum_{j=1}^v w_j(1 - z_j)} \sum_{j=1}^v w_j \frac{|x_j - y_j|}{(x_j + y_j)}$$

where z_j is an indicator variable for zero values in both x_j and y_j.

NOTE: This measure is similar to the scaled CANBERRA distance, $d_{19s}(x, y)$, but without considering the zero-valued (x_j, y_j) pairs in the scaling.

COSINE cosine coefficient

$$s_{20}(x, y) = \frac{\sum_{j=1}^v w_j(x_j y_j)}{\sqrt{\sum_{j=1}^v w_j x_j{}^2 \sum_{j=1}^v w_j y_j{}^2}}$$

DOT dot (inner) product coefficient

$$s_{21}(x, y) = [\sum_{j=1}^v w_j(x_j y_j)] / \sum_{j=1}^v w_j$$

OVERLAP sum of the minimum values

$$s_{22}(x, y) = \sum_{j=1}^v w_j[\min(x_j, y_j)]$$

DOVERLAP maximum of the sum of the x and the sum of y minus overlap

$$d_{23}(x, y) = \max(\sum_{j=1}^v w_j x_j, \sum_{j=1}^v w_j y_j) - s_{22}(x, y)$$

CHISQ chi-square

If the data represent the frequency counts, chi-square dissimilarity between two sets of

frequencies can be computed. A 2-by-v contingency table is illustrated to explain how the chi-square dissimilarity is computed as follows:

	Variable				Row
Observation	Var 1	Var 2	...	Var v	Sum
X	x_1	x_2	...	x_v	r_x
Y	y_1	y_2	...	y_v	r_y
Column Sum	c_1	c_2	...	c_v	T

where

$$\begin{aligned} r_x &= \textstyle\sum_{j=1}^{v} w_j x_j \\ r_y &= \textstyle\sum_{j=1}^{v} w_j y_j \\ c_j &= w_j(x_j + y_j) \\ T &= r_x + r_y = \textstyle\sum_{j=1}^{v} c_j \end{aligned}$$

The chi-square measure is computed as follows:

$$d_{24}(x, y) = \left(\sum_{j=1}^{v} \frac{(w_j x_j - E(x_j))^2}{E(x_j)} + \sum_{j=1}^{v} \frac{(w_j y_j - E(y_j))^2}{E(y_j)}\right) W / \left(\sum_{j=1}^{v} w_j\right)$$

where for j= 1, 2, ..., v

$$\begin{aligned} E(x_j) &= r_x c_j / T \\ E(y_j) &= r_y c_j / T \end{aligned}$$

CHI — square root of chi-square

$$d_{25}(x, y) = \sqrt{d_{23}(x, y)}$$

PHISQ — phi-square

This is the CHISQ dissimilarity normalized by the sum of weights

$$d_{26}(x, y) = d_{24}(x, y) / \left(\sum_{j=1}^{v} w_j\right)$$

PHI — square root of phi-square

$$d_{27}(x, y) = \sqrt{d_{25}(x, y)}$$

Methods That Accept Symmetric Nominal Variables

The following notation is used for computing $d_{28}(x, y)$ to $s_{35}(x, y)$. Notice that only the nonmissing pairs are discussed here; all the pairs with at least one missing value are excluded from any of the computations in the following section because $w_j = 0$ if either x_j or y_j is missing.

M — nonmissing matches

$M = \sum_{j=1}^{v} w_j \delta_{x,y}^{j}$, where

$$\begin{aligned} \delta_{x,y}^{j} &= 1, \text{ if } x_j = y_j \\ \delta_{x,y}^{j} &= 0, \text{ otherwise} \end{aligned}$$

X nonmissing mismatches

$X = \sum_{j=1}^{v} w_j \delta_{x,y}^j$, where

$$\begin{aligned} \delta_{x,y}^j &= 1, \text{ if } x_j \neq y_j \\ \delta_{x,y}^j &= 0, \text{ otherwise} \end{aligned}$$

N total nonmissing pairs

$N = \sum_{j=1}^{v} w_j$

The following list presents distance methods that accept symmetric nominal variables. Similar methods are grouped together.

HAMMING Hamming distance

$d_{28}(x, y) = X$

MATCH simple matching coefficient

$s_{29}(x, y) = M/N$

DMATCH simple matching coefficient transformed to Euclidean distance

$d_{30}(x, y) = \sqrt{1 - M/N} = \sqrt{(X/N)}$

DSQMATCH simple matching coefficient transformed to squared Euclidean distance

$d_{31}(x, y) = 1 - M/N = X/N$

HAMANN Hamann coefficient

$s_{32}(x, y) = (M - X)/N$

RT Roger and Tanimoto

$s_{33}(x, y) = M/(M + 2X)$

SS1 Sokal and Sneath 1

$s_{34}(x, y) = 2M/(2M + X)$

SS3 Sokal and Sneath 3. The coefficient between an observation and itself is always indeterminate (missing) because there is no mismatch.

$s_{35}(x, y) = M/X$

Methods That Accept Asymmetric Nominal and Ratio Variables

The following notation is used for computing $s_{36}(x, y)$ to $d_{41}(x, y)$. Notice that only the nonmissing pairs are discussed in this section; all the pairs that have at least one missing value are excluded from any of the computations here because $w_j = 0$ if either x_j or y_j is missing.

Also, the observed nonmissing data of an asymmetric binary variable can have only two possible outcomes: presence or absence. Therefore, the notation, *PX* (present mismatches), always has a value of zero for an asymmetric binary variable.

X mismatches with at least one present

$X = \sum_{j=1}^{v} w_j \delta_{x,y}^j$, where

$$\begin{aligned} \delta_{x,y}^j &= 1, \text{ if } x_j \neq y_j \text{ and not both } x_j \text{ and } y_j \text{ are absent} \\ \delta_{x,y}^j &= 0, \text{ otherwise} \end{aligned}$$

PM present matches

$PM = \sum_{j=1}^{v} w_j \delta_{x,y}^j$, where

$$\begin{aligned} \delta_{x,y}^j &= 1, \text{ if } x_j = y_j \text{ and both } x_j \text{ and } y_j \text{ are present} \\ \delta_{x,y}^j &= 0, \text{ otherwise} \end{aligned}$$

PX present mismatches

$PX = \sum_{j=1}^{v} w_j \delta_{x,y}^j$, where

$$\begin{aligned} \delta_{x,y}^j &= 1, \text{ if } x_j \neq y_j \text{ and both } x_j \text{ and } y_j \text{ are present} \\ \delta_{x,y}^j &= 0, \text{ otherwise} \end{aligned}$$

PP both present = *PM* + *PX*

P at least one present = *PM* + *X*

PAX present-absent mismatches

$PAX = \sum_{j=1}^{v} w_j \delta_{x,y}^j$, where

$$\begin{aligned} \delta_{x,y}^j &= 1, \text{ if } x_j \neq y_j \text{ and either } x_j \text{ is present and } y_j \text{ is absent or} \\ &\quad\ \ x_j \text{ is absent and } y_j \text{ is present} \\ \delta_{x,y}^j &= 0 \text{ otherwise} \end{aligned}$$

N total nonmissing pairs

$N = \sum_{j=1}^{v} w_j$

The following list presents distance methods that accept asymmetric nominal and ratio variables:

JACCARD Jaccard similarity coefficient

The JACCARD method is equivalent to the SIMRATIO method if there are only ratio variables; if there are both ratio and asymmetric nominal variables, the coefficient is

computed as the sum of the coefficient from the ratio variables (SIMRATIO) and the coefficient from the asymmetric nominal variables.

$s_{36}(x, y) = s_{16}(x, y) + PM/P$

DJACCARD Jaccard dissimilarity coefficient

The DJACCARD method is equivalent to the DISRATIO method if there are only ratio variables; if there are both ratio and asymmetric nominal variables, the coefficient is computed as the sum of the coefficient from the ratio variables (DISRATIO) and the coefficient from the asymmetric nominal variables.

$d_{37}(x, y) = d_{17}(x, y) + X/P$

Methods That Accept Asymmetric Nominal Variables

With reference to the notation defined in the previous section, the following list presents distance methods that accept asymmetric nominal variables:

DICE Dice coefficient or Czekanowski/Sorensen similarity coefficient
$s_{38}(x, y) = 2PM/(P + PM)$

RR Russell and Rao. This is the binary equivalent of the dot product coefficient.
$s_{39}(x, y) = PM/N$

BLWNM | BRAYCURTIS Binary Lance and Williams, also known as Bray and Curtis coefficient

$d_{40}(x, y) = X/(PAX + 2PP)$

K1 Kulcynski 1. The coefficient between an observation and itself is always indeterminate (missing) because there is no mismatch.
$d_{41}(x, y) = PM/X$

Missing Values

Standardization versus No Standardization

You can replace the missing values with or without standardization. Missing values are replaced after standardization by specifying either the REPLACE option in the PROC DISTANCE statement or the MISSING= option in the VAR statement.

To replace missing values without standardization, use the following two options:

- the NOSTD option in the PROC DISTANCE statement. This option suppresses standardization but still replaces the missing values with the location of the method or the numeric value specified in the MISSING= option in the VAR statement.
- the REPONLY option in the PROC DISTANCE statement. PROC DISTANCE replaces missing values with the location of the standardization method or with the numeric value specified in the MISSING= option in the VAR statement. This approach assumes that standardization is not mandatory (see the section "Standardization" on page 2816).

Eliminating Observations with Missing Values

If you specify the NOMISS option, PROC DISTANCE omits observations with any missing values in the analyzed variables from computation of the location and scale measures.

Distance Measures

If you specify the NOMISS option, PROC DISTANCE generates missing distance for observations with missing values. If the NOMISS option is not specified, the sum of total weights, no matter if an observation is missing or not, is incorporated into the computation of some of the proximity measures. See the section "Details: DISTANCE Procedure" on page 2835 for the formulas and descriptions.

Formatted versus Unformatted Values

PROC DISTANCE uses the formatted values from a character variable, if the variable has a format—for example, one assigned by a format statement. PROC DISTANCE uses the unformatted values from a numeric variable, even if it has a format.

Output Data Sets

OUT= Data Set

The DISTANCE procedure always produces an output data set, regardless of whether you specify the OUT= option in the PROC DISTANCE statement. PROC DISTANCE displays no output. Use PROC PRINT to display the output data set.

The output data set contains the following variables:

- the ID variable, if any
- the BY variables, if any
- the COPY variables, if any
- the FREQ variable, if any
- the WEIGHT variable, if any
- the new distance variables, named from PREFIX= options along with the ID values, or from the default values

The output data set is of type TYPE=DISTANCE or TYPE=SIMILAR. See the METHOD= option for more information about the output data set types. Data set types do not persist when you copy or modify a data set. You must specify the TYPE= data set option for the new data set, as in the following example:

```
data dist2(type=distance);
   set dist;
run;
```

See the OUT= option for more information about data set type persistence.

OUTSDZ= Data Set

The output data set is a copy of the DATA= data set except that the analyzed variables have been standardized. Analyzed variables are those listed in the VAR statement.

Examples: DISTANCE Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 43.1: Divorce Grounds—the Jaccard Coefficient

(View the complete code for this example (distanx1.sas) in the example repository.)

A wide variety of distance and similarity measures are used in cluster analysis (Anderberg 1973; Sneath and Sokal 1973). If your data are in coordinate form and you want to use a non-Euclidean distance for clustering, you can compute a distance matrix by using the DISTANCE procedure.

Similarity measures must be converted to dissimilarities before being used in PROC CLUSTER. Such conversion can be done in a variety of ways, such as taking reciprocals or subtracting from a large value. The choice of conversion method depends on the application and the similarity measure. If applicable, PROC DISTANCE provides a corresponding dissimilarity measure for each similarity measure.

In the following example, the observations are states. Binary-valued variables correspond to various grounds for divorce and indicate whether the grounds for divorce apply in each of the U.S. states. A value of "1" indicates that the ground for divorce applies, and a value of "0" indicates the opposite. The 0-0 matches are treated as totally irrelevant; therefore, each variable has an asymmetric nominal level of measurement. The absence value is 0.

The DISTANCE procedure is used to compute the Jaccard coefficient (Anderberg 1973, pp. 89, 115, and 117) between each pair of states. The Jaccard coefficient is defined as the number of variables that are coded as 1 for both states divided by the number of variables that are coded as 1 for either or both states. Since dissimilarity measures are required by PROC CLUSTER, the DJACCARD coefficient is selected. Output 43.1.1 displays the distance matrix between the first 10 states.

The CENTROID method is used to perform the cluster analysis, and the resulting tree diagram from PROC CLUSTER is saved into the tree output data set. Output 43.1.2 displays the cluster history.

The TREE procedure generates nine clusters in the output data set out. After being sorted by the state, the out data set is then merged with the input data set divorce. After being sorted by the state, the merged data set is printed to display the cluster membership as shown in Output 43.1.3.

The following statements produce Output 43.1.1 through Output 43.1.3:

```
data divorce;
   length State $ 15;
   input State &$
         Incompatibility Cruelty Desertion Non_Support Alcohol
         Felony Impotence Insanity Separation @@;
   datalines;
Alabama          1 1 1 1 1 1 1 1 1    Alaska           1 1 1 0 1 1 1 1 0
Arizona          1 0 0 0 0 0 0 0 0    Arkansas         0 1 1 1 1 1 1 1 1
California       1 0 0 0 0 0 0 1 0    Colorado         1 0 0 0 0 0 0 0 0
Connecticut      1 1 1 1 1 1 0 1 1    Delaware         1 0 0 0 0 0 0 0 1
```

```
   ... more lines ...

Wisconsin        1 0 0 0 0 0 0 0 1    Wyoming          1 0 0 0 0 0 0 1 1
;

title 'Grounds for Divorce';
proc distance data=divorce method=djaccard absent=0 out=distjacc;
   var anominal(Incompatibility--Separation);
   id state;
run;

proc print data=distjacc(obs=10);
   id state; var alabama--georgia;
   title2 'First 10 States';
run;
title2;

proc cluster data=distjacc method=centroid
             pseudo outtree=tree;
   id state;
   var alabama--wyoming;
run;

proc tree data=tree noprint n=9 out=out;
   id state;
run;

proc sort;
   by state;
run;

data clus;
   merge divorce out;
   by state;
run;

proc sort;
   by cluster;
run;

proc print;
   id state;
   var Incompatibility--Separation;
   by cluster;
run;
```

Output 43.1.1 Distance Matrix Based on the Jaccard Coefficient

Grounds for Divorce
First 10 States

State	Alabama	Alaska	Arizona	Arkansas	California	Colorado	Connecticut	Delaware	Florida	Georgia
Alabama	0.00000	.	.	.	.	.	.	.	.	.
Alaska	0.22222	0.00000	.	.	.	.	.	.	.	.
Arizona	0.88889	0.85714	0.00000	.	.	.	.	.	.	.
Arkansas	0.11111	0.33333	1.00000	0.00000	.	.	.	.	.	.
California	0.77778	0.71429	0.50000	0.88889	0.00000	.	.	.	.	.
Colorado	0.88889	0.85714	0.00000	1.00000	0.50000	0.00000	.	.	.	.
Connecticut	0.11111	0.33333	0.87500	0.22222	0.75000	0.87500	0.00000	.	.	.
Delaware	0.77778	0.87500	0.50000	0.88889	0.66667	0.50000	0.75000	0.00000	.	.
Florida	0.77778	0.71429	0.50000	0.88889	0.00000	0.50000	0.75000	0.66667	0.00000	.
Georgia	0.22222	0.00000	0.85714	0.33333	0.71429	0.85714	0.33333	0.87500	0.71429	0

Output 43.1.2 Clustering History

Grounds for Divorce

The CLUSTER Procedure
Centroid Hierarchical Cluster Analysis

Root-Mean-Square Distance Between Observations	0.694873

Output 43.1.2 *continued*

Cluster History

Number of Clusters	Clusters Joined		Freq	Pseudo F Statistic	Pseudo t-Squared	Norm Centroid Distance	Tie
49	Arizona	Colorado	2	.	.	0	T
48	California	Florida	2	.	.	0	T
47	Alaska	Georgia	2	.	.	0	T
46	Delaware	Hawaii	2	.	.	0	T
45	Connecticut	Idaho	2	.	.	0	T
44	CL49	Iowa	3	.	.	0	T
43	CL47	Kansas	3	.	.	0	T
42	CL44	Kentucky	4	.	.	0	T
41	CL42	Michigan	5	.	.	0	T
40	CL41	Minnesota	6	.	.	0	T
39	CL43	Mississippi	4	.	.	0	T
38	CL40	Missouri	7	.	.	0	T
37	CL38	Montana	8	.	.	0	T
36	CL37	Nebraska	9	.	.	0	T
35	North Dakota	Oklahoma	2	.	.	0	T
34	CL36	Oregon	10	.	.	0	T
33	Massachusetts	Rhode Island	2	.	.	0	T
32	New Hampshire	Tennessee	2	.	.	0	T
31	CL46	Washington	3	.	.	0	T
30	CL31	Wisconsin	4	.	.	0	T
29	Nevada	Wyoming	2	.	.	0	
28	Alabama	Arkansas	2	1561	.	0.1599	T
27	CL33	CL32	4	479	.	0.1799	T
26	CL39	CL35	6	265	.	0.1799	T
25	CL45	West Virginia	3	231	.	0.1799	
24	Maryland	Pennsylvania	2	199	.	0.2399	
23	CL28	Utah	3	167	3.2	0.2468	
22	CL27	Ohio	5	136	5.4	0.2698	
21	CL26	Maine	7	111	8.9	0.2998	
20	CL23	CL21	10	75.2	8.7	0.3004	
19	CL25	New Jersey	4	71.8	6.5	0.3053	T
18	CL19	Texas	5	69.1	2.5	0.3077	
17	CL20	CL22	15	48.7	9.9	0.3219	
16	New York	Virginia	2	50.1	.	0.3598	
15	CL18	Vermont	6	49.4	2.9	0.3797	
14	CL17	Illinois	16	47.0	3.2	0.4425	
13	CL14	CL15	22	29.2	15.3	0.4722	
12	CL48	CL29	4	29.5	.	0.4797	T
11	CL13	CL24	24	27.6	4.5	0.5042	
10	CL11	South Dakota	25	28.4	2.4	0.5449	
9	Louisiana	CL16	3	30.3	3.5	0.5844	
8	CL34	CL30	14	23.3	.	0.7196	
7	CL8	CL12	18	19.3	15.0	0.7175	
6	CL10	South Carolina	26	21.4	4.2	0.7384	
5	CL6	New Mexico	27	24.0	4.7	0.8303	
4	CL5	Indiana	28	28.9	4.1	0.8343	

Output 43.1.2 *continued*

Cluster History

Number of Clusters	Clusters Joined		Freq	Pseudo F Statistic	Pseudo t-Squared	Norm Centroid Distance	Tie
3	CL4	CL9	31	31.7	10.9	0.8472	
2	CL3	North Carolina	32	55.1	4.1	1.0017	
1	CL2	CL7	50	.	55.1	1.0663	

Output 43.1.3 Cluster Membership

Grounds for Divorce

CLUSTER=1

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
Arizona	1	0	0	0	0	0	0	0	0
Colorado	1	0	0	0	0	0	0	0	0
Iowa	1	0	0	0	0	0	0	0	0
Kentucky	1	0	0	0	0	0	0	0	0
Michigan	1	0	0	0	0	0	0	0	0
Minnesota	1	0	0	0	0	0	0	0	0
Missouri	1	0	0	0	0	0	0	0	0
Montana	1	0	0	0	0	0	0	0	0
Nebraska	1	0	0	0	0	0	0	0	0
Oregon	1	0	0	0	0	0	0	0	0

CLUSTER=2

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
California	1	0	0	0	0	0	0	1	0
Florida	1	0	0	0	0	0	0	1	0
Nevada	1	0	0	0	0	0	0	1	1
Wyoming	1	0	0	0	0	0	0	1	1

Output 43.1.3 *continued*

CLUSTER=3

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
Alabama	1	1	1	1	1	1	1	1	1
Alaska	1	1	1	0	1	1	1	1	0
Arkansas	0	1	1	1	1	1	1	1	1
Connecticut	1	1	1	1	1	1	0	1	1
Georgia	1	1	1	0	1	1	1	1	0
Idaho	1	1	1	1	1	1	0	1	1
Illinois	0	1	1	0	1	1	1	0	0
Kansas	1	1	1	0	1	1	1	1	0
Maine	1	1	1	1	1	0	1	1	0
Maryland	0	1	1	0	0	1	1	1	1
Massachusetts	1	1	1	1	1	1	1	0	1
Mississippi	1	1	1	0	1	1	1	1	0
New Hampshire	1	1	1	1	1	1	1	0	0
New Jersey	0	1	1	0	1	1	0	1	1
North Dakota	1	1	1	1	1	1	1	1	0
Ohio	1	1	1	0	1	1	1	0	1
Oklahoma	1	1	1	1	1	1	1	1	0
Pennsylvania	0	1	1	0	0	1	1	1	0
Rhode Island	1	1	1	1	1	1	1	0	1
South Dakota	0	1	1	1	1	1	0	0	0
Tennessee	1	1	1	1	1	1	1	0	0
Texas	1	1	1	0	0	1	0	1	1
Utah	0	1	1	1	1	1	1	1	0
Vermont	0	1	1	1	0	1	0	1	1
West Virginia	1	1	1	0	1	1	0	1	1

CLUSTER=4

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
Delaware	1	0	0	0	0	0	0	0	1
Hawaii	1	0	0	0	0	0	0	0	1
Washington	1	0	0	0	0	0	0	0	1
Wisconsin	1	0	0	0	0	0	0	0	1

CLUSTER=5

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
Louisiana	0	0	0	0	0	1	0	0	1
New York	0	1	1	0	0	1	0	0	1
Virginia	0	1	0	0	0	1	0	0	1

CLUSTER=6

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
South Carolina	0	1	1	0	1	0	0	0	1

Output 43.1.3 *continued*

CLUSTER=7

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
New Mexico	1	1	1	0	0	0	0	0	0

CLUSTER=8

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
Indiana	1	0	0	0	0	1	1	1	0

CLUSTER=9

State	Incompatibility	Cruelty	Desertion	Non_Support	Alcohol	Felony	Impotence	Insanity	Separation
North Carolina	0	0	0	0	0	0	1	1	1

Example 43.2: Financial Data—Stock Dividends

(View the complete code for this example (distanx2.sas) in the example repository.)

The following data set contains the average dividend yields for 15 utility stocks in the United States. The observations are names of the companies, and the variables correspond to the annual dividend yields for the period 1986–1990. The objective is to group similar stocks into clusters.

Before the cluster analysis is performed, the correlation similarity is chosen for measuring the closeness between each observation. Since distance type of measures are required by PROC CLUSTER, METHOD=DCORR is used in the PROC DISTANCE statement to transform the correlation measures to the distance measures. Notice that in Output 43.2.1, all the values in the distance matrix are between 0 and $\sqrt{2}$.

PROC CLUSTER performs hierarchical clustering by using agglomerative methods based on the distance data created from the previous PROC DISTANCE statement. Since the cubic clustering criterion is not suitable for distance data, only the pseudo F statistic is requested to identify the number of clusters.

The two clustering methods are Ward's and the average linkage methods. Since the results of the pseudo t^2 statistic from both Ward's and the average linkage methods contain many missing values, only the plot of the pseudo F statistic versus the number of clusters is requested along with the dendrogram by specifying PLOTS(ONLY)=(PSF DENDROGRAM) in the PROC CLUSTER statement.

Both Output 43.2.2 and Output 43.2.3 suggest four clusters. Both methods produce the same clustering result, as shown in Output 43.2.4 and Output 43.2.5. The four clusters are as follows:

- Cincinnati G&E and Detroit Edison
- Texas Utilities and Pennsylvania Power & Light
- Union Electric, Iowa-Ill Gas & Electric, Oklahoma Gas & Electric, and Wisconsin Energy
- Orange & Rockland Utilities, Kentucky Utilities, Kansas Power & Light, Allegheny Power, Green Mountain Power, Dominion Resources, and Minnesota Power & Light

```
title 'Stock Dividends';

data stock;
   length Company $ 27;
   input Company &$  Div_1986 Div_1987 Div_1988 Div_1989 Div_1990;
   datalines;
Cincinnati G&E               8.4    8.2    8.4    8.1    8.0
Texas Utilities              7.9    8.9   10.4    8.9    8.3
Detroit Edison               9.7   10.7   11.4    7.8    6.5
Orange & Rockland Utilities  6.5    7.2    7.3    7.7    7.9
Kentucky Utilities           6.5    6.9    7.0    7.2    7.5
Kansas Power & Light         5.9    6.4    6.9    7.4    8.0
Union Electric               7.1    7.5    8.4    7.8    7.7
Dominion Resources           6.7    6.9    7.0    7.0    7.4
Allegheny Power              6.7    7.3    7.8    7.9    8.3
Minnesota Power & Light      5.6    6.1    7.2    7.0    7.5
Iowa-Ill Gas & Electric      7.1    7.5    8.5    7.8    8.0
Pennsylvania Power & Light   7.2    7.6    7.7    7.4    7.1
Oklahoma Gas & Electric      6.1    6.7    7.4    6.7    6.8
Wisconsin Energy             5.1    5.7    6.0    5.7    5.9
Green Mountain Power         7.1    7.4    7.8    7.8    8.3
;

proc distance data=stock method=dcorr out=distdcorr;
   var interval(div_1986 div_1987 div_1988 div_1989 div_1990);
   id company;
run;

proc print data=distdcorr;
   id company;
   title2 'Distance Matrix for 15 Utility Stocks';
run;
title2;

ods graphics on;

/* compute pseudo statistic versus number of clusters and create plot */
proc cluster data=distdcorr method=ward pseudo plots(only)=(psf dendrogram);
   id company;
run;

/* compute pseudo statistic versus number of clusters and create plot */
proc cluster data=distdcorr method=average pseudo plots(only)=(psf dendrogram);
   id company;
run;
```

Output 43.2.1 Distance Matrix Based on the DCORR Coefficient

Stock Dividends
Distance Matrix for 15 Utility Stocks

Company	Cincinnati_G_E	Texas_Utilities	Detroit_Edison	Orange___Rockland_Utilities	Kentucky_Utilities
Cincinnati G&E	0.00000	.	.	.	.
Texas Utilities	0.82056	0.00000	.	.	.
Detroit Edison	0.40511	0.65453	0.00000	.	.
Orange & Rockland Utilities	1.35380	0.88583	1.27306	0.00000	.
Kentucky Utilities	1.35581	0.92539	1.29382	0.12268	0.00000
Kansas Power & Light	1.34227	0.94371	1.31696	0.19905	0.12874
Union Electric	0.98516	0.29043	0.89048	0.68798	0.71824
Dominion Resources	1.32945	0.96853	1.29016	0.33290	0.21510
Allegheny Power	1.30492	0.81666	1.24565	0.17844	0.15759

Company	Kansas_Power___Light	Union_Electric	Dominion_Resources	Allegheny_Power
Cincinnati G&E	.	.	.	.
Texas Utilities	.	.	.	.
Detroit Edison	.	.	.	.
Orange & Rockland Utilities	.	.	.	.
Kentucky Utilities	.	.	.	.
Kansas Power & Light	0.00000	.	.	.
Union Electric	0.72082	0.00000	.	.
Dominion Resources	0.24189	0.76587	0.00000	.
Allegheny Power	0.17029	0.58452	0.27819	0.00000

Company	Minnesota_Power___Light	Iowa_Ill_Gas___Electric	Pennsylvania_Power___Light
Cincinnati G&E	.	.	.
Texas Utilities	.	.	.
Detroit Edison	.	.	.
Orange & Rockland Utilities	.	.	.
Kentucky Utilities	.	.	.
Kansas Power & Light	.	.	.
Union Electric	.	.	.
Dominion Resources	.	.	.
Allegheny Power	.	.	.

Company	Oklahoma_Gas___Electric	Wisconsin_Energy	Green_Mountain_Power
Cincinnati G&E	.	.	.
Texas Utilities	.	.	.
Detroit Edison	.	.	.
Orange & Rockland Utilities	.	.	.
Kentucky Utilities	.	.	.
Kansas Power & Light	.	.	.
Union Electric	.	.	.
Dominion Resources	.	.	.
Allegheny Power	.	.	.

Output 43.2.1 *continued*

Stock Dividends
Distance Matrix for 15 Utility Stocks

Company	Cincinnati_G_E	Texas_Utilities	Detroit_Edison	Orange___Rockland_Utilities	Kentucky_Utilities
Minnesota Power & Light	1.24069	0.74082	1.20432	0.32581	0.30462
Iowa-Ill Gas & Electric	1.04924	0.43100	0.97616	0.61166	0.61760
Pennsylvania Power & Light	0.74931	0.37821	0.44256	1.03566	1.08878
Oklahoma Gas & Electric	1.00604	0.30141	0.86200	0.68021	0.70259
Wisconsin Energy	1.17988	0.54830	1.03081	0.45013	0.47184
Green Mountain Power	1.30397	0.88063	1.27176	0.26948	0.17909

Company	Kansas_Power___Light	Union_Electric	Dominion_Resources	Allegheny_Power
Minnesota Power & Light	0.27231	0.48372	0.35733	0.15615
Iowa-Ill Gas & Electric	0.61736	0.16923	0.63545	0.47900
Pennsylvania Power & Light	1.12876	0.63285	1.14354	1.02358
Oklahoma Gas & Electric	0.73158	0.17122	0.72977	0.58391
Wisconsin Energy	0.53381	0.37405	0.51969	0.37522
Green Mountain Power	0.15377	0.64869	0.17360	0.13958

Company	Minnesota_Power___Light	Iowa_Ill_Gas___Electric	Pennsylvania_Power___Light
Minnesota Power & Light	0.00000	.	.
Iowa-Ill Gas & Electric	0.36368	0.00000	.
Pennsylvania Power & Light	0.99384	0.75596	0.00000
Oklahoma Gas & Electric	0.50744	0.19673	0.60216
Wisconsin Energy	0.36319	0.30259	0.76085
Green Mountain Power	0.19370	0.52083	1.09269

Company	Oklahoma_Gas___Electric	Wisconsin_Energy	Green_Mountain_Power
Minnesota Power & Light	.	.	.
Iowa-Ill Gas & Electric	.	.	.
Pennsylvania Power & Light	.	.	.
Oklahoma Gas & Electric	0.00000	.	.
Wisconsin Energy	0.28070	0.00000	.
Green Mountain Power	0.64175	0.44814	0

Output 43.2.2 Pseudo F versus Number of Clusters When METHOD=WARD

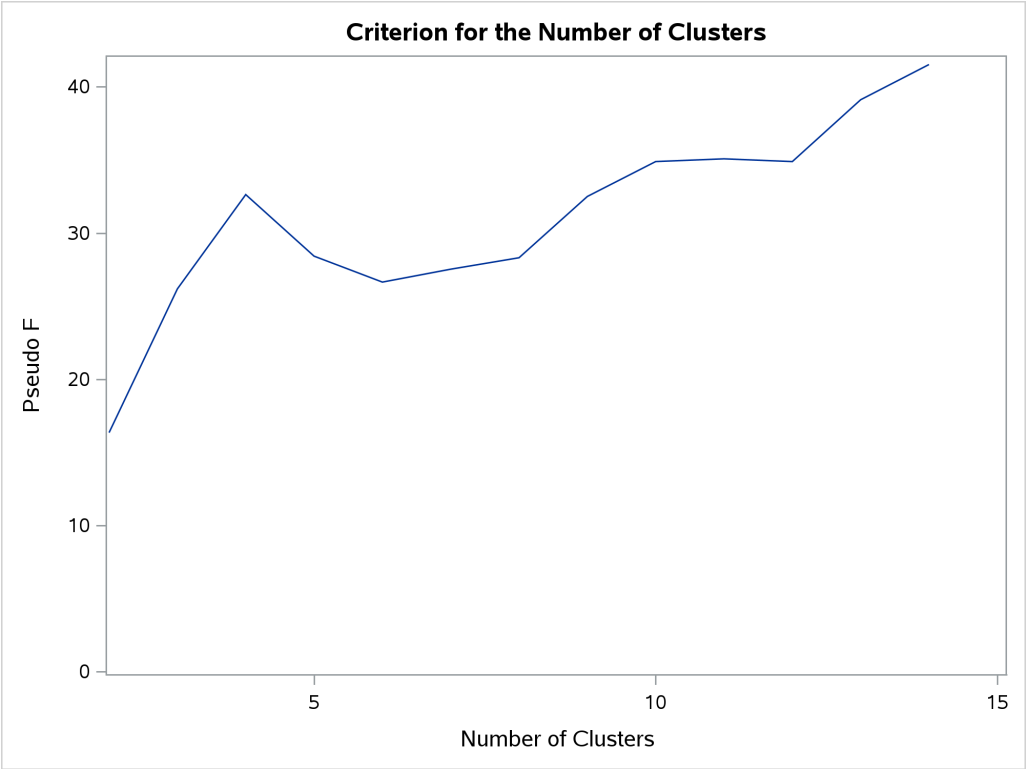

Output 43.2.3 Pseudo F versus Number of Clusters When METHOD=AVERAGE

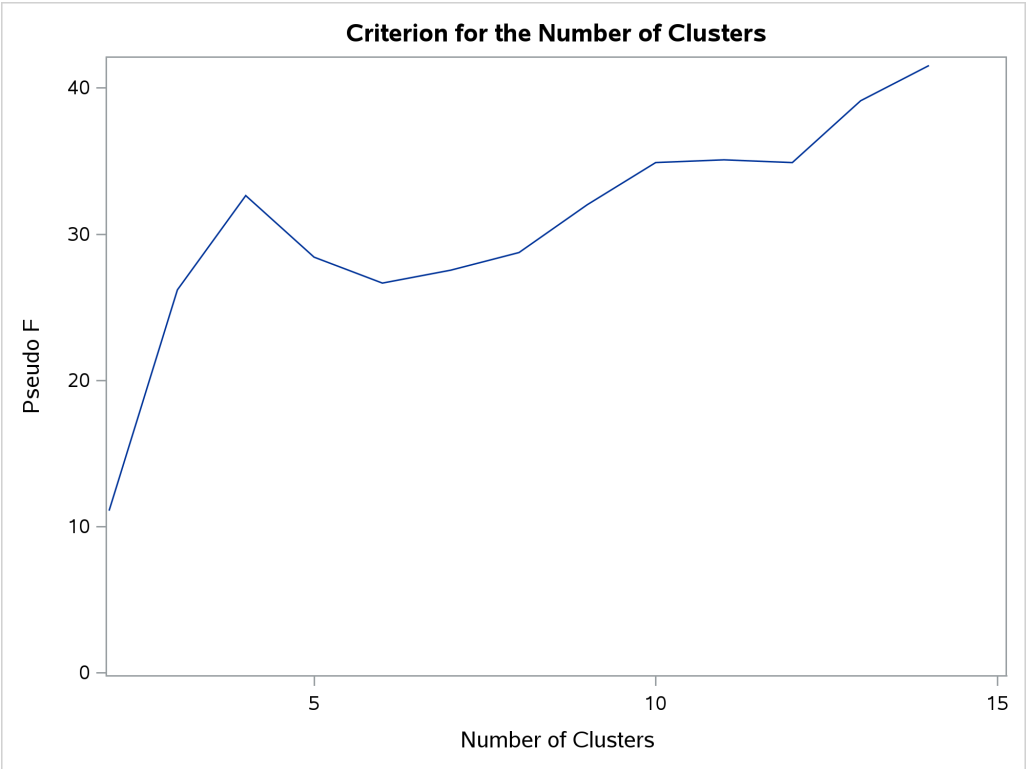

Output 43.2.4 Dendrogram of Semipartial R-Square Values When METHOD=WARD

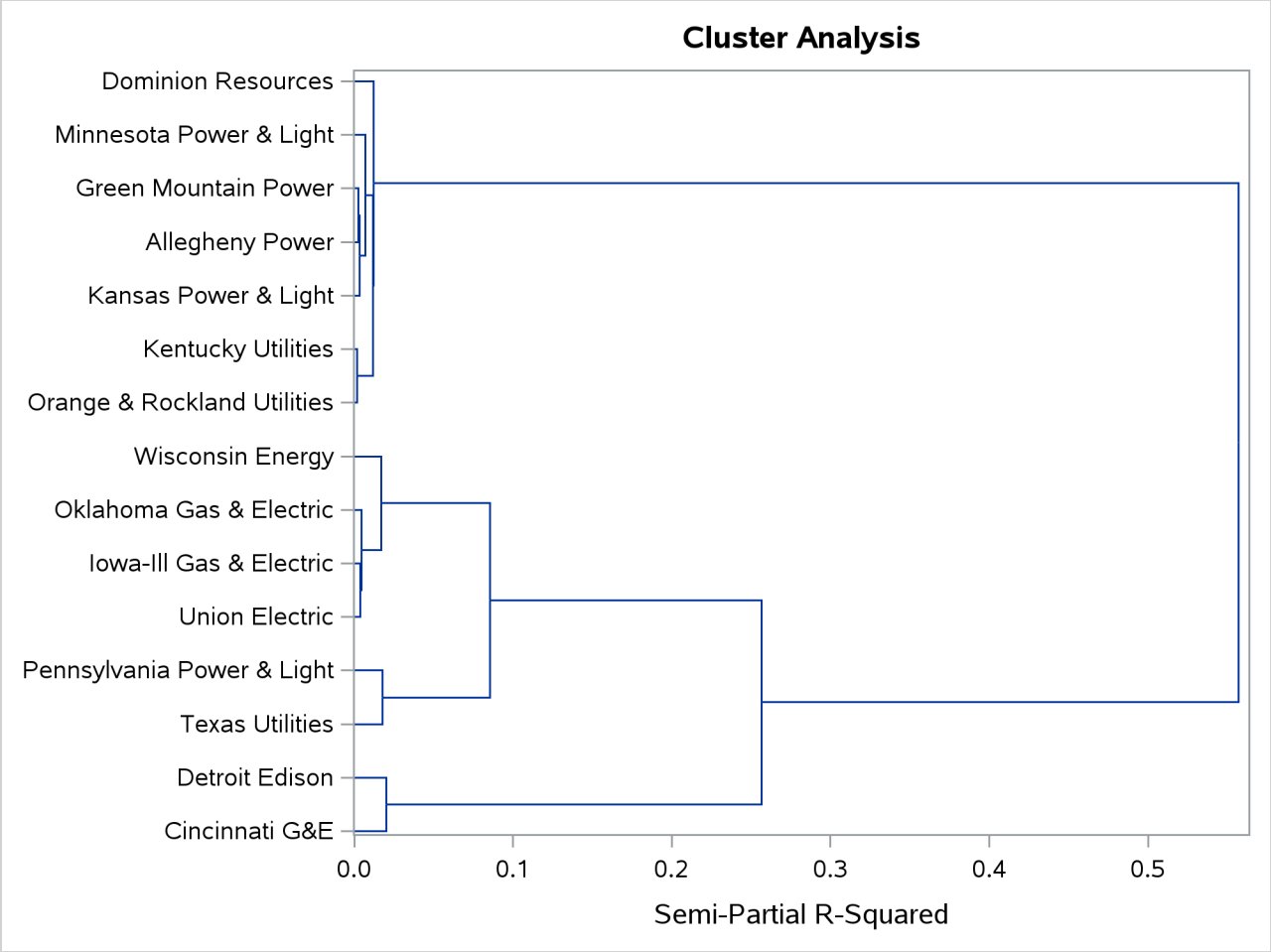

Output 43.2.5 Dendrogram of Average Distance between Clusters When METHOD=AVERAGE

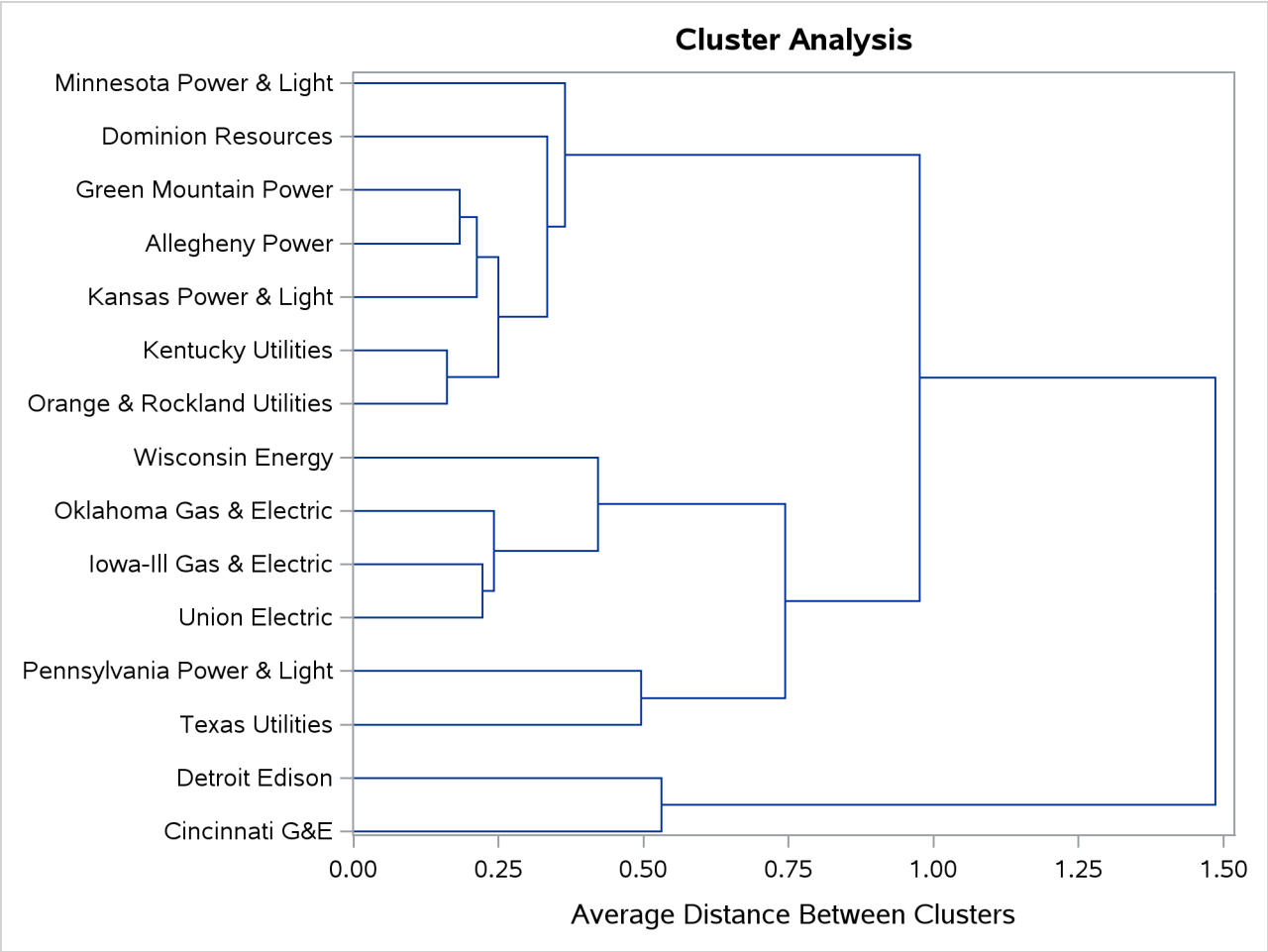

References

Anderberg, M. R. (1973). *Cluster Analysis for Applications*. New York: Academic Press.

Gower, J. C., and Legendre, P. (1986). "Metric and Euclidean Properties of Dissimilarity Coefficients." *Journal of Classification* 3:5–48.

Hand, D. J., Daly, F., Lunn, A. D., McConway, K. J., and Ostrowski, E. (1994). *A Handbook of Small Data Sets*. London: Chapman & Hall.

Lance, G. N., and Williams, W. T. (1967). "Mixed-Data Classificatory Programs, Part I: Agglomerative Systems." *Australian Computer Journal* 1:15–20.

Sneath, P. H. A., and Sokal, R. R. (1973). *Numerical Taxonomy*. San Francisco: W. H. Freeman.

Chapter 44
The FACTOR Procedure

Contents

Overview: FACTOR Procedure . . . **2860**
- Background . . . 2861
- Outline of Use . . . 2863

Getting Started: FACTOR Procedure . . . **2868**

Syntax: FACTOR Procedure . . . **2873**
- PROC FACTOR Statement . . . 2873
- BY Statement . . . 2892
- FREQ Statement . . . 2893
- PARTIAL Statement . . . 2893
- PATHDIAGRAM Statement . . . 2893
- PRIORS Statement . . . 2899
- VAR Statement . . . 2899
- WEIGHT Statement . . . 2899

Details: FACTOR Procedure . . . **2900**
- Input Data Set . . . 2900
- Output Data Sets . . . 2902
- Number of Factors to Retain . . . 2904
- Variable Weights and Variance Explained . . . 2905
- Simplicity Functions for Rotations . . . 2906
- Confidence Intervals and the Salience of Factor Loadings . . . 2907
- Factor Scores . . . 2908
- Heywood Cases and Other Anomalies about Communality Estimates . . . 2909
- Missing Values . . . 2910
- Cautions . . . 2911
- Time Requirements . . . 2911
- Displayed Output . . . 2912
- ODS Table Names . . . 2916
- ODS Graphics . . . 2918

Examples: FACTOR Procedure . . . **2919**
- Example 44.1: Principal Component Analysis . . . 2919
- Example 44.2: Principal Factor Analysis . . . 2925
- Example 44.3: Determining the Number of Factors to Retain . . . 2944
- Example 44.4: Maximum Likelihood Factor Analysis . . . 2948
- Example 44.5: Using Confidence Intervals to Locate Salient Factor Loadings . . . 2955
- Example 44.6: Creating Path Diagrams for Factor Solutions . . . 2959

References . . . **2965**

Overview: FACTOR Procedure

The FACTOR procedure performs a variety of common factor and component analyses and rotations. Input can be multivariate data, a correlation matrix, a covariance matrix, a factor pattern, or a matrix of scoring coefficients. The procedure can factor either the correlation or covariance matrix, and you can save most results in an output data set.

PROC FACTOR can process output from other procedures. For example, it can rotate the canonical coefficients from multivariate analyses in the GLM procedure.

The methods for factor extraction are principal component analysis, principal factor analysis, iterated principal factor analysis, unweighted least squares factor analysis, maximum likelihood (canonical) factor analysis, alpha factor analysis, image component analysis, and Harris component analysis. A variety of methods for prior communality estimation is also available.

Specific methods for orthogonal rotation are varimax, quartimax, biquartimax, equamax, parsimax, and factor parsimax. Oblique versions of these methods are also available. In addition, quartimin, biquartimin, and covarimin methods for (direct) oblique rotation are available. General methods for orthogonal rotation are orthomax with user-specified gamma, Crawford-Ferguson family with user-specified weights on variable parsimony and factor parsimony, and generalized Crawford-Ferguson family with user-specified weights. General methods for oblique rotation are direct oblimin with user-specified tau, Crawford-Ferguson family with user-specified weights on variable parsimony and factor parsimony, generalized Crawford-Ferguson family with user-specified weights, promax with user-specified exponent, Harris-Kaiser case II with user-specified exponent, and Procrustes with a user-specified target pattern.

Output includes means, standard deviations, correlations, Kaiser's measure of sampling adequacy, eigenvalues, a scree plot, eigenvectors, prior and final communality estimates, the unrotated factor pattern, residual and partial correlations, the rotated primary factor pattern, the primary factor structure, interfactor correlations, the reference structure, reference axis correlations, the variance explained by each factor both ignoring and eliminating other factors, plots of both rotated and unrotated factors, squared multiple correlation of each factor with the variables, standard error estimates, confidence limits, coverage displays, scoring coefficients, and path diagrams.

The FACTOR procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Any topics that are not given explicit references are discussed in Mulaik (1972) or Harman (1976).

Background

See Chapter 98, "The PRINCOMP Procedure," for a discussion of principal component analysis. See Chapter 33, "The CALIS Procedure," for a discussion of confirmatory factor analysis.

Common factor analysis was invented by Spearman (1904). Kim and Mueller (1978a, b) provide a very elementary discussion of the common factor model. Gorsuch (1974) presents a broad survey of factor analysis, and Gorsuch (1974) and Cattell (1978) are useful as guides to practical research methodology. Harman (1976) gives a lucid discussion of many of the more technical aspects of factor analysis, especially oblique rotation. Morrison (1976) and Mardia, Kent, and Bibby (1979) provide excellent statistical treatments of common factor analysis. Mulaik (1972) provides the most thorough and authoritative general reference on factor analysis and is highly recommended to anyone familiar with matrix algebra. Stewart (1981) gives a nontechnical presentation of some issues to consider when deciding whether or not a factor analysis might be appropriate.

A frequent source of confusion in the field of factor analysis is the term *factor*. It sometimes refers to a hypothetical, unobservable variable, as in the phrase *common factor*. In this sense, *factor analysis* must be distinguished from component analysis since a component is an observable linear combination. *Factor* is also used in the sense of *matrix factor,* in that one matrix is a factor of a second matrix if the first matrix multiplied by its transpose equals the second matrix. In this sense, *factor analysis* refers to all methods of data analysis that use matrix factors, including component analysis and common factor analysis.

A *common factor* is an unobservable, hypothetical variable that contributes to the variance of at least two of the observed variables. The unqualified term "factor" often refers to a common factor. A *unique factor* is an unobservable, hypothetical variable that contributes to the variance of only one of the observed variables. The model for common factor analysis posits one unique factor for each observed variable.

The equation for the common factor model is

$$y_{ij} = x_{i1}b_{1j} + x_{i2}b_{2j} + \cdots + x_{iq}b_{qj} + e_{ij}$$

where

y_{ij}	is the value of the ith observation on the jth variable
x_{ik}	is the value of the ith observation on the kth common factor
b_{kj}	is the regression coefficient of the kth common factor for predicting the jth variable
e_{ij}	is the value of the ith observation on the jth unique factor
q	is the number of common factors

It is assumed, for convenience, that all variables have a mean of 0. In matrix terms, these equations reduce to

$$\mathbf{Y} = \mathbf{XB} + \mathbf{E}$$

In the preceding equation, $\mathbf{X}$ is the matrix of factor scores, and $\mathbf{B}'$ is the factor pattern.

There are two critical assumptions:

- The unique factors are uncorrelated with each other.
- The unique factors are uncorrelated with the common factors.

In principal component analysis, the residuals are generally correlated with each other. In common factor analysis, the unique factors play the role of residuals and are defined to be uncorrelated both with each other and with the common factors. Each common factor is assumed to contribute to at least two variables; otherwise, it would be a unique factor.

When the factors are initially extracted, it is also assumed, for convenience, that the common factors are uncorrelated with each other and have unit variance. In this case, the common factor model implies that the covariance s_{jk} between the *j*th and *k*th variables, $j \neq k$, is given by

$$s_{jk} = b_{1j}b_{1k} + b_{2j}b_{2k} + \cdots + b_{qj}b_{qk}$$

or

$$\mathbf{S} = \mathbf{B}'\mathbf{B} + \mathbf{U}^2$$

where $\mathbf{S}$ is the covariance matrix of the observed variables, and $\mathbf{U}^2$ is the diagonal covariance matrix of the unique factors.

If the original variables are standardized to unit variance, the preceding formula yields correlations instead of covariances. It is in this sense that common factors explain the correlations among the observed variables. When considering the diagonal elements of standardized $\mathbf{S}$, the variance of the *j*th variable is expressed as

$$s_{jj} = 1 = b_{1j}^2 + b_{2j}^2 + \cdots + b_{qj}^2 + [\mathbf{U}^2]_{jj}$$

where $b_{1j}^2 + b_{2j}^2 + \cdots + b_{qj}^2$ and $[\mathbf{U}^2]_{jj}$ are the communality and uniqueness, respectively, of the *j*th variable. The communality represents the extent of the overlap with the common factors. In other words, it is the proportion of variance accounted for by the common factors.

The difference between the correlation predicted by the common factor model and the actual correlation is the *residual correlation*. A good way to assess the goodness of fit of the common factor model is to examine the residual correlations.

The common factor model implies that the partial correlations among the variables, removing the effects of the common factors, must all be zero. When the common factors are removed, only unique factors, which are by definition uncorrelated, remain.

The assumptions of common factor analysis imply that the common factors are, in general, not linear combinations of the observed variables. In fact, even if the data contain measurements on the entire population of observations, you cannot compute the scores of the observations on the common factors. Although the common factor scores cannot be computed directly, they can be estimated in a variety of ways.

The problem of factor score indeterminacy has led several factor analysts to propose methods yielding components that can be considered approximations to common factors. Since these components are defined as linear combinations, they are computable. The methods include Harris component analysis and image component analysis. The advantage of producing determinate component scores is offset by the fact that, even if the data fit the common factor model perfectly, component methods do not generally recover the correct factor solution. You should not use any type of component analysis if you really want a common factor analysis (Dziuban and Harris 1973; Lee and Comrey 1979).

After the factors are estimated, it is necessary to interpret them. Interpretation usually means assigning to each common factor a name that reflects the *salience* of the factor in predicting each of the observed

variables—that is, the coefficients in the pattern matrix corresponding to the factor. Factor interpretation is a subjective process. It can sometimes be made less subjective by *rotating* the common factors—that is, by applying a nonsingular linear transformation. A rotated pattern matrix in which all the coefficients are close to 0 or ± 1 is easier to interpret than a pattern with many intermediate elements. Therefore, most rotation methods attempt to optimize a simplicity function of the rotated pattern matrix that measures, in some sense, how close the elements are to 0 or ± 1. Because the loading estimates are subject to sampling variability, it is useful to obtain the standard error estimates for the loadings for assessing the uncertainty due to random sampling. Notice that the *salience* of a factor loading refers to the magnitude of the loading, while statistical *significance* refers to the statistical evidence against a particular hypothetical value. A loading significantly different from 0 does not automatically mean it must be salient. For example, if salience is defined as a magnitude larger than 0.4 while the entire 95% confidence interval for a loading lies between 0.1 and 0.3, the loading is statistically significant larger than 0 but it is not salient. Under the maximum likelihood method, you can obtain standard errors and confidence intervals for judging the salience of factor loadings.

After the initial factor extraction, the common factors are uncorrelated with each other. If the factors are rotated by an *orthogonal transformation,* the rotated factors are also uncorrelated. If the factors are rotated by an *oblique transformation,* the rotated factors become correlated. Oblique rotations often produce more useful patterns than do orthogonal rotations. However, a consequence of correlated factors is that there is no single unambiguous measure of the importance of a factor in explaining a variable. Thus, for oblique rotations, the pattern matrix does not provide all the necessary information for interpreting the factors; you must also examine the *factor structure* and the *reference structure*.

Rotating a set of factors does not change the statistical explanatory power of the factors. You cannot say that any rotation is better than any other rotation from a statistical point of view; all rotations, orthogonal or oblique, are equally good statistically. Therefore, the choice among different rotations must be based on nonstatistical grounds. For most applications, the preferred rotation is that which is most easily interpretable, or most compatible with substantive theories.

If two rotations give rise to different interpretations, those two interpretations must not be regarded as conflicting. Rather, they are two different ways of looking at the same thing, two different points of view in the common-factor space. Any conclusion that depends on one and only one rotation being correct is invalid.

Outline of Use

Principal Component Analysis

One important type of analysis performed by the FACTOR procedure is principal component analysis. The following statements result in a principal component analysis:

```
proc factor;
run;
```

The output includes all the eigenvalues and the pattern matrix for eigenvalues greater than one.

Most applications require additional output. For example, you might want to compute principal component scores for use in subsequent analyses or obtain a graphical aid to help decide how many components to keep. You can save the results of the analysis in a permanent SAS data library by using the OUTSTAT= option. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. Assuming that your SAS data library has the libref save and that the data are in a SAS data set called raw, you could do a principal component analysis as follows:

```
proc factor data=raw method=principal scree mineigen=0 score
     outstat=save.fact_all;
run;
```

The SCREE option produces a plot of the eigenvalues that is helpful in deciding how many components to use. Alternatively, you can use the PLOTS=SCREE option to produce high-quality scree plots. The MINEIGEN=0 option causes all components with variance greater than zero to be retained. The SCORE option requests that scoring coefficients be computed. The OUTSTAT= option saves the results in a specially structured SAS data set. The name of the data set, in this case fact_all, is arbitrary. To compute principal component scores, use the SCORE procedure:

```
proc score data=raw score=save.fact_all out=save.scores;
run;
```

The SCORE procedure uses the data and the scoring coefficients that are saved in save.fact_all to compute principal component scores. The component scores are placed in variables named Factor1, Factor2, . . . , Factor*n* and are saved in the data set save.scores. If you know ahead of time how many principal components you want to use, you can obtain the scores directly from PROC FACTOR by specifying the NFACTORS=*n* and OUT= options. To get scores from three principal components, specify the following:

```
proc factor data=raw method=principal
     nfactors=3 out=save.scores;
run;
```

To plot the scores for the first three components, use the SGSCATTER procedure:

```
proc sgscatter;
   matrix factor1-factor3;
run;
```

Principal Factor Analysis

The simplest and computationally most efficient method of common factor analysis is principal factor analysis, which is obtained in the same way as principal component analysis except for the use of the PRIORS= option. The usual form of the initial analysis is as follows:

```
proc factor data=raw method=principal scree
     mineigen=0 priors=smc outstat=save.fact_all;
run;
```

The squared multiple correlations (SMC) of each variable with all the other variables are used as the prior communality estimates. If your correlation matrix is singular, you should specify PRIORS=MAX instead of PRIORS=SMC. The SCREE and MINEIGEN= options serve the same purpose as in the preceding principal component analysis. Saving the results with the OUTSTAT= option enables you to examine the eigenvalues and scree plot before deciding how many factors to rotate and to try several different rotations without re-extracting the factors. The OUTSTAT= data set is automatically marked TYPE=FACTOR, so the FACTOR procedure realizes that it contains statistics from a previous analysis instead of raw data.

After looking at the eigenvalues to estimate the number of factors, you can try some rotations. Two and three factors can be rotated with the following statements:

```
proc factor data=save.fact_all method=principal n=2
     rotate=promax reorder score outstat=save.fact_2;
run;
proc factor data=save.fact_all method=principal n=3
     rotate=promax reorder score outstat=save.fact_3;
run;
```

The output data set from the previous run is used as input for these analyses. The options N=2 and N=3 specify the number of factors to be rotated. The specification ROTATE=PROMAX requests a promax rotation, which has the advantage of providing both orthogonal and oblique rotations with only one invocation of PROC FACTOR. The REORDER option causes the variables to be reordered in the output so that variables associated with the same factor appear next to each other.

You can now compute and plot factor scores for the two-factor promax-rotated solution as follows:

```
proc score data=raw score=save.fact_2 out=save.scores;
run;

proc sgplot;
   scatter y=factor2 x=factor1;
run;
```

Maximum Likelihood Factor Analysis

Although principal factor analysis is perhaps the most commonly used method of common factor analysis, most statisticians prefer maximum likelihood (ML) factor analysis (Lawley and Maxwell 1971). The ML method of estimation has desirable asymptotic properties (Bickel and Doksum 1977) and produces better estimates than principal factor analysis in large samples. You can test hypotheses about the number of common factors by using the ML method. You can also obtain standard error and confidence interval estimates for many classes of rotated or unrotated factor loadings, factor correlations, and structure loadings under the ML theory.

The unrotated ML solution is equivalent to Rao's canonical factor solution (Rao 1955) and Howe's solution maximizing the determinant of the partial correlation matrix (Morrison 1976). Thus, as a descriptive method, ML factor analysis does not require a multivariate normal distribution. The validity of Bartlett's χ^2 test for the number of factors does require approximate normality plus additional regularity conditions that are usually satisfied in practice (Geweke and Singleton 1980). Bartlett's test of sphericity in the context of factor analysis is equivalent to Bartlett's χ^2 test for zero common factors. This test is routinely displayed in the maximum likelihood factor analysis output.

Lawley and Maxwell (1971) derive the standard error formulas for unrotated loadings, while Archer and Jennrich (1973) and Jennrich (1973, 1974) derive the standard error formulas for several classes of rotated solutions. Extended formulas for computing standard errors in various situations appear in Browne et al. (2010); Hayashi and Yung (1999); Yung and Hayashi (2001). A combination of these methods is used in PROC FACTOR to compute standard errors in an efficient manner. Confidence intervals are computed by using the asymptotic normality of the estimates. To ensure that the confidence intervals fall within the admissible parameter range, transformation methods due to Browne (1982) are used. The validity of the standard error estimates and confidence limits requires the assumptions of multivariate normality and a fixed number of factors.

The ML method is more computationally demanding than principal factor analysis for two reasons. First, the communalities are estimated iteratively, and each iteration takes about as much computer time as principal

factor analysis. The number of iterations typically ranges from about five to twenty. Second, if you want to extract different numbers of factors, as is often the case, you must run the FACTOR procedure once for each number of factors. Therefore, an ML analysis can take 100 times as long as a principal factor analysis. This does not include the time for computing standard error estimates, which is even more computationally demanding. For analyses with fewer than 35 variables, the computing time for the ML method, including the computation of standard errors, usually ranges from a few seconds to well under a minute. This seems to be a reasonable performance.

You can use principal factor analysis to get a rough idea of the number of factors before doing an ML analysis. If you think that there are between one and three factors, you can use the following statements for the ML analysis:

```
proc factor data=raw method=ml n=1
     outstat=save.fact1;
run;
proc factor data=raw method=ml n=2 rotate=promax
     outstat=save.fact2;
run;
proc factor data=raw method=ml n=3 rotate=promax
     outstat=save.fact3;
run;
```

The output data sets can be used for trying different rotations, computing scoring coefficients, or restarting the procedure in case it does not converge within the allotted number of iterations.

If you can determine how many factors should be retained before an analysis, as in the following statements, you can get the standard errors and confidence limits to aid interpretations for the ML analysis:

```
proc factor data=raw method=ml n=3 rotate=quartimin se
   cover=.4;
run;
```

In this analysis, you specify the quartimin rotation in the ROTATE= option. The SE option requests the computation of standard error estimates. In the COVER= option, you require absolute values of 0.4 or greater in order for loadings to be salient. In the output of coverage display, loadings that are salient would have their entire confidence intervals spanning beyond the 0.4 mark (or the –0.4 mark in the opposite direction). Only those salient loadings should be used for interpreting the factors. See the section "Confidence Intervals and the Salience of Factor Loadings" on page 2907 for more details.

The ML method cannot be used with a singular correlation matrix, and it is especially prone to Heywood cases. See the section "Heywood Cases and Other Anomalies about Communality Estimates" on page 2909 for a discussion of Heywood cases. If you have problems with ML, the best alternative is to use the METHOD=ULS option for unweighted least squares factor analysis.

Factor Rotation

After the initial factor extraction, the factors are uncorrelated with each other. If the factors are rotated by an *orthogonal transformation,* the rotated factors are also uncorrelated. If the factors are rotated by an *oblique transformation,* the rotated factors become correlated. Oblique rotations often produce more useful patterns than orthogonal rotations do. However, a consequence of correlated factors is that there is no single unambiguous measure of the importance of a factor in explaining a variable. Thus, for oblique rotations, the pattern matrix does not provide all the necessary information for interpreting the factors; you must also examine the *factor structure* and the *reference structure*.

Nowadays, most rotations are done analytically. There are many choices for orthogonal and oblique rotations. An excellent summary of a wide class of analytic rotations is in Crawford and Ferguson (1970). The Crawford-Ferguson family of orthogonal rotations includes the orthomax rotation as a subclass and the popular varimax rotation as a special case. To illustrate these relationships, the following four specifications for orthogonal rotations with different ROTATE= options will give the same results for a data set with nine observed variables:

```
    /* Orthogonal Crawford-Ferguson Family with
       variable parsimony weight = nvar - 1 = 8, and
       factor parsimony weight = 1  */
proc factor data=raw n=3 rotate=orthcf(8,1);
run;

    /* Orthomax without the GAMMA= option */
proc factor data=raw n=3 rotate=orthomax(1);
run;

    /* Orthomax with the GAMMA= option */
proc factor data=raw n=3 rotate=orthomax gamma=1;
run;

    /* Varimax */
proc factor data=raw n=3 rotate=varimax;
run;
```

You can also get the oblique versions of the varimax in two equivalent ways:

```
    /* Oblique Crawford-Ferguson Family with
       variable parsimony weight = nvar - 1 = 8, and
       factor parsimony weight = 1; */
proc factor data=raw n=3 rotate=oblicf(8,1);
run;

    /* Oblique Varimax */
proc factor data=raw n=3 rotate=obvarimax;
run;
```

Jennrich (1973) proposes a generalized Crawford-Ferguson family that includes the Crawford-Ferguson family and the (direct) oblimin family (see Harman 1976) as subclasses. The better-known quartimin rotation is a special case of the oblimin class, and hence a special case of the generalized Crawford-Ferguson family. For example, the following four specifications of oblique rotations are equivalent:

```
   /* Oblique generalized Crawford-Ferguson Family
      with weights 0, 1, 0, -1 */
proc factor data=raw n=3 rotate=obligencf(0,1,0,-1);
run;

   /* Oblimin family without the TAU= option */
proc factor data=raw n=3 rotate=oblimin(0);
run;

   /* Oblimin family with the TAU= option */
proc factor data=raw n=3 rotate=oblimin tau=0;
```

```
run;

   /* Quartimin */
proc factor data=raw n=3 rotate=quartimin;
run;
```

In addition to the generalized Crawford-Ferguson family, the available oblique rotation methods in PROC FACTOR include Harris-Kaiser, promax, and Procrustes. See the section "Simplicity Functions for Rotations" on page 2906 for details about the definitions of various rotations. See Harman (1976) and Mulaik (1972) for further information.

Getting Started: FACTOR Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/facgs1.sas.)

The following example demonstrates how you can use the FACTOR procedure to perform common factor analysis and factor rotation.

In this example, 103 police officers were rated by their supervisors on 14 scales (variables). You conduct a common factor analysis on these variables to see what latent factors are operating behind these ratings. The overall rating variable is excluded from the factor analysis.

The following DATA step creates the SAS data set jobratings:

```
options validvarname=any;
data jobratings;
   input ('Communication Skills'n
      'Problem Solving'n
      'Learning Ability'n
      'Judgment under Pressure'n
      'Observational Skills'n
      'Willingness to Confront Problems'n
      'Interest in People'n
      'Interpersonal Sensitivity'n
      'Desire for Self-Improvement'n
      'Appearance'n
      'Dependability'n
      'Physical Ability'n
      'Integrity'n
      'Overall Rating'n) (1.);
   datalines;
26838853879867
74758876857667
56757863775875
67869777988997
99997798878888
89897899888799
89999889899798
87794798468886
```

```
   ... more lines ...

99899899899899
76656399567486
;
```

The following statements invoke the FACTOR procedure:

```
proc factor data=jobratings(drop='Overall Rating'n) priors=smc
   rotate=varimax;
run;
```

The DATA= option in PROC FACTOR specifies the SAS data set jobratings as the input data set. The DROP= option drops the Overall Rating variable from the analysis. To conduct a common factor analysis, you need to set the prior communality estimate to less than one for each variable. Otherwise, the factor solution would simply be a recast of the principal components solution, in which "factors" are linear combinations of observed variables. However, in the common factor model you always assume that observed variables are functions of underlying factors. In this example, the PRIORS= option specifies that the squared multiple correlations (SMC) of each variable with all the other variables are used as the prior communality estimates. Note that squared multiple correlations are usually less than one. By default, the principal factor extraction is used if the METHOD= option is not specified. To facilitate interpretations, the ROTATE= option specifies the VARIMAX orthogonal factor rotation to be used.

The output from the factor analysis is displayed in Figure 44.1 through Figure 44.5.

As displayed in Figure 44.1, the prior communality estimates are set to the squared multiple correlations. Figure 44.1 also displays the table of eigenvalues (the variances of the principal factors) of the reduced correlation matrix. Each row of the table pertains to a single eigenvalue. Following the column of eigenvalues are three measures of each eigenvalue's relative size and importance. The first of these displays the difference between the eigenvalue and its successor. The last two columns display the individual and cumulative proportions that the corresponding factor contributes to the total variation. The last line displayed in Figure 44.1 states that three factors are retained, as determined by the PROPORTION criterion.

Figure 44.1 Table of Eigenvalues from PROC FACTOR

The FACTOR Procedure
Initial Factor Method: Principal Factors

Prior Communality Estimates: SMC						
Communication Skills	Problem Solving	Learning Ability	Judgment under Pressure	Observational Skills	Willingness to Confront Problems	Interest in People
0.62981394	0.58657431	0.61009871	0.63766021	0.67187583	0.64779805	0.75641519

Interpersonal Sensitivity	Desire for Self-Improvement	Appearance	Dependability	Physical Ability	Integrity
0.75584891	0.57460176	0.45505304	0.63449045	0.42245324	0.68195454

Figure 44.1 *continued*

Eigenvalues of the Reduced Correlation Matrix: Total = 8.06463816 Average = 0.62035678				
	Eigenvalue	Difference	Proportion	Cumulative
1	6.17760549	4.71531946	0.7660	0.7660
2	1.46228602	0.90183348	0.1813	0.9473
3	0.56045254	0.28093933	0.0695	1.0168
4	0.27951322	0.04766016	0.0347	1.0515
5	0.23185305	0.16113428	0.0287	1.0802
6	0.07071877	0.07489624	0.0088	1.0890
7	-.00417747	0.03387533	-0.0005	1.0885
8	-.03805279	0.04776534	-0.0047	1.0838
9	-.08581814	0.02438060	-0.0106	1.0731
10	-.11019874	0.01452741	-0.0137	1.0595
11	-.12472615	0.02356465	-0.0155	1.0440
12	-.14829080	0.05823605	-0.0184	1.0256
13	-.20652684		-0.0256	1.0000

3 factors will be retained by the PROPORTION criterion.

Figure 44.2 displays the initial factor pattern matrix. The factor pattern matrix represents standardized regression coefficients for predicting the variables by using the extracted factors. Because the initial factors are uncorrelated, the pattern matrix is also equal to the correlations between variables and the common factors.

Figure 44.2 Factor Pattern Matrix from PROC FACTOR

Factor Pattern			
	Factor1	Factor2	Factor3
Communication Skills	0.75441	0.07707	-0.25551
Problem Solving	0.68590	0.08026	-0.34788
Learning Ability	0.65904	0.34808	-0.25249
Judgment under Pressure	0.73391	-0.21405	-0.23513
Observational Skills	0.69039	0.45292	0.10298
Willingness to Confront Problems	0.66458	0.47460	0.09210
Interest in People	0.70770	-0.53427	0.10979
Interpersonal Sensitivity	0.64668	-0.61284	-0.07582
Desire for Self-Improvement	0.73820	0.12506	0.09062
Appearance	0.57188	0.20052	0.16367
Dependability	0.79475	-0.04516	0.16400
Physical Ability	0.51285	0.10251	0.34860
Integrity	0.74906	-0.35091	0.18656

The pattern matrix suggests that Factor1 represents general ability. All loadings for Factor1 in the Factor Pattern are at least 0.5. Factor2 consists of high positive loadings on certain task-related skills (Willingness to Confront Problems, Observational Skills, and Learning Ability) and high negative loadings on some interpersonal skills (Interpersonal Sensitivity, Interest in People, and Integrity). This factor measures individuals' relative strength in these skills. Theoretically, individuals with high positive scores on this factor would exhibit better task-related skills than interpersonal skills. Individuals with high negative scores

would exhibit better interpersonal skills than task-related skills. Individuals with scores near zero would have those skills balanced. Factor3 does not have a cluster of very high or very low factor loadings. Therefore, interpreting this factor is difficult.

Figure 44.3 displays the proportion of variance explained by each factor and the final communality estimates, including the total communality. The final communality estimates are the proportion of variance of the variables accounted for by the common factors. When the factors are orthogonal, the final communalities are calculated by taking the sum of squares of each row of the factor pattern matrix.

Figure 44.3 Variance Explained and Final Communality Estimates

Variance Explained by Each Factor		
Factor1	**Factor2**	**Factor3**
6.1776055	1.4622860	0.5604525

Final Communality Estimates: Total = 8.200344						
Communication Skills	**Problem Solving**	**Learning Ability**	**Judgment under Pressure**	**Observational Skills**	**Willingness to Confront Problems**	**Interest in People**
0.64036292	0.59791844	0.61924167	0.63972863	0.69237485	0.67538695	0.79833968

Interpersonal Sensitivity	**Desire for Self-Improvement**	**Appearance**	**Dependability**	**Physical Ability**	**Integrity**
0.79951357	0.56879171	0.39403630	0.66056907	0.39504805	0.71903222

Figure 44.4 displays the results of the VARIMAX rotation of the three extracted factors and the corresponding orthogonal transformation matrix. The rotated factor pattern matrix is calculated by postmultiplying the original factor pattern matrix (Figure 44.4) by the transformation matrix.

Figure 44.4 Transformation Matrix and Rotated Factor Pattern

Orthogonal Transformation Matrix			
	1	**2**	**3**
1	0.59125	0.59249	0.54715
2	-0.80080	0.51170	0.31125
3	0.09557	0.62219	-0.77701

Figure 44.4 *continued*

Rotated Factor Pattern			
	Factor1	Factor2	Factor3
Communication Skills	0.35991	0.32744	0.63530
Problem Solving	0.30802	0.23102	0.67058
Learning Ability	0.08679	0.41149	0.66512
Judgment under Pressure	0.58287	0.17901	0.51764
Observational Skills	0.05533	0.70488	0.43870
Willingness to Confront Problems	0.02168	0.69391	0.43978
Interest in People	0.85677	0.21422	0.13562
Interpersonal Sensitivity	0.86587	0.02239	0.22200
Desire for Self-Improvement	0.34498	0.55775	0.37242
Appearance	0.19319	0.54327	0.24814
Dependability	0.52174	0.54981	0.29337
Physical Ability	0.25445	0.57321	0.04165
Integrity	0.74172	0.38033	0.15567

The rotated factor pattern matrix is somewhat simpler to interpret. If a magnitude of at least 0.5 is required to indicate a salient variable-factor relationship, Factor1 now represents interpersonal skills (Interpersonal Sensitivity, Interest in People, Integrity, Judgment Under Pressure, and Dependability). Factor2 measures physical skills and job enthusiasm (Observational Skills, Willingness to Confront Problems, Physical Ability, Desire for Self-Improvement, Dependability, and Appearance). Factor3 measures cognitive skills (Communication Skills, Problem Solving, Learning Ability, and Judgment Under Pressure).

However, using 0.5 for determining a salient variable-factor relationship does not take sampling variability into account. If the underlying assumptions for the maximum likelihood estimation are approximately satisfied, you can output standard error estimates and the confidence intervals with METHOD=ML. You can then determine the salience of the variable-factor relationship by using the coverage displays. See the section "Confidence Intervals and the Salience of Factor Loadings" on page 2907 for more details.

Figure 44.5 displays the variance explained by each factor and the final communality estimates after the orthogonal rotation. Even though the variances explained by the rotated factors are different from that of the unrotated factor (compare with Figure 44.3), the cumulative variance explained by the common factors remains the same. Note also that the final communalities for variables, as well as the total communality, remain unchanged after rotation. Although rotating a factor solution will not increase or decrease the statistical quality of the factor model, it can simplify the interpretations of the factors and redistribute the variance explained by the factors.

Figure 44.5 Variance Explained and Final Communality Estimates after Rotation

Variance Explained by Each Factor		
Factor1	Factor2	Factor3
3.1024330	2.7684489	2.3294622

Figure 44.5 *continued*

Final Communality Estimates: Total = 8.200344						
Communication Skills	Problem Solving	Learning Ability	Judgment under Pressure	Observational Skills	Willingness to Confront Problems	Interest in People
0.64036292	0.59791844	0.61924167	0.63972863	0.69237485	0.67538695	0.79833968

Interpersonal Sensitivity	Desire for Self-Improvement	Appearance	Dependability	Physical Ability	Integrity
0.79951357	0.56879171	0.39403630	0.66056907	0.39504805	0.71903222

Syntax: FACTOR Procedure

The following statements are available in the FACTOR procedure:

PROC FACTOR < *options* > ;
  VAR *variables* ;
  PRIORS *communalities* ;
  PATHDIAGRAM < *options* > ;
  PARTIAL *variables* ;
  FREQ *variable* ;
  WEIGHT *variable* ;
  BY *variables* ;

Usually only the VAR statement is needed in addition to the PROC FACTOR statement. The descriptions of the BY, FREQ, PARTIAL, PRIORS, VAR, and WEIGHT statements follow the description of the PROC FACTOR statement in alphabetical order.

PROC FACTOR Statement

PROC FACTOR < *options* > ;

The PROC FACTOR statement invokes the FACTOR procedure. The *options* listed in Table 44.1 are available in the PROC FACTOR statement.

Table 44.1 Options Available in the PROC FACTOR Statement

Option	Description
Data Set Options	
DATA=	Specifies input SAS data set
OUT=	Specifies output SAS data set
OUTSTAT=	Specifies output data set containing statistical results
TARGET=	Specifies input data set containing the target pattern for rotation

Table 44.1 *continued*

Option	Description
Factor Extraction and Communalities	
HEYWOOD	Sets to 1 any communality greater than 1
METHOD=	Specifies the estimation method
PRIORS=	Specifies the method for computing prior communality estimates
RANDOM=	Specifies the seed for pseudo-random number generation
ULTRAHEYWOOD	Allows communalities to exceed 1
Number of Factors	
MINEIGEN=	Specifies the smallest eigenvalue for retaining a factor
NFACTORS=	Specifies the number of factors to retain or the factor extraction criterion to use
PROPORTION=	Specifies the proportion of common variance in extracted factors
Data Analysis Options	
ALPHA=	Specifies the confidence level for interval construction
COVARIANCE	Requests factoring of the covariance matrix
COVER=	Computes the confidence interval and specifies the coverage reference point
MAP	Requests a minimum average partial (MAP) correlation analysis
NOINT	Omits the intercept from computing covariances or correlations
PARALLEL	Requests a parallel analysis
SE	Requests the standard error estimates in ML estimation
VARDEF=	Specifies the divisor used in calculating covariances or correlations
WEIGHT	Factors a weighted correlation or covariance matrix
Rotation Method and Properties	
GAMMA=	Specifies the orthomax weight
HKPOWER=	Specifies the power in Harris-Kaiser rotation
NORM=	Specifies the method for row normalization in rotation
NOPROMAXNORM	Turns off row normalization in promax rotation
POWER=	Specifies the power to be used in promax rotation
PREROTATE=	Specifies the prerotation method in promax rotation
RCONVERGE=	Specifies the convergence criterion for rotation cycles
RITER=	Specifies the maximum number of cycles for rotation
ROTATE=	Specifies the rotation method
TAU=	Specifies the oblimin weight
ODS Graphics	
PLOTS=	Specifies ODS Graphics selection

Table 44.1 *continued*

Option	Description
Control Display Output	
ALL	Displays all optional output except plots
CORR	Displays the (partial) correlation matrix
EIGENVECTORS	Displays the eigenvectors of the reduced correlation matrix
FLAG=	Specifies the minimum absolute value to be flagged in the correlation and loading matrices
FUZZ=	Specifies the maximum absolute value to be displayed as missing in the correlation and loading matrices
MSA	Computes Kaiser's measure of sampling adequacy and the related partial correlations
NOPRINT	Suppresses the display of all output
NPLOTS=	Specifies the number of factors to be plotted
PLOT	Plots the rotated factor pattern
PLOTREF	Plots the reference structure
PREPLOT	Plots the factor pattern before rotation
PRINT	Displays the input factor pattern or scoring coefficients and related statistics
REORDER	Reorders the rows (variables) of various factor matrices
RESIDUALS	Displays the residual correlation matrix and the associated partial correlation matrix
ROUND	Prints correlation and loading matrices with rounded values
SCORE	Displays the factor scoring coefficients
SCREE	Displays the scree plot of the eigenvalues
SIMPLE	Displays means, standard deviations, and number of observations
Numerical Properties	
CONVERGE=	Specifies the convergence criterion
MAXITER=	Specifies the maximum number of iterations
NTHREADS=	Specifies the maximum number of simultaneous computational threads available to the procedure
SINGULAR=	Specifies the singularity criterion
Miscellaneous	
NOCORR	Excludes the correlation matrix from the OUTSTAT= data set
NOBS=	Specifies the number of observations
PARPREFIX=	Specifies the prefix for the residual variables in the output data sets
PREFIX=	Specifies the prefix for naming factors

ALL

displays all optional output except plots. When the input data set is TYPE=CORR, TYPE=UCORR, TYPE=COV, TYPE=UCOV, or TYPE=FACTOR, simple statistics, correlations, and MSA are not displayed.

ALPHA=*p*

specifies the level of confidence $1 - p$ for interval construction. By default, $p = 0.05$, corresponding to $1 - p = 95\%$ confidence intervals. If p is greater than one, it is interpreted as a percentage and divided by 100. With multiple confidence intervals to be constructed, the ALPHA= value is applied to each interval construction one at a time. This will not control the coverage probability of the intervals simultaneously. To control familywise coverage probability, you might consider supplying a nonconventional p by using methods such as Bonferroni adjustment.

CONVERGE=*p*

CONV=*p*

specifies the convergence criterion for the METHOD=PRINIT, METHOD=ULS, METHOD=ALPHA, or METHOD=ML option. Iteration stops when the maximum change in the communalities is less than the value of the CONVERGE= option. The default value is 0.001. Negative values are not allowed.

CORR

C

displays the correlation matrix or partial correlation matrix.

COVARIANCE

COV

requests factoring of the covariance matrix instead of the correlation matrix. The COV option is effective only with the METHOD=PRINCIPAL, METHOD=PRINIT, METHOD=ULS, or METHOD=IMAGE option. For other methods, PROC FACTOR produces the same results with or without the COV option.

COVER <=*p***>**

CI <=*p***>**

computes the confidence intervals and optionally specifies the value of factor loading for coverage detection. By default, $p = 0$. The specified value is represented by an asterisk (*) in the coverage display. This is useful for determining the salience of loadings. For example, if COVER=0.4, a display '0*[ ]' indicates that the entire confidence interval is above 0.4, implying strong evidence for the salience of the loading. See the section "Confidence Intervals and the Salience of Factor Loadings" on page 2907 for more details.

DATA=*SAS-data-set*

specifies the input data set, which can be an ordinary SAS data set or a specially structured SAS data set as described in the section "Input Data Set" on page 2900. If the DATA= option is omitted, the most recently created SAS data set is used.

EIGENVECTORS
EV

displays the eigenvectors of the reduced correlation matrix, of which the diagonal elements are replaced with the communality estimates. When METHOD=ML, the eigenvectors are for the weighted reduced correlation matrix. PROC FACTOR chooses the solution that makes the sum of the elements of each eigenvector nonnegative. If the sum of the elements is equal to zero, then the sign depends on how the number is rounded off.

FLAG=*p*

flags absolute values larger than *p* with an asterisk in the correlation and loading matrices. Negative values are not allowed for *p*. Values printed in the matrices are multiplied by 100 and rounded to the nearest integer (see the ROUND option). The FLAG= option has no effect when standard errors or confidence intervals are also printed.

FUZZ=*p*

prints correlations and factor loadings with absolute values less than *p* printed as missing. For partial correlations, the FUZZ= value is divided by 2. For residual correlations, the FUZZ= value is divided by 4. The exact values in any matrix can be obtained from the OUTSTAT= and ODS output data sets. Negative values are not allowed. The FUZZ= option has no effect when standard errors or confidence intervals are also printed.

GAMMA=*p*

specifies the orthomax weight used with the option ROTATE=ORTHOMAX or PREROTATE=ORTHOMAX. Alternatively, you can use ROTATE=ORTHOMAX(*p*) with *p* representing the orthomax weight. There is no restriction on valid values for the orthomax weight, although the most common values are between zero and the number of variables. The default GAMMA= value is one, resulting in the varimax rotation. See the section "Simplicity Functions for Rotations" on page 2906 for more details.

HEYWOOD
HEY

sets to 1 any communality greater than 1, allowing iterations to proceed. See the section "Heywood Cases and Other Anomalies about Communality Estimates" on page 2909 for a discussion of Heywood cases.

HKPOWER=*p*
HKP=*p*

specifies the power of the square roots of the eigenvalues to use to rescale the eigenvectors for Harris-Kaiser rotation (ROTATE=HK), assuming that the factors are extracted by the principal factor method. If the principal factor method is not used for factor extraction, the eigenvectors are replaced by the normalized columns of the unrotated factor matrix and the eigenvalues are replaced by the column normalizing constants.

Values of *p* between 0.0 and 1.0 are reasonable; a value of 1.0 is equivalent to an orthogonal rotation, with the varimax rotation as the default. You can also specify this option with an orthogonal rotation, such as ROTATE=QUARTIMAX, ROTATE=BIQUARTIMAX, ROTATE=EQUAMAX, or ROTATE=ORTHOMAX, and so on. Such a combination specifies a Harris-Kaiser case II orthoblique rotation that produces an oblique factor solution in general. By default, HKPOWER=0.0, which yields the independent cluster solution, in which each variable tends to have a large loading on only one factor.

MAP

requests a minimum average partial (MAP) correlation analysis that uses squared (Velicer 1976) and fourth-powered (Velicer, Eaton, and Fava 2000) partial correlations. Although you can use MAP analysis to suggest the number of factors, this option merely performs MAP analysis and does not affect the number of factors that are extracted. To use the MAP criterion for determining the number of factors, specify NFACTORS=MAP2 or NFACTORS=MAP4 in the PROC FACTOR statement. To obtain a graphical summary of the MAP analysis, specify PLOTS=MAP, PLOTS=MAP2, or PLOTS=MAP4. You cannot use the MAP option if METHOD=IMAGE, PATTERN, or SCORE, or if the number of observations is smaller than the number of variables.

MAXITER=*n*

specifies the maximum number of iterations for factor extraction. You can use the MAXITER= option with the PRINIT, ULS, ALPHA, or ML method. The default is 30.

METHOD=*name*

M=*name*

specifies the method for extracting factors. The default is METHOD=PRINCIPAL unless the DATA= data set is TYPE=FACTOR, in which case the default is METHOD=PATTERN. Valid values for *name* are as follows:

ALPHA | A produces alpha factor analysis.

HARRIS | H yields Harris component analysis of $\mathbf{S}^{-1}\mathbf{R}\mathbf{S}^{-1}$ (Harris 1962), a noniterative approximation to canonical component analysis.

IMAGE | I yields principal component analysis of the image covariance matrix, not the image analysis of Kaiser (1963, 1970) or Kaiser and Rice (1974). A nonsingular correlation matrix is required.

ML | M performs maximum likelihood factor analysis with an algorithm due, except for minor details, to Fuller (1987). The option METHOD=ML requires a nonsingular correlation matrix.

PATTERN reads a factor pattern from a TYPE=FACTOR, TYPE=CORR, TYPE=UCORR, TYPE=COV, or TYPE=UCOV data set. If you create a TYPE=FACTOR data set in a DATA step, only observations containing the factor pattern (_TYPE_='PATTERN') and, if the factors are correlated, the interfactor correlations (_TYPE_='FCORR') are required.

PRINCIPAL | PRIN | P yields principal component analysis if no PRIORS option or statement is used or if you specify PRIORS=ONE; if you specify a PRIORS statement or a PRIORS= value other than PRIORS=ONE, a principal factor analysis is performed.

PRINIT yields iterated principal factor analysis.

SCORE reads scoring coefficients (_TYPE_='SCORE') from a TYPE=FACTOR, TYPE=CORR, TYPE=UCORR, TYPE=COV, or TYPE=UCOV data set. The data set must also contain either a correlation or a covariance matrix. Scoring coefficients are also displayed if you specify the OUT= option.

ULS | U produces unweighted least squares factor analysis.

MINEIGEN=*p*

MIN=*p*

specifies the smallest eigenvalue for which to retain a factor. If you specify two or more of the MINEIGEN=, NFACTORS=*n*, and PROPORTION= options, the number of factors retained is the minimum number that satisfies any of the criteria. The MINEIGEN= option cannot be used with either the METHOD=PATTERN option or the METHOD=SCORE option. Negative values of *p* are not allowed. By default, MINEIGEN=0 unless you omit both the NFACTORS=*n* and the PROPORTION= options and one of the following conditions holds:

- If you specify the METHOD=ALPHA or METHOD=HARRIS option, then MINEIGEN=1.
- If you specify the METHOD=IMAGE option, then

$$\text{MINEIGEN} = \frac{\text{total image variance}}{\text{number of variables}}$$

- For any other METHOD= specification, if prior communality estimates of 1.0 are used, then

$$\text{MINEIGEN} = \frac{\text{total weighted variance}}{\text{number of variables}}$$

  When an unweighted correlation matrix is factored, this value is 1.

MSA

produces the partial correlations between each pair of variables controlling for all other variables (the negative anti-image correlations) and Kaiser's measure of sampling adequacy (Kaiser 1970; Kaiser and Rice 1974; Cerny and Kaiser 1977). This option is ignored if METHOD=IMAGE.

NFACTORS=*n* | *name*

NFACT=*n* | *name*

N=*n* | *name*

specifies either the maximum number of factors (*n*) or a *name* that represents a specific factor extraction criterion.

NFACTORS=*n* specifies the maximum number of factors to be extracted and determines the amount of memory to be allocated for factor matrices. This is the default option, with *n* equal to the number of variables. Specifying a number that is small relative to the number of variables can substantially decrease the amount of memory required to run PROC FACTOR, especially with oblique rotations. If you specify two or more of the NFACTORS=*n*, MINEIGEN=, and PROPORTION= options, the retained number of factors is the minimum number that satisfies any of the criteria. If you specify the option NFACTORS=0, eigenvalues are computed but no factors are extracted. If you specify the option NFACTORS=–1, neither eigenvalues nor factors are computed. You can use the NFACTORS=*n* option with the METHOD=PATTERN or METHOD=SCORE option to specify a smaller number of factors than are present in the data set.

Alternatively, you can specify one of the following as the *name* for the factor extraction criterion:

MAP | MAP2 requests a minimum average partial (MAP) correlation analysis (Velicer 1976) to determine the number of factors to extract. All other criteria for determining the number of factors are ignored. This criterion selects the number of factors that results in the smallest average squared partial correlations among variables. You

cannot use this criterion if METHOD=IMAGE, PATTERN, or SCORE, or if the number of observations is smaller than the number of variables.

To request a minimum average partial correlation analysis without using the MAP criterion for determining the number of extracted factors, use the MAP option in the PROC FACTOR statement.

MAP4 requests a minimum average partial (MAP) correlation analysis similar to the NFACTORS=MAP option, but using the smallest average fourth-powered partial correlations among variables (Velicer, Eaton, and Fava 2000). You cannot use this criterion if METHOD=IMAGE, PATTERN, or SCORE, or if the number of observations is smaller than the number of variables.

PARALLEL<(*suboptions*)> requests a parallel analysis (Glorfeld 1995; Horn 1965) to determine the number of factors to extract. All other criteria for determining the number of factors are ignored. To request a parallel analysis without using its criterion for determining the number of extracted factors, use the PARALLEL option in the PROC FACTOR statement. You cannot use this criterion if METHOD=IMAGE, PATTERN, or SCORE, or if the number of observations is smaller than the number of variables.

This criterion selects the number of factors that corresponds to the first n consecutive eigenvalues of the sample correlation matrix that are significantly greater than the corresponding eigenvalues of random correlation matrices. PROC FACTOR generates these random correlation matrices by simulation from a multivariate standard normal distribution. A factor is accepted if an observed eigenvalue is greater than the critical value at a specified one-sided α-level, with reference to the corresponding simulated distribution of random eigenvalues. As soon as an observed eigenvalue is less than or equal to the corresponding critical value, no more factors are counted. To obtain a graphical summary of the results of the parallel analysis, specify PLOTS=PARALLEL.

To fine-tune the parallel analysis, the NFACTORS=PARALLEL option accepts the same *suboptions* as those described for the PARALLEL option.

NOBS=*n*

specifies the number of observations. If the DATA= input data set is a raw data set, *nobs* is defined by default to be the number of observations in the raw data set. The NOBS= option overrides this default definition. If the DATA= input data set contains a covariance, correlation, or scalar product matrix, the number of observations can be specified either by using the NOBS= option in the PROC FACTOR statement or by including a _TYPE_='N' observation in the DATA= input data set.

NOCORR

prevents the correlation matrix from being transferred to the OUTSTAT= data set when you specify the METHOD=PATTERN option. The NOCORR option greatly reduces memory requirements when there are many variables but few factors. The NOCORR option is not effective if the correlation matrix is required for other requested output; for example, if the scores or the residual correlations are displayed (for example, by using the SCORE, RESIDUALS, or ALL option).

NOINT

omits the intercept from the analysis; covariances or correlations are not corrected for the mean.

NOPRINT

suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

NOPROMAXNORM

NOPMAXNORM

turns off the default row normalization of the prerotated factor pattern, which is used in computing the promax target matrix.

NORM=COV | KAISER | NONE | RAW | WEIGHT

specifies the method for normalizing the rows of the factor pattern for rotation. If you specify the option NORM=KAISER, Kaiser's normalization is used ($\sum_j p_{ij}^2 = 1$). If you specify the option NORM=WEIGHT, the rows are weighted by the Cureton-Mulaik technique (Cureton and Mulaik 1975). If you specify the option NORM=COV, the rows of the pattern matrix are rescaled to represent covariances instead of correlations. If you specify the option NORM=NONE or NORM=RAW, normalization is not performed. The default is NORM=KAISER.

NPLOTS=*n*

NPLOT=*n*

specifies the number of factors to be plotted. The default is to plot all factors. The smallest allowable value is 2. If you specify the option NPLOTS=*n*, all pairs of the first *n* factors are plotted, producing a total of $n(n-1)/2$ plots.

NTHREADS=*n*

THREADS=*n*

specifies the maximum number of simultaneous computational threads available to the procedure. By default, the procedure uses the values of the THREADS and CPUCOUNT system variables to determine the number of computational threads. To explicitly request this default behavior, specify $n < 1$. To disable multithreading within PROC FACTOR, specify NTHREADS=1.

Multithreading is available only for parallel analysis and the generalized Crawford-Ferguson family of factor rotations. For parallel analysis, multithreading is enabled by default. For generalized Crawford-Ferguson rotations, multithreading is enabled by default only if the number of factors is 20 or greater. You can enable multithreading for rotation (for any number of factors) by specifying $n > 1$.

OUT=*SAS-data-set*

creates a data set containing all the data from the DATA= data set plus variables called Factor1, Factor2, and so on, containing estimated factor scores. The DATA= data set must contain multivariate data, not correlations or covariances. You must also specify the NFACTORS=*n* option to determine the number of factor score variables. If you specify partial variables in the PARTIAL statement, the OUT= data set will also contain the residual variables that are used for factor analysis. The output data set is described in detail in the section "Output Data Sets" on page 2902. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTSTAT=*SAS-data-set*

specifies an output data set containing most of the results of the analysis. The output data set is described in detail in the section "Output Data Sets" on page 2902. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PARALLEL < (*suboptions*) >

requests a parallel analysis as described by Glorfeld (1995) and Horn (1965). Although you can use parallel analysis to suggest the number of factors, this option merely performs the parallel analysis and does not affect the number of factors that are extracted. To use the parallel analysis criterion for determining the number of factors, specify NFACTORS=PARALLEL in the PROC FACTOR statement. To obtain a graphical summary of the results of the parallel analysis, specify PLOTS=PARALLEL. You cannot use the PARALLEL option if METHOD=IMAGE, PATTERN, or SCORE, or if the number of observations is smaller than the number of variables.

To conduct a parallel analysis, PROC FACTOR compares the eigenvalues of the sample correlation matrix to the corresponding eigenvalues of random correlation matrices. These random correlation matrices are simulated from a multivariate standard normal distribution. A factor is accepted if an observed eigenvalue is greater than the critical value at a specified one-sided α-level, with reference to the corresponding simulated distribution of random eigenvalues. As soon as an observed eigenvalue is less than or equal to the corresponding critical value, no more factors are counted.

To fine-tune the parallel analysis, you can specify any of the following *suboptions*:

ALPHA=*p* specifies the one-sided α-level for computing the critical value at the upper-end of the simulated distribution of eigenvalues, where *p* must be between 0 and 1. By default, ALPHA=0.05.

NSIMS=*n* specifies the number of simulations to use to construct an empirical distribution of eigenvalues for the parallel analysis, where *n* must be greater than 200. By default, NSIMS=1000.

SEED=*n* specifies a positive integer as a starting value for the pseudorandom number generator that is used to simulate correlation matrices for the parallel analysis, where *n* must be an integer ≥ 1. If you do not specify the SEED= suboption, the time of day is used to initialize the pseudorandom number sequence.

PARPREFIX=*name*

specifies the prefix for the residual variables in the OUT= and the OUTSTAT= data sets when partial variables are specified in the PARTIAL statement.

PLOT

plots the factor pattern after rotation. This option produces printer plots. High-quality ODS graphical plots for factor patterns can be requested with the PLOTS=LOADINGS or PLOTS=INITLOADINGS option.

PLOTREF
plots the reference structure instead of the default factor pattern after oblique rotation.

PLOTS < (*global-plot-options*) > = *plot-request* < (*options*) >

PLOTS < (*global-plot-options*) > = (*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >)
specifies one or more ODS graphical plots in PROC FACTOR. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. Here are some examples:

```
plots=all
plots(flip)=loadings
plots=(loadings(flip) scree(unpack))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc factor plots=all;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

For an example containing graphical displays of factor analysis results, see Example 44.2.

The following table shows the available *plot-requests* and their available *options*:

Plot-Request	**Plot Description**	**Suboptions**
ALL	All available plots	All
INITLOADINGS	Unrotated factor loadings	CIRCLE=, FLIP, NPLOTS=, PLOTREF, and VECTOR
LOADINGS	Rotated factor loadings	CIRCLE=, FLIP, NPLOTS=, PLOTREF, and VECTOR
MAP	MAP2 and MAP4	
MAP2	Average squared partial correlations	
MAP4	Average fourth-powered partial correlations	
NONE	No ODS graphical plots	
PARALLEL	Parallel analysis	
PATHDIAGRAM	Path diagram	
PRELOADINGS	Prerotated factor loadings	CIRCLE=, FLIP, NPLOTS=, PLOTREF, and VECTOR
SCREE	Scree and variance explained	UNPACK

The following are the available *global-plot-options* or *options* for plots:

CIRCLE < = *numbers* > | **CIRCLES** < = *numbers* > draws circular reference lines in scatter plots or vector plots of factor loadings. You can specify the locations of the circular reference lines in the *numbers* list. Each number indicates the proportion or percentage of area of the unit circle that is enclosed by the specified circle. Each of the *numbers* must lie between 0 and 100, inclusively. When a number is between 0

and 1 (inclusively), it is interpreted as a proportion; otherwise, it is interpreted as a percentage. The maximum number of circles is 5.

The CIRCLE option applies to the scatter or vector plots requested by the INITLOADINGS, LOADINGS, and PRELOADINGS options. By default, a unit-circle, which represents 100% of the total area, is drawn for the vector plots. However, no circle will be drawn for scatter plots unless the CIRCLE option is specified. Two special cases for this option are: (1) With no *numbers* following the CIRCLE option, a 100% circle will be drawn. (2) With CIRCLE=0, no circle will be drawn. This special case is primarily used to turn off the default unit-circle in vector plots.

FLIP switches the X and Y axes. It applies to the INITLOADINGS, LOADINGS, and PRELOADINGS *plot-requests*.

NPLOTS=*n* | **NPLOT**=*n* specifies the number of factors n ($n \geq 2$) to be plotted. It applies to the INITLOADINGS, LOADINGS, and PRELOADINGS *plot-requests*. Since this option can also be specified in the PROC FACTOR statement, the final value of n is determined by the following steps. The NPLOTS= value of the PROC FACTOR is read first. If the NPLOTS= option is specified as a *global-plot-option*, the value of n will be updated. Then, if the NPLOTS= option is again specified in an individual *plot-request*, the value will be updated again for that individual *plot-request*. For example, in the following statement, four factors are extracted with the N=4 option:

```
proc factor n=4 nplots=3 plots(nplots=4)=
                 (loadings preloadings(nplots=2));
```

Initially, plots of the first three factors are specified with the NPLOTS=3 option. When you are producing ODS graphical plots, the *global-plot-option* NPLOTS=4 is used. As a result, the LOADINGS *plot-request* will produce plots for all pairs of the first 4 factors. However, because the NPLOTS=2 is specified locally for the PRELOADINGS *plot-request*, it will produce a prerotated factor loading plot for the first two factors only.

The default NPLOTS= value is 5 or the total number of factors (m), whichever is smaller. If you specify an NPLOTS= value that is greater than m, NPLOTS=m will be used.

PATHDIAGRAM creates a path diagram for the last factor model. The last factor model refers to the initial factor solution if you do not specify any rotations. It refers to the rotated factor solution if you use the ROTATE= option. The path diagram shows the links between factors and variables, the factor correlations, and the error variances in the model.

The path diagram does not display all non-zero directed links between factors and variables. It displays only those directed links that have factor loading estimates at 0.3 or bigger in magnitude. PROC FACTOR uses this 0.3-criterion by default. You can set your own criterion by using the FUZZ= option in the PROC FACTOR statement.

If you use both the METHOD=ML and SE options in the PROC FACTOR statement, the statistical significance of the factor loading estimate is also required for displaying the corresponding directed link between a variable and a factor. The

default level of significance is 0.05. You can set your own level of significance by using the ALPHA= option in the PROC FACTOR statement.

Alternatively, you can produce path diagrams by using the PATHDIAGRAM Statement, which provides many options that enables you to create highly customized path diagrams.

PLOTREF plots the reference structures rather than the factor pattern loadings. It applies to the INITLOADINGS, LOADINGS, and PRELOADINGS *plot-requests* when the factor solution is oblique. This option can also be set globally as an option in the PROC FACTOR statement.

UNPACK plots component graphs separately. It applies to the SCREE *plot-request* only.

VECTOR plots loadings in vector form. It applies to the INITLOADINGS, LOADINGS, and PRELOADINGS *plot-requests* when the factor solution is orthogonal. For oblique solutions, the VECTOR option is ignored and the default scatter plots for factor loadings or reference structures are displayed.

Be aware that the PLOT option in the PROC FACTOR statement requests only the printer plots of factor loadings. The current option PLOTS= or PLOT=, however, is for ODS graphical plots.

You can specify options for the requested ODS graphical plots as *global-plot-options* or as local *options*. *Global-plot-options* apply to all appropriate individual *plot-requests* specified. For example, because the SCREE plot is not subject to axes flipping, the following two specifications are equivalent:

```
plots(flip)=(loadings preloadings scree)
plots=(loadings(flip) preloadings(flip) scree)
```

Options specified locally after each *plot-request* apply to that plot-request only. For example, consider the following specification:

```
plots=(scree(unpack) loadings(plotref) preloadings(flip))
```

The FLIP option applies to the PRELOADINGS *plot-request* but not the LOADINGS *plot-request*; the PLOTREF option applies to the LOADINGS *plot-request* but not the PRELOADINGS *plot-request*; and the UNPACK option applies to the SCREE *plot-request* only.

POWER=*n*

specifies the power to be used in computing the target pattern for the option ROTATE=PROMAX. Valid values must be integers ≥ 1. The default value is 3. You can also specify the POWER= value in the ROTATE= option—for example, ROTATE=PROMAX(4).

PREFIX=*name*

specifies a prefix for naming the factors. By default, the names are Factor1, Factor2, . . . , Factor*n*. If you specify PREFIX=ABC, the factors are named ABC1, ABC2, ABC3, and so on. The number of characters in the prefix plus the number of digits required to designate the variables should not exceed the current name length defined by the VALIDVARNAME= system option.

PREPLOT

plots the factor pattern before rotation. This option produces printer plots. High-quality ODS graphical plots for factor patterns can be requested with the PLOTS=PRELOADINGS option.

PREROTATE=*name*

PRE=*name*

specifies the prerotation method for the option ROTATE=PROMAX. Any rotation method other than PROMAX or PROCRUSTES can be used. The default is PREROTATE=VARIMAX. If a previously rotated pattern is read using the option METHOD=PATTERN, you should specify the PREROTATE=NONE option.

PRINT

displays the input factor pattern or scoring coefficients and related statistics. In oblique cases, the reference and factor structures are computed and displayed. The PRINT option is effective only with the option METHOD=PATTERN or METHOD=SCORE.

PRIORS=*name*

specifies a method for computing prior communality estimates. You can specify numeric values for the prior communality estimates by using the PRIORS statement. Valid values for *name* are as follows:

ASMC | A sets the prior communality estimates proportional to the squared multiple correlations but adjusted so that their sum is equal to that of the maximum absolute correlations (Cureton 1968).

INPUT | I reads the prior communality estimates from the first observation with either _TYPE_='PRIORS' or _TYPE_='COMMUNAL' in the DATA= data set (which cannot be TYPE=DATA).

MAX | M sets the prior communality estimate for each variable to its maximum absolute correlation with any other variable.

ONE | O sets all prior communalities to 1.0.

RANDOM | R sets the prior communality estimates to pseudo-random numbers uniformly distributed between 0 and 1.

SMC | S sets the prior communality estimate for each variable to its squared multiple correlation with all other variables.

The default prior communality estimates are as follows:

METHOD=	PRIORS=
PRINCIPAL	ONE
PRINIT	ONE
ALPHA	SMC
ULS	SMC
ML	SMC
HARRIS	SMC
IMAGE	SMC
PATTERN	(not applicable)
SCORE	(not applicable)

Because the use of SMC as prior communalities is a defining feature of the HARRIS and IMAGE methods, you cannot set any prior communalities other than SMC for these two methods. The PRIORS= option is not applicable to the PATTERN and SCORE methods because these methods do not use any prior communalities.

By default, the options METHOD=PRINIT, METHOD=ULS, METHOD=ALPHA, and METHOD=ML stop iterating and set the number of factors to 0 if an estimated communality exceeds 1. The options HEYWOOD and ULTRAHEYWOOD allow processing to continue.

PROPORTION=*p*

PERCENT=*p*

P=*p*

specifies the proportion of common variance to be accounted for by the retained factors. The proportion of common variance is computed using the total prior communality estimates as the basis. If the value is greater than one, it is interpreted as a percentage and divided by 100. The options PROPORTION=0.75 and PERCENT=75 are equivalent. The default value is 1.0 or 100%. You cannot specify the PROPORTION= option with the METHOD=PATTERN or METHOD=SCORE option. If you specify two or more of the PROPORTION=, NFACTORS=*n* , and MINEIGEN= options, the number of factors retained is the minimum number satisfying any of the criteria.

RANDOM=*n*

specifies a positive integer as a starting value for the pseudo-random number generator for use with the option PRIORS=RANDOM. If you do not specify the RANDOM= option, the time of day is used to initialize the pseudo-random number sequence. Valid values must be integers ≥ 1.

RCONVERGE=*p*

RCONV=*p*

specifies the convergence criterion value *p* ($p > 0$) for rotation cycles. Rotation stops when the scaled change of the simplicity function value is less than *p*. Mathematically, the convergence criterion is

$$|f_{new} - f_{old}|/K < p$$

where f_{new} and f_{old} are simplicity function values of the current cycle and the previous cycle, respectively, $K = \max(1, |f_{old}|)$ is a scaling factor, and *p* is 1E–9 by default.

REORDER

RE

causes the rows (variables) of various factor matrices to be reordered on the output. Variables with their highest absolute loading (reference structure loading for oblique rotations) on the first factor are displayed first, from largest to smallest loading, followed by variables with their highest absolute loading on the second factor, and so on. The order of the variables in the output data set is not affected. The factors are not reordered.

RESIDUALS

RES

displays the residual correlation matrix and the associated partial correlation matrix. The diagonal elements of the residual correlation matrix are the unique variances.

RITER=*n*

specifies the maximum number of cycles *n* for factor rotation. Except for promax and Procrustes, you can use the RITER= option with all rotation methods. The default *n* is the maximum between 10 times the number of variables and 100.

ROTATE=*name*

R=*name*

specifies the rotation method. The default is ROTATE=NONE.

Valid *names* for orthogonal rotations are as follows:

BIQUARTIMAX | BIQMAX specifies orthogonal biquartimax rotation. This corresponds to the specification ROTATE=ORTHOMAX(.5).

EQUAMAX | E specifies orthogonal equamax rotation. This corresponds to the specification ROTATE=ORTHOMAX with GAMMA=*number of factors*/2.

FACTORPARSIMAX | FPA specifies orthogonal factor parsimax rotation. This corresponds to the specification ROTATE=ORTHOMAX with GAMMA=*number of variables*.

NONE | N specifies that no rotation be performed, leaving the original orthogonal solution.

ORTHCF(*p1*,*p2*) | **ORCF**(*p1*,*p2*) specifies the orthogonal Crawford-Ferguson rotation with the weights *p1* and *p2* for variable parsimony and factor parsimony, respectively. See the definitions of weights in the section "Simplicity Functions for Rotations" on page 2906.

ORTHGENCF(*p1*,*p2*,*p3*,*p4*) | **ORGENCF**(*p1*,*p2*,*p3*,*p4*) specifies the orthogonal generalized Crawford-Ferguson rotation with the four weights *p1*, *p2*, *p3*, and *p4*. See the definitions of weights in the section "Simplicity Functions for Rotations" on page 2906.

ORTHOMAX<(*p*)> | **ORMAX**<(*p*)> specifies the orthomax rotation with orthomax weight *p*. If ROTATE=ORTHOMAX is used, the default *p* value is 1 unless specified otherwise in the GAMMA= option. Alternatively, ROTATE=ORTHOMAX(*p*) specifies *p* as the orthomax weight or the GAMMA= value. See the definition of the orthomax weight in the section "Simplicity Functions for Rotations" on page 2906.

PARSIMAX | PA specifies orthogonal parsimax rotation. This corresponds to the specification ROTATE=ORTHOMAX with

$$\text{GAMMA} = \frac{nvar \times (nfact - 1)}{nvar + nfact - 2}$$

where *nvar* is the number of variables and *nfact* is the number of factors.

QUARTIMAX | QMAX | Q specifies orthogonal quartimax rotation. This corresponds to the specification ROTATE=ORTHOMAX(0).

VARIMAX | V specifies orthogonal varimax rotation. This corresponds to the specification ROTATE=ORTHOMAX with GAMMA=1.

Valid *names* for oblique rotations are as follows:

BIQUARTIMIN | BIQMIN specifies biquartimin rotation. It corresponds to the specification ROTATE=OBLIMIN(.5) or ROTATE=OBLIMIN with TAU=0.5.

COVARIMIN | CVMIN specifies covarimin rotation. It corresponds to the specification ROTATE=OBLIMIN(1) or ROTATE=OBLIMIN with TAU=1.

HK<(*p*)> | H<(*p*)> specifies Harris-Kaiser case II orthoblique rotation. The Harris-Kaiser rotation makes use of the orthogonal rotation algorithm to produce factor solutions that are, in general, oblique. When specifying this option, you can use the HKPOWER= option to specify the power of the square roots of the eigenvalues by which the eigenvectors are scaled, assuming that the factors are extracted by the principal factor method. For other extraction methods, the unrotated factor pattern is column normalized. The power is then applied to the column normalizing constants, instead of the eigenvalues. You can also use ROTATE=HK(*p*), with *p* representing the HKPOWER= value. The default associated orthogonal rotation with ROTATE=HK is the varimax rotation without Kaiser normalization. You might associate the Harris-Kaiser with other orthogonal rotations by using the ROTATE= option together with the HKPOWER= option.

OBBIQUARTIMAX | OBIQMAX specifies oblique biquartimax rotation.

OBEQUAMAX | OE specifies oblique equamax rotation.

OBFACTORPARSIMAX | OFPA specifies oblique factor parsimax rotation.

OBLICF(*p1*,*p2*) | **OBCF**(*p1*,*p2*) specifies the oblique Crawford-Ferguson rotation with the weights *p1* and *p2* for variable parsimony and factor parsimony, respectively. See the definitions of weights in the section "Simplicity Functions for Rotations" on page 2906.

OBLIGENCF(*p1*,*p2*,*p3*,*p4*) | **OBGENCF**(*p1*,*p2*,*p3*,*p4*) specifies the oblique generalized Crawford-Ferguson rotation with the four weights *p1*, *p2*, *p3*, and *p4*. See the definitions of weights in the section "Simplicity Functions for Rotations" on page 2906.

OBLIMIN<(*p*)> | OBMIN<(*p*)> specifies the oblimin rotation with oblimin weight *p*. If ROTATE=OBLIMIN is used, the default *p* value is zero unless specified otherwise in the TAU= option. Alternatively, ROTATE=OBLIMIN(*p*) specifies *p* as the oblimin weight or the TAU= value. See the definition of the oblimin weight in the section "Simplicity Functions for Rotations" on page 2906.

OBPARSIMAX | OPA specifies oblique parsimax rotation.

OBQUARTIMAX | OQMAX specifies oblique quartimax rotation. This is the same as the QUARTIMIN method.

OBVARIMAX | OV specifies oblique varimax rotation.

PROCRUSTES specifies oblique Procrustes rotation with the target pattern provided by the TARGET= data set. The unrestricted least squares method is used with factors scaled to unit variance after rotation.

PROMAX<(*p*)> **| P**<(*p*)> specifies oblique promax rotation. You can use the PREROTATE= option to set the desirable prerotation method, orthogonal or oblique. When used with ROTATE=PROMAX, the POWER= option lets you specify the power for forming the target. You can also use ROTATE=PROMAX(*p*), where *p* represents the POWER= value.

QUARTIMIN | QMIN specifies quartimin rotation. It is the same as the oblique quartimax method. It also corresponds to the specification ROTATE=OBLIMIN(0) or ROTATE=OBLIMIN with TAU=0.

ROUND

prints correlation and loading matrices with entries multiplied by 100 and rounded to the nearest integer. The exact values can be obtained from the OUTSTAT= and ODS output data sets. The ROUND option also flags absolute values larger than the FLAG= value with an asterisk in correlation and loading matrices (see the FLAG= option). If the FLAG= option is not specified, the root mean square of all the values in the matrix printed is used as the default FLAG= value. The ROUND option has no effect when standard errors or confidence intervals are also printed.

SCORE

displays the factor scoring coefficients. The squared multiple correlation of each factor with the variables is also displayed except in the case of unrotated principal components. The SCORE option also outputs the factor scoring coefficients in the _TYPE_=SCORE or _TYPE_=USCORE observations in the OUTSTAT= data set. Unless you specify the NOINT option in PROC FACTOR, the scoring coefficients should be applied to standardized variables—variables that are centered by subtracting the original variable means and then divided by the original variable standard deviations. With the NOINT option, the scoring coefficients should be applied to data without centering.

SCREE

displays a scree plot of the eigenvalues (Cattell 1966, 1978; Cattell and Vogelman 1977; Horn and Engstrom 1979). This option produces printer plots. High-quality scree plots can be requested with the PLOTS=SCREE option.

SE

STDERR

computes standard errors for various classes of unrotated and rotated solutions under the maximum likelihood estimation.

SIMPLE

S

displays means, standard deviations, and the number of observations.

SINGULAR=*p*

SING=*p*

specifies the singularity criterion, where $0 < p < 1$. The default value is 1E–8.

TARGET=*SAS-data-set*

specifies an input data set containing the target pattern for Procrustes rotation (see the description of the ROTATE= option). The TARGET= data set must contain variables with the same names as those being factored. Each observation in the TARGET= data set becomes one column of the target factor pattern. Missing values are treated as zeros. The _NAME_ and _TYPE_ variables are not required and are ignored if present.

TAU=*p*

specifies the oblimin weight used with the option ROTATE=OBLIMIN or PREROTATE=OBLIMIN. Alternatively, you can use ROTATE=OBLIMIN(*p*) with *p* representing the oblimin weight. There is no restriction on valid values for the oblimin weight, although for practical purposes a negative or zero value is recommended. The default TAU= value is 0, resulting in the quartimin rotation. See the section "Simplicity Functions for Rotations" on page 2906 for more details.

ULTRAHEYWOOD

ULTRA

allows communalities to exceed 1. The ULTRAHEYWOOD option can cause convergence problems because communalities can become extremely large, and ill-conditioned Hessians might occur. See the section "Heywood Cases and Other Anomalies about Communality Estimates" on page 2909 for a discussion of Heywood cases.

VARDEF=DF | N | WDF | WEIGHT | WGT

specifies the divisor used in the calculation of variances and covariances. The default value is VARDEF=DF. The values and associated divisors are displayed in the following table where *i*=0 if the NOINT option is used and *i*=1 otherwise, and where *k* is the number of partial variables specified in the PARTIAL statement.

Value	Description	Divisor
DF	Degrees of freedom	$n - k - i$
N	Number of observations	$n - k$
WDF	Sum of weights DF	$\sum_i w_i - k - i$
WEIGHT | WGT	Sum of weights	$\sum_i w_i - k$

WEIGHT

factors a weighted correlation or covariance matrix. The WEIGHT option can be used only with the METHOD=PRINCIPAL, METHOD=PRINIT, METHOD=ULS, or METHOD=IMAGE option. The input data set must be of type CORR, UCORR, COV, UCOV, or FACTOR, and the variable weights are obtained from an observation with _TYPE_='WEIGHT'.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC FACTOR to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the FACTOR procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

If you specify the TARGET= option and the TARGET= data set does not contain any of the BY variables, then the entire TARGET= data set is used as a Procrustean target for each BY group in the DATA= data set.

If the TARGET= data set contains some but not all of the BY variables, or if some BY variables do not have the same type or length in the TARGET= data set as in the DATA= data set, then PROC FACTOR displays an error message and stops.

If all the BY variables appear in the TARGET= data set with the same type and length as in the DATA= data set, then each BY group in the TARGET= data set is used as a Procrustean target for the corresponding BY group in the DATA= data set. The BY groups in the TARGET= data set must be in the same order as in the DATA= data set. If you specify the NOTSORTED option in the BY statement, there must be identical BY groups in the same order in both data sets. If you do not specify the NOTSORTED option, some BY groups can appear in one data set but not in the other.

For more information about BY-group processing, see the “Grouping Data” section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If a variable in the data set represents the frequency of occurrence for the other values in the observation, include the variable's name in a FREQ statement. The procedure then treats the data set as if each observation appears n times, where n is the value of the FREQ variable for the observation. The total number of observations is considered to be equal to the sum of the FREQ variable when the procedure determines degrees of freedom for significance probabilities.

If the value of the FREQ variable is missing or is less than one, the observation is not used in the analysis. If the value is not an integer, the value is truncated to an integer.

The WEIGHT and FREQ statements have a similar effect, except in determining the number of observations for significance tests.

PARTIAL Statement

PARTIAL *variables* **;**

If you want the analysis to be based on a partial correlation or covariance matrix, use the PARTIAL statement to list the variables that are used to partial out the variables in the analysis.

PATHDIAGRAM Statement

PATHDIAGRAM < *options* > **;**

You can use the PATHDIAGRAM statement to specify and modify the layout algorithm, to control the formatting of estimates, and to fine-tune many graphical and nongraphical features of path diagrams. You can use multiple PATHDIAGRAM statements to produce path diagrams that have different styles and graphical features.

Specifying a PATHDIAGRAM statement without any *options* has the same effect as specifying the PLOTS=PATHDIAGRAM option in the PROC FACTOR statement. Both produce a default path diagram for the last factor solution in a PROC FACTOR run. The default path diagrams show the links between factors and variables, the factor correlations, and the error variances in the model. For more information about the default path diagram, see the PLOTS=PATHDIAGRAM option.

The *options* in the PATHDIAGRAM statement can be classified into three categories, which are summarized in Table 44.2 through Table 44.4. The following three tables summarize these *options*. An alphabetical listing of these *options* that includes more details follows the tables.

Table 44.2 shows the *options* that you can use to specify the path diagram layout algorithm, to set the criteria for displaying the directed paths between variables and factors, and to specify the size of factors relative to the observed variables.

Table 44.2 Options for Controlling the Layout

Option	Description
ALPHA=	Specifies the significance level of the loading estimate that is required in order to display the corresponding link between a variable and a factor
ARRANGE=	Specifies the algorithm for laying out variables in the path diagram
COVER=	Specifies the salience criterion on the loading estimate
FACTORSIZE=	Specifies the size of latent factors relative to observed variables
FUZZ=	Specifies the magnitude of the loading estimate that is required in order to display the corresponding link between a variable and a factor
SCALE=	Specifies the scale factor for the node size

Table 44.3 shows the *options* that you can use to control the display of parameter estimates in output path diagrams.

Table 44.3 Options for Displaying Parameter Estimates

Option	Description
DECP=	Specifies the number of decimal places in the estimates
NOESTIM	Disables the display of all numerical estimates
NOERRVAR	Disables the display of error variances
NOFACTORVAR	Disables the display of factor variances
NOVARIANCE	Disables the display of all variances

Table 44.4 shows the *options* that you can use to specify the title, the path diagram label, and the variable labels.

Table 44.4 Options for Specifying Titles and Labels

Option	Description
DESIGNHEIGHT=	Sets the height, in pixels, of the output path diagram
DESIGNWIDTH=	Sets the width, in pixels, of the output path diagram
DIAGRAMLABEL=	Specifies the label of the diagram in the ODS
LABEL=	Specifies labels of the nodes that are shown in the path diagram
NODELABEL=	Specifies whether the variable names or labels are used to label nodes
NOTITLE	Suppresses the display of the title
TITLE=	Specifies the title to display in the output path diagram

ALPHA=α

ALPHALOAD=α

specifies the significance level (α-level) of the loading estimate that is required in order to display the corresponding directed link (path) between a variable and a factor. If α is greater than 1, it is interpreted as a percentage and divided by 100. If the p-value of a loading estimate is greater than α,

the loading estimate is insignificant and PROC FACTOR does not display the corresponding link in the path diagram. By default, $\alpha = 0.05$.

The ALPHA= option applies only when you specify the METHOD=ML option in the PROC FACTOR statement and when standard errors are computed in the analysis (for example, by specifying the SE option in the PROC FACTOR statement).

If you specify the ALPHA= option in the PROC FACTOR statement, all PATHDIAGRAM statements use the same α value that is specified in the ALPHA= option in the PROC FACTOR statement unless you respecify the ALPHA= option in individual PATHDIAGRAM statements.

NOTE: The p-value of a loading estimate is computed by using a reference sampling distribution that has a specific mean value. This mean value reflects the criterion for determining the salience of loading estimates. You can use the COVER= option or the SALIENCE= option to specify the salience criterion. By default, COVER=0; so PROC FACTOR displays all directed links between variables and factors that are significantly greater than 0.

ARRANGE=*name*

ARRANGEMENT=*name*

METHOD=*name*

specifies the algorithm for laying out the variables in the path diagram. You can specify the following *names*:

FLOW specifies the process-flow algorithm.

GRIP specifies the GRIP (graph drawing with intelligent placement) algorithm.

GROUPEDFLOW specifies the grouped-flow algorithm.

By default, ARRANGE=FLOW if the number of observed variables is less than 15; otherwise, the default is ARRANGE=GRIP. The reason for switching the default layout algorithm is that when the number of observed variables becomes large, the process-flow algorithm might run out of vertical space for aligning all observed variables in a vertical line. In that case, the GRIP algorithm might be a better choice because the observed variables tend to scatter around the space rather than being aligned vertically. See Example 44.6 for the use of the ARRANGE= option. For more information and for general uses of these layout algorithms, see the section "The Process-Flow, Grouped-Flow, and GRIP Layout Algorithms" on page 1784 in Chapter 33, "The CALIS Procedure."

COVER=*p*

SALIENCE=*p*

specifies the salience criterion, p, for the loading estimate. In order to display a loading estimate and its corresponding link between a variable and a factor in the path diagram, the magnitude of the loading estimate must be significantly greater than p. By default, $p = 0$ and the significance level (α-level) is 0.05. You can specify the significance level in the ALPHA= option.

The COVER= option applies only when you specify METHOD=ML in the PROC FACTOR statement and when standard errors are computed in the analysis (for example, by specifying the SE option in the PROC FACTOR statement).

If you specify the COVER= option in the PROC FACTOR statement, any PATHDIAGRAM statement that does not include a COVER= option uses the value that is specified in the PROC FACTOR statement.

DECP=*i*

sets the number of decimal places in the estimates that are displayed in the path diagram, where *i* is between 0 and 4. The displayed estimates are at most seven digits long, including the decimal point for the nonzero value of *i*. By default, DECP=2.

DESIGNHEIGHT=*i*

DH=*i*

sets the height of the path diagram, in number of pixels, where *i* is between 100 and 10,922. The default heights are 800, 500, and 600 for ARRANGE=FLOW, GROUPEDFLOW, and GRIP, respectively. Typically, you might want to set a larger design height and width when your path diagram contains more nodes or variables.

DESIGNWIDTH=*i*

DW=*i*

sets the width of the path diagram, in number of pixels, where *i* is between 100 and 10,922. The default widths are 450, 720, and 600 for ARRANGE=FLOW, GROUPEDFLOW, and GRIP, respectively. Typically, you might want to set a larger design width and height when your path diagram contains more nodes or variables.

DIAGRAMLABEL=*name*

DLABEL=*name*

specifies the label of the path diagram. You can use any valid SAS name or quoted string of up to 256 characters for *name*. However, only up to 40 characters of the label are used by ODS. The following statements show two example label specifications:

```
pathdiagram diagramlabel=MyFactorModel;
pathdiagram diagramlabel="Varimax-Rotated Factor Solution";
```

If you do not specify this option, PROC FACTOR uses the *name* that is provided in the TITLE= option. If you specify neither the DIAGRAMLABEL= option nor the TITLE= option, PROC FACTOR uses "Path Diagram" for the label when there is only one path diagram. When there is more than one path diagram, a unique number is appended to the label of each path diagram. For example, "Path Diagram 3" is the third path diagram in the output.

FACTORSIZE=*size*

FACTSIZE=*size*

specifies the size of latent factors relative to the size of observed variables, where *size* is between 0.2 and 5. By default, FACTSIZE=1.5, which means that the size ratio of factors to observed variables is about 3 to 2.

FUZZ=*p*

specifies the magnitude, $p > 0$, of the factor loading estimate that is required in order to display the corresponding directed link between a variable and a factor. If the magnitude of a loading estimate is less than *p*, then PROC FACTOR does not display the corresponding directed link in the path diagram. By default, FUZZ=0.3.

If you specify the FUZZ= option in the PROC FACTOR statement, any PATHDIAGRAM statement that does not include a FUZZ= option uses the value that is specified in the PROC FACTOR statement.

If you specify METHOD=ML and standard errors are computed, PROC FACTOR displays only those directed links (paths) between variables and factors that are statistically significant in the path diagram. In this situation, only the criteria that are specified by the ALPHA= and COVER= options are used and the FUZZ= option is irrelevant. When METHOD=ML is not specified or standard errors are not computed, PROC FACTOR uses the criterion that is specified by the FUZZ= option.

LABEL= [*varlabel* < **,** *varlabel* . . . >**]** | **{***varlabel* < **,** *varlabel* . . . >**}**

specifies the labels of variables to be displayed in path diagrams, where each *varlabel* has the following form:

*variable***=***label*

You can use any valid SAS names or quoted strings of up to 256 characters for *labels*. The *labels* identify the corresponding variables or factors in output path diagrams. For example, instead of using original variable names such as x1 and Factor1 in the path diagram, the following statement specifies the use of more meaningful labels:

```
pathdiagram label=[x1="Simple Math" Factor1="Math Ability"];
```

This option is not the only way that you can provide labels for variables. For example, you can also use the LABEL statement to specify labels for observed variables. PROC FACTOR uses the following rules to determine the label for a node (variable) in the path diagram:

1. If you specify the label for a variable or a factor by using the LABEL= option in the PATHDIAGRAM statement, the associated node (variable) uses this label in the output path diagram. Proceed to the next rule if the label of a node is not resolved.
2. If the NODELABEL=VARNAME option is specified, the associated node uses the original variable name as its label in the output path diagram. Otherwise, proceed to the next rule.
3. If the label of a variable is specified in a LABEL statement, the associated node uses this label in the output path diagram. Otherwise, proceed to the next rule.
4. The associated node uses the original variable name as its label in the output path diagram.

NODELABEL=VARNAME | VARLABEL

specifies whether the variables (nodes) in path diagrams are labeled by the original variable names (VARNAME) or their variable labels (VARLABEL), which are provided by specifying the LABEL statement. If you provide variable labels (applicable only to observed variables) in the LABEL statement, PROC FACTOR uses those provided labels unless you specify this option.

This option is not the only determinant of the final labels of nodes in the path diagram. The specifications in the LABEL= option of the PATHDIAGRAM statement are also considered. For the rules that PROC FACTOR uses to determine the node labels, see the LABEL= option.

NOERRVAR

NOERRORVARIANCE

suppresses the default display of error variances, which are represented as double-headed paths that are attached to observed variables.

NOESTIM
NOEST

suppresses the default display of all numerical estimates in path diagrams.

NOFACTORVAR
NOFACTORVARIANCE

suppresses the default display of factor variances, which are represented as double-headed paths that are attached to factors.

NOTITLE

suppresses the display of the default title. You can use the TITLE= option to provide your own title.

NOVARIANCE

suppresses the default display of all variances. This option has the same effect as specifying both the NOFACTORVAR and NOERRVAR options.

SCALE=*n*
DIAGRAMSCALE=*n*

specifies the scaling factor, *n*, for the node size relative to the dimensions of the path diagram. Valid values of *n* are between 0 and 6. This option applies to the ARRANGE=GRIP layout only.

PROC FACTOR uses certain default pixel dimensions for the nodes in path diagrams that have default design dimensions (see the DESIGNHEIGHT= and DESIGNWIDTH= options for the default design dimensions). The ratio of this node dimension to the design dimension defines the point at which SCALE=1. SCALE= option values greater than 1 enlarge the nodes (relative to the design dimensions). SCALE= option values less than 1 shrink the nodes (relative to the design dimensions). Hence, you can accommodate more nodes (variables) in your path diagram by setting smaller SCALE= option values.

If you use the GRIP layout algorithm, PROC FACTOR automatically adjusts the SCALE= value according to the number of nodes in the path diagram, as shown in the following table:

Number of Nodes	SCALE=
14 or less	1.00
15–19	0.95
20–24	0.90
25–29	0.85
30–34	0.80
35–39	0.75
40–44	0.70
45–49	0.65
50–59	0.60
60–69	0.55
70 or more	0.50

Although these values yield reasonable relative node sizes in different situations, you can always adjust the relative node size by setting the SCALE= option value manually. For example, if you have 33 nodes in your path diagram and some nodes appear to be overlapping, then you can consider setting a SCALE= option value that is less than 0.8. When you are satisfied with the SCALE= option value, you can then improve the resolution of the path diagram by using the DESIGNHEIGHT= and DESIGNWIDTH= options.

TITLE=*name*

specifies the title of the path diagram. You can use any valid SAS name or a quoted string of up to 256 characters for *name*. If you do not specify this option, PROC FACTOR uses "Path Diagram" for the title when there is only one path diagram. A unique number (for example, "Path Diagram 3") is appended to the title of each path diagram when there is more than one path diagram.

PRIORS Statement

PRIORS *communalities* **;**

The PRIORS statement specifies numeric values between 0.0 and 1.0 for the prior communality estimates for each variable. The first numeric value corresponds to the first variable in the VAR statement, the second value to the second variable, and so on. The number of numeric values must equal the number of variables. For example:

```
proc factor;
   var     x  y  z;
   priors .7 .8 .9;
run;
```

You can specify various methods for computing prior communality estimates with the PRIORS= option in the PROC FACTOR statement. See the description of that option for more information about the default prior communality estimates.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the numeric variables to be analyzed. If the VAR statement is omitted, all numeric variables not specified in other statements are analyzed.

WEIGHT Statement

WEIGHT *variable* **;**

If you want to use relative weights for each observation in the input data set, specify a variable containing weights in a WEIGHT statement. This is often done when the variance associated with each observation is different and the values of the weight variable are proportional to the reciprocals of the variances. If a variable value is negative or is missing, it is excluded from the analysis.

Details: FACTOR Procedure

Input Data Set

The FACTOR procedure can read an ordinary SAS data set containing raw data or a special data set specified as a TYPE=CORR, TYPE=UCORR, TYPE=SSCP, TYPE=COV, TYPE=UCOV, or TYPE=FACTOR data set containing previously computed statistics. A TYPE=CORR data set can be created by the CORR procedure or various other procedures such as the PRINCOMP procedure. It contains means, standard deviations, the sample size, the correlation matrix, and possibly other statistics if it is created by some procedure other than PROC CORR. A TYPE=COV data set is similar to a TYPE=CORR data set but contains a covariance matrix. A TYPE=UCORR or TYPE=UCOV data set contains a correlation or covariance matrix that is not corrected for the mean. The default VAR variable list does not include Intercept if the DATA= data set is TYPE=SSCP. If the Intercept variable is explicitly specified in the VAR statement with a TYPE=SSCP data set, the NOINT option is activated. A TYPE=FACTOR data set can be created by the FACTOR procedure and is described in the section "Output Data Sets" on page 2902.

If your data set has many observations and you plan to run FACTOR several times, you can save computer time by first creating a TYPE=CORR data set and using it as input to PROC FACTOR, as in the following statements:

```
proc corr data=raw out=correl;     /* create TYPE=CORR data set */
proc factor data=correl method=ml; /* maximum likelihood        */
proc factor data=correl;           /* principal components      */
```

The data set created by the CORR procedure is automatically given the TYPE=CORR data set option, so you do not have to specify TYPE=CORR. However, if you use a DATA step with a SET statement to modify the correlation data set, you must use the TYPE=CORR attribute in the new data set. You can use a VAR statement with PROC FACTOR when reading a TYPE=CORR data set to select a subset of the variables or change the order of the variables.

Problems can arise from using the CORR procedure when there are missing data. By default, PROC CORR computes each correlation from all observations that have values present for the pair of variables involved (pairwise deletion). The resulting correlation matrix might have negative eigenvalues. If you specify the NOMISS option with the CORR procedure, observations with any missing values are completely omitted from the calculations (listwise deletion), and there is no danger of negative eigenvalues.

PROC FACTOR can also create a TYPE=FACTOR data set, which includes all the information in a TYPE=CORR data set, and use it for repeated analyses. For a TYPE=FACTOR data set, the default value of the METHOD= option is PATTERN. The following PROC FACTOR statements produce the same results as the previous example:

```
proc factor data=raw method=ml outstat=fact; /* max. likelihood */
proc factor data=fact method=prin;      /* principal components */
```

You can use a TYPE=FACTOR data set to try several different rotation methods on the same data without repeatedly extracting the factors. In the following example, the second and third PROC FACTOR statements use the data set fact created by the first PROC FACTOR statement:

```
proc factor data=raw outstat=fact; /* principal components */
proc factor rotate=varimax;        /* varimax rotation     */
proc factor rotate=quartimax;      /* quartimax rotation   */
```

You can create a TYPE=CORR, TYPE=UCORR, or TYPE=FACTOR data set in a DATA step for PROC FACTOR to read as input. For example, in the following a TYPE=CORR data set is created and is read as input data set by the subsequent PROC FACTOR statement:

```
data correl(type=corr);
   _TYPE_='CORR';
   input _NAME_ $ x y z;
   datalines;
x  1.0  .   .
y   .7 1.0  .
z   .5  .4 1.0
;
proc factor;
run;
```

Be sure to specify the TYPE= option in parentheses after the data set name in the DATA statement and include the _TYPE_ and _NAME_ variables. In a TYPE=CORR data set, only the correlation matrix (_TYPE_='CORR') is necessary. It can contain missing values as long as every pair of variables has at least one nonmissing value.

You can also create a TYPE=FACTOR data set containing only a factor pattern (_TYPE_='PATTERN') and use the FACTOR procedure to rotate it, as these statements show:

```
data pat(type=factor);
   _TYPE_='PATTERN';
   input _NAME_ $ x y z;
   datalines;
factor1  .5  .7  .3
factor2  .8  .2  .8
;
proc factor rotate=promax prerotate=none;
run;
```

If the input factors are oblique, you must also include the interfactor correlation matrix with _TYPE_='FCORR', as shown here:

```
data pat(type=factor);
   input _TYPE_ $ _NAME_ $ x y z;
   datalines;
pattern factor1  .5  .7  .3
pattern factor2  .8  .2  .8
fcorr   factor1 1.0  .2  .
fcorr   factor2  .2 1.0  .
;
proc factor rotate=promax prerotate=none;
run;
```

Some procedures, such as the PRINCOMP and CANDISC procedures, produce TYPE=CORR or TYPE=UCORR data sets containing scoring coefficients (_TYPE_='SCORE' or _TYPE_= 'USCORE'). These coefficients can be input to PROC FACTOR and rotated by using the METHOD=SCORE option, as in

the following statements:

```
proc princomp data=raw n=2 outstat=prin;
run;
proc factor data=prin method=score rotate=varimax;
run;
```

Notice that the input data set prin must contain the correlation matrix as well as the scoring coefficients.

Output Data Sets

The OUT= Data Set

The OUT= data set contains all the data in the DATA= data set plus new variables called Factor1, Factor2, and so on, containing estimated factor scores. Each estimated factor score is computed as a linear combination of the standardized values of the variables that are factored. The coefficients are always displayed if the OUT= option is specified, and they are labeled "Standardized Scoring Coefficients."

If partial variables are specified in the PARTIAL statement, the factor analysis is on the residuals of the variables, which are regressed on the partial variables. In this case, the OUT= data set also contains the (unstandardized) residuals, which are prefixed by R_ by default. For example, the residual of variable X is named R_X in the OUT= data set. You might also assign the prefix by the PARPREFIX= option. Because the residuals are factor-analyzed, the estimated factor scores are computed as linear combinations of the standardized values of the residuals, but not the original variables.

The OUTSTAT= Data Set

The OUTSTAT= data set is similar to the TYPE=CORR or TYPE=UCORR data set produced by the CORR procedure, but it is a TYPE=FACTOR data set and it contains many results in addition to those produced by PROC CORR. The OUTSTAT= data set contains observations with _TYPE_='UCORR' and _TYPE_='USTD' if you specify the NOINT option.

The output data set contains the following variables:

- the BY variables, if any
- two new character variables, _TYPE_ and _NAME_
- the variables analyzed—those in the VAR statement, or, if there is no VAR statement, all numeric variables not listed in any other statement. If partial variables are specified in the PARTIAL statement, the residuals are included instead. By default, the residual variable names are prefixed by R_, unless you specify something different in the PARPREFIX= option.

Each observation in the output data set contains some type of statistic as indicated by the _TYPE_ variable. The _NAME_ variable is blank except where otherwise indicated. The values of the _TYPE_ variable are as follows:

MEAN	means
STD	standard deviations

USTD	uncorrected standard deviations
N	sample size
CORR	correlations. The _NAME_ variable contains the name of the variable corresponding to each row of the correlation matrix.
UCORR	uncorrected correlations. The _NAME_ variable contains the name of the variable corresponding to each row of the uncorrected correlation matrix.
IMAGE	image coefficients. The _NAME_ variable contains the name of the variable corresponding to each row of the image coefficient matrix.
IMAGECOV	image covariance matrix. The _NAME_ variable contains the name of the variable corresponding to each row of the image covariance matrix.
COMMUNAL	final communality estimates
PRIORS	prior communality estimates, or estimates from the last iteration for iterative methods
WEIGHT	variable weights
SUMWGT	sum of the variable weights
EIGENVAL	eigenvalues
UNROTATE	unrotated factor pattern. The _NAME_ variable contains the name of the factor.
SE_UNROT	standard error estimates for the unrotated loadings. The _NAME_ variable contains the name of the factor.
RESIDUAL	residual correlations. The _NAME_ variable contains the name of the variable corresponding to each row of the residual correlation matrix.
PRETRANS	transformation matrix from prerotation. The _NAME_ variable contains the name of the factor.
PREFCORR	prerotated interfactor correlations. The _NAME_ variable contains the name of the factor.
SE_PREFC	standard error estimates for prerotated interfactor correlations. The _NAME_ variable contains the name of the factor.
PREROTAT	prerotated factor pattern. The _NAME_ variable contains the name of the factor.
SE_PREPA	standard error estimates for the prerotated loadings. The _NAME_ variable contains the name of the factor.
PRERCORR	prerotated reference axis correlations. The _NAME_ variable contains the name of the factor.
PREREFER	prerotated reference structure. The _NAME_ variable contains the name of the factor.
PRESTRUC	prerotated factor structure. The _NAME_ variable contains the name of the factor.
SE_PREST	standard error estimates for prerotated structure loadings. The _NAME_ variable contains the name of the factor.
PRESCORE	prerotated scoring coefficients. The _NAME_ variable contains the name of the factor.
TRANSFOR	transformation matrix from rotation. The _NAME_ variable contains the name of the factor.
FCORR	interfactor correlations. The _NAME_ variable contains the name of the factor.

SE_FCORR	standard error estimates for interfactor correlations. The _NAME_ variable contains the name of the factor.
PATTERN	factor pattern. The _NAME_ variable contains the name of the factor.
SE_PAT	standard error estimates for the rotated loadings. The _NAME_ variable contains the name of the factor.
RCORR	reference axis correlations. The _NAME_ variable contains the name of the factor.
REFERENC	reference structure. The _NAME_ variable contains the name of the factor.
STRUCTUR	factor structure. The _NAME_ variable contains the name of the factor.
SE_STRUC	standard error estimates for structure loadings. The _NAME_ variable contains the name of the factor.
SCORE	scoring coefficients to be applied to standardized variables if the SCORE option is specified on the PROC FACTOR statement. The _NAME_ variable contains the name of the factor.
USCORE	scoring coefficients to be applied without subtracting the mean from the raw variables if the SCORE option is specified on the PROC FACTOR statement. The _NAME_ variable contains the name of the factor.

Number of Factors to Retain

PROC FACTOR provides a variety of factor extraction methods, which you can specify by using the METHOD= option. Typically, the maximum number of factors that can be extracted is the same as the number of variables that are being factor-analyzed. Because factor analysis can be viewed as a data-reduction technique, the actual number of factors that researchers want to retain after the extraction is usually much smaller.

PROC FACTOR provides the following three options to enable you to control the number of factors to retain in factor solutions: MINEIGEN=, NFACTORS=, and PROPORTION=. Although each of the MINEIGEN= and PROPORTION= options specifies a criterion value to determine the number of factors, the NFACTORS=*n* option specifies an explicit number *n* for the maximum number of factors to retain. In addition, you can use the NFACTORS=*name* option to request that either the minimum average partial correlation (MAP) analysis or parallel analysis criterion be used for determining the number of factors. The MAP and parallel analysis criteria are explained at the end of this section.

If you do not specify any of these options, PROC FACTOR essentially sets default criterion values for the MINEIGEN= and PROPORTION= options and retains the minimum number of factors that can satisfy either one of these criteria. See the MINEIGEN= and PROPORTION= options for the default criterion values, which depend on the factor extraction method that you use.

If you specify two or more options among the MINEIGEN=, NFACTORS=*n*, and PROPORTION= options, the number of factors retained is the minimum number that satisfies any of the criteria. However, if you specify one of the options NFACTORS=MAP, NFACTORS=MAP2, NFACTORS=MAP4, or NFACTORS=PARALLEL, the retained number of factors is determined by the specified option (MAP, MAP2, MAP4, or PARALLEL, respectively) and all other criteria are ignored. The following explains the MAP and parallel analysis criteria.

A minimum average partial correlation analysis first performs a principal components analysis on the sample correlation matrix. Next, averages of the squared (for MAP2) or fourth-powered (for MAP4) partial correlations between the variables are computed after recursively controlling for one, two, and so on, components (Velicer 1976). The selected number of factors is the number of the controlled components that corresponds to the smallest average squared or fourth-powered partial correlation. If you want to conduct a minimum average partial correlation analysis without using its criterion to determine the number of factors to extract, use the MAP option instead of the NFACTORS= option.

As first proposed by Horn (1965), a parallel analysis selects the number of factors by comparing the eigenvalues in the sample correlation matrix with the eigenvalues of random correlation matrices that are generated from a multivariate standard normal distribution. PROC FACTOR simulates a large number of random correlation matrices and constructs an empirical distribution for each positional eigenvalue: largest, second-largest, and so on (Glorfeld 1995). A factor is suggested if an observed eigenvalue is greater than the critical value at a specified one-sided α-level, with reference to the corresponding simulated distribution of random eigenvalues. As soon as an observed eigenvalue is less than or equal to the corresponding critical value, no more factors are counted. In other words, only the first n consecutive significant eigenvalues are counted for the number of factors.

If you want to conduct a parallel analysis without using its criterion to determine the number of factors being extracted, use the PARALLEL option instead of the NFACTORS=PARALLEL option.

Variable Weights and Variance Explained

A principal component analysis of a correlation matrix treats all variables as equally important. A principal component analysis of a covariance matrix gives more weight to variables with larger variances. A principal component analysis of a covariance matrix is equivalent to an analysis of a weighted correlation matrix, where the weight of each variable is equal to its variance. Variables with large weights tend to have larger loadings on the first component and smaller residual correlations than variables with small weights.

You might want to give weights to variables by using values other than their variances. Mulaik (1972) explains how to obtain a maximally reliable component by means of a weighted principal component analysis. With the FACTOR procedure, you can indirectly give arbitrary weights to the variables by using the COV option and rescaling the variables to have variance equal to the desired weight, or you can give arbitrary weights directly by using the WEIGHT option and including the weights in a TYPE=CORR data set.

Arbitrary variable weights can be used with the METHOD=PRINCIPAL, METHOD=PRINIT, METHOD=ULS, or METHOD=IMAGE option. Alpha and ML factor analyses compute variable weights based on the communalities (Harman 1976, pp. 217–218). For alpha factor analysis, the weight of a variable is the reciprocal of its communality. In ML factor analysis, the weight is the reciprocal of the uniqueness. Harris component analysis uses weights equal to the reciprocal of one minus the squared multiple correlation of each variable with the other variables.

For uncorrelated factors, the variance explained by a factor can be computed with or without taking the weights into account. The usual method for computing variance accounted for by a factor is to take the sum of squares of the corresponding column of the factor pattern, yielding an unweighted result. If the square of each loading is multiplied by the weight of the variable before the sum is taken, the result is the weighted variance explained, which is equal to the corresponding eigenvalue except in image analysis. Whether the weighted or unweighted result is more important depends on the purpose of the analysis.

In the case of correlated factors, the variance explained by a factor can be computed with or without taking the other factors into account. If you want to ignore the other factors, the variance explained is given by the weighted or unweighted sum of squares of the appropriate column of the factor structure since the factor structure contains simple correlations. If you want to subtract the variance explained by the other factors from the amount explained by the factor in question (the Type II variance explained), you can take the weighted or unweighted sum of squares of the appropriate column of the reference structure because the reference structure contains semipartial correlations. There are other ways of measuring the variance explained. For example, given a prior ordering of the factors, you can eliminate from each factor the variance explained by previous factors and compute a Type I variance explained. Harman (1976, pp. 268–270) provides another method, which is based on direct and joint contributions.

Simplicity Functions for Rotations

To rotate a factor pattern is to apply a nonsingular linear transformation to the unrotated factor pattern matrix. An optimal transformation is usually defined as a minimum or maximum point of a simplicity function. Different rotation methods are based on different simplicity functions employed.

For the promax or the Procrustes rotation, the simplicity function used is the sum of squared differences between the rotated factor pattern and the target matrix. The optimal transformation is obtained by minimizing this simplicity function with respect to the choices of all possible transformation.

For the class of the generalized Crawford-Ferguson family (Jennrich 1973), the simplicity function being optimized is

$$f = k_1 Z + k_2 H + k_3 V + k_4 Q$$

where

$$Z = (\sum_j \sum_i b_{ij}^2)^2, \quad H = \sum_i (\sum_j b_{ij}^2)^2$$

$$V = \sum_j (\sum_i b_{ij}^2)^2, \quad Q = \sum_j \sum_i b_{ij}^4$$

k_1, k_2, k_3, and k_4 are constants, and b_{ij} represents an element of the rotated pattern matrix. Except for specialized research purposes, it is uncommon in practice to use this simplicity function directly for rotation. However, this simplicity function reduces to many well-known classes of rotations. One of these is the Crawford-Ferguson family (Crawford and Ferguson 1970), which minimizes

$$f_{cf} = c_1(H - Q) + c_2(V - Q)$$

where c_1 and c_2 are constants, $(H - Q)$ represents variable (row) parsimony, and $(V - Q)$ represents factor (column) parsimony. Therefore, the relative importance of both the variable parsimony and of the factor parsimony is adjusted using the constants c_1 and c_2. The orthomax class (see Harman 1976) maximizes the function

$$f_{or} = pQ - \gamma V$$

where γ is the orthomax weight and is usually between 0 and the number of variables p. The oblimin class minimizes the function

$$f_{ob} = p(H - Q) - \tau(Z - V)$$

where τ is the oblimin weight. For practical purposes, a negative or zero value for τ is recommended.

All of the preceding definitions are for rotations without row normalization. For rotations with Kaiser normalization, the definition of b_{ij} is replaced by b_{ij}/h_i, where h_i is the communality estimate of variable i.

Confidence Intervals and the Salience of Factor Loadings

The traditional approach to determining salient loadings (loadings that are considered large in absolute values) employs rules of thumb such as 0.3 or 0.4. However, this does not use the statistical evidence efficiently. The asymptotic normality of the distribution of factor loadings enables you to construct confidence intervals to gauge the salience of factor loadings. To guarantee the range-respecting properties of confidence intervals, a transformation procedure such as in CEFA (Browne et al. 2010) is used. For example, because the orthogonal rotated factor loading θ must be bounded between –1 and +1, the Fisher transformation

$$\varphi = \frac{1}{2}\log(\frac{1+\theta}{1-\theta})$$

is employed so that φ is an unbounded parameter. Assuming the asymptotic normality of $\hat{\varphi}$, a symmetric confidence interval for φ is constructed. Then, a back-transformation on the confidence limits yields an asymmetric confidence interval for θ. Applying the results of Browne (1982), a $(1-\alpha)100\%$ confidence interval for the orthogonal factor loading θ is

$$(\hat{\theta}_l = \frac{a/b-1}{a/b+1},\ \hat{\theta}_u = \frac{a\times b-1}{a\times b+1})$$

where

$$a = \frac{1+\hat{\theta}}{1-\hat{\theta}},\ \ b = \exp(z_{\alpha/2}\times\frac{2\hat{\sigma}}{1-\hat{\theta}^2})$$

and $\hat{\theta}$ is the estimated factor loading, $\hat{\sigma}$ is the standard error estimate of the factor loading, and $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile point of a standard normal distribution.

Once the confidence limits are constructed, you can use the corresponding coverage displays for determining the salience of the variable-factor relationship. In a coverage display, the COVER= value is represented by an asterisk (*). The following table summarizes various displays and their interpretations.

Table 44.5 Interpretations of the Coverage Displays

Positive Estimate	Negative Estimate	COVER=0 Specified	Interpretation
[0]*	*[0]		The estimate is not significantly different from zero, and the CI covers a region of values that are smaller in magnitude than the COVER= value. This is strong statistical evidence for the nonsalience of the variable-factor relationship.
0[ ]*	*[ ]0		The estimate is significantly different from zero, but the CI covers a region of values that are smaller in magnitude than the COVER= value. This is strong statistical evidence for the nonsalience of the variable-factor relationship.

Table 44.5 *continued*

Positive Estimate	Negative Estimate	COVER=0 Specified	Interpretation
[0*]	[*0]	[0]	The estimate is not significantly different from zero or the COVER= value. The population value might have been larger or smaller in magnitude than the COVER= value. There is no statistical evidence for the salience of the variable-factor relationship.
0[*]	[*]0		The estimate is significantly different from zero but not from the COVER= value. This is marginal statistical evidence for the salience of the variable-factor relationship.
0*[ ]	[ ]*0	0[ ] or [ ]0	The estimate is significantly different from zero, and the CI covers a region of values that are larger in magnitude than the COVER= value. This is strong statistical evidence for the salience of the variable-factor relationship.

See Example 44.5 for an illustration of the use of confidence intervals for interpreting factors.

Factor Scores

The FACTOR procedure can compute estimated factor scores directly if you specify the NFACTORS=*n* and OUT= options, or indirectly using the SCORE procedure. The latter method is preferable if you use the FACTOR procedure interactively to determine the number of factors, the rotation method, or various other aspects of the analysis. To compute factor scores for each observation by using the SCORE procedure, do the following:

- Use the SCORE option in the PROC FACTOR statement.
- Create a TYPE=FACTOR output data set with the OUTSTAT= option.
- Use the SCORE procedure with both the raw data and the TYPE=FACTOR data set.
- Do not use the TYPE= option in the PROC SCORE statement.

For example, the following statements could be used:

```
proc factor data=raw score outstat=fact;
run;
proc score data=raw score=fact out=scores;
run;
```

or

```
proc corr   data=raw out=correl;
run;
proc factor data=correl score outstat=fact;
run;
proc score  data=raw score=fact out=scores;
run;
```

For a more detailed example, see Example 109.1 in Chapter 109, "The SCORE Procedure."

A component analysis (principal, image, or Harris) produces scores with mean zero and variance one. If you have done a common factor analysis, the true factor scores have mean zero and variance one, but the computed factor scores are only estimates of the true factor scores. These estimates have mean zero but variance equal to the squared multiple correlation of the factor with the variables. The estimated factor scores might have small nonzero correlations even if the true factors are uncorrelated.

Heywood Cases and Other Anomalies about Communality Estimates

Since communalities are squared correlations, you would expect them always to lie between 0 and 1. It is a mathematical peculiarity of the common factor model, however, that final communality estimates might exceed 1. If a communality equals 1, the situation is referred to as a Heywood case, and if a communality exceeds 1, it is an ultra-Heywood case. An ultra-Heywood case implies that some unique factor has negative variance, a clear indication that something is wrong. Possible causes include the following:

- bad prior communality estimates
- too many common factors
- too few common factors
- not enough data to provide stable estimates
- the common factor model is not an appropriate model for the data

An ultra-Heywood case renders a factor solution invalid. Factor analysts disagree about whether or not a factor solution with a Heywood case can be considered legitimate.

With METHOD=PRINIT, METHOD=ULS, METHOD=ALPHA, or METHOD=ML, the FACTOR procedure, by default, stops iterating and sets the number of factors to 0 if an estimated communality exceeds 1. To enable processing to continue with a Heywood or ultra-Heywood case, you can use the HEYWOOD or ULTRAHEYWOOD option in the PROC FACTOR statement. The HEYWOOD option sets the upper bound of any communality to 1, while the ULTRAHEYWOOD option allows communalities to exceed 1.

Theoretically, the communality of a variable should not exceed its reliability. Violation of this condition is called a quasi-Heywood case and should be regarded with the same suspicion as an ultra-Heywood case.

Elements of the factor structure and reference structure matrices can exceed 1 only in the presence of an ultra-Heywood case. On the other hand, an element of the factor pattern might exceed 1 in an oblique rotation.

The maximum likelihood method is especially susceptible to quasi- or ultra-Heywood cases. During the iteration process, a variable with high communality is given a high weight; this tends to increase its communality, which increases its weight, and so on.

It is often stated that the squared multiple correlation of a variable with the other variables is a lower bound to its communality. This is true if the common factor model fits the data perfectly, but it is not generally the case with real data. A final communality estimate that is less than the squared multiple correlation can, therefore, indicate poor fit, possibly due to not enough factors. It is by no means as serious a problem as an ultra-Heywood case. Factor methods that use the Newton-Raphson method can actually produce communalities less than 0, a result even more disastrous than an ultra-Heywood case.

The squared multiple correlation of a factor with the variables might exceed 1, even in the absence of ultra-Heywood cases. This situation is also cause for alarm. Alpha factor analysis seems to be especially prone to this problem, but it does not occur with maximum likelihood. If a squared multiple correlation is negative, there are too many factors retained.

With data that do not fit the common factor model perfectly, you can expect some of the eigenvalues to be negative. If an iterative factor method converges properly, the sum of the eigenvalues corresponding to rejected factors should be 0; hence, some eigenvalues are positive and some negative. If a principal factor analysis fails to yield any negative eigenvalues, the prior communality estimates are probably too large. Negative eigenvalues cause the cumulative proportion of variance explained to exceed 1 for a sufficiently large number of factors. The cumulative proportion of variance explained by the retained factors should be approximately 1 for principal factor analysis and should converge to 1 for iterative methods. Occasionally, a single factor can explain more than 100 percent of the common variance in a principal factor analysis, indicating that the prior communality estimates are too low.

If a squared canonical correlation or a coefficient alpha is negative, there are too many factors retained.

Principal component analysis, unlike common factor analysis, has none of these problems if the covariance or correlation matrix is computed correctly from a data set with no missing values. Various methods for missing value correlation or severe rounding of the correlations can produce negative eigenvalues in principal components.

Missing Values

If the DATA= data set contains data (rather than a matrix or factor pattern), then observations with missing values for any variables in the analysis are omitted from the computations. If a correlation or covariance matrix is read, it can contain missing values as long as every pair of variables has at least one nonmissing entry. Missing values in a pattern or scoring coefficient matrix are treated as zeros.

Cautions

- The amount of time that FACTOR takes is roughly proportional to the cube of the number of variables. Factoring 100 variables, therefore, takes about 1,000 times as long as factoring 10 variables. Iterative methods (PRINIT, ALPHA, ULS, ML) can also take 100 times as long as noniterative methods (PRINCIPAL, IMAGE, HARRIS).
- No computer program is capable of reliably determining the optimal number of factors, since the decision is ultimately subjective. You should not blindly accept the number of factors obtained by default; instead, use your own judgment to make a decision.
- Singular correlation matrices cause problems with the options MSA, PRIORS=ASMC, PRIORS=SMC, and METHOD=ML. Singularities can result from using a variable that is the sum of other variables, coding too many dummy variables from a classification variable, or having more variables than observations.
- If you use the CORR procedure to compute the correlation matrix and there are missing data and the NOMISS option is not specified, then the correlation matrix might have negative eigenvalues.
- If a TYPE=CORR, TYPE=UCORR, or TYPE=FACTOR data set is copied or modified using a DATA step, the new data set does not automatically have the same TYPE as the old data set. You must specify the TYPE= data set option in the DATA statement. If you try to analyze a data set that has lost its TYPE=CORR attribute, PROC FACTOR displays a warning message saying that the data set contains _NAME_ and _TYPE_ variables but analyzes the data set as an ordinary SAS data set.
- For a TYPE=FACTOR data set, the default is METHOD=PATTERN, not METHOD=PRIN.

Time Requirements

n = number of observations
v = number of variables
f = number of factors
i = number of iterations during factor extraction
r = length of iterations during factor rotation

The time required to compute...	Is roughly proportional to
an overall factor analysis	iv^3
the correlation matrix	nv^2
PRIORS=SMC or ASMC	v^3
PRIORS=MAX	v^2
eigenvalues	v^3
final eigenvectors	fv^2
generalized Crawford-Ferguson family of rotations, PROMAX, or HK	rvf^2
ROTATE=PROCRUSTES	vf^2

Each iteration in the PRINIT or ALPHA method requires computation of eigenvalues and f eigenvectors.

Each iteration in the ML or ULS method requires computation of eigenvalues and $v - f$ eigenvectors.

The amount of time that PROC FACTOR takes is roughly proportional to the cube of the number of variables. Factoring 100 variables, therefore, takes about 1000 times as long as factoring 10 variables. Iterative methods (PRINIT, ALPHA, ULS, ML) can also take 100 times as long as noniterative methods (PRINCIPAL, IMAGE, HARRIS).

Displayed Output

PROC FACTOR output includes the following:

- Input data type, numbers of records read and used for raw data input, the number of observations that is specified in the NOBS= option in the PROC FACTOR statement, and the number of observations used in significance tests
- Mean and Std Dev (standard deviation) of each variable and the number of observations, if you specify the SIMPLE option
- Correlations, if you specify the CORR option
- Inverse Correlation Matrix, if you specify the ALL option
- Partial Correlations Controlling all other Variables (negative anti-image correlations), if you specify the MSA option. If the data are appropriate for the common factor model, the partial correlations should be small.
- Kaiser's Measure of Sampling Adequacy (Kaiser 1970; Kaiser and Rice 1974; Cerny and Kaiser 1977), both overall and for each variable, if you specify the MSA option. The MSA is a summary of how small the partial correlations are relative to the ordinary correlations. Values greater than 0.8 can be considered good. Values less than 0.5 require remedial action, either by deleting the offending variables or by including other variables related to the offenders.

- Prior Communality Estimates, unless 1.0s are used or unless you specify the METHOD=IMAGE, METHOD=HARRIS, METHOD=PATTERN, or METHOD=SCORE option
- Squared Multiple Correlations of each variable with all the other variables, if you specify the METHOD=IMAGE or METHOD=HARRIS option
- Image Coefficients, if you specify the METHOD=IMAGE option
- Image Covariance Matrix, if you specify the METHOD=IMAGE option
- Preliminary Eigenvalues based on the prior communalities, if you specify the METHOD=PRINIT, METHOD=ALPHA, METHOD=ML, or METHOD=ULS option. The table produced includes the Total and the Average of the eigenvalues, the Difference between successive eigenvalues, the Proportion of variation represented, and the Cumulative proportion of variation.
- Parallel Analysis, if you specify the PARALLEL or NFACTORS=PARALLEL option. This includes the eigenvalues of the proper correlation matrix, the simulated critical values from the parallel analysis, the simulated average eigenvalues from the parallel analysis, and the number of threads used to construct the parallel analysis. (By default, the simulated average eigenvalues and the number of threads are not printed.)
- Average Partial Correlations (squared and fourth-powered) after controlling for different numbers of principal components, if you specify the MAP, NFACTORS=MAP2, or NFACTORS=MAP4 option.
- the number of factors that are retained, unless you specify the METHOD=PATTERN or METHOD=SCORE option
- the Scree Plot of Eigenvalues, if you specify the SCREE option. The preliminary eigenvalues are used if you specify the METHOD=PRINIT, METHOD=ALPHA, METHOD=ML, or METHOD=ULS option. You can request the corresponding high-quality graphical plot by using the PLOTS= option.
- the iteration history, if you specify the METHOD=PRINIT, METHOD=ALPHA, METHOD=ML, or METHOD=ULS option. The table produced contains the iteration number (Iter); the Criterion being optimized (Jöreskog 1977); the Ridge value for the iteration if you specify the METHOD=ML or METHOD=ULS option; the maximum Change in any communality estimate; and the Communalities.
- Significance tests, if you specify the option METHOD=ML, including Bartlett's chi-square, df, and Prob $> \chi^2$ for H_0: No common factors and H_0: Factors retained are sufficient to explain the correlations. The H_0 test for no common factors is equivalent to Bartlett's test of sphericity. The variables should have an approximate multivariate normal distribution for the probability levels to be valid. Lawley and Maxwell (1971) suggest that the number of observations should exceed the number of variables by 50 or more, although Geweke and Singleton (1980) claim that as few as 10 observations are adequate with five variables and one common factor. Certain regularity conditions must also be satisfied for Bartlett's χ^2 test to be valid (Geweke and Singleton 1980), but in practice these conditions are usually satisfied. The notation Prob>chi**2 means "the probability under the null hypothesis of obtaining a greater χ^2 statistic than that observed." The chi-square value is displayed with and without Bartlett's correction.
- Akaike's Information Criterion, if you specify the METHOD=ML option. Akaike's information criterion (AIC) (Akaike 1973, 1974, 1987) is a general criterion for estimating the best number of parameters to include in a model when maximum likelihood estimation is used. The number of factors that yields the smallest value of AIC is considered best. Like the chi-square test, AIC tends to include factors that are statistically significant but inconsequential for practical purposes.

- Schwarz's Bayesian Criterion, if you specify the METHOD=ML option. Schwarz's Bayesian Criterion (SBC) (Schwarz 1978) is another criterion, similar to AIC, for determining the best number of parameters. The number of factors that yields the smallest value of SBC is considered best; SBC seems to be less inclined to include trivial factors than either AIC or the chi-square test.
- Tucker and Lewis's reliability coefficient, if you specify the METHOD=ML option (Tucker and Lewis 1973)
- Squared Canonical Correlations, if you specify the METHOD=ML option. These are the same as the squared multiple correlations for predicting each factor from the variables.
- Coefficient Alpha for Each Factor, if you specify the METHOD=ALPHA option
- Eigenvectors, if you specify the EIGENVECTORS or ALL option, unless you also specify the METHOD=PATTERN or METHOD=SCORE option
- Eigenvalues of the (Weighted) (Reduced) (Image) Correlation or Covariance Matrix, unless you specify the METHOD=PATTERN or METHOD=SCORE option. Included are the Total and the Average of the eigenvalues, the Difference between successive eigenvalues, the Proportion of variation represented, and the Cumulative proportion of variation.
- the Factor Pattern, which is equal to both the matrix of standardized regression coefficients for predicting variables from common factors and the matrix of correlations between variables and common factors since the extracted factors are uncorrelated. Standard error estimates are included if the SE option is specified with METHOD=ML. Confidence limits and coverage displays are included if COVER= option is specified with METHOD=ML.
- Variance explained by each factor, both Weighted and Unweighted, if variable weights are used
- Final Communality Estimates, including the Total communality; or Final Communality Estimates and Variable Weights, including the Total communality, both Weighted and Unweighted, if variable weights are used. Final communality estimates are the squared multiple correlations for predicting the variables from the estimated factors, and they can be obtained by taking the sum of squares of each row of the factor pattern, or a weighted sum of squares if variable weights are used.
- Residual Correlations with Uniqueness on the Diagonal, if you specify the RESIDUAL or ALL option
- Root Mean Square Off-Diagonal Residuals, both Over-all and for each variable, if you specify the RESIDUAL or ALL option
- Partial Correlations Controlling Factors, if you specify the RESIDUAL or ALL option
- Root Mean Square Off-Diagonal Partials, both Over-all and for each variable, if you specify the RESIDUAL or ALL option
- Plots of Factor Pattern for unrotated factors, if you specify the PREPLOT option. The number of plots is determined by the NPLOTS= option. You can request the corresponding high-quality graphical plots by using the PLOTS= option.
- Variable Weights for Rotation, if you specify the NORM=WEIGHT option
- Factor Weights for Rotation, if you specify the HKPOWER= option

- Orthogonal Transformation Matrix, if you request an orthogonal rotation
- Rotated Factor Pattern, if you request an orthogonal rotation. Standard error estimates are included if the SE option is specified with METHOD=ML. Confidence limits and coverage displays are included if COVER= option is specified with METHOD=ML.
- Variance explained by each factor after rotation. If you request an orthogonal rotation and if variable weights are used, both weighted and unweighted values are produced.
- Target Matrix for Procrustean Transformation, if you specify the ROTATE=PROMAX or ROTATE=PROCRUSTES option
- the Procrustean Transformation Matrix, if you specify the ROTATE=PROMAX or ROTATE=PROCRUSTES option
- the Normalized Oblique Transformation Matrix, if you request an oblique rotation, which, for the option ROTATE=PROMAX, is the product of the prerotation and the Procrustes rotation
- Inter-factor Correlations, if you specify an oblique rotation. Standard error estimates are included if the SE option is specified with METHOD=ML. Confidence limits and coverage displays are included if COVER= option is specified with METHOD=ML.
- Rotated Factor Pattern (Std Reg Coefs), if you specify an oblique rotation, giving standardized regression coefficients for predicting the variables from the factors. Standard error estimates are included if the SE option is specified with METHOD=ML. Confidence limits and coverage displays are included if COVER= option is specified with METHOD=ML.
- Reference Axis Correlations if you specify an oblique rotation. These are the partial correlations between the primary factors when all factors other than the two being correlated are partialed out.
- Reference Structure (Semipartial Correlations), if you request an oblique rotation. The reference structure is the matrix of semipartial correlations (Kerlinger and Pedhazur 1973) between variables and common factors, removing from each common factor the effects of other common factors. If the common factors are uncorrelated, the reference structure is equal to the factor pattern.
- Variance explained by each factor eliminating the effects of all other factors, if you specify an oblique rotation. Both Weighted and Unweighted values are produced if variable weights are used. These variances are equal to the (weighted) sum of the squared elements of the reference structure corresponding to each factor.
- Factor Structure (Correlations), if you request an oblique rotation. The (primary) factor structure is the matrix of correlations between variables and common factors. If the common factors are uncorrelated, the factor structure is equal to the factor pattern. Standard error estimates are included if the SE option is specified with METHOD=ML. Confidence limits and coverage displays are included if COVER= option is specified with METHOD=ML.
- Variance explained by each factor ignoring the effects of all other factors, if you request an oblique rotation. Both Weighted and Unweighted values are produced if variable weights are used. These variances are equal to the (weighted) sum of the squared elements of the factor structure corresponding to each factor.
- Final Communality Estimates for the rotated factors if you specify the ROTATE= option. The estimates should equal the unrotated communalities.

- Squared Multiple Correlations of the Variables with Each Factor, if you specify the SCORE or ALL option, except for unrotated principal components
- Standardized Scoring Coefficients, if you specify the SCORE or ALL option
- Plots of the Factor Pattern for rotated factors, if you specify the PLOT option and you request an orthogonal rotation. The number of plots is determined by the NPLOTS= option. You can request the corresponding high-quality graphical plots by using the PLOTS= option.
- Plots of the Reference Structure for rotated factors, if you specify the PLOT option and you request an oblique rotation. The number of plots is determined by the NPLOTS= option. Included are the Reference Axis Correlation and the Angle between the Reference Axes for each pair of factors plotted. You can request the corresponding high-quality graphical plots by using the PLOTS= option.
- A path diagram for the final factor solution if you specify the PLOTS=PATHDIAGRAM option or the PATHDIAGRAM statements.

If you specify the ROTATE=PROMAX option, the output includes results for both the prerotation and the Procrustes rotation.

ODS Table Names

PROC FACTOR assigns a name to each table that it creates. You can use these names to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the Table 44.6. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 44.6 ODS Tables Produced by PROC FACTOR

ODS Table Name	Description	Option
AlphaCoef	Coefficient alpha for each factor	METHOD=ALPHA
AveParCorrControlPC	Average partial correlations after controlling for principal components	MAP or NFACTORS=MAP, MAP2, or MAP4
CanCorr	Squared canonical correlations	METHOD=ML
CondStdDev	Conditional standard deviations	SIMPLE with PARTIAL
ConvergenceStatus	Convergence status	METHOD=PRINIT, ALPHA, ML, or ULS
Corr	Correlations	CORR
Eigenvalues	Eigenvalues	Default
Eigenvectors	Eigenvectors	EIGENVECTORS
FactorPattern	Factor pattern	Default except METHOD=PATTERN or SCORE
FactorStructure	Factor structure	ROTATE= any oblique rotation
FactorWeightRotate	Factor weights for rotation	HKPOWER=

Table 44.6 *continued*

ODS Table Name	Description	Option
FinalCommun	Final communalities	Default except METHOD=ALPHA, ML, or HARRIS
FinalCommunWgt	Final communalities with weights	METHOD=ALPHA, ML, or HARRIS; METHOD=IMAGE, PRINCIPAL, PRINIT, or ULS with WEIGHT
FitMeasures	Measures of fit	METHOD=ML
ImageCoef	Image coefficients	METHOD=IMAGE
ImageCov	Image covariance matrix	METHOD=IMAGE
ImageFactors	Image factor matrix	METHOD=IMAGE
InputFactorPattern	Input factor pattern	METHOD=PATTERN with PRINT or ALL
InputScoreCoef	Standardized input scoring coefficients	METHOD=SCORE with PRINT or ALL
InterFactorCorr	Interfactor correlations	ROTATE= any oblique rotation
InvCorr	Inverse correlation matrix	ALL
IterHistory	Iteration history	METHOD=PRINIT, ALPHA, ML, or ULS
MultipleCorr	Squared multiple correlations	METHOD=IMAGE or HARRIS
NObs	Number of records and observations, input data type	Default
NormObliqueTrans	Normalized oblique transformation matrix	ROTATE=PROCRUSTES or PROMAX
ObliqueRotFactPat	Rotated factor pattern	ROTATE= any oblique rotation
ObliqueTrans	Oblique transformation matrix	HKPOWER= or ROTATE= any oblique rotation except PROCRUSTES or PROMAX
OrthRotFactPat	Rotated factor pattern	ROTATE= any orthogonal rotation
OrthTrans	Orthogonal transformation matrix	ROTATE= any orthogonal rotation
ParallelAnalysis	Parallel analysis results	PARALLEL or NFACTORS=PARALLEL
ParCorrControlFactor	Partial correlations after controlling for factors	RESIDUAL
ParCorrControlVar	Partial correlations after controlling for other variables	MSA
PartialCorr	Partial correlations	CORR with PARTIAL
PriorCommunalEst	Prior communality estimates	PRIORS=, METHOD=ML or ALPHA
ProcrustesTarget	Target matrix for Procrustean transformation	ROTATE=PROCRUSTES or PROMAX
ProcrustesTrans	Procrustean transformation matrix	ROTATE=PROCRUSTES or PROMAX
RMSOffDiagPartials	Root mean square off-diagonal partials	RESIDUAL
RMSOffDiagResids	Root mean square off-diagonal residuals	RESIDUAL
ReferenceAxisCorr	Reference axis correlations	ROTATE= any oblique rotation
ReferenceStructure	Reference structure	ROTATE= any oblique rotation

Table 44.6 *continued*

ODS Table Name	Description	Option
ResCorrUniqueDiag	Residual correlations with uniqueness on the diagonal	RESIDUAL
SamplingAdequacy	Kaiser's measure of sampling adequacy	MSA
SignifTests	Significance tests	METHOD=ML
SimpleStatistics	Simple statistics	SIMPLE
StdScoreCoef	Standardized scoring coefficients	SCORE
VarExplain	Variance explained	Default except METHOD=ALPHA, ML, or HARRIS
VarExplainWgt	Variance explained with weights	METHOD=ALPHA, ML, or HARRIS; METHOD=IMAGE, PRINCIPAL, PRINIT, or ULS with WEIGHT
VarFactorCorr	Squared multiple correlations of the variables with each factor	SCORE
VarWeightRotate	Variable weights for rotation	NORM=WEIGHT, ROTATE=

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

The names of the graphs that PROC FACTOR generates are listed in Table 44.7, along with the required statements and options.

Table 44.7 Graphs Produced by PROC FACTOR

ODS Graph Name	Plot Description	Option
InitPatternPlot	Initial factor pattern	PLOTS=INITLOADINGS
InitRefStructurePlot	Initial reference structures	PLOTS=INITLOADINGS and PLOTREF
MAPPlot	Minimum average partial correlation analysis	PLOTS=MAP, MAP2, or MAP4

Table 44.7 *continued*

ODS Graph Name	Plot Description	Option
ParallelPlot	Parallel analysis	PLOTS=PARALLEL
PathDiagram	Path diagram	PLOTS=PATHDIAGRAM or PATHDIAGRAM statement
PatternPlot	Rotated factor pattern	PLOTS=LOADINGS
PrePatternPlot	Prerotated factor pattern	PLOTS=PRELOADINGS
PreRefStructurePlot	Prerotated reference structures	PLOTS=PRELOADINGS and PLOTREF
RefStructurePlot	Rotated reference structures	PLOTS=LOADINGS and PLOTREF
ScreePlot	Scree and variance explained plots	PLOTS=SCREE
VariancePlot	Plot of explained variance	PLOTS=SCREE(UNPACK)

Examples: FACTOR Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 44.1: Principal Component Analysis

(View the complete code for this example (facex1.sas) in the example repository.)

This example analyzes socioeconomic data provided by Harman (1976). The five variables represent total population (Population), median school years (School), total employment (Employment), miscellaneous professional services (Services), and median house value (HouseValue). Each observation represents one of twelve census tracts in the Los Angeles Standard Metropolitan Statistical Area.

You conduct a principal component analysis by using the following statements:

```
data SocioEconomics;
   input Population School Employment Services HouseValue;
   datalines;
5700     12.8      2500      270       25000
1000     10.9      600       10        10000
3400     8.8       1000      10        9000
3800     13.6      1700      140       25000
4000     12.8      1600      140       25000
8200     8.3       2600      60        12000
1200     11.4      400       10        16000
9100     11.5      3300      60        14000
9900     12.5      3400      180       18000
9600     13.7      3600      390       25000
9600     9.6       3300      80        12000
```

```
9400     11.4      4000      100       13000
;

proc factor data=SocioEconomics simple corr;
run;
```

You begin with the specification of the raw data set with 12 observations. Then you use the DATA= option in the PROC FACTOR statement to specify the data set in the analysis. You also set the SIMPLE and CORR options for additional output results, which are shown in Output 44.1.2 and Output 44.1.3, respectively.

By default, PROC FACTOR assumes that all initial communalities are 1, which is the case for the current principal component analysis. If you intend to find common factors instead, use the PRIORS= option or the PRIORS statement to set initial communalities to values less than 1, which results in extracting the principal factors rather than the principal components. See Example 44.2 for the specification of a principal factor analysis.

For the current principal component analysis, the first output table is displayed in the Output 44.1.1.

Output 44.1.1 Principal Component Analysis: Number of Observations

Five Socioeconomic Variables
See Page 14 of Harman: Modern Factor Analysis, 3rd Ed
Principal Component Analysis

The FACTOR Procedure

Input Data Type	Raw Data
Number of Records Read	12
Number of Records Used	12
N for Significance Tests	12

In Output 44.1.1, the input data type is shown to be raw data. PROC FACTOR also accepts other data type such as correlations and covariances. See Example 44.5 for the use of correlations as input data. For the current raw data set, PROC FACTOR reads in 12 records and all these 12 records are used. When there are missing values in the data set, these two numbers might not match due to the dropping of the records with missing values. The last row of the table shows that $N = 12$ is used in the significance tests conducted in the analysis.

The SIMPLE option specified in the PROC FACTOR statement generates the means and standard deviations of all observed variables in the analysis, as shown in Output 44.1.2.

Output 44.1.2 Principal Component Analysis: Simple Statistics

Means and Standard Deviations from 12 Observations		
Variable	**Mean**	**Std Dev**
Population	6241.667	3439.9943
School	11.442	1.7865
Employment	2333.333	1241.2115
Services	120.833	114.9275
HouseValue	17000.000	6367.5313

The ranges of means and standard deviations for the analysis are quite large. Variables are measured on quite different scales. However, this is not an issue because PROC FACTOR basically analyzes the standardized scales (that is, the correlations) of the variables.

The CORR option specified in the PROC FACTOR statement generates the output of the observed correlations in Output 44.1.3.

Output 44.1.3 Principal Component Analysis: Correlations

Correlations					
	Population	**School**	**Employment**	**Services**	**HouseValue**
Population	1.00000	0.00975	0.97245	0.43887	0.02241
School	0.00975	1.00000	0.15428	0.69141	0.86307
Employment	0.97245	0.15428	1.00000	0.51472	0.12193
Services	0.43887	0.69141	0.51472	1.00000	0.77765
HouseValue	0.02241	0.86307	0.12193	0.77765	1.00000

The correlation matrix shown in Output 44.1.3 is analyzed by PROC FACTOR.

The first step of principal component analysis is to look at the eigenvalues of the correlation matrix. The larger eigenvalues are extracted first. Because there are five observed variables, five eigenvalues can be extracted, as shown in Output 44.1.4.

Output 44.1.4 Principal Component Analysis: Eigenvalues

Eigenvalues of the Correlation Matrix: Total = 5 Average = 1				
	Eigenvalue	**Difference**	**Proportion**	**Cumulative**
1	2.87331359	1.07665350	0.5747	0.5747
2	1.79666009	1.58182321	0.3593	0.9340
3	0.21483689	0.11490283	0.0430	0.9770
4	0.09993405	0.08467868	0.0200	0.9969
5	0.01525537		0.0031	1.0000

In Output 44.1.4, the two largest eigenvalues are 2.8733 and 1.7967, which together account for 93.4% of the standardized variance. Thus, the first two principal components provide an adequate summary of the data for most purposes. Three components, which explain 97.7% of the variation, should be sufficient for almost any application. PROC FACTOR retains the first two components on the basis of the eigenvalues-greater-than-one rule since the third eigenvalue is only 0.2148.

To express the observed variables as functions of the components (or factors, in general), you consult the factor loading matrix as shown in Output 44.1.5.

Output 44.1.5 Principal Component Analysis: Factor Pattern

Factor Pattern		
	Factor1	Factor2
Population	0.58096	0.80642
School	0.76704	-0.54476
Employment	0.67243	0.72605
Services	0.93239	-0.10431
HouseValue	0.79116	-0.55818

The factor pattern is often referred to as the factor loading matrix in factor analysis. The elements in the loading matrix are called factor loadings. There are at least two ways you can interpret these factor loadings. First, you can use this table to express the observed variables as functions of the extracted factors (or components, as in the current analysis). Each row of the factor loadings tells you the linear combination of the factor or component scores that would yield the expected value of the associated variable. Second, you can interpret each loading as a correlation between an observed variable and a factor or component, provided that the factor solution is an orthogonal one (that is, factors are uncorrelated), such as the current initial factor solution. Hence, the factor loadings indicate how strongly the variables and the factors or components are related.

In Output 44.1.5, the first component (labeled "Factor1") has large positive loadings for all five variables. Its correlation with Services (0.9324) is especially high. The second component is basically a contrast of Population (0.8064) and Employment (0.7261) against School (–0.5448) and HouseValue (–0.5582), with a very small loading on Services (–0.1043).

The total variance explained by the two components are shown in Output 44.1.6.

Output 44.1.6 Principal Component Analysis: Total Variance Explained by Factors

Variance Explained by Each Factor	
Factor1	Factor2
2.8733136	1.7966601

The first and second component account for 2.8733 and 1.7967, respectively, of the total variance of 5. In the initial factor solution, the total variance explained by the factors or components are the same as the eigenvalues extracted. (Compare the total variance with the eigenvalues shown in Output 44.1.4.) Due to the dropping of the less important components, the sum of these two numbers is 4.6700, which is only a little bit less than total variance 5 of the original correlation matrix.

You can also look at the variance explained by the two components for each observed variables in Output 44.1.7.

Output 44.1.7 Principal Component Analysis: Final Communality Estimates

Final Communality Estimates: Total = 4.669974				
Population	School	Employment	Services	HouseValue
0.98782629	0.88510555	0.97930583	0.88023562	0.93750041

In Output 44.1.7, the final communality estimates show that all the variables are well accounted for by the two

components, with final communality estimates ranging from 0.8802 for Services to 0.9878 for Population. The sum of the communalities is 4.6700, which is the same as the sum of the variance explained by the two components, as shown in Output 44.1.6.

Principal Component Analysis by PROC FACTOR and PROC PRINCOMP

The principal component analysis by PROC FACTOR emphasizes how the principal components explain the observed variables. The factor loadings in the factor pattern as shown in Output 44.1.5 are the coefficients for combining the factor/component scores to yield the observed variable scores when the expected error residuals are zero. For example, the predicted standardized value of Population given the factor/component scores for Factor1 and Factor2 is given by:

$$\text{Population} = 0.58096 \times \text{Factor1} + 0.80642 \times \text{Factor2}$$

If you are primarily interested in getting the component scores as linear combinations of the observed variables, the factor loading matrix table is not the right one for you. However, you might request the standardized scoring coefficients by adding the SCORE option in the FACTOR statement:

```
proc factor data=SocioEconomics n=5 score;
run;
```

In the preceding PROC FACTOR statement, N=5 is specified for retaining all five components. This is done for comparing the PROC FACTOR results with those of PROC PRINCOMP, which is described later. The SCORE option requests the display of the standardized scoring coefficients, which are shown in Output 44.1.8.

Output 44.1.8 Principal Component Analysis: Scoring Coefficients for Computing Component Scores

	Standardized Scoring Coefficients				
	Factor1	**Factor2**	**Factor3**	**Factor4**	**Factor5**
Population	0.20219	0.44884	0.12841	0.64542	5.58240
School	0.26695	-0.30320	1.48612	-1.11846	1.41574
Employment	0.23403	0.40411	0.53496	0.07256	-5.65135
Services	0.32450	-0.05806	-1.43273	-1.58288	-0.00100
HouseValue	0.27535	-0.31068	-0.30129	2.41419	-0.66734

In Output 44.1.8, each factor/component is expressed as a linear combination of the standardized observed variables. For example, the first principal component or Factor1 is computed as:

$$0.2022\times\text{Population}+0.2670\times\text{School}+0.2340\times\text{Employment}+0.3245\times\text{Services}+0.2753\times\text{HouseValue}$$

Again, when applying this formula you must use the standardized observed variables (with means 0 and standard deviations 1), but not the raw data.

Apart from some scaling differences, the set of scoring coefficients obtained from PROC FACTOR are equivalent to those obtained from PROC PRINCOMP, as specified by the following statement:

```
proc princomp data=SocioEconomics;
run;
```

PROC PRINCOMP displays the scoring coefficients as eigenvectors, which are shown in Output 44.1.9.

Output 44.1.9 Principal Component Analysis by PROC PRINCOMP: Eigenvectors

	Eigenvectors				
	Prin1	Prin2	Prin3	Prin4	Prin5
Population	0.342730	0.601629	0.059517	0.204033	0.689497
School	0.452507	-.406414	0.688822	-.353571	0.174861
Employment	0.396695	0.541665	0.247958	0.022937	-.698014
Services	0.550057	-.077817	-.664076	-.500386	-.000124
HouseValue	0.466738	-.416429	-.139649	0.763182	-.082425

For example, to get the first principal component score, you use the following formula:

$$0.3427\times\text{Population}+0.4525\times\text{School}+0.3967\times\text{Employment}+0.5500\times\text{Services}+0.4667\times\text{HouseValue}$$

This formula is not exactly the same as the one shown by using PROC FACTOR. All scoring coefficients in PROC FACTOR are smaller, approximately a factor of 0.59 to those coefficients obtained from PROC PRINCOMP. The reason for the scalar difference is that PROC FACTOR assumes all factors/components to have variance of 1, while PROC PRINCOMP creates components that have variances equal to the eigenvalues. You can do a simple rescaling of the standardized scoring coefficients obtained from PROC FACTOR so that they match the associated eigenvectors from the PROC PRINCOMP. Basically, you need to rescale each column of the standardized scoring coefficients obtained from PROC FACTOR to have the sum of squares equaling one, which is a defining characteristic of eigenvectors. This could be accomplished by dividing each coefficient by the square root of the corresponding column sum of squares.

For the present example, you can use PROC STDIZE to do the rescaling, as shown in the following statements:

```
proc factor data=SocioEconomics n=5 score;
   ods output StdScoreCoef=Coef;
run;

proc stdize method=ustd mult=.44721 data=Coef out=eigenvectors;
   var Factor1-Factor5;
run;

proc print data=eigenvectors;
run;
```

First, you create an output set Coef for the standardized scoring coefficients by the ODS OUTPUT statement. Note that "StdScoreCoef" is the ODS table that contains the standardized scoring coefficients as shown in Output 44.1.8. (See Table 44.6 for all ODS table names for PROC FACTOR.) Next, you use METHOD=USTD in the PROC STDIZE statement to divide the output coefficients by the corresponding uncorrected (for mean) standard deviations. The following formula shows the relationship between the uncorrected standard deviation and the sum of squares:

$$\text{uncorrected standard deviation} = \sqrt{\text{sum of squares}/N}$$

Recall that what you intend to divide from each coefficient is its square root of the corresponding column sum of squares. Therefore, to adjust for what PROC STDIZE does using METHOD=USTD, you have to multiply each variable by a constant term of $1/\sqrt{N}$ in the standardization. For the current example, this constant term is 0.44721 ($= 1/\sqrt{5}$) and is specified through the MULT= option in the PROC STDIZE statement. With the OUT= option, the rescaled scoring coefficients are saved in the SAS data set eigenvectors. The printout of the data set in Output 44.1.10 shows the rescaled standardized scoring coefficients obtained from PROC FACTOR.

Output 44.1.10 Rescaled Standardized Scoring Coefficients

Obs	Variable	Factor1	Factor2	Factor3	Factor4	Factor5
1	Population	0.34273	0.60162	0.05952	0.20403	0.68949
2	School	0.45250	-0.40641	0.68882	-0.35357	0.17486
3	Employment	0.39669	0.54166	0.24796	0.02294	-0.69801
4	Services	0.55005	-0.07782	-0.66407	-0.50038	-0.00012
5	HouseValue	0.46673	-0.41643	-0.13965	0.76318	-0.08242

As you can see, these standardized scoring coefficients are essentially the same as those obtained from PROC PRINCOMP, as shown in Output 44.1.9. This example shows that principal component analyses by PROC FACTOR and PROC PRINCOMP are indeed equivalent. PROC PRINCOMP emphasizes more the linear combinations of the variables to form the components, while PROC FACTOR expresses variables as linear combinations of the components in the output. If a principal component analysis of the data is all you need in a particular application, there is no reason to use PROC FACTOR instead of PROC PRINCOMP. Therefore, the following examples focus on common factor analysis for which that you can apply only PROC FACTOR, but not PROC PRINCOMP.

Example 44.2: Principal Factor Analysis

(View the complete code for this example (facex2.sas) in the example repository.)

This example uses the data presented in Example 44.1 and performs a principal factor analysis with squared multiple correlations for the prior communality estimates. Unlike Example 44.1, which analyzes the principal components (with default PRIORS=ONE), the current analysis is based on a common factor model. To use a common factor model, you specify PRIORS=SMC in the PROC FACTOR statement, as shown in the following:

```
ods graphics on;

proc factor data=SocioEconomics
   priors=smc msa residual
   rotate=promax reorder
   outstat=fact_all
   plots=(scree initloadings preloadings loadings);
run;

ods graphics off;
```

In the PROC FACTOR statement, you include several other options to help you analyze the results. To help determine whether the common factor model is appropriate, you request the Kaiser's measure of sampling adequacy with the MSA option. You specify the RESIDUALS option to compute the residual correlations and partial correlations.

The ROTATE= and REORDER options are specified to enhance factor interpretability. The ROTATE=PROMAX option produces an orthogonal varimax prerotation (default) followed by an oblique Procrustes rotation, and the REORDER option reorders the variables according to their largest factor loadings. An OUTSTAT= data set is created by PROC FACTOR and displayed in Output 44.2.15.

PROC FACTOR can produce high-quality graphs that are very useful for interpreting the factor solutions. To request these graphs, ODS Graphics must be enabled. All ODS graphs in PROC FACTOR are requested with the PLOTS= option. In this example, you request a scree plot (SCREE) and loading plots for the factor matrix during the following three stages: initial unrotated solution (INITLOADINGS), prerotated (varimax) solution (PRELOADINGS), and promax-rotated solution (LOADINGS). The scree plot helps you determine the number of factors, and the loading plots help you visualize the patterns of factor loadings during various stages of analyses.

Principal Factor Analysis: Kaiser's MSA and Factor Extraction Results

Output 44.2.1 displays the results of the partial correlations and Kaiser's measure of sampling adequacy.

Output 44.2.1 Principal Factor Analysis: Partial Correlations and Kaiser's MSA

Partial Correlations Controlling all other Variables					
	Population	School	Employment	Services	HouseValue
Population	1.00000	-0.54465	0.97083	0.09612	0.15871
School	-0.54465	1.00000	0.54373	0.04996	0.64717
Employment	0.97083	0.54373	1.00000	0.06689	-0.25572
Services	0.09612	0.04996	0.06689	1.00000	0.59415
HouseValue	0.15871	0.64717	-0.25572	0.59415	1.00000

Kaiser's Measure of Sampling Adequacy: Overall MSA = 0.57536759				
Population	School	Employment	Services	HouseValue
0.47207897	0.55158839	0.48851137	0.80664365	0.61281377

If the data are appropriate for the common factor model, the partial correlations (controlling all other variables) should be small compared to the original correlations. For example, the partial correlation between the variables School and HouseValue is 0.65, slightly less than the original correlation of 0.86 (see Output 44.1.3). The partial correlation between Population and School is –0.54, which is much larger in absolute value than the original correlation; this is an indication of trouble. Kaiser's MSA is a summary, for each variable and for all variables together, of how much smaller the partial correlations are than the original correlations. Values of 0.8 or 0.9 are considered good, while MSAs below 0.5 are unacceptable. The variables Population, School, and Employment have very poor MSAs. Only the Services variable has a good MSA. The overall MSA of 0.58 is sufficiently poor that additional variables should be included in the analysis to better define the common factors. A commonly used rule is that there should be at least three variables per factor. In the following analysis, you determine that there are two common factors in these data. Therefore, more variables are needed for a reliable analysis.

Output 44.2.2 displays the results of the principal factor extraction.

Output 44.2.2 Principal Factor Analysis: Factor Extraction

Prior Communality Estimates: SMC				
Population	School	Employment	Services	HouseValue
0.96859160	0.82228514	0.96918082	0.78572440	0.84701921

Output 44.2.2 *continued*

Eigenvalues of the Reduced Correlation Matrix: Total = 4.39280116 Average = 0.87856023				
	Eigenvalue	**Difference**	**Proportion**	**Cumulative**
1	2.73430084	1.01823217	0.6225	0.6225
2	1.71606867	1.67650586	0.3907	1.0131
3	0.03956281	0.06408626	0.0090	1.0221
4	-.02452345	0.04808427	-0.0056	1.0165
5	-.07260772		-0.0165	1.0000

The square multiple correlations are shown as prior communality estimates in Output 44.2.2. The PRIORS=SMC option basically replaces the diagonal of the original observed correlation matrix by these square multiple correlations. Because the square multiple correlations are usually less than one, the resulting correlation matrix for factoring is called the reduced correlation matrix. In the current example, the SMCs are all fairly large; hence, you expect the results of the principal factor analysis to be similar to those in the principal component analysis.

The first two largest positive eigenvalues of the reduced correlation matrix account for 101.31% of the common variance. This is possible because the reduced correlation matrix, in general, is not necessarily positive definite, and negative eigenvalues for the matrix are possible. A pattern like this suggests that you might not need more than two common factors. The scree and variance explained plots of Output 44.2.3 clearly support the conclusion that two common factors are present. Showing in the left panel of Output 44.2.3 is the scree plot of the eigenvalues of the reduced correlation matrix. A sharp bend occurs at the third eigenvalue, reinforcing the conclusion that two common factors are present. These cumulative proportions of common variance explained by factors are plotted in the right panel of Output 44.2.3, which shows that the curve essentially flattens out after the second factor.

Output 44.2.3 Scree and Variance Explained Plots

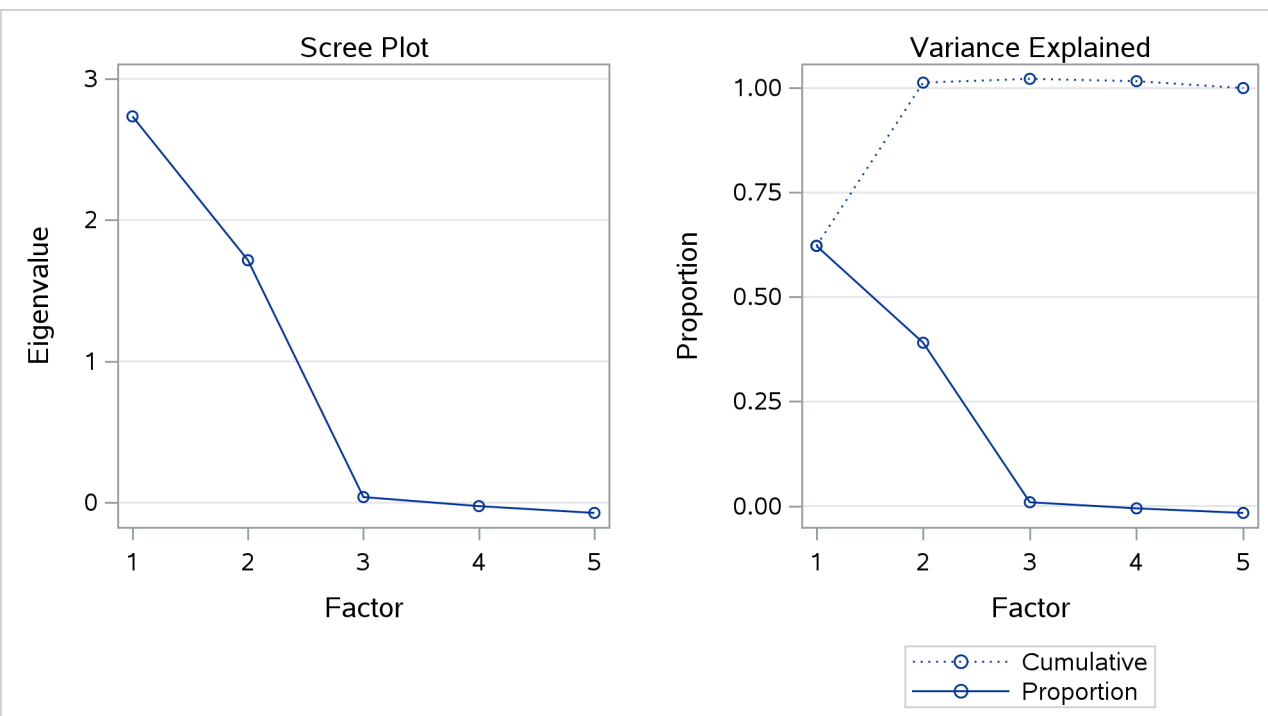

Principal Factor Analysis: Initial Factor Solution

For the current analysis, PROC FACTOR retains two factors by certain default criteria. This decision agrees with the conclusion drawn by inspecting the scree plot. The principal factor pattern with the two factors is displayed in Output 44.2.4. This factor pattern is similar to the principal component pattern seen in Output 44.1.5 of Example 44.1. For example, the variable Services has the largest loading on the first factor, and the Population variable has the smallest. The variables Population and Employment have large positive loadings on the second factor, and the HouseValue and School variables have large negative loadings.

Output 44.2.4 Initial Factor Pattern Matrix and Communalities

Factor Pattern		
	Factor1	Factor2
Services	0.87899	-0.15847
HouseValue	0.74215	-0.57806
Employment	0.71447	0.67936
School	0.71370	-0.55515
Population	0.62533	0.76621

Variance Explained by Each Factor	
Factor1	Factor2
2.7343008	1.7160687

Final Communality Estimates: Total = 4.450370				
Population	School	Employment	Services	HouseValue
0.97811334	0.81756387	0.97199928	0.79774304	0.88494998

Comparing the current factor loading matrix in Output 44.2.4 with that in Output 44.1.5 in Example 44.1, you notice that the variables are arranged differently in the two output tables. This is due to the use of the REORDER option in the current analysis. The advantage of using this option might not be very obvious in Output 44.2.4, but you can see its value when looking at the rotated solutions, as shown in Output 44.2.7 and Output 44.2.11.

The final communality estimates are all fairly close to the priors (shown in Output 44.2.2). Only the communality for the variable HouseValue increased appreciably, from 0.847 to 0.885. Therefore, you are sure that all the common variance is accounted for.

Output 44.2.5 shows that the residual correlations (off-diagonal elements) are low, the largest being 0.03. The partial correlations are not quite as impressive, since the uniqueness values are also rather small. These results indicate that the squared multiple correlations are good but not quite optimal communality estimates.

Output 44.2.5 Residual and Partial Correlations

Residual Correlations With Uniqueness on the Diagonal					
	Population	School	Employment	Services	HouseValue
Population	0.02189	-0.01118	0.00514	0.01063	0.00124
School	-0.01118	0.18244	0.02151	-0.02390	0.01248
Employment	0.00514	0.02151	0.02800	-0.00565	-0.01561
Services	0.01063	-0.02390	-0.00565	0.20226	0.03370
HouseValue	0.00124	0.01248	-0.01561	0.03370	0.11505

Output 44.2.5 *continued*

Root Mean Square Off-Diagonal Residuals: Overall = 0.01693282				
Population	**School**	**Employment**	**Services**	**HouseValue**
0.00815307	0.01813027	0.01382764	0.02151737	0.01960158

Partial Correlations Controlling Factors					
	Population	**School**	**Employment**	**Services**	**HouseValue**
Population	1.00000	-0.17693	0.20752	0.15975	0.02471
School	-0.17693	1.00000	0.30097	-0.12443	0.08614
Employment	0.20752	0.30097	1.00000	-0.07504	-0.27509
Services	0.15975	-0.12443	-0.07504	1.00000	0.22093
HouseValue	0.02471	0.08614	-0.27509	0.22093	1.00000

Root Mean Square Off-Diagonal Partials: Overall = 0.18550132				
Population	**School**	**Employment**	**Services**	**HouseValue**
0.15850824	0.19025867	0.23181838	0.15447043	0.18201538

As displayed in Output 44.2.6, the unrotated factor pattern reveals two tight clusters of variables, with the variables HouseValue and School at the negative end of Factor2 axis and the variables Employment and Population at the positive end. The Services variable is in between but closer to the HouseValue and School variables. A good rotation would place the axes so that most variables would have zero loadings on most factors. As a result, the axes would appear as though they are put through the variable clusters.

Output 44.2.6 Unrotated Factor Loading Plot

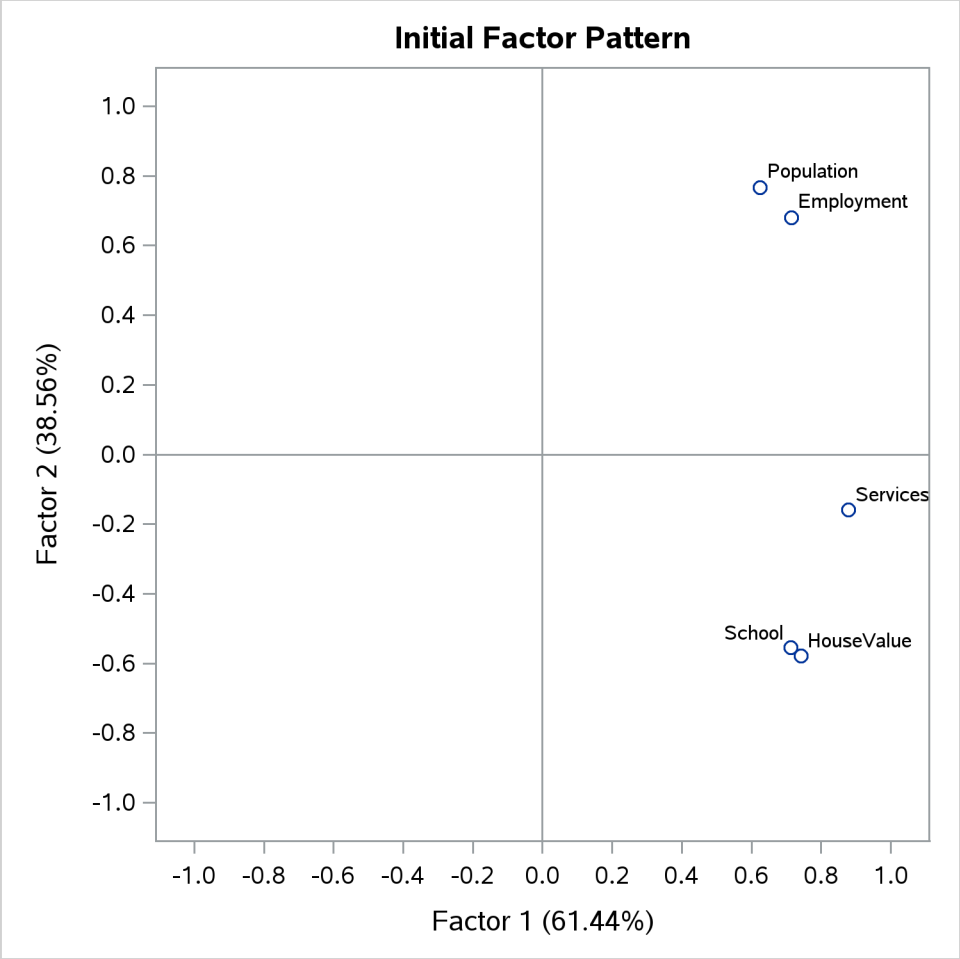

Principal Factor Analysis: Varimax Prerotation

In Output 44.2.7, the results of the varimax prerotation are shown. To yield the varimax-rotated factor loading (pattern), the initial factor loading matrix is postmultiplied by an orthogonal transformation matrix. This orthogonal transformation matrix is shown in Output 44.2.7, followed by the varimax-rotated factor pattern. This rotation or transformation leads to small loadings of Population and Employment on the first factor and small loadings of HouseValue and School on the second factor. Services appears to have a larger loading on the first factor than it has on the second factor, although both loadings are substantial. Hence, Services appears to be factorially complex.

With the REORDER option in effect, you can see the variable clusters clearly in the factor pattern. The first factor is associated more with the first three variables (first three rows of variables): HouseValue, School, and Services. The second factor is associated more with the last two variables (last two rows of variables): Population and Employment.

For orthogonal factor solutions such as the current varimax-rotated solution, you can also interpret the values in the factor loading (pattern) matrix as correlations. For example, HouseValue and Factor 1 have a high correlation at 0.94, while Population and Factor 1 have a low correlation at 0.02.

Output 44.2.7 Varimax Rotation: Transform Matrix and Rotated Pattern

Orthogonal Transformation Matrix		
	1	**2**
1	0.78895	0.61446
2	-0.61446	0.78895

Rotated Factor Pattern		
	Factor1	**Factor2**
HouseValue	0.94072	-0.00004
School	0.90419	0.00055
Services	0.79085	0.41509
Population	0.02255	0.98874
Employment	0.14625	0.97499

Variance Explained by Each Factor	
Factor1	**Factor2**
2.3498567	2.1005128

Final Communality Estimates: Total = 4.450370				
Population	**School**	**Employment**	**Services**	**HouseValue**
0.97811334	0.81756387	0.97199928	0.79774304	0.88494998

The variance explained by the factors are more evenly distributed in the varimax-rotated solution, as compared with that of the unrotated solution. Indeed, this is a typical fact for any kinds of factor rotation. In the current example, before the varimax rotation the two factors explain 2.73 and 1.72, respectively, of the common variance (see Output 44.2.4). After the varimax rotation the two rotated factors explain 2.35 and 2.10, respectively, of the common variance. However, the total variance accounted for by the factors remains unchanged after the varimax rotation. This invariance property is also observed for the communalities of the variables after the rotation, as evidenced by comparing the current communality estimates in Output 44.2.7 with those in Output 44.2.4.

Output 44.2.8 shows the graphical plot of the varimax-rotated factor loadings. Clearly, HouseValue and School cluster together on the Factor 1 axis, while Population and Employment cluster together on the Factor 2 axis. Service is closer to the cluster of HouseValue and School.

Output 44.2.8 Varimax-Rotated Factor Loadings

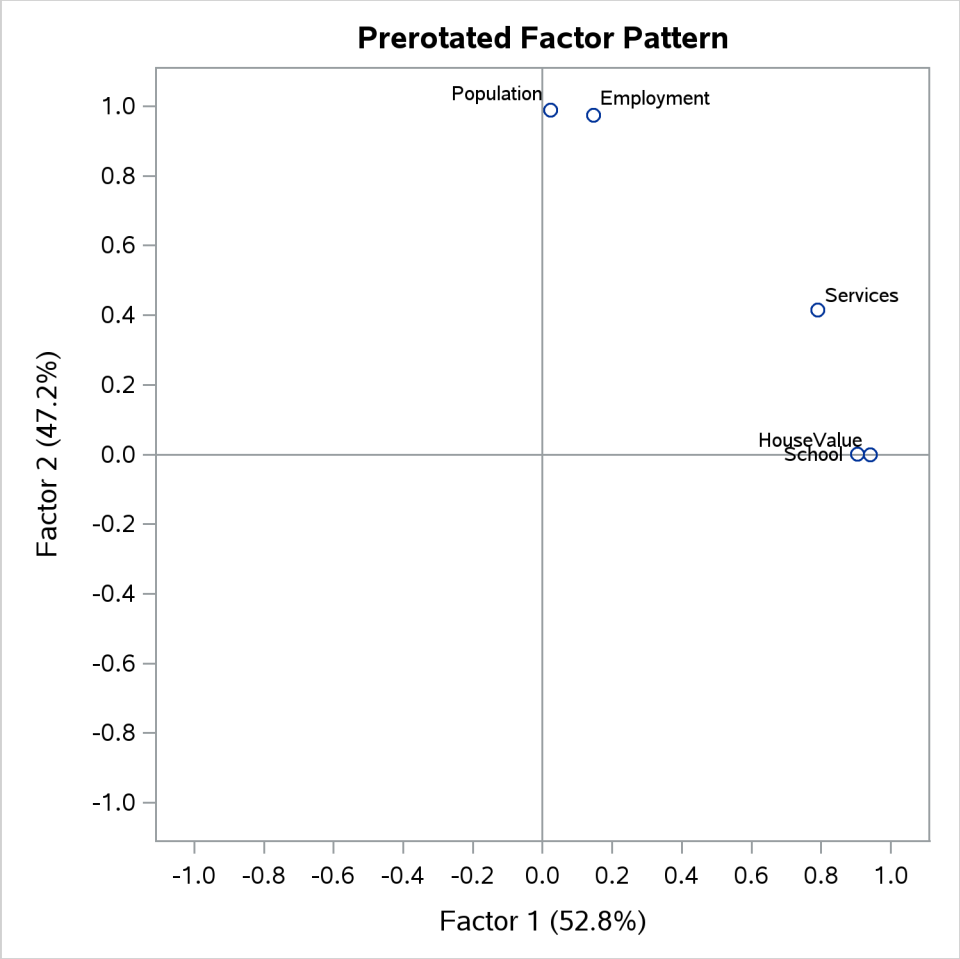

An alternative to the scatter plot of factor loadings is the so-called vector plot of loadings, which is shown in Output 44.2.9. The vector plot is requested with the suboption VECTOR in the PLOTS= option. That is:

```
plots=preloadings(vector)
```

This generates the vector plot of loadings in Output 44.2.9.

Output 44.2.9 Varimax-Rotated Factor Loadings: Vector Plot

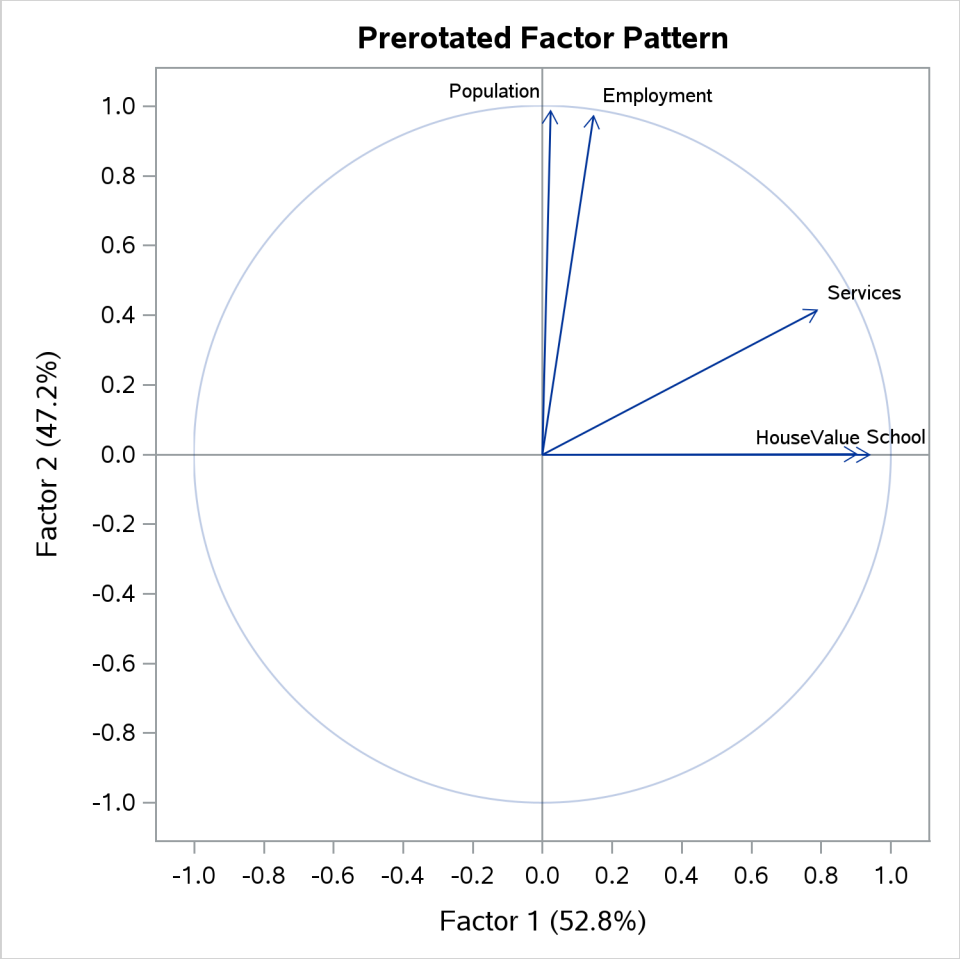

Principal Factor Analysis: Oblique Promax Rotation

For some researchers, the varimax-rotated factor solution in the preceding section might be good enough to provide them useful and interpretable results. For others who believe that common factors are seldom orthogonal, an obliquely rotated factor solution might be more desirable, or at least should be attempted.

PROC FACTOR provides a very large class of oblique factor rotations. The current example shows a particular one—namely, the promax rotation as requested by the ROTATE=PROMAX option.

The results of the promax rotation are shown in Output 44.2.10 and Output 44.2.11. The corresponding plot of factor loadings is shown in Output 44.2.12.

Output 44.2.10 Promax Rotation: Procrustean Target and Transformation

Target Matrix for Procrustean Transformation		
	Factor1	Factor2
HouseValue	1.00000	-0.00000
School	1.00000	0.00000
Services	0.69421	0.10045
Population	0.00001	1.00000
Employment	0.00326	0.96793

Procrustean Transformation Matrix		
	1	2
1	1.04117	-0.09865
2	-0.10572	0.96303

Normalized Oblique Transformation Matrix		
	1	2
1	0.73803	0.54202
2	-0.70555	0.86528

Output 44.2.10 shows the Procrustean target, to which the varimax factor pattern is rotated, followed by the display of the Procrustean transformation matrix. This is the matrix that transforms the varimax factor pattern so that the rotated pattern is as close as possible to the Procrustean target. However, because the variances of factors have to be fixed at 1 during the oblique transformation, a normalized version of the Procrustean transformation matrix is the one that is actually used in the transformation. This normalized transformation matrix is shown at the bottom of Output 44.2.10. Using this transformation matrix leads to the promax-rotated factor solution, as shown in Output 44.2.11.

Output 44.2.11 Promax Rotation: Factor Correlations and Factor Pattern

Inter-Factor Correlations		
	Factor1	Factor2
Factor1	1.00000	0.20188
Factor2	0.20188	1.00000

Rotated Factor Pattern (Standardized Regression Coefficients)		
	Factor1	Factor2
HouseValue	0.95558	-0.09792
School	0.91842	-0.09352
Services	0.76053	0.33932
Population	-0.07908	1.00192
Employment	0.04799	0.97509

After the promax rotation, the factors are no longer uncorrelated. As shown in Output 44.2.11, the correlation of the two factors is now 0.20. In the (initial) unrotated and the varimax solutions, the two factors are not correlated.

In addition to allowing the factors to be correlated, in an oblique factor solution you seek a pattern of factor

loadings that is more "differentiated" (referred to as the "simple structures" in the literature). The more differentiated the loadings, the easier the interpretation of the factors.

For example, factor loadings of Services and Population on Factor 2 are 0.415 and 0.989, respectively, in the (orthogonal) varimax-rotated factor pattern (see Output 44.2.7). With the (oblique) promax rotation (see Output 44.2.11), these two loadings become even more differentiated with values 0.339 and 1.002, respectively. Overall, however, the factor patterns before and after the promax rotation do not seem to differ too much. This fact is confirmed by comparing the graphical plots of factor loadings. The plots in Output 44.2.12 (promax-rotated factor loadings) and Output 44.2.8 (varimax-rotated factor loadings) show very similar patterns.

Output 44.2.12 Promax Rotation: Factor Loading Plot

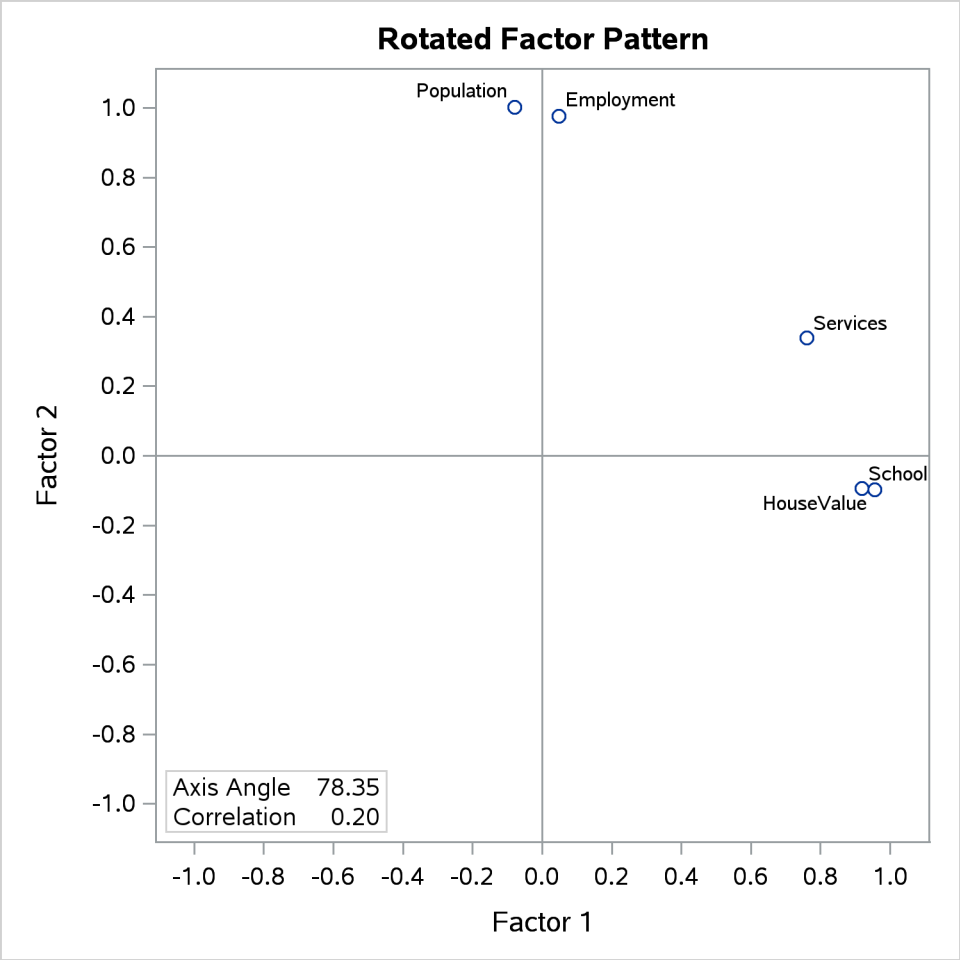

Unlike the orthogonal factor solutions where you can interpret the factor loadings as correlations between variables and factors, in oblique factor solutions such as the promax solution, you have to turn to the factor structure matrix for examining the correlations between variables and factors. Output 44.2.13 shows the factor structures of the promax-rotated solution.

Output 44.2.13 Promax Rotation: Factor Structures and Final Communalities

Factor Structure (Correlations)		
	Factor1	Factor2
HouseValue	0.93582	0.09500
School	0.89954	0.09189
Services	0.82903	0.49286
Population	0.12319	0.98596
Employment	0.24484	0.98478

Variance Explained by Each Factor Ignoring Other Factors	
Factor1	Factor2
2.4473495	2.2022803

Final Communality Estimates: Total = 4.450370				
Population	School	Employment	Services	HouseValue
0.97811334	0.81756387	0.97199928	0.79774304	0.88494998

Basically, the factor structure matrix shown in Output 44.2.13 reflects a similar pattern to the factor pattern matrix shown in Output 44.2.11. The critical difference is that you can have the correlation interpretation only by using the factor structure matrix. For example, in the factor structure matrix shown in Output 44.2.13, the correlation between Population and Factor 2 is 0.986. The corresponding value shown in the factor pattern matrix in Output 44.2.11 is 1.002, which certainly cannot be interpreted as a correlation coefficient.

Common variance explained by the promax-rotated factors are 2.447 and 2.202, respectively, for the two factors. Unlike the orthogonal factor solutions (for example, the prerotated varimax solution), variance explained by these promax-rotated factors do not sum up to the total communality estimate 4.45. In oblique factor solutions, variance explained by oblique factors cannot be partitioned for the factors. Variance explained by a common factor is computed while ignoring the contributions from the other factors.

However, the communalities for the variables, as shown in the bottom of Output 44.2.13, do not change from rotation to rotation. They are still the same set of communalities in the initial, varimax-rotated, and promax-rotated solutions. This is a basic fact about factor rotations: they only redistribute the variance explained by the factors; the total variance explained by the factors for any variable (that is, the communality of the variable) remains unchanged.

In the literature of exploratory factor analysis, reference axes had been an important tool in factor rotation. Nowadays, rotations are seldom done through the uses of the reference axes. Despite that, results about reference axes do provide additional information for interpreting factor analysis results. For the current example of the promax rotation, PROC FACTOR shows the relevant results about the reference axes in Output 44.2.14.

Output 44.2.14 Promax Rotation: Reference Axis Correlations and Reference Structures

Reference Axis Correlations		
	Factor1	Factor2
Factor1	1.00000	-0.20188
Factor2	-0.20188	1.00000

Output 44.2.14 *continued*

Reference Structure (Semipartial Correlations)		
	Factor1	**Factor2**
HouseValue	0.93591	-0.09590
School	0.89951	-0.09160
Services	0.74487	0.33233
Population	-0.07745	0.98129
Employment	0.04700	0.95501

Variance Explained by Each Factor Eliminating Other Factors	
Factor1	**Factor2**
2.2480892	2.0030200

To explain the results in the reference-axis system, some geometric interpretations of the factor axes are needed. Consider a single factor in a system of n common factors in an oblique factor solution. Taking away the factor under consideration, the remaining $n - 1$ factors span a hyperplane in the factor space of $n - 1$ dimensions. The vector that is orthogonal to this hyperplane is the reference axis (reference vector) of the factor under consideration. Using the same definition for the remaining factors, you have n reference vectors for n factors.

A factor in an oblique factor solution can be considered as the sum of two independent components: its associated reference vector and a component that is overlapped with all other factors. In other words, the reference vector of a factor is a unique part of the factor that is not predictable from all other factors. Thus, the loadings on a reference vector are the unique effects of the corresponding factor, partialing out the effects from all other factors. The variances explained by a reference vector are the unique variances explained by the corresponding factor, partialing out the variances explained by all other factors.

Output 44.2.14 shows the reference axis correlations. The correlation between the reference vectors is –0.20. Next, Output 44.2.14 shows the loadings on the reference vectors in the table entitled "Reference Structure (Semipartial Correlations)." As explained previously, loadings on a reference vector are also the unique effects of the corresponding factor, partialing out the effects from the all other factors. For example, the unique effect of Factor 1 on HouseValue is 0.936. Another important property of the reference vector system is that loadings on a reference vector are also correlations between the variables and the corresponding factor, partialing out the correlations between the variables and other factors. This means that the loading 0.936 in the reference structure table is the unique correlation between HouseValue and Factor 1, partialing out the correlation between HouseValue with Factor 2. Hence, as suggested by the title of table, all loadings reported in the "Reference Structure (Semipartial Correlations)" can be interpreted as semipartial correlations between variables and factors.

The last table shown in Output 44.2.14 are the variances explained by the reference vectors. As explained previously, these are also unique variances explained by the factors, partialing out the variances explained by all other factors (or eliminating all other factors, as suggested by the title of the table). In the current example, Factor 1 explains 2.248 of the variable variances, partialing out all variable variances explained by Factor 2.

Notice that factor pattern (shown in Output 44.2.11), factor structures (correlations, shown in Output 44.2.13), and reference structures (semipartial correlations, shown in Output 44.2.14) give you different information about the oblique factor solutions such as the promax-rotated solution. However, for orthogonal factor

solutions such as the varimax-rotated solution, factor structures and reference structures are all the same as the factor pattern.

Principal Factor Analysis: Factor Rotations with Factor Pattern Input

The promax rotation is one of the many rotations that PROC FACTOR provides. You can specify many different rotation algorithms by using the ROTATE= options. In this section, you explore different rotated factor solutions from the initial principal factor solution. Specifically, you want to examine the factor patterns yielded by the quartimax transformation (an orthogonal transformation) and the Harris-Kaiser (an oblique transformation), respectively.

Rather than analyzing the entire problem again with new rotations, you can simply use the OUTSTAT= data set from the preceding factor analysis results.

First, the OUTSTAT= data set is printed using the following statements:

```
proc print data=fact_all;
run;
```

The output data set is displayed in Output 44.2.15.

Output 44.2.15 Output Data Set

Factor Output Data Set

Obs	_TYPE_	_NAME_	Population	School	Employment	Services	HouseValue
1	MEAN		6241.67	11.4417	2333.33	120.833	17000.00
2	STD		3439.99	1.7865	1241.21	114.928	6367.53
3	N		12.00	12.0000	12.00	12.000	12.00
4	CORR	Population	1.00	0.0098	0.97	0.439	0.02
5	CORR	School	0.01	1.0000	0.15	0.691	0.86
6	CORR	Employment	0.97	0.1543	1.00	0.515	0.12
7	CORR	Services	0.44	0.6914	0.51	1.000	0.78
8	CORR	HouseValue	0.02	0.8631	0.12	0.778	1.00
9	COMMUNAL		0.98	0.8176	0.97	0.798	0.88
10	PRIORS		0.97	0.8223	0.97	0.786	0.85
11	EIGENVAL		2.73	1.7161	0.04	-0.025	-0.07
12	UNROTATE	Factor1	0.63	0.7137	0.71	0.879	0.74
13	UNROTATE	Factor2	0.77	-0.5552	0.68	-0.158	-0.58
14	RESIDUAL	Population	0.02	-0.0112	0.01	0.011	0.00
15	RESIDUAL	School	-0.01	0.1824	0.02	-0.024	0.01
16	RESIDUAL	Employment	0.01	0.0215	0.03	-0.006	-0.02
17	RESIDUAL	Services	0.01	-0.0239	-0.01	0.202	0.03
18	RESIDUAL	HouseValue	0.00	0.0125	-0.02	0.034	0.12
19	PRETRANS	Factor1	0.79	-0.6145	.	.	.
20	PRETRANS	Factor2	0.61	0.7889	.	.	.
21	PREROTAT	Factor1	0.02	0.9042	0.15	0.791	0.94
22	PREROTAT	Factor2	0.99	0.0006	0.97	0.415	-0.00
23	TRANSFOR	Factor1	0.74	-0.7055	.	.	.
24	TRANSFOR	Factor2	0.54	0.8653	.	.	.
25	FCORR	Factor1	1.00	0.2019	.	.	.
26	FCORR	Factor2	0.20	1.0000	.	.	.
27	PATTERN	Factor1	-0.08	0.9184	0.05	0.761	0.96
28	PATTERN	Factor2	1.00	-0.0935	0.98	0.339	-0.10
29	RCORR	Factor1	1.00	-0.2019	.	.	.
30	RCORR	Factor2	-0.20	1.0000	.	.	.
31	REFERENC	Factor1	-0.08	0.8995	0.05	0.745	0.94
32	REFERENC	Factor2	0.98	-0.0916	0.96	0.332	-0.10
33	STRUCTUR	Factor1	0.12	0.8995	0.24	0.829	0.94
34	STRUCTUR	Factor2	0.99	0.0919	0.98	0.493	0.09

Various results from the previous factor analysis are saved in this data set, including the initial unrotated solution (its factor pattern is saved in observations with _TYPE_=UNROTATE), the prerotated varimax solution (its factor pattern is saved in observations with _TYPE_=PREROTAT), and the oblique promax solution (its factor pattern is saved in observations with _TYPE_=PATTERN).

When PROC FACTOR reads in an input data set with TYPE=FACTOR, the observations with _TYPE_=PATTERN are treated as the initial factor pattern to be rotated by PROC FACTOR. Hence, it is important that you provide the correct initial factor pattern for PROC FACTOR to read in.

In the current example, you need to provide the unrotated solution from the preceding analysis as the input factor pattern. The following statements create a TYPE=FACTOR data set fact2 from the preceding

OUTSTAT= data set fact_all:

```
data fact2(type=factor);
   set fact_all;
   if _TYPE_ in('PATTERN' 'FCORR') then delete;
   if _TYPE_='UNROTATE' then _TYPE_='PATTERN';
run;
```

In these statements, you delete observations with _TYPE_=PATTERN or _TYPE_=FCORR, which are for the promax-rotated factor solution, and change observations with _TYPE_=UNROTATE to _TYPE_=PATTERN in the new data set fact2. In this way, the initial orthogonal factor pattern matrix is saved in the observations with _TYPE_=PATTERN.

You use this new data set and rotate the initial solution to another oblique solution with the ROTATE=QUARTIMAX option, as shown in the following statements:

```
proc factor data=fact2 rotate=quartimax reorder;
run;
```

As shown in Output 44.2.16, the new rotation uses a TYPE=FACTOR input data set.

Output 44.2.16 Quartimax Rotation With Input Factor Pattern

Quartimax Rotation From a TYPE=FACTOR Data Set

The FACTOR Procedure

Input Data Type	FACTOR
N Set/Assumed in Data Set	12
N for Significance Tests	12

The quartimax-rotated factor pattern is displayed in Output 44.2.17.

Output 44.2.17 Quartimax-Rotated Factor Pattern

Orthogonal Transformation Matrix

	1	2
1	0.80138	0.59815
2	-0.59815	0.80138

Rotated Factor Pattern

	Factor1	Factor2
HouseValue	0.94052	-0.01933
School	0.90401	-0.01799
Services	0.79920	0.39878
Population	0.04282	0.98807
Employment	0.16621	0.97179

Variance Explained by Each Factor

Factor1	Factor2
2.3699941	2.0803754

The quartimax rotation produces an orthogonal transformation matrix shown at the top of Output 44.2.17. After the transformation, the factor pattern is shown next. Compared with the varimax-rotated factor pattern (see Output 44.2.7), the quartimax-rotated factor pattern shows some differences. The loadings of HouseValue and School on Factor 1 drop only slightly in the quartimax factor pattern, while the loadings of Services, Population, and Employment on Factor 1 gain relatively larger amounts. The total variance explained by Factor 1 in the varimax-rotated solution (see Output 44.2.7) is 2.350, while it is 2.370 after the quartimax-rotation. In other words, more variable variances are explained by the first factor in the quartimax factor pattern than in the varimax factor pattern. Although not very strongly demonstrated in the current example, this illustrates a well-known property about the quartimax rotation: it tends to produce a general factor for all variables.

Another oblique rotation is now explored. The Harris-Kaiser transformation weighted by the Cureton-Mulaik technique is applied to the initial factor pattern. To achieve this, you use the ROTATE=HK and NORM=WEIGHT options in the following PROC FACTOR statement:

```
ods graphics on;

proc factor data=fact2 rotate=hk norm=weight reorder plots=loadings;
run;

ods graphics off;
```

Output 44.2.18 shows the variable weights in the rotation.

Output 44.2.18 Harris-Kaiser Rotation: Weights

Variable Weights for Rotation				
Population	**School**	**Employment**	**Services**	**HouseValue**
0.95982747	0.93945424	0.99746396	0.12194766	0.94007263

While all other variables have weights at least as large as 0.93, the weight for Services is only 0.12. This means that due to its small weight, Services is not as important as the other variables for determining the rotation (transformation). This makes sense when you look at the initial unrotated factor pattern plot in Output 44.2.6. In the plot, there are two main clusters of variables, and Services does not seem to fall into either of the clusters. In order to yield a Harris-Kaiser rotation (transformation) that would gear towards to two clusters, the Cureton-Mulaik weighting essentially downweights the contribution from Services in the factor rotation.

The results of the Harris-Kaiser factor solution are displayed in Output 44.2.19, with a graphical plot of rotated loadings displayed in Output 44.2.20.

Output 44.2.19 Harris-Kaiser Rotation: Factor Correlations and Factor Pattern

Inter-Factor Correlations		
	Factor1	**Factor2**
Factor1	1.00000	0.08358
Factor2	0.08358	1.00000

Output 44.2.19 *continued*

Rotated Factor Pattern (Standardized Regression Coefficients)		
	Factor1	**Factor2**
HouseValue	0.94048	0.00279
School	0.90391	0.00327
Services	0.75459	0.41892
Population	-0.06335	0.99227
Employment	0.06152	0.97885

Because the Harris-Kaiser produces an oblique factor solution, you compare the current results with that of the promax (see Output 44.2.11), which also produces an oblique factor solution. The correlation between the factors in the Harris-Kaiser solution is 0.084; this value is much smaller than the same correlation in the promax solution, which is 0.201. However, the Harris-Kaiser rotated factor pattern shown in Output 44.2.19 is more or less the same as that of the promax-rotated factor pattern shown in Output 44.2.11. Which solution would you consider to be more reasonable or interpretable?

From the statistical point of view, the Harris-Kaiser and promax factor solutions are equivalent. They explain the observed variable relationships equally well. From the simplicity point of view, however, you might prefer to interpret the Harris-Kaiser solution because the factor correlation is smaller. In other words, the factors in the Harris-Kaiser solution do not overlap that much conceptually; hence they should be more distinctive to interpret. However, in practice simplicity in factor correlations might not the only principle to consider. Researchers might actually expect to have some factors to be highly correlated based on theoretical or substantive grounds.

Although the Harris-Kaiser and the promax factor patterns are very similar, the graphical plots of the loadings from the two solutions paint slightly different pictures. The plot of the promax-rotated loadings is shown in Output 44.2.12, while the plot of the loadings for the current Harris-Kaiser solution is shown in Output 44.2.20.

Output 44.2.20 Harris-Kaiser Rotation: Factor Loading Plot

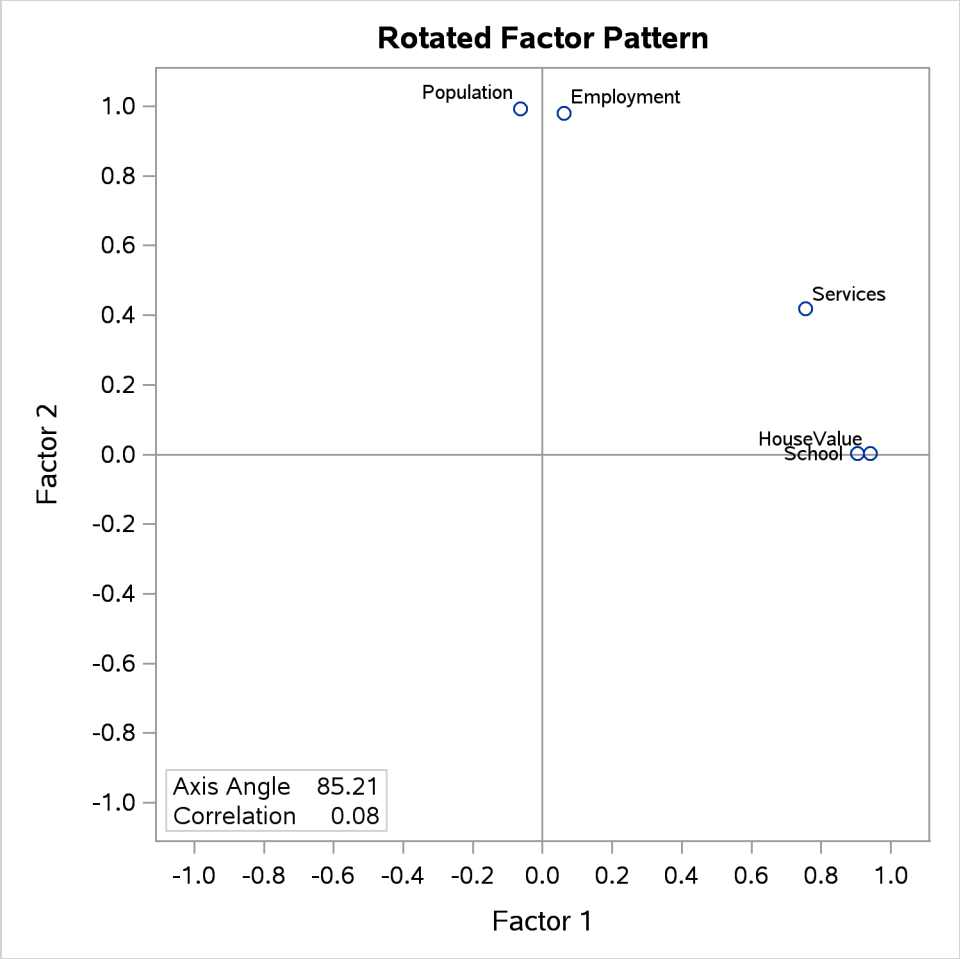

The two factor axes in the Harris-Kaiser rotated pattern (Output 44.2.20) clearly cut through the centers of the two variable clusters, while the Factor 1 axis in the promax solution lies above a variable cluster (Output 44.2.12). The reason for this subtle difference is that in the Harris-Kaiser rotation, the Services is a "loner" that has been downweighted by the Cureton-Mulaik technique (see its relatively small weight in Output 44.2.18). As a result, the rotated axes are basically determined by the two variable clusters in the Harris-Kaiser rotation.

As far as the current discussion goes, it is not recommending one rotation method over another. Rather, it simply illustrates how you could control certain types of characteristics of factor rotation through the many options supported by PROC FACTOR. Should you prefer an orthogonal rotation to an oblique rotation? Should you choose the oblique factor solution with the smallest factor correlations? Should you use a weighting scheme that would enable you to find independent variable clusters? While PROC FACTOR enables you to explore all these alternatives, you must consult advanced textbooks and published articles to get satisfactory and complete answers to these questions.

Example 44.3: Determining the Number of Factors to Retain

(View the complete code for this example (facex3.sas) in the example repository.)

This example uses the data presented in Example 44.1 to demonstrate the following advanced methods that help you determine the number of factors to retain:

- minimum average partial correlation (MAP) analysis
- parallel analysis

To use these methods, you can specify the MAP option or the PARALLEL option in the PROC FACTOR statement. These options display the results of the requested analysis but do not use the results to determine the number of factors to retain. To use either analysis to determine the number of factors to retain, specify the NFACTORS=*name* option with the desired argument (MAP, MAP2, MAP4, or PARALLEL). If you do not specify these advanced methods in the NFACTORS= option, the procedure determines the number of factors by using some basic criteria, which are based on the specified or default values of the MINEIGEN=, NFACTORS=*n*, and PROPORTION= options. For more information, see the section "Number of Factors to Retain" on page 2904.

In Example 44.1, the following specification causes PROC FACTOR to retain two factors by the basic criteria:

```
proc factor data=Socioeconomics
   simple corr;
run;
```

The output shows that:

2 factors will be retained by the MINEIGEN criterion.

This example repeats the analysis by specifying the MAP and parallel analysis options:

```
ods graphics on;

proc factor data=SocioEconomics
   parallel(alpha=0.01 nsims=10000 seed=20170229) map
   plots=(parallel map);
run;

ods graphics off;
```

It would be interesting to see whether these two advanced methods would suggest the same number of factors as that suggested by the default use of the MINEIGEN criterion in Example 44.1.

Parallel Analysis

Parallel analysis compares each of eigenvalues of the input data correlation matrix to an empirical distribution of eigenvalues. Each eigenvalue from the input correlation matrix that exceeds a critical value (based on a one-sided α-level) in the corresponding empirical distribution suggests a factor to be retained. For more information, see the PARALLEL option and the section "Number of Factors to Retain" on page 2904.

The parallel analysis results are displayed in Output 44.3.1. The output table summarizes the eigenvalues of the input correlation matrix and compares those eigenvalues to the critical values of the parallel analysis. Because the first two eigenvalues of the input correlation matrix exceed their corresponding critical values, the parallel analysis suggests that two factors be retained, in agreement with the number retained by the MINEIGEN= criterion in Example 44.1.

Output 44.3.1 Table of Results for the Parallel Analysis

Parallel Analysis: NSims=10000 Seed=20170229		
	Simulated Crit Val	Observed Eigenvalue
1	2.6144	**2.8733***
2	1.7246	**1.7967***
3	1.2150	0.2148
4	0.8836	0.0999
5	0.5703	0.0153
* Retained Dimension (Obs > Crit, alpha=0.01)		

You can use parallel analysis to directly determine the number of factors to retain by specifying NFACTORS=PARALLEL as follows:

```
proc factor data=SocioEconomics
   nfactors=parallel(alpha=0.01 nsims=10000 seed=20170229);
run;
```

When you request a parallel analysis (using either the PARALLEL option or the NFACTORS=PARALLEL option in the PROC FACTOR statement), you can obtain a graphical summary of the results by also specifying the PLOTS=PARALLEL option. As shown in Output 44.3.2, the scree plot for a parallel analysis contains two lines. The first line is a traditional scree plot and shows the eigenvalues of the sample correlation matrix, sorted in descending order. The second line is constructed from the critical values that are obtained from the parallel analysis. The suggested number of factors to retain is indicated by the intersection of these two lines.

Output 44.3.2 Graphical Summary of the Parallel Analysis

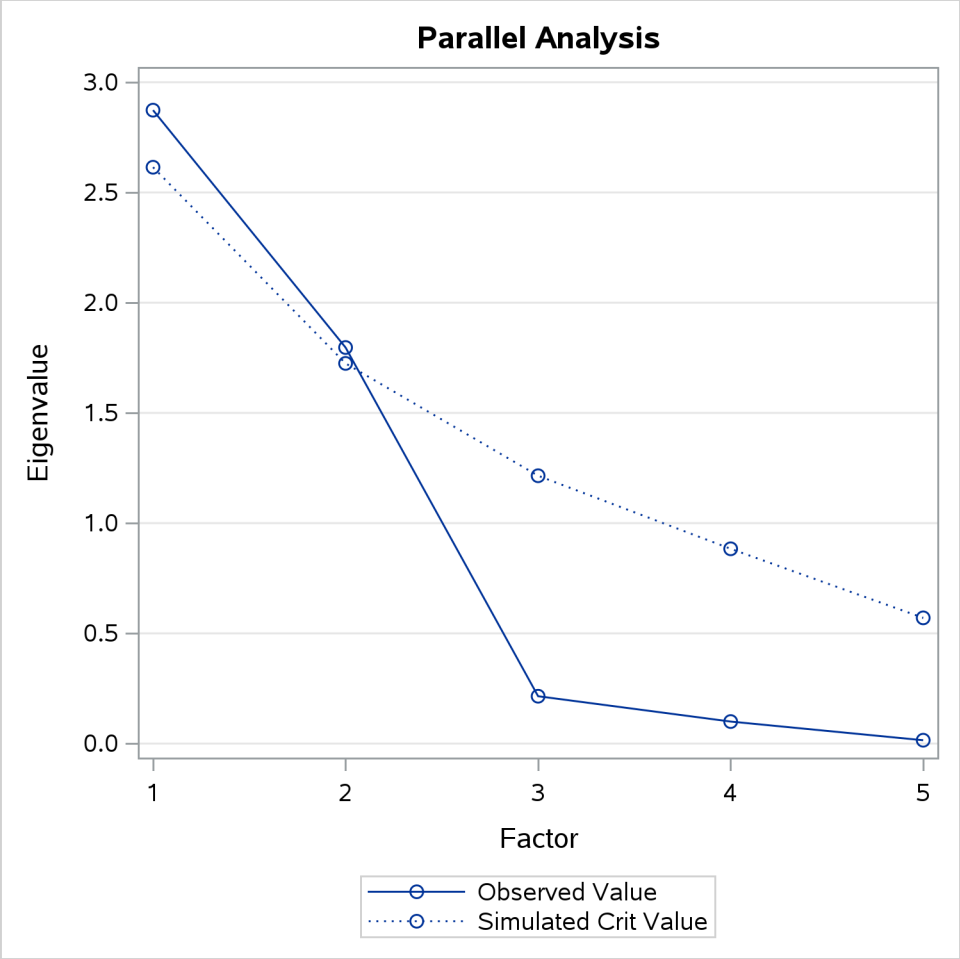

You can manipulate some parallel analysis options to assess the numerical stability of the parallel analysis results. That is, you can specify different values in the ALPHA=*p*, NSIMS=*n*, and SEED=*n* suboptions to determine whether the suggested number of factors would change. For a complete definition of these suboptions, see the PARALLEL option description. You can use the ALPHA=*p* suboption to adjust how conservative the parallel analysis will be. As *p* decreases, the critical values in the parallel analysis increase, resulting in a more conservative number of factors suggested by the parallel analysis. You can use the NSIMS=*n* suboption to balance the trade-off between the accuracy of the simulated critical values and the computational time required for the parallel analysis. For data sets that contain few observations relative to the number of variables, you need to use a larger NSIMS= value to ensure the numerical accuracy of the simulated critical values. Finally, you can specify the SEED= value so that the simulation result of the parallel analysis is reproducible.

Minimum Average Partial Correlation Analysis (MAP)

The MAP computes averages of squared and fourth-powered partial correlations of variables after recursively controlling for the first, second, and so on, principal components. The number of partialed-out components that corresponds to the minimum average of the squared or fourth-powered partial correlations suggests the number of factors to be retained. For more information, see the MAP option and the section "Number of Factors to Retain" on page 2904.

The minimum average partial correlation analysis results are displayed in Output 44.3.3. The output table contains the average squared and fourth-powered partial correlations after controlling for the indicated number of principal components. Because the minimum value (for both squared and fourth-powered partial correlations) occurs when two components are partialed out, the MAP analysis suggests that two factors be retained. This is in agreement with the number retained by the MINEIGEN= criterion in Example 44.1.

Output 44.3.3 Table of Results for the MAP

Average Partial Correlations Controlling Principal Components		
N Prin Comp Partialed	**Squared**	**Fourth-Powered**
0	0.3270	0.2151
1	0.4568	0.3188
2	0.2400*	0.0962*
3	0.4474	0.3470
4	1.0000	1.0000
*** MAP = Minimum Values in Columns**		

You can use a MAP analysis to directly determine the number of factors to retain by specifying NFACTORS=MAP2 or NFACTORS=MAP4. For example, the following statements request that the number of factors be determined by identifying the number of controlled principal components that results in the minimum average of the squared partial correlations:

```
proc factor data=SocioEconomics
   nfactors=map2;
run;
```

When you request a minimum average partial correlation analysis (using either the MAP option or the NFACTORS=MAP2 or NFACTORS=MAP4 option), you can obtain a graphical summary of the results with the PLOTS=MAP option. As shown in Output 44.3.4, the MAP plot provides a visualization of the MAP results that are shown in Output 44.3.3.

Output 44.3.4 Graphical Summary of the MAP

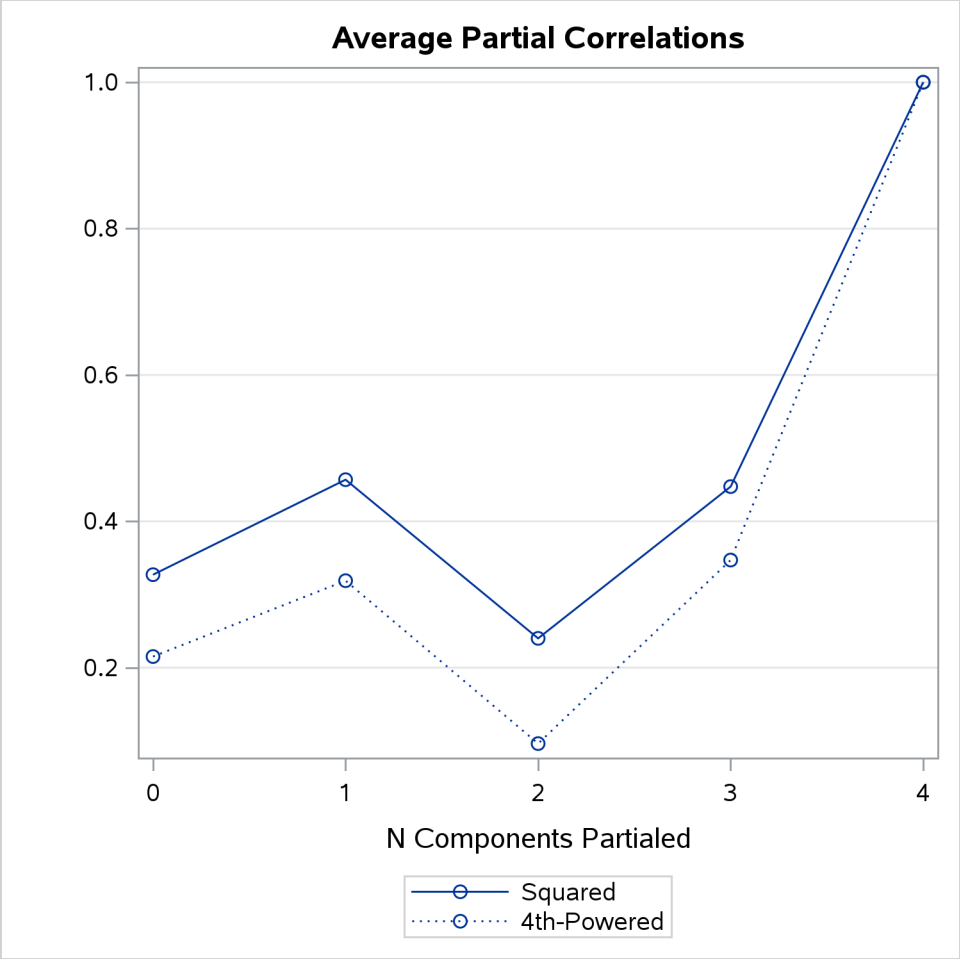

Example 44.4: Maximum Likelihood Factor Analysis

(View the complete code for this example (facex4.sas) in the example repository.)

This example uses maximum likelihood factor analyses for one, two, and three factors. It is already apparent from the principal factor analysis that the best number of common factors is almost certainly two. The one- and three-factor ML solutions reinforce this conclusion and illustrate some of the numerical problems that can occur. The following statements produce Output 44.4.1 through Output 44.4.3:

```
title3 'Maximum Likelihood Factor Analysis with One Factor';
proc factor data=SocioEconomics method=ml heywood n=1;
run;

title3 'Maximum Likelihood Factor Analysis with Two Factors';
proc factor data=SocioEconomics method=ml heywood n=2;
run;
```

```
title3 'Maximum Likelihood Factor Analysis with Three Factors';
proc factor data=SocioEconomics method=ml heywood n=3;
run;
```

Output 44.4.1 displays the results of the analysis with one factor.

Output 44.4.1 Maximum Likelihood Factor Analysis

Maximum Likelihood Factor Analysis with One Factor

The FACTOR Procedure

Input Data Type	Raw Data
Number of Records Read	12
Number of Records Used	12
N for Significance Tests	12

Maximum Likelihood Factor Analysis with One Factor

The FACTOR Procedure
Initial Factor Method: Maximum Likelihood

Prior Communality Estimates: SMC				
Population	**School**	**Employment**	**Services**	**HouseValue**
0.96859160	0.82228514	0.96918082	0.78572440	0.84701921

Preliminary Eigenvalues: Total = 76.1165859 Average = 15.2233172				
	Eigenvalue	**Difference**	**Proportion**	**Cumulative**
1	63.7010086	50.6462895	0.8369	0.8369
2	13.0547191	12.7270798	0.1715	1.0084
3	0.3276393	0.6749199	0.0043	1.0127
4	-0.3472805	0.2722202	-0.0046	1.0081
5	-0.6195007		-0.0081	1.0000

1 factor will be retained by the NFACTOR criterion.

Iteration	Criterion	Ridge	Change	Communalities				
1	6.5429218	0.0000	0.1033	0.93828	0.72227	1.00000	0.71940	0.74371
2	3.1232699	0.0000	0.7288	0.94566	0.02380	1.00000	0.26493	0.01487

Convergence criterion satisfied.

Significance Tests Based on 12 Observations			
Test	**DF**	**Chi-Square**	**Pr > ChiSq**
H0: No common factors	10	54.2517	<.0001
HA: At least one common factor			
H0: 1 Factor is sufficient	5	24.4656	0.0002
HA: More factors are needed			

Output 44.4.1 *continued*

Chi-Square without Bartlett's Correction	34.355969
Akaike's Information Criterion	24.355969
Schwarz's Bayesian Criterion	21.931436
Tucker and Lewis's Reliability Coefficient	0.120231

Squared Canonical Correlations

Factor1
1.0000000

Eigenvalues of the Weighted Reduced Correlation Matrix: Total = 0 Average = 0

	Eigenvalue	Difference
1	Infty	Infty
2	1.92716032	2.15547340
3	-.22831308	0.56464322
4	-.79295630	0.11293464
5	-.90589094	

Factor Pattern

	Factor1
Population	0.97245
School	0.15428
Employment	1.00000
Services	0.51472
HouseValue	0.12193

Variance Explained by Each Factor

Factor	Weighted	Unweighted
Factor1	17.8010629	2.24926004

Final Communality Estimates and Variable Weights

Total Communality: Weighted = 17.801063 Unweighted = 2.249260

Variable	Communality	Weight
Population	0.94565561	18.4011648
School	0.02380349	1.0243839
Employment	1.00000000	Infty
Services	0.26493499	1.3604239
HouseValue	0.01486595	1.0150903

The solution on the second iteration is so close to the optimum that PROC FACTOR cannot find a better solution; hence you receive this message:

```
Convergence criterion satisfied.
```

When this message appears, you should try rerunning PROC FACTOR with different prior communality estimates to make sure that the solution is correct. In this case, other prior estimates lead to the same solution or possibly to worse local optima, as indicated by the information criteria or the chi-square values.

The variable Employment has a communality of 1.0 and, therefore, an infinite weight that is displayed next to the final communality estimate as a missing/infinite value. The first eigenvalue is also infinite. Infinite values are ignored in computing the total of the eigenvalues and the total final communality.

Output 44.4.2 displays the results of the analysis with two factors. The analysis converges without incident. This time, however, the Population variable is a Heywood case.

Output 44.4.2 Maximum Likelihood Factor Analysis: Two Factors

Input Data Type	Raw Data
Number of Records Read	12
Number of Records Used	12
N for Significance Tests	12

Prior Communality Estimates: SMC

Population	School	Employment	Services	HouseValue
0.96859160	0.82228514	0.96918082	0.78572440	0.84701921

Preliminary Eigenvalues:
Total = 76.1165859 Average = 15.2233172

	Eigenvalue	Difference	Proportion	Cumulative
1	63.7010086	50.6462895	0.8369	0.8369
2	13.0547191	12.7270798	0.1715	1.0084
3	0.3276393	0.6749199	0.0043	1.0127
4	-0.3472805	0.2722202	-0.0046	1.0081
5	-0.6195007		-0.0081	1.0000

2 factors will be retained by the NFACTOR criterion.

Iteration	Criterion	Ridge	Change	Communalities			
1	0.3431221	0.0000	0.0471	1.00000	0.80672	0.95058	0.79348	0.89412
2	0.3072178	0.0000	0.0307	1.00000	0.80821	0.96023	0.81048	0.92480
3	0.3067860	0.0000	0.0063	1.00000	0.81149	0.95948	0.81677	0.92023
4	0.3067373	0.0000	0.0022	1.00000	0.80985	0.95963	0.81498	0.92241
5	0.3067321	0.0000	0.0007	1.00000	0.81019	0.95955	0.81569	0.92187

Convergence criterion satisfied.

Significance Tests Based on 12 Observations

Test	DF	Chi-Square	Pr > ChiSq
H0: No common factors	10	54.2517	<.0001
HA: At least one common factor			
H0: 2 Factors are sufficient	1	2.1982	0.1382
HA: More factors are needed			

Output 44.4.2 *continued*

Chi-Square without Bartlett's Correction	3.3740530
Akaike's Information Criterion	1.3740530
Schwarz's Bayesian Criterion	0.8891463
Tucker and Lewis's Reliability Coefficient	0.7292200

Squared Canonical Correlations

Factor1	Factor2
1.0000000	0.9518891

Eigenvalues of the Weighted Reduced Correlation Matrix: Total = 19.7853157 Average = 4.94632893

	Eigenvalue	Difference	Proportion	Cumulative
1	Infty	Infty		
2	19.7853143	19.2421292	1.0000	1.0000
3	0.5431851	0.5829564	0.0275	1.0275
4	-0.0397713	0.4636411	-0.0020	1.0254
5	-0.5034124		-0.0254	1.0000

Factor Pattern

	Factor1	Factor2
Population	1.00000	0.00000
School	0.00975	0.90003
Employment	0.97245	0.11797
Services	0.43887	0.78930
HouseValue	0.02241	0.95989

Variance Explained by Each Factor

Factor	Weighted	Unweighted
Factor1	24.4329707	2.13886057
Factor2	19.7853143	2.36835294

Final Communality Estimates and Variable Weights

Total Communality: Weighted = 44.218285 Unweighted = 4.507214

Variable	Communality	Weight
Population	1.00000000	Infty
School	0.81014489	5.2682940
Employment	0.95957142	24.7246669
Services	0.81560348	5.4256462
HouseValue	0.92189372	12.7996793

The results of the three-factor analysis are shown in Output 44.4.3.

Output 44.4.3 Maximum Likelihood Factor Analysis: Three Factors

Input Data Type	Raw Data
Number of Records Read	12
Number of Records Used	12
N for Significance Tests	12

Prior Communality Estimates: SMC

Population	School	Employment	Services	HouseValue
0.96859160	0.82228514	0.96918082	0.78572440	0.84701921

Preliminary Eigenvalues:
Total = 76.1165859 Average = 15.2233172

	Eigenvalue	Difference	Proportion	Cumulative
1	63.7010086	50.6462895	0.8369	0.8369
2	13.0547191	12.7270798	0.1715	1.0084
3	0.3276393	0.6749199	0.0043	1.0127
4	-0.3472805	0.2722202	-0.0046	1.0081
5	-0.6195007		-0.0081	1.0000

3 factors will be retained by the NFACTOR criterion.

Warning: Too many factors for a unique solution.

Iteration	Criterion	Ridge	Change	Communalities				
1	0.1798029	0.0313	0.0501	0.96081	0.84184	1.00000	0.80175	0.89716
2	0.0016405	0.0313	0.0678	0.98081	0.88713	1.00000	0.79559	0.96500
3	0.0000041	0.0313	0.0094	0.98195	0.88603	1.00000	0.80498	0.96751
4	0.0000000	0.0313	0.0006	0.98202	0.88585	1.00000	0.80561	0.96735

ERROR: Converged, but not to a proper optimum.

Try a different 'PRIORS' statement.

Significance Tests Based on 12 Observations

Test	DF	Chi-Square	Pr > ChiSq
H0: No common factors	10	54.2517	<.0001
HA: At least one common factor			
H0: 3 Factors are sufficient	-2	0.0000	.
HA: More factors are needed			

Chi-Square without Bartlett's Correction	0.0000003
Akaike's Information Criterion	4.0000003
Schwarz's Bayesian Criterion	4.9698136
Tucker and Lewis's Reliability Coefficient	0.0000000

Squared Canonical Correlations

Factor1	Factor2	Factor3
1.0000000	0.9751895	0.6894465

Output 44.4.3 *continued*

Eigenvalues of the Weighted Reduced Correlation Matrix: Total = 41.5254193 Average = 10.3813548				
	Eigenvalue	Difference	Proportion	Cumulative
1	Infty	Infty		
2	39.3054826	37.0854258	0.9465	0.9465
3	2.2200568	2.2199693	0.0535	1.0000
4	0.0000875	0.0002949	0.0000	1.0000
5	-0.0002075		-0.0000	1.0000

Factor Pattern			
	Factor1	Factor2	Factor3
Population	0.97245	-0.11233	-0.15409
School	0.15428	0.89108	0.26083
Employment	1.00000	-0.00000	-0.00000
Services	0.51472	0.72416	-0.12766
HouseValue	0.12193	0.97227	-0.08473

Variance Explained by Each Factor		
Factor	Weighted	Unweighted
Factor1	54.6115241	2.24926004
Factor2	39.3054826	2.27634375
Factor3	2.2200568	0.11525433

Final Communality Estimates and Variable Weights		
Total Communality: Weighted = 96.137063 Unweighted = 4.640858		
Variable	Communality	Weight
Population	0.98201660	55.6066901
School	0.88585165	8.7607194
Employment	1.00000000	Infty
Services	0.80564301	5.1444261
HouseValue	0.96734687	30.6251078

In the results, a warning message is displayed:

```
WARNING:  Too many factors for a unique solution.
```

The number of parameters in the model exceeds the number of elements in the correlation matrix from which they can be estimated, so an infinite number of different perfect solutions can be obtained. The criterion approaches zero at an improper optimum, as indicated by this message:

```
Converged, but not to a proper optimum.
```

The degrees of freedom for the chi-square test are –2, so a probability level cannot be computed for three factors. Note also that the variable Employment is a Heywood case again.

The probability levels for the chi-square test are 0.0001 for the hypothesis of no common factors, 0.0002 for one common factor, and 0.1382 for two common factors. Therefore, the two-factor model seems to be an adequate representation. Akaike's information criterion and Schwarz's Bayesian criterion attain their minimum values at two common factors, so there is little doubt that two factors are appropriate for these data.

Example 44.5: Using Confidence Intervals to Locate Salient Factor Loadings

(View the complete code for this example (facex5.sas) in the example repository.)

This example illustrates how you can use the standard errors and confidence intervals to understand the pattern of factor loadings under the maximum likelihood estimation. There are nine tests and you want a three-factor solution (N=3) for a correlation matrix based on 200 observations. The following statements define the input data set and specify the desirable analysis by the FACTOR procedure:

```
data test(type=corr);
   title 'Quartimin-Rotated Factor Solution with Standard Errors';
   input _name_ $ test1-test9;
   _type_ = 'corr';
   datalines;
Test1      1  .561  .602  .290  .404  .328  .367  .179 -.268
Test2   .561     1  .743  .414  .526  .442  .523  .289 -.399
Test3   .602  .743     1  .286  .343  .361  .679  .456 -.532
Test4   .290  .414  .286     1  .677  .446  .412  .400 -.491
Test5   .404  .526  .343  .677     1  .584  .408  .299 -.466
Test6   .328  .442  .361  .446  .584     1  .333  .178 -.306
Test7   .367  .523  .679  .412  .408  .333     1  .711 -.760
Test8   .179  .289  .456  .400  .299  .178  .711     1 -.725
Test9  -.268 -.399 -.532 -.491 -.466 -.306 -.760 -.725     1
;

title2 'A nine-variable-three-factor example';
proc factor data=test method=ml reorder rotate=quartimin
   nobs=200 n=3 se cover=.45 alpha=.1;
run;
```

In the PROC FACTOR statement, you apply quartimin rotation with (default) Kaiser normalization. You define loadings with magnitudes greater than 0.45 to be salient (COVER=0.45) and use 90% confidence intervals (ALPHA=0.1) to judge the salience. The REORDER option is specified so that variables that have similar loadings with factors are clustered together.

After the quartimin rotation, the correlation matrix for factors is shown in Output 44.5.1.

Output 44.5.1 Quartimin-Rotated Factor Correlations with Standard Errors

Inter-Factor Correlations With 90% confidence limits Estimate/StdErr/LowerCL/UpperCL			
	Factor1	**Factor2**	**Factor3**
Factor1	1.00000	0.41283	0.38304
	0.00000	0.06267	0.06060
	.	0.30475	0.27919
	.	0.51041	0.47804
Factor2	0.41283	1.00000	0.47006
	0.06267	0.00000	0.05116
	0.30475	.	0.38177
	0.51041	.	0.54986
Factor3	0.38304	0.47006	1.00000
	0.06060	0.05116	0.00000
	0.27919	0.38177	.
	0.47804	0.54986	.

The factors are medium to highly correlated. The confidence intervals seem to be very wide, suggesting that the estimation of factor correlations might not be very accurate for this sample size. For example, the 90% confidence interval for the correlation between Factor1 and Factor2 is (0.30, 0.51), a range of 0.21. You might need a larger sample to get a narrower interval, or you might need a better estimation.

Next, coverage displays for factor loadings are shown in Output 44.5.2.

Output 44.5.2 Using the Rotated Factor Pattern to Interpret the Factors

Rotated Factor Pattern (Standardized Regression Coefficients)
With 90% confidence limits; Cover |*| = 0.45?
Estimate/StdErr/LowerCL/UpperCL/Coverage Display

	Factor1	Factor2	Factor3
test8	0.86810	-0.05045	0.00114
	0.03282	0.03185	0.03087
	0.80271	-0.10265	-0.04959
	0.91286	0.00204	0.05187
	0*[]	*[0]	[0]*
test7	0.73204	0.27296	0.01098
	0.04434	0.05292	0.03838
	0.65040	0.18390	-0.05211
	0.79697	0.35758	0.07399
	0*[]	0[]*	[0]*
test9	-0.79654	-0.01230	-0.17307
	0.03948	0.04225	0.04420
	-0.85291	-0.08163	-0.24472
	-0.72180	0.05715	-0.09955
	[]*0	*[0]	*[]0
test3	0.27715	0.91156	-0.19727
	0.05489	0.04877	0.02981
	0.18464	0.78650	-0.24577
	0.36478	0.96481	-0.14778
	0[]*	0*[]	*[]0
test2	0.01063	0.71540	0.20500
	0.05060	0.05148	0.05496
	-0.07248	0.61982	0.11310
	0.09359	0.79007	0.29342
	[0]*	0*[]	0[]*
test1	-0.07356	0.63815	0.13983
	0.04245	0.05380	0.05597
	-0.14292	0.54114	0.04682
	-0.00348	0.71839	0.23044
	*[]0	0*[]	0[]*
test5	0.00863	0.03234	0.91282
	0.04394	0.04387	0.04509
	-0.06356	-0.03986	0.80030
	0.08073	0.10421	0.96323
	[0]*	[0]*	0*[]
test4	0.22357	-0.07576	0.67925
	0.05956	0.03640	0.05434
	0.12366	-0.13528	0.57955
	0.31900	-0.01569	0.75891
	0[]*	*[]0	0*[]
test6	-0.04295	0.21911	0.53183
	0.05114	0.07481	0.06905
	-0.12656	0.09319	0.40893
	0.04127	0.33813	0.63578
	*[0]	0[]*	0[*]

The coverage displays in Output 44.5.2 show that Test8, Test7, and Test9 have salient relationships with Factor1. The coverage displays are either '0*[ ]' or '[ ]*0', indicating that the entire 90% confidence intervals for the corresponding loadings are beyond the salience value at 0.45. On the other hand, the coverage display for Test3 on Factor1 is '0[ ]*'. This indicates that even though the loading estimate is significantly larger than zero, it is not large enough to be salient. Similarly, Test3, Test2, and Test1 have salient relationships with Factor2, while Test5 and Test4 have salient relationships with Factor3. For Test6, its relationship with Factor3 is a little bit ambiguous; the 90% confidence interval approximately covers values between 0.40

and 0.64. This means that the population value might have been smaller or larger than 0.45. It is marginal evidence for a salient relationship.

For oblique factor solutions, some researchers prefer to examine the factor structure loadings, which represent correlations, for determining salient relationships. In Output 44.5.3, the factor structure loadings and the associated standard error estimates and coverage displays are shown.

Output 44.5.3 Using the Factor Structure to Interpret the Factors

Factor Structure (Correlations)
With 90% confidence limits; Cover |*| = 0.45?
Estimate/StdErr/LowerCL/UpperCL/Coverage Display

	Factor1	Factor2	Factor3
test8	0.84771	0.30847	0.30994
	0.02871	0.06593	0.06263
	0.79324	0.19641	0.20363
	0.88872	0.41257	0.40904
	0*[]	0[]*	0[]*
test7	0.84894	0.58033	0.41970
	0.02688	0.05265	0.06060
	0.79834	0.48721	0.31523
	0.88764	0.66041	0.51412
	0*[]	0*[]	0[*]
test9	-0.86791	-0.42248	-0.48396
	0.02522	0.06187	0.05504
	-0.90381	-0.51873	-0.56921
	-0.81987	-0.31567	-0.38841
	[]*0	[*]0	[*]0
test3	0.57790	0.93325	0.33738
	0.05069	0.02953	0.06779
	0.48853	0.86340	0.22157
	0.65528	0.96799	0.44380
	0*[]	0*[]	0[]*
test2	0.38449	0.81615	0.54535
	0.06143	0.03106	0.05456
	0.27914	0.75829	0.44946
	0.48070	0.86126	0.62883
	0[*]	0*[]	0[*]
test1	0.24345	0.67351	0.41162
	0.06864	0.04284	0.05995
	0.12771	0.59680	0.30846
	0.35264	0.73802	0.50522
	0[]*	0*[]	0[*]
test5	0.37163	0.46498	0.93132
	0.06092	0.04979	0.03277
	0.26739	0.37923	0.85159
	0.46727	0.54282	0.96894
	0[*]	0[*]	0*[]
test4	0.45248	0.33583	0.72927
	0.05876	0.06289	0.04061
	0.35072	0.22867	0.65527
	0.54367	0.43494	0.78941
	0[*]	0[]*	0*[]
test6	0.25122	0.45137	0.61837
	0.07140	0.05858	0.05051
	0.13061	0.34997	0.52833
	0.36450	0.54232	0.69465
	0[]*	0[*]	0*[]

The interpretations based on the factor structure matrix do not change much from that based on the factor

loadings except for Test3 and Test9. Test9 now has a salient correlation with Factor3. For Test3, it has salient correlations with both Factor1 and Factor2. Fortunately, there are still tests that have salient correlations only with either Factor1 or Factor2 (but not both). This would make interpretations of factors less problematic.

Example 44.6: Creating Path Diagrams for Factor Solutions

(View the complete code for this example (facex6.sas) in the example repository.)

The section "Getting Started: FACTOR Procedure" on page 2868 analyzes a data set that contains 14 ratings of 103 police officers to demonstrate some basic techniques in factor analysis. To illustrate the creation and uses of path diagrams, this example analyzes this data set again by using the following statements:

```
ods graphics on;

proc factor data=jobratings(drop='Overall Rating'n)
   priors=smc rotate=quartimin plots=pathdiagram;
label
   'Judgment under Pressure'n ='Judgment'
   'Communication Skills'n = 'Comm Skills'
   'Interpersonal Sensitivity'n = 'Sensitivity'
   'Willingness to Confront Problems'n = 'Confront Problems'
   'Desire for Self-Improvement'n = 'Self-Improve'
   'Observational Skills'n = 'Obs Skills'
   'Dependability'n = 'Dependable';
run;
```

The PRIORS=SMC option specifies that the squared multiple correlations are to be used as the prior communality estimates. As a result, the factors are extracted by the principal factor method. The ROTATE=QUARTIMIN option requests the use of the quartimin rotation to obtain the final factor solution. The PLOTS=PATHDIAGRAM option requests a path diagram for the final solution. The LABEL statement specifies labels for variables.

When variables do not have labels, PROC FACTOR displays the variable names in path diagrams. But when variables have labels, PROC FACTOR displays labels, instead of variable names, in path diagrams. Because some variables in this example have very long variables names, PROC FACTOR might truncate these long names in the output path diagram. Therefore, to avoid truncations in the output diagram, you can either create a data set with shorter variable names or use the LABEL statement to specify shorter labels. This example illustrates the use of the LABEL statement.

Except for the PLOTS=PATHDIAGRAM option, previous examples have already described the FACTOR options that are used in this example. Therefore, this example focuses only on the creation of path diagrams.

Output 44.6.1 and Output 44.6.2 show the quartimin-rotated factor correlations and factor pattern, respectively.

Output 44.6.1 Quartimin-Rotated Factor Correlations

Inter-Factor Correlations			
	Factor1	**Factor2**	**Factor3**
Factor1	1.00000	0.36103	0.34823
Factor2	0.36103	1.00000	0.55721
Factor3	0.34823	0.55721	1.00000

Output 44.6.2 Quartimin-Rotated Factor Pattern

Rotated Factor Pattern (Standardized Regression Coefficients)				
		Factor1	**Factor2**	**Factor3**
Communication Skills	Comm Skills	0.21280	0.13541	0.61091
Problem Solving		0.17222	0.01424	0.68767
Learning Ability		-0.09904	0.24961	0.65430
Judgment under Pressure	Judgment	0.49876	-0.02005	0.48879
Observational Skills	Obs Skills	-0.15748	0.67661	0.30273
Willingness to Confront Problems	Confront Problems	-0.19106	0.66639	0.31135
Interest in People		0.84249	0.12734	-0.00710
Interpersonal Sensitivity	Sensitivity	0.87832	-0.12964	0.15116
Desire for Self-Improvement	Self-Improve	0.19078	0.49891	0.23297
Appearance		0.05254	0.53846	0.10857
Dependability	Dependable	0.39241	0.50035	0.12032
Physical Ability		0.14404	0.63901	-0.15220
Integrity		0.68277	0.32719	-0.01887

Output 44.6.3 shows the path diagram for the quartimin-rotated factor solution. The path diagram represents correlations among factors by double-headed links or paths. For example, Output 44.6.3 represents the correlation between Factor1 and Factor2 by a curved doubled-headed link. The numerical value, 0.36, is the correlation between the two factors, as can be verified from the table in Output 44.6.1. Similarly, Output 44.6.3 shows other factor correlations by curved doubled-headed links.

Output 44.6.3 Default Path Diagram for the Quartimin-Rotated Solution

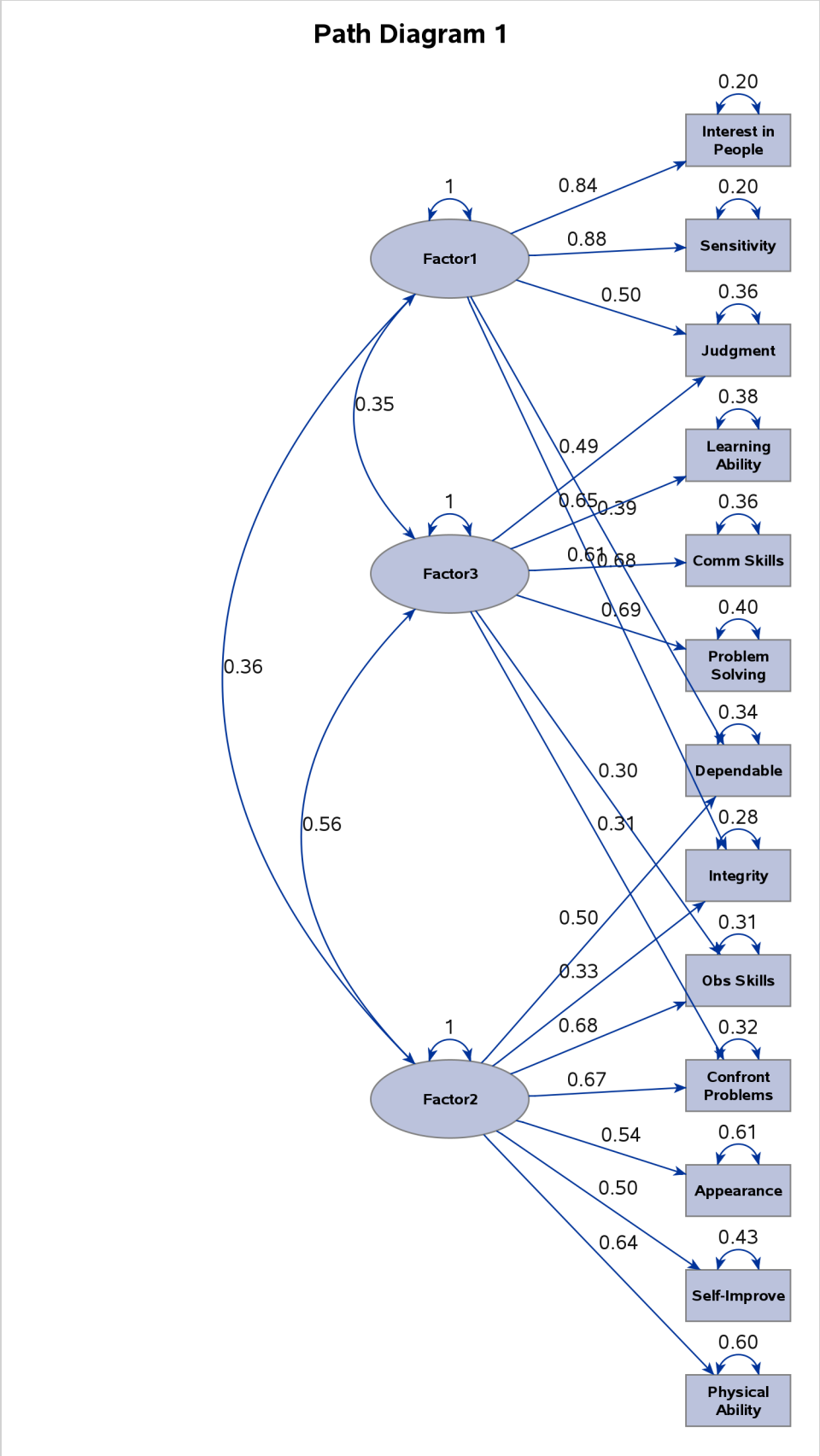

The path diagram in Output 44.6.3 also represents factor variances and error variances by double-headed links. However, each of these links points to an individual variable, rather than to a pair of variables as the double-headed links for correlations do. The path diagram also displays the numerical values of factor variances or error variances next to the associated links.

The directed links from factors to variables in the path diagram represent the effects of factors on the variables. The path diagram displays the numerical values of these effects, which are the loading estimates that are shown in Output 44.6.2. However, to aid the interpretation of the factors, the path diagram does not show all factor loadings or their corresponding links. By default, the path diagram displays only the links that have loadings greater than 0.3 in magnitude. For example, instead of associating Factor1 with all variables, the path diagram in Output 44.6.3 displays only five directed links from Factor1 to the variables. The weaker links that have loadings less than 0.3 are not shown.

The use of the 0.3 loading value (or greater in magnitude) for relating factors to variables is referred to as the "0.3-rule" in the field of factor analysis. However, this is only a convention, and sometimes you might want to use a different criterion to interpret the factors. For example, the path diagram in Output 44.6.3 shows that variables Dependability, Integrity, and Observational Skills are all associated with more than one factor. Hence, factors might not be interpreted unambiguously.

One way to tackle this interpretation problem is to set a stricter criterion for interpreting factors. You can use the FUZZ= option to set such a criterion. For example, you specify the following PATHDIAGRAM statement to display only the strong directed links that are associated with a 0.4 or greater magnitude in the loading estimates:

```
pathdiagram fuzz=0.4 title='Directed Paths with Loadings Greater Than 0.4';
```

The preceding statement also uses the TITLE= option to specify a customized title for the path diagram. Output 44.6.4 shows the resulting path diagram. In this path diagram, only one observed variable is linked to two factors. All other observed variables link to unique factors. Therefore, compared to the path diagram in Output 44.6.3, the path diagram in Output 44.6.4 provides a much "cleaner" picture for interpreting the factors.

Output 44.6.4 Path Diagram Showing Strong Links

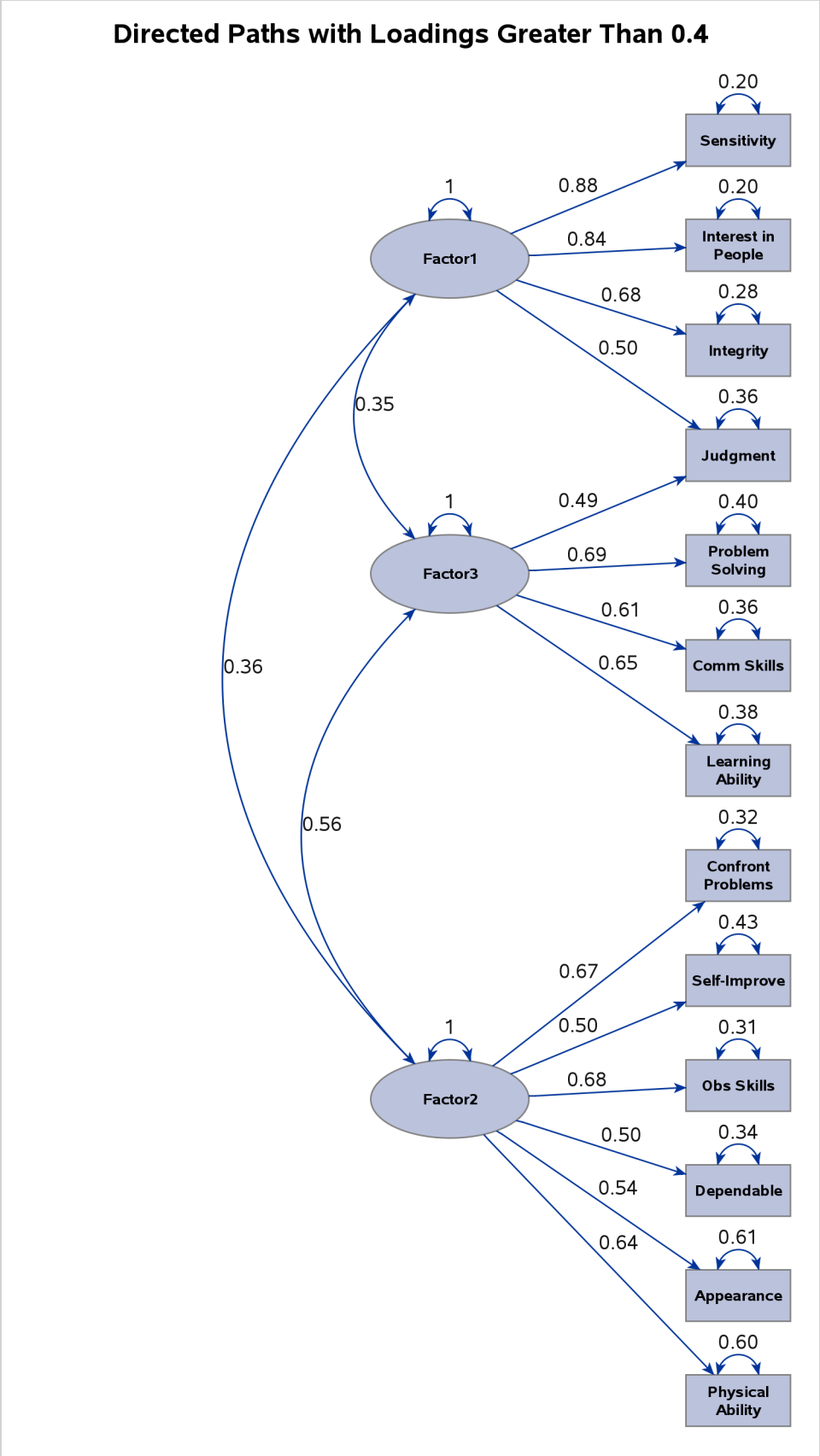

The current example has 13 observed variables in the path diagram. By default, PROC FACTOR uses the process-flow algorithm to lay out the variables. However, when the number of observed variables becomes large, the process-flow algorithm needs a lot of vertical space to align all observed variables in a vertical line. Displaying such a "long" path diagram in limited space (for example, in a page) might compromise the clarity of the path diagram.

To handle this issue, PROC FACTOR switches to the GRIP algorithm when the number of variables is greater than 14. However, you can override the layout algorithm whenever you find it useful to do so. For example, the ARRANGE=GRIP option in the following PATHDIAGRAM statement requests that the GRIP algorithm be used:

```
pathdiagram fuzz=0.4 arrange=grip scale=0.85 notitle;
```

The SCALE= option shrinks the nodes so that the nodes are well-separated in the path diagram. If you do not use this option, some nodes would have been overlapped. The NOTITLE option suppresses the display of the title. Output 44.6.5 shows the resulting path diagram, which spreads out the variables instead of aligning them vertically, as it does when it uses the process-flow algorithm in Output 44.6.4.

Output 44.6.5 Path Diagram Showing Strong Links by Using the ARRANGE=GRIP Algorithm

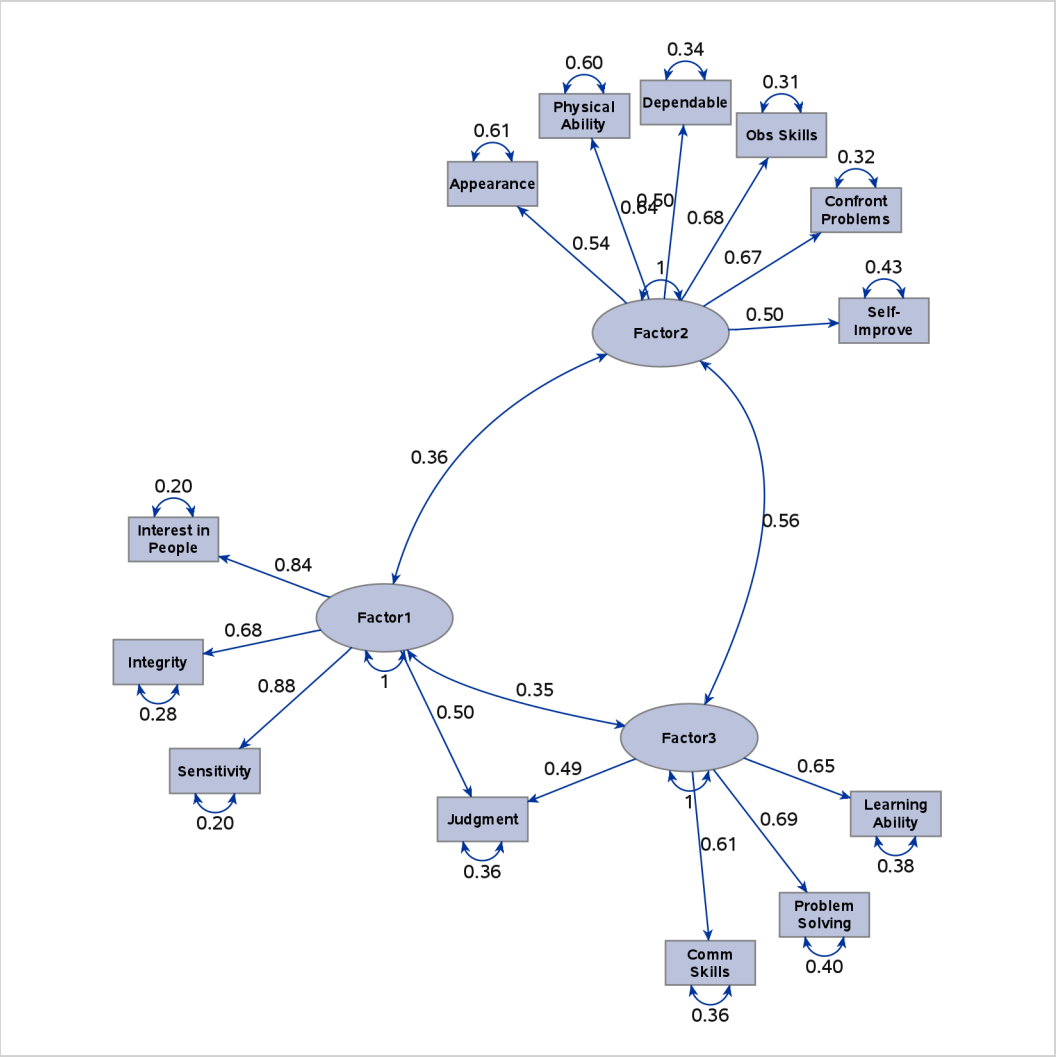

For more information about the options for customizing path diagrams, see the PATHDIAGRAM statement.

References

Akaike, H. (1973). "Information Theory and an Extension of the Maximum Likelihood Principle." In *Proceedings of the Second International Symposium on Information Theory*, edited by B. N. Petrov and F. Csáki, 267–281. Budapest: Akademiai Kiado.

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Akaike, H. (1987). "Factor Analysis and AIC." *Psychometrika* 52:317–332.

Archer, C. O., and Jennrich, R. I. (1973). "Standard Errors for Orthogonally Rotated Factor Loadings." *Psychometrika* 38:581–592.

Bickel, P. J., and Doksum, K. A. (1977). *Mathematical Statistics*. San Francisco: Holden-Day.

Browne, M. W. (1982). "Covariance Structures." In *Topics in Applied Multivariate Analysis*, edited by D. M. Hawkins, 72–141. Cambridge: Cambridge University Press.

Browne, M. W., Cudeck, R., Tateneni, K., and Mels, G. (2010). "CEFA: Comprehensive Exploratory Factor Analysis, Version 3.04." `https://u.osu.edu/quantpsy/michael-brownes-software/`.

Cattell, R. B. (1966). "The Scree Test for the Number of Factors." *Multivariate Behavioral Research* 1:245–276.

Cattell, R. B. (1978). *The Scientific Use of Factor Analysis*. New York: Plenum.

Cattell, R. B., and Vogelman, S. (1977). "A Comprehensive Trial of the Scree and KG Criteria for Determining the Number of Factors." *Multivariate Behavioral Research* 12:289–325.

Cerny, B. A., and Kaiser, H. F. (1977). "A Study of a Measure of Sampling Adequacy for Factor-Analytic Correlation Matrices." *Multivariate Behavioral Research* 12:43–47.

Crawford, C. B., and Ferguson, G. A. (1970). "A General Rotation Criterion and Its Use in Orthogonal Rotation." *Psychometrika* 35:321–332.

Cureton, E. E. (1968). *A Factor Analysis of Project TALENT Tests and Four Other Test Batteries.* Interim Report 4 to the U.S. Office of Education, Cooperative Research Project No. 3051. Palo Alto, CA: Project TALENT Office, American Institutes for Research and University of Pittsburgh.

Cureton, E. E., and Mulaik, S. A. (1975). "The Weighted Varimax Rotation and the Promax Rotation." *Psychometrika* 40:183–195.

Dziuban, C. D., and Harris, C. W. (1973). "On the Extraction of Components and the Applicability of the Factor Model." *American Educational Research Journal* 10:93–99.

Fuller, W. A. (1987). *Measurement Error Models*. New York: John Wiley & Sons.

Geweke, J., and Singleton, K. J. (1980). "Interpreting the Likelihood Ratio Statistic in Factor Models When Sample Size Is Small." *Journal of the American Statistical Association* 75:133–137.

Glorfeld, L. W. (1995). "An Improvement on Horn's Parallel Analysis Methodology for Selecting the Correct Number of Factors to Retain." *Educational and Psychological Measurement* 55:377–393.

Gorsuch, R. L. (1974). *Factor Analysis*. Philadelphia: W. B. Saunders.

Harman, H. H. (1976). *Modern Factor Analysis*. 3rd ed. Chicago: University of Chicago Press.

Harris, C. W. (1962). "Some Rao-Guttman Relationships." *Psychometrika* 27:247–263.

Hayashi, K., and Yung, Y.-F. (1999). "Standard Errors for the Class of Orthomax-Rotated Factor Loadings: Some Matrix Results." *Psychometrika* 64:451–460.

Horn, J. L. (1965). "A Rationale and Test for the Number of Factors in Factor Analysis." *Psychometrika* 30:179–185.

Horn, J. L., and Engstrom, R. (1979). "Cattell's Scree Test in Relation to Bartlett's Chi-Square Test and Other Observations on the Number of Factors Problem." *Multivariate Behavioral Research* 14:283–300.

Jennrich, R. I. (1973). "Standard Errors for Obliquely Rotated Factor Loadings." *Psychometrika* 38:593–604.

Jennrich, R. I. (1974). "Simplified Formulae for Standard Errors in Maximum-Likelihood Factor Analysis." *British Journal of Mathematical and Statistical Psychology* 27:122–131.

Jöreskog, K. G. (1977). "Factor Analysis by Least-Squares and Maximum Likelihood Methods." In *Statistical Methods for Digital Computers*, edited by K. Enslein, A. Ralston, and H. Wilf, 125–165. New York: John Wiley & Sons.

Kaiser, H. F. (1963). "Image Analysis." In *Problems in Measuring Change*, edited by C. W. Harris, 156–166. Madison: University of Wisconsin Press.

Kaiser, H. F. (1970). "A Second Generation Little Jiffy." *Psychometrika* 35:401–415.

Kaiser, H. F., and Rice, J. (1974). "Little Jiffy, Mark IV." *Educational and Psychological Measurement* 34:111–117.

Kerlinger, F. N., and Pedhazur, E. J. (1973). *Multiple Regression in Behavioral Research*. New York: Holt, Rinehart & Winston.

Kim, J. O., and Mueller, C. W. (1978a). *Factor Analysis: Statistical Methods and Practical Issues.* Vol. 07-014 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Kim, J. O., and Mueller, C. W. (1978b). *Introduction to Factor Analysis: What It Is and How to Do It.* Vol. 07-013 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Lawley, D. N., and Maxwell, A. E. (1971). *Factor Analysis as a Statistical Method.* New York: Macmillan.

Lee, H. B., and Comrey, A. L. (1979). "Distortions in a Commonly Used Factor Analytic Procedure." *Multivariate Behavioral Research* 14:301–321.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Morrison, D. F. (1976). *Multivariate Statistical Methods*. 2nd ed. New York: McGraw-Hill.

Mulaik, S. A. (1972). *The Foundations of Factor Analysis*. New York: McGraw-Hill.

Rao, C. R. (1955). "Estimation and Tests of Significance in Factor Analysis." *Psychometrika* 20:93–111.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Spearman, C. (1904). "General Intelligence Objectively Determined and Measured." *American Journal of Psychology* 15:201–293.

Stewart, D. W. (1981). "The Application and Misapplication of Factor Analysis in Marketing Research." *Journal of Marketing Research* 18:51–62.

Tucker, L. R., and Lewis, C. (1973). "A Reliability Coefficient for Maximum Likelihood Factor Analysis." *Psychometrika* 38:1–10.

Velicer, W. F. (1976). "Determining the Number of Components from the Matrix of Partial Correlations." *Psychometrika* 41:321–327.

Velicer, W. F., Eaton, C. A., and Fava, J. L. (2000). "Construct Explication through Factor or Component Analysis: A Review and Evaluation of Alternative Procedures for Determining the Number of Factors or Components." In *Problems and Solutions in Human Assessment: Honoring Douglas N. Jackson at Seventy*, edited by R. D. Goffin and E. Helmes, 41–71. Boston: Kluwer.

Yung, Y.-F., and Hayashi, K. (2001). "A Computationally Efficient Method for Obtaining Standard Error Estimates for the Promax and Related Solutions." *British Journal of Mathematical and Statistical Psychology* 54:125–138.

Chapter 45
The FASTCLUS Procedure

Contents

Overview: FASTCLUS Procedure **2970**
- Background 2971

Getting Started: FASTCLUS Procedure **2972**

Syntax: FASTCLUS Procedure **2979**
- PROC FASTCLUS Statement 2979
- BY Statement 2986
- FREQ Statement 2987
- ID Statement 2988
- VAR Statement 2988
- WEIGHT Statement 2988

Details: FASTCLUS Procedure **2988**
- Updates in the FASTCLUS Procedure 2988
- Missing Values 2989
- Output Data Sets 2989
- Computational Resources 2993
- Using PROC FASTCLUS 2994
- Displayed Output 2996
- ODS Table Names 2998

Examples: FASTCLUS Procedure **2999**
- Example 45.1: Fisher's Iris Data 2999
- Example 45.2: Outliers 3007

References **3016**

Overview: FASTCLUS Procedure

The FASTCLUS procedure performs a disjoint cluster analysis on the basis of distances computed from one or more quantitative variables. The observations are divided into clusters such that every observation belongs to one and only one cluster; the clusters do not form a tree structure as they do in the CLUSTER procedure. If you want separate analyses for different numbers of clusters, you can run PROC FASTCLUS once for each analysis. Alternatively, to do hierarchical clustering on a large data set, use PROC FASTCLUS to find initial clusters, and then use those initial clusters as input to PROC CLUSTER.

By default, the FASTCLUS procedure uses Euclidean distances, so the cluster centers are based on least squares estimation. This kind of clustering method is often called a *k-means model*, since the cluster centers are the means of the observations assigned to each cluster when the algorithm is run to complete convergence. Each iteration reduces the least squares criterion until convergence is achieved.

Often there is no need to run the FASTCLUS procedure to convergence. PROC FASTCLUS is designed to find good clusters (but not necessarily the best possible clusters) with only two or three passes through the data set. The initialization method of PROC FASTCLUS guarantees that, if there exist clusters such that all distances between observations in the same cluster are less than all distances between observations in different clusters, and if you tell PROC FASTCLUS the correct number of clusters to find, it can always find such a clustering without iterating. Even with clusters that are not as well separated, PROC FASTCLUS usually finds initial seeds that are sufficiently good that few iterations are required. Hence, by default, PROC FASTCLUS performs only one iteration.

The initialization method used by the FASTCLUS procedure makes it sensitive to outliers. PROC FASTCLUS can be an effective procedure for detecting outliers because outliers often appear as clusters with only one member.

The FASTCLUS procedure can use an L_p (least pth powers) clustering criterion (Spath 1985, pp. 62–63) instead of the least squares (L_2) criterion used in k-means clustering methods. The LEAST=p option specifies the power p to be used. Using the LEAST= option increases execution time since more iterations are usually required, and the default iteration limit is increased when you specify LEAST=p. Values of p less than 2 reduce the effect of outliers on the cluster centers compared with least squares methods; values of p greater than 2 increase the effect of outliers.

The FASTCLUS procedure is intended for use with large data sets, with 100 or more observations. With small data sets, the results can be highly sensitive to the order of the observations in the data set.

PROC FASTCLUS uses algorithms that place a larger influence on variables with larger variance, so it might be necessary to standardize the variables before performing the cluster analysis. See the "Using PROC FASTCLUS" section for standardization details.

PROC FASTCLUS produces brief summaries of the clusters it finds. For more extensive examination of the clusters, you can request an output data set containing a cluster membership variable.

Background

The FASTCLUS procedure combines an effective method for finding initial clusters with a standard iterative algorithm for minimizing the sum of squared distances from the cluster means. The result is an efficient procedure for disjoint clustering of large data sets. PROC FASTCLUS was directly inspired by the Hartigan (1975) *leader algorithm* and the MacQueen (1967) *k-means algorithm*. PROC FASTCLUS uses a method that Anderberg (1973) calls *nearest centroid sorting*. A set of points called *cluster seeds* is selected as a first guess of the means of the clusters. Each observation is assigned to the nearest seed to form temporary clusters. The seeds are then replaced by the means of the temporary clusters, and the process is repeated until no further changes occur in the clusters. Similar techniques are described in most references on clustering (Anderberg 1973; Hartigan 1975; Everitt 1980; Spath 1980).

The FASTCLUS procedure differs from other nearest centroid sorting methods in the way the initial cluster seeds are selected. The importance of initial seed selection is demonstrated by Milligan (1980).

The clustering is done on the basis of Euclidean distances computed from one or more numeric variables. If there are missing values, PROC FASTCLUS computes an adjusted distance by using the nonmissing values. Observations that are very close to each other are usually assigned to the same cluster, while observations that are far apart are in different clusters.

The FASTCLUS procedure operates in four steps:

1. Observations called *cluster seeds* are selected.
2. If you specify the DRIFT option, temporary clusters are formed by assigning each observation to the cluster with the nearest seed. Each time an observation is assigned, the cluster seed is updated as the current mean of the cluster. This method is sometimes called *incremental*, *on-line*, or *adaptive* training.
3. If the maximum number of iterations is greater than zero, clusters are formed by assigning each observation to the nearest seed. After all observations are assigned, the cluster seeds are replaced by either the cluster means or other location estimates (cluster centers) appropriate to the LEAST=*p* option. This step can be repeated until the changes in the cluster seeds become small or zero (MAXITER=$n \geq 1$).
4. Final clusters are formed by assigning each observation to the nearest seed.

If PROC FASTCLUS runs to complete convergence, the final cluster seeds will equal the cluster means or cluster centers. If PROC FASTCLUS terminates before complete convergence, which often happens with the default settings, the final cluster seeds might not equal the cluster means or cluster centers. If you want complete convergence, specify CONVERGE=0 and a large value for the MAXITER= option.

The initial cluster seeds must be observations with no missing values. You can specify the maximum number of seeds (and, hence, clusters) by using the MAXCLUSTERS= option. You can also specify a minimum distance by which the seeds must be separated by using the RADIUS= option.

PROC FASTCLUS always selects the first complete (no missing values) observation as the first seed. The next complete observation that is separated from the first seed by at least the distance specified in the RADIUS= option becomes the second seed. Later observations are selected as new seeds if they are separated from all previous seeds by at least the radius, as long as the maximum number of seeds is not exceeded.

If an observation is complete but fails to qualify as a new seed, PROC FASTCLUS considers using it to replace one of the old seeds. Two tests are made to see if the observation can qualify as a new seed.

First, an old seed is replaced if the distance between the observation and the closest seed is greater than the minimum distance between seeds. The seed that is replaced is selected from the two seeds that are closest to each other. The seed that is replaced is the one of these two with the shortest distance to the closest of the remaining seeds when the other seed is replaced by the current observation.

If the observation fails the first test for seed replacement, a second test is made. The observation replaces the nearest seed if the smallest distance from the observation to all seeds other than the nearest one is greater than the shortest distance from the nearest seed to all other seeds. If the observation fails this test, PROC FASTCLUS goes on to the next observation.

You can specify the REPLACE= option to limit seed replacement. You can omit the second test for seed replacement (REPLACE=PART), causing PROC FASTCLUS to run faster, but the seeds selected might not be as widely separated as those obtained by the default method. You can also suppress seed replacement entirely by specifying REPLACE=NONE. In this case, PROC FASTCLUS runs much faster, but you must choose a good value for the RADIUS= option in order to get good clusters. This method is similar to the Hartigan (1975, pp. 74–78) leader algorithm and the *simple cluster seeking algorithm* described by Tou and Gonzalez (1974, pp. 90–92).

Getting Started: FASTCLUS Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/fastgs.sas.)

The following example demonstrates how to use the FASTCLUS procedure to compute disjoint clusters of observations in a SAS data set.

The data in this example are measurements taken on 159 freshwater fish caught from the same lake (Laengelmaevesi) near Tampere in Finland. This data set is available from Puranen.

The species (bream, parkki, pike, perch, roach, smelt, and whitefish), weight, three different length measurements (measured from the nose of the fish to the beginning of its tail, the notch of its tail, and the end of its tail), height, and width of each fish are tallied. The height and width are recorded as percentages of the third length variable.

Suppose that you want to group empirically the fish measurements into clusters and that you want to associate the clusters with the species. You can use the FASTCLUS procedure to perform a cluster analysis.

The following DATA step creates the SAS data set Fish:

```
proc format;
   value specfmt
      1='Bream'
      2='Roach'
      3='Whitefish'
      4='Parkki'
      5='Perch'
      6='Pike'
      7='Smelt';
```

```
run;

data fish (drop=HtPct WidthPct);
   title 'Fish Measurement Data';
   input Species Weight Length1 Length2 Length3 HtPct WidthPct @@;

   *** transform variables;
   if Weight <= 0 or Weight =. then delete;
   Weight3=Weight**(1/3);
   Height=HtPct*Length3/(Weight3*100);
   Width=WidthPct*Length3/(Weight3*100);
   Length1=Length1/Weight3;
   Length2=Length2/Weight3;
   Length3=Length3/Weight3;
   logLengthRatio=log(Length3/Length1);

   format Species specfmt.;
   symbol = put(Species, specfmt2.);
   datalines;
1  242.0 23.2 25.4 30.0 38.4 13.4 1  290.0 24.0 26.3 31.2 40.0 13.8
1  340.0 23.9 26.5 31.1 39.8 15.1 1  363.0 26.3 29.0 33.5 38.0 13.3
1  430.0 26.5 29.0 34.0 36.6 15.1 1  450.0 26.8 29.7 34.7 39.2 14.2
1  500.0 26.8 29.7 34.5 41.1 15.3 1  390.0 27.6 30.0 35.0 36.2 13.4
1  450.0 27.6 30.0 35.1 39.9 13.8 1  500.0 28.5 30.7 36.2 39.3 13.7
1  475.0 28.4 31.0 36.2 39.4 14.1 1  500.0 28.7 31.0 36.2 39.7 13.3
1  500.0 29.1 31.5 36.4 37.8 12.0 1     .  29.5 32.0 37.3 37.3 13.6
1  600.0 29.4 32.0 37.2 40.2 13.9 1  600.0 29.4 32.0 37.2 41.5 15.0
1  700.0 30.4 33.0 38.3 38.8 13.8 1  700.0 30.4 33.0 38.5 38.8 13.5

   ... more lines ...

7   19.7 13.2 14.3 15.2 18.9 13.6 7   19.9 13.8 15.0 16.2 18.1 11.6
;
```

The double trailing at sign (@@) in the INPUT statement specifies that observations are input from each line until all values are read. The variables are rescaled in order to adjust for dimensionality. Because the new variables Weight3–logLengthRatio depend on the variable Weight, observations with missing values for Weight are not added to the data set. Consequently, there are 157 observations in the SAS data set Fish.

In the Fish data set, the variables are not measured in the same units and cannot be assumed to have equal variance. Therefore, it is necessary to standardize the variables before performing the cluster analysis.

The following statements standardize the variables and perform a cluster analysis on the standardized data:

```
proc stdize data=Fish out=Stand method=std;
   var Length1 logLengthRatio Height Width Weight3;
run;

proc fastclus data=Stand out=Clust
              maxclusters=7 maxiter=100;
   var Length1 logLengthRatio Height Width Weight3;
run;
```

The STDIZE procedure is first used to standardize all the analysis variables to a mean of 0 and standard deviation of 1. The procedure creates the output data set Stand to contain the transformed variables (for

detailed information,see Chapter 115, "The STDIZE Procedure").

The FASTCLUS procedure then uses the data set Stand as input and creates the data set Clust. This output data set contains the original variables and two new variables, Cluster and Distance. The variable Cluster contains the cluster number to which each observation has been assigned. The variable Distance gives the distance from the observation to its cluster seed.

It is usually desirable to try several values of the MAXCLUSTERS= option. A reasonable beginning for this example is to use MAXCLUSTERS=7, since there are seven species of fish represented in the data set Fish.

The VAR statement specifies the variables used in the cluster analysis.

The results from this analysis are displayed in the following figures.

Figure 45.1 Initial Seeds Used in the FASTCLUS Procedure

Fish Measurement Data

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=7 Maxiter=100 Converge=0.02

Initial Seeds					
Cluster	**Length1**	**logLengthRatio**	**Height**	**Width**	**Weight3**
1	1.388338414	-0.979577858	-1.594561848	-2.254050655	2.103447062
2	-1.117178039	-0.877218192	-0.336166276	2.528114070	1.170706464
3	2.393997461	-0.662642015	-0.930738701	-2.073879107	-1.839325419
4	-0.495085516	-0.964041012	-0.265106856	-0.028245072	1.536846394
5	-0.728772773	0.540096664	1.130501398	-1.207930053	-1.107018207
6	-0.506924177	0.748211648	1.762482687	0.211507596	1.368987826
7	1.573996573	-0.796593995	-0.824217424	1.561715851	-1.607942726

Criterion Based on Final Seeds =	0.3979

the Figure 45.1 displays the table of initial seeds used for each variable and cluster. The first line in the figure displays the option settings for REPLACE, RADIUS, MAXCLUSTERS, and MAXITER. These options, with the exception of MAXCLUSTERS and MAXITER, are set at their respective default values (REPLACE=FULL, RADIUS=0). Both the MAXCLUSTERS= and MAXITER= options are set in the PROC FASTCLUS statement.

Next, PROC FASTCLUS produces a table of summary statistics for the clusters. Figure 45.2 displays the number of observations in the cluster (frequency) and the root mean squared standard deviation. The next two columns display the largest Euclidean distance from the cluster seed to any observation within the cluster and the number of the nearest cluster.

The last column of the table displays the distance between the centroid of the nearest cluster and the centroid of the current cluster. A centroid is the point having coordinates that are the means of all the observations in the cluster.

Figure 45.2 Cluster Summary Table from the FASTCLUS Procedure

Cluster Summary						
Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	17	0.5064	1.7781		4	2.5106
2	19	0.3696	1.5007		4	1.5510
3	13	0.3803	1.7135		1	2.6704
4	13	0.4161	1.3976		7	1.4266
5	11	0.2466	0.6966		6	1.7301
6	34	0.3563	1.5443		5	1.7301
7	50	0.4447	2.3915		4	1.4266

Figure 45.3 displays the table of statistics for the variables. The table lists for each variable the total standard deviation, the pooled within-cluster standard deviation and the R-square value for predicting the variable from the cluster. The ratio of between-cluster variance to within-cluster variance (R^2 to $1 - R^2$) appears in the last column.

Figure 45.3 Statistics for Variables Used in the FASTCLUS Procedure

Statistics for Variables				
Variable	Total STD	Within STD	R-Square	RSQ/(1-RSQ)
Length1	1.00000	0.31428	0.905030	9.529606
logLengthRatio	1.00000	0.39276	0.851676	5.741989
Height	1.00000	0.20917	0.957929	22.769295
Width	1.00000	0.55558	0.703200	2.369270
Weight3	1.00000	0.47251	0.785323	3.658162
OVER-ALL	1.00000	0.40712	0.840631	5.274764

Pseudo F Statistic =	131.87

Approximate Expected Over-All R-Squared =	0.57420

Cubic Clustering Criterion =	37.808

The pseudo F statistic, approximate expected overall R square, and cubic clustering criterion (CCC) are listed at the bottom of the figure. You can compare values of these statistics by running PROC FASTCLUS with different values for the MAXCLUSTERS= option. The R square and CCC values are not valid for correlated variables.

Values of the cubic clustering criterion greater than 2 or 3 indicate good clusters. Values between 0 and 2 indicate potential clusters, but they should be taken with caution; large negative values can indicate outliers.

PROC FASTCLUS next produces the within-cluster means and standard deviations of the variables, displayed in Figure 45.4.

Figure 45.4 Cluster Means and Standard Deviations from the FASTCLUS Procedure

Cluster Means					
Cluster	**Length1**	**logLengthRatio**	**Height**	**Width**	**Weight3**
1	1.747808245	-0.868605685	-1.327226832	-1.128760946	0.806373599
2	-0.405231510	-0.979113021	-0.281064162	1.463094486	1.060450065
3	2.006796315	-0.652725165	-1.053213440	-1.224020795	-1.826752838
4	-0.136820952	-1.039312574	-0.446429482	0.162596336	0.278560318
5	-0.850130601	0.550190242	1.245156076	-0.836585750	-0.567022647
6	-0.843912827	1.522291347	1.511408739	-0.380323563	0.763114370
7	-0.165570970	-0.048881276	-0.353723615	0.546442064	-0.668780782

Cluster Standard Deviations					
Cluster	**Length1**	**logLengthRatio**	**Height**	**Width**	**Weight3**
1	0.3418476428	0.3544065543	0.1666302451	0.6172880027	0.7944227150
2	0.3129902863	0.3592350778	0.1369052680	0.5467406493	0.3720119097
3	0.2962504486	0.1740941675	0.1736086707	0.7528475622	0.0905232968
4	0.3254364840	0.2836681149	0.1884592934	0.4543390702	0.6612055341
5	0.1781837609	0.0745984121	0.2056932592	0.2784540794	0.3832002850
6	0.2273744242	0.3385584051	0.2046010964	0.5143496067	0.4025849044
7	0.3734733622	0.5275768119	0.2551130680	0.5721303628	0.4223181710

It is useful to study further the clusters calculated by the FASTCLUS procedure. One method is to look at a frequency tabulation of the clusters with other classification variables. The following statements invoke the FREQ procedure to crosstabulate the empirical clusters with the variable Species:

```
proc freq data=Clust;
   tables Species*Cluster;
run;
```

Figure 45.5 displays the marked division between clusters.

Figure 45.5 Frequency Table of Cluster versus Species

Fish Measurement Data

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Species by CLUSTER CLUSTER(Cluster)							
Species	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Total**
Bream	0	0	0	0	0	34	0	34
	0.00	0.00	0.00	0.00	0.00	21.66	0.00	21.66
	0.00	0.00	0.00	0.00	0.00	100.00	0.00	
	0.00	0.00	0.00	0.00	0.00	100.00	0.00	
Roach	0	0	0	0	0	0	19	19
	0.00	0.00	0.00	0.00	0.00	0.00	12.10	12.10
	0.00	0.00	0.00	0.00	0.00	0.00	100.00	
	0.00	0.00	0.00	0.00	0.00	0.00	38.00	
Whitefish	0	2	0	1	0	0	3	6
	0.00	1.27	0.00	0.64	0.00	0.00	1.91	3.82
	0.00	33.33	0.00	16.67	0.00	0.00	50.00	
	0.00	10.53	0.00	7.69	0.00	0.00	6.00	
Parkki	0	0	0	0	11	0	0	11
	0.00	0.00	0.00	0.00	7.01	0.00	0.00	7.01
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	
Perch	0	17	0	12	0	0	27	56
	0.00	10.83	0.00	7.64	0.00	0.00	17.20	35.67
	0.00	30.36	0.00	21.43	0.00	0.00	48.21	
	0.00	89.47	0.00	92.31	0.00	0.00	54.00	
Pike	17	0	0	0	0	0	0	17
	10.83	0.00	0.00	0.00	0.00	0.00	0.00	10.83
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
Smelt	0	0	13	0	0	0	1	14
	0.00	0.00	8.28	0.00	0.00	0.00	0.64	8.92
	0.00	0.00	92.86	0.00	0.00	0.00	7.14	
	0.00	0.00	100.00	0.00	0.00	0.00	2.00	
Total	17	19	13	13	11	34	50	157
	10.83	12.10	8.28	8.28	7.01	21.66	31.85	100.00

For cases in which you have three or more clusters, you can use the CANDISC and SGPLOT procedures to obtain a graphical check on the distribution of the clusters. In the following statements, the CANDISC and SGPLOT procedures are used to compute canonical variables and plot the clusters:

```
proc candisc data=Clust out=Can noprint;
   class Cluster;
   var Length1 logLengthRatio Height Width Weight3;
run;

proc sgplot data=Can;
   scatter y=Can2 x=Can1 / group=Cluster;
run;
```

First, the CANDISC procedure is invoked to perform a canonical discriminant analysis by using the data set Clust and creating the output SAS data set Can. The NOPRINT option suppresses display of the output. The CLASS statement specifies the variable Cluster to define groups for the analysis. The VAR statement specifies the variables used in the analysis.

Next, the SGPLOT procedure plots the two canonical variables from PROC CANDISC, Can1 and Can2. The SCATTER statement specifies the variable Cluster as the group identification variable. The resulting plot (Figure 45.6) illustrates the spatial separation of the clusters calculated in the FASTCLUS procedure.

Figure 45.6 Plot of Canonical Variables and Cluster Value

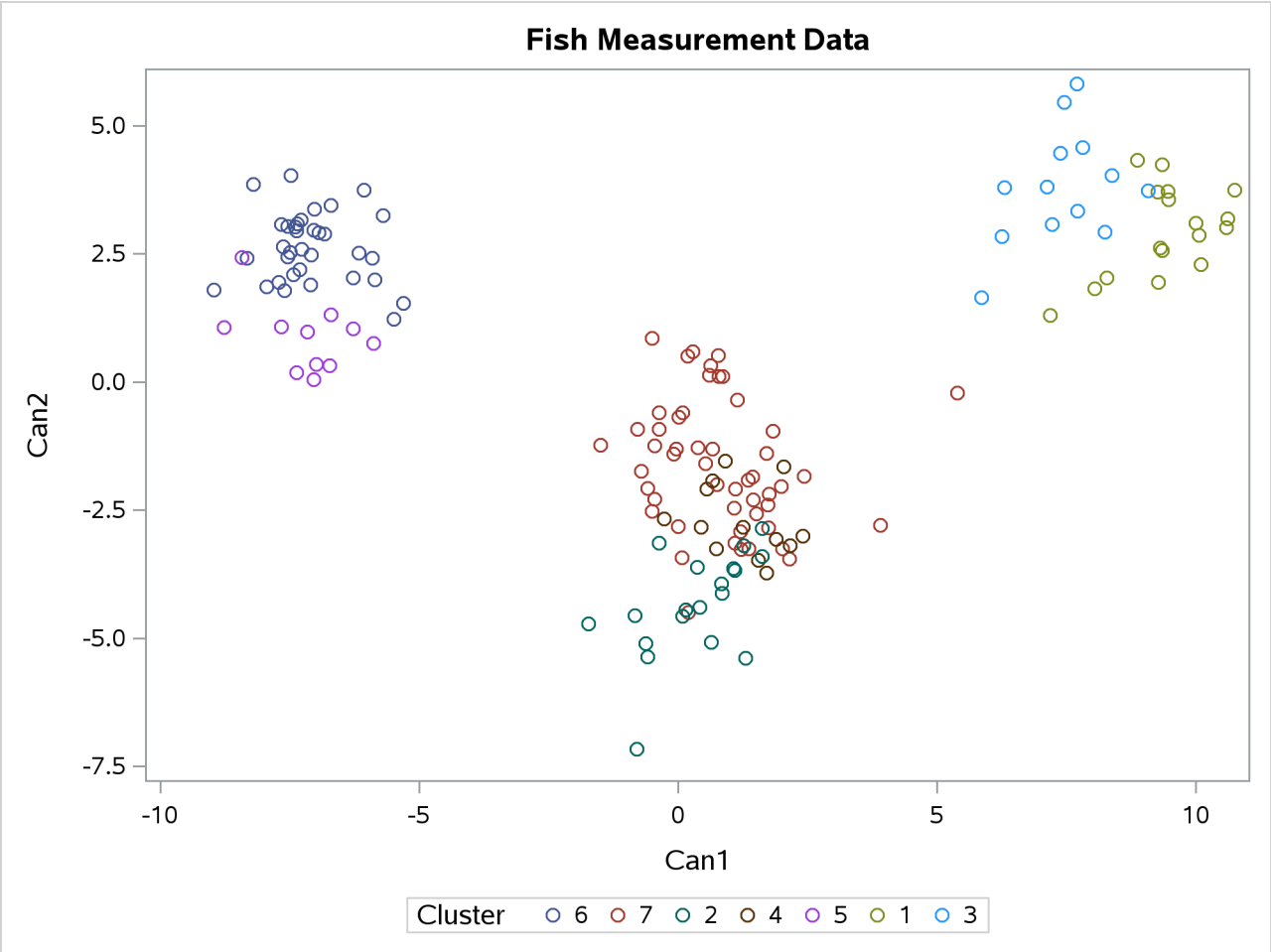

Syntax: FASTCLUS Procedure

The following statements are available in the FASTCLUS procedure:

PROC FASTCLUS <**MAXCLUSTERS=***n*> <**RADIUS=***t*> <*options*> **;**
  VAR *variables* **;**
  ID *variables* **;**
  FREQ *variable* **;**
  WEIGHT *variable* **;**
  BY *variables* **;**

Usually you need only the VAR statement in addition to the PROC FASTCLUS statement. The BY, FREQ, ID, VAR, and WEIGHT statements are described in alphabetical order after the PROC FASTCLUS statement.

PROC FASTCLUS Statement

PROC FASTCLUS <**MAXCLUSTERS=***n*> <**RADIUS=***t*> <*options*> **;**

The PROC FASTCLUS statement invokes the FASTCLUS procedure. You must specify the MAXCLUSTERS= option or RADIUS= option or both in the PROC FASTCLUS statement.

MAXCLUSTERS=*n*

MAXC=*n*

specifies the maximum number of clusters permitted. If you omit the MAXCLUSTERS= option, a value of 100 is assumed.

RADIUS=*t*

R=*t*

establishes the minimum distance criterion for selecting new seeds. No observation is considered as a new seed unless its minimum distance to previous seeds exceeds the value given by the RADIUS= option. The default value is 0. If you specify the REPLACE=RANDOM option, the RADIUS= option is ignored.

You can specify the following options in the PROC FASTCLUS statement. Table 45.1 summarizes the options available in the PROC FASTCLUS statement.

Table 45.1 PROC FASTCLUS Statement Options

Option	Description
Specify Input and Output Data Sets	
DATA=	Specifies input data set
INSTAT=	Specifies input SAS data set previously created by the OUTSTAT= option
SEED=	Specifies input SAS data set for selecting initial cluster seeds
VARDEF=	Specifies divisor for variances

Table 45.1 *continued*

Option	Description
Output Data Processing	
CLUSTER=	Specifies name for cluster membership variable in OUTSEED= and OUT= data sets
CLUSTERLABEL=	Specifies label for cluster membership variable in OUTSEED= and OUT= data sets
OUT=	Specifies output SAS data set containing original data and cluster assignments
OUTITER	Specifies writing to OUTSEED= data set on every iteration
OUTSEED= or MEAN=	Specifies output SAS data set containing cluster centers
OUTSTAT=	Specifies output SAS data set containing statistics
Initial Clusters	
DRIFT	Permits cluster to seeds to drift during initialization
MAXCLUSTERS=	Specifies maximum number of clusters
RADIUS=	Specifies minimum distance for selecting new seeds
RANDOM=	Specifies seed to initializes pseudo-random number generator
REPLACE=	Specifies seed replacement method
Clustering Methods	
CONVERGE=	Specifies convergence criterion
DELETE=	Deletes cluster seeds with few observations
LEAST=	Optimizes an L_p criterion, where $1 \leq p \leq \infty$
MAXITER=	Specifies maximum number of iterations
STRICT	Prevents an observation from being assigned to a cluster if its distance to the nearest cluster seed is large
Arcane Algorithmic Options	
BINS=	Specifies number of bins used for computing medians for LEAST=1
HC=	Specifies criterion for updating the homotopy parameter
HP=	Specifies initial value of the homotopy parameter
IRLS	Uses an iteratively reweighted least squares method instead of the modified Ekblom-Newton method for $1 < p < 2$
Missing Values	
IMPUTE	Imputes missing values after final cluster assignment
NOMISS	Excludes observations with missing values
Control Displayed Output	
DISTANCE	Displays distances between cluster centers
LIST	Displays cluster assignments for all observations
NOPRINT	Suppresses displayed output
SHORT	Suppresses display of large matrices
SUMMARY	Suppresses display of all results except for the cluster summary
VARIABLESAREUNCORRELATED	Suppresses warning in output

The following list provides details on these options. The list is in alphabetical order.

BINS=*n*

specifies the number of bins used in the bin-sort algorithm for computing medians for LEAST=1. By default, PROC FASTCLUS uses from 10 to 100 bins, depending on the amount of memory available. Larger values use more memory and make each iteration somewhat slower, but they can reduce the number of iterations. Smaller values have the opposite effect. The minimum value of *n* is 5.

CLUSTER=*name*

specifies a name for the variable in the OUTSEED= and OUT= data sets that indicates cluster membership. The default name for this variable is CLUSTER.

CLUSTERLABEL=*name*

specifies a label for the variable CLUSTER in the OUTSEED= and OUT= data sets. By default this variable has no label.

CONVERGE=*c*

CONV=*c*

specifies the convergence criterion. Any nonnegative value is permitted. The default value is 0.0001 for all values of *p* if LEAST=*p* is explicitly specified; otherwise, the default value is 0.02. Iterations stop when the maximum relative change in the cluster seeds is less than or equal to the convergence criterion and additional conditions on the homotopy parameter, if any, are satisfied (see the HP= option). The relative change in a cluster seed is the distance between the old seed and the new seed divided by a scaling factor. If you do not specify the LEAST= option, the scaling factor is the minimum distance between the initial seeds. If you specify the LEAST= option, the scaling factor is an L_1 scale estimate and is recomputed on each iteration. Specify the CONVERGE= option only if you specify a MAXITER= value greater than 1.

DATA=*SAS-data-set*

specifies the input data set containing observations to be clustered. If you omit the DATA= option, the most recently created SAS data set is used. The data must be coordinates, not distances, similarities, or correlations.

DELETE=*n*

deletes cluster seeds to which *n* or fewer observations are assigned. Deletion occurs after processing for the DRIFT option is completed and after each iteration specified by the MAXITER= option. Cluster seeds are not deleted after the final assignment of observations to clusters, so in rare cases a final cluster might not have more than *n* members. The DELETE= option is ineffective if you specify MAXITER=0 and do not specify the DRIFT option. By default, no cluster seeds are deleted.

DISTANCE | DIST

computes distances between the cluster means.

DRIFT

executes the second of the four steps described in the section "Background" on page 2971. After initial seed selection, each observation is assigned to the cluster with the nearest seed. After an observation is processed, the seed of the cluster to which it is assigned is recalculated as the mean of the observations currently assigned to the cluster. Thus, the cluster seeds drift about rather than remaining fixed for the duration of the pass.

HC=*c*

HP=*p_1* **<** *p_2* **>**

pertains to the homotopy parameter for LEAST=*p*, where $1 < p < 2$. You should specify these options only if you encounter convergence problems when you use the default values.

For $1 < p < 2$, PROC FASTCLUS tries to optimize a perturbed variant of the L_p clustering criterion (Gonin and Money 1989, pp. 5–6).

When the homotopy parameter is 0, the optimization criterion is equivalent to the clustering criterion. For a large homotopy parameter, the optimization criterion approaches the least squares criterion and is therefore easy to optimize. Beginning with a large homotopy parameter, PROC FASTCLUS gradually decreases it by a factor in the range [0.01,0.5] over the course of the iterations. When both the homotopy parameter and the convergence measure are sufficiently small, the optimization process is declared to have converged.

If the initial homotopy parameter is too large or if it is decreased too slowly, the optimization can require many iterations. If the initial homotopy parameter is too small or if it is decreased too quickly, convergence to a local optimum is likely. The following list gives details on setting the homotopy parameter.

HC=*c* specifies the criterion for updating the homotopy parameter. The homotopy parameter is updated when the maximum relative change in the cluster seeds is less than or equal to *c*. The default is the minimum of 0.01 and 100 times the value of the CONVERGE= option.

HP=*p_1* specifies p_1 as the initial value of the homotopy parameter. The default is 0.05 if the modified Ekblom-Newton method is used; otherwise, it is 0.25.

HP=*p_1 p_2* also specifies p_2 as the minimum value for the homotopy parameter, which must be reached for convergence. The default is the minimum of p_1 and 0.01 times the value of the CONVERGE= option.

IMPUTE

requests imputation of missing values after the final assignment of observations to clusters. If an observation that is assigned (or would have been assigned) to a cluster has a missing value for variables used in the cluster analysis, the missing value is replaced by the corresponding value in the cluster seed to which the observation is assigned (or would have been assigned). If the observation cannot be assigned to a cluster, missing value replacement depends on whether or not the NOMISS option is specified. If NOMISS is not specified, missing values are replaced by the mean of all observations in the DATA= data set having a value for that variable. If NOMISS is specified, missing values are replace by the mean of only observations used in the analysis. (A weighted mean is used if a variable is specified in the WEIGHT statement.) For information about cluster assignment see the section "OUT= Data Set" on page 2989. If you specify the IMPUTE option, the imputed values are not used in computing cluster statistics.

If you also request an OUT= data set, it contains the imputed values.

INSTAT=*SAS-data-set*

reads a SAS data set previously created with the FASTCLUS procedure by using the OUTSTAT= option. If you specify the INSTAT= option, no clustering iterations are performed and no output is displayed. Only cluster assignment and imputation are performed as an OUT= data set is created.

IRLS

causes PROC FASTCLUS to use an iteratively reweighted least squares method instead of the modified Ekblom-Newton method. If you specify the IRLS option, you must also specify LEAST=*p*, where $1 < p < 2$. Use the IRLS option only if you encounter convergence problems with the default method.

LEAST=*p* **| MAX**

L=*p* **| MAX**

causes PROC FASTCLUS to optimize an L_p criterion, where $1 \leq p \leq \infty$ (Spath 1985, pp. 62–63). Infinity is indicated by LEAST=MAX. The value of this clustering criterion is displayed in the iteration history.

If you do not specify the LEAST= option, PROC FASTCLUS uses the least squares (L_2) criterion. However, the default number of iterations is only 1 if you omit the LEAST= option, so the optimization of the criterion is generally not completed. If you specify the LEAST= option, the maximum number of iterations is increased to permit the optimization process a chance to converge. See the MAXITER= *n* option for details.

Specifying the LEAST= option also changes the default convergence criterion from 0.02 to 0.0001. See the CONVERGE= *c* for details.

When LEAST=2, PROC FASTCLUS tries to minimize the root mean squared difference between the data and the corresponding cluster means.

When LEAST=1, PROC FASTCLUS tries to minimize the mean absolute difference between the data and the corresponding cluster medians.

When LEAST=MAX, PROC FASTCLUS tries to minimize the maximum absolute difference between the data and the corresponding cluster midranges.

For general values of *p*, PROC FASTCLUS tries to minimize the *p*th root of the mean of the *p*th powers of the absolute differences between the data and the corresponding cluster seeds. Large values of p may cause floating-point underflow or overflow.

The divisor in the clustering criterion is either the number of nonmissing data used in the analysis or, if there is a WEIGHT statement, the sum of the weights corresponding to all the nonmissing data used in the analysis (that is, an observation with *n* nonmissing data contributes *n* times the observation weight to the divisor). The divisor is not adjusted for degrees of freedom.

The method for updating cluster seeds during iteration depends on the LEAST= option, as follows (Gonin and Money 1989).

LEAST=*p*	Algorithm for Computing Cluster Seeds
$p = 1$	Bin sort for median
$1 < p < 2$	Modified Merle-Spath if you specify IRLS; otherwise modified Ekblom-Newton
$p = 2$	Arithmetic mean
$2 < p < \infty$	Newton
$p = \infty$	Midrange

During the final pass, a modified Merle-Spath step is taken to compute the cluster centers for $1 \leq p < 2$ or $2 < p < \infty$.

If you specify the LEAST=*p* option with a value other than 2, PROC FASTCLUS computes pooled scale estimates analogous to the root mean squared standard deviation but based on *p*th power deviations instead of squared deviations.

LEAST=*p*	Scale Estimate
$p = 1$	Mean absolute deviation
$1 < p < \infty$	Root mean *p*th-power absolute deviation
$p = \infty$	Maximum absolute deviation

The divisors for computing the mean absolute deviation or the root mean *p*th-power absolute deviation are adjusted for degrees of freedom just like the divisors for computing standard deviations. This adjustment can be suppressed by the VARDEF= option.

LIST

lists all observations, giving the value of the ID variable (if any), the number of the cluster to which the observation is assigned, and the distance between the observation and the final cluster seed.

MAXITER=*n*

specifies the maximum number of iterations for recomputing cluster seeds. When the value of the MAXITER= option is greater than zero, PROC FASTCLUS executes the third of the four steps described in the section "Background" on page 2971. In each iteration, each observation is assigned to the nearest seed, and the seeds are recomputed as the means of the clusters.

The default value of the MAXITER= option depends on the LEAST=*p* option.

LEAST=*p*	MAXITER=
Not specified	1
$p = 1$	20
$1 < p < 1.5$	50
$1.5 \leq p < 2$	20
$p = 2$	10
$2 < p \leq \infty$	20

MEAN=*SAS-data-set*

creates an output data set to contain the cluster means and other statistics for each cluster. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

NOMISS

excludes observations with missing values from the analysis. However, if you also specify the IMPUTE option, observations with missing values are included in the final cluster assignments.

NOPRINT

suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*

creates an output data set to contain all the original data, plus the new variables CLUSTER and DISTANCE. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTITER

outputs information from the iteration history to the OUTSEED= data set, including the cluster seeds at each iteration.

OUTSEED=*SAS-data-set*

OUTS=*SAS-data-set*

is another name for the MEAN= data set, provided because the data set can contain location estimates other than means. The MEAN= option is still accepted.

OUTSTAT=*SAS-data-set*

creates an output data set to contain various statistics, especially those not included in the OUTSEED= data set. Unlike the OUTSEED= data set, the OUTSTAT= data set is not suitable for use as a SEED= data set in a subsequent PROC FASTCLUS step.

RANDOM=*n*

specifies a positive integer as a starting value for the pseudo-random number generator for use with REPLACE=RANDOM. If you do not specify the RANDOM= option, the time of day is used to initialize the pseudo-random number sequence.

REPLACE=FULL | PART | NONE | RANDOM

specifies how seed replacement is performed, as follows:

FULL requests default seed replacement as described in the section "Background" on page 2971.

PART requests seed replacement only when the distance between the observation and the closest seed is greater than the minimum distance between seeds.

NONE suppresses seed replacement.

RANDOM selects a simple pseudo-random sample of complete observations as initial cluster seeds.

SEED=*SAS-data-set*

specifies an input data set from which initial cluster seeds are to be selected. If you do not specify the SEED= option, initial seeds are selected from the DATA= data set. The SEED= data set must contain the same variables that are used in the data analysis.

SHORT
suppresses the display of the initial cluster seeds, cluster means, and standard deviations.

STRICT
STRICT=*s*
prevents an observation from being assigned to a cluster if its distance to the nearest cluster seed exceeds the value of the STRICT= option. If you specify the STRICT option without a numeric value, you must also specify the RADIUS= option, and its value is used instead. In the OUT= data set, observations that are not assigned due to the STRICT= option are given a negative cluster number, the absolute value of which indicates the cluster with the nearest seed.

SUMMARY
suppresses the display of the initial cluster seeds, statistics for variables, cluster means, and standard deviations.

VARDEF=DF | N | WDF | WEIGHT | WGT
specifies the divisor to be used in the calculation of variances and covariances. The default value is VARDEF=DF. The possible values of the VARDEF= option and associated divisors are as follows.

Value	Description	Divisor
DF	Error degrees of freedom	$n - c$
N	Number of observations	n
WDF	Sum of weights DF	$(\sum_i w_i) - c$
WEIGHT | WGT	Sum of weights	$\sum_i w_i$

In the preceding definitions, c represents the number of clusters.

VARIABLESAREUNCORRELATED
suppresses the warning, displayed in the listing when there are two or more VAR variables, concerning the validity of the Approximate Expected Over-All R-Squared and the Cubic Clustering Criterion when the variables used in clustering are correlated. Note that the FASTCLUS procedure does not compute correlations; the warning is for a potential problem.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC FASTCLUS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the FASTCLUS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

If you specify the SEED= option and the SEED= data set does not contain any of the BY variables, then the entire SEED= data set is used to obtain initial cluster seeds for each BY group in the DATA= data set.

If the SEED= data set contains some but not all of the BY variables, or if some BY variables do not have the same type or length in the SEED= data set as in the DATA= data set, then PROC FASTCLUS displays an error message and stops.

If all the BY variables appear in the SEED= data set with the same type and length as in the DATA= data set, then each BY group in the SEED= data set is used to obtain initial cluster seeds for the corresponding BY group in the DATA= data set. All BY groups in the DATA= data set must also appear in the SEED= data set. The BY groups in the SEED= data set must be in the same order as in the DATA= data set. If you specify the NOTSORTED option in the BY statement, both data sets must contain exactly the same BY groups in the same order. If you do not specify NOTSORTED, some BY groups can appear in the SEED= data set but not in the DATA= data set; such BY groups are not used in the analysis.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If a variable in the data set represents the frequency of occurrence for the other values in the observation, include the variable's name in a FREQ statement. The procedure then treats the data set as if each observation appears *n* times, where *n* is the value of the FREQ variable for the observation.

If the value of the FREQ variable is missing or less than or equal to zero, the observation is not used in the analysis. The exact values of the FREQ variable are used in computations: frequency values are not truncated to integers. The total number of observations is considered to be equal to the sum of the FREQ variable when the procedure determines degrees of freedom for significance probabilities.

The WEIGHT and FREQ statements have a similar effect, except in determining the number of observations for significance tests.

ID Statement

ID *variable* **;**

The ID variable, which can be character or numeric, identifies observations on the output when you specify the LIST option.

VAR Statement

VAR *variables* **;**

The VAR statement lists the numeric variables to be used in the cluster analysis. If you omit the VAR statement, all numeric variables not listed in other statements are used.

WEIGHT Statement

WEIGHT *variable* **;**

The values of the WEIGHT variable are used to compute weighted cluster means. The WEIGHT and FREQ statements have a similar effect, except the WEIGHT statement does not alter the degrees of freedom or the number of observations. The WEIGHT variable can take nonintegral values. An observation is used in the analysis only if the value of the WEIGHT variable is greater than zero.

Details: FASTCLUS Procedure

Updates in the FASTCLUS Procedure

Some FASTCLUS procedure options and statements have changed from previous versions. The differences are as follows:

- Values of the FREQ variable are no longer truncated to integers. Noninteger variables specified in the FREQ statement produce results different from those in previous releases.
- The IMPUTE option produces different cluster standard deviations and related statistics. When you specify the IMPUTE option, imputed values are no longer used in computing cluster statistics. This change causes the cluster standard deviations and other statistics computed from the standard deviations to be different from those in previous releases.
- The INSTAT= option reads a SAS data set previously created with the FASTCLUS procedure by using the OUTSTAT= option. If you specify the INSTAT= option, no clustering iterations are performed and no output is produced. Only cluster assignment and imputation are performed as an OUT= data set is created.

- The OUTSTAT= data set contains additional information used for imputation. _TYPE_=SEED corresponds to values that are cluster seeds. Observations previously designated _TYPE_='SCALE' are now _TYPE_='DISPERSION'.

Missing Values

Observations with all missing values are excluded from the analysis. If you specify the NOMISS option, observations with any missing values are excluded. Observations with missing values cannot be cluster seeds.

The distance between an observation with missing values and a cluster seed is obtained by computing the squared distance based on the nonmissing values, multiplying by the ratio of the number of variables, n, to the number of variables having nonmissing values, m, and taking the square root:

$$\sqrt{\left(\frac{n}{m}\right)\sum(x_i - s_i)^2}$$

where

$$\begin{aligned} n &= \text{number of variables} \\ m &= \text{number of variables with nonmissing values} \\ x_i &= \text{value of the } i\text{th variable for the observation} \\ s_i &= \text{value of the } i\text{th variable for the seed} \end{aligned}$$

If you specify the LEAST=p option with a power p other than 2 (the default), the distance is computed using

$$\left(\left(\frac{n}{m}\right)\sum(x_i - s_i)^p\right)^{\frac{1}{p}}$$

The summation is taken over variables with nonmissing values.

The IMPUTE option fills in missing values in the OUT= output data set.

Output Data Sets

OUT= Data Set

The OUT= data set contains the following:

- the original variables
- a new variable indicating the cluster assignment status of each observation. The value will be less than the permitted number of clusters (see the MAXCLUSTERS= option) if the procedure detects fewer clusters than the maximum. A positive value indicates the cluster to which the observation was assigned. A negative value indicates that the observation was not assigned to a cluster (see the STRICT option), and the absolute value indicates the cluster to which the observation would have been assigned. If the value is missing, the observation cannot be assigned to any cluster. You can specify the variable name with the CLUSTER= option. The default name is CLUSTER.

- a new variable, DISTANCE, giving the distance from the observation to its cluster seed

If you specify the IMPUTE option, the OUT= data set also contains a new variable, _IMPUTE_, giving the number of imputed values in each observation.

OUTSEED= Data Set

The OUTSEED= data set contains one observation for each cluster. The variables are as follows:

- the BY variables, if any
- a new variable giving the cluster number. You can specify the variable name with the CLUSTER= option. The default name is CLUSTER.
- either the FREQ variable or a new variable called _FREQ_ giving the number of observations in the cluster
- the WEIGHT variable, if any
- a new variable, _RMSSTD_, giving the root mean squared standard deviation for the cluster. See Chapter 40, "The CLUSTER Procedure," for details.
- a new variable, _RADIUS_, giving the maximum distance between any observation in the cluster and the cluster seed
- a new variable, _GAP_, containing the distance between the current cluster mean and the nearest other cluster mean. The value is the centroid distance given in the output.
- a new variable, _NEAR_, specifying the cluster number of the nearest cluster
- the VAR variables giving the cluster means

If you specify the LEAST=*p* option with a value other than 2, the _RMSSTD_ variable is replaced by the _SCALE_ variable, which contains the pooled scale estimate analogous to the root mean squared standard deviation but based on *p*th-power deviations instead of squared deviations:

LEAST=1	mean absolute deviation
LEAST=*p*	root mean *p*th-power absolute deviation
LEAST=MAX	maximum absolute deviation

If you specify the OUTITER option, there is one set of observations in the OUTSEED= data set for each pass through the data set (that is, one set for initial seeds, one for each iteration, and one for the final clusters). Also, several additional variables appear:

ITER	is the iteration number. For the initial seeds, the value is 0. For the final cluster means or centers, the _ITER_ variable is one greater than the last iteration reported in the iteration history.
CRIT	is the clustering criterion as described under the LEAST= option.

CHANGE is the maximum over clusters of the relative change in the cluster seed from the previous iteration. The relative change in a cluster seed is the distance between the old seed and the new seed divided by a scaling factor. If you do not specify the LEAST= option, the scaling factor is the minimum distance between the initial seeds. If you specify the LEAST= option, the scaling factor is an L_1 scale estimate and is recomputed on each iteration.

HOMPAR is the value of the homotopy parameter. This variable appears only for LEAST=p with $1 < p < 2$.

BINSIZ is the maximum bin size used for estimating medians. This variable appears only for LEAST=1.

If you specify the OUTITER option, the variables _SCALE_ or _RMSSTD_, _RADIUS_, _NEAR_, and _GAP_ have missing values except for the last pass.

You can use the OUTSEED= data set as a SEED= input data set for a subsequent analysis.

OUTSTAT= Data Set

The variables in the OUTSTAT= data set are as follows:

- BY variables, if any
- a new character variable, _TYPE_, specifying the type of statistic given by other variables (see Table 45.2 and Table 45.3)
- a new numeric variable giving the cluster number. You can specify the variable name with the CLUSTER= option. The default name is CLUSTER.
- a new numeric variable, OVER_ALL, containing statistics that apply over all of the VAR variables
- the VAR variables giving statistics for particular variables

The values of _TYPE_ for all LEAST= options are given in Table 45.2.

Table 45.2 _TYPE_

TYPE	Contents of VAR Variables	Contents of OVER_ALL
INITIAL	Initial seeds	Missing
CRITERION	Missing	Optimization criterion (see the LEAST= option); this value is displayed just before the "Cluster Summary" table.
CENTER	Cluster centers (see the LEAST= option)	Missing
SEED	Cluster seeds: additional information used for imputation	

Table 45.2 *continued*

TYPE	Contents of VAR variables	Contents of OVER_ALL
DISPERSION	Dispersion estimates for each cluster (see the LEAST= option); these values are displayed in a separate row with title depending on the LEAST= option	Dispersion estimates pooled over variables (see the LEAST= option); these values are displayed in the "Cluster Summary" table with label depending on the LEAST= option.
FREQ	Frequency of each cluster omitting observations with missing values for the VAR variable; these values are not displayed	Frequency of each cluster based on all observations with any nonmissing value; these values are displayed in the "Cluster Summary" table.
WEIGHT	Sum of weights for each cluster omitting observations with missing values for the VAR variable; these values are not displayed	Sum of weights for each cluster based on all observations with any nonmissing value; these values are displayed in the "Cluster Summary" table.

Observations with _TYPE_='WEIGHT' are included only if you specify the WEIGHT statement.

The _TYPE_ values included only for least squares clustering are given Table 45.3. Least squares clustering is obtained by omitting the LEAST= option or by specifying LEAST=2.

Table 45.3 _TYPE_

TYPE	Contents of VAR Variables	Contents of OVER_ALL
MEAN	Mean for the total sample; this is not displayed	Missing
STD	Standard deviation for the total sample; labeled "Total STD" in the output	Standard deviation pooled over all the VAR variables; labeled "Total STD" in the output
WITHIN_STD	Pooled within-cluster standard deviation	Within cluster standard deviation pooled over clusters and all the VAR variables
RSQ	R square for predicting the variable from the clusters; labeled "R-Squared" in the output	R square pooled over all the VAR variables; labeled "R-Squared" in the output

Table 45.3 *continued*

TYPE	Contents of VAR variables	Contents of OVER_ALL
RSQ_RATIO	$\frac{R^2}{1-R^2}$; labeled "RSQ/(1-RSQ)" in the output	$\frac{R^2}{1-R^2}$; labeled "RSQ/(1-RSQ)" in the output
PSEUDO_F	Missing	Pseudo F statistic
ESRQ	Missing	Approximate expected value of R square under the null hypothesis of a single uniform cluster
CCC	Missing	Cubic clustering criterion

Computational Resources

Let

$$\begin{aligned} n &= \text{number of observations} \\ v &= \text{number of variables} \\ c &= \text{number of clusters} \\ p &= \text{number of passes over the data set} \end{aligned}$$

Memory

The memory required is approximately $4(19v + 12cv + 10c + 2\max(c+1, v))$ bytes.

If you request the DISTANCE option, an additional $4c(c+1)$ bytes of space is needed.

Time

The overall time required by PROC FASTCLUS is roughly proportional to $nvcp$ if c is small with respect to n.

Initial seed selection requires one pass over the data set. If the observations are in random order, the time required is roughly proportional to

$$nvc + vc^2$$

unless you specify REPLACE=NONE. In that case, a complete pass might not be necessary, and the time is roughly proportional to mvc, where $c \le m \le n$.

The DRIFT option, each iteration, and the final assignment of cluster seeds each require one pass, with time for each pass roughly proportional to nvc.

For greatest efficiency, you should list the variables in the VAR statement in order of decreasing variance.

Using PROC FASTCLUS

Before using PROC FASTCLUS, decide whether your variables should be standardized in some way, since variables with large variances tend to have more effect on the resulting clusters than those with small variances. If all variables are measured in the same units, standardization might not be necessary. Otherwise, some form of standardization is strongly recommended. The STDIZE procedure provides a variety of standardization methods, including robust scale estimators (for detailed information, see Chapter 115, "The STDIZE Procedure").

The FACTOR or PRINCOMP procedure can compute standardized principal component scores. The ACECLUS procedure can transform the variables according to an estimated within-cluster covariance matrix.

Nonlinear transformations of the variables can change the number of population clusters and should therefore be approached with caution. For most applications, the variables should be transformed so that equal differences are of equal practical importance. An interval scale of measurement is required. Ordinal or ranked data are generally not appropriate.

PROC FASTCLUS produces relatively little output. In most cases you should create an output data set and use another procedure such as PRINT, SGPLOT, MEANS, DISCRIM, or CANDISC to study the clusters. It is usually desirable to try several values of the MAXCLUSTERS= option. Macros are useful for running PROC FASTCLUS repeatedly with other procedures.

A simple application of PROC FASTCLUS with two variables to examine the 2- and 3-cluster solutions can proceed as follows:

```
proc stdize method=std out=stan;
   var v1 v2;
run;

proc fastclus data=stan out=clust maxclusters=2;
   var v1 v2;
run;

proc sgplot;
   scatter y=v2 x=v1 / markerchar=cluster;
run;

proc fastclus data=stan out=clust maxclusters=3;
   var v1 v2;
run;

proc sgplot;
   scatter y=v2 x=v1 / markerchar=cluster;
run;
```

If you have more than two variables, you can use the CANDISC procedure to compute canonical variables for plotting the clusters. For example:

```
proc stdize method=std out=stan;
   var v1-v10;
run;

proc fastclus data=stan out=clust maxclusters=3;
   var v1-v10;
run;

proc candisc out=can;
   var v1-v10;
   class cluster;
run;

proc sgplot;
   scatter y=can2 x=can1 / markerchar=cluster;
run;
```

If the data set is not too large, it might also be helpful to use the following to list the clusters:

```
proc sort;
   by cluster distance;
run;

proc print;
   by cluster;
run;
```

By examining the values of DISTANCE, you can determine if any observations are unusually far from their cluster seeds.

It is often advisable, especially if the data set is large or contains outliers, to make a preliminary PROC FASTCLUS run with a large number of clusters, perhaps 20 to 100. Use MAXITER=0 and OUTSEED=*SAS-data-set*. You can save time on subsequent runs if you select cluster seeds from this output data set by using the SEED= option.

You should check the preliminary clusters for outliers, which often appear as clusters with only one member. Use a DATA step to delete outliers from the data set created by the OUTSEED= option before using it as a SEED= data set in later runs. If there are severe outliers, you should specify the STRICT option in the subsequent PROC FASTCLUS runs to prevent the outliers from distorting the clusters.

You can use the OUTSEED= data set with the SGPLOT procedure to plot _GAP_ by _FREQ_. An overlay of _RADIUS_ by _FREQ_ provides a baseline against which to compare the values of _GAP_. Outliers appear in the upper-left area of the plot, with large _GAP_ values and small _FREQ_ values. Good clusters appear in the upper-right area, with large values of both _GAP_ and _FREQ_. Good potential cluster seeds appear in the lower right, as well as in the upper-right, since large _FREQ_ values indicate high-density regions. Small _FREQ_ values in the left part of the plot indicate poor cluster seeds because the points are in low-density regions. It often helps to remove all clusters with small frequencies even though the clusters might not be remote enough to be considered outliers. Removing points in low-density regions improves cluster separation and provides visually sharper cluster outlines in scatter plots.

Displayed Output

Unless the SHORT or SUMMARY option is specified, PROC FASTCLUS displays the following:

- Initial Seeds, cluster seeds selected after one pass through the data
- Change in Cluster Seeds for each iteration, if you specify MAXITER=$n > 1$

If you specify the LEAST=p option, with ($1 < p < 2$), and you omit the IRLS option, an additional column is displayed in the Iteration History table. This column contains a character to identify the method used in each iteration. PROC FASTCLUS chooses the most efficient method to cluster the data at each iterative step, given the condition of the data. Thus, the method chosen is data dependent. The possible values are described as follows:

Value	Method
N	Newton's method
I or L	Iteratively weighted least squares (IRLS)
1	IRLS step, halved once
2	IRLS step, halved twice
3	IRLS step, halved three times

PROC FASTCLUS displays a Cluster Summary, giving the following for each cluster:

- Cluster number
- Frequency, the number of observations in the cluster
- Weight, the sum of the weights of the observations in the cluster, if you specify the WEIGHT statement
- RMS Std Deviation, the root mean squared across variables of the cluster standard deviations, which is equal to the root mean square distance between observations in the cluster
- Maximum Distance from Seed to Observation, the maximum distance from the cluster seed to any observation in the cluster
- Nearest Cluster, the number of the cluster with mean closest to the mean of the current cluster
- Centroid Distance, the distance between the centroids (means) of the current cluster and the nearest other cluster

A table of statistics for each variable is displayed unless you specify the SUMMARY option. The table contains the following:

- Total STD, the total standard deviation
- Within STD, the pooled within-cluster standard deviation
- R-Square, the R square for predicting the variable from the cluster

- RSQ/(1 - RSQ), the ratio of between-cluster variance to within-cluster variance ($R^2/(1-R^2)$)
- OVER-ALL, all of the previous quantities pooled across variables

PROC FASTCLUS also displays the following:

- Pseudo F Statistic,

  $$\frac{\frac{R^2}{c-1}}{\frac{1-R^2}{n-c}}$$

  where R square is the observed overall R square, c is the number of clusters, and n is the number of observations. The pseudo F statistic was suggested by Caliński and Harabasz (1974). See Milligan and Cooper (1985) and Cooper and Milligan (1988) regarding the use of the pseudo F statistic in estimating the number of clusters. See Example 40.2 in Chapter 40, "The CLUSTER Procedure," for a comparison of pseudo F statistics.

- Observed Overall R-Square, if you specify the SUMMARY option
- Approximate Expected Overall R-Square, the approximate expected value of the overall R square under the uniform null hypothesis assuming that the variables are uncorrelated. The value is missing if the number of clusters is greater than one-fifth the number of observations.
- Cubic Clustering Criterion, computed under the assumption that the variables are uncorrelated. The value is missing if the number of clusters is greater than one-fifth the number of observations.

  If you are interested in the approximate expected R square or the cubic clustering criterion but your variables are correlated, you should cluster principal component scores from the PRINCOMP procedure. Both of these statistics are described by Sarle (1983). The performance of the cubic clustering criterion in estimating the number of clusters is examined by Milligan and Cooper (1985) and Cooper and Milligan (1988).

- Distances Between Cluster Means, if you specify the DISTANCE option

Unless you specify the SHORT or SUMMARY option, PROC FASTCLUS displays the following:

- Cluster Means for each variable
- Cluster Standard Deviations for each variable

ODS Table Names

PROC FASTCLUS assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 45.4. For more information on ODS, see Chapter 23, "Using the Output Delivery System."

Table 45.4 ODS Tables Produced by PROC FASTCLUS

ODS Table Name	Description	Statement	Option
ApproxExpOverAllRSq	Approximate expected overall R-square, single number	PROC FASTCLUS	Default
CCC	CCC, Cubic Clustering Criterion, single number	PROC FASTCLUS	Default
ClusterList	Cluster listing, obs, id, and distances	PROC FASTCLUS	LIST
ClusterSum	Cluster summary, cluster number, distances	PROC FASTCLUS	PRINTALL
ClusterCenters	Cluster centers	PROC FASTCLUS	Default
ClusterDispersion	Cluster dispersion	PROC FASTCLUS	Default
ConvergenceStatus	Convergence status	PROC FASTCLUS	PRINTALL
Criterion	Criterion based on final seeds, single number	PROC FASTCLUS	Default
DistBetweenClust	Distance between clusters	PROC FASTCLUS	Default
InitialSeeds	Initial seeds	PROC FASTCLUS	Default
IterHistory	Iteration history, various statistics for each iteration	PROC FASTCLUS	PRINTALL
MinDist	Minimum distance between initial seeds, single number	PROC FASTCLUS	PRINTALL
NumberOfBins	Number of bins	PROC FASTCLUS	Default
ObsOverAllRSquare	Observed overall R-square, single number	PROC FASTCLUS	SUMMARY
PrelScaleEst	Preliminary L(1) scale estimate, single number	PROC FASTCLUS	PRINTALL
PseudoFStat	Pseudo F statistic, single number	PROC FASTCLUS	Default
SimpleStatistics	Simple statistics for input variables	PROC FASTCLUS	Default
VariableStat	Statistics for variables within clusters	PROC FASTCLUS	Default

Examples: FASTCLUS Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 45.1: Fisher's Iris Data

(View the complete code for this example (fastex1.sas) in the example repository.)

The iris data published by Fisher (1936) have been widely used for examples in discriminant analysis and cluster analysis. The sepal length, sepal width, petal length, and petal width are measured in millimeters on 50 iris specimens from each of three species, *Iris setosa, I. versicolor,* and *I. virginica*. Mezzich and Solomon (1980) discuss a variety of cluster analyses of the iris data.

In this example, the FASTCLUS procedure is used to find two and then three clusters. In the following code, an output data set is created, and PROC FREQ is invoked to compare the clusters with the species classification. See Output 45.1.1 and Output 45.1.2 for these results.

For three clusters, you can use the CANDISC procedure to compute canonical variables for plotting the clusters. See Output 45.1.3 and Output 45.1.4 for the results.

```
proc format;
   value specname
      1='Setosa    '
      2='Versicolor'
      3='Virginica ';
run;

data iris;
   title 'Fisher (1936) Iris Data';
   input SepalLength SepalWidth PetalLength PetalWidth Species @@;
   format Species specname.;
   label SepalLength='Sepal Length in mm.'
         SepalWidth ='Sepal Width in mm.'
         PetalLength='Petal Length in mm.'
         PetalWidth ='Petal Width in mm.';
   datalines;
50 33 14 02 1 64 28 56 22 3 65 28 46 15 2 67 31 56 24 3
63 28 51 15 3 46 34 14 03 1 69 31 51 23 3 62 22 45 15 2
59 32 48 18 2 46 36 10 02 1 61 30 46 14 2 60 27 51 16 2
65 30 52 20 3 56 25 39 11 2 65 30 55 18 3 58 27 51 19 3
68 32 59 23 3 51 33 17 05 1 57 28 45 13 2 62 34 54 23 3
77 38 67 22 3 63 33 47 16 2 67 33 57 25 3 76 30 66 21 3
49 25 45 17 3 55 35 13 02 1 67 30 52 23 3 70 32 47 14 2

   ... more lines ...

55 23 40 13 2 66 30 44 14 2 68 28 48 14 2 54 34 17 02 1
51 37 15 04 1 52 35 15 02 1 58 28 51 24 3 67 30 50 17 2
63 33 60 25 3 53 37 15 02 1
```

```
;

proc fastclus data=iris maxc=2 maxiter=10 out=clus;
   var SepalLength SepalWidth PetalLength PetalWidth;
run;

proc freq;
   tables cluster*species;
run;

proc fastclus data=iris maxc=3 maxiter=10 out=clus;
   var SepalLength SepalWidth PetalLength PetalWidth;
run;

proc freq;
   tables cluster*Species;
run;

proc candisc anova out=can;
   class cluster;
   var SepalLength SepalWidth PetalLength PetalWidth;
   title2 'Canonical Discriminant Analysis of Iris Clusters';
run;

proc sgplot data=Can;
   scatter y=Can2 x=Can1 / group=Cluster;
   title2 'Plot of Canonical Variables Identified by Cluster';
run;
```

Output 45.1.1 Fisher's Iris Data: PROC FASTCLUS with MAXC=2 and PROC FREQ

Fisher (1936) Iris Data

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=2 Maxiter=10 Converge=0.02

Initial Seeds				
Cluster	SepalLength	SepalWidth	PetalLength	PetalWidth
1	43.00000000	30.00000000	11.00000000	1.00000000
2	77.00000000	26.00000000	69.00000000	23.00000000

Minimum Distance Between Initial Seeds = 70.85196

Iteration History			
		Relative Change in Cluster Seeds	
Iteration	Criterion	1	2
1	11.0638	0.1904	0.3163
2	5.3780	0.0596	0.0264
3	5.0718	0.0174	0.00766

Convergence criterion is satisfied.

Output 45.1.1 *continued*

Criterion Based on Final Seeds =	5.0417

Cluster Summary

Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	53	3.7050	21.1621		2	39.2879
2	97	5.6779	24.6430		1	39.2879

Statistics for Variables

Variable	Total STD	Within STD	R-Square	RSQ/(1-RSQ)
SepalLength	8.28066	5.49313	0.562896	1.287784
SepalWidth	4.35866	3.70393	0.282710	0.394137
PetalLength	17.65298	6.80331	0.852470	5.778291
PetalWidth	7.62238	3.57200	0.781868	3.584390
OVER-ALL	10.69224	5.07291	0.776410	3.472463

Pseudo F Statistic =	513.92

Approximate Expected Over-All R-Squared =	0.51539

Cubic Clustering Criterion =	14.806

WARNING: The two values above are invalid for correlated variables.

Cluster Means

Cluster	SepalLength	SepalWidth	PetalLength	PetalWidth
1	50.05660377	33.69811321	15.60377358	2.90566038
2	63.01030928	28.86597938	49.58762887	16.95876289

Cluster Standard Deviations

Cluster	SepalLength	SepalWidth	PetalLength	PetalWidth
1	3.427350930	4.396611045	4.404279486	2.105525249
2	6.336887455	3.267991438	7.800577673	4.155612484

Fisher (1936) Iris Data

The FREQ Procedure

Frequency
Percent
Row Pct
Col Pct

Table of CLUSTER by Species

CLUSTER(Cluster)	Setosa	Versicolor	Virginica	Total
1	50 33.33 94.34 100.00	3 2.00 5.66 6.00	0 0.00 0.00 0.00	53 35.33
2	0 0.00 0.00 0.00	47 31.33 48.45 94.00	50 33.33 51.55 100.00	97 64.67
Total	50 33.33	50 33.33	50 33.33	150 100.00

Output 45.1.2 Fisher's Iris Data: PROC FASTCLUS with MAXC=3 and PROC FREQ

Fisher (1936) Iris Data

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=3 Maxiter=10 Converge=0.02

Initial Seeds				
Cluster	SepalLength	SepalWidth	PetalLength	PetalWidth
1	58.00000000	40.00000000	12.00000000	2.00000000
2	77.00000000	38.00000000	67.00000000	22.00000000
3	49.00000000	25.00000000	45.00000000	17.00000000

Minimum Distance Between Initial Seeds = 38.23611

Iteration History				
		Relative Change in Cluster Seeds		
Iteration	Criterion	1	2	3
1	6.7591	0.2652	0.3205	0.2985
2	3.7097	0	0.0459	0.0317
3	3.6427	0	0.0182	0.0124

Convergence criterion is satisfied.

Criterion Based on Final Seeds = 3.6289

Cluster Summary						
Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	50	2.7803	12.4803		3	33.5693
2	38	4.0168	14.9736		3	17.9718
3	62	4.0398	16.9272		2	17.9718

Statistics for Variables				
Variable	Total STD	Within STD	R-Square	RSQ/(1-RSQ)
SepalLength	8.28066	4.39488	0.722096	2.598359
SepalWidth	4.35866	3.24816	0.452102	0.825156
PetalLength	17.65298	4.21431	0.943773	16.784895
PetalWidth	7.62238	2.45244	0.897872	8.791618
OVER-ALL	10.69224	3.66198	0.884275	7.641194

Pseudo F Statistic = 561.63

Approximate Expected Over-All R-Squared = 0.62728

Cubic Clustering Criterion = 25.021

Output 45.1.2 *continued*

WARNING: The two values above are invalid for correlated variables.

Cluster Means				
Cluster	**SepalLength**	**SepalWidth**	**PetalLength**	**PetalWidth**
1	50.06000000	34.28000000	14.62000000	2.46000000
2	68.50000000	30.73684211	57.42105263	20.71052632
3	59.01612903	27.48387097	43.93548387	14.33870968

Cluster Standard Deviations				
Cluster	**SepalLength**	**SepalWidth**	**PetalLength**	**PetalWidth**
1	3.524896872	3.790643691	1.736639965	1.053855894
2	4.941550255	2.900924461	4.885895746	2.798724562
3	4.664100551	2.962840548	5.088949673	2.974997167

Fisher (1936) Iris Data

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of CLUSTER by Species				
		Species			
	CLUSTER(Cluster)	**Setosa**	**Versicolor**	**Virginica**	**Total**
	1	50 33.33 100.00 100.00	0 0.00 0.00 0.00	0 0.00 0.00 0.00	50 33.33
	2	0 0.00 0.00 0.00	2 1.33 5.26 4.00	36 24.00 94.74 72.00	38 25.33
	3	0 0.00 0.00 0.00	48 32.00 77.42 96.00	14 9.33 22.58 28.00	62 41.33
	Total	50 33.33	50 33.33	50 33.33	150 100.00

Output 45.1.3 Fisher's Iris Data Using PROC CANDISC

Fisher (1936) Iris Data
Canonical Discriminant Analysis of Iris Clusters

The CANDISC Procedure

Total Sample Size	150	**DF Total**	149
Variables	4	**DF Within Classes**	147
Classes	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Output 45.1.3 *continued*

Class Level Information				
CLUSTER	Variable Name	Frequency	Weight	Proportion
1	_1	50	50.0000	0.333333
2	_2	38	38.0000	0.253333
3	_3	62	62.0000	0.413333

Fisher (1936) Iris Data
Canonical Discriminant Analysis of Iris Clusters

The CANDISC Procedure

Univariate Test Statistics

F Statistics, Num DF=2, Den DF=147

Variable	Label	Total Standard Deviation	Pooled Standard Deviation	Between Standard Deviation	R-Square	R-Square / (1-RSq)	F Value	Pr > F
SepalLength	Sepal Length in mm.	8.2807	4.3949	8.5893	0.7221	2.5984	190.98	<.0001
SepalWidth	Sepal Width in mm.	4.3587	3.2482	3.5774	0.4521	0.8252	60.65	<.0001
PetalLength	Petal Length in mm.	17.6530	4.2143	20.9336	0.9438	16.7849	1233.69	<.0001
PetalWidth	Petal Width in mm.	7.6224	2.4524	8.8164	0.8979	8.7916	646.18	<.0001

Average R-Square	
Unweighted	0.7539604
Weighted by Variance	0.8842753

Multivariate Statistics and F Approximations

S=2 M=0.5 N=71

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.03222337	164.55	8	288	<.0001
Pillai's Trace	1.25669612	61.29	8	290	<.0001
Hotelling-Lawley Trace	21.06722883	377.66	8	203.4	<.0001
Roy's Greatest Root	20.63266809	747.93	4	145	<.0001

NOTE: F Statistic for Roy's Greatest Root is an upper bound.

NOTE: F Statistic for Wilks' Lambda is exact.

Output 45.1.3 *continued*

Fisher (1936) Iris Data
Canonical Discriminant Analysis of Iris Clusters

The CANDISC Procedure

	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq) Eigenvalue	Difference	Proportion	Cumulative
1	0.976613	0.976123	0.003787	0.953774	20.6327	20.1981	0.9794	0.9794
2	0.550384	0.543354	0.057107	0.302923	0.4346		0.0206	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero					
	Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
1	0.03222337	164.55	8	288	<.0001
2	0.69707749	21.00	3	145	<.0001

Fisher (1936) Iris Data
Canonical Discriminant Analysis of Iris Clusters

The CANDISC Procedure

Total Canonical Structure			
Variable	Label	Can1	Can2
SepalLength	Sepal Length in mm.	0.831965	0.452137
SepalWidth	Sepal Width in mm.	-0.515082	0.810630
PetalLength	Petal Length in mm.	0.993520	0.087514
PetalWidth	Petal Width in mm.	0.966325	0.154745

Between Canonical Structure			
Variable	Label	Can1	Can2
SepalLength	Sepal Length in mm.	0.956160	0.292846
SepalWidth	Sepal Width in mm.	-0.748136	0.663545
PetalLength	Petal Length in mm.	0.998770	0.049580
PetalWidth	Petal Width in mm.	0.995952	0.089883

Pooled Within Canonical Structure			
Variable	Label	Can1	Can2
SepalLength	Sepal Length in mm.	0.339314	0.716082
SepalWidth	Sepal Width in mm.	-0.149614	0.914351
PetalLength	Petal Length in mm.	0.900839	0.308136
PetalWidth	Petal Width in mm.	0.650123	0.404282

Output 45.1.3 *continued*

Fisher (1936) Iris Data
Canonical Discriminant Analysis of Iris Clusters

The CANDISC Procedure

Total-Sample Standardized Canonical Coefficients

Variable	Label	Can1	Can2
SepalLength	Sepal Length in mm.	0.047747341	1.021487262
SepalWidth	Sepal Width in mm.	-0.577569244	0.864455153
PetalLength	Petal Length in mm.	3.341309573	-1.283043758
PetalWidth	Petal Width in mm.	0.996451144	0.900476563

Pooled Within-Class Standardized Canonical Coefficients

Variable	Label	Can1	Can2
SepalLength	Sepal Length in mm.	0.0253414487	0.5421446856
SepalWidth	Sepal Width in mm.	-.4304161258	0.6442092294
PetalLength	Petal Length in mm.	0.7976741592	-.3063023132
PetalWidth	Petal Width in mm.	0.3205998034	0.2897207865

Raw Canonical Coefficients

Variable	Label	Can1	Can2
SepalLength	Sepal Length in mm.	0.0057661265	0.1233581748
SepalWidth	Sepal Width in mm.	-.1325106494	0.1983303556
PetalLength	Petal Length in mm.	0.1892773419	-.0726814163
PetalWidth	Petal Width in mm.	0.1307270927	0.1181359305

Class Means on Canonical Variables

CLUSTER	Can1	Can2
1	-6.131527227	0.244761516
2	4.931414018	0.861972277
3	1.922300462	-0.725693908

Output 45.1.4 Plot of Fisher's Iris Data Using PROC CANDISC

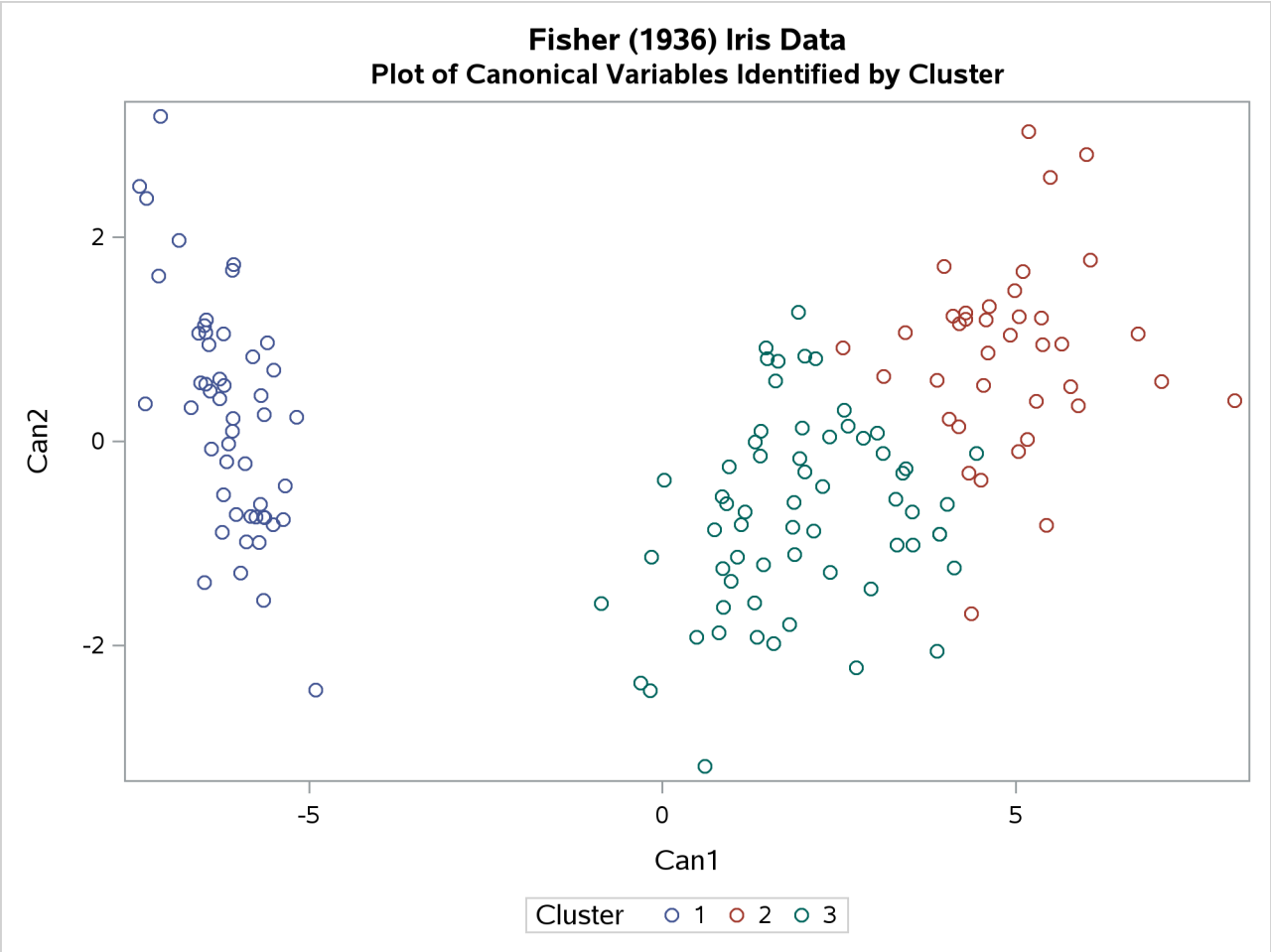

Example 45.2: Outliers

(View the complete code for this example (fastex2.sas) in the example repository.)

This example involves data artificially generated to contain two clusters and several severe outliers. A preliminary analysis specifies 20 clusters and outputs an OUTSEED= data set to be used for a diagnostic plot. The exact number of initial clusters is not important; similar results could be obtained with 10 or 50 initial clusters. Examination of the plot suggests that clusters with more than five (again, the exact number is not important) observations can yield good seeds for the main analysis. A DATA step deletes clusters with five or fewer observations, and the remaining cluster means provide seeds for the next PROC FASTCLUS analysis.

Two clusters are requested; the LEAST= option specifies the mean absolute deviation criterion (LEAST=1). Values of the LEAST= option less than 2 reduce the effect of outliers on cluster centers.

The next analysis also requests two clusters; the STRICT= option is specified to prevent outliers from distorting the results. The STRICT= value is chosen to be close to the _GAP_ and _RADIUS_ values of the larger clusters in the diagnostic plot; the exact value is not critical.

A final PROC FASTCLUS run assigns the outliers to clusters.

The following SAS statements implement these steps, and the results are displayed in Output 45.2.3 through Output 45.2.8. First, an artificial data set is created with two clusters and some outliers. Then PROC FASTCLUS is run with many clusters to produce an OUTSEED= data set. A diagnostic plot using the variables _GAP_ and _RADIUS_ is then produced using the SGSCATTER procedure. The results from these steps are shown in Output 45.2.1 and Output 45.2.2.

```
title 'Using PROC FASTCLUS to Analyze Data with Outliers';
data x;
   drop n;
   do n=1 to 100;
      x=rannor(12345)+2;
      y=rannor(12345);
      output;
   end;
   do n=1 to 100;
      x=rannor(12345)-2;
      y=rannor(12345);
      output;
   end;
   do n=1 to 10;
      x=10*rannor(12345);
      y=10*rannor(12345);
      output;
   end;
run;

title2 'Preliminary PROC FASTCLUS Analysis with 20 Clusters';
proc fastclus data=x outseed=mean1 maxc=20 maxiter=0 summary;
   var x y;
run;

proc sgscatter data=mean1;
   compare y=(_gap_ _radius_) x=_freq_;
run;
```

Output 45.2.1 Preliminary Analysis of Data with Outliers Using PROC FASTCLUS

Using PROC FASTCLUS to Analyze Data with Outliers
Preliminary PROC FASTCLUS Analysis with 20 Clusters

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=20 Maxiter=0

Criterion Based on Final Seeds =	0.6873

Output 45.2.1 *continued*

Cluster Summary

Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	8	0.4753	1.1924		19	1.7205
2	1	.	0		6	6.2847
3	44	0.6252	1.6774		5	1.4386
4	1	.	0		20	5.2130
5	38	0.5603	1.4528		3	1.4386
6	2	0.0542	0.1085		2	6.2847
7	1	.	0		14	2.5094
8	2	0.6480	1.2961		1	1.8450
9	1	.	0		7	9.4534
10	1	.	0		18	4.2514
11	1	.	0		16	4.7582
12	20	0.5911	1.6291		16	1.5601
13	5	0.6682	1.4244		3	1.9553
14	1	.	0		7	2.5094
15	5	0.4074	1.2678		3	1.7609
16	22	0.4168	1.5139		19	1.4936
17	8	0.4031	1.4794		5	1.5564
18	1	.	0		10	4.2514
19	45	0.6475	1.6285		16	1.4936
20	3	0.5719	1.3642		15	1.8999

Pseudo F Statistic = 207.58

Observed Over-All R-Squared = 0.95404

Approximate Expected Over-All R-Squared = 0.96103

Cubic Clustering Criterion = -2.503

WARNING: The two values above are invalid for correlated variables.

Output 45.2.2 Preliminary Analysis of Data with Outliers: Plot Using PROC SGSCATTER

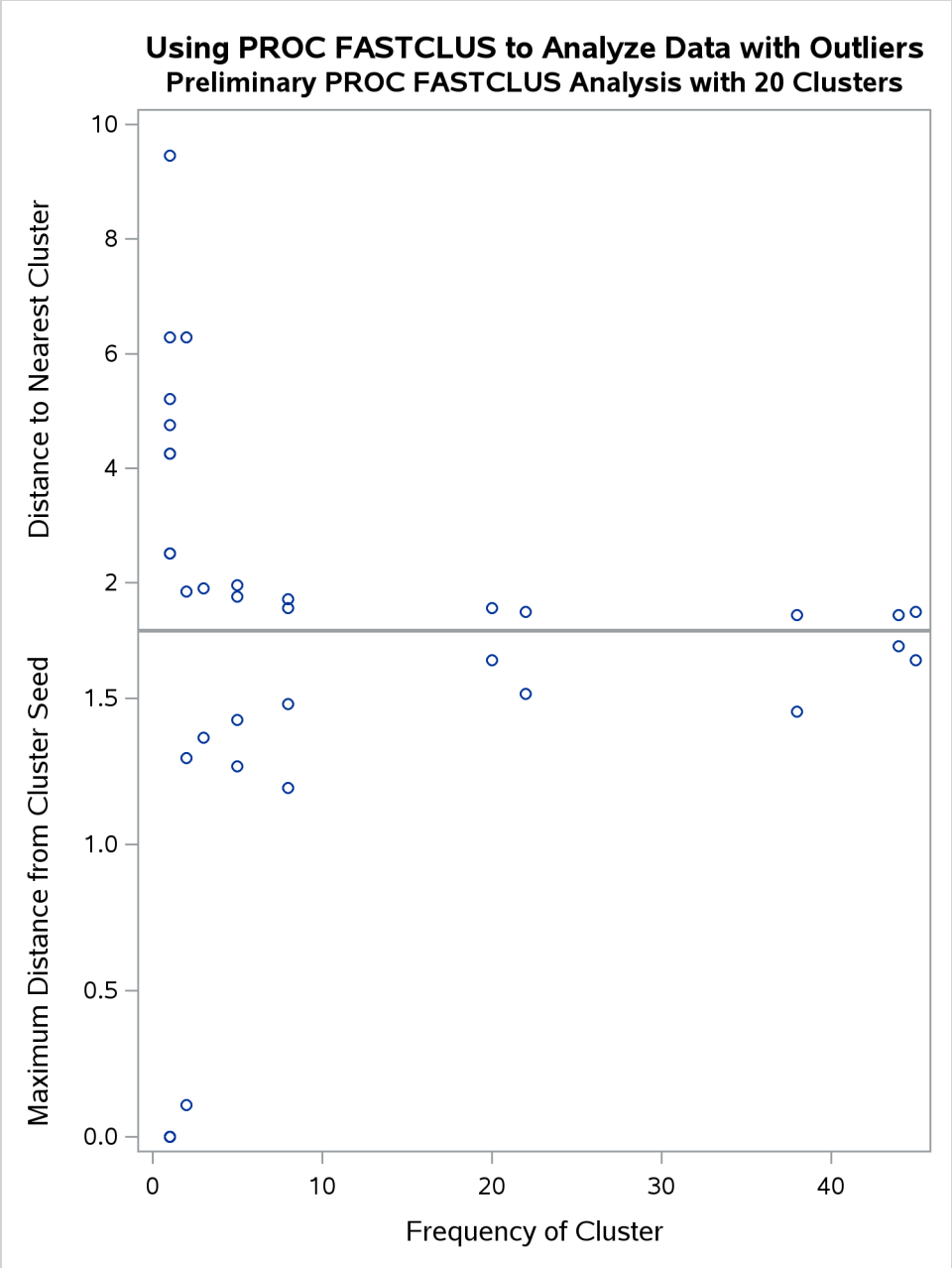

In the following SAS statements, a DATA step is used to remove low frequency clusters, then the FASTCLUS procedure is run again, selecting seeds from the high frequency clusters in the previous analysis using LEAST=1 clustering criterion. The results are shown in Output 45.2.3 and Output 45.2.4.

```
data seed;
   set mean1;
   if _freq_>5;
run;

title2 'PROC FASTCLUS Analysis Using LEAST= Clustering Criterion';
```

```
title3 'Values < 2 Reduce Effect of Outliers on Cluster Centers';
proc fastclus data=x seed=seed maxc=2 least=1 out=out;
   var x y;
run;

proc sgplot data=out;
   scatter y=y x=x / group=cluster;
run;
```

Output 45.2.3 Analysis of Data with Outliers Using the LEAST= Option

Using PROC FASTCLUS to Analyze Data with Outliers
PROC FASTCLUS Analysis Using LEAST= Clustering Criterion
Values < 2 Reduce Effect of Outliers on Cluster Centers

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=2 Maxiter=20 Converge=0.0001 Least=1

Initial Seeds		
Cluster	x	y
1	2.794174248	-0.065970836
2	-2.027300384	-2.051208579

Minimum Distance Between Initial Seeds = 6.806712

Preliminary L(1) Scale Estimate = 2.796579

Number of Bins = 100

Iteration History				
			Relative Change in Cluster Seeds	
Iteration	Criterion	Maximum Bin Size	1	2
1	1.3983	0.2263	0.4091	0.6696
2	1.0776	0.0226	0.00511	0.0452
3	1.0771	0.00226	0.00229	0.00234
4	1.0771	0.000396	0.000253	0.000144
5	1.0771	0.000396	0	0

Convergence criterion is satisfied.

Criterion Based on Final Seeds = 1.0771

Cluster Summary						
Cluster	Frequency	Mean Absolute Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Medians
1	102	1.1278	24.1622		2	4.2585
2	108	1.0494	14.8292		1	4.2585

Output 45.2.3 *continued*

Cluster Medians		
Cluster	**x**	**y**
1	1.923023887	0.222482918
2	-1.826721743	-0.286253041

Mean Absolute Deviations from Final Seeds		
Cluster	**x**	**y**
1	1.113465261	1.142120480
2	0.890331835	1.208370913

Output 45.2.4 Analysis Plot of Data with Outliers

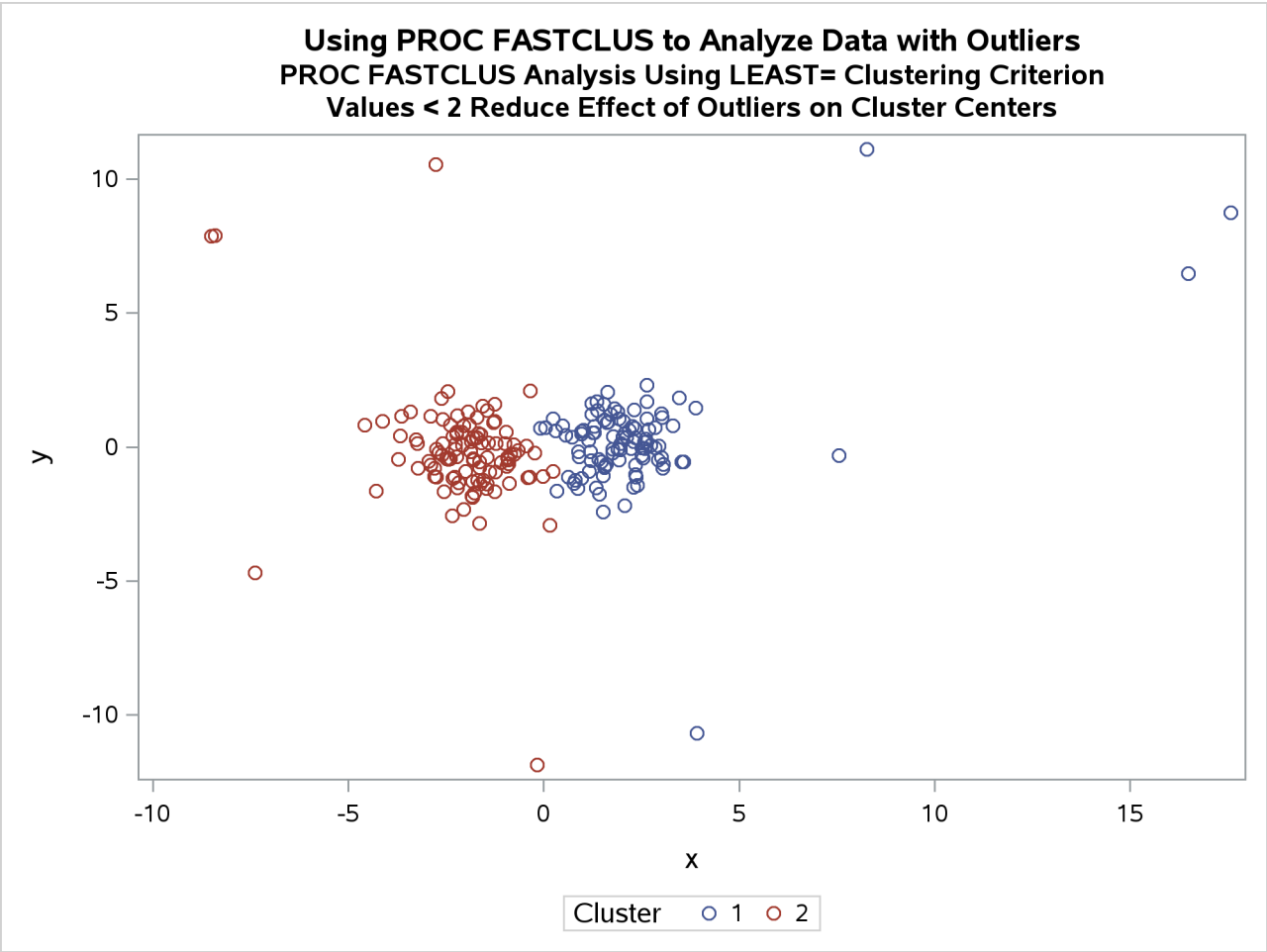

The FASTCLUS procedure is run again, selecting seeds from high frequency clusters in the previous analysis. STRICT= prevents outliers from distorting the results. The results are shown in Output 45.2.5 and Output 45.2.6.

```
title2 'PROC FASTCLUS Analysis Using STRICT= to Omit Outliers';
proc fastclus data=x seed=seed
     maxc=2 strict=3.0 out=out outseed=mean2;
   var x y;
run;

proc sgplot data=out;
   scatter y=y  x=x / group=cluster;
run;
```

Output 45.2.5 Cluster Analysis with Outliers Omitted: PROC FASTCLUS SGPLOT

Using PROC FASTCLUS to Analyze Data with Outliers
PROC FASTCLUS Analysis Using STRICT= to Omit Outliers

The FASTCLUS Procedure
Replace=FULL Radius=0 Strict=3 Maxclusters=2 Maxiter=1

Initial Seeds		
Cluster	x	y
1	2.794174248	-0.065970836
2	-2.027300384	-2.051208579

Criterion Based on Final Seeds = 0.9515

Cluster Summary						
Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	99	0.9501	2.9589		2	3.7666
2	99	0.9290	2.8011		1	3.7666

12 Observation(s) were not assigned to a cluster
because the minimum distance to a cluster seed exceeded the STRICT= value.

Statistics for Variables				
Variable	Total STD	Within STD	R-Square	RSQ/(1-RSQ)
x	2.06854	0.87098	0.823609	4.669219
y	1.02113	1.00352	0.039093	0.040683
OVER-ALL	1.63119	0.93959	0.669891	2.029303

Pseudo F Statistic = 397.74

Approximate Expected Over-All R-Squared = 0.60615

Cubic Clustering Criterion = 3.197

WARNING: The two values above are invalid for correlated variables.

Cluster Means		
Cluster	x	y
1	1.825111432	0.141211701
2	-1.919910712	-0.261558725

Output 45.2.5 *continued*

Cluster Standard Deviations		
Cluster	**x**	**y**
1	0.889549271	1.006965219
2	0.852000588	1.000062579

Output 45.2.6 Cluster Analysis with Outliers Omitted: Plot Using PROC SGPLOT

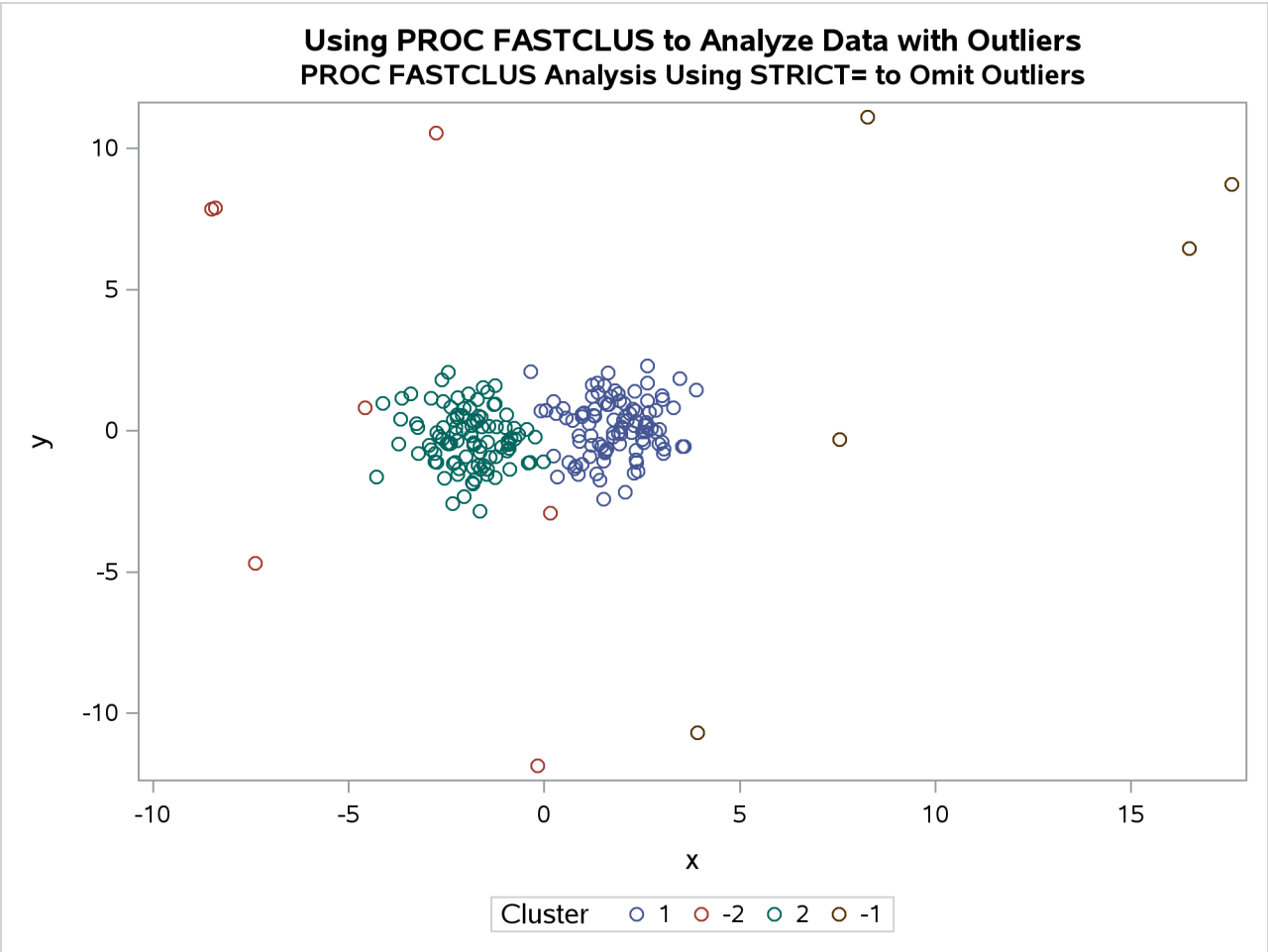

Finally, the FASTCLUS procedure is run one more time with zero iterations to assign outliers and tails to clusters. The results are show in Output 45.2.7 and Output 45.2.8.

```
title2 'Final PROC FASTCLUS Analysis Assigning Outliers to Clusters';
proc fastclus data=x seed=mean2 maxc=2 maxiter=0 out=out;
   var x y;
run;

proc sgplot data=out;
   scatter y=y x=x / group=cluster;
run;
```

Output 45.2.7 Cluster Analysis with Outliers Omitted: PROC FASTCLUS

Using PROC FASTCLUS to Analyze Data with Outliers
Final PROC FASTCLUS Analysis Assigning Outliers to Clusters

The FASTCLUS Procedure
Replace=FULL Radius=0 Maxclusters=2 Maxiter=0

Initial Seeds		
Cluster	x	y
1	1.825111432	0.141211701
2	-1.919910712	-0.261558725

Criterion Based on Final Seeds = 2.0594

Cluster Summary						
Cluster	Frequency	RMS Std Deviation	Maximum Distance from Seed to Observation	Radius Exceeded	Nearest Cluster	Distance Between Cluster Centroids
1	103	2.2569	17.9426		2	4.3753
2	107	1.8371	11.7362		1	4.3753

Statistics for Variables				
Variable	Total STD	Within STD	R-Square	RSQ/(1-RSQ)
x	2.92721	1.95529	0.555950	1.252000
y	2.15248	2.14754	0.009347	0.009435
OVER-ALL	2.56922	2.05367	0.364119	0.572621

Pseudo F Statistic = 119.11

Approximate Expected Over-All R-Squared = 0.49090

Cubic Clustering Criterion = -5.338

WARNING: The two values above are invalid for correlated variables.

Cluster Means		
Cluster	x	y
1	2.280017469	0.263940765
2	-2.075547895	-0.151348765

Cluster Standard Deviations		
Cluster	x	y
1	2.412264861	2.089922815
2	1.379355878	2.201567557

Output 45.2.8 Cluster Analysis with Outliers Omitted: Plot Using PROC SGPLOT

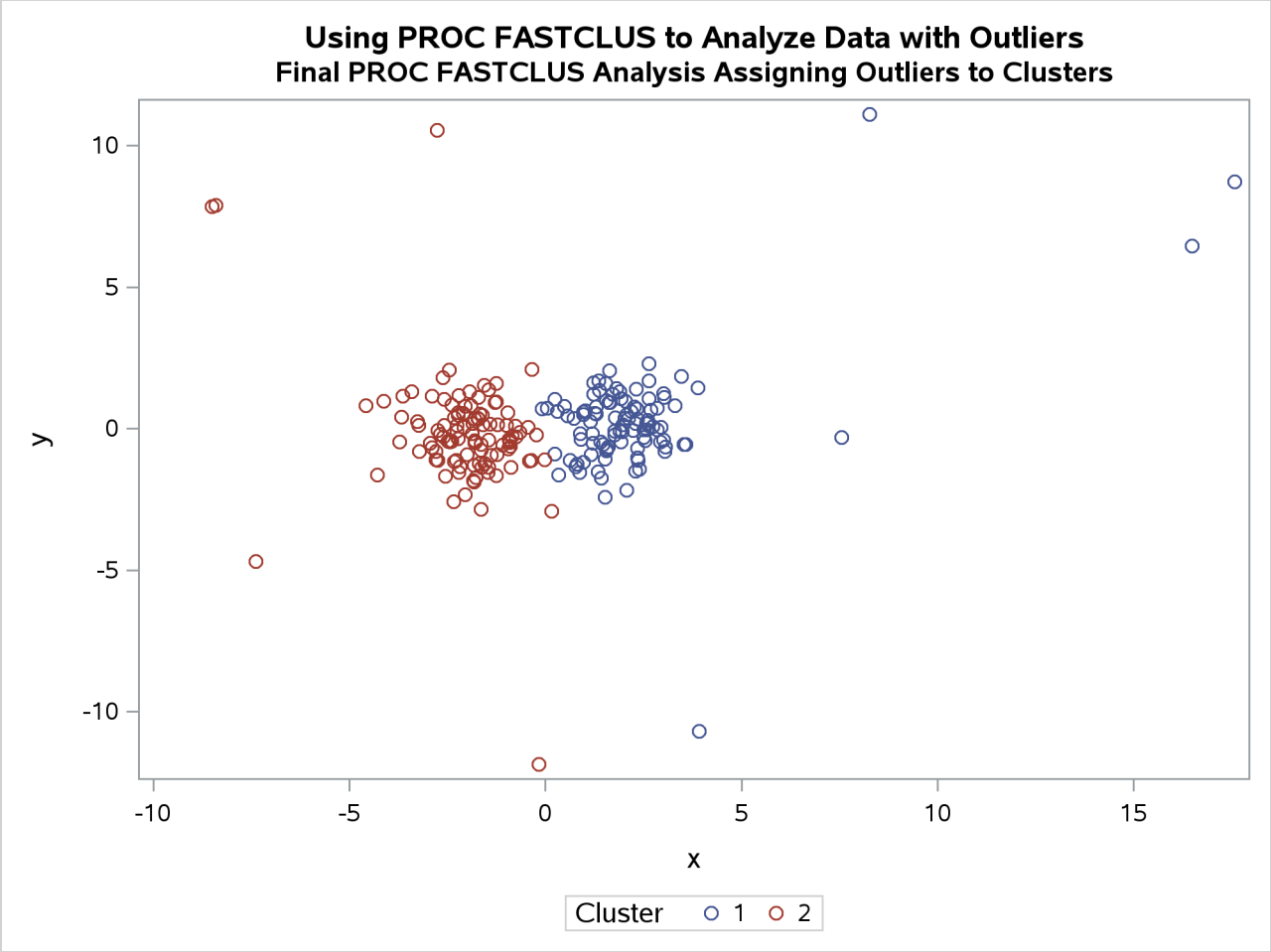

References

Anderberg, M. R. (1973). *Cluster Analysis for Applications*. New York: Academic Press.

Caliński, T., and Harabasz, J. (1974). "A Dendrite Method for Cluster Analysis." *Communications in Statistics—Theory and Methods* 3:1–27.

Cooper, M. C., and Milligan, G. W. (1988). "The Effect of Error on Determining the Number of Clusters." In *Proceedings of the International Workshop on Data Analysis, Decision Support, and Expert Knowledge Representation in Marketing and Related Areas of Research*, 319–328. Berlin: Springer-Verlag.

Everitt, B. S. (1980). *Cluster Analysis*. 2nd ed. London: Heineman Educational Books.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Gonin, R., and Money, A. H. (1989). *Nonlinear L_p-Norm Estimation*. New York: Marcel Dekker.

Hartigan, J. A. (1975). *Clustering Algorithms*. New York: John Wiley & Sons.

MacQueen, J. B. (1967). "Some Methods for Classification and Analysis of Multivariate Observations." *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* 1:281–297.

Mezzich, J. E., and Solomon, H. (1980). *Taxonomy and Behavioral Science*. New York: Academic Press.

Milligan, G. W. (1980). "An Examination of the Effect of Six Types of Error Perturbation on Fifteen Clustering Algorithms." *Psychometrika* 45:325–342.

Milligan, G. W., and Cooper, M. C. (1985). "An Examination of Procedures for Determining the Number of Clusters in a Data Set." *Psychometrika* 50:159–179.

Puranen, J. (1917). "Fish Catch data set (1917)." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Sarle, W. S. (1983). *Cubic Clustering Criterion*. Technical Report A-108, SAS Institute Inc., Cary, NC.

Spath, H. (1980). *Cluster Analysis Algorithms*. Chichester, UK: Ellis Horwood.

Spath, H. (1985). *Cluster Dissection and Analysis*. Chichester, UK: Ellis Horwood.

Tou, J. T., and Gonzalez, R. C. (1974). *Pattern Recognition Principles*. Reading, MA: Addison-Wesley.

Chapter 46
The FMM Procedure

Contents

Overview: FMM Procedure . . . **3020**
- Basic Features . . . 3021
- Assumptions . . . 3022
- Notation for the Finite Mixture Model . . . 3022
  - Homogeneous Mixtures . . . 3023
  - Special Mixtures . . . 3023
- PROC FMM Contrasted with Other SAS Procedures . . . 3023

Getting Started: FMM Procedure . . . **3024**
- Mixture Modeling for Binomial Overdispersion: “Student,” Pearson, Beer, and Yeast . 3024
- Modeling Zero-Inflation: Is It Better to Fish Poorly or Not to Have Fished at All? . . . 3030
- Looking for Multiple Modes: Are Galaxies Clustered? . . . 3038
  - Comparison with Roeder’s Method . . . 3045

Syntax: FMM Procedure . . . **3048**
- PROC FMM Statement . . . 3048
- BAYES Statement . . . 3062
- BY Statement . . . 3071
- CLASS Statement . . . 3072
- FREQ Statement . . . 3072
- ID Statement . . . 3072
- MODEL Statement . . . 3073
  - Response Variable Options . . . 3074
  - Model Options . . . 3076
- OUTPUT Statement . . . 3081
- PERFORMANCE Statement . . . 3084
- PROBMODEL Statement . . . 3085
- RANDSTART Statement . . . 3087
- RESTRICT Statement . . . 3089
- WEIGHT Statement . . . 3091

Details: FMM Procedure . . . **3091**
- A Gentle Introduction to Finite Mixture Models . . . 3091
  - The Form of the Finite Mixture Model . . . 3091
  - Mixture Models Contrasted with Mixing and Mixed Models: Untangling the Terminology Web . . . 3092
  - Overdispersion . . . 3093
- Log-Likelihood Functions for Response Distributions . . . 3094
- Bayesian Analysis . . . 3102

Conjugate Sampling . . . 3102
Metropolis-Hastings Algorithm . . . 3102
Latent Variables via Data Augmentation . . . 3103
Prior Distributions . . . 3104
Parameterization of Model Effects . . . 3105
Random Starting Values . . . 3105
Default Output . . . 3106
Model Information . . . 3106
Class Level Information . . . 3106
Number of Observations . . . 3106
Response Profile . . . 3107
Default Output for Maximum Likelihood . . . 3107
Default Output for Bayesian Estimation . . . 3109
ODS Table Names . . . 3110
ODS Graphics . . . 3113
Examples: FMM Procedure . . . **3114**
Example 46.1: Modeling Mixing Probabilities: All Mice Are Equal, but Some Mice Are More Equal Than Others . . . 3114
Example 46.2: The Usefulness of Custom Starting Values: When Do Cows Eat? . . . 3123
Example 46.3: Enforcing Homogeneity Constraints: Count and Dispersion—It Is All Over! . . . 3132
Example 46.4: Modeling Multinomial Overdispersion: Town and Country . . . 3137
References . . . **3141**

Overview: FMM Procedure

The FMM procedure fits statistical models to data for which the distribution of the response is a finite mixture of distributions—that is, each response is drawn with unknown probability from one of several distributions. You can use PROC FMM to model the component distributions in addition to the mixing probabilities; see the section "A Gentle Introduction to Finite Mixture Models" on page 3091 for more precise definitions and discussion of similar but distinct modeling methodologies.

Classical statistical models are a special case of the finite mixture models in which the distribution of the data has only a single component.

Finite mixture models are useful for the following applications:

- estimating multimodal or heavy-tailed densities
- fitting zero-inflated or hurdle models to count data with excess zeros
- modeling overdispersed data
- fitting regression models with complex error distributions

- classifying observations according to predicted component probabilities
- accounting for unobservable, omitted variables
- estimating switching regressions

The FMM procedure is designed to fit finite mixtures of regression models or finite mixtures of generalized linear models in which the covariates and regression structure can be the same across components or might be different. You can fit finite mixture models by maximum likelihood or Bayesian methods.

For more information about the differences between the FMM procedure and other statistical modeling procedures in SAS/STAT software, see the section "PROC FMM Contrasted with Other SAS Procedures" on page 3023.

Basic Features

The FMM procedure estimates the parameters in univariate finite mixture models and produces various statistics to evaluate parameters and model fit. The following list summarizes some basic features of the FMM procedure:

- maximum likelihood estimation for all models
- Markov chain Monte Carlo estimation for many models, including zero-inflated Poisson models
- many built-in link and distribution functions for modeling, including the beta, shifted *t*, Weibull, beta-binomial, and generalized Poisson distributions, in addition to many standard members of the exponential family of distributions
- specialized built-in mixture models such as binomial cluster models and multinomial cluster models (Morel and Nagaraj 1993; Morel and Neerchal 1997; Neerchal and Morel 1998)
- acceptance of multiple MODEL statements to build mixture models in which the model effects, distributions, or link functions vary across mixture components
- model-building syntax using CLASS and effect-based MODEL statements familiar from many other SAS/STAT procedures (for example, the GLM, GLIMMIX, and MIXED procedures)
- evaluation of sequences of mixture models when you specify ranges for the number of components
- simple syntax to impose linear equality and inequality constraints among parameters
- ability to model regression and classification effects in the mixing probabilities through the PROBMODEL statement
- ability to incorporate full or partially known component membership into the analysis through the PARTIAL= option in the PROC FMM statement
- OUTPUT statement that produces a SAS data set with important statistics for interpreting mixture models, such as component log likelihoods and prior and posterior probabilities
- ability to add zero-inflation to any univariate model

- output data set with posterior parameter values for the Markov chain
- high degree of multithreading for high-performance optimization and Monte Carlo sampling
- ability to generate and evaluate multiple sets of random starting values

The FMM procedure uses ODS Graphics to create graphical output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the FMM procedure, see the PLOTS options in the PROC FMM statement.

Assumptions

The FMM procedure makes the following assumptions in fitting statistical models:

- The number of components k in the finite mixture is known a priori and is not a parameter to be estimated.
- The parameters of the components are distinct a priori.
- The observations are uncorrelated.

Notation for the Finite Mixture Model

The general expression for the finite mixture model fitted with the FMM procedure is as follows:

$$f(y) = \sum_{j=1}^{k} \pi_j(\mathbf{z}, \boldsymbol{\alpha}_j)\, p_j(y; \mathbf{x}_j' \boldsymbol{\beta}_j, \phi_j)$$

The number of components in the mixture is denoted as k. The mixture probabilities π_j can depend on regressor variables $\mathbf{z}$ and parameters $\boldsymbol{\alpha}_j$. By default, the FMM procedure models these probabilities using a logit transform if $k = 2$ and as a generalized logit model if $k > 2$. The component distributions p_j can also depend on regressor variables in $\mathbf{x}_j$, regression parameters $\boldsymbol{\beta}_j$, and possibly scale parameters ϕ_j. Notice that the component distributions p_j are indexed by j because the distributions might belong to different families. For example, in a two-component model, you might model one component as a normal (Gaussian) variable and the second component as a variable with a t distribution with low degrees of freedom to manage overdispersion.

The mixture probabilities π_j satisfy $\pi_j \geq 0$, for all j, and

$$\sum_{j=1}^{k} \pi_j(\mathbf{z}, \boldsymbol{\alpha}_j) = 1$$

Homogeneous Mixtures

If the component distributions are of the same distributional form, the mixture is called homogeneous. In most applications of homogeneous mixtures, the mixing probabilities do not depend on regression parameters. The general model then simplifies to

$$f(y) = \sum_{j=1}^{k} \pi_j \, p(y; \mathbf{x}'\boldsymbol{\beta}_j, \phi_j)$$

Because the component distributions depend on regression parameters $\boldsymbol{\beta}_j$, this model is known as a homogeneous regression mixture. A homogeneous regression mixture assumes that the regression effects are the same across the components, although the FMM procedure does not impose such a restriction. If the component distributions do not contain regression effects, the model

$$f(y) = \sum_{j=1}^{k} \pi_j \, p(y; \mu_j, \phi_j)$$

is *the* homogeneous mixture model. A classical case is the estimation of a continuous density as a k-component mixture of normal distributions.

Special Mixtures

The FMM procedure enables you to fit several special mixture models. The Morel-Neerchal binomial cluster model (Morel and Nagaraj 1993; Morel and Neerchal 1997; Neerchal and Morel 1998) is a mixture of binomial distributions in which the success probabilities depend on the mixing probabilities. The multinomial cluster model is a generalization of the binomial cluster model. It is a mixture of multinomial distributions in which the outcome probability vector depends on the mixing probabilities.

Zero-inflated count models are obtained as two-component mixtures where one component is a classical count model—such as the Poisson or negative binomial model—and the other component is a distribution that is concentrated at zero. If the nondegenerate part of this special mixture is a zero-truncated model, the resulting two-component mixture is known as a hurdle model (Cameron and Trivedi 1998).

PROC FMM Contrasted with Other SAS Procedures

Because the FMM procedure fits finite mixtures of generalized linear models, it can also fit standard forms of these models in which the distribution of the data does not follow a mixture. This enables you to use the FMM procedure to estimate parameters in models that can be fit with the CATMOD, LOGISTIC, GENMOD, or GLIMMIX procedures. However, the FMM procedure does not fit models that have random effects.

The FMM procedure has limited postprocessing capabilities compared to some other statistical procedures that are based on linear models. Concepts that are well understood and commonplace in linear models, such as (linear) estimable functions, estimability, and least squares means, do not apply to mixture models in the same way. For example, even the computation of a predicted value is not without ambiguity. You can estimate the means in the component distributions in addition to the overall mean of the mixture.

The FMM procedure provides a limited number of built-in distributions and link functions. User-defined distributions or link functions are not supported. Mixture models with component distributions that are not supported by the FMM procedure can be fit with the NLMIXED procedure.

For Bayesian estimation, the FMM procedure implements a small number of highly specialized sampling algorithms. These algorithms are very efficient and specifically designed for generalized linear models and their mixtures. This limits, for example, the allowable specifications for prior distributions of the model parameters. Models that do not fit the targeted algorithms of the FMM procedure can be fit with the MCMC procedure.

Getting Started: FMM Procedure

Mixture Modeling for Binomial Overdispersion: "Student," Pearson, Beer, and Yeast

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/fmmgs1.sas`.)

The following example demonstrates how you can model a complicated, two-component binomial mixture distribution, either with maximum likelihood or with Bayesian methods, with a few simple PROC FMM statements.

William Sealy Gosset, a chemist at the Arthur Guinness Son and Company brewery in Dublin, joined the statistical laboratory of Karl Pearson in 1906–1907 to study statistics. At first Gosset—who published all but one paper under the pseudonym "Student" because his employer forbade publications by employees after a co-worker had disclosed trade secrets—worked on the Poisson limit to the binomial distribution, using haemacytometer yeast cell counts. Gosset's interest in studying small-sample (and limit) problems was motivated by the small sample sizes he typically saw in his work at the brewery.

Subsequently, Gosset's yeast count data have been examined and revisited by many authors. In 1915, Karl Pearson undertook his own examination and realized that the variability in "Student's" data exceeded that consistent with a Poisson distribution. Pearson (1915) bemoans the fact that if this were so, "it is certainly most unfortunate that such material should have been selected to illustrate Poisson's limit to the binomial."

Using a count of Gosset's yeast cell counts on the 400 squares of a haemacytometer (Table 46.1), Pearson argues that a mixture process would explain the heterogeneity (beyond the Poisson).

Table 46.1 "Student's" Yeast Cell Counts

Number of Cells	0	1	2	3	4	5
Frequency	213	128	37	18	3	1

Pearson fits various models to these data, chief among them a mixture of two binomial series

$$\nu_1(p_1+q_1)^\theta + \nu_2(p_2+q_2)^\theta$$

where θ is real-valued and thus the binomial series expands to

$$(p+q)^\theta = \sum_{k=0}^{\infty} \frac{\Gamma(\theta+1)}{\Gamma(k+1)\Gamma(\theta-k+1)}\, p^k\, q^{\theta-k}$$

Pearson's fitted model has $\theta = 4.89997$, $\nu_1 = 356.986$, $\nu_2 = 43.014$ (corresponding to a mixing proportion of $356.986/(43.014 + 356.986) = 0.892$), and estimated success probabilities in the binomial components of 0.1017 and 0.4514, respectively. The success probabilities indicate that although the data have about a 90% chance of coming from a distribution with small success probability of about 0.1, there is a 10% chance of coming from a distribution with a much larger success probability of about 0.45.

If θ is an integer, the binomial series is the cumulative mass function of a binomial random variable. The value of θ suggests that a suitable model for these data could also be constructed as a two-component mixture of binomial random variables as follows:

$$f(y) = \pi \,\mathrm{binomial}(5, \mu_1) + (1 - \pi)\,\mathrm{binomial}(5, \mu_2)$$

The binomial sample size n=5 is suggested by Pearson's estimate of $\theta = 4.89997$ and the fact that the largest cell count in Table 46.1 is 5.

The following DATA step creates a SAS data set from the data in Table 46.1.

```
data yeast;
   input count f;
   n = 5;
   datalines;
   0     213
   1     128
   2      37
   3      18
   4       3
   5       1
;
```

The two-component binomial model is fit with the FMM procedure with the following statements:

```
proc fmm data=yeast;
   model count/n =  / k=2;
   freq f;
run;
```

Because the events/trials syntax is used in the MODEL statement, PROC FMM defaults to the binomial distribution. The K=2 option specifies that the number of components is fixed and known to be two. The FREQ statement indicates that the data are grouped; for example, the first observation represents 213 squares on the haemacytometer where no yeast cells were found.

The "Model Information" and "Number of Observations" tables in Figure 46.1 convey that the fitted model is a two-component homogeneous binomial mixture with a logit link function. The mixture is *homogeneous* because there are no model effects in the MODEL statement and because both component distributions belong to the same distributional family. By default, PROC FMM estimates the model parameters by maximum likelihood.

Although only six observations are read from the data set, the data represent 400 observations (squares on the haemacytometer). Because a constant binomial sample size of 5 is assumed, the data represent 273 successes (finding a yeast cell) out of 2,000 Bernoulli trials.

Figure 46.1 Model Information for Yeast Cell Model

The FMM Procedure

Model Information	
Data Set	WORK.YEAST
Response Variable (Events)	count
Response Variable (Trials)	n
Frequency Variable	f
Type of Model	Homogeneous Mixture
Distribution	Binomial
Components	2
Link Function	Logit
Estimation Method	Maximum Likelihood

Number of Observations Read	6
Number of Observations Used	6
Sum of Frequencies Read	400
Sum of Frequencies Used	400
Number of Events	273
Number of Trials	2000

The estimated intercepts (on the logit scale) for the two binomial means are –2.2316 and –0.2974, respectively. These values correspond to binomial success probabilities of 0.09695 and 0.4262, respectively (Figure 46.2). The two components mix with probabilities 0.8799 and 1–0.8799 = 0.1201. These values are generally close to the values found by Pearson (1915) using infinite binomial series instead of binomial mass functions.

Figure 46.2 Maximum Likelihood Estimates

Parameter Estimates for Binomial Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|	Inverse Linked Estimate
1	Intercept	-2.2316	0.1522	-14.66	<.0001	0.09695
2	Intercept	-0.2974	0.3655	-0.81	0.4158	0.4262

Parameter Estimates for Mixing Probabilities

		Linked Scale			
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.8799	1.9913	0.5725	3.48	0.0005
2	0.1201	-1.9913			

To obtain fitted values and other observationwise statistics under the stipulated two-component model, you can add the OUTPUT statement to the previous PROC FMM run. The following statements request componentwise predicted values and the posterior probabilities:

```
proc fmm data=yeast;
   model count/n =  / k=2;
   freq f;
   output out=fmmout pred(components) posterior;
```

```
run;
data fmmout;
   set fmmout;
   PredCount_1 = post_1 * f;
   PredCount_2 = post_2 * f;
run;
proc print data=fmmout;
run;
```

The DATA step following the PROC FMM step computes the predicted cell counts in each component (Figure 46.3). The predicted means in the components, 0.48476 and 2.13099, are close to the values determined by Pearson (0.4983 and 2.2118), as are the predicted cell counts.

Figure 46.3 Predicted Cell Counts

Obs	count	f	n	Pred_1	Pred_2	Post_1	Post_2	PredCount_1	PredCount_2
1	0	213	5	0.096951	0.42620	0.98606	0.01394	210.030	2.9698
2	1	128	5	0.096951	0.42620	0.91089	0.08911	116.594	11.4058
3	2	37	5	0.096951	0.42620	0.59638	0.40362	22.066	14.9341
4	3	18	5	0.096951	0.42620	0.17598	0.82402	3.168	14.8323
5	4	3	5	0.096951	0.42620	0.02994	0.97006	0.090	2.9102
6	5	1	5	0.096951	0.42620	0.00444	0.99556	0.004	0.9956

Gosset, who was interested in small-sample statistical problems, investigated the use of prior knowledge in mathematical-statistical analysis—for example, deriving the sampling distribution of the correlation coefficient after having assumed a uniform prior distribution for the coefficient in the population (Aldrich 1997). Pearson also was not opposed to using prior information, especially uniform priors that reflect "equal distribution of ignorance." Fisher, on the other hand, would not have any of it: the best estimator in his opinion is obtained by a criterion that is absolutely independent of prior assumptions about probabilities of particular values. He objected to the insinuation that his derivations in the work on the correlation were deduced from Bayes theorem (Fisher 1921).

The preceding analysis of the yeast cell count data uses maximum likelihood methods that are free of prior assumptions. The following analysis takes instead a Bayesian approach, assuming a beta prior distribution for the binomial success probabilities and a uniform prior distribution for the mixing probabilities. The changes from the previous FMM run are the addition of the ODS GRAPHICS, PERFORMANCE, and BAYES statements and the SEED=12345 option.

```
ods graphics on;
proc fmm data=yeast seed=12345;
   model count/n = / k=2;
   freq f;
   performance cpucount=2;
   bayes;
run;
ods graphics off;
```

With ODS Graphics enabled, PROC FMM produces diagnostic trace plots for the posterior samples. Bayesian analyses are sensitive to the random number seed and thread count; the SEED= and CPUCOUNT= options ensure consistent results for the purposes of this example. The SEED=12345 option in the PROC FMM statement determines the random number seed for the random number generator used in the analysis. The

CPUCOUNT=2 option in the PERFORMANCE statement sets the number of available processors to two. The BAYES statement requests a Bayesian analysis.

The "Bayes Information" table in Figure 46.4 provides basic information about the Markov chain Monte Carlo sampler. Because the model is a homogeneous mixture, the FMM procedure applies an efficient conjugate sampling algorithm with a posterior sample size of 10,000 samples after a burn-in size of 2,000 samples. The "Prior Distributions" table displays the prior distribution for each parameter along with its mean and variance and the initial value in the chain. Notice that in this situation all three prior distributions reduce to a uniform distribution on (0, 1).

Figure 46.4 Basic Information about MCMC Sampler

The FMM Procedure

Bayes Information	
Sampling Algorithm	Conjugate
Data Augmentation	Latent Variable
Initial Values of Chain	Data Based
Burn-In Size	2000
MC Sample Size	10000
MC Thinning	1
Parameters in Sampling	3
Mean Function Parameters	2
Scale Parameters	0
Mixing Prob Parameters	1
Number of Threads	2

Prior Distributions

Component	Parameter	Distribution	Mean	Variance	Initial Value
1	**Success Probability**	Beta(1, 1)	0.5000	0.08333	0.1365
2	**Success Probability**	Beta(1, 1)	0.5000	0.08333	0.1365
1	**Probability**	Dirichlet(1, 1)	0.5000	0.08333	0.6180

The FMM procedure produces a log note for this model, indicating that the sampled quantities are not the linear predictors on the logit scale, but are the actual population parameters (on the data scale):

```
NOTE: Bayesian results for this model (no regressor variables,
      non-identity link) are displayed on the data scale, not the
      linked scale. You can obtain results on the linked (=linear)
      scale by requesting a Metropolis-Hastings sampling algorithm.
```

The trace panel for the success probability in the first binomial component is shown in Figure 46.5. Note that the first component in this Bayesian analysis corresponds to the second component in the MLE analysis. The plots in this panel can be used to diagnose the convergence of the Markov chain. If the chain has not converged, inferences cannot be made based on quantities derived from the chain. You generally look for the following:

- a smooth unimodal distribution of the posterior estimates in the density plot displayed on the lower right

- good mixing of the posterior samples in the trace plot at the top of the panel (good mixing is indicated when the trace traverses the support of the distribution and appears to have reached a stationary distribution)

Figure 46.5 Trace Panel for Success Probability in First Component

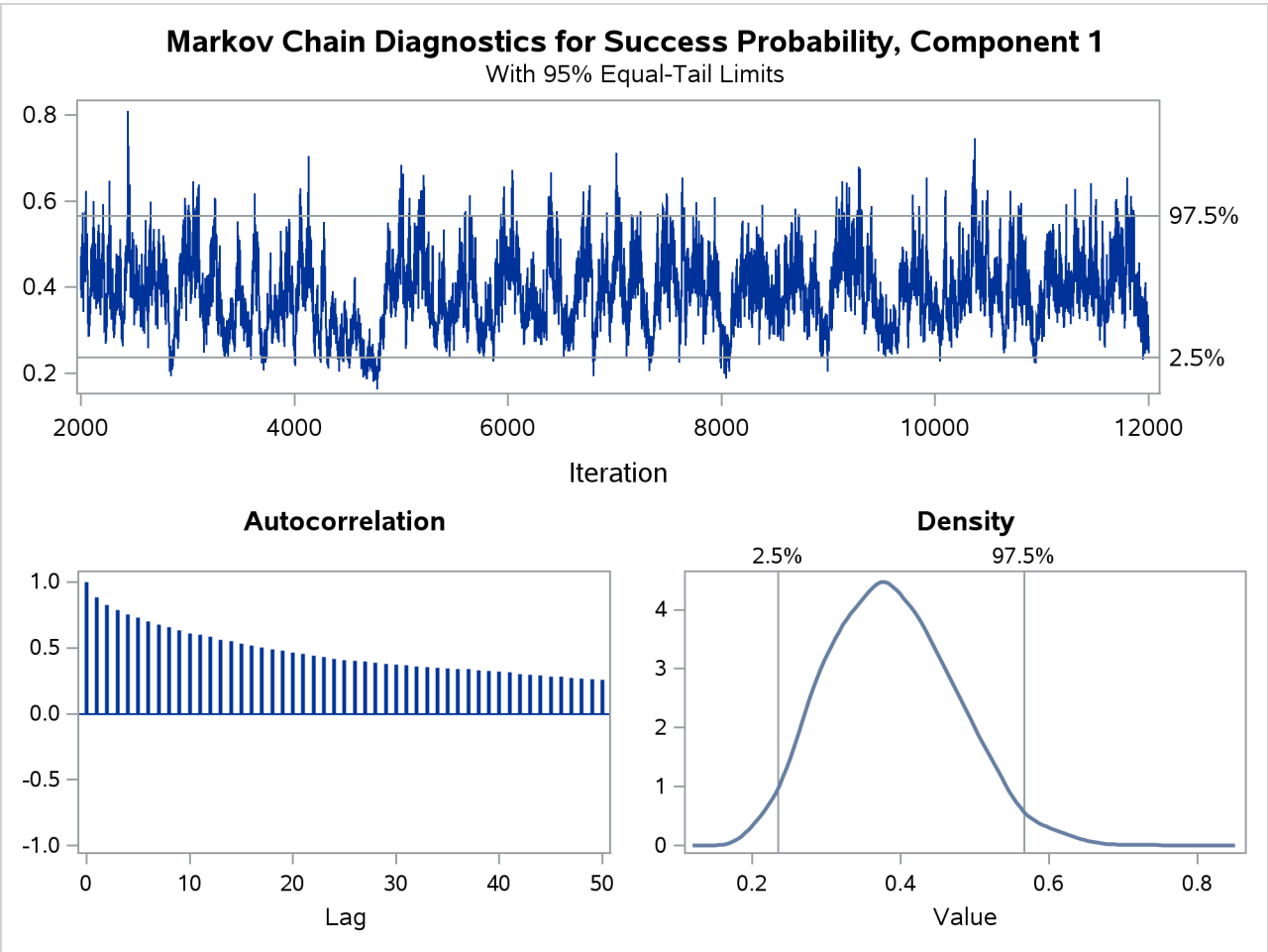

The autocorrelation plot in Figure 46.5 shows fairly high and sustained autocorrelation among the posterior estimates. While this is generally not a problem, you can affect the degree of autocorrelation among the posterior estimates by running a longer chain and thinning the posterior estimates; see the NMC= and THIN= options in the BAYES statement.

Both the trace plot and the density plot in Figure 46.5 are indications of successful convergence.

Figure 46.6 reports selected results that summarize the 10,000 posterior samples. The arithmetic means of the success probabilities in the two components are 0.3884 and 0.0905, respectively. The posterior mean of the mixing probability is 0.1771. These values are similar to the maximum likelihood parameter estimates in Figure 46.2 (after swapping components).

Figure 46.6 Summaries for Posterior Estimates

Posterior Summaries							
					Percentiles		
Component	Parameter	N	Mean	Standard Deviation	25	50	75
1	Success Probability	10000	0.3884	0.0861	0.3254	0.3835	0.4457
2	Success Probability	10000	0.0905	0.0162	0.0811	0.0923	0.1017
1	Probability	10000	0.1771	0.0978	0.1073	0.1534	0.2227

Posterior Intervals						
			Equal-Tail			
Component	Parameter	Alpha	Interval		HPD Interval	
1	Success Probability	0.050	0.2355	0.5663	0.2224	0.5494
2	Success Probability	0.050	0.0538	0.1171	0.0572	0.1187
1	Probability	0.050	0.0564	0.4311	0.0424	0.3780

Note that the standard errors in Figure 46.2 are not comparable to those in Figure 46.6, because the standard errors for the MLEs are expressed on the logit scale and the Bayesian estimates are expressed on the data scale. You can add the METROPOLIS option in the BAYES statement to sample the quantities on the logit scale.

The "Posterior Intervals" table in Figure 46.6 displays 95% credible intervals (equal-tail intervals and intervals of highest posterior density). It can be concluded that the component with the higher success probability contributes less than 40% to the process.

Modeling Zero-Inflation: Is It Better to Fish Poorly or Not to Have Fished at All?

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/fmmgs2.sas`.)

The following example shows how you can use PROC FMM to model data that contain more zero values than expected.

Many count data show an excess of zeros relative to the frequency of zeros expected under a reference model. An excess of zeros leads to overdispersion because the process is more variable than a standard count data model. Different mechanisms can lead to excess zeros. For example, suppose that the data are generated from two processes with different distribution functions—one process generates the zero counts, and the other process generates nonzero counts. In the vernacular of Cameron and Trivedi (1998), such a model is called a *hurdle* model. With a certain probability—the probability of a nonzero count—a hurdle is crossed, and events are being generated. Hurdle models are useful, for example, to model the number of doctor visits per year. Once the decision to see a doctor has been made—the hurdle has been overcome—a certain number of visits follow.

Hurdle models are closely related to zero-inflated models. Both can be expressed as two-component mixtures in which one component has a degenerate distribution at zero and the other component is a count model. In a hurdle model, the count model follows a zero-truncated distribution. In a zero-inflated model, the count

model has a nonzero probability of generating zeros. Formally, a zero-inflated model can be written as

$$\Pr(Y = y) = \pi p_1 + (1 - \pi) p_2(y, \mu)$$
$$p_1 = \begin{cases} 1 & y = 0 \\ 0 & \text{otherwise} \end{cases}$$

where $p_2(y, \mu)$ is a standard count model with mean μ and support $y \in \{0, 1, 2, \ldots\}$.

The following data illustrates the use of a zero-inflated model. In a survey of park attendees, randomly selected individuals were asked about the number of fish they caught in the last six months. Along with that count, the gender and age of each sampled individual was recorded. The following DATA step displays the data for the analysis:

```
data catch;
   input gender $ age count @@;
   datalines;
  F   54  18   M   37   0   F   48  12   M   27   0
  M   55   0   M   32   0   F   49  12   F   45  11
  M   39   0   F   34   1   F   50   0   M   52   4
  M   33   0   M   32   0   F   23   1   F   17   0
  F   44   5   M   44   0   F   26   0   F   30   0
  F   38   0   F   38   0   F   52  18   M   23   1
  F   23   0   M   32   0   F   33   3   M   26   0
  F   46   8   M   45   5   M   51  10   F   48   5
  F   31   2   F   25   1   M   22   0   M   41   0
  M   19   0   M   23   0   M   31   1   M   17   0
  F   21   0   F   44   7   M   28   0   M   47   3
  M   23   0   F   29   3   F   24   0   M   34   1
  F   19   0   F   35   2   M   39   0   M   43   6
;
```

At first glance, the prevalence of zeros in the DATA set is apparent. Many park attendees did not catch any fish. These zero counts are made up of two populations: attendees who do not fish and attendees who fish poorly. A zero-inflation mechanism thus appears reasonable for this application because a zero count can be produced by two separate distributions.

The following statements fit a standard Poisson regression model to these data. A common intercept is assumed for men and women, and the regression slope varies with gender.

```
proc fmm data=catch;
   class gender;
   model count = gender*age / dist=Poisson;
run;
```

Figure 46.7 displays information about the model and data set. The "Model Information" table conveys that the model is a single-component Poisson model (a Poisson GLM) and that parameters are estimated by maximum likelihood. There are two levels in the CLASS variable gender, with females preceding males.

Figure 46.7 Model Information and Class Levels in Poisson Regression

The FMM Procedure

Model Information	
Data Set	WORK.CATCH
Response Variable	count
Type of Model	Generalized Linear (GLM)
Distribution	Poisson
Components	1
Link Function	Log
Estimation Method	Maximum Likelihood

Class Level Information		
Class	**Levels**	**Values**
gender	2	F M

Number of Observations Read	52
Number of Observations Used	52

The "Fit Statistics" and "Parameter Estimates" tables from the maximum likelihood estimation of the Poisson GLM are shown in Figure 46.8. If the model is not overdispersed, the Pearson statistic should roughly equal the number of observations in the data set minus the number of parameters. With n=52, there is evidence of overdispersion in these data.

Figure 46.8 Fit Results in Poisson Regression

Fit Statistics	
-2 Log Likelihood	182.7
AIC (Smaller is Better)	188.7
AICC (Smaller is Better)	189.2
BIC (Smaller is Better)	194.6
Pearson Statistic	85.9573

Parameter Estimates for Poisson Model					
Effect	**gender**	**Estimate**	**Standard Error**	**z Value**	**Pr > |z|**
Intercept		-3.9811	0.5439	-7.32	<.0001
age*gender	F	0.1278	0.01149	11.12	<.0001
age*gender	M	0.1044	0.01224	8.53	<.0001

Suppose that the cause of overdispersion is zero-inflation of the count data. The following statements fit a zero-inflated Poisson model.

```
proc fmm data=catch;
   class gender;
   model count = gender*age / dist=Poisson ;
   model       +            / dist=Constant;
run;
```

There are two MODEL statements, one for each component of the mixture. Because the distributions are different for the components, you cannot specify the mixture model with a single MODEL statement. The first MODEL statement identifies the response variable for the model (count) and defines a Poisson model with intercept and gender-specific slopes. The second MODEL statement uses the continuation operator ("+") and adds a model with a degenerate distribution by using DIST=CONSTANT. Because the mass of the constant is placed by default at zero, the second MODEL statement adds a zero-inflation component to the model. It is sufficient to specify the response variable in one of the MODEL statements; you use the "=" sign in that statement to separate the response variable from the model effects.

Figure 46.9 displays the "Model Information" and "Optimization Information" tables for this run of the FMM procedure. The model is now identified as a zero-inflated Poisson (ZIP) model with two components, and the parameters continue to be estimated by maximum likelihood. The "Optimization Information" table shows that there are four parameters in the optimization (compared to three parameters in the Poisson GLM model). The four parameters correspond to three parameters in the mean function (intercept and two gender-specific slopes) and the mixing probability.

Figure 46.9 Model and Optimization Information in the ZIP Model

The FMM Procedure

Model Information	
Data Set	WORK.CATCH
Response Variable	count
Type of Model	Zero-inflated Poisson
Components	2
Estimation Method	Maximum Likelihood

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	4
Mean Function Parameters	3
Scale Parameters	0
Mixing Prob Parameters	1
Number of Threads	4

Results from fitting the ZIP model by maximum likelihood are shown in Figure 46.10. The –2 log likelihood and the information criteria suggest a much-improved fit over the single-component Poisson model (compare Figure 46.10 to Figure 46.8). The Pearson statistic is reduced by factor 2 compared to the Poisson model and suggests a better fit than the standard Poisson model.

Figure 46.10 Maximum Likelihood Results for the ZIP model

Fit Statistics	
-2 Log Likelihood	145.6
AIC (Smaller is Better)	153.6
AICC (Smaller is Better)	154.5
BIC (Smaller is Better)	161.4
Pearson Statistic	43.4467
Effective Parameters	4
Effective Components	2

Figure 46.10 *continued*

Parameter Estimates for Poisson Model						
Component	Effect	gender	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept		-3.5215	0.6448	-5.46	<.0001
1	age*gender	F	0.1216	0.01344	9.04	<.0001
1	age*gender	M	0.1056	0.01394	7.58	<.0001

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.6972	0.8342	0.4768	1.75	0.0802
2	0.3028	-0.8342			

The number of effective parameters and components shown in Figure 46.8 equals the values from Figure 46.9. This is not always the case because components can collapse (for example, when the mixing probability approaches zero or when two components have identical parameter estimates). In this example, both components and all four parameters are identifiable. The Poisson regression and the zero process mix, with a probability of approximately 0.6972 attributed to the Poisson component.

The FMM procedure enables you to fit some mixture models by Bayesian techniques. The following statements add the BAYES statement to the previous PROC FMM statements:

```
proc fmm data=catch seed=12345;
   class gender;
   model count = gender*age / dist=Poisson;
   model       +            / dist=constant;
   performance cpucount=2;
   bayes;
run;
```

The "Model Information" table indicates that the model parameters are estimated by Markov chain Monte Carlo techniques, and it displays the random number seed (Figure 46.11). This is useful if you did not specify a seed to identify the seed value that reproduces the current analysis. The "Bayes Information" table provides basic information about the Monte Carlo sampling scheme. The sampling method uses a data augmentation scheme to impute component membership and then the Gamerman (1997) algorithm to sample the component-specific parameters. The 2,000 burn-in samples are followed by 10,000 Monte Carlo samples without thinning.

Figure 46.11 Model, Bayes, and Prior Information in the ZIP Model

The FMM Procedure

Model Information	
Data Set	WORK.CATCH
Response Variable	count
Type of Model	Zero-inflated Poisson
Components	2
Estimation Method	Markov Chain Monte Carlo
Random Number Seed	12345

Figure 46.11 *continued*

Bayes Information	
Sampling Algorithm	Gamerman
Data Augmentation	Latent Variable
Initial Values of Chain	ML Estimates
Burn-In Size	2000
MC Sample Size	10000
MC Thinning	1
Parameters in Sampling	4
Mean Function Parameters	3
Scale Parameters	0
Mixing Prob Parameters	1
Number of Threads	2

Prior Distributions

Component	Effect	gender	Distribution	Mean	Variance	Initial Value
1	**Intercept**		Normal(0, 1000)	0	1000.00	-3.5215
1	**age*gender**	F	Normal(0, 1000)	0	1000.00	0.1216
1	**age*gender**	M	Normal(0, 1000)	0	1000.00	0.1056
1	**Probability**		Dirichlet(1, 1)	0.5000	0.08333	0.6972

The "Prior Distributions" table identifies the prior distributions, their parameters for the sampled quantities, and their initial values. The prior distribution of parameters associated with model effects is a normal distribution with mean 0 and variance 1,000. The prior distribution for the mixing probability is a Dirichlet(1,1), which is identical to a uniform distribution (Figure 46.11). Because the second mixture component is a degeneracy at zero with no associated parameters, it does not appear in the "Prior Distributions" table in Figure 46.11.

Figure 46.12 displays descriptive statistics about the 10,000 posterior samples. Recall from Figure 46.10 that the maximum likelihood estimates were –3.5215, 0.1216, 0.1056, and 0.6972, respectively. With this choice of prior, the means of the posterior samples are generally close to the MLEs in this example. The "Posterior Intervals" table displays 95% intervals of equal-tail probability and 95% intervals of highest posterior density (HPD) intervals.

Figure 46.12 Posterior Summaries and Intervals in the ZIP Model

Posterior Summaries

Component	Effect	gender	N	Mean	Standard Deviation	Percentiles 25	50	75
1	**Intercept**		10000	-3.5524	0.6509	-3.9922	-3.5359	-3.0875
1	**age*gender**	F	10000	0.1220	0.0136	0.1124	0.1218	0.1314
1	**age*gender**	M	10000	0.1058	0.0140	0.0961	0.1055	0.1153
1	**Probability**		10000	0.6938	0.0945	0.6293	0.6978	0.7605

Figure 46.12 *continued*

Posterior Intervals							
Component	Effect	gender	Alpha	Equal-Tail Interval		HPD Interval	
1	Intercept		0.050	-4.8693	-2.3222	-4.8927	-2.3464
1	age*gender	F	0.050	0.0960	0.1494	0.0961	0.1494
1	age*gender	M	0.050	0.0792	0.1339	0.0796	0.1341
1	Probability		0.050	0.5041	0.8688	0.5025	0.8666

You can generate trace plots for the posterior parameter estimates by enabling ODS Graphics:

```
ods graphics on;
ods select TADPanel;
proc fmm data=catch seed=12345;
   class gender;
   model count = gender*age / dist=Poisson;
   model       +            / dist=constant;
   performance cpucount=2;
   bayes;
run;
ods graphics off;
```

A separate trace panel is produced for each sampled parameter, and the panels for the gender-specific slopes are shown in Figure 46.13. There is good mixing in the chains: the modest autocorrelation that diminishes after about 10 successive samples. By default, the FMM procedure transfers the credible intervals for each parameter from the “Posterior Intervals” table to the trace plot and the density plot in the trace panel.

Figure 46.13 Trace Panels for Gender-Specific Slopes

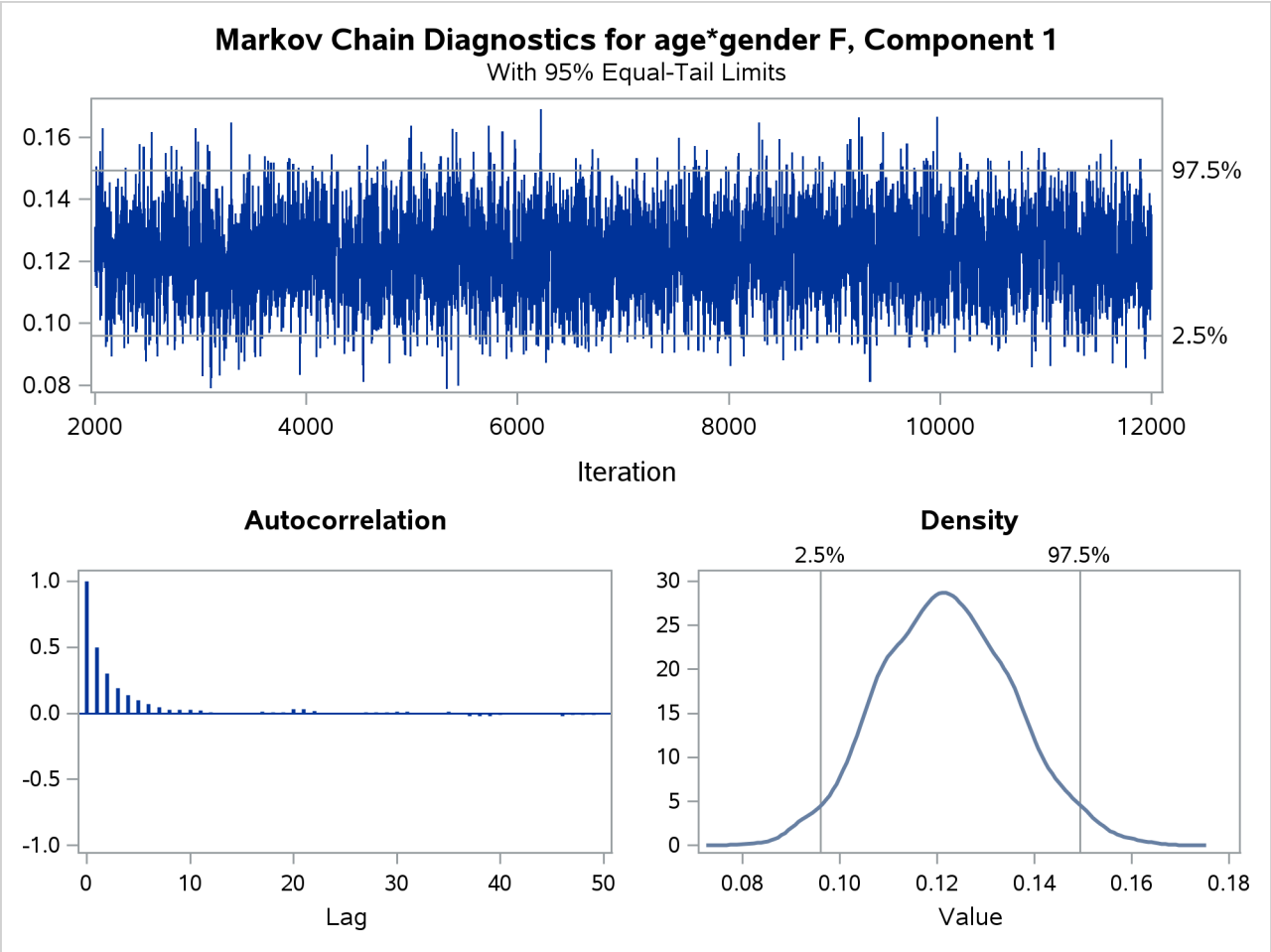

Figure 46.13 *continued*

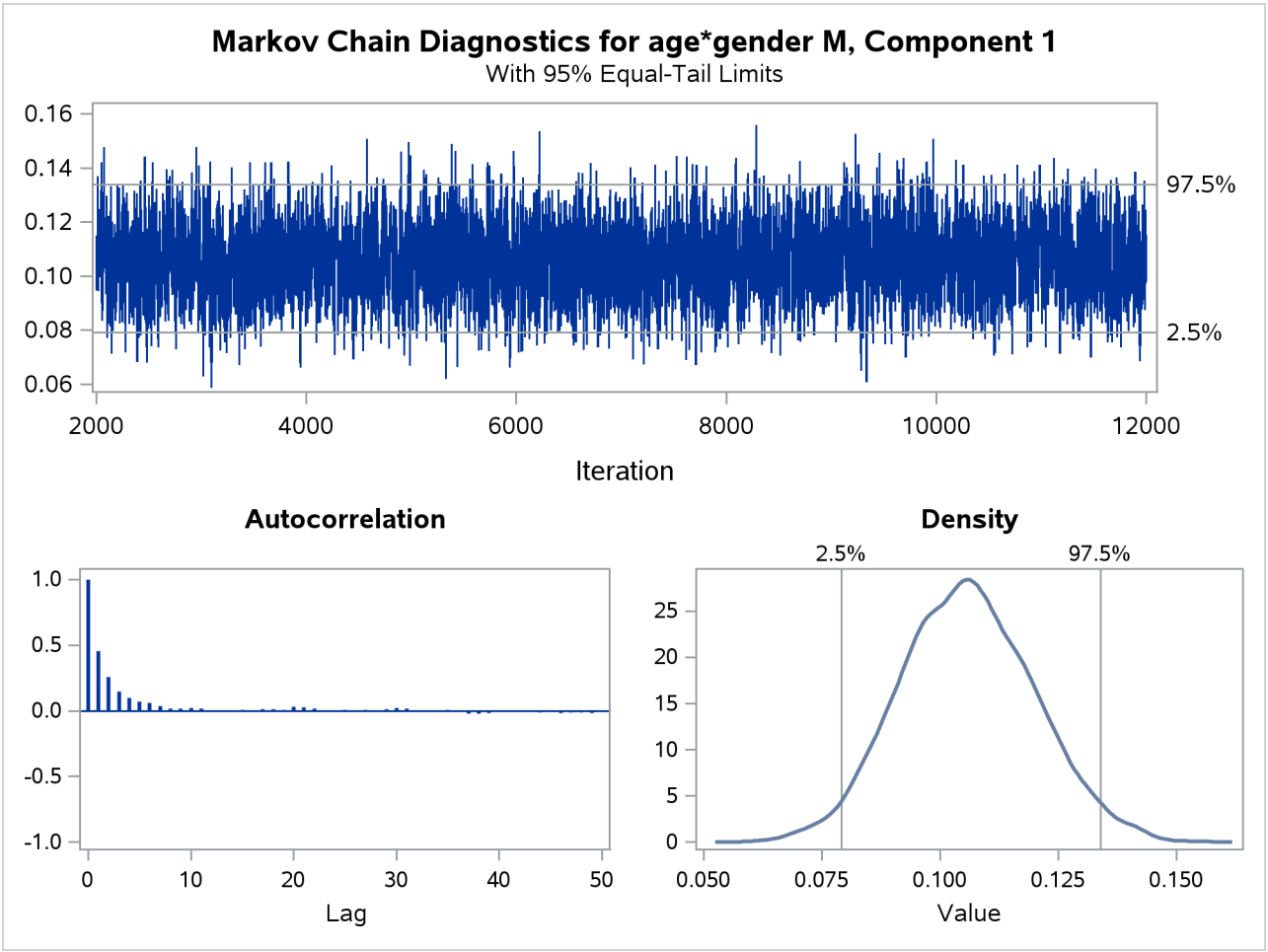

Looking for Multiple Modes: Are Galaxies Clustered?

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/fmmgs3.sas`.)

Mixture modeling is essentially a generalized form of one-dimensional cluster analysis. The following example shows how you can use PROC FMM to explore the number and nature of Gaussian clusters in univariate data.

Roeder (1990) presents data from the Corona Borealis sky survey with the velocities of 82 galaxies in a narrow slice of the sky. Cosmological theory suggests that the observed velocity of each galaxy is proportional to its distance from the observer. Thus, the presence of multiple modes in the density of these velocities could indicate a clustering of the galaxies at different distances.

The following DATA step re-creates the data set in Roeder (1990). The computed variable v represents the measured velocity in thousands of kilometers per second.

```
title "FMM Analysis of Galaxies Data";
data galaxies;
   input velocity @@;
   v = velocity / 1000;
   datalines;
9172  9350  9483  9558  9775  10227 10406 16084 16170 18419
18552 18600 18927 19052 19070 19330 19343 19349 19440 19473
19529 19541 19547 19663 19846 19856 19863 19914 19918 19973
19989 20166 20175 20179 20196 20215 20221 20415 20629 20795
20821 20846 20875 20986 21137 21492 21701 21814 21921 21960
22185 22209 22242 22249 22314 22374 22495 22746 22747 22888
22914 23206 23241 23263 23484 23538 23542 23666 23706 23711
24129 24285 24289 24366 24717 24990 25633 26960 26995 32065
32789 34279
;
```

Analysis of potentially multimodal data is a natural application of finite mixture models. In this case, the modeling is complicated by the question of the variance for each of the components. Using identical variances for each component could obscure underlying structure, but the additional flexibility granted by component-specific variances might introduce spurious features.

You can use PROC FMM to prepare analyses for equal and unequal variances and use one of the available fit statistics to compare the resulting models. You can use the model selection facility to explore models that have varying numbers of mixture components—say, from three to seven as investigated in Roeder (1990). The following statements select the best unequal-variance model by using Akaike's information criterion (AIC), which has a built-in penalty for model complexity:

```
title2 "Three to Seven Components, Unequal Variances";
ods graphics on;
proc fmm data=galaxies criterion=AIC;
   model v = / kmin=3 kmax=7;
run;
```

The KMIN= and KMAX= options indicate the smallest and largest number of components to consider. Figure 46.14, Figure 46.15, and Figure 46.16 show a selection of the output from this unequal variances model.

Figure 46.14 Model Selection for Galaxy Data Assuming Unequal Variances

FMM Analysis of Galaxies Data
Three to Seven Components, Unequal Variances

The FMM Procedure

Model Information	
Data Set	WORK.GALAXIES
Response Variable	v
Type of Model	Homogeneous Mixture
Distribution	Normal
Min Components	3
Max Components	7
Link Function	Identity
Estimation Method	Maximum Likelihood

Figure 46.14 *continued*

Component Evaluation for Mixture Models										
	Number of Components		Number of Parameters							
Model ID	Total	Eff.	Total	Eff.	-2 Log L	AIC	AICC	BIC	Pearson	Max Gradient
1	3	3	8	8	406.96	422.96	424.94	442.22	82.00	0.000024
2	4	4	11	11	406.96	428.96	432.74	455.44	82.00	0.00012
3	5	5	14	14	406.96	434.96	441.23	468.66	82.00	0.000048
4	6	6	17	17	406.96	440.96	450.53	481.88	82.00	0.000013
5	7	7	20	20	406.96	446.96	460.73	495.10	82.00	0.000092

The model with 3 components (ID=1) was selected as 'best' based on the AIC statistic.

Fit Statistics	
-2 Log Likelihood	407.0
AIC (Smaller is Better)	423.0
AICC (Smaller is Better)	424.9
BIC (Smaller is Better)	442.2
Pearson Statistic	82.0001
Effective Parameters	8
Effective Components	3

Parameter Estimates for Normal Model					
Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	9.7101	0.1597	60.80	<.0001
2	Intercept	33.0444	0.5322	62.09	<.0001
3	Intercept	21.4039	0.2597	82.41	<.0001
1	Variance	0.1785	0.09542		
2	Variance	0.8496	0.6937		
3	Variance	4.8567	0.8098		

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.0854	-2.3308	0.3959	-5.89	<.0001
2	0.0366	-3.1781	0.5893	-5.39	<.0001
3	0.8781	0			

Figure 46.15 Density Plot for Best (Three-Component) Model Assuming Unequal Variances

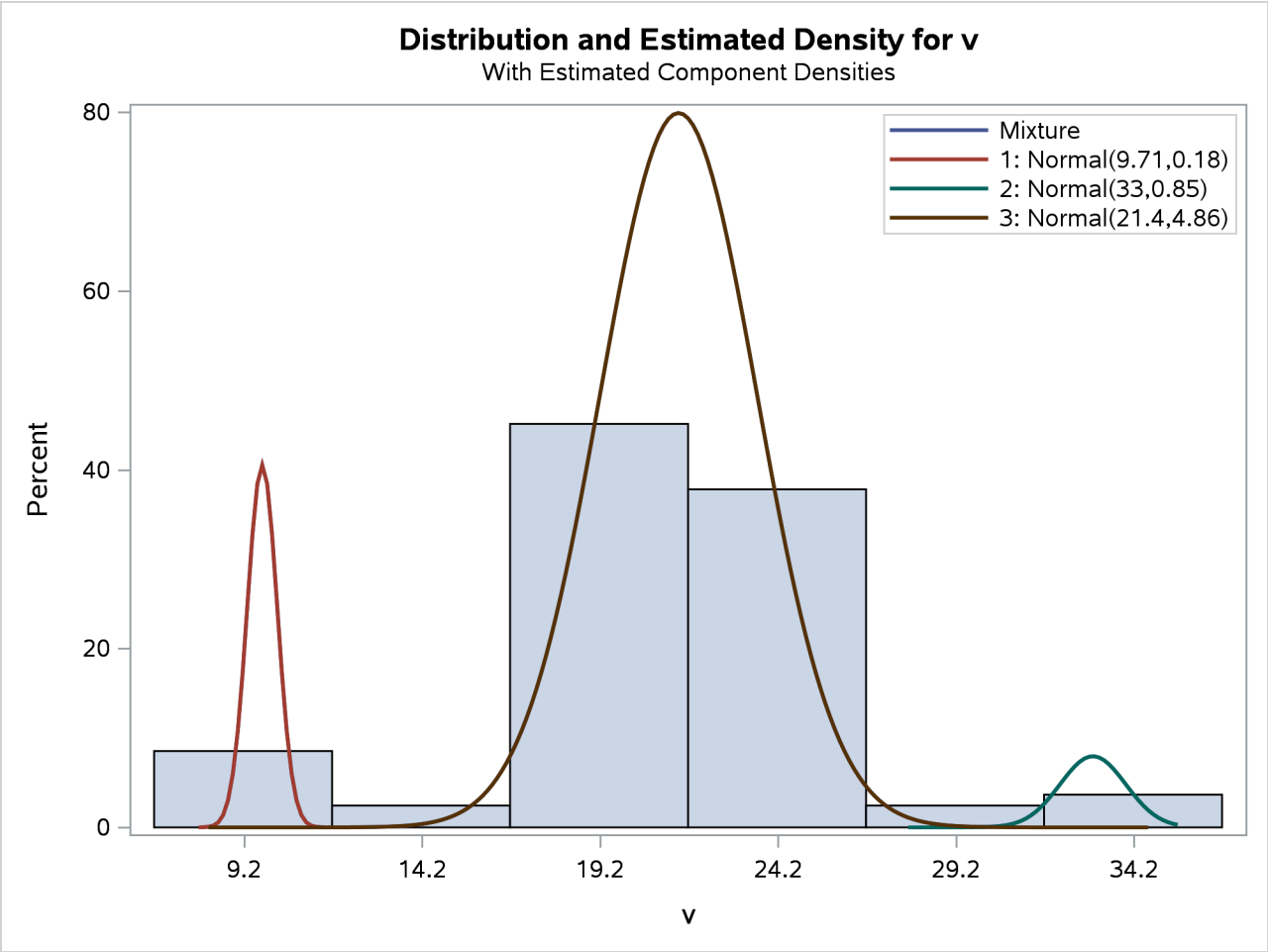

Figure 46.16 Criterion Panel Plot for Model Selection Assuming Unequal Variances

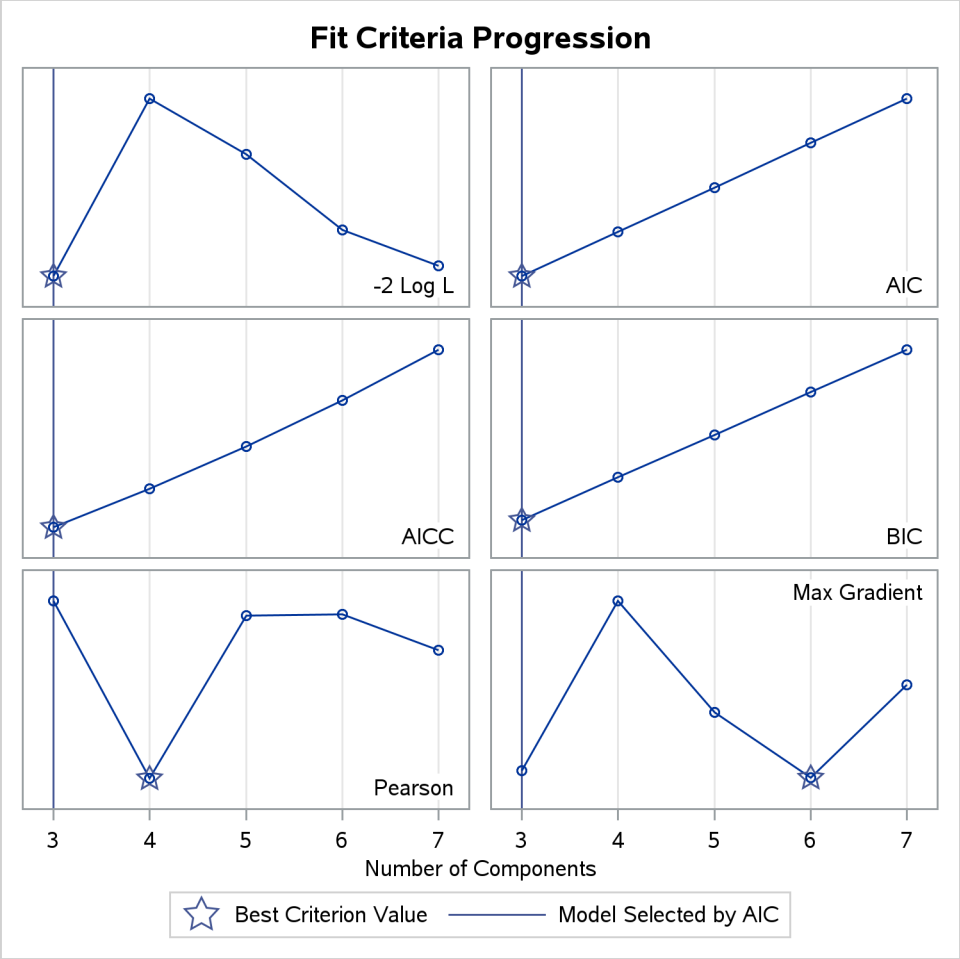

This example uses the AIC for model selection. Figure 46.16 shows the AIC and other model fit criteria for each of the fitted models.

To require that the separate components have identical variances, add the EQUATE=SCALE option in the MODEL statement:

```
title2 "Three to Seven Components, Equal Variances";
proc fmm data=galaxies criterion=AIC gconv=0;
   model v = / kmin=3 kmax=7  equate=scale;
run;
```

The GCONV= convergence criterion is turned off in this PROC FMM run to avoid the early stoppage of the iterations when the relative gradient changes little between iterations. Turning the criterion off usually ensures that convergence is achieved with a small absolute gradient of the objective function.

The output for equal variances is shown in Figure 46.17 and Figure 46.18.

Figure 46.17 Model Selection for Galaxy Data Assuming Equal Variances

FMM Analysis of Galaxies Data
Three to Seven Components, Equal Variances

The FMM Procedure

Model Information	
Data Set	WORK.GALAXIES
Response Variable	v
Type of Model	Homogeneous Mixture
Distribution	Normal
Min Components	3
Max Components	7
Link Function	Identity
Estimation Method	Maximum Likelihood

Component Evaluation for Mixture Models

Model ID	Number of Components Total	Number of Components Eff.	Number of Parameters Total	Number of Parameters Eff.	-2 Log L	AIC	AICC	BIC	Pearson	Max Gradient
1	**3**	**3**	**6**	**6**	478.74	490.74	491.86	505.18	82.00	1.197E-6
2	**4**	**4**	**8**	**8**	416.49	432.49	434.47	451.75	82.00	2.184E-6
3	**5**	**5**	**10**	**10**	416.49	436.49	439.59	460.56	82.00	4.313E-6
4	**6**	**6**	**12**	**12**	416.49	440.49	445.02	469.37	82.00	6.294E-6
5	**7**	**7**	**14**	**14**	416.49	444.49	450.76	478.19	82.00	4.885E-6

The model with 4 components (ID=2) was selected as 'best' based on the AIC statistic.

Fit Statistics	
-2 Log Likelihood	416.5
AIC (Smaller is Better)	432.5
AICC (Smaller is Better)	434.5
BIC (Smaller is Better)	451.7
Pearson Statistic	82.0000
Effective Parameters	8
Effective Components	4

Parameter Estimates for Normal Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	**Intercept**	23.5058	0.3460	67.93	<.0001
2	**Intercept**	33.0440	0.7610	43.42	<.0001
3	**Intercept**	20.0086	0.3029	66.06	<.0001
4	**Intercept**	9.7103	0.4981	19.50	<.0001
1	**Variance**	1.7354	0.3905		
2	**Variance**	1.7354	0.3905		
3	**Variance**	1.7354	0.3905		
4	**Variance**	1.7354	0.3905		

Figure 46.17 *continued*

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.3503	1.4118	0.4497	3.14	0.0017
2	0.0366	-0.8473	0.6901	-1.23	0.2195
3	0.5277	1.8216	0.4205	4.33	<.0001
4	0.0854	0			

Figure 46.18 Density Plot for Best (Four-Component) Model Assuming Equal Variances

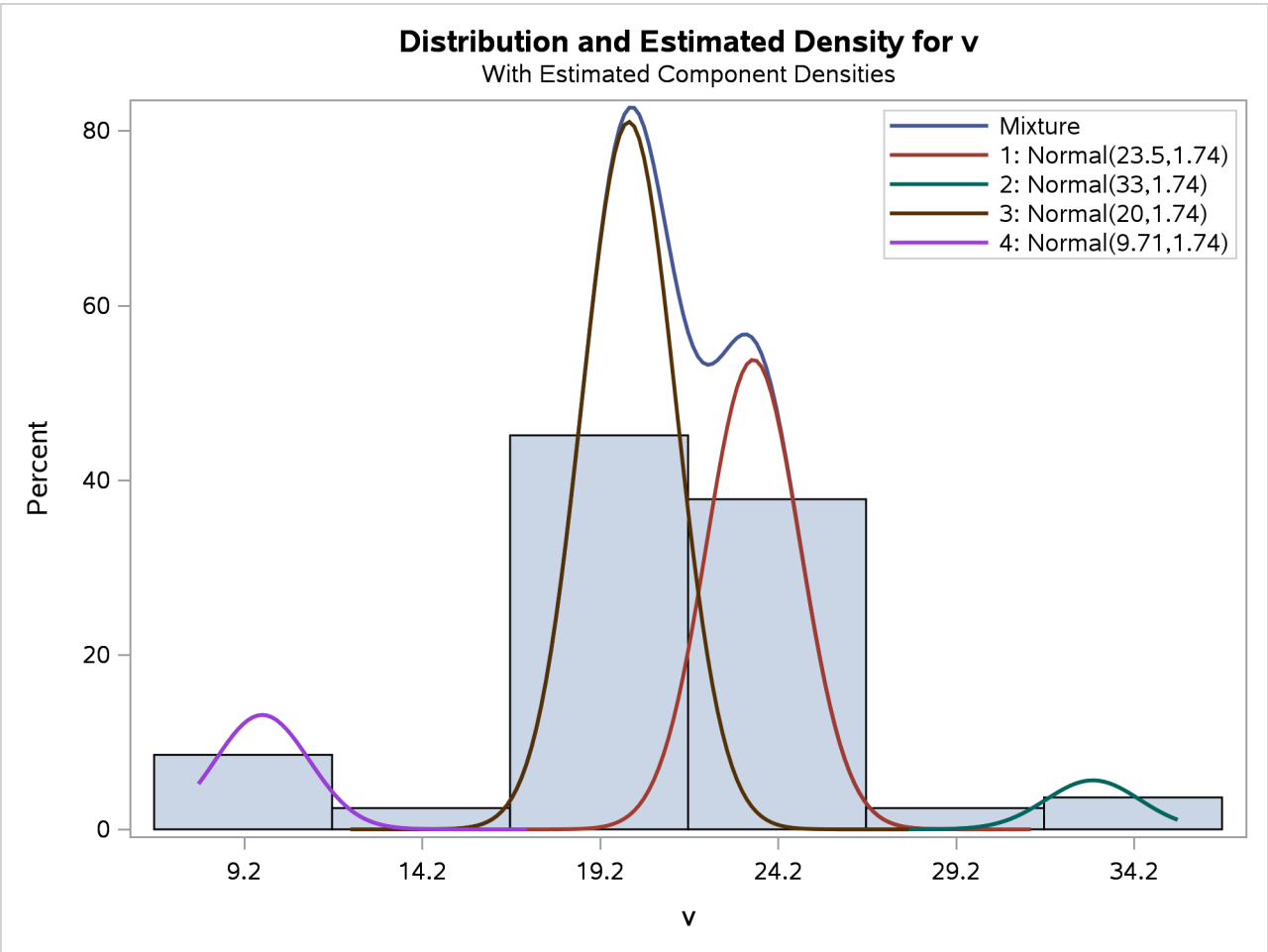

Not surprisingly, the two variance specifications produce different optimal models. The unequal variance specification favors a three-component model while the equal variance specification favors a four-component model. Comparison of the AIC fit statistics, 423.0 and 432.5, indicates that the three-component, unequal variance model provides the best overall fit.

Comparison with Roeder's Method

It is important to note that Roeder's original analysis proceeds in a different manner than the finite mixture modeling presented here. The technique presented by Roeder first develops a "best" range of scale parameters based on a specific criterion. Roeder then uses fixed scale parameters taken from this range to develop optimal equal-scale Gaussian mixture models.

You can reproduce Roeder's point estimate for the density by specifying a five-component Gaussian mixture. In addition, use the EQUATE=SCALE option in the MODEL statement and a RESTRICT statement fixing the first component's scale parameter at 0.9025 (Roeder's $h = 0.95$, scale$= h^2$). The combination of these options produces a mixture of five Gaussian components, each with variance 0.9025. The following statements conduct this analysis:

```
title2 "Five Components, Equal Variances = 0.9025";
proc fmm data=galaxies;
   model v = / K=5 equate=scale;
   restrict int 0 (scale 1) = 0.9025;
run;
```

The output is shown in Figure 46.19 and Figure 46.20.

Figure 46.19 Reproduction of Roeder's Five-Component Analysis of Galaxy Data

FMM Analysis of Galaxies Data
Five Components, Equal Variances = 0.9025

The FMM Procedure

Model Information	
Data Set	WORK.GALAXIES
Response Variable	v
Type of Model	Homogeneous Mixture
Distribution	Normal
Components	5
Link Function	Identity
Estimation Method	Maximum Likelihood

Fit Statistics	
-2 Log Likelihood	412.2
AIC (Smaller is Better)	430.2
AICC (Smaller is Better)	432.7
BIC (Smaller is Better)	451.9
Pearson Statistic	82.5549
Effective Parameters	9
Effective Components	5

Linear Constraints at Solution		
k = 1		**Constraint Active**
Variance =	0.90	Yes

Figure 46.19 *continued*

Parameter Estimates for Normal Model					
Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	26.3266	0.7778	33.85	<.0001
2	Intercept	33.0443	0.5485	60.25	<.0001
3	Intercept	9.7101	0.3591	27.04	<.0001
4	Intercept	23.0295	0.2294	100.38	<.0001
5	Intercept	19.7187	0.1784	110.55	<.0001
1	Variance	0.9025	0		
2	Variance	0.9025	0		
3	Variance	0.9025	0		
4	Variance	0.9025	0		
5	Variance	0.9025	0		

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.0397	-2.4739	0.7084	-3.49	0.0005
2	0.0366	-2.5544	0.6016	-4.25	<.0001
3	0.0854	-1.7071	0.4141	-4.12	<.0001
4	0.3678	-0.2466	0.2699	-0.91	0.3609
5	0.4706	0			

Figure 46.20 Density Plot for Roeder's Analysis

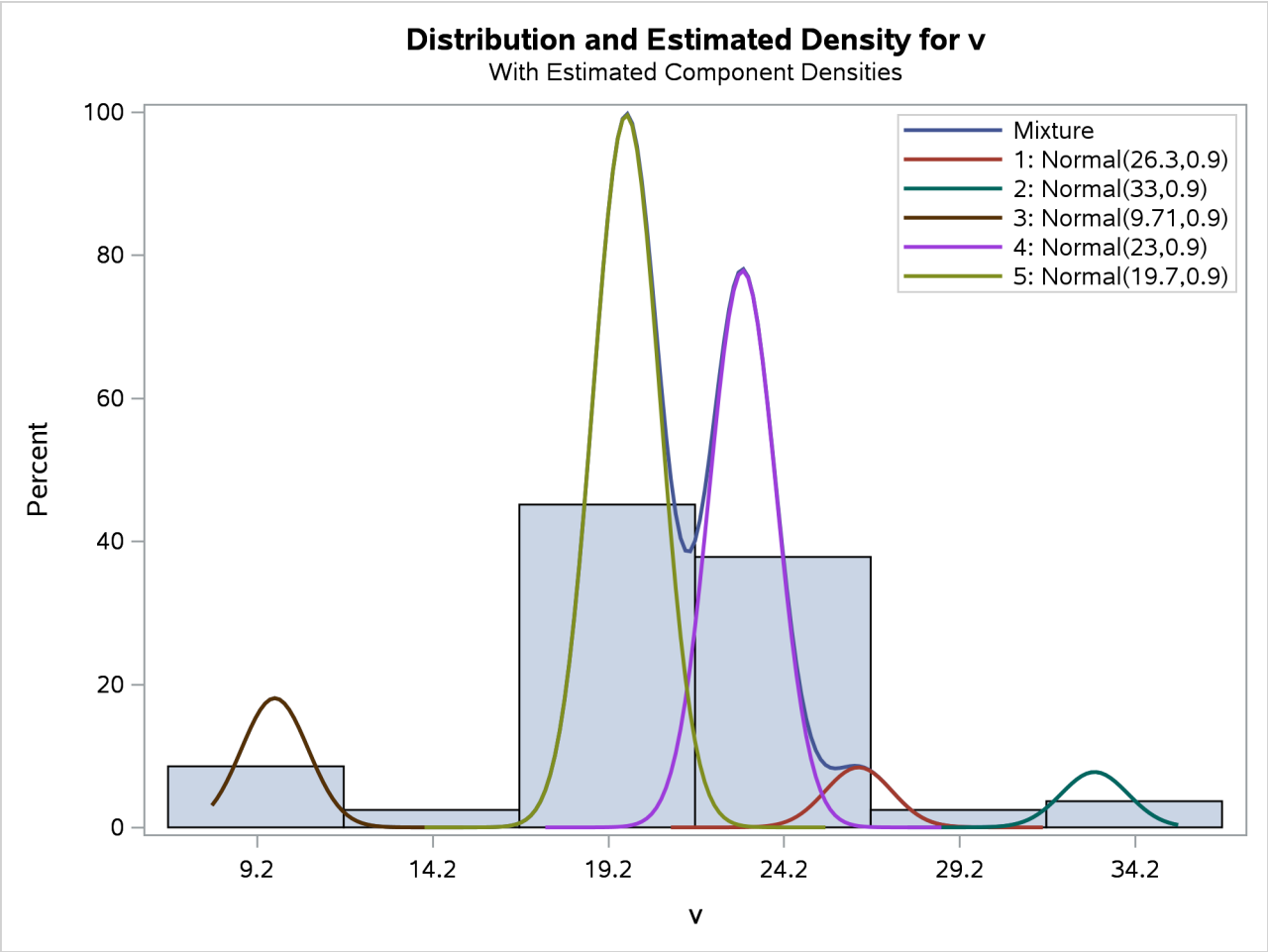

Syntax: FMM Procedure

The following statements are available in the FMM procedure:

```
PROC FMM <options> ;
    BAYES bayes-options ;
    BY variables ;
    CLASS variables </ TRUNCATE> ;
    FREQ variable ;
    ID variables ;
    MODEL response<(response-options)> = <effects> </ model-options> ;
    MODEL events/trials = <effects> </ model-options> ;
    MODEL              + <effects> </ model-options> ;
    OUTPUT <OUT=SAS-data-set>
           <keyword<(keyword-options)> <=name>> ...
           <keyword<(keyword-options)> <=name>> </ options> ;
    PERFORMANCE performance-options ;
    PROBMODEL <effects> </ probmodel-options> ;
    RANDSTART <randstart-options> ;
    RESTRICT <'label'> constraint-specification <, ..., constraint-specification>
             <operator <value>> </ option> ;
    WEIGHT variable ;
```

The PROC FMM statement and at least one MODEL statement is required. The CLASS, RESTRICT, and MODEL statements can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statements. The RESTRICT statements must appear after the MODEL statements.

PROC FMM Statement

```
PROC FMM <options> ;
```

The PROC FMM statement invokes the FMM procedure. Table 46.2 summarizes the *options* available in the PROC FMM statement. These and other *options* in the PROC FMM statement are then described fully in alphabetical order.

Table 46.2 PROC FMM Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
EXCLUSION=	Specifies how the procedure responds to support violations in the data
NAMELEN=	Specifies the length of effect names
ORDER=	Determines the sort order of CLASS variables
SEED=	Specifies the random number seed for analyses that require random number draws

Table 46.2 *continued*

Option	Description
Displayed Output	
COMPONENTINFO	Displays information about the mixture components
CORR	Displays the asymptotic correlation matrix of the maximum likelihood parameter estimates or the empirical correlation matrix of the Bayesian posterior estimates
COV	Displays the asymptotic covariance matrix of the maximum likelihood parameter estimates or the empirical covariance matrix of the Bayesian posterior estimates
COVI	Displays the inverse of the covariance matrix of the parameter estimates
FITDETAILS	Displays fit information for all examined models
ITDETAILS	Adds estimates and gradients to the "Iteration History" table
NOCLPRINT	Suppresses the "Class Level Information" table completely or partially
NOITPRINT	Suppresses the "Iteration History Information" table
NOPRINT	Suppresses tabular and graphical output
PARMSTYLE=	Specifies how parameters are displayed in ODS tables
PLOTS	Produces ODS statistical graphics
Computational Options	
CRITERION=	Specifies the criterion used in model selection
NOCENTER	Prevents centering and scaling of the regressor variables
PARTIAL=	Specifies a variable that defines a partial classification
Options Related to Optimization	
ABSCONV=	Tunes an absolute function convergence criterion
ABSFCONV=	Tunes an absolute function difference convergence criterion
ABSGCONV=	Tunes the absolute gradient convergence criterion
FCONV=	Specifies a relative function convergence criterion that is based on a relative change of the function value
FCONV2=	Specifies a relative function convergence criterion that is based on a predicted reduction of the objective function
GCONV=	Tunes the relative gradient convergence criterion
MAXITER=	Specifies the maximum number of iterations in any optimization
MAXFUNC=	Specifies the maximum number of function evaluations in any optimization
MAXTIME=	Specifies the upper limit of CPU time in seconds for any optimization
MINITER=	Specifies the minimum number of iterations in any optimization
TECHNIQUE=	Selects the optimization technique
Singularity Tolerances	
INVALIDLOGL=	Tunes the value assigned to an invalid component log likelihood
SINGCHOL=	Tunes singularity for Cholesky decompositions
SINGRES=	Tunes singularity for the residual variance

Table 46.2 *continued*

Option	Description
SINGULAR=	Tunes general singularity criterion
ZEROPROB=	Tunes component weight threshold for number of effective components

You can specify the following *options* in the PROC FMM statement.

ABSCONV=*r*

ABSTOL=*r*

specifies an absolute function convergence criterion. For minimization, the termination criterion is $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of r is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

ABSFCONV=*r*

ABSFTOL=*r*

specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, the termination criterion is a small change of the function value in successive iterations:

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}(k)$ is defined as the vertex with the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default value is r=0.

ABSGCONV=*r*

ABSGTOL=*r*

specifies an absolute gradient convergence criterion. The termination criterion is a small maximum absolute gradient element:

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $g_j(\cdot)$ is the gradient of the objective function with respect to the jth parameter. This criterion is not used by the NMSIMP technique. The default value is r=1E–5.

COMPONENTINFO

COMPINFO

CINFO

produces a table with additional details about the fitted model components.

COV

produces the covariance matrix of the parameter estimates. For maximum likelihood estimation, this matrix is based on the inverse (projected) Hessian matrix. For Bayesian estimation, it is the empirical covariance matrix of the posterior estimates. The covariance matrix is shown for all parameters, even if they did not participate in the optimization or sampling.

COVI

produces the inverse of the covariance matrix of the parameter estimates. For maximum likelihood estimation, the covariance matrix is based on the inverse (projected) Hessian matrix. For Bayesian estimation, it is the empirical covariance matrix of the posterior estimates. This matrix is then inverted by sweeping, and rows and columns that correspond to linear dependencies or singularities are zeroed.

CORR

produces the correlation matrix of the parameter estimates. For maximum likelihood estimation this matrix is based on the inverse (projected) Hessian matrix. For Bayesian estimation, it is based on the empirical covariance matrix of the posterior estimates.

CRITERION=*keyword*

CRIT=*keyword*

specifies the criterion by which the FMM procedure ranks models when multiple models are evaluated during maximum likelihood estimation. You can choose from the following criteria to rank models by specifying the appropriate *keyword*:

AIC	uses Akaike's information criterion.
AICC	uses the bias-corrected AIC criterion.
BIC	uses the Bayesian information criterion.
GRADIENT	uses the largest element of the gradient (in absolute value).
LOGL | LL	uses the mixture log likelihood.
PEARSON	uses the Pearson statistic.

The default for maximum likelihood estimation is CRITERION=BIC.

In Bayesian model selection, you can choose from the following criteria to rank models:

DIC	uses the deviance information criterion.
LOGL | LL	uses the mixture log likelihood

The default for Bayesian estimation is CRITERION=DIC.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC FMM. The default is the most recently created data set.

EXCLUSION=NONE | ANY | ALL

EXCLUDE=NONE | ANY | ALL

specifies how the FMM procedure handles support violations of observations. For example, in a mixture of two Poisson variables, negative response values are not possible. However, in a mixture of a Poisson and a normal variable, negative values are possible, and their likelihood contribution to the Poisson component is zero. An observation that violates the support of one component distribution of

the model might be a valid response with respect to one or more other component distributions. This requires some nuanced handling of support violations in mixture models.

The default exclusion technique, EXCLUSION=ALL, removes an observation from the analysis only if it violates the support of all component distributions. The other extreme, EXCLUSION=NONE, permits an observation into the analysis regardless of support violations. EXCLUSION=ANY removes observations from the analysis if the response violates the support of any component distributions. In the single-component case, EXCLUSION=ALL and EXCLUSION=ANY are identical.

FCONV=*r*

FTOL=*r*

specifies a relative function convergence criterion that is based on the relative change of the function value. For all techniques except NMSIMP, PROC FMM terminates when there is a small relative change of the function value in successive iterations:

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex with the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default is $r = 10^{-\mathrm{FDIGITS}}$, where FDIGITS is by default $-\log_{10}\{\epsilon\}$, and ϵ is the machine precision.

FCONV2=*r*

FTOL2=*r*

specifies a relative function convergence criterion that is based on the predicted reduction of the objective function. For all techniques except NMSIMP, the termination criterion is a small predicted reduction

$$df^{(k)} \approx f(\boldsymbol{\theta}^{(k)}) - f(\boldsymbol{\theta}^{(k)} + \mathbf{s}^{(k)})$$

of the objective function. The predicted reduction

$$\begin{aligned} df^{(k)} &= -\mathbf{g}^{(k)\prime}\mathbf{s}^{(k)} - \frac{1}{2}\mathbf{s}^{(k)\prime}\mathbf{H}^{(k)}\mathbf{s}^{(k)} \\ &= -\frac{1}{2}\mathbf{s}^{(k)\prime}\mathbf{g}^{(k)} \\ &\leq r \end{aligned}$$

is computed by approximating the objective function f by the first two terms of the Taylor series and substituting the Newton step:

$$\mathbf{s}^{(k)} = -[\mathbf{H}^{(k)}]^{-1}\mathbf{g}^{(k)}$$

For the NMSIMP technique, the termination criterion is a small standard deviation of the function values of the $n+1$ simplex vertices $\boldsymbol{\theta}_l^{(k)}$, $l = 0, \ldots, n$,

$$\sqrt{\frac{1}{n+1}\sum_l \left[f(\boldsymbol{\theta}_l^{(k)}) - \overline{f}(\boldsymbol{\theta}^{(k)})\right]^2} \leq r$$

where $\overline{f}(\boldsymbol{\theta}^{(k)}) = \frac{1}{n+1}\sum_l f(\boldsymbol{\theta}_l^{(k)})$. If there are n_{act} boundary constraints active at $\boldsymbol{\theta}^{(k)}$, the mean and standard deviation are computed only for the $n + 1 - n_{act}$ unconstrained vertices.

The default value is r = 1E–6 for the NMSIMP technique and r = 0 otherwise.

FITDETAILS

requests that the "Optimization Information," "Iteration History," "Convergence Status," and "Fit Statistics" tables be produced for all optimizations when models with different number of components are evaluated. For example, the following statements fit a binomial regression model with up to three components and produces fit and optimization information for all three:

```
proc fmm fitdetails;
   model y/n  = x / kmax=3;
run;
```

Without the FITDETAILS option, only the "Fit Statistics" table for the selected model is displayed.

In Bayesian estimation, the FITDETAILS option displays the following tables for each model that the procedure fits: "Bayes Information," "Iteration History," "Prior Information," "Fit Statistics," "Posterior Summaries," "Posterior Intervals," and any requested diagnostics tables. The "Iteration History" table appears only if the BAYES statement includes the INITIAL=MLE option.

Without the FITDETAILS option, these tables are listed only for the selected model.

GCONV=*r*

GTOL=*r*

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, the termination criterion is a small normalized predicted function reduction:

$$\frac{\mathrm{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathrm{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathrm{g}(\cdot)$ is the gradient. For the CONGRA technique (where a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathrm{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathrm{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathrm{g}(\boldsymbol{\psi}^{(k)}) - \mathrm{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \, |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The default value is r = 1E–8.

HESSIAN

displays the Hessian matrix of the model. This option is not available for Bayesian estimation.

INVALIDLOGL=*r*

specifies the value assumed by the FMM procedure if a log likelihood cannot be computed (for example, because the value of the response variable falls outside of the response distribution's support). The default value is –1E20.

ITDETAILS

adds parameter estimates and gradients to the "Iteration History" table. If the FMM procedure centers or scales the model variables (or both), the parameter estimates and gradients reported during the iteration refer to that scale. You can suppress centering and scaling with the NOCENTER option.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, and NEWRAP: 125
- QUANEW and DBLDOG: 500
- CONGRA: 1000
- NMSIMP: 3000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that is specified by the MAXFUNC= option. You can choose the optimization technique with the TECHNIQUE= option.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, and NEWRAP: 50
- QUANEW and DBLDOG: 200
- CONGRA: 400
- NMSIMP: 1000

These default values also apply when *n* is specified as a missing value. You can choose the optimization technique with the TECHNIQUE= option.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The time is checked only at the end of each iteration. Therefore, the actual run time might be longer than the specified time. By default, CPU time is not limited.

MINITER=*n*

MINIT=*n*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

NAMELEN=*number*
specifies the length to which long effect names are shortened. The default and minimum value is 20.

NOCENTER
requests that regressor variables not be centered or scaled. By default the FMM procedure centers and scales columns of the **X** matrix if the models contain intercepts. If NOINT options in MODEL statements are in effect, the columns of **X** are scaled but not centered. Centering and scaling can help with the stability of estimation and sampling algorithms. The FMM procedure does not produce a table of the centered and scaled coefficients and provides no user control over the type of centering and scaling that is applied. The NOCENTER option turns any centering and scaling off and processes the raw values of the continuous variables.

NOCLPRINT< =*number* **>**
suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying a *number* helps to reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOITPRINT
suppresses the display of the "Iteration History Information" table.

NOPRINT
suppresses the normal display of tabular and graphical results. The NOPRINT option is useful when you want to create only one or more output data sets with the procedure. This option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

ORDER=*order-type*
specifies the sort order for the levels of CLASS variables. This ordering determines which parameters in the model correspond to each level in the data.

You can specify the following values for *order-type*:

DATA
sorts the levels by order of appearance in the input data set.

FORMATTED
sorts the levels by external formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value.

FREQ
sorts the levels by descending frequency count; levels with the most observations come first in the order.

INTERNAL
sorts the levels by unformatted value.

FREQDATA
sorts the levels by order of descending frequency count, and within counts by order of appearance in the input data set when counts are tied.

FREQFORMATTED
sorts the levels by order of descending frequency count, and within counts by formatted value (as above) when counts are tied.

FREQINTERNAL
sorts the levels by order of descending frequency count, and within counts by unformatted value when counts are tied.

When the default ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. To order numeric class levels with no explicit format by their BEST12. formatted values, you can specify this format explicitly for the CLASS variables.

When FORMATTED and INTERNAL values are involved, the sort order is machine-dependent.

When the response variable appears in a CLASS statement, the ORDER= option in the PROC FMM statement applies to its sort order. For example, in the following statements the sort order of the wheeze variable is determined by the order of appearance in the input data set because the response variable appears in the CLASS statement:

```
proc fmm order=data;
   class city wheeze;
   model wheeze = city age / dist=binary s;
run;
```

However, in the following statements the sort order of the wheeze variable is determined by the formatted value (the default *response-option* in the MODEL statement):

```
proc fmm order=data;
   class city;
   model wheeze = city age / dist=binary s;
run;
```

The ORDER= option in the PROC FMM statement has no effect on the sort order of the wheeze variable because it does not appear in the CLASS statement.

When you specify a *response-option* in the MODEL statement, it overrides the ORDER= option in the PROC FMM statement.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARMSTYLE=EFFECT | LABEL

specifies the display style for parameters and effects. The FMM procedure can display parameters in two styles:

EFFECT displays parameters and effects in the EFFECT style (which is used by the MIXED and GLIMMIX procedure, for example), which identifies a parameter with an "Effect" column and adds separate columns for the CLASS variables in the model.

LABEL displays parameters and effects in the LABEL style, which creates one column, named Parameter, that combines the relevant information about a parameter into a single column. If your model contains multiple CLASS variables, this style might use space more economically.

The EFFECT style is the default for models that contain effects; otherwise the LABEL style is used (for example, in homogeneous mixtures). You can change the display style by using the PARMSTYLE= option. Regardless of the display style, ODS output data sets that contain information about parameter estimates contain columns for both styles.

PARTIAL=*variable*

MEMBERSHIP=*variable*

specifies a variable in the input data set that identifies component membership. You can specify missing values for observations whose component membership is undetermined; this is known as a partial classification (McLachlan and Peel 2000, p. 75). For observations with known membership, the likelihood contribution is no longer a mixture. If observation i is known to be a member of component m, then its log likelihood contribution is

$$\log\left\{\pi_m(\mathbf{z},\boldsymbol{\alpha}_m)\ \ p_m(y;\mathbf{x}_m'\boldsymbol{\beta}_m,\phi_m)\right\}$$

Otherwise, if membership is undetermined, it is

$$\log\left\{\sum_{j=1}^{k}\ \pi_j(\mathbf{z},\boldsymbol{\alpha}_j)p_j(y;\mathbf{x}_j'\boldsymbol{\beta}_j,\phi_j)\right\}$$

The *variable* that you specify in this option can be numeric or character. In case of a character variable, the *variable* must also appear in the CLASS statement. If it also appears in the CLASS statement, the membership assignment is made on the basis of the levelized values of the variable, as shown in the "Class Level Information" table. If the *variable* is numeric, any nonmissing values of the *variable* must take values in the range $1, 2, \ldots, k$, where k is the number of mixture components. Invalid *variable* values are ignored.

In a model in which label switching is a problem, you can sometimes avoid switching by assigning just a few observations to categories. For example, in a three-component model, switching might be prevented if you assign the observation that has the smallest response value to the first component and the observation that has the largest response value to the last component.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the graphical output that is produced through ODS Graphics.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc fmm data=yeast seed=12345;
   model count/n = / k=2;
   freq f;
   performance cpucount=2;
   bayes;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

Global Plot Options

The *global-plot-options* apply to all relevant plots generated by the FMM procedure. The *global-plot-options* supported by the FMM procedure are as follows:

UNPACKPANEL

UNPACK

displays each plot separately. (By default, some plots can appear together in a single panel.)

ONLY

produces only the specified plots. This option is useful if you do not want the procedure to generate all default plots, but only the ones specified.

Specific Plot Options

The following listing describes the specific plots and their *options*.

ALL

requests that all plots appropriate for the analysis be produced.

NONE

requests that no ODS graphics be produced.

DENSITY < (*density-options*) >

requests a plot of the data histogram and mixture density function. This is a default plot in models that have no effects in the MODEL statements and is available only in these models. Furthermore, all distributions that are involved in the mixture must be continuous. You can specify the following *density-options* to modify the plot:

CUMULATIVE

CDF

displays the histogram and densities in cumulative form.

NBINS=*n*

BINS=*n*

specifies the number of bins in the histogram; *n* is greater than or equal to 0. By default, the FMM procedure computes a suitable bin width and number of bins, based on the range of the response and the number of usable observations. The option has no effect for binary data.

NOCOMPONENTS

NOCOMP

suppresses the component densities from the plot. If the component densities are displayed, they are scaled so that their sum equals the mixture density at any point on the graph. In single-component models, this option has no effect.

NODENSITY

NODENS

suppresses the computation of the mixture density and the component densities. If you specify the NOHISTOGRAM and the NODENSITY option, no plot is produced.

NOLABEL

suppresses the component identification with labels. By default, the FMM procedure labels component densities in the legend of the plot. If you do not specify a model label with the LABEL= option in the MODEL statement, an identifying label is constructed from the parameter estimates that are associated with the component. In this case the parameter values are not necessarily the mean and variance of the distribution; the values used to identify the densities on the plot are chosen to simplify linking between graphical and tabular results.

NOHISTOGRAM

NOHIST

suppresses the computation of the histogram of the raw values. If you specify the NOHISTOGRAM and the NODENSITY option, no plot is produced.

NPOINTS=*n*

N=*n*

specifies the number of values used to compute the density functions; *n* is greater than or equal to 0. The default is N=200.

WIDTH=*value*

BINWIDTH=*value*

specifies the bin width for the histogram. The *value* is specified in units of the response variable and must be positive. The option has no effect for binary data.

TRACE < (*tadpanel-options*) >
requests a trace panel with posterior diagnostics for a Bayesian analysis. If a BAYES statement is present, the trace panel plots are generated by default, one for each sampled parameter. You can specify the following *tadpanel-options* to modify the plot:

BOX
BOXPLOT
replaces the autocorrelation plot with a box plot of the posterior sample.

SMOOTH=NONE | MEAN | SPLINE
adds a reference estimate to the trace plot. By default, SMOOTH=NONE. SMOOTH=MEAN uses the arithmetic mean of the trace as the reference. SMOOTH=SPLINE adds a penalized B-spline.

REFERENCE=*reference-style*
adds reference lines to the density plot, trace plot, and box plot. You can specify the following *reference-styles*:

NONE suppresses the reference lines.

EQT requests equal-tail intervals.

HPD requests intervals of highest posterior density. The level for the credible or HPD intervals is chosen based on the "Posterior Interval Statistics" table.

PERCENTILES (or PERC) for percentiles. The largest and smallest percentiles in the "Posterior Summary Statistics" table are used.

The default is REFERENCE=EQT.

UNPACK
unpacks the panel graphic and displays its elements as separate plots.

CRITERIONPANEL < (*critpanel-options*) >
requests a plot for comparing the model fit criteria for different numbers of components. This plot is available only if you also specify the KMAX option in at least one MODEL statement. The plot includes different criteria, depending on whether you are using maximum likelihood or Bayesian estimation. You can specify the following *critpanel-option* to modify the plot:

UNPACK
unpacks the panel plot and displays its elements as separate plots, one for each fit criterion.

SEED=*n*
determines the random number seed for analyses that depend on a random number stream. If you do not specify a seed or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. The largest possible value for the seed is $2^{31} - 1$. The seed value is reported in the "Model Information" table.

You can use the SYSRANDOM and SYSRANEND macro variables after a PROC FMM run to query the initial and final seed values. However, using the final seed value as the starting seed for a subsequent analysis does not continue the random number stream where the previous analysis left off.

The SYSRANEND macro variable provides a mechanism to pass on seed values to ensure that the sequence of random numbers is the same every time you run an entire program.

Analyses that use the same (nonzero) seed are not completely reproducible if they are executed with a different number of threads because the random number streams in separate threads are independent. You can control the number of threads used by the FMM procedure with system options or through the PERFORMANCE statement in the FMM procedure.

SINGCHOL=*number*
tunes the singularity criterion in Cholesky decompositions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGRES=*number*
sets the tolerance for which the residual variance or scale parameter is considered to be zero. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGULAR=*number*
tunes the general singularity criterion applied by the FMM procedure in sweeps and inversions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

TECHNIQUE=*keyword*
TECH=*keyword*
specifies the optimization technique to obtain maximum likelihood estimates. You can choose from the following techniques by specifying the appropriate *keyword*:

CONGRA	performs a conjugate-gradient optimization.
DBLDOG	performs a version of double-dogleg optimization.
NEWRAP	performs a Newton-Raphson optimization combining a line-search algorithm with ridging.
NMSIMP	performs a Nelder-Mead simplex optimization.
NONE	performs no optimization.
NRRIDG	performs a Newton-Raphson optimization with ridging.
QUANEW	performs a dual quasi-Newton optimization.
TRUREG	performs a trust-region optimization.

The default is TECH=QUANEW.

For more details about these optimization methods, see the section "Choosing an Optimization Algorithm" on page 545 in Chapter 20, "Shared Concepts and Topics."

ZEROPROB=*number*
tunes the threshold (a value between 0 and 1) below which the FMM procedure considers a component mixing probability to be zero. This affects the calculation of the number of effective components. The default is the square root of the machine epsilon; this is approximately 1E–8 on most computers.

BAYES Statement

BAYES *bayes-options* **;**

The BAYES statement requests that the parameters of the model be estimated by Markov chain Monte Carlo sampling techniques. The FMM procedure can use maximum likelihood to estimate the parameters of all models that are supported by the procedure. Bayesian estimation is available for only a subset of these models.

In Bayesian analysis, it is essential to examine the convergence of the Markov chains before you proceed with posterior inference. With ODS Graphics turned on, the FMM procedure produces graphics at the end of the procedure output; you can visually examine the convergence of the chain in these graphics. You cannot base inferences on a Markov chain that has not converged.

The output produced for a Bayesian analysis is markedly different from that for a frequentist (maximum likelihood) analysis for the following reasons:

- Parameter estimates do not have the same interpretation in the two analyses. Parameters are fixed unknown constants in the frequentist context and random variables in a Bayesian analysis.
- The results of a Bayesian analysis are summarized through chain diagnostics and posterior summary statistics and intervals.
- The FMM procedure samples the mixing probabilities in Bayesian models directly, rather than mapping them onto a logistic (or other) scale.

The FMM procedure applies highly specialized sampling algorithms in Bayesian models. For single-component models without effects, a conjugate sampling algorithm is used where possible. For models in the exponential family that contain effects, the sampling algorithm is based on Gamerman (1997). For the normal and *t* distributions, a conjugate sampler is the default sampling algorithm for models with and without effects. In multi-component models, the sampling algorithm is based on latent variable sampling through data augmentation (Frühwirth-Schnatter 2006) and the Gamerman or conjugate sampler. Because of this specialization, the options for controlling the prior distributions of the parameters are limited.

Table 46.3 summarizes the *bayes-options* available in the BAYES statement. The full assortment of options is then described in alphabetical order.

Table 46.3 BAYES Statement Options

Option	Description
Options Related to Sampling	
INITIAL=	Specifies how to construct initial values
NBI=	Specifies the number of burn-in samples
NMC=	Specifies the number of samples after burn-in
METROPOLIS	Forces a Metropolis-Hastings sampling algorithm even if conjugate sampling is possible
OUTPOST=	Generates a data set that contains the posterior estimates
THIN=	Controls the thinning of the Markov chain

Table 46.3 *continued*

Option	Description
Specification of Prior Information	
MIXPRIORPARMS	Specifies the prior parameters for the Dirichlet distribution of the mixing probabilities
BETAPRIORPARMS=	Specifies the parameters of the normal prior distribution for individual parameters in the $\boldsymbol{\beta}$ vector
MUPRIORPARMS=	Specifies the parameters of the prior distribution for the means in homogeneous mixtures without effects
PHIPRIORPARMS=	Specifies the parameters of the inverse gamma prior distribution for the scale parameters in homogeneous mixtures
PRIOROPTIONS	Specifies additional options used in the determination of the prior distribution
Posterior Summary Statistics and Convergence Diagnostics	
DIAGNOSTICS=	Displays convergence diagnostics for the Markov chain
STATISTICS	Displays posterior summary information for the Markov chain
Other Options	
ESTIMATE=	Specifies which estimate is used for the computation of OUTPUT statistics and graphics
TIMEINC=	Specifies the time interval to report on sampling progress (in seconds)

You can specify the following *bayes-options* in the BAYES statement.

BETAPRIORPARMS=*pair-specification*

BETAPRIORPARMS(*pair-specification* **...** *pair-specification***)**

specifies the parameters for the normal prior distribution of the parameters that are associated with model effects (βs). The *pair-specification* is of the form (a, b), and the values a and b are the mean and variance of the normal distribution, respectively. This option overrides the PRIOROPTIONS option.

The form of the BETAPRIORPARMS with an equal sign and a single pair is used to specify one pair of prior parameters that applies to all components in the mixture. In the following example, the two intercepts and the two regression coefficients all have a $N(0, 100)$ prior distribution:

```
proc fmm;
   model y = x / k=2;
   bayes betapriorparms=(0,100);
run;
```

You can also provide a list of pairs to specify different sets of prior parameters for the various regression parameters and components. For example:

```
proc fmm;
   model y = x/ k=2;
   bayes betapriorparms( (0,10) (0,20) (.,.) (3,100) );
```

```
run;
```

The simple linear regression in the first component has a $N(0, 10)$ prior for the intercept and a $N(0, 20)$ prior for the slope. The prior for the intercept in the second component uses the FMM default, whereas the prior for the slope is $N(3, 100)$.

DIAGNOSTICS=ALL | NONE | (*keyword-list*)

DIAG=ALL | NONE | (*keyword-list*)

controls the computation of diagnostics for the posterior chain. You can request all posterior diagnostics by specifying DIAGNOSTICS=ALL or suppress the computation of posterior diagnostics by specifying DIAGNOSTICS=NONE. The following *keywords* enable you to select subsets of posterior diagnostics; the default is DIAGNOSTICS=(AUTOCORR).

AUTOCORR < (LAGS= *numeric-list*) >

computes for each sampled parameter the autocorrelations of lags specified in the LAGS= list. Elements in the list are truncated to integers, and repeated values are removed. If the LAGS= option is not specified, autocorrelations are computed by default for lags 1, 5, 10, and 50. See the section "Autocorrelations" on page 174 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

ESS

computes an estimate of the effective sample size (Kass et al. 1998), the correlation time, and the efficiency of the chain for each parameter. See the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

GEWEKE < (*geweke-options*) >

computes the Geweke spectral density diagnostics (Geweke 1992), which are essentially a two-sample t test between the first f_1 portion and the last f_2 portion of the chain. The default is $f_1 = 0.1$ and $f_2 = 0.5$, but you can choose other fractions by using the following *geweke-options*:

FRAC1=*value*

specifies the fraction f_1 for the first window.

FRAC2=*value*

specifies the fraction f_2 for the second window.

See the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

HEIDELBERGER < (*Heidel-options*) >

HEIDEL < (*Heidel-options*) >

computes the Heidelberger and Welch diagnostic (which consists of a stationarity test and a half-width test) for each variable. The stationary diagnostic test tests the null hypothesis that the posterior samples are generated from a stationary process. If the stationarity test is passed, a half-width test is then carried out. See the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

These diagnostics are not performed by default. You can specify the DIAGNOSTICS=HEIDELBERGER option to request these diagnostics, and you can also specify suboptions, such as DIAGNOSTICS=HEIDELBERGER(EPS=0.05), as follows:

SALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the stationarity test. By default, SALPHA=0.05.

HALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the half-width test. By default, HALPHA=0.05.

EPS=*value*
specifies a small positive number ϵ such that if the half-width is less than ϵ times the sample mean of the retaining iterates, the half-width test is passed. By default, EPS=0.1.

MCERROR

MCSE
computes an estimate of the Monte Carlo standard error for each sampled parameter. See the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

MAXLAG=*n*
specifies the largest lag used in computing the effective sample size and the Monte Carlo standard error. Specifying this option implies the ESS and MCERROR options. The default is MAXLAG=250.

RAFTERY <(*Raftery-options***)>**

RL <(*Raftery-options***)>**
computes the Raftery and Lewis diagnostics, which evaluate the accuracy of the estimated quantile ($\hat{\theta}_Q$ for a given $\mathrm{Q} \in (0, 1)$) of a chain. $\hat{\theta}_Q$ can achieve any degree of accuracy when the chain is allowed to run for a long time. The algorithm stops when the estimated probability $\hat{P}_Q = \Pr(\theta \leq \hat{\theta}_Q)$ reaches within $\pm$R of the value Q with probability S; that is, $\Pr(\mathrm{Q} - \mathrm{R} \leq \hat{\mathrm{P}}_Q \leq \mathrm{Q} + \mathrm{R}) = \mathrm{S}$. See the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The *Raftery-options* enable you to specify Q, R, S, and a precision level ϵ for a stationary test.

These diagnostics are not performed by default. You can specify the DIAGNOSTICS=RAFERTY option to request these diagnostics, and you can also specify suboptions, such as DIAGNOSTICS=RAFERTY(QUANTILE=0.05), as follows:

QUANTILE=*value*

Q=*value*
specifies the order (a value between 0 and 1) of the quantile of interest. By default, QUANTILE=0.025.

ACCURACY=*value*

R=*value*
specifies a small positive number as the margin of error for measuring the accuracy of estimation of the quantile. By default, ACCURACY=0.005.

PROB=*value*

S=*value*

specifies the probability of attaining the accuracy of the estimation of the quantile. By default, PROB=0.95.

EPS=*value*

specifies the tolerance level (a small positive number between 0 and 1) for the stationary test. By default, EPS=0.001.

MIXPRIORPARMS=K

MIXPRIORPARMS(*value-list***)**

specifies the parameters used in constructing the Dirichlet prior distribution for the mixing parameters. If you specify MIXPRIORPARMS=K, the parameters of the k-dimensional Dirichlet distribution are a vector that contains the number of components in the model (k), whatever that might be. You can specify an explicit list of parameters in *value-list*. If the MIXPRIORPARMS option is not specified, the default Dirichlet parameter vector is a vector of length k of ones. This results in a uniform prior over the unit simplex; for k=2, this is the uniform distribution. See the section "Prior Distributions" on page 3104 for the distribution function of the Dirichlet as used by the FMM procedure.

ESTIMATE=MEAN | MAP

determines which overall estimate is used, based on the posterior sample, in the computation of OUTPUT statistics and certain ODS graphics. By default, the arithmetic average of the (thinned) posterior sample is used. If you specify ESTIMATE=MAP, the parameter vector is used that corresponds to the maximum log posterior density in the posterior sample. In any event, a message is written to the SAS log if postprocessing results depend on a summary estimate of the posterior sample.

INITIAL=DATA | MLE | MODE | RANDOM

determines how initial values for the Markov chain are obtained. The default when a conjugate sampler is used is INITIAL=DATA, in which case the FMM procedure uses the same algorithm to obtain data-dependent starting values as it uses for maximum likelihood estimation. If no conjugate sampler is available or if you use the METROPOLIS option to explicitly request that it not be used, then the default is INITIAL=MLE, in which case the maximum likelihood estimates are used as the initial values. If the maximum likelihood optimization fails, the FMM procedure switches to the default INITIAL=DATA.

The options INITIAL=MODE and INITIAL=RANDOM use the mode and random draws from the prior distribution, respectively, to obtain initial values. If the mode does not exist or if it falls on the boundary of the parameter space, the prior mean is used instead.

METROPOLIS

requests that the FMM procedure use the Metropolis-Hastings sampling algorithm based on Gamerman (1997), even in situations where a conjugate sampler is available.

MUPRIORPARMS=*pair-specification*

MUPRIORPARMS(*pair-specification* **...** *pair-specification***)**

specifies the parameters for the means in homogeneous mixtures without regression coefficients. The *pair-specification* is of the form (a, b), where a and b are the two parameters of the prior distribution, optionally delimited with a comma. The actual distribution of the parameter is implied by the distribution selected in the MODEL statement. For example, it is a normal distribution for a mixture of

normals, a gamma distribution for a mixture of Poisson variables, a beta distribution for a mixture of binary variables, and an inverse gamma distribution for a mixture of exponential variables. This option overrides the PRIOROPTIONS option.

The parameters correspond as follows:

Beta: The parameters correspond to the α and β parameters of the beta prior distribution such that its mean is $\mu = \alpha/(\alpha + \beta)$ and its variance is $\mu(1 - \mu)/(\alpha + \beta + 1)$.

Normal: The parameters correspond to the mean and variance of the normal prior distribution.

Gamma: The parameters correspond to the α and β parameters of the gamma prior distribution such that its mean is α/β and its variance is α/β^2.

Inverse gamma: The parameters correspond to the α and β parameters of the inverse gamma prior distribution such that its mean is $\mu = \beta/(\alpha - 1)$ and its variance is $\mu^2/(\alpha - 2)$.

The two techniques for specifying the prior parameters with the MUPRIORPARMS option are as follows:

- Specify an equal sign and a single pair of values:

```
proc fmm seed=12345;
   model y = / k=2;
   bayes mupriorparms=(0,50);
run;
```

- Specify a list of parameter pairs within parentheses:

```
proc fmm seed=12345;
   model y = / k=2;
   bayes mupriorparms( (.,.) (1.4,10.5));
run;
```

If you specify an invalid value (outside of the parameter space for the prior distribution), the FMM procedure chooses the default value and writes a message to the SAS log. If you want to use the default values for a particular parameter, you can also specify missing values in the *pair-specification*. For example, the preceding list specification assigns default values for the first component and uses the values 1.4 and 10.5 for the mean and variance of the normal prior distribution in the second component. The first example assigns a $N(0, 50)$ prior distribution to the means in both components.

NBI=*n*

specifies the number of burn-in samples. During the burn-in phase, chains are not saved. The default is NBI=2000.

NMC=*n*

SAMPLE=*n*

specifies the number of Monte Carlo samples after the burn-in. Samples after the burn-in phase are saved unless they are thinned with the THIN= option. The default is NMC=10000.

OUTPOST<(*outpost-options*)>=*data-set*

requests that the posterior sample be saved to a SAS data set. In addition to variables that contain log likelihood and log posterior values, the OUTPOST data set contains variables for the parameters. The variable names for the parameters are generic (Parm_1, Parm_2, . . ., Parm_p). The labels of the parameters are descriptive and correspond to the "Parameter Mapping" table that is produced when the OUTPOST= option is in effect.

You can specify the following *outpost-options* in parentheses:

LOGPRIOR

adds the value of the log prior distribution to the data set.

NONSINGULAR | NONSING | COMPRESS

eliminates parameters that correspond to singular columns in the design matrix (and were not sampled) from the posterior data set. This is the default.

SINGULAR | SING

adds columns of zeros to the data set in positions that correspond to singularities in the model or to parameters that were not sampled for other reasons. By default, these columns of zeros are not written to the posterior data set.

PHIPRIORPARMS=*pair-specification*

PHIPRIORPARMS(*pair-specification* . . . *pair-specification*)

specifies the parameters for the inverse gamma prior distribution of the scale parameters (ϕ's) in the model. The *pair-specification* is of the form (a, b), and the values are chosen such that the prior distribution has mean $\mu = b/(a-1)$ and variance $\mu^2/(a-2)$.

The form of the PHIPRIORPARMS with an equal sign and a single pair is used to specify one pair of prior parameters that applies to all components in the mixture. For example:

```
proc fmm seed=12345;
   model y = / k=2;
   bayes phipriorparms=(2.001,1.001);
run;
```

The form with a list of pairs is used to specify different prior parameters for the scale parameters in different components. For example:

```
proc fmm seed=12345;
   model y = / k=2;
   bayes phipriorparms( (.,1.001)  (3.001,2.001) );
run;
```

If you specify an invalid value (outside of the parameter space for the prior distribution), the FMM procedure chooses the default value and writes a message to the SAS log. If you want to use the default values for a particular parameter, you can also specify missing values in the *pair-specification*. For example, the preceding list specification assigns default values for the first component a prior parameter and uses the value 1.001 for the b prior parameter. The second pair assigns 3.001 and 2.001 for the a and b prior parameters, respectively.

PRIOROPTIONS <=>(*prior-options*)

PRIOROPTS <=>(*prior-options*)

specifies options related to the construction of the prior distribution and the choice of their parameters. Some *prior-options* apply only in particular models. The BETAPRIORPARMS= and MUPRIORPARMS= options override this option.

You can specify the following *prior-options*:

CONDITIONAL | COND

chooses a conditional prior specification for the homogeneous normal and *t* distribution response components. The default prior specification in these models is an independence prior where the mean of the *h*th component has prior $\mu_h \sim N(a, b)$. The conditional prior is characterized by $\mu_h \sim N(a, \sigma_h^2/b)$.

DEPENDENT | DEP

chooses a data-dependent prior for the homogeneous models without effects. The prior parameters *a* and *b* are chosen as follows, based on the distribution in the MODEL statement:

Binary and binomial: $a = \bar{y}/(1-\bar{y})$, *b*=1, and the prior distribution for the success probability is $\mathrm{beta}(a, b)$.

Poisson: $a = 1$, $b = 1/\bar{y}$, and the prior distribution for μ is $\mathrm{gamma}(a, b)$. See Frühwirth-Schnatter (2006, p. 280) and Viallefont, Richardson, and Greene (2002).

Exponential: $a = 3$, $b = 2\bar{y}$, and the prior distribution for μ is inverse gamma with parameters *a* and *b*.

Normal and *t*: Under the default independence prior, the prior distribution for μ is $N(\bar{y}, fs^2)$ where *f* is the variance factor from the VAR= option and

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

Under the default conditional prior specification, the prior for μ_h is $N(a, \sigma_h^2/b)$ where $a = \bar{y}$ and $b = 2.6/(\max\{y\} - \min\{y\})$. The prior for the scale parameter is inverse gamma with parameters 1.28 and $0.36s^2$. For further details, see Raftery (1996) and Frühwirth-Schnatter (2006, p. 179).

VAR=*f*

specifies the variance for normal prior distributions. The default is VAR=1000. This factor is used, for example, in determining the prior variance of regression coefficients or in determining the prior variance of means in homogeneous mixtures of *t* or normal distributions (unless a data-dependent prior is used).

MLE<=*r*>

specifies that the prior distribution for regression variables be based on a multivariate normal distribution centered at the MLEs and whose dispersion is a multiple *r* of the asymptotic MLE covariance matrix. The default is MLE=10. In other words, if you specify PRIOROPTIONS(MLE), the FMM procedure chooses the prior distribution for the regression variables as $N(\widehat{\beta}, 10\mathrm{Var}[\widehat{\beta}])$

where $\widehat{\beta}$ is the vector of maximum likelihood estimates. The prior for the scale parameter is inverse gamma with parameters 1.28 and $0.36s^2$ where

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

For further details, see Raftery (1996) and Frühwirth-Schnatter (2006, p. 179). If you specify PRIOROPTIONS(MLE) for the regression parameters, then the data-dependent prior is used for the scale parameter; see the PRIOROPTIONS(DEPENDENT) option above.

The MLE option is not available for mixture models in which the parameters are estimated directly on the data scale, such as homogeneous mixture models or mixtures of distributions without model effects for which a conjugate sampler is available. By using the METROPOLIS option, you can always force the FMM procedure to abandon a conjugate sampler in favor of a Metropolis-Hastings sampling algorithm to which the MLE option applies.

STATISTICS < (*global-options*) > = **ALL** | **NONE** | *keyword* | (*keyword-list*)

SUMMARIES < (*global-options*) > = **ALL** | **NONE** | *keyword* | (*keyword-list*)

controls the number of posterior statistics produced. Specifying STATISTICS=ALL is equivalent to specifying STATISTICS=(SUMMARY INTERVAL). To suppress the computation of posterior statistics, specify STATISTICS=NONE. The default is STATISTICS=(SUMMARY INTERVAL). See the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

The *global-options* include the following:

ALPHA=*numeric-list*

controls the coverage levels of the equal-tail credible intervals and the credible intervals of highest posterior density (HPD) credible intervals. The ALPHA= values must be between 0 and 1. Each ALPHA= value produces a pair of $100(1-\alpha)\%$ equal-tail and HPD credible intervals for each sampled parameter. The default is ALPHA=0.05, which results in 95% credible intervals for the parameters.

PERCENT=*numeric-list*

requests the percentile points of the posterior samples. The values in *numeric-list* must be greater than or equal to 0 and less than or equal to 100. The default is PERCENT=(25 50 75), which yields for each parameter the 25th, 50th, and 75th percentiles, respectively.

The list of *keywords* includes the following:

SUMMARY

produces the means, standard deviations, and percentile points for the posterior samples. The default is to produce the 25th, 50th, and 75th percentiles; you can modify this list with the global PERCENT= option.

INTERVAL

produces equal-tail and HPD credible intervals. The default is to produce the 95% equal-tail credible intervals and 95% HPD credible intervals, but you can use the ALPHA= *global-option* to request credible intervals for any probabilities.

THIN=*n*

THINNING=*n*

controls the thinning of the Markov chain after the burn-in. Only one in every k samples is used when THIN=k, and if NBI=n_0 and NMC=n, the number of samples kept is

$$\left[\frac{n_0 + n}{k}\right] - \left[\frac{n_0}{k}\right]$$

where $[a]$ represents the integer part of the number a. The default is THIN=1—that is, all samples are kept after the burn-in phase.

TIMEINC=*n*

specifies a time interval in seconds to report progress during the burn-in and sampling phase. The time interval is approximate, because the minimum time interval in which the FMM procedure can respond depends on the multithreading configuration.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC FMM to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the FMM procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Because sorting the data changes the order in which PROC FMM reads observations, the sort order for the levels of the CLASS variable might be affected if you have specified ORDER=DATA in the PROC FMM statement.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / **TRUNCATE** > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC FMM statement.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence for each observation. PROC FMM treats each observation as if it appears f times, where f is the value of the FREQ variable for the observation. If it is not an integer, the frequency value is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

ID Statement

ID *variables* **;**

The ID statement specifies a list of variables that are included in the OUT= data set of the OUTPUT statement. If no ID statement is specified, all variables from the input data set are copied into the output data set.

MODEL Statement

MODEL *response* < **(***response-options***)** > **=** < *effects* > < / *model-options* > **;**

MODEL *events/trials* **=** < *effects* > < / *model-options* > **;**

MODEL **+** < *effects* > < / *model-options* > **;**

The MODEL statement defines elements of the mixture model, such as the model effects, the distribution, and the link function. At least one MODEL statement is required. You can specify more than one MODEL statement. Each MODEL statement identifies one or more components of a mixture. For example, if components differ in their distributions, link functions, or regressor variables, then you can use separate MODEL statements to define the components. If the finite mixture model is homogeneous—meaning that all components share the same regressors, distribution, and link function—then you can specify the mixture model by using a single MODEL statement and specifying the K= option.

The FMM procedure includes an intercept in each model by default. You can remove it by using the NOINT option.

You can specify the dependent variable by using either the *response* syntax or the *events/trials* syntax. The *events/trials* syntax is specific to models for binomial-type data. A binomial(n, π) variable is the sum of n independent Bernoulli trials with event probability π. Each Bernoulli trial results in either an event or a nonevent (with probability $1-\pi$). The value of the second variable, *trials*, gives the number n of Bernoulli trials. The value of the first variable, *events*, is the number of events out of n. The values of both *events* and (*trials* – *events*) must be nonnegative, and the value of *trials* must be positive. Other distributions that allow the *events/trials* syntax are the beta-binomial distribution and the binomial cluster model.

If you use the *events/trials* syntax, the FMM procedure defaults to the binomial distribution. If you use the *response* syntax, the procedure defaults to the normal distribution unless the response variable is a character variable or is listed in the CLASS statement.

You use a similar syntax to fit multinomial models, Dirichlet-multinomial models, and multinomial cluster models, except that for these distributions you specify multiple dependent variables, one for each value of the multinomial response. For these models, you can specify either multiple *response* variables or multiple *event* variables. If you use multiple *response* or multiple *event* variables, the FMM procedure defaults to the multinomial distribution. If you use the multiple *response* syntax, PROC FMM treats the total of these responses as fixed.

The FMM procedure supports a continuation-style syntax in MODEL statements. Because a mixture has only one set of response variables, it is sufficient to specify the response variable in one MODEL statement. Other MODEL statements can use the continuation symbol "+" before the specification of effects. For example, the following statements fit a three-component binomial mixture model:

```
class A;
model y/n = x / k=2;
model     + A;
```

The first MODEL statement uses the "=" sign to separate response information from effect information and specifies the response variable by using the *events/trials* syntax. This determines that the distribution is binomial. This MODEL statement adds to the mixture model two components that have different intercepts and regression slopes. The second MODEL statement adds to the mixture model another component in which the mean is a function of the classification main effect for variable A. The response is also binomial; it is a continuation from the previous MODEL statement.

There are two sets of options in the MODEL statement. The *response-options* determine how the FMM procedure models probabilities for binary data. The *model-options* control other aspects of model formation and inference. Table 46.4 summarizes the *response-options* and *model-options* available in the MODEL statement. These are subsequently discussed in detail in alphabetical order by option category.

Table 46.4 Summary of MODEL Statement Options

Option	Description
Response Variable Options	
DESCENDING	Reverses the order of response categories
EVENT=	Specifies the event category in binary models
ORDER=	Specifies the sort order for the response variable
REFERENCE=	Specifies the reference category in binary models
Model Building	
DIST=	Specifies the response distribution
LINK=	Specifies the link function
K=	Specifies the number of mixture components
KMAX=	Specifies the maximum number of mixture components
KMIN=	Specifies the minimum number of mixture components
KRESTART	Requests that the starting values for each analysis be determined separately instead of sequentially
NOINT	Excludes fixed-effect intercept from model
OFFSET=	Specifies the offset variable for linear predictor
Statistical Computations and Output	
ALPHA=α	Determines the confidence level $(1-\alpha)$
CL	Displays confidence limits for fixed-effects parameter estimates
EQUATE=	Imposes simple equality constraints on parameters in this model
LABEL=	Identifies the model
PARMS	Provides starting values for the parameters in this model

Response Variable Options

Response variable options determine how the FMM procedure models probabilities for binary data.

You can specify the following *response-options* by enclosing them in parentheses after the *response* variable. The default is ORDER=FORMATTED.

DESCENDING
DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC FMM orders the response categories according to the ORDER= option and then reverses that order.

EVENT='*category*' | *keyword*

specifies the event category for the binary response model. PROC FMM models the probability of the event category. You can specify the value (formatted, if a format is applied) of the event category in quotes, or you can specify one of the following *keywords*:

FIRST

designates the first ordered category as the event. This is the default.

LAST

designates the last ordered category as the event.

ORDER=*order-type*

specifies the sort order for the levels of the response variable. You can specify the following values for *order-type*:

DATA

sorts the levels by order of appearance in the input data set.

FORMATTED

sorts the levels by external formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value.

FREQ

sorts the levels by descending frequency count; levels with the most observations come first in the order.

INTERNAL

sorts the levels by unformatted value.

FREQDATA

sorts the levels by order of descending frequency count, and within counts by order of appearance in the input data set when counts are tied.

FREQFORMATTED

sorts the levels by order of descending frequency count, and within counts by formatted value (as above) when counts are tied.

FREQINTERNAL

sorts the levels by order of descending frequency count, and within counts by unformatted value when counts are tied.

When ORDER=FORMATTED (the default) for numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC FMM run or in the DATA step that created the data set), the levels are ordered by their internal (numeric) value. If you specify the ORDER= option in the MODEL statement and the ORDER= option in the PROC FMM statement, the former takes precedence.

By default, ORDER=FORMATTED. For the FORMATTED and INTERNAL values, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | *keyword*

REF='*category*' | *keyword*

specifies the reference category for categorical models. For the binary response model, specifying one response category as the reference category is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotation marks, or you can specify one of the following *keywords*:

FIRST

designates the first-ordered category as the reference category.

LAST

designates the last-ordered category as the reference category. This is the default.

The reference category in multinomial, Dirichlet-multinomial, and multinomial cluster models is determined by the order in which you specify the response variables.

Model Options

ALPHA=*number*

requests that the procedure construct confidence intervals that have confidence level 1−*number* for each parameter estimate. The value of *number* must be between 0 and 1; the default is 0.05.

CL

requests that PROC FMM construct confidence limits for each parameter estimate. The confidence level is 0.95 by default; you can change it by using the ALPHA= option.

DISTRIBUTION=*keyword*

DIST=*keyword*

specifies the probability distribution for a mixture component.

If you specify the DIST= option and you do not specify a link function by using the LINK= option, a default link function is chosen (see Table 46.5). If you do not specify a distribution, the FMM procedure defaults to the normal distribution for continuous response variables and to the binary distribution for classification or character variables, unless you use the *events/trials* syntax in the MODEL statement. If you use the *events/trials* syntax, PROC FMM defaults to the binomial distribution.

Table 46.5 lists *keywords* that you can specify for the DISTRIBUTION= option and their corresponding default link functions. For generalized linear models with these distributions, you can find expressions for the log-likelihood functions in the section "Log-Likelihood Functions for Response Distributions" on page 3094.

Table 46.5 Keyword Values of the DIST= Option

keyword	Alias	Distribution	Default Link Function
BETA		Beta	Logit
BETABINOMIAL	**BETABIN**,**BB**	Beta-binomial	Logit
BINARY	**BERNOULLI**	Binary	Logit
BINOMIAL	**BIN**	Binomial	Logit
BINOMCLUSTER	**BCLUS**	Binomial cluster	Logit

Table 46.5 *continued*

DIST=	Alias	Distribution	Default Link Function
CONSTANT <(*c*)>	**DEGENERATE** <(*c*)>	Degenerate	N/A
DIRICHLETMULTINOMIAL	**DIRIMULT**,**DM**	Dirichlet-multinomial	Generalized logit
EXPONENTIAL	**EXPO**	Exponential	Log
FOLDEDNORMAL	**FNORMAL**	Folded normal	Identity
GAMMA	**GAM**	Gamma	Log
GAUSSIAN	**NORMAL**	Normal	Identity
GENPOISSON	**GPOISSON**	Generalized Poisson	Log
GEOMETRIC	**GEOM**	Geometric	Log
INVGAUSS	**IGAUSSIAN**, **IG**	Inverse Gaussian	Inverse squared (power(–2))
LOGNORMAL	**LOGN**	Lognormal	Identity
MULTINOMIAL	**MULTI**	Multinomial	Generalized logit
MULTINOMCLUSTER	**MCLUS**	Multinomial cluster	Logit
NEGBINOMIAL	**NEGBIN**, **NB**	Negative binomial	Log
POISSON	**POI**	Poisson	Log
T <(ν)>	**STUDENT** <(ν)>	t	Identity
TRUNCEXPO <(a,b)>	**TEXPO** <(a,b)>	Truncated exponential	Log
TRUNCLOGN <(a,b)>	**TLOGN** <(a,b)>	Lognormal	Identity
TRUNCNEGBIN	**TNEGBIN**, **TNB**	Negative binomial	Log
TRUNCNORMAL <(a,b)>	**TNORMAL** <(a,b)>	Truncated normal	Identity
TRUNCPOISSON	**TPOISSON**, **TPOI**	Truncated Poisson	Log
UNIFORM <(a,b)>	**UNIF** <(a,b)>	Uniform	N/A
WEIBULL		Weibull	Log

Note that the PROC FMM default link for the gamma or exponential distribution is not the canonical link (the reciprocal link).

The binomial cluster and multinomial cluster models are multiple-component models that are described in Morel and Nagaraj (1993); Morel and Neerchal (1997); Neerchal and Morel (1998). See Example 46.1 for an application of the binomial cluster model in a teratological experiment. See Example 46.4 for an application of the multinomial cluster model to housing survey data.

If you use the *events/trials* syntax, then the default distribution is the binomial distribution and only the following choices are available: DIST=BINOMIAL, DIST=BETABINOMIAL, and DIST=BINOMCLUSTER. The *trials* variable is ignored for all other distributions. This enables you to fit models in which some components have a binomial or binomial-like distribution. For example, suppose that the variable n is a binomial denominator and the variable logn is its logarithm. Then the following statements model a two-component mixture of a binomial and Poisson count model:

```
model y/n  = ;
model      +  / dist=Poisson offset=logn;
```

You use the OFFSET= option in the second MODEL statement to specify that the Poisson counts refer to different base counts, because the trial variable *n* is ignored in the second model.

If DIST=BINOMIAL is specified without the *events/trials* syntax, then n=1 is used for the default number of trials.

Similarly, if you specify multiple dependent variables by using the *response . . . response* and *event . . . event /trials* notation, the default distribution is the multinomial and only DIST=MULTINOMIAL, DIST=DIRICHLETMULTINOMIAL, and DIST=MULTINOMCLUSTER are available.

DIST=TRUNCNEGBIN and DIST=TRUNCPOISSON are zero-truncated versions of DIST=NEGBINOMIAL and DIST=POISSON, respectively—that is, only the value of 0 is excluded from the support.

For DIST=TRUNCEXPO, DIST=TRUNCLOGN, and DIST=TRUNCNORMAL, you must specify the lower (*a*) and upper (*b*) truncation points of the distribution. For example:

DIST=TRUNCEXPO<(*a*,*b*)>

DIST=TRUNCLOGN<(*a*,*b*)>

DIST=TRUNCNORMAL<(*a*,*b*)>

Each of these distributions is the conditional version of its corresponding nontruncated distribution that is confined to the support $[a, b]$ (inclusive). You can specify a missing value (.) for either *a* or *b* to truncate only on the other side; that is, *a*=. indicates a right-truncated distribution, and *b*=. indicates a left-truncated distribution.

For several distribution specifications you can provide additional optional parameters to further define the distribution. These optional parameters are listed in the following:

CONSTANT<(*c*)> The number *c* specifies the value where the mass is concentrated. The default is DIST=CONSTANT(0), so you can add zero-inflation to any model by adding a MODEL statement with DIST=CONSTANT.

T<(ν)> The number ν specifies the degrees of freedom for the (shifted) t distribution. The default is DIST=T(3); this leads to a heavy-tailed distribution for which the variance is defined. See the section "Log-Likelihood Functions for Response Distributions" on page 3094 for the density function of the shifted t_ν distribution.

UNIFORM<(*a*,*b*)> The values *a* and *b* define the support of the uniform distribution, $a < b$. By default, $a = 0$ and $b = 1$.

EQUATE=MEAN | SCALE | NONE | EFFECTS(*effect-list***)**

specifies simple sets of parameter constraints across the components in a MODEL statement; the default is EQUATE=NONE. This option is available only for maximum likelihood estimation. If you specify EQUATE=MEAN, the parameters that determine the mean are reduced to a single set that is applicable to all components in the MODEL statement. If you specify EQUATE=SCALE, a single parameter represents the common scale for all components in the MODEL statement. The EFFECTS option enables you to force the parameters for the chosen model effects to be equal across components; however, the number of parameters is unaffected.

For example, the following statements fit a two-component multiple regression model in which the coefficients for variable logd vary by component and the intercepts and coefficients for variable dose are the same for the two components:

```
proc fmm;
   model num = dose logd / equate=effects(int dose) k=2;
run;
```

To fix all coefficients across the two components, you can write the MODEL statement as

```
model num = dose logd / equate=effects(int dose logd) k=2;
```

or

```
model num = dose logd / equate=mean k=2;
```

If you restrict all parameters in a *k*-component MODEL statement to be equal, the FMM procedure reduces the model to *k*=1.

K=*n*

NUMBER=*n*

specifies the number of components the MODEL statement contributes to the overall mixture. For binomial cluster models and multinomial cluster models, this option is not available because these are multiple-component models by definition.

KMAX=*n*

specifies the maximum number of components the MODEL statement contributes to the overall mixture.

If the maximum number of components in the mixture, as determined by all KMAX= options, is larger than the minimum number of components, the FMM procedure fits all possible models and displays summary fit information for the sequence of evaluated models. The "best" model according to the CRITERION= option in the PROC FMM statement is then chosen, and the remaining output and analyses performed by PROC FMM pertain to this "best" model.

When you use MCMC methods to estimate the parameters of a mixture, you need to ensure that the chain for a given value of *k* has converged; otherwise, comparisons among models that have varying numbers of components might not be meaningful. You can use the FITDETAILS option to display summary and diagnostic information for the MCMC chains from each model.

If you specify the KMIN= option but not the KMAX= option, then the default value for the KMAX= option is the value of the KMIN= option (unless KMIN=0, in which case the KMAX= option is set to 1).

KMIN=*n*

specifies the minimum number of components that the MODEL statement contributes to the overall mixture. When you use MCMC methods to estimate the parameters of a mixture, you need to ensure that the chain for a given value of *k* has converged; otherwise, comparisons among models that have varying numbers of components might not be meaningful.

KRESTART

requests that the starting values for each analysis (that is, for each unique number of components as determined by the KMIN= and KMAX= options) be determined separately, in the same way as if no other analyses were performed. If you do not specify the KRESTART option, then the starting values for each analysis are based on results from the previous analysis with one less component.

LABEL='*label***'**

specifies an optional label for the model that is used to identify the model in printed output, on graphics, and in data sets created from ODS tables.

LINK=*keyword*

specifies the link function in the model. The *keywords* and expressions for the associated link functions are shown in Table 46.6.

Table 46.6 Link Functions in MODEL Statement of the FMM Procedure

LINK=	Alias	Link Function	$g(\mu) = \eta =$
CLOGLOG	**CLL**	Complementary log-log	$\log(-\log(1-\mu))$
IDENTITY	**ID**	Identity	μ
LOG		Log	$\log(\mu)$
LOGIT		Logit	$\log(\mu/(1-\mu))$
LOGLOG		Log-log	$-\log(-\log(\mu))$
PROBIT	**NORMIT**	Probit	$\Phi^{-1}(\mu)$
POWER(λ)	**POW**(λ)	Power with exponent λ= *number*	$\begin{cases} \mu^{\lambda} & \text{if } \lambda \neq 0 \\ \log(\mu) & \text{if } \lambda = 0 \end{cases}$
POWERMINUS2		Power with exponent –2	$1/\mu^2$
RECIPROCAL	**INVERSE**	Reciprocal	$1/\mu$

The default link functions for the various distributions are shown in Table 46.5.

NOINT

excludes the intercept from the model. An intercept is included by default, unless the distribution is DIST=CONSTANT or DIST=UNIFORM.

OFFSET=*variable*

specifies the offset variable function for the linear predictor in the model. An offset variable can be thought of as a regressor variable whose regression coefficient is known to be 1. For example, you can use an offset in a Poisson model when counts have been obtained in time intervals of different lengths. With a log link function, you can model the counts as Poisson variables with the logarithm of the time interval as the offset variable.

PARAMETERS(*parameter-specification***)**

PARMS(*parameter-specification***)**

specifies starting values for the model parameters. If no PARMS option is specified, the FMM procedure determines starting values by using a data-dependent algorithm. To determine initial values for the Markov chain in Bayesian estimation, see the INITIAL= option in the BAYES statement. The

specification of the parameters takes the following form: parameters in the mean function precede the scale parameters, and parameters for different components are separated by commas.

The following statements specify starting parameters for a two-component normal model. The initial values for the intercepts are 1 and –3; the initial values for the variances are 0.5 and 4.

```
proc fmm;
   model y = / k=2 parms(1 0.5, -3 4);
run;
```

You can specify missing values for parameters whose starting values are to be determined by the default method. Only values for parameters that participate in the optimization are specified. The values for model effects are specified on the linear (linked) scale.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*>
< *keyword*< (*keyword-options*) > < **=***name* > > . . .
< *keyword*< (*keyword-options*) > < **=***name* > > < / *options* > **;**

The OUTPUT statement creates a data set that contains observationwise statistics that are computed after fitting the model. By default, all variables in the original data set are included in the output data set. You can use the ID statement to limit the variables copied from the input data set to the output data set.

The output statistics are computed based on the parameter estimates of the converged model if the parameters are estimated by maximum likelihood. If a Bayesian analysis is performed, the output statistics are computed based on the arithmetic mean in the posterior sample. You can change to the maximum posterior estimate with the ESTIMATE=MAP option in the BAYES statement.

You can specify the following syntax elements in the OUTPUT statement before the slash (/).

OUT=*SAS-data-set*
specifies the name of the output data set. If the OUT= option is omitted, the procedure uses the DATA*n* convention to name the output data set.

keyword< (*keyword-options*) > < **=***name* >
specifies a statistic to include in the output data set and optionally assigns the variable the name *name*. If you do not provide a name, the FMM procedure assigns a default name based on the type of statistic requested. If you provide a name for a statistic that leads to multiple output statistics, the name is modified to index the associated component number. For multivariate responses, names are also modified to index the associated response.

You can use the *keyword-options* to control which type of a particular statistic is computed. The following are valid values for *keyword* and *keyword-options*:

PREDICTED< (COMPONENT | OVERALL) >

PRED< (COMPONENT | OVERALL) >

MEAN< (COMPONENT | OVERALL) >

requests predicted values (predicted means) for the response variable. The predictions in the output data set are mapped onto the data scale in all cases except for a multinomial, binomial, or binary response with *events/trials* syntax and when PREDTYPE=COUNT has not been specified. In this case, for the binomial or binary response, the predictions are predicted success probabilities. For the multinomial response, the predictions are probabilities for each response category.

The default is to compute the predicted value for the mixture (OVERALL). You can request predictions for the means of the component distributions by adding the COMPONENT suboption in parentheses. The predicted values for some distributions are not identical to the parameter modeled as μ. For example, in the lognormal distribution the predicted mean is $\exp\{\mu + 0.5\phi\}$ where μ and ϕ are the parameters of an underlying normal process; see the section "Log-Likelihood Functions for Response Distributions" on page 3094 for details.

RESIDUAL< (COMPONENT | OVERALL) >

RESID< (COMPONENT | OVERALL) >

requests residuals for the response or residuals in the component distributions. Only "raw" residuals on the data scale are computed (observed minus predicted).

VARIANCE< (COMPONENT | OVERALL) >

VAR< (COMPONENT | OVERALL) >

requests variances for the mixture or the component distributions. For multivariate responses, the output statistic represents the diagonal entry in the associated covariance matrix.

LOGLIKE< (COMPONENT | OVERALL) >

LOGL< (COMPONENT | OVERALL) >

requests values of the log-likelihood function for the mixture or the components. For observations used in the analysis, the overall computed value is the observations' contribution to the log likelihood; if a FREQ statement is present, the frequency is accounted for in the computed value. In other words, if all observations in the input data set have been used in the analysis, adding the value of the log-likelihood contributions in the OUTPUT data set produces the negative of the final objective function value in the "Iteration History" table. By default, the log-likelihood contribution to the mixture is computed. You can request the individual mixture component contributions with the COMPONENT suboption.

MIXPROBS< (COMPONENT | MAX) >

MIXPROB< (COMPONENT | MAX) >

PRIOR< (COMPONENT | MAX) >

MIXWEIGHTS< (COMPONENT | MAX) >

requests that the prior weights $\pi_j(\mathbf{z}, \boldsymbol{\alpha}_j)$ be added to the OUTPUT data set. By default, the probabilities are output for all components. You can limit the output to a single statistic, the largest mixing probability, with the MAX suboption.

NOTE: The keyword "prior" is used here because of long-standing practice to refer to the mixing probabilities as prior weights. This must not be confused with the prior distribution and its parameters in a Bayesian analysis.

POSTERIOR< (COMPONENT | MAX) >
POST< (COMPONENT | MAX) >
PROB< (COMPONENT | MAX) >

requests that the posterior weights

$$\frac{\pi_j(\mathbf{z}, \boldsymbol{\alpha}_j)\, p_j(y; \mathbf{x}'_j \boldsymbol{\beta}_j, \phi_j)}{\sum_{j=1}^{k} \pi_j(\mathbf{z}, \boldsymbol{\alpha}_j)\, p_j(y; \mathbf{x}'_j \boldsymbol{\beta}_j, \phi_j)}$$

be added to the OUTPUT data set. By default, the probabilities are output for all components. You can limit the output to a single statistic, the largest posterior probability, with the MAX suboption.

NOTE: The keyword "posterior" is used here because of long-standing practice to refer to these probabilities as posterior probabilities. This must not be confused with the posterior distribution in a Bayesian analysis.

LINP
XBETA

requests that the linear predictors for the models be added to the OUTPUT data set.

CLASS | CATEGORY | GROUP

adds the estimated component membership to the OUTPUT data set. An observation is associated with the component that has the highest posterior probability.

MAXPOST | MAXPROB

adds the highest posterior probability to the OUTPUT data set.

A *keyword* can appear multiple times. For example, the following OUTPUT statement requests predicted values for the mixture in addition to the predicted means in the individual components:

```
output out=fmmout pred=MixtureMean pred(component)=CompMean;
```

In a three-component model, this produces four variables in the fmmout data set: MixtureMean, CompMean_1, CompMean_2, and CompMean_3.

You can specify the following *options* in the OUTPUT statement after a slash (/).

ALLSTATS

requests that all statistics are computed. If you do not use a *keyword* to assign a name, the FMM procedure uses the default name.

NOVAR

requests that variables from the input data set not be added to the output data set. This option does not apply to variables listed in the BY statement or to variables listed in the ID statement.

PREDTYPE=PROB | COUNT

specifies the type of predicted values that are produced for a multinomial, binomial, or binary response with *events/trials* syntax. If PREDTYPE=PROB, the predicted values are success probabilities. If PREDTYPE=COUNT, the predicted values are success counts. The default is PREDTYPE=PROB. In multinomial models that do not use *events/trials* syntax, PREDTYPE=COUNT produces predicted counts only if all responses in an observation are nonmissing.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement enables you to control the performance characteristics of the FMM procedure (for example, the number of CPUs, the number of threads for multithreading, and so on). By default, the FMM procedure performs many analyses in multiple threads, and the number of threads equals the number of CPUs. Certain system and configuration options also can control the number of CPUs available to a SAS session or whether multithreaded computations are permissible. For example, you can set the number of available processors to two with

```
options cpucount=2;
```

The FMM procedure then acts as though two processors were available, regardless of the number of physically available processors.

The FMM procedure applies multithreading to the following analytical tasks:

Starting values: all starting value computations that require a pass through the data.

Optimization: all evaluations of objective function, gradient, and Hessian; computation of covariance matrix.

Bayesian analysis: all sample passes through the data, formation of cross-product matrices, sampling of latent variables, and posterior diagnostics.

Scoring and ODS Graphics: computation of all output statistics and statistics for the construction of graphics that require passes through the data.

You can specify the following *performance-options*:

CPUCOUNT=*n* **| ACTUAL**

specifies the number of processors available to the FMM procedure; the number *n* must be between 1 and 1024. CPUCOUNT=ACTUAL sets the number of available processors equal to the number of physical processors.

DETAILS

requests a table with timing detail for the tasks performed by the FMM procedure.

NOTHREADS
disables multithreaded computations.

THREADS=YES | NO
enables or disables multithreaded processing. The number of threads used by the FMM procedure is displayed in the "Bayes Information" or "Optimization Information" table. It typically equals the number of available CPUs, which can be different from the number of physical CPUs, and can be modified with the global CPUCOUNT SAS option or with the CPUCOUNT= option in the PERFORMANCE statement.

PROBMODEL Statement

PROBMODEL < *effects* > < / *probmodel-options* > **;**

The PROBMODEL statement defines the model effects for the mixing probabilities and their link function and starting values. Model effects (other than the implied intercept) are not supported with Bayesian estimation. By default, the FMM procedure models mixing probabilities on the logit scale for two-component models and as generalized logit models in situations with more than two components. The PROBMODEL statement is not required.

The generalized logit model with k categories has a common vector of regressor or design variables, $\mathbf{z}$, $k-1$ parameter vectors that vary with category, and one linear predictor whose value is constant. The constant linear predictor is assigned by the FMM procedure to the last component in the model, and its value is zero ($\boldsymbol{\alpha}_k = \mathbf{0}$). The probability of observing category $1 \le j \le k$ is then

$$\pi_j(\mathbf{z}, \boldsymbol{\alpha}_j) = \frac{\exp\{\mathbf{z}'\boldsymbol{\alpha}_j\}}{\sum_{i=1}^{k} \exp\{\mathbf{z}'\boldsymbol{\alpha}_i\}}$$

For k=2, the generalized logit model reduces to a model with the logit link (a logistic model); hence the attribute *generalized* logit.

By default, an intercept is included in the model for the mixing probabilities. If you suppress the intercept by using the NOINT option, you must specify at least one effect in the statement.

You can specify the following *probmodel-options* in the PROBMODEL statement after the slash (/):

ALPHA=*number*
requests that confidence intervals that have the confidence level $1 - \mathit{number}$ be constructed for the parameters in the probability model. The value of *number* must be between 0 and 1; the default is 0.05. If the probability model is simple—that is, it does not contain any effects—the confidence intervals are produced for the estimated parameters (on the logit scale) and for the mixing probabilities. This option has no effect when you perform Bayesian estimation. You can modify credible interval settings by specifying the STATISTICS(ALPHA=) option in the BAYES statement.

CL
requests that confidence limits be constructed for each of the parameter estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

LINK=*keyword*

specifies the link function in the model for the mixing probabilities. The default is a logit link for models with two components. For models with more than two components, only the generalized logit link is available. The *keywords* and expressions for the associated link functions for two-component models are shown in Table 46.7.

Table 46.7 Link Functions in the PROBMODEL Statement

LINK=	Link Function	$g(\mu) = \eta =$
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$
LOGIT	Logit	$\log(\mu/(1-\mu))$
LOGLOG	Log-log	$-\log(-\log(\mu))$
PROBIT | NORMIT	Probit	$\Phi^{-1}(\mu)$

NOINT

excludes the intercept from the model for the mixing probabilities. An intercept is included by default. If you suppress the intercept by using this option, you must specify at least one other effect for the mixing probabilities, because an empty probability model is not meaningful.

PARAMETERS(*parameter-specification***)**

PARMS(*parameter-specification***)**

specifies starting values for the parameters. The specification of the parameters takes the following form: parameters in the mean function appear in a list, and parameters for different components are separated by commas. Starting values are given on the linked scale, not in terms of probabilities. Also, you need to specify starting values for each of the first k–1 components in a k-component model. The linear predictor for the last component is always assumed to be 0.

The following statements specify a three-component mixture of multiple regression models. The PROBMODEL statement does not list any effects; a standard "intercept-only" generalized logit model is used to model the mixing probabilities.

```
proc fmm;
   model y = x1 x2 / k=3;
   probmodel  / parms(2, 1);
run;
```

There are three linear predictors in the model for the mixing probabilities, α_1, α_2, and α_3. If you have starting values of $\alpha_1 = 2$, $\alpha_2 = 1$, and $\alpha_3 = 0$, this leads to initial mixing probabilities of

$$\pi_1 = \frac{e^2}{e^2 + e^1 + e^0} = 0.24$$

$$\pi_2 = \frac{e^1}{e^2 + e^1 + e^0} = 0.66$$

$$\pi_3 = \frac{e^0}{e^2 + e^1 + e^0} = 0.1$$

You can specify missing values for parameters whose starting values are to be determined by the default method.

RANDSTART Statement

RANDSTART < *randstart-options* > **;**

The RANDSTART statement enables you use randomly generated sets of starting values for mixture model analysis with the maximum likelihood method. This statement is not available for a Bayesian analysis that is performed by the FMM procedure, and it is not available for model selection.

If you specify the RANDSTART statement, PROC FMM determines the maximum likelihood estimates by using a two-stage process. In the first stage, the procedure generates *n1* sets of starting values. To generate the sets of starting values, PROC FMM uses its typical starting values. These values are determined either by the procedure's default method of finding starting values or from the values that you specify in the PARAMETERS option in the MODEL and PROBMODEL statements. This set of values is used as the first starting set. The remaining starting sets are generated around this first set, with random variation controlled by a scale value *s*. You can control this variation by using the STD= option.

The procedure then uses each set of starting values to initialize a separate optimization. Each optimization is terminated according to an initial convergence criterion *r1*. The goal of the first stage is to identify the *n2* ($\leq$ *n1*) sets of starting values that have the largest maximum likelihood values after this optimization. In the second stage, the optimization for these *n2* sets is continued according to a stricter convergence criterion, which you specify by using the convergence criteria in the PROC FMM statement. For more information about this process, see the section "Random Starting Values" on page 3105.

To fine-tune the two-stage random starting values process, you can specify the following *randstart-options*:

ALLITER

includes the iteration history for the first stage of estimation in the "Iteration History" table. By default, only the iteration history for the second stage is displayed.

MAXFIRST=*n*

specifies the maximum number of starting value sets to generate in the first stage.

This option is effective only if the number of starting sets that converge in the first stage is less than the number that you specify in the NSECOND= option. The procedure generates more starting sets than the number that you specify in the NFIRST= option if the number of starting value sets that converge in the first stage is less than the value of the NSECOND= option.

The procedure generates no more than *n* sets, where *n* must be greater than or equal to the values that you specify in the NFIRST= and NSECOND= options.

The default value of the MAXFIRST= option depends on the values of the NFIRST= and NSECOND= options. If you specify NFIRST=*n1*, the default is $2 \times n1$. If you specify NSECOND=*n2* but do not specify a value for the NFIRST= option, the default is $20 \times n2$. If you do not specify either the NFIRST= or NSECOND= option, then by default MAXFIRST=200.

NFIRST=*n1*

specifies the number of starting value sets to generate for the first stage of analysis. The procedure generates at least *n1* starting sets to obtain *n2* starting sets that converge according to the criterion that you specify in the RANDGCONV= option.

When you specify this option, the procedure generates no fewer than *n1* starting sets, where *n1* must be greater than or equal to the value of the NSECOND= option and less than or equal to the value of the MAXFIRST= option.

The default value depends on the values of the NSECOND= and MAXFIRST= options. If you specify MAXFIRST=*n*, the default value is *n*/2. If you specify NSECOND=*n2* but do not specify the MAXFIRST= option, the default value is $10 \times n2$. If you do not specify either the NSECOND= or MAXFIRST= option, then by default NFIRST=100.

NSECOND=*n2*

specifies the number of starting sets to carry forward into the second stage of optimization. These starting sets achieve convergence in the first stage by meeting the convergence criterion that you specify in the RANDGCONV= option. In the second stage, these starting sets continue to be optimized under a stricter convergence criterion that you specify in the PROC FMM statement. At the end of this second stage, the starting set that has the best log likelihood is the set of maximum likelihood estimates that PROC FMM uses to produce all remaining tables and output.

The value of *n2* must be less than or equal to the values that you specify in the NFIRST= and MAXFIRST= options.

The default value depends on the values of the NFIRST= and MAXFIRST= options. If you specify NFIRST=*n1*, the default is *n1*/10. If you do not specify a value for the NFIRST= option but you do specify MAXFIRST=*n*, the default is *n*/20. If you do not specify either the NFIRST= or MAXFIRST= option, then by default NSECOND=10.

RANDGCONV=*r1*

specifies the convergence criterion to use in the first stage of optimization. This option is defined similarly to the GCONV= option in the PROC FMM statement, but it is used only in the first stage of the two-stage random starting method. The value of *r1* must be greater than zero and greater than the value of the GCONV= option.

Each starting set in the first stage undergoes iterative optimization until it converges according to the criterion *r1*. If you do not specify the GCONV= option in the PROC FMM statement, then by default RANDGCONV=1E–2. If you specify GCONV=*r* and the value of *r* is between 0 and 1, the default RANDGCONV= value is the square root of *r*. If you specify GCONV=*r* and the value of *r* is greater than or equal to 1, then by default RANDGCONV=2*r*.

STD=*s*

specifies the standard deviation for the random draw process that generates the starting sets. The value of *s* must be greater than or equal to 1E–1.

The random drawing process centers on the default starting values or the values that you specify in the PARAMETERS option of the MODEL or PROBMODEL statement.

In the random drawing, scale parameters are drawn from a lognormal distribution whose standard deviation is equal to *s*. Mean parameters are drawn from a normal distribution whose standard deviation is equal to *s*.

By default, STD=2.

RESTRICT Statement

RESTRICT < '*label*' > *constraint-specification* < **,** . . . **,** *constraint-specification* >
< *operator* < *value* > > < / *option* > **;**

The RESTRICT statement enables you to specify linear equality or inequality constraints among the parameters of a mixture model. These restrictions are incorporated into the maximum likelihood analysis. This statement is not available for a Bayesian analysis that is performed by the FMM procedure. It is not available for multinomial or Dirichlet-multinomial models that have more than two response variables or more than one event variable.

Following are reasons why you might want to place constraints and restrictions on the model parameters:

- to fix a parameter at a particular value
- to equate parameters in different components in a mixture
- to impose order conditions on the parameters in a model
- to specify contrasts among the parameters that the fitted model should honor

A restriction is composed of a left-hand side and a right-hand side, separated by an operator. If the operator and right-hand side are not specified, the restriction is assumed to be an equality constraint against zero. If the right-hand side is not specified, the value is assumed to be zero.

An individual *constraint-specification* is written in (nearly) the same form as estimable linear functions are specified in the ESTIMATE statement of the GLM, MIXED, or GLIMMIX procedure. The *constraint-specification* takes the form

model-effect value-list < . . . *model-effect value-list* > < **(SCALE=** *value***)** >

At least one *model-effect* must be specified followed by one or more values in the *value-list*. The values in the list correspond to the multipliers of the corresponding parameter that is associated with the position in the model effect. If you specify more values in the *value-list* than the *model-effect* occupies in the model design matrix, the extra coefficients are ignored.

To specify restrictions for effects in specific components in the model, separate the *constraint-specification* by commas. The following statements provide an example:

```
proc fmm;
   class A;
   model y/n = A x /  k = 2;
   restrict A 1 0 -1;
   restrict x 2, x -1 >= 0.5;
run;
```

The linear predictors for this two-component model can be written as

$$\eta_1 = \beta_{10} + \alpha_{11} A_1 + \cdots + \alpha_{1a} A_a + x\beta_{11}$$
$$\eta_2 = \beta_{20} + \alpha_{21} A_1 + \cdots + \alpha_{2a} A_a + x\beta_{21}$$

where A_k is the binary variable associated with the kth level of A.

The first RESTRICT statement applies only to the first component and specifies that the parameter estimates that are associated with the first and third level of the A effect are identical. In terms of the linear predictor, the restriction can be written as

$$\alpha_{11} - \alpha_{13} = 0$$

Now suppose that A has only two levels. Then the FMM procedure ignores the value –1 in the first RESTRICT statement and imposes the restriction

$$\alpha_{11} = 0$$

on the fitted model.

The second RESTRICT statement involves parameters in two different components of the model. In terms of the linear predictors, the restriction can be written as

$$2\beta_{11} - \beta_{21} \geq \frac{1}{2}$$

When restrictions are specified explicitly through the RESTRICT statement or implied through the EQUATE=EFFECTS option in the MODEL statement, the FMM procedure lists all restrictions after the model fit in a table of linear constraints and indicates whether a particular constraint is active at the converged solution.

The following operators can be specified to separate the left- and right-hand sides of the restriction: **=**, **>**, **<**, **>=**, **<=**.

Some distributions involve scale parameters (the parameter ϕ in the expressions of the log likelihood) and you can also use the *constraint-specification* to involve a component's scale parameter in a constraint. To this end, assign a value to the keyword SCALE, separated from the model effects and value lists with parentheses. The following statements fit a two-component normal model and restrict the component variances to be equal:

```
proc fmm;
   model y = / k=2;
   restrict int 0 (scale  1),
            int 0 (scale -1);
run;
```

The intercept specification is necessary because each *constraint-specification* requires at least one model effect. The zero coefficient ensures that the intercepts are not involved in the restriction. Instead, the RESTRICT statement leads to $\phi_1 - \phi_2 = 0$.

You can specify the following *option* in the RESTRICT statement after a slash (/).

DIVISOR=*value*
specifies a *value* by which all coefficients on the right-hand side and left-hand side of the restriction are divided.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement is used to perform a weighted analysis. Consult the section "Log-Likelihood Functions for Response Distributions" on page 3094 for expressions on how weight variables are included in the log-likelihood functions. Because the probability structure of a mixture model is different from that of a classical statistical model, the presence of a weight variable in a mixture model *cannot* be interpreted as altering the variance of an observation.

Observations with nonpositive or missing weights are not included in the PROC FMM analysis. If a WEIGHT statement is not included, all observations used in the analysis are assigned a weight of 1.

Details: FMM Procedure

A Gentle Introduction to Finite Mixture Models

The Form of the Finite Mixture Model

Suppose that you observe realizations of a random variable Y, the distribution of which depends on an unobservable random variable S that has a discrete distribution. S can occupy one of k states, the number of which might be unknown but is at least known to be finite. Because S is not observable, it is frequently referred to as a latent variable.

Let π_j denote the probability that S takes on state j. Conditional on $S = j$, the distribution of the response Y is assumed to be $f_j(y; \alpha_j, \boldsymbol{\beta}_j | S = j)$. In other words, each distinct state j of the random variable S leads to a particular distributional form f_j and set of parameters $\{\alpha_j, \boldsymbol{\beta}_j\}$ for Y.

Let $\{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$ denote the collection of α_j and $\boldsymbol{\beta}_j$ parameters across all $j = 1$ to k. The marginal distribution of Y is obtained by summing the joint distribution of Y and S over the states in the support of S:

$$\begin{aligned} f(y; \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \sum_{j=1}^{k} \Pr(S = j)\, f(y; \alpha_j, \boldsymbol{\beta}_j | S = j) \\ &= \sum_{j=1}^{k} \pi_j\, f(y; \alpha_j, \boldsymbol{\beta}_j | S = j) \end{aligned}$$

This is a mixture of distributions, and the π_j are called the mixture (or prior) probabilities. Because the number of states k of the latent variable S is finite, the entire model is termed a finite mixture (of distributions) model.

The finite mixture model can be expressed in a more general form by representing $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ in terms of regressor variables and parameters with optional additional scale parameters for $\boldsymbol{\beta}$. The section "Notation for the Finite Mixture Model" on page 3022 develops this in detail.

Mixture Models Contrasted with Mixing and Mixed Models: Untangling the Terminology Web

Statistical terminology can have its limitations. The terms mixture, mixing, and mixed models are sometimes used interchangeably, causing confusion. Even worse, the terms arise in related situations. One application needs to be eliminated from the discussion in this documentation: mixture experiments, where design factors are the proportions with which components contribute to a blend, are not mixture models and do not fall under the purview of the FMM procedure. However, the data from a mixture experiment might be analyzed with a mixture model, a mixing model, or a mixed model, besides other types of statistical models.

Suppose that you observe realizations of random variable Y and assume that Y follows some distribution $f(y;\boldsymbol{\alpha},\boldsymbol{\beta})$ that depends on parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Furthermore, suppose that the model is found to be deficient in the sense that the variability implied by the fitted model is less than the observed variability in the data, a condition known as overdispersion (see the section "Overdispersion" on page 3093). To tackle the problem the statistical model needs to be modified to allow for more variability. Clearly, one way of doing this is to introduce additional random variables into the process. Mixture, mixing, and mixed models are simply different ways of adding such random variables. The section "The Form of the Finite Mixture Model" on page 3091 explains how mixture models add a discrete state variable S. The following two subsections explain how mixing and mixed models instead assume variation for a natural parameter or in the mean function.

Mixing Models

Suppose that the model is modified to allow for some random quantity U, which might be one of the parameters of the model or a quantity related to the parameters. Now there are two distributions to cope with: the conditional distribution of the response given the random effect U,

$$f(y;\boldsymbol{\alpha},\boldsymbol{\beta}|u)$$

and the marginal distribution of the data. If U is continuous, the marginal distribution is obtained by integration:

$$f(y;\boldsymbol{\alpha},\boldsymbol{\beta}) = \int f(y;\boldsymbol{\alpha},\boldsymbol{\beta}|u)\, f(u)\, du$$

Otherwise, it is obtained by summation over the support of U:

$$f(y;\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum_u \Pr(U = u)\, f(y;\boldsymbol{\alpha},\boldsymbol{\beta}|u)$$

The important entity for statistical estimation is the marginal distribution $f(y;\boldsymbol{\alpha},\boldsymbol{\beta})$; the conditional distribution is often important for model description, genesis, and interpretation.

In a mixing model the marginal distribution is known and is typically of a well-known form. For example, if $Y|n$ has a binomial(n,μ) distribution and n follows a Poisson distribution, then the marginal distribution of Y is Poisson. The preceding operation is called *mixing* a binomial distribution with a Poisson distribution. Similarly, when mixing a Poisson(λ) distribution with a gamma(a,b) distribution for λ, a negative binomial distribution results as the marginal distribution. Other important mixing models involve mixing a binomial(n,μ) random variable with a beta(a,b) distribution for the binomial success probability μ. This results in a distribution known as the beta-binomial.

The finite mixtures have in common with the mixing models the introduction of random effects into the model to vary some or all of the parameters at random.

Mixed Models

The difference between a mixing and a mixed model is that the conditional distribution is not that important in the mixing model. It matters to motivate the overdispersed reference model and to arrive at the marginal distribution. Inferences with respect to the conditional distribution, such as predicting the random variable U, are not performed in mixing models. In a mixed model the random variable U typically follows a continuous distribution—almost always a normal distribution. The random effects usually do not model the natural parameters of the distribution; instead, they are involved in linear predictors that relate to the conditional mean. For example, a linear mixed model is a model in which the response and the random effects are normally distributed, and the random effects enter the conditional mean function linearly:

$$\begin{aligned}
\mathbf{Y} &= \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{U} + \boldsymbol{\epsilon} \\
\mathbf{U} &\sim N(\mathbf{0}, \mathbf{G}) \\
\boldsymbol{\epsilon} &\sim N(\mathbf{0}, \mathbf{R}) \\
\mathrm{Cov}[\mathbf{U}, \boldsymbol{\epsilon}] &= \mathbf{0}
\end{aligned}$$

The conditional and marginal distributions are then

$$\begin{aligned}
\mathbf{Y}|\mathbf{U} &\sim N(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{U} + \boldsymbol{\epsilon}, \mathbf{R}) \\
\mathbf{Y} &\sim N(\mathbf{X}\boldsymbol{\beta}, \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R})
\end{aligned}$$

For this model, because of the linearity in the mean and the normality of the random effects, you could also refer to mixing the normal vector $\mathbf{Y}$ with the normal vector $\mathbf{U}$, because the marginal distribution is known. The linear mixed model can be fit with the MIXED procedure. When the conditional distribution is not normal and the random effects are normal, the marginal distribution does not have a closed form. In this class of mixed models, called generalized linear mixed models, model approximations and numerical integration methods are commonly used in model fitting; see for example, those models fit by the GLIMMIX and NLMIXED procedures. Chapter 6, "Introduction to Mixed Modeling Procedures," contains details about the various classes of mixed models and about the relevant SAS/STAT procedures.

The previous expression for the marginal variance in the linear mixed model, $\mathrm{var}[\mathbf{Y}] = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$, emphasizes again that the variability in the marginal distribution of a model that contains random effects exceeds the variability in a model without the random effects ($\mathbf{R}$).

The finite mixtures have in common with the mixed models that the marginal distribution is not necessarily a well-known model, but is expressed through a formal integration over the random-effects distribution. In contrast to the mixed models, in particular those involving nonnormal distributions or nonlinear elements, this integration is rather trivial; it reduces to a weighted and finite sum of densities or mass functions.

Overdispersion

Overdispersion is the condition by which the data are more dispersed than is permissible under a reference model. Overdispersion arises only if the variability a model can capture is limited (for example, because of a functional relationship between mean and variance). For example, a model for normal data can never be overdispersed in this sense, although the reasons that lead to overdispersion also negatively affect a misspecified model for normal data. For example, omitted variables increase the residual variance estimate because variability that should have been modeled through changes in the mean is now "picked up" as error variability.

Overdispersion is important because an overdispersed model can lead to misleading inferences and conclusions. However, diagnosing and remedying overdispersion is complicated. In order to handle it appropriately,

the source of overdispersion must be identified. For example, overdispersion can arise from any of the following conditions alone or in combination:

- omitted variables and model effects
- omitted random effects (a source of random variation is not being modeled or is modeled as a fixed effect)
- correlation among the observations
- incorrect distributional assumptions
- incorrectly specified mean-variance relationships
- outliers in the data

As discussed in the previous section, introducing randomness into a system increases its variability. Mixture, mixed, and mixing models have thus been popular in modeling data that appear overdispersed. Finite mixture models are particularly powerful in this regard, because even low-order mixtures of basic, symmetric distributions (such as two- or three-component mixtures of normal or t distributions) enable you to model data with multiple modes, heavy tails, and skewness. In addition, the latent variable S provides a natural way to accommodate omitted, unobservable variables into the model.

One approach to remedy overdispersion is to apply simple modifications of the variance function of the reference model. For example, with binomial-type data this approach replaces the variance of the binomial count variable $Y \sim \text{Binomial}(n, \mu)$, $\text{Var}[Y] = n \times \mu(1-\mu)$ with a scaled version, $\phi n \times \mu(1-\mu)$, where ϕ is called an overdispersion parameter, $\phi > 0$.

In addressing overdispersion problems, it is important to tackle the problem at its root. A missing scale factor on the variance function is hardly ever the root cause of overdispersion; it is only the easiest remedy.

Log-Likelihood Functions for Response Distributions

The FMM procedure calculates the log likelihood that corresponds to a particular response distribution according to the following formulas. The response distribution is the distribution specified (or chosen by default) through the DIST= option in the MODEL statement. The parameterizations used for log-likelihood functions of these distributions were chosen to facilitate expressions in terms of mean parameters that are modeled through an (inverse) link functions and in terms of scale parameters. These are not necessarily the parameterizations in which parameters of prior distributions are specified in a Bayesian analysis of homogeneous mixtures. See the section "Prior Distributions" on page 3104 for details about the parameterizations of prior distributions.

The FMM procedure includes all constant terms in the computation of densities or mass functions. In the expressions that follow, l denotes the log-likelihood function, ϕ denotes a general scale parameter, μ_i is the "mean", and w_i is a weight from the use of a WEIGHT statement.

For some distributions (for example, the Weibull distribution) μ_i is not the mean of the distribution. The parameter μ_i is the quantity that is modeled as $g^{-1}(\mathbf{x}'\boldsymbol{\beta})$, where $g^{-1}(\cdot)$ is the inverse link function and the $\mathbf{x}$ vector is constructed based on the effects in the MODEL statement. Situations in which the parameter μ does not represent the mean of the distribution are explicitly mentioned in the list that follows.

The parameter ϕ is frequently labeled as a "Scale" parameter in output from the FMM procedure. It is not necessarily the scale parameter of the particular distribution.

Beta (μ, ϕ)

$$\begin{aligned} l(\mu_i, \phi; y_i, w_i) &= \log\left\{\frac{\Gamma(\phi/w_i)}{\Gamma(\mu_i\phi/w_i)\Gamma((1-\mu_i)\phi/w_i)}\right\} \\ &+ (\mu_i\phi/w_i - 1)\log\{y_i\} \\ &+ ((1-\mu_i)\phi/w_i - 1)\log\{1-y_i\} \end{aligned}$$

This parameterization of the beta distribution is due to Ferrari and Cribari-Neto (2004) and has properties $\mathrm{E}[Y] = \mu$, $\mathrm{Var}[Y] = \mu(1-\mu)/(1+\phi)$, $\phi > 0$.

Beta-binomial $(n; \mu, \phi)$

$$\begin{aligned} \phi &= (1-\rho^2)/\rho^2 \\ l(\mu_i, \rho; y_i) &= \log\{\Gamma(n_i+1)\} - \log\{\Gamma(y_i+1)\} \\ &- \log\{\Gamma(n_i - y_i + 1)\} \\ &+ \log\{\Gamma(\phi)\} - \log\{\Gamma(n_i+\phi)\} + \log\{\Gamma(y_i+\phi\mu_i)\} \\ &+ \log\{\Gamma(n_i - y_i + \phi(1-\mu_i))\} - \log\{\Gamma(\phi\mu_i)\} \\ &- \log\{\Gamma(\phi(1-\mu_i))\} \\ l(\mu_i, \rho; y_i, w_i) &= w_i l(\mu_i, \rho; y_i) \end{aligned}$$

where y_i and n_i are the *events* and *trials* in the *events/trials* syntax and $0 < \mu < 1$. This parameterization of the beta-binomial model presents the distribution as a special case of the Dirichlet-Multinomial distribution—see, for example, Neerchal and Morel (1998). In this parameterization, $\mathrm{E}[Y] = n\mu$ and $\mathrm{Var}[Y] = n\mu(1-\mu)(1+(n-1)/(\phi+1))$, $0 \le \rho \le 1$. The FMM procedure models the parameter ϕ and labels it "Scale" on the procedure output. For other parameterizations of the beta-binomial model, see Griffiths (1973) or Williams (1975).

Binomial $(n; \mu)$

$$\begin{aligned} l(\mu_i; y_i) &= y_i\log\{\mu_i\} + (n_i - y_i)\log\{1-\mu_i\} \\ &+ \log\{\Gamma(n_i+1)\} - \log\{\Gamma(y_i+1)\} \\ &- \log\{\Gamma(n_i - y_i + 1)\} \\ l(\mu_i; y_i, w_i) &= w_i\, l(\mu_i; y_i) \end{aligned}$$

where y_i and n_i are the *events* and *trials* in the *events/trials* syntax and $0 < \mu < 1$. In this parameterization $\mathrm{E}[Y] = n\mu$, $\mathrm{Var}[Y] = n\mu(1-\mu)$.

Binomial Cluster $(n; \mu, \pi)$

$$\begin{aligned} z &= \log\{\Gamma(n_i+1)\} - \log\{\Gamma(y_i+1)\} - \log\{\Gamma(n_i - y_i + 1)\} \\ \mu_i^* &= (1-\mu_i)\pi_i \end{aligned}$$

$$\begin{aligned} l(\mu_i, \pi_i; y_i) = z + \log\{&\pi_i(\mu_i^* + \mu_i)^{y_i}(1-\mu_i^*-\mu_i)^{n_i-y_i} \\ &+ (1-\pi_i)(\mu_i^*)^{y_i}(1-\mu_i^*)^{n_i-y_i}\} \end{aligned}$$

$$l(\mu_i, \pi_i; y_i, w_i) = w_i l(\mu_i, \pi_i; y_i)$$

In this parameterization, $\mathrm{E}[Y] = n\pi$ and $\mathrm{Var}[Y] = n\pi(1-\pi)\left\{1 + \mu^2(n-1)\right\}$. The binomial cluster model is a two-component mixture of a binomial$(n, \mu^* + \mu)$ and a binomial(n, μ^*) random variable. This mixture is unusual in that it fixes the number of components and because the mixing probability π appears in the moments of the mixture components. For further details, see Morel and Nagaraj (1993); Morel and Neerchal (1997); Neerchal and Morel (1998) and Example 46.1 in this chapter. The expressions for the mean and variance in the binomial cluster model are identical to those of the beta-binomial model shown previously, with $\pi_{bc} = \mu_{bb}$, $\mu_{bc} = \rho_{bb}$.

The FMM procedure models the parameter μ through the MODEL statement and the parameter π through the PROBMODEL statement.

Constant (c)

$$l(y_i) = \begin{cases} 0 & |y_i - c| < \epsilon \\ -1\mathrm{E}20 & |y_i - c| \geq \epsilon \end{cases}$$

The extreme value when $|y_i - c| \geq \epsilon$ is an approximation of $\log(0) = -\infty$, chosen so that $\exp\{l(y_i)\}$ yields a likelihood of 0. You can change this value by using the INVALIDLOGL= option in the PROC FMM statement. The constant distribution is useful for modeling overdispersion due to zero-inflation (or inflation of the process at support c).

The DIST=CONSTANT distribution is useful for modeling an inflated probability of observing a particular value (0, by default) in data from other discrete distributions, as demonstrated in the section "Modeling Zero-Inflation: Is It Better to Fish Poorly or Not to Have Fished at All?" on page 3030. Although it is syntactically valid to mix a constant distribution with a continuous distribution, such as DIST=LOGNORMAL, such a mixture is not mathematically appropriate, because the constant log likelihood is the logarithm of a probability, whereas a continuous log likelihood is the logarithm of a probability density function. If you want to mix a constant distribution with a continuous distribution, you could model the constant as a very narrow continuous distribution, such as DIST=UNIFORM$(c-\delta, c+\delta)$ for a small value δ. However, fitting this type of mixture model can be sensitive to roundoff error and other numerical inaccuracies. Instead, the following approach is mathematically equivalent and more numerically stable:

1. Estimate the mixing probability $\mathrm{P}(Y = c)$ as the proportion of observations in the data set such that $|y_i - c| < \epsilon$.
2. Estimate the parameters of the continuous distribution from the observations for which $|y_i - c| \geq \epsilon$.

Dirichlet-multinomial $(n, \boldsymbol{\mu}, \phi)$

$$\begin{aligned} l(\boldsymbol{\mu}_i, \phi; \mathbf{y}_i) &= \log\{\Gamma(n_i + 1)\} + \log\{\Gamma(\phi)\} + \sum_{j=1}^{M} \log\{\Gamma(y_{ij} + \phi\mu_{ij})\} \\ &\quad - \sum_{j=1}^{M} \log\{\Gamma(y_{ij} + 1)\} - \sum_{j=1}^{M} \log\{\Gamma(\phi\mu_{ij})\} - \log\{\Gamma(\sum_{j=1}^{M}[y_{ij} + \phi\mu_{ij}])\} \\ l(\boldsymbol{\mu}_i, \phi; \mathbf{y}_i, w_i) &= w_i l(\boldsymbol{\mu}_i, \phi; \mathbf{y}_i) \end{aligned}$$

where M is the number of levels of the response variable, $n_i = \sum_{j=1}^{M} y_{ij}$, $0 < \mu_{ij} < 1$ for each j, and $\mu_{iM} = 1 - \sum_{j=1}^{M-1} \mu_{ij}$.

The mean and variance are $\mathrm{E}[\mathbf{Y}] = n\boldsymbol{\mu}$ and $\mathrm{Var}[\mathbf{Y}] = n\left(\mathrm{Diag}(\boldsymbol{\mu}) - \boldsymbol{\mu}\boldsymbol{\mu}^T\right)\left(\frac{n+\phi}{1+\phi}\right)$, respectively.

Exponential (μ)

$$l(\mu_i; y_i, w_i) = \begin{cases} -\log\{\mu_i\} - y_i/\mu_i & w_i = 1 \\ w_i \log\left\{\frac{w_i y_i}{\mu_i}\right\} - \frac{w_i y_i}{\mu_i} - \log\{y_i \Gamma(w_i)\} & w_i \neq 1 \end{cases}$$

In this parameterization, $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \mu^2$.

Folded Normal (μ, ϕ)

$$\begin{aligned} l(\mu_i, \phi; y_i, w_i) = &-\frac{1}{2}\log\{2\pi\} - \frac{1}{2}\log\{\phi/w_i\} \\ &+ \log\left\{\exp\left\{\frac{-w_i(y_i - \mu_i)^2}{2\phi}\right\} + \exp\left\{\frac{-w_i(y_i + \mu_i)^2}{2\phi}\right\}\right\} \end{aligned}$$

If X has a normal distribution with mean μ and variance ϕ, then $Y = |X|$ has a folded normal distribution and log-likelihood function $l(\mu, \phi; y, w)$ for $y \geq 0$. The folded normal distribution arises, for example, when normally distributed measurements are observed, but their signs are not observed. The mean and variance of the folded normal in terms of the underlying $\mathrm{N}(\mu, \phi)$ distribution are

$$\begin{aligned} \mathrm{E}[Y] =& \frac{1}{\sqrt{2\pi\phi}} \exp\left\{-\frac{\mu^2}{2/\phi}\right\} + \mu\left(1 - 2\Phi\left(-\mu/\sqrt{\phi}\right)\right) \\ \mathrm{Var}[Y] =& \phi + \mu^2 - \mathrm{E}[Y]^2 \end{aligned}$$

The FMM procedure models the folded normal distribution through the mean μ and variance ϕ of the underlying normal distribution. When the FMM procedure computes output statistics for the response variable (for example when you use the OUTPUT statement), the mean and variance of the response Y are reported. Similarly, the fit statistics apply to the distribution of $Y = |X|$, not the distribution of X. When you model a folded normal variable, the response input variable should be positive; the FMM procedure treats negative values of Y as a support violation.

Gamma (μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = w_i\phi \log\left\{\frac{w_i y_i \phi}{\mu_i}\right\} - \frac{w_i y_i \phi}{\mu_i} - \log\{y_i\} - \log\{\Gamma(w_i\phi)\}$$

In this parameterization, $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \mu^2/\phi$, respectively, where $\phi > 0$. This parameterization of the gamma distribution differs from that in the GLIMMIX procedure, which expresses the log-likelihood function in terms of $1/\phi$ in order to achieve a variance function suitable for mixed model analysis.

Geometric (μ)

$$\begin{aligned} l(\mu_i; y_i, w_i) = &\, y_i \log\left\{\frac{\mu_i}{w_i}\right\} - (y_i + w_i)\log\left\{1 + \frac{\mu_i}{w_i}\right\} \\ &+ \log\left\{\frac{\Gamma(y_i + w_i)}{\Gamma(w_i)\Gamma(y_i + 1)}\right\} \end{aligned}$$

In this parameterization, $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \mu + \mu^2$. The geometric distribution is a special case of the negative binomial distribution with $\phi = 1$.

Generalized Poisson (μ, ϕ)

$$\begin{aligned}
\xi_i &= 1 - \exp\{-\phi\}/w_i \\
\mu_i^* &= \mu_i - \xi(\mu_i - y_i) \\
l(\mu_i^*, \xi_i; y_i, w_i) &= \log\{\mu_i^* - \xi_i y_i\} + (y_i - 1)\log\{\mu_i^*\} \\
&\quad - \mu_i^* - \log\{\Gamma(y_i + 1)\}
\end{aligned}$$

In this parameterization, $\mathrm{E}[Y] = \mu$, $\mathrm{Var}[Y] = \mu/(1-\xi)^2$, and $\phi \geq 0$. The FMM procedure models the mean μ through the effects in the MODEL statement and applies a log link by default. The generalized Poisson distribution provides an overdispersed alternative to the Poisson distribution; $\phi = \xi_i = 0$ produces the mass function of a regular Poisson random variable. For details about the generalized Poisson distribution and a comparison with the negative binomial distribution, see Joe and Zhu (2005).

Inverse Gaussian (μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{y_i \phi \mu_i^2} + \log\left\{\frac{\phi y_i^3}{w_i}\right\} + \log\{2\pi\}\right]$$

The variance is $\mathrm{Var}[Y] = \phi\mu^3$, $\phi > 0$.

Lognormal (μ, ϕ)

$$\begin{aligned}
z_i &= \log\{y_i\} - \mu_i \\
l(\mu_i, \phi; y_i, w_i) &= -\frac{1}{2}\left(2\log\{y_i\} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\} + \frac{w_i z_i^2}{\phi}\right)
\end{aligned}$$

If $X = \log\{Y\}$ has a normal distribution with mean μ and variance ϕ, then Y has the log-likelihood function $l(\mu_i, \phi; y_i, w_i)$. The FMM procedure models the lognormal distribution and not the "shortcut" version you can obtain by taking the logarithm of a random variable and modeling that as normally distributed. The two approaches are not equivalent, and the approach taken by PROC FMM is the actual lognormal distribution. Although the lognormal model is a member of the exponential family of distributions, it is not in the "natural" exponential family because it cannot be written in canonical form.

In terms of the parameters μ and ϕ of the underlying normal process for X, the mean and variance of Y are $\mathrm{E}[Y] = \exp\{\mu\}\sqrt{\omega}$ and $\mathrm{Var}[Y] = \exp\{2\mu\}\omega(\omega - 1)$, respectively, where $\omega = \exp\{\phi\}$. When you request predicted values with the OUTPUT statement, the FMM procedure computes $\mathrm{E}[Y]$ and not μ.

Multinomial $(n, \boldsymbol{\mu})$

$$\begin{aligned}
l(\boldsymbol{\mu}_i; \mathbf{y}_i) &= \sum_{j=1}^{M} y_{ij}\log(\mu_{ij}) + \log\{\Gamma(n_i + 1)\} - \sum_{j=1}^{M}\log\{\Gamma(y_{ij} + 1)\} \\
l(\boldsymbol{\mu}_i; \mathbf{y}_i, w_i) &= w_i l(\boldsymbol{\mu}_i, \mathbf{y}_i)
\end{aligned}$$

where M is the number of levels of the response variable, $n_i = \sum_{j=1}^{M} y_{ij}$, $0 < \mu_{ij} < 1$ for each j, and $\sum_{j=1}^{M} \mu_{ij} = 1$.

The mean and variance are $\mathrm{E}[\mathbf{Y}] = n\boldsymbol{\mu}$ and $\mathrm{Var}[\mathbf{Y}] = n(\mathrm{Diag}(\boldsymbol{\mu}) - \boldsymbol{\mu}\boldsymbol{\mu}^T)$, respectively, where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_{M-1})$.

Multinomial Cluster $(n, \mu, \boldsymbol{\pi})$

$$K(\mu_i, \boldsymbol{\pi}_i, \mathbf{y}_i) = \sum_{h=1}^{M} \left\{ \pi_{ih} \prod_{g=1}^{M} [\pi_{ig}(1-\mu_i) + \mu_i 1(g=h)]^{y_{ig}} \right\}$$

$$l(\mu_i, \boldsymbol{\pi}_i, \mathbf{y}_i) = \log\{\Gamma(n_i+1)\} - \sum_{g=1}^{M} \log\{\Gamma(y_{ij}+1)\} + \log\{K(\mu_i, \boldsymbol{\pi}_i, \mathbf{y}_i)\}$$

$$l(\mu_i, \boldsymbol{\pi}_i, \mathbf{y}_i, w_i) = w_i l(\mu_i, \boldsymbol{\pi}_i, \mathbf{y}_i)$$

The multinomial cluster model for a response that has M levels is an M-component mixture of multinomials. The parameter vector for each component multinomial is based on the mixing probability vector $\boldsymbol{\pi}$ and an additional factor μ. As is true for the binomial cluster model, the number of components is fixed and the mixing probabilities in $\boldsymbol{\pi}$ appear in the moments for the mixture components. For more information, see Morel and Nagaraj (1993) and Example 46.4 in this chapter.

The mean and variance are $\mathrm{E}[\mathbf{Y}] = n\boldsymbol{\pi}$ and $\mathrm{Var}[\mathbf{Y}] = n(\mathrm{Diag}(\boldsymbol{\pi}) - \boldsymbol{\pi}\boldsymbol{\pi}^T)\{1+\mu^2(n-1)\}$, respectively.

Negative Binomial (μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = y_i \log\left\{\frac{\phi\mu_i}{w_i}\right\} - (y_i + w_i/\phi)\log\left\{1 + \frac{\phi\mu_i}{w_i}\right\} + \log\left\{\frac{\Gamma(y_i + w_i/\phi)}{\Gamma(w_i/\phi)\Gamma(y_i+1)}\right\}$$

The variance is $\mathrm{Var}[Y] = \mu + \phi\mu^2$, $\phi > 0$.

For a given ϕ, the negative binomial distribution is a member of the exponential family. The parameter ϕ is related to the scale of the data because it is part of the variance function. However, it cannot be factored from the variance, as is the case with the ϕ parameter in many other distributions.

Normal (μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{\phi} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\}\right]$$

The mean and variance are $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \phi$, respectively, where $\phi > 0$.

Poisson (μ)

$$l(\mu_i; y_i, w_i) = w_i(y_i \log\{\mu_i\} - \mu_i - \log\{\Gamma(y_i+1)\})$$

The mean and variance are $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \mu$, respectively.

(Shifted) T $(\nu; \mu, \phi)$

$$z_i = -0.5\log\{\phi/w_i\} + \log\{\Gamma(0.5(\nu+1)\} - \log\{\Gamma(0.5\nu)\} - 0.5 \times \log\{\pi\nu\}$$

$$l(\mu_i, \phi; y_i, w_i) = -\left(\frac{\nu+1}{2}\right)\log\left\{1 + \frac{w_i}{\nu}\frac{(y_i - \mu_i)^2}{\phi}\right\} + z_i$$

$\nu > 0, \phi > 0$. In this parameterization, $\mathrm{E}[Y] = \mu$ when $\nu > 1$ and $\mathrm{Var}[Y] = \phi\nu/(\nu - 2)$ when $\nu > 2$. Note that this form of the t distribution is not a noncentral distribution, but that of a shifted central t random variable.

Truncated Exponential $(\mu; a, b)$

$$l(\mu_i; a, b, y_i, w_i) = w_i \log\left\{\frac{w_i y_i}{\mu_i}\right\} - \frac{w_i y_i}{\mu_i} - \log\{y_i \Gamma(w_i)\} - \log\left[\frac{\gamma\left(w_i, \frac{w_i b}{\mu_i}\right)}{\Gamma(w_i)} - \frac{\gamma\left(w_i, \frac{w_i a}{\mu_i}\right)}{\Gamma(w_i)}\right]$$

where

$$\gamma(c_1, c_2) = \int_0^{c_2} t^{c_1 - 1} \exp(-t)\mathrm{d}t$$

is the lower incomplete gamma function. The mean and variance are

$$\mathrm{E}[Y] = \frac{(a + \mu_i)\exp(-a/\mu_i) - (b + \mu_i)\exp(-b/\mu_i)}{\exp(-a/\mu_i) - \exp(-b/\mu_i)}$$

$$\mathrm{Var}[Y] = \frac{(a^2 + 2a\mu_i + 2\mu_i^2)\exp(-a/\mu_i) - (b^2 + 2b\mu_i + 2\mu_i^2)\exp(-b/\mu_i)}{\exp(-a/\mu_i) - \exp(-b/\mu_i)} - (\mathrm{E}[Y])^2$$

Truncated Lognormal $(\mu, \phi; a, b)$

$$z_i = \log\{y_i\} - \mu_i$$

$$l(\mu_i, \phi; a, b, y_i, w_i) = -\frac{1}{2}\left(2\log\{y_i\} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\} + \frac{w_i z_i^2}{\phi}\right) - \log\left\{\Phi\left[\sqrt{w_i/\phi}(\log b - \mu_i)\right] - \Phi\left[\sqrt{w_i/\phi}(\log a - \mu_i)\right]\right\}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. The mean and variance are

$$\mathrm{E}[Y] = \exp(\mu_i + 0.5\phi)\frac{\Phi\left(\sqrt{\phi} - \frac{\log a - \mu_i}{\sqrt{\phi}}\right) - \Phi\left(\sqrt{\phi} - \frac{\log b - \mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{\log b - \mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{\log a - \mu_i}{\sqrt{\phi}}\right)}$$

$$\mathrm{Var}[Y] = \exp(2\mu_i + 2\phi)\frac{\Phi\left(2\sqrt{\phi} - \frac{\log a - \mu_i}{\sqrt{\phi}}\right) - \Phi\left(2\sqrt{\phi} - \frac{\log b - \mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{\log b - \mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{\log a - \mu_i}{\sqrt{\phi}}\right)} - (\mathrm{E}[Y])^2$$

Truncated Negative Binomial (μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = y_i \log\left\{\frac{\phi\mu_i}{w_i}\right\} - (y_i + w_i/\phi)\log\left\{1 + \frac{\phi\mu_i}{w_i}\right\} + \log\left\{\frac{\Gamma(y_i + w_i/\phi)}{\Gamma(w_i/\phi)\Gamma(y_i + 1)}\right\} - \log\left\{1 - \left(\frac{\phi\mu_i}{w_i} + 1\right)^{-w_i/\phi}\right\}$$

The mean and variance are

$$\mathrm{E}[Y] = \mu_i \left\{1 - (\phi\mu_i + 1)^{-1/\phi}\right\}^{-1}$$
$$\mathrm{Var}[Y] = (1 + \phi\mu_i + \mu_i)\mathrm{E}[Y] - (\mathrm{E}[Y])^2$$

Truncated Normal $(\mu, \phi; a, b)$

$$l(\mu_i, \phi; a, b, y_i, w_i) = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{\phi} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\}\right] - \log\left\{\Phi\left[\sqrt{w_i/\phi}(b - \mu_i)\right] - \Phi\left[\sqrt{w_i/\phi}(a - \mu_i)\right]\right\}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. The mean and variance are

$$\mathrm{E}[Y] = \mu_i + \sqrt{\phi}\frac{\mathrm{phi}\left(\frac{a-\mu_i}{\sqrt{\phi}}\right) - \mathrm{phi}\left(\frac{b-\mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{b-\mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{a-\mu_i}{\sqrt{\phi}}\right)}$$
$$\mathrm{Var}[Y] = \phi\left[1 + \frac{\frac{a-\mu_i}{\sqrt{\phi}}\mathrm{phi}\left(\frac{a-\mu_i}{\sqrt{\phi}}\right) - \frac{b-\mu_i}{\sqrt{\phi}}\mathrm{phi}\left(\frac{b-\mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{b-\mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{a-\mu_i}{\sqrt{\phi}}\right)} - \left\{\frac{\mathrm{phi}\left(\frac{a-\mu_i}{\sqrt{\phi}}\right) - \mathrm{phi}\left(\frac{b-\mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{b-\mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{a-\mu_i}{\sqrt{\phi}}\right)}\right\}^2\right]$$

where $\mathrm{phi}(\cdot)$ is the probability density function of the standard normal distribution.

Truncated Poisson (μ)

$$l(\mu_i; y_i, w_i) = w_i(y_i \log\{\mu_i\} - \log\{\exp(\mu_i) - 1\} - \log\{\Gamma(y_i + 1)\})$$

The mean and variance are

$$\mathrm{E}[Y] = \frac{\mu_i}{1 - \exp(-\mu_i)}$$
$$\mathrm{Var}[Y] = \frac{\mu_i\,[1 - \exp(-\mu_i) - \mu_i \exp(-\mu_i)]}{[1 - \exp(-\mu_i)]^2}$$

Uniform (a, b)

$$l(\mu_i; y_i, w_i) = -\log\{b - a\}$$

The mean and variance are $\mathrm{E}[Y] = 0.5(a + b)$ and $\mathrm{Var}[Y] = (b - a)^2/12$, respectively.

Weibull (μ, ϕ)

$$l(\mu_i, \phi; y_i) = -w_i\left\{\frac{\phi - 1}{\phi}\log\left(\frac{y_i}{\mu_i}\right) + \log(\mu_i\phi) + \exp\left[\log\left(\frac{y_i}{\mu_i}\right)/\phi\right]\right\}$$

In this particular parameterization of the two-parameter Weibull distribution, the mean and variance of the random variable Y are $\mathrm{E}[Y] = \mu\Gamma(1 + \phi)$ and $\mathrm{Var}[Y] = \mu^2\left\{\Gamma(1 + 2\phi) - \Gamma^2(1 + \phi)\right\}$, respectively.

Bayesian Analysis

Conjugate Sampling

The FMM procedure uses Bayesian analysis via a conjugate Gibbs sampler if the model belongs to a small class of mixture models for which a conjugate sampler is available. See the section "Gibbs Sampler" on page 157 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a general discussion of Gibbs sampling. Table 46.8 summarizes the models for which conjugate and Metropolis-Hastings samplers are available.

Table 46.8 Availability of Conjugate and Metropolis Samplers in the FMM Procedure

Effects (exclusive of intercept)	Distributions	Available Samplers
No	Normal or T	Conjugate or Metropolis-Hastings
Yes	Normal or T	Conjugate or Metropolis-Hastings
No	Binomial, binary, Poisson, exponential	Conjugate or Metropolis-Hastings
Yes	Binomial, binary, Poisson, exponential	Metropolis-Hastings only

The conjugate sampler enjoys greater efficiency than the Metropolis-Hastings sampler and has the advantage of sampling in terms of the natural parameters of the distribution.

You can always switch to the Metropolis-Hastings sampling algorithm in any model by adding the METROPOLIS option in the BAYES statement.

Metropolis-Hastings Algorithm

If Metropolis-Hastings is the only sampler available for the specified model (see Table 46.8) or if the METROPOLIS option is specified in the BAYES statement, PROC FMM uses the Metropolis-Hastings approach of Gamerman (1997). See the section "Metropolis and Metropolis-Hastings Algorithms" on page 156 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a general discussion of the Metropolis-Hastings algorithm.

The Gamerman (1997) algorithm derives a specific density that is used to generate proposals for the component-specific parameters $\boldsymbol{\beta}_j$. The form of this proposal density is multivariate normal, with mean $\mathbf{m}_j$ and covariance matrix $\mathbf{C}_j$ derived as follows.

Suppose $\boldsymbol{\beta}_j$ is the vector of model coefficients in the jth component and suppose that $\boldsymbol{\beta}_j$ has prior distribution $N(\mathbf{a}, \mathbf{R})$. Consider a generalized linear model (GLM) with link function $g(\mu) = \eta = \mathbf{x}'\boldsymbol{\beta}$ and variance function $a(\mu)$. The pseudo-response and weight in the GLM for a weighted least squares step are

$$y^* = \eta + (y - \mu)/\frac{\partial \mu}{\partial \eta}$$

$$w = \frac{\partial \mu}{\partial \eta}/a(\mu)$$

If the model contains offsets or FREQ or WEIGHT statements, or if a *trials* variable is involved, suitable adjustments are made to these quantities.

In each component, $j = 1, \ldots, k$, form an adjusted cross-product matrix with a "pseudo" border

$$\begin{bmatrix} \mathbf{X}_j'\mathbf{W}_j\mathbf{X}_j + \mathbf{R}^{-1} & \mathbf{X}_j'\mathbf{W}_j\mathbf{y}_j^* + \mathbf{R}^{-1}\mathbf{a} \\ \mathbf{y}_j^{*\prime}\mathbf{W}_j\mathbf{X}_j + \mathbf{a}'\mathbf{R}^{-1} & c \end{bmatrix}$$

where $\mathbf{W}_j$ is a diagonal matrix formed from the pseudo-weights w, $\mathbf{y}^*$ is a vector of pseudo-responses, and c is arbitrary. This is basically a system of normal equations with ridging, and the degree of ridging is governed by the precision and mean of the normal prior distribution of the coefficients. Sweeping on the leading partition leads to

$$\begin{aligned} \mathbf{C}_j &= \left(\mathbf{X}_j'\mathbf{W}_j\mathbf{X}_j + \mathbf{R}^{-1}\right)^- \\ \mathbf{m}_j &= \mathbf{C}_j\left(\mathbf{X}_j'\mathbf{W}_j\mathbf{y}_j^* + \mathbf{R}^{-1}\mathbf{a}\right) \end{aligned}$$

where the generalized inverse is a reflexive, g_2-inverse (see the section "Linear Model Theory" on page 55 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," for details).

PROC FMM then generates a proposed parameter vector from the resulting multivariate normal distribution, and then accepts or rejects this proposal according to the appropriate Metropolis-Hastings thresholds.

Latent Variables via Data Augmentation

In order to fit finite Bayesian mixture models, the FMM procedure treats the mixture model as a missing data problem and introduces an assignment variable **S** as in Dempster, Laird, and Rubin (1977). Because **S** is not observable, it is frequently referred to as a latent variable. The unobservable variable **S** assigns an observation to a component in the mixture model. The number of states, k, might be unknown, but it is known to be finite. Conditioning on the latent variable **S**, the component memberships of each observation is assumed to be known, and Bayesian estimation is straightforward for each component in the finite mixture model. That is, conditional on $S = j$, the distribution of the response is now assumed to be $f(y; \alpha_j, \beta_j | S = j)$. In other words, each distinct state of the random variable **S** leads to a distinct set of parameters. The parameters in each component individually are then updated using a conjugate Gibbs sampler (where available) or a Metropolis-Hastings sampling algorithm.

The FMM procedure assumes that the random variable **S** has a discrete multinomial distribution with probability π_j of belonging to a component j; it can occupy one of k states. The distribution for the latent variable **S** is

$$f(S_i = j | \pi_1, \ldots, \pi_k) = \text{multinomial}(1, \pi_1, \ldots, \pi_k)$$

where $f(\cdot|\cdot)$ denotes a conditional probability density. The parameters in the density π_j denote the probability that S takes on state j.

The FMM procedure assumes a conjugate Dirichlet prior distribution on the mixture proportions π_j written as:

$$p(\boldsymbol{\pi}) \quad = \quad \text{Dirichlet}(a_1, \ldots, a_k)$$

where $p(\cdot)$ indicates a prior distribution.

Using Bayes' theorem, the likelihood function and prior distributions determine a conditionally conjugate posterior distribution of **S** and $\boldsymbol{\pi}$ from the multinomial distribution and Dirichlet distribution, respectively.

Prior Distributions

The following list displays the parameterization of prior distributions for situations in which the FMM procedure uses a conjugate sampler in mixture models without model effects and certain basic distributions (binary, binomial, exponential, Poisson, normal, and *t*). You specify the parameters *a* and *b* in the formulas below in the MUPRIORPARMS= and PHIPRIORPARMS= options in the BAYES statement in these models.

Beta(a, b)

$$f(y) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\, y^{a-1}\,(1-y)^{b-1}$$

where $a > 0$, $b > 0$. In this parameterization, the mean and variance of the distribution are $\mu = a/(a+b)$ and $\mu(1-\mu)/(a+b+1)$, respectively. The beta distribution is the prior distribution for the success probability in binary and binomial distributions when conjugate sampling is used.

Dirichlet$(a_1, \ldots, a_k)$

$$f(\mathbf{y}) = \frac{\Gamma\left(\sum_{i=1}^{k} a_i\right)}{\prod_{i=1}^{k}\Gamma(a_i)}\, y_1^{a_1-1}\,\ldots\, y_k^{a_k-1}$$

where $\sum_{i=1}^{k} y_i = 1$ and the parameters $a_i > 0$. If any a_i were zero, an improper density would result. The Dirichlet density is the prior distribution for the mixture probabilities. You can affect the choice of the a_i through the MIXPRIORPARMS option in the BAYES statement. If *k*=2, the Dirichlet is the same as the beta(a, b) distribution.

Gamma(a, b)

$$f(y) = \frac{b^a}{\Gamma(a)}\, y^{a-1}\,\exp\{-by\}$$

where $a > 0$, $b > 0$. In this parameterization, the mean and variance of the distribution are $\mu = a/b$ and μ/b, respectively. The gamma distribution is the prior distribution for the mean parameter of the Poisson distribution when conjugate sampling is used.

Inverse gamma(a, b)

$$f(y) = \frac{b^a}{\Gamma(a)}\, y^{-a-1}\,\exp\{-b/y\}$$

where $a > 0$, $b > 0$. In this parameterization, the mean and variance of the distribution are $\mu = b/(a-1)$ if $a > 1$ and $\mu^2/(a-2)$ if $a > 2$, respectively. The inverse gamma distribution is the prior distribution for the mean parameter of the exponential distribution when conjugate sampling is used. It is also the prior distribution for the scale parameter ϕ in all models.

Multinomial$(1, \pi_1, \ldots, \pi_k)$

$$f(\mathbf{y}) = \frac{1}{y_1!\cdots y_k!}\pi_1^{y_1}\,\cdots\,\pi_k^{y_k}$$

where $\sum_{j=1}^{k} y_j = n$, $y_j \geq 0$, $\sum_{j=1}^{k}\pi_j = 1$, and n is the number of observations included in the analysis. The multinomial density is the prior distribution for the mixture proportions. The mean and variance of Y_j are $\mu_j = \pi_j$ and $\mu_j(1-\mu_j)$, respectively.

Normal(a, b)

$$f(y) = \frac{a}{\sqrt{2\pi b}} \exp\left\{-\frac{1}{2}\frac{(y-a)^2}{b}\right\}$$

where $b > 0$. The mean and variance of the distribution are $\mu = a$ and b, respectively. The normal distribution is the prior distribution for the mean parameter of the normal and t distribution when conjugate sampling is used.

When a MODEL statement contains effects or if you specify the METROPOLIS option in the BAYES statement, the prior distribution for the regression parameters is multivariate normal, and you can specify the means and variances of the parameters in the BETAPRIORPARMS= option in the BAYES statement.

Parameterization of Model Effects

PROC FMM constructs a finite mixture model according to the specifications in the CLASS, MODEL, and PROBMODEL statements. Each effect in the MODEL statement generates one or more columns in the matrix **X** for that model. The same **X** matrix applies to all components that are associated with the MODEL statement. Each effect in the PROBMODEL statement generates one or more columns in the matrix **Z** from which the linear predictors in the model for the mixture probability models is formed. The same **Z** matrix applies to all components.

The formation of effects from continuous and classification variables in the FMM procedure follows the same general rules and techniques as for other linear modeling procedures. See the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

Random Starting Values

The iterative optimization steps for maximum likelihood estimation of mixture models might not always find estimates that achieve a global maximum or even a best local maximum in likelihood. One way to alleviate this problem is to generate multiple sets of starting values for the optimization and find the best solution over those sets. PROC FMM supports this approach by automating a two-stage method that generates multiple sets of random starting values in maximum likelihood estimation. You can use this two-stage method by specifying the RANDSTART statement.

In the first stage, the procedure uses its default starting values as the first set of starting values. If you specify starting values by using the PARAMETERS option in either the MODEL statement or the PROBMODEL statement, PROC FMM uses those values as the first set of starting values instead. Then, the procedure randomly draws multiple sets of starting values around the values of the first random set; the variability is controlled by a standard deviation parameter, which you can adjust by using the STD= option. You can use the NFIRST= option to control the number of sets that the procedure generates in the first stage. The procedure repeatedly optimizes the model, using each set of random starting values in turn to start the optimization. It uses the convergence criterion that you specify in the RANDGCONV= option to terminate each fit in the first stage.

In the second stage, PROC FMM identifies the sets that have the best log likelihood at the end of the first stage and continues the fit for each of those sets. You can use the NSECOND= option to control the number

of sets to carry forward from the first stage. If not enough of the first-stage sets converge to supply the second stage, the procedure generates additional sets and optimizes by using those sets. The procedure will not generate more than the maximum number of sets that you specify in the MAXFIRST= option. At the end of the second stage, the procedure identifies the model that has the best fit, and it uses the corresponding converged estimates as the maximum likelihood estimates for the model.

The "Random Start Information" table displays the values that control the two-stage fitting process.

The "Random Start Convergence" table displays the random starting sets, the final objective function from optimization, and an indicator of convergence. This table contains identifiers for the starting sets and the stages of fitting. These identifiers are also displayed in the "Iteration History" table.

The "Parameter Mapping" table provides a mapping between the parameters in the "Random Start Convergence" table and their roles in the associated mixture component.

Default Output

The following sections describe the output that PROC FMM produces by default. The output is organized into various tables, which are discussed in the order of appearance for maximum likelihood and Bayesian estimation, respectively.

Model Information

The "Model Information" table displays basic information about the model, such as the response variable, frequency variable, link function, and the model category that the FMM procedure determined based on your input and options. The "Model Information" table is one of a few tables that are produced irrespective of estimation technique. Most other tables are specific to Bayes or maximum likelihood estimation.

If the analysis depends on generated random numbers, the "Model Information" table also displays the random number seed used to initialize the random number generators. If you repeat the analysis and pass this seed value in the SEED= option in the PROC FMM statement, an identical stream of random numbers results.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels with the ORDER= option in the PROC FMM statement. You can suppress the "Class Level Information" table completely or partially with the NOCLPRINT option in the PROC FMM statement.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used. If the *events/trials* syntax is used for the response, the table also displays the number of events and trials used in the analysis.

Note that the number of observations "used" in the analysis is not unambiguous in a mixture model. An observation that is "unusable" for one component distribution (because the response value is outside of the support of the distribution) might still be usable in the mixture model when the response value is in the

support of another component distribution. You can affect the way in which PROC FMM handles exclusion of observations due to support violations with the EXCLUSION= option in the PROC FMM statement.

Response Profile

For binary data, the "Response Profile" table displays the ordered value from which the FMM procedure determines the probability being modeled as an event for binary data. For each response category level, the frequency used in the analysis is reported.

Default Output for Maximum Likelihood

Optimization Information

The "Optimization Information" table displays basic information about the optimization setup to determine the maximum likelihood estimates, such as the optimization technique, the parameters that participate in the optimization, and the number of threads used for the calculations. This table is not produced during model selection—that is, if the KMAX= option is specified in the MODEL statement.

Iteration History

The "Iteration History" table displays for each iteration of the optimization the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element. The objective function used in the optimization in the FMM procedure is the negative of the mixture log likelihood; consequently, PROC FMM performs a minimization. This table is not produced if you specify the KMAX= option in the MODEL statement. If you wish to see the "Iteration History" table in this setting, you must also specify the FITDETAILS option in the PROC FMM statement.

Convergence Status

The convergence status table is a small ODS table that follows the "Iteration History" table in the default output. In the listing, it appears as a message that identifies whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the "Convergence Status" table to an output data set, a numeric Status variable is added that allows you to assess convergence programmatically. The values of the Status variable encode the following:

0 Convergence was achieved or an optimization was not performed (because of TECHNIQUE=NONE).

1 The objective function could not be improved.

2 Convergence was not achieved because of a user interrupt or because a limit was exceeded, such as the maximum number of iterations or the maximum number of function evaluations. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC FMM statement.

3 Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Fit Statistics

The "Fit Statistics" table displays a variety of fit measures based on the mixture log likelihood in addition to the Pearson statistic. All statistics are presented in "smaller is better" form. If you are fitting a single-component normal, gamma, or inverse Gaussian model, the table also contains the unscaled Pearson statistic. If you are fitting a mixture model or the model has been fitted under restrictions, the table also contains the number of effective components and the number of effective parameters.

The calculation of the information criteria uses the following formulas, where p denotes the number of effective parameters, n denotes the number of observations used (or the sum of the frequencies used if a FREQ statement is present), and l is the log likelihood of the mixture evaluated at the converged estimates:

$$\mathrm{AIC} = -2l + 2p$$

$$\mathrm{AICC} = \begin{cases} -2l + 2pn/(n-p-1) & n > p+2 \\ -2l + 2p(p+2) & \text{otherwise} \end{cases}$$

$$\mathrm{BIC} = -2l + p\log(n)$$

The Pearson statistic is computed simply as

$$\text{Pearson statistic} = \sum_{i=1}^{n} f_i \frac{(y_i - \widehat{\mu}_i)^2}{\widehat{\mathrm{Var}}[Y_i]}$$

where n denotes the number of observations used in the analysis, f_i is the frequency associated with the ith observation (or 1 if no frequency is specified), μ_i is the mean of the mixture, and the denominator is the variance of the ith observation in the mixture. Note that the mean and variance in this expression are not those of the component distributions, but the mean and variance of the mixture:

$$\mu_i = \mathrm{E}[Y_i] = \sum_{j=1}^{k} \pi_{ij}\mu_{ij}$$

$$\mathrm{Var}[Y_i] = -\mu_i^2 + \sum_{j=1}^{k} \pi_{ij}\left(\sigma_{ij}^2 + \mu_{ij}^2\right)$$

where μ_{ij} and σ_{ij}^2 are the mean and variance, respectively, for observation i in the jth component distribution and π_{ij} is the mixing probability for observation i in component j.

The unscaled Pearson statistic is computed with the same expression as the Pearson statistic with n, f_i, and μ_i as previously defined, but the scale parameter ϕ is set to 1 in the $\widehat{\mathrm{Var}}[Y_i]$ expression.

The number of effective components and the number of effective parameters are determined by examining the converged solution for the parameters that are associated with model effects and the mixing probabilities. For example, if a component has an estimated mixing probability of zero, the values of its parameter estimates are immaterial. You might argue that all parameters should be counted towards the penalty in the information criteria. But a component with zero mixing probability in a k-component model effectively reduces the model to a $(k-1)$-component model. A situation of an overfit model, for which a parameter penalty needs to be taken when calculating the information criteria, is a different situation; here the mixing probability might be small, possibly close to zero.

Parameter Estimates

The parameter estimates, their estimated (asymptotic) standard errors, and p-values for the hypothesis that the parameter is zero are presented in the "Parameter Estimates" table. A separate table is produced for each MODEL statement, and the components that are associated with a MODEL statement are identified with an overall component count variable that counts across MODEL statements. If you assign a label to a model with the LABEL= option in the MODEL statement, the label appears in the title of the "Parameter Estimates" table. Otherwise, the internal label generated by the FMM procedure is used.

If the MODEL statement does not contain effects and the link function is not the identity, the inversely linked estimate is also displayed in the table. For many distributions, the inverse linked estimate is the estimated mean on the data scale. For example, in a binomial or binary model, it represents the estimated probability of an event. For some distributions (for example, the Weibull distribution), the inverse linked estimate is not the component distribution mean.

If you request confidence intervals with the CL or ALPHA= option in the MODEL statement, confidence limits are produced for the estimate on the linear scale. If the inverse linked estimate is displayed, confidence intervals for that estimate are also produced by inversely linking the confidence bounds on the linear scale.

Mixing Probabilities

If you fit a model with more than one component, the table of mixing probabilities is produced. If there are no effects in the PROBMODEL statement or if there is no PROBMODEL statement, the parameters are reported on the linear scale and as mixing probabilities. If model effects are present, only the linear parameters (on the scale of the logit, generalized logit, probit, and so on) are displayed.

Default Output for Bayesian Estimation

Bayes Information

This table provides basic information about the sampling algorithm. The FMM procedure uses either a conjugate sampler or a Metropolis-Hastings sampling algorithm based on Gamerman (1997). The table reveals, for example, how many model parameters are sampled, how many parameters associated with mixing probabilities are sampled, and how many threads are used to perform multithreaded analysis.

Prior Distributions

The "Prior Distributions" table lists for each sampled parameter the prior distribution and its parameters. The mean and variance (if they exist) for those values of the parameters are also displayed, along with the initial value for the parameter in the Markov chain. The Component column in this table identifies the mixture component to which a particular parameter belongs. You can control how the FMM procedure determines initial values with the INITIAL= option in the BAYES statement.

Bayesian Fit Statistics

The "Bayesian Fit Statistics" table shows three measures based on the posterior sample. The "Average -2 Log Likelihood" is derived from the average mixture log-likelihood for the data, where the average is taken over the posterior sample. The deviance information criterion (DIC) is a Bayesian measure of model fit and the effective number of parameters (p_D) is a penalization term used in the computation of the DIC. For more information about the DIC and p_D, see the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Posterior Summaries

The arithmetic mean, standard deviation, and percentiles of the posterior distribution of the parameter estimates are displayed in the "Posterior Summaries" table. By default, the FMM procedure computes the 25th, 50th (median), and 75th percentiles of the sampling distribution. You can modify the percentiles through suboptions of the STATISTICS option in the BAYES statement. If a parameter corresponds to a singularity in the design and was removed from sampling for that purpose, it is also displayed in the table of posterior summaries (and in other tables that relate to output from the BAYES statement). The posterior sample size for such a parameter is shown as N = 0.

Posterior Intervals

The table of "Posterior Intervals" displays equal-tail intervals and intervals of highest posterior density for each parameter. By default, intervals are computed for an α-level of 0.05, which corresponds to 95% intervals. You can modify this confidence level by providing one or more α values in the ALPHA= suboption of the STATISTICS option in the BAYES statement. The computation of these intervals is detailed in section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Posterior Autocorrelations

Autocorrelations for the posterior estimates are computed by default for autocorrelation lags 1, 5, 10, and 50, provided that a sufficient number of posterior samples is available. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for the computation of posterior autocorrelations and their utility in diagnosing convergence of Markov chains. You can modify the list of lags for which posterior autocorrelations are calculated with the AUTOCORR suboption of the DIAGNOSTICS= option in the BAYES statement.

Effective Sample Sizes

The "Effective Sample Sizes" table displays the effective sample size (Kass et al. 1998) for each of the parameters in the model. The effective sample size (ESS) for an MCMC sample is another assessment of the mixing of the Markov chain. For more information about the ESS, see the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

ODS Table Names

Each table created by PROC FMM has a name associated with it, and you must use this name to reference the table when you use ODS statements. These names are listed in Table 46.9.

Table 46.9 ODS Tables Produced by PROC FMM

Table Name	Description	Required Statement / Option
AutoCorr	Autocorrelation among posterior estimates	BAYES
BayesInfo	Basic information about Bayesian estimation	BAYES
ClassLevels	Level information from the CLASS statement	CLASS

Table 46.9 *continued*

Table Name	Description	Required Statement / Option
CompDescription	Component description in models with varying number of components	KMAX= in MODEL with ML estimation
CompEvaluation	Comparison of mixture models with varying number of components	KMAX= in MODEL with ML estimation
CompInfo	Component information	COMPONENTINFO option in PROC FMM statement
ConvergenceStatus	Status of optimization at conclusion of optimization	Default output
Constraints	Linear equality and inequality constraints	RESTRICT statement or EQUATE=EFFECTS option in MODEL statement
Corr	Asymptotic correlation matrix of parameter estimates (ML) or empirical correlation matrix of the Bayesian posterior estimates	CORR option in PROC FMM statement
Cov	Asymptotic covariance matrix of parameter estimates (ML) or empirical covariance matrix of the Bayesian posterior estimates	COV option in PROC FMM statement
CovI	Inverse of the covariance matrix of the parameter estimates	COVI option in PROC FMM statement
ESS	Effective sample sizes	BAYES
FitStatistics	Fit statistics	Default output
Geweke	Geweke diagnostics (Geweke 1992) for Markov chains	DIAG=GEWEKE option in BAYES statement
Heidelberger	Heidelberger and Welch diagnostics (Heidelberger and Welch 1981, 1983) for Markov chains	DIAG=HEIDELBERGER option in BAYES statement
Hessian	Hessian matrix from the maximum likelihood optimization, evaluated at the converged estimates	HESSIAN
IterHistory	Optimizer iteration history	Default output for maximum likelihood estimation; included in Bayesian estimation when BAYES statement includes the INITIAL=MLE option and PROC FMM statement includes the FITDETAILS option

Table 46.9 *continued*

Table Name	Description	Required Statement / Option
MCSE	Monte Carlo standard errors	DIAG=MCERROR in BAYES statement
MixingProbs	Solutions for the parameter estimates associated with effects in PROBMODEL statements	Default output for ML estimation if number of components is greater than 1
ModelInfo	Model information	Default output
NObs	Number of observations read and used, number of trials and events	Default output
OptInfo	Optimization information	Default output for ML estimation
ParameterEstimates	Solutions for the parameter estimates associated with effects in MODEL statements	Default output for ML estimation
ParameterMap	Mapping of parameter names to OUTPOST= data set or RandomStartConvergence table	OUTPOST= option in BAYES statement or RANDSTART statement
PostIntervals	Equal-tail and highest posterior density intervals for posterior estimates	BAYES
PostSummaries	Summary statistics for posterior estimates	BAYES
PriorInfo	Prior distributions and initial value of Markov chain	BAYES
Raftery	Raftery and Lewis diagnostics (Raftery and Lewis 1992, 1995) for Markov chains	DIAG=RAFTERY option in BAYES statement
RandomStartConvergence	Starting values and convergence indicator for each random starting set	RANDSTART statement
RandomStartInfo	Information about random starting values	RANDSTART statement
ResponseProfile	Response categories and category modeled	Default output in models with binary response

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC FMM generates are listed in Table 46.10, along with the required statements and options.

Table 46.10 Graphs Produced by PROC FMM

ODS Graph Name	Plot Description	Option
TADPanel	Panel of diagnostic graphics to assess convergence of Markov chains	BAYES
DensityPlot	Histogram and density with component distributions	Default plot for homogeneous mixtures
CriterionPanel	Panel of plots showing progression of model fit criteria for mixtures with different numbers of components	KMIN= and KMAX= options in MODEL statement

Examples: FMM Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 46.1: Modeling Mixing Probabilities: All Mice Are Equal, but Some Mice Are More Equal Than Others

(View the complete code for this example (fmmex01.sas) in the example repository.)

This example demonstrates how you can model the means and mixture proportions separately in a binomial cluster model. It also compares the binomial cluster model to the beta-binomial model.

In a typical teratological experiment, the offspring of animals that were exposed to a toxin during pregnancy are studied for malformation. If you count the number of malformed offspring in a litter of size n, then this count is typically not binomially distributed. The responses of the offspring from the same litter are not independent; hence their sum does not constitute a binomial random variable. Relative to a binomial model, data from teratological experiments exhibit overdispersion because ignoring positive correlation among the responses tends to overstate the precision of the parameter estimates. Overdispersion mechanisms are briefly discussed in the section "Overdispersion" on page 3093.

In this application, the focus is on mixtures and models that involve a mixing mechanism. The mixing approach (Williams 1975; Haseman and Kupper 1979) supposes that the binomial success probability is a random variable that follows a $\text{beta}(\alpha, \beta)$ distribution:

$$
\begin{aligned}
Y|\mu &\sim \text{Binomial}(n, \mu) \\
\mu &\sim \text{Beta}(\alpha, \beta) \\
Y &\sim \text{Beta-binomial}(n, \mu, \phi) \\
\text{E}[Y] &= n\pi \\
\text{Var}[Y] &= n\pi(1-\pi)\left\{1 + \mu^2(n-1)\right\}
\end{aligned}
$$

If $\mu = 0$, then the beta-binomial distribution reduces to a standard binomial model with success probability π. The parameterization of the beta-binomial distribution used by the FMM procedure is based on Neerchal and Morel (1998), see the section "Log-Likelihood Functions for Response Distributions" on page 3094 for details.

Morel and Nagaraj (1993); Morel and Neerchal (1997); Neerchal and Morel (1998) propose a different model to capture dependency within binomial clusters. Their model is a two-component mixture that gives rise to the same mean and variance function as the beta-binomial model. The genesis is different, however. In the binomial cluster model of Morel and Neerchal, suppose there is a cluster of n Bernoulli outcomes with success probability π. The number of responses in the cluster decomposes into $N \leq n$ outcomes that all respond with either "success" or "failure"; the important aspect is that they all respond identically. The remaining $n - N$ Bernoulli outcomes respond independently, so the sum of successes in this group is a binomial$(n - N, \pi)$ random variable. Denote the probability with which cluster members fall into the group of identical respondents as μ. Then $1 - \mu$ is the probability that a response belongs to the group of independent Bernoulli outcomes.

It is easy to see how this process of dividing the individual Bernoulli outcomes creates clustering. The binomial cluster model can be written as the two-component mixture

$$\Pr(Y = y) = \pi \Pr(U = y) + (1 - \pi) \Pr(V = y)$$

where $U \sim \text{Binomial}(n, \mu^* + \mu)$, $V \sim \text{Binomial}(n, \mu^*)$, and $\mu^* = (1 - \mu)\pi$. This mixture model is somewhat unusual because the mixing probability π appears as a parameter in the component distributions. The two probabilities involved, π and μ, have the following interpretation: π is the unconditional probability of success for any observation, and μ is the probability with which the Bernoulli observations respond identically. The complement of this probability, $1 - \mu$, is the probability with which the Bernoulli outcomes respond independently. If $\mu = 0$, then the two-component mixture reduces to a standard binomial model that has success probability π. Because both π and μ are involved in the success probabilities of the two binomial variables in the mixture, you can affect these binomial means by specifying effects in the PROBMODEL statement (for the π's) or the MODEL statement (for the μ's). You would vary the success probabilities μ_1 and μ_2 by using the MODEL statement in a "straight" two-component binomial mixture,

$$\pi \text{Binomial}(n, \mu_1) + (1 - \pi)\text{Binomial}(n, \mu_2)$$

You can fit the beta-binomial model by specifying DIST=BETABIN and the binomial cluster model by specifying DIST=BINOMCLUSTER in the MODEL statement.

Morel and Neerchal (1997) report data from a completely randomized design that studies the teratogenicity of phenytoin in 81 pregnant mice. The treatment structure of the experiment is an augmented factorial. In addition to an untreated control, mice received 60 mg/kg of phenytoin (PHT), 100 mg/kg of trichloropropene oxide (TCPO), and their combination. The design was augmented with a control group that was treated with water. As in Morel and Neerchal (1997), the two control groups are combined here into a single group.

The following DATA step creates the data for this analysis as displayed in Table 1 of Morel and Neerchal (1997). The second DATA step creates continuous variables x1–x3 to match the parameterization of these authors.

```
data ossi;
   length tx $8;
   input tx$ n @@;
   do i=1 to n;
      input y m @@;
      output;
   end;
   drop i;
   datalines;
Control  18 8 8 9  9  7  9 0  5 3  3 5 8 9 10 5 8 5 8 1 6 0 5
            8 8 9 10  5  5 4  7 9 10 6 6 3  5
Control  17 8 9 7 10 10 10 1  6 6  6 1 9 8  9 6 7 5 5 7 9
            2 5 5  6  2  8 1  8 0  2 7 8 5  7
PHT      19 1 9 4  9  3  7 4  7 0  7 0 4 1  8 1 7 2 7 2 8 1 7
            0 2 3 10  3  7 2  7 0  8 0 8 1 10 1 1
TCPO     16 0 5 7 10  4  4 8 11 6 10 6 9 3  4 2 8 0 6 0 9
            3 6 2  9  7  9 1 10 8  8 6 9
PHT+TCPO 11 2 2 0  7  1  8 7  8 0 10 0 4 0  6 0 7 6 6 1 6 1 7
;

data ossi;
```

```
   set ossi;
   array xx{3} x1-x3;
   do i=1 to 3; xx{i}=0; end;
   pht  = 0;
   tcpo = 0;
   if (tx='TCPO') then do;
      xx{1} = 1;
      tcpo  = 100;
   end; else if (tx='PHT') then do;
      xx{2} = 1;
      pht   = 60;
   end; else if (tx='PHT+TCPO') then do;
      pht  = 60;
      tcpo = 100;
      xx{1} = 1; xx{2} = 1; xx{3}=1;
   end;
run;
```

The FMM procedure models the mean parameters μ through the MODEL statement and the mixing proportions π through the PROBMODEL statement. In the binomial cluster model, you can place a regression structure on either set of probabilities, and the regression structure does not need to be the same. In the following statements, the unconditional probability of ossification is modeled as a two-way factorial, whereas the intralitter effect—the propensity to group within a cluster—is assumed to be constant:

```
proc fmm data=ossi;
   class pht tcpo;
   model y/m = / dist=binomcluster;
   probmodel pht tcpo pht*tcpo;
run;
```

The CLASS statement declares the PHT and TCPO variables as classification variables. They affect the analysis through their levels, not through their numeric values. The MODEL statement declares the distribution of the data to follow a binomial cluster model. The FMM procedure then automatically assumes that the model is a two-component mixture. An intercept is included by default. The PROBMODEL statement declares the effect structure for the mixing probabilities. The unconditional probability of ossification of a fetus depends on the main effects and the interaction in the factorial.

The "Model Information" table displays important details about the model fit with the FMM procedure (Output 46.1.1). Although no K= option was specified in the MODEL statement, the FMM procedure recognizes the model as a two-component model. The "Class Level Information" table displays the levels and values of the PHT and TCPO variables. Eighty-one observations are read from the data and are used in the analysis. These observations comprise 287 events and 585 total outcomes.

Output 46.1.1 Model Information in Binomial Cluster Model with Constant Clustering Probability

The FMM Procedure

Model Information	
Data Set	WORK.OSSI
Response Variable (Events)	y
Response Variable (Trials)	m
Type of Model	Binomial Cluster
Distribution	Binomial Cluster
Components	2
Link Function	Logit
Estimation Method	Maximum Likelihood

Class Level Information		
Class	**Levels**	**Values**
pht	2	0 60
tcpo	2	0 100

Number of Observations Read	81
Number of Observations Used	81
Number of Events	287
Number of Trials	585

The "Optimization Information" table in Output 46.1.2 gives details about the maximum likelihood optimization. By default, the FMM procedure uses a quasi-Newton algorithm. The model contains five parameters, four of which are part of the model for the mixing probabilities. The fifth parameter is the intercept in the model for μ.

Output 46.1.2 Optimization in Binomial Cluster Model with Constant Clustering Probability

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	5
Mean Function Parameters	1
Scale Parameters	0
Mixing Prob Parameters	4
Number of Threads	4

Output 46.1.2 *continued*

Iteration History				
Iteration	Evaluations	Objective Function	Change	Max Gradient
0	5	174.9272389		43.78769
1	2	154.1318074	20.79543149	11.23460
2	3	153.2669361	0.86487133	6.888215
3	2	152.8497428	0.41719329	3.541977
4	3	152.6175603	0.23218248	2.783556
5	3	152.5479530	0.06960730	1.146807
6	3	152.5268493	0.02110374	0.034367
7	3	152.5267121	0.00013715	0.011511
8	3	152.5267080	0.00000415	0.000202
9	3	152.5267080	0.00000000	4.001E-6

Convergence criterion (GCONV=1E-8) satisfied.

Fit Statistics	
-2 Log Likelihood	305.1
AIC (Smaller is Better)	315.1
AICC (Smaller is Better)	315.9
BIC (Smaller is Better)	327.0
Pearson Statistic	89.2077
Effective Parameters	5
Effective Components	2

After nine iterations, the iterative optimization converges. The –2 log likelihood at the converged solution is 305.1, and the Pearson statistic is 89.2077. The FMM procedure computes the Pearson statistic as a general goodness-of-fit measure that expresses the closeness of the fitted model to the data.

The estimates of the parameters in the conditional probability μ and in the unconditional probability π are given in Output 46.1.3. The intercept estimate in the model for μ is 0.3356. Because the default link in the binomial cluster model is the logit link, the estimate of the conditional probability is

$$\widehat{\mu} = \frac{1}{1 + \exp\{-0.3356\}} = 0.5831$$

This value is displayed in the "Inverse Linked Estimate" column. There is greater than a 50% chance that the individual fetuses in a litter provide the same response. The clustering tendency is substantial.

Output 46.1.3 Parameter Estimates in Binomial Cluster Model with Constant Clustering Probability

Parameter Estimates for Binomial Cluster Model						
Component	Effect	Estimate	Standard Error	z Value	Pr > |z|	Inverse Linked Estimate
1	Intercept	0.3356	0.1714	1.96	0.0503	0.5831

Output 46.1.3 *continued*

Parameter Estimates for Mixing Probabilities							
Component	Effect	pht	tcpo	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept			-1.2194	0.4690	-2.60	0.0093
1	pht	0		0.9129	0.5608	1.63	0.1036
1	pht	60		0	.	.	.
1	tcpo		0	0.3295	0.5534	0.60	0.5516
1	tcpo		100	0	.	.	.
1	pht*tcpo	0	0	0.6162	0.6678	0.92	0.3561
1	pht*tcpo	0	100	0	.	.	.
1	pht*tcpo	60	0	0	.	.	.
1	pht*tcpo	60	100	0	.	.	.

The "Mixing Probabilities" table displays the estimates of the parameters in the model for π on the logit scale (Output 46.1.3). Table 46.11 constructs the estimates of the unconditional probabilities of ossification.

Table 46.11 Estimates of Ossification Probabilities

PHT	TCPO	$\widehat{\eta}$	$\widehat{\pi}$
0	0	$-1.2194 + 0.9129 + 0.3295 + 0.6162 = 0.6392$	0.6546
60	0	$-1.2194 + 0.3295 = -0.8899$	0.2911
0	100	$-1.2194 + 0.9129 = -0.3065$	0.4240
60	100	-1.2194	0.2280

Morel and Neerchal (1997) considered a model in which the intralitter effects also depend on the treatments. This model is fit with the FMM procedure with the following statements:

```
proc fmm data=ossi;
   class pht tcpo;
   model y/m = pht tcpo pht*tcpo / dist=binomcluster;
   probmodel   pht tcpo pht*tcpo;
run;
```

The –2 log likelihood of this model is much reduced compared to the previous model with constant conditional probability (compare 287.8 in Output 46.1.4 with 305.1 in Output 46.1.2). The likelihood-ratio statistic of 17.3 is significant, $\Pr(\chi^2_3 > 17.3 = 0.0006)$. Varying the conditional probabilities by treatment improved the model fit significantly.

Output 46.1.4 Fit Statistics and Parameter Estimates in Binomial Cluster Model

The FMM Procedure

Fit Statistics	
-2 Log Likelihood	287.8
AIC (Smaller is Better)	303.8
AICC (Smaller is Better)	305.8
BIC (Smaller is Better)	323.0
Pearson Statistic	85.5998
Effective Parameters	8
Effective Components	2

Parameter Estimates for Binomial Cluster Model

Component	Effect	pht	tcpo	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept			1.8213	0.5889	3.09	0.0020
1	pht	0		-1.4962	0.6630	-2.26	0.0240
1	pht	60		0	.	.	.
1	tcpo		0	-3.1828	1.1261	-2.83	0.0047
1	tcpo		100	0	.	.	.
1	pht*tcpo	0	0	3.3736	1.1953	2.82	0.0048
1	pht*tcpo	0	100	0	.	.	.
1	pht*tcpo	60	0	0	.	.	.
1	pht*tcpo	60	100	0	.	.	.

Parameter Estimates for Mixing Probabilities

Component	Effect	pht	tcpo	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept			-0.7394	0.5395	-1.37	0.1705
1	pht	0		0.4351	0.6203	0.70	0.4830
1	pht	60		0	.	.	.
1	tcpo		0	-0.5342	0.5893	-0.91	0.3646
1	tcpo		100	0	.	.	.
1	pht*tcpo	0	0	1.4055	0.7080	1.99	0.0471
1	pht*tcpo	0	100	0	.	.	.
1	pht*tcpo	60	0	0	.	.	.
1	pht*tcpo	60	100	0	.	.	.

Table 46.12 computes the conditional probabilities in the four treatment groups. Recall that the previous model estimated a constant clustering probability of 0.5831.

Table 46.12 Estimates of Clustering Probabilities

PHT	TCPO	$\widehat{\eta}$	$\widehat{\mu}$
0	0	$1.8213 - 1.4962 - 3.1828 + 3.3736 = 0.5159$	0.6262
60	0	$1.8213 - 3.1828 = -1.3615$	0.2040
0	100	$1.8213 - 1.4962 = 0.3251$	0.5806
60	100	1.8213	0.8607

The presence of phenytoin alone reduces the probability of response clustering within the litter. The presence of trichloropropene oxide alone does not have a strong effect on the clustering. The simultaneous presence of both agents substantially increases the probability of clustering.

The following statements fit the binomial cluster model in the parameterization of Morel and Neerchal (1997).

```
proc fmm data=ossi;
   model y/m = x1-x3 / dist=binomcluster;
   probmodel   x1-x3;
run;
```

The model fit is the same as in the previous model (compare the "Fit Statistics" tables in Output 46.1.5 and Output 46.1.4). The parameter estimates change due to the reparameterization of the treatment effects and match the results in Table III of Morel and Neerchal (1997).

Output 46.1.5 Fit Statistics and Estimates (Morel and Neerchal Parameterization)

The FMM Procedure

Fit Statistics	
-2 Log Likelihood	287.8
AIC (Smaller is Better)	303.8
AICC (Smaller is Better)	305.8
BIC (Smaller is Better)	323.0
Pearson Statistic	85.5999
Effective Parameters	8
Effective Components	2

Parameter Estimates for Binomial Cluster Model

Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	0.5159	0.2603	1.98	0.0475
1	x1	-0.1908	0.4006	-0.48	0.6339
1	x2	-1.8774	0.9946	-1.89	0.0591
1	x3	3.3736	1.1953	2.82	0.0048

Parameter Estimates for Mixing Probabilities

Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	0.5669	0.2455	2.31	0.0209
1	x1	-0.8712	0.3924	-2.22	0.0264
1	x2	-1.8405	0.3413	-5.39	<.0001
1	x3	1.4055	0.7080	1.99	0.0471

The following sets of statements fit the binomial and beta-binomial models, respectively, as single-component mixtures in the parameterization akin to the first binomial cluster model. Note that the model effects that affect the underlying Bernoulli success probabilities are specified in the MODEL statement, in contrast to the binomial cluster model.

```
proc fmm data=ossi;
   model y/m = x1-x3 / dist=binomial;
run;
```

```
proc fmm data=ossi;
   model y/m = x1-x3 / dist=betabinomial;
run;
```

The Pearson statistic for the beta-binomial model (Output 46.1.6) indicates a much better fit compared to the single-component binomial model (Output 46.1.7). This is not surprising because these data are obviously overdispersed relative to a binomial model because the Bernoulli outcomes are not independent. The difference between the binomial cluster and the beta-binomial model lies in the mechanism by which the correlations are induced:

- a mixing mechanism in the beta-binomial model that leads to a common shared random effect among all offspring in a cluster
- a mixture specification in the binomial cluster model that divides the offspring in a litter into identical and independent responders

Output 46.1.6 Fit Statistics in Binomial Model

The FMM Procedure

Fit Statistics	
-2 Log Likelihood	401.8
AIC (Smaller is Better)	409.8
AICC (Smaller is Better)	410.3
BIC (Smaller is Better)	419.4
Pearson Statistic	252.1

Output 46.1.7 Fit Statistics in Beta-Binomial Model

The FMM Procedure

Fit Statistics	
-2 Log Likelihood	306.6
AIC (Smaller is Better)	316.6
AICC (Smaller is Better)	317.4
BIC (Smaller is Better)	328.5
Pearson Statistic	87.5379

Example 46.2: The Usefulness of Custom Starting Values: When Do Cows Eat?

(View the complete code for this example (fmmex02.sas) in the example repository.)

This example with a mixture of normal and Weibull distributions illustrates the benefits of specifying starting values for some of the components.

The data for this example were generously provided by Dr. Luciano A. Gonzalez of the Lethbridge Research Centre of Agriculture and Agri-Food Canada and his collaborator, Dr. Bert Tolkamp, from the Scottish Agricultural College.

The outcome variable of interest is the logarithm of a time interval between consecutive visits by cattle to feeders. The intervals fall into three categories:

- short breaks within meals—such as when an animal stops eating for a moment and resumes shortly thereafter
- somewhat longer breaks when eating is interrupted to go have a drink of water
- long breaks between meals

Modeling such time interval data is important to understand the feeding behavior and biology of the animals and to derive other biological parameters such as the probability of an animal to stop eating after it has consumed a certain amount of a given food. Because there are three distinct biological categories, data of this nature are frequently modeled as three-component mixtures. The point at which the second and third components cross over is used to separate feeding events into meals.

The original data set comprises 141,414 observations of log feeding intervals. For the purpose of presentation in this document, where space is limited, the data have been rounded to precision 0.05 and grouped by frequency. The following DATA step displays the modified data used in this example. A comparison with the raw data and the results obtained in a full analysis of the original data show that the grouping does not alter the presentation or conclusions in a way that matters for the purpose of this example.

```
data cattle;
   input LogInt Count @@;
   datalines;
 0.70   195  1.10   233  1.40   355  1.60   563
 1.80   822  1.95   926  2.10  1018  2.20  1712
 2.30  3190  2.40  2212  2.50  1692  2.55  1558
 2.65  1622  2.70  1637  2.75  1568  2.85  1599
 2.90  1575  2.95  1526  3.00  1537  3.05  1561
 3.10  1555  3.15  1427  3.20  2852  3.25  1396
 3.30  1343  3.35  2473  3.40  1310  3.45  2453
 3.50  1168  3.55  2300  3.60  2174  3.65  2050
 3.70  1926  3.75  1849  3.80  1687  3.85  2416
 3.90  1449  3.95  2095  4.00  1278  4.05  1864
 4.10  1672  4.15  2104  4.20  1443  4.25  1341
 4.30  1685  4.35  1445  4.40  1369  4.45  1284
 4.50  1523  4.55  1367  4.60  1027  4.65  1491
 4.70  1057  4.75  1155  4.80  1095  4.85  1019
```

```
  4.90  1158  4.95  1088  5.00  1075  5.05   912
  5.10  1073  5.15   803  5.20   924  5.25   916
  5.30   784  5.35   751  5.40   766  5.45   833
  5.50   748  5.55   725  5.60   674  5.65   690
  5.70   659  5.75   695  5.80   529  5.85   639
  5.90   580  5.95   557  6.00   524  6.05   473
  6.10   538  6.15   444  6.20   456  6.25   453
  6.30   374  6.35   406  6.40   409  6.45   371
  6.50   320  6.55   334  6.60   353  6.65   305
  6.70   302  6.75   301  6.80   263  6.85   218
  6.90   255  6.95   240  7.00   219  7.05   202
  7.10   192  7.15   180  7.20   162  7.25   126
  7.30   148  7.35   173  7.40   142  7.45   163
  7.50   152  7.55   149  7.60   139  7.65   161
  7.70   174  7.75   179  7.80   188  7.85   239
  7.90   225  7.95   213  8.00   235  8.05   256
  8.10   272  8.15   290  8.20   320  8.25   355
  8.30   307  8.35   311  8.40   317  8.45   335
  8.50   369  8.55   365  8.60   365  8.65   396
  8.70   419  8.75   467  8.80   468  8.85   515
  8.90   558  8.95   623  9.00   712  9.05   716
  9.10   829  9.15   803  9.20   834  9.25   856
  9.30   838  9.35   842  9.40   826  9.45   834
  9.50   798  9.55   801  9.60   780  9.65   849
  9.70   779  9.75   737  9.80   683  9.85   686
  9.90   626  9.95   582 10.00   522 10.05   450
 10.10   443 10.15   375 10.20   342 10.25   285
 10.30   254 10.35   231 10.40   195 10.45   186
 10.50   143 10.55   100 10.60    73 10.65    49
 10.70    28 10.75    36 10.80    16 10.85     9
 10.90     5 10.95     6 11.00     4 11.05     1
 11.15     1 11.25     4 11.30     2 11.35     5
 11.40     4 11.45     3 11.50     1
 ;
```

If you scan the columns for the Count variable in the DATA step, the prevalence of values between 2 and 5 units of LogInt is apparent, as is a long right tail. To explore these data graphically, the following statements produce a histogram of the data and a kernel density estimate of the density of the LogInt variable.

```
ods graphics on;
proc kde data=cattle;
   univar LogInt / bwm=4;
   freq count;
run;
```

Output 46.2.1 Histogram and Kernel Density for LogInt

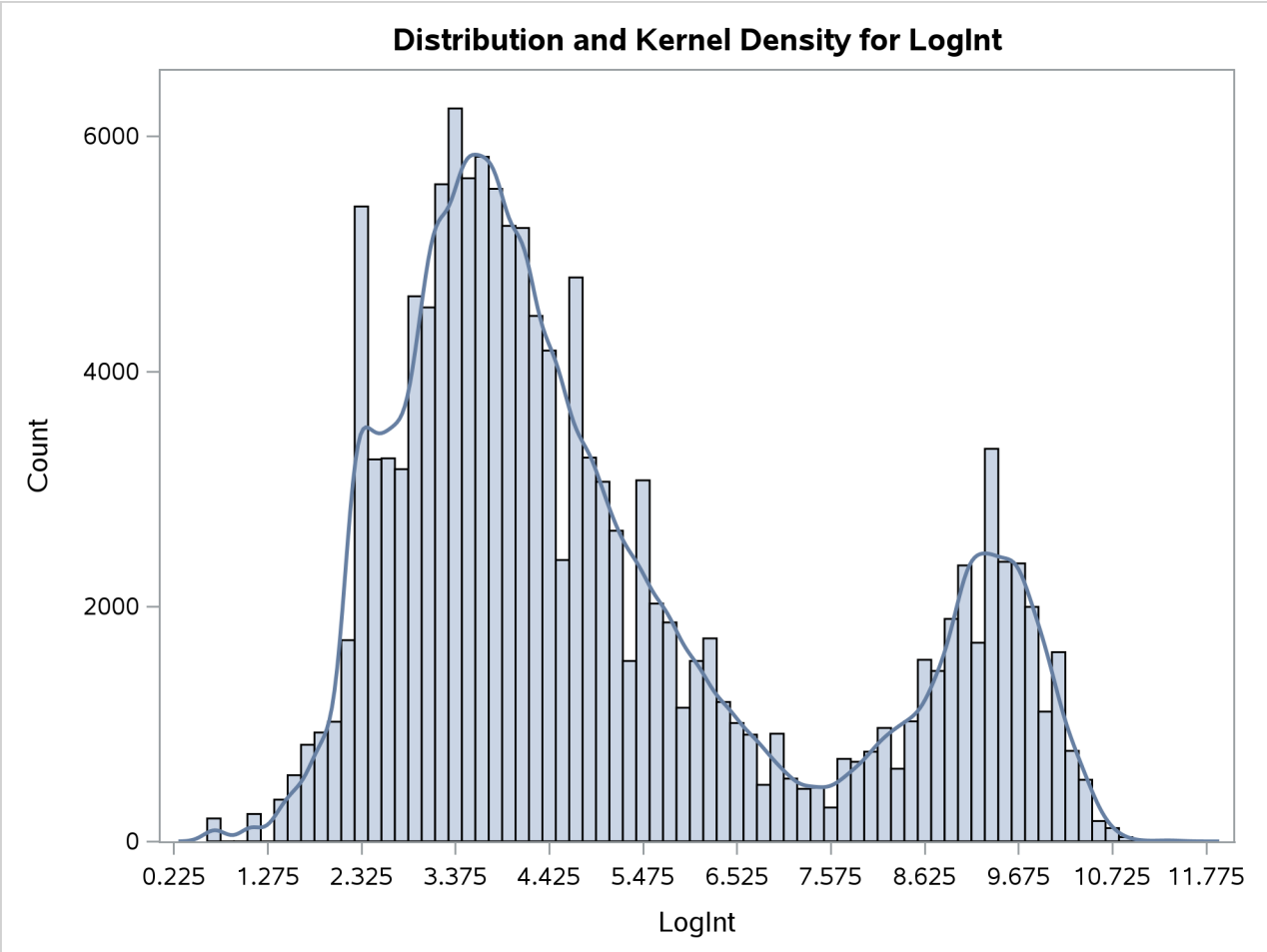

Two modes are clearly visible in Output 46.2.1. Given the biological background, one would expect that three components contribute to the mixture. The histogram would suggest either a two-component mixture with modes near 4 and 9, or a three-component mixture with modes near 3, 5, and 9.

Following Dr. Gonzalez' suggestion, the process is modeled as a three-component mixture of two normal distributions and a Weibull distribution. The Weibull distribution is chosen because it can have long left and right tails and it is popular in modeling data that relate to time intervals.

```
proc fmm data=cattle gconv=0;
   model LogInt = / dist=normal k=2 parms(3 1, 5 1);
   model        + / dist=weibull;
   freq count;
run;
```

The GCONV= convergence criterion is turned off in this PROC FMM run to avoid the early stoppage of the iterations when the relative gradient changes little between iterations. Turning the criterion off usually ensures that convergence is achieved with a small absolute gradient of the objective function. The PARMS option in the first MODEL statement provides starting values for the means and variances for the parameters of the normal distributions. The means for the two components are started at $\mu = 3$ and $\mu = 5$, respectively. Specifying starting values is generally not necessary. However, the choice of starting values can play an

important role in modeling finite mixture models; the importance of the choice of starting values in this example is discussed further below.

The "Model Information" table shows that the model is a three-component mixture and that the FMM procedure considers the estimation of a density to be the purpose of modeling. The procedure draws this conclusion from the absence of effects in the MODEL statements. There are 187 observations in the data set, but these actually represent 141,414 measurements (Output 46.2.2).

Output 46.2.2 Model Information and Number of Observations

The FMM Procedure

Model Information	
Data Set	WORK.CATTLE
Response Variable	LogInt
Frequency Variable	Count
Type of Model	Density Estimation
Components	3
Estimation Method	Maximum Likelihood

Number of Observations Read	187
Number of Observations Used	187
Sum of Frequencies Read	141414
Sum of Frequencies Used	141414

There are eight parameters in the optimization: the means and variances of the two normal distributions, the μ and ϕ parameter of the Weibull distribution, and the two mixing probabilities (Output 46.2.3). At the converged solution, the –2 log likelihood is 563,153 and all parameters and components are effective—that is, the model is not overspecified in the sense that components have collapsed during the model fitting. The Pearson statistic is close to the number of observations in the data set, indicating a good fit.

Output 46.2.3 Optimization Information and Fit Statistics

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	8
Mean Function Parameters	3
Scale Parameters	3
Mixing Prob Parameters	2
Lower Boundaries	3
Upper Boundaries	0
Number of Threads	4

Fit Statistics	
-2 Log Likelihood	563153
AIC (Smaller is Better)	563169
AICC (Smaller is Better)	563169
BIC (Smaller is Better)	563248
Pearson Statistic	141458
Effective Parameters	8
Effective Components	3

Output 46.2.4 displays the parameter estimates for the three models and for the mixing probabilities. The order in which the "Parameter Estimates" tables appear in the output corresponds to the order in which the MODEL statements were specified.

Output 46.2.4 Optimization Information and Fit Statistics

Parameter Estimates for Normal Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	3.3415	0.01260	265.16	<.0001
2	Intercept	4.8940	0.05447	89.84	<.0001
1	Variance	0.6718	0.01287		
2	Variance	1.4497	0.05247		

Parameter Estimates for Weibull Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|	Inverse Linked Estimate
3	Intercept	2.2531	0.000506	4452.11	<.0001	9.5174
3	Scale	0.06848	0.000427			

Parameter Estimates for Mixing Probabilities

Component	Mixing Probability	Linked Scale GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.4545	0.8106	0.03409	23.78	<.0001
2	0.3435	0.5305	0.04640	11.43	<.0001
3	0.2021	0			

The estimated means of the two normal components are 3.3415 and 4.8940, respectively. Note that the means are displayed here as Intercept. The inverse linked estimate is not produced because the default link for the normal distribution is the identity link; hence the Estimate column represents the means of the component distributions. The parameter estimates in the Weibull model are $\widehat{\beta}_0 = 2.2531$, $\widehat{\phi} = 0.06848$, and $\widehat{\mu} = \exp\{\widehat{\beta}_0\} = 9.5174$. In the Weibull distribution, the μ parameter does not estimate the mean of the distribution, the maximum likelihood estimate of the distribution's mean is $\widehat{\mu}\Gamma(\widehat{\phi} + 1) = 9.1828$.

The estimated mixing probabilities are $\widehat{\pi}_1 = 0.4545$, $\widehat{\pi}_2 = 0.3435$, and $\widehat{\pi_3} = 0.2021$. In other words, the estimated distribution of log feeding intervals is a 45:35:20 mixture of an $\mathrm{N}(3.3415, 0.6718)$, a $\mathrm{N}(4.8940, 1.4497)$, and a $\mathrm{Weibull}(9.5174, 0.06848)$ distribution.

You can obtain a graphical display of the observed and estimated distribution of these data by enabling ODS Graphics. The PLOTS option in the PROC FMM statement modifies the default density plot by adding the densities of the mixture components:

```
ods select DensityPlot;
proc fmm data=cattle gconv=0;
   model LogInt = / dist=normal k=2 parms(3 1, 5 1);
   model        + / dist=weibull;
   freq count;
run;
```

Output 46.2.5 Observed and Estimated Densities in the Three-Component Model

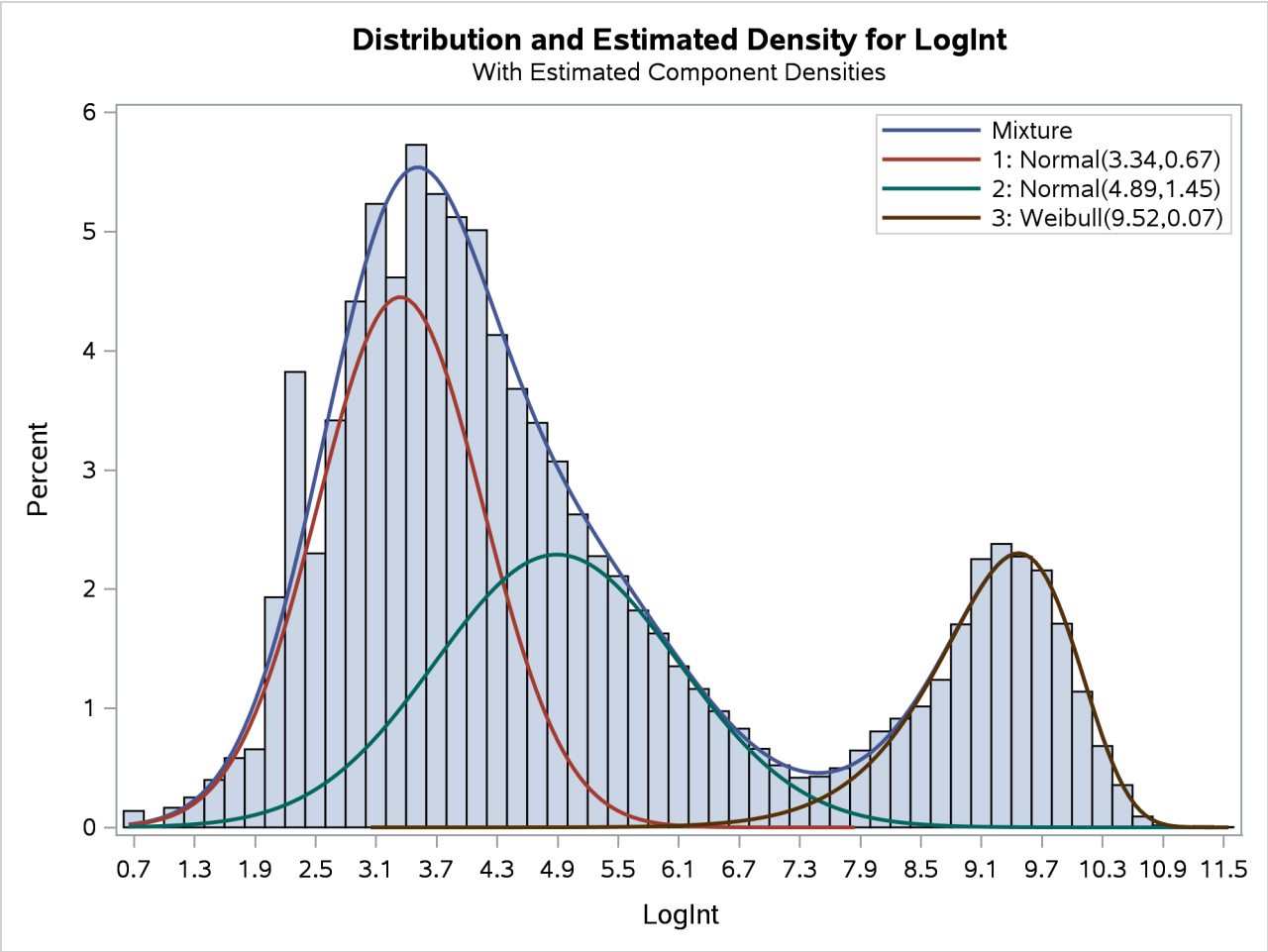

The estimated mixture density matches the histogram of the observed data closely (Output 46.2.5). The component densities are displayed in such a way that, at each point in the support of the LogInt variable, their sum combines to the overall mixture density. The three components in the mixtures are well separated.

The excellent quality of the fit is even more evident when the distributions are displayed cumulatively by adding the CUMULATIVE option in the DENSITY option (Output 46.2.6):

```
ods select DensityPlot;
proc fmm data=cattle plot=density(cumulative) gconv=0;
   model LogInt = / dist=normal k=2 parms(3 1, 5 1);
   model        + / dist=weibull;
   freq count;
run;
```

The component cumulative distribution functions are again scaled so that their sum produces the overall mixture cumulative distribution function. Because of this scaling, the percentage reached at the maximum value of LogInt corresponds to the mixing probabilities in Output 46.2.4.

Output 46.2.6 Observed and Estimated Cumulative Densities in the Three-Component Model

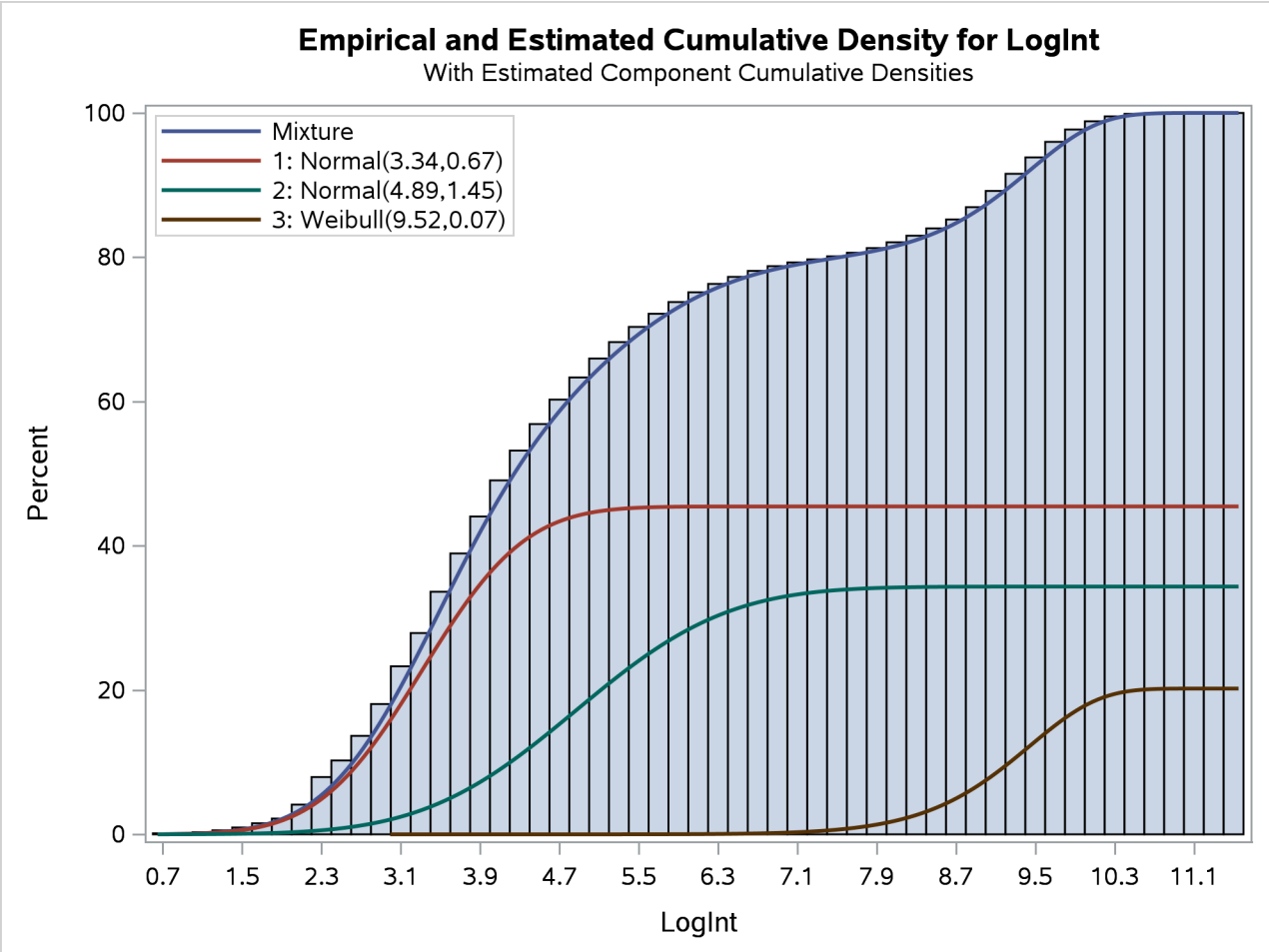

The importance of starting values for the parameter estimates was mentioned previously. Suppose that different starting values are selected for the three components (for example, the default starting values).

```
proc fmm data=cattle gconv=0;
   model LogInt = / dist=normal k=2;
   model        + / dist=weibull;
   freq count;
run;
ods graphics off;
```

The fit statistics and parameter estimates from this run are displayed in Output 46.2.7, and the density plot is shown in Output 46.2.8.

Output 46.2.7 Fit Statistics and Parameter Estimates

The FMM Procedure

Fit Statistics	
-2 Log Likelihood	564431
AIC (Smaller is Better)	564447
AICC (Smaller is Better)	564447
BIC (Smaller is Better)	564526
Pearson Statistic	141228
Effective Parameters	8
Effective Components	3

Parameter Estimates for Normal Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	4.9106	0.02604	188.56	<.0001
2	Intercept	9.2883	0.005031	1846.28	<.0001
1	Variance	1.7410	0.02753		
2	Variance	0.4158	0.005086		

Parameter Estimates for Weibull Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|	Inverse Linked Estimate
3	Intercept	1.2908	0.002790	462.71	<.0001	3.6358
3	Scale	0.2093	0.001311			

Parameter Estimates for Mixing Probabilities

Component	Mixing Probability	Linked Scale GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.3745	-0.1505	0.03678	-4.09	<.0001
2	0.1902	-0.8280	0.01922	-43.08	<.0001
3	0.4353	0			

All components are active; no collapsing of components occurred. However, a closer look at the "Parameter Estimates" tables in Output 46.2.7 shows an important difference from the tables in Output 46.2.4. The means of the two normal distributions are now 4.9106 and 9.2883. Previously, the means were 3.3415 and 4.8940. The "position" of the Weibull distribution has moved from right to left, and the third component is now modeled by a symmetric normal distribution (Output 46.2.8). The mixture probabilities have also changed—in particular, for the first and third component.

Output 46.2.8 Three-Component Model with Default Starting Values

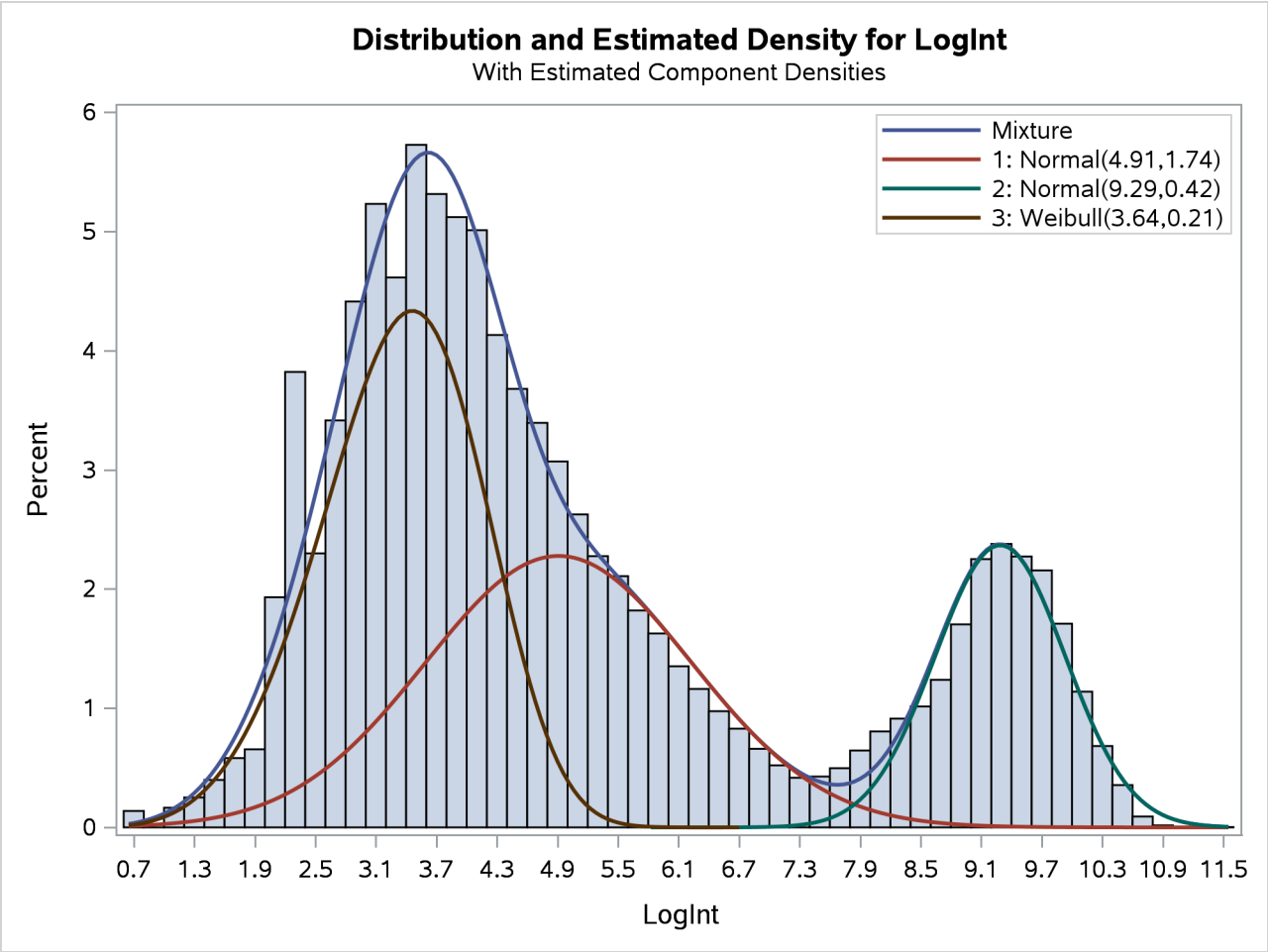

Such switching is not uncommon in mixture modeling. As judged by the information criteria, the model in which the Weibull distribution is the component with the smallest mean does not fit the data as well as the first model in which the specification of the starting values guided the optimization towards placing the normal distributions first. The converged solution found in the last FMM run represents a local minimum of the log-likelihood surface. There are other local minima—for example, when components are removed from the model, which is tantamount to estimating the associated mixture probabilities as zero.

Example 46.3: Enforcing Homogeneity Constraints: Count and Dispersion—It Is All Over!

(View the complete code for this example (fmmex03.sas) in the example repository.)

The following example demonstrates how you can use either the EQUATE= option in the MODEL statement or the RESTRICT statement to impose homogeneity constraints on chosen model effects.

The data for this example were presented by Margolin, Kaplan, and Zeiger (1981) and analyzed by various authors applying a number of techniques. The following DATA step shows the number of revertant salmonella colonies (variable num) at six levels of quinoline dosing (variable dose). There are three replicate plates at each dose of quinoline.

```
data assay;
   label dose = 'Dose of quinoline (microg/plate)'
         num  = 'Observed number of colonies';
   input dose @;
   logd = log(dose+10);
   do i=1 to 3; input num@; output; end;
   datalines;
   0  15 21 29
  10  16 18 21
  33  16 26 33
 100  27 41 60
 333  33 38 41
1000  20 27 42
;
```

The basic notion is that the data are overdispersed relative to a Poisson distribution in which the logarithm of the mean count is modeled as a linear regression in dose (in $\mu g/\text{plate}$) and in the derived variable $\log\{\text{dose} + 10\}$ (Lawless 1987). The log of the expected count of revertants is thus

$$\beta_0 + \beta_1 \text{dose} + \beta_2 \log\{\text{dose} + 10\}$$

The following statements fit a standard Poisson regression model to these data:

```
proc fmm data=assay;
   model num = dose logd / dist=Poisson;
run;
```

The Pearson statistic for this model is rather large compared to the number of degrees of freedom ($18-3 = 15$). The ratio $46.2707/15 = 3.08$ indicates an overdispersion problem in the Poisson model (Output 46.3.1).

Output 46.3.1 Result of Fitting Poisson Regression Models

The FMM Procedure

Number of Observations Read	18
Number of Observations Used	18

Output 46.3.1 *continued*

Fit Statistics	
-2 Log Likelihood	136.3
AIC (Smaller is Better)	142.3
AICC (Smaller is Better)	144.0
BIC (Smaller is Better)	144.9
Pearson Statistic	46.2707

Parameter Estimates for Poisson Model				
Effect	Estimate	Standard Error	z Value	Pr > |z|
Intercept	2.1728	0.2184	9.95	<.0001
dose	-0.00101	0.000245	-4.13	<.0001
logd	0.3198	0.05700	5.61	<.0001

Breslow (1984) accounts for overdispersion by including a random effect in the predictor for the log rate and applying a quasi-likelihood technique to estimate the parameters. Wang et al. (1996) examine these data using mixtures of Poisson regression models. They fit several two- and three-component Poisson regression mixtures. Examining the log likelihoods, AIC, and BIC criteria, they eventually settle on a two-component model in which the intercepts vary by category and the regression coefficients are the same. This mixture model can be written as

$$
\begin{aligned}
f(y) =& \pi \frac{1}{y!}\lambda_1^y \exp\{-\lambda_1\} + (1-\pi)\frac{1}{y!}\lambda_2^y \exp\{-\lambda_2\} \\
\lambda_1 =& \exp\{\beta_{01} + \beta_1 \text{dose} + \beta_2 \log[\text{dose} + 10]\} \\
\lambda_2 =& \exp\{\beta_{02} + \beta_1 \text{dose} + \beta_2 \log[\text{dose} + 10]\}
\end{aligned}
$$

This model is fit with the FMM procedure with the following statements:

```
proc fmm data=assay;
   model num = dose logd / dist=Poisson k=2
                           equate=effects(dose logd);
run;
```

The EQUATE= option in the MODEL statement places constraints on the optimization and makes the coefficients for dose and logd homogeneous across components in the model. Output 46.3.2 displays the "Fit Statistics" and parameter estimates in the mixture. The Pearson statistic is drastically reduced compared to the Poisson regression model in Output 46.3.1. With $18 - 5 = 13$ degrees of freedom, the ratio of the Pearson and the degrees of freedom is now $16.1573/13 = 1.2429$. Note that the effective number of parameters was used to compute the degrees of freedom, not the total number of parameters, because of the equality constraints.

Output 46.3.2 Result for Two-Component Poisson Regression Mixture

The FMM Procedure

Fit Statistics	
-2 Log Likelihood	121.8
AIC (Smaller is Better)	131.8
AICC (Smaller is Better)	136.8
BIC (Smaller is Better)	136.3
Pearson Statistic	16.1573
Effective Parameters	5
Effective Components	2

Parameter Estimates for Poisson Model

Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	1.9097	0.2654	7.20	<.0001
1	dose	-0.00126	0.000273	-4.62	<.0001
1	logd	0.3639	0.06602	5.51	<.0001
2	Intercept	2.4770	0.2731	9.07	<.0001
2	dose	-0.00126	0.000273	-4.62	<.0001
2	logd	0.3639	0.06602	5.51	<.0001

Parameter Estimates for Mixing Probabilities

		Linked Scale			
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.8173	1.4984	0.6875	2.18	0.0293
2	0.1827	-1.4984			

You could also have used RESTRICT statements to impose the homogeneity constraints on the model fit, as shown in the following statements:

```
proc fmm data=assay;
   model num = dose logd / dist=Poisson k=2;
   restrict 'common dose' dose 1, dose -1;
   restrict 'common logd' logd 1, logd -1;
run;
```

The first RESTRICT statement equates the coefficients for the dose variable in the two components, and the second RESTRICT statement accomplishes the same for the coefficients of the logd variable. If the right-hand side of a restriction is not specified, PROC FMM defaults to equating the left-hand side of the restriction to zero. The "Linear Constraints" table in Output 46.3.3 shows that both linear equality constraints are active. The parameter estimates match the previous FMM run.

Output 46.3.3 Result for Two-Component Mixture with RESTRICT Statements

The FMM Procedure

Linear Constraints at Solution					
Label	**k = 1**	**k = 2**			**Constraint Active**
common dose	dose	- dose	=	0	Yes
common logd	logd	- logd	=	0	Yes

Parameter Estimates for Poisson Model					
Component	**Effect**	**Estimate**	**Standard Error**	**z Value**	**Pr > |z|**
1	**Intercept**	1.9097	0.2654	7.20	<.0001
1	**dose**	-0.00126	0.000273	-4.62	<.0001
1	**logd**	0.3639	0.06602	5.51	<.0001
2	**Intercept**	2.4770	0.2731	9.07	<.0001
2	**dose**	-0.00126	0.000273	-4.62	<.0001
2	**logd**	0.3639	0.06602	5.51	<.0001

Parameter Estimates for Mixing Probabilities					
			Linked Scale		
Component	**Mixing Probability**	**Logit(Prob)**	**Standard Error**	**z Value**	**Pr > |z|**
1	0.8173	1.4984	0.6875	2.18	0.0293
2	0.1827	-1.4984			

Wang et al. (1996) note that observation 12 with a revertant colony count of 60 is comparably high. The following statements remove the observation from the analysis and fit their selected model:

```
proc fmm data=assay(where=(num ne 60));
   model num = dose logd / dist=Poisson k=2
                           equate=effects(dose logd);
run;
```

Output 46.3.4 Result for Two-Component Model without Outlier

The FMM Procedure

Fit Statistics	
-2 Log Likelihood	111.5
AIC (Smaller is Better)	121.5
AICC (Smaller is Better)	126.9
BIC (Smaller is Better)	125.6
Pearson Statistic	16.5987
Effective Parameters	5
Effective Components	2

Output 46.3.4 *continued*

Parameter Estimates for Poisson Model					
Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	2.2272	0.3022	7.37	<.0001
1	dose	-0.00065	0.000445	-1.46	0.1440
1	logd	0.2432	0.1045	2.33	0.0199
2	Intercept	2.5477	0.3331	7.65	<.0001
2	dose	-0.00065	0.000445	-1.46	0.1440
2	logd	0.2432	0.1045	2.33	0.0199

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.5777	0.3134	1.7261	0.18	0.8559
2	0.4223	-0.3134			

The ratio of Pearson Statistic over degrees of freedom (12) is only slightly worse than in the previous model; the loss of 5% of the observations carries a price (Output 46.3.4). The parameter estimates for the two intercepts are now fairly close. If the intercepts were identical, then the two-component model would collapse to the Poisson regression model:

```
proc fmm data=assay(where=(num ne 60));
   model num = dose logd / dist=Poisson;
run;
```

Output 46.3.5 Result of Fitting Poisson Regression Model without Outlier

The FMM Procedure

Number of Observations Read	17
Number of Observations Used	17

Fit Statistics	
-2 Log Likelihood	114.1
AIC (Smaller is Better)	120.1
AICC (Smaller is Better)	121.9
BIC (Smaller is Better)	122.5
Pearson Statistic	27.8008

Parameter Estimates for Poisson Model				
Effect	Estimate	Standard Error	z Value	Pr > |z|
Intercept	2.3164	0.2244	10.32	<.0001
dose	-0.00072	0.000258	-2.78	0.0055
logd	0.2603	0.05996	4.34	<.0001

Compared to the same model applied to the full data, the Pearson statistic is much reduced (compare 46.2707 in Output 46.3.1 to 27.8008 in Output 46.3.5). The outlier—or *over*count, if you will—induces at least some

of the *over*dispersion.

Example 46.4: Modeling Multinomial Overdispersion: Town and Country

(View the complete code for this example (fmmex04.sas) in the example repository.)

This example illustrates how you can use the multinomial distribution to model a discrete response that has multiple levels, and how you can use the multinomial cluster model to address overdispersion in multinomial models. The data are survey results from random samples of neighborhoods in both rural and urban areas of Montevideo, Minnesota. There are 18 rural neighborhoods and 17 urban neighborhoods in the survey. In each sampled neighborhood, five households were selected to be interviewed about their level of satisfaction with their homes. The families rated their level of satisfaction as "Unsatisfied," "Satisfied," or "Very Satisfied." These data have previously been analyzed in Brier (1980), Koehler and Wilson (1986), Wilson (1989), and Morel and Nagaraj (1993).

The data include a location type and the numbers of households that respond at each satisfaction level:

```
data housing;
   label us    = 'Unsatisfied'
         s     = 'Satisfied'
         vs    = 'Very Satisfied';
   input type $ us s vs @@;
   datalines;
rural 3 2 0  rural 3 2 0  rural 0 5 0  rural 3 2 0  rural 0 5 0
rural 4 1 0  rural 3 2 0  rural 2 3 0  rural 4 0 1  rural 0 4 1
rural 2 3 0  rural 4 1 0  rural 4 1 0  rural 1 2 2  rural 4 1 0
rural 1 3 1  rural 4 1 0  rural 5 0 0
urban 0 4 1  urban 0 5 0  urban 0 3 2  urban 3 2 0  urban 2 3 0
urban 1 3 1  urban 4 1 0  urban 4 0 1  urban 0 3 2  urban 1 2 2
urban 0 5 0  urban 3 2 0  urban 2 3 0  urban 2 2 1  urban 4 0 1
urban 0 4 1  urban 4 1 0
;
```

The following statements fit a single-component multinomial model to these data, including the location type in the mean model for the multinomial. The response variables are the counts for each observation in vector form.

```
proc fmm data=housing;
   class type;
   model us s vs = Type  / dist=multinomial;
   output out=Pred pred;
run;
```

The model includes the only available covariate, Type, as an explanatory variable for the mean of the multinomial distribution. You use the OUTPUT statement and the PRED keyword to direct PROC FMM to include predicted values for each observation in the Pred output data set.

The "Model Information" table in Output 46.4.1 lists the response variables and indicates that this is a single-component multinomial model. The "Fit Statistics" table shows the associated fit statistics for the model.

Output 46.4.1 Model Information and Fit Statistics for the Multinomial Model

The FMM Procedure

Model Information	
Data Set	WORK.HOUSING
Response Variable	us
Response Variable	s
Response Variable	vs
Type of Model	Homogeneous Regression Mixture
Distribution	Multinomial
Components	1
Link Function	Logit
Estimation Method	Maximum Likelihood

Fit Statistics	
-2 Log Likelihood	194.1
AIC (Smaller is Better)	202.1
AICC (Smaller is Better)	203.4
BIC (Smaller is Better)	208.3
Pearson Statistic	107.3

The parameter estimates capture the relationship between the explanatory variable Type and the different response levels, "Unsatisfied," "Satisfied," and "Very Satisfied." To maintain identifiability, the FMM procedure uses two sets of parameters for the three response variables to parameterize this model. Output 46.4.2 shows the resulting parameter estimates.

Output 46.4.2 Parameter Estimates for the Multinomial Model

Parameter Estimates for Multinomial Model

Response	Effect	type	Estimate	Standard Error	z Value	Pr > |z|
1	**Intercept**		0.9163	0.3416	2.68	0.0073
1	**type**	rural	1.3244	0.5813	2.28	0.0227
1	**type**	urban	0	.	.	.
2	**Intercept**		1.2763	0.3265	3.91	<.0001
2	**type**	rural	0.7519	0.5770	1.30	0.1925
2	**type**	urban	0	.	.	.

The Response column indicates the level of the response that is associated with the parameter set. In this model, Response 1 corresponds to the "Unsatisfied" level and Response 2 corresponds to the "Satisfied" level. This corresponds to the order in which you specify the response variables in the MODEL statement. The "Very Satisfied" level does not appear because of identifiability constraints; the corresponding parameter estimates are set to 0, which means that you can treat the "Very Satisfied" level as the reference level. The estimates of the intercept and the rural effect are positive for both of the other levels, indicating that the estimated proportion at the "Very Satisfied" level is smaller than the proportion at the other two levels for both rural and urban locations.

The following statements compute the predicted proportions for the rural and urban locations from the Pred output data set by normalizing the predicted counts for each location:

```
data Pred; set Pred;
   Pred_1 = Pred_1 / (us + s + vs);
   Pred_2 = Pred_2 / (us + s + vs);
   Pred_3 = Pred_3 / (us + s + vs);
run;

proc sort data=Pred nodupkey;
  by type;
proc print data=pred noobs;
  var type pred:;
run;
```

Output 46.4.3 shows the predicted proportions at each response level for each location type. As in Output 46.4.2, the order reflects the order in which you specified the responses in the MODEL statement. Pred_1 corresponds to "Unsatisfied", Pred_2 corresponds to "Satisfied," and Pred_3 corresponds to "Very Satisfied."

Output 46.4.3 Predicted Proportions for Multinomial

type	Pred_1	Pred_2	Pred_3
rural	0.52222	0.42222	0.05556
urban	0.35294	0.50588	0.14118

The estimates of response proportions for the two location types indicate a difference in the distribution of satisfaction levels for the rural and urban populations. In particular, the urban population shows a smaller proportion of respondents in the "Unsatisfied" category (Pred_1).

The number of degrees of freedom is $N \times (R - 1) - p$, where N is the number of observations, R is the number of levels in the multinomial response, and p is the number of parameters in the model. The ratio of the Pearson statistic to the degrees of freedom is then 107.3 / (35 × 2 – 4) = 1.625; this is larger than 1 and so indicates potential overdispersion.

One explanation for overdispersion might be correlation. It is likely that the families in these households meet and talk with one another, which might result in some influence of opinions about housing satisfaction. The observations are not independent in this case; if you model the proportion of each level of satisfaction based only on location type, you will miss this interhousehold influence.

The multinomial cluster model (Morel and Nagaraj 1993) is based on the idea of "clumping"; that is, some proportion μ of the observed population responds in the same way. In the context of the housing satisfaction data, this means that the clumped responders all express the same satisfaction level. The remaining households respond according to a multinomial distribution with parameter $\boldsymbol{\pi}$.

In this model, the clumped responders respond identically with one of the three levels of satisfaction, and that level is not observable. This discrete latent factor makes a mixture of three multinomials an appropriate method. The difference between this mixture and a general mixture of multinomials is the role of the clumping proportion μ and the use of the mixing probabilities in the mean model. In this model, the mixing probabilities $\boldsymbol{\pi}$ also define the multinomial distribution that governs the distribution of the non-clumped responses.

The following statements fit a multinomial cluster model to these data:

```
proc fmm data=housing;
   class type;
   model us s vs = Type / dist=multinomcluster;
   output out=Pred pred;
   probmodel Type;
run;
```

You include Type in the mean for the underlying multinomial distribution by using the PROBMODEL statement and also in the mean for the clumping parameter μ by using the MODEL statement. Output 46.4.4 shows model information and fit statistics for this multinomial cluster model. Because the model specifies three response variables, the resulting mixture model has three components.

Output 46.4.4 Model Information and Fit Statistics for the Multinomial Cluster Model

The FMM Procedure

Model Information	
Data Set	WORK.HOUSING
Response Variable	us
Response Variable	s
Response Variable	vs
Type of Model	Multinomial Cluster
Distribution	Multinomial Cluster
Components	3
Link Function	Logit
Estimation Method	Maximum Likelihood

Fit Statistics	
-2 Log Likelihood	182.9
AIC (Smaller is Better)	194.9
AICC (Smaller is Better)	197.9
BIC (Smaller is Better)	204.3
Pearson Statistic	61.9809
Effective Parameters	6
Effective Components	3

The fit statistics are generally better for the multinomial cluster model. However, Output 46.4.5 indicates that the parameters in the mean model for the clumping probability μ are not significantly different from 0. There does not appear to be strong evidence for a clumping effect as modeled by the multinomial cluster model.

Output 46.4.5 Parameter Estimates for the Multinomial Cluster Model

Parameter Estimates for Multinomial Cluster Model

Component	Effect	type	Estimate	Standard Error	z Value	Pr > |z|
1	**Intercept**		-0.3696	0.4385	-0.84	0.3992
1	**type**	rural	0.09401	0.6312	0.15	0.8816
1	**type**	urban	0	.	.	.

In the multinomial cluster model, the predicted proportions are the same as the mixing probabilities.

Output 46.4.6 shows the parameter estimates for the mixing probabilities.

Output 46.4.6 Mixing Probability Parameter Estimates for the Multinomial Cluster Model

Parameter Estimates for Mixing Probabilities						
Component	Effect	type	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept		0.6383	0.4106	1.55	0.1201
1	type	rural	1.4138	0.6781	2.08	0.0371
1	type	urban	0	.	.	.
2	Intercept		1.1077	0.3741	2.96	0.0031
2	type	rural	0.7900	0.6527	1.21	0.2262
2	type	urban	0	.	.	.

As in the multinomial example, the estimates for the intercept and rural effect are positive for both the "Unsatisfied" and "Satisfied" response levels, indicating that these levels have larger predicted proportions than the "Very Satisfied" level.

You can use the same approach as before to produce the predicted proportions.

Output 46.4.7 shows the predicted proportions at each level of the response for each location type.

Output 46.4.7 Predicted Proportions for the Multinomial Cluster Model

type	Pred_1	Pred_2	Pred_3
rural	0.50367	0.43163	0.06471
urban	0.31977	0.51133	0.16890

By comparing Output 46.4.7 with Output 46.4.3, you can see that the proportion estimates are not markedly different between the models. This is consistent with the lack of significance in the multinomial cluster model's clumping parameters.

References

Aldrich, J. (1997). "R. A. Fisher and the Making of Maximum Likelihood, 1912–1922." *Statistical Science* 12:162–176.

Breslow, N. E. (1984). "Extra-Poisson Variation in Log-Linear Models." *Journal of the Royal Statistical Society, Series C* 33:38–44.

Brier, S. S. (1980). "Analysis of Contingency Tables under Cluster Sampling." *Biometrika* 67:591–596.

Cameron, A. C., and Trivedi, P. K. (1998). *Regression Analysis of Count Data*. Cambridge: Cambridge University Press.

Celeux, G., Forbes, F., Robert, C. P., and Titterington, D. M. (2006). "Deviance Information Criteria for Missing Data Models." *Bayesian Analysis* 1:651–674.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society, Series B* 39:1–38.

Everitt, B. S., and Hand, D. J. (1981). *Finite Mixture Distributions*. London: Chapman & Hall.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). "Beta Regression for Modelling Rates and Proportions." *Journal of Applied Statistics* 31:799–815.

Fisher, R. A. (1921). "On the 'Probable Error' of a Coefficient of Correlation Deduced from a Small Sample." *Metron* 1:3–32.

Frühwirth-Schnatter, S. (2006). *Finite Mixture and Markov Switching Models*. New York: Springer.

Gamerman, D. (1997). "Sampling from the Posterior Distribution in Generalized Linear Models." *Statistics and Computing* 7:57–68.

Geweke, J. (1992). "Evaluating the Accuracy of Sampling-Based Approaches to Calculating Posterior Moments." In *Bayesian Statistics*, vol. 4, edited by J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, 169–193. Oxford: Clarendon Press.

Griffiths, D. A. (1973). "Maximum Likelihood Estimation for the Beta-Binomial Distribution and an Application to the Household Distribution of the Total Number of Cases of a Disease." *Biometrics* 29:637–648.

Haseman, J. K., and Kupper, L. L. (1979). "Analysis of Dichotomous Response Data from Certain Toxicological Experiments." *Biometrics* 35:281–293.

Heidelberger, P., and Welch, P. D. (1981). "A Spectral Method for Confidence Interval Generation and Run Length Control in Simulations." *Communications of the ACM* 24:233–245.

Heidelberger, P., and Welch, P. D. (1983). "Simulation Run Length Control in the Presence of an Initial Transient." *Operations Research* 31:1109–1144.

Joe, H., and Zhu, R. (2005). "Generalized Poisson Distribution: The Property of Mixture of Poisson and Comparison with Negative Binomial Distribution." *Biometrical Journal* 47:219–229.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998). "Markov Chain Monte Carlo in Practice: A Roundtable Discussion." *American Statistician* 52:93–100.

Koehler, K. J., and Wilson, J. R. (1986). "Chi-Square Tests for Comparing Vectors of Proportions for Several Cluster Samples." *Communications in Statistics—Theory and Methods* 15:2977–2990.

Lawless, J. F. (1987). "Negative Binomial and Mixed Poisson Regression." *Canadian Journal of Statistics* 15:209–225.

Margolin, B. H., Kaplan, N. L., and Zeiger, E. (1981). "Statistical Analysis of the Ames Salmonella Microsome Test." *Proceedings of the National Academy of Sciences* 76:3779–3783.

McLachlan, G. J., and Peel, D. (2000). *Finite Mixture Models*. New York: John Wiley & Sons.

Morel, J. G., and Nagaraj, N. K. (1993). "A Finite Mixture Distribution for Modelling Multinomial Extra Variation." *Biometrika* 80:363–371.

Morel, J. G., and Neerchal, N. K. (1997). "Clustered Binary Logistic Regression in Teratology Data Using a Finite Mixture Distribution." *Statistics in Medicine* 16:2843–2853.

Neerchal, N. K., and Morel, J. G. (1998). "Large Cluster Results for Two Parametric Multinomial Extra Variation Models." *Journal of the American Statistical Association* 93:1078–1087.

Pearson, K. (1915). "On Certain Types of Compound Frequency Distributions in Which the Components Can Be Individually Described by Binomial Series." *Biometrika* 11:139–144.

Raftery, A. E. (1996). "Hypothesis Testing and Model Selection." In *Markov Chain Monte Carlo in Practice*, edited by W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, 163–188. London: Chapman & Hall.

Raftery, A. E., and Lewis, S. M. (1992). "One Long Run with Diagnostics: Implementation Strategies for Markov Chain Monte Carlo." *Statistical Science* 7:493–497.

Raftery, A. E., and Lewis, S. M. (1995). "The Number of Iterations, Convergence Diagnostics, and Generic Metropolis Algorithms." In *Markov Chain Monte Carlo in Practice*, edited by W. R. Gilks, D. J. Spiegelhalter, and S. Richardson, 115–130. London: Chapman & Hall.

Richardson, S. (2002). "Discussion of Spiegelhalter et al." *Journal of the Royal Statistical Society, Series B* 64:631.

Roeder, K. (1990). "Density Estimation with Confidence Sets Exemplified by Superclusters and Voids in the Galaxies." *Journal of the American Statistical Association* 85:617–624.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and Van der Linde, A. (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society, Series B* 64:583–616. With discussion.

Titterington, D. M., Smith, A. F. M., and Makov, U. E. (1985). *Statistical Analysis of Finite Mixture Distributions*. New York: John Wiley & Sons.

Viallefont, V., Richardson, S., and Greene, P. J. (2002). "Bayesian Analysis of Poisson Mixtures." *Journal of Nonparametric Statistics* 14:181–202.

Wang, P., Puterman, M. L., Cockburn, I., and Le, N. (1996). "Mixed Poisson Regression Models with Covariate Dependent Rates." *Biometrics* 52:381–400.

Williams, D. A. (1975). "The Analysis of Binary Responses from Toxicological Experiments Involving Reproduction and Teratogenicity." *Biometrics* 31:949–952.

Wilson, J. R. (1989). "Chi-Square Tests for Overdispersion with Multiparameter Estimates." *Journal of the Royal Statistical Society, Series C* 38:441–453.

Chapter 47
The FREQ Procedure

Contents

Overview: FREQ Procedure **3146**
Getting Started: FREQ Procedure **3148**
Frequency Tables and Statistics 3148
Agreement Study 3155
Syntax: FREQ Procedure **3157**
PROC FREQ Statement 3158
BY Statement 3160
EXACT Statement 3161
OUTPUT Statement 3170
TABLES Statement 3182
TEST Statement 3236
WEIGHT Statement 3240
Details: FREQ Procedure **3241**
Inputting Frequency Counts 3241
Grouping with Formats 3242
Missing Values 3243
In-Database Computation 3245
Statistical Computations 3246
Definitions and Notation 3246
Chi-Square Tests and Statistics 3248
Measures of Association 3253
Binomial Proportion 3263
Sensitivity and Specificity 3271
Risks and Risk Differences 3272
Common Risk Difference 3283
Odds Ratio and Relative Risks 3287
Cochran-Armitage Test for Trend 3297
Jonckheere-Terpstra Test 3298
Tests and Measures of Agreement 3300
Cochran-Mantel-Haenszel Statistics 3307
Gail-Simon Test for Qualitative Interactions 3317
Exact Statistics 3318
Computational Resources 3322
Output Data Sets 3323
Displayed Output 3326
ODS Table Names 3336

ODS Graphics 3340
Examples: FREQ Procedure **3341**
Example 47.1: Output Data Set of Frequencies 3341
Example 47.2: Frequency Dot Plots 3344
Example 47.3: Chi-Square Goodness-of-Fit Tests 3347
Example 47.4: Binomial Proportions 3351
Example 47.5: Analysis of a 2x2 Contingency Table 3353
Example 47.6: Output Data Set of Chi-Square Statistics 3356
Example 47.7: Cochran-Mantel-Haenszel Statistics 3358
Example 47.8: Cochran-Armitage Trend Test 3360
Example 47.9: Friedman's Chi-Square Test 3363
Example 47.10: Cochran's Q Test 3365
References **3367**

Overview: FREQ Procedure

The FREQ procedure produces one-way to *n*-way frequency and contingency (crosstabulation) tables. For two-way tables, PROC FREQ computes tests and measures of association. For *n*-way tables, PROC FREQ provides stratified analysis by computing statistics within strata and across strata.

For one-way frequency tables, PROC FREQ provides goodness-of-fit tests for equal proportions or specified null proportions. For one-way tables, PROC FREQ also provides confidence limits and tests for binomial proportions, including tests for noninferiority and equivalence.

For contingency tables, PROC FREQ can compute various statistics to examine the relationships between two classification variables. For some pairs of variables, you might want to examine the existence or strength of any association between the variables. To determine if an association exists, PROC FREQ computes chi-square tests. To estimate the strength of an association, PROC FREQ computes measures of association that tend to be close to zero when there is no association and close to the maximum (or minimum) value when there is perfect association. The statistics for contingency tables include the following:

- chi-square tests and measures
- measures of association
- risks (binomial proportions) and risk differences for 2×2 tables
- odds ratios and relative risks for 2×2 tables
- tests for trend
- tests and measures of agreement
- Cochran-Mantel-Haenszel statistics

PROC FREQ computes asymptotic standard errors, confidence intervals, and tests for measures of association and measures of agreement. Exact p-values and confidence intervals are available for many test statistics and measures. PROC FREQ also performs analyses that adjust for stratification variables by computing statistics within and across strata for n-way tables. These statistics include Cochran-Mantel-Haenszel statistics and measures of agreement.

In choosing measures of association to use in analyzing a two-way table, you should consider the study design (which indicates whether the row and column variables are dependent or independent), the measurement scale of the variables (nominal, ordinal, or interval), the type of association that each measure is designed to detect, and any assumptions required for valid interpretation of a measure. You should exercise care in selecting measures that are appropriate for your data.

Similar comments apply to the choice and interpretation of test statistics. For example, the Mantel-Haenszel chi-square statistic requires an ordinal scale for both variables and is designed to detect a linear association. The Pearson chi-square, on the other hand, is appropriate for all variables and can detect any kind of association, but it is less powerful for detecting a linear association because its power is dispersed over a greater number of degrees of freedom (except for 2×2 tables).

For more information about selecting the appropriate statistical analyses, see Agresti (2007) and Stokes, Davis, and Koch (2012).

Several SAS procedures produce frequency counts; only PROC FREQ computes chi-square tests for one-way to n-way tables and measures of association and agreement for contingency tables. Other procedures to consider for counting include the TABULATE and UNIVARIATE procedures. When you want to produce contingency tables and tests of association for sample survey data, you can use PROC SURVEYFREQ. For more information, see Chapter 15, "Introduction to Survey Procedures." When you want to fit models to categorical data, you can use a procedure such as CATMOD, GENMOD, GLIMMIX, LOGISTIC, PROBIT, or SURVEYLOGISTIC. For more information, see Chapter 9, "Introduction to Categorical Data Analysis Procedures."

PROC FREQ uses the Output Delivery System (ODS), a SAS subsystem that provides capabilities for displaying and controlling the output from SAS procedures. ODS enables you to convert any of the output from PROC FREQ into a SAS data set. See the section "ODS Table Names" on page 3336 for more information.

PROC FREQ uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For information about the statistical graphics that PROC FREQ produces, see the PLOTS= option in the TABLES statement and the section "ODS Graphics" on page 3340.

Getting Started: FREQ Procedure

Frequency Tables and Statistics

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/freqgs1.sas.)

The FREQ procedure provides easy access to statistics for testing for association in a crosstabulation table.

In this example, high school students applied for courses in a summer enrichment program; these courses included journalism, art history, statistics, graphic arts, and computer programming. The students accepted were randomly assigned to classes with and without internships in local companies. Table 47.1 contains counts of the students who enrolled in the summer program by gender and whether they were assigned an internship slot.

Table 47.1 Summer Enrichment Data

		Enrollment		
Gender	**Internship**	**Yes**	**No**	**Total**
boys	yes	35	29	64
boys	no	14	27	41
girls	yes	32	10	42
girls	no	53	23	76

The SAS data set SummerSchool is created by inputting the summer enrichment data as cell count data, or providing the frequency count for each combination of variable values. The following DATA step statements create the SAS data set SummerSchool:

```
data SummerSchool;
   input Gender $ Internship $ Enrollment $ Count @@;
   datalines;
boys  yes yes 35   boys  yes no 29
boys   no yes 14   boys   no no 27
girls yes yes 32   girls yes no 10
girls  no yes 53   girls  no no 23
;
```

The variable Gender takes the values ‘boys’ or ‘girls,’ the variable Internship takes the values ‘yes’ and ‘no,’ and the variable Enrollment takes the values ‘yes’ and ‘no.’ The variable Count contains the number of students that correspond to each combination of data values. The double at sign (@@) indicates that more than one observation is included on a single data line. In this DATA step, two observations are included on each line.

Researchers are interested in whether there is an association between internship status and summer program enrollment. The Pearson chi-square statistic is an appropriate statistic to assess the association in the corresponding 2×2 table. The following PROC FREQ statements specify this analysis.

You specify the table for which you want to compute statistics with the TABLES statement. You specify the statistics you want to compute with options after a slash (/) in the TABLES statement.

```
proc freq data=SummerSchool order=data;
   tables Internship*Enrollment / chisq;
   weight Count;
run;
```

The ORDER= option controls the order in which variable values are displayed in the rows and columns of the table. By default, the values are arranged according to the alphanumeric order of their unformatted values. If you specify ORDER=DATA, the data are displayed in the same order as they occur in the input data set. Here, because 'yes' appears before 'no' in the data, 'yes' appears first in any table. Other options for controlling order include ORDER=FORMATTED, which orders according to the formatted values, and ORDER=FREQ, which orders by descending frequency count.

In the TABLES statement, Internship*Enrollment specifies a table where the rows are internship status and the columns are program enrollment. The CHISQ option requests chi-square statistics for assessing association between these two variables. Because the input data are in cell count form, the WEIGHT statement is required. The WEIGHT statement names the variable Count, which provides the frequency of each combination of data values.

Figure 47.1 presents the crosstabulation of Internship and Enrollment. In each cell, the values printed under the cell count are the table percentage, row percentage, and column percentage, respectively. For example, in the first cell, 63.21 percent of the students offered courses with internships accepted them and 36.79 percent did not.

Figure 47.1 Crosstabulation Table

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Internship by Enrollment		
		Enrollment	
Internship	yes	no	Total
yes	67 30.04 63.21 50.00	39 17.49 36.79 43.82	106 47.53
no	67 30.04 57.26 50.00	50 22.42 42.74 56.18	117 52.47
Total	134 60.09	89 39.91	223 100.00

Figure 47.2 displays the statistics produced by the CHISQ option. The Pearson chi-square statistic is labeled 'Chi-Square' and has a value of 0.8189 with 1 degree of freedom. The associated p-value is 0.3655, which means that there is no significant evidence of an association between internship status and program enrollment. The other chi-square statistics have similar values and are asymptotically equivalent. The other statistics (phi coefficient, contingency coefficient, and Cramér's V) are measures of association derived from the Pearson chi-square. For Fisher's exact test, the two-sided p-value is 0.4122, which also shows no association between internship status and program enrollment.

Figure 47.2 Statistics Produced with the CHISQ Option

Statistic	DF	Value	Prob
Chi-Square	1	0.8189	0.3655
Likelihood Ratio Chi-Square	1	0.8202	0.3651
Continuity Adj. Chi-Square	1	0.5899	0.4425
Mantel-Haenszel Chi-Square	1	0.8153	0.3666
Phi Coefficient		0.0606	
Contingency Coefficient		0.0605	
Cramer's V		0.0606	

Fisher's Exact Test	
Cell (1,1) Frequency (F)	67
Left-sided Pr <= F	0.8513
Right-sided Pr >= F	0.2213
Table Probability (P)	0.0726
Two-sided Pr <= P	0.4122

The analysis, so far, has ignored gender. However, it might be of interest to ask whether program enrollment is associated with internship status after adjusting for gender. You can address this question by doing an analysis of a set of tables (in this case, by analyzing the set consisting of one for boys and one for girls). The Cochran-Mantel-Haenszel (CMH) statistic is appropriate for this situation: it addresses whether rows and columns are associated after controlling for the stratification variable. In this case, you would be stratifying by gender.

The PROC FREQ statements for this analysis are very similar to those for the first analysis, except that there is a third variable, Gender, in the TABLES statement. When you cross more than two variables, the two rightmost variables construct the rows and columns of the table, respectively, and the leftmost variables determine the stratification.

The following PROC FREQ statements also request frequency plots for the crosstabulation tables. PROC FREQ produces these plots by using ODS Graphics to create graphs as part of the procedure output. ODS Graphics must be enabled before producing plots. The PLOTS(ONLY)=FREQPLOT option requests frequency plots. The TWOWAY=CLUSTER *plot-option* specifies a cluster layout for the two-way frequency plots.

```
ods graphics on;
proc freq data=SummerSchool;
   tables Gender*Internship*Enrollment /
          chisq cmh plots(only)=freqplot(twoway=cluster);
   weight Count;
run;
ods graphics off;
```

This execution of PROC FREQ first produces two individual crosstabulation tables of Internship by Enrollment: one for boys and one for girls. Frequency plots and chi-square statistics are produced for each individual table. Figure 47.3, Figure 47.4, and Figure 47.5 show the results for boys. Note that the chi-square statistic for boys is significant at the $\alpha = 0.05$ level of significance. Boys offered a course with an internship are more likely to enroll than boys who are not.

Figure 47.4 displays the frequency plot of Internship by Enrollment for boys. By default, frequency plots are displayed as bar charts. You can use PLOTS= options to request dot plots instead of bar charts, to change the orientation of the bars from vertical to horizontal, and to change the scale from frequencies to percents. You can also use PLOTS= options to specify other two-way layouts (stacked, vertical groups, or horizontal groups) and to change the primary grouping from column levels to row levels.

Figure 47.6, Figure 47.7, and Figure 47.8 display the crosstabulation table, frequency plot, and chi-square statistics for girls. You can see that there is no evidence of association between internship offers and program enrollment for girls.

Figure 47.3 Crosstabulation Table for Boys

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table 1 of Internship by Enrollment Controlling for Gender=boys		
		Enrollment	
Internship	**no**	**yes**	**Total**
no	27 25.71 65.85 48.21	14 13.33 34.15 28.57	41 39.05
yes	29 27.62 45.31 51.79	35 33.33 54.69 71.43	64 60.95
Total	56 53.33	49 46.67	105 100.00

Figure 47.4 Frequency Plot for Boys

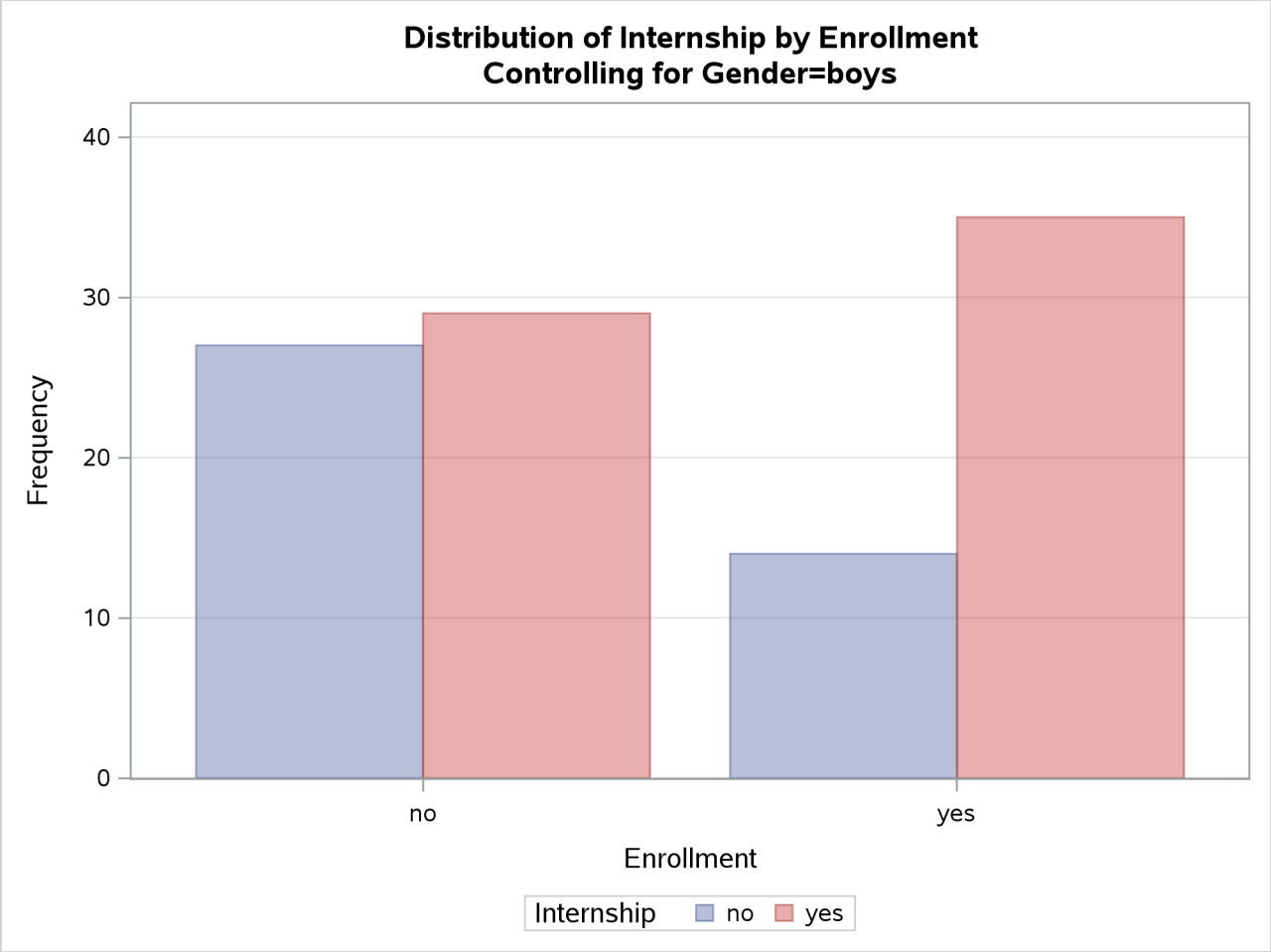

Figure 47.5 Chi-Square Statistics for Boys

Statistic	DF	Value	Prob
Chi-Square	1	4.2366	0.0396
Likelihood Ratio Chi-Square	1	4.2903	0.0383
Continuity Adj. Chi-Square	1	3.4515	0.0632
Mantel-Haenszel Chi-Square	1	4.1963	0.0405
Phi Coefficient		0.2009	
Contingency Coefficient		0.1969	
Cramer's V		0.2009	

Fisher's Exact Test	
Cell (1,1) Frequency (F)	27
Left-sided Pr <= F	0.9885
Right-sided Pr >= F	0.0311
Table Probability (P)	0.0196
Two-sided Pr <= P	0.0467

Figure 47.6 Crosstabulation Table for Girls

Frequency Percent Row Pct Col Pct	Table 2 of Internship by Enrollment Controlling for Gender=girls Enrollment		
Internship	**no**	**yes**	**Total**
no	23 19.49 30.26 69.70	53 44.92 69.74 62.35	76 64.41
yes	10 8.47 23.81 30.30	32 27.12 76.19 37.65	42 35.59
Total	33 27.97	85 72.03	118 100.00

Figure 47.7 Frequency Plot for Girls

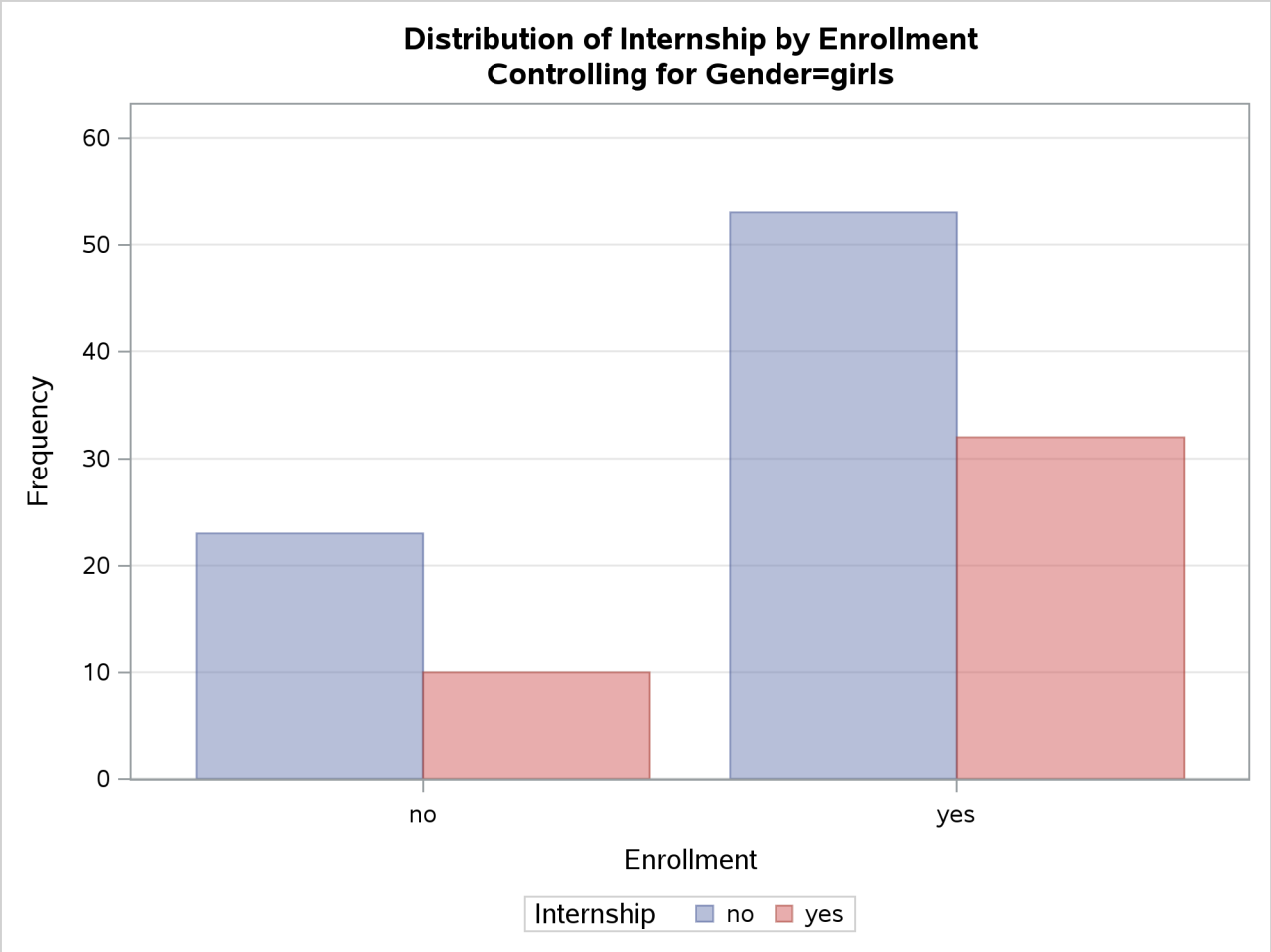

Figure 47.8 Chi-Square Statistics for Girls

Statistic	DF	Value	Prob
Chi-Square	1	0.5593	0.4546
Likelihood Ratio Chi-Square	1	0.5681	0.4510
Continuity Adj. Chi-Square	1	0.2848	0.5936
Mantel-Haenszel Chi-Square	1	0.5545	0.4565
Phi Coefficient		0.0688	
Contingency Coefficient		0.0687	
Cramer's V		0.0688	

Fisher's Exact Test	
Cell (1,1) Frequency (F)	23
Left-sided Pr <= F	0.8317
Right-sided Pr >= F	0.2994
Table Probability (P)	0.1311
Two-sided Pr <= P	0.5245

These individual table results demonstrate the occasional problems with combining information into one table and not accounting for information in other variables such as Gender. Figure 47.9 contains the CMH results. There are three summary (CMH) statistics; which one you use depends on whether your rows and/or columns have an order in $r \times c$ tables. However, in the case of 2×2 tables, ordering does not matter and all three statistics take the same value. The CMH statistic follows the chi-square distribution under the hypothesis of no association, and here, it takes the value 4.0186 with 1 degree of freedom. The associated p-value is 0.0450, which indicates a significant association at the $\alpha = 0.05$ level.

Thus, when you adjust for the effect of gender in these data, there is an association between internship and program enrollment. But, if you ignore gender, no association is found. Note that the CMH option also produces other statistics, including estimates and confidence limits for relative risk and odds ratios for 2×2 tables and the Breslow-Day Test. These results are not displayed here.

Figure 47.9 Test for the Hypothesis of No Association

Cochran-Mantel-Haenszel Statistics (Based on Table Scores)				
Statistic	Alternative Hypothesis	DF	Value	Prob
1	Nonzero Correlation	1	4.0186	0.0450
2	Row Mean Scores Differ	1	4.0186	0.0450
3	General Association	1	4.0186	0.0450

Agreement Study

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/freqgs2.sas.)

Medical researchers are interested in evaluating the efficacy of a new treatment for a skin condition. Dermatologists from participating clinics were trained to conduct the study and to evaluate the condition. After the training, two dermatologists examined patients with the skin condition from a pilot study and rated the same patients. The possible evaluations are terrible, poor, marginal, and clear. Table 47.2 contains the data.

Table 47.2 Skin Condition Data

	Dermatologist 2			
Dermatologist 1	**Terrible**	**Poor**	**Marginal**	**Clear**
Terrible	10	4	1	0
Poor	5	10	12	2
Marginal	2	4	12	5
Clear	0	2	6	13

The following DATA step statements create the SAS data set SkinCondition. The dermatologists' evaluations of the patients are contained in the variables Derm1 and Derm2; the variable Count is the number of patients given a particular pair of ratings.

```
data SkinCondition;
   input Derm1 $ Derm2 $ Count;
   datalines;
terrible terrible 10
terrible     poor 4
terrible marginal 1
terrible    clear 0
poor     terrible 5
poor         poor 10
poor     marginal 12
poor        clear 2
marginal terrible 2
marginal     poor 4
marginal marginal 12
marginal    clear 5
clear    terrible 0
clear        poor 2
clear    marginal 6
clear       clear 13
;
```

The following PROC FREQ statements request an agreement analysis of the skin condition data. In order to evaluate the agreement of the diagnoses (a possible contribution to measurement error in the study), the *kappa coefficient* is computed.

The TABLES statement requests a crosstabulation of the variables Derm1 and Derm2. The AGREE option in the TABLES statement requests the kappa coefficient, together with its standard error and confidence limits. The KAPPA option in the TEST statement requests a test for the null hypothesis that kappa is 0, which

indicates that the agreement is purely by chance. The NOPRINT option in the TABLES statement suppresses the display of the two-way table. The PLOTS= option requests an agreement plot for the two dermatologists. ODS Graphics must be enabled before producing plots.

```
ods graphics on;
proc freq data=SkinCondition order=data;
   tables Derm1*Derm2 /
          agree noprint plots=agreeplot;
   test kappa;
   weight Count;
run;
ods graphics off;
```

Figure 47.10 and Figure 47.11 show the results. The kappa coefficient has the value 0.3449, which indicates some agreement between the dermatologists, and the hypothesis test confirms that you can reject the null hypothesis of no agreement. This conclusion is further supported by the confidence interval of (0.2030, 0.4868), which suggests that the true kappa is greater than 0. The AGREE option also produces Bowker's symmetry test and the weighted kappa coefficient, but that output is not shown here. Figure 47.11 displays the agreement plot for the ratings of the two dermatologists.

Figure 47.10 Agreement Study

The FREQ Procedure

Statistics for Table of Derm1 by Derm2

Kappa Statistics

Statistic	Estimate	Standard Error	95% Confidence Limits	
Simple Kappa	0.3449	0.0724	0.2030	0.4868
Weighted Kappa	0.5082	0.0655	0.3798	0.6366

Test of H0: Kappa = 0

Estimate	H0 Std Err	Z	Pr > Z	Pr > |Z|
0.3449	0.0612	5.6366	<.0001	<.0001

Figure 47.11 Agreement Plot

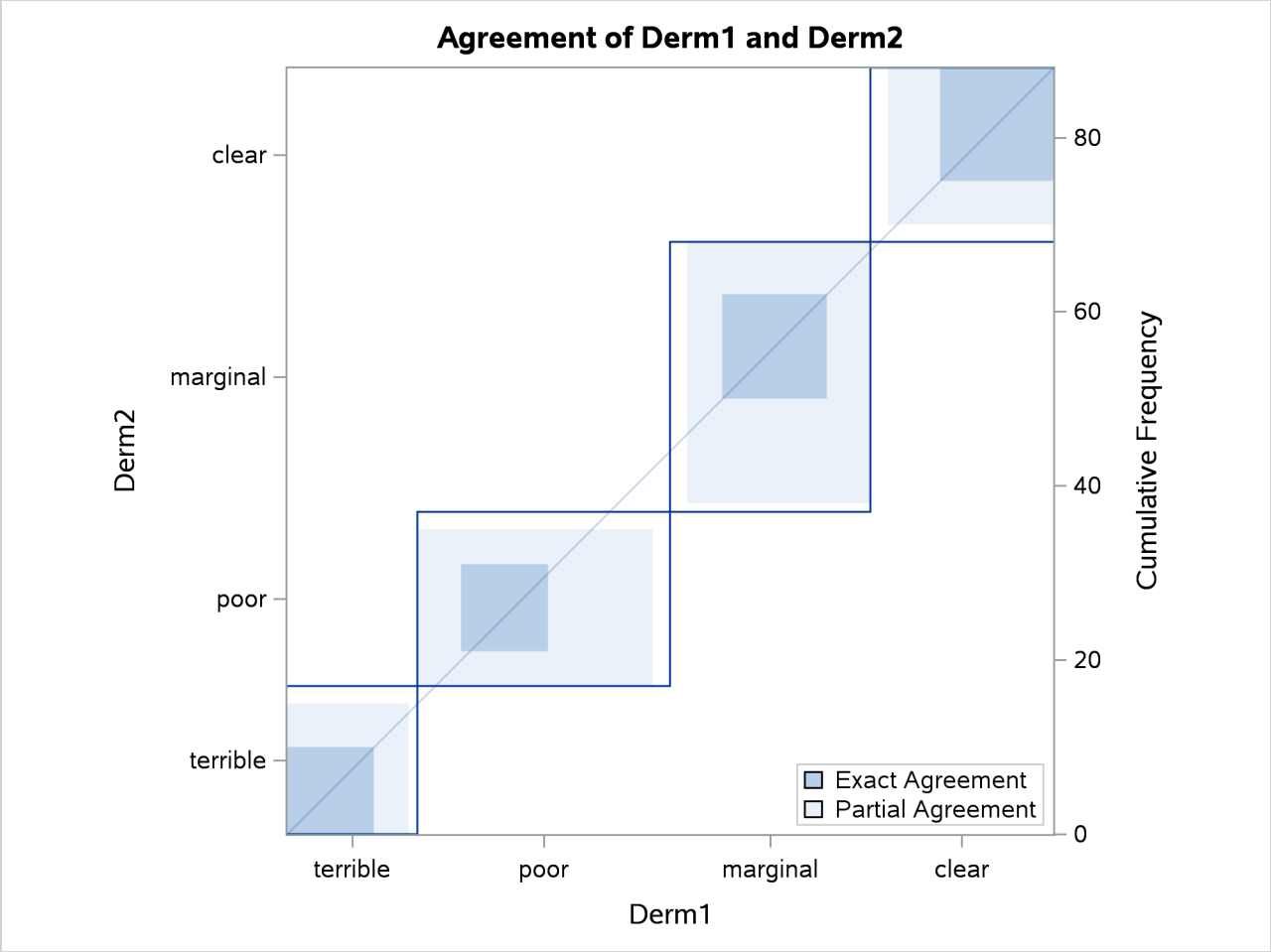

Syntax: FREQ Procedure

The following statements are available in the FREQ procedure:

```
PROC FREQ <options> ;
    BY variables ;
    EXACT statistic-options </ computation-options> ;
    OUTPUT <OUT=SAS-data-set> output-options ;
    TABLES requests </ options> ;
    TEST options ;
    WEIGHT variable </ option> ;
```

The PROC FREQ statement is the only required statement for the FREQ procedure. If you specify the following statements, PROC FREQ produces a one-way frequency table for each variable in the most recently created data set.

```
proc freq;
run;
```

Table 47.3 summarizes the basic functions of the procedure statements. The following sections provide detailed syntax information for the BY, EXACT, OUTPUT, TABLES, TEST, and WEIGHT statements in alphabetical order after the description of the PROC FREQ statement.

Table 47.3 Summary of PROC FREQ Statements

Statement	Description
BY	Provides separate analyses for each BY group
EXACT	Requests exact tests
OUTPUT	Requests an output data set
TABLES	Specifies tables and requests analyses
TEST	Requests tests for measures of association and agreement
WEIGHT	Identifies a weight variable

PROC FREQ Statement

PROC FREQ <*options*> **;**

The PROC FREQ statement invokes the FREQ procedure. Optionally, it also identifies the input data set. By default, the procedure uses the most recently created SAS data set.

Table 47.4 lists the *options* available in the PROC FREQ statement. Descriptions of the *options* follow in alphabetical order.

Table 47.4 PROC FREQ Statement Options

Option	Description
COMPRESS	Begins the next one-way table on the current page
DATA=	Names the input data set
FORMCHAR=	Specifies the outline and cell divider characters for crosstabulation tables
NLEVELS	Displays the number of levels for all TABLES variables
NOPRINT	Suppresses all displayed output
ORDER=	Specifies the order for reporting variable values
PAGE	Displays one table per page

You can specify the following *options*:

COMPRESS

begins display of the next one-way frequency table on the same page as the preceding one-way table if there is enough space to begin the table. By default, the next one-way table begins on the current page only if the entire table fits on that page. The COMPRESS option is not valid with the PAGE option.

DATA=*SAS-data-set*

names the *SAS-data-set* to be analyzed by PROC FREQ. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

FORMCHAR(1,2,7)=*'formchar-string'*

defines the characters to use to construct cell outlines and dividers for crosstabulation tables. This option affects only the SAS monospace output destination.

PROC FREQ uses 3 of the 20 formatting characters that SAS provides. You can specify three characters in *formchar-string* to draw the vertical separators (1), the horizontal separators (2), and the vertical-horizontal intersections (7) in crosstabulation tables. By default, PROC FREQ uses FORMCHAR(1,2,7)='|-+'. Table 47.5 summarizes the formatting characters that PROC FREQ uses.

Table 47.5 Formatting Characters Used by PROC FREQ

Position	Default	Used to Draw
1	|	Vertical separators
2	-	Horizontal separators
7	+	Intersections of vertical and horizontal separators

To produce crosstabulation tables that have no cell outlines or dividers, you can specify all blanks in *formchar-string*—for example, FORMCHAR(1,2,7)=' ' (three blanks).

You can use any characters in *formchar-string*. If you use hexadecimal characters, you must put **x** after the closing quotation mark. For information about which hexadecimal codes to use for which characters, see the documentation for your hardware.

For more information about formatting characters, see the TABULATE procedure in the *Base SAS Procedures Guide*.

NLEVELS

displays the "Number of Variable Levels" table, which provides the number of levels for each variable named in the TABLES statements. For more information, see the section "Number of Variable Levels Table" on page 3326. PROC FREQ determines the variable levels from the formatted variable values, as described in the section "Grouping with Formats" on page 3242.

NOPRINT

suppresses the display of all output. You can use the NOPRINT option when you only want to create an output data set. See the section "Output Data Sets" on page 3323 for information about the output data sets produced by PROC FREQ. Note that the NOPRINT option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

NOTE: A NOPRINT option is also available in the TABLES statement. It suppresses display of the crosstabulation tables but allows display of the requested statistics.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the order of the variable levels in the frequency and crosstabulation tables, which you request in the TABLES statement.

The ORDER= option can take the following values:

Value of ORDER=	Levels Ordered By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=INTERNAL. The FORMATTED and INTERNAL orders are machine-dependent. The ORDER= option does not apply to missing values, which are always ordered first.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

PAGE

displays only one table per page. Otherwise, PROC FREQ displays multiple tables per page as space permits. The PAGE option is not valid with the COMPRESS option.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC FREQ to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the FREQ procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the “Grouping Data” section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

EXACT Statement

EXACT *statistic-options* < / *computation-options* > **;**

The EXACT statement requests exact tests and confidence limits for selected statistics. The *statistic-options* identify which statistics to compute, and the *computation-options* specify options for computing exact statistics. For more information, see the section "Exact Statistics" on page 3318.

NOTE: PROC FREQ computes exact tests by using fast and efficient algorithms that are superior to direct enumeration. Exact tests are appropriate when a data set is small, sparse, skewed, or heavily tied. For some large problems, computation of exact tests might require a large amount of time and memory. Consider using asymptotic tests for such problems. Alternatively, when asymptotic methods might not be sufficient for such large problems, consider using Monte Carlo estimation of exact p-values. You can request Monte Carlo estimation by specifying the MC *computation-option* in the EXACT statement. For more information, see the section "Computational Resources" on page 3320.

Statistic Options

The *statistic-options* specify which exact tests and confidence limits to compute. Table 47.6 lists the available *statistic-options* and the exact statistics that are computed. Descriptions of the *statistic-options* follow Table 47.6 in alphabetical order.

For one-way tables, PROC FREQ provides exact p-values for the binomial proportion test, the chi-square goodness-of-fit test, and the likelihood ratio chi-square test. PROC FREQ also provides exact (Clopper-Pearson) confidence limits for the binomial proportion.

For two-way tables, PROC FREQ provides exact p-values for the following tests: Pearson chi-square test, likelihood ratio chi-square test, Mantel-Haenszel chi-square test, Fisher's exact test, Jonckheere-Terpstra test, Cochran-Armitage test for trend, and the symmetry test. PROC FREQ also provides exact p-values for tests of the following statistics: Pearson correlation coefficient, Spearman correlation coefficient, Kendall's tau-b, Stuart's tau-c, Somers' $D(C|R)$, Somers' $D(R|C)$, simple kappa coefficient, and weighted kappa coefficient.

For 2×2 tables, PROC FREQ provides the exact McNemar's test, exact confidence limits for the odds ratio, and Barnard's unconditional exact test for the risk (proportion) difference. PROC FREQ also provides exact unconditional confidence limits for the risk (proportion) difference and for the relative risk (ratio of proportions). For stratified 2×2 tables, PROC FREQ provides Zelen's exact test for equal odds ratios, exact confidence limits for the common odds ratio, and an exact test for the common odds ratio.

Most of the *statistic-option* names listed in Table 47.6 are identical to the corresponding option names in the TABLES and OUTPUT statements. You can request exact computations for groups of statistics by using *statistic-options* that are identical to the TABLES statement options CHISQ, MEASURES, and AGREE. For example, when you specify the CHISQ *statistic-option* in the EXACT statement, PROC FREQ computes exact p-values for the Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests for two-way tables. You can request an exact test for an individual statistic by specifying the corresponding *statistic-option* from the list in Table 47.6.

Using the EXACT Statement with the TABLES Statement

You must use a TABLES statement with the EXACT statement. If you use only one TABLES statement, you do not need to specify the same options in both the TABLES and EXACT statements; when you specify a

statistic-option in the EXACT statement, PROC FREQ automatically invokes the corresponding TABLES statement option. However, when you use an EXACT statement with multiple TABLES statements, you must specify options in the TABLES statements to request statistics. PROC FREQ then provides exact tests or confidence limits for those statistics that you also specify in the EXACT statement.

Table 47.6 EXACT Statement Statistic Options

Statistic Option	Exact Statistics
AGREE	McNemar's test (for 2×2 tables), simple kappa test, weighted kappa test
BARNARD	Barnard's test (for 2×2 tables)
BINOMIAL | BIN	Binomial proportion tests for one-way tables
CHISQ	Chi-square goodness-of-fit test for one-way tables; Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests for two-way tables
COMOR	Confidence limits for the common odds ratio, common odds ratio test (for $h \times 2 \times 2$ tables)
EQOR | ZELEN	Zelen's test for equal odds ratios (for $h \times 2 \times 2$ tables)
FISHER	Fisher's exact test
JT	Jonckheere-Terpstra test
KAPPA	Test for the simple kappa coefficient
KENTB | TAUB	Test for Kendall's tau-b
LRCHI	Likelihood ratio chi-square test (one-way and two-way tables)
MCNEM	McNemar's test (for 2×2 tables)
MEASURES	Tests for the Pearson correlation and Spearman correlation, confidence limits for the odds ratio (for 2×2 tables)
MHCHI	Mantel-Haenszel chi-square test
OR | ODDSRATIO	Confidence limits for the odds ratio (for 2×2 tables)
PCHI	Pearson chi-square test (one-way and two-way tables)
PCORR	Test for the Pearson correlation coefficient
RELRISK	Confidence limits for the relative risk (for 2×2 tables)
RISKDIFF	Confidence limits for the risk difference (for 2×2 tables)
SCORR	Test for the Spearman correlation coefficient
SMDCR	Test for Somers' $D(C\|R)$
SMDRC	Test for Somers' $D(R\|C)$
STUTC | TAUC	Test for Stuart's tau-c
SYMMETRY | BOWKER	Symmetry test
TREND	Cochran-Armitage test for trend
WTKAPPA | WTKAP	Test for the weighted kappa coefficient

You can specify the following *statistic-options*:

AGREE

requests McNemar's exact test, an exact test for the simple kappa coefficient, and an exact test for the weighted kappa coefficient. For more information, see the sections "Tests and Measures of Agreement" on page 3300 and "Exact Statistics" on page 3318.

For McNemar's test, you can specify the null hypothesis ratio of discordant proportions by using the AGREE(MNULLRATIO=) option in the TABLES statement; by default, MNULLRATIO=1. For the weighted kappa coefficient, you can request Fleiss-Cohen weights by specifying the AGREE(WT=FC) option in the TABLES statement; by default, PROC FREQ computes the weighted kappa coefficient by using Cicchetti-Allison agreement weights.

McNemar's test is available for 2×2 tables. Kappa coefficients are defined only for square two-way tables, where the number of rows equals the number of columns. If your table is not square because some observations have weights of 0, you can specify the ZEROS option in the WEIGHT statement to include these observations in the analysis. For more information, see the section "Tables with Zero-Weight Rows or Columns" on page 3307.

For 2×2 tables, the weighted kappa coefficient is equivalent to the simple kappa coefficient, and PROC FREQ displays only analyses for the simple kappa coefficient.

BARNARD

requests Barnard's exact unconditional test for the risk (proportion) difference for 2×2 tables. For more information, see the section "Barnard's Unconditional Exact Test" on page 3282.

To request exact unconditional confidence limits for the risk difference. you can specify the RISKDIFF option in the EXACT statement. The RISKDIFF option in the TABLES statement provides asymptotic tests and several types of confidence limits for the risk difference. For more information, see the section "Risks and Risk Differences" on page 3272.

BINOMIAL

BIN

requests an exact test for the binomial proportion (for one-way tables). For more information, see the section "Binomial Tests" on page 3267. You can specify the null hypothesis proportion by using the BINOMIAL(P=) option in the TABLES statement; by default, P=0.5.

The BINOMIAL option in the TABLES statement provides exact (Clopper-Pearson) confidence limits for the binomial proportion by default. You can specify the BINOMIAL(CL=MIDP) option in the TABLES statement to request exact mid-p confidence limits for the binomial proportion. The BINOMIAL option in the TABLES statement also provides asymptotic (Wald) tests and several other confidence limit types for the binomial proportion. For more information, see the section "Binomial Proportion" on page 3263.

CHISQ

requests the following exact chi-square tests for two-way tables: Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square. For more information, see the section "Chi-Square Tests and Statistics" on page 3248. The CHISQ option in the TABLES statement provides asymptotic tests for these statistics.

For one-way tables, the CHISQ option requests an exact chi-square goodness-of-fit test. You can specify null hypothesis proportions for this test by using the CHISQ(TESTP=) option in the TABLES statement. By default, the one-way chi-square test is based on the null hypothesis of equal proportions. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248.

COMOR

requests an exact test and exact confidence limits for the common odds ratio for multiway 2×2 tables. For more information, see the section "Exact Confidence Limits for the Common Odds Ratio" on page 3315. The CMH option in the TABLES statement provides Mantel-Haenszel and logit estimates of the common odds ratio along with their asymptotic confidence limits.

EQOR

ZELEN

requests Zelen's exact test for equal odds ratios for multiway 2×2 tables. For more information, see the section "Zelen's Exact Test for Equal Odds Ratios" on page 3314. The CMH option in the TABLES statement provides an (asymptotic) Breslow-Day test for homogeneity of odds ratios.

FISHER

requests Fisher's exact test. For more information, see the sections "Fisher's Exact Test" on page 3252 and "Exact Statistics" on page 3318. For 2×2 tables, the CHISQ option in the TABLES statement provides Fisher's exact test. For general $R \times C$ tables, Fisher's exact test is also known as the Freeman-Halton test.

JT

requests an exact Jonckheere-Terpstra test. For more information, see the sections "Jonckheere-Terpstra Test" on page 3298 and "Exact Statistics" on page 3318. The JT option in the TABLES statement provides an asymptotic Jonckheere-Terpstra test.

KAPPA

requests an exact test for the simple kappa coefficient. For more information, see the sections "Simple Kappa Coefficient" on page 3301 and "Exact Statistics" on page 3318. The AGREE option in the TABLES statement provides the simple kappa estimate, standard error, and confidence limits. The KAPPA option in the TEST statement provides an asymptotic test for the simple kappa coefficient.

Kappa coefficients are defined only for square two-way tables, where the number of rows equals the number of columns. If your table is not square because some observations have weights of 0, you can specify the ZEROS option in the WEIGHT statement to include these observations in the analysis. For more information, see the section "Tables with Zero-Weight Rows or Columns" on page 3307.

KENTB

TAUB

requests an exact test for Kendall's tau-*b*. For more information, see the sections "Kendall's Tau-b" on page 3255 and "Exact Statistics" on page 3318. The MEASURES option in the TABLES statement provides an estimate and standard error of Kendall's tau-*b*. The KENTB option in the TEST statement provides an asymptotic test for Kendall's tau-*b*.

LRCHI

requests an exact test for the likelihood ratio chi-square for two-way tables. For more information, see the sections "Likelihood Ratio Chi-Square Test" on page 3251 and "Exact Statistics" on page 3318. The CHISQ option in the TABLES statement provides an asymptotic likelihood ratio chi-square test for two-way tables.

For one-way tables, the LRCHI option requests an exact likelihood ratio goodness-of-fit test. You can specify null hypothesis proportions by using the CHISQ(TESTP=) option in the TABLES statement. By default, the one-way test is based on the null hypothesis of equal proportions. For more information, see the section "Likelihood Ratio Chi-Square Test for One-Way Tables" on page 3250.

MCNEM

requests an exact McNemar's test. For more information, see the sections "McNemar's Test" on page 3300 and "Exact Statistics" on page 3318. You can specify the null hypothesis ratio of discordant proportions by using the AGREE(MNULLRATIO=) option in the TABLES statement; by default, MNULLRATIO=1. The AGREE option in the TABLES statement provides an asymptotic McNemar's test.

MEASURES

requests exact tests for the Pearson and Spearman correlations. For more information, see the sections "Pearson Correlation Coefficient" on page 3257, "Spearman Rank Correlation Coefficient" on page 3258, and "Exact Statistics" on page 3318. The PCORR and SCORR options in the TEST statement provide asymptotic tests for the Pearson and Spearman correlations, respectively.

The MEASURES option also requests exact confidence limits for the odds ratio for 2×2 tables. For more information, see the subsection Exact Confidence Limits in the section "Confidence Limits for the Odds Ratio" on page 3288. You can also request exact confidence limits for the odds ratio by specifying the OR option in the EXACT statement.

MHCHI

requests an exact test for the Mantel-Haenszel chi-square. For more information, see the sections "Mantel-Haenszel Chi-Square Test" on page 3251 and "Exact Statistics" on page 3318. The CHISQ option in the TABLES statement provides an asymptotic Mantel-Haenszel chi-square test.

OR

ODDSRATIO

requests exact confidence limits for the odds ratio for 2×2 tables. For more information, see the subsection "Exact Confidence Limits" in the section "Confidence Limits for the Odds Ratio" on page 3288.

You can request exact mid-p confidence limits for the odds ratio by specifying the OR(CL=MIDP) option in the TABLES statement. The OR(CL=) option in the TABLES statement also provides other types of confidence limits for the odds ratio. For more information, see the section "Confidence Limits for the Odds Ratio" on page 3288.

The ALPHA= option in the TABLES statement determines the confidence level of the exact confidence limits; by default, ALPHA=0.05, which produces 95% confidence limits for the odds ratio.

PCHI

requests an exact test for the Pearson chi-square for two-way tables. For more information, see the sections "Pearson Chi-Square Test for Two-Way Tables" on page 3249 and "Exact Statistics" on page 3318. The CHISQ option in the TABLES statement provides an asymptotic Pearson chi-square test.

For one-way tables, the PCHI option requests an exact chi-square goodness-of-fit test. You can specify null hypothesis proportions by using the CHISQ(TESTP=) option in the TABLES statement. By default, the goodness-of-fit test is based on the null hypothesis of equal proportions. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248.

PCORR

requests an exact test for the Pearson correlation coefficient. For more information, see the sections "Pearson Correlation Coefficient" on page 3257 and "Exact Statistics" on page 3318. The MEASURES option in the TABLES statement provides the estimate and standard error of the Pearson correlation. The PCORR option in the TEST statement provides an asymptotic test for the Pearson correlation.

RELRISK < (*options*) >

requests exact unconditional confidence limits for the relative risk for 2×2 tables. By default, the exact confidence limits are computed by inverting two separate one-sided exact tests that are based on the score statistic (Chan and Zhang 1999). For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291.

The RELRISK(CL=) option in the TABLES statement provides additional types of confidence limits for the relative risk. For more information, see the section "Confidence Limits for the Risk Difference" on page 3274.

The ALPHA= option in the TABLES statement determines the confidence level; by default, ALPHA=0.05, which produces 95% confidence limits for the relative risk.

You can specify the following *options*:

COLUMN=1 | 2 | BOTH

specifies the table column of the relative risk. By default, COLUMN=1, which provides exact confidence limits for the column 1 relative risk. COLUMN=BOTH provides exact confidence limits for both column 1 and column 2 relative risks.

METHOD=NOSCORE | SCORE | SCORE2

specifies the computation method for the exact confidence limits. By default, METHOD=SCORE.

You can specify one of the following methods:

NOSCORE

computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the unstandardized relative risk (Santner and Snell 1980). For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291.

SCORE

computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the score statistic (Chan and Zhang 1999). For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291.

SCORE2

computes the exact confidence limits by inverting a single two-sided exact test that is based on the score statistic (Agresti and Min 2001). For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291.

RISKDIFF < (*options*) >

requests exact unconditional confidence limits for the risk difference for 2 × 2 tables. By default, the exact confidence limits are computed by inverting two separate one-sided exact tests that are based on the score statistic (Chan and Zhang 1999). For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

The RISKDIFF(CL=) option in the TABLES statement provides additional types of confidence limits for the risk difference. For more information, see the section "Confidence Limits for the Risk Difference" on page 3274.

The ALPHA= option in the TABLES statement determines the confidence level; by default, ALPHA=0.05, which produces 95% confidence limits for the risk difference.

You can specify the following *options*:

COLUMN=1 | 2 | BOTH

specifies the table column of the risk difference. By default, COLUMN=BOTH and the exact confidence limits are displayed in the 'Risk Estimates' tables. If you specify the RISKDIFF(NORISKS) option in the TABLES statement to suppress the 'Risk Estimates' tables, by default, COLUMN=1 and the exact confidence limits are displayed in the 'Risk Difference Confidence Limits' table.

METHOD=NOSCORE | SCORE | SCORE2

specifies the computation method for the exact confidence limits. By default, METHOD=SCORE.

You can specify one of the following methods:

NOSCORE

computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the unstandardized risk difference (Santner and Snell 1980). For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

SCORE

computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the score statistic (Chan and Zhang 1999). For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

SCORE2

computes the exact confidence limits by inverting a single two-sided exact test that is based on the score statistic (Agresti and Min 2001). For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

SCORR

requests an exact test for the Spearman correlation coefficient. For more information, see the sections "Spearman Rank Correlation Coefficient" on page 3258 and "Exact Statistics" on page 3318. The MEASURES option in the TABLES statement provides the estimate and standard error of the Spearman correlation. The SCORR option in the TEST statement provides an asymptotic test for the Spearman correlation.

SMDCR

requests an exact test for Somers' $D(C|R)$. For more information, see the sections "Somers' D" on page 3257 and "Exact Statistics" on page 3318. The MEASURES option in the TABLES statement provides the estimate and standard error of Somers' $D(C|R)$. The SMDCR option in the TEST statement provides an asymptotic test for Somers' $D(C|R)$.

SMDRC

requests an exact test for Somers' $D(R|C)$. For more information, see the sections "Somers' D" on page 3257 and "Exact Statistics" on page 3318. The MEASURES option in the TABLES statement provides the estimate and standard error of Somers' $D(R|C)$. The SMDRC option in the TEST statement provides an asymptotic test for Somers' $D(C|R)$.

STUTC

TAUC

requests an exact test for Stuart's tau-c. For more information, see the sections "Stuart's Tau-c" on page 3256 and "Exact Statistics" on page 3318. The MEASURES option in the TABLES statement provides the estimate and standard error of Stuart's tau-c. The STUTC option in the TEST statement provides an asymptotic test for Stuart's tau-c.

SYMMETRY

BOWKER

requests an exact symmetry test. This test is available for square $R \times R$ two-way tables where the table dimension R is greater than 2. For more information, see the section "Exact Symmetry Test" on page 3301. The AGREE option in the TABLES statement provides an asymptotic symmetry test.

TREND

requests the exact Cochran-Armitage test for trend. For more information, see the sections "Cochran-Armitage Test for Trend" on page 3297 and "Exact Statistics" on page 3318. The TREND option in the TABLES statement provides an asymptotic Cochran-Armitage test for trend. This test is available for tables of dimensions $2 \times C$ or $R \times 2$.

WTKAPPA

WTKAP

requests an exact test for the weighted kappa coefficient. For more information, see the sections "Weighted Kappa Coefficient" on page 3303 and "Exact Statistics" on page 3318. By default, PROC FREQ computes the weighted kappa coefficient by using Cicchetti-Allison agreement weights. You can request Fleiss-Cohen agreement weights by specifying the AGREE(WT=FC) option in the TABLES statement.

Kappa coefficients are defined only for square two-way tables, where the number of rows equals the number of columns. If your table is not square because some observations have weights of 0, you can specify the ZEROS option in the WEIGHT statement to include these observations in the analysis. For more information, see the section "Tables with Zero-Weight Rows or Columns" on page 3307.

For 2×2 tables, the weighted kappa coefficient is equivalent to the simple kappa coefficient, and PROC FREQ displays only analyses for the simple kappa coefficient.

Computation Options

The *computation-options* specify options for computing exact statistics. You can specify the following *computation-options*:

ALPHA=α

specifies the level of the confidence limits for Monte Carlo p-value estimates. The value of α must be between 0 and 1; a confidence level of α produces $100(1-\alpha)\%$ confidence limits. By default ALPHA=0.01, which produces 99% confidence limits for the Monte Carlo estimates.

This option invokes the MC option.

MAXTIME=*value*

specifies the maximum clock time (in seconds) that PROC FREQ can use to compute an exact p-value. If the procedure does not complete the computation within the specified time, the computation terminates. The maximum time *value* must be a positive number. This option is available for exact p-value computation and for Monte Carlo estimation of exact p-values. For more information, see the section "Computational Resources" on page 3320.

MC

requests Monte Carlo estimation of exact p-values instead of direct exact p-value computation. Monte Carlo estimation can be useful for large problems where exact computations require a substantial amount of time and memory but asymptotic approximations might not be sufficient. For more information, see the section "Monte Carlo Estimation" on page 3321.

This option is available for all EXACT *statistic-options* except the BINOMIAL option and the following options that apply only to 2×2 or $h \times 2 \times 2$ tables: BARNARD, COMOR, EQOR, MCNEM, OR, RELRISK, and RISKDIFF. PROC FREQ always computes exact tests or confidence limits (not Monte Carlo estimates) for these statistics.

The ALPHA=, N=, and SEED= options invoke the MC option.

MIDP

requests exact mid p-values for the exact tests. The exact mid p-value is defined as the exact p-value minus half the exact point probability. For more information, see the section "Definition of p-Values" on page 3319.

The MIDP option is available for all EXACT statement *statistic-options* except the following: BARNARD, EQOR, OR, RELRISK, and RISKDIFF. You cannot specify both the MIDP option and the MC option.

N=*n*

specifies the number of samples for Monte Carlo estimation. The value of *n* must be a positive integer. Larger values of *n* produce more precise estimates of exact p-values. Because larger values of *n* generate more samples, the computation time increases. By default, N=10,000.

This option invokes the MC option.

PFORMAT=*format-name* | **EXACT**

specifies the display format for exact *p*-values. PROC FREQ applies this format to one- and two-sided exact *p*-values, exact point probabilities, and exact mid *p*-values. By default, PROC FREQ displays exact *p*-values in the PVALUE6.4 format.

You can provide a *format-name* or you can specify PFORMAT=EXACT to control the format of exact *p*-values. The value of *format-name* can be any standard SAS numeric format or a user-defined format. The format length must not exceed 24. For information about formats, see the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS format in *SAS Formats and Informats: Reference*.

If you specify PFORMAT=EXACT, PROC FREQ uses the 6.4 format to display exact *p*-values that are greater than or equal to 0.001; the procedure uses the E10.3 format to display values that are between 0.000 and 0.001.

POINT

requests exact point probabilities for the exact tests. The exact point probability is the exact probability that the test statistic equals the observed value. For more information, see the section "Definition of *p*-Values" on page 3319.

The POINT option is available for all EXACT statement *statistic-options* except the following: BARNARD, EQOR, OR, RELRISK, and RISKDIFF. You cannot specify both the POINT option and the MC option.

SEED=*number*

specifies the initial seed for random number generation for Monte Carlo estimation. The value of the SEED= option must be an integer. If you do not specify the SEED= option or if the SEED= value is negative or 0, PROC FREQ uses the time of day from the computer's clock to obtain the initial seed.

This option invokes the MC option.

OUTPUT Statement

OUTPUT <**OUT=***SAS-data-set*> *output-options* **;**

The OUTPUT statement creates a SAS data set that stores statistics that PROC FREQ computes. Table 47.7 lists the statistics that are available in the output data set. You can identify which statistics to include by specifying *output-options*.

When you specify an OUTPUT statement, you must also specify a TABLES statement, which defines tables and requests statistical analyses. The OUTPUT statement stores statistics for a single table request. If you use multiple TABLES statements, the contents of the output data set correspond to the last TABLES statement. If you use multiple table requests in a single TABLES statement, the contents of the output data set correspond to the last table request. Only one OUTPUT statement can be specified in a single invocation of PROC FREQ.

For a one-way or two-way table, the output data set contains one observation that stores statistics for the table. For a multiway table, the output data set contains an observation for each two-way subtable (stratum) of the multiway table. If you request summary statistics for the multiway table, the output data set also contains an observation that includes the across-strata summary statistics. If you specify a BY statement, the output data

set contains an observation or set of observations for each BY group. For more information, see the section "Contents of the OUTPUT Statement Output Data Set" on page 3324.

This output data set (which the OUTPUT statement produces) is not the same as the output data set that the OUT= option in the TABLES statement produces. The OUTPUT statement creates a data set that contains statistics (such as the Pearson chi-square and its p-value), while the OUT= option in the TABLES statement creates a data set that contains frequency counts and percentages. For more information, see the section "Output Data Sets" on page 3323.

You can also use the Output Delivery System (ODS) to store statistics that PROC FREQ computes. ODS can create a SAS data set from any table that PROC FREQ produces. For more information, see the section "ODS Table Names" on page 3336 and Chapter 23, "Using the Output Delivery System."

You can specify the following *options* in the OUTPUT statement:

OUT=*SAS-data-set*
specifies the name of the output data set. When you use an OUTPUT statement but do not use the OUT= option, PROC FREQ creates a data set and names it by using the DATA*n* convention.

output-options
specify the statistics to include in the output data set. Table 47.7 lists the *output-options* that are available in the OUTPUT statement, together with the TABLES statement options that are required to produce the statistics. Descriptions of the *output-options* follow the table in alphabetical order.

You can specify *output-options* to request individual statistics, or you can request groups of statistics by using *output-options* that are identical to the group options in the TABLES statement (for example, the CHISQ, MEASURES, CMH, AGREE, and ALL options).

When you specify an *output-option*, the output data set includes statistics from the corresponding analysis. In addition to the estimate or test statistic, the output data set includes associated values such as standard errors, confidence limits, p-values, and degrees of freedom. For more information, see the section "Contents of the OUTPUT Statement Output Data Set" on page 3324.

To store a statistic in the output data set, you must also request computation of that statistic with the appropriate TABLES, EXACT, or TEST statement option. For example, the PCHI *output-option* includes the Pearson chi-square in the output data set. You must also request computation of the Pearson chi-square by specifying the CHISQ option in the TABLES statement. Or, if you use only one TABLES statement, you can request computation of the Pearson chi-square by specifying the PCHI or CHISQ option in the EXACT statement. Table 47.7 lists the TABLES statement options that are required to produce the OUTPUT data set statistics.

Table 47.7 OUTPUT Statement Output Options

Output Option	Output Data Set Statistics	Required TABLES Statement Option
AGREE	McNemar's test (2×2 tables), Bowker's test, simple and weighted kappas; for multiple strata, overall simple and weighted kappas, tests for equal kappas, and Cochran's Q ($2 \times 2 \times \cdots \times 2$ tables)	AGREE
AJCHI	Continuity-adjusted chi-square (2×2 tables)	CHISQ

Table 47.7 *continued*

Output Option	Output Data Set Statistics	Required TABLES Statement Option
ALL	CHISQ, MEASURES, and CMH statistics; N (number of observations)	ALL
BDCHI	Breslow-Day test ($h \times 2 \times 2$ tables)	CMH, CMH1, or CMH2
BINOMIAL | BIN	Binomial statistics (one-way tables)	BINOMIAL
CHISQ	For one-way tables, goodness-of-fit test; for two-way tables, Pearson, likelihood ratio, continuity-adjusted, and Mantel-Haenszel chi-squares, Fisher's exact test (2×2 tables), phi and contingency coefficients, Cramér's V	CHISQ
CMH	Cochran-Mantel-Haenszel (CMH) correlation, row mean scores (ANOVA), and general association statistics; for 2×2 tables, logit and Mantel-Haenszel common odds ratios and relative risks, Breslow-Day test	CMH
CMH1	CMH statistics, except row mean scores (ANOVA) and general association statistics	CMH or CMH1
CMH2	CMH statistics, except general association statistic	CMH or CMH2
CMHCOR	CMH correlation statistic	CMH, CMH1, or CMH2
CMHGA	CMH general association statistic	CMH
CMHRMS	CMH row mean scores (ANOVA) statistic	CMH or CMH2
COCHQ	Cochran's Q ($2 \times 2 \times \cdots \times 2$ tables)	AGREE or CQ
CONTGY	Contingency coefficient	CHISQ
CRAMV	Cramér's V	CHISQ
EQKAP	Test for equal simple kappas	AGREE
EQOR | ZELEN	Zelen's test for equal odds ratios ($h \times 2 \times 2$ tables)	CMH and EXACT EQOR
EQWKP	Test for equal weighted kappas	AGREE
FISHER	Fisher's exact test	CHISQ or FISHER [1]
GAILSIMON | GS	Gail-Simon test	GAILSIMON
GAMMA	Gamma	MEASURES
JT	Jonckheere-Terpstra test	JT
KAPPA	Simple kappa coefficient	AGREE
KENTB | TAUB	Kendall's tau-b	MEASURES
LAMCR	Lambda asymmetric ($C\|R$)	MEASURES
LAMDAS	Lambda symmetric	MEASURES
LAMRC	Lambda asymmetric ($R\|C$)	MEASURES
LGOR	Logit common odds ratio	CMH, CMH1, or CMH2
LGRRC1	Logit common relative risk, column 1	CMH, CMH1, or CMH2
LGRRC2	Logit common relative risk, column 2	CMH, CMH1, or CMH2
LRCHI	Likelihood ratio chi-square	CHISQ
MCNEM	McNemar's test (2×2 tables)	AGREE

[1] CHISQ computes Fisher's exact test for 2×2 tables. Use the FISHER option to compute Fisher's exact test for general $r \times c$ tables.

Table 47.7 *continued*

Output Option	Output Data Set Statistics	Required TABLES Statement Option
MEASURES	Gamma, Kendall's tau-*b*, Stuart's tau-*c*, Somers' $D(C\|R)$ and $D(R\|C)$, Pearson and Spearman correlations, lambda asymmetric $(C\|R)$ and $(R\|C)$, lambda symmetric, uncertainty coefficients $(C\|R)$ and $(R\|C)$, symmetric uncertainty coefficient; odds ratio and relative risks (2×2 tables)	MEASURES
MHCHI	Mantel-Haenszel chi-square	CHISQ
MHOR | COMOR	Mantel-Haenszel common odds ratio	CMH, CMH1, or CMH2
MHRRC1	Mantel-Haenszel common relative risk, column 1	CMH, CMH1, or CMH2
MHRRC2	Mantel-Haenszel common relative risk, column 2	CMH, CMH1, or CMH2
N	Number of nonmissing observations	
NMISS	Number of missing observations	
OR | ODDSRATIO	Odds ratio (2×2 tables)	MEASURES, OR, or RELRISK
PCHI	Chi-square goodness-of-fit test (one-way tables), Pearson chi-square (two-way tables)	CHISQ
PCORR	Pearson correlation coefficient	MEASURES
PHI	Phi coefficient	CHISQ
PLCORR	Polychoric correlation coefficient	PLCORR
RDIF1	Column 1 risk difference (row 1 – row 2)	RISKDIFF
RDIF2	Column 2 risk difference (row 1 – row 2)	RISKDIFF
RELRISK	Odds ratio and relative risks (2×2 tables)	MEASURES or RELRISK
RISKDIFF	Risks and risk differences (2×2 tables)	RISKDIFF
RISKDIFF1	Risks and risk difference, column 1	RISKDIFF
RISKDIFF2	Risks and risk difference, column 2	RISKDIFF
RRC1 | RELRISK1	Relative risk, column 1	MEASURES or RELRISK
RRC2 | RELRISK2	Relative risk, column 2	MEASURES or RELRISK
RSK1 | RISK1	Column 1 overall risk	RISKDIFF
RSK11 | RISK11	Column 1 risk for row 1	RISKDIFF
RSK12 | RISK12	Column 2 risk for row 1	RISKDIFF
RSK2 | RISK2	Column 2 overall risk	RISKDIFF
RSK21 | RISK21	Column 1 risk for row 2	RISKDIFF
RSK22 | RISK22	Column 2 risk for row 2	RISKDIFF
SCORR	Spearman correlation coefficient	MEASURES
SMDCR	Somers' $D(C\|R)$	MEASURES
SMDRC	Somers' $D(R\|C)$	MEASURES
STUTC | TAUC	Stuart's tau-*c*	MEASURES
TREND	Cochran-Armitage test for trend	TREND
TSYMM | BOWKER	Bowker's symmetry test	AGREE
U	Symmetric uncertainty coefficient	MEASURES
UCR	Uncertainty coefficient $(C\|R)$	MEASURES

Table 47.7 *continued*

Output Option	Output Data Set Statistics	Required TABLES Statement Option
URC	Uncertainty coefficient $(R\|C)$	MEASURES
WTKAPPA | WTKAP	Weighted kappa coefficient	AGREE

You can specify the following *output-options*:

AGREE

includes the following tests and measures of agreement in the output data set: McNemar's test (for 2×2 tables), Bowker's symmetry test, the simple kappa coefficient, and the weighted kappa coefficient. For multiway tables, the AGREE option also includes the following statistics in the output data set: overall simple and weighted kappa coefficients, tests for equal simple and weighted kappa coefficients, and Cochran's Q test.

AGREE statistics are computed only for square tables, where the number of rows equals the number of columns. PROC FREQ provides Bowker's symmetry test and weighted kappa coefficients for tables larger than 2×2. (For 2×2 tables, Bowker's test is identical to McNemar's test, and the weighted kappa coefficient equals the simple kappa coefficient.) Cochran's Q is available for $2 \times 2 \times \cdots \times 2$ tables.

The AGREE option in the TABLES statement requests computation of tests and measures of agreement. For more information, see the section "Tests and Measures of Agreement" on page 3300.

AJCHI

includes the continuity-adjusted chi-square in the output data set. The continuity-adjusted chi-square is available for 2×2 tables and is provided by the CHISQ option in the TABLES statement. For more information, see the section "Continuity-Adjusted Chi-Square Test" on page 3251.

ALL

includes all statistics that are requested by the CHISQ, MEASURES, and CMH *output-options* in the output data set. ALL also includes the number of nonmissing observations (total frequency), which can be requested separately by the N *output-option*.

BDCHI

includes the Breslow-Day test in the output data set. The Breslow-Day test for homogeneity of odds ratios is computed for multiway 2×2 tables and is provided by the CMH, CMH1, and CMH2 options in the TABLES statement. For more information, see the section "Breslow-Day Test for Homogeneity of the Odds Ratios" on page 3313.

BINOMIAL

BIN

includes the following binomial statistics for one-way tables: number of observations (total frequency), number of successes (frequency of designated level), binomial proportion estimate, confidence limits, and tests. The BINOMIAL option in the TABLES statement requests computation of binomial statistics, which are available for one-way tables. For more information, see the section "Binomial Proportion" on page 3263.

CHISQ

includes the following chi-square tests and measures in the output data set for two-way tables: Pearson chi-square, likelihood ratio chi-square, Mantel-Haenszel chi-square, phi coefficient, contingency coefficient, and Cramér's V. For 2×2 tables, CHISQ also includes Fisher's exact test and the continuity-adjusted chi-square in the output data set. For more information, see the section "Chi-Square Tests and Statistics" on page 3248. For one-way tables, CHISQ includes the chi-square goodness-of-fit test in the output data set. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248. The CHISQ option in the TABLES statement requests computation of these statistics.

If you specify the CHISQ(WARN=OUTPUT) option in the TABLES statement, the CHISQ option also includes the variable WARN_PCHI in the output data set. This variable indicates the validity warning for the asymptotic Pearson chi-square test.

CMH

includes the following Cochran-Mantel-Haenszel statistics in the output data set: correlation, row mean scores (ANOVA), and general association. For 2×2 tables, the CMH option also includes the Mantel-Haenszel and logit estimates of the common odds ratio and relative risks. For multiway (stratified) 2×2 tables, the CMH option includes the Breslow-Day test for homogeneity of odds ratios. The CMH option in the TABLES statement requests computation of these statistics. For more information, see the section "Cochran-Mantel-Haenszel Statistics" on page 3307.

If you specify the CMH(MANTELFLEISS) option in the TABLES statement, the CMH option includes the Mantel-Fleiss analysis in the output data set. The variables MF_CMH and WARN_CMH contain the Mantel-Fleiss criterion and the warning indicator, respectively.

CMH1

includes the CMH statistics in the output data set, with the exception of the row mean scores (ANOVA) statistic and the general association statistic. The CMH1 option in the TABLES statement requests computation of these statistics. For more information, see the section "Cochran-Mantel-Haenszel Statistics" on page 3307.

CMH2

includes the CMH statistics in the output data set, with the exception of the general association statistic. The CMH2 option in the TABLES statement requests computation of these statistics. For more information, see the section "Cochran-Mantel-Haenszel Statistics" on page 3307.

CMHCOR

includes the Cochran-Mantel-Haenszel correlation statistic in the output data set. The CMH option in the TABLES statement requests computation of this statistic. For more information, see the section "Correlation Statistic" on page 3309.

CMHGA

includes the Cochran-Mantel-Haenszel general association statistic in the output data set. The CMH option in the TABLES statement requests computation of this statistic. For more information, see the section "General Association Statistic" on page 3310.

CMHRMS

includes the Cochran-Mantel-Haenszel row mean scores (ANOVA) statistic in the output data set. The CMH option in the TABLES statement requests computation of this statistic. For more information, see the section "ANOVA (Row Mean Scores) Statistic" on page 3309.

COCHQ

includes Cochran's Q test in the output data set. Both the AGREE option and the CQ option in the TABLES statement request computation of this test, which is available for multiway tables in which each variable has two levels ($2 \times 2 \times \cdots \times 2$ tables). For more information, see the section "Cochran's Q Test" on page 3307.

CONTGY

includes the contingency coefficient in the output data set. The CHISQ option in the TABLES statement requests computation of the contingency coefficient. For more information, see the section "Contingency Coefficient" on page 3253.

CRAMV

includes Cramér's V in the output data set. The CHISQ option in the TABLES statement requests computation of Cramér's V. For more information, see the section "Cramér's V" on page 3253.

EQKAP

includes the test for equal simple kappa coefficients in the output data set. The AGREE option in the TABLES statement requests computation of this test, which is available for multiway, square ($h \times r \times r$) tables. For more information, see the section "Tests for Equal Kappa Coefficients" on page 3306.

EQOR

ZELEN

includes Zelen's exact test for equal odds ratios in the output data set. The EQOR option in the EXACT statement requests computation of this test, which is available for multiway 2×2 tables. For more information, see the section "Zelen's Exact Test for Equal Odds Ratios" on page 3314.

EQWKP

includes the test for equal weighted kappa coefficients in the output data set. The AGREE option in the TABLES statement requests computation of this test. The test for equal weighted kappas is available for multiway, square ($h \times r \times r$) tables where $r > 2$. For more information, see the section "Tests for Equal Kappa Coefficients" on page 3306.

FISHER

includes Fisher's exact test in the output data set. For 2×2 tables, the CHISQ option in the TABLES statement provides Fisher's exact test. For tables larger than 2×2, the FISHER option in the EXACT statement provides Fisher's exact test. For more information, see the section "Fisher's Exact Test" on page 3252.

GAILSIMON

GS

includes the Gail-Simon test for qualitative interaction in the output data set. The GAILSIMON option in the TABLES statement requests computation of this test. For more information, see the section "Gail-Simon Test for Qualitative Interactions" on page 3317.

GAMMA

includes the gamma statistic in the output data set. The MEASURES option in the TABLES statement requests computation of the gamma statistic. For more information, see the section "Gamma" on page 3255.

JT

includes the Jonckheere-Terpstra test in the output data set. The JT option in the TABLES statement requests the Jonckheere-Terpstra test. For more information, see the section "Jonckheere-Terpstra Test" on page 3298.

KAPPA

includes the simple kappa coefficient in the output data set. The AGREE option in the TABLES statement requests computation of kappa, which is available for square tables (where the number of rows equals the number of columns). For multiway square tables, the KAPPA option also includes the overall kappa coefficient in the output data set. For more information, see the sections "Simple Kappa Coefficient" on page 3301 and "Overall Kappa Coefficient" on page 3306.

KENTB

TAUB

includes Kendall's tau-*b* in the output data set. The MEASURES option in the TABLES statement requests computation of Kendall's tau-*b*. For more information, see the section "Kendall's Tau-b" on page 3255.

LAMCR

includes the asymmetric lambda $\lambda(C|R)$ in the output data set. The MEASURES option in the TABLES statement requests computation of lambda. For more information, see the section "Lambda (Asymmetric)" on page 3261.

LAMDAS

includes the symmetric lambda in the output data set. The MEASURES option in the TABLES statement requests computation of lambda. For more information, see the section "Lambda (Symmetric)" on page 3261.

LAMRC

includes the asymmetric lambda $\lambda(R|C)$ in the output data set. The MEASURES option in the TABLES statement requests computation of lambda. For more information, see the section "Lambda (Asymmetric)" on page 3261.

LGOR

includes the logit estimate of the common odds ratio in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for 2×2 tables. For more information, see the section "Adjusted Odds Ratio and Relative Risk Estimates" on page 3311.

LGRRC1

includes the logit estimate of the common relative risk (column 1) in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for 2×2 tables. For more information, see the section "Adjusted Odds Ratio and Relative Risk Estimates" on page 3311.

LGRRC2

includes the logit estimate of the common relative risk (column 2) in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for 2×2 tables. For more information, see the section "Adjusted Odds Ratio and Relative Risk Estimates" on page 3311.

LRCHI

includes the likelihood ratio chi-square in the output data set. The CHISQ option in the TABLES statement requests computation of the likelihood ratio chi-square. For more information, see the section "Likelihood Ratio Chi-Square Test" on page 3251.

MCNEM

includes McNemar's test (for 2×2 tables) in the output data set. The AGREE option in the TABLES statement requests computation of McNemar's test. For more information, see the section "McNemar's Test" on page 3300.

MEASURES

includes the following measures of association in the output data set: gamma, Kendall's tau-*b*, Stuart's tau-*c*, Somers' $D(C|R)$, Somers' $D(R|C)$, Pearson and Spearman correlation coefficients, lambda (symmetric and asymmetric), and uncertainty coefficients (symmetric and asymmetric). For 2×2 tables, the MEASURES option also includes the odds ratio, column 1 relative risk, and column 2 relative risk. The MEASURES option in the TABLES statement requests computation of these statistics. For more information, see the section "Measures of Association" on page 3253.

MHCHI

includes the Mantel-Haenszel chi-square in the output data set. The CHISQ option in the TABLES statement requests computation of the Mantel-Haenszel chi-square. For more information, see the section "Mantel-Haenszel Chi-Square Test" on page 3251.

MHOR

COMOR

includes the Mantel-Haenszel estimate of the common odds ratio in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for 2×2 tables. For more information, see the section "Adjusted Odds Ratio and Relative Risk Estimates" on page 3311.

MHRRC1

includes the Mantel-Haenszel estimate of the common relative risk (column 1) in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for 2×2 tables. For more information, see the section "Adjusted Odds Ratio and Relative Risk Estimates" on page 3311.

MHRRC2

includes the Mantel-Haenszel estimate of the common relative risk (column 2) in the output data set. The CMH option in the TABLES statement requests computation of this statistic, which is available for 2×2 tables. For more information, see the section "Adjusted Odds Ratio and Relative Risk Estimates" on page 3311.

N

includes the number of nonmissing observations (total frequency) for the table request.

NMISS

includes the number (frequency) of missing observations for the table request. For more information, see the section "Missing Values" on page 3243.

OR

ODDSRATIO

RROR

includes the odds ratio (for 2×2 tables) in the output data set. The MEASURES, OR, and RELRISK options in the TABLES statement request this statistic. For more information, see the section "Odds Ratio" on page 3287.

PCHI

includes the Pearson chi-square in the output data set for two-way tables. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249. For one-way tables, the PCHI option includes the chi-square goodness-of-fit test in the output data set. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248. The CHISQ option in the TABLES statement requests computation of these statistics.

If you specify the CHISQ(WARN=OUTPUT) option in the TABLES statement, the PCHI option also includes the variable WARN_PCHI in the output data set. This variable indicates the validity warning for the asymptotic Pearson chi-square test.

PCORR

includes the Pearson correlation coefficient in the output data set. The MEASURES option in the TABLES statement requests computation of the Pearson correlation. For more information, see the section "Pearson Correlation Coefficient" on page 3257.

PHI

includes the phi coefficient in the output data set. The CHISQ option in the TABLES statement requests computation of the phi coefficient. For more information, see the section "Phi Coefficient" on page 3253.

PLCORR

includes the polychoric correlation coefficient in the output data set. For 2×2 tables, this statistic is known as the tetrachoric correlation coefficient. The PLCORR option in the TABLES statement requests computation of the polychoric correlation. For more information, see the section "Polychoric Correlation" on page 3260.

RDIF1

includes the column 1 risk difference (row 1 – row 2) in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for 2×2 tables. For more information, see the section "Risks and Risk Differences" on page 3272.

RDIF2

includes the column 2 risk difference (row 1 – row 2) in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for 2×2 tables. For more information, see the section "Risks and Risk Differences" on page 3272.

RELRISK

includes the column 1 and column 2 relative risks (for 2×2 tables) in the output data set. The MEASURES and RELRISK options in the TABLES statement request these statistics. For more information, see the section "Relative Risks" on page 3290.

RISKDIFF

includes risks (binomial proportions) and risk differences for 2×2 tables in the output data set. These statistics include the row 1 risk, row 2 risk, total (overall) risk, and risk difference (row 1 – row 2) for column 1 and column 2. The RISKDIFF option in the TABLES statement requests computation of these statistics. For more information, see the section "Risks and Risk Differences" on page 3272.

RISKDIFF1

includes column 1 risks (binomial proportions) and risk differences for 2×2 tables in the output data set. These statistics include the row 1 risk, row 2 risk, total (overall) risk, and risk difference (row 1 – row 2). The RISKDIFF option in the TABLES statement requests computation of these statistics. For more information, see the section "Risks and Risk Differences" on page 3272.

RISKDIFF2

includes column 2 risks (binomial proportions) and risk differences for 2×2 tables in the output data set. These statistics include the row 1 risk, row 2 risk, total (overall) risk, and risk difference (row 1 – row 2). The RISKDIFF option in the TABLES statement requests computation of these statistics. For more information, see the section "Risks and Risk Differences" on page 3272.

RRC1

RELRISK1

includes the column 1 relative risk in the output data set. The MEASURES and RELRISK options in the TABLES statement request relative risks, which are available for 2×2 tables. For more information, see the section "Odds Ratio and Relative Risks" on page 3287.

RRC2

RELRISK2

includes the column 2 relative risk in the output data set. The MEASURES and RELRISK options in the TABLES statement request relative risks, which are available for 2×2 tables. For more information, see the section "Odds Ratio and Relative Risks" on page 3287.

RSK1

RISK1

includes the overall column 1 risk in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for 2×2 tables. For more information, see the section "Risks and Risk Differences" on page 3272.

RSK11

RISK11

includes the column 1 risk for row 1 in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for 2×2 tables. For more information, see the section "Risks and Risk Differences" on page 3272.

RSK12

RISK12

includes the column 2 risk for row 1 in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for 2×2 tables. For more information, see the section "Risks and Risk Differences" on page 3272.

RSK2

RISK2

includes the overall column 2 risk in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences. For more information, see the section "Risks and Risk Differences" on page 3272.

RSK21

RISK21

includes the column 1 risk for row 2 in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for 2×2 tables. For more information, see the section "Risks and Risk Differences" on page 3272.

RSK22

RISK22

includes the column 2 risk for row 2 in the output data set. The RISKDIFF option in the TABLES statement requests computation of risks and risk differences, which are available for 2×2 tables. For more information, see the section "Risks and Risk Differences" on page 3272.

SCORR

includes the Spearman correlation coefficient in the output data set. The MEASURES option in the TABLES statement requests computation of the Spearman correlation. For more information, see the section "Spearman Rank Correlation Coefficient" on page 3258.

SMDCR

includes Somers' $D(C|R)$ in the output data set. The MEASURES option in the TABLES statement requests computation of Somers' D. For more information, see the section "Somers' D" on page 3257.

SMDRC

includes Somers' $D(R|C)$ in the output data set. The MEASURES option in the TABLES statement requests computation of Somers' D. For more information, see the section "Somers' D" on page 3257.

STUTC

TAUC

includes Stuart's tau-c in the output data set. The MEASURES option in the TABLES statement requests computation of tau-c. For more information, see the section "Stuart's Tau-c" on page 3256.

TREND

includes the Cochran-Armitage test for trend in the output data set. The TREND option in the TABLES statement requests computation of the trend test. This test is available for tables of dimension $2 \times C$ or $R \times 2$. For more information, see the section "Cochran-Armitage Test for Trend" on page 3297.

TSYMM

BOWKER

includes Bowker's symmetry test in the output data set. The AGREE option in the TABLES statement requests computation of Bowker's test. For more information, see the section "Bowker's Symmetry Test" on page 3301.

U

includes the uncertainty coefficient (symmetric) in the output data set. The MEASURES option in the TABLES statement requests computation of the uncertainty coefficient. For more information, see the section "Uncertainty Coefficient (Symmetric)" on page 3263.

UCR

includes the asymmetric uncertainty coefficient $U(C|R)$ in the output data set. The MEASURES option in the TABLES statement requests computation of the uncertainty coefficient. For more information, see the section "Uncertainty Coefficients (Asymmetric)" on page 3262.

URC

includes the asymmetric uncertainty coefficient $U(R|C)$ in the output data set. The MEASURES option in the TABLES statement requests computation of the uncertainty coefficient. For more information, see the section "Uncertainty Coefficients (Asymmetric)" on page 3262.

WTKAPPA

WTKAP

includes the weighted kappa coefficient in the output data set. The AGREE option in the TABLES statement requests computation of weighted kappa, which is available for square tables larger than 2×2. For multiway tables, the WTKAPPA option also includes the overall weighted kappa coefficient in the output data set. For more information, see the sections "Weighted Kappa Coefficient" on page 3303 and "Overall Kappa Coefficient" on page 3306.

TABLES Statement

TABLES *requests* < / *options* > **;**

The TABLES statement requests one-way to n-way frequency and crosstabulation tables and statistics for those tables.

If you omit the TABLES statement, PROC FREQ generates one-way frequency tables for all data set variables that are not listed in the other statements.

The following argument is required in the TABLES statement.

requests

specify the frequency and crosstabulation tables to produce. A request is composed of one variable name or several variable names separated by asterisks. To request a one-way frequency table, use a single variable. To request a two-way crosstabulation table, use an asterisk between two variables. To request a multiway table (an n-way table, where $n > 2$), separate the variables with asterisks. The unique values of these variables form the rows, columns, and strata of the table. You can include up to 50 variables in a single multiway table request.

For two-way to multiway tables, the values of the last variable form the crosstabulation table columns, and the values of the next-to-last variable form the rows. Each level (or combination of levels) of the other variables forms one stratum. PROC FREQ produces a separate crosstabulation table for each stratum. For example, a specification of A*B*C*D in a TABLES statement produces k tables, where k is the number of different combinations of values for A and B. Each table lists the values for C down the side and the values for D across the top.

You can use multiple TABLES statements in the PROC FREQ step. PROC FREQ builds all the table requests in one pass of the data, so that there is essentially no loss of efficiency. You can also specify any number of table requests in a single TABLES statement. To specify multiple table requests quickly, use a grouping syntax by placing parentheses around several variables and joining other variables or variable combinations. For example, the statements shown in Table 47.8 illustrate grouping syntax.

Table 47.8 Grouping Syntax

TABLES Request	Equivalent to
A*(B C)	A*B A*C
(A B)*(C D)	A*C B*C A*D B*D
(A B C)*D	A*D B*D C*D
A – – C	A B C
(A – – C)*D	A*D B*D C*D

The TABLES statement variables are one or more variables from the DATA= input data set. These variables can be either character or numeric, but the procedure treats them as categorical variables. PROC FREQ uses the formatted values of the TABLES variable to determine the categorical variable levels. So if you assign a format to a variable with a FORMAT statement, PROC FREQ formats the values before dividing observations into the levels of a frequency or crosstabulation table. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

If you use PROC FORMAT to create a user-written format that combines missing and nonmissing values into one category, PROC FREQ treats the entire category of formatted values as missing. See the discussion in the section "Grouping with Formats" on page 3242 for more information.

By default, the frequency or crosstabulation table lists the values of both character and numeric variables in ascending order based on internal (unformatted) variable values. You can change the order of the values in the table by specifying the ORDER= option in the PROC FREQ statement. To list the values in ascending order by formatted value, use ORDER=FORMATTED.

Without Options

If you request a one-way frequency table for a variable without specifying options, PROC FREQ produces frequencies, cumulative frequencies, percentages of the total frequency, and cumulative percentages for each value of the variable. If you request a two-way or an *n*-way crosstabulation table without specifying any options, PROC FREQ produces crosstabulation tables that include cell frequencies, cell percentages of the total frequency, cell percentages of row frequencies, and cell percentages of column frequencies. The procedure excludes observations with missing values from the table but displays the total frequency of missing observations following each table.

Options

Table 47.9 lists the *options* available in the TABLES statement. Descriptions of the *options* follow in alphabetical order.

Table 47.9 TABLES Statement Options

Option	Description
Control Statistical Analysis	
AGREE	Requests tests and measures of classification agreement
ALL	Requests tests and measures of association produced by the CHISQ, MEASURES, and CMH options
ALPHA=	Sets confidence level for confidence limits
BINOMIAL | BIN	Requests binomial proportions, confidence limits, and tests for one-way tables
CHISQ	Requests chi-square tests and measures based on chi-square
CL	Requests confidence limits for MEASURES statistics
CMH	Requests all Cochran-Mantel-Haenszel statistics
CMH1	Requests CMH correlation statistic, adjusted odds ratios, and adjusted relative risks
CMH2	Requests CMH correlation and row mean scores (ANOVA) statistics, adjusted odds ratios, and adjusted relative risks
COMMONRISKDIFF	Requests common risk difference for $h \times 2 \times 2$ tables
CQ	Requests Cochran's Q test
FISHER	Requests Fisher's exact test for tables larger than 2×2
GAILSIMON	Requests Gail-Simon test for qualitative interactions
JT	Requests Jonckheere-Terpstra test
MEASURES	Requests measures of association
MISSING	Treats missing values as nonmissing
OR	Requests the odds ratio for 2×2 tables
PLCORR	Requests polychoric correlation
RELRISK	Requests relative risks for 2×2 tables
RISKDIFF	Requests risks and risk differences for 2×2 tables
SCORES=	Specifies type of row and column scores
SENSPEC	Requests sensitivity and specificity for 2×2 tables
TREND	Requests Cochran-Armitage test for trend
Control Additional Table Information	
CELLCHI2	Displays cell contributions to the Pearson chi-square statistic

Table 47.9 *continued*

Option	Description
CUMCOL	Displays cumulative column percentages
DEVIATION	Displays deviations of cell frequencies from expected values
EXPECTED	Displays expected cell frequencies
MISSPRINT	Displays missing value frequencies
ONEWAY	Specifies content of one-way frequency tables
PRINTKWTS	Displays kappa coefficient weights
SCOROUT	Displays row and column scores
SPARSE	Includes all possible combinations of variable levels in the LIST table and OUT= data set
TOTPCT	Displays percentages of total frequency for n-way tables ($n > 2$)
Control Displayed Output	
CROSSLIST	Displays crosstabulation tables in ODS column format
FORMAT=	Formats frequencies in crosstabulation tables
FORMATP=	Formats percentages in crosstabulation tables
LIST	Displays two-way to n-way tables in list format
MAXLEVELS=	Specifies maximum number of levels to display in one-way tables
NOCOL	Suppresses display of column percentages
NOCUM	Suppresses display of cumulative frequencies and percentages
NOFREQ	Suppresses display of frequencies
NOPERCENT	Suppresses display of percentages
NOPRINT	Suppresses display of crosstabulation tables but displays statistics
NOROW	Suppresses display of row percentages
NOSPARSE	Suppresses zero-frequency levels in the CROSSLIST table, LIST table, and OUT= data set
NOWARN	Suppresses log warning message for the chi-square test
Produce Statistical Graphics	
PLOTS=	Requests plots from ODS Graphics
Create an Output Data Set	
OUT=	Names an output data set to contain frequency counts
OUTCUM	Includes cumulative frequencies and percentages in the output data set for one-way tables
OUTEXPECT	Includes expected frequencies in the output data set
OUTPCT	Includes row, column, and two-way table percentages in the output data set

You can specify the following *options*:

AGREE < (*agree-options*) >

requests tests and measures of classification agreement for square tables. This option provides the simple and weighted kappa coefficients along with their standard errors and confidence limits. For multiway tables, this option also produces the overall simple and weighted kappa coefficients (along with their standard errors and confidence limits) and tests for equal kappas among strata. For 2×2

tables, this option provides McNemar's test; for square tables that have more than two response categories (levels), this option provides Bowker's symmetry test. For multiway tables that have two response categories, this option also produces Cochran's Q test. (You can request Cochran's Q test separately by specifying the CQ option.) For more information, see the section "Tests and Measures of Agreement" on page 3300.

Measures of agreement can be computed only for square tables, where the number of rows equals the number of columns. If your table is not square because some observations have weights of 0, you can specify the ZEROS option in the WEIGHT statement to include these observations in the analysis. For more information, see the section "Tables with Zero-Weight Rows or Columns" on page 3307.

For 2×2 tables, the weighted kappa coefficient is equivalent to the simple kappa coefficient, and PROC FREQ displays only analyses for the simple kappa coefficient.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

You can specify the EXACT statement to request McNemar's exact test (for 2×2 tables), an exact symmetry test, and exact tests for the simple and weighted kappa coefficients. For more information, see the section "Exact Statistics" on page 3318.

You can specify the following *agree-options*:

AC1

requests the AC1 agreement coefficient. For more information, see the section "AC1 Agreement Coefficient" on page 3306.

DFSYM=*df* | ADJUST

controls the degrees of freedom for Bowker's symmetry test. You can specify the value of the degrees of freedom (*df*), or you can specify DFSYM=ADJUST to adjust the degrees of freedom for empty table cells. The value of *df* must be a positive number. By default, *df* is $R(R-1)/2$, where R is the dimension of the two-way table.

When you specify DFSYM=ADJUST, the degrees of freedom are reduced by the number of off-diagonal table-cell pairs that have a total frequency of 0. By default, the degrees of freedom count all off-diagonal table-cell pairs. For more information, see the section "Bowker's Symmetry Test" on page 3301.

KAPPADETAILS

DETAILS

displays the "Kappa Details" table, which includes the following statistics for the simple kappa coefficient: observed agreement, chance-expected agreement, maximum kappa, and the B_n measure. If the two-way table is 2×2, the "Kappa Details" table also includes the prevalence index and the bias index. For more information, see the section "Simple Kappa Coefficient" on page 3301.

If the two-way table is larger than 2×2, this option also displays the "Weighted Kappa Details" table, which includes the observed agreement and chance-expected agreement components of the weighted kappa coefficient. For more information, see the section "Weighted Kappa Coefficient" on page 3303.

MNULLRATIO=*value*

specifies the null *value* of the ratio of discordant proportions for McNemar's test. By default, MNULLRATIO=1. For more information, see the section "McNemar's Test" on page 3300.

NULLKAPPA=*value*

requests the simple kappa coefficient test and specifies the null *value* for the test. The null value must be between –1 and 1. By default, NULLKAPPA=0. For more information, see the section "Simple Kappa Coefficient" on page 3301.

This option is not available when you specify the KAPPA option in the EXACT statement, which requests an exact test for the kappa coefficient.

NULLWTKAPPA=*value*

requests the weighted kappa coefficient test and specifies the null *value* for the test. The null value must be between –1 and 1. By default, NULLWTKAPPA=0. For more information, see the section "Weighted Kappa Coefficient" on page 3303.

This option is not available when you specify the WTKAPPA option in the EXACT statement, which requests an exact test for the weighted kappa coefficient.

PABAK

requests the prevalence-adjusted bias-adjusted kappa coefficient. For more information, see the section "Prevalence-Adjusted Bias-Adjusted Kappa" on page 3305.

PRINTKWTS

displays the agreement weights that PROC FREQ uses to compute the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. By default, PROC FREQ uses Cicchetti-Allison agreement weights. If you specify the WT=FC option, the procedure uses Fleiss-Cohen agreement weights. For more information, see the section "Weighted Kappa Coefficient" on page 3303.

TABLES=RESTORE

displays the following agreement tables (which are produced by the AGREE option) in legacy, factoid (label-value) format: "McNemar's Test," "Symmetry Test," "Monte Carlo Exact Symmetry Test," "Simple Kappa Coefficient," "Simple Kappa Test," "Weighted Kappa Coefficient," "Weighted Kappa Test," "Monte Carlo Estimates for the Exact Test," "Overall Kappa Coefficient," "Tests for Equal Kappa Coefficients," and "Cochran's Q."

By default, PROC FREQ displays all agreement tables in tabular form.

WT=FC

specifies Fleiss-Cohen agreement weights in the computation of the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. By default, PROC FREQ uses Cicchetti-Allison agreement weights to compute the weighted kappa coefficient. For more information, see the section "Weighted Kappa Coefficient" on page 3303.

WTKAPPADETAILS

displays the "Weighted Kappa Details" table, which includes the observed agreement and chance-expected agreement components of the weighted kappa coefficient. This information is available for two-way tables that are larger than 2×2. For more information, see the section "Weighted Kappa Coefficient" on page 3303.

ALL

requests all tests and measures that are produced by the CHISQ, MEASURES, and CMH options. You can control the number of CMH statistics to compute by specifying the CMH1 or CMH2 option.

ALPHA=α

specifies the level of confidence limits. The value of α must be between 0 and 1; a confidence level of α produces $100(1-\alpha)\%$ confidence limits. By default ALPHA=0.05, which produces 95% confidence limits.

This option applies to confidence limits that you request in the TABLES statement. The ALPHA= option in the EXACT statement applies to confidence limits for Monte Carlo estimates of exact p-values, which you request by specifying the MC option in the EXACT statement.

BINOMIAL < (*binomial-options*) >

BIN < (*binomial-options*) >

requests the binomial proportion for one-way tables. By default, PROC FREQ provides the asymptotic standard error, asymptotic Wald and exact (Clopper-Pearson) confidence limits, and the asymptotic equality test for the binomial proportion.

You can specify *binomial-options* in parentheses after the BINOMIAL option. The LEVEL= *binomial-option* identifies the variable level for which to compute the proportion. If you do not specify this option, PROC FREQ computes the proportion for the first level that appears in the one-way frequency table. The P= *binomial-option* specifies the null proportion for the binomial tests. If you do not specify this option, PROC FREQ uses 0.5 as the null proportion for the binomial tests.

You can also specify *binomial-options* to request additional tests and confidence limits for the binomial proportion. The EQUIV, NONINF, and SUP *binomial-options* request tests of equivalence, noninferiority, and superiority, respectively. The CL= *binomial-option* requests confidence limits for the binomial proportion.

You can specify the level for the binomial confidence limits in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits. As part of the noninferiority, superiority, and equivalence analyses, PROC FREQ provides null-based equivalence limits that have a confidence coefficient of $100(1-2\alpha)\%$ (Schuirmann 1999). In these analyses, the default of ALPHA=0.05 produces 90% equivalence limits. For more information, see the sections "Noninferiority Test" on page 3268 and "Equivalence Test" on page 3270.

To request exact tests for the binomial proportion, you can specify the BINOMIAL option in the EXACT statement. PROC FREQ computes exact p-values for all binomial tests that you request, which can include noninferiority, superiority, and equivalence tests, in addition to the equality test that the BINOMIAL option produces by default.

For more information, see the section "Binomial Proportion" on page 3263.

Table 47.10 summarizes the *binomial-options*.

Table 47.10 BINOMIAL Options

Option	Description
CORRECT	Requests continuity correction
LEVEL=	Specifies the variable level

Table 47.10 *continued*

Option	Description
OUTLEVEL	Includes the level in the output data sets
P=	Specifies the null proportion
Request Confidence Limits	
CL=AGRESTICOULL | AC	Requests Agresti-Coull confidence limits
CL=BLAKER	Requests Blaker confidence limits
CL=EXACT | CLOPPERPEARSON	Requests exact (Clopper-Pearson) confidence limits
CL=JEFFREYS	Requests Jeffreys confidence limits
CL=LIKELIHOODRATIO | LR	Requests likelihood ratio confidence limits
CL=LOGIT	Requests logit confidence limits
CL=MIDP	Requests exact mid-*p* confidence limits
CL=WALD	Requests Wald confidence limits
CL=WILSON | SCORE	Requests Wilson (score) confidence limits
Request Tests	
EQUIV | EQUIVALENCE	Requests an equivalence test
MARGIN=	Specifies the test margin
NONINF | NONINFERIORITY	Requests a noninferiority test
SUP | SUPERIORITY	Requests a superiority test
VAR=NULL | SAMPLE	Specifies the test variance

You can specify the following *binomial-options*:

CL=*type* **|** (*types*)

requests confidence limits for the binomial proportion. You can specify one or more *types* of confidence limits. When you specify only one *type*, you can omit the parentheses around the request. PROC FREQ displays the confidence limits in the "Binomial Confidence Limits" table.

The ALPHA= option determines the level of the confidence limits that the CL= *binomial-option* provides. By default, ALPHA=0.05, which produces 95% confidence limits for the binomial proportion.

You can specify the CL= *binomial-option* with or without requests for binomial tests. The confidence limits that CL= produces do not depend on the tests that you request and do not use the value of the test margin (which you can specify in the MARGIN= *binomial-option*).

If you do not specify the CL= *binomial-option*, the BINOMIAL option displays Wald and exact (Clopper-Pearson) confidence limits in the "Binomial Proportion" table.

You can specify the following *types*:

AGRESTICOULL

AC

requests Agresti-Coull confidence limits for the binomial proportion. For more information, see the section "Agresti-Coull Confidence Limits" on page 3264.

BLAKER

requests Blaker confidence limits for the binomial proportion. For more information, see the section "Blaker Confidence Limits" on page 3265.

EXACT

CLOPPERPEARSON

requests exact (Clopper-Pearson) confidence limits for the binomial proportion. For more information, see the section "Exact (Clopper-Pearson) Confidence Limits" on page 3264.

If you do not specify the CL= *binomial-option*, PROC FREQ displays Wald and exact (Clopper-Pearson) confidence limits in the "Binomial Proportion" table. To request exact tests for the binomial proportion, you can specify the BINOMIAL option in the EXACT statement.

JEFFREYS < (*option*) >

requests Jeffreys confidence limits for the binomial proportion. For more information, see the section "Jeffreys Confidence Limits" on page 3265.

You can specify the following *option*:

MODIFY

provides modified Jeffreys confidence limits (Brown, Cai, and DasGupta 2001) to improve the coverage of the confidence limits for proportions near 0 or 1. For more information, see the section "Jeffreys Confidence Limits" on page 3265.

LIKELIHOODRATIO

LR

requests likelihood ratio confidence limits for the binomial proportion. For more information, see the section "Likelihood Ratio Confidence Limits" on page 3265.

LOGIT

requests logit confidence limits for the binomial proportion. For more information, see the section "Logit Confidence Limits" on page 3266.

MIDP

requests exact mid-*p* confidence limits for the binomial proportion. For more information, see the section "Mid-*p* Confidence Limits" on page 3266.

WALD < (CORRECT) >

requests Wald confidence limits for the binomial proportion. For more information, see the section "Wald Confidence Limits" on page 3263.

If you specify CL=WALD(CORRECT), the Wald confidence limits include a continuity correction. If you specify the CORRECT *binomial-option*, both the Wald confidence limits and the Wald tests include continuity corrections.

If you do not specify the CL= *binomial-option*, PROC FREQ displays Wald and exact (Clopper-Pearson) confidence limits in the "Binomial Proportion" table.

WILSON < (*option*) >

SCORE < (*option*) >

requests Wilson confidence limits for the binomial proportion. These are also known as *score* confidence limits. For more information, see the section "Wilson (Score) Confidence Limits" on page 3266.

You can specify one of the following *options*:

ADAPT

provides adapted Wilson confidence limits (Agresti and Coull 1998) to improve the coverage of the confidence limits for proportions near 0 or 1. Adapted confidence limits are computed when the number of successes is 1 or $n - 1$. For more information, see the section "Wilson (Score) Confidence Limits" on page 3266.

CORRECT

includes a continuity correction in the Wilson confidence limits. For more information, see the section "Wilson (Score) Confidence Limits" on page 3266.

MODIFY

provides modified Wilson confidence limits (Brown, Cai, and DasGupta 2001) to improve the coverage of the confidence limits for proportions near 0 or 1. For more information, see the section "Wilson (Score) Confidence Limits" on page 3266.

CORRECT

includes a continuity correction in the Wald confidence limits, Wald tests, and Wilson confidence limits.

You can request continuity corrections individually for Wald or Wilson confidence limits by specifying the CL=WALD(CORRECT) or CL=WILSON(CORRECT) *binomial-option*, respectively.

EQUIV

EQUIVALENCE

requests a test of equivalence for the binomial proportion. For more information, see the section "Equivalence Test" on page 3270. You can specify the equivalence test margins, the null proportion, and the variance type in the MARGIN=, P=, and VAR= *binomial-options*, respectively. To request an exact equivalence test, you can specify the BINOMIAL option in the EXACT statement.

LEVEL=*level-number* | '*level-value*'

specifies the variable level for the binomial proportion. You can specify the *level-number*, which is the order in which the level appears in the one-way frequency table. Or you can specify the *level-value*, which is the formatted value of the variable level. The *level-number* must be a positive integer. You must enclose the *level-value* in single quotes.

By default, PROC FREQ computes the binomial proportion for the first variable level that appears in the one-way frequency table.

MARGIN=*value* | **(***lower***,** *upper***)**

specifies the margin for the noninferiority, superiority, and equivalence tests, which you can request by specifying the NONINF, SUP, and EQUIV *binomial-options*, respectively. By default, MARGIN=0.2.

For noninferiority and superiority tests, specify a single *value* in the MARGIN= option. The MARGIN= *value* must be a positive number. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC FREQ converts that number to a proportion. PROC FREQ treats the value 1 as 1%.

For noninferiority and superiority tests, the test limits must be between 0 and 1. The limits are determined by the null proportion value (which you can specify in the P= *binomial-option*) and by the margin value. The noninferiority limit is the null proportion minus the margin. By default, the null proportion is 0.5 and the margin is 0.2, which produces a noninferiority limit of 0.3. The superiority limit is the null proportion plus the margin, which is 0.7 by default.

For an equivalence test, you can specify a single MARGIN= *value*, or you can specify both *lower* and *upper* values. If you specify a single MARGIN= *value*, it must be a positive number, as described previously. If you specify a single MARGIN= *value* for an equivalence test, PROC FREQ uses –*value* as the lower margin and *value* as the upper margin for the test. If you specify both *lower* and *upper* values for an equivalence test, you can specify them in proportion form as numbers between –1 and 1. Or you can specify them in percentage form as numbers between –100 and 100, and PROC FREQ converts the numbers to proportions. The value of *lower* must be less than the value of *upper*.

The equivalence limits must be between 0 and 1. The equivalence limits are determined by the null proportion value (which you can specify in the P= *binomial-option*) and by the margin values. The lower equivalence limit is the null proportion plus the lower margin. By default, the null proportion is 0.5 and the lower margin is –0.2, which produces a lower equivalence limit of 0.3. The upper equivalence limit is the null proportion plus the upper margin, which is 0.7 by default.

For more information, see the sections "Noninferiority Test" on page 3268 and "Equivalence Test" on page 3270.

NONINF

NONINFERIORITY

requests a test of noninferiority for the binomial proportion. For more information, see the section "Noninferiority Test" on page 3268. You can specify the noninferiority test margin, the null proportion, and the variance type in the MARGIN=, P=, and VAR= *binomial-options*, respectively. To request an exact noninferiority test, you can specify the BINOMIAL option in the EXACT statement.

OUTLEVEL

includes the variables LevelNumber and LevelValue in all ODS output data sets that PROC FREQ produces when you specify the BINOMIAL option in the TABLES statement. The OUTLEVEL option also includes the variables LevelNumber and LevelValue in the statistics output data set that PROC FREQ produces when you specify the BINOMIAL option in the OUTPUT statement.

The LevelNumber and LevelValue variables identify the analysis variable level for which PROC FREQ computes the binomial proportion. The value of LevelNumber is the order of the level in the one-way frequency table. The value of LevelValue is the formatted value of the level. You can specify the OUTLEVEL *binomial-option* with or without the LEVEL= *binomial-option*.

P=*value*

specifies the null hypothesis proportion for the binomial tests. The null proportion *value* must be a positive number. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form (as a number between 1 and 100), and PROC FREQ converts that number to a proportion. PROC FREQ treats the value 1 as 1%. By default, P=0.5.

SUP

SUPERIORITY

requests a test of superiority for the binomial proportion. For more information, see the section "Superiority Test" on page 3269. You can specify the superiority test margin, the null proportion, and the variance type in the MARGIN=, P=, and VAR= *binomial-options*, respectively. To request an exact superiority test, you can specify the BINOMIAL option in the EXACT statement.

VAR=NULL | SAMPLE

specifies the type of variance to use in the Wald tests of noninferiority, superiority, and equivalence. If you specify VAR=SAMPLE, PROC FREQ computes the variance estimate by using the sample proportion. If you specify VAR=NULL, PROC FREQ computes a test-based variance by using the null hypothesis proportion (which you can specify in the P= *binomial-option*). For more information, see the sections "Noninferiority Test" on page 3268 and "Equivalence Test" on page 3270. By default, VAR=SAMPLE.

CELLCHI2

displays cell chi-squares in the crosstabulation table. A cell chi-square is the table cell's contribution to the Pearson chi-square statistic. The cell chi-square is computed as $(frequency - expected)^2/expected$, where *frequency* is the table cell frequency (count) and *expected* is the expected cell frequency, which is computed under the null hypothesis that the row and column variables are independent. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249.

This option applies to two-way and multiway tables that are displayed in the default crosstabulation cell format or in CROSSLIST format. This option has no effect for one-way frequency tables or for tables that are produced by the LIST option. To display cell chi-squares in one-way frequency tables, you can specify the ONEWAY(CELLCHI2) option.

CHISQ <(*chisq-options***)>**

requests chi-square tests of homogeneity or independence and measures of association that are based on the chi-square statistic. For two-way tables, the chi-square tests include the Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests. The chi-square measures include the phi coefficient, contingency coefficient, and Cramér's V. For 2×2 tables, the CHISQ option also provides Fisher's exact test and the continuity-adjusted chi-square test. For more information, see the section "Chi-Square Tests and Statistics" on page 3248.

For one-way tables, the CHISQ option provides the Pearson chi-square goodness-of-fit test. You can also request the likelihood ratio goodness-of-fit test for one-way tables by specifying the LRCHI *chisq-option* in parentheses after the CHISQ option. By default, the one-way chi-square tests are based on the null hypothesis of equal proportions. Alternatively, you can provide null hypothesis proportions or frequencies by specifying the TESTP= or TESTF= *chisq-option*, respectively. See the section "Chi-Square Test for One-Way Tables" on page 3248 for more information.

To request Fisher's exact test for tables larger than 2×2, specify the FISHER option in the EXACT statement. Exact p-values are also available for the Pearson, likelihood ratio, and Mantel-Haenszel chi-square tests. See the description of the EXACT statement for more information.

You can specify the following *chisq-options*:

DF=*df*

specifies the degrees of freedom for the chi-square tests. The value of *df* must not be 0. If the value of *df* is positive, PROC FREQ uses *df* as the degrees of freedom for the chi-square tests. If the value of *df* is negative, PROC FREQ uses *df* to adjust the default degrees of freedom for the chi-square tests.

By default for one-way tables, the value of *df* is $(n - 1)$, where n is the number of variable levels in the table. By default for two-way tables, the value of *df* is $(r - 1)(c - 1)$, where r is the number of rows in the table and c is the number of columns. See the sections "Chi-Square Test for One-Way Tables" on page 3248 and "Chi-Square Tests and Statistics" on page 3248 for more information.

If you specify a negative value of *df*, PROC FREQ adjusts the default degrees of freedom by adding the (negative) value of *df* to the default value to produce the adjusted degrees of freedom. The adjusted degrees of freedom must be positive.

The DF= *chisq-option* specifies or adjusts the degrees of freedom for the following chi-square tests: the Pearson and likelihood ratio goodness-of-fit tests for one-way tables; and the Pearson, likelihood ratio, and Mantel-Haenszel chi-square tests for two-way tables.

LRCHI

requests the likelihood ratio goodness-of-fit test for one-way tables. See the section "Likelihood Ratio Chi-Square Test for One-Way Tables" on page 3250 for more information.

By default, this test is based on the null hypothesis of equal proportions. You can provide null hypothesis proportions or frequencies by specifying the TESTP= or TESTF= *chisq-option*, respectively. You can request an exact likelihood ratio goodness-of-fit test by specifying the LRCHI option in the EXACT statement.

TESTF=(*values***)|** *SAS-data-set*

specifies null hypothesis frequencies for the one-way chi-square goodness-of-fit tests. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248. You can list the null frequencies as *values* in parentheses after TESTF=. Or you can provide the null frequencies in a secondary input data set by specifying TESTF=*SAS-data-set*. The TESTF=*SAS-data-set* cannot be the same data set that you specify in the DATA= option. You can specify only one TESTF= or TESTP= data set in a single invocation of the procedure.

If you list the null frequencies as *values*, you can separate the *values* with blanks or commas. The *values* must be positive numbers. The number of *values* must equal the number of variable levels in the one-way table. The sum of the *values* must equal the total frequency for the one-way table. Order the *values* to match the order in which the corresponding variable levels appear in the one-way frequency table.

If you provide the null frequencies in a secondary input data set (TESTF=*SAS-data-set*), the variable that contains the null frequencies should be named _TESTF_, TestFrequency, or Frequency. The null frequencies must be positive numbers. The number of frequencies must equal the number of levels in the one-way frequency table, and the sum of the frequencies must equal the total frequency for the one-way table. Order the null frequencies in the data set to match the order in which the corresponding variable levels appear in the one-way frequency table.

TESTP=(*values***)**| *SAS-data-set*

specifies null hypothesis proportions for the one-way chi-square goodness-of-fit tests. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248. You can list the null proportions as *values* in parentheses after TESTP=. Or you can provide the null proportions in a secondary input data set by specifying TESTP=*SAS-data-set*. The TESTP=*SAS-data-set* cannot be the same data set that you specify in the DATA= option. You can specify only one TESTF= or TESTP= data set in a single invocation of the procedure.

If you list the null proportions as *values*, you can separate the *values* with blanks or commas. The *values* must be positive numbers. The number of *values* must equal the number of variable levels in the one-way table. Order the *values* to match the order in which the corresponding variable levels appear in the one-way frequency table. You can specify *values* in probability form as numbers between 0 and 1, where the proportions sum to 1. Or you can specify *values* in percentage form as numbers between 0 and 100, where the percentages sum to 100.

If you provide the null proportions in a secondary input data set (TESTP=*SAS-data-set*), the variable that contains the null proportions should be named _TESTP_, TestPercent, or Percent. The null proportions must be positive numbers. The number of proportions must equal the number of levels in the one-way frequency table. You can provide the proportions in probability form as numbers between 0 and 1, where the proportions sum to 1. Or you can provide the proportions in percentage form as numbers between 0 and 100, where the percentages sum to 100. Order the null proportions in the data set to match the order in which the corresponding variable levels appear in the one-way frequency table.

WARN=*type* | **(***types***)**

controls the warning message for the validity of the asymptotic Pearson chi-square test. By default, PROC FREQ displays a warning message when more than 20% of the table cells have expected frequencies that are less than 5. If you specify the NOPRINT option in the PROC FREQ statement, the procedure displays the warning in the log; otherwise, the procedure displays the warning as a footnote in the chi-square table. You can use the WARN= option to suppress the warning and to include a warning indicator in the output data set.

You can specify one or more of the following *types* in the WARN= option. If you specify more than one *type* value, enclose the values in parentheses after WARN=. For example, `warn = (output noprint)`.

Value of WARN=	Description
OUTPUT	Adds a warning indicator variable to the output data set
NOLOG	Suppresses the chi-square warning message in the log
NOPRINT	Suppresses the chi-square warning message in the display
NONE	Suppresses the chi-square warning message entirely

If you specify the WARN=OUTPUT option, the ODS output data set ChiSq contains a variable named Warning that equals 1 for the Pearson chi-square observation when more than 20% of the table cells have expected frequencies that are less than 5 and equals 0 otherwise. If you specify WARN=OUTPUT and also specify the CHISQ option in the OUTPUT statement, the statistics output data set contains a variable named WARN_PCHI that indicates the warning.

The WARN=NOLOG option has the same effect as the NOWARN option in the TABLES statement.

CL

requests confidence limits for the measures of association, which you can request by specifying the MEASURES option. For more information, see the sections "Measures of Association" on page 3253 and "Confidence Limits" on page 3254. You can set the level of the confidence limits by using the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

If you omit the MEASURES option, the CL option invokes MEASURES. The CL option is equivalent to the MEASURES(CL) option.

CMH < (*cmh-options*) >

requests Cochran-Mantel-Haenszel statistics, which test for association between the row and column variables after adjusting for the remaining variables in a multiway table. The Cochran-Mantel-Haenszel statistics include the nonzero correlation statistic, the row mean scores (ANOVA) statistic, and the general association statistic. In addition, for 2×2 tables, the CMH option provides the adjusted Mantel-Haenszel and logit estimates of the odds ratio and relative risks, together with their confidence limits. For stratified 2×2 tables, the CMH option provides the Breslow-Day test for homogeneity of odds ratios. (To request Tarone's adjustment for the Breslow-Day test, specify the BDT *cmh-option*.) For more information, see the section "Cochran-Mantel-Haenszel Statistics" on page 3307.

You can use the CMH1 or CMH2 option to control the number of CMH statistics that PROC FREQ computes.

For stratified 2×2 tables, you can request Zelen's exact test for equal odds ratios by specifying the EQOR option in the EXACT statement. For more information, see the section "Zelen's Exact Test for Equal Odds Ratios" on page 3314. You can request exact confidence limits for the common odds ratio by specifying the COMOR option in the EXACT statement. This option also provides a common odds ratio test. For more information, see the section "Exact Confidence Limits for the Common Odds Ratio" on page 3315.

You can specify the following *cmh-options* in parentheses after the CMH option. These *cmh-options*, which apply to stratified 2×2 tables, are also available with the CMH1 or CMH2 option.

BDT

requests Tarone's adjustment in the Breslow-Day test for homogeneity of odds ratios. For more information, see the section "Breslow-Day Test for Homogeneity of the Odds Ratios" on page 3313.

GAILSIMON < (COLUMN=1 | 2) >

GS < (COLUMN=1 | 2) >

requests the Gail-Simon test for qualitative interaction, which applies to stratified 2×2 tables. For more information, see the section "Gail-Simon Test for Qualitative Interactions" on page 3317.

The COLUMN= option specifies the column of the risk differences to use to compute the Gail-Simon test. By default, PROC FREQ uses column 1 risk differences. If you specify COLUMN=2, PROC FREQ uses column 2 risk differences.

The GAILSIMON *cmh-option* is equivalent to the GAILSIMON option in the TABLES statement.

I2

requests the I-square measure of heterogeneity for stratified 2×2 tables. I-square is computed from a Q test that is based on odds ratios. The I2 *cmh-option* invokes the QOR *cmh-option*. For more information, see the section "I-Square Measure of Heterogeneity" on page 3314.

MANTELFLEISS

MF

requests the Mantel-Fleiss criterion for the Mantel-Haenszel statistic for stratified 2×2 tables. For more information, see the section "Mantel-Fleiss Criterion" on page 3310.

QOR

requests a Q test for heterogeneity of odds ratios for stratified 2×2 tables. For more information, see the section "Q Test for Homogeneity of Odds Ratios" on page 3313.

CMH1 < (*cmh-options*) >

requests the Cochran-Mantel-Haenszel correlation statistic. This option does not provide the CMH row mean scores (ANOVA) statistic or the general association statistic, which are provided by the CMH option. For tables larger than 2×2, the CMH1 option requires less memory than the CMH option, which can require an enormous amount of memory for large tables.

For 2×2 tables, the CMH1 option also provides the adjusted Mantel-Haenszel and logit estimates of the odds ratio and relative risks, together with their confidence limits. For stratified 2×2 tables, the CMH1 option provides the Breslow-Day test for homogeneity of odds ratios.

The *cmh-options* for CMH1 are the same as the *cmh-options* that are available with the CMH option. For more information, see the description of the CMH option.

CMH2 < (*cmh-options*) >

requests the Cochran-Mantel-Haenszel correlation statistic and the row mean scores (ANOVA) statistic. This option does not provide the CMH general association statistic, which is provided by the CMH option. For tables larger than 2×2, the CMH2 option requires less memory than the CMH option, which can require an enormous amount of memory for large tables.

For 2×2 tables, the CMH1 option also provides the adjusted Mantel-Haenszel and logit estimates of the odds ratio and relative risks, together with their confidence limits. For stratified 2×2 tables, the CMH1 option provides the Breslow-Day test for homogeneity of odds ratios.

The *cmh-options* for CMH2 are the same as the *cmh-options* that are available with the CMH option. For more information, see the description of the CMH option.

COMMONRISKDIFF < *options* >

requests the common (stratified) risk difference for multiway 2×2 tables, where the risk difference is the difference between the row 1 proportion and the row 2 proportion in a 2×2 table. By default, this option provides Mantel-Haenszel and summary score estimates of the common risk difference, together with their confidence limits. For more information, see the section "Common Risk Difference" on page 3283.

You can specify the following *options* to request confidence limit types and tests for the common risk difference:

CL=*type* | (*types*)

requests confidence limits for the common risk difference. You can specify one or more *types* of confidence limits. When you specify only one *type*, you can omit the parentheses. You can specify CL=NONE to suppress the "Confidence Limits for the Common Risk Difference" table.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits for the common risk difference.

You can specify one or more of the following *types*:

K

KLINGENBERG

requests Klingenberg confidence limits for the Mantel-Haenszel common risk difference. For more information, see the section "Klingenberg Confidence Limits" on page 3284.

MH

requests Mantel-Haenszel confidence limits, which are computed by using Mantel-Haenszel stratum weights and the Sato variance estimator (Sato 1989). For more information, see the section "Mantel-Haenszel Confidence Limits and Test" on page 3283.

MN

requests Miettinen-Nurminen confidence limits for the common risk difference (Miettinen and Nurminen 1985).

MNMH

requests confidence limits that are computed by using the Miettinen-Nurminen method with Mantel-Haenszel weights (Lu 2008).

MR

MINRISK

requests minimum risk confidence limits, which are computed by using minimum risk weights. For more information, see the section "Minimum Risk Confidence Limits and Test" on page 3284.

NEWCOMBE

requests stratified Newcombe confidence limits that use Mantel-Haenszel weights to combine the stratum components. For more information, see the section "Stratified Newcombe Confidence Limits" on page 3286.

NEWCOMBEMR

requests stratified Newcombe confidence limits that use minimum risk weights to combine the stratum components. For more information, see the section "Stratified Newcombe Confidence Limits" on page 3286.

NONE

suppresses the "Confidence Limits for the Common Risk Difference" table.

SCORE

requests summary score confidence limits. For more information, see the section "Summary Score Confidence Limits" on page 3286.

COLUMN=1 | 2

specifies the table column for which to compute the common risk difference statistics.

By default, COLUMN=1 unless you specify the column for the risk difference plot or for the stratum-level risk difference statistics. In this case, when you omit the COMMONRISKDIFF(COLUMN=) option, PROC FREQ provides the common risk difference statistics for the same column that you specify for the plot or for the stratum-level statistics.

CORRECT=NO

removes the continuity correction in the minimum risk confidence limits and in the minimum risk test, which you can request by specifying the CL=MR and TEST=MR options, respectively. For more information, see the section "Minimum Risk Confidence Limits and Test" on page 3284.

PRINTWTS < =*type* | (*types*) >

displays the stratum weights together with the stratum risk differences and frequencies. By default, this option displays the weight type or types for the confidence limits and tests that you request. Optionally, you can specify the weight type to display.

You can specify one or more of the following *types*:

MH

displays Mantel-Haenszel stratum weights. For more information, see the section "Mantel-Haenszel Confidence Limits and Test" on page 3283.

MN

displays Miettinen-Nurminen stratum weights.

MR

displays minimum risk stratum weights. For more information, see the section "Minimum Risk Confidence Limits and Test" on page 3284.

SCORE

displays summary score stratum weights. For more information, see the section "Summary Score Confidence Limits" on page 3286.

TEST < =*type* | (*types*) >

requests common risk difference tests. You can specify one or more *types*. When you specify only one *type*, you can omit the parentheses. If you do not specify *types*, this option provides tests that correspond to the confidence limit *types* that you specify in the CL= option.

You can specify one or more of the following *types*:

MH

requests a Mantel-Haenszel test, which is computed by using Mantel-Haenszel stratum weights and the Sato variance estimator (Sato 1989). For more information, see the section "Mantel-Haenszel Confidence Limits and Test" on page 3283.

MR < (VAR=SAMPLE) >

MINRISK < (VAR=SAMPLE) >

requests the minimum risk test, which is computed by using minimum risk weights. If you specify VAR=SAMPLE, PROC FREQ uses the sample (observed) variance estimate instead of a null variance estimate to compute the minimum risk test statistic. For more information, see the section "Minimum Risk Confidence Limits and Test" on page 3284.

SCORE

requests the summary score test. For more information, see the section "Summary Score Confidence Limits" on page 3286.

CQ

COCHQ

requests Cochran's Q test of marginal homogeneity, which can be computed for multiway tables in which each variable has two levels ($2 \times 2 \times \cdots \times 2$ tables). For more information, see the section "Cochran's Q Test" on page 3307.

The AGREE option provides Cochran's Q test together with several other measures of agreement. You can use the CQ option to compute Cochran's Q test without the other measures of agreement.

CROSSLIST < (*options*) >

displays crosstabulation tables by using an ODS column format instead of the default crosstabulation table cell format. In CROSSLIST tables, the rows correspond to the crosstabulation table cells, and the columns correspond to descriptive statistics such as frequencies and percentages. For more information about the contents of CROSSLIST tables, see the section "Two-Way and Multiway Tables" on page 3328.

By default, CROSSLIST tables display all levels of the column variable within each level of the row variable, including any levels that have frequencies of 0. By default, multiway CROSSLIST tables display all levels of the row variable within each stratum, including any row levels that have frequencies of 0 in the stratum. To suppress variable levels that have frequencies of 0, you can specify the NOSPARSE option.

You cannot specify both the LIST option and the CROSSLIST option in the same TABLES statement.

You can specify the following *options*:

CELLCHI2

displays cell chi-squares in the CROSSLIST table. A cell chi-square is the table cell's contribution to the Pearson chi-square statistic. The cell chi-square is computed as $(\mathit{frequency} - \mathit{expected})^2/\mathit{expected}$, where *frequency* is the table cell frequency (count) and *expected* is the expected cell frequency, which is computed under the null hypothesis that the row and column variables are independent. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249.

DEVIATION

displays deviations in the CROSSLIST table. A deviation is the difference between the observed table cell frequency and the expected frequency (*frequency – expected*). The expected frequencies are computed under the null hypothesis that the row and column variables are independent. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249. You can display the expected frequencies in the CROSSLIST table by specifying the CROSSLIST(EXPECTED) option.

EXPECTED

displays expected frequencies in the CROSSLIST table. Expected table cell frequencies are computed as the product of the corresponding row totals and column totals divided by the total frequency of the two-way table. These are the expected frequencies under the null hypothesis that the row and column variables are independent. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249.

FORMAT=*format-name*

specifies the format for frequencies, expected frequencies, and deviations in the CROSSLIST table.

You can specify any standard SAS numeric format, or you can specify a numeric format that is defined by the FORMAT procedure. The format length must not exceed 24. By default, PROC FREQ uses the BEST8. format to display frequencies in the CROSSLIST table.

To change formats in any table that PROC FREQ displays, you can use PROC TEMPLATE. For more information, see the chapter "The TEMPLATE Procedure" in the *SAS Output Delivery System: User's Guide*.

FORMATP=*format-name*

specifies the format for percentages in the CROSSLIST table.

You can specify any standard SAS numeric format, or you can specify a numeric format that is defined by the FORMAT procedure. The format length must not exceed 24. By default, PROC FREQ uses the format 6.2 to display percentages in the CROSSLIST table.

To change formats in any table that PROC FREQ displays, you can use PROC TEMPLATE. For more information, see the chapter "The TEMPLATE Procedure" in the *SAS Output Delivery System: User's Guide*.

NOCOL

suppresses the display of column percentages in the CROSSLIST table.

NOFREQ

suppresses the display of frequencies in the CROSSLIST table.

NOPERCENT

suppresses the display of table percentages (the Percent column) in the CROSSLIST table. To suppress the display of row or column percentages, you can specify the CROSSLIST(NOROW) or CROSSLIST(NOCOL) option, respectively.

NOROW

suppresses the display of row percentages in the CROSSLIST table.

PEARSONRES

displays Pearson residuals in the CROSSLIST table. The Pearson residual is the square root of the table cell's contribution to the Pearson chi-square statistic. The Pearson residual is computed as $(frequency - expected)/\sqrt{expected}$, where *frequency* is the table cell frequency (count) and *expected* is the expected table cell frequency, which is computed under the null hypothesis that the row and column variables are independent. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249.

You can display the expected values, deviations, and cell chi-squares in the CROSSLIST table by specifying the CROSSLIST(EXPECTED), CROSSLIST(DEVIATION), and CROSSLIST(CELLCHI2) options, respectively.

STDRES

displays standardized residuals in the CROSSLIST table. The standardized residual is the ratio of (*frequency* – *expected*) to its standard error, where *frequency* is the table cell frequency (count) and *expected* is the expected table cell frequency, which is computed under the null hypothesis that the row and column variables are independent. For more information, see the section "Standardized Residuals" on page 3249.

You can display the expected values and deviations in the CROSSLIST table by specifying the CROSSLIST(EXPECTED) and CROSSLIST(DEVIATION) options, respectively.

TOTPCT

displays percentages of the total multiway table frequency for multiway tables (n-way tables where $n > 2$). By default, the CROSSLIST table displays percentages of the two-way table (stratum) frequency but does not display percentages of the total multiway table frequency. For more information, see the section "Two-Way and Multiway Tables" on page 3328.

CUMCOL

displays the cumulative column percentages in the cells of the crosstabulation table. This option applies only to crosstabulation tables that are displayed in the default crosstabulation cell format.

DEVIATION

displays deviations in the crosstabulation table. A deviation is the difference between the observed table cell frequency and the expected frequency (*frequency* – *expected*). The expected frequencies are computed under the null hypothesis that the row and column variables are independent. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249. You can display the expected values by specifying the EXPECTED option.

This option applies to two-way and multiway tables that are displayed in the default crosstabulation cell format or in CROSSLIST format. To display deviations in one-way frequency tables, you can specify the ONEWAY(DEVIATION) option.

EXPECTED

displays expected frequencies in the crosstabulation table. Expected cell frequencies are computed as the product of the row marginals and the column marginals divided by the total frequency of the two-way table. These are the expected frequencies under the null hypothesis that the row and column variables are independent. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249.

This option applies to two-way and multiway tables that are displayed in the default crosstabulation cell format or in CROSSLIST format. To display expected frequencies in one-way frequency tables, you can specify the ONEWAY(EXPECTED) option.

FISHER

requests Fisher's exact test for tables that are larger than 2×2. (For 2×2 tables, the CHISQ option provides Fisher's exact test.) This test is also known as the Freeman-Halton test. See the sections "Fisher's Exact Test" on page 3252 and "Exact Statistics" on page 3318 for more information.

If you omit the CHISQ option in the TABLES statement, the FISHER option invokes CHISQ. You can also request Fisher's exact test by specifying the FISHER option in the EXACT statement.

NOTE: PROC FREQ computes exact tests by using fast and efficient algorithms that are superior to direct enumeration. Exact tests are appropriate when a data set is small, sparse, skewed, or heavily tied. For some large problems, computation of exact tests might require a substantial amount of time and memory. Consider using asymptotic tests for such problems. Alternatively, when asymptotic methods might not be sufficient for such large problems, consider using Monte Carlo estimation of exact *p*-values. You can request Monte Carlo estimation by specifying the MC *computation-option* in the EXACT statement. See the section "Computational Resources" on page 3320 for more information.

FORMAT=*format-name*

specifies the format for frequencies, expected frequencies, and deviations in crosstabulation tables.

You can specify any standard SAS numeric format, or you can specify a numeric format that is defined by the FORMAT procedure. The format length must not exceed 24.

This option applies to crosstabulation tables that are displayed in default crosstabulation cell form, CROSSLIST form, or LIST form. You can use the ONEWAY(FORMAT=) option to specify the format for frequencies in one-way frequency tables.

To change display formats in any table that PROC FREQ produces, you can use PROC TEMPLATE. For more information, see the chapter "The TEMPLATE Procedure" in the *SAS Output Delivery System: User's Guide*.

FORMATP=*format-name*

specifies the format for percentages in crosstabulation tables.

You can specify any standard SAS numeric format, or you can specify a numeric format that is defined by the FORMAT procedure. The format length must not exceed 24.

This option applies to crosstabulation tables that are displayed in default crosstabulation cell form, CROSSLIST form, or LIST form. You can use the ONEWAY(FORMATP=) option to specify the format for percentages in one-way tables.

To change display formats in any table that PROC FREQ produces, you can use PROC TEMPLATE. For more information, see the chapter "The TEMPLATE Procedure" in the *SAS Output Delivery System: User's Guide*.

GAILSIMON < (COLUMN=1 | 2) >

GS < (COLUMN=1 | 2) >

requests the Gail-Simon test for qualitative interaction, which applies to stratified 2×2 tables. For more information, see the section "Gail-Simon Test for Qualitative Interactions" on page 3317.

The COLUMN= option specifies the column of the risk differences to use to compute the Gail-Simon test. By default, PROC FREQ uses column 1 risk differences. If you specify COLUMN=2, PROC FREQ uses column 2 risk differences.

JT

requests the Jonckheere-Terpstra test. For more information, see the section "Jonckheere-Terpstra Test" on page 3298. To request exact *p*-values for the Jonckheere-Terpstra test, specify the JT option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

LIST < (*options*) >

displays two-way and multiway tables by using a list format instead of the default crosstabulation cell format. This option displays an entire multiway table in a single table instead of separate two-way (stratum) tables. In list format, each row of the table corresponds to a single crosstabulation table cell. For more information, see the section "Two-Way and Multiway Tables" on page 3328.

By default, a table in list format does not display cells for which the frequency is 0 unless you specify the ZEROS option in the WEIGHT statement and do not specify the NOSPARSE option in the TABLES statement.

The LIST option is not available in the same TABLES statement together with statistic options. When you specify statistic options in a TABLES statement, you can display the crosstabulation tables by using the default format or the CROSSLIST format. However, you can specify the LIST option and statistic options in different TABLES statements in the same invocation of PROC FREQ.

You can specify the following *options*:

FORMAT=*format-name*

specifies the format for frequencies and cumulative frequencies in the LIST crosstabulation table.

You can specify any standard SAS numeric format, or you can specify a numeric format that is defined by the FORMAT procedure. The format length must not exceed 24. By default, PROC FREQ uses the BEST8. format to display frequencies in the LIST crosstabulation table.

To change display formats in any table that PROC FREQ produces, you can use PROC TEMPLATE. For more information, see the chapter "The TEMPLATE Procedure" in the *SAS Output Delivery System: User's Guide*.

FORMATP=*format-name*

specifies the format for percentages and cumulative percentages in the LIST crosstabulation table.

You can specify any standard SAS numeric format, or you can specify a numeric format that is defined by the FORMAT procedure. The format length must not exceed 24. By default, PROC FREQ uses the format 6.2 to display percentages in the LIST table.

To change display formats in any table that PROC FREQ produces, you can use PROC TEMPLATE. For more information, see the chapter "The TEMPLATE Procedure" in the *SAS Output Delivery System: User's Guide*.

NOCUM

suppresses the display of cumulative frequencies and cumulative percentages in the LIST crosstabulation table.

NOFREQ

suppresses the display of frequencies and cumulative frequencies in the LIST crosstabulation table.

NOPERCENT

suppresses the display of percentages and cumulative percentages in the LIST crosstabulation table.

MAXLEVELS=*n*

specifies the maximum number of variable levels to display in one-way frequency tables. The value of *n* must be a positive integer. PROC FREQ displays the first *n* variable levels, matching the order in which the levels appear in the one-way frequency table. (The ORDER= option controls the order of the variable levels. By default, ORDER=INTERNAL, which orders the variable levels by unformatted value.)

The MAXLEVELS= option also applies to one-way frequency plots, which you can request by specifying the PLOTS=FREQPLOT option when ODS Graphics is enabled.

If you specify the MISSPRINT option to display missing levels in the frequency table, the MAXLEVELS= option displays the first *n* nonmissing levels.

The MAXLEVELS= option does not apply to the OUT= output data set, which includes all variable levels. The MAXLEVELS= option does not affect the computation of percentages, statistics, or tests for the one-way table; these values are based on the complete table.

MEASURES <(CL)>

requests measures of association and their asymptotic standard errors. This option provides the following measures: gamma, Kendall's tau-*b*, Stuart's tau-*c*, Somers' $D(C|R)$, Somers' $D(R|C)$, Pearson and Spearman correlation coefficients, lambda (symmetric and asymmetric), and uncertainty coefficients (symmetric and asymmetric). If you specify the CL option in parentheses after the MEASURES option, PROC FREQ provides confidence limits for the measures of association. For more information, see the section "Measures of Association" on page 3253.

For 2×2 tables, the MEASURES option also provides the odds ratio, column 1 relative risk, column 2 relative risk, and their asymptotic Wald confidence limits. You can request the odds ratio and relative risks separately (without the other measures of association) by specifying the RELRISK option. You can request confidence limits for the odds ratio by specifying the OR(CL=) option.

You can use the TEST statement to request asymptotic tests for the following measures of association: gamma, Kendall's tau-*b*, Stuart's tau-*c*, Somers' $D(C|R)$, Somers' $D(R|C)$, and Pearson and Spearman correlation coefficients. You can use the EXACT statement to request exact confidence limits for the odds ratio, exact unconditional confidence limits for the relative risks, and exact tests for the following measures of association: Kendall's tau-*b*, Stuart's tau-*c*, Somers' $D(C|R)$ and $D(R|C)$, and Pearson and Spearman correlation coefficients. For more information, see the descriptions of the TEST and EXACT statements and the section "Exact Statistics" on page 3318.

MISSING

treats missing values as a valid nonmissing level for all TABLES variables. The MISSING option displays the missing levels in frequency and crosstabulation tables and includes them in all calculations of percentages, tests, and measures.

By default, if you do not specify the MISSING or MISSPRINT option, an observation is excluded from a table if it has a missing value for any of the variables in the TABLES request. When PROC FREQ excludes observations with missing values, it displays the total frequency of missing observations following the table. See the section "Missing Values" on page 3243 for more information.

MISSPRINT

displays missing value frequencies in frequency and crosstabulation tables but does not include the missing value frequencies in any computations of percentages, tests, or measures.

By default, if you do not specify the MISSING or MISSPRINT option, an observation is excluded from a table if it has a missing value for any of the variables in the TABLES request. When PROC FREQ excludes observations with missing values, it displays the total frequency of missing observations following the table. See the section "Missing Values" on page 3243 for more information.

NOCOL

suppresses the display of column percentages in crosstabulation tables. A table cell's column percentage is computed as the cell frequency divided by the corresponding column frequency.

NOCUM

suppresses the display of cumulative frequencies and percentages in one-way frequency tables and in list-format crosstabulation tables (which you request by specifying the LIST option).

NOFREQ

suppresses the display of table cell frequencies in crosstabulation tables. This option also suppresses row total frequencies.

In list-format crosstabulation tables (which you can request by specifying the LIST option), the NOFREQ option suppresses the display of frequencies and cumulative frequencies. To suppress display of frequencies and cumulative frequencies in one-way frequency tables, you can specify the ONEWAY(NOFREQ) option.

NOPERCENT

suppresses the display of percentages in crosstabulation tables. These percentages include table cell, row, and column percentages of the total two-way table frequency. To suppress the display of cell percentages of row or column totals, you can specify the NOROW or NOCOL option, respectively.

For one-way frequency tables and list-format crosstabulation tables, the NOPERCENT option suppresses the display of percentages and cumulative percentages.

NOPRINT

suppresses the display of frequency and crosstabulation tables but displays all requested tests and statistics. To suppress the display of all output, including tests and statistics, use the NOPRINT option in the PROC FREQ statement.

NOROW

suppresses the display of row percentages in crosstabulation tables. A table cell's row percentage is computed as the cell frequency divided by the corresponding row frequency.

NOSPARSE

suppresses the display of zero-frequency table cells in LIST and CROSSLIST tables. This option also excludes zero-frequency table cells from OUT= output data sets.

When you specify the ZEROS option in the WEIGHT statement, PROC FREQ includes observations that have weights of 0 in the analysis. By default, zero-frequency table cells are displayed in the LIST table and included in the OUT= data set. You can use the NOSPARSE option to exclude the zero-frequency table cells.

When you specify the CROSSLIST option, by default the CROSSLIST table displays all levels of the column variable within each level of the row variable (including any levels that have frequencies of 0). By default for multiway CROSSLIST tables, the CROSSLIST table displays all levels of the row variable within each stratum of the table (including any row levels that have frequencies of 0 in the stratum). You can use the NOSPARSE option to suppress the zero-frequency variable levels in the CROSSLIST table.

NOWARN

suppresses the log warning message for the validity of the asymptotic Pearson chi-square test. By default, PROC FREQ provides a validity warning for the asymptotic Pearson chi-square test when more than 20% of the table cells have expected frequencies that are less than 5. This warning message appears in the log if you specify the NOPRINT option in the PROC FREQ statement,

The NOWARN option is equivalent to the CHISQ(WARN=NOLOG) option. You can also use the CHISQ(WARN=) option to suppress the warning message in the display and to request a warning variable in the chi-square ODS output data set or in the OUTPUT data set.

ONEWAY (*options*)

controls the statistics to display in one-way frequency tables. For more information about the contents of one-way frequency tables, see the section "One-Way Frequency Tables" on page 3326.

You can specify the following *options*:

CELLCHI2

displays cell chi-squares in the one-way frequency table. A cell chi-square is the table cell's contribution to the Pearson chi-square statistic. The cell chi-square is computed as $(\mathit{frequency} - \mathit{expected})^2/\mathit{expected}$, where *frequency* is the table cell (level) frequency and *expected* is the expected frequency. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248.

DEVIATION

displays deviations in the one-way frequency table. A deviation is the difference between the observed table cell (level) frequency and the expected frequency (*frequency* – *expected*), where the expected frequencies are the null hypothesis frequencies for the chi-square goodness-of-fit test.

By default, the expected frequency for each level is the total frequency divided by the number of levels. Alternatively, you can provide null frequencies or null proportions by specifying the CHISQ(TESTF=) or CHISQ(TESTP=) option, respectively. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248.

EXPECTED

displays expected frequencies in the one-way frequency table, where the expected frequencies are the null hypothesis frequencies for the chi-square goodness-of-fit test.

By default, the expected frequency for each class level is the total frequency divided by the number of levels. Alternatively, you can provide null frequencies or null proportions by specifying the CHISQ(TESTF=) or CHISQ(TESTP=) option, respectively. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248.

When you specify the CHISQ(TESTF=) option, the one-way frequency table displays the specified null hypothesis frequencies by default, and the EXPECTED option has no effect.

FORMAT=*format-name*

specifies the format for frequencies, expected frequencies, and deviations in the one-way frequency table.

You can specify any standard SAS numeric format, or you can specify a numeric format that is defined by the FORMAT procedure. The format length must not exceed 24. By default, PROC FREQ uses the BEST8. format to display frequencies in the one-way frequency table.

To change display formats in any table that PROC FREQ produces, you can use PROC TEMPLATE. For more information, see the chapter "The TEMPLATE Procedure" in the *SAS Output Delivery System: User's Guide*.

FORMATP=*format-name*

specifies the format for percentages in the one-way frequency table.

You can specify any standard SAS numeric format, or you can specify a numeric format that is defined by the FORMAT procedure. The format length must not exceed 24. By default, PROC FREQ uses the format 6.2 to display percentages in the one-way frequency table.

To change display formats in any table that PROC FREQ produces, you can use PROC TEMPLATE. For more information, see the chapter "The TEMPLATE Procedure" in the *SAS Output Delivery System: User's Guide*.

MAXLEVELS=*n*

specifies the maximum number of variable levels to display in the one-way frequency table. The value of *n* must be a positive integer. PROC FREQ displays the first *n* variable levels, matching the order in which the levels appear in the one-way frequency table. (The ORDER= option controls the order of the variable levels. By default, ORDER=INTERNAL, which orders the variable levels by unformatted values.)

The MAXLEVELS= option also applies to one-way frequency plots, which you can request by specifying the PLOTS=FREQPLOT option when ODS Graphics is enabled.

If you specify the MISSPRINT option to display missing levels in frequency tables, MAXLEVELS=*n* displays the first *n* nonmissing levels.

The MAXLEVELS= option does not apply to the OUT= output data set, which includes all variable levels. The MAXLEVELS= option does not affect the computation of percentages, statistics, or tests for the one-way table; these values are based on the complete table.

NOCUM

suppresses the display of cumulative frequencies and percentages in one-way frequency tables.

NOFREQ

suppresses the display of frequencies and cumulative frequencies in one-way frequency tables.

NOPERCENT

suppresses the display of percentages and cumulative percentages in one-way frequency tables.

RESIDUAL

displays Pearson residuals in the one-way frequency table. The Pearson residual is the square root of the table cell's contribution to the Pearson chi-square statistic. The Pearson residual is computed as $(\mathit{frequency} - \mathit{expected})/\sqrt{\mathit{expected}}$, where *frequency* is the table cell frequency (count) and *expected* is the expected table cell frequency. For more information, see the section "Chi-Square Test for One-Way Tables" on page 3248.

OR < (CL=*type* | (*types*) >

ODDSRATIO < (CL=*type* | (*types*) >

requests the odds ratio and confidence limits for 2×2 tables. For more information, see the section "Odds Ratio" on page 3287.

You can specify one or more *types* of confidence limits. When you specify only one confidence limit *type*, you can omit the parentheses around the request. PROC FREQ displays the confidence limits in the "Confidence Limits for the Odds Ratio" table.

Specifying the OR option without the CL= option is equivalent to specifying the RELRISK option, which produces the "Odds Ratio and Relative Risks" table. For more information, see the description of the RELRISK option. When you specify the OR(CL=) option, PROC FREQ does not produce the "Odds Ratio and Relative Risks" table unless you also specify the RELRISK or MEASURES option.

The ALPHA= option determines the confidence level; by default, ALPHA=0.05, which produces 95% confidence limits for the odds ratio.

You can specify the following *types*:

EXACT

displays exact confidence limits for the odds ratio in the "Confidence Limits for the Odds Ratio" table. (By default, PROC FREQ displays the exact confidence limits in a separate table.) You must also request computation of the exact confidence limits by specifying the OR option in the EXACT statement. For more information, see the subsection "Exact Confidence Limits" in the section "Confidence Limits for the Odds Ratio" on page 3288.

LR

LIKELIHOODRATIO

requests likelihood ratio confidence limits for the odds ratio. For more information, see the subsection "Likelihood Ratio Confidence Limits" in the section "Confidence Limits for the Odds Ratio" on page 3288.

MIDP

requests exact mid-*p* confidence limits for the odds ratio. For more information, see the subsection "Exact Mid-*p* Confidence Limits" in the section "Confidence Limits for the Odds Ratio" on page 3288.

SCORE < (CORRECT=NO) >

requests score confidence limits for the odds ratio. For more information, see the subsection "Score Confidence Limits" in the section "Confidence Limits for the Odds Ratio" on page 3288. If you specify CORRECT=NO, PROC FREQ provides the uncorrected form of the score confidence limits.

WALD

requests asymptotic Wald confidence limits, which are based on a log transformation of the odds ratio. For more information, see the subsection "Wald Confidence Limits" in the section "Confidence Limits for the Odds Ratio" on page 3288.

WALDMODIFIED

requests Wald modified confidence limits for the odds ratio. For more information, see the subsection "Wald Modified Confidence Limits" in the section "Confidence Limits for the Odds Ratio" on page 3288.

OUT=*SAS-data-set*

names an output data set that contains frequency or crosstabulation table counts and percentages. If more than one table request appears in the TABLES statement, the contents of the OUT= data set correspond to the last table request in the TABLES statement. The OUT= data set variable COUNT contains the frequencies and the variable PERCENT contains the percentages. For more information, see the section "Output Data Sets" on page 3323. You can specify the following options to include additional information in the OUT= data set: OUTCUM, OUTEXPECT, and OUTPCT.

OUTCUM

includes cumulative frequencies and cumulative percentages in the OUT= data set for one-way tables. The variable CUM_FREQ contains the cumulative frequencies, and the variable CUM_PCT contains the cumulative percentages. For more information, see the section "Output Data Sets" on page 3323. The OUTCUM option has no effect for two-way or multiway tables.

OUTEXPECT

includes expected cell frequencies in the OUT= data set for crosstabulation tables. The variable EXPECTED contains the expected cell frequencies. For more information, see the section "Output Data Sets" on page 3323. The EXPECTED option has no effect for one-way tables.

OUTPCT

includes the following additional variables in the OUT= data set for crosstabulation tables:

PCT_COL	percentage of column frequency
PCT_ROW	percentage of row frequency
PCT_TABL	percentage of stratum (two-way table) frequency, for n-way tables where $n > 2$

For more information, see the section "Output Data Sets" on page 3323. The OUTPCT option has no effect for one-way tables.

PLCORR < (*options*) >

POLYCHORIC < (*options*) >

requests the polychoric correlation coefficient and its asymptotic standard error. For 2×2 tables, this statistic is more commonly known as the tetrachoric correlation coefficient and is labeled as such in the displayed output. For more information, see the section "Polychoric Correlation" on page 3260.

If you also specify the CL or MEASURES(CL) option, PROC FREQ provides confidence limits for the polychoric correlation. If you specify the PLCORR option in the TEST statement, the procedure provides Wald and likelihood ratio tests for the polychoric correlation. The PLCORR option invokes the MEASURES option.

You can specify the following *options*:

ADJUST

replaces a 2×2 table cell frequency of 0 with 0.5 before computing the tetrachoric correlation (Brown and Benedetti 1977a, p. 353). To maintain the row and column marginal frequencies, adjacent cell frequencies are decreased by 0.5 and the opposite cell frequency is increased by 0.5.

This option is available for 2×2 tables and is applied only when a single cell frequency is 0. It has no effect when both off-diagonal cell frequencies are 0 (and therefore the correlation is 1) or when both diagonal cell frequencies are 1 (and therefore the correlation is –1).

CONVERGE=*value*

specifies the convergence criterion. The *value* must be a positive number. By default, CONVERGE=0.0001. Iterative computation of the polychoric correlation stops when the convergence measure falls below *value* or when the number of iterations exceeds the MAXITER= *number*, whichever happens first. For parameter values that are less than 0.01, PROC FREQ evaluates convergence by using the absolute difference instead of the relative difference. For more information, see the section "Polychoric Correlation" on page 3260.

MAXITER=*number*

specifies the maximum *number* of iterations. The value of *number* must be a positive integer. By default, MAXITER=50. Iterative computation of the polychoric correlation stops when the number of iterations exceeds the maximum *number* or when the convergence measure falls below the CONVERGE= *value*, whichever happens first. For more information, see the section "Polychoric Correlation" on page 3260.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*plot-options*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*plot-options*) > < . . . *plot-request* < (*plot-options*) > >) >

controls the plots that are produced through ODS Graphics. *Plot-requests* identify the plots, and *plot-options* control the appearance and content of the plots. You can specify *plot-options* in parentheses after a *plot-request*. A *global-plot-option* applies to all plots for which it is available unless it is altered by a specific *plot-option*. You can specify *global-plot-options* in parentheses after the PLOTS option.

When you specify only one *plot-request*, you can omit the parentheses around the request. For example:

```
plots=all
plots=freqplot
plots=(freqplot oddsratioplot)
```

```
plots(only)=(cumfreqplot deviationplot)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc freq;
   tables treatment*response / chisq plots=freqplot;
   weight wt;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, PROC FREQ produces all plots that are associated with the analyses that you request, with the exception of the frequency, cumulative frequency, and mosaic plots. To produce a frequency plot or cumulative frequency plot when ODS Graphics is enabled, you must specify the FREQPLOT or CUMFREQPLOT *plot-request*, respectively, in the PLOTS= option, or you must specify the PLOTS=ALL option. To produce a mosaic plot when ODS Graphics is enabled, you must specify the MOSAICPLOT *plot-request* in the PLOTS= option, or you must specify the PLOTS=ALL option.

PROC FREQ produces the remaining plots (listed in Table 47.11) by default when you request the corresponding TABLES statement options. You can suppress default plots and request specific plots by using the PLOTS(ONLY)= option; PLOTS(ONLY)=(*plot-requests*) produces only the plots that are specified as *plot-requests*. You can suppress all plots by specifying the PLOTS=NONE option. The PLOTS option has no effect when you specify the NOPRINT option in the PROC FREQ statement.

Plot Requests

Table 47.11 lists the available *plot-requests* together with their required TABLES statement options. Descriptions of the *plot-requests* follow the table in alphabetical order.

Table 47.11 Plot Requests

Plot Request	Description	Required TABLES Statement Option
AGREEPLOT	Agreement plot	AGREE ($r \times r$ table)
ALL	All plots	None
CUMFREQPLOT	Cumulative frequency plot	One-way table request
DEVIATIONPLOT	Deviation plot	CHISQ (one-way table)
FREQPLOT	Frequency plot	Any table request
KAPPAPLOT	Kappa plot	AGREE ($h \times r \times r$ table)
MOSAICPLOT	Mosaic plot	Two-way or multiway table request
NONE	No plots	None
ODDSRATIOPLOT	Odds ratio plot	MEASURES, OR, or RELRISK ($h \times 2 \times 2$ table)
RELRISKPLOT	Relative risk plot	MEASURES or RELRISK ($h \times 2 \times 2$ table)
RISKDIFFPLOT	Risk difference plot	RISKDIFF ($h \times 2 \times 2$ table)
WTKAPPAPLOT	Weighted kappa plot	AGREE ($h \times r \times r$ table, $r > 2$)

You can specify the following *plot-requests*:

AGREEPLOT < (*plot-options*) >

requests an agreement plot (Bangdiwala and Bryan 1987), An agreement plot displays the strength of agreement in a two-way table, where the row and column variables represent two independent ratings of n subjects. For information about agreement plots, see Bangdiwala (1988), Bangdiwala et al. (2008), and Friendly (2000, Section 3.7.2).

To produce an agreement plot, you must also specify the AGREE option in the TABLES statement. Agreement statistics and plots are available for two-way square tables, where the number of rows equals the number of columns.

Table 47.12 lists the *plot-options* that are available for agreement plots. For descriptions of the *plot-options*, see the subsection "Plot Options" in this section.

Table 47.12 Plot Options for AGREEPLOT

Plot Option	Description	Values
LEGEND=	Legend	NO or YES*
PARTIAL=	Partial agreement	NO or YES*
SHOWSCALE=	Frequency scale	NO or YES*
STATS	Statistics	None

*Default

If you specify the STATS *plot-option*, the agreement plot displays the values of the kappa coefficient, the weighted kappa coefficient, the B_n measure (Bangdiwala and Bryan 1987), and the sample size. PROC FREQ stores these statistics in an ODS table named BnMeasure, which is not displayed. For more information, see the section "ODS Table Names" on page 3336.

ALL

requests all plots that are associated with the specified analyses. Table 47.11 lists the available *plot-requests* and the corresponding analysis options. If you specify the PLOTS=ALL option, PROC FREQ produces the frequency, cumulative frequency, and mosaic plots that are associated with the tables that you request. (These plots are not produced by default when ODS Graphics is enabled.)

CUMFREQPLOT < (*plot-options*) >

requests a plot of cumulative frequencies. Cumulative frequency plots are available for one-way frequency tables.

To produce a cumulative frequency plot, you must specify the CUMFREQPLOT *plot-request* in the PLOTS= option, or you must specify the PLOTS=ALL option. PROC FREQ does not produce cumulative frequency plots by default when ODS Graphics is enabled.

Table 47.13 lists the *plot-options* that are available for cumulative frequency plots. For descriptions of the *plot-options*, see the subsection "Plot Options" in this section.

Table 47.13 Plot Options for CUMFREQPLOT

Plot Option	Description	Values
ORIENT=	Orientation	HORIZONTAL or VERTICAL*
SCALE=	Scale	FREQ* or PERCENT
TYPE=	Type	BARCHART* or DOTPLOT

*Default

DEVIATIONPLOT < (*plot-options*) >

requests a plot of relative deviations from expected frequencies. Deviation plots are available for chi-square analysis of one-way frequency tables. To produce a deviation plot, you must also specify the CHISQ option in the TABLES statement for a one-way frequency table.

Table 47.14 lists the *plot-options* that are available for deviation plots.

Table 47.14 Plot Options for DEVIATIONPLOT

Plot Option	Description	Values
NOSTAT	No statistic	None
ORIENT=	Orientation	HORIZONTAL or VERTICAL*
PLOTVAR=	Plot variable	CELLCHI2, DEVIATION*, and RESIDUAL
SCALE=	Scale	ABS, NONE*, and PERCENT
TYPE=	Type	BARCHART* or DOTPLOT

*Default

For descriptions of the NOSTAT, ORIENT=, and TYPE= *plot-options*, see the subsection "Plot Options" in this section.

You can specify the following *plot-options*:

PLOTVAR=CELLCHI2 | DEVIATION | RESIDUAL

specifies the deviation statistic to display in the deviation plot. By default, PLOTVAR=DEVIATION, which displays the relative deviations. For each class level, the relative deviation is computed as the deviation divided by the expected frequency under the null hypothesis ((*frequency* – *expected*) / *expected*).

PLOTVAR=CELLCHI2 displays the cell chi-square contributions, which are computed as $(frequency - expected)^2/expected$. For more information, see the ONEWAY(CELLCHI2) option.

PLOTVAR=RESIDUAL displays the Pearson residuals, which are computed as $(frequency - expected)/\sqrt{expected}$. For more information, see the ONEWAY(RESIDUAL) option.

SCALE=ABS | NONE | PERCENT

specifies the scale of the deviation statistic to display. By default, SCALE=NONE, which displays the unscaled deviations. SCALE=ABS displays the absolute values of the deviation statistics (relative deviations or residuals). SCALE=ABS has no effect for the cell chi-squares, which are nonnegative. SCALE=PERCENT displays the percentages of the total of the absolute deviations.

The SCALE=PERCENT *global-plot-option* invokes the DEVIATIONPLOT(SCALE=PERCENT) *plot-option*. You can alter this by specifying the individual DEVIATIONPLOT(SCALE=) *plot-option*.

FREQPLOT < (*plot-options*) >

requests a frequency plot. Frequency plots are available for frequency and crosstabulation tables. For multiway crosstabulation tables, PROC FREQ provides a two-way frequency plot for each stratum (two-way table).

To produce a frequency plot, you must specify the FREQPLOT *plot-request* in the PLOTS= option, or you must specify the PLOTS=ALL option. PROC FREQ does not produce frequency plots by default when ODS Graphics is enabled.

Table 47.15 lists the *plot-options* that are available for frequency plots. For descriptions of the *plot-options*, see the subsection "Plot Options" in this section.

Table 47.15 Plot Options for FREQPLOT

Plot Option	Description	Values
GROUPBY=**	Primary group	COLUMN* or ROW
NPANELPOS=**	Sections per panel	Number (4*)
ORIENT=	Orientation	HORIZONTAL or VERTICAL*
SCALE=	Scale	FREQ*, GROUPPERCENT**, LOG, PERCENT, SQRT
TWOWAY=**	Two-way layout	CLUSTER, GROUPHORIZONTAL, GROUPVERTICAL*, or STACKED
TYPE=	Type	BARCHART* or DOTPLOT

*Default

**For two-way tables

You can specify the following *plot-options* for all frequency plots: ORIENT=, SCALE=, and TYPE=. You can specify the following *plot-options* for frequency plots of two-way (and multiway) tables: GROUPBY=, NPANELPOS=, and TWOWAY=. The NPANELPOS= *plot-option* is not available with the TWOWAY=CLUSTER or TWOWAY=STACKED layout, which is always displayed in a single panel.

By default, PROC FREQ displays frequency plots as bar charts. To display frequency plots as dot plots, specify TYPE=DOTPLOT. To plot percentages instead of frequencies, specify SCALE=PERCENT. For two-way tables, there are four frequency plot layouts available, which you can request by specifying the TWOWAY= *plot-option*. For more information, see the subsection "Plot Options" in this section.

By default, graph cells in a two-way layout are first grouped by column variable levels; row variable levels are then displayed within the column variable levels. To group first by row variable levels, specify GROUPBY=ROW.

KAPPAPLOT < (*plot-options*) >

requests a plot of kappa statistics along with confidence limits. Kappa plots are available for multiway square tables and display the kappa statistic (with confidence limits) for each two-way table (stratum). Kappa plots also display the overall kappa statistic unless you specify the COMMON=NO *plot-option*. To produce a kappa plot, you must specify the AGREE option in the TABLES statement to compute kappa statistics.

Table 47.16 lists the *plot-options* that are available for kappa plots. For descriptions of the *plot-options*, see the subsection "Plot Options" in this section.

Table 47.16 Plot Options for KAPPAPLOT and WTKAPPAPLOT

Plot Option	Description	Values
CLDISPLAY=	Error bar type	BAR, LINE, LINEARROW, SERIF*, or SERIFARROW
COMMON=	Overall kappa	NO or YES*
NPANELPOS=	Statistics per graphic	Number (all*)
ORDER=	Order of two-way levels	ASCENDING or DESCENDING
RANGE=	Range to display	Values or CLIP
STATS	Statistic values	None

*Default

MOSAICPLOT < (*plot-options*) >

requests a mosaic plot. Mosaic plots are available for two-way and multiway crosstabulation tables; for multiway tables, PROC FREQ provides a mosaic plot for each two-way table (stratum).

To produce a mosaic plot, you must specify the MOSAICPLOT *plot-request* in the PLOTS= option, or you must specify the PLOTS=ALL option. PROC FREQ does not produce mosaic plots by default when ODS Graphics is enabled.

Mosaic plots display tiles that correspond to the crosstabulation table cells. The areas of the tiles are proportional to the frequencies of the table cells. The column variable is displayed on the X axis, and the tile widths are proportional to the relative frequencies of the column variable levels. The row variable is displayed on the Y axis, and the tile heights are proportional to the relative frequencies of the row levels within column levels. For more information, see Friendly (2000).

By default, the colors of the tiles correspond to the row variable levels. If you specify the COLORSTAT= *plot-option*, the tiles are colored according to the values of the Pearson or standardized residuals.

You can specify the following *plot-options*:

COLORSTAT < =PEARSONRES | STDRES >

colors the mosaic plot tiles according to the values of residuals. If you specify COLORSTAT=PEARSONRES, the tiles are colored according to the Pearson residuals of the corresponding table cells. For more information, see the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249. If you specify COLORSTAT=STDRES, the tiles are colored according to the standardized residuals of the corresponding table cells. For more information, see the section "Standardized Residuals" on page 3249. You can display the Pearson or standardized residuals in the CROSSLIST table by specifying the CROSSLIST(PEARSONRES) or CROSSLIST(STDRES) option, respectively.

SQUARE

produces a square mosaic plot, where the height of the Y axis equals the width of the X axis. In a square mosaic plot, the scale of the relative frequencies is the same on both axes. By default, PROC FREQ produces a rectangular mosaic plot.

NONE

suppresses all plots.

ODDSRATIOPLOT < (*plot-options*) >

requests a plot of odds ratios along with confidence limits. Odds ratio plots are available for multiway 2×2 tables and display the odds ratio (with confidence limits) for each 2×2 table (stratum). To produce an odds ratio plot, you must also specify the MEASURES, OR, or RELRISK option in the TABLES statement to compute the odds ratios.

Table 47.17 lists the *plot-options* that are available for odds ratio plots. For descriptions of the *plot-options*, see the subsection "Plot Options" in this section.

Table 47.17 Plot Options for ODDSRATIOPLOT, RELRISKPLOT, and RISKDIFFPLOT

Plot Option	Description	Values
CL=	Confidence limit type	Type
CLDISPLAY=	Error bar type	BAR, LINE, LINEARROW, SERIF*, or SERIFARROW
COLUMN=**	Risk column	1* or 2
COMMON=	Common value	NO or YES*
LOGBASE=***	Axis scale	2, E, or 10
NPANELPOS=	Statistics per graphic	Number (all*)
ORDER=	Order of two-way levels	ASCENDING or DESCENDING
RANGE=	Range to display	Values or CLIP
STATS	Statistic values	None

*Default

**Available for RELRISKPLOT and RISKDIFFPLOT

***Available for ODDSRATIOPLOT and RELRISKPLOT

You can specify one of the following confidence limit types for the odds ratio plot: exact (CL=EXACT), likelihood ratio (CL=LR), exact mid-p (CL=MIDP), score (CL=SCORE), Wald (CL=WALD), or Wald modified (CL=WALDMODIFIED). By default, the odds ratio plot displays

Wald confidence limits. For more information, see the descriptions of the CL= *plot-option* and the OR(CL=) option.

To display exact confidence limits in the odds ratio plot, you must also request their computation by specifying the OR option in the EXACT statement.

When CL=WALD or CL=EXACT, the odds ratio plot displays the common odds ratio by default when it is available. To compute the common odds ratio along with Wald confidence limits, specify the CMH option in the TABLES statement. To compute the common odds ratio along with exact confidence limits, specify the COMOR option in the EXACT statement. To suppress display of the common odds ratio, specify COMMON=NO.

RELRISKPLOT < (*plot-options*) >

requests a plot of relative risks along with confidence limits. Relative risk plots are available for multiway 2×2 tables and display the relative risk (with confidence limits) for each 2×2 table (stratum). To produce a relative risk plot, you must also specify the MEASURES or RELRISK option in the TABLES statement to compute relative risks.

Table 47.17 lists the *plot-options* that are available for relative risk plots. For descriptions of the *plot-options*, see the subsection "Plot Options" in this section.

You can specify one of the following confidence limit types for the relative risk plot: exact (CL=EXACT), likelihood ratio (CL=LR), score (CL=SCORE), Wald (CL=WALD), or Wald modified (CL=WALDMODIFIED). By default, the relative risk plot displays Wald confidence limits. For more information, see the descriptions of the CL= *plot-option* and the RELRISK(CL=) option.

To display exact confidence limits in the relative risk plot, you must also request their computation by specifying the RELRISK option in the EXACT statement. The risk column that you specify for the confidence limits must match the risk column that you specify for the plot.

The relative risk plot displays the common relative risk by default when you specify CL=WALD and the CMH option in the TABLES statement. To suppress display of the common relative risk, specify COMMON=NO.

In addition to the *plot-options* in Table 47.17, you can specify the following *plot-option*:

FOOTNOTE=NO

suppresses the footnote that identifies the column for which the relative risk is computed.

RISKDIFFPLOT < (*plot-options*) >

requests a plot of risk (proportion) differences along with confidence limits for multiway 2×2 tables. The risk difference plot displays the risk difference (with confidence limits) for each 2×2 table (stratum). Optionally, the plot also displays the common risk difference.

To produce a risk difference plot, you must also specify the RISKDIFF option in the TABLES statement to compute risk differences.

Table 47.17 lists the *plot-options* that are available for risk difference plots. For descriptions of the *plot-options*, see the subsection "Plot Options" in this section.

You can specify the confidence limit type for the stratum risk differences by using the CL= *plot-option*. You can specify one of the following confidence limit types: Agresti-Caffo (CL=AC),

exact (CL=EXACT), Hauck-Anderson (CL=HA), Miettinen-Nurminen (score) (CL=MN), Newcombe (CL=NEWCOMBE), and Wald (CL=WALD). By default, the plot displays Wald confidence limits for the stratum risk differences. For more information, see the descriptions of the CL= *plot-option* and the RISKDIFF(CL=) option.

To display exact confidence limits in the risk difference plot, you must also request their computation by specifying the RISKDIFF option in the EXACT statement. The risk column that you specify for the confidence limits must match the risk column that you specify for the plot.

By default, the risk difference plot displays the common risk difference when you specify the RISKDIFF(COMMON) or COMMONRISKDIFF option unless you specify the CL=EXACT *plot-option*. To suppress display of the common risk difference, specify COMMON=NO.

In addition to the *plot-options* in Table 47.17, you can specify the following *plot-options*:

CLNOTE=NO
suppresses the note that identifies the confidence limit type.

COMMON=*type*
specifies the *type* of confidence limits to display for the common risk difference.

The default common confidence limit *type* depends on the stratum confidence limit *type*. By default, COMMON=MN if CL=MN and COMMON=NEWCOMBE if CL=NEWCOMBE. Otherwise, COMMON=MH by default.

You can specify one of the following *types*:

K
KLINGENBERG
displays Klingenberg confidence limits. For more information, see the COMMONRISKDIFF(CL=K) option and the section "Klingenberg Confidence Limits" on page 3284.

MH
displays Mantel-Haenszel confidence limits. For more information, see the COMMONRISKDIFF(CL=MH) option and the section "Mantel-Haenszel Confidence Limits and Test" on page 3283.

MN
displays Miettinen-Nurminen confidence limits. For more information, see the COMMONRISKDIFF(CL=MN) option.

MNMH
displays confidence limits that are computed by using the Miettinen-Nurminen method with Mantel-Haenszel weights. For more information, see the COMMONRISKDIFF(CL=MNMH) option.

MR
MINRISK
displays minimum risk confidence limits. For more information, see the COMMONRISKDIFF(CL=MR) option and the section "Minimum Risk Confidence Limits and Test" on page 3284.

NEWCOMBE
displays stratified Newcombe confidence limits that use Mantel-Haenszel weights to combine the stratum components. For more information, see the COMMONRISKDIFF(CL=NEWCOMBE) option and the section "Stratified Newcombe Confidence Limits" on page 3286.

NEWCOMBEMR
displays stratified Newcombe confidence limits that use minimum risk weights to combine the stratum components. For more information, see the COMMONRISKDIFF(CL=NEWCOMBEMR) option and the section "Stratified Newcombe Confidence Limits" on page 3286.

NONE
suppresses the common risk difference in the risk difference plot.

SCORE
displays summary score confidence limits. For more information, see the COMMONRISKDIFF(CL=SCORE) option and the section "Summary Score Confidence Limits" on page 3286.

FOOTNOTE=NO
suppresses the footnote that identifies the column for which the risk difference is computed.

WTKAPPAPLOT < (*plot-options*) >
requests a plot of weighted kappa coefficients along with confidence limits. Weighted kappa plots are available for multiway square tables and display the weighted kappa coefficient (with confidence limits) for each two-way table (stratum). Weighted kappa plots also display the overall weighted kappa coefficient unless you specify the COMMON=NO *plot-option*.

To produce a weighted kappa plot, you must specify the AGREE option in the TABLES statement to compute weighted kappa coefficients, and the table dimension must be greater than 1.

Table 47.16 lists the *plot-options* that are available for weighted kappa plots. For descriptions of the *plot-options*, see the subsection "Plot Options" in this section.

Global Plot Options

A *global-plot-option* applies to all plots for which the option is available unless it is altered by an individual *plot-option*. You can specify *global-plot-options* in parentheses after the PLOTS option. For example:

```
plots(order=ascending stats)=(riskdiffplot oddsratioplot)
plots(only)=freqplot
```

The following *plot-options* are available as *global-plot-options*: CLDISPLAY=, COLUMN=, COMMON=, EXACT, LOGBASE=, NPANELPOS=, ORDER=, ORIENT=, RANGE=, SCALE=, STATS, and TYPE=. For descriptions of these *plot-options*, see the subsection "Plot Options" in this section.

In addition to these *plot-options*, you can specify the following *global-plot-option*:

ONLY

suppresses the default plots and requests only the plots that are specified as *plot-requests*.

Plot Options

You can specify the following *plot-options* in parentheses after a *plot-request*:

CL=*type*

specifies the *type* of confidence limits to display. You can specify the CL= *plot-option* when you specify any of the following *plot-requests*: ODDSRATIOPLOT, RELRISKPLOT, and RISKDIFFPLOT.

For odds ratio plots (ODDSRATIOPLOT), the available confidence limit types include the following: exact (CL=EXACT), likelihood ratio (CL=LR), exact mid-*p* (CL=MIDP), score (CL=SCORE), Wald (CL=WALD), and Wald modified (CL=WALDMODIFIED). For more information, see the description of the OR(CL=) option and the section "Confidence Limits for the Odds Ratio" on page 3288. By default, CL=WALD. When you specify CL=EXACT to display exact confidence limits, you must also request computation of exact confidence limits by specifying the OR option in the EXACT statement.

For relative risk plots (RELRISKPLOT), the available confidence limit types include the following: exact (CL=EXACT), likelihood ratio (CL=LR), score (CL=SCORE), Wald (CL=WALD), and Wald modified (CL=WALDMODIFIED). For more information, see the description of the RELRISK(CL=) option and the section "Confidence Limits for the Relative Risk" on page 3291. By default, CL=WALD. When you specify CL=EXACT to display exact confidence limits, you must also request computation of exact confidence limits by specifying the RELRISK option in the EXACT statement.

For risk difference plots (RISKDIFFPLOT), the available confidence limit types include the following: Agresti-Caffo (CL=AC), exact (CL=EXACT), Hauck-Anderson (CL=HA), Miettinen-Nurminen (score) (CL=MN), Newcombe (CL=NEWCOMBE), and Wald (CL=WALD). For more information, see the description of the RISKDIFF(CL=) option and the section "Confidence Limits for the Risk Difference" on page 3274. By default, CL=WALD. When you specify CL=EXACT to display exact confidence limits in the plot, you must also request computation of exact confidence limits by specifying the RISKDIFF option in the EXACT statement.

CLDISPLAY=BAR < *width* **> | LINE | LINEARROW | SERIF | SERIFARROW**

controls the appearance of the confidence limit error bars. You can specify the CLDISPLAY= *plot-option* when you specify the following *plot-requests*: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

By default, CLDISPLAY=SERIF, which displays the confidence limits as lines with serifs. CLDISPLAY=LINE displays the confidence limits as plain lines without serifs. The CLDISPLAY=SERIFARROW and CLDISPLAY=LINEARROW *plot-options* display arrowheads on any error bars that are clipped by the RANGE= *plot-option*; if an entire error bar is cut from the plot, the plot displays an arrowhead that points toward the statistic.

CLDISPLAY=BAR displays the confidence limits as bars. By default, the width of the bars equals the size of the marker for the estimate. You can control the width of the bars and the size of the marker by specifying the value of *width* as a percentage of the distance between bars, $0 < width \leq 1$. The bar might disappear when the value of *width* is very small.

COLUMN=1 | 2

specifies the table column for which to compute the risks (proportions) for the relative risk plot (RELRISKPLOT) and the risk difference plot (RISKDIFFPLOT). If you specify COLUMN=1, the plot displays the column 1 relative risks or risk differences; if you specify COLUMN=2, the plot displays the column 2 relative risks or risk differences.

If you omit this option for the relative risk plot, the plot uses the table column that you specify for the relative risk statistics in the RELRISK(COLUMN=) option. If you omit both of these COLUMN= options, then by default, COLUMN=1 for the relative risk plot.

If you omit the COLUMN= option for the risk difference plot, the plot uses the table column that you specify for the risk difference statistics in the RISKDIFF(COLUMN=) option. If you omit both of these COLUMN= options, then by default, COLUMN=1 for the risk difference plot.

COMMON=NO | YES

controls the display of the common (overall) statistic in plots that display stratum (two-way table) statistics for multiway tables. You can specify the COMMON= *plot-option* when you specify the following *plot-requests*: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

COMMON=NO suppresses display of the common statistic and its confidence limits. By default, COMMON=YES, which displays the common statistic and its confidence limits when these values are available. For more information, see the descriptions of the *plot-requests*.

EXACT

requests display of exact confidence limits instead of asymptotic confidence limits. You can specify the EXACT *plot-option* when you specify the following *plot-requests*: ODDSRATIOPLOT, RELRISKPLOT, and RISKDIFFPLOT. The EXACT *plot-option* is equivalent to the CL=EXACT *plot-option*.

When you specify the EXACT *plot-option*, you must also request computation of exact confidence limits by specifying the appropriate *statistic-option* in the EXACT statement.

GROUPBY=COLUMN | ROW

specifies the primary grouping for two-way frequency plots, which you can request by specifying the FREQPLOT *plot=request*. By default, GROUPBY=COLUMN, which groups graph cells first by column variable and displays row variable levels within column variable levels. You can specify GROUPBY=ROW to group first by row variable. In two-way and multiway table requests, the column variable is the last variable specified and forms the columns of the crosstabulation table. The row variable is the next-to-last variable specified and forms the rows of the table.

By default for a bar chart that is displayed in the TWOWAY=STACKED layout, bars correspond to the column variable levels, and row levels are displayed (stacked) within each column bar. By default for a bar chart that is displayed in the TWOWAY=CLUSTER layout, bars are first grouped by column variable levels, and row levels are displayed as adjacent bars within each column-level group. You can reverse the default row and column variable grouping by specifying GROUPBY=ROW.

LOGBASE=2 | E | 10

applies to the odds ratio plot (ODDSRATIOPLOT) and the relative risk plot (RELRISKPLOT). This *plot-option* displays the odds ratio or relative risk axis on the log scale that you specify.

LEGEND=NO | YES

applies to the agreement plot (AGREEPLOT). LEGEND=NO suppresses the legend that identifies the areas of exact and partial agreement. By default, LEGEND=YES.

NOSTAT

applies to the deviation plot (DEVIATIONPLOT). NOSTAT suppresses the chi-square *p*-value that deviation plot displays by default.

NPANELPOS=*n*

divides the plot into multiple panels that display at most |*n*| statistics or sections.

If *n* is positive, the number of statistics or sections per panel is balanced; if *n* is negative, the number of statistics per panel is not balanced. For example, suppose you want to display 21 odds ratios. NPANELPOS=20 displays two panels, the first with 11 odds ratios and the second with 10 odds ratios; NPANELPOS=–20 displays 20 odds ratios in the first panel but only 1 odds ratio in the second panel. This *plot-option* is available for all plots except mosaic plots and one-way weighted frequency plots.

For two-way frequency plots (FREQPLOT), NPANELPOS=*n* requests that panels display at most |*n*| sections, where sections correspond to row or column variable levels, depending on the type of plot and the grouping. By default, *n*=4 and each panel includes at most four sections. This *plot-option* applies to two-way plots that are displayed in the TWOWAY=GROUPVERTICAL or TWOWAY=GROUPHORIZONTAL layout. The NPANELPOS= *plot-option* does not apply to the TWOWAY=CLUSTER and TWOWAY=STACKED layouts, which are always displayed in a single panel.

For plots that display statistics along with confidence limits, NPANELPOS=*n* requests that panels display at most |*n*| statistics. By default, *n*=0 and all statistics are displayed in a single panel. This *plot-option* applies to the following plots: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

ORDER=ASCENDING | DESCENDING

displays the two-way table (strata) statistics in order of the statistic value. You can specify the ORDER= *plot-option* when you specify the following *plot-requests*: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

If you specify ORDER=ASCENDING or ORDER=DESCENDING, the plot displays the statistics in ascending or descending order, respectively. By default, the order of the statistics in the plot matches the order that the two-way table strata appear in the multiway table display.

ORIENT=HORIZONTAL | VERTICAL

controls the orientation of the plot. You can specify the ORIENT= *plot-option* when you specify the following *plot-requests*: CUMFREQPLOT, DEVIATIONPLOT, and FREQPLOT.

ORIENT=HORIZONTAL places the variable levels on the Y axis and the frequencies, percentages, or statistic values on the X axis. ORIENT=VERTICAL places the variable levels on the X axis. The default orientation is ORIENT=VERTICAL for bar charts (TYPE=BARCHART) and ORIENT=HORIZONTAL for dot plots (TYPE=DOTPLOT).

PARTIAL=NO | YES

controls the display of partial agreement in the agreement plot (AGREEPLOT). PARTIAL=NO suppresses the display of partial agreement. When you specify PARTIAL=NO, the agreement plot displays only exact agreement. Exact agreement includes the diagonal cells of the square table, where the row and column variable levels are the same. Partial agreement includes the adjacent off-diagonal table cells, where the row and column values are within one level of exact agreement. By default, PARTIAL=YES.

RANGE=(< *min*> < **,** *max*> **)| CLIP**

specifies the range of values to display. You can specify the RANGE= *plot-option* when you specify the following *plot-requests*: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

If you specify RANGE=CLIP, the confidence limits are clipped and the display range is determined by the minimum and maximum values of the statistics. By default, the display range includes all confidence limits.

SCALE=FREQ | GROUPPERCENT | LOG | PERCENT | SQRT

specifies the scale of the frequencies to display. This *plot-option* is available for frequency plots (FREQPLOT) and cumulative frequency plots (CUMFREQPLOT).

By default, SCALE=FREQ, which displays unscaled frequencies. SCALE=PERCENT displays percentages (relative frequencies) of the total frequency. SCALE=LOG displays log (base 10) frequencies. SCALE=SQRT displays square roots of the frequencies, producing a plot known as a *rootogram*.

SCALE=GROUPPERCENT is available for two-way frequency plots. This option displays the row or column percentages instead of the overall percentages (of the table frequency). By default (or when you specify the GROUPBY=COLUMN *plot-option*), SCALE=GROUPPERCENT displays the column percentages. If you specify the GROUPBY=ROW *plot-option*, the primary grouping of graph cells is by row variable level and the plot displays row percentages. For more information, see the description of the GROUPBY= *plot-option*.

SHOWSCALE=NO | YES

controls the display of the cumulative frequency scale on the right side of the agreement plot (AGREEPLOT). SHOWSCALE=NO suppresses the display of the scale. By default, SHOWSCALE=YES.

STATS

displays statistic values in the plot. For the following *plot-requests*, the STATS *plot-option* displays the statistics and their confidence limits on the right side of the plot: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

For the agreement plot (AGREEPLOT), the STATS *plot-option* displays the values of the kappa statistic, the weighted kappa statistic, the B_n measure (Bangdiwala and Bryan 1987), and the sample size. PROC FREQ stores these statistics in an ODS table named BnMeasure, which is not displayed. For more information, see the section "ODS Table Names" on page 3336.

If you do not request the STATS *plot-option*, these plots do not display the statistic values.

TWOWAY=CLUSTER | GROUPHORIZONTAL | GROUPVERTICAL | STACKED

specifies the layout for two-way frequency plots.

All TWOWAY= layouts are available for bar charts (TYPE=BARCHART). All TWOWAY= layouts except TWOWAY=CLUSTER are available for dot plots (TYPE=DOTPLOT). The ORIENT= and GROUPBY= *plot-options* are available for all TWOWAY= layouts.

The default two-way layout is TWOWAY=GROUPVERTICAL, which produces a grouped plot that has a vertical common baseline. By default for bar charts (TYPE=BARCHART, ORIENT=VERTICAL), the X axis displays column variable levels, and the Y axis displays frequencies. The plot includes a vertical (Y-axis) block for each row variable level. The relative positions of the graph cells in this plot layout are the same as the relative positions of the table cells in the crosstabulation table. You can reverse the default row and column grouping by specifying the GROUPBY=ROW *plot-option*.

The TWOWAY=GROUPHORIZONTAL layout produces a grouped plot that has a horizontal common baseline. By default (GROUPBY=COLUMN), the plot displays a block on the X axis for each column variable level. Within each column-level block, the plot displays row variable levels.

The TWOWAY=STACKED layout produces stacked displays of frequencies. By default (GROUPBY=COLUMN) in a stacked bar chart, the bars correspond to column variable levels, and row levels are stacked within each column level. By default in a stacked dot plot, the dotted lines correspond to column levels, and cell frequencies are plotted as data dots on the corresponding column line. The dot color identifies the row level.

The TWOWAY=CLUSTER layout, which is available only for bar charts, displays groups of adjacent bars. By default, the primary grouping is by column variable level, and row levels are displayed within each column level.

You can reverse the default row and column grouping in any layout by specifying the GROUPBY=ROW *plot-option*. By default, GROUPBY=COLUMN, which groups first by column variable.

TYPE=BARCHART | DOTPLOT

specifies the plot type (format) of the frequency (FREQPLOT), cumulative frequency (CUMFREQPLOT), and deviation plots (DEVIATIONPLOT). TYPE=BARCHART produces a bar chart and TYPE=DOTPLOT produces a dot plot. By default, TYPE=BARCHART.

PRINTKWTS

displays the agreement weights that PROC FREQ uses to compute the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. By default, PROC FREQ uses the Cicchetti-Allison form of agreement weights. If you specify the AGREE(WT=FC) option, the procedure uses the Fleiss-Cohen form of agreement weights. For more information, see the section "Weighted Kappa Coefficient" on page 3303.

This option has no effect unless you also specify the AGREE option to compute the weighted kappa coefficient. The PRINTKWTS option is equivalent to the AGREE(PRINTKWTS) option.

RELRISK < (*relrisk-options*) >

requests relative risk measures and their confidence limits for 2 × 2 tables. These measures include the odds ratio, the column 1 relative risk, and the column 2 relative risk. For more information, see the section "Odds Ratio and Relative Risks" on page 3287. By default, PROC FREQ displays the relative risk measures and their asymptotic Wald confidence limits in the "Odds Ratio and Relative Risks" table. You can also obtain this table by specifying the MEASURES option, which produces other measures of association in addition to the relative risks.

You can specify *relrisk-options* in parentheses after the RELRISK option to request tests and additional confidence limits for the column 1 or column 2 relative risk. Table 47.18 summarizes the *relrisk-options*.

When you request tests or additional confidence limit types for the relative risk, PROC FREQ does not display the "Odds Ratio and Relative Risks" table unless you also specify the PRINTALL *relrisk-option*.

Table 47.18 RELRISK (Relative Risk) Options

Option	Description
COLUMN=1 | 2	Specifies the risk column
PRINTALL	Displays "Odds Ratio and Relative Risks" table
Request Confidence Limits	
CL=EXACT	Displays exact confidence limits
CL=LR	Requests likelihood ratio confidence limits
CL=SCORE	Requests score confidence limits
CL=WALD	Requests Wald confidence limits
CL=WALDMODIFIED	Requests Wald modified confidence limits
Request Tests	
EQUAL(NULL=)	Requests an equality test
EQUIV | EQUIVALENCE	Requests an equivalence test
MARGIN=	Specifies the test margin
METHOD=	Specifies the test method
NONINF | NONINFERIORITY	Requests a noninferiority test
SUP | SUPERIORITY	Requests a superiority test

You can specify the following *relrisk-options*:

CL=*type* | (*types*)

specifies confidence limit types for the relative risk. You can specify one or more *types* of confidence limits. When you specify only one *type*, you can omit the parentheses around the request. When you specify the CL= *relrisk-option*, PROC FREQ displays the confidence limits in the "Confidence Limits for the Relative Risk" table.

The ALPHA= option determines the level of the confidence limits that the CL= *relrisk-option* provides. By default, ALPHA=0.05, which produces 95% confidence limits for the relative risk.

You can specify the following *types*:

EXACT

displays exact unconditional confidence limits for the relative risk in the "Confidence Limits for the Relative Risk" table. (By default, PROC FREQ displays the exact confidence limits in a separate table.) You must also request computation of the exact confidence limits by specifying the RELRISK option in the EXACT statement. For more information, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291.

LR

LIKELIHOOD RATIO

requests likelihood ratio confidence limits for the relative risk. For more information, see the subsection "Likelihood Ratio Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291.

SCORE < (CORRECT=NO) >

requests score confidence limits for the relative risk. For more information, see the subsection "Score Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291. If you specify CORRECT=NO, PROC FREQ provides the uncorrected form of the confidence limits.

WALD

requests asymptotic Wald confidence limits, which are based on a log transformation of the relative risk. For more information, see the subsection "Wald Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291.

WALDMODIFIED

requests Wald modified confidence limits for the relative risk. For more information, see the subsection "Wald Modified Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291.

COLUMN=1 | 2

specifies the table column for which to compute the relative risk confidence limits (which you request by specifying the CL= *relrisk-option*) and the relative risk tests (EQUAL, EQUIV, NONINF, and SUP).

This option has no effect on the "Odds Ratio and Relative Risks" table, which displays both column 1 and column 2 relative risks.

By default, COLUMN=1 unless you specify the column for the relative risk plot by using the RELRISKPLOT(COLUMN=) option. When you specify the column for the plot (and omit the RELRISK(COLUMN=) option), PROC FREQ uses the same column for both the plot and the relative risk statistics.

EQUAL < (NULL=*value*)>

requests an equality test for the relative risk. For more information, see the subsection "Equality Test" in the section "Relative Risk Tests" on page 3294.

You can specify the null hypothesis *value* of the relative risk in the NULL= option. The null *value* must be a positive number. By default, NULL=1. You can specify the equality test method in the METHOD= *relrisk-option*. By default, PROC FREQ produces a Wald test (METHOD=WALD).

EQUIV

EQUIVALENCE

requests an equivalence test for the relative risk. For more information, see the subsection "Equivalence Test" in the section "Relative Risk Tests" on page 3294. You can specify the test method in the METHOD= *relrisk-option*, and you can specify the test margins in the MARGIN= *relrisk-option*. By default, METHOD=WALD and MARGIN=(0.8,1.25).

MARGIN=*value* | **(***lower***,** *upper***)**

specifies the margin for the noninferiority, superiority, and equivalence tests, which you request by specifying the NONINF, SUP, and EQUIV *relrisk-options*, respectively. By default, MARGIN=0.8 for noninferiority tests, MARGIN=1.25 for superiority tests, and MARGIN=(0.8,1.25) for equivalence tests.

For noninferiority and superiority tests, specify a single *value* in the MARGIN= option. The *value* must be a positive number. For a noninferiority test, the *value* should be less than 1; for a superiority test, the *value* should be greater than 1.

For an equivalence test, you can specify a single MARGIN= *value*, or you can specify both *lower* and *upper* values. All values must be positive numbers. If you specify a single *value*, PROC FREQ uses *value* as the lower margin and the inverse of *value* as the upper margin. If you specify both *lower* and *upper* values, the value of *lower* must be less than the value of *upper*.

METHOD=*method*

specifies the method to be used for the equality, equivalence, noninferiority, and superiority tests, which you request by specifying the EQUAL, EQUIV, NONINF, and SUP *relrisk-options*, respectively. By default, METHOD=WALD.

You can specify one of the following *methods*:

FM

SCORE

requests Farrington-Manning (score) tests for the equality, equivalence, noninferiority, and superiority analyses of the relative risk. For more information, see the subsection "Farrington-Manning (Score) Test" in the section "Relative Risk Tests" on page 3294.

LR

LIKELIHOODRATIO

requests likelihood ratio tests for the equality, equivalence, noninferiority, and superiority analyses of the relative risk. For more information, see the subsection "Likelihood Ratio Test" in the section "Relative Risk Tests" on page 3294.

WALD

requests Wald tests for the equality, equivalence, noninferiority, and superiority analyses of the relative risk. For more information, see the subsection "Wald Test" in the section "Relative Risk Tests" on page 3294.

WALDMODIFIED

requests Wald modified tests for the equality, equivalence, noninferiority, and superiority analyses of the relative risk. For more information, see the subsection "Wald Modified Test" in the section "Relative Risk Tests" on page 3294.

NONINF

NONINFERIORITY

requests a noninferiority test for the relative risk. For more information, see the subsection "Noninferiority Test" in the section "Relative Risk Tests" on page 3294. You can specify the test method in the METHOD= *relrisk-option*, and you can specify the margin in the MARGIN= *relrisk-option*. By default, METHOD=WALD and MARGIN=0.8.

PRINTALL

displays the "Odds Ratio and Relative Risks" table when you request tests or additional confidence limits by specifying *relrisk-options*. By default, PROC FREQ does not display this table when you request tests or additional confidence limits for the relative risk.

SUP

SUPERIORITY

requests a superiority test for the relative risk. For more information, see the subsection "Superiority Test" in the section "Relative Risk Tests" on page 3294. You can specify the test method in the METHOD= *relrisk-option*, and you can specify the margin in the MARGIN= *relrisk-option*. By default, METHOD=WALD and MARGIN=1.25.

RISKDIFF < (*riskdiff-options*) >

requests risks (binomial proportions) and risk differences for 2×2 tables. By default, this option provides the row 1 risk, row 2 risk, total (overall) risk, and risk difference (row 1 – row 2), together with their asymptotic standard errors and Wald confidence limits; by default, this option also provides exact (Clopper-Pearson) confidence limits for the row 1, row 2, and total risks. You can request exact unconditional confidence limits for the risk difference by specifying the RISKDIFF option in the EXACT statement. PROC FREQ displays these results in the column 1 and column 2 "Risk Estimates" tables (which you can suppress by specifying the NORISKS *riskdiff-option*).

You can specify *riskdiff-options* in parentheses after the RISKDIFF option to request tests and additional confidence limits for the risk difference, in addition to estimates of the common risk difference for multiway 2×2 tables. Table 47.19 summarizes the *riskdiff-options*.

The CL= *riskdiff-option* requests confidence limits for the risk difference. Available confidence limit types include Agresti-Caffo, exact unconditional, Hauck-Anderson, Miettinen-Nurminen (score), Newcombe, and Wald. Continuity-corrected Newcombe and Wald confidence limits are also available. You can request more than one type of confidence limits in the same analysis. PROC FREQ displays the confidence limits in the "Confidence Limits for the Risk Difference" table.

The CL=EXACT *riskdiff-option* displays exact unconditional confidence limits in the "Confidence Limits for the Risk Difference" table. When you specify CL=EXACT, you must also request computation of the exact confidence limits by specifying the RISKDIFF option in the EXACT statement.

The EQUAL, EQUIV, NONINF, and SUP *riskdiff-options* request tests of equality, equivalence, noninferiority, and superiority, respectively, for the risk difference. Available test methods include Farrington-Manning (score), Hauck-Anderson, and Wald. Newcombe (hybrid-score) confidence limits are available for the equivalence, noninferiority, and superiority analyses.

As part of the noninferiority, superiority, and equivalence analyses, PROC FREQ provides null-based equivalence limits that have a confidence coefficient of $100(1 - 2\alpha)\%$ (Schuirmann 1999). The ALPHA= option determines the confidence level; by default, ALPHA=0.05, which produces 90% equivalence limits for these analyses. For more information, see the sections "Noninferiority Tests" on page 3279 and "Equivalence Test" on page 3281.

Table 47.19 RISKDIFF (Proportion Difference) Options

Option	Description
COLUMN=1 | 2	Specifies the risk column
COMMON	Requests common risk difference
CORRECT	Requests continuity correction
NORISKS	Suppresses default risk tables
Request Confidence Limits	
CL=AC	Requests Agresti-Caffo confidence limits
CL=EXACT	Displays exact confidence limits
CL=HA	Requests Hauck-Anderson confidence limits
CL=MN | SCORE	Requests Miettinen-Nurminen confidence limits
CL=NEWCOMBE	Requests Newcombe confidence limits
CL=WALD	Requests Wald confidence limits
Request Tests	
EQUAL(NULL=)	Requests an equality test
EQUIV | EQUIVALENCE	Requests an equivalence test
MARGIN=	Specifies the test margin
METHOD=	Specifies the test method
NONINF | NONINFERIORITY	Requests a noninferiority test
SUP | SUPERIORITY	Requests a superiority test
VAR=SAMPLE | NULL	Specifies the test variance

You can specify the following *riskdiff-options*:

CL=*type* **|** **(***types***)**

requests confidence limits for the risk difference. You can specify one or more *types* of confidence limits. When you specify only one *type*, you can omit the parentheses around the request. PROC FREQ displays the confidence limits in the "Confidence Limits for the Risk Difference" table.

The ALPHA= option determines the level of the confidence limits. By default, ALPHA=0.05, which produces 95% confidence limits for the risk difference.

You can specify the CL= *riskdiff-option* with or without requests for risk difference tests. The confidence limits that CL= produces do not depend on the tests that you request and do not use the value of the test margin (which you can specify in the MARGIN= *riskdiff-option*).

You can specify the following *types*:

AC

AGRESTICAFFO

requests Agresti-Caffo confidence limits for the risk difference. For more information, see the subsection "Agresti-Caffo Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

EXACT

displays exact unconditional confidence limits for the risk difference in the "Confidence Limits for the Risk Difference" table. You must also request computation of the exact confidence limits by specifying the RISKDIFF option in the EXACT statement.

By default, PROC FREQ computes the exact confidence limits by inverting two separate one-sided exact tests that are based on the score statistic. For more information, see the RISKDIFF option in the EXACT statement and the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

By default, PROC FREQ also displays these exact confidence limits in the "Risk Estimates" table. You can suppress this table by specifying the NORISKS *riskdiff-option*.

HA

requests Hauck-Anderson confidence limits for the risk difference. For more information, see the subsection "Hauck-Anderson Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

MN < (CORRECT=NO | MEE) >

SCORE < (CORRECT=NO | MEE) >

requests Miettinen-Nurminen (score) confidence limits for the risk difference. For more information, see the subsection "Miettinen-Nurminen (Score) Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274. By default, the Miettinen-Nurminen confidence limits include a bias correction factor (Miettinen and Nurminen 1985; Newcombe and Nurminen 2011). If you specify CL=MN(CORRECT=NO), PROC FREQ provides the uncorrected form of the confidence limits (Mee 1984).

NEWCOMBE < (CORRECT) >

requests Newcombe hybrid-score confidence limits for the risk difference. If you specify CL=NEWCOMBE(CORRECT) or the CORRECT *riskdiff-option*, the Newcombe confidence limits include a continuity correction. For more information, see the subsection "Newcombe Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

WALD < (CORRECT) >

requests Wald confidence limits for the risk difference. If you specify CL=WALD(CORRECT) or the CORRECT *riskdiff-option*, the Wald confidence limits include a continuity correction. For more information, see the subsection "Wald Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

COLUMN=1 | 2 | BOTH

specifies the table column for which to compute the risk difference confidence limits (which you request by specifying the CL= *riskdiff-option*), the risk difference tests (EQUAL, EQUIV, NONINF, and SUP), and the common risk difference statistics (COMMON). COLUMN=BOTH does not apply to the common risk difference statistics.

By default, COLUMN=1 unless you specify the column for the risk difference plot by using the RISKDIFFPLOT(COLUMN=) option. When you specify the column for the plot (and omit the RISKDIFF(COLUMN=) option), PROC FREQ uses the same column for both the plot and the risk difference statistics.

This option has no effect on the "Risk Estimates" table, which is produced for both column 1 and column 2. You can suppress the "Risk Estimates" table by specifying the NORISKS *riskdiff-option*.

COMMON

requests estimates of the common (overall) risk difference for multiway 2×2 tables. This option provides Mantel-Haenszel and summary score estimates for the common risk difference, together with their confidence limits. If you specify the RISKDIFF(CL=NEWCOMBE) option, the RISKDIFF(COMMON) option also provides Newcombe confidence limits for the common risk difference. For more information, see the section "Common Risk Difference" on page 3283.

You can use the COMMONRISKDIFF option to request additional confidence limits and tests for the common risk difference.

CORRECT

includes a continuity correction in the Wald confidence limits, Wald tests, and Newcombe confidence limits. For more information, see the section "Risks and Risk Differences" on page 3272.

EQUAL < (NULL=*value*)>

requests an equality test for the risk difference. For more information, see the section "Equality Tests" on page 3278.

You can specify the null hypothesis *value* of the risk difference in the NULL= option. By default, NULL=0. You can specify the null *value* in proportion form as a number between –1 and 1, or you can specify the null *value* in percentage form as a number between –100 and 100. When the *value* is between –100 and –1 or between 1 and 100, PROC FREQ converts the number to a proportion. PROC FREQ treats the values –1 and 1 as percentages.

You can specify the equality test method by using the METHOD= *riskdiff-option*. By default, PROC FREQ produces a Wald test (METHOD=WALD). By default, PROC FREQ uses the sample variance to compute the Wald test statistic; if you specify the VAR=NULL *riskdiff-option*, PROC FREQ uses the null (test-based) variance.

EQUIV

EQUIVALENCE

requests an equivalence test for the risk difference. For more information, see the section "Equivalence Test" on page 3281. You can specify the test method in the METHOD= *riskdiff-option*, and you can specify the margins in the MARGIN= *riskdiff-option*. By default, METHOD=WALD and MARGIN=0.2.

MARGIN=*value* | (*lower*, *upper*)

specifies the margin for the noninferiority, superiority, and equivalence tests, which you request by specifying the NONINF, SUP, and EQUIV *riskdiff-options*, respectively. By default, MARGIN=0.2.

For noninferiority and superiority tests, specify a single *value* in the MARGIN= option. The *value* must be a positive number. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC FREQ converts that number to a proportion. PROC FREQ treats the value 1 as 1%.

For an equivalence test, you can specify a single MARGIN= *value*, or you can specify both *lower* and *upper* values. If you specify a single *value*, it must be a positive number, as described previously. If you specify a single *value* for an equivalence test, PROC FREQ uses –*value* as the lower margin and *value* as the upper margin for the test. If you specify both *lower* and *upper* values for an equivalence test, you can specify them in proportion form as numbers between –1 and 1. Or you can specify them in percentage form as numbers between –100 and 100, and PROC FREQ converts the numbers to proportions. The value of *lower* must be less than the value of *upper*.

METHOD=*method*

specifies the method to be used for the equality, equivalence, noninferiority, and superiority tests, which you request by specifying the EQUAL, EQUIV, NONINF, and SUP *riskdiff-options*, respectively. By default, METHOD=WALD.

You can specify the following *methods*:

FM

SCORE

requests Farrington-Manning (score) tests for the equality, equivalence, noninferiority, and superiority analyses. For more information, see the subsection "Farrington-Manning (Score) Test" in the section "Noninferiority Tests" on page 3279.

HA

requests Hauck-Anderson tests for the equality, equivalence, noninferiority, and superiority analyses. For more information, see the subsection "Hauck-Anderson Test" in the section "Noninferiority Tests" on page 3279.

NEWCOMBE

requests Newcombe (hybrid-score) confidence limits for the equivalence, noninferiority, and superiority analyses. If you specify the CORRECT *riskdiff-option*, the Newcombe confidence limits include a continuity correction. For more information, see the subsection "Newcombe Noninferiority Analysis" in the section "Noninferiority Tests" on page 3279.

WALD

requests Wald tests for the equality, equivalence, noninferiority, and superiority analyses. By default, PROC FREQ uses the sample variance to compute these Wald test statistics; if you specify the VAR=NULL *riskdiff-option*, PROC FREQ uses the null (test-based) variance. If you specify the CORRECT *riskdiff-option*, the Wald tests and confidence limits include a continuity correction. For more information, see the subsection "Wald Test" in the section "Noninferiority Tests" on page 3279.

NONINF

NONINFERIORITY

requests a noninferiority test for the risk difference. For more information, see the section "Noninferiority Tests" on page 3279. You can specify the test method in the METHOD= *riskdiff-option*, and you can specify the margin in the MARGIN= *riskdiff-option*. By default, METHOD=WALD and MARGIN=0.2.

NORISKS

suppresses display of the "Risk Estimates" tables, which the RISKDIFF option produces by default for column 1 and column 2. The "Risk Estimates" tables contain the risks and risk differences, together with their asymptotic standard errors, Wald confidence limits, and exact confidence limits.

SUP

SUPERIORITY

requests a superiority test for the risk difference. For more information, see the section "Superiority Test" on page 3281. You can specify the test method in the METHOD= *riskdiff-option*, and you can specify the margin in the MARGIN= *riskdiff-option*. By default, METHOD=WALD and MARGIN=0.2.

VAR=NULL | SAMPLE

specifies the type of variance to use in the Wald tests of equality, equivalence, noninferiority, and superiority. By default (or if you specify VAR=SAMPLE), PROC FREQ uses the sample variance to compute the Wald test statistics. If you specify VAR=NULL, PROC FREQ uses the null (test-based) variance. For more information, see the sections "Equality Tests" on page 3278 and "Noninferiority Tests" on page 3279.

SCORES=*type*

specifies the type of row and column scores that PROC FREQ uses to compute the following statistics: Mantel-Haenszel chi-square, Pearson correlation, Cochran-Armitage test for trend, weighted kappa coefficient, and Cochran-Mantel-Haenszel statistics. The value of *type* can be one of the following:

- **MODRIDIT**
- **RANK**
- **RIDIT**
- **TABLE**

See the section "Scores" on page 3247 for descriptions of these score types.

If you do not specify the SCORES= option, PROC FREQ uses SCORES=TABLE by default. For character variables, the row and column TABLE scores are the row and column numbers. That is, the TABLE score is 1 for row 1, 2 for row 2, and so on. For numeric variables, the row and column TABLE scores equal the variable values. For more information, see the section "Scores" on page 3247. Using MODRIDIT, RANK, or RIDIT scores yields nonparametric analyses.

You can use the SCOROUT option to display the row and column scores.

SCOROUT

displays the row and column scores that PROC FREQ uses to compute score-based tests and statistics. You can specify the score type by using the SCORES= option. For more information, see the section "Scores" on page 3247.

The scores are computed and displayed only when PROC FREQ computes statistics for two-way tables. You can use ODS to store the scores in an output data set. See the section "ODS Table Names" on page 3336 for more information.

SENSPEC < (*options*) >

requests estimates of sensitivity, specificity, accuracy, positive predictive value, and negative predictive value for 2×2 tables. The "Sensitivity and Specificity" table provides these estimates together with their standard errors and confidence limits. For more information, see the section "Sensitivity and Specificity" on page 3271.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

You can specify the following *options*:

NOTE< =YES | NO >

controls the footnote that identifies the reference level (true positives) in the "Sensitivity and Specificity" table. NOTE=NO suppresses this footnote and NOTE< =YES > displays it. By default, PROC FREQ displays the footnote only when you specify the reference level by using the REFCELL=, REFCOLUMN=, and/or REFROW= options.

REFCELL=*number*

specifies the 2×2 table cell to use as the true-positive classification (positive row variable level and positive column variable level) in the SENSPEC computations. For more information, see the section "Sensitivity and Specificity" on page 3271.

You can specify the true-positive cell *number* as 1, 2, 3, or 4; these numbers correspond to the cells of the 2×2 crosstabulation table. By default, REFCELL=1, which uses table cell (1,1) as the true-positive cell (and the corresponding row 1 and column 1 as the positive row and column levels, respectively). REFCELL=2, 3, and 4 use table cells (1,2), (2,1), and (2,2), respectively, as the true-positive cells.

You cannot specify this option together with the REFROW= or REFCOLUMN= option. The true-positive cell *number* determines the corresponding positive row and column levels.

REFCOLUMN=*number* | '*level-value*'

specifies the column level to use as the positive column level in the SENSPEC computations. For more information, see the section "Sensitivity and Specificity" on page 3271.

You can specify the positive column level *number* as 1 or 2, which is the order in which the level appears in the 2×2 crosstabulation table. Or you can specify the positive column *level-value*, which is the formatted value of the level. The *level-value* must be enclosed in single quotes. By default, REFCOLUMN=1.

REFROW=*number* | '*level-value*'

specifies the row level to use as the positive row level in the SENSPEC computations. For more information, see the section "Sensitivity and Specificity" on page 3271.

You can specify the positive row level *number* as 1 or 2, which is the order in which the level appears in the 2×2 crosstabulation table. Or you can specify the positive row *level-value*, which is the formatted value of the level. The *level-value* must be enclosed in single quotes. By default, REFROW=1.

SPARSE

reports all possible combinations of variable values in two-way and multiway tables, even if a combination does not occur in the data. This option applies only to crosstabulation tables that are displayed in LIST format and to OUT= output data sets.

When you specify the SPARSE option together with the LIST option, the LIST tables display all combinations of variable values, including levels that have a frequency of 0. By default, LIST tables do not display zero-frequency levels. When you specify the SPARSE option together with the OUT= option, the OUT= output data set includes empty (zero-frequency) crosstabulation table cells. By default, the OUT= output data set does not include zero-frequency table cells.

For more information, see the section "Missing Values" on page 3243.

TOTPCT

displays the percentage of the total multiway table frequency in multiway crosstabulation tables (n-way tables, where $n > 2$). By default, crosstabulation tables display percentages of the two-way table (stratum) frequency but do not display percentages of the total multiway table frequency. For more information, see the section "Two-Way and Multiway Tables" on page 3328.

By default, tables in list format (which you can request by specifying the LIST option) display the percentages of the total multiway table frequency. The variable PERCENT in the OUT= output data set also provides the percentages of the total multiway table frequency.

TREND

requests the Cochran-Armitage test for trend. The table must be $2 \times C$ or $R \times 2$ to compute the trend test. For more information, see the section "Cochran-Armitage Test for Trend" on page 3297. To request exact p-values for the trend test, specify the TREND option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

TEST Statement

TEST *test-options* **;**

The TEST statement requests asymptotic tests for measures of association and measures of agreement. The *test-options* identify which tests to compute. Table 47.20 lists the available *test-options*, together with their corresponding TABLES statement options. Descriptions of the *test-options* follow the table in alphabetical order.

For each measure of association or agreement that you request in the TEST statement, PROC FREQ provides an asymptotic test that the measure is 0. The procedure displays the asymptotic standard error under the null hypothesis, the test statistic, and the one-sided and two-sided p-values. PROC FREQ also provides

confidence limits for the measure. The ALPHA= option in the TABLES statement determines the confidence level; by default, ALPHA=0.05, which provides 95% confidence limits. For more information, see the sections "Asymptotic Tests" on page 3254 and "Confidence Limits" on page 3254. For information about the individual measures, see the sections "Measures of Association" on page 3253 and "Tests and Measures of Agreement" on page 3300.

You can also request exact tests for selected measures of association and agreement by using the EXACT statement. For more information, see the section "Exact Statistics" on page 3318.

Using the TEST Statement with the TABLES Statement

You must use a TABLES statement with the TEST statement. If you use only one TABLES statement, you do not need to specify the same options in both the TABLES and TEST statements; when you specify an option in the TEST statement, PROC FREQ automatically invokes the corresponding TABLES statement option. However, when you use the TEST statement with multiple TABLES statements, you must specify options in the TABLES statements to request statistics; PROC FREQ then provides asymptotic tests for those statistics that you specify in the TEST statement.

Table 47.20 TEST Statement Options

Test Option	Asymptotic Tests	Required TABLES Statement Option
AGREE	Simple and weighted kappa coefficients	AGREE
GAMMA	Gamma	ALL or MEASURES
KAPPA	Simple kappa coefficient	AGREE
KENTB | TAUB	Kendall's tau-*b*	ALL or MEASURES
MEASURES	Gamma, Kendall's tau-*b*, Stuart's tau-*c*, Somers' $D(C\|R)$, Somers' $D(R\|C)$, Pearson and Spearman correlations	ALL or MEASURES
PCORR	Pearson correlation coefficient	ALL or MEASURES
PLCORR	Polychoric correlation	PLCORR
SCORR	Spearman correlation coefficient	ALL or MEASURES
SMDCR	Somers' $D(C\|R)$	ALL or MEASURES
SMDRC	Somers' $D(R\|C)$	ALL or MEASURES
STUTC | TAUC	Stuart's tau-*c*	ALL or MEASURES
WTKAPPA | WTKAP	Weighted kappa coefficient	AGREE

You can specify the following *test-options*:

AGREE

requests asymptotic tests for the simple kappa coefficient and the weighted kappa coefficient. For more information, see the sections "Simple Kappa Coefficient" on page 3301 and "Weighted Kappa Coefficient" on page 3303.

By default, these tests are based on null values of 0; you can specify nonzero null values for the simple kappa and weighted kappa tests by using the AGREE(NULLKAPPA=) and AGREE(NULLWTKAPPA=) options, respectively, in the TABLES statement.

The AGREE option in the TABLES statement provides estimates, standard errors, and confidence

limits for kappa coefficients. You can request exact tests for kappa coefficients by using the EXACT statement.

Kappa coefficients are defined only for square tables, where the number of rows equals the number of columns. Kappa coefficients are not computed for tables that are not square. For 2×2 tables, the weighted kappa coefficient is identical to the simple kappa coefficient, and PROC FREQ presents only the simple kappa coefficient.

GAMMA

requests an asymptotic test for the gamma statistic. For more information, see the section "Gamma" on page 3255. The MEASURES option in the TABLES statement provides the gamma statistic and its asymptotic standard error.

KAPPA

requests an asymptotic test for the simple kappa coefficient. For more information, see the section "Simple Kappa Coefficient" on page 3301.

By default, the null value of kappa for this test is 0; you can specify a nonzero null value by using the AGREE(NULLKAPPA=) option in the TABLES statement.

The AGREE option in the TABLES statement provides the kappa statistic, its standard error, and its confidence limits. You can request an exact test for the simple kappa coefficient by specifying the KAPPA option in the EXACT statement.

Kappa coefficients are defined only for square tables, where the number of rows equals the number of columns. PROC FREQ does not compute kappa coefficients for tables that are not square.

KENTB

TAUB

requests an asymptotic test for Kendall's tau-*b*. For more information, see the section "Kendall's Tau-b" on page 3255.

The MEASURES option in the TABLES statement provides Kendall's tau-*b* and its standard error. You can request an exact test for Kendall's tau-*b* by specifying the KENTB option in the EXACT statement.

MEASURES

requests asymptotic tests for the following measures of association: gamma, Kendall's tau-*b*, Pearson correlation coefficient, Somers' $D(C|R)$, Somers' $D(R|C)$, Spearman correlation coefficient, and Stuart's tau-*c*. For more information, see the section "Measures of Association" on page 3253.

The MEASURES option in the TABLES statement provides measures of association and their asymptotic standard errors. You can request exact tests for selected measures by using the EXACT statement.

PCORR

requests an asymptotic test for the Pearson correlation coefficient. For more information, see the section "Pearson Correlation Coefficient" on page 3257.

The MEASURES option in the TABLES statement provides the Pearson correlation and its standard error. You can request an exact test for the Pearson correlation by specifying the PCORR option in the EXACT statement.

PLCORR

requests Wald and likelihood ratio tests for the polychoric correlation coefficient. For more information, see the section "Polychoric Correlation" on page 3260.

The PLCORR option in the TABLES statement provides the polychoric correlation and its standard error.

SCORR

requests an asymptotic test for the Spearman correlation coefficient. For more information, see the section "Spearman Rank Correlation Coefficient" on page 3258.

The MEASURES option in the TABLES statement provides the Spearman correlation and its standard error. You can request an exact test for the Spearman correlation by specifying the SCORR option in the EXACT statement.

SMDCR

requests an asymptotic test for Somers' $D(C|R)$. For more information, see the section "Somers' D" on page 3257.

The MEASURES option in the TABLES statement provides Somers' $D(C|R)$ and its standard error. You can request an exact test for Somers' $D(C|R)$ by specifying the SMDCR option in the EXACT statement.

SMDRC

requests an asymptotic test for Somers' $D(R|C)$. For more information, see the section "Somers' D" on page 3257.

The MEASURES option in the TABLES statement provides Somers' $D(R|C)$ and its standard error. You can request an exact test for Somers' $D(R|C)$ by specifying the SMDRC option in the EXACT statement.

STUTC

TAUC

requests an asymptotic test for Stuart's tau-c. For more information, see the section "Stuart's Tau-c" on page 3256.

The MEASURES option in the TABLES statement provides Stuart's tau-c and its standard error. You can request an exact test for Stuart's tau-c by specifying the STUTC option in the EXACT statement.

WTKAPPA

WTKAP

requests an asymptotic test for the weighted kappa coefficient. For more information, see the section "Weighted Kappa Coefficient" on page 3303.

By default, the null value of weighted kappa for this test is 0; you can specify a nonzero null value by using the AGREE(NULLWTKAPPA=) option in the TABLES statement.

The AGREE option in the TABLES statement provides the weighted kappa coefficient, its standard error, and confidence limits. You can request an exact test for the weighted kappa by specifying the WTKAPPA option in the EXACT statement.

Kappa coefficients are defined only for square tables, where the number of rows equals the number of columns. PROC FREQ does not compute kappa coefficients for tables that are not square. For 2×2

tables, the weighted kappa coefficient is identical to the simple kappa coefficient, and PROC FREQ presents only the simple kappa coefficient.

WEIGHT Statement

WEIGHT *variable* < / *option* > **;**

The WEIGHT statement names a numeric variable that provides a weight for each observation in the input data set. The WEIGHT statement is most commonly used to input cell count data. See the section "Inputting Frequency Counts" on page 3241 for more information. If you use a WEIGHT statement, PROC FREQ assumes that an observation represents *n* observations, where *n* is the value of *variable*. The value of the WEIGHT variable is not required to be an integer.

If the value of the WEIGHT variable is missing, PROC FREQ does not use that observation in the analysis. If the value of the WEIGHT variable is 0, PROC FREQ ignores the observation unless you specify the ZEROS option, which includes observations that have weights of 0. If you do not specify a WEIGHT statement, PROC FREQ assigns a weight of 1 to each observation. The sum of the WEIGHT variable values represents the total number of observations.

If any value of the WEIGHT variable is negative, PROC FREQ displays the frequencies computed from the weighted values but does not compute percentages and statistics. If you create an output data set by using the OUT= option in the TABLES statement, PROC FREQ assigns missing values to the PERCENT variable. PROC FREQ also assigns missing values to the variables that the OUTEXPECT and OUTPCT options provide. If any value of the WEIGHT variable is negative, you cannot create an output data set by using the OUTPUT statement because statistics are not computed when there are negative weights.

You can specify the following *option*:

ZEROS

includes observations that have weights of 0. By default, PROC FREQ ignores observations that have weights of 0.

If you specify the ZEROS option, frequency and crosstabulation tables display levels that contain only zero-weight observations. If you do not specify the ZEROS option, PROC FREQ does not process observations that have weights of 0 and therefore does not display levels that contain only zero-weight observations.

When you specify the ZEROS option, PROC FREQ includes zero-weight levels in chi-square tests and binomial computations for one-way tables. This makes it possible to compute binomial tests and estimates for a reference level that contains no observations with positive weights.

For two-way tables, the ZEROS option enables computation of kappa statistics when there are levels that contain no observations with positive weights. For more information, see the section "Tables with Zero-Weight Rows or Columns" on page 3307.

Even when you specify the ZEROS option, PROC FREQ does not compute CHISQ or MEASURES statistics for two-way tables that contain a zero-weight row or column because most of these statistics are undefined in this case.

By default, the ZEROS option invokes the SPARSE option in the TABLES statement, which includes zero-weight table cells in the LIST table and OUT= data set. To suppress zero-weight cells, you can specify the NOSPARSE option in the TABLES statement.

Details: FREQ Procedure

Inputting Frequency Counts

PROC FREQ can use either raw data or cell count data to produce frequency and crosstabulation tables. *Raw data*, also known as case-record data, report the data as one record for each subject or sample member. *Cell count data* report the data as a table, listing all possible combinations of data values along with the frequency counts. This way of presenting data often appears in published results.

The following DATA step statements store raw data in a SAS data set:

```
data Raw;
   input Subject $ R C @@;
   datalines;
01 1 1  02 1 1  03 1 1  04 1 1  05 1 1
06 1 2  07 1 2  08 1 2  09 2 1  10 2 1
11 2 1  12 2 1  13 2 2  14 2 2  14 2 2
;
```

You can store the same data as cell counts by using the following DATA step statements:

```
data CellCounts;
   input R C Count @@;
   datalines;
1 1 5   1 2 3
2 1 4   2 2 3
;
```

The variable R contains the values for the rows, and the variable C contains the values for the columns. The variable Count contains the cell count for each row and column combination.

Both the Raw data set and the CellCounts data set produce identical frequency counts, two-way tables, and statistics. When using the CellCounts data set, you must include a WEIGHT statement to specify that the variable Count contains cell counts. For example, the following PROC FREQ statements create a two-way crosstabulation table by using the CellCounts data set:

```
proc freq data=CellCounts;
   tables R*C;
   weight Count;
run;
```

Grouping with Formats

PROC FREQ groups a variable's values according to its formatted values. If you assign a format to a variable with a FORMAT statement, PROC FREQ formats the variable values before dividing observations into the levels of a frequency or crosstabulation table.

For example, suppose that variable X has the values 1.1, 1.4, 1.7, 2.1, and 2.3. Each of these values appears as a level in the frequency table. If you decide to round each value to a single digit, include the following statement in the PROC FREQ step:

```
format X 1.;
```

Now the table lists the frequency count for formatted level 1 as two and for formatted level 2 as three.

PROC FREQ treats formatted character variables in the same way. The formatted values are used to group the observations into the levels of a frequency table or crosstabulation table. PROC FREQ uses the entire value of a character format to classify an observation.

You can also use the FORMAT statement to assign formats that were created with the FORMAT procedure to the variables. User-written formats determine the number of levels for a variable and provide labels for a table. If you use the same data with different formats, you can produce frequency counts and statistics for different classifications of the variable values.

When you use PROC FORMAT to create a user-written format that combines missing and nonmissing values into one category, PROC FREQ treats the entire category of formatted values as missing. For example, a questionnaire codes 1 as yes, 2 as no, and 8 as a no answer. The following PROC FORMAT statements create a user-written format:

```
proc format;
   value Questfmt 1   ='Yes'
                  2   ='No'
                  8,. ='Missing';
run;
```

When you use a FORMAT statement to assign Questfmt. to a variable, the variable's frequency table no longer includes a frequency count for the response of 8. You must use the MISSING or MISSPRINT option in the TABLES statement to list the frequency for no answer. The frequency count for this level includes observations with either a value of 8 or a missing value (.).

The frequency or crosstabulation table lists the values of both character and numeric variables in ascending order based on internal (unformatted) variable values unless you change the order with the ORDER= option. To list the values in ascending order by formatted values, use ORDER=FORMATTED in the PROC FREQ statement.

For more information about the FORMAT statement, see *SAS Formats and Informats: Reference*.

Missing Values

When the value of the WEIGHT variable is missing, PROC FREQ does not include that observation in the analysis.

PROC FREQ treats missing BY variable values like any other BY variable value. The missing values form a separate BY group.

If an observation has a missing value for a variable in a TABLES request, by default PROC FREQ does not include that observation in the frequency or crosstabulation table. Also by default, PROC FREQ does not include observations with missing values in the computation of percentages and statistics. The procedure displays the number of missing observations following each table.

PROC FREQ also reports the number of missing values in output data sets. The TABLES statement OUT= data set includes an observation that contains the missing value frequency. The NMISS option in the OUTPUT statement provides an output data set variable that contains the missing value frequency.

The following options change the way in which PROC FREQ handles missing values of TABLES variables:

MISSPRINT displays missing value frequencies in frequency or crosstabulation tables but does not include them in computations of percentages or statistics.

MISSING treats missing values as a valid nonmissing level for all TABLES variables. Displays missing levels in frequency and crosstabulation tables and includes them in computations of percentages and statistics.

This example shows the three ways that PROC FREQ can handle missing values of TABLES variables. The following DATA step statements create a data set with a missing value for the variable A:

```
data one;
   input A Freq;
   datalines;
1 2
2 2
. 2
;
```

The following PROC FREQ statements request a one-way frequency table for the variable A. The first request does not specify a missing value option. The second request specifies the MISSPRINT option in the TABLES statement. The third request specifies the MISSING option in the TABLES statement.

```
proc freq data=one;
   tables A;
   weight Freq;
   title 'Default';
run;
proc freq data=one;
   tables A / missprint;
   weight Freq;
   title 'MISSPRINT Option';
run;
proc freq data=one;
   tables A / missing;
```

```
    weight Freq;
    title 'MISSING Option';
  run;
```

Figure 47.12 displays the frequency tables produced by this example. The first table shows PROC FREQ's default behavior for handling missing values. The observation with a missing value of the TABLES variable A is not included in the table, and the frequency of missing values is displayed following the table. The second table, for which the MISSPRINT option is specified, displays the missing observation but does not include its frequency when computing the total frequency and percentages. The third table shows that PROC FREQ treats the missing level as a valid nonmissing level when the MISSING option is specified. The table displays the missing level, and PROC FREQ includes this level when computing frequencies and percentages.

Figure 47.12 Missing Values in Frequency Tables

Default

The FREQ Procedure

A	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	2	50.00	2	50.00
2	2	50.00	4	100.00
Frequency Missing = 2				

MISSPRINT Option

The FREQ Procedure

A	Frequency	Percent	Cumulative Frequency	Cumulative Percent
.	2	.	.	.
1	2	50.00	2	50.00
2	2	50.00	4	100.00
Frequency Missing = 2				

MISSING Option

The FREQ Procedure

A	Frequency	Percent	Cumulative Frequency	Cumulative Percent
.	2	33.33	2	33.33
1	2	33.33	4	66.67
2	2	33.33	6	100.00

When a combination of variable values in a two-way table is missing, PROC FREQ assigns 0 to the frequency count of the corresponding table cell. By default, PROC FREQ does not include missing combinations in the LIST display or the OUT= output data set. To include missing combinations in the LIST display and the OUT= output data set, you can specify the SPARSE option in the TABLES statement.

In-Database Computation

The FREQ procedure can use in-database computation to construct frequency and crosstabulation tables when the DATA= input data set is stored as a table in a supported database management system (DBMS). When large data sets are stored in an external database, the transfer of the data sets to computers that run SAS can be affected by performance, security, and resource management issues. SAS in-database processing can greatly reduce data transfer by having the database perform the initial data aggregation. For more information about in-database processing, see *SAS/ACCESS for Relational Databases: Reference*.

In-database computation for PROC FREQ supports the following database management systems:

- Amazon Redshift
- DB2
- Google BigQuery
- Greenplum
- Hadoop
- Impala
- Microsoft SQL Server
- MySQL
- Netezza
- Oracle
- PostgreSQL
- SAP HANA
- Snowflake
- Spark
- Teradata
- Vertica
- Yellowbrick

PROC FREQ performs in-database computation by using SQL implicit pass-through. The procedure generates SQL queries that are based on the tables that you request in the TABLES statement. The database executes these SQL queries to construct initial summary tables, which are then transmitted to PROC FREQ. The procedure uses this summary information to perform the remaining analyses and tasks in the usual way (out of the database). Instead of transferring the entire data set over the network between the database and SAS software, in-database computation transfers only the summary tables. This can substantially reduce processing time when the dimensions of the summary tables (in terms of rows and columns) are much smaller

than the dimensions of the entire database table (in terms of individual observations). In addition, in-database summarization uses efficient parallel processing, which can also provide performance advantages.

In-database computation is controlled by the SQLGENERATION option, which you can specify in either a LIBNAME statement or an OPTIONS statement. For information about the SQLGENERATION option and other options that affect in-database computation, see the section "In-Database Procedures" in *SAS/ACCESS for Relational Databases: Reference*. By default, PROC FREQ uses in-database computation when possible. PROC FREQ has no procedure options that control in-database computation.

PROC FREQ uses formatted values to group observations into the levels of frequency and crosstabulation tables. For more information, see the section "Grouping with Formats" on page 3242. If formats are available in the database, in-database summarization uses the formats. If formats are not available in the database, the in-database summarization uses the raw data values, and PROC FREQ performs the final, formatted classification (out of the database). For more information, see the section "Deploying and Using SAS Formats in Teradata" in *SAS/ACCESS for Relational Databases: Reference*.

The order of observations is not inherently defined for DBMS tables. The following options relate to the order of observations and therefore should not be specified for PROC FREQ in-database computation:

- If you specify the FIRSTOBS= or OBS= data set option, PROC FREQ does not perform in-database computation.
- If you specify the NOTSORTED option in the BY statement, PROC FREQ in-database computation ignores it and uses the default ASCENDING order for BY variables.
- If you specify the ORDER=DATA option for input data in a DBMS table, PROC FREQ computation might produce different results for separate runs of the same analysis. In addition to determining the order of variable levels in crosstabulation table displays, the ORDER= option can also affect the values of many of the test statistics and measures that PROC FREQ computes.

Statistical Computations

Definitions and Notation

A two-way table represents the crosstabulation of row variable X and column variable Y. Let the table row values or levels be denoted by X_i, $i = 1, 2, \ldots, R$, and the column values by Y_j, $j = 1, 2, \ldots, C$. Let n_{ij} denote the frequency of the table cell in the ith row and jth column and define the following notation:

$$n_{i\cdot} = \sum_j n_{ij} \qquad \text{(row totals)}$$

$$n_{\cdot j} = \sum_i n_{ij} \qquad \text{(column totals)}$$

$$n = \sum_i \sum_j n_{ij} \qquad \text{(overall total)}$$

$$p_{ij} = n_{ij}/n \qquad \text{(cell percentages)}$$

$$p_{i\cdot} = n_{i\cdot}/n \qquad \text{(row percentages of total)}$$

$$p_{\cdot j} = n_{\cdot j}/n \qquad \text{(column percentages of total)}$$

$$R_i = \text{score for row } i$$
$$C_j = \text{score for column } j$$
$$\bar{R} = \sum_i n_{i.} R_i / n \quad \text{(average row score)}$$
$$\bar{C} = \sum_j n_{.j} C_j / n \quad \text{(average column score)}$$
$$A_{ij} = \sum_{k>i} \sum_{l>j} n_{kl} + \sum_{k<i} \sum_{l<j} n_{kl}$$
$$D_{ij} = \sum_{k>i} \sum_{l<j} n_{kl} + \sum_{k<i} \sum_{l>j} n_{kl}$$
$$P = \sum_i \sum_j n_{ij} A_{ij} \quad \text{(twice the number of concordances)}$$
$$Q = \sum_i \sum_j n_{ij} D_{ij} \quad \text{(twice the number of discordances)}$$

Scores

PROC FREQ uses scores of the variable values to compute the Mantel-Haenszel chi-square, Pearson correlation, Cochran-Armitage test for trend, weighted kappa coefficient, and Cochran-Mantel-Haenszel statistics. The SCORES= option in the TABLES statement specifies the score type that PROC FREQ uses. The available score types are TABLE, RANK, RIDIT, and MODRIDIT scores. The default score type is TABLE. Using MODRIDIT, RANK, or RIDIT scores yields nonparametric analyses.

For numeric variables, table scores are the values of the row and column levels. If the row or column variable is formatted, then the table score is the internal numeric value corresponding to that level. If two or more numeric values are classified into the same formatted level, then the internal numeric value for that level is the smallest of these values. For character variables, table scores are defined as the row numbers and column numbers (that is, 1 for the first row, 2 for the second row, and so on).

Rank scores, which you request with the SCORES=RANK option, are defined as

$$R_i^1 = \sum_{k<i} n_{k.} + (n_{i.} + 1)/2 \qquad i = 1, 2, \ldots, R$$
$$C_j^1 = \sum_{l<j} n_{.l} + (n_{.j} + 1)/2 \qquad j = 1, 2, \ldots, C$$

where R_i^1 is the rank score of row i, and C_j^1 is the rank score of column j. Note that rank scores yield midranks for tied values.

Ridit scores, which you request with the SCORES=RIDIT option, are defined as rank scores standardized by the sample size (Bross 1958; Mack and Skillings 1980). Ridit scores are derived from the rank scores as

$$R_i^2 = R_i^1 / n \qquad i = 1, 2, \ldots, R$$
$$C_j^2 = C_j^1 / n \qquad j = 1, 2, \ldots, C$$

Modified ridit scores (SCORES=MODRIDIT) represent the expected values of the order statistics of the uniform distribution on (0,1) (Van Elteren 1960; Lehmann and D'Abrera 2006). Modified ridit scores are

derived from rank scores as

$$R_i^3 = R_i^1/(n+1) \qquad i = 1, 2, \ldots, R$$
$$C_j^3 = C_j^1/(n+1) \qquad j = 1, 2, \ldots, C$$

Chi-Square Tests and Statistics

The CHISQ option provides chi-square tests of homogeneity or independence and measures of association that are based on the chi-square statistic. When you specify the CHISQ option in the TABLES statement, PROC FREQ computes the following chi-square tests for each two-way table: Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests. PROC FREQ provides the following measures of association that are based on the Pearson chi-square statistic: phi coefficient, contingency coefficient, and Cramér's V. For 2×2 tables, the CHISQ option also provides Fisher's exact test and the continuity-adjusted chi-square statistic. You can request Fisher's exact test for general $R \times C$ tables by specifying the FISHER option in the TABLES or EXACT statement.

If you specify the CHISQ option for one-way tables, PROC FREQ provides a one-way Pearson chi-square goodness-of-fit test. If you specify the CHISQ(LRCHI) option for one-way tables, PROC FREQ also provides a one-way likelihood ratio chi-square test. The other tests and statistics that the CHISQ option produces are available only for two-way tables.

For two-way tables, the null hypothesis for the chi-square tests is no association between the row variable and the column variable. When the sample size n is large, the test statistics have asymptotic chi-square distributions under the null hypothesis. When the sample size is not large, or when the data set is sparse or heavily tied, exact tests might be more appropriate than asymptotic tests. PROC FREQ provides exact p-values for the Pearson chi-square, likelihood ratio chi-square, and Mantel-Haenszel chi-square tests, in addition to Fisher's exact test. For one-way tables, PROC FREQ provides exact p-values for the Pearson and likelihood ratio chi-square goodness-of-fit tests. You can request these exact tests by specifying the corresponding options in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

The Mantel-Haenszel chi-square statistic is appropriate only when both variables lie on an ordinal scale. The other chi-square tests and statistics in this section are appropriate for either nominal or ordinal variables. The following sections give the formulas that PROC FREQ uses to compute the chi-square tests and statistics. For more information about these statistics, see Agresti (2007) and Stokes, Davis, and Koch (2012), and the other references cited.

Chi-Square Test for One-Way Tables

For one-way frequency tables, the CHISQ option in the TABLES statement provides a chi-square goodness-of-fit test. Let C denote the number of classes, or levels, in the one-way table. Let f_i denote the frequency of class i (or the number of observations in class i) for $i = 1, 2, \ldots, C$. Then PROC FREQ computes the one-way chi-square statistic as

$$Q_P = \sum_{i=1}^{C} (f_i - e_i)^2 / e_i$$

where e_i is the expected frequency for class i under the null hypothesis.

In the test for equal proportions, which is the default for the CHISQ option, the null hypothesis specifies equal proportions of the total sample size for each class. Under this null hypothesis, the expected frequency

for each class equals the total sample size divided by the number of classes,

$$e_i = n \,/\, C \quad \text{for} \;\; i = 1, 2, \ldots, C$$

In the test for specified frequencies, which PROC FREQ computes when you input null hypothesis frequencies by using the TESTF= option, the expected frequencies are the TESTF= values that you specify. In the test for specified proportions, which PROC FREQ computes when you input null hypothesis proportions by using the TESTP= option, the expected frequencies are determined from the specified TESTP= proportions p_i as

$$e_i = p_i \times n \quad \text{for} \;\; i = 1, 2, \ldots, C$$

Under the null hypothesis (of equal proportions, specified frequencies, or specified proportions), Q_P has an asymptotic chi-square distribution with C–1 degrees of freedom.

In addition to the asymptotic test, you can request an exact one-way chi-square test by specifying the CHISQ option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Pearson Chi-Square Test for Two-Way Tables

The Pearson chi-square for two-way tables involves the differences between the observed and expected frequencies, where the expected frequencies are computed under the null hypothesis of independence. The Pearson chi-square statistic is computed as

$$Q_P = \sum_i \sum_j (n_{ij} - e_{ij})^2 / e_{ij}$$

where n_{ij} is the observed frequency in table cell (i, j) and e_{ij} is the expected frequency for table cell (i, j). The expected frequency is computed under the null hypothesis that the row and column variables are independent,

$$e_{ij} = (n_{i\cdot} \times n_{\cdot j}) \,/\, n$$

When the row and column variables are independent, Q_P has an asymptotic chi-square distribution with $(R–1)(C–1)$ degrees of freedom. For large values of Q_P, this test rejects the null hypothesis in favor of the alternative hypothesis of general association.

In addition to the asymptotic test, you can request an exact Pearson chi-square test by specifying the PCHI or CHISQ option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

For 2×2 tables, the Pearson chi-square is also appropriate for testing the equality of two binomial proportions. For $R \times 2$ and $2 \times C$ tables, the Pearson chi-square tests the homogeneity of proportions. For more information, see Fienberg (1980).

Standardized Residuals

When you specify the CROSSLIST(STDRES) option in the TABLES statement for two-way or multiway tables, PROC FREQ displays the standardized residuals in the CROSSLIST table.

The standardized residual of a crosstabulation table cell is the ratio of (*frequency* – *expected*) to its standard error, where *frequency* is the table cell frequency and *expected* is the estimated expected cell frequency. The expected frequency is computed under the null hypothesis that the row and column variables are independent. See the section "Pearson Chi-Square Test for Two-Way Tables" on page 3249 for more information.

PROC FREQ computes the standardized residual of table cell (i, j) as

$$(n_{ij} - e_{ij}) \;/\; \sqrt{e_{ij}(1 - p_{i\cdot})(1 - p_{\cdot j})}$$

where n_{ij} is the observed frequency of table cell (i, j), e_{ij} is the expected frequency of the table cell, $p_{i\cdot}$ is the proportion in row i $(n_{i\cdot}/n)$, and $p_{\cdot j}$ is the proportion in column j $(n_{\cdot j}/n)$. The expected frequency of table cell (i, j) is computed as

$$e_{ij} = (n_{i\cdot} \times n_{\cdot j}) \;/\; n$$

Under the null hypothesis of independence, each standardized residual has an asymptotic standard normal distribution. See section 2.4.5 of Agresti (2007) for more information.

Likelihood Ratio Chi-Square Test for One-Way Tables

For one-way frequency tables, the CHISQ(LRCHI) option in the TABLES statement provides a likelihood ratio chi-square goodness-of-fit test. By default, the likelihood ratio test is based on the null hypothesis of equal proportions in the C classes (levels) of the one-way table. If you specify null hypothesis proportions or frequencies by using the CHISQ(TESTP=) or CHISQ(TESTF=) option, respectively, the likelihood ratio test is based on the null hypothesis values that you specify.

PROC FREQ computes the one-way likelihood ratio test as

$$G^2 = 2 \sum_{i=1}^{C} f_i \log(f_i/e_i)$$

where f_i is the observed frequency of class i, and e_i is the expected frequency of class i under the null hypothesis.

For the null hypothesis of equal proportions, the expected frequency of each class is the total sample size divided by the number of classes,

$$e_i = n \;/\; C \quad \text{for} \;\; i = 1, 2, \ldots, C$$

If you provide null hypothesis frequencies by specifying the CHISQ(TESTF=) option in the TABLES statement, the expected frequencies are the TESTF= values that you specify. If you provide null hypothesis proportions by specifying the CHISQ(TESTP=) option in the TABLES statement, PROC FREQ computes the expected frequencies as

$$e_i = p_i \times n \quad \text{for} \;\; i = 1, 2, \ldots, C$$

where the proportions p_i are the TESTP= values that you specify.

Under the null hypothesis (of equal proportions, specified frequencies, or specified proportions), the likelihood ratio statistic G^2 has an asymptotic chi-square distribution with C–1 degrees of freedom.

In addition to the asymptotic test, you can request an exact one-way likelihood ratio chi-square test by specifying the LRCHI option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Likelihood Ratio Chi-Square Test

The likelihood ratio chi-square involves the ratios between the observed and expected frequencies. The likelihood ratio chi-square statistic is computed as

$$G^2 = 2\sum_i \sum_j n_{ij} \log\left(n_{ij}/e_{ij}\right)$$

where n_{ij} is the observed frequency in table cell (i, j) and e_{ij} is the expected frequency for table cell (i, j).

When the row and column variables are independent, G^2 has an asymptotic chi-square distribution with $(R–1)(C–1)$ degrees of freedom.

In addition to the asymptotic test, you can request an exact likelihood ratio chi-square test by specifying the LRCHI or CHISQ option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Continuity-Adjusted Chi-Square Test

The continuity-adjusted chi-square for 2×2 tables is similar to the Pearson chi-square, but it is adjusted for the continuity of the chi-square distribution. The continuity-adjusted chi-square is most useful for small sample sizes. The use of the continuity adjustment is somewhat controversial; this chi-square test is more conservative (and more like Fisher's exact test) when the sample size is small. As the sample size increases, the continuity-adjusted chi-square becomes more like the Pearson chi-square.

The continuity-adjusted chi-square statistic is computed as

$$Q_C = \sum_i \sum_j \left(\max(0, |n_{ij} - e_{ij}| - 0.5) \right)^2 \ / \ e_{ij}$$

Under the null hypothesis of independence, Q_C has an asymptotic chi-square distribution with $(R–1)(C–1)$ degrees of freedom.

Mantel-Haenszel Chi-Square Test

The Mantel-Haenszel chi-square statistic tests the alternative hypothesis that there is a linear association between the row variable and the column variable. Both variables must lie on an ordinal scale. The Mantel-Haenszel chi-square statistic is computed as

$$Q_{\mathrm{MH}} = (n-1)r^2$$

where r is the Pearson correlation between the row variable and the column variable. For a description of the Pearson correlation, see the "Pearson Correlation Coefficient" on page 3257. The Pearson correlation and thus the Mantel-Haenszel chi-square statistic use the scores that you specify in the SCORES= option in the TABLES statement. See Mantel and Haenszel (1959) and Landis, Heyman, and Koch (1978) for more information.

Under the null hypothesis of no association, Q_{MH} has an asymptotic chi-square distribution with 1 degree of freedom.

In addition to the asymptotic test, you can request an exact Mantel-Haenszel chi-square test by specifying the MHCHI or CHISQ option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Fisher's Exact Test

Fisher's exact test is another test of association between the row and column variables. This test assumes that the row and column totals are fixed and uses the hypergeometric distribution to compute probabilities of possible tables conditional on the observed row and column totals. Fisher's exact test does not depend on any large-sample distribution assumptions, and so it is appropriate even for small sample sizes and for sparse tables.

2 × 2 Tables For 2 × 2 tables, PROC FREQ gives the following information for Fisher's exact test: table probability, two-sided p-value, left-sided p-value, and right-sided p-value. The table probability is the hypergeometric probability of the observed table, and is in fact the value of the test statistic for Fisher's exact test.

Where p is the hypergeometric probability of a specific table with the observed row and column totals, Fisher's exact p-values are computed by summing probabilities p over defined sets of tables,

$$\text{Prob} = \sum_{A} p$$

The two-sided p-value is the sum of all possible table probabilities (conditional on the observed row and column totals) that are less than or equal to the observed table probability. For the two-sided p-value, the set A includes all possible tables with hypergeometric probabilities less than or equal to the probability of the observed table. A small two-sided p-value supports the alternative hypothesis of association between the row and column variables.

For 2 × 2 tables, one-sided p-values for Fisher's exact test are defined in terms of the frequency of the cell in the first row and first column of the table, the (1,1) cell. Denoting the observed (1,1) cell frequency by n_{11}, the left-sided p-value for Fisher's exact test is the probability that the (1,1) cell frequency is less than or equal to n_{11}. For the left-sided p-value, the set A includes those tables with a (1,1) cell frequency less than or equal to n_{11}. A small left-sided p-value supports the alternative hypothesis that the probability of an observation being in the first cell is actually less than expected under the null hypothesis of independent row and column variables.

Similarly, for a right-sided alternative hypothesis, A is the set of tables where the frequency of the (1,1) cell is greater than or equal to that in the observed table. A small right-sided p-value supports the alternative that the probability of the first cell is actually greater than that expected under the null hypothesis.

Because the (1,1) cell frequency completely determines the 2 × 2 table when the marginal row and column sums are fixed, these one-sided alternatives can be stated equivalently in terms of other cell probabilities or ratios of cell probabilities. The left-sided alternative is equivalent to an odds ratio less than 1, where the odds ratio is $(n_{11}n_{22}/n_{12}n_{21})$. The left-sided alternative is also equivalent to the column 1 risk for row 1 being less than the column 1 risk for row 2, $p_{1|1} < p_{1|2}$. Similarly, the right-sided alternative is equivalent to the column 1 risk for row 1 being greater than the column 1 risk for row 2, $p_{1|1} > p_{1|2}$. For more information, see Agresti (2007).

R × C Tables Fisher's exact test was extended to general $R \times C$ tables by Freeman and Halton (1951), and this test is also known as the Freeman-Halton test. For $R \times C$ tables, the two-sided p-value definition is the same as for 2 × 2 tables. The set A contains all tables with p less than or equal to the probability of the observed table. A small p-value supports the alternative hypothesis of association between the row and column variables. For $R \times C$ tables, Fisher's exact test is inherently two-sided. The alternative hypothesis is defined only in terms of general, and not linear, association. Therefore, Fisher's exact test does not have right-sided or left-sided p-values for general $R \times C$ tables.

For $R \times C$ tables, PROC FREQ computes Fisher's exact test by using the network algorithm of Mehta and Patel (1983), which provides a faster and more efficient solution than direct enumeration. See the section "Exact Statistics" on page 3318 for more details.

Phi Coefficient

The phi coefficient is a measure of association derived from the Pearson chi-square. The range of the phi coefficient is $-1 \leq \phi \leq 1$ for 2×2 tables. For tables larger than 2×2, the range is $0 \leq \phi \leq \min(\sqrt{R-1}, \sqrt{C-1})$ (Liebetrau 1983). The phi coefficient is computed as

$$\phi = (n_{11}n_{22} - n_{12}n_{21}) \,/\, \sqrt{n_{1\cdot}n_{2\cdot}n_{\cdot 1}n_{\cdot 2}} \quad \text{for } 2 \times 2 \text{ tables}$$

$$\phi = \sqrt{Q_P/n} \quad \text{otherwise}$$

See Fleiss, Levin, and Paik (2003, pp. 98–99) for more information.

Contingency Coefficient

The contingency coefficient is a measure of association derived from the Pearson chi-square. The range of the contingency coefficient is $0 \leq P \leq \sqrt{(m-1)/m}$, where $m = \min(R, C)$ (Liebetrau 1983). The contingency coefficient is computed as

$$P = \sqrt{Q_P \,/\, (Q_P + n)}$$

See Kendall and Stuart (1979, pp. 587–588) for more information.

Cramér's V

Cramér's V is a measure of association derived from the Pearson chi-square. It is designed so that the attainable upper bound is always 1. The range of Cramér's V is $-1 \leq V \leq 1$ for 2×2 tables; for tables larger than 2×2, the range is $0 \leq V \leq 1$. Cramér's V is computed as

$$V = \phi \quad \text{for } 2 \times 2 \text{ tables}$$

$$V = \sqrt{\frac{Q_P/n}{\min(R-1, C-1)}} \quad \text{otherwise}$$

See Kendall and Stuart (1979, p. 588) for more information.

Measures of Association

When you specify the MEASURES option in the TABLES statement, PROC FREQ computes several statistics that describe the association between the row and column variables of the contingency table. The following are measures of ordinal association that consider whether the column variable Y tends to increase as the row variable X increases: gamma, Kendall's tau-*b*, Stuart's tau-*c*, and Somers' *D*. These measures are appropriate for ordinal variables, and they classify pairs of observations as *concordant* or *discordant*. A pair is concordant if the observation with the larger value of X also has the larger value of Y. A pair is discordant if the observation with the larger value of X has the smaller value of Y. See Agresti (2007) and the other references cited for the individual measures of association.

The Pearson correlation coefficient and the Spearman rank correlation coefficient are also appropriate for ordinal variables. The Pearson correlation describes the strength of the linear association between the row and column variables, and it is computed by using the row and column scores specified by the SCORES=

option in the TABLES statement. The Spearman correlation is computed with rank scores. The polychoric correlation (requested by the PLCORR option) also requires ordinal variables and assumes that the variables have an underlying bivariate normal distribution. The following measures of association do not require ordinal variables and are appropriate for nominal variables: lambda asymmetric, lambda symmetric, and the uncertainty coefficients.

PROC FREQ computes estimates of the measures according to the formulas given in the following sections. For each measure, PROC FREQ computes an asymptotic standard error (ASE), which is the square root of the asymptotic variance denoted by Var in the following sections.

Confidence Limits

If you specify the CL option in the TABLES statement, PROC FREQ computes asymptotic confidence limits for all MEASURES statistics. The confidence coefficient is determined according to the value of the ALPHA= option, which, by default, is 0.05 and produces 95% confidence limits.

The confidence limits are computed as

$$\text{Est} \ \pm \ (z_{\alpha/2} \times \text{ASE})$$

where Est is the estimate of the measure, $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution, and ASE is the asymptotic standard error of the estimate.

Asymptotic Tests

For each measure that you specify in the TEST statement, PROC FREQ computes an asymptotic test of the null hypothesis that the measure is 0. Asymptotic tests are available for the following measures of association: gamma, Kendall's tau-*b*, Stuart's tau-*c*, Somers' $D(C|R)$, Somers' $D(R|C)$, the Pearson correlation coefficient, and the Spearman rank correlation coefficient. To compute an asymptotic test, PROC FREQ uses a standardized test statistic z, which has an asymptotic standard normal distribution under the null hypothesis. The test statistic is computed as

$$z = \text{Est} \ / \ \sqrt{\text{Var}_0(\text{Est})}$$

where Est is the estimate of the measure and $\text{Var}_0(\text{Est})$ is the variance of the estimate under the null hypothesis. Formulas for $\text{Var}_0(\text{Est})$ for the individual measures of association are given in the following sections.

Note that the ratio of Est to $\sqrt{\text{Var}_0(\text{Est})}$ is the same for the following measures: gamma, Kendall's tau-*b*, Stuart's tau-*c*, Somers' $D(C|R)$, and Somers' $D(R|C)$. Therefore, the tests for these measures are identical. For example, the *p*-values for the test of $H_0\colon \text{gamma} = 0$ equal the *p*-values for the test of $H_0\colon \text{tau} - b = 0$.

PROC FREQ computes one-sided and two-sided *p*-values for each of these tests. When the test statistic z is greater than its null hypothesis expected value of 0, PROC FREQ displays the right-sided *p*-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. A small right-sided *p*-value supports the alternative hypothesis that the true value of the measure is greater than 0. When the test statistic is less than or equal to 0, PROC FREQ displays the left-sided *p*-value, which is the probability of a smaller value of the statistic occurring under the null hypothesis. A small left-sided *p*-value supports the alternative hypothesis that the true value of the measure is less than 0. The one-sided *p*-value P_1 can be expressed as

$$P_1 = \begin{cases} \text{Prob}(Z > z) & \text{if} \quad z > 0 \\ \text{Prob}(Z < z) & \text{if} \quad z \le 0 \end{cases}$$

where Z has a standard normal distribution. The two-sided p-value P_2 is computed as

$$P_2 = \mathrm{Prob}(|Z| > |z|)$$

Exact Tests

Exact tests are available for the following measures of association: Kendall's tau-b, Stuart's tau-c, Somers' $D(C|R)$ and $(R|C)$, the Pearson correlation coefficient, and the Spearman rank correlation coefficient. If you request an exact test for a measure of association in the EXACT statement, PROC FREQ computes the exact test of the hypothesis that the measure is 0. For more information, see the section "Exact Statistics" on page 3318.

Gamma

The gamma (Γ) statistic is based only on the number of concordant and discordant pairs of observations. It ignores tied pairs (that is, pairs of observations that have equal values of X or equal values of Y). Gamma is appropriate only when both variables lie on an ordinal scale. The range of gamma is $-1 \leq \Gamma \leq 1$. If the row and column variables are independent, gamma tends to be close to 0. Gamma is computed as

$$G = (P - Q) \,/\, (P + Q)$$

and the asymptotic variance is

$$\mathrm{Var}(G) = \frac{16}{(P+Q)^4} \sum_i \sum_j n_{ij} (QA_{ij} - PD_{ij})^2$$

For 2×2 tables, gamma is equivalent to Yule's Q. See Goodman and Kruskal (1979) and Agresti (2002) for more information.

The variance under the null hypothesis that gamma equals 0 is computed as

$$\mathrm{Var}_0(G) = \frac{4}{(P+Q)^2} \left(\sum_i \sum_j n_{ij} (A_{ij} - D_{ij})^2 - (P-Q)^2/n \right)$$

For more information, see Brown and Benedetti (1977b).

Kendall's Tau-b

Kendall's tau-b (τ_b) is similar to gamma except that tau-b uses a correction for ties. Tau-b is appropriate only when both variables lie on an ordinal scale. The range of tau-b is $-1 \leq \tau_b \leq 1$. Kendall's tau-b is computed as

$$t_b = (P - Q) \,/\, \sqrt{w_r w_c}$$

and the asymptotic variance is

$$\mathrm{Var}(t_b) = \frac{1}{w^4} \left(\sum_i \sum_j n_{ij} (2wd_{ij} + t_b v_{ij})^2 - n^3 t_b^2 (w_r + w_c)^2 \right)$$

where

$$
\begin{aligned}
w &= \sqrt{w_r w_c} \\
w_r &= n^2 - \sum_i n_{i.}^2 \\
w_c &= n^2 - \sum_j n_{.j}^2 \\
d_{ij} &= A_{ij} - D_{ij} \\
v_{ij} &= n_{i.} w_c + n_{.j} w_r
\end{aligned}
$$

See Kendall (1955) for more information.

The variance under the null hypothesis that tau-*b* equals 0 is computed as

$$\mathrm{Var}_0(t_b) = \frac{4}{w_r w_c}\left(\sum_i \sum_j n_{ij}(A_{ij} - D_{ij})^2 - (P - Q)^2/n\right)$$

For more information, see Brown and Benedetti (1977b).

PROC FREQ also provides an exact test for the Kendall's tau-*b*. You can request this test by specifying the KENTB option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Stuart's Tau-c

Stuart's tau-*c* (τ_c) makes an adjustment for table size in addition to a correction for ties. Tau-*c* is appropriate only when both variables lie on an ordinal scale. The range of tau-*c* is $-1 \leq \tau_c \leq 1$. Stuart's tau-*c* is computed as

$$t_c = m(P - Q) \;/\; n^2(m - 1)$$

and the asymptotic variance is

$$\mathrm{Var}(t_c) = \frac{4m^2}{(m-1)^2 n^4}\left(\sum_i \sum_j n_{ij} d_{ij}^2 - (P - Q)^2/n\right)$$

where $m = \min(R, C)$ and $d_{ij} = A_{ij} - D_{ij}$. The variance under the null hypothesis that tau-*c* equals 0 is the same as the asymptotic variance

$$\mathrm{Var}_0(t_c) = \mathrm{Var}(t_c)$$

For more information, see Brown and Benedetti (1977b).

PROC FREQ also provides an exact test for the Stuart's tau-*c*. You can request this test by specifying the STUTC option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Somers' D

Somers' $D(C|R)$ and Somers' $D(R|C)$ are asymmetric modifications of tau-b. $C|R$ indicates that the row variable X is regarded as the independent variable and the column variable Y is regarded as dependent. Similarly, $R|C$ indicates that the column variable Y is regarded as the independent variable and the row variable X is regarded as dependent. Somers' D differs from tau-b in that it uses a correction only for pairs that are tied on the independent variable. Somers' D is appropriate only when both variables lie on an ordinal scale. The range of Somers' D is $-1 \leq D \leq 1$. Somers' $D(C|R)$ is computed as

$$D(C|R) = (P - Q) \;/\; w_r$$

and its asymptotic variance is

$$\mathrm{Var}(D(C|R)) = \frac{4}{w_r^4} \sum_i \sum_j n_{ij} \big(w_r d_{ij} - (P - Q)(n - n_{i.})\big)^2$$

where $d_{ij} = A_{ij} - D_{ij}$ and

$$w_r = n^2 - \sum_i n_{i.}^2$$

For more information, see Somers (1962); Goodman and Kruskal (1979); Liebetrau (1983).

The variance under the null hypothesis that $D(C|R)$ equals 0 is computed as

$$\mathrm{Var}_0(D(C|R)) = \frac{4}{w_r^2} \left(\sum_i \sum_j n_{ij} (A_{ij} - D_{ij})^2 - (P - Q)^2/n \right)$$

For more information, see Brown and Benedetti (1977b).

Formulas for Somers' $D(R|C)$ are obtained by interchanging the indices.

PROC FREQ also provides exact tests for Somers' $D(C|R)$ and $(R|C)$. You can request these tests by specifying the SMDCR and SMDCR options in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Pearson Correlation Coefficient

The Pearson correlation coefficient (ρ) is computed by using the scores specified in the SCORES= option. This measure is appropriate only when both variables lie on an ordinal scale. The range of the Pearson correlation is $-1 \leq \rho \leq 1$. The Pearson correlation coefficient is computed as

$$r = v/w = s_{rc}/\sqrt{s_r s_c}$$

and its asymptotic variance is

$$\mathrm{Var}(r) = \frac{1}{w^4} \sum_i \sum_j n_{ij} \left(w(R_i - \bar{R})(C_j - \bar{C}) - \frac{b_{ij} v}{2w} \right)^2$$

where R_i and C_j are the row and column scores and

$$
\begin{aligned}
s_r &= \sum_i \sum_j n_{ij}(R_i - \bar{R})^2 \\
s_c &= \sum_i \sum_j n_{ij}(C_j - \bar{C})^2 \\
s_{rc} &= \sum_i \sum_j n_{ij}(R_i - \bar{R})(C_j - \bar{C})
\end{aligned}
$$

$$
\begin{aligned}
b_{ij} &= (R_i - \bar{R})^2 s_c + (C_j - \bar{C})^2 s_r \\
v &= s_{rc} \\
w &= \sqrt{s_r s_c}
\end{aligned}
$$

For more information, see Snedecor and Cochran (1989).

The SCORES= option in the TABLES statement determines the type of row and column scores used to compute the Pearson correlation (and other score-based statistics). The default is SCORES=TABLE. See the section "Scores" on page 3247 for details about the available score types and how they are computed.

The variance under the null hypothesis that the correlation equals 0 is computed as

$$
\mathrm{Var}_0(r) = \left(\sum_i \sum_j n_{ij}(R_i - \bar{R})^2 (C_j - \bar{C})^2 - s_{rc}^2/n \right) \; / \; s_r s_c
$$

This expression for the variance is derived for multinomial sampling in a contingency table framework, and it differs from the form obtained under the assumption that both variables are continuous and normally distributed. For more information, see Brown and Benedetti (1977b).

PROC FREQ also provides an exact test for the Pearson correlation coefficient. You can request this test by specifying the PCORR option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Spearman Rank Correlation Coefficient

The Spearman correlation coefficient (ρ_s) is computed by using rank scores, which are defined in the section "Scores" on page 3247. This measure is appropriate only when both variables lie on an ordinal scale. The range of the Spearman correlation is $-1 \le \rho_s \le 1$. The Spearman correlation coefficient is computed as

$$
r_s = v \; / \; w
$$

and its asymptotic variance is

$$
\mathrm{Var}(r_s) = \frac{1}{n^2 w^4} \sum_i \sum_j n_{ij}(z_{ij} - \bar{z})^2
$$

where R_i and C_j are the row and column rank scores and

$$
\begin{aligned}
v &= \sum_i \sum_j n_{ij} R(i) C(j) \\
w &= \frac{1}{12}\sqrt{FG} \\
F &= n^3 - \sum_i n_{i.}^3 \\
G &= n^3 - \sum_j n_{.j}^3 \\
R(i) &= R_i - n/2 \\
C(j) &= C_j - n/2 \\
\bar{z} &= \frac{1}{n} \sum_i \sum_j n_{ij} z_{ij} \\
z_{ij} &= w v_{ij} - v w_{ij}
\end{aligned}
$$

$$
\begin{aligned}
v_{ij} &= n \Bigg(R(i)C(j) + \frac{1}{2}\sum_l n_{il} C(l) + \frac{1}{2}\sum_k n_{kj} R(k) + \\
&\qquad \sum_l \sum_{k>i} n_{kl} C(l) + \sum_k \sum_{l>j} n_{kl} R(k) \Bigg) \\
w_{ij} &= \frac{-n}{96w}\left(F n_{.j}^2 + G n_{i.}^2 \right)
\end{aligned}
$$

For more information, see Snedecor and Cochran (1989).

The variance under the null hypothesis that the correlation equals 0 is computed as

$$\mathrm{Var}_0(r_s) = \frac{1}{n^2 w^2} \sum_i \sum_j n_{ij} (v_{ij} - \bar{v})^2$$

where

$$\bar{v} = \sum_i \sum_j n_{ij} v_{ij} / n$$

This expression for the variance is derived for multinomial sampling in a contingency table framework, and it differs from the form obtained under the assumption that both variables are continuous and normally distributed. For more information, see Brown and Benedetti (1977b).

PROC FREQ also provides an exact test for the Spearman correlation coefficient. You can request this test by specifying the SCORR option in the EXACT statement. For more information, see the section "Exact Statistics" on page 3318.

Polychoric Correlation

When you specify the PLCORR option in the TABLES statement, PROC FREQ computes the polychoric correlation and its standard error. The polychoric correlation is based on the assumption that the two ordinal, categorical variables of the frequency table have an underlying bivariate normal distribution. The polychoric correlation coefficient is the maximum likelihood estimate of the product-moment correlation between the underlying normal variables. The range of the polychoric correlation is from –1 to 1. For 2×2 tables, the polychoric correlation is also known as the tetrachoric correlation (and it is labeled as such in the displayed output). See Drasgow (1986) for an overview of polychoric correlation coefficient.

Olsson (1979) gives the likelihood equations and the asymptotic standard errors for estimating the polychoric correlation. The underlying continuous variables relate to the observed crosstabulation table through thresholds, which define a range of numeric values that correspond to each categorical (table) level. PROC FREQ uses Olsson's maximum likelihood method for simultaneous estimation of the polychoric correlation and the thresholds. (Olsson also presents a two-step method that estimates the thresholds first.)

PROC FREQ iteratively solves the likelihood equations by using a Newton-Raphson algorithm. The initial estimates of the thresholds are computed from the inverse of the normal distribution function at the cumulative marginal proportions of the table. Iterative computation of the polychoric correlation stops when the convergence measure falls below the convergence criterion or when the maximum number of iterations is reached, whichever occurs first. For parameter values that are less than 0.01, the procedure evaluates convergence by using the absolute difference instead of the relative difference. The PLCORR(CONVERGE=) option specifies the convergence criterion, which is 0.0001 by default. The PLCORR(MAXITER=) option specifies the maximum number of iterations, which is 20 by default.

If you specify the CL option in the TABLES statement, PROC FREQ provides confidence limits for the polychoric correlation. The confidence limits are computed as

$$\hat{\rho} \ \pm \ (\ z_{\alpha/2} \times \mathrm{SE}(\hat{\rho}) \ )$$

where $\hat{\rho}$ is the estimate of the polychoric correlation, $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution, and $\mathrm{SE}(\hat{\rho})$ is the standard error of the polychoric correlation estimate.

If you specify the PLCORR option in the TEST statement, PROC FREQ provides Wald and likelihood ratio tests of the null hypothesis that the polychoric correlation is 0. The Wald test statistic is computed as

$$z = \hat{\rho} \ / \ \mathrm{SE}(\hat{\rho})$$

which has a standard normal distribution under the null hypothesis. PROC FREQ computes one-sided and two-sided p-values for the Wald test. When the test statistic z is greater than its null expected value of 0, PROC FREQ displays the right-sided p-value. When the test statistic is less than or equal to 0, PROC FREQ displays the left-sided p-value.

The likelihood ratio statistic for the polychoric correlation is computed as

$$G^2 = -2 \ \log(L_0/L_1)$$

where L_0 is the value of the likelihood function (Olsson 1979) when the polychoric correlation is 0, and L_1 is the value of the likelihood function at the maximum (where all parameters are replaced by their maximum likelihood estimates). Under the null hypothesis, the likelihood ratio statistic has an asymptotic chi-square distribution with 1 degree of freedom.

Lambda (Asymmetric)

Asymmetric lambda, $\lambda(C|R)$, is interpreted as the probable improvement in predicting the column variable Y given knowledge of the row variable X. The range of asymmetric lambda is $0 \leq \lambda(C|R) \leq 1$. Asymmetric lambda $(C|R)$ is computed as

$$\lambda(C|R) = \frac{\sum_i r_i - r}{n - r}$$

and its asymptotic variance is

$$\mathrm{Var}(\lambda(C|R)) = \frac{n - \sum_i r_i}{(n-r)^3}\left(\sum_i r_i + r - 2\sum_i (r_i \mid l_i = l)\right)$$

where

$$\begin{aligned} r_i &= \max_j(n_{ij}) \\ r &= \max_j(n_{\cdot j}) \\ c_j &= \max_i(n_{ij}) \\ c &= \max_i(n_{i\cdot}) \end{aligned}$$

The values of l_i and l are determined as follows. Denote by l_i the unique value of j such that $r_i = n_{ij}$, and let l be the unique value of j such that $r = n_{\cdot j}$. Because of the uniqueness assumptions, ties in the frequencies or in the marginal totals must be broken in an arbitrary but consistent manner. In case of ties, l is defined as the smallest value of j such that $r = n_{\cdot j}$.

For those columns containing a cell (i, j) for which $n_{ij} = r_i = c_j$, cs_j records the row in which c_j is assumed to occur. Initially cs_j is set equal to –1 for all j. Beginning with i=1, if there is at least one value j such that $n_{ij} = r_i = c_j$, and if $cs_j = -1$, l_i is defined to be the smallest such value of j, and cs_j is set equal to i. Otherwise, if $n_{il} = r_i$, l_i is defined to be equal to l. If neither condition is true, l_i is taken to be the smallest value of j such that $n_{ij} = r_i$.

The formulas for lambda asymmetric $(R|C)$ can be obtained by interchanging the indices.

For more information, see Goodman and Kruskal (1979).

Lambda (Symmetric)

The nondirectional lambda is the average of the two asymmetric lambdas, $\lambda(C|R)$ and $\lambda(R|C)$. Its range is $0 \leq \lambda \leq 1$. Lambda symmetric is computed as

$$\lambda = \frac{\sum_i r_i + \sum_j c_j - r - c}{2n - r - c} = \frac{w - v}{w}$$

and its asymptotic variance is computed as

$$\mathrm{Var}(\lambda) = \frac{1}{w^4}\Big(wvy - 2w^2\big(n - \sum_i\sum_j (n_{ij} \mid j = l_i, i = k_j)\big) - 2v^2(n - n_{kl})\Big)$$

where

$$
\begin{aligned}
r_i &= \max_j(n_{ij}) \\
r &= \max_j(n_{\cdot j}) \\
c_j &= \max_i(n_{ij}) \\
c &= \max_i(n_{i\cdot}) \\
w &= 2n - r - c \\
v &= 2n - \sum_i r_i - \sum_j c_j \\
x &= \sum_i (r_i \mid l_i = l) \; + \; \sum_j (c_j \mid k_j = k) \; + \; r_k \; + \; c_l \\
y &= 8n - w - v - 2x
\end{aligned}
$$

The definitions of l_i and l are given in the previous section. The values k_j and k are defined in a similar way for lambda asymmetric $(R|C)$.

For more information, see Goodman and Kruskal (1979).

Uncertainty Coefficients (Asymmetric)

The uncertainty coefficient $U(C|R)$ measures the proportion of uncertainty (entropy) in the column variable Y that is explained by the row variable X. Its range is $0 \leq U(C|R) \leq 1$. The uncertainty coefficient is computed as

$$U(C|R) = (H(X) + H(Y) - H(XY)) \;/\; H(Y) = v/w$$

and its asymptotic variance is

$$\mathrm{Var}(U(C|R)) = \frac{1}{n^2 w^4} \sum_i \sum_j n_{ij} \left(H(Y) \log\left(\frac{n_{ij}}{n_{i\cdot}}\right) + (H(X) - H(XY)) \log\left(\frac{n_{\cdot j}}{n}\right)\right)^2$$

where

$$
\begin{aligned}
v &= H(X) + H(Y) - H(XY) \\
w &= H(Y) \\
H(X) &= -\sum_i \left(\frac{n_{i\cdot}}{n}\right) \log\left(\frac{n_{i\cdot}}{n}\right) \\
H(Y) &= -\sum_j \left(\frac{n_{\cdot j}}{n}\right) \log\left(\frac{n_{\cdot j}}{n}\right) \\
H(XY) &= -\sum_i \sum_j \left(\frac{n_{ij}}{n}\right) \log\left(\frac{n_{ij}}{n}\right)
\end{aligned}
$$

The formulas for the uncertainty coefficient $U(R|C)$ can be obtained by interchanging the indices.

For more information, see Theil (1972, pp. 115–120) and Goodman and Kruskal (1979).

Uncertainty Coefficient (Symmetric)

The uncertainty coefficient U is the symmetric version of the two asymmetric uncertainty coefficients. Its range is $0 \le U \le 1$. The uncertainty coefficient is computed as

$$U = 2\,(H(X) + H(Y) - H(XY))\ /\ (H(X) + H(Y))$$

and its asymptotic variance is

$$\mathrm{Var}(U) = 4 \sum_i \sum_j \frac{n_{ij}\left(H(XY)\log\left(\frac{n_{i\cdot}n_{\cdot j}}{n^2}\right) - (H(X) + H(Y))\log\left(\frac{n_{ij}}{n}\right)\right)^2}{n^2\,(H(X) + H(Y))^4}$$

where $H(X)$, $H(Y)$, and $H(XY)$ are defined in the previous section. For more information, see Goodman and Kruskal (1979).

Binomial Proportion

If you specify the BINOMIAL option in the TABLES statement, PROC FREQ computes the binomial proportion for one-way tables. By default, this is the proportion of observations in the first variable level that appears in the output. (You can use the LEVEL= option to specify a different level for the proportion.) The binomial proportion is computed as

$$\hat{p} = n_1\ /\ n$$

where n_1 is the frequency of the first (or designated) level and n is the total frequency of the one-way table. The standard error of the binomial proportion is computed as

$$\mathrm{se}(\hat{p}) = \sqrt{\hat{p}\,(1 - \hat{p})\ /\ n}$$

Binomial Confidence Limits

PROC FREQ provides Wald and exact (Clopper-Pearson) confidence limits for the binomial proportion. You can also request the following binomial confidence limit types by specifying the BINOMIAL(CL=) option: Agresti-Coull, Blaker, Jeffreys, exact mid-*p*, likelihood ratio, logit, and Wilson (score). For more information, see Brown, Cai, and DasGupta (2001), Agresti and Coull (1998), and Newcombe (1998b), in addition to the references cited for each confidence limit type.

Wald Confidence Limits Wald asymptotic confidence limits are based on the normal approximation to the binomial distribution. PROC FREQ computes the Wald confidence limits for the binomial proportion as

$$\hat{p}\ \pm\ (\ z_{\alpha/2} \times \mathrm{se}(\hat{p})\ )$$

where $z_{\alpha/2}$ is the $100(1 - \alpha/2)$th percentile of the standard normal distribution. The confidence level α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

If you specify CL=WALD(CORRECT) or the CORRECT *binomial-option*, PROC FREQ includes a continuity correction of $1/2n$ in the Wald asymptotic confidence limits. The purpose of this correction is to adjust for the difference between the normal approximation and the discrete binomial distribution. See Fleiss, Levin, and Paik (2003) for more information. The continuity-corrected Wald confidence limits for the binomial proportion are computed as

$$\hat{p}\ \pm\ (\ z_{\alpha/2} \times \mathrm{se}(\hat{p})\ +\ (1/2n)\ )$$

Exact (Clopper-Pearson) Confidence Limits Exact (Clopper-Pearson) confidence limits for the binomial proportion are constructed by inverting the equal-tailed test based on the binomial distribution. This method is attributed to Clopper and Pearson (1934). The exact confidence limits P_L and P_U satisfy the following equations, for $n_1 = 1, 2, \ldots n - 1$:

$$\sum_{x=n_1}^{n} \binom{n}{x} P_L^{\,x} (1 - P_L)^{\,n-x} = \alpha/2$$

$$\sum_{x=0}^{n_1} \binom{n}{x} P_U^{\,x} (1 - P_U)^{\,n-x} = \alpha/2$$

The lower confidence limit is 0 when $n_1 = 0$, and the upper confidence limit is 1 when $n_1 = n$.

PROC FREQ computes the exact (Clopper-Pearson) confidence limits by using the F distribution as

$$P_L = \left(1 + \frac{n - n_1 + 1}{n_1\ F(\ \alpha/2,\ 2n_1,\ 2(n - n_1 + 1)\ )}\right)^{-1}$$

$$P_U = \left(1 + \frac{n - n_1}{(n_1 + 1)\ F(\ 1 - \alpha/2,\ 2(n_1 + 1),\ 2(n - n_1)\ )}\right)^{-1}$$

where $F(\alpha/2, b, c)$ is the $(\alpha/2)$th percentile of the F distribution with b and c degrees of freedom. See Leemis and Trivedi (1996) for a derivation of this expression. Also see Collett (1991) for more information about exact binomial confidence limits.

Because this is a discrete problem, the confidence coefficient (coverage probability) of the exact (Clopper-Pearson) interval is not exactly $(1 - \alpha)$ but is at least $(1 - \alpha)$. Thus, this confidence interval is conservative. Unless the sample size is large, the actual coverage probability can be much larger than the target value. For more information about the performance of these confidence limits, see Agresti and Coull (1998), Brown, Cai, and DasGupta (2001), and Leemis and Trivedi (1996).

Agresti-Coull Confidence Limits If you specify the CL=AGRESTICOULL *binomial-option*, PROC FREQ computes Agresti-Coull confidence limits for the binomial proportion as

$$\tilde{p}\ \pm\ (\ z_{\alpha/2} \times \sqrt{\tilde{p}\ (1 - \tilde{p})\ /\ \tilde{n}}\ )$$

where

$$\begin{aligned} \tilde{n}_1 &= n_1 + z_{\alpha/2}^2/2 \\ \tilde{n} &= n + z_{\alpha/2}^2 \\ \tilde{p} &= \tilde{n}_1\ /\ \tilde{n} \end{aligned}$$

The Agresti-Coull confidence interval has the same general form as the standard Wald interval but uses $\tilde{p}$ in place of $\hat{p}$. For $\alpha = 0.05$, the value of $z_{\alpha/2}$ is close to 2, and this interval is the "add 2 successes and 2 failures" adjusted Wald interval of Agresti and Coull (1998).

Blaker Confidence Limits If you specify the CL=BLAKER *binomial-option*, PROC FREQ computes Blaker confidence limits for the binomial proportion, which are constructed by inverting the two-sided exact Blaker test (Blaker 2000). The $100(1-\alpha)\%$ Blaker confidence interval consists of all values of the proportion p_0 for which the test statistic $B(p_0, n_1)$ falls in the acceptance region,

$$\{p_0 : B(p_0, n_1) > \alpha\}$$

where

$$B(p_0, n_1) \;=\; \mathrm{Prob}(\; \gamma(p_0, X) \le \gamma(p_0, n_1) \mid p_0 \;)$$

$$\gamma(p_0, n_1) \;=\; \min(\; \mathrm{Prob}(\; X \ge n_1 \mid p_0 \;),\; \mathrm{Prob}(\; X \le n_1 \mid p_0 \;)\;)$$

and X is a binomial random variable. For more information, see Blaker (2000).

Jeffreys Confidence Limits If you specify the CL=JEFFREYS *binomial-option*, PROC FREQ computes Jeffreys confidence limits for the binomial proportion as

$$\big(\; \beta(\alpha/2,\; n_1 + 1/2,\; n - n_1 + 1/2),\;\; \beta(1-\alpha/2,\; n_1 + 1/2,\; n - n_1 + 1/2) \;\big)$$

where $\beta(\alpha, b, c)$ is the αth percentile of the beta distribution with shape parameters b and c. The lower confidence limit is set to 0 when $n_1 = 0$, and the upper confidence limit is set to 1 when $n_1 = n$. This is an equal-tailed interval based on the noninformative Jeffreys prior for a binomial proportion. For more information, see Brown, Cai, and DasGupta (2001). For information about using beta priors for inference on the binomial proportion, see Berger (1985).

When you specify CL=JEFFREYS(MODIFY), the procedure provides modified Jeffreys confidence limits (Brown, Cai, and DasGupta 2001, p. 113) to improve the coverage of the confidence limits for proportions near 0 or 1. When $n_1 = 0$, the modified version replaces the upper confidence limit with $p_l = 1 - (\alpha/2)^{1/n}$. When $n_1 = n$, the modified version replaces the lower confidence limit with $1 - p_l$. The modified version also sets the lower confidence limit to 0 when $n_1 = 1$ and sets the upper confidence limit to 1 when $n_1 = n - 1$. Otherwise (when $1 < n_1 < n - 1$), the Jeffreys confidence limits are not modified.

Likelihood Ratio Confidence Limits If you specify the CL=LIKELIHOODRATIO *binomial-option*, PROC FREQ computes likelihood ratio confidence limits for the binomial proportion by inverting the likelihood ratio test. The likelihood ratio test statistic for the null hypothesis that the proportion equals p_0 can be expressed as

$$L(p_0) \;=\; -2\,(n_1 \log(\hat{p}/p_0) \;+\; (n - n_1)\log((1-\hat{p})/(1-p_0)))$$

The $100(1-\alpha)\%$ likelihood ratio confidence interval consists of all values of p_0 for which the test statistic $L(p_0)$ falls in the acceptance region,

$$\{p_0 : L(p_0) < \chi^2_{1,\alpha}\}$$

where $\chi^2_{1,\alpha}$ is the $100(1-\alpha)$th percentile of the chi-square distribution with 1 degree of freedom. PROC FREQ finds the confidence limits by iterative computation. For more information, see Fleiss, Levin, and Paik (2003), Brown, Cai, and DasGupta (2001), Agresti (2013), and Newcombe (1998b).

Logit Confidence Limits If you specify the CL=LOGIT *binomial-option*, PROC FREQ computes logit confidence limits for the binomial proportion, which are based on the logit transformation $Y = \log(\hat{p}/(1-\hat{p}))$. Approximate confidence limits for Y are computed as

$$Y_L = \log(\hat{p}/(1-\hat{p})) - z_{\alpha/2}\sqrt{n/(n_1(n-n_1))}$$

$$Y_U = \log(\hat{p}/(1-\hat{p})) + z_{\alpha/2}\sqrt{n/(n_1(n-n_1))}$$

The confidence limits for Y are inverted to produce $100(1-\alpha)\%$ logit confidence limits P_L and P_U for the binomial proportion p as

$$P_L = \exp(Y_L/(1+\exp(Y_L))$$

$$P_U = \exp(Y_U/(1+\exp(Y_U))$$

For more information, see Brown, Cai, and DasGupta (2001) and Korn and Graubard (1998).

Mid-*p* Confidence Limits If you specify the CL=MIDP *binomial-option*, PROC FREQ computes exact mid-*p* confidence limits for the binomial proportion by inverting two one-sided binomial tests that include mid-*p* tail areas. The mid-*p* approach replaces the probability of the observed frequency with half of that probability in the Clopper-Pearson sum, which is described in the section "Exact (Clopper-Pearson) Confidence Limits" on page 3264. The exact mid-*p* confidence limits P_L and P_U are the solutions to the equations

$$\sum_{x=n_1+1}^{n}\binom{n}{x}P_L^{\,x}(1-P_L)^{\,n-x} + \frac{1}{2}\binom{n}{n_1}P_L^{\,n_1}(1-P_L)^{\,n-n_1} = \alpha/2$$

$$\sum_{x=0}^{n_1-1}\binom{n}{x}P_U^{\,x}(1-P_U)^{\,n-x} + \frac{1}{2}\binom{n}{n_1}P_U^{\,n_1}(1-P_U)^{\,n-n_1} = \alpha/2$$

For more information, see Agresti and Gottard (2007), Agresti (2013), Newcombe (1998b), and Brown, Cai, and DasGupta (2001).

Wilson (Score) Confidence Limits If you specify the CL=WILSON *binomial-option*, PROC FREQ computes Wilson confidence limits for the binomial proportion. These are also known as score confidence limits (Wilson 1927). The confidence limits are based on inverting the normal test that uses the null proportion in the variance (the score test). Wilson confidence limits are the roots of

$$|p-\hat{p}| = z_{\alpha/2}\sqrt{p(1-p)/n}$$

and are computed as

$$\left(\hat{p} + z_{\alpha/2}^2/2n \pm z_{\alpha/2}\sqrt{\left(\hat{p}(1-\hat{p}) + z_{\alpha/2}^2/4n\right)/n}\right) \Big/ \left(1 + z_{\alpha/2}^2/n\right)$$

When you specify CL=WILSON(CORRECT) or the CORRECT *binomial-option*, PROC FREQ provides continuity-corrected Wilson confidence limits, which are computed as the roots of

$$|p-\hat{p}| - 1/2n = z_{\alpha/2}\sqrt{p(1-p)/n}$$

The Wilson interval has been shown to have better performance than the Wald interval and the exact (Clopper-Pearson) interval. For more information, see Agresti and Coull (1998), Brown, Cai, and DasGupta (2001), and Newcombe (1998b).

When you specify CL=WILSON(ADAPT), PROC FREQ provides adapted Wilson confidence limits (Agresti and Coull 1998, p. 125) to improve the coverage of the confidence limits for proportions near 0 or 1. When $n_1 = 1$, the adapted version replaces the lower confidence limit with $(-\log(1-\alpha)/n)$. When $n_1 = (n-1)$, the adapted version replaces the upper confidence limit with $(1+\log(1-\alpha)/n)$.

When you specify CL=WILSON(MODIFY), PROC FREQ provides modified Wilson confidence limits (Brown, Cai, and DasGupta 2001, pp. 112–113) to improve the coverage of the confidence limits for proportions near 0 or 1. The modified version replaces the lower confidence limit when $n_1 = 1, \ldots, n_1^*$, where $n_1^* = 2$ for $n \leq 50$ and $n_1^* = 3$ otherwise. The modified lower confidence limit is computed as $\chi^2(2n_1, \alpha)/2n$, where $\chi^2(2n_1, \alpha)$ is the 100αth percentile of the chi-square distribution with $2n_1$ degrees of freedom. Similarly, the modified version replaces the upper confidence limit when $n_1 = n - n_1^*, \ldots, n-1$. The modified upper confidence limit is computed as $1 - \chi^2(2(n-n_1), \alpha)/2n$.

Binomial Tests

The BINOMIAL option provides an asymptotic equality test for the binomial proportion by default. You can also specify *binomial-options* to request tests of noninferiority, superiority, and equivalence for the binomial proportion. If you specify the BINOMIAL option in the EXACT statement, PROC FREQ also computes exact p-values for the tests that you request with the *binomial-options*.

Equality Test PROC FREQ computes an asymptotic test of the hypothesis that the binomial proportion equals p_0, where you can specify the value of p_0 with the P= *binomial-option*. If you do not specify a null value with P=, PROC FREQ uses $p_0 = 0.5$ by default. The binomial test statistic is computed as

$$z = (\hat{p} - p_0)/\mathrm{se}$$

By default, the standard error is based on the null hypothesis proportion as

$$\mathrm{se} = \sqrt{p_0(1-p_0)/n}$$

If you specify the VAR=SAMPLE *binomial-option*, the standard error is computed from the sample proportion as

$$\mathrm{se} = \sqrt{\hat{p}(1-\hat{p})/n}$$

If you specify the CORRECT *binomial-option*, PROC FREQ includes a continuity correction in the asymptotic test statistic, towards adjusting for the difference between the normal approximation and the discrete binomial distribution. For more information, see Fleiss, Levin, and Paik (2003). The continuity correction of $(1/2n)$ is subtracted from the numerator of the test statistic if $(\hat{p} - p_0)$ is positive; otherwise, the continuity correction is added to the numerator.

PROC FREQ computes one-sided and two-sided p-values for this test. When the test statistic z is greater than 0 (its expected value under the null hypothesis), PROC FREQ computes the right-sided p-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. A small right-sided p-value supports the alternative hypothesis that the true value of the proportion is greater than p_0. When the test statistic is less than or equal to 0, PROC FREQ computes the left-sided p-value, which is the probability of a smaller value of the statistic occurring under the null hypothesis. A small left-sided p-value supports the

alternative hypothesis that the true value of the proportion is less than p_0. The one-sided p-value P_1 can be expressed as

$$P_1 = \begin{cases} \text{Prob}(Z > z) & \text{if} \quad z > 0 \\ \text{Prob}(Z < z) & \text{if} \quad z \leq 0 \end{cases}$$

where Z has a standard normal distribution. The two-sided p-value is computed as $P_2 = 2 \times P_1$.

If you specify the BINOMIAL option in the EXACT statement, PROC FREQ also computes an exact test of the null hypothesis H_0: $p = p_0$. To compute the exact test, PROC FREQ uses the binomial probability function,

$$\text{Prob}(X = x \mid p_0) = \binom{n}{x} p_0^{\,x} \, (1 - p_0)^{\,(n-x)} \qquad \text{for} \quad x = 0, 1, 2, \ldots, n$$

where the variable X has a binomial distribution with parameters n and p_0. To compute the left-sided p-value, $\text{Prob}(X \leq n_1)$, PROC FREQ sums the binomial probabilities over x from 0 to n_1. To compute the right-sided p-value, $\text{Prob}(X \geq n_1)$, PROC FREQ sums the binomial probabilities over x from n_1 to n. The exact one-sided p-value is the minimum of the left-sided and right-sided p-values,

$$P_1 = \min\left(\text{Prob}(X \leq n_1 \mid p_0), \ \text{Prob}(X \geq n_1 \mid p_0) \right)$$

and the exact two-sided p-value is computed as $P_2 = 2 \times P_1$.

Noninferiority Test If you specify the NONINF *binomial-option*, PROC FREQ provides a noninferiority test for the binomial proportion. The null hypothesis for the noninferiority test is

$$H_0\colon p - p_0 \leq -\delta$$

versus the alternative

$$H_a\colon p - p_0 > -\delta$$

where δ is the noninferiority margin and p_0 is the null proportion. Rejection of the null hypothesis indicates that the binomial proportion is not inferior to the null value. See Chow, Shao, and Wang (2003) for more information.

You can specify the value of δ with the MARGIN= *binomial-option*, and you can specify p_0 with the P= *binomial-option*. By default, $\delta = 0.2$ and $p_0 = 0.5$.

PROC FREQ provides an asymptotic Wald test for noninferiority. The test statistic is computed as

$$z = (\hat{p} - p_0^*) \ / \ \text{se}$$

where p_0^* is the noninferiority limit,

$$p_0^* = p_0 - \delta$$

By default, the standard error is computed from the sample proportion as

$$\text{se} = \sqrt{\hat{p}(1 - \hat{p})/n}$$

If you specify the VAR=NULL *binomial-option*, the standard error is based on the noninferiority limit (determined by the null proportion and the margin) as

$$\text{se} = \sqrt{p_0^*(1 - p_0^*)/n}$$

If you specify the CORRECT *binomial-option*, PROC FREQ includes a continuity correction in the asymptotic test statistic z. The continuity correction of $(1/2n)$ is subtracted from the numerator of the test statistic if $(\hat{p} - p_0^*)$ is positive; otherwise, the continuity correction is added to the numerator.

The p-value for the noninferiority test is

$$P_z = \text{Prob}(Z > z)$$

where Z has a standard normal distribution.

As part of the noninferiority analysis, PROC FREQ provides asymptotic Wald confidence limits for the binomial proportion. These confidence limits are computed as described in the section "Wald Confidence Limits" on page 3263 but use the same standard error (VAR=NULL or VAR=SAMPLE) as the noninferiority test statistic z. The confidence coefficient is $100(1 - 2\alpha)\%$ (Schuirmann 1999). By default, if you do not specify the ALPHA= option, the noninferiority confidence limits are 90% confidence limits. You can compare the confidence limits to the noninferiority limit, $p_0^* = p_0 - \delta$.

If you specify the BINOMIAL option in the EXACT statement, PROC FREQ provides an exact noninferiority test for the binomial proportion. The exact p-value is computed by using the binomial probability function with parameters p_0^* and n,

$$P_x = \sum_{k=n_1}^{k=n} \binom{n}{k} (p_0^*)^{\,k}\,(1 - p_0^*)^{\,(n-k)}$$

For more information, see Chow, Shao, and Wang (2003, p. 116). If you request exact binomial statistics, PROC FREQ also includes exact (Clopper-Pearson) confidence limits for the binomial proportion in the equivalence analysis display. For more information, see the section "Exact (Clopper-Pearson) Confidence Limits" on page 3264.

Superiority Test If you specify the SUP *binomial-option*, PROC FREQ provides a superiority test for the binomial proportion. The null hypothesis for the superiority test is

$$H_0\colon p - p_0 \le \delta$$

versus the alternative

$$H_a\colon p - p_0 > \delta$$

where δ is the superiority margin and p_0 is the null proportion. Rejection of the null hypothesis indicates that the binomial proportion is superior to the null value. You can specify the value of δ with the MARGIN= *binomial-option*, and you can specify the value of p_0 with the P= *binomial-option*. By default, $\delta = 0.2$ and $p_0 = 0.5$.

The superiority analysis is identical to the noninferiority analysis but uses a positive value of the margin δ in the null hypothesis. The superiority limit equals $p_0 + \delta$. The superiority computations follow those in the section "Noninferiority Test" on page 3268 but replace $-\delta$ with δ. See Chow, Shao, and Wang (2003) for more information.

Equivalence Test If you specify the EQUIV *binomial-option*, PROC FREQ provides an equivalence test for the binomial proportion. The null hypothesis for the equivalence test is

$$H_0\colon p - p_0 \le \delta_L \quad \text{or} \quad p - p_0 \ge \delta_U$$

versus the alternative

$$H_a\colon \delta_L < p - p_0 < \delta_U$$

where δ_L is the lower margin, δ_U is the upper margin, and p_0 is the null proportion. Rejection of the null hypothesis indicates that the binomial proportion is equivalent to the null value. See Chow, Shao, and Wang (2003) for more information.

You can specify the value of the margins δ_L and δ_U with the MARGIN= *binomial-option*. If you do not specify MARGIN=, PROC FREQ uses lower and upper margins of –0.2 and 0.2 by default. If you specify a single margin value δ, PROC FREQ uses lower and upper margins of $-\delta$ and δ. You can specify the null proportion p_0 with the P= *binomial-option*. By default, $p_0 = 0.5$.

PROC FREQ computes two one-sided tests (TOST) for equivalence analysis (Schuirmann 1987). The TOST approach includes a right-sided test for the lower margin and a left-sided test for the upper margin. The overall p-value is taken to be the larger of the two p-values from the lower and upper tests.

For the lower margin, the asymptotic Wald test statistic is computed as

$$z_L = (\hat{p} - p_L^*) \;/\; \text{se}$$

where the lower equivalence limit is

$$p_L^* = p_0 + \delta_L$$

By default, the standard error is computed from the sample proportion as

$$\text{se} = \sqrt{\hat{p}(1-\hat{p})/n}$$

If you specify the VAR=NULL *binomial-option*, the standard error is based on the lower equivalence limit (determined by the null proportion and the lower margin) as

$$\text{se} = \sqrt{p_L^*(1-p_L^*)/n}$$

If you specify the CORRECT *binomial-option*, PROC FREQ includes a continuity correction in the asymptotic test statistic z_L. The continuity correction of $(1/2n)$ is subtracted from the numerator of the test statistic $(\hat{p} - p_L^*)$ if the numerator is positive; otherwise, the continuity correction is added to the numerator.

The p-value for the lower margin test is

$$P_{z,L} = \text{Prob}(Z > z_L)$$

The asymptotic test for the upper margin is computed similarly. The Wald test statistic is

$$z_U = (\hat{p} - p_U^*) \;/\; \text{se}$$

where the upper equivalence limit is

$$p_U^* = p_0 + \delta_U$$

By default, the standard error is computed from the sample proportion. If you specify the VAR=NULL *binomial-option*, the standard error is based on the upper equivalence limit as

$$\text{se} = \sqrt{p_U^*(1 - p_U^*)/n}$$

If you specify the CORRECT *binomial-option*, PROC FREQ includes a continuity correction of $(1/2n)$ in the asymptotic test statistic z_U.

The p-value for the upper margin test is

$$P_{z,U} = \text{Prob}(Z < z_U)$$

Based on the two one-sided tests (TOST), the overall p-value for the test of equivalence equals the larger p-value from the lower and upper margin tests, which can be expressed as

$$P_z = \max(P_{z,L}\ ,\ P_{z,U})$$

As part of the equivalence analysis, PROC FREQ provides asymptotic Wald confidence limits for the binomial proportion. These confidence limits are computed as described in the section "Wald Confidence Limits" on page 3263, but use the same standard error (VAR=NULL or VAR=SAMPLE) as the equivalence test statistics and have a confidence coefficient of $100(1 - 2\alpha)\%$ (Schuirmann 1999). By default, if you do not specify the ALPHA= option, the equivalence confidence limits are 90% limits. If you specify VAR=NULL, separate standard errors are computed for the lower and upper margin tests, each based on the null proportion and the corresponding (lower or upper) margin. The confidence limits are computed by using the maximum of these two standard errors. You can compare the confidence limits to the equivalence limits, $(p_0 + \delta_L,\ p_0 + \delta_U)$.

If you specify the BINOMIAL option in the EXACT statement, PROC FREQ also provides an exact equivalence test by using two one-sided exact tests (TOST). The procedure computes lower and upper margin exact tests by using the binomial probability function as described in the section "Noninferiority Test" on page 3268. The overall exact p-value for the equivalence test is taken to be the larger p-value from the lower and upper margin exact tests. If you request exact statistics, PROC FREQ also includes exact (Clopper-Pearson) confidence limits in the equivalence analysis display. The confidence coefficient is $100(1 - 2\alpha)\%$ (Schuirmann 1999). For more information, see the section "Exact (Clopper-Pearson) Confidence Limits" on page 3264.

Sensitivity and Specificity

The SENSPEC option in the TABLES statement provides estimates of sensitivity, specificity, positive predictive value, and negative predictive value for 2×2 tables. These measures are conditional (row and column) proportions in the 2×2 crosstabulation table. In sensitivity analysis, the row variable might represent a positive or negative diagnostic test result, and the column variable might represent the presence or absence of a condition. For more information, see Fleiss, Levin, and Paik (2003). The SENSPEC option also provides an estimate of accuracy, which is the proportion of correct classifications.

By default, the SENSPEC computations use table cell (1,1) as the reference cell that represents the true positives (positive row value and positive column value). You can specify a different reference cell by using the SENSPEC REFCELL= suboption or the REFCOLUMN= and REFROW= suboptions. For example, REFCELL=4 specifies table cell (2,2) as the true-positive cell (and row 2 and column 2 as the positive row and column levels, respectively). The notation in this section assumes that table cell (1,1) is the reference cell.

Sensitivity is computed as the column proportion for table cell (1,1), which is the ratio of the frequency in table cell (1,1) to the total frequency in column 1 of the 2×2 table. Sensitivity is denoted by

$$\mathrm{SN} = n_{11} \,/\, n_{\cdot 1}$$

Specificity is computed as the column proportion for table cell (2,2), which is the ratio of the frequency in table cell (2,2) to the total frequency in column 2. Specificity is denoted as

$$\mathrm{SP} = n_{22} \,/\, n_{\cdot 2}$$

The positive predictive value is the row proportion for table cell (1,1), which is computed as

$$\mathrm{PPV} = n_{11} \,/\, n_{1\cdot}$$

The negative predictive value is the row proportion for table cell (2,2), which is computed as

$$\mathrm{NPV} = n_{22} \,/\, n_{2\cdot}$$

Accuracy (correct classification rate) is the overall proportion in table cells (1,1) and (2,2), which is computed as

$$\mathrm{A} = (n_{11} + n_{22}) \,/\, n$$

The "Sensitivity and Specificity" table provides the estimates together with their standard errors and Wald confidence limits. PROC FREQ computes the standard errors and Wald confidence limits for these proportions as described in the section "Risks and Risk Differences" on page 3272. The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

Risks and Risk Differences

The RISKDIFF option in the TABLES statement provides estimates of risks (binomial proportions) and risk differences for 2×2 tables. This analysis might be appropriate when comparing the proportion of some characteristic for two groups, where row 1 and row 2 correspond to the two groups, and the columns correspond to two possible characteristics or outcomes. For example, the row variable might be a treatment or dose, and the column variable might be the response. For more information, see Collett (1991); Fleiss, Levin, and Paik (2003); Stokes, Davis, and Koch (2012).

Let the frequencies of the 2×2 table be represented as follows:

	Column 1	Column 2	Total
Row 1	n_{11}	n_{12}	$n_{1\cdot}$
Row 2	n_{21}	n_{22}	$n_{2\cdot}$
Total	$n_{\cdot 1}$	$n_{\cdot 2}$	n

By default when you specify the RISKDIFF option, PROC FREQ provides estimates of the row 1 risk (proportion), the row 2 risk, the overall risk, and the risk difference for column 1 and for column 2 of the 2×2 table. The risk difference is defined as the row 1 risk minus the row 2 risk. The risks are binomial proportions of their rows (row 1, row 2, or overall), and the computation of their standard errors and Wald confidence limits follow the binomial proportion computations, which are described in the section "Binomial Proportion" on page 3263.

The column 1 risk for row 1 is the proportion of row 1 observations classified in column 1,

$$\hat{p}_1 = n_{11} \ / \ n_{1\cdot}$$

which estimates the conditional probability of the column 1 response, given the first level of the row variable. The column 1 risk for row 2 is the proportion of row 2 observations classified in column 1,

$$\hat{p}_2 = n_{21} \ / \ n_{2\cdot}$$

The overall column 1 risk is the proportion of all observations classified in column 1,

$$\hat{p} = n_{\cdot 1} \ / \ n$$

The column 1 risk difference compares the risks for the two rows, and it is computed as the column 1 risk for row 1 minus the column 1 risk for row 2,

$$\hat{d} = \hat{p}_1 - \hat{p}_2$$

The standard error of the column 1 risk for row *i* is computed as

$$\mathrm{se}(\hat{p}_i) = \sqrt{\hat{p}_i \ (1 - \hat{p}_i) \ / \ n_{i\cdot}}$$

The standard error of the overall column 1 risk is computed as

$$\mathrm{se}(\hat{p}) = \sqrt{\hat{p} \ (1 - \hat{p}) \ / \ n}$$

Where the two rows represent independent binomial samples, the standard error of the column 1 risk difference is computed as

$$\mathrm{se}(\hat{d}) = \sqrt{\hat{p}_1(1 - \hat{p}_1)/n_{1\cdot} \ + \ \hat{p}_2(1 - \hat{p}_2)/n_{2\cdot}}$$

The computations are similar for the column 2 risks and risk difference.

Confidence Limits

By default, the RISKDIFF option provides Wald asymptotic confidence limits for the risks (row 1, row 2, and overall) and the risk difference. By default, the RISKDIFF option also provides exact (Clopper-Pearson) confidence limits for the risks. You can suppress the display of this information by specifying the NORISKS *riskdiff-option*. You can specify *riskdiff-options* to request tests and other types of confidence limits for the risk difference. For more information, see the sections "Confidence Limits for the Risk Difference" on page 3274 and "Risk Difference Tests" on page 3278.

The risks are equivalent to the binomial proportions of their corresponding rows. This section describes the Wald confidence limits that are provided by default when you specify the RISKDIFF option. The BINOMIAL option provides additional confidence limit types and tests for risks (binomial proportions). For more information, see the sections "Binomial Confidence Limits" on page 3263 and "Binomial Tests" on page 3267.

The Wald confidence limits are based on the normal approximation to the binomial distribution. PROC FREQ computes the Wald confidence limits for the risks and risk differences as

$$\mathrm{Est} \ \pm \ (\ z_{\alpha/2} \times \mathrm{se}(\mathrm{Est}) \ )$$

where Est is the estimate, $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution, and se(Est) is the standard error of the estimate. The confidence level α is determined by the value of the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

If you specify the CORRECT *riskdiff-option*, PROC FREQ includes continuity corrections in the Wald confidence limits for the risks and risk differences. The purpose of a continuity correction is to adjust for the difference between the normal approximation and the binomial distribution, which is discrete. See Fleiss, Levin, and Paik (2003) for more information. The continuity-corrected Wald confidence limits are computed as

$$\text{Est} \ \pm \ (\ z_{\alpha/2} \times \text{se}(\text{Est}) + c \ )$$

where c is the continuity correction. For the row 1 risk, $c = (1/2n_{1\cdot})$; for the row 2 risk, $c = (1/2n_{2\cdot})$; for the overall risk, $c = (1/2n)$; and for the risk difference, $c = ((1/n_{1\cdot} + 1/n_{2\cdot})/2)$. The column 1 and column 2 risks use the same continuity correction.

By default when you specify the RISKDIFF option, PROC FREQ also provides exact (Clopper-Pearson) confidence limits for the column 1, column 2, and overall risks. These confidence limits are constructed by inverting the equal-tailed test that is based on the binomial distribution. For more information, see the section "Exact (Clopper-Pearson) Confidence Limits" on page 3264.

Confidence Limits for the Risk Difference PROC FREQ provides the following confidence limit types for the risk difference: Agresti-Caffo, exact unconditional, Hauck-Anderson, Miettinen-Nurminen (score), Newcombe (hybrid-score), and Wald confidence limits. Continuity-corrected forms of Newcombe and Wald confidence limits are also available.

The confidence coefficient for the confidence limits produced by the CL= *riskdiff-option* is $100(1-\alpha)\%$, where the value of α is determined by the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits. This differs from the test-based confidence limits that are provided with the equivalence, noninferiority, and superiority tests, which have a confidence coefficient of $100(1-2\alpha)\%$ (Schuirmann 1999). For more information, see the section "Risk Difference Tests" on page 3278.

Agresti-Caffo Confidence Limits

Agresti-Caffo confidence limits for the risk difference are computed as

$$\tilde{d} \ \pm \ (\ z_{\alpha/2} \times \text{se}(\tilde{d}) \ )$$

where $\tilde{d} = \tilde{p}_1 - \tilde{p}_2$, $\tilde{p}_i = (n_{i1} + 1)/(n_{i\cdot} + 2)$,

$$\text{se}(\tilde{d}) = \sqrt{\tilde{p}_1(1-\tilde{p}_2)/(n_{1\cdot} + 2) \ + \ \tilde{p}_2(1-\tilde{p}_2)/(n_{2\cdot} + 2)}$$

and $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution.

The Agresti-Caffo interval adjusts the Wald interval for the risk difference by adding a pseudo-observation of each type (success and failure) to each sample. See Agresti and Caffo (2000) and Agresti and Coull (1998) for more information.

Hauck-Anderson Confidence Limits

Hauck-Anderson confidence limits for the risk difference are computed as

$$\hat{d} \ \pm \ (\ c \ + \ z_{\alpha/2} \times \text{se}(\hat{d}) \ )$$

where $\hat{d} = \hat{p}_1 - \hat{p}_2$ and $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. The standard error is computed from the sample proportions as

$$\mathrm{se}(\hat{d}) = \sqrt{\hat{p}_1(1-\hat{p}_1)/(n_{1\cdot}-1) \;+\; \hat{p}_2(1-\hat{p}_2)/(n_{2\cdot}-1)}$$

The Hauck-Anderson continuity correction c is computed as

$$c = 1 \;/\; \big(2\ \min(n_{1\cdot},\ n_{2\cdot})\big)$$

For more information, see Hauck and Anderson (1986). The subsection "Hauck-Anderson Test" in the section "Noninferiority Tests" on page 3279 describes the corresponding noninferiority test.

Miettinen-Nurminen (Score) Confidence Limits

Miettinen-Nurminen (score) confidence limits for the risk difference (Miettinen and Nurminen 1985) are computed by inverting score tests for the risk difference. A score-based test statistic for the null hypothesis that the risk difference equals δ can be expressed as

$$T(\delta) = (\hat{d} - \delta)/\sqrt{\widetilde{\mathrm{Var}}(\delta)}$$

where $\hat{d}$ is the observed value of the risk difference $(\hat{p}_1 - \hat{p}_2)$,

$$\widetilde{\mathrm{Var}}(\delta) = (n/(n-1))\ \left(\ \tilde{p}_1(\delta)(1-\tilde{p}_1(\delta))/n_1 + \tilde{p}_2(\delta)(1-\tilde{p}_2(\delta))/n_2\ \right)$$

and $\tilde{p}_1(\delta)$ and $\tilde{p}_2(\delta)$ are the maximum likelihood estimates of the row 1 and row 2 risks (proportions) under the restriction that the risk difference is δ. For more information, see Miettinen and Nurminen (1985, pp. 215–216) and Miettinen (1985, chapter 12).

The $100(1-\alpha)\%$ confidence interval for the risk difference consists of all values of δ for which the score test statistic $T(\delta)$ falls in the acceptance region,

$$\{\delta : T(\delta) < z_{\alpha/2}\}$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. PROC FREQ finds the confidence limits by iterative computation, which stops when the iteration increment falls below the convergence criterion or when the maximum number of iterations is reached, whichever occurs first. By default, the convergence criterion is 0.00000001 and the maximum number of iterations is 100.

By default, the Miettinen-Nurminen confidence limits include the bias correction factor $n/(n-1)$ in the computation of $\widetilde{\mathrm{Var}}(\delta)$ (Miettinen and Nurminen 1985, p. 216). For more information, see Newcombe and Nurminen (2011). If you specify the CL=MN(CORRECT=NO) *riskdiff-option*, PROC FREQ does not include the bias correction factor in this computation (Mee 1984). See also Agresti (2002, p. 77). The uncorrected confidence limits are labeled as "Miettinen-Nurminen-Mee" confidence limits in the displayed output.

The maximum likelihood estimates of p_1 and p_2, subject to the constraint that the risk difference is δ, are computed as

$$\tilde{p}_1 = 2u\cos(w) - b/3a \quad \text{and} \quad \tilde{p}_2 = \tilde{p}_1 - \delta$$

where

$$
\begin{aligned}
w &= (\pi + \cos^{-1}(v/u^3))/3 \\
v &= b^3/(3a)^3 - bc/6a^2 + d/2a \\
u &= \mathrm{sign}(v)\sqrt{b^2/(3a)^2 - c/3a} \\
a &= 1 + \theta \\
b &= -(1 + \theta + \hat{p}_1 + \theta\hat{p}_2 + \delta(\theta + 2)) \\
c &= \delta^2 + \delta(2\hat{p}_1 + \theta + 1) + \hat{p}_1 + \theta\hat{p}_2 \\
d &= -\hat{p}_1\delta(1 + \delta) \\
\theta &= n_{2\cdot}/n_{1\cdot}
\end{aligned}
$$

For more information, see Farrington and Manning (1990, p. 1453).

Newcombe Confidence Limits

Newcombe (hybrid-score) confidence limits for the risk difference are constructed from the Wilson score confidence limits for each of the two individual proportions. The confidence limits for the individual proportions are used in the standard error terms of the Wald confidence limits for the proportion difference. See Newcombe (1998a) and Barker et al. (2001) for more information.

Wilson score confidence limits for p_1 and p_2 are the roots of

$$|p_i - \hat{p}_i| = z_{\alpha/2}\sqrt{p_i(1 - p_i)/n_{i\cdot}}$$

for $i = 1, 2$. The confidence limits are computed as

$$\left(\hat{p}_i \;+\; z^2_{\alpha/2}/2n_{i\cdot} \;\pm\; z_{\alpha/2}\sqrt{\left(\hat{p}_i(1 - \hat{p}_i) + z^2_{\alpha/2}/4n_{i\cdot}\right)/n_{i\cdot}}\right) \;/\; \left(1 + z^2_{\alpha/2}/n_{i\cdot}\right)$$

For more information, see the section "Wilson (Score) Confidence Limits" on page 3266.

Denote the lower and upper Wilson score confidence limits for p_1 as L_1 and U_1, and denote the lower and upper confidence limits for p_2 as L_2 and U_2. The Newcombe confidence limits for the proportion difference ($d = p_1 - p_2$) are computed as

$$d_L = (\hat{p}_1 - \hat{p}_2) \;-\; \sqrt{(\hat{p}_1 - L_1)^2 \;+\; (U_2 - \hat{p}_2)^2}$$

$$d_U = (\hat{p}_1 - \hat{p}_2) \;+\; \sqrt{(U_1 - \hat{p}_1)^2 \;+\; (\hat{p}_2 - L_2)^2}$$

If you specify the CORRECT *riskdiff-option*, PROC FREQ provides continuity-corrected Newcombe confidence limits. By including a continuity correction of $1/2n_{i\cdot}$, the Wilson score confidence limits for the individual proportions are computed as the roots of

$$|p_i - \hat{p}_i| - 1/2n_{i\cdot} = z_{\alpha/2}\sqrt{p_i(1 - p_i)/n_{i\cdot}}$$

The continuity-corrected confidence limits for the individual proportions are then used to compute the proportion difference confidence limits d_L and d_U.

Wald Confidence Limits

Wald confidence limits for the risk difference are computed as

$$\hat{d} \pm (z_{\alpha/2} \times \mathrm{se}(\hat{d}))$$

where $\hat{d} = \hat{p}_1 - \hat{p}_2$, $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. and the standard error is computed from the sample proportions as

$$\mathrm{se}(\hat{d}) = \sqrt{\hat{p}_1(1-\hat{p}_1)/n_{1\cdot} + \hat{p}_2(1-\hat{p}_2)/n_{2\cdot}}$$

If you specify the CORRECT *riskdiff-option*, the Wald confidence limits include a continuity correction c,

$$\hat{d} \pm (c + z_{\alpha/2} \times \mathrm{se}(\hat{d}))$$

where $c = (1/n_{1\cdot} + 1/n_{2\cdot})/2$.

The subsection "Wald Test" in the section "Noninferiority Tests" on page 3279 describes the corresponding noninferiority test.

Exact Unconditional Confidence Limits

If you specify the RISKDIFF option in the EXACT statement, PROC FREQ provides exact unconditional confidence limits for the risk difference ($d = p_1 - p_2$). The exact unconditional approach fixes the row margins of the 2×2 table and eliminates the nuisance parameter p_2 by using the maximum p-value (worst-case scenario) over all possible values of p_2 (Santner and Snell 1980). The conditional approach, which is described in the section "Exact Statistics" on page 3318, does not apply to the risk difference because of the nuisance parameter (Agresti 1992).

By default, PROC FREQ computes the confidence limits by the tail method, which inverts two separate one-sided exact tests of the risk difference, where the tests are based on the score statistic (Chan and Zhang 1999). The size of each one-sided exact test is at most $\alpha/2$, and the confidence coefficient is at least $(1-\alpha)$. If you specify the RISKDIFF(METHOD=NOSCORE) option in the EXACT statement, PROC FREQ computes the confidence limits by inverting two separate one-sided exact tests that are based on the unstandardized risk difference. If you specify the RISKDIFF(METHOD=SCORE2) option in the EXACT statement, PROC FREQ computes the confidence limits by inverting a single two-sided exact test that is based on the score statistic (Agresti and Min 2001).

The score statistic is a less discrete statistic than the unstandardized risk difference and produces less conservative confidence limits (Agresti and Min 2001). For more information, see Santner et al. (2007). The section "Miettinen-Nurminen (Score) Confidence Limits" describe computation of the risk difference score statistic. For more information, see Miettinen and Nurminen (1985) and Farrington and Manning (1990).

PROC FREQ computes the exact unconditional confidence limits as follows. The risk difference is defined as the difference between the row 1 and row 2 risks (proportions), $d = p_1 - p_2$, and n_1 and n_2 denote the row totals of the 2×2 table. The joint probability function for the table can be expressed in terms of the table cell frequencies, the risk difference, and the nuisance parameter p_2 as

$$f(n_{11}, n_{21}; n_1, n_2, d, p_2) = \binom{n_1}{n_{11}} (d + p_2)^{n_{11}} (1 - d - p_2)^{n_1 - n_{11}} \times \binom{n_2}{n_{21}} p_2^{n_{21}} (1 - p_2)^{n_2 - n_{21}}$$

For the tail method (which inverts two separate one-sided exact tests), the $100(1-\alpha/2)\%$ confidence limits for the risk difference are computed as

$$\begin{aligned} d_L &= \sup\,(d_* : P_U(d_*) > \alpha/2) \\ d_U &= \inf\,(d_* : P_L(d_*) > \alpha/2) \end{aligned}$$

where

$$\begin{aligned} P_U(d_*) &= \sup_{p_2}\Big(\sum_{A,T(a)\geq t_0} f(n_{11},n_{21};n_1,n_2,d_*,p_2)\Big) \\ P_L(d_*) &= \sup_{p_2}\Big(\sum_{A,T(a)\leq t_0} f(n_{11},n_{21};n_1,n_2,d_*,p_2)\Big) \end{aligned}$$

The set A includes all 2×2 tables in which the row sums are n_1 and n_2, $T(a)$ denotes the value of the test statistic for table a in A, and t_0 is the value of the test statistic for the observed table. The test statistic is either the score statistic (by default) or the unstandardized risk difference. To compute $P_U(d_*)$, the sum includes probabilities of those tables for which $(T(a) \geq t_0)$. For a fixed value of d_*, $P_U(d_*)$ is defined as the maximum sum over all possible values of p_2.

The two-sided score method evaluates the p-values $P_U(d_*)$ and $P_L(d_*)$ by comparing $|T(a)|$ to $|t_0|$. To compute the confidence limits d_L and d_u, the two-sided method compares the p-values to α. For more information, see Agresti and Min (2001) and Santner et al. (2007).

Risk Difference Tests

PROC FREQ provides tests of equality, noninferiority, superiority, and equivalence for the risk (proportion) difference. The following analysis methods are available: Wald (with and without continuity correction), Hauck-Anderson, Farrington-Manning (score), and Newcombe (with and without continuity correction). You can specify the method by using the METHOD= *riskdiff-option*; by default, PROC FREQ provides Wald tests.

Equality Tests The equality test for the risk difference tests the null hypothesis that the risk difference equals the null value. You can specify a null value by using the EQUAL(NULL=) *riskdiff-option*; by default, the null value is 0. This test can be expressed as $H_0{:}\,d = d_0$ versus the alternative $H_a{:}\,d \neq d_0$, where $d = p_1 - p_2$ denotes the risk difference (for column 1 or column 2) and d_0 denotes the null value.

The test statistic is computed as

$$z = (\hat{d} - d_0)/\mathrm{se}(\hat{d})$$

where the standard error $\mathrm{se}(\hat{d})$ is computed by using the method that you specify. Available methods for the equality test include Wald (with and without continuity correction), Hauck-Anderson, and Farrington-Manning (score). For a description of the standard error computation, see the subsections "Wald Test," "Hauck-Anderson Test," and "Farrington-Manning (Score) Test," respectively, in the section "Noninferiority Tests" on page 3279.

PROC FREQ computes one-sided and two-sided p-values for equality tests. When the test statistic z is greater than 0, PROC FREQ displays the right-sided p-value, which is the probability of a larger value occurring under the null hypothesis. The one-sided p-value can be expressed as

$$P_1 = \begin{cases} \mathrm{Prob}(Z > z) & \text{if} \quad z > 0 \\ \mathrm{Prob}(Z < z) & \text{if} \quad z \leq 0 \end{cases}$$

where Z has a standard normal distribution. The two-sided p-value is computed as $P_2 = 2 \times P_1$.

Noninferiority Tests If you specify the NONINF *riskdiff-option*, PROC FREQ provides a noninferiority test for the risk difference, or the difference between two proportions. The null hypothesis for the noninferiority test is

$$H_0\colon p_1 - p_2 \leq -\delta$$

versus the alternative

$$H_a\colon p_1 - p_2 > -\delta$$

where δ is the noninferiority margin. Rejection of the null hypothesis indicates that the row 1 risk is not inferior to the row 2 risk. See Chow, Shao, and Wang (2003) for more information.

You can specify the value of δ with the MARGIN= *riskdiff-option*. By default, $\delta = 0.2$. You can specify the test method with the METHOD= *riskdiff-option*. The following methods are available for the risk difference noninferiority analysis: Wald (with and without continuity correction), Hauck-Anderson, Farrington-Manning (score), and Newcombe (with and without continuity correction). The Wald, Hauck-Anderson, and Farrington-Manning methods provide tests and corresponding test-based confidence limits; the Newcombe method provides only confidence limits. If you do not specify METHOD=, PROC FREQ uses the Wald test by default.

The confidence coefficient for the test-based confidence limits is $100(1 - 2\alpha)\%$ (Schuirmann 1999). By default, if you do not specify the ALPHA= option, these are 90% confidence limits. You can compare the confidence limits to the noninferiority limit, $-\delta$.

The following sections describe the noninferiority analysis methods for the risk difference.

Wald Test

If you specify the METHOD=WALD *riskdiff-option*, PROC FREQ provides an asymptotic Wald test of noninferiority for the risk difference. This is also the default method. The Wald test statistic is computed as

$$z = (\hat{d} + \delta) \;/\; \mathrm{se}(\hat{d})$$

where $(\hat{d} = \hat{p}_1 - \hat{p}_2)$ estimates the risk difference and δ is the noninferiority margin.

By default, the standard error for the Wald test is computed from the sample proportions as

$$\mathrm{se}(\hat{d}) = \sqrt{\hat{p}_1(1 - \hat{p}_1)/n_{1\cdot} \;+\; \hat{p}_2(1 - \hat{p}_2)/n_{2\cdot}}$$

If you specify the VAR=NULL *riskdiff-option*, the standard error is based on the null hypothesis that the risk difference equals $-\delta$ (Dunnett and Gent 1977). The standard error is computed as

$$\mathrm{se}(\hat{d}) = \sqrt{\tilde{p}(1 - \tilde{p})/n_{2\cdot} \;+\; (\tilde{p} - \delta)(1 - \tilde{p} + \delta)/n_{1\cdot}}$$

where

$$\tilde{p} = (n_{11} + n_{21} + \delta n_{1\cdot})/n$$

If you specify the CORRECT *riskdiff-option*, the test statistic includes a continuity correction. The continuity correction is subtracted from the numerator of the test statistic if the numerator is greater than 0; otherwise, the continuity correction is added to the numerator. The value of the continuity correction is $(1/n_{1\cdot} + 1/n_{2\cdot})/2$.

The p-value for the Wald noninferiority test is $P_z = \mathrm{Prob}(Z > z)$, where Z has a standard normal distribution.

Hauck-Anderson Test

If you specify the METHOD=HA *riskdiff-option*, PROC FREQ provides the Hauck-Anderson test for noninferiority. The Hauck-Anderson test statistic is computed as

$$z = (\hat{d} + \delta \pm c) \,/\, \mathrm{se}(\hat{d})$$

where $\hat{d} = \hat{p}_1 - \hat{p}_2$ and the standard error is computed from the sample proportions as

$$\mathrm{se}(\hat{d}) = \sqrt{\hat{p}_1(1-\hat{p}_1)/(n_{1\cdot} - 1) \;+\; \hat{p}_2(1-\hat{p}_2)/(n_{2\cdot} - 1)}$$

The Hauck-Anderson continuity correction c is computed as

$$c = 1 \,/\, \big(2 \min(n_{1\cdot}, n_{2\cdot})\big)$$

The p-value for the Hauck-Anderson noninferiority test is $P_z = \mathrm{Prob}(Z > z)$, where Z has a standard normal distribution. See Hauck and Anderson (1986) and Schuirmann (1999) for more information.

Farrington-Manning (Score) Test

If you specify the METHOD=FM *riskdiff-option*, PROC FREQ provides the Farrington-Manning (score) test of noninferiority for the risk difference. A score test statistic for the null hypothesis that the risk difference equals $-\delta$ can be expressed as

$$z = (\hat{d} + \delta) \,/\, \mathrm{se}(\hat{d})$$

where $\hat{d}$ is the observed value of the risk difference $(\hat{p}_1 - \hat{p}_2)$,

$$\mathrm{se}(\hat{d}) = \sqrt{\tilde{p}_1(1-\tilde{p}_1)/n_{1\cdot} \;+\; \tilde{p}_2(1-\tilde{p}_2)/n_{2\cdot}}.$$

and $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of the row 1 and row 2 risks (proportions) under the restriction that the risk difference is $-\delta$. The p-value for the noninferiority test is $P_z = \mathrm{Prob}(Z > z)$, where Z has a standard normal distribution. For more information, see Miettinen and Nurminen (1985); Miettinen (1985); Farrington and Manning (1990); Dann and Koch (2005).

The maximum likelihood estimates of p_1 and p_1, subject to the constraint that the risk difference is $-\delta$, are computed as

$$\tilde{p}_1 = 2u\cos(w) - b/3a \quad \text{and} \quad \tilde{p}_2 = \tilde{p}_1 + \delta$$

where

$$\begin{aligned}
w &= (\pi + \cos^{-1}(v/u^3))/3 \\
v &= b^3/(3a)^3 - bc/6a^2 + d/2a \\
u &= \mathrm{sign}(v)\sqrt{b^2/(3a)^2 - c/3a} \\
a &= 1 + \theta \\
b &= -(1 + \theta + \hat{p}_1 + \theta\hat{p}_2 - \delta(\theta + 2)) \\
c &= \delta^2 - \delta(2\hat{p}_1 + \theta + 1) + \hat{p}_1 + \theta\hat{p}_2 \\
d &= \hat{p}_1\delta(1-\delta) \\
\theta &= n_{2\cdot}/n_{1\cdot}
\end{aligned}$$

For more information, see Farrington and Manning (1990, p. 1453).

Newcombe Noninferiority Analysis

If you specify the METHOD=NEWCOMBE *riskdiff-option*, PROC FREQ provides a noninferiority analysis that is based on Newcombe hybrid-score confidence limits for the risk difference. The confidence coefficient for the confidence limits is $100(1-2\alpha)\%$ (Schuirmann 1999). By default, if you do not specify the ALPHA= option, these are 90% confidence limits. You can compare the confidence limits with the noninferiority limit, $-\delta$. If you specify the CORRECT *riskdiff-option*, the confidence limits includes a continuity correction. See the subsection "Newcombe Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274 for more information.

Superiority Test If you specify the SUP *riskdiff-option*, PROC FREQ provides a superiority test for the risk difference. The null hypothesis is

$$H_0:: p_1 - p_2 \leq \delta$$

versus the alternative

$$H_a: p_1 - p_2 > \delta$$

where δ is the superiority margin. Rejection of the null hypothesis indicates that the row 1 proportion is superior to the row 2 proportion. You can specify the value of δ with the MARGIN= *riskdiff-option*. By default, $\delta = 0.2$.

The superiority analysis is identical to the noninferiority analysis but uses a positive value of the margin δ in the null hypothesis. The superiority computations follow those in the section "Noninferiority Tests" on page 3279 by replacing $-\delta$ by δ. See Chow, Shao, and Wang (2003) for more information.

Equivalence Test If you specify the EQUIV *riskdiff-option*, PROC FREQ provides an equivalence test for the risk difference, or the difference between two proportions. The null hypothesis for the equivalence test is

$$H_0: p_1 - p_2 \leq -\delta_L \quad \text{or} \quad p_1 - p_2 \geq \delta_U$$

versus the alternative

$$H_a: \delta_L < p_1 - p_2 < \delta_U$$

where δ_L is the lower margin and δ_U is the upper margin. Rejection of the null hypothesis indicates that the two binomial proportions are equivalent. See Chow, Shao, and Wang (2003) for more information.

You can specify the value of the margins δ_L and δ_U with the MARGIN= *riskdiff-option*. If you do not specify MARGIN=, PROC FREQ uses lower and upper margins of –0.2 and 0.2 by default. If you specify a single margin value δ, PROC FREQ uses lower and upper margins of $-\delta$ and δ. You can specify the test method with the METHOD= *riskdiff-option*. The following methods are available for the risk difference equivalence analysis: Wald (with and without continuity correction), Hauck-Anderson, Farrington-Manning (score), and Newcombe (with and without continuity correction). The Wald, Hauck-Anderson, and Farrington-Manning methods provide tests and corresponding test-based confidence limits; the Newcombe method provides only confidence limits. If you do not specify METHOD=, PROC FREQ uses the Wald test by default.

PROC FREQ computes two one-sided tests (TOST) for equivalence analysis (Schuirmann 1987). The TOST approach includes a right-sided test for the lower margin δ_L and a left-sided test for the upper margin δ_U. The overall p-value is taken to be the larger of the two p-values from the lower and upper tests.

The section "Noninferiority Tests" on page 3279 gives details about the Wald, Hauck-Anderson, Farrington-Manning (score), and Newcombe methods for the risk difference. The lower margin equivalence test statistic takes the same form as the noninferiority test statistic but uses the lower margin value δ_L in place of $-\delta$. The upper margin equivalence test statistic take the same form as the noninferiority test statistic but uses the upper margin value δ_U in place of $-\delta$.

The test-based confidence limits for the risk difference are computed according to the equivalence test method that you select. If you specify METHOD=WALD with VAR=NULL, or METHOD=FM, separate standard errors are computed for the lower and upper margin tests. In this case, the test-based confidence limits are computed by using the maximum of these two standard errors. These confidence limits have a confidence coefficient of $100(1-2\alpha)\%$ (Schuirmann 1999). By default, if you do not specify the ALPHA= option, these are 90% confidence limits. You can compare the test-based confidence limits to the equivalence limits, (δ_L, δ_U).

Barnard's Unconditional Exact Test

The BARNARD option in the EXACT statement provides an unconditional exact test for the risk (proportion) difference for 2×2 tables. The reference set for the unconditional exact test consists of all 2×2 tables that have the same row sums as the observed table (Barnard 1945, 1947, 1949). This differs from the reference set for exact conditional inference, which is restricted to the set of tables that have the same row sums and the same column sums as the observed table. See the sections "Fisher's Exact Test" on page 3252 and "Exact Statistics" on page 3318 for more information.

The test statistic is the standardized risk difference, which is computed as

$$T = d/\sqrt{p_{\cdot 1}(1-p_{\cdot 1})(1/n_1 + 1/n_2)}$$

where the risk difference d is defined as the difference between the row 1 and row 2 risks (proportions), $d = (n_{11}/n_1 - n_{21}/n_2)$; n_1 and n_2 are the row 1 and row 2 totals, respectively; and $p_{\cdot 1}$ is the overall proportion in column 1, $(n_{11} + n_{21})/n$.

Under the null hypothesis that the risk difference is 0, the joint probability function for a table can be expressed in terms of the table cell frequencies, the row totals, and the unknown parameter π as

$$f(n_{11}, n_{21}; n_1, n_2, \pi) = \binom{n_1}{n_{11}}\binom{n_2}{n_{21}}\pi^{n_{11}+n_{21}}(1-\pi)^{n-n_{11}-n_{21}}$$

where π is the common value of the risk (proportion).

PROC FREQ sums the table probabilities over the reference set for those tables where the test statistic is greater than or equal to the observed value of the test statistic. This sum can be expressed as

$$\mathrm{Prob}(\pi) = \sum_{A, T(a) \geq t_0} f(n_{11}, n_{21}; n_1, n_2, \pi)$$

where the set A contains all 2×2 tables with row sums equal to n_1 and n_2, and $T(a)$ denotes the value of the test statistic for table a in A. The sum includes probabilities of those tables for which $(T(a) \geq t_0)$, where t_0 is the value of the test statistic for the observed table.

The sum $\mathrm{Prob}(\pi)$ depends on the unknown value of π. To compute the exact p-value, PROC FREQ eliminates the nuisance parameter π by taking the maximum value of $\mathrm{Prob}(\pi)$ over all possible values of π,

$$\mathrm{Prob} = \sup_{(0 \leq \pi \leq 1)} (\mathrm{Prob}(\pi))$$

See Suissa and Shuster (1985) and Mehta and Senchaudhuri (2003).

Common Risk Difference

If you specify the COMMONRISKDIFF option in the TABLES statement, PROC FREQ provides estimates, confidence limits, and tests for the common (overall) risk difference for multiway 2×2 tables.

Mantel-Haenszel Confidence Limits and Test

PROC FREQ computes the Mantel-Haenszel estimate, confidence limits, and test for the common risk difference by using Mantel-Haenszel stratum weights (Mantel and Haenszel 1959) and the Sato variance estimator (Sato 1989). The Mantel-Haenszel estimate of the common risk difference is

$$\widehat{d}_{\mathrm{MH}} = \sum_h \widehat{d}_h \, w_h$$

where $\widehat{d}_h$ is the risk difference in stratum h and

$$w_h = \frac{n_{h1\cdot} n_{h2\cdot}}{n_h} \Big/ \sum_i \frac{n_{i1\cdot} n_{i2\cdot}}{n_i}$$

is the Mantel-Haenszel weight of stratum h. The column 1 risk difference in stratum (2×2 table) h is computed as

$$\widehat{d}_h = \hat{p}_{h1} - \hat{p}_{h2} = (n_{h11}/n_{h1\cdot}) - (n_{h21}/n_{h2\cdot})$$

where $\hat{p}_{h1}$ is the proportion of row 1 observations that are classified in column 1 and $\hat{p}_{h2}$ is the proportion or row 2 observations that are classified in column 1. The column 2 risk is computed in the same way. For more information, see Agresti (2013, p. 231).

PROC FREQ computes the variance of $\widehat{d}_{\mathrm{MH}}$ (Sato 1989) as

$$\widehat{\sigma}^2(\widehat{d}_{\mathrm{MH}}) = \left(\widehat{d}_{\mathrm{MH}} \sum_h P_h + \sum_h Q_h \right) \Big/ \left(\sum_h n_{h1\cdot} n_{h2\cdot} / n_h \right)^2$$

where

$$P_h = \left(n_{h1\cdot}^2 n_{h21} - n_{h2\cdot}^2 n_{h11} + n_{h1\cdot} n_{h2\cdot} (n_{h2\cdot} - n_{h1\cdot})/2 \right) / n_h^2$$

$$Q_h = \left(n_{h11} (n_{h2\cdot} - n_{h21}) + n_{h21} (n_{h1\cdot} - n_{h11}) \right) / 2 n_h$$

The $100(1-\alpha)\%$ confidence limits for the common risk difference are

$$\widehat{d}_{\mathrm{MH}} \pm \left(z_{\alpha/2} \times \widehat{\sigma}(\widehat{d}_{\mathrm{MH}}) \right)$$

If you specify the COMMONRISKDIFF(TEST=MH) option, PROC FREQ provides a Mantel-Haenszel test of the null hypothesis that the common risk difference is 0, which is computed as $z_{\mathrm{MH}} = \widehat{d}_{\mathrm{MH}} \,/\, \widehat{\sigma}(\widehat{d}_{\mathrm{MH}})$. The two-sided p-value is $\mathrm{Prob}(|Z| > |z_{\mathrm{MH}}|)$, where Z has a standard normal distribution.

Klingenberg Confidence Limits

Klingenberg confidence limits (Klingenberg 2014) for the Mantel-Haenszel common risk difference are based on inverting a test of homogeneity that uses the null form of the Sato variance estimator (Sato 1989). For performance evaluation of Klingenberg confidence limits, see Fisher (2015) and Klingenberg (2014).

The $100(1-\alpha)\%$ Klingenberg confidence limits for the common risk difference are

$$\widehat{d}_{\mathrm{Mid}} \pm M_{\alpha/2}$$

where M (margin of error) is computed as

$$M_{\alpha/2} = \sqrt{\widehat{d}^2_{\mathrm{Mid}} - \widehat{d}^2_{\mathrm{MH}} + z^2_{\alpha/2}\,(Q/W^2)}$$

and the confidence interval midpoint is computed as

$$\widehat{d}_{\mathrm{Mid}} = \widehat{d}_{\mathrm{MH}} + 0.5\, z^2_{\alpha/2}\,(P/W^2)$$

The values P, Q, and W are computed as

$$\begin{aligned} P &= \sum_h P_h \\ Q &= \sum_h Q_h \\ W &= \sum_h n_{h1\cdot}\, n_{h2\cdot} \,/\, n_h \end{aligned}$$

where h denotes the stratum, and P_h and Q_h are defined in the section "Mantel-Haenszel Confidence Limits and Test" on page 3283.

Minimum Risk Confidence Limits and Test

PROC FREQ computes the minimum risk estimate, confidence limits, and test for the common risk difference by using the method of Mehrotra and Railkar (2000). The stratum estimates are weighted by minimum risk weights, which minimize the mean square error of the estimate of the common risk difference. Minimum risk weights are designed to improve precision and reduce bias (compared to other weighting strategies) and can minimize the power loss that can occur when underlying assumptions are not met. For more information, see Mehrotra (2001) and Dmitrienko et al. (2005, section 1.3.3).

The minimum risk estimate of the common risk difference is

$$\widehat{d}_{\mathrm{MR}} = \sum_h \widehat{d}_h\, w^*_h$$

where $\widehat{d}_h$ is the risk difference in stratum h and w^*_h is the minimum risk weight of stratum h (which is described in the section "Minimum Risk Weights" on page 3285). The variance of $\widehat{d}_{\mathrm{MR}}$ is estimated by

$$\widehat{V}(\widehat{d}_{\mathrm{MR}}) = \sum_h w^{*2}_h\, \widehat{V}_h$$

where $\widehat{V}_h$ (the variance estimate of the stratum h risk difference) is computed as

$$\widehat{V}_h = \hat{p}_{h1}(1-\hat{p}_{h1})/n_{h1\cdot} + \hat{p}_{h2}(1-\hat{p}_{h2})/n_{h2\cdot}$$

The $100(1-\alpha)\%$ minimum risk confidence limits for the common risk difference are

$$\widehat{d}_{\mathrm{MR}} \pm \left(c + z_{\alpha/2}\sqrt{\widehat{V}(\widehat{d}_{\mathrm{MR}})}\right)$$

where the continuity correction is

$$c = 0.1875 \,/\, \sum_h (n_{h1\cdot} n_{h2\cdot}/n_h)$$

The continuity correction is applied only when $c < |\widehat{d}_{\mathrm{MR}}|$ (Fleiss, Levin, and Paik 2003). You can remove the continuity correction by specifying the COMMONRISKDIFF(CORRECT=NO) option.

By default, the minimum risk test is computed as

$$z_{\mathrm{MR}} = \left(\widehat{d}_{\mathrm{MR}} \pm c\right) \,/\, \sqrt{\widehat{V}_0(\widehat{d}_{\mathrm{MR}})}$$

The continuity correction c is subtracted from $\widehat{d}_{\mathrm{MR}}$ if $\widehat{d}_{\mathrm{MR}} > 0$ and added to $\widehat{d}_{\mathrm{MR}}$ if $\widehat{d}_{\mathrm{MR}} < 0$. The null variance of the common risk difference is estimated by

$$\widehat{V}_0(\widehat{d}_{\mathrm{MR}}) = \sum_h w_h^{*2}\, \widehat{V}_0(\widehat{d}_h)$$

where $\widehat{V}_0(\widehat{d}_h)$ (an estimate of the variance of the stratum h risk difference under the null hypothesis) is

$$\widehat{V}_0(\widehat{d}_h) = \bar{p}_h(1-\bar{p}_h)\,(1/n_{h1\cdot} + 1/n_{h2\cdot})$$

and

$$\bar{p}_h = (n_{h1\cdot}\hat{p}_{h1\cdot} + n_{h2\cdot}\hat{p}_{h2\cdot}) \,/\, (n_{h1\cdot} + n_{h2\cdot})$$

The two-sided p-value is $\mathrm{Prob}(|Z| > |z_{\mathrm{MR}}|)$, where Z has a standard normal distribution.

If you specify the VAR=SAMPLE option for COMMONRISKDIFF(TEST=MR), PROC FREQ uses the sample variance estimate $\widehat{V}(\widehat{d}_{\mathrm{MR}})$ instead of the null variance estimate $\widehat{V}_0(\widehat{d}_{\mathrm{MR}})$ in the denominator of the test statistic z_{MR}. If you specify the COMMONRISKDIFF(CORRECT=NO) option, the continuity correction is not included in the test statistic.

Minimum Risk Weights The estimate of the minimum risk weight for stratum h is defined by Mehrotra and Railkar (2000) as

$$w_h^* = \frac{\beta_h}{\sum_i \widehat{V}_i^{-1}} - \left(\frac{\alpha_h \widehat{V}_h^{-1}}{\sum_i \widehat{V}_i^{-1} + \sum_i \alpha_i \widehat{d}_i \widehat{V}_i^{-1}}\right)\left(\frac{\sum_i \widehat{d}_i \beta_i}{\sum_i \widehat{V}_i^{-1}}\right)$$

where

$$\alpha_h = \widehat{d}_h \sum_i \widehat{V}_i^{-1} - \sum_i \widehat{d}_i \widehat{V}_i^{-1}$$

$$\beta_h = \widehat{V}_h^{-1}\left(1 + \alpha_h \sum_i f_i \widehat{d}_i\right)$$

and f_h is the fraction in stratum h

$$f_h = n_h \,/\, \sum_i n_h$$

All sums are over the s strata (2×2 tables) in the multiway table request, $\widehat{d}_i$ denotes the risk difference estimate in stratum i, and $\widehat{V}_i$ denotes the sample variance estimate of the risk difference in stratum i.

Summary Score Confidence Limits

PROC FREQ computes the summary score estimate of the common risk difference (Agresti 2013, p. 231) by using inverse-variance stratum weights and Miettinen-Nurminen (score) confidence limits for the stratum risk differences. For more information, see the section "Miettinen-Nurminen (Score) Confidence Limits."

The score confidence interval for the risk difference in stratum h can be expressed as $\widehat{d}'_h \pm z_{\alpha/2}\, s'_h$, where $\widehat{d}'_h$ is the midpoint of the score confidence interval and s'_h is the width of the confidence interval divided by $2z_{\alpha/2}$. The summary score estimate of the common risk difference is computed as

$$\widehat{d}_S = \sum_h \widehat{d}'_h\, w'_h$$

where

$$w'_h = (1/{s'_h}^2) \;/\; \sum_i (1/{s'_i}^2)$$

The variance of $\widehat{d}_S$ is computed as

$$\widehat{\sigma}^2(\widehat{d}_S) = 1 \;/\; \sum_h (1/{s'_h}^2)$$

The $100(1-\alpha)\%$ summary score confidence limits for the common risk difference are

$$\widehat{d}_S \;\pm\; \left(z_{\alpha/2} \times \widehat{\sigma}(\widehat{d}_S)\right)$$

If you specify the COMMONRISKDIFF(TEST=SCORE) option, PROC FREQ provides a summary score test of the null hypothesis that the common risk difference is 0. The test statistic is $z_S = \widehat{d}_S/\widehat{\sigma}(\widehat{d}_S)$ The two-sided p-value is $\mathrm{Prob}(|Z| > |z_S|)$ where Z has a standard normal distribution.

Stratified Newcombe Confidence Limits

PROC FREQ computes stratified Newcombe confidence limits for the common risk (proportion) difference by using the method of Yan and Su (2010). The stratified Newcombe confidence limits are constructed from stratified Wilson confidence limits for the common (overall) row proportions. By default, the strata are weighted by Mantel-Haenszel weights; if you specify the COMMONRISKDIFF(CL=NEWCOMBEMR) option, the strata are weighted by minimum risk weights.

PROC FREQ first computes individual Wilson confidence limits for the row proportions in each 2×2 table (stratum), as described in the section "Wilson (Score) Confidence Limits" on page 3266. These stratum Wilson confidence limits are then combined to form stratified Wilson confidence limits for the overall row proportions by using stratum weights (either Mantel-Haenszel or minimum risk). The confidence levels of the stratum Wilson confidence limits are chosen so that the overall confidence coefficient (for the stratified Wilson confidence limits) is $100(1-\alpha)\%$ (Yan and Su 2010).

Denote the lower and upper stratified Wilson score confidence limits for the common row 1 proportion as L_1 and U_1, respectively, and denote the lower and upper stratified Wilson confidence limits for the common row 2 proportion as L_2 and U_2, respectively. The $100(1-\alpha)\%$ stratified Newcombe confidence limits for the common risk (proportion) difference are

$$L = \widehat{d} - z_{\alpha/2}\sqrt{\lambda_1 L_1(1-L_1) + \lambda_2 U_2(1-U_2)}$$

$$U = \widehat{d} + z_{\alpha/2}\sqrt{\lambda_2 L_2(1-L_2) + \lambda_1 U_1(1-U_1)}$$

where $\widehat{d}$ is the weighted estimate of the common risk difference and

$$\lambda_1 = \sum_h w_h^2 \,/\, n_{h1\cdot}$$

$$\lambda_2 = \sum_h w_h^2 \,/\, n_{h2\cdot}$$

By default, the strata are weighted by Mantel-Haenszel weights, which are defined as

$$w_h = \frac{n_{h1\cdot} n_{h2\cdot}}{n_h} \,/\, \sum_i \frac{n_{i1\cdot} n_{i2\cdot}}{n_i}$$

and the weighted estimate of the common risk difference is $\widehat{d}_{\mathrm{MH}}$. For more information, see the section "Mantel-Haenszel Confidence Limits and Test" on page 3283. Optionally, the strata are weighted by minimum risk weights, and the weighted estimate of the common risk difference is $\widehat{d}_{\mathrm{MR}}$. For more information, see the section "Minimum Risk Confidence Limits and Test" on page 3284.

When there is a single stratum, the stratified Newcombe confidence interval is equivalent to the (unstratified) Newcombe confidence interval. For more information, see the subsection "Newcombe Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274. See also Kim and Won (2013).

Odds Ratio and Relative Risks

Odds Ratio

The odds ratio is a useful measure of association for a variety of study designs. For a retrospective design called a *case-control study*, the odds ratio can be used to estimate the relative risk when the probability of positive response is small (Agresti 2002). In a case-control study, two independent samples are identified based on a binary (yes-no) response variable, and the conditional distribution of a binary explanatory variable is examined within fixed levels of the response variable. For more information, see Stokes, Davis, and Koch (2012), Agresti (2013), and Agresti (2007).

The odds of a positive response (column 1) in row 1 is n_{11}/n_{12}. Similarly, the odds of a positive response in row 2 is n_{21}/n_{22}. The odds ratio is formed as the ratio of the row 1 odds to the row 2 odds. The odds ratio for a 2×2 table is defined as

$$\mathrm{OR} = \frac{n_{11}/n_{12}}{n_{21}/n_{22}} = \frac{n_{11}\, n_{22}}{n_{12}\, n_{21}}$$

The odds ratio can be any nonnegative number. When the row and column variables are independent, the true value of the odds ratio is 1. An odds ratio greater than 1 indicates that the odds of a positive response are higher in row 1 than in row 2. An odds ratio less than 1 indicates that the odds of a positive response are higher in row 2. The strength of association increases as the deviation from 1 increases.

The transformation $G = (\mathrm{OR} - 1)/(\mathrm{OR} + 1)$ transforms the odds ratio to the range (–1,1), where $G = 0$ when OR = 1; $G = -1$ when OR = 0; and G approaches 1 as OR approaches infinity. G is the gamma statistic, which PROC FREQ computes when you specify the MEASURES option.

Confidence Limits for the Odds Ratio The following types of confidence limits are available for the odds ratio: exact, exact mid-p, likelihood ratio, score, Wald, and Wald modified.

Wald Confidence Limits

The asymptotic Wald confidence limits are based on a log transformation of the odds ratio (Woolf 1955; Haldane 1956). PROC FREQ computes the Wald confidence limits as

$$\left(\mathrm{OR} \times \exp(-z\sqrt{v}),\ \ \mathrm{OR} \times \exp(z\sqrt{v}) \right)$$

where

$$v = \mathrm{Var}(\log(\mathrm{OR})) = 1/n_{11} + 1/n_{12} + 1/n_{21} + 1/n_{22}$$

and z is the $100(1-\alpha/2)$th percentile of the standard normal distribution. The confidence level α is determined by the ALPHA= option in the TABLES statement; by default, ALPHA=0.05, which produces 95% confidence limits for the odds ratio. If any of the four cell frequencies are 0, v is undefined and the Wald confidence limits cannot be computed. For more information, see Agresti (2013, p. 70).

Wald Modified Confidence Limits

PROC FREQ computes Wald modified confidence limits (Haldane 1956) for the odds ratio by replacing the n_{ij} by $(n_{ij} + 0.5)$ in the estimate and variance as follows:

$$\mathrm{OR}_m = \frac{(n_{11}+0.5)\ (n_{22}+0.5)}{(n_{12}+0.5)\ (n_{21}+0.5)}$$

$$v = \mathrm{Var}(\log(\mathrm{OR}_m)) = 1/(n_{11}+0.5) + 1/(n_{12}+0.5) + 1/(n_{21}+0.5) + 1/(n_{22}+0.5)$$

The modified confidence limits are computed as

$$\left(\mathrm{OR}_m \times \exp(-z\sqrt{v}),\ \ \mathrm{OR}_m \times \exp(z\sqrt{v}) \right)$$

where z is the $100(1-\alpha/2)$th percentile of the standard normal distribution. For more information, see Fleiss, Levin, and Paik (2003) and Agresti (2013).

Score Confidence Limits

Score confidence limits for the odds ratio (Miettinen and Nurminen 1985) are computed by inverting score tests for the odds ratio. A score-based chi-square test statistic for the null hypothesis that the odds ratio is θ can be expressed as

$$Q(\theta) = \{n_{1\cdot}\,(\hat{p}_1 - \tilde{p}_1)\}^2 \ /\ \{n/(n-1)\}\ \{1/\ (n_{1\cdot}\tilde{p}_1(1-\tilde{p}_1)) + 1/\ (n_{2\cdot}\tilde{p}_2(1-\tilde{p}_2))\}^{-1}$$

where $\hat{p}_1$ is the observed row 1 risk $(n_{11}/n_{1\cdot})$, and $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of the row 1 and row 2 risks under the restriction that the odds ratio $(n_{11}n_{22}/n_{12}n_{21})$ is θ. For more information, see Miettinen and Nurminen (1985) and Miettinen (1985, chapter 14).

The $100(1-\alpha)\%$ score confidence interval for the odds ratio consists of all values of θ for which the test statistic $Q(\theta)$ falls in the acceptance region,

$$\{\theta : Q(\theta) < \chi^2_{1,\alpha}\}$$

where $\chi^2_{1,\alpha}$ is the $100(1-\alpha)$th percentile of the chi-square distribution with 1 degree of freedom. For more information about score confidence limits, see Agresti (2013).

By default, the score confidence limits include the bias correction factor $n/(n-1)$ in the denominator of $Q(\theta)$ (Miettinen and Nurminen 1985, p. 217). If you specify the CL=SCORE(CORRECT=NO) option, PROC FREQ does not include this factor in the computation.

The maximum likelihood estimates of p_1 and p_2, subject to the constraint that the odds ratio is θ, are computed as

$$\tilde{p}_2 = \left(-b + \sqrt{b^2 - 4ac}\right)/2a \quad \text{and} \quad \tilde{p}_1 = \tilde{p}_2\theta/\left(1 + \tilde{p}_2(\theta - 1)\right)$$

where

$$\begin{aligned} a &= n_{2\cdot}(\theta - 1) \\ b &= n_{1\cdot}\theta + n_{2\cdot} - \hat{p}_{\cdot 1}(\theta - 1) \\ c &= -\hat{p}_{\cdot 1} \end{aligned}$$

For more information, see Miettinen and Nurminen (1985, pp. 217–218) and Miettinen (1985, chapter 14).

Likelihood Ratio Confidence Limits

Likelihood ratio (profile likelihood) confidence limits for the odds ratio are computed by inverting likelihood ratio tests. The likelihood ratio test statistic for the null hypothesis that the odds ratio is θ can be expressed as

$$G^2(\theta) = 2\left(n_{11}\log(\hat{p}_1/\tilde{p}_1) + n_{12}\log((1-\hat{p}_1)/(1-\tilde{p}_1)) + n_{21}\log(\hat{p}_2/\tilde{p}_2) + n_{22}\log((1-\hat{p}_2)/(1-\tilde{p}_2))\right)$$

where $\hat{p}_i$ is the observed row i risk $(n_{11}/n_{1\cdot})$ and $\tilde{p}_i$ is the maximum likelihood estimate of the row i risk under the restriction that the odds ratio is θ. The computation of the maximum likelihood estimates is described in the subsection "Score Confidence Limits" in this section. For more information, see Agresti (2013), Miettinen and Nurminen (1985), and Miettinen (1985, chapter 14).

The $100(1-\alpha)\%$ likelihood ratio confidence interval for the odds ratio consists of all values of θ for which the test statistic $G^2(\theta)$ falls in the acceptance region,

$$\{\theta : G^2(\theta) < \chi^2_{1,\alpha}\}$$

where $\chi^2_{1,\alpha}$ is the $100(1-\alpha)$th percentile of the chi-square distribution with 1 degree of freedom.

Exact Confidence Limits

PROC FREQ computes exact confidence limits for the odds ratio by inverting two one-sided (equal-tail) exact tests that are based on the noncentral hypergeometric distribution, where the distribution is conditional on the observed marginal totals of the 2×2 table. The exact confidence limits ϕ_1 and ϕ_2 are the solutions to the equations

$$\begin{aligned} \sum_{i=n_{11}}^{n_{\cdot 1}} f(i : n_{\cdot 1},\ n_{1\cdot},\ n_{2\cdot},\ \phi_1) &= \alpha/2 \\ \sum_{i=0}^{n_{11}} f(i : n_{\cdot 1},\ n_{1\cdot},\ n_{2\cdot},\ \phi_2) &= \alpha/2 \end{aligned}$$

where

$$f(i : n_{\cdot 1},\ n_{1\cdot},\ n_{2\cdot},\ \phi) = \binom{n_{1\cdot}}{i}\binom{n_{2\cdot}}{n_{\cdot 1} - i}\phi^i \ \Big/ \ \sum_{i=0}^{n_{\cdot 1}} \binom{n_{1\cdot}}{i}\binom{n_{2\cdot}}{n_{\cdot 1} - i}\phi^i$$

For more information, see Fleiss, Levin, and Paik (2003), Thomas (1971), and Gart (1971).

Because this is a discrete problem, the confidence coefficient for the exact confidence interval is not exactly $(1 - \alpha)$ but is at least $(1 - \alpha)$; thus, these confidence limits are conservative. For more information, see Agresti (1992).

When the odds ratio is 0, which occurs when either $n_{11} = 0$ or $n_{22} = 0$, PROC FREQ sets the lower exact confidence limit to 0 and determines the upper limit by using the level α (instead of $\alpha/2$). Similarly, when the odds ratio is infinity, which occurs when either $n_{12} = 0$ or $n_{21} = 0$, PROC FREQ sets the upper exact confidence limit to infinity and determines the lower limit by using level α.

Exact Mid-p Confidence Limits

PROC FREQ computes exact mid-*p* confidence limits for the odds ratio by inverting two one-sided hypergeometric tests that include mid-*p* tail areas. The mid-*p* approach replaces the probability of the observed table with half of that probability in the hypergeometric probability sums, which are described in the subsection "Exact Confidence Limits" in this section. The exact mid-*p* confidence limits ϕ_1 and ϕ_2 are the solutions to the equations

$$\begin{aligned}
\sum_{i=n_{11}+1}^{n_{\cdot 1}} \big(f(i : n_{\cdot 1},\ n_{1\cdot},\ n_{2\cdot},\ \phi_1)\big) \ + \ (1/2) f(n_{11} : n_{\cdot 1},\ n_{1\cdot},\ n_{2\cdot},\ \phi_1) \quad &= \quad \alpha/2 \\
\sum_{i=0}^{n_{11}-1} \big(f(i : n_{\cdot 1},\ n_{1\cdot},\ n_{2\cdot},\ \phi_2)\big) \ + \ (1/2) f(n_{11} : n_{\cdot 1},\ n_{1\cdot},\ n_{2\cdot},\ \phi_2) \quad &= \quad \alpha/2
\end{aligned}$$

where

$$f(i : n_{\cdot 1}, n_{1\cdot}, n_{2\cdot}, \phi) = \binom{n_{1\cdot}}{i}\binom{n_{2\cdot}}{n_{\cdot 1} - i}\phi^i \ \Big/ \ \sum_{i=0}^{n_{\cdot 1}} \binom{n_{1\cdot}}{i}\binom{n_{2\cdot}}{n_{\cdot 1} - i}\phi^i$$

For more information, see Agresti (2013).

When the odds ratio is 0, which occurs when either $n_{11} = 0$ or $n_{22} = 0$, PROC FREQ sets the lower exact confidence limit to 0 and determines the upper limit by using the level α (instead of $\alpha/2$). Similarly, when the odds ratio is infinity, which occurs when either $n_{12} = 0$ or $n_{21} = 0$, PROC FREQ sets the upper exact confidence limit to infinity and determines the lower limit by using level α.

Relative Risks

Relative risks are useful measures in *cohort* (prospective) study designs, where two samples are identified based on the presence or absence of an explanatory factor. The two samples are observed in future time for the binary (yes-no) response variable under study. Relative risks are also useful in cross-sectional studies, where two variables are observed simultaneously. For more information, see Stokes, Davis, and Koch (2012) and Agresti (2007).

The relative risk is the ratio of the row 1 risk to the row 2 risk in a 2×2 table. The column 1 risk in row 1 is the proportion of row 1 observations that are classified in column 1, which can be expressed as

$$p_1 = n_{11} \ / \ n_{1\cdot}$$

Similarly, the column 1 risk in row 2 is

$$p_2 = n_{21} \ / \ n_{2\cdot}$$

The column 1 relative risk is defined as

$$R = p_1 \ / \ p_2$$

A relative risk greater than 1 indicates that the probability of positive response is greater in row 1 than in row 2. Similarly, a relative risk less than 1 indicates that the probability of positive response is less in row 1 than in row 2. The strength of association increases as the deviation from 1 increases.

Confidence Limits for the Relative Risk PROC FREQ provides the following types of confidence limits for the relative risk: exact unconditional, likelihood ratio, score, Wald, and Wald modified.

Wald Confidence Limits

The asymptotic Wald confidence limits are based on a log transformation of the relative risk. PROC FREQ computes the Wald confidence limits for the column 1 relative risk as

$$\left(\hat{r} \times \exp(-z\sqrt{v}), \ \ \hat{r} \times \exp(z\sqrt{v}) \right)$$

where $\hat{r}$ is the observed value of the relative risk, $\hat{p}_1/\hat{p}_2$, and

$$v = \mathrm{Var}(\log(\hat{r})) = 1/n_{11} + 1/n_{21} - 1/n_{1\cdot} - 1/n_{2\cdot}$$

and z is the $100(1-\alpha/2)$th percentile of the standard normal distribution. The confidence level α is determined by the ALPHA= option in the TABLES statement; by default, ALPHA=0.05, which produces 95% confidence limits. If either cell frequency n_{11} or n_{21} is 0, v is undefined and the Wald confidence limits cannot be computed.

PROC FREQ computes the confidence limits for the column 2 relative risk in the same way.

Wald Modified Confidence Limits

PROC FREQ computes Wald modified confidence limits (Haldane 1956) for the relative risk by replacing the n_{ij} with $(n_{ij} + 0.5)$ and the $n_{i\cdot}$ with $(n_{i\cdot} + 0.5)$ in the estimate and variance as follows:

$$\hat{r}_m = \frac{(n_{11} + 0.5)/(n_{1\cdot} + 0.5)}{(n_{21} + 0.5)/(n_{2\cdot} + 0.5)}$$

$$v = \mathrm{Var}(\log(\hat{r}_m)) = 1/(n_{11} + 0.5) + 1/(n_{21} + 0.5) - 1/(n_{1\cdot} + 0.5) - 1/(n_{2\cdot} + 0.5)$$

The confidence limits are computed as

$$\left(\hat{r}_m \times \exp(-z\sqrt{v}), \ \ \hat{r}_m \times \exp(z\sqrt{v}) \right)$$

where z is the $100(1-\alpha/2)$th percentile of the standard normal distribution. For more information, see Fleiss, Levin, and Paik (2003) and Agresti (2013).

Score Confidence Limits

Score confidence limits (Miettinen and Nurminen 1985; Farrington and Manning 1990) are computed by inverting score tests for the relative risk. A score-based chi-square test statistic for the null hypothesis that the relative risk is r_0 can be expressed as

$$Q(r_0) = (\hat{p}_1 - r_0\hat{p}_2)^2 \;/\; \widetilde{\mathrm{Var}}(r_0)$$

where $\hat{p}_1$ and $\hat{p}_2$ are the observed row 1 and row 2 risks (proportions), respectively,

$$\widetilde{\mathrm{Var}}(r_0) = (n/(n-1))\;\left(\;\tilde{p}_1(1-\tilde{p}_1)/n_{1\cdot} \;+\; {r_0}^2\;\tilde{p}_2(1-\tilde{p}_2)/n_{2\cdot}\;\right)$$

where $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of p_1 and p_2, respectively, under the null hypothesis that the relative risk is r_0. For more information, see Miettinen and Nurminen (1985) and Miettinen (1985, chapter 13).

The $100(1-\alpha)\%$ score confidence interval for the relative risk consists of all values of r_0 for which the test statistic $Q(r_0)$ falls in the acceptance region,

$$\{r_0 : Q(r_0) < \chi^2_{1,\alpha}\}$$

where $\chi^2_{1,\alpha}$ is the $100(1-\alpha)$th percentile of the chi-square distribution with 1 degree of freedom. For more information, see Agresti (2013).

By default, the score confidence limits include the bias correction factor $n/(n-1)$ in the denominator of $Q(r_0)$ (Miettinen and Nurminen 1985, p. 217). If you specify the CL=SCORE(CORRECT=NO) option, PROC FREQ does not include this factor in the computation.

The maximum likelihood estimates of p_1 and p_2, subject to the constraint that the relative risk is r_0, are computed as

$$\tilde{p}_1 = \left(-b - \sqrt{b^2 - 4ac}\right)/2a \quad \text{and} \quad \tilde{p}_2 = \tilde{p}_1/r_0$$

where

$$\begin{aligned} a &= 1+\theta \\ b &= -(r_0(1+\theta\hat{p}_2) + \theta + \hat{p}_1) \\ c &= r_0(\hat{p}_1 + \theta\hat{p}_2) \\ \theta &= n_{2\cdot}/n_{1\cdot} \end{aligned}$$

For more information, see Farrington and Manning (1990, p. 1454) and Miettinen and Nurminen (1985, p. 217).

Likelihood Ratio Confidence Limits

Likelihood ratio (profile likelihood) confidence limits for the relative risk are computed by inverting likelihood ratio tests. The likelihood ratio test statistic for the null hypothesis that the relative risk ratio is r_0 can be expressed as

$$G^2(r_0) \;=\; 2\left(n_{11}\log(\hat{p}_1/\tilde{p}_1) + n_{12}\log((1-\hat{p}_1)/(1-\tilde{p}_1)) + n_{21}\log(\hat{p}_2/\tilde{p}_2) + n_{22}\log((1-\hat{p}_2)/(1-\tilde{p}_2))\right)$$

where $\hat{p}_i$ is the observed row i risk $(n_{i1}/n_{i\cdot})$ and $\tilde{p}_i$ is the maximum likelihood estimate of the row i risk under the restriction that the relative risk is r_0. Expressions for the maximum likelihood estimates $\tilde{p}_1$ and $\tilde{p}_2$ are given in the subsection "Score Confidence Limits" in this section. For more information, see Miettinen and Nurminen (1985) and Miettinen (1985, chapter 13).

The $100(1-\alpha)\%$ likelihood ratio confidence interval for the relative risk consists of all values of r_0 for which the test statistic $G^2(r_0)$ falls in the acceptance region,

$$\{\theta : G^2(r_0) < \chi^2_{1,\alpha}\}$$

where $\chi^2_{1,\alpha}$ is the $100(1-\alpha)$th percentile of the chi-square distribution with 1 degree of freedom.

Exact Unconditional Confidence Limits

If you specify the RELRISK option in the EXACT statement, PROC FREQ provides exact unconditional confidence limits for the relative risk. The exact unconditional approach fixes the row margins of the 2×2 table and eliminates the nuisance parameter p_2 by using the maximum p-value (worst-case scenario) over all possible values of p_2 (Santner and Snell 1980). The conditional approach, which is described in the section "Exact Statistics" on page 3318, does not apply to the relative risk because of the nuisance parameter (Agresti 1992).

By default, PROC FREQ computes the confidence limits by the tail method, which inverts two separate one-sided exact tests of the relative risk, where the tests are based on the score statistic (Chan and Zhang 1999). The size of each one-sided exact test is at most $\alpha/2$, and the confidence coefficient is at least $(1-\alpha)$. If you specify the RELRISK(METHOD=NOSCORE) option in the EXACT statement, PROC FREQ computes the confidence limits by inverting two separate one-sided exact tests that are based on the unstandardized relative risk. If you specify the RELRISK(METHOD=SCORE2) option in the EXACT statement, PROC FREQ computes the confidence limits by inverting a single two-sided exact test that is based on the score statistic (Agresti and Min 2001).

PROC FREQ uses the relative risk score statistic (or the modified form of the unstandardized relative risk) to compute the exact confidence limits as described in the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274.

The score statistic is a less discrete statistic than the unstandardized risk difference and produces less conservative confidence limits (Agresti and Min 2001). For more information, see Santner et al. (2007). The relative risk score statistic (Miettinen and Nurminen 1985; Farrington and Manning 1990) is computed as

$$z(r_0) = (\hat{p}_1 - r_0\hat{p}_2) \;/\; \mathrm{se}(r_0)$$

where

$$\mathrm{se}(r_0) = \sqrt{\tilde{p}_1(1-\tilde{p}_1)/n_{1\cdot} \;+\; r_0{}^2\;\tilde{p}_2(1-\tilde{p}_2)/n_{2\cdot}}$$

where $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of p_1 and p_2 under the restriction that the relative risk is r_0. Expressions for the maximum likelihood estimates $\tilde{p}_1$ and $\tilde{p}_2$ are given in the subsection "Score Confidence Limits" in this section. For more information, see Farrington and Manning (1990, p. 1454) and Miettinen and Nurminen (1985, p. 217).

When the confidence limits are computed by using the unstandardized relative risk as the test statistic (METHOD=NOSCORE), PROC FREQ uses a modified form of the relative risk to ensure that the statistic

is defined when there are zero-frequency table cells. The modified form adds 0.5 to the table cell and row frequencies (Gart and Nam 1988) and is computed as

$$\hat{r}_m = \frac{(n_{11} + 0.5)/(n_{1\cdot} + 0.5)}{(n_{21} + 0.5)/(n_{2\cdot} + 0.5)}$$

For more information, see the subsection "Wald Modified Confidence Limits" in this section.

Relative Risk Tests PROC FREQ provides tests of equality, noninferiority, superiority, and equivalence for the relative risk. The following analysis methods are available: Wald (which is based on a log transformation), Wald modified, score, and likelihood ratio. You can specify the method by using the METHOD= *relrisk-option*; by default, PROC FREQ provides Wald tests.

Equality Test

An equality test for the relative risk can be expressed as

$$H_0\colon R = r_0$$

versus the alternative

$$H_a\colon R \neq r_0$$

where $R = p_1/p_2$ denotes the relative risk (for column 1 or column 2) and r_0 denotes the null value. You can specify a null value by using the EQUAL(NULL=) *relrisk-option*; by default, the null value is 1.

The test statistic is computed by the method that you specify; by default, PROC FREQ uses the Wald test. For information about test statistic computation, see the subsections "Wald Test," "Wald Modified Test," "Farrington-Manning (Score) Test," and "Likelihood Ratio Test" in this section.

For the Wald and score methods, the test statistics z have standard normal distributions under the null hypothesis. For the likelihood ratio test, the test statistic G^2 has a chi-square distribution with 1 degree of freedom under the null hypothesis.

When the test statistic z is greater than 0, PROC FREQ displays the right-sided p-value, which is the probability of a larger value occurring under the null hypothesis. The one-sided p-value can be expressed as

$$P_1 = \begin{cases} \text{Prob}(Z > z) & \text{if} \quad z > 0 \\ \text{Prob}(Z < z) & \text{if} \quad z \leq 0 \end{cases}$$

where Z has a standard normal distribution. The two-sided p-value is computed as $P_2 = 2 \times P_1$.

Noninferiority Test

A noninferiority test for the relative risk can be expressed as

$$H_0\colon R \leq \delta$$

versus the alternative

$$H_a\colon R > \delta$$

where $R = p_1/p_2$ denotes the relative risk (for column 1 or column 2) and δ denotes the noninferiority margin (limit). You can specify the margin by using the MARGIN= *relrisk-option*; by default, the noninferiority

margin is 0.8. The noninferiority margin for a relative risk test should be less than 1. Rejection of the null hypothesis indicates that the row 1 risk is not inferior to the row 2 risk. For more information, see Chow, Shao, and Wang (2008).

The test statistic z is computed by the method that you specify. For information about test statistic computation, see the subsections "Wald Test," "Wald Modified Test," "Farrington-Manning (Score) Test," and "Likelihood Ratio Test" in this section. The test statistic z is computed by using the noninferiority margin (limit) as the null value of the relative risk. Under the null hypothesis, the test statistic has a standard normal distribution. The p-value for the noninferiority test is the right-sided p-value (the probability that $Z > z$).

As part of the noninferiority analysis, PROC FREQ also provides confidence limits for the relative risk. The confidence coefficient is $100(1 - 2\alpha)\%$ (Schuirmann 1999). The confidence level α is determined by the ALPHA= option in the TABLES statement; by default, ALPHA=0.05, which produces 90% confidence limits for the noninferiority analysis. You can compare the confidence limits to the value of the noninferiority limit δ.

Superiority Test

A superiority test for the relative risk can be expressed as

$$H_0: R \leq \delta$$

versus the alternative

$$H_a: R > \delta$$

where $R = p_1/p_2$ denotes the relative risk (for column 1 or column 2) and δ denotes the superiority margin (limit). You can specify the margin by using the MARGIN= *relrisk-option*; by default, the superiority margin is 1.25. The superiority margin for a relative risk test should be greater than 1. Rejection of the null hypothesis indicates that the row 1 risk is superior to the row 2 risk. For more information, see Chow, Shao, and Wang (2008).

The test statistic z is computed by using the superiority margin (limit) as the null value of the relative risk. Under the null hypothesis, the test statistic has a standard normal distribution. The p-value for the superiority test is the right-sided p-value (the probability that $Z > z$).

The computations for the superiority analysis are the same as the computations for the noninferiority analysis, which are described in the subsection "Noninferiority Test" in this section.

Equivalence Test

An equivalence test for the relative risk can be expressed as

$$H_0: R \leq \delta_L \quad \text{or} \quad R \geq \delta_U$$

versus the alternative

$$H_a: \delta_L < R < \delta_U$$

where δ_L is the lower margin and δ_U is the upper margin. Rejection of the null hypothesis indicates that the two risks are equivalent. For more information, see Chow, Shao, and Wang (2008).

You can specify the margins by using the MARGIN= *relrisk-option*; by default, the lower margin is 0.8 and the upper margin is 1.25. If you specify a single margin value, PROC FREQ uses this value as the lower margin for the equivalence test and computes the upper margin as the inverse of the lower margin.

PROC FREQ computes two one-sided tests (TOST) for equivalence analysis (Schuirmann 1987), which include a right-sided test for the lower margin δ_L and a left-sided test for the upper margin δ_U. The lower test statistic uses the lower margin as the null relative risk value, and the p-value is the right-sided probability ($Z > z_L$). The upper test statistic uses the upper margin as the null value, and the p-value is the left-sided probability ($Z < z_U$). The overall p-value is taken to be the larger of the two p-values for the lower and upper tests.

The test statistics are computed by the method that you specify. For more information about the test statistic computation, see the subsections "Wald Test," "Wald Modified Test," "Farrington-Manning (Score) Test," and "Likelihood Ratio Test" in this section.

As part of the equivalence analysis, PROC FREQ also provides confidence limits for the relative risk. The confidence coefficient is $100(1-2\alpha)\%$ (Schuirmann 1999). The confidence level α is determined by the ALPHA= option in the TABLES statement; by default, ALPHA=0.05, which produces 90% confidence limits for the equivalence analysis. You can compare the confidence limits to the equivalence limits, which are δ_L and δ_U.

Wald Test

The Wald test statistic (which is based on a log transformation of the relative risk) is computed as $z(r_0) = (\log(\hat{r}) - \log(r_0))/\sqrt{v}$, where $\hat{r}$ is the relative risk estimate ($\hat{p}_1/\hat{p}_2$), r_0 is the null value of the relative risk, and

$$v = \mathrm{Var}(\log(\hat{r})) = 1/n_{11} + 1/n_{21} - 1/n_{1\cdot} - 1/n_{2\cdot}$$

The null value is determined by the type of test (equality, noninferiority, superiority, or equivalence) and the null or margin values that you specify. The side of the p-value and the interpretation of the test are also determined by the type of test; for more information, see the subsections "Equality Test," "Noninferiority Test," "Superiority Test," and "Equivalence Test" in this section.

Wald Modified Test

The Wald modified test statistic is computed by replacing the n_{ij} with $(n_{ij} + 0.5)$ and the $n_{i\cdot}$ with $(n_{i\cdot} + 0.5)$ in the relative risk estimate and variance. The test statistic is computed as $z(r_0) = (\log(\hat{r}_m) - \log(r_0))/\sqrt{v}$, where r_0 is the null value of the relative risk,

$$\hat{r}_m = \frac{(n_{11} + 0.5)/(n_{1\cdot} + 0.5)}{(n_{21} + 0.5)/(n_{2\cdot} + 0.5)}$$

$$v = \mathrm{Var}(\log(\hat{r}_m)) = 1/(n_{11} + 0.5) + 1/(n_{21} + 0.5) - 1/(n_{1\cdot} + 0.5) - 1/(n_{2\cdot} + 0.5)$$

The null value is determined by the type of test (equality, noninferiority, superiority, or equivalence) and the null or margin values that you specify. The side of the p-value and the interpretation of the test are also determined by the type of test; for more information, see the subsections "Equality Test," "Noninferiority Test," "Superiority Test," and "Equivalence Test" in this section.

Farrington-Manning (Score) Test

The relative risk score test statistic (Miettinen and Nurminen 1985; Farrington and Manning 1990) for the null value r_0 is computed as

$$z(r_0) = (\hat{p}_1 - r_0\hat{p}_2) \;/\; \mathrm{se}(r_0)$$

where

$$\mathrm{se}(r_0) = \sqrt{\tilde{p}_1(1-\tilde{p}_1)/n_{1\cdot} \; + \; r_0{}^2 \; \tilde{p}_2(1-\tilde{p}_2)/n_{2\cdot}}$$

where $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of p_1 and p_2 under the null value r_0. Expressions for the maximum likelihood estimates $\tilde{p}_1$ and $\tilde{p}_2$ are given in the subsection "Score Confidence Limits" in this section.

The null value is determined by the type of test (equality, noninferiority, superiority, or equivalence) and the null or margin values that you specify. The side of the p-value and the interpretation of the test are also determined by the type of test; for more information, see the subsections "Equality Test," "Noninferiority Test," "Superiority Test," and "Equivalence Test" in this section.

Likelihood Ratio Test

The likelihood ratio statistic for the null relative risk value r_0 is computed as

$$G^2(r_0) \; = \; 2\left(n_{11}\log(\hat{p}_1/\tilde{p}_1) + n_{12}\log((1-\hat{p}_1)/(1-\tilde{p}_1)) + n_{21}\log(\hat{p}_2/\tilde{p}_2) + n_{22}\log((1-\hat{p}_2)/(1-\tilde{p}_2))\right)$$

where $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of p_1 and p_2 under the null value r_0. Expressions for the maximum likelihood estimates $\tilde{p}_1$ and $\tilde{p}_2$ are given in the subsection "Score Confidence Limits" in this section. For more information, see Miettinen and Nurminen (1985) and Miettinen (1985, chapter 13).

PROC FREQ computes the likelihood ratio test statistic $z(r_0)$ for the noninferiority, superiority, and equivalence tests as $\sqrt{G^2(r_0)}$, where the sign is positive if the estimate is greater than the null value ($\hat{r} \geq r_0$) and negative otherwise ($\hat{r} < r_0$).

The null value is determined by the type of test (equality, noninferiority, superiority, or equivalence) and the null or margin values that you specify. The side of the p-value and the interpretation of the test are also determined by the type of test; for more information, see the subsections "Equality Test," "Noninferiority Test," "Superiority Test," and "Equivalence Test" in this section.

Cochran-Armitage Test for Trend

The TREND option in the TABLES statement provides the Cochran-Armitage test for trend, which tests for trend in binomial proportions across levels of a single factor or covariate. This test is appropriate for a two-way table where one variable has two levels and the other variable is ordinal. The two-level variable represents the response, and the other variable represents an explanatory variable with ordered levels. When the two-way has two columns and R rows, PROC FREQ tests for trend across the R levels of the row variable, and the binomial proportion is computed as the proportion of observations in the first column. When the table has two rows and C columns, PROC FREQ tests for trend across the C levels of the column variable, and the binomial proportion is computed as the proportion of observations in the first row.

The trend test is based on the regression coefficient for the weighted linear regression of the binomial proportions on the scores of the explanatory variable levels. For more information, see Margolin (1988) and Agresti (2002). If the table has two columns and R rows, the trend test statistic is computed as

$$T = \sum_{i=1}^{R} n_{i1}(R_i - \bar{R}) \; / \; \sqrt{p_{\cdot 1}\,(1-p_{\cdot 1})\,s^2}$$

where R_i is the score of row i, $\bar{R}$ is the average row score, and

$$s^2 = \sum_{i=1}^{R} n_{i\cdot}(R_i - \bar{R})^2$$

The SCORES= option in the TABLES statement determines the type of row scores used in computing the trend test (and other score-based statistics). By default, SCORES=TABLE. For more information, see the section "Scores" on page 3247. For character variables, the table scores for the row variable are the row numbers (for example, 1 for the first row, 2 for the second row, and so on). For numeric variables, the table score for each row is the numeric value of the row level. When you perform the trend test, the explanatory variable might be numeric (for example, dose of a test substance), and the variable values might be appropriate scores. If the explanatory variable has ordinal levels that are not numeric, you can assign meaningful scores to the variable levels. Sometimes equidistant scores, such as the table scores for a character variable, might be appropriate. For more information on choosing scores for the trend test, see Margolin (1988).

The null hypothesis for the Cochran-Armitage test is no trend, which means that the binomial proportion $p_{i1} = n_{i1}/n_{i\cdot}$ is the same for all levels of the explanatory variable. Under the null hypothesis, the trend statistic has an asymptotic standard normal distribution.

PROC FREQ computes one-sided and two-sided p-values for the trend test. When the test statistic is greater than its null hypothesis expected value of 0, PROC FREQ displays the right-sided p-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. A small right-sided p-value supports the alternative hypothesis of increasing trend in proportions from row 1 to row R. When the test statistic is less than or equal to 0, PROC FREQ displays the left-sided p-value. A small left-sided p-value supports the alternative of decreasing trend.

The one-sided p-value for the trend test is computed as

$$P_1 = \begin{cases} \mathrm{Prob}(Z > T) & \text{if} \quad T > 0 \\ \mathrm{Prob}(Z < T) & \text{if} \quad T \leq 0 \end{cases}$$

where Z has a standard normal distribution. The two-sided p-value is computed as

$$P_2 = \mathrm{Prob}(|Z| > |T|)$$

PROC FREQ also provides exact p-values for the Cochran-Armitage trend test. You can request the exact test by specifying the TREND option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Jonckheere-Terpstra Test

The JT option in the TABLES statement provides the Jonckheere-Terpstra test, which is a nonparametric test for ordered differences among classes. It tests the null hypothesis that the distribution of the response variable does not differ among classes. It is designed to detect alternatives of ordered class differences, which can be expressed as $\tau_1 \leq \tau_2 \leq \cdots \leq \tau_R$ (or $\tau_1 \geq \tau_2 \geq \cdots \geq \tau_R$), with at least one of the inequalities being strict, where τ_i denotes the effect of class i. For such ordered alternatives, the Jonckheere-Terpstra test can be preferable to tests of more general class difference alternatives, such as the Kruskal–Wallis test (produced by the WILCOXON option in the NPAR1WAY procedure). See Pirie (1983) and Hollander and Wolfe (1999) for more information about the Jonckheere-Terpstra test.

The Jonckheere-Terpstra test is appropriate for a two-way table in which an ordinal column variable represents the response. The row variable, which can be nominal or ordinal, represents the classification variable. The levels of the row variable should be ordered according to the ordering you want the test to detect. The order of variable levels is determined by the ORDER= option in the PROC FREQ statement. By default, ORDER=INTERNAL, which orders by unformatted values. If you specify ORDER=DATA, PROC FREQ

orders values according to their order in the input data set. For more information about how to order variable levels, see the ORDER= option.

The Jonckheere-Terpstra test statistic is computed by first forming $R(R-1)/2$ Mann-Whitney counts $M_{i,i'}$, where $i < i'$, for pairs of rows in the contingency table,

$$\begin{aligned} M_{i,i'} &= \{ \text{ number of times } X_{i,j} < X_{i',j'}, \quad j = 1,\ldots,n_{i.}; \quad j' = 1,\ldots,n_{i'.} \} \\ &+ \tfrac{1}{2} \{ \text{ number of times } X_{i,j} = X_{i',j'}, \quad j = 1,\ldots,n_{i.}; \quad j' = 1,\ldots,n_{i'.} \} \end{aligned}$$

where $X_{i,j}$ is response j in row i. The Jonckheere-Terpstra test statistic is computed as

$$J = \sum_{1 \le i < i' \le R} \sum M_{i,i'}$$

This test rejects the null hypothesis of no difference among classes for large values of J. Asymptotic p-values for the Jonckheere-Terpstra test are obtained by using the normal approximation for the distribution of the standardized test statistic. The standardized test statistic is computed as

$$J^* = (J - \mathrm{E}_0(J)) \ / \ \sqrt{\mathrm{Var}_0(J)}$$

where $\mathrm{E}_0(J)$ and $\mathrm{Var}_0(J)$ are the expected value and variance of the test statistic under the null hypothesis,

$$\mathrm{E}_0(J) = \left(n^2 - \sum_i n_{i.}^2 \right) / 4$$

$$\mathrm{Var}_0(J) = A/72 + B/\left(36n(n-1)(n-2)\right) + C/\left(8n(n-1)\right)$$

where

$$A = n(n-1)(2n+5) - \sum_i n_{i.}(n_{i.}-1)(2n_{i.}+5) - \sum_j n_{.j}(n_{.j}-1)(2n_{.j}+5)$$

$$B = \left(\sum_i n_{i.}(n_{i.}-1)(n_{i.}-2) \right) \left(\sum_j n_{.j}(n_{.j}-1)(n_{.j}-2) \right)$$

$$C = \left(\sum_i n_{i.}(n_{i.}-1) \right) \left(\sum_j n_{.j}(n_{.j}-1) \right)$$

PROC FREQ computes one-sided and two-sided p-values for the Jonckheere-Terpstra test. When the standardized test statistic is greater than its null hypothesis expected value of 0, PROC FREQ displays the right-sided p-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. A small right-sided p-value supports the alternative hypothesis of increasing order from row 1 to row R. When the standardized test statistic is less than or equal to 0, PROC FREQ displays the left-sided p-value. A small left-sided p-value supports the alternative of decreasing order from row 1 to row R.

The one-sided p-value for the Jonckheere-Terpstra test, P_1, is computed as

$$P_1 = \begin{cases} \mathrm{Prob}(Z > J^*) & \text{if} \quad J^* > 0 \\ \mathrm{Prob}(Z < J^*) & \text{if} \quad J^* \le 0 \end{cases}$$

where Z has a standard normal distribution. The two-sided p-value, P_2, is computed as

$$P_2 = \text{Prob}(|Z| > |J^*|)$$

PROC FREQ also provides exact p-values for the Jonckheere-Terpstra test. You can request the exact test by specifying the JT option in the EXACT statement. See the section "Exact Statistics" on page 3318 for more information.

Tests and Measures of Agreement

When you specify the AGREE option in the TABLES statement, PROC FREQ computes tests and measures of agreement for square tables (for which the number of rows equals the number of columns). By default, these statistics include McNemar's test for 2×2 tables, Bowker's symmetry test, the simple kappa coefficient, and the weighted kappa coefficient. For multiple strata (n-way tables, where $n > 2$), the AGREE option provides the overall simple and weighted kappa coefficients, in addition to tests for equal kappas (simple and weighted) among strata. For $2 \times 2 \times \cdots \times 2$ tables, the AGREE option provides Cochran's Q test.

Optionally, PROC FREQ provides kappa tests and other agreement statistics. In addition to the asymptotic tests described in this section, PROC FREQ provides exact p-values for McNemar's test, the simple kappa coefficient test, and the weighted kappa coefficient test. You can request these exact tests by specifying the corresponding options in the EXACT statement. For more information, see the section "Exact Statistics" on page 3318.

The following sections provide the formulas that PROC FREQ uses to compute agreement statistics. For information about the use and interpretation of these statistics, see Agresti (2002, 2007); Fleiss, Levin, and Paik (2003), and the other references cited for each statistic.

McNemar's Test

PROC FREQ computes McNemar's test (McNemar 1947) for 2×2 tables when you specify the AGREE option. This test is appropriate when you are analyzing data from matched pairs of subjects with a dichotomous (yes-no) response. By default, the null hypothesis for McNemar's test is marginal homogeneity, which can be expressed as $p_{1\cdot} = p_{\cdot 1}$; this is equivalent to a discordant proportion ratio (p_{12}/p_{21}) of 1. The corresponding test statistic is computed as

$$Q_M = (n_{12} - n_{21})^2 \;/\; (n_{12} + n_{21})$$

Under the null hypothesis, Q_M has an asymptotic chi-square distribution with 1 degree of freedom.

Optionally, you can specify the null ratio of discordant proportions (p_{12}/p_{21}) by using the AGREE(MNULLRATIO=) option. When the null ratio is r, McNemar's test is computed as

$$Q_M(r) = (n_{12} - e_{12})^2)/e_{12} \;+\; (n_{21} - e_{21})^2/e_{21}$$

where $e_{12} = D/(1 + 1/r)$, $e_{21} = D/(1 + r)$, and D is the number of discordant pairs, $(n_{12} + n_{21})$. Under the null hypothesis, $Q_M(r)$ has an asymptotic chi-square distribution with 1 degree of freedom.

PROC FREQ also computes an exact p-value for McNemar's test when you specify the MCNEM option in the EXACT statement.

Bowker's Symmetry Test

The null hypothesis for Bowker's symmetry test (Bowker 1948) is symmetric table-cell proportions, which can be expressed as $p_{ij} = p_{ji}$ for all off-diagonal pairs of table cells. For 2×2 tables, Bowker's test is identical to McNemar's test; therefore, PROC FREQ provides Bowker's test only for square tables that are larger than 2×2.

Bowker's symmetry test is computed as

$$Q_B = \sum_{i<j}\sum (n_{ij} - n_{ji})^2 \;/\; (n_{ij} + n_{ji})$$

For large samples, Q_B has an asymptotic chi-square distribution with $R(R-1)/2$ degrees of freedom under the null hypothesis of symmetry, where R is the dimension of the square, two-way table.

By default, the number of degrees of freedom for this test ($R(R-1)/2$) is the number of off-diagonal table-cell comparisons. You can specify the number of degrees of freedom in the AGREE(DFSYM=) option. Alternatively, you can specify the AGREE(DFSYM=ADJUST) option, which reduces the degrees of freedom by the number of off-diagonal table-cell pairs that have a total frequency of 0. For more information, see Hoenig, Morgan, and Brown (1995).

Exact Symmetry Test When you specify the SYMMETRY option in the EXACT statement, PROC FREQ provides an exact symmetry test by using the method of Krauth (1973). This exact test is computed by conditioning on the observed frequency sums of the complementary off-diagonal table-cell pairs ($n_{ij} + n_{ji}$). PROC FREQ evaluates the symmetry test statistic for all tables in the reference set, which includes all possible tables in which the frequency sums of the off-diagonal table-cell pairs match the corresponding frequency sums in the observed table. The exact p-value is then computed as the sum of the table probabilities for those tables for which the symmetry test statistic is greater than or equal to the observed test statistic. The table probabilities are computed as products of $R(R-1)/2$ binomial probabilities (which correspond to the off-diagonal table-cell pairs in tables of dimension R) by using the binomial proportion 0.5 under the null hypothesis of symmetry. For more information, see the section "Exact Statistics" on page 3318.

Alternatively, you can request a Monte Carlo estimate of the exact p-value by specifying the SYMMETRY option together with the MC *computation-option* in the EXACT statement. The Monte Carlo computation for the exact symmetry test is conditional on the same reference set that the exact test uses (tables in which the frequency sums of the off-diagonal table-cell pairs match the corresponding sums in the observed table). For more information, see the section "Monte Carlo Estimation" on page 3321.

Simple Kappa Coefficient

The simple kappa coefficient (Cohen 1960) is a measure of interrater agreement. PROC FREQ computes the simple kappa coefficient as

$$\hat{\kappa} = (P_o - P_e) \;/\; (1 - P_e)$$

where $P_o = \sum_i p_{ii}$ and $P_e = \sum_i p_{i.}p_{.i}$. The component P_o is the proportion of observed agreement, and the component P_e represents the proportion of chance-expected agreement.

If the two response variables are viewed as two independent ratings of the n subjects, the kappa coefficient is +1 when there is complete agreement of the raters. When the observed agreement exceeds the chance-expected agreement, the kappa coefficient is positive, and its magnitude reflects the strength of agreement. When the observed agreement is less than the chance-expected agreement, the kappa coefficient is negative. The minimum value of kappa is between –1 and 0, depending on the marginal proportions of the table.

PROC FREQ computes the asymptotic variance of the simple kappa coefficient as

$$\mathrm{Var}(\hat{\kappa}) = (A + B - C) \ / \ (1 - P_e)^2 \, n$$

where

$$\begin{aligned} A &= \sum_i p_{ii} \left(1 - (p_{i\cdot} + p_{\cdot i})(1 - \hat{\kappa})\right)^2 \\ B &= (1 - \hat{\kappa})^2 \sum\sum_{i \neq j} p_{ij} (p_{\cdot i} + p_{j\cdot})^2 \\ C &= (\, \hat{\kappa} - P_e(1 - \hat{\kappa}) \,)^2 \end{aligned}$$

For more information, see Fleiss, Cohen, and Everitt (1969).

Confidence limits for the simple kappa coefficient are computed as

$$\hat{\kappa} \ \pm \ \left(z_{\alpha/2} \times \sqrt{\mathrm{Var}(\hat{\kappa})} \right)$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. The value of α is determined by the ALPHA= option; by default ALPHA=0.05, which produces 95% confidence limits.

PROC FREQ provides an asymptotic test for the simple kappa coefficient. By default, the null hypothesis value of kappa is 0; alternatively, you can specify a nonzero null value of kappa (by using the AGREE(NULLKAPPA=) option in the TABLES statement). When the null value of kappa is nonzero, PROC FREQ computes the test statistic as

$$z = (\hat{\kappa} - \kappa_0) \ / \ \sqrt{\mathrm{Var}(\hat{\kappa})}$$

where κ_0 is the null value that you specify and $\mathrm{Var}(\hat{\kappa})$ is the variance of the kappa coefficient.

When the null value of kappa is 0, PROC FREQ computes the test statistic as

$$z = \hat{\kappa} \ / \ \sqrt{\mathrm{Var}_0(\hat{\kappa})}$$

where $\mathrm{Var}_0(\hat{\kappa})$ is the variance of the kappa coefficient under the null hypothesis (that kappa is 0) and is computed as

$$\mathrm{Var}_0(\hat{\kappa}) = \left(P_e + P_e^2 - \sum_i p_{i\cdot} p_{\cdot i} (p_{i\cdot} + p_{\cdot i}) \right) \ / \ (1 - P_e)^2 \, n$$

This test statistic has an asymptotic standard normal distribution under the null hypothesis. For more information, see Fleiss, Levin, and Paik (2003).

PROC FREQ also provides an exact test for the simple kappa coefficient. You can request the exact test by specifying the KAPPA or AGREE option in the EXACT statement. For more information, see the section "Exact Statistics" on page 3318.

Kappa Details When you specify the AGREE(KAPPADETAILS) option, PROC FREQ displays the "Kappa Details" table, which includes the observed agreement P_o, chance-expected agreement P_e, maximum kappa, and B_n measure.

The maximum kappa, which is the maximum possible value of the kappa coefficient given the marginal proportions of the two-way table, is computed as

$$\max(\kappa) = (\max(P_o) - P_e) \ / \ (1 - P_e)$$

where

$$\max(P_o) = \left(\sum_i \min(n_{i\cdot}, \ n_{\cdot i}) \right) \ / \ n$$

The B_n measure (Bangdiwala 1988; Bangdiwala et al. 2008) is computed as

$$B_n = \left(\sum_i n_{ii}^2 \right) / \left(\sum_i \sum_j n_{i\cdot} n_{\cdot i} \right)$$

For 2×2 tables, the "Kappa Details" table also includes the prevalence index and the bias index. The prevalence index is the absolute difference between the agreement proportions, $|p_{11} - p_{22}|$. The bias index is the absolute difference between the disagreement proportions, $|p_{12} - p_{21}|$. For more information, see Sim and Wright (2005) and Byrt, Bishop, and Carlin (1993).

Weighted Kappa Coefficient

The weighted kappa coefficient is a generalization of the simple kappa coefficient that uses weights to quantify the relative differences between categories. For 2×2 tables, the weighted kappa coefficient is equivalent to the simple kappa coefficient; therefore, PROC FREQ displays the weighted kappa coefficient only for tables larger than 2×2. PROC FREQ computes the kappa weights from the column scores, by using either Cicchetti-Allison weights or Fleiss-Cohen weights, both of which are described in the section "Kappa Weights" on page 3305. The kappa weights w_{ij} are constructed so that $0 \leq w_{ij} < 1$ for all $i \neq j$, $w_{ii} = 1$ for all i, and $w_{ij} = w_{ji}$. The weighted kappa coefficient is computed as

$$\hat{\kappa}_w = \left(P_{o(w)} - P_{e(w)}\right) \ / \ \left(1 - P_{e(w)}\right)$$

where

$$P_{o(w)} = \sum_i \sum_j w_{ij} \, p_{ij}$$

$$P_{e(w)} = \sum_i \sum_j w_{ij} \, p_{i\cdot} p_{\cdot j}$$

The component $P_{o(w)}$ is the proportion of observed (weighted) agreement, and the component $P_{e(w)}$ represents the proportion of chance-expected (weighted) agreement. When you specify the AGREE(WTKAPDETAILS) option, PROC FREQ displays these components in the "Weighted Kappa Details" table.

PROC FREQ computes the asymptotic variance of the weighted kappa coefficient as

$$\mathrm{Var}(\hat{\kappa}_w) = \left(\sum_i \sum_j p_{ij} \left(w_{ij} - (\overline{w}_{i\cdot} + \overline{w}_{\cdot j})(1 - \hat{\kappa}_w) \right)^2 - \left(\hat{\kappa}_w - P_{e(w)}(1 - \hat{\kappa}_w) \right)^2 \right) \Big/ (1 - P_{e(w)})^2 \, n$$

where

$$\overline{w}_{i\cdot} = \sum_j p_{\cdot j} w_{ij}$$

$$\overline{w}_{\cdot j} = \sum_i p_{i\cdot} w_{ij}$$

For more information, see Fleiss, Cohen, and Everitt (1969).

Confidence limits for the weighted kappa coefficient are computed as

$$\hat{\kappa}_w \ \pm \ \left(z_{\alpha/2} \times \sqrt{\mathrm{Var}(\hat{\kappa}_w)} \right)$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. The value of α is determined by the ALPHA= option; by default ALPHA=0.05, which produces 95% confidence limits.

PROC FREQ provides an asymptotic test for the weighted kappa coefficient. By default, the null hypothesis value of weighted kappa is 0; alternatively, you can specify a nonzero null value of weighted kappa (by using the AGREE(NULLWTKAPPA=) option in the TABLES statement). When the null value of weighted kappa is nonzero, PROC FREQ computes the test statistic as

$$z = (\hat{\kappa}_w - \kappa_{w(0)}) \ / \ \sqrt{\mathrm{Var}(\hat{\kappa}_w)}$$

where $\kappa_{w(0)}$ is the null value that you specify and $\mathrm{Var}(\hat{\kappa}_w)$ is the variance of the weighted kappa coefficient.

When the null value of weighted kappa is 0, PROC FREQ computes the test statistic as

$$z = \hat{\kappa}_w \ / \ \sqrt{\mathrm{Var}_0(\hat{\kappa}_w)}$$

where $\mathrm{Var}_0(\hat{\kappa}_w)$ is the variance of the weighted kappa coefficient under the null hypothesis (that weighted kappa is 0) and is computed as

$$\mathrm{Var}_0(\hat{\kappa}_w) = \left(\sum_i \sum_j p_{i\cdot} p_{\cdot j} \left(w_{ij} - (\overline{w}_{i\cdot} + \overline{w}_{\cdot j}) \right)^2 - P_{e(w)}^2 \right) \Big/ (1 - P_{e(w)})^2 \, n$$

This test statistic has an asymptotic standard normal distribution under the null hypothesis. For more information, see Fleiss, Levin, and Paik (2003).

PROC FREQ also provides an exact test for the weighted kappa coefficient. You can request the exact test by specifying the KAPPA or AGREE option in the EXACT statement. For more information, see the section "Exact Statistics" on page 3318.

Kappa Weights PROC FREQ computes kappa coefficient weights by using the column scores and one of the two available weight types. The column scores are determined by the SCORES= option in the TABLES statement. The two available types of kappa weights are Cicchetti-Allison and Fleiss-Cohen weights. By default, PROC FREQ uses Cicchetti-Allison weights. If you specify the AGREE(WT=FC) option, PROC FREQ uses Fleiss-Cohen weights to compute the weighted kappa coefficient.

PROC FREQ computes Cicchetti-Allison kappa coefficient weights as

$$w_{ij} = 1 - \frac{|C_i - C_j|}{C_C - C_1}$$

where C_i is the score for column i and C is the number of categories or columns. For more information, see Cicchetti and Allison (1971).

The SCORES= option in the TABLES statement determines the type of column scores used to compute the kappa weights (and other score-based statistics). By default, SCORES=TABLE. For more information, see the section "Scores" on page 3247. For numeric variables, table scores are the values of the variable levels. You can assign numeric values to the levels in a way that reflects their level of similarity. For example, suppose you have four levels and order them according to similarity. If you assign them values of 0, 2, 4, and 10, the Cicchetti-Allison kappa weights take the following values: $w_{12} = 0.8$, $w_{13} = 0.6$, $w_{14} = 0$, $w_{23} = 0.8$, $w_{24} = 0.2$, and $w_{34} = 0.4$. Note that when there are only two categories (that is, $C = 2$), the weighted kappa coefficient is identical to the simple kappa coefficient.

If you specify the AGREE(WT=FC) option in the TABLES statement, PROC FREQ computes Fleiss-Cohen kappa coefficient weights as

$$w_{ij} = 1 - \frac{(C_i - C_j)^2}{(C_C - C_1)^2}$$

For more information, see Fleiss and Cohen (1973).

For the preceding example, the Fleiss-Cohen kappa weights are $w_{12} = 0.96$, $w_{13} = 0.84$, $w_{14} = 0$, $w_{23} = 0.96$, $w_{24} = 0.36$, and $w_{34} = 0.64$.

Prevalence-Adjusted Bias-Adjusted Kappa

When you specify the AGREE(PABAK) option, PROC FREQ provides the prevalence-adjusted bias-adjusted kappa coefficient (PABAK) (Byrt, Bishop, and Carlin 1993). This coefficient is computed as

$$\hat{\kappa}_a = (P_o - 1/R) \ / \ (1 - 1/R)$$

where $P_o = \sum_i p_{ii}$ and R is the dimension of the square, two-way table. The component P_o is the proportion of observed agreement, and the component $1/R$ represents the chance-expected agreement. When the table is 2×2, $\hat{\kappa}_a = 2P_o - 1$. For more information, see Sim and Wright (2005), Xie (2013), and Holley and Guilford (1964).

PROC FREQ computes the variance of the prevalence-adjusted bias-adjusted kappa as

$$\mathrm{Var}(\hat{\kappa}_a) = (R/(R-1))^2 \, (P_o(1 - P_o)/n)$$

Confidence limits are computed as

$$\hat{\kappa}_a \ \pm \ \left(z_{\alpha/2} \times \sqrt{\mathrm{Var}(\hat{\kappa}_a)} \right)$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. The value of α is determined by the ALPHA= option; by default ALPHA=0.05, which produces 95% confidence limits.

AC1 Agreement Coefficient

When you specify the AGREE(AC1) option, PROC FREQ provides Gwet's first-order agreement coefficient, AC1 (Gwet 2008). This coefficient is computed as

$$\hat{\gamma} = \left(P_o - P_{e(\gamma)}\right) \ / \ \left(1 - P_{e(\gamma)}\right)$$

where $P_o = \sum_i p_{ii}$, $P_e = \sum_i e_i(1-e_i)/(R-1)$, and $e_i = (p_{i\cdot} + p_{\cdot i})/2$ The component P_o is the proportion of observed agreement, and the component $P_{e(\gamma)}$ represents the proportion of chance-expected agreement. For more information, see Xie (2013) and Blood and Spratt (2007).

PROC FREQ computes the variance of AC1 as

$$\mathrm{Var}(\hat{\gamma}) = \left(P_o(1-P_o) - 4(1-\hat{\gamma})A + 4(1-\hat{\gamma}^2)B\right) \ / \ n(1-P_{e(\gamma)})^2$$

where

$$A = \sum_i p_{ii}(1-e_i)/(R-1) - P_o P_{e(\gamma)}$$

$$B = \sum_i \sum_j p_{ij} \left(1 - (e_i + e_j)/2\right)^2 / (R-1)^2 - P_{e(\gamma)}^2$$

Confidence limits for AC1 are computed as

$$\hat{\gamma} \ \pm \ \left(z_{\alpha/2} \times \sqrt{\mathrm{Var}(\hat{\gamma})} \right)$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. The value of α is determined by the ALPHA= option; by default ALPHA=0.05, which produces 95% confidence limits.

Overall Kappa Coefficient

When there are multiple strata, PROC FREQ combines the stratum-level estimates of kappa into an overall estimate of the supposed common value of kappa. Assume there are q strata, indexed by $h = 1, 2, \ldots, q$, and let $\mathrm{Var}(\hat{\kappa}_h)$ denote the variance of $\hat{\kappa}_h$. The estimate of the overall kappa coefficient is computed as

$$\hat{\kappa}_T = \sum_{h=1}^{q} \frac{\hat{\kappa}_h}{\mathrm{Var}(\hat{\kappa}_h)} \ / \ \sum_{h=1}^{q} \frac{1}{\mathrm{Var}(\hat{\kappa}_h)}$$

For more information, see Fleiss, Levin, and Paik (2003).

PROC FREQ computes an estimate of the overall weighted kappa in the same way.

Tests for Equal Kappa Coefficients

When there are multiple strata, the following chi-square statistic tests whether the stratum-level values of kappa are equal:

$$Q_K = \sum_{h=1}^{q} (\hat{\kappa}_h - \hat{\kappa}_T)^2 \ / \ \mathrm{Var}(\hat{\kappa}_h)$$

Under the null hypothesis of equal kappas for the q strata, Q_K has an asymptotic chi-square distribution with q–1 degrees of freedom. See Fleiss, Levin, and Paik (2003) for more information. PROC FREQ computes a test for equal weighted kappa coefficients in the same way.

Cochran's Q Test

Cochran's Q is computed for multiway tables in which each variable has two levels—that is, for $2 \times 2 \times \cdots \times 2$ tables. Cochran's Q statistic is used to test the homogeneity of the one-dimensional margins. Let m denote the number of variables and N denote the total number of subjects. Cochran's Q statistic is computed as

$$Q_C = m(m-1)\left(\sum_{j=1}^{m} T_j^2 - T^2/m\right) \Big/ \left(mT - \sum_{k=1}^{N} S_k^2\right)$$

where T_j is the number of positive responses for variable j, T is the total number of positive responses over all variables, and S_k is the number of positive responses for subject k. Under the null hypothesis, Cochran's Q has an asymptotic chi-square distribution with m–1 degrees of freedom. For more information, see Cochran (1950). When there are only two binary response variables (m=2), Cochran's Q simplifies to McNemar's test. When there are more than two response categories, you can test for marginal homogeneity by using the repeated measures capabilities of the CATMOD procedure.

Tables with Zero-Weight Rows or Columns

The AGREE statistics are defined only for square tables, where the number of rows equals the number of columns; if a table is not square, PROC FREQ does not compute AGREE statistics for the table. In the kappa statistic framework, where two independent raters assign ratings to each of n subjects, suppose one of the raters does not use all possible r rating levels. If the corresponding table contains r rows but only r–1 columns, the table is not square and PROC FREQ does not compute AGREE statistics. To create a square table in this situation, you can use the ZEROS option in the WEIGHT statement, which includes zero-weight observations in the analysis. You can include zero-weight observations in the input data set to represent any rating levels that are not used by a rater, so that the input data set has at least one observation for each possible rater and rating combination. When you use this input data set and specify the ZEROS option, the analysis includes all rating levels (even when all levels are not actually assigned by both raters). The resulting table (of rater 1 by rater 2) is a square table, and AGREE statistics can be computed.

For more information, see the description of the ZEROS option in the WEIGHT statement. By default, PROC FREQ does not process observations that have weights of 0 because these observations do not contribute to the total frequency count, and because many of the tests and measures of association are undefined for tables that contain zero-weight rows or columns. However, kappa statistics are defined for tables that contain zero-weight rows or columns, and the ZEROS option enables you to input zero-weight observations and construct the tables needed to compute kappa statistics.

Cochran-Mantel-Haenszel Statistics

The CMH option in the TABLES statement gives a stratified statistical analysis of the relationship between the row and column variables after controlling for the strata variables in a multiway table. For example, for the table request A*B*C*D, the CMH option provides an analysis of the relationship between C and D, after controlling for A and B. The stratified analysis provides a way to adjust for the possible confounding effects of A and B without being forced to estimate parameters for them.

The CMH analysis produces Cochran-Mantel-Haenszel statistics, which include the correlation statistic, the ANOVA (row mean scores) statistic, and the general association statistic. For 2×2 tables, the CMH option also provides Mantel-Haenszel and logit estimates of the common odds ratio and the common relative risks, in addition to the Breslow-Day test for homogeneity of the odds ratios.

Exact statistics are also available for stratified 2×2 tables. If you specify the EQOR option in the EXACT statement, PROC FREQ provides Zelen's exact test for equal odds ratios. If you specify the COMOR option in the EXACT statement, PROC FREQ provides exact confidence limits for the common odds ratio and an exact test that the common odds ratio equals one.

Let the number of strata be denoted by q, indexing the strata by $h = 1, 2, \ldots, q$. Each stratum contains a contingency table with X representing the row variable and Y representing the column variable. For table h, denote the cell frequency in row i and column j by n_{hij}, with corresponding row and column marginal totals denoted by $n_{hi.}$ and $n_{h.j}$, and the overall stratum total by n_h.

Because the formulas for the Cochran-Mantel-Haenszel statistics are more easily defined in terms of matrices, the following notation is used. Vectors are presumed to be column vectors unless they are transposed ($'$).

$$
\begin{aligned}
\mathbf{n}'_{hi} &= (n_{hi1}, n_{hi2}, \ldots, n_{hiC}) && (1 \times C) \\
\mathbf{n}'_{h} &= (\mathbf{n}'_{h1}, \mathbf{n}'_{h2}, \ldots, \mathbf{n}'_{hR}) && (1 \times RC) \\
p_{hi\cdot} &= n_{hi\cdot} \,/\, n_h && (1 \times 1) \\
p_{h\cdot j} &= n_{h\cdot j} \,/\, n_h && (1 \times 1) \\
\mathbf{P}'_{h*\cdot} &= (p_{h1\cdot}, p_{h2\cdot}, \ldots, p_{hR\cdot}) && (1 \times R) \\
\mathbf{P}'_{h\cdot *} &= (p_{h\cdot 1}, p_{h\cdot 2}, \ldots, p_{h\cdot C}) && (1 \times C)
\end{aligned}
$$

Assume that the strata are independent and that the marginal totals of each stratum are fixed. The null hypothesis, H_0, is that there is no association between X and Y in any of the strata. The corresponding model is the multiple hypergeometric; this implies that, under H_0, the expected value and covariance matrix of the frequencies are, respectively,

$$\mathbf{m}_h = \mathbf{E}[\mathbf{n}_h \mid H_0] = n_h(\mathbf{P}_{h\cdot *} \otimes \mathbf{P}_{h*\cdot})$$

$$\mathbf{Var}[\mathbf{n}_h \mid H_0] = c\,\big(\,(\mathbf{D}_{\mathbf{P}h\cdot *} - \mathbf{P}_{h\cdot *}\mathbf{P}'_{h\cdot *}) \otimes (\mathbf{D}_{\mathbf{P}h*\cdot} - \mathbf{P}_{h*\cdot}\mathbf{P}'_{h*\cdot})\,\big)$$

where

$$c = n_h^2 \,/\, (n_h - 1)$$

and where $\otimes$ denotes Kronecker product multiplication and $\mathbf{D_a}$ is a diagonal matrix with the elements of $\mathbf{a}$ on the main diagonal.

The generalized CMH statistic (Landis, Heyman, and Koch 1978) is defined as

$$Q_{\mathrm{CMH}} = \mathbf{G}'\mathbf{V_G}^{-1}\mathbf{G}$$

where

$$
\begin{aligned}
\mathbf{G} &= \sum_h \mathbf{B}_h(\mathbf{n}_h - \mathbf{m}_h) \\
\mathbf{V_G} &= \sum_h \mathbf{B}_h\,(\mathbf{Var}[\mathbf{n}_h \mid H_0])\,\mathbf{B}'_h
\end{aligned}
$$

and where

$$\mathbf{B}_h = \mathbf{C}_h \otimes \mathbf{R}_h$$

is a matrix of fixed constants based on column scores $\mathbf{C}_h$ and row scores $\mathbf{R}_h$. When the null hypothesis is true, the CMH statistic has an asymptotic chi-square distribution with degrees of freedom equal to the rank of $\mathbf{B}_h$. If $\mathbf{V_G}$ is found to be singular, PROC FREQ prints a message and sets the value of the CMH statistic to missing.

PROC FREQ computes three CMH statistics by using this formula for the generalized CMH statistic, with different row and column score definitions for each statistic. The CMH statistics that PROC FREQ computes are the correlation statistic, the ANOVA (row mean scores) statistic, and the general association statistic. These statistics test the null hypothesis of no association against different alternative hypotheses. The following sections describe the computation of these CMH statistics.

CAUTION: The CMH statistics have low power for detecting an association in which the patterns of association for some of the strata are in the opposite direction of the patterns displayed by other strata. Thus, a nonsignificant CMH statistic suggests either that there is no association or that no pattern of association has enough strength or consistency to dominate any other pattern.

Correlation Statistic

The correlation statistic, popularized by Mantel and Haenszel, has 1 degree of freedom and is known as the Mantel-Haenszel statistic (Mantel and Haenszel 1959; Mantel 1963).

The alternative hypothesis for the correlation statistic is that there is a linear association between X and Y in at least one stratum. If either X or Y does not lie on an ordinal (or interval) scale, this statistic is not meaningful.

To compute the correlation statistic, PROC FREQ uses the formula for the generalized CMH statistic with the row and column scores determined by the SCORES= option in the TABLES statement. See the section "Scores" on page 3247 for more information about the available score types. The matrix of row scores $\mathbf{R}_h$ has dimension $1 \times R$, and the matrix of column scores $\mathbf{C}_h$ has dimension $1 \times C$.

When there is only one stratum, this CMH statistic reduces to $(n-1)r^2$, where r is the Pearson correlation coefficient between X and Y. When nonparametric (RANK or RIDIT) scores are specified, the statistic reduces to $(n-1)r_s^2$, where r_s is the Spearman rank correlation coefficient between X and Y. When there is more than one stratum, this CMH statistic becomes a stratum-adjusted correlation statistic.

ANOVA (Row Mean Scores) Statistic

The ANOVA statistic can be used only when the column variable Y lies on an ordinal (or interval) scale so that the mean score of Y is meaningful. For the ANOVA statistic, the mean score is computed for each row of the table, and the alternative hypothesis is that, for at least one stratum, the mean scores of the R rows are unequal. In other words, the statistic is sensitive to location differences among the R distributions of Y.

The matrix of column scores $\mathbf{C}_h$ has dimension $1 \times C$, and the column scores are determined by the SCORES= option.

The matrix of row scores $\mathbf{R}_h$ has dimension $(R-1) \times R$ and is created internally by PROC FREQ as

$$\mathbf{R}_h = [\mathbf{I}_{R-1}, -\mathbf{J}_{R-1}]$$

where $\mathbf{I}_{R-1}$ is an identity matrix of rank $R-1$ and $\mathbf{J}_{R-1}$ is an $(R-1) \times 1$ vector of ones. This matrix has the effect of forming $R-1$ independent contrasts of the R mean scores.

When there is only one stratum, this CMH statistic is essentially an analysis of variance (ANOVA) statistic in the sense that it is a function of the variance ratio F statistic that would be obtained from a one-way ANOVA on the dependent variable Y. If nonparametric scores are specified in this case, the ANOVA statistic is a Kruskal-Wallis test.

When there is more than one stratum, this CMH statistic corresponds to a stratum-adjusted ANOVA or Kruskal-Wallis test. In the special case where there is one subject per row and one subject per column in the contingency table of each stratum, this CMH statistic is identical to Friedman's chi-square. See Example 47.9 for an illustration.

General Association Statistic

The alternative hypothesis for the general association statistic is that, for at least one stratum, there is some kind of association between X and Y. This statistic is always interpretable because it does not require an ordinal scale for either X or Y.

For the general association statistic, the matrix $\mathbf{R}_h$ is the same as the one used for the ANOVA statistic. The matrix $\mathbf{C}_h$ is defined similarly as

$$\mathbf{C}_h = [\mathbf{I}_{C-1}, -\mathbf{J}_{C-1}]$$

PROC FREQ generates both score matrices internally. When there is only one stratum, the general association CMH statistic reduces to $Q_P(n-1)/n$, where Q_P is the Pearson chi-square statistic. When there is more than one stratum, the CMH statistic becomes a stratum-adjusted Pearson chi-square statistic. Note that a similar adjustment can be made by summing the Pearson chi-squares across the strata. However, the latter statistic requires a large sample size in each stratum to support the resulting chi-square distribution with $q(R–1)(C–1)$ degrees of freedom. The CMH statistic requires only a large overall sample size because it has only $(R–1)(C–1)$ degrees of freedom.

See Cochran (1954); Mantel and Haenszel (1959); Mantel (1963); Birch (1965); Landis, Heyman, and Koch (1978).

Mantel-Fleiss Criterion

If you specify the CMH(MANTELFLEISS) option in the TABLES statement, PROC FREQ computes the Mantel-Fleiss criterion for stratified 2×2 tables. The Mantel-Fleiss criterion can be used to assess the validity of the chi-square approximation for the distribution of the Mantel-Haenszel statistic for 2×2 tables. For more information, see Mantel and Fleiss (1980); Mantel and Haenszel (1959); Stokes, Davis, and Koch (2012); Dmitrienko et al. (2005).

The Mantel-Fleiss criterion is computed as

$$\mathrm{MF} = \min\left(\left[\sum_h m_{h11} - \sum_h (n_{h11})_L\right], \left[\sum_h (n_{h11})_U - \sum_h m_{h11}\right]\right)$$

where m_{h11} is the expected value of n_{h11} under the hypothesis of no association between the row and column variables in table h, $(n_{h11})_L$ is the minimum possible value of the table cell frequency, and $(n_{h11})_U$ is the maximum possible value,

$$\begin{aligned} m_{h11} &= n_{h1\cdot}\, n_{h\cdot 1} \,/\, n_h \\ (n_{h11})_L &= \max(\,0,\ \ n_{h1\cdot} - n_{h\cdot 2}\,) \\ (n_{h11})_U &= \min(\,n_{h\cdot 1},\ \ n_{h1\cdot}\,) \end{aligned}$$

The Mantel-Fleiss guideline accepts the validity of the Mantel-Haenszel approximation when the value of the criterion is at least 5. When the criterion is less than 5, PROC FREQ displays a warning.

Adjusted Odds Ratio and Relative Risk Estimates

The CMH option provides adjusted odds ratio and relative risk estimates for stratified 2×2 tables. For each of these measures, PROC FREQ computes a Mantel-Haenszel estimate and a logit estimate. These estimates apply to n-way table requests in the TABLES statement, when the row and column variables both have two levels.

For example, for the table request A*B*C*D, if the row and column variables C and D both have two levels, PROC FREQ provides odds ratio and relative risk estimates, adjusting for the confounding variables A and B.

The choice of an appropriate measure depends on the study design. For case-control (retrospective) studies, the odds ratio is appropriate. For cohort (prospective) or cross-sectional studies, the relative risk is appropriate. See the section "Odds Ratio and Relative Risks" on page 3287 for more information on these measures.

Throughout this section, z denotes the $100(1-\alpha/2)$th percentile of the standard normal distribution.

Odds Ratio, Case-Control Studies PROC FREQ provides Mantel-Haenszel and logit estimates for the common odds ratio for stratified 2×2 tables.

Mantel-Haenszel Estimator

The Mantel-Haenszel estimate of the common odds ratio is computed as

$$\mathrm{OR}_{\mathrm{MH}} = \left(\sum_h n_{h11}\, n_{h22}/n_h \right) \Big/ \left(\sum_h n_{h12}\, n_{h21}/n_h \right)$$

It is always computed unless the denominator is 0. For more information, see Mantel and Haenszel (1959) and Agresti (2002).

To compute confidence limits for the common odds ratio, PROC FREQ uses the Robins, Breslow, and Greenland (1986) variance estimate for $\log(\mathrm{OR}_{\mathrm{MH}})$. The $100(1-\alpha/2)\%$ confidence limits for the common odds ratio are

$$\left(\mathrm{OR}_{\mathrm{MH}} \times \exp(-z\hat{\sigma}),\ \ \mathrm{OR}_{\mathrm{MH}} \times \exp(z\hat{\sigma}) \right)$$

where

$$\begin{aligned}
\hat{\sigma}^2 &= \widehat{\mathrm{Var}}(\ \log(\mathrm{OR}_{\mathrm{MH}})\ ) \\
&= \frac{\sum_h (n_{h11}+n_{h22})(n_{h11}\, n_{h22})/n_h^2}{2\left(\sum_h n_{h11}\, n_{h22}/n_h\right)^2} \\
&\quad + \frac{\sum_h [(n_{h11}+n_{h22})(n_{h12}\, n_{h21}) + (n_{h12}+n_{h21})(n_{h11}\, n_{h22})]/n_h^2}{2\left(\sum_h n_{h11}\, n_{h22}/n_h\right)\left(\sum_h n_{h12}\, n_{h21}/n_h\right)} \\
&\quad + \frac{\sum_h (n_{h12}+n_{h21})(n_{h12}\, n_{h21})/n_h^2}{2\left(\sum_h n_{h12}\, n_{h21}/n_h\right)^2}
\end{aligned}$$

Note that the Mantel-Haenszel odds ratio estimator is less sensitive to small n_h than the logit estimator.

Logit Estimator

The adjusted logit estimate of the common odds ratio (Woolf 1955) is computed as

$$\mathrm{OR_L} = \exp\left(\sum_h w_h \log(\mathrm{OR}_h) \;/\; \sum_h w_h\right)$$

and the corresponding $100(1-\alpha)\%$ confidence limits are

$$\left(\mathrm{OR_L} \times \exp\left(-z/\sqrt{\sum_h w_h}\right),\;\; \mathrm{OR_L} \times \exp\left(z/\sqrt{\sum_h w_h}\right)\right)$$

where OR_h is the odds ratio for stratum h, and

$$w_h = 1/\mathrm{Var}(\log(\mathrm{OR}_h))$$

If any table cell frequency in a stratum h is 0, PROC FREQ adds 0.5 to each cell frequency in the stratum before computing OR_h and w_h (Haldane 1956) for the logit estimate. The procedure provides a warning when this occurs.

Relative Risks, Cohort Studies PROC FREQ provides Mantel-Haenszel and logit estimates of the common relative risks for stratified 2×2 tables.

Mantel-Haenszel Estimator

The Mantel-Haenszel estimate of the common relative risk for column 1 is computed as

$$\mathrm{RR_{MH}} = \left(\sum_h n_{h11}\, n_{h2\cdot} \,/\, n_h\right) \;/\; \left(\sum_h n_{h21}\, n_{h1\cdot} \,/\, n_h\right)$$

It is always computed unless the denominator is 0. See Mantel and Haenszel (1959) and Agresti (2002) for more information.

To compute confidence limits for the common relative risk, PROC FREQ uses the Greenland and Robins (1985) variance estimate for $\log(\mathrm{RR_{MH}})$. The $100(1-\alpha/2)\%$ confidence limits for the common relative risk are

$$\left(\mathrm{RR_{MH}} \times \exp(-z\hat{\sigma}),\;\; \mathrm{RR_{MH}} \times \exp(z\hat{\sigma})\right)$$

where

$$\hat{\sigma}^2 = \widehat{\mathrm{Var}}(\,\log(\mathrm{RR_{MH}})\,) = \frac{\sum_h (n_{h1\cdot}\, n_{h2\cdot}\, n_{h\cdot 1} - n_{h11}\, n_{h21}\, n_h)/n_h^2}{\left(\sum_h n_{h11}\, n_{h2\cdot}/n_h\right)\left(\sum_h n_{h21}\, n_{h1\cdot}/n_h\right)}$$

Logit Estimator

The adjusted logit estimate of the common relative risk for column 1 is computed as

$$\mathrm{RR_L} = \exp\left(\sum_h w_h \log(\mathrm{RR}_h) \;/\; \sum w_h\right)$$

and the corresponding $100(1-\alpha)\%$ confidence limits are

$$\left(\mathrm{RR_L} \times \exp\left(-z \;/\; \sqrt{\sum_h w_h} \right),\;\; \mathrm{RR_L} \times \exp\left(z \;/\; \sqrt{\sum_h w_h} \right) \right)$$

where RR_h is the column 1 relative risk estimate for stratum h and

$$w_h = 1 \;/\; \mathrm{Var}(\log(\mathrm{RR}_h))$$

If n_{h11} or n_{h21} is 0, PROC FREQ adds 0.5 to each cell frequency in the stratum before computing RR_h and w_h for the logit estimate. The procedure prints a warning when this occurs. For more information, see Kleinbaum, Kupper, and Morgenstern (1982, Sections 17.4 and 17.5).

Breslow-Day Test for Homogeneity of the Odds Ratios

When you specify the CMH option, PROC FREQ computes the Breslow-Day test for stratified 2×2 tables. It tests the null hypothesis that the odds ratios for the q strata are equal. When the null hypothesis is true, the statistic has approximately a chi-square distribution with q–1 degrees of freedom. See Breslow and Day (1980) and Agresti (2007) for more information.

The Breslow-Day statistic is computed as

$$Q_{\mathrm{BD}} = \sum_h (n_{h11} - \mathrm{E}(n_{h11} \mid \mathrm{OR_{MH}}))^2 \;/\; \mathrm{Var}(n_{h11} \mid \mathrm{OR_{MH}})$$

where E and Var denote expected value and variance, respectively. The summation does not include any table that contains a row or column that has a total frequency of 0. If $\mathrm{OR_{MH}}$ is 0 or undefined, PROC FREQ does not compute the statistic and prints a warning message.

For the Breslow-Day test to be valid, the sample size should be relatively large in each stratum, and at least 80% of the expected cell counts should be greater than 5. Note that this is a stricter sample size requirement than the requirement for the Cochran-Mantel-Haenszel test for $q \times 2 \times 2$ tables, in that each stratum sample size (not just the overall sample size) must be relatively large. Even when the Breslow-Day test is valid, it might not be very powerful against certain alternatives, as discussed in Breslow and Day (1980).

If you specify the BDT option, PROC FREQ computes the Breslow-Day test with Tarone's adjustment, which subtracts an adjustment factor from Q_{BD} to make the resulting statistic asymptotically chi-square. The Breslow-Day-Tarone statistic is computed as

$$Q_{\mathrm{BDT}} = Q_{\mathrm{BD}} - \left(\sum_h (n_{h11} - \mathrm{E}(n_{h11} \mid \mathrm{OR_{MH}})) \right)^2 \;/\; \sum_h \mathrm{Var}(n_{h11} \mid \mathrm{OR_{MH}})$$

See Tarone (1985); Jones et al. (1989); Breslow (1996) for more information.

Q Test for Homogeneity of Odds Ratios

PROC FREQ computes a Q test for homogeneity of odds ratios as

$$Q = \sum_h w_h (\theta_h - \bar{\theta})^2$$

where θ_h is the log odds ratio in stratum h and $\bar{\theta}$ is the logit estimate of the common log odds ratio. The stratum weights w_h are

$$w_h = 1/\mathrm{Var}(\theta_h)$$

where

$$\mathrm{Var}(\theta_h) = 1/n_{h11} + 1/n_{h12} + 1/n_{h21} + 1/n_{h22}$$

If any table cell frequency in a stratum is 0, PROC FREQ adds 0.5 to each cell frequency in the stratum before computing θ_h and w_h. For more information, see the sections "Odds Ratio" on page 3287 and "Adjusted Odds Ratio and Relative Risk Estimates" on page 3311.

Under the null hypothesis of homogeneity, the Q statistic has approximately a chi-square distribution with k–1 degrees of freedom, where k is the number of strata.

I-Square Measure of Heterogeneity

The I-square statistic (Higgins and Thompson 2002) is a measure of heterogeneity among strata for stratified 2×2 tables. I-square is expressed in percentage form and can be interpreted as the proportion of total variability that is due to between-strata variability. For more information, see Higgins et al. (2003) and Thorlund et al. (2012).

PROC FREQ computes I-square for the Q test for odds ratios as

$$I^2 = \max\left(100\% \times (Q - (k-1))/Q,\ 0 \right)$$

where k is the number of strata and Q is described in the section "Q Test for Homogeneity of Odds Ratios" on page 3313.

PROC FREQ computes uncertainty limits for I-square by using the test-based method of Higgins and Thompson (2002). This method constructs confidence limits for H, where $H^2 = Q/(k-1)$. When $Q > k$ or $k = 2$, the standard error of $\log(H)$ is computed as

$$\mathrm{SE}_1(\log(H)) = \left(\log(Q) - \log(k-1) \right) \ / \ 2\left(\sqrt{2Q} - \sqrt{2k-3}\right)$$

When $Q \le k$ and $k > 2$, the standard error of $\log(H)$ is computed as

$$\mathrm{SE}_0(\log(H)) = \sqrt{\left(1 - (1/3(k-2)^2) \right) \ / \ 2(k-2)}$$

The $100(1-\alpha)\%$ confidence limits for H are

$$\left(H \times \exp(-z_{\alpha/2} \times \mathrm{SE}(\log(H)),\ \ H \times \exp(z_{\alpha/2} \times \mathrm{SE}(\log(H)) \right)$$

The uncertainty limits for I^2 are computed by transforming the confidence limits for H, where $I^2 = 1 - (1/H^2)$.

When I^2 is 0, PROC FREQ sets the lower confidence limit to 0 and determines the upper limit by using the level α (instead of $\alpha/2$).

Zelen's Exact Test for Equal Odds Ratios

If you specify the EQOR option in the EXACT statement, PROC FREQ computes Zelen's exact test for equal odds ratios for stratified 2×2 tables. Zelen's test is an exact counterpart to the Breslow-Day asymptotic test for equal odds ratios. The reference set for Zelen's test includes all possible $q \times 2 \times 2$ tables with the same row, column, and stratum totals as the observed multiway table and with the same sum of cell (1,1) frequencies as the observed table. The test statistic is the probability of the observed $q \times 2 \times 2$ table conditional on the fixed margins, which is a product of hypergeometric probabilities.

The p-value for Zelen's test is the sum of all table probabilities that are less than or equal to the observed table probability, where the sum is computed over all tables in the reference set determined by the fixed margins and the observed sum of cell (1,1) frequencies. This test is similar to Fisher's exact test for two-way tables. For more information, see Zelen (1971); Hirji (2006); Agresti (1992). PROC FREQ computes Zelen's exact test by using the polynomial multiplication algorithm of Hirji et al. (1996).

Exact Confidence Limits for the Common Odds Ratio

If you specify the COMOR option in the EXACT statement, PROC FREQ computes exact confidence limits for the common odds ratio for stratified 2×2 tables. This computation assumes that the odds ratio is constant over all the 2×2 tables. Exact confidence limits are constructed from the distribution of $S = \sum_h n_{h11}$, conditional on the marginal totals of the 2×2 tables.

Because this is a discrete problem, the confidence coefficient for these exact confidence limits is not exactly $(1 - \alpha)$ but is at least $(1 - \alpha)$. Thus, these confidence limits are conservative. See Agresti (1992) for more information.

PROC FREQ computes exact confidence limits for the common odds ratio by using an algorithm based on Vollset, Hirji, and Elashoff (1991). See also Mehta, Patel, and Gray (1985).

Conditional on the marginal totals of 2×2 table h, let the random variable S_h denote the frequency of table cell (1,1). Given the row totals $n_{h1\cdot}$ and $n_{h2\cdot}$ and column totals $n_{h\cdot 1}$ and $n_{h\cdot 2}$, the lower and upper bounds for S_h are l_h and u_h,

$$\begin{aligned} l_h &= \max\,(\,0,\;\; n_{h1\cdot} - n_{h\cdot 2}\,) \\ u_h &= \min\,(\,n_{h1\cdot},\;\; n_{h\cdot 1}\,) \end{aligned}$$

Let C_{s_h} denote the hypergeometric coefficient,

$$C_{s_h} = \binom{n_{h\cdot 1}}{s_h}\binom{n_{h\cdot 2}}{n_{h1\cdot} - s_h}$$

and let ϕ denote the common odds ratio. Then the conditional distribution of S_h is

$$P(\,S_h = s_h \mid n_{1\cdot},\; n_{\cdot 1},\; n_{\cdot 2}\,) = C_{s_h}\,\phi^{\,s_h} \Big/ \sum_{x\,=\,l_h}^{x\,=\,u_h} C_x\,\phi^{\,x}$$

Summing over all the 2×2 tables, $S = \sum_h S_h$, and the lower and upper bounds of S are l and u,

$$l = \sum_h l_h \quad \text{and} \quad u = \sum_h u_h$$

The conditional distribution of the sum S is

$$P(\,S = s \mid n_{h1\cdot},\; n_{h\cdot 1},\; n_{h\cdot 2};\;\; h = 1, \ldots, q\,) = C_s\,\phi^{\,s} \Big/ \sum_{x\,=\,l}^{x\,=\,u} C_x\,\phi^{\,x}$$

where

$$C_s = \sum_{s_1 + \cdots + s_q\,=\,s} \left(\prod_h C_{s_h} \right)$$

Let s_0 denote the observed sum of cell (1,1) frequencies over the q tables. The following two equations are solved iteratively for lower and upper confidence limits for the common odds ratio, ϕ_1 and ϕ_2:

$$\sum_{x=s_0}^{x=u} C_x\, \phi_1^{\,x} \Big/ \sum_{x=l}^{x=u} C_x\, \phi_1^{\,x} \;=\; \alpha/2$$

$$\sum_{x=l}^{x=s_0} C_x\, \phi_2^{\,x} \Big/ \sum_{x=l}^{x=u} C_x\, \phi_2^{\,x} \;=\; \alpha/2$$

When the observed sum s_0 equals the lower bound l, PROC FREQ sets the lower confidence limit to 0 and determines the upper limit with level α. Similarly, when the observed sum s_0 equals the upper bound u, PROC FREQ sets the upper confidence limit to infinity and determines the lower limit with level α.

When you specify the COMOR option in the EXACT statement, PROC FREQ also computes the exact test that the common odds ratio equals one. Setting $\phi = 1$, the conditional distribution of the sum S under the null hypothesis becomes

$$P_0(\, S = s \mid n_{h1\cdot},\, n_{h\cdot 1},\, n_{h\cdot 2};\;\; h = 1, \ldots, q\,) = C_s \Big/ \sum_{x=l}^{x=u} C_x$$

The point probability for this exact test is the probability of the observed sum s_0 under the null hypothesis, conditional on the marginals of the stratified 2×2 tables, and is denoted by $P_0(s_0)$. The expected value of S under the null hypothesis is

$$\mathrm{E}_0(S) = \sum_{x=l}^{x=u} x\, C_x \Big/ \sum_{x=l}^{x=u} C_x$$

The one-sided exact p-value is computed from the conditional distribution as $P_0(S >= s_0)$ or $P_0(S \le s_0)$, depending on whether the observed sum s_0 is greater or less than $\mathrm{E}_0(S)$,

$$P_1 \;=\; P_0(\, S >= s_0\,) = \sum_{x=s_0}^{x=u} C_x \Big/ \sum_{x=l}^{x=u} C_x \quad \text{if} \;\; s_0 > \mathrm{E}_0(S)$$

$$P_1 \;=\; P_0(\, S <= s_0\,) = \sum_{x=l}^{x=s_0} C_x \Big/ \sum_{x=l}^{x=u} C_x \quad \text{if} \;\; s_0 \le \mathrm{E}_0(S)$$

PROC FREQ computes two-sided p-values for this test according to three different definitions. A two-sided p-value is computed as twice the one-sided p-value, setting the result equal to one if it exceeds one,

$$P_2^{\,a} = 2 \times P_1$$

In addition, a two-sided p-value is computed as the sum of all probabilities less than or equal to the point probability of the observed sum s_0, summing over all possible values of s, $\;l \le s \le u$,

$$P_2^{\,b} = \sum_{l \le s \le u : P_0(s) \le P_0(s_0)} P_0(s)$$

Also, a two-sided p-value is computed as the sum of the one-sided p-value and the corresponding area in the opposite tail of the distribution, equidistant from the expected value,

$$P_2^{\,c} = P_0\,(\, |S - \mathrm{E}_0(S)| \;\ge\; |s_0 - \mathrm{E}_0(S)|\,)$$

Gail-Simon Test for Qualitative Interactions

The GAILSIMON option in the TABLES statement provides the Gail-Simon test for qualitative interaction for stratified 2×2 tables. For more information, see Gail and Simon (1985); Silvapulle (2001); Dmitrienko et al. (2005).

The Gail-Simon test is based on the risk differences in stratified 2×2 tables, where the risk difference is defined as the row 1 risk (proportion in column 1) minus the row 2 risk. For more information, see the section "Risks and Risk Differences" on page 3272. By default, PROC FREQ uses column 1 risks to compute the Gail-Simon test. If you specify the GAILSIMON(COLUMN=2) option, PROC FREQ uses column 2 risks.

PROC FREQ computes the Gail-Simon test statistics as described in Gail and Simon (1985),

$$
\begin{aligned}
Q- &= \sum_h (d_h/s_h)^2 \; I(d_h > 0) \\
Q+ &= \sum_h (d_h/s_h)^2 \; I(d_h < 0) \\
Q &= \min\,(Q-,\; Q+)
\end{aligned}
$$

where d_h is the risk difference in table h, s_h is the standard error of the risk difference, and $I(d_h > 0)$ equals 1 if $d_h > 0$ and 0 otherwise. Similarly, $I(d_h < 0)$ equals 1 if $d_h < 0$ and 0 otherwise. The q 2×2 tables (strata) are indexed by $h = 1, 2, \ldots, q$.

The p-values for the Gail-Simon statistics are computed as

$$
\begin{aligned}
P(Q-) &= \sum_h (1 - F_h(Q-))\; B(h; n = q, p = 0.5) \\
P(Q+) &= \sum_h (1 - F_h(Q+))\; B(h; n = q, p = 0.5) \\
P(Q) &= \sum_{h=1}^{q-1} (1 - F_h(Q))\; B(h; n = (q-1), p = 0.5)
\end{aligned}
$$

where $F_h(\cdot)$ is the cumulative chi-square distribution function with h degrees of freedom and $B(h; n, p)$ is the binomial probability function with parameters n and p. The statistic Q tests the null hypothesis of no qualitative interaction. The statistic $Q-$ tests the null hypothesis of positive risk differences. A small p-value for $Q-$ indicates negative differences; similarly, a small p-value for $Q+$ indicates positive risk differences.

Exact Statistics

Exact statistics can be useful in situations where the asymptotic assumptions are not met and therefore the asymptotic p-values might not be close approximations for the true p-values. Standard asymptotic methods involve the assumption that the test statistic follows a particular distribution when the sample size is sufficiently large. When the sample size is not large, asymptotic results might not be valid. Asymptotic results might also be unreliable when the distribution of the data is sparse, skewed, or heavily tied. For more information, see Agresti (2007) and Bishop, Fienberg, and Holland (1975). Exact computations are based on the statistical theory of exact conditional inference for contingency tables, which is reviewed by Agresti (1992).

In addition to the computation of exact p-values, PROC FREQ provides the option to estimate exact p-values by Monte Carlo simulation. This can be useful for large problems where exact computations require a substantial amount of time and memory but asymptotic approximations might not be sufficient.

Exact p-values are available for many tests that PROC FREQ performs. For one-way tables, PROC FREQ provides exact p-values for the binomial proportion test, the chi-square goodness-of-fit test, and the likelihood ratio chi-square test. PROC FREQ also provides exact (Clopper-Pearson) confidence limits for the binomial proportion.

For two-way tables, PROC FREQ provides exact p-values for the following tests: Pearson chi-square test, likelihood ratio chi-square test, Mantel-Haenszel chi-square test, Fisher's exact test, Jonckheere-Terpstra test, Cochran-Armitage test for trend, and the symmetry test. PROC FREQ also provides exact p-values for tests of the following statistics: Pearson correlation coefficient, Spearman correlation coefficient, Kendall's tau-b, Stuart's tau-c, Somers' $D(C|R)$, Somers' $D(R|C)$, simple kappa coefficient, and weighted kappa coefficient.

For 2×2 tables, PROC FREQ provides the exact McNemar's test, exact confidence limits for the odds ratio, and Barnard's unconditional exact test for the risk (proportion) difference. PROC FREQ also provides exact unconditional confidence limits for the risk (proportion) difference and for the relative risk (ratio of proportions). For stratified 2×2 tables, PROC FREQ provides Zelen's exact test for equal odds ratios, exact confidence limits for the common odds ratio, and an exact test for the common odds ratio.

The following sections summarize the exact computational algorithms, define the exact p-values that PROC FREQ computes, discuss the computational resource requirements, and describe the Monte Carlo estimation option.

Computational Algorithms

PROC FREQ computes exact p-values for general $R \times C$ tables by using the network algorithm, which was developed by Mehta and Patel (1983). This algorithm provides a substantial advantage over direct enumeration, which can be very time-consuming and feasible only for small problems. See Agresti (1992) for a review of algorithms for computation of exact p-values, and see Mehta, Patel, and Tsiatis (1984) and Mehta, Patel, and Senchaudhuri (1991) for information about the performance of the network algorithm.

To implement the network algorithm, PROC FREQ defines a reference set from the input data. For most exact tests that PROC FREQ provides, the reference set includes all tables that have the same marginal row and column sums as the observed table. Corresponding to the reference set, the network algorithm forms a directed acyclic network consisting of nodes in a number of stages. A path through the network corresponds to a distinct table in the reference set. The distances between nodes are defined so that the total distance of a path through the network is the corresponding value of the test statistic. At each node, the algorithm computes the shortest and longest path distances for all the paths that pass through that node. For statistics

that can be expressed as a linear combination of cell frequencies multiplied by increasing row and column scores, PROC FREQ computes shortest and longest path distances by using the algorithm of Agresti, Mehta, and Patel (1990). For statistics of other forms, PROC FREQ computes an upper bound for the longest path and a lower bound for the shortest path by following the approach of Valz and Thompson (1994).

The longest and shortest path distances (bounds) for a node are compared to the value of the test statistic to determine whether all paths through the node contribute to the p-value, no paths through the node contribute to the p-value, or neither of these situations occurs. If all paths through the node contribute, the p-value is incremented accordingly, and these paths are eliminated from further analysis. If no paths contribute, these paths are eliminated from further analysis. Otherwise, the algorithm continues to process this node and the associated paths. The algorithm finishes when all nodes have been accounted for.

PROC FREQ performs the network algorithm by using full numerical precision to represent all statistics, row and column scores, and other quantities in the computations. Although it is possible to use rounding to improve the speed and memory requirements of the algorithm, PROC FREQ does not use rounding because it might reduce the accuracy of the results.

For one-way tables, PROC FREQ computes the exact chi-square goodness-of-fit test by the method of Radlow and Alf (1975). PROC FREQ generates all possible one-way tables with the observed total sample size and number of categories. For each possible table, PROC FREQ compares its chi-square value with the value for the observed table. If the table's chi-square value is greater than or equal to the observed chi-square, PROC FREQ increments the exact p-value by the probability of that table, which is calculated under the null hypothesis by using the multinomial frequency distribution. By default, the null hypothesis states that all categories have equal proportions. If you specify null hypothesis proportions or frequencies by using the TESTP= or TESTF= option in the TABLES statement, PROC FREQ calculates the exact chi-square test based on that null hypothesis.

Other exact computations are described in sections about the individual statistics. For information about the computation of exact confidence limits and tests for the binomial proportion, see the section "Binomial Proportion" on page 3263. For information about computation of exact confidence limits for the odds ratio, see the subsection "Exact Confidence Limits" in the section "Confidence Limits for the Odds Ratio" on page 3288. For information about other exact computations, see the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Risk Difference" on page 3274, the subsection "Exact Unconditional Confidence Limits" in the section "Confidence Limits for the Relative Risk" on page 3291, and the sections "Exact Symmetry Test" on page 3301, "Exact Confidence Limits for the Common Odds Ratio" on page 3315 and "Zelen's Exact Test for Equal Odds Ratios" on page 3314.

Definition of p-Values

For several tests in PROC FREQ, the test statistic is nonnegative, and large values of the test statistic indicate a departure from the null hypothesis. Such nondirectional tests include the Pearson chi-square, the likelihood ratio chi-square, the Mantel-Haenszel chi-square, Fisher's exact test for tables larger than 2×2, McNemar's test, the symmetry test, and the one-way chi-square goodness-of-fit test. The exact p-value for a nondirectional test is the sum of probabilities for those tables having a test statistic greater than or equal to the value of the observed test statistic.

There are other tests where it might be appropriate to test against either a one-sided or a two-sided alternative hypothesis. For example, when you test the null hypothesis that the true parameter value equals 0 ($T = 0$), the alternative of interest might be one-sided ($T \leq 0$, or $T \geq 0$) or two-sided ($T \neq 0$). Such tests include the Pearson correlation coefficient, Spearman correlation coefficient, Jonckheere-Terpstra test, Cochran-Armitage test for trend, simple kappa coefficient, and weighted kappa coefficient. For these tests, PROC FREQ displays

the right-sided p-value when the observed value of the test statistic is greater than its expected value. The right-sided p-value is the sum of probabilities for those tables for which the test statistic is greater than or equal to the observed test statistic. Otherwise, when the observed test statistic is less than or equal to the expected value, PROC FREQ displays the left-sided p-value. The left-sided p-value is the sum of probabilities for those tables for which the test statistic is less than or equal to the one observed. The one-sided p-value P_1 can be expressed as

$$P_1 = \begin{cases} \text{Prob(Test Statistic} \geq t \text{)} & \text{if} \quad t > \mathrm{E}_0(T) \\ \text{Prob(Test Statistic} \leq t \text{)} & \text{if} \quad t \leq \mathrm{E}_0(T) \end{cases}$$

where t is the observed value of the test statistic and $\mathrm{E}_0(T)$ is the expected value of the test statistic under the null hypothesis. PROC FREQ computes the two-sided p-value as the sum of the one-sided p-value and the corresponding area in the opposite tail of the distribution of the statistic, equidistant from the expected value. The two-sided p-value P_2 can be expressed as

$$P_2 = \mathrm{Prob}\left(|\text{Test Statistic} - \mathrm{E}_0(T)| \geq |t - \mathrm{E}_0(T)| \right)$$

If you specify the POINT option in the EXACT statement, PROC FREQ provides exact point probabilities for the exact tests. The exact point probability is the exact probability that the test statistic equals the observed value.

If you specify the MIDP option in the EXACT statement, PROC FREQ provides exact mid-p-values. The exact mid p-value is defined as the exact p-value minus half the exact point probability, which equals the average of $\mathrm{Prob}(\text{Test Statistic} \geq t)$ and $\mathrm{Prob}(\text{Test Statistic} > t)$ for a right-sided test. The exact mid p-value is smaller and less conservative than the non-adjusted exact p-value. For more information, see Agresti (2013, section 1.1.4) and Hirji (2006, sections 2.5 and 2.11.1).

Computational Resources

PROC FREQ uses relatively fast and efficient algorithms for exact computations. These algorithms, together with improvements in computing power, make it feasible to perform exact computations for data where previously only asymptotic methods could be applied. Nevertheless, depending on your available computing resources, exact computations for some very large problems might require a prohibitive amount of time and memory. For such large problems, consider whether exact methods are really needed or whether asymptotic methods might give results that are very close to the exact results while requiring much less computing time and memory. When asymptotic methods might not be sufficient for such large problems, consider using Monte Carlo estimation of exact p-values, which is described in the section "Monte Carlo Estimation" on page 3321.

There is no formula that can predict in advance how much time and memory are needed to compute an exact p-value for a specific data set and test. The time and memory requirements depend on several factors, which include the following: the total number of observations, the number of rows and columns in the table, the particular arrangement of the observations into table cells, and the test to be performed. Generally, larger problems (in terms of total sample size, number of rows, and number of columns) tend to require more time and memory. For a fixed total sample size, time and memory requirements tend to increase as the number of rows and number of columns increase because of the corresponding increase in the number of reference set tables. For a fixed sample size, time and memory requirements also tend to increase as the marginal row and column totals become more homogeneous. For more information, see Agresti, Mehta, and Patel (1990) and Gail and Mantel (1977).

While PROC FREQ is computing an exact p-value, you can terminate the computation by pressing the system interrupt key sequence and choosing to stop computations. For more information, see the *SAS Companion* for your system. After you terminate an exact computation, PROC FREQ completes all other remaining tasks. The procedure reports missing values for any exact p-values that were not computed before termination.

To limit the amount of time that PROC FREQ uses for exact computations, you can specify the MAXTIME= option in the EXACT statement. This option sets the maximum amount of clock time (in seconds) that PROC FREQ can use to compute an exact p-value. If PROC FREQ does not finish an exact computation in the time that you specify, the procedure terminates the computation and completes the remaining tasks.

Monte Carlo Estimation

When you specify the MC option in the EXACT statement, PROC FREQ computes Monte Carlo estimates of exact p-values. Monte Carlo estimation can be useful for large problems where exact computations require a substantial amount of time and memory but asymptotic approximations might not be sufficient. Monte Carlo estimates are available for all exact tests that PROC FREQ provides except the binomial proportion test and those tests that apply only to 2×2 or $h \times 2 \times 2$ tables.

To describe the precision of a Monte Carlo estimate, PROC FREQ provides the asymptotic standard error and $100(1-\alpha)\%$ confidence limits. You can specify the confidence level α in the ALPHA= option in the EXACT statement; by default, ALPHA=0.01, which produces 99% confidence limits.

You can specify the number of Monte Carlo samples by using the N=n option in the EXACT statement. By default, PROC FREQ uses 10,000 samples to compute a Monte Carlo estimate. To improve the precision of the Monte Carlo estimates, you can specify a larger value of n; this increases the computation time because more samples are generated. To reduce the computation time, you can specify a smaller value of n.

PROC FREQ computes a Monte Carlo estimate of an exact p-value by generating a random sample of tables from the reference set for the exact test. For most exact tests that PROC FREQ provides, the reference set includes tables that have the same total sample size, row sums, and column sums as the observed table. (For the exact symmetry test, the reference set includes tables that have the same total sample size as the observed table and the same frequency sums of the off-diagonal table cell pairs.)

PROC FREQ generates a random sample of tables from the reference set by using the algorithm of Agresti, Wackerly, and Boyett (1979), which generates tables in proportion to their hypergeometric probabilities conditional on the marginal frequencies. For each sample table, PROC FREQ computes the value of the test statistic and compares it to the value of the test statistic for the observed table. To estimate a right-sided p-value, PROC FREQ counts all sample tables for which the test statistic is greater than or equal to the observed test statistic. The estimate of the p-value is the number of these tables divided by the total number of sample tables, which can be expressed as

$$\begin{aligned}
\hat{p}_{\mathrm{mc}} &= m \,/\, n \\
m &= \text{number of samples for which (test statistic} \geq t_o) \\
n &= \text{total number of samples} \\
t_o &= \text{observed test statistic}
\end{aligned}$$

PROC FREQ computes estimates of left-sided and two-sided exact p-values similarly. For left-sided exact p-values, PROC FREQ evaluates whether the sample test statistics are less than or equal to the observed test statistic. For two-sided exact p-values, PROC FREQ compares sample test statistics to the observed test statistic by using the definition of the two-sided p-value (P_2) for the test. For more information, see the section "Definition of p-Values" on page 3319 and descriptions of the individual tests.

The variable m has a binomial distribution with n trials and success probability p. The asymptotic standard error of the Monte Carlo estimate is

$$\mathrm{se}(\hat{p}_{\mathrm{mc}}) = \sqrt{\hat{p}_{\mathrm{mc}}\,(1-\hat{p}_{\mathrm{mc}})\,/\,(n-1)}$$

PROC FREQ constructs asymptotic confidence limits for the exact p-value as

$$\hat{p}_{\mathrm{mc}} \pm \left(z_{\alpha/2} \times \mathrm{se}(\hat{p}_{\mathrm{mc}})\right)$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution and the confidence level α is determined by the ALPHA= option in the EXACT statement.

When the Monte Carlo estimate $\hat{p}_{\mathrm{mc}}$ is 0, PROC FREQ computes confidence limits for the p-value as

$$\left(0,\ \ 1-\alpha^{(1/n)}\right)$$

When the Monte Carlo estimate $\hat{p}_{\mathrm{mc}}$ is 1, PROC FREQ computes confidence limits for the p-value as

$$\left(\alpha^{(1/n)},\ \ 1\right)$$

Computational Resources

For each variable in a table request, PROC FREQ stores all of the levels in memory. If all variables are numeric and not formatted, this requires about 84 bytes for each variable level. When there are character variables or formatted numeric variables, the memory that is required depends on the formatted variable lengths, with longer formatted lengths requiring more memory. The number of levels for each variable is limited only by the largest integer that your operating environment can store.

For any single crosstabulation table requested, PROC FREQ builds the entire table in memory, regardless of whether the table has cell frequencies of 0. Thus, if the numeric variables A, B, and C each have 10 levels, PROC FREQ requires 2520 bytes to store the variable levels for the table request A*B*C, as follows:

```
3 variables * 10 levels/variable * 84 bytes/level
```

In addition, PROC FREQ requires 8000 bytes to store the table cell frequencies

```
1000 cells * 8 bytes/cell
```

even though there might be only 10 observations.

When the variables have many levels or when there are many multiway tables, your computer might not have enough memory to construct the tables. If PROC FREQ runs out of memory while constructing tables, it stops collecting levels for the variable with the most levels and returns the memory that is used by that variable. The procedure then builds the tables that do not contain the disabled variables.

If there is not enough memory for your table request and if increasing the available memory is impractical, you can reduce the number of multiway tables or variable levels. If you are not using the CMH or AGREE option in the TABLES statement to compute statistics across strata, reduce the number of multiway tables

by using PROC SORT to sort the data set by one or more of the variables or by using the DATA step to create an index for the variables. Then remove the sorted or indexed variables from the TABLES statement and include a BY statement that uses these variables. You can also reduce memory requirements by using a FORMAT statement in the PROC FREQ step to reduce the number of levels. In addition, reducing the formatted variable lengths reduces the amount of memory that is needed to store the variable levels. For more information about using formats, see the section "Grouping with Formats" on page 3242.

Output Data Sets

PROC FREQ produces two types of output data sets that you can use with other statistical and reporting procedures. You can request these data sets as follows:

- Specify the OUT= option in a TABLES statement. This creates an output data set that contains frequency or crosstabulation table counts and percentages
- Specify an OUTPUT statement. This creates an output data set that contains statistics.

PROC FREQ does not display the output data sets. Use PROC PRINT, PROC REPORT, or any other SAS reporting tool to display an output data set.

In addition to these two output data sets, you can create a SAS data set from any piece of PROC FREQ output by using the Output Delivery System. See the section "ODS Table Names" on page 3336 for more information.

Contents of the TABLES Statement Output Data Set

The OUT= option in the TABLES statement creates an output data set that contains one observation for each combination of variable values (or table cell) in the last table request. By default, each observation contains the frequency and percentage for the table cell. When the input data set contains missing values, the output data set also contains an observation with the frequency of missing values. The output data set includes the following variables:

- BY variables
- table request variables, such as A, B, C, and D in the table request A*B*C*D
- COUNT, which contains the table cell frequency
- PERCENT, which contains the table cell percentage

If you specify the OUTEXPECT option in the TABLES statement for a two-way or multiway table, the output data set also includes expected frequencies. If you specify the OUTPCT option for a two-way or multiway table, the output data set also includes row, column, and table percentages. The additional variables are as follows:

- EXPECTED, which contains the expected frequency
- PCT_TABL, which contains the percentage of two-way table frequency, for n-way tables where $n > 2$

- PCT_ROW, which contains the percentage of row frequency
- PCT_COL, which contains the percentage of column frequency

If you specify the OUTCUM option in the TABLES statement for a one-way table, the output data set also includes cumulative frequencies and cumulative percentages. The additional variables are as follows:

- CUM_FREQ, which contains the cumulative frequency
- CUM_PCT, which contains the cumulative percentage

The OUTCUM option has no effect for two-way or multiway tables.

The following PROC FREQ statements create an output data set of frequencies and percentages:

```
proc freq;
   tables A A*B / out=D;
run;
```

The output data set D contains frequencies and percentages for the table of A by B, which is the last table request listed in the TABLES statement. If A has two levels (1 and 2), B has three levels (1,2, and 3), and no table cell count is 0 or missing, the output data set D includes six observations, one for each combination of A and B levels. The first observation corresponds to A=1 and B=1; the second observation corresponds to A=1 and B=2; and so on. The data set includes the variables COUNT and PERCENT. The value of COUNT is the number of observations with the given combination of A and B levels. The value of PERCENT is the percentage of the total number of observations with that A and B combination.

When PROC FREQ combines different variable values into the same formatted level, the output data set contains the smallest internal value for the formatted level. For example, suppose a variable X has the values 1.1., 1.4, 1.7, 2.1, and 2.3. When you submit the statement

```
format X 1.;
```

in a PROC FREQ step, the formatted levels listed in the frequency table for X are 1 and 2. If you create an output data set with the frequency counts, the internal values of the levels of X are 1.1 and 1.7. To report the internal values of X when you display the output data set, use a format of 3.1 for X.

Contents of the OUTPUT Statement Output Data Set

The OUTPUT statement creates a SAS data set that contains statistics computed by PROC FREQ. Table 47.7 lists the statistics that can be stored in the output data set. You identify which statistics to include by specifying *output-options*. For more information, see the description of the OUTPUT statement.

If you specify multiple TABLES statements or multiple table requests in a single TABLES statement, the contents of the output data set correspond to the last table request.

For a one-way table or a two-way table, the output data set contains one observation that stores the requested statistics for the table. For a multiway table, the output data set contains an observation for each two-way table (stratum) of the multiway crosstabulation. If you request summary statistics for the multiway table, the output data set also contains an observation that stores the across-strata summary statistics. If you use a BY

statement, the output data set contains an observation (for one-way or two-way tables) or set of observations (for multiway tables) for each BY group.

The OUTPUT data set can include the following variables:

- BY variables
- Variables that identify the stratum for multiway tables, such as A and B in the table request A*B*C*D
- Variables that contain the specified statistics

In addition to the specified estimate or test statistic, the output data set includes associated values such as standard errors, confidence limits, *p*-values, and degrees of freedom.

PROC FREQ constructs variable names for the statistics in the output data set by enclosing the *output-option* names in underscores. Variable names for the corresponding standard errors, confidence limits, *p*-values, and degrees of freedom are formed by combining the *output-option* names with prefixes that identify the associated values. Table 47.21 lists the prefixes and their descriptions.

Table 47.21 Output Data Set Variable Name Prefixes

Prefix	Description
E_	Asymptotic standard error (ASE)
L_	Lower confidence limit
U_	Upper confidence limit
E0_	Null hypothesis ASE
Z_	Standardized value
DF_	Degrees of freedom
P_	*p*-value
P2_	Two-sided *p*-value
PL_	Left-sided *p*-value
PR_	Right-sided *p*-value
XP_	Exact *p*-value
XP2_	Exact two-sided *p*-value
XPL_	Exact left-sided *p*-value
XPR_	Exact right-sided *p*-value
XPT_	Exact point probability
XMP_	Exact mid *p*-value
XL_	Exact lower confidence limit
XU_	Exact upper confidence limit

For example, the PCHI *output-option* in the OUTPUT statement includes the Pearson chi-square test in the output data set. The variable names for the Pearson chi-square statistic, its degrees of freedom, and the corresponding *p*-value are _PCHI_, DF_PCHI, and P_PCHI, respectively.

Displayed Output

Number of Variable Levels Table

If you specify the NLEVELS option in the PROC FREQ statement, PROC FREQ displays the "Number of Variable Levels" table. This table provides the number of levels for all variables named in the TABLES statements. PROC FREQ determines the variable levels from the formatted variable values. For more information, see the section "Grouping with Formats" on page 3242. The "Number of Variable Levels" table contains the following information:

- Variable name
- Levels, which is the total number of levels of the variable
- Number of Nonmissing Levels, if there are missing levels for any of the variables
- Number of Missing Levels, if there are missing levels for any of the variables

One-Way Frequency Tables

PROC FREQ displays one-way frequency tables for all one-way table requests in the TABLES statements, unless you specify the NOPRINT option in the PROC FREQ statement or the NOPRINT option in the TABLES statement. For a one-way table showing the frequency distribution of a single variable, PROC FREQ displays the name of the variable and its values. For each variable value or level, PROC FREQ displays the following information:

- Frequency count, which is the number of observations in the level. (The ONEWAY(NOFREQ) option suppresses the frequencies.)
- Test Frequency count, if you specify the CHISQ and TESTF= options to request a chi-square goodness-of-fit test for specified frequencies. (The ONEWAY(NOFREQ) option suppresses the test frequencies.)
- Expected frequency, if you specify the ONEWAY(EXPECTED) option
- Deviation of the frequency from the expected value, if you specify the ONEWAY(DEVIATION) option
- Pearson Residual, if you specify the ONEWAY(RESIDUAL) option
- Cell Chi-Square, which is the contribution to the Pearson chi-square statistic, if you specify the ONEWAY(CELLCHI2) option
- Percent, which is the percentage of the total number of observations. (The ONEWAY(NOPERCENT) option suppresses the percentages.)
- Test Percent, if you specify the CHISQ and TESTP= options to request a chi-square goodness-of-fit test for specified percents. (The ONEWAY(NOPERCENT) option suppresses the test percentages.)
- Cumulative Frequency, which is the accumulated frequency of the current level and all levels that are listed above it in the table. The last cumulative frequency in the table is the total number of nonmissing observations. (Both the ONEWAY(NOCUM) option and the ONEWAY(NOFREQ) option suppress the cumulative frequencies.)

- Cumulative Percent, which is the accumulated percentage of the current level and all levels that are listed above it in the table. (Both the ONEWAY(NOCUM) option and the ONEWAY(NOPERCENT) option suppress the cumulative percentages.)

The one-way table also displays the Frequency Missing, which is the number of observations with missing values.

Statistics for One-Way Frequency Tables

For one-way tables, two statistical options are available in the TABLES statement. The CHISQ option provides a chi-square goodness-of-fit test, and the BINOMIAL option provides binomial proportion statistics and tests. PROC FREQ displays the following information, unless you specify the NOPRINT option in the PROC FREQ statement:

- If you specify the CHISQ option for a one-way table, PROC FREQ provides a chi-square goodness-of-fit test, displaying the Chi-Square statistic, the degrees of freedom (DF), and the probability value (Pr > ChiSq). If you specify the CHISQ option in the EXACT statement, PROC FREQ also displays the exact probability value for this test. If you specify the POINT option with the CHISQ option in the EXACT statement, PROC FREQ displays the exact point probability for the test statistic. If you specify the MIDP option in the EXACT statement, PROC FREQ displays the exact mid *p*-value for the chi-square test.

- If you specify the BINOMIAL option for a one-way table, PROC FREQ displays the estimate of the binomial Proportion, which is the proportion of observations in the first class listed in the one-way table. PROC FREQ also displays the asymptotic standard error (ASE) and the asymptotic (Wald) and exact (Clopper-Pearson) confidence limits by default. For the binomial proportion test, PROC FREQ displays the asymptotic standard error under the null hypothesis (ASE Under H0), the standardized test statistic (Z), and the one-sided and two-sided probability values.

  If you specify the BINOMIAL option in the EXACT statement, PROC FREQ also displays the exact one-sided and two-sided probability values for this test. If you specify the POINT option with the BINOMIAL option in the EXACT statement, PROC FREQ displays the exact point probability for the test. If you specify the MIDP option in the EXACT statement, PROC FREQ displays the exact mid *p*-value for the binomial proportion test.

- If you request binomial confidence limits by specifying the BINOMIAL(CL=) option, PROC FREQ displays the "Binomial Confidence Limits" table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request. In addition to Wald and Clopper-Pearson (Exact) confidence limits, you can request the following confidence limit types for the binomial proportion: Agresti-Coull, Blaker, Jeffreys, Likelihood Ratio, Logit, Mid-p, and Wilson (score).

- If you request a binomial noninferiority or superiority test by specifying the NONINF or SUP *binomial-option*, PROC FREQ displays a Noninferiority Analysis or Superiority Analysis table that contains the following information: the binomial Proportion, the test ASE (under H0 or Sample), the test statistic Z, the probability value, the noninferiority or superiority limit, and the test confidence limits. If you specify the BINOMIAL option in the EXACT statement, PROC FREQ also provides the exact probability value for the test, and exact test confidence limits.

- If you request a binomial equivalence test by specifying the EQUIV *binomial-option*, PROC FREQ displays an Equivalence Analysis table that contains the following information: binomial Proportion

and the test ASE (under H0 or Sample). PROC FREQ displays two one-sided tests (TOST) for equivalence, which include test statistics (Z) and probability values for the Lower and Upper tests, together with the Overall probability value. PROC FREQ also displays the equivalence limits and the test-based confidence limits. If you specify the BINOMIAL option in the EXACT statement, PROC FREQ provides exact probability values for the TOST and exact test-based confidence limits.

Two-Way and Multiway Tables

PROC FREQ displays all multiway table requests in the TABLES statements, unless you specify the NOPRINT option in the PROC FREQ statement or the NOPRINT option in the TABLES statement.

For two-way to multiway crosstabulation tables, the values of the last variable in the table request form the table columns. The values of the next-to-last variable form the rows. Each level (or combination of levels) of the other variables forms one stratum.

There are three ways to display multiway tables in PROC FREQ. By default, PROC FREQ displays multiway tables as separate two-way crosstabulation tables for each stratum of the multiway table. Also by default, PROC FREQ displays these two-way crosstabulation tables in table cell format. Alternatively, if you specify the CROSSLIST option, PROC FREQ displays the two-way crosstabulation tables in ODS column format. If you specify the LIST option, PROC FREQ displays multiway tables in list format, which presents the entire multiway crosstabulation in a single table.

Crosstabulation Tables

By default, PROC FREQ displays two-way crosstabulation tables in table cell format. The row variable values are listed down the side of the table, the column variable values are listed across the top of the table, and each row and column variable level combination forms a table cell.

Each cell of a crosstabulation table can contain the following information:

- Frequency, which is the number of observations in the table cell. (The NOFREQ option suppresses this information.)
- Expected frequency under the hypothesis of independence, if you specify the EXPECTED option
- Deviation of the cell frequency from the expected value, if you specify the DEVIATION option
- Cell Chi-Square, which is the cell's contribution to the total chi-square statistic, if you specify the CELLCHI2 option
- Tot Pct, which is the cell's percentage of the total multiway table frequency, for n-way tables when $n > 2$, if you specify the TOTPCT option
- Percent, which is the cell's percentage of the total (two-way table) frequency. (The NOPERCENT option suppresses this information.)
- Row Pct, or the row percentage, which is the cell's percentage of the total frequency for its row. (The NOROW option suppresses this information.)
- Col Pct, or column percentage, which is the cell's percentage of the total frequency for its column. (The NOCOL option suppresses this information.)
- Cumulative Col%, or cumulative column percentage, if you specify the CUMCOL option

The table also displays the Frequency Missing, which is the number of observations with missing values.

CROSSLIST Tables

If you specify the CROSSLIST option, PROC FREQ displays two-way crosstabulation tables in ODS column format. The CROSSLIST column format is different from the default crosstabulation table cell format, but the CROSSLIST table provides the same information (frequencies, percentages, and other statistics) as the default crosstabulation table.

In the CROSSLIST table format, the rows of the display correspond to the crosstabulation table cells, and the columns of the display correspond to descriptive statistics such as frequencies and percentages. Each table cell is identified by the values of its TABLES row and column variable levels, with all column variable levels listed within each row variable level. The CROSSLIST table also provides row totals, column totals, and overall table totals.

For a crosstabulation table in CROSSLIST format, PROC FREQ displays the following information:

- the row variable name and values
- the column variable name and values
- Frequency, which is the number of observations in the table cell. (The CROSSLIST(NOFREQ) option suppresses the frequencies.)
- Expected cell frequency under the hypothesis of independence, if you specify the CROSSLIST(EXPECTED) option
- Deviation of the cell frequency from the expected value, if you specify the CROSSLIST(DEVIATION) option
- Standardized Residual, if you specify the CROSSLIST(STDRES) option
- Pearson Residual, if you specify the CROSSLIST(PEARSONRES) option
- Cell Chi-Square, which is the cell's contribution to the total chi-square statistic, if you specify the CROSSLIST(CELLCHI2) option
- Total Percent, which is the cell's percentage of the total multiway table frequency for n-way tables when $n > 2$, if you specify the CROSSLIST(TOTPCT) option
- Percent, which is the cell's percentage of the total (two-way table) frequency. (The CROSSLIST(NOPERCENT) option suppresses the percentages.)
- Row Percent, which is the cell's percentage of the total frequency in its row. (The CROSSLIST(NOROW) option suppresses the row percentages.)
- Column Percent, which is the cell's percentage of the total frequency in its column. (The CROSSLIST(NOCOL) option suppresses the column percentages.)

The table also displays the Frequency Missing, which is the number of observations with missing values.

LIST Tables

If you specify the LIST option in the TABLES statement, PROC FREQ displays multiway tables in a list format rather than as crosstabulation tables. The LIST option displays the entire multiway table in one table, instead of displaying a separate two-way table for each stratum. The LIST option is not available when you also request statistical options. Unlike the default crosstabulation output, the LIST output does not display row percentages, column percentages, and optional information such as expected frequencies and cell chi-squares.

For a multiway table in list format, PROC FREQ displays the following information:

- the variable names and values
- Frequency, which is the number of observations in the table cell (level). (The LIST(NOFREQ) option suppresses the frequencies.)
- Percent, which is the level's percentage of the total number of observations. (The LIST(NOPERCENT) option suppresses the percentages.)
- Cumulative Frequency, which is the accumulated frequency of the current level and all levels that are listed above it in the table. The last cumulative frequency in the table is the total number of nonmissing observations. (Both the LIST(NOFREQ) option and the LIST(NOCUM) option suppress the cumulative frequencies.)
- Cumulative Percent, which is the accumulated percentage of the current level and all levels that are listed above it in the table. (Both the LIST(NOPERCENT) option and the LIST(NOCUM) option suppress the cumulative percentages.)

The table also displays the Frequency Missing, which is the number of observations with missing values.

Statistics for Two-Way and Multiway Tables

PROC FREQ computes statistical tests and measures for crosstabulation tables, depending on which statements and options you specify. You can suppress the display of these results by specifying the NOPRINT option in the PROC FREQ statement. With any of the following information, PROC FREQ also displays the Sample Size and the Frequency Missing.

- If you specify the SCOROUT option in the TABLES statement, PROC FREQ displays the Row Scores and Column Scores that it uses for statistical computations. The Row Scores table displays the row variable values and the Score corresponding to each value. The Column Scores table displays the column variable values and the corresponding Scores. PROC FREQ also identifies the score type used to compute the row and column scores. You can specify the score type with the SCORES= option in the TABLES statement.
- If you specify the CHISQ option, PROC FREQ displays the following statistics for each two-way table: Pearson Chi-Square, Likelihood Ratio Chi-Square, Continuity-Adjusted Chi-Square (for 2×2 tables), Mantel-Haenszel Chi-Square, the Phi Coefficient, the Contingency Coefficient, and Cramér's V. For each test statistic, PROC FREQ also displays the degrees of freedom (DF) and the probability value (Prob).

- If you specify the CHISQ option for 2×2 tables, PROC FREQ also displays Fisher's exact test. The test output includes the cell (1,1) frequency (F), the exact left-sided and right-sided probability values, the table probability (P), and the exact two-sided probability value. If you specify the POINT option in the EXACT statement, PROC FREQ displays the exact point probability for Fisher's exact test. If you specify the MIDP option in the EXACT statement, PROC FREQ displays the Mid p-Value for the test.

- If you specify the FISHER option in the TABLES statement (or, equivalently, the FISHER option in the EXACT statement), PROC FREQ displays Fisher's exact test for tables larger than 2×2. The test output includes the table probability (P) and the probability value. If you specify the POINT option in the EXACT statement, PROC FREQ displays the exact point probability for Fisher's exact test. If you specify the MIDP option in the EXACT statement, PROC FREQ displays the Mid p-Value for the test.

- If you specify the PCHI, LRCHI, or MHCHI option in the EXACT statement, PROC FREQ displays the corresponding exact test: Pearson Chi-Square, Likelihood Ratio Chi-Square, or Mantel-Haenszel Chi-Square, respectively. The test output includes the test statistic, the degrees of freedom (DF), and the asymptotic and exact probability values. If you also specify the POINT option in the EXACT statement, PROC FREQ displays the point probability for each exact test requested. If you specify the MIDP option in the EXACT statement, PROC FREQ displays the exact mid p-value for each test. If you specify the CHISQ option in the EXACT statement, PROC FREQ displays exact probability values for all three of these chi-square tests.

- If you specify the MC option in the EXACT statement, PROC FREQ displays Monte Carlo estimates for all exact p-values that you request in the EXACT statement. The Monte Carlo output includes the p-value Estimate, its Confidence Limits, the Number of Samples used to compute the Monte Carlo estimate, and the Initial Seed for random number generation.

- If you specify the MEASURES option, PROC FREQ displays the following statistics and their asymptotic standard errors (ASE) for each two-way table: Gamma, Kendall's Tau-b, Stuart's Tau-c, Somers' $D(C|R)$, Somers' $D(R|C)$, Pearson Correlation, Spearman Correlation, Lambda Asymmetric $(C|R)$, Lambda Asymmetric $(R|C)$, Lambda Symmetric, Uncertainty Coefficient $(C|R)$, Uncertainty Coefficient $(R|C)$, and Uncertainty Coefficient Symmetric. If you specify the CL option, PROC FREQ also displays confidence limits for these measures.

- If you specify the PLCORR option, PROC FREQ displays the polychoric correlation and its asymptotic standard error (ASE). For 2×2 tables, this statistic is known as the tetrachoric correlation (and is labeled as such in the displayed output). If you specify the CL option, PROC FREQ also displays confidence limits for the polychoric correlation. If you specify the PLCORR option in the TEST statement, PROC FREQ displays the polychoric correlation, asymptotic standard error (ASE), confidence limits, and the following: the standardized test statistic (Z), the corresponding one-sided and two-sided probability values, the likelihood ratio (LR) chi-square, and the probability value (Pr > ChiSq).

- If you specify the GAMMA, KENTB, STUTC, SMDCR, SMDRC, PCORR, or SCORR option in the TEST statement, PROC FREQ displays asymptotic tests for Gamma, Kendall's Tau-b, Stuart's Tau-c, Somers' $D(C|R)$, Somers' $D(R|C)$, the Pearson Correlation, or the Spearman Correlation, respectively. If you specify the MEASURES option in the TEST statement, PROC FREQ displays all these asymptotic tests. The test output includes the statistic, its asymptotic standard error (ASE), Confidence Limits, the ASE under the null hypothesis H0, the standardized test statistic (Z), and the one-sided and two-sided probability values.

- If you specify the KENTB, STUTC, SMDCR, SMDRC, PCORR, or SCORR option in the EXACT statement, PROC FREQ displays asymptotic and exact tests for the corresponding measure of association: Kendall's Tau-b, Stuart's Tau-c, Somers' $D(C|R)$, Somers' $D(R|C)$, the Pearson Correlation, or the Spearman Correlation, respectively. The test output includes the correlation, its asymptotic standard error (ASE), Confidence Limits, the ASE under the null hypothesis H0, the standardized test statistic (Z), and the asymptotic and exact one-sided and two-sided probability values. If you also specify the POINT option in the EXACT statement, PROC FREQ displays the point probability for each exact test requested. If you specify the MIDP option in the EXACT statement, PROC FREQ displays the exact Mid p-Value for each test.

- If you specify the SENSPEC option for 2×2 tables, PROC FREQ displays the "Sensitivity and Specificity" table. This table displays the Estimate, Standard Error, and Confidence Limits for the following statistics: Sensitivity, Specificity, Accuracy, Positive Predictive Value, and Negative Predictive Value.

- If you specify the RISKDIFF option for 2×2 tables, PROC FREQ displays the Column 1 and Column 2 Risk Estimates. For each column, PROC FREQ displays the Row 1 Risk, Row 2 Risk, Total Risk, and Risk Difference, together with their asymptotic standard errors (ASE) and Asymptotic Confidence Limits. PROC FREQ also displays Exact Confidence Limits for the Row 1 Risk, Row 2 Risk, and Total Risk. If you specify the RISKDIFF option in the EXACT statement, PROC FREQ provides unconditional Exact Confidence Limits for the Risk Difference. You can suppress this table by specifying the RISKDIFF(NORISKS) option.

- If you specify the RISKDIFF(CL=) option for 2×2 tables, PROC FREQ displays the "Confidence Limits for the Proportion (Risk) Difference" table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request (Agresti-Caffo, Exact, Hauck-Anderson, Miettinen-Nurminen, Newcombe, or Wald).

- If you specify the RISKDIFF(NONINF) option for 2×2 tables, PROC FREQ displays the "Noninferiority Analysis for the Risk Difference" table, which includes the Risk Difference, test ASE, standardized test statistic Z, probability value (Pr > Z), Noninferiority Limit, and (test-based) Confidence Limits.

- If you specify the RISKDIFF(SUP) option for 2×2 tables, PROC FREQ displays the "Superiority Analysis for the Risk Difference" table, which includes the Risk Difference, test ASE, standardized test statistic Z, probability value (Pr > Z), Superiority Limit, and (test-based) Confidence Limits.

- If you specify the RISKDIFF(EQUIV) option for 2×2 tables, PROC FREQ displays the "Equivalence Analysis for the Risk Difference" table, which includes the Risk Difference, test ASE, Equivalence Limits, and (test-based) Confidence Limits. PROC FREQ also displays the "Two One-Sided Tests (TOST)" table, which includes test statistics (Z) and P-Values for the Lower Margin and Upper Margin tests, along with the Overall P-Value.

- If you specify the RISKDIFF(EQUAL) option for 2×2 tables, PROC FREQ displays the "Risk Difference Test" table, which includes the Risk Difference, test ASE, standardized test statistic Z, One-sided probability value (Pr > Z or Pr < Z), and Two-sided probability value (Pr > |Z|).

- If you specify the MEASURES option or the RELRISK option for 2×2 tables, PROC FREQ displays the "Odds Ratio and Relative Risks" table, which includes the following statistics with their confidence limits: Odds Ratio, Relative Risk (Column 1), and Relative Risk (Column 2). If you specify the OR option in the EXACT statement, PROC FREQ also displays the "Exact Confidence Limits for the Odds

Ratio" table. If you specify the RELRISK option in the EXACT statement, PROC FREQ displays the "Exact Confidence Limits for the Relative Risk" table.

- If you specify the OR(CL=) option for 2×2 tables, PROC FREQ displays the "Confidence Limits for the Odds Ratio" table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request (Exact, Mid-p, Likelihood Ratio, Score, Wald, or Wald Modified).

- If you specify the RELRISK(CL=) option for 2×2 tables, PROC FREQ displays the "Confidence Limits for the Relative Risk" table, which includes the Lower and Upper Confidence Limits for each confidence limit Type that you request (Exact, Likelihood Ratio, Score, Wald, or Wald Modified).

- If you specify the RELRISK(NONINF) option, PROC FREQ displays the "Noninferiority Analysis for the Relative Risk" table, which includes the Relative Risk, standardized test statistic Z, probability value (Pr > Z), Noninferiority Limit, and Confidence Limits.

- If you specify the RELRISK(SUP) option, PROC FREQ displays the "Superiority Analysis for the Relative Risk" table, which includes the Relative Risk, standardized test statistic Z, probability value (Pr > Z), Superiority Limit, and Confidence Limits.

- If you specify the RELRISK(EQUIV) option, PROC FREQ displays the "Equivalence Analysis for the Relative Risk" table, which includes the Relative Risk, Equivalence Limits, and Confidence Limits. PROC FREQ also displays the "Two One-Sided Tests(TOST)" table, which includes test statistics (Z) and P-Values for the Lower Margin and Upper Margin tests, along with the Overall P-Value.

- If you specify the RELRISK(EQUAL) option, PROC FREQ displays the "Relative Risk Test" table, which includes the Relative Risk, standardized test statistic Z, One-sided probability value (Pr > Z or Pr < Z), and Two-sided probability value (Pr > |Z|).

- If you specify the TREND option, PROC FREQ displays the Cochran-Armitage Trend Test for tables that are $2 \times C$ or $R \times 2$. For this test, PROC FREQ gives the Statistic (Z) and the one-sided and two-sided probability values. If you specify the TREND option in the EXACT statement, PROC FREQ also displays the exact one-sided and two-sided probability values for this test. If you specify the POINT option with the TREND option in the EXACT statement, PROC FREQ displays the exact point probability for the test statistic. If you specify the MIDP option in the EXACT statement, PROC FREQ displays the exact Mid p-Value for the trend test.

- If you specify the JT option, PROC FREQ displays the Jonckheere-Terpstra Test, showing the Statistic (JT), the standardized test statistic (Z), and the one-sided and two-sided probability values. If you specify the JT option in the EXACT statement, PROC FREQ also displays the exact one-sided and two-sided probability values for this test. If you specify the POINT option with the JT option in the EXACT statement, PROC FREQ displays the exact point probability for the test statistic. If you specify the MIDP option in the EXACT statement, PROC FREQ displays the exact Mid p-Value for the Jonckheere-Terpstra test.

- If you specify the AGREE option for a 2×2 table, PROC FREQ displays the "McNemar's Test" table. This table includes the McNemar test statistic (chi-square), the degrees of freedom, and the p-value. If you specify the MCNEM option in the EXACT statement, this table also includes the exact p-value. If you specify the POINT option or the MIDP option in the EXACT statement, the "McNemar's Test" table includes the exact point probability or the exact mid p-value, respectively.

- If you specify the AGREE option for a square table of dimension greater than 2, PROC FREQ displays the "Symmetry Test" table. This table displays Bowker's symmetry test statistic (chi-square), the degrees of freedom, and the *p*-value. If you specify the SYMMETRY option in the EXACT statement, this table also includes the exact *p*-value. If you specify the POINT option or the MIDP option in the EXACT statement, the "Symmetry Test" table includes the exact point probability or the exact mid *p*-value, respectively.

- The AGREE option also produces the "Kappa Statistics" table, which displays the simple kappa coefficient. If the dimension of the two-way table is greater than 2, the "Kappa Statistics" table includes the weighted kappa coefficient. If you specify the AGREE(AC1) option or the AGREE(PABAK) option, this table includes the AC1 agreement coefficient or the prevalence-adjusted bias-adjusted kappa (PABAK), respectively. The "Kappa Statistics" table displays the standard error and confidence limits for each agreement statistic.

- If you specify the AGREE(KAPPADETAILS) option, PROC FREQ displays the "Kappa Details" table, which includes the observed agreement, the chance-expected agreement, the maximum kappa, and the B_N measure. For 2×2 tables, the "Kappa Details" table also includes the prevalence index and the bias index.

- If you specify the AGREE(WTKAPPADETAILS) or AGREE(KAPPADETAILS) option for a square table of dimension greater than 2, PROC FREQ produces the "Weighted Kappa Details" table, which displays the observed agreement and the chance-expected agreement components of the weighted kappa coefficient.

- If you specify the AGREE(PRINTKWTS) option for a square table of dimension greater than 2, PROC FREQ displays the matrix of agreement weights in the "Kappa Coefficient Weights" table.

- If you request a simple kappa coefficient test, PROC FREQ produces the "Kappa Test" table. You can request this test by specifying the KAPPA option in the TEST statement, the KAPPA option in the EXACT statement, or the AGREE(NULLKAPPA=) option in the TABLES statement. The "Kappa Test" table displays the kappa coefficient, null test value, standard error (when the null value is 0), standardized test statistic (Z), and one-sided and two-sided *p*-values.

  If you request an exact test (by specifying the KAPPA option in the EXACT statement), the "Kappa Test" table also includes the exact one-sided and two-sided *p*-values. If you specify the POINT option or the MIDP option in the EXACT statement, the "Kappa Test" table includes the point probability or the exact mid *p*-value, respectively.

- If you request a weighted kappa coefficient test for a square table of dimension greater than 2, PROC FREQ produces the "Weighted Kappa Test" table. You can request this test by specifying the WTKAPPA option in the TEST statement, the WTKAPPA option in the EXACT statement, or the AGREE(NULLWTKAPPA=) option in the TABLES statement. The "Weighted Kappa Test" table displays the weighted kappa coefficient, null test value, standard error (when the null value is 0), standardized test statistic (Z), and one-sided and two-sided *p*-values.

  If you request an exact test (by specifying the WTKAPPA option in the EXACT statement), the "Weighted Kappa Test" table also includes the exact one-sided and two-sided *p*-values. If you specify the POINT option or the MIDP option in the EXACT statement, the "Weighted Kappa Test" table includes the point probability or the exact mid *p*-value, respectively.

- If you specify the AGREE option for a multiway square table, PROC FREQ displays the "Overall Kappa Coefficients" table, which includes the overall simple kappa coefficient together with its standard

error and confidence limits. This table also includes the overall weighted kappa coefficient if the two-way table dimension is greater than 2.

- For multiway square tables, the AGREE option also produces the "Tests for Equal Kappa Coefficients" table. This table includes the chi-square statistic, degrees of freedom, and *p*-value for the test of equal simple kappa coefficients (over all strata). If the two-way table dimension is greater than 2, this table also includes the test for equal weighted kappa coefficients.

- For multiway tables in which each variable has two levels, both the AGREE option and the CQ option display the "Cochran's Q" table, which includes Cochran's Q statistic (to test for marginal homogeneity), the degrees of freedom, and the *p*-value.

- If you specify the COMMONRISKDIFF option for a multiway 2×2 table, PROC FREQ displays the "Confidence Limits for the Common Risk Difference" table, which includes the Method, Value of the common risk difference, Standard Error, and Confidence Limits for each confidence limit type that you request (Klingenberg, Mantel-Haenszel, Miettinen-Nurminen, Miettinen-Nurminen MH, Minimum Risk, Newcombe, Newcombe MR, or Summary Score).

- If you specify the COMMONRISKDIFF(TEST) option for a multiway 2×2 table, PROC FREQ displays the "Common Risk Difference Tests" table, which includes Method, Risk Difference, Z, and Pr > |Z| for each test that you request (Mantel-Haenszel, Minimum Risk, or Summary Score).

- If you specify the COMMONRISKDIFF(PRINTWTS) option for a multiway 2×2 table, PROC FREQ displays the "Stratum Weights" table, which includes the following information for each stratum (2×2 table): Stratum index, variable levels, Risk Difference, Frequency, Fraction, and the stratum weights that you request (Mantel-Haenszel, Miettinen-Nurminen, Minimum Risk, or Summary Score Weights).

- If you specify the CMH option, PROC FREQ displays Cochran-Mantel-Haenszel Statistics for the following three alternative hypotheses: Nonzero Correlation, Row Mean Scores Differ (ANOVA Statistic), and General Association. For each of these statistics, PROC FREQ gives the degrees of freedom (DF) and the probability value (Prob). If you specify the MANTELFLEISS option, PROC FREQ displays the Mantel-Fleiss Criterion for 2×2 tables. For 2×2 tables, PROC FREQ also displays Estimates of the Common Relative Risk for Case-Control and Cohort studies, together with their confidence limits. These include both Mantel-Haenszel and Logit stratum-adjusted estimates of the common Odds Ratio, Column 1 Relative Risk, and Column 2 Relative Risk. Also for 2×2 tables, PROC FREQ displays the Breslow-Day Test for Homogeneity of the Odds Ratios. For this test, PROC FREQ gives the Chi-Square, the degrees of freedom (DF), and the probability value (Pr > ChiSq).

- If you specify the CMH(QOR) option for a stratified 2×2 table, PROC FREQ displays the "Q Test for Homogeneity of Odds Ratios" table, which includes the Chi-Square, the degrees of freedom (DF), and the probability value (Pr > ChiSq).

- If you specify the CMH(I2) option for a stratified 2×2 table, PROC FREQ displays the "I-Square Measure of Heterogeneity" table, which includes the I-Square, the degrees of freedom (DF), and the Confidence Limits.

- If you specify the CMH option in the TABLES statement and also specify the COMOR option in the EXACT statement for a multiway 2×2 table, PROC FREQ displays exact confidence limits for the Common Odds Ratio. PROC FREQ also displays the Exact Test of H0: Common Odds Ratio = 1. The test output includes the Cell (1,1) Sum (S), Mean of S Under H0, One-sided Pr <= S, and Point Pr = S. PROC FREQ also provides exact two-sided probability values for the test, computed according

to the following three methods: 2 * One-sided, Sum of probabilities <= Point probability, and Pr >= |S - Mean|. If you specify the MIDP option in the EXACT statement, PROC FREQ provides the exact Mid *p*-Value for the common odds ratio test.

- If you specify the CMH option in the TABLES statement and also specify the EQOR option in the EXACT statement for a multiway 2×2 table, PROC FREQ computes Zelen's exact test for equal odds ratios. PROC FREQ displays Zelen's test along with the asymptotic Breslow-Day test produced by the CMH option. PROC FREQ displays the test statistic, Zelen's Exact Test (P), and the probability value, Exact Pr <= P.

- If you specify the GAILSIMON option in the TABLES statement for a multiway 2×2 tables, PROC FREQ displays the Gail-Simon test for qualitative interactions. The display include the following statistics and their *p*-values: Q+ (Positive Risk Differences), Q- (Negative Risk Differences), and Q (Two-Sided).

ODS Table Names

PROC FREQ assigns a name to each table that it creates. You can use these names to refer to tables when you use the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 47.22 lists the ODS table names together with their descriptions and the options that are required to produce the tables.

Table 47.22 ODS Tables Produced by PROC FREQ

ODS Table Name	Description	Statement	Option
BarnardsTest	Barnard's exact test	EXACT	BARNARD
Binomial	Binomial proportion	TABLES	BINOMIAL
BinomialCLs	Binomial confidence limits	TABLES	BINOMIAL(CL=)
BinomialEquiv	Binomial equivalence analysis	TABLES	BINOMIAL(EQUIV)
BinomialEquivLimits	Binomial equivalence limits	TABLES	BINOMIAL(EQUIV)
BinomialEquivTest	Binomial equivalence test	TABLES	BINOMIAL(EQUIV)
BinomialNoninf	Binomial noninferiority test	TABLES	BINOMIAL(NONINF)
BinomialSup	Binomial superiority test	TABLES	BINOMIAL(SUP)
BinomialTest	Binomial proportion test	TABLES	BINOMIAL
BnMeasure	Agreement measures	TABLES	PLOTS=AGREEPLOT(STATS)
BreslowDayTest	Breslow-Day test	TABLES	CMH ($h \times 2 \times 2$ table)
ChiSq	Chi-square tests	TABLES	CHISQ
CMH	Cochran-Mantel-Haenszel statistics	TABLES	CMH
CochransQ	Cochran's Q	TABLES	AGREE or CQ ($2 \times 2 \times \cdots \times 2$ table)
ColScores	Column scores	TABLES	SCOROUT
CommonOdds-RatioCl	Exact confidence limits for the common odds ratio	EXACT	COMOR ($h \times 2 \times 2$ table)
CommonOdds-	Common odds ratio exact test	EXACT	COMOR

Table 47.22 *continued*

ODS Table Name	Description	Statement	Option
RatioTest			($h \times 2 \times 2$ table)
CommonPdiff	Common risk difference confidence limits	TABLES	COMMONRISKDIFF ($h \times 2 \times 2$ table)
CommonPdiffTests	Common risk difference tests	TABLES	COMMONRISKDIFF(TESTS) ($h \times 2 \times 2$ table)
CommonRelRisks	Common relative risks	TABLES	CMH ($h \times 2 \times 2$ table)
CrossList	Crosstabulation table in column format	TABLES	CROSSLIST (n-way table, $n > 1$)
CrossTabFreqs	Crosstabulation table	TABLES	(n-way table, $n > 1$)
EqualKappaTest	Test for equal simple kappas	TABLES	AGREE ($h \times 2 \times 2$ table)
EqualKappaTests	Tests for equal kappas	TABLES	AGREE ($h \times r \times r$ table, $r > 2$)
EqualOddsRatios	Tests for equal odds ratios	EXACT	EQOR ($h \times 2 \times 2$ table)
FishersExact	Fisher's exact test	EXACT or TABLES or TABLES	FISHER FISHER or EXACT CHISQ (2×2 table)
FishersExactMC	Monte Carlo estimates for Fisher's exact test	EXACT	FISHER / MC
GailSimonTest	Gail-Simon test	TABLES	GAILSIMON ($h \times 2 \times 2$ table)
Gamma	Gamma	TEST	GAMMA
GammaTest	Gamma test	TEST	GAMMA
I2	I-square measure	TABLES	CMH(I2) ($h \times 2 \times 2$ table)
JTTest	Jonckheere-Terpstra test	TABLES	JT
JTTestMC	Monte Carlo estimates for Jonckheere-Terpstra exact test	EXACT	JT / MC
KappaDetails	Kappa details	TABLES	AGREE(KAPPADETAILS)
KappaMC	Monte Carlo exact test for simple kappa coefficient	EXACT	KAPPA / MC
KappaStatistics	Kappa statistics	TABLES	AGREE
KappaTest	Simple kappa test	TEST or EXACT or TABLES	KAPPA KAPPA AGREE(NULLKAPPA=)
KappaWeights	Kappa weights	TABLES	AGREE(PRINTKWTS)
List	List format multiway table	TABLES	LIST
LRChiSq	Likelihood ratio chi-square exact test	EXACT	LRCHI
LRChiSqMC	Monte Carlo exact test for likelihood ratio chi-square	EXACT	LRCHI / MC
MantelFleiss	Mantel-Fleiss criterion	TABLES	CMH(MANTELFLEISS) ($h \times 2 \times 2$ table)

Table 47.22 *continued*

ODS Table Name	Description	Statement	Option
McNemarsTest	McNemar's test	TABLES	AGREE (2×2 table)
Measures	Measures of association	TABLES	MEASURES
MHChiSq	Mantel-Haenszel chi-square exact test	EXACT	MHCHI
MHChiSqMC	Monte Carlo exact test for Mantel-Haenszel chi-square	EXACT	MHCHI / MC
NLevels	Number of variable levels	PROC	NLEVELS
OddsRatioCLs	Odds ratio confidence limits	TABLES	OR(CL=) (2×2 table)
OddsRatioExactCL	Exact confidence limits for the odds ratio	EXACT	OR (2×2 table)
OneWayChiSq	One-way chi-square test	TABLES	CHISQ (one-way table)
OneWayChiSqMC	Monte Carlo exact test for one-way chi-square	EXACT	CHISQ / MC (one-way table)
OneWayFreqs	One-way frequencies	PROC or TABLES	(no TABLES stmt) (one-way table)
OneWayLRChiSq	One-way likelihood ratio chi-square test	TABLES	CHISQ(LRCHI) (one-way table)
OverallKappa	Overall simple kappa	TABLES	AGREE ($h \times 2 \times 2$ table)
OverallKappas	Overall kappa coefficients	TABLES	AGREE ($h \times r \times r$ table, $r > 2$)
PdiffCLs	Risk difference confidence limits	TABLES	RISKDIFF(CL=) (2×2 table)
PdiffEquiv	Equivalence analysis for the risk difference	TABLES	RISKDIFF(EQUIV) (2×2 table)
PdiffEquivTest	Equivalence test for the risk difference	TABLES	RISKDIFF(EQUIV) (2×2 table)
PdiffNoninf	Noninferiority test for the risk difference	TABLES	RISKDIFF(NONINF) (2×2 table)
PdiffSup	Superiority test for the risk difference	TABLES	RISKDIFF(SUP) (2×2 table)
PdiffTest	Risk difference test	TABLES	RISKDIFF(EQUAL) (2×2 table)
PearsonChiSq	Pearson chi-square exact test	EXACT	PCHI
PearsonChiSqMC	Monte Carlo exact test for Pearson chi-square	EXACT	PCHI / MC
PearsonCorr	Pearson correlation	TEST or EXACT	PCORR PCORR
PearsonCorrMC	Monte Carlo exact test for Pearson correlation	EXACT	PCORR / MC
PearsonCorrTest	Pearson correlation test	TEST or EXACT	PCORR PCORR
PlCorr	Polychoric correlation	TEST	PLCORR

Table 47.22 *continued*

ODS Table Name	Description	Statement	Option
PlCorrTest	Polychoric correlation test	TEST	PLCORR
QOR	Q test for odds ratios	TABLES	CMH(QOR) ($h \times 2 \times 2$ table)
RelativeRiskCLs	Relative risk confidence limits	TABLES	RELRISK(CL=) (2×2 table)
RelativeRisks	Relative risk estimates	TABLES	RELRISK or MEASURES (2×2 table)
RelRisk1ExactCL	Exact confidence limits for column 1 relative risk	EXACT	RELRISK (2×2 table)
RelRisk2ExactCL	Exact confidence limits for column 2 relative risk	EXACT	RELRISK (2×2 table)
RelriskEquiv	Equivalence analysis for the relative risk	TABLES	RELRISK(EQUIV) (2×2 table)
RelriskEquivTest	Equivalence test for the relative risk	TABLES	RELRISK(EQUIV) (2×2 table)
RelriskNoninf	Noninferiority test for the relative risk	TABLES	RELRISK(NONINF) (2×2 table)
RelriskSup	Superiority test for the relative risk	TABLES	RELRISK(SUP) (2×2 table)
RelriskTest	Relative risk test	TABLES	RELRISK(EQUAL) (2×2 table)
RiskDiffCol1	Column 1 risk estimates	TABLES	RISKDIFF (2×2 table)
RiskDiffCol2	Column 2 risk estimates	TABLES	RISKDIFF (2×2 table)
RowScores	Row scores	TABLES	SCOROUT
SenSpec	Sensitivity and specificity	TABLES	SENSPEC (2×2 table)
SomersDCR	Somers' $D(C\|R)$	TEST or EXACT	SMDCR SMDCR
SomersDCRMC	Monte Carlo exact test for Somers' $D(C\|R)$	EXACT	SMDCR / MC
SomersDCRTest	Somers' $D(C\|R)$ test	TEST or EXACT	SMDCR SMDCR
SomersDRC	Somers' $D(R\|C)$	TEST or EXACT	SMDRC SMDRC
SomersDRCMC	Monte Carlo exact test for Somers' $D(R\|C)$	EXACT	SMDRC / MC
SomersDRCTest	Somers' $D(R\|C)$ test	TEST or EXACT	SMDRC SMDRC
SpearmanCorr	Spearman correlation	TEST or EXACT	SCORR SCORR
SpearmanCorrMC	Monte Carlo exact test for Spearman correlation	EXACT	SCORR / MC
SpearmanCorrTest	Spearman correlation test	TEST or EXACT	SCORR SCORR

Table 47.22 *continued*

ODS Table Name	Description	Statement	Option
StratumWeights	Stratum risk differences and weights	TABLES	COMMONRISKDIFF ($h \times 2 \times 2$ table)
SymmetryMC	Monte Carlo exact symmetry test	EXACT	SYMMETRY / MC
SymmetryTest	Symmetry test	TABLES	AGREE
TauB	Kendall's tau-*b*	TEST	KENTB
		or EXACT	KENTB
TauBMC	Monte Carlo exact test for Kendall's tau-*b*	EXACT	KENTB / MC
TauBTest	Kendall's tau-*b* test	TEST	KENTB
		or EXACT	KENTB
TauC	Stuart's tau-*c*	TEST	STUTC
		or EXACT	STUTC
TauCMC	Monte Carlo exact test for Stuart's tau-*c*	EXACT	STUTC / MC
TauCTest	Stuart's tau-*c* test	TEST	STUTC
		or EXACT	STUTC
TrendTest	Cochran-Armitage trend test	TABLES	TREND
TrendTestMC	Monte Carlo exact test for trend	EXACT	TREND / MC
WtKappaDetails	Weighted kappa details	TABLES	AGREE(WTKAPPADETAILS)
WtKappaMC	Monte Carlo exact test for weighted kappa coefficient	EXACT	WTKAPPA / MC
WtKappaTest	Weighted kappa test	TEST	WTKAPPA
		or EXACT	WTKAPPA
		or TABLES	AGREE(NULLWTKAPPA=)

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, you can request specific plots by specifying PLOTS= option in the TABLES statement. To produce a frequency plot or cumulative frequency plot, you must specify the FREQPLOT or CUMFREQPLOT *plot-request*, respectively, in the PLOTS= option. To produce a mosaic plot, you must

specify the MOSAICPLOT *plot-request* in the PLOTS= option. You can also produce frequency, cumulative frequency, and mosaic plots by specifying the PLOTS=ALL option. By default, PROC FREQ produces all other plots that are associated with the analyses that you request in the TABLES statement. You can suppress the default plots and request specific plots by using the PLOTS(ONLY)= option. For more information, see the description of the PLOTS= option.

PROC FREQ assigns a name to each graph that it creates by using ODS Graphics. You can use these names to refer to the graphs. Table 47.23 lists the names of the graphs that PROC FREQ generates together with their descriptions, their PLOTS= options (*plot-requests*), and the TABLES statement options that are required to produce the graphs.

Table 47.23 Graphs Produced by PROC FREQ

ODS Graph Name	Description	PLOTS= Option	TABLES Statement Option
AgreePlot	Agreement plot	AGREEPLOT	AGREE ($r \times r$ table)
CumFreqPlot	Cumulative frequency plot	CUMFREQPLOT	One-way table request
DeviationPlot	Deviation plot	DEVIATIONPLOT	CHISQ (one-way table)
FreqPlot	Frequency plot	FREQPLOT	Any table request
KappaPlot	Kappa plot	KAPPAPLOT	AGREE ($h \times r \times r$ table)
MosaicPlot	Mosaic plot	MOSAICPLOT	Two-way or multiway table request
ORPlot	Odds ratio plot	ODDSRATIOPLOT	MEASURES, OR, or RELRISK ($h \times 2 \times 2$ table)
RelRiskPlot	Relative risk plot	RELRISKPLOT	MEASURES or RELRISK ($h \times 2 \times 2$ table)
RiskDiffPlot	Risk difference plot	RISKDIFFPLOT	RISKDIFF ($h \times 2 \times 2$ table)
WtKappaPlot	Weighted kappa plot	WTKAPPAPLOT	AGREE ($h \times r \times r$ table, $r > 2$)

Examples: FREQ Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 47.1: Output Data Set of Frequencies

(View the complete code for this example (freqex1.sas) in the example repository.)

The eye and hair color of children from two different regions of Europe are recorded in the data set Color. Instead of recording one observation per child, the data are recorded as cell counts, where the variable Count contains the number of children exhibiting each of the 15 eye and hair color combinations. The data set does not include missing combinations.

The following DATA step statements create the SAS data set Color:

```
data Color;
   input Region Eyes $ Hair $ Count @@;
   label Eyes  ='Eye Color'
         Hair  ='Hair Color'
         Region='Geographic Region';
   datalines;
1 blue  fair   23  1 blue  red     7  1 blue  medium 24
1 blue  dark   11  1 green fair   19  1 green red     7
1 green medium 18  1 green dark   14  1 brown fair   34
1 brown red     5  1 brown medium 41  1 brown dark   40
1 brown black   3  2 blue  fair   46  2 blue  red    21
2 blue  medium 44  2 blue  dark   40  2 blue  black   6
2 green fair   50  2 green red    31  2 green medium 37
2 green dark   23  2 brown fair   56  2 brown red    42
2 brown medium 53  2 brown dark   54  2 brown black  13
;
```

The following PROC FREQ statements read the Color data set and create an output data set that contains the frequencies, percentages, and expected cell frequencies of the two-way table of Eyes by Hair. The TABLES statement requests three tables: a frequency table for Eyes, a frequency table for Hair, and a crosstabulation table for Eyes by Hair. The OUT= option creates the FreqCount data set, which contains the crosstabulation table frequencies. The OUTEXPECT option outputs the expected table cell frequencies to FreqCount, and the SPARSE option includes cell frequencies of 0 in the output data set. The WEIGHT statement specifies that the variable Count contains the observation weights. These statements create Output 47.1.1 through Output 47.1.3.

```
proc freq data=Color;
   tables Eyes Hair Eyes*Hair / out=FreqCount outexpect sparse;
   weight Count;
   title 'Eye and Hair Color of European Children';
run;

proc print data=FreqCount noobs;
   title2 'Output Data Set from PROC FREQ';
run;
```

Output 47.1.1 displays the two frequency tables produced by PROC FREQ: one showing the distribution of eye color, and one showing the distribution of hair color. By default, PROC FREQ lists the variables values in alphabetical order. The ‘Eyes*Hair’ specification produces a crosstabulation table, shown in Output 47.1.2, with eye color defining the table rows and hair color defining the table columns. A cell frequency of 0 for green eyes and black hair indicates that this eye and hair color combination does not occur in the data.

The output data set FreqCount (Output 47.1.3) contains frequency counts and percentages for the last table requested in the TABLES statement, Eyes by Hair. Because the SPARSE option is specified, the data set includes the observation that has a frequency of 0. The variable Expected contains the expected frequencies, as requested by the OUTEXPECT option.

Output 47.1.1 Frequency Tables

Eye and Hair Color of European Children

The FREQ Procedure

Eye Color

Eyes	Frequency	Percent	Cumulative Frequency	Cumulative Percent
blue	222	29.13	222	29.13
brown	341	44.75	563	73.88
green	199	26.12	762	100.00

Hair Color

Hair	Frequency	Percent	Cumulative Frequency	Cumulative Percent
black	22	2.89	22	2.89
dark	182	23.88	204	26.77
fair	228	29.92	432	56.69
medium	217	28.48	649	85.17
red	113	14.83	762	100.00

Output 47.1.2 Crosstabulation Table

Table of Eyes by Hair

Frequency Percent Row Pct Col Pct	Hair(Hair Color)					
Eyes(Eye Color)	**black**	**dark**	**fair**	**medium**	**red**	**Total**
blue	6 0.79 2.70 27.27	51 6.69 22.97 28.02	69 9.06 31.08 30.26	68 8.92 30.63 31.34	28 3.67 12.61 24.78	222 29.13
brown	16 2.10 4.69 72.73	94 12.34 27.57 51.65	90 11.81 26.39 39.47	94 12.34 27.57 43.32	47 6.17 13.78 41.59	341 44.75
green	0 0.00 0.00 0.00	37 4.86 18.59 20.33	69 9.06 34.67 30.26	55 7.22 27.64 25.35	38 4.99 19.10 33.63	199 26.12
Total	22 2.89	182 23.88	228 29.92	217 28.48	113 14.83	762 100.00

Output 47.1.3 Output Data Set of Frequencies

Eye and Hair Color of European Children
Output Data Set from PROC FREQ

Eyes	Hair	COUNT	EXPECTED	PERCENT
blue	black	6	6.409	0.7874
blue	dark	51	53.024	6.6929
blue	fair	69	66.425	9.0551
blue	medium	68	63.220	8.9239
blue	red	28	32.921	3.6745
brown	black	16	9.845	2.0997
brown	dark	94	81.446	12.3360
brown	fair	90	102.031	11.8110
brown	medium	94	97.109	12.3360
brown	red	47	50.568	6.1680
green	black	0	5.745	0.0000
green	dark	37	47.530	4.8556
green	fair	69	59.543	9.0551
green	medium	55	56.671	7.2178
green	red	38	29.510	4.9869

Example 47.2: Frequency Dot Plots

(View the complete code for this example (freqex2.sas) in the example repository.)

This example produces frequency dot plots for the children's eye and hair color data from Example 47.1.

PROC FREQ produces plots by using ODS Graphics to create graphs as part of the procedure output. Frequency plots are available for any frequency or crosstabulation table request. You can display frequency plots as bar charts or dot plots. You can use *plot-options* to specify the orientation (vertical or horizontal), scale, and layout of the plots.

The following PROC FREQ statements request frequency tables and dot plots. The first TABLES statement requests a one-way frequency table of Hair and a crosstabulation table of Eyes by Hair. The PLOTS= option requests frequency plots for the tables, and the TYPE=DOTPLOT *plot-option* specifies dot plots. By default, frequency plots are produced as bar charts. ODS Graphics must be enabled before producing plots.

The second TABLES statement requests a crosstabulation table of Region by Hair and a frequency dot plot for this table. The SCALE=PERCENT *plot-option* plots percentages instead of frequency counts. SCALE=LOG and SCALE=SQRT *plot-options* are also available to plot log frequencies and square roots of frequencies, respectively.

The ORDER=FREQ option in the PROC FREQ statement orders the variable levels by frequency. This order applies to the frequency and crosstabulation table displays and also to the corresponding frequency plots.

```
ods graphics on;
proc freq data=Color order=freq;
   tables Hair Hair*Eyes / plots=freqplot(type=dotplot);
   tables Hair*Region / plots=freqplot(type=dotplot scale=percent);
   weight Count;
```

```
   title 'Eye and Hair Color of European Children';
run;
ods graphics off;
```

Output 47.2.1, Output 47.2.2, and Output 47.2.3 display the dot plots produced by PROC FREQ. By default, the orientation of dot plots is horizontal, which places the variable levels on the Y axis. You can specify the ORIENT=VERTICAL *plot-option* to request a vertical orientation. For two-way plots, you can use the TWOWAY= *plot-option* to specify the plot layout. The default layout (shown in Output 47.2.2 and Output 47.2.3) is GROUPVERTICAL. Two-way layouts STACKED and GROUPHORIZONTAL are also available.

Output 47.2.1 One-Way Frequency Dot Plot

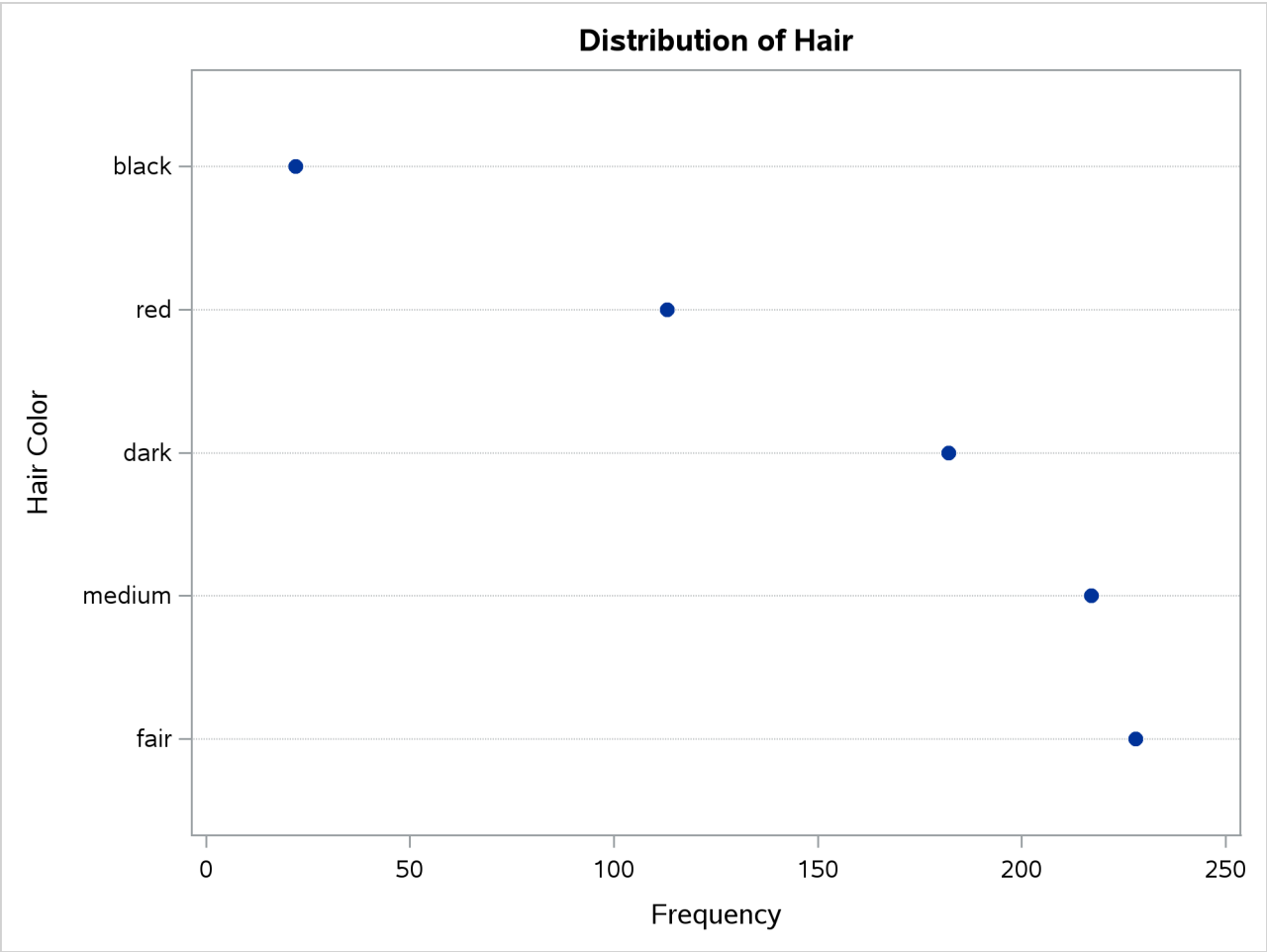

Output 47.2.2 Two-Way Frequency Dot Plot

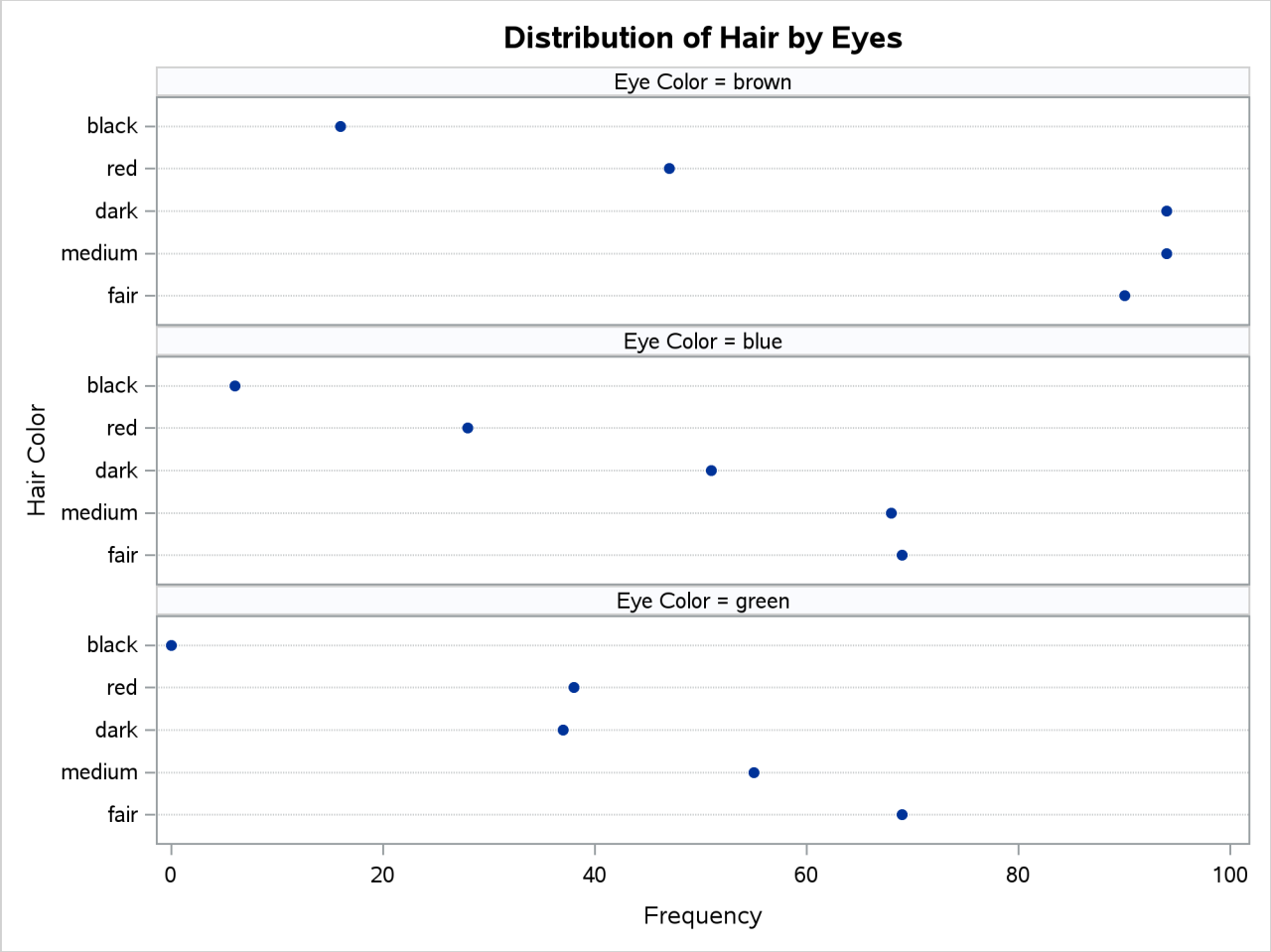

Output 47.2.3 Two-Way Percent Dot Plot

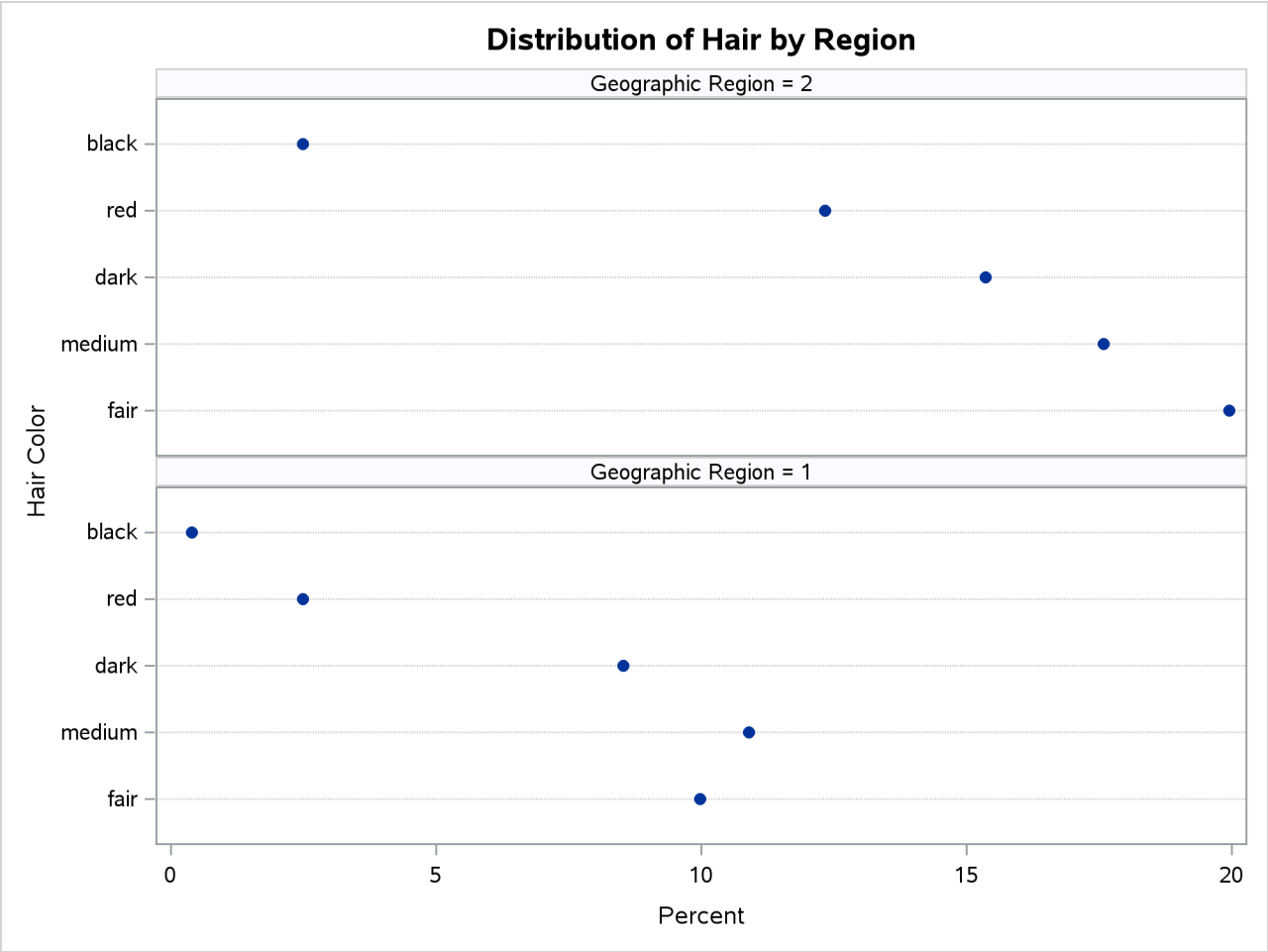

Example 47.3: Chi-Square Goodness-of-Fit Tests

(View the complete code for this example (freqex3.sas) in the example repository.)

This example examines whether the children's hair color (from Example 47.1) has a specified multinomial distribution for the two geographical regions. The hypothesized distribution of hair color is 30% fair, 12% red, 30% medium, 25% dark, and 3% black.

In order to test the hypothesis for each region, the data are first sorted by Region. Then the FREQ procedure uses a BY statement to produce a separate table for each BY group (Region). The option ORDER=DATA orders the variable values (hair color) in the frequency table by their order in the input data set. The TABLES statement requests a frequency table for hair color, and the option NOCUM suppresses the display of the cumulative frequencies and percentages.

The CHISQ option requests a chi-square goodness-of-fit test for the frequency table of Hair. The TESTP= option specifies the hypothesized (or test) percentages for the chi-square test; the number of percentages listed equals the number of table levels, and the percentages sum to 100%. The TESTP= percentages are listed in the same order as the corresponding variable levels appear in frequency table.

The PLOTS= option requests a deviation plot, which is associated with the CHISQ option and displays the relative deviations from the test frequencies. The TYPE=DOTPLOT *plot-option* requests a dot plot instead of the default type, which is a bar chart. ODS Graphics must be enabled before producing plots. These statements produce Output 47.3.1 through Output 47.3.4.

```
proc sort data=Color;
   by Region;
run;

ods graphics on;
proc freq data=Color order=data;
   tables Hair / nocum chisq testp=(30 12 30 25 3)
                 plots(only)=deviationplot(type=dotplot);
   weight Count;
   by Region;
   title 'Hair Color of European Children';
run;
ods graphics off;
```

Output 47.3.1 Frequency Table and Chi-Square Test for Region 1

Hair Color of European Children

The FREQ Procedure

Geographic Region=1

Hair Color			
Hair	**Frequency**	**Percent**	**Test Percent**
fair	76	30.89	30.00
red	19	7.72	12.00
medium	83	33.74	30.00
dark	65	26.42	25.00
black	3	1.22	3.00

Geographic Region=1

Chi-Square Test for Specified Proportions	
Chi-Square	7.7602
DF	4
Pr > ChiSq	0.1008

Output 47.3.1 shows the frequency table and chi-square test for Region 1. The frequency table lists the variable values (hair color) in the order in which they appear in the data set. The "Test Percent" column lists the hypothesized percentages for the chi-square test. Always check that you have ordered the TESTP= percentages to correctly match the order of the variable levels.

Output 47.3.2 shows the deviation plot for Region 1, which displays the relative deviations from the hypothesized values. The relative deviation for a level is the difference between the observed and hypothesized (test) percentage divided by the test percentage. You can suppress the chi-square *p*-value that is displayed by default in the deviation plot by specifying the NOSTATS *plot-option*.

Output 47.3.2 Deviation Plot for Region 1

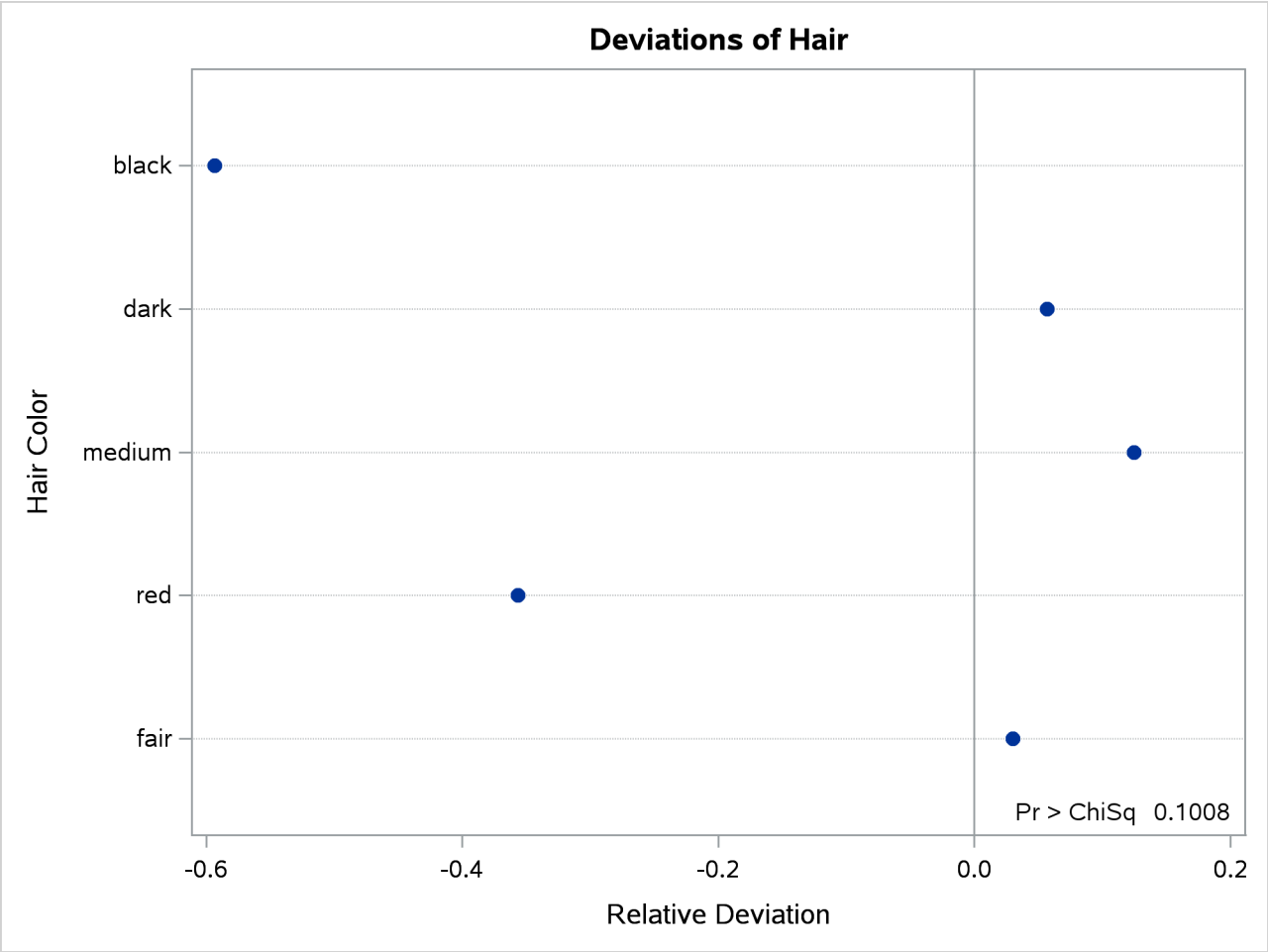

Output 47.3.3 and Output 47.3.4 show the results for Region 2. PROC FREQ computes a chi-square statistic for each region. The chi-square statistic is significant at the 0.05 level for Region 2 (p=0.0003) but not for Region 1. This indicates a significant departure from the hypothesized percentages in Region 2.

Output 47.3.3 Frequency Table and Chi-Square Test for Region 2

Hair Color of European Children

The FREQ Procedure

Geographic Region=2

Hair Color			
Hair	**Frequency**	**Percent**	**Test Percent**
fair	152	29.46	30.00
red	94	18.22	12.00
medium	134	25.97	30.00
dark	117	22.67	25.00
black	19	3.68	3.00

Output 47.3.3 *continued*

Geographic Region=2

Chi-Square Test for Specified Proportions	
Chi-Square	21.3824
DF	4
Pr > ChiSq	0.0003

Output 47.3.4 Deviation Plot for Region 2

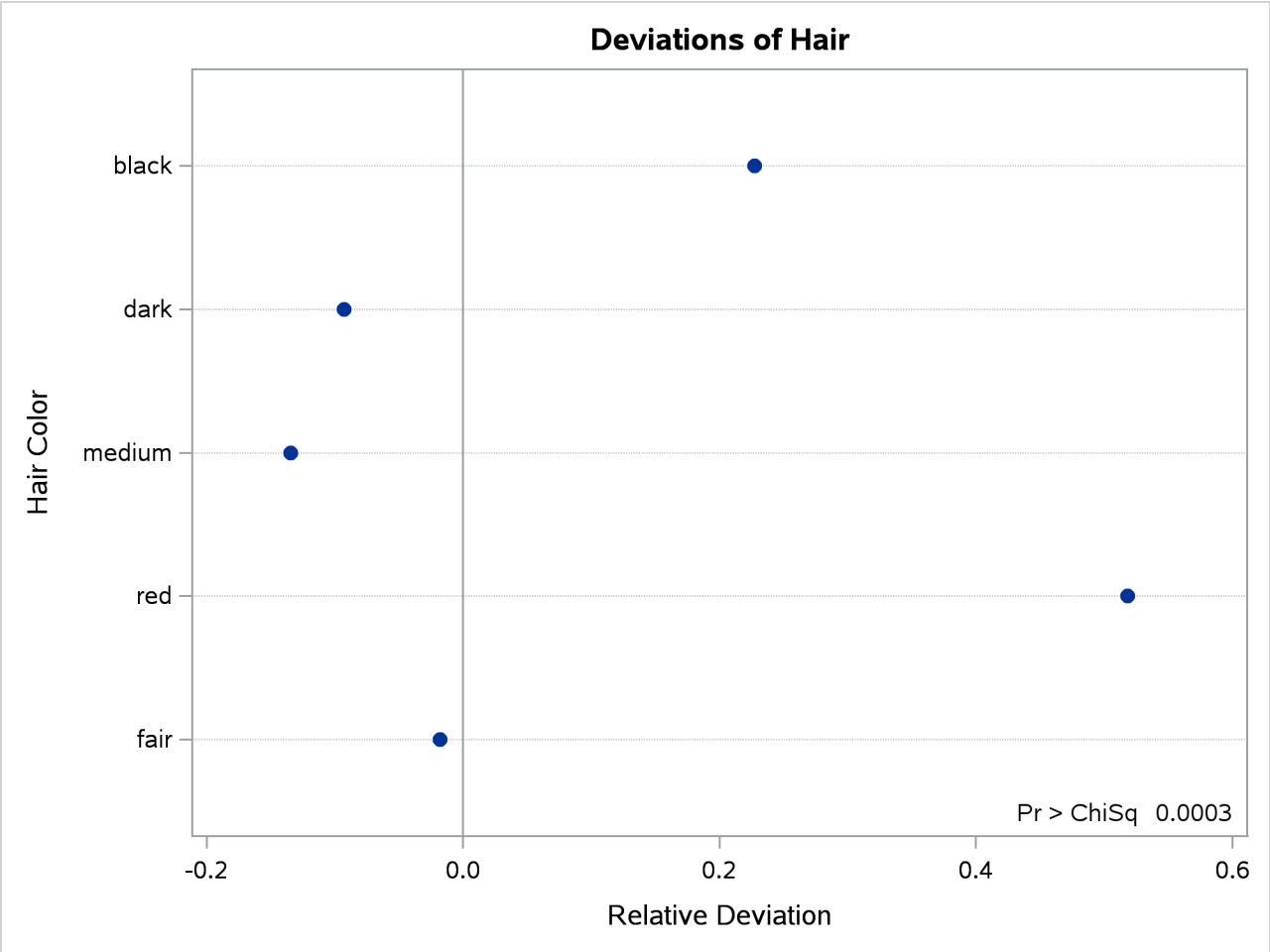

Example 47.4: Binomial Proportions

(View the complete code for this example (freqex4.sas) in the example repository.)

In this example, PROC FREQ computes binomial proportions, confidence limits, and tests. The example uses the eye and hair color data from Example 47.1. By default, PROC FREQ computes the binomial proportion as the proportion of observations in the first level of the one-way table. You can designate a different level by using the LEVEL= *binomial-option*.

The following PROC FREQ statements compute the proportion of children with brown eyes (from the data set in Example 47.1) and test the null hypothesis that the population proportion equals 50%. These statements also compute an equivalence test for the proportion of children with fair hair.

The first TABLES statement requests a one-way frequency table for the variable Eyes. The BINOMIAL option requests the binomial proportion, confidence limits, and test. PROC FREQ computes the proportion with Eyes = 'brown', which is the first level displayed in the table. The AC, WILSON, and EXACT *binomial-options* request the following confidence limits types: Agresti-Coull, Wilson (score), and exact (Clopper-Pearson). By default, PROC FREQ provides Wald and exact (Clopper-Pearson) confidence limits for the binomial proportion. The BINOMIAL option also produces an asymptotic Wald test that the proportion is 0.5. You can specify a different test proportion in the P= *binomial-option*. The ALPHA=0.1 option specifies that $\alpha = 10\%$, which produces 90% confidence limits.

The second TABLES statement requests a one-way frequency table for the variable Hair. The BINOMIAL option requests the proportion for the first level, Hair = 'fair'. The EQUIV *binomial-option* requests an equivalence test for the binomial proportion. The P=.28 option specifies 0.28 as the null hypothesis proportion, and the MARGIN=.1 option specifies 0.1 as the equivalence test margin.

```
proc freq data=Color order=freq;
   tables Eyes / binomial(ac wilson exact) alpha=.1;
   tables Hair / binomial(equiv p=.28 margin=.1);
   weight Count;
   title 'Hair and Eye Color of European Children';
run;
```

Output 47.4.1 displays the results for eye color, and Output 47.4.2 displays the results for hair color.

Output 47.4.1 Binomial Proportion for Eye Color

Hair and Eye Color of European Children

The FREQ Procedure

Eye Color				
Eyes	Frequency	Percent	Cumulative Frequency	Cumulative Percent
brown	341	44.75	341	44.75
blue	222	29.13	563	73.88
green	199	26.12	762	100.00

Output 47.4.1 *continued*

Binomial Proportion	
Eyes = brown	
Proportion	0.4475
ASE	0.0180

Confidence Limits for the Binomial Proportion		
Proportion = 0.4475		
Type	90% Confidence Limits	
Agresti-Coull	0.4181	0.4773
Clopper-Pearson (Exact)	0.4174	0.4779
Wilson	0.4181	0.4773

Test of H0: Proportion = 0.5	
ASE under H0	0.0181
Z	-2.8981
One-sided Pr < Z	0.0019
Two-sided Pr > |Z|	0.0038

The frequency table in Output 47.4.1 displays the values of Eyes in order of descending frequency count. PROC FREQ computes the proportion of children in the first level displayed in the frequency table, Eyes = 'brown'. Output 47.4.1 displays the binomial proportion confidence limits and test. The confidence limits are 90% confidence limits. If you do not specify the ALPHA= option, PROC FREQ computes 95% confidence limits by default. Because the value of Z is less than 0, PROC FREQ displays the a left-sided p-value (0.0019). This small p-value supports the alternative hypothesis that the true value of the proportion of children with brown eyes is less than 50%.

Output 47.4.2 displays the equivalence test results produced by the second TABLES statement. The null hypothesis proportion is 0.28 and the equivalence margins are –0.1 and 0.1, which yield equivalence limits of 0.18 and 0.38. PROC FREQ provides two one-sided tests (TOST) for equivalence. The small p-value indicates rejection of the null hypothesis in favor of the alternative that the proportion is equivalent to the null value.

Output 47.4.2 Binomial Proportion for Hair Color

Hair Color				
Hair	Frequency	Percent	Cumulative Frequency	Cumulative Percent
fair	228	29.92	228	29.92
medium	217	28.48	445	58.40
dark	182	23.88	627	82.28
red	113	14.83	740	97.11
black	22	2.89	762	100.00

Output 47.4.2 *continued*

Equivalence Analysis	
H0: P - p0 <= Lower Margin or >= Upper Margin	
Ha: Lower Margin < P - p0 < Upper Margin	
p0 = 0.28 Lower Margin = -0.1 Upper Margin = 0.1	
Proportion	ASE (Sample)
0.2992	0.0166

Two One-Sided Tests (TOST)			
Test	Z		P-Value
Lower Margin	7.1865	Pr > Z	<.0001
Upper Margin	-4.8701	Pr < Z	<.0001
Overall			<.0001

Equivalence Limits		90% Confidence Limits	
0.1800	0.3800	0.2719	0.3265

Example 47.5: Analysis of a 2x2 Contingency Table

(View the complete code for this example (freqex5.sas) in the example repository.)

This example computes chi-square tests and Fisher's exact test to compare the probability of coronary heart disease for two types of diet. It also estimates the relative risks and computes exact confidence limits for the odds ratio.

The data set FatComp contains hypothetical data for a case-control study of high fat diet and the risk of coronary heart disease. The data are recorded as cell counts, where the variable Count contains the frequencies for each exposure and response combination. The data set is sorted in descending order by the variables Exposure and Response, so that the first cell of the 2×2 table contains the frequency of positive exposure and positive response. The FORMAT procedure creates formats to identify the type of exposure and response with character values.

```
proc format;
   value ExpFmt 1='High Cholesterol Diet'
                0='Low Cholesterol Diet';
   value RspFmt 1='Yes'
                0='No';
run;

data FatComp;
   input Exposure Response Count;
   label Response='Heart Disease';
   datalines;
0 0  6
0 1  2
1 0  4
1 1 11
;
```

```
proc sort data=FatComp;
   by descending Exposure descending Response;
run;
```

In the following PROC FREQ statements, ORDER=DATA option orders the contingency table values by their order in the input data set. The TABLES statement requests a two-way table of Exposure by Response. The CHISQ option produces several chi-square tests, and the RELRISK option produces relative risk measures. The EXACT statement requests the exact Pearson chi-square test and exact confidence limits for the odds ratio.

```
proc freq data=FatComp order=data;
   format Exposure ExpFmt. Response RspFmt.;
   tables Exposure*Response / chisq relrisk;
   exact pchi or;
   weight Count;
   title 'Case-Control Study of High Fat/Cholesterol Diet';
run;
```

The contingency table in Output 47.5.1 displays the variable values so that the first table cell contains the frequency for the first cell in the data set (the frequency of positive exposure and positive response).

Output 47.5.1 Contingency Table

Case-Control Study of High Fat/Cholesterol Diet

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Exposure by Response		
		Response(Heart Disease)	
Exposure	Yes	No	Total
High Cholesterol Diet	11 47.83 73.33 84.62	4 17.39 26.67 40.00	15 65.22
Low Cholesterol Diet	2 8.70 25.00 15.38	6 26.09 75.00 60.00	8 34.78
Total	13 56.52	10 43.48	23 100.00

Output 47.5.2 displays the chi-square statistics. Because the expected counts in some of the table cells are small, PROC FREQ gives a warning that the asymptotic chi-square tests might not be appropriate. In this case, the exact tests are appropriate. The alternative hypothesis for this analysis states that coronary heart disease is more likely to be associated with a high fat diet, and therefore a one-sided test is appropriate. Fisher's exact right-sided test analyzes whether the probability of heart disease in the high fat group exceeds the probability of heart disease in the low fat group; because this p-value is small, the alternative hypothesis is supported.

The odds ratio, displayed in Output 47.5.3, provides an estimate of the relative risk when an event is rare. This estimate indicates that the odds of heart disease is 8.25 times higher in the high fat diet group; however, the wide confidence limits indicate that this estimate has low precision.

Output 47.5.2 Chi-Square Statistics

Statistic	DF	Value	Prob
Chi-Square	1	4.9597	0.0259
Likelihood Ratio Chi-Square	1	5.0975	0.0240
Continuity Adj. Chi-Square	1	3.1879	0.0742
Mantel-Haenszel Chi-Square	1	4.7441	0.0294
Phi Coefficient		0.4644	
Contingency Coefficient		0.4212	
Cramer's V		0.4644	

WARNING: 50% of the cells have expected counts less than 5.
(Asymptotic) Chi-Square may not be a valid test.

Pearson Chi-Square Test	
Chi-Square	4.9597
DF	1
Asymptotic Pr > ChiSq	0.0259
Exact Pr >= ChiSq	0.0393

Fisher's Exact Test	
Cell (1,1) Frequency (F)	11
Left-sided Pr <= F	0.9967
Right-sided Pr >= F	0.0367
Table Probability (P)	0.0334
Two-sided Pr <= P	0.0393

Output 47.5.3 Relative Risk

Odds Ratio and Relative Risks			
Statistic	Value	95% Confidence Limits	
Odds Ratio	8.2500	1.1535	59.0029
Relative Risk (Column 1)	2.9333	0.8502	10.1204
Relative Risk (Column 2)	0.3556	0.1403	0.9009

Odds Ratio	
Odds Ratio	8.2500
Asymptotic Conf Limits	
95% Lower Conf Limit	1.1535
95% Upper Conf Limit	59.0029
Exact Conf Limits	
95% Lower Conf Limit	0.8677
95% Upper Conf Limit	105.5488

Example 47.6: Output Data Set of Chi-Square Statistics

(View the complete code for this example (freqex6.sas) in the example repository.)

This example uses the Color data from Example 47.1 to output the Pearson chi-square and the likelihood ratio chi-square statistics to a SAS data set. The following PROC FREQ statements create a two-way table of eye color versus hair color.

```
proc freq data=Color order=data;
   tables Eyes*Hair / expected cellchi2 norow nocol chisq;
   output out=ChiSqData n nmiss pchi lrchi;
   weight Count;
   title 'Chi-Square Tests for 3 by 5 Table of Eye and Hair Color';
run;

proc print data=ChiSqData noobs;
   title1 'Chi-Square Statistics for Eye and Hair Color';
   title2 'Output Data Set from the FREQ Procedure';
run;
```

The EXPECTED option displays expected cell frequencies in the crosstabulation table, and the CELLCHI2 option displays the cell contribution to the overall chi-square. The NOROW and NOCOL options suppress the display of row and column percents in the crosstabulation table. The CHISQ option produces chi-square tests.

The OUTPUT statement creates the ChiSqData output data set and specifies the statistics to include. The N option requests the number of nonmissing observations, the NMISS option stores the number of missing observations, and the PCHI and LRCHI options request Pearson and likelihood ratio chi-square statistics, respectively, together with their degrees of freedom and *p*-values.

The preceding statements produce Output 47.6.1 and Output 47.6.2. The contingency table in Output 47.6.1 displays eye and hair color in the order in which they appear in the Color data set. The Pearson chi-square statistic in Output 47.6.2 provides evidence of an association between eye and hair color (p=0.0073). The cell chi-square values show that most of the association is due to more green-eyed children with fair or red hair and fewer with dark or black hair. The opposite occurs with the brown-eyed children.

Output 47.6.3 displays the output data set created by the OUTPUT statement. It includes one observation that contains the sample size, the number of missing values, and the chi-square statistics and corresponding degrees of freedom and *p*-values as in Output 47.6.2.

Output 47.6.1 Contingency Table

Chi-Square Tests for 3 by 5 Table of Eye and Hair Color

The FREQ Procedure

Frequency Expected Cell Chi-Square Percent	Table of Eyes by Hair					
		Hair(Hair Color)				
Eyes(Eye Color)	fair	red	medium	dark	black	Total
blue	69	28	68	51	6	222
	66.425	32.921	63.22	53.024	6.4094	
	0.0998	0.7357	0.3613	0.0772	0.0262	
	9.06	3.67	8.92	6.69	0.79	29.13
green	69	38	55	37	0	199
	59.543	29.51	56.671	47.53	5.7454	
	1.5019	2.4422	0.0492	2.3329	5.7454	
	9.06	4.99	7.22	4.86	0.00	26.12
brown	90	47	94	94	16	341
	102.03	50.568	97.109	81.446	9.8451	
	1.4187	0.2518	0.0995	1.935	3.8478	
	11.81	6.17	12.34	12.34	2.10	44.75
Total	228	113	217	182	22	762
	29.92	14.83	28.48	23.88	2.89	100.00

Output 47.6.2 Chi-Square Statistics

Statistic	DF	Value	Prob
Chi-Square	8	20.9248	0.0073
Likelihood Ratio Chi-Square	8	25.9733	0.0011
Mantel-Haenszel Chi-Square	1	3.7838	0.0518
Phi Coefficient		0.1657	
Contingency Coefficient		0.1635	
Cramer's V		0.1172	

Output 47.6.3 Output Data Set

Chi-Square Statistics for Eye and Hair Color
Output Data Set from the FREQ Procedure

N	NMISS	_PCHI_	DF_PCHI	P_PCHI	_LRCHI_	DF_LRCHI	P_LRCHI
762	0	20.9248	8	.007349898	25.9733	8	.001061424

Example 47.7: Cochran-Mantel-Haenszel Statistics

(View the complete code for this example (freqex7.sas) in the example repository.)

The data set Migraine contains hypothetical data for a clinical trial of migraine treatment. Subjects of both genders receive either a new drug therapy or a placebo. Their response to treatment is coded as 'Better' or 'Same'. The data are recorded as cell counts, and the number of subjects for each treatment and response combination is recorded in the variable Count.

```
data Migraine;
   input Gender $ Treatment $ Response $ Count @@;
   datalines;
female Active  Better 16   female Active  Same 11
female Placebo Better  5   female Placebo Same 20
male   Active  Better 12   male   Active  Same 16
male   Placebo Better  7   male   Placebo Same 19
;
```

The following PROC FREQ statements create a multiway table stratified by Gender, where Treatment forms the rows and Response forms the columns. The RELRISK option in the TABLES statement requests the odds ratio and relative risks for the two-way tables of Treatment by Response. The PLOTS= option requests a relative risk plot, which shows the relative risk and its confidence limits for each level of Gender and overall. The CMH option requests Cochran-Mantel-Haenszel statistics for the multiway table. For this stratified 2×2 table, the CMH option also produces estimates of the common relative risk and the Breslow-Day test for homogeneity of the odds ratios. The NOPRINT option suppresses the display of the crosstabulation tables.

```
ods graphics on;
proc freq data=Migraine;
   tables Gender*Treatment*Response /
          relrisk plots(only)=relriskplot(stats) cmh noprint;
   weight Count;
   title 'Clinical Trial for Treatment of Migraine Headaches';
run;
ods graphics off;
```

Output 47.7.1 through Output 47.7.4 show the results of the analysis. The relative risk plot (Output 47.7.1) displays the relative risks and confidence limits for the two levels of Gender and for the overall (common) relative risk. Output 47.7.2 displays the CMH statistics. For a stratified 2×2 table, the three CMH statistics test the same hypothesis. The significant p-value (0.004) indicates that the association between treatment and response remains strong after adjusting for gender.

The CMH option also produces a table of overall relative risks, as shown in Output 47.7.3. Because this is a prospective study, the relative risk estimate assesses the effectiveness of the new drug; the "Cohort (Col1 Risk)" values are the appropriate estimates for the first column (the risk of improvement). The probability of migraine improvement with the new drug is just over two times the probability of improvement with the placebo.

The large p-value for the Breslow-Day test (0.2218) in Output 47.7.4 indicates no significant gender difference in the odds ratios.

Output 47.7.1 Relative Risk Plot

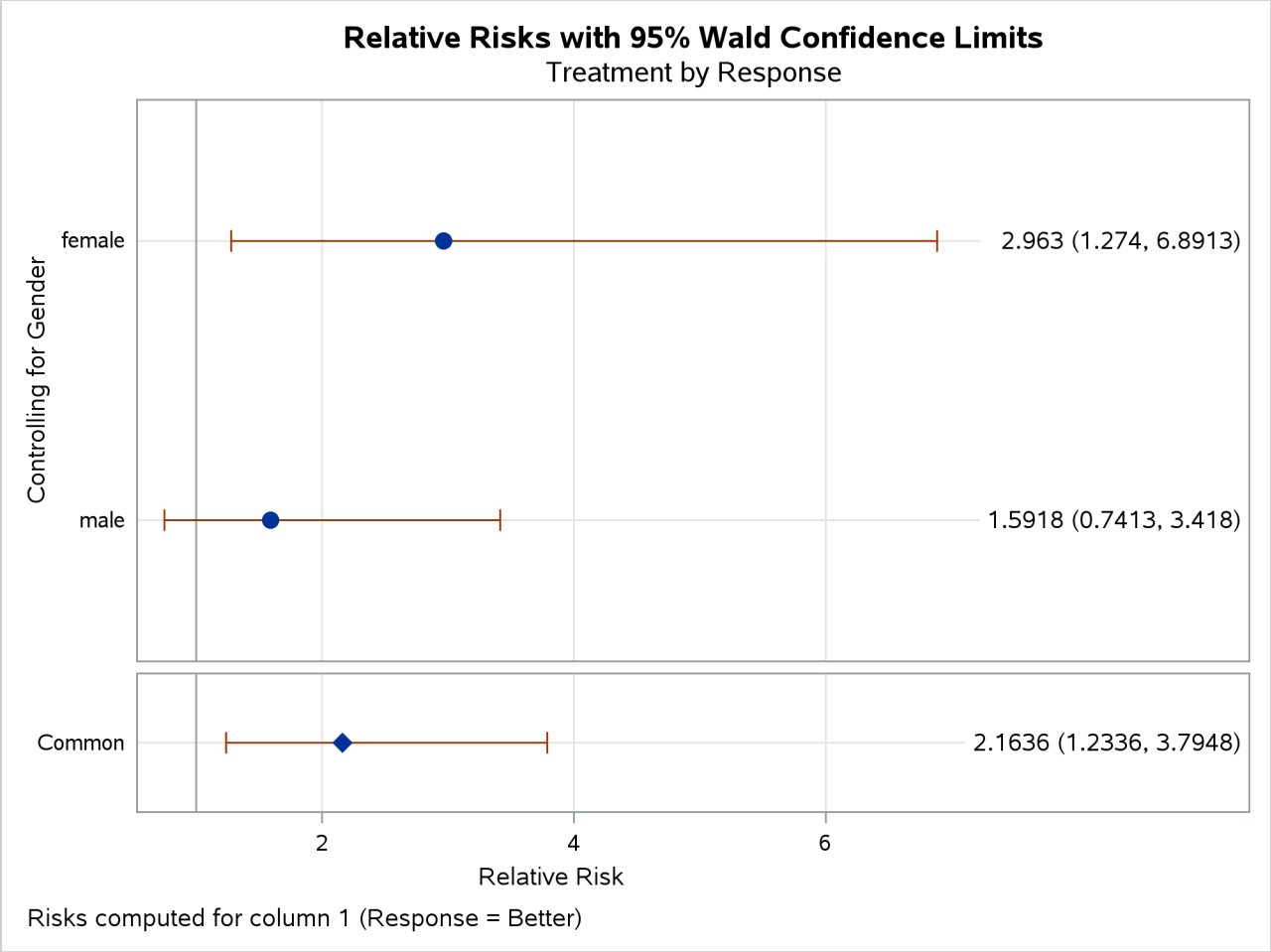

Output 47.7.2 Cochran-Mantel-Haenszel Statistics

Cochran-Mantel-Haenszel Statistics (Based on Table Scores)				
Statistic	**Alternative Hypothesis**	**DF**	**Value**	**Prob**
1	**Nonzero Correlation**	1	8.3052	0.0040
2	**Row Mean Scores Differ**	1	8.3052	0.0040
3	**General Association**	1	8.3052	0.0040

Output 47.7.3 CMH Option: Common Relative Risks

Common Odds Ratio and Relative Risks				
Statistic	**Method**	**Value**	**95% Confidence Limits**	
Odds Ratio	Mantel-Haenszel	3.3132	1.4456	7.5934
	Logit	3.2941	1.4182	7.6515
Relative Risk (Column 1)	Mantel-Haenszel	2.1636	1.2336	3.7948
	Logit	2.1059	1.1951	3.7108
Relative Risk (Column 2)	Mantel-Haenszel	0.6420	0.4705	0.8761
	Logit	0.6613	0.4852	0.9013

Output 47.7.4 CMH Option: Breslow-Day Test

Breslow-Day Test for Homogeneity of Odds Ratios	
Chi-Square	1.4929
DF	1
Pr > ChiSq	0.2218

Example 47.8: Cochran-Armitage Trend Test

(View the complete code for this example (freqex8.sas) in the example repository.)

The data set Pain contains hypothetical data for a clinical trial of a drug therapy to control pain. The clinical trial investigates whether adverse responses increase with larger drug doses. Subjects receive either a placebo or one of four drug doses. An adverse response is recorded as Adverse='Yes'; otherwise, it is recorded as Adverse='No'. The number of subjects for each drug dose and response combination is contained in the variable Count.

```
data pain;
   input Dose Adverse $ Count @@;
   datalines;
0 No 26   0 Yes  6
1 No 26   1 Yes  7
2 No 23   2 Yes  9
3 No 18   3 Yes 14
4 No  9   4 Yes 23
;
```

The following PROC FREQ statements provide a trend analysis. The TABLES statement requests a table of Adverse by Dose. The MEASURES option produces measures of association, and the CL option produces confidence limits for these measures. The TREND option tests for a trend across the ordinal values of the variable Dose with the Cochran-Armitage test. The PLOTS= option requests a mosaic plot of Adverse by Dose.

The EXACT statement produces exact p-values for this test, and the MAXTIME= option terminates the exact computations if they do not complete within 60 seconds. The TEST statement computes an asymptotic test for Somers' $D(R|C)$.

```
ods graphics on;
proc freq data=Pain;
   tables Adverse*Dose / trend measures cl
          plots=mosaicplot;
   test smdrc;
   exact trend / maxtime=60;
   weight Count;
   title 'Clinical Trial for Treatment of Pain';
run;
ods graphics off;
```

Output 47.8.1 through Output 47.8.4 display the results of the analysis. The "Col Pct" values in Output 47.8.1 show the expected increasing trend in the proportion of adverse effects with the increasing dosage (from 18.75% to 71.88%). The corresponding mosaic plot (Output 47.8.2) also shows this increasing trend.

Output 47.8.1 Contingency Table

Clinical Trial for Treatment of Pain

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Adverse by Dose					
		Dose				
Adverse	**0**	**1**	**2**	**3**	**4**	**Total**
No	26	26	23	18	9	102
	16.15	16.15	14.29	11.18	5.59	63.35
	25.49	25.49	22.55	17.65	8.82	
	81.25	78.79	71.88	56.25	28.13	
Yes	6	7	9	14	23	59
	3.73	4.35	5.59	8.70	14.29	36.65
	10.17	11.86	15.25	23.73	38.98	
	18.75	21.21	28.13	43.75	71.88	
Total	32	33	32	32	32	161
	19.88	20.50	19.88	19.88	19.88	100.00

Output 47.8.2 Mosaic Plot

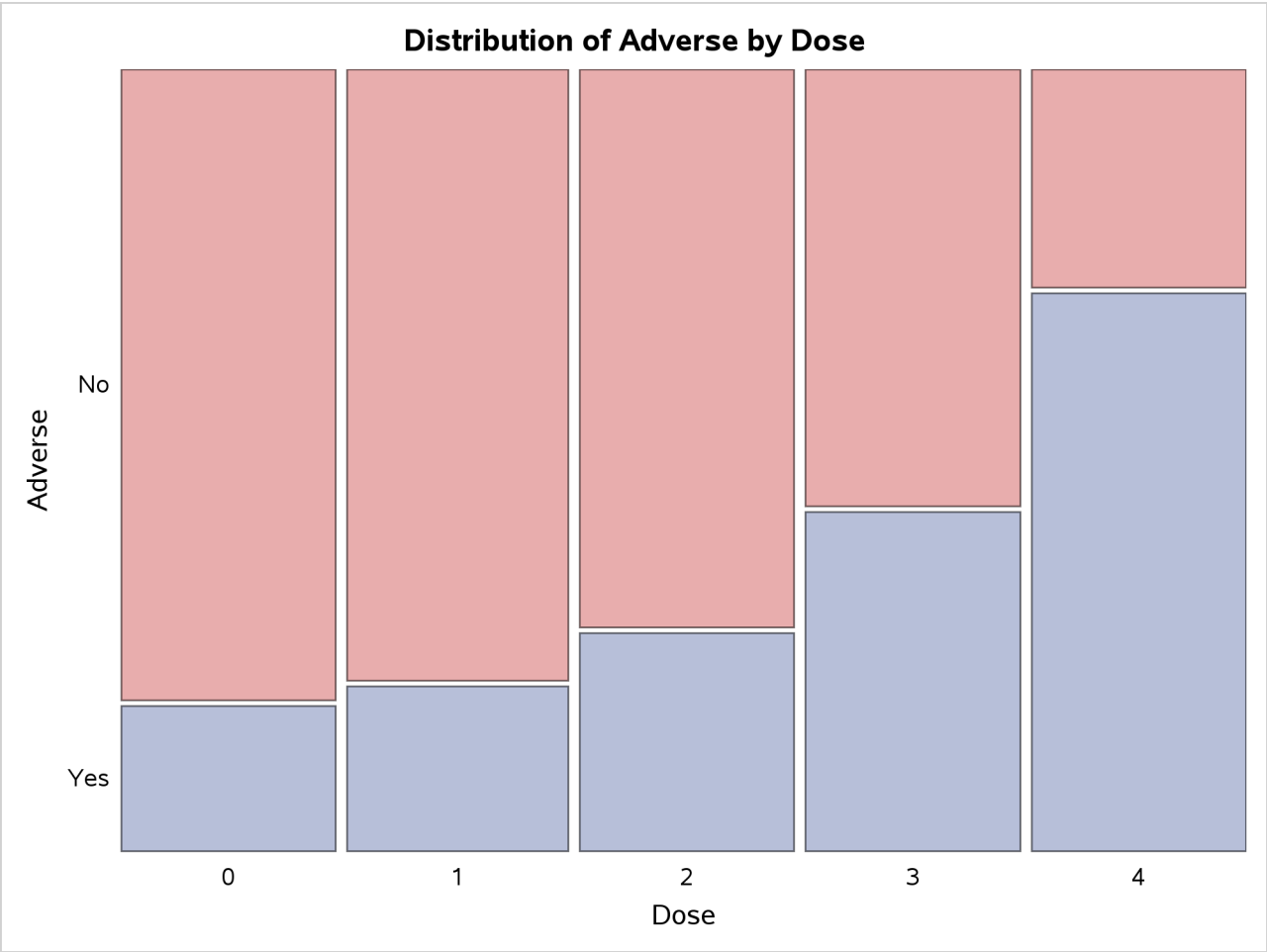

Output 47.8.3 displays the measures of association produced by the MEASURES option. Somers' $D(R|C)$

measures the association treating the row variable (Adverse) as the response and the column variable (Dose) as a predictor. Because the asymptotic 95% confidence limits do not contain 0, this indicates a strong positive association. Similarly, the Pearson and Spearman correlation coefficients show evidence of a strong positive association, as hypothesized.

The Cochran-Armitage test (Output 47.8.4) supports the trend hypothesis. The small left-sided *p*-values for the Cochran-Armitage test indicate that the probability of the Row 1 level (Adverse='No') decreases as Dose increases or, equivalently, that the probability of the Row 2 level (Adverse='Yes') increases as Dose increases. The two-sided *p*-value tests against either an increasing or decreasing alternative. This is an appropriate hypothesis when you want to determine whether the drug has progressive effects on the probability of adverse effects but the direction is unknown.

Output 47.8.3 Measures of Association

Statistic	Value	ASE	95% Confidence Limits	
Gamma	0.5313	0.0935	0.3480	0.7146
Kendall's Tau-b	0.3373	0.0642	0.2114	0.4631
Stuart's Tau-c	0.4111	0.0798	0.2547	0.5675
Somers' D C|R	0.4427	0.0837	0.2786	0.6068
Somers' D R|C	0.2569	0.0499	0.1592	0.3547
Pearson Correlation	0.3776	0.0714	0.2378	0.5175
Spearman Correlation	0.3771	0.0718	0.2363	0.5178
Lambda Asymmetric C|R	0.1250	0.0662	0.0000	0.2547
Lambda Asymmetric R|C	0.2373	0.0837	0.0732	0.4014
Lambda Symmetric	0.1604	0.0621	0.0388	0.2821
Uncertainty Coefficient C|R	0.0515	0.0191	0.0140	0.0890
Uncertainty Coefficient R|C	0.1261	0.0467	0.0346	0.2175
Uncertainty Coefficient Symmetric	0.0731	0.0271	0.0199	0.1262

Somers' D R|C	
Somers' D R|C	0.2569
ASE	0.0499
95% Lower Conf Limit	0.1592
95% Upper Conf Limit	0.3547

Test of H0: Somers' D R|C = 0	
ASE under H0	0.0499
Z	5.1511
One-sided Pr > Z	<.0001
Two-sided Pr > |Z|	<.0001

Output 47.8.4 Trend Test

Cochran-Armitage Trend Test	
Statistic (Z)	-4.7918
Asymptotic Test	
One-sided Pr < Z	<.0001
Two-sided Pr > |Z|	<.0001
Exact Test	
One-sided Pr <= Z	<.0001
Two-sided Pr >= |Z|	<.0001

Example 47.9: Friedman's Chi-Square Test

(View the complete code for this example (freqex9.sas) in the example repository.)

Friedman's test is a nonparametric test for treatment differences in a randomized complete block design. Each block of the design might be a subject or a homogeneous group of subjects. If blocks are groups of subjects, the number of subjects in each block must equal the number of treatments. Treatments are randomly assigned to subjects within each block. If there is one subject per block, the subjects are repeatedly measured once under each treatment. The order of treatments is randomized for each subject.

In this setting, Friedman's test is identical to the ANOVA (row means scores) CMH statistic when the analysis uses rank scores (SCORES=RANK). The three-way table uses subject (or subject group) as the stratifying variable, treatment as the row variable, and response as the column variable. PROC FREQ handles ties by assigning midranks to tied response values. If there are multiple subjects per treatment in each block, the ANOVA CMH statistic is a generalization of Friedman's test.

The data set Hypnosis contains data from a study investigating whether hypnosis has the same effect on skin potential (measured in millivolts) for four emotions (Lehmann and D'Abrera 2006, p. 264). Eight subjects are asked to display fear, joy, sadness, and calmness under hypnosis. The data are recorded as one observation per subject for each emotion.

```
data Hypnosis;
   length Emotion $ 10;
   input Subject Emotion $ SkinResponse @@;
   datalines;
1 fear 23.1  1 joy 22.7  1 sadness 22.5  1 calmness 22.6
2 fear 57.6  2 joy 53.2  2 sadness 53.7  2 calmness 53.1
3 fear 10.5  3 joy  9.7  3 sadness 10.8  3 calmness  8.3
4 fear 23.6  4 joy 19.6  4 sadness 21.1  4 calmness 21.6
5 fear 11.9  5 joy 13.8  5 sadness 13.7  5 calmness 13.3
6 fear 54.6  6 joy 47.1  6 sadness 39.2  6 calmness 37.0
7 fear 21.0  7 joy 13.6  7 sadness 13.7  7 calmness 14.8
8 fear 20.3  8 joy 23.6  8 sadness 16.3  8 calmness 14.8
;
```

In the following PROC FREQ statements, the TABLES statement creates a three-way table stratified by Subject and a two-way table; the variables Emotion and SkinResponse form the rows and columns

of each table. The CMH2 option produces the first two Cochran-Mantel-Haenszel statistics, the option SCORES=RANK specifies that rank scores are used to compute these statistics, and the NOPRINT option suppresses the contingency tables. These statements produce Output 47.9.1 and Output 47.9.2.

```
proc freq data=Hypnosis;
   tables Subject*Emotion*SkinResponse /
          cmh2 scores=rank noprint;
run;

proc freq data=Hypnosis;
   tables Emotion*SkinResponse /
          cmh2 scores=rank noprint;
run;
```

Because the CMH statistics in Output 47.9.1 are based on rank scores, the Row Mean Scores Differ statistic is identical to Friedman's chi-square ($Q = 6.45$). The p-value of 0.0917 indicates that differences in skin potential response for different emotions are significant at the 10% level but not at the 5% level.

When you do not stratify by subject, the Row Mean Scores Differ CMH statistic is identical to a Kruskal-Wallis test and is not significant ($p = 0.9038$ in Output 47.9.2). Thus, adjusting for subject is critical to reducing the background variation due to subject differences.

Output 47.9.1 CMH Statistics: Stratifying by Subject

The FREQ Procedure

Summary Statistics for Emotion by SkinResponse Controlling for Subject

Cochran-Mantel-Haenszel Statistics (Based on Rank Scores)				
Statistic	Alternative Hypothesis	DF	Value	Prob
1	Nonzero Correlation	1	0.2400	0.6242
2	Row Mean Scores Differ	3	6.4500	0.0917

Output 47.9.2 CMH Statistics: No Stratification

The FREQ Procedure

Summary Statistics for Emotion by SkinResponse

Cochran-Mantel-Haenszel Statistics (Based on Rank Scores)				
Statistic	Alternative Hypothesis	DF	Value	Prob
1	Nonzero Correlation	1	0.0001	0.9933
2	Row Mean Scores Differ	3	0.5678	0.9038

Example 47.10: Cochran's Q Test

(View the complete code for this example (freqex10.sas) in the example repository.)

When a binary response is measured several times or under different conditions, Cochran's Q tests that the marginal probability of a positive response is unchanged across the times or conditions. When there are more than two response categories, you can use the CATMOD procedure to fit a repeated-measures model.

The data set Drugs contains data for a study of three drugs to treat a chronic disease (Agresti 2002). Forty-six subjects receive drugs A, B, and C. The response to each drug is either favorable ('F') or unfavorable ('U').

```
proc format;
   value $ResponseFmt 'F'='Favorable'
                      'U'='Unfavorable';
run;

data drugs;
   input Drug_A $ Drug_B $ Drug_C $ Count @@;
   datalines;
F F F  6   U F F  2
F F U 16   U F U  4
F U F  2   U U F  6
F U U  4   U U U  6
;
```

The following statements create one-way frequency tables of the responses to each drug. The AGREE option produces Cochran's Q and other measures of agreement for the three-way table. These statements produce Output 47.10.1 through Output 47.10.5.

```
proc freq data=Drugs;
   tables Drug_A Drug_B Drug_C / nocum;
   tables Drug_A*Drug_B*Drug_C / agree noprint;
   format Drug_A Drug_B Drug_C $ResponseFmt.;
   weight Count;
   title 'Study of Three Drug Treatments for a Chronic Disease';
run;
```

The one-way frequency tables in Output 47.10.1 provide the marginal response for each drug. For drugs A and B, 61% of the subjects reported a favorable response; for drug C, 35% of the subjects reported a favorable response. Output 47.10.2 and Output 47.10.3 display measures of agreement for the 'Favorable' and 'Unfavorable' levels of drug A, respectively. McNemar's test shows a strong discordance between drugs B and C when the response to drug A is favorable.

Output 47.10.1 One-Way Frequency Tables

Study of Three Drug Treatments for a Chronic Disease

The FREQ Procedure

Drug_A	Frequency	Percent
Favorable	28	60.87
Unfavorable	18	39.13

Output 47.10.1 *continued*

Drug_B	Frequency	Percent
Favorable	28	60.87
Unfavorable	18	39.13

Drug_C	Frequency	Percent
Favorable	16	34.78
Unfavorable	30	65.22

Output 47.10.2 Measures of Agreement for Drug A Favorable

McNemar's Test		
Chi-Square	DF	Pr > ChiSq
10.8889	1	0.0010

Simple Kappa Coefficient			
Estimate	Standard Error	95% Confidence Limits	
-0.0328	0.1167	-0.2615	0.1960

Output 47.10.3 Measures of Agreement for Drug A Unfavorable

McNemar's Test		
Chi-Square	DF	Pr > ChiSq
0.4000	1	0.5271

Simple Kappa Coefficient			
Estimate	Standard Error	95% Confidence Limits	
-0.1538	0.2230	-0.5909	0.2832

Output 47.10.4 displays the overall kappa coefficient. The small negative value of kappa indicates no agreement between drug B response and drug C response.

Output 47.10.4 Overall Measures of Agreement

Overall Kappa Coefficient			
Estimate	Standard Error	95% Confidence Limits	
-0.0588	0.1034	-0.2615	0.1439

Test for Equal Kappas		
Chi-Square	DF	Pr > ChiSq
0.2314	1	0.6305

Cochran's Q is statistically significant (p=0.0145 in Output 47.10.5), which leads to rejection of the hypothesis that the probability of favorable response is the same for the three drugs.

Output 47.10.5 Cochran's Q Test

Cochran's Q, for Drug_A by Drug_B by Drug_C		
Chi-Square	**DF**	**Pr > ChiSq**
8.4706	2	0.0145

References

Agresti, A. (1992). "A Survey of Exact Inference for Contingency Tables." *Statistical Science* 7:131–177.

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Agresti, A. (2007). *An Introduction to Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Agresti, A. (2013). *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Agresti, A., and Caffo, B. (2000). "Simple and Effective Confidence Intervals for Proportions and Differences of Proportions Result from Adding Two Successes and Two Failures." *American Statistician* 54:280–288.

Agresti, A., and Coull, B. A. (1998). "Approximate Is Better Than 'Exact' for Interval Estimation of Binomial Proportions." *American Statistician* 52:119–126.

Agresti, A., and Gottard, A. (2007). "Nonconservative Exact Small-Sample Inference for Discrete Data." *Computational Statistics and Data Analysis* 51:6447–6458.

Agresti, A., Mehta, C. R., and Patel, N. R. (1990). "Exact Inference for Contingency Tables with Ordered Categories." *Journal of the American Statistical Association* 85:453–458.

Agresti, A., and Min, Y. (2001). "On Small-Sample Confidence Intervals for Parameters in Discrete Distributions." *Biometrics* 57:963–971.

Agresti, A., Wackerly, D., and Boyett, J. M. (1979). "Exact Conditional Tests for Cross-Classifications: Approximation of Attained Significance Levels." *Psychometrika* 44:75–83.

Bangdiwala, S. I. (1988). *The Agreement Chart*. Technical report, Department of Biostatistics, University of North Carolina at Chapel Hill.

Bangdiwala, S. I., and Bryan, H. E. (1987). "Using SAS Software Graphical Procedures for the Observer Agreement Chart." In *Proceedings of the Twelfth Annual SAS Users Group International Conference*, 1083–1088. Cary, NC: SAS Institute Inc.

Bangdiwala, S. I., Haedo, A. S., Natal, M. L., and Villaveces, A. (2008). "The Agreement Chart as an Alternative to the Receiver-Operating Characteristic Curve for Diagnostic Tests." *Journal of Clinical Epidemiology* 61:866–874.

Barker, L., Rolka, H., Rolka, D., and Brown, C. (2001). "Equivalence Testing for Binomial Random Variables: Which Test to Use?" *American Statistician* 55:279–287.

Barnard, G. A. (1945). "A New Test for 2×2 Tables." *Nature* 156:177.

Barnard, G. A. (1947). "Significance Tests for 2 × 2 Tables." *Biometrika* 34:123–138.

Barnard, G. A. (1949). "Statistical Inference." *Journal of the Royal Statistical Society, Series B* 11:115–139.

Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis*. 2nd ed. New York: Springer-Verlag.

Birch, M. W. (1965). "The Detection of Partial Association, Part 2: The General Case." *Journal of the Royal Statistical Society, Series B* 27:111–124.

Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1975). *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, MA: MIT Press.

Blaker, H. (2000). "Confidence Curves and Improved Exact Confidence Intervals for Discrete Distributions." *Canadian Journal of Statistics* 28:783–798.

Blood, E., and Spratt, K. F. (2007). "Disagreement on Agreement: Two Alternative Agreement Coefficients." In *Proceedings of the SAS Global Forum 2007 Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/forum2007/186-2007.pdf.

Bowker, A. H. (1948). "Bowker's Test for Symmetry." *Journal of the American Statistical Association* 43:572–574.

Breslow, N. E. (1996). "Statistics in Epidemiology: The Case-Control Study." *Journal of the American Statistical Association* 91:14–26.

Breslow, N. E., and Day, N. E. (1980). *The Analysis of Case-Control Studies*. Statistical Methods in Cancer Research, IARC Scientific Publications, vol. 1, no. 32. Lyon: International Agency for Research on Cancer.

Breslow, N. E., and Day, N. E. (1987). *The Design and Analysis of Cohort Studies*. Statistical Methods in Cancer Research, IARC Scientific Publications, vol. 2, no. 82. Lyon: International Agency for Research on Cancer.

Bross, I. D. J. (1958). "How to Use Ridit Analysis." *Biometrics* 14:18–38.

Brown, L. D., Cai, T. T., and DasGupta, A. (2001). "Interval Estimation for a Binomial Proportion." *Statistical Science* 16:101–133.

Brown, M. B., and Benedetti, J. K. (1977a). "On the Mean and Variance of the Tetrachoric Correlation Coefficient." *Psychometrika* 42:347–355.

Brown, M. B., and Benedetti, J. K. (1977b). "Sampling Behavior of Tests for Correlation in Two-Way Contingency Tables." *Journal of the American Statistical Association* 72:309–315.

Byrt, T., Bishop, J., and Carlin, J. B. (1993). "Bias, Prevalence, and Kappa." *Journal of Clinical Epidemiology* 46:423–429.

Chan, I. S. F. (1998). "Exact Tests of Equivalence and Efficacy with a Non-zero Lower Bound for Comparative Studies." *Statistics in Medicine* 17:1403–1413.

Chan, I. S. F. (2003). "Proving Non-inferiority or Equivalence of Two Treatments with Dichotomous Endpoints Using Exact Methods." *Statistical Methods in Medical Research* 12:37–58.

Chan, I. S. F., and Zhang, Z. (1999). "Test-Based Exact Confidence Intervals for the Difference of Two Binomial Proportions." *Biometrics* 55:1202–1209.

Chow, S.-C., Shao, J., and Wang, H. (2003). *Sample Size Calculations in Clinical Research*. Boca Raton, FL: CRC Press.

Chow, S.-C., Shao, J., and Wang, H. (2008). *Sample Size Calculations in Clinical Research*. 2nd ed. Boca Raton, FL: Chapman & Hall/CRC.

Cicchetti, D. V., and Allison, T. (1971). "A New Procedure for Assessing Reliability of Scoring EEG Sleep Recordings." *American Journal of EEG Technology* 11:101–109.

Clopper, C. J., and Pearson, E. S. (1934). "The Use of Confidence or Fiducial Limits Illustrated in the Case of the Binomial." *Biometrika* 26:404–413.

Cochran, W. G. (1950). "The Comparison of Percentages in Matched Samples." *Biometrika* 37:256–266.

Cochran, W. G. (1954). "Some Methods for Strengthening the Common χ^2 Tests." *Biometrics* 10:417–451.

Cohen, J. (1960). "A Coefficient of Agreement for Nominal Scales." *Educational and Psychological Measurement* 20:37–46.

Collett, D. (1991). *Modelling Binary Data*. London: Chapman & Hall.

Dann, R. S., and Koch, G. G. (2005). "Review and Evaluation of Methods for Computing Confidence Intervals for the Ratio of Two Proportions and Considerations for Non-inferiority Clinical Trials." *Journal of Biopharmaceutical Statistics* 15:85–107.

Dmitrienko, A., Molenberghs, G., Chuang-Stein, C., and Offen, W. (2005). *Analysis of Clinical Trials Using SAS: A Practical Guide*. Cary, NC: SAS Institute Inc.

Drasgow, F. (1986). "Polychoric and Polyserial Correlations." In *Encyclopedia of Statistical Sciences*, vol. 7, edited by S. Kotz, N. L. Johnson, and C. B. Read. New York: John Wiley & Sons.

Dunnett, C. W., and Gent, M. (1977). "Significance Testing to Establish Equivalence between Treatments, with Special Reference to Data in the Form of 2×2 Tables." *Biometrics* 33:593–602.

Farrington, C. P., and Manning, G. (1990). "Test Statistics and Sample Size Formulae for Comparative Binomial Trials with Null Hypothesis of Non-zero Risk Difference or Non-unity Relative Risk." *Statistics in Medicine* 9:1447–1454.

Fienberg, S. E. (1980). *The Analysis of Cross-Classified Categorical Data*. 2nd ed. Cambridge, MA: MIT Press.

Fisher, K. (2015). "Evaluating Methods of Estimating Common Risk Difference for Stratified Binomial Trials for Less Common Events." Poster presented at the Duke-Industry Statistics Symposium, Durham, NC. https://sites.duke.edu/biostatsworkshop2015/files/2015/10/P8_Poster_for_Duke_Symposium_Kate_Fisher_101915_Final.pdf.

Fleiss, J. L., and Cohen, J. (1973). "The Equivalence of Weighted Kappa and the Intraclass Correlation Coefficient as Measures of Reliability." *Educational and Psychological Measurement* 33:613–619.

Fleiss, J. L., Cohen, J., and Everitt, B. S. (1969). "Large-Sample Standard Errors of Kappa and Weighted Kappa." *Psychological Bulletin* 72:323–327.

Fleiss, J. L., Levin, B., and Paik, M. C. (2003). *Statistical Methods for Rates and Proportions*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Freeman, G. H., and Halton, J. H. (1951). "Note on an Exact Treatment of Contingency, Goodness of Fit, and Other Problems of Significance." *Biometrika* 38:141–149.

Friendly, M. (2000). *Visualizing Categorical Data*. Cary, NC: SAS Institute Inc.

Gail, M. H., and Mantel, N. (1977). "Counting the Number of $r \times c$ Contingency Tables with Fixed Margins." *Journal of the American Statistical Association* 72:859–862.

Gail, M. H., and Simon, R. (1985). "Tests for Qualitative Interactions between Treatment Effects and Patient Subsets." *Biometrics* 41:361–372.

Gart, J. J. (1971). "The Comparison of Proportions: A Review of Significance Tests, Confidence Intervals, and Adjustments for Stratification." *Review of the International Statistical Institute* 39:148–169.

Gart, J. J., and Nam, J. (1988). "Approximate Interval Estimation of the Ratio of Binomial Parameters: A Review and Corrections for Skewness." *Biometrics* 44:323–338.

Goodman, L. A., and Kruskal, W. H. (1979). *Measures of Association for Cross Classification*. New York: Springer-Verlag.

Greenland, S., and Robins, J. M. (1985). "Estimation of a Common Effect Parameter from Sparse Follow-Up Data." *Biometrics* 41:55–68.

Gwet, K. L. (2008). "Computing Inter-rater Reliability and Its Variance in the Presence of High Agreement." *British Journal of Mathematical and Statistical Psychology* 61:29–48.

Haldane, J. B. S. (1956). "The Estimation and Significance of the Logarithm of a Ratio of Frequencies." *Annals of Human Genetics* 20:309–311.

Hauck, W. W., and Anderson, S. (1986). "A Comparison of Large-Sample Confidence Interval Methods for the Difference of Two Binomial Probabilities." *American Statistician* 40:318–322.

Higgins, J. P. T., and Thompson, S. G. (2002). "Quantifying Heterogeneity in a Meta-analysis." *Statistics in Medicine* 21:1539–1558.

Higgins, J. P. T., Thompson, S. G., Deeks, J. J., and Altman, D. G. (2003). "Measuring Inconsistency in Meta-analyses." *BMJ* 327:557–560.

Hirji, K. F. (2006). *Exact Analysis of Discrete Data*. Boca Raton, FL: Chapman & Hall/CRC.

Hirji, K. F., Vollset, S. E., Reis, I. M., and Afifi, A. A. (1996). "Exact Tests for Interaction in Several 2×2 Tables." *Journal of Computational and Graphical Statistics* 5:209–224.

Hoenig, J. M., Morgan, M. J., and Brown, C. A. (1995). "Analysing Differences between Two Age Determination Methods by Tests of Symmetry." *Canadian Journal of Fisheries and Aquatic Sciences* 52:364–368.

Hollander, M., and Wolfe, D. A. (1999). *Nonparametric Statistical Methods*. 2nd ed. New York: John Wiley & Sons.

Holley, J. W., and Guilford, J. P. (1964). "A Note on the G Index of Agreement." *Educational and Psychological Measurement* 24:749–753.

Jones, M. P., O'Gorman, T. W., Lemka, J. H., and Woolson, R. F. (1989). "A Monte Carlo Investigation of Homogeneity Tests of the Odds Ratio under Various Sample Size Configurations." *Biometrics* 45:171–181.

Kendall, M. G. (1955). *Rank Correlation Methods*. 2nd ed. London: Charles Griffin.

Kendall, M. G., and Stuart, A. (1979). *The Advanced Theory of Statistics.* 4th ed. Vol. 2. New York: Macmillan.

Kim, Y., and Won, S. (2013). "Adjusted Proportion Difference and Confidence Interval in Stratified Randomized Trials." In *Proceedings of PharmaSUG 2013 (Pharmaceutical Industry SAS Users Group).* Paper SP04. Cary, NC: SAS Institute Inc.

Kleinbaum, D. G., Kupper, L. L., and Morgenstern, H. (1982). *Epidemiologic Research: Principles and Quantitative Methods*. New York: Van Nostrand Reinhold.

Klingenberg, B. (2014). "A New and Improved Confidence Interval for the Mantel-Haenszel Risk Difference." *Statistics in Medicine* 33:2968–2983.

Korn, E. L., and Graubard, B. I. (1998). "Confidence Intervals for Proportions with Small Expected Number of Positive Counts Estimated from Survey Data." *Survey Methodology* 24:193–201.

Krauth, J. (1973). "Nichtparametrische Ansätze zur Auswertung von Verlaufskurven [Nonparametric approaches to analyzing time effect curves]." *Biometrische Zeitschrift* 15:557–566.

Landis, J. R., Heyman, E. R., and Koch, G. G. (1978). "Average Partial Association in Three-Way Contingency Tables: A Review and Discussion of Alternative Tests." *International Statistical Review* 46:237–254.

Leemis, L. M., and Trivedi, K. S. (1996). "A Comparison of Approximate Interval Estimators for the Bernoulli Parameter." *American Statistician* 50:63–68.

Lehmann, E. L., and D'Abrera, H. J. M. (2006). *Nonparametrics: Statistical Methods Based on Ranks*. Rev. ed. New York: Springer.

Liebetrau, A. M. (1983). *Measures of Association.* Vol. 32 of Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Lu, K. (2008). "Cochran-Mantel-Haenszel Weighted Miettinen and Nurminen Method for Confidence Intervals of the Difference in Binomial Proportions from Stratified 2x2 Samples." In *Proceedings of the Joint Statistical Meetings*, 3888–3893. Alexandria, VA: American Statistical Association.

Mack, G. A., and Skillings, J. H. (1980). "A Friedman-Type Rank Test for Main Effects in a Two-Factor ANOVA." *Journal of the American Statistical Association* 75:947–951.

Mantel, N. (1963). "Chi-Square Tests with One Degree of Freedom: Extensions of the Mantel-Haenszel Procedure." *Journal of the American Statistical Association* 58:690–700.

Mantel, N., and Fleiss, J. L. (1980). "Minimum Expected Cell Size Requirements for the Mantel-Haenszel One-Degree-of-Freedom Chi-Square Test and a Related Rapid Procedure." *American Journal of Epidemiology* 112:129–134.

Mantel, N., and Haenszel, W. (1959). "Statistical Aspects of Analysis of Data from Retrospective Studies of Disease." *Journal of the National Cancer Institute* 22:719–748.

Margolin, B. H. (1988). "Test for Trend in Proportions." In *Encyclopedia of Statistical Sciences*, vol. 9, edited by S. Kotz, N. L. Johnson, and C. B. Read. New York: John Wiley & Sons.

McNemar, Q. (1947). "Note on the Sampling Error of the Difference between Correlated Proportions or Percentages." *Psychometrika* 12:153–157.

Mee, R. W. (1984). "Confidence Bounds for the Difference between Two Probabilities." *Biometrics* 40:1175–1176.

Mehrotra, D. V. (2001). "Stratification Issues with Binary Endpoints." *Drug Information Journal* 35:1343–1350.

Mehrotra, D. V., and Railkar, R. (2000). "Minimum Risk Weights for Comparing Treatments in Stratified Binomial Trials." *Statistics in Medicine* 19:811–825.

Mehta, C. R., and Patel, N. R. (1983). "A Network Algorithm for Performing Fisher's Exact Test in $r \times c$ Contingency Tables." *Journal of the American Statistical Association* 78:427–434.

Mehta, C. R., Patel, N. R., and Gray, R. J. (1985). "Computing an Exact Confidence Interval for the Common Odds Ratio in Several 2×2 Contingency Tables." *Journal of the American Statistical Association* 80:969–973.

Mehta, C. R., Patel, N. R., and Senchaudhuri, P. (1991). "Exact Stratified Linear Rank Tests for Binary Data." In *Computing Science and Statistics: Proceedings of the Twenty-Third Symposium on the Interface*, edited by E. M. Keramidas, 200–207. Fairfax Station, VA: Interface Foundation.

Mehta, C. R., Patel, N. R., and Tsiatis, A. A. (1984). "Exact Significance Testing to Establish Treatment Equivalence with Ordered Categorical Data." *Biometrics* 40:819–825.

Mehta, C. R., and Senchaudhuri, P. (2003). "Conditional versus Unconditional Exact Tests for Comparing Two Binomials." Cambridge, MA: Cytel Software Corporation.

Miettinen, O. S. (1985). *Theoretical Epidemiology: Principles of Occurrence in Research Medicine*. New York: John Wiley & Sons.

Miettinen, O. S., and Nurminen, M. M. (1985). "Comparative Analysis of Two Rates." *Statistics in Medicine* 4:213–226.

Newcombe, R. G. (1998a). "Interval Estimation for the Difference between Independent Proportions: Comparison of Eleven Methods." *Statistics in Medicine* 17:873–890.

Newcombe, R. G. (1998b). "Two-Sided Confidence Intervals for the Single Proportion: Comparison of Seven Methods." *Statistics in Medicine* 17:857–872.

Newcombe, R. G., and Nurminen, M. M. (2011). "In Defence of Score Intervals for Proportions and Their Differences." *Communications in Statistics—Theory and Methods* 40:1271–1282.

Olsson, U. (1979). "Maximum Likelihood Estimation of the Polychoric Correlation Coefficient." *Psychometrika* 12:443–460.

Pirie, W. (1983). "Jonckheere Tests for Ordered Alternatives." In *Encyclopedia of Statistical Sciences*, vol. 4, edited by S. Kotz, N. L. Johnson, and C. B. Read. New York: John Wiley & Sons.

Radlow, R., and Alf, E. F. (1975). "An Alternate Multinomial Assessment of the Accuracy of the Chi-Square Test of Goodness of Fit." *Journal of the American Statistical Association* 70:811–813.

Robins, J. M., Breslow, N., and Greenland, S. (1986). "Estimators of the Mantel-Haenszel Variance Consistent in Both Sparse Data and Large-Strata Limiting Models." *Biometrics* 42:311–323.

Santner, T. J., Pradhan, V., Senchaudhuri, P., Mehta, C. R., and Tamhane, A. (2007). "Small-Sample Comparisons of Confidence Intervals for the Difference of Two Independent Binomial Proportions." *Computational Statistics and Data Analysis* 51:5791–5799.

Santner, T. J., and Snell, M. K. (1980). "Small-Sample Confidence Intervals for $p_1 - p_2$ and p_1/p_2 in 2×2 Contingency Tables." *Journal of the American Statistical Association* 75:386–394.

Sato, T. (1989). "On the Variance Estimator of the Mantel-Haenszel Risk Difference." *Biometrics* 45:1323–1324. Letter to the editor.

Schuirmann, D. J. (1987). "A Comparison of the Two One-Sided Tests Procedure and the Power Approach for Assessing the Equivalence of Average Bioavailability." *Journal of Pharmacokinetics and Biopharmaceutics* 15:657–680.

Schuirmann, D. J. (1999). "Confidence Interval Methods for Bioequivalence Testing with Binomial Endpoints." In *Proceedings of the Biopharmaceutical Section*, 227–232. Alexandria, VA: American Statistical Association.

Silvapulle, M. J. (2001). "Tests against Qualitative Interaction: Exact Critical Values and Robust Tests." *Biometrics* 57:1157–1165.

Sim, J., and Wright, C. C. (2005). "The Kappa Statistic in Reliability Studies: Use, Interpretation, and Sample Size Requirements." *Physical Therapy* 85:257–268.

Snedecor, G. W., and Cochran, W. G. (1989). *Statistical Methods*. 8th ed. Ames: Iowa State University Press.

Somers, R. H. (1962). "A New Asymmetric Measure of Association for Ordinal Variables." *American Sociological Review* 27:799–811.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2012). *Categorical Data Analysis Using SAS*. 3rd ed. Cary, NC: SAS Institute Inc.

Suissa, S., and Shuster, J. J. (1985). "Exact Unconditional Sample Sizes for the 2×2 Binomial Trial." *Journal of the Royal Statistical Society, Series A* 148:317–327.

Tarone, R. E. (1985). "On Heterogeneity Tests Based on Efficient Scores." *Biometrika* 72:91–95.

Theil, H. (1972). *Statistical Decomposition Analysis*. Amsterdam: North-Holland.

Thomas, D. G. (1971). "Algorithm AS-36: Exact Confidence Limits for the Odds Ratio in a 2×2 Table." *Journal of the Royal Statistical Society, Series C* 20:105–110.

Thorlund, K., Imberger, G., Johnston, B. C., Walsh, M., Awad, T., Thabane, L., Gluud, C., et al. (2012). "Evolution of Heterogeneity (I^2) Estimates and Their 95% Confidence Intervals in Large Meta-analyses." *PLoS ONE* 7:1–8. https://doi.org/10.1371/journal.pone.0039471.

Valz, P. D., and Thompson, M. E. (1994). "Exact Inference for Kendall's S and Spearman's ρ with Extensions to Fisher's Exact Test in $r \times c$ Contingency Tables." *Journal of Computational and Graphical Statistics* 3:459–472.

Van Elteren, P. H. (1960). "On the Combination of Independent Two-Sample Tests of Wilcoxon." *Bulletin of the International Statistical Institute* 37:351–361.

Vollset, S. E., Hirji, K. F., and Elashoff, R. M. (1991). "Fast Computation of Exact Confidence Limits for the Common Odds Ratio in a Series of 2×2 Tables." *Journal of the American Statistical Association* 86:404–409.

Wilson, E. B. (1927). "Probable Inference, the Law of Succession, and Statistical Inference." *Journal of the American Statistical Association* 22:209–212.

Woolf, B. (1955). "On Estimating the Relationship between Blood Group and Disease." *Annals of Human Genetics* 19:251–253.

Xie, Q. (2013). "Agree or Disagree? A Demonstration of an Alternative Statistic to Cohen's Kappa for Measuring the Extent and Reliability of Agreement between Observers." Paper presented at FCSM Research Conference, Nov. 7–9, Washington, DC.

Yan, X., and Su, X. G. (2010). "Stratified Wilson and Newcombe Confidence Intervals for Multiple Binomial Proportions." *Statistics in Biopharmaceutical Research* 2:329–335.

Zelen, M. (1971). "The Analysis of Several 2×2 Contingency Tables." *Biometrika* 58:129–137.

Chapter 48
The GAM Procedure

Contents

Overview: GAM Procedure . . . **3376**
Getting Started: GAM Procedure . . . **3377**
Syntax: GAM Procedure . . . **3380**
- PROC GAM Statement . . . 3380
- BY Statement . . . 3382
- CLASS Statement . . . 3383
- FREQ Statement . . . 3384
- MODEL Statement . . . 3385
- OUTPUT Statement . . . 3389
- SCORE Statement . . . 3391

Details: GAM Procedure . . . **3391**
- Missing Values . . . 3391
- Nonparametric Regression . . . 3391
- Additive Models and Generalized Additive Models . . . 3392
- Forms of Additive Models . . . 3393
- Estimates from PROC GAM . . . 3394
- Backfitting and Local Scoring Algorithms . . . 3394
- Smoothers . . . 3397
- Selection of Smoothing Parameters . . . 3398
- Confidence Intervals for Smoothers . . . 3400
- Distribution Family and Canonical Link . . . 3401
- Dispersion Parameter . . . 3403
- Computational Resources . . . 3403
- ODS Table Names . . . 3405
- ODS Graphics . . . 3406

Examples: GAM Procedure . . . **3407**
- Example 48.1: Generalized Additive Model with Binary Data . . . 3407
- Example 48.2: Poisson Regression Analysis of Component Reliability . . . 3413
- Example 48.3: Comparing PROC GAM with PROC LOESS . . . 3418

References . . . **3429**

Overview: GAM Procedure

The GAM procedure fits generalized additive models as defined by Hastie and Tibshirani (1990). This procedure provides powerful tools for nonparametric regression and smoothing.

Nonparametric regression relaxes the usual assumption of linearity and enables you to uncover relationships between the independent variables and the dependent variable that might otherwise be missed. SAS provides many procedures for nonparametric regression, such as the LOESS procedure for local regression and the TPSPLINE procedure for thin-plate smoothing splines. The generalized additive models fit by the GAM procedure combine the following:

- an additivity assumption (Stone 1985) that enables relatively many nonparametric relationships to be explored simultaneously
- the distributional flexibility of generalized linear models (Nelder and Wedderburn 1972)

Thus, you can use the GAM procedure when you have multiple independent variables whose effect you want to model nonparametrically, or when the dependent variable is not normally distributed. See the section "Nonparametric Regression" on page 3391 for more details on the form of generalized additive models.

The GAM procedure does the following:

- provides nonparametric estimates for additive models
- supports the use of multidimensional data
- supports multiple SCORE statements
- fits both generalized semiparametric additive models and generalized additive models
- enables you to choose a particular model by specifying the model degrees of freedom or smoothing parameter
- produces graphs with ODS Graphics

Getting Started: GAM Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/gamgs.sas.)

The following example illustrates the use of the GAM procedure to explore in a nonparametric way how two factors affect a response. The data come from a study of the factors affecting patterns of insulin-dependent diabetes mellitus in children (Sochett et al. 1987). The objective is to investigate the dependence of the level of serum C-peptide on various other factors in order to understand the patterns of residual insulin secretion. The response measurement is the logarithm of C-peptide concentration (pmol/ml) at diagnosis, and the predictor measurements are age and base deficit (a measure of acidity).

```
title 'Patterns of Diabetes';
data diabetes;
   input Age BaseDeficit CPeptide @@;
   logCP = log(CPeptide);
   datalines;
5.2    -8.1  4.8   8.8  -16.1  4.1  10.5   -0.9  5.2
10.6   -7.8  5.5  10.4  -29.0  5.0   1.8  -19.2  3.4
12.7  -18.9  3.4  15.6  -10.6  4.9   5.8   -2.8  5.6
1.9   -25.0  3.7   2.2   -3.1  3.9   4.8   -7.8  4.5
7.9   -13.9  4.8   5.2   -4.5  4.9   0.9  -11.6  3.0
11.8   -2.1  4.6   7.9   -2.0  4.8  11.5   -9.0  5.5
10.6  -11.2  4.5   8.5   -0.2  5.3  11.1   -6.1  4.7
12.8   -1.0  6.6  11.3   -3.6  5.1   1.0   -8.2  3.9
14.5   -0.5  5.7  11.9   -2.0  5.1   8.1   -1.6  5.2
13.8  -11.9  3.7  15.5   -0.7  4.9   9.8   -1.2  4.8
11.0  -14.3  4.4  12.4   -0.8  5.2  11.1  -16.8  5.1
5.1    -5.1  4.6   4.8   -9.5  3.9   4.2  -17.0  5.1
6.9    -3.3  5.1  13.2   -0.7  6.0   9.9   -3.3  4.9
12.5  -13.6  4.1  13.2   -1.9  4.6   8.9  -10.0  4.9
10.8  -13.5  5.1
;
```

The following statements perform the desired analysis. The PROC GAM statement invokes the procedure and specifies the diabetes data set as input. The MODEL statement specifies logCP as the response variable and names Age and BaseDeficit as independent variables with univariate smoothing splines and the default of four degrees of freedom.

```
ods graphics on;
proc gam data=diabetes;
   model logCP = spline(Age) spline(BaseDeficit);
run;
```

The results are shown in Figure 48.1 and Figure 48.2.

Figure 48.1 Summary Statistics

Patterns of Diabetes

The GAM Procedure
Dependent Variable: logCP
Smoothing Model Component(s): spline(Age) spline(BaseDeficit)

Summary of Input Data Set	
Number of Observations	43
Number of Missing Observations	0
Distribution	Gaussian
Link Function	Identity

Iteration Summary and Fit Statistics	
Final Number of Backfitting Iterations	5
Final Backfitting Criterion	5.542745E-10
The Deviance of the Final Estimate	0.4180791724

Figure 48.1 shows two tables. The first table summarizes the input data set and the distributional family used for the model; the second table summarizes the convergence criterion for backfitting.

Figure 48.2 Analysis of Model

Regression Model Analysis
Parameter Estimates

Parameter	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1.48141	0.05120	28.93	<.0001
Linear(Age)	0.01437	0.00437	3.28	0.0024
Linear(BaseDeficit)	0.00807	0.00247	3.27	0.0025

Smoothing Model Analysis
Fit Summary for Smoothing Components

Component	Smoothing Parameter	DF	GCV	Num Unique Obs
Spline(Age)	0.995582	3.000000	0.011675	37
Spline(BaseDeficit)	0.995299	3.000000	0.012437	39

Smoothing Model Analysis
Analysis of Deviance

Source	DF	Sum of Squares	Chi-Square	Pr > ChiSq
Spline(Age)	3.00000	0.150761	12.2605	0.0065
Spline(BaseDeficit)	3.00000	0.081273	6.6095	0.0854

Figure 48.2 displays summary statistics for the model. It consists of three tables. The first is the "Parameter Estimates" table for the parametric part of the model. It indicates that the linear trends for both Age and BaseDeficit are highly significant. The second table is the summary of smoothing components of the nonparametric part of the model. This table presents the smoothing parameter and degrees of freedom (DF) for each component. By default, each smoothing component has approximately 4 DF. For univariate spline

components, one DF is taken up by the (parametric) linear part of the model, so the remaining approximate DF is 3. Finally, the third table is the "Analysis of Deviance" table for the nonparametric component of the model.

With ODS Graphics enabled, PROC GAM produces by default a panel of plots of partial prediction curves of smoothing components. In these plots, the partial prediction for a predictor such as Age is its nonparametric contribution to the model, $s(\mathrm{Age})$. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the GAM procedure, see the section "ODS Graphics" on page 3406.

Plots for both predictors (Figure 48.3) show a strong quadratic pattern, with a possible indication of higher-order behavior. Further investigation is required to determine whether these patterns are real or not.

Figure 48.3 Partial Predictions for Each Predictor

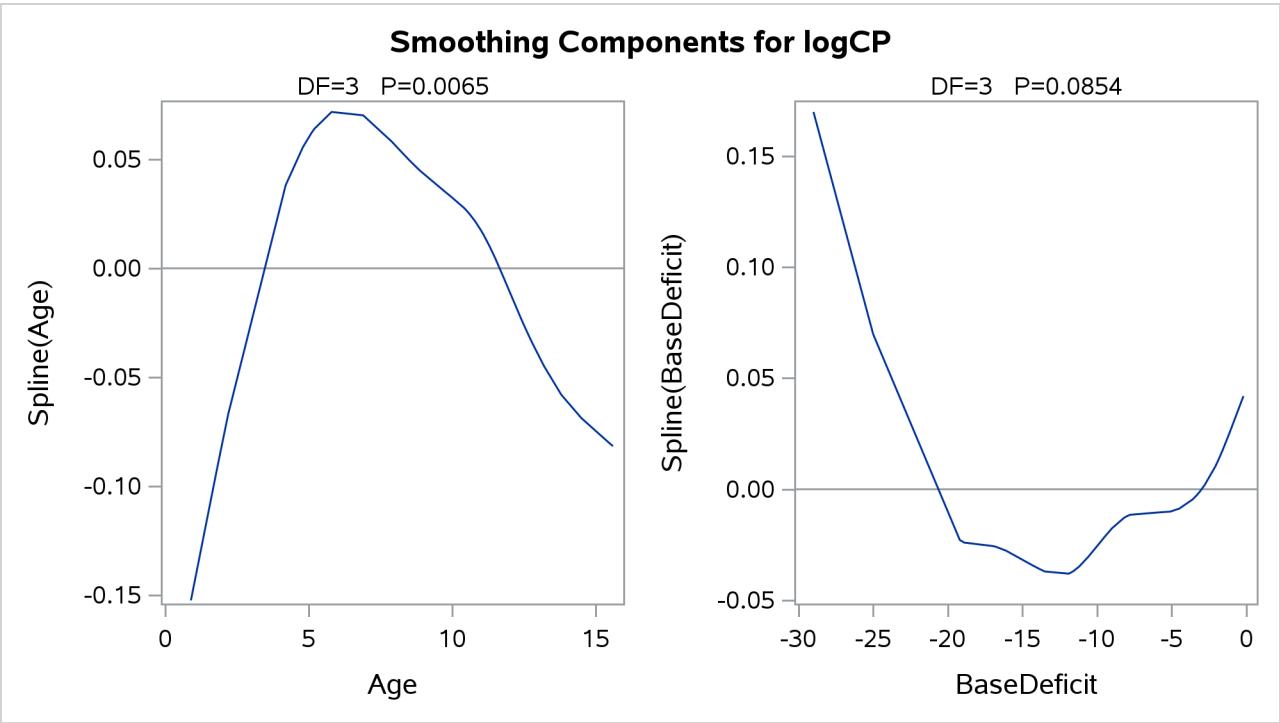

Syntax: GAM Procedure

The following statements are available in the GAM procedure:

PROC GAM < *options* > **;**
  CLASS *variable* < **(***options***)** > < *variable* < **(***options***)** > . . . > < / *options* > **;**
  MODEL *dependent* < **(***options***)** > **=** < **PARAM(***effects***)** > < *smoothing-effects* > < / *options* > **;**
  SCORE DATA= *SAS-data-set* **OUT=** *SAS-data-set* **;**
  OUTPUT OUT= *SAS-data-set* < *keyword*< **=***prefix* > . . . *keyword*< **=***prefix* > > **;**
  BY *variables* **;**
  FREQ *variable* **;**

The syntax of the GAM procedure is similar to that of other regression procedures in the SAS System. The PROC GAM and MODEL statements are required. The CLASS statement, if specified, must precede the MODEL statement. The CLASS and SCORE statements can appear multiple times; all other statements must appear only once.

The syntax for PROC GAM is described in the following sections in alphabetical order after the description of the PROC GAM statement.

PROC GAM Statement

PROC GAM < *options* > **;**

The PROC GAM statement invokes the procedure. Table 48.1 summarizes the *options* available in the PROC GAM statement.

Table 48.1 PROC GAM Statement Options

Option	Description
DATA=	Specifies the input SAS data set
DESCENDING	Reverses the sort order of all CLASS variables
ORDER=	Specifies the sort order for CLASS variables
PLOTS	Controls plots produced through ODS Graphics

You can specify the following *options*.

DATA=*SAS-data-set*
specifies the SAS data set to be read by PROC GAM. The default value is the most recently created data set.

DESCENDING
DESC
reverses the sort order of all classification variables (specified in the CLASS statement). If both the DESCENDING and ORDER= options are specified, PROC GAM orders the categories according to the ORDER= option and then reverses that order. This option has the same effect as the classi-

fication variable option DESCENDING in the CLASS statement and the response variable option DESCENDING in the MODEL statement.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of all classification variables (specified in the CLASS statement). This ordering determines which parameters in the model correspond to each level in the data. Note that the ORDER= option in the CLASS statement and the ORDER= response variable option in the MODEL statement override the ORDER= option in the PROC GAM statement.

PLOTS < (*global-plot-options*) > < = *plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >
controls the plots produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. Here are some examples:

```
plots=all

plots=components(commonaxes)

plots(unpack)=components(commonaxes clm)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc gam data=test plots(unpack)=components(commonaxes clm);
   model z=spline(x) spline(y);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

With ODS Graphics enabled, PROC GAM produces by default a panel of plots of partial prediction curves of smoothing components.

Global Plot Options

The *global-plot-options* apply to all plots generated by the GAM procedure, unless altered by a *specific-plot-option*.

UNPACK
separately displays the smoothing component plots that are by default presented in panels. Use this option if you want to access individual smoothing component plots within the panel.

Specific Plot Options

The following listing describes the specific plots and their *options*.

ALL
requests that all plots be produced.

NONE
suppresses all plots.

COMPONENTS | COMPONENT < (*components-options*) >
displays a panel of smoothing component plots. The following *components-options* are available:

ADDITIVE
produces the additive component plots for spline and loess effects. The additive component plots combine the linear trend and the nonparametric prediction for each spline or loess effect.

CLM
includes confidence limits in the smoothing component plots. By default, 95% confidence limits are produced, but you can change the significance level by specifying the ALPHA= option in the MODEL statement. Note that producing these limits can be computationally intensive for large data sets.

COMMONAXES
specifies that smoothing component plots use a common vertical axis. This enables you to visually judge relative effect size.

UNPACK
displays the smoothing components individually.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC GAM to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the GAM procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > < *variable* < (*options*) > . . . > < / *options* > ;

The CLASS statement names the classification variables to be used in the analysis. The CLASS statement must precede the MODEL statement. You can specify various *options* for each variable by enclosing them in parentheses after the variable name. You can also specify global *options* for the CLASS statement by placing them after a slash (/). Global *options* are applied to all the variables specified in the CLASS statement. If you specify more than one CLASS statement, the global *options* specified on any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the global *options*.

DESCENDING
DESC

reverses the sort order of the classification variable. If both the DESCENDING and ORDER= options are specified, PROC GAM orders the categories according to the ORDER= option and then reverses that order.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the categories of categorical variables. This ordering determines which parameters in the model correspond to each level in the data. When the default ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. The following table shows how PROC GAM interprets values of the ORDER= option.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For FORMATTED and INTERNAL, the sort order is machine-dependent. For more information on sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REF=*'level'* | *keyword*

specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*. The default is REF=LAST.

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

TRUNCATE< =*n* >

specifies the length n of CLASS variable values to use in determining CLASS variable levels. If you specify TRUNCATE without the length n, the first 16 characters of the formatted values are used. The default is to use the full formatted length of the CLASS variable. The TRUNCATE option is available only as a global option.

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a variable that provides frequencies for each observation in the DATA= data set. Specifically, if n is the value of the FREQ variable for a given observation, then that observation is used n times.

The analysis produced by using a FREQ statement reflects the expanded number of observations. You can produce the same analysis (without the FREQ statement) by first creating a new data set that contains the expanded number of observations. For example, if the value of the FREQ variable is 5 for the first observation, the first five observations in the new data set are identical. Each observation in the old data set is replicated n_i times in the new data set, where n_i is the value of the FREQ variable for that observation.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

The FREQ statement is not available when a loess smoother is included in the model.

MODEL Statement

MODEL *dependent* < **(***options***)** > **=** < **PARAM(***effects***)** > < *smoothing-effects* > < / *options* > **;**

MODEL *event/trials* **=** < **PARAM(***effects***)** > < *smoothing-effects* > < / *options* > **;**

The MODEL statement specifies the dependent variable and the independent effects you want to use in the model. Specify the independent parametric variables inside the parentheses of PARAM( ). The parametric variables can be either classification variables or continuous variables. Classification variables must be declared in a CLASS statement. Interactions between variables can also be included as parametric effects. Multiple PARAM() specifications are allowed in the MODEL statement. The syntax for the specification of effects is the same as for the GLM procedure (Chapter 53, "The GLM Procedure").

Only continuous variables can be specified in *smoothing-effects*. Any number of *smoothing-effects* can be specified, as follows:

Smoothing Effect	Meaning
SPLINE(*variable* < , **DF**=*number* >)	Fits a smoothing spline with the variable and with DF=*number*
LOESS(*variable* < , **DF**=*number* >)	Fits a local regression with the variable and with DF=*number*
SPLINE2(*variable1*, *variable2* < ,**DF**=*number* >)	Fits a bivariate thin-plate smoothing spline with *variable1* and *variable2* and with DF=*number*

The number specified in the DF= option must be positive. If you specify neither the DF= option nor the METHOD=GCV in the MODEL statement, then the default is DF=4. Note that for univariate spline and loess components, a degree of freedom is used by default to account for the linear portion of the model, so the value displayed in the "Fit Summary" and "Analysis of Deviance" tables will be one less than the value you specify.

Both parametric effects and smoothing effects are optional. If none are specified, a model that contains only an intercept is fitted.

If only parametric variables are present, PROC GAM fits a parametric linear model by using the terms inside the parentheses of PARAM( ). If only smoothing effects are present, PROC GAM fits a nonparametric additive model. If both types of effect are present, PROC GAM fits a semiparametric model by using the parametric effects as the linear part of the model.

Table 48.2 shows how to specify various models for a dependent variable y and independent variables x, x1, and x2. $s_i(\ ), i = 1, 2$ are nonparametric smooth functions.

Table 48.2 Syntax for Common GAM Models

Type of Model	Syntax for `model`	Mathematical Form
Parametric	`y=param(x1 x2)`	$E(Y\|X = x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$
Nonparametric	`y=spline(x)`	$E(Y\|X = x) = \beta_0 + \beta_1 x + s(x)$
Nonparametric	`y=loess(x)`	$E(Y\|X = x) = \beta_0 + \beta_1 x + s(x)$
Semiparametric	`y=spline(x1) param(x2)`	$E(Y\|X = x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + s(x_1)$
Additive	`y=spline(x1) spline(x2)`	$E(Y\|X = x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + s_1(x_1) + s_2(x_2)$
Thin-plate spline	`y=spline2(x1,x2)`	$E(Y\|X = x) = \beta_0 + s(x_1, x_2)$

Table 48.3 summarizes the *options* available in the MODEL statement.

Table 48.3 MODEL Statement Options

Option	Description
Response Variable Options	
DESCENDING	Reverses the order of the response categories
EVENT=	Specifies the event category for the binary response model
ORDER=	Specifies the sort order for the response variable
REFERENCE=	Specifies the reference category for the binary response model
Model Options	
ALPHA=	Specifies the significance level
ANODEV=	Specifies the method used to analyze smoothing effects
DIST=	Specifies the distribution family
EPSILON=	Specifies the convergence criterion for the backfitting algorithm
EPSSCORE=	Specifies the convergence criterion for the local scoring algorithm
ITPRINT	Produces an iteration summary table for the smoothing effects
MAXITER=	Specifies the maximum number of iterations for the backfitting algorithm
MAXITSCORE=	Specifies the maximum number of iterations for the local scoring algorithm
METHOD=	Specifies the method for selecting the value of the smoothing parameter
OFFSET=	Specifies an offset for the linear predictor

Response Variable Options

Response variable options determine how the GAM procedure models probabilities for binary data.

You can specify the following *options* by enclosing them in parentheses after the response variable. See the section "CLASS Statement" on page 3383 for more detail.

DESCENDING
DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC GAM orders the response categories according to the ORDER= option and then reverses that order.

EVENT='*category*' | *keyword*

specifies the event category for the binary response model. PROC GAM models the probability of the event category. You can specify the value (formatted, if a format is applied) of the event category in quotes, or you can specify one of the following *keywords*. The default is EVENT=FIRST.

FIRST

designates the first ordered category as the event.

LAST

designates the last ordered category as the event.

One of the most common sets of response levels is $\{0, 1\}$, with 1 representing the event for which the probability is to be modeled. Consider the example where Y takes the value 1 and 0 for event and nonevent, respectively, and X is the explanatory variable. By default, PROC GAM models the probability that Y = 0. To model the probability that Y = 1 , specify the following MODEL statement:

```
model Y (event='1') = X;
```

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the response variable. By default, ORDER=FORMATTED. When ORDER=FORMATTED, the values of numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC GAM run or in the DATA step that created the data set), are ordered by their internal (numeric) value. If you specify the ORDER= option in the MODEL statement and the ORDER= option in the CLASS statement, the former takes precedence. The following table shows the interpretation of the ORDER= values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

For the FORMATTED and INTERNAL values, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | *keyword*

REF='*category*' | *keyword*

specifies the reference category for the binary response model. Specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotes, or you can specify one of the following *keywords*:

FIRST designates the first ordered category as the reference.

LAST designates the last ordered category as the reference. This is the default.

The REFERENCE= option is ignored when the EVENT= option is present.

Model Options

ALPHA=*number*

specifies the significance level α of the confidence limits on the final nonparametric component estimates when you request confidence limits to be included in the output data set. Specify *number* as a value between 0 and 1. The default value is 0.05. See the section "OUTPUT Statement" on page 3389 for more information about the OUTPUT statement.

ANODEV=*type*

specifies the *type* of method to be used to produce the "Analysis of Deviance" table for smoothing effects. The available choices are as follows:

REFIT specifies that PROC GAM perform χ^2 tests by fitting nested GAM models. This is the default choice if you do not specify the ANODEV= option. This choice requires fitting separate GAM models where one smoothing term is omitted from each model.

NOREFIT specifies that PROC GAM perform approximate tests of smoothing effects. To test each smoothing effect, a weighted least squares model is fitted to the remaining parametric part of the model while keeping other nonlinear smoothers fixed. For details, see Hastie (1991). This choice requires only a single GAM fitting to be performed, which reduces the time of the procedure.

NONE requests that the procedure not produce the "Analysis of Deviance" table for smoothing effects.

DIST=*distribution-id*

LINK=*distribution-id*

specifies the distribution family used in the model. The choices for *distribution-id* are displayed in Table 48.5. See "Distribution Family and Canonical Link" on page 3401 for more information.

Table 48.5 Distribution Families for GAM Models

DIST=	Distribution	Link Function	Response Data Type
GAUSSIAN | GAUS | NORM	Normal (Gaussian)	Identity	Continuous variables
BINOMIAL | LOGI | BIN	Binomial	Logit	Binary variables
POISSON | POIS | LOGL	Poisson	Log	Nonnegative discrete variables
GAMMA | GAMM	Gamma	Negative reciprocal	Positive continuous variables
IGAUSSIAN | IGAU | INVG	Inverse Gaussian	Squared reciprocal	Positive continuous variables

Canonical link functions are used with those distributions. Although alternative links are possible theoretically, the final fit of nonparametric regression models is relatively insensitive to the precise choice of link functions. Therefore, only the canonical link for each distribution family is implemented

in PROC GAM. The loess smoother is not available for DIST=BINOMIAL when the number of trials is greater than 1.

EPSILON=*number*
specifies the convergence criterion for the backfitting algorithm. The default value is 1E–8.

EPSSCORE=*number*
specifies the convergence criterion for the local scoring algorithm. The default value is 1E–8.

ITPRINT
produces an iteration summary table for the smoothing effects when doing backfitting and local scoring.

MAXITER=*number*
specifies the maximum number of iterations for the backfitting algorithm. The default value is 50.

MAXITSCORE=*number*
specifies the maximum number of iterations for the local scoring algorithm. The default value is 100.

METHOD=GCV
specifies that the value of the smoothing parameter should be selected by generalized cross validation. If you specify both METHOD=GCV and the DF= option for some smoothing effects, the user-specified DF values are used for those effects in which the DF= option is specified. The GCV criterion is used for determining smoothness of the remaining effects. See the section "Selection of Smoothing Parameters" on page 3398 for more details on the GCV method.

OFFSET=*variable*
specifies an offset for the linear predictor. An offset plays the role of a predictor whose coefficient is known to be 1. For example, you can use an offset in a Poisson model when counts have been obtained in time intervals of different lengths. With a log link function, you can model the counts as Poisson variables with the logarithm of the time interval as the offset variable. The offset variable cannot appear in the CLASS statement or elsewhere in the MODEL statement.

OUTPUT Statement

OUTPUT OUT= *SAS-data-set* < *keyword*< **=***prefix* > . . . *keyword*< **=***prefix* > > **;**

The OUTPUT statement creates a new SAS data set that contains diagnostic measures calculated after fitting the model.

All the variables in the original data set are included in the new data set, along with new variables that are created with *keywords* specified in the OUTPUT statement. These new variables contain the values of a variety of statistics and diagnostic measures that are calculated for each observation in the data set. If no *keywords* are present, the OUT= data set contains only the original data set and predicted values. The predicted values include the linear predictor for the response and the prediction for each smoothing term in the model. When you specify a distribution family with the DIST= or LINK= option in the MODEL statement, predicted response values after applying the inverse link function are also included. Predicted values are computed for observations that have missing response values, nonmissing values in the explanatory variables, and values of the smoothing variables that are within the smoothing ranges of the fitted model.

You can specify the following *options* in the OUTPUT statement:

OUT=*SAS-data-set*
specifies the name of the new data set to contain the diagnostic measures. This option is required.

keyword < =*prefix* >
specifies the statistics to include in the output data set. The *keywords* and the statistics they represent are as follows:

PREDICTED predicted values for each smoothing component and overall predicted values on the response scale at design points. The prediction for each spline or loess term is only for the nonlinear component of each smoother.

LINP linear prediction values on the link scale at design points

UCLM upper confidence limits for each predicted smoothing component

LCLM lower confidence limits for each predicted smoothing component

ADIAG diagonal element of the hat matrix associated with the observation for each smoothing spline component

RESIDUAL residual standardized by its weights

STD standard deviation of the prediction for each smoothing component

ALL all statistics in this list

The names of the new variables that contain the statistics are formed by concatenating a *prefix* and the corresponding variable names. If you do not specify a *prefix*, the names are formed by using default prefixes listed in the following table:

Keyword	Prefix
PRED	**P_**
LINP	**LINP_**
UCLM	**UCLM_**
LCLM	**LCLM_**
ADIAG	**ADIAG_**
RESID	**R_**
STD	**STD_** (for spline)
	STDP_ (for loess)

For example, suppose that you have a dependent variable y and an independent smoothing variable x. If you specify the keywords PRED=MyP_ and ADIAG=MyA_, the new variables in the output SAS data set are MyP_y, MyP_x, and MyA_x. If you specify the keywords PRED and ADIAG without prefixes, the new variables are P_y, P_x, and ADIAG_x.

SCORE Statement

SCORE DATA= *SAS-data-set* **OUT=** *SAS-data-set* **;**

The SCORE statement calculates predicted values for a new data set. All the variables in the DATA= data set are included in the OUT= data set, along with the predicted values. The predicted values consist of predicted responses after the inverse link function transformation, predicted values of all smoothing terms, and predicted values on the link scale. Predicted values are computed for observations with missing response values whose values of the specified explanatory variables are nonmissing, and whose values of the specified smoothing variables are within the smoothing ranges of the fitted model. The predicted variables use the same naming convention as the OUTPUT statement. If you have multiple data sets to score, you can specify multiple SCORE statements.

The following *options* must be specified in the SCORE statement:

DATA=*SAS-data-set*
specifies an input SAS data set containing all the variables included in independent effects in the MODEL statement. The predicted response is computed for each observation in the SCORE DATA= data set.

OUT=*SAS-data-set*
specifies the name of the SAS data set to contain the predictions.

Details: GAM Procedure

Missing Values

When fitting a model, PROC GAM excludes any observation with missing values for an explanatory variable, offset variable, or dependent variable. However, if only the response is missing, predicted values can be computed and output to a data set by using the OUTPUT or SCORE statement.

Nonparametric Regression

Nonparametric regression relaxes the usual assumption of linearity and enables you to explore the data more flexibly, uncovering structure in the data that might otherwise be missed.

However, many forms of nonparametric regression do not perform well when the number of independent variables in the model is large. The sparseness of data in this setting causes the variances of the estimates to be unacceptably large unless the sample size is extremely large. The problem of rapidly increasing variance for increasing dimensionality is sometimes referred to as the “curse of dimensionality.” Interpretability is another problem with nonparametric regression based on kernel and smoothing spline estimates. These estimates contain information about the relationship between the dependent and independent variables, and the information is often difficult to comprehend.

To overcome these difficulties, additive models were proposed by some researchers, for example, Stone (1985). These models estimate an additive approximation to the multivariate regression function. The benefits of an additive approximation are at least twofold. First, since each of the individual additive terms is estimated by using a univariate smoother, the curse of dimensionality is avoided, at the cost of not being able to approximate universally. Second, estimates of the individual terms explain how the dependent variable changes with the corresponding independent variables.

To extend the additive model to a wide range of distribution families, Hastie and Tibshirani (1990) proposed generalized additive models. These models enable the mean of the dependent variable to depend on an additive predictor through a nonlinear link function. The models permit the response probability distribution to be any member of the exponential family of distributions. Many widely used statistical models belong to this general class; they include additive models for Gaussian data, nonparametric logistic models for binary data, and nonparametric log-linear models for Poisson data.

Additive Models and Generalized Additive Models

This section describes the methodology and the fitting procedure behind generalized additive models.

Let Y be a response random variable and $X_1, X_2, \ldots, X_p$ be a set of predictor variables. A regression procedure can be viewed as a method for estimating the expected value of Y given the values of $X_1, X_2, \ldots, X_p$. The standard linear regression model assumes a linear form for the dependency of Y on X:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

where $E(\epsilon) = 0$ and $\mathrm{Var}(\epsilon) = \sigma^2$. Given a sample, estimates of $\beta_0, \beta_1, \ldots, \beta_p$ are usually obtained by the least squares method.

The additive model generalizes the linear model by modeling the dependency as

$$Y = s_0 + s_1(X_1) + s_2(X_2) + \cdots + s_p(X_p) + \epsilon$$

where $s_j(X)$, $j = 1, 2, \ldots, p$, are smooth functions, $E(\epsilon) = 0$ and $\mathrm{Var}(\epsilon) = \sigma^2$.

In order to be estimable, the smooth functions s_i have to satisfy standardized conditions such as $E\left(s_j(X_j)\right) = 0$. These functions are not given a parametric form but instead are estimated in a nonparametric fashion.

While traditional linear models and additive models can be used in most statistical data analysis, there are types of problems for which they are not appropriate. For example, the normal distribution might not be adequate for modeling discrete responses such as counts or bounded responses such as proportions.

Generalized additive models address these difficulties, extending additive models to many other distributions besides just the normal. Thus, generalized additive models can be applied to a much wider range of data analysis problems.

Like generalized linear models, generalized additive models consist of a random component, an additive component, and a link function relating the two components. The response Y, the random component, is assumed to have exponential family density

$$f_Y(y; \theta, \phi) = \exp\left\{\frac{y\theta - b(\theta)}{a(\phi)} + c(y, \phi)\right\}$$

where θ is called the natural parameter and ϕ is the scale parameter. The mean of the response variable μ is related to the set of covariates $X_1, X_2, \ldots, X_p$ by a link function g. The quantity

$$\eta = s_0 + \sum_{j=1}^{p} s_j(X_j)$$

defines the additive component, where $s_1(\,), \ldots, s_p(\,)$ are smooth functions, and the relationship between μ and η is defined by $g(\mu) = \eta$. The most commonly used link function is the canonical link, for which $\eta = \theta$.

Generalized additive models and generalized linear models can be applied in similar situations, but they serve different analytic purposes. Generalized linear models emphasize estimation and inference for the parameters of the model, while generalized additive models focus on exploring data nonparametrically. Generalized additive models are more suitable for exploring the data and visualizing the relationship between the dependent variable and the independent variables.

Forms of Additive Models

Suppose that y is a continuous variable and x1 and x2 are two explanatory variables of interest. To fit an additive model, you can use a MODEL statement similar to that used in many regression procedures in the SAS System:

```
model y = spline(x1) spline(x2);
```

This model statement requires the procedure to fit the following model:

$$\eta(x_1, x_2) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + s_1(x_1) + s_2(x_2)$$

where the $s_i(\,)$ terms denote nonparametric spline functions of the respective explanatory variables.

The GAM procedure can fit semiparametric models. The following MODEL statement assumes a linear relation with x1 and an unknown functional relation with x2:

```
model y = param(x1) spline(x2);
```

If you want to fit a model containing a functional two-way interaction between x1 and x2, you can use the following MODEL statement:

```
model y = spline2(x1,x2);
```

In this case, the GAM procedure fits a model equivalent to that of PROC TPSPLINE.

Estimates from PROC GAM

PROC GAM provides the capability to fit both nonparametric and semiparametric models. So that you can better understand the underlying trend of any given factor, PROC GAM separates the linear trend from any general nonparametric trend during the fitting as well as in the final report. This makes it easy to determine whether the significance of a smoothing variable is associated with a simple linear trend or a more complicated pattern.

For example, suppose you want to fit a semiparametric model as

$$y = \alpha_0 + \alpha_1 z + f_1(x_1) + f_2(x_2)$$

The GAM estimate for this model is

$$y = \widehat{\alpha_0} + \widehat{\alpha_1} z + \widehat{\beta_1} x_1 + \widehat{\beta_2} x_2 + \widehat{s_1}(x_1) + \widehat{s_2}(x_2)$$

where $\widehat{s_1}$ and $\widehat{s_2}$ are linear-adjusted nonparametric estimates of the f_1 and f_2 effects. The p-values for $\widehat{\alpha_0}$, $\widehat{\alpha_1}$, $\widehat{\beta_1}$, and $\widehat{\beta_2}$ are reported in the parameter estimates table. $\widehat{\beta_1}$ and $\widehat{\beta_2}$ are the estimates labeled Linear(x1) and Linear(x2) in the table. The p-values for $\widehat{s_1}$ and $\widehat{s_2}$ are reported in the analysis of deviance table.

Only $\widehat{s_1}$, $\widehat{s_2}$, and $\hat{y}$ are output to the output data set, with the corresponding variable names P_x1, P_x2, and P_y. For Gaussian data, the complete marginal prediction for variable x1 is:

$$\widehat{\beta_1} x_1 + \mathrm{P_}x_1$$

If the additive component plots are requested by the ADDITIVE suboption, the additive component for variable x2 is computed as:

$$\widehat{\beta_2}(x_2 - \bar{x}_2) + \mathrm{P_}x_2$$

where $\bar{x}_2$ is the mean for variable x2.

Backfitting and Local Scoring Algorithms

Much of the development and notation in this section follows Hastie and Tibshirani (1986).

Additive Models

Consider the estimation of the smoothing terms $s_0, s_1(\,), \ldots, s_p(\,)$ in the additive model

$$\eta(X) = s_0 + \sum_{j=1}^{p} s_j(X_j)$$

where $E\left(s_j(X_j)\right) = 0$ for every j. Since the algorithm for additive models is the basis for fitting generalized additive models, the algorithm for additive models is discussed first.

Many ways are available to approach the formulation and estimation of additive models. The backfitting algorithm is a general algorithm that can fit an additive model with any regression-type fitting mechanisms.

Define the kth set of partial residuals as

$$R_k = Y - s_0 - \sum_{j \neq k} s_j(X_j)$$

then $E(R_k|X_k) = s_k(X_k)$. This observation provides a way to estimate each smoothing function $s_k(\,)$ given estimates $\{\hat{s}_j(\,), j \neq k\}$ for all the others. The resulting iterative procedure is known as the backfitting algorithm (Friedman and Stuetzle 1981). The following formulation is taken from Hastie and Tibshirani (1986).

The Backfitting Algorithm

The unweighted form of the backfitting algorithm is as follows:

1. **Initialization:**
   $s_0 = E(Y), s_1^{(1)} = s_2^{(1)} = \cdots = s_p^{(1)} = 0, m = 0$
2. **Iterate:**
   $m = m + 1;$
   for $j = 1$ to p do:
   $R_j = Y - s_0 - \sum_{k=1}^{j-1} s_k^{(m)}(X_k) - \sum_{k=j+1}^{p} s_k^{(m-1)}(X_k);$
   $s_j^{(m)} = E(R_j|X_j);$
3. **Until:**
   $\mathrm{RSS} = \frac{1}{n}\left\|Y - s_0 - \sum_{j=1}^{p} s_j^{(m)}(X_j)\right\|^2$ fails to decrease, or satisfies the convergence criterion.

In the preceding notation, $s_j^{(m)}(\,)$ denotes the estimate of $s_j(\,)$ at the mth iteration. It can be shown that with many smoothers (including linear regression, univariate and bivariate splines, and combinations of these), RSS never increases at any step. This implies that the algorithm always converges (Hastie and Tibshirani 1986). Note, however, that for distributions other than Gaussian, numerical instabilities with weights can cause convergence problems. Even when the algorithm converges, the individual functions need not be unique, since dependence among the covariates can lead to more than one representation for the same fitted surface.

A weighted backfitting algorithm has the same form as for the unweighted case, except that the smoothers are weighted. In PROC GAM, weights are used with non-Gaussian data in the local scoring procedure described later in this section.

The GAM procedure uses the following condition as the convergence criterion for the backfitting algorithm:

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{p}\left(s_j^{(m-1)}(\mathbf{X}_{ij}) - s_j^{(m)}(\mathbf{X}_{ij})\right)^2}{1 + \sum_{i=1}^{n}\sum_{j=1}^{p}\left(s_j^{(m-1)}(\mathbf{X}_{ij})\right)^2} \leq \epsilon$$

where $\epsilon = 10^{-8}$ by default; you can change this with the EPSILON= option in the MODEL statement.

Generalized Additive Models

The algorithm described so far fits only additive models. The algorithm for generalized additive models is a little more complicated. Generalized additive models extend generalized linear models in the same manner that additive models extend linear regression models—that is, by replacing the form $\alpha + \sum_j \beta_j \mathbf{x}_j$ with the additive form $\alpha + \sum_j f_j(\mathbf{x}_j)$. See "Generalized Linear Models Theory" on page 3655 in Chapter 51, "The GENMOD Procedure," for more information.

PROC GAM fits generalized additive models by using a modified form of adjusted dependent variable regression, as described for generalized linear models in McCullagh and Nelder (1989), with the additive predictor taking the role of the linear predictor. Hastie and Tibshirani (1986) call this the *local scoring algorithm*. Important components of this algorithm depend on the link function for each distribution, as shown in the following table.

Distribution	**Link**	**Adjusted Dependent (z)**	**Weights (w)**
Normal	μ	y	1
Binomial	$\log\left(\frac{\mu}{n-\mu}\right)$	$\eta + (y-\mu)/n\mu(1-\mu)$	$n\mu(1-\mu)$
Gamma	$-1/\mu$	$\eta + (y-\mu)/\mu^2$	μ^2
Poisson	$\log(\mu)$	$\eta + (y-\mu)/\mu$	μ
Inverse Gaussian	$1/\mu^2$	$\eta - 2(y-\mu)/\mu^3$	$\mu^3/4$

Once the distribution and hence these quantities are defined, the local scoring algorithm proceeds as follows.

The General Local Scoring Algorithm

1. **Initialization:**
   $s_i = g(E(y)), s_1^0 = s_2^0 = \cdots = s_p^0 = 0, m = 0$

2. **Iterate:**
   $m = m + 1;$
   Form the predictor $\boldsymbol{\eta}$, mean $\boldsymbol{\mu}$, weights $\mathbf{w}$, and adjusted dependent variable $\mathbf{z}$ based on their corresponding values from the previous iteration:

$$
\begin{aligned}
\eta_i^{(m-1)} &= s_0 + \sum_{j=1}^{p} s_j^{(m-1)}(x_{ij}) \\
\mu_i^{(m-1)} &= g^{-1}\left(\eta_i^{(m-1)}\right) \\
w_i &= \left(V_i^{(m-1)}\right)^{-1} \cdot \left[\left(\frac{\partial\mu}{\partial\eta}\right)_i^{(m-1)}\right]^2 \\
z_i &= \eta_i^{(m-1)} + \left(y_i - \mu_i^{(m-1)}\right) \cdot \left(\frac{\partial\eta}{\partial\mu}\right)_i^{(m-1)}
\end{aligned}
$$

   where $V_i^{(m-1)}$ is the variance of Y at $\mu_i^{(m-1)}$. Fit an additive model to $\mathbf{z}$ by using the backfitting algorithm with weights $\mathbf{w}$ to obtain estimated functions $s_j^{(m)}(\,), j = 1, \ldots, p;$

3. **Until:**
   The convergence criterion is satisfied or the deviance fails to decrease. The deviance is an extension to generalized linear models of the RSS; see "Goodness of Fit" on page 3662 in Chapter 51, "The GENMOD Procedure," for a definition.

The GAM procedure uses the following condition as the convergence criterion for local scoring:

$$\frac{\sum_{i=1}^{n} w_i \sum_{j=1}^{p} \left(s_j^{(m-1)}(\mathbf{X}_{ij}) - s_j^{(m)}(\mathbf{X}_{ij}) \right)^2}{\sum_{i=1}^{n} w_i \left(1 + \sum_{j=1}^{p} \left(s_j^{(m-1)}(\mathbf{X}_{ij}) \right)^2 \right)} \leq \epsilon^s$$

where $\epsilon^s = 10^{-8}$ by default; you can change this with the EPSSCORE= option in the MODEL statement.

The estimating procedure for generalized additive models consists of two loops. Inside each step of the local scoring algorithm (outer loop), a weighted backfitting algorithm (inner loop) is used until convergence or until the RSS fails to decrease. Then, based on the estimates from this weighted backfitting algorithm, a new set of weights is calculated and the next iteration of the scoring algorithm starts. The scoring algorithm stops when the convergence criterion is satisfied or the deviance of the estimates stops decreasing.

Smoothers

You can specify three types of smoothers in the MODEL statement:

- SPLINE(x) specifies a cubic smoothing spline term for variable x
- LOESS(x) specifies a loess term for variable x
- SPLINE2(x1, x2) specifies a thin-plate smoothing spline term for variables x1 and x2

A smoother is a tool for summarizing the trend of a response measurement Y as a function of one or more predictor measurements $X_1, \ldots, X_p$. It produces an estimate of the trend that is less variable than Y itself. An important property of a smoother is its nonparametric nature. It does not assume a rigid form for the dependence of Y on $X_1, \ldots, X_p$. This section gives a brief overview of the smoothers that can be used with the GAM procedure.

Cubic Smoothing Spline

A smoothing spline is the solution to the following optimization problem: among all functions $\eta(x)$ with two continuous derivatives, find one that minimizes the penalized least square

$$\sum_{i=1}^{n} (y_i - \eta(x_i))^2 + \lambda \int_a^b \left(\eta''(t) \right)^2 \mathrm{d}t$$

where λ is a fixed constant and $a \leq x_1 \leq \cdots \leq x_n \leq b$. The first term measures closeness to the data while the second term penalizes curvature in the function. It can be shown that there exists an explicit, unique minimizer, and that minimizer is a natural cubic spline with knots at the unique values of x_i.

The value $\lambda/(1+\lambda)$ is the *smoothing parameter*. When λ is large, the smoothing parameter is close to 1, producing a smoother curve; small values of λ, corresponding to smoothing parameters near 0, are apt to produce rougher curves, more nearly interpolating the data.

Local Regression

Local regression was proposed by Cleveland, Devlin, and Grosse (1988). The idea of local regression is that at a predictor x, the regression function $\eta(x)$ can be locally approximated by the value of a function in some specified parametric class. Such a local approximation is obtained by fitting a regression surface to the data points within a chosen neighborhood of the point x. A weighted least squares algorithm is used to fit linear functions of the predictors at the centers of neighborhoods. The radius of each neighborhood is chosen so that the neighborhood contains a specified percentage of the data points. The smoothing parameter for the local regression procedure, which controls the smoothness of the estimated curve, is the fraction of the data in each local neighborhood. Data points in a given local neighborhood are weighted by a smooth decreasing function of their distance from the center of the neighborhood. See Chapter 78, "The LOESS Procedure," for more details.

Thin-Plate Smoothing Spline

The thin-plate smoothing spline is a multivariate version of the cubic smoothing spline. The theoretical foundations for the thin-plate smoothing spline are described in Duchon (1976, 1977); Meinguet (1979). The smoothing parameter for the thin-plate smoothing spline smoother is the parameter that controls the smoothness penalty. When the smoothing parameter is close to 0, the fit is close to an interpolation. When the smoothing parameter is very large, the fit is a smooth surface. Further results and applications are given in Wahba and Wendelberger (1980). See Chapter 125, "The TPSPLINE Procedure," for more details.

Selection of Smoothing Parameters

CV and GCV

The smoothers discussed here have a single smoothing parameter. In choosing the smoothing parameter, cross validation can be used. Cross validation works by leaving points (x_i, y_i) out one at a time, estimating the squared residual for smooth function at x_i based on the remaining $n-1$ data points, and choosing the smoother to minimize the sum of those squared residuals. This mimics the use of training and test samples for prediction. The cross validation function is defined as

$$\mathrm{CV}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{\eta}_{\lambda}^{(-i)}(x_i)\right)^2$$

where $\hat{\eta}_{\lambda}^{(-i)}(x_i)$ indicates the fit at x_i, computed by leaving out the ith data point. The quantity $n\mathrm{CV}(\lambda)$ is sometimes called the prediction sum of squares, or PRESS (Allen 1974).

All of the smoothers fit by the GAM procedure can be formulated as a linear combination of the sample responses

$$\hat{\eta}(x) = \mathbf{A}(\lambda)\mathbf{y}$$

for some matrix $\mathbf{A}(\lambda)$, which depends on λ. (The matrix $\mathbf{A}(\lambda)$ depends on x and the sample data as well, but this dependence is suppressed in the preceding equation.) Let a_{ii} be the ith diagonal element of $\mathbf{A}(\lambda)$. Then the CV function can be expressed as

$$\mathrm{CV}(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{(y_i - \hat{\eta}_{\lambda}(x_i))}{1-a_{ii}}\right)^2$$

In most cases, it is very time-consuming to compute the quantity a_{ii} individually. To solve this computational problem, Wahba (1990) has proposed the generalized cross validation function (GCV) that can be used to solve a wide variety of problems involving selection of a parameter to minimize the prediction risk.

The GCV function is defined as

$$\mathrm{GCV}(\lambda) = \frac{n \sum_{i=1}^{n} (y_i - \hat{\eta}_\lambda(x_i))^2}{(n - \mathrm{Trace}(\mathbf{A}(\lambda)))^2}$$

The GCV formula simply replaces the a_{ii} with $\mathrm{Trace}(\mathbf{A}(\lambda))/n$. Therefore, it can be viewed as a weighted version of CV. In most of the cases of interest, GCV is closely related to CV but much easier to compute. Specify the METHOD=GCV option in the MODEL statement in order to use the GCV function to choose the smoothing parameters.

Degrees of Freedom

The estimated GAM model can be expressed as

$$\hat{\eta}(X) = \hat{s}_0 + \sum_{j=1}^{p} \mathbf{A}_j(\lambda_j) Y$$

Because the weights are calculated based on previous iteration during the local scoring iteration, the matrices $\mathbf{A}_j$ might depend on Y for non-Gaussian data. However, for the final iteration, the $\mathbf{A}_j$ matrix for the spline smoothers has the same role as the projection matrix in linear regression; therefore, nonparametric degrees of freedom (DF) for the *j*th spline smoother can be defined as

$$\mathrm{DF}(j\,\mathrm{th\ spline\ smoother}) = \mathrm{Trace}(\mathbf{A}_j(\lambda_j))$$

For loess smoothers $\mathbf{A}_j$ is not symmetric and so is not a projection matrix. In this case PROC GAM uses

$$\mathrm{DF}(j\,\mathrm{th\ loess\ smoother}) = \mathrm{Trace}(\mathbf{A}_j(\lambda_j)'\mathbf{A}_j(\lambda_j))$$

The GAM procedure gives you the option of specifying the degrees of freedom for each individual smoothing component. If you choose a particular value for the degrees of freedom, then during every local scoring iteration the procedure will search for a corresponding smoothing parameter lambda that yields the specified value or comes as close as possible. The final estimate for the smoother during this local scoring iteration will be based on this lambda. Note that for univariate spline and loess components, an additional degree of freedom is used by default to account for the linear portion of the model, so the value displayed in the “Fit Summary” and “Analysis of Deviance” tables will be one less than the value you specify.

Confidence Intervals for Smoothers

Buja, Hastie, and Tibshirani (1989) showed that each smoothing function estimate from the backfitting algorithm is the result of a linear mapping applied to the working response, if the backfitting algorithm converges.

The smoothing function estimate can be expressed as

$$\hat{s}_j(\mathbf{x}_j) = \mathbf{H}_j \mathbf{z}$$

where $\mathbf{x}_j$ is the jth covariate and $\mathbf{z}$ is the adjusted dependent variable that is formed in the local scoring algorithm. If the errors are independent and identically distributed, then

$$\mathrm{Cov}(\hat{s}_j) = \sigma^2 \mathbf{H}_j \mathbf{H}_j'$$

where $\sigma^2 = \mathrm{Var}(\mathbf{z})$.

However, direct computation of $\mathbf{H}_j$ is formidable within the backfitting framework. Hastie and Tibshirani (1990) proposed using each individual smoothing matrix $\mathbf{A}_j(\lambda_j)$ as a substitute for the linear operator $\mathbf{H}_j$ when computing confidence intervals. In the GAM procedure, curvewise confidence intervals for smoothing splines and pointwise confidence intervals for loess are provided in the output data set.

Curvewise Confidence Interval for Smoothing Spline Smoothers

Viewing the spline model as a Bayesian model, Wahba (1983) proposes Bayesian confidence intervals for smoothing spline estimates as:

$$\hat{s}_\lambda(x_i) \pm z_{\alpha/2}\sqrt{\hat{\mathbf{V}}_{ii}(\lambda)}$$

where $\hat{\mathbf{V}}_{ii}(\lambda)$ is the ith diagonal element of the Bayesian posterior covariance matrix $\hat{\mathbf{V}}$ and $z_{\alpha/2}$ is the $1-\alpha/2$ quantile of the standard normal distribution. The confidence intervals are interpreted as intervals "across the function" as opposed to pointwise intervals.

Suppose that you fit a spline estimate to experimental data that consist of a true function f and a random error term ϵ_i. In repeated experiments, it is likely that about $100(1-\alpha)\%$ of the confidence intervals cover the corresponding true values, although some values are covered every time and other values are not covered by the confidence intervals most of the time. This effect is more pronounced when the true response curve or surface has small regions of particularly rapid change.

In the GAM procedure, let the smoothing matrix for the nonlinear part of the jth spline term be $\tilde{\mathbf{A}}_j$ after the linear part is separated out from $\mathbf{A}_j(\lambda)$. The Bayesian posterior variance for the nonlinear part is computed as

$$\hat{\mathbf{V}}_j = \hat{\phi}\tilde{\mathbf{A}}_j \mathbf{W}^{-1}$$

where $\hat{\phi}$ is the dispersion parameter estimate and $\mathbf{W}$ is the weight matrix from the final local scoring iteration. If you specify UCLM, LCLM, ADIAG, and STD options in the OUTPUT statement, the statistics are derived based on $\hat{\mathbf{V}}_j$.

When you request both the ADDITIVE and CLM suboptions in the PLOTS=COMPONENTS option, each of the smoothing component plots displays a confidence band for the total contribution of each smoothing

spline smoother. The confidence band is derived from the total variance that is contributed by both linear and nonlinear parts by the *j*th term

$$\hat{\phi}\left(\mathbf{x}_j'(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{x}_j + \tilde{\mathbf{A}}_j\mathbf{W}^{-1}\right)$$

Pointwise Confidence Interval for Loess Smoothers

As shown in Cleveland, Devlin, and Grosse (1988), the smoothing matrix $\mathbf{A}(\lambda)$ for a loess smoother is asymmetric. The confidence intervals are computed as follows:

$$\hat{s}_\lambda(x_i) \pm z_{\alpha/2}\sqrt{\hat{\mathbf{V}}_{ii}(\lambda)}$$

where $\hat{\mathbf{V}}_{ii}(\lambda)$ is the *i*th diagonal element of the covariance matrix $\hat{\mathbf{V}}$ and $z_{\alpha/2}$ is the $1-\alpha/2$ quantile of the standard normal distribution.

In the GAM procedure, let the smoothing matrix for the nonlinear part of the *j*th loess term be $\tilde{\mathbf{A}}_j$ after the linear part is separated out from $\mathbf{A}_j(\lambda)$. The covariance matrix for the nonlinear part is then

$$\hat{\mathbf{V}}_j = \hat{\phi}\tilde{\mathbf{A}}_j\mathbf{W}^{-1}\tilde{\mathbf{A}}_j'$$

where $\hat{\phi}$ is the dispersion parameter estimate and $\mathbf{W}$ is the weight matrix from the final local scoring iteration. If you specify UCLM, LCLM, and STD options in the OUTPUT statement, the statistics are derived based on $\hat{\mathbf{V}}_j$.

When you request both the ADDITVE and CLM suboptions in the PLOTS=COMPONENTS option, each of the smoothing component plots displays confidence intervals for total prediction of each loess smoother. The confidence intervals are derived from the total variance that is contributed by both the linear and nonlinear parts by the *j*th term

$$\hat{\phi}\left(\mathbf{x}_j'(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{x}_j + \tilde{\mathbf{A}}_j\mathbf{W}^{-1}\tilde{\mathbf{A}}_j'\right)$$

Distribution Family and Canonical Link

In general, there is not just one reasonable link function for a given response variable distribution. For parametric models, the choice of link function can lead to substantively different estimates and tests. However, the inherent flexibility of nonparametric models makes them less likely to be sensitive to the precise choice of link function. Thus, for simplicity and computational efficiency, the GAM procedure uses only the canonical link for each distribution, as discussed in the following sections.

The Gaussian Model

For a Gaussian model, the link function is the identity function, and the generalized additive model is the same as the additive model. The Gaussian model is selected by default or when you specify the DIST=GAUSSIAN option in the MODEL statement.

The Binomial Model

The binomial model is selected by specifying the DIST=BINOMIAL option in the MODEL statement. A binomial response model assumes that the proportion of successes Y is such that Y has a $\mathrm{Bi}(n, p(x))$ distribution. $\mathrm{Bi}(n, p(x))$ refers to the binomial distribution with the parameters n and $p(x)$. Often the data are binary, in which case $n = 1$. The canonical link is

$$g(p) = \log \frac{p}{n-p} = \eta$$

By default, PROC GAM models the probability of the response level with the *lower ordered value*. Ordered values are assigned to response levels in ascending sorted order and are displayed in the "Response Profiles" table. For binary data, if your event category has a higher Ordered Value, then by default the nonevent is modeled. The effect of modeling the nonevent is to change the signs of the estimated coefficients for linear terms in the model for the event. You can change which probability is modeled by specifying the EVENT=, DESCENDING, or ORDER= response variable options in the MODEL statement.

The Poisson Model

The Poisson model is selected by specifying the DIST=POISSON option in the MODEL statement. The link function for the Poisson model is the log function. Assuming that the mean of the Poisson distribution is $\mu(x)$, the dependence of $\mu(x)$ and independent variables $x_1, \ldots, x_k$ is

$$g(\mu) = \log(\mu) = \eta$$

The Gamma Model

The gamma model is selected by specifying the DIST=GAMMA option in the MODEL statement. Let the mean of the gamma distribution be $\mu(x)$. The canonical link function for the gamma distribution is $-1/\mu(x)$. Note that this link function is the negative of the default link function in PROC GENMOD for a gamma model. The relationship between $\mu(x)$ and the independent variables $x_1, \ldots, x_k$ is

$$g(\mu) = -\frac{1}{\mu} = \eta$$

The Inverse Gaussian Model

The inverse Gaussian model is selected by specifying the DIST=IGAUSSIAN option in the MODEL statement. Let the mean of the inverse Gaussian distribution be $\mu(x)$. The canonical link function for inverse Gaussian distribution is $1/\mu^2$. Therefore, the relationship between $\mu(x)$ and the independent variables $x_1, \ldots, x_k$ is

$$g(\mu) = \frac{1}{\mu^2} = \eta$$

Dispersion Parameter

Continuous distributions in the exponential family (Gaussian, gamma, and inverse Gaussian) have a dispersion parameter that can be estimated by the scaled deviance. For these continuous response distributions, PROC GAM incorporates this dispersion parameter estimate into standard errors of the parameter estimates, prediction standard errors of spline and loess components, and chi-square statistics. The discrete distributions used in GAM (binomial and Poisson) do not have a dispersion parameter. For more details on the distributions, dispersion parameter, and deviance, see "Generalized Linear Models Theory" on page 3655 in Chapter 51, "The GENMOD Procedure."

Computational Resources

Since PROC GAM implements a doubly iterative method (inner backfitting iterations within each local scoring iteration), data are accessed multiple times in performing a fit. To expedite the data access, PROC GAM keeps the data used in the analysis in memory.

Let

$$
\begin{aligned}
n &= \text{number of observations used in the analysis} \\
p_r &= \text{number of parametric variables} \\
p_s &= \text{number of univariate spline smoothers} \\
p_l &= \text{number of loess smoothers} \\
p_b &= \text{number of bivariate thin-plate spline smoothers} \\
p &= p_r + p_s + p_l + p_b \\
p_n &= p_s + p_l + p_b \\
m &= \text{maximum number of iterations for the backfitting algorithm}
\end{aligned}
$$

In addition to the space to store the data ($8np$ bytes), the minimum working space (in bytes) needed for fitting a model using PROC GAM is

$$(16+8p_r)(n+2p_r)+(160+48p+16p_s+8p_b+8p_l)n+8p+32p_b+32p_s+8m+8n+(4n+4)p_s+4.$$

For fitting bivariate thin-plate smoothing spline variables, an extra $80 + 120n + 8n^2 + 8p_b$ bytes of memory are needed. For fitting loess variables, an extra $48n + 16p_l$ bytes of memory are needed. If model inference or confidence limits are requested, additional memory is required.

It is difficult to provide accurate estimates of the time required to fit a GAM model. Both the backfitting algorithm and the local scoring algorithm are iterative techniques whose convergence rates depend on the particular data being analyzed. Furthermore, the time required depends on the types of smoothers that you specify, as well as on the inferential information you request.

You can estimate the time required for problems with a larger number of observations by observing the time required for smaller problems and then using the following growth rules (obtained using by simulations) that show that the time required grows proportionally with the following:

- n^3 when at least one bivariate thin-plate spline is used
- $n^{3/2}$ when only loess smoothers are used
- n when only univariate smoothing splines are used

For additive models (models with Gaussian response distribution) with a fixed number of observations, the time required is roughly proportional to $p_n^{3/2}$. For generalized additive models (models with non-Gaussian distributions), the computation time grows more rapidly as p_n increases. This is harder to quantify as it depends on the distribution family and the number of iterations required for the local scoring algorithm to converge.

Figure 48.4 Feasible Problem Sizes for Different Smoothers

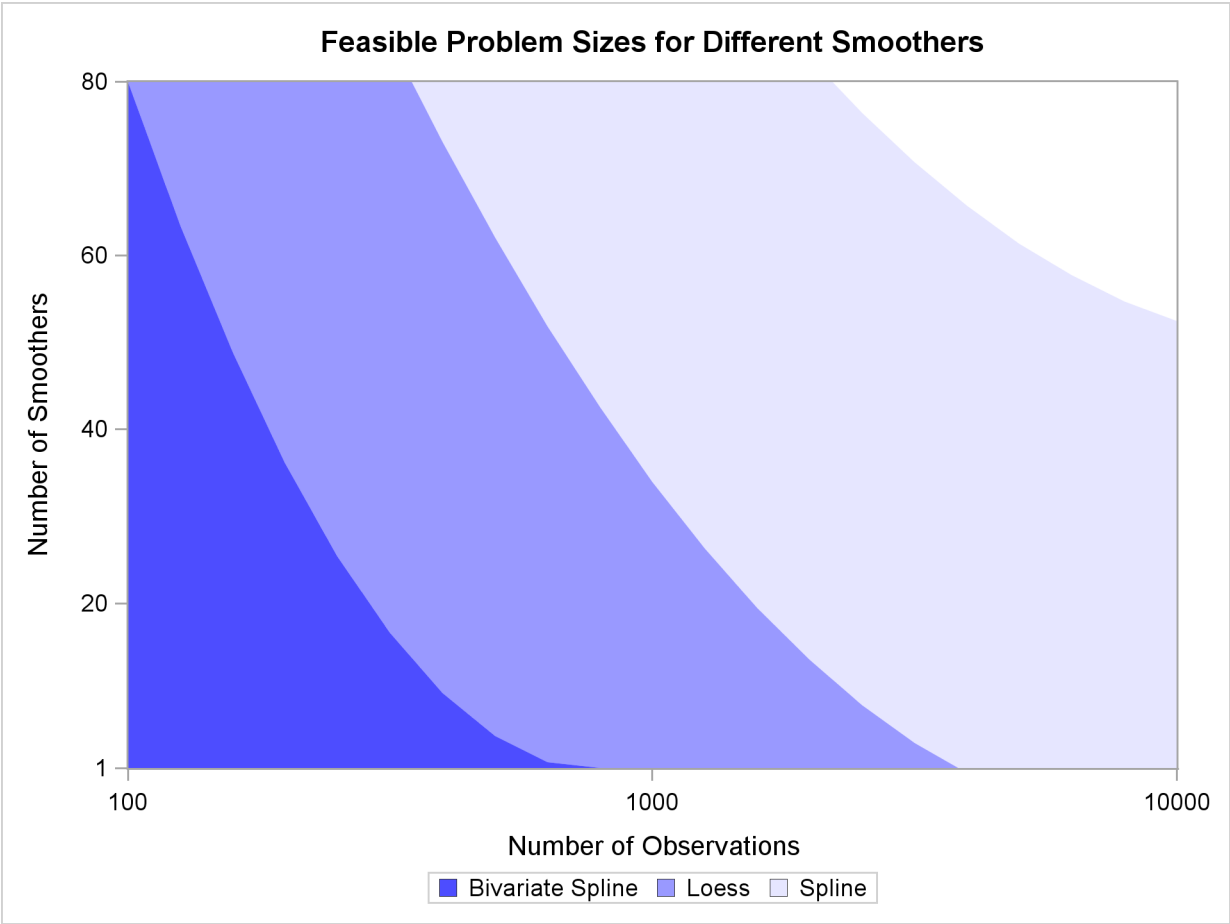

Figure 48.4 shows a rough estimation of feasible sizes for the smoothers that you can use, as a function of the number of observations and number of smoothing components. This figure depicts the regions where you can expect a single fit of an additive model to finish within a few minutes on a typical Pentium 4 system.

Note that the times reflected in Figure 48.4 are based on fitting additive models (no local scoring iterations) when no analysis of deviance or confidence limits are computed. The time required for fitting generalized additive models grows proportionally with the number of the local scoring iterations. Furthermore, analysis of deviance (if you do not request the fast approximations with the ANODEV option) requires fitting multiple GAM models as each smoothing component is omitted sequentially, and so the time estimates need to

be multiplied by the number of smoothing components when analysis of deviance is performed. Finally computation of confidence limits for each individual smoother increases the time required, especially when loess smoothers are utilized.

For univariate spline smoothers, subject to the aforementioned caveats, problems that correspond to all shaded regions in Figure 48.4 can be completed within a few minutes. For univariate loess smoothers, the two darkest regions are feasible. For bivariate spline smoothers, problems that correspond to only the darkest shading can be completed in the order of a few minutes. The problems that correspond to the upper right unshaded region might be possible, but they require long computation times.

ODS Table Names

PROC GAM assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 48.6 ODS Tables Produced by PROC GAM

ODS Table Name	Description	Statement	Option
ANODEV	Analysis of deviance table for smoothing variables	PROC GAM	Default
ClassInfo	Summary of classification variables	PROC GAM	Default
ConvergenceStatus	Convergence status of the local scoring algorithm	PROC GAM	Default
InputSummary	Input data summary	PROC GAM	Default
IterationInfo	Iteration history table	MODEL	ITPRINT
IterSummary	Iteration summary	PROC GAM	Default
FitSummary	Fit parameters and fit summary	PROC GAM	Default
ParameterEstimates	Parameter estimation for regression variables	PROC GAM	Default
ResponseProfile	Frequency counts for binary models	MODEL	DIST=BINOMIAL

By referring to the names of such tables, you can use the ODS OUTPUT statement to place one or more of these tables in output data sets.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, the GAM procedure by default produces plots of the partial predictions for each nonparametric predictor in the model. Use the PLOTS option in the PROC GAM statement to control aspects of these plots.

ODS Graph Names

PROC GAM assigns a name to each graph it creates by using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 48.7.

Table 48.7 Graphs Produced by PROC GAM

ODS Graph Name	Plot Description	PLOTS= Option
SmoothingComponentPlot	Panel of multiple partial prediction curves	COMPONENTS
SmoothingComponentPlot	Unpacked partial prediction curves	COMPONENTS(UNPACK)

By default, partial prediction plots for each component are displayed in panels containing at most six plots. If you specify more than six smoothing components, multiple panels are used. Use the PLOTS(UNPACK) option in the PROC GAM statement to display these plots individually.

Examples: GAM Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 48.1: Generalized Additive Model with Binary Data

(View the complete code for this example (gamex1.sas) in the example repository.)

This example illustrates the capabilities of the GAM procedure and compares it to the GENMOD procedure. From this example, you can see that PROC GAM is very useful in visualizing the data and detecting the nonlinearity among the variables.

The data used in this example are based on a study by Bell et al. (1994). Bell and his associates studied the result of multiple-level thoracic and lumbar laminectomy, a corrective spinal surgery commonly performed on children. The data in the study consist of retrospective measurements on 83 patients. The specific outcome of interest is the presence (1) or absence (0) of kyphosis, defined as a forward flexion of the spine of at least 40 degrees from vertical. The available predictor variables are age in months at time of the operation (Age), the starting of vertebrae levels involved in the operation (StartVert), and the number of levels involved (NumVert). The goal of this analysis is to identify risk factors for kyphosis. PROC GENMOD can be used to investigate the relationship among kyphosis and the predictors. The following statements create the data kyphosis and fit a logistic model specifying linear effects for the three predictors:

```
title 'Comparing PROC GAM with PROC GENMOD';
data kyphosis;
   input Age StartVert NumVert Kyphosis @@;
   datalines;
71  5  3  0     158 14 3 0    128 5   4 1
2   1  5  0     1   15 4 0    1   16  2 0
61  17 2  0     37  16 3 0    113 16  2 0
59  12 6  1     82  14 5 1    148 16  3 0
18  2  5  0     1   12 4 0    243 8   8 0
168 18 3  0     1   16 3 0    78  15  6 0
175 13 5  0     80  16 5 0    27  9   4 0
22  16 2  0     105 5  6 1    96  12  3 1
131 3  2  0     15  2  7 1    9   13  5 0
12  2 14  1     8   6  3 0    100 14  3 0
4   16 3  0     151 16 2 0    31  16  3 0
125 11 2  0     130 13 5 0    112 16  3 0
140 11 5  0     93  16 3 0    1   9   3 0
52  6  5  1     20  9  6 0    91  12  5 1
73  1  5  1     35  13 3 0    143 3   9 0
61  1  4  0     97  16 3 0    139 10  3 1
136 15 4  0     131 13 5 0    121 3   3 1
177 14 2  0     68  10 5 0    9   17  2 0
139 6  10 1     2   17 2 0    140 15  4 0
72  15 5  0     2   13 3 0    120 8   5 1
51  9  7  0     102 13 3 0    130 1   4 1
114 8  7  1     81  1  4 0    118 16  3 0
```

```
118 16 4  0     17  10 4 0    195 17  2 0
159 13 4  0     18  11 4 0    15  16  5 0
158 15 4  0     127 12 4 0    87  16  4 0
206 10 4  0     11  15 3 0    178 15  4 0
157 13 3  1     26  13 7 0    120 13  2 0
42  6  7  1     36  13 4 0
;

proc genmod data=kyphosis descending;
   model Kyphosis = Age StartVert NumVert/link=logit dist=binomial;
run;
```

The GENMOD analysis of the independent variable effects is shown in Output 48.1.1. Based on these results, the only significant factor is StartVert with a log odds ratio of –0.1972. The variable NumVert has a *p*-value of 0.0904 with a log odds ratio of 0.3031.

Output 48.1.1 GENMOD Analysis: Partial Output

Comparing PROC GAM with PROC GENMOD

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	-1.2497	1.2424	-3.6848	1.1853	1.01	0.3145
Age	1	0.0061	0.0055	-0.0048	0.0170	1.21	0.2713
StartVert	1	-0.1972	0.0657	-0.3260	-0.0684	9.01	0.0027
NumVert	1	0.3031	0.1790	-0.0477	0.6540	2.87	0.0904
Scale	0	1.0000	0.0000	1.0000	1.0000		

Note: The scale parameter was held fixed.

The GENMOD procedure assumes a strict linear relationship between the predictors and the response function, which is the logit (log odds) in this model. The following SAS statements use PROC GAM to investigate a less restrictive model, with moderately flexible spline terms for each of the predictors:

```
title 'Comparing PROC GAM with PROC GENMOD';
proc gam data=kyphosis;
   model Kyphosis (event='1') = spline(Age      ,df=3)
                                spline(StartVert,df=3)
                                spline(NumVert  ,df=3) / dist=binomial;
run;
```

The MODEL statement requests an additive model with a univariate smoothing spline for each term. The response variable option EVENT= chooses Kyphosis=1 (presence) as the event so that the probability of presence of kyphosis is modeled. The option DIST=BINOMIAL with binary responses specifies a logistic model. Each term is fit by using a univariate smoothing spline with three degrees of freedom. Of these three degrees of freedom, one is taken up by the linear portion of the fit and two are left for the nonlinear spline portion. Although this might seem to be an unduly modest amount of flexibility, it is better to be conservative with a data set this small.

Output 48.1.2 and Output 48.1.3 list the output from PROC GAM.

Output 48.1.2 Summary Statistics

Comparing PROC GAM with PROC GENMOD

The GAM Procedure
Dependent Variable: Kyphosis
Smoothing Model Component(s): spline(Age) spline(StartVert) spline(NumVert)

Summary of Input Data Set	
Number of Observations	83
Number of Missing Observations	0
Distribution	Binomial
Link Function	Logit

Response Profile		
Ordered Value	Kyphosis	Total Frequency
1	0	65
2	1	18

Note: PROC GAM is modeling the probability that Kyphosis=1. One way to change this to model the probability that Kyphosis=0 is to specify the response variable option EVENT='0'.

Iteration Summary and Fit Statistics	
Number of local scoring iterations	9
Local scoring convergence criterion	2.6635652E-9
Final Number of Backfitting Iterations	1
Final Backfitting Criterion	5.2326578E-9
The Deviance of the Final Estimate	46.610922438

Output 48.1.3 Model Fit Statistics

Regression Model Analysis
Parameter Estimates

Parameter	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	-2.01533	1.45620	-1.38	0.1706
Linear(Age)	0.01213	0.00794	1.53	0.1308
Linear(StartVert)	-0.18615	0.07628	-2.44	0.0171
Linear(NumVert)	0.38347	0.19102	2.01	0.0484

Smoothing Model Analysis
Fit Summary for Smoothing Components

Component	Smoothing Parameter	DF	GCV	Num Unique Obs
Spline(Age)	0.999996	2.000000	328.512831	66
Spline(StartVert)	0.999551	2.000000	317.646685	16
Spline(NumVert)	0.921758	2.000000	20.144056	10

Output 48.1.3 *continued*

Smoothing Model Analysis Analysis of Deviance				
Source	DF	Sum of Squares	Chi-Square	Pr > ChiSq
Spline(Age)	2.00000	10.494369	10.4944	0.0053
Spline(StartVert)	2.00000	5.494968	5.4950	0.0641
Spline(NumVert)	2.00000	2.184518	2.1845	0.3355

The critical part of the GAM results is the "Analysis of Deviance" table, shown in Output 48.1.3. For each smoothing effect in the model, this table gives a χ^2 test comparing the deviance between the full model and the model without the nonparametric component of this variable. The analysis of deviance results indicate that the nonparametric effect of Age is highly significant, the nonparametric effect of StartVert is nearly significant, and the nonparametric effect of NumVert is insignificant at the 5% level.

PROC GAM can also perform approximate analysis of deviance for smoothing effects by using the ANODEV=NOREFIT option, as in the following statements:

```
title 'PROC GAM with Approximate Analysis of Deviance';
proc gam data=kyphosis;
   model Kyphosis (event='1') = spline(Age      ,df=3)
                                spline(StartVert,df=3)
                                spline(NumVert  ,df=3) /
                                    dist=binomial anodev=norefit;
run;
```

The "Approximate Analysis of Deviance" table shown in Output 48.1.4 yields similar conclusions to those of the "Analysis of Deviance" table (Output 48.1.3). In addition to fitting the model using all the specified smoothing effects, the default ANODEV=REFIT option requires fitting p additional subset models to p smoothing effects. Each submodel is fit by omitting one smoothing term from the model. By contrast, the ANODEV=NOREFIT option keeps the nonparametric terms fixed and requires a weighted least squares fit for only the parametric part of the model. Hence, GAM with the ANODEV=NOREFIT option is computationally inexpensive and is useful for obtaining approximate analysis of deviance results for models with many smoothing effects. This option assumes that the remaining nonparametric terms do not change much with the deletion of one nonparametric component. It should be used with caution when a model contains highly correlated predictors.

Output 48.1.4 Approximate Analysis of Deviance Table

PROC GAM with Approximate Analysis of Deviance

The GAM Procedure
Dependent Variable: Kyphosis
Smoothing Model Component(s): spline(Age) spline(StartVert) spline(NumVert)

Smoothing Model Analysis Approximate Analysis of Deviance			
Source	DF	Chi-Square	Pr > ChiSq
Spline(Age)	2.00000	7.0888	0.0289
Spline(StartVert)	2.00000	5.0431	0.0803
Spline(NumVert)	2.00000	2.2471	0.3251

Plots of the partial predictions for each predictor can be used to investigate why PROC GAM and PROC GENMOD produce different results. The following statements use ODS Graphics to produce plots of the individual smoothing components. The CLM suboption in the PLOTS=COMPONENTS option adds a curvewise Bayesian confidence band to each smoothing component, while the COMMONAXES suboption forces all three smoothing component plots to share the same vertical axis limits, allowing a visual judgment of the relative nonparametric effect sizes.

```
ods graphics on;

proc gam data=kyphosis plots=components(clm commonaxes);
   model Kyphosis (event='1') = spline(Age      ,df=3)
                                spline(StartVert,df=3)
                                spline(NumVert  ,df=3) / dist=binomial;
run;

ods graphics off;
```

For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the GAM procedure, see the section "ODS Graphics" on page 3406. The smoothing component plots are displayed in Output 48.1.5.

Output 48.1.5 Partial Prediction for Each Predictor

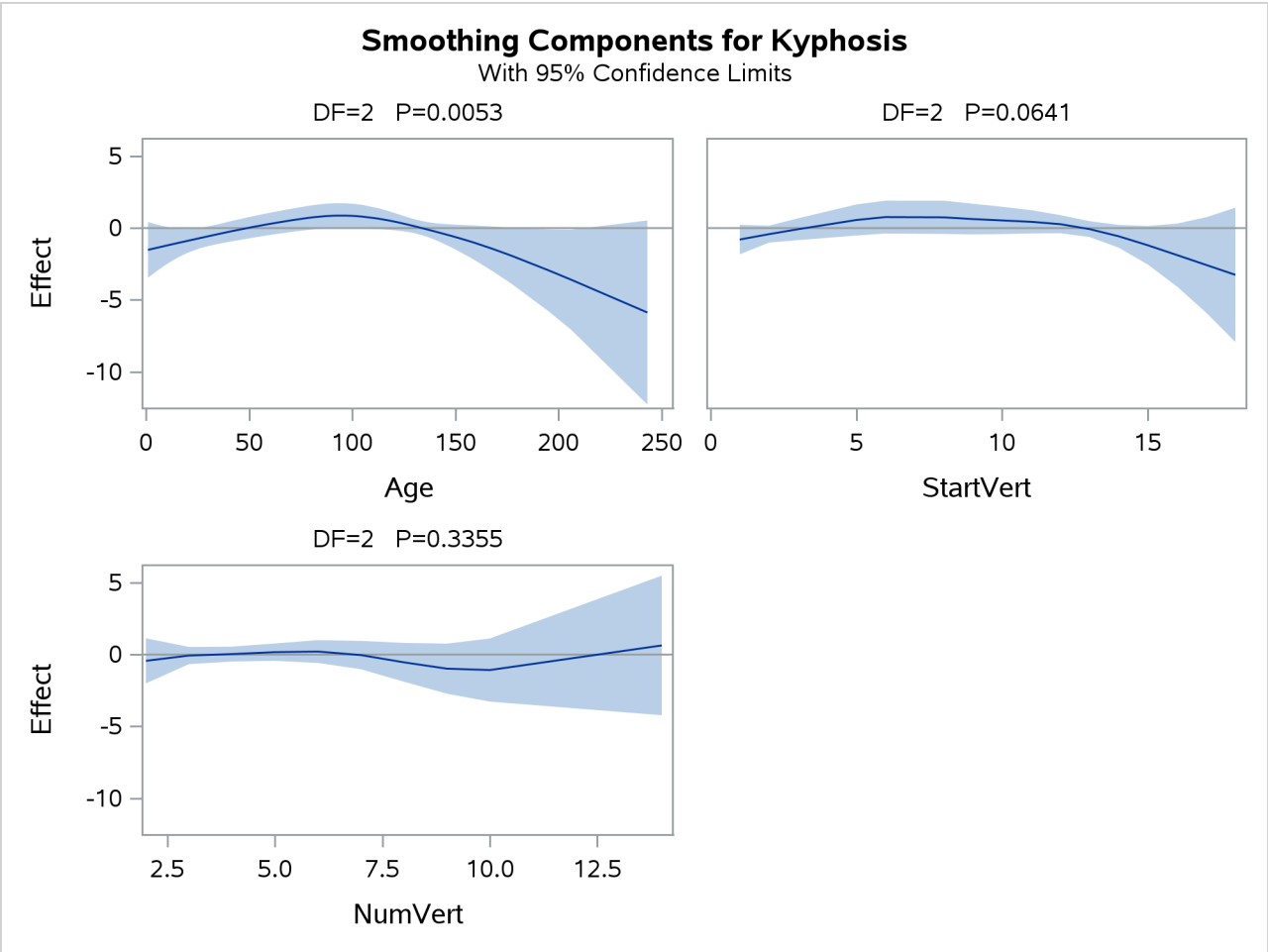

The plots show that the partial predictions corresponding to both Age and StartVert have a quadratic pattern, while NumVert has a more complicated but ultimately nonsignificant pattern.

An important difference between the first analysis of these data with PROC GENMOD and the subsequent analysis with PROC GAM is that PROC GAM indicates that Age has a significant but nonlinear association with kyphosis. The difference is due to the fact that the generalized linear model includes only the linear effect of Age whereas the generalized additive model allows a more complex relationship, which the plots indicate is nearly quadratic. Having used the GAM procedure to discover an appropriate form of the dependence of Kyphosis on each of the three independent variables, you can use the GENMOD procedure to fit and assess the corresponding parametric model. The following statements fit a generalized linear model with quadratic terms for all three variables. The parameter estimates are shown in Output 48.1.6.

```
title 'Comparing PROC GAM with PROC GENMOD';
proc genmod data=kyphosis descending;
   model kyphosis = Age       Age      *Age
                    StartVert StartVert*StartVert
                    NumVert   NumVert  *NumVert /
                        link=logit  dist=binomial;
run;
```

Output 48.1.6 Logistic Model with Quadratic Terms

Comparing PROC GAM with PROC GENMOD

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	-5.8134	2.5618	-10.8345	-0.7923	5.15	0.0233
Age	1	0.0819	0.0345	0.0143	0.1496	5.63	0.0176
Age*Age	1	-0.0004	0.0002	-0.0008	-0.0000	4.32	0.0376
StartVert	1	0.4394	0.3234	-0.1944	1.0733	1.85	0.1742
StartVert*StartVert	1	-0.0396	0.0202	-0.0791	-0.0001	3.86	0.0495
NumVert	1	0.3798	0.5988	-0.7939	1.5535	0.40	0.5259
NumVert*NumVert	1	0.0020	0.0420	-0.0803	0.0843	0.00	0.9621
Scale	0	1.0000	0.0000	1.0000	1.0000		

Note: The scale parameter was held fixed.

The p-value for the χ^2 test is 0.0376 for dropping the quadratic term of Age, 0.0495 for dropping the quadratic term of StartVert, and 0.9621 for dropping this quadratic term of NumVert. The results for the quadratic GENMOD model are consistent with the GAM results.

Example 48.2: Poisson Regression Analysis of Component Reliability

(View the complete code for this example (gamex2.sas) in the example repository.)

In this example, the number of maintenance repairs on a complex system are modeled as realizations of Poisson random variables. The system under investigation has a large number of components, which occasionally break down and are replaced or repaired. During a four-year period, the system was observed to be in a state of steady operation, meaning that the rate of operation remained approximately constant. A monthly maintenance record is available for that period, which tracks the number of components removed for maintenance each month. The data are listed in the following statements, which create a SAS data set:

```
title 'Analysis of Component Reliability';
data equip;
   input year month removals @@;
   datalines;
1987   1  2 1987   2  4 1987   3  3
1987   4  3 1987   5  3 1987   6  8
1987   7  2 1987   8  6 1987   9  3
1987  10  9 1987  11  4 1987  12 10
1988   1  4 1988   2  6 1988   3  4
1988   4  4 1988   5  3 1988   6  5
1988   7  3 1988   8  4 1988   9  5
1988  10  3 1988  11  6 1988  12  3
1989   1  2 1989   2  6 1989   3  1
1989   4  5 1989   5  5 1989   6  4
1989   7  2 1989   8  2 1989   9  2
1989  10  5 1989  11  1 1989  12 10
1990   1  3 1990   2  8 1990   3 12
1990   4  7 1990   5  3 1990   6  2
1990   7  4 1990   8  3 1990   9  0
1990  10  6 1990  11  6 1990  12  6
;
```

For planning purposes, it is of interest to understand the long- and short-term trends in the maintenance needs of the system. Over the long term, it is suspected that the quality of new components and repair work improves over time, so the number of component removals would tend to decrease from year to year. It is not known whether the robustness of the system is affected by seasonal variations in the operating environment, but this possibility is also of interest.

Because the maintenance record is in the form of counts, the number of removals are modeled as realizations of Poisson random variables. Denote by λ_{ij} the unobserved component removal rate for year i and month j. Since the data were recorded at regular intervals (from a system operating at a constant rate), each λ_{ij} is assumed to be a function of year and month only.

A preliminary two-way analysis is performed by using PROC GENMOD to make broad inferences on repair trends. A log-link is specified for the model

$$\log \lambda_{ij} \quad = \quad \mu + \alpha_i^Y + \alpha_j^M$$

where μ is a grand mean, α_i^Y is the effect of the ith year, and α_j^M is the effect of the jth month.

In the following statements, the CLASS statement declares the variables year and month as categorical. Type III sum of squares are requested to test whether there is an overall effect of year and/or month.

```
title2 'Two-way model';
proc genmod data=equip;
   class year month;
   model removals=year month / dist=Poisson link=log type3;
run;
```

Output 48.2.1 displays the listed Type III statistics for the fitted model. With the test for year effects yielding a p-value of 0.4527, there is no evidence of a long-term trend in maintenance rates. Apparently, the quality of new or repaired components did not change between 1987 and 1990. However, the test for monthly trends does yield a small p-value of 0.0321, indicating that seasonal trends are significant at the $\alpha = 0.05$ level.

Output 48.2.1 PROC GENMOD Listing for Type III Analysis

Analysis of Component Reliability
Two-way model

The GENMOD Procedure

LR Statistics For Type 3 Analysis			
Source	DF	Chi-Square	Pr > ChiSq
year	3	2.63	0.4527
month	11	21.12	0.0321

If year is dropped from the model, the focus of the analysis is now on identifying the form of the underlying seasonal trend, which is a task that PROC GAM is especially suited for. PROC GAM will be used to fit both a reduced categorical model, with year eliminated, and a nonparametric spline model. Although PROC GENMOD also has the capability to fit categorical models, as demonstrated earlier, PROC GAM will be used here to fit both models for a better comparison.

The following PROC GAM statements specify the reduced categorical model and write predicted values to a data set. For this part of the analysis, a CLASS statement is again used to specify that month is a categorical variable. In the follow-up, the seasonal effect will be treated as a nonparametric function of month.

```
title2 'One-way model';
proc gam data=equip;
   class month;
   model removals=param(month) / dist=Poisson;
   output out=est p;
run;
```

The following statements use the SGPLOT procedure to generate a plot of the estimated seasonal trend. The plot is displayed in Output 48.2.2.

```
proc sort data=est;by month;run;
proc sgplot data=est;
   title "Predicted Seasonal Trend";
   yaxis label="Number of Removals";
   xaxis integer values=(1 to 12);
   scatter x=Month y=Removals / name="points"
                                legendLabel="Removals";
```

```
   series  x=Month y=p_Removals / name="line"
                                  legendLabel="Predicted Removals"
                                  lineattrs = GRAPHFIT;
   discretelegend "points" "line";
run;
```

Output 48.2.2 Predicted Seasonal Trend from a Parametric Model Fit Using a CLASS Statement

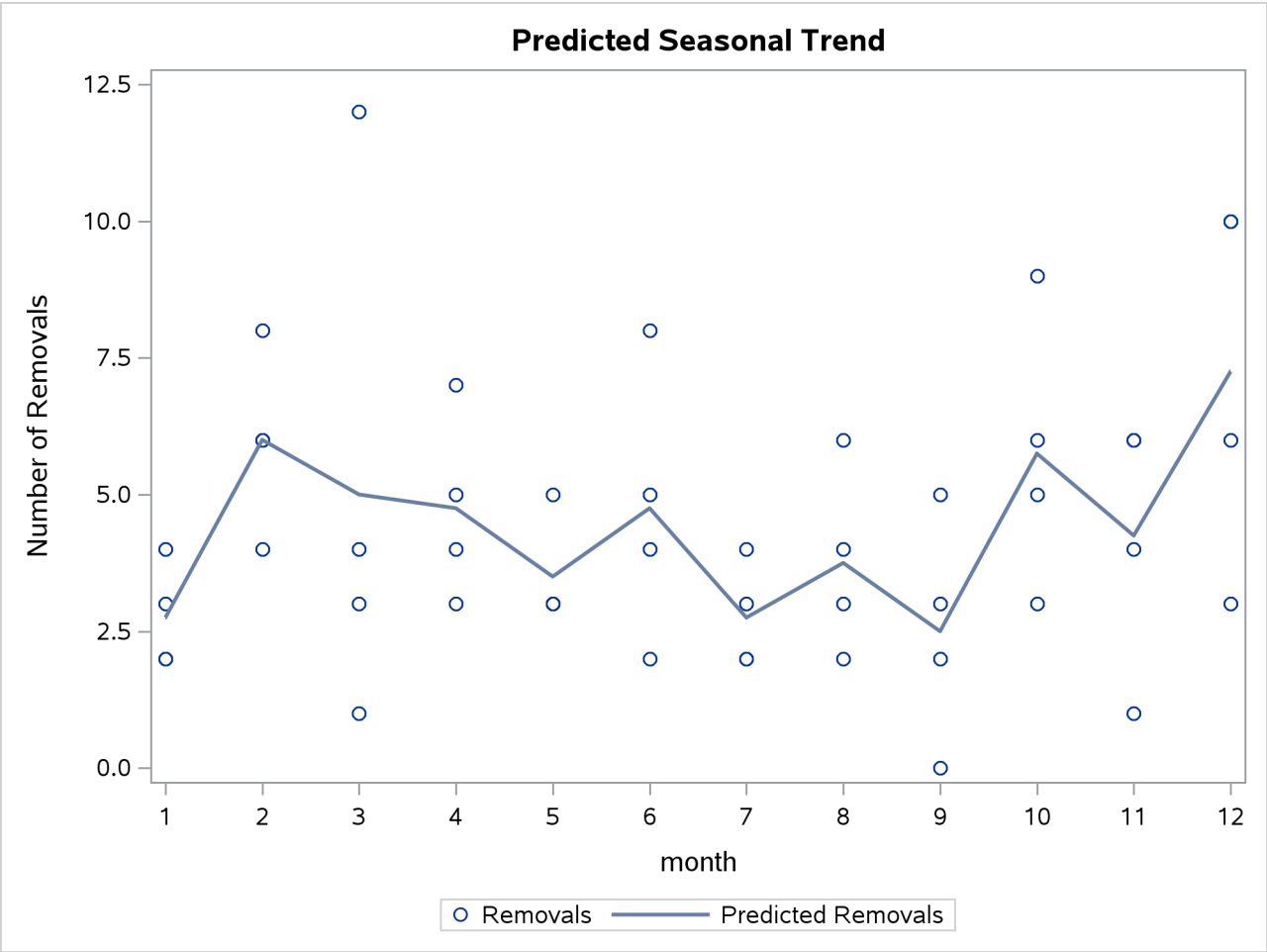

The predicted repair rates shown in Output 48.2.2 form a jagged seasonal pattern. Ignoring the month-to-month fluctuations, which are difficult to explain and can be artifacts of random noise, the general removal rate trend is high in winter and low in summer.

One advantage of nonparametric regression is its ability to highlight general trends in the data, such as those described earlier, and to attribute local fluctuations to unexplained random noise. The nonparametric regression model used by PROC GAM specifies that the underlying removal rates λ_j are of the form

$$\log \lambda_j = \beta_0 + \beta_1 Month_j + s(Month_j)$$

where β_1 is a linear coefficient and $s(\,)$ is a nonparametric regression function. β_1 and $s(\,)$ define the linear and nonparametric parts, respectively, of the seasonal trend.

The following statements request that PROC GAM fit a cubic spline model to the monthly repair data. The output listing is displayed in Output 48.2.3 and Output 48.2.4.

```
title 'Analysis of Component Reliability';
title2 'Spline model';
proc gam data=equip;
   model removals=spline(month) / dist=Poisson method=gcv;
run;
```

The METHOD=GCV option is used to determine an appropriate level of smoothing.

Output 48.2.3 PROC GAM Listing for Cubic Spline Regression Using the METHOD=GCV Option

Analysis of Component Reliability
Spline model

The GAM Procedure
Dependent Variable: removals
Smoothing Model Component(s): spline(month)

Summary of Input Data Set	
Number of Observations	48
Number of Missing Observations	0
Distribution	Poisson
Link Function	Log

Output 48.2.4 Model Fit Statistics

Regression Model Analysis
Parameter Estimates

Parameter	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1.34594	0.14509	9.28	<.0001
Linear(month)	0.02274	0.01893	1.20	0.2362

Smoothing Model Analysis
Fit Summary for Smoothing Components

Component	Smoothing Parameter	DF	GCV	Num Unique Obs
Spline(month)	0.901512	1.879980	0.115848	12

Smoothing Model Analysis
Analysis of Deviance

Source	DF	Sum of Squares	Chi-Square	Pr > ChiSq
Spline(month)	1.87998	8.877764	8.8778	0.0103

Notice in the listing in Output 48.2.4 that the DF value chosen for the nonlinear portion of the spline by minimizing GCV is about 1.88, which is smaller than the default value of 3. This indicates that the spline model of the seasonal trend is relatively simple. As indicated by the "Analysis of Deviance" table, it is a significant feature of the data. The table lists a p-value of 0.0103 for the hypothesis of no seasonal trend. Note also that the "Parameter Estimates" table lists a p-value of 0.2362 for the hypothesis of no linear factor in the seasonal trend indicating no significant linear trend.

The following statements use ODS Graphics to plot the smoothing component for the effect of Month on predicted repair rates. The CLM suboption for the PLOTS=COMPONENTS option adds a 95% confidence band to the fit.

```
ods graphics on;

proc gam data=equip plots=components(clm);
   model removals=spline(month) / dist=Poisson method=gcv;
run;

ods graphics off;
```

For general information about ODS graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the GAM procedure, see the section "ODS Graphics" on page 3406. The smoothing component plot is displayed in Output 48.2.5.

In Output 48.2.5, it is apparent that the pattern of repair rates follows the general pattern observed in Output 48.2.2. However, the plot in Output 48.2.5 is much cleaner because the month-to-month fluctuations are smoothed out to reveal the broader seasonal trend.

Output 48.2.5 Estimated Nonparametric Factor of Seasonal Trend, Along with 95% Confidence Bounds

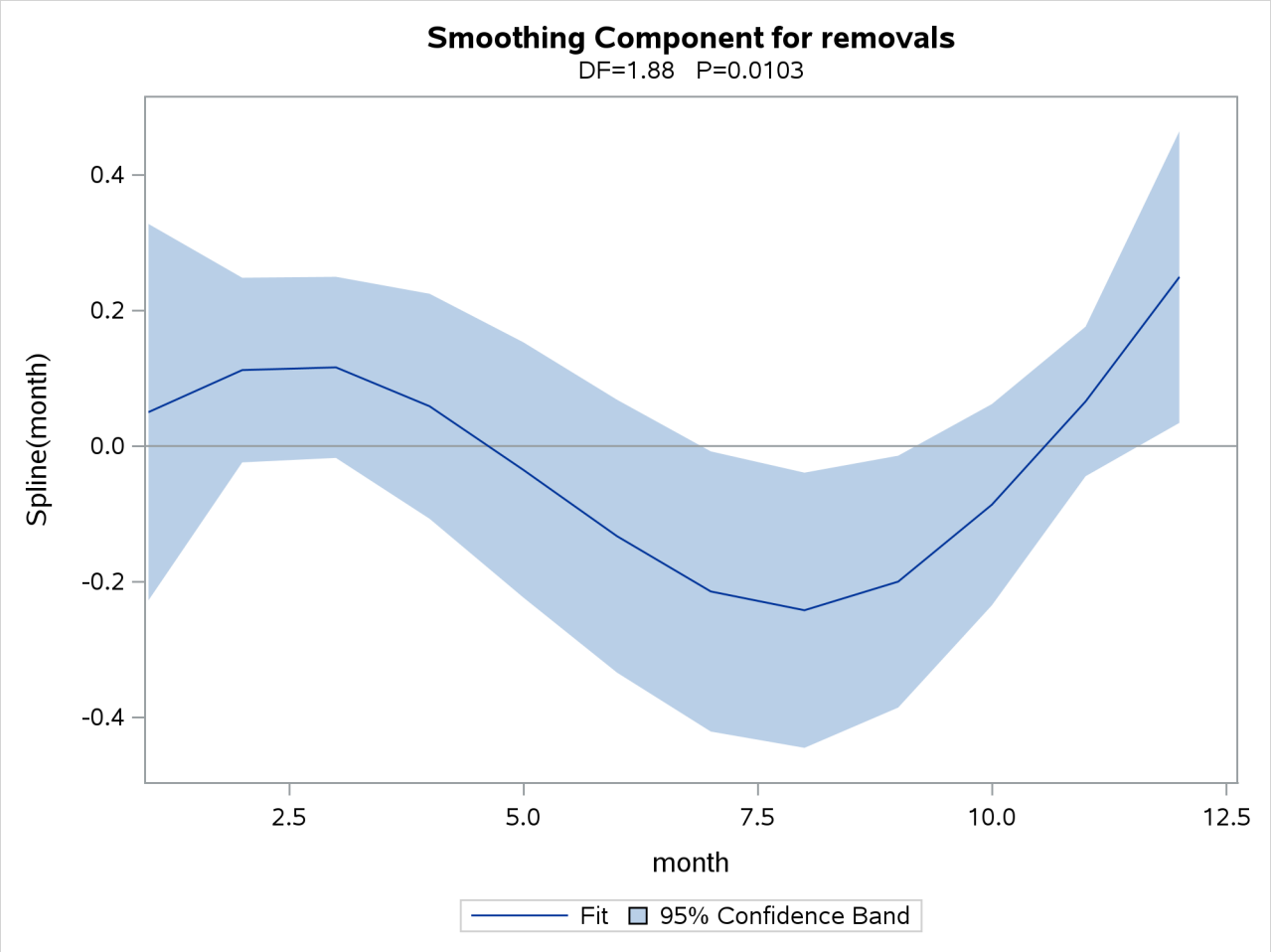

In Output 48.2.1 the small p-value ($p = 0.0321$) for the hypothesis of no seasonal trend indicates that the

data exhibit significant seasonal structure. Output 48.2.5 is a graphical illustration of the seasonality of the number of removals.

Example 48.3: Comparing PROC GAM with PROC LOESS

(View the complete code for this example (gamex3.sas) in the example repository.)

In an analysis of simulated data from a hypothetical chemistry experiment, additive nonparametric regression performed by PROC GAM is compared to the unrestricted multidimensional procedure of PROC LOESS.

In each repetition of the experiment, a catalyst is added to a chemical solution, thereby inducing synthesis of a new material. The data are measurements of the temperature of the solution, the amount of catalyst added, and the yield of the chemical reaction. The following statements read and plots the raw data.

```
data ExperimentA;
   format Temperature f4.0 Catalyst f6.3 Yield f8.3;
   input Temperature Catalyst Yield @@;
   datalines;
80  0.005 6.039  80 0.010 4.719  80 0.015 6.301
80  0.020 4.558  80 0.025 5.917  80 0.030 4.365
80  0.035 6.540  80 0.040 5.063  80 0.045 4.668
80  0.050 7.641  80 0.055 6.736  80 0.060 7.255
80  0.065 5.515  80 0.070 5.260  80 0.075 4.813
80  0.080 4.465  90 0.005 4.540  90 0.010 3.553
90  0.015 5.611  90 0.020 4.586  90 0.025 6.503
90  0.030 4.671  90 0.035 4.919  90 0.040 6.536
90  0.045 4.799  90 0.050 6.002  90 0.055 6.988
90  0.060 6.206  90 0.065 5.193  90 0.070 5.783
90  0.075 6.482  90 0.080 5.222 100 0.005 5.042
100 0.010 5.551 100 0.015 4.804 100 0.020 5.313
100 0.025 4.957 100 0.030 6.177 100 0.035 5.433
100 0.040 6.139 100 0.045 6.217 100 0.050 6.498
100 0.055 7.037 100 0.060 5.589 100 0.065 5.593
100 0.070 7.438 100 0.075 4.794 100 0.080 3.692
110 0.005 6.005 110 0.010 5.493 110 0.015 5.107
110 0.020 5.511 110 0.025 5.692 110 0.030 5.969
110 0.035 6.244 110 0.040 7.364 110 0.045 6.412
110 0.050 6.928 110 0.055 6.814 110 0.060 8.071
110 0.065 6.038 110 0.070 6.295 110 0.075 4.308
110 0.080 7.020 120 0.005 5.409 120 0.010 7.009
120 0.015 6.160 120 0.020 7.408 120 0.025 7.123
120 0.030 7.009 120 0.035 7.708 120 0.040 5.278
120 0.045 8.111 120 0.050 8.547 120 0.055 8.279
120 0.060 8.736 120 0.065 6.988 120 0.070 6.283
120 0.075 7.367 120 0.080 6.579 130 0.005 7.629
130 0.010 7.171 130 0.015 5.997 130 0.020 6.587
130 0.025 7.335 130 0.030 7.209 130 0.035 8.259
130 0.040 6.530 130 0.045 8.400 130 0.050 7.218
130 0.055 9.167 130 0.060 9.082 130 0.065 7.680
130 0.070 7.139 130 0.075 7.275 130 0.080 7.544
140 0.005 4.860 140 0.010 5.932 140 0.015 3.685
140 0.020 5.581 140 0.025 4.935 140 0.030 5.197
```

```
140 0.035 5.559 140 0.040 4.836 140 0.045 5.795
140 0.050 5.524 140 0.055 7.736 140 0.060 5.628
140 0.065 6.644 140 0.070 3.785 140 0.075 4.853
140 0.080 6.006
;

proc sort data=ExperimentA;
   by Temperature Catalyst;
run;

proc template;
   define statgraph surface;
      dynamic _X _Y _Z _T;
      begingraph;
         entrytitle _T;
         layout overlay3d/
            xaxisopts=(linearopts=(tickvaluesequence=
                       (start=85 end=135 increment=25)))
            yaxisopts=(linearopts=(tickvaluesequence=
                       (start=0 end=0.08 increment=0.04)))
            rotate=30 cube=false;
         surfaceplotparm x=_X y=_Y z=_Z;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=ExperimentA template=surface;
   dynamic _X='Temperature' _Y='Catalyst' _Z='Yield' _T='Raw Data';
run;
```

The plot is displayed in Output 48.3.1. A surface fitted to the plot of Output 48.3.1 by PROC LOESS will be of a very general (and flexible) type, since the procedure requires only weak assumptions about the structure of the dependencies among the data. PROC GAM, on the other hand, makes stronger structural assumptions by restricting the fitted surface to an additive form. These differences will be demonstrated in this example.

Output 48.3.1 Surface Plot of Yield by Temperature and Amount of Catalyst

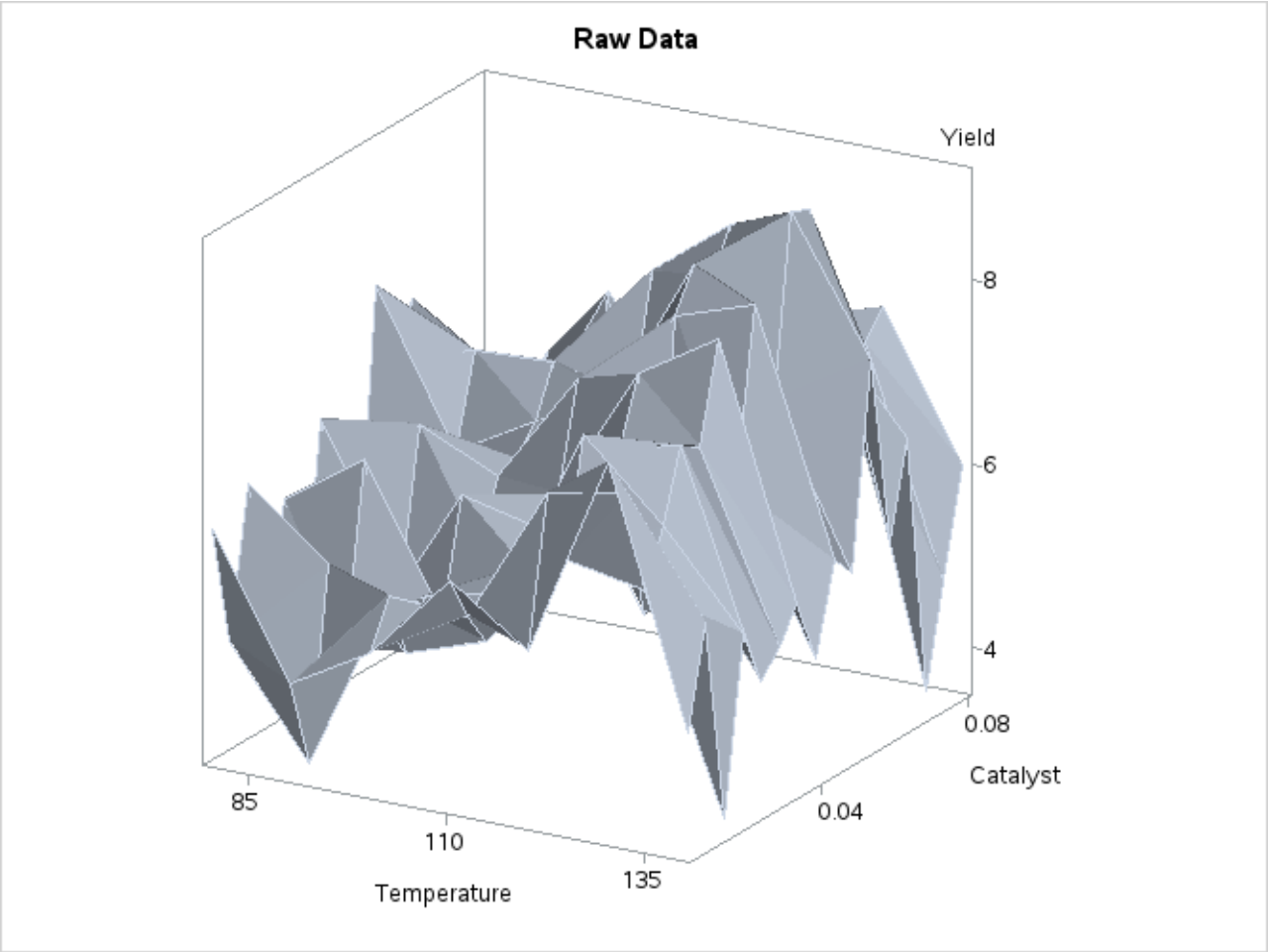

The following statements request that both PROC LOESS and PROC GAM fit surfaces to the data:

```
ods output ScoreResults=PredLOESS;
proc loess data=ExperimentA;
   model Yield = Temperature Catalyst
                 / scale=sd select=gcv degree=2;
   score;
run;

proc gam data=PredLoess;
   model Yield = loess(Temperature) loess(Catalyst) / method=gcv;
   output out=PredGAM p=Gam_p_;
run;
```

In both cases the smoothing parameter was chosen as the value that minimizes GCV. This is performed automatically by PROC LOESS and PROC GAM.

The following statements generate plots of the predicted yields, which are displayed in Output 48.3.2:

```
proc template;
   define statgraph surface1;
      begingraph;
         entrytitle "Fitted Surface";
         layout lattice/columns=2;
            layout
            overlay3d/xaxisopts=(linearopts=(tickvaluesequence=
                                 (start=85 end=135 increment=25)))
                     yaxisopts=(linearopts=(tickvaluesequence=
                                 (start=0 end=0.08 increment=0.04)))
                     zaxisopts=(label="P_Yield")
                     rotate=30 cube=0;
               entry "PROC LOESS"/location=outside valign=top
                                   textattrs=graphlabeltext;
               surfaceplotparm x=Temperature y=Catalyst z=p_Yield;
            endlayout;
            layout
            overlay3d/xaxisopts=(linearopts=(tickvaluesequence=
                                 (start=85 end=135 increment=25)))
                     yaxisopts=(linearopts=(tickvaluesequence=
                                 (start=0 end=0.08 increment=0.04)))
                     rotate=30 cube=0
                     zaxisopts=(label="P_Yield")
                     rotate=30 cube=0;
               entry "PROC GAM"/location=outside valign=top
                                 textattrs=graphlabeltext;
               surfaceplotparm x=Temperature y=Catalyst z=Gam_p_Yield;
            endlayout;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=PredGAM template=surface1;
run;
```

Output 48.3.2 Fitted Regression Surfaces

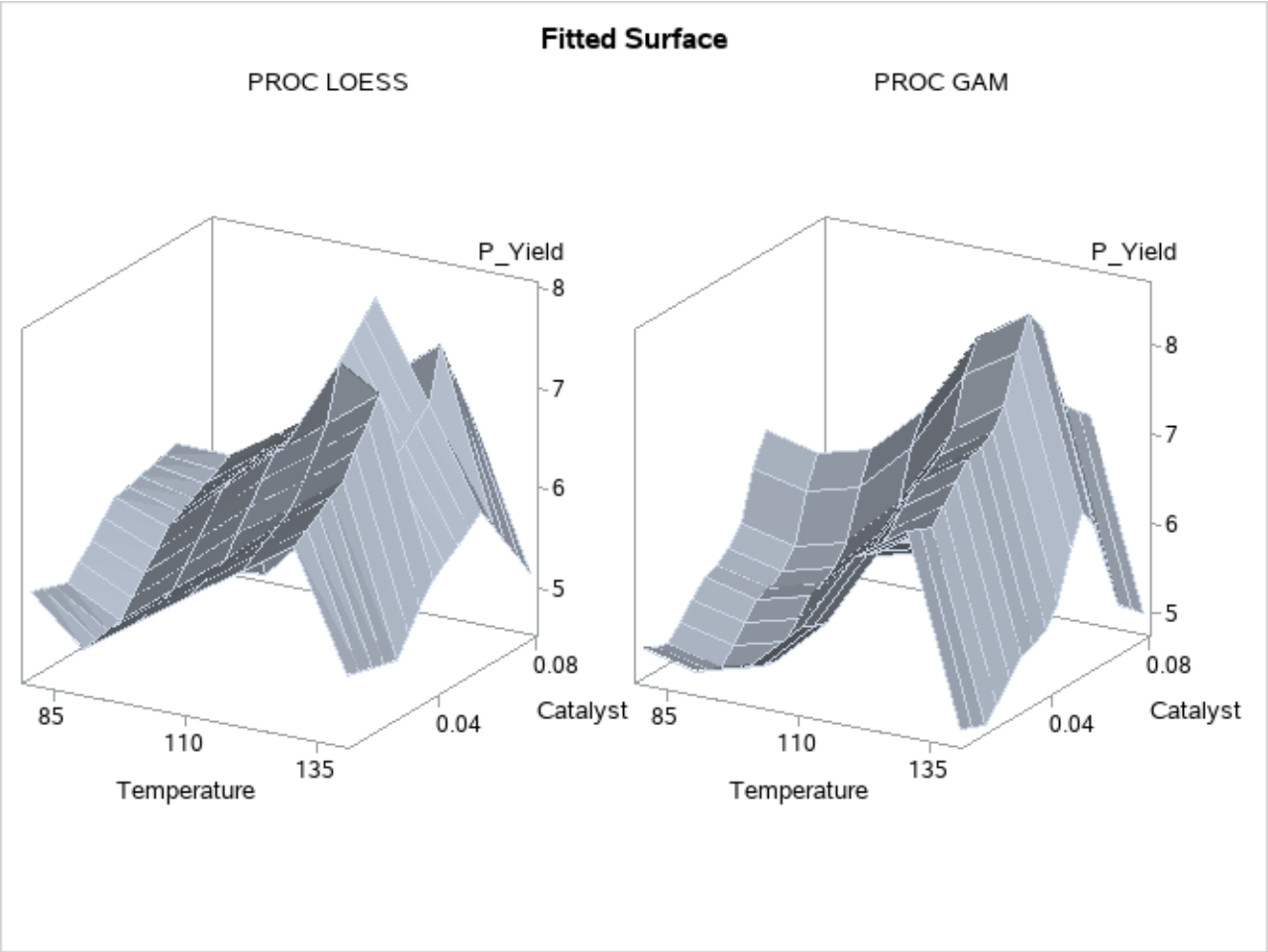

Though both PROC LOESS and PROC GAM use the statistical technique loess, it is apparent from Output 48.3.2 that the manner in which it is applied is very different. By smoothing out the data in local neighborhoods, PROC LOESS essentially fits a surface to the data in pieces, one neighborhood at a time. The local regions are treated independently, so separate areas of the fitted surface are only weakly related. PROC GAM imposes additive structure, requiring that cross sections of the fitted surface always have the same shape and thereby relating regions that have a common value of the same individual regressor variable. Under that restriction, the loess technique need not be applied to the entire multidimensional scatter plot, but only to one-dimensional cross sections of the data.

The advantage of using additive model fitting is that its statistical power is directed toward univariate smoothing, and so it is able to discern the finer details of any underlying structure in the data. Regression data can be very sparse when viewed in the context of multidimensional space, even when every individual set of regressor values densely covers its range. This is the familiar curse of dimensionality. Sparse data greatly restrict the effectiveness of nonparametric procedures, but additive model fitting, when appropriate, is one way to overcome this limitation.

To examine these properties, you can use ODS Graphics to generate plots of cross sections of the unrestricted (PROC LOESS) and additive (PROC GAM) fitted surfaces for the variable Catalyst, as shown in the following statements:

```
proc template;
   define statgraph projection;
      begingraph;
         entrytitle "Cross Sections of Fitted Surfaces";
         layout lattice/rows=2 columndatarange=unionall
                       columngutter=10;
            columnAxes;
               columnAxis / display=all griddisplay=auto_on;
            endColumnAxes;

            layout overlay/
               xaxisopts=(display=none)
               yaxisopts=(label="LOESS Prediction"
               linearopts=(viewmin=2 viewmax=10));
               seriesplot x=Catalyst y=p_Yield /
                  group=temperature
                  name="Temperature";
            endlayout;

            layout overlay/
               xaxisopts=(display=none)
               yaxisopts=(label="GAM Prediction"
               linearopts=(viewmin=2 viewmax=10));
               seriesplot x=Catalyst y=Gam_p_Yield /
                  group=temperature
                  name="Temperature";
            endlayout;

            columnheaders;
               discreteLegend "Temperature" / title = "Temperature";
            endcolumnheaders;

         endlayout;
      endgraph;
   end;
run;

proc sgrender data=PredGAM template=projection;
run;
```

The plots are displayed in Output 48.3.3.

Output 48.3.3 Cross Sections of Fitted Regression Surfaces

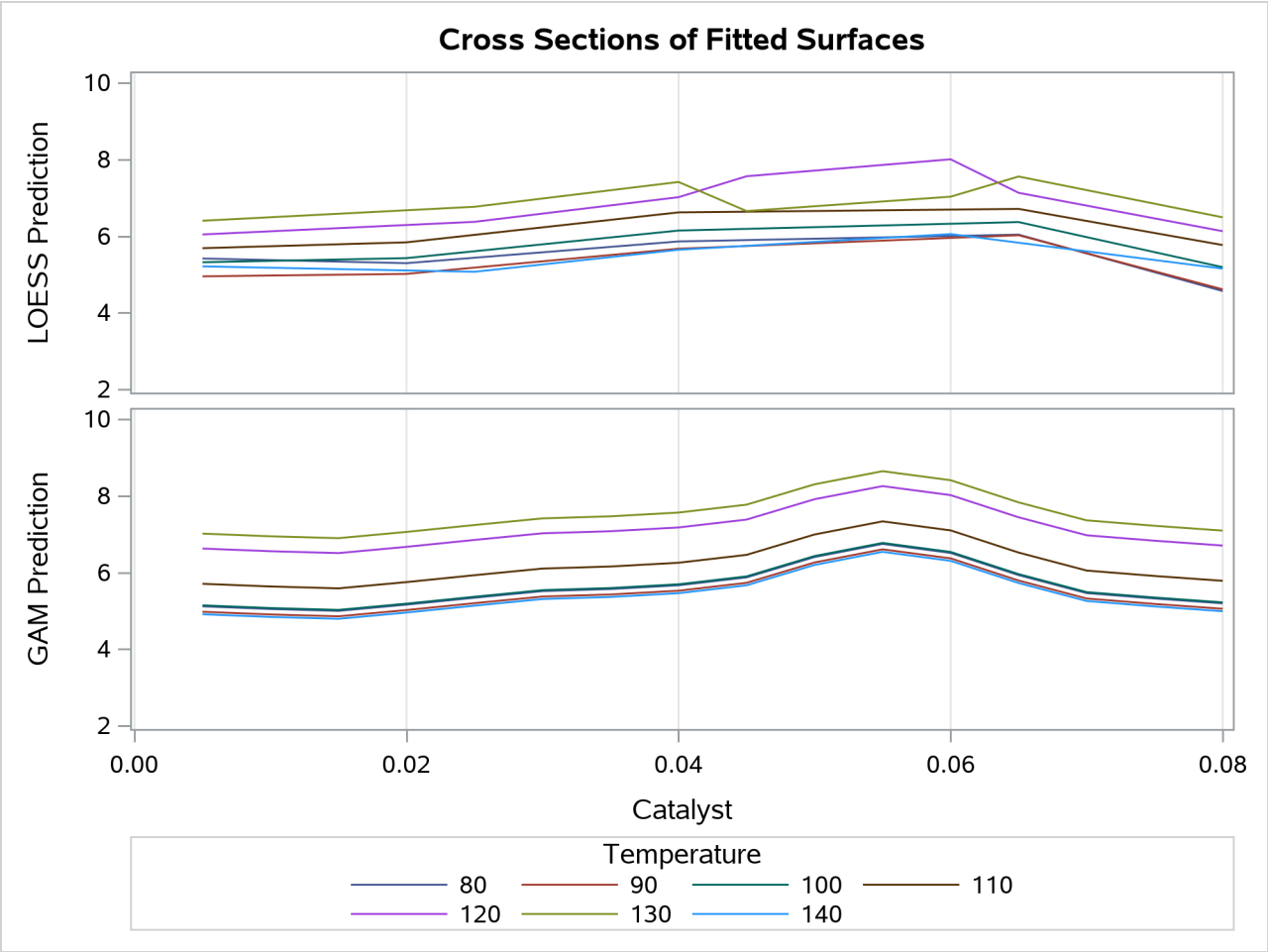

Notice that the cross sections in the top panel (PROC LOESS) of Output 48.3.3 have varying shapes, while every cross section in the bottom panel (PROC GAM) is the same curve shifted vertically. This illustrates precisely the kind of structural differences that distinguish additive models. A second important comparison to make between Output 48.3.2 and Output 48.3.3 is the level of detail in the fitted regression surfaces. Cross sections of the PROC LOESS surface are rather flat, but those of the additive surface have a clear shape. In particular, the ridge near Catalyst=0.055 is only vaguely evident in the PROC LOESS surface, but it is plainly revealed by the additive procedure.

For an example of a situation where unrestricted multidimensional fitting is preferred over additive regression, consider the following simulated data from a similar experiment. The following statements create another SAS data set and plot.

```
data ExperimentB;
   format Temperature f4.0 Catalyst f6.3 Yield f8.3;
   input Temperature Catalyst Yield @@;
   datalines;
80  0.005  9.115  80 0.010  9.275  80 0.015  9.160
80  0.020  7.065  80 0.025  6.054  80 0.030  4.899
80  0.035  4.504  80 0.040  4.238  80 0.045  3.232
80  0.050  3.135  80 0.055  5.100  80 0.060  4.802
```

```
80  0.065  8.218  80 0.070  7.679  80 0.075  9.669
80  0.080  9.071  90 0.005  7.085  90 0.010  6.814
90  0.015  4.009  90 0.020  4.199  90 0.025  3.377
90  0.030  2.141  90 0.035  3.500  90 0.040  5.967
90  0.045  5.268  90 0.050  6.238  90 0.055  7.847
90  0.060  7.992  90 0.065  7.904  90 0.070 10.184
90  0.075  7.914  90 0.080  6.842 100 0.005  4.497
100 0.010  2.565 100 0.015  2.637 100 0.020  2.436
100 0.025  2.525 100 0.030  4.474 100 0.035  6.238
100 0.040  7.029 100 0.045  8.183 100 0.050  8.939
100 0.055  9.283 100 0.060  8.246 100 0.065  6.927
100 0.070  7.062 100 0.075  5.615 100 0.080  4.687
110 0.005  3.706 110 0.010  3.154 110 0.015  3.726
110 0.020  4.634 110 0.025  5.970 110 0.030  8.219
110 0.035  8.590 110 0.040  9.097 110 0.045  7.887
110 0.050  8.480 110 0.055  6.818 110 0.060  7.666
110 0.065  4.375 110 0.070  3.994 110 0.075  3.630
110 0.080  2.685 120 0.005  4.697 120 0.010  4.268
120 0.015  6.507 120 0.020  7.747 120 0.025  9.412
120 0.030  8.761 120 0.035  8.997 120 0.040  7.538
120 0.045  7.003 120 0.050  6.010 120 0.055  3.886
120 0.060  4.897 120 0.065  2.562 120 0.070  2.714
120 0.075  3.141 120 0.080  5.081 130 0.005  8.729
130 0.010  7.460 130 0.015  9.549 130 0.020 10.049
130 0.025  8.131 130 0.030  7.553 130 0.035  6.191
130 0.040  6.272 130 0.045  4.649 130 0.050  3.884
130 0.055  2.522 130 0.060  4.366 130 0.065  3.272
130 0.070  4.906 130 0.075  6.538 130 0.080  7.380
140 0.005  8.991 140 0.010  8.029 140 0.015  8.417
140 0.020  8.049 140 0.025  4.608 140 0.030  5.025
140 0.035  2.795 140 0.040  3.123 140 0.045  3.407
140 0.050  4.183 140 0.055  3.750 140 0.060  6.316
140 0.065  5.799 140 0.070  7.992 140 0.075  7.835
140 0.080  8.985
;

proc sort data=ExperimentB;
   by Temperature Catalyst;
run;

proc sgrender data=ExperimentB template=surface;
   dynamic _X='Temperature' _Y='Catalyst' _Z='Yield' _T='Raw Data';
run;
```

A plot of the raw data is displayed in Output 48.3.4.

Output 48.3.4 Raw Data from Experiment B

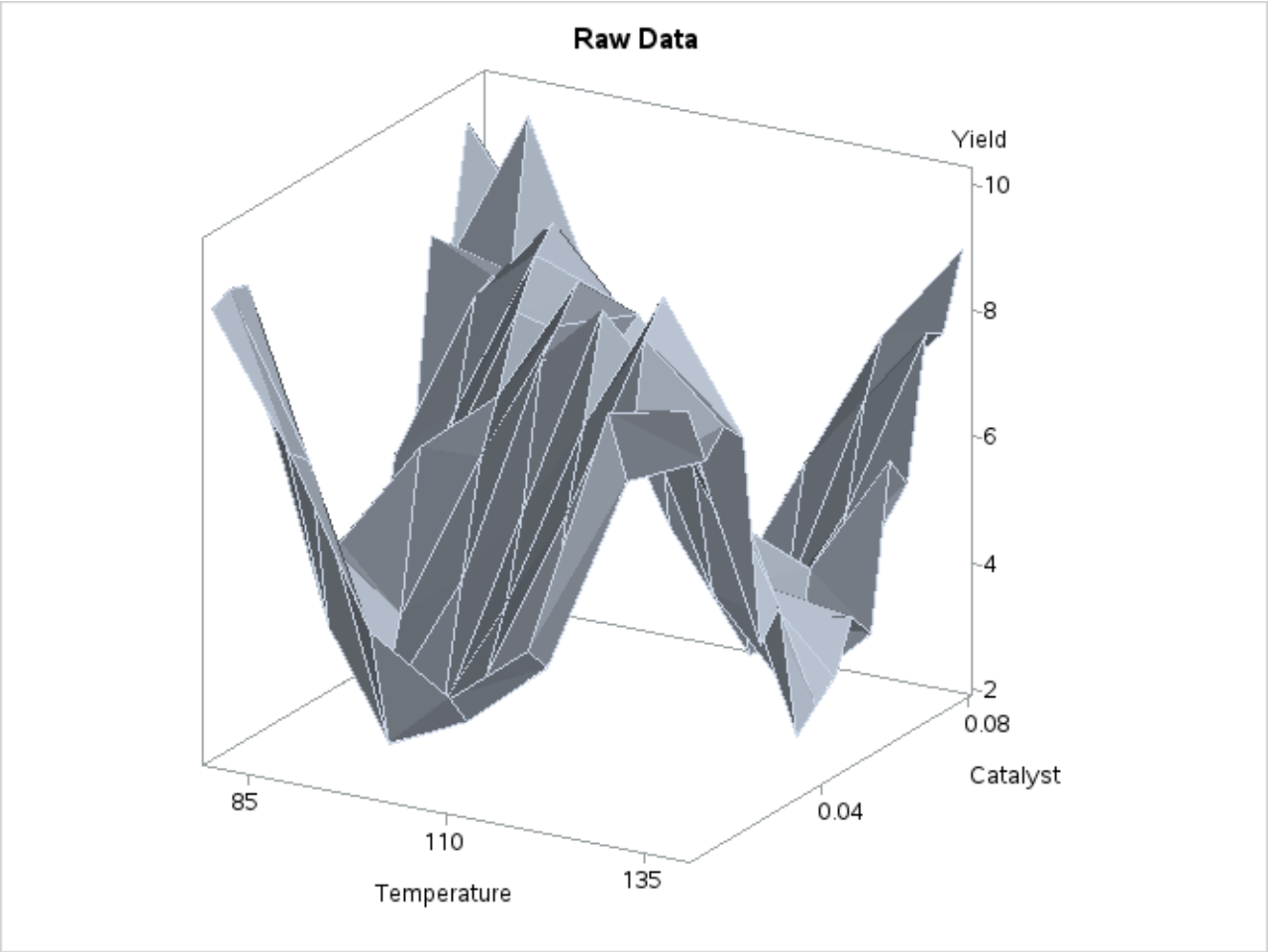

Though the surface displayed in Output 48.3.4 is quite jagged, a distinct feature of the plot is a large ridge that runs diagonally across its surface. One would expect that the ridge would appear in the fitted regression surface of an appropriate nonparametric procedure. Nevertheless, between PROC LOESS and PROC GAM, only PROC LOESS is able to capture this significant feature.

The SAS program for fitting the new data is essentially the same as that for the data set from the first experiment and produces output data set PredGAMb for this experiment. As in Output 48.3.2, multivariate and additive fitted surfaces for these data are displayed in Output 48.3.5.

Output 48.3.5 Fitted Regression Surfaces

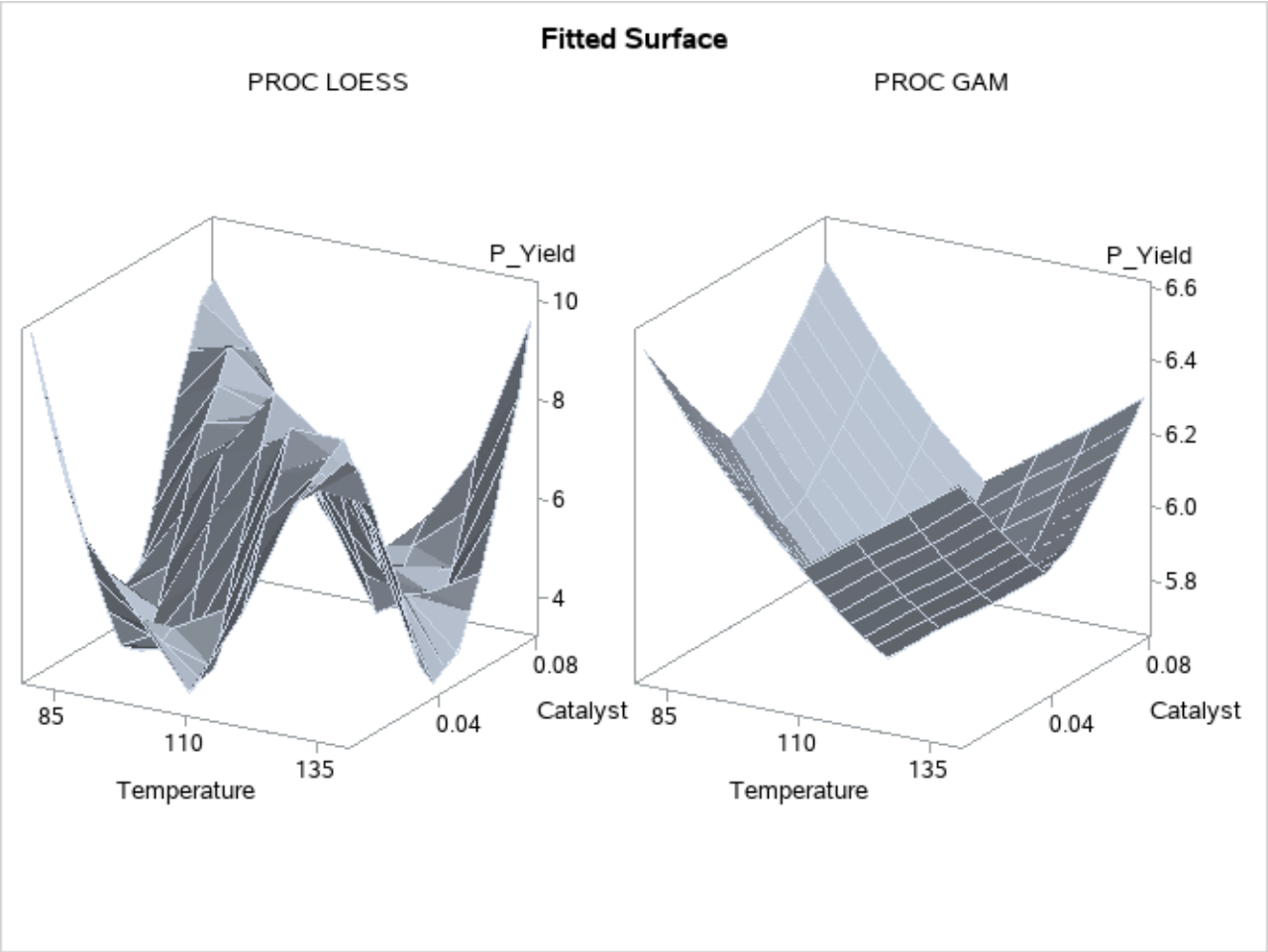

It is clear from Output 48.3.5 that the results of PROC LOESS and PROC GAM are completely different. While the plot in the left panel resembles the raw data plot in Output 48.3.4, the plot in the right panel is essentially featureless.

To understand what is happening, compare the scatter plots of Yield by Catalyst for the two data sets in this example. These are generated by the following statements and displayed in Output 48.3.6.

```
data PredGAM;
   set PredGAM;
   rename Yield=Yield_a;
run;

data PredGAMb;
   set PredGAMb;
   set PredGAM(keep=Yield_a);
run;

proc template;
   define statgraph scatter2;
      dynamic _X _Y1 _Y2;
      begingraph;
```

```
            entrytitle "Scatter Plots of Yield by Catalyst";
            layout lattice/rows=2 columndatarange=unionall
                           rowdatarange=unionall
                           columngutter=15;
               columnAxes;
                  columnAxis / display=all griddisplay=auto_on;
               endColumnAxes;

               layout overlay/
                  xaxisopts=(display=none)
                  yaxisopts=(label="Yield of Experiment A"
                  linearopts=(viewmin=2 viewmax=10));
                  scatterplot x=_X y=_Y1;
               endlayout;

               layout overlay/
                  xaxisopts=(display=none)
                  yaxisopts=(label="Yield of Experiment B"
                  linearopts=(viewmin=2 viewmax=10));
                  scatterplot x=_X y=_Y2;
               endlayout;

            endlayout;
         endgraph;
      end;
   run;

   proc sgrender data=PredGAMb template=scatter2;
      dynamic _X='Catalyst' _Y1='Yield_a' _Y2='Yield';
   run;

   ods graphics off;
```

The top panel of Output 48.3.6 hints at the same kind of structure exhibited in the fitted cross sections of Output 48.3.3. In PROC GAM, the additive model component corresponding to Catalyst is fit to a similar scatter plot, with the partial residuals computed in the backfitting algorithm, so it is able to capture the trend seen here. In contrast, when the second data set is viewed from the perspective of Output 48.3.6, the diagonal ridge apparent in Output 48.3.4 is washed out, and no clear structure shows up in the scatter plot. As a result, the additive model fit produced by PROC GAM is relatively featureless.

Output 48.3.6 Scatter Plots of Yield by Catalyst

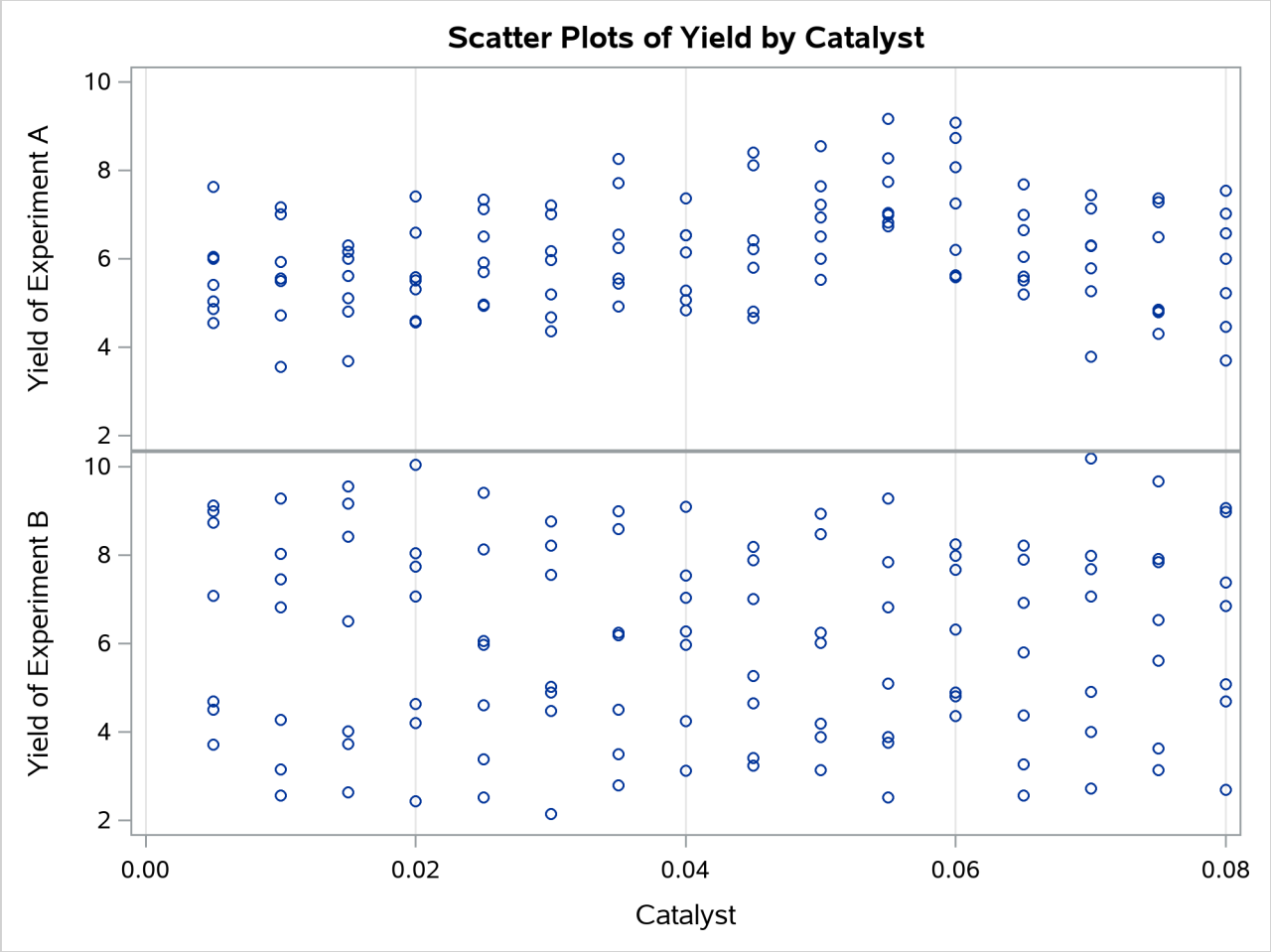

References

Allen, D. M. (1974). "The Relationship between Variable Selection and Data Augmentation and a Method of Prediction." *Technometrics* 16:125–127.

Bell, D. F., Walker, J. L., O'Connor, G., and Tibshirani, R. (1994). "Spinal Deformity after Multiple-Level Cervical Laminectomy in Children." *Spine* 19:406–411.

Buja, A., Hastie, T. J., and Tibshirani, R. (1989). "Linear Smoothers and Additive Models." *Annals of Statistics* 17:453–510.

Cleveland, W. S., Devlin, S. J., and Grosse, E. (1988). "Regression by Local Fitting." *Journal of Econometrics* 37:87–114.

Duchon, J. (1976). "Fonctions-spline et espérances conditionnelles de champs gaussiens [Spline functions and conditional expectations of Gaussian fields]." *Annales scientifiques de l'Université de Clermont-Ferrand 2, Série Mathématique* 14:19–27.

Duchon, J. (1977). "Splines Minimizing Rotation-Invariant Semi-norms in Sobolev Spaces." In *Constructive Theory of Functions of Several Variables*, edited by W. Schempp and K. Zeller, 85–100. New York: Springer-Verlag.

Friedman, J. H., and Stuetzle, W. (1981). "Projection Pursuit Regression." *Journal of the American Statistical Association* 76:817–823.

Hastie, T. J. (1991). "Generalized Additive Models." In *Statistical Models in S*, edited by J. M. Chambers and T. J. Hastie, 249–307. Pacific Grove, CA: Chapman & Hall/CRC.

Hastie, T. J., and Tibshirani, R. J. (1986). "Generalized Additive Models." *Statistical Science* 1:297–318.

Hastie, T. J., and Tibshirani, R. J. (1990). *Generalized Additive Models*. New York: Chapman & Hall.

Houghton, A. N., Flannery, J., and Viola, M. V. (1980). "Malignant Melanoma in Connecticut and Denmark." *International Journal of Cancer* 25:95–104.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Meinguet, J. (1979). "Multivariate Interpolation at Arbitrary Points Made Simple." *Journal of Applied Mathematics and Physics* 30:292–304.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Sochett, E. B., Daneman, D., Clarson, C., and Ehrich, R. M. (1987). "Factors Affecting and Patterns of Residual Insulin Secretion during the First Year of Type I (Insulin Dependent) Diabetes Mellitus in Children." *Diabetologia* 30:453–459.

Stone, C. J. (1985). "Additive Regression and Other Nonparametric Models." *Annals of Statistics* 13:689–705.

Wahba, G. (1983). "Bayesian 'Confidence Intervals' for the Cross Validated Smoothing Spline." *Journal of the Royal Statistical Society, Series B* 45:133–150.

Wahba, G. (1990). *Spline Models for Observational Data*. Philadelphia: SIAM.

Wahba, G., and Wendelberger, J. (1980). "Some New Mathematical Methods for Variational Objective Analysis Using Splines and Cross Validation." *Monthly Weather Review* 108:1122–1145.

Chapter 49
The GAMPL Procedure

Contents

Overview: GAMPL Procedure . . . **3432**
- PROC GAMPL Features . . . 3432
- PROC GAMPL Contrasted with PROC GAM . . . 3433

Getting Started: GAMPL Procedure . . . **3434**

Syntax: GAMPL Procedure . . . **3440**
- PROC GAMPL Statement . . . 3440
- CLASS Statement . . . 3446
- FREQ Statement . . . 3447
- ID Statement . . . 3447
- MODEL Statement . . . 3447
- OUTPUT Statement . . . 3456
- PERFORMANCE Statement . . . 3458
- WEIGHT Statement . . . 3458

Details: GAMPL Procedure . . . **3458**
- Missing Values . . . 3458
- Thin-Plate Regression Splines . . . 3458
- Generalized Additive Models . . . 3461
- Model Evaluation Criteria . . . 3463
- Fitting Algorithms . . . 3464
- Degrees of Freedom . . . 3465
- Model Inference . . . 3466
- Dispersion Parameter . . . 3467
- Tests for Smoothing Components . . . 3467
- Computational Method: Multithreading . . . 3468
- Choosing an Optimization Technique . . . 3469
  - First- or Second-Order Techniques . . . 3469
  - Technique Descriptions . . . 3471
- Displayed Output . . . 3473
- ODS Table Names . . . 3476
- ODS Graphics . . . 3477

Examples: GAMPL Procedure . . . **3478**
- Example 49.1: Scatter Plot Smoothing . . . 3478
- Example 49.2: Nonparametric Logistic Regression . . . 3483
- Example 49.3: Semiparametric Negative Binomial Model for Mackerel Egg Density . 3490
- Example 49.4: Nonparametric Tweedie Regression . . . 3497

References . . . **3502**

Overview: GAMPL Procedure

The GAMPL procedure is a high-performance procedure that fits generalized additive models that are based on low-rank regression splines (Wood 2006). This procedure provides powerful tools for nonparametric regression and smoothing.

Generalized additive models are extensions of generalized linear models. They relax the linearity assumption in generalized linear models by allowing spline terms in order to characterize nonlinear dependency structures. Each spline term is constructed by the thin-plate regression spline technique (Wood 2003). A roughness penalty is applied to each spline term by a smoothing parameter that controls the balance between goodness of fit and the roughness of the spline curve. PROC GAMPL fits models for standard distributions in the exponential family, such as normal, Poisson, and gamma distributions.

PROC GAMPL Features

PROC GAMPL offers the following basic features:

- estimates the regression parameters of a generalized additive model that has fixed smoothing parameters by using penalized likelihood estimation
- estimates the smoothing parameters of a generalized additive model by using either the performance iteration method or the outer iteration method
- estimates the regression parameters of a generalized linear model by using maximum likelihood techniques
- tests the total contribution of each spline term based on the Wald statistic
- provides model-building syntax in the CLASS statement and effect-based parametric effects in the MODEL statement, which are used in other SAS/STAT analytic procedures (in particular, the GLM, LOGISTIC, GLIMMIX, and MIXED procedures)
- provides response-variable options
- enables you to construct a spline term by using multiple variables
- provides control options for constructing a spline term, such as fixed degrees of freedom, initial smoothing parameter, fixed smoothing parameter, smoothing parameter search range, user-supplied knot values, and so on
- provides multiple link functions for any distribution
- provides a WEIGHT statement for weighted analysis
- provides a FREQ statement for grouped analysis
- provides an OUTPUT statement to produce a data set that has predicted values and other observation-wise statistics
- produces graphs via ODS Graphics

Because the GAMPL procedure is a high-performance analytical procedure, it exploits all the available cores and concurrent threads.

PROC GAMPL Contrasted with PROC GAM

Both the GAMPL procedure and the GAM procedure in SAS/STAT software fit generalized additive models. However, the GAMPL procedure uses different approaches for constructing spline basis expansions, fitting generalized additive models, and testing smoothing components. The GAMPL procedure focuses on automatic smoothing parameter selection by using global model-evaluation criteria to find optimal models. The GAM procedure focuses on constructing models by fitting partial residuals against each smoothing term. In general, you should not expect similar results from the two procedures. The following subsections summarize the differences. For more information about the GAM procedure, see Chapter 48, "The GAM Procedure."

Constructing Spline Basis Expansions

The GAMPL procedure uses thin-plate regression splines to construct basis expansions for each spline term, and each term allows multiple variables. The GAM procedure uses univariate or bivariate smoothing splines to construct basis expansions, and each term allows only one or two variables. The thin-plate regression splines that PROC GAMPL uses are low-rank approximations to multivariate smoothing splines. The GAM procedure also allows loess smoothers.

Fitting Generalized Additive Models

The GAMPL procedure fits a generalized additive model that has fixed smoothing parameters by using a global design matrix and a roughness penalty matrix. The GAM procedure uses partial residuals to fit against single smoothing terms. For models that have unknown smoothing parameters, the GAMPL procedure estimates smoothing parameters simultaneously by optimizing global criteria such as generalized cross validation (GCV) and the unbiased risk estimator (UBRE). The GAM procedure estimates each smoothing parameter by optimizing the local criterion GCV one spline term at a time.

Distribution Families and Link Functions

The GAMPL procedure supports all the distribution families and all the link functions that the GAM procedure supports. In addition, PROC GAMPL fits models in the negative binomial and Tweedie families. PROC GAMPL supports any link function for each distribution, whereas PROC GAM supports only the canonical link for each distribution.

Testing Smoothing Components

The GAMPL procedure tests the total contribution for a spline term, including both linear and nonlinear trends. The GAM procedure tests the existence of nonlinearity for a spline term beyond the linear trend.

Model Inference

A global Bayesian posterior covariance matrix is available for models that are fitted by the GAMPL procedure. The confidence limits for prediction of each observation are available, in addition to componentwise confidence limits. For generalized additive models that are fitted by the GAM procedure, only the componentwise confidence limits are available, and they are based on the partial residuals for each smoothing term. The degrees of freedom for generalized additive models that are fitted by the GAMPL procedure is defined as the trace of the degrees-of-freedom matrix. The degrees of freedom for generalized additive models that are fitted by the GAM procedure is approximated by summing the trace of the smoothing matrix for each smoothing term.

Getting Started: GAMPL Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpgamgs1.sas.)

This example concerns the proportions and demographic and geographic characteristics of votes that were cast in 3,107 counties in the United States in the 1980 presidential election. You can use the data set sashelp.Vote1980 directly from the SASHELP library or download it from the StatLib Datasets Archive (Vlachos 1998). For more information about the data set, see Pace and Barry (1997).

The data set contains 3,107 observations and seven variables. The dependent variable LogVoteRate is the logarithm transformation of the proportion of the county population who voted for any candidate. The six explanatory variables are the number of people in the county 18 years of age or older (Pop), the number of people in the county who have a 12th-grade or higher education (Edu), the number of owned housing units (Houses), the aggregate income (Income), and the scaled longitude and latitude of geographic centroids (Longitude and Latitude).

The following statements produce the plot of LogVoteRate with respect to the geographic locations Longitude and Latitude:

```
%let off0 = offsetmin=0 offsetmax=0
            linearopts=(thresholdmin=0 thresholdmax=0);
proc template;
   define statgraph surface;
      dynamic _title _z;
      begingraph / designwidth=defaultDesignHeight;
         entrytitle _title;
         layout overlay / xaxisopts=(&off0) yaxisopts=(&off0);
            contourplotparm z=_z y=Latitude x=Longitude / gridded=FALSE;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=sashelp.Vote1980 template=surface;
   dynamic _title = 'US County Vote Proportion in the 1980 Election'
           _z      = 'LogVoteRate';
run;
```

Figure 49.1 shows the map of the logarithm transformation of the proportion of the county population who voted for any candidate in the 1980 US presidential election.

Figure 49.1 US County Vote Proportion in the 1980 Election

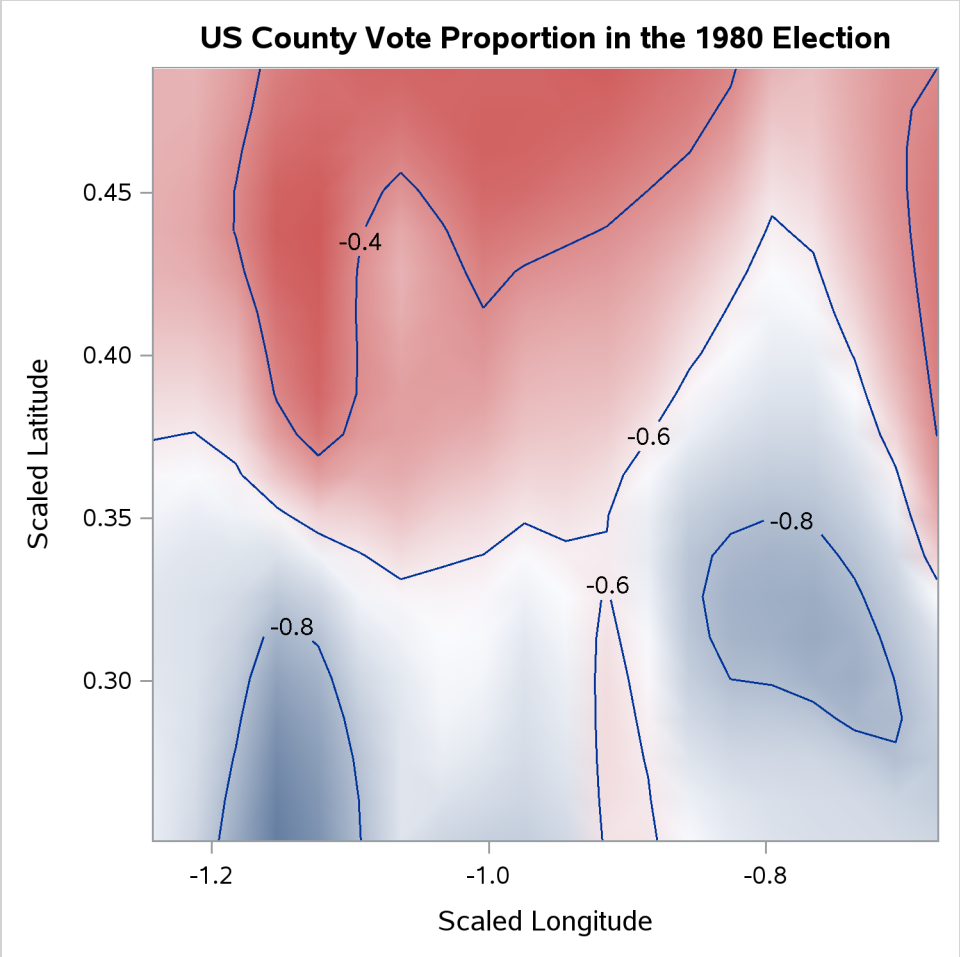

The objective is to explore the nonlinear dependency structure between the dependent variable and demographic variables (Pop, Edu, Houses, and Income), in addition to the spatial variations on geographic variables (Longitude and Latitude). The following statements use thin-plate regression splines to fit a generalized additive model:

```
ods graphics on;

proc gampl data=sashelp.Vote1980 plots seed=12345;
   model LogVoteRate = spline(Pop   ) spline(Edu) spline(Houses)
                       spline(Income) spline(Longitude Latitude);
   id Longitude Latitude;
   output out=VotePred;
run;
```

With ODS Graphics enabled by the first statement, the PLOTS option in the PROC GAMPL statement requests a smoothing component panel of fitted spline terms. The SEED option specifies the random seed so that you can reproduce the analysis.

The default output from this analysis is presented in Figure 49.2 through Figure 49.10.

The "Performance Information" table in Figure 49.2 shows that PROC GAMPL executed in single-machine mode (that is, on the server where SAS is installed). When high-performance procedures run in single-machine mode, they use concurrently scheduled threads. In this case, four threads were used.

Figure 49.2 Performance Information

The GAMPL Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Figure 49.3 displays the "Model Information" table. The response variable LogVoteRate is modeled by using a normal distribution whose mean is modeled by an identity link function. The GAMPL procedure uses the performance iteration method and the GCV criterion as the fitting criterion. PROC GAMPL searches for the optimum smoothing parameters by using the Newton-Raphson algorithm to optimize the fitting criterion. The random number seed is set to 12,345. Random number generation is used for sampling from observations to form spline knots and truncated eigendecomposition. Changing the random number seed might yield slightly different model fits.

Figure 49.3 Model Information

Model Information	
Data Source	SASHELP.VOTE1980
Response Variable	LogVoteRate
Distribution	Normal
Link Function	Identity
Fitting Method	Performance Iteration
Fitting Criterion	GCV
Optimization Technique for Smoothing	Newton-Raphson
Random Number Seed	12345

Figure 49.4 displays the "Number of Observations" table. All 3,107 observations in the data set are used in the analysis. For data sets that have missing or invalid values, the number of used observations might be less than the number of observations read.

Figure 49.4 Number of Observations

Number of Observations Read	3107
Number of Observations Used	3107

Figure 49.5 displays the convergence status of the performance iteration method.

Figure 49.5 Convergence Status

The performance iteration converged after 3 steps.

Figure 49.6 shows the "Fit Statistics" table. The penalized log likelihood and the roughness penalty are

displayed. You can use effective degrees of freedom to compare generalized additive models with generalized linear models that do not have spline transformations. Information criteria such as Akaike's information criterion (AIC), Akaike's bias-corrected information criterion (AICC), and Schwarz Bayesian information criterion (BIC) can also be used for comparisons. These criteria penalize the –2 log likelihood for effective degrees of freedom. The GCV criterion is used to compare against other generalized additive models or models that are penalized.

Figure 49.6 Fit Statistics

Fit Statistics	
Penalized Log Likelihood	2729.51482
Roughness Penalty	0.25787
Effective Degrees of Freedom	48.70944
Effective Degrees of Freedom for Error	3055.44725
AIC (smaller is better)	-5361.86863
AICC (smaller is better)	-5360.28467
BIC (smaller is better)	-5067.59479
GCV (smaller is better)	0.01042

.

The "Parameter Estimates" table in Figure 49.7 shows the regression parameter and dispersion parameter estimates. In this model, the intercept is the only regression parameter because (1) all variables are characterized by spline terms and no parametric effects are present and (2) the intercept absorbs the constant effect that is extracted from each spline term to make fitted splines identifiable. The dispersion parameter is estimated by maximizing the likelihood, given other model parameters.

Figure 49.7 Regression Parameter Estimates

Parameter Estimates					
Parameter	**DF**	**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Intercept	1	-0.576234	0.001803	102119.645	<.0001
Dispersion	1	0.010103	0.014287		

The "Estimates for Smoothing Components" table is shown in Figure 49.8. For each spline term, the effective degrees of freedom, the estimated smoothing parameter, and the corresponding roughness penalty are displayed. The table also displays additional information about spline terms, such as the number of parameters, penalty matrix rank, and number of spline knots.

Figure 49.8 Estimates for Smoothing Components

Estimates for Smoothing Components						
Component	**Effective DF**	**Smoothing Parameter**	**Roughness Penalty**	**Number of Parameters**	**Rank of Penalty Matrix**	**Number of Knots**
Spline(Pop)	7.80559	0.0398	0.0114	9	10	2000
Spline(Edu)	7.12453	0.2729	0.0303	9	10	2000
Spline(Houses)	7.20940	0.1771	0.0370	9	10	2000
Spline(Income)	5.92854	0.7498	0.0488	9	10	2000
Spline(Longitude Latitude)	18.64138	0.000359	0.1304	19	20	2000

Figure 49.9 displays the hypothesis testing results for each smoothing component. The null hypothesis for each spline term is whether the total dependency on each variable is 0. The effective degrees of freedom for both fit and test is displayed.

Figure 49.9 Tests for Smoothing Components

Tests for Smoothing Components

Component	Effective DF	Effective DF for Test	F Value	Pr > F
Spline(Pop)	7.80559	8	1443.64	<.0001
Spline(Edu)	7.12453	8	153.94	<.0001
Spline(Houses)	7.20940	8	1213.94	<.0001
Spline(Income)	5.92854	7	43.17	<.0001
Spline(Longitude Latitude)	18.64138	19	1619.15	<.0001

Figure 49.10 displays the "Smoothing Component Panel" for all the spline terms used in the model. It displays predicted spline curves and 95% Bayesian posterior confidence bands for each univariate spline term.

Figure 49.10 Smoothing Component Panel

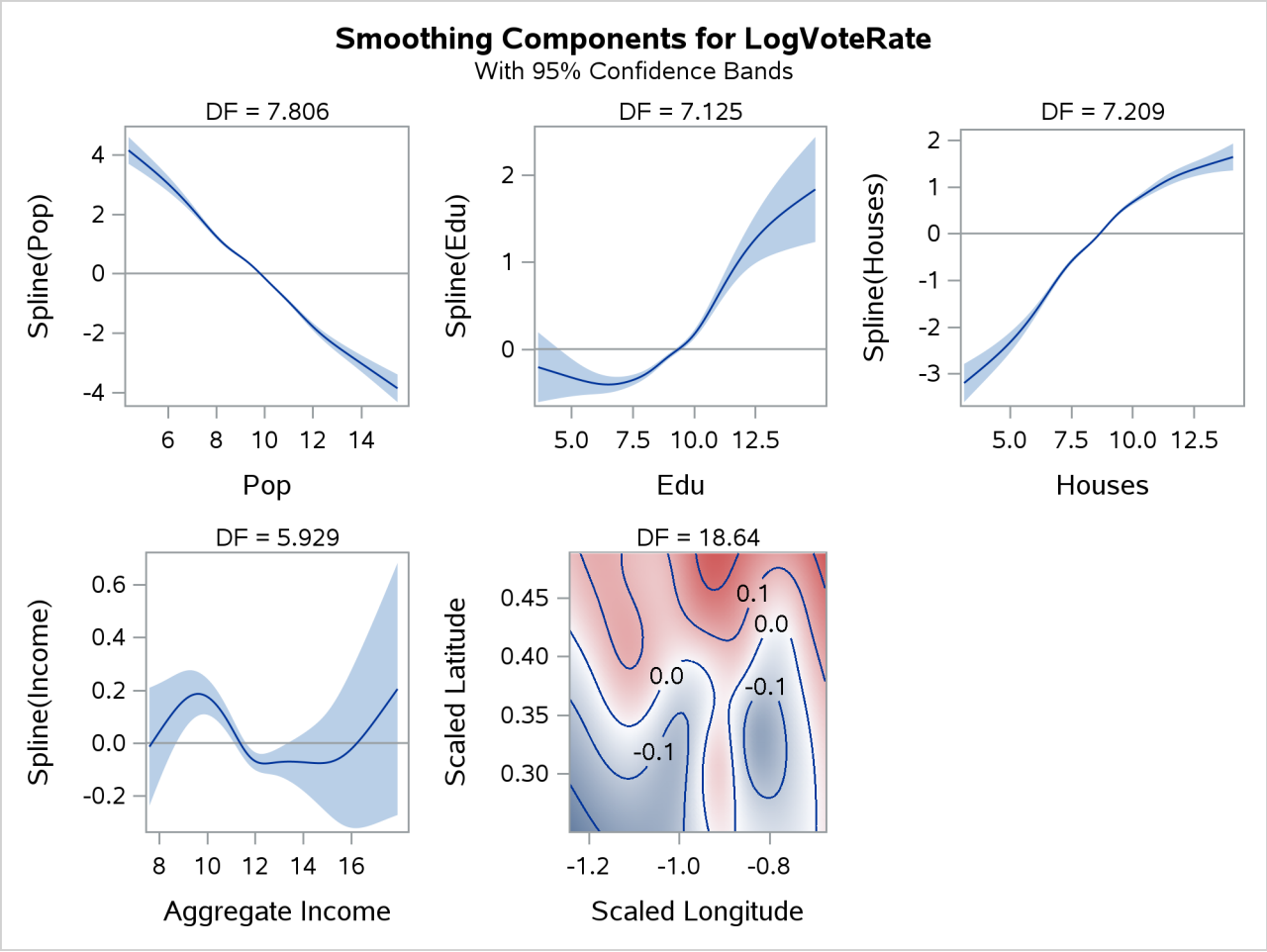

The preceding program contains an ID statement and an OUTPUT statement. The ID statement transfers the specified variables (Longitude and Latitude) from the input data set, sashelp.Vote1980, to the output data set, VotePred. The OUTPUT statement requests the prediction for each observation by default and saves the results in the data set VotePred. The following run of the SGRENDER procedure produces the fitted surface of the log vote proportion in the 1980 presidential election:

```
proc sgrender data=VotePred template=surface;
   dynamic _title='Predicted US County Vote Proportion in the 1980 Election'
           _z    ='Pred';
run;
```

Figure 49.11 shows the map of predictions of the logarithm transformation of the proportion of county population who voted for any candidates in the 1980 US presidential election from the fitted generalized additive model.

Figure 49.11 Predicted US County Vote Proportion in the 1980 Election

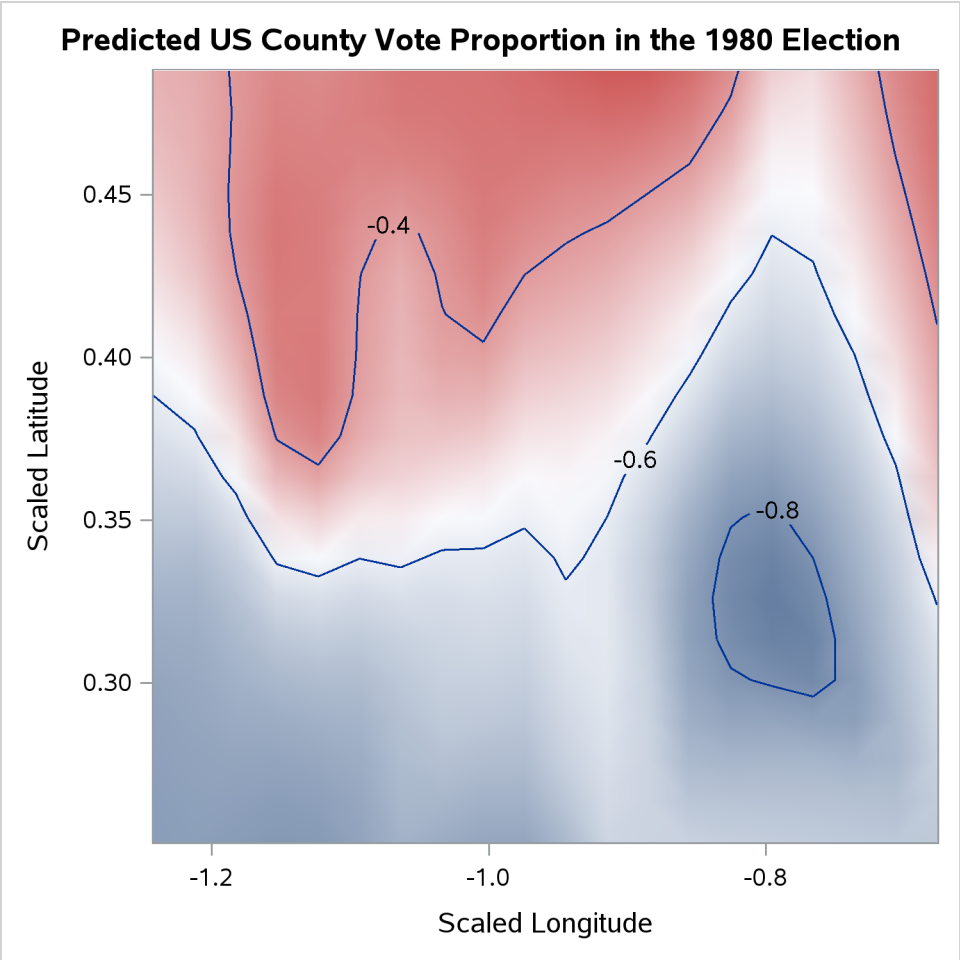

Compared to the map of the logarithm transformations of the proportion of votes cast shown in Figure 49.1, the map of the predictions of the logarithm transformations of the proportion of votes cast has a smoother surface.

Syntax: GAMPL Procedure

The following statements are available in the GAMPL procedure:

```
PROC GAMPL <options> ;
    CLASS variable <(options)>... <variable <(options)>> </global-options> ;
    MODEL response <(response-options)> = <PARAM(effects)>
            <spline-effects> </model-options> ;
    MODEL events / trials = <PARAM(effects)> <spline-effects> </model-options> ;
    OUTPUT <OUT=SAS-data-set> <keyword <=name>>... <keyword <=name>> </options> ;
    PERFORMANCE performance-options ;
    FREQ variable ;
    ID variables ;
    WEIGHT variable ;
```

The PROC GAMPL statement and at least one MODEL statement are required. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statements. The following sections describe the PROC GAMPL statement and then describe the other statements in alphabetical order.

PROC GAMPL Statement

```
PROC GAMPL <options> ;
```

The PROC GAMPL statement invokes the procedure. Table 49.1 summarizes the available options in the PROC GAMPL statement by function. The options are then described fully in alphabetical order.

Table 49.1 PROC GAMPL Statement Options

Option	Description
Basic Options	
ALPHA=	Specifies a global significance level
DATA=	Specifies the input data set
NAMELEN=	Limits the length of effect names
SEED=	Sets the seed for pseudorandom number generation
Display Options	
ITDETAILS	Displays the “Iteration History” table
NOCLPRINT	Limits or suppresses the display of classification variable levels
NOPRINT	Suppresses ODS output
PLOTS=	Controls plots that are produced through ODS Graphics
Optimization Subject Options	
PLIKEOPTIONS	Sets optimization parameters for likelihood estimation
SMOOTHOPTIONS	Sets optimization parameters for smoothing parameter estimation

Table 49.1 *continued*

Option	Description
Tolerance Options	
SINGCHOL=	Tunes the singularity criterion for Cholesky decompositions
SINGSWEEP=	Tunes the singularity criterion for the sweep operator
SINGULAR=	Tunes the general singularity criterion
User-Defined Format Options	
FMTLIBXML=	Specifies the file reference for a format stream

You can specify the following *options* in the PROC GAMPL statement.

ALPHA=*number*

specifies a global significance level for the hypothesis testing of smoothing components and the construction of Bayesian confidence bands of predictions. The confidence level is 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05. You can override this global significance level for Bayesian confidence bands of predictions by specifying the ALPHA= option in the OUTPUT statement.

DATA=*SAS-data-set*

names the input SAS data set for PROC GAMPL to use. The default is the most recently created data set.

FMTLIBXML=*file-ref*

specifies the file reference for the XML stream that contains user-defined format definitions. For information about how to generate an XML stream for your formats, see the section "Working with Formats" on page 568.

ITDETAILS

adds to the "Iteration History" table the current values of the parameter estimates and their gradients. If the optimization algorithm is used to determine at least one smoothing parameter, the table lists values for smoothing parameters. If all smoothing parameters are fixed or a parametric generalized linear model is specified, the table lists values for regression parameters. These quantities are reported only for parameters that participate in the optimization.

NAMELEN=*number*

specifies the length to which long effect names are shortened. The value of *number* must be an integer greater than or equal to 20. By default, NAMELEN=20.

NOCLPRINT<=*number***>**

suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying a *number* helps reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOPRINT
suppresses the generation of ODS output.

PLIKEOPTIONS(*optimization-parameters***)**
sets optimization parameters for either maximum or penalized likelihood estimation. For more information about how to specify *optimization-parameters*, see the section "Optimization Parameters" on page 3443.

PLOTS < (*global-plot-option***) > < = *plot-requests* < (***option***) > >**
controls the plots that are produced through ODS Graphics. When ODS Graphics is enabled, PROC GAMPL produces by default a panel of plots of partial prediction curves or surfaces of smoothing components.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc gampl plots;
   model y=spline(x1) spline(x2);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *global-plot-option*, which applies to the smoothing component plots that the GAMPL procedure generates:

UNPACK | UNPACKPANEL
suppresses paneling. By default, multiple smoothing component plots can appear in some output panels. Specify UNPACK to get each plot individually.

You can specify the following *plot-requests*:

ALL
requests that all default plots be produced.

COMPONENTS < (*component-option***) >**
plots a panel of smoothing components of the fitted model. You can specify the following *component-option*:

COMMONAXES
requests that the smoothing component plots use a common vertical axis except for bivariate contour plots. This option enables you to visually judge the relative effect size.

NONE
suppresses all plots.

SEED=*number*
specifies an integer that is used to start the pseudorandom number generator for subset sampling from observations to form knots if necessary and for truncated eigendecomposition. If you do not specify this option, or if *number* ≤ 0, the seed is generated from the time of day, which is read from the computer's clock.

SINGCHOL=*number*
tunes the singularity criterion in Cholesky decompositions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGSWEEP=*number*
tunes the singularity criterion in sweep operations that determine collinearity between spline basis expansions. The default is 1E–8.

SINGULAR=*number*
tunes the general singularity criterion that is applied in sweep and inversion operations. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SMOOTHOPTIONS(*optimization-parameters***)**
specifies optimization parameters for smoothing parameter estimation. For more information about how to specify *optimization-parameters*, see the section "Optimization Parameters" on page 3443.

Optimization Parameters

You can specify *optimization-parameters* for both the PLIKEOPTIONS and SMOOTHOPTIONS options. Depending on the modeling context, some optimization parameters might have no effect. For parametric generalized linear models or generalized additive models that have fixed smoothing parameters, any optimization parameters that you specify in the SMOOTHOPTIONS option are ignored. For the performance iteration method, only the ABSFCONV=, FCONV=, and MAXITER= options are effective for PLIKEOPTIONS. The optimization algorithm is considered to have converged when any one of the convergence criteria that are specified in *optimization-parameters* is satisfied. Table 49.2 lists the available optimization parameters for both the PLIKEOPTIONS and SMOOTHOPTIONS options.

Table 49.2 Optimization Parameters

Option	Description
ABSCONV=	Tunes the absolute function convergence criterion
ABSFCONV=	Tunes the absolute function difference convergence criterion
ABSGCONV=	Tunes the absolute gradient convergence criterion
FCONV=	Tunes the relative function difference convergence criterion
GCONV=	Tunes the relative gradient convergence criterion
MAXFUNC=	Specifies the maximum number of function evaluations in any optimization
MAXITER=	Chooses the maximum number of iterations in any optimization
MAXTIME=	Specifies the upper limit of CPU time (in seconds) for any optimization
MINITER=	Specifies the minimum number of iterations in any optimization
TECHNIQUE=	Selects the optimization technique

You can specify the following *optimization-parameters*:

ABSCONV=*r*

ABSTOL=*r*

specifies an absolute function convergence criterion. For minimization, termination requires $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of r is the negative square root of the largest double-precision value, which serves only as a protection against overflow.

ABSFCONV=*r* **<** *n* **>**

ABSFTOL=*r* **<** *n* **>**

specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, termination requires a small change of the function value in successive iterations,

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}(k)$ is defined as the vertex that has the lowest function value and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process terminates. By default, ABSFCONV=0.

ABSGCONV=*r* **<** *n* **>**

ABSGTOL=*r* **<** *n* **>**

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small,

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the jth parameter. This criterion is not used by the NMSIMP technique. The default value is r = 1E–8. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process terminates.

FCONV=*r* **<** *n* **>**

FTOL=*r* **<** *n* **>**

specifies a relative function difference convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex that has the lowest function value and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex.

The default value is $r = 2 \times \epsilon$, where ϵ is the machine precision. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process terminates.

GCONV=*r* < *n* >

GTOL=*r* < *n* >

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, termination requires that the normalized predicted function reduction be small,

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the CONGRA technique (where a reliable Hessian estimate, $\mathbf{H}$, is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \; |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The default value is r = 1E–8. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process terminates.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values are as follows, depending on the optimization technique that you specify in the TECHNIQUE= option:

- TRUREG, NRRIDG, NEWRAP: $n = 125$
- QUANEW, DBLDOG: $n = 500$
- CONGRA: $n = 1{,}000$
- NMSIMP: $n = 3{,}000$

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed n.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on the optimization technique that you specify in the TECHNIQUE option:

- TRUREG, NRRIDG, NEWRAP: $n = 50$
- QUANEW, DBLDOG: $n = 200$
- CONGRA: $n = 400$
- NMSIMP: $n = 1{,}000$

These default values also apply when n is specified as a missing value.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The default value is the largest floating-point double representation of your computer. The time that you specify in this option is checked only once at the end of each iteration. Therefore, the actual running time can be longer than *r*.

MINITER=*n*

MINIT=*n*

specifies the minimum number of iterations. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms might behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations. By default, MINITER=0.

TECHNIQUE=*keyword*

TECH=*keyword*

specifies the optimization technique for obtaining smoothing parameter estimates and regression parameter estimates. You can choose from the following techniques:

CONGRA	performs a conjugate-gradient optimization.
DBLDOG	performs a version of double-dogleg optimization.
NEWRAP	performs a Newton-Raphson optimization with line search.
NMSIMP	performs a Nelder-Mead simplex optimization.
NONE	performs no optimization.
NRRIDG	performs a Newton-Raphson optimization with ridging.
QUANEW	performs a dual quasi-Newton optimization.
TRUREG	performs a trust-region optimization

By default, TECHNIQUE=NEWRAP for the performance iteration (METHOD=PERFORMANCE), and TECHNIQUE=QUANEW for the outer iteration (METHOD=OUTER).

For more information, see the section "Choosing an Optimization Technique" on page 3469.

CLASS Statement

CLASS *variable* < (*options*) >. . . < *variable* < (*options*) > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. The CLASS statement must precede the MODEL statement. You can list the response variable for binary models in the CLASS statement, but this is not necessary.

The CLASS statement is documented in the section "CLASS Statement" on page 572.

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. PROC GAMPL treats each observation as if it appeared f times, where the frequency value f is the value of the FREQ variable for the observation. If f is not an integer, then f is truncated to an integer. If f is less than 1 or missing, the observation is not used in the analysis. When you do not specify the FREQ statement, each observation is assigned a frequency of 1.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are to be transferred to the output data set that you specify in the OUTPUT statement.

For more information, see the section "ID Statement" on page 576.

MODEL Statement

MODEL *response* < **(***response-options***)** > **=** < **PARAM(***effects***)** > < *spline-effects* > < / *model-options* > **;**

MODEL *events* / *trials* **=** < **PARAM(***effects***)** > < *spline-effects* > < / *model-options* > **;**

The MODEL statement specifies the response (dependent or target) variable and the predictor (independent or explanatory) effects of the model. You can specify the response in the form of a single variable or in the form of a ratio of two variables, which are denoted *events/trials*. The first form applies to all distribution families; the second form applies only to summarized binomial response data. When you have binomial data, the *events* variable contains the number of positive responses (or events) and the *trials* variable contains the number of trials. The values of both *events* and (*trials* – *events*) must be nonnegative, and the value of *trials* must be positive. If you specify a single *response* variable that is in a CLASS statement, then the response is assumed to be binary.

You can specify parametric effects that are constructed from variables in the input data set and include the effects in the parentheses of a PARAM( ) option, which can appear multiple times. For information about constructing the model effects, see the section "Parameterization of Model Effects" on page 412.

You can specify *spline-effects* by including independent variables inside the parentheses of the SPLINE( ) option. Only continuous variables (not classification variables) can be specified in *spline-effects*. Each *spline-effect* can have at least one variable and optionally some *spline-options*. You can specify any number of *spline-effects*. The following table shows some examples.

Spline Effect Specification	Meaning
`Spline(x)`	Constructs the univariate spline with x and uses the observed data points as knots. The maximum degrees of freedom is 10. PROC GAMPL uses an optimization algorithm to determine the optimal smoothing parameter.
`Spline(x1/knots=list(1 to 10))`	Constructs the univariate spline by using x1 and a supplied list of knots from 1 to 10. PROC GAMPL uses an optimization algorithm to determine the optimal smoothing parameter.
`Spline(x2 x3/smooth=0.3)`	Constructs the bivariate spline by using x2 and x3 and a fixed smoothing parameter 0.3.
`Spline(x4 x5 x6/maxdf=40)`	Constructs the trivariate spline by using x4, x5, and x6 and a maximum of 40 degrees of freedom. PROC GAMPL uses an optimization algorithm to determine the optimal smoothing parameter.

Both parametric effects and spline effects are optional. If none are specified, a model that contains only an intercept is fitted. If only parametric effects are present, PROC GAMPL fits a parametric generalized linear model by using the terms inside the parentheses of all PARAM( ) terms. If only spline effects are present, PROC GAMPL fits a nonparametric additive model. If both types of effects are present, PROC GAMPL fits a semiparametric model by using the parametric effects as the linear part of the model.

There are three sets of options in the MODEL statement. The *response-options* determine how the GAMPL procedure models probabilities for binary data. The *spline-options* controls how each spline term forms basis expansions. The *model-options* control other aspects of model formation and inference. Table 49.4 summarizes these options.

Table 49.4 MODEL Statement Options

Option	Description
Response Variable Options for Binary Models	
DESCENDING	Reverses the response categories
EVENT=	Specifies the event category
ORDER=	Specifies the sort order
REF=	Specifies the reference category
Smoothing Options for Spline Effects	
DETAILS	Requests detailed spline information
DF=	Specifies the fixed degrees of freedom
INITSMOOTH=	Specifies the starting value for the smoothing parameter
KNOTS=	Specifies the knots to be used for constructing the spline
M=	Specifies polynomial orders for constructing the spline
MAXDF=	Specifies the maximum degrees of freedom
MAXKNOTS=	Specifies the maximum number of knots to be used for constructing the spline
MAXSMOOTH=	Specifies the upper bound for the smoothing parameter

Table 49.4 *continued*

Option	Description
MINSMOOTH=	Specifies the lower bound for the smoothing parameter
SMOOTH=	Specifies a fixed smoothing parameter
Model Options	
ALLOBS	Requests all nonmissing values of spline variables for constructing spline basis functions regardless of other model variables
CRITERION=	Specifies the model evaluation criterion
DISPERSION | PHI=	Specifies the fixed dispersion parameter
DISTRIBUTION | DIST=	Specifies the response distribution
FDHESSIAN	Requests a finite-difference Hessian for smoothing parameter selection
INITIALPHI=	Specifies the starting value of the dispersion parameter
LINK=	Specifies the link function
NORMALIZE	Requests normalized spline basis functions for model fitting
MAXPHI=	Specifies the upper bound for searching the dispersion parameter
METHOD=	Specifies the algorithm for selecting smoothing parameters
MINPHI=	Specifies the lower bound for searching the dispersion parameter
OFFSET=	Specifies the offset variable
RIDGE=	Specifies the ridge parameter
SCALE=	Specifies the method for estimating the dispersion parameter

Response Variable Options

Response variable options determine how the GAMPL procedure models probabilities for binary data.

You can specify the following *response-options* by enclosing them in parentheses after the *response* variable.

DESCENDING

DESC

reverses the order of the response categories. If you specify both the DESCENDING and ORDER= options, PROC GAMPL orders the response categories according to the ORDER= option and then reverses that order.

EVENT=‘*category*’ | FIRST | LAST

specifies the event category for the binary response model. PROC GAMPL models the probability of the event category. The EVENT= option has no effect when there are more than two response categories.

You can specify any of the following:

‘*category*’

specifies that observations whose value matches *category* (formatted, if a format is applied) in quotation marks represent events in the data. For example, the following statements specify that observations that have a formatted value of ‘1’ represent events in the data. The probability that is modeled by the GAMPL procedure is thus the probability that the variable def takes on the (formatted) value ‘1’.

```
proc gampl data=MyData;
   class A B C;
   model def(event ='1') = param(A B C) spline(x1 x2 x3);
run;
```

FIRST
designates the first ordered category as the event.

LAST
designates the last ordered category as the event.

By default, EVENT=FIRST.

ORDER=DATA | FORMATTED | INTERNAL
ORDER=FREQ | FREQDATA | FREQFORMATTED | FREQINTERNAL
specifies the sort order for the levels of the *response* variable. When ORDER=FORMATTED (the default) for numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC GAMPL run or in the DATA step that created the data set), the levels are ordered by their internal (numeric) value. Table 49.5 shows the interpretation of the ORDER= option.

Table 49.5 Sort Order

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count (levels that have the most observations come first in the order)
FREQDATA	Order of descending frequency count; within counts by order of appearance in the input data set when counts are tied
FREQFORMATTED	Order of descending frequency count; within counts by formatted value when counts are tied
FREQINTERNAL	Order of descending frequency count; within counts by unformatted value when counts are tied
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For the FORMATTED and INTERNAL orders, the sort order is machine-dependent.

For more information about sort order, see the chapter about the SORT procedure in *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Language Reference: Concepts*.

REF='*category*' | **FIRST** | **LAST**

specifies the reference category for the binary response model. Specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify any of the following:

'*category*'

specifies that observations whose value matches *category* (formatted, if a format is applied) are designated as the reference.

FIRST

designates the first ordered category as the reference

LAST

designates the last ordered category as the reference.

By default, REF=LAST.

Spline Options

DETAILS

requests a detailed spline specification information table.

DF=*number*

specifies a fixed degrees of freedom. When you specify this option, no smoothing parameter selection is performed on the spline term. If *number* is not an integer, then *number* is truncated to an integer.

INITSMOOTH=*number*

specifies the starting value for a smoothing parameter. The *number* must be nonnegative.

KNOTS=*method*

specifies the *method* for supplying user-defined knot values instead of using data values for constructing basis expansions. You can use the following *methods* for supplying the knots:

LIST(*list-of-values***)**

specifies a list of values as knots for the spline construction. For a multivariate spline term, the listed values are taken as multiple row vectors, where each vector has values that are ordered by specified variables. If the last row vector of knots contains fewer values than the number of variables, then the last row vector is ignored. For example, the following specification of a spline term produces two actual knot vectors ($\boldsymbol{k}_1$ and $\boldsymbol{k}_2$) and the value 5 is ignored.

```
spline(x1 x2/knots=list(1 2 3 4 5))
```

Table 49.6 Knot Values for a Bivariate Spline with a Supplied List

	x1	x2
$\boldsymbol{k}_1$	1	2
$\boldsymbol{k}_2$	3	4

EQUAL(*n*)

specifies the number of equally spaced interior knots for every variable in a spline term. Two boundary knots are automatically added to the knot list for each variable such that the total number of knots is $(n + 2)^d$, where d is the number of variables in the spline term. For a multivariate spline term, knot values for each variable are determined independently from the corresponding boundary values. For example, if the boundary points for x1 are 1 and 5 and the boundary points for x2 are 2 and 6, then the following specification of a spline term produces nine actual knots ($\boldsymbol{k}_1 — \boldsymbol{k}_9$), which consist of two boundary knots and one interior knot for each variable.

```
spline(x1 x2/knots=equal(1))
```

Table 49.7 Knot Values for a Bivariate Spline with One Interior Knot

	x1	x2
$\boldsymbol{k}_1$	1	2
$\boldsymbol{k}_2$	1	4
$\boldsymbol{k}_3$	1	6
$\boldsymbol{k}_4$	3	2
$\boldsymbol{k}_5$	3	4
$\boldsymbol{k}_6$	3	6
$\boldsymbol{k}_7$	5	2
$\boldsymbol{k}_8$	5	4
$\boldsymbol{k}_9$	5	6

M=*number*

specifies the order of the derivative in the penalty term. The *number* must be a positive integer. The default is $\max(2, \mathrm{int}(d/2) + 1)$, where d is the number of variables in the spline term.

MAXDF=*number*

specifies the maximum *number* of degrees of freedom. When a thin-plate regression spline is formed, the specified *number* is used for constructing a low-rank penalty matrix to approximate the penalty matrix via the truncated eigendecomposition. The *number* must be greater than $\binom{m+d-1}{d}$, where m is the derivative order that is specified in the M= option. The default is $10 \times d$. For more information, see the section "Thin-Plate Regression Splines" on page 3458.

MAXKNOTS=*number*

specifies the maximum *number* of knots if data points are used to form knots. If KNOTS=LIST(*list-of-values*) is not specified, PROC GAMPL forms knots from unique data points. If the number of unique data points is greater than *number*, a subset of size *number* is formed by random sampling from all unique data points. The *number* cannot exceed the largest integer that can be stored on the given machine. By default, MAXKNOTS=2000.

MAXSMOOTH=*number*
specifies the upper bound for the smoothing parameter. The default is the largest double-precision value.

MINSMOOTH=*number*
specifies the lower bound for the smoothing parameter. By default, MINSMOOTH=0.

SMOOTH=*number*
specifies a fixed smoothing parameter. When you specify this option, no smoothing parameter selection is performed on the spline term.

Model Options

ALLOBS
requests that all nonmissing values of the variables be specified in a spline term for constructing the spline basis functions, regardless of whether other model variables are missing.

CRITERION=*criterion*
specifies the model evaluation criterion for selecting smoothing parameters for *spline-effects*. You can specify the following values:

GACV< (FACTOR=*number* | **GAMMA=***number***) >**
uses the generalized approximate cross validation (GACV) criterion to evaluate models.

GCV< (FACTOR=*number* | **GAMMA=***number***) >**
uses the generalized cross validation (GCV) criterion to evaluate models.

UBRE< (FACTOR=*number* | **GAMMA=***number***) >**
uses the unbiased risk estimator (UBRE) criterion to evaluate models.

The default *criterion* depends on the value of the DISTRIBUTION= option. For distributions that involve dispersion parameters, GCV is the default. For distributions without dispersion parameters, UBRE is the default. For all three criteria, you can optionally use the FACTOR= option to specify an extra tuning parameter in order to penalize more for model roughness. The value of *number* must be greater than or equal to 1. For more information about the model evaluation criteria, see "Model Evaluation Criteria" on page 3463.

DISPERSION=*number*
PHI=*number*
specifies a fixed dispersion parameter for distributions that have a dispersion parameter. The dispersion parameter that is used in all computations is fixed at *number*; it is not estimated.

DISTRIBUTION=*keyword*
specifies the response distribution for the model. The *keywords* and their associated distributions are shown in Table 49.8.

Table 49.8 Built-In Distribution Functions

DISTRIBUTION=	Distribution Function
BINARY	Binary
BINOMIAL	Binary or binomial
GAMMA	Gamma
INVERSEGAUSSIAN | IG	Inverse Gaussian
NEGATIVEBINOMIAL | NB	Negative binomial
NORMAL | GAUSSIAN	Normal
POISSON	Poisson
TWEEDIE< (*Tweedie-options*) **>**	Tweedie

When DISTRIBUTION=TWEEDIE, you can specify the following *Tweedie-options*:

INITIALP=
specifies a starting value for iterative estimation of the Tweedie power parameter.

P=
requests a fixed Tweedie power parameter.

If you do not specify a link function in the LINK= option, a default link function is used. The default link function for each distribution is shown in Table 49.9. You can use any link function shown in Table 49.10 by specifying the LINK= option. Other commonly used link functions for each distribution are shown in Table 49.9.

Table 49.9 Default and Commonly Used Link Functions

DISTRIBUTION=	Default Link Function	Other Commonly Used Link Functions
BINARY	Logit	Probit, complementary log-log, log-log
BINOMIAL	Logit	Probit, complementary log-log, log-log
GAMMA	Reciprocal	Log
INVERSEGAUSSIAN | IG	Reciprocal square	Log
NEGATIVEBINOMIAL | NB	Log	
NORMAL | GAUSSIAN	Identity	Log
POISSON	Log	
TWEEDIE	Log	

FDHESSIAN
requests that the second-order derivatives (Hessian) be computed using finite-difference approximations based on evaluation of the first-order derivatives (gradient). This option might be useful if the analytical Hessian takes a long time to compute.

INITIALPHI=*number*

specifies a starting value for iterative maximum likelihood estimation of the dispersion parameter for distributions that have a dispersion parameter.

LINK=*keyword*

specifies the link function for the model. The *keywords* and the associated link functions are shown in Table 49.10. Default and commonly used link functions for the available distributions are shown in Table 49.9.

Table 49.10 Built-In Link Functions

LINK=	Link Function	$g(\mu) = \eta =$
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$
IDENTITY | ID	Identity	μ
INV | RECIP	Reciprocal	$1/\mu$
INV2	Reciprocal square	$1/\mu^2$
LOG	Logarithm	$\log(\mu)$
LOGIT	Logit	$\log(\mu/(1-\mu))$
LOGLOG	Log-log	$-\log(-\log(\mu))$
PROBIT	Probit	$\Phi^{-1}(\mu)$

$\Phi^{-1}(\cdot)$ denotes the quantile function of the standard normal distribution.

MAXPHI=*number*

specifies an upper bound for maximum likelihood estimation of the dispersion parameter for distributions that have a dispersion parameter.

METHOD=OUTER | PERFORMANCE

specifies the algorithm for selecting smoothing parameters for *spline-effects*. You can specify the following values:

OUTER specifies the outer iteration method for selecting smoothing parameters. For more information about the method, see the section "Outer Iteration" on page 3464.

PERFORMANCE specifies the performance iteration method for selecting smoothing parameters. For more information about the method, see the section "Performance Iteration" on page 3465.

By default, METHOD=PERFORMANCE.

MINPHI=*number*

specifies a lower bound for maximum likelihood estimation of the dispersion parameter for distributions that have a dispersion parameter.

NORMALIZE
requests normalized spline basis functions for model fitting. After the regression spline basis functions are computed, each column is standardized to have a unit standard error. The corresponding penalty matrix is also scaled accordingly. This option might be helpful when you have badly scaled data.

OFFSET=*variable*
specifies a *variable* to be used as an offset to the linear predictor. An offset plays the role of an effect whose coefficient is known to be 1. The offset variable cannot appear in the CLASS statement or elsewhere in the MODEL statement. Observations that have missing values for the offset variable are excluded from the analysis.

RIDGE=*number*
allows a ridge parameter such that a diagonal matrix $\mathbf{H} : \mathbf{H}_{ii} = \textit{number}$ is added to the optimization problem with respect to regression parameters:

$$\min(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \boldsymbol{\beta}'\mathbf{S}_{\boldsymbol{\lambda}}\boldsymbol{\beta} + \boldsymbol{\beta}'\mathbf{H}\boldsymbol{\beta} \quad \text{with respect to} \quad \boldsymbol{\beta}$$

By default, RIDGE=0. Specifying a small ridge parameter might be helpful if the model matrix $\mathbf{X}'\mathbf{X} + \mathbf{S}_{\boldsymbol{\lambda}}$ is close to singular.

SCALE=DEVIANCE | MLE | PEARSON
specifies the method for estimating the scale and dispersion parameters. You can specify the following values:

DEVIANCE
estimates the dispersion parameter by using the deviance statistic.

MLE
computes the dispersion parameter by maximizing the likelihood or penalized likelihood.

PEARSON
estimates the dispersion parameter by using Pearson's statistic.

By default, SCALE=MLE.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< *keyword* < **=***name* > >. . . < *keyword* < **=***name* > > < / *options* > **;**

The OUTPUT statement creates a data set that contains observationwise statistics that are computed after the model is fitted. The variables in the input data set are *not* included in the output data set in order to avoid data duplication for large data sets; however, variables that are specified in the ID statement are included.

The computation of the output statistics is based on the final parameter estimates. If the model fit does not converge, missing values are produced for the quantities that depend on the estimates.

You can specify the following syntax elements in the OUTPUT statement before the slash (/).

OUT=*SAS-data-set*

DATA=*SAS-data-set*

specifies the name of the output data set. If you omit this option, PROC GAMPL uses the DATA*n* convention to name the output data set.

keyword **<=***name***>**

specifies a statistic to include in the output data set and optionally assigns a *name* to the variable. If you do not provide a *name*, the GAMPL procedure assigns a default name based on the *keyword*.

You can specify the following *keywords* for adding statistics to the OUTPUT data set:

LINP | XBETA

requests the linear predictor $\eta = \mathbf{x}'\boldsymbol{\beta}$. For observations in which only the response variable is missing, values of the linear predictor are computed even though these observations do not affect the model fit. The default *name* is Xbeta.

LOWER

requests a lower Bayesian confidence band value for the predicted value. The default *name* is Lower.

PEARSON | PEARS | RESCHI

requests the Pearson residual, $(y-\mu)/\sqrt{\mathrm{V}(\mu)}$, where μ is the estimate of the predicted response mean and $\mathrm{V}(\mu)$ is the response distribution variance function. The default *name* is Pearson.

PREDICTED | PRED | P

requests predicted values for the response variable. For observations in which only the response variable is missing, the predicted values are computed even though these observations do not affect the model fit. The default *name* is Pred.

RESIDUAL | RESID | R

requests the raw residual, $y-\mu$, where μ is the estimate of the predicted mean. The default *name* is Residual.

STD

requests a standard error for the linear predictor. The default *name* is Std.

UPPER

requests an upper Bayesian confidence band value for the predicted value. The default *name* is Upper.

You can specify the following *options* in the OUTPUT statement after the slash (/):

ALPHA=*number*

specifies the significance level for the construction of Bayesian confidence bands in the OUTPUT data set. The confidence level is $1-$ *number*.

COMPONENT

requests componentwise statistics for all spline terms if LINP, LOWER, STD, or UPPER is specified as a *keyword*.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

You can use this statement to define performance parameters for multithreaded computing, and you can request details about performance results.

The PERFORMANCE statement is documented in the section "PERFORMANCE Statement" on page 569.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, then all observations that are used in the analysis are assigned a weight of 1.

Details: GAMPL Procedure

Missing Values

Any observation that has missing values for the response, frequency, weight, offset, or explanatory variables is excluded from the analysis; however, missing values are valid for response and explanatory variables if you specify the MISSING option in the CLASS statement. Observations that have a nonpositive weight or a frequency less than 1 are also excluded. For Poisson, negative binomial and Tweedie distributions, observations that have a negative response value are excluded. For gamma and inverse Gaussian distributions, observations that have a nonpositive response value are excluded.

The estimated linear predictor and the fitted means are not computed for any observation that has missing offset or explanatory variable values. However, if only the response value is missing, the linear predictor and the fitted means can be computed and output to a data set by using the OUTPUT statement.

Thin-Plate Regression Splines

The GAMPL procedure uses thin-plate regression splines (Wood 2003) to construct spline basis expansions. The thin-plate regression splines are based on thin-plate smoothing splines (Duchon 1976, 1977). Compared to thin-plate smoothing splines, thin-plate regression splines produce fewer basis expansions and thus make direct fitting of generalized additive models possible.

Thin-Plate Smoothing Splines

Consider the problem of estimating a smoothing function f of $\mathbf{x}$ with d covariates from n observations. The model assumes

$$y_i = f(\mathbf{x}_i) + \epsilon_i, i = 1, \ldots, n$$

Then the thin-plate smoothing splines estimate the smoothing function f by minimizing the penalized least squares function:

$$\sum_{i=1}^{n} (y_i - f(\mathbf{x}_i))^2 + \lambda J_{m,d}(f)$$

The penalty term $\lambda J_{m,d}(f)$ includes the function that measures roughness on the f estimate:

$$J_{m,d}(f) = \int \cdots \int \sum_{\alpha_1 + \cdots + \alpha_d = m} \frac{m!}{\alpha_1! \cdots \alpha_d!} \left(\frac{\partial^m f}{\partial {x_1}^{\alpha_1} \cdots \partial {x_d}^{\alpha_d}} \right)^2 \mathrm{d}x_1 \cdots \mathrm{d}x_d$$

The parameter m (which corresponds to the M= option for a spline effect) specifies how the penalty is applied to the function roughness. Function derivatives whose order is less than m are not penalized. The relation $2m > d$ must be satisfied.

The penalty term also includes the smoothing parameter $\lambda \in [0, \infty)$, which controls the trade-off between the model's fidelity to the data and the function smoothness of f. When $\lambda = 0$, the function estimate corresponds to an interpolation. When $\lambda \to \infty$, the function estimate becomes the least squares fit. By using the defined penalized least squares criterion and a fixed λ value, you can explicitly express the estimate of the smooth function f in the following form:

$$f_\lambda(\mathbf{x}) = \sum_{j=1}^{M} \theta_j \phi_j(\mathbf{x}) + \sum_{i}^{n} \delta_i \eta_{m,d}(\|\mathbf{x} - \mathbf{x}_i\|)$$

In the expression of $f_\lambda(\mathbf{x})$, δ_i and θ_j are coefficients to be estimated. The functions $\phi_j(\mathbf{x})$ correspond to unpenalized polynomials of $\mathbf{x}$ with degrees up to $m - 1$. The total number of these polynomials is $M = \binom{m+d-1}{d}$. The function $\eta_{m,d}$ models the extra nonlinearity besides the polynomials and is a function of the Euclidean distance r between any $\mathbf{x}$ value and an observed $\mathbf{x}_i$ value:

$$\eta_{m,d}(r) = \begin{cases} \frac{(-1)^{m+1+d/2}}{2^{2m-1}\pi^{d/2}(m-1)!(m-d/2)!} r^{2m-d} \log(r) & \text{if } d \text{ is even} \\ \frac{\Gamma(d/2-m)}{2^{2m}\pi^{d/2}(m-1)!} r^{2m-d} & \text{if } d \text{ is odd} \end{cases}$$

Define the penalty matrix $\mathbf{E}$ such that each entry $E_{ij} = \eta_{m,d}(\|\mathbf{x}_i - \mathbf{x}_j\|)$, let $\mathbf{y}$ be the vector of the response, let $\mathbf{T}$ be the matrix where each row is formed by $\phi_j(\mathbf{x})$, and let $\boldsymbol{\theta}$ and $\boldsymbol{\delta}$ be vectors of coefficients θ_j and δ_i. Then you can obtain the function estimate f_λ from the following minimization problem:

$$\min \|\mathbf{y} - \mathbf{T}\boldsymbol{\theta} - \mathbf{E}\boldsymbol{\delta}\|^2 + \lambda \boldsymbol{\delta}'\mathbf{E}\boldsymbol{\delta} \quad \text{subject to} \quad \mathbf{T}'\boldsymbol{\delta} = \mathbf{0}$$

For more information about thin-plate smoothing splines, see Chapter 125, "The TPSPLINE Procedure."

Low-Rank Approximation

Given the representation of the thin-plate smoothing spline, the estimate of f involves as many parameters as the number of unique data points. Solving $(\boldsymbol{\theta}, \boldsymbol{\delta})$ with an optimum λ becomes difficult for large problems.

Because the matrix $\mathbf{E}$ is symmetric and nonnegative definite, the eigendecomposition can be taken as $\mathbf{E} = \mathbf{UDU}'$, where $\mathbf{D}$ is the diagonal matrix of eigenvalues d_i of $\mathbf{E}$, and $\mathbf{U}$ is the matrix of eigenvectors that corresponds to d_i. The truncated eigendecomposition forms $\tilde{\mathbf{E}}_k$, which is an approximation to $\mathbf{E}$ such that

$$\tilde{\mathbf{E}}_k = \mathbf{U}_k \mathbf{D}_k \mathbf{U}_k'$$

where $\mathbf{D}_k$ is a diagonal matrix that contains the k most extreme eigenvalues in descending order of absolute values: $|\tilde{d}_1| > \cdots > |\tilde{d}_k|$. $\mathbf{U}_k$ is the matrix that is formed by columns of eigenvectors that correspond to the eigenvalues in $\mathbf{D}_k$.

The approximation $\tilde{\mathbf{E}}_k$ not only reduces the dimension from $n \times n$ of $\mathbf{E}$ to $n \times k$ but also is optimal in two senses. First, $\tilde{\mathbf{E}}_k$ minimizes the spectral norm $\|\mathbf{E} - \mathbf{F}_k\|_2$ between $\mathbf{E}$ and all rank k matrices $\mathbf{F}_k$. Second, $\tilde{\mathbf{E}}_k$ also minimizes the worst possible change that is introduced by the eigenspace truncation as defined by

$$\max_{\boldsymbol{\delta} \neq \mathbf{0}} \frac{\boldsymbol{\delta}'(\mathbf{E} - \mathbf{G}_k)\boldsymbol{\delta}}{\|\boldsymbol{\delta}\|^2}$$

where $\mathbf{G}_k$ is formed by any k eigenvalues and corresponding eigenvectors. For more information, see Wood (2003).

Now given $\mathbf{E} \approx \tilde{\mathbf{E}}_k$ and $\tilde{\mathbf{E}}_k = \mathbf{U}_k \mathbf{D}_k \mathbf{U}_k'$, and letting $\boldsymbol{\delta}_k = \mathbf{U}_k' \boldsymbol{\delta}$, the minimization problem becomes

$$\min \|\mathbf{y} - \mathbf{T}\boldsymbol{\theta} - \mathbf{U}_k \mathbf{D}_k \boldsymbol{\delta}_k\|^2 + \lambda \boldsymbol{\delta}_k' \mathbf{D}_k \boldsymbol{\delta}_k \quad \text{subject to} \quad \mathbf{T}'\mathbf{U}_k \boldsymbol{\delta}_k = \mathbf{0}$$

You can turn the constrained optimization problem into an unconstrained one by using any orthogonal column basis $\mathbf{Z}$. One way to form $\mathbf{Z}$ is via the QR decomposition of $\mathbf{U}_k'\mathbf{T}$:

$$\mathbf{U}_k'\mathbf{T} = [\mathbf{Q}_1 \mathbf{Q}_2] \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}$$

Let $\mathbf{Z} = \mathbf{Q}_2$. Then it is verified that

$$\mathbf{T}'\mathbf{U}_k \mathbf{Z} = \mathbf{R}'\mathbf{Q}_1'\mathbf{Q}_2 = \mathbf{0}$$

So for $\boldsymbol{\delta}_k$ such that $\mathbf{T}'\mathbf{U}_k \boldsymbol{\delta}_k = \mathbf{0}$, it is true that $\boldsymbol{\delta}_k = \mathbf{Z}\tilde{\boldsymbol{\delta}}$. Now the problem becomes the unconstrained optimization,

$$\min \|\mathbf{y} - \mathbf{T}\boldsymbol{\theta} - \mathbf{U}_k \mathbf{D}_k \mathbf{Z}\tilde{\boldsymbol{\delta}}\|^2 + \lambda \tilde{\boldsymbol{\delta}}' \mathbf{Z}' \mathbf{D}_k \mathbf{Z} \tilde{\boldsymbol{\delta}}$$

Let

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\theta} \\ \tilde{\boldsymbol{\delta}} \end{bmatrix}, \quad \mathbf{X} = [\mathbf{T} : \mathbf{U}_k \mathbf{D}_k \mathbf{Z}], \quad \text{and} \quad \mathbf{S} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Z}'\mathbf{D}_k\mathbf{Z} \end{bmatrix}$$

The optimization is simplified as

$$\min \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \boldsymbol{\beta}' \mathbf{S} \boldsymbol{\beta} \quad \text{with respect to} \quad \boldsymbol{\beta}$$

Generalized Additive Models

Generalized Linear Models

All probability distributions that the GAMPL procedure fits are members of an exponential family of distributions. For the specification of an exponential family, see the section "Generalized Linear Models Theory" in Chapter 51, "The GENMOD Procedure."

Table 49.11 lists and defines some common notation that is used in the context of generalized linear models and generalized additive models.

Table 49.11 Common Notation

Notation	Meaning
ℓ	Log-likelihood
ℓ_p	Penalized log-likelihood
D	Deviance
D_p	Penalized deviance
g	Link function
g^{-1}	Inverse link function
$\boldsymbol{\mu}$	Response mean $\boldsymbol{\mu} = g^{-1}(\boldsymbol{\eta})$
$\boldsymbol{\eta}$	Linear predictor $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$
ϕ	Dispersion parameter
$\mathbf{z}$	Column of adjusted response variable
$\boldsymbol{\nu}$	Column of response variance
ω_i	Prior weight for each observation
w_i	Adjusted weight for each observation
$\mathbf{W}$	Diagonal matrix of adjusted weights

The GAMPL procedure supports the following distributions: binary, binomial, gamma, inverse Gaussian, negative binomial, normal (Gaussian), Poisson and Tweedie.

For forms of log-likelihood functions for each of the probability distributions, see the section "Log-Likelihood Functions" in Chapter 51, "The GENMOD Procedure." For forms of the deviance for each of the probability distributions, see the section "Goodness of Fit" in Chapter 51, "The GENMOD Procedure."

Generalized Additive Models

Generalized additive models are extensions of generalized linear models (Nelder and Wedderburn 1972). For each observation that has a response Y_i and a row vector of the model matrix $\mathbf{x}_i$, both generalized additive models and generalized linear models assume the model additivity

$$g(\mu_i) = f_1(\mathbf{x}_{i1}) + \cdots + f_p(\mathbf{x}_{ip})$$

where $\mu_i = E(Y_i)$ and Y_i is independently distributed in some exponential family. Generalized linear models further assume model linearity by $f_j(\mathbf{x}_{ij}) = \mathbf{x}_{ij}\beta_j$ for $j = 1, \ldots, p$. Generalized additive models relax the linearity assumption by allowing some smoothing functions f_j to characterize the dependency. The GAMPL procedure constructs the smoothing functions by using thin-plate regression splines. For more

information about generalized additive models and other type of smoothing functions, see Chapter 48, "The GAM Procedure."

Consider a generalized additive model that has some linear terms $\mathbf{X}_L$ with coefficients $\boldsymbol{\beta}_L$ and p smoothing functions f_j. Each smoothing function can be constructed by thin-plate regression splines with a smoothing parameter λ_j. Using the notations described in the section "Low-Rank Approximation" on page 3460, you can characterize each smoothing function by

$$\boldsymbol{\beta}_j = \begin{bmatrix} \boldsymbol{\theta}_j \\ \tilde{\boldsymbol{\delta}}_j \end{bmatrix}, \quad \mathbf{X}_j = [\mathbf{T}_j : \mathbf{U}_{kj}\mathbf{D}_{kj}\mathbf{Z}_j], \quad \text{and} \quad \mathbf{S}_j = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Z}_j'\mathbf{D}_{kj}\mathbf{Z}_j \end{bmatrix}$$

Notice that each smoothing function representation contains a zero-degree polynomial that corresponds to a constant. Having multiple constant terms makes the smoothing functions unidentifiable. The solution is to include a global constant term (that is, the intercept) in the model and enforce the centering constraint to each smoothing function. You can write the constraint as

$$\mathbf{1}'\mathbf{X}_j\boldsymbol{\beta}_j = 0$$

By using a similar approach as the linear constraint for thin-plate regression splines, you obtain the orthogonal column basis $\mathbf{V}_j$ via the QR decomposition of $\mathbf{X}_j'\mathbf{1}$ such that $\mathbf{1}'\mathbf{X}_j\mathbf{V}_j = 0$. Each smoothing function can be reparameterized as $\tilde{\mathbf{X}}_j = \mathbf{X}_j\mathbf{V}_j$.

Let $\mathbf{X} = [\mathbf{X}_L : \tilde{\mathbf{X}}_1 : \cdots : \tilde{\mathbf{X}}_p]$ and $\boldsymbol{\beta}' = [\boldsymbol{\beta}_L' : \boldsymbol{\beta}_1' : \cdots : \boldsymbol{\beta}_p']$. Then the generalized additive model can be represented as $g(\boldsymbol{\mu}) = \mathbf{X}\boldsymbol{\beta}$. The roughness penalty matrix is represented as a block diagonal matrix:

$$\mathbf{S}_{\boldsymbol{\lambda}} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \cdots & & \mathbf{0} \\ \mathbf{0} & \lambda_1\mathbf{S}_1 & \cdots & & \mathbf{0} \\ & & \ddots & & \\ \vdots & \vdots & \lambda_j\mathbf{S}_j & & \vdots \\ & & & \ddots & \\ \mathbf{0} & \mathbf{0} & \cdots & & \lambda_p\mathbf{S}_p \end{bmatrix}$$

Then the roughness penalty is measured in the quadratic form $\boldsymbol{\beta}'\mathbf{S}_{\boldsymbol{\lambda}}\boldsymbol{\beta}$.

Penalized Likelihood Estimation

Given a fixed vector of smoothing parameters, $\boldsymbol{\lambda} = [\lambda_i \ldots \lambda_p]'$, you can fit the generalized additive models by the penalized likelihood estimation. In contrast to the maximum likelihood estimation, penalized likelihood estimation obtains an estimate for $\boldsymbol{\beta}$ by maximizing the penalized log likelihood,

$$\ell_p(\boldsymbol{\beta}) = \ell(\boldsymbol{\beta}) - \frac{1}{2}\boldsymbol{\beta}'\mathbf{S}_{\boldsymbol{\lambda}}\boldsymbol{\beta}$$

Any optimization technique that you can use for maximum likelihood estimation can also be used for penalized likelihood estimation. If first-order derivatives are required for the optimization technique, you can compute the gradient as

$$\frac{\partial \ell_p}{\partial \boldsymbol{\beta}} = \frac{\partial \ell}{\partial \boldsymbol{\beta}} - \mathbf{S}_{\boldsymbol{\lambda}}\boldsymbol{\beta}$$

If second-order derivatives are required for the optimization technique, you can compute the Hessian as

$$\frac{\partial^2 \ell_p}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}'} = \frac{\partial^2 \ell}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}'} - \mathbf{S}_{\boldsymbol{\lambda}}$$

In the gradient and Hessian forms, $\partial \ell / \partial \boldsymbol{\beta}$ and $\partial^2 \ell / (\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}')$ are the corresponding gradient and Hessian, respectively, for the log-likelihood for generalized linear models. For more information about optimization techniques, see the section "Choosing an Optimization Technique" on page 3469.

Model Evaluation Criteria

Given a fixed set of smoothing parameters $\boldsymbol{\lambda}$ in which each λ_i controls the smoothness of each spline term, you can fit a generalized additive model by the penalized likelihood estimation. There are infinitely many sets of smoothing parameters. In order to search optimum models, some model evaluation criteria need to be defined to quantify the model goodness-of-fit. The GAMPL procedure uses the following model evaluation criteria:

- generalized cross validation (GCV), $\mathcal{V}_g$ (Craven and Wahba 1979)
- unbiased risk estimator (UBRE), $\mathcal{V}_u$ (Craven and Wahba 1979)
- generalized approximate cross validation (GACV), $\mathcal{V}_a$ (Xiang and Wahba 1996)

Consider the optimization problem

$$\min(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + \boldsymbol{\beta}'\mathbf{S}_{\boldsymbol{\lambda}}\boldsymbol{\beta} \quad \text{with respect to} \quad \boldsymbol{\beta}$$

The parameter estimate for $\boldsymbol{\beta}$ can be represented as

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X} + \mathbf{S}_{\boldsymbol{\lambda}})^{-1}\mathbf{X}'\mathbf{y}$$

And the smoothing matrix (also called the influence matrix or hat matrix) is thus represented as

$$\mathbf{H}_{\boldsymbol{\lambda}} = \mathbf{X}(\mathbf{X}'\mathbf{X} + \mathbf{S}_{\boldsymbol{\lambda}})^{-1}\mathbf{X}'$$

With the defined smoothing matrix, you can form the model evaluation criteria as follows:

$$\begin{aligned}
\mathcal{V}_g(\boldsymbol{\lambda}) &= \frac{n\|\mathbf{y} - \mathbf{H}_{\boldsymbol{\lambda}}\mathbf{y}\|^2}{(\mathrm{tr}(\mathbf{I} - \gamma\mathbf{H}_{\boldsymbol{\lambda}}))^2} \\
\mathcal{V}_u(\boldsymbol{\lambda}) &= \frac{1}{n}\|\mathbf{y} - \mathbf{H}_{\boldsymbol{\lambda}}\mathbf{y}\|^2 - \frac{2}{n}\sigma^2\mathrm{tr}(\mathbf{I} - \gamma\mathbf{H}_{\boldsymbol{\lambda}}) + \sigma^2 \\
\mathcal{V}_a(\boldsymbol{\lambda}) &= \frac{1}{n}\|\mathbf{y} - \mathbf{H}_{\boldsymbol{\lambda}}\mathbf{y}\|^2 \left(1 + 2\gamma\frac{\mathrm{tr}(\mathbf{H}_{\boldsymbol{\lambda}})}{\mathrm{tr}(\mathbf{I} - \mathbf{H}_{\boldsymbol{\lambda}})}\right)
\end{aligned}$$

In the equations, $\gamma \geq 1$ (which corresponds to the GAMMA= suboption of the CRITERION= option) is the tuning parameter that is sometimes used to enforce smoother models.

The GAMPL procedure uses fitting algorithms that involve minimizing the model evaluation criterion with respect to unknown smoothing parameters $\boldsymbol{\lambda}$.

Fitting Algorithms

For models that assume a normally distributed response variable, you can minimize the model evaluation criteria directly by searching for optimal smoothing parameters. For models that have nonnormal distributions, you cannot use the model evaluation criteria directly because the involved statistics keep changing between iterations. The GAMPL procedure enables you to use either of two fitting approaches to search for optimum models: the outer iteration method and the performance iteration method. The outer iteration method modifies the model evaluation criteria so that a global objective function can be minimized in order to find the best smoothing parameters. The performance iteration method minimizes a series of objective functions in an iterative fashion and then obtains the optimum smoothing parameters at convergence. For large data sets, the performance iteration method usually converges faster than the outer iteration method.

Outer Iteration

The outer iteration method is outlined in Wood (2006). The method uses modified model evaluation criteria, which are defined as follows:

$$
\begin{aligned}
\mathcal{V}_g^o(\boldsymbol{\lambda}) &= \frac{nD_{\boldsymbol{\lambda}}(\boldsymbol{\mu})}{(n-\gamma\mathrm{tr}(\mathbf{H}_{\boldsymbol{\lambda}}))^2} \\
\mathcal{V}_u^o(\boldsymbol{\lambda}) &= \frac{D_{\boldsymbol{\lambda}}(\boldsymbol{\mu})}{n} - \frac{2}{n}\sigma^2\mathrm{tr}(\mathbf{I}-\gamma\mathbf{H}_{\boldsymbol{\lambda}}) + \sigma^2 \\
\mathcal{V}_a^o(\boldsymbol{\lambda}) &= \frac{D_{\boldsymbol{\lambda}}(\boldsymbol{\mu})}{n} + \frac{2\gamma\mathrm{tr}(\mathbf{H}_{\boldsymbol{\lambda}})P_{\boldsymbol{\lambda}}}{n\mathrm{tr}(\mathbf{I}-\mathbf{H}_{\boldsymbol{\lambda}})}
\end{aligned}
$$

By replacing $\|\mathbf{y}-\mathbf{H}_{\boldsymbol{\lambda}}\mathbf{y}\|^2$ with model deviance $D_{\boldsymbol{\lambda}}(\boldsymbol{\mu})$, the modified model evaluation criteria relate to the smoothing parameter $\boldsymbol{\lambda}$ in a direct way such that the analytic gradient and Hessian are available in explicit forms. The Pearson's statistic $P_{\boldsymbol{\lambda}}$ is used in the GACV criterion $\mathcal{V}_a^o(\boldsymbol{\lambda})$ (Wood 2008). The algorithm for the outer iteration is thus as follows:

1. Initialize smoothing parameters by taking one step of performance iteration based on adjusted response and adjusted weight except for spline terms with initial values that are specified in the INITSMOOTH= option.

2. Search for the best smoothing parameters by minimizing the modified model evaluation criteria. The optimization process stops when any of the convergence criteria that are specified in the SMOOTHOPTIONS option is met. At each optimization step:

   a) Initialize by setting initial regression parameters $\boldsymbol{\beta} = \{g(\hat{\mathbf{y}}), 0, \ldots, 0\}'$. Set the initial dispersion parameter if necessary.

   b) Search for the best regression parameters $\boldsymbol{\beta}$ by minimizing the penalized deviance D_p (or maximizing the penalized likelihood ℓ_p for negative binomial distribution). The optimization process stops when any of the convergence criteria that are specified in the PLIKEOPTIONS option is met.

   c) At convergence, evaluate derivatives of the model evaluation criteria with respect to $\boldsymbol{\lambda}$ by using $\partial D_p/\partial\boldsymbol{\beta}$, $\partial^2 D_p/(\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}')$, $\partial\boldsymbol{\beta}/\partial\lambda_j$, and $\partial^2\boldsymbol{\beta}/(\partial\lambda_j\partial\lambda_k)$.

Step 2b usually converges quickly because it is essentially penalized likelihood estimation given that $D_p = 2\phi(\ell_{\max} - \ell) + \boldsymbol{\beta}\mathbf{S}_{\boldsymbol{\lambda}}\boldsymbol{\beta}'$. Step 2c contains involved computation by using the chain rule of derivatives. For more information about computing derivatives of $\mathcal{V}_g^o$ and $\mathcal{V}_u^o$, see Wood (2008, 2011).

Performance Iteration

The performance iteration method is proposed by Gu and Wahba (1991). Wood (2004) modifies and stabilizes the algorithm for fitting generalized additive models. The algorithm for the performance iteration method is as follows:

1. Initialize smoothing parameters $\boldsymbol{\lambda} = \{1, \ldots, 1\}$, except for spline terms whose initial values are specified in the INITSMOOTH= option. Set the initial regression parameters $\boldsymbol{\beta} = \{g(\bar{\mathrm{y}}), 0, \ldots, 0\}'$. Set the initial dispersion parameter if necessary.

2. Repeat the following steps until any of these three conditions is met: (1) the absolute function change in penalized likelihood is sufficiently small; (2) the absolute relative function change in penalized likelihood is sufficiently small; (3) the number of iterations exceeds the maximum iteration limit.

   a) Form the adjusted response and adjusted weight from $\boldsymbol{\mu} = g^{-1}(\boldsymbol{\eta})$. For each observation,

   $$z_i = \eta_i + (y_i - \mu_i)/\mu_i', \quad w_i = \omega_i \mu_i'^2/v_i$$

   b) Search for the best smoothing parameters for the current iteration based on valid adjusted response values and adjusted weight values.

   c) Use the smoothing parameters to construct the linear predictor and the predicted means.

   d) Obtain an estimate for the dispersion parameter if necessary.

In step 2b, you can use different optimization techniques to search for the best smoothing parameters. The Newton-Raphson optimization is efficient in finding the optimum $\boldsymbol{\lambda}$ where the first- and second-order derivatives are available. For more information about computing derivatives of $\mathcal{V}_g$ and $\mathcal{V}_u$ with respect to $\boldsymbol{\lambda}$, see Wood (2004).

Degrees of Freedom

Let $\mathbf{W}$ be the adjusted weight matrix at convergence, and let $\mathbf{S}_{\boldsymbol{\lambda}}$ be the roughness penalty matrix with selected smoothing parameters. The degrees of freedom matrix is defined as in Wood (2006):

$$\mathbf{F} = (\mathbf{X}'\mathbf{W}\mathbf{X} + \mathbf{S}_{\boldsymbol{\lambda}})^{-1}\mathbf{X}'\mathbf{W}\mathbf{X}$$

Given the adjusted response $\mathbf{z}$, the parameter estimate is shown to be $\tilde{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}\mathbf{z}$ for the model without penalization, and the parameter estimate is $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X} + \mathbf{S}_{\boldsymbol{\lambda}})^{-1}\mathbf{X}'\mathbf{W}\mathbf{z} = \mathbf{F}\tilde{\boldsymbol{\beta}}$ with penalization. $\mathbf{F}$ is thus the matrix that projects or maps the unpenalized parameter estimates to the penalized ones.

The model degrees of freedom is given as

$$\mathrm{df} = \mathrm{tr}(\mathbf{F})$$

And the degrees of freedom for error is given as

$$\mathrm{df}_r = n - 2\mathrm{df} + \mathrm{tr}(\mathbf{FF})$$

For the *j*th spline term, the degrees of freedom for the component is defined to be the trace of the submatrix of $\mathbf{F}$ that corresponds to parameter estimates $\boldsymbol{\beta}_j$:

$$\mathrm{df}_j = \mathrm{tr}(\mathbf{F}_j)$$

The degrees of freedom for the smoothing component test of the *j*th term is defined similarly as

$$\mathrm{df}_j^t = 2\mathrm{df}_j - \mathrm{tr}((\mathbf{FF})_j)$$

Model Inference

Wahba (1983) proposes a Bayesian covariance matrix for parameter estimates $\boldsymbol{\beta}$ by interpreting a smoothing spline as a posterior mean. Nychka (1988) shows that the derived Bayesian posterior confidence limits work well from frequentist viewpoints. The Bayesian posterior covariance matrix for the parameters is

$$\mathbf{V}_{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X} + \mathbf{S}_{\boldsymbol{\lambda}})^{-1}\sigma^2$$

The posterior distribution for $\boldsymbol{\beta}$ is thus

$$\boldsymbol{\beta}|\mathbf{y} \sim N(\hat{\boldsymbol{\beta}}, \mathbf{V}_{\boldsymbol{\beta}})$$

For a particular point whose design row is vector $\mathbf{x}$, the prediction is $\mathbf{x}\hat{\boldsymbol{\beta}}$ and the standard error is $\sqrt{\mathbf{x}\mathbf{V}_{\boldsymbol{\beta}}\mathbf{x}'}$. The Bayesian posterior confidence limits are thus

$$\left(\mathbf{x}\hat{\boldsymbol{\beta}} \pm z_{\alpha/2}\sqrt{\mathbf{x}\mathbf{V}_{\boldsymbol{\beta}}\mathbf{x}'}\right)$$

where $z_{\alpha/2}$ is the $1 - \alpha/2$ quantile of the standard normal distribution.

For the *j*th spline term, the prediction for the component is formed by $\mathbf{x}_j\hat{\boldsymbol{\beta}}$, where $\mathbf{x}_j$ is a row vector of zeros except for columns that correspond to basis expansions of the *j*th spline term. And the standard error for the component is $\sqrt{\mathbf{x}_j\mathbf{V}_{\boldsymbol{\beta}}\mathbf{x}_j'}$.

Dispersion Parameter

Some distribution families (Gaussian, gamma, inverse Gaussian, negative binomial, and Tweedie) have a dispersion parameter that you can specify in the DISPERSION= option in the MODEL statement or you can estimate from the data. The following three suboptions for the SCALE= option in the MODEL statement correspond to three ways to estimate the dispersion parameter:

DEVIANCE
estimates the dispersion parameter by the deviance, given the regression parameter estimates:

$$\hat{\phi} = \frac{\sum_i D_i(y_i, \mu_i)}{n - \mathrm{df}}$$

MLE
estimates the dispersion parameter by maximizing the penalized likelihood, given the regression parameter estimates:

$$\hat{\phi} = \underset{\phi}{\operatorname{argmax}}\ \ell_p(\hat{\boldsymbol{\beta}}, \phi)$$

The MLE option is the only option that you can use to estimate the dispersion parameter for the negative binomial and Tweedie distributions.

PEARSON
estimates the dispersion parameter by Pearson's statistic, given the regression parameter estimates:

$$\hat{\phi} = \frac{\sum_i \omega_i (y_i - \mu_i)^2 / \nu_i}{n - \mathrm{df}}$$

If the dispersion parameter is estimated, it contributes one additional degree of freedom to the fitted model.

Tests for Smoothing Components

The GAMPL procedure performs a smoothing component test on the null hypotheses $f_j = 0$ for the jth component. In contrast to the analysis of deviance that is used in PROC GAM (which tests existence of nonlinearity for each smoothing component), the smoothing component test used in PROC GAMPL tests for the existence of a contribution for each smoothing component.

The hypothesis test is based on the Wald statistic. Define $\mathbf{X}_j$ as the matrix of all zeros except for columns that correspond to basis expansions of the jth spline term. Then the column vector of predictions is $\hat{\mathbf{f}}_j = \mathbf{X}_j \hat{\boldsymbol{\beta}}$, and the covariance matrix for the predictions is $\mathbf{V}_j = \mathbf{X}_j \mathbf{V}_{\boldsymbol{\beta}} \mathbf{X}_j'$. The Wald statistic for testing is

$$T_r = \hat{\mathbf{f}}_j' \mathbf{V}_j^{r-} \hat{\mathbf{f}}_j = \hat{\boldsymbol{\beta}}' \mathbf{X}_j' (\mathbf{X}_j \mathbf{V}_{\boldsymbol{\beta}} \mathbf{X}_j')^{r-} \mathbf{X}_j \hat{\boldsymbol{\beta}}$$

where $\mathbf{V}_j^{r-}$ is the rank-r pseudo-inverse of $\mathbf{V}_j$. If $\mathbf{R}_j$ is the Cholesky root for $\mathbf{X}_j' \mathbf{X}_j$ such that $\mathbf{R}_j' \mathbf{R} = \mathbf{X}_j' \mathbf{X}_j$, then the test statistic can be written as

$$T_r = \hat{\boldsymbol{\beta}}' \mathbf{R}_j' (\mathbf{R}_j \mathbf{V}_{\boldsymbol{\beta}} \mathbf{R}_j')^{r-} \mathbf{R}_j \hat{\boldsymbol{\beta}}$$

Wood (2012) proposes using the df_j^t degrees of freedom for test (which is defined in the section "Degrees of Freedom" on page 3465) as the rank r. Because spline terms in fitted models often have noninteger degrees of freedom, the GAMPL procedure uses a rounded value of df_j^t as the rank:

$$r = \begin{cases} \lfloor \mathrm{df}_j^t \rfloor & \text{if } \mathrm{df}_j^t - \lfloor \mathrm{df}_j^t \rfloor <= 0.05 \text{ or } \mathrm{df}_j^t < 1 \\ \lceil \mathrm{df}_j^t \rceil & \text{otherwise} \end{cases}$$

Let K be a symmetric and nonnegative definite matrix, and let its eigenvalues be sorted as $d_1 > d_2 > \cdots$; then the rank-r pseudo-inverse of K is formed by

$$K^{r-} = U_k \begin{bmatrix} d_1^{-1} & & \\ & \ddots & \\ & & d_r^{-1} \end{bmatrix} U_k'$$

where U_k are formed by columns of eigenvectors that correspond to the r eigenvalues.

Under the null hypothesis, the Wald statistic T_r approximately follows the chi-square distribution $T_r \sim \chi_r^2$. For an observed test statistic t_r, the p-value for rejecting the null hypothesis is computed as $P(\chi_r^2 > t_r)$ if the dispersion parameter is constant, or $P(F_{r,\mathrm{df}_r} > t_r)$ with df_r error degrees of freedom if the dispersion parameter is estimated.

Be cautious when you interpret the results of the smoothing component test because p-values are computed by approximation and the test does not take the smoothing parameter selection process into account.

Computational Method: Multithreading

Threading is the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). Each task is associated with a thread. Multithreading is the concurrent execution of threads. When multithreading is possible, you can realize substantial performance gains compared to the performance you get from sequential (single-threaded) execution.

The number of threads that the GAMPL procedure spawns is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the number of CPUs in the CPUCOUNT= SAS system option. For example, if you specify the following statement, the GAMPL procedure determines threading as if it were executing on a system that had four CPUs, regardless of the actual CPU count:

```
options cpucount=4;
```

- You can specify the NTHREADS= option in the PERFORMANCE statement to control the number of threads. This specification overrides the CPUCOUNT= system option. Specify NTHREADS=1 to force single-threaded execution.

The GAMPL procedure allocates one thread per CPU by default.

The tasks that are multithreaded by the GAMPL procedure are primarily defined by dividing the data that are processed on a single machine among the threads—that is, the GAMPL procedure implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and PROC GAMPL is running with four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following:

- variable levelization
- effect levelization
- formation of the initial crossproducts matrix
- truncated eigendecomposition
- formation of spline basis expansions
- objective function calculation
- gradient calculation
- Hessian calculation
- scoring of observations
- computing predictions for smoothing component plots

In addition, operations on matrices such as sweeps can be multithreaded if the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Choosing an Optimization Technique

First- or Second-Order Techniques

The factors that affect how you choose an optimization technique for a particular problem are complex. Although the default technique works well for most problems, you might occasionally benefit from trying several different techniques.

For many optimization problems, computing the gradient takes more computer time than computing the function value. Computing the Hessian sometimes takes *much* more computer time and memory than computing the gradient, especially when there are many decision variables. Unfortunately, optimization techniques that do not use some kind of Hessian approximation usually require many more iterations than techniques that use a Hessian matrix; as a result, the total run time of these techniques is often longer. Techniques that do not use the Hessian also tend to be less reliable. For example, they can terminate more easily at stationary points than at global optima.

Table 49.12 shows which derivatives are required for each optimization technique.

Table 49.12 Derivatives Required

Technique	First-Order	Second-Order
TRUREG	x	x
NEWRAP	x	x
NRRIDG	x	x
QUANEW	x	-
DBLDOG	x	-
CONGRA	x	-
NMSIMP	-	-

The second-order derivative techniques (TRUREG, NEWRAP, and NRRIDG) are best for small problems for which the Hessian matrix is not expensive to compute. Sometimes the NRRIDG technique can be faster than the TRUREG technique, but TRUREG can be more stable. The NRRIDG technique requires only one matrix with $p(p+1)/2$ double words, where p denotes the number of parameters in the optimization. TRUREG and NEWRAP require two such matrices.

The QUANEW and DBLDOG first-order derivative techniques are best for medium-sized problems for which the objective function and the gradient can be evaluated much faster than the Hessian. In general, the QUANEW and DBLDOG techniques require more iterations than TRUREG, NRRIDG, and NEWRAP, but each iteration can be much faster. The QUANEW and DBLDOG techniques require only the gradient to update an approximate Hessian, and they require slightly less memory than TRUREG or NEWRAP.

The CONGRA first-order derivative technique is best for large problems for which the objective function and the gradient can be computed much faster than the Hessian and for which too much memory is required to store the (approximate) Hessian. In general, the CONGRA technique requires more iterations than QUANEW or DBLDOG, but each iteration can be much faster. Because CONGRA requires only a factor of p double-word memory, many large applications can be solved only by CONGRA.

The no-derivative technique NMSIMP is best for small problems for which derivatives are not continuous or are very difficult to compute.

Each optimization technique uses one or more convergence criteria that determine when it has converged. A technique is considered to have converged when any one of the convergence criteria is satisfied. For example, under the default settings, the QUANEW technique converges if the value of the ABSGCONV= option is less than 1E–5, the value of the FCONV= option is less than $2 \times \epsilon$, or the value of the GCONV= option is less than 1E–8.

By default, the GAMPL procedure applies the NEWRAP technique to optimization for selecting smoothing parameters by using the performance iteration method. For the outer iteration method, the GAMPL procedure applies the QUANEW technique for selecting smoothing parameters.

Technique Descriptions

The following subsections provide details about each optimization technique and follow the same order as Table 49.12.

Trust Region Optimization (TRUREG)

The trust region technique (TRUREG) uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function $f(\boldsymbol{\psi})$ have continuous first- and second-order derivatives inside the feasible region.

The trust region technique iteratively optimizes a quadratic approximation to the nonlinear objective function within a hyperelliptic trust region that has radius Δ and constrains the step size that corresponds to the quality of the quadratic approximation. The trust region technique that PROC GAMPL uses is based on Dennis, Gay, and Welsch (1981); Gay (1983); Moré and Sorensen (1983).

The trust region technique performs well for small- to medium-sized problems, and it does not need many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, either the QUANEW technique or the CONGRA technique might be more efficient.

Newton-Raphson Optimization with Line Search (NEWRAP)

The Newton-Raphson optimization with line search (NEWRAP) technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

If second-order derivatives are computed efficiently and precisely, the NEWRAP technique can perform well for medium-sized to large problems, and it does not need many function, gradient, and Hessian calls.

This technique uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. Otherwise, a combination of ridging and line search is performed to compute successful steps. If the Hessian is not positive-definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive-definite (Eskow and Schnabel 1991).

In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The line-search technique uses quadratic interpolation and cubic extrapolation.

Newton-Raphson Ridge Optimization (NRRIDG)

The Newton-Raphson ridge optimization (NRRIDG) technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

This technique uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. If at least one of these two conditions is not satisfied, a multiple of the identity matrix is added to the Hessian matrix.

Because the NRRIDG technique uses an orthogonal decomposition of the approximate Hessian, each iteration of NRRIDG can be slower than an iteration of the NEWRAP technique, which works with a Cholesky decomposition. However, NRRIDG usually requires fewer iterations than NEWRAP.

The NRRIDG technique performs well for small- to medium-sized problems, and it does not require many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, either the QUANEW technique or the CONGRA technique might be more efficient.

Dual Quasi-Newton Optimization (QUANEW)

The dual quasi-Newton (QUANEW) technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$, and it does not need to compute second-order derivatives, because they are approximated. It works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. However, the QUANEW technique requires more iterations in general than the TRUREG, NEWRAP, and NRRIDG techniques, which compute second-order derivatives. The QUANEW technique provides an appropriate balance between the speed and stability that are required for most generalized linear model applications.

The QUANEW technique that PROC GAMPL uses is the dual quasi-Newton technique, which updates the Cholesky factor of an approximate Hessian.

In each iteration, a line search is performed along the search direction to find an approximate optimum. The line-search technique uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions (Fletcher 1987). One of the Goldstein conditions can be violated if the feasible region defines an upper limit of the step size. Violating the left-side Goldstein condition can affect the positive-definiteness of the quasi-Newton update. In that case, either the update is skipped or the iterations are restarted by using an identity matrix, resulting in the steepest descent or ascent search direction.

Double-Dogleg Optimization (DBLDOG)

The double-dogleg optimization (DBLDOG) technique combines the ideas of the quasi-Newton and trust region techniques. In each iteration, the double-dogleg technique computes the step $\mathbf{s}^{(k)}$ as the linear combination of the steepest descent or ascent search direction $\mathbf{s}_1^{(k)}$ and a quasi-Newton search direction $\mathbf{s}_2^{(k)}$:

$$\mathbf{s}^{(k)} = \alpha_1 \mathbf{s}_1^{(k)} + \alpha_2 \mathbf{s}_2^{(k)}$$

The step must remain within a prespecified trust region radius (Fletcher 1987, p. 107). Thus, the DBLDOG subroutine uses the dual quasi-Newton update but does not perform a line search.

The double-dogleg optimization technique works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. The implementation is based on Dennis and Mei (1979) and Gay (1983), but it is extended for dealing with boundary and linear constraints. The DBLDOG technique generally requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which require second-order derivatives; however, each of the DBLDOG iterations is computationally cheap. Furthermore, the DBLDOG technique requires only gradient calls for updating of the Cholesky factor of an approximate Hessian.

Conjugate Gradient Optimization (CONGRA)

Second-order derivatives are not required by the CONGRA technique and are not even approximated. The CONGRA technique can be expensive in function and gradient calls, but it requires only $O(p)$ memory for unconstrained optimization. In general, the technique must perform many iterations to obtain a precise solution, but each of the CONGRA iterations is computationally cheap.

The CONGRA technique should be used for optimization problems that have large p. For the unconstrained or boundary-constrained case, the CONGRA technique requires only $O(p)$ bytes of working memory, whereas all other optimization techniques require order $O(p^2)$ bytes of working memory. During p successive iterations that are uninterrupted by restarts or changes in the working set, the CONGRA technique computes a cycle of p conjugate search directions. In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The line-search technique uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit for the step size.

Nelder-Mead Simplex Optimization (NMSIMP)

The Nelder-Mead simplex technique does not use any derivatives and does not assume that the objective function has continuous derivatives. The objective function itself needs to be continuous. This technique is quite expensive in the number of function calls, and it might be unable to generate precise results for $p \gg 40$.

The original Nelder-Mead simplex technique is implemented and extended to boundary constraints. This technique does not compute the objective for infeasible points, but it changes the shape of the simplex and adapts to the nonlinearities of the objective function. This change contributes to an increased speed of convergence and uses a special termination criterion.

Displayed Output

The following sections describe the output that the GAMPL procedure produces by default. The output is organized into various tables, which are discussed in the order of their appearance.

Performance Information

The "Performance Information" table is produced by default. It displays the number of threads used.

If you specify the DETAILS option in the PERFORMANCE statement, PROC GAMPL also produces a "Timing" table, which displays elapsed times (absolute and relative) for the main tasks of the procedure.

Data Access Information

The "Data Access Information" table is produced by default. For the input and output data sets, it displays the libref and data set name, the engine that was used to access the data, the role (input or output) of the data set, and the path that the data followed to reach the computation.

Model Information

The "Model Information" table displays basic information about the model, such as the response variable, frequency variable, link function, and model category that the GAMPL procedure determines based on your input and options. The "Model Information" table also displays the distribution of the data that PROC GAMPL assumes. For information about the supported response distributions, see the section "DISTRIBUTION=*keyword*" on page 3453.

Number of Observations

The "Number of Observations" table displays the number of observations that are read from the input data set and the number of observations that are used in the analysis. If a FREQ statement is present, the sum of the frequencies read and used is displayed. If the events/trials syntax is used, the number of events and trials is also displayed.

Response Profile

The "Response Profile" table displays the ordered values from which the GAMPL procedure determines the probability that is modeled as an event in binary models. For each response category level, the frequency that is used in the analysis is reported. You can affect the ordering of the response values by specifying *response-options* in the MODEL statement. For binary models, the note that follows the "Response Profile" table indicates which outcome is modeled as the event in binary models and which value serves as the reference category.

The "Response Profile" table is not produced for binomial (events/trials) data. You can find information about the number of events and trials in the "Number of Observations" table.

Class Level Information

The "Class Level Information" table lists the levels of every variable that is specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress the "Class Level Information" table completely or partially by specifying the NOCLPRINT= option in the PROC GAMPL statement.

If the classification variables use reference parameterization, the "Class Level Information" table also displays the reference value for each variable.

Specifications for Spline(*spline-variables*)

The "Specifications for Spline(*spline-variables*)" table displays basic information (such as number of variables, specified degrees of freedom, search range for the smoothing parameter, and so on) about how to construct a spline term that the GAMPL procedure uses to construct basis expansions and search for the smoothing parameter. PROC GAMPL generates the "Specifications for Spline(*spline-variables*)" table only when you specify the DETAILS option for a spline term.

Optimization Iteration History

For each iteration of the optimization, the "Iteration History" table displays the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element. The objective function that PROC GAMPL uses in the optimization is normalized by default to enable comparisons across data sets that have different sampling intensity.

If you specify the ITDETAILS option in the PROC GAMPL statement, information about the parameter estimates and gradients in the course of the optimization is added to the "Iteration History" table.

For a parametric generalized linear model or for a generalized additive model that has fixed smoothing parameters, the "Iteration History" table displays information about regression parameter estimates and gradients. For a generalized additive model that has unknown smoothing parameters, the "Iteration History" table displays information about smoothing parameter estimates and gradients. If the performance iteration method is used, a column of performance iteration steps is added to the table.

Convergence Status

The convergence status table is a small ODS table that follows the "Iteration History" table in the default output. In the listing, this table appears as a message that indicates whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the convergence status table to an output data set, a numeric Status variable is added that enables you to programmatically assess convergence. The values of the Status variable encode the following:

0 Convergence was achieved, or an optimization was not performed because TECHNIQUE=NONE was specified.

1 The objective function could not be improved.

2 Convergence was not achieved because of a user interrupt or because a limit (such as the maximum number of iterations or the maximum number of function evaluations) was reached. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC GAMPL statement.

3 Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Fit Statistics

The "Fit Statistics" table displays a variety of likelihood-based measures of fit in addition to the model roughness measurement. All information criteria are presented in "smaller is better" form.

The calculation of the information criteria uses the following formulas, where df denotes the model degrees of freedom, f denotes the number of frequencies used, and ℓ is the log likelihood that is evaluated at the converged estimates:

$$\mathrm{AIC} = -2\ell + 2\mathrm{df}$$

$$\mathrm{AICC} = \begin{cases} -2\ell + 2\mathrm{df}\, f/(f - \mathrm{df} - 1) & \text{when } f > \mathrm{df} + 2 \\ -2\ell + 2\mathrm{df}(\mathrm{df} + 2) & \text{otherwise} \end{cases}$$

$$\mathrm{BIC} = -2\ell + \mathrm{df}\log(f)$$

If no FREQ statement is specified, then f equals n, the number of observations that are used.

Parameter Estimates

The "Parameter Estimates" table displays the regression parameter estimates, their estimated (asymptotic) standard errors, chi-square statistics, and p-values for the hypothesis that the parameter is 0.

Estimates for Smoothing Components

The "Estimates for Smoothing Components" table displays a summary of the fitted spline terms, including effective degrees of freedom, smoothing parameters, roughness penalty values, number of parameters, rank of penalty matrix, and number of knots.

Tests for Smoothing Components

The "Tests for Smoothing Components" table displays effective degrees of freedom, effective degrees of freedom for test, F/χ_2 statistics, and p-values for rejecting the hypothesis that the smoothing component has zero contribution.

ODS Table Names

Each table that the GAMPL procedure creates has a name that is associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 49.13.

Table 49.13 ODS Tables Produced by PROC GAMPL

Table Name	Description	Statement	Option
ClassLevels	Level information from the CLASS statement	CLASS	
ConvergenceStatus	Status of optimization at conclusion of optimization	Default output	
DataAccessInfo	Information about modes of data access	Default output	
FitStatistics	Fit statistics	Default output	
LikelihoodHist	Iteration history for maximum likelihood estimation or penalized likelihood estimation	PROC GAMPL	ITDETAILS
ModelInfo	Information about the modeling environment	Default output	
NObs	Number of observations read and used, and number of events and trials, if applicable	Default output	
ParameterEstimates	Solutions for the parameter estimates that are associated with effects in MODEL statements	Default output	
PerformanceInfo	Information about the high-performance computing environment	Default output	
ResponseProfile	Response categories and the category that is modeled in models for binary and multinomial data	Default output	
SmoothingEstimates	Information for spline terms after model fitting	Default output	

Table 49.13 *continued*

Table Name	Description	Statement	Option
SmoothingHist	Iteration history for smoothing parameter estimation	PROC GAMPL	ITDETAILS
SmoothingTests	Smoothing components test result	Default output	
SplineDetails	Information about spline construction and smoothing parameter search	MODEL	SPLINE(*variables* / DETAILS)
Timing	Absolute and relative times for tasks performed by the procedure	PERFORMANCE	DETAILS

By referring to the names of such tables, you can use the ODS OUTPUT statement to place one or more of these tables in output data sets.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, the GAMPL procedure by default produces plots of the partial predictions for each spline term in the model. Use the PLOTS option in the PROC GAMPL statement to control aspects of these plots.

PROC GAMPL assigns a name to each graph that it creates by using ODS. You can use these names to refer to the graphs when using ODS. The names are listed in Table 49.14.

Table 49.14 Graphs Produced by PROC GAMPL

ODS Graph Name	Plot Description	PLOTS= Option
SmoothingComponentPanel	Panel of multiple prediction curves	Default COMPONENTS
SmoothingComponentPlot	Unpacked prediction curves	PLOTS(UNPACK) PLOTS=COMPONENTS(UNPACK)

By default, prediction plots for each spline component are displayed in panels that contain at most six plots. If you specify more than six smoothing components, multiple panels are used. Use the PLOTS(UNPACK) option in the PROC GAMPL statement to display these plots individually.

Examples: GAMPL Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 49.1: Scatter Plot Smoothing

(View the complete code for this example (hpgamex1.sas) in the example repository.)

This example shows how you can use PROC GAMPL to perform scatter plot smoothing.

The example uses the LIDAR data set (Ruppert, Wand, and Carroll 2003). This data set is used in many books and journals to illustrate different smoothing techniques. Scientists use a technique known as LIDAR (light detection and ranging), which uses laser reflections to detect chemical compounds in the atmosphere. The following DATA step creates the data set Lidar:

```
title 'Scatter Plot Smoothing';
data Lidar;
   input Range LogRatio @@;
   datalines;
390  -0.05035573    391  -0.06009706    393  -0.04190091    394   -0.0509847
396  -0.05991345    397  -0.02842392    399  -0.05958421    400  -0.03988881
402  -0.02939582    403  -0.03949445    405  -0.04764749    406     -0.06038
408  -0.03123034    409  -0.03816584    411  -0.07562269    412  -0.05001751
414   -0.0457295    415  -0.07766966    417  -0.02460641    418  -0.07133184

   ... more lines ...

702   -0.4716702    703   -0.7801088    705   -0.6668431    706   -0.5783479
708   -0.7874522    709   -0.6156956    711   -0.8967602    712   -0.7077379
714    -0.672567    715   -0.6218413    717   -0.8657611    718    -0.557754
720   -0.8026684
;
```

In this data set, Range records the distance that light travels before it is reflected back to the source. LogRatio is the logarithm of the ratio of light that is received from two laser sources. The objective is to use scatter plot smoothing to discover the nonlinear pattern in the data set. SAS provides different methods (for example local regression) for scatter plot smoothing. You can perform scatter plot smoothing by using the SGPLOT procedure, as shown in the following statements:

```
proc sgplot data=Lidar;
   scatter  x=Range y=LogRatio;
   loess    x=Range y=LogRatio / nomarkers;
   pbspline x=Range y=LogRatio / nomarkers;
run;
```

Output 49.1.1 shows the scatter plot of Range and LogRatio and the smoothing curves that are fitted by local regression and penalized B-splines smoothing techniques.

Output 49.1.1 Scatter Plot Smoothing

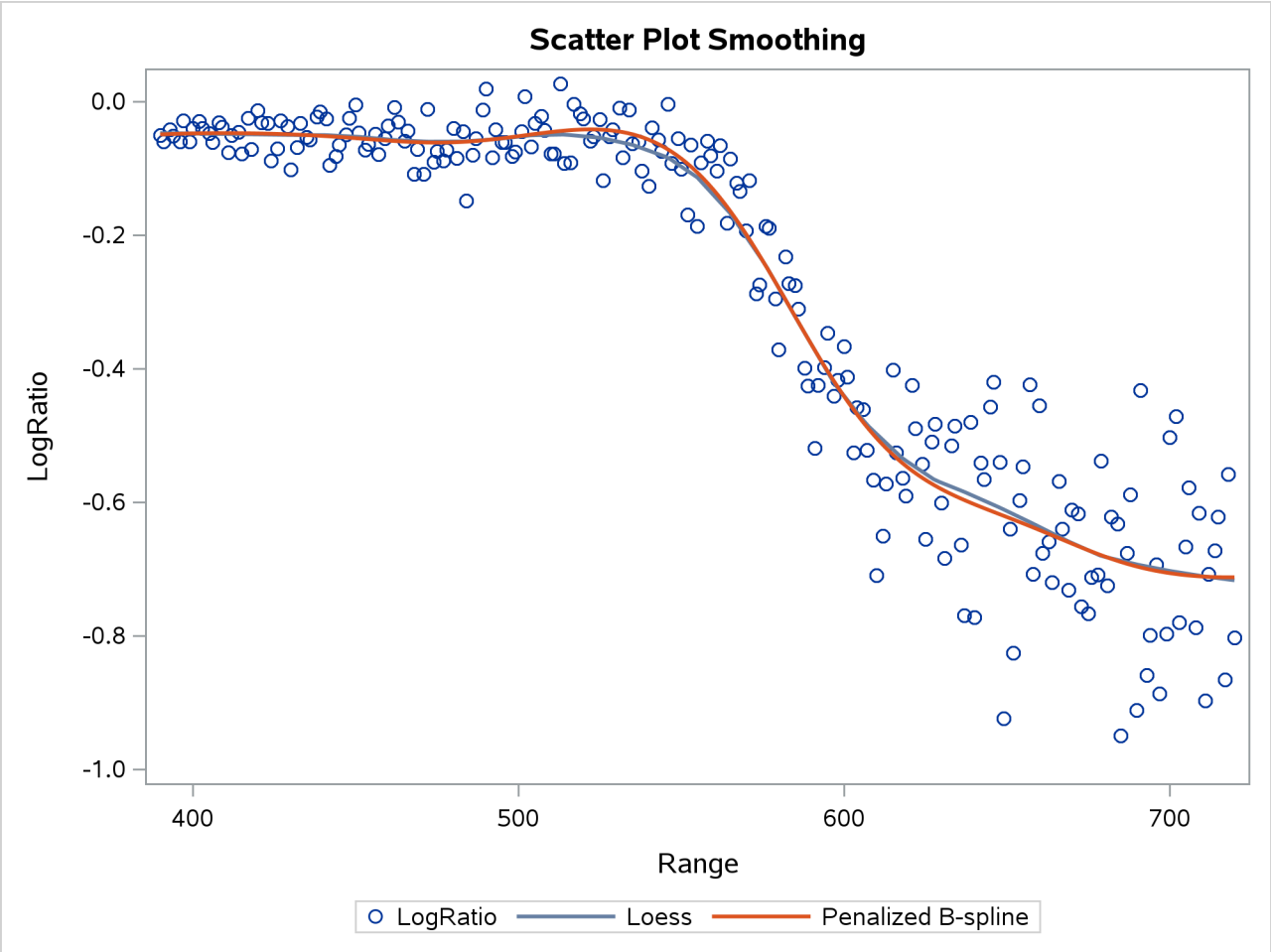

Both scatter plot smoothing techniques show a significant nonlinear structure between Range and LogRatio that cannot be easily modeled by ordinary polynomials. You can also use the GAMPL procedure to perform scatter plot smoothing on this data set, as in the following statements:

```
proc gampl data=Lidar seed=12345;
   model LogRatio = spline(Range/details);
   output out=LidarOut pred=p;
run;
```

The "Specifications for Spline(Range)" table in Output 49.1.2 displays the specifications for constructing the spline term for Range. The maximum degrees of freedom is 10, which sets the upper limit of effective degrees of freedom for the spline term to be 9 after one degree of freedom is absorbed in the intercept. The order of the derivative in the penalty is 2, which means the unpenalized portion of the spline term involves polynomials with degrees up to 2.

Output 49.1.2 Spline Specification

Scatter Plot Smoothing

The GAMPL Procedure

Specifications for Spline(Range)	
Number of Variables	1
Rank of Penalty Approximation	10
Order of Derivative in the Penalty	2
Maximum Number of Knots	2000

The "Fit Statistics" table in Output 49.1.3 shows the summary statistics for the fitted model.

Output 49.1.3 Fit Statistics

Fit Statistics	
Penalized Log Likelihood	251.44124
Roughness Penalty	0.00000426
Effective Degrees of Freedom	9.99988
Effective Degrees of Freedom for Error	211.00000
AIC (smaller is better)	-482.88271
AICC (smaller is better)	-481.83512
BIC (smaller is better)	-448.90148
GCV (smaller is better)	0.00654

The "Estimates for Smoothing Components" table in Output 49.1.4 shows that the effective degrees of freedom for the spline term of Range is approximately 8 after the GCV criterion is optimized with respect to the smoothing parameter. The roughness penalty is small, suggesting that there is an important contribution from the penalized part of thin-plate regression splines beyond nonpenalized polynomials.

Output 49.1.4 Estimates for Smoothing Components

Estimates for Smoothing Components						
Component	Effective DF	Smoothing Parameter	Roughness Penalty	Number of Parameters	Rank of Penalty Matrix	Number of Knots
Spline(Range)	7.99988	1.0000	4.263E-6	9	10	221

Because the optimal model is obtained by searching in a functional space that is constrained by the maximum degrees of freedom for a spline term, you might wonder whether PROC GAMPL produces a much different model if you increase the value. The following statements fit another model in which the maximum degrees of freedom is increased to 20:

```
proc gampl data=Lidar seed=12345;
   model LogRatio = spline(Range/maxdf=20);
   output out=LidarOut2 pred=p2;
run;
```

Output 49.1.5 displays fit summary statistics for the second model. The model fit statistics from the second model are very close to the ones from the first model, indicating that the second model is not much different

from the first model.

Output 49.1.5 Fit Statistics

Scatter Plot Smoothing

The GAMPL Procedure

Fit Statistics	
Penalized Log Likelihood	250.95136
Roughness Penalty	0.06254
Effective Degrees of Freedom	10.04384
Effective Degrees of Freedom for Error	209.03211
AIC (smaller is better)	-481.87757
AICC (smaller is better)	-480.82094
BIC (smaller is better)	-447.74696
GCV (smaller is better)	0.00657

Output 49.1.6 shows that the effective degrees of freedom for the spline term of Range is slightly larger than 8, which is understandable because increasing the maximum degrees of freedom expands the functional space for model searching. Functions in the expanded space can provide a better fit to the data, but they are also penalized more because the roughness penalty value for the second model is much larger than the one for the first model. This suggests that functions in the expanded space do not help much, given the nonlinear relationship between Range and LogRatio.

Output 49.1.6 Estimates for Smoothing Components

Estimates for Smoothing Components						
Component	Effective DF	Smoothing Parameter	Roughness Penalty	Number of Parameters	Rank of Penalty Matrix	Number of Knots
Spline(Range)	8.04384	23134.6	0.0625	19	20	221

The two fitted models are all based on thin-plate regression splines, in which polynomials that have degrees higher than 2 are penalized. You might wonder whether allowing higher-order polynomials yields a much different model. The following statements fit a third spline model by penalizing polynomials that have degrees higher than 3:

```
proc gampl data=Lidar seed=12345;
   model LogRatio = spline(Range/m=3);
   output out=LidarOut3 pred=p3;
run;
```

The fit summary statistics shown in Output 49.1.7 are close to the ones from the previous two models, albeit slightly smaller.

Output 49.1.7 Fit Statistics

Scatter Plot Smoothing

The GAMPL Procedure

Fit Statistics	
Penalized Log Likelihood	249.79779
Roughness Penalty	9.440383E-9
Effective Degrees of Freedom	10.00000
Effective Degrees of Freedom for Error	211.00000
AIC (smaller is better)	-479.59559
AICC (smaller is better)	-478.54797
BIC (smaller is better)	-445.61397
GCV (smaller is better)	0.00664

As shown in Output 49.1.8, the effective degrees of freedom for the spline term where polynomials with degrees less than 4 are allowed without penalization is 8. The roughness penalty is quite small compared to the previous two fits. This also suggests that there are important contributions from the penalized part of the thin-plate regression splines even after the nonpenalized polynomials are raised to order 3.

Output 49.1.8 Estimates for Smoothing Components

Estimates for Smoothing Components						
Component	Effective DF	Smoothing Parameter	Roughness Penalty	Number of Parameters	Rank of Penalty Matrix	Number of Knots
Spline(Range)	8.00000	1.0000	9.44E-9	9	10	221

The following statements use the DATA step to merge the predictions from the three scatter plot smoothing fits by PROC GAMPL and use the SGPLOT procedure to visualize them:

```
data LidarPred;
   merge Lidar LidarOut LidarOut2 LidarOut3;
run;

proc sgplot data=LidarPred;
   scatter x=Range y=LogRatio / markerattrs=GraphData1(size=7);
   series  x=Range y=p        / lineattrs  =GraphData2(thickness=2)
                                legendlabel="Spline 1";
   series  x=Range y=p2       / lineattrs  =GraphData3(thickness=2)
                                legendlabel="Spline 2";
   series  x=Range y=p3       / lineattrs  =GraphData4(thickness=2)
                                legendlabel="Spline 3";
run;
```

Output 49.1.9 displays the scatter plot smoothing fits by PROC GAMPL under three different spline specifications.

Output 49.1.9 Scatter Plot Smoothing

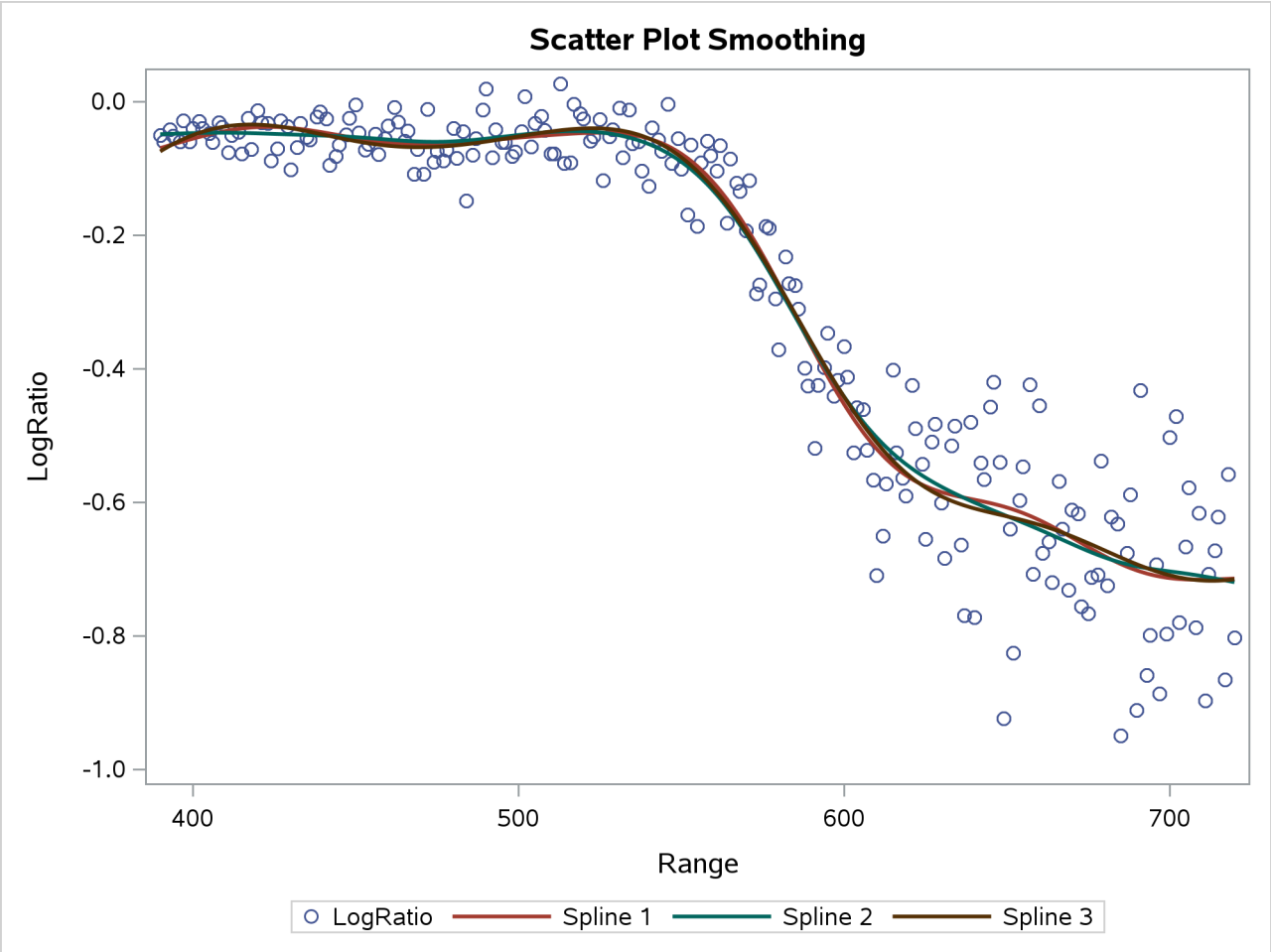

Example 49.2: Nonparametric Logistic Regression

(View the complete code for this example (hpgamex2.sas) in the example repository.)

This example shows how you can use PROC GAMPL to build a nonparametric logistic regression model for a diabetes data set that contains a binary response and then use that model to classify observations.

Diabetes is a major American disease. The American Diabetes Association estimates that over 8% of Americans have diabetes, and diabetes costs Americans over $175 billion a year. The National Institute of Diabetes and Digestive and Kidney Diseases (NIDDK) has been studying diabetes and obesity in the Akimel O'otham (formerly known as Pima Indians) in Arizona for over 30 years. Smith et al. (1988) prepared some of the NIDDK data for forecasting the onset of diabetes and then donated the data for community use. In this particular example, data are for female patients who are at least 21 years old. The objective of this study is to investigate the relationship between a diabetes diagnosis and variables that represent physiological measurements and medical attributes. Some missing or invalid observations are removed from the analysis. The reduced data set contains 532 records. The following DATA step creates the data set DiabetesStudy:

```
title 'Diabetes Study';
data DiabetesStudy;
   input NPreg Glucose Pressure Triceps BMI Pedigree Age Diabetes Test@@;
   datalines;
 6  148  72  35  33.6  0.627  50  1  1    1   85  66  29  26.6  0.351  31  0  1
 1   89  66  23  28.1  0.167  21  0  0    3   78  50  32    31  0.248  26  1  0
 2  197  70  45  30.5  0.158  53  1  0    5  166  72  19  25.8  0.587  51  1  1
 0  118  84  47  45.8  0.551  31  1  1    1  103  30  38  43.3  0.183  33  0  1
 3  126  88  41  39.3  0.704  27  0  0    9  119  80  35    29  0.263  29  1  0

   ... more lines ...

 1  128  48  45  40.5  0.613  24  1  0    2  112  68  22  34.1  0.315  26  0  0
 1  140  74  26  24.1  0.828  23  0  1    2  141  58  34  25.4  0.699  24  0  1
 7  129  68  49  38.5  0.439  43  1  1    0  106  70  37  39.4  0.605  22  0  0
 1  118  58  36  33.3  0.261  23  0  1    8  155  62  26    34  0.543  46  1  0
;
```

The data set contains nine variables, including the binary response variable Diabetes. Table 49.15 describes the variables.

Table 49.15 Variables in the DiabetesStudy Data Set

Variable	Description
NPreg	Number of pregnancies
Glucose	Two-hour plasma glucose concentration in an oral glucose tolerance test
Pressure	Diastolic blood pressure (mm Hg)
Triceps	Triceps skin fold thickness (mm)
BMI	Body mass index (weight in kg/(height in m)^2)
Pedigree	Diabetes pedigree function
Age	Age (years)
Diabetes	0 if test negative for diabetes, 1 if test positive
Test	0 for training role, 1 for test

The Test variable splits the data set into training and test subsets. The training observations (whose Test value is 0) hold approximately 59.4% of the data. To build a model that is based on the training data and evaluate its performance by predicting the test data, you use the following statements to create a new variable, Result, whose value is the same as that of the Diabetes variable for a training observation and is missing for a test observation:

```
data DiabetesStudy;
   set DiabetesStudy;
   Result = Diabetes;
   if Test=1 then Result=.;
run;
```

As a starting point of your analysis, you can build a parametric logistic regression model on the training data and predict the test data. The following statements use PROC HPLOGISTIC to perform the analysis:

```
proc hplogistic data=DiabetesStudy;
   model Diabetes(event='1') = NPreg Glucose Pressure Triceps
                               BMI Pedigree Age;
   partition role=Test(test='1' train='0');
run;
```

Output 49.2.1 shows the summary statistics from the parametric logistic regression model.

Output 49.2.1 Fit Statistics

Diabetes Study

The HPLOGISTIC Procedure

Fit Statistics		
Description	Training	Testing
-2 Log Likelihood	295.79	176.90
AIC (smaller is better)	311.79	192.90
AICC (smaller is better)	312.24	193.65
BIC (smaller is better)	342.19	219.37

Output 49.2.2 shows fit statistics for both training and test subsets of the data, including the misclassification error for the test data.

Output 49.2.2 Partition Fit Statistics

Partition Fit Statistics		
Statistic	Training	Testing
Area under the ROCC	0.8607	0.8547
Average Square Error	0.1452	0.1401
Hosmer-Lemeshow Test	0.9726	0.1487
Misclassification Error	0.2061	0.2178
R-Square	0.3438	0.2557
Max-rescaled R-Square	0.4695	0.3706
McFadden's R-Square	0.3197	0.2522
Mean Difference	0.3780	0.3605
Somers' D	0.7213	0.7093
True Negative Fraction	0.8702	0.8435
True Positive Fraction	0.6639	0.6182

The parametric logistic regression model is restricted in the sense that all variables affect the response in strictly linear fashion. If you are uncertain that a variable is an important factor and its contribution is linear in predicting the response, you might want to choose a nonparametric logistic regression model to fit the data. You can use PROC GAMPL to form a nonparametric model by including the spline transformation of each explanatory variable, as shown in the following statements:

```
proc gampl data=DiabetesStudy seed=12345;
   model Result(event='1') = spline(NPreg)     spline(Glucose)
                             spline(Pressure)  spline(Triceps)
                             spline(BMI)       spline(Pedigree)
                             spline(Age)  / dist=binary;
```

```
run;
```

Because the response variable Result is binary, the DIST=BINARY option in the MODEL statement specifies a binary distribution for the response variable. By default, PROC GAMPL models the probability of the first ordered response category, which is a negative diabetes testing result in this case. The EVENT= option specifically requests that PROC GAMPL model the probability of positive diabetes testing results. The "Response Profile" table in Output 49.2.3 shows the frequencies of the response in both categories.

Output 49.2.3 Response Profile

Diabetes Study

The GAMPL Procedure

Response Profile		
Ordered Value	Result	Total Frequency
1	0	208
2	1	122

You are modeling the probability that Result='1'.

Output 49.2.4 lists the summary statistics from the nonparametric logistic regression model, which include spline transformations of all variables.

Output 49.2.4 Fit Statistics

Fit Statistics	
Penalized Log Likelihood	-130.55095
Roughness Penalty	3.20011
Effective Degrees of Freedom	25.85896
Effective Degrees of Freedom for Error	302.97752
AIC (smaller is better)	309.61970
AICC (smaller is better)	314.20203
BIC (smaller is better)	407.86031
UBRE (smaller is better)	0.14225

The "Tests for Smoothing Components" table in Output 49.2.5 shows approximate tests results. Although some spline terms are significant, others are not. The null testing hypothesis is whether the total contribution from a variable is 0. So you can form a reduced model by removing those nonsignificant spline terms from the model. In this case, spline transformations for NPreg, Pressure, BMI, and Triceps are dropped from the model because their p-values are larger than the 0.1 nominal level.

Output 49.2.5 Tests for Smoothing Components

Tests for Smoothing Components				
Component	Effective DF	Effective DF for Test	Chi-Square	Pr > ChiSq
Spline(NPreg)	1.00001	1	0.0758	0.7831
Spline(Glucose)	1.00000	1	38.4662	<.0001
Spline(Pressure)	7.80438	8	10.2605	0.2472
Spline(Triceps)	1.00000	1	0.6343	0.4258
Spline(BMI)	8	8	11.8047	0.1601
Spline(Pedigree)	1.00002	1	8.5323	0.0035
Spline(Age)	5.05456	7	15.7671	0.0273

The following statements use PROC GAMPL to fit a reduced nonparametric logistic regression model. The OUTPUT statement requests predicted probabilities for both training and test data sets. The ID statement requests that the Diabetes and Test variables also be included in the output data set so that you can use them to identify test observations and compute misclassification errors.

```
ods graphics on;
proc gampl data=DiabetesStudy plots seed=12345;
   model Result(event='1') = spline(Glucose)
                             spline(Pedigree) spline(Age) / dist=binary;
   output out=DiabetesStudyOut;
   id Diabetes Test;
run;
```

Output 49.2.6 shows the summary statistics from the reduced nonparametric logistic regression model. The values of the information criteria are better than of the parametric logistic regression model.

Output 49.2.6 Fit Statistics

Diabetes Study

The GAMPL Procedure

Fit Statistics	
Penalized Log Likelihood	-149.85765
Roughness Penalty	2.85613
Effective Degrees of Freedom	8.05242
Effective Degrees of Freedom for Error	320.61181
AIC (smaller is better)	312.96402
AICC (smaller is better)	313.41826
BIC (smaller is better)	343.55593
UBRE (smaller is better)	-0.00230

In the "Estimates for Smoothing Components" table in Output 49.2.7, PROC GAMPL reports that the effective degrees of freedom value for spline transformations of Glucose is quite close to 1. This suggests strictly linear form for Glucose. For Pedigree, the degrees of freedom value demonstrates a moderate amount of nonlinearity. For Age, the degrees of freedom value is much larger than 1. The measure suggests a nonlinear pattern in the dependency of the response on Age.

Output 49.2.7 Estimates for Smoothing Components

Estimates for Smoothing Components						
Component	Effective DF	Smoothing Parameter	Roughness Penalty	Number of Parameters	Rank of Penalty Matrix	Number of Knots
Spline(Glucose)	1.00000	9.032E10	2.711E-7	9	10	110
Spline(Pedigree)	1.51071	0.4383	0.5086	9	10	283
Spline(Age)	4.54171	69.8810	2.3475	9	10	42

The "Tests for Smoothing Components" table in Output 49.2.8 shows that all spline transformations are significant in predicting diabetes testing results.

Output 49.2.8 Tests for Smoothing Components

Tests for Smoothing Components				
Component	Effective DF	Effective DF for Test	Chi-Square	Pr > ChiSq
Spline(Glucose)	1.00000	1	53.0363	<.0001
Spline(Pedigree)	1.51071	2	9.9354	0.0070
Spline(Age)	4.54171	6	23.0661	0.0008

The smoothing component panel (which is produced by the PLOTS option and is shown in Output 49.2.9) visualizes the spline transformations for the four variables in addition to 95% Bayesian curvewise confidence bands. For Glucose, the spline transformation is almost a straight line. For Pedigree, the spline transformation shows a slightly nonlinear trend. For Age, the dependency is obviously nonlinear.

Output 49.2.9 Smoothing Components Panel

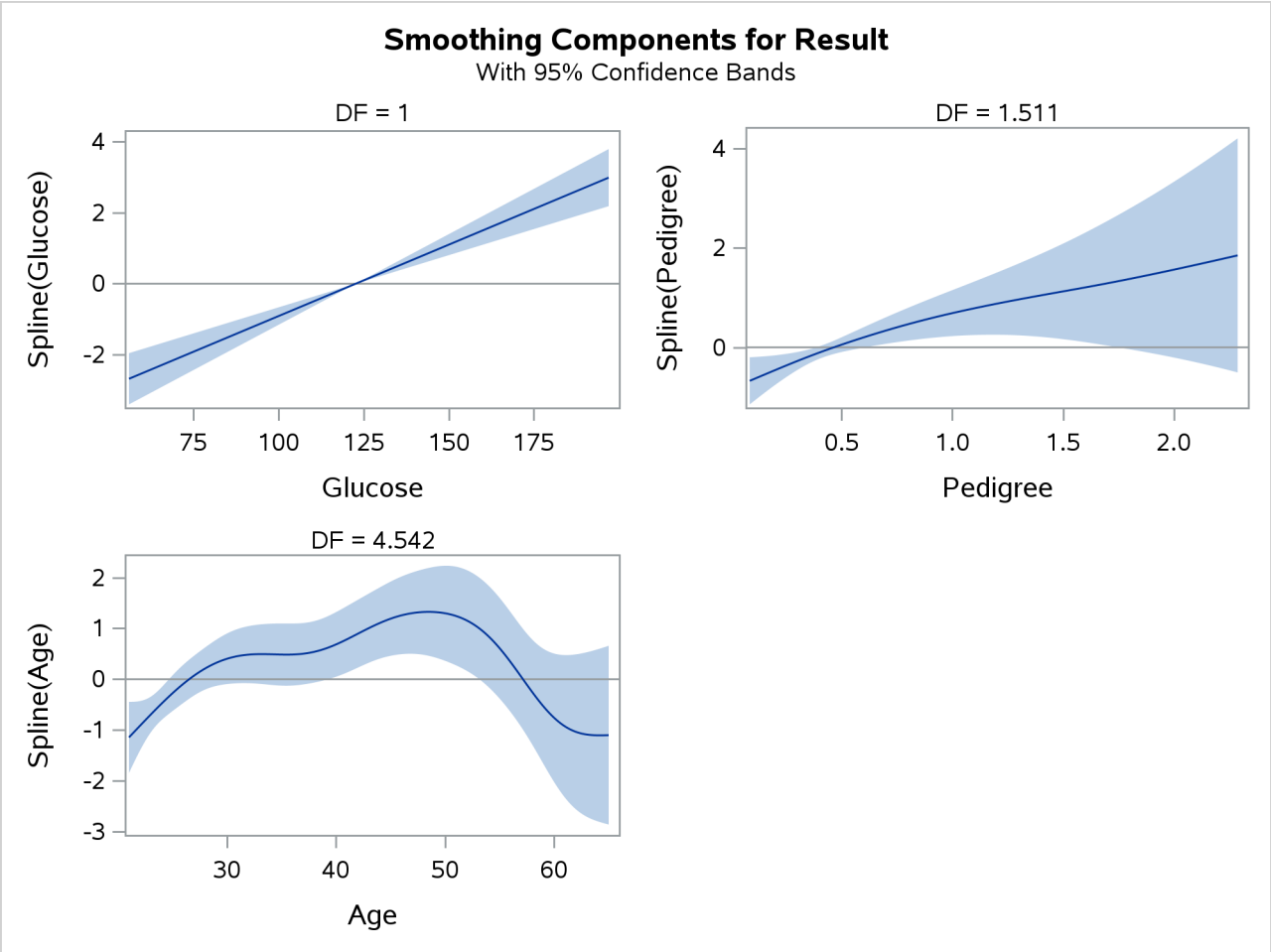

The following statements compute the misclassification error on the test data set from the reduced nonparametric logistic regression model that PROC GAMPL produces:

```
data test;
   set DiabetesStudyOut(where=(Test=1));
   if ((Pred>0.5 & Diabetes=1) | (Pred<0.5 & Diabetes=0))
   then Error=0;
   else Error=1;
run;

proc freq data=test;
   tables Diabetes*Error/nocol norow;
run;
```

Output 49.2.10 shows the misclassification errors for observations in the test set and observations of each response category. The error is smaller than the error from the parametric logistic regression model.

Output 49.2.10 Crosstabulation Table for Test Set Prediction

Diabetes Study

The FREQ Procedure

Frequency Percent	Table of Diabetes by Error		
		Error	
Diabetes	0	1	Total
0	130 64.36	17 8.42	147 72.77
1	35 17.33	20 9.90	55 27.23
Total	165 81.68	37 18.32	202 100.00

Example 49.3: Semiparametric Negative Binomial Model for Mackerel Egg Density

(View the complete code for this example (hpgamex3.sas) in the example repository.)

This example demonstrates how you can use PROC GAMPL to fit a semiparametric negative binomial model for a count data set that has overdispersions.

The example concerns a study of mackerel egg density. The data are a subset of the 1992 mackerel egg survey that was conducted over the Porcupine Bank west of Ireland. The survey took place in the peak spawning area. Scientists took samples by hauling a net up from deep sea to the sea surface. Then they counted the number of spawned mackerel eggs and used other geographic information to estimate the sizes and distributions of spawning stocks. The data set is used as an example in Bowman and Azzalini (1997).

The following DATA step creates the data set Mackerel. This data set contains 634 observations and five variables. The response variable Egg_Count is the number of mackerel eggs that are collected from each sampling net. Longitude and Latitude are the location values in degrees east and north, respectively, of each sample station. Net_Area is the area of the sampling net in square meters. Depth records the sea bed depth in meters at the sampling location, and Distance is the distance in geographic degrees from the sample location to the continental shelf edge.

```
title 'Mackerel Egg Density Study';
data Mackerel;
   input Egg_Count Longitude Latitude  Net_Area Depth Distance;
   datalines;
   0     -4.65     44.57     0.242    4342    0.8395141177
   0     -4.48     44.57     0.242    4334    0.8591926336
   0      -4.3     44.57     0.242    4286    0.8930152895
   1     -2.87     44.02     0.242    1438    0.3956408691
   4     -2.07     44.02     0.242     166    0.0400088237
   3     -2.13     44.02     0.242     460    0.0974234463
   0     -2.27     44.02     0.242     810    0.2362566569

   ... more lines ...
```

```
  22     -4.22     46.25      0.19     205    0.1181120828
  21     -4.28     46.25      0.19     237     0.129990854
   0     -4.73     46.25      0.19    2500    0.3346500536
   5     -4.25     47.23      0.19     114     0.718192582
   3     -3.72     47.25      0.19     100    0.9944669778
   0     -3.25     47.25      0.19      64    1.2639918431
;
```

The response values are counts, so the Poisson distribution might be a reasonable model. The statistic of interest is the mackerel egg density, which can be formed as

$$\text{Density} = E(\text{Count})/\text{Net_Area}$$

This is equivalent to a Poisson regression that uses the response variable Egg_Count, an offset variable $\log(\text{Net_Area})$, and other covariates.

The following statements produce the plot of the mackerel egg density with respect to the sampling station location:

```
data Temp;
   set Mackerel;
   Density = Egg_Count/Net_Area;
run;

%let off0 = offsetmin=0 offsetmax=0
            linearopts=(thresholdmin=0 thresholdmax=0);
%let off0 = xaxisopts=(&off0) yaxisopts=(&off0);
proc template;
   define statgraph surface;
      dynamic _title _z;
      begingraph / designwidth=defaultDesignHeight;
         entrytitle _title;
         layout overlay / &off0;
            contourplotparm z=_z y=Latitude x=Longitude / gridded=FALSE;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=Temp template=surface;
   dynamic _title='Mackerel Egg Density' _z='Density';
run;
```

Output 49.3.1 displays the mackerel egg density in the sampling area. The black hole in the upper right corner is caused by missing values in that area.

Output 49.3.1 Mackerel Egg Density

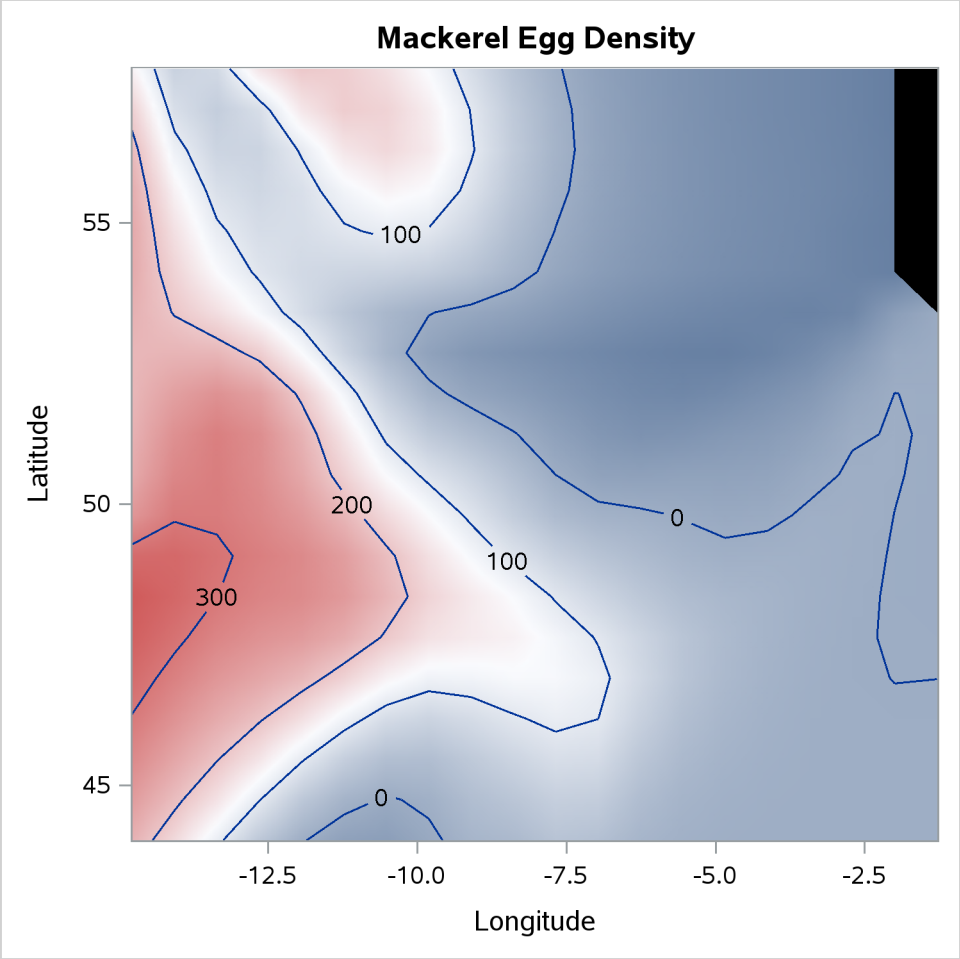

In this example, the dependent variable is the mackerel egg count, the independent variables are the geographical information about each of the sampling stations, and the offset variable is the logarithm of the sampling area. The following statements use PROC GAMPL to fit the semiparametric Poisson regression model. In the program, one parametric term for Distance models the linear dependency of egg density, and another univariate spline term for Depth models the nonlinear dependency of egg density. The maximum degrees of freedom for the univariate spline term is set to 5 to enforce a smooth fit. One additional bivariate spline term for both Longitude and Latitude models the nonlinear spatial effect. To allow more flexibility of the bivariate spline term, its maximum degrees of freedom is increased to 40.

```
data Mackerel;
   set Mackerel;
   Depth = Depth/1000;
   Log_Net_Area = log(Net_Area);
run;

proc gampl data=Mackerel plots seed=321;
   model Egg_Count = param(Distance) spline(Depth/maxdf=5)
                     spline(Longitude Latitude/maxdf=40)
                      / offset=Log_Net_Area dist=poisson;
   output out=MackerelOut1 pred=Pred1 pearson=Pea1;
```

```
run;
```

Output 49.3.2 displays the summary statistics from the Poisson model.

Output 49.3.2 Fit Statistics

Mackerel Egg Density Study

The GAMPL Procedure

Fit Statistics	
Penalized Log Likelihood	-2792.30379
Roughness Penalty	8.67818
Effective Degrees of Freedom	43.40594
Effective Degrees of Freedom for Error	589.83438
AIC (smaller is better)	5662.74129
AICC (smaller is better)	5669.27963
BIC (smaller is better)	5855.98667
UBRE (smaller is better)	7.10457

The "Tests for Smoothing Components" table in Output 49.3.3 shows that both univariate and bivariate spline terms are significant.

Output 49.3.3 Tests for Smoothing Components

Tests for Smoothing Components				
Component	Effective DF	Effective DF for Test	Chi-Square	Pr > ChiSq
Spline(Depth)	3.85366	4	166.2721	<.0001
Spline(Longitude Latitude)	37.55228	39	4968.8033	<.0001

The smoothing component panel in Output 49.3.4 contains a contour for the spatial effect by Longitude and Latitude and one fitted univariate spline for Depth, both demonstrating significantly nonlinear structures.

Output 49.3.4 Smoothing Components Panel

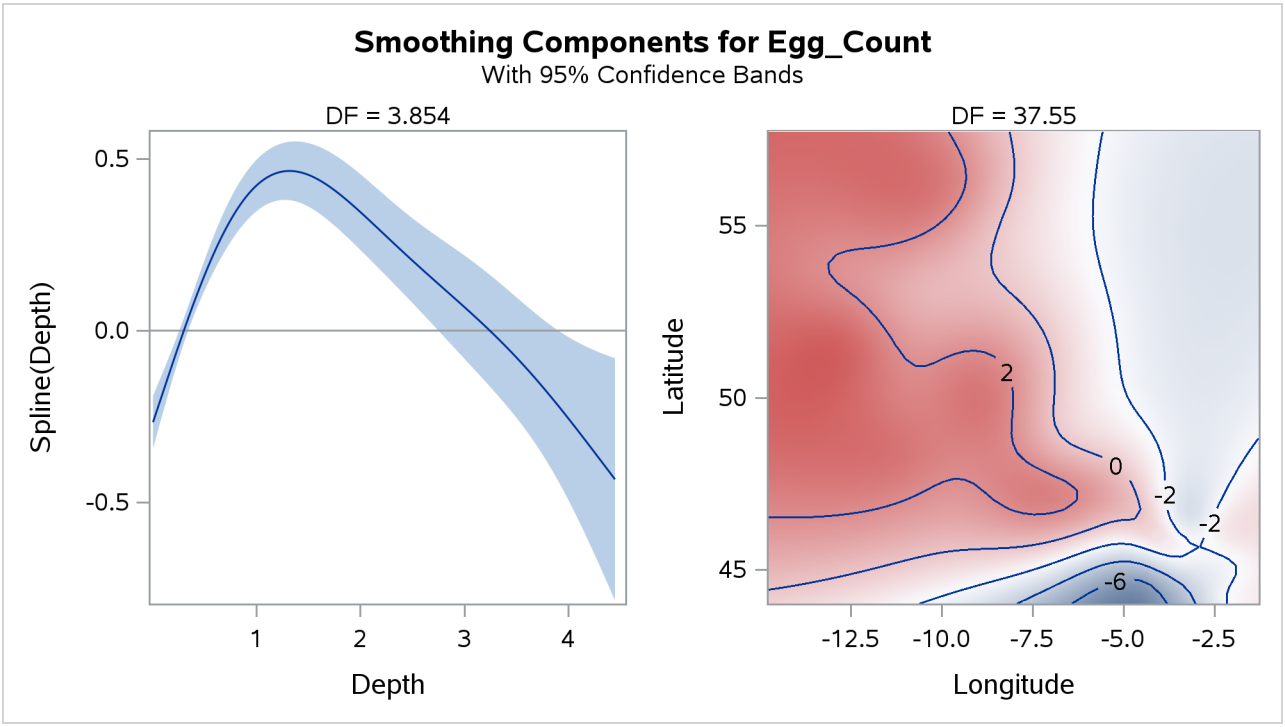

Overdispersion sometimes occurs for count data. One way to measure whether the data are overdispersed is to determine whether Pearson's chi-square divided by the error degrees of freedom is much larger than 1. You can compute that statistic by taking the sum of squares of the Pearson residuals in the MackerelOut1 data set and then dividing that sum by the error degrees of freedom that is reported in Output 49.3.2. The computed value is approximately 8, which is much larger than 1, indicating the existence of overdispersion in this data set.

There are many ways to solve the overdispersion problem. One approach is to fit a negative binomial distribution in which the variance of the mean contains a dispersion parameter. The following statements use PROC GAMPL to fit the semiparametric negative binomial regression model:

```
proc gampl data=Mackerel plots seed=321;
   model Egg_Count = param(Distance) spline(Depth/maxdf=5)
                     spline(Longitude Latitude/maxdf=40)
                      / offset=Log_Net_Area dist=negbin;
   output out=MackerelOut2 pred=Pred2 pearson=Pea2;
run;
```

Output 49.3.5 displays the summary statistics from the negative binomial model. The model's effective degrees of freedom is less than that of the Poisson model, even though the dispersion parameter costs one extra degree of freedom to fit. The values of the information criteria are also much less, indicating a better fit.

Output 49.3.5 Fit Statistics

Mackerel Egg Density Study

The GAMPL Procedure

Fit Statistics	
Penalized Log Likelihood	-1561.91513
Roughness Penalty	17.57290
Effective Degrees of Freedom	38.88268
Effective Degrees of Freedom for Error	590.88741
AIC (smaller is better)	3184.02271
AICC (smaller is better)	3189.24304
BIC (smaller is better)	3357.13028
UBRE (smaller is better)	0.53927

The "Tests for Smoothing Components" table in Output 49.3.6 shows that both univariate and bivariate spline terms are significant.

Output 49.3.6 Tests for Smoothing Components

Tests for Smoothing Components				
Component	Effective DF	Effective DF for Test	F Value	Pr > F
Spline(Depth)	3.71286	4	32.88	<.0001
Spline(Longitude Latitude)	32.16982	37	1260.67	<.0001

The smoothing component panel in Output 49.3.7 contains a fitted surface and a curve for the two spline terms. Compared to the Poisson model, the fitted surface and curves are smoother. It is possible that the nonlinear dependency structure becomes smoother after its partial variations are accounted for by the dispersion parameter.

Output 49.3.7 Smoothing Components Panel

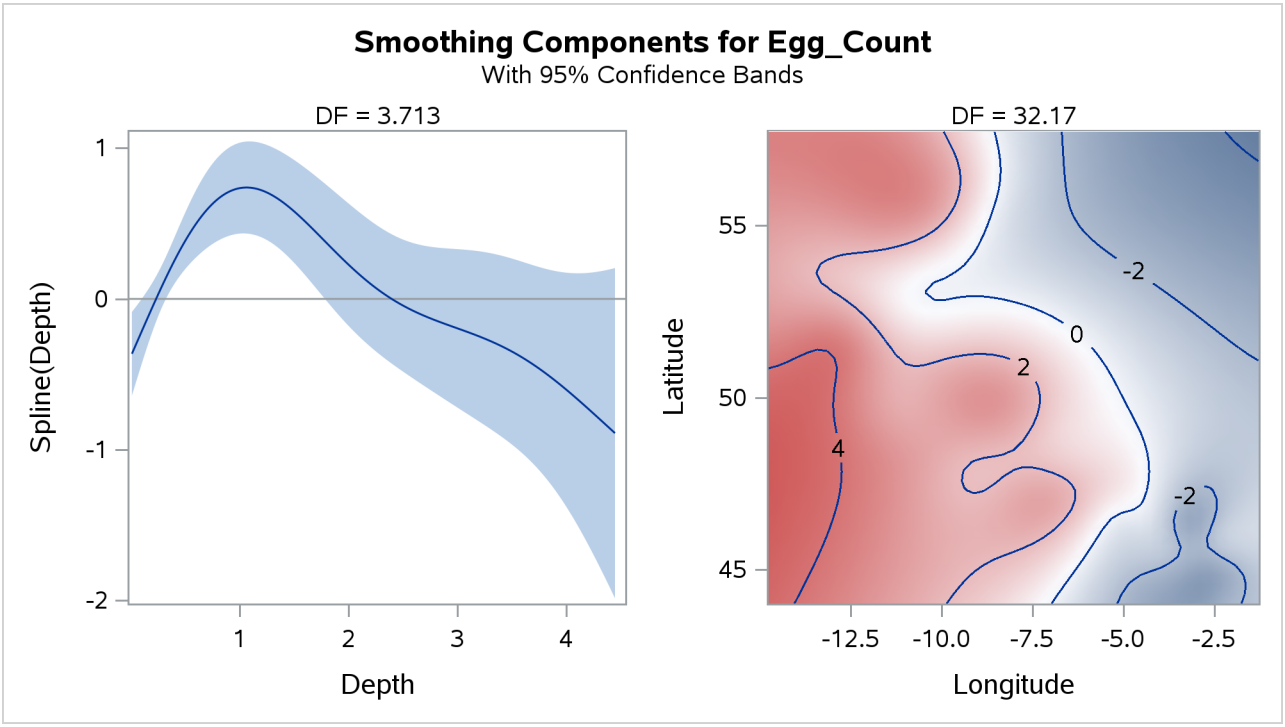

Pearson's chi-square divided by the error degrees of freedom for the negative binomial model is approximately 1.5, which is close to 1. This suggests that the negative binomial model explains the overdispersion in the data.

The following DATA step and statements visualize the contours of mackerel egg density for both Poisson and negative binomial models:

```
data SurfaceFit;
   set Mackerel(keep=Longitude Latitude Net_Area Egg_Count);
   set MackerelOut1(keep=Pred1);
   set MackerelOut2(keep=Pred2);
   Pred1 = Pred1 / Net_Area;
   Pred2 = Pred2 / Net_Area;
run;

%let eopt = location=outside valign=top textattrs=graphlabeltext;
proc template;
   define statgraph surfaces;
      begingraph / designheight=360px;
         layout lattice / columns=2;
            layout overlay / &off0;
               entry "Poisson Model" / &eopt;
               contourplotparm z=Pred1 y=Latitude x=Longitude /
                               gridded=FALSE;
            endlayout;
            layout overlay / &off0;
               entry "Negative Binomial Model" / &eopt;
               contourplotparm z=Pred2 y=Latitude x=Longitude /
```

```
                                    gridded=FALSE;
                 endlayout;
              endlayout;
           endgraph;
       end;
   run;

   proc sgrender data=SurfaceFit template=surfaces;
   run;
```

Output 49.3.8 displays two fitted surfaces from the Poisson model and the negative binomial model, respectively.

Output 49.3.8 Fitted Surfaces from Poisson Model and Negative Binomial Model

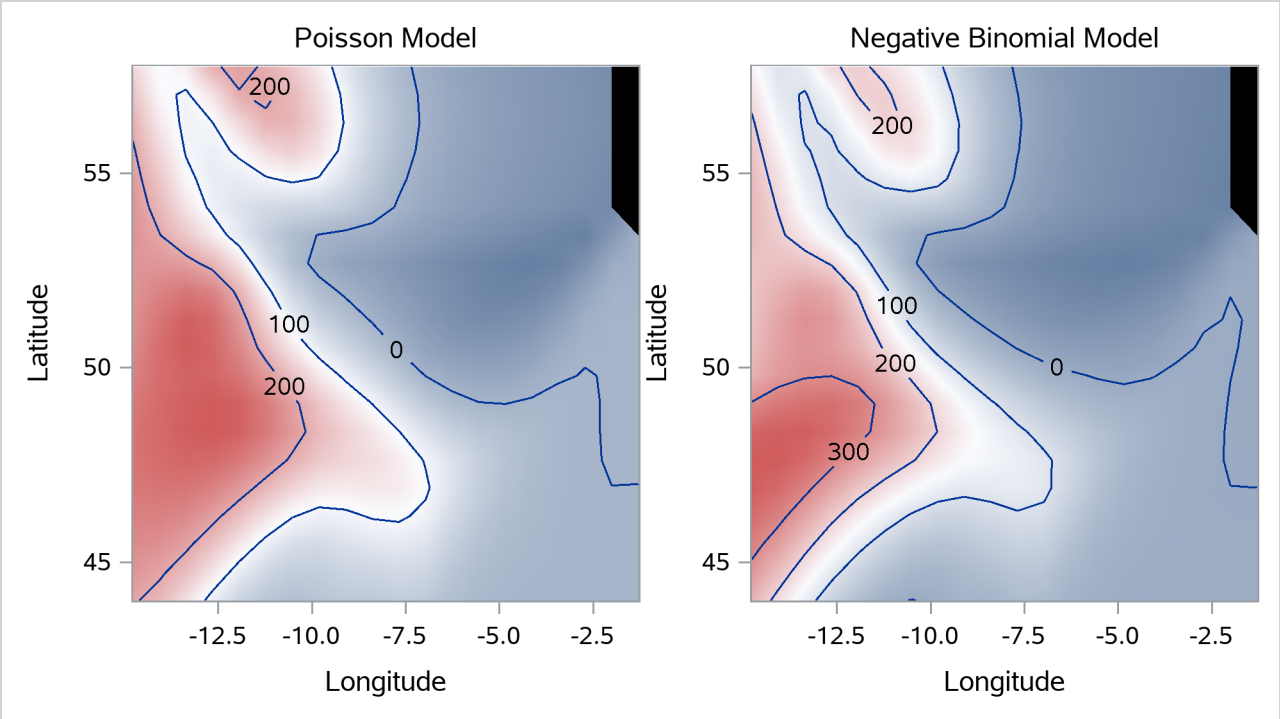

Example 49.4: Nonparametric Tweedie Regression

(View the complete code for this example (hpgamex4.sas) in the example repository.)

This example illustrates how you can use the GAMPL procedure to fit a nonparametric regression model with spline effects for a response variable that has a Tweedie distribution.

You can view this model as a generalized linear model because the Tweedie distribution belongs to the exponential family. The Tweedie distribution is particularly useful for modeling response variables that are continuous for positive values and take the value 0 with a positive probability. For example, in the insurance industry the Tweedie distribution is often assumed for claims, which are positive for customers who have filed claims and are 0 for other customers. You can use the Tweedie distribution to model such response variables

without special transformations. For more information about this distribution, see the section "Tweedie Distribution for Generalized Linear Models" on page 3676.

The GENMOD procedure fits Tweedie regression models that incorporate linear effects for the covariates as illustrated in "Example 51.12: Tweedie Regression" on page 3768. However, as demonstrated in this example, the assumption of linearity can be too restrictive, and you might want to use the GAMPL procedure to explore the nonlinear dependency structures in the data.

The following DATA step simulates a response variable Y by using a Tweedie distribution and four continuous predictors. Each continuous predictor is sampled independently from the uniform distribution $U(0, 1)$. The linear predictor η for the true model is formed by applying additive nonlinear transformations to x_1, x_2, and x_3:

$$\eta = \frac{2\sin(\pi x_1) + 0.8\exp(2x_2) + 0.2x_3^{11}(10(1-x_3))^6 + 10(10x_3)^3(1-x_3)^{10}}{20}$$

The predictor x_4 is a nuisance parameter that does not enter the model. The parameters for the Tweedie distribution are $\phi = 0.4$ and $p = 1.5$, and the link function is $\log\mu = \eta$, where the expected value is $\mathrm{E}(Y) = \mu$ and the variance is $\mathrm{Var}(Y) = \phi\mu^p$. The power parameter p controls the variance of the distribution. When p is between 1 and 2, as in this example, a Tweedie random variable can be generated from a compound Poisson distribution (Smyth 1996).

```
title 'Nonparametric Tweedie Model';
%let phi=0.4;
%let power=1.5;

data one;
   do i=1 to 1000;

      /* Sample the predictors */
      x1=ranuni(1);
      x2=ranuni(1);
      x3=ranuni(1);
      x4=ranuni(1);

      /* Apply nonlinear transformations to predictors */
      f1=2*sin(3.14159265*x1);
      f2=exp(2*x2)*0.8;
      f3=0.2*x3**11*(10*(1-x3))**6+10*(10*x3)**3*(1-x3)**10;
      xb=f1+f2+f3;
      xb=xb/20;
      mu=exp(xb);

      /* Compute parameters of compound Poisson distribution */
      lambda=mu**(2-&power)/(&phi*(2-&power));
      alpha=(2-&power)/(&power-1);
      gamma=&phi*(&power-1)*(mu**(&power-1));

      /* Simulate the response */
      rpoi=ranpoi(1,lambda);
      if rpoi=0 then y=0;
      else do;
         y=0;
         do j=1 to rpoi;
```

```
            y=y+rangam(1,alpha);
         end;
         y=y*gamma;
      end;
      output;
   end;
run;
```

You can use PROC GENMOD to fit a parametric model for Y by using the following statements:

```
proc genmod data=one;
   model y=x1 x2 x3 x4/dist=tweedie;
run;
```

Equivalently, you can also use PROC GAMPL to fit a parametric linear model by using the PARAM option in the MODEL statement, as shown in the following statements.

```
proc gampl data=one seed=1234;
   model y=param(x1 x2 x3 x4)/dist=tweedie;
run;
```

The "Fit Statistics" table in Output 49.4.1 shows the summary statistics for the fitted parametric Tweedie model.

Output 49.4.1 Fit Statistics

Nonparametric Tweedie Model

The GAMPL Procedure

Fit Statistics	
Penalized Log Likelihood	-1178.69507
Roughness Penalty	0
Effective Degrees of Freedom	7.00000
Effective Degrees of Freedom for Error	993.00000
AIC (smaller is better)	2371.39014
AICC (smaller is better)	2371.50305
BIC (smaller is better)	2405.74443

The "Parameter Estimates" table in Output 49.4.2 shows the estimates for regression parameters in addition to dispersion and power parameters (ϕ and p).

Output 49.4.2 Parameter Estimates

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	0.441872	0.067335	43.0637	<.0001
x1	1	-0.043723	0.064966	0.4530	0.5009
x2	1	0.202190	0.064216	9.9135	0.0016
x3	1	-0.253928	0.066626	14.5257	0.0001
x4	1	-0.061118	0.066040	0.8565	0.3547
Dispersion	1	0.416005	0.016987		
Power	1	1.524591	0.048676		

If you are uncertain about the relationship between the linked mean for the fitted Tweedie model and the covariates, you can add spline terms to the model to explore possible nonlinear dependence. The following statements fit a nonparametric Tweedie model and produce a plot of fitted smoothing components.

```
proc gampl data=one seed=1234 plots;
   model y=spline(x1) spline(x2) spline(x3) spline(x4)/dist=tweedie;
run;
```

The "Fit Statistics" table in Output 49.4.3 shows the summary statistics for the fitted nonparametric Tweedie model. Compared to the parametric model, the nonparametric model has more effective degrees of freedom, but smaller AIC and AICC values. The penalized likelihood value is larger for the nonparametric model. These statistics indicate that the nonparametric model provides a better fit.

Output 49.4.3 Fit Statistics

Nonparametric Tweedie Model

The GAMPL Procedure

Fit Statistics	
Penalized Log Likelihood	-1157.17502
Roughness Penalty	2.62230
Effective Degrees of Freedom	14.11920
Effective Degrees of Freedom for Error	983.62098
AIC (smaller is better)	2339.96615
AICC (smaller is better)	2340.39965
BIC (smaller is better)	2409.25974
GCV (smaller is better)	0.39845

The "Regression Parameter Estimates" table in Output 49.4.4 lists the estimate for the intercept, in addition to the estimates for the dispersion and power parameters for the Tweedie model. Both variance parameters are close to the true values.

Output 49.4.4 Regression Parameter Estimates

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	0.356825	0.018329	379.0079	<.0001
Dispersion	1	0.400459	0.518084		
Power	1	1.501771	1.514450		

The "Estimates for Smoothing Components" table in Output 49.4.5 shows the fitted information for the spline terms in the model. The degrees of freedom value and the roughness penalty value suggest some moderate nonlinear relationship between the linked mean and the three covariates x_1, x_2, and x_3. The spline term for the nuisance parameter x_4 has a linear form.

Output 49.4.5 Estimates for Smoothing Components

Estimates for Smoothing Components						
Component	Effective DF	Smoothing Parameter	Roughness Penalty	Number of Parameters	Rank of Penalty Matrix	Number of Knots
Spline(x1)	2.10969	0.9560	0.5659	9	10	1000
Spline(x2)	2.38340	0.5604	0.3814	9	10	1000
Spline(x3)	5.62608	0.0145	1.6750	9	10	1000
Spline(x4)	1.00002	133078	3.662E-6	9	10	1000

The "Tests for Smoothing Components" table in Output 49.4.6 shows the approximate Wald test for the four spline terms. The spline term for the nuisance predictor x_4 is not significant.

Output 49.4.6 Tests for Smoothing Components

Tests for Smoothing Components				
Component	Effective DF	Effective DF for Test	F Value	Pr > F
Spline(x1)	2.10969	3	4.76	0.0027
Spline(x2)	2.38340	3	13.64	<.0001
Spline(x3)	5.62608	7	49.02	<.0001
Spline(x4)	1.00002	1	0.88	0.3484

Output 49.4.7 displays the "Smoothing Component Panel" for all the spline terms used in the model. For x_1, x_2, and x_3, the fitted curves are reasonably smooth and close to the true functions. For x_4, the spline plot shows a strict a linear fit, with 95% Bayesian confidence bands covering the horizontal line at 0. This reinforces the conclusion that x_4 does not contribute to the model and can be removed from further analysis.

Output 49.4.7 Smoothing Component Panel

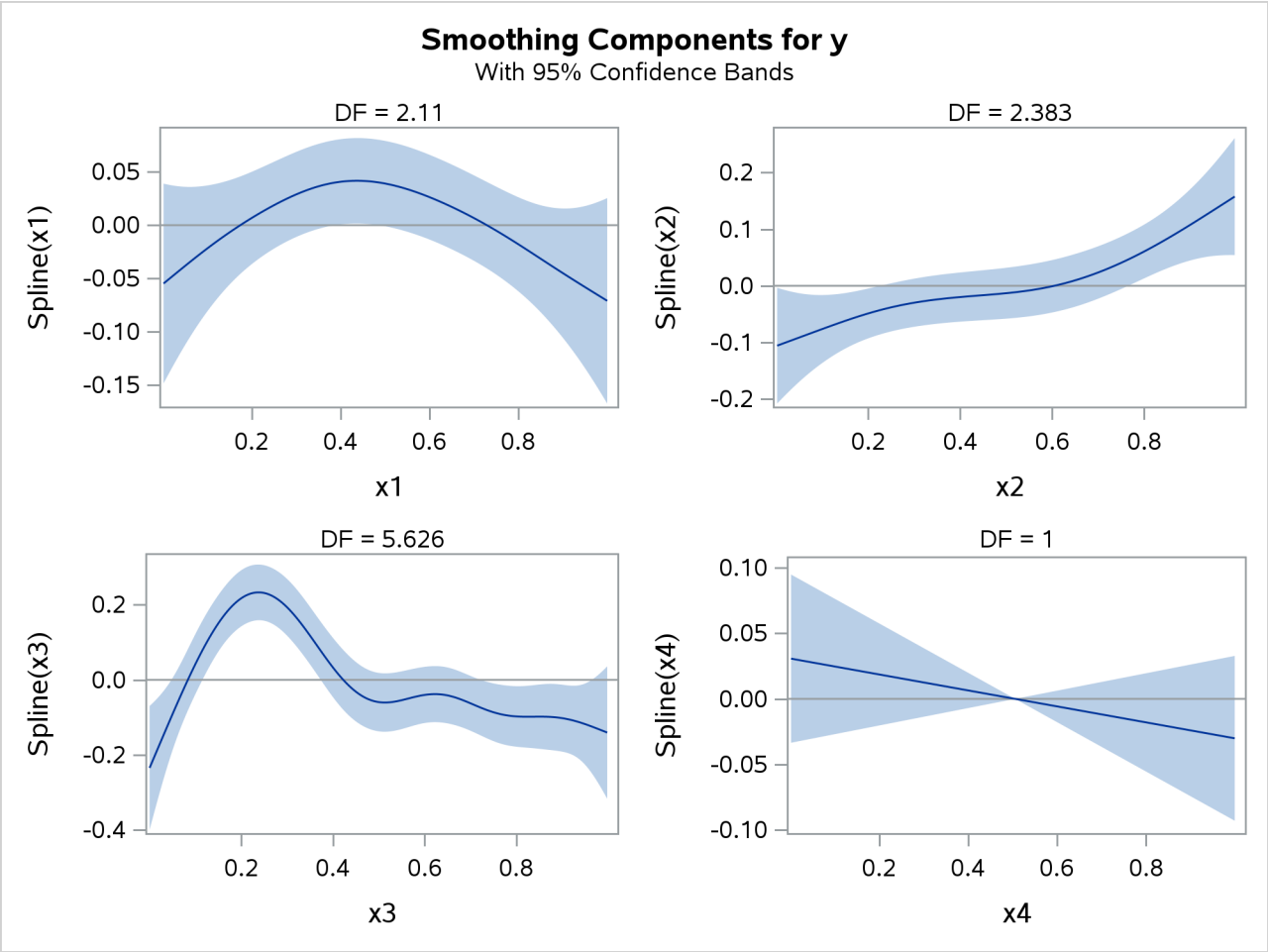

References

Bowman, A. W., and Azzalini, A. (1997). *Applied Smoothing Techniques for Data Analysis*. New York: Oxford University Press.

Craven, P., and Wahba, G. (1979). "Smoothing Noisy Data with Spline Functions." *Numerical Mathematics* 31:377–403.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981). "An Adaptive Nonlinear Least-Squares Algorithm." *ACM Transactions on Mathematical Software* 7:348–368.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Duchon, J. (1976). "Fonctions-spline et espérances conditionnelles de champs gaussiens [Spline functions and conditional expectations of Gaussian fields]." *Annales scientifiques de l'Université de Clermont-Ferrand 2, Série Mathématique* 14:19–27.

Duchon, J. (1977). "Splines Minimizing Rotation-Invariant Semi-norms in Sobolev Spaces." In *Constructive Theory of Functions of Several Variables*, edited by W. Schempp and K. Zeller, 85–100. New York: Springer-Verlag.

Eskow, E., and Schnabel, R. B. (1991). "Algorithm 695: Software for a New Modified Cholesky Factorization." *ACM Transactions on Mathematical Software* 17:306–312.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Gu, C., and Wahba, G. (1991). "Minimizing GCV/GML Scores with Multiple Smoothing Parameters via the Newton Method." *SIAM Journal on Scientific Computing* 12:383–398.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Nychka, D. (1988). "Bayesian Confidence Intervals for Smoothing Splines." *Journal of the American Statistical Association* 83:1134–1143.

Pace, R. K., and Barry, R. (1997). "Quick Computation of Spatial Autoregressive Estimators." *Geographical Analysis* 29:232–247.

Ruppert, D., Wand, M. P., and Carroll, R. J. (2003). *Semiparametric Regression*. Cambridge: Cambridge University Press.

Smith, J. W., Everhart, J. E., Dickson, W. C., Knowler, W. C., and Johannes, R. S. (1988). "Using the ADAP Learning Algorithm to Forecast the Onset of Diabetes Mellitus." In *Proceedings of the Symposium on Computer Applications and Medical Care*, 261–265. Los Alamitos, CA: IEEE Computer Society Press.

Smyth, G. K. (1996). "Regression Analysis of Quantity Data with Exact Zeros." In *Proceedings of the Second Australia-Japan Workshop on Stochastic Models in Engineering, Technology, and Management*, edited by R. J. Wilson, S. Osaki, and D. N. P. Murthy, 572–580. Queensland, Australia: Technology Management Centre, University of Queensland.

Vlachos, P. (1998). "StatLib—Datasets Archive." `http://lib.stat.cmu.edu/datasets/`.

Wahba, G. (1983). "Bayesian 'Confidence Intervals' for the Cross Validated Smoothing Spline." *Journal of the Royal Statistical Society, Series B* 45:133–150.

Wood, S. (2003). "Thin Plate Regression Splines." *Journal of the Royal Statistical Society, Series B* 65:95–114.

Wood, S. (2004). "Stable and Efficient Multiple Smoothing Parameter Estimation for Generalized Additive Models." *Journal of the American Statistical Association* 99:673–686.

Wood, S. (2006). *Generalized Additive Models*. Boca Raton, FL: Chapman & Hall/CRC.

Wood, S. (2008). "Fast Stable Direct Fitting and Smoothness Selection for Generalized Additive Models." *Journal of the Royal Statistical Society, Series B* 70:495–518.

Wood, S. (2011). "Fast Stable Restricted Maximum Likelihood and Marginal Likelihood Estimation of Semiparametric Generalized Linear Models." *Journal of the Royal Statistical Society, Series B* 73:3–36.

Wood, S. (2012). "On *p*-Values for Smooth Components of an Extended Generalized Additive Model." *Biometrika* 1–8. http://biomet.oxfordjournals.org/content/early/2012/10/18/biomet.ass048.abstract.

Xiang, D., and Wahba, G. (1996). "A Generalized Approximate Cross Validation for Smoothing Splines with Non-Gaussian Data." *Statistica Sinica* 6:675–692.

Chapter 50
The GEE Procedure

Contents

Overview: GEE Procedure **3506**
Getting Started **3507**
Syntax: GEE Procedure **3510**
PROC GEE Statement 3511
BY Statement 3512
CLASS Statement 3512
EFFECTPLOT Statement 3513
ESTIMATE Statement 3514
FREQ Statement 3516
LSMEANS Statement 3516
LSMESTIMATE Statement 3517
MARGINS Statement 3519
MISSMODEL Statement 3520
MODEL Statement 3520
OUTPUT Statement 3524
REPEATED Statement 3525
SLICE Statement 3529
STORE Statement 3529
WEIGHT Statement 3529
Details: GEE Procedure **3530**
Generalized Estimating Equations 3530
Alternating Logistic Regression 3534
Weighted Generalized Estimating Equations under the MAR Assumption 3538
Type 3 Analysis 3541
ODS Table Names 3542
ODS Graphics 3544
Examples: GEE Procedure **3544**
Example 50.1: Comparison of the Marginal and Random Effect Models for Binary Data 3545
Example 50.2: Log-Linear Model for Count Data 3548
Example 50.3: Weighted GEE for Longitudinal Data That Have Missing Values . . . 3552
Example 50.4: GEE for Binary Data with Logit Link Function 3556
Example 50.5: Alternating Logistic Regression for Ordinal Multinomial Data 3559
Example 50.6: GEE for Nominal Multinomial Data 3562
References **3564**

Overview: GEE Procedure

The GEE procedure implements the generalized estimating equations (GEE) approach (Liang and Zeger 1986), which extends the generalized linear model to handle longitudinal data (Stokes, Davis, and Koch 2012; Fitzmaurice, Laird, and Ware 2011; Diggle et al. 2002). For longitudinal studies, missing data are common, and they can be caused by dropouts or skipped visits. If missing responses depend on previous responses, the usual GEE approach can lead to biased estimates. So the GEE procedure also implements the weighted GEE method to handle missing responses that are caused by dropouts in longitudinal studies (Robins and Rotnitzky 1995; Preisser, Lohman, and Rathouz 2002). The GEE procedure in does not support the weighted GEE method for the multinomial distribution for polytomous responses.

The GEE method fits a marginal model to longitudinal data. The regression parameters in the marginal model are interpreted as population-averaged. For more information about the GEE method, see Fitzmaurice, Laird, and Ware (2011); Hardin and Hilbe (2003); Diggle et al. (2002); Lipsitz et al. (1994).

The GEE procedure compares most closely to the GENMOD procedure in SAS/STAT software. Both procedures implement the standard generalized estimating equation approach for longitudinal data; this approach is appropriate for complete data or when data are missing completely at random (MCAR). When the data are missing at random (MAR), the weighted GEE method produces valid inference. Molenberghs and Kenward (2007); Fitzmaurice, Laird, and Ware (2011); Mallinckrodt (2013); O'Kelly and Ratitch (2014) describe the weighted GEE method.

The GEE procedure includes alternating logistic regression (ALR) analysis for binary and ordinal multinomial responses. In ordinary GEEs, the association between pairs of responses are modeled with correlations. The ALR approach provides an alternative by using the log odds ratio to model the association between pairs. For more information about the log odds ratio and the ALR method, see the section "Alternating Logistic Regression" on page 3534. For binary responses the ALR algorithm of Carey, Zeger, and Diggle (1993) is implemented in both the GEE and GENMOD procedures. The GEE procedure also implements the ALR algorithm of Heagerty and Zeger (1996), which extends the ALR approach to ordinal multinomial responses. An ordinary GEE with the independent working correlation structure is also available for both nominal and ordinal multinomial data.

Getting Started

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/geegs1.sas.)

This section illustrates some of the basic features of the GEE procedure by analyzing longitudinal data from Stokes, Davis, and Koch (2012).

In this study, researchers followed 25 children at ages 8, 9, 10, and 11 years. The goal of this study is to investigate the health effects of air pollution on children. The binary response is the wheezing status of the children at four different ages. The explanatory variables are age, city, and passive smoking index (with values 0, 1, 2) that represented the degree of smoking in the home. The responses for individual children are assumed to be equally correlated, implying an exchangeable correlation structure.

The following statements create the data set Children:

```
data Children;
   input ID City$ @@;
   do i=1 to 4;
      input Age Smoke Symptom @@;
      output;
   end;
   datalines;
 1 steelcity  8 0 1  9 0 1  10 0 1  11 0 0
 2 steelcity  8 2 1  9 2 1  10 2 1  11 1 0
 3 steelcity  8 2 1  9 2 0  10 1 0  11 0 0
 4 greenhills 8 0 0  9 1 1  10 1 1  11 0 0
 5 steelcity  8 0 0  9 1 0  10 1 0  11 1 0
 6 greenhills 8 0 1  9 0 0  10 0 0  11 0 1
 7 steelcity  8 1 1  9 1 1  10 0 1  11 0 0
 8 greenhills 8 1 0  9 1 0  10 1 0  11 2 0
 9 greenhills 8 2 1  9 2 0  10 1 1  11 1 0
10 steelcity  8 0 0  9 0 0  10 0 0  11 1 0
11 steelcity  8 1 1  9 0 0  10 0 0  11 0 1
12 greenhills 8 0 0  9 0 0  10 0 0  11 0 0
13 steelcity  8 2 1  9 2 1  10 1 0  11 0 1
14 greenhills 8 0 1  9 0 1  10 0 0  11 0 0
15 steelcity  8 2 0  9 0 0  10 0 0  11 2 1
16 greenhills 8 1 0  9 1 0  10 0 0  11 1 0
17 greenhills 8 0 0  9 0 1  10 0 1  11 1 1
18 steelcity  8 1 1  9 2 1  10 0 0  11 1 0
19 steelcity  8 2 1  9 1 0  10 0 1  11 0 0
20 greenhills 8 0 0  9 0 1  10 0 1  11 0 0
21 steelcity  8 1 0  9 1 0  10 1 0  11 2 1
22 greenhills 8 0 1  9 0 1  10 0 0  11 0 0
23 steelcity  8 1 1  9 1 0  10 0 1  11 0 0
24 greenhills 8 1 0  9 1 1  10 1 1  11 2 1
25 greenhills 8 0 1  9 0 0  10 0 0  11 0 0
;
```

The following statements fit the model by the GEE method:

```
proc gee data=Children descending;
   class ID City;
   model Symptom = City Age Smoke / dist=bin link=logit;
   repeated subject=ID / type=exch covb corrw;
run;
```

Both the MODEL statement and the REPEATED statement are required.

The DIST=BIN and LINK=LOGIT options in the MODEL statement request a logistic regression with the variable Symptom as the response and City, Age, and Smoke as explanatory variables.

The REPEATED statement specifies the correlation structure and requests various tables in the output. The SUBJECT=ID option requests that individual subjects be identified in the input data set by the variable ID, which must be listed in the CLASS statement. Measurements of individual subjects at ages 8, 9, 10, and 11 are in the proper order in the data set, so the WITHIN= option is not required. The TYPE=EXCH option specifies an exchangeable working correlation structure, the COVB option requests the parameter estimate covariance matrix, and the CORRW option requests the working correlation matrix.

Figure 50.1 shows the "Model Information" table, which provides information about the specified logistic regression model and the input data set.

Figure 50.1 Model Information

The GEE Procedure

Model Information	
Data Set	WORK.CHILDREN
Distribution	Binomial
Link Function	Logit
Dependent Variable	Symptom

Figure 50.2 displays general information about the GEE analysis. Each subject has four measurements.

Figure 50.2 GEE Model Information

GEE Model Information	
Correlation Structure	Exchangeable
Subject Effect	ID (25 levels)
Number of Clusters	25
Correlation Matrix Dimension	4
Maximum Cluster Size	4
Minimum Cluster Size	4

Figure 50.3 displays the model-based and empirical covariance matrices of the parameter estimates.

Figure 50.3 Covariance Matrices of Parameter Estimates

Covariance Matrix (Model-Based)				
	Prm1	**Prm2**	**Prm4**	**Prm5**
Prm1	3.26069	-0.16313	-0.32274	-0.12257
Prm2	-0.16313	0.24015	0.002520	0.03422
Prm4	-0.32274	0.002520	0.03379	0.004471
Prm5	-0.12257	0.03422	0.004471	0.09533

Covariance Matrix (Empirical)				
	Prm1	**Prm2**	**Prm4**	**Prm5**
Prm1	4.09770	-0.55261	-0.37280	-0.29397
Prm2	-0.55261	0.29538	0.03719	0.09143
Prm4	-0.37280	0.03719	0.03550	0.02064
Prm5	-0.29397	0.09143	0.02064	0.07957

The exchangeable working correlation matrix is displayed in Figure 50.4.

Figure 50.4 Working Correlation Matrix

Working Correlation Matrix				
	Obs 1	**Obs 2**	**Obs 3**	**Obs 4**
Obs 1	1.0000	0.0883	0.0883	0.0883
Obs 2	0.0883	1.0000	0.0883	0.0883
Obs 3	0.0883	0.0883	1.0000	0.0883
Obs 4	0.0883	0.0883	0.0883	1.0000

The parameter estimates table, shown in Figure 50.5, contains parameter estimates, standard errors, confidence intervals, Z scores, and p-values for the parameter estimates. Empirical standard error estimates are used in this table. You can create a table that uses model-based standard errors by specifying the MODELSE option in the REPEATED statement. The results indicate that smoking exposure is significant with a p-value of 0.0211, Age is marginally influential with a p-value of 0.0893, and City does not influence wheezing. The parameter estimate for Age is –0.3201, which indicates that the odds ratio of wheezing for the children at the higher age group compared to those in the lower age group is $e^{-0.3201} = 0.726$.

Figure 50.5 GEE Parameter Estimates Table

Parameter Estimates for Response Model with Empirical Standard Error Estimates							
Parameter		**Estimate**	**Standard Error**	**95% Confidence Limits**		**Z**	**Pr > |Z|**
Intercept		2.2615	2.0243	-1.7060	6.2290	1.12	0.2639
City	greenhil	0.0418	0.5435	-1.0234	1.1070	0.08	0.9387
City	steelcit	0.0000	0.0000	0.0000	0.0000	.	.
Age		-0.3201	0.1884	-0.6894	0.0492	-1.70	0.0893
Smoke		0.6506	0.2821	0.0978	1.2035	2.31	0.0211

Goodness-of-fit criteria for the model are displayed in Figure 50.6. For more information about the quasi-likelihood information criterion (QIC), see the section "Quasi-likelihood Information Criterion" on page 3532.

Figure 50.6 Model Fit Criteria

GEE Fit Criteria	
QIC	137.1373
QICu	136.2173

Syntax: GEE Procedure

The following statements are available in the GEE procedure. Items within < > are optional.

```
PROC GEE <options> ;
    BY variables ;
    CLASS variable <(options)> ... <variable <(options)>> </ options> ;
    EFFECTPLOT <plot-type <(plot-definition-options)>> </ options> ;
    ESTIMATE <'label'> estimate-specification </ options> ;
    FREQ | FREQUENCY variable ;
    LSMEANS <model-effects> </ options> ;
    LSMESTIMATE model-effect <'label'> values <divisor=n> <, ... <'label'> values <divisor=n>>
                    </ options> ;
    MARGINS model-effects </ options> ;
    MISSMODEL <effects> </ options> ;
    MODEL response = <effects > </ options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=name ... keyword=name> ;
    REPEATED SUBJECT=subject-effect </ options> ;
    SLICE model-effect </ options> ;
    STORE <OUT= >item-store-name </ LABEL='label'> ;
    WEIGHT variable ;
```

The syntax of the GEE procedure compares most closely to that of the GENMOD procedures. The PROC GEE, MODEL, and REPEATED statements are required. All other statements can appear only once. The following sections describe the PROC GEE statement and then describe the other statements in alphabetical order.

PROC GEE Statement

PROC GEE < *options* > **;**

The PROC GEE statement invokes the GEE procedure. Table 50.1 summarizes the *options* available in the PROC GEE statement.

Table 50.1 PROC GEE Statement Options

Option	Description
DATA=	Specifies the input data set
DESCENDING	Sorts the response variable in the reverse of the default order
NAMELEN=	Specifies the length of effect names
ORDER=	Specifies the sort order of CLASS variable
PLOTS	Controls the plots that are produced through ODS Graphics

You can specify the following *options*.

DATA=*SAS-data-set*
specifies the SAS data set that contains the data to be analyzed. If you omit the DATA= option, PROC GEE uses the most recently created SAS data set.

DESCENDING
DESCEND
DESC
requests that the levels of the response variable for the binomial model that uses a single-variable response syntax be sorted in the reverse of the default order.

NAMELEN=*number*
specifies the length to which long effect names are shortened. The default and minimum value is 20.

PLOTS < **=** *plot-request* >
controls the plots produced through ODS Graphics. For example:

```
proc gee plots=histogram;
   model y=x1;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *plot-requests*:

ALL
requests that all default plots be produced.

HISTOGRAM
creates a histogram for the predicted weights from the missingness model.

NONE
suppresses all plots.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC GEE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the GEE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / *options* > **;**

The CLASS statement names the classification *variables* to be used in the analysis. If the CLASS statement is used, it must appear before the MODEL statement.

Classification variables can be either character or numeric. CLASS levels are determined from the formatted values of the *variables*. Thus, you can use formats to group values into levels. For more information, see the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

You can specify the following *options* for classification *variables*:

DESCENDING
DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC GEE orders the categories according to the ORDER= option and then reverses that order.

ORDER=*order-type*

specifies the sort order for the categories of categorical variables. This ordering determines which parameters in the model correspond to each level in the data. When the default ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. Table 50.2 shows how PROC GEE interprets values of the ORDER= option.

Table 50.2 Sort Order for Categorical Variables

order-type	**Levels Sorted By**
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels that have the most observations come first in the order
INTERNAL	Unformatted value

For the FORMATTED and INTERNAL values, the sort order is machine-dependent. If you specify the ORDER= option in the MODEL statement and the ORDER= option in the CLASS statement, the former takes precedence.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

EFFECTPLOT Statement

EFFECTPLOT < *plot-type* < *(plot-definition-options)* > > < / *options* > **;**

The EFFECTPLOT statement produces a display of the fitted model and provides options for changing and enhancing the displays. Table 50.3 describes the available *plot-types* and their *plot-definition-options*.

Table 50.3 *Plot-Types* and *Plot-Definition-Options*

Plot-Type and Description	**Plot-Definition-Options**
BOX Displays a box plot of continuous response data at each level of a CLASS effect, with predicted values superimposed and connected by a line. This is an alternative to the INTERACTION *plot-type*.	PLOTBY= variable or CLASS effect X= CLASS variable or effect

Table 50.3 *continued*

Plot-Type and Description	Plot-Definition-Options
CONTOUR Displays a contour plot of predicted values against two continuous covariates.	PLOTBY= variable or CLASS effect X= continuous variable Y= continuous variable
FIT Displays a curve of predicted values versus a continuous variable.	PLOTBY= variable or CLASS effect X= continuous variable
INTERACTION Displays a plot of predicted values (possibly with error bars) versus the levels of a CLASS effect. The predicted values are connected with lines and can be grouped by the levels of another CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= CLASS variable or effect
MOSAIC Displays a mosaic plot of predicted values using up to three CLASS effects.	PLOTBY= variable or CLASS effect X= CLASS effects
SLICEFIT Displays a curve of predicted values versus a continuous variable grouped by the levels of a CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= continuous variable

For full details about the syntax and options of the EFFECTPLOT statement, see the section "EFFECTPLOT Statement" on page 447 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 50.4 summarizes the *options* available in the ESTIMATE statement.

Table 50.4 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the **L** matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement" on page 475 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

FREQUENCY *variable* **;**

The *variable* in the FREQ statement identifies a variable in the input data set that contains the frequency of occurrence of each observation. PROC GEE treats each observation as if it appeared n times, where n is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If it is less than 1 or missing, the observation is not used. The frequencies must be the same for all observations within each subject.

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 50.5 summarizes the *options* available in the LSMEANS statement.

Table 50.5 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion

Table 50.5 *continued*

Option	Description
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 50.6 summarizes the *options* available in the LSMESTIMATE statement.

Table 50.6 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MARGINS Statement

MARGINS *model-effects* < / *options* > **;**

The MARGINS statement computes and compares predictive margins of fixed effects. The predictive margin for a specific level (group) of a fixed effect represents the average predicted response if all the observations in the data set were in that group (Lane and Nelder 1982; Chang, Gelman, and Pagano 1982). You can compute predictive margins for any effect in the MODEL statement that involves only classification variables.

Table 50.7 summarizes the *options* available in the MARGINS statement.

Table 50.7 MARGINS Statement Options

Option	Description
Construction and Computation of Predictive Margins	
AT	Modifies the covariate value in computing predictive margins
DIFF	Computes differences of predictive margins
SLICEBY=	Partitions tests of interaction effects
SLICEDIFF	Computes differences of sliced predictive margins and determines the types of differences
USEWEIGHT	Specifies whether to use WEIGHT variables to compute predictive margins
Degrees of Freedom and *p*-Values	
ADJUST=	Specifies the method of multiple comparison adjustment of predictive margin differences
ALPHA=α	Specifies the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion
Statistical Output	
CL	Constructs confidence limits for predictive margins and/or predictive margin differences

For more information about the syntax of the MARGINS statement, see the section "MARGINS Statement" on page 525 in Chapter 20, "Shared Concepts and Topics."

NOTE: By default, chi-square and z tests are performed, and weights are not used in the computations. The MARGINS statement cannot be used with weighted GEE models.

MISSMODEL Statement

MISSMODEL *effects* < / *options* > **;**

The MISSMODEL statement requests a weighted GEE analysis. It specifies a logistic regression that is used to estimate the weights under the MAR assumption. If the pattern of missing data is intermittent (not dropout), the GEE procedure terminates and does not perform an analysis.

You can use the same effects or different effects in the MODEL and MISSMODEL statements. Explanatory variables can be continuous or classification variables. Classification variables can be character or numeric. Explanatory variables that represent nominal (classification) data must be declared in a CLASS statement. Interactions between variables can also be included as effects. Columns of the design matrix are automatically generated for classification variables and interactions. The syntax for effects is the same as for the GLM procedure. For more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

You can specify the following *options* after a slash (/).

MAXWEIGHT=*number*
truncates the predicted weights from the missingness model if they are larger than *number*, where *number* ≥ 1.

TYPE=OBSLEVEL | SUBLEVEL
specifies the type of weighted GEE method. You can specify the following values:

OBSLEVEL specifies the observation-level weighted GEE method.

SUBLEVEL specifies the subject-level weighted GEE method.

By default, TYPE=OBSLEVEL.

MODEL Statement

MODEL *response* **=** < *effects* > < / *options* > **;**

MODEL *events/trials* **=** < *effects* > < / *options* > **;**

The MODEL statement specifies the response (dependent variable) and the effects (explanatory variables). If you omit the explanatory variables, PROC GEE fits an intercept-only model. An intercept term is included in the model by default. You can remove the intercept by specifying the NOINT option.

You can specify the response in the form of a single variable (*response*) or in the form of a ratio of two variables (*events/trials*). The first form is applicable to all responses. The second form is applicable only to summarized binomial response data. When each observation in the input data set contains the number of events (for example, successes) and the number of trials from a set of binomial trials, use the *events/trials* syntax.

In the *events/trials* model syntax, you specify two variables: one for the event counts and one for trial counts. These two variables are separated by a slash (/). The value of the *events* variable must be nonnegative, and the value of the *trials* variable must be equal to or greater than the value of the *events* variable for an observation to be valid. The *events* and *trials* variables can take non-integer values.

When each observation in the input data set contains a single trial from a binomial experiment, use the *response* form of the MODEL statement. The response variable can be numeric or character. The ordering of response levels is critical in these models.

Responses for the Poisson distribution must be all nonnegative, but they can be non-integer values.

The *effects* in the MODEL statement consist of an explanatory variable or combination of variables. Explanatory variables can be continuous or classification variables. Classification variables can be character or numeric. Explanatory variables that represent nominal (classification) data must be declared in a CLASS statement. Interactions between variables can also be included as effects. Columns of the design matrix are automatically generated for classification variables and interactions. The syntax for specifying effects is the same as for the GLM procedure. For more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

Table 50.8 summarizes the *options* available in the MODEL statement.

Table 50.8 MODEL Statement Options

Option	Description
ALPHA=	Sets the confidence coefficient
DIST=	Specifies the probability distribution
LINK=	Specifies the link function
NOINT	Requests no intercept term
NOSCALE	Holds the scale parameter fixed
OFFSET=	Specifies a variable in the input data set to be used as an offset
SCALE=	Specifies the value used for the scale
TYPE3	Computes statistics for Type 3 contrasts
WALD	Requests Wald statistics for Type 3 contrasts

You can specify the following *options* after a slash (/).

ALPHA=*number*

sets the confidence coefficient for parameter confidence intervals to 1−*number*. The value of *number* must be between 0 and 1. The default value of *number* is 0.05.

DIST=*keyword*

D=*keyword*

ERROR=*keyword*

ERR=*keyword*

specifies the built-in probability distribution to use in the model. If you specify the DIST= option and you omit the LINK= option, a default link function is chosen as displayed in Table 50.9. If you specify neither the DIST= option nor the LINK= option, then the GEE procedure defaults to the normal distribution with the identity link function.

Table 50.9 Distributions and Default Link Functions

DIST=	Distribution	Default Link Function
BINOMIAL | BIN | B	Binomial	Logit
GAMMA | GAM | G	Gamma	Reciprocal
IGAUSSIAN | IG	Inverse Gaussian	Reciprocal square
MULTINOMIAL | MULT	Multinomial	Cumulative logit
NEGBIN | NB	Negative binomial	Log
NORMAL | NOR | N	Normal	Identity
POISSON | POI | P	Poisson	Log

LINK=*keyword*

specifies the link function in the model. You can specify the *keywords* shown in Table 50.10.

Table 50.10 Built-In Link Functions of the GEE Procedure

LINK=	Link Function	$g(\mu) = \eta =$
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$
CUMCLL | CCLL	Cumulative complementary log-log	$\log(-\log(1-\pi))$
CUMLOGIT| CLOGIT	Cumulative logit	$\log(\pi/(1-\pi))$
CUMPROBIT | CPROBIT	Cumulative probit	$\Phi^{-1}(\pi)$
GLOGIT	Generalized logit	
IDENTITY | ID	Identity	μ
LOG	Log	$\log(\mu)$
LOGIT	Logit	$\log(\mu/(1-\mu))$
PROBIT	Probit	$\Phi^{-1}(\mu)$
INVERSE | RECIPROCAL	Reciprocal	$1/\mu$
POWERMINUS2	Power with exponent –2	$1/\mu^2$

For the probit and cumulative probit links, $\Phi^{-1}(\cdot)$ denotes the quantile function of the standard normal distribution. If you do not specify the LINK= option, then by default the canonical link function is used if you specify the DIST= option. Otherwise, if you omit the DIST= option, the identity link function is used.

The cumulative link functions are appropriate only for the multinomial distribution with ordinal responses, with cumulative probabilities indicated by π. The GLOGIT link function is appropriate only for the multinomial distribution with nominal responses.

NOINT

requests that no intercept term be included in the model. An intercept is included unless this option is specified.

NOSCALE

holds the scale parameter fixed. Otherwise, for the normal, inverse Gaussian, and gamma distributions, the scale parameter is estimated by maximum likelihood. If you omit the SCALE= option, the scale parameter is fixed at the value 1.

OFFSET=*variable*

specifies a variable in the input data set to be used as an offset variable. This variable cannot be a CLASS variable, the response variable, or any of the explanatory variables.

SCALE=*number*

SCALE=PEARSON | P

PSCALE

SCALE=DEVIANCE | D

DSCALE

specifies the value used for the scale parameter when the NOSCALE option is used. For the binomial and Poisson distributions, which have no free scale parameter, this can be used to specify an *overdispersed* model. If the NOSCALE option is not specified, then *number* is used as an initial estimate of the scale parameter.

Specifying SCALE=PEARSON or SCALE=P is the same as specifying the PSCALE option. This fixes the scale parameter at the value 1 in the estimation procedure. After the parameter estimates are determined, the exponential family dispersion parameter is assumed to be given by Pearson's chi-square statistic divided by the degrees of freedom, and all statistics such as standard errors are adjusted appropriately.

Specifying SCALE=DEVIANCE or SCALE=D is the same as specifying the DSCALE option. This fixes the scale parameter at a value of 1 in the estimation procedure.

TYPE3

requests that statistics for Type 3 contrasts be computed for each effect specified in the MODEL statement. The default analysis is to compute score statistics for the contrasts. Type 3 analyses using the score statistics are not supported for nominal response data or weighted GEE methods. Wald statistics are computed if the WALD option is also specified.

WALD

requests Wald statistics for Type 3 contrasts. You must also specify the TYPE3 option in order to compute Type 3 Wald statistics.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*> < *keyword*=*name* . . . *keyword*=*name*> **;**

The OUTPUT statement creates a new SAS data set that contains all the variables in the input data set and, optionally, the estimated linear predictors (XBETA) and their standard error estimates, predicted values of the mean, and confidence limits for predicted values.

If you use the multinomial distribution with one of the cumulative link functions for ordinal data, the data set also contains variables named _ORDER_ and _LEVEL_ that indicate the levels of the ordinal response variable and the values of the variable in the input data set corresponding to the sorted levels. These variables indicate that the predicted value for a given observation is the probability that the response variable is as large as the value of the _LEVEL_ variable. Residuals and other diagnostic statistics are not available for the multinomial distribution.

The estimated linear predictor, its standard error estimate, and the predicted values and their confidence intervals are computed for all observations in which the explanatory variables are all nonmissing, even if the response is missing. By adding observations with missing response values to the input data set, you can compute these statistics for new observations or for settings of the explanatory variables not present in the data without affecting the model fit.

The following list explains specifications in the OUTPUT statement.

OUT=*SAS-data-set*

specifies the output data set. If you omit the OUT=option, the output data set is created and given a default name that uses the DATAn convention.

keyword=name

specifies the statistics to be included in the output data set and names the new variables that contain the statistics. Specify a keyword for each desired statistic (see the following list of keywords), an equal sign, and the name of the new variable or variables to contain the statistic.

Although you can use the OUTPUT statement without any *keyword=name* specifications, the output data set then contains only the original variables and, possibly, the variables Level and Value (if you use the multinomial model with ordinal data).

The *keywords* allowed and the statistics they represent are as follows:

LOWER | L represents the lower confidence limit for the predicted value of the mean, or the lower confidence limit for the probability that the response is less than or equal to the value of Level or Value. The confidence coefficient is determined by the ALPHA=*number* option in the MODEL statement as $(1 - number) \times 100\%$. The default confidence coefficient is 95%.

PREDICTED | PRED | PROB | P represents the predicted value of the mean of the response or the predicted probability that the response variable is less than or equal to the value of _LEVEL_ if the multinomial model for ordinal data is used (in other words, $\Pr(\mathrm{Y} \leq _\mathrm{LEVEL}_)$, where Y is the response variable).

RESCHI represents the Pearson (chi) residual for identifying observations that are poorly accounted for by the model. This option is not available for the multinomial distribution.

RESRAW represents the raw residual for identifying poorly fitted observations. This option is not available for the multinomial distribution.

STDXBETA represents the standard error estimate of XBETA (see the XBETA keyword).

UPPER | U represents the upper confidence limit for the predicted value of the mean, or the upper confidence limit for the probability that the response is less than or equal to the value of Level or Value. The confidence coefficient is determined by the ALPHA=*number* option in the MODEL statement as $(1 - number) \times 100\%$. The default confidence coefficient is 95%.

XBETA represents the estimate of the linear predictor $\mathbf{x}_i'\boldsymbol{\beta}$ for observation i, or $\alpha_j + \mathbf{x}_i'\boldsymbol{\beta}$, where j is the corresponding ordered value of the response variable for the multinomial model with ordinal data. If there is an offset, it is included in $\mathbf{x}_i'\boldsymbol{\beta}$.

REPEATED Statement

REPEATED SUBJECT=*subject-effect* < / *options* > **;**

The REPEATED statement specifies the correlation structure of the responses for GEE model fitting. In addition, the REPEATED statement controls the iterative fitting algorithm and specifies optional output.

Table 50.11 summarizes the *options* available in the REPEATED statement.

Table 50.11 REPEATED Statement Options

Option	Description
ALPHAINIT=	Specifies initial values for log odds ratio regression parameters
CONVERGE=	Specifies the convergence criterion for GEE parameter estimation
CORRB	Displays the estimated correlation matrix
CORRW	Displays the estimated working correlation matrix
COVB	Displays the estimated covariance matrix
ECORRB	Displays the estimated empirical correlation matrix
ECOVB	Displays the estimated empirical covariance matrix
INITIAL=	Specifies initial values of the regression parameters estimation
INTERCEPT=	Specifies an initial value of the intercept
LOGOR=	Specifies the use of alternating logistic regression and a model for the log odds ratio
MAXITER=	Specifies the maximum number of iterations
MCORRB	Displays the estimated model-based correlation matrix
MCOVB	Displays the estimated model-based covariance matrix
MODELSE	Displays a parameter estimates table with the model-based standard errors
SUBCLUSTER=	Specifies a variable that defines subclusters
SUBJECT=	Identifies a different subject (cluster)
TYPE=	Specifies the working correlation matrix structure
WITHIN=	Specifies the order of measurements within subjects
ZDATA=	Specifies the full $\mathbf{z}$ matrix
ZROW=	Specifies the rows of the $\mathbf{z}$ matrix

You must specify the SUBJECT= option:

SUBJECT=*subject-effect*
identifies subjects in the input data set. The *subject-effect* can be a single variable, an interaction effect, a nested effect, or a combination. Each distinct value (level) of the effect identifies a different subject (cluster). Responses from different subjects are assumed to be statistically independent, and responses within subjects are assumed to be correlated. You must specify a *subject-effect*, and you must list variables that are used in defining the *subject-effect* in the CLASS statement.

You can also specify the following *options* after a slash (/) to control how the model is fit and what output is produced:

ALPHAINIT=*numbers*
specifies initial values for log odds ratio regression parameters if you specify the option LOGOR= for data that have either binary or ordinal multinomial responses. The default value of *numbers* is 0.01.

CONVERGE=*number*
specifies the convergence criterion for GEE parameter estimation. If the maximum absolute difference between regression parameter estimates is less than *number* on two successive iterations, convergence is declared. If the absolute value of a regression parameter estimate is greater than 0.08, then the absolute difference normalized by the regression parameter value is used instead of the absolute difference. The default value of *number* is 0.0001.

CORRB
displays the estimated regression parameter correlation matrix. Both model-based and empirical correlations are displayed.

CORRW
displays the estimated working correlation matrix. If you specify TYPE=EXCH for the exchangeable working correlation structure, then the CORRW option is not needed to view the estimated correlation, because a table that contains the single estimated correlation is printed by default.

COVB
displays the estimated regression parameter covariance matrix. Both model-based and empirical covariances are displayed.

ECORRB
displays the estimated regression parameter empirical correlation matrix.

ECOVB
displays the estimated regression parameter empirical covariance matrix.

INITIAL=*numbers*
specifies initial values of the regression parameters estimation, other than the intercept parameter, for GEE estimation. If you do not specify this option, then the estimated regression parameters (assuming independence for all responses) are used for the initial values.

INTERCEPT=*number*

specifies an initial value of the intercept regression parameter in the GEE model.

LOGOR=*log-odds-ratio-structure-keyword*

specifies the use of the alternating logistic regression (ALR) method and the regression model structure for the log odds ratio. For data that have either a binary or ordinal multinomial response distribution, the ALR method uses the log odds ratio to model the association of the responses from subjects. For more information about the ALR method and examples of specifying log odds ratio models, see the section "Alternating Logistic Regression" on page 3534. You can specify the values that are shown in Table 50.12.

Table 50.12 Log Odds Ratio Regression Structures

Keyword	Log Odds Ratio Regression Structure
EXCH	Exchangeable
FULLCLUST	Fully parameterized clusters
LOGORVAR(*variable*)	Indicator variable for specifying block effects
NESTK	*k*-nested
NEST1	1-nested
ZFULL	Fully specified **z** matrix specified in ZDATA= data set
ZREP	Single cluster specification for replicated **z** matrix specified in ZDATA= data set
ZREP(*matrix*)	Single cluster specification for replicated **z** matrix

For ordinal multinomial data, only the exchangeable regression structure that is specified by LOGOR=EXCH is supported. You should specify the option LOGOR= or TYPE=, but not both.

MAXITER=*number*

MAXIT=*number*

specifies the maximum *number* of iterations allowed in the iterative GEE estimation process. By default, MAXITER=50.

MCORRB

displays the estimated regression parameter model-based correlation matrix.

MCOVB

displays the estimated regression parameter model-based covariance matrix.

MODELSE

displays a parameter estimates table that uses model-based standard errors for inference. By default, a "Parameter Estimates" table that is based on empirical standard errors is displayed.

SUBCLUSTER=*variable*

SUBCLUST=*variable*

specifies a *variable* that defines subclusters for the 1-nested or *k*-nested log odds ratio association modeling structures for data that have a binary response distribution. A 1-nested or *k*-nested modeling structure is specified in the option LOGOR=, and *variable* must be listed in the CLASS statement. For definitions of the 1-nested and *k*-nested modeling structures, see the section "Specifying Log Odds Ratio Models" on page 3535.

TYPE=*correlation-structure-keyword*

CORR=*correlation-structure-keyword*

specifies the structure of the working correlation matrix that is used to model the correlation of the responses from subjects for ordinary GEEs. You can specify the values that are shown in Table 50.13 (for definitions of the correlation matrix types, see Table 50.14 in the section "Details: GEE Procedure" on page 3530).

Table 50.13 Correlation Structure Types

Keyword	Correlation Structure Type
AR | **AR**(1)	Autoregressive(1)
EXCH | **CS**	Exchangeable
IND	Independent
MDEP(*number*)	m-dependent, where m = *number*
UNSTR | **UN**	Unstructured
USER(*matrix*) | **FIXED**(*matrix*)	Fixed, user-specified correlation matrix

For example, the following option specifies a fixed 4×4 correlation matrix:

```
type=user( 1.0  0.9  0.8  0.6
           0.9  1.0  0.9  0.8
           0.8  0.9  1.0  0.9
           0.6  0.8  0.9  1.0 )
```

By default, TYPE=IND. When you specify the alternating logistic regression method by using the LOGOR= option, you should not specify the TYPE= option.

WITHINSUBJECT=*within-subject-effect*

WITHIN=*within-subject-effect*

defines an effect that specifies the order of measurements within subjects. Each distinct level of the *within-subject-effect* defines a different response from the same subject. If the data are in proper order within each subject, you can omit this option.

If some measurements do not appear in the data for some subjects, this option properly orders the existing measurements and treats the omitted measurements as missing values.

If you do not specify the WITHIN= option for the standard GEE method, missing values are assumed to be the last values and are not used; the remaining observations are then ordered in the sequence in which they are provided in the input data set. If you do not specify the WITHIN= option for the weighted GEE method, the observations are assumed to be ordered in the sequence in which they are provided in the input data set.

Variables that are used in defining the *within-subject-effect* must be listed in the CLASS statement, and the *within-subject-effect* level ordering depends on the options that you specify in the CLASS statement.

ZDATA=*SAS-data-set*
specifies a SAS data set that contains either the full **z** matrix for log odds ratio association modeling for data with binary responses or the **z** matrix for a single complete cluster to be replicated for all clusters.

ZROW=*variable-list*
specifies the variables in the ZDATA= data set that correspond to rows of the **z** matrix for log odds ratio association modeling for data with binary responses.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement identifies a *variable* in the input data set to be used as the exponential family dispersion parameter weight for each observation. The exponential family dispersion parameter is divided by the WEIGHT variable value for each observation.

The WEIGHT variable value does not have to be an integer; if the value is less than or equal to 0 or if it is missing, the corresponding observation is not used.

Details: GEE Procedure

Generalized Estimating Equations

The marginal model is commonly used in analyzing longitudinal data when the population-averaged effect is of interest. To estimate the regression parameters in the marginal model, Liang and Zeger (1986) proposed the generalized estimating equations method, which is widely used.

Suppose $y_{ij}, j = 1, \ldots, n_i, i = 1, \ldots, K$, represent the *j*th response of the *i*th subject, which has a vector of covariates x_{ij}. There are n_i measurements on subject *i*, and the maximum number of measurements per subject is *T*.

Suppose the responses of the *i*th subject be $\mathbf{Y}_i = [y_{i1}, \ldots, y_{in_i}]'$ with corresponding means $\boldsymbol{\mu}_i = [\mu_{i1}, \ldots, \mu_{in_i}]'$. For generalized linear models, the marginal mean μ_{ij} of the response y_{ij} is related to a linear predictor through a link function $g(\mu_{ij}) = \mathbf{x}'_{ij}\boldsymbol{\beta}$, and the variance of y_{ij} depends on the mean through a variance function $v(\mu_{ij})$.

An estimate of the parameter $\boldsymbol{\beta}$ in the marginal model can be obtained by solving the generalized estimating equations,

$$\mathbf{S}(\boldsymbol{\beta}) = \sum_{i=1}^{K} \frac{\partial \boldsymbol{\mu}'_i}{\partial \boldsymbol{\beta}} \mathbf{V}_i^{-1} (\mathbf{Y}_i - \boldsymbol{\mu}_i(\boldsymbol{\beta})) = \mathbf{0}$$

where $\mathbf{V}_i$ is the working covariance matrix of $\mathbf{Y}_i$.

Only the mean and the covariance of $\mathbf{Y}_i$ are required in the GEE method; a full specification of the joint distribution of the correlated responses is not needed. This is particularly convenient because the joint distribution for noncontinuous responses involves high-order associations and is complicated to specify. Moreover, the regression parameter estimates are consistent even when the working covariance is incorrectly specified. Because of these properties, the GEE method is popular in situations where the marginal effect is of interest and the responses are not continuous. However, the GEE approach can lead to biased estimates when missing responses depend on previous responses. The weighted GEE method, which is described in the section "Weighted Generalized Estimating Equations under the MAR Assumption" on page 3538, can provide unbiased estimates.

Working Correlation Matrix

Suppose $\mathbf{R}_i(\boldsymbol{\alpha})$ is an $n_i \times n_i$ "working" correlation matrix that is fully specified by the vector of parameters $\boldsymbol{\alpha}$. The covariance matrix of $\mathbf{Y}_i$ is modeled as

$$\mathbf{V}_i = \phi \mathbf{A}_i^{\frac{1}{2}} \mathbf{W}_i^{-\frac{1}{2}} \mathbf{R}(\boldsymbol{\alpha}) \mathbf{W}_i^{-\frac{1}{2}} \mathbf{A}_i^{\frac{1}{2}}$$

where $\mathbf{A}_i$ is an $n_i \times n_i$ diagonal matrix whose *j*th diagonal element is $v(\mu_{ij})$ and $\mathbf{W}_i$ is an $n_i \times n_i$ diagonal matrix whose *j*th diagonal is w_{ij}, where w_{ij} is a weight variable that is specified in the WEIGHT statement. If there is no WEIGHT statement, $w_{ij} = 1$ for all *i* and *j*. If $\mathbf{R}_i(\boldsymbol{\alpha})$ is the true correlation matrix of $\mathbf{Y}_i$, then $\mathbf{V}_i$ is the true covariance matrix of $\mathbf{Y}_i$.

In practice, the working correlation matrix is usually unknown and must be estimated. It is estimated in the iterative fitting process by using the current value of the parameter vector $\boldsymbol{\beta}$ to compute appropriate functions of the Pearson residual:

$$e_{ij} = \frac{y_{ij} - \mu_{ij}}{\sqrt{v(\mu_{ij})/w_{ij}}}$$

If you specify the working correlation matrix as $\mathbf{R_0} = \mathbf{I}$, which is the identity matrix, the GEE reduces to the independence estimating equation.

Table 50.14 shows the working correlation structures that are supported by the GEE procedure and the estimators that are used to estimate the working correlations.

Table 50.14 Working Correlation Structures and Estimators

Working Correlation Structure	Estimator
Fixed $\mathrm{Corr}(Y_{ij}, Y_{ik}) = r_{jk}$ where r_{jk} is the jkth element of a constant, user-specified correlation matrix $\mathbf{R_0}$	The working correlation is not estimated in this case.
Independent $\mathrm{Corr}(Y_{ij}, Y_{ik}) = \begin{cases} 1 & j = k \\ 0 & j \neq k \end{cases}$	The working correlation is not estimated in this case.
***m*-dependent** $\mathrm{Corr}(Y_{ij}, Y_{i,j+t}) = \begin{cases} 1 & t = 0 \\ \alpha_t & t = 1, 2, \ldots, m \\ 0 & t > m \end{cases}$	$\hat{\alpha}_t = \frac{1}{(K_t - p)\phi} \sum_{i=1}^{K} \sum_{j \leq n_i - t} e_{ij} e_{i,j+t}$ $K_t = \sum_{i=1}^{K} (n_i - t)$
Exchangeable $\mathrm{Corr}(Y_{ij}, Y_{ik}) = \begin{cases} 1 & j = k \\ \alpha & j \neq k \end{cases}$	$\hat{\alpha} = \frac{1}{(N^* - p)\phi} \sum_{i=1}^{K} \sum_{j<k} e_{ij} e_{ik}$ $N^* = 0.5 \sum_{i=1}^{K} n_i (n_i - 1)$
Unstructured $\mathrm{Corr}(Y_{ij}, Y_{ik}) = \begin{cases} 1 & j = k \\ \alpha_{jk} & j \neq k \end{cases}$	$\hat{\alpha}_{jk} = \frac{1}{(K - p)\phi} \sum_{i=1}^{K} e_{ij} e_{ik}$
Autoregressive AR(1) $\mathrm{Corr}(Y_{ij}, Y_{i,j+t}) = \alpha^t$ for $t = 0, 1, 2, \ldots, n_i - j$	$\hat{\alpha} = \frac{1}{(K_1 - p)\phi} \sum_{i=1}^{K} \sum_{j \leq n_i - 1} e_{ij} e_{i,j+1}$ $K_1 = \sum_{i=1}^{K} (n_i - 1)$

Dispersion Parameter

The dispersion parameter ϕ is estimated by

$$\hat{\phi} = \frac{1}{N-p} \sum_{i=1}^{K} \sum_{j=1}^{n_i} e_{ij}^2$$

where $N = \sum_{i=1}^{K} n_i$ is the total number of measurements and p is the number of regression parameters.

The square root of $\hat{\phi}$ is reported by PROC GEE as the scale parameter in the "Parameter Estimates for Response Model with Model-Based Standard Error" output table. If a fixed scale parameter is specified by using the NOSCALE option in the MODEL statement, then the fixed value is used in estimating the model-based covariance matrix and standard errors.

Quasi-likelihood Information Criterion

The quasi-likelihood information criterion (QIC) was developed by Pan (2001) as a modification of Akaike's information criterion (AIC) to apply to models fit by the GEE approach.

Define the quasi-likelihood under the independent working correlation assumption, evaluated with the parameter estimates under the working correlation of interest as

$$Q(\hat{\boldsymbol{\beta}}(R), \phi) = \sum_{i=1}^{K} \sum_{j=1}^{n_i} Q(\hat{\boldsymbol{\beta}}(R), \phi; (Y_{ij}, \mathbf{X}_{ij}))$$

where the quasi-likelihood contribution of the jth observation in the ith cluster is defined in the section "Quasi-likelihood Functions" on page 3533 and $\hat{\boldsymbol{\beta}}(R)$ are the parameter estimates that are obtained by using the GEE approach with the working correlation of interest R.

QIC is defined as

$$\mathrm{QIC}(R) = -2Q(\hat{\boldsymbol{\beta}}(R), \phi) + 2\mathrm{trace}(\hat{\Omega}_I \hat{V}_R)$$

where $\hat{V}_R$ is the robust covariance estimate and $\hat{\Omega}_I$ is the inverse of the model-based covariance estimate under the independent working correlation assumption, evaluated at $\hat{\boldsymbol{\beta}}(R)$, which are the parameter estimates that are obtained by using the GEE approach with the working correlation of interest R.

PROC GEE also computes an approximation to $\mathrm{QIC}(R)$, which is defined by Pan (2001) as

$$\mathrm{QIC}_u(R) = -2Q(\hat{\boldsymbol{\beta}}(R), \phi) + 2p$$

where p is the number of regression parameters.

Pan (2001) notes that QIC is appropriate for selecting regression models and working correlations, whereas QIC_u is appropriate only for selecting regression models.

Quasi-likelihood Functions

See McCullagh and Nelder (1989) and Hardin and Hilbe (2003) for discussions of quasi-likelihood functions. The contribution of observation j in cluster i to the quasi-likelihood function that is evaluated at the regression parameters $\boldsymbol{\beta}$ is expressed by $Q(\boldsymbol{\beta}, \phi; (Y_{ij}, \mathbf{X}_{ij})) = \frac{Q_{ij}}{\phi}$, where Q_{ij} is defined in the following list. These definitions are used in the computation of the quasi-likelihood information criteria (QIC) for goodness of fit of models that are fit by the GEE approach. The w_{ij} are prior weights, if any, that are specified in the WEIGHT or FREQ statement. Note that the definition of the quasi-likelihood for the negative binomial differs from that given in McCullagh and Nelder (1989). The definition used here allows the negative binomial quasi-likelihood to approach the Poisson as $k \to 0$.

- Normal:

$$Q_{ij} = -\frac{1}{2} w_{ij} (y_{ij} - \mu_{ij})^2$$

- Inverse Gaussian:

$$Q_{ij} = \frac{w_{ij}(\mu_{ij} - .5 y_{ij})}{\mu_{ij}^2}$$

- Gamma:

$$Q_{ij} = -w_{ij} \left[\frac{y_{ij}}{\mu_{ij}} + \log(\mu_{ij}) \right]$$

- Negative binomial:

$$Q_{ij} = w_{ij} \left[\log \Gamma \left(y_{ij} + \frac{1}{k} \right) - \log \Gamma \left(\frac{1}{k} \right) + y_{ij} \log \left(\frac{k \mu_{ij}}{1 + k \mu_{ij}} \right) + \frac{1}{k} \log \left(\frac{1}{1 + k \mu_{ij}} \right) \right]$$

- Poisson:

$$Q_{ij} = w_{ij} (y_{ij} \log(\mu_{ij}) - \mu_{ij})$$

- Binomial:

$$Q_{ij} = w_{ij} [r_{ij} \log(p_{ij}) + (n_{ij} - r_{ij}) \log(1 - p_{ij})]$$

- Multinomial (s categories):

$$Q_{ij} = w_{ij} \sum_{k=1}^{s} y_{ijk} \log(\mu_{ijk})$$

Alternating Logistic Regression

If the responses are binary (that is, they take only two values), then there is an alternative method to account for the association among the measurements. The alternating logistic regressions (ALR) algorithm of Carey, Zeger, and Diggle (1993) models the association between pairs of responses by using log odds ratios instead of using correlations, as ordinary GEEs do. The ALR algorithm of Heagerty and Zeger (1996) extends the method to GEEs that have ordinal multinomial responses (that is, they fall into one of C ordered categories).

ALR for Binary Data

For binary data, the correlation between the jth and kth response is, by definition,

$$\mathrm{Corr}(Y_{ij}, Y_{ik}) = \frac{\Pr(Y_{ij} = 1, Y_{ik} = 1) - \mu_{ij}\mu_{ik}}{\sqrt{\mu_{ij}(1-\mu_{ij})\mu_{ik}(1-\mu_{ik})}}$$

The joint probability in the numerator satisfies the following bounds, by elementary properties of probability, because $\mu_{ij} = \Pr(Y_{ij} = 1)$:

$$\max(0, \mu_{ij} + \mu_{ik} - 1) \le \Pr(Y_{ij} = 1, Y_{ik} = 1) \le \min(\mu_{ij}, \mu_{ik})$$

Therefore, the correlation is constrained to be within limits that depend in a complicated way on the means of the data.

The odds ratio, defined as

$$\mathrm{OR}(Y_{ij}, Y_{ik}) = \frac{\Pr(Y_{ij} = 1, Y_{ik} = 1)\Pr(Y_{ij} = 0, Y_{ik} = 0)}{\Pr(Y_{ij} = 1, Y_{ik} = 0)\Pr(Y_{ij} = 0, Y_{ik} = 1)}$$

is not constrained by the means and is preferred, in some cases, to correlations for binary data.

The ALR algorithm seeks to model the logarithm of the odds ratio, $\gamma_{ijk} = \log(\mathrm{OR}(Y_{ij}, Y_{ik}))$, as

$$\gamma_{ijk} = \mathbf{z}'_{ijk}\boldsymbol{\alpha}$$

where $\boldsymbol{\alpha}$ is a $q \times 1$ vector of regression parameters and $\mathbf{z}_{ijk}$ is a fixed, specified vector of coefficients.

The parameter γ_{ijk} can take any value in $(-\infty, \infty)$, with $\gamma_{ijk} = 0$ corresponding to no association.

The log odds ratio, when modeled in this way with a regression model, can take different values in subgroups defined by $\mathbf{z}_{ijk}$. For example, $\mathbf{z}_{ijk}$ can define subgroups within clusters, or it can define "block effects" between clusters.

You specify a GEE model for binary data that uses log odds ratios by specifying a model for the mean, as in ordinary GEEs, and by specifying a model for the log odds ratios. You can use any of the link functions appropriate for binary data in the model for the mean, such as logistic, probit, or complementary log-log.

ALR for Ordinal Multinomial Data

For ordinal multinomial data, let $O_{ij}, i = 1, \ldots, K, j = 1, \ldots, n_i$, denote the jth measurement on the ith subject. To apply the ALR algorithm, the responses O_{ij} are represented by a vector $\mathbf{Y}_{ij} = \left[Y_{ij1}, \ldots, Y_{ijC-1}\right]'$ of cumulative indicator variables $Y_{ijc} = \mathrm{I}(O_{i,j} \leq c)$. You model the cumulative probabilities $\mu_{ijc} = E\left(Y_{ijc}\right)$ by using a cumulative link function,

$$g\left(\mu_{ijc}\right) = \boldsymbol{\beta}_c + \mathbf{x}'_{ij}\boldsymbol{\beta}, \text{ for } c = 1, \ldots, C-1$$

where $\beta_1, \beta_2, \ldots, \beta_{C-1}$ are increasing intercept terms that depend only on the level c. Let the binary vector that represents the responses of the ith subject be $\mathbf{Y}_i = \left[\mathbf{Y}_{i1}, \ldots, \mathbf{Y}_{in_i}\right]'$ with corresponding means $\boldsymbol{\mu}_i = \left[\mu_{i1}, \ldots, \mu_{in_i}\right]'$.

The log odds ratio between two indicator variables Y_{ijc_1} and Y_{ikc_2} is modeled as

$$\gamma_{i(jk)(c_1c_2)} = \log(\mathrm{OR}(Y_{ijc_1}, Y_{ikc_2})) = \mathbf{z}'_{i(jk)(c_1c_2)}\boldsymbol{\alpha}$$

for $q \times 1$ regression parameters $\boldsymbol{\alpha}$ and fixed coefficients $\mathbf{z}_{i(jk)(c_1c_2)}$. As in Carey, Zeger, and Diggle (1993), $\boldsymbol{\alpha}$ then provides a vector of regression parameters in a logistic model for the conditional expectation $\xi_{i(jk)(c_1c_2)} = E\left(Y_{ijc_1}|Y_{ikc_2}\right)$. To estimate $\boldsymbol{\alpha}$, the conditional expectation is considered for all pairs Y_{ijc_1} and Y_{ikc_2} with $j < k$. Let

$$\begin{aligned}
\boldsymbol{\xi}_{i(jk)} &= \left[\xi_{i(jk)(11)}, \xi_{i(jk)(12)}, \ldots, \xi_{i(jk)(21)}, \ldots, \xi_{i(jk)(C-1,C-1)}\right]' \\
\boldsymbol{\xi}_i &= \left[\boldsymbol{\xi}_{i(12)}, \boldsymbol{\xi}_{i(13)}, \ldots, \boldsymbol{\xi}_{i(23)}, \ldots, \boldsymbol{\xi}_{i(n_i-1n_i)}\right]' \\
\mathbf{Y}^*_i &= \Big[\overbrace{Y_{i1} \otimes e_{C-1}, \ldots, Y_{i1} \otimes e_{C-1}}^{n_i-1}, \underbrace{Y_{i2} \otimes e_{C-1}, \ldots, Y_{i2} \otimes e_{C-1}}_{n_i-2}, \ldots, \overbrace{Y_{in_i-1} \otimes e_{C-1}}^{1}\Big]'
\end{aligned}$$

where $\otimes$ denotes the Kronecker product and e_l denotes a vector of dimension l composed of ones. The difference $\mathbf{Y}^*_i - \boldsymbol{\xi}_i$ represents the residuals of the model for the conditional expectation.

For both binary and multinomial data, the ALR estimates for $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ are the simultaneous solutions to the estimating equations

$$\begin{aligned}
\mathbf{S}_1(\boldsymbol{\beta}, \boldsymbol{\alpha}) &= \sum_{i=1}^{K} \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_{i11}^{-1} \left(\mathbf{Y}_i - \boldsymbol{\mu}_i(\boldsymbol{\beta})\right) = \mathbf{0} \\
\mathbf{S}_2(\boldsymbol{\beta}, \boldsymbol{\alpha}) &= \sum_{i=1}^{K} \frac{\partial \boldsymbol{\xi}_i}{\partial \boldsymbol{\alpha}}' \mathbf{V}_{i33}^{-1} \left(\mathbf{Y}^*_i - \boldsymbol{\xi}_i\right) = \mathbf{0}
\end{aligned}$$

where $\mathbf{V}_{i11} = \mathrm{cov}\left(\mathbf{Y}_i\right)$ and $\mathbf{V}_{i33} = \mathrm{diag}\left[\boldsymbol{\xi}_i\left(1 - \boldsymbol{\xi}_i\right)\right]$. The fitting algorithm alternates between a GEE step to update the model for the mean and a logistic regression step to update the log odds ratio model. Upon convergence, the ALR algorithm provides estimates of the regression parameters for the mean, $\boldsymbol{\beta}$; the regression parameters for the log odds ratios, $\boldsymbol{\alpha}$; their standard errors; and their covariances.

Specifying Log Odds Ratio Models

Specifying a regression model for the log odds ratio requires you to specify the rows of the matrix $\mathbf{z}$. For binary data, there is a row $\mathbf{z}_{ijk}$ for each cluster i and within-cluster pair (j, k). For ordinal multinomial data, there is a row $\mathbf{z}_{i(jk)(c_1c_2)}$ for each cluster i, within-cluster pair (j, k), and choice of levels (c_1, c_2).

For ordinal multinomial data, the GEE procedure supports only the ALR method that uses a fully exchangeable regression structure for the log odds ratio. In a fully exchangeable model, the log odds ratio is constant for all

clusters i, within-cluster pair (j,k), and levels (c_1, c_2). You select a fully exchangeable model for the log odds ratio by specifying LOGOR=EXCH.

For binary data, the GEE procedure provides several methods of specifying $\mathbf{z}_{ijk}$. You apply these methods by specifying LOGOR=*keyword* and associated options in the REPEATED statement. The supported *keywords* and the resulting log odds ratio models are described as follows:

EXCH specifies exchangeable log odds ratios. In this model, the log odds ratio is a constant for all clusters i and pairs (j,k). The parameter α is the common log odds ratio.

$$\mathbf{z}_{ijk} = 1 \quad \text{for all} \quad i, j, k$$

FULLCLUST specifies fully parameterized clusters. Each cluster is parameterized in the same way, and there is a parameter for each unique pair within clusters. If a complete cluster is of size n, then there are $\frac{n(n-1)}{2}$ parameters in the vector $\boldsymbol{\alpha}$. For example, if a full cluster is of size 4, then there are $\frac{4\times 3}{2} = 6$ parameters, and the $\mathbf{z}$ matrix is of the form

$$\mathbf{Z} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The elements of $\boldsymbol{\alpha}$ correspond to log odds ratios for cluster pairs in the following order:

Pair	Parameter
(1,2)	Alpha1
(1,3)	Alpha2
(1,4)	Alpha3
(2.3)	Alpha4
(2,4)	Alpha5
(3,4)	Alpha6

LOGORVAR(*variable*) specifies log odds ratios by cluster. The argument *variable* is a variable name that defines the "block effects" between clusters. The log odds ratios are constant within clusters, but they take a different value for each different value of the *variable*. For example, if Center is a variable in the input data set that takes a different value for k treatment centers, then when you specify LOGOR=LOGORVAR(Center), you get a model that has different log odds ratios for each of the k centers, constant within center.

NESTK specifies k-nested log odds ratios. You must also specify the SUBCLUST=*variable* option to define subclusters within clusters. Within each cluster, PROC GEE computes a log odds ratio parameter for pairs that have the same value of *variable* for both members of the pair and one log odds ratio parameter for each unique combination of different values of *variable*.

NEST1 specifies 1-nested log odds ratios. You must also specify the SUBCLUST=*variable* option to define subclusters within clusters. There are two log odds ratio parameters for this model. Pairs that have the same value of *variable* correspond to one parameter; pairs that have different values of *variable* correspond to the other parameter. For example, if patients are clustered by hospital and subclusters are the wards within those hospitals, then the outcomes of patients within the same ward have one log odds ratio parameter, and the outcomes of patients from different wards have the other parameter.

ZFULL specifies the full **z** matrix. You must also specify a SAS data set that contains the **z** matrix by using the ZDATA=*data-set-name* option. Each observation in the data set corresponds to one row of the **z** matrix. You must specify the ZDATA data set as if all clusters are complete—that is, as if all clusters are the same size and there are no missing observations. The ZDATA data set has $K[n_{max}(n_{max}-1)/2]$ observations, where K is the number of clusters and n_{max} is the maximum cluster size. If the members of cluster i are ordered as $1, 2, \ldots, n$, then the rows of the **z** matrix must be specified for pairs in the order $(1,2), (1,3), \ldots, (1,n), (2,3), \ldots, (2,n), \ldots, (n-1,n)$. The variables that you specify in the REPEATED statement for the SUBJECT effect must also be present in the ZDATA= data set to identify clusters. You must specify variables in the data set that define the columns of the **z** matrix by using the ZROW=*variable-list* option. If there are q columns (q variables in *variable-list*), then there are q log odds ratio parameters. You can optionally specify variables that indicate the cluster pairs corresponding to each row of the **z** matrix by using the YPAIR=(*variable1, variable2*) option. If you specify this option, the data from the ZDATA data set are sorted within each cluster by *variable1* and *variable2*. See Example 50.4 for an example of specifying a full **z** matrix.

ZREP specifies a replicated **z** matrix. You specify **z** matrix data exactly as you do for the ZFULL option case, except that you specify only one complete cluster. The **z** matrix for the one cluster is replicated for each cluster. The number of observations in the ZDATA data set is $\frac{n_{max}(n_{max}-1)}{2}$, where n_{max} is the size of a complete cluster (a cluster with no missing observations).

ZREP(*matrix*) specifies direct input of the replicated **z** matrix. You specify the **z** matrix for one cluster by using the syntax LOGOR=ZREP ($(y_j \;\; y_k)\; z_{jk1} \;\; z_{jk2} \cdots z_{jkq}, \cdots$), where y_j and y_k are numbers that represent a pair of observations from the ith cluster and the values $z_{jk1}, z_{jk2}, \ldots, z_{jkq}$ make up the corresponding row $\mathbf{z}_{ijk}$ of the **z** matrix. The number of specified rows is $\frac{n_{max}(n_{max}-1)}{2}$, where n_{max} is the size of a complete cluster (a cluster with no missing observations). For example,

```
logor =  zrep((1 2) 1 0,
              (1 3) 1 0,
              (1 4) 1 0,
              (2 3) 1 1,
              (2 4) 1 1,
              (3 4) 1 1)
```

specifies the $\frac{4\times 3}{2} = 6$ rows of the **z** matrix for a cluster of size 4 with $q = 2$ log odds ratio parameters. The log odds ratio for the pairs (1 2), (1 3), (1 4) is α_1, and the log odds ratio for the pairs (2 3), (2 4), (3 4) is $\alpha_1 + \alpha_2$.

Weighted Generalized Estimating Equations under the MAR Assumption

In longitudinal studies, response measurements are often missing because of skipped visits or dropouts. Suppose r_{ij} is the indicator that the response y_{ij} is observed, where $r_{ij} = 1$ if y_{ij} is observed and 0 otherwise. Missing data patterns can be classified into two types: dropout and intermittent. A dropout occurs if an individual skips a particular visit and then never comes back for subsequent visits. That is, if $r_{ij} = 0$, then $r_{ik} = 0$ for all $k > j$. Otherwise, the missing data pattern is intermittent. Intermittent patterns can be quite complicated; only dropout patterns are considered here.

The mechanism for missingness can be described by a statistical model for the probability of observing a missing value, and making the right assumption about the mechanism is crucial to methods that handle missing data. Missingness mechanisms are classified into three types: missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR) (Rubin 1976).

Assumptions about longitudinal data that include missing responses caused by dropouts are classified as follows:

- The data are said to be MCAR if the probability of a missing response is independent of its past, current, and future responses conditional on the covariates. That is, $P(r_{ij} = 0|\mathbf{Y}_i, \mathbf{X}_i) = P(r_{ij} = 0|\mathbf{X}_i)$.
- The data are said to be MAR if the probability of a missing response is independent of its current and future responses conditional on the observed past responses and the covariates. That is, $P(r_{ij} = 0|r_{ij-1} = 1, X_i, Y_i) = P(r_{ij} = 0|r_{ij-1} = 1, X_i, y_{i1}, \ldots, y_{ij-1})$. MAR is a weaker assumption than MCAR.
- The data are said to be MNAR if the probability of a missing response depends on the unobserved responses. MNAR is the most general and the most problematic missing-data scenario.

The GEE procedure implements two different weighted methods (observation-specific and subject-specific) of estimating the regression parameter $\boldsymbol{\beta}$ when dropouts occur. Both methods provide consistent estimates if the data are MAR. The weighted GEE methods are not supported for the multinomial distribution for polytomous responses.

Observation-Specific Weighted GEE Method

Suppose w_{ij} is the weight for y_{ij}, which is defined as the inverse probability of observing y_{ij}. In other words, $w_{ij} = P(r_{ij} = 1|X_i, Y_i)^{-1}$. Suppose W_i is a $T \times T$ diagonal matrix whose jth diagonal is $r_{ij} w_{ij}$. The responses for the ith subject are $\mathbf{Y}_i = (y_{i1}, y_{i2}, \ldots, y_{iT})'$. Consider the following weighted generalized estimating equations (Robins and Rotnitzky 1995; Preisser, Lohman, and Rathouz 2002):

$$\mathbf{S}_{ow}(\boldsymbol{\beta}) = \sum_{i=1}^{K} \frac{\partial \boldsymbol{\mu}_i'}{\partial \boldsymbol{\beta}} \mathbf{V}_i^{-1} W_i (\mathbf{Y}_i - \boldsymbol{\mu}_i(\boldsymbol{\beta})) = \mathbf{0}$$

Unlike the standard generalized estimating equations, the weighted generalized estimating equations are unbiased when the observations are appropriately weighted and lead to consistent estimates of $\boldsymbol{\beta}$.

The weights w_{ij} are often unknown in practice and are estimated by a logistic regression model under the MAR assumption. Specifically, suppose that $\lambda_{ij} = P(r_{ij} = 1|r_{ij-1} = 1, X_i, Y_i)$ denotes the probability of observing the response y_{ij} given its observed previous responses.

Under the MAR assumption,

$$\lambda_{ij} = P(r_{ij} = 1 | r_{ij-1} = 1, X_i, Y_i) = P(r_{ij} = 1 | r_{ij-1} = 1, X_i, Y_1, \ldots, Y_{j-1})$$

Using the observed data, λ_{ij} can be predicted from a logistic regression model,

$$\text{logit}\{\lambda_{ij}\} = z_{ij}\boldsymbol{\alpha}$$

where the z_{ij} are predictors that usually include the covariates x_{ij}, the past responses, and the indicators for visit times. The dropout process implies that the estimated probability of observing y_{ij} can be expressed as a cumulative product of conditional probabilities:

$$\hat{P}(r_{ij} = 1 | X_i, Y_i) = \lambda_{i1}(\hat{\boldsymbol{\alpha}}) \times \lambda_{i2}(\hat{\boldsymbol{\alpha}}) \times \cdots \times \lambda_{ij}(\hat{\boldsymbol{\alpha}})$$

With the estimated weights $\hat{w}_{ij} = \hat{P}(r_{ij} = 1 | X_i, Y_i)^{-1}$, the regression parameter $\boldsymbol{\beta}$ is estimated by solving the equation for $\mathbf{S}_{ow}(\boldsymbol{\beta})$.

The regression parameter $\boldsymbol{\beta}$ can be estimated by solving for $\mathbf{S}_{ow}(\boldsymbol{\beta})$ after plugging in the estimated weights. The fitting algorithm is described in the section "Fitting Algorithm for Weighted GEE" on page 3540.

Subject-Specific Weighted GEE Method

Unlike the observation-specific weighted method, which assigns an observation-specific weight to each observation, the subject-specific weighted method assigns a single weight to each subject. In other words, all the observations from a subject receive the same weight. Specifically, the subject-specific weighted method obtains the regression parameter estimates by solving the equations

$$\mathbf{S}_{sw}(\boldsymbol{\beta}) = \sum_{i=1}^{K} \mathbf{D}_i' \mathbf{V}_i^{-1} w_i (\mathbf{Y}_i - \boldsymbol{\mu}_i(\boldsymbol{\beta})) = \mathbf{0}$$

where the responses for the ith subject are $\mathbf{Y}_i = (y_{i1}, y_{i2}, \ldots, y_{in_i})'$ and the weight w_i for subject i is the inverse probability of a subject i dropping out at the observed time (Fitzmaurice, Molenberghs, and Lipsitz 1995; Preisser, Lohman, and Rathouz 2002). Note that the weight w_i is a scalar, in contrast to the weight matrix $\mathbf{W}_i$ that the observation-specific weighted GEE method uses.

The subject-specific weighted estimating equations are also unbiased when the subjects are appropriately weighted and lead to consistent estimates of the regression parameters $\boldsymbol{\beta}$.

The weight w_i is usually unknown in practice and needs to be estimated. Suppose subject i drops out at time $m_i = \sum_{j=1}^{T} r_{ij} + 1$. Assume that the first visit y_{i1} is always observed with $r_{i1} = 1$. Thus, the dropout times m_i range from 2 to $T+1$. Note that a dropout time of $T+1$ indicates that subject i completes all the T visits and dropout does not occur.

The weight w_i is defined as follows: if subject i drops out before completing the last visit (that is, $m_i \leq T$), then $w_i = P(r_{im_i} = 0, r_{im_i-1} = 1 | X_i, Y_i)^{-1}$; otherwise, the subject completes all the T visits (that is, $m_i = T + 1$), and $w_i = P(r_{iT} = 1 | X_i, Y_i)^{-1}$.

Similar to the process for the observation-specific weighted method, the dropout process for the subject-specific weighted method implies that subject-specific weights can be estimated as a cumulative product of conditional probabilities:

$$\hat{w}_i = P(r_{im_i} = 0, r_{im_i-1} = 1 | X_i, Y_i)^{-1} = [\lambda_{i1}(\hat{\boldsymbol{\alpha}}) \times \cdots \times \lambda_{im_i-1}(\hat{\boldsymbol{\alpha}}) \times (1 - \lambda_{im_i}(\hat{\boldsymbol{\alpha}}))]^{-1}, \text{ if } m_i \leq T$$
$$\hat{w}_i = P(r_{im_i-1} = 1 | X_i, Y_i)^{-1} = [\lambda_{i1}(\hat{\boldsymbol{\alpha}}) \times \lambda_{i2}(\hat{\boldsymbol{\alpha}}) \times \cdots \times \lambda_{im_i-1}(\hat{\boldsymbol{\alpha}})]^{-1}, \text{ if } m_i = T + 1$$

Thus, the subject-specific weights $\hat{w}_i$ can be obtained after λ_{ij} is estimated by fitting a logistic regression to the data (r_{ij}, z_{ij}).

The regression parameter $\boldsymbol{\beta}$ from the subject-specific weighted GEE method can be estimated by solving for $\mathbf{S}_{sw}(\boldsymbol{\beta})$ after plugging in the estimated weights. The fitting algorithm is described in the section "Fitting Algorithm for Weighted GEE" on page 3540. The subject-specific weighting scheme was originally developed for computational convenience. Preisser, Lohman, and Rathouz (2002) showed that the observation-level weighted GEE method produces more efficient estimates than the cluster-level weighted GEE method for incomplete longitudinal binary data.

Fitting Algorithm for Weighted GEE

The following fitting algorithm fits marginal models by using the observation-specific or the subject-specific weighted GEE method when the dropout process is missing at random:

1. Fit a logistic regression to the data (r_{ij}, z_{ij}) to obtain an estimate of $\boldsymbol{\alpha}$ and estimate the weights.
2. Compute an initial estimate of $\boldsymbol{\beta}$ by using an ordinary generalized linear model, assuming independence of the responses.
3. Compute the working correlation matrix $\mathbf{R}$ based on the standardized residuals, the current estimate of $\boldsymbol{\beta}$, and the specified structure of $\mathbf{R}$.
4. Compute the estimated covariance matrix:

$$\mathbf{V}_i = \phi \mathbf{A}_i^{\frac{1}{2}} \hat{\mathbf{R}}(\boldsymbol{\alpha}) \mathbf{A}_i^{\frac{1}{2}}$$

5. Update $\hat{\boldsymbol{\beta}}$:

$$\hat{\boldsymbol{\beta}}_{r+1} = \hat{\boldsymbol{\beta}}_r + \left[\sum_{i=1}^{K} \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}} \right]^{-1} \left[\sum_{i=1}^{K} \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} \mathbf{W}_i (\mathbf{Y}_i - \boldsymbol{\mu}_i) \right]$$

   where $\mathbf{Y}_i$, $\boldsymbol{\mu}_i$, $\mathbf{V}_i$, and $\mathbf{W}_i$ are as follows:

   - For the observation-specific weighted method, $\mathbf{Y}_i = (y_{i1}, y_{i2}, \ldots, y_{iT})'$; $\boldsymbol{\mu}_i$ and $\mathbf{V}_i$ are its corresponding mean vector and working covariance matrix, respectively; and $\mathbf{W}_i$ is a $T \times T$ diagonal matrix whose jth diagonal is $r_{ij}\hat{w}_{ij}$.
   - For the subject-specific weighted method, $\mathbf{Y}_i = (y_{i1}, y_{i2}, \ldots, y_{in_i})'$; $\boldsymbol{\mu}_i$ and $\mathbf{V}_i$ are its corresponding mean vector and working covariance matrix, respectively; and $\mathbf{W}_i$ is a $n_i \times n_i$ diagonal matrix whose jth diagonal is $\hat{w}_i$.

6. Repeat steps 3–5 until convergence.

Note that you can use the WEIGHT statement in the GENMOD procedure to perform a two-stage strategy that is often used in practice to obtain the weighted GEE estimates. You fit a logistic regression to the data (r_{ij}, z_{ij}) to obtain the weights as described in the preceding steps. Then you estimate $\boldsymbol{\beta}$ by specifying the estimated weights in the WEIGHT statement in PROC GENMOD for the GEE analysis. For the subject-specific weighted GEE method, this approach is appropriate for any working correlation structure. However,

for the observation-specific weighted method, this approach is appropriate only for the independent working correlation structure.

The two-stage approach results in standard errors that are larger than those that are produced by using the MISSMODEL statement in the GEE procedure (because PROC GENMOD treats the weights as fixed and known). Thus, the two-stage approach that uses PROC GENMOD results in conservative inference (Fitzmaurice, Laird, and Ware 2011). The GEE procedure computes the parameter estimate covariances as described in (Fitzmaurice, Laird, and Ware 2011) and Preisser, Lohman, and Rathouz (2002).

Missing Data

Suppose that each subject in a longitudinal study is measured at T times. In other words, for the ith subject you measure T responses $(y_{i1}, y_{i2}, \ldots, y_{iT})$ and T corresponding covariates $(\mathbf{x}_{i1}, \mathbf{x}_{i2}, , \ldots, x_{iT})$.

By default, the GEE procedure handles missing data in the same manner as the standard GEE method in the GENMOD procedure. The working correlation matrix is estimated from data that contain both intermittent and dropout types of missing values by using the all-available-pairs method, in which all nonmissing pairs of data are used in the moment estimators. The resulting covariances and standard errors are valid under the missing completely at random (MCAR) assumption. For more information, see the section "Missing Data" on page 3681 in Chapter 51, "The GENMOD Procedure."

When you specify the MISSMODEL statement in the GEE procedure to use the weighted GEE method to analyze the data, the procedure uses observations that have missing values in the response, provided that the missing values for all subjects are caused by dropouts. If the missing values are intermittent for any of the subjects, then the weighted GEE method does not apply and the procedure terminates.

For the observation-specific weighted GEE method, the covariates for all the observations for a subject must be observed, regardless of whether the response is missing. For each subject, the input data set must provide T observations.

For the subject-specific weighted GEE method, the covariates for a subject who drops out at time k must be observed for the observations up to and including time k. The input data set must provide at least k observations for this subject. The covariates must be observed for all observations on a subject who completes the study, and the input data set must provide T observations for this subject.

For more information about how weighted GEE methods handle missing values, see Fitzmaurice, Laird, and Ware (2011) and Preisser, Lohman, and Rathouz (2002).

Type 3 Analysis

A Type 3 analysis is similar to the Type 3 sums of squares used in PROC GLM, except that generalized score tests for Type 3 contrasts instead of Type 3 sums of squares are computed. Briefly, a Type 3 estimable function (contrast) for an effect is a linear function of the model parameters that involves the parameters of the effect and any interactions with that effect. A test of the hypothesis that the Type 3 contrast for a main effect is equal to 0 is intended to test the significance of the main effect in the presence of interactions. For more information about Type 3 estimable functions, see Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions." Also see Littell, Freund, and Spector (1991).

Boos (1992) and Rotnitzky and Jewell (1990) describe score tests applicable to testing $\mathbf{L}'\boldsymbol{\beta} = \mathbf{0}$ in GEEs, where $\mathbf{L}'$ is a user-specified $r \times p$ contrast matrix or a contrast for a Type 3 test of hypothesis.

Let $\tilde{\boldsymbol{\beta}}$ be the regression parameters that result from solving the GEE under the restricted model $\mathbf{L}'\boldsymbol{\beta} = \mathbf{0}$, and let $\mathbf{S}(\tilde{\boldsymbol{\beta}})$ be the generalized estimating equation values at $\tilde{\boldsymbol{\beta}}$.

The generalized score statistic is

$$T = \mathbf{S}(\tilde{\boldsymbol{\beta}})'\boldsymbol{\Sigma}_m\mathbf{L}(\mathbf{L}'\boldsymbol{\Sigma}_e\mathbf{L})^{-1}\mathbf{L}'\boldsymbol{\Sigma}_m\mathbf{S}(\tilde{\boldsymbol{\beta}})$$

where $\boldsymbol{\Sigma}_m$ is the model-based covariance estimate and $\boldsymbol{\Sigma}_e$ is the empirical covariance estimate. The p-values for T are computed based on the chi-square distribution with r degrees of freedom, where r is the rank of $\mathbf{L}$.

A Type 3 analysis can consume considerable computation time because a constrained model is fitted for each effect. Wald statistics for Type 3 contrasts are computed if you specify the WALD option. Wald statistics for contrasts use less computation time than likelihood ratio statistics but might be less accurate indicators of the significance of the effect of interest. The Wald statistic for testing $\mathbf{L}'\boldsymbol{\beta} = \mathbf{0}$ is defined by

$$S = (\mathbf{L}'\hat{\boldsymbol{\beta}})'(\mathbf{L}'\boldsymbol{\Sigma}_e\mathbf{L})^{-1}(\mathbf{L}'\hat{\boldsymbol{\beta}})$$

where $\mathbf{L}$ is the contrast matrix, $\boldsymbol{\beta}$ are the GEE parameter estimates, and $\boldsymbol{\Sigma}_e$ is the empirical covariance estimate. The asymptotic distribution of S is chi-square with r degrees of freedom, where r is the rank of $\mathbf{L}$.

The results of this type of analysis do not depend on the order in which the terms are specified in the MODEL statement. Type 3 analyses that use score statistics are not supported for nominal response data or weighted GEE methods. Type 3 analyses can be conducted using the Wald statistics for all the models that the GEE procedure supports.

The preceding development for Type 3 tests assumes that the rank of the empirical covariance matrix $\boldsymbol{\Sigma}_e$ is not less than the row rank of the contrast matrix $\mathbf{L}$. When the rank of $\boldsymbol{\Sigma}_e$ is less than the row rank of $\mathbf{L}$, estimability of the function is not sufficient to ensure that the chi-square test statistic has a unique value no matter what kind of generalized inverse is used to compute $(\mathbf{L}'\boldsymbol{\Sigma}_e\mathbf{L})^{-}$.

Although it is extremely rare, it is possible in practice that the uniqueness condition is not satisfied. For example, if the number of clusters is less than the number of nonsingular parameters in the model, then the matrix of coefficients for testing the overall null does not satisfy the uniqueness condition. If this condition is not satisfied, then the chi-square statistic for testing $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ is not invariant to the choice of the g_2-inverse of $\mathbf{L}\boldsymbol{\Sigma}_e\mathbf{L}'$. This chi-square test is not recommended when the uniqueness condition is not satisfied. An alternative approach would be to increase the number of clusters or to find a parsimonious model so that the number of parameters is less than the number of clusters. When the rank of $\boldsymbol{\Sigma}_e$ is less than the row rank of $\mathbf{L}$ for a test, the procedure prints a note to the SAS log.

ODS Table Names

PROC GEE assigns a name to each table that it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. Table 50.15 lists these names. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 50.15 ODS Tables Produced by PROC GEE

ODS Table Name	Description	Statement	Option
ClassLevels	Classification variable levels	CLASS	Default
Coef	Coefficients for LS-means	LSMEANS	E

Table 50.15 *continued*

ODS Table Name	Description	Statement	Option
Diffs	Differences of LS-means	LSMEANS	DIFF
Estimates	Estimates of contrasts	ESTIMATE	Default
GEEEmpPEst	Parameter estimates with empirical standard errors	REPEATED	Default
GEEExchCorr	Exchangeable working correlation value	REPEATED	TYPE=EXCH
GEEFitCriteria	QIC fit criteria	REPEATED	Default
GEELogORInfor	GEE log odds ratio model information	REPEATED	LOGOR=
GEEModInfo	GEE model information	REPEATED	Default
GEEModPEst	Parameter estimates with model-based standard errors	REPEATED	MODELSE
GEENCorr	Model-based correlation matrix	REPEATED	MCORRB
GEENCov	Model-based covariance matrix	REPEATED	MCOVB
GEERCorr	Empirical correlation matrix	REPEATED	ECORRB
GEERCov	Empirical covariance matrix	REPEATED	ECOVB
GEEWCorr	GEE working correlation matrix	REPEATED	CORRW
LSMeans	LS-means	LSMEANS	Default
LSMLines	Lines display for LS-means	LSMEANS	LINES
Margins	Predictive margins	MARGINS	Default
MarginsDiffs	Differences of predictive margins	MARGINS	DIFF
MarginsSliceDiffs	Differences of sliced predictive margins	MARGINS	SLICEBY= or SLICEDIFF
MarginsSliceTests	Tests of sliced predictive margins	MARGINS	SLICEBY=
MissModelPEst	Parameter estimates for the missingness model	MISSMODEL	Default
MissPattern	Frequency counts for dropout times	MISSMODEL	Default
ModelInfo	Model information	MODEL	Default
NObs	Number of observations summary		Default
ParmInfo	Parameter indices	REPEATED	MCORRB, MCOVB, ECORRB, ECOVB
ResponseProfile	Frequency counts for binary models	MODEL	DIST=BINOMIAL
Type3	Type 3 tests	MODEL	TYPE3

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

ODS Graph Names

PROC GEE assigns a name to each graph it creates using ODS. You can use these names to refer to the graphs when you use ODS. Table 50.16 lists the names.

To request these graphs, ODS Graphics must be enabled and you must specify the statement and option that are indicated in Table 50.16.

Table 50.16 Graph Produced by PROC GEE

ODS Graph Name	Description	Statement	Option
Histogram	Histogram of predicted weights from the missingness model	PROC GEE	PLOTS=

Examples: GEE Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

The following examples illustrate some of the capabilities of the GEE procedure. These examples are not intended to represent definitive analyses of the data sets that are presented here.

Example 50.1: Comparison of the Marginal and Random Effect Models for Binary Data

(View the complete code for this example (geeex1.sas) in the example repository.)

A clinical trial (Stokes, Davis, and Koch 2012) was conducted to compare two treatments for a respiratory illness. Patients in each of two centers were randomly assigned to two groups: one group received the active treatment and one group received a placebo.

During treatment, respiratory status was determined for each of four visits and is represented by the variable Outcome (coded here as 0 = poor, 1 = good). The variables Center, Treatment, Sex, and Baseline (baseline respiratory status) are classification variables that have two levels. The variable Age (age at time of entry into the study) is a continuous variable.

All 111 patients completed the study. That is, there are no missing data for responses or covariates. The following statements create the data set Resp:

```
data Resp;
   input Center ID Treatment $ Sex $ Age Baseline Visit1-Visit4;
   datalines;
1  1 P M 46 0 0 0 0 0
1  2 P M 28 0 0 0 0 0
1  3 A M 23 1 1 1 1 1
1  4 P M 44 1 1 1 1 0
1  5 P F 13 1 1 1 1 1
1  6 A M 34 0 0 0 0 0

   ... more lines ...

2 51 A M 43 1 1 1 1 0
2 52 A F 39 0 1 1 1 1
2 53 A M 68 0 1 1 1 1
2 54 A F 63 1 1 1 1 1
2 55 A M 31 1 1 1 1 1
;

data Resp;
   set Resp;
   Visit=1;  Outcome=Visit1;  output;
   Visit=2;  Outcome=Visit2;  output;
   Visit=3;  Outcome=Visit3;  output;
   Visit=4;  Outcome=Visit4;  output;
run;
```

Suppose y_{ij} represents the respiratory status of patient i at the jth visit, $j = 1, \ldots, 4$, and $\mu_{ij} = \mathrm{E}(y_{ij})$ represents the mean of the respiratory status. Logistic regression is commonly used to analyze binary response data. You can use the variance function for the binomial distribution, $v(\mu_{ij}) = \mu_{ij}(1 - \mu_{ij})$, and the logit link function, $g(\mu_{ij}) = \log(\mu_{ij}/(1 - \mu_{ij}))$. The model for the mean is $g(\mu_{ij}) = \mathbf{x}_{ij}{}'\boldsymbol{\beta}$, where $\boldsymbol{\beta}$ is a vector of regression parameters to be estimated.

The following SAS statements perform the GEE model fit:

```
proc gee data=Resp descend;
   class ID Treatment Center Sex Baseline;
   model Outcome=Treatment Center Sex Age Baseline /
         dist=bin link=logit;
   repeated subject=ID(Center) / corr=exch corrw;
run;
```

Both the MODEL statement and the REPEATED statement are required.

In the MODEL statement, you use the DIST=BIN and LINK=LOGIT options to specify a logistic regression, and you specify Outcome as the response variable and Treatment, Center, Sex, Age, and Baseline as the explanatory variables. The DESCEND option in the PROC GEE statement requests that the probability that Outcome = 1 be modeled. If the DESCEND option had not been specified, the probability that Outcome = 0 would be modeled by default.

You use the REPEATED statement to specify the subject and the correlation structure of the responses. The SUBJECT=ID(CENTER) option specifies that the observations in any single cluster are uniquely identified by Center and ID. An equivalent specification is SUBJECT=ID*CENTER. Because the same ID values are used in each center, one of these specifications is needed. If ID values were unique across all centers, SUBJECT=ID could be specified. The option TYPE=EXCH specifies the exchangeable working correlation structure.

The "Model Information" table displayed in Output 50.1.1 provides information about the specified logistic regression model and the input data set.

Output 50.1.1 Model Information

The GEE Procedure

Model Information	
Data Set	WORK.RESP
Distribution	Binomial
Link Function	Logit
Dependent Variable	Outcome

General information about the GEE analysis is displayed in Output 50.1.2, and model fit criteria for the model are displayed in Output 50.1.3.

Output 50.1.2 Model Fitting Information

GEE Model Information	
Correlation Structure	Exchangeable
Subject Effect	ID(Center) (111 levels)
Number of Clusters	111
Correlation Matrix Dimension	4
Maximum Cluster Size	4
Minimum Cluster Size	4

Output 50.1.3 Model Fitting Information

GEE Fit Criteria	
QIC	512.5723
QICu	499.4873

The results of GEE model fitting are displayed in Output 50.1.4. If you specify no other options, the standard errors, confidence intervals, Z scores, and p-values are based on empirical standard error estimates. You can specify the MODELSE option in the REPEATED statement to create a table that is based on model-based standard error estimates.

Output 50.1.4 Results of Model Fitting

Parameter Estimates for Response Model with Empirical Standard Error Estimates							
Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept		1.6391	0.5247	0.6107	2.6675	3.12	0.0018
Treatment	A	1.2654	0.3467	0.5859	1.9448	3.65	0.0003
Treatment	P	0.0000	0.0000	0.0000	0.0000	.	.
Center	1	-0.6495	0.3532	-1.3418	0.0428	-1.84	0.0660
Center	2	0.0000	0.0000	0.0000	0.0000	.	.
Sex	F	0.1368	0.4402	-0.7261	0.9996	0.31	0.7560
Sex	M	0.0000	0.0000	0.0000	0.0000	.	.
Age		-0.0188	0.0130	-0.0442	0.0067	-1.45	0.1480
Baseline	0	-1.8457	0.3460	-2.5238	-1.1676	-5.33	<.0001
Baseline	1	0.0000	0.0000	0.0000	0.0000	.	.

Treatment and Baseline appear to be strongly influential, and Center might be marginally significant.

For comparison, a generalized linear mixed model is fitted to the data set to obtain subject-specific effects. Specifically, consider the logistic regression model,

$$\mathrm{logit}(\mathrm{E}(y_{ij}|b_i)) = \mathbf{x}_{ij}{}'\boldsymbol{\beta}^* + b_i$$

where the random effect b_i is normally distributed with zero mean and variance, $\mathrm{Var}(b_i) = \sigma_b^2$.

The following statements use the GLIMMIX procedure to fit a generalized linear mixed model:

```
proc glimmix data=Resp;
   class ID Treatment Center Sex Baseline;
   model Outcome (desc)=Treatment Center Sex Age Baseline /
         dist=binary solution;
   random ID(Center);
run;
```

Output 50.1.5 displays the parameter estimates for the fixed effects in the generalized linear mixed model.

Output 50.1.5 Parameter Estimates

The GLIMMIX Procedure

Solutions for Fixed Effects

Effect	Treatment	Sex	Center	Baseline	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept					1.7936	0.6292	105	2.85	0.0053
Treatment	A				1.4758	0.3898	333	3.79	0.0002
Treatment	P				0	.	.	.	.
Center			1		-0.7201	0.4051	105	-1.78	0.0784
Center			2		0	.	.	.	.
Sex		F			0.1732	0.5034	333	0.34	0.7310
Sex		M			0	.	.	.	.
Age					-0.02011	0.01507	333	-1.33	0.1831
Baseline				0	-2.1343	0.3971	333	-5.38	<.0001
Baseline				1	0	.	.	.	.

From Output 50.1.4 and Output 50.1.5, you can see that the parameter estimates from the marginal model and the mixed-effects model differ. For example, the estimated treatment effects are 1.2654 and 1.4758 from the marginal model and the mixed-effects model, respectively.

The interpretation of the model effects in the marginal and random models differs. For example, the estimated treatment effect from the marginal model indicates that, on average, the odds of a good response for the patients is $e^{1.2654} = 3.5$ times higher when they receive the active treatment versus the placebo. The estimated treatment effect from the generalized linear mixed model indicates that an individual patient's odds of a good response is $e^{1.4758} = 4.4$ times higher when the patient receives the active treatment versus the placebo.

The choice of the marginal model or a subject-specific model often depends on the goal of your analysis: whether you are interested in population-averaged effects or subject-specific effects. For more information, see Diggle et al. (2002); Fitzmaurice, Laird, and Ware (2011).

Example 50.2: Log-Linear Model for Count Data

(View the complete code for this example (geeex2.sas) in the example repository.)

The following example demonstrates how you can fit a GEE model to count data. The data are analyzed by Diggle, Liang, and Zeger (1994). The response is the number of epileptic seizures, which was measured at the end of each of eight two-week treatment periods over sixteen weeks. The first eight weeks were the baseline period (during which no treatment was given), and the second eight weeks were the treatment period, during which patients received either a placebo or the drug progabide. The question of scientific interest is whether progabide is effective in reducing the rate of epileptic seizures.

The following DATA step creates the data set Seizure:

```
data Seizure;
   input ID Count Visit  Trt Age Weeks;
   datalines;
104 11 0 0 31 8
```

```
104 5 1 0 31 2
104 3 2 0 31 2
104 3 3 0 31 2
104 3 4 0 31 2
106 11 0 0 30 8

   ... more lines ...

236 12 0 1 37 8
236 1 1 1 37 2
236 4 2 1 37 2
236 3 3 1 37 2
236 2 4 1 37 2
;
```

The following DATA step creates a log time interval variable for use as an offset and an indicator variable for whether the observation is for a baseline measurement or a visit measurement. Patient 207 is deleted as an outlier, which was done in the Diggle et al. (2002) analysis:

```
data Seizure;
   set Seizure;
   if ID ne 207;
   if Visit = 0 then do;
      X1=0;
      Ltime = log(8);
   end;
   else do;
      X1=1;
      Ltime=log(2);
   end;
run;
```

Poisson regression is commonly used to model count data. In this example, the log-linear Poisson model is specified by $V(\mu) = \mu$ (the Poisson variance function) and a log link function,

$$\log(E(Y_{ij})) = \beta_0 + x_{i1}\beta_1 + x_{i2}\beta_2 + x_{i1}x_{i2}\beta_3 + \log(t_{ij})$$

where

$$Y_{ij} = \text{number of epileptic seizures in interval } j$$

$$t_{ij} = \text{length of interval } j$$

$$x_{i1} = \begin{cases} 1: & \text{weeks 8–16 (treatment)} \\ 0: & \text{weeks 0–8 (baseline)} \end{cases}$$

$$x_{i2} = \begin{cases} 1: & \text{progabide group} \\ 0: & \text{placebo group} \end{cases}$$

Because the visits represent repeated measurements, the responses from the same individual are correlated and inferences need to take this into account. The correlations between the counts are modeled as $r_{ij} = \alpha$, $i \neq j$ (exchangeable correlations).

In this model, the regression parameters are interpreted in terms of the log seizure rate that is displayed in Table 50.17.

Table 50.17 Interpretation of Regression Parameters

Treatment	Visit	$\log(E(Y_{ij})/t_{ij})$
Placebo	Baseline	β_0
	1–4	$\beta_0 + \beta_1$
Progabide	Baseline	$\beta_0 + \beta_2$
	1–4	$\beta_0 + \beta_1 + \beta_2 + \beta_3$

The difference between the log seizure rates in the pretreatment (baseline) period and the treatment periods is β_1 for the placebo group and $\beta_1 + \beta_3$ for the progabide group. A value of $\beta_3 < 0$ indicates a reduction in the seizure rate.

The following statements perform the analysis:

```
proc gee data = Seizure;
   class ID Visit;
   model Count = X1 Trt X1*Trt / dist=poisson link=log offset= Ltime;
   repeated subject = ID / within = Visit type=unstr covb corrw;
run;
```

In the MODEL statement, Count is the response variable, and X1, Trt, and the interaction X1*Trt are the explanatory variables. You request Poisson regression with the DIST=POISSON and the LINK=LOG options. The offset variable is often used in Poisson regression to account for different exposures. In this case, the OFFSET= option specifies Ltime as the offset variable representing different time intervals.

In the REPEATED statement, the SUBJECT= option indicates that the variable ID identifies the observations from a single cluster, and the TYPE=UNSTR option specifies the unstructured working correlation structure. The CORRW option requests that the working correlation matrix be displayed.

The "Model Information" table that is displayed in Output 50.2.1 provides information about the specified model and the input data set.

Output 50.2.1 Model Information

The GEE Procedure

Model Information	
Data Set	WORK.SEIZURE
Distribution	Poisson
Link Function	Log
Dependent Variable	Count
Offset Variable	Ltime

Output 50.2.2 displays general information about the GEE model analysis.

Output 50.2.2 GEE Model Information

GEE Model Information	
Correlation Structure	Unstructured
Within-Subject Effect	Visit (5 levels)
Subject Effect	ID (58 levels)
Number of Clusters	58
Correlation Matrix Dimension	5
Maximum Cluster Size	5
Minimum Cluster Size	5

Output 50.2.3 displays the parameter estimate covariance matrices, which are requested by the COVB option. Both model-based and empirical covariances are produced.

Output 50.2.3 Covariance Matrices of Parameter Estimate

Covariance Matrix (Model-Based)				
	Prm1	**Prm2**	**Prm3**	**Prm4**
Prm1	0.01210	0.004902	-0.01210	-0.004902
Prm2	0.004902	0.006660	-0.004902	-0.006660
Prm3	-0.01210	-0.004902	0.02461	0.01299
Prm4	-0.004902	-0.006660	0.01299	0.01852

Covariance Matrix (Empirical)				
	Prm1	**Prm2**	**Prm3**	**Prm4**
Prm1	0.02597	-0.003069	-0.02597	0.003069
Prm2	-0.003069	0.008597	0.003069	-0.008597
Prm3	-0.02597	0.003069	0.03841	-0.006196
Prm4	0.003069	-0.008597	-0.006196	0.02237

The exchangeable working correlation matrix is displayed in Output 50.2.4. It shows that there are noticeable correlations among the respective visits.

Output 50.2.4 Working Correlation Matrix

Working Correlation Matrix					
	Obs 1	**Obs 2**	**Obs 3**	**Obs 4**	**Obs 5**
Obs 1	1.0000	0.7920	0.7190	0.8111	0.6582
Obs 2	0.7920	1.0000	0.4859	0.6552	0.4566
Obs 3	0.7190	0.4859	1.0000	0.6988	0.4171
Obs 4	0.8111	0.6552	0.6988	1.0000	0.6464
Obs 5	0.6582	0.4566	0.4171	0.6464	1.0000

The parameter estimates table, shown in Output 50.2.5, contains parameter estimates, standard errors, confidence intervals, Z scores, and p-values for the parameter estimates. Empirical standard error estimates are used in this table.

Output 50.2.5 Parameter Estimates Table

Parameter Estimates for Response Model with Empirical Standard Error Estimates						
Parameter	Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept	1.3309	0.1612	1.0151	1.6468	8.26	<.0001
X1	0.1128	0.0927	-0.0689	0.2945	1.22	0.2237
Trt	-0.1034	0.1960	-0.4875	0.2807	-0.53	0.5978
X1*Trt	-0.3162	0.1496	-0.6093	-0.0231	-2.11	0.0345

The estimate of β_3 is –0.3162, which indicates that progabide is effective in reducing the rate of epileptic seizures.

Model fit criteria for the model are displayed in Output 50.2.6. These criteria are used in selecting regression models and working correlations.

Output 50.2.6 Model Fit Criteria

GEE Fit Criteria	
QIC	-1036.2782
QICu	-1041.7984

Example 50.3: Weighted GEE for Longitudinal Data That Have Missing Values

(View the complete code for this example (geeex3.sas) in the example repository.)

This example shows how you can use the GEE procedure to analyze longitudinal data that contain missing values. The data set is taken from a longitudinal study of women who used contraception during one year (Fitzmaurice, Laird, and Ware 2011). In this study, 1,151 women were randomly assigned to one of two treatments: 100 mg or 150 mg of depot medroxyprogesterone acetate (DMPA) at baseline and at three-month intervals. The response variable indicates their amenorrhea status during the four three-month intervals. The question of interest is whether the treatment has an effect on the rate of the amenorrhea over time. The example follows the analysis done by Fitzmaurice, Laird, and Ware (2011).

The following statements create the data set Amenorrhea:

```
data Amenorrhea;
   input ID Dose Time Y@@;
   datalines;
   1     0       1     0
   1     0       2     .
   1     0       3     .
   1     0       4     .

   ... more lines ...

1150     1       4     1
1151     1       1     1
```

```
1151     1       2     1
1151     1       3     1
1151     1       4     1
;
```

The variables in the data are as follows:

- ID: patient's ID
- Y: indicator of amenorrhea status (1 for amenorrhea; 0 otherwise)
- Time: four consecutive three-month intervals with values 1, 2, 3, and 4
- Dose: 0 for treatment with 100 mg injection; 1 for treatment with 150 mg injection

To prepare for the analysis, two additional variables are created:

- Prevy: the patient's amenorrhea status in the previous three-month interval. For the baseline visit, this is set to an arbitrary nonmissing value (0 here). In this release of PROC GEE, this arbitrary value must be nonmissing and valid for the response variable—for example, it should be 0 or 1 for a binary response—but it does not otherwise affect the results.
- Ctime: a copy of Time, which you can include in the marginal model as a continuous effect and also in the missingness model as a classification effect

The following statements add these two variables to the data set:

```
data Amenorrhea;
   set Amenorrhea;
   by ID;
   Prevy=lag(Y);
   if first.id then Prevy=0;
   Time=Time-1;
   Ctime=Time;
run;
```

Suppose y_{ij} denotes the amenorrhea status of woman i at the jth visit, $j = 1, \ldots, 4$, and suppose $\mu_{ij} = \mathrm{P}(y_{ij} = 1)$ denotes the average rate of high dosage. To explore whether the treatment has an effect on the rate of amenorrhea over time, consider the following marginal model:

$$\mathrm{logit}(\mu_{ij}) = \beta_0 + \beta_1 \mathrm{time}_{ij} + \beta_2 \mathrm{time}_{ij}^2 + \beta_3 \mathrm{dose}_i + \beta_4 \mathrm{dose}_i \times \mathrm{time} + \beta_5 \mathrm{dose}_i \times \mathrm{time}^2$$

Of the 1,151 women in this study, 576 are from the low-dose group, and 575 are from the high-dose group. For the low-dose group, 62.67% of the women completed the trial; for the high-dose group, 61.39% of the women completed this trial. Thus, both groups have substantial dropouts.

To obtain the weights for the weighted GEE analysis, consider the following logistic regression model for missingness:

$$\begin{aligned}\mathrm{logit}\, p(r_{ij} = 1 | r_{ij-1} = 1, \mathrm{dose}_i, \mathrm{ctime}_{ij}, y_{ij-1}) =& \alpha_0 + \alpha_1 I(\mathrm{ctime}_{ij} = 1) + \alpha_2 I(\mathrm{ctime}_{ij} = 2) \\ &+ \alpha_3 \mathrm{dose}_i + \alpha_4 y_{ij-1} + \alpha_5 \mathrm{dose}_i \times y_{ij-1}\end{aligned}$$

The following statements use the observation-specific weighted GEE method and the specified response and missingness models to analyze the data:

```
ods graphics on;
proc gee data=Amenorrhea desc plots=histogram;
   class ID Ctime;
   missmodel Ctime Prevy Dose Dose*Prevy / type=obslevel;
   model Y = Time Dose Time*Time Dose*Time Dose*Time*Time / dist=bin;
   repeated subject=ID / within=Ctime corr=cs;
run;
```

The MODEL statement specifies logistic regression and the model effects. The DESCEND option in the PROC GEE statement models the probability that Y = 1.

The REPEATED statement requests GEE analysis. The SUBJECT=ID option specifies that observations from the same subject are identified by ID. The TYPE=CS option specifies the compound symmetric working correlation structure.

The MISSMODEL statement requests the weighted GEE analysis. It specifies the logistic regression model for missingness. Note that no response variable is needed in weighted GEE analysis to specify a missingness model because the response is completely determined by the response variable in the MODEL statement. Without the MISSMODEL statement, PROC GEE would use the standard GEE approach, the same as provided by PROC GENMOD. The TYPE=OBSLEVEL option requests observation-specific weights.

Output 50.3.1 shows the parameter estimates for the missingness model. The estimate of α_4 is –0.4514 with a p-value of 0.0053, which suggests that the possibility that a participant will drop out is related to her previous amenorrhea status. This suggests that the assumption of MAR is more appropriate than that of MCAR.

Output 50.3.1 Parameter Estimates for the Missingness Model

Parameter Estimates for Missingness Model

Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept		2.3967	0.1438	2.1149	2.6785	16.67	<.0001
Ctime	0	0.0000	0.0000	0.0000	0.0000	.	.
Ctime	1	-0.7286	0.1439	-1.0106	-0.4466	-5.06	<.0001
Ctime	2	-0.5919	0.1469	-0.8798	-0.3040	-4.03	<.0001
Ctime	3	0.0000	0.0000	0.0000	0.0000	.	.
Prevy		-0.4514	0.1619	-0.7687	-0.1341	-2.79	0.0053
Dose		0.0680	0.1313	-0.1893	0.3253	0.52	0.6046
Prevy*Dose		-0.2381	0.2196	-0.6685	0.1923	-1.08	0.2782

The classification variable Ctime has two levels whose estimates are equal to zero. One is the reference level Ctime = 3. The first level, Ctime = 0, also has an estimate of zero, because the first visit is always observed and the first level is never used in estimating the weights in the missing model.

Output 50.3.2 displays the results of the weighted GEE analysis.

Output 50.3.2 Parameter Estimates for Amenorrhea Data Analysis Using Weighted GEE

The GEE Procedure

Parameter Estimates for Response Model with Empirical Standard Error Estimates						
Parameter	**Estimate**	**Standard Error**	**95% Confidence Limits**		**Z**	**Pr > |Z|**
Intercept	-1.4965	0.1072	-1.7067	-1.2863	-13.95	<.0001
Time	0.5379	0.1334	0.2764	0.7994	4.03	<.0001
Dose	0.1061	0.1491	-0.1861	0.3983	0.71	0.4767
Time*Time	-0.0037	0.0405	-0.0831	0.0757	-0.09	0.9275
Dose*Time	0.4092	0.1903	0.0362	0.7823	2.15	0.0315
Dose*Time*Time	-0.1264	0.0577	-0.2395	-0.0134	-2.19	0.0284

The estimate of β_4 (the parameter estimate for the Dose*Time interaction) is 0.4092, which indicates that the change of amenorrhea rate over time depends on the dose of DMPA. Specifically, for women in the low-dose group, the amenorrhea rates μ_{ij} at the four consecutive time intervals are 0.1830, 0.2764, 0.3928, and 0.5210 and for women in the high-dose group, the amenorrhea rate are 0.1997, 0.3609, 0.4963, and 0.5701. In other words, the amenorrhea rate increases over time for both treatments, and the rates of increase are slightly different.

You can request subject-level weights by specifying the TYPE=SUBLEVEL option. The results (not shown here) from the subject-level weighted method are similar to the results from the observation-level weighted method. Both of the weighted GEE methods provide unbiased regression parameter estimates if the missingness model is specified correctly. Preisser, Lohman, and Rathouz (2002) note that the observation-level weighted GEE produces more efficient estimates than the cluster-level weighted GEE produces for incomplete longitudinal binary data.

Large weights can have impacts on the parameter estimates. Consequently, it is recommended that you check the distribution of the estimated weights. If there are large weights, you might consider trimming them by specifying the MAXWEIGHT= option in the MISSMODEL statement. Output 50.3.3 shows that the estimated weights in this example range between 1 and 2.1, so no trimming is needed.

Output 50.3.3 Histogram of Estimated Weights

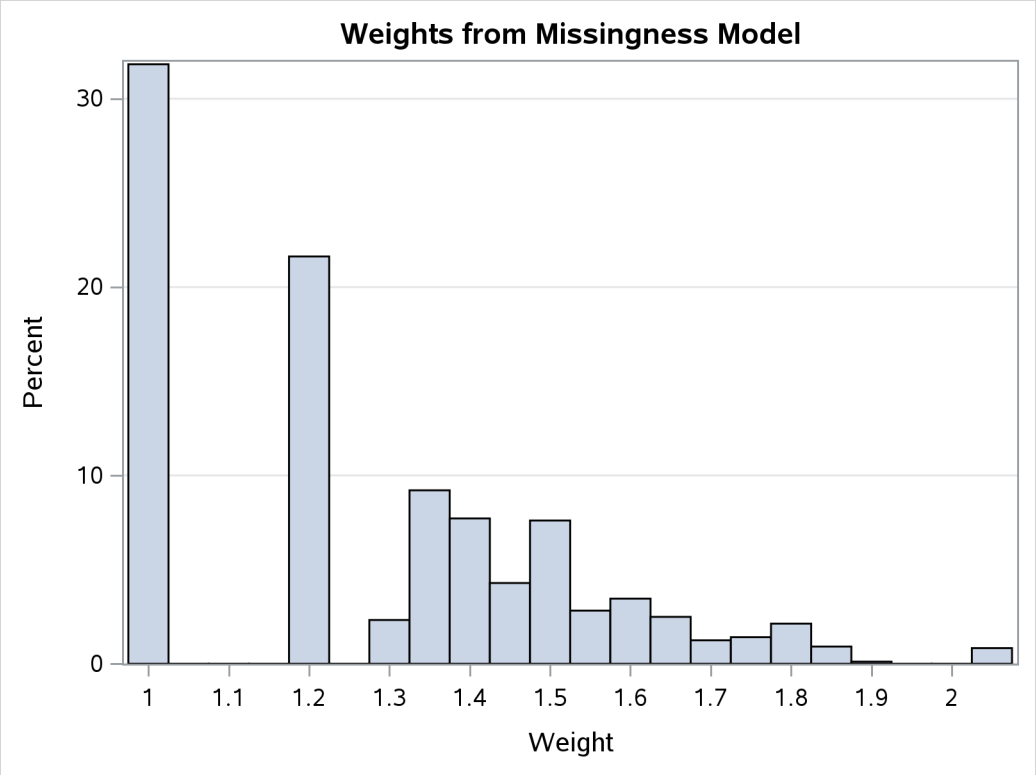

Example 50.4: GEE for Binary Data with Logit Link Function

(View the complete code for this example (geeex4.sas) in the example repository.)

Because the respiratory data in Example 50.1 are binary, you can use the alternating logistic regression (ALR) method and model associations by using the log odds ratios instead of working correlations. This example fits a "fully parameterized cluster" model for the log odds ratio. That is, there is a log odds ratio parameter for each unique pair of responses within clusters, and all clusters are parameterized identically. The following statements fit the same regression model for the mean as in Example 50.1 but use a regression model for the log odds ratios instead of a working correlation. LOGOR=FULLCLUST specifies a fully parameterized log odds ratio model.

```
proc gee data=Resp descend;
   class ID Treatment Center Sex Baseline;
   model Outcome=Treatment Center Sex Age Baseline / dist=bin;
   repeated  subject=ID(Center) / logor=fullclust;
run;
```

The results of fitting the model are displayed in Output 50.4.1.

Output 50.4.1 Results of ALR Model Fitting

The GEE Procedure

Parameter Estimates for Response Model with Empirical Standard Error Estimates							
Parameter		**Estimate**	**Standard Error**	**95% Confidence Limits**		**Z**	**Pr > |Z|**
Intercept		1.6001	0.5128	0.5950	2.6052	3.12	0.0018
Treatment	A	1.2611	0.3406	0.5934	1.9287	3.70	0.0002
Treatment	P	0.0000	0.0000	0.0000	0.0000	.	.
Center	1	-0.6287	0.3486	-1.3119	0.0545	-1.80	0.0713
Center	2	0.0000	0.0000	0.0000	0.0000	.	.
Sex	F	0.1024	0.4362	-0.7526	0.9575	0.23	0.8144
Sex	M	0.0000	0.0000	0.0000	0.0000	.	.
Age		-0.0162	0.0125	-0.0407	0.0084	-1.29	0.1977
Baseline	0	-1.8980	0.3404	-2.5652	-1.2308	-5.58	<.0001
Baseline	1	0.0000	0.0000	0.0000	0.0000	.	.
Alpha1		1.6109	0.4892	0.6522	2.5696	3.29	0.0010
Alpha2		1.0771	0.4834	0.1297	2.0246	2.23	0.0259
Alpha3		1.5875	0.4735	0.6594	2.5155	3.35	0.0008
Alpha4		2.1224	0.5022	1.1381	3.1068	4.23	<.0001
Alpha5		1.8818	0.4686	0.9634	2.8001	4.02	<.0001
Alpha6		2.1046	0.4949	1.1347	3.0745	4.25	<.0001

The parameters Alpha1 through Alpha6 estimate the log odds ratio for each unique within-cluster pair. The correspondence between the log odds ratio parameters and within-cluster pairs is displayed in Output 50.4.2.

Output 50.4.2 Log Odds Ratio Parameters

Log Odds Ratio Parameter Information	
Parameter	**Group**
Alpha1	(1, 2)
Alpha2	(1, 3)
Alpha3	(1, 4)
Alpha4	(2, 3)
Alpha5	(2, 4)
Alpha6	(3, 4)

Model goodness-of-fit criteria are shown in Output 50.4.3.

Output 50.4.3 ALR Model Fit Criteria

GEE Fit Criteria	
QIC	511.8589
QICu	499.6516

The QIC for the ALR model shown in Output 50.4.3 is 511.86, whereas the QIC for the unstructured working correlation model shown in Output 50.1.3 is 512.34, indicating that the ALR model has a slightly better fit.

You can fit the same model by fully specifying the **z** matrix; for the definition of the **z** matrix, see the section "Specifying Log Odds Ratio Models" on page 3535. The following statements create a data set that contains the full **z** matrix:

```
data zin;
   keep id center z1-z6 y1 y2;
   array zin(6) z1-z6;
   set resp;
   by center id;
   if first.id
      then do;
      t = 0;
      do m = 1 to 4;
         do n = m+1 to 4;
            do j = 1 to 6;
               zin(j) = 0;
            end;
            y1 = m;
            y2 = n;
            t + 1;
            zin(t) = 1;
            output;
         end;
      end;
   end;
run;

proc print data=zin (obs=12);
run;
```

Output 50.4.4 displays the full **z** matrix for the first two clusters. The **z** matrix is identical for all clusters in this example.

Output 50.4.4 Full **z** Matrix Data Set

Obs	z1	z2	z3	z4	z5	z6	Center	ID	y1	y2
1	1	0	0	0	0	0	1	1	1	2
2	0	1	0	0	0	0	1	1	1	3
3	0	0	1	0	0	0	1	1	1	4
4	0	0	0	1	0	0	1	1	2	3
5	0	0	0	0	1	0	1	1	2	4
6	0	0	0	0	0	1	1	1	3	4
7	1	0	0	0	0	0	1	2	1	2
8	0	1	0	0	0	0	1	2	1	3
9	0	0	1	0	0	0	1	2	1	4
10	0	0	0	1	0	0	1	2	2	3
11	0	0	0	0	1	0	1	2	2	4
12	0	0	0	0	0	1	1	2	3	4

The following statements fit the model for fully parameterized clusters by fully specifying the **z** matrix. The

results are identical to those shown previously.

```
proc gee data=Resp descend;
   class ID Treatment Center Sex Baseline;
   model Outcome=Treatment Center Sex Age Baseline / dist=bin;
   repeated  subject=ID(Center) / logor=zfull
                                  zdata=zin
                                  zrow =(z1-z6)
                                  ypair=(y1 y2);
run;
```

Example 50.5: Alternating Logistic Regression for Ordinal Multinomial Data

(View the complete code for this example (geeex5.sas) in the example repository.)

This example illustrates how you use the GEE procedure and alternating logistic regression (ALR) to analyze ordinal multinomial data. A clinical trial was conducted to evaluate the effectiveness of the drug auranofin for treating arthritis (Lipsitz, Kim, and Zhao 1994). Patients were assigned to one of two groups: one group was treated with auranofin, and the other group received a placebo. The treatment that a patient received is recorded in the variable Treatment (coded here as 1 = auranofin and 0 = placebo).

The response was self-assessment of arthritis recorded at one-, three-, and five-month follow-up visits. The responses are recorded in the Rating variable and are coded as 1 = very poor, 2 = poor, 3 = fair, 4 = good, and 5 = very good. This coding of Rating is finer than the coding in Lipsitz, Kim, and Zhao (1994), where only three levels were used. The visit numbers are recorded in the classification variable Visit, whose value is 1, 3, or 5. An initial self-assessment that uses the same coding as Rating is recorded in the variable Baseline. The variable Age records the participants' ages (in years) at the baseline visit and is treated as a continuous variable. One participant missed all visits and is not considered. There are an additional 15 missed visits from eight participants who dropped out, and there are four participants who missed a single visit. A weighted GEE is not used because the GEE procedure does not support the weighted GEE method for the multinomial distribution.

The following DATA step creates the data set Arthritis:

```
data Arthritis;
   input ID Rating Sex Age Treatment Baseline Visit;
   datalines;
1 4 2 54 2 2 1
1 5 2 54 2 2 3
1 5 2 54 2 2 5
2 4 1 41 1 3 1
2 4 1 41 1 3 3
2 4 1 41 1 3 5

   ... more lines ...

301 2 2 64 1 2 5
302 2 2 55 1 2 1
302 3 2 55 1 2 3
302 3 2 55 1 2 5
;
```

The following SAS statements use PROC GEE to fit a model that has a fully exchangeable working correlation structure:

```
proc gee data=Arthritis;
   class Sex ID Treatment Baseline Visit;
   model Rating= Visit Treatment Baseline / dist=multinomial;
   repeated subject=ID / within=Visit logor=exch;
run;
```

You specify LOGOR=EXCH in the REPEATED statement to select the ALR method that has a fully exchangeable model for the log odds ratio. The results of the ALR model fitting are displayed in Output 50.5.1.

Output 50.5.1 Parameter Estimates for Arthritis Data Using ALR

The GEE Procedure

Parameter Estimates for Response Model with Empirical Standard Error Estimates

Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept1		-6.7502	0.4267	-7.5865	-5.9138	-15.82	<.0001
Intercept2		-4.6310	0.3968	-5.4087	-3.8533	-11.67	<.0001
Intercept3		-2.6735	0.3749	-3.4083	-1.9387	-7.13	<.0001
Intercept4		-0.3838	0.3710	-1.1109	0.3433	-1.03	0.3008
Visit	1	0.3740	0.1148	0.1489	0.5991	3.26	0.0011
Visit	3	0.3641	0.1116	0.1455	0.5828	3.26	0.0011
Visit	5	0.0000	0.0000	0.0000	0.0000	.	.
Treatment	1	0.5552	0.1673	0.2273	0.8830	3.32	0.0009
Treatment	2	0.0000	0.0000	0.0000	0.0000	.	.
Baseline	1	3.9457	0.5352	2.8969	4.9946	7.37	<.0001
Baseline	2	3.3052	0.4268	2.4686	4.1418	7.74	<.0001
Baseline	3	2.7483	0.3790	2.0054	3.4911	7.25	<.0001
Baseline	4	1.4013	0.4132	0.5914	2.2113	3.39	0.0007
Baseline	5	0.0000	0.0000	0.0000	0.0000	.	.
Alpha1		1.6447	0.1693	1.3130	1.9764	9.72	<.0001

The parameter Alpha1, which is used to estimate the log odds ratio, is included in Output 50.5.1.

To fit the ALR model, each response is coded as a vector of binary variables and the log odds ratio models the association between pairs of responses. For more information about the log odds ratio and the ALR method for ordinal multinomial data, see the section "ALR for Ordinal Multinomial Data" on page 3535. The ALR model fit criteria are shown in Output 50.5.2.

Output 50.5.2 ALR Model Fit Criteria

GEE Fit Criteria	
QIC	2241.9540
QICu	2259.8575

For comparison, the following SAS statements use PROC GEE to fit the same marginal model by using an independent working correlation structure:

```
proc gee data=Arthritis;
   class Sex ID Treatment Baseline Visit;
   model Rating= Visit Treatment Baseline / dist=multinomial;
   repeated subject=ID / within=Visit;
run;
```

When the data have multinomial responses, the independent working correlation structure is the only structure supported for ordinary GEEs. In Output 50.5.1 and Output 50.5.3, you can see slight differences in the parameter estimates between the model that you fit by using ALR and the model that you fit by using an independent working correlation structure.

Output 50.5.3 Parameter Estimates for Arthritis Data Using Independent Working Correlation

The GEE Procedure

Parameter Estimates for Response Model with Empirical Standard Error Estimates

Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept1		-6.7528	0.4227	-7.5812	-5.9244	-15.98	<.0001
Intercept2		-4.6719	0.3953	-5.4466	-3.8972	-11.82	<.0001
Intercept3		-2.7138	0.3730	-3.4449	-1.9828	-7.28	<.0001
Intercept4		-0.4129	0.3689	-1.1360	0.3102	-1.12	0.2631
Visit	1	0.3852	0.1160	0.1578	0.6125	3.32	0.0009
Visit	3	0.3725	0.1118	0.1534	0.5916	3.33	0.0009
Visit	5	0.0000	0.0000	0.0000	0.0000	.	.
Treatment	1	0.5643	0.1679	0.2352	0.8933	3.36	0.0008
Treatment	2	0.0000	0.0000	0.0000	0.0000	.	.
Baseline	1	3.9533	0.5351	2.9046	5.0020	7.39	<.0001
Baseline	2	3.3264	0.4250	2.4934	4.1593	7.83	<.0001
Baseline	3	2.7672	0.3769	2.0285	3.5059	7.34	<.0001
Baseline	4	1.4252	0.4112	0.6192	2.2312	3.47	0.0005
Baseline	5	0.0000	0.0000	0.0000	0.0000	.	.

The QIC for the ALR model shown in Output 50.5.2 is 2241.95, whereas the QIC for the independent working correlation model shown in Output 50.5.4 is 2269.82, indicating a slightly better fit for the ALR model.

Output 50.5.4 Model Fit Criteria

GEE Fit Criteria	
QIC	2269.8166
QICu	2259.7693

Example 50.6: GEE for Nominal Multinomial Data

(View the complete code for this example (geeex6.sas) in the example repository.)

This example illustrates how you use the GEE procedure to analyze nominal multinomial data. A two-year study was conducted to assess the impact of access to Section 8 housing as a means of providing independent housing to the severely mentally ill homeless (Hurlbut, Wood, and Hough 1996). In this study, half of the 362 clients received Section 8 housing certificates. The assignment of Section 8 housing certificates is recorded in the variable Sec; 0 indicates clients who did not receive a certificate, and 1 indicates clients who received a certificate.

Every six months during the study, research staff interviewed all 362 clients, who provided data about their living arrangements in the previous 60 days. Clients' living arrangements were also recorded during a baseline interview. The time of interviews is recorded in the variable Time, whose value is 0, 6, 12, or 24 (for the number of months since the study began). There were a total of 159 missed interviews. The variable Housing records the living arrangement of a client and is coded as 0 (street living), 1 (community living), or 2 (independent living). The following statements create the data set Housing:

```
data Housing;
   input ID Housing Time Sec;
   datalines;
1 1 0 1
1 2 6 1
1 2 12 1
1 2 24 1
2 1 0 1
2 2 6 1

   ... more lines ...

362 1 0 0
362 1 6 0
362 1 12 0
362 1 24 0
;
```

The following SAS statements use PROC GEE to fit a model to nominal multinomial data:

```
proc gee data=Housing;
   class ID Housing Time SEC;
   model Housing=Sec / dist=multinomial link=glogit;
   repeated subject=ID / within=Time;
run;
```

An ordinary GEE that has an independent working correlation structure is fit. This model is the only option supported for data that have nominal multinomial responses. In the MODEL statement, you specify LINK=GLOGIT to indicate that the responses are nominal. In the generalized logit model, you model baseline category logits. By default, the GEE procedure chooses the last response category as the baseline category. If your nominal response has J categories, then the baseline logit for category j and subject i is

$$\log(\mu_{ij}/\mu_{iJ}) = \eta_{ij} = \mathbf{x}_i'\boldsymbol{\beta}_j$$

and

$$\mu_{ij} = \frac{\exp(\eta_{ij})}{\sum_{k=1}^{J} \exp(\eta_{ik})}$$

$$\eta_{iJ} = 0$$

The results of fitting the model are displayed in Output 50.6.1.

Output 50.6.1 Results of Model Fitting

The GEE Procedure

Parameter Estimates for Response Model with Empirical Standard Error Estimates

Parameter	Housing	Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept	**0**	-0.9532	0.1266	-1.2013	-0.7051	-7.53	<.0001
Intercept	**1**	-0.6562	0.1064	-0.8647	-0.4477	-6.17	<.0001
Sec 0	**0**	0.9226	0.1850	0.5599	1.2853	4.99	<.0001
Sec 0	**1**	1.2645	0.1642	0.9426	1.5863	7.70	<.0001
Sec 1	**0**	0.0000	0.0000	0.0000	0.0000	.	.
Sec 1	**1**	0.0000	0.0000	0.0000	0.0000	.	.

The positive estimates for the classification variable Sec = 0 at each response category, Housing = 0 and 1, indicate an increased probability that a client will live independently when given access to Section 8 housing. The model fit criteria are shown in Output 50.6.2

Output 50.6.2 Model Fit Criteria

GEE Fit Criteria	
QIC	2675.2174
QICu	2671.4680

For comparison, the following SAS statements treat the responses as ordinal and use PROC GEE to fit a marginal model by using an independent working correlation structure:

```
proc gee data=Housing;
   class ID Housing Time SEC;
   model Housing=Sec / dist=multinomial;
   repeated subject=ID / within=Time;
run;
```

The cumulative logit link function is the default option that is used to fit the model. Because the generalized logit link function is not specified, the responses are treated as ordinal multinomial data. The results for the model that is fit by treating the responses as ordinal are displayed in Output 50.6.3.

Output 50.6.3 Results of Model Fitting

The GEE Procedure

Parameter Estimates for Response Model with Empirical Standard Error Estimates							
Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept1		-1.6917	0.1242	-1.9352	-1.4481	-13.62	<.0001
Intercept2		0.0112	0.0960	-0.1770	0.1994	0.12	0.9072
Sec	0	0.8224	0.1327	0.5624	1.0824	6.20	<.0001
Sec	1	0.0000	0.0000	0.0000	0.0000	.	.

Treating the responses as ordinal results in a single parameter estimate that is related to the classification variable Sec. The QIC for the model that is fit by treating the responses as nominal (shown in Output 50.6.2) is 2675.21, whereas the QIC for the model that is fit by treating the responses as ordinal (shown in Output 50.6.4) is 2710.50, indicating a slightly better fit when the responses are treated as nominal.

Output 50.6.4 Model Fit Criteria

GEE Fit Criteria	
QIC	2710.4971
QICu	2707.2983

References

Boos, D. (1992). "On Generalized Score Tests." *American Statistician* 46:327–333.

Carey, V., Zeger, S. L., and Diggle, P. J. (1993). "Modelling Multivariate Binary Data with Alternating Logistic Regressions." *Biometrika* 80:517–526.

Chang, I.-M., Gelman, R., and Pagano, M. (1982). "Corrected Group Prognostic Curves and Summary Statistics." *Journal of Chronic Diseases* 35:669–674.

Diggle, P. J., Heagerty, P., Liang, K.-Y., and Zeger, S. L. (2002). *Analysis of Longitudinal Data*. 2nd ed. New York: Oxford University Press.

Diggle, P. J., Liang, K.-Y., and Zeger, S. L. (1994). *Analysis of Longitudinal Data*. Oxford: Clarendon Press.

Fitzmaurice, G. M., Laird, N. M., and Ware, J. H. (2011). *Applied Longitudinal Analysis*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Fitzmaurice, G. M., Molenberghs, G., and Lipsitz, S. R. (1995). "Regression Models for Longitudinal Binary Responses with Informative Drop-Outs." *Journal of the Royal Statistical Society, Series B* 57:691–704.

Hardin, J. W., and Hilbe, J. M. (2003). *Generalized Estimating Equations*. Boca Raton, FL: Chapman & Hall/CRC.

Heagerty, P., and Zeger, S. L. (1996). "Marginal Regression Models for Clustered Ordinal Measurements." *Journal of the American Statistical Association* 91:1024–1036.

Hurlbut, M. S., Wood, P. A., and Hough, R. L. (1996). "Providing Independent Housing for the Homeless Mentally Ill: A Novel Approach to Evaluating Long-Term Longitudinal Housing Patterns." *Journal of Community Psychology* 24:291–310.

Lane, P. W., and Nelder, J. A. (1982). "Analysis of Covariance and Standardization as Instances of Prediction." *Biometrics* 38:613–621.

Liang, K.-Y., and Zeger, S. L. (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika* 73:13–22.

Lipsitz, S. R., Fitzmaurice, G. M., Orav, E. J., and Laird, N. M. (1994). "Performance of Generalized Estimating Equations in Practical Situations." *Biometrics* 50:270–278.

Lipsitz, S. R., Kim, K., and Zhao, L. (1994). "Analysis of Repeated Categorical Data Using Generalized Estimating Equations." *Statistics in Medicine* 13:1149–1163.

Littell, R. C., Freund, R. J., and Spector, P. C. (1991). *SAS System for Linear Models*. 3rd ed. Cary, NC: SAS Institute Inc.

Mallinckrodt, C. (2013). *Preventing and Treating Missing Data in Longitudinal Clinical Trials: A Practical Guide*. Cambridge: Cambridge University Press.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Molenberghs, G., and Kenward, M. G. (2007). *Missing Data in Clinical Studies*. New York: John Wiley & Sons.

O'Kelly, M., and Ratitch, B. (2014). *Clinical Trials with Missing Data: A Guide for Practitioners*. Chichester, UK: John Wiley & Sons.

Pan, W. (2001). "Akaike's Information Criterion in Generalized Estimating Equations." *Biometrics* 57:120–125.

Preisser, J. S., Lohman, K. K., and Rathouz, P. J. (2002). "Performance of Weighted Estimating Equations for Longitudinal Binary Data with Drop-Outs Missing at Random." *Statistics in Medicine* 21:3035–3054.

Robins, J. M., and Rotnitzky, A. (1995). "Semiparametric Efficiency in Multivariate Regression Models with Missing Data." *Journal of the American Statistical Association* 90:122–129.

Rotnitzky, A., and Jewell, N. P. (1990). "Hypothesis Testing of Regression Parameters in Semiparametric Generalized Linear Models for Cluster Correlated Data." *Biometrika* 77:485–497.

Rubin, D. B. (1976). "Inference and Missing Data." *Biometrika* 63:581–592.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2012). *Categorical Data Analysis Using SAS*. 3rd ed. Cary, NC: SAS Institute Inc.

Chapter 51
The GENMOD Procedure

Contents

Overview: GENMOD Procedure . . . **3569**
- What Is a Generalized Linear Model? . . . 3570
- Examples of Generalized Linear Models . . . 3571
- The GENMOD Procedure . . . 3572

Getting Started: GENMOD Procedure . . . **3575**
- Poisson Regression . . . 3575
- Bayesian Analysis of a Linear Regression Model . . . 3580
- Generalized Estimating Equations . . . 3591

Syntax: GENMOD Procedure . . . **3594**
- PROC GENMOD Statement . . . 3595
- ASSESS Statement . . . 3600
- BAYES Statement . . . 3601
- BY Statement . . . 3611
- CLASS Statement . . . 3612
- CODE Statement . . . 3615
- CONTRAST Statement . . . 3616
- DEVIANCE Statement . . . 3618
- EFFECTPLOT Statement . . . 3619
- ESTIMATE Statement . . . 3620
- EXACT Statement . . . 3621
- EXACTOPTIONS Statement . . . 3624
- FREQ Statement . . . 3627
- FWDLINK Statement . . . 3627
- INVLINK Statement . . . 3627
- LSMEANS Statement . . . 3628
- LSMESTIMATE Statement . . . 3629
- MARGINS Statement . . . 3630
- MODEL Statement . . . 3631
- OUTPUT Statement . . . 3642
- Programming Statements . . . 3645
- REPEATED Statement . . . 3647
- SLICE Statement . . . 3652
- STORE Statement . . . 3652
- STRATA Statement . . . 3652
- VARIANCE Statement . . . 3654
- WEIGHT Statement . . . 3654

ZEROMODEL Statement . . . 3654
Details: GENMOD Procedure . . . **3655**
Generalized Linear Models Theory . . . 3655
Specification of Effects . . . 3666
Parameterization Used in PROC GENMOD . . . 3667
Type 1 Analysis . . . 3668
Type 3 Analysis . . . 3669
Confidence Intervals for Parameters . . . 3670
F Statistics . . . 3671
Lagrange Multiplier Statistics . . . 3672
Predicted Values of the Mean . . . 3672
Residuals . . . 3673
Multinomial Models . . . 3674
Zero-Inflated Models . . . 3675
Tweedie Distribution for Generalized Linear Models . . . 3676
Generalized Estimating Equations . . . 3678
Assessment of Models Based on Aggregates of Residuals . . . 3687
Case Deletion Diagnostic Statistics . . . 3690
Bayesian Analysis . . . 3694
Exact Logistic and Exact Poisson Regression . . . 3699
Response Level Ordering . . . 3702
Missing Values . . . 3703
Displayed Output for Classical Analysis . . . 3703
Displayed Output for Bayesian Analysis . . . 3711
Displayed Output for Exact Analysis . . . 3713
ODS Table Names . . . 3714
ODS Graphics . . . 3718
Examples: GENMOD Procedure . . . **3720**
Example 51.1: Logistic Regression . . . 3720
Example 51.2: Normal Regression, Log Link . . . 3722
Example 51.3: Gamma Distribution Applied to Life Data . . . 3724
Example 51.4: Ordinal Model for Multinomial Data . . . 3727
Example 51.5: GEE for Binary Data with Logit Link Function . . . 3730
Example 51.6: Log Odds Ratios and the ALR Algorithm . . . 3733
Example 51.7: Log-Linear Model for Count Data . . . 3736
Example 51.8: Model Assessment of Multiple Regression Using Aggregates of Residuals 3740
Example 51.9: Assessment of a Marginal Model for Dependent Data . . . 3748
Example 51.10: Bayesian Analysis of a Poisson Regression Model . . . 3751
Example 51.11: Exact Poisson Regression . . . 3765
Example 51.12: Tweedie Regression . . . 3768
References . . . **3770**

Overview: GENMOD Procedure

The GENMOD procedure fits generalized linear models, as defined by Nelder and Wedderburn (1972). The class of generalized linear models is an extension of traditional linear models that allows the mean of a population to depend on a *linear predictor* through a nonlinear *link function* and allows the response probability distribution to be any member of an exponential family of distributions. Many widely used statistical models are generalized linear models. These include classical linear models with normal errors, logistic and probit models for binary data, and log-linear models for multinomial data. Many other useful statistical models can be formulated as generalized linear models by the selection of an appropriate link function and response probability distribution.

See McCullagh and Nelder (1989) for a discussion of statistical modeling using generalized linear models. The books by Aitkin et al. (1989) and Dobson (1990) are also excellent references with many examples of applications of generalized linear models. Firth (1991) provides an overview of generalized linear models. Myers, Montgomery, and Vining (2002) provide applications of generalized linear models in the engineering and physical sciences. Collett (2003) and Hilbe (2009) provide comprehensive accounts of generalized linear models when the responses are binary.

The analysis of correlated data arising from repeated measurements when the measurements are assumed to be multivariate normal has been studied extensively. However, the normality assumption might not always be reasonable; for example, different methodology must be used in the data analysis when the responses are discrete and correlated. Generalized estimating equations (GEEs) provide a practical method with reasonable statistical efficiency to analyze such data.

Liang and Zeger (1986) introduced GEEs as a method of dealing with correlated data when, except for the correlation among responses, the data can be modeled as a generalized linear model. For example, correlated binary and count data in many cases can be modeled in this way.

The GENMOD procedure can fit models to correlated responses by the GEE method. You can use PROC GENMOD to fit models with most of the correlation structures from Liang and Zeger (1986) by using GEEs. For more details on GEEs, see Hardin and Hilbe (2003); Diggle, Liang, and Zeger (1994); Lipsitz et al. (1994).

Bayesian analysis of generalized linear models can be requested by using the BAYES statement in the GENMOD procedure. In Bayesian analysis, the model parameters are treated as random variables, and inference about parameters is based on the posterior distribution of the parameters, given the data. The posterior distribution is obtained using Bayes' theorem as the likelihood function of the data weighted with a prior distribution. The prior distribution enables you to incorporate knowledge or experience of the likely range of values of the parameters of interest into the analysis. If you have no prior knowledge of the parameter values, you can use a noninformative prior distribution, and the results of the Bayesian analysis will be very similar to a classical analysis based on maximum likelihood. A closed form of the posterior distribution is often not feasible, and a Markov chain Monte Carlo method by Gibbs sampling is used to simulate samples from the posterior distribution. See Chapter 8, "Introduction to Bayesian Analysis Procedures," for an introduction to the basic concepts of Bayesian statistics. Also see the section "Bayesian Analysis: Advantages and Disadvantages" on page 154 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a discussion of the advantages and disadvantages of Bayesian analysis. See Ibrahim, Chen, and Sinha (2001) for a detailed description of Bayesian analysis.

In a Bayesian analysis, a Gibbs chain of samples from the posterior distribution is generated for the model parameters. Summary statistics (mean, standard deviation, quartiles, HPD and credible intervals,

correlation matrix) and convergence diagnostics (autocorrelations; Gelman-Rubin, Geweke, Raftery-Lewis, and Heidelberger and Welch tests; the effective sample size; and Monte Carlo standard errors) are computed for each parameter, as well as the correlation matrix and the covariance matrix of the posterior sample. Trace plots, posterior density plots, and autocorrelation function plots that are created using ODS Graphics are also provided for each parameter.

The GENMOD procedure enables you to perform exact logistic regression, also called exact conditional binary logistic regression, and exact Poisson regression, also called exact conditional Poisson regression, by specifying one or more EXACT statements. You can test individual parameters or conduct a joint test for several parameters. The procedure computes two exact tests: the exact conditional score test and the exact conditional probability test. You can request exact estimation of specific parameters and corresponding odds ratios where appropriate. Point estimates, standard errors, and confidence intervals are provided.

The GENMOD procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

What Is a Generalized Linear Model?

A traditional linear model is of the form

$$y_i = \mathbf{x}_i'\boldsymbol{\beta} + \varepsilon_i$$

where y_i is the response variable for the ith observation. The quantity $\mathbf{x}_i$ is a column vector of covariates, or explanatory variables, for observation i that is known from the experimental setting and is considered to be fixed, or nonrandom. The vector of unknown coefficients $\boldsymbol{\beta}$ is estimated by a least squares fit to the data $\mathbf{y}$. The ε_i are assumed to be independent, normal random variables with zero mean and constant variance. The expected value of y_i, denoted by μ_i, is

$$\mu_i = \mathbf{x}_i'\boldsymbol{\beta}$$

While traditional linear models are used extensively in statistical data analysis, there are types of problems such as the following for which they are not appropriate.

- It might not be reasonable to assume that data are normally distributed. For example, the normal distribution (which is continuous) might not be adequate for modeling counts or measured proportions that are considered to be discrete.
- If the mean of the data is naturally restricted to a range of values, the traditional linear model might not be appropriate, since the linear predictor $\mathbf{x}_i'\boldsymbol{\beta}$ can take on any value. For example, the mean of a measured proportion is between 0 and 1, but the linear predictor of the mean in a traditional linear model is not restricted to this range.
- It might not be realistic to assume that the variance of the data is constant for all observations. For example, it is not unusual to observe data where the variance increases with the mean of the data.

A generalized linear model extends the traditional linear model and is therefore applicable to a wider range of data analysis problems. A generalized linear model consists of the following components:

- The linear component is defined just as it is for traditional linear models:

  $$\eta_i = \mathbf{x}_i'\boldsymbol{\beta}$$

- A monotonic differentiable link function g describes how the expected value of y_i is related to the linear predictor η_i:

  $$g(\mu_i) = \mathbf{x}_i'\boldsymbol{\beta}$$

- The response variables y_i are independent for $i = 1, 2, \ldots$ and have a probability distribution from an exponential family. This implies that the variance of the response depends on the mean μ through a *variance function* V:

  $$\mathrm{Var}(y_i) = \frac{\phi V(\mu_i)}{w_i}$$

  where ϕ is a constant and w_i is a known weight for each observation. The *dispersion parameter* ϕ is either known (for example, for the binomial or Poisson distribution, $\phi = 1$) or must be estimated.

See the section "Response Probability Distributions" on page 3655 for the form of a probability distribution from the exponential family of distributions.

As in the case of traditional linear models, fitted generalized linear models can be summarized through statistics such as parameter estimates, their standard errors, and goodness-of-fit statistics. You can also make statistical inference about the parameters by using confidence intervals and hypothesis tests. However, specific inference procedures are usually based on asymptotic considerations, since exact distribution theory is not available or is not practical for all generalized linear models.

Examples of Generalized Linear Models

You construct a generalized linear model by deciding on response and explanatory variables for your data and choosing an appropriate link function and response probability distribution. Some examples of generalized linear models follow. Explanatory variables can be any combination of continuous variables, classification variables, and interactions.

Traditional Linear Model

- response variable: a continuous variable
- distribution: normal
- link function: identity, $g(\mu) = \mu$

Logistic Regression

- response variable: a proportion
- distribution: binomial
- link function: logit, $g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$

Poisson Regression in Log-Linear Model

- response variable: a count
- distribution: Poisson
- link function: log, $g(\mu) = \log(\mu)$

Gamma Model with Log Link

- response variable: a positive, continuous variable
- distribution: gamma
- link function: log, $g(\mu) = \log(\mu)$

The GENMOD Procedure

The GENMOD procedure fits a generalized linear model to the data by maximum likelihood estimation of the parameter vector $\boldsymbol{\beta}$. There is, in general, no closed form solution for the maximum likelihood estimates of the parameters. The GENMOD procedure estimates the parameters of the model numerically through an iterative fitting process. The dispersion parameter ϕ is also estimated by maximum likelihood or, optionally, by the residual deviance or by Pearson's chi-square divided by the degrees of freedom. Covariances, standard errors, and p-values are computed for the estimated parameters based on the asymptotic normality of maximum likelihood estimators. A number of popular link functions and probability distributions are available in the GENMOD procedure. The built-in link functions are as follows:

- identity: $g(\mu) = \mu$
- logit: $g(\mu) = \log(\mu/(1-\mu))$
- probit: $g(\mu) = \Phi^{-1}(\mu)$, where Φ is the standard normal cumulative distribution function
- power: $g(\mu) = \begin{cases} \mu^{\lambda} & \text{if } \lambda \neq 0 \\ \log(\mu) & \text{if } \lambda = 0 \end{cases}$
- log: $g(\mu) = \log(\mu)$
- complementary log-log: $g(\mu) = \log(-\log(1-\mu))$

The available distributions and associated variance functions are as follows:

- normal: $V(\mu) = 1$
- binomial (proportion): $V(\mu) = \mu(1-\mu)$
- Poisson: $V(\mu) = \mu$

- gamma: $V(\mu) = \mu^2$
- inverse Gaussian: $V(\mu) = \mu^3$
- negative binomial: $V(\mu) = \mu + k\mu^2$
- geometric: $V(\mu) = \mu + \mu^2$
- multinomial
- zero-inflated Poisson
- zero-inflated negative binomial

The negative binomial and zero-inflated negative binomial are distributions with an additional parameter k in the variance function. PROC GENMOD estimates k by maximum likelihood, or you can optionally set it to a constant value. For discussions of the negative binomial distribution, see McCullagh and Nelder (1989); Hilbe (1994, 2007); Long (1997); Cameron and Trivedi (1998); Lawless (1987).

The multinomial distribution is sometimes used to model a response that can take values from a number of categories. The binomial is a special case of the multinomial with two categories. See the section "Multinomial Models" on page 3674 and McCullagh and Nelder (1989, Chapter 5) for a description of the multinomial distribution.

The zero-inflated Poisson and zero-inflated negative binomial are included in PROC GENMOD even though they are not generalized linear models. They are useful extensions of generalized linear models. See the section "Zero-Inflated Models" on page 3675 for information about the zero-inflated distributions. Models for data with correlated responses fit by the GEE method are not available for zero-inflated distributions.

In addition, you can easily define your own link functions or distributions through DATA step programming statements used within the procedure.

An important aspect of generalized linear modeling is the selection of explanatory variables in the model. Changes in goodness-of-fit statistics are often used to evaluate the contribution of subsets of explanatory variables to a particular model. The deviance, defined to be twice the difference between the maximum attainable log likelihood and the log likelihood of the model under consideration, is often used as a measure of goodness of fit. The maximum attainable log likelihood is achieved with a model that has a parameter for every observation. See the section "Goodness of Fit" on page 3662 for formulas for the deviance.

One strategy for variable selection is to fit a sequence of models, beginning with a simple model with only an intercept term, and then to include one additional explanatory variable in each successive model. You can measure the importance of the additional explanatory variable by the difference in deviances or fitted log likelihoods between successive models. Asymptotic tests computed by the GENMOD procedure enable you to assess the statistical significance of the additional term.

The GENMOD procedure enables you to fit a sequence of models, up through a maximum number of terms specified in a MODEL statement. A table summarizes twice the difference in log likelihoods between each successive pair of models. This is called a *Type 1* analysis in the GENMOD procedure, because it is analogous to Type I (sequential) sums of squares in the GLM procedure. As with the PROC GLM Type I sums of squares, the results from this process depend on the order in which the model terms are fit.

The GENMOD procedure also generates a *Type 3* analysis analogous to Type III sums of squares in the GLM procedure. A Type 3 analysis does not depend on the order in which the terms for the model are specified. A

GENMOD procedure Type 3 analysis consists of specifying a model and computing likelihood ratio statistics for Type III contrasts for each term in the model. The contrasts are defined in the same way as they are in the GLM procedure. The GENMOD procedure optionally computes Wald statistics for Type III contrasts. This is computationally less expensive than likelihood ratio statistics, but it is thought to be less accurate because the specified significance level of hypothesis tests based on the Wald statistic might not be as close to the actual significance level as it is for likelihood ratio tests.

A Type 3 analysis generalizes the use of Type III estimable functions in linear models. Briefly, a Type III estimable function (contrast) for an effect is a linear function of the model parameters that involves the parameters of the effect and any interactions with that effect. A test of the hypothesis that the Type III contrast for a main effect is equal to 0 is intended to test the significance of the main effect in the presence of interactions. See Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions," for more information about Type III estimable functions. Also see Littell, Freund, and Spector (1991).

Additional features of the GENMOD procedure include the following:

- likelihood ratio statistics for user-defined contrasts—that is, linear functions of the parameters and *p*-values based on their asymptotic chi-square distributions
- estimated values, standard errors, and confidence limits for user-defined contrasts and least squares means
- ability to create a SAS data set corresponding to most tables displayed by the procedure (see Table 51.14 and Table 51.15)
- confidence intervals for model parameters based on either the profile likelihood function or asymptotic normality
- syntax similar to that of PROC GLM for the specification of the response and model effects, including interaction terms and automatic coding of classification variables
- ability to fit GEE models for clustered response data
- ability to perform Bayesian analysis by Gibbs sampling

Getting Started: GENMOD Procedure

Poisson Regression

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/genmgs1.sas.)

You can use the GENMOD procedure to fit a variety of statistical models. A typical use of PROC GENMOD is to perform Poisson regression.

You can use the Poisson distribution to model the distribution of cell counts in a multiway contingency table. Aitkin et al. (1989) have used this method to model insurance claims data. Suppose the following hypothetical insurance claims data are classified by two factors: age group (with two levels) and car type (with three levels).

```
data insure;
   input n c car$ age;
   ln = log(n);
   datalines;
500   42  small  1
1200  37  medium 1
100    1  large  1
400  101  small  2
500   73  medium 2
300   14  large  2
;
```

In the preceding data set, the variable n represents the number of insurance policyholders and the variable c represents the number of insurance claims. The variable car is the type of car involved (classified into three groups) and the variable age is the age group of a policyholder (classified into two groups).

You can use PROC GENMOD to perform a Poisson regression analysis of these data with a log link function. This type of model is sometimes called a *log-linear model*.

Assume that the number of claims c has a Poisson probability distribution and that its mean, μ_i, is related to the factors car and age for observation i by

$$\begin{aligned}
\log(\mu_i) &= \log(\mathsf{n}_i) + \mathbf{x}_i'\boldsymbol{\beta} \\
&= \log(\mathsf{n}_i) + \beta_0 + \\
&\quad \mathsf{car}_i(1)\beta_1 + \mathsf{car}_i(2)\beta_2 + \mathsf{car}_i(3)\beta_3 + \\
&\quad \mathsf{age}_i(1)\beta_4 + \mathsf{age}_i(2)\beta_5
\end{aligned}$$

The indicator variables $\mathsf{car}_i(j)$ and $\mathsf{age}_i(j)$ are associated with the jth level of the variables car and age for observation i

$$\mathsf{car}_i(j) = \begin{cases} 1 & \text{if } \mathsf{car} = j \\ 0 & \text{if } \mathsf{car} \neq j \end{cases}$$

The βs are unknown parameters to be estimated by the procedure. The logarithm of the variable n is used as an *offset*—that is, a regression variable with a constant coefficient of 1 for each observation. A log-linear relationship between the mean and the factors car and age is specified by the log link function. The log link function ensures that the mean number of insurance claims for each car and age group predicted from the fitted model is positive.

The following statements invoke the GENMOD procedure to perform this analysis:

```
proc genmod data=insure;
   class car age;
   model c = car age / dist   = poisson
                       link   = log
                       offset = ln;
run;
```

The variables car and age are specified as CLASS variables so that PROC GENMOD automatically generates the indicator variables associated with car and age.

The MODEL statement specifies c as the response variable and car and age as explanatory variables. An intercept term is included by default. Thus, the model matrix $\mathbf{X}$ (the matrix that has as its ith row the transpose of the covariate vector for the ith observation) consists of a column of 1s representing the intercept term and columns of 0s and 1s derived from indicator variables representing the levels of the car and age variables.

That is, the model matrix is

$$\mathbf{X} = \left[\begin{array}{c|ccc|cc} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array}\right]$$

where the first column corresponds to the intercept, the next three columns correspond to the variable car, and the last two columns correspond to the variable age.

The response distribution is specified as Poisson, and the link function is chosen to be log. That is, the Poisson mean parameter μ is related to the linear predictor by

$$\log(\mu) = \mathbf{x}_i'\boldsymbol{\beta}$$

The logarithm of n is specified as an offset variable, as is common in this type of analysis. In this case, the offset variable serves to normalize the fitted cell means to a per-policyholder basis, since the total number of claims, not individual policyholder claims, is observed. PROC GENMOD produces the following default output from the preceding statements.

Figure 51.1 Model Information

The GENMOD Procedure

Model Information	
Data Set	WORK.INSURE
Distribution	Poisson
Link Function	Log
Dependent Variable	c
Offset Variable	ln

The "Model Information" table displayed in Figure 51.1 provides information about the specified model and the input data set.

Figure 51.2 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
car	3	large medium small
age	2	1 2

Figure 51.2 displays the "Class Level Information" table, which identifies the levels of the classification variables that are used in the model. Note that car is a character variable, and the values are sorted in alphabetical order. This is the default sort order, but you can select different sort orders with the ORDER= option in the PROC GENMOD statement.

Figure 51.3 Goodness of Fit

Criteria For Assessing Goodness Of Fit			
Criterion	**DF**	**Value**	**Value/DF**
Deviance	2	2.8207	1.4103
Scaled Deviance	2	2.8207	1.4103
Pearson Chi-Square	2	2.8416	1.4208
Scaled Pearson X2	2	2.8416	1.4208
Log Likelihood		837.4533	
Full Log Likelihood		-16.4638	
AIC (smaller is better)		40.9276	
AICC (smaller is better)		80.9276	
BIC (smaller is better)		40.0946	

The "Criteria For Assessing Goodness Of Fit" table displayed in Figure 51.3 contains statistics that summarize the fit of the specified model. These statistics are helpful in judging the adequacy of a model and in comparing it with other models under consideration. If you compare the deviance of 2.8207 with its asymptotic chi-square with 2 degrees of freedom distribution, you find that the p-value is 0.24. This indicates that the specified model fits the data reasonably well.

Figure 51.4 Analysis of Parameter Estimates

Analysis Of Maximum Likelihood Parameter Estimates								
Parameter		DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		1	-1.3168	0.0903	-1.4937	-1.1398	212.73	<.0001
car	large	1	-1.7643	0.2724	-2.2981	-1.2304	41.96	<.0001
car	medium	1	-0.6928	0.1282	-0.9441	-0.4414	29.18	<.0001
car	small	0	0.0000	0.0000	0.0000	0.0000	.	.
age	1	1	-1.3199	0.1359	-1.5863	-1.0536	94.34	<.0001
age	2	0	0.0000	0.0000	0.0000	0.0000	.	.
Scale		0	1.0000	0.0000	1.0000	1.0000		

Note: The scale parameter was held fixed.

Figure 51.4 displays the "Analysis Of Parameter Estimates" table, which summarizes the results of the iterative parameter estimation process. For each parameter in the model, PROC GENMOD displays columns with the parameter name, the degrees of freedom associated with the parameter, the estimated parameter value, the standard error of the parameter estimate, the confidence intervals, and the Wald chi-square statistic and associated *p*-value for testing the significance of the parameter to the model. If a column of the model matrix corresponding to a parameter is found to be linearly dependent, or *aliased*, with columns corresponding to parameters preceding it in the model, PROC GENMOD assigns it zero degrees of freedom and displays a value of zero for both the parameter estimate and its standard error.

This table includes a row for a scale parameter, even though there is no free scale parameter in the Poisson distribution. See the section "Response Probability Distributions" on page 3655 for the form of the Poisson probability distribution. PROC GENMOD allows the specification of a scale parameter to fit overdispersed Poisson and binomial distributions. In such cases, the SCALE row indicates the value of the overdispersion scale parameter used in adjusting output statistics. See the section "Overdispersion" on page 3665 for more about overdispersion and the meaning of the SCALE parameter output by the GENMOD procedure. PROC GENMOD displays a note indicating that the scale parameter is fixed—that is, not estimated by the iterative fitting process.

It is usually of interest to assess the importance of the main effects in the model. Type 1 and Type 3 analyses generate statistical tests for the significance of these effects. You can request these analyses with the TYPE1 and TYPE3 options in the MODEL statement, as follows:

```
proc genmod data=insure;
   class car age;
   model c = car age / dist   = poisson
                       link   = log
                       offset = ln
                       type1
                       type3;
run;
```

The results of these analyses are summarized in the figures that follow.

Figure 51.5 Type 1 Analysis

The GENMOD Procedure

LR Statistics For Type 1 Analysis				
Source	**Deviance**	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	175.1536			
car	107.4620	2	67.69	<.0001
age	2.8207	1	104.64	<.0001

In the table for Type 1 analysis displayed in Figure 51.5, each entry in the deviance column represents the deviance for the model containing the effect for that row and all effects preceding it in the table. For example, the deviance corresponding to car in the table is the deviance of the model containing an intercept and car. As more terms are included in the model, the deviance decreases.

Entries in the chi-square column are likelihood ratio statistics for testing the significance of the effect added to the model containing all the preceding effects. The chi-square value of 67.69 for car represents twice the difference in log likelihoods between fitting a model with only an intercept term and a model with an intercept and car. Since the scale parameter is set to 1 in this analysis, this is equal to the difference in deviances. Since two additional parameters are involved, this statistic can be compared with a chi-square distribution with two degrees of freedom. The resulting p-value (labeled Pr>Chi) of less than 0.0001 indicates that this variable is highly significant. Similarly, the chi-square value of 104.64 for age represents the difference in log likelihoods between the model with the intercept and car and the model with the intercept, car, and age. This effect is also highly significant, as indicated by the small p-value.

Figure 51.6 Type 3 Analysis

LR Statistics For Type 3 Analysis			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
car	2	72.82	<.0001
age	1	104.64	<.0001

The Type 3 analysis results in the same conclusions as the Type 1 analysis. The Type 3 chi-square value for the car variable, for example, is twice the difference between the log likelihood for the model with the variables Intercept, car, and age included and the log likelihood for the model with the car variable excluded. The hypothesis tested in this case is the significance of the variable car given that the variable age is in the model. In other words, it tests the additional contribution of car in the model.

The values of the Type 3 likelihood ratio statistics for the car and age variables indicate that both of these factors are highly significant in determining the claims performance of the insurance policyholders.

Bayesian Analysis of a Linear Regression Model

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/genmgs2.sas.)

Neter et al. (1996) describe a study of 54 patients undergoing a certain kind of liver operation in a surgical unit. The data set Surg contains survival time and certain covariates for each patient. Observations for the first 20 patients in the data set Surg are shown in Figure 51.7.

Figure 51.7 Surgical Unit Data

Obs	x1	x2	x3	x4	y	logy	Logx1
1	6.7	62	81	2.59	200	2.3010	1.90211
2	5.1	59	66	1.70	101	2.0043	1.62924
3	7.4	57	83	2.16	204	2.3096	2.00148
4	6.5	73	41	2.01	101	2.0043	1.87180
5	7.8	65	115	4.30	509	2.7067	2.05412
6	5.8	38	72	1.42	80	1.9031	1.75786
7	5.7	46	63	1.91	80	1.9031	1.74047
8	3.7	68	81	2.57	127	2.1038	1.30833
9	6.0	67	93	2.50	202	2.3054	1.79176
10	3.7	76	94	2.40	203	2.3075	1.30833
11	6.3	84	83	4.13	329	2.5172	1.84055
12	6.7	51	43	1.86	65	1.8129	1.90211
13	5.8	96	114	3.95	830	2.9191	1.75786
14	5.8	83	88	3.95	330	2.5185	1.75786
15	7.7	62	67	3.40	168	2.2253	2.04122
16	7.4	74	68	2.40	217	2.3365	2.00148
17	6.0	85	28	2.98	87	1.9395	1.79176
18	3.7	51	41	1.55	34	1.5315	1.30833
19	7.3	68	74	3.56	215	2.3324	1.98787
20	5.6	57	87	3.02	172	2.2355	1.72277

Consider the model

$$\mathrm{Y} = \beta_0 + \beta_1 \mathrm{LogX1} + \beta_2 \mathrm{X2} + \beta_3 \mathrm{X3} + \beta_4 \mathrm{X4} + \epsilon$$

where Y is the survival time, LogX1 is log(blood-clotting score), X2 is a prognostic index, X3 is an enzyme function test score, X4 is a liver function test score, and ϵ is an $N(0, \sigma^2)$ error term.

A question of scientific interest is whether blood clotting score has a positive effect on survival time. Using PROC GENMOD, you can obtain a maximum likelihood estimate of the coefficient and construct a null point hypothesis to test whether β_1 is equal to 0. However, if you are interested in finding the probability that the coefficient is positive, Bayesian analysis offers a convenient alternative. You can use Bayesian analysis to directly estimate the conditional probability, $\Pr(\beta_1 > 0|\mathbf{Y})$, using the posterior distribution samples, which are produced as part of the output by PROC GENMOD.

The example that follows shows how to use PROC GENMOD to carry out a Bayesian analysis of the linear model with a normal error term. The SEED= option is specified to maintain reproducibility; no other options

are specified in the BAYES statement. By default, a uniform prior distribution is assumed on the regression coefficients. The uniform prior is a flat prior on the real line with a distribution that reflects ignorance of the location of the parameter, placing equal likelihood on all possible values the regression coefficient can take. Using the uniform prior in the following example, you would expect the Bayesian estimates to resemble the classical results of maximizing the likelihood. If you can elicit an informative prior distribution for the regression coefficients, you should use the COEFFPRIOR= option to specify it. A default noninformative gamma prior is used for the scale parameter σ.

You should make sure that the posterior distribution samples have achieved convergence before using them for Bayesian inference. PROC GENMOD produces three convergence diagnostics by default. If ODS Graphics is enabled as specified in the following SAS statements, diagnostic plots are also displayed. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more information about convergence diagnostics and their interpretation.

Summary statistics of the posterior distribution samples are produced by default. However, these statistics might not be sufficient for carrying out your Bayesian inference, and further processing of the posterior samples might be necessary. The following SAS statements request the Bayesian analysis, and the OUTPOST= option saves the samples in the SAS data set PostSurg for further processing:

```
proc genmod data=Surg;
   model y = Logx1 X2 X3 X4 / dist=normal;
   bayes seed=1 OutPost=PostSurg;
run;
```

The results of this analysis are shown in the following figures. The "Model Information" table in Figure 51.8 summarizes information about the model you fit and the size of the simulation.

Figure 51.8 Model Information

The GENMOD Procedure

Bayesian Analysis

Model Information		
Data Set	WORK.SURG	
Burn-In Size	2000	
MC Sample Size	10000	
Thinning	1	
Sampling Algorithm	Conjugate	
Distribution	Normal	
Link Function	Identity	
Dependent Variable	y	Survival Time

The "Analysis of Maximum Likelihood Parameter Estimates" table in Figure 51.9 summarizes maximum likelihood estimates of the model parameters.

Figure 51.9 Maximum Likelihood Parameter Estimates

Analysis Of Maximum Likelihood Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits	
Intercept	1	-730.559	85.4333	-898.005	-563.112
Logx1	1	171.8758	38.2250	96.9561	246.7954
x2	1	4.3019	0.5566	3.2109	5.3929
x3	1	4.0309	0.4996	3.0517	5.0100
x4	1	18.1377	12.0721	-5.5232	41.7986
Scale	1	59.8591	5.7599	49.5705	72.2832

Note: The scale parameter was estimated by maximum likelihood.

Since no prior distributions for the regression coefficients were specified, the default noninformative uniform distributions shown in the "Uniform Prior for Regression Coefficients" table in Figure 51.10 are used. Noninformative priors are appropriate if you have no prior knowledge of the likely range of values of the parameters, and if you want to make probability statements about the parameters or functions of the parameters. See, for example, Ibrahim, Chen, and Sinha (2001) for more information about choosing prior distributions.

Figure 51.10 Regression Coefficient Priors

The GENMOD Procedure

Bayesian Analysis

Uniform Prior for Regression Coefficients	
Parameter	Prior
Intercept	Constant
Logx1	Constant
x2	Constant
x3	Constant
x4	Constant

The default noninformative improper prior distribution for the normal dispersion parameter is shown in the "Independent Prior Distributions for Model Parameters" table in Figure 51.11.

Figure 51.11 Scale Parameter Prior

Independent Prior Distributions for Model Parameters	
Parameter	Prior Distribution
Dispersion	Improper

By default, the maximum likelihood estimates of the regression parameters are used as the starting values for the simulation when noninformative prior distributions are used. These are listed in the "Initial Values and Seeds" table in Figure 51.12.

Figure 51.12 MCMC Initial Values and Seeds

Initial Values of the Chain							
Chain	Seed	Intercept	Logx1	x2	x3	x4	Dispersion
1	1	-730.559	171.8758	4.301896	4.030878	18.1377	3449.176

Summary statistics for the posterior sample are displayed in the "Fit Statistics," "Descriptive Statistics for the Posterior Sample," "Interval Statistics for the Posterior Sample," and "Posterior Correlation Matrix" tables in Figure 51.13, Figure 51.14, Figure 51.15, and Figure 51.16, respectively.

Figure 51.13 Fit Statistics

Fit Statistics	
DIC (smaller is better)	607.796
pD (effective number of parameters)	6.062

Figure 51.14 Descriptive Statistics

The GENMOD Procedure

Bayesian Analysis

Posterior Summaries						
				Percentiles		
Parameter	N	Mean	Standard Deviation	25%	50%	75%
Intercept	10000	-730.0	91.2102	-789.6	-729.6	-670.5
Logx1	10000	171.7	40.6455	144.2	171.6	198.6
x2	10000	4.2988	0.5952	3.9029	4.2919	4.6903
x3	10000	4.0308	0.5359	3.6641	4.0267	4.3921
x4	10000	18.0858	12.9123	9.4471	18.1230	26.8141
Dispersion	10000	4113.1	867.7	3497.2	3995.9	4606.4

Figure 51.15 Interval Statistics

Posterior Intervals					
		Equal-Tail			
Parameter	Alpha	Interval		HPD Interval	
Intercept	0.050	-908.6	-549.8	-906.9	-549.1
Logx1	0.050	91.9723	252.5	94.1279	254.0
x2	0.050	3.1091	5.4778	3.1705	5.5167
x3	0.050	2.9803	5.1031	2.9227	5.0343
x4	0.050	-7.3043	43.6387	-8.8440	41.8229
Dispersion	0.050	2741.5	6096.6	2540.1	5810.0

Figure 51.16 Posterior Sample Correlation Matrix

Posterior Correlation Matrix						
Parameter	**Intercept**	**Logx1**	**x2**	**x3**	**x4**	**Dispersion**
Intercept	1.000	-0.857	-0.579	-0.712	0.582	0.000
Logx1	-0.857	1.000	0.286	0.491	-0.640	0.007
x2	-0.579	0.286	1.000	0.302	-0.489	-0.009
x3	-0.712	0.491	0.302	1.000	-0.618	-0.006
x4	0.582	-0.640	-0.489	-0.618	1.000	0.003
Dispersion	0.000	0.007	-0.009	-0.006	0.003	1.000

Since noninformative prior distributions were used, the posterior sample means, standard deviations, and interval statistics shown in Figure 51.13 and Figure 51.14 are consistent with the maximum likelihood estimates shown in Figure 51.9.

By default, PROC GENMOD computes three convergence diagnostics: the lag1, lag5, lag10, and lag50 autocorrelations (Figure 51.17); Geweke diagnostic statistics (Figure 51.18); and effective sample sizes (Figure 51.19). There is no indication that the Markov chain has not converged. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more information about convergence diagnostics and their interpretation.

Figure 51.17 Posterior Sample Autocorrelations

The GENMOD Procedure

Bayesian Analysis

Posterior Autocorrelations				
Parameter	**Lag 1**	**Lag 5**	**Lag 10**	**Lag 50**
Intercept	-0.0059	-0.0037	-0.0152	0.0010
Logx1	-0.0002	-0.0064	-0.0066	-0.0054
x2	-0.0120	-0.0026	-0.0267	-0.0168
x3	0.0036	0.0033	-0.0035	0.0004
x4	0.0034	-0.0064	0.0083	-0.0124
Dispersion	-0.0011	0.0091	-0.0279	0.0037

Figure 51.18 Geweke Diagnostic Statistics

Geweke Diagnostics		
Parameter	**z**	**Pr > |z|**
Intercept	-1.0815	0.2795
Logx1	1.6667	0.0956
x2	0.0977	0.9222
x3	0.2506	0.8021
x4	-1.1082	0.2678
Dispersion	0.2451	0.8064

Figure 51.19 Effective Sample Sizes

Effective Sample Sizes			
		Autocorrelation	
Parameter	**ESS**	**Time**	**Efficiency**
Intercept	10000.0	1.0000	1.0000
Logx1	10000.0	1.0000	1.0000
x2	10245.2	0.9761	1.0245
x3	10000.0	1.0000	1.0000
x4	10000.0	1.0000	1.0000
Dispersion	10000.0	1.0000	1.0000

Trace, autocorrelation, and density plots for the seven model parameters, shown in Figure 51.20 through Figure 51.25, are useful in diagnosing whether the Markov chain of posterior samples has converged. These plots show no evidence that the chain has not converged. See the section "Visual Analysis via Trace Plots" on page 163 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for help with interpreting these diagnostic plots.

Figure 51.20 Diagnostic Plots for Intercept

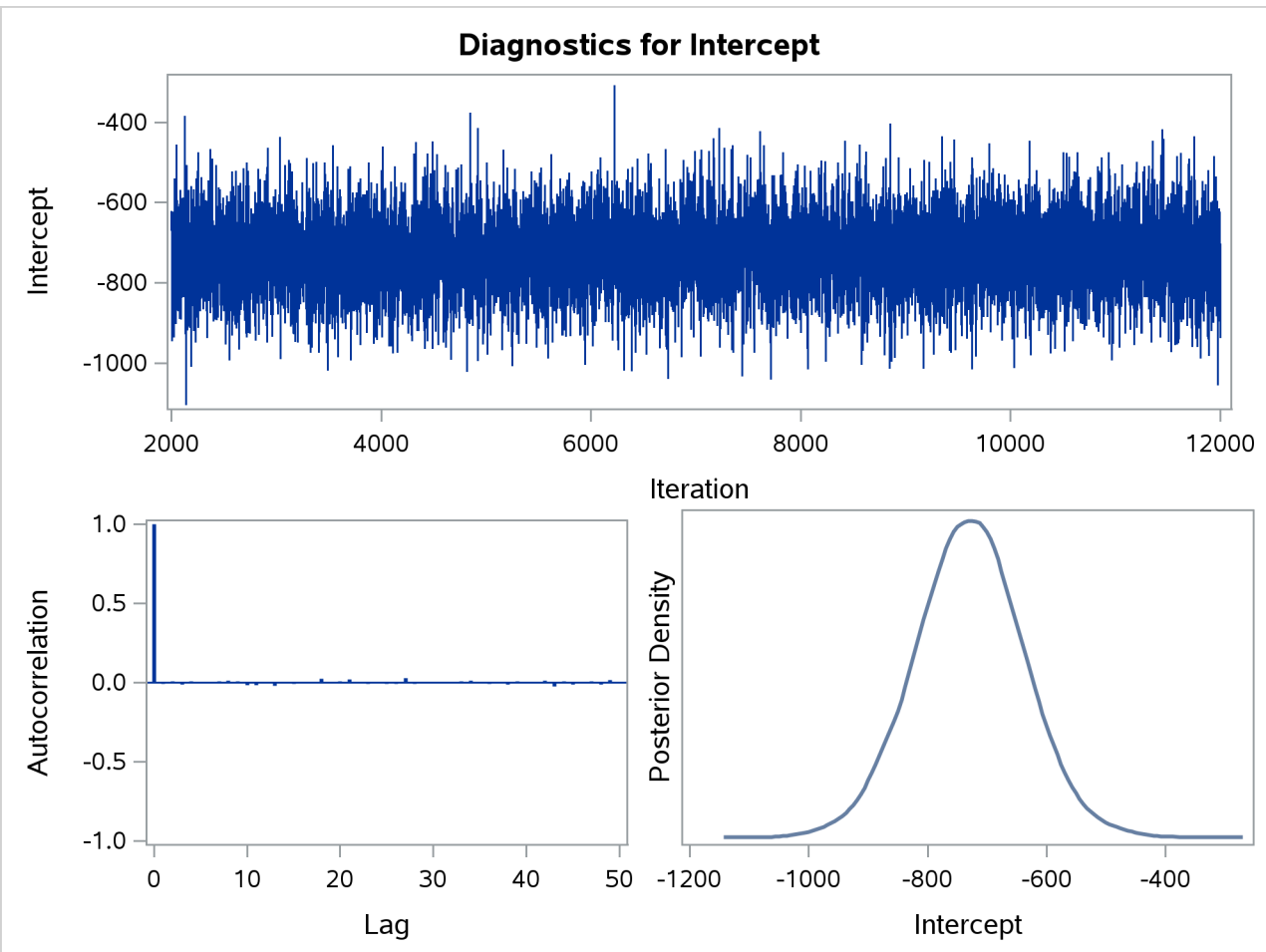

Figure 51.21 Diagnostic Plots for logX1

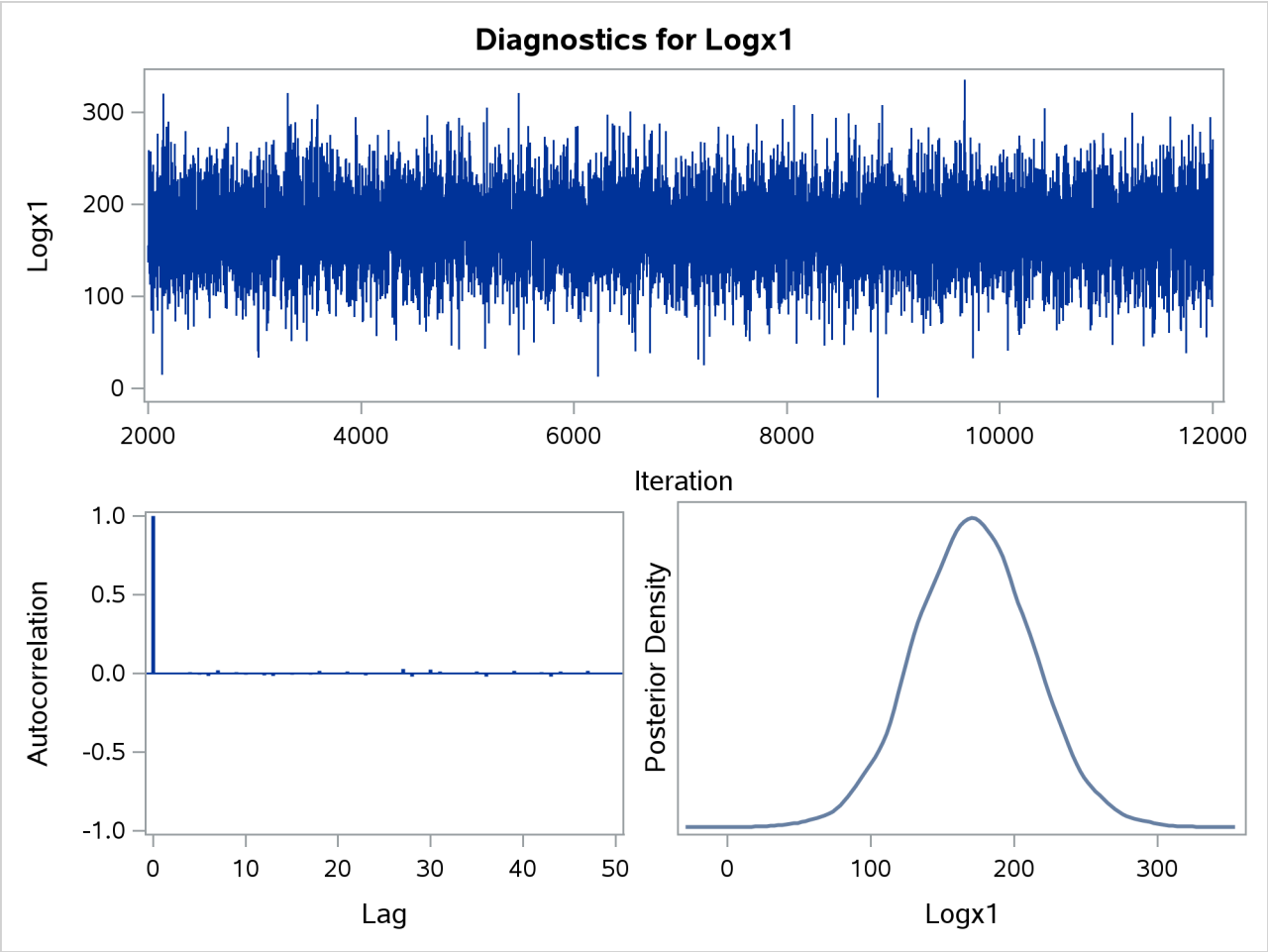

Figure 51.22 Diagnostic Plots for X2

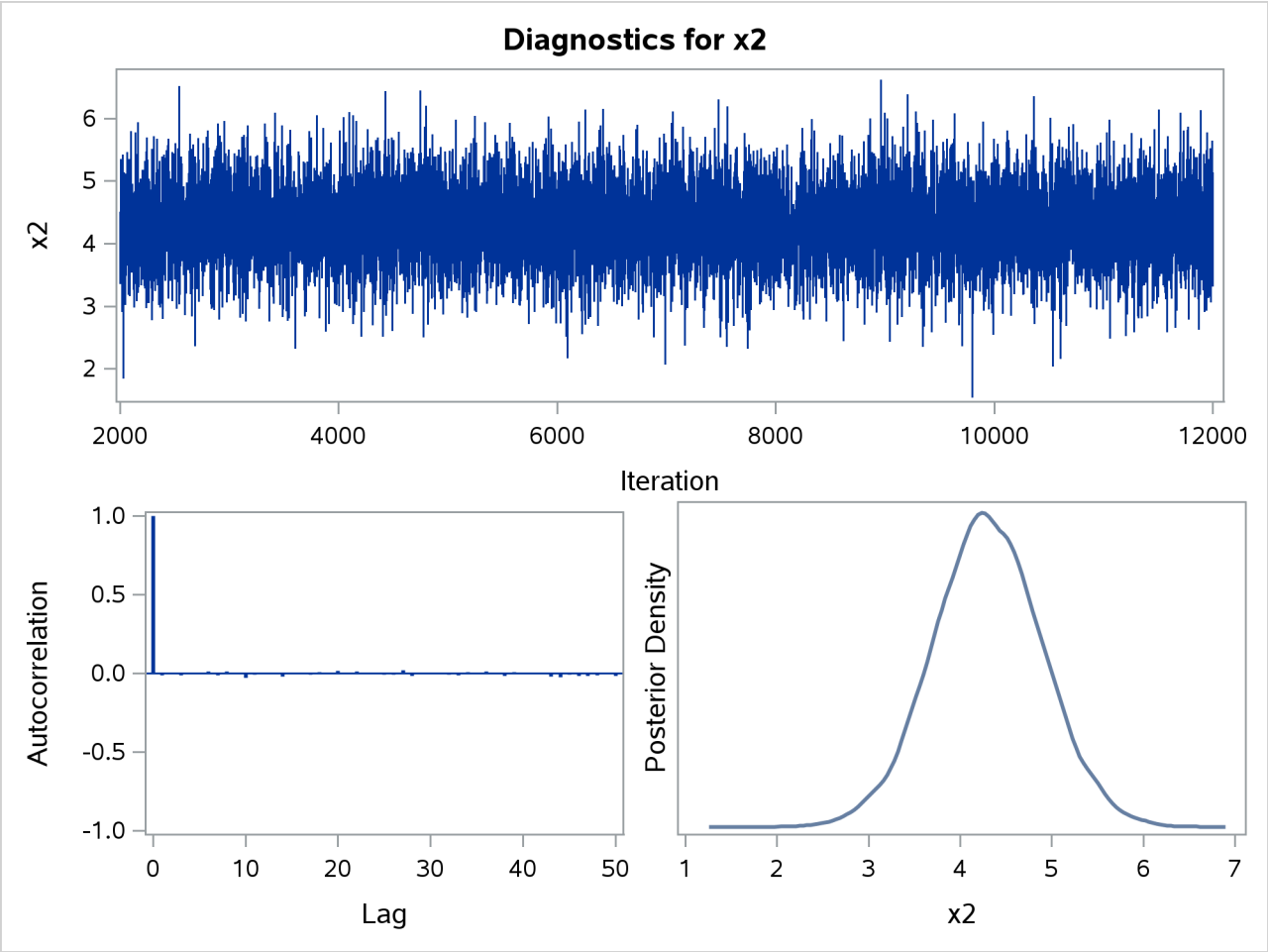

Figure 51.23 Diagnostic Plots for X3

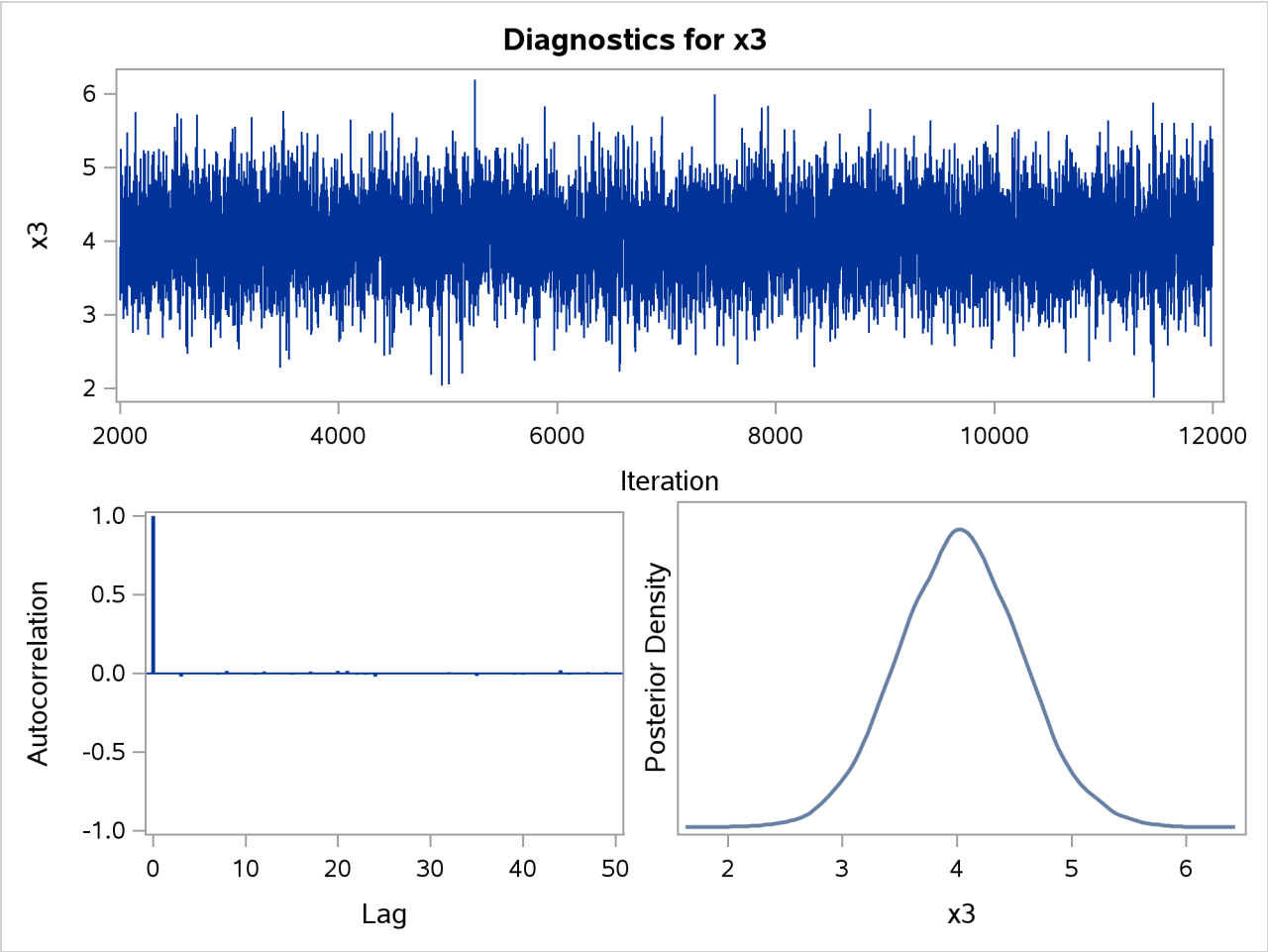

Figure 51.24 Diagnostic Plots for X4

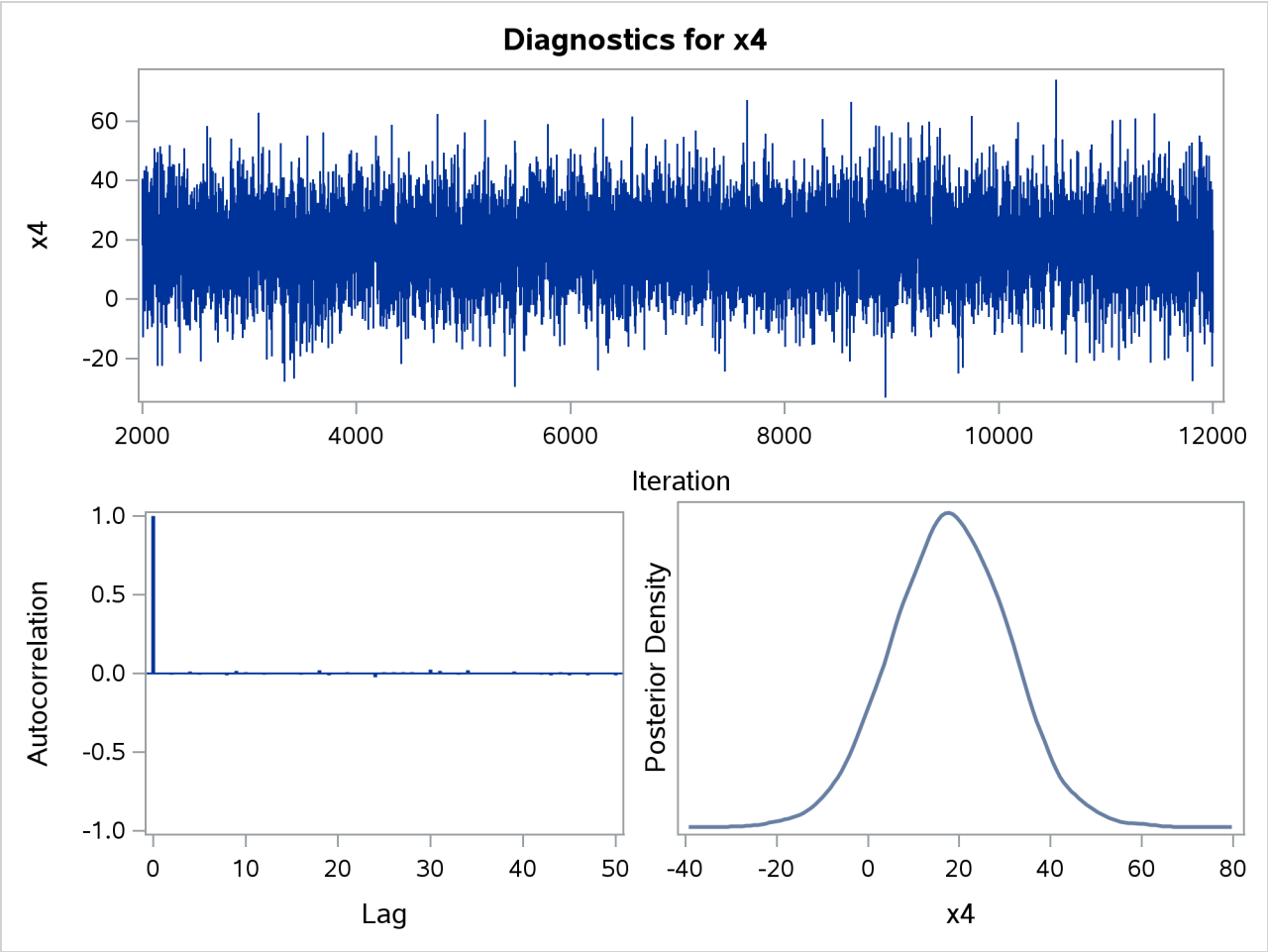

Figure 51.25 Diagnostic Plots for X5

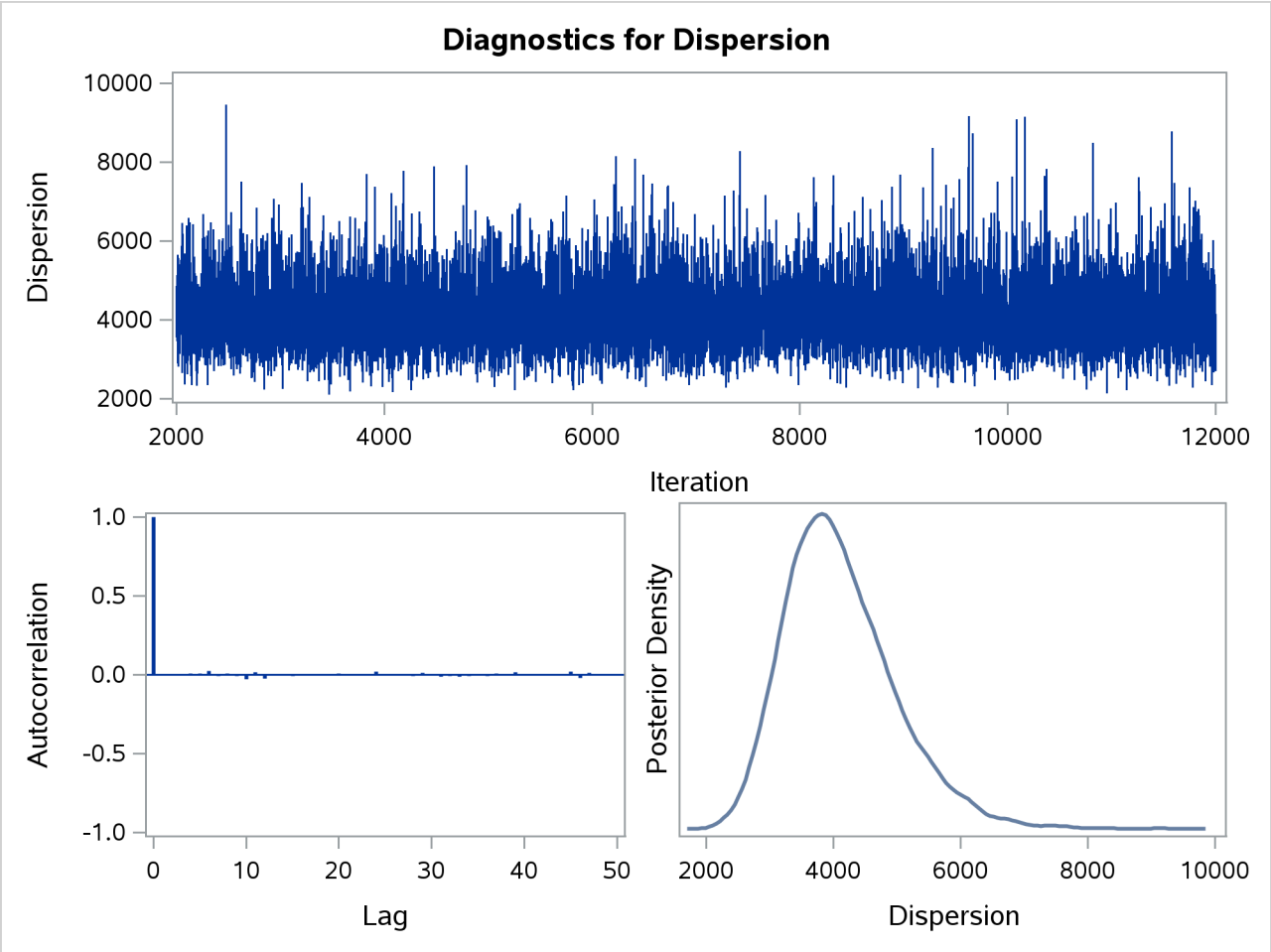

Suppose, for illustration, a question of scientific interest is whether blood clotting score has a positive effect on survival time. Since the model parameters are regarded as random quantities in a Bayesian analysis, you can answer this question by estimating the conditional probability of β_1 being positive, given the data, $\Pr(\beta_1 > 0|\mathbf{Y})$, from the posterior distribution samples. The following SAS statements compute the estimate of the probability of β_1 being positive:

```
data Prob;
   set PostSurg;
   Indicator = (logX1 > 0);
   label Indicator= 'log(Blood Clotting Score) > 0';
run;

proc Means data = Prob(keep=Indicator) n mean;
run;
```

As shown in Figure 51.26, there is a 1.00 probability of a positive relationship between the logarithm of a blood clotting score and survival time, adjusted for the other covariates.

Figure 51.26 Probability That $\beta_1 > 0$

The MEANS Procedure

Analysis Variable : Indicator log(Blood Clotting Score) > 0	
N	Mean
10000	0.9999000

Generalized Estimating Equations

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/genmgs3.sas.)

This section illustrates the use of the REPEATED statement to fit a GEE model, using repeated measures data from the "Six Cities" study of the health effects of air pollution (Ware et al. 1984). The data analyzed are the 16 selected cases in Lipsitz et al. (1994). The binary response is the wheezing status of 16 children at ages 9, 10, 11, and 12 years. A value of 1 of wheezing status indicates the occurrence of wheezing. The mean response is modeled as a logistic regression model by using the explanatory variables city of residence, age, and maternal smoking status at the particular age. The binary responses for individual children are assumed to be equally correlated, implying an exchangeable correlation structure.

The data set and SAS statements that fit the model by the GEE method are as follows:

```
data six;
   input case city$ @@;
   do i=1 to 4;
      input age smoke wheeze @@;
      output;
   end;
   datalines;
 1 portage   9 0 1  10 0 1  11 0 1  12 0 0
 2 kingston  9 1 1  10 2 1  11 2 0  12 2 0
 3 kingston  9 0 1  10 0 0  11 1 0  12 1 0
 4 portage   9 0 0  10 0 1  11 0 1  12 1 0
 5 kingston  9 0 0  10 1 0  11 1 0  12 1 0
 6 portage   9 0 0  10 1 0  11 1 0  12 1 0
 7 kingston  9 1 0  10 1 0  11 0 0  12 0 0
 8 portage   9 1 0  10 1 0  11 1 0  12 2 0
 9 portage   9 2 1  10 2 0  11 1 0  12 1 0
10 kingston  9 0 0  10 0 0  11 0 0  12 1 0
11 kingston  9 1 1  10 0 0  11 0 1  12 0 1
12 portage   9 1 0  10 0 0  11 0 0  12 0 0
13 kingston  9 1 0  10 0 1  11 1 1  12 1 1
14 portage   9 1 0  10 2 0  11 1 0  12 2 1
15 kingston  9 1 0  10 1 0  11 1 0  12 2 1
16 portage   9 1 1  10 1 1  11 2 0  12 1 0
;
```

```
proc genmod data=six;
   class case city;
   model  wheeze(event='1') = city age smoke  /  dist=bin;
   repeated  subject=case / type=exch covb corrw;
run;
```

The CLASS statement and the MODEL statement specify the model for the mean of the wheeze variable response as a logistic regression with city, age, and smoke as independent variables, just as for an ordinary logistic regression. The EVENT='1' option in the MODEL statement models the probability that wheeze = 1. If the EVENT='1' option had not been specified, the probability that wheeze = 0 would be modeled by default.

The REPEATED statement invokes the GEE method, specifies the correlation structure, and controls the displayed output from the GEE model. The option SUBJECT=CASE specifies that individual subjects be identified in the input data set by the variable case. The SUBJECT= variable case must be listed in the CLASS statement. Measurements on individual subjects at ages 9, 10, 11, and 12 are in the proper order in the data set, so the WITHINSUBJECT= option is not required. The TYPE=EXCH option specifies an exchangeable working correlation structure, the COVB option specifies that the parameter estimate covariance matrix be displayed, and the CORRW option specifies that the final working correlation be displayed.

Initial parameter estimates for iterative fitting of the GEE model are computed as in an ordinary generalized linear model, as described previously. Results of the initial model fit displayed as part of the generated output are not shown here. Statistics for the initial model fit such as parameter estimates, standard errors, deviances, and Pearson chi-squares do not apply to the GEE model and are valid only for the initial model fit. The following figures display information that applies to the GEE model fit.

Figure 51.27 displays general information about the GEE model fit.

Figure 51.27 GEE Model Information

The GENMOD Procedure

GEE Model Information	
Correlation Structure	Exchangeable
Subject Effect	case (16 levels)
Number of Clusters	16
Correlation Matrix Dimension	4
Maximum Cluster Size	4
Minimum Cluster Size	4

Figure 51.28 displays the parameter estimate covariance matrices specified by the COVB option. Both model-based and empirical covariances are produced.

Figure 51.28 GEE Parameter Estimate Covariance Matrices

Covariance Matrix (Model-Based)				
	Prm1	**Prm2**	**Prm4**	**Prm5**
Prm1	5.74947	-0.22257	-0.53472	0.01655
Prm2	-0.22257	0.45478	-0.002410	0.01876
Prm4	-0.53472	-0.002410	0.05300	-0.01658
Prm5	0.01655	0.01876	-0.01658	0.19104

Covariance Matrix (Empirical)				
	Prm1	**Prm2**	**Prm4**	**Prm5**
Prm1	9.33994	-0.85104	-0.83253	-0.16534
Prm2	-0.85104	0.47368	0.05736	0.04023
Prm4	-0.83253	0.05736	0.07778	-0.002364
Prm5	-0.16534	0.04023	-0.002364	0.13051

The exchangeable working correlation matrix specified by the CORRW option is displayed in Figure 51.29.

Figure 51.29 GEE Working Correlation Matrix

Working Correlation Matrix				
	Col1	**Col2**	**Col3**	**Col4**
Row1	1.0000	0.1648	0.1648	0.1648
Row2	0.1648	1.0000	0.1648	0.1648
Row3	0.1648	0.1648	1.0000	0.1648
Row4	0.1648	0.1648	0.1648	1.0000

The parameter estimates table, displayed in Figure 51.30, contains parameter estimates, standard errors, confidence intervals, Z scores, and p-values for the parameter estimates. Empirical standard error estimates are used in this table. A table that displays model-based standard errors can be created by using the REPEATED statement option MODELSE.

Figure 51.30 GEE Parameter Estimates Table

Analysis Of GEE Parameter Estimates Empirical Standard Error Estimates							
Parameter		**Estimate**	**Standard Error**	**95% Confidence Limits**		**Z**	**Pr > |Z|**
Intercept		1.2751	3.0561	-4.7148	7.2650	0.42	0.6765
city	kingston	0.1223	0.6882	-1.2266	1.4713	0.18	0.8589
city	portage	0.0000	0.0000	0.0000	0.0000	.	.
age		-0.2036	0.2789	-0.7502	0.3431	-0.73	0.4655
smoke		-0.0935	0.3613	-0.8016	0.6145	-0.26	0.7957

Syntax: GENMOD Procedure

The following statements are available in the GENMOD procedure. Items within the < > are optional.

```
PROC GENMOD <options> ;
    ASSESS | ASSESSMENT VAR=(effect)| LINK </ options> ;
    BAYES <options> ;
    BY variables ;
    CLASS variable <(options)> ... < variable <(options)>> </ options> ;
    CODE <options> ;
    CONTRAST 'label' contrast-specification </ options> ;
    DEVIANCE variable = expression ;
    EFFECTPLOT <plot-type <(plot-definition-options)>> </ options> ;
    ESTIMATE 'label' effect values <, ... effect values> </ options> ;
    EXACT <'label'> <INTERCEPT> <effects> </ options> ;
    EXACTOPTIONS options ;
    FREQ | FREQUENCY variable ;
    FWDLINK variable = expression ;
    INVLINK variable = expression ;
    LSMEANS <model-effects> </ options> ;
    LSMESTIMATE model-effect <'label'> values <divisor=n> <, ... <'label'> values <divisor=n>>
                    </ options> ;
    MARGINS model-effects </ options> ;
    MODEL response = < effects > </ options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=name ... keyword=name> ;
    Programming statements ;
    REPEATED SUBJECT=subject-effect </ options> ;
    SLICE model-effect </ options> ;
    STORE <OUT= >item-store-name </ LABEL='label'> ;
    STRATA variable <(option)> ... < variable <(option)>> </ options> ;
    WEIGHT | SCWGT variable ;
    VARIANCE variable = expression ;
    ZEROMODEL < effects > </ options> ;
```

The ASSESS, BAYES, BY, CLASS, CODE, CONTRAST, DEVIANCE, ESTIMATE, FREQUENCY, FWDLINK, INVLINK, MODEL, OUTPUT, REPEATED, VARIANCE, WEIGHT, and ZEROMODEL statements, as well as the programming statements, are described in full after the PROC GENMOD statement in alphabetical order. The EFFECTPLOT, LSMEANS, LSMESTIMATE, MARGINS, SLICE, and STORE statements are common to many procedures. Summary descriptions of functionality and syntax for these statements are also given after the PROC GENMOD statement in alphabetical order, and full documentation about them is available in Chapter 20, "Shared Concepts and Topics."

The PROC GENMOD statement invokes the GENMOD procedure. All statements other than the MODEL statement are optional. The CLASS statement, if present, must precede the MODEL statement, and the CONTRAST and EXACT statements must come after the MODEL statement.

PROC GENMOD Statement

PROC GENMOD < *options* > **;**

The PROC GENMOD statement invokes the GENMOD procedure. Table 51.1 summarizes the *options* available in the PROC GENMOD statement.

Table 51.1 PROC GENMOD Statement Options

Option	Description
DATA=	Specifies the input data set
DESCENDING	Sorts response variable in the reverse of the default order
EXACTONLY	Requests only the exact analyses
NAMELEN=	Specifies the length of effect names
ORDER=	Specifies the sort order of CLASS variable
PLOTS	Controls the plots produced through ODS Graphics
RORDER=	Specifies the sort order for the levels of the response variable

You can specify the following *options*.

DATA=*SAS-data-set*

specifies the SAS data set containing the data to be analyzed. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

DESCENDING

DESCEND

DESC

specifies that the levels of the response variable for the ordinal multinomial model and the binomial model with single variable response syntax be sorted in the reverse of the default order. For example, if RORDER=FORMATTED (the default), the DESCENDING option causes the levels to be sorted from highest to lowest instead of from lowest to highest. If RORDER=FREQ, the DESCENDING option causes the levels to be sorted from lowest frequency count to highest instead of from highest to lowest.

EXACTONLY

requests only the exact analyses. The asymptotic analysis that PROC GENMOD usually performs is suppressed.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters long, where *n* is a value between 20 and 200 characters. The default length is 20 characters.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

The ORDER= option can be useful when you use the CONTRAST or ESTIMATE statement because it determines which parameters in the model correspond to each level in the data.

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

specifies plots to be created using ODS Graphics. Many of the observational statistics in the output data set can be plotted using this option. You are not required to create an output data set in order to produce a plot. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=all
plots=predicted
plots=(predicted reschi)
plots(unpack)=dfbeta
```

ODS Graphics must be enabled before plots can be requested. For example:

```
proc genmod plots=all;
   model y = x;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

Any specified *global-plot-options* apply to all plots that are specified with *plot-requests*. The following *global-plot-options* are available.

CLUSTERLABEL
displays formatted levels of the SUBJECT= effect instead of plot symbols. This option applies only to diagnostic statistics for models fit by GEEs that are plotted against cluster number, and provides a way to identify cluster level names with corresponding ordered cluster numbers.

UNPACK
displays multiple plots individually. The default is to display related multiple plots in a panel.

See the section "OUTPUT Statement" on page 3642 for definitions of the statistics specified with the *plot-requests*. The *plot-requests* include the following:

ALL
produces all available plots.

COOKSD
DOBS
plots the Cook's distance statistic as a function of observation number.

DFBETA
plots the $\boldsymbol{\beta}$ deletion statistic as a function of observation number for each regression parameter in the model.

DFBETAS
plots the standardized $\boldsymbol{\beta}$ deletion statistic as a function of observation number for each regression parameter in the model.

LEVERAGE
plots the leverage as a function of observation number.

OVERDISP
plots the predicted variance as a function of the predicted mean for a zero-inflated response.

PREDICTED<(*option*)>
plots predicted values with confidence limits as a function of observation number. The PREDICTED plot request has the following *option*:

CLM
includes confidence limits in the predicted value plot.

PZERO
plots the zero inflation probability for zero-inflated Poisson and negative binomial models as a function of observation number.

RESCHI<(*options*)>
The RESCHI plot request has the following *options*:

INDEX
plots as a function of observation number.

XBETA
plots as a function of linear predictor.

If you do not specify an *option*, Pearson residuals are plotted as a function of observation number.

RESDEV< (*options*) >
plots deviance residuals. The RESDEV plot request has the following *options*:

INDEX
plots as a function of observation number.

XBETA
plots as a function of linear predictor.

If you do not specify an *option*, deviance residuals are plotted as a function of observation number.

RESLIK< (*options*) >
plots likelihood residuals. The RESLIK plot request has the following *options*:

INDEX
plots as a function of observation number.

XBETA
plots as a function of linear predictor.

If you do not specify an *option*, likelihood residuals are plotted as a function of observation number.

RESRAW< (*options*) >
plots raw residuals. The RESRAW plot request has the following *options*:

INDEX
plots as a function of observation number.

XBETA
plots as a function of linear predictor.

If you do not specify an *option*, raw residuals are plotted as a function of observation number.

STDRESCHI< (*options*) >
plots standardized Pearson residuals. The STDRESCHI plot request has the following *options*:

INDEX
plots as a function of observation number.

XBETA
plots as a function of linear predictor.

If you do not specify an *option*, standardized Pearson residuals are plotted as a function of observation number.

STDRESDEV< (*options*) >
plots standardized deviance residuals. The STDRESDEV plot request has the following *options*:

INDEX
plots as a function of observation number.

XBETA
plots as a function of linear predictor.

If you do not specify an *option*, standardized deviance residuals are plotted as a function of observation number.

If you fit a model by using generalized estimating equations (GEEs), the following additional *plot-requests* are available:

CLEVERAGE
plots the cluster leverage as a function of ordered cluster.

CLUSTERCOOKSD
DCLS
plots the cluster Cook's distance statistic as a function of ordered cluster.

CLUSTERDFIT
MCLS
plots the studentized cluster Cook's distance statistic as a function of ordered cluster.

DFBETAC
plots the cluster deletion statistic as a function of ordered cluster for each regression parameter in the model.

DFBETACS
plots the standardized cluster deletion statistic as a function of ordered cluster for each regression parameter in the model.

RORDER=*keyword*
specifies the sort order for the levels of the response variable. This order determines which intercept parameter in the model corresponds to each level in the data. If RORDER=FORMATTED for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. The following table displays the valid *keywords* and describes how PROC GENMOD interprets them.

RORDER=*keyword*	**Levels Sorted by**
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, RORDER=FORMATTED. For RORDER=FORMATTED and RORDER=INTERNAL, the sort order is machine dependent. The DESCENDING option in the PROC GENMOD statement

causes the response variable to be sorted in the reverse of the order displayed in the previous table. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide*.

The NOPRINT option, which suppresses displayed output in other SAS procedures, is not available in the PROC GENMOD statement. However, you can use the Output Delivery System (ODS) to suppress all displayed output, store all output on disk for further analysis, or create SAS data sets from selected output. You can suppress all displayed output with the statement **ODS SELECT NONE;** and turn displayed output back on with the statement **ODS SELECT ALL;**. See Table 51.14 and Table 51.15 for the names of output tables available from PROC GENMOD. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

ASSESS Statement

ASSESS VAR=(*effect***)| LINK** < / *options* > **;**

ASSESSMENT VAR=(*effect***)| LINK** < / *options* > **;**

The ASSESS statement computes and plots, using ODS Graphics, model-checking statistics based on aggregates of residuals. See the section "Assessment of Models Based on Aggregates of Residuals" on page 3687 for details about the model assessment methods available in GENMOD.

The types of aggregates available are cumulative residuals, moving sums of residuals, and loess smoothed residuals. If you do not specify which aggregate to use, the assessments are based on cumulative sums. PROC GENMOD uses ODS Graphics for graphical displays. For specific information about the graphics available in PROC GENMOD, see the section "ODS Graphics" on page 3718.

You must specify either LINK or VAR= in order to create an analysis.

LINK requests the assessment of the link function by performing the analysis with respect to the linear predictor.

VAR=(*effect***)** specifies that the functional form of a covariate be checked by performing the analysis with respect to the variable identified by the effect. The effect must be specified in the MODEL statement and must contain only continuous variables (variables not listed in a CLASS statement).

You can specify the following *options* after the slash (/).

CRPANEL

requests that a plot with four panels showing just a few of the paths from the default aggregate plot to make it easier to compare simulated and observed paths. The plot in each panel contains aggregates of the observed residuals and two simulated curves (fewer if NPATHS= is less than 8).

LOESS< (*number*) >

LOWESS< (*number*) >

requests model assessment based on loess smoothed residuals with optional *number* the fraction of data used; *number* must be between zero and one. If *number* is not specified, the default value one-third is used.

NPATHS=*number*

NPATH=*number*

PATHS=*number*

PATH=*number*

specifies the number of simulated paths to plot in the default aggregate residuals plot. The default value of *number* is twenty.

RESAMPLE< =*number* **>**

RESAMPLES< =*number* **>**

specifies that a p-value be computed based on 1,000 simulated paths, or *number* paths, if *number* is specified.

SEED=*number*

specifies a seed for the normal random number generator used in creating simulated realizations of aggregates of residuals for plots and estimating p-values. Specifying a seed enables you to produce identical graphs and p-values from one run of the procedure to the next run. If a seed is not specified, or if *number* is negative or zero, a random number seed is derived from the time of day.

WINDOW< (*number***) >**

requests assessment based on a moving sum window of width *number*. If *number* is not specified, a value of one-half of the range of the x-coordinate is used.

BAYES Statement

BAYES < *options* > **;**

The BAYES statement requests a Bayesian analysis of the regression model by using Gibbs sampling. The Bayesian posterior samples (also known as the chain) for the regression parameters are not tabulated. The Bayesian posterior samples (also known as the chain) for the regression parameters can be output to a SAS data set. Table 51.2 summarizes the *options* available in the BAYES statement.

Table 51.2 BAYES Statement Options

Option	Description
Monte Carlo Options	
INITIAL=	Specifies the initial values of the chain
INITIALMLE	Specifies that maximum likelihood estimates be used as initial values of the chain
METROPOLIS=	Specifies the use of a Metropolis step in the ARMS algorithm
NBI=	Specifies the number of burn-in iterations
NMC=	Specifies the number of iterations after burn-in
SAMPLING=	Specifies the algorithm used to sample the posterior distribution
SEED=	Specifies the random number generator seed
THINNING=	Controls the thinning of the Markov chain

Table 51.2 *continued*

Option	Description
Model and Prior Options	
COEFFPRIOR=	Specifies the prior of the regression coefficients
DISPERSIONPRIOR=	Specifies the prior of the dispersion parameter
PRECISIONPRIOR=	Specifies the prior of the precision parameter
SCALEPRIOR=	Specifies the prior of the scale parameter
Summary Statistics and Convergence Diagnostics	
DIAGNOSTICS=	Displays convergence diagnostics
PLOTS=	Displays diagnostic plots
STATISTICS=	Displays summary statistics of the posterior samples
Posterior Samples	
OUTPOST=	Names a SAS data set for the posterior samples

The following list describes these *options* and their *suboptions*.

COEFFPRIOR=JEFFREYS< (*option*) > | NORMAL< (*options*) > | UNIFORM

COEFF=JEFFREYS< (*options*) > | NORMAL< (*options*) > | UNIFORM

CPRIOR=JEFFREYS< (*options*) > | NORMAL< (*options*) > | UNIFORM

specifies the prior distribution for the regression coefficients. The default is COEFFPRIOR=UNIFORM, which specifies the noninformative and improper prior of a constant.

Jeffreys' prior is specified by COEFFPRIOR=JEFFREYS, which can be followed by the following option in parentheses. Jeffreys' prior is proportional to $|I(\boldsymbol{\beta})|^{\frac{1}{2}}$, where $I(\boldsymbol{\beta})$ is the Fisher information matrix. See the section "Jeffreys' Prior" on page 3696 and Ibrahim and Laud (1991) for more details.

CONDITIONAL

specifies that the Jeffreys' prior, conditional on the current Markov chain value of the generalized linear model precision parameter τ, is proportional to $|\tau\mathbf{I}(\boldsymbol{\beta})|^{\frac{1}{2}}$.

The normal prior is specified by COEFFPRIOR=NORMAL, which can be followed by one of the following *options* enclosed in parentheses. However, if you do not specify an *option*, the normal prior $N(\mathbf{0}, 10^6\mathbf{I})$, where $\mathbf{I}$ is the identity matrix, is used. See the section "Normal Prior" on page 3697 for more details.

CONDITIONAL

specifies that the normal prior, conditional on the current Markov chain value of the generalized linear model precision parameter τ, is $N(\boldsymbol{\mu}, \tau^{-1}\boldsymbol{\Sigma})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of the normal prior specified by other normal options.

INPUT=*SAS-data-set*

specifies a SAS data set containing the mean and covariance information of the normal prior. The data set must have a _TYPE_ variable to represent the type of each observation and a variable for each regression coefficient. If the data set also contains a _NAME_ variable, the values of this variable are used to identify the covariances for the _TYPE_='COV' observations; otherwise, the _TYPE_='COV' observations are assumed to be in the same order as the explanatory variables

in the MODEL statement. PROC GENMOD reads the mean vector from the observation with _TYPE_='MEAN' and reads the covariance matrix from observations with _TYPE_='COV'.

For an independent normal prior, specify either variances or a covariance matrix in addition to the means. You can specify the variances in the _TYPE_='VAR' option, or you can equivalently specify the precisions (inverse of the variances) in the _TYPE_='PRECISION' option. Each variable in the INPUT= data set corresponds to a parameter in the model, and the variable name must match the parameter name. Parameter names can be found in any of the tables (such as the "Initial Values of the Chain" table) in the Bayesian Analysis section of the results. See Example 51.10 for an example.

RELVAR< =*c* >
specifies the normal prior $N(\mathbf{0}, c\mathbf{J})$, where $\mathbf{J}$ is a diagonal matrix with diagonal elements equal to the variances of the corresponding ML estimator. By default, $c = 10^6$.

VAR< =*c* >
specifies the normal prior $N(\mathbf{0}, c\mathbf{I})$, where $\mathbf{I}$ is the identity matrix.

DIAGNOSTICS=ALL | NONE | (*keyword-list*)

DIAG=ALL | NONE | (*keyword-list*)
controls the number of diagnostics produced. You can request all the following diagnostics by specifying DIAGNOSTICS=ALL. If you do not want any of these diagnostics, specify DIAGNOSTICS=NONE. If you want some but not all of the diagnostics, or if you want to change certain settings of these diagnostics, specify a subset of the following *keywords*. The default is DIAGNOSTICS=(AUTOCORR ESS GEWEKE).

AUTOCORR < (LAGS= *numeric-list*) >
computes the autocorrelations of lags given by LAGS= list for each parameter. Elements in the list are truncated to integers and repeated values are removed. If the LAGS= option is not specified, autocorrelations of lags 1, 5, 10, and 50 are computed for each variable. See the section "Autocorrelations" on page 174 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

ESS
computes Carlin's estimate of the effective sample size, the correlation time, and the efficiency of the chain for each parameter. See the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

GELMAN < (*gelman-options*) >
computes the Gelman and Rubin convergence diagnostics. You can specify one or more of the following *gelman-options*:

NCHAIN | N=*number*
specifies the number of parallel chains used to compute the diagnostic, and must be 2 or larger. The default is NCHAIN=3. If an INITIAL= data set is used, NCHAIN defaults to the number of rows in the INITIAL= data set. If any number other than this is specified with the NCHAIN= option, the NCHAIN= value is ignored.

ALPHA=*value*
specifies the significance level for the upper bound. The default is ALPHA=0.05, resulting in a 97.5% bound.

See the section "Gelman and Rubin Diagnostics" on page 167 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

GEWEKE < (*geweke-options***) >**
computes the Geweke spectral density diagnostics, which are essentially a two-sample t test between the first f_1 portion and the last f_2 portion of the chain. The default is $f_1 = 0.1$ and $f_2 = 0.5$, but you can choose other fractions by using the following *geweke-options*:

FRAC1=*value*
specifies the fraction f_1 for the first window.

FRAC2=*value*
specifies the fraction f_2 for the second window.

See the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

HEIDELBERGER < (*heidel-options***) >**
computes the Heidelberger and Welch diagnostic for each variable, which consists of a stationarity test of the null hypothesis that the sample values form a stationary process. If the stationarity test is not rejected, a halfwidth test is then carried out. Optionally, you can specify one or more of the following *heidel-options*:

SALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the stationarity test.

HALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the halfwidth test.

EPS=*value*
specifies a positive number ϵ such that if the halfwidth is less than ϵ times the sample mean of the retained iterates, the halfwidth test is passed.

See the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

MCSE

MCERROR
computes the Monte Carlo standard error for each parameter. The Monte Caro standard error, which measures the simulation accuracy, is the standard error of the posterior mean estimate and is calculated as the posterior standard deviation divided by the square root of the effective sample size. See the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

RAFTERY< (*raftery-options*) >

computes the Raftery and Lewis diagnostics that evaluate the accuracy of the estimated quantile ($\hat{\theta}_Q$ for a given $Q \in (0,1)$) of a chain. $\hat{\theta}_Q$ can achieve any degree of accuracy when the chain is allowed to run for a long time. A stopping criterion is when the estimated probability $\hat{P}_Q = \Pr(\theta \leq \hat{\theta}_Q)$ reaches within $\pm R$ of the value Q with probability S; that is, $\Pr(Q - R \leq \hat{P}_Q \leq Q + R) = S$. The following *raftery-options* enable you to specify Q, R, S, and a precision level ϵ for the test:

QUANTILE | Q=*value*

specifies the order (a value between 0 and 1) of the quantile of interest. The default is 0.025.

ACCURACY | R=*value*

specifies a small positive number as the margin of error for measuring the accuracy of estimation of the quantile. The default is 0.005.

PROBABILITY | S=*value*

specifies the probability of attaining the accuracy of the estimation of the quantile. The default is 0.95.

EPSILON | EPS=*value*

specifies the tolerance level (a small positive number) for the stationary test. The default is 0.001.

See the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

DISPERSIONPRIOR=GAMMA< (*options*) > | IGAMMA< (*options*) > | IMPROPER

DPRIOR=GAMMA< (*options*) > | IGAMMA< (*options*) > | IMPROPER

specifies that Gibbs sampling be performed on the generalized linear model dispersion parameter and the prior distribution for the dispersion parameter, if there is a dispersion parameter in the model. For models that do not have a dispersion parameter (the Poisson and binomial), this option is ignored. Note that you can specify Gibbs sampling on either the dispersion parameter ϕ, the scale parameter $\sigma = \phi^{\frac{1}{2}}$, or the precision parameter $\tau = \phi^{-1}$, with the DPRIOR=, SPRIOR=, and PPRIOR= options, respectively. These three parameters are transformations of one another, and you should specify Gibbs sampling for only one of them.

A gamma prior $G(a,b)$ with density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$ is specified by DISPERSIONPRIOR=GAMMA, which can be followed by one of the following *gamma-options* enclosed in parentheses. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. See the section "Gamma Prior" on page 3696 for details. The default is $G(10^{-4}, 10^{-4})$.

RELSHAPE< =*c* >

specifies independent $G(c\hat{\phi}, c)$ distribution, where $\hat{\phi}$ is the MLE of the dispersion parameter. With this choice of hyperparameters, the mean of the prior distribution is $\hat{\phi}$ and the variance is $\frac{\hat{\phi}}{c}$. By default, c=10^{-4}.

SHAPE=*a*

ISCALE=*b*

when both specified, results in a $G(a, b)$ prior.

SHAPE=*c*

when specified alone, results in a $G(c, c)$ prior.

ISCALE=*c*

when specified alone, results in a $G(c, c)$ prior.

An inverse gamma prior $\mathrm{IG}(a, b)$ with density $f(t) = \frac{b^a}{\Gamma(a)} t^{-(a+1)} \mathrm{e}^{-b/t}$ is specified by DISPERSIONPRIOR=IGAMMA, which can be followed by one of the following inverse gamma *options* enclosed in parentheses. The hyperparameters a and b are the shape and scale parameters of the inverse gamma distribution, respectively. See the section "Inverse Gamma Prior" on page 3696 for details. The default is $\mathrm{IG}(2.001, 0.001)$.

RELSHAPE< =*c* **>**

specifies independent $\mathrm{IG}(\frac{c+\hat{\phi}}{\hat{\phi}}, c)$ distribution, where $\hat{\phi}$ is the MLE of the dispersion parameter. With this choice of hyperparameters, the mean of the prior distribution is $\hat{\phi}$. By default, c=10^{-4}.

SHAPE=*a*

SCALE=*b*

when both specified, results in a $\mathrm{IG}(a, b)$ prior.

SHAPE=*c*

when specified alone, results in an $\mathrm{IG}(c, c)$ prior.

SCALE=*c*

when specified alone, results in an $\mathrm{IG}(c, c)$ prior.

An improper prior with density $f(t)$ proportional to t^{-1} is specified with DISPERSIONPRIOR=IMPROPER.

INITIAL=*SAS-data-set*

specifies the SAS data set that contains the initial values of the Markov chains. The INITIAL= data set must contain all the variables of the model. You can specify multiple rows as the initial values of the parallel chains for the Gelman-Rubin statistics, but posterior summaries, diagnostics, and plots are computed only for the first chain. If the data set also contains the variable _SEED_, the value of the _SEED_ variable is used as the seed of the random number generator for the corresponding chain.

INITIALMLE

specifies that maximum likelihood estimates of the model parameters be used as initial values of the Markov chain. If this option is not specified, estimates of the mode of the posterior distribution obtained by optimization are used as initial values.

METROPOLIS=YES | NO

specifies the use of a Metropolis step to generate Gibbs samples for posterior distributions that are not log concave. The default value is METROPOLIS=YES.

NBI=*number*
specifies the number of burn-in iterations before the chains are saved. The default is 2000.

NMC=*number*
specifies the number of iterations after the burn-in. The default is 10000.

OUTPOST=*SAS-data-set*

OUT=*SAS-data-set*
names the SAS data set that contains the posterior samples. See the sections "OUTPOST= Output Data Set" on page 3698 and "Posterior Samples Output Data Set" on page 3695 for more information. Alternatively, you can create the output data set by specifying an ODS OUTPUT statement as follows:

ODS OUTPUT POSTERIORSAMPLE=*SAS-data-set*

PRECISIONPRIOR=GAMMA< (*options***) > | IMPROPER**

PPRIOR=GAMMA< (*options***) > | IMPROPER**
specifies that Gibbs sampling be performed on the generalized linear model precision parameter and the prior distribution for the precision parameter, if there is a precision parameter in the model. For models that do not have a precision parameter (the Poisson and binomial), this option is ignored. Note that you can specify Gibbs sampling on either the dispersion parameter ϕ, the scale parameter $\sigma = \phi^{\frac{1}{2}}$, or the precision parameter $\tau = \phi^{-1}$, with the DPRIOR=, SPRIOR=, and PPRIOR= options, respectively. These three parameters are transformations of one another, and you should specify Gibbs sampling for only one of them.

A gamma prior $G(a, b)$ with density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$ is specified by PRECISIONPRIOR=GAMMA, which can be followed by one of the following *gamma-options* enclosed in parentheses. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. See the section "Gamma Prior" on page 3696 for details. The default is $G(10^{-4}, 10^{-4})$.

RELSHAPE< =*c* **>**
specifies independent $G(c\hat{\tau}, c)$ distribution, where $\hat{\tau}$ is the MLE of the dispersion parameter. With this choice of hyperparameters, the mean of the prior distribution is $\hat{\tau}$ and the variance is $\frac{\hat{\tau}}{c}$. By default, $c = 10^{-4}$.

SHAPE=*a*

ISCALE=*b*
when both specified, results in a $G(a, b)$ prior.

SHAPE=*c*
when specified alone, results in an $G(c, c)$ prior.

ISCALE=*c*
when specified alone, results in an $G(c, c)$ prior.

An improper prior with density $f(t)$ proportional to t^{-1} is specified with PRECISIONPRIOR=IMPROPER.

PLOTS< (*global-plot-options*) >=*plot-request*

PLOTS< (*global-plot-options*) >=(*plot-request* < ... *plot-request* >)

controls the display of diagnostic plots. Three types of plots can be requested: trace plots, autocorrelation function plots, and kernel density plots. By default, the plots are displayed in panels unless the *global-plot-option* UNPACK is specified. Also, when you are specifying more than one type of plots, the plots are displayed by parameters unless the *global-plot-option* GROUPBY is specified. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. For example:

```
plots=none
plots(unpack)=trace
plots=(trace autocorr)
```

ODS Graphics must be enabled before requesting plots. For example, the following SAS statements enable ODS Graphics:

```
ods graphics on;
proc genmod;
   model y=x;
   bayes plots=trace;
run;
ods graphics off;
```

The *global-plot-options* are as follows:

FRINGE

creates a fringe plot on the X axis of the density plot.

GROUPBY=PARAMETER

GROUPBY=TYPE

specifies how the plots are grouped when there is more than one type of plot.

GROUPBY=TYPE

specifies that the plots be grouped by type.

GROUPBY=PARAMETER

specifies that the plots be grouped by parameter.

GROUPBY=PARAMETER is the default.

LAGS=*n*

specifies that autocorrelations be plotted up to lag *n*. If this option is not specified, autocorrelations are plotted up to lag 50.

SMOOTH

displays a fitted penalized B-spline curve for each trace plot.

UNPACKPANEL
UNPACK
specifies that all paneled plots be unpacked, meaning that each plot in a panel is displayed separately.

The *plot-requests* include the following:

ALL
specifies all types of plots. PLOTS=ALL is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

AUTOCORR
displays the autocorrelation function plots for the parameters.

DENSITY
displays the kernel density plots for the parameters.

NONE
suppresses all diagnostic plots.

TRACE
displays the trace plots for the parameters. See the section "Visual Analysis via Trace Plots" on page 163 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

SAMPLING=*option*
specifies an algorithm used to sample the posterior distribution. The following *options* are available:

ARMS
GIBBS
use the ARMS algorithm.

GAMERMAN
GAM
use the Gamerman algorithm. This is the default method.

IM
use the independent Metropolis algorithm.

SCALEPRIOR=GAMMA< (*options***) > | IMPROPER**
SPRIOR=GAMMA< (*options***) > | IMPROPER**
specifies that Gibbs sampling be performed on the generalized linear model scale parameter and the prior distribution for the scale parameter, if there is a scale parameter in the model. For models that do not have a scale parameter (the Poisson and binomial), this option is ignored. Note that you can specify Gibbs sampling on either the dispersion parameter ϕ, the scale parameter $\sigma = \phi^{\frac{1}{2}}$, or the precision parameter $\tau = \phi^{-1}$, with the DPRIOR=, SPRIOR=, and PPRIOR= options, respectively. These three parameters are transformations of one another, and you should specify Gibbs sampling for only one of them.

A gamma prior $G(a, b)$ with density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$ is specified by SCALEPRIOR=GAMMA, which can be followed by one of the following *gamma-options* enclosed in parentheses. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. See the section "Gamma Prior" on page 3696 for details. The default is $G(10^{-4}, 10^{-4})$.

RELSHAPE< =*c*>
specifies independent $G(c\hat{\sigma}, c)$ distribution, where $\hat{\sigma}$ is the MLE of the dispersion parameter. With this choice of hyperparameters, the mean of the prior distribution is $\hat{\sigma}$ and the variance is $\frac{\hat{\sigma}}{c}$. By default, $c = 10^{-4}$.

SHAPE=*a*
ISCALE=*b*
when both specified, results in a $G(a, b)$ prior.

SHAPE=*c*
when specified alone, results in an $G(c, c)$ prior.

ISCALE=*c*
when specified alone, results in an $G(c, c)$ prior.

An improper prior with density $f(t)$ proportional to t^{-1} is specified with SCALEPRIOR=IMPROPER.

SEED=*number*
specifies an integer seed in the range 1 to $2^{31} - 1$ for the random number generator in the simulation. Specifying a seed enables you to reproduce identical Markov chains for the same specification. If the SEED= option is not specified, or if you specify a nonpositive seed, a random seed is derived from the time of day.

STATISTICS < (*global-options*) > = ALL | NONE | *keyword* | (*keyword-list*)
STATS < (*global-options*) > = ALL | NONE | *keyword* | (*keyword-list*)
controls the number of posterior statistics produced. Specifying STATISTICS=ALL is equivalent to specifying STATISTICS= (SUMMARY INTERVAL COV CORR). If you do not want any posterior statistics, you specify STATISTICS=NONE. The default is STATISTICS=(SUMMARY INTERVAL). See the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details. The *global-options* include the following:

ALPHA=*numeric-list*
controls the probabilities of the credible intervals. The ALPHA= values must be between 0 and 1. Each ALPHA= value produces a pair of 100(1–ALPHA)% equal-tail and HPD intervals for each parameters. The default is the value of the ALPHA= option in the MODEL statement, or 0.05 if that option is not specified (yielding the 95% credible intervals for each parameter).

PERCENT=*numeric-list*
requests the percentile points of the posterior samples. The PERCENT= values must be between 0 and 100. The default is PERCENT=25, 50, 75, which yield the 25th, 50th, and 75th percentile points, respectively, for each parameter.

The list of *keywords* includes the following:

CORR
produces the posterior correlation matrix.

COV
produces the posterior covariance matrix.

SUMMARY
produces the means, standard deviations, and percentile points for the posterior samples. The default is to produce the 25th, 50th, and 75th percentile points, but you can use the global PERCENT= option to request specific percentile points.

INTERVAL
produces equal-tail credible intervals and HPD intervals. The default is to produce the 95% equal-tail credible intervals and 95% HPD intervals, but you can use the global ALPHA= option to request intervals of any probabilities.

THINNING=*number*
THIN=*number*
controls the thinning of the Markov chain. Only one in every k samples is used when THINNING=k, and if NBI=n_0 and NMC=n, the number of samples kept is

$$\left[\frac{n_0 + n}{k}\right] - \left[\frac{n_0}{k}\right]$$

where $[a]$ represents the integer part of the number a. The default is THINNING=1.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC GENMOD to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the GENMOD procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *global-options* > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. Response variables do not need to be specified in the CLASS statement.

The CLASS statement must precede the MODEL statement. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*. You can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING

DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC GENMOD orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (., ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values of the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so this option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC GENMOD interprets values of the ORDER= option:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. You can specify any of the *keywords* shown in the following table. The default is PARAM=GLM. Design matrix columns are created from CLASS variables according to the corresponding coding schemes.

Value of PARAM=	Coding
EFFECT	Effect coding
GLM	Less-than-full-rank reference cell coding (this *keyword* can be used only in a global option)
ORDINAL **THERMOMETER**	Cumulative parameterization for an ordinal CLASS variable
POLYNOMIAL **POLY**	Polynomial coding
REFERENCE **REF**	Reference cell coding
ORTHEFFECT	Orthogonalizes PARAM=EFFECT coding
ORTHORDINAL **ORTHOTHERM**	Orthogonalizes PARAM=ORDINAL coding
ORTHPOLY	Orthogonalizes PARAM=POLYNOMIAL coding
ORTHREF	Orthogonalizes PARAM=REFERENCE coding

All parameterizations are full rank, except for the GLM parameterization. The REF= option in the CLASS statement determines the reference level for EFFECT and REFERENCE coding and for their orthogonal parameterizations. It also indirectly determines the reference level for a singular GLM parameterization through the order of levels.

If a PARAM= option is specified as a variable option for some variables, then any variables for which PARAM= is not specified use either the EFFECT parameterization if the global PARAM= option is not specified, or the full-rank parameterization indicated in the global PARAM= option if specified. If the global PARAM=GLM option is specified and PARAM= is also specified for some variables, GLM parameterization is used for all variables.

If PARAM=ORTHPOLY or PARAM=POLY and the classification variable is numeric, then the ORDER= option in the CLASS statement is ignored, and the internal unformatted values are used. For more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

REF=*'level' | keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*:

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< *=n* >

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

Class Variable Default Parameterization

If you do not specify the PARAM= option, the default PARAM=GLM parameterization is used.

Class Variable Naming Convention

Parameter names for a CLASS predictor variable are constructed by concatenating the CLASS variable name with the CLASS levels. However, for the POLYNOMIAL and orthogonal parameterizations, parameter names are formed by concatenating the CLASS variable name and keywords that reflect the parameterization. For examples and more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

Class Variable Parameterization with Unbalanced Designs

PROC GENMOD initially parameterizes the CLASS variables by looking at the levels of the variables across the complete data set. If you have an *unbalanced* replication of levels across variables or BY groups, then the design matrix and the parameter interpretation might be different from what you expect. For example, suppose you have a model that has one CLASS variable A with three levels (1, 2, and 3) and another CLASS variable B with two levels (1 and 2). If the third level of A occurs only with the first level of B, if you use the EFFECT parameterization, and if your model contains the effect A(B) and an intercept, then the design for A within the second level of B is not a differential effect. In particular, the design looks like the following:

		Design Matrix			
		A(B=1)		A(B=2)	
B	A	A1	A2	A1	A2
1	1	1	0	0	0
1	2	0	1	0	0
1	3	–1	–1	0	0
2	1	0	0	1	0
2	2	0	0	0	1

PROC GENMOD detects linear dependency among the last two design variables and sets the parameter for A2(B=2) to zero, resulting in an interpretation of these parameters as if they were reference- or dummy-coded. The REFERENCE or GLM parameterization might be more appropriate for such problems.

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 51.3 summarizes the *options* available in the CODE statement.

Table 51.3 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

CONTRAST Statement

CONTRAST '*label*' *contrast-specification* < / *options* > **;**

The CONTRAST statement provides a means of obtaining a test of a specified hypothesis concerning the model parameters. This is accomplished by specifying a matrix $\mathbf{L}$ for testing the hypothesis $\mathbf{L}'\boldsymbol{\beta} = 0$. You must be familiar with the details of the model parameterization that PROC GENMOD uses. For more information, see the section "Parameterization Used in PROC GENMOD" on page 3667 and the section "CLASS Statement" on page 3612. Computed statistics are based on the asymptotic chi-square distribution of the likelihood ratio statistic, or the generalized score statistic for GEE models, with degrees of freedom determined by the number of linearly independent rows in the $\mathbf{L}'$ matrix. You can request Wald chi-square statistics with the Wald option in the CONTRAST statement.

There is no limit to the number of CONTRAST statements that you can specify, but they must appear after the MODEL statement and after the ZEROMODEL statement for zero-inflated models. Statistics for multiple CONTRAST statements are displayed in a single table.

The elements of the CONTRAST statement are as follows:

label identifies the contrast on the output. A label is required for every contrast specified. Labels can be up to 20 characters and must be enclosed in single quotes.

contrast-specification identifies the effects and their coefficients from which the $\mathbf{L}$ matrix is formed. The *contrast-specification* can be specified in two different ways. The first method applies to all models except the zero-inflated (ZI) distributions (zero-inflated Poisson and zero-inflated negative binomial), and the syntax is:

effect values < ,. . . *effect values* >

The second method of specifying a contrast applies only to ZI models, and the syntax is:

effect values < ,. . . *effect values* > **@ZERO** *effect values* < ,. . . *effect values* >

where

effect identifies an effect that appears in the MODEL statement. The value INTERCEPT or intercept can be used as an effect when an intercept is included in the model. You do not need to include all effects that are included in the MODEL statement.

values are constants that are elements of the $\mathbf{L}$ vector associated with the effect.

options specifies CONTRAST statement options.

Specification of sets of *effect values* before the @ZERO separator results in a row of the $\mathbf{L}'$ matrix with coefficients for *effects* in the regression part of the model set to *values* and with the coefficients for the zero-inflation part of the model set to zero. Specification of sets of *effect values* after the @ZERO separator results in a row of the $\mathbf{L}$ matrix with the coefficients for the regression part of the model set to zero and with the coefficients of *effects* in the zero-inflation part of the model set to *values*. For example, the statements

```
class a;
model y=a;
contrast 'Label1' A 1 -1;
```

specify an $\mathbf{L}'$ matrix with one row with coefficients 1 for the first level of A and –1 for the second level of A.

The statements

```
class a b;
model y=a / dist=zip;
zeromodel b;
contrast 'Label2' A 1 -1 @zero B 1 -1;
```

specify an $\mathbf{L}'$ matrix with two rows: the first row has coefficients 1 for the first level of A, –1 for the second level of A, and zeros for all levels of B; the second row has coefficients 0 for all levels of A, 1 for the first level of B, and –1 for the second level of B.

The rows of $\mathbf{L}'$ are specified in order and are separated by commas.

If you use the default less-than-full-rank PROC GLM CLASS variable parameterization, each row of the $\mathbf{L}'$ matrix is checked for estimability. If PROC GENMOD finds a contrast to be nonestimable, it displays missing values in corresponding rows in the results. See Searle (1971) for a discussion of estimable functions. If the elements of $\mathbf{L}'$ are not specified for an effect that contains a specified effect, then the elements of the specified effect are distributed over the levels of the higher-order effect just as the GLM procedure does for its CONTRAST and ESTIMATE statements. For example, suppose that the model contains effects A and B and their interaction A*B. If you specify a CONTRAST statement involving A alone, the $\mathbf{L}'$ matrix contains nonzero terms for both A and A*B, since A*B contains A.

When you use any of the full-rank PARAM= CLASS variable options, all parameters are directly estimable, and rows of $\mathbf{L}'$ are not checked for estimability.

If an effect is not specified in the CONTRAST statement, all of its coefficients in the $\mathbf{L}'$ matrix are set to 0. If too many values are specified for an effect, the extra ones are ignored. If too few values are specified, the remaining ones are set to 0.

PROC GENMOD handles missing level combinations of classification variables in the same manner as the GLM and MIXED procedures. Parameters that correspond to missing level combinations are not included in the model. This convention can affect the way in which you specify the $\mathbf{L}$ matrix in your CONTRAST statement.

If you specify the WALD option, the test of hypothesis is based on a Wald chi-square statistic. If you omit the WALD option, the test statistic computed depends on whether an ordinary generalized linear model or a GEE-type model is specified.

For an ordinary generalized linear model, the CONTRAST statement computes the likelihood ratio statistic. This is defined to be twice the difference between the log likelihood of the model unconstrained by the contrast and the log likelihood with the model fitted under the constraint that the linear function of the parameters defined by the contrast is equal to 0. A p-value is computed based on the asymptotic chi-square distribution of the chi-square statistic.

If you specify a GEE model with the REPEATED statement, the test is based on a score statistic. The GEE model is fit under the constraint that the linear function of the parameters defined by the contrast is equal to 0. The score chi-square statistic is computed based on the generalized score function. See the section "Generalized Score Statistics" on page 3686 for more information.

The degrees of freedom is the number of linearly independent constraints implied by the CONTRAST statement—that is, the rank of $\mathbf{L}$.

You can specify the following *options* after a slash (/).

E

requests that the $\mathbf{L}$ matrix be displayed.

SINGULAR=*number*

EPSILON=*number*

tunes the estimability checking. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the absolute value of the element of $\mathbf{v}$ with the largest absolute value. Let $\mathbf{K}'$ be any row in the contrast matrix $\mathbf{L}$. Define C to be equal to ABS($\mathbf{K}'$) if ABS($\mathbf{K}'$) is greater than 0; otherwise, C equals 1. If ABS($\mathbf{K}' - \mathbf{K}'\mathbf{T}$) is greater than C**number*, then $\mathbf{K}$ is declared nonestimable. $\mathbf{T}$ is the Hermite form matrix $(\mathbf{X}'\mathbf{X})^{-}(\mathbf{X}'\mathbf{X})$, and $(\mathbf{X}'\mathbf{X})^{-}$ represents a generalized inverse of the matrix $\mathbf{X}'\mathbf{X}$. The value for *number* must be between 0 and 1; the default value is 1E–4. The SINGULAR= option in the MODEL statement affects the computation of the generalized inverse of the matrix $\mathbf{X}'\mathbf{X}$. It might also be necessary to adjust this value for some data.

WALD

requests that a Wald chi-square statistic be computed for the contrast rather than the default likelihood ratio or score statistic. The Wald statistic for testing $\mathbf{L}'\boldsymbol{\beta} = \mathbf{0}$ is defined by

$$S = (\mathbf{L}'\hat{\boldsymbol{\beta}})'(\mathbf{L}'\boldsymbol{\Sigma}\mathbf{L})^{-}(\mathbf{L}'\hat{\boldsymbol{\beta}})$$

where $\hat{\boldsymbol{\beta}}$ is the maximum likelihood estimate and $\boldsymbol{\Sigma}$ is its estimated covariance matrix. The asymptotic distribution of S is χ^2_r, where r is the rank of $\mathbf{L}$. Computed p-values are based on this distribution.

If you specify a GEE model with the REPEATED statement, $\boldsymbol{\Sigma}$ is the empirical covariance matrix estimate.

DEVIANCE Statement

DEVIANCE *variable*=*expression* **;**

You can specify a probability distribution other than those available in PROC GENMOD by using the DEVIANCE and VARIANCE statements. You do not need to specify the DEVIANCE or VARIANCE statement if you use the DIST= MODEL statement option to specify a probability distribution. The *variable* identifies the deviance contribution from a single observation to the procedure, and it must be a valid SAS variable name that does not appear in the input data set. The *expression* can be any arithmetic expression supported by the DATA step language, and it is used to define the functional dependence of the deviance on the mean and the response. You use the automatic variables _MEAN_ and _RESP_ to represent the mean and response in the *expression*.

Alternatively, the deviance function can be defined using programming statements (see the section "Programming Statements" on page 3645) and assigned to a variable, which is then listed as the *expression*. This form is convenient for using complex statements such as IF-THEN/ELSE clauses.

The DEVIANCE statement is ignored unless the VARIANCE statement is also specified.

EFFECTPLOT Statement

EFFECTPLOT < *plot-type* < *(plot-definition-options)* > > < / *options* > **;**

The EFFECTPLOT statement produces a display of the fitted model and provides options for changing and enhancing the displays. Table 51.4 describes the available *plot-types* and their *plot-definition-options*.

Table 51.4 *Plot-Types* and *Plot-Definition-Options*

Plot-Type and Description	Plot-Definition-Options
BOX Displays a box plot of continuous response data at each level of a CLASS effect, with predicted values superimposed and connected by a line. This is an alternative to the INTERACTION *plot-type*.	PLOTBY= variable or CLASS effect X= CLASS variable or effect
CONTOUR Displays a contour plot of predicted values against two continuous covariates.	PLOTBY= variable or CLASS effect X= continuous variable Y= continuous variable
FIT Displays a curve of predicted values versus a continuous variable.	PLOTBY= variable or CLASS effect X= continuous variable
INTERACTION Displays a plot of predicted values (possibly with error bars) versus the levels of a CLASS effect. The predicted values are connected with lines and can be grouped by the levels of another CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= CLASS variable or effect
MOSAIC Displays a mosaic plot of predicted values using up to three CLASS effects.	PLOTBY= variable or CLASS effect X= CLASS effects
SLICEFIT Displays a curve of predicted values versus a continuous variable grouped by the levels of a CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= continuous variable

For full details about the syntax and options of the EFFECTPLOT statement, see the section "EFFECTPLOT Statement" on page 447 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE '*label*' *contrast-specification* < / *options* > **;**

The ESTIMATE statement is similar to a CONTRAST statement, except only one-row $\mathbf{L}'$ matrices are permitted.

The elements of the ESTIMATE statement are as follows:

label identifies the contrast on the output. A label is required for every contrast specified. Labels can be up to 20 characters and must be enclosed in single quotes.

contrast-specification identifies the effects and their coefficients from which the $\mathbf{L}$ matrix is formed. The *contrast-specification* can be specified in two different ways. The first method applies to all models except the zero-inflated (ZI) distributions (zero-inflated Poisson and zero-inflated negative binomial), and the syntax is:

effect values < . . . *effect values* >

The second method of specifying a contrast applies only to ZI models, and the syntax is:

effect values < . . . *effect values* > **@ZERO** *effect values* < . . . *effect values* >

where

effect identifies an effect that appears in the MODEL statement. The value INTERCEPT or intercept can be used as an effect when an intercept is included in the model. You do not need to include all effects that are included in the MODEL statement.

values are constants that are elements of the $\mathbf{L}$ vector associated with the effect.

options specifies options for the ESTIMATE statement.

For ZI models, sets of *effects values* before the @ZERO separator correspond to the regression part of the model with regression parameters $\boldsymbol{\beta}$, and *effects values* after the @ZERO separator correspond to the zero-inflation part of the model with regression parameters $\boldsymbol{\gamma}$. In the case of ZI models, a one-row $\mathbf{L}'$ matrix is created for the regression part of the model, another one-row $\mathbf{L}'$ matrix is created for the zero-inflation part of the model, and separate estimates for the two $\mathbf{L}$ matrices are computed and displayed.

If you use the default less-than-full-rank GLM CLASS variable parameterization, each row is checked for estimability. If PROC GENMOD finds a contrast to be nonestimable, it displays missing values in corresponding rows in the results. See Searle (1971) for a discussion of estimable functions.

The actual estimate, $\mathbf{L}'\boldsymbol{\beta}$ (and $\mathbf{L}'\boldsymbol{\gamma}$ for ZI models), its approximate standard error, and confidence limits are displayed. Additionally, the corresponding estimate on the mean scale (defined as the inverse link function applied to $\mathbf{L}'\boldsymbol{\beta}$), and confidence limits are displayed. Wald chi-square tests that $\mathbf{L}'\boldsymbol{\beta} = 0$ and $\mathbf{L}'\boldsymbol{\gamma} = 0$ are also displayed.

The approximate standard error of the estimate is computed as the square root of $\mathbf{L}'\hat{\boldsymbol{\Sigma}}\mathbf{L}$, where $\hat{\boldsymbol{\Sigma}}$ is the estimated covariance matrix of the parameter estimates. If you specify a GEE model in the REPEATED statement, $\hat{\boldsymbol{\Sigma}}$ is the empirical covariance matrix estimate.

If you specify the EXP option, then $\exp(\mathbf{L}'\boldsymbol{\beta})$, its standard error, and its confidence limits are also displayed.

The construction of the $\mathbf{L}$ vector and the checking for estimability for an ESTIMATE statement follow the same rules as listed under the CONTRAST statement.

You can specify the following *options* in the ESTIMATE statement after a slash (/).

ALPHA=*number*
requests that a confidence interval be constructed with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default value is 0.05.

DIVISOR=*number*
specifies a value by which to divide all coefficients so that fractional coefficients can be entered as integer numerators. For example, you can use

```
estimate '1/3(A1+A2) - 2/3A3' a 1 1 -2 / divisor=3;
```

instead of

```
estimate '1/3(A1+A2) - 2/3A3' a 0.33333 0.33333 -0.66667;
```

E
requests that the $\mathbf{L}$ matrix coefficients be displayed.

EXP
requests that $\exp(\mathbf{L}'\boldsymbol{\beta})$, its standard error, and its confidence limits be computed. If you specify the EXP option, standard errors are computed using the delta method. Confidence limits are computed by exponentiating the confidence limits for $\mathbf{L}'\boldsymbol{\beta}$.

SINGULAR=*number*
EPSILON=*number*
tunes the estimability checking as described for the CONTRAST statement.

EXACT Statement

EXACT <'*label*'> <**INTERCEPT**> < *effects*> < / *options*> **;**

The EXACT statement performs exact tests of the parameters for the specified *effects* and optionally estimates the parameters and outputs the exact conditional distributions. You can specify the keyword INTERCEPT and any *effects* in the MODEL statement. Inference on the parameters of the specified effects is performed by conditioning on the sufficient statistics of all the other model parameters (possibly including the intercept).

You can specify several EXACT statements, but they must follow the MODEL statement. Each statement can optionally include an identifying *label*. If several EXACT statements are specified, any statement without a label is assigned a label of the form "Exact*n*," where *n* indicates the *n*th EXACT statement. The label is included in the headers of the displayed exact analysis tables.

If a STRATA statement is also specified, then a stratified exact logistic regression or a stratified exact Poisson regression is performed. The model contains a different intercept for each stratum, and these intercepts are conditioned out of the model along with any other nuisance parameters (parameters for effects specified in the MODEL statement that are not in the EXACT statement).

The ASSESSMENT, BAYES, CONTRAST, EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, OUTPUT, SLICE, and STORE statements are not available with an exact analysis. Exact

analyses are not performed when you specify a WEIGHT statement, or a model other than LINK=LOGIT with DIST=BIN or LINK=LOG with DIST=POISSON. An OFFSET= variable is not available with exact logistic regression. Exact estimation is not available for ordinal response models.

For classification variables, use of the reference parameterization is recommended.

The following *options* can be specified in each EXACT statement after a slash (/):

ALPHA=*number*
specifies the level of significance α for $100(1-\alpha)\%$ confidence limits for the parameters or odds ratios. The value of *number* must be between 0 and 1. By default, *number* is equal to the value of the ALPHA= option in the MODEL statement, or 0.05 if that option is not specified.

CLTYPE=EXACT | MIDP
requests either the exact or mid-p confidence intervals for the parameter estimates. By default, the exact intervals are produced. The confidence coefficient can be specified with the ALPHA= option. The mid-p interval can be modified with the MIDPFACTOR= option. See the section "Exact Logistic and Exact Poisson Regression" on page 3699 for details.

ESTIMATE < =*keyword* **>**
estimates the individual parameters (conditioned on all other parameters) for the effects specified in the EXACT statement. For each parameter, a point estimate, a standard error, a confidence interval, and a p-value for a two-sided test that the parameter is zero are displayed. Note that the two-sided p-value is twice the one-sided p-value. You can optionally specify one of the following *keywords*:

PARM specifies that the parameters be estimated. This is the default.

ODDS specifies that the odds ratios be estimated. If you have classification variables, then you must also specify the PARAM=REF option in the CLASS statement.

BOTH specifies that both the parameters and odds ratios be estimated.

JOINT
performs the joint test that all of the parameters are simultaneously equal to zero, performs individual hypothesis tests for the parameter of each continuous variable, and performs joint tests for the parameters of each classification variable. The joint test is indicated in the "Conditional Exact Tests" table by the label "Joint."

JOINTONLY
performs only the joint test of the parameters. The test is indicated in the "Conditional Exact Tests" table by the label "Joint." When this option is specified, individual tests for the parameters of each continuous variable and joint tests for the parameters of the classification variables are not performed.

MIDPFACTOR=δ_1 | (δ_1, δ_2)
sets the tie factors used to produce the mid-p hypothesis statistics and the mid-p confidence intervals. δ_1 modifies both the hypothesis tests and confidence intervals, while δ_2 affects only the hypothesis tests. By default, $\delta_1 = 0.5$ and $\delta_2 = 1.0$. See the section "Exact Logistic and Exact Poisson Regression" on page 3699 for details.

ONESIDED

requests one-sided confidence intervals and *p*-values for the individual parameter estimates and odds ratios. The one-sided *p*-value is the smaller of the left- and right-tail probabilities for the observed sufficient statistic of the parameter under the null hypothesis that the parameter is zero. The two-sided *p*-values (default) are twice the one-sided *p*-values. See the section "Exact Logistic and Exact Poisson Regression" on page 3699 for more details.

OUTDIST=*SAS-data-set*

names the SAS data set that contains the exact conditional distributions. This data set contains all of the exact conditional distributions that are required to process the corresponding EXACT statement. This data set contains the possible sufficient statistics for the parameters of the effects specified in the EXACT statement, the counts, and, when hypothesis tests are performed on the parameters, the probability of occurrence and the score value for each sufficient statistic. When you request an OUTDIST= data set, the observed sufficient statistics are displayed in the "Sufficient Statistics" table. See the section "OUTDIST= Output Data Set" on page 3700 for more information.

EXACT Statement Examples

In the following example, two exact tests are computed: one for x1 and the other for x2. The test for x1 is based on the exact conditional distribution of the sufficient statistic for the x1 parameter given the observed values of the sufficient statistics for the intercept, x2, and x3 parameters; likewise, the test for x2 is conditional on the observed sufficient statistics for the intercept, x1, and x3.

```
proc genmod;
   model y= x1 x2 x3/d=b;
   exact x1 x2;
run;
```

PROC GENMOD determines, from all the specified EXACT statements, the distinct conditional distributions that need to be evaluated. For example, there is only one exact conditional distribution for the following two EXACT statements:

```
exact 'One' x1 / estimate=parm;
exact 'Two' x1 / estimate=parm onesided;
```

For each EXACT statement, individual tests for the parameters of the specified effects are computed unless the JOINTONLY option is specified. Consider the following EXACT statements:

```
exact 'E12' x1 x2 / estimate;
exact 'E1'  x1    / estimate;
exact 'E2'  x2    / estimate;
exact 'J12' x1 x2 / joint;
```

In the E12 statement, the parameters for x1 and x2 are estimated and tested separately. Specifying the E12 statement is equivalent to specifying both the E1 and E2 statements. In the J12 statement, the joint test for the parameters of x1 and x2 is computed in addition to the individual tests for x1 and x2.

EXACTOPTIONS Statement

EXACTOPTIONS *options* **;**

The EXACTOPTIONS statement specifies *options* that apply to every EXACT statement in the program. The following *options* are available:

ABSFCONV=*value*

specifies the absolute function convergence criterion. Convergence requires a small change in the log-likelihood function in subsequent iterations,

$$|l_i - l_{i-1}| < value$$

where l_i is the value of the log-likelihood function at iteration i.

By default, ABSFCONV=1E–12. You can also specify the FCONV= and XCONV= criteria; optimizations are terminated as soon as one criterion is satisfied.

ADDTOBS

adds the observed sufficient statistic to the sampled exact distribution if the statistic was not sampled. This option has no effect unless the METHOD=NETWORKMC option is specified and the ESTIMATE option is specified in the EXACT statement. If the observed statistic has not been sampled, then the parameter estimate does not exist; by specifying this option, you can produce (biased) estimates.

BUILDSUBSETS

builds every distribution for sampling. By default, some exact distributions are created by taking a subset of a previously generated exact distribution. When the METHOD=NETWORKMC option is invoked, this subsetting behavior has the effect of using fewer than the desired n samples; see the N= option for more details. Use the BUILDSUBSETS option to suppress this subsetting.

EPSILON=*value*

controls how the partial sums $\sum_{i=1}^{j} y_i \mathbf{x}_i$ are compared. *value* must be between 0 and 1; by default, *value*=1E–8.

FCONV=*value*

specifies the relative function convergence criterion. Convergence requires a small relative change in the log-likelihood function in subsequent iterations,

$$\frac{|l_i - l_{i-1}|}{|l_{i-1}| + 1\text{E–}6} < value$$

where l_i is the value of the log likelihood at iteration i.

By default, FCONV=1E–8. You can also specify the ABSFCONV= and XCONV= criteria; if you specify more than one criterion, then optimizations are terminated as soon as one criterion is satisfied.

MAXTIME=*seconds*

specifies the maximum clock time (in seconds) that PROC GENMOD can use to calculate the exact distributions. If the limit is exceeded, the procedure halts all computations and prints a note to the SAS log. The default maximum clock time is seven days.

METHOD=*keyword*

specifies which exact conditional algorithm to use for every EXACT statement specified. You can specify one of the following *keywords*:

DIRECT invokes the multivariate shift algorithm of Hirji, Mehta, and Patel (1987). This method directly builds the exact distribution, but it can require an excessive amount of memory in its intermediate stages. METHOD=DIRECT is invoked by default when you are conditioning out at most the intercept.

NETWORK invokes an algorithm described in Mehta, Patel, and Senchaudhuri (1992). This method builds a network for each parameter that you are conditioning out, combines the networks, then uses the multivariate shift algorithm to create the exact distribution. The NETWORK method can be faster and require less memory than the DIRECT method. The NETWORK method is invoked by default for most analyses.

NETWORKMC invokes the hybrid network and Monte Carlo algorithm of Mehta, Patel, and Senchaudhuri (1992). This method creates a network, then samples from that network; this method does not reject any of the samples at the cost of using a large amount of memory to create the network. METHOD=NETWORKMC is most useful for producing parameter estimates for problems that are too large for the DIRECT and NETWORK methods to handle and for which asymptotic methods are invalid—for example, for sparse data on a large grid.

N=*n*

specifies the number of Monte Carlo samples to take when you specify the METHOD=NETWORKMC option. By default, $n = 10{,}000$. If PROC GENMOD cannot obtain n samples because of a lack of memory, then a note is printed in the SAS log (the number of valid samples is also reported in the listing) and the analysis continues.

The number of samples used to produce any particular statistic might be smaller than n. For example, let $X1$ and $X2$ be continuous variables, denote their joint distribution by $f(X1,X2)$, and let $f(X1 \mid X2 = x2)$ denote the marginal distribution of $X1$ conditioned on the observed value of $X2$. If you request the JOINT test of $X1$ and $X2$, then n samples are used to generate the estimate $\hat{f}(X1,X2)$ of $f(X1,X2)$, from which the test is computed. However, the parameter estimate for $X1$ is computed from the subset of $\hat{f}(X1,X2)$ that has $X2 = x2$, and this subset need not contain n samples. Similarly, the distribution for each level of a classification variable is created by extracting the appropriate subset from the joint distribution for the CLASS variable.

In some cases, the marginal sample size can be too small to admit accurate estimation of a particular statistic; a note is printed in the SAS log when a marginal sample size is less than 100. Increasing n increases the number of samples used in a marginal distribution; however, if you want to control the sample size exactly, you can either specify the BUILDSUBSETS option or do both of the following:

- Remove the JOINT option from the EXACT statement.
- Create dummy variables in a DATA step to represent the levels of a CLASS variable, and specify them as independent variables in the MODEL statement.

NOLOGSCALE

specifies that computations for the exact conditional models be computed by using normal scaling. Log scaling can handle numerically larger problems than normal scaling; however, computations in the log scale are slower than computations in normal scale.

ONDISK

uses disk space instead of random access memory to build the exact conditional distribution. Use this option to handle larger problems at the cost of slower processing.

SEED=*seed*

specifies the initial seed for the random number generator used to take the Monte Carlo samples when you specify the METHOD=NETWORKMC option. The value of the SEED= option must be an integer. If you do not specify a seed, or if you specify a value less than or equal to 0, then PROC GENMOD uses the time of day from the computer's clock to generate an initial seed.

STATUSN=*number*

prints a status line in the SAS log after every *number* of Monte Carlo samples when you specify the METHOD=NETWORKMC option. The number of samples that are taken and the current exact p-value for testing the significance of the model are displayed. You can use this status line to track the progress of the computation of the exact conditional distributions.

STATUSTIME=*seconds*

specifies the time interval (in seconds) for printing a status line in the SAS log. You can use this status line to track the progress of the computation of the exact conditional distributions. The time interval that you specify is approximate; the actual time interval varies. By default, no status reports are produced.

XCONV=*value*

specifies the relative parameter convergence criterion. Convergence requires a small relative parameter change in subsequent iterations,

$$\max_j |\delta_j^{(i)}| < value$$

where

$$\delta_j^{(i)} = \begin{cases} \beta_j^{(i)} - \beta_j^{(i-1)} & |\beta_j^{(i-1)}| < 0.01 \\ \frac{\beta_j^{(i)} - \beta_j^{(i-1)}}{\beta_j^{(i-1)}} & \text{otherwise} \end{cases}$$

and $\beta_j^{(i)}$ is the estimate of the jth parameter at iteration i.

By default, XCONV=1E–4. You can also specify the ABSFCONV= and FCONV= criteria; if more than one criterion is specified, then optimizations are terminated as soon as one criterion is satisfied.

FREQ Statement

FREQ *variable* **;**

FREQUENCY *variable* **;**

The *variable* in the FREQ statement identifies a variable in the input data set containing the frequency of occurrence of each observation. PROC GENMOD treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If it is not an integer, the frequency value is truncated to an integer. If it is less than 1 or missing, the observation is not used. In the case of models fit with generalized estimating equations (GEEs), the frequencies apply to the subject/cluster and therefore must be the same for all observations within each subject.

FWDLINK Statement

FWDLINK *variable*=*expression* **;**

You can define a link function other than a built-in link function by using the FWDLINK statement. If you use the MODEL statement option LINK= to specify a link function, you do not need to use the FWDLINK statement. The *variable* identifies the link function to the procedure. The *expression* can be any arithmetic expression supported by the DATA step language, and it is used to define the functional dependence on the mean.

Alternatively, the link function can be defined by using programming statements (see the section "Programming Statements" on page 3645) and assigned to a variable, which is then listed as the *expression*. The second form is convenient for using complex statements such as IF-THEN/ELSE clauses. The GENMOD procedure automatically computes derivatives of the link function required for iterative fitting. You must specify the inverse of the link function in the INVLINK statement when you specify the FWDLINK statement to define the link function. You use the automatic variable _MEAN_ to represent the mean in the preceding *expression*.

INVLINK Statement

INVLINK *variable*=*expression* **;**

If you define a link function in the FWDLINK statement, then you must define the inverse link function by using the INVLINK statement. If you use the MODEL statement option LINK= to specify a link function, you do not need to use the INVLINK statement. The *variable* identifies the inverse link function to the procedure. The *expression* can be any arithmetic expression supported by the DATA step language, and it is used to define the functional dependence on the linear predictor.

Alternatively, the inverse link function can be defined using programming statements (see the section "Programming Statements" on page 3645) and assigned to a variable, which is then listed as the *expression*. The second form is convenient for using complex statements such as IF-THEN/ELSE clauses. The automatic variable _XBETA_ represents the linear predictor in the preceding *expression*.

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 51.5 summarizes the *options* available in the LSMEANS statement. If you specify the BAYES statement, the ADJUST=, STEPDOWN, and LINES options are ignored. The PLOTS= option is not available for a maximum likelihood analysis; it is available only for a Bayesian analysis.

If you specify a zero-inflated model (that is, a model for either the zero-inflated Poisson or the zero-inflated negative binomial distribution), then the least squares means are computed only for effects in the model for the distribution mean, and not for effects in the zero-inflation probability part of the model.

Table 51.5 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers

Table 51.5 *continued*

Option	Description
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For details about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 51.6 summarizes the *options* available in the LSMESTIMATE statement.

Table 51.6 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference

Table 51.6 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MARGINS Statement

MARGINS *model-effects* < / *options* > **;**

The MARGINS statement computes and compares predictive margins of fixed effects. The predictive margin for a specific level (group) of a fixed effect represents the average predicted response if all the observations in the data set were in that group (Lane and Nelder 1982; Chang, Gelman, and Pagano 1982). You can compute predictive margins for any effect in the MODEL statement that involves only classification variables.

Table 51.7 summarizes the *options* available in the MARGINS statement.

Table 51.7 MARGINS Statement Options

Option	Description
Construction and Computation of Predictive Margins	
AT	Modifies the covariate value in computing predictive margins
DIFF	Computes differences of predictive margins
SLICEBY=	Partitions tests of interaction effects
SLICEDIFF	Computes differences of sliced predictive margins and determines the types of differences
USEWEIGHT	Specifies whether to use WEIGHT variables to compute predictive margins

Table 51.7 *continued*

Option	Description
Degrees of Freedom and *p*-Values	
ADJUST=	Specifies the method of multiple comparison adjustment of predictive margin differences
ALPHA=α	Specifies the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion
Statistical Output	
CL	Constructs confidence limits for predictive margins and/or predictive margin differences

For more information about the syntax of the MARGINS statement, see the section "MARGINS Statement" on page 525 in Chapter 20, "Shared Concepts and Topics."

NOTE: By default, chi-square and z tests are performed, and weights are not used in the computations. If your model has classification variables, then the MARGINS statement is allowed only if you also specify the PARAM=GLM option. The MARGINS statement cannot be used with zero-inflated distributions, Bayesian analysis, or exact analysis.

MODEL Statement

MODEL *response* < *(variable-options)* > **=** < *effects* > < / *options* > **;**

MODEL *events/trials* **=** < *effects* > < / *options* > **;**

The MODEL statement specifies the response, or dependent variable, and the effects, or explanatory variables. If you omit the explanatory variables, the procedure fits an intercept-only model. An intercept term is included in the model by default. You can remove the intercept by specifying the NOINT option.

You can specify the response in the form of a single variable or in the form of a ratio of two variables denoted by *events/trials*. The first form is applicable to all responses. The second form is applicable only to summarized binomial response data. When each observation in the input data set contains the number of events (for example, successes) and the number of trials from a set of binomial trials, use the *events/trials* syntax.

In the *events/trials* model syntax, you specify two variables that contain the event and trial counts. These two variables are separated by a slash (/). The values of both *events* and (*trials–events*) must be nonnegative, and the value of the *trials* variable must be greater than 0 for an observation to be valid. The variable *events* or *trials* can take noninteger values.

When each observation in the input data set contains a single trial from a binomial or multinomial experiment, use the first form of the preceding MODEL statements. The response variable can be numeric or character. Variable options specific to the response variable can be specified in parentheses immediately after the response variable. Identifying the event level for binomial responses and ordering of response levels for multinomial responses are critical in these models. You can use the response variable options to do this.

Responses for the Poisson distribution must be all nonnegative, but they can be noninteger values.

The effects in the MODEL statement consist of an explanatory variable or combination of variables. Explanatory variables can be continuous or classification variables. Classification variables can be character or numeric. Explanatory variables representing nominal, or classification, data must be declared in a CLASS statement. Interactions between variables can also be included as effects. Columns of the design matrix are automatically generated for classification variables and interactions. The syntax for specification of effects is the same as for the GLM procedure. See the section "Specification of Effects" on page 3666 for more information. Also see Chapter 53, "The GLM Procedure."

Response Variable Options

DESCENDING | DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC GENMOD orders the response categories according to the ORDER= option and then reverses that order. For more information, see the section "Response Level Ordering" on page 3702.

EVENT='*category*' | *keyword*

specifies the event category for the binary response model. PROC GENMOD models the probability of the event category. The EVENT= option has no effect when there are more than two response categories. You can specify the value (formatted if a format is applied) of the event category in quotation marks, or you can specify one of the following *keywords*:

FIRST

designates the first-ordered category as the event.

LAST

designates the last-ordered category as the event.

By default, EVENT=FIRST.

One of the most common sets of response levels is {0,1}, where 1 represents the event for which the probability is to be modeled. Consider the example where Y takes the values 1 and 0 for event and nonevent, respectively, and Exposure is the explanatory variable. To specify the value 1 as the event category, use the following MODEL statement:

```
model Y(event='1') = Exposure;
```

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the response variable. The following table displays the available ORDER= options.

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | *keyword*

REF='*category*' | *keyword*

specifies the reference category for the binary response model. Specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotation marks, or you can specify one of the following *keywords*:

FIRST designates the first-ordered category as the reference.

LAST designates the last-ordered category as the reference.

By default, REF=LAST.

Table 51.8 summarizes the *options* available in the MODEL statement.

Table 51.8 MODEL Statement Options

Option	Description
AGGREGATE=	Specifies the subpopulations
ALPHA=	Sets the confidence coefficient
CICONV=	Sets the convergence criterion for profile likelihood confidence intervals
CL	Displays confidence limits for predicted values
CODING=	Uses effect coding for all classification variables
CONVERGE=	Sets the convergence criterion
CONVH=	Sets the relative Hessian convergence criterion
CORRB	Displays the parameter estimate correlation matrix
COVB	Displays the parameter estimate covariance matrix
DESCENDING	Reverses the order of the response categories
DIAGNOSTICS	Displays case deletion diagnostic statistics
DIST=	Specifies the built-in probability distribution
EVENT=	Specifies the event category for the binary response model
EXACTMAX	Names a variable used for performing an exact Poisson regression
EXPECTED	Computes covariances and associated statistics by using the expected Fisher information matrix
ID=	Displays the values of *variable* in the input data set in the OBSTATS table
INITIAL=	Sets initial values for parameter estimates
INTERCEPT=	Initializes the intercept term
ITPRINT	Displays the iteration history for all iterative processes
LINK=	Specifies the link function
LOGNB	Computes the maximum likelihood estimate and confidence limits of k-based $\log(k)$
LRCI	Computes two-sided confidence intervals for the partially likelihood function

Table 51.8 *continued*

Option	Description
MAXITER=	Sets the maximum allowable number of iterations for all iterative computation processes
NOINT	Requests that no intercept term
NOLOGNB	Computes the maximum likelihood estimate and confidence limits of k based on k
NOSCALE	Holds the scale parameter fixed
OBSTATS	Displays an additional table of statistics
OFFSET=	Specifies a variable in the input data set to be used as an offset
ORDER=	Specifies the sort order for the levels of the response variable
PREDICTED	Displays predicted values and associated statistics
REFERENCE=	Specifies the reference category for the binary response model
RESIDUALS	Displays residuals and standardized residuals
SCALE=	Sets the value used for the scale
SCORING=	Computes the Hessian matrix using the Fisher scoring method
SINGULAR=	Sets the tolerance for testing singularity
TYPE1	Performs a Type 1 analysis
TYPE3	Computes statistics for Type 3 contrasts
WALD	Requests Wald statistics for Type 3 contrasts
WALDCI	Computes two-sided Wald confidence intervals
XVARS	Includes the regression variables in the OBSTATS table

You can specify the following *options* in the MODEL statement after a slash (/).

AGGREGATE= (*variable-list***) | *variable***

AGGREGATE

specifies the subpopulations on which the Pearson chi-square and the deviance are calculated. This option applies only to the multinomial distribution or the binomial distribution with binary (single trial syntax) response. It is ignored if specified for other cases. Observations with common values in the given list of variables are regarded as coming from the same subpopulation. This affects the computation of the deviance and Pearson chi-square statistics. Variables in the list can be any variables in the input data set. Specifying the AGGREGATE option is equivalent to specifying the AGGREGATE= option with a variable list that includes all explanatory variables in the MODEL statement. Pearson chi-square and deviance statistics are not computed for multinomial models unless this option is specified.

ALPHA=*number*

ALPH=*number*

A=*number*

sets the confidence coefficient for parameter confidence intervals to 1−*number*. The value of *number* must be between 0 and 1. The default value of *number* is 0.05.

CICONV=*number*

sets the convergence criterion for profile likelihood confidence intervals. See the section "Confidence Intervals for Parameters" on page 3670 for the definition of convergence. The value of *number* must be between 0 and 1. By default, CICONV=1E–4.

CL

requests that confidence limits for predicted values be displayed (see the OBSTATS option).

CODING=EFFECT | FULLRANK

specifies that effect coding be used for all classification variables in the model. This is the same as specifying PARAM=EFFECT as a CLASS statement option.

CONVERGE=*number*

sets the convergence criterion. The value of *number* must be between 0 and 1. The iterations are considered to have converged when the maximum change in the parameter estimates between iteration steps is less than the value specified. The change is a relative change if the parameter is greater than 0.01 in absolute value; otherwise, it is an absolute change. By default, CONVERGE=1E–4. This convergence criterion is used in parameter estimation for a single model fit, Type 1 statistics, and likelihood ratio statistics for Type 3 analyses and CONTRAST statements.

CONVH=*number*

sets the relative Hessian convergence criterion. The value of *number* must be between 0 and 1. After convergence is determined with the change in parameter criterion specified with the CONVERGE= option, the quantity $tc = \frac{\mathbf{g}'\mathbf{H}^{-1}\mathbf{g}}{|f|}$ is computed and compared to *number*, where $\mathbf{g}$ is the gradient vector, $\mathbf{H}$ is the Hessian matrix for the model parameters, and f is the log-likelihood function. If tc is greater than *number*, a warning that the relative Hessian convergence criterion has been exceeded is printed. This criterion detects the occasional case where the change in parameter convergence criterion is satisfied, but a maximum in the log-likelihood function has not been attained. By default, CONVH=1E–4.

CORRB

requests that the parameter estimate correlation matrix be displayed.

COVB

requests that the parameter estimate covariance matrix be displayed.

DIAGNOSTICS

INFLUENCE

requests that case deletion diagnostic statistics be displayed (see the OBSTATS option).

DIST=*keyword*

D=*keyword*

ERROR=*keyword*

ERR=*keyword*

specifies the built-in probability distribution to use in the model. If you specify the DIST= option and you omit a user-defined link function, a default link function is chosen as displayed in the following table. If you specify no distribution and no link function, then the GENMOD procedure defaults to the normal distribution with the identity link function. Models for data with correlated responses fit by the GEE method are not available for the zero-inflated distributions.

DIST=	Distribution	Default Link Function
BINOMIAL | BIN | B	Binomial	Logit
GAMMA | GAM | G	Gamma	Inverse (power(–1))
GEOMETRIC | GEOM	Geometric	Log
IGAUSSIAN | IG	Inverse Gaussian	Inverse squared (power(–2))
MULTINOMIAL | MULT	Multinomial	Cumulative logit
NEGBIN | NB	Negative binomial	Log
NORMAL | NOR | N	Normal	Identity
POISSON | POI | P	Poisson	Log
TWEEDIE< (*Tweedie-options*) >	Tweedie	Log
ZIP	Zero-inflated Poisson	Log/logit
ZINB	Zero-inflated negative binomial	Log/logit

You can specify the following *Tweedie-options* when you specify DIST=TWEEDIE.

INITIALP=*starting-value*
specifies a starting value for iterative estimation of the Tweedie power parameter.

P=*power-parameter*
specifies a fixed Tweedie power parameter.

EPSILON=*tolerance*
specifies the tolerance for series approximation of the Tweedie density function.

OFFSET=*constant-value*
specifies a constant value to be added to the response variable for evaluating the extended quasi-likelihood. By default, OFFSET=0.5.

NTHREADS=*number*
specifies the number of threads to be used in computation.

EXACTMAX< =*variable* **>**
names a variable to be used for performing an exact Poisson regression. For each observation, the integer part of the EXACTMAX value should be nonnegative and at least as large as the response value. If the EXACTMAX option is specified without a variable, then default values are computed. See the section "Exact Logistic and Exact Poisson Regression" on page 3699 for information about using this option.

EXPECTED
requests that the expected Fisher information matrix be used to compute parameter estimate covariances and the associated statistics. The default action is to use the observed Fisher information matrix. This option does not affect the model fitting, only the way in which the covariance matrix is computed (see the SCORING= option.)

ID=*variable*

causes the values of *variable* in the input data set to be displayed in the OBSTATS table. If an explicit format for *variable* has been defined, the formatted values are displayed. If the OBSTATS option is not specified, this option has no effect.

INITIAL=*numbers*

sets initial values for parameter estimates in the model. The default initial parameter values are weighted least squares estimates based on using the response data as the initial mean estimate. This option can be useful in case of convergence difficulty. The intercept parameter is initialized with the INTERCEPT= option and is not included here. The values are assigned to the variables in the MODEL statement in the same order in which they appear in the MODEL statement. The order of levels for CLASS variables is determined by the ORDER= option. Note that some levels of classification variables can be aliased; that is, they correspond to linearly dependent parameters that are not estimated by the procedure. Initial values must be assigned to all levels of classification variables, regardless of whether they are aliased or not. The procedure ignores initial values corresponding to parameters not being estimated. If you specify a BY statement, all classification variables must take on the same number of levels in each BY group. Otherwise, classification variables in some of the BY groups are assigned incorrect initial values. Types of INITIAL= specifications are illustrated in the following table.

Type of List	Specification
List separated by blanks	INITIAL = 3 4 5
List separated by commas	INITIAL = 3, 4, 5
x to y	INITIAL = 3 to 5
x to y by z	INITIAL = 3 to 5 by 1
Combination of list types	INITIAL = 1, 3 to 5, 9

INTERCEPT=*number* | *number-list*

initializes the intercept term to *number* for parameter estimation. If you specify both the INTERCEPT= and the NOINT options, the intercept term is not estimated, but an intercept term of *number* is included in the model. If you specify a multinomial model for ordinal data, you can specify a *number-list* for the multiple intercepts in the model.

ITPRINT

displays the iteration history for all iterative processes: parameter estimation, fitting constrained models for contrasts and Type 3 analyses, and profile likelihood confidence intervals. The last evaluation of the gradient and the negative of the Hessian (second derivative) matrix are also displayed for parameter estimation. If you perform a Bayesian analysis by specifying the BAYES statement, the iteration history for computing the mode of the posterior distribution is also displayed.

This option might result in a large amount of displayed output, especially if some of the optional iterative processes are selected.

LINK=*keyword*

specifies the link function to use in the model. The *keywords* and their associated built-in link functions are as follows.

LINK=	Link Function
CUMCLL	
CCLL	Cumulative complementary log-log
CUMLOGIT	
CLOGIT	Cumulative logit
CUMPROBIT	
CPROBIT	Cumulative probit
CLOGLOG	
CLL	Complementary log-log
IDENTITY	
ID	Identity
LOG	Log
LOGIT	Logit
PROBIT	Probit
POWER(*number*) | **POW**(*number*)	Power with λ= *number*

If no LINK= option is supplied and there is a user-defined link function, the user-defined link function is used. If you specify neither the LINK= option nor a user-defined link function, then the default canonical link function is used if you specify the DIST= option. Otherwise, if you omit the DIST= option, the identity link function is used.

The cumulative link functions are appropriate only for the multinomial distribution.

LOGNB

specifies that the maximum likelihood estimate and confidence limits of the negative binomial dispersion parameter k be computed based $\log(k)$. This is the default method used for the negative binomial dispersion parameter, so that specifying no option or specifying the LOGNB option have the same effect. The GENMOD procedure computes the maximum likelihood estimate of $\log(k)$ and computes confidence limits based on the asymptotic normality of $\log(k)$ rather than of k. The results are always reported in terms of k rather than of $\log(k)$. This method ensures that the estimate and confidence limits for k are positive. See Meeker and Escobar (1998, p. 163) for details about this method of computing confidence limits.

LRCI

requests that two-sided confidence intervals for all model parameters be computed based on the profile likelihood function. This is sometimes called the partially maximized likelihood function. See the section "Confidence Intervals for Parameters" on page 3670 for more information about the profile likelihood function. This computation is iterative and can consume a relatively large amount of CPU time. The confidence coefficient can be selected with the ALPHA=*number* option. The resulting confidence coefficient is 1−*number*. The default confidence coefficient is 0.95.

MAXITER=*number*

MAXIT=*number*

sets the maximum allowable number of iterations for all iterative computation processes in PROC GENMOD. By default, MAXITER=50.

NOINT

requests that no intercept term be included in the model. An intercept is included unless this option is specified.

NOLOGNB

specifies that the maximum likelihood estimate and confidence limits of the negative binomial dispersion parameter k be computed based on k rather than $\log(k)$. If this option is not specified, then the GENMOD procedure computes the maximum likelihood estimate of $\log(k)$ and computes confidence limits based on the asymptotic normality of $\log(k)$ rather than of k. The results are always reported in terms of k rather than of $\log(k)$. This method ensures that the estimate and confidence limits for k are positive. See Meeker and Escobar (1998, p. 163) for details about this method of computing confidence limits.

NOSCALE

holds the scale parameter fixed. Otherwise, for the normal, inverse Gaussian, and gamma distributions, the scale parameter is estimated by maximum likelihood. If you omit the SCALE= option, the scale parameter is fixed at the value 1.

OBSTATS

specifies that an additional table of statistics be displayed. Formulas for the statistics are given in the section "Predicted Values of the Mean" on page 3672, the section "Residuals" on page 3673, and the section "Case Deletion Diagnostic Statistics" on page 3690. Residuals and fit diagnostics are not computed for multinomial models.

For each observation, the following items are displayed:

- the value of the response variable (variables if the data are binomial), frequency, and weight variables
- the values of the regression variables
- predicted mean, $\hat{\mu} = g^{-1}(\eta)$, where $\eta = \mathbf{x}_i'\hat{\boldsymbol{\beta}}$ is the linear predictor and g is the link function. If there is an offset, it is included in $\mathbf{x}_i'\hat{\boldsymbol{\beta}}$, except when you have a multinomial model, for which it is not included.
- estimate of the linear predictor $\mathbf{x}_i'\hat{\boldsymbol{\beta}}$. If there is an offset, it is included in $\mathbf{x}_i'\hat{\boldsymbol{\beta}}$.
- standard error of the linear predictor $\mathbf{x}_i'\hat{\boldsymbol{\beta}}$
- the value of the Hessian weight at the final iteration
- lower confidence limit of the predicted value of the mean. The confidence coefficient is specified with the ALPHA= option. See the section "Confidence Intervals on Predicted Values" on page 3672 for the computational method.
- upper confidence limit of the predicted value of the mean
- raw residual, defined as $Y - \mu$
- Pearson, or chi residual, defined as the square root of the contribution for the observation to the Pearson chi-square—that is,

  $$\frac{Y - \mu}{\sqrt{V(\mu)/w}}$$

  where Y is the response, μ is the predicted mean, w is the value of the prior weight variable specified in a WEIGHT statement, and $\mathrm{V}(\mu)$ is the variance function evaluated at μ.

- the standardized Pearson residual
- deviance residual, defined as the square root of the deviance contribution for the observation, with sign equal to the sign of the raw residual
- the standardized deviance residual
- the likelihood residual
- a Cook distance–type statistic for assessing the influence of individual observations on overall model fit
- observation leverage
- DFBETA, defined as an approximation to $\hat{\beta} - \hat{\beta_{[i]}}$ for each parameter estimate $\hat{\beta}$, where $\hat{\beta_{[i]}}$ is the parameter estimate with the ith observation deleted
- standardized DFBETA, defined as DFBETA, normalized by its standard deviation
- zero inflation probability for zero-inflated models
- the mean of a zero-inflated response

The following additional cluster deletion diagnostic statistics are created and displayed for each cluster if a REPEATED statement is specified:

- a Cook distance–type statistic for assessing the influence of entire clusters on overall model fit
- a studentized Cook distance–type statistic for assessing influence of clusters
- cluster leverage
- cluster DFBETA for assessing the influence of entire clusters on individual parameter estimates
- cluster DFBETA normalized by its standard deviation

If you specify the multinomial distribution, only regression variable values, response values, predicted values, confidence limits for the predicted values, and the linear predictor are displayed in the table. Residuals and other diagnostic statistics are not available for the multinomial distribution.

The RESIDUALS, DIAGNOSTICS | INFLUENCE, PREDICTED, XVARS, and CL options cause only subgroups of the observation statistics to be displayed. You can specify more than one of these *options* to include different subgroups of statistics.

The ID=*variable* option causes the values of *variable* in the input data set to be displayed in the table. If an explicit format for *variable* has been defined, the formatted values are displayed.

If a REPEATED statement is present, a table is displayed for the GEE model specified in the REPEATED statement. Regression variables, response values, predicted values, confidence limits for the predicted values, linear predictor, raw residuals, Pearson residuals for each observation in the input data set are available. Case deletion diagnostic statistics are available for each observation and for each cluster.

OFFSET=*variable*

specifies a variable in the input data set to be used as an offset variable. This variable cannot be a CLASS variable, and it cannot be the response variable or one of the explanatory variables.

When you perform an exact Poisson regression with an OFFSET= variable but the EXACTMAX= option is not specified, then if o_i is the offset for the ith observation, floor(exp(o_i)) should be greater than or equal to the response value. See the section "Exact Logistic and Exact Poisson Regression" on page 3699 for information about the use of the offset in the exact Poisson model.

PREDICTED

PRED

P

requests that predicted values, the linear predictor, its standard error, and the Hessian weight be displayed (see the OBSTATS option).

RESIDUALS

R

requests that residuals and standardized residuals be displayed. Residuals and other diagnostic statistics are not available for the multinomial distribution (see the OBSTATS option).

SCALE=*number*

SCALE=PEARSON | P

PSCALE

SCALE=DEVIANCE | D

DSCALE

sets the value used for the scale parameter where the NOSCALE option is used. For the binomial and Poisson distributions, which have no free scale parameter, this can be used to specify an *overdispersed* model. In this case, the parameter covariance matrix and the likelihood function are adjusted by the scale parameter. See the section "Dispersion Parameter" on page 3665 and the section "Overdispersion" on page 3665 for more information. If the NOSCALE option is not specified, then *number* is used as an initial estimate of the scale parameter.

Specifying SCALE=PEARSON or SCALE=P is the same as specifying the PSCALE option. This fixes the scale parameter at the value 1 in the estimation procedure. After the parameter estimates are determined, the exponential family dispersion parameter is assumed to be given by Pearson's chi-square statistic divided by the degrees of freedom, and all statistics such as standard errors and likelihood ratio statistics are adjusted appropriately.

Specifying SCALE=DEVIANCE or SCALE=D is the same as specifying the DSCALE option. This fixes the scale parameter at a value of 1 in the estimation procedure.

After the parameter estimates are determined, the exponential family dispersion parameter is assumed to be given by the deviance divided by the degrees of freedom. All statistics such as standard errors and likelihood ratio statistics are adjusted appropriately.

SCORING=*number*

requests that on iterations up to *number*, the Hessian matrix be computed using the Fisher scoring method. For further iterations, the full Hessian matrix is computed. The default value is 1. A value of 0 causes all iterations to use the full Hessian matrix, and a value greater than or equal to the value of the MAXITER option causes all iterations to use Fisher scoring. The value of the SCORING= option must be 0 or a positive integer.

SINGULAR=*number*

sets the tolerance for testing singularity of the information matrix and the crossproducts matrix. Roughly, the test requires that a pivot be at least this number times the original diagonal value. By default, *number* is 10^7 times the machine epsilon. The default *number* is approximately 10^{-9} on most machines. This value also controls the check on estimability for ESTIMATE and CONTRAST statements.

TYPE1

requests that a Type 1, or sequential, analysis be performed. This consists of sequentially fitting models, beginning with the null (intercept term only) model and continuing up to the model specified in the MODEL statement. The likelihood ratio statistic between each successive pair of models is computed and displayed in a table.

A Type 1 analysis is not available for GEE models, since there is no associated likelihood.

TYPE3

requests that statistics for Type 3 contrasts be computed for each effect specified in the MODEL statement. The default analysis is to compute likelihood ratio statistics for the contrasts or score statistics for GEEs. Wald statistics are computed if the WALD option is also specified.

WALD

requests Wald statistics for Type 3 contrasts. You must also specify the TYPE3 option in order to compute Type 3 Wald statistics.

WALDCI

requests that two-sided Wald confidence intervals for all model parameters be computed based on the asymptotic normality of the parameter estimators. This computation is not as time-consuming as the LRCI method, since it does not involve an iterative procedure. However, it is thought to be less accurate, especially for small sample sizes. The confidence coefficient can be selected with the ALPHA= option in the same way as for the LRCI option.

XVARS

requests that the regression variables be included in the OBSTATS table.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *keyword***=***name* . . . *keyword***=***name* > **;**

The OUTPUT statement creates a new SAS data set that contains all the variables in the input data set and, optionally, the estimated linear predictors (XBETA) and their standard error estimates, the weights for the Hessian matrix, predicted values of the mean, confidence limits for predicted values, residuals, and case deletion diagnostics. Residuals and diagnostic statistics are not computed for multinomial models.

You can also request these statistics with the OBSTATS, PREDICTED, RESIDUALS, DIAGNOSTICS | INFLUENCE, CL, or XVARS option in the MODEL statement. You can then create a SAS data set containing them with ODS OUTPUT commands.

You might prefer to specify the OUTPUT statement for requesting these statistics since the following are true:

- The OUTPUT statement produces no tabular output.
- The OUTPUT statement creates a SAS data set more efficiently than ODS. This can be an advantage for large data sets.
- You can specify the individual statistics to be included in the SAS data set.

If you use the multinomial distribution with one of the cumulative link functions for ordinal data, the data set also contains variables named _ORDER_ and _LEVEL_ that indicate the levels of the ordinal response variable and the values of the variable in the input data set corresponding to the sorted levels. These variables indicate that the predicted value for a given observation is the probability that the response variable is as large as the value of the _LEVEL_ variable. Residuals and other diagnostic statistics are not available for the multinomial distribution.

The estimated linear predictor, its standard error estimate, and the predicted values and their confidence intervals are computed for all observations in which the explanatory variables are all nonmissing, even if the response is missing. By adding observations with missing response values to the input data set, you can compute these statistics for new observations or for settings of the explanatory variables not present in the data without affecting the model fit.

The following list explains specifications in the OUTPUT statement.

OUT=*SAS-data-set*

specifies the output data set. If you omit the OUT=option, the output data set is created and given a default name that uses the DATA*n* convention.

keyword=name

specifies the statistics to be included in the output data set and names the new variables that contain the statistics. Specify a keyword for each desired statistic (see the following list of keywords), an equal sign, and the name of the new variable or variables to contain the statistic. You can list only one variable after the equal sign for all the statistics, except for the case deletion diagnostics for individual parameter estimates, DFBETA, DFBETAS, DFBETAC, and DFBETACS. You can list variables enclosed in parentheses to correspond to the variables in the model, or you can specify the keyword *_all_*, without parentheses, to include deletion diagnostics for all of the parameters in the model.

Although you can use the OUTPUT statement without any *keyword=name* specifications, the output data set then contains only the original variables and, possibly, the variables Level and Value (if you use the multinomial model with ordinal data). Note that the residuals and deletion diagnostics are not available for the multinomial model with ordinal data. Some of the case deletion diagnostic statistics apply only to models for correlated data specified with a REPEATED statement. If you request these statistics for ordinary generalized linear models, the values of the corresponding variables are set to missing in the output data set. Formulas for the statistics are given in the section "Predicted Values of the Mean" on page 3672, the section "Residuals" on page 3673, and the section "Case Deletion Diagnostic Statistics" on page 3690.

The *keywords* allowed and the statistics they represent are as follows:

DFBETA | DBETA represents the effect of deleting an observation on parameter estimates. If you specify the keyword *_all_* after the equal sign, variables named DFBETA_*ParameterName* will be included in the output data set to contain the values of the diagnostic statistic to measure the influence of deleting a single observation on the individual parameter estimates. *ParameterName* is the name of the regression model parameter formed from the input variable names concatenated with the appropriate levels, if classification variables are involved.

DFBETAS | DBETAS represents the effect of deleting an observation on standardized parameter estimates. If you specify the keyword *_all_* after the equal sign, variables named

DFBETAS_*ParameterName* will be included in the output data set to contain the values of the diagnostic statistic to measure the influence of deleting a single observation on the individual parameter estimates. *ParameterName* is the name of the regression model parameter formed from the input variable names concatenated with the appropriate levels, if classification variables are involved.

DOBS | COOKSD represents the Cook distance–type statistic to measure the influence of deleting a single observation on the overall model fit.

HESSWGT represents the diagonal element of the weight matrix used in computing the Hessian matrix.

H | LEVERAGE represents the leverage of a single observation.

LOWER | L represents the lower confidence limit for the predicted value of the mean, or the lower confidence limit for the probability that the response is less than or equal to the value of Level or Value. The confidence coefficient is determined by the ALPHA=*number* option in the MODEL statement as $(1 - number) \times 100\%$. The default confidence coefficient is 95%.

PREDICTED | PRED | PROB | P represents the predicted value of the mean of the response or the predicted probability that the response variable is less than or equal to the value of _LEVEL_ if the multinomial model for ordinal data is used (in other words, $\Pr(\mathrm{Y} \leq _\mathrm{LEVEL}_)$, where Y is the response variable).

PZERO represents the zero-inflation probability for zero-inflated models.

RESCHI represents the Pearson (chi) residual for identifying observations that are poorly accounted for by the model.

RESDEV represents the deviance residual for identifying poorly fitted observations.

RESLIK represents the likelihood residual for identifying poorly fitted observations.

RESRAW represents the raw residual for identifying poorly fitted observations.

STDRESCHI represents the standardized Pearson (chi) residual for identifying observations that are poorly accounted for by the model.

STDRESDEV represents the standardized deviance residual for identifying poorly fitted observations.

STDXBETA represents the standard error estimate of XBETA (see the XBETA keyword).

UPPER | U represents the upper confidence limit for the predicted value of the mean, or the upper confidence limit for the probability that the response is less than or equal to the value of Level or Value. The confidence coefficient is determined by the ALPHA=*number* option in the MODEL statement as $(1 - number) \times 100\%$. The default confidence coefficient is 95%.

XBETA represents the estimate of the linear predictor $\mathbf{x}_i'\boldsymbol{\beta}$ for observation i, or $\alpha_j + \mathbf{x}_i'\boldsymbol{\beta}$, where j is the corresponding ordered value of the response variable for the multinomial model with ordinal data. If there is an offset, it is included in $\mathbf{x}_i'\boldsymbol{\beta}$.

The *keywords* in the following list apply only to models specified with a REPEATED statement, fit by generalized estimating equations (GEEs).

CLEVERAGE represents the leverage of a cluster.

CLUSTER represents the numerical cluster index, in order of sorted clusters.

DCLS | CLUSTERCOOKSD represents the Cook distance–type statistic to measure the influence of deleting an entire cluster on the overall model fit.

DFBETAC | DBETAC represents the effect of deleting an entire cluster on parameter estimates. If you specify the keyword *_all_* after the equal sign, variables named DFBETAC_*ParameterName* will be included in the output data set to contain the values of the diagnostic statistic to measure the influence of deleting the cluster on the individual parameter estimates. *ParameterName* is the name of the regression model parameter formed from the input variable names concatenated with the appropriate levels, if classification variables are involved.

DFBETACS | DBETACS represents the effect of deleting an entire cluster on normalized parameter estimates. If you specify the keyword *_all_* after the equal sign, variables named DFBETACS_*ParameterName* will be included in the output data set to contain the values of the diagnostic statistic to measure the influence of deleting the cluster on the individual parameter estimates, normalized by their standard errors. *ParameterName* is the name of the regression model parameter formed from the input variable names concatenated with the appropriate levels, if classification variables are involved.

MCLS | CLUSTERDFIT represents the studentized Cook distance–type statistic to measure the influence of deleting an entire cluster on the overall model fit.

Programming Statements

Although the most commonly used link and probability distributions are available as built-in functions, the GENMOD procedure enables you to define your own link functions and response probability distributions by using the FWDLINK, INVLINK, VARIANCE, and DEVIANCE statements. The variables assigned in these statements can have values computed in programming statements.

These programming statements can occur anywhere between the PROC GENMOD statement and the RUN statement. Variable names used in programming statements must be unique. Variables from the input data set can be referenced in programming statements. The mean, linear predictor, and response are represented by the automatic variables _MEAN_, _XBETA_, and _RESP_, respectively, which can be referenced in your programming statements. Programming statements are used to define the functional dependencies of the link function, the inverse link function, the variance function, and the deviance function on the mean, linear predictor, and response variable.

The following statements illustrate the use of programming statements. Even though you usually request the Poisson distribution by specifying DIST=POISSON as a MODEL statement option, you can define the variance and deviance functions for the Poisson distribution by using the VARIANCE and DEVIANCE statements. For example, the following statements perform the same analysis as the Poisson regression example in the section "Getting Started: GENMOD Procedure" on page 3575.

The statements must be in logical order for computation, just as in a DATA step.

```
proc genmod;
   class car age;
   a = _MEAN_;
   y = _RESP_;
   d = 2 * ( y * log( y / a ) - ( y - a ) );
   variance var = a;
   deviance dev = d;
   model c = car age / link = log offset = ln;
run;
```

The variables var and dev are dummy variables used internally by the procedure to identify the variance and deviance functions. Any valid SAS variable names can be used.

Similarly, the log link function and its inverse could be defined with the FWDLINK and INVLINK statements, as follows:

```
fwdlink link = log(_MEAN_);
invlink ilink = exp(_XBETA_);
```

These statements are for illustration, and they work well for most Poisson regression problems. If, however, in the iterative fitting process, the mean parameter becomes too close to 0, or a 0 response value occurs, an error condition occurs when the procedure attempts to evaluate the log function. You can circumvent this kind of problem by using IF-THEN/ELSE clauses or other conditional statements to check for possible error conditions and appropriately define the functions for these cases.

Data set variables can be referenced in user definitions of the link function and response distributions by using programming statements and the FWDLINK, INVLINK, DEVIANCE, and VARIANCE statements.

See the DEVIANCE, VARIANCE, FWDLINK, and INVLINK statements for more information.

The syntax of programming statements used in PROC GENMOD is identical to that used in the NLMIXED procedure and the GLIMMIX procedure (see Chapter 89, "The NLMIXED Procedure," and Chapter 52, "The GLIMMIX Procedure") and the MODEL procedure (see the *SAS/ETS User's Guide*). Most of the programming statements that can be used in the DATA step can also be used in the GENMOD procedure. See *SAS DATA Step Statements: Reference* for a description of SAS programming statements. The following are some commonly used programming statements.

ABORT;
ARRAY *arrayname* < **[** *dimensions* **]** > < **$** > < *variables-and-constants* >**;**
CALL *name* < **(***expression* < **,** *expression* . . . >**)** >**;**
DELETE;
DO < *variable* **=** *expression* < **TO** *expression* > < **BY** *expression* > >
< **,** *expression* < **TO** *expression* > < **BY** *expression* > > . . .
< **WHILE** *expression* > < **UNTIL** *expression* >**;**
END;
GOTO *statement-label***;**
IF *expression***;**
IF *expression* **THEN** *program-statement***;**
ELSE *program-statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement-label***;**
PUT < *variable* > < **=** > . . . **;**
RETURN;
SELECT < **(***expression***)** >**;**
STOP;
SUBSTR(*variable***,** *index***,** *length***)=** *expression***;**
WHEN (*expression***)***program-statement***;**
OTHERWISE *program-statement***;**

REPEATED Statement

REPEATED SUBJECT=*subject-effect* < / *options* > **;**

The REPEATED statement specifies the covariance structure of multivariate responses for GEE model fitting in the GENMOD procedure. In addition, the REPEATED statement controls the iterative fitting algorithm used in GEEs and specifies optional output. Other GENMOD procedure statements, such as the MODEL and CLASS statements, are used in the same way as they are for ordinary generalized linear models to specify the regression model for the mean of the responses.

Table 51.9 summarizes the *options* available in the REPEATED statement.

Table 51.9 REPEATED Statement Options

Option	Description
ALPHAINIT=	Specifies initial values for log odds ratio regression parameters
CONVERGE=	Specifies the convergence criterion for GEE parameter estimation
CORRB	Displays the estimated correlation matrix
CORRW	Displays the estimated working correlation matrix
COVB	Displays the estimated covariance matrix
ECORRB	Displays the estimated empirical correlation matrix
ECOVB	Displays the estimated empirical covariance matrix
INITIAL=	Specifies initial values of the regression parameters estimation
INTERCEPT=	Specifies either an initial or a fixed value of the intercept
LOGOR=	Specifies the regression structure of the log odds ratio

Table 51.9 *continued*

Option	Description
MAXITER=	Specifies the maximum number of iterations
MCORRB	Displays the estimated model-based correlation matrix
MCOVB	Displays the estimated model-based covariance matrix
MODELSE	Displays an analysis of parameter estimates table
PRINTMLE	Displays an analysis of maximum likelihood parameter estimates table
RUPDATE=	Specifies the number of iterations between updates of the working correlation matrix
SORTED	Groups by subject and sorts within subject
SUBCLUSTER=	Specifies a variable defining subclusters
SUBJECT=	Identifies a different subject, or cluster
TYPE=	Specifies the working correlation matrix structure
V6CORR	Uses the SAS 'Version 6' method of computing normalized Pearson chi-square
WITHIN=	Specifies the order of measurements within subjects
YPAIR=	Specifies the pairs of responses
ZDATA=	Specifies the full **z** matrix
ZROW=	Specifies the rows of the **z** matrix

SUBJECT=*subject-effect*

identifies subjects in the input data set. The *subject-effect* can be a single variable, an interaction effect, a nested effect, or a combination. Each distinct value, or level, of the effect identifies a different subject, or cluster. Responses from different subjects are assumed to be statistically independent, and responses within subjects are assumed to be correlated. A *subject-effect* must be specified, and variables used in defining the *subject-effect* must be listed in the CLASS statement. The input data set does not need to be sorted by subject (see the SORTED option).

The *options* control how the model is fit and what output is produced. You can specify the following *options* after a slash (/).

ALPHAINIT=*numbers*

specifies initial values for log odds ratio regression parameters if the LOGOR= option is specified for binary data. If this option is not specified, an initial value of 0.01 is used for all the parameters.

CONVERGE=*number*

specifies the convergence criterion for GEE parameter estimation. If the maximum absolute difference between regression parameter estimates is less than the value of *number* on two successive iterations, convergence is declared. If the absolute value of a regression parameter estimate is greater than 0.08, then the absolute difference normalized by the regression parameter value is used instead of the absolute difference. The default value of *number* is 0.0001.

CORRW
displays the estimated working correlation matrix. If you specify an exchangeable working correlation structure with the CORR=EXCH option, the CORRW option is not needed to view the estimated correlation, since a table is printed by default that contains the single estimated correlation.

CORRB
displays the estimated regression parameter correlation matrix. Both model-based and empirical correlations are displayed.

COVB
displays the estimated regression parameter covariance matrix. Both model-based and empirical covariances are displayed.

ECORRB
displays the estimated regression parameter empirical correlation matrix.

ECOVB
displays the estimated regression parameter empirical covariance matrix.

INTERCEPT=*number*
specifies either an initial or a fixed value of the intercept regression parameter in the GEE model. If you specify the NOINT option in the MODEL statement, then the intercept is fixed at the value of *number*.

INITIAL=*numbers*
specifies initial values of the regression parameters estimation, other than the intercept parameter, for GEE estimation. If this option is not specified, the estimated regression parameters assuming independence for all responses are used for the initial values.

LOGOR=*log-odds-ratio-structure-keyword*
specifies the regression structure of the log odds ratio used to model the association of the responses from subjects for binary data. The response syntax must be of the single variable type, the distribution must be binomial, and the data must be binary. Table 51.10 displays the log odds ratio structure *keywords* and the corresponding log odds ratio regression structures. See the section "Alternating Logistic Regressions" on page 3682 for definitions of the log odds ratio types and examples of specifying log odds ratio models. You should specify either the LOGOR= or the TYPE= option, but not both.

Table 51.10 Log Odds Ratio Regression Structures

Keyword	Log Odds Ratio Regression Structure
EXCH	Exchangeable
FULLCLUST	Fully parameterized clusters
LOGORVAR(*variable*)	Indicator variable for specifying block effects
NESTK	k-nested
NEST1	1-nested
ZFULL	Fully specified **z** matrix specified in ZDATA= data set
ZREP	Single cluster specification for replicated **z** matrix specified in ZDATA= data set
ZREP(*matrix*)	Single cluster specification for replicated **z** matrix

MAXITER=*number*

MAXIT=*number*

specifies the maximum number of iterations allowed in the iterative GEE estimation process. The default number is 50.

MCORRB

displays the estimated regression parameter model-based correlation matrix.

MCOVB

displays the estimated regression parameter model-based covariance matrix.

MODELSE

displays an analysis of parameter estimates table that uses model-based standard errors for inference. By default, an "Analysis of Parameter Estimates" table based on empirical standard errors is displayed.

PRINTMLE

displays an analysis of maximum likelihood parameter estimates table. The maximum likelihood estimates are not displayed unless this *option* is specified.

RUPDATE=*number*

specifies the number of iterations between updates of the working correlation matrix. For example, RUPDATE=5 specifies that the working correlation is updated once for every five regression parameter updates. The default value of *number* is 1; that is, the working correlation is updated every time the regression parameters are updated.

SORTED

specifies that the input data are grouped by subject and sorted within subject. If this *option* is not specified, then the procedure internally sorts by *subject-effect* and *within subject-effect*, if a *within subject-effect* is specified.

SUBCLUSTER=*variable*

SUBCLUST=*variable*

specifies a variable defining subclusters for the 1-nested or k-nested log odds ratio association modeling structures. This variable must be listed in the CLASS statement.

TYPE=*correlation-structure keyword*

CORR=*correlation-structure keyword*

specifies the structure of the working correlation matrix used to model the correlation of the responses from subjects. Table 51.11 displays the correlation structure *keywords* and the corresponding correlation structures. The default working correlation type is the independent (CORR=IND). See the section "Details: GENMOD Procedure" on page 3655 for definitions of the correlation matrix types. You should specify LOGOR= or TYPE= but not both.

Table 51.11 Correlation Structure Types

Keyword	Correlation Matrix Type
AR	
AR(1)	Autoregressive(1)
EXCH	
CS	Exchangeable
IND	Independent
MDEP(*number*)	m-dependent with m=number
UNSTR	
UN	Unstructured
USER	
FIXED(*matrix*)	Fixed, user-specified correlation matrix

For example, you can specify a fixed 4×4 correlation matrix with the following option:

```
type=user( 1.0  0.9  0.8  0.6
           0.9  1.0  0.9  0.8
           0.8  0.9  1.0  0.9
           0.6  0.8  0.9  1.0 )
```

V6CORR

specifies that the SAS 'Version 6' method of computing the normalized Pearson chi-square be used for working correlation estimation and for model-based covariance matrix scale factor.

WITHINSUBJECT | WITHIN=*within subject-effect*

defines an effect specifying the order of measurements within subjects. Each distinct level of the *within subject-effect* defines a different response from the same subject. If the data are in proper order within each subject, you do not need to specify this option.

If some measurements do not appear in the data for some subjects, this *option* properly orders the existing measurements and treats the omitted measurements as missing values. If the WITHINSUBJECT= option is not used in this situation, measurements might be improperly ordered and missing values assumed for the last measurements in a cluster.

Variables used in defining the *within subject-effect* must be listed in the CLASS statement.

YPAIR=*variable-list*

specifies the variables in the ZDATA= data set corresponding to pairs of responses for log odds ratio association modeling.

ZDATA=*SAS-data-set*

specifies a SAS data set containing either the full $\mathbf{z}$ matrix for log odds ratio association modeling or the $\mathbf{z}$ matrix for a single complete cluster to be replicated for all clusters.

ZROW=*variable-list*
specifies the variables in the ZDATA= data set corresponding to rows of the **z** matrix for log odds ratio association modeling.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

STRATA Statement

STRATA *variable* < **(***option***)** > . . . < *variable* < **(***option***)** > > < / *options* > **;**

The STRATA statement names the *variables* that define *strata* or *matched sets* to use in *stratified exact* logistic regression of binary response data, or a *stratified exact* Poisson regression of count data. An EXACT statement must also be specified.

Observations that have the same *variable* values are in the same matched set. For a stratified logistic model, you can analyze 1:1, 1:n, m:n, and general m_i:n_i matched sets where the number of cases and controls varies across strata. For a stratified Poisson model, you can have any number of observations in each stratum. At least one variable must be specified to invoke the stratified analysis, and the usual unconditional asymptotic analysis is not performed. The stratified logistic model has the form

$$\text{logit}(\pi_{hi}) = \alpha_h + \mathbf{x}'_{hi}\boldsymbol{\beta}$$

where π_{hi} is the event probability for the ith observation in stratum h with covariates $\mathbf{x}_{hi}$ and where the stratum-specific intercepts α_h are the nuisance parameters that are to be conditioned out.

STRATA variables can also be specified in the MODEL statement as classification or continuous covariates; however, the effects are nondegenerate only when crossed with a nonstratification variable. Specifying several STRATA statements is the same as specifying one STRATA statement that contains all the strata variables. The STRATA variables can be either character or numeric, and the formatted values of the STRATA variables

determine the levels. Thus, you can also use formats to group values into levels; see the discussion of the FORMAT procedure in the *Base SAS Procedures Guide*.

The "Strata Summary" table is displayed by default. For an exact logistic regression, it displays the number of strata that have a specific number of events and non-events. For example, if you are analyzing a 1:5 matched study, this table enables you to verify that every stratum in the analysis has exactly one event and five non-events. Strata that contain only events or only non-events are reported in this table, but such strata are uninformative and are not used in the analysis. For an exact Poisson regression, the "Strata Summary" table displays the number of strata that contain a specific number of observations, which enables you to check whether every stratum in the analysis has the same number of observations.

The ASSESSMENT, BAYES, CONTRAST, EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, OUTPUT, REPEATED, SLICE, and STORE statements are not available with a STRATA statement. Exact analyses are not performed when you specify a WEIGHT statement, or a model other than LINK=LOGIT with DIST=BIN or LINK=LOG with DIST=POISSON. An OFFSET= variable is not available with exact logistic regression.

The following *option* can be specified for a stratification variable by enclosing the *option* in parentheses after the variable name, or it can be specified globally for all STRATA variables after a slash (/).

MISSING
treats missing values ('.', ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid STRATA variable values.

The following strata *options* are also available after the slash:

CHECKDEPENDENCY=*keyword*
CHECK=*keyword*
specifies which variables are to be tested for dependency before the analysis is performed. The available *keywords* are as follows:

NONE performs no dependence checking. Typically, a message about a singular information matrix is displayed if you have dependent variables. Dependent variables can be identified after the analysis by noting any missing parameter estimates.

COVARIATES checks dependence between covariates and an added intercept. Dependent covariates are removed from the analysis. However, covariates that are linear functions of the strata variable might not be removed, which results in a singular information matrix message being displayed in the SAS log. This is the default.

ALL checks dependence between all the strata and covariates. This *option* can adversely affect performance if you have a large number of strata.

NOSUMMARY
suppresses the display of the "Strata Summary" table.

INFO
displays the "Strata Information" table, which includes the stratum number, levels of the STRATA variables that define the stratum, and the total frequency for each stratum. Since the number of strata can be very large, this table is displayed only by request.

VARIANCE Statement

VARIANCE *variable* **=** *expression* **;**

You can specify a probability distribution other than the built-in distributions by using the VARIANCE and DEVIANCE statements. The variable name *variable* identifies the variance function to the procedure. The *expression* is used to define the functional dependence on the mean, and it can be any arithmetic expression supported by the DATA step language. You use the automatic variable _MEAN_ to represent the mean in the expression.

Alternatively, you can define the variance function with programming statements, as detailed in the section "Programming Statements" on page 3645. This form is convenient for using complex statements such as IF-THEN/ELSE clauses. Derivatives of the variance function for use during optimization are computed automatically. The DEVIANCE statement must also appear when the VARIANCE statement is used to define the variance function.

WEIGHT Statement

WEIGHT | SCWGT *variable* **;**

The WEIGHT statement identifies a *variable* in the input data set to be used as the exponential family dispersion parameter weight for each observation. The exponential family dispersion parameter is divided by the WEIGHT variable value for each observation. This is true regardless of whether the parameter is estimated by the procedure or specified in the MODEL statement with the SCALE= option. It is also true for distributions such as the Poisson and binomial that are not usually defined to have a dispersion parameter. For these distributions, a WEIGHT variable weights the overdispersion parameter, which has the default value of 1.

The WEIGHT variable does not have to be an integer; if it is less than or equal to 0 or if it is missing, the corresponding observation is not used.

ZEROMODEL Statement

ZEROMODEL *effects* < / *options* > **;**

The ZEROMODEL statement enables you to perform zero-inflated Poisson regression or zero-inflated negative binomial regression when those respective distributions are specified by the DIST= option in the MODEL statement. The effects in the ZEROMODEL statement consist of explanatory variables or combinations of variables for the zero-inflation probability regression model in a zero-inflated model. The same effects can be used in both the ZEROMODEL statement and the MODEL statement, or effects can be used in one statement or the other separately. Explanatory variables can be continuous or classification variables. Classification variables can be character or numeric. Explanatory variables representing nominal, or classification, data must be declared in a CLASS statement. Interactions between variables can also be included as effects. Columns of the design matrix are automatically generated for classification variables and interactions. The syntax for specification of effects is the same as for the GLM procedure. See the section "Specification of Effects" on page 3666 for more information. Also see Chapter 53, "The GLM Procedure."

You can specify the following *option* in the ZEROMODEL statement after a slash (/).

LINK=*keyword*

specifies the link function to use in the model. The *keywords* and their associated link functions are as follows.

LINK=	Link Function
CLOGLOG	
CLL	Complementary log-log
LOGIT	Logit
PROBIT	Probit

If no LINK= option is supplied, the LOGIT link is used. User-defined link functions are not allowed.

Details: GENMOD Procedure

Generalized Linear Models Theory

This is a brief introduction to the theory of generalized linear models.

Response Probability Distributions

In generalized linear models, the response is assumed to possess a probability distribution of the exponential form. That is, the probability density of the response Y for continuous response variables, or the probability function for discrete responses, can be expressed as

$$f(y) = \exp\left\{\frac{y\theta - b(\theta)}{a(\phi)} + c(y,\phi)\right\}$$

for some functions a, b, and c that determine the specific distribution. For fixed ϕ, this is a one-parameter exponential family of distributions. The functions a and c are such that $a(\phi) = \phi/w$ and $c = c(y, \phi/w)$, where w is a known weight for each observation. A variable representing w in the input data set can be specified in the WEIGHT statement. If no WEIGHT statement is specified, $w_i = 1$ for all observations.

Standard theory for this type of distribution gives expressions for the mean and variance of Y:

$$\begin{aligned} \mathrm{E}(Y) &= b'(\theta) \\ \mathrm{Var}(Y) &= \frac{b''(\theta)\phi}{w} \end{aligned}$$

where the primes denote derivatives with respect to θ. If μ represents the mean of Y, then the variance expressed as a function of the mean is

$$\mathrm{Var}(Y) = \frac{V(\mu)\phi}{w}$$

where V is the *variance function*.

Probability distributions of the response Y in generalized linear models are usually parameterized in terms of the mean μ and dispersion parameter ϕ instead of the *natural parameter* θ. The probability distributions that are available in the GENMOD procedure are shown in the following list. The zero-inflated Poisson and zero-inflated negative binomial distributions are not generalized linear models. However, the zero-inflated distributions are included in PROC GENMOD since they are useful extensions of generalized linear models. See Long (1997) for a discussion of the zero-inflated Poisson and zero-inflated negative binomial distributions. The PROC GENMOD scale parameter and the variance of Y are also shown.

- Normal:

$$\begin{aligned} f(y) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right] \quad \text{for } -\infty < y < \infty \\ \phi &= \sigma^2 \\ \text{scale} &= \sigma \\ \text{Var}(Y) &= \sigma^2 \end{aligned}$$

- Inverse Gaussian:

$$\begin{aligned} f(y) &= \frac{1}{\sqrt{2\pi y^3}\sigma} \exp\left[-\frac{1}{2y}\left(\frac{y-\mu}{\mu\sigma}\right)^2\right] \quad \text{for } 0 < y < \infty \\ \phi &= \sigma^2 \\ \text{scale} &= \sigma \\ \text{Var}(Y) &= \sigma^2\mu^3 \end{aligned}$$

- Gamma:

$$\begin{aligned} f(y) &= \frac{1}{\Gamma(\nu)y}\left(\frac{y\nu}{\mu}\right)^{\nu} \exp\left(-\frac{y\nu}{\mu}\right) \quad \text{for } 0 < y < \infty \\ \phi &= \nu^{-1} \\ \text{scale} &= \nu \\ \text{Var}(Y) &= \frac{\mu^2}{\nu} \end{aligned}$$

- Geometric: This is a special case of the negative binomial with $k = 1$.

$$\begin{aligned} f(y) &= \frac{(\mu)^y}{(1+\mu)^{y+1}} \quad \text{for } y = 0, 1, 2, \ldots \\ \phi &= 1 \\ \text{Var}(Y) &= \mu(1+\mu) \end{aligned}$$

- Negative binomial:

$$\begin{aligned} f(y) &= \frac{\Gamma(y+1/k)}{\Gamma(y+1)\Gamma(1/k)}\frac{(k\mu)^y}{(1+k\mu)^{y+1/k}} \quad \text{for } y = 0, 1, 2, \ldots \\ \phi &= 1 \\ \text{dispersion} &= k \\ \mathrm{Var}(Y) &= \mu + k\mu^2 \end{aligned}$$

- Poisson:

$$\begin{aligned} f(y) &= \frac{\mu^y \mathrm{e}^{-\mu}}{y!} \quad \text{for } y = 0, 1, 2, \ldots \\ \phi &= 1 \\ \mathrm{Var}(Y) &= \mu \end{aligned}$$

- Binomial:

$$\begin{aligned} f(y) &= \binom{n}{r}\mu^r(1-\mu)^{n-r} \quad \text{for } y = \frac{r}{n},\ r = 0, 1, 2, \ldots, n \\ \phi &= 1 \\ \mathrm{Var}(Y) &= \frac{\mu(1-\mu)}{n} \end{aligned}$$

- Multinomial:

$$\begin{aligned} f(y_1, y_2, \ldots, y_k) &= \frac{m!}{y_1! y_2! \cdots y_k!} p_1^{y_1} p_2^{y_2} \cdots p_k^{y_k} \\ \phi &= 1 \end{aligned}$$

- Zero-inflated Poisson:

$$\begin{aligned} f(y) &= \begin{cases} \omega + (1-\omega)\mathrm{e}^{-\lambda} & \text{for } y = 0 \\ (1-\omega)\frac{\lambda^y \mathrm{e}^{-\lambda}}{y!} & \text{for } y = 1, 2, \ldots \end{cases} \\ \phi &= 1 \\ \mu = \mathrm{E}(Y) &= (1-\omega)\lambda \\ \mathrm{Var}(Y) &= (1-\omega)\lambda(1+\omega\lambda) \\ &= \mu + \frac{\omega}{1-\omega}\mu^2 \end{aligned}$$

- Zero-inflated negative binomial:

$$
\begin{aligned}
f(y) &= \begin{cases} \omega + (1-\omega)(1+k\lambda)^{-\frac{1}{k}} & \text{for } y = 0 \\ (1-\omega)\frac{\Gamma(y+1/k)}{\Gamma(y+1)\Gamma(1/k)}\frac{(k\lambda)^y}{(1+k\lambda)^{y+1/k}} & \text{for } y = 1, 2, \ldots \end{cases} \\
\phi &= 1 \\
\text{dispersion} &= k \\
\mu = \mathrm{E}(Y) &= (1-\omega)\lambda \\
\mathrm{Var}(Y) &= (1-\omega)\lambda(1+\omega\lambda+k\lambda) \\
&= \mu + \left(\frac{\omega}{1-\omega} + \frac{k}{1-\omega}\right)\mu^2
\end{aligned}
$$

- Tweedie ($1 < p < 2$):

$$
\begin{aligned}
f(y) &= \begin{cases} e^{-\lambda} & \text{for } y = 0 \\ e^{-y/\gamma}e^{-\lambda}\sum_{n=1}^{\infty}\frac{\gamma^{-n\alpha}}{\Gamma(n\alpha)}y^{n\alpha-1}\frac{\lambda^n}{n!} & \text{for } y > 0 \end{cases} \\
\phi &= \frac{\lambda^{1-p}(\alpha\gamma)^{2-p}}{2-p} \\
\mu = \mathrm{E}(Y) &= \lambda\alpha\gamma \\
\mathrm{Var}(Y) &= \lambda\alpha\gamma^2 + \lambda\alpha^2\gamma^2
\end{aligned}
$$

The negative binomial and the zero-inflated negative binomial distributions contain a parameter k, called the negative binomial dispersion parameter. This is not the same as the generalized linear model dispersion ϕ, but it is an additional distribution parameter that must be estimated or set to a fixed value.

For the binomial distribution, the response is the binomial proportion $Y = events/trials$. The variance function is $V(\mu) = \mu(1-\mu)$, and the binomial trials parameter n is regarded as a weight w.

The density function for the Tweedie distribution when $1 < p < 2$ is expressed in terms of the parameters of the compound Poisson distribution. For more information about this representation, see the section "Tweedie Distribution for Generalized Linear Models" on page 3676. For $p > 2$, the Tweedie random variable has positive support and its density function $f(y)$ can be expressed in terms of stable distributions as defined in Hougaard (1986).

If a weight variable is present, ϕ is replaced with ϕ/w, where w is the weight variable.

PROC GENMOD works with a scale parameter that is related to the exponential family dispersion parameter ϕ instead of working with ϕ itself. The scale parameters are related to the dispersion parameter as shown previously with the probability distribution definitions. Thus, the scale parameter output in the "Analysis of Parameter Estimates" table is related to the exponential family dispersion parameter. If you specify a constant scale parameter with the SCALE= option in the MODEL statement, it is also related to the exponential family dispersion parameter in the same way.

Link Function

For distributions other than the zero-inflated Poisson or zero-inflated negative binomial, the mean μ_i of the response in the ith observation is related to a linear predictor through a monotonic differentiable link function g.

$$g(\mu_i) = \mathbf{x}_i' \boldsymbol{\beta}$$

Here, $\mathbf{x}_i$ is a fixed known vector of explanatory variables, and $\boldsymbol{\beta}$ is a vector of unknown parameters.

There are two link functions and linear predictors associated with zero-inflated distributions: one for the zero inflation probability ω, and another for the mean parameter λ. See the section "Zero-Inflated Models" on page 3675 for more details about zero-inflated distributions.

Log-Likelihood Functions

Log-likelihood functions for the distributions that are available in the procedure are parameterized in terms of the means μ_i and the dispersion parameter ϕ. Zero-inflated log likelihoods are parameterized in terms two parameters, λ and ω. The parameter ω is the zero-inflation probability, and λ is a function of the distribution mean. The relationship between the mean of the zero-inflated Poisson and zero-inflated negative binomial distributions and the parameter λ is defined in the section "Response Probability Distributions" on page 3655. The term y_i represents the response for the ith observation, f_i represents a frequency weight that is specified in a FREQ statement, and w_i represents a known dispersion weight that is specified in a WEIGHT statement. If no WEIGHT statement is specified, then $w_i = 1$ for all observations. If no FREQ statement is specified, then $f_i = 1$ for all observations. The log-likelihood functions are of the form

$$L(\mathbf{y}, \boldsymbol{\mu}, \phi) = \sum_i f_i \log\left(f(y_i, \mu_i, \phi)\right)$$

where the sum is over the observations. The forms of the individual contributions

$$l_i = \log\left(f(y_i, \mu_i, \phi)\right)$$

are shown in the following list; the parameterizations are expressed in terms of the mean and dispersion parameters.

For the discrete distributions (binomial, multinomial, negative binomial, and Poisson), the functions computed as the sum of the l_i terms are not proper log-likelihood functions, since terms involving binomial coefficients or factorials of the observed counts are dropped from the computation of the log likelihood, and a dispersion parameter ϕ is included in the computation. Deletion of factorial terms and inclusion of a dispersion parameter do not affect parameter estimates or their estimated covariances for these distributions, and this is the function used in maximum likelihood estimation. The value of ϕ used in computing the reported log-likelihood function is either the final estimated value, or the fixed value, if the dispersion parameter is fixed. Even though it is not a proper log-likelihood function in all cases, the function computed as the sum of the l_i terms is reported in the output as the *log likelihood*. The proper log-likelihood function is also computed as the sum of the ll_i terms in the following list, and it is reported as the *full log likelihood* in the output.

- Normal:

$$ll_i = l_i = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{\phi} + \log\left(\frac{\phi}{w_i}\right) + \log(2\pi)\right]$$

- Inverse Gaussian:

$$ll_i = l_i = -\frac{1}{2}\left[\frac{w_i(y_i-\mu_i)^2}{y_i\mu_i^2\phi} + \log\left(\frac{\phi y_i^3}{w_i}\right) + \log(2\pi)\right]$$

- Gamma:

$$ll_i = l_i = \frac{w_i}{\phi}\log\left(\frac{w_i y_i}{\phi\mu_i}\right) - \frac{w_i y_i}{\phi\mu_i} - \log(y_i) - \log\left(\Gamma\left(\frac{w_i}{\phi}\right)\right)$$

- Negative binomial:

$$l_i = y_i\log\left(\frac{k\mu_i}{w_i}\right) - (y_i + w_i/k)\log\left(1+\frac{k\mu_i}{w_i}\right) + \log\left(\frac{\Gamma(y_i+w_i/k)}{\Gamma(w_i/k)}\right)$$

$$ll_i = y_i\log\left(\frac{k\mu_i}{w_i}\right) - (y_i + w_i/k)\log\left(1+\frac{k\mu_i}{w_i}\right) + \log\left(\frac{\Gamma(y_i+w_i/k)}{\Gamma(y_i+1)\Gamma(w_i/k)}\right)$$

- Poisson:

$$l_i = \frac{w_i}{\phi}[y_i\log(\mu_i) - \mu_i]$$

$$ll_i = w_i[y_i\log(\mu_i) - \mu_i - \log(y_i!)]$$

- Binomial:

$$l_i = \frac{w_i}{\phi}[r_i\log(p_i) + (n_i - r_i)\log(1-p_i)]$$

$$ll_i = w_i[\log\begin{pmatrix} n_i \\ r_i \end{pmatrix} + r_i\log(p_i) + (n_i - r_i)\log(1-p_i)]$$

- Multinomial (k categories):

$$l_i = \frac{w_i}{\phi}\sum_{j=1}^{k} y_{ij}\log(\mu_{ij})$$

$$ll_i = w_i[\log(m_i!) + \sum_{j=1}^{k}(y_{ij}\log(\mu_{ij}) - \log(y_{ij}!))]$$

- Zero-inflated Poisson:

$$l_i = ll_i = \begin{cases} w_i\log[\omega_i + (1-\omega_i)\exp(-\lambda_i)] & y_i = 0 \\ w_i[\log(1-\omega_i) + y_i\log(\lambda_i) - \lambda_i - \log(y_i!)] & y_i > 0 \end{cases}$$

- Zero-inflated negative binomial:

$$l_i = ll_i = \begin{cases} \log[\omega_i + (1-\omega_i)(1+\frac{k}{w_i}\lambda)^{-\frac{w_i}{k}}] & y_i = 0 \\ \\ \log(1-\omega_i) + y_i \log\left(\frac{k\lambda}{w_i}\right) & \\ -(y_i + \frac{w_i}{k}) \log\left(1+\frac{k\lambda}{w_i}\right) & \\ +\log\left(\frac{\Gamma(y_i+\frac{w_i}{k})}{\Gamma(y_i+1)\Gamma(\frac{w_i}{k})}\right) & y_i > 0 \end{cases}$$

- Tweedie:

$$l_i = ll_i = \log\left(f(y_i, \mu_i, \phi/\omega_i, p)\right)$$

Maximum Likelihood Fitting

The GENMOD procedure uses a ridge-stabilized Newton-Raphson algorithm to maximize the log-likelihood function $L(\mathbf{y}, \boldsymbol{\mu}, \phi)$ with respect to the regression parameters. By default, the procedure also produces maximum likelihood estimates of the scale parameter as defined in the section "Response Probability Distributions" on page 3655 for the normal, inverse Gaussian, negative binomial, and gamma distributions.

On the rth iteration, the algorithm updates the parameter vector $\boldsymbol{\beta}_r$ with

$$\boldsymbol{\beta}_{r+1} = \boldsymbol{\beta}_r - \mathbf{H}^{-1}\mathbf{s}$$

where $\mathbf{H}$ is the Hessian (second derivative) matrix, and $\mathbf{s}$ is the gradient (first derivative) vector of the log-likelihood function, both evaluated at the current value of the parameter vector. That is,

$$\mathbf{s} = [s_j] = \left[\frac{\partial L}{\partial \beta_j}\right]$$

and

$$\mathbf{H} = [h_{ij}] = \left[\frac{\partial^2 L}{\partial \beta_i \partial \beta_j}\right]$$

In some cases, the scale parameter is estimated by maximum likelihood. In these cases, elements corresponding to the scale parameter are computed and included in $\mathbf{s}$ and $\mathbf{H}$.

If $\eta_i = \mathbf{x}_i'\boldsymbol{\beta}$ is the linear predictor for observation i and g is the link function, then $\eta_i = g(\mu_i)$, so that $\mu_i = g^{-1}(\mathbf{x}_i'\boldsymbol{\beta})$ is an estimate of the mean of the ith observation, obtained from an estimate of the parameter vector $\boldsymbol{\beta}$.

The gradient vector and Hessian matrix for the regression parameters are given by

$$\begin{aligned} \mathbf{s} &= \sum_i \frac{f_i w_i (y_i - \mu_i)\mathbf{x}_i}{V(\mu_i) g'(\mu_i)\phi} \\ \mathbf{H} &= -\mathbf{X}'\mathbf{W}_o\mathbf{X} \end{aligned}$$

where $\mathbf{X}$ is the design matrix, $\mathbf{x}_i$ is the transpose of the ith row of $\mathbf{X}$, and V is the variance function. The matrix $\mathbf{W}_o$ is diagonal with its ith diagonal element

$$w_{oi} = w_{ei} + f_i w_i (y_i - \mu_i) \frac{V(\mu_i)g''(\mu_i) + V'(\mu_i)g'(\mu_i)}{(V(\mu_i))^2 (g'(\mu_i))^3 \phi}$$

where

$$w_{ei} = \frac{f_i w_i}{\phi V(\mu_i)(g'(\mu_i))^2}$$

The primes denote derivatives of g and V with respect to μ. The negative of $\mathbf{H}$ is called the observed information matrix. The expected value of $\mathbf{W}_o$ is a diagonal matrix $\mathbf{W}_e$ with diagonal values w_{ei}. If you replace $\mathbf{W}_o$ with $\mathbf{W}_e$, then the negative of $\mathbf{H}$ is called the expected information matrix. $\mathbf{W}_e$ is the weight matrix for the Fisher scoring method of fitting. Either $\mathbf{W}_o$ or $\mathbf{W}_e$ can be used in the update equation. The GENMOD procedure uses Fisher scoring for iterations up to the number specified by the SCORING option in the MODEL statement, and it uses the observed information matrix on additional iterations.

Covariance and Correlation Matrix

The estimated covariance matrix of the parameter estimator is given by

$$\mathbf{\Sigma} = -\mathbf{H}^{-1}$$

where $\mathbf{H}$ is the Hessian matrix evaluated using the parameter estimates on the last iteration. Note that the dispersion parameter, whether estimated or specified, is incorporated into $\mathbf{H}$. Rows and columns corresponding to aliased parameters are not included in $\mathbf{\Sigma}$.

The correlation matrix is the normalized covariance matrix. That is, if σ_{ij} is an element of $\mathbf{\Sigma}$, then the corresponding element of the correlation matrix is $\sigma_{ij}/\sigma_i\sigma_j$, where $\sigma_i = \sqrt{\sigma_{ii}}$.

Goodness of Fit

Two statistics that are helpful in assessing the goodness of fit of a given generalized linear model are the scaled deviance and Pearson's chi-square statistic. For a fixed value of the dispersion parameter ϕ, the scaled deviance is defined to be twice the difference between the maximum achievable log likelihood and the log likelihood at the maximum likelihood estimates of the regression parameters.

Note that these statistics are not valid for GEE models.

If $l(\mathbf{y}, \boldsymbol{\mu})$ is the log-likelihood function expressed as a function of the predicted mean values $\boldsymbol{\mu}$ and the vector $\mathbf{y}$ of response values, then the scaled deviance is defined by

$$D^*(\mathbf{y}, \boldsymbol{\mu}) = 2(l(\mathbf{y}, \mathbf{y}) - l(\mathbf{y}, \boldsymbol{\mu}))$$

For specific distributions, this can be expressed as

$$D^*(\mathbf{y}, \boldsymbol{\mu}) = \frac{D(\mathbf{y}, \boldsymbol{\mu})}{\phi}$$

where D is the deviance. The following table displays the deviance for each of the probability distributions available in PROC GENMOD. The deviance cannot be directly calculated for zero-inflated models. Twice the negative of the log likelihood is reported instead of the proper deviance for the zero-inflated Poisson and zero-inflated negative binomial.

Distribution	**Deviance**
Normal	$\sum_i f_i w_i (y_i - \mu_i)^2$
Poisson	$2\sum_i f_i w_i \left[y_i \log\left(\frac{y_i}{\mu_i}\right) - (y_i - \mu_i)\right]$
Binomial	$2\sum_i f_i w_i m_i \left[y_i \log\left(\frac{y_i}{\mu_i}\right) + (1 - y_i)\log\left(\frac{1-y_i}{1-\mu_i}\right)\right]$
Gamma	$2\sum_i f_i w_i \left[-\log\left(\frac{y_i}{\mu_i}\right) + \frac{y_i-\mu_i}{\mu_i}\right]$
Inverse Gaussian	$\sum_i \frac{f_i w_i (y_i-\mu_i)^2}{\mu_i^2 y_i}$
Multinomial	$\sum_i \sum_j f_i w_i y_{ij} \log\left(\frac{y_{ij}}{p_{ij} m_i}\right)$
Negative binomial	$2\sum_i f_i \left[y \log(y/\mu) - (y + w_i/k)\log\left(\frac{y+w_i/k}{\mu+w_i/k}\right)\right]$
Zero-inflated Poisson	$-2\sum_i f_i \begin{cases} w_i \log[\omega_i + (1-\omega_i)\exp(-\mu_i)] & y_i = 0 \\ w_i[\log(1-\omega_i) + y_i \log(\mu_i) - \mu_i - \log(y_i!)] & y_i > 0 \end{cases}$
Zero-inflated negative binomial	$-2\sum_i f_i \begin{cases} \log[\omega_i + (1-\omega_i)(1+\frac{k}{w_i}\lambda)] & y_i = 0 \\ \log(1-\omega_i) + y_i \log\left(\frac{k\lambda}{w_i}\right) - (y_i + \frac{w_i}{k})\log\left(1+\frac{k\lambda}{w_i}\right) + \log\left(\frac{\Gamma(y_i+\frac{w_i}{k})}{\Gamma(y_i+1)\Gamma(\frac{w_i}{k})}\right) & y_i > 0 \end{cases}$
Tweedie	$-2\sum_i f_i w_i \left[\frac{y_i^{2-p} - (2-p)y_i\mu_i^{1-p} + (1-p)\mu_i^{2-p}}{(1-p)(2-p)}\right]$

In the binomial case, $y_i = r_i/m_i$, where r_i is a binomial count and m_i is the binomial number of trials parameter.

In the multinomial case, y_{ij} refers to the observed number of occurrences of the *j*th category for the *i*th subpopulation defined by the AGGREGATE= variable, m_i is the total number in the *i*th subpopulation, and p_{ij} is the category probability.

Pearson's chi-square statistic is defined as

$$X^2 = \sum_i \frac{f_i w_i (y_i - \mu_i)^2}{V(\mu_i)}$$

and the scaled Pearson's chi-square is X^2/ϕ.

In the Tweedie case, the unit deviance is based on the quasi-likelihood function. Taking the limit of the deviance as $p \to 1$ and $p \to 2$ reduces to the deviance for the Poisson and gamma distributions, respectively. For more information about the Tweedie distribution, see the section "Tweedie Distribution for Generalized Linear Models" on page 3676.

The scaled version of both of these statistics, under certain regularity conditions, has a limiting chi-square distribution, with degrees of freedom equal to the number of observations minus the number of parameters estimated. The scaled version can be used as an approximate guide to the goodness of fit of a given model. Use caution before applying these statistics to ensure that all the conditions for the asymptotic distributions hold. McCullagh and Nelder (1989) advise that differences in deviances for nested models can be better approximated by chi-square distributions than the deviances can themselves.

In cases where the dispersion parameter is not known, an estimate can be used to obtain an approximation to the scaled deviance and Pearson's chi-square statistic. One strategy is to fit a model that contains a sufficient number of parameters so that all systematic variation is removed, estimate ϕ from this model, and then use this estimate in computing the scaled deviance of submodels. The deviance or Pearson's chi-square divided by its degrees of freedom is sometimes used as an estimate of the dispersion parameter ϕ. For example, since the limiting chi-square distribution of the scaled deviance $D^* = D/\phi$ has $n - p$ degrees of freedom, where n is the number of observations and p is the number of parameters, equating D^* to its mean and solving for ϕ yields $\hat{\phi} = D/(n - p)$. Similarly, an estimate of ϕ based on Pearson's chi-square X^2 is $\hat{\phi} = X^2/(n - p)$. Alternatively, a maximum likelihood estimate of ϕ can be computed by the procedure, if desired. See the discussion in the section "Type 1 Analysis" on page 3668 for more about the estimation of the dispersion parameter.

Other Fit Statistics

The Akaike information criterion (AIC) is a measure of goodness of model fit that balances model fit against model simplicity. AIC has the form

$$\mathrm{AIC} = -2\mathrm{LL} + 2p$$

where p is the number of parameters estimated in the model, and LL is the log likelihood evaluated at the value of the estimated parameters. An alternative form is the corrected AIC given by

$$\mathrm{AICC} = -2\mathrm{LL} + 2p\frac{n}{n - p - 1}$$

where n is the total number of observations used.

The Bayesian information criterion (BIC) is a similar measure. BIC is defined by

$$\mathrm{BIC} = -2\mathrm{LL} + p\log(n)$$

See Akaike (1981, 1979) for details of AIC and BIC. See Simonoff (2003) for a discussion of using AIC, AICC, and BIC with generalized linear models. These criteria are useful in selecting among regression models, with smaller values representing better model fit. PROC GENMOD uses the full log likelihoods defined in the section "Log-Likelihood Functions" on page 3659, with all terms included, for computing all of the criteria.

Dispersion Parameter

There are several options available in PROC GENMOD for handling the exponential distribution dispersion parameter. The NOSCALE and SCALE options in the MODEL statement affect the way in which the dispersion parameter is treated. If you specify the SCALE=DEVIANCE option, the dispersion parameter is estimated by the deviance divided by its degrees of freedom. If you specify the SCALE=PEARSON option, the dispersion parameter is estimated by Pearson's chi-square statistic divided by its degrees of freedom.

Otherwise, values of the SCALE and NOSCALE options and the resultant actions are displayed in the following table.

NOSCALE	**SCALE=*value***	**Action**
Present	Present	Scale fixed at *value*
Present	Not present	Scale fixed at 1
Not present	Not present	Scale estimated by ML
Not present	Present	Scale estimated by ML, starting point at *value*
Present (negative binomial)	Not present	k fixed at 0

The meaning of the scale parameter displayed in the "Analysis Of Parameter Estimates" table is different for the gamma distribution than for the other distributions. The relation of the scale parameter as used by PROC GENMOD to the exponential family dispersion parameter ϕ is displayed in the following table. For the binomial and Poisson distributions, ϕ is the overdispersion parameter, as defined in the "Overdispersion" section, which follows.

Distribution	**Scale**
Normal	$\sqrt{\phi}$
Inverse Gaussian	$\sqrt{\phi}$
Gamma	$1/\phi$
Binomial	$\sqrt{\phi}$
Poisson	$\sqrt{\phi}$

In the case of the negative binomial distribution, PROC GENMOD reports the "dispersion" parameter estimated by maximum likelihood. This is the negative binomial parameter k defined in the section "Response Probability Distributions" on page 3655.

Overdispersion

Overdispersion is a phenomenon that sometimes occurs in data that are modeled with the binomial or Poisson distributions. If the estimate of dispersion after fitting, as measured by the deviance or Pearson's chi-square, divided by the degrees of freedom, is not near 1, then the data might be *overdispersed* if the dispersion estimate is greater than 1 or *underdispersed* if the dispersion estimate is less than 1. A simple way to model this situation is to allow the variance functions of these distributions to have a multiplicative overdispersion factor ϕ:

- Binomial: $V(\mu) = \phi\mu(1-\mu)$
- Poisson: $V(\mu) = \phi\mu$

An alternative method to allow for overdispersion in the Poisson distribution is to fit a negative binomial distribution, where $V(\mu) = \mu + k\mu^2$, instead of the Poisson. The parameter k can be estimated by maximum likelihood, thus allowing for overdispersion of a specific form. This is different from the multiplicative overdispersion factor ϕ, which can accommodate many forms of overdispersion.

The models are fit in the usual way, and the parameter estimates are not affected by the value of ϕ. The covariance matrix, however, is multiplied by ϕ, and the scaled deviance and log likelihoods used in likelihood ratio tests are divided by ϕ. The profile likelihood function used in computing confidence intervals is also divided by ϕ. If you specify a WEIGHT statement, ϕ is divided by the value of the WEIGHT variable for each observation. This has the effect of multiplying the contributions of the log-likelihood function, the gradient, and the Hessian by the value of the WEIGHT variable for each observation.

The SCALE= option in the MODEL statement enables you to specify a value of $\sigma = \sqrt{\phi}$ for the binomial and Poisson distributions. If you specify the SCALE=DEVIANCE option in the MODEL statement, the procedure uses the deviance divided by degrees of freedom as an estimate of ϕ, and all statistics are adjusted appropriately. You can use Pearson's chi-square instead of the deviance by specifying the SCALE=PEARSON option.

The function obtained by dividing a log-likelihood function for the binomial or Poisson distribution by a dispersion parameter is not a legitimate log-likelihood function. It is an example of a *quasi-likelihood* function. Most of the asymptotic theory for log likelihoods also applies to quasi-likelihoods, which justifies computing standard errors and likelihood ratio statistics by using quasi-likelihoods instead of proper log likelihoods. For details on quasi-likelihood functions, see McCullagh and Nelder (1989, Chapter 9), McCullagh (1983); Hardin and Hilbe (2003).

Although the estimate of the dispersion parameter is often used to indicate overdispersion or underdispersion, this estimate might also indicate other problems such as an incorrectly specified model or outliers in the data. You should carefully assess whether this type of model is appropriate for your data.

Specification of Effects

Each term in a model is called an effect. Effects are specified in the MODEL statement. You specify effects with a special notation that uses variable names and operators. There are two types of variables, *classification* (or CLASS) variables and *continuous* variables. There are two primary types of operators, *crossing* and *nesting*. A third type, the *bar* operator, is used to simplify effect specification. Crossing is the type of operator most commonly used in generalized linear models.

Variables that identify classification levels are called CLASS variables in SAS and are specified in a CLASS statement. These might also be called *categorical, qualitative, discrete,* or *nominal* variables. CLASS variables can be either character or numeric. The values of CLASS variables are called *levels*. For example, the CLASS variable Sex could have the levels 'male' and 'female'.

In a model, an explanatory variable that is not declared in a CLASS statement is assumed to be continuous. Continuous variables must be numeric. For example, the heights and weights of subjects in an experiment are continuous variables.

The types of effects most useful in generalized linear models are shown in the following list. Assume that A, B, and C are classification variables and that X1 and X2 are continuous variables.

- Regressor effects are specified by writing continuous variables by themselves: X1, X2.

- Polynomial effects are specified by using asterisks to join two or more continuous variables: X1*X2.
- Main effects are specified by writing classification variables by themselves: A, B, C.
- Crossed effects (interactions) are specified by joining two or more classification variables with asterisks: A*B, B*C, A*B*C.
- Nested effects are specified by following a main effect or crossed effect with a classification variable or list of classification variables enclosed in parentheses: B(A), C(B A), A*B(C). In this example, B(A) is "B nested within A."
- Combinations of continuous and classification variables can be specified in the same way by using the crossing and nesting operators.

The bar operator consists of two effects joined with a vertical bar (|). It is shorthand notation for including the left-hand side, the right-hand side, and the cross between them as effects in the model. For example, the expression A | B is equivalent to A B A*B. The effects in the bar operator can be classification variables, continuous variables, or combinations of effects defined using operators. Multiple bars are permitted. For example, A | B | C means A B C A*B A*C B*C A*B*C.

You can specify the maximum number of variables in any effect that results from bar evaluation by specifying the maximum number, preceded by an @ sign. For example, A | B | C@2 results in effects that involve no more than two variables: A B C A*B A*C B*C.

Parameterization Used in PROC GENMOD

Design Matrix

The linear predictor part of a generalized linear model is

$$\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$$

where $\boldsymbol{\beta}$ is an unknown parameter vector and $\mathbf{X}$ is a known design matrix. By default, all models automatically contain an intercept term; that is, the first column of $\mathbf{X}$ contains all 1s. Additional columns of $\mathbf{X}$ are generated for classification variables, regression variables, and any interaction terms included in the model. It is important to understand the ordering of classification variable parameters when you use the ESTIMATE or CONTRAST statement. The ordering of these parameters is displayed in the "CLASS Level Information" table and in tables that display the parameter estimates of the fitted model.

When you specify an overparameterized model by using the PARAM=GLM option in the CLASS statement, some columns of $\mathbf{X}$ can be linearly dependent on other columns. For example, when you specify a model that consists of an intercept term and a classification variable, the column that corresponds to any one of the levels of the classification variable is linearly dependent on the other columns of $\mathbf{X}$. The columns of $\mathbf{X}'\mathbf{X}$ are checked in the order in which the model is specified for dependence on preceding columns. If a dependency is found, the parameter that corresponds to the dependent column is set to 0 along with its standard error to indicate that it is not estimated. The order in which the levels of a classification variable are checked for dependencies can be set by the ORDER= option in the PROC GENMOD statement or by the ORDER= option in the CLASS statement. For full-rank parameterizations, the columns of the $\mathbf{X}$ matrix are designed to be linearly independent.

You can exclude the intercept term from the model by specifying the NOINT option in the MODEL statement.

Missing Level Combinations

PROC GENMOD handles missing level combinations of classification variables in the same manner as the GLM and MIXED procedures. Parameters that correspond to missing level combinations are not included in the model.

Type 1 Analysis

A Type 1 analysis consists of fitting a sequence of models, beginning with a simple model that has only an intercept term, and continuing through a model of specified complexity, fitting one additional effect at each step. Likelihood ratio statistics—that is, twice the difference of the log likelihoods—are computed between successive models. This type of analysis is sometimes called an analysis of deviance because, if the dispersion parameter is held fixed for all models, it is equivalent to computing differences of scaled deviances. The asymptotic distribution of the likelihood ratio statistics, under the hypothesis that the additional parameters included in the model are equal to 0, is a chi-square with degrees of freedom equal to the difference in the number of parameters estimated in the successive models. Thus, these statistics can be used to test the hypothesis of the significance of each additional term that is fitted. For zero-inflated models, two analyses are performed. The first analysis adds terms to the mean model, assuming a full zero-inflated model; and the second analysis adds terms to the zero-inflated model, assuming a full-mean model. The results from these analyses appear in separate tables.

This type of analysis is not available for GEE models, because the deviance is not computed for this type of model.

If the dispersion parameter ϕ is known, it can be included in the models; if it is unknown, there are two strategies allowed by PROC GENMOD. The dispersion parameter can be estimated from a maximal model by the deviance or Pearson's chi-square divided by degrees of freedom, as discussed in the section "Goodness of Fit" on page 3662, and this value can be used in all models. An alternative is to consider the dispersion to be an additional unknown parameter for each model and estimate it by maximum likelihood on each step. By default, PROC GENMOD estimates scale by maximum likelihood at each step.

A table of likelihood ratio statistics is produced, along with the associated p-values based on the asymptotic chi-square distributions.

If you specify either the SCALE=DEVIANCE or SCALE=PEARSON option in the MODEL statement, the dispersion parameter is estimated using the deviance or Pearson's chi-square statistic, and F statistics are computed in addition to the chi-square statistics for assessing the significance of each additional term in the Type 1 analysis. See the section "F Statistics" on page 3671 for a definition of F statistics.

This Type 1 analysis has the general property that the results depend on the order in which the terms of the model are fitted. The terms are fitted in the order in which they are specified in the MODEL statement.

Type 3 Analysis

A Type 3 analysis is similar to the Type III sums of squares used in PROC GLM, except that likelihood ratios are used instead of sums of squares. First, a Type III estimable function is defined for an effect of interest in exactly the same way as in PROC GLM. Then maximum likelihood estimates are computed under the constraint that the Type III function of the parameters is equal to 0, by using constrained optimization. Let the resulting constrained parameter estimates be $\tilde{\boldsymbol{\beta}}$ and the log likelihood be $l(\tilde{\boldsymbol{\beta}})$. Then the likelihood ratio statistic

$$S = 2(l(\hat{\boldsymbol{\beta}}) - l(\tilde{\boldsymbol{\beta}}))$$

where $\hat{\boldsymbol{\beta}}$ is the unconstrained estimate, has an asymptotic chi-square distribution under the hypothesis that the Type III contrast is equal to 0, with degrees of freedom equal to the number of parameters associated with the effect.

When a Type 3 analysis is requested, PROC GENMOD produces a table that contains the likelihood ratio statistics, degrees of freedom, and p-values based on the limiting chi-square distributions for each effect in the model. If you specify either the DSCALE or PSCALE option in the MODEL statement, F statistics are also computed for each effect.

Options for handling the dispersion parameter are the same as for a Type 1 analysis. The dispersion parameter can be specified to be a known value, estimated from the deviance or Pearson's chi-square divided by degrees of freedom, or estimated by maximum likelihood individually for the unconstrained and constrained models. By default, PROC GENMOD estimates scale by maximum likelihood for each model fit.

The results of this type of analysis do not depend on the order in which the terms are specified in the MODEL statement.

A Type 3 analysis can consume considerable computation time since a constrained model is fitted for each effect. Wald statistics for Type 3 contrasts are computed if you specify the WALD option. Wald statistics for contrasts use less computation time than likelihood ratio statistics but might be less accurate indicators of the significance of the effect of interest. The Wald statistic for testing $\mathbf{L}'\boldsymbol{\beta} = \mathbf{0}$, where $\mathbf{L}$ is the contrast matrix, is defined by

$$S = (\mathbf{L}'\hat{\boldsymbol{\beta}})'(\mathbf{L}'\hat{\boldsymbol{\Sigma}}\mathbf{L})^{-}(\mathbf{L}'\hat{\boldsymbol{\beta}})$$

where $\boldsymbol{\beta}$ is the maximum likelihood estimate and $\boldsymbol{\Sigma}$ is its estimated covariance matrix. The asymptotic distribution of S is chi-square with r degrees of freedom, where r is the rank of $\mathbf{L}$.

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, performing the Type 3 test of a main effect corresponds to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions." Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are zero. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), performing the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses

reference parameterization (as specified by the PARAM=REF option in the CLASS statement), performing the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects. For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).

If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimable function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.

Generalized score tests for Type III contrasts are computed for GEE models if you specify the TYPE3 option in the MODEL statement when a REPEATED statement is also used. See the section "Generalized Score Statistics" on page 3686 for more information about generalized score statistics. Wald tests are also available with the Wald option in the CONTRAST statement. In this case, the robust covariance matrix estimate is used for $\boldsymbol{\Sigma}$ in the Wald statistic.

Confidence Intervals for Parameters

Likelihood Ratio-Based Confidence Intervals

PROC GENMOD produces likelihood ratio-based confidence intervals, also known as profile likelihood confidence intervals, for parameter estimates for generalized linear models. These are not computed for GEE models, since there is no likelihood for this type of model. Suppose that the parameter vector is $\boldsymbol{\beta} = [\beta_0, \beta_1, \ldots, \beta_p]'$ and that you want a confidence interval for β_j. The profile likelihood function for β_j is defined as

$$l^*(\beta_j) = \max_{\tilde{\boldsymbol{\beta}}} l(\boldsymbol{\beta})$$

where $\tilde{\boldsymbol{\beta}}$ is the vector $\boldsymbol{\beta}$ with the jth element fixed at β_j and l is the log-likelihood function. If $l = l(\hat{\boldsymbol{\beta}})$ is the log likelihood evaluated at the maximum likelihood estimate $\hat{\boldsymbol{\beta}}$, then $2(l - l^*(\beta_j))$ has a limiting chi-square distribution with one degree of freedom if β_j is the true parameter value. A $(1-\alpha)100\%$ confidence interval for β_j is

$$\{\beta_j : l^*(\beta_j) \geq l_0 = l - 0.5\chi^2_{1-\alpha,1}\}$$

where $\chi^2_{1-\alpha,1}$ is the $100(1-\alpha)$th percentile of the chi-square distribution with one degree of freedom. The endpoints of the confidence interval can be found by solving numerically for values of β_j that satisfy equality in the preceding relation. PROC GENMOD solves this by starting at the maximum likelihood estimate of $\boldsymbol{\beta}$. The log-likelihood function is approximated with a quadratic surface, for which an exact solution is possible. The process is iterated until convergence to an endpoint is attained. The process is repeated for the other endpoint.

Convergence is controlled by the CICONV= option in the MODEL statement. Suppose ϵ is the number specified in the CICONV= option. The default value of ϵ is 10^{-4}. Let the parameter of interest be β_j, and define $\mathbf{r} = \mathbf{u}_j$, the unit vector with a 1 in position j and 0s elsewhere. Convergence is declared on the current iteration if the following two conditions are satisfied:

$$|l^*(\beta_j) - l_0| \leq \epsilon$$
$$(\mathbf{s} + \lambda\mathbf{r})'\mathbf{H}^{-1}(\mathbf{s} + \lambda\mathbf{r}) \leq \epsilon$$

where $l^*(\beta_j)$, $\mathbf{s}$, and $\mathbf{H}$ are the log likelihood, the gradient, and the Hessian evaluated at the current parameter vector and λ is a constant computed by the procedure. The first condition for convergence means that the log-likelihood function must be within ϵ of the correct value, and the second condition means that the gradient vector must be proportional to the restriction vector $\mathbf{r}$.

When you specify the LRCI option in the MODEL statement, PROC GENMOD computes profile likelihood confidence intervals for all parameters in the model, including the scale parameter, if there is one. The interval endpoints are displayed in a table as well as the values of the remaining parameters at the solution.

Wald Confidence Intervals

You can request that PROC GENMOD produce Wald confidence intervals for the parameters. The $(1-\alpha)100\%$ Wald confidence interval for a parameter β is defined as

$$\hat{\beta} \pm z_{1-\alpha/2}\hat{\sigma}$$

where z_p is the $100p$th percentile of the standard normal distribution, $\hat{\beta}$ is the parameter estimate, and $\hat{\sigma}$ is the estimate of its standard error.

F Statistics

Suppose that D_0 is the deviance resulting from fitting a generalized linear model and that D_1 is the deviance from fitting a submodel. Then, under appropriate regularity conditions, the asymptotic distribution of $(D_1 - D_0)/\phi$ is chi-square with r degrees of freedom, where r is the difference in the number of parameters between the two models and ϕ is the dispersion parameter. If ϕ is unknown, and $\hat{\phi}$ is an estimate of ϕ based on the deviance or Pearson's chi-square divided by degrees of freedom, then, under regularity conditions, $(n-p)\hat{\phi}/\phi$ has an asymptotic chi-square distribution with $n-p$ degrees of freedom. Here, n is the number of observations and p is the number of parameters in the model that is used to estimate ϕ. Thus, the asymptotic distribution of

$$F = \frac{D_1 - D_0}{r\hat{\phi}}$$

is the F distribution with r and $n - p$ degrees of freedom, assuming that $(D_1 - D_0)/\phi$ and $(n - p)\hat{\phi}/\phi$ are approximately independent.

This F statistic is computed for the Type 1 analysis, Type 3 analysis, and hypothesis tests specified in CONTRAST statements when the dispersion parameter is estimated by either the deviance or Pearson's chi-square divided by degrees of freedom, as specified by the DSCALE or PSCALE option in the MODEL statement. In the case of a Type 1 analysis, model 0 is the higher-order model obtained by including one additional effect in model 1. For a Type 3 analysis and hypothesis tests, model 0 is the full specified model and model 1 is the submodel obtained from constraining the Type III contrast or the user-specified contrast to be 0.

Lagrange Multiplier Statistics

When you select the NOINT or NOSCALE option, restrictions are placed on the intercept or scale parameters. Lagrange multiplier, or score, statistics are computed in these cases. These statistics assess the validity of the restrictions, and they are computed as

$$\chi^2 = \frac{s^2}{V}$$

where s is the component of the score vector evaluated at the restricted maximum corresponding to the restricted parameter and $V = \mathbf{I}_{11} - \mathbf{I}_{12}\mathbf{I}_{22}^{-1}\mathbf{I}_{21}$. The matrix $\mathbf{I}$ is the information matrix, 1 refers to the restricted parameter, and 2 refers to the rest of the parameters.

Under regularity conditions, this statistic has an asymptotic chi-square distribution with one degree of freedom, and p-values are computed based on this limiting distribution.

If you set $k = 0$ in a negative binomial model, s is the score statistic of Cameron and Trivedi (1998) for testing for overdispersion in a Poisson model against alternatives of the form $V(\mu) = \mu + k\mu^2$.

See Rao (1973, p. 417) for more details.

Predicted Values of the Mean

Predicted Values

A predicted value, or fitted value, of the mean μ_i corresponding to the vector of covariates $\mathbf{x}_i$ is given by

$$\hat{\mu}_i = g^{-1}(\mathbf{x}_i'\hat{\boldsymbol{\beta}})$$

where g is the link function, regardless of whether $\mathbf{x}_i$ corresponds to an observation or not. That is, the response variable can be missing and the predicted value is still computed for valid $\mathbf{x}_i$. In the case where $\mathbf{x}_i$ does not correspond to a valid observation, $\mathbf{x}_i$ is not checked for estimability. You should check the estimability of $\mathbf{x}_i$ in this case in order to ensure the uniqueness of the predicted value of the mean. If there is an offset, it is included in the predicted value computation.

Confidence Intervals on Predicted Values

Approximate confidence intervals for predicted values of the mean can be computed as follows. The variance of the linear predictor $\eta_i = \mathbf{x}_i'\hat{\boldsymbol{\beta}}$ is estimated by

$$\sigma_x^2 = \mathbf{x}_i'\boldsymbol{\Sigma}\mathbf{x}_i$$

where $\boldsymbol{\Sigma}$ is the estimated covariance of $\hat{\boldsymbol{\beta}}$. The robust estimate of the covariance is used for $\boldsymbol{\Sigma}$ in the case of models fit with GEEs.

Approximate $100(1-\alpha)\%$ confidence intervals are computed as

$$g^{-1}\left(\mathbf{x}_i'\hat{\beta} \pm z_{1-\alpha/2}\sigma_x\right)$$

where z_p is the $100p$th percentile of the standard normal distribution and g is the link function. If either endpoint in the argument is outside the valid range of arguments for the inverse link function, the corresponding confidence interval endpoint is set to missing.

Residuals

The GENMOD procedure computes three kinds of residuals. Residuals are available for all generalized linear models except multinomial models for ordinal response data, for which residuals are not available. Raw residuals and Pearson residuals are available for models fit with generalized estimating equations (GEEs).

The raw residual is defined as

$$r_i = y_i - \mu_i$$

where y_i is the ith response and μ_i is the corresponding predicted mean. You can request raw residuals in an output data set with the keyword RESRAW in the OUTPUT statement.

The Pearson residual is the square root of the ith contribution to the Pearson's chi-square:

$$r_{Pi} = (y_i - \mu_i)\sqrt{\frac{w_i}{V(\mu_i)}}$$

You can request Pearson residuals in an output data set with the keyword RESCHI in the OUTPUT statement.

Finally, the deviance residual is defined as the square root of the contribution of the ith observation to the deviance, with the sign of the raw residual:

$$r_{Di} = \sqrt{d_i}(\mathrm{sign}(y_i - \mu_i))$$

You can request deviance residuals in an output data set with the keyword RESDEV in the OUTPUT statement. For more information about the deviance computations, see the section "Goodness of Fit" on page 3662.

The adjusted Pearson, deviance, and likelihood residuals are defined by Agresti (2002); Williams (1987); Davison and Snell (1991). These residuals are useful for outlier detection and for assessing the influence of single observations on the fitted model.

For the generalized linear model, the variance of the ith individual observation is given by

$$v_i = \frac{\phi V(\mu_i)}{w_i}$$

where ϕ is the dispersion parameter, w_i is a user-specified prior weight (if not specified, $w_i = 1$), μ_i is the mean, and $V(\mu_i)$ is the variance function. Let

$$w_{ei} = v_i^{-1}(g'(\mu_i))^{-2}$$

for the ith observation, where $g'(\mu_i)$ is the derivative of the link function, evaluated at μ_i. Let $\mathbf{W}_e$ be the diagonal matrix with w_{ei} denoting the ith diagonal element. The weight matrix $\mathbf{W}_e$ is used in computing the expected information matrix.

Define h_i as the ith diagonal element of the matrix

$$\mathbf{W}_e^{\frac{1}{2}}\mathbf{X}(\mathbf{X}'\mathbf{W}_e\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}_e^{\frac{1}{2}}$$

The Pearson residuals, standardized to have unit asymptotic variance, are given by

$$r_{Pi} = \frac{y_i - \mu_i}{\sqrt{v_i(1 - h_i)}}$$

You can request standardized Pearson residuals in an output data set with the keyword STDRESCHI in the OUTPUT statement. The deviance residuals, standardized to have unit asymptotic variance, are given by

$$r_{Di} = \frac{\mathrm{sign}(y_i - \mu_i)\sqrt{d_i}}{\sqrt{\phi(1-h_i)}}$$

where d_i is the contribution to the total deviance from observation i, and $\mathrm{sign}(y_i - \mu_i)$ is 1 if $y_i - \mu_i$ is positive and –1 if $y_i - \mu_i$ is negative. You can request standardized deviance residuals in an output data set with the keyword STDRESDEV in the OUTPUT statement. The likelihood residuals are defined by

$$r_{Gi} = \mathrm{sign}(y_i - \mu_i)\sqrt{(1-h_i)r_{Di}^2 + h_i r_{Pi}^2}$$

You can request likelihood residuals in an output data set with the keyword RESLIK in the OUTPUT statement.

Multinomial Models

This type of model applies to cases where an observation can fall into one of k categories. Binary data occur in the special case where $k = 2$. If there are m_i observations in a subpopulation i, then the probability distribution of the number falling into the k categories $\mathbf{y}_i = (y_{i1}, y_{i2}, \ldots, y_{ik})$ can be modeled by the multinomial distribution, defined in the section "Response Probability Distributions" on page 3655, with $\sum_j y_{ij} = m_i$. The multinomial model is an *ordinal* model if the categories have a natural order.

Residuals are not available in the OBSTATS table or the output data set for multinomial models.

By default, and consistently with binomial models, the GENMOD procedure orders the response categories for ordinal multinomial models from lowest to highest and models the probabilities of the lower response levels. You can change the way PROC GENMOD orders the response levels with the RORDER= option in the PROC GENMOD statement. The order that PROC GENMOD uses is shown in the "Response Profiles" output table described in the section "Response Profile" on page 3704.

The GENMOD procedure supports only the ordinal multinomial model. If $(p_{i1}, p_{i2}, \ldots, p_{ik})$ are the category probabilities, the cumulative category probabilities are modeled with the same link functions used for binomial data. Let $P_{ir} = \sum_{j=1}^{r} p_{ij}$, $r = 1, 2, \ldots, k-1$, be the cumulative category probabilities (note that $P_{ik} = 1$). The ordinal model is

$$g(P_{ir}) = \mu_r + \mathbf{x}'\boldsymbol{\beta} \quad \text{for} \quad r = 1, 2, \ldots, k-1$$

where $\mu_1, \mu_2, \ldots, \mu_{k-1}$ are intercept terms that depend only on the categories and $\mathbf{x}_i$ is a vector of covariates that does not include an intercept term. The logit, probit, and complementary log-log link functions g are available. These are obtained by specifying the MODEL statement options DIST=MULTINOMIAL and LINK=CUMLOGIT (cumulative logit), LINK=CUMPROBIT (cumulative probit), or LINK=CUMCLL (cumulative complementary log-log). Alternatively,

$$P_{ir} = \mathrm{F}(\mu_r + \mathbf{x}'\boldsymbol{\beta}) \quad \text{for} \quad r = 1, 2, \ldots, k-1$$

where $\mathrm{F} = g^{-1}$ is a cumulative distribution function for the logistic, normal, or extreme-value distribution.

PROC GENMOD estimates the intercept parameters $\mu_1, \mu_2, \ldots, \mu_{k-1}$ and regression parameters $\boldsymbol{\beta}$ by maximum likelihood.

The subpopulations i are defined by constant values of the AGGREGATE= variable. This has no effect on the parameter estimates, but it does affect the deviance and Pearson chi-square statistics; it also affects parameter estimate standard errors if you specify the SCALE=DEVIANCE or SCALE=PEARSON option.

Zero-Inflated Models

Count data that have an incidence of zeros greater than expected for the underlying probability distribution of counts can be modeled with a zero-inflated distribution. In GENMOD, the underlying distribution can be either Poisson or negative binomial. See Lambert (1992), Long (1997) and Cameron and Trivedi (1998) for more information about zero-inflated models. The population is considered to consist of two types of individuals. The first type gives Poisson or negative binomial distributed counts, which might contain zeros. The second type always gives a zero count. Let λ be the underlying distribution mean and ω be the probability of an individual being of the second type. The parameter ω is called here the *zero-inflation probability*, and is the probability of zero counts in excess of the frequency predicted by the underlying distribution. You can request that the zero inflation probability be displayed in an output data set with the PZERO keyword. The probability distribution of a zero-inflated Poisson random variable Y is given by

$$\Pr(Y = y) = \begin{cases} \omega + (1-\omega)\mathrm{e}^{-\lambda} & \text{for } y = 0 \\ (1-\omega)\frac{\lambda^y \mathrm{e}^{-\lambda}}{y!} & \text{for } y = 1, 2, \ldots \end{cases}$$

and the probability distribution of a zero-inflated negative binomial random variable Y is given by

$$\Pr(Y = y) = \begin{cases} \omega + (1-\omega)(1+k\lambda)^{-\frac{1}{k}} & \text{for } y = 0 \\ (1-\omega)\frac{\Gamma(y+1/k)}{\Gamma(y+1)\Gamma(1/k)}\frac{(k\lambda)^y}{(1+k\lambda)^{y+1/k}} & \text{for } y = 1, 2, \ldots \end{cases}$$

where k is the negative binomial dispersion parameter.

You can model the parameters ω and λ in GENMOD with the regression models:

$$\begin{aligned} h(\omega_i) &= \mathbf{z}_i'\boldsymbol{\gamma} \\ g(\lambda_i) &= \mathbf{x}_i'\boldsymbol{\beta} \end{aligned}$$

where h is one of the binary link functions: logit, probit, or complementary log-log. The link function h is the logit link by default, or the link function option specified in the ZEROMODEL statement. The link function g is the log link function by default, or the link function specified in the MODEL statement, for both the Poisson and the negative binomial. The covariates $\mathbf{z}_i$ for observation i are determined by the model specified in the ZEROMODEL statement, and the covariates $\mathbf{x}_i$ are determined by the model specified in the MODEL statement. The regression parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are estimated by maximum likelihood.

The mean and variance of Y for the zero-inflated Poisson are given by

$$\begin{aligned} \mathrm{E}(Y) &= \mu = (1-\omega)\lambda \\ \mathrm{Var}(Y) &= \mu + \frac{\omega}{1-\omega}\mu^2 \end{aligned}$$

and for the zero-inflated negative binomial by

$$\begin{aligned} \mathrm{E}(Y) &= \mu = (1-\omega)\lambda \\ \mathrm{Var}(Y) &= \mu + \left(\frac{\omega}{1-\omega} + \frac{k}{1-\omega}\right)\mu^2 \end{aligned}$$

You can request that the mean of Y be displayed for each observation in an output data set with the PRED keyword.

Tweedie Distribution for Generalized Linear Models

The Tweedie (1984) distribution has nonnegative support and can have a discrete mass at zero, making it useful to model responses that are a mixture of zeros and positive values. The Tweedie distribution belongs to the exponential family, so it conveniently fits in the generalized linear models framework. According to such parameterization, the mean and variance for the Tweedie random variable are $\mathrm{E}(Y) = \mu$ and $\mathrm{Var}(Y) = \phi\mu^p$, respectively, where ϕ is the dispersion parameter and p is an extra parameter that controls the variance of the distribution.

The Tweedie family of distributions includes several important distributions for generalized linear models. When $p = 0$, the Tweedie distribution degenerates to the normal distribution; when $p = 1$, it becomes a Poisson distribution; when $p = 2$, it becomes a gamma distribution; when $p = 3$, it is an inverse Gaussian distribution.

Except for these special cases, the probability density function for the Tweedie distribution does not have a closed form and can at best be expressed in terms of series. Numerical approximations are needed to evaluate the density function. Dunn and Smyth (2005) propose using a finite series and provide a formula to determine its lower and upper indices in order to achieve a desired accuracy. Alternatively, you can apply the Fourier transformation on the characteristic function (Dunn and Smyth 2008). These approximations tend to be expensive when a high level of accuracy is demanded or the data volume becomes large. PROC GENMOD uses the series method unless it becomes complicated to do so. In this case, the method that is based on the Fourier transformation is used. The accuracy of approximation is controlled by the EPSILON= option, whose default value is 10^{-5}.

The Tweedie distribution is not defined when p is between 0 and 1. In practice, the most interesting range is from 1 to 2 in which the Tweedie distribution gradually loses its mass at 0 as it shifts from a Poisson distribution to a gamma distribution. In this case, the Tweedie random variable Y can be generated from a compound Poisson distribution (Smyth 1996) as

$$\begin{aligned} Y &= \Sigma_{i=1}^{T} X_i \\ T &\sim \mathrm{Poisson}(\lambda) \\ X_i &\sim \mathrm{gamma}(\alpha, \gamma) \end{aligned}$$

where $Y = 0$ if $T = 0$, T and X_i are statistically independent, and $\mathrm{gamma}(\alpha, \gamma)$ denotes a gamma random variable that has mean $\alpha\gamma$ and variance $\alpha\gamma^2$. These parameters are determined by the Tweedie parameters as follows:

$$\begin{aligned} \lambda &= \frac{\mu^{2-p}}{\phi(2-p)} \\ \alpha &= \frac{2-p}{p-1} \\ \gamma &= \phi(p-1)\mu^{p-1} \end{aligned}$$

Inversely, given the Tweedie distributional parameters, the parameters of the compound Poisson distribution are determined as follows:

$$
\begin{aligned}
\mu &= \lambda\alpha\gamma \\
p &= \frac{\alpha+2}{\alpha+1} \\
\phi &= \frac{\lambda^{1-p}(\alpha\gamma)^{2-p}}{2-p}
\end{aligned}
$$

In terms of generalized linear models parameterizations, the canonical parameter θ for the Tweedie density can be expressed as

$$
\theta = \begin{cases} \frac{\mu^{1-p}}{1-p} & p \neq 1 \\ \log\mu & p = 1 \end{cases}
$$

and the function $b(\theta)$ is

$$
b(\theta) = \begin{cases} \frac{\mu^{2-p}}{2-p} & p \neq 2 \\ \log\mu & p = 2 \end{cases}
$$

Because of the intractability of differentiating the gradient functions with respect to the variance parameters, PROC GENMOD uses a quasi-Newton approach to maximize the likelihood function, where the Hessian matrix is approximated by taking finite differences of the gradient functions. Convergence is determined by a union of two criteria: the relative gradient convergence criterion is set to 10^{-9}, and the relative function convergence criterion is set to 2×10^{-9}. Convergence is declared when at least one of the criteria is attained during the quasi-Newton iteration.

Before PROC GENMOD maximizes the approximate likelihood, it first maximizes the following extended log quasi-likelihood which is constructed according to the definition of McCullagh and Nelder (1989, Chapter 9) as

$$
Q_p(\mathbf{y}, \boldsymbol{\mu}, \phi, p) = \sum_i q(y_i, \mu_i, \phi, p)
$$

where the contribution from an observation is

$$
q(y_i, \mu_i, \phi, p) = -0.5\log(2\pi\phi y_i^p/w_i) - w_i\left(\frac{y_i^{2-p} - (2-p)y_i\mu_i^{1-p} + (1-p)\mu_i^{2-p}}{(1-p)(2-p)}\right)/\phi
$$

and w_i is the weight for the observation from the WEIGHT statement.

The range of parameter p for the quasi-likelihood is from 1 to 2. For a specified P= value outside this range, PROC GENMOD skips optimization of the quasi-likelihood. To maintain numerical stability, PROC GENMOD imposes a lower bound of 1.1 and a upper bound of 1.99 for computation with the quasi-likelihood. The full-likelihood solution imposes the same lower bound but no upper bound. The estimates that are obtained from optimizing the quasi-likelihood are usually near the full-likelihood solution so that fewer iterations are needed for maximizing the more expensive full likelihood.

Generalized Estimating Equations

Let y_{ij}, $j = 1, \ldots, n_i$, $i = 1, \ldots, K$, represent the *j*th measurement on the *i*th subject. There are n_i measurements on subject *i* and $\sum_{i=1}^{K} f_i n_i$ total measurements where f_i is the cluster frequency that you specify in the FREQUENCY statement. If you do not specify a FREQUENCY statement, $f_i = 1$ for all observations. The frequencies must be the same for all observations within each cluster.

Correlated data are modeled using the same link function and linear predictor setup (systematic component) as the independence case. The random component is described by the same variance functions as in the independence case, but the covariance structure of the correlated measurements must also be modeled. In the rest of this section, $v(\mu)$ is the variance function of the specified distribution. For the case of the negative binomial, the variance function is fixed at $v(\mu) = \mu + k\mu^2$, where k is either the maximum likelihood estimate of the negative binomial dispersion parameter or the value specified in the NOSCALE and SCALE= options in the MODEL statement.

Let the vector of measurements on the *i*th subject be $\mathbf{Y}_i = [y_{i1}, \ldots, y_{in_i}]'$ with corresponding vector of means $\boldsymbol{\mu}_i = [\mu_{i1}, \ldots, \mu_{in_i}]'$, and let $\mathbf{V}_i$ be the covariance matrix of $\mathbf{Y}_i$. Let the vector of independent, or explanatory, variables for the *j*th measurement on the *i*th subject be

$$\mathbf{x}_{ij} = [x_{ij1}, \ldots, x_{ijp}]'$$

The generalized estimating equation of Liang and Zeger (1986) for estimating the $p \times 1$ vector of regression parameters $\boldsymbol{\beta}$ is an extension of the independence estimating equation to correlated data and is given by

$$\mathbf{S}(\boldsymbol{\beta}) = \sum_{i=1}^{K} f_i \mathbf{D}_i' \mathbf{V}_i^{-1} (\mathbf{Y}_i - \boldsymbol{\mu}_i(\boldsymbol{\beta})) = \mathbf{0}$$

where

$$\mathbf{D}_i = \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}$$

Because

$$g(\mu_{ij}) = \mathbf{x}_{ij}{}'\boldsymbol{\beta}$$

where g is the link function, the $p \times n_i$ matrix of partial derivatives of the mean with respect to the regression parameters for the *i*th subject is given by

$$\mathbf{D}_i' = \frac{\partial \boldsymbol{\mu}_i'}{\partial \boldsymbol{\beta}} = \begin{bmatrix} \dfrac{x_{i11}}{g'(\mu_{i1})} & \cdots & \dfrac{x_{in_i1}}{g'(\mu_{in_i})} \\ \vdots & & \vdots \\ \dfrac{x_{i1p}}{g'(\mu_{i1})} & \cdots & \dfrac{x_{in_ip}}{g'(\mu_{in_i})} \end{bmatrix}$$

Working Correlation Matrix

Let $\mathbf{R}_i(\boldsymbol{\alpha})$ be an $n_i \times n_i$ "working" correlation matrix that is fully specified by the vector of parameters $\boldsymbol{\alpha}$. The covariance matrix of $\mathbf{Y}_i$ is modeled as

$$\mathbf{V}_i = \phi \mathbf{A}_i^{\frac{1}{2}} \mathbf{W}_i^{-\frac{1}{2}} \mathbf{R}(\boldsymbol{\alpha}) \mathbf{W}_i^{-\frac{1}{2}} \mathbf{A}_i^{\frac{1}{2}}$$

where $\mathbf{A}_i$ is an $n_i \times n_i$ diagonal matrix with $v(\mu_{ij})$ as the jth diagonal element and $\mathbf{W}_i$ is an $n_i \times n_i$ diagonal matrix with w_{ij} as the jth diagonal, where w_{ij} is a weight specified with the WEIGHT statement. If there is no WEIGHT statement, $w_{ij} = 1$ for all i and j. If $\mathbf{R}_i(\boldsymbol{\alpha})$ is the true correlation matrix of $\mathbf{Y}_i$, then $\mathbf{V}_i$ is the true covariance matrix of $\mathbf{Y}_i$.

The working correlation matrix is usually unknown and must be estimated. It is estimated in the iterative fitting process by using the current value of the parameter vector $\boldsymbol{\beta}$ to compute appropriate functions of the Pearson residual

$$e_{ij} = \frac{y_{ij} - \mu_{ij}}{\sqrt{v(\mu_{ij})/w_{ij}}}$$

If you specify the working correlation as $\mathbf{R}_0 = \mathbf{I}$, which is the identity matrix, the GEE reduces to the independence estimating equation.

Following are the structures of the working correlation supported by the GENMOD procedure and the estimators used to estimate the working correlations.

Working Correlation Structure	Estimator
Fixed $\mathrm{Corr}(Y_{ij}, Y_{ik}) = r_{jk}$ where r_{jk} is the jkth element of a constant, user-specified correlation matrix $\mathbf{R}_0$.	The working correlation is not estimated in this case.
Independent $\mathrm{Corr}(Y_{ij}, Y_{ik}) = \begin{cases} 1 & j = k \\ 0 & j \neq k \end{cases}$	The working correlation is not estimated in this case.
***m*-dependent** $\mathrm{Corr}(Y_{ij}, Y_{i,j+t}) = \begin{cases} 1 & t = 0 \\ \alpha_t & t = 1, 2, \ldots, m \\ 0 & t > m \end{cases}$	$\hat{\alpha}_t = \frac{1}{(K_t - p)\phi} \sum_{i=1}^{K} f_i \sum_{j \leq n_i - t} e_{ij} e_{i,j+t}$ $K_t = \sum_{i=1}^{K} f_i (n_i - t)$
Exchangeable $\mathrm{Corr}(Y_{ij}, Y_{ik}) = \begin{cases} 1 & j = k \\ \alpha & j \neq k \end{cases}$	$\hat{\alpha} = \frac{1}{(N^* - p)\phi} \sum_{i=1}^{K} f_i \sum_{j<k} e_{ij} e_{ik}$ $N^* = 0.5 \sum_{i=1}^{K} f_i n_i (n_i - 1)$
Unstructured $\mathrm{Corr}(Y_{ij}, Y_{ik}) = \begin{cases} 1 & j = k \\ \alpha_{jk} & j \neq k \end{cases}$	$\hat{\alpha}_{jk} = \frac{1}{(K - p)\phi} \sum_{i=1}^{K} f_i e_{ij} e_{ik}$

Table 51.12 *continued*

Working Correlation Structure	Estimator
Autoregressive	
AR(1)	
$\mathrm{Corr}(Y_{ij}, Y_{i,j+t}) = \alpha^t$ for $t = 0, 1, 2, \ldots, n_i - j$	$\hat{\alpha} = \frac{1}{(K_1 - p)\phi} \sum_{i=1}^{K} f_i \sum_{j \le n_i - 1} e_{ij} e_{i,j+1}$ $K_1 = \sum_{i=1}^{K} f_i (n_i - 1)$

Dispersion Parameter

The dispersion parameter ϕ is estimated by

$$\hat{\phi} = \frac{1}{N - p} \sum_{i=1}^{K} f_i \sum_{j=1}^{n_i} e_{ij}^2$$

where $N = \sum_{i=1}^{K} f_i n_i$ is the total number of measurements and p is the number of regression parameters.

The square root of $\hat{\phi}$ is reported by PROC GENMOD as the scale parameter in the "Analysis of GEE Parameter Estimates Model-Based Standard Error Estimates" output table. If a fixed scale parameter is specified with the NOSCALE option in the MODEL statement, then the fixed value is used in estimating the model-based covariance matrix and standard errors.

Fitting Algorithm

The following is an algorithm for fitting the specified model by using GEEs. Note that this is not in general a likelihood-based method of estimation, so that inferences based on likelihoods are not possible for GEE methods.

1. Compute an initial estimate of $\boldsymbol{\beta}$ with an ordinary generalized linear model assuming independence.
2. Compute the working correlations $\mathbf{R}$ based on the standardized residuals, the current $\boldsymbol{\beta}$, and the assumed structure of $\mathbf{R}$.
3. Compute an estimate of the covariance:

$$\mathbf{V}_i = \phi \mathbf{A}_i^{\frac{1}{2}} \mathbf{W}_i^{-\frac{1}{2}} \hat{\mathbf{R}}(\boldsymbol{\alpha}) \mathbf{W}_i^{-\frac{1}{2}} \mathbf{A}_i^{\frac{1}{2}}$$

4. Update $\boldsymbol{\beta}$:

$$\boldsymbol{\beta}_{r+1} = \boldsymbol{\beta}_r + \left[\sum_{i=1}^{K} f_i \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}} \right]^{-1} \left[\sum_{i=1}^{K} f_i \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} (\mathbf{Y}_i - \boldsymbol{\mu}_i) \right]$$

5. Repeat steps 2-4 until convergence.

Missing Data

See Diggle, Liang, and Zeger (1994, Chapter 11) for a discussion of missing values in longitudinal data. Suppose that you intend to take measurements $Y_{i1}, \ldots, Y_{in}$ for the ith unit. Missing values for which Y_{ij} are missing whenever Y_{ik} is missing for all $j \geq k$ are called *dropouts*. Otherwise, missing values that occur intermixed with nonmissing values are *intermittent* missing values. The GENMOD procedure can estimate the working correlation from data containing both types of missing values by using the *all available pairs* method, in which all nonmissing pairs of data are used in the moment estimators of the working correlation parameters defined previously. The resulting covariances and standard errors are valid under the missing completely at random (MCAR) assumption.

For example, for the unstructured working correlation model,

$$\hat{\alpha}_{jk} = \frac{1}{(K' - p)\phi} \sum f_i e_{ij} e_{ik}$$

where the sum is over the units that have nonmissing measurements at times j and k, and K' is the number of units with nonmissing measurements at j and k. Estimates of the parameters for other working correlation types are computed in a similar manner, using available nonmissing pairs in the appropriate moment estimators.

The contribution of the ith unit to the parameter update equation is computed by omitting the elements of $(\mathbf{Y}_i - \boldsymbol{\mu}_\mathrm{i})$, the columns of $\mathbf{D}_i' = \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\beta}}'$, and the rows and columns of $\mathbf{V}_i$ corresponding to missing measurements.

Parameter Estimate Covariances

The *model-based* estimator of $\mathrm{Cov}(\hat{\boldsymbol{\beta}})$ is given by

$$\boldsymbol{\Sigma}_m(\hat{\boldsymbol{\beta}}) = \mathbf{I}_0^{-1}$$

where

$$\mathbf{I}_0 = \sum_{i=1}^{K} f_i \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}$$

This is the GEE equivalent of the inverse of the Fisher information matrix that is often used in generalized linear models as an estimator of the covariance estimate of the maximum likelihood estimator of $\boldsymbol{\beta}$. It is a consistent estimator of the covariance matrix of $\hat{\boldsymbol{\beta}}$ if the mean model and the working correlation matrix are correctly specified.

The estimator

$$\boldsymbol{\Sigma}_e = \mathbf{I}_0^{-1} \mathbf{I}_1 \mathbf{I}_0^{-1}$$

is called the *empirical*, or *robust*, estimator of the covariance matrix of $\hat{\boldsymbol{\beta}}$, where

$$\mathbf{I}_1 = \sum_{i=1}^{K} f_i \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} \mathrm{Cov}(\mathbf{Y}_i) \mathbf{V}_i^{-1} \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}$$

It has the property of being a consistent estimator of the covariance matrix of $\hat{\boldsymbol{\beta}}$, even if the working correlation matrix is misspecified—that is, if $\mathrm{Cov}(\mathbf{Y}_i) \neq \mathbf{V}_i$. For further information about the robust variance estimate, see Zeger, Liang, and Albert (1988); Royall (1986); White (1982). In computing $\boldsymbol{\Sigma}_e$, $\boldsymbol{\beta}$ and ϕ are replaced by estimates, and $\mathrm{Cov}(\mathbf{Y}_i)$ is replaced by the estimate

$$(\mathbf{Y}_i - \boldsymbol{\mu}_i(\hat{\boldsymbol{\beta}}))(\mathbf{Y}_i - \boldsymbol{\mu}_i(\hat{\boldsymbol{\beta}}))'$$

Multinomial GEEs

Lipsitz, Kim, and Zhao (1994) and Miller, Davis, and Landis (1993) describe how to extend GEEs to multinomial data. Currently, only the independent working correlation is available for multinomial models in PROC GENMOD.

Alternating Logistic Regressions

If the responses are binary (that is, they take only two values), then there is an alternative method to account for the association among the measurements. The alternating logistic regressions (ALR) algorithm of Carey, Zeger, and Diggle (1993) models the association between pairs of responses with log odds ratios, instead of with correlations, as ordinary GEEs do.

For binary data, the correlation between the *j*th and *k*th response is, by definition,

$$\mathrm{Corr}(Y_{ij}, Y_{ik}) = \frac{\Pr(Y_{ij} = 1, Y_{ik} = 1) - \mu_{ij}\mu_{ik}}{\sqrt{\mu_{ij}(1-\mu_{ij})\mu_{ik}(1-\mu_{ik})}}$$

The joint probability in the numerator satisfies the following bounds, by elementary properties of probability, since $\mu_{ij} = \Pr(Y_{ij} = 1)$:

$$\max(0, \mu_{ij} + \mu_{ik} - 1) \le \Pr(Y_{ij} = 1, Y_{ik} = 1) \le \min(\mu_{ij}, \mu_{ik})$$

The correlation, therefore, is constrained to be within limits that depend in a complicated way on the means of the data.

The odds ratio, defined as

$$\mathrm{OR}(Y_{ij}, Y_{ik}) = \frac{\Pr(Y_{ij} = 1, Y_{ik} = 1)\Pr(Y_{ij} = 0, Y_{ik} = 0)}{\Pr(Y_{ij} = 1, Y_{ik} = 0)\Pr(Y_{ij} = 0, Y_{ik} = 1)}$$

is not constrained by the means and is preferred, in some cases, to correlations for binary data.

The ALR algorithm seeks to model the logarithm of the odds ratio, $\gamma_{ijk} = \log(\mathrm{OR}(Y_{ij}, Y_{ik}))$, as

$$\gamma_{ijk} = \mathbf{z}'_{ijk}\boldsymbol{\alpha}$$

where $\boldsymbol{\alpha}$ is a $q \times 1$ vector of regression parameters and $\mathbf{z}_{ijk}$ is a fixed, specified vector of coefficients.

The parameter γ_{ijk} can take any value in $(-\infty, \infty)$ with $\gamma_{ijk} = 0$ corresponding to no association.

The log odds ratio, when modeled in this way with a regression model, can take different values in subgroups defined by $\mathbf{z}_{ijk}$. For example, $\mathbf{z}_{ijk}$ can define subgroups within clusters, or it can define "block effects" between clusters.

You specify a GEE model for binary data that uses log odds ratios by specifying a model for the mean, as in ordinary GEEs, and a model for the log odds ratios. You can use any of the link functions appropriate for binary data in the model for the mean, such as logistic, probit, or complementary log-log. The ALR algorithm alternates between a GEE step to update the model for the mean and a logistic regression step to update the log odds ratio model. Upon convergence, the ALR algorithm provides estimates of the regression parameters for the mean, $\boldsymbol{\beta}$, the regression parameters for the log odds ratios, $\boldsymbol{\alpha}$, their standard errors, and their covariances.

Specifying Log Odds Ratio Models

Specifying a regression model for the log odds ratio requires you to specify rows of the $\mathbf{z}$ matrix $\mathbf{z}_{ijk}$ for each cluster i and each unique within-cluster pair (j, k). The GENMOD procedure provides several methods of specifying $\mathbf{z}_{ijk}$. These are controlled by the LOGOR=*keyword* and associated options in the REPEATED statement. The supported *keywords* and the resulting log odds ratio models are described as follows.

EXCH specifies exchangeable log odds ratios. In this model, the log odds ratio is a constant for all clusters i and pairs (j, k). The parameter α is the common log odds ratio.

$$\mathbf{z}_{ijk} = 1 \quad \text{for all} \quad i, j, k$$

FULLCLUST specifies fully parameterized clusters. Each cluster is parameterized in the same way, and there is a parameter for each unique pair within clusters. If a complete cluster is of size n, then there are $\frac{n(n-1)}{2}$ parameters in the vector $\boldsymbol{\alpha}$. For example, if a full cluster is of size 4, then there are $\frac{4\times 3}{2} = 6$ parameters, and the $\mathbf{z}$ matrix is of the form

$$\mathbf{Z} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The elements of $\boldsymbol{\alpha}$ correspond to log odds ratios for cluster pairs in the following order:

Pair	Parameter
(1,2)	Alpha1
(1,3)	Alpha2
(1,4)	Alpha3
(2.3)	Alpha4
(2,4)	Alpha5
(3,4)	Alpha6

LOGORVAR(*variable*) specifies log odds ratios by cluster. The argument *variable* is a variable name that defines the "block effects" between clusters. The log odds ratios are constant within clusters, but they take a different value for each different value of the *variable*. For example, if Center is a variable in the input data set taking a different value for k treatment centers, then specifying LOGOR=LOGORVAR(Center) requests a model with different log odds ratios for each of the k centers, constant within center.

NESTK specifies k-nested log odds ratios. You must also specify the SUBCLUST=*variable* option to define subclusters within clusters. Within each cluster, PROC GENMOD computes a log odds ratio parameter for pairs having the same value of *variable* for both members of the pair and one log odds ratio parameter for each unique combination of different values of *variable*.

NEST1 specifies 1-nested log odds ratios. You must also specify the SUBCLUST=*variable* option to define subclusters within clusters. There are two log odds ratio parameters for this model. Pairs having the same value of *variable* correspond to one parameter; pairs having different values of *variable* correspond to the other parameter. For example, if clusters are hospitals and subclusters are wards within hospitals, then patients within the same ward have one log odds ratio parameter, and patients from different wards have the other parameter.

ZFULL specifies the full **z** matrix. You must also specify a SAS data set containing the **z** matrix with the ZDATA=*data-set-name* option. Each observation in the data set corresponds to one row of the **z** matrix. You must specify the ZDATA data set as if all clusters are complete—that is, as if all clusters are the same size and there are no missing observations. The ZDATA data set has $K[n_{max}(n_{max}-1)/2]$ observations, where K is the number of clusters and n_{max} is the maximum cluster size. If the members of cluster i are ordered as $1, 2, \ldots, n$, then the rows of the **z** matrix must be specified for pairs in the order $(1,2), (1,3), \ldots, (1,n), (2,3), \ldots, (2,n), \ldots, (n-1,n)$. The variables specified in the REPEATED statement for the SUBJECT effect must also be present in the ZDATA= data set to identify clusters. You must specify variables in the data set that define the columns of the **z** matrix by the ZROW=*variable-list* option. If there are q columns (q variables in *variable-list*), then there are q log odds ratio parameters. You can optionally specify variables indicating the cluster pairs corresponding to each row of the **z** matrix with the YPAIR=(*variable1, variable2*) option. If you specify this option, the data from the ZDATA data set are sorted within each cluster by *variable1* and *variable2*. See Example 51.6 for an example of specifying a full **z** matrix.

ZREP specifies a replicated **z** matrix. You specify **z** matrix data exactly as you do for the ZFULL case, except that you specify only one complete cluster. The **z** matrix for the one cluster is replicated for each cluster. The number of observations in the ZDATA data set is $\frac{n_{max}(n_{max}-1)}{2}$, where n_{max} is the size of a complete cluster (a cluster with no missing observations).

ZREP(*matrix*) specifies direct input of the replicated **z** matrix. You specify the **z** matrix for one cluster with the syntax LOGOR=ZREP ($(y_1 \;\; y_2)z_1 \;\; z_2 \cdots z_q, \cdots$), where y_1 and y_2 are numbers representing a pair of observations and the values $z_1, z_2, \ldots, z_q$ make up the corresponding row of the **z** matrix. The number of rows specified is $\frac{n_{max}(n_{max}-1)}{2}$, where n_{max} is the size of a complete cluster (a cluster with no missing observations). For example,

```
logor =  zrep((1 2) 1 0,
              (1 3) 1 0,
              (1 4) 1 0,
              (2 3) 1 1,
              (2 4) 1 1,
              (3 4) 1 1)
```

specifies the $\frac{4\times3}{2} = 6$ rows of the **z** matrix for a cluster of size 4 with $q = 2$ log odds ratio parameters. The log odds ratio for the pairs (1 2), (1 3), (1 4) is α_1, and

the log odds ratio for the pairs (2 3), (2 4), (3 4) is $\alpha_1 + \alpha_2$.

Quasi-likelihood Information Criterion

The quasi-likelihood information criterion (QIC) was developed by Pan (2001) as a modification of the Akaike information criterion (AIC) to apply to models fit by GEEs.

Define the quasi-likelihood under the independence working correlation assumption, evaluated with the parameter estimates under the working correlation of interest as

$$Q(\hat{\boldsymbol{\beta}}(R), \phi) = \sum_{i=1}^{K} f_i \sum_{j=1}^{n_i} Q(\hat{\boldsymbol{\beta}}(R), \phi; (Y_{ij}, \mathbf{X}_{ij}))$$

where the quasi-likelihood contribution of the jth observation in the ith cluster is defined in the section "Quasi-likelihood Functions" on page 3685 and $\hat{\boldsymbol{\beta}}(R)$ are the parameter estimates obtained from GEEs with the working correlation of interest R.

QIC is defined as

$$\mathrm{QIC}(R) = -2Q(\hat{\boldsymbol{\beta}}(R), \phi) + 2\mathrm{trace}(\hat{\Omega}_I \hat{V}_R)$$

where $\hat{V}_R$ is the robust covariance estimate and $\hat{\Omega}_I$ is the inverse of the model-based covariance estimate under the independent working correlation assumption, evaluated at $\hat{\boldsymbol{\beta}}(R)$, the parameter estimates obtained from GEEs with the working correlation of interest R.

PROC GENMOD also computes an approximation to $\mathrm{QIC}(R)$ defined by Pan (2001) as

$$\mathrm{QIC}_u(R) = -2Q(\hat{\boldsymbol{\beta}}(R), \phi) + 2p$$

where p is the number of regression parameters.

Pan (2001) notes that QIC is appropriate for selecting regression models and working correlations, whereas QIC_u is appropriate only for selecting regression models.

Quasi-likelihood Functions

See McCullagh and Nelder (1989) and Hardin and Hilbe (2003) for discussions of quasi-likelihood functions. The contribution of observation j in cluster i to the quasi-likelihood function evaluated at the regression parameters $\boldsymbol{\beta}$ is given by $Q(\boldsymbol{\beta}, \phi; (Y_{ij}, \mathbf{X}_{ij})) = \frac{Q_{ij}}{\phi}$, where Q_{ij} is defined in the following list. These are used in the computation of the quasi-likelihood information criteria (QIC) for goodness of fit of models fit with GEEs. The w_{ij} are prior weights, if any, specified with the WEIGHT or FREQ statements. Note that the definition of the quasi-likelihood for the negative binomial differs from that given in McCullagh and Nelder (1989). The definition used here allows the negative binomial quasi-likelihood to approach the Poisson as $k \to 0$.

- Normal:

$$Q_{ij} = -\frac{1}{2} w_{ij} (y_{ij} - \mu_{ij})^2$$

- Inverse Gaussian:

$$Q_{ij} = \frac{w_{ij}(\mu_{ij} - .5y_{ij})}{\mu_{ij}^2}$$

- Gamma:

$$Q_{ij} = -w_{ij}\left[\frac{y_{ij}}{\mu_{ij}} + \log(\mu_{ij})\right]$$

- Negative binomial:

$$Q_{ij} = w_{ij}\left[\log\Gamma\left(y_{ij} + \frac{1}{k}\right) - \log\Gamma\left(\frac{1}{k}\right) + y_{ij}\log\left(\frac{k\mu_{ij}}{1+k\mu_{ij}}\right) + \frac{1}{k}\log\left(\frac{1}{1+k\mu_{ij}}\right)\right]$$

- Poisson:

$$Q_{ij} = w_{ij}(y_{ij}\log(\mu_{ij}) - \mu_{ij})$$

- Binomial:

$$Q_{ij} = w_{ij}[r_{ij}\log(p_{ij}) + (n_{ij} - r_{ij})\log(1 - p_{ij})]$$

- Multinomial (s categories):

$$Q_{ij} = w_{ij}\sum_{k=1}^{s} y_{ijk}\log(\mu_{ijk})$$

Generalized Score Statistics

Boos (1992) and Rotnitzky and Jewell (1990) describe score tests applicable to testing $\mathbf{L}'\boldsymbol{\beta} = \mathbf{0}$ in GEEs, where $\mathbf{L}'$ is a user-specified $r \times p$ contrast matrix or a contrast for a Type 3 test of hypothesis.

Let $\tilde{\boldsymbol{\beta}}$ be the regression parameters resulting from solving the GEE under the restricted model $\mathbf{L}'\boldsymbol{\beta} = \mathbf{0}$, and let $\mathbf{S}(\tilde{\boldsymbol{\beta}})$ be the generalized estimating equation values at $\tilde{\boldsymbol{\beta}}$.

The generalized score statistic is

$$T = \mathbf{S}(\tilde{\boldsymbol{\beta}})'\boldsymbol{\Sigma}_m\mathbf{L}(\mathbf{L}'\boldsymbol{\Sigma}_e\mathbf{L})^{-1}\mathbf{L}'\boldsymbol{\Sigma}_m\mathbf{S}(\tilde{\boldsymbol{\beta}})$$

where $\boldsymbol{\Sigma}_m$ is the model-based covariance estimate and $\boldsymbol{\Sigma}_e$ is the empirical covariance estimate. The p-values for T are computed based on the chi-square distribution with r degrees of freedom.

The preceding development for score tests assumes that the rank of the empirical covariance matrix $\boldsymbol{\Sigma}_e$ is not less than the row rank of the contrast matrix $\mathbf{L}$. When the rank of $\boldsymbol{\Sigma}_e$ is less than the row rank of $\mathbf{L}$, estimability of the function is not sufficient to ensure that the chi-square test statistic has a unique value no matter what kind of generalized inverse is used to compute $(\mathbf{L}'\boldsymbol{\Sigma}_e\mathbf{L})^{-}$.

Although it is extremely rare, it is possible in practice that the uniqueness condition is not satisfied. For example, if the number of clusters is less than the number of nonsingular parameters in the model, then the matrix of coefficients for testing the overall null does not satisfy the uniqueness condition. If this condition is not satisfied, then the chi-square statistic for testing $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ is not invariant to the choice of the g_2-inverse of $\mathbf{L}\boldsymbol{\Sigma}_e\mathbf{L}'$. This chi-square test is not recommended when the uniqueness condition is not satisfied. An alternative approach would be to increase the number of clusters or to find a parsimonious model so that the number of parameters is less than the number of clusters. When the rank of $\boldsymbol{\Sigma}_e$ is less than the row rank of $\mathbf{L}$ for a test, the procedure prints a note to the SAS log.

Assessment of Models Based on Aggregates of Residuals

Lin, Wei, and Ying (2002) present graphical and numerical methods for model assessment based on the cumulative sums of residuals over certain coordinates (such as covariates or linear predictors) or some related aggregates of residuals. The distributions of these stochastic processes under the assumed model can be approximated by the distributions of certain zero-mean Gaussian processes whose realizations can be generated by simulation. Each observed residual pattern can then be compared, both graphically and numerically, with a number of realizations from the null distribution. Such comparisons enable you to assess objectively whether the observed residual pattern reflects anything beyond random fluctuation. These procedures are useful in determining appropriate functional forms of covariates and link function. You use the ASSESS|ASSESSMENT statement to perform this kind of model-checking with cumulative sums of residuals, moving sums of residuals, or LOESS smoothed residuals. See Example 51.8 and Example 51.9 for examples of model assessment.

Let the model for the mean be

$$g(\mu_i) = \mathbf{x}_i'\boldsymbol{\beta}$$

where μ_i is the mean of the response y_i and $\mathbf{x}_i$ is the vector of covariates for the ith observation. Denote the raw residual resulting from fitting the model as

$$e_i = y_i - \hat{\mu}_i$$

and let x_{ij} be the value of the jth covariate in the model for observation i. Then to check the functional form of the jth covariate, consider the cumulative sum of residuals with respect to x_{ij},

$$W_j(x) = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} I(x_{ij} \le x) e_i$$

where $I()$ is the indicator function. For any x, $W_j(x)$ is the sum of the residuals with values of x_j less than or equal to x.

Denote the score, or gradient vector, by

$$U(\boldsymbol{\beta}) = \sum_{i=1}^{n} h(\mathbf{x}'\boldsymbol{\beta})\mathbf{x}_i(y_i - \nu(\mathbf{x}'\boldsymbol{\beta}))$$

where $\nu(r) = g^{-1}(r)$, and

$$h(r) = \frac{1}{g'(\nu(r))V(\nu(r))}$$

Let $\mathbf{J}$ be the Fisher information matrix

$$\mathbf{J}(\boldsymbol{\beta}) = -\frac{\partial U(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}'}$$

Define

$$\hat{W}_j(x) = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} [I(x_{ij} \le x) + \boldsymbol{\eta}'(x;\hat{\boldsymbol{\beta}})\mathbf{J}^{-1}(\hat{\boldsymbol{\beta}})\mathbf{x}_i h(\mathbf{x}'\hat{\boldsymbol{\beta}})] e_i Z_i$$

where

$$\boldsymbol{\eta}(x;\boldsymbol{\beta}) = -\sum_{i=1}^{n} I(x_{ij} \le x)\frac{\partial \nu(\mathbf{x}_i'\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$$

and Z_i are independent $N(0,1)$ random variables. Then the conditional distribution of $\hat{W}_j(x)$, given $(y_i, \mathbf{x}_i), i = 1,\ldots,n$, under the null hypothesis H_0 that the model for the mean is correct, is the same asymptotically as $n \to \infty$ as the unconditional distribution of $W_j(x)$ (Lin, Wei, and Ying 2002).

You can approximate realizations from the null hypothesis distribution of $W_j(x)$ by repeatedly generating normal samples $Z_i, i = 1,\ldots,n$, while holding $(y_i, \mathbf{x}_i), i = 1,\ldots,n$, at their observed values and computing $\hat{W}_j(x)$ for each sample.

You can assess the functional form of covariate *j* by plotting a few realizations of $\hat{W}_j(x)$ on the same plot as the observed $W_j(x)$ and visually comparing to see how typical the observed $W_j(x)$ is of the null distribution samples.

You can supplement the graphical inspection method with a Kolmogorov-type supremum test. Let s_j be the observed value of $S_j = \sup_x |W_j(x)|$. The *p*-value $\Pr[S_j \ge s_j]$ is approximated by $\Pr[\hat{S}_j \ge s_j]$, where $\hat{S}_j = \sup_x |\hat{W}_j(x)|$. $\Pr[\hat{S}_j \ge s_j]$ is estimated by generating realizations of $\hat{W}_j(.)$ (1,000 is the default number of realizations).

You can check the link function instead of the *j*th covariate by using values of the linear predictor $\mathbf{x}_i'\hat{\boldsymbol{\beta}}$ in place of values of the *j*th covariate x_{ij}. The graphical and numerical methods described previously are then sensitive to inadequacies in the link function.

An alternative aggregate of residuals is the moving sum statistic

$$W_j(x,b) = \frac{1}{\sqrt{n}}\sum_{i=1}^{n} I(x-b \le x_{ij} \le x)e_i$$

If you specify the keyword WINDOW(*b*), then the moving sum statistic with window size *b* is used instead of the cumulative sum of residuals, with $I(x-b \le x_{ij} \le x)$ replacing $I(x_{ij} \le x)$ in the earlier equation.

If you specify the keyword LOESS(*f*), loess smoothed residuals are used in the preceding formulas, where *f* is the fraction of the data to be used at a given point. If *f* is not specified, $f = \frac{1}{3}$ is used. For data $(Y_i, X_i), i = 1,\ldots,n$, define *r* as the nearest integer to nf and *h* as the *r*th smallest among $|X_i - x|, i = 1,\ldots,n$. Let

$$K_i(x) = K\left(\frac{X_i - x}{h}\right)$$

where

$$K(t) = \frac{70}{81}(1-|t|^3)^3 I(-1 \le t \le 1)$$

Define

$$w_i(x) = K_i(x)[S_2(x) - (X_i - x)S_1(x)]$$

where

$$S_1(x) = \sum_{i=1}^{n} K_i(x)(X_i - x)$$

$$S_2(x) = \sum_{i=1}^{n} K_i(x)(X_i - x)^2$$

Then the loess estimate of Y at x is defined by

$$\hat{Y}(x) = \sum_{i=1}^{n} \frac{w_i(x)}{\sum_{i=1}^{n} w_i(x)} Y_i$$

Loess smoothed residuals for checking the functional form of the jth covariate are defined by replacing Y_i with e_i and X_i with x_{ij}. To implement the graphical and numerical assessment methods, $I(x_{ij} \leq x)$ is replaced with $\frac{w_i(x)}{\sum_{i=1}^{n} w_i(x)}$ in the formulas for $W_j(x)$ and $\hat{W}_j(x)$.

You can perform the model checking described earlier for marginal models for dependent responses fit by generalized estimating equations (GEEs). Let y_{ik} denote the kth measurement on the ith cluster, $i = 1, \ldots, K$, $k = 1, \ldots, n_i$, and let $\mathbf{x}_{ik}$ denote the corresponding vector of covariates. The marginal mean of the response $\mu_{ik} = \mathrm{E}(y_{ik})$ is assumed to depend on the covariate vector by

$$g(\mu_{ik}) = \mathbf{x}_{ik}'\boldsymbol{\beta}$$

where g is the link function.

Define the vector of residuals for the ith cluster as

$$\mathbf{e}_i = (e_{i1}, \ldots, e_{in_i})' = (y_{i1} - \hat{\mu}_{i1}, \ldots, y_{in_i} - \hat{\mu}_{in_i})'$$

You use the following extension of $W_j(x)$ defined earlier to check the functional form of the jth covariate:

$$W_j(x) = \frac{1}{\sqrt{K}} \sum_{i=1}^{K} \sum_{k=1}^{n_i} I(x_{ikj} \leq x) e_{ik}$$

where x_{ikj} is the jth component of $\mathbf{x}_{ik}$.

The null distribution of $W_j(x)$ can be approximated by the conditional distribution of

$$\hat{W}_j(x) = \frac{1}{\sqrt{K}} \sum_{i=1}^{K} \left\{ \sum_{k=1}^{n_i} I(x_{ikj} \leq x) e_{ik} + \boldsymbol{\eta}'(x, \hat{\boldsymbol{\beta}}) \mathbf{I}_0^{-1} \hat{\mathbf{D}}_i' \hat{\mathbf{V}}_i^{-1} \mathbf{e}_i \right\} Z_i$$

where $\hat{\mathbf{D}}_i$ and $\hat{\mathbf{V}}_i$ are defined as in the section "Generalized Estimating Equations" on page 3678 with the unknown parameters replaced by their estimated values,

$$\boldsymbol{\eta}(x, \boldsymbol{\beta}) = -\sum_{i=1}^{K}\sum_{k=1}^{n_i} I(x_{ikj} \le x)\frac{\partial \mu_{ik}}{\partial \boldsymbol{\beta}}$$

$$\mathbf{I}_0 = \sum_{i=1}^{K} \hat{\mathbf{D}}_i' \hat{\mathbf{V}}_i^{-1} \hat{\mathbf{D}}_i$$

and $Z_i, i = 1, \ldots, K$, are independent $N(0, 1)$ random variables. You replace x_{ikj} with the linear predictor $\mathbf{x}_{ik}'\hat{\boldsymbol{\beta}}$ in the preceding formulas to check the link function.

Case Deletion Diagnostic Statistics

For ordinary generalized linear models, regression diagnostic statistics developed by Williams (1987) can be requested in an output data set or in the OBSTATS table by specifying the DIAGNOSTICS | INFLUENCE option in the MODEL statement. These diagnostics measure the influence of an individual observation on model fit, and generalize the one-step diagnostics developed by Pregibon (1981) for the logistic regression model for binary data.

Preisser and Qaqish (1996) further generalized regression diagnostics to apply to models for correlated data fit by generalized estimating equations (GEEs), where the influence of entire clusters of correlated observations, or the influence of individual observations within a cluster, is measured. These diagnostic statistics can be requested in an output data set or in the OBSTATS table if a model for correlated data is specified with a REPEATED statement.

The next two sections use the following notation:

$\hat{\boldsymbol{\beta}}$ is the maximum likelihood estimate of the regression parameters $\boldsymbol{\beta}$, or, in the case of correlated data, the solution of the GEEs.

$\hat{\boldsymbol{\beta}}_{[i]}$ is the corresponding estimate evaluated with the *i*th observation deleted, or, in the case of correlated data, with the *i*th cluster deleted.

p is the dimension of the regression parameter vector $\boldsymbol{\beta}$.

r_{pi} is the standardized Pearson residual $\frac{y_i - \mu_i}{\sqrt{v_i(1-h_i)}}$, where v_i is the variance of the *i*th response and h_i is the leverage defined in the section "H | LEVERAGE" on page 3691.

v_i is the variance of response i, $\mathrm{Var}(Y_i) = \phi V(\mu_i)$, where $V(\mu)$ is the variance function and ϕ is the dispersion parameter.

w_i is the prior weight of the *i*th observation specified with the WEIGHT statement. If there is no WEIGHT statement, $w_i = 1$ for all i.

All unknown quantities are replaced by their estimated values in the following two sections.

Diagnostics for Ordinary Generalized Linear Models

The following statistics are available for generalized linear models.

DFBETA

The DFBETA statistic for measuring the influence of the ith observation is defined as the one-step approximation to the difference in the MLE of the regression parameter vector and the MLE of the regression parameter vector without the ith observation. This one-step approximation assumes a Fisher scoring step, and is given by

$$\hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{[i]} \approx \mathrm{DFBETA}_i = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}_i'\mathbf{W}_i^{\frac{1}{2}}(1-h_i)^{-\frac{1}{2}}r_{pi}$$

where h_i is the leverage defined in the section "H | LEVERAGE" on page 3691.

DFBETAS

The standardized DFBETA statistic for assessing the influence of the ith observation on the jth regression parameter is defined as the DFBETA statistic for the jth parameter divided by its estimated standard deviation, where the standard deviation is estimated from all the data.

$$\mathrm{DFBETAS}_{ij} = \mathrm{DFBETA}_{ij}/\hat{\sigma}(\beta_j)$$

DOBS | COOKD | COOKSD

In normal linear regression, the influence of observation i can be measured by Cook's distance (Cook and Weisberg 1982). A measure of influence of observation i for generalized linear models that is equivalent to Cook's distance for normal linear regression is given by

$$\mathrm{DOBS}_i = p^{-1}h_i(1-h_i)^{-1}r_{pi}^2$$

where h_i is the leverage defined in the section "H | LEVERAGE" on page 3691. This measure is the one-step approximation to $2p^{-1}[L(\hat{\boldsymbol{\beta}}) - L(\hat{\boldsymbol{\beta}}_{[i]})]$, where $L(\boldsymbol{\beta})$ is the log likelihood evaluated at $\boldsymbol{\beta}$.

H | LEVERAGE

The Fisher scores, or expected, weight for observation i is $w_{ei} = \frac{w_i}{\phi V(\mu_i)(g'(\mu_i))^2}$. Let $\mathbf{W}$ be the diagonal matrix with w_{ei} as the ith diagonal. The leverage h_i of the ith observation is defined as the ith diagonal element of the hat matrix

$$\mathbf{H} = \mathbf{W}^{\frac{1}{2}}\mathbf{X}(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}^{\frac{1}{2}}$$

Diagnostics for Models Fit by Generalized Estimating Equations (GEEs)

The diagnostic statistics in this section were developed by Preisser and Qaqish (1996). See the section "Generalized Estimating Equations" on page 3678 for further information and notation for generalized estimating equations (GEEs). The following additional notation is used in this section.

Partition the design matrix $\mathbf{X}$ and response vector $\mathbf{Y}$ by cluster; that is, let $\mathbf{X} = (X_1', \ldots, X_K')'$, and $\mathbf{Y} = (Y_1', \ldots, Y_K')'$ corresponding to the K clusters.

Let n_i be the number of responses for cluster i, and denote by $N = \sum_{i=1}^{K} n_i$ the total number of observations. Denote by A_i the $n_i \times n_i$ diagonal matrix with $V(\mu_{ij})$ as the jth diagonal element. If there is a WEIGHT statement, the diagonal element of A_i is $V(\mu_{ij})/w_{ij}$, where w_{ij} is the specified weight of the jth observation in the ith cluster. Let $\mathbf{B}$ the $N \times N$ diagonal matrix with $g'(\mu_{ij})$ as diagonal elements, $i = 1, \ldots, K$, $j = 1, \ldots, n_i$. Let $\mathbf{B}_i$ the $n_i \times n_i$ diagonal matrix corresponding to cluster i with $g'(\mu_{ij})$ as the jth diagonal element.

Let $\mathbf{W}$ be the $N \times N$ block diagonal weight matrix whose ith block, corresponding to the ith cluster, is the $n_i \times n_i$ matrix

$$\mathbf{W}_{ei} = \mathbf{B}_i^{-1}\mathbf{A}_i^{-\frac{1}{2}}\mathbf{R}_i^{-1}(\hat{\boldsymbol{\alpha}})\mathbf{A}_i^{-\frac{1}{2}}\mathbf{B}_i^{-1}$$

where $\mathbf{R}_i$ is the working correlation matrix for cluster i.

Let

$$Q_i = \mathbf{X}_i(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}_i'$$

where $\mathbf{X}_i$ is the $n_i \times p$ design matrix corresponding to cluster i.

Define the adjusted residual vector as

$$\mathbf{E} = \mathbf{B}(\mathbf{Y} - \hat{\boldsymbol{\mu}})$$

and $\mathbf{E}_i = \mathbf{B}_i(\mathbf{Y}_i - \hat{\boldsymbol{\mu}}_i)$, the estimated residual for the ith cluster.

Let the subscript $[i]$ denote estimates evaluated without the ith cluster, $[it]$ estimates evaluated using all the data except the tth observation of the ith cluster, and let $i[t]$ denote matrices corresponding to the ith cluster without the tth observation.

The following statistics are available for generalized estimating equation models.

CH | CLUSTERH | CLEVERAGE

The leverage of cluster i is contained in the matrix $\mathbf{H}_i = \mathbf{Q}_i\mathbf{W}_{ei}$, and is summarized by the trace of $\mathbf{H}_i$,

$$ch_i = \mathrm{tr}(\mathbf{H}_i)$$

The leverage h_i of the tth observation in the ith cluster is the tth diagonal element of $\mathbf{H}_i$.

DFBETAC

The effect of deleting cluster i on the estimated parameter vector is given by the following one-step approximation for $\hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{[i]}$:

$$\mathrm{DBETAC}_i = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}_i'(\mathbf{W}_{ei}^{-1} - \mathbf{Q}_i)^{-1}\mathbf{E}_i$$

DFBETACS

The cluster deletion statistic DFBETAC can be standardized using the variances of $\hat{\boldsymbol{\beta}}$ based on the complete data. The standardized one-step approximation for the change in $\hat{\beta}_j$ due to deletion of cluster i is

$$\mathrm{DBETACS}_{ij} = \frac{\mathrm{DBETAC}_{ij}}{\hat{\phi}[(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}]_{jj}^{\frac{1}{2}}}$$

DFBETA

Partition the matrices $\mathbf{W}_{ei}$ and $\mathbf{V}_i$ as

$$\mathbf{W}_{ei} = \begin{pmatrix} W_{eit} & \mathbf{W}_{eit[t]} \\ \mathbf{W}_{ei[t]t} & \mathbf{W}_{ei[t]} \end{pmatrix}$$

$$\mathbf{V}_i = \mathbf{W}_{ei}^{-1} = \begin{pmatrix} \mathbf{V}_{it} & \mathbf{V}_{it[t]} \\ \mathbf{V}_{i[t]t} & \mathbf{V}_{i[t]} \end{pmatrix}$$

and let $\mathbf{E}_{it} = \mathbf{B}_{it}(\mathbf{Y}_{it} - \hat{\mu}_{it})$ and $\mathbf{E}_{i[t]} = \mathbf{B}_{i[t]}(\mathbf{Y}_{i[t]} - \hat{\mu}_{i[t]})$.

The effect of deleting the tth observation from the ith cluster is given by the following one-step approximation to $\hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{[it]}$:

$$\mathbf{DBETAO}_{it} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\tilde{\mathbf{X}}'_{it}\frac{\tilde{E}_{it}}{W_{eit}^{-1} - \tilde{Q}_{it}}$$

where $\tilde{\mathbf{X}}_{it} = \mathbf{X}_{it} - \mathbf{V}_{it[t]}\mathbf{V}_{i[t]}^{-1}\mathbf{X}_{i[t]}$, $\tilde{Q}_{it} = \tilde{\mathbf{X}}_{it}(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\tilde{X}'_{it}$, and $\tilde{E}_{it} = \mathbf{E}_{it} - \mathbf{V}_{it[t]}\mathbf{V}_{i[t]}^{-1}\mathbf{E}_{i[t]}$. Note that W_{eit}, $\tilde{Q}_{it}$, and $\tilde{E}_{it}$ are scalars.

DFBETAS

The observation deletion statistic DFBETA can be standardized using the variances of $\hat{\boldsymbol{\beta}}$ based on the complete data. The standardized one-step approximation for the change in $\hat{\beta}_j$ due to deletion of observation t in cluster i is

$$\mathrm{DBETAOS}_{itj} = \frac{\mathrm{DBETAO}_{itj}}{\hat{\phi}[(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}]_{jj}^{\frac{1}{2}}}$$

DCLS | CLUSTERCOOKD | CLUSTERCOOKSD

A measure of the standardized influence of the subset m of observations on the overall fit is $(\hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{[m]})'(\mathbf{X}'\mathbf{W}\mathbf{X})(\hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{[m]})/p\hat{\phi}$. For deletion of cluster i, this is approximated by

$$\mathrm{DCLS}_i = \mathbf{E}_i'(\mathbf{W}_{ei}^{-1} - \mathbf{Q}_i)^{-1})\mathbf{Q}_i(\mathbf{W}_{ei}^{-1} - \mathbf{Q}_i)^{-1})\mathbf{E}_i/p\hat{\phi}$$

DOBS | COOKD | COOKSD

The measure of overall fit in the section "DCLS | CLUSTERCOOKD | CLUSTERCOOKSD" on page 3693 for the deletion of the tth observation in the ith cluster is approximated by

$$\mathrm{DOBS}_{it} = \frac{\tilde{E}_{it}^2\tilde{Q}_{it}}{p\hat{\phi}(W_{eit}^{-1} - \tilde{Q}_{it})^2}$$

where $\tilde{E}_{it}$, $\tilde{Q}_{it}$, and W_{eit} are defined in the section "DFBETA" on page 3693. In the case of the independence working correlation, this is equal to the measure for ordinary generalized linear models defined in the section "DOBS | COOKD | COOKSD" on page 3691.

MCLS | CLUSTERDFIT

A studentized distance measure of the type defined in the section "DCLS | CLUSTERCOOKD | CLUSTERCOOKSD" on page 3693 of the influence of the ith cluster is given by

$$\mathrm{MCLS}_i = \mathbf{E}_i'(\mathbf{W}_{ei}^{-1} - \mathbf{Q}_i)^{-1}\mathbf{H}_i\mathbf{E}_i / p\hat{\phi}$$

Bayesian Analysis

In generalized linear models, the response has a probability distribution from a family of distributions of the exponential form. That is, the probability density of the response Y for continuous response variables, or the probability function for discrete responses, can be expressed as

$$f(y) = \exp\left\{\frac{y\theta - b(\theta)}{a(\phi)} + c(y,\phi)\right\}$$

for some functions a, b, and c that determine the specific distribution. The canonical parameters θ depend only on the means of the response μ_i, which are related to the regression parameters β through the link function $g(\mu_i) = x'\beta$. The additional parameter ϕ is the dispersion parameter. The GENMOD procedure estimates the regression parameters and the scale parameter $\sigma = \phi^{\frac{1}{2}}$ by maximum likelihood. However, the GENMOD procedure can also provide Bayesian estimates of the regression parameters and either the scale σ, the dispersion ϕ, or the precision $\tau = \phi^{-1}$ by sampling from the posterior distribution. Except where noted, the following discussion applies to either σ, ϕ, or τ, although ϕ is used to illustrate the formulas. Note that the Poisson and binomial distributions do not have a dispersion parameter, and the dispersion is considered to be fixed at $\phi = 1$. The ASSESS, CONTRAST, ESTIMATE, OUTPUT, and REPEATED statements, if specified, are ignored. Also ignored are the PLOTS= option in the PROC GENMOD statement and the following options in the MODEL statement: ALPHA=, CORRB, COVB, TYPE1, TYPE3, SCALE=DEVIANCE (DSCALE), SCALE=PEARSON (PSCALE), OBSTATS, RESIDUALS, XVARS, PREDICTED, DIAGNOSTICS, and SCALE= for Poisson and binomial distributions. The multinomial and zero-inflated Poisson distributions are not available for Bayesian analysis.

See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for information about assessing the convergence of the chain of posterior samples.

Several algorithms, specified with the SAMPLING= option in the BAYES statement, are available in GENMOD for drawing samples from the posterior distribution.

ARMS Algorithm for Gibbs Sampling

This section provides details for Bayesian analysis by Gibbs sampling in generalized linear models. See the section "Gibbs Sampler" on page 157 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a general discussion of Gibbs sampling. See Gilks, Richardson, and Spiegelhalter (1996) for a discussion of applications of Gibbs sampling to a number of different models, including generalized linear models.

Let $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)'$ be the parameter vector. For generalized linear models, the θ_is are the regression coefficients β_is and the dispersion parameter ϕ. Let $L(D|\boldsymbol{\theta})$ be the likelihood function, where D is the observed data. Let $\pi(\boldsymbol{\theta})$ be the prior distribution. The full conditional distribution of $[\theta_i|\theta_j, i \neq j]$ is proportional to the joint distribution; that is,

$$\pi(\theta_i|\theta_j, i \neq j, D) \propto L(D|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

For instance, the one-dimensional conditional distribution of θ_1 given $\theta_j = \theta_j^*, 2 \le j \le k$, is computed as

$$\pi(\theta_1|\theta_j = \theta_j^*, 2 \le j \le k, D) = L(D|(\boldsymbol{\theta} = (\theta_1, \theta_2^*, \ldots, \theta_k^*)')p(\boldsymbol{\theta} = (\theta_1, \theta_2^*, \ldots, \theta_k^*)')$$

Suppose you have a set of arbitrary starting values $\{\theta_1^{(0)}, \ldots, \theta_k^{(0)}\}$. Using the ARMS (adaptive rejection Metropolis sampling) algorithm (Gilks and Wild 1992; Gilks, Best, and Tan 1995), you can do the following:

draw $\theta_1^{(1)}$ from $[\theta_1|\theta_2^{(0)}, \ldots, \theta_k^{(0)}]$

draw $\theta_2^{(1)}$ from $[\theta_2|\theta_1^{(1)}, \theta_3^{(0)}, \ldots, \theta_k^{(0)}]$

...

draw $\theta_k^{(1)}$ from $[\theta_k|\theta_1^{(1)}, \ldots, \theta_{k-1}^{(1)}]$

This completes one iteration of the Gibbs sampler. After one iteration, you have $\{\theta_1^{(1)}, \ldots, \theta_k^{(1)}\}$. After n iterations, you have $\{\theta_1^{(n)}, \ldots, \theta_k^{(n)}\}$. PROC GENMOD implements the ARMS algorithm provided by Gilks (2003) to draw a sample from a full conditional distribution. See the section "Adaptive Rejection Sampling Algorithm" on page 158 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more information about the ARMS algorithm.

Gamerman Algorithm

The Gamerman algorithm, unlike a Gibbs sampling algorithm, samples parameters from their multivariate posterior conditional distribution. The algorithm uses the structure of generalized linear models to efficiently sample from the posterior distribution of the model parameters. For a detailed description and explanation of the algorithm, see Gamerman (1997) and the section "Gamerman Algorithm" on page 160 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The Gamerman algorithm is the default method used to sample from the posterior distribution. See any of the introductory references in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a discussion of conjugate prior distributions for a linear model with the normal distribution.

Independence Metropolis Algorithm

The independence Metropolis algorithm is another sampling algorithm that draws multivariate samples from the posterior distribution. See the section "Independence Sampler" on page 160 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

Posterior Samples Output Data Set

You can output posterior samples into a SAS data set through ODS. The following SAS statement outputs the posterior samples into the SAS data set Post:

ODS OUTPUT POSTERIORSAMPLE=*Post*

You can alternatively create the SAS data set Post with the **OUTPOST=***Post* option in the BAYES statement.

The data set also includes the variables LogPost and LogLike, which represent the log of the posterior likelihood and the log of the likelihood, respectively.

Priors for Model Parameters

The model parameters are the regression coefficients and the dispersion parameter (or the precision or scale), if the model has one. The priors for the dispersion parameter and the priors for the regression coefficients are assumed to be independent, while you can have a joint multivariate normal prior for the regression coefficients.

Dispersion, Precision, or Scale Parameter

Gamma Prior The gamma distribution $G(a, b)$ has a probability density function

$$f(u) = \frac{b(bu)^{a-1}\mathrm{e}^{-bu}}{\Gamma(a)}, \qquad u > 0$$

where a is the shape parameter and b is the inverse-scale parameter. The mean is $\frac{a}{b}$ and the variance is $\frac{a}{b^2}$.

Improper Prior The joint prior density is given by

$$p(u) \propto u^{-1}, \qquad u > 0$$

Inverse Gamma Prior The inverse gamma distribution $\mathrm{IG}(a, b)$ has a probability density function

$$f(u) = \frac{b^a}{\Gamma(a)} u^{-(a+1)}\mathrm{e}^{-b/u}, \qquad u > 0$$

where a is the shape parameter and b is the scale parameter. The mean is $\frac{b}{a-1}$ if $a > 1$, and the variance is $\frac{b^2}{(a-1)^2(a-2)}$ if $a > 2$.

Regression Coefficients

Let $\boldsymbol{\beta}$ be the regression coefficients.

Jeffreys' Prior The joint prior density is given by

$$p(\boldsymbol{\beta}) \propto |\mathbf{I}(\boldsymbol{\beta})|^{\frac{1}{2}}$$

where $\mathbf{I}(\boldsymbol{\beta})$ is the Fisher information matrix for the model. If the underlying model has a scale parameter (for example, a normal linear regression model), then the Fisher information matrix is computed with the scale parameter set to a fixed value of one.

If you specify the CONDITIONAL option, then Jeffreys' prior, conditional on the current Markov chain value of the generalized linear model precision parameter τ, is given by

$$|\tau\mathbf{I}(\boldsymbol{\beta})|^{\frac{1}{2}}$$

where τ is the model precision parameter.

See Ibrahim and Laud (1991) for a full discussion, with examples, of Jeffreys' prior for generalized linear models.

Normal Prior Assume $\boldsymbol{\beta}$ has a multivariate normal prior with mean vector $\boldsymbol{\beta}_0$ and covariance matrix $\boldsymbol{\Sigma}_0$. The joint prior density is given by

$$p(\boldsymbol{\beta}) \propto \mathrm{e}^{-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\beta}_0)'\boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\beta}_0)}$$

If you specify the CONDITIONAL option, then, conditional on the current Markov chain value of the generalized linear model precision parameter τ, the joint prior density is given by

$$p(\boldsymbol{\beta}) \propto \mathrm{e}^{-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\beta}_0)'\tau\boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\beta}_0)}$$

Uniform Prior The joint prior density is given by

$$p(\boldsymbol{\beta}) \propto 1$$

Deviance Information Criterion

Let θ_i be the model parameters at iteration i of the Gibbs sampler and let $\mathrm{LL}(\theta_i)$ be the corresponding model log likelihood. PROC GENMOD computes the following fit statistics defined by Spiegelhalter et al. (2002):

- Effective number of parameters:

$$p_D = \overline{\mathrm{LL}(\theta)} - \mathrm{LL}(\bar{\theta})$$

- Deviance information criterion (DIC):

$$\mathrm{DIC} = \overline{\mathrm{LL}(\theta)} + p_D$$

where

$$\begin{aligned} \overline{\mathrm{LL}(\theta)} &= \frac{1}{n}\sum_{i=1}^{n}\mathrm{LL}(\theta_i) \\ \bar{\theta} &= \frac{1}{n}\sum_{i=1}^{n}\theta_i \end{aligned}$$

PROC GENMOD uses the full log likelihoods defined in the section "Log-Likelihood Functions" on page 3659, with all terms included, for computing the DIC.

Posterior Distribution

Denote the observed data by D.

The posterior distribution is

$$\pi(\boldsymbol{\beta}|D) \propto L_P(D|\boldsymbol{\beta})p(\boldsymbol{\beta})$$

where $L_P(D|\boldsymbol{\beta})$ is the likelihood function with regression coefficients $\boldsymbol{\beta}$ as parameters.

Starting Values of the Markov Chains

When the BAYES statement is specified, PROC GENMOD generates one Markov chain containing the approximate posterior samples of the model parameters. Additional chains are produced when the Gelman-Rubin diagnostics are requested. Starting values (or initial values) can be specified in the INITIAL= data set in the BAYES statement. If INITIAL= option is not specified, PROC GENMOD picks its own initial values for the chains.

Denote $[x]$ as the integral value of x. Denote $\hat{s}(X)$ as the estimated standard error of the estimator X.

Regression Coefficients

For the first chain that the summary statistics and regression diagnostics are based on, the default initial values are estimates of the mode of the posterior distribution. If the INITIALMLE option is specified, the initial values are the maximum likelihood estimates; that is,

$$\beta_i^{(0)} = \hat{\beta}_i$$

Initial values for the rth chain ($r \geq 2$) are given by

$$\beta_i^{(0)} = \hat{\beta}_i \pm \left(2 + \left[\frac{r}{2}\right]\right)\hat{s}(\hat{\beta}_i)$$

with the plus sign for odd r and minus sign for even r.

Dispersion, Scale, or Precision Parameter λ

Let λ be the generalized linear model parameter you choose to sample, either the dispersion, scale, or precision parameter. Note that the Poisson and binomial distributions do not have this additional parameter.

For the first chain that the summary statistics and regression diagnostics are based on, the default initial values are estimates of the mode of the posterior distribution. If the INITIALMLE option is specified, the initial values are the maximum likelihood estimates; that is,

$$\lambda^{(0)} = \hat{\lambda}$$

The initial values of the rth chain ($r \geq 2$) are given by

$$\lambda^{(0)} = \hat{\lambda}\mathrm{e}^{\pm\left([\frac{r}{2}]+2\right)\hat{s}(\hat{\lambda})}$$

with the plus sign for odd r and minus sign for even r.

OUTPOST= Output Data Set

The OUTPOST= data set contains the generated posterior samples. There are 3+n variables, where n is the number of model parameters. The variable Iteration represents the iteration number, the variable LogLike contains the log of the likelihood, and the variable LogPost contains the log of the posterior. The other n variables represent the draws of the Markov chain for the model parameters.

Exact Logistic and Exact Poisson Regression

The theory of exact logistic regression, also called exact conditional logistic regression, is described in the section "Exact Conditional Logistic Regression" on page 6071 in Chapter 79, "The LOGISTIC Procedure." The following discussion of exact Poisson regression, also called exact conditional Poisson regression, uses the notation given in that section.

Note that in exact logistic regression, the coefficients $C(\boldsymbol{t})$ are the number of possible response vectors $\boldsymbol{y}$ that generate $\boldsymbol{t}$: $C(\boldsymbol{t}) = ||\{\boldsymbol{y} : \boldsymbol{y}'\mathbf{X} = \boldsymbol{t}'\}||$. However, when performing an exact Poisson regression, this value is replaced by

$$C(\boldsymbol{t}) = \sum_{\Omega} \prod_{i=1}^{n} \frac{N_i^{y_i}}{y_i!}$$

where $\Omega = \{\boldsymbol{y}: \boldsymbol{y}'\mathbf{X} = \boldsymbol{t}\}$ and $N_i = \exp(o_i)$ is the exponential of the offset o_i for observation i. If an offset variable is not specified, then $N_i = 1$.

The probability density function (PDF) for $\mathbf{T}$ is created by summing over all candidate sequences $\boldsymbol{y}$ that generate an observable $\boldsymbol{t}$

$$\Pr(\mathbf{T} = \boldsymbol{t}) = \frac{C(\boldsymbol{t}) \exp(\boldsymbol{t}'\boldsymbol{\beta})}{\prod_{i=1}^{n} \exp(N_i e^{\boldsymbol{x}_i'\boldsymbol{\beta}})}$$

However, the conditional likelihood of $\mathbf{T}_{\mathrm{I}}$ given $\mathbf{T}_{\mathrm{N}} = \boldsymbol{t}_{\mathrm{N}}$ has the same form as that for exact logistic regression.

For details about hypothesis testing and estimation, see the sections "Hypothesis Tests" on page 6073 and "Inference for a Single Parameter" on page 6074 in Chapter 79, "The LOGISTIC Procedure." See the section "Computational Resources for Exact Logistic Regression" on page 6082 in Chapter 79, "The LOGISTIC Procedure," for some computational notes about exact analyses.

In exact logistic binary regression, each component $y_i, i = 1, \ldots, n$, of $\boldsymbol{y}$ can take a value of 0 or 1, so there are a finite number, 2^n, of candidate $\boldsymbol{y}$ vectors to be considered. Since a Poisson-distributed response variable can take an infinite number of values, exact Poisson regression should evaluate an infinite number of $\boldsymbol{y}$ vectors. However, by identifying the maximum value of y_i to check, S_i, for each observation i, the number of candidate $\boldsymbol{y}$ vectors to check is reduced to $\prod_{i=1}^{n} S_i$. On a practical level, as S_i becomes large the probability of the Poisson random variable achieving this value drops to zero, so S_i can be thought of as the point at which the value does not matter. You can provide these maxima by specifying either an OFFSET= variable, o_i, or an EXACTMAX= variable, e_i, or you can let the algorithm choose a maximum for you. The way these two options interact to provide a maximum is described in the following list:

1. If an EXACTMAX= variable is specified, then $S_i = e_i$.
2. If the EXACTMAX option is specified without a variable, or if neither the EXACTMAX= nor OFFSET= options are specified, then you must also condition out the intercept or you must specify the STRATA statement. If you are conditioning out the intercept, then every S_i has an effective maximum of $\sum_{i=1}^{n} f_i y_{0i}$, where $\boldsymbol{y}_0$ is the observed response and f_i is the frequency of the observation; this is the sufficient statistic for the intercept term. If you are performing a stratified analysis, these sums are computed within each stratum.

3. If an offset variable is specified and the EXACTMAX option is not specified, then $S_i = \max(\text{ceil}(\exp(o_i)), 1)$. For example, if you have N_i rats in cage i and you are modeling the proportion that acquire a disease, then you would set your offsets to $o_i = \log(N_i)$ so that $S_i = N_i$. In this case, the offsets must also satisfy $S_i \geq y_i$.

OUTDIST= Output Data Set

The OUTDIST= data set contains every exact conditional distribution necessary to process the corresponding EXACT statement. For example, the following statements create one distribution for the x1 parameter and another for the x2 parameters, and produce the data set dist shown in Table 51.13:

```
data test;
   input y x1 x2 count;
   datalines;
0 0 0 1
1 0 0 1
0 1 1 2
1 1 1 1
1 0 2 3
1 1 2 1
1 2 0 3
1 2 1 2
1 2 2 1
;

proc genmod data=test exactonly;
   class x2 / param=ref;
   model y=x1 x2 / d=b;
   exact x1 x2/ outdist=dist;
run;
proc print data=dist;
run;
```

Table 51.13 OUTDIST= Data Set

Obs	x1	x20	x21	Count	Score	Prob
1	.	0	0	3	5.81151	0.03333
2	.	0	1	15	1.66031	0.16667
3	.	0	2	9	3.12728	0.10000
4	.	1	0	15	1.46523	0.16667
5	.	1	1	18	0.21675	0.20000
6	.	1	2	6	4.58644	0.06667
7	.	2	0	19	1.61869	0.21111
8	.	2	1	2	3.27293	0.02222
9	.	3	0	3	6.27189	0.03333
10	2	.	.	6	3.03030	0.12000
11	3	.	.	12	0.75758	0.24000
12	4	.	.	11	0.00000	0.22000
13	5	.	.	18	0.75758	0.36000
14	6	.	.	3	3.03030	0.06000

The first nine observations in the dist data set contain an exact distribution for the parameters of the x2 effect (hence the values for the x1 parameter are missing), and the remaining five observations are for the x1 parameter. If a joint distribution was created, there would be observations with values for both the x1 and x2 parameters. For CLASS variables, the corresponding parameters in the dist data set are identified by concatenating the variable name with the appropriate classification level.

The data set contains the possible sufficient statistics of the parameters for the effects specified in the EXACT statement, and the Count variable contains the number of different responses that yield these statistics. In particular, there are six possible response vectors $\mathbf{y}$ for which the dot product $\mathbf{y}'\mathbf{x1}$ was equal to 2, and for which $\mathbf{y}'\mathbf{x20}$, $\mathbf{y}'\mathbf{x21}$, and $\mathbf{y}'\mathbf{1}$ were equal to their actual observed values (displayed in the "Sufficient Statistics" table).

NOTE: If you are performing an exact Poisson analysis, then the Count variable is replaced by a variable named Weight.

When hypothesis tests are performed on the parameters, the Prob variable contains the probability of obtaining that statistic (which is just the count divided by the total count), and the Score variable contains the score for that statistic.

The OUTDIST= data set can contain a different exact conditional distribution for each specified EXACT statement. For example, consider the following EXACT statements:

```
exact 'O1'   x1    /           outdist=o1;
exact 'OJ12' x1 x2 / jointonly outdist=oj12;
exact 'OA12' x1 x2 / joint     outdist=oa12;
exact 'OE12' x1 x2 / estimate  outdist=oe12;
```

The O1 statement outputs a single exact conditional distribution. The OJ12 statement outputs only the joint distribution for x1 and x2. The OA12 statement outputs three conditional distributions: one for x1, one for x2, and one jointly for x1 and x2. The OE12 statement outputs two conditional distributions: one for x1 and the

other for x2. Data set oe12 contains both the x1 and x2 variables; the distribution for x1 has missing values in the x2 column while the distribution for x2 has missing values in the x1 column.

Response Level Ordering

For binary response data, PROC GENMOD fits the following model by default:

$$\Phi^{-1}\left(\frac{p-C}{1-C}\right) = \mathbf{x}'\boldsymbol{\beta}$$

where p is the probability of the response level identified as the first level in the "Response Profile" table in the output and Φ is the normal cumulative distribution function. By default, the covariate vector $\mathbf{x}$ contains an intercept term. This is sometimes called Abbot's formula.

Because of the symmetry of the normal (and logistic) distribution, the effect of reversing the order of the two response values is to change the signs of $\boldsymbol{\beta}$ in the preceding equation.

By default, response levels appear in ascending, sorted order (that is, the lowest level appears first, and then the next lowest, and so on). There are a number of ways that you can control the sort order of the response categories and, therefore, which level is assigned the first ordered level. One of the most common sets of response levels is {0,1}, with 1 representing the event with the probability that is to be modeled.

Consider the example where Y takes the values 1 and 0 for event and nonevent, respectively, and EXPOSURE is the explanatory variable. By default, PROC GENMOD assigns the first ordered level to response level 0, causing the probability of the nonevent to be modeled. There are several ways to change this.

Besides recoding the variable Y, you can do the following:

- Explicitly state which response level is to be modeled by using the response variable option EVENT= in the MODEL statement:

```
model Y(event='1') = Exposure;
```

- Specify the nonevent category for the response variable in the response variable option REF= in the MODEL statement:

```
model Y(ref='0') = Exposure;
```

- Specify the response variable option DESCENDING in the MODEL statement to assign the lowest ordered value to Y=1:

```
model Y(descending)=Exposure;
```

- Assign a format to Y such that the first formatted value (when the formatted values are put in sorted order) corresponds to the event. For the following example, Y=0 could be assigned formatted value 'nonevent' and Y=1 could be assigned formatted value 'event.' Since ORDER=FORMATTED by default, Y=1 becomes the first ordered level.

```
proc format;
   value disease 1='event' 0='nonevent';
run;
proc genmod;
   model y=exposure / dist=bin;
   format y disease.;
run;
```

- Arrange the input data set so that Y=1 appears first and use the ORDER=DATA option in the PROC GENMOD statement. Because ORDER=DATA sorts levels in order of their appearance in the data set, Y=1 becomes the first ordered level. Note that this option causes classification variables to be sorted by their order of appearance in the data set, also.

Missing Values

For generalized linear models, PROC GENMOD ignores any observation with a missing value for any variable involved in the model. You can score an observation in an output data set by setting only the response value to missing. For models fit with generalized estimating equations (GEEs), observations with missing values within a cluster are not used, and all available pairs are used in estimating the working correlation matrix. Clusters with fewer observations than the full cluster size are treated as having missing observations occurring at the end of the cluster. You can specify the order of missing observations with the WITHINSUBJECT= option. See the section "Missing Data" on page 3681 for more information about missing values in GEEs.

Displayed Output for Classical Analysis

The following output is produced by the GENMOD procedure. Note that some of the tables are optional and appear only in conjunction with the REPEATED statement and its options or with options in the MODEL statement. For details, see the section "ODS Table Names" on page 3714.

Model Information

The "Model Information" table displays the two-level data set name, the response distribution, the link function, the response variable name, the offset variable name, the frequency variable name, the scale weight variable name, the number of observations used, the number of events if events/trials format is used for response, the number of trials if events/trials format is used for response, the sum of frequency weights, the number of missing values in data set, and the number of invalid observations (for example, negative or 0 response values with gamma distribution or number of observations with events greater than trials with binomial distribution).

Class Level Information

If you use classification variables in the model, PROC GENMOD displays the levels of classification variables specified in the CLASS statement and in the MODEL statement. The levels are displayed in the same sorted order used to generate columns in the design matrix.

Response Profile

If you specify an ordinal model for the multinomial distribution, a table titled "Response Profile" is displayed containing the ordered values of the response variable and the number of occurrences of the values used in the model.

Iteration History for Parameter Estimates

If you specify the ITPRINT model option, PROC GENMOD displays a table containing the following for each iteration in the Newton-Raphson procedure for model fitting: the iteration number, the ridge value, the log likelihood, and values of all parameters in the model.

Criteria for Assessing Goodness of Fit

In the "Criteria for Assessing Goodness of Fit" table, PROC GENMOD displays the degrees of freedom for deviance and Pearson's chi-square, equal to the number of observations minus the number of regression parameters estimated, the deviance, the deviance divided by degrees of freedom, the scaled deviance, the scaled deviance divided by degrees of freedom, Pearson's chi-square, Pearson's chi-square divided by degrees of freedom, the scaled Pearson's chi-square, the scaled Pearson's chi-square divided by degrees of freedom, the log likelihood (excludes factorial terms) the full log likelihood, the Akaike information criterion, the corrected Akaike information criterion, and the Bayesian information criterion. The information in this table is valid only for maximum likelihood model fitting, and the table is not printed if the REPEATED statement is specified.

Last Evaluation of the Gradient

If you specify the model option ITPRINT, the GENMOD procedure displays the last evaluation of the gradient vector.

Last Evaluation of the Hessian

If you specify the model option ITPRINT, the GENMOD procedure displays the last evaluation of the Hessian matrix.

Analysis of (Initial) Parameter Estimates

The "Analysis of (Initial) Parameter Estimates" table contains the results from fitting a generalized linear model to the data. If you specify the REPEATED statement, these GLM parameter estimates are used as initial values for the GEE solution, and are displayed only if the PRINTMLE option in the REPEATED statement is specified. For each parameter in the model, PROC GENMOD displays the parameter name, as follows:

- the variable name for continuous regression variables

- the variable name and level for classification variables and interactions involving classification variables
- SCALE for the scale variable related to the dispersion parameter

In addition, PROC GENMOD displays the degrees of freedom for the parameter, the estimate value, the standard error, the Wald chi-square value, the *p*-value based on the chi-square distribution, and the confidence limits (Wald or profile likelihood) for parameters.

Lagrange Multiplier Statistics

If you specify that either the model intercept or the scale parameter is fixed, for those distributions that have a distribution scale parameter, the GENMOD procedure displays a table of Lagrange multiplier, or score, statistics for testing the validity of the constrained parameter that contains the test statistic, and the *p*-value.

Estimated Covariance Matrix

If you specify the model option COVB, the GENMOD procedure displays the estimated covariance matrix, defined as the inverse of the information matrix at the final iteration. This is based on the expected information matrix if the EXPECTED option is specified in the MODEL statement. Otherwise, it is based on the Hessian matrix used at the final iteration. This is, by default, the observed Hessian unless altered by the SCORING option in the MODEL statement.

Estimated Correlation Matrix

If you specify the CORRB model option, PROC GENMOD displays the estimated correlation matrix. This is based on the expected information matrix if the EXPECTED option is specified in the MODEL statement. Otherwise, it is based on the Hessian matrix used at the final iteration. This is, by default, the observed Hessian unless altered by the SCORING option in the MODEL statement.

Iteration History for LR Confidence Intervals

If you specify the ITPRINT and LRCI model options, PROC GENMOD displays an iteration history table for profile likelihood-based confidence intervals. For each parameter in the model, PROC GENMOD displays the parameter identification number, the iteration number, the log-likelihood value, parameter values.

Likelihood Ratio-Based Confidence Intervals for Parameters

If you specify the LRCI and the ITPRINT options in the MODEL statement, a table is displayed that summarizes profile likelihood-based confidence intervals for all parameters. For each parameter in the model, the table displays the confidence coefficient, the parameter identification number, lower and upper endpoints of confidence intervals for the parameter, and values of all other parameters at the solution.

LR Statistics for Type 1 Analysis

If you specify the TYPE1 model option, a table is displayed that contains the name of the effect, the deviance for the model including the effect and all previous effects, the degrees of freedom for the effect, the likelihood ratio statistic for testing the significance of the effect, and the *p*-value computed from the chi-square distribution with the effect's degrees of freedom.

If you specify either the SCALE=DEVIANCE or SCALE=PEARSON option in the MODEL statement, columns are displayed that contain the name of the effect, the deviance for the model including the effect and all previous effects, the numerator degrees of freedom, the denominator degrees of freedom, the chi-square statistic for testing the significance of the effect, the p-value computed from the chi-square distribution with numerator degrees of freedom, the F statistic for testing the significance of the effect, and the p-value based on the F distribution.

Iteration History for Type 3 Contrasts

If you specify the model options ITPRINT and TYPE3, an iteration history table is displayed for fitting the model with Type 3 contrast constraints for each effect that contains the effect name, the iteration number, the ridge value, the log likelihood, and values of all parameters.

LR Statistics for Type 3 Analysis

If you specify the TYPE3 model option, a table is displayed that contains, for each effect in the model, the name of the effect, the likelihood ratio statistic for testing the significance of the effect, the degrees of freedom for the effect, and the p-value computed from the chi-square distribution.

If you specify either the SCALE=DEVIANCE or SCALE=PEARSON option in the MODEL statement, columns are displayed that contain the name of the effect, the likelihood ratio statistic for testing the significance of the effect, the F statistic for testing the significance of the effect, the numerator degrees of freedom, the denominator degrees of freedom, the p-value based on the F distribution, and the p-value computed from the chi-square distribution with the numerator's degrees of freedom.

Wald Statistics for Type 3 Analysis

If you specify the TYPE3 and WALD model options, a table is displayed that contains the name of the effect, the degrees of freedom of the effect, the Wald statistic for testing the significance of the effect, and the p-value computed from the chi-square distribution.

Parameter Information

If you specify the ITPRINT, COVB, CORRB, WALDCI, or LRCI option in the MODEL statement, or if you specify a CONTRAST statement, a table is displayed that identifies parameters with numbers, rather than names, for use in tables and matrices where a compact identifier for parameters is helpful. For each parameter, the table contains an index number that identifies the parameter, and the parameter name, including level information for effects containing classification variables.

Observation Statistics

If you specify the OBSTATS option in the MODEL statement, PROC GENMOD displays a table containing miscellaneous statistics. Residuals and case deletion diagnostic statistics are not available for the multinomial distribution. Case deletion diagnostics are not available for zero-inflated models.

For each observation in the input data set, the following are displayed:

- the value of the response variable
- the predicted value of the mean

- the value of the linear predictor The value of an OFFSET variable is added to the linear predictor.
- the estimated standard error of the linear predictor
- the value of the negative of the weight in the Hessian matrix at the final iteration. This is the expected weight if the EXPECTED option is specified in the MODEL statement. Otherwise, it is the weight used in the final iteration. That is, it is the observed weight unless the SCORING= option has been specified.
- approximate lower and upper endpoints for a confidence interval for the predicted value of the mean
- raw residual
- Pearson residual
- deviance residual
- standardized Pearson residual
- standardized deviance residual
- likelihood residual
- leverage
- Cook's distance statistic
- DFBETA statistic, for each parameter
- standardized DFBETA statistic, for each parameter
- zero-inflation probability for zero-inflated models
- response mean for zero-inflated models

ESTIMATE Statement Results

If you specify a REPEATED statement, the ESTIMATE statement results apply to the specified GEE model. Otherwise, they apply to the specified generalized linear model.

For each ESTIMATE statement, the table contains the contrast label, the estimated value of the contrast, the standard error of the estimate, the significance level α, $(1-\alpha) \times 100\%$ confidence intervals for contrast, the Wald chi-square statistic for the contrast, and the p-value computed from the chi-square distribution. The mean of the contrast, defined as the inverse link function applied to the contrast, and $(1-\alpha) \times 100\%$ confidence intervals for the mean are also displayed.

If you specify the EXP option, an additional row is displayed with statistics for the exponentiated value of the contrast.

CONTRAST Coefficients

If you specify the CONTRAST or ESTIMATE statement and you specify the E option, a table titled "Coefficients For Contrast *label*" is displayed, where *label* is the label specified in the CONTRAST statement. The table contains the contrast label, and the rows of the contrast matrix.

Iteration History for Contrasts

If you specify the ITPRINT option, an iteration history table is displayed for fitting the model with contrast constraints for each effect. The table contains the contrast label, the iteration number, the ridge value, the log likelihood, and values of all parameters.

CONTRAST Statement Results

If you specify a REPEATED statement, the CONTRAST statement results apply to the specified GEE model. Otherwise, they apply to the specified generalized linear model.

A table is displayed that contains the contrast label, the degrees of freedom for the contrast, and the likelihood ratio, score, or Wald statistic for testing the significance of the contrast. Score statistics are used in GEE models, likelihood ratio statistics are used in generalized linear models, and Wald statistics are used in both. Also displayed are the p-value computed from the chi-square distribution, and the type of statistic computed for this contrast: Wald, LR, or score.

If you specify either the SCALE=DEVIANCE or SCALE=PEARSON option for generalized linear models, columns are displayed that contain the contrast label, the likelihood ratio statistic for testing the significance of the contrast, the F statistic for testing the significance of the contrast, the numerator degrees of freedom, the denominator degrees of freedom, the p-value based on the F distribution, and the p-value computed from the chi-square distribution with numerator degrees of freedom.

LSMEANS Coefficients

If you specify the LSMEANS statement and you specify the E option, the "Coefficients for *effect* Least Squares Means" table is displayed, where *effect* is the effect specified in the LSMEANS statement. The table contains the effect names and the rows of least squares means coefficients.

Least Squares Means

If you specify the LSMEANS statement, the "Least Squares Means" table is displayed. The table contains for each effect the following: the effect name, and for each level of each effect the following:

- the least squares mean estimate
- standard error
- chi-square value
- p-value computed from the chi-square distribution

If you specify the DIFF option, a table titled "Differences of Least Squares Means" is displayed containing corresponding statistics for the differences between the least squares means for the levels of each effect.

GEE Model Information

If you specify the REPEATED statement, the "GEE Model Information" table displays the correlation structure of the working correlation matrix or the log odds ratio structure, the within-subject effect, the subject effect, the number of clusters, the correlation matrix dimension, and the minimum and maximum cluster size.

Log Odds Ratio Parameter Information

If you specify the REPEATED statement and specify a log odds ratio model for binary data with the LOGOR= option, then the "Log Odds Ratio Parameter Information" table is displayed showing the correspondence between data pairs and log odds ratio model parameters.

Iteration History for GEE Parameter Estimates

If you specify the REPEATED statement and the MODEL statement option ITPRINT, the "Iteration History For GEE Parameter Estimates" table is displayed. The table contains the parameter identification number, the iteration number, and values of all parameters.

Last Evaluation of the Generalized Gradient and Hessian

If you specify the REPEATED statement and select ITPRINT as a model option, PROC GENMOD displays the "Last Evaluation Of The Generalized Gradient And Hessian" table.

GEE Parameter Estimate Covariance Matrices

If you specify the REPEATED statement and the COVB option, PROC GENMOD displays the "Covariance Matrix (Model-Based)" and "Covariance Matrix (Empirical)" tables.

GEE Parameter Estimate Correlation Matrices

If you specify the REPEATED statement and the CORRB option, PROC GENMOD displays the "Correlation Matrix (Model-Based)" and "Correlation Matrix (Empirical)" tables.

GEE Working Correlation Matrix

If you specify the REPEATED statement and the CORRW option, PROC GENMOD displays the "Working Correlation Matrix" table.

GEE Fit Criteria

If you specify the REPEATED statement, PROC GENMOD displays the quasi-likelihood information criteria for model fit QIC and QIC_u in the "GEE Fit Criteria" table.

Analysis of GEE Parameter Estimates

If you specify the REPEATED statement, PROC GENMOD uses empirical standard error estimates to compute and display the "Analysis Of GEE Parameter Estimates Empirical Standard Error Estimates" table that contains the parameter names as follows:

- the variable name for continuous regression variables
- the variable name and level for classification variables and interactions involving classification variables
- "Scale" for the scale variable related to the dispersion parameter

In addition, the parameter estimate, the empirical standard error, a 95% confidence interval, and the Z score and p-value are displayed for each parameter.

If you specify the MODELSE option in the REPEATED statement, the "Analysis Of GEE Parameter Estimates Model-Based Standard Error Estimates" table based on model-based standard errors is also produced.

GEE Observation Statistics

If you specify the OBSTATS option in the REPEATED statement, PROC GENMOD displays a table containing miscellaneous statistics. For each observation in the input data set, the following are displayed:

- the value of the response variable and all other variables in the model, denoted by the variable names
- the predicted value of the mean
- the value of the linear predictor
- the standard error of the linear predictor
- confidence limits for the predicted values
- raw residual
- Pearson residual
- cluster number
- leverage
- cluster leverage
- cluster Cook's distance statistic
- studentized cluster Cook's distance statistic
- individual observation Cook's distance statistic
- cluster DFBETA statistic for each parameter
- cluster standardized DFBETA statistic for each parameter
- individual observation DFBETA statistic for each parameter
- individual observation standardized DFBETA statistic for each parameter

Displayed Output for Bayesian Analysis

If a Bayesian analysis is requested with a BAYES statement, the displayed output includes the following.

Model Information

The "Model Information" table displays the two-level data set name, the number of burn-in iterations, the number of iterations after the burn-in, the number of thinning iterations, the response distribution, the link function, the response variable name, the offset variable name, the frequency variable name, the scale weight variable name, the number of observations used, the number of events if events/trials format is used for response, the number of trials if events/trials format is used for response, the sum of frequency weights, the number of missing values in data set, and the number of invalid observations (for example, negative or 0 response values with gamma distribution or number of observations with events greater than trials with binomial distribution).

Class Level Information

The "Class Level Information" table displays the levels of classification variables if you specify a CLASS statement.

Maximum Likelihood Estimates

The "Analysis of Maximum Likelihood Parameter Estimates" table displays the maximum likelihood estimate of each parameter, the estimated standard error of the parameter estimator, and confidence limits for each parameter.

Coefficient Prior

The "Coefficient Prior" table displays the prior distribution of the regression coefficients.

Independent Prior Distributions for Model Parameters

The "Independent Prior Distributions for Model Parameters" table displays the prior distributions of additional model parameters (scale, exponential scale, Weibull scale, Weibull shape, gamma shape).

Initial Values and Seeds

The "Initial Values and Seeds" table displays the initial values and random number generator seeds for the Gibbs chains.

Fit Statistics

The "Fit Statistics" table displays the deviance information criterion (DIC) and the effective number of parameters.

Descriptive Statistics of the Posterior Samples

The "Descriptive Statistics of the Posterior Sample" table contains the size of the sample, the mean, the standard deviation, and the quartiles for each model parameter.

Interval Estimates for Posterior Sample

The "Interval Estimates for Posterior Sample" table contains the HPD intervals and the credible intervals for each model parameter.

Correlation Matrix of the Posterior Samples

The "Correlation Matrix of the Posterior Samples" table is produced if you include the CORR suboption in the SUMMARY= option in the BAYES statement. This table displays the sample correlation of the posterior samples.

Covariance Matrix of the Posterior Samples

The "Covariance Matrix of the Posterior Samples" table is produced if you include the COV suboption in the SUMMARY= option in the BAYES statement. This table displays the sample covariance of the posterior samples.

Autocorrelations of the Posterior Samples

The "Autocorrelations of the Posterior Samples" table displays the lag1, lag5, lag10, and lag50 autocorrelations for each parameter.

Gelman and Rubin Diagnostics

The "Gelman and Rubin Diagnostics" table is produced if you include the GELMAN suboption in the DIAGNOSTIC= option in the BAYES statement. This table displays the estimate of the potential scale reduction factor and its 97.5% upper confidence limit for each parameter.

Geweke Diagnostics

The "Geweke Diagnostics" table displays the Geweke statistic and its p-value for each parameter.

Raftery and Lewis Diagnostics

The "Raftery Diagnostics" tables is produced if you include the RAFTERY suboption in the DIAGNOSTIC= option in the BAYES statement. This table displays the Raftery and Lewis diagnostics for each variable.

Heidelberger and Welch Diagnostics

The "Heidelberger and Welch Diagnostics" table is displayed if you include the HEIDELBERGER suboption in the DIAGNOSTIC= option in the BAYES statement. This table shows the results of a stationary test and a halfwidth test for each parameter.

Effective Sample Size

The "Effective Sample Size" table displays, for each parameter, the effective sample size, the correlation time, and the efficiency.

Monte Carlo Standard Errors

The "Monte Carlo Standard Errors" table displays, for each parameter, the Monte Carlo standard error, the posterior sample standard deviation, and the ratio of the two.

Displayed Output for Exact Analysis

If an exact analysis is requested with an EXACT statement, the displayed output includes the following tables. If the METHOD=NETWORKMC option is specified, the test and estimate tables are renamed "Monte Carlo" tables and a Monte Carlo standard error column ($\sqrt{p(1-p)/n}$) is displayed.

Sufficient Statistics

Displays if you request an OUTDIST= data set in an EXACT statement. The table lists the parameters and their observed sufficient statistics.

(Monte Carlo) Conditional Exact Tests

This table tests the hypotheses that the parameters of interest are insignificant. See the section "Exact Logistic and Exact Poisson Regression" on page 3699 for details.

(Monte Carlo) Exact Parameter Estimates

Displays if you specify the ESTIMATE option in the EXACT statement. This table gives individual parameter estimates for each variable (conditional on the values of all the other parameters in the model), confidence limits, and a two-sided p-value (twice the one-sided p-value) for testing that the parameter is zero. See the section "Exact Logistic and Exact Poisson Regression" on page 3699 for details.

(Monte Carlo) Exact Odds Ratios

Displays if you specify the ESTIMATE=ODDS or ESTIMATE=BOTH option in the EXACT statement. See the section "Exact Logistic and Exact Poisson Regression" on page 3699 for details.

Strata Summary

Displays if a STRATA statement is also specified. Shows the pattern of the number of events and the number of nonevents, or of the number of observations, in a stratum. See the section "STRATA Statement" on page 3652 for more information.

Strata Information

Displays if a STRATA statement is specified with the INFO option.

ODS Table Names

PROC GENMOD assigns a name to each table that it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed separately in Table 51.14 for a maximum likelihood analysis, in Table 51.15 for a Bayesian analysis, and in Table 51.16 for an Exact analysis. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 51.14 ODS Tables Produced in PROC GENMOD for a Classical Analysis

ODS Table Name	Description	Statements	Option
AssessmentSummary	Model assessment summary	ASSESS	Default
ClassLevels	Classification variable levels	CLASS	Default
Contrasts	Tests of contrasts	CONTRAST	Default
ContrastCoef	Contrast coefficients	CONTRAST	E
ConvergenceStatus	Convergence status	MODEL	Default
CorrB	Parameter estimate correlation matrix	MODEL	CORRB
CovB	Parameter estimate covariance matrix	MODEL	COVB
Estimates	Estimates of contrasts	ESTIMATE	Default
EstimateCoef	Contrast coefficients	ESTIMATE	E
GEEEmpPEst	GEE parameter estimates with empirical standard errors	REPEATED	Default
GEEExchCorr	GEE exchangeable working correlation value	REPEATED	TYPE=EXCH
GEEFitCriteria	GEE QIC fit criteria	REPEATED	Default
GEELogORInfo	GEE log odds ratio model information	REPEATED	LOGOR=
GEEModInfo	GEE model information	REPEATED	Default
GEEModPEst	GEE parameter estimates with model-based standard errors	REPEATED	MODELSE
GEENCorr	GEE model-based correlation matrix	REPEATED	MCORRB
GEENCov	GEE model-based covariance matrix	REPEATED	MCOVB
GEERCorr	GEE empirical correlation matrix	REPEATED	ECORRB
GEERCov	GEE empirical covariance matrix	REPEATED	ECOVB

Table 51.14 *continued*

ODS Table Name	Description	Statements	Option
GEEWCorr	GEE working correlation matrix	REPEATED	CORRW
IterContrasts	Iteration history for contrasts	MODEL CONTRAST	ITPRINT
IterLRCI	Iteration history for likelihood ratio confidence intervals	MODEL	LRCI ITPRINT
IterParms	Iteration history for parameter estimates	MODEL	ITPRINT
IterParmsGEE	Iteration history for GEE parameter estimates	MODEL REPEATED	ITPRINT
IterType3	Iteration history for Type 3 statistics	MODEL	TYPE3 ITPRINT
LRCI	Likelihood ratio confidence intervals	MODEL	LRCI ITPRINT
Coef	Coefficients for least squares means	LSMEANS	E
Diffs	Least squares means differences	LSMEANS	DIFF
LSMeans	Least squares means	LSMEANS	Default
LagrangeStatistics	Lagrange statistics	MODEL	NOINT | NOSCALE
LastGEEGrad	Last evaluation of the generalized gradient and Hessian	MODEL REPEATED	ITPRINT
LastGradHess	Last evaluation of the gradient and Hessian	MODEL	ITPRINT
LinDep	Linearly dependent rows of contrasts	CONTRAST	Default
Margins	Predictive margins	MARGINS	Default
MarginsDiffs	Differences of predictive margins	MARGINS	DIFF
MarginsSliceDiffs	Differences of sliced predictive margins	MARGINS	SLICEBY= or SLICEDIFF
MarginsSliceTests	Tests of sliced predictive margins	MARGINS	SLICEBY=
ModelANOVA	Type 3 tests or joint tests	MODEL	TYPE3 without REPEATED | ZEROMODEL
ModelInfo	Model information	MODEL	Default
Modelfit	Goodness-of-fit statistics	MODEL	Default without REPEATED
NObs	Number of observations summary		Default
NonEst	Nonestimable rows of contrasts	CONTRAST	Default

Table 51.14 *continued*

ODS Table Name	Description	Statements	Option
ObStats	Observation-wise statistics	MODEL	OBSTATS | CL | PREDICTED | RESIDUALS | XVARS
ParameterEstimates	Parameter estimates	MODEL	Default without REPEATED | PRINTMLE with REPEATED
ParmInfo	Parameter indices	MODEL	Default
ResponseProfile	Frequency counts for multinomial and binary models	MODEL	DIST=MULTINOMIAL | DIST=BINOMIAL
Type1	Type 1 tests	MODEL	TYPE1
Type3	Type 3 tests or joint tests for GEE model	MODEL REPEATED	TYPE3
Type3Zero	Type 3 tests or joint tests for zero-inflated model	MODEL ZEROMODEL	TYPE3
ZeroParameterEstimates	Parameter estimates for zero-inflated model	ZEROMODEL	Default

Table 51.15 ODS Tables Produced in PROC GENMOD for a Bayesian Analysis

ODS Table Name	Description	Statement	Option
AutoCorr	Autocorrelations of the posterior samples	BAYES	Default
ClassLevels	Classification variable levels	CLASS	Default
CoeffPrior	Prior distribution of the regression coefficients	BAYES	Default
ConvergenceStatus	Convergence status of maximum likelihood estimation	MODEL	Default
Corr	Correlation matrix of the posterior samples	BAYES	SUMMARY=CORR
ESS	Effective sample size	BAYES	Default
FitStatistics	Fit statistics	BAYES	Default
Gelman	Gelman and Rubin convergence diagnostics	BAYES	DIAG=GELMAN
Geweke	Geweke convergence diagnostics	BAYES	Default
Heidelberger	Heidelberger and Welch convergence diagnostics	BAYES	DIAG=HEIDELBERGER
InitialValues	Initial values of the Markov chains	BAYES	Default
IterParms	Iteration history for parameter estimates	MODEL	ITPRINT

Table 51.15 *continued*

ODS Table Name	Description	Statement	Option
LastGradHess	Last evaluation of the gradient and Hessian for maximum likelihood estimation	MODEL	ITPRINT
MCSE	Monte Carlo standard errors	BAYES	DIAG=MCSE
ModelInfo	Model information	PROC	Default
NObs	Number of observations		Default
ParameterEstimates	Maximum likelihood estimates of model parameters	MODEL	Default
ParmInfo	Parameter indices	MODEL	Default
ParmPrior	Prior distribution for scale and shape	BAYES	Default
PostIntervals	HPD and equal-tail intervals of the posterior samples	BAYES	Default
PosteriorSample	Posterior samples (for ODS output data set only)	BAYES	
PostSummaries	Summary statistics of the posterior samples	BAYES	Default
Raftery	Raftery and Lewis convergence diagnostics	BAYES	DIAG=RAFTERY

Table 51.16 ODS Tables Produced in PROC GENMOD for an Exact Analysis

ODS Table Name	Description	Statement	Option
ExactParmEst	Parameter estimates	EXACT	ESTIMATE, ESTIMATE=PARM, ESTIMATE=BOTH
ExactTests	Conditional exact tests	EXACT	Default
ExpExactParmEst	Exact odds ratios	EXACT	ESTIMATE=ODDS, ESTIMATE=BOTH
NStrataIgnored	Number of uninformative strata	STRATA	Default
StrataSummary	Number of strata with specific response frequencies	STRATA	Default
StrataInfo	Event and nonevent frequencies for each stratum	STRATA	INFO
SuffStats	Sufficient statistics	EXACT	OUTDIST=

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

Some graphs are produced by default; other graphs are produced by using statements and options. You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC GENMOD generates are listed in Table 51.17, along with the required statements and options.

ODS Graph Names

PROC GENMOD assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 51.17.

To request these graphs, ODS Graphics must be enabled and you must specify the statement and options indicated in Table 51.17.

Table 51.17 Graphs Produced by PROC GENMOD

ODS Graph Name	Description	Statement	Option
ADPanel	Autocorrelation function and density panel	BAYES	PLOTS=(AUTOCORR DENSITY)
AutocorrPanel	Autocorrelation function panel	BAYES	PLOTS= AUTOCORR
AutocorrPlot	Autocorrelation function plot	BAYES	PLOTS(UNPACK)=AUTOCORR
ClusterCooksDPlot	Cluster Cook's *D* by cluster number	PROC	PLOTS=
ClusterDFFITPlot	Cluster DFFIT by cluster number	PROC	PLOTS=
ClusterLeveragePlot	Cluster leverage by cluster number	PROC	PLOTS=
CooksDPlot	Cook's distance	PROC	PLOTS=
CumResidPanel	Panel of aggregates of residuals	ASSESS	CRPANEL
CumulativeResiduals	Model assessment based on aggregates of residuals	ASSESS	Default
DevianceResidByXBeta	Deviance residuals by linear predictor	PROC	PLOTS=
DevianceResidualPlot	Deviance values	PROC	PLOTS=

Table 51.17 *continued*

ODS Table Name	Description	Statement	Option
DFBETAByCluster	Cluster DFBeta by cluster number	PROC	PLOTS=
DFBETAPlot	DFBeta	PROC	PLOTS=
DiagnosticPlot	Panel of residuals, influence, and diagnostic statistics	PROC MODEL REPEATED	PLOTS=
LeveragePlot	Leverage	PROC	PLOTS=
LikeResidByXBeta	Likelihood residuals by linear predictor	PROC	PLOTS=
LikeResidualPlot	Likelihood residuals	PROC	PLOTS=
PearsonResidByXBeta	Pearson residuals by linear predictor	PROC	PLOTS=
PearsonResidualPlot	Pearson residuals	PROC	PLOTS=
PredictedByObservation	Predicted values	PROC	PLOTS=
RawResidByXBeta	Raw residuals by linear predictor	PROC	PLOTS=
RawResidualPlot	Raw residuals	PROC	PLOTS=
StdDevianceResidByXBeta	Standardized deviance residuals by linear predictor	PROC	PLOTS=
StdDevianceResidualPlot	Standardized deviance residuals	PROC	PLOTS=
StdDFBETAByCluster	Standardized cluster DFBeta by cluster number	PROC	PLOTS=
StdDFBETAPlot	Standardized DFBeta	PROC	PLOTS=
StdPearsonResidByXBeta	Standardized Pearson residuals by linear predictor	PROC	PLOTS=
StdPearsonResidualPlot	Standardized Pearson residuals	PROC	PLOTS=
TAPanel	Trace and autocorrelation function panel	BAYES	PLOTS=(TRACE AUTOCORR)
TADPanel	Trace, autocorrelation, and density function panel	BAYES	Default
TDPanel	Trace and density panel	BAYES	PLOTS=(TRACE DENSITY)
TracePanel	Trace panel	BAYES	PLOTS=TRACE
TracePlot	Trace plot	BAYES	PLOTS(UNPACK)=TRACE
ZeroInflationProbPlot	Zero-inflation probabilities	PROC	PLOTS=

Examples: GENMOD Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The following examples illustrate some of the capabilities of the GENMOD procedure. These are not intended to represent definitive analyses of the data sets presented here. You should refer to the texts cited in the references for guidance on complete analysis of data by using generalized linear models.

Example 51.1: Logistic Regression

(View the complete code for this example (genmex1.sas) in the example repository.)

In an experiment comparing the effects of five different drugs, each drug is tested on a number of different subjects. The outcome of each experiment is the presence or absence of a positive response in a subject. The following artificial data represent the number of responses r in the n subjects for the five different drugs, labeled A through E. The response is measured for different levels of a continuous covariate x for each drug. The drug type and the continuous covariate x are explanatory variables in this experiment. The number of responses r is modeled as a binomial random variable for each combination of the explanatory variable values, with the binomial number of trials parameter equal to the number of subjects n and the binomial probability equal to the probability of a response.

The following DATA step creates the data set:

```
data drug;
   input drug$ x r n @@;
   datalines;
A  .1   1  10   A  .23  2  12   A  .67  1   9
B  .2   3  13   B  .3   4  15   B  .45  5  16   B  .78  5  13
C  .04  0  10   C  .15  0  11   C  .56  1  12   C  .7   2  12
D  .34  5  10   D  .6   5   9   D  .7   8  10
E  .2  12  20   E  .34 15  20   E  .56 13  15   E  .8  17  20
;
```

A logistic regression for these data is a generalized linear model with response equal to the binomial proportion r/n. The probability distribution is binomial, and the link function is logit. For these data, drug and x are explanatory variables. The probit and the complementary log-log link functions are also appropriate for binomial data.

PROC GENMOD performs a logistic regression on the data in the following SAS statements:

```
proc genmod data=drug;
   class drug;
   model r/n = x drug / dist = bin
                        link = logit
                        lrci;
run;
```

Since these data are binomial, you use the events/trials syntax to specify the response in the MODEL statement. Profile likelihood confidence intervals for the regression parameters are computed using the LRCI option. General model and data information is produced in Output 51.1.1.

Output 51.1.1 Model Information

The GENMOD Procedure

Model Information	
Data Set	WORK.DRUG
Distribution	Binomial
Link Function	Logit
Response Variable (Events)	r
Response Variable (Trials)	n

The five levels of the CLASS variable DRUG are displayed in Output 51.1.2.

Output 51.1.2 CLASS Variable Levels

Class Level Information		
Class	**Levels**	**Values**
drug	5	A B C D E

In the "Criteria For Assessing Goodness Of Fit" table displayed in Output 51.1.3, the value of the deviance divided by its degrees of freedom is less than 1. A p-value is not computed for the deviance; however, a deviance that is approximately equal to its degrees of freedom is a possible indication of a good model fit. Asymptotic distribution theory applies to binomial data as the number of binomial trials parameter n becomes large for each combination of explanatory variables. McCullagh and Nelder (1989) caution against the use of the deviance alone to assess model fit. The model fit for each observation should be assessed by examination of residuals. The OBSTATS option in the MODEL statement produces a table of residuals and other useful statistics for each observation.

Output 51.1.3 Goodness-of-Fit Criteria

Criteria For Assessing Goodness Of Fit			
Criterion	**DF**	**Value**	**Value/DF**
Deviance	12	5.2751	0.4396
Scaled Deviance	12	5.2751	0.4396
Pearson Chi-Square	12	4.5133	0.3761
Scaled Pearson X2	12	4.5133	0.3761
Log Likelihood		-114.7732	
Full Log Likelihood		-23.7343	
AIC (smaller is better)		59.4686	
AICC (smaller is better)		67.1050	
BIC (smaller is better)		64.8109	

In the "Analysis Of Parameter Estimates" table displayed in Output 51.1.4, chi-square values for the explanatory variables indicate that the parameter values other than the intercept term are all significant. The scale parameter is set to 1 for the binomial distribution. When you perform an overdispersion analysis, the value of the overdispersion parameter is indicated here. See the section "Overdispersion" on page 3665 for a discussion of overdispersion.

Output 51.1.4 Parameter Estimates

Analysis Of Maximum Likelihood Parameter Estimates								
Parameter		DF	Estimate	Standard Error	Likelihood Ratio 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		1	0.2792	0.4196	-0.5336	1.1190	0.44	0.5057
x		1	1.9794	0.7660	0.5038	3.5206	6.68	0.0098
drug	A	1	-2.8955	0.6092	-4.2280	-1.7909	22.59	<.0001
drug	B	1	-2.0162	0.4052	-2.8375	-1.2435	24.76	<.0001
drug	C	1	-3.7952	0.6655	-5.3111	-2.6261	32.53	<.0001
drug	D	1	-0.8548	0.4838	-1.8072	0.1028	3.12	0.0773
drug	E	0	0.0000	0.0000	0.0000	0.0000	.	.
Scale		0	1.0000	0.0000	1.0000	1.0000		

Note: The scale parameter was held fixed.

The preceding table contains the profile likelihood confidence intervals for the explanatory variable parameters requested with the LRCI option. Wald confidence intervals are displayed by default. Profile likelihood confidence intervals are considered to be more accurate than Wald intervals (see Aitkin et al. (1989)), especially with small sample sizes. You can specify the confidence coefficient with the ALPHA= option in the MODEL statement. The default value of 0.05, corresponding to 95% confidence limits, is used here. See the section "Confidence Intervals for Parameters" on page 3670 for a discussion of profile likelihood confidence intervals.

Example 51.2: Normal Regression, Log Link

(View the complete code for this example (genmex2.sas) in the example repository.)

Consider the following data, where x is an explanatory variable and y is the response variable. It appears that y varies nonlinearly with x and that the variance is approximately constant. A normal distribution with a log link function is chosen to model these data; that is, $\log(\mu_i) = \mathbf{x}_i'\boldsymbol{\beta}$ so that $\mu_i = \exp(\mathbf{x}_i'\boldsymbol{\beta})$.

```
data nor;
   input x y;
   datalines;
0 5
0 7
0 9
1 7
1 10
1 8
2 11
2 9
3 16
3 13
3 14
4 25
4 24
5 34
5 32
```

```
5 30
;
```

The following SAS statements produce the analysis with the normal distribution and log link:

```
proc genmod data=nor;
   model y = x / dist = normal
                 link = log;
   output out       = Residuals
          pred      = Pred
          resraw    = Resraw
          reschi    = Reschi
          resdev    = Resdev
          stdreschi = Stdreschi
          stdresdev = Stdresdev
          reslik    = Reslik;
run;
```

The OUTPUT statement is specified to produce a data set that contains predicted values and residuals for each observation. This data set can be useful for further analysis, such as residual plotting.

The results from these statements are displayed in Output 51.2.1.

Output 51.2.1 Log-Linked Normal Regression

The GENMOD Procedure

Model Information	
Data Set	WORK.NOR
Distribution	Normal
Link Function	Log
Dependent Variable	y

Criteria For Assessing Goodness Of Fit			
Criterion	DF	Value	Value/DF
Deviance	14	52.3000	3.7357
Scaled Deviance	14	16.0000	1.1429
Pearson Chi-Square	14	52.3000	3.7357
Scaled Pearson X2	14	16.0000	1.1429
Log Likelihood		-32.1783	
Full Log Likelihood		-32.1783	
AIC (smaller is better)		70.3566	
AICC (smaller is better)		72.3566	
BIC (smaller is better)		72.6743	

Analysis Of Maximum Likelihood Parameter Estimates							
Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	1.7214	0.0894	1.5461	1.8966	370.76	<.0001
x	1	0.3496	0.0206	0.3091	0.3901	286.64	<.0001
Scale	1	1.8080	0.3196	1.2786	2.5566		

Note: The scale parameter was estimated by maximum likelihood.

The PROC GENMOD scale parameter, in the case of the normal distribution, is the standard deviation. By default, the scale parameter is estimated by maximum likelihood. You can specify a fixed standard deviation by using the NOSCALE and SCALE= options in the MODEL statement.

```
proc print data=Residuals;
run;
```

Output 51.2.2 Data Set of Predicted Values and Residuals

Obs	x	y	Pred	Reschi	Resraw	Resdev	Stdreschi	Stdresdev	Reslik
1	0	5	5.5921	-0.59212	-0.59212	-0.59212	-0.34036	-0.34036	-0.34036
2	0	7	5.5921	1.40788	1.40788	1.40788	0.80928	0.80928	0.80928
3	0	9	5.5921	3.40788	3.40788	3.40788	1.95892	1.95892	1.95892
4	1	7	7.9324	-0.93243	-0.93243	-0.93243	-0.54093	-0.54093	-0.54093
5	1	10	7.9324	2.06757	2.06757	2.06757	1.19947	1.19947	1.19947
6	1	8	7.9324	0.06757	0.06757	0.06757	0.03920	0.03920	0.03920
7	2	11	11.2522	-0.25217	-0.25217	-0.25217	-0.14686	-0.14686	-0.14686
8	2	9	11.2522	-2.25217	-2.25217	-2.25217	-1.31166	-1.31166	-1.31166
9	3	16	15.9612	0.03878	0.03878	0.03878	0.02249	0.02249	0.02249
10	3	13	15.9612	-2.96122	-2.96122	-2.96122	-1.71738	-1.71738	-1.71738
11	3	14	15.9612	-1.96122	-1.96122	-1.96122	-1.13743	-1.13743	-1.13743
12	4	25	22.6410	2.35897	2.35897	2.35897	1.37252	1.37252	1.37252
13	4	24	22.6410	1.35897	1.35897	1.35897	0.79069	0.79069	0.79069
14	5	34	32.1163	1.88366	1.88366	1.88366	1.22914	1.22914	1.22914
15	5	32	32.1163	-0.11634	-0.11634	-0.11634	-0.07592	-0.07592	-0.07592
16	5	30	32.1163	-2.11634	-2.11634	-2.11634	-1.38098	-1.38098	-1.38098

The data set of predicted values and residuals (Output 51.2.2) is created by the OUTPUT statement. You can use the PLOTS= option in the PROC GENMOD statement to create plots of predicted values and residuals. Note that raw, Pearson, and deviance residuals are equal in this example. This is a characteristic of the normal distribution and is not true in general for other distributions.

Example 51.3: Gamma Distribution Applied to Life Data

(View the complete code for this example (genmex3.sas) in the example repository.)

Life data are sometimes modeled with the gamma distribution. Although PROC GENMOD does not analyze censored data or provide other useful lifetime distributions such as the Weibull or lognormal, it can be used for modeling complete (uncensored) data with the gamma distribution, and it can provide a statistical test for the exponential distribution against other gamma distribution alternatives. See Lawless (2003) or Nelson (1982) for applications of the gamma distribution to life data.

The following data represent failure times of machine parts, some of which are manufactured by manufacturer A and some by manufacturer B.

```
data A;
   input lifetime @@;
   mfg = 'A';
   datalines;
620  470  260  89   388  242
```

```
103  100  39   460  284  1285
218  393  106  158  152  477
403  103  69   158  818  947
399  1274 32   12   134  660
548  381  203  871  193  531
317  85   1410 250  41   1101
32   421  32   343  376  1512
1792 47   95   76   515  72
1585 253  6    860  89   1055
537  101  385  176  11   565
164  16   1267 352  160  195
1279 356  751  500  803  560
151  24   689  1119 1733 2194
763  555  14   45   776  1
;

data B;
   input lifetime @@;
   mfg = 'B';
   datalines;
1747 945  12   1453 14   150
20   41   35   69   195  89
1090 1868 294  96   618  44
142  892  1307 310  230  30
403  860  23   406  1054 1935
561  348  130  13   230  250
317  304  79   1793 536  12
9    256  201  733  510  660
122  27   273  1231 182  289
667  761  1096 43   44   87
405  998  1409 61   278  407
113  25   940  28   848  41
646  575  219  303  304  38
195  1061 174  377  388  10
246  323  198  234  39   308
55   729  813  1216 1618 539
6    1566 459  946  764  794
35   181  147  116  141  19
380  609  546
;

data lifdat;
   set A B;
run;
```

The following SAS statements use PROC GENMOD to compute Type 3 statistics to test for differences between the two manufacturers in machine part life. Type 3 statistics are identical to Type 1 statistics in this case, since there is only one effect in the model. The log link function is selected to ensure that the mean is positive.

```
proc genmod data = lifdat;
   class mfg;
   model lifetime = mfg / dist=gamma
                          link=log
```

```
                                  type3;
run;
```

The output from these statements is displayed in Output 51.3.1.

Output 51.3.1 Gamma Model of Life Data

The GENMOD Procedure

Model Information	
Data Set	WORK.LIFDAT
Distribution	Gamma
Link Function	Log
Dependent Variable	lifetime

Class Level Information		
Class	Levels	Values
mfg	2	A B

Criteria For Assessing Goodness Of Fit			
Criterion	DF	Value	Value/DF
Deviance	199	287.0591	1.4425
Scaled Deviance	199	237.5335	1.1936
Pearson Chi-Square	199	211.6870	1.0638
Scaled Pearson X2	199	175.1652	0.8802
Log Likelihood		-1432.4177	
Full Log Likelihood		-1432.4177	
AIC (smaller is better)		2870.8353	
AICC (smaller is better)		2870.9572	
BIC (smaller is better)		2880.7453	

Analysis Of Maximum Likelihood Parameter Estimates					Wald 95% Confidence Limits			
Parameter		DF	Estimate	Standard Error			Wald Chi-Square	Pr > ChiSq
Intercept		1	6.1302	0.1043	5.9257	6.3347	3451.61	<.0001
mfg	A	1	0.0199	0.1559	-0.2857	0.3255	0.02	0.8985
mfg	B	0	0.0000	0.0000	0.0000	0.0000	.	.
Scale		1	0.8275	0.0714	0.6987	0.9800		

Note: The scale parameter was estimated by maximum likelihood.

LR Statistics For Type 3 Analysis			
Source	DF	Chi-Square	Pr > ChiSq
mfg	1	0.02	0.8985

The *p*-value of 0.8985 for the chi-square statistic in the Type 3 table indicates that there is no significant difference in the part life between the two manufacturers.

Using the following statements, you can refit the model without using the manufacturer as an effect. The LRCI option in the MODEL statement is specified to compute profile likelihood confidence intervals for the mean life and scale parameters.

```
proc genmod data = lifdat;
   model lifetime = / dist=gamma
                      link=log
                      lrci;
run;
```

Output 51.3.2 displays the results of fitting the model with the mfg effect omitted.

Output 51.3.2 Refitting of the Gamma Model: Omitting the mfg Effect

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	Likelihood Ratio 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	6.1391	0.0775	5.9904	6.2956	6268.10	<.0001
Scale	1	0.8274	0.0714	0.6959	0.9762		

Note: The scale parameter was estimated by maximum likelihood.

The intercept is the estimated log mean of the fitted gamma distribution, so that the mean life of the parts is

$$\mu = \exp(\text{INTERCEPT}) = \exp(6.1391) = 463.64$$

The SCALE parameter used in PROC GENMOD is the inverse of the gamma dispersion parameter, and it is sometimes called the gamma *index parameter*. See the section "Response Probability Distributions" on page 3655 for the definition of the gamma probability density function. A value of 1 for the index parameter corresponds to the exponential distribution . The estimated value of the scale parameter is 0.8274. The 95% profile likelihood confidence interval for the scale parameter is (0.6959, 0.9762), which does not contain 1. The hypothesis of an exponential distribution for the data is, therefore, rejected at the 0.05 level. A confidence interval for the mean life is

$$(\exp(5.99), \exp(6.30)) = (399.57, 542.18)$$

Example 51.4: Ordinal Model for Multinomial Data

(View the complete code for this example (genmex4.sas) in the example repository.)

This example illustrates how you can use the GENMOD procedure to fit a model to data measured on an ordinal scale. The following statements create a SAS data set called Icecream. The data set contains the results of a hypothetical taste test of three brands of ice cream. The three brands are rated for taste on a five-point scale from very good (vg) to very bad (vb). An analysis is performed to assess the differences in the ratings of the three brands. The variable taste contains the ratings, and the variable brand contains the brands tested. The variable count contains the number of testers rating each brand in each category.

The following statements create the Icecream data set:

```
data Icecream;
   input count brand$ taste$;
   datalines;
70  ice1 vg
71  ice1 g
151 ice1 m
30  ice1 b
46  ice1 vb
20  ice2 vg
36  ice2 g
130 ice2 m
74  ice2 b
70  ice2 vb
50  ice3 vg
55  ice3 g
140 ice3 m
52  ice3 b
50  ice3 vb
;
```

The following statements fit a cumulative logit model to the ordinal data with the variable taste as the response and the variable brand as a covariate. The variable count is used as a FREQ variable.

```
proc genmod data=Icecream rorder=data;
   freq count;
   class brand;
   model taste = brand / dist=multinomial
                         link=cumlogit
                         aggregate=brand
                         type1;
   estimate 'LogOR12' brand 1 -1 / exp;
   estimate 'LogOR13' brand 1  0  -1 / exp;
   estimate 'LogOR23' brand 0  1  -1 / exp;
run;
```

The AGGREGATE=BRAND option in the MODEL statement specifies the variable brand as defining multinomial populations for computing deviances and Pearson chi-squares. The RORDER=DATA option specifies that the taste variable levels be ordered by their order of appearance in the input data set—that is, from very good (vg) to very bad (vb). By default, the response is sorted in increasing ASCII order. Always check the "Response Profiles" table to verify that response levels are appropriately ordered. The TYPE1 option requests a Type 1 test for the significance of the covariate brand.

If $\gamma_j(\mathbf{x}) = \Pr(\text{taste} \leq j)$ is the cumulative probability of the jth or lower taste category, then the odds ratio comparing $\mathbf{x}_1$ to $\mathbf{x}_2$ is as follows:

$$\frac{\gamma_j(\mathbf{x}_1)/(1-\gamma_j(\mathbf{x}_1))}{\gamma_j(\mathbf{x}_2)/(1-\gamma_j(\mathbf{x}_2))} = \exp[(\mathbf{x}_1 - \mathbf{x}_2)'\boldsymbol{\beta}]$$

See McCullagh and Nelder (1989, Chapter 5) for details on the cumulative logit model. The ESTIMATE statements compute log odds ratios comparing each of brands. The EXP option in the ESTIMATE statements exponentiates the log odds ratios to form odds ratio estimates. Standard errors and confidence intervals

are also computed. Output 51.4.1 displays general information about the model and data, the levels of the CLASS variable brand, and the total number of occurrences of the ordered levels of the response variable taste.

Output 51.4.1 Ordinal Model Information

The GENMOD Procedure

Model Information	
Data Set	WORK.ICECREAM
Distribution	Multinomial
Link Function	Cumulative Logit
Dependent Variable	taste
Frequency Weight Variable	count

Class Level Information		
Class	**Levels**	**Values**
brand	3	ice1 ice2 ice3

Response Profile		
Ordered Value	**taste**	**Total Frequency**
1	vg	140
2	g	162
3	m	421
4	b	156
5	vb	166

Output 51.4.2 displays estimates of the intercept terms and covariates and associated statistics. The intercept terms correspond to the four cumulative logits defined on the taste categories in the order shown in Output 51.4.1. That is, Intercept1 is the intercept for the first cumulative logit, $\log(\frac{p_1}{1-p_1})$, Intercept2 is the intercept for the second cumulative logit, $\log(\frac{p_1+p_2}{1-(p_1+p_2)})$, and so forth.

Output 51.4.2 Parameter Estimates

Analysis Of Maximum Likelihood Parameter Estimates								
Parameter		**DF**	**Estimate**	**Standard Error**	**Wald 95% Confidence Limits**		**Wald Chi-Square**	**Pr > ChiSq**
Intercept1		1	-1.8578	0.1219	-2.0967	-1.6189	232.35	<.0001
Intercept2		1	-0.8646	0.1056	-1.0716	-0.6576	67.02	<.0001
Intercept3		1	0.9231	0.1060	0.7154	1.1308	75.87	<.0001
Intercept4		1	1.8078	0.1191	1.5743	2.0413	230.32	<.0001
brand	ice1	1	0.3847	0.1370	0.1162	0.6532	7.89	0.0050
brand	ice2	1	-0.6457	0.1397	-0.9196	-0.3719	21.36	<.0001
brand	ice3	0	0.0000	0.0000	0.0000	0.0000	.	.
Scale		0	1.0000	0.0000	1.0000	1.0000		

Note: The scale parameter was held fixed.

The Type 1 test displayed in Output 51.4.3 indicates that Brand is highly significant; that is, there are significant differences among the brands. The log odds ratios and odds ratios in the "ESTIMATE Statement

Results" table indicate the relative differences among the brands. For example, the odds ratio of 2.8 in the "Exp(LogOR12)" row indicates that the odds of brand 1 being in lower taste categories is 2.8 times the odds of brand 2 being in lower taste categories. Since, in this ordering, the lower categories represent the more favorable taste results, this indicates that brand 1 scored significantly better than brand 2. This is also apparent from the data in this example.

Output 51.4.3 Type 1 Tests and Odds Ratios

LR Statistics For Type 1 Analysis				
Source	Deviance	DF	Chi-Square	Pr > ChiSq
Intercepts	65.9576			
brand	9.8654	2	56.09	<.0001

Contrast Estimate Results										
		Mean					L'Beta			
Label	Mean Estimate	Confidence Limits		L'Beta Estimate	Standard Error	Alpha	Confidence Limits		Chi-Square	Pr > ChiSq
LogOR12	0.7370	0.6805	0.7867	1.0305	0.1401	0.05	0.7559	1.3050	54.11	<.0001
Exp(LogOR12)				2.8024	0.3926	0.05	2.1295	3.6878		
LogOR13	0.5950	0.5290	0.6577	0.3847	0.1370	0.05	0.1162	0.6532	7.89	0.0050
Exp(LogOR13)				1.4692	0.2013	0.05	1.1233	1.9217		
LogOR23	0.3439	0.2850	0.4081	-0.6457	0.1397	0.05	-0.9196	-0.3719	21.36	<.0001
Exp(LogOR23)				0.5243	0.0733	0.05	0.3987	0.6894		

Example 51.5: GEE for Binary Data with Logit Link Function

(View the complete code for this example (genmex5.sas) in the example repository.)

Output 51.5.1 displays a partial listing of a SAS data set of clinical trial data comparing two treatments for a respiratory disorder. See "Gee Model for Binary Data" in the SAS/STAT Sample Program Library for the complete data set. These data are from Stokes, Davis, and Koch (2000).

Patients in each of two centers are randomly assigned to groups receiving the active treatment or a placebo. During treatment, respiratory status, represented by the variable outcome (coded here as 0=poor, 1=good), is determined for each of four visits. The variables center, treatment, sex, and baseline (baseline respiratory status) are classification variables with two levels. The variable age (age at time of entry into the study) is a continuous variable.

Explanatory variables in the model are Intercept (x_{ij1}), treatment (x_{ij2}), center (x_{ij3}), sex (x_{ij4}), age (x_{ij5}), and baseline (x_{ij6}), so that $x' = [x_{ij1}, x_{ij2}, \ldots, x_{ij6}]$ is the vector of explanatory variables. Indicator variables for the classification explanatory variables can be automatically generated by listing them in the CLASS statement in PROC GENMOD. To be consistent with the analysis in Stokes, Davis, and Koch (2000), the four classification explanatory variables are coded as follows via options in the CLASS statement:

$$x_{ij2} = \begin{cases} 0 \text{ placebo} \\ 1 \text{ active} \end{cases} \qquad x_{ij3} = \begin{cases} 0 \text{ center } 1 \\ 1 \text{ center } 2 \end{cases}$$

$$x_{ij4} = \begin{cases} 0 \text{ male} \\ 1 \text{ female} \end{cases} \qquad x_{ij6} = \begin{cases} 0\ 0 \\ 1\ 1 \end{cases}$$

Suppose y_{ij} represents the respiratory status of patient i at the jth visit, $j = 1, \ldots, 4$, and $\mu_{ij} = \mathrm{E}(y_{ij})$ represents the mean of the respiratory status. Since the response data are binary, you can use the variance function for the binomial distribution $v(\mu_{ij}) = \mu_{ij}(1-\mu_{ij})$ and the logit link function $g(\mu_{ij}) = \log(\mu_{ij}/(1-\mu_{ij}))$. The model for the mean is $g(\mu_{ij}) = \mathbf{x}_{ij}{}'\boldsymbol{\beta}$, where $\boldsymbol{\beta}$ is a vector of regression parameters to be estimated.

Output 51.5.1 Respiratory Disorder Data

Obs	center	id	treatment	sex	age	baseline	visit1	visit2	visit3	visit4	visit	outcome
1	1	1	P	M	46	0	0	0	0	0	1	0
2	1	1	P	M	46	0	0	0	0	0	2	0
3	1	1	P	M	46	0	0	0	0	0	3	0
4	1	1	P	M	46	0	0	0	0	0	4	0
5	1	2	P	M	28	0	0	0	0	0	1	0
6	1	2	P	M	28	0	0	0	0	0	2	0
7	1	2	P	M	28	0	0	0	0	0	3	0
8	1	2	P	M	28	0	0	0	0	0	4	0
.												
.												
.												
214	2	1	P	F	39	0	0	0	0	0	1	0
215	2	1	P	F	39	0	0	0	0	0	2	0
216	2	1	P	F	39	0	0	0	0	0	3	0
217	2	1	P	F	39	0	0	0	0	0	4	0
218	2	2	A	M	25	0	0	1	1	1	1	0
219	2	2	A	M	25	0	0	1	1	1	2	1
220	2	2	A	M	25	0	0	1	1	1	3	1
221	2	2	A	M	25	0	0	1	1	1	4	1
.												
.												
.												
.												

The GEE solution is requested with the REPEATED statement in the GENMOD procedure. The option SUBJECT=ID(CENTER) specifies that the observations in any single cluster are uniquely identified by both center and id. An equivalent specification is SUBJECT=ID*CENTER. Since the same id values are used in each center, one of these specifications is needed. If id values were unique across all centers, SUBJECT=ID would be specified.

The option TYPE=UNSTR specifies the unstructured working correlation structure. The MODEL statement specifies the regression model for the mean with the binomial distribution variance function. The following SAS statements perform the GEE model fit:

```
proc genmod data=resp;
   class id treatment(ref="P") center(ref="1") sex(ref="M")
      baseline(ref="0");
   model outcome(event='1')=treatment center sex age baseline / dist=bin;
   repeated subject=id(center) / corr=unstr corrw;
run;
```

These statements first fit the generalized linear (GLM) model specified in the MODEL statement. The parameter estimates from the generalized linear model fit are not shown in the output, but they are used as initial values for the GEE solution. The EVENT='1' option in the MODEL statement models the probability that outcome = 1. If the EVENT='1' option had not been specified, the probability that outcome = 0 would be modeled by default.

Information about the GEE model is displayed in Output 51.5.2. The results of GEE model fitting are displayed in Output 51.5.3. Model goodness-of-fit criteria are displayed in Output 51.5.4. If you specify no other options, the standard errors, confidence intervals, Z scores, and p-values are based on empirical standard error estimates. You can specify the MODELSE option in the REPEATED statement to create a table based on model-based standard error estimates.

Output 51.5.2 Model Fitting Information

The GENMOD Procedure

GEE Model Information	
Correlation Structure	Unstructured
Subject Effect	id(center) (111 levels)
Number of Clusters	111
Correlation Matrix Dimension	4
Maximum Cluster Size	4
Minimum Cluster Size	4

Output 51.5.3 Results of Model Fitting

Working Correlation Matrix				
	Col1	**Col2**	**Col3**	**Col4**
Row1	1.0000	0.3351	0.2140	0.2953
Row2	0.3351	1.0000	0.4429	0.3581
Row3	0.2140	0.4429	1.0000	0.3964
Row4	0.2953	0.3581	0.3964	1.0000

Output 51.5.3 *continued*

Analysis Of GEE Parameter Estimates Empirical Standard Error Estimates							
Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept		-0.8882	0.4568	-1.7835	0.0071	-1.94	0.0519
treatment	A	1.2442	0.3455	0.5669	1.9214	3.60	0.0003
treatment	P	0.0000	0.0000	0.0000	0.0000	.	.
center	2	0.6558	0.3512	-0.0326	1.3442	1.87	0.0619
center	1	0.0000	0.0000	0.0000	0.0000	.	.
sex	F	0.1128	0.4408	-0.7512	0.9768	0.26	0.7981
sex	M	0.0000	0.0000	0.0000	0.0000	.	.
age		-0.0175	0.0129	-0.0427	0.0077	-1.36	0.1728
baseline	1	1.8981	0.3441	1.2237	2.5725	5.52	<.0001
baseline	0	0.0000	0.0000	0.0000	0.0000	.	.

Output 51.5.4 Model Fit Criteria

GEE Fit Criteria	
QIC	512.3416
QICu	499.6081

The nonsignificance of age and sex make them candidates for omission from the model.

Example 51.6: Log Odds Ratios and the ALR Algorithm

(View the complete code for this example (genmex6.sas) in the example repository.)

Since the respiratory data in Example 51.5 are binary, you can use the ALR algorithm to model the log odds ratios instead of using working correlations to model associations. In this example, a "fully parameterized cluster" model for the log odds ratio is fit. That is, there is a log odds ratio parameter for each unique pair of responses within clusters, and all clusters are parameterized identically. The following statements fit the same regression model for the mean as in Example 51.5 but use a regression model for the log odds ratios instead of a working correlation. The LOGOR=FULLCLUST option specifies a fully parameterized log odds ratio model.

```
proc genmod data=resp;
   class id treatment(ref="P") center(ref="1") sex(ref="M")
      baseline(ref="0");
   model outcome(event='1')=treatment center sex age baseline / dist=bin;
   repeated  subject=id(center) / logor=fullclust;
run;
```

The results of fitting the model are displayed in Output 51.6.1 along with a table that shows the correspondence between the log odds ratio parameters and the within-cluster pairs. Model goodness-of-fit criteria are shown in Output 51.6.2. The QIC for the ALR model shown in Output 51.6.2 is 511.86, whereas the QIC for the

unstructured working correlation model shown in Output 51.5.4 is 512.34, indicating that the ALR model is a slightly better fit.

Output 51.6.1 Results of Model Fitting

The GENMOD Procedure

Log Odds Ratio Parameter Information	
Parameter	**Group**
Alpha1	(1, 2)
Alpha2	(1, 3)
Alpha3	(1, 4)
Alpha4	(2, 3)
Alpha5	(2, 4)
Alpha6	(3, 4)

Analysis Of GEE Parameter Estimates
Empirical Standard Error Estimates

Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept		-0.9266	0.4513	-1.8111	-0.0421	-2.05	0.0400
treatment	A	1.2611	0.3406	0.5934	1.9287	3.70	0.0002
treatment	P	0.0000	0.0000	0.0000	0.0000	.	.
center	2	0.6287	0.3486	-0.0545	1.3119	1.80	0.0713
center	1	0.0000	0.0000	0.0000	0.0000	.	.
sex	F	0.1024	0.4362	-0.7526	0.9575	0.23	0.8144
sex	M	0.0000	0.0000	0.0000	0.0000	.	.
age		-0.0162	0.0125	-0.0407	0.0084	-1.29	0.1977
baseline	1	1.8980	0.3404	1.2308	2.5652	5.58	<.0001
baseline	0	0.0000	0.0000	0.0000	0.0000	.	.
Alpha1		1.6109	0.4892	0.6522	2.5696	3.29	0.0010
Alpha2		1.0771	0.4834	0.1297	2.0246	2.23	0.0259
Alpha3		1.5875	0.4735	0.6594	2.5155	3.35	0.0008
Alpha4		2.1224	0.5022	1.1381	3.1068	4.23	<.0001
Alpha5		1.8818	0.4686	0.9634	2.8001	4.02	<.0001
Alpha6		2.1046	0.4949	1.1347	3.0745	4.25	<.0001

Output 51.6.2 Model Fit Criteria

GEE Fit Criteria	
QIC	511.8589
QICu	499.6516

You can fit the same model by fully specifying the **z** matrix. The following statements create a data set containing the full **z** matrix:

```
data zin;
   keep id center z1-z6 y1 y2;
   array zin(6) z1-z6;
   set resp;
   by center id;
   if first.id
      then do;
      t = 0;
      do m = 1 to 4;
         do n = m+1 to 4;
            do j = 1 to 6;
               zin(j) = 0;
            end;
            y1 = m;
            y2 = n;
            t + 1;
            zin(t) = 1;
            output;
         end;
      end;
   end;
run;

proc print data=zin (obs=12);
run;
```

Output 51.6.3 displays the full **z** matrix for the first two clusters. The **z** matrix is identical for all clusters in this example.

Output 51.6.3 Full **z** Matrix Data Set

Obs	z1	z2	z3	z4	z5	z6	center	id	y1	y2
1	1	0	0	0	0	0	1	1	1	2
2	0	1	0	0	0	0	1	1	1	3
3	0	0	1	0	0	0	1	1	1	4
4	0	0	0	1	0	0	1	1	2	3
5	0	0	0	0	1	0	1	1	2	4
6	0	0	0	0	0	1	1	1	3	4
7	1	0	0	0	0	0	1	2	1	2
8	0	1	0	0	0	0	1	2	1	3
9	0	0	1	0	0	0	1	2	1	4
10	0	0	0	1	0	0	1	2	2	3
11	0	0	0	0	1	0	1	2	2	4
12	0	0	0	0	0	1	1	2	3	4

The following statements fit the model for fully parameterized clusters by fully specifying the **z** matrix. The results are identical to those shown previously.

```
proc genmod data=resp;
   class id treatment(ref="P") center(ref="1") sex(ref="M")
      baseline(ref="0");
   model outcome(event='1')=treatment center sex age baseline / dist=bin;
```

```
    repeated  subject=id(center) / logor=zfull
                                   zdata=zin
                                   zrow =(z1-z6)
                                   ypair=(y1 y2);
run;
```

Example 51.7: Log-Linear Model for Count Data

(View the complete code for this example (genmex7.sas) in the example repository.)

In this example the data, from Thall and Vail (1990), concern the treatment of people suffering from epileptic seizure episodes. These data are also analyzed in Diggle, Liang, and Zeger (1994). The data consist of the number of epileptic seizures in an eight-week baseline period, before any treatment, and in each of four two-week treatment periods, in which patients received either a placebo or the drug Progabide in addition to other therapy. A portion of the data is displayed in Table 51.18. See "Gee Model for Count Data, Exchangeable Correlation" in the SAS/STAT Sample Program Library for the complete data set.

Table 51.18 Epileptic Seizure Data

Patient ID	Treatment	Baseline	Visit1	Visit2	Visit3	Visit4
104	Placebo	11	5	3	3	3
106	Placebo	11	3	5	3	3
107	Placebo	6	2	4	0	5
.						
.						
.						
101	Progabide	76	11	14	9	8
102	Progabide	38	8	7	9	4
103	Progabide	19	0	4	3	0
.						
.						
.						

Model the data as a log-linear model with $V(\mu) = \mu$ (the Poisson variance function) and

$$\log(E(Y_{ij})) = \beta_0 + x_{i1}\beta_1 + x_{i2}\beta_2 + x_{i1}x_{i2}\beta_3 + \log(t_{ij})$$

where

$$Y_{ij} = \text{number of epileptic seizures in interval } j$$

$$t_{ij} = \text{length of interval } j$$

$$x_{i1} = \begin{cases} 1: & \text{weeks 8–16 (treatment)} \\ 0: & \text{weeks 0–8 (baseline)} \end{cases}$$

$$x_{i2} = \begin{cases} 1: & \text{progabide group} \\ 0: & \text{placebo group} \end{cases}$$

The correlations between the counts are modeled as $r_{ij} = \alpha$, $i \neq j$ (exchangeable correlations). For comparison, the correlations are also modeled as independent (identity correlation matrix). In this model, the regression parameters have the interpretation in terms of the log seizure rate displayed in Table 51.19.

Table 51.19 Interpretation of Regression Parameters

Treatment	Visit	$\log(E(Y_{ij})/t_{ij})$
Placebo	Baseline	β_0
	1–4	$\beta_0 + \beta_1$
Progabide	Baseline	$\beta_0 + \beta_2$
	1–4	$\beta_0 + \beta_1 + \beta_2 + \beta_3$

The difference between the log seizure rates in the pretreatment (baseline) period and the treatment periods is β_1 for the placebo group and $\beta_1 + \beta_3$ for the Progabide group. A value of $\beta_3 < 0$ indicates a reduction in the seizure rate.

Output 51.7.1 lists the first 14 observations of the data, which are arranged as one visit per observation:

Output 51.7.1 Partial Listing of the Seizure Data

Obs	id	y	visit	trt	bline	age
1	104	5	1	0	11	31
2	104	3	2	0	11	31
3	104	3	3	0	11	31
4	104	3	4	0	11	31
5	106	3	1	0	11	30
6	106	5	2	0	11	30
7	106	3	3	0	11	30
8	106	3	4	0	11	30
9	107	2	1	0	6	25
10	107	4	2	0	6	25
11	107	0	3	0	6	25
12	107	5	4	0	6	25
13	114	4	1	0	8	36
14	114	4	2	0	8	36

Some further data manipulations create an observation for the baseline measures, a log time interval variable for use as an offset, and an indicator variable for whether the observation is for a baseline measurement or a visit measurement. Patient 207 is deleted as an outlier, as in the Diggle, Liang, and Zeger (1994) analysis. The following statements prepare the data for analysis with PROC GENMOD:

```
data new;
   set thall;
   output;
   if visit=1 then do;
      y=bline;
      visit=0;
      output;
   end;
```

```
run;

data new;
   set new;
   if id ne 207;
   if visit=0 then do;
      x1=0;
      ltime=log(8);
   end;
   else do;
      x1=1;
      ltime=log(2);
   end;
run;
```

For comparison with the GEE results, an ordinary Poisson regression is first fit. The results are shown in Output 51.7.2.

Output 51.7.2 Maximum Likelihood Estimates

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	1.3476	0.0341	1.2809	1.4144	1565.44	<.0001
x1	1	0.1108	0.0469	0.0189	0.2027	5.58	0.0181
trt	1	-0.1080	0.0486	-0.2034	-0.0127	4.93	0.0264
x1*trt	1	-0.3016	0.0697	-0.4383	-0.1649	18.70	<.0001
Scale	0	1.0000	0.0000	1.0000	1.0000		

Note: The scale parameter was held fixed.

The GEE solution is requested with the REPEATED statement in the GENMOD procedure. The SUBJECT=ID option indicates that the variable id describes the observations for a single cluster, and the CORRW option displays the working correlation matrix. The TYPE= option specifies the correlation structure; the value EXCH indicates the exchangeable structure.

The following statements perform the analysis:

```
proc genmod data=new;
   class id;
   model y=x1 | trt / d=poisson offset=ltime;
   repeated subject=id / corrw covb type=exch;
run;
```

These statements first fit a generalized linear model (GLM) to these data by maximum likelihood. The estimates are not shown in the output, but are used as initial values for the GEE solution.

Information about the GEE model is displayed in Output 51.7.3. The results of fitting the model are displayed in Output 51.7.4. Compare these with the model of independence displayed in Output 51.7.2. The parameter estimates are nearly identical, but the standard errors for the independence case are underestimated. The coefficient of the interaction term, β_3, is highly significant under the independence model and marginally significant with the exchangeable correlations model.

Output 51.7.3 GEE Model Information

The GENMOD Procedure

GEE Model Information	
Correlation Structure	Exchangeable
Subject Effect	id (58 levels)
Number of Clusters	58
Correlation Matrix Dimension	5
Maximum Cluster Size	5
Minimum Cluster Size	5

Output 51.7.4 GEE Parameter Estimates

Analysis Of GEE Parameter Estimates
Empirical Standard Error Estimates

Parameter	Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept	1.3476	0.1574	1.0392	1.6560	8.56	<.0001
x1	0.1108	0.1161	-0.1168	0.3383	0.95	0.3399
trt	-0.1080	0.1937	-0.4876	0.2716	-0.56	0.5770
x1*trt	-0.3016	0.1712	-0.6371	0.0339	-1.76	0.0781

Table 51.20 displays the regression coefficients, standard errors, and normalized coefficients that result from fitting the model with independent and exchangeable working correlation matrices.

Table 51.20 Results of Model Fitting

Variable	Correlation Structure	Coef.	Std. Error	Coef./S.E.
Intercept	Exchangeable	1.35	0.16	8.56
	Independent	1.35	0.03	39.52
Visit (x_1)	Exchangeable	0.11	0.12	0.95
	Independent	0.11	0.05	2.36
Treat (x_2)	Exchangeable	–0.11	0.19	–0.56
	Independent	–0.11	0.05	–2.22
$x_1 * x_2$	Exchangeable	–0.30	0.17	–1.76
	Independent	–0.30	0.07	–4.32

The fitted exchangeable correlation matrix is specified with the CORRW option and is displayed in Output 51.7.5.

Output 51.7.5 Working Correlation Matrix

Working Correlation Matrix					
	Col1	Col2	Col3	Col4	Col5
Row1	1.0000	0.5941	0.5941	0.5941	0.5941
Row2	0.5941	1.0000	0.5941	0.5941	0.5941
Row3	0.5941	0.5941	1.0000	0.5941	0.5941
Row4	0.5941	0.5941	0.5941	1.0000	0.5941
Row5	0.5941	0.5941	0.5941	0.5941	1.0000

If you specify the COVB option, you produce both the model-based (naive) and the empirical (robust) covariance matrices. Output 51.7.6 contains these estimates.

Output 51.7.6 Covariance Matrices

Covariance Matrix (Model-Based)				
	Prm1	Prm2	Prm3	Prm4
Prm1	0.01223	0.001520	-0.01223	-0.001520
Prm2	0.001520	0.01519	-0.001520	-0.01519
Prm3	-0.01223	-0.001520	0.02495	0.005427
Prm4	-0.001520	-0.01519	0.005427	0.03748

Covariance Matrix (Empirical)				
	Prm1	Prm2	Prm3	Prm4
Prm1	0.02476	-0.001152	-0.02476	0.001152
Prm2	-0.001152	0.01348	0.001152	-0.01348
Prm3	-0.02476	0.001152	0.03751	-0.002999
Prm4	0.001152	-0.01348	-0.002999	0.02931

The two covariance estimates are similar, indicating an adequate correlation model.

Example 51.8: Model Assessment of Multiple Regression Using Aggregates of Residuals

(View the complete code for this example (genmex8.sas) in the example repository.)

This example illustrates the use of cumulative residuals to assess the adequacy of a normal linear regression model.

Neter et al. (1996, Section 8.2) describe a study of 54 patients undergoing a certain kind of liver operation in a surgical unit. The data consist of the survival time and certain covariates. After a model selection procedure, they arrived at the following model:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \epsilon$$

where Y is the logarithm (base 10) of the survival time; X_1, X_2, X_3 are *blood-clotting score*, *prognostic index*, and *enzyme function*, respectively; and ϵ is a normal error term. A listing of the SAS data set containing the

data is shown in Output 51.8.1. The variables Y, X1, X2, and X3 correspond to Y, X_1, X_2, and X_3, and LogX1 is $\log(X_1)$. The PROC GENMOD fit of the model is shown in Output 51.8.2. The analysis first focuses on the adequacy of the functional form of X_1, *blood-clotting score*.

Output 51.8.1 Surgical Unit Example Data

Obs	Y	X1	X2	X3	LogX1
1	2.3010	6.7	62	81	0.82607
2	2.0043	5.1	59	66	0.70757
3	2.3096	7.4	57	83	0.86923
4	2.0043	6.5	73	41	0.81291
5	2.7067	7.8	65	115	0.89209
6	1.9031	5.8	38	72	0.76343
7	1.9031	5.7	46	63	0.75587
8	2.1038	3.7	68	81	0.56820
9	2.3054	6.0	67	93	0.77815
10	2.3075	3.7	76	94	0.56820
11	2.5172	6.3	84	83	0.79934
12	1.8129	6.7	51	43	0.82607
13	2.9191	5.8	96	114	0.76343
14	2.5185	5.8	83	88	0.76343
15	2.2253	7.7	62	67	0.88649
16	2.3365	7.4	74	68	0.86923
17	1.9395	6.0	85	28	0.77815
18	1.5315	3.7	51	41	0.56820
19	2.3324	7.3	68	74	0.86332
20	2.2355	5.6	57	87	0.74819
21	2.0374	5.2	52	76	0.71600
22	2.1335	3.4	83	53	0.53148
23	1.8451	6.7	26	68	0.82607
24	2.3424	5.8	67	86	0.76343
25	2.4409	6.3	59	100	0.79934
26	2.1584	5.8	61	73	0.76343
27	2.2577	5.2	52	86	0.71600
28	2.7589	11.2	76	90	1.04922
29	1.8573	5.2	54	56	0.71600
30	2.2504	5.8	76	59	0.76343
31	1.8513	3.2	64	65	0.50515
32	1.7634	8.7	45	23	0.93952
33	2.0645	5.0	59	73	0.69897
34	2.4698	5.8	72	93	0.76343
35	2.0607	5.4	58	70	0.73239
36	2.2648	5.3	51	99	0.72428
37	2.0719	2.6	74	86	0.41497
38	2.0792	4.3	8	119	0.63347
39	2.1790	4.8	61	76	0.68124
40	2.1703	5.4	52	88	0.73239
41	1.9777	5.2	49	72	0.71600
42	1.8751	3.6	28	99	0.55630
43	2.6840	8.8	86	88	0.94448
44	2.1847	6.5	56	77	0.81291
45	2.2810	3.4	77	93	0.53148
46	2.0899	6.5	40	84	0.81291
47	2.4928	4.5	73	106	0.65321
48	2.5999	4.8	86	101	0.68124

Output 51.8.1 *continued*

Obs	Y	X1	X2	X3	LogX1
49	2.1987	5.1	67	77	0.70757
50	2.4914	3.9	82	103	0.59106
51	2.0934	6.6	77	46	0.81954
52	2.0969	6.4	85	40	0.80618
53	2.2967	6.4	59	85	0.80618
54	2.4955	8.8	78	72	0.94448

In order to assess the adequacy of the fitted multiple regression model, the ASSESS statement in the following SAS statements is used to create the plots of cumulative residuals against X1 shown in Output 51.8.3 and Output 51.8.4 and the summary table in Output 51.8.5:

```
ods graphics on;

proc genmod data=Surg;
   model Y = X1 X2 X3 / scale=Pearson;
   assess var=(X1) / resample=10000
                     seed=603708000
                     crpanel;
run;
```

Output 51.8.2 Regression Model for Linear X1

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	0.4836	0.0426	0.4001	0.5672	128.71	<.0001
X1	1	0.0692	0.0041	0.0612	0.0772	288.17	<.0001
X2	1	0.0093	0.0004	0.0085	0.0100	590.45	<.0001
X3	1	0.0095	0.0003	0.0089	0.0101	966.07	<.0001
Scale	0	0.0469	0.0000	0.0469	0.0469		

Note: The scale parameter was estimated by the square root of Pearson's Chi-Square/DOF.

See Lin, Wei, and Ying (2002) for details about model assessment that uses cumulative residual plots. The RESAMPLE= keyword specifies that a *p*-value be computed based on a sample of 10,000 simulated residual paths. A random number seed is specified by the SEED= keyword for reproducibility. If you do not specify the seed, one is derived from the time of day. The keyword CRPANEL specifies that the panel of four cumulative residual plots shown in Output 51.8.4 be created, each with two simulated paths. The single residual plot with 20 simulated paths in Output 51.8.3 is created by default.

To request these graphs, ODS Graphics must be enabled and you must specify the ASSESS statement. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the GENMOD procedure, see the section "ODS Graphics" on page 3718.

Output 51.8.3 Cumulative Residual Plot for Linear X1 Fit

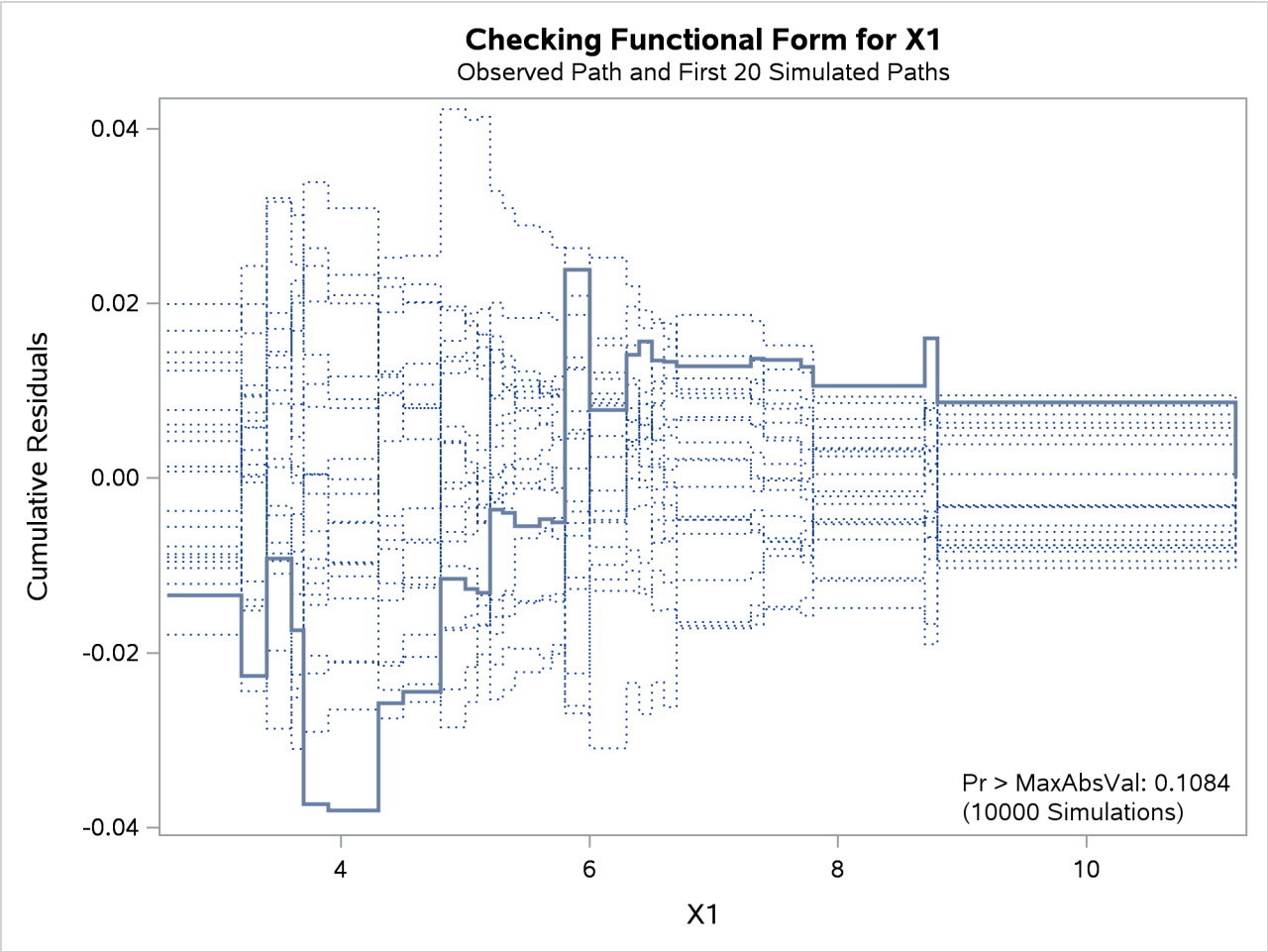

Output 51.8.4 Cumulative Residual Panel Plot for Linear X1 Fit

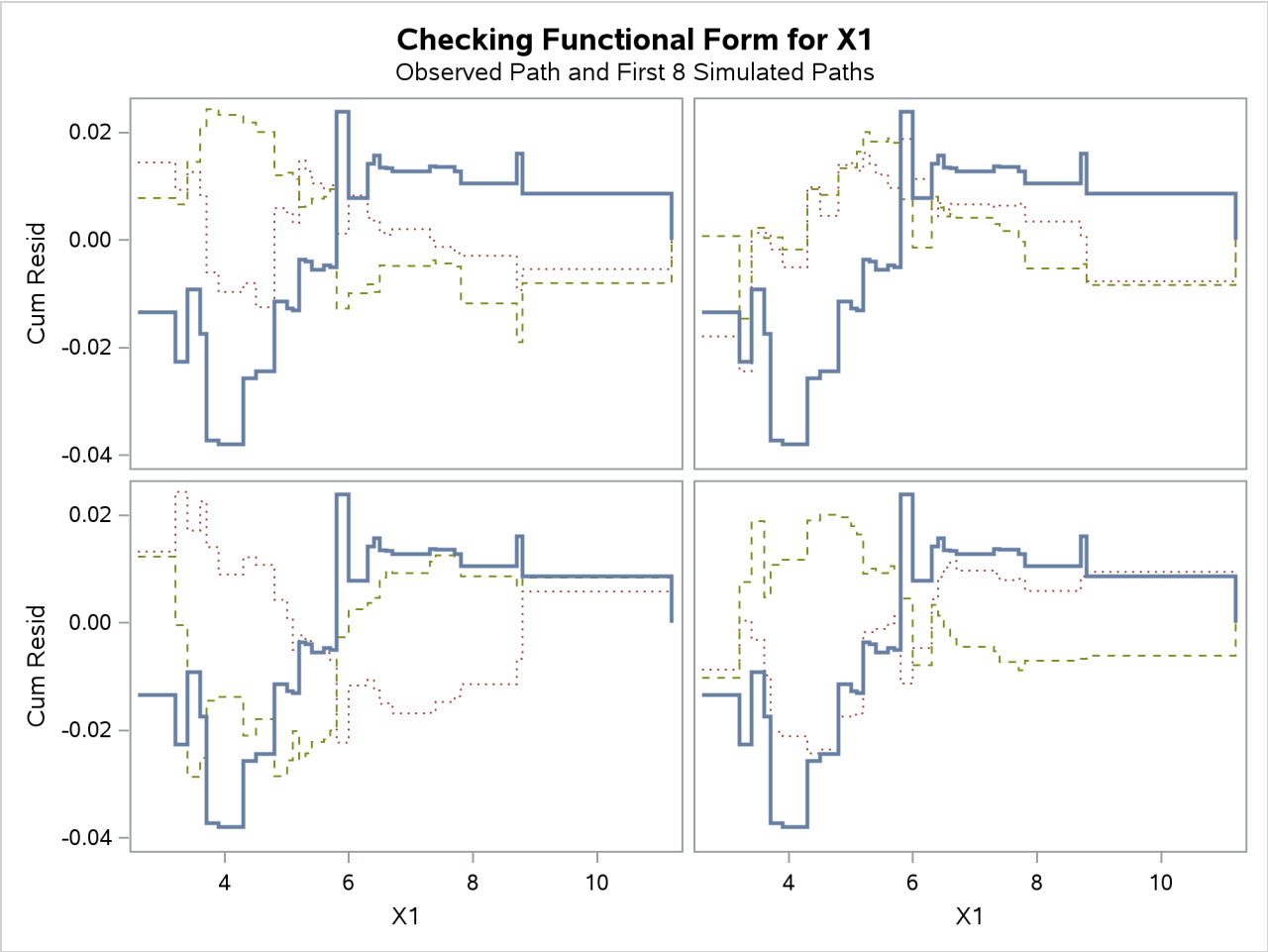

Output 51.8.5 Summary of Model Assessment

Assessment Summary				
Assessment Variable	Maximum Absolute Value	Replications	Seed	Pr > MaxAbsVal
X1	0.0380	10000	603708000	0.1084

The p-value of 0.1084 reported on Output 51.8.3 and Output 51.8.5 suggests that a more adequate model might be possible. The observed cumulative residuals in Output 51.8.3 and Output 51.8.4, represented by the heavy lines, seem atypical of the simulated curves, represented by the light lines, reinforcing the conclusion that a more appropriate functional form for X1 is possible.

The cumulative residual plots in Output 51.8.6 provide guidance in determining a more appropriate functional form. The four curves were created from simple forms of model misspecification by using simulated data. The mean models of the data and the fitted model are shown in Table 51.21.

Output 51.8.6 Typical Cumulative Residual Patterns

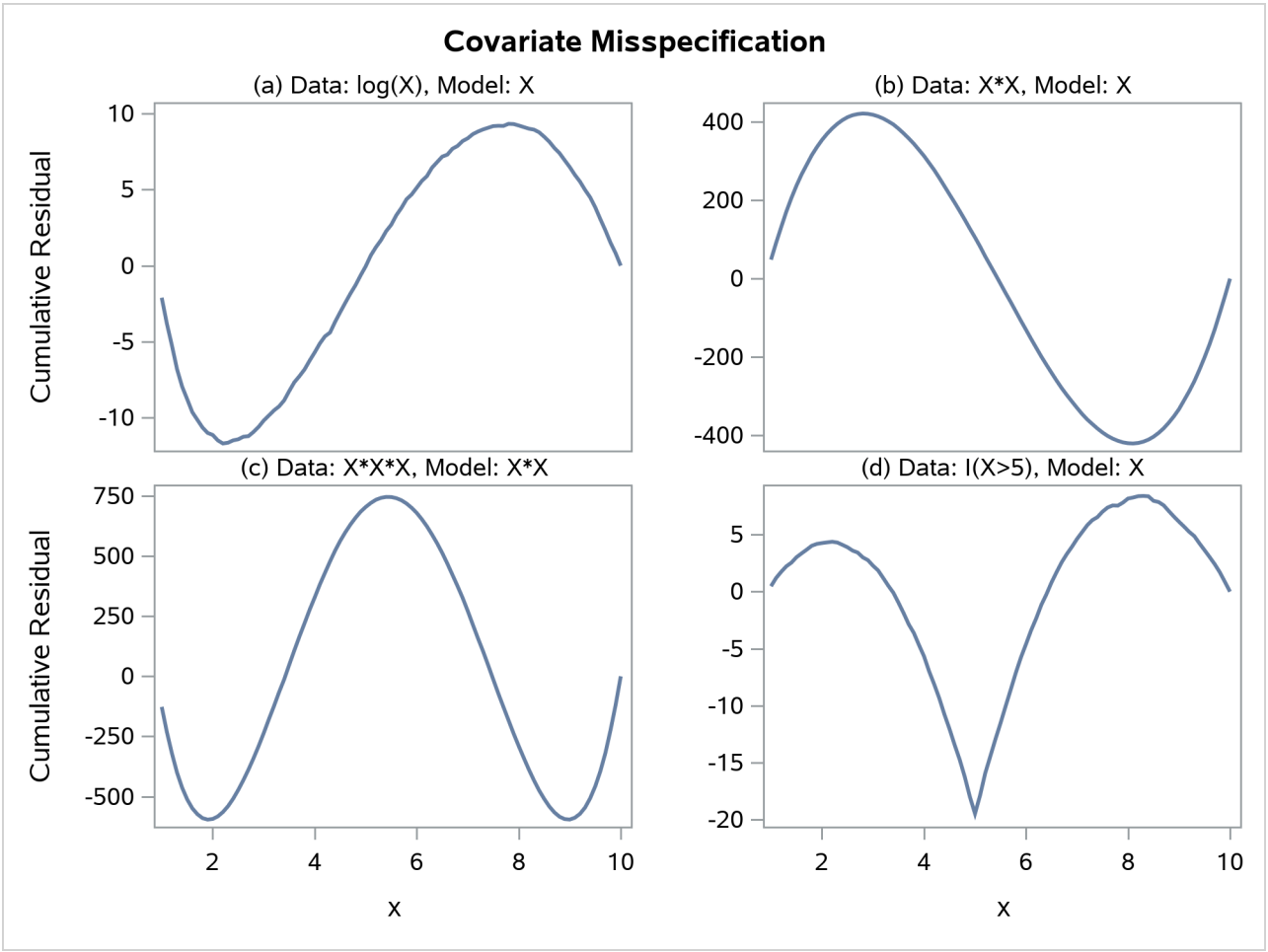

Table 51.21 Model Misspecifications

Plot	Data E(Y)	Fitted Model E(Y)
(a)	$\log(X)$	X
(b)	$X + X^2$	X
(c)	$X + X^2 + X^3$	$X + X^2$
(d)	$I(X > 5)$	X

The observed cumulative residual pattern in Output 51.8.3 and Output 51.8.4 most resembles the behavior of the curve in plot (a) of Output 51.8.6, indicating that $\log(X_1)$ might be a more appropriate term in the model than X_1.

The following SAS statements fit a model with LogX1 in place of X1 and request a model assessment:

```
proc genmod data=Surg;
   model Y = LogX1 X2 X3 / scale=Pearson;
   assess var=(LogX1) / resample=10000
                        seed=603708000;
run;
```

The revised model fit is shown in Output 51.8.7, the p-value from the simulation is 0.4777, and the cumulative residuals plotted in Output 51.8.8 show no systematic trend. The log transformation for X1 is more appropriate. Under the revised model, the p-values for testing the functional forms of X2 and X3 are 0.20 and 0.63, respectively; and the p-value for testing the linearity of the model is 0.65. Thus, the revised model seems reasonable.

Output 51.8.7 Multiple Regression Model with Log(X1)

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	0.1844	0.0504	0.0857	0.2832	13.41	0.0003
LogX1	1	0.9121	0.0491	0.8158	1.0083	345.05	<.0001
X2	1	0.0095	0.0004	0.0088	0.0102	728.62	<.0001
X3	1	0.0096	0.0003	0.0090	0.0101	1139.73	<.0001
Scale	0	0.0434	0.0000	0.0434	0.0434		

Note: The scale parameter was estimated by the square root of Pearson's Chi-Square/DOF.

Output 51.8.8 Cumulative Residual Plot with Log(X1)

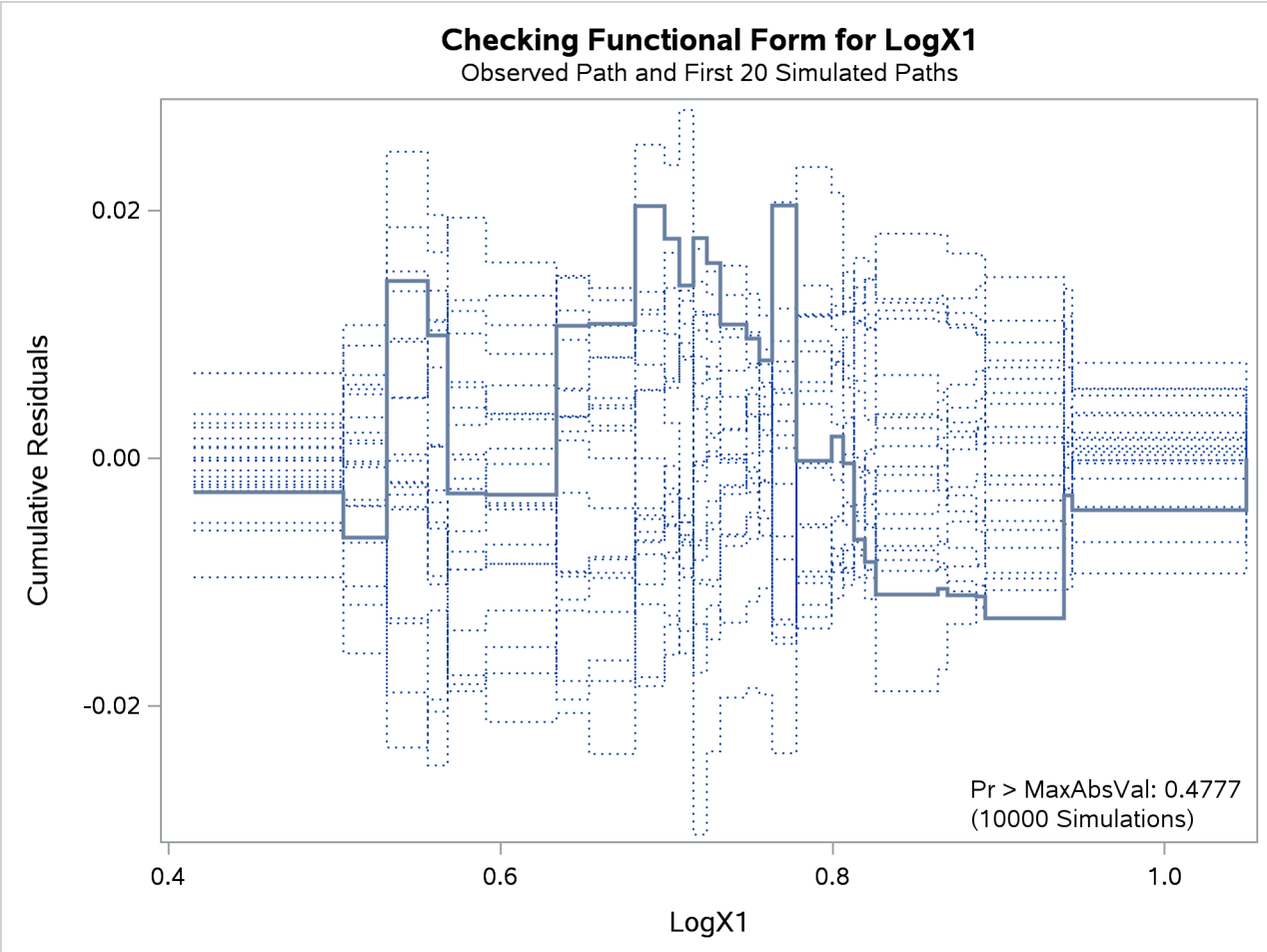

Example 51.9: Assessment of a Marginal Model for Dependent Data

(View the complete code for this example (genmex9.sas) in the example repository.)

This example illustrates the use of cumulative residuals to assess the adequacy of a marginal model for dependent data fit by generalized estimating equations (GEEs). The assessment methods are applied to CD4 count data from an AIDS clinical trial reported by Fischl, Richman, and Hansen (1990) and reanalyzed by Lin, Wei, and Ying (2002). The study randomly assigned 360 HIV patients to the drug AZT and 351 patients to placebo. CD4 counts were measured repeatedly over the course of the study. The data used here are the 4328 measurements taken in the first 40 weeks of the study.

The analysis focuses on the time trend of the response. The first model considered is

$$E(y_{ik}) = \beta_0 + \beta_1 T_{ik} + \beta_2 T_{ik}^2 + \beta_3 R_i T_{ik} + \beta_4 R_i T_{ik}^2$$

where T_{ik} is the time (in weeks) of the kth measurement on the ith patient, y_{ik} is the CD4 count at T_{ik} for the ith patient, and R_i is the indicator of AZT for the ith patient. Normal errors and an independent working correlation are assumed.

The following statements create the SAS data set cd4:

```
data cd4;
   input Id Y Time Time2 TrtTime TrtTime2;
   Time3 = Time2 * Time;
   TrtTime3 = TrtTime2 * Time;
   datalines;
1     264.00024       -0.28571        0.08163       -0.28571        0.08163
1     175.00070        4.14286       17.16327        4.14286       17.16327
1     306.00150        8.14286       66.30612        8.14286       66.30612
1     331.99835       12.14286      147.44898       12.14286      147.44898
1     309.99929       16.14286      260.59184       16.14286      260.59184
1     185.00077       28.71429      824.51020       28.71429      824.51020
1     175.00070       40.14286     1611.44898       40.14286     1611.44898
2     574.99998       -0.57143        0.32653        0.00000        0.00000

   ... more lines ...

711     363.99859        8.14286       66.30612        8.14286       66.30612
711     488.00224       12.14286      147.44898       12.14286      147.44898
711     240.00026       18.14286      329.16327       18.14286      329.16327
;
```

The following SAS statements fit the preceding model, create the cumulative residual plot in Output 51.9.1, and compute a p-value for the model.

To request these graphs, ODS Graphics must be enabled and you must specify the ASSESS statement. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the GENMOD procedure, see the section "ODS Graphics" on page 3718.

Here, the SAS data set variables Time, Time2, TrtTime, and TrtTime2 correspond to T_{ik}, T_{ik}^2, $R_i T_{ik}$, and $R_i T_{ik}^2$, respectively. The variable Id identifies individual patients.

```
ods graphics on;

proc genmod data=cd4;
   class Id;
   model Y = Time Time2 TrtTime TrtTime2;
   repeated sub=Id;
   assess var=(Time) / resample
                       seed=603708000;
run;
```

Output 51.9.1 Cumulative Residual Plot for Quadratic Time Fit

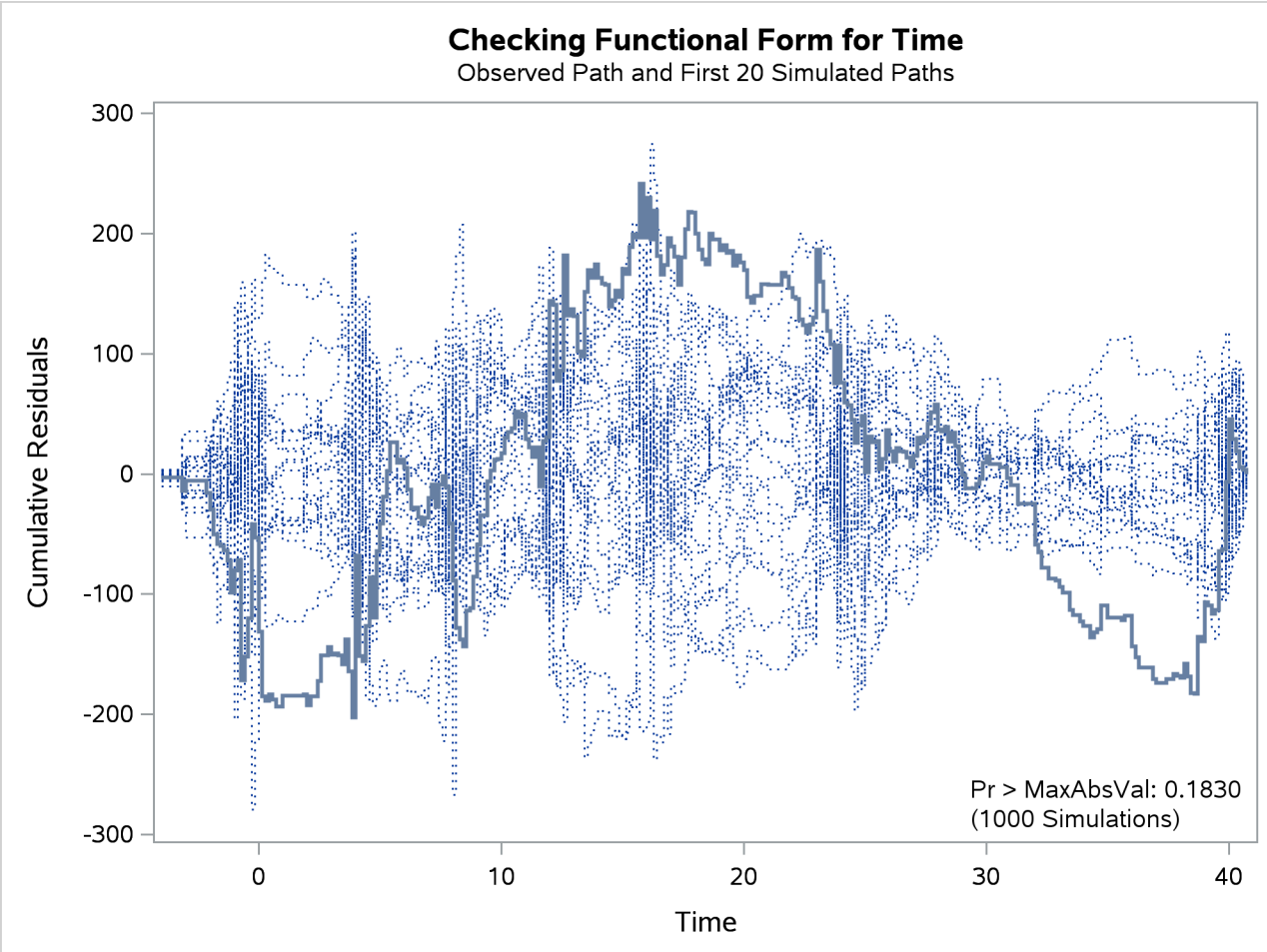

The cumulative residual plot in Output 51.9.1 displays cumulative residuals versus time for the model and 20 simulated realizations. The associated p-value, also shown in Output 51.9.1, is 0.18. These results indicate that a more satisfactory model might be possible. The observed cumulative residual pattern most resembles plot (c) in Output 51.8.6, suggesting cubic time trends.

The following SAS statements fit the model, create the plot in Output 51.9.2, and compute a p-value for a model with the additional terms T_{ik}^3 and $R_i T_{ik}^3$:

```
proc genmod data=cd4;
   class Id;
   model Y = Time Time2 Time3 TrtTime TrtTime2 TrtTime3;
   repeated sub=Id;
   assess var=(Time) / resample
                       seed=603708000;
run;
```

Output 51.9.2 Cumulative Residual Plot for Cubic Time Fit

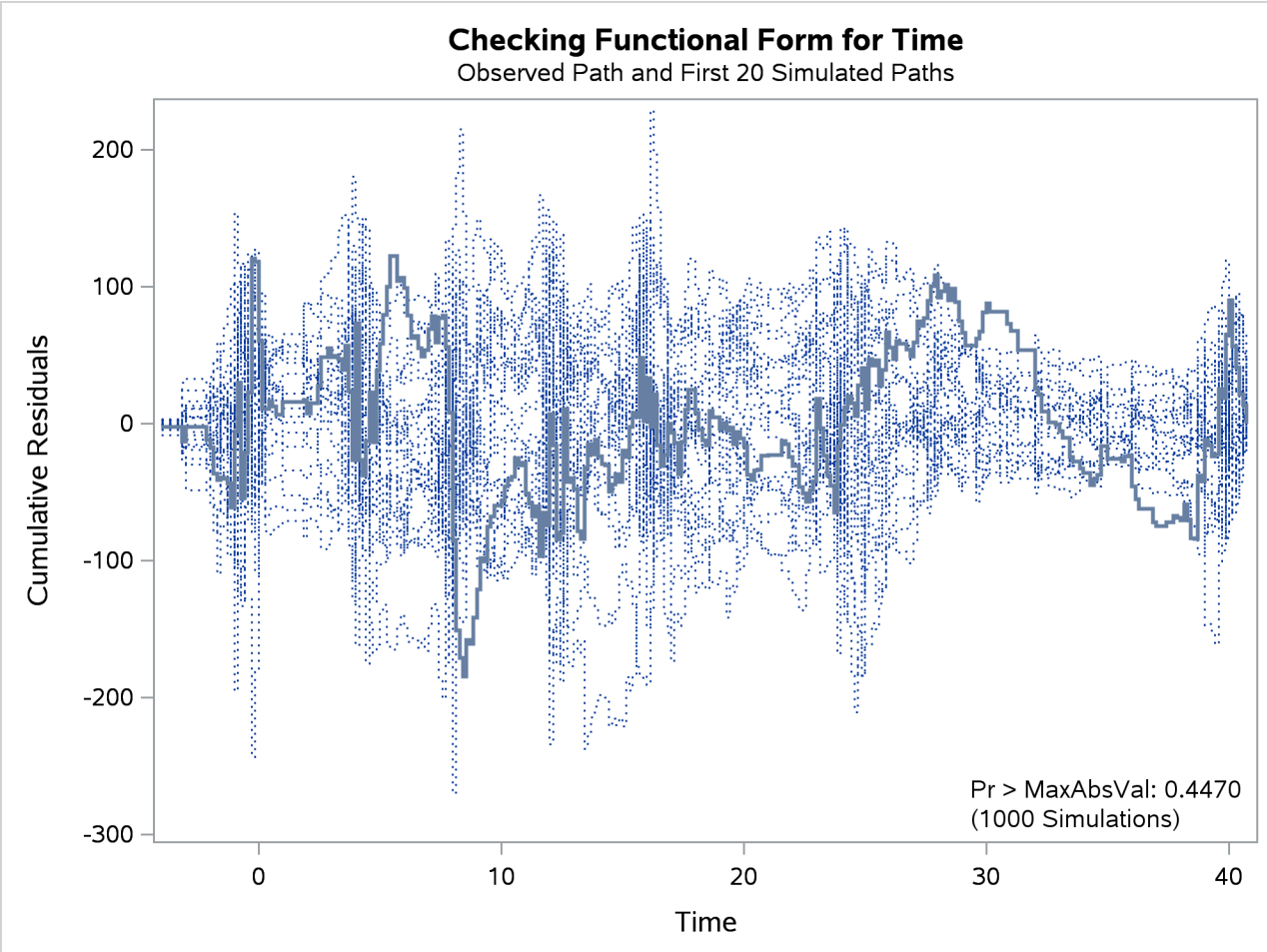

The observed cumulative residual pattern appears more typical of the simulated realizations, and the p-value is 0.45, indicating that the model with cubic time trends is more appropriate.

Example 51.10: Bayesian Analysis of a Poisson Regression Model

(View the complete code for this example (genmex10.sas) in the example repository.)

This example illustrates a Bayesian analysis of a log-linear Poisson regression model. Consider the following data on patients from clinical trials. The data set is a subset of the data described in Ibrahim, Chen, and Lipsitz (1999).

```
data Liver;
   input X1-X6 Y;
   datalines;
19.1358    50.0110     51.000      0        0       1        3
23.5970    18.4959      3.429      0        0       1        9
20.0474    56.7699      3.429      1        1       0        6
28.0277    59.7836      4.000      0        0       1        6
```

```
28.6851    74.1589      5.714      1        0       1        1
18.8092    31.0630      2.286      0        1       1       61
28.7201    52.9178     37.286      1        0       1        6
21.3669    61.6603     54.143      0        1       1        6
23.7332    42.2904      0.571      1        0       1       21
20.4783    22.1260     19.000      1        0       1        6

   ... more lines ...

17.0993    48.8384      3.000      0        0       0       9
19.1327    65.3425      2.571      1        0       0       1
17.3010    51.4493      4.429      1        0       0       6
;
```

The primary interest is in prediction of the number of cancerous liver nodes when a patient enters the trials, by using six other baseline characteristics. The number of nodes is modeled by a Poisson regression model with the six baseline characteristics as covariates. The response and regression variables are as follows:

Y	Number of Cancerous Liver Nodes
X1	Body Mass Index
X2	Age, in Years
X3	Time Since Diagnosis of Disease, in Weeks
X4	Two Biochemical Markers (each classified as normal=1 or abnormal=0)
X5	Anti Hepatitis B Antigen
X6	Associated Jaundice (yes=1, no=0)

Two analyses are performed using PROC GENMOD. The first analysis uses noninformative normal prior distributions, and the second analysis uses an informative normal prior for one of the regression parameters.

In the following BAYES statement, COEFFPRIOR=NORMAL specifies a noninformative independent normal prior distribution with zero mean and variance 10^6 for each parameter.

The initial analysis is performed using PROC GENMOD to obtain Bayesian estimates of the regression coefficients by using the following SAS statements:

```
proc genmod data=Liver;
   model Y = X1-X6 / dist=Poisson link=log;
   bayes seed=1 coeffprior=normal;
run;
```

Maximum likelihood estimates of the model parameters are computed by default. These are shown in the "Analysis of Maximum Likelihood Parameter Estimates" table in Output 51.10.1.

Output 51.10.1 Maximum Likelihood Parameter Estimates

The GENMOD Procedure

Bayesian Analysis

Analysis Of Maximum Likelihood Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits	
Intercept	1	2.4508	0.2284	2.0032	2.8984
X1	1	-0.0044	0.0080	-0.0201	0.0114
X2	1	-0.0135	0.0024	-0.0181	-0.0088
X3	1	-0.0029	0.0022	-0.0072	0.0014
X4	1	-0.2715	0.0795	-0.4272	-0.1157
X5	1	0.3215	0.0832	0.1585	0.4845
X6	1	0.2077	0.0827	0.0456	0.3698
Scale	0	1.0000	0.0000	1.0000	1.0000

Note: The scale parameter was held fixed.

Noninformative independent normal prior distributions with zero means and variances of 10^6 were used in the initial analysis. These are shown in Output 51.10.2.

Output 51.10.2 Regression Coefficient Priors

The GENMOD Procedure

Bayesian Analysis

Independent Normal Prior for Regression Coefficients		
Parameter	Mean	Precision
Intercept	0	1E-6
X1	0	1E-6
X2	0	1E-6
X3	0	1E-6
X4	0	1E-6
X5	0	1E-6
X6	0	1E-6

Initial values for the Markov chain are listed in the "Initial Values and Seeds" table in Output 51.10.3. The random number seed is also listed so that you can reproduce the analysis. Since no seed was specified, the seed shown was derived from the time of day.

Output 51.10.3 MCMC Initial Values and Seeds

Initial Values of the Chain								
Chain	Seed	Intercept	X1	X2	X3	X4	X5	X6
1	1	2.450813	-0.00435	-0.01347	-0.00291	-0.27149	0.321507	0.207713

Summary statistics for the posterior sample are displayed in the "Fit Statistics," "Descriptive Statistics for the Posterior Sample," "Interval Statistics for the Posterior Sample," and "Posterior Correlation Matrix" tables in Output 51.10.4, Output 51.10.5, Output 51.10.6, and Output 51.10.7, respectively. Since noninformative prior distributions for the regression coefficients were used, the mean and standard deviations of the posterior distributions for the model parameters are close to the maximum likelihood estimates and standard errors.

Output 51.10.4 Fit Statistics

Fit Statistics	
DIC (smaller is better)	829.810
pD (effective number of parameters)	7.005

Output 51.10.5 Descriptive Statistics

The GENMOD Procedure

Bayesian Analysis

Posterior Summaries						
				Percentiles		
Parameter	N	Mean	Standard Deviation	25%	50%	75%
Intercept	10000	2.4483	0.2320	2.2903	2.4493	2.6093
X1	10000	-0.00475	0.00809	-0.0101	-0.00466	0.000851
X2	10000	-0.0134	0.00237	-0.0150	-0.0134	-0.0118
X3	10000	-0.00303	0.00220	-0.00445	-0.00298	-0.00150
X4	10000	-0.2703	0.0799	-0.3241	-0.2725	-0.2190
X5	10000	0.3202	0.0828	0.2642	0.3209	0.3775
X6	10000	0.2106	0.0838	0.1533	0.2111	0.2663

Output 51.10.6 Interval Statistics

Posterior Intervals					
		Equal-Tail			
Parameter	Alpha	Interval		HPD Interval	
Intercept	0.050	1.9903	2.9059	2.0289	2.9321
X1	0.050	-0.0209	0.0108	-0.0211	0.0106
X2	0.050	-0.0181	-0.00870	-0.0184	-0.00908
X3	0.050	-0.00761	0.00105	-0.00745	0.00113
X4	0.050	-0.4257	-0.1063	-0.4314	-0.1152
X5	0.050	0.1563	0.4804	0.1574	0.4811
X6	0.050	0.0450	0.3777	0.0468	0.3788

Output 51.10.7 Posterior Sample Correlation Matrix

Posterior Correlation Matrix							
Parameter	**Intercept**	**X1**	**X2**	**X3**	**X4**	**X5**	**X6**
Intercept	1.000	-0.708	-0.432	-0.046	-0.261	-0.185	-0.422
X1	-0.708	1.000	-0.202	-0.047	-0.035	0.078	0.129
X2	-0.432	-0.202	1.000	0.035	0.076	0.054	0.117
X3	-0.046	-0.047	0.035	1.000	0.027	-0.042	-0.077
X4	-0.261	-0.035	0.076	0.027	1.000	-0.024	0.127
X5	-0.185	0.078	0.054	-0.042	-0.024	1.000	-0.037
X6	-0.422	0.129	0.117	-0.077	0.127	-0.037	1.000

Posterior sample autocorrelations for each model parameter are shown in Output 51.10.8. The autocorrelation after 10 lags is negligible for all parameters, indicating good mixing in the Markov chain.

Output 51.10.8 Posterior Sample Autocorrelations

The GENMOD Procedure

Bayesian Analysis

Posterior Autocorrelations				
Parameter	**Lag 1**	**Lag 5**	**Lag 10**	**Lag 50**
Intercept	0.3037	0.0152	0.0095	-0.0170
X1	0.3398	0.0025	0.0003	0.0052
X2	0.3036	0.0061	0.0003	-0.0062
X3	0.3489	0.0190	-0.0064	-0.0210
X4	0.2868	0.0213	0.0157	-0.0107
X5	0.2854	0.0108	-0.0288	-0.0012
X6	0.3078	0.0230	0.0073	0.0062

The p-values for the Geweke test statistics shown in Output 51.10.9 all indicate convergence of the MCMC. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more information about convergence diagnostics and their interpretation.

Output 51.10.9 Geweke Diagnostic Statistics

Geweke Diagnostics		
Parameter	**z**	**Pr > |z|**
Intercept	-0.6533	0.5135
X1	0.3418	0.7325
X2	0.3609	0.7182
X3	-0.3345	0.7380
X4	0.2851	0.7755
X5	-0.5266	0.5985
X6	1.1285	0.2591

The effective sample sizes for each parameter are shown in Output 51.10.10.

Output 51.10.10 Effective Sample Sizes

Effective Sample Sizes			
		Autocorrelation	
Parameter	ESS	Time	Efficiency
Intercept	4880.3	2.0491	0.4880
X1	4844.2	2.0643	0.4844
X2	5139.3	1.9458	0.5139
X3	4551.2	2.1972	0.4551
X4	4953.6	2.0187	0.4954
X5	5330.5	1.8760	0.5331
X6	4988.1	2.0048	0.4988

Trace, autocorrelation, and density plots for the seven model parameters are shown in Output 51.10.11 through Output 51.10.17. All indicate satisfactory convergence of the Markov chain.

Output 51.10.11 Diagnostic Plots for Intercept

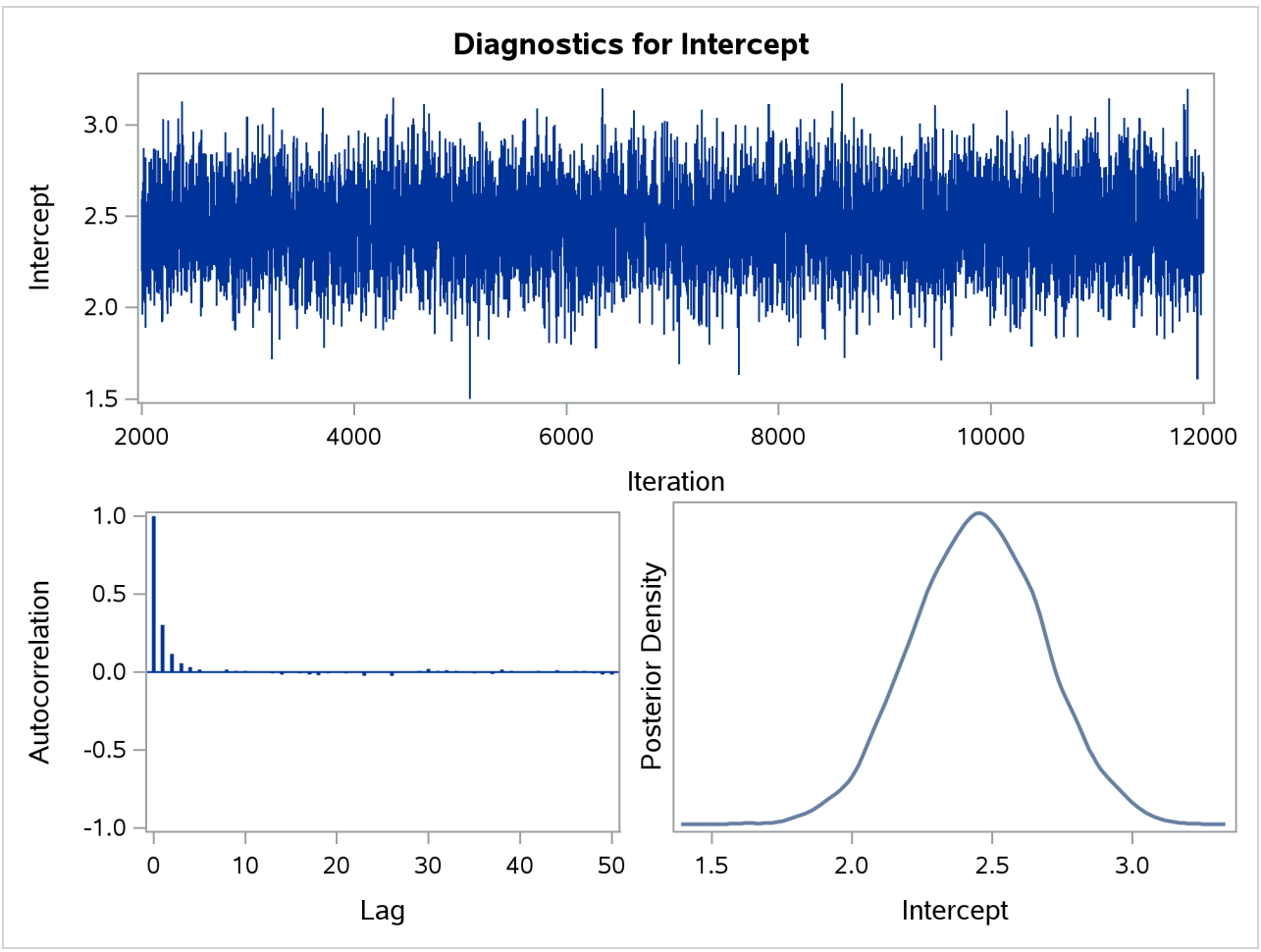

Output 51.10.12 Diagnostic Plots for X1

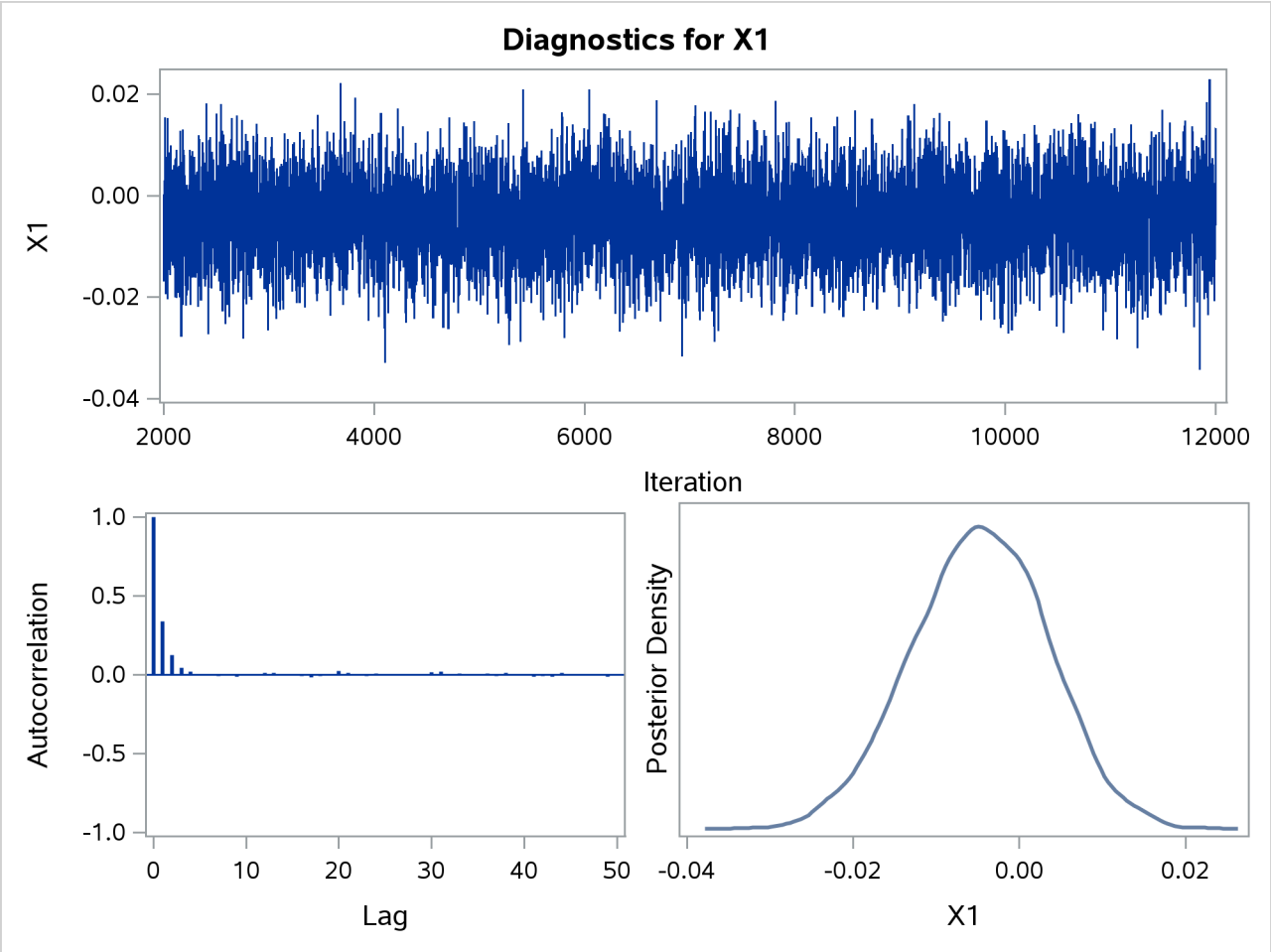

Output 51.10.13 Diagnostic Plots for X2

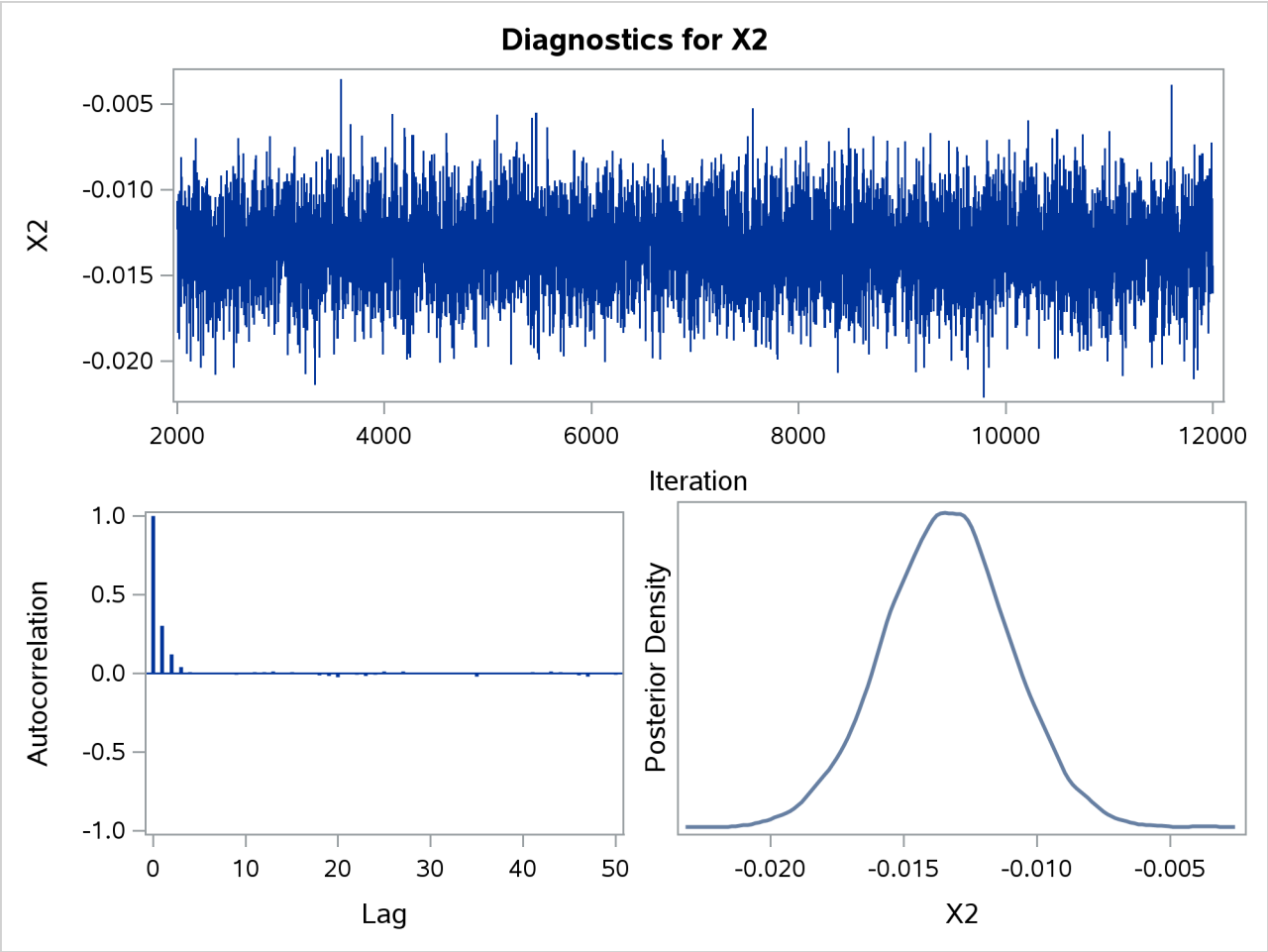

Output 51.10.14 Diagnostic Plots for X3

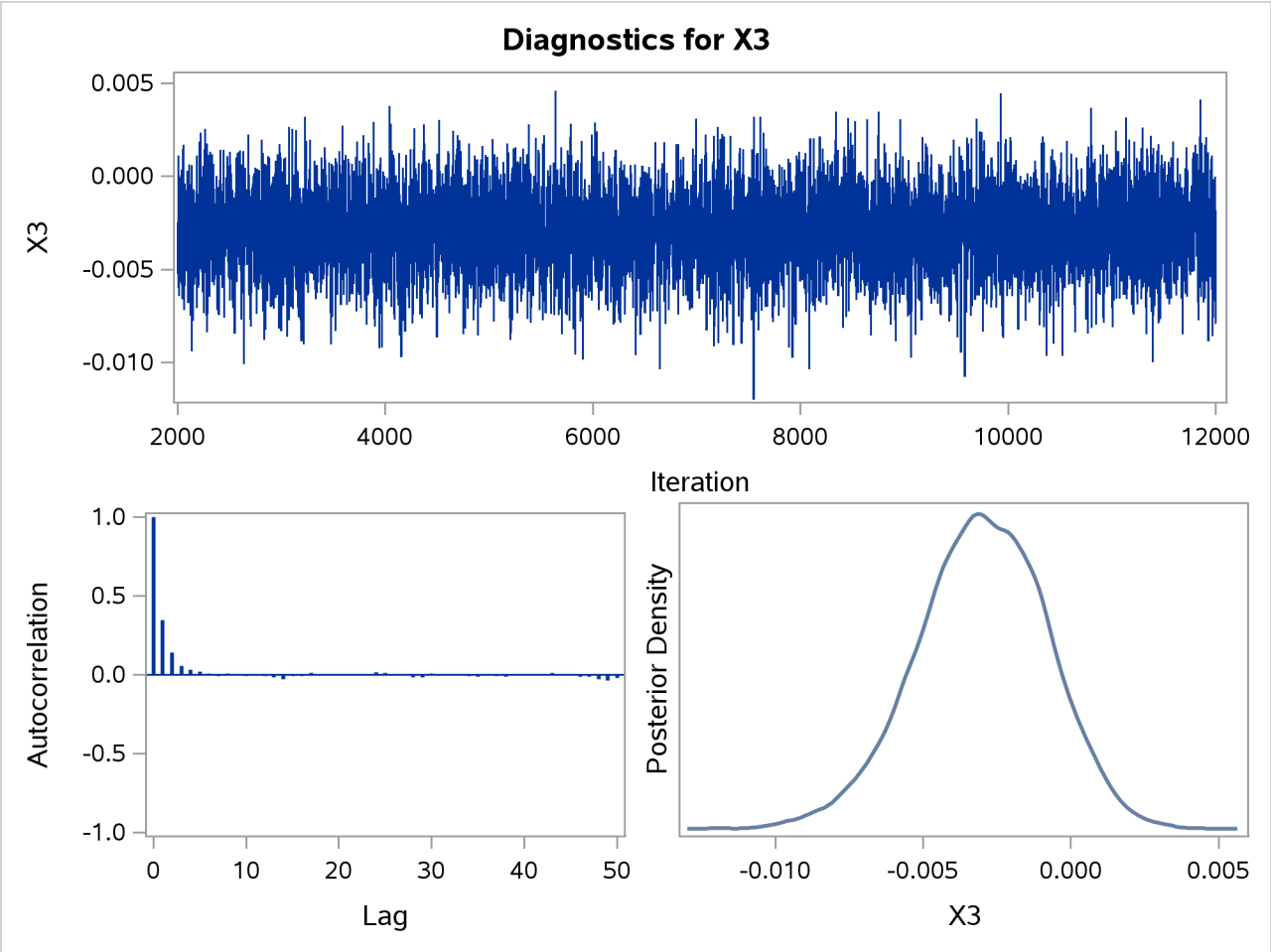

Output 51.10.15 Diagnostic Plots for X4

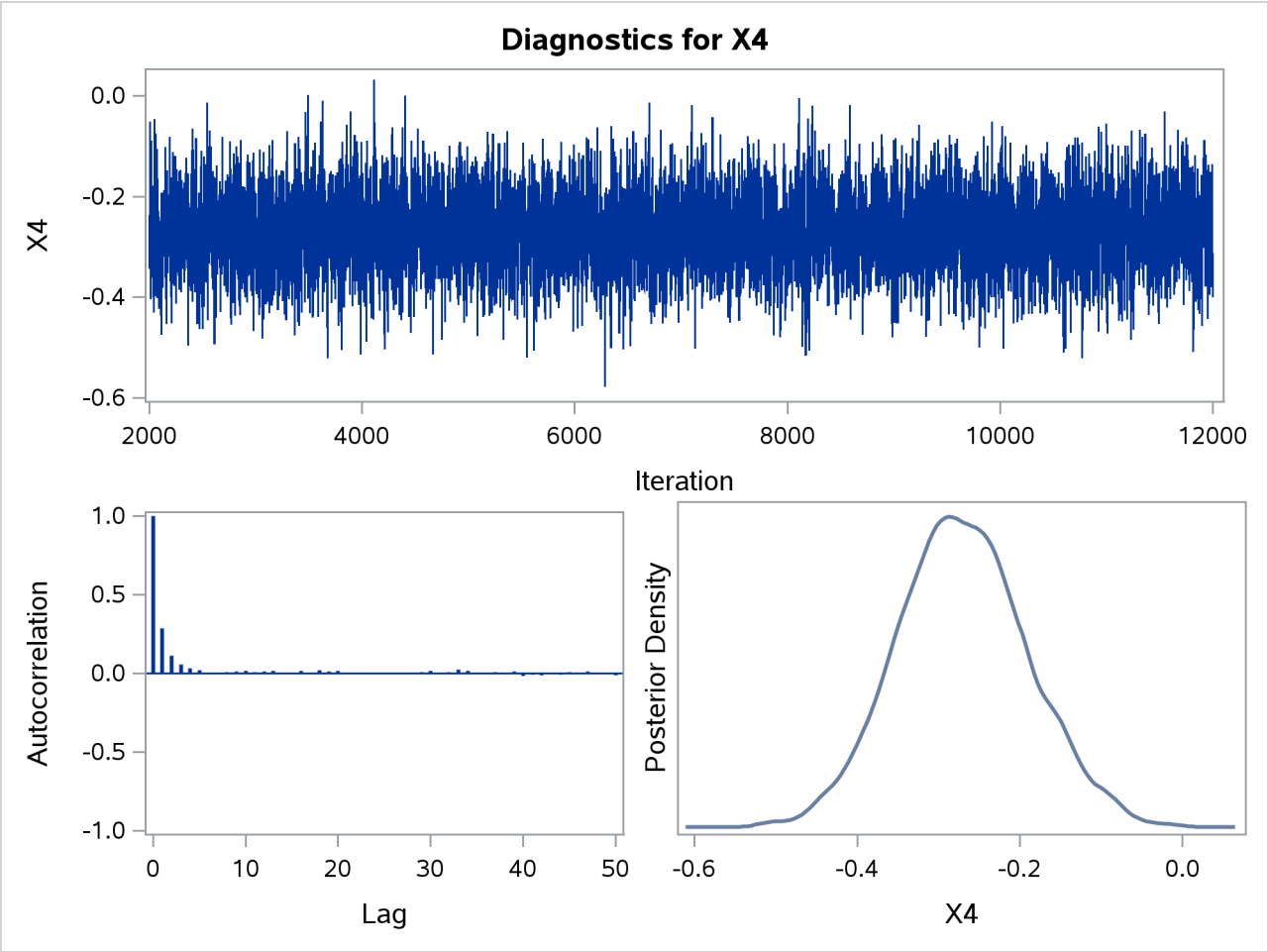

Output 51.10.16 Diagnostic Plots for X5

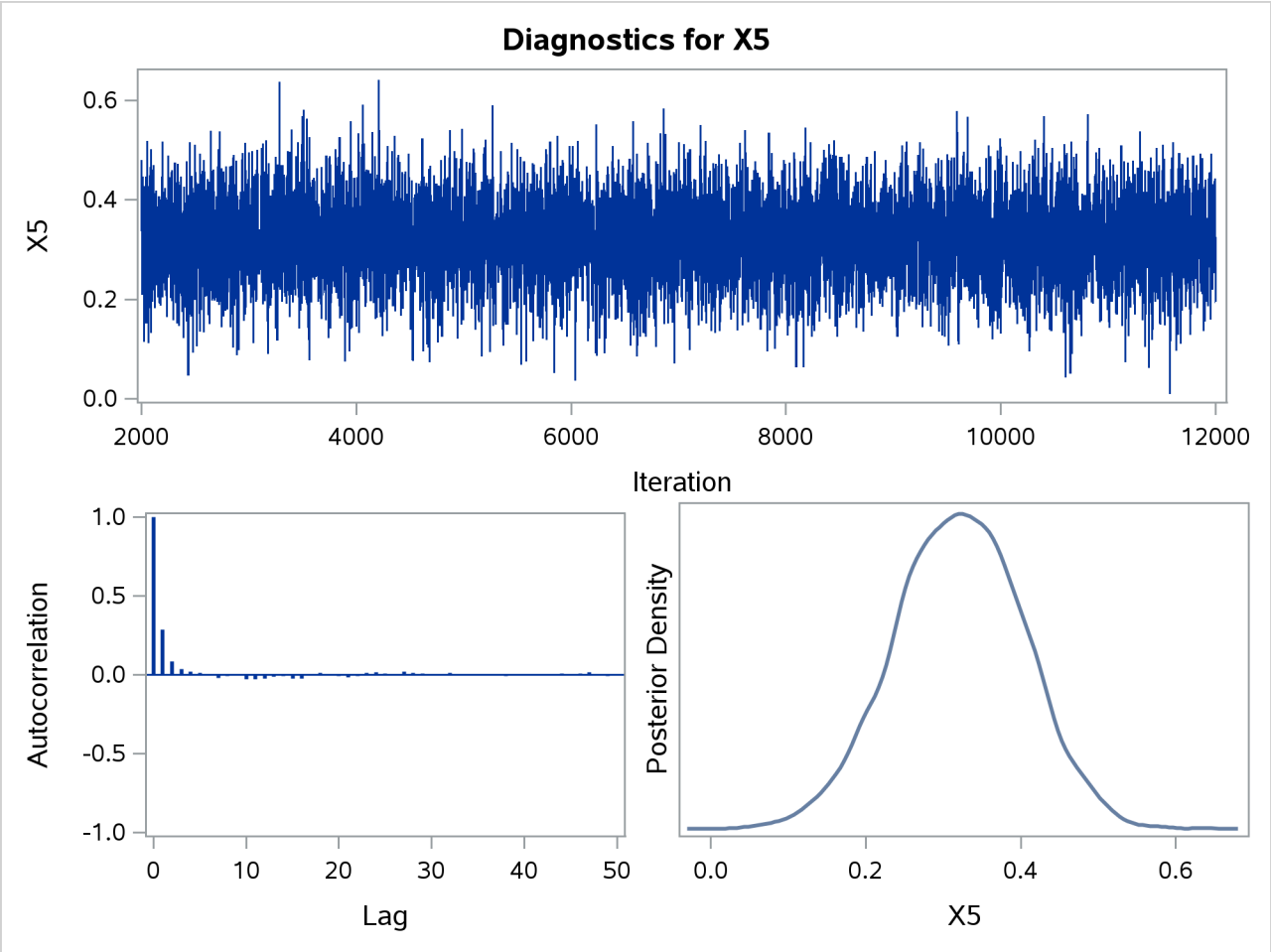

Output 51.10.17 Diagnostic Plots for X6

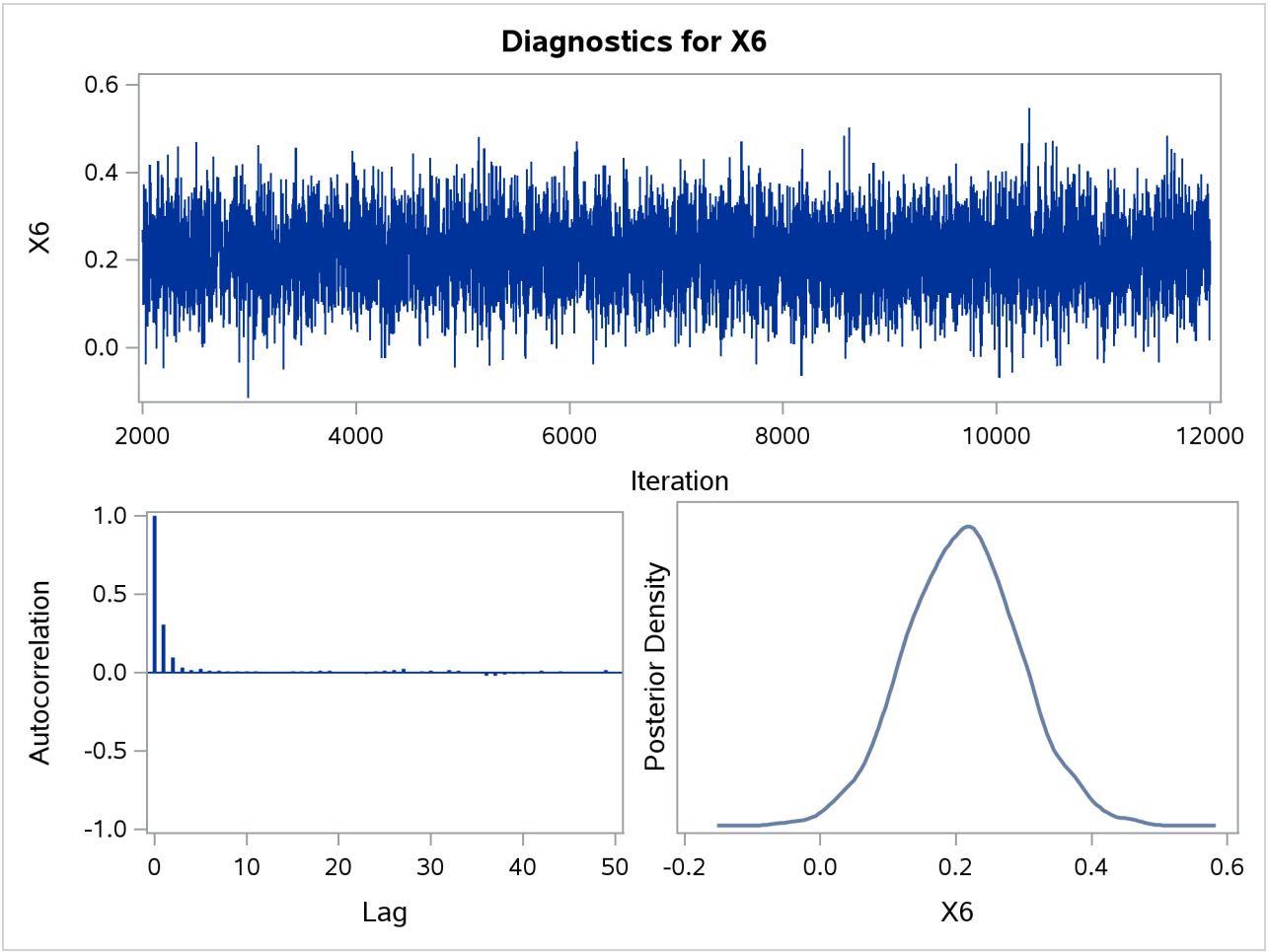

In order to illustrate the use of an informative prior distribution, suppose that researchers expect that a unit increase in body mass index (X1) will be associated with an increase in the mean number of nodes of between 10% and 20%, and they want to incorporate this prior knowledge in the Bayesian analysis. For log-linear models, the mean and linear predictor are related by $\log(\mu_i) = \boldsymbol{x}_i'\boldsymbol{\beta}$. If $\mathsf{X1}_1$ and $\mathsf{X1}_2$ are two values of body mass index, μ_1 and μ_2 are the two mean values, and all other covariates remain equal for the two values of X1, then

$$\frac{\mu_1}{\mu_2} = \exp(\beta(\mathsf{X1}_1 - \mathsf{X1}_2))$$

so that for a unit change in X1,

$$\frac{\mu_1}{\mu_2} = \exp(\beta)$$

If $1.1 \le \frac{\mu_1}{\mu_2} \le 1.2$, then $1.1 \le \exp(\beta) \le 1.2$, or $0.095 \le \beta \le 0.182$. This gives you guidance in specifying a prior distribution for the β for body mass index. Taking the mean of the prior normal distribution to be the

midrange of the values of β, and taking $\mu \pm 2\sigma$ to be the extremes of the range, an $N(0.1385, 0.0005)$ is the resulting prior distribution. The second analysis uses this informative normal prior distribution for the coefficient of X1 and uses independent noninformative normal priors with zero means and variances equal to 10^6 for the remaining model regression parameters.

In the following BAYES statement, COEFFPRIOR=NORMAL(INPUT=NormalPrior) specifies the normal prior distribution for the regression coefficients with means and variances contained in the data set NormalPrior.

An analysis is performed using PROC GENMOD to obtain Bayesian estimates of the regression coefficients by using the following SAS statements:

```
data NormalPrior;
   input _type_ $ Intercept X1-X6;
   datalines;
Var  1e6   0.0005      1e6      1e6      1e6      1e6      1e6
Mean 0.0   0.1385      0.0      0.0      0.0      0.0      0.0
;

proc genmod data=Liver;
   model Y = X1-X6 / dist=Poisson link=log;
   bayes seed=1 plots=none coeffprior=normal(input=NormalPrior);
run;
```

The prior distributions for the regression parameters are shown in Output 51.10.18.

Output 51.10.18 Regression Coefficient Priors

The GENMOD Procedure

Bayesian Analysis

Independent Normal Prior for Regression Coefficients		
Parameter	**Mean**	**Precision**
Intercept	0	1E-6
X1	0.1385	2000
X2	0	1E-6
X3	0	1E-6
X4	0	1E-6
X5	0	1E-6
X6	0	1E-6

Initial values for the MCMC are shown in Output 51.10.19. The initial values of the covariates are joint estimates of their posterior modes. The prior distribution for X1 is informative, so the initial value of X1 is further from the MLE than the rest of the covariates. Initial values for the rest of the covariates are close to their MLEs, since noninformative prior distributions were specified for them.

Output 51.10.19 MCMC Initial Values and Seeds

Initial Values of the Chain								
Chain	**Seed**	**Intercept**	**X1**	**X2**	**X3**	**X4**	**X5**	**X6**
1	1	2.14282	0.010595	-0.01434	-0.00301	-0.28062	0.334983	0.231213

Goodness-of-fit, summary, and interval statistics are shown in Output 51.10.20. Except for X1, the statistics shown in Output 51.10.20 are very similar to the previous statistics for noninformative priors shown in Output 51.10.4 through Output 51.10.7. The point estimate for X1 is now positive. This is expected because the prior distribution on β_1 is quite informative. The distribution reflects the belief that the coefficient is positive. The $N(0.1385, 0.0005)$ distribution places the majority of its probability density on positive values. As a result, the posterior density of β_1 places more likelihood on positive values than in the noninformative case.

Output 51.10.20 Fit Statistics

Fit Statistics	
DIC (smaller is better)	833.074
pD (effective number of parameters)	6.869

The GENMOD Procedure

Bayesian Analysis

Posterior Summaries

Parameter	N	Mean	Standard Deviation	Percentiles 25%	50%	75%
Intercept	10000	2.1419	0.2157	1.9965	2.1430	2.2894
X1	10000	0.0103	0.00684	0.00573	0.0104	0.0150
X2	10000	-0.0143	0.00233	-0.0159	-0.0142	-0.0127
X3	10000	-0.00318	0.00218	-0.00467	-0.00314	-0.00170
X4	10000	-0.2806	0.0800	-0.3336	-0.2793	-0.2266
X5	10000	0.3341	0.0832	0.2788	0.3341	0.3906
X6	10000	0.2333	0.0826	0.1774	0.2325	0.2880

Posterior Intervals

Parameter	Alpha	Equal-Tail Interval		HPD Interval	
Intercept	0.050	1.7225	2.5574	1.7293	2.5632
X1	0.050	-0.00344	0.0235	-0.00345	0.0234
X2	0.050	-0.0188	-0.00970	-0.0189	-0.00980
X3	0.050	-0.00757	0.00108	-0.00733	0.00121
X4	0.050	-0.4365	-0.1200	-0.4391	-0.1256
X5	0.050	0.1657	0.4966	0.1682	0.4987
X6	0.050	0.0695	0.3959	0.0725	0.3981

Example 51.11: Exact Poisson Regression

(View the complete code for this example (genmex11.sas) in the example repository.)

The following data, taken from Cox and Snell (1989, pp. 10–11), consists of the number, Notready, of ingots that are not ready for rolling, out of Total tested, for several combinations of heating time and soaking time:

```
data ingots;
   input Heat Soak Notready Total @@;
   lnTotal= log(Total);
   datalines;
7 1.0 0 10  14 1.0 0 31  27 1.0 1 56  51 1.0 3 13
7 1.7 0 17  14 1.7 0 43  27 1.7 4 44  51 1.7 0  1
7 2.2 0  7  14 2.2 2 33  27 2.2 0 21  51 2.2 0  1
7 2.8 0 12  14 2.8 0 31  27 2.8 1 22  51 4.0 0  1
7 4.0 0  9  14 4.0 0 19  27 4.0 1 16
;
```

The following invocation of PROC GENMOD fits an asymptotic (unconditional) Poisson regression model to the data. The variable Notready is specified as the response variable, and the continuous predictors Heat and Soak are defined in the CLASS statement as categorical predictors that use reference coding. Specifying the offset variable as lnTotal enables you to model the ratio Notready/Total.

```
proc genmod data=ingots;
   class Heat Soak / param=ref;
   model Notready=Heat Soak / offset=lnTotal dist=Poisson link=log;
   exact Heat Soak / joint estimate;
   exactoptions statustime=10;
run;
```

The EXACT statement is specified to additionally fit an exact conditional Poisson regression model. Specifying the lnTotal offset variable models the ratio Notready/Total; in this case, the Total variable contains the largest possible response value for each observation. The JOINT option produces a joint test for the significance of the covariates, along with the usual marginal tests. The ESTIMATE option produces exact parameter estimates for the covariates. The STATUSTIME=10 option is specified in the EXACTOPTIONS statement for monitoring the progress of the results; this example can take several minutes to complete due to the JOINT option. If you run out of memory, see the SAS Companion for your system for information about how to increase the available memory.

The "Criteria For Assessing Goodness Of Fit" table is displayed in Output 51.11.1. Comparing the deviance of 10.9363 to an asymptotic chi-square distribution with 11 degrees of freedom, you find that the p-value is 0.449. This indicates that the specified model fits the data reasonably well.

Output 51.11.1 Unconditional Goodness of Fit Criteria

The GENMOD Procedure

Criteria For Assessing Goodness Of Fit			
Criterion	**DF**	**Value**	**Value/DF**
Deviance	11	10.9363	0.9942
Scaled Deviance	11	10.9363	0.9942
Pearson Chi-Square	11	9.3722	0.8520
Scaled Pearson X2	11	9.3722	0.8520
Log Likelihood		-7.2408	
Full Log Likelihood		-12.9038	
AIC (smaller is better)		41.8076	
AICC (smaller is better)		56.2076	
BIC (smaller is better)		49.3631	

From the "Analysis Of Parameter Estimates" table in Output 51.11.2, you can see that only two of the Heat parameters are deemed significant. Looking at the standard errors, you can see that the unconditional analysis had convergence difficulties with the Heat=7 parameter (Standard Error=264324.6), which means you cannot fit this unconditional Poisson regression model to this data.

Output 51.11.2 Unconditional Maximum Likelihood Parameter Estimates

Analysis Of Maximum Likelihood Parameter Estimates							
Parameter		**DF**	**Estimate**	**Standard Error**	**Wald 95% Confidence Limits**		**Wald Chi-Square**	**Pr > ChiSq**
Intercept		1	-1.5700	1.1657	-3.8548	0.7147	1.81	0.1780
Heat	7	1	-27.6129	264324.6	-518094	518039.0	0.00	0.9999
Heat	14	1	-3.0107	1.0025	-4.9756	-1.0458	9.02	0.0027
Heat	27	1	-1.7180	0.7691	-3.2253	-0.2106	4.99	0.0255
Soak	1	1	-0.2454	1.1455	-2.4906	1.9998	0.05	0.8304
Soak	1.7	1	0.5572	1.1217	-1.6412	2.7557	0.25	0.6193
Soak	2.2	1	0.4079	1.2260	-1.9951	2.8109	0.11	0.7394
Soak	2.8	1	-0.1301	1.4234	-2.9199	2.6597	0.01	0.9272
Scale		0	1.0000	0.0000	1.0000	1.0000		

Note: The scale parameter was held fixed.

Following the output from the asymptotic analysis, the exact conditional Poisson regression results are displayed, as shown in Output 51.11.3.

Output 51.11.3 Exact Tests

The GENMOD Procedure

Exact Conditional Analysis

Exact Conditional Tests				
			p-Value	
Effect	Test	Statistic	Exact	Mid
Joint	Score	18.3665	0.0137	0.0137
	Probability	1.294E-6	0.0471	0.0471
Heat	Score	15.8259	0.0023	0.0022
	Probability	0.000175	0.0063	0.0062
Soak	Score	1.4612	0.8683	0.8646
	Probability	0.00735	0.8176	0.8139

The Joint test in the "Conditional Exact Tests" table in Output 51.11.3 is produced by specifying the JOINT option in the EXACT statement. The *p*-values for this test indicate that the parameters for Heat and Soak are jointly significant as explanatory effects in the model. If the Heat variable is the only explanatory variable in your model, then the rows of this table labeled as "Heat" show the joint significance of all the Heat effect parameters in that reduced model. In this case, a model that contains only the Heat parameters still explains a significant amount of the variability; however, you can see that a model that contains only the Soak parameters would not be significant.

The "Exact Parameter Estimates" table in Output 51.11.4 displays parameter estimates and tests of significance for the levels of the CLASS variables. Again, the Heat=7 parameter has some difficulties; however, in the exact analysis, a *median unbiased estimate* is computed for the parameter instead of a maximum likelihood estimate. The confidence limits show that the Heat variable contains some explanatory power, while the categorical Soak variable is insignificant and can be dropped from the model.

Output 51.11.4 Exact Parameter Estimates

Exact Parameter Estimates						
				95% Confidence Limits		
Parameter		Estimate	Standard Error			p-Value
Heat	7	-2.7552 *	.	-Infinity	-0.7864	0.0099
Heat	14	-3.0255	1.0128	-5.7450	-0.6194	0.0113
Heat	27	-1.7846	0.8065	-3.6779	0.2260	0.0844
Soak	1	-0.3231	1.1717	-2.8673	3.6754	1.0000
Soak	1.7	0.5375	1.1284	-1.8056	4.4588	1.0000
Soak	2.2	0.4035	1.2347	-2.5785	4.5054	1.0000
Soak	2.8	-0.1661	1.4214	-4.5490	4.2168	1.0000

Note: * indicates a median unbiased estimate and a one-sided p-value.

NOTE: If you want to make predictions from the exact results, you can obtain an estimate for the intercept parameter by specifying the INTERCEPT keyword in the EXACT statement. You should also remove the JOINT option to reduce the amount of time and memory consumed.

Example 51.12: Tweedie Regression

(View the complete code for this example (genmex12.sas) in the example repository.)

The following SAS statements simulate 250 observations, which are based on an underlying Tweedie generalized linear model (GLM) that exploits its connection with the compound Poisson distribution. A natural logarithm link function is assumed for modeling the response variable (yTweedie), and there are five categorical variables (C1–C5), each of which has four numerical levels and two continuous variables (D1 and D2). By design, two of the categorical variables, C3 and C4, and one of the two continuous variables, D2, have no effect on the response. The dispersion parameter is set to 0.5, and the power parameter is set to 1.5.

```
%let nObs = 250;
%let nClass = 5;
%let nLevs = 4;
%let seed = 100;

data tmp1;
   array c{&nClass};

   keep c1-c&nClass yTweedie d1 d2;

   /* Tweedie parms */
   phi=0.5;
   p=1.5;

   do i=1 to &nObs;

      do j=1 to &nClass;
         c{j} = int(ranuni(1)*&nLevs);
      end;

      d1 = ranuni(&seed);
      d2 = ranuni(&seed);

      xBeta   =  0.5*((c2<2) - 2*(c1=1) + 0.5*c&nClass + 0.05*d1);
      mu      =  exp(xBeta);

      /* Poisson distributions parms */
      lambda = mu**(2-p)/(phi*(2-p));
      /* Gamma distribution parms */
      alpha = (2-p)/(p-1);
      gamma = phi*(p-1)*(mu**(p-1));

      rpoi = ranpoi(&seed,lambda);
      if rpoi=0 then yTweedie=0;
      else do;
         yTweedie=0;
         do j=1 to rpoi;
         yTweedie = yTweedie + rangam(&seed,alpha);
         end;
         yTweedie = yTweedie * gamma;
      end;
```

```
      output;
   end;
run;
```

The following SAS statements invoke PROC GENMOD to fit the Tweedie GLM with the log link using all of the categorical and continuous variables. A Type III analysis is requested by the TYPE3 option in the MODEL statement.

```
proc genmod data=tmp1;
   class C1-C5;
   model yTweedie = C1-C5 D1 D2 / dist=Tweedie type3;
run;
```

The "Criteria For Assessing Goodness Of Fit" table is displayed in Output 51.12.1. The scaled Pearson χ^2 is close to 1, indicating that the specified model fits the data well.

Output 51.12.1 Tweedie Goodness of Fit Criteria

The GENMOD Procedure

Criteria For Assessing Goodness Of Fit			
Criterion	**DF**	**Value**	**Value/DF**
Pearson Chi-Square	232	101.9124	0.4393
Scaled Pearson X2	232	251.5826	1.0844
Log Likelihood		-297.2106	
Full Log Likelihood		-297.2106	
AIC (smaller is better)		634.4212	
AICC (smaller is better)		638.0893	
BIC (smaller is better)		704.8504	

The "LR Statistics For Type 3 Analysis" table is displayed in Output 51.12.2. As expected, the p-values for C3, C4, and d2 are not statistically significant at the 5% level.

Output 51.12.2 Type III Analysis of Covariate Effects

LR Statistics For Type 3 Analysis			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
c1	3	85.46	<.0001
c2	3	48.18	<.0001
c3	3	0.56	0.9050
c4	3	9.38	0.0247
c5	3	47.76	<.0001
d1	1	0.00	0.9595
d2	1	1.31	0.2518

You can fix the power parameter for fitting the Tweedie GLM by using the P= option. The following SAS statements fit the model for C1, C2 and D1, while holding the power parameter at 1.5:

```
proc genmod data=tmp1;
   class C1 C2;
   model yTweedie = C1 C2 D1 / dist=Tweedie(p=1.5) type3;
run;
```

The parameter estimates are displayed in Output 51.12.3.

Output 51.12.3 Tweedie Maximum Likelihood Parameter Estimates

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter		DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		1	0.3440	0.1347	0.0801	0.6080	6.53	0.0106
c1	0	1	-0.0722	0.1101	-0.2880	0.1436	0.43	0.5120
c1	1	1	-0.8952	0.1196	-1.1296	-0.6607	56.01	<.0001
c1	2	1	0.0770	0.1073	-0.1334	0.2873	0.51	0.4733
c1	3	0	0.0000	0.0000	0.0000	0.0000	.	.
c2	0	1	0.6138	0.1161	0.3862	0.8414	27.93	<.0001
c2	1	1	0.5103	0.1150	0.2849	0.7356	19.70	<.0001
c2	2	1	0.1001	0.1215	-0.1380	0.3381	0.68	0.4099
c2	3	0	0.0000	0.0000	0.0000	0.0000	.	.
d1		1	-0.0211	0.1493	-0.3136	0.2714	0.02	0.8876
Dispersion		1	0.4951	0.0398	0.4172	0.5731		
Power		0	1.5000	0.0000	1.5000	1.5000		

Note: The Tweedie dispersion parameter was estimated by maximum likelihood.
Note: The Tweedie power parameter was held fixed.

References

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Aitkin, M., Anderson, D. A., Francis, B., and Hinde, J. (1989). *Statistical Modelling in GLIM*. Oxford: Oxford Science Publications.

Akaike, H. (1979). "A Bayesian Extension of the Minimum AIC Procedure of Autoregressive Model Fitting." *Biometrika* 66:237–242.

Akaike, H. (1981). "Likelihood of a Model and Information Criteria." *Journal of Econometrics* 16:3–14.

Boos, D. (1992). "On Generalized Score Tests." *American Statistician* 46:327–333.

Cameron, A. C., and Trivedi, P. K. (1998). *Regression Analysis of Count Data*. Cambridge: Cambridge University Press.

Carey, V., Zeger, S. L., and Diggle, P. J. (1993). "Modelling Multivariate Binary Data with Alternating Logistic Regressions." *Biometrika* 80:517–526.

Chang, I.-M., Gelman, R., and Pagano, M. (1982). "Corrected Group Prognostic Curves and Summary Statistics." *Journal of Chronic Diseases* 35:669–674.

Collett, D. (2003). *Modelling Binary Data*. 2nd ed. London: Chapman & Hall.

Cook, R. D., and Weisberg, S. (1982). *Residuals and Influence in Regression*. New York: Chapman & Hall.

Cox, D. R., and Snell, E. J. (1989). *The Analysis of Binary Data*. 2nd ed. London: Chapman & Hall.

Davison, A. C., and Snell, E. J. (1991). "Residuals and Diagnostics." In *Statistical Theory and Modelling*, edited by D. V. Hinkley, N. Reid, and E. J. Snell, 83–106. London: Chapman & Hall.

Diggle, P. J., Liang, K.-Y., and Zeger, S. L. (1994). *Analysis of Longitudinal Data*. Oxford: Clarendon Press.

Dobson, A. (1990). *An Introduction to Generalized Linear Models*. London: Chapman & Hall.

Dunn, P. K., and Smyth, G. K. (2005). "Series Evaluation of Tweedie Exponential Dispersion Model Densities." *Statistics and Computing* 15:267–280.

Dunn, P. K., and Smyth, G. K. (2008). "Series Evaluation of Tweedie Exponential Dispersion Model Densities by Fourier Inversion." *Statistics and Computing* 18:73–86.

Firth, D. (1991). "Generalized Linear Models." In *Statistical Theory and Modelling*, edited by D. V. Hinkley, N. Reid, and E. J. Snell, 55–82. London: Chapman & Hall.

Fischl, M. A., Richman, D. D., and Hansen, N. (1990). "The Safety and Efficacy of Zidovudine (AZT) in the Treatment of Subjects with Mildly Symptomatic Human Immunodeficiency Virus Type I (HIV) Infection." *Annals of Internal Medicine* 112:727–737.

Gamerman, D. (1997). "Sampling from the Posterior Distribution in Generalized Linear Models." *Statistics and Computing* 7:57–68.

Gilks, W. R. (2003). "Adaptive Metropolis Rejection Sampling (ARMS)." Software from MRC Biostatistics Unit, Cambridge, UK. http://www.maths.leeds.ac.uk/~wally.gilks/adaptive.rejection/web_page/Welcome.html.

Gilks, W. R., Best, N. G., and Tan, K. K. C. (1995). "Adaptive Rejection Metropolis Sampling within Gibbs Sampling." *Journal of the Royal Statistical Society, Series C* 44:455–472.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996). *Markov Chain Monte Carlo in Practice*. London: Chapman & Hall.

Gilks, W. R., and Wild, P. (1992). "Adaptive Rejection Sampling for Gibbs Sampling." *Journal of the Royal Statistical Society, Series C* 41:337–348.

Hardin, J. W., and Hilbe, J. M. (2003). *Generalized Estimating Equations*. Boca Raton, FL: Chapman & Hall/CRC.

Hilbe, J. M. (1994). "Log Negative Binomial Regression Using the GENMOD Procedure in SAS/STAT Software." In *Proceedings of the Nineteenth Annual SAS Users Group International Conference*, 1199–1204. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI94/Sugi-94-205%20Hilbe.pdf.

Hilbe, J. M. (2007). *Negative Binomial Regression*. New York: Cambridge University Press.

Hilbe, J. M. (2009). *Logistic Regression Models*. London: Chapman & Hall/CRC.

Hirji, K. F., Mehta, C. R., and Patel, N. R. (1987). "Computing Distributions for Exact Logistic Regression." *Journal of the American Statistical Association* 82:1110–1117.

Hougaard, P. (1986). "Survival Models for Heterogeneous Populations Derived from Stable Distributions." *Biometrika* 73:387–396.

Ibrahim, J. G., Chen, M.-H., and Lipsitz, S. R. (1999). "Monte Carlo EM for Missing Covariates in Parametric Regression Models." *Biometrics* 55:591–596.

Ibrahim, J. G., Chen, M.-H., and Sinha, D. (2001). *Bayesian Survival Analysis*. New York: Springer-Verlag.

Ibrahim, J. G., and Laud, P. W. (1991). "On Bayesian Analysis of Generalized Linear Models Using Jeffreys' Prior." *Journal of the American Statistical Association* 86:981–986.

Lambert, D. (1992). "Zero-Inflated Poisson Regression with an Application to Defects in Manufacturing." *Technometrics* 34:1–14.

Lane, P. W., and Nelder, J. A. (1982). "Analysis of Covariance and Standardization as Instances of Prediction." *Biometrics* 38:613–621.

Lawless, J. F. (1987). "Negative Binomial and Mixed Poisson Regression." *Canadian Journal of Statistics* 15:209–225.

Lawless, J. F. (2003). *Statistical Model and Methods for Lifetime Data*. 2nd ed. New York: John Wiley & Sons.

Liang, K.-Y., and Zeger, S. L. (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika* 73:13–22.

Lin, D. Y., Wei, L. J., and Ying, Z. (2002). "Model-Checking Techniques Based on Cumulative Residuals." *Biometrics* 58:1–12.

Lipsitz, S. R., Fitzmaurice, G. M., Orav, E. J., and Laird, N. M. (1994). "Performance of Generalized Estimating Equations in Practical Situations." *Biometrics* 50:270–278.

Lipsitz, S. R., Kim, K., and Zhao, L. (1994). "Analysis of Repeated Categorical Data Using Generalized Estimating Equations." *Statistics in Medicine* 13:1149–1163.

Littell, R. C., Freund, R. J., and Spector, P. C. (1991). *SAS System for Linear Models*. 3rd ed. Cary, NC: SAS Institute Inc.

Long, J. S. (1997). *Regression Models for Categorical and Limited Dependent Variables*. Thousand Oaks, CA: Sage Publications.

McCullagh, P. (1983). "Quasi-likelihood Functions." *Annals of Statistics* 11:59–67.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Mehta, C. R., Patel, N. R., and Senchaudhuri, P. (1992). "Exact Stratified Linear Rank Tests for Ordered Categorical and Binary Data." *Journal of Computational and Graphical Statistics* 1:21–40.

Miller, M. E., Davis, C. S., and Landis, J. R. (1993). "The Analysis of Longitudinal Polytomous Data: Generalized Estimating Equations and Connections with Weighted Least Squares." *Biometrics* 49:1033–1044.

Muller, K. E., and Fetterman, B. A. (2002). *Regression and ANOVA: An Integrated Approach Using SAS Software*. Cary, NC: SAS Institute Inc.

Myers, R. H., Montgomery, D. C., and Vining, G. G. (2002). *Generalized Linear Models with Applications in Engineering and the Sciences*. New York: John Wiley & Sons.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Nelson, W. (1982). *Applied Life Data Analysis*. New York: John Wiley & Sons.

Neter, J., Kutner, M. H., Nachtsheim, C. J., and Wasserman, W. (1996). *Applied Linear Statistical Models*. 4th ed. Chicago: Irwin.

Pan, W. (2001). "Akaike's Information Criterion in Generalized Estimating Equations." *Biometrics* 57:120–125.

Pregibon, D. (1981). "Logistic Regression Diagnostics." *Annals of Statistics* 9:705–724.

Preisser, J. S., and Qaqish, B. F. (1996). "Deletion Diagnostics for Generalised Estimating Equations." *Biometrika* 83:551–562.

Rao, C. R. (1973). *Linear Statistical Inference and Its Applications*. 2nd ed. New York: John Wiley & Sons.

Rotnitzky, A., and Jewell, N. P. (1990). "Hypothesis Testing of Regression Parameters in Semiparametric Generalized Linear Models for Cluster Correlated Data." *Biometrika* 77:485–497.

Royall, R. M. (1986). "Model Robust Inference Using Maximum Likelihood Estimators." *International Statistical Review* 54:221–226.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Simonoff, J. S. (2003). *Analyzing Categorical Data*. New York: Springer-Verlag.

Smyth, G. K. (1996). "Regression Analysis of Quantity Data with Exact Zeros." In *Proceedings of the Second Australia-Japan Workshop on Stochastic Models in Engineering, Technology, and Management*, edited by R. J. Wilson, S. Osaki, and D. N. P. Murthy, 572–580. Queensland, Australia: Technology Management Centre, University of Queensland.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and Van der Linde, A. (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society, Series B* 64:583–616. With discussion.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2000). *Categorical Data Analysis Using the SAS System*. 2nd ed. Cary, NC: SAS Institute Inc.

Thall, P. F., and Vail, S. C. (1990). "Some Covariance Models for Longitudinal Count Data with Overdispersion." *Biometrics* 46:657–671.

Tweedie, M. C. K. (1984). "An Index Which Distinguishes between Some Important Exponential Families." In *Statistics: Applications and New Directions—Proceedings of the Indian Statistical Institute Golden Jubilee International Conference*, edited by J. K. Ghosh and J. Roy, 579–604. Calcutta: Indian Statistical Institute.

Ware, J. H., Dockery, S. A., III, Speizer, F. E., and Ferris, B. G., Jr. (1984). "Passive Smoking, Gas Cooking, and Respiratory Health of Children Living in Six Cities." *American Review of Respiratory Diseases* 129:366–374.

White, H. (1982). "Maximum Likelihood Estimation of Misspecified Models." *Econometrica* 50:1–25.

Williams, D. A. (1987). "Generalized Linear Model Diagnostics Using the Deviance and Single Case Deletions." *Journal of the Royal Statistical Society, Series C* 36:181–191.

Zeger, S. L., Liang, K.-Y., and Albert, P. S. (1988). "Models for Longitudinal Data: A Generalized Estimating Equation Approach." *Biometrics* 44:1049–1060.

Chapter 52
The GLIMMIX Procedure

Contents

Overview: GLIMMIX Procedure **3778**
- Basic Features 3779
- Assumptions 3780
- Notation for the Generalized Linear Mixed Model 3781
  - The Basic Model 3781
  - G-Side and R-Side Random Effects and Covariance Structures 3781
  - Relationship with Generalized Linear Models 3782
- PROC GLIMMIX Contrasted with Other SAS Procedures 3783

Getting Started: GLIMMIX Procedure **3784**
- Logistic Regressions with Random Intercepts 3784

Syntax: GLIMMIX Procedure **3790**
- PROC GLIMMIX Statement 3791
- BY Statement 3818
- CLASS Statement 3819
- CODE Statement 3820
- CONTRAST Statement 3820
- COVTEST Statement 3824
- EFFECT Statement 3832
- ESTIMATE Statement 3832
- FREQ Statement 3838
- ID Statement 3838
- LSMEANS Statement 3839
- LSMESTIMATE Statement 3852
- MARGINS Statement 3860
- MODEL Statement 3866
  - Response Variable Options 3868
  - Model Options 3869
- NLOPTIONS Statement 3879
- OUTPUT Statement 3880
- PARMS Statement 3885
- RANDOM Statement 3890
- SLICE Statement 3911
- STORE Statement 3911
- WEIGHT Statement 3911
- Programming Statements 3911
- User-Defined Link or Variance Function 3913

Implied Variance Functions . . . 3913
Automatic Variables . . . 3914
Details: GLIMMIX Procedure . . . **3917**
Generalized Linear Models Theory . . . 3917
Maximum Likelihood . . . 3917
Scale and Dispersion Parameters . . . 3920
Quasi-likelihood for Independent Data . . . 3921
Effects of Adding Overdispersion . . . 3922
Generalized Linear Mixed Models Theory . . . 3922
Model or Integral Approximation . . . 3922
Pseudo-likelihood Estimation Based on Linearization . . . 3924
Maximum Likelihood Estimation Based on Laplace Approximation . . . 3929
Maximum Likelihood Estimation Based on Adaptive Quadrature . . . 3932
Aspects Common to Adaptive Quadrature and Laplace Approximation . . . 3935
Notes on Bias of Estimators . . . 3937
Pseudo-likelihood Estimation for Weighted Multilevel Models . . . 3938
GLM Mode or GLMM Mode . . . 3941
Statistical Inference for Covariance Parameters . . . 3942
The Likelihood Ratio Test . . . 3942
One- and Two-Sided Testing, Mixture Distributions . . . 3943
Handling the Degenerate Distribution . . . 3945
Wald Versus Likelihood Ratio Tests . . . 3945
Confidence Bounds Based on Likelihoods . . . 3945
Degrees of Freedom Methods . . . 3949
Between-Within Degrees of Freedom Approximation . . . 3949
Containment Degrees of Freedom Approximation . . . 3949
Satterthwaite Degrees of Freedom Approximation . . . 3949
Kenward-Roger Degrees of Freedom Approximation . . . 3951
Empirical Covariance (“Sandwich”) Estimators . . . 3952
Residual-Based Estimators . . . 3952
Design-Adjusted MBN Estimator . . . 3953
Exploring and Comparing Covariance Matrices . . . 3955
Predictive Margins Compared with LS-Means . . . 3956
Processing by Subjects . . . 3957
Radial Smoothing Based on Mixed Models . . . 3959
From Penalized Splines to Mixed Models . . . 3959
Knot Selection . . . 3960
Odds and Odds Ratio Estimation . . . 3965
The Odds Ratio Estimates Table . . . 3966
Odds or Odds Ratio . . . 3968
Odds Ratios in Multinomial Models . . . 3969
Parameterization of Generalized Linear Mixed Models . . . 3970
Intercept . . . 3970
Interaction Effects . . . 3970

Nested Effects . . . 3970
Implications of the Non-Full-Rank Parameterization . . . 3971
Missing Level Combinations . . . 3971
Notes on the EFFECT Statement . . . 3971
Positional and Nonpositional Syntax for Contrast Coefficients . . . 3973
Response-Level Ordering and Referencing . . . 3976
Comparing the GLIMMIX and MIXED Procedures . . . 3977
Singly or Doubly Iterative Fitting . . . 3979
Default Estimation Techniques . . . 3981
Default Output . . . 3982
Model Information . . . 3982
Class Level Information . . . 3982
Number of Observations . . . 3982
Response Profile . . . 3983
Dimensions . . . 3983
Optimization Information . . . 3983
Iteration History . . . 3984
Convergence Status . . . 3984
Fit Statistics . . . 3985
Covariance Parameter Estimates . . . 3986
Type III Tests of Fixed Effects . . . 3986
Notes on Output Statistics . . . 3986
ODS Table Names . . . 3988
ODS Graphics . . . 3990
ODS Graph Names . . . 3990
Diagnostic Plots . . . 3992
Graphics for LS-Mean Comparisons . . . 3997
Examples: GLIMMIX Procedure . . . **4008**
Example 52.1: Binomial Counts in Randomized Blocks . . . 4008
Example 52.2: Mating Experiment with Crossed Random Effects . . . 4018
Example 52.3: Smoothing Disease Rates; Standardized Mortality Ratios . . . 4026
Example 52.4: Quasi-likelihood Estimation for Proportions with Unknown Distribution 4035
Example 52.5: Joint Modeling of Binary and Count Data . . . 4043
Example 52.6: Radial Smoothing of Repeated Measures Data . . . 4050
Example 52.7: Isotonic Contrasts for Ordered Alternatives . . . 4062
Example 52.8: Adjusted Covariance Matrices of Fixed Effects . . . 4063
Example 52.9: Testing Equality of Covariance and Correlation Matrices . . . 4068
Example 52.10: Multiple Trends Correspond to Multiple Extrema in Profile Likelihoods 4076
Example 52.11: Maximum Likelihood in Proportional Odds Model with Random Effects 4081
Example 52.12: Fitting a Marginal (GEE-Type) Model . . . 4087
Example 52.13: Response Surface Comparisons with Multiplicity Adjustments . . . 4092
Example 52.14: Generalized Poisson Mixed Model for Overdispersed Count Data . . 4100
Example 52.15: Comparing Multiple B-Splines . . . 4107
Example 52.16: Diallel Experiment with Multimember Random Effects . . . 4114

Example 52.17: Linear Inference Based on Summary Data 4116
Example 52.18: Weighted Multilevel Model for Survey Data 4122
Example 52.19: Quadrature Method for Multilevel Model 4126
Example 52.20: Comparison of Predictive Margins in Multicenter Study 4129
References **4134**

Overview: GLIMMIX Procedure

The GLIMMIX procedure fits statistical models to data with correlations or nonconstant variability and where the response is not necessarily normally distributed. These models are known as generalized linear mixed models (GLMM).

GLMMs, like linear mixed models, assume normal (Gaussian) random effects. Conditional on these random effects, data can have any distribution in the exponential family. The exponential family comprises many of the elementary discrete and continuous distributions. The binary, binomial, Poisson, and negative binomial distributions, for example, are discrete members of this family. The normal, beta, gamma, and chi-square distributions are representatives of the continuous distributions in this family. In the absence of random effects, the GLIMMIX procedure fits generalized linear models (fit by the GENMOD procedure).

GLMMs are useful for the following applications:

- estimating trends in disease rates
- modeling CD4 counts in a clinical trial over time
- modeling the proportion of infected plants on experimental units in a design with randomly selected treatments or randomly selected blocks
- predicting the probability of high ozone levels in counties
- modeling skewed data over time
- analyzing customer preference
- joint modeling of multivariate outcomes

Such data often display correlations among some or all observations as well as nonnormality. The correlations can arise from repeated observation of the same sampling units, shared random effects in an experimental design, spatial (temporal) proximity, multivariate observations, and so on.

The GLIMMIX procedure does not fit hierarchical models with nonnormal random effects. With the GLIMMIX procedure you select the distribution of the response variable conditional on normally distributed random effects.

For more information about the differences between the GLIMMIX procedure and SAS procedures that specialize in certain subsets of the GLMM models, see the section "PROC GLIMMIX Contrasted with Other SAS Procedures" on page 3783.

Basic Features

The GLIMMIX procedure enables you to specify a generalized linear mixed model and to perform confirmatory inference in such models. The syntax is similar to that of the MIXED procedure and includes CLASS, MODEL, and RANDOM statements. For instructions on how to specify PROC MIXED REPEATED effects with PROC GLIMMIX, see the section "Comparing the GLIMMIX and MIXED Procedures" on page 3977. The following are some of the basic features of PROC GLIMMIX.

- SUBJECT= and GROUP= options, which enable blocking of variance matrices and parameter heterogeneity
- choice of linearization approach or integral approximation by quadrature or Laplace method for mixed models with nonlinear random effects or nonnormal distribution
- choice of linearization about expected values or expansion about current solutions of best linear unbiased predictors
- flexible covariance structures for random and residual random effects, including variance components, unstructured, autoregressive, and spatial structures
- CONTRAST, ESTIMATE, LSMEANS, and LSMESTIMATE statements, which produce hypothesis tests and estimable linear combinations of effects
- NLOPTIONS statement, which enables you to exercise control over the numerical optimization. You can choose techniques, update methods, line search algorithms, convergence criteria, and more. Or, you can choose the default optimization strategies selected for the particular class of model you are fitting.
- computed variables with SAS programming statements inside of PROC GLIMMIX (except for variables listed in the CLASS statement). These computed variables can appear in the MODEL, RANDOM, WEIGHT, or FREQ statement.
- grouped data analysis
- user-specified link and variance functions
- choice of model-based variance-covariance estimators for the fixed effects or empirical (sandwich) estimators to make analysis robust against misspecification of the covariance structure and to adjust for small-sample bias
- joint modeling for multivariate data. For example, you can model binary and normal responses from a subject jointly and use random effects to relate (fuse) the two outcomes.
- multinomial models for ordinal and nominal outcomes
- univariate and multivariate low-rank mixed model smoothing

Assumptions

The primary assumptions underlying the analyses performed by PROC GLIMMIX are as follows:

- If the model contains random effects, the distribution of the data conditional on the random effects is known. This distribution is either a member of the exponential family of distributions or one of the supplementary distributions provided by the GLIMMIX procedure. In models without random effects, the unconditional (marginal) distribution is assumed to be known for maximum likelihood estimation, or the first two moments are known in the case of quasi-likelihood estimation.
- The conditional expected value of the data takes the form of a linear mixed model after a monotonic transformation is applied.
- The problem of fitting the GLMM can be cast as a singly or doubly iterative optimization problem. The objective function for the optimization is a function of either the actual log likelihood, an approximation to the log likelihood, or the log likelihood of an approximated model.

For a model containing random effects, the GLIMMIX procedure, by default, estimates the parameters by applying pseudo-likelihood techniques as in Wolfinger and O'Connell (1993) and Breslow and Clayton (1993). In a model without random effects (GLM models), PROC GLIMMIX estimates the parameters by maximum likelihood, restricted maximum likelihood, or quasi-likelihood. See the section "Singly or Doubly Iterative Fitting" on page 3979 about when the GLIMMIX procedure applies noniterative, singly and doubly iterative algorithms, and the section "Default Estimation Techniques" on page 3981 about the default estimation methods. You can also fit generalized linear mixed models by maximum likelihood where the marginal distribution is numerically approximated by the Laplace method (METHOD=LAPLACE) or by adaptive Gaussian quadrature (METHOD=QUAD).

Once the parameters have been estimated, you can perform statistical inferences for the fixed effects and covariance parameters of the model. Tests of hypotheses for the fixed effects are based on Wald-type tests and the estimated variance-covariance matrix. The COVTEST statement enables you to perform inferences about covariance parameters based on likelihood ratio tests.

PROC GLIMMIX uses the Output Delivery System (ODS) for displaying and controlling the output from SAS procedures. ODS enables you to convert any of the output from PROC GLIMMIX into a SAS data set. See the section "ODS Table Names" on page 3988 for more information.

The GLIMMIX procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the GLIMMIX procedure, see the PLOTS options in the PROC GLIMMIX and LSMEANS statements.

Notation for the Generalized Linear Mixed Model

This section introduces the mathematical notation used throughout the chapter to describe the generalized linear mixed model (GLMM). See the section "Details: GLIMMIX Procedure" on page 3917 for a description of the fitting algorithms and the mathematical-statistical details.

The Basic Model

Suppose $\mathbf{Y}$ represents the $(n \times 1)$ vector of observed data and $\boldsymbol{\gamma}$ is a $(r \times 1)$ vector of random effects. Models fit by the GLIMMIX procedure assume that

$$\mathrm{E}[\mathbf{Y}|\boldsymbol{\gamma}] = g^{-1}(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma})$$

where $g(\cdot)$ is a differentiable monotonic link function and $g^{-1}(\cdot)$ is its inverse. The matrix $\mathbf{X}$ is an $(n \times p)$ matrix of rank k, and $\mathbf{Z}$ is an $(n \times r)$ design matrix for the random effects. The random effects are assumed to be normally distributed with mean $\mathbf{0}$ and variance matrix $\mathbf{G}$.

The GLMM contains a linear mixed model inside the inverse link function. This model component is referred to as the linear predictor,

$$\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}$$

The variance of the observations, conditional on the random effects, is

$$\mathrm{Var}[\mathbf{Y}|\boldsymbol{\gamma}] = \mathbf{A}^{1/2}\mathbf{R}\mathbf{A}^{1/2}$$

The matrix $\mathbf{A}$ is a diagonal matrix and contains the variance functions of the model. The variance function expresses the variance of a response as a function of the mean. The GLIMMIX procedure determines the variance function from the DIST= option in the MODEL statement or from the user-supplied variance function (see the section "Implied Variance Functions" on page 3913). The matrix $\mathbf{R}$ is a variance matrix specified by the user through the RANDOM statement. If the conditional distribution of the data contains an additional scale parameter, it is either part of the variance functions or part of the $\mathbf{R}$ matrix. For example, the gamma distribution with mean μ has the variance function $a(\mu) = \mu^2$ and $\mathrm{Var}[Y|\boldsymbol{\gamma}] = \mu^2\phi$. If your model calls for G-side random effects only (see the next section), the procedure models $\mathbf{R} = \phi\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. Table 52.22 identifies the distributions for which $\phi \equiv 1$.

G-Side and R-Side Random Effects and Covariance Structures

The GLIMMIX procedure distinguishes two types of random effects. Depending on whether the parameters of the covariance structure for random components in your model are contained in $\mathbf{G}$ or in $\mathbf{R}$, the procedure distinguishes between "G-side" and "R-side" random effects. The associated covariance structures of $\mathbf{G}$ and $\mathbf{R}$ are similarly termed the G-side and R-side covariance structure, respectively. R-side effects are also called "residual" effects. Simply put, if a random effect is an element of $\boldsymbol{\gamma}$, it is a G-side effect and you are modeling the G-side covariance structure; otherwise, you are modeling the R-side covariance structure of the model. Models without G-side effects are also known as marginal (or population-averaged) models. Models fit with the GLIMMIX procedure can have none, one, or more of each type of effect.

Note that an R-side effect in the GLIMMIX procedure is equivalent to a REPEATED effect in the MIXED procedure. The R-side covariance structure in the GLIMMIX procedure is the covariance structure that you would formulate with the REPEATED statement in the MIXED procedure. In the GLIMMIX procedure all

random effects and their covariance structures are specified through the RANDOM statement. See the section "Comparing the GLIMMIX and MIXED Procedures" on page 3977 for a comparison of the GLIMMIX and MIXED procedures.

The columns of $\mathbf{X}$ are constructed from effects listed on the right side in the MODEL statement. Columns of $\mathbf{Z}$ and the variance matrices $\mathbf{G}$ and $\mathbf{R}$ are constructed from the RANDOM statement.

The $\mathbf{R}$ matrix is by default the scaled identity matrix, $\mathbf{R} = \phi\mathbf{I}$. The scale parameter ϕ is set to one if the distribution does not have a scale parameter, such as in the case of the binary, binomial, Poisson, and exponential distribution (see Table 52.22). To specify a different $\mathbf{R}$ matrix, use the RANDOM statement with the _RESIDUAL_ keyword or the RESIDUAL option. For example, to specify that the Time effect for each patient is an R-side effect with a first-order autoregressive covariance structure, use the RESIDUAL option:

```
random time / type=ar(1) subject=patient residual;
```

To add a multiplicative overdispersion parameter, use the _RESIDUAL_ keyword:

```
random _residual_;
```

You specify the link function $g(\cdot)$ with the LINK= option in the MODEL statement or with programming statements. You specify the variance function that controls the matrix $\mathbf{A}$ with the DIST= option in the MODEL statement or with programming statements.

Unknown quantities subject to estimation are the fixed-effects parameter vector $\boldsymbol{\beta}$ and the covariance parameter vector $\boldsymbol{\theta}$ that comprises all unknowns in $\mathbf{G}$ and $\mathbf{R}$. The random effects $\boldsymbol{\gamma}$ are not parameters of the model in the sense that they are not estimated. The vector $\boldsymbol{\gamma}$ is a vector of random variables. The solutions for $\boldsymbol{\gamma}$ are predictors of these random variables.

Relationship with Generalized Linear Models

Generalized linear models (Nelder and Wedderburn 1972; McCullagh and Nelder 1989) are a special case of GLMMs. If $\boldsymbol{\gamma} = \mathbf{0}$ and $\mathbf{R} = \phi\mathbf{I}$, the GLMM reduces to either a generalized linear model (GLM) or a GLM with overdispersion. For example, if $\mathbf{Y}$ is a vector of Poisson variables so that $\mathbf{A}$ is a diagonal matrix containing $\mathrm{E}[\mathbf{Y}] = \boldsymbol{\mu}$ on the diagonal, then the model is a Poisson regression model for $\phi = 1$ and overdispersed relative to a Poisson distribution for $\phi > 1$. Because the Poisson distribution does not have an extra scale parameter, you can model overdispersion by adding the following statement to your GLIMMIX program:

```
random _residual_;
```

If the only random effect is an overdispersion effect, PROC GLIMMIX fits the model by (restricted) maximum likelihood and not by one of the methods specific to GLMMs.

PROC GLIMMIX Contrasted with Other SAS Procedures

The GLIMMIX procedure generalizes the MIXED and GENMOD procedures in two important ways. First, the response can have a nonnormal distribution. The MIXED procedure assumes that the response is normally (Gaussian) distributed. Second, the GLIMMIX procedure incorporates random effects in the model and so allows for subject-specific (conditional) and population-averaged (marginal) inference. The GENMOD procedure allows only for marginal inference.

The GLIMMIX and MIXED procedure are closely related; see the syntax and feature comparison in the section "Comparing the GLIMMIX and MIXED Procedures" on page 3977. The remainder of this section compares the GLIMMIX procedure with the GENMOD, NLMIXED, LOGISTIC, and CATMOD procedures.

The GENMOD procedure fits generalized linear models for independent data by maximum likelihood. It can also handle correlated data through the marginal GEE approach of Liang and Zeger (1986) and Zeger and Liang (1986). The GEE implementation in the GENMOD procedure is a marginal method that does not incorporate random effects. The GEE estimation in the GENMOD procedure relies on R-side covariances only, and the unknown parameters in $\mathbf{R}$ are estimated by the method of moments. The GLIMMIX procedure allows G-side random effects and R-side covariances. PROC GLIMMIX can fit marginal (GEE-type) models, but the covariance parameters are not estimated by the method of moments. The parameters are estimated by likelihood-based techniques. When the GLIMMIX and GENMOD procedures fit a generalized linear model where the distribution contains a scale parameter, such as the normal, gamma, inverse Gaussian, or negative binomial distribution, the scale parameter is reported in the "Parameter Estimates" table. For some distributions, the parameterization of this parameter differs. See the section "Scale and Dispersion Parameters" on page 3920 for details about how the GLIMMIX procedure parameterizes the log-likelihood functions and information about how the reported quantities differ between the two procedures.

Many of the fit statistics and tests in the GENMOD procedure are based on the likelihood. In a GLMM it is not always possible to derive the log likelihood of the data. Even if the log likelihood is tractable, it might be computationally infeasible. In some cases, the objective function must be constructed based on a substitute model. In other cases, only the first two moments of the marginal distribution can be approximated. Consequently, obtaining likelihood-based tests and statistics is difficult for many generalized linear mixed models. The GLIMMIX procedure relies heavily on linearization and Taylor-series techniques to construct Wald-type test statistics and confidence intervals. Likelihood ratio tests and confidence intervals for covariance parameters are available in the GLIMMIX procedure through the COVTEST statement.

The NLMIXED procedure fits nonlinear mixed models where the conditional mean function is a general nonlinear function. The class of generalized linear mixed models is a special case of the nonlinear mixed models; hence some of the models you can fit with PROC NLMIXED can also be fit with the GLIMMIX procedure. The NLMIXED procedure relies by default on approximating the marginal log likelihood through adaptive Gaussian quadrature. In the GLIMMIX procedure, maximum likelihood estimation by adaptive Gaussian quadrature is available with the METHOD=QUAD option in the PROC GLIMMIX statement. The default estimation methods thus differ between the NLMIXED and GLIMMIX procedures, because adaptive quadrature is possible for only a subset of the models available with the GLIMMIX procedure. If you choose METHOD=LAPLACE or METHOD=QUAD(QPOINTS=1) in the PROC GLIMMIX statement for a generalized linear mixed model, the GLIMMIX procedure performs maximum likelihood estimation based on a Laplace approximation of the marginal log likelihood. This is equivalent to the QPOINTS=1 option in the NLMIXED procedure.

The LOGISTIC and CATMOD procedures also fit generalized linear models; PROC LOGISTIC accommodates the independence case only. Binary, binomial, multinomial models for ordered data, and generalized

logit models that can be fit with PROC LOGISTIC can also be fit with the GLIMMIX procedure. The diagnostic tools and capabilities specific to such data implemented in the LOGISTIC procedure go beyond the capabilities of the GLIMMIX procedure.

Getting Started: GLIMMIX Procedure

Logistic Regressions with Random Intercepts

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/gmxgs1.sas.)

Researchers investigated the performance of two medical procedures in a multicenter study. They randomly selected 15 centers for inclusion. One of the study goals was to compare the occurrence of side effects for the procedures. In each center n_A patients were randomly selected and assigned to procedure "A," and n_B patients were randomly assigned to procedure "B." The following DATA step creates the data set for the analysis:

```
data multicenter;
   input center group$ n sideeffect;
   datalines;
 1  A  32  14
 1  B  33  18
 2  A  30   4
 2  B  28   8
 3  A  23  14
 3  B  24   9
 4  A  22   7
 4  B  22  10
 5  A  20   6
 5  B  21  12
 6  A  19   1
 6  B  20   3
 7  A  17   2
 7  B  17   6
 8  A  16   7
 8  B  15   9
 9  A  13   1
 9  B  14   5
10  A  13   3
10  B  13   1
11  A  11   1
11  B  12   2
12  A  10   1
12  B   9   0
13  A   9   2
13  B   9   6
14  A   8   1
14  B   8   1
15  A   7   1
```

```
15  B   8   0
;
```

The variable group identifies the two procedures, n is the number of patients who received a given procedure in a particular center, and sideeffect is the number of patients who reported side effects.

If Y_{iA} and Y_{iB} denote the number of patients in center i who report side effects for procedures A and B, respectively, then—for a given center—these are independent binomial random variables. To model the probability of side effects for the two drugs, π_{iA} and π_{iB}, you need to account for the fixed group effect and the random selection of centers. One possibility is to assume a model that relates group and center effects linearly to the logit of the probabilities:

$$\log\left\{\frac{\pi_{iA}}{1-\pi_{iA}}\right\} = \beta_0 + \beta_A + \gamma_i$$
$$\log\left\{\frac{\pi_{iB}}{1-\pi_{iB}}\right\} = \beta_0 + \beta_B + \gamma_i$$

In this model, $\beta_A - \beta_B$ measures the difference in the logits of experiencing side effects, and the γ_i are independent random variables due to the random selection of centers. If you think of β_0 as the overall intercept in the model, then the γ_i are random intercept adjustments. Observations from the same center receive the same adjustment, and these vary randomly from center to center with variance $\mathrm{Var}[\gamma_i] = \sigma_c^2$.

Because π_{iA} is the conditional mean of the sample proportion, $\mathrm{E}[Y_{iA}/n_{iA}|\gamma_i] = \pi_{iA}$, you can model the sample proportions as binomial ratios in a generalized linear mixed model. The following statements request this analysis under the assumption of normally distributed center effects with equal variance and a logit link function:

```
proc glimmix data=multicenter;
   class center group;
   model sideeffect/n = group / solution;
   random intercept / subject=center;
run;
```

The PROC GLIMMIX statement invokes the procedure. The CLASS statement instructs the procedure to treat the variables center and group as classification variables. The MODEL statement specifies the response variable as a sample proportion by using the *events/trials* syntax. In terms of the previous formulas, sideeffect/n corresponds to Y_{iA}/n_{iA} for observations from group A and to Y_{iB}/n_{iB} for observations from group B. The SOLUTION option in the MODEL statement requests a listing of the solutions for the fixed-effects parameter estimates. Note that because of the *events/trials* syntax, the GLIMMIX procedure defaults to the binomial distribution, and that distribution's default link is the logit link. The RANDOM statement specifies that the linear predictor contains an intercept term that randomly varies at the level of the center effect. In other words, a random intercept is drawn separately and independently for each center in the study.

The results of this analysis are shown in Figure 52.1–Figure 52.9.

The "Model Information Table" in Figure 52.1 summarizes important information about the model you fit and about aspects of the estimation technique.

Figure 52.1 Model Information

The GLIMMIX Procedure

Model Information	
Data Set	WORK.MULTICENTER
Response Variable (Events)	sideeffect
Response Variable (Trials)	n
Response Distribution	Binomial
Link Function	Logit
Variance Function	Default
Variance Matrix Blocked By	center
Estimation Technique	Residual PL
Degrees of Freedom Method	Containment

PROC GLIMMIX recognizes the variables sideeffect and n as the numerator and denominator in the *events/trials* syntax, respectively. The distribution—conditional on the random center effects—is binomial. The marginal variance matrix is block-diagonal, and observations from the same center form the blocks. The default estimation technique in generalized linear mixed models is residual pseudo-likelihood with a subject-specific expansion (METHOD=RSPL).

The "Class Level Information" table lists the levels of the variables specified in the CLASS statement and the ordering of the levels. The "Number of Observations" table displays the number of observations read and used in the analysis (Figure 52.2).

Figure 52.2 Class Level Information and Number of Observations

Class Level Information		
Class	**Levels**	**Values**
center	15	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
group	2	A B

Number of Observations Read	30
Number of Observations Used	30
Number of Events	155
Number of Trials	503

There are two variables listed in the CLASS statement. The center variable has fifteen levels, and the group variable has two levels. Because the response is specified through the *events/trial* syntax, the "Number of Observations" table also contains the total number of events and trials used in the analysis.

The "Dimensions" table lists the size of relevant matrices (Figure 52.3).

Figure 52.3 Dimensions

Dimensions	
G-side Cov. Parameters	1
Columns in X	3
Columns in Z per Subject	1
Subjects (Blocks in V)	15
Max Obs per Subject	2

There are three columns in the **X** matrix, corresponding to an intercept and the two levels of the group variable. For each subject (center), the **Z** matrix contains only an intercept column.

The "Optimization Information" table provides information about the methods and size of the optimization problem (Figure 52.4).

Figure 52.4 Optimization Information

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	1
Lower Boundaries	1
Upper Boundaries	0
Fixed Effects	Profiled
Starting From	Data

The default optimization technique for generalized linear mixed models with binomial data is the quasi-Newton method. Because a residual likelihood technique is used to compute the objective function, only the covariance parameters participate in the optimization. A lower boundary constraint is placed on the variance component for the random center effect. The solution for this variance cannot be less than zero.

The "Iteration History" table displays information about the progress of the optimization process. After the initial optimization, the GLIMMIX procedure performed 15 updates before the convergence criterion was met (Figure 52.5). At convergence, the largest absolute value of the gradient was near zero. This indicates that the process stopped at an extremum of the objective function.

Figure 52.5 Iteration History and Convergence Status

Iteration History					
Iteration	**Restarts**	**Subiterations**	**Objective Function**	**Change**	**Max Gradient**
0	**0**	**5**	79.688580269	0.11807224	7.851E-7
1	**0**	**3**	81.294622554	0.02558021	8.209E-7
2	**0**	**2**	81.438701534	0.00166079	4.061E-8
3	**0**	**1**	81.444083567	0.00006263	2.273E-8
4	**0**	**1**	81.444265216	0.00000421	0.000025
5	**0**	**1**	81.444277364	0.00000383	0.000023
6	**0**	**1**	81.444266322	0.00000348	0.000021
7	**0**	**1**	81.44427636	0.00000316	0.000019
8	**0**	**1**	81.444267235	0.00000287	0.000017
9	**0**	**1**	81.44427553	0.00000261	0.000016
10	**0**	**1**	81.44426799	0.00000237	0.000014
11	**0**	**1**	81.444274844	0.00000216	0.000013
12	**0**	**1**	81.444268614	0.00000196	0.000012
13	**0**	**1**	81.444274277	0.00000178	0.000011
14	**0**	**1**	81.444269129	0.00000162	9.772E-6
15	**0**	**0**	81.444273808	0.00000000	9.102E-6

Convergence criterion (PCONV=1.11022E-8) satisfied.

The "Fit Statistics" table lists information about the fitted model (Figure 52.6).

Figure 52.6 Fit Statistics

Fit Statistics	
-2 Res Log Pseudo-Likelihood	81.44
Generalized Chi-Square	30.69
Gener. Chi-Square / DF	1.10

Twice the negative of the residual log likelihood in the final pseudo-model equaled 81.44. The ratio of the generalized chi-square statistic and its degrees of freedom is close to 1. This is a measure of the residual variability in the marginal distribution of the data.

The "Covariance Parameter Estimates" table displays estimates and asymptotic estimated standard errors for all covariance parameters (Figure 52.7).

Figure 52.7 Covariance Parameter Estimates

Covariance Parameter Estimates			
Cov Parm	Subject	Estimate	Standard Error
Intercept	center	0.6176	0.3181

The variance of the random center intercepts on the logit scale is estimated as $\widehat{\sigma}_c^2 = 0.6176$.

The "Parameter Estimates" table displays the solutions for the fixed effects in the model (Figure 52.8).

Figure 52.8 Parameter Estimates

Solutions for Fixed Effects						
Effect	group	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		-0.8071	0.2514	14	-3.21	0.0063
group	A	-0.4896	0.2034	14	-2.41	0.0305
group	B	0	.	.	.	.

Because of the fixed-effects parameterization used in the GLIMMIX procedure, the "Intercept" effect is an estimate of $\beta_0 + \beta_B$, and the "A" group effect is an estimate of $\beta_A - \beta_B$, the log odds ratio. The associated estimated probabilities of side effects in the two groups are

$$\widehat{\pi}_A = \frac{1}{1 + \exp\{0.8071 + 0.4896\}} = 0.2147$$

$$\widehat{\pi}_B = \frac{1}{1 + \exp\{0.8071\}} = 0.3085$$

There is a significant difference between the two groups ($p = 0.0305$).

The "Type III Tests of Fixed Effect" table displays significance tests for the fixed effects in the model (Figure 52.9).

Figure 52.9 Type III Tests of Fixed Effects

Type III Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
group	1	14	5.79	0.0305

Because the group effect has only two levels, the p-value for the effect is the same as in the "Parameter Estimates" table, and the "F Value" is the square of the "t Value" shown there.

You can produce the estimates of the average logits in the two groups and their predictions on the scale of the data with the LSMEANS statement in PROC GLIMMIX:

```
ods select lsmeans;
proc glimmix data=multicenter;
   class center group;
   model sideeffect/n = group / solution;
   random intercept / subject=center;
   lsmeans group / cl ilink;
run;
```

The LSMEANS statement requests the least squares means of the group effect on the logit scale. The CL option requests their confidence limits. The ILINK option adds estimates, standard errors, and confidence limits on the mean (probability) scale (Figure 52.10).

Figure 52.10 Least Squares Means

The GLIMMIX Procedure

group Least Squares Means

group	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper	Mean	Standard Error Mean	Lower Mean	Upper Mean
A	-1.2966	0.2601	14	-4.99	0.0002	0.05	-1.8544	-0.7388	0.2147	0.04385	0.1354	0.3233
B	-0.8071	0.2514	14	-3.21	0.0063	0.05	-1.3462	-0.2679	0.3085	0.05363	0.2065	0.4334

The "Estimate" column displays the least squares mean estimate on the logit scale, and the "Mean" column represents its mapping onto the probability scale. The "Lower" and "Upper" columns are 95% confidence limits for the logits in the two groups. The "Lower Mean" and "Upper Mean" columns are the corresponding confidence limits for the probabilities of side effects. These limits are obtained by inversely linking the confidence bounds on the linear scale, and thus are not symmetric about the estimate of the probabilities.

Syntax: GLIMMIX Procedure

The following statements are available in the GLIMMIX procedure:

```
PROC GLIMMIX <options> ;
    BY variables ;
    CLASS variable <(REF= option)> ... <variable <(REF= option)>> </global-options> ;
    CODE <options> ;
    CONTRAST 'label' contrast-specification <, contrast-specification> <, ... > </options> ;
    COVTEST <'label'> <test-specification> </options> ;
    EFFECT effect-specification ;
    ESTIMATE 'label' contrast-specification <(divisor=n)>
                <, 'label' contrast-specification <(divisor=n)>> <, ... > </options> ;
    FREQ variable ;
    ID variables ;
    LSMEANS fixed-effects </options> ;
    LSMESTIMATE fixed-effect <'label'> values <divisor=n>
                    <, <'label'> values <divisor=n>> <, ... > </options> ;
    MARGINS fixed-effects </options> ;
    MODEL response<(response-options)> = <fixed-effects> </model-options> ;
    MODEL events/trials = <fixed-effects> </model-options> ;
    NLOPTIONS <options> ;
    OUTPUT <OUT=SAS-data-set>
            <keyword<(keyword-options)> <=name>> ...
            <keyword<(keyword-options)> <=name>> </options> ;
    PARMS (value-list)... </options> ;
    RANDOM random-effects </options> ;
    SLICE model-effect </options> ;
    STORE <OUT=>item-store-name </ LABEL='label'> ;
    WEIGHT variable ;
    Programming statements ;
```

The CLASS, CONTRAST, COVTEST, EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, RANDOM and SLICE statements and the programming statements can appear multiple times. The PROC GLIMMIX and MODEL statements are required, and the MODEL statement must appear after the CLASS statement if a CLASS statement is included. The EFFECT statements must appear before the MODEL statement.

The SLICE statement is also available in many other procedures. A summary description of functionality and syntax for this statement is given in this chapter. You can find full documentation in the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

PROC GLIMMIX Statement

PROC GLIMMIX < *options* > **;**

The PROC GLIMMIX statement invokes the GLIMMIX procedure. Table 52.1 summarizes the *options* available in the PROC GLIMMIX statement. These and other *options* in the PROC GLIMMIX statement are then described fully in alphabetical order.

Table 52.1 PROC GLIMMIX Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
METHOD=	Determines estimation method
NOFIT	Does not fit the model
NOPROFILE	Includes scale parameter in optimization
NOREML	Determines computation of scale parameters in GLM models
ORDER=	Determines the sort order of CLASS variables
OUTDESIGN	Writes **X** and/or **Z** matrices to a SAS data set
PROFILE	Profile scale parameters from the optimization
Displayed Output	
ASYCORR	Displays the asymptotic correlation matrix of the covariance parameter estimates
ASYCOV	Displays the asymptotic covariance matrix of the covariance parameter estimates
GRADIENT	Displays the gradient of the objective function with respect to the parameter estimates
HESSIAN	Displays the Hessian matrix
ITDETAILS	Adds estimates and gradients to the “Iteration History”
NAMELEN=	Specifies the length of long effect names
NOBSDETAIL	Shows data exclusions
NOCLPRINT	Suppresses “Class Level Information” completely or in part
ODDSRATIO	Requests odds ratios
PLOTS	Produces ODS statistical graphics
SUBGRADIENT	Writes subject-specific gradients to a SAS data set
Optimization Options	
MAXOPT=	Specifies the number of optimizations
Computational Options	
CHOLESKY	Constructs and solves mixed model equations using the Cholesky root of the **G** matrix
EMPIRICAL	Computes empirical (“sandwich”) estimators
EXPHESSIAN	Uses the expected Hessian matrix to compute the covariance matrix of nonprofiled parameters
INFOCRIT	Affects the computation of information criteria
INITGLM	Uses fixed-effects starting values via generalized linear model

Table 52.1 *continued*

Option	Description
INITITER=	Sets the number of initial GLM steps
NOBOUND	Unbounds the covariance parameter estimates
NOINITGLM	Does not use fixed-effects starting values via generalized linear model
SCORING=	Applies Fisher scoring where applicable
SCOREMOD	Bases the Hessian matrix in GLMMs on a modified scoring algorithm
Singularity Tolerances	
ABSPCONV=	Determines the absolute parameter estimate convergence criterion for PL
FDIGITS=	Specifies significant digits in computing objective function
PCONV=	Specifies the relative parameter estimate convergence criterion for PL
SINGCHOL=	Tunes singularity for Cholesky decompositions
SINGRES=	Tunes singularity for the residual variance
SINGULAR=	Tunes general singularity criterion
Debugging Output	
LIST	Lists model program and variables

You can specify the following *options* in the PROC GLIMMIX statement.

ABSPCONV=*r*

specifies an absolute parameter estimate convergence criterion for doubly iterative estimation methods. For such methods, the GLIMMIX procedure by default examines the *relative* change in parameter estimates between optimizations (see PCONV=). The purpose of the ABSPCONV= criterion is to stop the process when the *absolute* change in parameter estimates is less than the tolerance criterion r. The criterion is based on fixed effects and covariance parameters.

Note that this convergence criterion does not affect the convergence criteria applied within any individual optimization. In order to change the convergence behavior within an optimization, you can change the ABSCONV=, ABSFCONV=, ABSGCONV=, ABSXCONV=, FCONV=, or GCONV= option in the NLOPTIONS statement.

ASYCORR

produces the asymptotic correlation matrix of the covariance parameter estimates. It is computed from the corresponding asymptotic covariance matrix (see the description of the ASYCOV option, which follows).

ASYCOV

requests that the asymptotic covariance matrix of the covariance parameter estimates be displayed. By default, this matrix is the observed inverse Fisher information matrix, which equals $m\mathbf{H}^{-1}$, where $\mathbf{H}$ is the Hessian (second derivative) matrix of the objective function. The factor m equals 1 in a GLM and equals 2 in a GLMM.

When you use the SCORING= option and PROC GLIMMIX converges without stopping the scoring algorithm, the procedure uses the expected Hessian matrix to compute the covariance matrix instead of the observed Hessian. Regardless of whether a scoring algorithm is used or the number of scoring iterations has been exceeded, you can request that the asymptotic covariance matrix be based on the expected Hessian with the EXPHESSIAN option in the PROC GLIMMIX statement. If a residual scale parameter is profiled from the likelihood equation, the asymptotic covariance matrix is adjusted for the presence of this parameter; details of this adjustment process are found in Wolfinger, Tobias, and Sall (1994) and in the section "Estimated Precision of Estimates" on page 3926.

CHOLESKY
CHOL

requests that the mixed model equations be constructed and solved by using the Cholesky root of the **G** matrix. This option applies only to estimation methods that involve mixed model equations. The Cholesky root algorithm has greater numerical stability but also requires more computing resources. When the estimated **G** matrix is not positive definite during a particular function evaluation, PROC GLIMMIX switches to the Cholesky algorithm for that evaluation and returns to the regular algorithm if $\widehat{\mathbf{G}}$ becomes positive definite again. When the CHOLESKY option is in effect, the procedure applies the algorithm all the time.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC GLIMMIX. The default is the most recently created data set.

EMPIRICAL< =CLASSICAL | HC0 >
EMPIRICAL< =DF | HC1 >
EMPIRICAL< =MBN< (*mbn-options***) > >**
EMPIRICAL< =ROOT | HC2 >
EMPIRICAL< =FIRORES | HC3 >
EMPIRICAL< =FIROEEQ< (*r***) > >**

requests that the covariance matrix of the parameter estimates be computed as one of the asymptotically consistent estimators, known as *sandwich* or *empirical* estimators. The name stems from the layering of the estimator. An empirically based estimate of the inverse variance of the parameter estimates (the "meat") is wrapped by the model-based variance estimate (the "bread").

Empirical estimators are useful for obtaining inferences that are not sensitive to the choice of the covariance model. In nonmixed models, they can help, for example, to allay the effects of variance heterogeneity on the tests of fixed effects. Empirical estimators can coarsely be grouped into likelihood-based and residual-based estimators. The distinction arises from the components used to construct the "meat" and "bread" of the estimator. If you specify the EMPIRICAL option without further qualifiers, the GLIMMIX procedure computes the classical sandwich estimator in the appropriate category.

Likelihood-Based Estimator

Let $\mathbf{H}(\boldsymbol{\alpha})$ denote the second derivative matrix of the log likelihood for some parameter vector $\boldsymbol{\alpha}$, and let $\mathbf{g}_i(\boldsymbol{\alpha})$ denote the gradient of the log likelihood with respect to $\boldsymbol{\alpha}$ for the ith of m independent sampling units. The gradient for the entire data is $\sum_{i=1}^{m} \mathbf{g}_i(\boldsymbol{\alpha})$. A sandwich estimator for the covariance matrix

of $\widehat{\boldsymbol{\alpha}}$ can then be constructed as (White 1982)

$$\mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1}\left(\sum_{i=1}^{m}\mathbf{g}_i(\widehat{\boldsymbol{\alpha}})\mathbf{g}_i(\widehat{\boldsymbol{\alpha}})'\right)\mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1}$$

If you fit a mixed model by maximum likelihood with Laplace or quadrature approximation (METHOD=LAPLACE, METHOD=QUAD), the GLIMMIX procedure constructs this likelihood-based estimator when you choose EMPIRICAL=CLASSICAL. If you choose EMPIRICAL=MBN, the likelihood-based sandwich estimator is further adjusted (see the section "Design-Adjusted MBN Estimator" on page 3953 for details). Because Laplace and quadrature estimation in GLIMMIX includes the fixed-effects parameters and the covariance parameters in the optimization, this empirical estimator adjusts the covariance matrix of both types of parameters. The following empirical estimators are not available with METHOD=LAPLACE or with METHOD=QUAD: EMPIRICAL=DF, EMPIRICAL=ROOT, EMPIRICAL=FIRORES, and EMPIRICAL=FIROEEQ.

Residual-Based Estimators

For a general model, let $\mathbf{Y}$ denote the response with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$, and let $\mathbf{D}$ be the matrix of first derivatives of $\boldsymbol{\mu}$ with respect to the fixed effects $\boldsymbol{\beta}$. The classical sandwich estimator (Huber 1967; White 1980) is

$$\widehat{\boldsymbol{\Omega}}\left(\sum_{i=1}^{m}\widehat{\mathbf{D}}_i'\widehat{\boldsymbol{\Sigma}}_i^{-1}\mathbf{e}_i\mathbf{e}_i'\widehat{\boldsymbol{\Sigma}}_i^{-1}\widehat{\mathbf{D}}_i\right)\widehat{\boldsymbol{\Omega}}$$

where $\boldsymbol{\Omega} = (\mathbf{D}'\boldsymbol{\Sigma}^{-1}\mathbf{D})^{-}$, $\mathbf{e}_i = \mathbf{y}_i - \widehat{\boldsymbol{\mu}}_i$, and m denotes the number of independent sampling units.

Since the expected value of $\mathbf{e}_i\mathbf{e}_i'$ does not equal $\boldsymbol{\Sigma}_i$, the classical sandwich estimator is biased, particularly if m is small. The estimator tends to underestimate the variance of $\widehat{\boldsymbol{\beta}}$. The EMPIRICAL=DF, ROOT, FIRORES, FIROEEQ, and MBN estimators are bias-corrected sandwich estimators. The DF estimator applies a simple sample size adjustment. The ROOT, FIRORES, and FIROEEQ estimators are based on Taylor series approximations applied to residuals and estimating equations. For uncorrelated data, the EMPIRICAL=FIRORES estimator can be motivated as a jackknife estimator.

In the case of a linear regression model, the various estimators reduce to the *heteroscedasticity-consistent covariance matrix* estimators (HCMM) of White (1980) and MacKinnon and White (1985). The classical estimator, HC0, was found to perform poorly in small samples. Based on simulations in regression models, MacKinnon and White (1985) and Long and Ervin (2000) strongly recommend the HC3 estimator. The sandwich estimators computed by the GLIMMIX procedure can be viewed as an extension of the HC0—HC3 estimators of MacKinnon and White (1985) to accommodate nonnormal data and correlated observations.

The MBN estimator, introduced as a residual-based estimator (Morel 1989; Morel, Bokossa, and Neerchal 2003), applies an additive adjustment to the residual crossproduct. It is controlled by three suboptions. The valid *mbn-options* are as follows: a sample size adjustment is applied when the DF suboption is in effect. The NODF suboption suppresses this component of the adjustment. The lower bound of the design effect parameter $0 \le r \le 1$ can be specified with the R= option. The magnitude of Morel's δ parameter is partly determined with the D= option ($d \ge 1$).

For details about the general expression for the residual-based estimators and their relationship, see the section "Empirical Covariance ("Sandwich") Estimators" on page 3952. The MBN estimator and its parameters are explained for residual- and likelihood-based estimators in the section "Design-Adjusted MBN Estimator" on page 3953.

The EMPIRICAL=DF estimator applies a simple, multiplicative correction factor to the classical estimator (Hinkley 1977). This correction factor is

$$c = \begin{cases} m/(m-k) & m > k \\ 1 & \text{otherwise} \end{cases}$$

where k is the rank of $\mathbf{X}$, and m equals the sum of all frequencies when PROC GLIMMIX is in GLM mode and equals the number of subjects in GLMM mode. For example, the following statements fit an overdispersed GLM:

```
proc glimmix empirical;
   model y = x;
   random _residual_;
run;
```

PROC GLIMMIX is in GLM mode, and the individual observations are the independent sampling units from which the sandwich estimator is constructed. If you use a SUBJECT= effect in the RANDOM statement, however, the procedure fits the model in GLMM mode and the subjects represent the sampling units in the construction of the sandwich estimator. In other words, the following statements fit a GEE-type model with independence working covariance structure and subjects (clusters) defined by the levels of ID:

```
proc glimmix empirical;
   class id;
   model y = x;
   random _residual_ / subject=id type=vc;
run;
```

See the section "GLM Mode or GLMM Mode" on page 3941 for information about how the GLIMMIX procedure determines the estimation mode.

The EMPIRICAL=ROOT estimator is based on the residual approximation in Kauermann and Carroll (2001), and the EMPIRICAL=FIRORES estimator is based on the approximation in Mancl and DeRouen (2001). The Kauermann and Carroll estimator requires the inverse square root of a nonsymmetric matrix. This square root matrix is obtained from the singular value decomposition in PROC GLIMMIX, and thus this sandwich estimator is computationally more demanding than others. In the linear regression case, the Mancl-DeRouen estimator can be motivated as a jackknife estimator, based on the "leave-one-out" estimates of $\widehat{\boldsymbol{\beta}}$; see MacKinnon and White (1985) for details.

The EMPIRICAL=FIROEEQ estimator is based on approximating an unbiased estimating equation (Fay and Graubard 2001). It is computationally less demanding than the estimator of Kauermann and Carroll (2001) and, in certain balanced cases, gives identical results. The optional number $0 \le r < 1$ is chosen to provide an upper bound on the correction factor. The default value for r is 0.75.

When you specify the EMPIRICAL option with a residual-based estimator, PROC GLIMMIX adjusts all standard errors and test statistics involving the fixed-effects parameters.

Sampling Units

Computation of an empirical variance estimator requires that the data can be processed by independent sampling units. This is always the case in GLMs. In this case, m, the number of independent units, equals the sum of the frequencies used in the analysis (see "Number of Observations" table). In GLMMs, empirical estimators can be computed only if the data comprise more than one subject as per the "Dimensions" table. See the section "Processing by Subjects" on page 3957 for information about how the GLIMMIX procedure determines whether the data can be processed by subjects. If a GLMM comprises only a single subject for a particular BY group, the model-based variance estimator is used instead of the empirical estimator, and a message is written to the log.

EXPHESSIAN

requests that the expected Hessian matrix be used in computing the covariance matrix of the nonprofiled parameters. By default, the GLIMMIX procedure uses the observed Hessian matrix in computing the asymptotic covariance matrix of covariance parameters in mixed models and the covariance matrix of fixed effects in models without random effects. The EXPHESSIAN option is ignored if the (conditional) distribution is not a member of the exponential family or is unknown. It is also ignored in models for nominal data.

FDIGITS=*r*

specifies the number of accurate digits in evaluations of the objective function. Fractional values are allowed. The default value is $r = -\log_{10} \epsilon$, where ϵ is the machine precision. The value of r is used to compute the interval size for the computation of finite-difference approximations of the derivatives of the objective function. It is also used in computing the default value of the FCONV= option in the NLOPTIONS statement.

GRADIENT

displays the gradient of the objective function with respect to the parameter estimates in the "Covariance Parameter Estimates" table and/or the "Parameter Estimates" table.

HESSIAN

HESS

H

displays the Hessian matrix of the optimization.

INFOCRIT=NONE | PQ | Q

IC=NONE | PQ | Q

determines the computation of information criteria in the "Fit Statistics" table. The GLIMMIX procedure computes various information criteria that typically apply a penalty to the (possibly restricted) log likelihood, log pseudo-likelihood, or log quasi-likelihood that depends on the number of parameters and/or the sample size. If IC=NONE, these criteria are suppressed in the "Fit Statistics" table. This is the default for models based on pseudo-likelihoods.

The AIC, AICC, BIC, CAIC, and HQIC fit statistics are various information criteria. AIC and AICC represent Akaike's information criteria (Akaike 1974) and a small sample bias corrected version thereof (for AICC, see Hurvich and Tsai 1989; Burnham and Anderson 1998). BIC represents Schwarz's Bayesian criterion (Schwarz 1978). Table 52.2 gives formulas for the criteria.

Table 52.2 Information Criteria

Criterion	Formula	Reference
AIC	$-2\ell + 2d$	Akaike (1974)
AICC	$-2\ell + 2dn^*/(n^* - d - 1)$	Hurvich and Tsai (1989) Burnham and Anderson (1998)
HQIC	$-2\ell + 2d \log\log n$	Hannan and Quinn (1979)
BIC	$-2\ell + d \log n$	Schwarz (1978)
CAIC	$-2\ell + d(\log n + 1)$	Bozdogan (1987)

Here, ℓ denotes the maximum value of the (possibly restricted) log likelihood, log pseudo-likelihood, or log quasi-likelihood, d is the dimension of the model, and n, n^* reflect the size of the data.

The IC=PQ option requests that the penalties include the number of fixed-effects parameters, when estimation in models with random effects is based on a residual (restricted) likelihood. For METHOD=MSPL, METHOD=MMPL, METHOD=LAPLACE, and METHOD=QUAD, IC=Q and IC=PQ produce the same results. IC=Q is the default for linear mixed models with normal errors, and the resulting information criteria are identical to the IC option in the MIXED procedure.

The quantities d, n, and n^* depend on the model and IC= option in the following way:

GLM: IC=Q and IC=PQ options have no effect on the computation.

- d equals the number of parameters in the optimization whose solutions do not fall on the boundary or are otherwise constrained. The scale parameter is included, if it is part of the optimization. If you use the PARMS statement to place a hold on a scale parameter, that parameter does not count toward d.
- n equals the sum of the frequencies (f) for maximum likelihood and quasi-likelihood estimation and $f - \mathrm{rank}(\mathbf{X})$ for restricted maximum likelihood estimation.
- n^* equals n, unless $n < d + 2$, in which case $n^* = d + 2$.

GLMM, IC=Q:

- d equals the number of effective covariance parameters—that is, covariance parameters whose solution does not fall on the boundary. For estimation of an unrestricted objective function (METHOD=MMPL, METHOD=MSPL, METHOD=LAPLACE, METHOD=QUAD), this value is incremented by $\mathrm{rank}(\mathbf{X})$.
- n equals the effective number of subjects as displayed in the "Dimensions" table, unless this value equals 1, in which case n equals the number of levels of the first G-side RANDOM effect specified. If the number of effective subjects equals 1 and there are no G-side random effects, n is determined as

$$n = \begin{cases} f - \mathrm{rank}(\mathbf{X}) & \mathrm{METHOD} = \mathrm{RMPL}, \mathrm{METHOD} = \mathrm{RSPL} \\ f & \text{otherwise} \end{cases}$$

  where f is the sum of frequencies used.
- n^* equals f or $f - \mathrm{rank}(\mathbf{X})$ (for METHOD=RMPL and METHOD=RSPL), unless this value is less than $d + 2$, in which case $n^* = d + 2$.

GLMM, IC=PQ: For METHOD=MSPL, METHOD=MMPL, METHOD=LAPLACE, and METHOD=QUAD, the results are the same as for IC=Q. For METHOD=RSPL and METHOD=RMPL, d equals the number of effective covariance parameters plus $\mathrm{rank}(\mathbf{X})$, and $n = n^*$ equals $f - \mathrm{rank}(\mathbf{X})$. The formulas for the information criteria thus agree with Verbeke and Molenberghs (2000, Table 6.7, p. 74) and Vonesh and Chinchilli (1997, p. 263).

INITGLM

requests that the estimates from a generalized linear model fit (a model without random effects) be used as the starting values for the generalized linear mixed model. This option is the default for METHOD=LAPLACE and METHOD=QUAD.

INITITER=*number*

specifies the maximum number of iterations used when a generalized linear model is fit initially to derive starting values for the fixed effects; see the INITGLM option. By default, the initial fit involves at most four iteratively reweighted least squares updates. You can change the upper limit of initial iterations with *number*. If the model does not contain random effects, this option has no effect.

ITDETAILS

adds parameter estimates and gradients to the "Iteration History" table.

LIST

requests that the model program and variable lists be displayed. This is a debugging feature and is not normally needed. When you use programming statements to define your statistical model, this option enables you to examine the complete set of statements submitted for processing. See the section "Programming Statements" for more details about how to use SAS statements with the GLIMMIX procedure.

MAXLMMUPDATE=*number*

MAXOPT=*number*

specifies the maximum number of optimizations for doubly iterative estimation methods based on linearizations. After each optimization, a new pseudo-model is constructed through a Taylor series expansion. This step is known as the linear mixed model update. The MAXLMMUPDATE option limits the number of updates and thereby limits the number of optimizations. If this option is not specified, *number* is set equal to the value specified in the MAXITER= option in the NLOPTIONS statement. If no MAXITER= value is given, *number* defaults to 20.

METHOD=RSPL | MSPL | RMPL | MMPL | LAPLACE | QUAD< (*quad-options***) >**

specifies the estimation method in a generalized linear mixed model (GLMM). The default is METHOD=RSPL.

Pseudo-Likelihood

Estimation methods ending in "PL" are pseudo-likelihood techniques. The first letter of the METHOD= identifier determines whether estimation is based on a residual likelihood ("R") or a maximum likelihood ("M"). The second letter identifies the expansion locus for the underlying approximation. Pseudo-likelihood methods for generalized linear mixed models can be cast in terms of Taylor series expansions (linearizations) of the GLMM. The expansion locus of the expansion is either the vector

of random effects solutions ("S") or the mean of the random effects ("M"). The expansions are also referred to as the "S"ubject-specific and "M"arginal expansions. The abbreviation "PL" identifies the method as a pseudo-likelihood technique.

Residual methods account for the fixed effects in the construction of the objective function, which reduces the bias in covariance parameter estimates. Estimation methods involving Taylor series create pseudo-data for each optimization. Those data are transformed to have zero mean in a residual method. While the covariance parameter estimates in a residual method are the maximum likelihood estimates for the transformed problem, the fixed-effects estimates are (estimated) generalized least squares estimates. In a likelihood method that is not residual based, both the covariance parameters and the fixed-effects estimates are maximum likelihood estimates, but the former are known to have greater bias. In some problems, residual likelihood estimates of covariance parameters are unbiased.

For more information about linearization methods for generalized linear mixed models, see the section "Pseudo-likelihood Estimation Based on Linearization" on page 3924.

Maximum Likelihood with Laplace Approximation

If you choose METHOD=LAPLACE with a generalized linear mixed model, PROC GLIMMIX approximates the marginal likelihood by using Laplace's method. Twice the negative of the resulting log-likelihood approximation is the objective function that the procedure minimizes to determine parameter estimates. Laplace estimates typically exhibit better asymptotic behavior and less small-sample bias than pseudo-likelihood estimators. On the other hand, the class of models for which a Laplace approximation of the marginal log likelihood is available is much smaller compared to the class of models to which PL estimation can be applied.

To determine whether Laplace estimation can be applied in your model, consider the marginal distribution of the data in a mixed model

$$
\begin{aligned}
p(\mathbf{y}) &= \int p(\mathbf{y}|\boldsymbol{\gamma})\, p(\boldsymbol{\gamma})\; d\boldsymbol{\gamma} \\
&= \int \exp\left\{\log\{p(\mathbf{y}|\boldsymbol{\gamma})\} + \log\{p(\boldsymbol{\gamma})\}\right\}\; d\boldsymbol{\gamma} \\
&= \int \exp\left\{n f(\mathbf{y}, \boldsymbol{\gamma})\right\}\; d\boldsymbol{\gamma}
\end{aligned}
$$

The function $f(\mathbf{y}, \boldsymbol{\gamma})$ plays an important role in the Laplace approximation: it is a function of the joint distribution of the data and the random effects (see the section "Maximum Likelihood Estimation Based on Laplace Approximation" on page 3929). In order to construct a Laplace approximation, PROC GLIMMIX requires a conditional log-likelihood $\log\{p(\mathbf{y}|\boldsymbol{\gamma})\}$ as well as the distribution of the G-side random effects. The random effects are always assumed to be normal with zero mean and covariance structure determined by the RANDOM statement. The conditional distribution is determined by the DIST= option of the MODEL statement or the default associated with a particular response type. Because a valid conditional distribution is required, R-side random effects are not permitted for METHOD=LAPLACE in the GLIMMIX procedure. In other words, the GLIMMIX procedure requires for METHOD=LAPLACE conditional independence without R-side overdispersion or covariance structure.

Because the marginal likelihood of the data is approximated numerically, certain features of the marginal distribution are not available—for example, you cannot display a marginal variance-covariance

matrix. Also, the procedure includes both the fixed-effects parameters and the covariance parameters in the optimization for Laplace estimation. Consequently, this setting imposes some restrictions with respect to available options for Laplace estimation. Table 52.3 lists the options that are assumed for METHOD=LAPLACE, and Table 52.4 lists the options that are not compatible with this estimation method.

The section "Maximum Likelihood Estimation Based on Laplace Approximation" contains details about Laplace estimation in PROC GLIMMIX.

Maximum Likelihood with Adaptive Quadrature

If you choose METHOD=QUAD in a generalized linear mixed model, the GLIMMIX procedure approximates the marginal log likelihood with an adaptive Gauss-Hermite quadrature. Compared to METHOD=LAPLACE, the models for which parameters can be estimated by quadrature are further restricted. In addition to the conditional independence assumption and the absence of R-side covariance parameters, it is required that models suitable for METHOD=QUAD can be processed by subjects. (See the section "Processing by Subjects" on page 3957 about how the GLIMMIX procedure determines whether the data can be processed by subjects.) This in turn requires that all RANDOM statements have SUBJECT= effects and in the case of multiple SUBJECT= effects that these form a containment hierarchy.

In a containment hierarchy each effect is contained by another effect, and the effect contained by all is considered "the" effect for subject processing. For example, the SUBJECT= effects in the following statements form a containment hierarchy:

```
proc glimmix;
   class A B block;
   model y = A B A*B;
   random intercept / subject=block;
   random intercept / subject=A*block;
run;
```

The block effect is contained in the A*block interaction and the data are processed by block. The SUBJECT= effects in the following statements do not form a containment hierarchy:

```
proc glimmix;
   class A B block;
   model y = A B A*B;
   random intercept / subject=block;
   random block     / subject=A;
run;
```

The section "Maximum Likelihood Estimation Based on Adaptive Quadrature" on page 3932 contains important details about the computations involved with quadrature approximations. The section "Aspects Common to Adaptive Quadrature and Laplace Approximation" on page 3935 contains information about issues that apply to Laplace and adaptive quadrature, such as the computation of the prediction variance matrix and the determination of starting values.

You can specify the following *quad-options* for METHOD=QUAD in parentheses:

EBDETAILS

reports details about the empirical Bayes suboptimization process should this suboptimization fail.

EBSSFRAC=*r*

specifies the step-shortening fraction to be used while computing empirical Bayes estimates of the random effects. The default value is $r = 0.8$, and it is required that $r > 0$.

EBSSTOL=*r*

specifies the objective function tolerance for determining the cessation of step shortening while computing empirical Bayes estimates of the random effects, $r \geq 0$. The default value is r=1E–8.

EBSTEPS=*n*

specifies the maximum number of Newton steps for computing empirical Bayes estimates of random effects, $n \geq 0$. The default value is n=50.

EBSUBSTEPS=*n*

specifies the maximum number of step shortenings for computing empirical Bayes estimates of random effects. The default value is n=20, and it is required that $n \geq 0$.

EBTOL=*r*

specifies the convergence tolerance for empirical Bayes estimation, $r \geq 0$. The default value is $r = \epsilon \times 1\mathrm{E}4$, where ϵ is the machine precision. This default value equals approximately 1E–12 on most machines.

FASTQUAD

FQ

requests the multilevel adaptive quadrature algorithm proposed by Pinheiro and Chao (2006). For a multilevel model, this algorithm reduces the number of random effects over which the integration for the marginal likelihood computation is carried out. The reduction in the dimension of the integral leads to the reduction in the number of conditional log-likelihood evaluations.

INITPL=*number*

requests that adaptive quadrature commence after performing up to *number* pseudo-likelihood updates. The initial pseudo-likelihood (PL) steps (METHOD=MSPL) can be useful to provide good starting values for the quadrature algorithm. If you choose *number* large enough so that the initial PL estimation converges, the process is equivalent to starting a quadrature from the PL estimates of the fixed-effects and covariance parameters. Because this also makes available the PL random-effects solutions, the adaptive step of the quadrature that determines the number of quadrature points can take this information into account.

Note that you can combine the INITPL option with the NOINITGLM option in the PROC GLIMMIX statement to define a precise path for starting value construction to the GLIMMIX procedure. For example, the following statement generates starting values in these steps:

```
proc glimmix method=quad(initpl=5);
```

1. A GLM without random effects is fit initially to obtain as starting values for the fixed effects. The INITITER= option in the PROC GLIMMIX statement controls the number of iterations in this step.

2. Starting values for the covariance parameters are then obtained by MIVQUE0 estimation (Goodnight 1978a), using the fixed-effects parameter estimates from step 1.
3. With these values up to five pseudo-likelihood updates are computed.
4. The PL estimates for fixed-effects, covariance parameters, and the solutions for the random effects are then used to determine the number of quadrature points and used as the starting values for the quadrature.

The first step (GLM fixed-effects estimates) is omitted, if you modify the previous statement as follows:

```
proc glimmix method=quad(initpl=5) noinitglm;
```

The NOINITGLM option is the default of the pseudo-likelihood methods you select with the METHOD= option.

QCHECK

performs an adaptive recalculation of the objective function (–2 log likelihood) at the solution. The increment of the quadrature points, starting from the number of points used in the optimization, follows the same rules as the determination of the quadrature point sequence at the starting values (see the QFAC= and QMAX= suboptions). For example, the following statement estimates the parameters based on a quadrature with seven nodes in each dimension:

```
proc glimmix method=quad(qpoints=7 qcheck);
```

Because the default search sequence is 1, 3, 5, 7, 9, 11, 21, 31, the QCHECK option computes the –2 log likelihood at the converged solution for 9, 11, 21, and 31 quadrature points and reports relative differences to the converged value and among successive values. The ODS table produced by this option is named QuadCheck.

CAUTION: This option is useful to diagnose the sensitivity of the likelihood approximation at the solution. It does **not** diagnose the stability of the solution under changes in the number of quadrature points. For example, if increasing the number of points from 7 to 9 does not alter the objective function, this does not imply that a quadrature with 9 points would arrive at the same parameter estimates as a quadrature with 7 points.

QFAC=*r*

determines the step size for the quadrature point sequence. If the GLIMMIX procedure determines the quadrature nodes adaptively, the log likelihoods are computed for nodes in a predetermined sequence. If N_{min} and N_{max} denote the values from the QMIN= and QMAX= suboptions, respectively, the sequence for values less than 11 is constructed in increments of 2 starting at N_{min}. Values greater than 11 are incremented in steps of *r*. The default value is *r*=10. The default sequence, without specifying the QMIN=, QMAX=, or QFAC= option, is thus 1, 3, 5, 7, 9, 11, 21, 31. By contrast, the following statement evaluates the sequence 8, 10, 30, 50:

```
proc glimmix method=quad(qmin=8,qmax=51,qfac=20);
```

QMAX=*n*
specifies an upper bound for the number of quadrature points. The default is *n*=31.

QMIN=*n*
specifies a lower bound for the number of quadrature points. The default is *n*=1 and the value must be less than the QMAX= value.

QPOINTS=*n*
determines the number of quadrature points in each dimension of the integral. Note that if there are *r* random effects for each subject, the GLIMMIX procedure evaluates n^r conditional log likelihoods for each observation to compute one value of the objective function. Increasing the number of quadrature nodes can substantially increase the computational burden. If you choose QPOINTS=1, the quadrature approximation reduces to the Laplace approximation. If you do not specify the number of quadrature points, it is determined adaptively by increasing the number of nodes at the starting values. See the section "Aspects Common to Adaptive Quadrature and Laplace Approximation" on page 3935 for details.

QTOL=*r*
specifies a relative tolerance criterion for the successive evaluation of log likelihoods for different numbers of quadrature points. When the GLIMMIX procedure determines the number of quadrature points adaptively, the number of nodes are increased until the QMAX=*n* limit is reached or until two successive evaluations of the log likelihood have a relative change of less than *r*. In the latter case, the lesser number of quadrature nodes is used for the optimization.

When you specify the FASTQUAD suboption to request the multilevel quadrature approximation for the marginal likelihood, gradients for the parameters in the optimization are computed by numerical finite-difference methods rather than analytically. These numerical gradients can in turn lead to Hessian matrices, log likelihoods, parameter estimates, and standard errors that are slightly different from what you would get without the FASTQUAD suboption.

The empirical estimators of covariance matrix of the parameter estimates that are produced by the EMPIRICAL option are not available when you specify the FASTQUAD suboption. Also, when you specify a nominal model, the FASTQUAD option is ignored.

The EBSSFRAC, EBSSTOL, EBSTEPS, EBSUBSTEPS, and EBTOL suboptions affect the suboptimization that leads to the empirical Bayes estimates of the random effects. Under normal circumstances, there is no reason to change from the default values. When the sub-optimizations fail, the optimization process can come to a halt. If the EBDETAILS option is in effect, you might be able to determine why the suboptimization fails and then adjust these values accordingly.

The QMIN, QMAX, QTOL, and QFAC suboptions determine the quadrature point search sequence for the adaptive component of estimation.

As for METHOD=LAPLACE, certain features of the marginal distribution are not available because the marginal likelihood of the data is approximated numerically. For example, you cannot display a marginal variance-covariance matrix. Also, the procedure includes both the fixed-effects and covariance parameters in the optimization for quadrature estimation. Consequently, this setting imposes some restrictions with respect to available options. Table 52.3 lists the options that are assumed for METHOD=QUAD and METHOD=LAPLACE, and Table 52.4 lists the options that are not compatible with these estimation methods.

Table 52.3 Defaults for METHOD=LAPLACE and METHOD=QUAD

Statement	Option
PROC GLIMMIX	NOPROFILE
PROC GLIMMIX	INITGLM
MODEL	NOCENTER

Table 52.4 Options Incompatible with METHOD=LAPLACE and METHOD=QUAD

Statement	Option
PROC GLIMMIX	EXPHESSIAN
PROC GLIMMIX	SCOREMOD
PROC GLIMMIX	SCORING
PROC GLIMMIX	PROFILE
MODEL	DDFM=KENWARDROGER
MODEL	DDFM=SATTERTHWAITE
MODEL	STDCOEF
RANDOM	RESIDUAL
RANDOM _RESIDUAL_	All R-side random effects
RANDOM	V
RANDOM	VC
RANDOM	VCI
RANDOM	VCORR
RANDOM	VI

In addition to the options displayed in Table 52.4, the NOBOUND option in the PROC GLIMMIX and the NOBOUND option in the PARMS statements are not available with METHOD=QUAD. Unbounding the covariance parameter estimates is possible with METHOD=LAPLACE, however.

When the FASTQUAD option is specified in METHOD=QUAD, the empirical estimator of the covariance matrix of the parameter estimates is not available. Also, the FASTQUAD option is not supported for nominal models. Table 52.5 lists the options that are not compatible with the FASTQUAD option.

Table 52.5 Options Incompatible with FASTQUAD Option in METHOD=QUAD

Statement	Options
PROC GLIMMIX	EMPIRICAL
MODEL	DIST=MULTINOMIAL and LINK=GLOGIT

No Random Effects Present

If the model does not contain G-side random effects or contains only a single overdispersion component, then the model belongs to the family of (overdispersed) generalized linear models if the distribution is known or the quasi-likelihood models for independent data if the distribution is not known. The GLIMMIX procedure then estimates model parameters by the following techniques:

- normally distributed data: residual maximum likelihood
- nonnormal data: maximum likelihood
- data with unknown distribution: quasi-likelihood

The METHOD= specification then has only an effect with respect to the divisor used in estimating the overdispersion component. With a residual method, the divisor is $f - k$, where f denotes the sum of the frequencies and k is the rank of $\mathbf{X}$. Otherwise, the divisor is f.

NAMELEN=*number*
specifies the length to which long effect names are shortened. The default and minimum value is 20.

NOBOUND
requests the removal of boundary constraints on covariance and scale parameters in mixed models. For example, variance components have a default lower boundary constraint of 0, and the NOBOUND option allows their estimates to be negative.

The NOBOUND option cannot be used for adaptive quadrature estimation with METHOD=QUAD. The scaling of the quadrature abscissas requires an inverse Cholesky root that is possibly not well defined when the $\mathbf{G}$ matrix of the mixed model is negative definite or indefinite. The Laplace approximation (METHOD=LAPLACE) is not subject to this limitation.

NOBSDETAIL
adds detailed information to the "Number of Observations" table to reflect how many observations were excluded from the analysis and for which reason.

NOCLPRINT< =*number* **>**
suppresses the display of the "Class Level Information" table, if you do not specify *number*. If you specify *number*, only levels with totals that are less than *number* are listed in the table.

NOFIT
suppresses fitting of the model. When the NOFIT option is in effect, PROC GLIMMIX produces the "Model Information," "Class Level Information," "Number of Observations," and "Dimensions" tables. These can be helpful to gauge the computational effort required to fit the model. For example, the "Dimensions" table informs you as to whether the GLIMMIX procedure processes the data by subjects, which is typically more computationally efficient than processing the data as a single subject. See the section "Processing by Subjects" for more information.

If you request a radial smooth with knot selection by k-d tree methods, PROC GLIMMIX also computes the knot locations of the smoother. You can then examine the knots without fitting the model. This enables you to try out different knot construction methods and bucket sizes. See the KNOTMETHOD=KDTREE option (and its suboptions) of the RANDOM statement.

If you combine the NOFIT option with the OUTDESIGN option, you can write the $\mathbf{X}$ and/or $\mathbf{Z}$ matrix of your model to a SAS data set without fitting the model.

NOINITGLM

requests that the starting values for the fixed effects not be obtained by first fitting a generalized linear model. This option is the default for the pseudo-likelihood estimation methods and for the linear mixed model. For the pseudo-likelihood methods, starting values can be implicitly defined based on an initial pseudo-data set derived from the data and the link function. For linear mixed models, starting values for the fixed effects are not necessary. The NOINITGLM option is useful in conjunction with the INITPL= suboption of METHOD=QUAD in order to perform initial pseudo-likelihood steps prior to an adaptive quadrature.

NOITPRINT

suppresses the display of the "Iteration History" table.

NOPROFILE

includes the scale parameter ϕ into the optimization for models that have such a parameter (see Table 52.22). By default, the GLIMMIX procedure profiles scale parameters from the optimization in mixed models. In generalized linear models, scale parameters are not profiled.

NOREML

determines the denominator for the computation of the scale parameter in a GLM for normal data and for overdispersion parameters. By default, the GLIMMIX procedure computes the scale parameter for the normal distribution as

$$\widehat{\phi} = \sum_{i=1}^{n} \frac{f_i (y_i - \widehat{y}_i)^2}{f - k}$$

where k is the rank of $\mathbf{X}$, f_i is the frequency associated with the ith observation, and $f = \sum f_i$. Similarly, the overdispersion parameter in an overdispersed GLM is estimated by the ratio of the Pearson statistic and $(f - k)$. If the NOREML option is in effect, the denominators are replaced by f, the sum of the frequencies. In a GLM for normal data, this yields the maximum likelihood estimate of the error variance. For this case, the NOREML option is a convenient way to change from REML to ML estimation.

In GLMM models fit by pseudo-likelihood methods, the NOREML option changes the estimation method to the nonresidual form. See the METHOD= option for the distinction between residual and nonresidual estimation methods.

ODDSRATIO

OR

requests that odds ratios be added to the output when applicable. Odds ratios and their confidence limits are reported only for models with logit, cumulative logit, or generalized logit link. Specifying the ODDSRATIO option in the PROC GLIMMIX statement has the same effect as specifying the ODDSRATIO option in the MODEL statement and in all LSMEANS statements. Note that the ODDSRATIO option in the MODEL statement has several suboptions that enable you to construct customized odds ratios. These suboptions are available only through the MODEL statement. For details about the interpretation and computation of odds and odds ratios with the GLIMMIX procedure, see the section "Odds and Odds Ratio Estimation" on page 3965.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This ordering determines which parameters in the model correspond to each level in the data, so the ORDER= option can be useful when you use CONTRAST or ESTIMATE statements. This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

When the response variable appears in a CLASS statement, the ORDER= option in the PROC GLIMMIX statement applies to its sort order. Specification of a *response-option* in the MODEL statement overrides the ORDER= option in the PROC GLIMMIX statement. For example, in the following statements the sort order of the wheeze variable is determined by the formatted value (default):

```
proc glimmix order=data;
   class city;
   model wheeze = city age / dist=binary s;
run;
```

The ORDER= option in the PROC GLIMMIX statement has no effect on the sort order of the wheeze variable because it does not appear in the CLASS statement. However, in the following statements the sort order of the wheeze variable is determined by the order of appearance in the input data set because the response variable appears in the CLASS statement:

```
proc glimmix order=data;
   class city wheeze;
   model wheeze = city age / dist=binary s;
run;
```

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUTDESIGN< (*options*) > < =*SAS-data-set* >

creates a data set that contains the contents of the **X** and **Z** matrix. If the data are processed by subjects as shown in the "Dimensions" table, then the **Z** matrix saved to the data set corresponds to a single subject. By default, the GLIMMIX procedure includes in the OUTDESIGN data set the **X** and **Z** matrix (if present) and the variables in the input data set. You can specify the following *options* in parentheses to control the contents of the OUTDESIGN data set:

NAMES

produces tables associating columns in the OUTDESIGN data set with fixed-effects parameter estimates and random-effects solutions.

NOMISS

excludes from the OUTDESIGN data set observations that were not used in the analysis.

NOVAR

excludes from the OUTDESIGN data set variables from the input data set. Variables listed in the BY and ID statements and variables needed for identification of SUBJECT= effects are always included in the OUTDESIGN data set.

X< =*prefix* >

saves the contents of the **X** matrix. The optional *prefix* is used to name the columns. The default naming prefix is "_X".

Z< =*prefix* >

saves the contents of the **Z** matrix. The optional *prefix* is used to name the columns. The default naming prefix is "_Z".

The order of the observations in the OUTDESIGN data set is the same as the order of the input data set. If you do not specify a data set with the OUTDESIGN option, the procedure uses the DATA*n* convention to name the data set.

PCONV=*r*

specifies the parameter estimate convergence criterion for doubly iterative estimation methods. The GLIMMIX procedure applies this criterion to fixed-effects estimates and covariance parameter estimates. Suppose $\widehat{\psi}_i^{(u)}$ denotes the estimate of the ith parameter at the uth optimization. The procedure terminates the doubly iterative process if the largest value

$$2 \times \frac{|\widehat{\psi}_i^{(u)} - \widehat{\psi}_i^{(u-1)}|}{|\widehat{\psi}_i^{(u)}| + |\widehat{\psi}_i^{(u-1)}|}$$

is less than r. To check an absolute convergence criteria as well, you can set the ABSPCONV= option in the PROC GLIMMIX statement. The default value for r is 1E8 times the machine epsilon, a product that equals about 1E–8 on most machines.

Note that this convergence criterion does not affect the convergence criteria applied within any individual optimization. In order to change the convergence behavior within an optimization, you can use the ABSCONV=, ABSFCONV=, ABSGCONV=, ABSXCONV=, FCONV=, or GCONV= option in the NLOPTIONS statement.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

requests that the GLIMMIX procedure produce statistical graphics via ODS Graphics.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc glimmix data=plants;
   class Block Type;
   model StemLength = Block Type;
   lsmeans type / diff=control plots=controlplot;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

For examples of the basic statistical graphics produced by the GLIMMIX procedure and aspects of their computation and interpretation, see the section "ODS Graphics" on page 3990 in this chapter. You can also request statistical graphics for least squares means through the PLOTS option in the LSMEANS statement, which gives you more control over the display compared to the PLOTS option in the PROC GLIMMIX statement.

Global Plot Options

The *global-plot-options* apply to all relevant plots generated by the GLIMMIX procedure. The *global-plot-options* supported by the GLIMMIX procedure are as follows:

OBSNO

uses the data set observation number to identify observations in tooltips, provided that the observation number can be determined. Otherwise, the number displayed in tooltips is the index of the observation as it is used in the analysis within the BY group.

UNPACKPANEL

UNPACK

displays each graph separately. (By default, some graphs can appear together in a single panel.)

Specific Plot Options

The following listing describes the specific plots and their *options*.

ALL

requests that all plots appropriate for the analysis be produced. In models with G-side random effects, residual plots are based on conditional residuals (by using the BLUPs of random effects) on the linear (linked) scale. Plots of least squares means differences are produced for LSMEANS statements without options that would contradict such a display.

ANOMPLOT

ANOM

requests an analysis of means display in which least squares means are compared against an average least squares mean (Ott 1967; Nelson 1982, 1991, 1993). See the DIFF= option in the LSMEANS statement for the computation of this average. Least squares mean ANOM plots are produced only for those fixed effects that are listed in LSMEANS statements that have options that do not contradict the display. For example, if you request ANOM plots with the PLOTS= option in the PROC GLIMMIX statement, the following LSMEANS statements produce analysis of mean plots for effects A and C:

```
lsmeans A / diff=anom;
lsmeans B / diff;
lsmeans C;
```

The DIFF option in the second LSMEANS statement implies all pairwise differences.

When differences against the average LS-mean are adjusted for multiplicity with the ADJUST=NELSON option in the LSMEANS statement, the ANOMPLOT display is adjusted accordingly.

BOXPLOT < (*boxplot-options*) >

requests box plots for the effects in your model that consist of classification effects only. Note that these effects can involve more than one classification variable (interaction and nested effects), but cannot contain any continuous variables. By default, the BOXPLOT request produces box plots of (conditional) residuals for the qualifying effects in the MODEL and RANDOM statements. See the discussion of the *boxplot-options* in a later section for information about how to tune your box plot request.

CONTROLPLOT

CONTROL

requests a display in which least squares means are visually compared against a reference level. LS-mean control plots are produced only for those fixed effects that are listed in LSMEANS statements that have options that do not contradict with the display. For example, the following statements produce control plots for effects A and C if you specify PLOTS=CONTROL in the PROC GLIMMIX statement:

```
lsmeans A / diff=control('1');
lsmeans B / diff;
lsmeans C;
```

The DIFF option in the second LSMEANS statement implies all pairwise differences.

When differences against a control level are adjusted for multiplicity with the ADJUST= option in the LSMEANS statement, the control plot display is adjusted accordingly.

DIFFPLOT< (*diffplot-options*) >

DIFFOGRAM < (*diffplot-options*) >

DIFF< (*diffplot-options*) >

requests a display of all pairwise least squares mean differences and their significance. When constructed from arithmetic means, the display is also known as a "mean-mean scatter plot" (Hsu 1996; Hsu and Peruggia 1994). For each comparison a line segment, centered at the LS-means in the pair, is drawn. The length of the segment corresponds to the projected width of a confidence interval for the least squares mean difference. Segments that fail to cross the 45-degree reference line correspond to significant least squares mean differences.

If you specify the ADJUST= option in the LSMEANS statement, the lengths of the line segments are adjusted for multiplicity.

LS-mean difference plots are produced only for those fixed effects listed in LSMEANS statements that have options that do not conflict with the display. For example, the following statements request differences against a control level for the A effect, all pairwise differences for the B effect, and the least squares means for the C effect:

```
lsmeans A / diff=control('1');
lsmeans B / diff;
lsmeans C;
```

The DIFF= type in the first statement contradicts a display of all pairwise differences. Difference plots are produced only for the B and C effects if you specify PLOTS=DIFF in the PROC GLIMMIX statement.

You can specify the following *diffplot-options*. The ABS and NOABS options determine the positioning of the line segments in the plot. When the ABS option is in effect (this is the default) all line segments are shown on the same side of the reference line. The NOABS option separates comparisons according to the sign of the difference. The CENTER option marks the center point for each comparison. This point corresponds to the intersection of two least squares means. The NOLINES option suppresses the display of the line segments that represent the confidence bounds for the differences of the least squares means. The NOLINES option implies the CENTER option. The default is to draw line segments in the upper portion of the plot area without marking the center point.

MEANPLOT< (*meanplot-options*) >

requests a display of the least squares means of effects specified in LSMEANS statements. The following *meanplot-options* affect the display. Upper and lower confidence limits are plotted when the CL option is used. When the CLBAND option is in effect, confidence limits are shown as bands and the means are connected. By default, least squares means are not joined by lines. You can achieve that effect with the JOIN or CONNECT option. Least squares means are displayed in the same order in which they appear in the "Least Squares Means" table. You can change that order for plotting purposes with the ASCENDING and DESCENDING options. The ILINK option requests that results be displayed on the inverse linked (the data) scale.

Note that there is also a MEANPLOT suboption of the PLOTS= option in the LSMEANS statement. In addition to the *meanplot-options* just described, you can also specify classification effects that give you more control over the display of interaction means through the PLOTBY= and SLICEBY= options. To display interaction means, you typically want to use the MEANPLOT

option in the LSMEANS statement. For example, the next statement requests a plot in which the levels of A are placed on the horizontal axis and the means that belong to the same level of B are joined by lines:

```
lsmeans A*B / plot=meanplot(sliceby=b join);
```

NONE
requests that no plots be produced.

ODDSRATIO < (*oddsratioplot-options*) >
requests a display of odds ratios and their confidence limits when the link function permits the computation of odds ratios (see the ODDSRATIO option in the MODEL statement). Possible suboptions of the ODDSRATIO plot request are described below under the heading "Odds Ratio Plot Options."

RESIDUALPANEL< (*residualplot-options*) >
requests a paneled display constructed from raw residuals. The panel consists of a plot of the residuals against the linear predictor or predicted mean, a histogram with normal density overlaid, a Q-Q plot, and a box plot of the residuals. The *residualplot-options* enable you to specify which type of residual is being graphed. These are further discussed below under the heading "Residual Plot Options."

STUDENTPANEL< (*residualplot-options*) >
requests a paneled display constructed from studentized residuals. The same panel organization is applied as for the RESIDUALPANEL plot type.

PEARSONPANEL< (*residualplot-options*) >
requests a paneled display constructed from Pearson residuals. The same panel organization is applied as for the RESIDUALPANEL plot type.

Residual Plot Options

The *residualplot-options* apply to the RESIDUALPANEL, STUDENTPANEL, and PEARSONPANEL displays. The primary function of these options is to control which type of a residual to display. The four types correspond to *keyword-options* as for output statistics in the OUTPUT statement. The *residualplot-options* take on the following values:

BLUP
CONDITIONAL
uses the predictors of the random effects in computing the residual.

ILINK
NONLINEAR
computes the residual on the inverse linked scale (the data scale).

NOBLUP

MARGINAL

does not use the predictors of the random effects in computing the residual.

NOILINK

LINEAR

computes the residual on the linked scale.

UNPACK

produces separate plots from the elements of the panel.

You can list a plot request one or more times with different options. For example, the following statements request a panel of marginal raw residuals, individual plots generated from a panel of the conditional raw residuals, and a panel of marginal studentized residuals:

```
ods graphics on;
proc glimmix plots=(ResidualPanel(marginal)
                    ResidualPanel(unpack conditional)
                    StudentPanel(marginal));
```

The default is to compute conditional residuals on the linear scale if the model contains G-side random effects (BLUP NOILINK). Not all combinations of the BLUP/NOBLUP and ILINK/NOILINK suboptions are possible for all residual types and models. For details, see the description of output statistics for the OUTPUT statement. Pearson residuals are always displayed against the linear predictor; all other residuals are graphed versus the linear predictor if the NOILINK suboption is in effect (default), and against the corresponding prediction on the mean scale if the ILINK option is in effect. See Table 52.17 for a definition of the residual quantities and exclusions.

Box Plot Options

The *boxplot-options* determine whether box plots are produced for residuals or for residuals and observed values, and for which model effects the box plots are constructed. The available *boxplot-options* are as follows:

BLOCK

BLOCKLEGEND

displays levels of up to four classification variables of the box plot effect by using block legends instead of axis tick values.

BLUP

CONDITIONAL

constructs box plots from conditional residuals—that is, residuals that use the estimated BLUPs of random effects.

FIXED
produces box plots for all fixed effects (MODEL statement) consisting entirely of classification variables.

GROUP
produces box plots for all GROUP= effects in RANDOM statements consisting entirely of classification variables.

ILINK
NONLINEAR
computes the residual on the scale of the data (the inverse linked scale).

NOBLUP
MARGINAL
constructs box plots from marginal residuals.

NOILINK
LINEAR
computes the residual on the linked scale.

NPANELPOS=*number*
specifies the number of box positions on the graphic and provides the capability to break a box plot into multiple graphics. If *number* is negative, no balancing of the number of boxes takes place and *number* is the maximum number of boxes per graphic. If *number* is positive, the number of boxes per graphic is balanced. For example, suppose that variable A has 125 levels. The following statements request that the number of boxes per plot results be balanced and result in six plots with 18 boxes each and one plot with 17 boxes:

```
ods graphics on;
proc glimmix plots=boxplot(npanelpos=20);
   class A;
   model y = A;
run;
```

If *number* is zero (this is the default), all levels of the effect are displayed in a single plot.

OBSERVED
adds box plots of the observed data for the selected effects.

PEARSON
constructs box plots from Pearson residuals rather than from the default residuals.

PSEUDO
adds box plots of the pseudo-data for the selected effects. This option is available only for the pseudo-likelihood estimation methods that construct pseudo-data.

RANDOM
produces box plots for all effects in RANDOM statements that consist entirely of classification variables. This does not include effects specified in the GROUP= or SUBJECT= option of the RANDOM statements.

RAW
constructs box plots from raw residuals (observed minus predicted).

STUDENT
constructs box plots from studentized residuals rather than from the default residuals.

SUBJECT
produces box plots for all SUBJECT= effects in RANDOM statements consisting entirely of classification variables.

USEINDEX
uses as the horizontal axis label the index of the effect level, rather than the formatted value(s). For classification variables with many levels or model effects that involve multiple classification variables, the formatted values identifying the effect levels might take up too much space as axis tick values, leading to extensive thinning. The USEINDEX option replaces tick values constructed from formatted values with the internal level number.

By default, box plots of residuals are constructed from the raw conditional residuals (on the linked scale) in linear mixed models and from Pearson residuals in all other models. Note that not all combinations of the BLUP/NOBLUP and ILINK/NOILINK suboptions are possible for all residual types and models. For details, see the description of output statistics for the OUTPUT statement.

Odds Ratio Plot Options

The *oddsratioplot-options* determine the display of odds ratios and their confidence limits. The computation of the odds ratios follows the ODDSRATIO option in the MODEL statement. The available *oddsratioplot-options* are as follows:

LOGBASE= 2 | E | 10
log-scales the odds ratio axis.

NPANELPOS=*n*
provides the capability to break an odds ratio plot into multiple graphics having at most $|n|$ odds ratios per graphic. If n is positive, then the number of odds ratios per graphic is balanced. If n is negative, then no balancing of the number of odds ratios takes place. For example, suppose you want to display 21 odds ratios. Then NPANELPOS=20 displays two plots, the first with 11 and the second with 10 odds ratios, and NPANELPOS=–20 displays 20 odds ratios in the first plot and a single odds ratio in the second. If n=0 (this is the default), then all odds ratios are displayed in a single plot.

ORDER=ASCENDING | DESCENDING
displays the odds ratios in sorted order. By default the odds ratios are displayed in the order in which they appear in the "Odds Ratio Estimates" table.

RANGE=(< *min* > < ,*max* >) | CLIP
specifies the range of odds ratios to display. If you specify RANGE=CLIP, then the confidence intervals are clipped and the range contains the minimum and maximum odds ratios. By default the range of view captures the extent of the odds ratio confidence intervals.

STATS
adds the numeric values of the odds ratio and its confidence limits to the graphic.

PROFILE
requests that scale parameters be profiled from the optimization, if possible. This is the default for generalized linear mixed models. In generalized linear models with normally distributed data, you can use the PROFILE option to request profiling of the residual variance.

SCOREMOD
requests that the Hessian matrix in GLMMs be based on a modified scoring algorithm, provided that PROC GLIMMIX is in scoring mode when the Hessian is evaluated. The procedure is in scoring mode during iteration, if the optimization technique requires second derivatives, the SCORING=*n* option is specified, and the iteration count has not exceeded *n*. The procedure also computes the expected (scoring) Hessian matrix when you use the EXPHESSIAN option in the PROC GLIMMIX statement.

The SCOREMOD option has no effect if the SCORING= or EXPHESSIAN option is not specified. The nature of the SCOREMOD modification to the expected Hessian computation is shown in Table 52.25, in the section "Pseudo-likelihood Estimation Based on Linearization" on page 3924. The modification can improve the convergence behavior of the GLMM compared to standard Fisher scoring and can provide a better approximation of the variability of the covariance parameters. For more details, see the section "Estimated Precision of Estimates" on page 3926.

SCORING=*number*
requests that Fisher scoring be used in association with the estimation method up to iteration *number*. By default, no scoring is applied. When you use the SCORING= option and PROC GLIMMIX converges without stopping the scoring algorithm, the procedure uses the expected Hessian matrix to compute approximate standard errors for the covariance parameters instead of the observed Hessian. If necessary, the standard errors of the covariance parameters as well as the output from the ASYCOV and ASYCORR options are adjusted.

If scoring stopped prior to convergence and you want to use the expected Hessian matrix in the computation of standard errors, use the EXPHESSIAN option in the PROC GLIMMIX statement.

Scoring is not possible in models for nominal data. It is also not possible for GLMs with unknown distribution or for those outside the exponential family. If you perform quasi-likelihood estimation, the GLIMMIX procedure is always in scoring mode and the SCORING= option has no effect. See the section "Quasi-likelihood for Independent Data" for a description of the types of models where GLIMMIX applies quasi-likelihood estimation.

The SCORING= option has no effect for optimization methods that do not involve second derivatives. See the TECHNIQUE= option in the NLOPTIONS statement and the section "Choosing an Optimization Algorithm" on page 545 in Chapter 20, "Shared Concepts and Topics," for details about first- and second-order algorithms.

SINGCHOL=*number*
tunes the singularity criterion in Cholesky decompositions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGRES=*number*
sets the tolerance for which the residual variance is considered to be zero. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGULAR=*number*
tunes the general singularity criterion applied by the GLIMMIX procedure in divisions and inversions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

STARTGLM
is an alias of the INITGLM option.

SUBGRADIENT< =*SAS-data-set* **>**
SUBGRAD< =*SAS-data-set* **>**
creates a data set with information about the gradient of the objective function. The contents and organization of the SUBGRADIENT= data set depend on the type of model. The following paragraphs describe the SUBGRADIENT= data set for the two major estimation modes. See the section "GLM Mode or GLMM Mode" on page 3941 for details about the estimation modes of the GLIMMIX procedure.

GLMM Mode If the GLIMMIX procedure operates in GLMM mode, the SUBGRADIENT= data set contains as many observations as there are usable subjects in the analysis. The maximum number of usable subjects is displayed in the "Dimensions" table. Gradient information is not written to the data set for subjects who do not contribute valid observations to the analysis. Note that the objective function in the "Iteration History" table is in terms of the –2 log (residual, pseudo-) likelihood. The gradients in the SUBGRADIENT= data set are gradients of that objective function.

The gradients are evaluated at the final solution of the estimation problem. If the GLIMMIX procedure fails to converge, then the information in the SUBGRADIENT= data set corresponds to the gradient evaluated at the last iteration or optimization.

The number of gradients saved to the SUBGRADIENT= data set equals the number of parameters in the optimization. For example, with METHOD=LAPLACE or METHOD=QUAD the fixed-effects parameters and the covariance parameters take part in the optimization. The order in which the gradients appear in the data set equals the order in which the gradients are displayed when the ITDETAILS option is in effect: gradients for fixed-effects parameters precede those for covariance parameters, and gradients are not reported for singular columns in the $\mathbf{X}'\mathbf{X}$ matrix. In models where the residual variance is profiled from the optimization, a subject-specific gradient is not reported for the residual variance. To decompose this gradient by subjects, add the NOPROFILE option in the PROC GLIMMIX statement. When the subject-specific gradients in the SUBGRADIENT= data set are summed, the totals equal the values reported by the GRADIENT option.

GLM Mode When you fit a generalized linear model (GLM) or a GLM with overdispersion, the SUBGRADIENT= data set contains the observation-wise gradients of the negative

log-likelihood function with respect to the parameter estimates. Note that this corresponds to the objective function in GLMs as displayed in the "Iteration History" table. However, the gradients displayed in the "Iteration History" for GLMs—when the ITDETAILS option is in effect—are possibly those of the centered and scaled coefficients. The gradients reported in the "Parameter Estimates" table and in the SUBGRADIENT= data set are gradients with respect to the uncentered and unscaled coefficients.

The gradients are evaluated at the final estimates. If the model does not converge, the gradients contain missing values. The gradients appear in the SUBGRADIENT= data set in the same order as in the "Parameter Estimates" table, with singular columns removed.

The variables from the input data set are added to the SUBGRADIENT= data set in GLM mode. The data set is organized in the same way as the input data set; observations that do not contribute to the analysis are transferred to the SUBGRADIENT= data set, but gradients are calculated only for observations that take part in the analysis. If you use an ID statement, then only the variables in the ID statement are transferred to the SUBGRADIENT= data set.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC GLIMMIX to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the GLIMMIX procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Since sorting the data changes the order in which PROC GLIMMIX reads observations, the sort order for the levels of the CLASS variables might be affected if you have also specified ORDER=DATA in the PROC GLIMMIX statement. This, in turn, affects specifications in the CONTRAST, ESTIMATE, or LSMESTIMATE statement.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(REF=** *option***)** > . . . < *variable* < **(REF=** *option***)** > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC GLIMMIX statement.

You can specify the following REF= option to indicate how the levels of an individual classification variable are to be ordered by enclosing it in parentheses after the variable name:

REF=*'level'* **| FIRST | LAST**

specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC GLIMMIX's singular parameterization. You can specify the *level* of the variable to use as the reference level; specify a value that corresponds to the formatted value of the variable if a format is assigned. Alternatively, you can specify REF=FIRST to designate that the first ordered level serve as the reference, or REF=LAST to designate that the last ordered level serve as the reference. To specify that REF=FIRST or REF=LAST be used for all classification variables, use the REF= *global-option* after the slash (/) in the CLASS statement.

You can specify the following *global-options* in the CLASS statement after a slash (/):

REF=FIRST | LAST

specifies a level of all classification variables to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC GLIMMIX's singular parameterization. Specify REF=FIRST to designate that the first ordered level for each classification variable serve as the reference. Specify REF=LAST to designate that the last ordered level serve as the reference. This option applies to all the variables specified in the CLASS statement. To specify different reference levels for different classification variables, use REF= options for individual variables.

TRUNCATE

specifies that class levels be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 52.6 summarizes the *options* available in the CODE statement.

Table 52.6 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

CONTRAST Statement

CONTRAST '*label*' *contrast-specification*
< **,** *contrast-specification* > < **,** . . . >
< / *options* > **;**

The CONTRAST statement provides a mechanism for obtaining custom hypothesis tests. It is patterned after the CONTRAST statement in PROC MIXED and enables you to select an appropriate inference space (McLean, Sanders, and Stroup 1991). The GLIMMIX procedure gives you greater flexibility in entering contrast coefficients for random effects, however, because it permits the usual *value*-oriented positional syntax for entering contrast coefficients, as well as a level-oriented syntax that simplifies entering coefficients for interaction terms and is designed to work with constructed effects that are defined through the EFFECT statement. The differences between the traditional and new-style coefficient syntax are explained in detail in the section "Positional and Nonpositional Syntax for Contrast Coefficients" on page 3973.

You can test the hypothesis $\mathbf{L}'\boldsymbol{\phi} = \mathbf{0}$, where $\mathbf{L}' = [\mathbf{K}'\ \mathbf{M}']$ and $\boldsymbol{\phi}' = [\boldsymbol{\beta}'\ \boldsymbol{\gamma}']$, in several inference spaces. The inference space corresponds to the choice of $\mathbf{M}$. When $\mathbf{M} = \mathbf{0}$, your inferences apply to the entire population from which the random effects are sampled; this is known as the *broad* inference space. When all elements of $\mathbf{M}$ are nonzero, your inferences apply only to the observed levels of the random effects. This is known as the *narrow* inference space, and you can also choose it by specifying all of the random effects as fixed.

The GLM procedure uses the narrow inference space. Finally, by zeroing portions of **M** corresponding to selected main effects and interactions, you can choose *intermediate* inference spaces. The broad inference space is usually the most appropriate; it is used when you do not specify random effects in the CONTRAST statement.

In the CONTRAST statement,

label identifies the contrast in the table. A label is required for every contrast specified. Labels can be up to 200 characters and must be enclosed in quotes.

contrast-specification identifies the fixed effects and random effects and their coefficients from which the **L** matrix is formed. The syntax representation of a *contrast-specification* is
< *fixed-effect values* . . . > < | *random-effect values* . . . >

fixed-effect identifies an effect that appears in the MODEL statement. The keyword INTERCEPT can be used as an effect when an intercept is fitted in the model. You do not need to include all effects that are in the MODEL statement.

random-effect identifies an effect that appears in the RANDOM statement. The first random effect must follow a vertical bar (|); however, random effects do not have to be specified.

values are constants that are elements of the **L** matrix associated with the fixed and random effects. There are two basic methods of specifying the entries of the **L** matrix. The traditional representation—also known as the positional syntax—relies on entering coefficients in the position they assume in the **L** matrix. For example, in the following statements the elements of **L** associated with the b main effect receive a 1 in the first position and a –1 in the second position:

```
class a b;
model y = a b a*b;
contrast 'B at A2' b 1 -1  a*b 0  0  1 -1;
```

The elements associated with the interaction receive a 1 in the third position and a –1 in the fourth position. In order to specify coefficients correctly for the interaction term, you need to know how the levels of a and b vary in the interaction, which is governed by the order of the variables in the CLASS statement. The nonpositional syntax is designed to make it easier to enter coefficients for interactions and is necessary to enter coefficients for effects constructed with the EFFECT statement. In square brackets you enter the coefficient followed by the associated levels of the CLASS variables. If B has two and A has three levels, the previous CONTRAST statement, by using nonpositional syntax for the interaction term, becomes

```
contrast 'B at A2' b 1 -1 a*b [1, 2 1] [-1, 2 2];
```

It assigns value 1 to the interaction where A is at level 2 and B is at level 1, and it assigns –1 to the interaction where both classification variables are at level 2. The comma separating the entry for the **L** matrix from the level indicators is optional. Further details about the nonpositional contrast syntax and its use with constructed effects can be found in the section "Positional and Nonpositional Syntax for Contrast Coefficients" on page 3973. Nonpositional syntax is available only for fixed-effects coefficients.

The rows of $\mathbf{L}'$ are specified in order and are separated by commas. The rows of the $\mathbf{K}'$ component of $\mathbf{L}'$ are specified on the left side of the vertical bars (|). These rows test the fixed effects and are, therefore, checked for estimability. The rows of the $\mathbf{M}'$ component of $\mathbf{L}'$ are specified on the right side of the vertical bars. They test the random effects, and no estimability checking is necessary.

If PROC GLIMMIX finds the fixed-effects portion of the specified contrast to be nonestimable (see the SINGULAR= option), then it displays missing values for the test statistics.

If the elements of $\mathbf{L}$ are not specified for an effect that contains a specified effect, then the elements of the unspecified effect are automatically "filled in" over the levels of the higher-order effect. This feature is designed to preserve estimability for cases where there are complex higher-order effects. The coefficients for the higher-order effect are determined by equitably distributing the coefficients of the lower-level effect as in the construction of least squares means. In addition, if the intercept is specified, it is distributed over all classification effects that are not contained by any other specified effect. If an effect is not specified and does not contain any specified effects, then all of its coefficients in $\mathbf{L}$ are set to 0. You can override this behavior by specifying coefficients for the higher-order effect.

If too many values are specified for an effect, the extra ones are ignored; if too few are specified, the remaining ones are set to 0. If no random effects are specified, the vertical bar can be omitted; otherwise, it must be present. If a SUBJECT effect is used in the RANDOM statement, then the coefficients specified for the effects in the RANDOM statement are equitably distributed across the levels of the SUBJECT effect. You can use the E option to see exactly what $\mathbf{L}$ matrix is used.

PROC GLIMMIX handles missing level combinations of classification variables similarly to PROC GLM and PROC MIXED. These procedures delete fixed-effects parameters corresponding to missing levels in order to preserve estimability. However, PROC MIXED and PROC GLIMMIX do not delete missing level combinations for random-effects parameters, because linear combinations of the random-effects parameters are always estimable. These conventions can affect the way you specify your CONTRAST coefficients.

The CONTRAST statement computes the statistic

$$F = \frac{\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}' \mathbf{L}(\mathbf{L}'\mathbf{C}\mathbf{L})^{-1}\mathbf{L}' \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}}{r}$$

where $r = \mathrm{rank}(\mathbf{L}'\mathbf{C}\mathbf{L})$, and approximates its distribution with an F distribution unless DDFM=NONE. If you select DDFM=NONE as the degrees-of-freedom method in the MODEL statement, and if you do not assign degrees of freedom to the contrast with the DF= option, then PROC GLIMMIX computes the test statistic $r \times F$ and approximates its distribution with a chi-square distribution. In the expression for F, $\mathbf{C}$ is an estimate of $\mathrm{Var}[\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\gamma}} - \boldsymbol{\gamma}]$; see the section "Estimated Precision of Estimates" on page 3926 and the section "Aspects Common to Adaptive Quadrature and Laplace Approximation" on page 3935 for details about the computation of $\mathbf{C}$ in a generalized linear mixed model.

The numerator degrees of freedom in the F approximation and the degrees of freedom in the chi-square approximation are equal to r. The denominator degrees of freedom are taken from the "Tests of Fixed Effects" table and correspond to the final effect you list in the CONTRAST statement. You can change the denominator degrees of freedom by using the DF= option.

You can specify the following *options* in the CONTRAST statement after a slash (/).

BYCATEGORY

BYCAT

requests that in models for nominal data (generalized logit models) the contrasts not be combined across response categories but reported separately for each category. For example, assume that the response variable Style is multinomial with three (unordered) categories. The following GLIMMIX statements fit a generalized logit model relating the preferred style of instruction to school and educational program effects:

```
proc glimmix data=school;
   class School Program;
   model Style(order=data) = School Program / s ddfm=none
                             dist=multinomial link=glogit;
   freq Count;
   contrast 'School 1 vs. 2' school 1 -1;
   contrast 'School 1 vs. 2' school 1 -1 / bycat;
run;
```

The first contrast compares school effects in all categories. This is a two-degrees-of-freedom contrast because there are two nonredundant categories. The second CONTRAST statement produces two single-degree-of-freedom contrasts, one for each nonreference Style category.

The BYCATEGORY option has no effect unless your model is a generalized (mixed) logit model.

CHISQ

requests that chi-square tests be performed for all contrasts in addition to any F tests. A chi-square statistic equals its corresponding F statistic times the numerator degrees of freedom, and these same degrees of freedom are used to compute the p-value for the chi-square test. This p-value will always be less than that for the F test, because it effectively corresponds to an F test with infinite denominator degrees of freedom.

DF=*number*

specifies the denominator degrees of freedom for the F test. For the degrees of freedom methods DDFM=BETWITHIN, DDFM=CONTAIN, and DDFM=RESIDUAL, the default is the denominator degrees of freedom taken from the "Tests of Fixed Effects" table and corresponds to the final effect you list in the CONTRAST statement. For DDFM=NONE, infinite denominator degrees of freedom are assumed by default, and for DDFM=SATTERTHWAITE and DDFM=KENWARDROGER, the denominator degrees of freedom are computed separately for each contrast.

E

requests that the $\mathbf{L}$ matrix coefficients for the contrast be displayed.

GROUP *coeffs*

sets up random-effect contrasts between different groups when a GROUP= variable appears in the RANDOM statement. By default, CONTRAST statement coefficients on random effects are distributed equally across groups. If you enter a multiple row contrast, you can also enter multiple rows for the GROUP coefficients. If the number of GROUP coefficients is less than the number of contrasts in the CONTRAST statement, the GLIMMIX procedure cycles through the GROUP coefficients. For example, the following two statements are equivalent:

```
contrast 'Trt 1 vs 2 @ x=0.4' trt 1 -1  0 | x 0.4,
                              trt 1  0 -1 | x 0.4,
                              trt 1 -1  0 | x 0.5,
                              trt 1  0 -1 | x 0.5 /
                group 1 -1, 1 0 -1, 1 -1, 1 0 -1;

contrast 'Trt 1 vs 2 @ x=0.4' trt 1 -1  0 | x 0.4,
                              trt 1  0 -1 | x 0.4,
                              trt 1 -1  0 | x 0.5,
                              trt 1  0 -1 | x 0.5 /
                group 1 -1, 1 0 -1;
```

SINGULAR=*number*

tunes the estimability checking. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the largest absolute value of the elements of $\mathbf{v}$. If ABS($\mathbf{K}' - \mathbf{K}'\mathbf{T}$) is greater than c**number* for any row of $\mathbf{K}'$ in the contrast, then $\mathbf{K}'\boldsymbol{\beta}$ is declared nonestimable. Here, $\mathbf{T}$ is the Hermite form matrix $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$, and c is ABS($\mathbf{K}'$), except when it equals 0, and then c is 1. The value for *number* must be between 0 and 1; the default is 1E–4.

SUBJECT *coeffs*

sets up random-effect contrasts between different subjects when a SUBJECT= variable appears in the RANDOM statement. By default, CONTRAST statement coefficients on random effects are distributed equally across subjects. Listing subject coefficients for multiple row CONTRAST statements follows the same rules as for GROUP coefficients.

COVTEST Statement

COVTEST <'*label*'> <*test-specification*> <*/ options*> **;**

The COVTEST statement provides a mechanism to obtain statistical inferences for the covariance parameters. Significance tests are based on the ratio of (residual) likelihoods or pseudo-likelihoods. Confidence limits and bounds are computed as Wald or likelihood ratio limits. You can specify multiple COVTEST statements.

The likelihood ratio test is obtained by fitting the model subject to the constraints imposed by the *test-specification*. The test statistic is formed as twice the difference of the (possibly restricted) log (pseudo-) likelihoods of the full and the reduced models. Note that fitting the null model does not necessarily require fewer computer resources compared to fitting the full model. The optimization settings for refitting the model are the same as for the full model and can be controlled with the NLOPTIONS statement.

Common questions in mixed modeling are whether variance components are zero, whether random effects are independent, and whether rows (columns) can be added or removed from an unstructured covariance matrix. When the parameters under the null hypothesis fall on the boundary of the parameter space, the distribution of the likelihood ratio statistic can be a complicated mixture of distributions. In certain situations it is known to be a relatively straightforward mixture of central chi-square distributions. When the GLIMMIX procedure recognizes the model and hypothesis as a case for which the mixture is readily available, the p-value of the likelihood ratio test is determined accordingly as a linear combination of central chi-square probabilities. The Note column in the "Likelihood Ratio Tests for Covariance Parameters" table along with the table's footnotes informs you about when mixture distributions are used in the calculation of p-values. You can find important statistical and computational details about likelihood ratio testing of covariance parameters with the GLIMMIX procedure in the section "Statistical Inference for Covariance Parameters" on page 3942.

In generalized linear mixed models that depend on pseudo-data, the GLIMMIX procedure fits the null model for a test of covariance parameters to the final pseudo-data of the converged optimization.

Table 52.7 summarizes the *options* available in the COVTEST statement.

Table 52.7 COVTEST Statement Options

Option	Description
Test Specification	
TESTDATA=	Reads in covariance parameter values from a SAS data set
GENERAL	Provides a general facility to test linear combinations of covariance parameters
Covariance Test Options	
CL	Requests confidence limits for the covariance parameter estimates
CLASSICAL	Computes the likelihood ratio test p-value using the classical method
DF=	Specifies the degrees of freedom
ESTIMATES	Displays the estimates of the covariance parameters under the null hypothesis
MAXITER=	Limits the number of iterations
PARMS	Displays the values of the covariance parameters under the null hypothesis
RESTART	Specifies that starting values for the covariance parameters
TOLERANCE=	Sets the tolerance level of the parameter space boundary
WALD	Produces Wald Z tests
WGHT=	Supplies weights for the computation of p-values

Test Specification

The *test-specification* in the COVTEST statement draws on *keywords* that represent a particular null hypothesis, lists or data sets of parameter values, or general contrast specifications. Valid *keywords* are as follows:

GLM | INDEP tests the model against a null model of complete independence. All G-side covariance parameters are eliminated and the R-side covariance structure is reduced to a diagonal structure.

DIAGG tests for a diagonal **G** matrix by constraining off-diagonal elements in **G** to zero. The R-side structure is not modified.

DIAGR | CINDEP tests for conditional independence by reducing the R-side covariance structure to diagonal form. The G-side structure is not modified.

HOMOGENEITY tests homogeneity of covariance parameters across groups by imposing equality constraints. For example, the following statements fit a one-way model with heterogeneous variances and test whether the model could be reduced to a one-way analysis with the same variance across groups:

```
proc glimmix;
   class A;
   model y = a;
   random _residual_ / group=A;
   covtest 'common variance' homogeneity;
run;
```

See Example 52.9 for an application with groups and unstructured covariance matrices.

START | INITIAL compares the final estimates to the starting values of the covariance parameter estimates. This option is useful, for example, if you supply starting values in the PARMS statement and want to test whether the optimization produced significantly better values. In GLMMs based on pseudo-data, the likelihoods that use the starting and the final values are based on the final pseudo-data.

ZEROG tests whether the **G** matrix can be reduced to a zero matrix. This eliminates all G-side random effects from the model.

Only a single *keyword* is permitted in the COVTEST statement. To test more complicated hypotheses, you can formulate tests with the following specifications.

TESTDATA=*data-set*

TDATA=*data-set*

reads in covariance parameter values from a SAS data set. The data set should contain the numerical variable Estimate or numerical variables named Covp*i*. The GLIMMIX procedure associates the values for Covp*i* with the *i*th covariance parameter.

For data sets containing the numerical variable Estimate, the GLIMMIX procedure fixes the *i*th covariance parameter value at the value of the *i*th observation in the data set. A missing value indicates not to fix the particular parameter. PROC GLIMMIX performs one likelihood ratio test for the TESTDATA= data set.

For data sets containing numerical variables named Covp*i*, the procedure performs one likelihood ratio test for each observation in the TESTDATA= data set. You do not have to specify a Covp*i* variable for every covariance parameter. If the value for the variable is not missing, PROC GLIMMIX fixes the associated covariance parameter in the null model. Consider the following statements:

```
data TestDataSet;
   input covp1 covp2 covp3;
   datalines;
. 0 .
0 0 .
. 0 0
0 0 0
;

proc glimmix method=mspl;
   class subject x;
   model y = x age x*age;
   random intercept age / sub=subject type=un;
```

```
   covtest testdata=TestDataSet;
run;
```

Because the **G** matrix is a (2×2) unstructured matrix, the first observation of the TestDataSet corresponds to zeroing the covariance between the random intercept and the random slope. When the reduced model is fit, the variances of the intercept and slope are reestimated. The second observation reduces the model to one with only a random slope in age. The third reduces the model to a random intercept model. The last observation eliminates the **G** matrix altogether.

Note that the tests associated with the first and last set of covariance parameters in TestDataSet can also be obtained by using *keywords*:

```
proc glimmix;
   class subject x;
   model y = x age x*age;
   random intercept age / sub=subject type=un;
   covtest DiagG;
   covtest GLM;
run;
```

value-list

supplies a list of values at which to fix the covariance parameters. A missing value in the list indicates that the covariance parameter is not fixed. If the list is shorter than the number of covariance parameters, missing values are assumed for all parameters not specified. The COVTEST statements that test the random intercept and random slope in the previous example are as follows:

```
proc glimmix;
   class subject x;
   model y = x age x*age;
   random intercept age / sub=subject type=un;
   covtest 0 0;
   covtest . 0 0;
run;
```

GENERAL *coefficients* **<,***coefficients***> <,... >**

CONTRAST *coefficients* **<,***coefficients***> <,... >**

provides a general facility to test linear combinations of covariance parameters. You can specify one or more sets of coefficients. The position of a coefficient in the list corresponds to the position of the parameter in the "Covariance Parameter Estimates" table. The linear combination of covariance parameters that is implied by each set of coefficients is tested against zero. If the list of coefficients is shorter than the number of covariance parameters, a zero coefficient is assumed for the remaining parameters.

For example, in a heterogeneous variance model with four groups, the following statements test the simultaneous hypothesis $H: \sigma_1^2 = \sigma_2^2, \sigma_3^2 = \sigma_4^2$:

```
proc glimmix;
   class A;
   model y = a;
   random _residual_ / group=A;
   covtest 'pair-wise homogeneity'
           general  1 -1  0  0,
                    0  0  1 -1;
run;
```

In a repeated measures study with four observations per subject, the COVTEST statement in the following example tests whether the four correlation parameters are identical:

```
proc glimmix;
   class subject drug time;
   model y = drug time drug*time;
   random _residual_ / sub=subject type=unr;
   covtest 'Homogeneous correlation'
        general  0 0 0 0 1 -1            ,
                 0 0 0 0 1  0 -1         ,
                 0 0 0 0 1  0  0 -1      ,
                 0 0 0 0 1  0  0  0 -1   ,
                 0 0 0 0 1  0  0  0  0 -1;
run;
```

Notice that the variances (the first four covariance parameters) are allowed to vary. The null model for this test is thus a heterogeneous compound symmetry model.

The degrees of freedom associated with these general linear hypotheses are determined as the rank of the matrix $\mathbf{LL}'$, where $\mathbf{L}$ is the $k \times q$ matrix of coefficients and q is the number of covariance parameters. Notice that the coefficients in a row do not have to sum to zero. The following statement tests $H\colon \theta_1 = 3\theta_2,\ \theta_3 = 0$:

```
covtest general 1 -3, 0 0 1;
```

Covariance Test Options

You can specify the following *options* in the COVTEST statement after a slash (/).

CL<(*suboptions*)>

requests confidence limits or bounds for the covariance parameter estimates. These limits are displayed as extra columns in the "Covariance Parameter Estimates" table.

The following *suboptions* determine the computation of confidence bounds and intervals. See the section "Statistical Inference for Covariance Parameters" on page 3942 for details about constructing likelihood ratio confidence limits for covariance parameters with PROC GLIMMIX.

ALPHA=*number*

determines the confidence level for constructing confidence limits for the covariance parameters. The value of *number* must be between 0 and 1, the default is 0.05, and the confidence level is 1 – *number*.

LOWERBOUND

LOWER

requests lower confidence bounds.

TYPE=*method*

determines how the GLIMMIX procedure constructs confidence limits for covariance parameters. The valid methods are PLR (or PROFILE), ELR (or ESTIMATED), and WALD. TYPE=PLR (TYPE=PROFILE) requests confidence bounds by inversion of the profile (restricted) likelihood ratio (PLR). If θ is the parameter of interest, L denotes the likelihood (possibly restricted and possibly a pseudo-likelihood), and $\boldsymbol{\theta}_2$ is the vector of the remaining (nuisance) parameters, then the profile likelihood is defined as

$$L(\boldsymbol{\theta}_2|\widetilde{\theta}) = \sup_{\boldsymbol{\theta}_2} L(\widetilde{\theta}, \boldsymbol{\theta}_2)$$

for a given value $\widetilde{\theta}$ of θ. If $L(\widehat{\boldsymbol{\theta}})$ is the overall likelihood evaluated at the estimates $\widehat{\boldsymbol{\theta}}$, the $(1-\alpha) \times 100\%$ confidence region for θ satisfies the inequality

$$2\left\{L(\widehat{\boldsymbol{\theta}}) - L(\boldsymbol{\theta}_2|\widetilde{\theta})\right\} \leq \chi^2_{1,(1-\alpha)}$$

where $\chi^2_{1,(1-\alpha)}$ is the cutoff from a chi-square distribution with one degree of freedom and α probability to its right. If a residual scale parameter ϕ is profiled from the estimation, and θ is expressed in terms of a ratio with ϕ during estimation, then profile likelihood confidence limits are constructed for the ratio of the parameter with the residual variance. A column showing the ratio estimates is added to the "Covariance Parameter Estimates" table in this case. To obtain profile likelihood ratio limits for the parameters, rather than their ratios, and for the residual variance, use the NOPROFILE option in the PROC GLIMMIX statement. Also note that METHOD=LAPLACE or METHOD=QUAD implies the NOPROFILE option.

The TYPE=ELR (TYPE=ESTIMATED) option constructs bounds from the estimated likelihood (Pawitan 2001), where nuisance parameters are held fixed at the (restricted) maximum (pseudo-) likelihood estimates of the model. Estimated likelihood intervals are computationally less demanding than profile likelihood intervals, but they do not take into account the variability of the nuisance parameters or the dependence among the covariance parameters. See the section "Statistical Inference for Covariance Parameters" on page 3942 for a geometric interpretation and comparison of ELR versus PLR confidence bounds. A $(1-\alpha) \times 100\%$ confidence region based on the estimated likelihood is defined by the inequality

$$2\left\{L(\widehat{\boldsymbol{\theta}}) - L(\widetilde{\theta}, \widehat{\boldsymbol{\theta}}_2)\right\} \leq \chi^2_{1,(1-\alpha)}$$

where $L(\widetilde{\theta}, \widehat{\boldsymbol{\theta}}_2)$ is the likelihood evaluated at $\widetilde{\theta}$ and the component of $\widehat{\boldsymbol{\theta}}$ that corresponds to $\boldsymbol{\theta}_2$. Estimated likelihood ratio intervals tend to perform well when the correlations between the parameter of interest and the nuisance parameters is small. Their coverage probabilities can fall short of the nominal coverage otherwise. You can display the correlation matrix of the covariance parameter estimates with the ASYCORR option in the PROC GLIMMIX statement.

If you choose TYPE=PLR or TYPE=ELR, the GLIMMIX procedure reports the right-tail probability of the associated single-degree-of-freedom likelihood ratio test along with the confidence bounds. This helps you diagnose whether solutions to the inequality could be found. If the reported probability exceeds α, the associated bound does not meet the inequality. This might occur, for example, when the parameter space is bounded and the likelihood at the boundary values has not dropped by a sufficient amount to satisfy the test inequality.

The TYPE=WALD method requests confidence limits based on the Wald-type statistic $Z_\theta = \widehat{\theta}/\mathrm{ease}(\widehat{\theta})$, where ease is the estimated asymptotic standard error of the covariance parameter. For parameters that have a lower boundary constraint of zero, a Satterthwaite approximation is used to construct limits of the form

$$\frac{\nu\widehat{\theta}}{\chi^2_{\nu,1-\alpha/2}} \le \theta \le \frac{\nu\widehat{\theta}}{\chi^2_{\nu,\alpha/2}}$$

where $\nu = 2Z^2$, and the denominators are quantiles of the χ^2 distribution with ν degrees of freedom. See Milliken and Johnson (1992) and Burdick and Graybill (1992) for similar techniques. For all other parameters, Wald Z-scores and normal quantiles are used to construct the limits. Such limits are also provided for variance components if you specify the NOBOUND option in the PROC GLIMMIX statement or the PARMS statement.

UPPERBOUND

UPPER

requests upper confidence bounds.

If you do not specify any *suboptions*, the default is to compute two-sided Wald confidence intervals with confidence level $1 - \alpha = 0.95$.

CLASSICAL

requests that the p-value of the likelihood ratio test be computed by the classical method. If $\widehat{\lambda}$ is the realized value of the test statistic in the likelihood ratio test,

$$p = \Pr\left(\chi^2_\nu \ge \widehat{\lambda}\right)$$

where ν is the degrees of freedom of the hypothesis.

DF=*value-list*

enables you to supply degrees of freedom $\nu_1, \ldots, \nu_k$ for the computation of p-values from chi-square mixtures. The mixture weights $w_1, \ldots, w_k$ are supplied with the WGHT= option. If no weights are specified, an equal weight distribution is assumed. If $\widehat{\lambda}$ is the realized value of the test statistic in the likelihood ratio test, PROC GLIMMIX computes the p-value as (Shapiro 1988)

$$p = \sum_{i=1}^{k} w_i \Pr\left(\chi^2_{\nu_i} \ge \widehat{\lambda}\right)$$

Note that $\chi^2_0 \equiv 0$ and that mixture weights are scaled to sum to one. If you specify more weights than degrees of freedom in *value-list*, the rank of the hypothesis (DF column) is substituted for the missing degrees of freedom.

Specifying a single value ν for *value-list* without giving mixture weights is equivalent to computing the p-value as

$$p = \Pr\left(\chi^2_\nu \geq \widehat{\lambda}\right)$$

For example, the following statements compute the p-value based on a chi-square distribution with one degree of freedom:

```
proc glimmix noprofile;
   class A sub;
   model score = A;
   random _residual_ / type=ar(1) subject=sub;
   covtest 'ELR low' 30.62555 0.7133361 / df=1;
run;
```

The DF column of the COVTEST output will continue to read 2 regardless of the DF= specification, however, because the DF column reflects the rank of the hypothesis and equals the number of constraints imposed on the full model.

ESTIMATES

EST

displays the estimates of the covariance parameters under the null hypothesis. Specifying the ESTIMATES option in one COVTEST statement has the same effect as specifying the option in every COVTEST statement.

MAXITER=*number*

limits the number of iterations when you are refitting the model under the null hypothesis to *number* iterations. If the null model does not converge before the limit is reached, no p-values are produced.

PARMS

displays the values of the covariance parameters under the null hypothesis. This option is useful if you supply multiple sets of parameter values with the TESTDATA= option. Specifying the PARMS option in one COVTEST statement has the same effect as specifying the option in every COVTEST statement.

RESTART

specifies that starting values for the covariance parameters for the null model are obtained by the same mechanism as starting values for the full models. For example, if you do not specify a PARMS statement, the RESTART option computes MIVQUE(0) estimates under the null model (Goodnight 1978a). If you provide starting values with the PARMS statement, the starting values for the null model are obtained by applying restrictions to the starting values for the full model.

By default, PROC GLIMMIX obtains starting values by applying null model restrictions to the converged estimates of the full model. Although this is computationally expedient, the method does not always lead to good starting values for the null model, depending on the nature of the model and hypothesis. In particular, when you receive a warning about parameters not specified under H_0 falling on the boundary, the RESTART option can be useful.

TOLERANCE=*r*

Values within tolerance $r \geq 0$ of the boundary of the parameter space are considered on the boundary when PROC GLIMMIX examines estimates of nuisance parameters under H_0 and determines whether mixture weights and degrees of freedom can be obtained. In certain cases, when parameters not specified under the null hypothesis are on boundaries, the asymptotic distribution of the likelihood ratio statistic is not a mixture of chi-squares (see, for example, case 8 in Self and Liang 1987). The default for *r* is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

WALD

produces Wald Z tests for the covariance parameters based on the estimates and asymptotic standard errors in the "Covariance Parameter Estimates" table.

WGHT=*value-list*

enables you to supply weights for the computation of p-values from chi-square mixtures. See the DF= option for details. Mixture weights are scaled to sum to one.

EFFECT Statement

EFFECT *effect-specification* **;**

The EFFECT statement enables you to construct special collections of columns for **X** or **Z** matrices in your model. These collections are referred to as *constructed effects* to distinguish them from the usual model effects formed from continuous or classification variables.

For details about the syntax of the EFFECT statement and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics." For specific details concerning the use of the EFFECT statement with the GLIMMIX procedure, see the section "Notes on the EFFECT Statement" on page 3971.

ESTIMATE Statement

ESTIMATE '*label*' *contrast-specification* < **(***divisor***=***n***)** >
< **,** '*label*' *contrast-specification* < **(***divisor***=***n***)** > > < **,** . . . >
< / *options* > **;**

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. As in the CONTRAST statement, the basic element of the ESTIMATE statement is the *contrast-specification*, which consists of MODEL and G-side random effects and their coefficients. Specifically, a *contrast-specification* takes the form

< *fixed-effect values* . . . > < | *random-effect values* . . . >

Based on the *contrast-specifications* in your ESTIMATE statement, PROC GLIMMIX constructs the matrix $\mathbf{L}' = [\mathbf{K}' \ \mathbf{M}']$, as in the CONTRAST statement, where **K** is associated with the fixed effects and **M** is associated with the G-side random effects. The GLIMMIX procedure supports nonpositional syntax for the coefficients of fixed effects in the ESTIMATE statement. For details see the section "Positional and Nonpositional Syntax for Contrast Coefficients" on page 3973.

PROC GLIMMIX then produces for each row $\mathbf{l}$ of $\mathbf{L}'$ an approximate t test of the hypothesis $H: \mathbf{l}\boldsymbol{\phi} = 0$, where $\boldsymbol{\phi} = [\boldsymbol{\beta}' \ \boldsymbol{\gamma}']'$. You can also obtain multiplicity-adjusted p-values and confidence limits for multirow estimates with the ADJUST= option. The output from multiple ESTIMATE statements is organized as follows. Results from unadjusted estimates are reported first in a single table, followed by separate tables for each of the adjusted estimates. Results from all ESTIMATE statements are combined in the "Estimates" ODS table.

Note that multirow estimates are permitted. Unlike the CONTRAST statement, you need to specify a '*label*' for every row of the multirow estimate, because PROC GLIMMIX produces one test per row. PROC GLIMMIX selects the degrees of freedom to match those displayed in the "Type III Tests of Fixed Effects" table for the final effect you list in the ESTIMATE statement. You can modify the degrees of freedom by using the DF= option. If you select DDFM=NONE and do not modify the degrees of freedom by using the DF= option, PROC GLIMMIX uses infinite degrees of freedom, essentially computing approximate z tests. If PROC GLIMMIX finds the fixed-effects portion of the specified estimate to be nonestimable, then it displays "Non-est" for the estimate entry.

Table 52.8 summarizes the *options* available in the ESTIMATE statement.

Table 52.8 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
GROUP	Sets up random-effect contrasts between different groups
SINGULAR=	Tunes the estimability checking difference
SUBJECT	Sets up random-effect contrasts between different subjects
Degrees of Freedom and *p*-Values	
ADJDFE=	Determines denominator degrees of freedom when p-values and confidence limits are adjusted for multiple comparisons
ADJUST=	Determines the method for multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1 - \alpha)$
DF=	Assigns a specific value to degrees of freedom for tests and confidence limits
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs t-type confidence limits
E	Prints the $\mathbf{L}$ matrix
Generalized Linear Modeling	
BYCATEGORY	Reports estimates separately for each category for models with nominal data
EXP	Displays exponentiated estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

ADJDFE=SOURCE | ROW

specifies how denominator degrees of freedom are determined when *p*-values and confidence limits are adjusted for multiple comparisons with the ADJUST= option. When you do not specify the ADJDFE= option, or when you specify ADJDFE=SOURCE, the denominator degrees of freedom for multiplicity-adjusted results are the denominator degrees of freedom for the final effect listed in the ESTIMATE statement from the "Type III Tests of Fixed Effects" table.

The ADJDFE=ROW setting is useful if you want multiplicity adjustments to take into account that denominator degrees of freedom are not constant across estimates. This can be the case, for example, when the DDFM=SATTERTHWAITE or DDFM=KENWARDROGER degrees-of-freedom method is in effect.

ADJUST=BON | SCHEFFE | SIDAK | SIMULATE< (*simoptions*) > | T

requests a multiple comparison adjustment for the *p*-values and confidence limits for the estimates. The adjusted quantities are produced in addition to the unadjusted quantities. Adjusted confidence limits are produced if the CL or ALPHA= option is in effect. For a description of the adjustments, see Chapter 53, "The GLM Procedure," Chapter 86, "The MULTTEST Procedure," and the documentation for the ADJUST= option in the LSMEANS statement. The ADJUST= option is ignored for generalized logit models.

If the STEPDOWN option is in effect, the *p*-values are further adjusted in a step-down fashion.

ALPHA=*number*

requests that a *t*-type confidence interval be constructed with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05. If DDFM=NONE and you do not specify degrees of freedom with the DF= option, PROC GLIMMIX uses infinite degrees of freedom, essentially computing a *z* interval.

BYCATEGORY

BYCAT

requests that in models for nominal data (generalized logit models) estimates be reported separately for each category. In contrast to the BYCATEGORY option in the CONTRAST statement, an ESTIMATE statement in a generalized logit model does not distribute coefficients by response category, because ESTIMATE statements always correspond to single rows of the **L** matrix.

For example, assume that the response variable Style is multinomial with three (unordered) categories. The following GLIMMIX statements fit a generalized logit model relating the preferred style of instruction to school and educational program effects:

```
proc glimmix data=school;
   class School Program;
   model Style(order=data) = School Program / s ddfm=none
                             dist=multinomial link=glogit;
   freq Count;
   estimate 'School 1 vs. 2' school 1 -1 / bycat;
   estimate 'School 1 vs. 2' school 1 -1;
run;
```

The first ESTIMATE statement compares school effects separately for each nonredundant category. The second ESTIMATE statement compares the school effects for the first non-reference category.

The BYCATEGORY option has no effect unless your model is a generalized (mixed) logit model.

CL

requests that *t*-type confidence limits be constructed. If DDFM=NONE and you do not specify degrees of freedom with the DF= option, PROC GLIMMIX uses infinite degrees of freedom, essentially computing a *z* interval. The confidence level is 0.95 by default. These intervals are adjusted for multiplicity when you specify the ADJUST= option.

DF=*number*

specifies the degrees of freedom for the *t* test and confidence limits. The default is the denominator degrees of freedom taken from the "Type III Tests of Fixed Effects" table and corresponds to the final effect you list in the ESTIMATE statement.

DIVISOR=*value-list*

specifies a list of values by which to divide the coefficients so that fractional coefficients can be entered as integer numerators. If you do not specify *value-list*, a default value of 1.0 is assumed. Missing values in the *value-list* are converted to 1.0.

If the number of elements in *value-list* exceeds the number of rows of the estimate, the extra values are ignored. If the number of elements in *value-list* is less than the number of rows of the estimate, the last value in *value-list* is copied forward.

If you specify a row-specific divisor as part of the specification of the estimate row, this value multiplies the corresponding divisor implied by the *value-list*. For example, the following statement divides the coefficients in the first row by 8, and the coefficients in the third and fourth row by 3:

```
estimate 'One vs. two'   A 2 -2  (divisor=2),
         'One vs. three' A 1  0 -1         ,
         'One vs. four'  A 3  0  0 -3      ,
         'One vs. five'  A 1  0  0  0  -1  / divisor=4,.,3;
```

Coefficients in the second row are not altered.

E

requests that the **L** matrix coefficients be displayed.

EXP

requests exponentiation of the estimate. When you model data with the logit, cumulative logit, or generalized logit link functions, and the estimate represents a log odds ratio or log cumulative odds ratio, the EXP option produces an odds ratio. See "Odds and Odds Ratio Estimation" on page 3965 for important details about the computation and interpretation of odds and odds ratio results with the GLIMMIX procedure. If you specify the CL or ALPHA= option, the (adjusted) confidence bounds are also exponentiated.

GROUP *coeffs*

sets up random-effect contrasts between different groups when a GROUP= variable appears in the RANDOM statement. By default, ESTIMATE statement coefficients on random effects are distributed equally across groups. If you enter a multirow estimate, you can also enter multiple rows for the GROUP coefficients. If the number of GROUP coefficients is less than the number of contrasts in the ESTIMATE statement, the GLIMMIX procedure cycles through the GROUP coefficients. For example, the following two statements are equivalent:

```
estimate 'Trt 1 vs 2 @ x=0.4' trt 1 -1  0 | x 0.4,
         'Trt 1 vs 3 @ x=0.4' trt 1  0 -1 | x 0.4,
         'Trt 1 vs 2 @ x=0.5' trt 1 -1  0 | x 0.5,
         'Trt 1 vs 3 @ x=0.5' trt 1  0 -1 | x 0.5 /
                group 1 -1, 1 0 -1, 1 -1, 1 0 -1;

estimate 'Trt 1 vs 2 @ x=0.4' trt 1 -1  0 | x 0.4,
         'Trt 1 vs 3 @ x=0.4' trt 1  0 -1 | x 0.4,
         'Trt 1 vs 2 @ x=0.5' trt 1 -1  0 | x 0.5,
         'Trt 1 vs 3 @ x=0.5' trt 1  0 -1 | x 0.5 /
                group 1 -1, 1 0 -1;
```

ILINK

requests that the estimate and its standard error are also reported on the scale of the mean (the inverse linked scale). PROC GLIMMIX computes the value on the mean scale by applying the inverse link to the estimate. The interpretation of this quantity depends on the *fixed-effect values* and *random-effect values* specified in your ESTIMATE statement and on the link function. In a model for binary data with logit link, for example, the following statements compute

$$\frac{1}{1+\exp\{-(\alpha_1-\alpha_2)\}}$$

where α_1 and α_2 are the fixed-effects solutions associated with the first two levels of the classification effect A:

```
proc glimmix;
   class A;
   model y = A / dist=binary link=logit;
   estimate 'A one vs. two' A 1 -1 / ilink;
run;
```

This quantity is not the difference of the probabilities associated with the two levels,

$$\pi_1-\pi_2=\frac{1}{1+\exp\{-\beta_0-\alpha_1\}}-\frac{1}{1+\exp\{-\beta_0-\alpha_2\}}$$

The standard error of the inversely linked estimate is based on the delta method. If you also specify the CL option, the GLIMMIX procedure computes confidence limits for the estimate on the mean scale. In multinomial models for nominal data, the limits are obtained by the delta method. In other models they are obtained from the inverse link transformation of the confidence limits for the estimate. The ILINK option is specific to an ESTIMATE statement.

LOWER

LOWERTAILED

requests that the p-value for the t test be based only on values less than the test statistic. A two-tailed test is the default. A lower-tailed confidence limit is also produced if you specify the CL or ALPHA= option.

Note that for ADJUST=SCHEFFE the one-sided adjusted confidence intervals and one-sided adjusted p-values are the same as the corresponding two-sided statistics, because this adjustment is based on only the right tail of the F distribution.

SINGULAR=*number*

tunes the estimability checking as documented for the SINGULAR= option in CONTRAST statement.

STEPDOWN< (*step-down-options*) >

requests that multiplicity adjustments for the *p*-values of estimates be further adjusted in a step-down fashion. Step-down methods increase the power of multiple testing procedures by taking advantage of the fact that a *p*-value will never be declared significant unless all smaller *p*-values are also declared significant. Note that the STEPDOWN adjustment combined with ADJUST=BON corresponds to the methods of Holm (1979) and "Method 2" of Shaffer (1986); this is the default. Using step-down-adjusted *p*-values combined with ADJUST=SIMULATE corresponds to the method of Westfall (1997).

If the degrees-of-freedom method is DDFM=KENWARDROGER or DDFM=SATTERTHWAITE, then step-down-adjusted *p*-values are produced only if the ADJDFE=ROW option is in effect.

Also, the STEPDOWN option affects only *p*-values, not confidence limits. For ADJUST=SIMULATE, the generalized least squares hybrid approach of Westfall (1997) is employed to increase Monte Carlo accuracy.

You can specify the following *step-down-options* in parentheses after the STEPDOWN option.

MAXTIME=*n*

specifies the time (in seconds) to spend computing the maximal logically consistent sequential subsets of equality hypotheses for TYPE=LOGICAL. The default is MAXTIME=60. If the MAXTIME value is exceeded, the adjusted tests are not computed. When this occurs, you can try increasing the MAXTIME value. However, note that there are common multiple comparisons problems for which this computation requires a huge amount of time—for example, all pairwise comparisons between more than 10 groups. In such cases, try to use TYPE=FREE (the default) or TYPE=LOGICAL(*n*) for small *n*.

ORDER=PVALUE | ROWS

specifies the order in which the step-down tests are performed. ORDER=PVALUE is the default, with estimates being declared significant only if all estimates with smaller (unadjusted) *p*-values are significant. If you specify ORDER=ROWS, then significances are evaluated in the order in which they are specified in the syntax.

REPORT

specifies that a report on the step-down adjustment be displayed, including a listing of the sequential subsets (Westfall 1997) and, for ADJUST=SIMULATE, the step-down simulation results.

TYPE=LOGICAL< (*n*) > **| FREE**

If you specify TYPE=LOGICAL, the step-down adjustments are computed by using maximal logically consistent sequential subsets of equality hypotheses (Shaffer 1986; Westfall 1997). Alternatively, for TYPE=FREE, sequential subsets are computed ignoring logical constraints. The TYPE=FREE results are more conservative than those for TYPE=LOGICAL, but they can be much more efficient to produce for many estimates. For example, it is not feasible to take logical constraints between all pairwise comparisons of more than about 10 groups. For this reason, TYPE=FREE is the default.

However, you can reduce the computational complexity of taking logical constraints into account by limiting the depth of the search tree used to compute them, specifying the optional depth

parameter as a number n in parentheses after TYPE=LOGICAL. As with TYPE=FREE, results for TYPE=LOGICAL(n) are conservative relative to the true TYPE=LOGICAL results, but even for TYPE=LOGICAL(0) they can be appreciably less conservative than TYPE=FREE and they are computationally feasible for much larger numbers of estimates. If you do not specify n or if $n = -1$, the full search tree is used.

SUBJECT *coeffs*

sets up random-effect contrasts between different subjects when a SUBJECT= variable appears in the RANDOM statement. By default, ESTIMATE statement coefficients on random effects are distributed equally across subjects. Listing subject coefficients for an ESTIMATE statement with multiple rows follows the same rules as for GROUP coefficients.

UPPER

UPPERTAILED

requests that the p-value for the t test be based only on values greater than the test statistic. A two-tailed test is the default. An upper-tailed confidence limit is also produced if you specify the CL or ALPHA= option.

Note that for ADJUST=SCHEFFE the one-sided adjusted confidence intervals and one-sided adjusted p-values are the same as the corresponding two-sided statistics, because this adjustment is based on only the right tail of the F distribution.

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set or one computed through PROC GLIMMIX programming statements that contains the frequency of occurrence for each observation. PROC GLIMMIX treats each observation as if it appears f times, where f is the value of the FREQ variable for the observation. If it is not an integer, the frequency value is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

The analysis produced by using a FREQ statement reflects the expanded number of observations. For an example of a FREQ statement in a model with random effects, see Example 52.11 in this chapter.

ID Statement

ID *variables* **;**

The ID statement specifies which quantities to include in the OUT= data set from the OUTPUT statement in addition to any statistics requested in the OUTPUT statement. If no ID statement is given, the GLIMMIX procedure includes all variables from the input data set in the OUT= data set. Otherwise, only the variables listed in the ID statement are included. Automatic variables such as _LINP_, _MU_, _VARIANCE_, etc. are not transferred to the OUT= data set unless they are listed in the ID statement.

The ID statement can be used to transfer computed quantities that depend on the model to an output data set. In the following example, two sets of Hessian weights are computed in a gamma regression with a

noncanonical link. The covariance matrix for the fixed effects can be constructed as the inverse of $\mathbf{X}'\mathbf{W}\mathbf{X}$. $\mathbf{W}$ is a diagonal matrix of the w_{ei} or w_{oi}, depending on whether the expected or observed Hessian matrix is desired, respectively.

```
proc glimmix;
   class group age;
   model cost = group age / s error=gamma link=pow(0.5);
   output out=gmxout pred=pred;
   id _variance_ wei woi;
   vpmu  = 2*_mu_;
   if (_mu_ > 1.0e-8) then do;
      gpmu  =  0.5  * (_mu_**(-0.5));
      gppmu = -0.25 * (_mu_**(-1.5));
      wei   = 1/(_phi_*_variance_*gpmu*gpmu);
      woi   = wei + (cost-_mu_) *
              (_variance_*gppmu + vpmu*gpmu) /
              (_variance_*_variance_*gpmu*gpmu*gpmu*_phi_);
   end;
run;
```

The variables _VARIANCE_ and _MU_ and other symbols are predefined by PROC GLIMMIX and can be used in programming statements. For rules and restrictions, see the section "Programming Statements" on page 3911.

LSMEANS Statement

LSMEANS *fixed-effects* < / *options* > **;**

The LSMEANS statement computes least squares means (LS-means) of fixed effects. As in the GLM and the MIXED procedures, LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs. The $\mathbf{L}$ matrix constructed to compute them is the same as the $\mathbf{L}$ matrix formed in PROC GLM; however, the standard errors are adjusted for the covariance parameters in the model. Least squares means computations are not supported for multinomial models.

Each LS-mean is computed as $\mathbf{L}\widehat{\boldsymbol{\beta}}$, where $\mathbf{L}$ is the coefficient matrix associated with the least squares mean and $\widehat{\boldsymbol{\beta}}$ is the estimate of the fixed-effects parameter vector. The approximate standard error for the LS-mean is computed as the square root of $\mathbf{L}\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}]\mathbf{L}'$. The approximate variance matrix of the fixed-effects estimates depends on the estimation method.

LS-means are constructed on the linked scale—that is, the scale on which the model effects are additive. For example, in a binomial model with logit link, the least squares means are predicted population margins of the logits.

LS-means can be computed for any effect in the MODEL statement that involves only CLASS variables. You can specify multiple effects in one LSMEANS statement or in multiple LSMEANS statements, and all LSMEANS statements must appear after the MODEL statement. As in the ESTIMATE statement, the $\mathbf{L}$ matrix is tested for estimability, and if this test fails, PROC GLIMMIX displays "Non-est" for the LS-means entries.

Assuming the LS-mean is estimable, PROC GLIMMIX constructs an approximate t test to test the null hypothesis that the associated population quantity equals zero. By default, the denominator degrees of

freedom for this test are the same as those displayed for the effect in the "Type III Tests of Fixed Effects" table. If the DDFM=SATTERTHWAITE or DDFM=KENWARDROGER option is specified in the MODEL statement, PROC GLIMMIX determines degrees of freedom separately for each test, unless the DDF= option overrides it for a particular effect. See the DDFM= option for more information. Table 52.9 summarizes options available in the LSMEANS statement. All LSMEANS *options* are subsequently discussed in alphabetical order.

Table 52.9 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Requests differences of LS-means
OM	Specifies weighting scheme for LS-mean computation as determined by the input data set
SINGULAR=	Tunes estimability checking
SLICE=	Partitions F tests (simple effects)
SLICEDIFF=	Requests simple effects differences
SLICEDIFFTYPE	Determines the type of simple difference
Degrees of Freedom and *p*-Values	
ADJDFE=	Determines whether to compute row-wise denominator degrees of freedom with DDFM=SATTERTHWAITE or DDFM=KENWARDROGER
ADJUST=	Determines the method for multiple comparison adjustment of LS-mean differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
DF=	Assigns specific value to degrees of freedom for tests and confidence limits
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and or mean differences
CORR	Displays correlation matrix of LS-means
COV	Displays covariance matrix of LS-means
E	Prints the **L** matrix
ILINK	Applies the inverse link transform to the LS-Means (not differences) and produces the standard errors on the inverse linked scale
LINES	Produces "Lines" display for pairwise LS-mean differences
ODDS	Reports odds of levels of fixed effects if permissible by the link function
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permissible by the link function
PLOTS=	Requests ODS statistical graphics of means and mean comparisons

You can specify the following *options* in the LSMEANS statement after a slash (/).

ADJDFE=ROW | SOURCE

specifies how denominator degrees of freedom are determined when *p*-values and confidence limits are adjusted for multiple comparisons with the ADJUST= option. When you do not specify the ADJDFE= option, or when you specify ADJDFE=SOURCE, the denominator degrees of freedom for multiplicity-adjusted results are the denominator degrees of freedom for the LS-mean effect in the "Type III Tests of Fixed Effects" table. When you specify ADJDFE=ROW, the denominator degrees of freedom for multiplicity-adjusted results correspond to the degrees of freedom displayed in the DF column of the "Differences of Least Squares Means" table.

The ADJDFE=ROW setting is particularly useful if you want multiplicity adjustments to take into account that denominator degrees of freedom are not constant across LS-mean differences. This can be the case, for example, when the DDFM=SATTERTHWAITE or DDFM=KENWARDROGER degrees-of-freedom method is in effect.

In one-way models with heterogeneous variance, combining certain ADJUST= options with the ADJDFE=ROW option corresponds to particular methods of performing multiplicity adjustments in the presence of heteroscedasticity. For example, the following statements fit a heteroscedastic one-way model and perform Dunnett's T3 method (Dunnett 1980), which is based on the studentized maximum modulus (ADJUST=SMM):

```
proc glimmix;
   class A;
   model y = A / ddfm=satterth;
   random _residual_ / group=A;
   lsmeans A / adjust=smm adjdfe=row;
run;
```

If you combine the ADJDFE=ROW option with ADJUST=SIDAK, the multiplicity adjustment corresponds to the T2 method of Tamhane (1979), while ADJUST=TUKEY corresponds to the method of Games-Howell (Games and Howell 1976). Note that ADJUST=TUKEY gives the exact results for the case of fractional degrees of freedom in the one-way model, but it does not take into account that the degrees of freedom are subject to variability. A more conservative method, such as ADJUST=SMM, might protect the overall error rate better.

Unless the ADJUST= option is specified in the LSMEANS statement, the ADJDFE= option has no effect.

ADJUST=BON

ADJUST=DUNNETT

ADJUST=NELSON

ADJUST=SCHEFFE

ADJUST=SIDAK

ADJUST=SIMULATE<(*simoptions*)>

ADJUST=SMM | GT2

ADJUST=TUKEY

requests a multiple comparison adjustment for the *p*-values and confidence limits for the differences of LS-means. The adjusted quantities are produced in addition to the unadjusted quantities. By default, PROC GLIMMIX performs all pairwise differences. If you specify ADJUST=DUNNETT, the

procedure analyzes all differences with a control level. If you specify ADJUST=NELSON, ANOM differences are taken. The ADJUST= option implies the DIFF option, unless the SLICEDIFF= option is specified.

The BON (Bonferroni) and SIDAK adjustments involve correction factors described in Chapter 53, "The GLM Procedure," and Chapter 86, "The MULTTEST Procedure"; also see Westfall and Young (1993) and Westfall et al. (1999). When you specify ADJUST=TUKEY and your data are unbalanced, PROC GLIMMIX uses the approximation described in Kramer (1956) and identifies the adjustment as "Tukey-Kramer" in the results. Similarly, when you specify ADJUST=DUNNETT or ADJUST=NELSON and the LS-means are correlated, the GLIMMIX procedure uses the factor-analytic covariance approximation described in Hsu (1992) and identifies the adjustment in the results as "Dunnett-Hsu" or "Nelson-Hsu," respectively. The approximation derives an approximate "effective sample sizes" for which exact critical values are computed. Note that computing the exact adjusted p-values and critical values for unbalanced designs can be computationally intensive, in particular for ADJUST=NELSON. A simulation-based approach, as specified by the ADJUST=SIM option, while nondeterministic, can provide inferences that are sufficiently accurate in much less time. The preceding references also describe the SCHEFFE and SMM adjustments.

Nelson's adjustment applies only to the analysis of means (Ott 1967; Nelson 1982, 1991, 1993), where LS-means are compared against an average LS-mean. It does not apply to all pairwise differences of least squares means, or to slice differences that you specify with the SLICEDIFF= option. See the DIFF=ANOM option for more details regarding the analysis of means with the GLIMMIX procedure.

The SIMULATE adjustment computes adjusted p-values and confidence limits from the simulated distribution of the maximum or maximum absolute value of a multivariate t random vector. All covariance parameters, except the residual scale parameter, are fixed at their estimated values throughout the simulation, potentially resulting in some underdispersion. The simulation estimates q, the true $(1-\alpha)$ quantile, where $1-\alpha$ is the confidence coefficient. The default α is 0.05, and you can change this value with the ALPHA= option in the LSMEANS statement.

The number of samples is set so that the tail area for the simulated q is within γ of $1-\alpha$ with $100(1-\epsilon)\%$ confidence. In equation form,

$$\Pr(|F(\widehat{q}) - (1-\alpha)| \leq \gamma) = 1 - \epsilon$$

where $\hat{q}$ is the simulated q and F is the true distribution function of the maximum; see Edwards and Berry (1987) for details. By default, $\gamma = 0.005$ and $\epsilon = 0.01$, placing the tail area of $\hat{q}$ within 0.005 of 0.95 with 99% confidence. The ACC= and EPS= *simoptions* reset γ and ϵ, respectively, the NSAMP= *simoption* sets the sample size directly, and the SEED= *simoption* specifies an integer used to start the pseudo-random number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. For additional descriptions of these and other simulation options, see the section "LSMEANS Statement" on page 4172 in Chapter 53, "The GLM Procedure."

If the STEPDOWN option is in effect, the p-values are further adjusted in a step-down fashion. For certain options and data, this adjustment is exact under an iid $N(0, \sigma^2)$ model for the dependent variable, in particular for the following:

- for ADJUST=DUNNETT when the means are uncorrelated
- for ADJUST=TUKEY with STEPDOWN(TYPE=LOGICAL) when the means are balanced and uncorrelated.

The first case is a consequence of the nature of the successive step-down hypotheses for comparisons with a control; the second employs an extension of the maximum studentized range distribution appropriate for partition hypotheses (Royen 1989). Finally, for STEPDOWN(TYPE=FREE), ADJUST=TUKEY employs the Royen (1989) extension in such a way that the resulting p-values are conservative.

ALPHA=*number*

requests that a t-type confidence interval be constructed for each of the LS-means with confidence level $1 - \mathit{number}$. The value of *number* must be between 0 and 1; the default is 0.05.

ASC

ASCENDING

sorts LS-means in ascending order in results that you display by specifying the LINES option.

AT *variable*=*value*

AT (*variable-list***)=(***value-list***)**

AT MEANS

enables you to modify the values of the covariates used in computing LS-means. By default, all covariate effects are set equal to their mean values for computation of standard LS-means. The AT option enables you to assign arbitrary values to the covariates. Additional columns in the output table indicate the values of the covariates.

If there is an effect containing two or more covariates, the AT option sets the effect equal to the product of the individual means rather than the mean of the product (as with standard LS-means calculations). The AT MEANS option sets covariates equal to their mean values (as with standard LS-means) and incorporates this adjustment to crossproducts of covariates.

As an example, consider the following invocation of PROC GLIMMIX:

```
proc glimmix;
   class A;
   model Y = A x1 x2 x1*x2;
   lsmeans A;
   lsmeans A / at means;
   lsmeans A / at x1=1.2;
   lsmeans A / at (x1 x2)=(1.2 0.3);
run;
```

For the first two LSMEANS statements, the LS-means coefficient for x1 is $\overline{x}_1$ (the mean of x1) and for x2 is $\overline{x}_2$ (the mean of x2). However, for the first LSMEANS statement, the coefficient for x1*x2 is $\overline{x_1 x_2}$, but for the second LSMEANS statement, the coefficient is $\overline{x}_1 \times \overline{x}_2$. The third LSMEANS statement sets the coefficient for x1 equal to 1.2 and leaves it at $\overline{x}_2$ for x2, and the final LSMEANS statement sets these values to 1.2 and 0.3, respectively.

Even if you specify a WEIGHT variable, the unweighted covariate means are used for the covariate coefficients if there is no AT specification. If you specify the AT option, WEIGHT or FREQ variables are taken into account as follows. The weighted covariate means are then used for the covariate coefficients for which no explicit AT values are given, or if you specify AT MEANS. Observations that do not contribute to the analysis because of a missing dependent variable are included in computing

the covariate means. You should use the E option in conjunction with the AT option to check that the modified LS-means coefficients are the ones you want.

The AT option is disabled if you specify the BYLEVEL option.

BYLEVEL

requests that separate margins be computed for each level of the LSMEANS effect.

The standard LS-means have equal coefficients across classification effects. The BYLEVEL option changes these coefficients to be proportional to the observed margins. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in the input data set. In this case, the resulting LS-means are actually equal to raw means for fixed-effects models and certain balanced random-effects models, but their estimated standard errors account for the covariance structure that you have specified. If a WEIGHT statement is specified, PROC GLIMMIX uses weighted margins to construct the LS-means coefficients.

If the AT option is specified, the BYLEVEL option disables it.

CL

requests that t-type confidence limits be constructed for each of the LS-means. If DDFM=NONE, then PROC GLIMMIX uses infinite degrees of freedom for this test, essentially computing a z interval. The confidence level is 0.95 by default; this can be changed with the ALPHA= option. If you specify an ADJUST= option, then the confidence limits are adjusted for multiplicity, but if you also specify STEPDOWN, then only p-values are step-down adjusted, not the confidence limits.

CORR

displays the estimated correlation matrix of the least squares means as part of the "Least Squares Means" table.

COV

displays the estimated covariance matrix of the least squares means as part of the "Least Squares Means" table.

DF=*number*

specifies the degrees of freedom for the t test and confidence limits. The default is the denominator degrees of freedom taken from the "Type III Tests of Fixed Effects" table corresponding to the LS-means effect.

DIFF<=*difftype***>**

PDIFF<=*difftype***>**

requests that differences of the LS-means be displayed. The optional *difftype* specifies which differences to produce, with possible values ALL, ANOM, CONTROL, CONTROLL, and CONTROLU. The ALL value requests all pairwise differences, and it is the default. The CONTROL *difftype* requests differences with a control, which, by default, is the first level of each of the specified LSMEANS effects.

The ANOM value requests differences between each LS-mean and the average LS-mean, as in the *analysis of means* (Ott 1967). The average is computed as a weighted mean of the LS-means, the weights being inversely proportional to the diagonal entries of the

$$\mathbf{L}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{L}'$$

matrix. If LS-means are nonestimable, this design-based weighted mean is replaced with an equally weighted mean. Note that the ANOM procedure in SAS/QC software implements both tables and graphics for the analysis of means with a variety of response types. For one-way designs and normal data with identity link, the DIFF=ANOM computations are equivalent to the results of PROC ANOM. If the LS-means being compared are uncorrelated, exact adjusted *p*-values and critical values for confidence limits can be computed in the analysis of means; see Nelson (1982, 1991, 1993); Guirguis and Tobias (2004) as well as the documentation for the ADJUST=NELSON option.

To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the CONTROL keyword. For example, if the effects A, B, and C are classification variables, each having two levels, 1 and 2, the following LSMEANS statement specifies the (1,2) level of A*B and the (2,1) level of B*C as controls:

```
lsmeans A*B B*C / diff=control('1' '2', '2' '1');
```

For multiple effects, the results depend upon the order of the list, and so you should check the output to make sure that the controls are correct.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*. The CONTROLL *difftype* tests whether the noncontrol levels are significantly smaller than the control; the upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing. Conversely, the CONTROLU *difftype* tests whether the noncontrol levels are significantly larger than the control; the upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

If you want to perform multiple comparison adjustments on the differences of LS-means, you must specify the ADJUST= option.

The differences of the LS-means are displayed in a table titled "Differences of Least Squares Means."

E

requests that the **L** matrix coefficients for the LSMEANS effects be displayed.

ILINK

requests that estimates and their standard errors in the "Least Squares Means" table also be reported on the scale of the mean (the inverse linked scale). The ILINK option is specific to an LSMEANS statement. If you also specify the CL option, the GLIMMIX procedure computes confidence intervals for the predicted means by applying the inverse link transform to the confidence limits on the linked (linear) scale. Standard errors on the inverse linked scale are computed by the delta method.

The GLIMMIX procedure applies the inverse link transform to the LS-mean reported in the Estimate column. In a logistic model, for example, this implies that the value reported as the inversely linked estimate corresponds to a predicted probability that is based on an average estimable function (the estimable function that produces the LS-mean on the linear scale). To compute average predicted probabilities, you can average the results from applying the ILINK option in the ESTIMATE statement for suitably chosen estimable functions.

LINES

presents results of comparisons between all pairs of least squares means by listing the means in descending order and indicating nonsignificant subsets by line segments beside the corresponding LS-means. When all differences have the same variance, these comparison lines are guaranteed to accurately reflect the inferences based on the corresponding tests, made by comparing the respective *p*-values to the value of the ALPHA= option (0.05 by default). However, equal variances might not be the case for differences between LS-means. If the variances are not all the same, then the comparison lines might be conservative, in the sense that if you base your inferences on the lines alone, you will detect fewer significant differences than the tests indicate. If there are any such differences, PROC GLIMMIX lists the pairs of means that are inferred to be significantly different by the tests but not by the comparison lines. Note, however, that in many cases, even though the variances are unequal, they are similar enough that the comparison lines accurately reflect the test inferences.

ODDS

requests that in models with logit, cumulative logit, and generalized logit link function the odds of the levels of the fixed effects are reported. If you specify the CL or ALPHA= option, confidence intervals for the odds are also computed. See the section "Odds and Odds Ratio Estimation" on page 3965 for further details about computation and interpretation of odds and odds ratios with the GLIMMIX procedure.

ODDSRATIO

OR

requests that LS-mean differences (DIFF, ADJUST= options) and simple effect comparisons (SLICEDIFF option) are also reported in terms of odds ratios. The ODDSRATIO option is ignored unless you use either the logit, cumulative logit, or generalized logit link function. If you specify the CL or ALPHA= option, confidence intervals for the odds ratios are also computed. These intervals are adjusted for multiplicity when you specify the ADJUST= option. See the section "Odds and Odds Ratio Estimation" on page 3965 for further details about computation and interpretation of odds and odds ratios with the GLIMMIX procedure.

OBSMARGINS

OM

specifies a potentially different weighting scheme for the computation of LS-means coefficients. The standard LS-means have equal coefficients across classification effects; however, the OM option changes these coefficients to be proportional to those found in the input data set. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in your data.

In computing the observed margins, PROC GLIMMIX uses all observations for which there are no missing or invalid independent variables, including those for which there are missing dependent variables. Also, if you use a WEIGHT statement, PROC GLIMMIX computes weighted margins to construct the LS-means coefficients. If your data are balanced, the LS-means are unchanged by the OM option.

The BYLEVEL option modifies the observed-margins LS-means. Instead of computing the margins across all of the input data set, PROC GLIMMIX computes separate margins for each level of the LSMEANS effect in question. In this case the resulting LS-means are actually equal to raw means for fixed-effects models and certain balanced random-effects models, but their estimated standard errors account for the covariance structure that you have specified.

You can use the E option in conjunction with either the OM or BYLEVEL option to check that the modified LS-means coefficients are the ones you want. It is possible that the modified LS-means are not estimable when the standard ones are estimable, or vice versa.

PDIFF

is the same as the DIFF option. See the description of the DIFF option on page 3844.

PLOT | PLOTS< =*plot-request*< (*options*) > >

PLOT | PLOTS< =(*plot-request*< (*options*) > < . . . *plot-request*< (*options*) > >) >

creates least squares means related graphs when ODS Graphics has been enabled and the plot request does not conflict with other *options* in the LSMEANS statement. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For examples of the basic statistical graphics for least squares means and aspects of their computation and interpretation, see the section "Graphics for LS-Mean Comparisons" on page 3997 in this chapter.

The *options* for a specific plot request (and their *suboptions*) of the LSMEANS statement include those for the PLOTS= option in the PROC GLIMMIX statement. You can specify classification effects in the MEANPLOT request of the LSMEANS statement to control the display of interaction means with the PLOTBY= and SLICEBY= suboptions; these are not available in the PLOTS= option in the PROC GLIMMIX statement. Options specified in the LSMEANS statement override those in the PLOTS= option in the PROC GLIMMIX statement.

The available *options* and *suboptions* are as follows.

ALL

requests that the default plots corresponding to this LSMEANS statement be produced. The default plot depends on the options in the statement.

ANOMPLOT

ANOM

requests an analysis of means display in which least squares means are compared to an average least squares mean. Least squares mean ANOM plots are produced only for those model effects listed in LSMEANS statements that have options that do not contradict with the display. For example, the following statements produce analysis of mean plots for effects A and C:

```
lsmeans A / diff=anom plot=anom;
lsmeans B / diff      plot=anom;
lsmeans C /           plot=anom;
```

The DIFF option in the second LSMEANS statement implies all pairwise differences.

CONTROLPLOT

CONTROL

requests a display in which least squares means are visually compared against a reference level. These plots are produced only for statements with options that are compatible with control differences. For example, the following statements produce control plots for effects A and C:

```
lsmeans A / diff=control('1') plot=control;
lsmeans B / diff              plot=control;
lsmeans C                     plot=control;
```

The DIFF option in the second LSMEANS statement implies all pairwise differences.

DIFFPLOT< (*diffplot-options*) >

DIFFOGRAM< (*diffplot-options*) >

DIFF< (*diffplot-options*) >

requests a display of all pairwise least squares mean differences and their significance. The display is also known as a "mean-mean scatter plot" when it is based on arithmetic means (Hsu 1996; Hsu and Peruggia 1994). For each comparison a line segment, centered at the LS-means in the pair, is drawn. The length of the segment corresponds to the projected width of a confidence interval for the least squares mean difference. Segments that fail to cross the 45-degree reference line correspond to significant least squares mean differences.

LS-mean difference plots are produced only for statements with options that are compatible with the display. For example, the following statements request differences against a control level for the A effect, all pairwise differences for the B effect, and the least squares means for the C effect:

```
lsmeans A / diff=control('1') plot=diff;
lsmeans B / diff              plot=diff;
lsmeans C                     plot=diff;
```

The DIFF= type in the first statement is incompatible with a display of all pairwise differences.

You can specify the following *diffplot-options*. The ABS and NOABS options determine the positioning of the line segments in the plot. When the ABS option is in effect, and this is the default, all line segments are shown on the same side of the reference line. The NOABS option separates comparisons according to the sign of the difference. The CENTER option marks the center point for each comparison. This point corresponds to the intersection of two least squares means. The NOLINES option suppresses the display of the line segments that represent the confidence bounds for the differences of the least squares means. The NOLINES option implies the CENTER option. The default is to draw line segments in the upper portion of the plot area without marking the center point.

MEANPLOT< (*meanplot-options*) >

requests displays of the least squares means.

The following *meanplot-options* control the display of the least squares means.

ASCENDING

displays the least squares means in ascending order. This option has no effect if means are sliced or displayed in separate plots.

CL

displays upper and lower confidence limits for the least squares means. By default, 95% limits are drawn. You can change the confidence level with the ALPHA= option. Confidence limits are drawn by default if the CL option is specified in the LSMEANS statement.

CLBAND

displays confidence limits as bands. This option implies the JOIN option.

DESCENDING
displays the least squares means in descending order. This option has no effect if means are sliced or displayed in separate plots.

ILINK
requests that means (and confidence limits) are displayed on the inverse linked scale.

JOIN
CONNECT
connects the least squares means with lines. This option is implied by the CLBAND option. If the effect contains nested variables, and a SLICEBY= effect contains classification variables that appear as crossed effects, this option is ignored.

SLICEBY=*fixed-effect*
specifies an effect by which to group the means in a single plot. For example, the following statement requests a plot in which the levels of A are placed on the horizontal axis and the means that belong to the same level of B are joined by lines:

```
lsmeans A*B / plot=meanplot(sliceby=b join);
```

Unless the LS-mean effect contains at least two classification variables, the SLICEBY= option has no effect. The *fixed-effect* does not have to be an effect in your MODEL statement, but it must consist entirely of classification variables.

PLOTBY=*fixed-effect*
specifies an effect by which to break interaction plots into separate displays. For example, the following statement requests for each level of C one plot of the A*B cell means that are associated with that level of C:

```
lsmeans A*B*C / plot=meanplot(sliceby=b plotby=c clband);
```

In each plot, levels of A are displayed on the horizontal axis, and confidence bands are drawn around the means that share the same level of B.

The PLOTBY= option has no effect unless the LS-mean effect contains at least three classification variables. The *fixed-effect* does not have to be an effect in the MODEL statement, but it must consist entirely of classification variables.

NONE
requests that no plots be produced.

When LS-mean calculations are adjusted for multiplicity by using the ADJUST= option, the plots are adjusted accordingly.

SINGULAR=*number*
tunes the estimability checking as documented for the CONTRAST statement.

SLICE=*fixed-effect* | (*fixed-effects*)

specifies effects by which to partition interaction LSMEANS effects. This can produce what are known as tests of simple effects (Winer 1971). For example, suppose that A*B is significant, and you want to test the effect of A for each level of B. The appropriate LSMEANS statement is

```
lsmeans A*B / slice=B;
```

This statement tests for the simple main effects of A for B, which are calculated by extracting the appropriate rows from the coefficient matrix for the A*B LS-means and by using them to form an F test.

The SLICE option produces F tests that test the simultaneous equality of cell means at a fixed level of the slice effect (Schabenberger, Gregoire, and Kong 2000). You can request differences of the least squares means while holding one or more factors at a fixed level with the SLICEDIFF= option.

The SLICE option produces a table titled "Tests of Effect Slices."

SLICEDIFF=*fixed-effect* | (*fixed-effects*)

SIMPLEDIFF=*fixed-effect* | (*fixed-effects*)

requests that differences of simple effects be constructed and tested against zero. Whereas the SLICE option extracts multiple rows of the coefficient matrix and forms an F test, the SLICEDIFF option tests pairwise differences of these rows. This enables you to perform multiple comparisons among the levels of one factor at a fixed level of the other factor. For example, assume that, in a balanced design, factors A and B have $a = 4$ and $b = 3$ levels, respectively. Consider the following statements:

```
proc glimmix;
   class a b;
   model y = a b a*b;
   lsmeans a*b / slice=a;
   lsmeans a*b / slicediff=a;
run;
```

The first LSMEANS statement produces four F tests, one per level of A. The first of these tests is constructed by extracting the three rows corresponding to the first level of A from the coefficient matrix for the A*B interaction. Call this matrix $\mathbf{L}_{a1}$ and its rows $\mathbf{l}_{a1}^{(1)}$, $\mathbf{l}_{a1}^{(2)}$, and $\mathbf{l}_{a1}^{(3)}$. The SLICE tests the two-degrees-of-freedom hypothesis

$$H: \begin{cases} \left(\mathbf{l}_{a1}^{(1)} - \mathbf{l}_{a1}^{(2)}\right)\boldsymbol{\beta} = 0 \\ \left(\mathbf{l}_{a1}^{(1)} - \mathbf{l}_{a1}^{(3)}\right)\boldsymbol{\beta} = 0 \end{cases}$$

In a balanced design, where μ_{ij} denotes the mean response if A is at level i and B is at level j, this hypothesis is equivalent to $H: \mu_{11} = \mu_{12} = \mu_{13}$. The SLICEDIFF option considers the three rows of $\mathbf{L}_{a1}$ in turn and performs tests of the difference between pairs of rows. How these differences are constructed depends on the SLICEDIFFTYPE= option. By default, all pairwise differences within the subset of $\mathbf{L}$ are considered; in the example this corresponds to tests of the form

$$H: \left(\mathbf{l}_{a1}^{(1)} - \mathbf{l}_{a1}^{(2)}\right)\boldsymbol{\beta} = 0$$

$$H: \left(\mathbf{l}_{a1}^{(1)} - \mathbf{l}_{a1}^{(3)}\right)\boldsymbol{\beta} = 0$$

$$H: \left(\mathbf{l}_{a1}^{(2)} - \mathbf{l}_{a1}^{(3)}\right)\boldsymbol{\beta} = 0$$

In the example, with $a = 4$ and $b = 3$, the second LSMEANS statement produces four sets of least squares means differences. Within each set, factor A is held fixed at a particular level and each set consists of three comparisons.

When the ADJUST= option is specified, the GLIMMIX procedure also adjusts the tests for multiplicity. The adjustment is based on the number of comparisons within each level of the SLICEDIFF= effect; see the SLICEDIFFTYPE= option. The Nelson adjustment is not available for slice differences.

SLICEDIFFTYPE< =*difftype*>

SIMPLEDIFFTYPE< =*difftype*>

determines the type of simple effect differences produced with the SLICEDIFF= option.

The possible values for the *difftype* are ALL, CONTROL, CONTROLL, and CONTROLU. The *difftype* ALL requests all simple effects differences, and it is the default. The *difftype* CONTROL requests the differences with a control, which, by default, is the first level of each of the specified LSMEANS effects.

To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the keyword CONTROL. For example, if the effects A, B, and C are classification variables, each having three levels (1, 2, and 3), the following LSMEANS statement specifies the (1,3) level of A*B as the control:

```
lsmeans A*B / slicediff=(A B)
              slicedifftype=control('1' '3');
```

This LSMEANS statement first produces simple effects differences holding the levels of A fixed, and then it produces simple effects differences holding the levels of B fixed. In the former case, level '3' of B serves as the control level. In the latter case, level '1' of A serves as the control.

For multiple effects, the results depend upon the order of the list, and so you should check the output to make sure that the controls are correct.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*. The CONTROLL *difftype* tests whether the noncontrol levels are significantly smaller than the control; the upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing. Conversely, the CONTROLU *difftype* tests whether the noncontrol levels are significantly larger than the control; the upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

STEPDOWN< (*step-down options*)>

requests that multiple comparison adjustments for the p-values of LS-mean differences be further adjusted in a step-down fashion. Step-down methods increase the power of multiple comparisons by taking advantage of the fact that a p-value will never be declared significant unless all smaller p-values are also declared significant. Note that the STEPDOWN adjustment combined with ADJUST=BON corresponds to the methods of Holm (1979) and "Method 2" of Shaffer (1986); this is the default. Using step-down-adjusted p-values combined with ADJUST=SIMULATE corresponds to the method of Westfall (1997).

If the degrees-of-freedom method is DDFM=KENWARDROGER or DDFM=SATTERTHWAITE, then step-down-adjusted p-values are produced only if the ADJDFE=ROW option is in effect.

Also, STEPDOWN affects only p-values, not confidence limits. For ADJUST=SIMULATE, the generalized least squares hybrid approach of Westfall (1997) is employed to increase Monte Carlo accuracy.

You can specify the following *step-down options* in parentheses:

MAXTIME=*n*

specifies the time (in seconds) to spend computing the maximal logically consistent sequential subsets of equality hypotheses for TYPE=LOGICAL. The default is MAXTIME=60. If the MAXTIME value is exceeded, the adjusted tests are not computed. When this occurs, you can try increasing the MAXTIME value. However, note that there are common multiple comparisons problems for which this computation requires a huge amount of time—for example, all pairwise comparisons between more than 10 groups. In such cases, try to use TYPE=FREE (the default) or TYPE=LOGICAL(n) for small n.

REPORT

specifies that a report on the step-down adjustment should be displayed, including a listing of the sequential subsets (Westfall 1997) and, for ADJUST=SIMULATE, the step-down simulation results.

TYPE=LOGICAL<(*n***)> | FREE**

If you specify TYPE=LOGICAL, the step-down adjustments are computed by using maximal logically consistent sequential subsets of equality hypotheses (Shaffer 1986; Westfall 1997). Alternatively, for TYPE=FREE, sequential subsets are computed ignoring logical constraints. The TYPE=FREE results are more conservative than those for TYPE=LOGICAL, but they can be much more efficient to produce for many comparisons. For example, it is not feasible to take logical constraints between all pairwise comparisons of more than 10 groups. For this reason, TYPE=FREE is the default.

However, you can reduce the computational complexity of taking logical constraints into account by limiting the depth of the search tree used to compute them, specifying the optional depth parameter as a number n in parentheses after TYPE=LOGICAL. As with TYPE=FREE, results for TYPE=LOGICAL(n) are conservative relative to the true TYPE=LOGICAL results, but even for TYPE=LOGICAL(0) they can be appreciably less conservative than TYPE=FREE and they are computationally feasible for much larger numbers of comparisons. If you do not specify n or if $n = -1$, the full search tree is used.

LSMESTIMATE Statement

LSMESTIMATE *fixed-effect* <'*label*'> *values* <*divisor*=*n*>
<, <'*label*'> *values* <*divisor*=*n*>> <, . . . >
</ *options*> ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among the least squares means. In contrast to the hypotheses tested with the ESTIMATE or CONTRAST statements, the LSMESTIMATE statement enables you to form linear combinations of the least squares means, rather than linear combination of fixed-effects parameter estimates and/or random-effects solutions. Multiple-row sets of coefficients are permitted.

The computation of an LSMESTIMATE involves two coefficient matrices. Suppose that the *fixed-effect* has n_l levels. Then the LS-means are formed as $\mathbf{L}_1\widehat{\boldsymbol{\beta}}$, where $\mathbf{L}_1$ is a $(n_l \times p)$ coefficient matrix. The $(k \times n_l)$ coefficient matrix $\mathbf{K}$ is formed from the *values* that you supply in the k rows of the LSMESTIMATE statement. The least squares means estimates then represent the $(k \times 1)$ vector

$$\mathbf{K}\mathbf{L}_1\boldsymbol{\beta} = \mathbf{L}\boldsymbol{\beta}$$

The GLIMMIX procedure supports nonpositional syntax for the coefficients (*values*) in the LSMESTIMATE statement. For details see the section "Positional and Nonpositional Syntax for Contrast Coefficients" on page 3973.

PROC GLIMMIX produces a t test for each row of coefficients specified in the LSMESTIMATE statement. You can adjust p-values and confidence intervals for multiplicity with the ADJUST= option. You can obtain an F test of single-row or multirow LSMESTIMATEs with the FTEST option.

Note that in contrast to a multirow estimate in the ESTIMATE statement, you specify only a single fixed effect in the LSMESTIMATE statement. The row labels are optional and follow the effects specification. For example, the following statements fit a split-split-plot design and compare the average of the third and fourth LS-mean of the whole-plot factor A to the first LS-mean of the factor:

```
proc glimmix;
   class a b block;
   model y = a b a*b /  s;
   random int a / sub=block;
   lsmestimate A 'a1 vs avg(a3,a4)' 2 0 -1 -1 divisor=2;
run;
```

The order in which coefficients are assigned to the least squares means corresponds to the order in which they are displayed in the "Least Squares Means" table. You can use the ELSM option to see how coefficients are matched to levels of the *fixed-effect*.

The optional *divisor=n* specification enables you to assign a separate divisor to each row of the LSMESTIMATE. You can also assign divisor values through the DIVISOR= option. See the documentation that follows for the interaction between the two ways of specifying divisors.

Many options of the LSMESTIMATE statement affect the computation of least squares means—for example, the AT=, BYLEVEL, and OM options. See the documentation for the LSMEANS statement for details.

Table 52.10 summarizes the *options* available in the LSMESTIMATE statement.

Table 52.10 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJDFE=	Determines denominator degrees of freedom when p-values and confidence limits are adjusted for multiple comparisons

Table 52.10 *continued*

Option	Description
ADJUST=	Determines the method for multiple comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
CHISQ	Requests a chi-square test in addition to the F test
DF=	Assigns a specific value to degrees of freedom for tests and confidence limits
FTEST	Produces an F test
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
Generalized Linear Modeling	
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

You can specify the following *options* in the LSMESTIMATE statement after a slash (/).

ADJDFE=SOURCE | ROW

specifies how denominator degrees of freedom are determined when p-values and confidence limits are adjusted for multiple comparisons with the ADJUST= option. When you do not specify the ADJDFE= option, or when you specify ADJDFE=SOURCE, the denominator degrees of freedom for multiplicity-adjusted results are the denominator degrees of freedom for the LS-mean effect in the "Type III Tests of Fixed Effects" table.

The ADJDFE=ROW setting is useful if you want multiplicity adjustments to take into account that denominator degrees of freedom are not constant across estimates. This can be the case, for example, when DDFM=SATTERTHWAITE or DDFM=KENWARDROGER is specified in the MODEL statement.

ADJUST=BON | SCHEFFE | SIDAK | SIMULATE< (*simoptions*) > | T

requests a multiple comparison adjustment for the p-values and confidence limits for the LS-mean estimates. The adjusted quantities are produced in addition to the unadjusted p-values and confidence limits. Adjusted confidence limits are produced if the CL or ALPHA= option is in effect. For a description of the adjustments, see Chapter 53, "The GLM Procedure," and Chapter 86, "The MULTTEST Procedure," as well as the documentation for the ADJUST= option in the LSMEANS statement.

Note that not all adjustment methods of the LSMEANS statement are available for the LSMESTIMATE statement. Multiplicity adjustments in the LSMEANS statement are designed specifically for differences of least squares means.

If you specify the STEPDOWN option, the p-values are further adjusted in a step-down fashion.

ALPHA=*number*

requests that a t-type confidence interval be constructed for each of the LS-means with confidence level $1 - \mathit{number}$. The value of *number* must be between 0 and 1; the default is 0.05.

AT *variable***=***value*

AT (*variable-list***)=(***value-list***)**

AT MEANS

enables you to modify the values of the covariates used in computing LS-means. See the AT option in the LSMEANS statement for details.

BYLEVEL

requests that PROC GLIMMIX compute separate margins for each level of the LSMEANS effect.

The standard LS-means have equal coefficients across classification effects. The BYLEVEL option changes these coefficients to be proportional to the observed margins. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in the input data set. In this case, the resulting LS-means are actually equal to raw means for fixed-effects models and certain balanced random-effects models, but their estimated standard errors account for the covariance structure that you have specified. If a WEIGHT statement is specified, PROC GLIMMIX uses weighted margins to construct the LS-means coefficients.

If the AT option is specified, the BYLEVEL option disables it.

CHISQ

requests that chi-square tests be performed in addition to F tests, when you request an F test with the FTEST option.

CL

requests that t-type confidence limits be constructed for each of the LS-means. If DDFM=NONE, then PROC GLIMMIX uses infinite degrees of freedom for this test, essentially computing a z interval. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

CORR

displays the estimated correlation matrix of the linear combination of the least squares means.

COV

displays the estimated covariance matrix of the linear combination of the least squares means.

DF=*number*

specifies the degrees of freedom for the t test and confidence limits. The default is the denominator degrees of freedom taken from the "Type III Tests of Fixed Effects" table corresponding to the LS-means effect.

DIVISOR=*value-list*

specifies a list of values by which to divide the coefficients so that fractional coefficients can be entered as integer numerators. If you do not specify *value-list*, a default value of 1.0 is assumed. Missing values in the *value-list* are converted to 1.0.

If the number of elements in *value-list* exceeds the number of rows of the estimate, the extra values are ignored. If the number of elements in *value-list* is less than the number of rows of the estimate, the last value in *value-list* is carried forward.

If you specify a row-specific divisor as part of the specification of the estimate row, this value multiplies the corresponding value in the *value-list*. For example, the following statement divides the coefficients in the first row by 8, and the coefficients in the third and fourth row by 3:

```
lsmestimate A 'One vs. two'   8 -8   divisor=2,
              'One vs. three' 1  0 -1         ,
              'One vs. four'  3  0  0 -3      ,
              'One vs. five'  3  0  0  0  -3  / divisor=4,.,3;
```

Coefficients in the second row are not altered.

E

requests that the **L** coefficients of the estimable function be displayed. These are the coefficients that apply to the fixed-effect parameter estimates. The E option displays the coefficients that you would need to enter in an equivalent ESTIMATE statement.

ELSM

requests that the **K** matrix coefficients be displayed. These are the coefficients that apply to the LS-means. This option is useful to ensure that you assigned the coefficients correctly to the LS-means.

EXP

requests exponentiation of the least squares means estimate. When you model data with the logit link function and the estimate represents a log odds ratio, the EXP option produces an odds ratio. See the section "Odds and Odds Ratio Estimation" on page 3965 for important details concerning the computation and interpretation of odds and odds ratio results with the GLIMMIX procedure. If you specify the CL or ALPHA= option, the (adjusted) confidence limits for the estimate are also exponentiated.

FTEST< (*joint-test-options*) **>**

JOINT< (*joint-test-options*) **>**

produces an F test that jointly tests the rows of the LSMESTIMATE against zero. If the LOWER or UPPER options are in effect or if you specify boundary values with the BOUNDS= suboption, the GLIMMIX procedure computes a simulation-based p-value for the constrained joint test. For more information about these simulation-based p-values, see the section "Joint Hypothesis Tests with Complex Alternatives, the Chi-Bar-Square Statistic" on page 489 in Chapter 20, "Shared Concepts and Topics." You can specify the following *joint-test-options* in parentheses:

ACC=γ

specifies the accuracy radius for determining the necessary sample size in the simulation-based approach of Silvapulle and Sen (2004) for tests with order restrictions. The value of γ must be strictly between 0 and 1; the default value is 0.005.

BOUNDS=*value-list*

specifies boundary values for the estimable linear function. The null value of the hypothesis is always zero. If you specify a positive boundary value z, the hypotheses are $H\colon \theta = 0$ vs. $H_a\colon \theta > 0$ with the added constraint that $\theta < z$. The same is true for negative boundary values. The alternative hypothesis is then $H_a\colon \theta < 0$ subject to the constraint $\theta > -|z|$. If you specify a missing value, the hypothesis is assumed to be two-sided. The BOUNDS option enables you to specify sets of one- and two-sided joint hypotheses. If all values in *value-list* are set to missing, the procedure performs a simulation-based p-value calculation for a two-sided test.

EPS=ϵ

specifies the accuracy confidence level for determining the necessary sample size in the simulation-based approach of Silvapulle and Sen (2004) for F tests with order restrictions. The value of ϵ must be strictly between 0 and 1; the default value is 0.01.

LABEL='*label*'

enables you to assign a label to the joint test that identifies the results in the "LSMFtest" table. If you do not specify a label, the first non-default label for the LSMESTIMATE rows is used to label the joint test.

NSAMP=*n*

specifies the number of samples for the simulation-based method of Silvapulle and Sen (2004). If *n* is not specified, it is constructed from the values of the ALPHA=α, the ACC=γ, and the EPS=ϵ options. With the default values for γ, ϵ, and α (0.005, 0.01, and 0.05, respectively), NSAMP=12,604 by default.

ILINK

requests that the estimate and its standard error also be reported on the scale of the mean (the inverse linked scale). PROC GLIMMIX computes the value on the mean scale by applying the inverse link to the estimate. The interpretation of this quantity depends on the coefficients that are specified in your LSMESTIMATE statement and the link function. For example, in a model for binary data with a logit link, the following LSMESTIMATE statement computes

$$q = \frac{1}{1 + \exp\{-(\tau_1 - \tau_2)\}}$$

where τ_1 and τ_2 are the least squares means associated with the first two levels of the classification effect A:

```
proc glimmix;
   class A;
   model y = A / dist=binary link=logit;
   lsmestimate A 1 -1 / ilink;
run;
```

The quantity q is not the difference of the probabilities associated with the two levels,

$$\pi_1 - \pi_2 = \frac{1}{1+\exp\{-\tau_1\}} - \frac{1}{1+\exp\{-\tau_2\}}$$

The standard error of the inversely linked estimate is based on the delta method. If you also specify the CL or ALPHA= option, the GLIMMIX procedure computes confidence intervals for the inversely linked estimate. These intervals are obtained by applying the inverse link to the confidence intervals on the linked scale.

JOINT< (*joint-test-options*) >
is an alias for the FTEST option.

LOWER
LOWERTAILED
requests that the p-value for the t test be based only on values that are less than the test statistic. A two-tailed test is the default. A lower-tailed confidence limit is also produced if you specify the CL or ALPHA= option.

Note that for ADJUST=SCHEFFE the one-sided adjusted confidence intervals and one-sided adjusted p-values are the same as the corresponding two-sided statistics, because this adjustment is based on only the right tail of the F distribution.

If you request an F test with the FTEST option, then a one-sided left-tailed order restriction is applied to all estimable functions, and the corresponding chi-bar-square statistic of Silvapulle and Sen (2004) is computed in addition to the two-sided, standard F or chi-square statistic. See the description of the FTEST option for information about how to control the computation of the simulation-based chi-bar-square statistic.

OBSMARGINS
OM
specifies a potentially different weighting scheme for the computation of LS-means coefficients. The standard LS-means have equal coefficients across classification effects; however, the OM option changes these coefficients to be proportional to those found in the input data set. See the OBSMARGINS option in the LSMEANS statement for further details.

SINGULAR=*number*
tunes the estimability checking as documented for the CONTRAST statement.

STEPDOWN< (*step-down-options*) >
requests that multiplicity adjustments for the p-values of LS-mean estimates be further adjusted in a step-down fashion. Step-down methods increase the power of multiple testing procedures by taking advantage of the fact that a p-value will never be declared significant unless all smaller p-values are also declared significant. Note that the STEPDOWN adjustment combined with ADJUST=BON corresponds to the Holm (1979) and "Method 2" of Shaffer (1986); this is the default. Using step-down-adjusted p-values combined with ADJUST=SIMULATE corresponds to the method of Westfall (1997).

If the degrees-of-freedom method is DDFM=KENWARDROGER or DDFM=SATTERTHWAITE, then step-down-adjusted p-values are produced only if the ADJDFE=ROW option is in effect.

Also, the STEPDOWN option affects only *p*-values, not confidence limits. For ADJUST=SIMULATE, the generalized least squares hybrid approach of Westfall (1997) is employed to increase Monte Carlo accuracy.

You can specify the following *step-down-options* in parentheses:

MAXTIME=*n*
specifies the time (in seconds) to spend computing the maximal logically consistent sequential subsets of equality hypotheses for TYPE=LOGICAL. The default is MAXTIME=60. If the MAXTIME value is exceeded, the adjusted tests are not computed. When this occurs, you can try increasing the MAXTIME value. However, note that there are common multiple comparisons problems for which this computation requires a huge amount of time—for example, all pairwise comparisons between more than 10 groups. In such cases, try to use TYPE=FREE (the default) or TYPE=LOGICAL(*n*) for small *n*.

ORDER=PVALUE | ROWS
specifies the order in which the step-down tests are performed. ORDER=PVALUE is the default, with LS-mean estimates being declared significant only if all LS-mean estimates with smaller (unadjusted) *p*-values are significant. If you specify ORDER=ROWS, then significances are evaluated in the order in which they are specified.

REPORT
specifies that a report on the step-down adjustment be displayed, including a listing of the sequential subsets (Westfall 1997) and, for ADJUST=SIMULATE, the step-down simulation results.

TYPE=LOGICAL< (*n*) > | FREE
If you specify TYPE=LOGICAL, the step-down adjustments are computed by using maximal logically consistent sequential subsets of equality hypotheses (Shaffer 1986; Westfall 1997). Alternatively, for TYPE=FREE, logical constraints are ignored when sequential subsets are computed. The TYPE=FREE results are more conservative than those for TYPE=LOGICAL, but they can be much more efficient to produce for many estimates. For example, it is not feasible to take logical constraints between all pairwise comparisons of more than about 10 groups. For this reason, TYPE=FREE is the default.

However, you can reduce the computational complexity of taking logical constraints into account by limiting the depth of the search tree used to compute them, specifying the optional depth parameter as a number *n* in parentheses after TYPE=LOGICAL. As with TYPE=FREE, results for TYPE=LOGICAL(*n*) are conservative relative to the true TYPE=LOGICAL results, but even for TYPE=LOGICAL(0), they can be appreciably less conservative than TYPE=FREE, and they are computationally feasible for much larger numbers of estimates. If you do not specify *n* or if *n* = –1, the full search tree is used.

UPPER
UPPERTAILED
requests that the *p*-value for the *t* test be based only on values that are greater than the test statistic. A two-tailed test is the default. An upper-tailed confidence limit is also produced if you specify the CL or ALPHA= option.

Note that for ADJUST=SCHEFFE the one-sided adjusted confidence intervals and one-sided adjusted p-values are the same as the corresponding two-sided statistics, because this adjustment is based on only the right tail of the F distribution.

If you request a joint test with the FTEST option, then a one-sided right-tailed order restriction is applied to all estimable functions, and the corresponding chi-bar-square statistic of Silvapulle and Sen (2004) is computed in addition to the two-sided, standard F or chi-square statistic. See the FTEST option for information about how to control the computation of the simulation-based chi-bar-square statistic.

MARGINS Statement

MARGINS *fixed-effects* < / *options* > **;**

The MARGINS statement computes predictive margins of fixed effects. Predictive margins can be computed for any effect in the MODEL statement that involves only classification variables.

The predictive margin for a specific level (group) of a fixed effect represents the average predicted response if all the observations in the data set had been in that group (Lane and Nelder 1982; Chang, Gelman, and Pagano 1982). You compute the predictive margin by fixing this effect at the specified level for all observations and averaging the predicted responses. The standard error of the predictive margin is computed by using the delta method.

For example, consider the logistic regression $\log P(y_{ij} = 1)/(1 - P(y_{ij} = 1)) = \alpha_i + \beta x_{ij}$, where α_i is the effect of the ith level of the classification effect α, $i = 1, \ldots, R$, and x_{ij} is the covariate for the jth observation in the ith level of the effect α, $j = 1, \ldots, n_i$. Then the predictive margin for level r of effect α is

$$\frac{1}{N}\sum_{i=1}^{R}\sum_{j=1}^{n_i} \exp(\hat{\alpha}_r + \hat{\beta}x_{ij})/(1 + \exp(\hat{\alpha}_r + \hat{\beta}x_{ij}))$$

where $N = \sum_{i=1}^{R} n_i$.

By definition, predictive margins are covariate-adjusted marginal means, as are LS-means. These two approaches are discussed in the section "Predictive Margins Compared with LS-Means" on page 3956.

You can specify multiple effects in one or multiple MARGINS statements, and all MARGINS statements must appear after the MODEL statement. Predictive margin computations are not supported for t, lognormal, or multinomial distributions or for constructed effects.

PROC GLIMMIX constructs an approximate t test to test the null hypothesis that the associated population quantity equals zero. By default, the denominator degrees of freedom for this test are the same as those displayed for the effect in the "Type III Tests of Fixed Effects" table. Table 52.11 summarizes options available in the MARGINS statement. All options for the MARGINS statement are discussed in alphabetical order after the table.

Table 52.11 MARGINS Statement Options

Option	Description
Construction and Computation of Predictive Margins	
AT	Modifies covariate value in computing predictive margins
DIFF	Requests differences of predictive margins
SLICEBY=	Partitions F tests
SLICEDIFF=	Requests differences of sliced predictive margins and determines the type of differences
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of predictive margin differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
DF=	Assigns specific value to degrees of freedom for tests and confidence limits
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion
Statistical Output	
CL	Constructs confidence limits for predictive margins and/or predictive margin differences

You can specify the following *options* in the MARGINS statement after a slash (/).

ADJUST=BON

ADJUST=DUNNETT

ADJUST=SCHEFFE

ADJUST=SIDAK

ADJUST=SIMULATE< (*simoptions*) >

ADJUST=SMM | GT2

ADJUST=TUKEY

requests a multiple comparison adjustment for the p-values and confidence limits for the differences of predictive margins. The adjusted quantities are produced in addition to the unadjusted quantities. By default, PROC GLIMMIX performs all pairwise differences. If you specify ADJUST=DUNNETT, the procedure analyzes all differences with a control level. The ADJUST= option does not imply the DIFF option.

The BON (Bonferroni) and SIDAK adjustments involve correction factors described in Chapter 53, "The GLM Procedure," and Chapter 86, "The MULTTEST Procedure"; also see Westfall and Young (1993) and Westfall et al. (1999). When you specify ADJUST=TUKEY and your data are unbalanced, PROC GLIMMIX uses the approximation described in Kramer (1956) and identifies the adjustment as "Tukey-Kramer" in the results. Similarly, when you specify ADJUST=DUNNETT and the predictive margins are correlated, the GLIMMIX procedure uses the factor-analytic covariance approximation described in Hsu (1992) and identifies the adjustment in the results as "Dunnett-Hsu". The approximation derives an approximate "effective sample sizes" for which exact critical values are computed. Note that computing the exact adjusted p-values and critical values for unbalanced designs can be computationally intensive. A simulation-based approach, as specified by the ADJUST=SIM option,

while nondeterministic, can provide inferences that are sufficiently accurate in much less time. The preceding references also describe the SCHEFFE and SMM adjustments.

The SIMULATE adjustment computes adjusted p-values and confidence limits from the simulated distribution of the maximum or maximum absolute value of a multivariate t random vector. All covariance parameters, except the residual scale parameter, are fixed at their estimated values throughout the simulation, potentially resulting in some underdispersion. The simulation estimates q, the true $(1-\alpha)$ quantile, where $1-\alpha$ is the confidence coefficient. The default α is 0.05, and you can change this value with the ALPHA= option in the MARGINS statement.

The number of samples is set so that the tail area for the simulated q is within γ of $1-\alpha$ with $100(1-\epsilon)\%$ confidence. In equation form,

$$\Pr(|F(\widehat{q})-(1-\alpha)| \leq \gamma) = 1-\epsilon$$

where $\hat{q}$ is the simulated q and F is the true distribution function of the maximum; see Edwards and Berry (1987) for details. By default, $\gamma = 0.005$ and $\epsilon = 0.01$, placing the tail area of $\hat{q}$ within 0.005 of 0.95 with 99% confidence. The ACC= and EPS= *simoptions* reset γ and ϵ, respectively, the NSAMP= *simoption* sets the sample size directly, and the SEED= *simoption* specifies an integer used to start the pseudo-random number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. For additional descriptions of these and other simulation options, see the section "LSMEANS Statement" on page 4172 in Chapter 53, "The GLM Procedure."

If the STEPDOWN option is in effect, the p-values are further adjusted in a step-down fashion. For certain options and data, this adjustment is exact under an iid $N(0,\sigma^2)$ model for the dependent variable, in particular for the following:

- for ADJUST=DUNNETT when the means are uncorrelated
- for ADJUST=TUKEY with STEPDOWN(TYPE=LOGICAL) when the means are balanced and uncorrelated.

The first case is a consequence of the nature of the successive step-down hypotheses for comparisons with a control; the second employs an extension of the maximum studentized range distribution appropriate for partition hypotheses (Royen 1989). Finally, for STEPDOWN(TYPE=FREE), ADJUST=TUKEY employs the Royen (1989) extension in such a way that the resulting p-values are conservative.

ALPHA=*number*

requests that a t-type confidence interval be constructed for each of the predictive margins with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

AT *variable***=***value*

AT (*variable-list***)=(***value-list***)**

AT MEANS

enables you to modify the values of the covariates used in computing predictive margins. By default, all covariate effects are set equal to their observed values for computation of predictive margins. The AT option enables you to assign arbitrary values to the covariates. Additional columns in the output table indicate the values of the covariates.

If there is an effect containing two or more covariates, the AT MEANS option sets the effect equal to the product of the individual means rather than the mean of the product.

As an example, consider the following invocation of PROC GLIMMIX:

```
proc glimmix;
   class A;
   model Y = A x1 x2 x1*x2;
   margins A;
   margins A / at means;
   margins A / at x1=1.2;
   margins A / at (x1 x2)=(1.2 0.3);
run;
```

The first MARGINS statement sets x1 and x2 to their observed values for each observation. In the second MARGINS statement, x1 is set to $\overline{x}_1$ (the mean of x1) and x2 is set to $\overline{x}_2$ (the mean of x2). Their interaction is set to $\overline{x}_1 \times \overline{x}_2$. The third MARGINS statement sets x1 to 1.2, and the final MARGINS statement sets these variables to 1.2 and 0.3, respectively.

CL

requests that t-type confidence limits be constructed for each of the predictive margins. If DDFM=NONE, then PROC GLIMMIX uses infinite degrees of freedom for this test, essentially computing a z interval. The confidence level is 0.95 by default; this can be changed with the ALPHA= option. If you specify an ADJUST= option, then the confidence limits are adjusted for multiplicity, but if you also specify STEPDOWN, then only p-values are step-down adjusted, not the confidence limits.

DF=*number*

specifies the degrees of freedom for the t test and confidence limits. The default is the denominator degrees of freedom taken from the "Type III Tests of Fixed Effects" table corresponding to the margins effect.

DIFF<=*difftype***>**

PDIFF<=*difftype***>**

requests that differences of the predictive margins be displayed. The optional *difftype* specifies which differences to produce, with possible values ALL, CONTROL, CONTROLL, and CONTROLU. The ALL value requests all pairwise differences, and it is the default. The CONTROL *difftype* requests differences with a control, which, by default, is the first level of each of the specified MARGINS effects.

To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the CONTROL keyword. For example, for CLASS variables A, B, and C, each having two levels, 1 and 2, the following MARGINS statement specifies the (1,2) level of A*B and the (2,1) level of B*C as controls:

```
margins A*B B*C / diff=control('1' '2', '2' '1');
```

For an effect that has multiple class variables, the quoted formatted values are assigned to the class variables in the order they appear in the CLASS statement.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*. The CONTROLL *difftype* tests whether the

noncontrol levels are significantly smaller than the control; the upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing. Conversely, the CONTROLU *difftype* tests whether the noncontrol levels are significantly larger than the control; the upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

If you want to perform multiple comparison adjustments on the differences of predictive margins, you must specify the ADJUST= option.

SLICEBY=*fixed-effect* | **(***fixed-effects***)**

specifies effects by which to partition interaction MARGINS effects. This produces tests of margins for each level of the specified slice effect.

For example, suppose that the margins effect is A*B. To test the margins of A*B for each level of A, you can specify the following statement

```
margins A*B / sliceby=A;
```

The SLICEBY option produces an F tests that test the simultaneous equality of the margins at a fixed level of the slice effect A. You can request differences of the margins while holding one or more factors at a fixed level with the SLICEDIFF= option.

SLICEDIFF< =*difftype* **>**

requests that the specified type of differences be constructed for the sliced margins and tested against zero. The possible values for the *difftype* are ALL, CONTROL, CONTROLL, and CONTROLU. The *difftype* ALL requests all pairwise differences, and it is the default. The *difftype* CONTROL, CONTROLL, and CONTROLU request the differences with a control. Whereas the SLICEBY option tests the simultaneous equality of the margins at a fixed level of the slice effect, the SLICEDIFF option tests pairwise differences of these margins. This enables you to perform multiple comparisons among the levels of one factor at a fixed level of the other factor.

For example, assume that, in a certain design, factors A and B have $a = 4$ and $b = 3$ levels, respectively. Consider the following statements:

```
proc glimmix;
   class A B;
   model y = A B A*B;
   margins A*B / sliceby=A;
   margins A*B / sliceby=A slicediff=all;
run;
```

The first MARGINS statement produces four F tests, one per level of A. Denote the three margins that correspond to the first level of A $\mathbf{m}_{a1}^{(1)}$, $\mathbf{m}_{a1}^{(2)}$, and $\mathbf{m}_{a1}^{(3)}$. Then the first F test tests the two-degrees-of-freedom hypothesis

$$H: \begin{cases} \mathbf{m}_{a1}^{(1)} - \mathbf{m}_{a1}^{(2)} = 0 \\ \mathbf{m}_{a1}^{(1)} - \mathbf{m}_{a1}^{(3)} = 0 \end{cases}$$

The SLICEDIFF option performs tests of the difference between all pairs of these three margins. In the example this corresponds to tests of the form

$$H: \mathbf{m}_{a1}^{(1)} - \mathbf{m}_{a1}^{(2)} = 0$$
$$H: \mathbf{m}_{a1}^{(1)} - \mathbf{m}_{a1}^{(3)} = 0$$
$$H: \mathbf{m}_{a1}^{(2)} - \mathbf{m}_{a1}^{(3)} = 0$$

In the example, with $a = 4$ and $b = 3$, the second MARGINS statement produces four sets of predictive margins differences. Within each set, factor A is held fixed at a particular level and each set consists of three comparisons.

For differences with a control, the default control is the first level of each of the specified MARGINS effect. To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the keyword CONTROL.

For example, if the effects A, B, and C are classification variables, each having three levels (1, 2, and 3), the following MARGINS statement specifies the (1,3) level of A*B as the control:

```
margins A*B / sliceby=(A B)
              slicediff=control('1' '3');
```

This MARGINS statement first produces predictive margins differences holding the levels of A fixed, and then it produces predictive margins differences holding the levels of B fixed. In the former case, level '3' of B serves as the control level. In the latter case, level '1' of A serves as the control.

For an effect that has multiple class variables, the quoted formatted values are assigned to the class variables in the order they appear in the CLASS statement.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*. The CONTROLL *difftype* tests whether the noncontrol levels are significantly smaller than the control; the upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing. Conversely, the CONTROLU *difftype* tests whether the noncontrol levels are significantly larger than the control; the upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

When the ADJUST= option is specified, the GLIMMIX procedure also adjusts the tests for multiplicity. The adjustment is based on the number of comparisons within each level of the SLICEBY= effect.

STEPDOWN< (*step-down options*) >
requests that multiple comparison adjustments for the p-values of predictive margins differences be further adjusted in a step-down fashion. Step-down methods increase the power of multiple comparisons by taking advantage of the fact that a p-value will never be declared significant unless all smaller p-values are also declared significant. Note that the STEPDOWN adjustment combined with ADJUST=BON corresponds to the methods of Holm (1979) and "Method 2" of Shaffer (1986); this is the default. Using step-down-adjusted p-values combined with ADJUST=SIMULATE corresponds to the method of Westfall (1997).

STEPDOWN affects only p-values, not confidence limits. For ADJUST=SIMULATE, the generalized least squares hybrid approach of Westfall (1997) is employed to increase Monte Carlo accuracy.

You can specify the following *step-down options* in parentheses:

MAXTIME=*n*

specifies the time (in seconds) to spend computing the maximal logically consistent sequential subsets of equality hypotheses for TYPE=LOGICAL. The default is MAXTIME=60. If the MAXTIME value is exceeded, the adjusted tests are not computed. When this occurs, you can try increasing the MAXTIME value. However, note that there are common multiple comparisons problems for which this computation requires a huge amount of time—for example, all pairwise comparisons between more than 10 groups. In such cases, try to use TYPE=FREE (the default) or TYPE=LOGICAL(*n*) for small *n*.

TYPE=LOGICAL< (*n***) > | FREE**

If you specify TYPE=LOGICAL, the step-down adjustments are computed by using maximal logically consistent sequential subsets of equality hypotheses (Shaffer 1986; Westfall 1997). Alternatively, for TYPE=FREE, sequential subsets are computed ignoring logical constraints. The TYPE=FREE results are more conservative than those for TYPE=LOGICAL, but they can be much more efficient to produce for many comparisons. For example, it is not feasible to take logical constraints between all pairwise comparisons of more than 10 groups. For this reason, TYPE=FREE is the default.

However, you can reduce the computational complexity of taking logical constraints into account by limiting the depth of the search tree used to compute them, specifying the optional depth parameter as a number *n* in parentheses after TYPE=LOGICAL. As with TYPE=FREE, results for TYPE=LOGICAL(*n*) are conservative relative to the true TYPE=LOGICAL results, but even for TYPE=LOGICAL(0) they can be appreciably less conservative than TYPE=FREE and they are computationally feasible for much larger numbers of comparisons. If you do not specify *n* or if $n = -1$, the full search tree is used.

MODEL Statement

MODEL *response* < **(***response-options***)** > **=** < *fixed-effects* > < / *model-options* > **;**

MODEL *events/trials* **=** < *fixed-effects* > < / *model-options* > **;**

The MODEL statement is required and names the dependent variable and the fixed effects. The *fixed-effects* determine the **X** matrix of the model (see the section "Notation for the Generalized Linear Mixed Model" for details). The specification of effects is the same as in the GLM or MIXED procedure. In contrast to PROC GLM, you do not specify random effects in the MODEL statement. However, in contrast to PROC GLM and PROC MIXED, continuous variables on the left and right side of the MODEL statement can be computed through PROC GLIMMIX programming statements.

An intercept is included in the fixed-effects model by default. It can be removed with the NOINT option.

The dependent variable can be specified by using either the *response* syntax or the *events/trials* syntax. The *events/trials* syntax is specific to models for binomial data. A binomial(n,π) variable is the sum of n independent Bernoulli trials with event probability π. Each Bernoulli trial results in either an event or a nonevent (with probability $1 - \pi$). You use the *events/trials* syntax to indicate to the GLIMMIX procedure that the Bernoulli outcomes are grouped. The value of the second variable, *trials*, gives the number n of Bernoulli trials. The value of the first variable, *events*, is the number of events out of n. The values of both *events* and (*trials*–*events*) must be nonnegative and the value of trials must be positive. Observations for which these conditions are not met are excluded from the analysis. If the *events/trials* syntax is used, the

GLIMMIX procedure defaults to the binomial distribution. The response is then the *events* variable. The *trials* variable is accounted in model fitting as an additional weight. If you use the *response* syntax, the procedure defaults to the normal distribution.

There are two sets of options in the MODEL statement. The *response-options* determine how the GLIMMIX procedure models probabilities for binary and multinomial data. The *model-options* control other aspects of model formation and inference. Table 52.12 summarizes the options available in the MODEL statement. These are subsequently discussed in detail in alphabetical order by option category.

Table 52.12 MODEL Statement Options

Option	Description
Response Variable Options	
DESCENDING	Reverses the order of response categories
EVENT=	Specifies the event category in binary models
ORDER=	Specifies the sort order for the response variable
REFERENCE=	Specifies the reference category in generalized logit models
Model Building	
DIST=	Specifies the response distribution
LINK=	Specifies the link function
NOINT	Excludes fixed-effect intercept from model
OFFSET=	Specifies the offset variable for linear predictor
OBSWEIGHT=	Specifies the weight variable for the observation level unit
Statistical Computations	
ALPHA=α	Determines the confidence level $(1-\alpha)$
CHISQ	Requests chi-square tests
DDF=	Specifies the denominator degrees of freedom (list)
DDFM=	Specifies the method for computing denominator degrees of freedom
HTYPE=	Selects the type of hypothesis test
LWEIGHT	Determines how weights are used
NOCENTER	Suppresses centering and scaling of $\mathbf{X}$ columns during the estimation phase
REFLINP	Specifies a value for the linear predictor
ZETA=	Tunes sensitivity in computing Type III functions
Statistical Output	
CL	Displays confidence limits for fixed-effects parameter estimates
CORRB	Displays the correlation matrix of fixed-effects parameter estimates
COVB	Displays the covariance matrix of fixed-effects parameter estimates
COVBI	Displays the inverse covariance matrix of fixed-effects parameter estimates
E, E1, E2, E3	Displays the $\mathbf{L}$ matrix coefficients
INTERCEPT	Adds a row for the intercept to test tables
ODDSRATIO	Displays odds ratios and confidence limits
SOLUTION	Displays fixed-effects parameter estimates (and scale parameter in GLM models)
STDCOEF	Displays standardized coefficients

Response Variable Options

Response variable options determine how the GLIMMIX procedure models probabilities for binary and multinomial data.

You can specify the following *options* by enclosing them in parentheses after the response variable. See the section "Response-Level Ordering and Referencing" on page 3976 for more detail and examples.

DESCENDING
DESC
reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC GLIMMIX orders the response categories according to the ORDER= option and then reverses that order.

EVENT='*category***' | *keyword***
specifies the event category for the binary response model. PROC GLIMMIX models the probability of the event category. The EVENT= option has no effect when there are more than two response categories. You can specify the value (formatted, if a format is applied) of the event category in quotes, or you can specify one of the following *keywords*:

FIRST
designates the first ordered category as the event. This is the default.

LAST
designates the last ordered category as the event.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the response variable. When ORDER=FORMATTED (the default) for numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC GLIMMIX run or in the DATA step that created the data set), the levels are ordered by their internal (numeric) value. If you specify the ORDER= option in the MODEL statement and the ORDER= option in the PROC GLIMMIX statement, the former takes precedence. The following table shows the interpretation of the ORDER= values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For the FORMATTED and INTERNAL values, the sort order is machine dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | *keyword*

REF='*category*' | *keyword*

specifies the reference category for the generalized logit model and the binary response model. For the generalized logit model, each nonreference category is contrasted with the reference category. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotes, or you can specify one of the following *keywords*:

FIRST

designates the first ordered category as the reference category.

LAST

designates the last ordered category as the reference category. This is the default.

Model Options

ALPHA=*number*

requests that a t-type confidence interval be constructed for each of the fixed-effects parameters with confidence level $1 - \mathit{number}$. The value of *number* must be between 0 and 1; the default is 0.05.

CHISQ

requests that chi-square tests be performed for all specified effects in addition to the F tests. Type III tests are the default; you can produce the Type I and Type II tests by using the HTYPE= option.

CL

requests that t-type confidence limits be constructed for each of the fixed-effects parameter estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

CORRB

produces the correlation matrix from the approximate covariance matrix of the fixed-effects parameter estimates.

COVB< (DETAILS) >

produces the approximate variance-covariance matrix of the fixed-effects parameter estimates $\widehat{\boldsymbol{\beta}}$. In a generalized linear mixed model this matrix typically takes the form $(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}$ and can be obtained by sweeping the mixed model equations; see the section "Estimated Precision of Estimates" on page 3926. In a model without random effects, it is obtained from the inverse of the observed or expected Hessian matrix. Which Hessian is used in the computation depends on whether the procedure is in scoring mode (see the SCORING= option in the PROC GLIMMIX statement) and whether the EXPHESSIAN option is in effect. Note that if you use EMPIRICAL= or DDFM=KENWARDROGER, the matrix displayed by the COVB option is the empirical (sandwich) estimator or the adjusted estimator, respectively.

The DETAILS suboption of the COVB option enables you to obtain a table of statistics about the covariance matrix of the fixed effects. If an adjusted estimator is used because of the EMPIRICAL= or DDFM=KENWARDROGER option, the GLIMMIX procedure displays statistics for the adjusted and unadjusted estimators as well as statistics comparing them. This enables you to diagnose, for example, changes in rank (because of an insufficient number of subjects for the empirical estimator) and to assess the extent of the covariance adjustment. In addition, the GLIMMIX procedure then displays the unadjusted (=model-based) covariance matrix of the fixed-effects parameter estimates. For more details, see the section "Exploring and Comparing Covariance Matrices" on page 3955.

COVBI

produces the inverse of the approximate covariance matrix of the fixed-effects parameter estimates.

DDF=*value-list*

DF=*value-list*

enables you to specify your own denominator degrees of freedom for the fixed effects. The *value-list* specification is a list of numbers or missing values (.) separated by commas. The degrees of freedom should be listed in the order in which the effects appear in the "Type III Tests of Fixed Effects" table. If you want to retain the default degrees of freedom for a particular effect, use a missing value for its location in the list. For example, the statement assigns 3 denominator degrees of freedom to A and 4.7 to A*B, while those for B remain the same:

```
model Y = A B A*B / ddf=3,.,4.7;
```

If you select a degrees-of-freedom method with the DDFM= option, then nonmissing, positive values in *value-list* override the degrees of freedom for the particular effect. For example, the statement assigns 3 and 6 denominator degrees of freedom in the test of the A main effect and the A*B interaction, respectively:

```
model Y = A B A*B / ddf=3,.,6  ddfm=Satterth;
```

The denominator degrees of freedom for the test for the B effect are determined from a Satterthwaite approximation.

Note that the DDF= and DDFM= options determine the degrees of freedom in the "Type I Tests of Fixed Effects," "Type II Tests of Fixed Effects," and "Type III Tests of Fixed Effects" tables. These degrees of freedom are also used in determining the degrees of freedom in tests and confidence intervals from the CONTRAST, ESTIMATE, LSMEANS, and LSMESTIMATE statements. Exceptions from this rule are noted in the documentation for the respective statements.

DDFM=*method*

specifies the method for computing the denominator degrees of freedom for the tests of fixed effects that result from the MODEL, CONTRAST, ESTIMATE, LSMEANS, and LSMESTIMATE statements.

You can specify the following *methods*:

BETWITHIN

BW

assigns within-subject degrees of freedom to a fixed effect if the fixed effect changes within a subject, and between-subject degrees of freedom otherwise. This method is the default for models with only R-side random effects and a SUBJECT= option. Computational details can be found in the section "Degrees of Freedom Methods" on page 3949.

CONTAIN

CON

invokes the containment method to compute denominator degrees of freedom. This method is the default when the model contains G-side random effects. Computational details can be found in the section "Degrees of Freedom Methods" on page 3949.

KENWARDROGER< (FIRSTORDER) >

KENROGER< (FIRSTORDER) >

KR< (FIRSTORDER) >

applies the (prediction) standard error and degrees-of-freedom correction detailed by Kenward and Roger (1997). This approximation involves adjusting the estimated variance-covariance matrix of the fixed and random effects in a manner similar to that of Prasad and Rao (1990); Harville and Jeske (1992); Kackar and Harville (1984). Satterthwaite-type degrees of freedom are then computed based on this adjustment. Computational details can be found in the section "Degrees of Freedom Methods" on page 3949.

KENWARDROGER2

KENROGER2

KR2

applies the (prediction) standard error and degrees-of-freedom correction that are detailed by Kenward and Roger (2009). This correction further reduces the precision estimator bias for the fixed and random effects under nonlinear covariance structures. Computational details can be found in the section "Degrees of Freedom Methods" on page 3949.

NONE

specifies that no denominator degrees of freedom be applied. PROC GLIMMIX then essentially assumes that infinite degrees of freedom are available in the calculation of p-values. The p-values for t tests are then identical to p-values that are derived from the standard normal distribution. In the case of F tests, the p-values are equal to those of chi-square tests that are determined as follows: if F_{obs} is the observed value of the F test with l numerator degrees of freedom, then

$$p = \Pr\{F_{l,\infty} > F_{obs}\} = \Pr\{\chi^2_l > lF_{obs}\}$$

Regardless of the DDFM= method, you can obtain these chi-square p-values with the CHISQ option in the MODEL statement.

RESIDUAL

RES

performs all tests by using the residual degrees of freedom, $n - \mathrm{rank}(\mathbf{X})$, where n is the sum of the frequencies of observations or the sum of frequencies of *event/trials* pairs. This method is the default degrees of freedom method for GLMs and overdispersed GLMs.

SATTERTHWAITE

SAT

performs a general Satterthwaite approximation for the denominator degrees of freedom in a generalized linear mixed model. This method is a generalization of the techniques that are described in Giesbrecht and Burns (1985); McLean and Sanders (1988); Fai and Cornelius (1996). The method can also include estimated random effects. Computational details can be found in the section "Degrees of Freedom Methods" on page 3949.

When the asymptotic variance matrix of the covariance parameters is found to be singular, a generalized inverse is used. Covariance parameters with zero variance then do not contribute to the degrees of freedom adjustment for DDFM=SATTERTH and DDFM=KENWARDROGER, and a message is written to the log.

DISTRIBUTION=*keyword*
DIST=*keyword*
D=*keyword*
ERROR=*keyword*
E=*keyword*

specifies the built-in (conditional) probability distribution of the data. If you specify the DIST= option and you do not specify a user-defined link function, a default link function is chosen according to the following table. If you do not specify a distribution, the GLIMMIX procedure defaults to the normal distribution for continuous response variables and to the multinomial distribution for classification or character variables, unless the *events/trial* syntax is used in the MODEL statement. If you choose the *events/trial* syntax, the GLIMMIX procedure defaults to the binomial distribution.

Table 52.14 lists the values of the DIST= option and the corresponding default link functions. For the case of generalized linear models with these distributions, you can find expressions for the log-likelihood functions in the section "Maximum Likelihood" on page 3917.

Table 52.14 Keyword Values of the DIST= Option

DIST=	Distribution	Default Link Function	Numeric Value
BETA	Beta	Logit	12
BINARY	Binary	Logit	4
BINOMIAL | **BIN** | **B**	Binomial	Logit	3
EXPONENTIAL | **EXPO**	Exponential	Log	9
GAMMA | **GAM**	Gamma	Log	5
GAUSSIAN | *G* | **NORMAL** | **N**	Normal	Identity	1
GEOMETRIC | **GEOM**	Geometric	Log	8
INVGAUSS | **IGAUSSIAN** | **IG**	Inverse Gaussian	Inverse squared (power(–2))	6
LOGNORMAL | **LOGN**	Lognormal	Identity	11
MULTINOMIAL | **MULTI** | **MULT**	Multinomial	Cumulative logit	NA
NEGBINOMIAL | **NEGBIN** | **NB**	Negative binomial	Log	7
POISSON | **POI** | **P**	Poisson	Log	2
TCENTRAL | **TDIST** | **T**	*t*	Identity	10
BYOBS(*variable*)	Multivariate	Varied	NA

Note that the PROC GLIMMIX default link for the gamma or exponential distribution is not the canonical link (the reciprocal link). If you specify DIST=BINOMIAL for a dependent variable that you specify using the *response* syntax, the dependent variable is assumed to be the ratio of the number of events to the number of trials. Observations whose dependent variable values are not between 0 and 1 are excluded from the analysis.

The numeric value in the last column of Table 52.14 can be used in combination with DIST=BYOBS. The BYOBS(*variable*) syntax designates a variable whose value identifies the distribution to which an observation belongs. If the variable is numeric, its values must match values in the last column of Table 52.14. If the variable is not numeric, an observation's distribution is identified by the first four

characters of the distribution's name in the leftmost column of the table. Distributions whose numeric value is "NA" cannot be used with DIST=BYOBS.

If the variable in BYOBS(*variable*) is a data set variable, it can also be used in the CLASS statement of the GLIMMIX procedure. For example, this provides a convenient method to model multivariate data jointly while varying fixed-effects components across outcomes. Assume that, for example, for each patient, a count and a continuous outcome were observed; the count data are modeled as Poisson data and the continuous data are modeled as gamma variates. The following statements fit a Poisson and a gamma regression model simultaneously:

```
proc sort data=yourdata;
   by patient;
run;
data yourdata;
   set yourdata;
   by patient;
   if first.patient then dist='POIS' else dist='GAMM';
run;
proc glimmix data=yourdata;
   class treatment dist;
   model y = dist treatment*dist / dist=byobs(dist);
run;
```

The two models have separate intercepts and treatment effects. To correlate the outcomes, you can share a random effect between the observations from the same patient:

```
proc glimmix data=yourdata;
   class treatment dist patient;
   model y = dist treatment*dist / dist=byobs(dist);
   random intercept / subject=patient;
run;
```

Or, you could use an R-side correlation structure:

```
proc glimmix data=yourdata;
   class treatment dist patient;
   model y = dist treatment*dist / dist=byobs(dist);
   random _residual_ / subject=patient type=un;
run;
```

Although DIST=BYOBS(*variable*) is used to model multivariate data, you only need a single response variable in PROC GLIMMIX. The responses are in "univariate" form. This allows, for example, different missing value patterns across the responses. It does, however, require that all response variables be numeric.

The default links that are assigned when DIST=BYOBS is in effect correspond to the respective default links in Table 52.14.

When you choose DIST=LOGNORMAL, the GLIMMIX procedure models the logarithm of the response variable as a normal random variable. That is, the mean and variance are estimated on the

logarithmic scale, assuming a normal distribution, $\log\{Y\} \sim N(\mu, \sigma^2)$. This enables you to draw on options that require a distribution in the exponential family—for example, by using a scoring algorithm in a GLM. To convert means and variances for $\log\{Y\}$ into those of Y, use the relationships

$$\begin{aligned} \mathrm{E}[Y] &= \exp\{\mu\}\sqrt{\omega} \\ \mathrm{Var}[Y] &= \exp\{2\mu\}\omega(\omega - 1) \\ \omega &= \exp\{\sigma^2\} \end{aligned}$$

The DIST=T option models the data as a shifted and scaled central t variable. This enables you to model data with heavy-tailed distributions. If Y denotes the response and X has a t_ν distribution with ν degrees of freedom, then PROC GLIMMIX models

$$Y = \mu + \sqrt{\phi}X$$

In this parameterization, Y has mean μ and variance $\phi\nu/(\nu - 2)$.

By default, $\nu = 3$. You can supply different degrees of freedom for the t variate as in the following statements:

```
proc glimmix;
   class a b;
   model y = b x b*x / dist=tcentral(9.6);
   random a;
run;
```

The GLIMMIX procedure does not accept values for the degrees of freedom parameter less than 3.0. If the t distribution is used with the DIST=BYOBS(*variable*) specification, the degrees of freedom are fixed at $\nu = 3$. For mixed models where parameters are estimated based on linearization, choosing DIST=T instead of DIST=NORMAL affects only the residual variance, which decreases by the factor $\nu/(\nu - 2)$.

The DIST=BETA option implements the parameterization of the beta distribution in Ferrari and Cribari-Neto (2004). If Y has a $\mathrm{beta}(\alpha, \beta)$ density, so that $\mathrm{E}[Y] = \mu = \alpha/(\alpha + \beta)$, this parameterization uses the variance function $a(\mu) = \mu(1 - \mu)$ and $\mathrm{Var}[Y] = a(\mu)/(1 + \phi)$.

See the section "Maximum Likelihood" on page 3917 for the log likelihoods of the distributions fitted by the GLIMMIX procedure.

E

requests that Type I, Type II, and Type III **L** matrix coefficients be displayed for all specified effects.

E1 | EI

requests that Type I **L** matrix coefficients be displayed for all specified effects.

E2 | EII

requests that Type II **L** matrix coefficients be displayed for all specified effects.

E3 | EIII

requests that Type III **L** matrix coefficients be displayed for all specified effects.

HTYPE=*value-list*

indicates the type of hypothesis test to perform on the fixed effects. Valid entries for values in the *value-list* are 1, 2, and 3, corresponding to Type I, Type II, and Type III tests. The default value is 3. You can specify several types by separating the values with a comma or a space. The ODS table names are "Tests1," "Tests2," and "Tests3" for the Type I, Type II, and Type III tests, respectively.

INTERCEPT

adds a row to the tables for Type I, II, and III tests corresponding to the overall intercept.

LINK=*keyword*

specifies the link function in the generalized linear mixed model. The *keywords* and their associated built-in link functions are shown in Table 52.15.

Table 52.15 Built-in Link Functions of the GLIMMIX Procedure

LINK=	Link Function	$g(\mu) = \eta =$	Numeric Value
CUMCLL | CCLL	Cumulative Complementary log-log	$\log(-\log(1-\pi))$	NA
CUMLOGIT | CLOGIT	Cumulative logit	$\log(\gamma/(1-\pi))$	NA
CUMLOGLOG	Cumulative log-log	$-\log(-\log(\pi))$	NA
CUMPROBIT | CPROBIT	Cumulative probit	$\Phi^{-1}(\pi)$	NA
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$	5
GLOGIT | GENLOGIT	Generalized logit		NA
IDENTITY | ID	Identity	μ	1
LOG	Log	$\log(\mu)$	4
LOGIT	Logit	$\log(\mu/(1-\mu))$	2
LOGLOG	Log-log	$-\log(-\log(\mu))$	6
PROBIT	Probit	$\Phi^{-1}(\mu)$	3
POWER(λ) | **POW**(λ)	Power with exponent λ= *number*	$\begin{cases} \mu^{\lambda} & \text{if } \lambda \neq 0 \\ \log(\mu) & \text{if } \lambda = 0 \end{cases}$	NA
POWERMINUS2	Power with exponent -2	$1/\mu^2$	8
RECIPROCAL | INVERSE	Reciprocal	$1/\mu$	7
BYOBS(*variable*)	Varied	Varied	NA

For the probit and cumulative probit links, $\Phi^{-1}(\cdot)$ denotes the quantile function of the standard normal distribution. For the other cumulative links, π denotes a cumulative category probability. The cumulative and generalized logit link functions are appropriate only for the multinomial distribution. When you choose a cumulative link function, PROC GLIMMIX assumes that the data are ordinal. When you specify LINK=GLOGIT, the GLIMMIX procedure assumes that the data are nominal (not ordered).

The numeric value in the rightmost column of Table 52.15 can be used in conjunction with LINK=BYOBS(*variable*). This syntax designates a *variable* whose values identify the link function associated with an observation. If the variable is numeric, its values must match those in the last

column of Table 52.15. If the variable is not numeric, an observation's link function is determined by the first four characters of the link's name in the first column. Those link functions whose numeric value is "NA" cannot be used with LINK=BYOBS(*variable*).

You can define your own link function through programming statements. See the section "User-Defined Link or Variance Function" on page 3913 for more information about how to specify a link function. If a user-defined link function is in effect, the specification in the LINK= option is ignored. If you specify neither the LINK= option nor a user-defined link function, then the default link function is chosen according to Table 52.14.

LWEIGHT=FIRSTORDER | FIRO

LWEIGHT=NONE

LWEIGHT=VAR

determines how weights are used in constructing the coefficients for Type I through Type III **L** matrices. The default is LWEIGHT=VAR, and the values of the WEIGHT variable are used in forming crossproduct matrices. If you specify LWEIGHT=FIRO, the weights incorporate the WEIGHT variable as well as the first-order weights of the linearized model. For LWEIGHT=NONE, the **L** matrix coefficients are based on the raw crossproduct matrix, whether a WEIGHT variable is specified or not.

NOCENTER

requests that the columns of the **X** matrix are not centered and scaled. By default, the columns of **X** are centered and scaled. Unless the NOCENTER option is in effect, **X** is replaced by $\mathbf{X}^*$ during estimation. The columns of $\mathbf{X}^*$ are computed as follows:

- In models with an intercept, the intercept column remains the same and the jth entry in row i of $\mathbf{X}^*$ is

$$x^*_{ij} = \frac{x_{ij} - \overline{x}_j}{\sqrt{\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}}$$

- In models without intercept, no centering takes place and the jth entry in row i of $\mathbf{X}^*$ is

$$x^*_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{n}(x_{ij} - \overline{x}_j)^2}}$$

The effects of centering and scaling are removed when results are reported. For example, if the covariance matrix of the fixed effects is printed with the COVB option of the MODEL statement, the covariances are reported in terms of the original parameters, not the centered and scaled versions. If you specify the STDCOEF option, fixed-effects parameter estimates and their standard errors are reported in terms of the standardized (scaled and/or centered) coefficients in addition to the usual results in noncentered form.

NOINT

requests that no intercept be included in the fixed-effects model. An intercept is included by default.

ODDSRATIO< (*odds-ratio-options*) >

OR< (*odds-ratio-options*) >

requests estimates of odds ratios and their confidence limits, provided the link function is the logit, cumulative logit, or generalized logit. Odds ratios are produced for the following:

- classification main effects, if they appear in the MODEL statement
- continuous variables in the MODEL statement, unless they appear in an interaction with a classification effect
- continuous variables in the MODEL statement at fixed levels of a classification effect, if the MODEL statement contains an interaction of the two
- continuous variables in the MODEL statement, if they interact with other continuous variables

You can specify the following *odds-ratio-options* to create customized odds ratio results.

AT *var-list*=*value-list*

specifies the reference values for continuous variables in the model. By default, the average value serves as the reference. Consider, for example, the following statements:

```
proc glimmix;
   class A;
   model y = A x A*x / dist=binary oddsratio;
run;
```

Odds ratios for A are based on differences of least squares means for which x is set to its mean. Odds ratios for x are computed by differencing two sets of least squares mean for the A factor. One set is computed at $x = \overline{x} + 1$, and the second set is computed at $x = \overline{x}$. The following MODEL statement changes the reference value for x to 3:

```
model y = A x A*x / dist=binary
                    oddsratio(at x=3);
```

DIFF< =*difftype* >

controls the type of differences for classification main effects. By default, odds ratios compare the odds of a response for level *j* of a factor to the odds of the response for the last level of that factor (DIFF=LAST). The DIFF=FIRST option compares the levels against the first level, DIFF=ALL produces odds ratios based on all pairwise differences, and DIFF=NONE suppresses odds ratios for classification main effects.

LABEL

displays a label in the "Odds Ratio Estimates" table. The table describes the comparison associated with the table row.

UNIT *var-list*=*value-list*

specifies the units in which the effects of continuous variable in the model are assessed. By default, odds ratios are computed for a change of one unit from the average. Consider a model with a classification factor A with 4 levels. The following statements produce an "Odds Ratio Estimates" table with 10 rows:

```
proc glimmix;
   class A;
   model y = A x A*x / dist=binary
                       oddsratio(diff=all unit x=2);
run;
```

The first $4 \times 3/2 = 6$ rows correspond to pairwise differences of levels of A. The underlying log odds ratios are computed as differences of A least squares means. In the least squares mean computation the covariate x is set to $\overline{x}$. The next four rows compare least squares means for A at x = $\overline{x} + 2$ and at x = $\overline{x}$. You can combine the AT and UNIT options to produce custom odds ratios. For example, the following statements produce an "Odds Ratio Estimates" table with 8 rows:

```
proc glimmix;
   class A;
   model y = A x x*z / dist=binary
                       oddsratio(diff=all
                                 at   x   = 3
                                 unit x z = 2 4);
run;
```

The first $4 \times 3/2 = 6$ rows correspond to pairwise differences of levels of A. The underlying log odds ratios are computed as differences of A least squares means. In the least squares mean computation, the covariate x is set to 3, and the covariate x*z is set to $3\overline{z}$. The next odds ratio measures the effect of a change in x. It is based on differencing the linear predictor for x = $3 + 2$ and x*z = $(3 + 2)\overline{z}$ with the linear predictor for x = 3 and x*z = $3\overline{z}$. The last odds ratio expresses a change in z by contrasting the linear predictors based on x = 3 and x*z = $3(\overline{z} + 4)$ with the predictor based on x = 3 and x*z = $3\overline{z}$.

To compute odds and odds ratios for general estimable functions and least squares means, see the ODDSRATIO option in the LSMEANS statement and the EXP options in the ESTIMATE and LSMESTIMATE statements.

For important details concerning interpretation and computation of odds ratios with the GLIMMIX procedure, see the section "Odds and Odds Ratio Estimation" on page 3965.

OFFSET=*variable*

specifies a variable to be used as an offset for the linear predictor. An offset plays the role of a fixed effect whose coefficient is known to be 1. You can use an offset in a Poisson model, for example, when counts have been obtained in time intervals of different lengths. With a log link function, you can model the counts as Poisson variables with the logarithm of the time interval as the offset variable. The offset variable cannot appear in the CLASS statement or elsewhere in the MODEL or RANDOM statement.

REFLINP=*r*

specifies a value for the linear predictor of the reference level in the generalized logit model for nominal data. By default *r*=0.

SOLUTION

S

requests that a solution for the fixed-effects parameters be produced. Using notation from the section "Notation for the Generalized Linear Mixed Model" on page 3781, the fixed-effects parameter estimates are $\widehat{\boldsymbol{\beta}}$, and their (approximate) estimated standard errors are the square roots of the diagonal elements of $\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}]$. This matrix commonly is of the form $(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}$ in GLMMs. You can output this approximate variance matrix with the COVB option. See the section "Details: GLIMMIX Procedure" on page 3917 on the construction of $\widehat{\mathbf{V}}$ in the various models.

Along with the estimates and their approximate standard errors, a t statistic is computed as the estimate divided by its standard error. The degrees of freedom for this t statistic matches the one appearing in the "Type III Tests of Fixed Effects" table under the effect containing the parameter. If DDFM=KENWARDROGER or DDFM=SATTERTHWAITE, the degrees of freedom are computed separately for each fixed-effect estimate, unless you override the value for any specific effect with the DDF=*value-list* option. The "Pr > |t|" column contains the two-tailed p-value corresponding to the t statistic and associated degrees of freedom. You can use the CL option to request confidence intervals for the fixed-effects parameters; they are constructed around the estimate by using a radius of the standard error times a percentage point from the t distribution.

STDCOEF

reports solutions for fixed effects in terms of the standardized (scaled and/or centered) coefficients. This option has no effect when the NOCENTER option is specified or in models for multinomial data.

OBSWEIGHT< =*variable*>

OBSWT< =*variable*>

specifies a variable to be used as the weight for the observation-level unit in a weighted multilevel model. If a weight variable is not specified in the OBSWEIGHT option, a weight of 1 is used. For details on the use of weights in multilevel models, see the section "Pseudo-likelihood Estimation for Weighted Multilevel Models" on page 3938.

ZETA=*number*

tunes the sensitivity in forming Type III functions. Any element in the estimable function basis with an absolute value less than *number* is set to 0. The default is 1E–8.

NLOPTIONS Statement

NLOPTIONS < *options* > **;**

Most models fit with the GLIMMIX procedure typically have one or more nonlinear parameters. Estimation requires nonlinear optimization methods. You can control the optimization through options in the NLOPTIONS statement.

Several estimation methods of the GLIMMIX procedure (METHOD=RSPL, MSPL, RMPL, MMPL) are doubly iterative in the following sense. The generalized linear mixed model is approximated by a linear mixed model based on current values of the covariance parameter estimates. The resulting linear mixed model is then fit, which is itself an iterative process (with some exceptions). On convergence, new covariance parameters and fixed-effects estimates are obtained and the approximated linear mixed model is updated. Its parameters are again estimated iteratively. It is thus reasonable to refer to *outer* and *inner* iterations.

The outer iterations involve the repeated updates of the linear mixed models, and the inner iterations are the iterative steps that lead to parameter estimates in any given linear mixed model. The NLOPTIONS statement controls the inner iterations. The outer iteration behavior can be controlled with options in the PROC GLIMMIX statement, such as the MAXLMMUPDATE=, PCONV=, and ABSPCONV= options. If the estimation method involves a singly iterative approach, then there is no need for the outer cycling and the model is fit in a single optimization controlled by the NLOPTIONS statement (see the section "Singly or Doubly Iterative Fitting" on page 3979).

The syntax and options of the NLOPTIONS statement are described in the section "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics."

Note that in a GLMM with pseudo-likelihood estimation, specifying TECHNIQUE=NONE has the same effect as specifying the NOITER option in the PARMS statement. If you estimate the parameters by METHOD=LAPLACE or METHOD=QUAD, TECHNIQUE=NONE applies to the optimization after starting values have been determined.

The GLIMMIX procedure applies the default optimization technique shown in Table 52.16, depending on your model.

Table 52.16 Default Techniques

Model Family	Setting	TECHNIQUE=
GLM	DIST=NORMAL LINK=IDENTITY	NONE
GLM	Otherwise	NEWRAP
GLMM	PARMS NOITER, PL	NONE
GLMM	Binary data, PL	NRRIDG
GLMM	Otherwise	QUANEW

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< *keyword*< (*keyword-options*) > < **=***name* > > . . .
< *keyword*< (*keyword-options*) > < **=***name* > > < / *options* > **;**

The OUTPUT statement creates a data set that contains predicted values and residual diagnostics, computed after fitting the model. By default, all variables in the original data set are included in the output data set.

You can use the ID statement to select a subset of the variables from the input data set as well as computed variables for adding to the output data set. If you reassign a data set variable through programming statements, the value of the variable from the input data set supersedes the recomputed value when observations are written to the output data set. If you list the variable in the ID statement, however, PROC GLIMMIX saves the current value of the variable after the programming statements have been executed.

For example, suppose that data set Scores contains the variables score, machine, and person. The following statements fit a model with fixed machine and random person effects. The variable score divided by 100 is

assumed to follow an inverse Gaussian distribution. The (conditional) mean and residuals are saved to the data set igausout. Because no ID statement is given, the variable score in the output data set contains the values from the input data set.

```
proc glimmix;
   class machine person;
   score = score/100;
   p = 4*_linp_;
   model score = machine  / dist=invgauss;
   random int / sub=person;
   output out=igausout pred=p resid=r;
run;
```

On the contrary, the following statements list explicitly which variables to save to the OUTPUT data set. Because the variable score is listed in the ID statement, and is (re-)assigned through programming statements, the values of score saved to the OUTPUT data set are the input values divided by 100.

```
proc glimmix;
   class machine person;
   score = score / 100;
   model score = machine  / dist=invgauss;
   random int / sub=person;
   output out=igausout pred=p resid=r;
   id machine score _xbeta_ _zgamma_;
run;
```

You can specify the following syntax elements in the OUTPUT statement before the slash (/).

OUT=*SAS-data-set*
specifies the name of the output data set. If the OUT= option is omitted, the procedure uses the DATA*n* convention to name the output data set.

keyword<(*keyword-options*)><=*name*>
specifies a statistic to include in the output data set and optionally assigns the variable the name *name*. You can use the *keyword-options* to control which type of a particular statistic to compute. The *keyword-options* can take on the following values:

BLUP uses the predictors of the random effects in computing the statistic.

ILINK computes the statistic on the scale of the data.

NOBLUP does not use the predictors of the random effects in computing the statistic.

NOILINK computes the statistic on the scale of the link function.

The default is to compute statistics by using BLUPs on the scale of the link function (the linearized scale). For example, the following OUTPUT statements are equivalent:

```
output out=out1  pred=predicted lcl=lower;
```

```
output out=out1  pred(blup noilink)=predicted
                 lcl (blup noilink)=lower;
```

If a particular combination of keyword and keyword options is not supported, the statistic is not computed and a message is produced in the SAS log.

A *keyword* can appear multiple times in the OUTPUT statement. Table 52.17 lists the *keywords* and the default names assigned by the GLIMMIX procedure if you do not specify a *name*. In this table, y denotes the observed response, and p denotes the linearized pseudo-data. See the section "Pseudo-likelihood Estimation Based on Linearization" on page 3924 for details on notation and the section "Notes on Output Statistics" on page 3986 for further details regarding the output statistics.

Table 52.17 Keywords for Output Statistics

Keyword	Options	Description	Expression	Name
PREDICTED	Default	Linear predictor	$\widehat{\eta} = \mathbf{x}'\widehat{\boldsymbol{\beta}} + \mathbf{z}'\widehat{\boldsymbol{\gamma}}$	Pred
	NOBLUP	Marginal linear predictor	$\widehat{\eta}_m = \mathbf{x}'\widehat{\boldsymbol{\beta}}$	PredPA
	ILINK	Predicted mean	$g^{-1}(\widehat{\eta})$	PredMu
	NOBLUP ILINK	Marginal mean	$g^{-1}(\widehat{\eta}_m)$	PredMuPA
STDERR	Default	Standard deviation of linear predictor	$\sqrt{\mathrm{Var}[\widehat{\eta} - \mathbf{z}'\boldsymbol{\gamma}]}$	StdErr
	NOBLUP	Standard deviation of marginal linear predictor	$\sqrt{\mathrm{Var}[\widehat{\eta}_m]}$	StdErrPA
	ILINK	Standard deviation of mean	$\sqrt{\mathrm{Var}[g^{-1}(\widehat{\eta} - \mathbf{z}'\boldsymbol{\gamma})]}$	StdErr
	NOBLUP ILINK	Standard deviation of marginal mean	$\sqrt{\mathrm{Var}[g^{-1}(\widehat{\eta}_m)]}$	StdErrMuPA
RESIDUAL	Default	Residual	$r = p - \widehat{\eta}$	Resid
	NOBLUP	Marginal residual	$r_m = p_m - \widehat{\eta}_m$	ResidPA
	ILINK	Residual on mean scale	$r_y = y - g^{-1}(\widehat{\eta})$	ResidMu
	NOBLUP ILINK	Marginal residual on mean scale	$r_{ym} = y - g^{-1}(\widehat{\eta}_m)$	ResidMuPA
PEARSON	Default	Pearson-type residual	$r/\sqrt{\widehat{\mathrm{Var}}[p\|\boldsymbol{\gamma}]}$	Pearson
	NOBLUP	Marginal Pearson-type residual	$r_m/\sqrt{\widehat{\mathrm{Var}}[p_m]}$	PearsonPA
	ILINK	Conditional Pearson-type mean residual	$r_y/\sqrt{\widehat{\mathrm{Var}}[Y\|\boldsymbol{\gamma}]}$	PearsonMu
STUDENT	Default	Studentized residual	$r/\sqrt{\widehat{\mathrm{Var}}[r]}$	Student
	NOBLUP	Studentized marginal residual	$r_m/\sqrt{\widehat{\mathrm{Var}}[r_m]}$	StudentPA

Table 52.17 *continued*

Keyword	Options	Description	Expression	Name
LCL	Default	Lower prediction limit for linear predictor		LCL
	NOBLUP	Lower confidence limit for marginal linear predictor		LCLPA
	ILINK	Lower prediction limit for mean		LCLMu
	NOBLUP ILINK	Lower confidence limit for marginal mean		LCLMuPA
UCL	Default	Upper prediction limit for linear predictor		UCL
	NOBLUP	Upper confidence limit for marginal linear predictor		UCLPA
	ILINK	Upper prediction limit for mean		UCLMu
	NOBLUP ILINK	Upper confidence limit for marginal mean		UCLMuPA
VARIANCE	Default	Conditional variance of pseudo-data	$\widehat{\mathrm{Var}}[p\|\boldsymbol{\gamma}]$	Variance
	NOBLUP	Marginal variance of pseudo-data	$\widehat{\mathrm{Var}}[p_m]$	VariancePA
	ILINK	Conditional variance of response	$\widehat{\mathrm{Var}}[Y\|\boldsymbol{\gamma}]$	Variance_Dep
	NOBLUP ILINK	Marginal variance of response	$\widehat{\mathrm{Var}}[Y]$	Variance_DepPA

Studentized residuals are computed only on the linear scale (scale of the link), unless the link is the identity, in which case the two scales are equal. The keywords RESIDUAL, PEARSON, STUDENT, and VARIANCE are not available with the multinomial distribution. You can use the following shortcuts to request statistics: PRED for PREDICTED, STD for STDERR, RESID for RESIDUAL, and VAR for VARIANCE. Output statistics that depend on the marginal variance $\mathrm{Var}[Y_i]$ are not available with METHOD=LAPLACE or METHOD=QUAD.

Table 52.18 summarizes the *options* available in the OUTPUT statement.

Table 52.18 OUTPUT Statement Options

Option	Description
ALLSTATS	Computes all statistics
ALPHA=α	Determines the confidence level $(1-\alpha)$
CPSEUDO	Changes the way in which marginal residuals are computed

Table 52.18 *continued*

Option	Description
DERIVATIVES	Adds derivatives of model quantities to the output data set
NOMISS	Outputs only observations used in the analysis
NOUNIQUE	Requests that names not be made unique
NOVAR	Requests that variables from the input data set not be added to the output data set
OBSCAT	Writes statistics to output data set only for the response level corresponding to the observed level of the observation
SYMBOLS	Adds computed variables to the output data set

You can specify the following *options* in the OUTPUT statement after a slash (/).

ALLSTATS

requests that all statistics are computed. If you do not use a keyword to assign a name, the GLIMMIX procedure uses the default name.

ALPHA=*number*

determines the coverage probability for two-sided confidence and prediction intervals. The coverage probability is computed as 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

CPSEUDO

changes the way in which marginal residuals are computed when model parameters are estimated by pseudo-likelihood methods. See the section "Notes on Output Statistics" on page 3986 for details.

DERIVATIVES

DER

adds derivatives of model quantities to the output data set. If, for example, the model fit requires the (conditional) log likelihood of the data, then the DERIVATIVES option writes for each observation the evaluations of the first and second derivatives of the log likelihood with respect to _LINP_ and _PHI_ to the output data set. The particular derivatives produced by the GLIMMIX procedure depend on the type of model and the estimation method.

NOMISS

requests that records be written to the output data only for those observations that were used in the analysis. By default, the GLIMMIX procedure produces output statistics for all observations in the input data set.

NOUNIQUE

requests that names not be made unique in the case of naming conflicts. By default, the GLIMMIX procedure avoids naming conflicts by assigning a unique name to each output variable. If you specify the NOUNIQUE option, variables with conflicting names are not renamed. In that case, the first variable added to the output data set takes precedence.

NOVAR

requests that variables from the input data set not be added to the output data set. This option does not apply to variables listed in the BY statement or to computed variables listed in the ID statement.

OBSCAT

requests that in models for multinomial data statistics be written to the output data set only for the response level that corresponds to the observed level of the observation.

SYMBOLS

SYM

adds to the output data set computed variables that are defined or referenced in the program.

PARMS Statement

PARMS <(*value-list*)> . . . </ *options*> **;**

The PARMS statement specifies initial values for the covariance or scale parameters, or it requests a grid search over several values of these parameters in generalized linear mixed models.

The *value-list* specification can take any of several forms:

m	a single value
$m_1, m_2, \ldots, m_n$	several values
m to n	a sequence where m equals the starting value, n equals the ending value, and the increment equals 1
m to n by i	a sequence where m equals the starting value, n equals the ending value, and the increment equals i
m_1, m_2 to m_3	mixed values and sequences

Using the PARMS Statement with a GLM

If you are fitting a GLM or a GLM with overdispersion, the scale parameters are listed at the end of the "Parameter Estimates" table in the same order as *value-list*. If you specify more than one set of initial values, PROC GLIMMIX uses only the first value listed for each parameter. Grid searches by using scale parameters are not possible for these models, because the fixed effects are part of the optimization.

Using the PARMS Statement with a GLMM

If you are fitting a GLMM, the *value-list* corresponds to the parameters as listed in the "Covariance Parameter Estimates" table. Note that this order can change depending on whether a residual variance is profiled or not; see the NOPROFILE option in the PROC GLIMMIX statement.

If you specify more than one set of initial values, PROC GLIMMIX performs a grid search of the objective function surface and uses the best point on the grid for subsequent analysis. Specifying a large number of grid points can result in long computing times.

Options in the PARMS Statement

You can specify the following *options* in the PARMS statement after a slash (/).

HOLD=*value-list*

specifies which parameter values PROC GLIMMIX should hold equal to the specified values. For example, the following statement constrains the first and third covariance parameters to equal 5 and 2, respectively:

```
parms (5) (3) (2) (3) / hold=1,3;
```

Covariance or scale parameters that are held fixed with the HOLD= option are treated as constrained parameters in the optimization. This is different from evaluating the objective function, gradient, and Hessian matrix at known values of the covariance parameters. A constrained parameter introduces a singularity in the optimization process. The covariance matrix of the covariance parameters (see the ASYCOV option of the PROC GLIMMIX statement) is then based on the projected Hessian matrix. As a consequence, the variance of parameters subjected to a HOLD= is zero. Such parameters do not contribute to the computation of denominator degrees of freedom with the DDFM=KENWARDROGER and DDFM=SATTERTHWAITE methods, for example. If you want to treat the covariance parameters as known, without imposing constraints on the optimization, you should use the NOITER option.

When you place a hold on all parameters (or when you specify the NOITER) option in a GLMM, you might notice that PROC GLIMMIX continues to produce an iteration history. Unless your model is a linear mixed model, several recomputations of the pseudo-response might be required in linearization-based methods to achieve agreement between the pseudo-data and the covariance matrix. In other words, the GLIMMIX procedure continues to update the fixed-effects estimates (and random-effects solutions) until convergence is achieved.

In certain models, placing a hold on covariance parameters implies that the procedure processes the parameters in the same order as if the NOPROFILE were in effect. This can change the order of the covariance parameters when you place a hold on one or more parameters. Models that are subject to this reordering are those with R-side covariance structures whose scale parameter could be profiled. This includes the TYPE=CS, TYPE=SP, TYPE=AR(1), TYPE=TOEP, and TYPE=ARMA(1,1) covariance structures.

LOWERB=*value-list*

enables you to specify lower boundary constraints for the covariance or scale parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the same order that PROC GLIMMIX uses for the *value-list* in the PARMS statement, and each number corresponds to the lower boundary constraint. A missing value instructs PROC GLIMMIX to use its default constraint, and if you do not specify numbers for all of the covariance parameters, PROC GLIMMIX assumes that the remaining ones are missing.

This option is useful, for example, when you want to constrain the **G** matrix to be positive definite in order to avoid the more computationally intensive algorithms required when **G** becomes singular. The corresponding statements for a random coefficients model are as follows:

```
proc glimmix;
   class person;
   model y = time;
```

```
   random int time / type=chol sub=person;
   parms / lowerb=1e-4,.,1e-4;
run;
```

Here, the TYPE=CHOL structure is used in order to specify a Cholesky root parameterization for the 2×2 unstructured blocks in **G**. This parameterization ensures that the **G** matrix is nonnegative definite, and the PARMS statement then ensures that it is positive definite by constraining the two diagonal terms to be greater than or equal to 1E–4.

NOBOUND

requests the removal of boundary constraints on covariance and scale parameters in mixed models. For example, variance components have a default lower boundary constraint of 0, and the NOBOUND option allows their estimates to be negative. See the NOBOUND option in the PROC GLIMMIX statement for further details.

NOITER

requests that no optimization of the covariance parameters be performed. This option has no effect in generalized linear models.

If you specify the NOITER option, PROC GLIMMIX uses the values for the covariance parameters given in the PARMS statement to perform statistical inferences. Note that the NOITER option is not equivalent to specifying a HOLD= value for all covariance parameters. If you use the NOITER option, covariance parameters are not constrained in the optimization. This prevents singularities that might otherwise occur in the optimization process.

If a residual variance is profiled, the parameter estimates can change from the initial values you provide as the residual variance is recomputed. To prevent an update of the residual variance, combine the NOITER option with the NOPROFILE option in the PROC GLIMMIX statements, as in the following code:

```
proc glimmix noprofile;
   class A B C rep mp sp;
   model y = A | B | C;
   random rep mp sp;
   parms (180) (200) (170) (1000) / noiter;
run;
```

When you specify the NOITER option in a model where parameters are estimated by pseudo-likelihood techniques, you might notice that the GLIMMIX procedure continues to produce an iteration history. Unless your model is a linear mixed model, several recomputations of the pseudo-response might be required in linearization-based methods to achieve agreement between the pseudo-data and the covariance matrix. In other words, the GLIMMIX procedure continues to update the profiled fixed-effects estimates (and random-effects solutions) until convergence is achieved. To prevent these updates, use the MAXLMMUPDATE= option in the PROC GLIMMIX statement. Specifying the NOITER option in the PARMS statement of a GLMM with pseudo-likelihood estimation has the same effect as choosing TECHNIQUE=NONE in the NLOPTIONS statement.

If you want to base initial fixed-effects estimates on the results of fitting a generalized linear model, then you can combine the NOITER option with the TECHNIQUE= option. For example, the following statements determine the starting values for the fixed effects by fitting a logistic model (without random effects) with the Newton-Raphson algorithm:

```
proc glimmix startglm inititer=10;
   class clinic A;
   model y/n = A / link=logit dist=binomial;
   random clinic;
   parms (0.4) / noiter;
   nloptions technique=newrap;
run;
```

The initial GLM fit stops at convergence or after at most 10 iterations, whichever comes first. The pseudo-data for the linearized GLMM is computed from the GLM estimates. The variance of the Clinic random effect is held constant at 0.4 during subsequent iterations that update the fixed effects only.

If you also want to combine the GLM fixed-effects estimates with known and fixed covariance parameter values without updating the fixed effects, you can add the MAXLMMUPDATE=0 option:

```
proc glimmix startglm inititer=10 maxlmmupdate=0;
   class clinic A;
   model y/n = A / link=logit dist=binomial;
   random clinic;
   parms (0.4) / noiter;
   nloptions technique=newrap;
run;
```

In a GLMM with parameter estimation by METHOD=LAPLACE or METHOD=QUAD the NOITER option also leads to an iteration history, since the fixed-effects estimates are part of the optimization and the PARMS statement places restrictions on only the covariance parameters.

Finally, the NOITER option can be useful if you want to obtain minimum variance quadratic unbiased estimates (with 0 priors), also known as MIVQUE0 estimates (Goodnight 1978a). Because MIVQUE0 estimates are starting values for covariance parameters—unless you provide (*value-list*) in the PARMS statement—the following statements produce MIVQUE0 mixed model estimates:

```
proc glimmix noprofile;
   class A B;
   model y = A;
   random int / subject=B;
   parms / noiter;
run;
```

PARMSDATA=*SAS-data-set*

PDATA=*SAS-data-set*

reads in covariance parameter values from a SAS data set. The data set should contain the numerical variable ESTIMATE or the numerical variables Covp1–Covp$q$, where q denotes the number of covariance parameters.

If the PARMSDATA= data set contains multiple sets of covariance parameters, the GLIMMIX procedure evaluates the initial objective function for each set and commences the optimization step by using the set with the lowest function value as the starting values. For example, the following SAS statements request that the objective function be evaluated for three sets of initial values:

```
data data_covp;
   input covp1-covp4;
   datalines;
  180 200 170 1000
  170 190 160  900
  160 180 150  800
;
proc glimmix;
   class A B C rep mainEU smallEU;
   model yield = A|B|C;
   random rep mainEU smallEU;
   parms / pdata=data_covp;
run;
```

Each set comprises four covariance parameters.

The order of the observations in a data set with the numerical variable Estimate corresponds to the order of the covariance parameters in the "Covariance Parameter Estimates" table. In a GLM, the PARMSDATA= option can be used to set the starting value for the exponential family scale parameter. A grid search is not conducted for GLMs if you specify multiple values.

The PARMSDATA= data set must not contain missing values.

If the GLIMMIX procedure is processing the input data set in BY groups, you can add the BY variables to the PARMSDATA= data set. If this data set is sorted by the BY variables, the GLIMMIX procedure matches the covariance parameter values to the current BY group. If the PARMSDATA= data set does not contain all BY variables, the data set is processed in its entirety for every BY group and a message is written to the log. This enables you to provide a single set of starting values across BY groups, as in the following statements:

```
data data_covp;
   input covp1-covp4;
   datalines;
  180 200 170 1000
;


proc glimmix;
   class A B C rep mainEU smallEU;
   model yield = A|B|C;
   random rep mainEU smallEU;
   parms / pdata=data_covp;
   by year;
run;
```

The same set of starting values is used for each value of the year variable.

UPPERB=*value-list*

enables you to specify upper boundary constraints on the covariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the same order that PROC GLIMMIX uses for the *value-list* in the PARMS statement, and each number corresponds to the upper boundary constraint. A missing value instructs PROC GLIMMIX to use its default constraint. If you do not specify numbers for all of the covariance parameters, PROC GLIMMIX assumes that the remaining ones are missing.

RANDOM Statement

RANDOM *random-effects* < / *options* > **;**

Using notation from "Notation for the Generalized Linear Mixed Model" on page 3781, the RANDOM statement defines the $\mathbf{Z}$ matrix of the mixed model, the random effects in the $\boldsymbol{\gamma}$ vector, the structure of $\mathbf{G}$, and the structure of $\mathbf{R}$.

The $\mathbf{Z}$ matrix is constructed exactly like the $\mathbf{X}$ matrix for the fixed effects, and the $\mathbf{G}$ matrix is constructed to correspond to the effects constituting $\mathbf{Z}$. The structures of $\mathbf{G}$ and $\mathbf{R}$ are defined by using the TYPE= option described on page 3897. The random effects can be classification or continuous effects, and multiple RANDOM statements are possible.

Some reserved *keywords* have special significance in the *random-effects* list. You can specify INTERCEPT (or INT) as a random effect to indicate the intercept. PROC GLIMMIX does not include the intercept in the RANDOM statement by default as it does in the MODEL statement. You can specify the _RESIDUAL_ keyword (or RESID, RESIDUAL, _RESID_) before the option slash (/) to indicate a residual-type (R-side) random component that defines the $\mathbf{R}$ matrix. Basically, the _RESIDUAL_ keyword takes the place of the *random-effect* if you want to specify R-side variances and covariance structures. These *keywords* take precedence over variables in the data set with the same name. If your data or the covariance structure requires that an effect is specified, you can use the RESIDUAL option to instruct the GLIMMIX procedure to model the R-side variances and covariances.

In order to add an overdispersion component to the variance function, simply specify a single residual random component. For example, the following statements fit a polynomial Poisson regression model with overdispersion. The variance function $a(\mu) = \mu$ is replaced by $\phi a(\mu)$:

```
proc glimmix;
   model count = x x*x / dist=poisson;
   random _residual_;
run;
```

Table 52.19 summarizes the *options* available in the RANDOM statement. All *options* are subsequently discussed in alphabetical order.

Table 52.19 RANDOM Statement Options

Option	Description
Construction of Covariance Structure	
GCOORD=	Determines coordinate association for G-side spatial structures with repeat levels
GROUP=	Varies covariance parameters by groups
LDATA=	Specifies a data set with coefficient matrices for TYPE= LIN
NOFULLZ	Eliminates columns in **Z** corresponding to missing values
RESIDUAL	Designates a covariance structure as R-side
SUBJECT=	Identifies the subjects in the model
TYPE=	Specifies the covariance structure
WEIGHT=	Specifies the weights for the subjects
Mixed Model Smoothing	
KNOTINFO	Displays spline knots
KNOTMAX=	Specifies the upper limit for knot construction
KNOTMETHOD	Specifies the method for constructing knots for radial smoother and penalized B-splines
KNOTMIN=	Specifies the lower limit for knot construction
Statistical Output	
ALPHA=α	Determines the confidence level $(1-\alpha)$
CL	Requests confidence limits for predictors of random effects
G	Displays the estimated **G** matrix
GC	Displays the Cholesky root (lower) of the estimated **G** matrix
GCI	Displays the inverse Cholesky root (lower) of the estimated **G** matrix
GCORR	Displays the correlation matrix that corresponds to the estimated **G** matrix
GI	Displays the inverse of the estimated **G** matrix
SOLUTION	Displays solutions $\widehat{\boldsymbol{\gamma}}$ of the G-side random effects
V	Displays blocks of the estimated **V** matrix
VC	Displays the lower-triangular Cholesky root of blocks of the estimated **V** matrix
VCI	Displays the inverse Cholesky root of blocks of the estimated **V** matrix
VCORR	Displays the correlation matrix corresponding to blocks of the estimated **V** matrix
VI	Displays the inverse of the blocks of the estimated **V** matrix

You can specify the following *options* in the RANDOM statement after a slash (/).

ALPHA=*number*

requests that a t-type confidence interval with confidence level 1 – *number* be constructed for the predictors of G-side random effects in this statement. The value of *number* must be between 0 and 1; the default is 0.05. Specifying the ALPHA= option implies the CL option.

CL

requests that *t*-type confidence limits be constructed for each of the predictors of G-side random effects in this statement. The confidence level is 0.95 by default; this can be changed with the ALPHA= option. The CL option implies the SOLUTION option.

G

requests that the estimated **G** matrix be displayed for G-side random effects associated with this RANDOM statement. PROC GLIMMIX displays blanks for values that are 0.

GC

displays the lower-triangular Cholesky root of the estimated **G** matrix for G-side random effects.

GCI

displays the inverse Cholesky root of the estimated **G** matrix for G-side random effects.

GCOORD=LAST | FIRST | MEAN

determines how the GLIMMIX procedure associates coordinates for TYPE=SP() covariance structures with effect levels for G-side random effects. In these covariance structures, you specify one or more variables that identify the coordinates of a data point. The levels of classification variables, on the other hand, can occur multiple times for a particular subject. For example, in the following statements the same level of A can occur multiple times, and the associated values of x might be different:

```
proc glimmix;
   class A B;
   model y = B;
   random A / type=sp(pow)(x);
run;
```

The GCOORD=LAST option determines the coordinates for a level of the random effect from the last observation associated with the level. Similarly, the GCOORD=FIRST and GCOORD=MEAN options determine the coordinate from the first observation and from the average of the observations. Observations not used in the analysis are not considered in determining the first, last, or average coordinate. The default is GCOORD=LAST.

GCORR

displays the correlation matrix that corresponds to the estimated **G** matrix for G-side random effects.

GI

displays the inverse of the estimated **G** matrix for G-side random effects.

GROUP=*effect*

GRP=*effect*

identifies groups by which to vary the covariance parameters. Each new level of the grouping effect produces a new set of covariance parameters. Continuous variables and computed variables are permitted as group effects. PROC GLIMMIX does not sort by the values of the continuous variable; rather, it considers the data to be from a new group whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of groups and also prevents the production of a large "Class Levels Information" table.

Specifying a GROUP effect can greatly increase the number of estimated covariance parameters, which can adversely affect the optimization process.

KNOTINFO
displays the number and coordinates of the knots as determined by the KNOTMETHOD= option.

KNOTMAX=*number-list*
provides upper limits for the values of random effects used in the construction of knots for TYPE=RSMOOTH. The items in *number-list* correspond to the random effects of the radial smooth. If the KNOTMAX= option is not specified, or if the value associated with a particular random effect is set to missing, the maximum is based on the values in the data set for KNOTMETHOD=EQUAL or KNOTMETHOD=KDTREE, and is based on the values in the knot data set for KNOTMETHOD=DATA.

KNOTMETHOD=KDTREE<(*tree-options***)>**
KNOTMETHOD=EQUAL<(*number-list***)>**
KNOTMETHOD=DATA(*SAS-data-set***)**
determines the method of constructing knots for the radial smoother fit with the TYPE=RSMOOTH covariance structure and the TYPE=PSPLINE covariance structure.

Unless you select the TYPE=RSMOOTH or TYPE=PSPLINE covariance structure, the KNOTMETHOD= option has no effect. The default for TYPE=RSMOOTH is KNOTMETHOD=KDTREE. For TYPE=PSPLINE, only equally spaced knots are used and you can use the optional *numberlist* argument of KNOTMETHOD=EQUAL to determine the number of interior knots for TYPE=PSPLINE.

Knot Construction for TYPE=RSMOOTH

PROC GLIMMIX fits a low-rank smoother, meaning that the number of knots is considerably less than the number of observations. By default, PROC GLIMMIX determines the knot locations based on the vertices of a k-d tree (Friedman, Bentley, and Finkel 1977; Cleveland and Grosse 1991). The k-d tree is a tree data structure that is useful for efficiently determining the m nearest neighbors of a point. The k-d tree also can be used to obtain a grid of points that adapts to the configuration of the data. The process starts with a hypercube that encloses the values of the random effects. The space is then partitioned recursively by splitting cells at the median of the data in the cell for the random effect. The procedure is repeated for all cells that contain more than a specified number of points, b. The value b is called the bucket size.

The k-d tree is thus a division of the data into cells such that cells representing leaf nodes contain at most b values. You control the building of the k-d tree through the BUCKET= *tree-option*. You control the construction of knots from the cell coordinates of the tree with the other *options* as follows.

BUCKET=*number*
determines the bucket size b. A larger bucket size will result in fewer knots. For k-d trees in more than one dimension, the correspondence between bucket size and number of knots is difficult to determine. It depends on the data configuration and on other suboptions. In the multivariate case, you might need to try out different bucket sizes to obtain the desired number of knots. The default value of *number* is 4 for univariate trees (a single random effect) and $\lfloor 0.1n \rfloor$ in the multidimensional case.

KNOTTYPE=*type*
specifies whether the knots are based on vertices of the tree cells or the centroid. The two possible values of *type* are VERTEX and CENTER. The default is KNOTTYPE=VERTEX. For multidimensional smoothing, such as smoothing across irregularly shaped spatial domains, the KNOTTYPE=CENTER option is useful to move knot locations away from the bounding hypercube toward the convex hull.

NEAREST
specifies that knot coordinates are the coordinates of the nearest neighbor of either the centroid or vertex of the cell, as determined by the KNOTTYPE= suboptions.

TREEINFO
displays details about the construction of the *k*-d tree, such as the cell splits and the split values.

See the section "Knot Selection" on page 3960 for a detailed example of how the specification of the bucket size translates into the construction of a *k*-d tree and the spline knots.

The KNOTMETHOD=EQUAL option enables you to define a regular grid of knots. By default, PROC GLIMMIX constructs 10 knots for one-dimensional smooths and 5 knots in each dimension for smoothing in higher dimensions. You can specify a different number of knots with the optional *number-list*. Missing values in the *number-list* are replaced with the default values. A minimum of two knots in each dimension is required. For example, the following statements use a rectangular grid of 35 knots, five knots for x1 combined with seven knots for x2:

```
proc glimmix;
   model y=;
   random x1 x2 / type=rsmooth knotmethod=equal(5 7);
run;
```

When you use the NOFIT option in the PROC GLIMMIX statement, the GLIMMIX procedure computes the knots but does not fit the model. This can be useful if you want to compare knot selections with different suboptions of KNOTMETHOD=KDTREE. Suppose you want to determine the number of knots based on a particular bucket size. The following statements compute and display the knots in a bivariate smooth, constructed from nearest neighbors of the vertices of a *k*-d tree with bucket size 10:

```
proc glimmix nofit;
   model y = Latitude Longitude;
   random Latitude Longitude / type=rsmooth
                   knotmethod=kdtree(knottype=vertex
                   nearest bucket=10) knotinfo;
run;
```

You can specify a data set that contains variables whose values give the knot coordinates with the KNOTMETHOD=DATA option. The data set must contain numeric variables with the same name as the radial smoothing *random-effects*. PROC GLIMMIX uses only the unique knot coordinates in the knot data set. This option is useful to provide knot coordinates different from those that can be produced from a *k*-d tree. For example, in spatial problems where the domain is irregularly shaped, you might want to determine knots by a space-filling algorithm. The following SAS statements invoke the OPTEX procedure to compute 45 knots that uniformly cover the convex hull of the data locations (see *SAS/QC User's Guide* for details about the OPTEX procedure).

```
proc optex coding=none;
   model latitude longitude / noint;
   generate n=45 criterion=u method=m_fedorov;
   output out=knotdata;
run;
proc glimmix;
   model y = Latitude Longitude;
   random Latitude Longitude / type=rsmooth
                   knotmethod=data(knotdata);
run;
```

Knot Construction for TYPE=PSPLINE

Only evenly spaced knots are supported when you fit penalized B-splines with the GLIMMIX procedure. For the TYPE=PSPLINE covariance structure, the *number-list* argument specifies the number m of interior knots, the default is $m = 10$. Suppose that $x_{(1)}$ and $x_{(n)}$ denote the smallest and largest values, respectively. For a B-spline of degree d (De Boor 2001), the interior knots are supplemented with d exterior knots below $x_{(1)}$ and $\max\{1, d\}$ exterior knots above $x_{(n)}$. PROC GLIMMIX computes the location of these $m + d + \max\{1, d\}$ knots as follows. Let $\delta_x = (x_{(n)} - x_{(1)})/(m + 1)$, then interior knots are placed at

$$x_{(1)} + j\delta_x, \quad j = 1, \ldots, m$$

The exterior knots are also evenly spaced with step size δ_x and start at $x_{(1)} \pm 100$ times the machine epsilon. At least one interior knot is required.

KNOTMIN=*number-list*

provides lower limits for the values of random effects used in the construction of knots for TYPE=RSMOOTH. The items in *number-list* correspond to the random effects of the radial smooth. If the KNOTMIN= option is not specified, or if the value associated with a particular random effect is set to missing, the minimum is based on the values in the data set for KNOTMETHOD=EQUAL or KNOTMETHOD=KDTREE, and is based on the values in the knot data set for KNOTMETHOD=DATA.

LDATA=*SAS-data-set*

reads the coefficient matrices $\mathbf{A}_1, \ldots, \mathbf{A}_q$ for the TYPE=LIN(q) option. You can specify the LDATA= data set in a sparse or dense form. In the sparse form the data set must contain the numeric variables Parm, Row, Col, and Value. The Parm variable contains the indices $i = 1, \ldots, q$ of the $\mathbf{A}_i$ matrices. The Row and Col variables identify the position within a matrix and the Value variable contains the matrix element. Values not specified for a particular row and column are set to zero. Missing values are allowed in the Value column of the LDATA= data set; these values are also replaced by zeros. The sparse form is particularly useful if the $\mathbf{A}$ matrices have only a few nonzero elements.

In the dense form the LDATA= data set contains the numeric variables Parm and Row (with the same function as above), in addition to the numeric variables Col1–Col$q$. If you omit one or more of the Col1–Col$q$ variables from the data set, zeros are assumed for the respective rows and columns of the $\mathbf{A}$ matrix. Missing values for Col1–Col$q$ are ignored in the dense form.

The GLIMMIX procedure assumes that the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_q$ are symmetric. In the sparse LDATA= form you do not need to specify off-diagonal elements in position (i, j) and (j, i). One of them is

sufficient. Row-column indices are converted in both storage forms into positions in lower triangular storage. If you specify multiple values in row ($\max\{i, j\}$ and column $\min\{i, j\}$) of a particular matrix, only the last value is used. For example, assume you are specifying elements of a 4×4 matrix. The lower triangular storage of matrix $\mathbf{A}_3$ defined by

```
data ldata;
   input parm row col value;
   datalines;
3  2  1  2
3  1  2  5
;
```

is

$$\begin{bmatrix} 0 & & & \\ 5 & 0 & & \\ 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

NOFULLZ

eliminates the columns in $\mathbf{Z}$ corresponding to missing levels of random effects involving CLASS variables. By default, these columns are included in $\mathbf{Z}$. It is sufficient to specify the NOFULLZ option on any G-side RANDOM statement.

RESIDUAL

RSIDE

specifies that the random effects listed in this statement be R-side effects. You use the RESIDUAL option in the RANDOM statement if the nature of the covariance structure requires you to specify an effect. For example, if it is necessary to order the columns of the R-side AR(1) covariance structure by the time variable, you can use the RESIDUAL option as in the following statements:

```
class time id;
random time / subject=id type=ar(1) residual;
```

SOLUTION

S

requests that the solution $\widehat{\boldsymbol{\gamma}}$ for the random-effects parameters be produced, if the statement defines G-side random effects.

The numbers displayed in the Std Err Pred column of the "Solution for Random Effects" table are not the standard errors of the $\widehat{\boldsymbol{\gamma}}$ displayed in the Estimate column; rather, they are the square roots of the prediction errors $\widehat{\boldsymbol{\gamma}}_i - \boldsymbol{\gamma}_i$, where $\widehat{\boldsymbol{\gamma}}_i$ is the predictor of the ith random effect and $\boldsymbol{\gamma}_i$ is the ith random effect. In pseudo-likelihood methods that are based on linearization, these EBLUPs are the estimated best linear unbiased predictors in the linear mixed pseudo-model. In models fit by maximum likelihood by using the Laplace approximation or by using adaptive quadrature, the SOLUTION option displays the empirical Bayes estimates (EBE) of $\boldsymbol{\gamma}_i$.

SUBJECT=*effect*

SUB=*effect*

identifies the subjects in your generalized linear mixed model. Complete independence is assumed across subjects. Specifying a subject effect is equivalent to nesting all other effects in the RANDOM statement within the subject effect.

Continuous variables and computed variables are permitted with the SUBJECT= option. PROC GLIMMIX does not sort by the values of the continuous variable but considers the data to be from a new subject whenever the value of the continuous variable changes from the previous observation. Using a continuous variable can decrease execution time for models with a large number of subjects and also prevents the production of a large "Class Levels Information" table.

TYPE=*covariance-structure*

specifies the covariance structure of **G** for G-side effects and the covariance structure of **R** for R-side effects.

Although a variety of structures are available, many applications call for either simple diagonal (TYPE=VC) or unstructured covariance matrices. The TYPE=VC (variance components) option is the default structure, and it models a different variance component for each random effect. It is recommended to model unstructured covariance matrices in terms of their Cholesky parameterization (TYPE=CHOL) rather than TYPE=UN.

If you want different covariance structures in different parts of **G**, you must use multiple RANDOM statements with different TYPE= options.

Valid values for *covariance-structure* are as follows. Examples are shown in Table 52.21.

The variances and covariances in the formulas that follow in the TYPE= descriptions are expressed in terms of generic random variables ξ_i and ξ_j. They represent the G-side random effects or the residual random variables for which the **G** or **R** matrices are constructed.

ANTE(1)

specifies a first-order ante-dependence structure (Kenward 1987; Patel 1991) parameterized in terms of variances and correlation parameters. If t ordered random variables $\xi_1, \ldots, \xi_t$ have a first-order ante-dependence structure, then each ξ_j, $j > 1$, is independent of all other $\xi_k, k < j$, given ξ_{j-1}. This Markovian structure is characterized by its inverse variance matrix, which is tridiagonal. Parameterizing an ANTE(1) structure for a random vector of size t requires $2t - 1$ parameters: variances $\sigma_1^2, \ldots, \sigma_t^2$ and $t - 1$ correlation parameters $\rho_1, \ldots, \rho_{t-1}$. The covariances among random variables ξ_i and ξ_j are then constructed as

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sqrt{\sigma_i^2 \sigma_j^2} \prod_{k=i}^{j-1} \rho_k$$

PROC GLIMMIX constrains the correlation parameters to satisfy $|\rho_k| < 1, \ \forall k$. For variable-order ante-dependence models see Macchiavelli and Arnold (1994).

AR(1)

specifies a first-order autoregressive structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \rho^{|i^*-j^*|}$$

The values i^* and j^* are derived for the ith and jth observations, respectively, and are not necessarily the observation numbers. For example, in the following statements the values correspond to the class levels for the time effect of the ith and jth observation within a particular subject:

```
proc glimmix;
   class time patient;
   model y = x x*x;
   random time / sub=patient type=ar(1) residual;
run;
```

PROC GLIMMIX imposes the constraint $|\rho| < 1$ for stationarity.

ARH(1)

specifies a heterogeneous first-order autoregressive structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sqrt{\sigma_i^2 \sigma_j^2}\, \rho^{|i^*-j^*|}$$

with $|\rho| < 1$. This covariance structure has the same correlation pattern as the TYPE=AR(1) structure, but the variances are allowed to differ.

ARMA(1,1)

specifies the first-order autoregressive moving-average structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 & i = j \\ \sigma^2 \gamma \rho^{|i^*-j^*|-1} & i \neq j \end{cases}$$

Here, ρ is the autoregressive parameter, γ models a moving-average component, and σ^2 is a scale parameter. In the notation of Fuller (1976, p. 68), $\rho = \theta_1$ and

$$\gamma = \frac{(1 + b_1\theta_1)(\theta_1 + b_1)}{1 + b_1^2 + 2b_1\theta_1}$$

The example in Table 52.21 and $|b_1| < 1$ imply that

$$b_1 = \frac{\beta - \sqrt{\beta^2 - 4\alpha^2}}{2\alpha}$$

where $\alpha = \gamma - \rho$ and $\beta = 1 + \rho^2 - 2\gamma\rho$. PROC GLIMMIX imposes the constraints $|\rho| < 1$ and $|\gamma| < 1$ for stationarity, although for some values of ρ and γ in this region the resulting covariance matrix is not positive definite. When the estimated value of ρ becomes negative, the computed covariance is multiplied by $\cos(\pi d_{ij})$ to account for the negativity.

CHOL<(*q*)>

specifies an unstructured variance-covariance matrix parameterized through its Cholesky root. This parameterization ensures that the resulting variance-covariance matrix is at least positive semidefinite. If all diagonal values are nonzero, it is positive definite. For example, a 2×2 unstructured covariance matrix can be written as

$$\mathrm{Var}[\boldsymbol{\xi}] = \begin{bmatrix} \theta_1 & \theta_{12} \\ \theta_{12} & \theta_2 \end{bmatrix}$$

Without imposing constraints on the three parameters, there is no guarantee that the estimated variance matrix is positive definite. Even if θ_1 and θ_2 are nonzero, a large value for θ_{12} can lead to a negative eigenvalue of $\mathrm{Var}[\boldsymbol{\xi}]$. The Cholesky root of a positive definite matrix $\mathbf{A}$ is a lower triangular matrix $\mathbf{C}$ such that $\mathbf{CC}' = \mathbf{A}$. The Cholesky root of the above 2×2 matrix can be written as

$$\mathbf{C} = \begin{bmatrix} \alpha_1 & 0 \\ \alpha_{12} & \alpha_2 \end{bmatrix}$$

The elements of the unstructured variance matrix are then simply $\theta_1 = \alpha_1^2$, $\theta_{12} = \alpha_1\alpha_{12}$, and $\theta_2 = \alpha_{12}^2 + \alpha_2^2$. Similar operations yield the generalization to covariance matrices of higher orders.

For example, the following statements model the covariance matrix of each subject as an unstructured matrix:

```
proc glimmix;
   class sub;
   model y = x;
   random _residual_ / subject=sub type=un;
run;
```

The next set of statements accomplishes the same, but the estimated $\mathbf{R}$ matrix is guaranteed to be nonnegative definite:

```
proc glimmix;
   class sub;
   model y = x;
   random _residual_ / subject=sub type=chol;
run;
```

The GLIMMIX procedure constrains the diagonal elements of the Cholesky root to be positive. This guarantees a unique solution when the matrix is positive definite.

The optional order parameter $q > 0$ determines how many bands below the diagonal are modeled. Elements in the lower triangular portion of $\mathbf{C}$ in bands higher than q are set to zero. If you consider the resulting covariance matrix $\mathbf{A} = \mathbf{CC}'$, then the order parameter has the effect of zeroing all off-diagonal elements that are at least q positions away from the diagonal.

Because of its good computational and statistical properties, the Cholesky root parameterization is generally recommended over a completely unstructured covariance matrix (TYPE=UN). However, it is computationally slightly more involved.

CS

specifies the compound-symmetry structure, which has constant variance and constant covariance

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \phi + \sigma & i = j \\ \sigma & i \neq j \end{cases}$$

The compound symmetry structure arises naturally with nested random effects, such as when subsampling error is nested within experimental error. The models constructed with the following two sets of GLIMMIX statements have the same marginal variance matrix, provided σ is positive:

```
proc glimmix;
   class block A;
   model y = block A;
   random block*A / type=vc;
run;


proc glimmix;
   class block A;
   model y = block A;
   random _residual_ / subject=block*A
                       type=cs;
run;
```

In the first case, the block*A random effect models the G-side experimental error. Because the distribution defaults to the normal, the $\mathbf{R}$ matrix is of form $\phi\mathbf{I}$ (see Table 52.22), and ϕ is the subsampling error variance. The marginal variance for the data from a particular experimental unit is thus $\sigma^2_{b*a}\mathbf{J} + \phi\mathbf{I}$. This matrix is of compound symmetric form.

Hierarchical random assignments or selections, such as subsampling or split-plot designs, give rise to compound symmetric covariance structures. This implies exchangeability of the observations on the subunit, leading to constant correlations between the observations. Compound symmetric structures are thus usually not appropriate for processes where correlations decline according to some metric, such as spatial and temporal processes.

Note that R-side compound-symmetry structures do not impose any constraint on σ. You can thus use an R-side TYPE=CS structure to emulate a variance-component model with unbounded estimate of the variance component.

CSH

specifies the heterogeneous compound-symmetry structure, which is an equi-correlation structure but allows for different variances

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sqrt{\sigma_i^2\sigma_j^2} & i = j \\ \rho\sqrt{\sigma_i^2\sigma_j^2} & i \neq j \end{cases}$$

FA(*q*)

specifies the factor-analytic structure with q factors (Jennrich and Schluchter 1986). This structure is of the form $\mathbf{\Lambda}\mathbf{\Lambda}' + \mathbf{D}$, where $\mathbf{\Lambda}$ is a $t \times q$ rectangular matrix and $\mathbf{D}$ is a $t \times t$ diagonal matrix with t different parameters. When $q > 1$, the elements of $\mathbf{\Lambda}$ in its upper-right corner (that is, the elements in the ith row and jth column for $j > i$) are set to zero to fix the rotation of the structure.

FA0(*q*)

specifies a factor-analytic structure with q factors of the form $\mathrm{Var}[\boldsymbol{\xi}] = \mathbf{\Lambda}\mathbf{\Lambda}'$, where $\mathbf{\Lambda}$ is a $t \times q$ rectangular matrix and t is the dimension of $\mathbf{Y}$. When $q > 1$, $\mathbf{\Lambda}$ is a lower triangular matrix. When $q < t$—that is, when the number of factors is less than the dimension of the matrix—this structure is nonnegative definite but not of full rank. In this situation, you can use it to approximate an unstructured covariance matrix.

HF

specifies a covariance structure that satisfies the general Huynh-Feldt condition (Huynh and Feldt 1970). For a random vector with t elements, this structure has $t + 1$ positive parameters and covariances

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma_i^2 & i = j \\ 0.5(\sigma_i^2 + \sigma_j^2) - \lambda & i \neq j \end{cases}$$

A covariance matrix $\mathbf{\Sigma}$ generally satisfies the Huynh-Feldt condition if it can be written as $\mathbf{\Sigma} = \boldsymbol{\tau}\mathbf{1}' + \mathbf{1}\boldsymbol{\tau}' + \lambda\mathbf{I}$. The preceding parameterization chooses $\tau_i = 0.5(\sigma_i^2 - \lambda)$. Several simpler covariance structures give rise to covariance matrices that also satisfy the Huynh-Feldt condition. For example, TYPE=CS, TYPE=VC, and TYPE=UN(1) are nested within TYPE=HF. You can use the COVTEST statement to test the HF structure against one of these simpler structures. Note also that the HF structure is nested within an unstructured covariance matrix.

The TYPE=HF covariance structure can be sensitive to the choice of starting values and the default MIVQUE(0) starting values can be poor for this structure; you can supply your own starting values with the PARMS statement.

LIN(*q*)

specifies a general linear covariance structure with q parameters. This structure consists of a linear combination of known matrices that you input with the LDATA= option. Suppose that you want to model the covariance of a random vector of length t, and further suppose that $\mathbf{A}_1, \ldots, \mathbf{A}_q$ are symmetric $(t \times t)$ matrices constructed from the information in the LDATA= data set. Then,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sum_{k=1}^{q} \theta_k [\mathbf{A}_k]_{ij}$$

where $[\mathbf{A}_k]_{ij}$ denotes the element in row i, column j of matrix $\mathbf{A}_k$.

Linear structures are very flexible and general. You need to exercise caution to ensure that the variance matrix is positive definite. Note that PROC GLIMMIX does not impose boundary constraints on the parameters $\theta_1, \ldots, \theta_k$ of a general linear covariance structure. For example, if classification variable A has 6 levels, the following statements fit a variance component structure for the random effect without boundary constraints:

```
data ldata;
   retain parm 1 value 1;
   do row=1 to 6; col=row; output; end;
run;

proc glimmix data=MyData;
   class A B;
   model Y = B;
   random A / type=lin(1) ldata=ldata;
run;
```

PSPLINE< (*options*) >

requests that PROC GLIMMIX form a B-spline basis and fits a penalized B-spline (P-spline, Eilers and Marx 1996) with random spline coefficients. This covariance structure is available only for G-side random effects and only a single continuous random effect can be specified with TYPE=PSPLINE. As for TYPE=RSMOOTH, PROC GLIMMIX forms a modified $\mathbf{Z}$ matrix and fits a mixed model in which the random variables associated with the columns of $\mathbf{Z}$ are independent with a common variance. The $\mathbf{Z}$ matrix is constructed as follows.

Denote as $\widetilde{\mathbf{Z}}$ the $(n \times K)$ matrix of B-splines of degree d and denote as $\mathbf{D}_r$ the $(K - r \times K)$ matrix of rth-order differences. For example, for $K = 5$,

$$\mathbf{D}_1 = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

$$\mathbf{D}_2 = \begin{bmatrix} 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \end{bmatrix}$$

$$\mathbf{D}_3 = \begin{bmatrix} 1 & -3 & 3 & -1 & 0 \\ 0 & 1 & -3 & 3 & -1 \end{bmatrix}$$

Then, the $\mathbf{Z}$ matrix used in fitting the mixed model is the $(n \times K - r)$ matrix

$$\mathbf{Z} = \widetilde{\mathbf{Z}}(\mathbf{D}_r'\mathbf{D}_r)^{-}\mathbf{D}_r'$$

The construction of the B-spline knots is controlled with the KNOTMETHOD= EQUAL(m) option and the DEGREE=d suboption of TYPE=PSPLINE. The total number of knots equals the number m of equally spaced interior knots plus d knots at the low end and $\max\{1, d\}$ knots at the high end. The number of columns in the B-spline basis equals $K = m + d + 1$. By default, the interior knots exclude the minimum and maximum of the random-effect values and are based on $m - 1$ equally spaced intervals. Suppose $x_{(1)}$ and $x_{(n)}$ are the smallest and largest random-effect values; then interior knots are placed at

$$x_{(1)} + j(x_{(n)} - x_{(1)})/(m + 1), \quad j = 1, \ldots, m$$

In addition, d evenly spaced exterior knots are placed below $x_{(1)}$ and $\max\{d, 1\}$ exterior knots are placed above $x_{(m)}$. The exterior knots are evenly spaced and start at $x_{(1)} \pm 100$ times the machine epsilon. For example, based on the defaults $d = 3$, $r = 3$, the following statements lead to 26 total knots and 21 columns in $\mathbf{Z}$, $m = 20$, $K = m + d + 1 = 24$, $K - r = 21$:

```
proc glimmix;
   model y = x;
   random x / type=pspline knotmethod=equal(20);
run;
```

Details about the computation and properties of B-splines can be found in De Boor (2001). You can extend or limit the range of the knots with the KNOTMIN= and KNOTMAX= options. Table 52.20 lists some of the parameters that control this covariance type and their relationships.

Table 52.20 P-Spline Parameters

Parameter	Description
d	Degree of B-spline, default $d = 3$
r	Order of differencing in construction of $\mathbf{D}_r$, default $r = 3$
m	Number of interior knots, default $m = 10$
$m + d + \max\{1, d\}$	Total number of knots
$K = m + d + 1$	Number of columns in B-spline basis
$K - r$	Number of columns in $\mathbf{Z}$

You can specify the following *options* for TYPE=PSPLINE:

DEGREE=*d* specifies the degree of the B-spline. The default is $d = 3$.

DIFFORDER=*r* specifies the order of the differencing matrix $\mathbf{D}_r$. The default and maximum is $r = 3$.

RSMOOTH<(*m* | **NOLOG**)>

specifies a radial smoother covariance structure for G-side random effects. This results in an approximate low-rank thin-plate spline where the smoothing parameter is obtained by the estimation method selected with the METHOD= option of the PROC GLIMMIX statement. The smoother is based on the automatic smoother in Ruppert, Wand, and Carroll (2003, Chapter 13.4–13.5), but with a different method of selecting the spline knots. See the section "Radial Smoothing Based on Mixed Models" on page 3959 for further details about the construction of the smoother and the knot selection.

Radial smoothing is possible in one or more dimensions. A univariate smoother is obtained with a single random effect, while multiple random effects in a RANDOM statement yield a multivariate smoother. Only continuous random effects are permitted with this covariance structure. If n_r denotes the number of continuous random effects in the RANDOM statement, then the covariance structure of the random effects $\boldsymbol{\gamma}$ is determined as follows. Suppose that $\mathbf{z}_i$ denotes the vector of random effects for the ith observation. Let $\boldsymbol{\tau}_k$ denote the $(n_r \times 1)$ vector of knot coordinates, $k = 1, \ldots, K$, and K is the total number of knots. The Euclidean distance between the knots is computed as

$$d_{kp} = ||\boldsymbol{\tau}_k - \boldsymbol{\tau}_p|| = \sqrt{\sum_{j=1}^{n_r} (\tau_{jk} - \tau_{jp})^2}$$

and the distance between knots and effects is computed as

$$h_{ik} = ||\mathbf{z}_i - \boldsymbol{\tau}_k|| = \sqrt{\sum_{j=1}^{n_r}(z_{ij} - \tau_{jk})^2}$$

The $\mathbf{Z}$ matrix for the GLMM is constructed as

$$\mathbf{Z} = \widetilde{\mathbf{Z}}\boldsymbol{\Omega}^{-1/2}$$

where the $(n \times K)$ matrix $\widetilde{\mathbf{Z}}$ has typical element

$$[\widetilde{\mathbf{Z}}]_{ik} = \begin{cases} h_{ik}^p & n_r \text{ odd} \\ h_{ik}^p \log\{h_{ik}\} & n_r \text{ even} \end{cases}$$

and the $(K \times K)$ matrix $\boldsymbol{\Omega}$ has typical element

$$[\boldsymbol{\Omega}]_{kp} = \begin{cases} d_{kp}^p & n_r \text{ odd} \\ d_{kp}^p \log\{d_{kp}\} & n_r \text{ even} \end{cases}$$

The exponent in these expressions equals $p = 2m - n_r$, where the optional value m corresponds to the derivative penalized in the thin-plate spline. A larger value of m will yield a smoother fit. The GLIMMIX procedure requires $p > 0$ and chooses by default $m = 2$ if $n_r < 3$ and $m = n_r/2 + 1$ otherwise. The NOLOG option removes the $\log\{h_{ik}\}$ and $\log\{d_{kp}\}$ terms from the computation of the $\widetilde{\mathbf{Z}}$ and $\boldsymbol{\Omega}$ matrices when n_r is even; this yields invariance under rescaling of the coordinates.

Finally, the components of $\boldsymbol{\gamma}$ are assumed to have equal variance σ_r^2. The "smoothing parameter" λ of the low-rank spline is related to the variance components in the model, $\lambda^2 = f(\phi, \sigma_r^2)$. See Ruppert, Wand, and Carroll (2003) for details. If the conditional distribution does not provide a scale parameter ϕ, you can add a single R-side residual parameter.

The knot selection is controlled with the KNOTMETHOD= option. The GLIMMIX procedure selects knots automatically based on the vertices of a k-d tree or reads knots from a data set that you supply. See the section "Radial Smoothing Based on Mixed Models" on page 3959 for further details on radial smoothing in the GLIMMIX procedure and its connection to a mixed model formulation.

SIMPLE

is an alias for TYPE=VC.

SP(EXP)(*c-list*)

models an exponential spatial or temporal covariance structure, where the covariance between two observations depends on a distance metric d_{ij}. The *c-list* contains the names of the numeric variables used as coordinates to determine distance. For a stochastic process in R^k, there are k elements in *c-list*. If the $(k \times 1)$ vectors of coordinates for observations i and j are $\mathbf{c}_i$ and $\mathbf{c}_j$, then PROC GLIMMIX computes the Euclidean distance

$$d_{ij} = ||\mathbf{c}_i - \mathbf{c}_j|| = \sqrt{\sum_{m=1}^{k}(c_{mi} - c_{mj})^2}$$

The covariance between two observations is then

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \exp\{-d_{ij}/\alpha\}$$

The parameter α is *not* what is commonly referred to as the range parameter in geostatistical applications. The practical range of a (second-order stationary) spatial process is the distance $d^{(p)}$ at which the correlations fall below 0.05. For the SP(EXP) structure, this distance is $d^{(p)} = 3\alpha$. PROC GLIMMIX constrains α to be positive.

SP(GAU)(*c-list*)

models a Gaussian covariance structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \exp\{-d_{ij}^2/\alpha^2\}$$

See TYPE=TYPE=SP(EXP) for the computation of the distance d_{ij}. The parameter α is related to the range of the process as follows. If the practical range $d^{(p)}$ is defined as the distance at which the correlations fall below 0.05, then $d^{(p)} = \sqrt{3}\alpha$. PROC GLIMMIX constrains α to be positive. See TYPE=SP(EXP) for the computation of the distance d_{ij} from the variables specified in *c-list*.

SP(LEAR)(*c-list*)

models a linear exponent autoregressive covariance structure as proposed by Simpson et al. (2010). For two observations with distance metric d_{ij}, the covariance is

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \begin{cases} \rho^{d_{\min}+\delta[(d_{ij}-d_{\min})/(d_{\max}-d_{\min})]} & i \neq j \\ 1 & i = j \end{cases}$$

where $d_{\min}$ and $d_{\max}$ are the smallest and largest distances between any two observations, $\delta \geq 0$ is the decay speed, and $0 \leq \rho < 1$. See TYPE=SP(EXP) for the computation of the distance d_{ij} from the variables specified in *c-list*. When the estimated value of ρ becomes negative, PROC GLIMMIX multiplies the computed covariance by $\cos(\pi d_{ij})$ to account for the negativity. When $d_{\min} = d_{\max}$, PROC GLIMMIX sets the computed covariance to $\sigma^2\rho^{d_{\min}}$. Note that GROUP= effect is not supported for TYPE=SP(LEAR).

For power analysis of repeated measures designs that have a LEAR correlation structure, see the section "POWER Statement" on page 4370 in Chapter 55, "The GLMPOWER Procedure."

SP(MAT)(*c-list*)

models a covariance structure in the Matérn class of covariance functions (Matérn 1986). The covariance is expressed in the parameterization of Handcock and Stein (1993); Handcock and Wallis (1994); it can be written as

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \frac{1}{\Gamma(\nu)} \left(\frac{d_{ij}\sqrt{\nu}}{\rho}\right)^{\nu} 2K_\nu \left(\frac{2d_{ij}\sqrt{\nu}}{\rho}\right)$$

The function K_ν is the modified Bessel function of the second kind of (real) order $\nu > 0$. The smoothness (continuity) of a stochastic process with covariance function in the Matérn class increases with ν. This class thus enables data-driven estimation of the smoothness properties of the process. The covariance is identical to the exponential model for $\nu = 0.5$ (TYPE=SP(EXP)(*c-list*)), while for $\nu = 1$ the model advocated by Whittle (1954) results. As $\nu \to \infty$, the model approaches the Gaussian covariance structure (TYPE=SP(GAU)(*c-list*)).

Note that the MIXED procedure offers covariance structures in the Matérn class in two parameterizations, TYPE=SP(MATERN) and TYPE=SP(MATHSW). The TYPE=SP(MAT) in the GLIMMIX procedure is equivalent to TYPE=SP(MATHSW) in the MIXED procedure.

Computation of the function K_ν and its derivatives is numerically demanding; fitting models with Matérn covariance structures can be time-consuming. Good starting values are essential.

SP(POW)(*c-list*)

models a power covariance structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \rho^{d_{ij}}$$

where $\rho \geq 0$. This is a reparameterization of the exponential structure, TYPE=SP(EXP). Specifically, $\log\{\rho\} = -1/\alpha$. See TYPE=SP(EXP) for the computation of the distance d_{ij} from the variables specified in *c-list*. When the estimated value of ρ becomes negative, the computed covariance is multiplied by $\cos(\pi d_{ij})$ to account for the negativity.

SP(POWA)(*c-list*)

models an anisotropic power covariance structure in k dimensions, provided that the coordinate list *c-list* has k elements. If c_{im} denotes the coordinate for the ith observation of the mth variable in *c-list*, the covariance between two observations is given by

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \rho_1^{|c_{i1}-c_{j1}|} \rho_2^{|c_{i2}-c_{j2}|} \ldots \rho_k^{|c_{ik}-c_{jk}|}$$

Note that for $k = 1$, TYPE=SP(POWA) is equivalent to TYPE=SP(POW), which is itself a reparameterization of TYPE=SP(EXP). When the estimated value of ρ_m becomes negative, the computed covariance is multiplied by $\cos(\pi|c_{im} - c_{jm}|)$ to account for the negativity.

SP(SPH)(*c-list*)

models a spherical covariance structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 \left\{1 - \frac{3d_{ij}}{2\alpha} + \frac{1}{2}\left(\frac{d_{ij}}{\alpha}\right)^3\right\} & d_{ij} \leq \alpha \\ 0 & d_{ij} > \alpha \end{cases}$$

The spherical covariance structure has a true range parameter. The covariances between observations are exactly zero when their distance exceeds α. See TYPE=SP(EXP) for the computation of the distance d_{ij} from the variables specified in *c-list*.

TOEP

models a Toeplitz covariance structure. This structure can be viewed as an autoregressive structure with order equal to the dimension of the matrix,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 & i = j \\ \sigma_{|i-j|} & i \neq j \end{cases}$$

TOEP(*q*)

specifies a banded Toeplitz structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 & i = j \\ \sigma_{|i-j|} & |i-j| < q \end{cases}$$

This can be viewed as a moving-average structure with order equal to $q - 1$. The specification TYPE=TOEP(1) is the same as $\sigma^2\mathbf{I}$, and it can be useful for specifying the same variance component for several effects.

TOEPH<(*q*)>

models a Toeplitz covariance structure. The correlations of this structure are banded as the TOEP or TOEP(q) structures, but the variances are allowed to vary:

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma_i^2 & i = j \\ \rho_{|i-j|}\sqrt{\sigma_i^2\sigma_j^2} & i \neq j \end{cases}$$

The correlation parameters satisfy $|\rho_{|i-j|}| < 1$. If you specify the optional value q, the correlation parameters with $|i - j| \geq q$ are set to zero, creating a banded correlation structure. The specification TYPE=TOEPH(1) results in a diagonal covariance matrix with heterogeneous variances.

UN<(*q*)>

specifies a completely general (unstructured) covariance matrix parameterized directly in terms of variances and covariances,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma_{ij}$$

The variances are constrained to be nonnegative, and the covariances are unconstrained. This structure is not constrained to be nonnegative definite in order to avoid nonlinear constraints; however, you can use the TYPE=CHOL structure if you want this constraint to be imposed by a Cholesky factorization. If you specify the order parameter q, then PROC GLIMMIX estimates only the first q bands of the matrix, setting elements in all higher bands equal to 0.

UNR<(*q*)>

specifies a completely general (unstructured) covariance matrix parameterized in terms of variances and correlations,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma_i\sigma_j\rho_{ij}$$

where σ_i denotes the standard deviation and the correlation ρ_{ij} is zero when $i = j$ and when $|i - j| \geq q$, provided the order parameter q is given. This structure fits the same model as the TYPE=UN(q) option, but with a different parameterization. The ith variance parameter is σ_i^2. The parameter ρ_{ij} is the correlation between the ith and jth measurements; it satisfies $|\rho_{ij}| < 1$. If you specify the order parameter q, then PROC GLIMMIX estimates only the first q bands of the matrix, setting all higher bands equal to zero.

VC

specifies standard variance components and is the default structure for both G-side and R-side covariance structures. In a G-side covariance structure, a distinct variance component is assigned to each effect. In an R-side structure TYPE=VC is usually used only to add overdispersion effects or with the GROUP= option to specify a heterogeneous variance model.

Table 52.21 Covariance Structure Examples

Description	Structure	Example
Variance Components	VC (default)	$\begin{bmatrix} \sigma_B^2 & 0 & 0 & 0 \\ 0 & \sigma_B^2 & 0 & 0 \\ 0 & 0 & \sigma_{AB}^2 & 0 \\ 0 & 0 & 0 & \sigma_{AB}^2 \end{bmatrix}$
Compound Symmetry	CS	$\begin{bmatrix} \sigma+\phi & \sigma & \sigma & \sigma \\ \sigma & \sigma+\phi & \sigma & \sigma \\ \sigma & \sigma & \sigma+\phi & \sigma \\ \sigma & \sigma & \sigma & \sigma+\phi \end{bmatrix}$
Heterogeneous CS	CSH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho & \sigma_1\sigma_4\rho \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho \\ \sigma_3\sigma_1\rho & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho & \sigma_4\sigma_2\rho & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$
First-Order Autoregressive	AR(1)	$\sigma^2 \begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 \\ \rho & 1 & \rho & \rho^2 \\ \rho^2 & \rho & 1 & \rho \\ \rho^3 & \rho^2 & \rho & 1 \end{bmatrix}$
Heterogeneous AR(1)	ARH(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho^2 & \sigma_1\sigma_4\rho^3 \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho^2 \\ \sigma_3\sigma_1\rho^2 & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho^3 & \sigma_4\sigma_2\rho & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$
Unstructured	UN	$\begin{bmatrix} \sigma_1^2 & \sigma_{21} & \sigma_{31} & \sigma_{41} \\ \sigma_{21} & \sigma_2^2 & \sigma_{32} & \sigma_{42} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 & \sigma_{43} \\ \sigma_{41} & \sigma_{42} & \sigma_{43} & \sigma_4^2 \end{bmatrix}$
Banded Main Diagonal	UN(1)	$\begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & 0 \\ 0 & 0 & 0 & \sigma_4^2 \end{bmatrix}$
Unstructured Correlations	UNR	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_{21} & \sigma_1\sigma_3\rho_{31} & \sigma_1\sigma_4\rho_{41} \\ \sigma_2\sigma_1\rho_{21} & \sigma_2^2 & \sigma_2\sigma_3\rho_{32} & \sigma_2\sigma_4\rho_{42} \\ \sigma_3\sigma_1\rho_{31} & \sigma_3\sigma_2\rho_{32} & \sigma_3^2 & \sigma_3\sigma_4\rho_{43} \\ \sigma_4\sigma_1\rho_{41} & \sigma_4\sigma_2\rho_{42} & \sigma_4\sigma_3\rho_{43} & \sigma_4^2 \end{bmatrix}$

Table 52.21 *continued*

Description	Structure	Example
Toeplitz	TOEP	$\begin{bmatrix} \sigma^2 & \sigma_1 & \sigma_2 & \sigma_3 \\ \sigma_1 & \sigma^2 & \sigma_1 & \sigma_2 \\ \sigma_2 & \sigma_1 & \sigma^2 & \sigma_1 \\ \sigma_3 & \sigma_2 & \sigma_1 & \sigma^2 \end{bmatrix}$
Toeplitz with Two Bands	TOEP(2)	$\begin{bmatrix} \sigma^2 & \sigma_1 & 0 & 0 \\ \sigma_1 & \sigma^2 & \sigma_1 & 0 \\ 0 & \sigma_1 & \sigma^2 & \sigma_1 \\ 0 & 0 & \sigma_1 & \sigma^2 \end{bmatrix}$
Heterogeneous Toeplitz	TOEPH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_1 & \sigma_1\sigma_3\rho_2 & \sigma_1\sigma_4\rho_3 \\ \sigma_2\sigma_1\rho_1 & \sigma_2^2 & \sigma_2\sigma_3\rho_1 & \sigma_2\sigma_4\rho_2 \\ \sigma_3\sigma_1\rho_2 & \sigma_3\sigma_2\rho_1 & \sigma_3^2 & \sigma_3\sigma_4\rho_1 \\ \sigma_4\sigma_1\rho_3 & \sigma_4\sigma_2\rho_2 & \sigma_4\sigma_3\rho_1 & \sigma_4^2 \end{bmatrix}$
Spatial Power	SP(POW)(*c-list*)	$\sigma^2 \begin{bmatrix} 1 & \rho^{d_{12}} & \rho^{d_{13}} & \rho^{d_{14}} \\ \rho^{d_{21}} & 1 & \rho^{d_{23}} & \rho^{d_{24}} \\ \rho^{d_{31}} & \rho^{d_{32}} & 1 & \rho^{d_{34}} \\ \rho^{d_{41}} & \rho^{d_{42}} & \rho^{d_{43}} & 1 \end{bmatrix}$
First-Order Autoregressive Moving-Average	ARMA(1,1)	$\sigma^2 \begin{bmatrix} 1 & \gamma & \gamma\rho & \gamma\rho^2 \\ \gamma & 1 & \gamma & \gamma\rho \\ \gamma\rho & \gamma & 1 & \gamma \\ \gamma\rho^2 & \gamma\rho & \gamma & 1 \end{bmatrix}$
First-Order Factor Analytic	FA(1)	$\begin{bmatrix} \lambda_1^2 + d_1 & \lambda_1\lambda_2 & \lambda_1\lambda_3 & \lambda_1\lambda_4 \\ \lambda_2\lambda_1 & \lambda_2^2 + d_2 & \lambda_2\lambda_3 & \lambda_2\lambda_4 \\ \lambda_3\lambda_1 & \lambda_3\lambda_2 & \lambda_3^2 + d_3 & \lambda_3\lambda_4 \\ \lambda_4\lambda_1 & \lambda_4\lambda_2 & \lambda_4\lambda_3 & \lambda_4^2 + d_4 \end{bmatrix}$
Huynh-Feldt	HF	$\begin{bmatrix} \sigma_1^2 & \frac{\sigma_1^2+\sigma_2^2}{2} - \lambda & \frac{\sigma_1^2+\sigma_3^2}{2} - \lambda \\ \frac{\sigma_2^2+\sigma_1^2}{2} - \lambda & \sigma_2^2 & \frac{\sigma_2^2+\sigma_3^2}{2} - \lambda \\ \frac{\sigma_3^2+\sigma_1^2}{2} - \lambda & \frac{\sigma_3^2+\sigma_2^2}{2} - \lambda & \sigma_3^2 \end{bmatrix}$
First-Order Ante-dependence	ANTE(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_1 & \sigma_1\sigma_3\rho_1\rho_2 \\ \sigma_2\sigma_1\rho_1 & \sigma_2^2 & \sigma_2\sigma_3\rho_2 \\ \sigma_3\sigma_1\rho_2\rho_1 & \sigma_3\sigma_2\rho_2 & \sigma_3^2 \end{bmatrix}$

V< =*value-list* **>**

requests that blocks of the estimated marginal variance-covariance matrix $\mathbf{V}(\widehat{\boldsymbol{\theta}})$ be displayed in generalized linear mixed models. This matrix is based on the last linearization as described in the section "The Pseudo-model" on page 3924. You can use the *value-list* to select the subjects for which the matrix is displayed. If *value-list* is not specified, the **V** matrix for the first subject is chosen.

Note that the *value-list* refers to subjects as the processing units in the "Dimensions" table. For example, the following statements request that the estimated marginal variance matrix for the second subject be

displayed:

```
proc glimmix;
   class A B;
   model y = B;
   random int / subject=A;
   random int / subject=A*B v=2;
run;
```

The subject effect for processing in this case is the A effect, because it is contained in the A*B interaction. If there is only a single subject as per the "Dimensions" table, then the V option displays an $(n \times n)$ matrix.

See the section "Processing by Subjects" on page 3957 for how the GLIMMIX procedure determines the number of subjects in the "Dimensions" table.

The GLIMMIX procedure displays blanks for values that are 0.

VC<=*value-list*>
displays the lower-triangular Cholesky root of the blocks of the estimated $\mathbf{V}(\widehat{\boldsymbol{\theta}})$ matrix. See the V option for the specification of *value-list*.

VCI<=*value-list*>
displays the inverse Cholesky root of the blocks of the estimated $\mathbf{V}(\widehat{\boldsymbol{\theta}})$ matrix. See the V option for the specification of *value-list*.

VCORR<=*value-list*>
displays the correlation matrix corresponding to the blocks of the estimated $\mathbf{V}(\widehat{\boldsymbol{\theta}})$ matrix. See the V option for the specification of *value-list*.

VI<=*value-list*>
displays the inverse of the blocks of the estimated $\mathbf{V}(\widehat{\boldsymbol{\theta}})$ matrix. See the V option for the specification of *value-list*.

WEIGHT<=*variable*>

WT<=*variable*>
specifies a variable to be used as the weight for the units at the current level in a weighted multilevel model. If a weight variable is not specified in the WEIGHT option, a weight of 1 is used. For details on the use of weights in multilevel models, see the section "Pseudo-likelihood Estimation for Weighted Multilevel Models" on page 3938.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects (Winer 1971).

The SLICE statement uses most of the options of the LSMEANS statement that are summarized in Table 52.9. The options SLICEDIFF=, SLICEDIFFTYPE=, and ODDS do not apply to the SLICE statement; in the SLICE statement, the relevant options for SLICEDIFF= and SLICEDIFFTYPE= are the SLICEBY= and the DIFF= options, respectively.

For details about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement replaces $\mathbf{R}$ with $\mathbf{W}^{-1/2}\mathbf{R}\mathbf{W}^{-1/2}$, where $\mathbf{W}$ is a diagonal matrix containing the weights. Observations with nonpositive or missing weights are not included in the resulting PROC GLIMMIX analysis. If a WEIGHT statement is not included, all observations used in the analysis are assigned a weight of 1.

Programming Statements

This section lists the programming statements available in PROC GLIMMIX to compute various aspects of the generalized linear mixed model or output quantities. For example, you can compute model effects, weights, frequency, subject, group, and other variables. You can use programming statements to define the mean and variance functions. This section also documents the differences between programming statements in PROC GLIMMIX and programming statements in the SAS DATA step. The syntax of programming statements used in PROC GLIMMIX is identical to that used in the NLMIXED procedure (see Chapter 89, "The NLMIXED Procedure," and the MODEL procedure (see the *SAS/ETS User's Guide*). Most of the programming statements that can be used in the DATA step can also be used in the GLIMMIX procedure. See *SAS DATA Step Statements: Reference* for a description of SAS programming statements. The following are valid statements:

ABORT;
ARRAY *arrayname* < **[** *dimensions* **]** > < **$** > < *variables-and-constants* >**;**
CALL *name* < **(***expression* < **,** *expression* . . . >**)** >**;**
DELETE;
DO < *variable* **=** *expression* < **TO** *expression* > < **BY** *expression* > >
< **,** *expression* < **TO** *expression* > < **BY** *expression* > > . . .
< **WHILE** *expression* > < **UNTIL** *expression* >**;**
END;
GOTO *statement-label***;**
IF *expression***;**
IF *expression* **THEN** *program-statement***;**
ELSE *program-statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement-label***;**
PUT < *variable* > < **=** > . . . **;**
RETURN;
SELECT < **(***expression***)** >**;**
STOP;
SUBSTR(*variable***,** *index***,** *length***)=** *expression***;**
WHEN (*expression***)***program-statement***;**
OTHERWISE *program-statement***;**

For the most part, the SAS programming statements work the same as they do in the SAS DATA step, as documented in *SAS Programmers Guide: Essentials*. However, there are several differences:

- The ABORT statement does not allow any arguments.
- The DO statement does not allow a character index variable. Thus

  ```
  do i = 1,2,3;
  ```

  is supported; however, the following statement is not supported:

  ```
  do i = 'A','B','C';
  ```

- The LAG function is not supported with PROC GLIMMIX.
- The PUT statement, used mostly for program debugging in PROC GLIMMIX, supports only some of the features of the DATA step PUT statement, and it has some features not available with the DATA step PUT statement:
  - The PROC GLIMMIX PUT statement does not support line pointers, factored lists, iteration factors, overprinting, _INFILE_, the colon (:) format modifier, or “$”.
  - The PROC GLIMMIX PUT statement does support expressions, but the expression must be enclosed in parentheses. For example, the following statement displays the square root of x:

```
put (sqrt(x));
```

– The PROC GLIMMIX PUT statement supports the item _PDV_ to display a formatted listing of all variables in the program. For example:

```
put _pdv_;
```

- The WHEN and OTHERWISE statements enable you to specify more than one target statement. That is, DO/END groups are not necessary for multiple statement WHENs. For example, the following syntax is valid:

```
select;
   when (exp1) stmt1;
               stmt2;
   when (exp2) stmt3;
               stmt4;
end;
```

The LINK statement is used in a program to jump immediately to the label *statement_label* and to continue program execution at that point. It is not used to specify a user-defined link function.

When coding your programming statements, you should avoid defining variables that begin with an underscore (_), because they might conflict with internal variables created by PROC GLIMMIX.

User-Defined Link or Variance Function

Implied Variance Functions

While link functions are not unique for each distribution (see Table 52.15 for the default link functions), the distribution does determine the variance function $a(\mu)$. This function expresses the variance of an observation as a function of the mean, apart from weights, frequencies, and additional scale parameters. The implied variance functions $a(\mu)$ of the GLIMMIX procedure are shown in Table 52.22 for the supported distributions. For the binomial distribution, n denotes the number of trials in the *events/trials* syntax. For the negative binomial distribution, k denotes the scale parameter. The multiplicative scale parameter ϕ is not included for the other distributions. The last column of the table indicates whether ϕ has a value equal to 1.0 for the particular distribution.

Table 52.22 Variance Functions in PROC GLIMMIX

DIST=	Distribution	Variance Function $a(\mu)$	$\phi \equiv 1$
BETA	Beta	$\mu(1-\mu)/(1+\phi)$	No
BINARY	Binary	$\mu(1-\mu)$	Yes
BINOMIAL | BIN | B	Binomial	$\mu(1-\mu)/n$	Yes
EXPONENTIAL | EXPO	Exponential	μ^2	Yes
GAMMA | GAM	Gamma	μ^2	No

Table 52.22 *continued*

DIST=	Distribution	Variance Function $a(\mu)$	$\phi \equiv 1$
GAUSSIAN | G | **NORMAL** | **N**	Normal	1	No
GEOMETRIC | **GEOM**	Geometric	$\mu + \mu^2$	Yes
INVGAUSS | **IGAUSSIAN** | **IG**	Inverse Gaussian	μ^3	No
LOGNORMAL | **LOGN**	Lognormal	1	No
NEGBINOMIAL | **NEGBIN** | **NB**	Negative binomial	$\mu + k\mu^2$	Yes
POISSON | **POI** | **P**	Poisson	μ	Yes
TCENTRAL | **TDIST** | **T**	t	$\nu/(\nu - 2)$	No

To change the variance function, you can use SAS programming statements and the predefined automatic variables, as outlined in the following section. Your definition of a variance function will override the DIST= option and its implied variance function. This has the following implication for parameter estimation with the GLIMMIX procedure. When a user-defined link is available, the distribution of the data is determined from the DIST= option, or the respective default for the type of response. In a GLM, for example, this enables maximum likelihood estimation. If a user-defined variance function is provided, the DIST= option is not honored and the distribution of the data is assumed unknown. In a GLM framework, only quasi-likelihood estimation is then available to estimate the model parameters.

Automatic Variables

To specify your own link or variance function you can use SAS programming statements and draw on the following automatic variables:

LINP is the current value of the linear predictor. It equals either $\widehat{\eta} = \mathbf{x}'\widehat{\boldsymbol{\beta}} + \mathbf{z}'\widehat{\boldsymbol{\gamma}} + o$ or $\widehat{\eta} = \mathbf{x}'\widehat{\boldsymbol{\beta}} + o$, where o is the value of the offset variable, or 0 if no offset is specified. The estimated random effects solutions $\widehat{\boldsymbol{\gamma}}$ are used in the calculation of the linear predictor during the model fitting phase, if a linearization expands about the current values of $\boldsymbol{\gamma}$. During the computation of output statistics, the EBLUPs are used if statistics depend on them. For example, the following statements add the variable p to the output data set glimmixout:

```
proc glimmix;
   model y = x / dist=binary;
   random int / subject=b;
   p = 1/(1+exp(-_linp_);
   output out=glimmixout;
   id p;
run;
```

Because no output statistics are requested in the OUTPUT statement that depend on the random-effects solutions (BLUPs, EBEs), the value of _LINP_ in this example equals $\mathbf{x}'\widehat{\boldsymbol{\beta}}$. On the contrary, the following statements also request conditional residuals on the logistic scale:

```
proc glimmix;
   model y = x / dist=binary;
   random int / subject=b;
   p = 1/(1+exp(-_linp_);
   output out=glimmixout resid(blup)=r;
   id p;
run;
```

The value of _LINP_ when computing the variable p is $\mathbf{x}'\widehat{\boldsymbol{\beta}}+\mathbf{z}'\widehat{\boldsymbol{\gamma}}$. To ensure that computed statistics are formed from $\mathbf{x}'\widehat{\boldsymbol{\beta}}$ and $\mathbf{z}'\widehat{\boldsymbol{\gamma}}$ terms as needed, it is recommended that you use the automatic variables _XBETA_ and _ZGAMMA_ instead of _LINP_.

_MU_ expresses the mean of an observation as a function of the linear predictor, $\widehat{\mu} = g^{-1}(\widehat{\eta})$.

_N_ is the observation number in the sequence of the data read.

VARIANCE is the estimate of the variance function, $a(\widehat{\mu})$.

XBETA equals $\mathbf{x}'\widehat{\boldsymbol{\beta}}$.

ZGAMMA equals $\mathbf{z}'\widehat{\boldsymbol{\gamma}}$.

The automatic variable _N_ is incremented whenever the procedure reads an observation from the data set. Observations that are not used in the analysis—for example, because of missing values or invalid weights—are counted. The counter is reset to 1 at the start of every new BY group. Only in some circumstances will _N_ equal the actual observation number. The symbol should thus be used sparingly to avoid unexpected results.

You must observe the following syntax rules when you use the automatic variables. The _LINP_ symbol cannot appear on the left side of programming statements; you cannot make an assignment to the _LINP_ variable. The value of the linear predictor is controlled by the CLASS, MODEL, and RANDOM statements as well as the current parameter estimates and solutions. You can, however, use the _LINP_ variable on the right side of other operations. Suppose, for example, that you want to transform the linear predictor prior to applying the inverse log link. The following statements are not valid because the linear predictor appears in an assignment:

```
proc glimmix;
   _linp_ = sqrt(abs(_linp_));
   _mu_   = exp(_linp_);
   model count = logtstd / dist=poisson;
run;
```

The next statements achieve the desired result:

```
proc glimmix;
   _mu_ = exp(sqrt(abs(_linp_)));
   model count = logtstd / dist=poisson;
run;
```

If the value of the linear predictor is altered in any way through programming statements, you need to ensure that an assignment to _MU_ follows. The assignment to variable P in the next set of statements is without effect:

```
proc glimmix;
   p = _linp_ + rannor(454);
   model count = logtstd / dist=poisson;
run;
```

A user-defined link function is implied by expressing _MU_ as a function of _LINP_. That is, if $\mu = g^{-1}(\eta)$, you are providing an expression for the inverse link function with programming statements. It is neither necessary nor possible to give an expression for the inverse operation, $\eta = g(\mu)$. The variance function is determined by expressing _VARIANCE_ as a function of _MU_. If the _MU_ variable appears in an assignment statement inside PROC GLIMMIX, the LINK= option of the MODEL statement is ignored. If the _VARIANCE_ function appears in an assignment statement, the DIST= option is ignored. Furthermore, the associated variance function per Table 52.22 is not honored. In short, user-defined expressions take precedence over built-in defaults.

If you specify your own link and variance function, the assignment to _MU_ must precede an assignment to the variable _VARIANCE_.

The following two sets of statements yield the same parameter estimates, but the models differ statistically:

```
proc glimmix;
   class block entry;
   model y/n = block entry / dist=binomial link=logit;
run;

proc glimmix;
   class block entry;
   prob = 1 / (1+exp(- _linp_));
   _mu_ = n * prob ;
   _variance_ = n * prob *(1-prob);
   model y = block entry;
run;
```

The first invocation of PROC GLIMMIX models the proportion y/n as a binomial proportion with a logit link. The DIST= and LINK= options are superfluous in this case, because the GLIMMIX procedure defaults to the binomial distribution in light of the *events/trials* syntax. The logit link is that distribution's default link. The second invocation of PROC GLIMMIX models the count variable y and takes the binomial sample size into account through assignments to the mean and variance function. In contrast to the first invocation, the distribution of y is unknown. Only its mean and variance are known. The model parameters are estimated by maximum likelihood in the first case and by quasi-likelihood in the second case.

Details: GLIMMIX Procedure

Generalized Linear Models Theory

A generalized linear model consists of the following:

- a linear predictor $\eta = \mathbf{x}'\boldsymbol{\beta}$
- a monotonic mapping between the mean of the data and the linear predictor
- a response distribution in the exponential family of distributions

A density or mass function in this family can be written as

$$f(y) = \exp\left\{\frac{y\theta - b(\theta)}{\phi} + c(y, f(\phi))\right\}$$

for some functions $b(\cdot)$ and $c(\cdot)$. The parameter θ is called the natural (canonical) parameter. The parameter ϕ is a scale parameter, and it is not present in all exponential family distributions. See Table 52.22 for a list of distributions for which $\phi \equiv 1$. In the case where observations are weighted, the scale parameter is replaced with ϕ/w in the preceding density (or mass function), where w is the weight associated with the observation y.

The mean and variance of the data are related to the components of the density, $\mathrm{E}[Y] = \mu = b'(\theta)$, $\mathrm{Var}[Y] = \phi b''(\theta)$, where primes denote first and second derivatives. If you express θ as a function of μ, the relationship is known as the natural link or the canonical link function. In other words, modeling data with a canonical link assumes that $\theta = \mathbf{x}'\boldsymbol{\beta}$; the effect contributions are additive on the canonical scale. The second derivative of $b(\cdot)$, expressed as a function of μ, is the variance function of the generalized linear model, $a(\mu) = b''(\theta(\mu))$. Note that because of this relationship, the distribution determines the variance function and the canonical link function. You cannot, however, proceed in the opposite direction. If you provide a user-specified variance function, the GLIMMIX procedure assumes that only the first two moments of the response distribution are known. The full distribution of the data is then unknown and maximum likelihood estimation is not possible. Instead, the GLIMMIX procedure then estimates parameters by quasi-likelihood.

Maximum Likelihood

The GLIMMIX procedure forms the log likelihoods of generalized linear models as

$$L(\boldsymbol{\mu}, \phi; \mathbf{y}) = \sum_{i=1}^{n} f_i \, l(\mu_i, \phi; y_i, w_i)$$

where $l(\mu_i, \phi; y_i, w_i)$ is the log likelihood contribution of the ith observation with weight w_i and f_i is the value of the frequency variable. For the determination of w_i and f_i, see the WEIGHT and FREQ statements. The individual log likelihood contributions for the various distributions are as follows.

Beta

$$
\begin{aligned}
l(\mu_i, \phi; y_i, w_i) = \log &\left\{ \frac{\Gamma(\phi/w_i)}{\Gamma(\mu_i\phi/w_i)\Gamma((1-\mu_i)\phi/w_i)} \right\} \\
&+ (\mu_i\phi/w_i - 1)\log\{y_i\} \\
&+ ((1-\mu_i)\phi/w_i - 1)\log\{1-y_i\}
\end{aligned}
$$

$\mathrm{Var}[Y] = \mu(1-\mu)/(1+\phi)$, $\phi > 0$. See Ferrari and Cribari-Neto (2004).

Binary

$$
l(\mu_i, \phi; y_i, w_i) = w_i(y_i \log\{\mu_i\} + (1-y_i)\log\{1-\mu_i\})
$$

$\mathrm{Var}[Y] = \mu(1-\mu)$, $\phi \equiv 1$.

Binomial

$$
\begin{aligned}
l(\mu_i, \phi; y_i, w_i) = &\ w_i\ (y_i \log\{\mu_i\} + (n_i - y_i)\log\{1-\mu_i\}) \\
&+ w_i\ (\log\{\Gamma(n_i+1)\} - \log\{\Gamma(y_i+1)\} - \log\{\Gamma(n_i - y_i + 1)\})
\end{aligned}
$$

where y_i and n_i are the *events* and *trials* in the *events/trials* syntax, and $0 < \mu < 1$. $\mathrm{Var}[Y/n] = \mu(1-\mu)/n$, $\phi \equiv 1$.

Exponential

$$
l(\mu_i, \phi; y_i, w_i) = \begin{cases} -\log\{\mu_i\} - y_i/\mu_i & w_i = 1 \\ w_i \log\left\{\frac{w_i y_i}{\mu_i}\right\} - \frac{w_i y_i}{\mu_i} - \log\{y_i \Gamma(w_i)\} & w_i \neq 1 \end{cases}
$$

$\mathrm{Var}[Y] = \mu^2$, $\phi \equiv 1$.

Gamma

$$
l(\mu_i, \phi; y_i, w_i) = w_i\phi \log\left\{\frac{w_i y_i \phi}{\mu_i}\right\} - \frac{w_i y_i \phi}{\mu_i} - \log\{y_i\} - \log\{\Gamma(w_i\phi)\}
$$

$\mathrm{Var}[Y] = \phi\mu^2$, $\phi > 0$.

Geometric

$$
\begin{aligned}
l(\mu_i, \phi; y_i, w_i) = &\ y_i \log\left\{\frac{\mu_i}{w_i}\right\} - (y_i + w_i)\log\left\{1 + \frac{\mu_i}{w_i}\right\} \\
&+ \log\left\{\frac{\Gamma(y_i + w_i)}{\Gamma(w_i)\Gamma(y_i+1)}\right\}
\end{aligned}
$$

$\mathrm{Var}[Y] = \mu + \mu^2$, $\phi \equiv 1$.

Inverse Gaussian

$$
l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{y_i \phi \mu_i^2} + \log\left\{\frac{\phi y_i^3}{w_i}\right\} + \log\{2\pi\}\right]
$$

$\mathrm{Var}[Y] = \phi\mu^3$, $\phi > 0$.

"Lognormal"

$$l(\mu_i, \phi; \log\{y_i\}, w_i) = -\frac{1}{2}\left[\frac{w_i(\log\{y_i\} - \mu_i)^2}{\phi} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\}\right]$$

$\mathrm{Var}[\log\{Y\}] = \phi$, $\phi > 0$.
If you specify DIST=LOGNORMAL with response variable Y, the GLIMMIX procedure assumes that $\log\{Y\} \sim N(\mu, \sigma^2)$. Note that the preceding density is not the density of Y.

Multinomial

$$l(\boldsymbol{\mu}_i, \phi; \mathbf{y}_i, w_i) = w_i \sum_{j=1}^{J} y_{ij} \log\{\mu_{ij}\}$$

$\phi \equiv 1$.

Negative Binomial

$$l(\mu_i, \phi; y_i, w_i) = y_i \log\left\{\frac{k\mu_i}{w_i}\right\} - (y_i + w_i/k)\log\left\{1 + \frac{k\mu_i}{w_i}\right\} + \log\left\{\frac{\Gamma(y_i + w_i/k)}{\Gamma(w_i/k)\Gamma(y_i + 1)}\right\}$$

$\mathrm{Var}[Y] = \mu + k\mu^2$, $k > 0, \phi \equiv 1$.
For a given k, the negative binomial distribution is a member of the exponential family. The parameter k is related to the scale of the data, because it is part of the variance function. However, it cannot be factored from the variance, as is the case with the ϕ parameter in many other distributions. The parameter k is designated as "Scale" in the "Parameter Estimates" table of the GLIMMIX procedure.

Normal (Gaussian)

$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{\phi} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\}\right]$$

$\mathrm{Var}[Y] = \phi$, $\phi > 0$.

Poisson

$$l(\mu_i, \phi; y_i, w_i) = w_i(y_i \log\{\mu_i\} - \mu_i - \log\{\Gamma(y_i + 1)\})$$

$\mathrm{Var}[Y] = \mu$, $\phi \equiv 1$.

Shifted T

$$\begin{aligned} z_i &= -0.5\log\{\phi/w_i\} + \log\{\Gamma(0.5(\nu + 1)\} \\ &\quad - \log\{\Gamma(0.5\nu)\} - 0.5 \times \log\{\pi\nu\} \\ l(\mu_i, \phi; y_i, w_i) &= -(\nu/2 + 0.5)\log\left\{1 + \frac{w_i}{\nu}\frac{(y_i - \mu_i)^2}{\phi}\right\} + z_i \end{aligned}$$

$\phi > 0, \nu > 0$, $\mathrm{Var}[Y] = \phi\nu/(\nu - 2)$.

Define the parameter vector for the generalized linear model as $\boldsymbol{\theta} = \boldsymbol{\beta}$, if $\phi \equiv 1$, and as $\boldsymbol{\theta} = [\boldsymbol{\beta}', \phi]'$ otherwise. $\boldsymbol{\beta}$ denotes the fixed-effects parameters in the linear predictor. For the negative binomial distribution, the relevant parameter vector is $\boldsymbol{\theta} = [\boldsymbol{\beta}', k]'$. The gradient and Hessian of the negative log likelihood are then

$$\mathbf{g} = -\frac{\partial L(\boldsymbol{\theta}; \mathbf{y})}{\partial \boldsymbol{\theta}} \qquad \mathbf{H} = -\frac{\partial^2 L(\boldsymbol{\theta}; \mathbf{y})}{\partial \boldsymbol{\theta}\, \partial \boldsymbol{\theta}'}$$

The GLIMMIX procedure computes the gradient vector and Hessian matrix analytically, unless your programming statements involve functions whose derivatives are determined by finite differences. If the procedure is in scoring mode, $\mathbf{H}$ is replaced by its expected value. PROC GLIMMIX is in scoring mode when the number n of SCORING=n iterations has not been exceeded and the optimization technique uses second derivatives, or when the Hessian is computed at convergence and the EXPHESSIAN option is in effect. Note that the objective function is the negative log likelihood when the GLIMMIX procedure fits a GLM model. The procedure performs a minimization problem in this case.

In models for independent data with known distribution, parameter estimates are obtained by the method of maximum likelihood. No parameters are profiled from the optimization. The default optimization technique for GLMs is the Newton-Raphson algorithm, except for Gaussian models with identity link, which do not require iterative model fitting. In the case of a Gaussian model, the scale parameter is estimated by restricted maximum likelihood, because this estimate is unbiased. The results from the GLIMMIX procedure agree with those from the GLM and REG procedures for such models. You can obtain the maximum likelihood estimate of the scale parameter with the NOREML option in the PROC GLIMMIX statement. To change the optimization algorithm, use the TECHNIQUE= option in the NLOPTIONS statement.

Standard errors of the parameter estimates are obtained from the inverse of the (observed or expected) second derivative matrix $\mathbf{H}$.

Scale and Dispersion Parameters

The parameter ϕ in the log-likelihood functions is a scale parameter. McCullagh and Nelder (1989, p. 29) refer to it as the dispersion parameter. With the exception of the normal distribution, ϕ does not correspond to the variance of an observation, the variance of an observation in a generalized linear model is a function of ϕ and μ. In a generalized linear model (GLM mode), the GLIMMIX procedure displays the estimate of ϕ is as "Scale" in the "Parameter Estimates" table. Note that for some distributions this scale is different from that reported by the GENMOD procedure in its "Parameter Estimates" table. The scale reported by PROC GENMOD is sometimes a transformation of the dispersion parameter in the log-likelihood function. Table 52.23 displays the relationship between the "Scale" entries reported by the two procedures in terms of the ϕ (or k) parameter in the PROC GLIMMIX log-likelihood functions.

Table 52.23 Scales in Parameter Estimates Table

Distribution	PROC GLIMMIX Reports	PROC GENMOD Reports
Beta	$\widehat{\phi}$	N/A
Gamma	$\widehat{\phi}$	$1/\widehat{\phi}$
Inverse Gaussian	$\widehat{\phi}$	$\sqrt{\widehat{\phi}}$
Negative binomial	$\widehat{k}$	$\widehat{k}$
Normal	$\widehat{\phi} = \widehat{\mathrm{Var}}[Y]$	$\sqrt{\widehat{\phi}}$

Note that for normal linear models, PROC GLIMMIX by default estimates the parameters by restricted maximum likelihood, whereas PROC GENMOD estimates the parameters by maximum likelihood. As a consequence, the scale parameter in the "Parameter Estimates" table of the GLIMMIX procedure coincides for these models with the mean-squared error estimate of the GLM or REG procedure. To obtain maximum likelihood estimates in a normal linear model in the GLIMMIX procedure, specify the NOREML option in the PROC GLIMMIX statement.

Quasi-likelihood for Independent Data

Quasi-likelihood estimation uses only the first and second moment of the response. In the case of independent data, this requires only a specification of the mean and variance of your data. The GLIMMIX procedure estimates parameters by quasi-likelihood, if the following conditions are met:

- The response distribution is unknown, because of a user-specified variance function.
- There are no G-side random effects.
- There are no R-side covariance structures or at most an overdispersion parameter.

Under some mild regularity conditions, the function

$$Q(\mu_i, y_i) = \int_{y_i}^{\mu_i} \frac{y_i - t}{\phi a(t)}\, dt$$

known as the log quasi-likelihood of the ith observation, has some properties of a log-likelihood function (McCullagh and Nelder 1989, p. 325). For example, the expected value of its derivative is zero, and the variance of its derivative equals the negative of the expected value of the second derivative. Consequently,

$$QL(\boldsymbol{\mu}, \phi, \mathbf{y}) = \sum_{i=1}^{n} f_i w_i \frac{Y_i - \mu_i}{\phi a(\mu_i)}$$

can serve as the score function for estimation. Quasi-likelihood estimation takes as the gradient and "Hessian" matrix—with respect to the fixed-effects parameters $\boldsymbol{\beta}$—the quantities

$$\mathbf{g}_{ql} = \left[g_{ql,j}\right] = \left[\frac{\partial QL(\boldsymbol{\mu}, \phi, \mathbf{y})}{\partial \beta_j}\right] = \mathbf{D}'\mathbf{V}^{-1}(\mathbf{Y} - \boldsymbol{\mu})/\phi$$

$$\mathbf{H}_{ql} = \left[h_{ql,jk}\right] = \left[\frac{\partial^2 QL(\boldsymbol{\mu}, \phi, \mathbf{y})}{\partial \beta_j \partial \beta_k}\right] = \mathbf{D}'\mathbf{V}^{-1}\mathbf{D}/\phi$$

In this expression, $\mathbf{D}$ is a matrix of derivatives of $\boldsymbol{\mu}$ with respect to the elements in $\boldsymbol{\beta}$, and $\mathbf{V}$ is a diagonal matrix containing variance functions, $\mathbf{V} = [a(\mu_1), \ldots, a(\mu_n)]$. Notice that $\mathbf{H}_{ql}$ is not the second derivative matrix of $Q(\boldsymbol{\mu}, \mathbf{y})$. Rather, it is the negative of the expected value of $\partial \mathbf{g}_{ql}/\partial \boldsymbol{\beta}$. $\mathbf{H}_{ql}$ thus has the form of a "scoring Hessian."

The GLIMMIX procedure fixes the scale parameter ϕ at 1.0 by default. To estimate the parameter, add the statement

```
random _residual_;
```

The resulting estimator (McCullagh and Nelder 1989, p. 328) is

$$\widehat{\phi} = \frac{1}{m}\sum_{i=1}^{n} f_i w_i \frac{(y_i - \widehat{\mu}_i)^2}{a(\widehat{\mu}_i)}$$

where $m = f - \mathrm{rank}\{\mathbf{X}\}$ if the NOREML option is in effect, $m = f$ otherwise, and f is the sum of the frequencies.

See Example 52.4 for an application of quasi-likelihood estimation with PROC GLIMMIX.

Effects of Adding Overdispersion

You can add a multiplicative overdispersion parameter to a generalized linear model in the GLIMMIX procedure with the statement

```
random _residual_;
```

For models in which $\phi \equiv 1$, this effectively lifts the constraint of the parameter. In models that already contain a ϕ or k scale parameter—such as the normal, gamma, or negative binomial model—the statement adds a multiplicative scalar (the overdispersion parameter, ϕ_o) to the variance function.

The overdispersion parameter is estimated from Pearson's statistic after all other parameters have been determined by (restricted) maximum likelihood or quasi-likelihood. This estimate is

$$\widehat{\phi}_o = \frac{1}{\phi^p m}\sum_{i=1}^{n} f_i w_i \frac{(y_i - \mu_i)^2}{a(\mu_i)}$$

where $m = f - \mathrm{rank}\{\mathbf{X}\}$ if the NOREML option is in effect, and $m = f$ otherwise, and f is the sum of the frequencies. The power p is –1 for the gamma distribution and 1 otherwise.

Adding an overdispersion parameter does not alter any of the other parameter estimates. It only changes the variance-covariance matrix of the estimates by a certain factor. If overdispersion arises from correlations among the observations, then you should investigate more complex random-effects structures.

Generalized Linear Mixed Models Theory

Model or Integral Approximation

In a generalized linear model, the log likelihood is well defined, and an objective function for estimation of the parameters is simple to construct based on the independence of the data. In a GLMM, several problems must be overcome before an objective function can be computed.

- The model might be vacuous in the sense that no valid joint distribution can be constructed either in general or for a particular set of parameter values. For example, if $\mathbf{Y}$ is an equicorrelated $(n \times 1)$ vector of binary responses with the same success probability and a symmetric distribution, then the lower bound on the correlation parameter depends on n and π (Gilliland and Schabenberger 2001). If further restrictions are placed on the joint distribution, as in Bahadur (1961), the correlation is also restricted from above.

- The dependency between mean and variance for nonnormal data places constraints on the possible correlation models that simultaneously yield valid joint distributions and a desired conditional distributions. Thus, for example, aspiring for conditional Poisson variates that are marginally correlated according to a spherical spatial process might not be possible.
- Even if the joint distribution is feasible mathematically, it still can be out of reach computationally. When data are independent, conditional on the random effects, the marginal log likelihood can in principle be constructed by integrating out the random effects from the joint distribution. However, numerical integration is practical only when the number of random effects is small and when the data have a clustered (subject) structure.

Because of these special features of generalized linear mixed models, many estimation methods have been put forth in the literature. The two basic approaches are (1) to approximate the objective function and (2) to approximate the model. Algorithms in the second category can be expressed in terms of Taylor series (linearizations) and are hence also known as linearization methods. They employ expansions to approximate the model by one based on pseudo-data with fewer nonlinear components. The process of computing the linear approximation must be repeated several times until some criterion indicates lack of further progress. Schabenberger and Gregoire (1996) list numerous algorithms based on Taylor series for the case of clustered data alone. The fitting methods based on linearizations are usually doubly iterative. The generalized linear mixed model is approximated by a linear mixed model based on current values of the covariance parameter estimates. The resulting linear mixed model is then fit, which is itself an iterative process. On convergence, the new parameter estimates are used to update the linearization, which results in a new linear mixed model. The process stops when parameter estimates between successive linear mixed model fits change only within a specified tolerance.

Integral approximation methods approximate the log likelihood of the GLMM and submit the approximated function to numerical optimization. Various techniques are used to compute the approximation: Laplace methods, quadrature methods, Monte Carlo integration, and Markov chain Monte Carlo methods. The advantage of integral approximation methods is to provide an actual objective function for optimization. This enables you to perform likelihood ratio tests among nested models and to compute likelihood-based fit statistics. The estimation process is singly iterative. The disadvantage of integral approximation methods is the difficulty of accommodating crossed random effects and multiple subject effects, and the inability to accommodate R-side covariance structures, even only R-side overdispersion. The number of random effects should be small for integral approximation methods to be practically feasible.

The advantages of linearization-based methods include a relatively simple form of the linearized model that typically can be fit based on only the mean and variance in the linearized form. Models for which the joint distribution is difficult—or impossible—to ascertain can be fit with linearization-based approaches. Models with correlated errors, a large number of random effects, crossed random effects, and multiple types of subjects are thus excellent candidates for linearization methods. The disadvantages of this approach include the absence of a true objective function for the overall optimization process and potentially biased estimates, especially for binary data when the number of observations per subject is small (see the section "Notes on Bias of Estimators" on page 3937 for further comments and considerations about the bias of estimates in generalized linear mixed models). Because the objective function to be optimized after each linearization update depends on the current pseudo-data, objective functions are not comparable across linearizations. The estimation process can fail at both levels of the double iteration scheme.

By default the GLIMMIX procedure fits generalized linear mixed models based on linearizations. The default estimation method in GLIMMIX for models containing random effects is a technique known as restricted

pseudo-likelihood (RPL) (Wolfinger and O'Connell 1993) estimation with an expansion around the current estimate of the best linear unbiased predictors of the random effects (METHOD=RSPL).

Two maximum likelihood estimation methods based on integral approximation are available in the GLIMMIX procedure. If you choose METHOD=LAPLACE in a GLMM, the GLIMMIX procedure performs maximum likelihood estimation based on a Laplace approximation of the marginal log likelihood. See the section "Maximum Likelihood Estimation Based on Laplace Approximation" on page 3929 for details about the Laplace approximation with PROC GLIMMIX. If you choose METHOD=QUAD in the PROC GLIMMIX statement in a generalized linear mixed model, the GLIMMIX procedure estimates the model parameters by adaptive Gauss-Hermite quadrature. See the section "Maximum Likelihood Estimation Based on Adaptive Quadrature" on page 3932 for details about the adaptive Gauss-Hermite quadrature approximation with PROC GLIMMIX.

The following subsections discuss the three estimation methods in turn. Keep in mind that your modeling possibilities are increasingly restricted in the order of these subsections. For example, in the class of generalized linear mixed models, the pseudo-likelihood estimation methods place no restrictions on the covariance structure, and Laplace estimation adds restriction with respect to the R-side covariance structure. Adaptive quadrature estimation further requires a clustered data structure—that is, the data must be processed by subjects.

Table 52.24 Model Restrictions Depending on Estimation Method

Method	Restriction
RSPL, RMPL	None
MSPL, MMPL	None
LAPLACE	No R-side effects
QUAD	No R-side effects Requires SUBJECT= effect Requires processing by subjects

Pseudo-likelihood Estimation Based on Linearization

The Pseudo-model

Recall from the section "Notation for the Generalized Linear Mixed Model" on page 3781 that

$$\mathrm{E}[\mathbf{Y}|\boldsymbol{\gamma}] = g^{-1}(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}) = g^{-1}(\boldsymbol{\eta}) = \boldsymbol{\mu}$$

where $\boldsymbol{\gamma} \sim N(\mathbf{0}, \mathbf{G})$ and $\mathrm{Var}[\mathbf{Y}|\boldsymbol{\gamma}] = \mathbf{A}^{1/2}\mathbf{R}\mathbf{A}^{1/2}$. Following Wolfinger and O'Connell (1993), a first-order Taylor series of $\boldsymbol{\mu}$ about $\widetilde{\boldsymbol{\beta}}$ and $\widetilde{\boldsymbol{\gamma}}$ yields

$$g^{-1}(\boldsymbol{\eta}) \doteq g^{-1}(\widetilde{\boldsymbol{\eta}}) + \widetilde{\boldsymbol{\Delta}}\mathbf{X}(\boldsymbol{\beta} - \widetilde{\boldsymbol{\beta}}) + \widetilde{\boldsymbol{\Delta}}\mathbf{Z}(\boldsymbol{\gamma} - \widetilde{\boldsymbol{\gamma}})$$

where

$$\widetilde{\boldsymbol{\Delta}} = \left(\frac{\partial g^{-1}(\boldsymbol{\eta})}{\partial \boldsymbol{\eta}}\right)_{\widetilde{\boldsymbol{\beta}},\widetilde{\boldsymbol{\gamma}}}$$

is a diagonal matrix of derivatives of the conditional mean evaluated at the expansion locus. Rearranging terms yields the expression

$$\widetilde{\boldsymbol{\Delta}}^{-1}(\boldsymbol{\mu} - g^{-1}(\widetilde{\boldsymbol{\eta}})) + \mathbf{X}\widetilde{\boldsymbol{\beta}} + \mathbf{Z}\widetilde{\boldsymbol{\gamma}} \doteq \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}$$

The left side is the expected value, conditional on $\boldsymbol{\gamma}$, of

$$\widetilde{\boldsymbol{\Delta}}^{-1}(\mathbf{Y} - g^{-1}(\widetilde{\boldsymbol{\eta}})) + \mathbf{X}\widetilde{\boldsymbol{\beta}} + \mathbf{Z}\widetilde{\boldsymbol{\gamma}} \equiv \mathbf{P}$$

and

$$\mathrm{Var}[\mathbf{P}|\boldsymbol{\gamma}] = \widetilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{1/2}\mathbf{R}\mathbf{A}^{1/2}\widetilde{\boldsymbol{\Delta}}^{-1}$$

You can thus consider the model

$$\mathbf{P} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

which is a linear mixed model with pseudo-response $\mathbf{P}$, fixed effects $\boldsymbol{\beta}$, random effects $\boldsymbol{\gamma}$, and $\mathrm{Var}[\boldsymbol{\epsilon}] = \mathrm{Var}[\mathbf{P}|\boldsymbol{\gamma}]$.

Objective Functions

Now define

$$\mathbf{V}(\boldsymbol{\theta}) = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \widetilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{1/2}\mathbf{R}\mathbf{A}^{1/2}\widetilde{\boldsymbol{\Delta}}^{-1}$$

as the marginal variance in the linear mixed pseudo-model, where $\boldsymbol{\theta}$ is the $(q \times 1)$ parameter vector containing all unknowns in $\mathbf{G}$ and $\mathbf{R}$. Based on this linearized model, an objective function can be defined, assuming that the distribution of $\mathbf{P}$ is known. The GLIMMIX procedure assumes that $\boldsymbol{\epsilon}$ has a normal distribution. The maximum log pseudo-likelihood (M*x*PL) and restricted log pseudo-likelihood (R*x*PL) for $\mathbf{P}$ are then

$$l(\boldsymbol{\theta}, \mathbf{p}) = -\frac{1}{2}\log|\mathbf{V}(\boldsymbol{\theta})| - \frac{1}{2}\mathbf{r}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{r} - \frac{f}{2}\log\{2\pi\}$$
$$l_R(\boldsymbol{\theta}, \mathbf{p}) = -\frac{1}{2}\log|\mathbf{V}(\boldsymbol{\theta})| - \frac{1}{2}\mathbf{r}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{r} - \frac{1}{2}\log|\mathbf{X}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{X}| - \frac{f-k}{2}\log\{2\pi\}$$

with $\mathbf{r} = \mathbf{p} - \mathbf{X}(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{p}$. f denotes the sum of the frequencies used in the analysis, and k denotes the rank of $\mathbf{X}$. The fixed-effects parameters $\boldsymbol{\beta}$ are profiled from these expressions. The parameters in $\boldsymbol{\theta}$ are estimated by the optimization techniques specified in the NLOPTIONS statement. The objective function for minimization is $-2l(\boldsymbol{\theta}, \mathbf{p})$ or $-2l_R(\boldsymbol{\theta}, \mathbf{p})$. At convergence, the profiled parameters are estimated and the random effects are predicted as

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{p}$$
$$\widehat{\boldsymbol{\gamma}} = \widehat{\mathbf{G}}\mathbf{Z}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\widehat{\mathbf{r}}$$

With these statistics, the pseudo-response and error weights of the linearized model are recomputed and the objective function is minimized again. The predictors $\widehat{\boldsymbol{\gamma}}$ are the estimated BLUPs in the approximated linear model. This process continues until the relative change between parameter estimates at two successive (outer) iterations is sufficiently small. See the PCONV= option in the PROC GLIMMIX statement for the computational details about how the GLIMMIX procedure compares parameter estimates across optimizations.

If the conditional distribution contains a scale parameter $\phi \neq 1$ (Table 52.22), the GLIMMIX procedure profiles this parameter in GLMMs from the log pseudo-likelihoods as well. To this end define

$$\mathbf{V}(\boldsymbol{\theta}^*) = \widetilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{1/2}\mathbf{R}^*\mathbf{A}^{1/2}\widetilde{\boldsymbol{\Delta}}^{-1} + \mathbf{Z}\mathbf{G}^*\mathbf{Z}'$$

where $\boldsymbol{\theta}^*$ is the covariance parameter vector with $q-1$ elements. The matrices $\mathbf{G}^*$ and $\mathbf{R}^*$ are appropriately reparameterized versions of $\mathbf{G}$ and $\mathbf{R}$. For example, if $\mathbf{G}$ has a variance component structure and $\mathbf{R} = \phi\mathbf{I}$, then $\boldsymbol{\theta}^*$ contains ratios of the variance components and ϕ, and $\mathbf{R}^* = \mathbf{I}$. The solution for $\widehat{\phi}$ is

$$\widehat{\phi} = \widehat{\mathbf{r}}'\mathbf{V}(\widehat{\boldsymbol{\theta}}^*)^{-1}\widehat{\mathbf{r}}/m$$

where $m = f$ for MxPL and $m = f - k$ for RxPL. Substitution into the previous functions yields the profiled log pseudo-likelihoods,

$$\begin{aligned} l(\boldsymbol{\theta}^*, \mathbf{p}) &= -\frac{1}{2}\log|\mathbf{V}(\boldsymbol{\theta}^*)| - \frac{f}{2}\log\left\{\mathbf{r}'\mathbf{V}(\boldsymbol{\theta}^*)^{-1}\mathbf{r}\right\} - \frac{f}{2}(1 + \log\{2\pi/f\}) \\ l_R(\boldsymbol{\theta}^*, \mathbf{p}) &= -\frac{1}{2}\log|\mathbf{V}(\boldsymbol{\theta}^*)| - \frac{f-k}{2}\log\left\{\mathbf{r}'\mathbf{V}(\boldsymbol{\theta}^*)^{-1}\mathbf{r}\right\} \\ &\quad -\frac{1}{2}\log|\mathbf{X}'\mathbf{V}(\boldsymbol{\theta}^*)^{-1}\mathbf{X}| - \frac{f-k}{2}(1 + \log\{2\pi/(f-k)\}) \end{aligned}$$

Profiling of ϕ can be suppressed with the NOPROFILE option in the PROC GLIMMIX statement.

Where possible, the objective function, its gradient, and its Hessian employ the sweep-based W-transformation (Hemmerle and Hartley 1973; Goodnight 1979; Goodnight and Hemmerle 1979). Further details about the minimization process in the general linear mixed model can be found in Wolfinger, Tobias, and Sall (1994).

Estimated Precision of Estimates

The GLIMMIX procedure produces estimates of the variability of $\widehat{\boldsymbol{\beta}}$, $\widehat{\boldsymbol{\theta}}$, and estimates of the prediction variability for $\widehat{\boldsymbol{\gamma}}$, $\mathrm{Var}[\widehat{\boldsymbol{\gamma}} - \boldsymbol{\gamma}]$. Denote as $\mathbf{S}$ the matrix

$$\mathbf{S} \equiv \widehat{\mathrm{Var}}[\mathbf{P}|\boldsymbol{\gamma}] = \widetilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}^{1/2}\mathbf{R}\mathbf{A}^{1/2}\widetilde{\boldsymbol{\Delta}}^{-1}$$

where all components on the right side are evaluated at the converged estimates. The mixed model equations (Henderson 1984) in the linear mixed (pseudo-)model are then

$$\begin{bmatrix} \mathbf{X}'\mathbf{S}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{S}^{-1}\mathbf{Z} \\ \mathbf{Z}'\mathbf{S}^{-1}\mathbf{X} & \mathbf{Z}'\mathbf{S}^{-1}\mathbf{Z} + \mathbf{G}(\widehat{\boldsymbol{\theta}})^{-1} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{S}^{-1}\mathbf{p} \\ \mathbf{Z}'\mathbf{S}^{-1}\mathbf{p} \end{bmatrix}$$

and

$$\begin{aligned} \mathbf{C} &= \begin{bmatrix} \mathbf{X}'\mathbf{S}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{S}^{-1}\mathbf{Z} \\ \mathbf{Z}'\mathbf{S}^{-1}\mathbf{X} & \mathbf{Z}'\mathbf{S}^{-1}\mathbf{Z} + \mathbf{G}(\widehat{\boldsymbol{\theta}})^{-1} \end{bmatrix}^- \\ &= \begin{bmatrix} \widehat{\boldsymbol{\Omega}} & -\widehat{\boldsymbol{\Omega}}\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{Z}\mathbf{G}(\widehat{\boldsymbol{\theta}}) \\ -\mathbf{G}(\widehat{\boldsymbol{\theta}})\mathbf{Z}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{X}\widehat{\boldsymbol{\Omega}} & \mathbf{M} + \mathbf{G}(\widehat{\boldsymbol{\theta}})\mathbf{Z}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{X}\widehat{\boldsymbol{\Omega}}\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{Z}\mathbf{G}(\widehat{\boldsymbol{\theta}}) \end{bmatrix} \end{aligned}$$

is the approximate estimated variance-covariance matrix of $[\widehat{\boldsymbol{\beta}}', \widehat{\boldsymbol{\gamma}}' - \boldsymbol{\gamma}']'$. Here, $\widehat{\boldsymbol{\Omega}} = (\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{X})^-$ and $\mathbf{M} = (\mathbf{Z}'\mathbf{S}^{-1}\mathbf{Z} + \mathbf{G}(\widehat{\boldsymbol{\theta}})^{-1})^{-1}$.

The square roots of the diagonal elements of $\widehat{\boldsymbol{\Omega}}$ are reported in the Standard Error column of the "Parameter Estimates" table. This table is produced with the SOLUTION option in the MODEL statement. The

prediction standard errors of the random-effects solutions are reported in the Std Err Pred column of the "Solution for Random Effects" table. This table is produced with the SOLUTION option in the RANDOM statement.

As a cautionary note, $\mathbf{C}$ tends to underestimate the true sampling variability of $[\widehat{\boldsymbol{\beta}}', \widehat{\boldsymbol{\gamma}}']'$, because no account is made for the uncertainty in estimating $\mathbf{G}$ and $\mathbf{R}$. Although inflation factors have been proposed (Kackar and Harville 1984; Kass and Steffey 1989; Prasad and Rao 1990), they tend to be small for data sets that are fairly well balanced. PROC GLIMMIX does not compute any inflation factors by default. The DDFM=KENWARDROGER option in the MODEL statement prompts PROC GLIMMIX to compute a specific inflation factor (Kenward and Roger 1997), along with Satterthwaite-based degrees of freedom.

If $\mathbf{G}(\widehat{\boldsymbol{\theta}})$ is singular, or if you use the CHOL option of the PROC GLIMMIX statement, the mixed model equations are modified as follows. Let $\mathbf{L}$ denote the lower triangular matrix so that $\mathbf{LL}' = \mathbf{G}(\widehat{\boldsymbol{\theta}})$. PROC GLIMMIX then solves the equations

$$\begin{bmatrix} \mathbf{X}'\mathbf{S}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{S}^{-1}\mathbf{ZL} \\ \mathbf{L}'\mathbf{Z}'\mathbf{S}^{-1}\mathbf{X} & \mathbf{L}'\mathbf{Z}'\mathbf{S}^{-1}\mathbf{ZL}+\mathbf{I} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\tau}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{S}^{-1}\mathbf{p} \\ \mathbf{L}'\mathbf{Z}'\mathbf{S}^{-1}\mathbf{p} \end{bmatrix}$$

and transforms $\widehat{\boldsymbol{\tau}}$ and a generalized inverse of the left-side coefficient matrix by using $\mathbf{L}$.

The asymptotic covariance matrix of the covariance parameter estimator $\widehat{\boldsymbol{\theta}}$ is computed based on the observed or expected Hessian matrix of the optimization procedure. Consider first the case where the scale parameter ϕ is not present or not profiled. Because $\boldsymbol{\beta}$ is profiled from the pseudo-likelihood, the objective function for minimization is $f(\boldsymbol{\theta}) = -2l(\boldsymbol{\theta}, \mathbf{p})$ for METHOD=MSPL and METHOD=MMPL and $f(\boldsymbol{\theta}) = -2l_R(\boldsymbol{\theta}, \mathbf{p})$ for METHOD=RSPL and METHOD=RMPL. Denote the observed Hessian (second derivative) matrix as

$$\mathbf{H} = \frac{\partial^2 f(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\,\partial\boldsymbol{\theta}'}$$

The GLIMMIX procedure computes the variance of $\widehat{\boldsymbol{\theta}}$ by default as $2\mathbf{H}^{-1}$. If the Hessian is not positive definite, a sweep-based generalized inverse is used instead. When the EXPHESSIAN option of the PROC GLIMMIX statement is used, or when the procedure is in scoring mode at convergence (see the SCORING option in the PROC GLIMMIX statement), the observed Hessian is replaced with an approximated expected Hessian matrix in these calculations.

Following Wolfinger, Tobias, and Sall (1994), define the following components of the gradient and Hessian in the optimization:

$$\begin{aligned}
\mathbf{g}_1 &= \frac{\partial}{\partial\boldsymbol{\theta}}\,\mathbf{r}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{r} \\
\mathbf{H}_1 &= \frac{\partial^2}{\partial\boldsymbol{\theta}\,\partial\boldsymbol{\theta}'}\,\log\{\mathbf{V}(\boldsymbol{\theta})\} \\
\mathbf{H}_2 &= \frac{\partial^2}{\partial\boldsymbol{\theta}\,\partial\boldsymbol{\theta}'}\,\mathbf{r}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{r} \\
\mathbf{H}_3 &= \frac{\partial^2}{\partial\boldsymbol{\theta}\,\partial\boldsymbol{\theta}'}\,\log\{|\mathbf{X}'\mathbf{V}(\boldsymbol{\theta})^{-1}\mathbf{X}|\}
\end{aligned}$$

Table 52.25 gives expressions for the Hessian matrix $\mathbf{H}$ depending on estimation method, profiling, and scoring.

Table 52.25 Hessian Computation in PROC GLIMMIX

Profiling	Scoring	M*x*PL	R*x*PL
No	No	$\mathbf{H}_1 + \mathbf{H}_2$	$\mathbf{H}_1 + \mathbf{H}_2 + \mathbf{H}_3$
No	Yes	$-\mathbf{H}_1$	$-\mathbf{H}_1 + \mathbf{H}_3$
No	Modified	$-\mathbf{H}_1$	$-\mathbf{H}_1 - \mathbf{H}_3$
Yes	No	$\begin{bmatrix} \mathbf{H}_1 + \mathbf{H}_2/\phi & -\mathbf{g}_2/\phi^2 \\ -\mathbf{g}_2'/\phi^2 & f/\phi^2 \end{bmatrix}$	$\begin{bmatrix} \mathbf{H}_1 + \mathbf{H}_2/\phi + \mathbf{H}_3 & -\mathbf{g}_2/\phi^2 \\ -\mathbf{g}_2'/\phi^2 & (f-k)/\phi^2 \end{bmatrix}$
Yes	Yes	$\begin{bmatrix} -\mathbf{H}_1 & -\mathbf{g}_2/\phi^2 \\ -\mathbf{g}_2'/\phi^2 & f/\phi^2 \end{bmatrix}$	$\begin{bmatrix} -\mathbf{H}_1 + \mathbf{H}_3 & -\mathbf{g}_2/\phi^2 \\ -\mathbf{g}_2'/\phi^2 & (f-k)/\phi^2 \end{bmatrix}$
Yes	Modified	$\begin{bmatrix} -\mathbf{H}_1 & -\mathbf{g}_2/\phi^2 \\ -\mathbf{g}_2'/\phi^2 & f/\phi^2 \end{bmatrix}$	$\begin{bmatrix} -\mathbf{H}_1 - \mathbf{H}_3 & -\mathbf{g}_2/\phi^2 \\ -\mathbf{g}_2'/\phi^2 & (f-k)/\phi^2 \end{bmatrix}$

The "Modified" expressions for the Hessian under scoring in R*x*PL estimation refer to a modified scoring method. In some cases, the modification leads to faster convergence than the standard scoring algorithm. The modification is requested with the SCOREMOD option in the PROC GLIMMIX statement.

Finally, in the case of a profiled scale parameter ϕ, the Hessian for the $(\boldsymbol{\theta}^*, \phi)$ parameterization is converted into that for the $\boldsymbol{\theta}$ parameterization as

$$\mathbf{H}(\boldsymbol{\theta}) = \mathbf{B}\mathbf{H}(\boldsymbol{\theta}^*, \phi)\mathbf{B}'$$

where

$$\mathbf{B} = \begin{bmatrix} 1/\phi & 0 & \cdots & 0 & 0 \\ 0 & 1/\phi & \cdots & 0 & 0 \\ 0 & \cdots & \cdots & 1/\phi & 0 \\ -\theta_1^*/\phi & -\theta_2^*/\phi & \cdots & -\theta_{q-1}^*/\phi & 1 \end{bmatrix}$$

Subject-Specific and Population-Averaged (Marginal) Expansions

There are two basic choices for the expansion locus of the linearization. A subject-specific (SS) expansion uses

$$\widetilde{\boldsymbol{\beta}} = \widehat{\boldsymbol{\beta}} \quad \widetilde{\boldsymbol{\gamma}} = \widehat{\boldsymbol{\gamma}}$$

which are the current estimates of the fixed effects and estimated BLUPs. The population-averaged (PA) expansion expands about the same fixed effects and the expected value of the random effects

$$\widetilde{\boldsymbol{\beta}} = \widehat{\boldsymbol{\beta}} \quad \widetilde{\boldsymbol{\gamma}} = \mathbf{0}$$

To recompute the pseudo-response and weights in the SS expansion, the BLUPs must be computed every time the objective function in the linear mixed model is maximized. The PA expansion does not require any

BLUPs. The four pseudo-likelihood methods implemented in the GLIMMIX procedure are the 2×2 factorial combination between two expansion loci and residual versus maximum pseudo-likelihood estimation. The following table shows the combination and the corresponding values of the METHOD= option (PROC GLIMMIX statement); METHOD=RSPL is the default.

Type of PL	Expansion Locus $\widehat{\boldsymbol{\gamma}}$	$\mathbf{E}[\boldsymbol{\gamma}]$
Residual	RSPL	RMPL
Maximum	MSPL	MMPL

Maximum Likelihood Estimation Based on Laplace Approximation

Objective Function

Let $\boldsymbol{\beta}$ denote the vector of fixed-effects parameters and $\boldsymbol{\theta}$ the vector of covariance parameters. For Laplace estimation in the GLIMMIX procedure, $\boldsymbol{\theta}$ includes the G-side parameters and a possible scale parameter ϕ, provided that the conditional distribution of the data contains such a scale parameter. $\boldsymbol{\theta}^*$ is the vector of the G-side parameters.

The marginal distribution of the data in a mixed model can be expressed as

$$\begin{aligned} p(\mathbf{y}) &= \int p(\mathbf{y}|\boldsymbol{\gamma}, \boldsymbol{\beta}, \phi)\, p(\boldsymbol{\gamma}|\boldsymbol{\theta}^*)\; d\boldsymbol{\gamma} \\ &= \int \exp\left\{\log\{p(\mathbf{y}|\boldsymbol{\gamma}, \boldsymbol{\beta}, \phi)\} + \log\{p(\boldsymbol{\gamma}|\boldsymbol{\theta}^*)\}\right\}\; d\boldsymbol{\gamma} \\ &= \int \exp\{c_l\, f(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma})\}\; d\boldsymbol{\gamma} \end{aligned}$$

If the constant c_l is large, the Laplace approximation of this integral is

$$L(\boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}, \mathbf{y}) = \left(\frac{2\pi}{c_l}\right)^{n_\gamma/2} |-f''(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}})|^{-1/2}\, e^{c_l f(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}})}$$

where n_γ is the number of elements in $\boldsymbol{\gamma}$, f'' is the second derivative matrix

$$f''(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}) = \frac{\partial^2 f(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}\, \partial \boldsymbol{\gamma}'}\Big|_{\widehat{\boldsymbol{\gamma}}}$$

and $\widehat{\boldsymbol{\gamma}}$ satisfies the first-order condition

$$\frac{\partial f(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}} = \mathbf{0}$$

The objective function for Laplace parameter estimation in the GLIMMIX procedure is $-2\log\{L(\boldsymbol{\beta}, \boldsymbol{\theta}\ \widehat{\boldsymbol{\gamma}}, \mathbf{y})\}$. The optimization process is singly iterative, but because $\widehat{\boldsymbol{\gamma}}$ depends on $\widehat{\boldsymbol{\beta}}$ and $\widehat{\boldsymbol{\theta}}$, the GLIMMIX procedure solves a suboptimization problem to determine for given values of $\widehat{\boldsymbol{\beta}}$ and $\widehat{\boldsymbol{\theta}}$ the random-effects solution vector that maximizes $f(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma})$.

When you have longitudinal or clustered data with m independent subjects or clusters, the vector of observations can be written as $\mathbf{y} = [\mathbf{y}_1', \ldots, \mathbf{y}_m']'$, where $\mathbf{y}_i$ is an $n_i \times 1$ vector of observations for subject (cluster) i ($i = 1, \ldots, m$). In this case, assuming conditional independence such that

$$p(\mathbf{y}_i|\boldsymbol{\gamma}_i) = \prod_{j=1}^{n_i} p(y_{ij}|\boldsymbol{\gamma}_i)$$

the marginal distribution of the data can be expressed as

$$\begin{aligned} p(\mathbf{y}) = \prod_{i=1}^{m} p(\mathbf{y}_i) &= \prod_{i=1}^{m} \int p(\mathbf{y}_i|\boldsymbol{\gamma}_i)\, p(\boldsymbol{\gamma}_i)\, d\boldsymbol{\gamma}_i \\ &= \prod_{i=1}^{m} \int \exp\{n_i f(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma}_i)\}\, d\boldsymbol{\gamma}_i \end{aligned}$$

where

$$\begin{aligned} n_i f(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma}_i) &= \log\{p(\mathbf{y}_i|\boldsymbol{\gamma}_i)\, p(\boldsymbol{\gamma}_i)\} \\ &= \sum_{j=1}^{n_i} \log\left\{p(y_{ij}|\boldsymbol{\gamma}_i)\right\} + \log\{p(\boldsymbol{\gamma}_i)\} \end{aligned}$$

When the number of observations within a cluster, n_i, is large, the Laplace approximation to the ith individual's marginal probability density function is

$$\begin{aligned} p(\mathbf{y}_i|\boldsymbol{\beta}, \boldsymbol{\theta}) &= \int \exp\{n_i f(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma}_i)\}\, d\boldsymbol{\gamma}_i \\ &= \frac{(2\pi)^{n_\gamma}/2}{|-n_i f''(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}_i)|^{1/2}} \exp\{n_i f(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}_i)\} \end{aligned}$$

where $n_{\gamma i}$ is the common dimension of the random effects, $\boldsymbol{\gamma}_i$. In this case, provided that the constant $c_l = \min\{n_i\}$ is large, the Laplace approximation to the marginal log likelihood is

$$\begin{aligned} \log\{L(\boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}, \mathbf{y})\} = \sum_{i=1}^{m} \Big\{ & n_i f(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}_i) + \frac{n_{\gamma i}}{2} \log\{2\pi\} \\ & - \frac{1}{2} \log|-n_i f''(\boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}_i)| \Big\} \end{aligned}$$

which serves as the objective function for the METHOD=LAPLACE estimator in PROC GLIMMIX.

The Laplace approximation implemented in the GLIMMIX procedure differs from that in Wolfinger (1993) and Pinheiro and Bates (1995) in important respects. Wolfinger (1993) assumed a flat prior for $\boldsymbol{\beta}$ and expanded the integrand around $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, leaving only the covariance parameters for the overall optimization. The "fixed" effects $\boldsymbol{\beta}$ and the random effects $\boldsymbol{\gamma}$ are determined in a suboptimization that takes the form of a linear mixed model step with pseudo-data. The GLIMMIX procedure involves only the random effects vector $\boldsymbol{\gamma}$ in the suboptimization. Pinheiro and Bates (1995) and Wolfinger (1993) consider a modified Laplace approximation that replaces the second derivative $f''(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}})$ with an (approximate) expected value, akin to scoring. The GLIMMIX procedure does not use an approximation to $f''(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}})$. The METHOD=RSPL estimates in PROC GLIMMIX are equivalent to the estimates obtained with the modified Laplace approximation in Wolfinger (1993). The objective functions of METHOD=RSPL and Wolfinger (1993) differ in a constant that depends on the number of parameters.

Asymptotic Properties and the Importance of Subjects

Suppose that the GLIMMIX procedure processes your data by subjects (see the section "Processing by Subjects" on page 3957) and let n_i denote the number of observations per subject, $i = 1, \ldots, s$. Arguments in Vonesh (1996) show that the maximum likelihood estimator based on the Laplace approximation is a consistent estimator to order $O_p\{\max\{1/\sqrt{s}\}, 1/\min\{n_i\}\}$. In other words, as the number of subjects and the number of observations per subject grows, the small-sample bias of the Laplace estimator disappears. Note that the term involving the number of subjects in this maximum relates to standard asymptotic theory, and the term involving the number of observations per subject relates to the accuracy of the Laplace approximation (Vonesh 1996). In the case where random effects enter the model linearly, the Laplace approximation is exact and the requirement that $\min\{n_i\} \to \infty$ can be dropped.

If your model is not processed by subjects but is equivalent to a subject model, the asymptotics with respect to s still apply, because the Hessian matrix of the suboptimization for $\boldsymbol{\gamma}$ breaks into s separate blocks. For example, the following two models are equivalent with respect to s and n_i, although only for the first model does PROC GLIMMIX process the data explicitly by subjects:

```
proc glimmix method=laplace;
   class sub A;
   model y = A;
   random intercept / subject=sub;
run;

proc glimmix method=laplace;
   class sub A;
   model y = A;
   random sub;
run;
```

The same holds, for example, for models with independent nested random effects. The following two models are equivalent, and you can derive asymptotic properties related to s and $\min\{n_i\}$ from the model in the first run:

```
proc glimmix method=laplace;
   class A B block;
   model y = A B A*B;
   random intercept A / subject=block;
run;

proc glimmix method=laplace;
   class A B block;
   model y = A B A*B;
   random block a*block;
run;
```

The Laplace approximation requires that the dimension of the integral does not increase with the size of the sample. Otherwise the error of the likelihood approximation does not diminish with n_i. This is the case, for example, with exchangeable arrays (Shun and McCullagh 1995), crossed random effects (Shun 1997), and correlated random effects of arbitrary dimension (Raudenbush, Yang, and Yosef 2000). Results in Shun (1997), for example, show that even in this case the standard Laplace approximation has smaller bias than pseudo-likelihood estimates.

Maximum Likelihood Estimation Based on Adaptive Quadrature

Quadrature methods, like the Laplace approximation, approximate integrals. If you choose METHOD=QUAD for a generalized linear mixed model, the GLIMMIX procedure approximates the marginal log likelihood with an adaptive Gauss-Hermite quadrature rule. Gaussian quadrature is particularly well suited to numerically evaluate integrals against probability measures (Lange 1999, Ch. 16). And Gauss-Hermite quadrature is appropriate when the density has kernel $\exp\{-x^2\}$ and integration extends over the real line, as is the case for the normal distribution. Suppose that $p(x)$ is a probability density function and the function $f(x)$ is to be integrated against it. Then the quadrature rule is

$$\int_{-\infty}^{\infty} f(x)p(x)\,dx \approx \sum_{i=1}^{N} w_i f(x_i)$$

where N denotes the number of quadrature points, the w_i are the quadrature weights, and the x_i are the abscissas. The Gaussian quadrature chooses abscissas in areas of high density, and if $p(x)$ is continuous, the quadrature rule is exact if $f(x)$ is a polynomial of up to degree $2N-1$. In the generalized linear mixed model the roles of $f(x)$ and $p(x)$ are played by the conditional distribution of the data given the random effects, and the random-effects distribution, respectively. Quadrature abscissas and weights are those of the standard Gauss-Hermite quadrature (Golub and Welsch 1969; see also Table 25.10 of Abramowitz and Stegun 1972; Evans 1993).

A numerical integration rule is called adaptive when it uses a variable step size to control the error of the approximation. For example, an adaptive trapezoidal rule uses serial splitting of intervals at midpoints until a desired tolerance is achieved. The quadrature rule in the GLIMMIX procedure is adaptive in the following sense: if you do not specify the number of quadrature points (nodes) with the QPOINTS= suboption of the METHOD=QUAD option, then the number of quadrature points is determined by evaluating the log likelihood at the starting values at a successively larger number of nodes until a tolerance is met (for more details see the text under the heading "Starting Values" in the next section). Furthermore, the GLIMMIX procedure centers and scales the quadrature points by using the empirical Bayes estimates (EBEs) of the random effects and the Hessian (second derivative) matrix from the EBE suboptimization. This centering and scaling improves the likelihood approximation by placing the abscissas according to the density function of the random effects. It is not, however, adaptiveness in the previously stated sense.

Objective Function

Let $\boldsymbol{\beta}$ denote the vector of fixed-effects parameters and $\boldsymbol{\theta}$ the vector of covariance parameters. For quadrature estimation in the GLIMMIX procedure, $\boldsymbol{\theta}$ includes the G-side parameters and a possible scale parameter ϕ, provided that the conditional distribution of the data contains such a scale parameter. $\boldsymbol{\theta}^*$ is the vector of the G-side parameters. The marginal distribution of the data for subject i in a mixed model can be expressed as

$$p(\mathbf{y}_i) = \int \cdots \int p(\mathbf{y}_i|\boldsymbol{\gamma}_i, \boldsymbol{\beta}, \phi)\; p(\boldsymbol{\gamma}_i|\boldsymbol{\theta}^*)\; d\boldsymbol{\gamma}_i$$

Suppose N_q denotes the number of quadrature points in each dimension (for each random effect) and r denotes the number of random effects. For each subject, obtain the empirical Bayes estimates of $\boldsymbol{\gamma}_i$ as the vector $\widehat{\boldsymbol{\gamma}}_i$ that minimizes

$$-\log\left\{p(\mathbf{y}_i|\boldsymbol{\gamma}_i, \boldsymbol{\beta}, \phi)p(\boldsymbol{\gamma}_i|\boldsymbol{\theta}^*)\right\} = f(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma}_i)$$

If $\mathbf{z} = [z_1, \ldots, z_{N_q}]$ are the standard abscissas for Gauss-Hermite quadrature, and $\mathbf{z}_j^* = [z_{j_1}, \ldots, z_{j_r}]$ is a point on the r-dimensional quadrature grid, then the centered and scaled abscissas are

$$\mathbf{a}_j^* = \widehat{\boldsymbol{\gamma}}_i + 2^{1/2} f''(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}_i)^{-1/2}\mathbf{z}_j^*$$

As for the Laplace approximation, f'' is the second derivative matrix with respect to the random effects,

$$f''(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}_i) = \frac{\partial^2 f(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma}_i)}{\partial \boldsymbol{\gamma}_i \partial \boldsymbol{\gamma}_i'}\Big|_{\widehat{\boldsymbol{\gamma}}_i}$$

These centered and scaled abscissas, along with the Gauss-Hermite quadrature weights $\mathbf{w} = [w_1, \ldots, w_{N_q}]$, are used to construct the r-dimensional integral by a sequence of one-dimensional rules

$$\begin{aligned} p(\mathbf{y}_i) &= \int \cdots \int p(\mathbf{y}_i|\boldsymbol{\gamma}_i, \boldsymbol{\beta}, \phi)\, p(\boldsymbol{\gamma}_i|\boldsymbol{\theta}^*)\, d\boldsymbol{\gamma}_i \\ &\approx 2^{r/2} |f''(\mathbf{y}_i, \boldsymbol{\beta}, \boldsymbol{\theta}; \widehat{\boldsymbol{\gamma}}_i)|^{-1/2} \\ &\quad \sum_{j_1=1}^{N_q} \cdots \sum_{j_r=1}^{N_q} \left[p(\mathbf{y}_i|\mathbf{a}_j^*, \boldsymbol{\beta}, \phi)\, p(\mathbf{a}_j^*|\boldsymbol{\theta}^*) \prod_{k=1}^{r} w_{j_k} \exp z_{j_k}^2 \right] \end{aligned}$$

The right-hand side of this expression, properly accumulated across subjects, is the objective function for adaptive quadrature estimation in the GLIMMIX procedure. The preceding expression constitutes a one-level adaptive Gaussian quadrature approximation.

As the number of random effects grows, the dimension of the integral increases accordingly. This increase can happen especially when you have nested random effects. In this case, the one-level quadrature approximation described earlier quickly becomes computationally infeasible. The following scenarios illustrate the relationship among the computational effort, the dimension of the random effects, and the number of quadrature nodes. Suppose that the A effect has four levels, and consider the following statements:

```
proc glimmix method=quad(qpoints=5);
   class A id;
   model y = / dist=negbin;
   random A / subject=id;
run;
```

For each subject, computing the marginal log likelihood requires the numerical evaluation of a four-dimensional integral. With the number of quadrature points set to five by the QPOINTS=5 option, this means that each marginal log-likelihood evaluation requires $5^4 = 625$ conditional log likelihoods to be computed for each observation on each pass through the data. As the number of quadrature points or the number of random effects increases, this constitutes a sizable computational effort. Suppose, for example, that just one additional random effect, B, with two levels is added as an interaction, as in the following statements:

```
proc glimmix method=quad(qpoints=5);
   class A B id;
   model y = / dist=negbin;
   random A A*B / subject=id;
run;
```

Now a single marginal likelihood calculation requires $5^{(4+8)}$ = 244,140,625 conditional log likelihoods for each observation on each pass through the data.

You can reduce the dimension of the random effects in the preceding PROC GLIMMIX code by factoring A out of the two random effects in the RANDOM statement, as shown in the following statements:

```
proc glimmix method=quad(qpoints=5);
   class A B id;
   model y = / dist=negbin;
   random int B / subject=id*A;
run;
```

With the random effects int and B, the preceding PROC GLIMMIX code requires the evaluation of $5^{(1+2)} = 125$ conditional log likelihoods for each observation on each pass through the data.

This idea of reducing the dimension of random effects is the key to the multilevel adaptive Gaussian quadrature algorithm described in Pinheiro and Chao (2006). By exploiting the sparseness in the random-effects design matrix Z, the multilevel quadrature algorithm reduces the dimension of the random effects to the sum of the dimensions of random effects from each level. You can use the FASTQUAD suboption in the METHOD=QUAD option to prompt PROC GLIMMIX to compute this multilevel quadrature approximation.

To see the effect of the FASTQUAD option, consider the following model for the preceding example:

```
proc glimmix method=quad(qpoints=5);
   class A B id;
   model y = / dist=negbin;
   random A A*B B/ subject=id;
run;
```

In this case, it is not possible to factor a single SUBJECT= variable out of all the random effects. Formulated in this one-level way, a single evaluation of the marginal likelihood requires the computing of $5^{(4+8+2)} = 488{,}281{,}250$ conditional log likelihoods for each observation on each pass through the data.

Alternatively, to take advantage of the multilevel quadrature approximation, you need to use the FASTQUAD option and explicitly specify the two-level structure by including one RANDOM statement for each level:

```
proc glimmix method=quad(qpoints=5 fastquad);
   class A B id;
   model y = / dist=negbin;
   random B     / subject=id;
   random int B / subject=id*A;
run;
```

The first RANDOM statement specifies the random effect B for the level that corresponds to id; the second RANDOM statement specifies the random effects int and B for the level that corresponds to id*A. With this specification, the multilevel quadrature approximation computes only $5^{(2+1+2)} = 3{,}125$ conditional log likelihoods for each observation on each pass through the data, where the exponent $(2 + 1 + 2)$ is the sum of the number of random effects in the two RANDOM statements.

In general, consider a two-level model in which m level-2 units are nested within each level-1 unit. In this case, the one-level N_q point adaptive quadrature approximation to a marginal likelihood that is an integral over r_1 level-1 random effects and r_2 level-2 random effects requires $N_q^{r_1 \times m + r_2}$ evaluations of the conditional log likelihoods for each observation. However, the two-level adaptive quadrature approximation requires only $N_q^{r_1 + r_2}$ evaluations of the conditional log likelihoods. By increasing exponentially with r_1 instead of with $r_1 \times m$, the multilevel quadrature algorithm significantly reduces the computational and memory requirements.

Quadrature or Laplace Approximation

If you select the quadrature rule with a single quadrature point, namely

```
proc glimmix method=quad(qpoints=1);
```

the results will be identical to METHOD=LAPLACE. Computationally, the two methods are not identical, however. METHOD=LAPLACE can be applied to a considerably larger class of models. For example, crossed random effects, models without subjects, or models with non-nested subjects can be handled with the Laplace approximation but not with quadrature. Furthermore, METHOD=LAPLACE draws on a number of computational simplifications that can increase its efficiency compared to a quadrature algorithm with a single node. For example, the Laplace approximation is possible with unbounded covariance parameter estimates (NOBOUND option in the PROC GLIMMIX statement) and can permit certain types of negative definite or indefinite **G** matrices. The adaptive quadrature approximation with scaled abscissas typically breaks down when **G** is not at least positive semidefinite.

In the multilevel quadrature algorithm, the total number of random effects is the sum of the number of random effects at each level. Still, as the number of random effects grows at any of the levels, quadrature approximation becomes computationally infeasible. Laplace approximation presents a computationally more expedient alternative.

If you wonder whether METHOD=LAPLACE would present a viable alternative to a model that you can fit with METHOD=QUAD, the "Optimization Information" table can provide some insights. The table contains as its last entry the number of quadrature points determined by PROC GLIMMIX to yield a sufficiently accurate approximation of the log likelihood (at the starting values). In many cases, a single quadrature node is sufficient, in which case the estimates are identical to those of METHOD=LAPLACE.

Aspects Common to Adaptive Quadrature and Laplace Approximation

Estimated Precision of Estimates

Denote as **H** the second derivative matrix

$$\mathbf{H} = -\frac{\partial^2 \log\{L(\boldsymbol{\beta}, \boldsymbol{\theta}\ \widehat{\boldsymbol{\gamma}})\}}{\partial[\boldsymbol{\beta}, \boldsymbol{\theta}]\partial[\boldsymbol{\beta}', \boldsymbol{\theta}']}$$

evaluated at the converged solution of the optimization process. Partition its inverse as

$$\mathbf{H}^{-1} = \begin{bmatrix} \mathbf{C}(\boldsymbol{\beta}, \boldsymbol{\beta}) & \mathbf{C}(\boldsymbol{\beta}, \boldsymbol{\theta}) \\ \mathbf{C}(\boldsymbol{\theta}, \boldsymbol{\beta}) & \mathbf{C}(\boldsymbol{\theta}, \boldsymbol{\theta}) \end{bmatrix}$$

For METHOD=LAPLACE and METHOD=QUAD, the GLIMMIX procedure computes **H** by finite forward differences based on the analytic gradient of $\log\{L(\boldsymbol{\beta}, \boldsymbol{\theta}\ \widehat{\boldsymbol{\gamma}})\}$. The partition $\mathbf{C}(\boldsymbol{\theta}, \boldsymbol{\theta})$ serves as the asymptotic covariance matrix of the covariance parameter estimates (ASYCOV option in the PROC GLIMMIX statement). The standard errors reported in the "Covariance Parameter Estimates" table are based on the diagonal entries of this partition.

If you request an empirical standard error matrix with the EMPIRICAL option in the PROC GLIMMIX statement, a likelihood-based sandwich estimator is computed based on the subject-specific gradients of the Laplace or quadrature approximation. The sandwich estimator then replaces $\mathbf{H}^{-1}$ in calculations following convergence.

To compute the standard errors and prediction standard errors of linear combinations of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, PROC GLIMMIX forms an approximate prediction variance matrix for $[\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\gamma}}]'$ from

$$\mathbf{P} = \begin{bmatrix} \mathbf{H}^{-1} & \mathbf{H}^{-1}\left(\frac{\partial\widehat{\boldsymbol{\gamma}}}{\partial[\boldsymbol{\beta},\boldsymbol{\theta}]}\right) \\ \left(\frac{\partial\widehat{\boldsymbol{\gamma}}}{\partial[\boldsymbol{\beta}',\boldsymbol{\theta}']}\right)\mathbf{H}^{-1} & \boldsymbol{\Gamma}^{-1} + \left(\frac{\partial\widehat{\boldsymbol{\gamma}}}{\partial[\boldsymbol{\beta}',\boldsymbol{\theta}']}\right)\mathbf{H}^{-1}\left(\frac{\partial\widehat{\boldsymbol{\gamma}}}{\partial[\boldsymbol{\beta},\boldsymbol{\theta}]}\right) \end{bmatrix}$$

where $\boldsymbol{\Gamma}$ is the second derivative matrix from the $\boldsymbol{\gamma}$ suboptimization that maximizes $f(\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\theta}; \boldsymbol{\gamma})$ for given values of $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$. The prediction variance submatrix for the random effects is based on approximating the conditional mean squared error of prediction as in Booth and Hobert (1998). Note that even in the normal linear mixed model, the approximate conditional prediction standard errors are not identical to the prediction standard errors you obtain by inversion of the mixed model equations.

Conditional Fit and Output Statistics

When you estimate the parameters of a mixed model by Laplace approximation or quadrature, the GLIMMIX procedure displays fit statistics related to the marginal distribution as well as the conditional distribution $p(\mathbf{y}|\widehat{\boldsymbol{\gamma}}, \widehat{\boldsymbol{\beta}}, \widehat{\phi})$. The ODS name of the "Conditional Fit Statistics" table is CondFitStatistics. Because the marginal likelihood is approximated numerically for these methods, statistics based on the marginal distribution are not available. Instead of the generalized Pearson chi-square statistic in the "Fit Statistics" table, PROC GLIMMIX reports the Pearson statistic of the conditional distribution in the "Conditional Fit Statistics" table.

The unavailability of the marginal distribution also affects the set of output statistics that can be produced with METHOD=LAPLACE and METHOD=QUAD. Output statistics and statistical graphics that depend on the marginal variance of the data are not available with these estimation methods.

User-Defined Variance Function

If you provide your own variance function, PROC GLIMMIX generally assumes that the (conditional) distribution of the data is unknown. Laplace or quadrature estimation would then not be possible. When you specify a variance function with METHOD=LAPLACE or METHOD=QUAD, the procedure assumes that the conditional distribution is normal. For example, consider the following statements to fit a mixed model to count data:

```
proc glimmix method=laplace;
   class sub;
   _variance_ = _phi_*_mu_;
   model count = x / s link=log;
   random int / sub=sub;
run;
```

The variance function and the link suggest an overdispersed Poisson model. The Poisson distribution cannot accommodate the extra scale parameter _PHI_, however. In this situation, the GLIMMIX procedure fits a mixed model with random intercepts, log link function, and variance function $\phi\mu$, assuming that the count variable is normally distributed, given the random effects.

Starting Values

Good starting values for the fixed effects and covariance parameters are important for Laplace and quadrature methods because the process commences with a suboptimization in which the empirical Bayes estimates of the random effects must be obtained before the optimization can get under way. Furthermore, the starting values are important for the adaptive choice of the number of quadrature points.

If you choose METHOD=LAPLACE or METHOD=QUAD and you do not provide starting values for the covariance parameters through the PARMS statement, the GLIMMIX procedure determines starting values in the following steps.

1. A GLM is fit initially to obtain starting values for the fixed-effects parameters. No output is produced from this stage. The number of initial iterations of this GLM fit can be controlled with the INITITER= option in the PROC GLIMMIX statement. You can suppress this step with the NOINITGLM option in the PROC GLIMMIX statement.

2. Given the fixed-effects estimates, starting values for the covariance parameters are computed by a MIVQUE0 step (Goodnight 1978a).

3. For METHOD=QUAD you can follow these steps with several pseudo-likelihood updates to improve on the estimates and to obtain solutions for the random effects. The number of pseudo-likelihood steps is controlled by the INITPL= suboption of METHOD=QUAD.

4. For METHOD=QUAD, if you do not specify the number of quadrature points with the suboptions of the METHOD option, the GLIMMIX procedure attempts to determine a sufficient number of points adaptively as follows. Suppose that N_q denotes the number of nodes in each dimension. If N_{min} and N_{max} denote the values from the QMIN= and QMAX= suboptions, respectively, the sequence for values less than 11 is constructed in increments of 2 starting at N_{min}. Values greater than 11 are incremented in steps of r. The default value is $r = 10$. The default sequence, without specifying the QMIN=, QMAX=, or QFAC= option, is thus $1, 3, 5, 7, 9, 11, 21, 31$. If the relative difference of the log-likelihood approximation for two values in the sequence is less than the QTOL=t value (default $t = 0.0001$), the GLIMMIX procedure uses the lesser value for N_q in the subsequent optimization. If the relative difference does not fall below the tolerance t for any two subsequent values in the sequence, no estimation takes place.

Notes on Bias of Estimators

Generalized linear mixed models are nonlinear models, and the estimation techniques rely on approximations to the log likelihood or approximations of the model. It is thus not surprising that the estimates of the covariance parameters and the fixed effects are usually not unbiased. Whenever estimates are biased, questions arise about the magnitude of the bias, its dependence on other model quantities, and the order of the bias. The order is important because it determines how quickly the bias vanishes while some aspect of the data increases. Typically, studies of asymptotic properties in models for hierarchical data suppose that the number of subjects (clusters) tends to infinity while the size of the clusters is held constant or grows at a particular rate. Note that asymptotic results so established do not extend to designs with fully crossed random effects, for example.

The following paragraphs summarize some important findings from the literature regarding the bias in covariance parameter and fixed-effects estimates with pseudo-likelihood, Laplace, and adaptive quadrature methods. The remarks draw in particular on results in Breslow and Lin (1995); Lin and Breslow (1996); Pinheiro and Chao (2006). Breslow and Lin (1995); Lin and Breslow (1996) study the "worst case" scenario of binary responses in a matched-pairs design. Their models have a variance component structure, comprising either a single variance component (a subject-specific random intercept; Breslow and Lin 1995) or a diagonal $\mathbf{G}$ matrix (Lin and Breslow 1996). They study the bias in the estimates of the fixed-effects $\boldsymbol{\beta}$ and the covariance parameters $\boldsymbol{\theta}$ when the variance components are near the origin and for a canonical link function.

The matched-pairs design gives rise to a generalized linear mixed model with a cluster (subject) size of 2. Recall that the pseudo-likelihood methods rely on a linearization and a probabilistic assumption that the pseudo-data so obtained follow a normal linear mixed model. Obviously, it is difficult to imagine how the subject-specific (conditional) distribution would follow a normal linear mixed models with binary data in a cluster size of 2. The bias in the pseudo-likelihood estimator of $\boldsymbol{\beta}$ is of order $||\boldsymbol{\theta}||$. The bias for the Laplace estimator of $\boldsymbol{\beta}$ is of smaller magnitude; its asymptotic bias has order $||\boldsymbol{\theta}||^2$.

The Laplace methods and the pseudo-likelihood method produce biased estimators of the variance component θ for the model considered in Breslow and Lin (1995). The order of the asymptotic bias for both estimation methods is θ, as θ approaches zero. Breslow and Lin (1995) comment on the fact that even with matched pairs, the bias vanishes very quickly in the binomial setting. If the conditional mean in the two groups is equal to 0.5, then the asymptotic bias factor of the pseudo-likelihood estimator is $1 - 1/(2n)$, where n is the binomial denominator. This term goes to 1 quickly as n increases. This result underlines the importance of grouping binary observations into binomial responses whenever possible.

The results of Breslow and Lin (1995) and Lin and Breslow (1996) are echoed in the simulation study in Pinheiro and Chao (2006). These authors also consider adaptive quadrature in models with nested, hierarchical, random effects and show that adaptive quadrature with a sufficient number of nodes leads to nearly unbiased—or least biased—estimates. Their results also show that results for binary data cannot so easily be ported to other distributions. Even with a cluster size of 2, the pseudo-likelihood estimates of fixed effects and covariance parameters are virtually unbiased in their simulation of a Poisson GLMM. Breslow and Lin (1995) and Lin and Breslow (1996) "eschew" the residual PL version (METHOD=RSPL) over the maximum likelihood form (METHOD=MSPL). Pinheiro and Chao (2006) consider both forms in their simulation study. As expected, the residual form shows less bias than the MSPL form, for the same reasons REML estimation leads to less biased estimates compared to ML estimation in linear mixed models. However, the gain is modest; see, for example, Table 1 in Pinheiro and Chao (2006). When the variance components are small, there are a sufficient number of observations per cluster, and a reasonable number of clusters, then pseudo-likelihood methods for binary data are very useful—they provide a computationally expedient alternative to numerical integration, and they allow the incorporation of R-side covariance structure into the model. Because many group randomized trials involve many observations per group and small random-effects variances, Murray, Varnell, and Blitstein (2004) refer to the use of conditional models for trials that have a binary outcome as an "overreaction."

Pseudo-likelihood Estimation for Weighted Multilevel Models

Multilevel models provide a flexible and powerful tool for the analysis of data that are observed in nested units at multiple levels. A multilevel model is a special case of generalized linear mixed models that can be handled by the GLIMMIX procedure. In proc GLIMMIX, the SUBJECT= option in the RANDOM statement identifies the clustering structure for the random effects. When the subjects of the multiple RANDOM statements are nested, the model is a multilevel model and each RANDOM statement corresponds to one level.

Using a pseudo-maximum-likelihood approach, you can extend the multilevel model framework to accommodate weights at different levels. Such an approach is very useful in analyzing survey data that arise from multistage sampling. In these sampling designs, survey weights are often constructed to account for unequal sampling probabilities, nonresponse adjustments, and poststratification.

The following survey example from a three-stage sampling design illustrates the use of multiple levels of weighting in a multilevel model. Extending this example to models that have more than three levels is straightforward. Let $i = 1, \ldots, n^{(3)}$, $j = 1, \ldots n_i^{(2)}$, and $k = 1, \ldots n_j^{(1)}$ denote the indices of units at level

3, level 2, and level 1, respectively. Let superscript (l) denote the lth level and $n^{(l)}$ denote the number of level-l units in the sample. Assume that the first-stage cluster (level-3 unit) i is selected with probability π_i; the second-stage cluster (level-2 unit) j is selected with probability $\pi_{j|i}$, given that the first-stage cluster i is already selected in the sample; and the third-stage unit (level-1 unit) k is selected with probability $\pi_{k|ij}$, given that the second-stage cluster j within the first-stage cluster i is already selected in the sample.

If you use the inverse selection probability weights $w_{j|i} = 1/\pi_{j|i}$, $w_{k|ij} = 1/\pi_{k|ij}$, the conditional log likelihood contribution of the first-stage cluster i is

$$\log\left(p(y_i|\gamma_i^{(2)}, \gamma_i^{(3)})\right) = \sum_{j=1}^{n_i^{(2)}} w_{j|i} \sum_{k=1}^{n_j^{(1)}} w_{k|ij} \log\left(p(y_{ijk}|\gamma_{ij}^{(2)}\gamma_i^{(3)})\right)$$

where $\gamma_{ij}^{(2)}$ is the random-effects vector for the jth second-stage cluster, $\gamma_i^{(2)} = (\gamma'_{i1}, \gamma'_{i2}, \ldots, \gamma'_{in_i^{(2)}})'$, and $\gamma_i^{(3)}$ is the random-effects vector for the ith first-stage cluster.

As with unweighted multilevel models, the adaptive quadrature method is used to compute the pseudo-likelihood of the first-stage cluster i:

$$p(y_i) = \int p(y_i|\gamma_i^{(2)}, \gamma_i^{(3)}) p(\gamma_i^{(2)}) p(\gamma_i^{(3)}) d(\gamma_i^{(2)}) d(\gamma_i^{(3)})$$

The total log pseudo-likelihood is

$$\log\left(p(y)\right) = \sum_{i=1}^{n^{(3)}} w_i \log\left(p(y_i)\right)$$

where $w_i = 1/\pi_i$.

To illustrate weighting in a multilevel model, consider the following data set. In these simulated data, the response y is a Poisson-distributed count, w3 is the weight for the first-stage clusters, w2 is the weight for the second-stage clusters, and w1 is the weight for the observation-level units.

```
data d;
   input A w3 AB w2 w1 y x;
   datalines;
1    6.1     1    5.3     7.1    56    -.214
1    6.1     1    5.3     3.9    41    0.732
1    6.1     2    7.3     6.3    50    0.372
1    6.1     2    7.3     3.9    36    -.892
1    6.1     3    4.6     8.4    39    0.424
1    6.1     3    4.6     6.3    35    -.200
2    8.5     1    4.8     7.4    30    0.868
2    8.5     1    4.8     6.7    25    0.110
2    8.5     2    8.4     3.5    36    0.004
2    8.5     2    8.4     4.1    28    0.755
2    8.5     3    .80     3.8    33    -.600
2    8.5     3    .80     7.4    30    -.525
3    9.5     1    8.2     6.7    32    -.454
3    9.5     1    8.2     7.1    24    0.458
3    9.5     2     11     4.8    31    0.162
3    9.5     2     11     7.9    27    1.099
```

```
3    9.5    3    3.9    3.8    15    -1.57
3    9.5    3    3.9    5.5    19    -.448
4    4.5    1    8.0    5.7    30    -.468
4    4.5    1    8.0    2.9    25    0.890
4    4.5    2    6.0    5.0    35    0.635
4    4.5    2    6.0    3.0    30    0.743
4    4.5    3    6.8    7.3    17    -.015
4    4.5    3    6.8    3.1    18    -.560
;
```

You can use the following statements to fit a weighted three-level model:

```
proc glimmix  method=quadrature empirical=classical;
   class A AB;
   model y = x / dist=poisson link=log obsweight=w1;
   random int  / subject=A weight=w3;
   random int  / subject=AB(A) weight=w2;
run;
```

The SUBJECT= option in the first and second RANDOM statements specifies the first-stage and second-stage clusters A and AB(A), respectively. The OBSWEIGHT= option in the MODEL statement specifies the variable for the weight at the observation level. The WEIGHT= option in the RANDOM statement specifies the variable for the weight at the level that is specified by the SUBJECT= option.

For inference about fixed effects and variance that are estimated by pseudo-likelihood, you can use the empirical (sandwich) variance estimators. For weighted multilevel models, the only empirical estimator available in PROC GLIMMIX is EMPIRICAL=CLASSICAL. The EMPIRICAL=CLASSICAL variance estimator can be described as follows.

Let $\alpha = (\beta', \theta')'$, where β is vector of the fixed-effects parameters and θ is the vector of covariance parameters. For an L-level model, Rabe-Hesketh and Skrondal (2006) show that the gradient can be written as a weighted sum of the gradients of the top-level units:

$$\sum_{i=1}^{n^{(L)}} w_i \frac{\partial \log(p(y_i;\alpha))}{\partial \alpha} \equiv \sum_{i=1}^{n^{(L)}} S_i(\alpha)$$

where $n^{(L)}$ is the number of level-L units and $S_i(\alpha)$ is the weighted score vector of the top-level unit i. The estimator of the "meat" of the sandwich estimator can be written as

$$J = \frac{n^{(L)}}{n^{(L)}-1} \sum_{i=1}^{n^{(L)}} S_i(\hat{\alpha}) S_i(\hat{\alpha})'$$

The empirical estimator of the covariance matrix of $\hat{\alpha}$ can be constructed as

$$H(\hat{\alpha})^{-1} J H(\hat{\alpha})^{-1}$$

where $H(\alpha)$ is the second derivative matrix of the log pseudo-likelihood with respect to α.

The covariance parameter estimators that are obtained by the pseudo-maximum-likelihood method can be biased when the sample size is small. Pfeffermann et al. (1998) and Rabe-Hesketh and Skrondal (2006) discuss two weight-scaling methods for reducing the biases of the covariance parameter estimators in a two-level model. To derive the scaling factor λ for a two-level model, let n_i denote the number of level-1

units in the level-2 unit i and let $w_{j|i}$ denote the weight of the jth level-1 unit in level-2 unit i. The first method computes an "apparent" cluster size as the "effective" sample size:

$$\sum_{j=1}^{n_i} \lambda w_{j|i} = \frac{(\sum_{j=1}^{n_i} w_{j|i})^2}{\sum_{j=1}^{n_i} w_{j|i}^2}$$

Therefore the scale factor is

$$\lambda = \frac{\sum_{j=1}^{n_i} w_{j|i}}{\sum_{j=1}^{n_i} w_{j|i}^2}$$

The second method sets the apparent cluster size equal to the actual cluster size so that the scale factor is

$$\lambda = \frac{n_i}{\sum_{j=1}^{n_i} w_{j|i}}$$

PROC GLIMMIX uses the weights provided in the data set directly. To use the scaled weights, you need to provide them in the data set.

GLM Mode or GLMM Mode

The GLIMMIX procedure uses two basic modes of parameter estimation, and it can be important for you to understand the differences between the two modes.

In GLM mode, the data are never correlated and there can be no G-side random effects. Typical examples are logistic regression and normal linear models. When you fit a model in GLM mode, the METHOD= option in the PROC GLIMMIX statement has no effect. PROC GLIMMIX estimates the parameters of the model by maximum likelihood, (restricted) maximum likelihood, or quasi-likelihood, depending on the distributional properties of the model (see the section "Default Estimation Techniques" on page 3981). The "Model Information" table tells you which estimation method was applied. In GLM mode, the individual observations are considered the sampling units. This has bearing, for example, on how sandwich estimators are computed (see the EMPIRICAL option and the section "Empirical Covariance ("Sandwich") Estimators" on page 3952).

In GLMM mode, the procedure assumes that the model contains random effects or possibly correlated errors, or that the data have a clustered structure. PROC GLIMMIX then estimates the parameters by using the techniques specified in the METHOD= option in the PROC GLIMMIX statement.

In general, adding one overdispersion parameter to a generalized linear model does not trigger GLMM mode. For example, the model that is defined by the following statements is fit in GLM mode:

```
proc glimmix;
   model y = x1 x2 / dist=poisson;
   random _residual_;
run;
```

The parameters of the fixed effects are estimated by maximum likelihood, and the covariance matrix of the fixed-effects parameters is adjusted by the overdispersion parameter.

In a model that contains uncorrelated data, you can trigger GLMM mode by specifying the SUBJECT= or GROUP= option in the RANDOM statement. For example, the following statements fit the model by using the residual pseudo-likelihood algorithm:

```
proc glimmix;
   class id;
   model y = x1 x2 / dist=poisson;
   random _residual_ / subject=id;
run;
```

If in doubt, you can determine whether a model was fit in GLM mode or GLMM mode. In GLM mode the "Covariance Parameter Estimates" table is not produced. Scale and dispersion parameters in the model appear in the "Parameter Estimates" table.

Statistical Inference for Covariance Parameters

The Likelihood Ratio Test

The likelihood ratio test (LRT) compares the likelihoods of two models where parameter estimates are obtained in two parameter spaces, the space Ω and the restricted subspace Ω_0. In the GLIMMIX procedure, the full model defines Ω and the *test-specification* in the COVTEST statement determines the null parameter space Ω_0. The likelihood ratio procedure consists of the following steps (see, for example, Bickel and Doksum 1977, p. 210):

1. Find the estimate $\widehat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta} \in \Omega$. Compute the likelihood $L(\widehat{\boldsymbol{\theta}})$.
2. Find the estimate $\widehat{\boldsymbol{\theta}}_0$ of $\boldsymbol{\theta} \in \Omega_0$. Compute the likelihood $L(\widehat{\boldsymbol{\theta}}_0)$.
3. Form the likelihood ratio

$$\overline{\lambda} = \frac{L(\widehat{\boldsymbol{\theta}})}{L(\widehat{\boldsymbol{\theta}}_0)}$$

4. Find a function $f\left(\overline{\lambda}\right)$ that has a known distribution. $f(\cdot)$ serves as the test statistic for the likelihood ratio test.

Please note the following regarding the implementation of these steps in the COVTEST statement of the GLIMMIX procedure.

- The function $f(\cdot)$ in step 4 is always taken to be

$$\lambda = 2\log\left\{\overline{\lambda}\right\}$$

  which is twice the difference between the log likelihoods for the full model and the model under the COVTEST restriction.
- For METHOD=RSPL and METHOD=RMPL, the test statistic is based on the restricted likelihood.
- For GLMMs involving pseudo-data, the test statistics are based on the pseudo-likelihood or the restricted pseudo-likelihood and are based on the final pseudo-data.

- The parameter space Ω for the full model is typically not an unrestricted space. The GLIMMIX procedure imposes boundary constraints for variance components and scale parameters, for example. The specification of the subspace Ω_0 must be consistent with these full-model constraints; otherwise the test statistic λ does not have the needed distribution. You can remove the boundary restrictions with the NOBOUND option in the PROC GLIMMIX statement or the NOBOUND option in the PARMS statement.

One- and Two-Sided Testing, Mixture Distributions

Consider testing the hypothesis $H_0\colon \theta_i = 0$. If Ω is the open interval $(0, \infty)$, then only a one-sided alternative hypothesis is meaningful,

$$H_0\colon \theta_i = 0 \quad H_a\colon \theta_i > 0$$

This is the appropriate set of hypotheses, for example, when θ_i is the variance of a G-side random effect. The positivity constraint on Ω is required for valid conditional and marginal distributions of the data. Verbeke and Molenberghs (2003) refer to this situation as the constrained case.

However, if one focuses on the validity of the marginal distribution alone, then negative values for θ_i might be permissible, provided that the marginal variance remains positive definite. In the vernacular or Verbeke and Molenberghs (2003), this is the unconstrained case. The appropriate alternative hypothesis is then two-sided,

$$H_0\colon \theta_i = 0 \quad H_a\colon \theta_i \neq 0$$

Several important issues are connected to the choice of hypotheses. The GLIMMIX procedure by default imposes constraints on some covariance parameters. For example, variances and scale parameters have a lower bound of 0. This implies a constrained setting with one-sided alternatives. If you specify the NOBOUND option in the PROC GLIMMIX statement, or the NOBOUND option in the PARMS statement, the boundary restrictions are lifted from the covariance parameters and the GLIMMIX procedure takes an unconstrained stance in the sense of Verbeke and Molenberghs (2003). The alternative hypotheses for variance components are then two-sided.

When $H_0\colon \theta_i = 0$ and $\Omega = (0, \infty)$, the value of θ_i under the null hypothesis is on the boundary of the parameter space. The distribution of the likelihood ratio test statistic λ is then nonstandard. In general, it is a mixture of distributions, and in certain special cases, it is a mixture of central chi-square distributions. Important contributions to the understanding of the asymptotic behavior of the likelihood ratio and score test statistic in this situation have been made by, for example, Self and Liang (1987); Shapiro (1988); Silvapulle and Silvapulle (1995). Stram and Lee (1994, 1995) applied the results of Self and Liang (1987) to likelihood ratio testing in the mixed model with uncorrelated errors. Verbeke and Molenberghs (2003) compared the score and likelihood ratio tests in random effects models with unstructured **G** matrix and provide further results on mixture distributions.

The GLIMMIX procedure recognizes the following special cases in the computation of p-values ($\widehat{\lambda}$ denotes the realized value of the test statistic). Notice that the probabilities of general chi-square mixture distributions do not equal linear combination of central chi-square probabilities (Davis 1977; Johnson, Kotz, and Balakrishnan 1994, Section 18.8).

1. ν parameters are tested, and neither parameters specified under H_0 nor nuisance parameters are on the boundary of the parameters space (Case 4 in Self and Liang 1987). The p-value is computed by the classical result:

$$p = \Pr\left(\chi^2_\nu \geq \widehat{\lambda}\right)$$

2. One parameter is specified under H_0 and it falls on the boundary. No other parameters are on the boundary (Case 5 in Self and Liang 1987).

$$p = \begin{cases} 1 & \widehat{\lambda} = 0 \\ 0.5\,\mathrm{Pr}\left(\chi_1^2 \geq \widehat{\lambda}\right) & \widehat{\lambda} > 0 \end{cases}$$

   Note that this implies a 50:50 mixture of a χ_0^2 and a χ_1^2 distribution. This is also Case 1 in Verbeke and Molenberghs (2000, p. 69).

3. Two parameters are specified under H_0, and one falls on the boundary. No nuisance parameters are on the boundary (Case 6 in Self and Liang 1987).

$$p = 0.5\,\mathrm{Pr}\left(\chi_1^2 \geq \widehat{\lambda}\right) + 0.5\,\mathrm{Pr}\left(\chi_2^2 \geq \widehat{\lambda}\right)$$

   A special case of this scenario is the addition of a random effect to a model with a single random effect and unstructured covariance matrix (Case 2 in Verbeke and Molenberghs 2000, p. 70).

4. Removing j random effects from $j + k$ uncorrelated random effects (Verbeke and Molenberghs 2003).

$$p = 2^{-j} \sum_{i=0}^{j} \binom{j}{i} \mathrm{Pr}\left(\chi_i^2 \geq \widehat{\lambda}\right)$$

   Note that this case includes the case of testing a single random effects variance against zero, which leads to a 50:50 mixture of a χ_0^2 and a χ_1^2 as in 2.

5. Removing a random effect from an unstructured **G** matrix (Case 3 in Verbeke and Molenberghs 2000, p. 71).

$$p = 0.5\,\mathrm{Pr}\left(\chi_k^2 \geq \widehat{\lambda}\right) + 0.5\,\mathrm{Pr}\left(\chi_{k-1}^2 \geq \widehat{\lambda}\right)$$

   where k is the number of random effects (columns of **G**) in the full model. Case 5 in Self and Liang (1987) describes a special case.

When the GLIMMIX procedure determines that estimates of nuisance parameters (parameters not specified under H_0) fall on the boundary, no mixture results are computed.

You can request that the procedure not use mixtures with the CLASSICAL option in the COVTEST statement. If mixtures are used, the Note column of the "Likelihood Ratio Tests of Covariance Parameters" table contains the "MI" entry. The "DF" entry is used when PROC GLIMMIX determines that the standard computation of p-values is appropriate. The "–" entry is used when the classical computation was used because the testing and model scenario does not match one of the special cases described previously.

Handling the Degenerate Distribution

Likelihood ratio testing in mixed models invariably involves the chi-square distribution with zero degrees of freedom. The χ^2_0 random variable is degenerate at 0, and it occurs in two important circumstances. First, it is a component of mixtures, where typically the value of the test statistic is not zero. In that case, the contribution of the χ^2_0 component of the mixture to the p-value is nil. Second, a degenerate distribution of the test statistic occurs when the null model is identical to the full model—for example, if you test a hypothesis that does not impose any (new) constraints on the parameter space. The following statements test whether the $\mathbf{R}$ matrix in a variance component model is diagonal:

```
proc glimmix;
   class a b;
   model y = a;
   random b a*b;
   covtest diagR;
run;
```

Because no R-side covariance structure is specified (all random effects are G-side effects), the $\mathbf{R}$ matrix is diagonal in the full model and the COVTEST statement does not impose any further restriction on the parameter space. The likelihood ratio test statistic is zero. The GLIMMIX procedure computes the p-value as the probability to observe a value at least as large as the test statistic under the null hypothesis. Hence,

$$p = \Pr(\chi^2_0 \geq 0) = 1$$

Wald Versus Likelihood Ratio Tests

The Wald test and the likelihood ratio tests are asymptotic tests, meaning that the distribution from which p-values are calculated for a finite number of samples draws on the distribution of the test statistic as the sample size grows to infinity. The Wald test is a simple test that is easy to compute based only on parameter estimates and their (asymptotic) standard errors. The likelihood ratio test, on the other hand, requires the likelihoods of the full model and the model reduced under H_0. It is computationally more demanding, but also provides the asymptotically more powerful and reliable test. The likelihood ratio test is almost always preferable to the Wald test, unless computational demands make it impractical to refit the model.

Confidence Bounds Based on Likelihoods

Families of statistical tests can be inverted to produce confidence limits for parameters. The confidence region for parameter θ is the set of values for which the corresponding test fails to reject $H\colon \theta = \theta_0$. When parameters are estimated by maximum likelihood or a likelihood-based technique, it is natural to consider the likelihood ratio test statistic for H in the test inversion. When there are multiple parameters in the model, however, you need to supply values for these nuisance parameters during the test inversion as well.

In the following, suppose that $\boldsymbol{\theta}$ is the covariance parameter vector and that one of its elements, θ, is the parameter of interest for which you want to construct a confidence interval. The other elements of $\boldsymbol{\theta}$ are collected in the nuisance parameter vector $\boldsymbol{\theta}_2$. Suppose that $\widehat{\boldsymbol{\theta}}$ is the estimate of $\boldsymbol{\theta}$ from the overall optimization and that $L(\widehat{\boldsymbol{\theta}})$ is the likelihood evaluated at that estimate. If estimation is based on pseudo-data, then $L(\widehat{\boldsymbol{\theta}})$ is the pseudo-likelihood based on the final pseudo-data. If estimation uses a residual (restricted) likelihood, then L denotes the restricted maximum likelihood and $\widehat{\boldsymbol{\theta}}$ is the REML estimate.

Profile Likelihood Bounds

The likelihood ratio test statistic for testing $H: \theta = \theta_0$ is

$$2\left\{\log\left\{L(\widehat{\boldsymbol{\theta}})\right\} - \log\left\{L(\theta_0, \widehat{\boldsymbol{\theta}}_2)\right\}\right\}$$

where $\widehat{\boldsymbol{\theta}}_2$ is the likelihood estimate of $\boldsymbol{\theta}_2$ under the restriction that $\theta = \theta_0$. To invert this test, a function is defined that returns the maximum likelihood for a fixed value of θ by seeking the maximum over the remaining parameters. This function is termed the profile likelihood (Pawitan 2001, Ch. 3.4),

$$\lambda_p = L(\boldsymbol{\theta}_2|\widetilde{\theta}) = \sup_{\boldsymbol{\theta}_2} L(\widetilde{\theta}, \boldsymbol{\theta}_2)$$

In computing λ_p, θ is fixed at $\widetilde{\theta}$ and $\boldsymbol{\theta}_2$ is estimated. In mixed models, this step typically requires a separate, iterative optimization to find the estimate of $\boldsymbol{\theta}_2$ while θ is held fixed. The $(1-\alpha) \times 100\%$ profile likelihood confidence interval for θ is then defined as the set of values for $\widetilde{\theta}$ that satisfy

$$2\left\{\log\left\{L(\widehat{\boldsymbol{\theta}})\right\} - \log\left\{L(\boldsymbol{\theta}_2|\widetilde{\theta})\right\}\right\} \le \chi^2_{1,(1-\alpha)}$$

The GLIMMIX procedure seeks the values $\widetilde{\theta}_l$ and $\widetilde{\theta}_u$ that mark the endpoints of the set around $\widehat{\theta}$ that satisfy the inequality. The values ($\widetilde{\theta}_l$ and $\widetilde{\theta}_u$) are then called the $(1-\alpha) \times 100\%$ confidence bounds for θ. Note that the GLIMMIX procedure assumes that the confidence region is not disjoint and relies on the convexity of $L(\widehat{\boldsymbol{\theta}})$.

It is not always possible to find values $\widetilde{\theta}_l$ and $\widetilde{\theta}_u$ that satisfy the inequalities. For example, when the parameter space is $(0, \infty)$ and

$$2\left\{\log\left\{L(\widehat{\boldsymbol{\theta}})\right\} - \log\left\{L(\boldsymbol{\theta}_2|0)\right\}\right\} > \chi^2_{1,(1-\alpha)}$$

a lower bound cannot be found at the desired confidence level. The GLIMMIX procedure reports the right-tail probabilities that are achieved by the underlying likelihood ratio statistic separately for lower and upper bounds.

Effect of Scale Parameter

When a scale parameter ϕ is eliminated from the optimization by profiling from the likelihood, some parameters might be expressed as ratios with ϕ in the optimization. This is the case, for example, in variance component models. The profile likelihood confidence bounds are reported on the scale of the parameter in the overall optimization. In case parameters are expressed as ratios with ϕ or functions of ϕ, the column RatioEstimate is added to the "Covariance Parameter Estimates" table. If parameters are expressed as ratios with ϕ and you want confidence bounds for the unscaled parameter, you can prevent profiling of ϕ from the optimization with the NOPROFILE option in the PROC GLIMMIX statement, or choose estimated likelihood confidence bounds with the TYPE=ELR suboption of the CL option in the COVTEST statement. Note that the NOPROFILE option is automatically in effect with METHOD=LAPLACE and METHOD=QUAD.

Estimated Likelihood Bounds

Computing profile likelihood ratio confidence bounds can be computationally expensive, because of the need to repeatedly estimate $\boldsymbol{\theta}_2$ in a constrained optimization. A computationally simpler method to construct confidence bounds from likelihood-based quantities is to use the estimated likelihood (Pawitan 2001, Ch. 10.7) instead of the profile likelihood. An estimated likelihood technique replaces the nuisance parameters in

the test inversion with some other estimate. If you choose the TYPE=ELR suboption of the CL option in the COVTEST statement, the GLIMMIX procedure holds the nuisance parameters fixed at the likelihood estimates. The estimated likelihood statistic for inversion is then

$$\lambda_e = L(\widetilde{\theta}, \widehat{\boldsymbol{\theta}}_2)$$

where $\widehat{\boldsymbol{\theta}}_2$ are the elements of $\widehat{\boldsymbol{\theta}}$ that correspond to the nuisance parameters. As the values of $\widetilde{\theta}$ are varied, no reestimation of $\boldsymbol{\theta}_2$ takes place. Although computationally more economical, estimated likelihood intervals do not take into account the variability associated with the nuisance parameters. Their coverage can be satisfactory if the parameter of interest is not (or only weakly) correlated with the nuisance parameters. Estimated likelihood ratio intervals can fall short of the nominal coverage otherwise.

Figure 52.11 depicts profile and estimated likelihood ratio intervals for the parameter σ in a two-parameter compound-symmetric model, $\boldsymbol{\theta} = [\sigma, \phi]'$, in which the correlation between the covariance parameters is small. The elliptical shape traces the set of values for which the likelihood ratio test rejects the hypothesis of equality with the solution. The interior of the ellipse is the "acceptance" region of the test. The solid and dashed lines depict the PLR and ELR confidence limits for σ, respectively. Note that both confidence limits intersect the ellipse and that the ELR interval passes through the REML estimate of ϕ. The PLR bounds are found as those points intersecting the ellipse, where ϕ equals the constrained REML estimate.

Figure 52.11 PLR and ELR Intervals, Small Correlation between Parameters

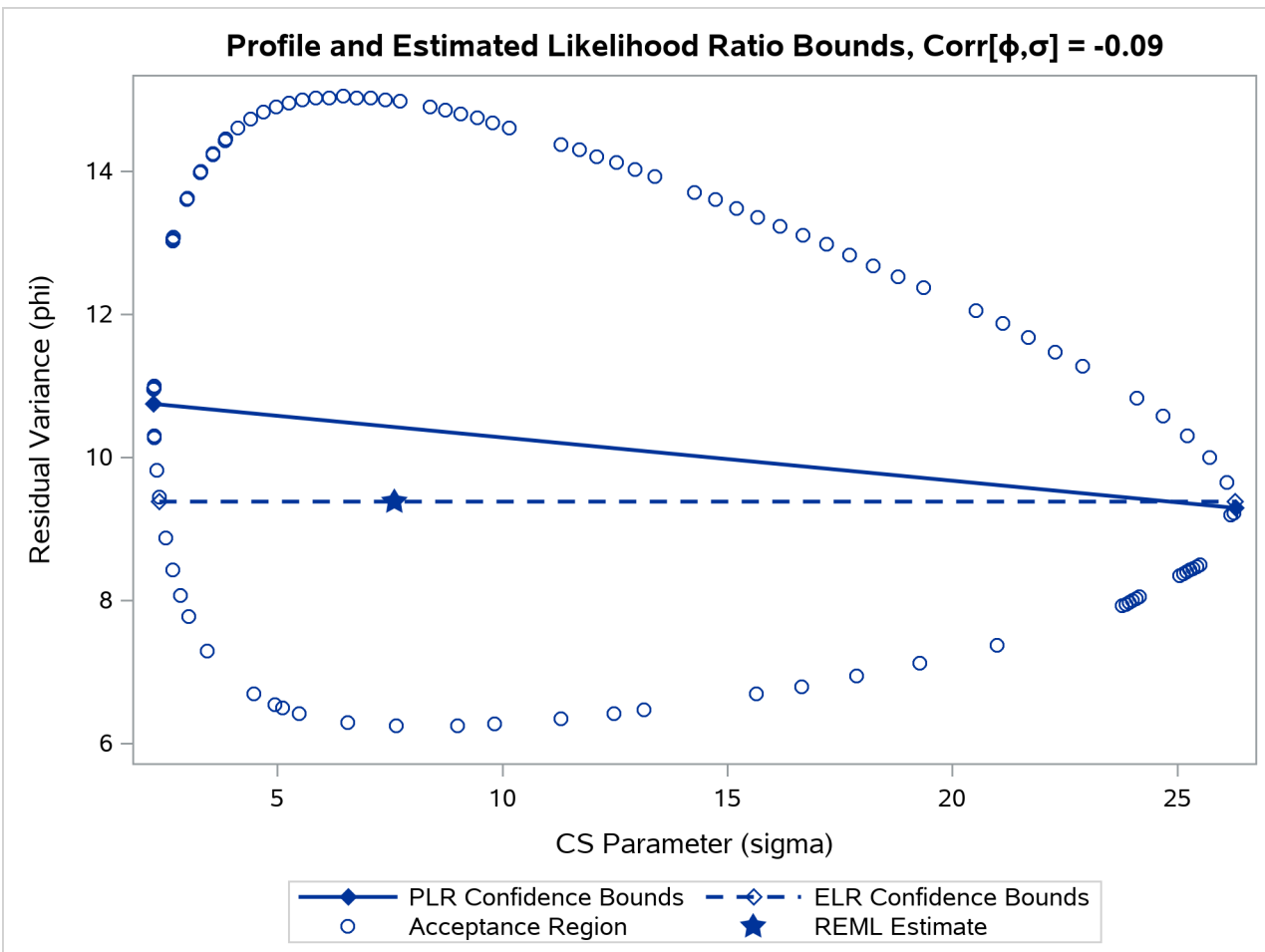

The major axes of the ellipse in Figure 52.11 are nearly aligned with the major axes of the coordinate system. As a consequence, the line connecting the PLR bounds passes close to the REML estimate in the full model. As a result, ELR bounds will be similar to PLR bounds. Figure 52.12 displays a different scenario, a two-parameter AR(1) covariance structure with a more substantial correlation between the AR(1) parameter (ρ) and the residual variance (ϕ).

Figure 52.12 PLR and ELR Intervals, Large Correlation between Parameters

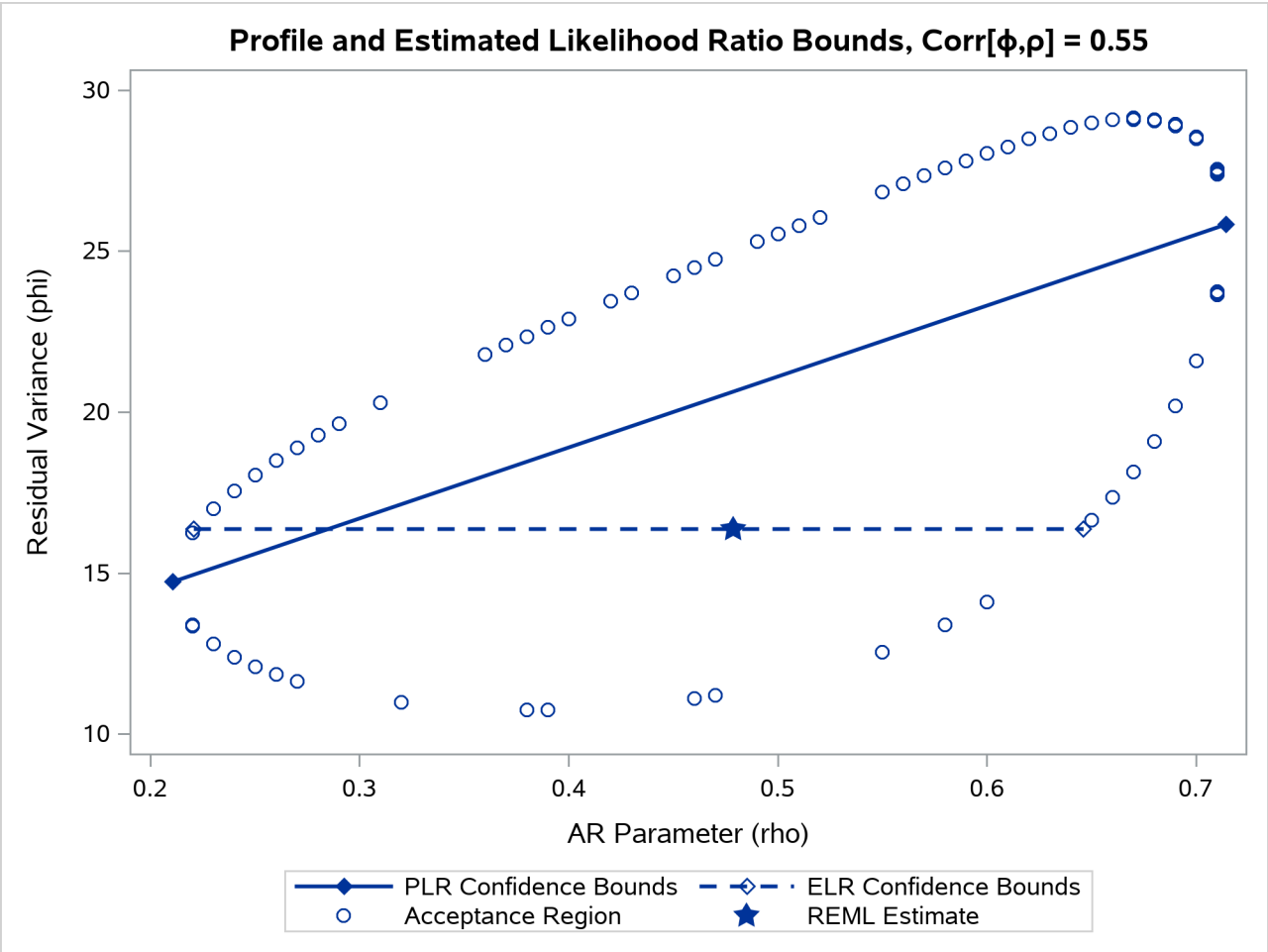

The correlation between the parameters yields an acceptance region whose major axes are not aligned with the axes of the coordinate system. The ELR bound for ρ passes through the REML estimate of ϕ from the full model and is much shorter than the PLR interval. The PLR interval aligns with the major axis of the acceptance region; it is the preferred confidence interval.

Degrees of Freedom Methods

Between-Within Degrees of Freedom Approximation

The DDFM=BETWITHIN option divides the residual degrees of freedom into between-subject and within-subject portions. PROC GLIMMIX then determines whether a fixed effect changes within any subject. If so, it assigns within-subject degrees of freedom to the effect; otherwise, it assigns the between-subject degrees of freedom to the effect (Schluchter and Elashoff 1990). If the GLIMMIX procedure does not process the data by subjects, the DDFM=BETWITHIN option has no effect. See the section "Processing by Subjects" on page 3957 for more information.

If multiple within-subject effects contain classification variables, the within-subject degrees of freedom are partitioned into components that correspond to the subject-by-effect interactions.

One exception to the preceding method is the case where you model only R-side covariation with an unstructured covariance matrix (TYPE=UN). In this case, all fixed effects are assigned the between-subject degrees of freedom to provide for better small-sample approximations to the relevant sampling distributions. The DDFM=BETWITHIN method is the default for models with only R-side random effects and a SUBJECT= option.

Containment Degrees of Freedom Approximation

The DDFM=CONTAIN method is carried out as follows: Denote the fixed effect in question as A and search the G-side random effect list for the effects that *syntactically* contain A. For example, the effect B(A) contains A, but the effect C does not, even if it has the same levels as B(A).

Among the random effects that contain A, compute their rank contributions to the $[\mathbf{X}\ \ \mathbf{Z}]$ matrix (in order). The denominator degrees of freedom that is assigned to A is the smallest of these rank contributions. If no effects are found, the denominator degrees of freedom for A is set equal to the residual degrees of freedom, $n - \mathrm{rank}[\mathbf{X}\ \ \mathbf{Z}]$. This choice of degrees of freedom is the same as for the tests performed for balanced split-plot designs and should be adequate for moderately unbalanced designs.

NOTE: If you have a $\mathbf{Z}$ matrix with a large number of columns, the overall memory requirements and the computing time after convergence can be substantial for the containment method. In this case, you might want to use a different degrees-of-freedom method, such as DDFM=RESIDUAL, DDFM=NONE, or DDFM=BETWITHIN.

Satterthwaite Degrees of Freedom Approximation

The DDFM=SATTERTHWAITE option in the MODEL statement requests that denominator degrees of freedom in t tests and F tests be computed according to a general Satterthwaite approximation.

The general Satterthwaite approximation computed in PROC GLIMMIX for the test

$$H: \mathbf{L}\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix} = \mathbf{0}$$

is based on the F statistic

$$F = \frac{\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}' \mathbf{L}'(\mathbf{L}\mathbf{C}\mathbf{L}')^{-1}\mathbf{L}\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}}{r}$$

where $r = \mathrm{rank}(\mathbf{LCL}')$ and $\mathbf{C}$ is the approximate variance matrix of $[\widehat{\boldsymbol{\beta}}', \widehat{\boldsymbol{\gamma}}' - \boldsymbol{\gamma}']'$. See the section "Estimated Precision of Estimates" on page 3926 and the section "Aspects Common to Adaptive Quadrature and Laplace Approximation" on page 3935.

The approximation proceeds by first performing the spectral decomposition $\mathbf{LCL}' = \mathbf{U}'\mathbf{DU}$, where $\mathbf{U}$ is an orthogonal matrix of eigenvectors and $\mathbf{D}$ is a diagonal matrix of eigenvalues, both of dimension $r \times r$. Define $\mathbf{b}_j$ to be the jth row of $\mathbf{UL}$, and let

$$\nu_j = \frac{2(D_j)^2}{\mathbf{g}_j'\mathbf{A}\mathbf{g}_j}$$

where D_j is the jth diagonal element of $\mathbf{D}$ and $\mathbf{g}_j$ is the gradient of $\mathbf{b}_j\mathbf{C}\mathbf{b}_j'$ with respect to $\boldsymbol{\theta}$, evaluated at $\widehat{\boldsymbol{\theta}}$. The matrix $\mathbf{A}$ is the asymptotic variance-covariance matrix of $\widehat{\boldsymbol{\theta}}$, which is obtained from the second derivative matrix of the likelihood equations. You can display this matrix with the ASYCOV option in the PROC GLIMMIX statement.

Finally, let

$$E = \sum_{j=1}^{r} \frac{\nu_j}{\nu_j - 2} I(\nu_j > 2)$$

where the indicator function eliminates terms for which $\nu_j \leq 2$. The degrees of freedom for F are then computed as

$$\nu = \frac{2E}{E - \mathrm{rank}(\mathbf{L})}$$

provided $E > r$; otherwise ν is set to 0.

In the one-dimensional case, when PROC GLIMMIX computes a t test, the Satterthwaite degrees of freedom for the t statistic

$$t = \frac{\mathbf{l}'\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}}{\mathbf{l}'\mathbf{Cl}}$$

are computed as

$$\nu = \frac{2(\mathbf{l}'\mathbf{Cl})^2}{\mathbf{g}'\mathbf{A}\mathbf{g}}$$

where $\mathbf{g}$ is the gradient of $\mathbf{l}'\mathbf{Cl}$ with respect to $\boldsymbol{\theta}$, evaluated at $\widehat{\boldsymbol{\theta}}$.

The calculation of Satterthwaite degrees of freedom requires extra memory to hold q matrices that are the size of the mixed model equations, where q is the number of covariance parameters. Extra computing time is also required to process these matrices. The implemented Satterthwaite method is intended to produce an accurate F approximation; however, the results can differ from those produced by PROC GLM. Also, the small-sample properties of this approximation have not been extensively investigated for the various models available with PROC GLIMMIX.

Kenward-Roger Degrees of Freedom Approximation

The DDFM=KENWARDROGER option prompts PROC GLIMMIX to compute the denominator degrees of freedom in t tests and F tests by using the approximation described in Kenward and Roger (1997). For inference on the linear combination $L\beta$ in a Gaussian linear model, they propose a scaled Wald statistic

$$\begin{aligned} F^* &= \lambda F \\ &= \frac{\lambda}{l}(\hat{\beta}-\beta)^T L(L^T\hat{\Phi}_A L)^{-1}L^T(\hat{\beta}-\beta), \end{aligned}$$

where $l = \mathrm{rank}(L)$, $\hat{\Phi}_A$ is a bias-adjusted estimator of the precision of $\hat{\beta}$, and $0 < \lambda < 1$. An appropriate $F_{l,m}$ approximation to the sampling distribution of F^* is derived by matching the first two moments of F^* with those from the approximating F distribution and solving for the values of λ and m. The value of m thus derived is the Kenward-Roger degrees of freedom. The precision estimator $\hat{\Phi}_A$ is bias-adjusted, in contrast to the conventional precision estimator $\Phi(\hat{\sigma}) = (X'V(\hat{\sigma})^{-1}X)^{-1}$, which is obtained by simply replacing σ with $\hat{\sigma}$ in $\Phi(\sigma)$, the asymptotic variance of $\hat{\beta}$. This method uses $\hat{\Phi}_A$ to address the fact that $\Phi(\hat{\sigma})$ is a biased estimator of $\Phi(\sigma)$, and $\Phi(\sigma)$ itself underestimates $\mathrm{var}(\hat{\beta})$ when σ is unknown. This bias-adjusted precision estimator is also discussed in Prasad and Rao (1990); Harville and Jeske (1992); Kackar and Harville (1984).

By default, the observed information matrix of the covariance parameter estimates is used in the calculations. For covariance structures that have nonzero second derivatives with respect to the covariance parameters, the Kenward-Roger covariance matrix adjustment includes a second-order term. This term can result in standard error shrinkage. Also, the resulting adjusted covariance matrix can then be indefinite and is not invariant under reparameterization. The FIRSTORDER suboption of the DDFM=KENWARDROGER option eliminates the second derivatives from the calculation of the covariance matrix adjustment. For scalar estimable functions, the resulting estimator is referred to as the Prasad-Rao estimator $\widetilde{m}^{@}$ in Harville and Jeske (1992). You can use the COVB(DETAILS) option to diagnose the adjustments that PROC GLIMMIX makes to the covariance matrix of fixed-effects parameter estimates. An application with DDFM=KENWARDROGER is presented in Example 52.8. The following are examples of covariance structures that generally lead to nonzero second derivatives: TYPE=ANTE(1), TYPE=AR(1), TYPE=ARH(1), TYPE=ARMA(1,1), TYPE=CHOL, TYPE=CSH, TYPE=FA0(q), TYPE=TOEPH, TYPE=UNR, and all TYPE=SP() structures.

DDFM=KENWARDROGER2 specifies an improved F approximation of the DDFM=KENWARD-ROGER type that uses a less biased precision estimator, as proposed by Kenward and Roger (2009). An important feature of the KR2 precision estimator is that it is invariant under reparameterization within the classes of intrinsically linear and intrinsically linear inverse covariance structures. For the invariance to hold within these two classes of covariance structures, a modified expected Hessian matrix is used in the computation of the covariance matrix of σ. The two cells classified as "Modified" scoring for RxPL estimation in Table 52.25 give the modified Hessian expressions for the cases where the scale parameter is profiled and not profiled. You can enforce the use of the modified expected Hessian matrix by specifying both the EXPHESSIAN and SCOREMOD options in the PROC GLIMMIX statement. Kenward and Roger (2009) note that for an intrinsically linear covariance parameterization, DDFM=KR2 produces the same precision estimator as that obtained using DDFM=KR(FIRSTORDER).

Empirical Covariance ("Sandwich") Estimators

Residual-Based Estimators

The GLIMMIX procedure can compute the classical sandwich estimator of the covariance matrix of the fixed effects, as well as several bias-adjusted estimators. This requires that the model is either an (overdispersed) GLM or a GLMM that can be processed by subjects (see the section "Processing by Subjects" on page 3957).

Consider a statistical model of the form

$$\mathbf{Y} = \boldsymbol{\mu} + \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim (\mathbf{0}, \boldsymbol{\Sigma})$$

The general expression of a sandwich covariance estimator is then

$$c \times \widehat{\boldsymbol{\Omega}} \left(\sum_{i=1}^{m} \mathbf{A}_i \widehat{\mathbf{D}}_i' \widehat{\boldsymbol{\Sigma}}_i^{-1} \mathbf{F}_i' \mathbf{e}_i \mathbf{e}_i' \mathbf{F}_i \widehat{\boldsymbol{\Sigma}}_i^{-1} \widehat{\mathbf{D}}_i \mathbf{A}_i \right) \widehat{\boldsymbol{\Omega}}$$

where $\mathbf{e}_i = \mathbf{y}_i - \widehat{\boldsymbol{\mu}}_i$, $\boldsymbol{\Omega} = (\mathbf{D}'\boldsymbol{\Sigma}^{-1}\mathbf{D})^{-}$.

For a GLMM estimated by one of the pseudo-likelihood techniques that involve linearization, you can make the following substitutions: $\mathbf{Y} \rightarrow \mathbf{P}$, $\boldsymbol{\Sigma} \rightarrow \mathbf{V}(\boldsymbol{\theta})$, $\mathbf{D} \rightarrow \mathbf{X}$, $\widehat{\boldsymbol{\mu}} \rightarrow \mathbf{X}\widehat{\boldsymbol{\beta}}$. These matrices are defined in the section "Pseudo-likelihood Estimation Based on Linearization" on page 3924.

The various estimators computed by the GLIMMIX procedure differ in the choice of the constant c and the matrices $\mathbf{F}_i$ and $\mathbf{A}_i$. You obtain the classical estimator, for example, with $c = 1$, and $\mathbf{F}_i = \mathbf{A}_i$ equal to the identity matrix.

The EMPIRICAL=ROOT estimator of Kauermann and Carroll (2001) is based on the approximation

$$\mathrm{Var}\left[\mathbf{e}_i \mathbf{e}_i'\right] \approx (\mathbf{I} - \mathbf{H}_i)\boldsymbol{\Sigma}_i$$

where $\mathbf{H}_i = \mathbf{D}_i \boldsymbol{\Omega} \mathbf{D}_i' \boldsymbol{\Sigma}_i^{-1}$. The EMPIRICAL=FIRORES estimator is based on the approximation

$$\mathrm{Var}\left[\mathbf{e}_i \mathbf{e}_i'\right] \approx (\mathbf{I} - \mathbf{H}_i)\boldsymbol{\Sigma}_i(\mathbf{I} - \mathbf{H}_i')$$

of Mancl and DeRouen (2001). Finally, the EMPIRICAL=FIROEEQ estimator is based on approximating an unbiased estimating equation (Fay and Graubard 2001). For this estimator, $\mathbf{A}_i$ is a diagonal matrix with entries

$$[\mathbf{A}_i]_{jj} = \left(1 - \min\{r, [\mathbf{Q}_i]_{jj}\}\right)^{-1/2}$$

where $\mathbf{Q}_i = \mathbf{D}_i' \widehat{\boldsymbol{\Sigma}}_i^{-1} \mathbf{D}_i \widehat{\boldsymbol{\Omega}}$. The optional number $0 \le r < 1$ is chosen to provide an upper bound on the correction factor. PROC GLIMMIX chooses as default value $r = 3/4$. The diagonal entries of $\mathbf{A}_i$ are then no greater than 2.

Table 52.26 summarizes the components of the computation for the GLMM based on linearization, where m denotes the number of subjects and k is the rank of $\mathbf{X}$.

Table 52.26 Empirical Covariance Estimators for a Linearized GLMM

EMPIRICAL=	c	$\mathbf{A}_i$	$\mathbf{F}_i$
CLASSICAL	1	$\mathbf{I}$	$\mathbf{I}$
DF	$\begin{cases} \frac{m}{m-k} & m > k \\ 1 & \text{otherwise} \end{cases}$	$\mathbf{I}$	$\mathbf{I}$
ROOT	1	$\mathbf{I}$	$(\mathbf{I} - \mathbf{H}_i')^{-1/2}$
FIRORES	1	$\mathbf{I}$	$(\mathbf{I} - \mathbf{H}_i')^{-1}$
FIROEEQ(r)	1	$\text{Diag}\{(1 - \min\{r, [\mathbf{Q}]_{jj}\})^{-1/2}\}$	$\mathbf{I}$

Computation of an empirical variance estimator requires that the data can be processed by independent sampling units. This is always the case in GLMs. In this case, m equals the sum of all frequencies. In GLMMs, the empirical estimators require that the data consist of multiple subjects. In that case, m equals the number of subjects as per the "Dimensions" table. The following section discusses how the GLIMMIX procedure determines whether the data can be processed by subjects. The section "GLM Mode or GLMM Mode" on page 3941 explains how PROC GLIMMIX determines whether a model is fit in GLM mode or in GLMM mode.

Design-Adjusted MBN Estimator

Morel (1989) and Morel, Bokossa, and Neerchal (2003) suggested a bias correction of the classical sandwich estimator that rests on an additive correction of the residual crossproducts and a sample size correction. This estimator is available with the EMPIRICAL=MBN option in the PROC GLIMMIX statement. In the notation of the previous section, the residual-based MBN estimator can be written as

$$\widehat{\mathbf{\Omega}} \left(\sum_{i=1}^{m} \widehat{\mathbf{D}}_i' \widehat{\mathbf{\Sigma}}_i^{-1} \left(c\mathbf{e}_i \mathbf{e}_i' + \mathbf{B}_i \right) \widehat{\mathbf{\Sigma}}_i^{-1} \widehat{\mathbf{D}}_i \right) \widehat{\mathbf{\Omega}}$$

where

- $c = (f-1)/(f-k) \times m/(m-1)$ or $c = 1$ when you specify the EMPIRICAL=MBN(NODF) option
- f is the sum of the frequencies
- k equals the rank of $\mathbf{X}$
- $\mathbf{B}_i = \delta_m \phi \widehat{\mathbf{\Sigma}}_i$
- $\phi = \max\left\{r, \text{trace}\left(\widehat{\mathbf{\Omega}}\mathbf{M}\right)/k^*\right\}$
- $\mathbf{M} = \sum_{i=1}^{m} \widehat{\mathbf{D}}_i' \widehat{\mathbf{\Sigma}}_i^{-1} \mathbf{e}_i \mathbf{e}_i' \widehat{\mathbf{\Sigma}}_i^{-1} \widehat{\mathbf{D}}_i$
- $k^* = k$ if $m \geq k$, otherwise k^* equals the number of nonzero singular values of $\widehat{\mathbf{\Omega}}\mathbf{M}$
- $\delta_m = k/(m-k)$ if $m > (d+1)k$ and $\delta_m = 1/d$ otherwise
- $d \geq 1$ and $0 \leq r \leq 1$ are parameters supplied with the *mbn-options* of the EMPIRICAL=MBN(*mbn-options*) option. The default values are $d = 2$ and $r = 1$. When the NODF option is in effect, the factor c is set to 1.

Rearranging terms, the MBN estimator can also be written as an additive adjustment to a sample-size corrected classical sandwich estimator

$$c \times \widehat{\mathbf{\Omega}} \left(\sum_{i=1}^{m} \widehat{\mathbf{D}}_i' \widehat{\mathbf{\Sigma}}_i^{-1} \mathbf{e}_i \mathbf{e}_i' \widehat{\mathbf{\Sigma}}_i^{-1} \widehat{\mathbf{D}}_i \right) \widehat{\mathbf{\Omega}} + \delta_m \phi \widehat{\mathbf{\Omega}}$$

Because δ_m is of order m^{-1}, the additive adjustment to the classical estimator vanishes as the number of independent sampling units (subjects) increases. The parameter ϕ is a measure of the design effect (Morel, Bokossa, and Neerchal 2003). Besides good statistical properties in terms of Type I error rates in small-m situations, the MBN estimator also has the desirable property of recovering rank when the number of sampling units is small. If $m < k$, the "meat" piece of the classical sandwich estimator is essentially a sum of rank one matrices. A small number of subjects relative to the rank of $\mathbf{X}$ can result in a loss of rank and subsequent loss of numerator degrees of freedom in tests. The additive MBN adjustment counters the rank exhaustion. You can examine the rank of an adjusted covariance matrix with the COVB(DETAILS) option in the MODEL statement.

When the principle of the MBN estimator is applied to the likelihood-based empirical estimator, you obtain

$$\mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1} \left(\sum_{i=1}^{m} c \mathbf{g}_i(\widehat{\boldsymbol{\alpha}}) \mathbf{g}_i(\widehat{\boldsymbol{\alpha}})' + \mathbf{B}_i \right) \mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1}$$

where $\mathbf{B}_i = -\delta_m \phi \mathbf{H}_i(\widehat{\boldsymbol{\alpha}})$, and $\mathbf{H}_i(\widehat{\boldsymbol{\alpha}})$ is the second derivative of the log likelihood for the ith sampling unit (subject) evaluated at the vector of parameter estimates, $\widehat{\boldsymbol{\alpha}}$. Also, $\mathbf{g}_i(\widehat{\boldsymbol{\alpha}})$ is the first derivative of the log likelihood for the ith sampling unit. This estimator is computed if you request EMPIRICAL=MBN with METHOD=LAPLACE or METHOD=QUAD.

In terms of adjusting the classical likelihood-based estimator (White 1982), the likelihood MBN estimator can be written as

$$c \times \mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1} \left(\sum_{i=1}^{m} \mathbf{g}_i(\widehat{\boldsymbol{\alpha}}) \mathbf{g}_i(\widehat{\boldsymbol{\alpha}})' \right) \mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1} - \delta_m \phi \mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1}$$

The parameter ϕ is determined as

- $\phi = \max \left\{ r, \operatorname{trace}\left(-\mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1} \mathbf{M} \right) / k^* \right\}$
- $\mathbf{M} = \sum_{i=1}^{m} \mathbf{g}_i(\widehat{\boldsymbol{\alpha}}) \mathbf{g}_i(\widehat{\boldsymbol{\alpha}})'$
- $k^* = k$ if $m \geq k$, otherwise k^* equals the number of nonzero singular values of $-\mathbf{H}(\widehat{\boldsymbol{\alpha}})^{-1} \mathbf{M}$

Exploring and Comparing Covariance Matrices

If you use an empirical (sandwich) estimator with the EMPIRICAL= option in the PROC GLIMMIX statement, the procedure replaces the model-based estimator of the covariance of the fixed effects with the sandwich estimator. This affects aspects of inference, such as prediction standard errors, tests of fixed effects, estimates, contrasts, and so forth. Similarly, if you choose the DDFM=KENWARDROGER degrees-of-freedom method in the MODEL statement, PROC GLIMMIX adjusts the model-based covariance matrix of the fixed effects according to Kenward and Roger (1997) or according to Kackar and Harville (1984) and Harville and Jeske (1992).

In this situation, the COVB(DETAILS) option in the MODEL statement has two effects. The GLIMMIX procedure displays the (adjusted) covariance matrix of the fixed effects and the model-based covariance matrix (the ODS name of the table with the model-based covariance matrix is CovBModelBased). The procedure also displays a table of statistics for the unadjusted and adjusted covariance matrix and for their comparison. The ODS name of this table is CovBDetails.

If the model-based covariance matrix is not replaced with an adjusted estimator, the COVB(DETAILS) option displays the model-based covariance matrix and provides diagnostic measures for it in the CovBDetails table.

The table generated by the COVB(DETAILS) option consists of several sections. See Example 52.8 for an application.

The trace and log determinant of covariance matrices are general scalar summaries that are sometimes used in direct comparisons, or in formulating other statistics, such as the difference of log determinants. The trace simply represents the sum of the variances of all fixed-effects parameters. If a matrix is indefinite, the determinant is reported instead of the log determinant.

The model-based and adjusted covariance matrices should have the same general makeup of eigenvalues. There should not be any negative eigenvalues, and they should have the same numbers of positive and zero eigenvalues. A reduction in rank due to the adjustment is troublesome for aspects of inference. Negative eigenvalues are listed in the table only if they occur, because a covariance matrix should be at least positive semi-definite. However, the GLIMMIX procedure examines the model-based and adjusted covariance matrix for negative eigenvalues. The condition numbers reported by PROC GLIMMIX for positive (semi-)definite matrices are computed as the ratio of the largest and smallest nonzero eigenvalue. A large condition number reflects poor conditioning of the matrix.

Matrix norms are extensions of the concept of vector norms to measure the "length" of a matrix. The Frobenius norm of an $(n \times m)$ matrix $\mathbf{A}$ is the direct equivalent of the Euclidean vector norm, the square root of the sum of the squared elements,

$$||\mathbf{A}||_F = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^2}$$

The ∞- and 1-norms of matrix $\mathbf{A}$ are the maximum absolute row and column sums, respectively:

$$||\mathbf{A}||_\infty = \max\left\{\sum_{j=1}^{m}|a_{ij}| \ : i = 1,\ldots,n\right\}$$
$$||\mathbf{A}||_1 = \max\left\{\sum_{i=1}^{n}|a_{ij}| \ : j = 1,\ldots,m\right\}$$

These two norms are identical for symmetric matrices.

The "Comparison" section of the CovBDetails table provides several statistics that set the matrices in relationship. The concordance correlation reported by the GLIMMIX procedure is a standardized measure of the closeness of the model-based and adjusted covariance matrix. It is a slight modification of the covariance concordance correlation in Vonesh, Chinchilli, and Pu (1996) and Vonesh and Chinchilli (1997, Ch. 8.3). Denote as $\mathbf{\Omega}$ the $(p \times p)$ model-based covariance matrix and as $\mathbf{\Omega}_a$ the adjusted matrix. Suppose that $\mathbf{K}$ is the matrix obtained from the identity matrix of size p by replacing diagonal elements corresponding to singular rows in $\mathbf{\Omega}$ with zeros. The lower triangular portion of $\mathbf{\Omega}^{-1/2}\mathbf{\Omega}_a\mathbf{\Omega}^{-1/2}$ is stored in vector $\boldsymbol{\omega}$ and the lower triangular portion of $\mathbf{K}$ is stored in vector $\mathbf{k}$. The matrix $\mathbf{\Omega}^{-1/2}$ is constructed from an eigenanalysis of $\mathbf{\Omega}$ and is symmetric. The covariance concordance correlation is then

$$r(\boldsymbol{\omega}) = 1 - \frac{||\boldsymbol{\omega} - \mathbf{k}||^2}{||\boldsymbol{\omega}||^2 + ||\mathbf{k}||^2}$$

This measure is 1 if $\mathbf{\Omega} = \mathbf{\Omega}_a$. If $\boldsymbol{\omega}$ is orthogonal to $\mathbf{k}$, there is total disagreement between the model-based and the adjusted covariance matrix and $r(\boldsymbol{\omega})$ is zero.

The discrepancy function reported by PROC GLIMMIX is computed as

$$d = \log\{|\mathbf{\Omega}|\} - \log\{|\mathbf{\Omega}_a|\} + \mathrm{trace}\{\mathbf{\Omega}_a\mathbf{\Omega}^-\} - \mathrm{rank}\{\mathbf{\Omega}\}$$

In diagnosing departures between an assumed covariance structure and $\mathrm{Var}[\mathbf{Y}]$—using an empirical estimator—Vonesh, Chinchilli, and Pu (1996) find that the concordance correlation is useful in detecting gross departures and propose $\lambda = n_s d$ to test the correctness of the assumed model, where n_s denotes the number of subjects.

Predictive Margins Compared with LS-Means

The MARGINS statement computes predictive margins of fixed effects. The LSMEANS statement computes least squares means (LS-means) of fixed effects. Both predictive margins and LS-means are covariate-adjusted marginal means. They differ in the following ways:

- LS-means estimate the marginal means over a balanced population—that is, a population in which there are an equal number of observations for all possible effect level combinations. In contrast, predictive margins are computed over the observed distribution of the covariates. You can use the OM option in the LSMEANS statement to change the LS-means coefficients to be proportional to those found in the input data set. For linear models, this adjustment leads to LS-means that are equal to predictive margins if the margin effect is not part of any interaction effect.

- LS-means are constructed on the linear predictor scale—that is, the scale of $x\beta + z\gamma$. In contrast, predictive margins are computed on the mean scale—that is, the scale of $g^{(-1)}(x\beta + z\gamma)$. You can use the ILINK option in the LSMEANS statement to apply the inverse link transform to the LS-means. Note that in general the transformed LS-means are not equal to predictive margins for nonlinear models. They are equal only when the values of all covariates and classification variables are either specified or fixed in the computation.

Processing by Subjects

Some mixed models can be expressed in different but mathematically equivalent ways with PROC GLIMMIX statements. While equivalent statements lead to equivalent statistical models, the data processing and estimation phase can be quite different, depending on how you write the GLIMMIX statements. For example, the particular use of the SUBJECT= option in the RANDOM statement affects data processing and estimation. Certain options are available only when the data are processed by subject, such as the EMPIRICAL option in the PROC GLIMMIX statement.

Consider a GLIMMIX model where variables A and Rep are classification variables with a and r levels, respectively. The following pairs of statements produce the same random-effects structure:

```
class Rep A;
random Rep*A;

class Rep A;
random intercept / subject=Rep*A;

class Rep A;
random Rep / subject=A;

class Rep A;
random A / subject=Rep;
```

In the first case, PROC GLIMMIX does not process the data by subjects because no SUBJECT= option was given. The computation of empirical covariance estimators, for example, will not be possible. The marginal variance-covariance matrix has the same block-diagonal structure as for cases 2–4, where each block consists of the observations belonging to a unique combination of Rep and A. More importantly, the dimension of the $\mathbf{Z}$ matrix of this model will be $n \times ra$, and $\mathbf{Z}$ will be sparse. In the second case, the $\mathbf{Z}_i$ matrix for each of the ra subjects is a vector of ones.

If the data can be processed by subjects, the procedure typically executes faster and requires less memory. The differences can be substantial, especially if the number of subjects is large. Recall that fitting of generalized linear mixed models might be doubly iterative. Small gains in efficiency for any one optimization can produce large overall savings.

If you interpret the intercept as "1," then a RANDOM statement with TYPE=VC (the default) and no SUBJECT= option can be converted into a statement with subject by dividing the random effect by the eventual subject effect. However, the presence of the SUBJECT= option does not imply processing by subject. If a RANDOM statement does not have a SUBJECT= effect, processing by subjects is not possible unless the random effect is a pure R-side overdispersion effect. In the following example, the data will not be processed by subjects, because the first RANDOM statement specifies a G-side component and does not use a SUBJECT= option:

```
proc glimmix;
   class A B;
   model y = B;
   random A;
   random B / subject=A;
run;
```

To allow processing by subjects, you can write the equivalent model with the following statements:

```
proc glimmix;
   class A B;
   model y = B;
   random int / subject=A;
   random B   / subject=A;
run;
```

If you denote a variance component effect X with subject effect S as X–(S), then the "calculus of random effects" applied to the first RANDOM statement reads A = Int*A = Int–(A) = A–(Int). For the second statement there are even more equivalent formulations: A*B = A*B*Int = A*B–(Int) = A–(B) = B–(A) = Int–(A*B).

If there are multiple subject effects, processing by subjects is possible if the effects are equal or contained in each other. Note that in the last example the A*B interaction is a random effect. The following statements give an equivalent specification to the previous model:

```
proc glimmix;
   class A B;
   model y = B;
   random int / subject=A;
   random A   / subject=B;
run;
```

Processing by subjects is not possible in this case, because the two subject effects are not syntactically equal or contained in each other. The following statements depict a case where subject effects are syntactically contained:

```
proc glimmix;
   class A B;
   model y = B;
   random int / subject=A;
   random int / subject=A*B;
run;
```

The A main effect is contained in the A*B interaction. The GLIMMIX procedure chooses as the subject effect for processing the effect that is contained in all other subject effects. In this case, the subjects are defined by the levels of A.

You can examine the "Model Information" and "Dimensions" tables to see whether the GLIMMIX procedure processes the data by subjects and which effect is used to define subjects. The "Model Information" table displays whether the marginal variance matrix is diagonal (GLM models), blocked, or not blocked. The "Dimensions" table tells you how many subjects (=blocks) there are.

Finally, nesting and crossing of interaction effects in subject effects are equivalent. The following two RANDOM statements are equivalent:

```
class Rep A;
random intercept / subject=Rep*A;

class Rep A;
random intercept / subject=Rep(A);
```

Radial Smoothing Based on Mixed Models

The radial smoother implemented with the TYPE=RSMOOTH option in the RANDOM statement is an approximate low-rank thin-plate spline as described in Ruppert, Wand, and Carroll (2003, Chapter 13.4–13.5). The following sections discuss in more detail the mathematical-statistical connection between mixed models and penalized splines and the determination of the number of spline knots and their location as implemented in the GLIMMIX procedure.

From Penalized Splines to Mixed Models

The connection between splines and mixed models arises from the similarity of the penalized spline fitting criterion to the minimization problem that yields the mixed model equations and solutions for $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. This connection is made explicit in the following paragraphs. An important distinction between classical spline fitting and its mixed model smoothing variant, however, lies in the nature of the spline coefficients. Although they address similar minimization criteria, the solutions for the spline coefficients in the GLIMMIX procedure are the solutions of random effects, not fixed effects. Standard errors of predicted values, for example, account for this source of variation.

Consider the linearized mixed pseudo-model from the section "The Pseudo-model" on page 3924, $\mathbf{P} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$. One derivation of the mixed model equations, whose solutions are $\widehat{\boldsymbol{\beta}}$ and $\widehat{\boldsymbol{\gamma}}$, is to maximize the joint density of $f(\boldsymbol{\gamma}, \boldsymbol{\epsilon})$ with respect to $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. This is not a true likelihood problem, because $\boldsymbol{\gamma}$ is not a parameter, but a random vector.

In the special case with $\mathrm{Var}[\boldsymbol{\epsilon}] = \phi\mathbf{I}$ and $\mathrm{Var}[\boldsymbol{\gamma}] = \sigma^2\mathbf{I}$, the maximization of $f(\boldsymbol{\gamma}, \boldsymbol{\epsilon})$ is equivalent to the minimization of

$$Q(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \phi^{-1}(\mathbf{p} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{\gamma})'(\mathbf{p} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{\gamma}) + \sigma^{-2}\boldsymbol{\gamma}'\boldsymbol{\gamma}$$

Now consider a linear spline as in Ruppert, Wand, and Carroll (2003, p. 108),

$$p_i = \beta_0 + \beta_1 x_i + \sum_{j=1}^{K} \gamma_j (x_i - t_j)_+$$

where the γ_j denote the spline coefficients at knots $t_1, \ldots, t_K$. The truncated line function is defined as

$$(x - t)_+ = \begin{cases} x - t & x > t \\ 0 & \text{otherwise} \end{cases}$$

If you collect the intercept and regressor x into the matrix $\mathbf{X}$, and if you collect the truncated line functions into the $(n \times K)$ matrix $\mathbf{Z}$, then fitting the linear spline amounts to minimization of the penalized spline criterion

$$Q^*(\boldsymbol{\beta}, \boldsymbol{\gamma}) = (\mathbf{p} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{\gamma})'(\mathbf{p} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{\gamma}) + \lambda^2\boldsymbol{\gamma}'\boldsymbol{\gamma}$$

where λ is the smoothing parameter.

Because minimizing $Q^*(\boldsymbol{\beta}, \boldsymbol{\gamma})$ with respect to $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ is equivalent to minimizing $Q^*(\boldsymbol{\beta}, \boldsymbol{\gamma})/\phi$, both problems lead to the same solution, and $\lambda = \phi/\sigma$ is the smoothing parameter. The mixed model formulation of spline smoothing has the advantage that the smoothing parameter is selected "automatically." It is a function of the covariance parameter estimates, which, in turn, are estimated according to the method you specify with the METHOD= option in the PROC GLIMMIX statement.

To accommodate nonnormal responses and general link functions, the GLIMMIX procedure uses $\mathrm{Var}[\boldsymbol{\epsilon}] = \phi\widetilde{\boldsymbol{\Delta}}^{-1}\mathbf{A}\widetilde{\boldsymbol{\Delta}}^{-1}$, where $\mathbf{A}$ is the matrix of variance functions and $\boldsymbol{\Delta}$ is the diagonal matrix of mean derivatives defined earlier. The correspondence between spline smoothing and mixed modeling is then one between a weighted linear mixed model and a weighted spline. In other words, the minimization criterion that yields the estimates $\widehat{\boldsymbol{\beta}}$ and solutions $\widehat{\boldsymbol{\gamma}}$ is then

$$Q(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \phi^{-1}(\mathbf{p} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{\gamma})'\widetilde{\boldsymbol{\Delta}}\mathbf{A}^{-1}\widetilde{\boldsymbol{\Delta}}(\mathbf{p} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{\gamma})' + \sigma^{-2}\boldsymbol{\gamma}'\boldsymbol{\gamma}$$

If you choose the TYPE=RSMOOTH covariance structure, PROC GLIMMIX chooses radial basis functions as the spline basis and transforms them to approximate a thin-plate spline as in Chapter 13.4 of Ruppert, Wand, and Carroll (2003). For computational expediency, the number of knots is chosen to be less than the number of data points. Ruppert, Wand, and Carroll (2003) recommend one knot per every four unique regressor values for one-dimensional smoothers. In the multivariate case, general recommendations are more difficult, because the optimal number and placement of knots depend on the spatial configuration of samples. Their recommendation for a bivariate smoother is one knot per four samples, but at least 20 and no more than 150 knots (Ruppert, Wand, and Carroll 2003, p. 257).

The magnitude of the variance component σ^2 depends on the metric of the random effects. For example, if you apply radial smoothing in time, the variance changes if you measure time in days or minutes. If the solution for the variance component is near zero, then a rescaling of the random effect data can help the optimization problem by moving the solution for the variance component away from the boundary of the parameter space.

Knot Selection

The GLIMMIX procedure computes knots for low-rank smoothing based on the vertices or centroids of a k-d tree. The default is to use the vertices of the tree as the knot locations, if you use the TYPE=RSMOOTH covariance structure. The construction of this tree amounts to a partitioning of the random regressor space until all partitions contain at most b observations. The number b is called the *bucket size* of the k-d tree. You can exercise control over the construction of the tree by changing the bucket size with the BUCKET= suboption of the KNOTMETHOD=KDTREE option in the RANDOM statement. A large bucket size leads to fewer knots, but it is not correct to assume that K, the number of knots, is simply $\lfloor n/b \rfloor$. The number of vertices depends on the configuration of the values in the regressor space. Also, coordinates of the bounding hypercube are vertices of the tree. In the one-dimensional case, for example, the extreme values of the random effect are vertices.

To demonstrate how the k-d tree partitions the random-effects space based on observed data and the influence of the bucket size, consider the following example from Chapter 78, "The LOESS Procedure." The SAS data set Gas contains the results of an engine exhaust emission study (Brinkman 1981). The covariate in this analysis, E, is a measure of the air-fuel mixture richness. The response, NOx, measures the nitric oxide concentration (in micrograms per joule, and normalized).

```
data Gas;
   input NOx E;
   format NOx E f5.3;
   datalines;
 4.818  0.831
 2.849  1.045
 3.275  1.021
 4.691  0.97
 4.255  0.825
 5.064  0.891
 2.118  0.71
 4.602  0.801
 2.286  1.074
 0.97   1.148
 3.965  1
 5.344  0.928
 3.834  0.767
 1.99   0.701
 5.199  0.807
 5.283  0.902
 3.752  0.997
 0.537  1.224
 1.64   1.089
 5.055  0.973
 4.937  0.98
 1.561  0.665
;
```

There are 22 observations in the data set, and the values of the covariate are unique. If you want to smooth these data with a low-rank radial smoother, you need to choose the number of knots, as well as their placement within the support of the variable E. The k-d tree construction depends on the observed values of the variable E; it is independent of the values of nitric oxide in the data. The following statements construct a tree based on a bucket size of $b = 11$ and display information about the tree and the selected knots:

```
ods select KDtree KnotInfo;
proc glimmix data=gas nofit;
   model NOx = e;
   random e  / type=rsmooth
               knotmethod=kdtree(bucket=11 treeinfo knotinfo);
run;
```

The NOFIT option prevents the GLIMMIX procedure from fitting the model. This option is useful if you want to investigate the knot construction for various bucket sizes. The TREEINFO and KNOTINFO suboptions of the KNOTMETHOD=KDTREE option request displays of the k-d tree and the knot coordinates derived from it. Construction of the tree commences by splitting the data in half. For $b = 11$, $n = 22$, neither of the two splits contains more than b observations and the process stops. With a single split value, and the two extreme values, the tree has two terminal nodes and leads to three knots (Figure 52.13). Note that for one-dimensional problems, vertices of the k-d tree always coincide with data values.

Figure 52.13 K-d Tree and Knots for Bucket Size 11

The GLIMMIX Procedure

kd-Tree for RSmooth(E)				
Node Number	Left Child	Right Child	Split Direction	Split Value
0	1	2	E	0.9280
1			TERMINAL	
2			TERMINAL	

Radial Smoother Knots for RSmooth(E)	
Knot Number	E
1	0.6650
2	0.9280
3	1.2240

If the bucket size is reduced to $b = 8$, the following statements produce the tree and knots in Figure 52.14:

```
ods select KDtree KnotInfo;
proc glimmix data=gas nofit;
   model NOx = e;
   random e / type=rsmooth
              knotmethod=kdtree(bucket=8 treeinfo knotinfo);
run;
```

The initial split value of 0.9280 leads to two sets of 11 observations. In order to achieve a partition into cells that contain at most eight observations, each initial partition is split at its median one more time. Note that one split value is greater and one split value is less than 0.9280.

Figure 52.14 K-d Tree and Knots for Bucket Size 8

The GLIMMIX Procedure

kd-Tree for RSmooth(E)				
Node Number	Left Child	Right Child	Split Direction	Split Value
0	1	2	E	0.9280
1	3	4	E	0.8070
2	5	6	E	1.0210
3			TERMINAL	
4			TERMINAL	
5			TERMINAL	
6			TERMINAL	

Figure 52.14 *continued*

Radial Smoother Knots for RSmooth(E)	
Knot Number	**E**
1	0.6650
2	0.8070
3	0.9280
4	1.0210
5	1.2240

A further reduction in bucket size to $b = 4$ leads to the tree and knot information shown in Figure 52.15.

Figure 52.15 K-d Tree and Knots for Bucket Size 4

The GLIMMIX Procedure

kd-Tree for RSmooth(E)				
Node Number	**Left Child**	**Right Child**	**Split Direction**	**Split Value**
0	1	2	E	0.9280
1	3	4	E	0.8070
2	9	10	E	1.0210
3	5	6	E	0.7100
4	7	8	E	0.8910
5			TERMINAL	
6			TERMINAL	
7			TERMINAL	
8			TERMINAL	
9	11	12	E	0.9800
10	13	14	E	1.0890
11			TERMINAL	
12			TERMINAL	
13			TERMINAL	
14			TERMINAL	

Figure 52.15 *continued*

Radial Smoother Knots for RSmooth(E)	
Knot Number	E
1	0.6650
2	0.7100
3	0.8070
4	0.8910
5	0.9280
6	0.9800
7	1.0210
8	1.0890
9	1.2240

The split value for $b = 11$ is also a split value for $b = 8$, the split values for $b = 8$ are a subset of those for $b = 4$, and so forth. Figure 52.16 displays the data and the location of split values for the three cases. For a one-dimensional problem (a univariate smoother), the vertices comprise the split values and the values on the bounding interval.

You might want to move away from the boundary, in particular for an irregular data configuration or for multivariate smoothing. The KNOTTYPE=CENTER suboption of the KNOTMETHOD= option chooses centroids of the leaf node cells instead of vertices. This tends to move the outer knot locations closer to the convex hull, but not necessarily to data locations. In the emission example, choosing a bucket size of $b = 11$ and centroids as knot locations yields two knots at E=0.7956 and E=1.076. If you choose the NEAREST suboption, then the nearest neighbor of a vertex or centroid will serve as the knot location. In this case, the knot locations are a subset of the data locations, regardless of the dimension of the smooth.

Figure 52.16 Vertices of k-d Trees for Various Bucket Sizes

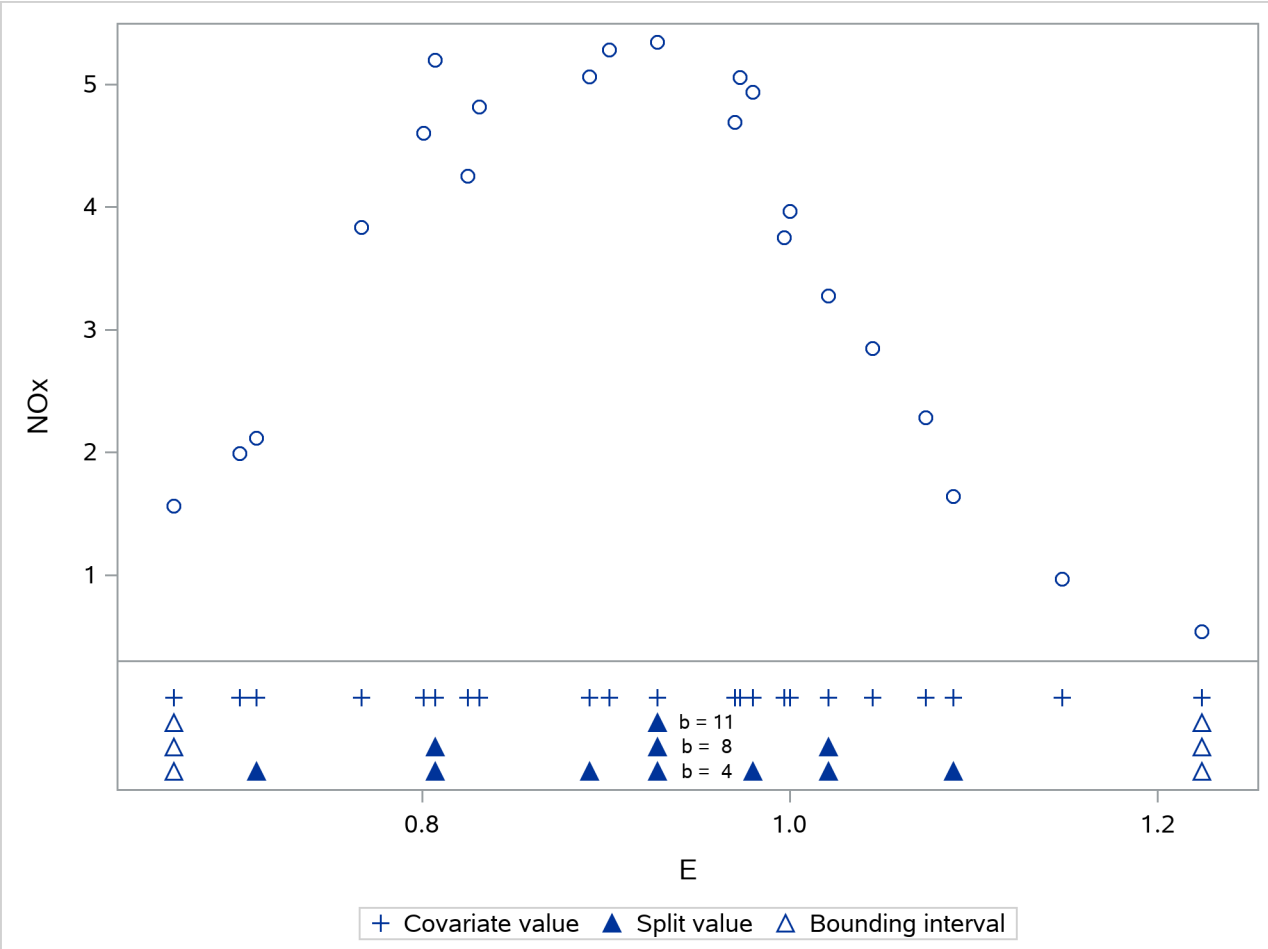

Odds and Odds Ratio Estimation

In models with a logit, generalized logit, or cumulative logit link, you can obtain estimates of odds ratios through the ODDSRATIO options in the PROC GLIMMIX, LSMEANS, and MODEL statements. This section provides details about the computation and interpretation of the computed quantities. Note that for these link functions the EXP option in the ESTIMATE and LSMESTIMATE statements also produces odds or odds ratios.

Consider first a model with a dichotomous outcome variable, linear predictor $\eta = \mathbf{x}'\boldsymbol{\beta} + \mathbf{z}'\boldsymbol{\gamma}$, and logit link function. Suppose that η_0 represents the linear predictor for a condition of interest. For example, in a simple logistic regression model with $\eta = \alpha + \beta x$, η_0 might correspond to the linear predictor at a particular value of the covariate—say, $\eta_0 = \alpha + \beta x_0$.

The modeled probability is $\pi = 1/(1 + \exp\{-\eta\})$, and the odds for $\eta = \eta_0$ are

$$\frac{\pi_0}{1 - \pi_0} = \frac{1/(1 + \exp\{-\eta_0\})}{\exp\{-\eta_0\}/(1 + \exp\{-\eta_0\})} = \exp\{\eta_0\}$$

Because η_0 is a logit, it represents the log odds. The odds ratio $\psi(\eta_1, \eta_0)$ is defined as the ratio of odds for η_1 and η_0,

$$\psi(\eta_1, \eta_0) = \exp\{\eta_1 - \eta_0\}$$

The odds ratio compares the odds of the outcome under the condition expressed by η_1 to the odds under the condition expressed by η_0. In the preceding simple logistic regression example, this ratio equals $\exp\{\beta(x_1 - x_0)\}$. The exponentiation of the estimate of β is thus an estimate of the odds ratio comparing conditions for which $x_1 - x_0 = 1$. If x and $x + 1$ represent standard and experimental conditions, for example, $\exp\{\beta\}$ compares the odds of the outcome under the experimental condition to the odds under the standard condition. For many other types of models, odds ratios can be expressed as simple functions of parameter estimates. For example, suppose you are fitting a logistic model with a single classification effect with three levels:

```
proc glimmix;
   class A;
   model y = A / dist=binary;
run;
```

The estimated linear predictor for level j of A is $\widehat{\eta}_j = \widehat{\beta} + \widehat{\alpha}_j$, $j = 1, 2, 3$. Because the $\mathbf{X}$ matrix is singular in this model due to the presence of an overall intercept, the solution for the intercept estimates $\beta + \alpha_3$, and the solution for the jth treatment effect estimates $\alpha_j - \alpha_3$. Exponentiating the solutions for α_1 and α_2 thus produces odds ratios comparing the odds for these levels against the third level of A.

Results designated as odds or odds ratios in the GLIMMIX procedure might reduce to simple exponentiations of solutions in the "Parameter Estimates" table, but they are computed by a different mechanism if the model contains classification variables. The computations rely on general estimable functions; for the MODEL, LSMEANS, and LSMESTIMATE statements, these functions are based on least squares means. This enables you to obtain odds ratio estimates in more complicated models that involve main effects and interactions, including interactions between continuous and classification variables.

In all cases, the results represent the exponentiation of a linear function of the fixed-effects parameters, $\eta = \mathbf{l}'\boldsymbol{\beta}$. If L_η and U_η are the confidence limits for η on the logit scale, confidence limits for the odds or the odds ratio are obtained as $\exp\{L_\eta\}$ and $\exp\{U_\eta\}$.

The Odds Ratio Estimates Table

This table is produced by the ODDSRATIO option in the MODEL statement. It consists of estimates of odds ratios and their confidence limits. Odds ratios are produced for the following:

- classification main effects, if they appear in the MODEL statement
- continuous variables in the MODEL statement, unless they appear in an interaction with a classification effect
- continuous variables in the MODEL statement at fixed levels of a classification effect, if the MODEL statement contains an interaction of the two.
- continuous variables in the MODEL statements if they interact with other continuous variables

The Default Table

Consider the following PROC GLIMMIX statements that fit a logistic model with one classification effect, one continuous variable, and their interaction (the ODDSRATIO option in the MODEL statement requests the "Odds Ratio Estimates" table).

```
proc glimmix;
   class A;
   model y = A x A*x / dist=binary oddsratio;
run;
```

By default, odds ratios are computed as follows:

- The covariate is set to its average, $\overline{x}$, and the least squares means for the A effect are obtained. Suppose $\mathbf{L}^{(1)}$ denotes the matrix of coefficients defining the estimable functions that produce the a least squares means $\mathbf{L}\widehat{\boldsymbol{\beta}}$, and $\mathbf{l}_j^{(1)}$ denotes the jth row of $\mathbf{L}^{(1)}$. Differences of the least squares means against the last level of the A factor are computed and exponentiated:

$$\begin{aligned}
\psi(A_1, A_a) &= \exp\left\{\left(\mathbf{l}_1^{(1)} - \mathbf{l}_a^{(1)}\right)\widehat{\boldsymbol{\beta}}\right\} \\
\psi(A_2, A_a) &= \exp\left\{\left(\mathbf{l}_2^{(1)} - \mathbf{l}_a^{(1)}\right)\widehat{\boldsymbol{\beta}}\right\} \\
&\vdots \\
\psi(A_{a-1}, A_a) &= \exp\left\{\left(\mathbf{l}_{a-1}^{(1)} - \mathbf{l}_a^{(1)}\right)\widehat{\boldsymbol{\beta}}\right\}
\end{aligned}$$

  The differences are checked for estimability. Notice that this set of odds ratios can also be obtained with the following LSMESTIMATE statement (assuming A has five levels):

```
lsmestimate A 1  0  0  0 -1,
              0  1  0  0 -1,
              0  0  1  0 -1,
              0  0  0  1 -1 / exp cl;
```

  You can also obtain the odds ratios with this LSMEANS statement (assuming the last level of A is coded as 5):

```
lsmeans A / diff=control('5') oddsratio cl;
```

- The odds ratios for the covariate must take into account that x occurs in an interaction with the A effect. A second set of least squares means are computed, where x is set to $\overline{x} + 1$. Denote the coefficients of the estimable functions for this set of least squares means as $\mathbf{L}^{(2)}$. Differences of the least squares means at a given level of factor A are then computed and exponentiated:

$$\begin{aligned}
\psi(A(\overline{x}+1)_1, A(\overline{x})_1) &= \exp\left\{\left(\mathbf{l}_1^{(2)} - \mathbf{l}_1^{(1)}\right)\widehat{\boldsymbol{\beta}}\right\} \\
\psi(A(\overline{x}+1)_2, A(\overline{x})_2) &= \exp\left\{\left(\mathbf{l}_2^{(2)} - \mathbf{l}_2^{(1)}\right)\widehat{\boldsymbol{\beta}}\right\} \\
&\vdots \\
\psi(A(\overline{x}+1)_a, A(\overline{x})_a) &= \exp\left\{\left(\mathbf{l}_a^{(2)} - \mathbf{l}_a^{(1)}\right)\widehat{\boldsymbol{\beta}}\right\}
\end{aligned}$$

The differences are checked for estimability. If the continuous covariate does not appear in an interaction with the A variable, only a single odds ratio estimate related to x would be produced, relating the odds of a one-unit shift in the regressor from $\overline{x}$.

Suppose you fit a model that contains interactions of continuous variables, as with the following statements:

```
proc glimmix;
   class A;
   model y = A x x*z / dist=binary oddsratio;
run;
```

In the computation of the A least squares means, the continuous effects are set to their means—that is, $\overline{x}$ and $\overline{xz}$. In the computation of odds ratios for x, linear predictors are computed at x = $\overline{x}$, x*z = $\overline{x} \times \overline{z}$ and at x = $\overline{x} + 1$, x*z = $(\overline{x} + 1)\overline{z}$.

Modifying the Default Table, Customized Odds Ratios

Several suboptions of the ODDSRATIO option in the MODEL statement are available to obtain customized odds ratio estimates. For customized odds ratios that cannot be obtained with these suboptions, use the EXP option in the ESTIMATE or LSMESTIMATE statement.

The type of differences constructed when the levels of a classification factor are varied is controlled by the DIFF= suboption. By default, differences against the last level are taken. DIFF=FIRST computes differences from the first level, and DIFF=ALL computes odds ratios based on all pairwise differences.

For continuous variables in the model, you can change both the reference value (with the AT suboption) and the units of change (with the UNIT suboption). By default, a one-unit change from the mean of the covariate is assessed. For example, the following statements produce all pairwise differences for the A factor:

```
proc glimmix;
   class A;
   model y = A x A*x / dist=binary
                       oddsratio(diff=all
                                 at   x=4
                                 unit x=3);
run;
```

The covariate x is set to the reference value $x = 4$ in the computation of the least squares means for the A odds ratio estimates. The odds ratios computed for the covariate are based on differencing this set of least squares means with a set of least squares means computed at $x = 4 + 3$.

Odds or Odds Ratio

The odds ratio is the exponentiation of a difference on the logit scale,

$$\psi(\eta_1, \eta_0) = \exp\{(\mathrm{l}_1 - \mathrm{l}_0)\boldsymbol{\beta}\}$$

and $\exp\{\mathrm{l}_1\boldsymbol{\beta}\}$ and $\exp\{\mathrm{l}_0\boldsymbol{\beta}\}$ are the corresponding odds. If the ODDSRATIO option is specified in a suitable model in the PROC GLIMMIX statement or the individual statements that support the option, odds ratios are computed in the "Odds Ratio Estimates" table (MODEL statement), the "Differences of Least Squares Means" table (LSMEANS / DIFF), and the "Simple Effect Comparisons of Least Squares Means" table (LSMEANS / SLICEDIFF=). Odds are computed in the "Least Squares Means" table.

Odds Ratios in Multinomial Models

The GLIMMIX procedure fits two kinds of models to multinomial data. Models with cumulative link functions apply to ordinal data, and generalized logit models are fit to nominal data. If you model a multinomial response with LINK=CUMLOGIT or LINK=GLOGIT, odds ratio results are available for these models.

In the generalized logit model, you model baseline category logits. By default, the GLIMMIX procedure chooses the last category as the reference category. If your nominal response has J categories, the baseline logit for category j is

$$\log\left\{\pi_j/\pi_J\right\} = \eta_j = \mathbf{x}'\boldsymbol{\beta}_j + \mathbf{z}'\mathbf{u}_j$$

and

$$\pi_j = \frac{\exp\{\eta_j\}}{\sum_{k=1}^{J}\exp\{\eta_k\}}$$
$$\eta_J = 0$$

As before, suppose that the two conditions to be compared are identified with subscripts 1 and 0. The log odds ratio of outcome j versus J for the two conditions is then

$$\begin{aligned}\log\left\{\psi\left(\eta_{j1},\eta_{j0}\right)\right\} &= \log\left\{\frac{\pi_{j1}/\pi_{J1}}{\pi_{j0}/\pi_{J0}}\right\} = \log\left\{\frac{\exp\{\eta_{j1}\}}{\exp\{\eta_{j0}\}}\right\}\\ &= \eta_{j1} - \eta_{j0}\end{aligned}$$

Note that the log odds ratios are again differences on the scale of the linear predictor, but they depend on the response category. The GLIMMIX procedure determines the estimable functions whose differences represent log odds ratios as discussed previously but produces separate estimates for each nonreference response category.

In models for ordinal data, PROC GLIMMIX models the logits of cumulative probabilities. Thus, the estimates on the linear scale represent log cumulative odds. The cumulative logits are formed as

$$\log\left\{\frac{\Pr(Y \le j)}{\Pr(Y > j)}\right\} = \eta_j = \alpha_j + \mathbf{x}'\boldsymbol{\beta} + \mathbf{z}'\boldsymbol{\gamma} = \alpha_j + \tilde{\eta}$$

so that the linear predictor depends on the response category only through the intercepts (cutoffs) $\alpha_1,\ldots,\alpha_{J-1}$. The odds ratio comparing two conditions represented by linear predictors η_{j1} and η_{j0} is then

$$\begin{aligned}\psi\left(\eta_{j1},\eta_{j0}\right) &= \exp\left\{\eta_{j1} - \eta_{j0}\right\}\\ &= \exp\left\{\tilde{\eta}_1 - \tilde{\eta}_0\right\}\end{aligned}$$

and is independent of category.

Parameterization of Generalized Linear Mixed Models

PROC GLIMMIX constructs a generalized linear mixed model according to the specifications in the CLASS, MODEL, and RANDOM statements. Each effect in the MODEL statement generates one or more columns in the matrix **X**, and each G-side effect in the RANDOM statement generates one or more columns in the matrix **Z**. R-side effects in the RANDOM statement do not generate model matrices; they serve only to index observations within subjects. This section shows how the GLIMMIX procedure builds **X** and **Z**. You can output the **X** and **Z** matrices to a SAS data set with the OUTDESIGN= option in the PROC GLIMMIX statement.

The general rules and techniques for parameterization of a linear model are given in "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics." The following paragraphs discuss how these rules differ in a mixed model, in particular, how parameterization differs between the **X** and the **Z** matrix.

Intercept

By default, all models automatically include a column of 1s in **X** to estimate a fixed-effect intercept parameter. You can use the NOINT option in the MODEL statement to suppress this intercept. The NOINT option is useful when you are specifying a classification effect in the MODEL statement and you want the parameter estimates to be in terms of the (linked) mean response for each level of that effect, rather than in terms of a deviation from an overall mean.

By contrast, the intercept is not included by default in **Z**. To obtain a column of 1s in **Z**, you must specify in the RANDOM statement either the INTERCEPT effect or some effect that has only one level.

Interaction Effects

Often a model includes interaction (crossed) effects. With an interaction, PROC GLIMMIX first reorders the terms to correspond to the order of the variables in the CLASS statement. Thus, B*A becomes A*B if A precedes B in the CLASS statement. Then, PROC GLIMMIX generates columns for all combinations of levels that occur in the data. The order of the columns is such that the rightmost variables in the cross index faster than the leftmost variables. Empty columns (which would contain all 0s) are not generated for **X**, but they are for **Z**.

See Table 20.6 in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics," for an example of an interaction parameterization.

Nested Effects

Nested effects are generated in the same manner as crossed effects. Hence, the design columns generated by the following two statements are the same (but the ordering of the columns is different):

Note that nested effects are often distinguished from interaction effects by the implied randomization structure of the design. That is, they usually indicate random effects within a fixed-effects framework. The fact that random effects can be modeled directly in the RANDOM statement might make the specification of nested effects in the MODEL statement unnecessary.

See Table 20.7 in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics," for an example of the parameterization of a nested effect.

Implications of the Non-Full-Rank Parameterization

For models with fixed effects involving classification variables, there are more design columns in **X** constructed than there are degrees of freedom for the effect. Thus, there are linear dependencies among the columns of **X**. In this event, all of the parameters are not estimable; there is an infinite number of solutions to the mixed model equations. The GLIMMIX procedure uses a generalized inverse (a g_2-inverse, Pringle and Rayner (1971), to obtain values for the estimates (Searle 1971). The solution values are not displayed unless you specify the SOLUTION option in the MODEL statement. The solution has the characteristic that estimates are 0 whenever the design column for that parameter is a linear combination of previous columns. With this parameterization, hypothesis tests are constructed to test linear functions of the parameters that are estimable.

Some procedures (such as the CATMOD and LOGISTIC procedures) reparameterize models to full rank by using restrictions on the parameters. PROC GLM, PROC MIXED, and PROC GLIMMIX do not reparameterize, making the hypotheses that are commonly tested more understandable. See Goodnight (1978b) for additional reasons for not reparameterizing.

Missing Level Combinations

PROC GLIMMIX handles missing level combinations of classification variables in the same manner as PROC GLM and PROC MIXED. These procedures delete fixed-effects parameters corresponding to missing levels in order to preserve estimability. However, PROC GLIMMIX does not delete missing level combinations for random-effects parameters because linear combinations of the random-effects parameters are always predictable. These conventions can affect the way you specify your CONTRAST and ESTIMATE coefficients.

Notes on the EFFECT Statement

Some restrictions and limitations for models that contain constructed effects are in place with the GLIMMIX procedure. Also, you should be aware of some special defaults and handling that apply only when the model contains constructed fixed and/or random effects.

- Constructed effects can be used in the MODEL and RANDOM statements but not to specify SUBJECT= or GROUP= effects.
- Computed variables are not supported in the specification of a constructed effect. All variables needed to form the collection of columns for a constructed effect must be in the data set.
- You cannot use constructed effects that comprise continuous variables or interactions with other constructed effects as the LSMEANS or LSMESTIMATE effect.
- The calculation of quantities that depend on least squares means, such as odds ratios in the "Odds Ratio Estimates" table, is not possible if the model contains fixed effects that consist of more than one constructed effects, unless all constructed effects are of spline type. For example, least squares means computations are not possible in the following model because the MM_AB*cvars effect contains two constructed effects:

```
proc glimmix;
   class A B C;
   effect MM_AB = MM(A B);
   effect cvars = COLLECTION(x1 x2 x3);
```

```
   model y = C MM_AB*cvars;
run;
```

- If the MODEL or RANDOM statement contains constructed effects, the default degrees-of-freedom method for mixed models is DDFM=KENWARDROGER. The containment degrees-of-freedom method (DDFM=CONTAIN) is not available in these models.

- If the model contains fixed spline effects, least squares means are computed at the average spline coefficients across the usable data, possibly further averaged over levels of CLASS variables that interact with the spline effects in the model. You can use the AT option in the LSMEANS and LSMESTIMATE statements to construct the splines for particular values of the covariates involved. Consider, for example, the following statements:

```
proc glimmix;
   class A;
   effect spl = spline(x);
   model y = A spl;
   lsmeans A;
   lsmeans A / at means;
   lsmeans A / at x=0.4;
run;
```

  Suppose that the spl effect contributes seven columns $[s_1, \ldots, s_7]$ to the **X** matrix. The least squares means coefficients for the spl effect in the first LSMEANS statement are $[\overline{s}_1, \ldots, \overline{s}_7]$ with the averages taken across the observations used in the analysis. The second LSMEANS statement computes the spline coefficient at the average value of x: $[s(\overline{x})_1, \ldots, s(\overline{x})_7]$. The final LSMEANS statement uses $[s(0.4)_1, \ldots, s(0.4)_7]$. Using the AT option for least squares means calculations with spline effects can resolve inestimability issues.

- Using a spline effect with B-spline basis in the RANDOM statement is **not** the same as using a penalized B-spline (P-spline) through the TYPE=PSPLINE option in the RANDOM statement. The following statement constructs a penalized B-spline by using mixed model methodology:

```
random x / type=pspline;
```

  The next set of statements defines a set of B-spline columns in the **Z** matrix with uncorrelated random effects and homogeneous variance:

```
effect bspline = spline(x);
random bspline / type=vc;
```

  This does not lead to a properly penalized fit. See the documentation on TYPE=PSPLINE about the construction of penalties for B-splines through the covariance matrix of random effects.

Positional and Nonpositional Syntax for Contrast Coefficients

When you define custom linear hypotheses with the CONTRAST or ESTIMATE statement, the GLIMMIX procedure sets up an **L** vector or matrix that conforms to the fixed-effects solutions or the fixed- and random-effects solutions. With the LSMESTIMATE statement, you specify coefficients of the matrix **K** that is then converted into a coefficient matrix that conforms to the fixed-effects solutions.

There are two methods for specifying the entries in a coefficient matrix (hereafter simply referred to as the **L** matrix), termed the positional and nonpositional methods. In the positional form, and this is the traditional method, you provide a list of values that occupy the elements of the **L** matrix associated with the effect in question in the order in which the values are listed. For traditional model effects comprising continuous and classification variables, the positional syntax is simpler in some cases (main effects) and more cumbersome in others (interactions). When you work with effects constructed through the EFFECT statement, the nonpositional syntax is essential.

Consider, for example, the following two-way model with interactions where factors A and B have three and two levels, respectively:

```
proc glimmix;
   class a b block;
   model y = a b a*b / ddfm=kr;
   random block a*block;
run;
```

To test the difference of the B levels at the second level of A with a CONTRAST statement (a slice), you need to assign coefficients 1 and –1 to the levels of B and to the levels of the interaction where A is at the second level. Two examples of equivalent CONTRAST statements by using positional and nonpositional syntax are as follows:

```
contrast 'B at A2' b 1 -1 a*b 0  0  1 -1      ;
contrast 'B at A2' b 1 -1 a*b [1 2 1] [-1 2 2];
```

Because A precedes B in the CLASS statement, the levels of the interaction are formed as $\alpha_1\beta_1, \alpha_1\beta_2, \alpha_2\beta_1, \alpha_2\beta_2, \ldots$. If B precedes A in the CLASS statement, you need to modify the coefficients accordingly:

```
proc glimmix;
   class b a block;
   model y = a b a*b / ddfm=kr;
   random block a*block;
   contrast 'B at A2' b 1 -1 a*b 0  1  0  0 -1     ;
   contrast 'B at A2' b 1 -1 a*b [1  1 2] [-1  2 2];
   contrast 'B at A2' b 1 -1 a*b [1, 1 2] [-1, 2 2];
run;
```

You can optionally separate the **L** value entry from the level indicators with a comma, as in the last CONTRAST statement.

The general syntax for defining coefficients with the nonpositional syntax is as follows:

effect-name [*multiplier* <,> *level-values*] ... < [*multiplier* <,> *level-values*] >

The first entry in square brackets is the multiplier that is applied to the elements of **L** for the effect after the *level-values* have been resolved and any necessary action forming **L** has been taken.

The *level-values* are organized in a specific form:

- The number of entries should equal the number of terms needed to construct the effect. For effects that do not contain any constructed effects, this number is simply the number of terms in the name of the effect.
- Values of continuous variables needed for the construction of the **L** matrix precede the level indicators of CLASS variables.
- If the effect involves constructed effects, then you need to provide as many continuous and classification variables as are needed for the effect formation. For example, if a grouping effect is defined as

  ```
  class c;
  effect v = vars(x1 x2 c);
  ```

  then a proper nonpositional syntax would be, for example,

  ```
  v [0.5,  0.2 0.3 3]
  ```

- If an effect contains both regular terms (old-style effects) and constructed effects, then the order of the coefficients is as follows: continuous values for old-style effects, class levels for CLASS variables in old-style effects, continuous values for constructed effects, and finally class levels needed for constructed effects.

  Assume that C has four levels so that effect v contributes six elements to the **L** matrix. When PROC GLIMMIX resolves this syntax, the values 0.2 and 0.3 are assigned to the positions for x1 and x2 and a 1 is associated with the third level of C. The resulting vector is then multiplied by 0.5 to produce

  $$[0.1 \quad 0.15 \quad 0 \quad 0 \quad 0.5 \quad 0]$$

Note that you enter the **levels** of the classification variables in the square brackets, not their formatted values. The ordering of the levels of CLASS variables can be gleaned from the "Class Level Information" table.

To specify values for continuous variables, simply give their value as one of the terms in the effect. The nonpositional syntax in the following ESTIMATE statement is read as "1-time the value 0.4 in the column associated with level 2 of A"

```
proc glimmix;
   class a;
   model y = a a*x / s;
   lsmeans a / e at x=0.4;
   estimate 'A2 at x=0.4' intercept 1 a 0 1 a*x [1,0.4 2] / e;
run;
```

Because the value before the comma serves as a multiplier, the same estimable function could also be constructed with the following statements:

```
estimate 'A2 at x=0.4' intercept 1 a 0 1 a*x [ 4,  0.1 2];
estimate 'A2 at x=0.4' intercept 1 a 0 1 a*x [ 2,  0.2 2];
estimate 'A2 at x=0.4' intercept 1 a 0 1 a*x [-1, -0.4 2];
```

Note that continuous variables needed to construct an effect are always listed before any CLASS variables.

When you work with constructed effects, the nonpositional syntax works in the same way. For example, the following model contains a classification effect and a B-spline. The first two ESTIMATE statements produce predicted values for level one of C when the continuous variable x takes on the values 20 and 10, respectively.

```
proc glimmix;
   class c;
   effect spl = spline(x / knotmethod=equal(5));
   model y = c spl;
   estimate 'C = 1 @ x=20' intercept 1 c 1 spl [1,20],
            'C = 1 @ x=10' intercept 1 c 1 spl [1,10];
   estimate 'Difference'   spl [1,20] [-1,10];
run;
```

The GLIMMIX procedure computes the spline coefficients for the first ESTIMATE statement based on $x = 20$, and similarly in the second statement for $x = 10$. The third ESTIMATE statement computes the difference of the predicted values. Because the spline effect does not interact with the classification variable, this difference does not depend on the level of C. If such an interaction is present, you can estimate the difference in predicted values for a given level of C by using the nonpositional syntax. Because the effect C*spl contains both old-style terms (C) and a constructed effect, you specify the values for the old-style terms before assigning values to constructed effects:

```
proc glimmix;
   class c;
   effect spl = spline(x / knotmethod=equal(5));
   model y = spl*c;
   estimate 'C2 = 1, x=20' intercept 1 c*spl [1,1 20];
   estimate 'C2 = 2, x=20' intercept 1 c*spl [1,2 20];
   estimate 'C diff at x=20' c*spl [1,1 20] [-1,2 20];
run;
```

It is recommended to add the E option to the CONTRAST, ESTIMATE, or LSMESTIMATE statement to verify that the **L** matrix is formed according to your expectations.

In any row of an ESTIMATE or CONTRAST statement you can choose positional and nonpositional syntax separately for each effect. You cannot mix the two forms of syntax for coefficients of a single effect, however. For example, the following statement is not proper because both forms of syntax are used for the interaction effect:

```
estimate 'A1B1 - A1B2' b 1 -1  a*b 0 1  [-1, 1 2];
```

Response-Level Ordering and Referencing

In models for binary and multinomial data, the response-level ordering is important because it reflects the following:

- which probability is modeled with binary data
- how categories are ordered for ordinal data
- which category serves as the reference category in nominal generalized logit models (models for nominal data)

You should view the "Response Profile" table to ensure that the categories are properly arranged and that the desired outcome is modeled. In this table, response levels are arranged by *Ordered Value*. The lowest response level is assigned Ordered Value 1, the next lowest is assigned Ordered Value 2, and so forth. In binary models, the probability modeled is the probability of the response level with the lowest Ordered Value.

You can change which probability is modeled and the Ordered Value in the "Response Profile" table with the DESCENDING, EVENT=, ORDER=, and REF= response variable options in the MODEL statement. See the section "Response Level Ordering" on page 6026 in Chapter 79, "The LOGISTIC Procedure," for examples about how to use these options to affect the probability being modeled for binary data.

For multinomial models, the response-level ordering affects two important aspects. In cumulative link models the categories are assumed ordered according to their Ordered Value in the "Response Profile" table. If the response variable is a character variable or has a format, you should check this table carefully as to whether the Ordered Values reflect the correct ordinal scale.

In generalized logit models (for multinomial data with unordered categories), one response category is chosen as the reference category in the formulation of the generalized logits. By default, the linear predictor in the reference category is set to 0, and the reference category corresponds to the entry in the "Response Profile" table with the highest Ordered Value. You can affect the assignment of Ordered Values with the DESCENDING and ORDER= options in the MODEL statement. You can choose a different reference category with the REF= option. The choice of the reference category for generalized logit models affects the results. It is sometimes recommended that you choose the category with the highest frequency as the reference (see, for example, Brown and Prescott 1999, p. 160). You can achieve this with the GLIMMIX procedure by combining the ORDER= and REF= options, as in the following statements:

```
proc glimmix;
   class preference;
   model preference(order=freq ref=first) = feature price /
                   dist=multinomial
                   link=glogit;
   random intercept / subject=store group=preference;
run;
```

The ORDER=FREQ option arranges the categories by descending frequency. The REF=FIRST option then selects the response category with the lowest Ordered Value—the most frequent category—as the reference.

Comparing the GLIMMIX and MIXED Procedures

The MIXED procedure is subsumed by the GLIMMIX procedure in the following sense:

- Linear mixed models are a special case in the family of generalized linear mixed models; a linear mixed model is a generalized linear mixed model where the conditional distribution is normal and the link function is the identity function.
- Most models that can be fit with the MIXED procedure can also be fit with the GLIMMIX procedure.

Despite this overlap in functionality, there are also some important differences between the two procedures. Awareness of these differences enables you to select the most appropriate tool in situations where you have a choice between procedures and to identify situations where a choice does not exist. Furthermore, the %GLIMMIX macro, which fits generalized linear mixed models by linearization methods, essentially calls the MIXED procedure repeatedly. If you are aware of the syntax differences between the procedures, you can easily convert your %GLIMMIX macro statements.

Important functional differences between PROC GLIMMIX and PROC MIXED for linear models and linear mixed models include the following:

- The MIXED procedure models R-side effects through the REPEATED statement and G-side effects through the RANDOM statement. The GLIMMIX procedure models all random components of the model through the RANDOM statement. You use the _RESIDUAL_ keyword or the RESIDUAL option in the RANDOM statement to model R-side covariance structure in the GLIMMIX procedure. For example, the PROC MIXED statement

  ```
  repeated / subject=id type=ar(1);
  ```

  is equivalent to the following RANDOM statement in the GLIMMIX procedure:

  ```
  random _residual_ / subject=id type=ar(1);
  ```

  If you need to specify an effect for levelization—for example, because the construction of the **R** matrix is order-dependent or because you need to account for missing values—the RESIDUAL option in the RANDOM statement of the GLIMMIX procedure is used to indicate that you are modeling an R-side covariance nature. For example, the PROC MIXED statements

  ```
  class time id;
  repeated time / subject=id type=ar(1);
  ```

  are equivalent to the following PROC GLIMMIX statements:

  ```
  class time id;
  random time / subject=id type=ar(1) residual;
  ```

- There is generally considerable overlap in the covariance structures available through the TYPE= option in the RANDOM statement in PROC GLIMMIX and through the TYPE= options in the RANDOM and REPEATED statements in PROC MIXED. However, the Kronecker-type structures, the geometrically anisotropic spatial structures, and the GDATA= option in the RANDOM statement of the MIXED procedure are currently not supported in the GLIMMIX procedure. The MIXED procedure, on the other hand, does not support TYPE=RSMOOTH and TYPE=PSPLINE.

- For normal linear mixed models, the (default) METHOD=RSPL in PROC GLIMMIX is identical to the default METHOD=REML in PROC MIXED. Similarly, METHOD=MSPL in PROC GLIMMIX is identical for these models to METHOD=ML in PROC MIXED. The GLIMMIX procedure does not support Type I through Type III (ANOVA) estimation methods for variance component models. Also, the procedure does not have a METHOD=MIVQUE0 option, but you can produce these estimates through the NOITER option in the PARMS statement.

- The MIXED procedure solves the iterative optimization problem by means of a ridge-stabilized Newton-Raphson algorithm. With the GLIMMIX procedure, you can choose from a variety of optimization methods via the NLOPTIONS statement. The default method for most GLMMs is a quasi-Newton algorithm. A ridge-stabilized Newton-Raphson algorithm, akin to the optimization method in the MIXED procedure, is available in the GLIMMIX procedure through the TECHNIQUE=NRRIDG option in the NLOPTIONS statement. Because of differences in the line-search methods, update methods, and the convergence criteria, you might get slightly different estimates with the two procedures in some instances. The GLIMMIX procedure, for example, monitors several convergence criteria simultaneously.

- You can produce predicted values, residuals, and confidence limits for predicted values with both procedures. The mechanics are slightly different, however. With the MIXED procedure you use the OUTPM= and OUTP= options in the MODEL statement to write statistics to data sets. With the GLIMMIX procedure you use the OUTPUT statement and indicate with *keywords* which "flavor" of a statistic to compute.

- The following GLIMMIX statements are not available in the MIXED procedure: COVTEST, EFFECT, FREQ, LSMESTIMATE, OUTPUT, and programming statements.

- A sampling-based Bayesian analysis as through the PRIOR statement in the MIXED procedure is not available in the GLIMMIX procedure.

- In the GLIMMIX procedure, several RANDOM statement options apply to the RANDOM statement in which they are specified. For example, the following statements in the GLIMMIX procedure request that the solution vector be printed for the A and A*B*C random effects and that the **G** matrix corresponding to the A*B interaction random effect be displayed:

```
random a     / s;
random a*b   / G;
random a*b*c / alpha=0.04;
```

  Confidence intervals with a 0.96 coverage probability are produced for the solutions of the A*B*C effect. In the MIXED procedure, the S option, for example, when specified in one RANDOM statement, applies to all RANDOM statements.

- If you select nonmissing values in the *value-list* of the DDF= option in the MODEL statement, PROC GLIMMIX uses these values to override degrees of freedom for this effect that might be determined otherwise. For example, the following statements request that the denominator degrees of freedom for tests and confidence intervals involving the A effect be set to 4:

```
proc glimmix;
   class block a b;
   model y = a b a*b / s ddf=4,.,. ddfm=satterthwaite;
   random block a*block / s;
   lsmeans a b a*b  / diff;
run;
```

  In the example, this applies to the "Type III Tests of Fixed Effects," "Least Squares Means," and "Differences of Least Squares Means" tables. In the MIXED procedure, the Satterthwaite approximation overrides the DDF= specification.

- The DDFM=BETWITHIN degrees-of-freedom method in the GLIMMIX procedure requires that the data be processed by subjects; see the section "Processing by Subjects" on page 3957.

- When you add the response variable to the CLASS statement, PROC GLIMMIX defaults to the multinomial distribution. Adding the response variable to the CLASS statement in PROC MIXED has no effect on the fitted model.

- The ODS name of the table for the solution of fixed effects is SolutionF in the MIXED procedure. In PROC GLIMMIX, the name of the table that contains fixed-effects solutions is ParameterEstimates. In generalized linear models, this table also contains scale parameters and overdispersion parameters. The MIXED procedure always produces a "Covariance Parameter Estimates" table. The GLIMMIX procedure produces this table only in mixed models or models with nontrivial R-side covariance structure.

- If you compute predicted values in the GLIMMIX procedure in a model with only R-side random components and missing values for the dependent variable, the predicted values will *not* be kriging predictions as is the case with the MIXED procedure.

Singly or Doubly Iterative Fitting

Depending on the structure of your model, the GLIMMIX procedure determines the appropriate approach for estimating the parameters of the model. The elementary algorithms fall into three categories:

1. **Noniterative algorithms**
   A closed form solution exists for all model parameters. Standard linear models with homoscedastic, uncorrelated errors can be fit with noniterative algorithms.

2. **Singly iterative algorithms**
   A single optimization, consisting of one or more iterations, is performed to obtain solutions for the parameter estimates by numerical techniques. Linear mixed models for normal data can be fit with singly iterative algorithms. Laplace and quadrature estimation for generalized linear mixed models uses a singly iterative algorithm with a separate suboptimization to compute the random-effects solutions as modes of the log-posterior distribution.

3. **Doubly iterative algorithms**
   A model of simpler structure is derived from the target model. The parameters of the simpler model are estimated by noniterative or singly iterative methods. Based on these new estimates, the model of simpler structure is rederived and another estimation step follows. The process continues until changes in the parameter estimates are sufficiently small between two recomputations of the simpler model or until some other criterion is met. The rederivation of the model can often be cast as a change of the response to some pseudo-data along with an update of implicit model weights.

Obviously, noniterative algorithms are preferable to singly iterative ones, which in turn are preferable to doubly iterative algorithms. Two drawbacks of doubly iterative algorithms based on linearization are that likelihood-based measures apply to the pseudo-data, not the original data, and that at the outer level the progress of the algorithm is tied to monitoring the parameter estimates. The advantage of doubly iterative algorithms, however, is to offer—at convergence—the statistical inference tools that apply to the simpler models.

The output and log messages contain information about which algorithm is employed. For a noniterative algorithm, PROC GLIMMIX produces a message that no optimization was performed. Noniterative algorithms are employed automatically for normal data with identity link.

You can determine whether a singly or doubly iterative algorithm was used, based on the "Iteration History" table and the "Convergence Status" table (Figure 52.17).

Figure 52.17 Iteration History and Convergence Status in Singly Iterative Fit

The GLIMMIX Procedure

Iteration History

Iteration	Restarts	Evaluations	Objective Function	Change	Max Gradient
0	0	4	83.039723731	.	13.63536
1	0	3	82.189661988	0.85006174	0.281308
2	0	3	82.189255211	0.00040678	0.000174
3	0	3	82.189255211	0.00000000	1.05E-10

Convergence criterion (GCONV=1E-8) satisfied.

The "Iteration History" table contains the Evaluations column that shows how many function evaluations were performed in a particular iteration. The convergence status message informs you which convergence criterion was met when the estimation process concluded. In a singly iterative fit, the criterion is one that applies to the optimization. In other words, it is one of the criteria that can be controlled with the NLOPTIONS statement: see the ABSCONV=, ABSFCONV=, ABSGCONV=, ABSXCONV=, FCONV=, or GCONV= option.

In a doubly iterative fit, the "Iteration History" table does not contain an Evaluations column. Instead it displays the number of iterations within an optimization (Subiterations column in Figure 52.18).

Figure 52.18 Iteration History and Convergence Status in Doubly Iterative Fit

Iteration History					
Iteration	Restarts	Subiterations	Objective Function	Change	Max Gradient
0	0	5	79.688580269	0.11807224	7.851E-7
1	0	3	81.294622554	0.02558021	8.209E-7
2	0	2	81.438701534	0.00166079	4.061E-8
3	0	1	81.444083567	0.00006263	2.273E-8
4	0	1	81.444265216	0.00000421	0.000025
5	0	1	81.444277364	0.00000383	0.000023
6	0	1	81.444266322	0.00000348	0.000021
7	0	1	81.44427636	0.00000316	0.000019
8	0	1	81.444267235	0.00000287	0.000017
9	0	1	81.44427553	0.00000261	0.000016
10	0	1	81.44426799	0.00000237	0.000014
11	0	1	81.444274844	0.00000216	0.000013
12	0	1	81.444268614	0.00000196	0.000012
13	0	1	81.444274277	0.00000178	0.000011
14	0	1	81.444269129	0.00000162	9.772E-6
15	0	0	81.444273808	0.00000000	9.102E-6

Convergence criterion (PCONV=1.11022E-8) satisfied.

The Iteration column then counts the number of optimizations. The "Convergence Status" table indicates that the estimation process concludes when a criterion is met that monitors the parameter estimates across optimization, namely the PCONV= or ABSPCONV= criterion.

You can control the optimization process with the GLIMMIX procedure through the NLOPTIONS statement. Its options affect the individual optimizations. In a doubly iterative scheme, these apply to all optimizations.

The default optimization techniques are TECHNIQUE=NONE for noniterative estimation, TECHNIQUE=NEWRAP for singly iterative methods in GLMs, TECHNIQUE=NRRIDG for pseudo-likelihood estimation with binary data, and TECHNIQUE=QUANEW for other mixed models.

Default Estimation Techniques

Based on the structure of the model, the GLIMMIX procedure selects the estimation technique for estimating the model parameters. If you fit a generalized linear mixed model, you can change the estimation technique with the METHOD= option in the PROC GLIMMIX statement. The defaults are determined as follows:

- generalized linear model
  - normal distribution: restricted maximum likelihood
  - all other distributions: maximum likelihood
- generalized linear model with overdispersion
  Parameters ($\boldsymbol{\beta}$; ϕ, if present) are estimated by (restricted) maximum likelihood as for generalized linear

models. The overdispersion parameter is estimated from the Pearson statistic after all other parameters have been estimated.

- generalized linear mixed models
  The default technique is METHOD=RSPL, corresponding to maximizing the residual log pseudo-likelihood with an expansion about the current solutions of the best linear unbiased predictors of the random effects. In models for normal data with identity link, METHOD=RSPL and METHOD=RMPL are equivalent to restricted maximum likelihood estimation, and METHOD=MSPL and METHOD=MMPL are equivalent to maximum likelihood estimation. This is reflected in the labeling of statistics in the "Fit Statistics" table.

Default Output

The following sections describe the output that PROC GLIMMIX produces by default. The output is organized into various tables, which are discussed in the order of appearance. Note that the contents of a table can change with the estimation method or the model being fit.

Model Information

The "Model Information" table displays basic information about the fitted model, such as the link and variance functions, the distribution of the response, and the data set. If important model quantities—for example, the response, weights, link, or variance function—are user-defined, the "Model Information" table displays the final assignment to the respective variable, as determined from your programming statements. If the table indicates that the variance matrix is blocked by an effect, then PROC GLIMMIX processes the data by subjects. The "Dimensions" table displays the number of subjects. For more information about processing by subjects, see the section "Processing by Subjects" on page 3957. The ODS name of the "Model Information" table is ModelInfo.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels with the ORDER= option in the PROC GLIMMIX statement. The ODS name of the "Class Level Information" table is ClassLevels.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used. If the *events/trials* syntax is used for the response, the table also displays the number of events and trials used in the analysis. The ODS name of the "Number of Observations" table is NObs.

Response Profile

For binary and multinomial data, the "Response Profile" table displays the Ordered Value from which the GLIMMIX procedure determines the following:

- the probability being modeled for binary data
- the ordering of categories for ordinal data
- the reference category for generalized logit models

For each response category level, the frequency used in the analysis is reported. The section "Response-Level Ordering and Referencing" on page 3976 explains how you can use the DESCENDING, EVENT=, ORDER=, and REF= options to affect the assignment of Ordered Values to the response categories. The ODS name of the "Response Profile" table is ResponseProfile.

Dimensions

The "Dimensions" table displays information from which you can determine the size of relevant matrices in the model. This table is useful in determining CPU time and memory requirements. The ODS name of the "Dimensions" table is "Dimensions."

Optimization Information

The "Optimization Information" table displays important details about the optimization process.

The optimization technique that is displayed in the table is the technique that applies to any single optimization. For singly iterative methods that is *the* optimization method.

The number of parameters that are updated in the optimization equals the number of parameters in this table minus the number of equality constraints. The number of constraints is displayed if you fix covariance parameters with the HOLD= option in the PARMS statement. The GLIMMIX procedure also lists the number of upper and lower boundary constraints. Note that the procedure might impose boundary constraints for certain parameters, such as variance components and correlation parameters. Covariance parameters for which a HOLD= was issued have an upper and lower boundary equal to the parameter value.

If a residual scale parameter is profiled from the optimization, it is also shown in the "Optimization Information" table.

In a GLMM for which the parameters are estimated by one of the linearization methods, you need to initiate the process of computing the pseudo-response. This can be done based on existing estimates of the fixed effects, or by using the data themselves—possibly after some suitable adjustment—as an estimate of the initial mean. The default in PROC GLIMMIX is to use the data themselves to derive initial estimates of the mean function and to construct the pseudo-data. The "Optimization Information" table shows how the pseudo-data are determined initially. Note that this issue is separate from the determination of starting values for the covariance parameters. These are computed as minimum variance quadratic unbiased estimates (with 0 priors, MIVQUE0; Goodnight 1978a) or obtained from the *value-list* in the PARMS statement.

The ODS name of the table is OptInfo.

Iteration History

The "Iteration History" table describes the progress of the estimation process. In singly iterative methods, the table displays the following:

- the iteration count, Iteration
- the number of restarts, Restarts
- the number of function evaluations, Evaluations
- the objective function, Objective
- the change in the objective function, Change
- the absolute value of the largest (projected) gradient, MaxGradient

Note that the change in the objective function is not the convergence criterion monitored by the GLIMMIX procedure. PROC GLIMMIX tracks several convergence criteria simultaneously; see the ABSCONV=, ABSFCONV=, ABSGCONV=, ABSXCONV=, FCONV=, or GCONV= option in the NLOPTIONS statement.

For doubly iterative estimation methods, the "Iteration History" table does not display the progress of the individual optimizations; instead, it reports on the progress of the outer iterations. Every row of the table then corresponds to an update of the linearization, the computation of a new set of pseudo-data, and a new optimization. In the listing, PROC GLIMMIX displays the following:

- the optimization count, Iteration
- the number of restarts, Restarts
- the number of iterations per optimization, Subiterations
- the change in the parameter estimates, Change
- the absolute value of the largest (projected) gradient at the end of the optimization, MaxGradient

By default, the change in the parameter estimates is expressed in terms of the relative PCONV criterion. If you request an absolute criterion with the ABSPCONV option of the PROC GLIMMIX statement, the change reflects the largest absolute difference since the last optimization.

If you specify the ITDETAILS option in the PROC GLIMMIX statement, parameter estimates and their gradients are added to the "Iteration History" table. The ODS name of the "Iteration History" table is IterHistory.

Convergence Status

The "Convergence Status" table contains a status message describing the reason for termination of the optimization. The message is also written to the log. The ODS name of the "Convergence Status" table is ConvergenceStatus, and you can query the nonprinting numeric variable Status to check for a successful optimization. This is useful in batch processing, or when processing BY groups, such as in simulations. Successful optimizations are indicated by the value 0 of the Status variable.

Fit Statistics

The "Fit Statistics" table provides statistics about the estimated model. The first entry of the table corresponds to the negative of twice the (possibly restricted) log likelihood, log pseudo-likelihood, or log quasi-likelihood. If the estimation method permits the true log likelihood or residual log likelihood, the description of the first entry reads accordingly. Otherwise, the fit statistics are preceded by the words *Pseudo-* or *Quasi-*, for Pseudo- and Quasi-Likelihood estimation, respectively.

Note that the (residual) log pseudo-likelihood in a GLMM is the (residual) log likelihood of a linearized model. You should not compare these values across different statistical models, even if the models are nested with respect to fixed and/or G-side random effects. It is possible that between two nested models the larger model has a smaller pseudo-likelihood. For this reason, IC=NONE is the default for GLMMs fit by pseudo-likelihood methods.

See the IC= option of the PROC GLIMMIX statement and Table 52.2 for the definition and computation of the information criteria reported in the "Fit Statistics" table.

For generalized linear models, the GLIMMIX procedure reports Pearson's chi-square statistic

$$X^2 = \sum_i \frac{w_i(y_i - \widehat{\mu}_i)^2}{a(\widehat{\mu}_i)}$$

where $a(\widehat{\mu}_i)$ is the variance function evaluated at the estimated mean.

For GLMMs, the procedure typically reports a generalized chi-square statistic,

$$X_g^2 = \widehat{\mathbf{r}}'\mathbf{V}(\widehat{\boldsymbol{\theta}}^*)^{-1}\widehat{\mathbf{r}}$$

so that the ratio of X^2 or X_g^2 and the degrees of freedom produces the usual residual dispersion estimate.

If the R-side scale parameter ϕ is not extracted from $\mathbf{V}$, the GLIMMIX procedure computes

$$X_g^2 = \widehat{\mathbf{r}}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\widehat{\mathbf{r}}$$

as the generalized chi-square statistic. This is the case, for example, if R-side covariance structures are varied by a GROUP= effect or if the scale parameter is not profiled for an R-side TYPE=CS, TYPE=SP, TYPE=AR, TYPE=TOEP, or TYPE=ARMA covariance structure.

For METHOD=LAPLACE, the generalized chi-square statistic is not reported. Instead, the Pearson statistic for the conditional distribution appears in the "Conditional Fit Statistics" table.

If your model contains smooth components (such as TYPE=RSMOOTH), then the "Fit Statistics" table also displays the residual degrees of freedom of the smoother. These degrees of freedom are computed as

$$df_{smooth,res} = f - \mathrm{trace}(\mathbf{S})$$

where $\mathbf{S}$ is the "smoother" matrix—that is, the matrix that produces the predicted values on the linked scale.

The ODS name of the "Fit Statistics" table is FitStatistics.

Covariance Parameter Estimates

In a GLMM, the "Covariance Parameter Estimates" table displays the estimates of the covariance parameters and their asymptotic standard errors. This table is produced only for generalized linear *mixed* models. In generalized linear models with scale parameter, or when an overdispersion parameter is present, the estimates of parameters related to the dispersion are displayed in the "Parameter Estimates" table.

The standard error of the covariance parameters is determined from the diagonal entries of the asymptotic variance matrix of the covariance parameter estimates. You can display this matrix with the ASYCOV option in the PROC GLIMMIX statement.

The ODS name of the "Covariance Parameter Estimates" table is CovParms.

Type III Tests of Fixed Effects

The "Type III Tests of Fixed Effects" table contains hypothesis tests for the significance of each of the fixed effects specified in the MODEL statement. By default, PROC GLIMMIX computes these tests by first constructing a Type III $\mathbf{L}$ matrix for each effect; see Chapter 16, "The Four Types of Estimable Functions." The $\mathbf{L}$ matrix is then used to construct the test statistic

$$F = \frac{\widehat{\boldsymbol{\beta}}'\mathbf{L}'(\mathbf{L}\mathbf{Q}\mathbf{L}')^{-1}\mathbf{L}\widehat{\boldsymbol{\beta}}}{\mathrm{rank}(\mathbf{L}\mathbf{Q}\mathbf{L}')}$$

where the matrix $\mathbf{Q}$ depends on the estimation method and options. For example, in a GLMM, the default is $\mathbf{Q} = (\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{X})^{-}$, where $\mathbf{V}(\boldsymbol{\theta})$ is the marginal variance of the pseudo-response. If you specify the DDFM=KENWARDROGER option, $\mathbf{Q}$ is the estimated variance matrix of the fixed effects, adjusted by the method of Kenward and Roger (1997). If the EMPIRICAL= option is in effect, $\mathbf{Q}$ corresponds to the selected sandwich estimator.

You can use the HTYPE= option in the MODEL statement to obtain tables of Type I (sequential) tests and Type II (adjusted) tests in addition to or instead of the table of Type III (partial) tests.

The ODS names of the "Type I Tests of Fixed Effects" through the "Type III Tests of Fixed Effects" tables are Tests1 through Tests3, respectively.

Notes on Output Statistics

Table 52.17 lists the statistics computed with the OUTPUT statement of the GLIMMIX procedure and their default names. This section provides further details about these statistics.

The distinction between prediction and confidence limits in Table 52.17 stems from the involvement of the predictors of the random effects. If the random-effect solutions (BLUPs, EBES) are involved, then the associated standard error used in computing the limits are standard errors of prediction rather than standard errors of estimation. The prediction limits are *not* limits for the prediction of a new observation.

The Pearson residuals in Table 52.17 are "Pearson-type" residuals, because the residuals are standardized by the square root of the marginal or conditional variance of an observation. Traditionally, Pearson residuals in generalized linear models are divided by the square root of the variance function. The GLIMMIX procedure divides by the square root of the variance so that marginal and conditional residuals have similar expressions. In other words, scale and overdispersion parameters are included.

When residuals or predicted values involve only the fixed effects part of the linear predictor (that is, $\widehat{\eta}_m = \mathbf{x}'\widehat{\boldsymbol{\beta}}$), then all model quantities are computed based on this predictor. For example, if the variance by which to standardize a marginal residual involves the variance function, then the variance function is also evaluated at the marginal mean, $g^{-1}(\widehat{\eta}_m)$. Thus the residuals $p - \widehat{\eta}$ and $p_m - \widehat{\eta}_m$ can also be expressed as $(y - \mu)/\partial\mu$ and $(y - \mu_m)/\partial\mu_m$, respectively, where $\partial\mu$ is the derivative with respect to the linear predictor. To construct the residual $p - \widehat{\eta}_m$ in a GLMM, you can add the value of _ZGAMMA_ to the conditional residual $p - \widehat{\eta}$. (The residual $p - \widehat{\eta}_m$ is computed instead of the default marginal residual when you specify the CPSEUDO option in the OUTPUT statement.) If the predictor involves the BLUPs, then all relevant expressions and evaluations involve the conditional mean $g^{-1}(\widehat{\eta})$.

The naming convention to add "PA" to quantities not involving the BLUPs is chosen to suggest the concept of a population average. When the link function is nonlinear, these are not truly population-averaged quantities, because $g^{-1}(\mathbf{x}'\boldsymbol{\beta})$ does not equal $\mathrm{E}[Y]$ in the presence of random effects. For example, if

$$\mu_i = g^{-1}(\mathbf{x}_i'\boldsymbol{\beta} + \mathbf{z}_i'\boldsymbol{\gamma}_i)$$

is the conditional mean for subject i, then

$$g^{-1}(\mathbf{x}_i'\widehat{\boldsymbol{\beta}})$$

does not estimate the average response in the population of subjects but the response of the average subject (the subject for which $\boldsymbol{\gamma}_i = \mathbf{0}$). For models with identity link, the average response and the response of the average subject are identical.

The GLIMMIX procedure obtains standard errors on the scale of the mean by the delta method. If the link is a nonlinear function of the linear predictor, these standard errors are only approximate. For example,

$$\mathrm{Var}[g^{-1}(\widehat{\eta}_m)] \doteq \left(\frac{\partial g^{-1}(t)}{\partial t}\bigg|_{\widehat{\eta}_m}\right)^2 \mathrm{Var}[\widehat{\eta}_m]$$

Confidence limits on the scale of the data are usually computed by applying the inverse link function to the confidence limits on the linked scale. The resulting limits on the data scale have the same coverage probability as the limits on the linked scale, but they are possibly asymmetric.

In generalized logit models, confidence limits on the mean scale are based on symmetric limits about the predicted mean in a category. Suppose that the multinomial response in such a model has J categories. The probability of a response in category i is computed as

$$\widehat{\mu}_i = \frac{\exp\{\widehat{\eta}_i\}}{\sum_{j=1}^{J}\exp\{\widehat{\eta}_i\}}$$

The variance of $\widehat{\mu}_i$ is then approximated as

$$\mathrm{Var}[\widehat{\mu}_i] \doteq \zeta = \boldsymbol{v}_i'\mathrm{Var}\begin{bmatrix}\widehat{\eta}_1 & \widehat{\eta}_2 & \cdots & \widehat{\eta}_J\end{bmatrix}\boldsymbol{v}_i$$

where $\boldsymbol{v}_i$ is a $J \times 1$ vector with kth element

$$\begin{array}{ll}\widehat{\mu}_i(1-\widehat{\mu}_i) & i = k\\ -\widehat{\mu}_i\widehat{\mu}_k & i \neq k\end{array}$$

The confidence limits in the generalized logit model are then obtained as

$$\widehat{\mu}_i \pm t_{\nu,\alpha/2}\sqrt{\zeta}$$

where $t_{\nu,\alpha/2}$ is the $100(1-\alpha/2)$th percentile from a t distribution with ν degrees of freedom. Confidence limits are truncated if they fall outside the [0, 1] interval.

ODS Table Names

Each table created by PROC GLIMMIX has a name associated with it, and you must use this name to reference the table when you use ODS statements. These names are listed in Table 52.27.

Table 52.27 ODS Tables Produced by PROC GLIMMIX

Table Name	Description	Required Statement / Option
AsyCorr	Asymptotic correlation matrix of covariance parameters	PROC GLIMMIX ASYCORR
AsyCov	Asymptotic covariance matrix of covariance parameters	PROC GLIMMIX ASYCOV
CholG	Cholesky root of the estimated **G** matrix	RANDOM / GC
CholV	Cholesky root of blocks of the estimated **V** matrix	RANDOM / VC
ClassLevels	Level information from the CLASS statement	Default output
Coef	**L** matrix coefficients	E option in MODEL, CONTRAST, ESTIMATE, LSMESTIMATE, or LSMEANS; ELSM option in LSMESTIMATE
ColumnNames	Name association for OUTDESIGN data set	PROC GLIMMIX OUTDESIGN(NAMES)
CondFitStatistics	Conditional fit statistics	PROC GLIMMIX METHOD=LAPLACE
Contrasts	Results from the CONTRAST statements	CONTRAST
ConvergenceStatus	Status of optimization at conclusion	Default output
CorrB	Approximate correlation matrix of fixed-effects parameter estimates	MODEL / CORRB
CovB	Approximate covariance matrix of fixed-effects parameter estimates	MODEL / COVB
CovBDetails	Details about model-based and/or adjusted covariance matrix of fixed effects	MODEL / COVB(DETAILS)
CovBI	Inverse of approximate covariance matrix of fixed-effects parameter estimates	MODEL / COVBI
CovBModelBased	Model-based (unadjusted) covariance matrix of fixed effects if DDFM=KR or EMPIRICAL option is used	MODEL / COVB(DETAILS)
CovParms	Estimated covariance parameters in GLMMs	Default output (in GLMMs)
CovTests	Results from COVTEST statements (except for confidence bounds)	COVTEST

Table 52.27 *continued*

Table Name	Description	Required Statement / Option
Diffs	Differences of LS-means	LSMEANS / DIFF (or PDIFF)
Dimensions	Dimensions of the model	Default output
Estimates	Results from ESTIMATE statements	ESTIMATE
FitStatistics	Fit statistics	Default
G	Estimated **G** matrix	RANDOM / G
GCorr	Correlation matrix from the estimated **G** matrix	RANDOM / GCORR
Hessian	Hessian matrix (observed or expected)	PROC GLIMMIX HESSIAN
InvCholG	Inverse Cholesky root of the estimated **G** matrix	RANDOM / GCI
InvCholV	Inverse Cholesky root of the blocks of the estimated **V** matrix	RANDOM / VCI
InvG	Inverse of the estimated **G** matrix	RANDOM / GI
InvV	Inverse of blocks of the estimated **V** matrix	RANDOM / VI
IterHistory	Iteration history	Default output
kdTree	*k*-d tree information	RANDOM / TYPE=RSMOOTH KNOTMETHOD= KDTREE(TREEINFO)
KnotInfo	Knot coordinates of low-rank spline smoother	RANDOM / TYPE=RSMOOTH KNOTINFO
LSMeans	LS-means	LSMEANS
LSMEstimates	Estimates among LS-means	LSMESTIMATE
LSMFtest	*F* test for LSMESTIMATEs	LSMESTIMATE / FTEST
LSMLines	Lines display for LS-means	LSMEANS / LINES
Margins	Predictive margins	MARGINS
MarginsDiffs	Differences of predictive margins	MARGINS / DIFF
MarginsSliceDiffs	Differences of sliced predictive margins	MARGINS / SLICEBY= SLICEDIFF=
MarginsSliceTests	Tests of sliced predictive margins	MARGINS / SLICEBY=
ModelInfo	Model information	Default output
NObs	Number of observations read and used, number of trials and events	Default output
OddsRatios	Odds ratios of parameter estimates	MODEL / ODDSRATIO
OptInfo	Optimization information	Default output
ParameterEstimates	Fixed-effects solution; overdispersion and scale parameter in GLMs	MODEL / S
ParmSearch	Parameter search values	PARMS
QuadCheck	Adaptive recalculation of quadrature approximation at solution	METHOD=QUAD(QCHECK)
ResponseProfile	Response categories and category modeled	Default output in models with binary or nominal response

Table 52.27 *continued*

Table Name	Description	Required Statement / Option
Slices	Tests of LS-means slices	LSMEANS / SLICE=
SliceDiffs	Differences of simple LS-means effects	LSMEANS / SLICEDIFF=
SolutionR	Random-effects solution vector	RANDOM / S
StandardizedCoefficients	Fixed-effects solutions from centered and/or scaled model	MODEL / STDCOEF
Tests1	Type I tests of fixed effects	MODEL / HTYPE=1
Tests2	Type II tests of fixed effects	MODEL / HTYPE=2
Tests3	Type III tests of fixed effects	Default output
V	Blocks of the estimated **V** matrix	RANDOM / V
VCorr	Correlation matrix from the blocks of the estimated **V** matrix	RANDOM / VCORR

The SLICE statement also creates tables, which are not listed in Table 52.27. For information about these tables, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

The following subsections provide information about the basic ODS statistical graphics produced by the GLIMMIX procedure. The graphics fall roughly into two categories: diagnostic plots and graphics for least squares means.

ODS Graph Names

The GLIMMIX procedure does not produce graphs by default. You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC GLIMMIX generates are listed in Table 52.28, along with the required statements and options.

Table 52.28 Graphs Produced by PROC GLIMMIX

ODS Graph Name	Plot Description	Option
AnomPlot	Plot of LS-mean differences against the average LS-mean	PLOTS=ANOMPLOT LSMEANS / PLOTS=ANOMPLOT
Boxplot	Box plots of residuals and/or observed values for model effects	PLOTS=BOXPLOT
ControlPlot	Plot of LS-mean differences against a control level	PLOTS=CONTROLPLOT LSMEANS / PLOTS=CONTROLPLOT
DiffPlot	Plot of LS-mean pairwise differences	PLOTS=DIFFPLOT LSMEANS / PLOTS=DIFFPLOT
MeanPlot	Plot of least squares means	PLOTS=MEANPLOT LSMEANS / PLOTS=MEANPLOT
ORPlot	Plot of odds ratios	PLOTS=ODDSRATIO
PearsonBoxplot	Box plot of Pearson residuals	PLOTS=PEARSONPANEL(UNPACK)
PearsonByPredicted	Pearson residuals vs. mean	PLOTS=PEARSONPANEL(UNPACK)
PearsonHistogram	Histogram of Pearson residuals	PLOTS=PEARSONPANEL(UNPACK)
PearsonPanel	Panel of Pearson residuals	PLOTS=PEARSONPANEL
PearsonQQplot	*Q*-*Q* plot of Pearson residuals	PLOTS=PEARSONPANEL(UNPACK)
ResidualBoxplot	Box plot of (raw) residuals	PLOTS=RESIDUALPANEL(UNPACK)
ResidualByPredicted	Residuals vs. mean or linear predictor	PLOTS=RESIDUALPANEL(UNPACK)
ResidualHistogram	Histogram of (raw) residuals	PLOTS=RESIDUALPANEL(UNPACK)
ResidualPanel	Panel of (raw) residuals	PLOTS=RESIDUALPANEL
ResidualQQplot	*Q*-*Q* plot of (raw) residuals	PLOTS=RESIDUALPANEL(UNPACK)
StudentBoxplot	Box plot of studentized residuals	PLOTS=STUDENTPANEL(UNPACK)
StudentByPredicted	Studentized residuals vs. mean or linear predictor	PLOTS=STUDENTPANEL(UNPACK)
StudentHistogram	Histogram of studentized residuals	PLOTS=STUDENTPANEL(UNPACK)
StudentPanel	Panel of studentized residuals	PLOTS=STUDENTPANEL
StudentQQplot	*Q*-*Q* plot of studentized residuals	PLOTS=STUDENTPANEL(UNPACK)

When ODS Graphics is enabled, the SLICE statement can produce plots that are associated with its analysis. For information about these plots, see the section "SLICE Statement" on page 550 in

Chapter 20, "Shared Concepts and Topics."

Diagnostic Plots

Residual Panels

There are three types of residual panels in the GLIMMIX procedure. Their makeup of four component plots is the same; the difference lies in the type of residual from which the panel is computed. Raw residuals are displayed with the PLOTS=RESIDUALPANEL option. Studentized residuals are displayed with the PLOTS=STUDENTPANEL option, and Pearson residuals with the PLOTS=PEARSONPANEL option. By default, conditional residuals are used in the construction of the panels if the model contains G-side random effects. For example, consider the following statements:

```
proc glimmix plots=residualpanel;
   class A;
   model y = x1 x2 / dist=Poisson;
   random int / sub=A;
run;
```

The parameters are estimated by a pseudo-likelihood method, and at the final stage pseudo-data are related to a linear mixed model with random intercepts. The residual panel is constructed from

$$r = p - \mathbf{x}'\widehat{\boldsymbol{\beta}} + \mathbf{z}'\widehat{\boldsymbol{\gamma}}$$

where p is the pseudo-data.

The following hypothetical data set contains yields of an industrial process. Material was available from five randomly selected vendors to produce a chemical reaction whose yield depends on two factors (pressure and temperature at 3 and 2 levels, respectively).

```
data Yields;
   input Vendor Pressure Temp Yield @@;
   datalines;
 1  1  1  10.20    1  1  2   9.48    1  2  1   9.74
 1  2  2   8.92    1  3  1  11.79    1  3  2   8.85
 2  1  1  10.43    2  1  2  10.59    2  2  1  10.29
 2  2  2  10.15    2  3  1  11.12    2  3  2   9.30
 3  1  1   6.46    3  1  2   7.34    3  2  1   9.44
 3  2  2   8.11    3  3  1   9.38    3  3  2   8.37
 4  1  1   7.36    4  1  2   9.92    4  2  1  10.99
 4  2  2  10.34    4  3  1  10.24    4  3  2   9.96
 5  1  1  11.72    5  1  2  10.60    5  2  1  11.28
 5  2  2   9.03    5  3  1  14.09    5  3  2   8.92
;
```

Consider a linear mixed model with a two-way factorial fixed-effects structure for pressure and temperature effects and independent, homoscedastic random effects for the vendors. The following statements fit this model and request panels of marginal and conditional residuals:

```
ods graphics on;

proc glimmix data=Yields
      plots=residualpanel(conditional marginal);
   class Vendor Pressure Temp;
   model Yield = Pressure Temp Pressure*Temp;
```

```
   random vendor;
run;

ods graphics off;
```

The suboptions of the RESIDUALPANEL request produce two panels. The panel of conditional residuals is constructed from $y - \mathbf{x}'\widehat{\boldsymbol{\beta}} - \mathbf{z}'\widehat{\boldsymbol{\gamma}}$ (Figure 52.19). The panel of marginal residuals is constructed from $y - \mathbf{x}'\widehat{\boldsymbol{\beta}}$ (Figure 52.20). Note that these residuals are deviations from the observed data, because the model is a normal linear mixed model, and hence it does not involve pseudo-data. Whenever the random-effects solutions $\widehat{\boldsymbol{\gamma}}$ are involved in constructing residuals, the title of the residual graphics identifies them as conditional residuals (Figure 52.19).

Figure 52.19 Conditional Residuals

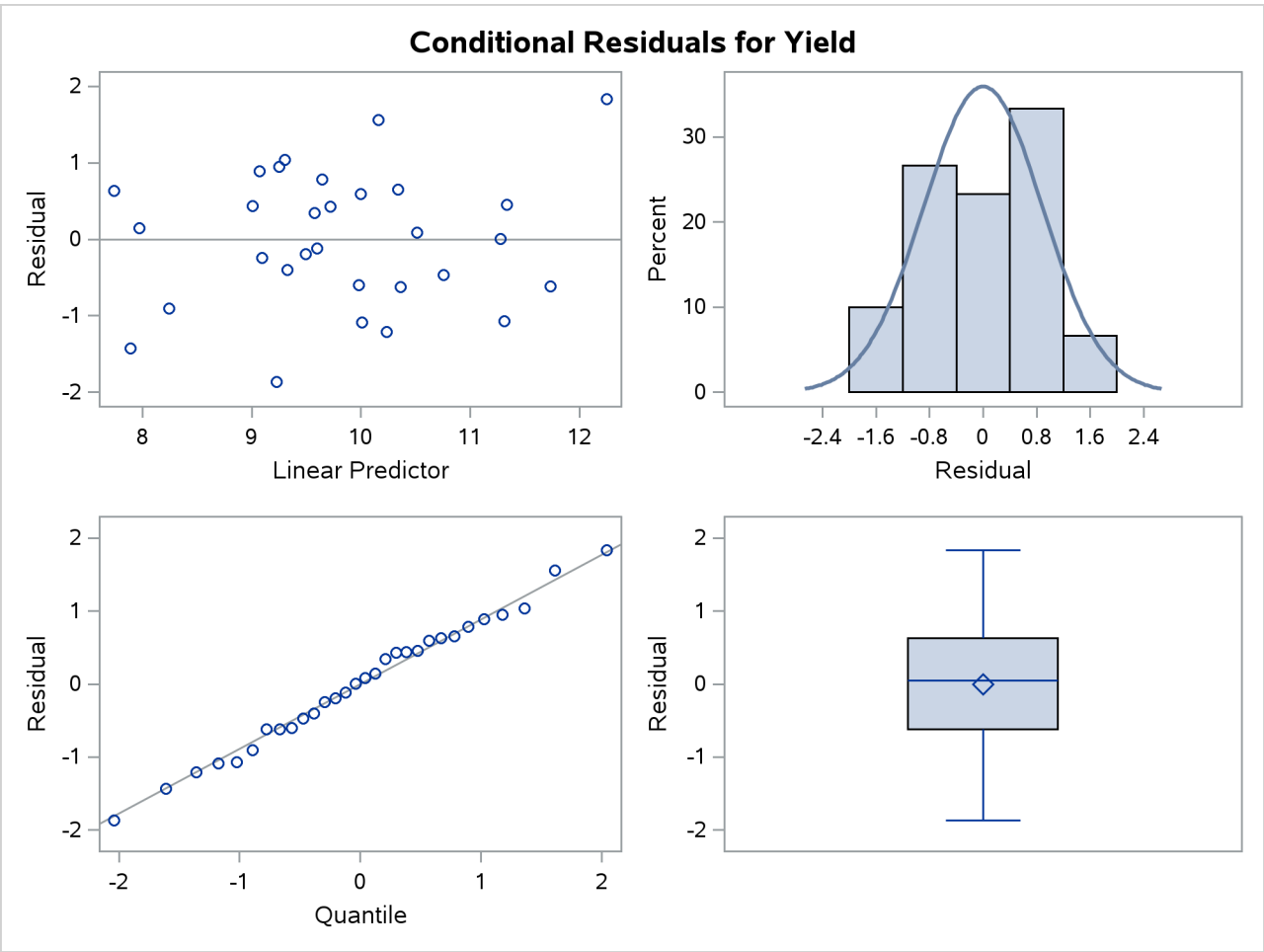

Figure 52.20 Marginal Residuals

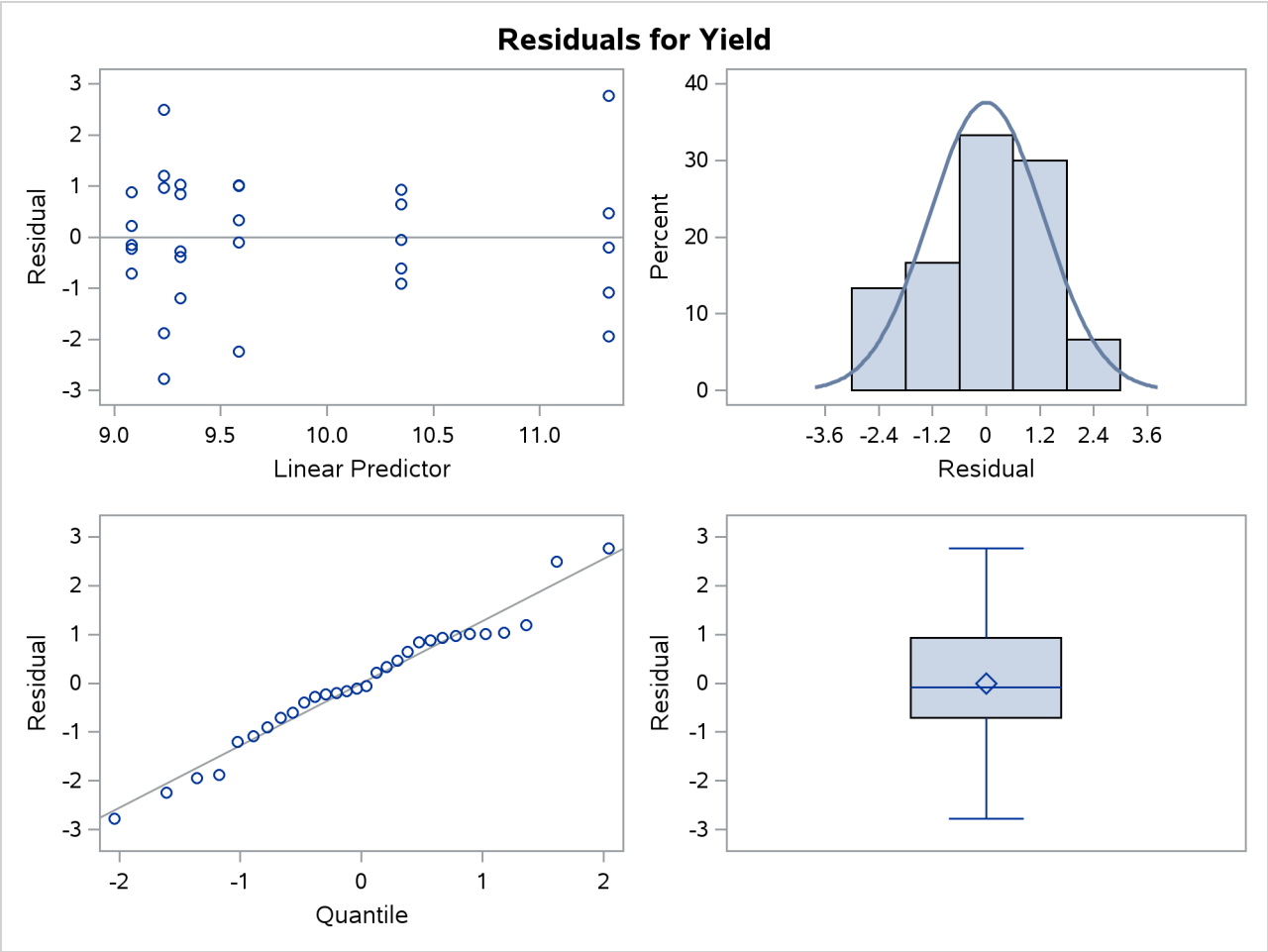

The predictor takes on only six values for the marginal residuals, corresponding to the combinations of three temperature and two pressure levels. The assumption of a zero mean for the vendor random effect seems justified; the marginal residuals in the upper-left plot of Figure 52.20 do not exhibit any trend. The conditional residuals in Figure 52.19 are smaller and somewhat closer to normality compared to the marginal residuals.

Box Plots

You can produce box plots of observed data, pseudo-data, and various residuals for effects in your model that consist of classification variables. Because you might not want to produce box plots for all such effects, you can request subsets with the suboptions of the BOXPLOT option in the PLOTS option. The BOXPLOT request in the following PROC GLIMMIX statement produces box plots for the random effects—in this case, the vendor effect. By default, PROC GLIMMIX constructs box plots from conditional residuals. The MARGINAL, CONDITIONAL, and OBSERVED suboptions instruct the procedure to construct three box plots for each random effect: box plots of the observed data (Figure 52.21), the marginal residuals (Figure 52.22), and the conditional residuals (Figure 52.23).

```
ods graphics on;

proc glimmix data=Yields
       plots=boxplot(random marginal conditional observed);
   class Vendor Pressure Temp;
   model Yield = Pressure Temp Pressure*Temp;
   random vendor;
run;

ods graphics off;
```

The observed vendor means in Figure 52.21 are different; in particular, vendors 3 and 5 appear to differ from the other vendors and from each other. There is also heterogeneity of variance in the five groups. The marginal residuals in Figure 52.22 continue to show the differences in means by vendor, because vendor enters the model as a random effect. The marginal means are adjusted for vendor effects only in the sense that the vendor variance component affects the marginal variance that is involved in the generalized least squares solution for the pressure and temperature effects.

Figure 52.21 Box Plots of Observed Values

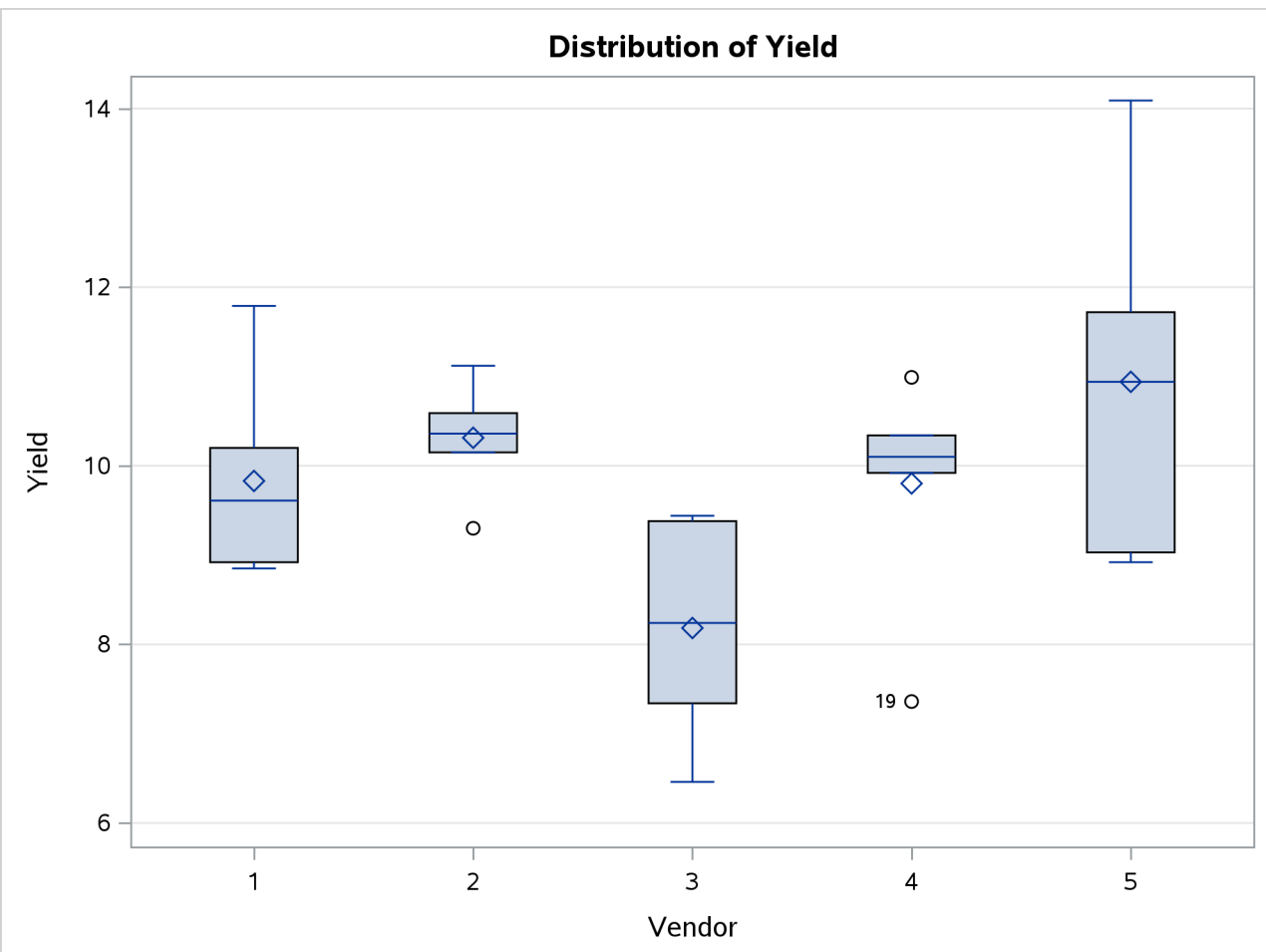

Figure 52.22 Box Plots of Marginal Residuals

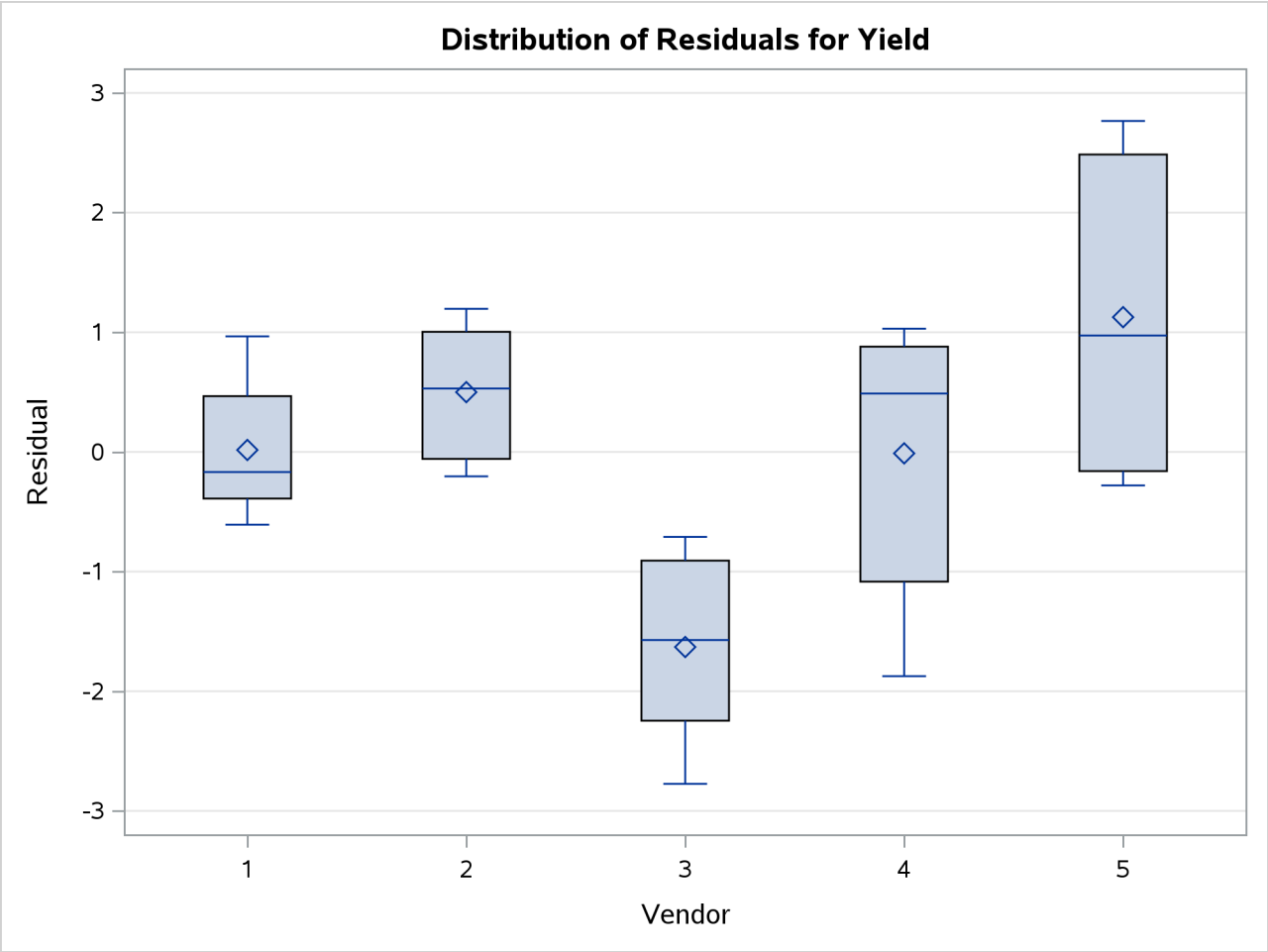

The conditional residuals account for the vendor effects through the empirical BLUPs. The means and medians have stabilized near zero, but some heterogeneity in these residuals remains (Figure 52.23).

Figure 52.23 Box Plots of Conditional Residuals

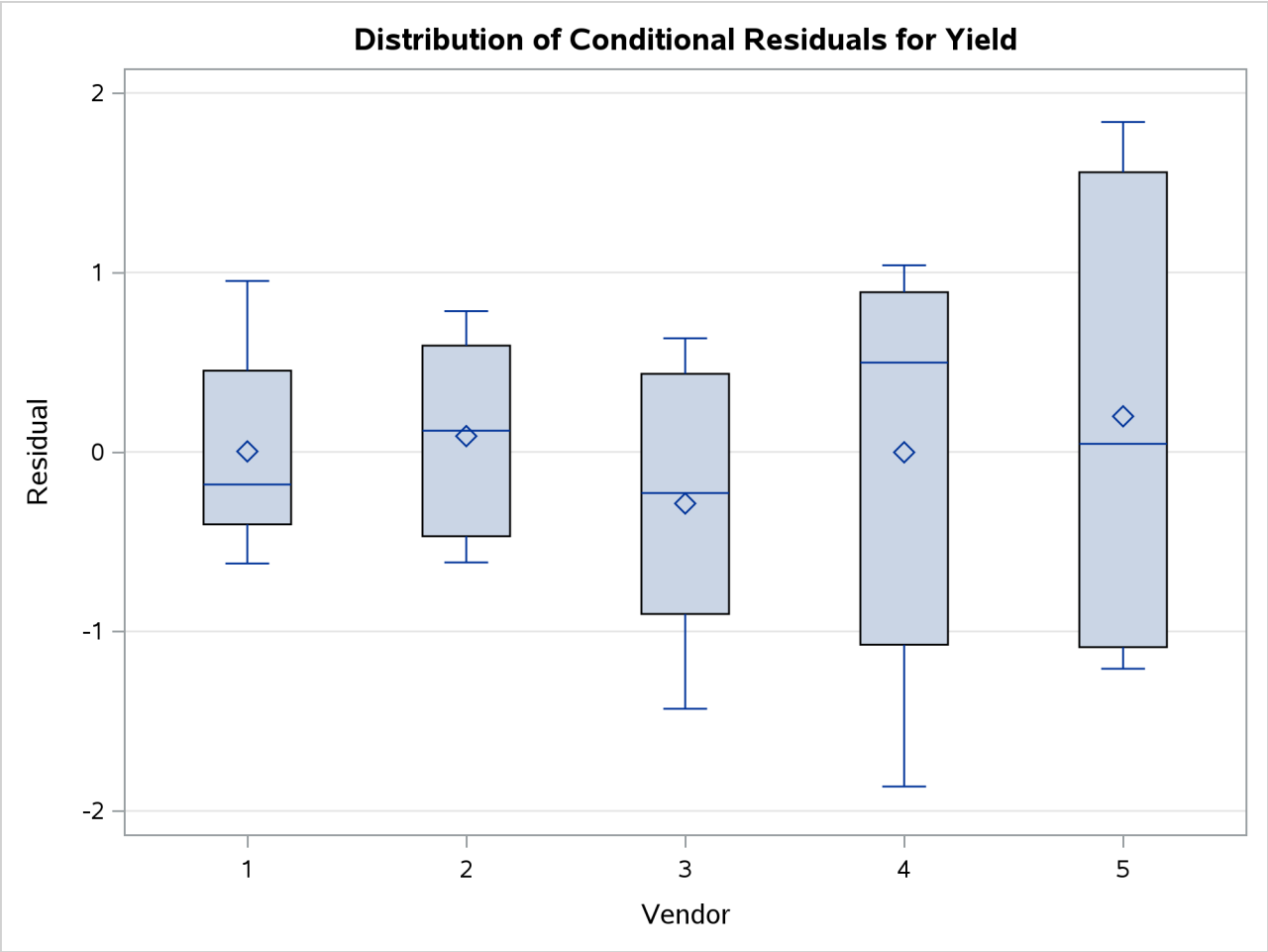

Graphics for LS-Mean Comparisons

The following subsections provide information about the ODS statistical graphics for least squares means produced by the GLIMMIX procedure. Mean plots display marginal or interaction means. The diffogram, control plot, and ANOM plot display least squares mean comparisons.

Mean Plots

The following SAS statements request a plot of the Pressure×Temp means in which the pressure trends are plotted for each temperature.

```
ods graphics on;
ods select CovParms Tests3 MeanPlot;
proc glimmix data=Yields;
   class Vendor Pressure Temp;
   model Yield = Pressure Temp Pressure*Temp;
   random Vendor;
   lsmeans Pressure*Temp / plot=mean(sliceby=Temp join);
run;
ods graphics off;
```

There is a significant effect of temperature and an interaction between pressure and temperature (Figure 52.24). Notice that the pressure main effect might be masked by the interaction. Because of the interaction, temperature comparisons depend on the pressure and vice versa. The mean plot option requests a display of the Pressure × Temp least squares means with separate trends for each temperature (Figure 52.25).

Figure 52.24 Tests for Fixed Effects

The GLIMMIX Procedure

Covariance Parameter Estimates

Cov Parm	Estimate	Standard Error
Vendor	0.8602	0.7406
Residual	1.1039	0.3491

Type III Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
Pressure	2	20	1.42	0.2646
Temp	1	20	6.48	0.0193
Pressure*Temp	2	20	3.82	0.0393

The interaction between the two effects is evident in the lack of parallelism in Figure 52.25. The masking of the pressure main effect can be explained by slopes of different sign for the two trends. Based on these results, inferences about the pressure effects are conducted for a specific temperature. For example, Figure 52.26 is produced by adding the following statement:

```
lsmeans pressure*temp / slicediff=temp slice=temp;
```

Figure 52.25 Interaction Plot for Pressure x Temperature

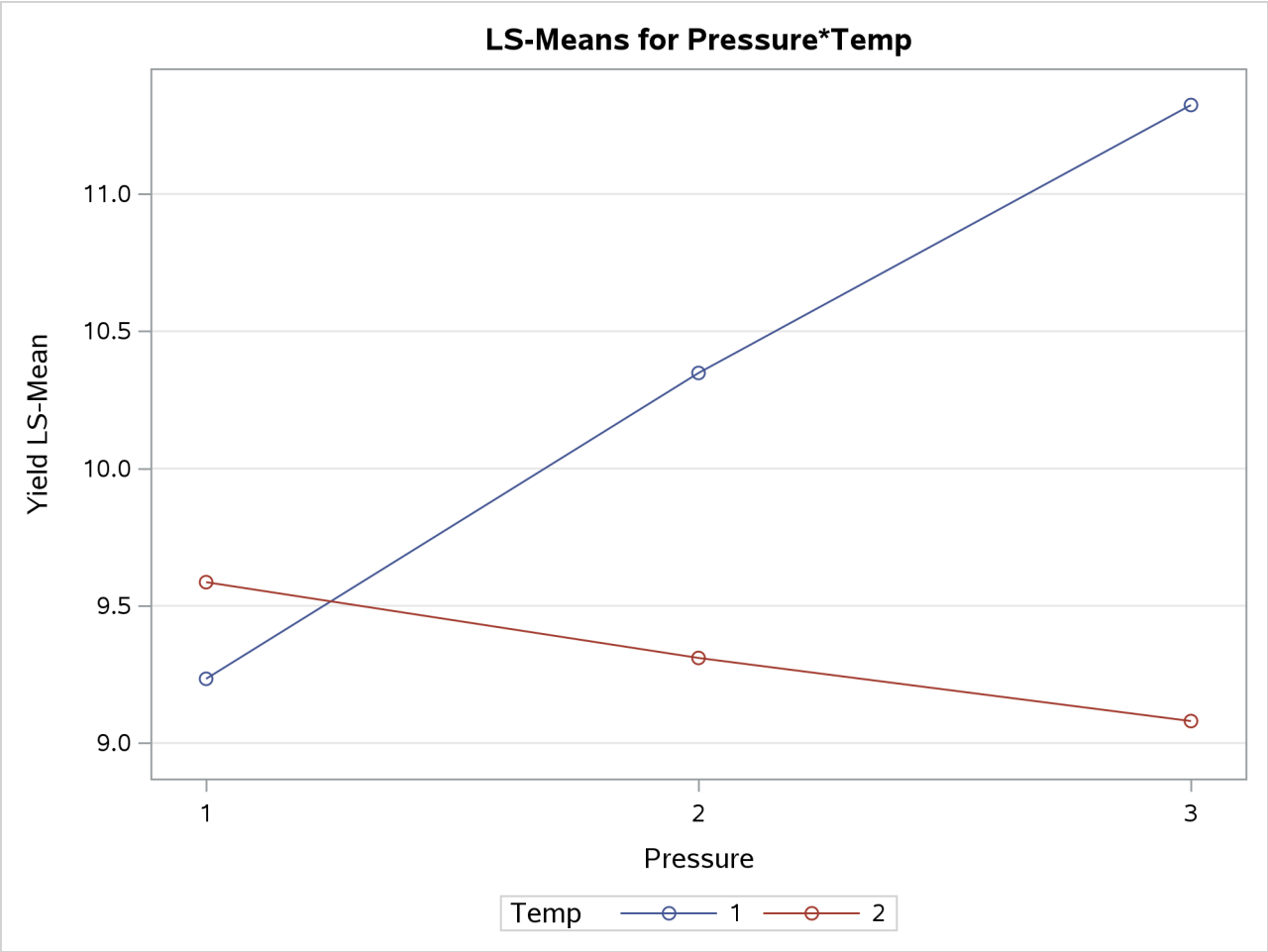

Figure 52.26 Pressure Comparisons at a Given Temperature

The GLIMMIX Procedure

Tests of Effect Slices for Pressure*Temp Sliced By Temp

Temp	Num DF	Den DF	F Value	Pr > F
1	2	20	4.95	0.0179
2	2	20	0.29	0.7508

Figure 52.26 *continued*

Simple Effect Comparisons of Pressure*Temp Least Squares Means By Temp							
Simple Effect Level	Pressure	_Pressure	Estimate	Standard Error	DF	t Value	Pr > |t|
Temp 1	1	2	-1.1140	0.6645	20	-1.68	0.1092
Temp 1	1	3	-2.0900	0.6645	20	-3.15	0.0051
Temp 1	2	3	-0.9760	0.6645	20	-1.47	0.1575
Temp 2	1	2	0.2760	0.6645	20	0.42	0.6823
Temp 2	1	3	0.5060	0.6645	20	0.76	0.4553
Temp 2	2	3	0.2300	0.6645	20	0.35	0.7329

The slope differences are evident by the change in sign for comparisons within temperature 1 and within temperature 2. There is a significant effect of pressure at temperature 1 ($p = 0.0179$), but not at temperature 2 ($p = 0.7508$).

Pairwise Difference Plot (Diffogram)

Graphical displays of LS-means-related analyses consist of plots of all pairwise differences (DiffPlot), plots of differences against a control level (ControlPlot), and plots of differences against an overall average (AnomPlot). The following data set is from an experiment to investigate how snapdragons grow in various soils (Stenstrom 1940). To eliminate the effect of local fertility variations, the experiment is run in blocks, with each soil type sampled in each block. See the "Examples" section of Chapter 53, "The GLM Procedure," for an in-depth analysis of these data.

```
data plants;
   input Type $ @;
   do Block = 1 to 3;
      input StemLength @;
      output;
   end;
   datalines;
Clarion   32.7 32.3 31.5
Clinton   32.1 29.7 29.1
Knox      35.7 35.9 33.1
ONeill    36.0 34.2 31.2
Compost   31.8 28.0 29.2
Wabash    38.2 37.8 31.9
Webster   32.5 31.1 29.7
;
```

The following statements perform the analysis of the experiment with the GLIMMIX procedure:

```
ods graphics on;
ods select LSMeans DiffPlot;

proc glimmix data=plants order=data plots=Diffogram;
   class Block Type;
   model StemLength = Block Type;
   lsmeans Type;
run;
```

```
ods graphics off;
```

The PLOTS= option in the PROC GLIMMIX statement requests that plots of pairwise least squares means differences are produced for effects that are listed in corresponding LSMEANS statements. This is the Type effect.

The Type LS-means are shown in Figure 52.27. Note that the order in which the levels appear corresponds to the order in which they were read from the data set. This was accomplished with the ORDER=DATA option in the PROC GLIMMIX statement.

Figure 52.27 Least Squares Means for Type Effect

The GLIMMIX Procedure

Type Least Squares Means

Type	Estimate	Standard Error	DF	t Value	Pr > |t|
Clarion	32.1667	0.7405	12	43.44	<.0001
Clinton	30.3000	0.7405	12	40.92	<.0001
Knox	34.9000	0.7405	12	47.13	<.0001
ONeill	33.8000	0.7405	12	45.64	<.0001
Compost	29.6667	0.7405	12	40.06	<.0001
Wabash	35.9667	0.7405	12	48.57	<.0001
Webster	31.1000	0.7405	12	42.00	<.0001

Because there are seven levels of Type in this analysis, there are $7(6-1)/2 = 21$ pairwise comparisons among the least squares means. The comparisons are performed in the following fashion: the first level of Type is compared against levels 2 through 7; the second level of Type is compared against levels 3 through 7; and so forth.

The default difference plot for these data is shown in Figure 52.28. The display is also known as a "mean-mean scatter plot" (Hsu 1996; Hsu and Peruggia 1994). It contains 21 lines rotated by 45 degrees counterclockwise, and a reference line (dashed 45-degree line). The (x, y) coordinate for the center of each line corresponds to the two least squares means being compared. Suppose that $\widehat{\eta}_{.i}$ and $\widehat{\eta}_{.j}$ denote the ith and jth least squares mean, respectively, for the effect in question, where $i < j$ according to the ordering of the effect levels. If the ABS option is in effect, which is the default, the line segment is centered at $(\min\{\widehat{\eta}_{i.}, \widehat{\eta}_{j.}\}, \max\{\widehat{\eta}_{i.}, \widehat{\eta}_{j.}\})$. Take, for example, the comparison of "Clarion" and "Compost" types. The respective estimates of their LS-means are $\widehat{\eta}_{.1} = 32.1667$ and $\widehat{\eta}_{.5} = 29.6667$. The center of the line segment for $H_0\colon \eta_{.1} = \eta_{.5}$ is placed at $(29.6667, 32.1667)$.

The length of the line segment for the comparison between means i and j corresponds to the width of the confidence interval for the difference $\eta_{.i} - \eta_{.j}$. This length is adjusted for the rotation in the plot. As a consequence, comparisons whose confidence interval covers zero cross the 45-degree reference line. These are the nonsignificant comparisons. Lines associated with significant comparisons do not touch or cross the reference line. Because these data are balanced, the estimated standard errors of all pairwise comparisons are identical, and the widths of the line segments are the same.

Figure 52.28 LS-Means Plot of Pairwise Differences

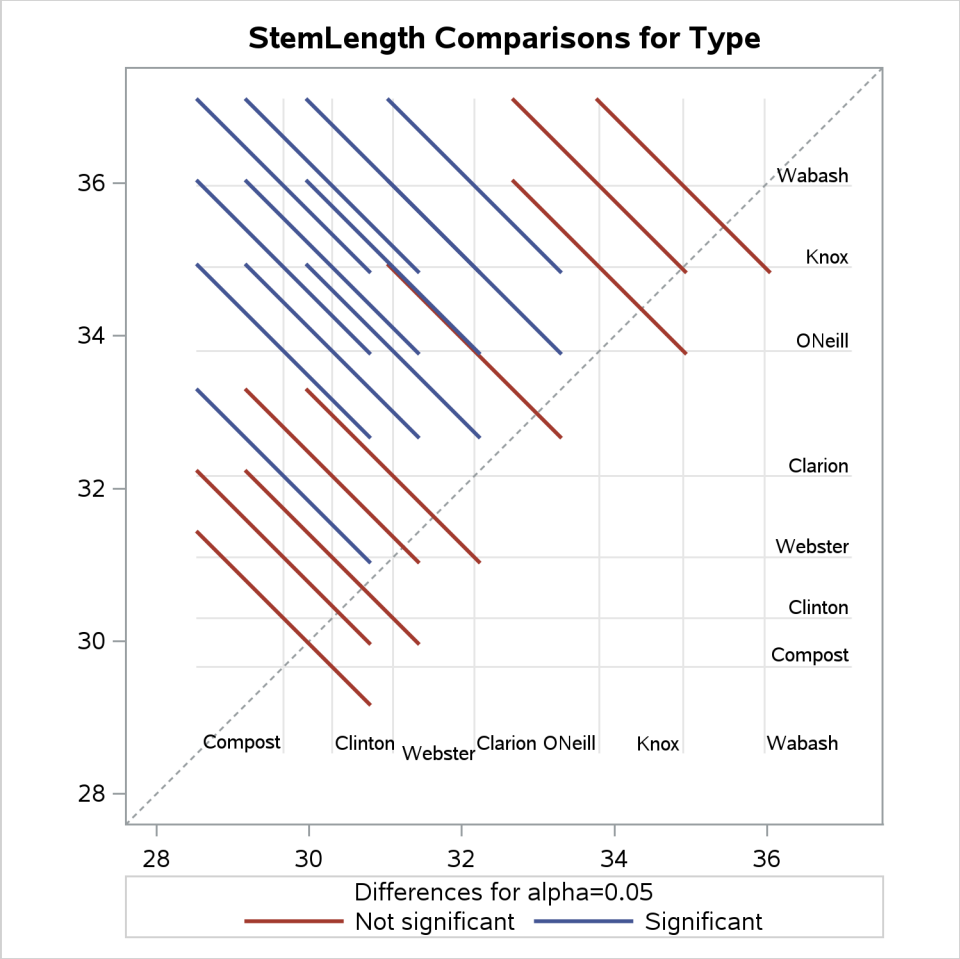

The background grid of the difference plot is drawn at the values of the least squares means for the seven type levels. These grid lines are used to find a particular comparison by intersection. Also, the labels of the grid lines indicate the ordering of the least squares means.

In the next set of statements, the NOABS and CENTER suboptions of the PLOTS=DIFFOGRAM option in the LSMEANS statement modify the appearance of the diffogram:

```
ods graphics on;
proc glimmix data=plants order=data;
   class Block Type;
   model StemLength = Block Type;
   lsmeans Type / plots=diffogram(noabs center);
run;
ods graphics off;
```

The NOABS suboption of the difference plot changes the way in which the GLIMMIX procedure places the line segments (Figure 52.29). If the NOABS suboption is in effect, the line segment is centered at the point $(\widehat{\eta}_{.i}, \widehat{\eta}_{.j})$, $i < j$. For example, the center of the line segment for a comparison of "Clarion" and "Compost" types is centered at $(\widehat{\eta}_{.1}, \widehat{\eta}_{.5}) = (32.1667, 29.6667)$. Whether a line segment appears above or below the reference line depends on the magnitude of the least squares means and the order of their appearance in the

"Least Squares Means" table. The CENTER suboption places a marker at the intersection of the least squares means.

Because the ABS option places lines on the same side of the 45-degree reference, it can help to visually discover groups of significant and nonsignificant differences. On the other hand, when the number of levels in the effect is large, the display can get crowded. The NOABS option can then provide a more accessible resolution.

Figure 52.29 Diffogram with NOABS and CENTER Options

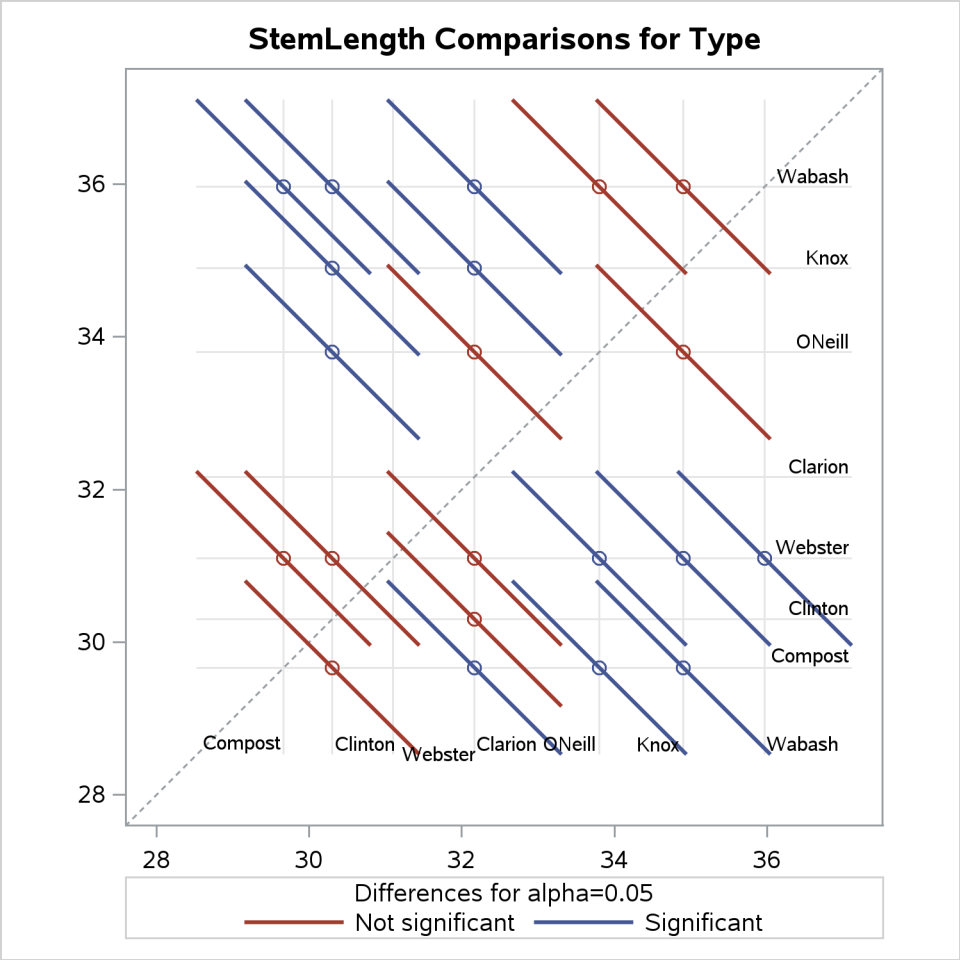

Least Squares Mean Control Plot

The following SAS statements create the same data set as before, except that one observation for Type="Knox" has been removed for illustrative purposes:

```
data plants;
   input Type $ @;
   do Block = 1 to 3;
      input StemLength @;
      output;
   end;
   datalines;
```

```
Clarion   32.7 32.3 31.5
Clinton   32.1 29.7 29.1
Knox      35.7 35.9   .
ONeill    36.0 34.2 31.2
Compost   31.8 28.0 29.2
Wabash    38.2 37.8 31.9
Webster   32.5 31.1 29.7
;
```

The following statements request control plots for effects in LSMEANS statements with compatible option:

```
ods graphics on;
ods select Diffs ControlPlot;

proc glimmix data=plants order=data plots=ControlPlot;
   class Block Type;
   model StemLength = Block Type;
   lsmeans Type / diff=control('Clarion') adjust=dunnett;
run;

ods graphics off;
```

The LSMEANS statement for the Type effect is compatible; it requests comparisons of Type levels against "Clarion," adjusted for multiplicity with Dunnett's method. Because "Clarion" is the first level of the effect, the LSMEANS statement is equivalent to

```
lsmeans type / diff=control adjust=dunnett;
```

The "Differences of Type Least Squares Means" table in Figure 52.30 shows the six comparisons between Type levels and the control level.

Figure 52.30 Least Squares Means Differences

The GLIMMIX Procedure

Differences of Type Least Squares Means
Adjustment for Multiple Comparisons: Dunnett

Type	_Type	Estimate	Standard Error	DF	t Value	Pr > |t|	Adj P
Clinton	Clarion	-1.8667	1.0937	11	-1.71	0.1159	0.3936
Knox	Clarion	2.7667	1.2430	11	2.23	0.0479	0.1854
ONeill	Clarion	1.6333	1.0937	11	1.49	0.1635	0.5144
Compost	Clarion	-2.5000	1.0937	11	-2.29	0.0431	0.1688
Wabash	Clarion	3.8000	1.0937	11	3.47	0.0052	0.0236
Webster	Clarion	-1.0667	1.0937	11	-0.98	0.3504	0.8359

The two rightmost columns of the table give the unadjusted and multiplicity-adjusted p-values. At the 5% significance level, both "Knox" and "Wabash" differ significantly from "Clarion" according to the unadjusted tests. After adjusting for multiplicity, only "Wabash" has a least squares mean significantly different from the control mean. Note that the standard error for the comparison involving "Knox" is larger than that for other comparisons because of the reduced sample size for that soil type.

In the plot of control differences a horizontal line is drawn at the value of the "Clarion" least squares mean. Vertical lines emanating from this reference line terminate in the least squares means for the other levels

(Figure 52.31).

The dashed upper and lower horizontal reference lines are the upper and lower decision limits for tests against the control level. If a vertical line crosses the upper or lower decision limit, the corresponding least squares mean is significantly different from the LS-mean in the control group. If the data had been balanced, the UDL and LDL would be straight lines, because all estimates $\widehat{\eta}_{.i} - \widehat{\eta}_{.j}$ would have had the same standard error. The limits for the comparison between "Knox" and "Clarion" are wider than for other comparisons, because of the reduced sample size for the "Knox" soil type.

Figure 52.31 LS-Means Plot of Differences against a Control

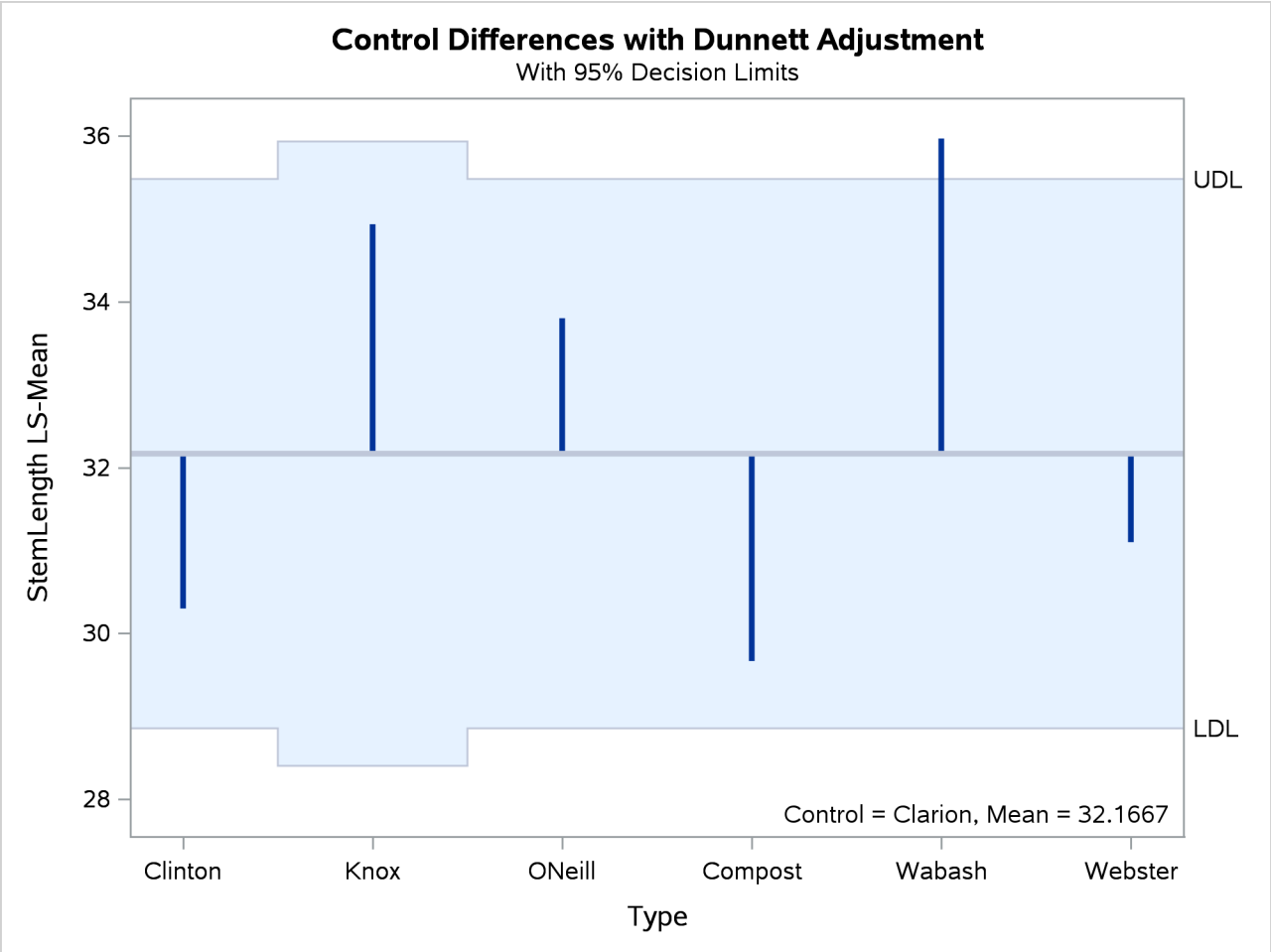

The significance level of the decision limits is determined from the ALPHA= level in the LSMEANS statement. The default are 95% limits. If you choose one-sided comparisons with DIFF=CONTROLL or DIFF=CONTROLU in the LSMEANS statement, only one of the decision limits is drawn.

Analysis of Means (ANOM) Plot

The analysis of means in PROC GLIMMIX compares least squares means not by contrasting them against each other as with all pairwise differences or control differences. Instead, the least squares means are compared against an average value. Consequently, there are k comparisons for a factor with k levels. The following statements request ANOM differences for the Type least squares means (Figure 52.32) and plots the differences (Figure 52.33):

```
ods graphics on;
ods select Diffs AnomPlot;
proc glimmix data=plants order=data plots=AnomPlot;
   class Block Type;
   model StemLength = Block Type;
   lsmeans Type / diff=anom;
run;
ods graphics off;
```

Figure 52.32 ANOM LS-Mean Differences

The GLIMMIX Procedure

Differences of Type Least Squares Means

Type	_Type	Estimate	Standard Error	DF	t Value	Pr > |t|
Clarion	Avg	-0.2635	0.7127	11	-0.37	0.7186
Clinton	Avg	-2.1302	0.7127	11	-2.99	0.0123
Knox	Avg	2.5032	0.9256	11	2.70	0.0205
ONeill	Avg	1.3698	0.7127	11	1.92	0.0809
Compost	Avg	-2.7635	0.7127	11	-3.88	0.0026
Wabash	Avg	3.5365	0.7127	11	4.96	0.0004
Webster	Avg	-1.3302	0.7127	11	-1.87	0.0888

At the 5% level, the “Clarion,” “O’Neill,” and “Webster” soil types are not significantly different from the average. Note that the artificial lack of balance introduced previously reduces the precision of the ANOM comparison for the “Knox” soil type.

Figure 52.33 LS-Means Analysis of Means (ANOM) Plot

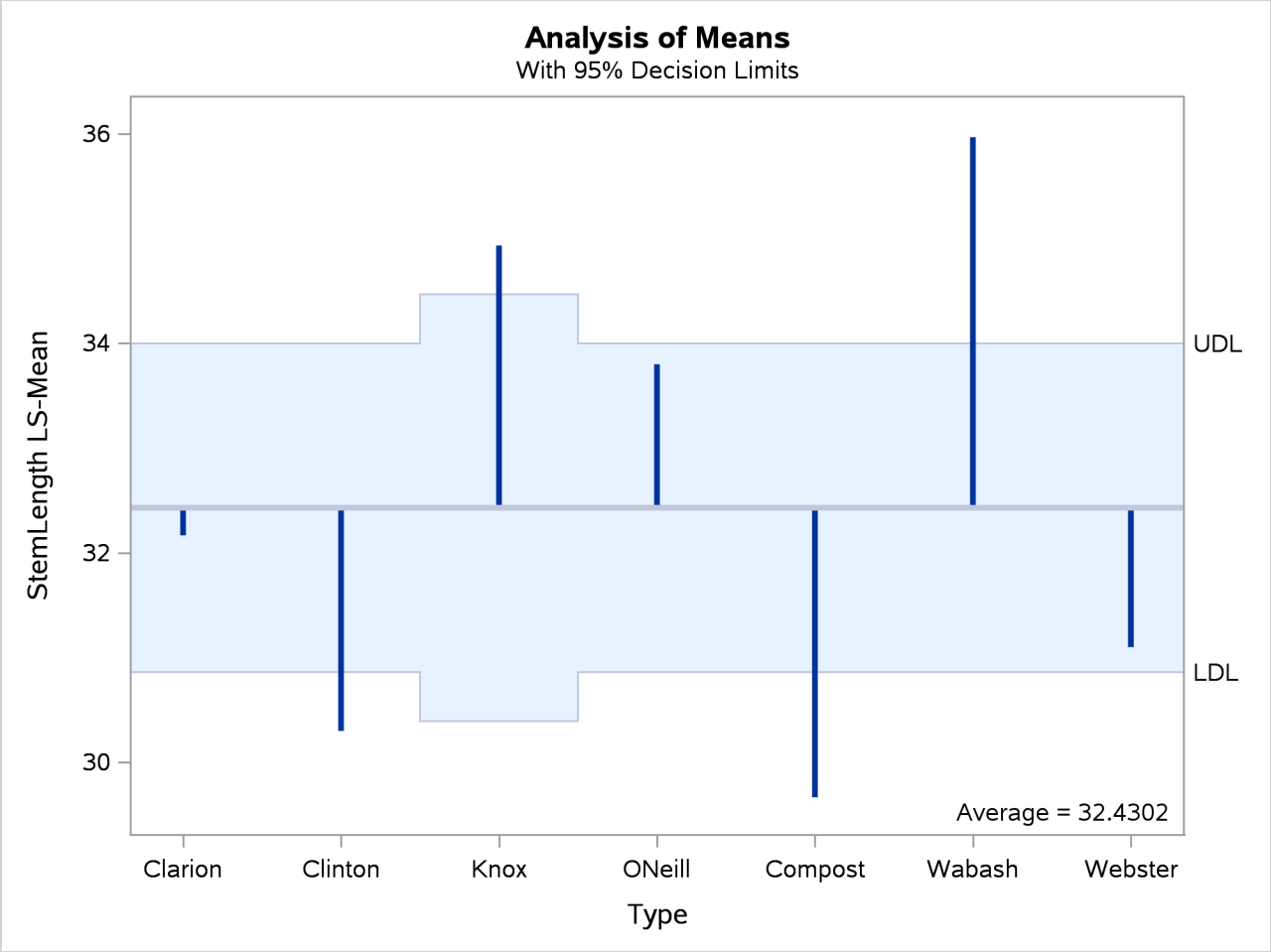

The reference line in the ANOM plot is drawn at the average. Vertical lines extend from this reference line upward or downward, depending on the magnitude of the least squares means compared to the reference value. This enables you to quickly see which levels perform above and below the average. The horizontal reference lines are 95% upper and lower decision limits. If a vertical line crosses the limits, you conclude that the least squares mean is significantly different (at the 5% significance level) from the average. You can adjust the comparisons for multiplicity by adding the ADJUST=NELSON option in the LSMEANS statement.

Examples: GLIMMIX Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 52.1: Binomial Counts in Randomized Blocks

(View the complete code for this example (gmxex01.sas) in the example repository.)

In the context of spatial prediction in generalized linear models, Gotway and Stroup (1997) analyze data from an agronomic field trial. Researchers studied 16 varieties (entries) of wheat for their resistance to infestation by the Hessian fly. They arranged the varieties in a randomized complete block design on an 8×8 grid. Each 4×4 quadrant of that arrangement constitutes a block.

The outcome of interest was the number of damaged plants (Y_{ij}) out of the total number of plants growing on the unit (n_{ij}). The two subscripts identify the block ($i = 1, \ldots, 4$) and the entry ($j = 1, \ldots, 16$). The following SAS statements create the data set. The variables lat and lng denote the coordinate of an experimental unit on the 8×8 grid.

```
data HessianFly;
   label Y = 'No. of damaged plants'
         n = 'No. of plants';
   input block entry lat lng n Y @@;
   datalines;
  1 14 1 1  8 2    1 16 1 2  9 1
  1  7 1 3 13 9    1  6 1 4  9 9
  1 13 2 1  9 2    1 15 2 2 14 7
  1  8 2 3  8 6    1  5 2 4 11 8
  1 11 3 1 12 7    1 12 3 2 11 8
  1  2 3 3 10 8    1  3 3 4 12 5
  1 10 4 1  9 7    1  9 4 2 15 8
  1  4 4 3 19 6    1  1 4 4  8 7
  2 15 5 1 15 6    2  3 5 2 11 9
  2 10 5 3 12 5    2  2 5 4  9 9
  2 11 6 1 20 10   2  7 6 2 10 8
  2 14 6 3 12 4    2  6 6 4 10 7
  2  5 7 1  8 8    2 13 7 2  6 0
  2 12 7 3  9 2    2 16 7 4  9 0
  2  9 8 1 14 9    2  1 8 2 13 12
  2  8 8 3 12 3    2  4 8 4 14 7
  3  7 1 5  7 7    3 13 1 6  7 0
  3  8 1 7 13 3    3 14 1 8  9 0
  3  4 2 5 15 11   3 10 2 6  9 7
  3  3 2 7 15 11   3  9 2 8 13 5
  3  6 3 5 16 9    3  1 3 6  8 8
  3 15 3 7  7 0    3 12 3 8 12 8
  3 11 4 5  8 1    3 16 4 6 15 1
  3  5 4 7 12 7    3  2 4 8 16 12
  4  9 5 5 15 8    4  4 5 6 10 6
  4 12 5 7 13 5    4  1 5 8 15 9
```

```
  4 15 6 5 17 6    4  6 6 6  8 2
  4 14 6 7 12 5    4  7 6 8 15 8
  4 13 7 5 13 2    4  8 7 6 13 9
  4  3 7 7  9 9    4 10 7 8  6 6
  4  2 8 5 12 8    4 11 8 6  9 7
  4  5 8 7 11 10   4 16 8 8 15 7
;
```

Analysis as a GLM

If infestations are independent among experimental units, and all plants within a unit have the same propensity for infestation, then the Y_{ij} are binomial random variables. The first model considered is a standard generalized linear model for independent binomial counts:

```
proc glimmix data=HessianFly;
   class block entry;
   model y/n = block entry / solution;
run;
```

The PROC GLIMMIX statement invokes the procedure. The CLASS statement instructs the GLIMMIX procedure to treat both block and entry as classification variables. The MODEL statement specifies the response variable and the fixed effects in the model. PROC GLIMMIX constructs the **X** matrix of the model from the terms on the right side of the MODEL statement. The GLIMMIX procedure supports two kinds of syntax for the response variable. This example uses the *events/trials* syntax. The variable y represents the number of successes (*events*) out of n Bernoulli *trials*. When the *events/trials* syntax is used, the GLIMMIX procedure automatically selects the binomial distribution as the response distribution. Once the distribution is determined, the procedure selects the link function for the model. The default link for binomial data is the logit link. The preceding statements are thus equivalent to the following statements:

```
proc glimmix data=HessianFly;
   class block entry;
   model y/n = block entry / dist=binomial link=logit solution;
run;
```

The SOLUTION option in the MODEL statement requests that solutions for the fixed effects (parameter estimates) be displayed.

The "Model Information" table describes the model and methods used in fitting the statistical model (Output 52.1.1).

The GLIMMIX procedure recognizes that this is a model for uncorrelated data (variance matrix is diagonal) and that parameters can be estimated by maximum likelihood. The default degrees-of-freedom method to denominator degrees of freedom for F tests and t tests is the RESIDUAL method. This corresponds to choosing $f - \mathrm{rank}(\mathbf{X})$ as the degrees of freedom, where f is the sum of the frequencies used in the analysis. You can change the degrees of freedom method with the DDFM= option in the MODEL statement.

Output 52.1.1 Model Information in GLM Analysis

The GLIMMIX Procedure

Model Information	
Data Set	WORK.HESSIANFLY
Response Variable (Events)	Y
Response Variable (Trials)	n
Response Distribution	Binomial
Link Function	Logit
Variance Function	Default
Variance Matrix	Diagonal
Estimation Technique	Maximum Likelihood
Degrees of Freedom Method	Residual

The "Class Level Information" table lists the levels of the variables specified in the CLASS statement and the ordering of the levels (Output 52.1.2). The "Number of Observations" table displays the number of observations read and used in the analysis.

Output 52.1.2 Class Level Information and Number of Observations

Class Level Information		
Class	**Levels**	**Values**
block	4	1 2 3 4
entry	16	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Number of Observations Read	64
Number of Observations Used	64
Number of Events	396
Number of Trials	736

The "Dimensions" table lists the size of relevant matrices (Output 52.1.3).

Output 52.1.3 Model Dimensions Information in GLM Analysis

Dimensions	
Columns in X	21
Columns in Z	0
Subjects (Blocks in V)	1
Max Obs per Subject	64

Because of the absence of G-side random effects in this model, there are no columns in the **Z** matrix. The 21 columns in the **X** matrix comprise the intercept, 4 columns for the block effect and 16 columns for the entry effect. Because no RANDOM statement with a SUBJECT= option was specified, the GLIMMIX procedure does not process the data by subjects (see the section "Processing by Subjects" on page 3957 for details about subject processing).

The "Optimization Information" table provides information about the methods and size of the optimization problem (Output 52.1.4).

Output 52.1.4 Optimization Information in GLM Analysis

Optimization Information	
Optimization Technique	Newton-Raphson
Parameters in Optimization	19
Lower Boundaries	0
Upper Boundaries	0
Fixed Effects	Not Profiled

With few exceptions, models fit with the GLIMMIX procedure require numerical methods for parameter estimation. The default optimization method for (overdispersed) GLM models is the Newton-Raphson algorithm. In this example, the optimization involves 19 parameters, corresponding to the number of linearly independent columns of the $\mathbf{X}'\mathbf{X}$ matrix.

The "Iteration History" table shows that the procedure converged after 3 iterations and 13 function evaluations (Output 52.1.5). The Change column measures the change in the objective function between iterations; however, this is not the monitored convergence criterion. The GLIMMIX procedure monitors several features simultaneously to determine whether to stop an optimization.

Output 52.1.5 Iteration History in GLM Analysis

Iteration History					
Iteration	**Restarts**	**Evaluations**	**Objective Function**	**Change**	**Max Gradient**
0	**0**	4	134.13393738	.	4.899609
1	**0**	3	132.85058236	1.28335502	0.206204
2	**0**	3	132.84724263	0.00333973	0.000698
3	**0**	3	132.84724254	0.00000009	3.029E-8

Convergence criterion (GCONV=1E-8) satisfied.

The "Fit Statistics" table lists information about the fitted model (Output 52.1.6). The –2 Log Likelihood values are useful for comparing nested models, and the information criteria AIC, AICC, BIC, CAIC, and HQIC are useful for comparing nonnested models. On average, the ratio between the Pearson statistic and its degrees of freedom should equal one in GLMs. Values larger than one indicate overdispersion. With a ratio of 2.37, these data appear to exhibit more dispersion than expected under a binomial model with block and varietal effects.

Output 52.1.6 Fit Statistics in GLM Analysis

Fit Statistics	
-2 Log Likelihood	265.69
AIC (smaller is better)	303.69
AICC (smaller is better)	320.97
BIC (smaller is better)	344.71
CAIC (smaller is better)	363.71
HQIC (smaller is better)	319.85
Pearson Chi-Square	106.74
Pearson Chi-Square / DF	2.37

The "Parameter Estimates" table displays the maximum likelihood estimates (Estimate), standard errors, and t tests for the hypothesis that the estimate is zero (Output 52.1.7).

Output 52.1.7 Parameter Estimates in GLM Analysis

Parameter Estimates

Effect	block	entry	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept			-1.2936	0.3908	45	-3.31	0.0018
block	1		-0.05776	0.2332	45	-0.25	0.8055
block	2		-0.1838	0.2303	45	-0.80	0.4289
block	3		-0.4420	0.2328	45	-1.90	0.0640
block	4		0	.	.	.	.
entry		1	2.9509	0.5397	45	5.47	<.0001
entry		2	2.8098	0.5158	45	5.45	<.0001
entry		3	2.4608	0.4956	45	4.97	<.0001
entry		4	1.5404	0.4564	45	3.38	0.0015
entry		5	2.7784	0.5293	45	5.25	<.0001
entry		6	2.0403	0.4889	45	4.17	0.0001
entry		7	2.3253	0.4966	45	4.68	<.0001
entry		8	1.3006	0.4754	45	2.74	0.0089
entry		9	1.5605	0.4569	45	3.42	0.0014
entry		10	2.3058	0.5203	45	4.43	<.0001
entry		11	1.4957	0.4710	45	3.18	0.0027
entry		12	1.5068	0.4767	45	3.16	0.0028
entry		13	-0.6296	0.6488	45	-0.97	0.3370
entry		14	0.4460	0.5126	45	0.87	0.3889
entry		15	0.8342	0.4698	45	1.78	0.0826
entry		16	0	.	.	.	.

The "Type III Tests of Fixed Effect" table displays significance tests for the two fixed effects in the model (Output 52.1.8).

Output 52.1.8 Type III Tests of Block and Entry Effects in GLM Analysis

Type III Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
block	3	45	1.42	0.2503
entry	15	45	6.96	<.0001

These tests are Wald-type tests, not likelihood ratio tests. The entry effect is clearly significant in this model with a p-value of <0.0001, indicating that the 16 wheat varieties are not equally susceptible to infestation by the Hessian fly.

Analysis with Random Block Effects

There are several possible reasons for the overdispersion noted in Output 52.1.6 (Pearson ratio = 2.37). The data might not follow a binomial distribution, one or more important effects might not have been accounted for in the model, or the data might be positively correlated. If important fixed effects have been omitted, then you might need to consider adding them to the model. Because this is a designed experiment, it is reasonable not to expect further effects apart from the block and entry effects that represent the treatment and error control design structure. The reasons for the overdispersion must lie elsewhere.

If overdispersion stems from correlations among the observations, then the model should be appropriately adjusted. The correlation can have multiple sources. First, it might not be the case that the plants within an experimental unit responded independently. If the probability of infestation of a particular plant is altered by the infestation of a neighboring plant within the same unit, the infestation counts are not binomial and a different probability model should be used. A second possible source of correlations is the lack of independence of experimental units. Even if treatments were assigned to units at random, they might not respond independently. Shared spatial soil effects, for example, can be the underlying factor. The following analyses take these spatial effects into account.

First, assume that the environmental effects operate at the scale of the blocks. By making the block effects random, the marginal responses will be correlated due to the fact that observations within a block share the same random effects. Observations from different blocks will remain uncorrelated, in the spirit of separate randomizations among the blocks. The next set of statements fits a generalized linear mixed model (GLMM) with random block effects:

```
proc glimmix data=HessianFly;
   class block entry;
   model y/n = entry / solution;
   random block;
run;
```

Because the conditional distribution—conditional on the block effects—is binomial, the marginal distribution will be overdispersed relative to the binomial distribution. In contrast to adding a multiplicative scale parameter to the variance function, treating the block effects as random changes the estimates compared to a model with fixed block effects.

In the presence of random effects and a conditional binomial distribution, PROC GLIMMIX does not use maximum likelihood for estimation. Instead, the GLIMMIX procedure applies a restricted (residual) pseudo-likelihood algorithm (Output 52.1.9). The “restricted” attribute derives from the same rationale by which restricted (residual) maximum likelihood methods for linear mixed models attain their name; the likelihood equations are adjusted for the presence of fixed effects in the model to reduce bias in covariance parameter estimates.

Output 52.1.9 Model Information in GLMM Analysis

The GLIMMIX Procedure

Model Information	
Data Set	WORK.HESSIANFLY
Response Variable (Events)	Y
Response Variable (Trials)	n
Response Distribution	Binomial
Link Function	Logit
Variance Function	Default
Variance Matrix	Not blocked
Estimation Technique	Residual PL
Degrees of Freedom Method	Containment

The "Class Level Information" and "Number of Observations" tables are as before (Output 52.1.10).

Output 52.1.10 Class Level Information and Number of Observations

Class Level Information		
Class	**Levels**	**Values**
block	4	1 2 3 4
entry	16	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Number of Observations Read	64
Number of Observations Used	64
Number of Events	396
Number of Trials	736

The "Dimensions" table indicates that there is a single G-side parameter, the variance of the random block effect (Output 52.1.11). The "Dimensions" table has changed from the previous model (compare Output 52.1.11 to Output 52.1.3). Note that although the block effect has four levels, only a single variance component is estimated. The **Z** matrix has four columns, however, corresponding to the four levels of the block effect. Because no SUBJECT= option is used in the RANDOM statement, the GLIMMIX procedure treats these data as having arisen from a single subject with 64 observations.

Output 52.1.11 Model Dimensions Information in GLMM Analysis

Dimensions	
G-side Cov. Parameters	1
Columns in X	17
Columns in Z	4
Subjects (Blocks in V)	1
Max Obs per Subject	64

The "Optimization Information" table indicates that a quasi-Newton method is used to solve the optimization problem. This is the default optimization method for GLMM models (Output 52.1.12).

Output 52.1.12 Optimization Information in GLMM Analysis

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	1
Lower Boundaries	1
Upper Boundaries	0
Fixed Effects	Profiled
Starting From	Data

In contrast to the Newton-Raphson method, the quasi-Newton method does not require second derivatives. Because the covariance parameters are not unbounded in this example, the procedure enforces a lower boundary constraint (zero) for the variance of the block effect, and the optimization method is changed to a dual quasi-Newton method. The fixed effects are profiled from the likelihood equations in this model. The resulting optimization problem involves only the covariance parameters.

The "Iteration History" table appears to indicate that the procedure converged after four iterations (Output 52.1.13). Notice, however, that this table has changed slightly from the previous analysis (see Output 52.1.5). The Evaluations column has been replaced by the Subiterations column, because the GLIMMIX procedure applied a doubly iterative fitting algorithm. The entire process consisted of five optimizations, each of which was iterative. The initial optimization required four iterations, the next one required three iterations, and so on.

Output 52.1.13 Iteration History in GLMM Analysis

Iteration History					
Iteration	**Restarts**	**Subiterations**	**Objective Function**	**Change**	**Max Gradient**
0	**0**	**4**	173.28473428	0.81019251	0.000197
1	**0**	**3**	181.66726674	0.17550228	0.000739
2	**0**	**2**	182.20789493	0.00614874	7.018E-6
3	**0**	**1**	182.21315596	0.00004386	1.213E-8
4	**0**	**0**	182.21317662	0.00000000	3.349E-6

Convergence criterion (PCONV=1.11022E-8) satisfied.

The "Fit Statistics" table shows information about the fit of the GLMM (Output 52.1.14). The log likelihood reported in the table is not the residual log likelihood of the data. It is the residual log likelihood for an approximated model. The generalized chi-square statistic measures the residual sum of squares in the final model, and the ratio with its degrees of freedom is a measure of variability of the observation about the mean model.

Output 52.1.14 Fit Statistics in GLMM Analysis

Fit Statistics	
-2 Res Log Pseudo-Likelihood	182.21
Generalized Chi-Square	107.96
Gener. Chi-Square / DF	2.25

The variance of the random block effects is rather small (Output 52.1.15).

Output 52.1.15 Estimated Covariance Parameters and Approximate Standard Errors

Covariance Parameter Estimates		
Cov Parm	Estimate	Standard Error
block	0.01116	0.03116

If the environmental effects operate on a spatial scale smaller than the block size, the random block model does not provide a suitable adjustment. From the coarse layout of the experimental area, it is not surprising that random block effects alone do not account for the overdispersion in the data. Adding a random component to a generalized linear model is different from adding a multiplicative overdispersion component, for example, via the PSCALE option in PROC GENMOD or a

```
random _residual_;
```

statement in PROC GLIMMIX. Such overdispersion components do not affect the parameter estimates, only their standard errors. A genuine random effect, on the other hand, affects both the parameter estimates and their standard errors (compare Output 52.1.16 to Output 52.1.7).

Output 52.1.16 Parameter Estimates for Fixed Effects in GLMM Analysis

Solutions for Fixed Effects						
Effect	entry	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		-1.4637	0.3738	3	-3.92	0.0296
entry	1	2.9609	0.5384	45	5.50	<.0001
entry	2	2.7807	0.5138	45	5.41	<.0001
entry	3	2.4339	0.4934	45	4.93	<.0001
entry	4	1.5347	0.4542	45	3.38	0.0015
entry	5	2.7653	0.5276	45	5.24	<.0001
entry	6	2.0014	0.4865	45	4.11	0.0002
entry	7	2.3518	0.4952	45	4.75	<.0001
entry	8	1.2927	0.4739	45	2.73	0.0091
entry	9	1.5663	0.4554	45	3.44	0.0013
entry	10	2.2896	0.5179	45	4.42	<.0001
entry	11	1.5018	0.4682	45	3.21	0.0025
entry	12	1.5075	0.4752	45	3.17	0.0027
entry	13	-0.5955	0.6475	45	-0.92	0.3626
entry	14	0.4573	0.5111	45	0.89	0.3758
entry	15	0.8683	0.4682	45	1.85	0.0702
entry	16	0	.	.	.	.

Output 52.1.17 Type III Test of Entry in GLMM Analysis

Type III Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
entry	15	45	6.90	<.0001

Because the block variance component is small, the Type III test for the variety effect in Output 52.1.17 is affected only very little compared to the GLM (Output 52.1.8).

Analysis with Smooth Spatial Trends

You can also consider these data in an observational sense, where the covariation of the observations is subject to modeling. Rather than deriving model components from the experimental design alone, environmental effects can be modeled by adjusting the mean and/or correlation structure. Gotway and Stroup (1997) and Schabenberger and Pierce (2002) supplant the coarse block effects with smooth-scale spatial components.

The model considered by Gotway and Stroup (1997) is a marginal model in that the correlation structure is modeled through residual-side (R-side) random components. This exponential covariance model is fit with the following statements:

```
proc glimmix data=HessianFly;
   class entry;
   model y/n = entry / solution ddfm=contain;
   random _residual_ / subject=intercept type=sp(exp)(lng lat);
run;
```

Note that the block effects have been removed from the statements. The keyword _RESIDUAL_ in the RANDOM statement instructs the GLIMMIX procedure to model the **R** matrix. Here, **R** is to be modeled as an exponential covariance structure matrix. The SUBJECT=INTERCEPT option means that all observations are considered correlated. Because the random effects are residual-type (R-side) effects, there are no columns in the **Z** matrix for this model (Output 52.1.18).

Output 52.1.18 Model Dimension Information in Marginal Spatial Analysis

The GLIMMIX Procedure

Dimensions	
R-side Cov. Parameters	2
Columns in X	17
Columns in Z per Subject	0
Subjects (Blocks in V)	1
Max Obs per Subject	64

In addition to the fixed effects, the GLIMMIX procedure now profiles one of the covariance parameters, the variance of the exponential covariance model (Output 52.1.19). This reduces the size of the optimization problem. Only a single parameter is part of the optimization, the “range” (SP(EXP)) of the spatial process.

Output 52.1.19 Optimization Information in Spatial Analysis

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	1
Lower Boundaries	1
Upper Boundaries	0
Fixed Effects	Profiled
Residual Variance	Profiled
Starting From	Data

The practical range of a spatial process is that distance at which the correlation between data points has decreased to at most 0.05. The parameter reported by the GLIMMIX procedure as SP(EXP) in Output 52.1.20 corresponds to one-third of the practical range. The practical range in this process is $3 \times 0.9052 = 2.7156$. Correlations extend beyond a single experimental unit, but they do not appear to exist on the scale of the block size.

Output 52.1.20 Estimates of Covariance Parameters

Covariance Parameter Estimates			
Cov Parm	Subject	Estimate	Standard Error
SP(EXP)	Intercept	0.9052	0.4404
Residual		2.5315	0.6974

The sill of the spatial process, the variance of the underlying residual effect, is estimated as 2.5315.

Output 52.1.21 Type III Test of Entry Effect in Spatial Analysis

Type III Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
entry	15	48	3.60	0.0004

The F value for the entry effect has been sharply reduced compared to the previous analyses. The smooth spatial variation accounts for some of the variation among the varieties (Output 52.1.21).

In this example three models were considered for the analysis of a randomized block design with binomial outcomes. If data are correlated, a standard generalized linear model often will indicate overdispersion relative to the binomial distribution. Two courses of action are considered in this example to address this overdispersion. First, the inclusion of G-side random effects models the correlation indirectly; it is induced through the sharing of random effects among responses from the same block. Second, the R-side spatial covariance structure models covariation directly. In generalized linear (mixed) models these two modeling approaches can lead to different inferences, because the models have different interpretation. The random block effects are modeled on the linked (logit) scale, and the spatial effects were modeled on the mean scale. Only in a linear mixed model are the two scales identical.

Example 52.2: Mating Experiment with Crossed Random Effects

(View the complete code for this example (gmxex02.sas) in the example repository.)

McCullagh and Nelder (1989, Ch. 14.5) describe a mating experiment—conducted by S. Arnold and P. Verell at the University of Chicago, Department of Ecology and Evolution—involving two geographically isolated populations of mountain dusky salamanders. One goal of the experiment was to determine whether barriers to interbreeding have evolved in light of the geographical isolation of the populations. In this case, matings within a population should be more successful than matings between the populations. The experiment conducted in the summer of 1986 involved 40 animals, 20 rough butt (R) and 20 whiteside (W) salamanders, with equal numbers of males and females. The animals were grouped into two sets of R males, two sets of R females, two sets of W males, and two sets of W females, so that each set comprised five salamanders. Each set was mated against one rough butt and one whiteside set, creating eight crossings. Within the pairings of

sets, each female was paired to three male animals. The salamander mating data have been used by a number of authors; see, for example, McCullagh and Nelder (1989); Schall (1991); Karim and Zeger (1992); Breslow and Clayton (1993); Wolfinger and O'Connell (1993); Shun (1997).

The following DATA step creates the data set for the analysis.

```
data salamander;
   input day fpop$ fnum mpop$ mnum mating @@;
   datalines;
4  rb  1 rb  1 1  4  rb  2 rb  5 1
4  rb  3 rb  2 1  4  rb  4 rb  4 1
4  rb  5 rb  3 1  4  rb  6 ws  9 1
4  rb  7 ws  8 0  4  rb  8 ws  6 0
4  rb  9 ws 10 0  4  rb 10 ws  7 0
4  ws  1 rb  9 0  4  ws  2 rb  7 0
4  ws  3 rb  8 0  4  ws  4 rb 10 0
4  ws  5 rb  6 0  4  ws  6 ws  5 0
4  ws  7 ws  4 1  4  ws  8 ws  1 1
4  ws  9 ws  3 1  4  ws 10 ws  2 1
8  rb  1 ws  4 1  8  rb  2 ws  5 1
8  rb  3 ws  1 0  8  rb  4 ws  2 1
8  rb  5 ws  3 1  8  rb  6 rb  9 1
8  rb  7 rb  8 0  8  rb  8 rb  6 1
8  rb  9 rb  7 0  8  rb 10 rb 10 0
8  ws  1 ws  9 1  8  ws  2 ws  6 0
8  ws  3 ws  7 0  8  ws  4 ws 10 1
8  ws  5 ws  8 1  8  ws  6 rb  2 0
8  ws  7 rb  1 1  8  ws  8 rb  4 0
8  ws  9 rb  3 1  8  ws 10 rb  5 0
12 rb  1 rb  5 1  12 rb  2 rb  3 1
12 rb  3 rb  1 1  12 rb  4 rb  2 1
12 rb  5 rb  4 1  12 rb  6 ws 10 1
12 rb  7 ws  9 0  12 rb  8 ws  7 0
12 rb  9 ws  8 1  12 rb 10 ws  6 1
12 ws  1 rb  7 1  12 ws  2 rb  9 0
12 ws  3 rb  6 0  12 ws  4 rb  8 1
12 ws  5 rb 10 0  12 ws  6 ws  3 1
12 ws  7 ws  5 1  12 ws  8 ws  2 1
12 ws  9 ws  1 1  12 ws 10 ws  4 0
16 rb  1 ws  1 0  16 rb  2 ws  3 1
16 rb  3 ws  4 1  16 rb  4 ws  5 0
16 rb  5 ws  2 1  16 rb  6 rb  7 0
16 rb  7 rb  9 1  16 rb  8 rb 10 0
16 rb  9 rb  6 1  16 rb 10 rb  8 0
16 ws  1 ws 10 1  16 ws  2 ws  7 1
16 ws  3 ws  9 0  16 ws  4 ws  8 1
16 ws  5 ws  6 0  16 ws  6 rb  4 0
16 ws  7 rb  2 0  16 ws  8 rb  5 0
16 ws  9 rb  1 1  16 ws 10 rb  3 1
20 rb  1 rb  4 1  20 rb  2 rb  1 1
20 rb  3 rb  3 1  20 rb  4 rb  5 1
20 rb  5 rb  2 1  20 rb  6 ws  6 1
20 rb  7 ws  7 0  20 rb  8 ws 10 1
20 rb  9 ws  9 1  20 rb 10 ws  8 1
```

```
20 ws  1 rb 10 0  20 ws  2 rb  6 0
20 ws  3 rb  7 0  20 ws  4 rb  9 0
20 ws  5 rb  8 0  20 ws  6 ws  2 0
20 ws  7 ws  1 1  20 ws  8 ws  5 1
20 ws  9 ws  4 1  20 ws 10 ws  3 1
24 rb  1 ws  5 1  24 rb  2 ws  2 1
24 rb  3 ws  3 1  24 rb  4 ws  4 1
24 rb  5 ws  1 1  24 rb  6 rb  8 1
24 rb  7 rb  6 0  24 rb  8 rb  9 1
24 rb  9 rb 10 1  24 rb 10 rb  7 0
24 ws  1 ws  8 1  24 ws  2 ws 10 0
24 ws  3 ws  6 1  24 ws  4 ws  9 1
24 ws  5 ws  7 0  24 ws  6 rb  1 0
24 ws  7 rb  5 1  24 ws  8 rb  3 0
24 ws  9 rb  4 0  24 ws 10 rb  2 0
;
```

The first observation, for example, indicates that rough butt female 1 was paired in the laboratory on day 4 of the experiment with rough butt male 1, and the pair mated. On the same day rough butt female 7 was paired with whiteside male 8, but the pairing did not result in mating of the animals.

The model adopted by many authors for these data comprises fixed effects for gender and population, their interaction, and male and female random effects. Specifically, let π_{RR}, π_{RW}, π_{WR}, and π_{WW} denote the mating probabilities between the populations, where the first subscript identifies the female partner of the pair. Then, you model

$$\log\left\{\frac{\pi_{kl}}{1-\pi_{kl}}\right\} = \tau_{kl} + \gamma_f + \gamma_m \quad k,l \in \{R,W\}$$

where γ_f and γ_m are independent random variables representing female and male random effects (20 each), and τ_{kl} denotes the average logit of mating between females of population k and males of population l. The following statements fit this model by pseudo-likelihood:

```
proc glimmix data=salamander;
   class fpop fnum mpop mnum;
   model mating(event='1') = fpop|mpop / dist=binary;
   random fpop*fnum mpop*mnum;
   lsmeans fpop*mpop / ilink;
run;
```

The response variable is the two-level variable mating. Because it is coded as zeros and ones, and because PROC GLIMMIX models by default the probability of the first level according to the response-level ordering, the EVENT='1' option instructs PROC GLIMMIX to model the probability of a successful mating. The distribution of the mating variable, conditional on the random effects, is binary.

The fpop*fnum effect in the RANDOM statement creates a random intercept for each female animal. Because fpop and fnum are CLASS variables, the effect has 20 levels (10 rb and 10 ws females). Similarly, the mpop*mnum effect creates the random intercepts for the male animals. Because no TYPE= is specified in the RANDOM statement, the covariance structure defaults to TYPE=VC. The random effects and their levels are independent, and each effect has its own variance component. Because the conditional distribution of the data, conditioned on the random effects, is binary, no extra scale parameter (ϕ) is added.

The LSMEANS statement requests least squares means for the four levels of the fpop*mpop effect, which are estimates of the cell means in the 2×2 classification of female and male populations. The ILINK option in

the LSMEANS statement requests that the estimated means and standard errors are also reported on the scale of the data. This yields estimates of the four mating probabilities, π_{RR}, π_{RW}, π_{WR}, and π_{WW}.

The "Model Information" table displays general information about the model being fit (Output 52.2.1).

Output 52.2.1 Analysis of Mating Experiment with Crossed Random Effects

The GLIMMIX Procedure

Model Information	
Data Set	WORK.SALAMANDER
Response Variable	mating
Response Distribution	Binary
Link Function	Logit
Variance Function	Default
Variance Matrix	Not blocked
Estimation Technique	Residual PL
Degrees of Freedom Method	Containment

The response variable mating follows a binary distribution (conditional on the random effects). Hence, the mean of the data is an event probability, π, and the logit of this probability is linearly related to the linear predictor of the model. The variance function is the default function that is implied by the distribution, $a(\pi) = \pi(1-\pi)$. The variance matrix is not blocked, because the GLIMMIX procedure does not process the data by subjects (see the section "Processing by Subjects" on page 3957 for details). The estimation technique is the default method for GLMMs, residual pseudo-likelihood (METHOD=RSPL), and degrees of freedom for tests and confidence intervals are determined by the containment method.

The "Class Level Information" table in Output 52.2.2 lists the levels of the variables listed in the CLASS statement, as well as the order of the levels.

Output 52.2.2 Class Level Information and Number of Observations

Class Level Information		
Class	**Levels**	**Values**
fpop	2	rb ws
fnum	10	1 2 3 4 5 6 7 8 9 10
mpop	2	rb ws
mnum	10	1 2 3 4 5 6 7 8 9 10

Number of Observations Read	120
Number of Observations Used	120

Note that there are two female populations and two male populations; also, the variables fnum and mnum have 10 levels each. As a consequence, the effects fpop*fnum and mpop*mnum identify the 20 females and males, respectively. The effect fpop*mpop identifies the four mating types.

The "Response Profile Table," which is displayed for binary or multinomial data, lists the levels of the response variable and their order (Output 52.2.3). With binary data, the table also provides information about which level of the response variable defines the event. Because of the EVENT='1' response variable option in the MODEL statement, the probability being modeled is that of the higher-ordered value.

Output 52.2.3 Response Profiles

Response Profile		
Ordered Value	mating	Total Frequency
1	0	50
2	1	70

The GLIMMIX procedure is modeling the probability that mating='1'.

There are two covariance parameters in this model, the variance of the fpop*fnum effect and the variance of the mpop*mnum effect (Output 52.2.4). Both parameters are modeled as G-side parameters. The nine columns in the **X** matrix comprise the intercept, two columns each for the levels of the fpop and mpop effects, and four columns for their interaction. The **Z** matrix has 40 columns, one for each animal. Because the data are not processed by subjects, PROC GLIMMIX assumes the data consist of a single subject (a single block in **V**).

Output 52.2.4 Model Dimensions Information

Dimensions	
G-side Cov. Parameters	2
Columns in X	9
Columns in Z	40
Subjects (Blocks in V)	1
Max Obs per Subject	120

The "Optimization Information" table displays basic information about the optimization (Output 52.2.5). The default technique for GLMMs is the quasi-Newton method. There are two parameters in the optimization, which correspond to the two variance components. The 17 fixed effects parameters are not part of the optimization. The initial optimization computes pseudo-data based on the response values in the data set rather than from estimates of a generalized linear model fit.

Output 52.2.5 Optimization Information

Optimization Information	
Optimization Technique	Newton-Raphson with Ridging
Parameters in Optimization	2
Lower Boundaries	2
Upper Boundaries	0
Fixed Effects	Profiled
Starting From	Data

The GLIMMIX procedure performs eight optimizations after the initial optimization (Output 52.2.6). That is, following the initial pseudo-data creation, the pseudo-data were updated eight more times and a total of nine linear mixed models were estimated.

Output 52.2.6 Iteration History and Convergence Status

Iteration History					
Iteration	Restarts	Subiterations	Objective Function	Change	Max Gradient
0	0	4	537.09173501	2.00000000	1.719E-8
1	0	3	544.12516903	0.66319780	1.14E-8
2	0	2	545.89139118	0.13539318	1.609E-6
3	0	2	546.10489538	0.01742065	5.89E-10
4	0	1	546.13075146	0.00212475	9.654E-7
5	0	1	546.13374731	0.00025072	1.346E-8
6	0	1	546.13409761	0.00002931	1.84E-10
7	0	0	546.13413861	0.00000000	4.285E-6

Convergence criterion (PCONV=1.11022E-8) satisfied.

The "Covariance Parameter Estimates" table lists the estimates for the two variance components and their estimated standard errors (Output 52.2.7). The heterogeneity (in the logit of the mating probabilities) among the females is considerably larger than the heterogeneity among the males.

Output 52.2.7 Estimated Covariance Parameters and Approximate Standard Errors

Covariance Parameter Estimates		
Cov Parm	Estimate	Standard Error
fpop*fnum	1.4099	0.8871
mpop*mnum	0.08963	0.4102

The "Type III Tests of Fixed Effects" table indicates a significant interaction between the male and female populations (Output 52.2.8). A comparison in the logits of mating success in pairs with R females and W females depends on whether the male partner in the pair is the same species. The "fpop*mpop Least Squares Means" table shows this effect more clearly (Output 52.2.9).

Output 52.2.8 Tests of Main Effects and Interaction

Type III Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
fpop	1	18	2.86	0.1081
mpop	1	17	4.71	0.0444
fpop*mpop	1	81	9.61	0.0027

Output 52.2.9 Interaction Least Squares Means

fpop*mpop Least Squares Means								
fpop	mpop	Estimate	Standard Error	DF	t Value	Pr > |t|	Mean	Standard Error Mean
rb	rb	1.1629	0.5961	81	1.95	0.0545	0.7619	0.1081
rb	ws	0.7839	0.5729	81	1.37	0.1750	0.6865	0.1233
ws	rb	-1.4119	0.6143	81	-2.30	0.0241	0.1959	0.09678
ws	ws	1.0151	0.5871	81	1.73	0.0876	0.7340	0.1146

In a pairing with a male rough butt salamander, the logit drops sharply from 1.1629 to –1.4119 when the male is paired with a whiteside female instead of a female from its own population. The corresponding estimated probabilities of mating success are $\widehat{\pi}_{RR} = 0.7619$ and $\widehat{\pi}_{WR} = 0.1959$. If the same comparisons are made in pairs with whiteside males, then you also notice a drop in the logit if the female comes from a different population, 1.0151 versus 0.7839. The change is considerably less, though, corresponding to mating probabilities of $\widehat{\pi}_{WW} = 0.7340$ and $\widehat{\pi}_{RW} = 0.6865$. Whiteside females appear to be successful with their own population. Whiteside males appear to succeed equally well with female partners of the two populations.

This insight into the factor-level comparisons can be amplified by graphing the least squares mean comparisons and by subsetting the differences of least squares means. This is accomplished with the following statements:

```
ods graphics on;
ods select DiffPlot SliceDiffs;
proc glimmix data=salamander;
   class fpop fnum mpop mnum;
   model mating(event='1') = fpop|mpop / dist=binary;
   random fpop*fnum mpop*mnum;
   lsmeans fpop*mpop / plots=diffplot;
   lsmeans fpop*mpop / slicediff=(mpop fpop);
run;
ods graphics off;
```

The PLOTS=DIFFPLOT option in the first LSMEANS statement requests a comparison plot that displays the result of all pairwise comparisons (Output 52.2.10). The SLICEDIFF=(mpop fpop) option requests differences of simple effects.

The comparison plot in Output 52.2.10 is also known as a mean-mean scatter plot (Hsu 1996). Each solid line in the plot corresponds to one of the possible $4 \times 3/2 = 6$ unique pairwise comparisons. The line is centered at the intersection of two least squares means, and the length of the line segments corresponds to the width of a 95% confidence interval for the difference between the two least squares means. The length of the segment is adjusted for the rotation. If a line segment crosses the dashed 45-degree line, the comparison between the two factor levels is not significant; otherwise, it is significant. The horizontal and vertical axes of the plot are drawn in least squares means units, and the grid lines are placed at the values of the least squares means.

The six pairs of least squares means comparisons separate into two sets of three pairs. Comparisons in the first set are significant; comparisons in the second set are not significant. For the significant set, the female partner in one of the pairs is a whiteside salamander. For the nonsignificant comparisons, the male partner in one of the pairs is a whiteside salamander.

Output 52.2.10 LS-Means Diffogram

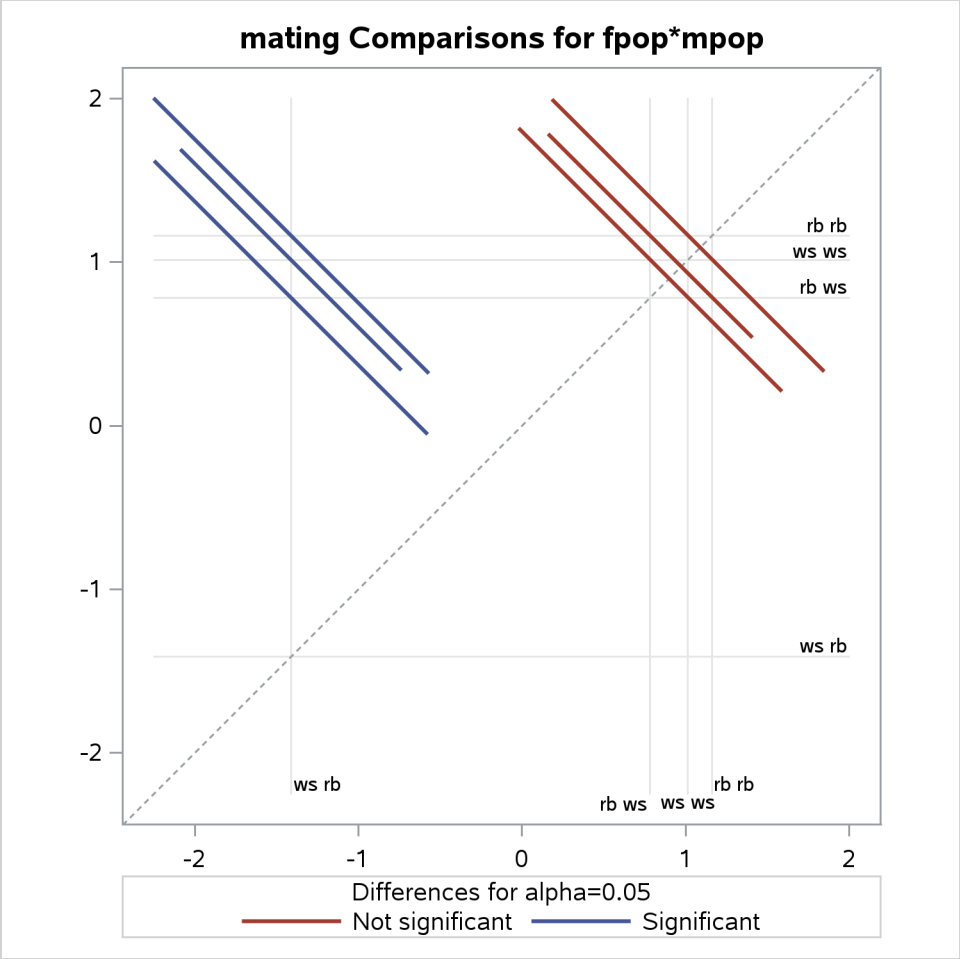

The "Simple Effect Comparisons" tables show the results of the SLICEDIFF= option in the second LSMEANS statement (Output 52.2.11).

Output 52.2.11 Simple Effect Comparisons

Simple Effect Comparisons of fpop*mpop Least Squares Means By mpop

Simple Effect Level	fpop	_fpop	Estimate	Standard Error	DF	t Value	Pr > |t|
mpop rb	rb	ws	2.5748	0.8458	81	3.04	0.0031
mpop ws	rb	ws	-0.2312	0.8092	81	-0.29	0.7758

Simple Effect Comparisons of fpop*mpop Least Squares Means By fpop

Simple Effect Level	mpop	_mpop	Estimate	Standard Error	DF	t Value	Pr > |t|
fpop rb	rb	ws	0.3790	0.6268	81	0.60	0.5471
fpop ws	rb	ws	-2.4270	0.6793	81	-3.57	0.0006

The first table of simple effect comparisons holds fixed the level of the mpop factor and compares the levels

of the fpop factor. Because there is only one possible comparison for each male population, there are two entries in the table. The first entry compares the logits of mating probabilities when the male partner is a rough butt, and the second entry applies when the male partner is from the whiteside population. The second table of simple effects comparisons applies the same logic, but holds fixed the level of the female partner in the pair. Note that these four comparisons are a subset of all six possible comparisons, eliminating those where both factors are varied at the same time. The simple effect comparisons show that there is no difference in mating probabilities if the male partner is a whiteside salamander, or if the female partner is a rough butt. Rough butt females also appear to mate indiscriminately.

Example 52.3: Smoothing Disease Rates; Standardized Mortality Ratios

(View the complete code for this example (gmxex03.sas) in the example repository.)

Clayton and Kaldor (1987, Table 1) present data on observed and expected cases of lip cancer in the 56 counties of Scotland between 1975 and 1980. The expected number of cases was determined by a separate multiplicative model that accounted for the age distribution in the counties. The goal of the analysis is to estimate the county-specific log-relative risks, also known as standardized mortality ratios (SMR).

If Y_i is the number of incident cases in county i and E_i is the expected number of incident cases, then the ratio of observed to expected counts, Y_i/E_i, is the standardized mortality ratio. Clayton and Kaldor (1987) assume there exists a relative risk λ_i that is specific to each county and is a random variable. Conditional on λ_i, the observed counts are independent Poisson variables with mean $E_i\lambda_i$.

An elementary mixed model for λ_i specifies only a random intercept for each county, in addition to a fixed intercept. Breslow and Clayton (1993), in their analysis of these data, also provide a covariate that measures the percentage of employees in agriculture, fishing, and forestry. The expanded model for the region-specific relative risk in Breslow and Clayton (1993) is

$$\lambda_i = \exp\{\beta_0 + \beta_1 x_i/10 + \gamma_i\}, \quad i = 1, \ldots, 56$$

where β_0 and β_1 are fixed effects, and the γ_i are county random effects.

The following DATA step creates the data set lipcancer. The expected number of cases is based on the observed standardized mortality ratio for counties with lip cancer cases, and based on the expected counts reported by Clayton and Kaldor (1987, Table 1) for the counties without cases. The sum of the expected counts then equals the sum of the observed counts.

```
data lipcancer;
   input county observed expected employment SMR;
   if (observed > 0) then expCount = 100*observed/SMR;
   else expCount = expected;
   datalines;
 1  9  1.4 16 652.2
 2 39  8.7 16 450.3
 3 11  3.0 10 361.8
 4  9  2.5 24 355.7
 5 15  4.3 10 352.1
 6  8  2.4 24 333.3
 7 26  8.1 10 320.6
 8  7  2.3  7 304.3
 9  6  2.0  7 303.0
```

```
10 20  6.6 16 301.7
11 13  4.4  7 295.5
12  5  1.8 16 279.3
13  3  1.1 10 277.8
14  8  3.3 24 241.7
15 17  7.8  7 216.8
16  9  4.6 16 197.8
17  2  1.1 10 186.9
18  7  4.2  7 167.5
19  9  5.5  7 162.7
20  7  4.4 10 157.7
21 16 10.5  7 153.0
22 31 22.7 16 136.7
23 11  8.8 10 125.4
24  7  5.6  7 124.6
25 19 15.5  1 122.8
26 15 12.5  1 120.1
27  7  6.0  7 115.9
28 10  9.0  7 111.6
29 16 14.4 10 111.3
30 11 10.2 10 107.8
31  5  4.8  7 105.3
32  3  2.9 24 104.2
33  7  7.0 10  99.6
34  8  8.5  7  93.8
35 11 12.3  7  89.3
36  9 10.1  0  89.1
37 11 12.7 10  86.8
38  8  9.4  1  85.6
39  6  7.2 16  83.3
40  4  5.3  0  75.9
41 10 18.8  1  53.3
42  8 15.8 16  50.7
43  2  4.3 16  46.3
44  6 14.6  0  41.0
45 19 50.7  1  37.5
46  3  8.2  7  36.6
47  2  5.6  1  35.8
48  3  9.3  1  32.1
49 28 88.7  0  31.6
50  6 19.6  1  30.6
51  1  3.4  1  29.1
52  1  3.6  0  27.6
53  1  5.7  1  17.4
54  1  7.0  1  14.2
55  0  4.2 16   0.0
56  0  1.8 10   0.0
;
```

Because the mean of the Poisson variates, conditional on the random effects, is $\mu_i = E_i\lambda_i$, applying a log link yields

$$\log\{\mu_i\} = \log\{E_i\} + \beta_0 + \beta_1 x_i/10 + \gamma_i$$

The term $\log\{E_i\}$ is an offset, a regressor variable whose coefficient is known to be one. Note that it is

assumed that the E_i are known; they are not treated as random variables.

The following statements fit this model by residual pseudo-likelihood:

```
proc glimmix data=lipcancer;
   class county;
   x    = employment / 10;
   logn = log(expCount);
   model observed = x / dist=poisson offset=logn
                        solution ddfm=none;
   random county;
   SMR_pred = 100*exp(_zgamma_ + _xbeta_);
   id employment SMR SMR_pred;
   output out=glimmixout;
run;
```

The offset is created with the assignment statement

```
logn = log(expCount);
```

and is associated with the linear predictor through the OFFSET= option in the MODEL statement. The statement

```
x = employment / 10;
```

transforms the covariate measuring percentage of employment in agriculture, fisheries, and forestry to agree with the analysis of Breslow and Clayton (1993). The DDFM=NONE option in the MODEL statement requests chi-square tests and z tests instead of the default F tests and t tests by setting the denominator degrees of freedom in tests of fixed effects to ∞.

The statement

```
SMR_pred = 100*exp(_zgamma_ + _xbeta_);
```

calculates the fitted standardized mortality rate. Note that the offset variable does not contribute to the exponentiated term.

The OUTPUT statement saves results of the calculations to the output data set glimmixout. The ID statement specifies that only the listed variables are written to the output data set.

Output 52.3.1 Model Information in Poisson GLMM

The GLIMMIX Procedure

Model Information	
Data Set	WORK.LIPCANCER
Response Variable	observed
Response Distribution	Poisson
Link Function	Log
Variance Function	Default
Offset Variable	logn = log(expCount);
Variance Matrix	Not blocked
Estimation Technique	Residual PL
Degrees of Freedom Method	None

Output 52.3.1 *continued*

Class Level Information		
Class	**Levels**	**Values**
county	56	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56

Number of Observations Read	56
Number of Observations Used	56

Dimensions	
G-side Cov. Parameters	1
Columns in X	2
Columns in Z	56
Subjects (Blocks in V)	1
Max Obs per Subject	56

The GLIMMIX procedure displays in the "Model Information" table that the offset variable was computed with programming statements and the final assignment statement from your GLIMMIX statements (Output 52.3.1). There are two columns in the **X** matrix, corresponding to the intercept and the regressor $x/10$. There are 56 columns in the **Z** matrix, however, one for each observation in the data set (Output 52.3.1).

The optimization involves only a single covariance parameter, the variance of the county effect (Output 52.3.2). Because this parameter is a variance, the GLIMMIX procedure imposes a lower boundary constraint; the solution for the variance is bounded by zero from below.

Output 52.3.2 Optimization Information in Poisson GLMM

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	1
Lower Boundaries	1
Upper Boundaries	0
Fixed Effects	Profiled
Starting From	Data

Following the initial creation of pseudo-data and the fit of a linear mixed model, the procedure goes through five more updates of the pseudo-data, each associated with a separate optimization (Output 52.3.3). Although the objective function in each optimization is the negative of twice the restricted maximum likelihood for that pseudo-data, there is no guarantee that across the outer iterations the objective function decreases in subsequent optimizations. In this example, minus twice the residual maximum likelihood at convergence takes on its smallest value at the initial optimization and increases in subsequent optimizations.

Output 52.3.3 Iteration History in Poisson GLMM

Iteration History					
Iteration	Restarts	Subiterations	Objective Function	Change	Max Gradient
0	0	4	123.64113992	0.20997891	3.848E-8
1	0	3	127.05866018	0.03393332	0.000048
2	0	2	127.48839749	0.00223427	5.753E-6
3	0	1	127.50502469	0.00006946	1.938E-7
4	0	1	127.50528068	0.00000118	6.089E-8
5	0	0	127.50528481	0.00000000	1.346E-6

Convergence criterion (PCONV=1.11022E-8) satisfied.

The "Covariance Parameter Estimates" table in Output 52.3.4 shows the estimate of the variance of the region-specific log-relative risks. There is significant county-to-county heterogeneity in risks. If the covariate were removed from the analysis, as in Clayton and Kaldor (1987), the heterogeneity in county-specific risks would increase. (The fitted SMRs in Table 6 of Breslow and Clayton (1993) were obtained without the covariate x in the model.)

Output 52.3.4 Estimated Covariance Parameters in Poisson GLMM

Covariance Parameter Estimates		
Cov Parm	Estimate	Standard Error
county	0.3567	0.09869

The "Solutions for Fixed Effects" table displays the estimates of β_0 and β_1 along with their standard errors and test statistics (Output 52.3.5). Because of the DDFM=NONE option in the MODEL statement, PROC GLIMMIX assumes that the degrees of freedom for the t tests of $H_0\colon \beta_j = 0$ are infinite. The p-values correspond to probabilities under a standard normal distribution. The covariate measuring employment percentages in agriculture, fisheries, and forestry is significant. This covariate might be a surrogate for the exposure to sunlight, an important risk factor for lip cancer.

Output 52.3.5 Fixed-Effects Parameter Estimates in Poisson GLMM

Solutions for Fixed Effects					
Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	-0.4406	0.1572	Infty	-2.80	0.0051
x	0.6799	0.1409	Infty	4.82	<.0001

You can examine the quality of the fit of this model with various residual plots. A panel of studentized residuals is requested with the following statements:

```
ods graphics on;
ods select StudentPanel;

proc glimmix data=lipcancer plots=studentpanel;
   class county;
```

```
    x    = employment / 10;
    logn = log(expCount);
    model observed = x / dist=poisson offset=logn s ddfm=none;
    random county;
run;

ods graphics off;
```

The graph in the upper-left corner of the panel displays studentized residuals plotted against the linear predictor (Output 52.3.6). The default of the GLIMMIX procedure is to use the estimated BLUPs in the construction of the residuals and to present them on the linear scale, which in this case is the logarithmic scale. You can change the type of the computed residual with the TYPE= suboptions of each paneled display. For example, the option PLOTS=STUDENTPANEL(TYPE=NOBLUP) would request a paneled display of the marginal residuals on the linear scale.

Output 52.3.6 Panel of Studentized Residuals

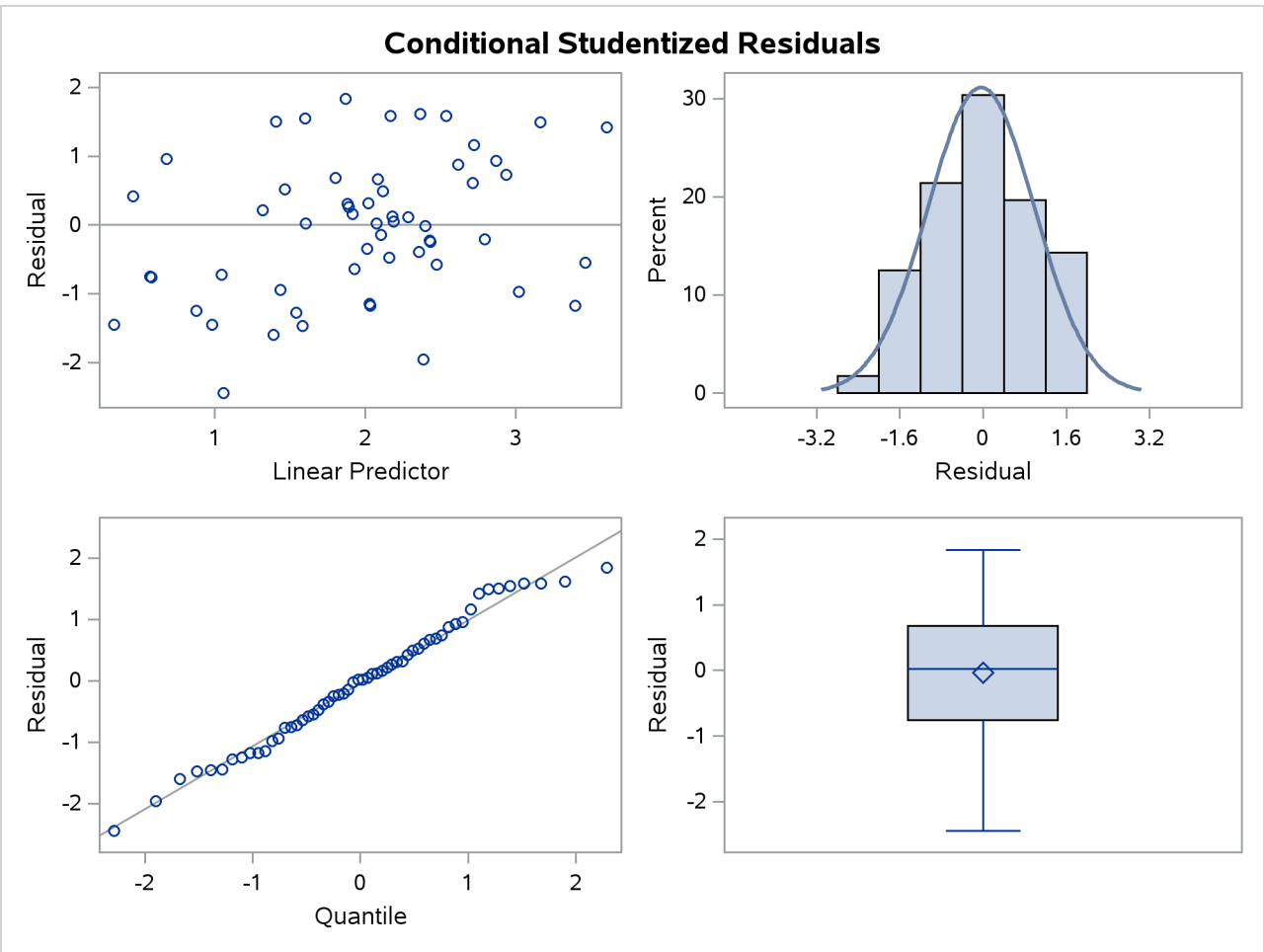

The graph in the upper-right corner of the panel shows a histogram with overlaid normal density. A Q-Q plot and a box plot are shown in the lower cells of the panel.

The following statements produce a graph of the observed and predicted standardized mortality ratios (Output 52.3.7):

```
proc template;
   define statgraph scatter;
      BeginGraph;
         layout overlayequated / yaxisopts=(label='Predicted SMR')
                                 xaxisopts=(label='Observed SMR')
                                 equatetype=square;
            lineparm y=0 slope=1 x=0 /
                   lineattrs = GraphFit(pattern=dash)
                   extend    = true;
            scatterplot y=SMR_pred x=SMR /
                    markercharacter = employment;
         endlayout;
      EndGraph;
   end;
run;
proc sgrender data=glimmixout template=scatter;
run;
```

In Output 52.3.7, fitted SMRs tend to be larger than the observed SMRs for counties with small observed SMR and smaller than the observed SMRs for counties with high observed SMR.

Output 52.3.7 Observed and Predicted SMRs; Data Labels Indicate Covariate Values

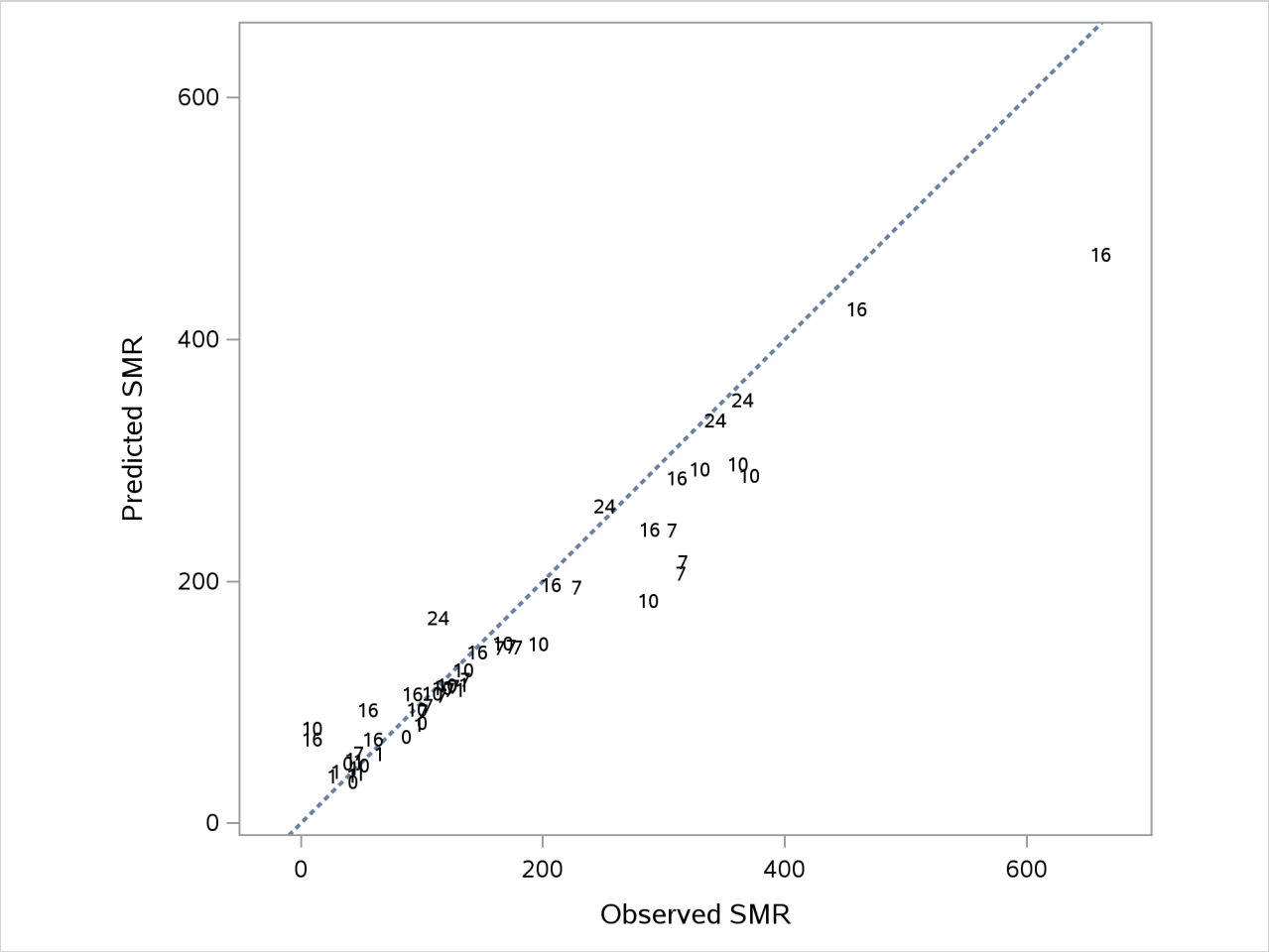

To demonstrate the impact of the random effects adjustment to the log-relative risks, the following statements fit a Poisson regression model (a GLM) by maximum likelihood:

```
proc glimmix data=lipcancer;
   x    = employment / 10;
   logn = log(expCount);
   model observed = x / dist=poisson offset=logn
                        solution ddfm=none;
   SMR_pred = 100*exp(_zgamma_ + _xbeta_);
   id employment SMR SMR_pred;
   output out=glimmixout;
run;
```

The GLIMMIX procedure defaults to maximum likelihood estimation because these statements fit a generalized linear model with nonnormal distribution. As a consequence, the SMRs are county specific only to the extent that the risks vary with the value of the covariate. But risks are no longer adjusted based on county-to-county heterogeneity in the observed incidence count.

Because of the absence of random effects, the GLIMMIX procedure recognizes the model as a generalized linear model and fits it by maximum likelihood (Output 52.3.8). The variance matrix is diagonal because the observations are uncorrelated.

Output 52.3.8 Model Information in Poisson GLM

The GLIMMIX Procedure

Model Information	
Data Set	WORK.LIPCANCER
Response Variable	observed
Response Distribution	Poisson
Link Function	Log
Variance Function	Default
Offset Variable	logn = log(expCount);
Variance Matrix	Diagonal
Estimation Technique	Maximum Likelihood
Degrees of Freedom Method	None

The "Dimensions" table shows that there are no G-side random effects in this model and no R-side scale parameter either (Output 52.3.9).

Output 52.3.9 Model Dimensions Information in Poisson GLM

Dimensions	
Columns in X	2
Columns in Z	0
Subjects (Blocks in V)	1
Max Obs per Subject	56

Because this is a GLM, the GLIMMIX procedure defaults to the Newton-Raphson algorithm, and the fixed effects (intercept and slope) comprise the parameters in the optimization (Output 52.3.10). (The default optimization technique for a GLM is the Newton-Raphson method.)

Output 52.3.10 Optimization Information in Poisson GLM

Optimization Information	
Optimization Technique	Newton-Raphson
Parameters in Optimization	2
Lower Boundaries	0
Upper Boundaries	0
Fixed Effects	Not Profiled

The estimates of β_0 and β_1 have changed from the previous analysis. In the GLMM, the estimates were $\widehat{\beta}_0 = -0.4406$ and $\widehat{\beta}_1 = 0.6799$ (Output 52.3.11).

Output 52.3.11 Parameter Estimates in Poisson GLM

Parameter Estimates					
Effect	**Estimate**	**Standard Error**	**DF**	**t Value**	**Pr > |t|**
Intercept	-0.5419	0.06951	Infty	-7.80	<.0001
x	0.7374	0.05954	Infty	12.38	<.0001

More importantly, without the county-specific adjustments through the best linear unbiased predictors of the random effects, the predicted SMRs are the same for all counties with the same percentage of employees in agriculture, fisheries, and forestry (Output 52.3.12).

Output 52.3.12 Observed and Predicted SMRs in Poisson GLM

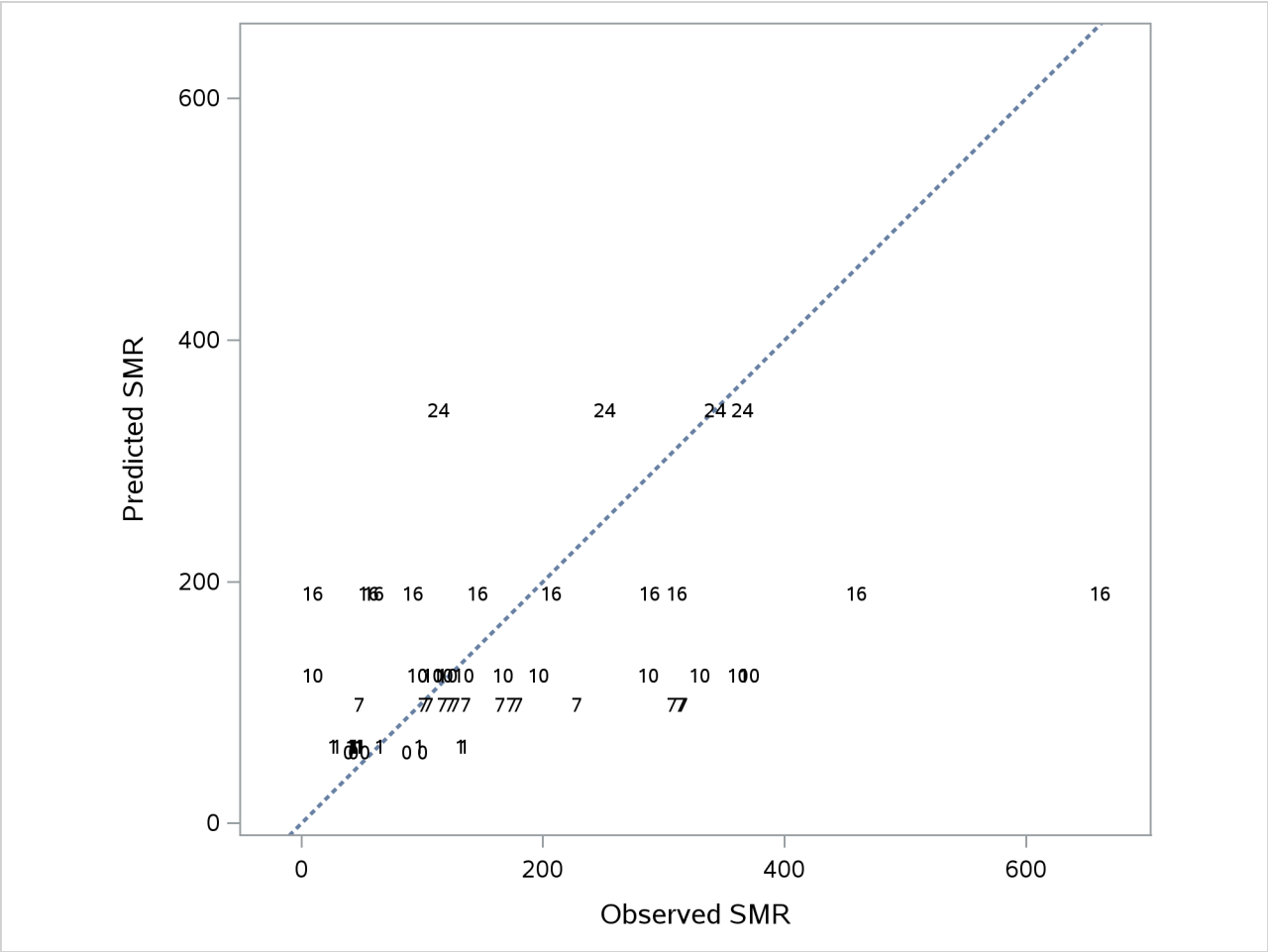

Example 52.4: Quasi-likelihood Estimation for Proportions with Unknown Distribution

(View the complete code for this example (gmxex04.sas) in the example repository.)

Wedderburn (1974) analyzes data on the incidence of leaf blotch (*Rhynchosporium secalis*) on barley.

The data represent the percentage of leaf area affected in a two-way layout with 10 barley varieties at nine sites. The following DATA step converts these data to proportions, as analyzed in McCullagh and Nelder (1989, Ch. 9.2.4). The purpose of the analysis is to make comparisons among the varieties, adjusted for site effects.

```
data blotch;
   array p{9} pct1-pct9;
   input variety pct1-pct9;
   do site = 1 to 9;
      prop = p{site}/100;
      output;
```

```
   end;
   drop pct1-pct9;
   datalines;
1  0.05  0.00  1.25  2.50  5.50  1.00  5.00  5.00 17.50
2  0.00  0.05  1.25  0.50  1.00  5.00  0.10 10.00 25.00
3  0.00  0.05  2.50  0.01  6.00  5.00  5.00  5.00 42.50
4  0.10  0.30 16.60  3.00  1.10  5.00  5.00  5.00 50.00
5  0.25  0.75  2.50  2.50  2.50  5.00 50.00 25.00 37.50
6  0.05  0.30  2.50  0.01  8.00  5.00 10.00 75.00 95.00
7  0.50  3.00  0.00 25.00 16.50 10.00 50.00 50.00 62.50
8  1.30  7.50 20.00 55.00 29.50  5.00 25.00 75.00 95.00
9  1.50  1.00 37.50  5.00 20.00 50.00 50.00 75.00 95.00
10 1.50 12.70 26.25 40.00 43.50 75.00 75.00 75.00 95.00
;
```

Little is known about the distribution of the leaf area proportions. The outcomes are not binomial proportions, because they do not represent the ratio of a count over a total number of Bernoulli trials. However, because the mean proportion μ_{ij} for variety j on site i must lie in the interval [0, 1], you can commence the analysis with a model that treats Prop as a "pseudo-binomial" variable:

$$
\begin{aligned}
\mathrm{E}[\mathsf{Prop}_{ij}] &= \mu_{ij} \\
\mu_{ij} &= 1/(1+\exp\{-\eta_{ij}\}) \\
\eta_{ij} &= \beta_0 + \alpha_i + \tau_j \\
\mathrm{Var}[\mathsf{Prop}_{ij}] &= \phi\mu_{ij}(1-\mu_{ij})
\end{aligned}
$$

Here, η_{ij} is the linear predictor for variety j on site i, α_i denotes the ith site effect, and τ_j denotes the jth barley variety effect. The logit of the expected leaf area proportions is linearly related to these effects. The variance function of the model is that of a binomial(n,μ_{ij}) variable, and ϕ is an overdispersion parameter. The moniker "pseudo-binomial" derives not from the pseudo-likelihood methods used to estimate the parameters in the model, but from treating the response variable as if it had first and second moment properties akin to a binomial random variable.

The model is fit in the GLIMMIX procedure with the following statements:

```
proc glimmix data=blotch;
   class site variety;
   model prop = site variety / link=logit dist=binomial;
   random _residual_;
   lsmeans variety / diff=control('1');
run;
```

The MODEL statement specifies the distribution as binomial and the logit link. Because the variance function of the binomial distribution is $a(\mu) = \mu(1-\mu)$, you use the statement

```
random _residual_;
```

to specify the scale parameter ϕ. The LSMEANS statement requests estimates of the least squares means for the barley variety. The DIFF=CONTROL('1') option requests tests of least squares means differences against the first variety.

The "Model Information" table in Output 52.4.1 describes the model and methods used in fitting the statistical model. It is assumed here that the data are binomial proportions.

Output 52.4.1 Model Information in Pseudo-binomial Analysis

The GLIMMIX Procedure

Model Information	
Data Set	WORK.BLOTCH
Response Variable	prop
Response Distribution	Binomial
Link Function	Logit
Variance Function	Default
Variance Matrix	Diagonal
Estimation Technique	Maximum Likelihood
Degrees of Freedom Method	Residual

The "Class Level Information" table in Output 52.4.2 lists the number of levels of the Site and Variety effects and their values. All 90 observations read from the data are used in the analysis.

Output 52.4.2 Class Levels and Number of Observations

Class Level Information		
Class	**Levels**	**Values**
site	9	1 2 3 4 5 6 7 8 9
variety	10	1 2 3 4 5 6 7 8 9 10

Number of Observations Read	90
Number of Observations Used	90

In Output 52.4.3, the "Dimensions" table shows that the model does not contain G-side random effects. There is a single covariance parameter, which corresponds to ϕ. The "Optimization Information" table shows that the optimization comprises 18 parameters (Output 52.4.3). These correspond to the 18 nonsingular columns of the $\mathbf{X}'\mathbf{X}$ matrix.

Output 52.4.3 Model Fit in Pseudo-binomial Analysis

Dimensions	
Covariance Parameters	1
Columns in X	20
Columns in Z	0
Subjects (Blocks in V)	1
Max Obs per Subject	90

Optimization Information	
Optimization Technique	Newton-Raphson
Parameters in Optimization	18
Lower Boundaries	0
Upper Boundaries	0
Fixed Effects	Not Profiled

Output 52.4.3 *continued*

Fit Statistics	
-2 Log Likelihood	57.15
AIC (smaller is better)	93.15
AICC (smaller is better)	102.79
BIC (smaller is better)	138.15
CAIC (smaller is better)	156.15
HQIC (smaller is better)	111.30
Pearson Chi-Square	6.39
Pearson Chi-Square / DF	0.09

There are significant site and variety effects in this model based on the approximate Type III F tests (Output 52.4.4).

Output 52.4.4 Tests of Site and Variety Effects in Pseudo-binomial Analysis

Type III Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
site	8	72	18.25	<.0001
variety	9	72	13.85	<.0001

Output 52.4.5 displays the Variety least squares means for this analysis. These are obtained by averaging

$$\text{logit}(\widehat{\mu}_{ij}) = \widehat{\eta}_{ij}$$

across the sites. In other words, LS-means are computed on the linked scale where the model effects are additive. Note that the least squares means are ordered by variety. The estimate of the expected proportion of infected leaf area for the first variety is

$$\widehat{\mu}_{.,1} = \frac{1}{1 + \exp\{4.38\}} = 0.0124$$

and that for the last variety is

$$\widehat{\mu}_{.,10} = \frac{1}{1 + \exp\{0.127\}} = 0.468$$

Output 52.4.5 Variety Least Squares Means in Pseudo-binomial Analysis

variety Least Squares Means					
variety	Estimate	Standard Error	DF	t Value	Pr > |t|
1	-4.3800	0.5643	72	-7.76	<.0001
2	-4.2300	0.5383	72	-7.86	<.0001
3	-3.6906	0.4623	72	-7.98	<.0001
4	-3.3319	0.4239	72	-7.86	<.0001
5	-2.7653	0.3768	72	-7.34	<.0001
6	-2.0089	0.3320	72	-6.05	<.0001
7	-1.8095	0.3228	72	-5.61	<.0001
8	-1.0380	0.2960	72	-3.51	0.0008
9	-0.8800	0.2921	72	-3.01	0.0036
10	-0.1270	0.2808	72	-0.45	0.6523

Because of the ordering of the least squares means, the differences against the first variety are also ordered from smallest to largest (Output 52.4.6).

Output 52.4.6 Variety Differences against the First Variety

Differences of variety Least Squares Means						
variety	_variety	Estimate	Standard Error	DF	t Value	Pr > |t|
2	1	0.1501	0.7237	72	0.21	0.8363
3	1	0.6895	0.6724	72	1.03	0.3086
4	1	1.0482	0.6494	72	1.61	0.1109
5	1	1.6147	0.6257	72	2.58	0.0119
6	1	2.3712	0.6090	72	3.89	0.0002
7	1	2.5705	0.6065	72	4.24	<.0001
8	1	3.3420	0.6015	72	5.56	<.0001
9	1	3.5000	0.6013	72	5.82	<.0001
10	1	4.2530	0.6042	72	7.04	<.0001

This analysis depends on your choice for the variance function that was implied by the binomial distribution. You can diagnose the distributional assumption by examining various graphical diagnostics measures. The following statements request a panel display of the Pearson-type residuals:

```
ods graphics on;
ods select PearsonPanel;
proc glimmix data=blotch plots=pearsonpanel;
   class site variety;
   model prop = site variety / link=logit dist=binomial;
   random _residual_;
run;
ods graphics off;
```

Output 52.4.7 clearly indicates that the chosen variance function is not appropriate for these data. As μ approaches zero or one, the variability in the residuals is less than that implied by the binomial variance function.

Output 52.4.7 Panel of Pearson-Type Residuals in Pseudo-binomial Analysis

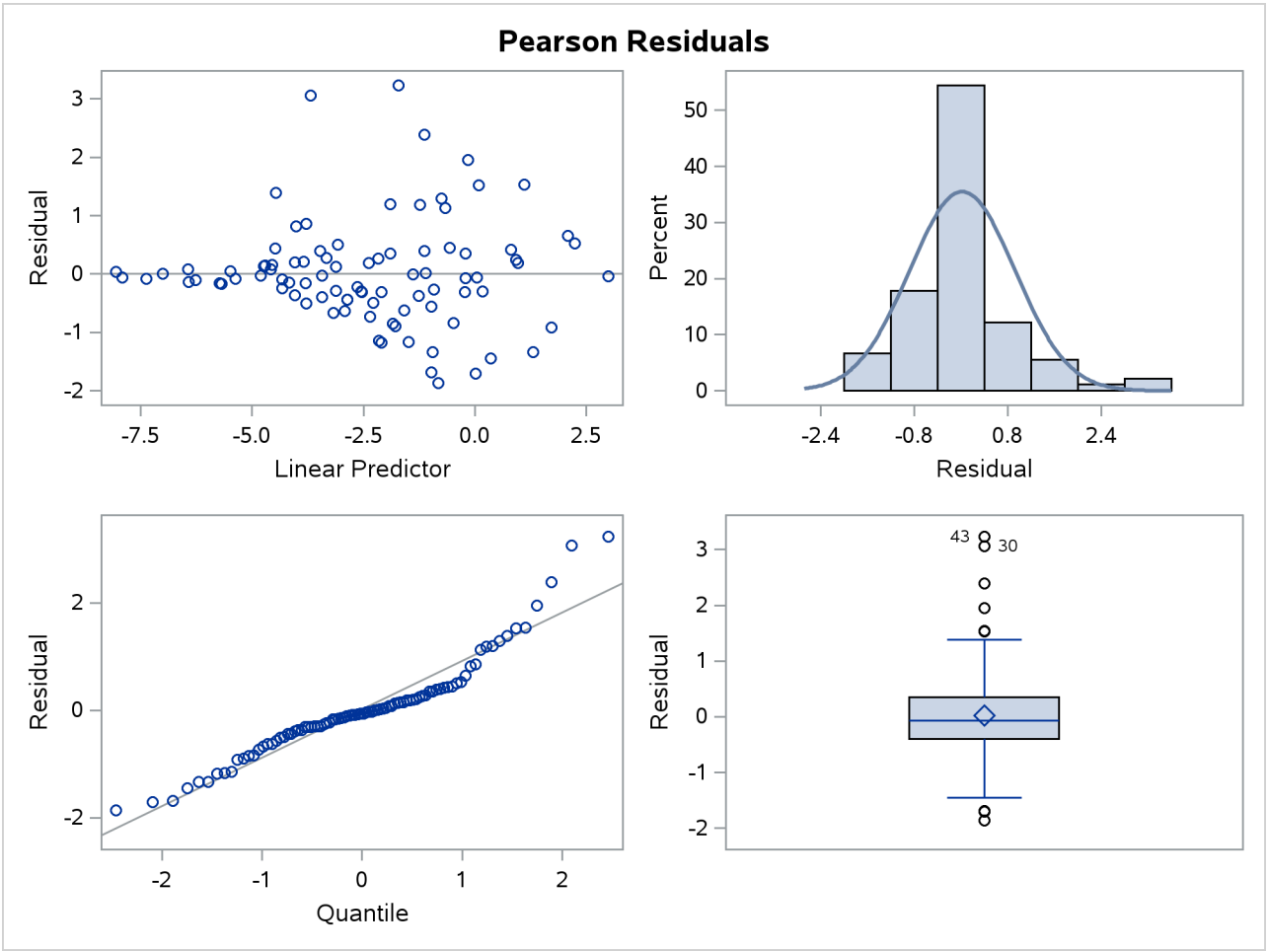

To remedy this situation, McCullagh and Nelder (1989) consider instead the variance function

$$\mathrm{Var}[\mathsf{Prop}_{ij}] = \mu_{ij}^2(1-\mu_{ij})^2$$

Imagine two varieties with $\mu_{.i} = 0.1$ and $\mu_{.k} = 0.5$. Under the binomial variance function, the variance of the proportion for variety k is 2.77 times larger than that for variety i. Under the revised model this ratio increases to $2.77^2 = 7.67$.

The analysis of the revised model is obtained with the next set of GLIMMIX statements. Because you need to model a variance function that does not correspond to any of the built-in distributions, you need to supply a function with an assignment to the automatic variable _VARIANCE_. The GLIMMIX procedure then considers the distribution of the data as unknown. The corresponding estimation technique is quasi-likelihood. Because this model does not include an extra scale parameter, you can drop the RANDOM _RESIDUAL_ statement from the analysis.

```
ods graphics on;
ods select ModelInfo FitStatistics LSMeans Diffs PearsonPanel;
proc glimmix data=blotch plots=pearsonpanel;
   class site variety;
```

```
    _variance_ = _mu_**2 * (1-_mu_)**2;
    model prop = site variety / link=logit;
    lsmeans variety / diff=control('1');
  run;
  ods graphics off;
```

The "Model Information" table in Output 52.4.8 now displays the distribution as "Unknown," because of the assignment made in the GLIMMIX statements to _VARIANCE_. The table also shows the expression evaluated as the variance function.

Output 52.4.8 Model Information in Quasi-likelihood Analysis

The GLIMMIX Procedure

Model Information	
Data Set	WORK.BLOTCH
Response Variable	prop
Response Distribution	Unknown
Link Function	Logit
Variance Function	_mu_**2 * (1-_mu_)**2
Variance Matrix	Diagonal
Estimation Technique	Quasi-Likelihood
Degrees of Freedom Method	Residual

The fit statistics of the model are now expressed in terms of the log quasi-likelihood. It is computed as

$$\sum_{i=1}^{9}\sum_{j=1}^{10}\int_{y_{ij}}^{\mu_{ij}} \frac{y_{ij}-t}{t^2(1-t)^2}\,dt$$

Twice the negative of this sum equals –85.74, which is displayed in the "Fit Statistics" table (Output 52.4.9).

The scaled Pearson statistic is now 0.99. Inclusion of an extra scale parameter ϕ would have little or no effect on the results.

Output 52.4.9 Fit Statistics in Quasi-likelihood Analysis

Fit Statistics	
-2 Log Quasi-Likelihood	-85.74
Quasi-AIC (smaller is better)	-49.74
Quasi-AICC (smaller is better)	-40.11
Quasi-BIC (smaller is better)	-4.75
Quasi-CAIC (smaller is better)	13.25
Quasi-HQIC (smaller is better)	-31.60
Pearson Chi-Square	71.17
Pearson Chi-Square / DF	0.99

The panel of Pearson-type residuals now shows a much more adequate distribution for the residuals and a reduction in the number of outlying residuals (Output 52.4.10).

Output 52.4.10 Panel of Pearson-Type Residuals (Quasi-likelihood)

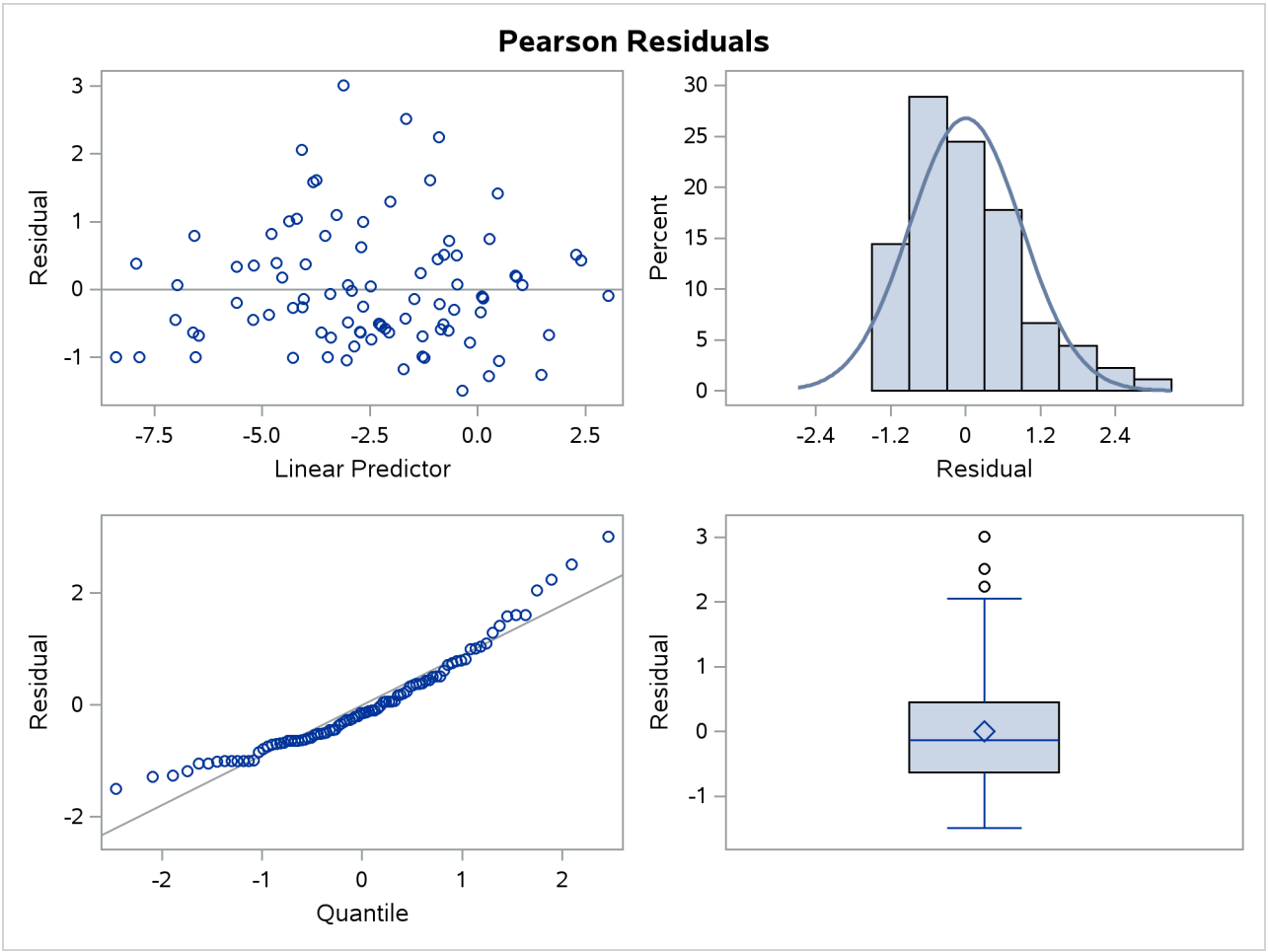

The least squares means are no longer ordered in size by variety (Output 52.4.11). For example, $\text{logit}(\widehat{\mu}_{.1}) > \text{logit}(\widehat{\mu}_{.2})$. Under the revised model, the second variety has a greater percentage of its leaf area covered by blotch, compared to the first variety. Varieties 5 and 6 and varieties 8 and 9 show similar reversal in ranking.

Output 52.4.11 Variety Least Squares Means in Quasi-likelihood Analysis

variety Least Squares Means

variety	Estimate	Standard Error	DF	t Value	Pr > |t|
1	-4.0453	0.3333	72	-12.14	<.0001
2	-4.5126	0.3333	72	-13.54	<.0001
3	-3.9664	0.3333	72	-11.90	<.0001
4	-3.0912	0.3333	72	-9.27	<.0001
5	-2.6927	0.3333	72	-8.08	<.0001
6	-2.7167	0.3333	72	-8.15	<.0001
7	-1.7052	0.3333	72	-5.12	<.0001
8	-0.7827	0.3333	72	-2.35	0.0216
9	-0.9098	0.3333	72	-2.73	0.0080
10	-0.1580	0.3333	72	-0.47	0.6369

Interestingly, the standard errors are constant among the LS-means (Output 52.4.11) and among the LS-means differences (Output 52.4.12). This is due to the fact that for the logit link

$$\frac{\partial \mu}{\partial \eta} = \mu(1-\mu)$$

which cancels with the square root of the variance function in the estimating equations. The analysis is thus orthogonal.

Output 52.4.12 Variety Differences in Quasi-likelihood Analysis

Differences of variety Least Squares Means						
variety	_variety	Estimate	Standard Error	DF	t Value	Pr > |t|
2	1	-0.4673	0.4714	72	-0.99	0.3249
3	1	0.07885	0.4714	72	0.17	0.8676
4	1	0.9541	0.4714	72	2.02	0.0467
5	1	1.3526	0.4714	72	2.87	0.0054
6	1	1.3286	0.4714	72	2.82	0.0062
7	1	2.3401	0.4714	72	4.96	<.0001
8	1	3.2626	0.4714	72	6.92	<.0001
9	1	3.1355	0.4714	72	6.65	<.0001
10	1	3.8873	0.4714	72	8.25	<.0001

Example 52.5: Joint Modeling of Binary and Count Data

(View the complete code for this example (gmxex05.sas) in the example repository.)

Clustered data arise when multiple observations are collected on the same sampling or experimental unit. Often, these multiple observations refer to the same attribute measured at different points in time or space. This leads to repeated measures, longitudinal, and spatial data, which are special forms of multivariate data. A different class of multivariate data arises when the multiple observations refer to different attributes.

The data set hernio, created in the following DATA step, provides an example of a bivariate outcome variable. It reflects the condition and length of hospital stay for 32 herniorrhaphy patients. These data are based on data given by Mosteller and Tukey (1977) and reproduced in Hand et al. (1994, pp. 390, 391). The data set that follows does not contain all the covariates given in these sources. The response variables are leave and los; these denote the condition of the patient upon leaving the operating room and the length of hospital stay after the operation (in days). The variable leave takes on the value one if a patient experiences a routine recovery, and the value zero if postoperative intensive care was required. The binary variable OKstatus distinguishes patients based on their postoperative physical status ("1" implies better status).

```
data hernio;
   input patient age gender$ OKstatus leave los;
   datalines;
 1   78  m   1   0   9
 2   60  m   1   0   4
 3   68  m   1   1   7
 4   62  m   0   1  35
 5   76  m   0   0   9
```

```
  6   76  m   1   1   7
  7   64  m   1   1   5
  8   74  f   1   1  16
  9   68  m   0   1   7
 10   79  f   1   0  11
 11   80  f   0   1   4
 12   48  m   1   1   9
 13   35  f   1   1   2
 14   58  m   1   1   4
 15   40  m   1   1   3
 16   19  m   1   1   4
 17   79  m   0   0   3
 18   51  m   1   1   5
 19   57  m   1   1   8
 20   51  m   0   1   8
 21   48  m   1   1   3
 22   48  m   1   1   5
 23   66  m   1   1   8
 24   71  m   1   0   2
 25   75  f   0   0   7
 26    2  f   1   1   0
 27   65  f   1   0  16
 28   42  f   1   0   3
 29   54  m   1   0   2
 30   43  m   1   1   3
 31    4  m   1   1   3
 32   52  m   1   1   8
;
```

While the response variable los is a Poisson count variable, the response variable leave is a binary variable. You can perform separate analysis for the two outcomes, for example, by fitting a logistic model for the operating room exit condition and a Poisson regression model for the length of hospital stay. This, however, would ignore the correlation between the two outcomes. Intuitively, you would expect that the length of postoperative hospital stay is longer for those patients who had more tenuous exit conditions.

The following DATA step converts the data set hernio from the multivariate form to the univariate form. In the multivariate form the responses are stored in separate variables. The GLIMMIX procedure requires the univariate data structure.

```
data hernio_uv;
   length dist $7;
   set hernio;
   response = (leave=1);
   dist     = "Binary";
   output;
   response = los;
   dist     = "Poisson";
   output;
   keep patient age OKstatus response dist;
run;
```

This DATA step expands the 32 observations in the data set hernio into 64 observations, stacking two observations per patient. The character variable dist identifies the distribution that is assumed for the respective observations within a patient. The first observation for each patient corresponds to the binary

response.

The following GLIMMIX statements fit a logistic regression model with two regressors (age and OKStatus) to the binary observations:

```
proc glimmix data=hernio_uv(where=(dist="Binary"));
   model response(event='1') = age OKStatus / s dist=binary;
run;
```

The EVENT=('1') response option requests that PROC GLIMMIX model the probability $\Pr(\text{leave} = 1)$—that is, the probability of routine recovery. The fit statistics and parameter estimates for this univariate analysis are shown in Output 52.5.1. The coefficient for the age effect is negative (–0.07725) and marginally significant at the 5% level ($p = 0.0491$). The negative sign indicates that the probability of routine recovery decreases with age. The coefficient for the OKStatus variable is also negative. Its large standard error and the p-value of 0.7341 indicate, however, that this regressor is not significant.

Output 52.5.1 Univariate Logistic Regression

The GLIMMIX Procedure

Fit Statistics	
-2 Log Likelihood	32.77
AIC (smaller is better)	38.77
AICC (smaller is better)	39.63
BIC (smaller is better)	43.17
CAIC (smaller is better)	46.17
HQIC (smaller is better)	40.23
Pearson Chi-Square	30.37
Pearson Chi-Square / DF	1.05

Parameter Estimates

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	5.7694	2.8245	29	2.04	0.0503
age	-0.07725	0.03761	29	-2.05	0.0491
OKstatus	-0.3516	1.0253	29	-0.34	0.7341

Based on the univariate logistic regression analysis, you would probably want to revisit the model, examine other regressor variables, test for gender effects and interactions, and so forth. The two-regressor model is sufficient for this example. It is illustrative to trace the relative importance of the two regressors through various types of models.

The next statements fit the same regressors to the count data:

```
proc glimmix data=hernio_uv(where=(dist="Poisson"));
   model response = age OKStatus / s dist=Poisson;
run;
```

For this response, both regressors appear to make significant contributions at the 5% significance level (Output 52.5.2). The sign of the coefficient seems appropriate; the length of hospital stay should increase with patient age and be shorter for patients with better preoperative health. The magnitude of the scaled Pearson statistic (4.48) indicates, however, that there is considerable overdispersion in this model. This could

be due to omitted variables or an improper distributional assumption. The importance of preoperative health status, for example, can change with a patient's age, which could call for an interaction term.

Output 52.5.2 Univariate Poisson Regression

The GLIMMIX Procedure

Fit Statistics	
-2 Log Likelihood	215.52
AIC (smaller is better)	221.52
AICC (smaller is better)	222.38
BIC (smaller is better)	225.92
CAIC (smaller is better)	228.92
HQIC (smaller is better)	222.98
Pearson Chi-Square	129.98
Pearson Chi-Square / DF	4.48

Parameter Estimates

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	1.2640	0.3393	29	3.72	0.0008
age	0.01525	0.004454	29	3.42	0.0019
OKstatus	-0.3301	0.1562	29	-2.11	0.0433

You can also model both responses jointly. The following statements request a multivariate analysis:

```
proc glimmix data=hernio_uv;
   class dist;
   model response(event='1') = dist dist*age dist*OKstatus /
                    noint s dist=byobs(dist);
run;
```

The DIST=BYOBS option in the MODEL statement instructs the GLIMMIX procedure to examine the variable dist in order to identify the distribution of an observation. The variable can be character or numeric. See the DIST= option of the MODEL statement for a list of the numeric codes for the various distributions that are compatible with the DIST=BYOBS formulation. Because no LINK= option is specified, the link functions are chosen as the default links that correspond to the respective distributions. In this case, the logit link is applied to the binary observations and the log link is applied to the Poisson outcomes. The dist variable is also listed in the CLASS statement, which enables you to use interaction terms in the MODEL statement to vary the regression coefficients by response distribution. The NOINT option is used here so that the parameter estimates of the joint model are directly comparable to those in Output 52.5.1 and Output 52.5.2.

The "Fit Statistics" and "Parameter Estimates" tables of this bivariate estimation process are shown in Output 52.5.3.

Output 52.5.3 Bivariate Analysis – Independence

The GLIMMIX Procedure

Fit Statistics			
	Binary	Poisson	
Description			Total
-2 Log Likelihood	32.77	215.52	248.29
AIC (smaller is better)	44.77	227.52	260.29
AICC (smaller is better)	48.13	230.88	261.77
BIC (smaller is better)	53.56	236.32	273.25
CAIC (smaller is better)	59.56	242.32	279.25
HQIC (smaller is better)	47.68	230.44	265.40
Pearson Chi-Square	30.37	129.98	160.35
Pearson Chi-Square / DF	1.05	4.48	2.76

Parameter Estimates						
Effect	dist	Estimate	Standard Error	DF	t Value	Pr > |t|
dist	Binary	5.7694	2.8245	58	2.04	0.0456
dist	Poisson	1.2640	0.3393	58	3.72	0.0004
age*dist	Binary	-0.07725	0.03761	58	-2.05	0.0445
age*dist	Poisson	0.01525	0.004454	58	3.42	0.0011
OKstatus*dist	Binary	-0.3516	1.0253	58	-0.34	0.7329
OKstatus*dist	Poisson	-0.3301	0.1562	58	-2.11	0.0389

The "Fit Statistics" table now contains a separate column for each response distribution, as well as an overall contribution. Because the model does not specify any random effects or R-side correlations, the log likelihoods are additive. The parameter estimates and their standard errors in this joint model are identical to those in Output 52.5.1 and Output 52.5.2. The p-values reflect the larger "sample size" in the joint analysis. Note that the coefficients would be different from the separate analyses if the dist variable had not been used to form interactions with the model effects.

There are two ways in which the correlations between the two responses for the same patient can be incorporated. You can induce them through shared random effects or model the dependency directly. The following statements fit a model that induces correlation:

```
proc glimmix data=hernio_uv;
   class patient dist;
   model response(event='1') = dist dist*age dist*OKstatus /
                    noint s dist=byobs(dist);
   random int / subject=patient;
run;
```

Notice that the patient variable has been added to the CLASS statement and as the SUBJECT= effect in the RANDOM statement.

The "Fit Statistics" table in Output 52.5.4 no longer has separate columns for each response distribution, because the data are not independent. The log (pseudo-)likelihood does not factor into additive component that correspond to distributions. Instead, it factors into components associated with subjects.

Output 52.5.4 Bivariate Analysis – Mixed Model

The GLIMMIX Procedure

Fit Statistics	
-2 Res Log Pseudo-Likelihood	226.71
Generalized Chi-Square	52.25
Gener. Chi-Square / DF	0.90

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	patient	0.2990	0.1116

Solutions for Fixed Effects

Effect	dist	Estimate	Standard Error	DF	t Value	Pr > |t|
dist	Binary	5.7783	2.9048	29	1.99	0.0562
dist	Poisson	0.8410	0.5696	29	1.48	0.1506
age*dist	Binary	-0.07572	0.03791	29	-2.00	0.0552
age*dist	Poisson	0.01875	0.007383	29	2.54	0.0167
OKstatus*dist	Binary	-0.4697	1.1251	29	-0.42	0.6794
OKstatus*dist	Poisson	-0.1856	0.3020	29	-0.61	0.5435

The estimate of the variance of the random patient intercept is 0.2990, and the estimated standard error of this variance component estimate is 0.1116. There appears to be significant patient-to-patient variation in the intercepts. The estimates of the fixed effects as well as their estimated standard errors have changed from the bivariate-independence analysis (see Output 52.5.3). When the length of hospital stay and the postoperative condition are modeled jointly, the preoperative health status (variable OKStatus) no longer appears significant. Compare this result to Output 52.5.3; in the separate analyses the initial health status was a significant predictor of the length of hospital stay. A further joint analysis of these data would probably remove this predictor from the model entirely.

A joint model of the second kind, where correlations are modeled directly, is fit with the following GLIMMIX statements:

```
proc glimmix data=hernio_uv;
   class patient dist;
   model response(event='1') = dist dist*age dist*OKstatus /
                   noint s dist=byobs(dist);
   random _residual_ / subject=patient type=chol;
run;
```

Instead of a shared G-side random effect, an R-side covariance structure is used to model the correlations. It is important to note that this is a marginal model that models covariation on the scale of the data. The previous model involves the $\mathbf{Z}\boldsymbol{\gamma}$ random components inside the linear predictor.

The _RESIDUAL_ keyword instructs PROC GLIMMIX to model the R-side correlations. Because of the SUBJECT=PATIENT option, data from different patients are independent, and data from a single patient follow the covariance model specified with the TYPE= option. In this case, a generally unstructured 2×2 covariance matrix is modeled, but in its Cholesky parameterization. This ensures that the resulting covariance matrix is at least positive semidefinite and stabilizes the numerical optimizations.

Output 52.5.5 Bivariate Analysis – Marginal Correlated Error Model

The GLIMMIX Procedure

Fit Statistics	
-2 Res Log Pseudo-Likelihood	240.98
Generalized Chi-Square	58.00
Gener. Chi-Square / DF	1.00

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
CHOL(1,1)	patient	1.0162	0.1334
CHOL(2,1)	patient	0.3942	0.3893
CHOL(2,2)	patient	2.0819	0.2734

Solutions for Fixed Effects

Effect	dist	Estimate	Standard Error	DF	t Value	Pr > |t|
dist	Binary	5.6514	2.8283	26	2.00	0.0563
dist	Poisson	1.2463	0.7189	26	1.73	0.0948
age*dist	Binary	-0.07568	0.03765	26	-2.01	0.0549
age*dist	Poisson	0.01548	0.009432	26	1.64	0.1128
OKstatus*dist	Binary	-0.3421	1.0384	26	-0.33	0.7445
OKstatus*dist	Poisson	-0.3253	0.3310	26	-0.98	0.3349

The "Covariance Parameter Estimates" table in Output 52.5.5 contains three entries for this model, corresponding to a (2×2) covariance matrix for each patient. The Cholesky root of the $\mathbf{R}$ matrix is

$$\mathbf{L} = \begin{bmatrix} 1.0162 & 0 \\ 0.3942 & 2.0819 \end{bmatrix}$$

so that the covariance matrix can be obtained as

$$\mathbf{LL}' = \begin{bmatrix} 1.0162 & 0 \\ 0.3942 & 2.0819 \end{bmatrix} \begin{bmatrix} 1.0162 & 0.3942 \\ 0 & 2.0819 \end{bmatrix} = \begin{bmatrix} 1.0326 & 0.4005 \\ 0.4005 & 4.4897 \end{bmatrix}$$

This is not the covariance matrix of the data, however, because the variance functions need to be accounted for.

The p-values in the "Solutions for Fixed Effects" table indicate the same pattern of significance and nonsignificance as in the conditional model with random patient intercepts.

Example 52.6: Radial Smoothing of Repeated Measures Data

(View the complete code for this example (gmxex06.sas) in the example repository.)

This example of a repeated measures study is taken from Diggle, Liang, and Zeger (1994, p. 100). The data consist of body weights of 27 cows, measured at 23 unequally spaced time points over a period of approximately 22 months. Following Diggle, Liang, and Zeger (1994), one animal is removed from the analysis, one observation is removed according to their Figure 5.7, and the time is shifted to start at 0 and is measured in 10-day increments. The design is a 2×2 factorial, and the factors are the infection of an animal with *M. paratuberculosis* and whether the animal is receiving iron dosing.

The following DATA steps create the data and arrange them in univariate format.

```
data times;
   input time1-time23;
   datalines;
 122  150  166  179  219  247  276  296  324  354  380  445
 478  508  536  569  599  627  655  668  723  751  781
;

data cows;
   if _n_ = 1 then merge times;
   array t{23} time1   - time23;
   array w{23} weight1 - weight23;
   input cow iron infection weight1-weight23 @@;
   do i=1 to 23;
      weight = w{i};
      tpoint = (t{i}-t{1})/10;
      output;
   end;
   keep cow iron infection tpoint weight;
   datalines;
  1 0 0  4.7    4.905  5.011  5.075  5.136  5.165  5.298  5.323
         5.416  5.438  5.541  5.652  5.687  5.737  5.814  5.799
         5.784  5.844  5.886  5.914  5.979  5.927  5.94
  2 0 0  4.868  5.075  5.193  5.22   5.298  5.416  5.481  5.521
         5.617  5.635  5.687  5.768  5.799  5.872  5.886  5.872
         5.914  5.966  5.991  6.016  6.087  6.098  6.153
  3 0 0  4.868  5.011  5.136  5.193  5.273  5.323  5.416  5.46
         5.521  5.58   5.617  5.687  5.72   5.753  5.784  5.784
         5.784  5.814  5.829  5.872  5.927  5.9    5.991
  4 0 0  4.828  5.011  5.136  5.193  5.273  5.347  5.438  5.561
         5.541  5.598  5.67    .     5.737  5.844  5.858  5.872
         5.886  5.927  5.94   5.979  6.052  6.028  6.12
  5 1 0  4.787  4.977  5.043  5.136  5.106  5.298  5.298  5.371
         5.438  5.501  5.561  5.652  5.67   5.737  5.784  5.768
         5.784  5.784  5.829  5.858  5.914  5.9    5.94
  6 1 0  4.745  4.868  5.043  5.106  5.22   5.298  5.347  5.347
         5.416  5.501  5.561  5.58   5.687  5.72   5.737  5.72
         5.737  5.753  5.768  5.784  5.844  5.844  5.9
  7 1 0  4.745  4.905  5.011  5.106  5.165  5.273  5.371  5.416
         5.416  5.521  5.541  5.635  5.687  5.704  5.784  5.768
         5.768  5.814  5.829  5.858  5.94   5.94   6.004
```

```
 8 0 1  4.942  5.106  5.136  5.193  5.298  5.347  5.46   5.521
        5.561  5.58   5.635  5.704  5.784  5.823  5.858  5.9
        5.94   5.991  6.016  6.064  6.052  6.016  5.979
 9 0 1  4.605  4.745  4.868  4.905  4.977  5.22   5.165  5.22
        5.22   5.247  5.298  5.416  5.501  5.521  5.58   5.58
        5.635  5.67   5.72   5.753  5.799  5.829  5.858
10 0 1  4.7    4.868  4.905  4.977  5.011  5.106  5.165  5.22
        5.22   5.22   5.273  5.384  5.438  5.438  5.501  5.501
        5.541  5.598  5.58   5.635  5.687  5.72   5.704
11 0 1  4.828  5.011  5.075  5.165  5.247  5.323  5.394  5.46
        5.46   5.501  5.541  5.609  5.687  5.704  5.72   5.704
        5.704  5.72   5.737  5.768  5.858  5.9    5.94
12 0 1  4.7    4.828  4.905  5.011  5.075  5.165  5.247  5.298
        5.298  5.323  5.416  5.505  5.561  5.58   5.561  5.635
        5.687  5.72   5.72   5.737  5.784  5.814  5.799
13 0 1  4.828  5.011  5.075  5.136  5.22   5.273  5.347  5.416
        5.438  5.416  5.521  5.628  5.67   5.687  5.72   5.72
        5.799  5.858  5.872  5.914  5.94   5.991  6.016
14 0 1  4.828  4.942  5.011  5.075  5.075  5.22   5.273  5.298
        5.323  5.298  5.394  5.489  5.541  5.58   5.617  5.67
        5.704  5.753  5.768  5.814  5.872  5.927  5.927
15 0 1  4.745  4.905  4.977  5.075  5.193  5.22   5.298  5.323
        5.394  5.394  5.438  5.583  5.617  5.652  5.687  5.72
        5.753  5.768  5.814  5.844  5.886  5.886  5.886
16 0 1  4.7    4.868  5.011  5.043  5.106  5.165  5.247  5.298
        5.347  5.371  5.438  5.455  5.617  5.635  5.704  5.737
        5.784  5.768  5.814  5.844  5.886  5.94   5.927
17 1 1  4.605  4.787  4.828  4.942  5.011  5.136  5.22   5.247
        5.273  5.247  5.347  5.366  5.416  5.46   5.541  5.481
        5.501  5.635  5.652  5.598  5.635  5.635  5.598
18 1 1  4.828  4.977  5.011  5.136  5.273  5.298  5.371  5.46
        5.416  5.416  5.438  5.557  5.617  5.67   5.72   5.72
        5.799  5.858  5.886  5.914  5.979  6.004  6.028
19 1 1  4.7    4.905  4.942  5.011  5.043  5.136  5.193  5.193
        5.247  5.22   5.323  5.338  5.371  5.394  5.438  5.416
        5.501  5.561  5.541  5.58   5.652  5.67   5.704
20 1 1  4.745  4.905  4.977  5.043  5.136  5.273  5.347  5.394
        5.416  5.394  5.521  5.617  5.617  5.617  5.67   5.635
        5.652  5.687  5.652  5.617  5.687  5.768  5.814
21 1 1  4.787  4.942  4.977  5.106  5.165  5.247  5.323  5.416
        5.394  5.371  5.438  5.521  5.521  5.561  5.635  5.617
        5.687  5.72   5.737  5.737  5.768  5.768  5.704
22 1 1  4.605  4.828  4.828  4.977  5.043  5.165  5.22   5.273
        5.247  5.22   5.298  5.375  5.371  5.416  5.501  5.501
        5.521  5.561  5.617  5.635  5.72   5.737  5.768
23 1 1  4.7    4.905  5.011  5.075  5.106  5.22   5.22   5.298
        5.323  5.347  5.416  5.472  5.501  5.541  5.598  5.598
        5.598  5.652  5.67   5.704  5.737  5.768  5.784
24 1 1  4.745  4.942  5.011  5.075  5.106  5.247  5.273  5.323
        5.347  5.371  5.416  5.481  5.501  5.541  5.598  5.598
        5.635  5.687  5.704  5.72   5.829  5.844  5.9
25 1 1  4.654  4.828  4.828  4.977  4.977  5.043  5.136  5.165
        5.165  5.165  5.193  5.204  5.22   5.273  5.371  5.347
        5.46   5.58   5.635  5.67   5.753  5.799  5.844
```

```
 26 1 1  4.828  4.977  5.011  5.106  5.165  5.22   5.273  5.323
         5.371  5.394  5.46   5.576  5.652  5.617  5.687  5.67
         5.72   5.784  5.784  5.784  5.829  5.814  5.844
;
```

The mean response profiles of the cows are not of particular interest; what matters are inferences about the Iron effect, the Infection effect, and their interaction. Nevertheless, the body weight of the cows changes over the 22-month period, and you need to account for these changes in the analysis. A reasonable approach is to apply the approximate low-rank smoother to capture the trends over time. This approach frees you from having to stipulate a parametric model for the response trajectories over time. In addition, you can test hypotheses about the smoothing parameter; for example, whether it should be varied by treatment.

The following statements fit a model with a 2×2 factorial treatment structure and smooth trends over time, choosing the Newton-Raphson algorithm with ridging for the optimization:

```
proc glimmix data=cows;
   t2 = tpoint / 100;
   class cow iron infection;
   model weight = iron infection iron*infection tpoint;
   random t2 / type=rsmooth subject=cow
                   knotmethod=kdtree(bucket=100 knotinfo);
   output out=gmxout pred(blup)=pred;
   nloptions tech=newrap;
run;
```

The continuous time effect appears in both the MODEL statement (tpoint) and the RANDOM statement (t2). Because the variance of the radial smoothing component depends on the temporal metric, the time scale was rescaled for the RANDOM effect to move the parameter estimate away from the boundary. The knots of the radial smoother are selected as the vertices of a k-d tree. Specifying BUCKET=100 sets the bucket size of the tree to $b = 100$. Because measurements at each time point are available for 26 (or 25) cows, this groups approximately four time points in a single bucket. The KNOTINFO keyword of the KNOTMETHOD= option requests a printout of the knot locations for the radial smoother. The OUTPUT statement saves the predictions of the mean of each observations to the data set gmxout. Finally, the TECH=NEWRAP option in the NLOPTIONS statement specifies the Newton-Raphson algorithm for the optimization technique.

The "Class Level Information" table lists the number of levels of the Cow, Iron, and Infection effects (Output 52.6.1).

Output 52.6.1 Model Information and Class Levels in Repeated Measures Analysis

The GLIMMIX Procedure

Model Information	
Data Set	WORK.COWS
Response Variable	weight
Response Distribution	Gaussian
Link Function	Identity
Variance Function	Default
Variance Matrix Blocked By	cow
Estimation Technique	Restricted Maximum Likelihood
Degrees of Freedom Method	Containment

Output 52.6.1 *continued*

Class Level Information		
Class	**Levels**	**Values**
cow	26	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
iron	2	0 1
infection	2	0 1

The "Radial Smoother Knots for RSmooth(t2)" table displays the knots computed from the vertices of the t2 k-d tree (Output 52.6.2). Notice that knots are spaced unequally and that the extreme time points are among the knot locations. The "Number of Observations" table shows that one observation was not used in the analysis. The 12th observation for cow 4 has a missing value.

Output 52.6.2 Knot Information and Number of Observations

Radial Smoother Knots for RSmooth(t2)	
Knot Number	**t2**
1	0
2	0.04400
3	0.1250
4	0.2020
5	0.3230
6	0.4140
7	0.5050
8	0.6010
9	0.6590

Number of Observations Read	598
Number of Observations Used	597

The "Dimensions" table shows that the model contains only two covariance parameters, the G-side variance of the spline coefficients (σ^2) and the R-side scale parameter (ϕ, Output 52.6.3). For each subject (cow), there are nine columns in the $\mathbf{Z}$ matrix, one per knot location. The GLIMMIX procedure processes these data by subjects (cows).

Output 52.6.3 Dimensions Information in Repeated Measures Analysis

Dimensions	
G-side Cov. Parameters	1
R-side Cov. Parameters	1
Columns in X	10
Columns in Z per Subject	9
Subjects (Blocks in V)	26
Max Obs per Subject	23

The "Optimization Information" table displays information about the optimization process. Because fixed effects and the residual scale parameter can be profiled from the optimization, the iterative algorithm involves

only a single covariance parameter, the variance of the spline coefficients (Output 52.6.4).

Output 52.6.4 Optimization Information in Repeated Measures Analysis

Optimization Information	
Optimization Technique	Newton-Raphson
Parameters in Optimization	1
Lower Boundaries	1
Upper Boundaries	0
Fixed Effects	Profiled
Residual Variance	Profiled
Starting From	Data

After 11 iterations, the optimization process terminates (Output 52.6.5). In this case, the absolute gradient convergence criterion was met.

Output 52.6.5 Iteration History and Convergence Status

Iteration History					
Iteration	**Restarts**	**Evaluations**	**Objective Function**	**Change**	**Max Gradient**
0	**0**	4	-1302.549272	.	20.33682
1	**0**	3	-1451.587367	149.03809501	9.940495
2	**0**	3	-1585.640946	134.05357887	4.71531
3	**0**	3	-1694.516203	108.87525722	2.176741
4	**0**	3	-1775.290458	80.77425512	0.978577
5	**0**	3	-1829.966584	54.67612585	0.425724
6	**0**	3	-1862.878184	32.91160012	0.175992
7	**0**	3	-1879.329133	16.45094874	0.066061
8	**0**	3	-1885.175082	5.84594887	0.020137
9	**0**	3	-1886.238032	1.06295071	0.00372
10	**0**	3	-1886.288519	0.05048659	0.000198
11	**0**	3	-1886.288673	0.00015425	6.364E-7

Convergence criterion (ABSGCONV=0.00001) satisfied.

The generalized chi-square statistic in the “Fit Statistics” table is small for this model (Output 52.6.6). There is very little residual variation. The radial smoother is associated with 433.55 residual degrees of freedom, computed as 597 minus the trace of the smoother matrix.

Output 52.6.6 Fit Statistics in Repeated Measures Analysis

Fit Statistics	
-2 Res Log Likelihood	-1886.29
AIC (smaller is better)	-1882.29
AICC (smaller is better)	-1882.27
BIC (smaller is better)	-1879.77
CAIC (smaller is better)	-1877.77
HQIC (smaller is better)	-1881.56
Generalized Chi-Square	0.47
Gener. Chi-Square / DF	0.00
Radial Smoother df(res)	433.55

The "Covariance Parameter Estimates" table in Output 52.6.7 displays the estimates of the covariance parameters. The variance of the random spline coefficients is estimated as $\widehat{\sigma}^2 = 0.5961$, and the scale parameter (=residual variance) estimate is $\widehat{\phi} = 0.0008$.

Output 52.6.7 Estimated Covariance Parameters

Covariance Parameter Estimates			
Cov Parm	Subject	Estimate	Standard Error
Var[RSmooth(t2)]	cow	0.5961	0.08144
Residual		0.000800	0.000059

The "Type III Tests of Fixed Effects" table displays F tests for the fixed effects in the MODEL statement (Output 52.6.8). There is a strong infection effect as well as the absence of an interaction between infection with *M. paratuberculosis* and iron dosing. It is important to note, however, that the interpretation of these tests rests on the assumption that the random effects in the mixed model have zero mean; in this case, the radial smoother coefficients.

Output 52.6.8 Tests of Fixed Effects

Type III Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
iron	1	358	3.59	0.0588
infection	1	358	21.16	<.0001
iron*infection	1	358	0.09	0.7637
tpoint	1	358	53.88	<.0001

A graph of the observed data and fitted profiles in the four groups is produced with the following statements (Output 52.6.9):

```
data plot;
   set gmxout;
   length group $26;
   if      (iron=0) and (infection=0) then group='Control Group (n=4)';
   else if (iron=1) and (infection=0) then group='Iron - No Infection (n=3)';
   else if (iron=0) and (infection=1) then group='No Iron - Infection (n=9)';
```

```
   else group = 'Iron - Infection (n=10)';
run;
proc sort data=plot; by group cow;
run;

proc sgpanel data=plot noautolegend;
   title 'Radial Smoothing With Cow-Specific Trends';
   label tpoint='Time' weight='log(Weight)';
   panelby group / columns=2 rows=2;
   scatter x=tpoint y=weight;
   series  x=tpoint y=pred / group=cow lineattrs=GraphFit;
run;
```

Output 52.6.9 Observed and Predicted Profiles

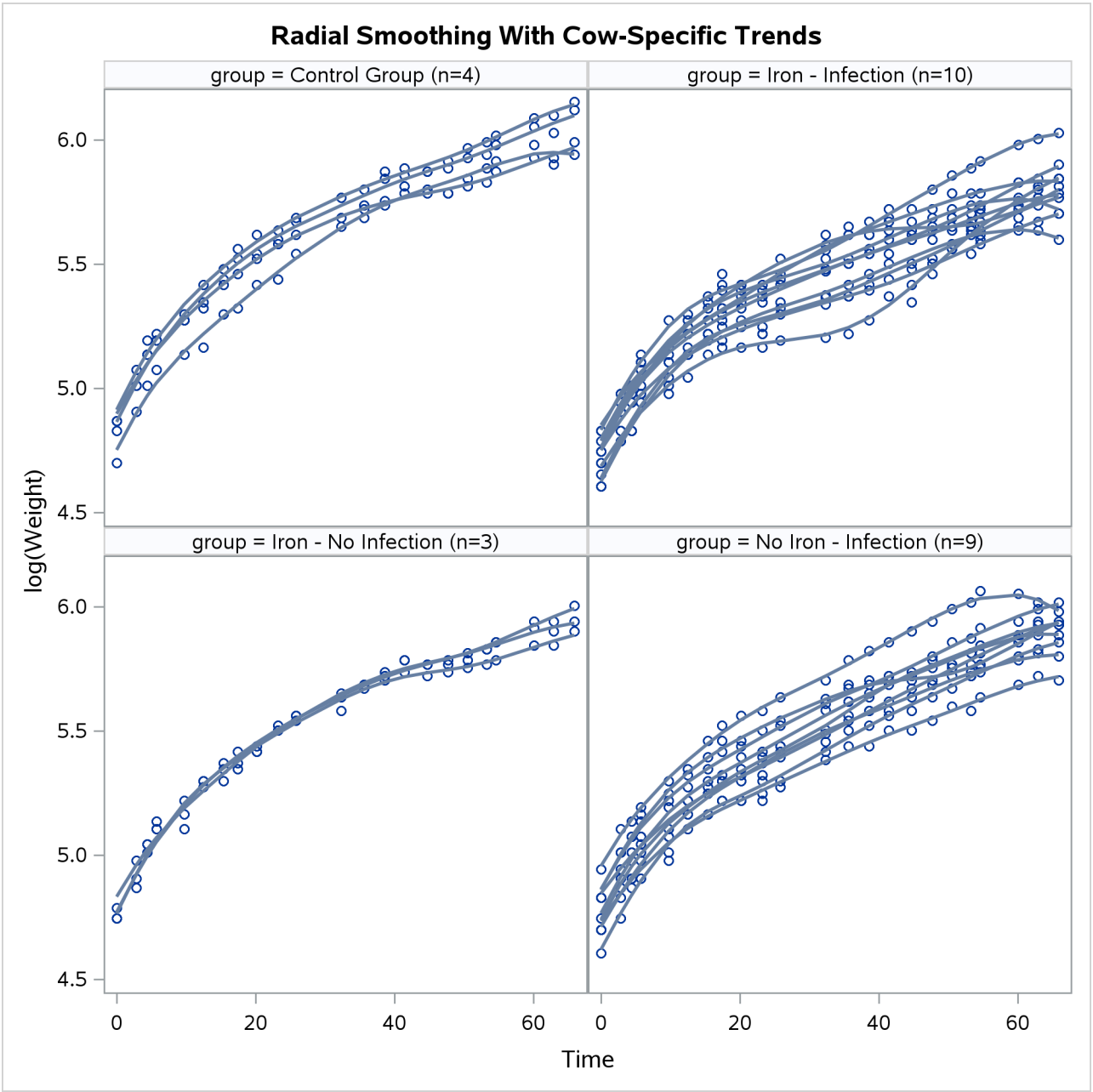

The trends are quite smooth, and you can see how the radial smoother adapts to the cow-specific profile. This is the reason for the small scale parameter estimate, $\widehat{\phi} = 0.008$. Comparing the panels at the top to the panels at the bottom of Output 52.6.9 reveals the effect of Infection. A comparison of the panels on the left to those on the right indicates the weak Iron effect.

The smoothing parameter in this analysis is related to the covariance parameter estimates. Because there is only one radial smoothing variance component, the amount of smoothing is the same in all four treatment groups. To test whether the smoothing parameter should be varied by group, you can refine the analysis of the previous model. The following statements fit the same general model, but they vary the covariance

parameters by the levels of the Iron*Infection interaction. This is accomplished with the GROUP= option in the RANDOM statement.

```
ods select OptInfo FitStatistics CovParms;
proc glimmix data=cows;
   t2 = tpoint / 100;
   class cow iron infection;
   model weight = iron infection iron*infection tpoint;
   random t2 / type=rsmooth
               subject=cow
               group=iron*infection
               knotmethod=kdtree(bucket=100);
   nloptions tech=newrap;
run;
```

All observations that have the same value combination of the Iron and Infection effects share the same covariance parameter. As a consequence, you obtain different smoothing parameters result in the four groups.

In Output 52.6.10, the "Optimization Information" table shows that there are now four covariance parameters in the optimization, one spline coefficient variance for each group.

Output 52.6.10 Analysis with Group-Specific Smoothing Parameter

The GLIMMIX Procedure

Optimization Information	
Optimization Technique	Newton-Raphson
Parameters in Optimization	4
Lower Boundaries	4
Upper Boundaries	0
Fixed Effects	Profiled
Residual Variance	Profiled
Starting From	Data

Fit Statistics	
-2 Res Log Likelihood	-1887.95
AIC (smaller is better)	-1877.95
AICC (smaller is better)	-1877.85
BIC (smaller is better)	-1871.66
CAIC (smaller is better)	-1866.66
HQIC (smaller is better)	-1876.14
Generalized Chi-Square	0.48
Gener. Chi-Square / DF	0.00
Radial Smoother df(res)	434.72

Covariance Parameter Estimates

Cov Parm	Subject	Group	Estimate	Standard Error
Var[RSmooth(t2)]	cow	iron*infection 0 0	0.4788	0.1922
Var[RSmooth(t2)]	cow	iron*infection 0 1	0.5152	0.1182
Var[RSmooth(t2)]	cow	iron*infection 1 0	0.4904	0.2195
Var[RSmooth(t2)]	cow	iron*infection 1 1	0.7105	0.1409
Residual			0.000807	0.000060

Varying this variance component by groups has changed the –2 Res Log Likelihood from –1886.29 to –1887.95 (Output 52.6.10). The difference, 1.66, can be viewed (asymptotically) as the realization of a chi-square random variable with three degrees of freedom. The difference is not significant ($p = 0.64586$). The "Covariance Parameter Estimates" table confirms that the estimates of the spline coefficient variance are quite similar in the four groups, ranging from 0.4788 to 0.7105.

Finally, you can apply a different technique for varying the temporal trends among the cows. From Output 52.6.9 it appears that an assumption of parallel trends within groups might be reasonable. In other words, you can fit a model in which the "overall" trend over time in each group is modeled nonparametrically, and this trend is shifted up or down to capture the behavior of the individual cow. You can accomplish this with the following statements:

```
ods select FitStatistics CovParms;
proc glimmix data=cows;
   t2 = tpoint / 100;
   class cow iron infection;
   model weight = iron infection iron*infection tpoint;
   random t2 / type=rsmooth
               subject=iron*infection
               knotmethod=kdtree(bucket=100);
   random intercept / subject=cow;
   output out=gmxout pred(blup)=pred;
   nloptions tech=newrap;
run;
```

There are now two subject effects in this analysis. The first RANDOM statement applies the radial smoothing and identifies the experimental conditions as the subject. For each condition, a separate realization of the random spline coefficients is obtained. The second RANDOM statement adds a random intercept to the trend for each cow. This random intercept results in the parallel shift of the trends over time.

Results from this analysis are shown in Output 52.6.11.

Output 52.6.11 Analysis with Parallel Shifts

The GLIMMIX Procedure

Fit Statistics	
-2 Res Log Likelihood	-1788.52
AIC (smaller is better)	-1782.52
AICC (smaller is better)	-1782.48
BIC (smaller is better)	-1788.52
CAIC (smaller is better)	-1785.52
HQIC (smaller is better)	-1788.52
Generalized Chi-Square	1.17
Gener. Chi-Square / DF	0.00
Radial Smoother df(res)	547.21

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Var[RSmooth(t2)]	iron*infection	0.5398	0.1940
Intercept	cow	0.007122	0.002173
Residual		0.001976	0.000121

Because the parallel shift model is not nested within either one of the previous models, the models cannot be compared with a likelihood ratio test. However, you can draw on the other fit statistics.

All statistics indicate that this model does not fit the data as well as the initial model that varies the spline coefficients by cow. The Pearson chi-square statistic is more than twice as large as in the previous model, indicating much more residual variation in the fit. On the other hand, this model generates only four sets of spline coefficients, one for each treatment group, and thus retains more residual degrees of freedom.

The "Covariance Parameter Estimates" table in Output 52.6.11 displays the solutions for the covariance parameters. The estimate of the variance of the spline coefficients is not that different from the estimate obtained in the first model (0.5961). The residual variance, however, has more than doubled.

Using similar SAS statements as previously, you can produce a plot of the observed and predicted profiles (Output 52.6.12).

The parallel shifts of the nonparametric smooths are clearly visible in Output 52.6.12. In the groups receiving only iron or only an infection, the parallel lines assumption holds quite well. In the control group and the group receiving iron and the infection, the parallel shift assumption does not hold as well. Two of the profiles in the iron-only group are nearly indistinguishable.

Output 52.6.12 Observed and Predicted Profiles

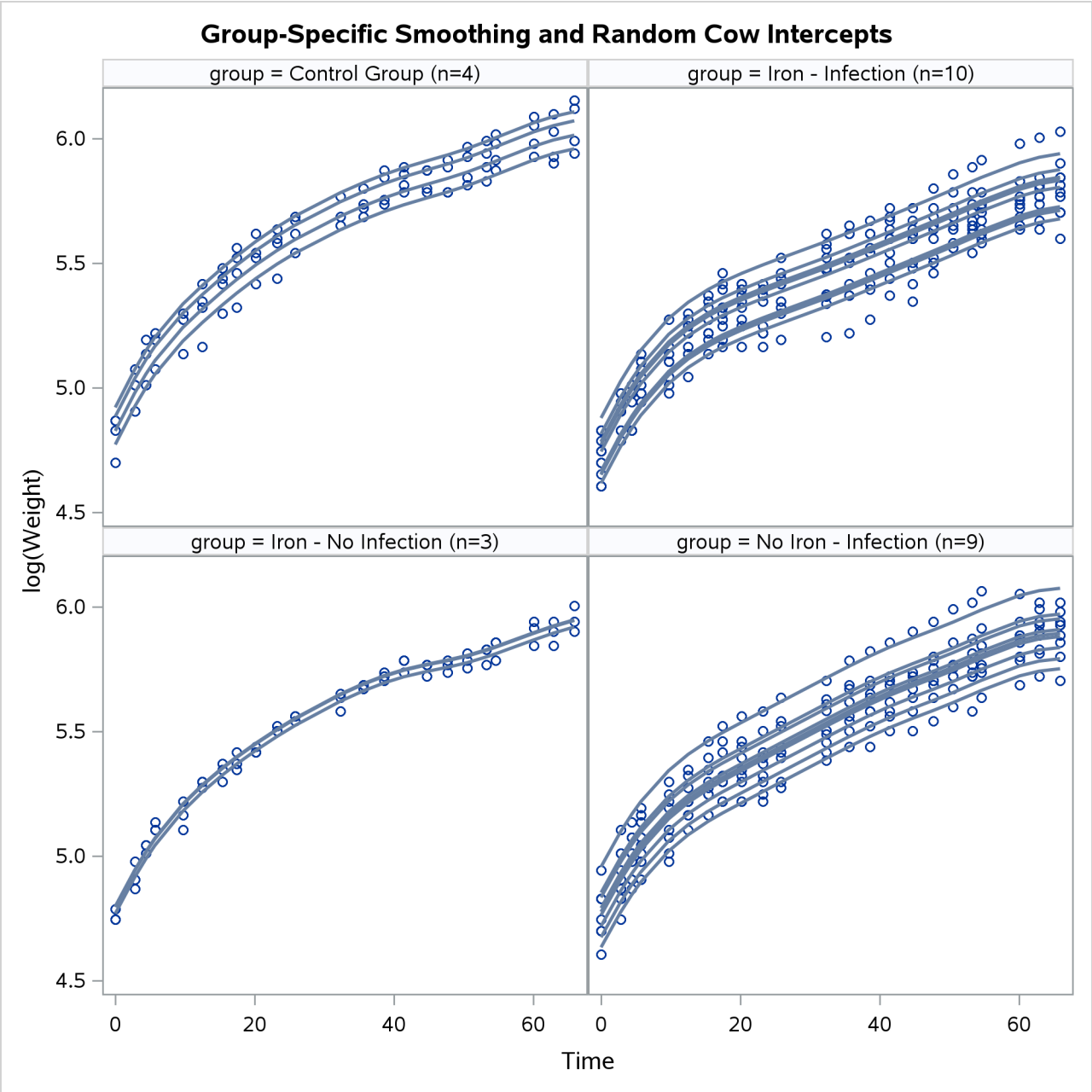

This example demonstrates that mixed model smoothing techniques can be applied not only to achieve scatter plot smoothing, but also to longitudinal or repeated measures data. You can then use the SUBJECT= option in the RANDOM statement to obtain independent sets of spline coefficients for different subjects, and the GROUP= option in the RANDOM statement to vary the degree of smoothing across groups. Also, radial smoothers can be combined with other random effects. For the data considered here, the appropriate model is one with a single smoothing parameter for all treatment group and cow-specific spline coefficients.

Example 52.7: Isotonic Contrasts for Ordered Alternatives

(View the complete code for this example (gmxex07.sas) in the example repository.)

Dose response studies often focus on testing for monotone increasing or decreasing behavior in the mean values of the dependent variable. Hirotsu and Srivastava (2000) demonstrate one approach by using data that originally appeared in Moriguchi (1976). The data, which follow, consist of ferrite cores subjected to four increasing temperatures. The response variable is the magnetic force of each core.

```
data FerriteCores;
   do Temp = 1 to 4;
      do rep = 1 to 5; drop rep;
         input MagneticForce @@;
         output;
      end;
   end;
   datalines;
10.8  9.9 10.7 10.4  9.7
10.7 10.6 11.0 10.8 10.9
11.9 11.2 11.0 11.1 11.3
11.4 10.7 10.9 11.3 11.7
;
```

It is of interest to test whether the magnetic force of the cores rises monotonically with temperature. The approach of Hirotsu and Srivastava (2000) depends on the lower confidence limits of the *isotonic contrasts* of the force means at each temperature, adjusted for multiplicity. The corresponding isotonic contrast compares the average of a particular group and the preceding groups with the average of the succeeding groups. You can compute adjusted confidence intervals for isotonic contrasts by using the LSMESTIMATE statement.

The following statements request an analysis of the FerriteCores data as a one-way design and multiplicity-adjusted lower confidence limits for the isotonic contrasts. For the multiplicity adjustment, the LSMESTIMATE statement employs simulation, which provides adjusted *p*-values and lower confidence limits that are exact up to Monte Carlo error.

```
proc glimmix data=FerriteCores;
   class Temp;
   model MagneticForce = Temp;
   lsmestimate Temp
        'avg(1:1)<avg(2:4)' -3  1  1  1 divisor=3,
        'avg(1:2)<avg(3:4)' -1 -1  1  1 divisor=2,
        'avg(1:3)<avg(4:4)' -1 -1 -1  3 divisor=3
        / adjust=simulate(seed=1) cl upper;
   ods select LSMestimates;
run;
```

The results are shown in Output 52.7.1.

Output 52.7.1 Analysis of LS-Means with Isotonic Contrasts

The GLIMMIX Procedure

Least Squares Means Estimates
Adjustment for Multiplicity: Simulated

Effect	Label	Estimate	Standard Error	DF	t Value	Tails	Pr > t	Adj P	Alpha	Lower	Upper	Adj Lower	Adj Upper
Temp	avg(1:1)<avg(2:4)	0.8000	0.1906	16	4.20	Upper	0.0003	0.0010	0.05	0.4672	Infty	0.3771	Infty
Temp	avg(1:2)<avg(3:4)	0.7000	0.1651	16	4.24	Upper	0.0003	0.0009	0.05	0.4118	Infty	0.3337	Infty
Temp	avg(1:3)<avg(4:4)	0.4000	0.1906	16	2.10	Upper	0.0260	0.0625	0.05	0.06721	Infty	-0.02291	Infty

With an adjusted p-value of 0.001, the magnetic force at the first temperature is significantly less than the average of the other temperatures. Likewise, the average of the first two temperatures is significantly less than the average of the last two ($p = 0.0009$). However, the magnetic force at the last temperature is not significantly greater than the average magnetic force of the others ($p = 0.0625$). These results indicate a significant monotone increase over the first three temperatures, but not across all four temperatures.

Example 52.8: Adjusted Covariance Matrices of Fixed Effects

(View the complete code for this example (gmxex08.sas) in the example repository.)

The following data are from Pothoff and Roy (1964) and consist of growth measurements for 11 girls and 16 boys at ages 8, 10, 12, and 14. Some of the observations are suspect (for example, the third observation for person 20); however, all of the data are used here for comparison purposes.

```
data pr;
   input child gender$ y1 y2 y3 y4;
   array yy y1-y4;
   do time=1 to 4;
      age = time*2 + 6;
      y   = yy{time};
      output;
   end;
   drop y1-y4;
   datalines;
 1   F   21.0    20.0    21.5    23.0
 2   F   21.0    21.5    24.0    25.5
 3   F   20.5    24.0    24.5    26.0
 4   F   23.5    24.5    25.0    26.5
 5   F   21.5    23.0    22.5    23.5
 6   F   20.0    21.0    21.0    22.5
 7   F   21.5    22.5    23.0    25.0
 8   F   23.0    23.0    23.5    24.0
 9   F   20.0    21.0    22.0    21.5
10   F   16.5    19.0    19.0    19.5
11   F   24.5    25.0    28.0    28.0
12   M   26.0    25.0    29.0    31.0
13   M   21.5    22.5    23.0    26.5
14   M   23.0    22.5    24.0    27.5
15   M   25.5    27.5    26.5    27.0
```

```
16   M   20.0    23.5    22.5    26.0
17   M   24.5    25.5    27.0    28.5
18   M   22.0    22.0    24.5    26.5
19   M   24.0    21.5    24.5    25.5
20   M   23.0    20.5    31.0    26.0
21   M   27.5    28.0    31.0    31.5
22   M   23.0    23.0    23.5    25.0
23   M   21.5    23.5    24.0    28.0
24   M   17.0    24.5    26.0    29.5
25   M   22.5    25.5    25.5    26.0
26   M   23.0    24.5    26.0    30.0
27   M   22.0    21.5    23.5    25.0
;
```

Jennrich and Schluchter (1986) analyze these data with various models for the fixed effects and the covariance structure. The strategy here is to fit a growth curve model for the boys and girls and to account for subject-to-subject variation through G-side random effects. In addition, serial correlation among the observations within each child is accounted for by a time series process. The data are assumed to be Gaussian, and their –2 restricted log likelihood is minimized to estimate the model parameters.

The following statements fit a mixed model in which a separate growth curve is assumed for each gender:

```
proc glimmix data=pr;
   class child gender time;
   model y = gender age gender*age / covb(details) ddfm=kr;
   random intercept age / type=chol sub=child;
   random time / subject=child type=ar(1) residual;
   ods select ModelInfo CovB CovBModelBased CovBDetails;
run;
```

The growth curve for an individual child differs from the gender-specific trend because of a random intercept and a random slope. The two G-side random effects are assumed to be correlated. Their unstructured covariance matrix is parameterized in terms of the Cholesky root to guarantee a positive (semi-)definite estimate. An AR(1) covariance structure is modeled for the observations over time for each child. Notice the RESIDUAL option in the second RANDOM statement. It identifies this as an R-side random effect.

The DDFM=KR option requests that the covariance matrix of the fixed-effect parameter estimates and denominator degrees of freedom for t and F tests are determined according to Kenward and Roger (1997). This is reflected in the "Model Information" table (Output 52.8.1).

Output 52.8.1 Model Information with DDFM=KR

The GLIMMIX Procedure

Model Information	
Data Set	WORK.PR
Response Variable	y
Response Distribution	Gaussian
Link Function	Identity
Variance Function	Default
Variance Matrix Blocked By	child
Estimation Technique	Restricted Maximum Likelihood
Degrees of Freedom Method	Kenward-Roger
Fixed Effects SE Adjustment	Kenward-Roger

The COVB option in the MODEL statement requests that the covariance matrix used for inference about fixed effects in this model is displayed; this is the Kenward-Roger-adjusted covariance matrix. The DETAILS suboption requests that the unadjusted covariance matrix is also displayed (Output 52.8.2). In addition, a table of diagnostic measures for the covariance matrices is produced.

Output 52.8.2 Model-Based and Adjusted Covariance Matrix

Model Based Covariance Matrix for Fixed Effects (Unadjusted)

Effect	gender	Row	Col1	Col2	Col3	Col4	Col5	Col6
Intercept		**1**	0.9969	-0.9969		-0.07620	0.07620	
gender	F	**2**	-0.9969	2.4470		0.07620	-0.1870	
gender	M	**3**						
age		**4**	-0.07620	0.07620		0.007581	-0.00758	
age*gender	F	**5**	0.07620	-0.1870		-0.00758	0.01861	
age*gender	M	**6**						

Covariance Matrix for Fixed Effects

Effect	gender	Row	Col1	Col2	Col3	Col4	Col5	Col6
Intercept		**1**	0.9724	-0.9724		-0.07412	0.07412	
gender	F	**2**	-0.9724	2.3868		0.07412	-0.1819	
gender	M	**3**						
age		**4**	-0.07412	0.07412		0.007256	-0.00726	
age*gender	F	**5**	0.07412	-0.1819		-0.00726	0.01781	
age*gender	M	**6**						

Diagnostics for Covariance Matrices of Fixed Effects

		Model-Based	Adjusted
Dimensions	**Rows**	6	6
	Non-zero entries	16	16
Summaries	**Trace**	3.4701	3.3843
	Log determinant	-11.95	-12.17
Eigenvalues	**> 0**	4	4
	= 0	2	2
	max abs	2.972	2.8988
	min abs non-zero	0.0009	0.0008
	Condition number	3467.8	3698.2
Norms	**Frobenius**	3.0124	2.9382
	Infinity	3.7072	3.6153
Comparisons	**Concordance correlation**		0.9979
	Discrepancy function		0.0084
	Frobenius norm of difference		0.0742
	Trace(Adjusted Inv(MBased))		3.7801

Determinant and inversion results apply to the nonsingular partitions of the covariance matrices.

The "Diagnostics for Covariance Matrices" table in Output 52.8.2 consists of several sections. The trace

and log determinant of covariance matrices are general scalar summaries that are sometimes used in direct comparisons, or in formulating further statistics, such as the difference of log determinants. The trace simply represents the sum of the variances of all fixed-effects parameters.

The two matrices have the same number of positive and zero eigenvalues; hence they are of the same rank. There are no negative eigenvalues; hence the matrices are positive semi-definite.

The "Comparisons" section of the table provides several statistics that set the matrices in relationship. The statistics enable you to assess the extent to which the adjustment affected the model-based matrix. If the two matrices are identical, the concordance correlation equals 1, the discrepancy function and the Frobenius norm of the differences equal 0, and the trace of the adjusted and the (generalized) inverse of the model-based matrix equals the rank. See the section "Exploring and Comparing Covariance Matrices" on page 3955 for computational details regarding these statistics. With increasing discrepancy between the matrices, the difference norm and discrepancy function increase, the concordance correlation falls below 1, and the trace deviates from the rank. In this particular example, there is strong agreement between the two matrices; the adjustment to the covariance matrix associated with DDFM=KR is only slight. It is noteworthy, however, that the trace of the adjusted covariance matrix falls short of the trace of the unadjusted one. Indeed, from Output 52.8.2 you can see that the diagonal elements of the adjusted covariance matrices are uniformly smaller than those of the model-based covariance matrix.

Standard error "shrinkage" for the Kenward-Roger covariance adjustment is due to the term $-0.25\mathbf{R}_{ij}$ in equation (3) of Kenward and Roger (1997), which is nonzero for covariance structures with second derivatives, such as the TYPE=ANTE(1), TYPE=AR(1), TYPE=ARH(1), TYPE=ARMA(1,1), TYPE=CHOL, TYPE=CSH, TYPE=FA0(q), TYPE=TOEPH, and TYPE=UNR structures and all TYPE=SP() structures.

For covariance structures that are linear in the parameters, $\mathbf{R}_{ij} = \mathbf{0}$. You can add the FIRSTORDER suboption to the DDFM=KR option to request that second derivative matrices $\mathbf{R}_{ij}$ are excluded from computing the covariance matrix adjustment. The resulting covariance adjustment is that of Kackar and Harville (1984) and Harville and Jeske (1992). This estimator is denoted as $\widetilde{m}^{@}$ in Harville and Jeske (1992) and is referred to there as the Prasad-Rao estimator after related work by Prasad and Rao (1990). This standard error adjustment is guaranteed to be positive (semi-)definite. The following statements fit the model with the Kackar-Harville-Jeske estimator and compare model-based and adjusted covariance matrices:

```
proc glimmix data=pr;
   class child gender time;
   model y = gender age gender*age / covb(details)
                                     ddfm=kr(firstorder);
   random intercept age / type=chol sub=child;
   random time / subject=child type=ar(1) residual;
   ods select ModelInfo CovB CovBDetails;
run;
```

The standard error adjustment is reflected in the "Model Information" table (Output 52.8.3).

Output 52.8.3 Model Information with DDFM=KR(FIRSTORDER)

The GLIMMIX Procedure

Model Information	
Data Set	WORK.PR
Response Variable	y
Response Distribution	Gaussian
Link Function	Identity
Variance Function	Default
Variance Matrix Blocked By	child
Estimation Technique	Restricted Maximum Likelihood
Degrees of Freedom Method	Kenward-Roger
Fixed Effects SE Adjustment	Prasad-Rao-Kackar-Harville-Jeske

Output 52.8.4 displays the adjusted covariance matrix. Notice that the elements of this matrix, in particular the diagonal elements, are larger in absolute value than those of the model-based estimator (Output 52.8.2).

Output 52.8.4 Adjusted Covariance Matrix and Comparison to Model-Based Estimator

Covariance Matrix for Fixed Effects

Effect	gender	Row	Col1	Col2	Col3	Col4	Col5	Col6
Intercept		**1**	1.0122	-1.0122		-0.07758	0.07758	
gender	F	**2**	-1.0122	2.4845		0.07758	-0.1904	
gender	M	**3**						
age		**4**	-0.07758	0.07758		0.007706	-0.00771	
age*gender	F	**5**	0.07758	-0.1904		-0.00771	0.01891	
age*gender	M	**6**						

Output 52.8.4 *continued*

Diagnostics for Covariance Matrices of Fixed Effects			
		Model-Based	Adjusted
Dimensions	Rows	6	6
	Non-zero entries	16	16
Summaries	Trace	3.4701	3.5234
	Log determinant	-11.95	-11.91
Eigenvalues	> 0	4	4
	= 0	2	2
	max abs	2.972	3.0176
	min abs non-zero	0.0009	0.0009
	Condition number	3467.8	3513.4
Norms	Frobenius	3.0124	3.0587
	Infinity	3.7072	3.7647
Comparisons	Concordance correlation		0.9999
	Discrepancy function		0.0003
	Frobenius norm of difference		0.0463
	Trace(Adjusted Inv(MBased))		4.0352
Determinant and inversion results apply to the nonsingular partitions of the covariance matrices.			

The "Comparisons" statistics show that the model-based and adjusted covariance matrix of the fixed-effects parameter estimates are very similar. The concordance correlation is near 1, the discrepancy is near zero, and the trace is very close to the number of positive eigenvalues. This is due to the balanced nature of these repeated measures data. Shrinkage of standard errors, however, can not occur with the Kackar-Harville-Jeske estimator.

Example 52.9: Testing Equality of Covariance and Correlation Matrices

(View the complete code for this example (gmxex09.sas) in the example repository.)

Fisher's iris data are widely used in multivariate statistics. They comprise measurements in millimeters of four flower attributes, the length and width of sepals and petals for 50 specimens from each of three species, *Iris setosa*, *I. versicolor*, and *I. virginica* (Fisher 1936).

When modeling multiple attributes from the same specimen, correlations among measurements from the same flower must be taken into account. Unstructured covariance matrices are common in this multivariate setting. Species comparisons can focus on comparisons of mean response, but comparisons of the variation and covariation are also of interest. In this example, the equivalence of covariance and correlation matrices among the species are examined.

The iris data set is available in the Sashelp library. The following step displays the first 10 observations of the iris data in multivariate format—that is, each observation contains multiple response variables. The DATA step that follows creates a data set in univariate form, where each observation corresponds to a single response variable. This is the form needed by the GLIMMIX procedure.

```
proc print data=Sashelp.iris(obs=10);
run;
```

Output 52.9.1 Fisher (1936) Iris Data

Obs	Species	SepalLength	SepalWidth	PetalLength	PetalWidth
1	Setosa	50	33	14	2
2	Setosa	46	34	14	3
3	Setosa	46	36	10	2
4	Setosa	51	33	17	5
5	Setosa	55	35	13	2
6	Setosa	48	31	16	2
7	Setosa	52	34	14	2
8	Setosa	49	36	14	1
9	Setosa	44	32	13	2
10	Setosa	50	35	16	6

```
data iris_univ;
   set sashelp.iris;
   retain id 0;
   array y (4) SepalLength SepalWidth PetalLength PetalWidth;
   id+1;
   do var=1 to 4;
      response = y{var};
      output;
   end;
   drop SepalLength SepalWidth PetalLength PetalWidth:;
run;
```

The following GLIMMIX statements fit a model with separate unstructured covariance matrices for each species:

```
ods select FitStatistics CovParms CovTests;
proc glimmix data=iris_univ;
   class species var id;
   model response = species*var;
   random _residual_ / type=un group=species subject=id;
   covtest homogeneity;
run;
```

The mean function is modeled as a cell-means model that allows for different means for each species and outcome variable. The covariances are modeled directly (R-side) rather than through random effects. The ID variable identifies the individual plant, so that responses from different plants are independent. The GROUP=SPECIES option varies the parameters of the unstructured covariance matrix by species. Hence, this model has 30 covariance parameters: 10 unique parameters for a (4×4) covariance matrix for each of three species.

The COVTEST statement requests a test of homogeneity—that is, it tests whether varying the covariance parameters by the group effect provides a significantly better fit compared to a model in which different groups share the same parameter.

Output 52.9.2 Fit Statistics for Analysis of Fisher's Iris Data

The GLIMMIX Procedure

Fit Statistics	
-2 Res Log Likelihood	2812.89
AIC (smaller is better)	2872.89
AICC (smaller is better)	2876.23
BIC (smaller is better)	2963.21
CAIC (smaller is better)	2993.21
HQIC (smaller is better)	2909.58
Generalized Chi-Square	588.00
Gener. Chi-Square / DF	1.00

The "Fit Statistics" table shows the –2 restricted (residual) log likelihood in the full model and other fit statistics (Output 52.9.2). The "-2 Res Log Likelihood" sets the benchmark against which a model with homogeneity constraint is compared. Output 52.9.3 displays the 30 covariance parameters in this model.

There appear to be substantial differences among the covariance parameters from different groups. For example, the residual variability of the petal length of the three species is 12.4249, 26.6433, and 40.4343, respectively. The homogeneity hypothesis restricts these variances to be equal and similarly for the other covariance parameters. The results from the COVTEST statement are shown in Output 52.9.4.

Output 52.9.3 Covariance Parameters Varied by Species (TYPE=UN)

Covariance Parameter Estimates

Cov Parm	Subject	Group	Estimate	Standard Error
UN(1,1)	id	Species Setosa	12.4249	2.5102
UN(2,1)	id	Species Setosa	9.9216	2.3775
UN(2,2)	id	Species Setosa	14.3690	2.9030
UN(3,1)	id	Species Setosa	1.6355	0.9052
UN(3,2)	id	Species Setosa	1.1698	0.9552
UN(3,3)	id	Species Setosa	3.0159	0.6093
UN(4,1)	id	Species Setosa	1.0331	0.5508
UN(4,2)	id	Species Setosa	0.9298	0.5859
UN(4,3)	id	Species Setosa	0.6069	0.2755
UN(4,4)	id	Species Setosa	1.1106	0.2244
UN(1,1)	id	Species Versicolor	26.6433	5.3828
UN(2,1)	id	Species Versicolor	8.5184	2.6144
UN(2,2)	id	Species Versicolor	9.8469	1.9894
UN(3,1)	id	Species Versicolor	18.2898	4.3398
UN(3,2)	id	Species Versicolor	8.2653	2.4149
UN(3,3)	id	Species Versicolor	22.0816	4.4612
UN(4,1)	id	Species Versicolor	5.5780	1.6617
UN(4,2)	id	Species Versicolor	4.1204	1.0641
UN(4,3)	id	Species Versicolor	7.3102	1.6891
UN(4,4)	id	Species Versicolor	3.9106	0.7901
UN(1,1)	id	Species Virginica	40.4343	8.1690
UN(2,1)	id	Species Virginica	9.3763	3.2213
UN(2,2)	id	Species Virginica	10.4004	2.1012
UN(3,1)	id	Species Virginica	30.3290	6.6262
UN(3,2)	id	Species Virginica	7.1380	2.7395
UN(3,3)	id	Species Virginica	30.4588	6.1536
UN(4,1)	id	Species Virginica	4.9094	2.5916
UN(4,2)	id	Species Virginica	4.7629	1.4367
UN(4,3)	id	Species Virginica	4.8824	2.2750
UN(4,4)	id	Species Virginica	7.5433	1.5240

Output 52.9.4 Likelihood Ratio Test of Homogeneity

Tests of Covariance Parameters Based on the Restricted Likelihood

Label	DF	-2 Res Log Like	ChiSq	Pr > ChiSq	Note
Homogeneity	20	2959.55	146.66	<.0001	DF

DF: P-value based on a chi-square with DF degrees of freedom.

Denote as $\mathbf{R}_k$ the covariance matrix for species $k = 1, 2, 3$ with elements σ_{ijk}. In processing the COVTEST

hypothesis $H_0\colon \mathbf{R}_1 = \mathbf{R}_2 = \mathbf{R}_3$, the GLIMMIX procedure fits a model that satisfies the constraints

$$\begin{aligned}
\sigma_{111} &= \sigma_{112} = \sigma_{113} \\
\sigma_{211} &= \sigma_{212} = \sigma_{213} \\
\sigma_{231} &= \sigma_{232} = \sigma_{233} \\
&\vdots \\
\sigma_{441} &= \sigma_{442} = \sigma_{443}
\end{aligned}$$

where σ_{ijk} is the covariance between the ith and jth variable for the kth species. The –2 restricted log likelihood of this restricted model is 2959.55 (Output 52.9.4). The change of 146.66 compared to the full model is highly significant. There is sufficient evidence to reject the notion of equal covariance matrices among the three iris species.

Equality of covariance matrices implies equality of correlation matrices, but the reverse is not true. Fewer constraints are needed to equate correlations because the diagonal entries of the covariance matrices are free to vary. In order to test the equality of the correlation matrices among the three species, you can parameterize the unstructured covariance matrix in terms of the correlations and use a COVTEST statement with general contrasts, as shown in the following statements:

```
ods select FitStatistics CovParms CovTests;
proc glimmix data=iris_univ;
   class species var id;
   model response = species*var;
   random _residual_ / type=unr group=species subject=id;
   covtest 'Equal Covariance Matrices'  homogeneity;
   covtest 'Equal Correlation Matrices' general
        0  0  0  0  1  0  0  0  0  0
        0  0  0  0 -1  0  0  0  0  0,
        0  0  0  0  1  0  0  0  0  0
        0  0  0  0  0  0  0  0  0  0
        0  0  0  0 -1  0  0  0  0  0,
        0  0  0  0  0  1  0  0  0  0
        0  0  0  0  0 -1  0  0  0  0,
        0  0  0  0  0  1  0  0  0  0
        0  0  0  0  0  0  0  0  0  0
        0  0  0  0  0 -1  0  0  0  0,
        0  0  0  0  0  0  1  0  0  0
        0  0  0  0  0  0 -1  0  0  0,
        0  0  0  0  0  0  1  0  0  0
        0  0  0  0  0  0  0  0  0  0
        0  0  0  0  0  0 -1  0  0  0,
        0  0  0  0  0  0  0  1  0  0
        0  0  0  0  0  0  0 -1  0  0,
        0  0  0  0  0  0  0  1  0  0
        0  0  0  0  0  0  0  0  0  0
        0  0  0  0  0  0  0 -1  0  0,
        0  0  0  0  0  0  0  0  1  0
        0  0  0  0  0  0  0  0 -1  0,
        0  0  0  0  0  0  0  0  1  0
        0  0  0  0  0  0  0  0  0  0
        0  0  0  0  0  0  0  0  -1 0,
```

```
          0  0  0  0  0  0  0  0  0  1
          0  0  0  0  0  0  0  0  0 -1,
          0  0  0  0  0  0  0  0  0  1
          0  0  0  0  0  0  0  0  0  0
          0  0  0  0  0  0  0  0  0 -1 / estimates;
run;
```

The TYPE=UNR structure is a reparameterization of TYPE=UN. The models provide the same fit, as seen by comparison of the "Fit Statistics" tables in Output 52.9.2 and Output 52.9.5. The covariance parameters are ordered differently, however. In each group, the four variances precede the six correlations (Output 52.9.5). The first COVTEST statement tests the homogeneity hypothesis in terms of the UNR parameterization, and the result is identical to the test in Output 52.9.4. The second COVTEST statement restricts the correlations to be equal across groups. If ρ_{ijk} is the correlation between the ith and jth variable for the kth species, the 12 restrictions are

$$\rho_{211} = \rho_{212} = \rho_{213}$$
$$\rho_{311} = \rho_{312} = \rho_{313}$$
$$\rho_{321} = \rho_{322} = \rho_{323}$$
$$\rho_{411} = \rho_{412} = \rho_{413}$$
$$\rho_{421} = \rho_{422} = \rho_{423}$$
$$\rho_{431} = \rho_{432} = \rho_{433}$$

The ESTIMATES option in the COVTEST statement requests that the GLIMMIX procedure display the covariance parameter estimates in the restricted model (Output 52.9.5).

Output 52.9.5 Fit Statistics, Covariance Parameters (TYPE=UNR), and Likelihood Ratio Tests for Equality of Covariance and Correlation Matrices

The GLIMMIX Procedure

Fit Statistics	
-2 Res Log Likelihood	2812.89
AIC (smaller is better)	2872.89
AICC (smaller is better)	2876.23
BIC (smaller is better)	2963.21
CAIC (smaller is better)	2993.21
HQIC (smaller is better)	2909.58
Generalized Chi-Square	588.00
Gener. Chi-Square / DF	1.00

Output 52.9.5 *continued*

Covariance Parameter Estimates

Cov Parm	Subject	Group	Estimate	Standard Error
Var(1)	id	Species Setosa	12.4249	2.5102
Var(2)	id	Species Setosa	14.3690	2.9030
Var(3)	id	Species Setosa	3.0159	0.6093
Var(4)	id	Species Setosa	1.1106	0.2244
Corr(2,1)	id	Species Setosa	0.7425	0.06409
Corr(3,1)	id	Species Setosa	0.2672	0.1327
Corr(3,2)	id	Species Setosa	0.1777	0.1383
Corr(4,1)	id	Species Setosa	0.2781	0.1318
Corr(4,2)	id	Species Setosa	0.2328	0.1351
Corr(4,3)	id	Species Setosa	0.3316	0.1271
Var(1)	id	Species Versicolor	26.6433	5.3828
Var(2)	id	Species Versicolor	9.8469	1.9894
Var(3)	id	Species Versicolor	22.0816	4.4612
Var(4)	id	Species Versicolor	3.9106	0.7901
Corr(2,1)	id	Species Versicolor	0.5259	0.1033
Corr(3,1)	id	Species Versicolor	0.7540	0.06163
Corr(3,2)	id	Species Versicolor	0.5605	0.09797
Corr(4,1)	id	Species Versicolor	0.5465	0.1002
Corr(4,2)	id	Species Versicolor	0.6640	0.07987
Corr(4,3)	id	Species Versicolor	0.7867	0.05445
Var(1)	id	Species Virginica	40.4343	8.1690
Var(2)	id	Species Virginica	10.4004	2.1012
Var(3)	id	Species Virginica	30.4588	6.1536
Var(4)	id	Species Virginica	7.5433	1.5240
Corr(2,1)	id	Species Virginica	0.4572	0.1130
Corr(3,1)	id	Species Virginica	0.8642	0.03616
Corr(3,2)	id	Species Virginica	0.4010	0.1199
Corr(4,1)	id	Species Virginica	0.2811	0.1316
Corr(4,2)	id	Species Virginica	0.5377	0.1015
Corr(4,3)	id	Species Virginica	0.3221	0.1280

Output 52.9.5 *continued*

Tests of Covariance Parameters
Based on the Restricted Likelihood

					Estimates H0									
Label	**DF**	**-2 Res Log Like**	**ChiSq**	**Pr > ChiSq**	**Est1**	**Est2**	**Est3**	**Est4**	**Est5**	**Est6**	**Est7**	**Est8**	**Est9**	**Est10**
Equal Covariance Matrices	20	2959.55	146.66	<.0001	26.5004	11.5395	18.5179	4.1883	0.5302	0.7562	0.3779	0.3645	0.4705	0.4845
Equal Correlation Matrices	12	2876.38	63.49	<.0001	16.4715	14.8656	4.8427	1.4392	0.5612	0.6827	0.4016	0.3844	0.4976	0.5219

Tests of Covariance Parameters
Based on the Restricted Likelihood

	Estimates H0													
Label	**Est11**	**Est12**	**Est13**	**Est14**	**Est15**	**Est16**	**Est17**	**Est18**	**Est19**	**Est20**	**Est21**	**Est22**	**Est23**	**Est24**
Equal Covariance Matrices	26.5004	11.5395	18.5179	4.1883	0.5302	0.7562	0.3779	0.3645	0.4705	0.4845	26.5004	11.5395	18.5179	4.1883
Equal Correlation Matrices	24.4020	9.1566	17.4434	3.0021	0.5612	0.6827	0.4016	0.3844	0.4976	0.5219	35.0544	10.8350	27.3593	8.1395

Tests of Covariance Parameters
Based on the Restricted Likelihood

	Estimates H0						
Label	**Est25**	**Est26**	**Est27**	**Est28**	**Est29**	**Est30**	**Note**
Equal Covariance Matrices	0.5302	0.7562	0.3779	0.3645	0.4705	0.4845	DF
Equal Correlation Matrices	0.5612	0.6827	0.4016	0.3844	0.4976	0.5219	DF

DF: P-value based on a chi-square with DF degrees of freedom.

The result of the homogeneity test is identical to that in Output 52.9.4. The hypothesis of equality of the correlation matrices is also rejected with a chi-square value of 63.49 and a p-value of < 0.0001. Notice, however, that the chi-square statistic is smaller than in the test of homogeneity due to the smaller number of restrictions imposed on the full model. The estimate of the common correlation matrix in the restricted model is

$$\begin{bmatrix} 1 & 0.561 & 0.683 & 0.384 \\ 0.561 & 1 & 0.402 & 0.498 \\ 0.683 & 0.402 & 1 & 0.522 \\ 0.384 & 0.498 & 0.522 & 1 \end{bmatrix}$$

Example 52.10: Multiple Trends Correspond to Multiple Extrema in Profile Likelihoods

(View the complete code for this example (gmxex10.sas) in the example repository.)

Observations for a period of 168 months for the "Southern Oscillation Index," measurements of monthly averaged atmospheric pressure differences between Easter Island and Darwin, Australia (Kahaner, Moler, and Nash 1989, Ch. 11.9; National Institute of Standards and Technology 1998) is available in the data set ENSO in the Sashelp library. These data are also used as an example in Chapter 78, "The LOESS Procedure." The following statements print the first 10 observations of this data set in Output 52.10.1.

```
proc print data=Sashelp.enso(obs=10);
run;
```

Output 52.10.1 El Niño Southern Oscillation Data

Obs	Month	Year	Pressure
1	1	0.08333	12.9
2	2	0.16667	11.3
3	3	0.25000	10.6
4	4	0.33333	11.2
5	5	0.41667	10.9
6	6	0.50000	7.5
7	7	0.58333	7.7
8	8	0.66667	11.7
9	9	0.75000	12.9
10	10	0.83333	14.3

Differences in atmospheric pressure create wind, and the differences recorded in the data set ENSO drive the trade winds in the southern hemisphere. Such time series often do not consist of a single trend or cycle. In this particular case, there are at least two known cycles that reflect the annual weather pattern and a longer cycle that represents the periodic warming of the Pacific Ocean (El Niño).

To estimate the trend in these data by using mixed model technology, you can apply a mixed model smoothing technique such as TYPE=RSMOOTH or TYPE=PSPLINE. The following statements fit a radial smoother to the ENSO data and obtain profile likelihoods for a series of values for the variance of the random spline coefficients:

```
data tdata;
   do covp1=0,0.0005,0.05,0.1,0.2,0.5,
            1,2,3,4,5,6,8,10,15,20,50,
            75,100,125,140,150,160,175,
            200,225,250,275,300,350;
      output;
   end;
run;

ods select FitStatistics CovParms CovTests;
proc glimmix data=sashelp.enso noprofile;
   model pressure = year;
   random year / type=rsmooth knotmethod=equal(50);
```

```
    parms (2) (10);
    covtest tdata=tdata / parms;
    ods output covtests=ct;
run;
```

The tdata data set contains value for the variance of the radial smoother variance for which the profile likelihood of the model is to be computed. The profile likelihood is obtained by setting the radial smoother variance at the specified value and estimating all other parameters subject to that constraint.

Because the model contains a residual variance and you need to specify nonzero values for the first covariance parameter, the NOPROFILE is added to the PROC GLIMMIX statements. If the residual variance is profiled from the estimation, you cannot fix covariance parameters at a given value, because they would be reexpressed during model fitting in terms of ratios with the profiled (and changing) variance.

The PARMS statement determines starting values for the covariance parameters for fitting the (full) model. The PARMS option in the COVTEST statement requests that the input parameters be added to the output and the output data set. This is useful for subsequent plotting of the profile likelihood function.

The "Fit Statistics" table displays the –2 restricted log likelihood of the model (897.76, Output 52.10.2). The estimate of the variance of the radial smoother coefficients is 3.5719.

The "Test of Covariance Parameters" table displays the –2 restricted log likelihood for each observation in the tdata set. Because the tdata data set specifies values for only the first covariance parameter, the second covariance parameter is free to vary and the values for –2 Res Log Like are profile likelihoods. Notice that for a number of values of CovP1 the chi-square statistic is missing in this table. For these values the –2 Res Log Like is *smaller* than that of the full model. The model did not converge to a global minimum of the negative restricted log likelihood.

Output 52.10.2 REML and Profile Likelihood Analysis

The GLIMMIX Procedure

Fit Statistics	
-2 Res Log Likelihood	897.76
AIC (smaller is better)	901.76
AICC (smaller is better)	901.83
BIC (smaller is better)	897.76
CAIC (smaller is better)	899.76
HQIC (smaller is better)	897.76
Generalized Chi-Square	1554.38
Gener. Chi-Square / DF	9.36
Radial Smoother df(res)	153.52

Covariance Parameter Estimates		
Cov Parm	Estimate	Standard Error
Var[RSmooth(Year)]	3.5719	3.7672
Residual	9.3638	1.3014

Output 52.10.2 *continued*

Tests of Covariance Parameters Based on the Restricted Likelihood

Label	DF	-2 Res Log Like	ChiSq	Pr > ChiSq	Input Parameters CovP1	CovP2	Note
WORK.TDATA	1	893.01	.	1.0000	0	9.3638	MI
WORK.TDATA	1	892.76	.	1.0000	0.000500	9.3638	DF
WORK.TDATA	1	897.34	.	1.0000	0.05000	9.3638	DF
WORK.TDATA	1	898.53	0.77	0.3816	0.1000	9.3638	DF
WORK.TDATA	1	899.38	1.62	0.2038	0.2000	9.3638	DF
WORK.TDATA	1	899.49	1.73	0.1888	0.5000	9.3638	DF
WORK.TDATA	1	898.83	1.07	0.3016	1.0000	9.3638	DF
WORK.TDATA	1	898.04	0.28	0.5967	2.0000	9.3638	DF
WORK.TDATA	1	897.79	0.03	0.8693	3.0000	9.3638	DF
WORK.TDATA	1	897.77	0.01	0.9145	4.0000	9.3638	DF
WORK.TDATA	1	897.86	0.10	0.7517	5.0000	9.3638	DF
WORK.TDATA	1	897.99	0.23	0.6311	6.0000	9.3638	DF
WORK.TDATA	1	898.27	0.51	0.4761	8.0000	9.3638	DF
WORK.TDATA	1	898.49	0.73	0.3919	10.0000	9.3638	DF
WORK.TDATA	1	898.70	0.94	0.3318	15.0000	9.3638	DF
WORK.TDATA	1	898.45	0.69	0.4068	20.0000	9.3638	DF
WORK.TDATA	1	892.63	.	1.0000	50.0000	9.3638	DF
WORK.TDATA	1	887.44	.	1.0000	75.0000	9.3638	DF
WORK.TDATA	1	883.79	.	1.0000	100.00	9.3638	DF
WORK.TDATA	1	881.55	.	1.0000	125.00	9.3638	DF
WORK.TDATA	1	880.72	.	1.0000	140.00	9.3638	DF
WORK.TDATA	1	880.34	.	1.0000	150.00	9.3638	DF
WORK.TDATA	1	880.07	.	1.0000	160.00	9.3638	DF
WORK.TDATA	1	879.85	.	1.0000	175.00	9.3638	DF
WORK.TDATA	1	879.86	.	1.0000	200.00	9.3638	DF
WORK.TDATA	1	880.21	.	1.0000	225.00	9.3638	DF
WORK.TDATA	1	880.80	.	1.0000	250.00	9.3638	DF
WORK.TDATA	1	881.56	.	1.0000	275.00	9.3638	DF
WORK.TDATA	1	882.44	.	1.0000	300.00	9.3638	DF
WORK.TDATA	1	884.41	.	1.0000	350.00	9.3638	DF

DF: P-value based on a chi-square with DF degrees of freedom.
MI: P-value based on a mixture of chi-squares.

The following statements plot the –2 restricted profile log likelihood (Output 52.10.3):

```
proc sgplot data=ct;
   series y=objective x=covp1;
run;
```

Output 52.10.3 –2 Restricted Profile Log Likelihood for Smoothing Variance

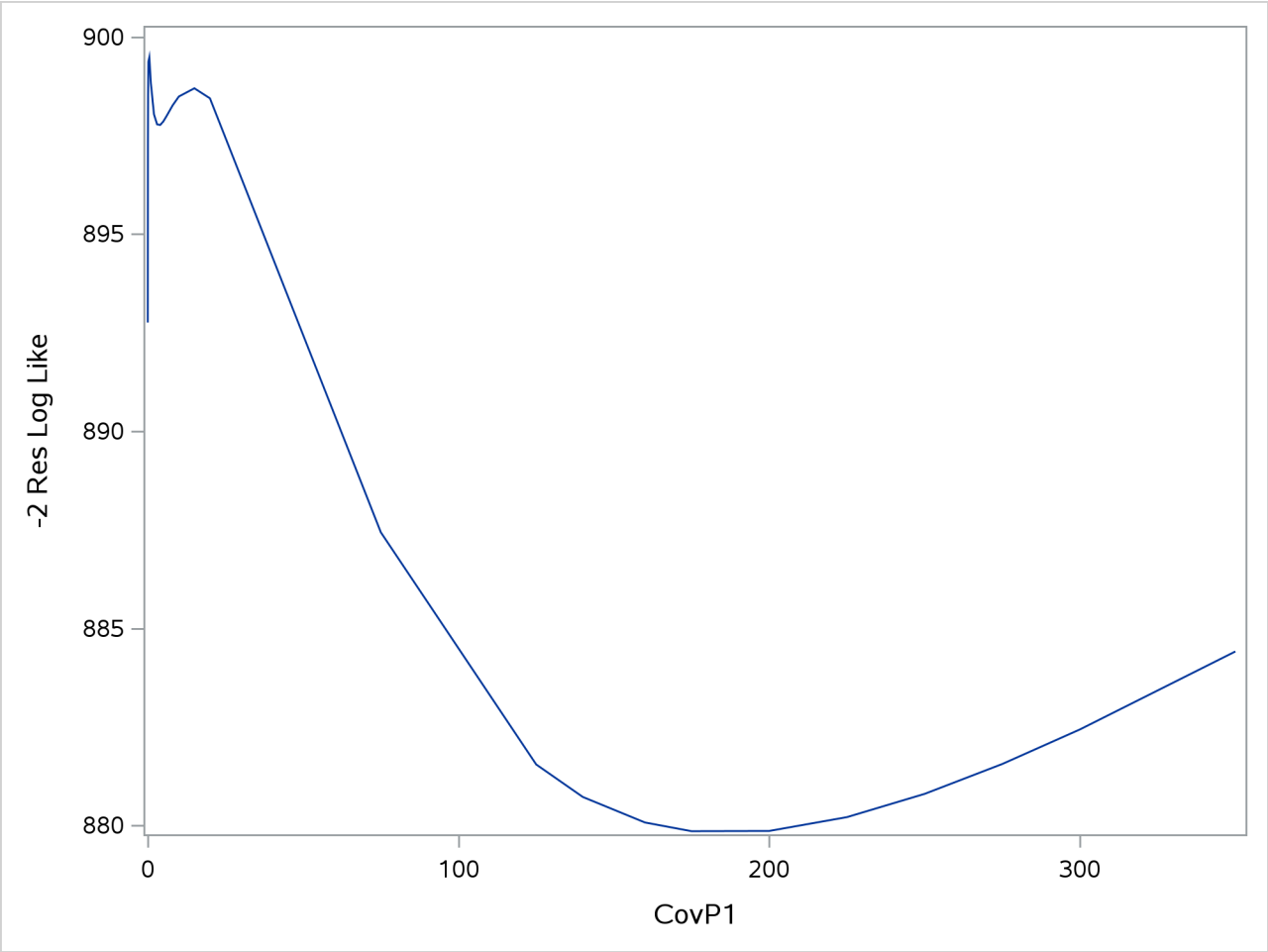

The local minimum at which the optimization stopped is clearly visible, as are a second local minimum near zero and the global minimum near 180.

The observed and predicted pressure differences that correspond to the three minima are shown in Output 52.10.4. These results were produced with the following statements:

```
proc glimmix data=sashelp.enso;
   model pressure = year;
   random year / type=rsmooth knotmethod=equal(50);
   parms (0) (10);
   output out=gmxout1 pred=pred1;
run;
proc glimmix data=sashelp.enso;
   model pressure = year;
   random year / type=rsmooth knotmethod=equal(50);
   output out=gmxout2 pred=pred2;
   parms (2) (10);
run;
proc glimmix data=sashelp.enso;
   model pressure = year;
   random year / type=rsmooth knotmethod=equal(50);
```

```
   output out=gmxout3 pred=pred3;
   parms (200) (10);
run;
data plotthis; merge gmxout1 gmxout2 gmxout3;
run;
proc sgplot data=plotthis;
   scatter x=year y=Pressure;
   series  x=year y=pred1 /
        lineattrs   = (pattern=solid thickness=2)
        legendlabel = "Var[RSmooth] = 0.0005"
        name        = "pred1";
   series  x=year y=pred2 /
        lineattrs   = (pattern=dot thickness=2)
        legendlabel = "Var[RSmooth] = 3.5719"
        name        = "pred2";
   series  x=year y=pred3 /
        lineattrs   = (pattern=dash thickness=2)
        legendlabel = "Var[RSmooth] = 186.71"
        name        = "pred3";
   keylegend "pred1" "pred2" "pred3" / across=2;
run;
```

Output 52.10.4 Observed and Predicted Pressure Differences

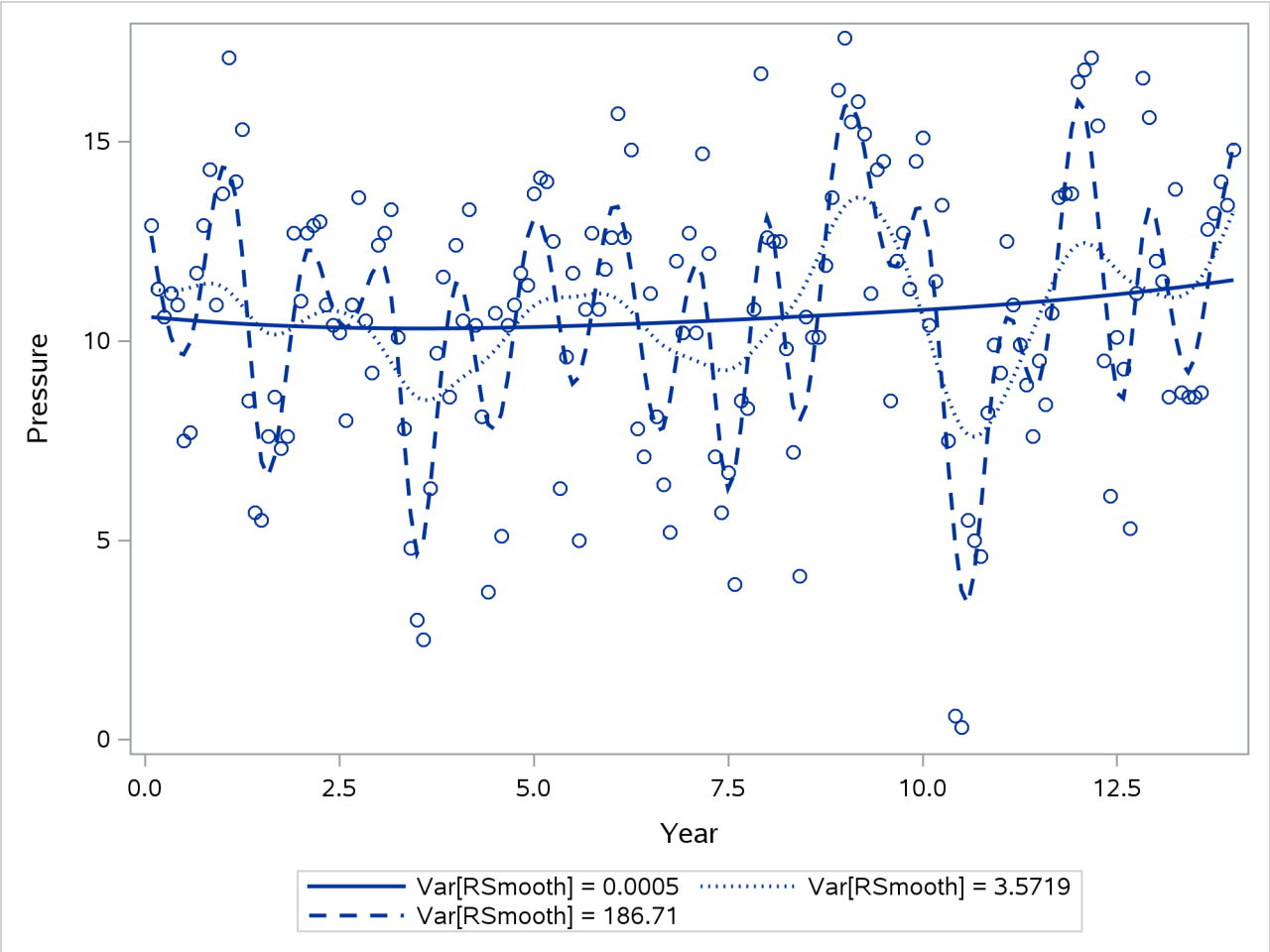

The one-year cycle ($\widehat{\sigma}_r^2 = 186.71$) and the El Niño cycle ($\widehat{\sigma}_r^2 = 3.5719$) are clearly visible. Notice that a larger smoother variance results in larger BLUPs and hence larger adjustments to the fixed-effects model. A large smoother variance thus results in a more wiggly fit. The third local minimum at $\widehat{\sigma}_r^2 = 0.0005$ applies only very small adjustments to the linear regression between pressure and time, creating slight curvature.

Example 52.11: Maximum Likelihood in Proportional Odds Model with Random Effects

(View the complete code for this example (gmxex11.sas) in the example repository.)

The data for this example are taken from Gilmour, Anderson, and Rae (1987) and concern the foot shape of 2,513 lambs that represent 34 sires. The foot shape of the animals was scored in three ordered categories. The following DATA step lists the data in multivariate form, where each observation corresponds to a sire and contains the outcomes for the three response categories in the variables k1, k2, and k3. For example, for the first sire the first foot shape category was observed for 52 of its offspring, foot shape category 2 was observed for 25 lambs, and none of its offspring was rated in foot shape category 3. The variables yr, b1, b2, and b3 represent contrasts of fixed effects.

```
data foot_mv;
   input yr b1 b2 b3 k1 k2 k3;
   sire = _n_;
   datalines;
  1  1  0  0  52 25  0
  1  1  0  0  49 17  1
  1  1  0  0  50 13  1
  1  1  0  0  42  9  0
  1  1  0  0  74 15  0
  1  1  0  0  54  8  0
  1  1  0  0  96 12  0
  1 -1  1  0  57 52  9
  1 -1  1  0  55 27  5
  1 -1  1  0  70 36  4
  1 -1  1  0  70 37  3
  1 -1  1  0  82 21  1
  1 -1  1  0  75 19  0
  1 -1 -1  0  17 12 10
  1 -1 -1  0  13 23  3
  1 -1 -1  0  21 17  3
 -1  0  0  1  37 41 23
 -1  0  0  1  47 24 12
 -1  0  0  1  46 25  9
 -1  0  0  1  79 32 11
 -1  0  0  1  50 23  5
 -1  0  0  1  63 18  8
 -1  0  0 -1  30 20  9
 -1  0  0 -1  31 33  3
 -1  0  0 -1  28 18  4
 -1  0  0 -1  42 27  4
 -1  0  0 -1  35 22  2
 -1  0  0 -1  33 18  3
 -1  0  0 -1  35 17  4
```

```
 -1  0  0 -1  26 13  2
 -1  0  0 -1  37 15  2
 -1  0  0 -1  36 14  1
 -1  0  0 -1  63 20  3
 -1  0  0 -1  41  8  1
;
```

In order to analyze these data as multinomial data with PROC GLIMMIX, the data need to be arranged in univariate form. The following DATA step creates three observations from each record in data set foot_mv and stores the category counts in the variable count:

```
data footshape; set foot_mv;
   array k{3};
   do Shape = 1 to 3;
      count = k{Shape};
      output;
   end;
   drop k:;
run;
```

Because the sires were selected at random, a model for the three-category response with fixed regression effects for yr, b1–b3, and with random sire effects is considered. Because the response categories are ordered, a proportional odds model is chosen (McCullagh 1980). Gilmour, Anderson, and Rae (1987) consider various analyses for these data. The following GLIMMIX statements fit a model with probit link for the cumulative probabilities by maximum likelihood where the marginal log likelihood is approximated by adaptive quadrature:

```
proc glimmix data=footshape method=quad;
   class sire;
   model Shape = yr b1 b2 b3 / s link=cumprobit dist=multinomial;
   random int / sub=sire s cl;
   ods output Solutionr=solr;
   freq count;
run;
```

The number of observations that share a particular response and covariate pattern (variable count) is used in the FREQ statement. The S and CL options request solutions for the sire effects. These are output to the data set solr for plotting.

The “Model Information” table shows that the parameters are estimated by maximum likelihood and that the marginal likelihood is approximated by Gauss-Hermite quadrature (Output 52.11.1).

Output 52.11.1 Model and Data Information

The GLIMMIX Procedure

Model Information	
Data Set	WORK.FOOTSHAPE
Response Variable	Shape
Response Distribution	Multinomial (ordered)
Link Function	Cumulative Probit
Variance Function	Default
Frequency Variable	count
Variance Matrix Blocked By	sire
Estimation Technique	Maximum Likelihood
Likelihood Approximation	Gauss-Hermite Quadrature
Degrees of Freedom Method	Containment

Number of Observations Read	102
Number of Observations Used	96
Sum of Frequencies Read	2513
Sum of Frequencies Used	2513

Response Profile

Ordered Value	Shape	Total Frequency
1	1	1636
2	2	731
3	3	146

The GLIMMIX procedure is modeling the probabilities of levels of Shape having lower Ordered Values in the Response Profile table.

The distribution of the data is multinomial with ordered categories. The ordering is implied by the choice of a link function for the cumulative probabilities. Because a frequency variable is specified, the number of observations as well as the number of frequencies is displayed. Observations with zero frequency—that is, foot shape categories that were not observed for a particular sire are not used in the analysis. The "Response Profile Table" shows the ordering of the response variable and gives a breakdown of the frequencies by category.

Output 52.11.2 Information about the Size of the Optimization Problem

Dimensions	
G-side Cov. Parameters	1
Columns in X	6
Columns in Z per Subject	1
Subjects (Blocks in V)	34
Max Obs per Subject	3

Output 52.11.2 *continued*

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	7
Lower Boundaries	1
Upper Boundaries	0
Fixed Effects	Not Profiled
Starting From	GLM estimates
Quadrature Points	1

With METHOD=QUAD, the "Dimensions" and "Optimization Information" tables are particularly important, because for this estimation methods both fixed effects and covariance parameters participate in the optimization (Output 52.11.2). For GLM models the optimization involves the fixed effects and possibly a single scale parameter. For mixed models the fixed effects are typically profiled from the optimization. Laplace and quadrature estimations are exceptions to these rules. Consequently, there are seven parameters in this optimization, corresponding to six fixed effects and one variance component. The variance component has a lower bound of 0. Also, because the fixed effects are part of the optimizations, PROC GLIMMIX initially performs a few GLM iterations to obtain starting values for the fixed effects. You can control the number of initial iterations with the INITITER= option in the PROC GLIMMIX statement.

The last entry in the "Optimization Information" table shows that—at the starting values—PROC GLIMMIX determined that a single quadrature point is sufficient to approximate the marginal log likelihood with the required accuracy. This approximation is thus identical to the Laplace method that is available with METHOD=LAPLACE.

For METHOD=LAPLACE and METHOD=QUAD, the GLIMMIX procedure produces fit statistics based on the conditional and marginal distribution (Output 52.11.3). Within the limits of the numeric likelihood approximation, the information criteria shown in the "Fit Statistics" table can be used to compare models, and the –2 log likelihood can be used to compare among nested models (nested with respect to fixed effects and/or the covariance parameters).

Output 52.11.3 Marginal and Conditional Fit Statistics

Fit Statistics	
-2 Log Likelihood	3870.12
AIC (smaller is better)	3884.12
AICC (smaller is better)	3884.17
BIC (smaller is better)	3894.81
CAIC (smaller is better)	3901.81
HQIC (smaller is better)	3887.76

Fit Statistics for Conditional Distribution	
-2 log L(Shape | r. effects)	3807.62

The variance of the sire effect is estimated as 0.04849 with estimated asymptotic standard error of 0.01673 (Output 52.11.4). Based on the magnitude of the estimate relative to the standard error, one might conclude that there is significant sire-to-sire variability. Because parameter estimation is based on maximum likelihood, a formal test of the hypothesis of no sire variability is possible. The category cutoffs for the cumulative

probabilities are 0.3781 and 1.6435. Except for b3, all fixed effects contrasts are significant.

Output 52.11.4 Parameter Estimates

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	sire	0.04849	0.01673

Solutions for Fixed Effects

Effect	Shape	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	1	0.3781	0.04907	29	7.71	<.0001
Intercept	2	1.6435	0.05930	29	27.72	<.0001
yr		0.1422	0.04834	2478	2.94	0.0033
b1		0.3781	0.07154	2478	5.28	<.0001
b2		0.3157	0.09709	2478	3.25	0.0012
b3		-0.09887	0.06508	2478	-1.52	0.1289

A likelihood ratio test for the sire variability can be carried out by adding a COVTEST statement to the PROC GLIMMIX statements (Output 52.11.5):

```
ods select FitStatistics CovParms Covtests;
proc glimmix data=footshape method=quad;
   class sire;
   model Shape = yr b1 b2 b3 / link=cumprobit dist=multinomial;
   random int / sub=sire;
   covtest GLM;
   freq count;
run;
```

The statement

```
covtest GLM;
```

compares the fitted model to a generalized linear model for independent data by removing the sire variance component from the model. Equivalently, you can specify

```
covtest 0;
```

which compares the fitted model against one where the sire variance is fixed at zero.

Output 52.11.5 Likelihood Ratio Test for Sire Variance

The GLIMMIX Procedure

Fit Statistics	
-2 Log Likelihood	3870.12
AIC (smaller is better)	3884.12
AICC (smaller is better)	3884.17
BIC (smaller is better)	3894.81
CAIC (smaller is better)	3901.81
HQIC (smaller is better)	3887.76

Output 52.11.5 *continued*

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	sire	0.04849	0.01673

Tests of Covariance Parameters Based on the Likelihood

Label	DF	-2 Log Like	ChiSq	Pr > ChiSq	Note
Independence	1	3915.29	45.17	<.0001	MI

MI: P-value based on a mixture of chi-squares.

The –2 Log Likelihood in the reduced model without the sire effect is 3915.29. Compared to the corresponding marginal fit statistic in the full model (3870.12), this results in a chi-square statistic of 45.17. Because the variance component for the sire effect has a natural lower bound of zero, PROC GLIMMIX performs the likelihood ratio test as a one-sided test. As indicated by the note, the *p*-value for this test is computed from a mixture of chi-square distributions, applying the results of Self and Liang (1987). There is significant evidence that the model without sire random effects does not fit the data as well.

In studies of heritability, one is often interested to rank individuals according to some measure of "breeding value." The following statements display the empirical Bayes estimates of the sire effects from ML estimation by quadrature along with prediction standard error bars (Output 52.11.6):

```
proc sort data=solr;
   by Estimate;
run;
data solr; set solr;
   length sire $2;
   obs  = _n_;
   sire = left(substr(Subject,6,2));
run;
proc sgplot data=solr;
   scatter x=obs y=estimate /
         markerchar  = sire
         yerrorupper = upper
         yerrorlower = lower;
   xaxis grid label='Sire Rank' values=(1 5 10 15 20 25 30);
   yaxis grid label='Predicted Sire Effect';
run;
```

Output 52.11.6 Ranked Predicted Sire Effects and Prediction Standard Errors

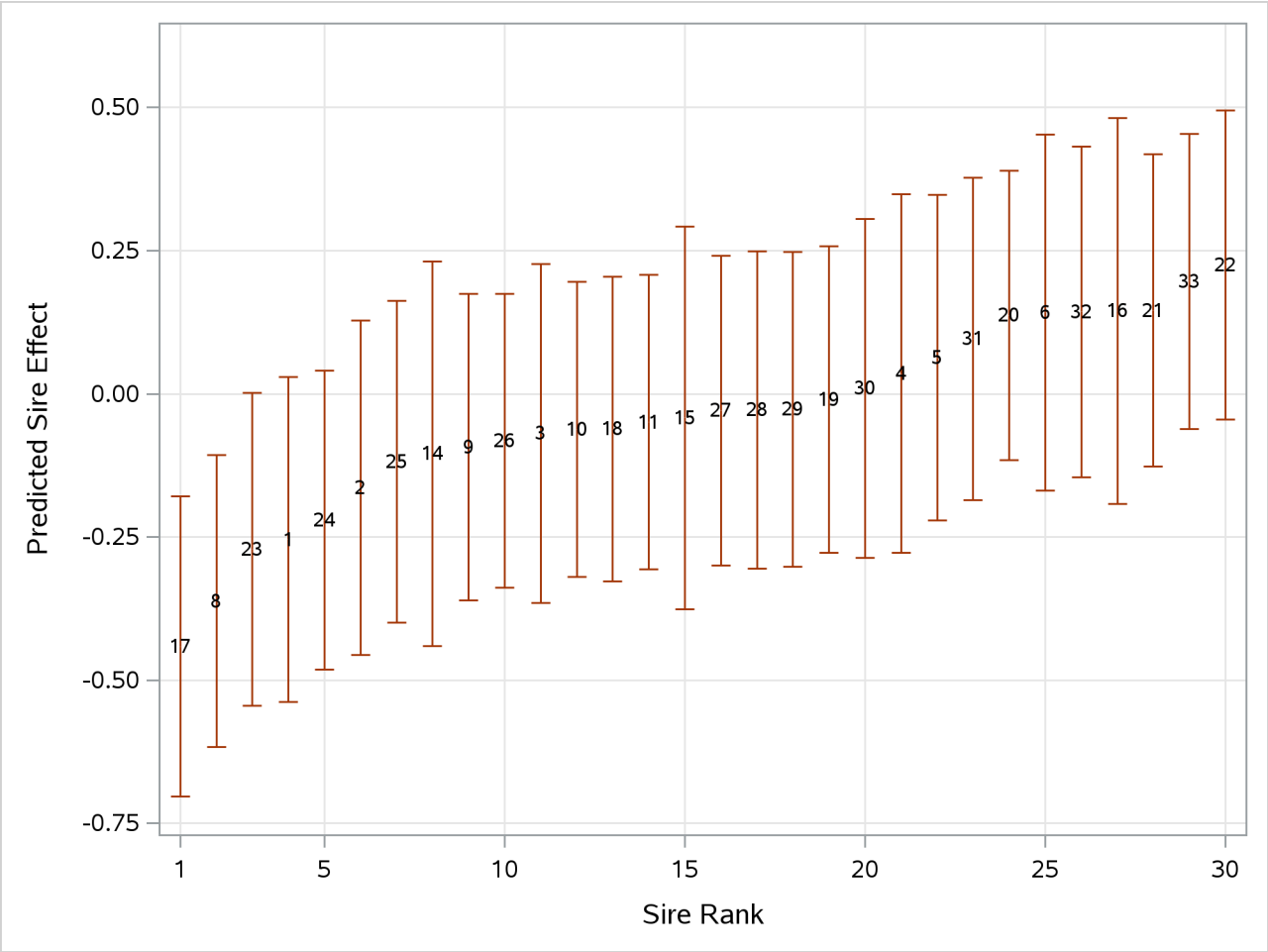

Example 52.12: Fitting a Marginal (GEE-Type) Model

(View the complete code for this example (gmxex12.sas) in the example repository.)

A marginal GEE-type model for clustered data is a model for correlated data that is specified through a mean function, a variance function, and a "working" covariance structure. Because the assumed covariance structure can be wrong, the covariance matrix of the parameter estimates is not based on the model alone. Rather, one of the empirical ("sandwich") estimators is used to make inferences robust against the choice of working covariance structure. PROC GLIMMIX can fit marginal models by using R-side random effects and drawing on the distributional specification in the MODEL statement to derive the link and variance functions. The EMPIRICAL= option in the PROC GLIMMIX statement enables you to choose one of a number of empirical covariance estimators.

The data for this example are from Thall and Vail (1990) and reflect the number of seizures of patients suffering from epileptic episodes. After an eight-week period without treatment, patients were observed four times in two-week intervals during which they received a placebo or the drug Progabide in addition to other therapy. These data are also analyzed in Example 51.7 of Chapter 51, "The GENMOD Procedure." The following DATA step creates the data set seizures. The variable id identifies the subjects in the study, and the

variable trt identifies whether a subject received the placebo (trt = 0) or the drug Progabide (trt = 1). The variable x1 takes on value 0 for the baseline measurement and 1 otherwise.

```
data seizures;
   array c{5};
   input id trt c1-c5;
   do i=1 to 5;
      x1    = (i > 1);
      ltime = (i=1)*log(8) + (i ne 1)*log(2);
      cnt   = c{i};
      output;
   end;
   keep id cnt x1 trt ltime;
   datalines;
101 1  76 11 14  9  8
102 1  38  8  7  9  4
103 1  19  0  4  3  0
104 0  11  5  3  3  3
106 0  11  3  5  3  3
107 0   6  2  4  0  5
108 1  10  3  6  1  3
110 1  19  2  6  7  4
111 1  24  4  3  1  3
112 1  31 22 17 19 16
113 1  14  5  4  7  4
114 0   8  4  4  1  4
116 0  66  7 18  9 21
117 1  11  2  4  0  4
118 0  27  5  2  8  7
121 1  67  3  7  7  7
122 1  41  4 18  2  5
123 0  12  6  4  0  2
124 1   7  2  1  1  0
126 0  52 40 20 23 12
128 1  22  0  2  4  0
129 1  13  5  4  0  3
130 0  23  5  6  6  5
135 0  10 14 13  6  0
137 1  46 11 14 25 15
139 1  36 10  5  3  8
141 0  52 26 12  6 22
143 1  38 19  7  6  7
145 0  33 12  6  8  4
147 1   7  1  1  2  3
201 0  18  4  4  6  2
202 0  42  7  9 12 14
203 1  36  6 10  8  8
204 1  11  2  1  0  0
205 0  87 16 24 10  9
206 0  50 11  0  0  5
208 1  22  4  3  2  4
209 1  41  8  6  5  7
210 0  18  0  0  3  3
211 1  32  1  3  1  5
```

```
213 0 111 37 29 28 29
214 1  56 18 11 28 13
215 0  18  3  5  2  5
217 0  20  3  0  6  7
218 1  24  6  3  4  0
219 0  12  3  4  3  4
220 0   9  3  4  3  4
221 1  16  3  5  4  3
222 0  17  2  3  3  5
225 1  22  1 23 19  8
226 0  28  8 12  2  8
227 0  55 18 24 76 25
228 1  25  2  3  0  1
230 0   9  2  1  2  1
232 1  13  0  0  0  0
234 0  10  3  1  4  2
236 1  12  1  4  3  2
238 0  47 13 15 13 12
;
```

The model fit initially with the following PROC GLIMMIX statements is a Poisson generalized linear model with effects for an intercept, the baseline measurement, the treatment, and their interaction:

```
proc glimmix data=seizures;
   model cnt = x1 trt x1*trt / dist=poisson offset=ltime
                               ddfm=none s;
run;
```

The DDFM=NONE option is chosen in the MODEL statement to produce chi-square and z tests instead of F and t tests.

Because the initial pretreatment time period is four times as long as the subsequent measurement intervals, an offset variable is used to standardize the counts. If Y_{ij} denotes the number of seizures of subject i in time interval j of length t_j, then Y_{ij}/t_j is the number of seizures per time unit. Modeling the average number per time unit with a log link leads to $\log\{\mathrm{E}[Y_{ij}/t_j]\} = \mathbf{x}'\boldsymbol{\beta}$ or $\log\{\mathrm{E}[Y_{ij}]\} = \mathbf{x}'\boldsymbol{\beta} + \log\{t_j\}$. The logarithm of time (variable ltime) thus serves as an offset. Suppose that β_0 denotes the intercept, β_1 the effect of x1, and β_2 the effect of trt. Then $\exp\{\beta_0\}$ is the expected number of seizures per week in the placebo group at baseline. The corresponding numbers in the treatment group are $\exp\{\beta_0 + \beta_2\}$ at baseline and $\exp\{\beta_0 + \beta_1 + \beta_2\}$ for postbaseline visits.

The "Model Information" table shows that the parameters in this Poisson model are estimated by maximum likelihood (Output 52.12.1). In addition to the default link and variance function, the variable ltime is used as an offset.

Output 52.12.1 Model Information in Poisson GLM

The GLIMMIX Procedure

Model Information	
Data Set	WORK.SEIZURES
Response Variable	cnt
Response Distribution	Poisson
Link Function	Log
Variance Function	Default
Offset Variable	ltime
Variance Matrix	Diagonal
Estimation Technique	Maximum Likelihood
Degrees of Freedom Method	None

Fit statistics and parameter estimates are shown in Output 52.12.2.

Output 52.12.2 Results from Fitting Poisson GLM

Fit Statistics	
-2 Log Likelihood	3442.66
AIC (smaller is better)	3450.66
AICC (smaller is better)	3450.80
BIC (smaller is better)	3465.34
CAIC (smaller is better)	3469.34
HQIC (smaller is better)	3456.54
Pearson Chi-Square	3015.16
Pearson Chi-Square / DF	10.54

Parameter Estimates

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	1.3476	0.03406	Infty	39.57	<.0001
x1	0.1108	0.04689	Infty	2.36	0.0181
trt	-0.1080	0.04865	Infty	-2.22	0.0264
x1*trt	-0.3016	0.06975	Infty	-4.32	<.0001

Because this is a generalized linear model, the large value for the ratio of the Pearson chi-square statistic and its degrees of freedom is indicative of a model shortcoming. The data are considerably more dispersed than is expected under a Poisson model. There could be many reasons for this overdispersion—for example, a misspecified mean model, data that might not be Poisson distributed, an incorrect variance function, and correlations among the observations. Because these data are repeated measurements, the presence of correlations among the observations from the same subject is a likely contributor to the overdispersion.

The following PROC GLIMMIX statements fit a marginal model with correlations. The model is a marginal one, because no G-side random effects are specified on which the distribution could be conditioned. The choice of the id variable as the SUBJECT effect indicates that observations from different IDs are uncorrelated. Observations from the same ID are assumed to follow a compound symmetry (equicorrelation) model. The EMPIRICAL option in the PROC GLIMMIX statement requests the classical sandwich estimator as the covariance estimator for the fixed effects:

```
proc glimmix data=seizures empirical;
   class id;
   model cnt = x1 trt x1*trt / dist=poisson offset=ltime
                               ddfm=none covb s;
   random _residual_ / subject=id type=cs vcorr;
run;
```

The "Model Information" table shows that the parameters are now estimated by residual pseudo-likelihood (compare Output 52.12.3 and Output 52.12.1). And in this fact lies the main difference between fitting marginal models with PROC GLIMMIX and with GEE methods as per Liang and Zeger (1986), where parameters of the working correlation matrix are estimated by the method of moments.

Output 52.12.3 Model Information in Marginal Model

The GLIMMIX Procedure

Model Information	
Data Set	WORK.SEIZURES
Response Variable	cnt
Response Distribution	Poisson
Link Function	Log
Variance Function	Default
Offset Variable	ltime
Variance Matrix Blocked By	id
Estimation Technique	Residual PL
Degrees of Freedom Method	None
Fixed Effects SE Adjustment	Sandwich - Classical

According to the compound symmetry model, there is substantial correlation among the observations from the same subject (Output 52.12.4).

Output 52.12.4 Covariance Parameter Estimates and Correlation Matrix

Estimated V Correlation Matrix for id 101

Row	Col1	Col2	Col3	Col4	Col5
1	1.0000	0.6055	0.6055	0.6055	0.6055
2	0.6055	1.0000	0.6055	0.6055	0.6055
3	0.6055	0.6055	1.0000	0.6055	0.6055
4	0.6055	0.6055	0.6055	1.0000	0.6055
5	0.6055	0.6055	0.6055	0.6055	1.0000

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
CS	id	6.4653	1.3833
Residual		4.2128	0.3928

The parameter estimates in Output 52.12.5 are the same as in the Poisson generalized linear model (Output 52.12.2), because of the balance in these data. The standard errors have increased substantially, however, by taking into account the correlations among the observations.

Output 52.12.5 GEE-Type Inference for Fixed Effects

Solutions for Fixed Effects					
Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	1.3476	0.1574	Infty	8.56	<.0001
x1	0.1108	0.1161	Infty	0.95	0.3399
trt	-0.1080	0.1937	Infty	-0.56	0.5770
x1*trt	-0.3016	0.1712	Infty	-1.76	0.0781

Empirical Covariance Matrix for Fixed Effects					
Effect	Row	Col1	Col2	Col3	Col4
Intercept	**1**	0.02476	-0.00115	-0.02476	0.001152
x1	**2**	-0.00115	0.01348	0.001152	-0.01348
trt	**3**	-0.02476	0.001152	0.03751	-0.00300
x1*trt	**4**	0.001152	-0.01348	-0.00300	0.02931

Example 52.13: Response Surface Comparisons with Multiplicity Adjustments

(View the complete code for this example (gmxex13.sas) in the example repository.)

Koch et al. (1990) present data for a multicenter clinical trial testing the efficacy of a respiratory drug in patients with respiratory disease. Within each of two centers, patients were randomly assigned to a placebo (P) or an active (A) treatment. Prior to treatment and at four follow-up visits, patient status was recorded in one of five ordered categories (0=terrible, 1=poor, . . . , 4=excellent). The following DATA step creates the SAS data set clinical for this study.

```
data Clinical;
   do Center =   1,  2;
      do Gender = 'F','M';
         do Drug   = 'A','P';
            input nPatient @@;
            do iPatient = 1 to nPatient;
               input ID Age (t0-t4) (1.) @@;
               output;
            end;
         end;
      end;
   end;
   datalines;
 2  53 32 12242  18 47 22344
 5   5 13 44444  19 31 21022  25 35 10000  28 36 23322
    36 45 22221
25  54 11 44442  12 14 23332  51 15 02333  20 20 33231
    16 22 12223  50 22 21344   3 23 33443  32 23 23444
    56 25 23323  35 26 12232  26 26 22222  21 26 24142
     8 28 12212  30 28 00121  33 30 33442  11 30 34443
    42 31 12311   9 31 33444  37 31 02321  23 32 34433
     6 34 11211  22 46 43434  24 48 23202  38 50 22222
```

```
     48 57 33434
24   43 13 34444  41 14 22123  34 15 22332  29 19 23300
     15 20 44444  13 23 33111  27 23 44244  55 24 34443
     17 25 11222  45 26 24243  40 26 12122  44 27 12212
     49 27 33433  39 23 21111   2 28 20000  14 30 10000
     31 37 10000  10 37 32332   7 43 23244  52 43 11132
      4 44 34342   1 46 22222  46 49 22222  47 63 22222
 4   30 37 13444  52 39 23444  23 60 44334  54 63 44444
12   28 31 34444   5 32 32234  21 36 33213  50 38 12000
      1 39 12112  48 39 32300   7 44 34444  38 47 23323
      8 48 22100  11 48 22222   4 51 34244  17 58 14220
23   12 13 44444  10 14 14444  27 19 33233  47 20 24443
     16 20 21100  29 21 33444  20 24 44444  25 25 34331
     15 25 34433   2 25 22444   9 26 23444  49 28 23221
     55 31 44444  43 34 24424  26 35 44444  14 37 43224
     36 41 34434  51 43 33442  37 52 12122  19 55 44444
     32 55 22331   3 58 44444  53 68 23334
16   39 11 34444  40 14 21232  24 15 32233  41 15 43334
     33 19 42233  34 20 32444  13 20 14444  45 33 33323
     22 36 24334  18 38 43000  35 42 32222  44 43 21000
      6 45 34212  46 48 44000  31 52 23434  42 66 33344
;
```

Westfall and Tobias (2007) define as the measure of efficacy the average of the ratings at the final two visits and model this average as a function of drug, baseline assessment score, and age. Hence, in their model, the expected efficacy for drug $d \in A$, P can be written as

$$\mathrm{E}[Y_d] = \beta_{0d} + \beta_{1d} t + \beta_{2d} a$$

where t is the baseline (pretreatment) assessment score and a is the patient's age at baseline. The age range for these data extends from 11 to 68 years. Suppose that the scientific question of interest is the comparison of the two response surfaces at a set of values $S_t \times S_a = \{0, 1, 2, 3, 4\} \times S_a$. In other words, you want to know for which values of the covariates the average response differs significantly between the treatment group and the placebo group. If the set of ages of interest is $\{10, 13, 16, \ldots, 70\}$, then this involves $5 \times 21 = 105$ comparisons, a massive multiple testing problem. The large number of comparisons and the fact that the set S_a is chosen somewhat arbitrarily require the application of multiplicity corrections in order to protect the familywise Type I error across the comparisons.

When testing hypotheses that have logical restrictions, the power of multiplicity corrected tests can be increased by taking the restrictions into account. Logical restrictions exist, for example, when not all hypotheses in a set can be simultaneously true. Westfall and Tobias (2007) extend the truncated closed testing procedure (TCTP) of Royen (1989) for pairwise comparisons in ANOVA to general contrasts. Their work is also an extension of the S2 method of Shaffer (1986); see also Westfall (1997). These methods are all *monotonic* in the (unadjusted) p-values of the individual tests, in the sense that if $p_j < p_i$ then the multiple test will never retain H_j while rejecting H_i. In terms of multiplicity-adjusted p-values $\tilde{p}_j$, monotonicity means that if $p_j < p_i$, then $\tilde{p}_j < \tilde{p}_i$.

Analysis as Normal Data with Averaged Endpoints

In order to apply the extended TCTP procedure of Westfall and Tobias (2007) to the problem of comparing response surfaces in the clinical trial, the following convenience macro is helpful to generate the comparisons for the ESTIMATE statement in PROC GLIMMIX:

```
%macro Contrast(from,to,byA,byT);
   %let nCmp = 0;
   %do age = &from %to &to %by &byA;
      %do t0  =  0 %to  4 %by &byT;
         %let nCmp = %eval(&nCmp+1);
      %end;
   %end;
   %let iCmp = 0;
   %do age = &from %to &to %by &byA;
      %do t0  =  0 %to  4 %by &byT;
         %let iCmp = %eval(&iCmp+1);
         "%trim(%left(&age)) %trim(%left(&t0))"
           drug     1    -1
           drug*age &age -&age
           drug*t0  &t0  -&t0
         %if (&icmp < &nCmp) %then %do; , %end;
      %end;
   %end;
%mend;
```

The following GLIMMIX statements fit the model to the data and compute the 105 contrasts that compare the placebo to the active response at 105 points in the two-dimensional regressor space:

```
proc glimmix data=clinical;
   t = (t3+t4)/2;
   class drug;
   model t = drug t0 age drug*age drug*t0;
   estimate %contrast(10,70,3,1)
              / adjust=simulate(seed=1)
                stepdown(type=logical);
   ods output Estimates=EstStepDown;
run;
```

Note that only a single ESTIMATE statement is used. Each of the 105 comparisons is one comparison in the multirow statement. The ADJUST option in the ESTIMATE statement requests multiplicity-adjusted *p*-values. The extended TCTP method is applied by specifying the STEPDOWN(TYPE=LOGICAL) option to compute step-down-adjusted *p*-values where logical constraints among the hypotheses are taken into account. The results from the ESTIMATE statement are saved to a data set for subsequent processing. Note also that the response, the average of the ratings at the final two visits, is computed with programming statements in PROC GLIMMIX.

The following statements print the 20 most significant estimated differences (Output 52.13.1):

```
proc sort data=EstStepDown;
   by Probt;
run;
proc print data=EstStepDown(obs=20);
   var Label Estimate StdErr Probt AdjP;
```

```
run;
```

Output 52.13.1 The First 20 Observations of the Estimates Data Set

Obs	Label	Estimate	StdErr	Probt	Adjp
1	37 2	0.8310	0.2387	0.0007	0.0071
2	40 2	0.8813	0.2553	0.0008	0.0071
3	34 2	0.7806	0.2312	0.0010	0.0071
4	43 2	0.9316	0.2794	0.0012	0.0071
5	46 2	0.9819	0.3093	0.0020	0.0071
6	31 2	0.7303	0.2338	0.0023	0.0081
7	49 2	1.0322	0.3434	0.0033	0.0107
8	52 2	1.0825	0.3807	0.0054	0.0167
9	40 3	0.7755	0.2756	0.0059	0.0200
10	37 3	0.7252	0.2602	0.0063	0.0201
11	43 3	0.8258	0.2982	0.0066	0.0201
12	28 2	0.6800	0.2461	0.0068	0.0215
13	55 2	1.1329	0.4202	0.0082	0.0239
14	46 3	0.8761	0.3265	0.0085	0.0239
15	34 3	0.6749	0.2532	0.0089	0.0257
16	43 1	1.0374	0.3991	0.0107	0.0329
17	46 1	1.0877	0.4205	0.0111	0.0329
18	49 3	0.9264	0.3591	0.0113	0.0329
19	40 1	0.9871	0.3827	0.0113	0.0329
20	58 2	1.1832	0.4615	0.0118	0.0329

Notice that the adjusted *p*-values (Adjp) are larger than the unadjusted *p*-values, as expected. Also notice that several comparisons share the same adjusted *p*-values. This is a result of the monotonicity of the extended TCTP method.

In order to compare the step-down-adjusted *p*-values to adjusted *p*-values that do not use step-down methods, replace the ESTIMATE statement in the previous statements with the following:

```
estimate %contrast2(10,70,3,1) / adjust=simulate(seed=1);
ods output Estimates=EstAdjust;
```

The following GLIMMIX invocations create output data sets named EstAdjust and EstUnAdjust that contain (non-step-down-) adjusted and unadjusted *p*-values:

```
proc glimmix data=clinical;
   t = (t3+t4)/2;
   class drug;
   model t = drug t0 age drug*age drug*t0;
   estimate %contrast(10,70,3,1)
              / adjust=simulate(seed=1);
   ods output Estimates=EstAdjust;
run;
proc glimmix data=clinical;
   t = (t3+t4)/2;
   class drug;
   model t = drug t0 age drug*age drug*t0;
   estimate %contrast(10,70,3,1);
```

```
    ods output Estimates=EstUnAdjust;
  run;
```

Output 52.13.2 shows a comparison of the significant comparisons ($p < 0.05$) based on unadjusted, adjusted, and step-down (TCTP) adjusted p-values. Clearly, the unadjusted results indicate the most significant results, but without protecting the Type I error rate for the group of tests. The adjusted p-values (filled circles) lead to a much smaller region in which the response surfaces between treatment and placebo are significantly different. The increased power of the TCTP procedure (open circles) over the standard multiplicity adjustment—without sacrificing Type I error protection—can be seen in the considerably larger region covered by the open circles.

Output 52.13.2 Comparison of Significance Regions

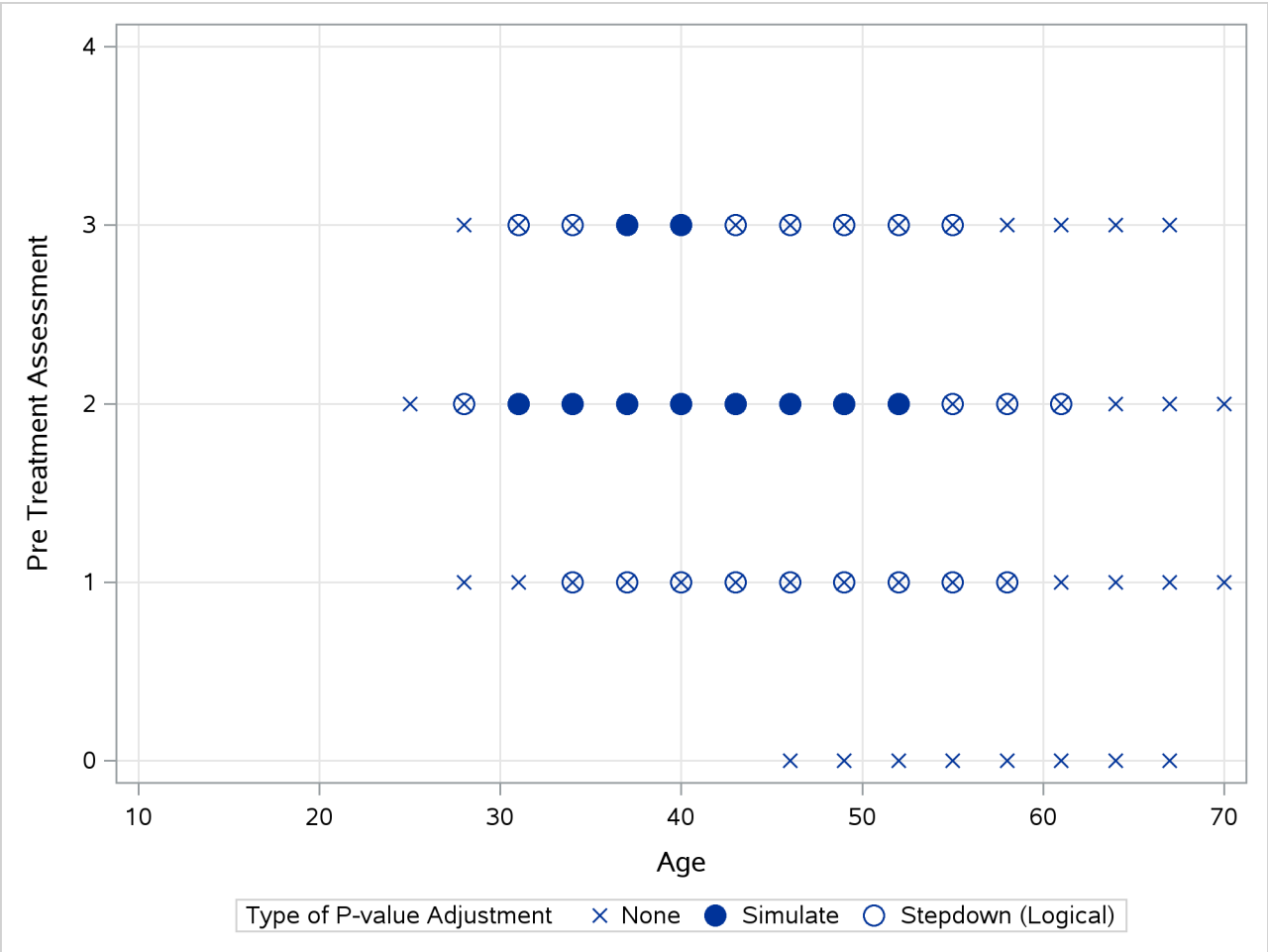

Ordinal Repeated Measure Analysis

The outcome variable in this clinical trial is an ordinal rating of patients in categories 0=terrible, 1=poor, 2=fair, 3=good, and 4=excellent. Furthermore, the observations from repeat visits for the same patients are likely correlated. The previous analysis removes the repeated measures aspect by defining efficacy as the average score at the final two visits. These averages are not normally distributed, however. The response surfaces for the two study arms can also be compared based on a model for ordinal data that takes correlation into account through random effects. Keeping with the theme of the previous analysis, the focus here for

illustrative purposes is on the final two visits, and the pretreatment assessment score serves as a covariate in the model.

The following DATA step rearranges the data from the third and fourth on-treatment visits in univariate form with separate observations for the visits by patient:

```
data clinical_uv;
   set clinical;
   array time{2} t3-t4;
   do i=1 to 2; rating = time{i}; output; end;
run;
```

The basic model for the analysis is a proportional odds model with cumulative logit link (McCullagh 1980) and $J = 5$ categories. In this model, separate intercepts (cutoffs) are modeled for the first $J - 1 = 4$ cumulative categories and the intercepts are monotonically increasing. This guarantees ordering of the cumulative probabilities and nonnegative category probabilities. Using the same covariate structure as in the previous analysis, the probability to observe a rating in at most category $k \leq 4$ is

$$\Pr(Y_d \leq k) = \frac{1}{1 + \exp\{-\eta_{kd}\}}$$
$$\eta_{kd} = \alpha_k + \beta_{0d} + \beta_{1d}t + \beta_{2d}a$$

Because only the intercepts are dependent on the category, contrasts comparing regression coefficients can be formulated in standard fashion. To accommodate the random and covariance structure of the repeated measures model, a random intercept γ_i is applied to the observations for each patient:

$$\Pr(Y_{id} \leq k) = \frac{1}{1 + \exp\{-\eta_{ikd}\}}$$
$$\eta_{ikd} = \alpha_k + \beta_{0d} + \beta_{1d}t + \beta_{2d}a + \gamma_i$$
$$\gamma_i \sim \text{iid}\, N(0, \sigma_\gamma^2)$$

The shared random effect of the two observations creates a marginal correlation. Note that the random effects do not depend on category.

The following GLIMMIX statements fit this ordinal repeated measures model by maximum likelihood via the Laplace approximation and compute TCTP-adjusted p-values for the 105 estimates:

```
proc glimmix data=clinical_uv method=laplace;
   class center id drug;
   model rating = drug t0 age drug*age drug*t0 /
                  dist=multinomial link=cumlogit;
   random intercept / subject=id(center);
   covtest 0;
   estimate %contrast(10,70,3,1)
             / adjust=simulate(seed=1)
               stepdown(type=logical);
   ods output Estimates=EstStepDownMulti;
run;
```

The combination of DIST=MULTINOMIAL and LINK=CUMLOGIT requests the proportional odds model. The SUBJECT= effect nests patient IDs within centers, because patient IDs in the data set clinical are not

unique within centers. (Specifying SUBJECT=ID*CENTER would have the same effect.) The COVTEST statement requests a likelihood ratio test for the significance of the random patient effect.

The estimate of the variance component for the random patient effect is substantial (Output 52.13.3), but so is its standard error.

Output 52.13.3 Model and Covariance Parameter Information

The GLIMMIX Procedure

Model Information	
Data Set	WORK.CLINICAL_UV
Response Variable	rating
Response Distribution	Multinomial (ordered)
Link Function	Cumulative Logit
Variance Function	Default
Variance Matrix Blocked By	ID(Center)
Estimation Technique	Maximum Likelihood
Likelihood Approximation	Laplace
Degrees of Freedom Method	Containment

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	ID(Center)	10.3483	3.2599

Tests of Covariance Parameters Based on the Likelihood

Label	DF	-2 Log Like	ChiSq	Pr > ChiSq	Note
Parameter list	1	604.70	57.64	<.0001	MI

MI: P-value based on a mixture of chi-squares.

The likelihood ratio test provides a better picture of the significance of the variance component. The difference in the –2 log likelihoods is 57.6, highly significant even if one does not apply the Self and Liang (1987) correction that halves the *p*-value in this instance.

The results for the 20 most significant estimates are requested with the following statements and shown in Output 52.13.4:

```
proc sort data=EstStepDownMulti;
   by Probt;
run;
proc print data=EstStepDownMulti(obs=20);
   var Label Estimate StdErr Probt AdjP;
run;
```

The *p*-values again show the “repeat” pattern corresponding to the monotonicity of the step-down procedure.

Output 52.13.4 The First 20 Estimates in the Ordinal Analysis

Obs	Label	Estimate	StdErr	Probt	Adjp
1	37 2	-2.7224	0.8263	0.0013	0.0133
2	40 2	-2.8857	0.8842	0.0015	0.0133
3	34 2	-2.5590	0.7976	0.0018	0.0133
4	43 2	-3.0491	0.9659	0.0021	0.0133
5	46 2	-3.2124	1.0660	0.0032	0.0133
6	31 2	-2.3957	0.8010	0.0034	0.0133
7	49 2	-3.3758	1.1798	0.0051	0.0164
8	52 2	-3.5391	1.3037	0.0077	0.0236
9	40 3	-2.6263	0.9718	0.0080	0.0267
10	37 3	-2.4630	0.9213	0.0087	0.0267
11	28 2	-2.2323	0.8362	0.0088	0.0278
12	43 3	-2.7897	1.0451	0.0088	0.0278
13	46 3	-2.9530	1.1368	0.0107	0.0291
14	55 2	-3.7025	1.4351	0.0112	0.0324
15	34 3	-2.2996	0.8974	0.0118	0.0337
16	49 3	-3.1164	1.2428	0.0136	0.0344
17	43 1	-3.3085	1.3438	0.0154	0.0448
18	58 2	-3.8658	1.5722	0.0155	0.0448
19	40 1	-3.1451	1.2851	0.0160	0.0448
20	46 1	-3.4718	1.4187	0.0160	0.0448

As previously, the comparisons were also performed with standard p-value adjustment via simulation. Output 52.13.5 displays the components of the regressor space in which the response surfaces differ significantly ($p < 0.05$) between the two treatment arms. As before, the most significant differences occur with unadjusted p-values at the cost of protecting only the individual Type I error rate. The standard multiplicity adjustment has considerably less power than the TCTP adjustment.

Output 52.13.5 Comparison of Significance Regions, Ordinal Analysis

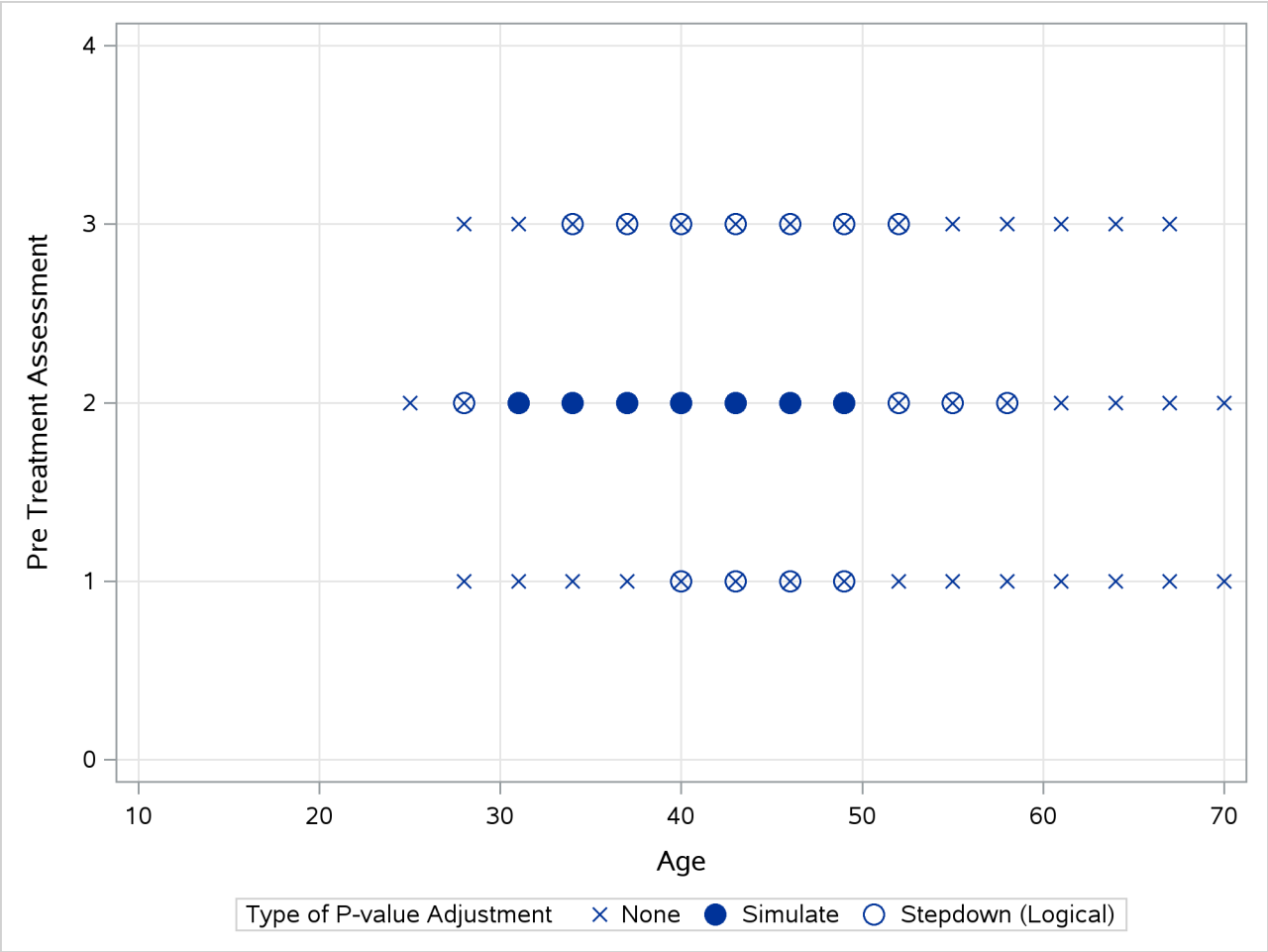

Example 52.14: Generalized Poisson Mixed Model for Overdispersed Count Data

(View the complete code for this example (gmxex14.sas) in the example repository.)

Overdispersion is the condition by which data appear more dispersed than is expected under a reference model. For count data, the reference models are typically based on the binomial or Poisson distributions. Among the many reasons for overdispersion are an incorrect model, an incorrect distributional specification, incorrect variance functions, positive correlation among the observations, and so forth. In short, correcting an overdispersion problem, if it exists, requires the appropriate remedy. Adding an R-side scale parameter to multiply the variance function is not necessarily the adequate correction. For example, Poisson-distributed data appear overdispersed relative to a Poisson model with regressors when an important regressor is omitted.

If the reference model for count data is Poisson, a number of alternative model formulations are available to increase the dispersion. For example, zero-inflated models add a proportion of zeros (usually from a Bernoulli process) to the zeros of a Poisson process. Hurdle models are two-part models where zeros and nonzeros are generated by different stochastic processes. Zero-inflated and hurdle models are described in

detail by Cameron and Trivedi (1998) and cannot be fit with the GLIMMIX procedure. See Section 15.5 in Littell et al. (2006) for examples of using the NLMIXED procedure to fit zero-inflated and hurdle models.

An alternative approach is to derive from the reference distribution a probability distribution that exhibits increased dispersion. By mixing a Poisson process with a gamma distribution for the Poisson parameter, for example, the negative binomial distribution results, which is thus overdispersed relative to the Poisson.

Joe and Zhu (2005) show that the generalized Poisson distribution can also be motivated as a Poisson mixture and hence provides an alternative to the negative binomial (NB) distribution. Like the NB, the generalized Poisson distribution has a scale parameter. It is heavier in the tails than the NB distribution and easily reduces to the standard Poisson. Joe and Zhu (2005) discuss further comparisons between these distributions.

The probability mass function of the generalized Poisson is given by

$$p(y) = \frac{\alpha}{y!}\,(\alpha + \xi y)^{y-1} \exp\{-\alpha - \xi y\}$$

where $y = 0, 1, 2, \ldots$, $\alpha > 0$, and $0 \le \xi < 1$ (Joe and Zhu 2005). Notice that for $\xi = 0$ the mass function of the standard Poisson distribution with mean α results. The mean and variance of Y in terms of the parameters α and ξ are given by

$$\begin{aligned} \mathrm{E}[Y] &= \frac{\alpha}{1-\xi} = \mu \\ \mathrm{Var}[Y] &= \frac{\alpha}{(1-\xi)^3} = \frac{\mu}{(1-\xi)^2} \end{aligned}$$

The log likelihood of the generalized Poisson can thus be written in terms of the mean μ and scale parameter ξ as

$$\begin{aligned} l(\mu, \xi; y) = {} & \log\{\mu(1-\xi)\} + (y-1)\log\{\mu - \xi(\mu - y)\} \\ & - (\mu - \xi(\mu - y)) - \log\{\Gamma(y+1)\} \end{aligned}$$

The data in the following DATA step are simulated counts. For each of $i = 1, \ldots, 30$ subjects a randomly varying number n_i of observations were drawn from a count regression model with a single covariate and excess zeros (compared to a Poisson distribution).

```
data counts;
   input ni @@;
   sub = _n_;
   do i=1 to ni;
      input x y @@;
      output;
   end;
   datalines;
  1 29 0
  6  2 0 82 5 33 0 15 2 35 0 79 0
 19 81 0 18 0 85 0 99 0 20 0 26 2 29 0 91 2 37 0 39 0  9 1 33 0
     3 0 60 0 87 2 80 0 75 0  3 0 63 1
  9 18 0 64 0 80 0  0 0 58 0  7 0 81 0 22 3 50 0
 15 91 0  2 1 14 0  5 2 27 1  8 1 95 0 76 0 62 0 26 2  9 0 72 1
    98 0 94 0 23 1
  2 34 0 95 0
 18 48 1  5 0 47 0 44 0 27 0 88 0 27 0 68 0 84 0 86 0 44 0 90 0
```

```
    63 0 27 0 47 0 25 0 72 0 62 1
 13 28 1 31 0 63 0 14 0 74 0 44 0 75 0 65 0 74 1 84 0 57 0 29 0
    41 0
  9 42 0  8 0 91 0 20 0 23 0 22 0 96 0 83 0 56 0
  3 64 0 64 1 15 0
  4  5 0 73 2 50 1 13 0
  2  0 0 41 0
 20 21 0 58 0  5 0 61 1 28 0 71 0 75 1 94 16 51 4 51 2 74 0  1 1
    34 0  7 0 11 0 60 3 31 0 75 0 62 0 54 1
  2 66 1 13 0
  5 83 7 98 1 11 1 28 0 18 0
 17 29 5 79 0 39 2 47 2 80 1 19 0 37 0 78 1 26 0 72 1  6 0 50 3
    50 4 97 0 37 2 51 0 45 0
 17 47 0 57 0 33 0 47 0  2 0 83 0 74 0 93 0 36 0 53 0 26 0 86 0
     6 0 17 0 30 0 70 1 99 0
  7 91 0 25 1 51 4 20 0 61 1 34 0 33 2
 14 60 0 87 0 94 0 29 0 41 0 78 0 50 0 37 0 15 0 39 0 22 0 82 0
    93 0  3 0
 16 68 0 26 1 19 0 60 1 93 3 65 0 16 0 79 0 14 0  3 1 90 0 28 3
    82 0 34 0 30 0 81 0
 19 48 3 48 1 43 2 54 0 45 9 53 0 14 0 92 5 21 1 20 0 73 0 99 0
    66 0 86 2 63 0 10 0 92 14 44 1 74 0
  8 34 1 44 0 62 0 21 0  7 0 17 0  0 2 49 0
 13 11 0 27 2 16 1 12 3 52 1 55 0  2 6 89 5 31 5 28 3 51 5 54 13
    64 0
  9  3 0 36 0 57 0 77 0 41 0 39 0 55 0 57 0 88 1
  7  2 0 80 0 41 1 20 0  2 0 27 0 40 0
 18 73 1 66 0 10 0 42 0 22 0 59 9 68 0 34 1 96 0 30 0 13 0 35 0
    51 2 47 0 60 1 55 4 83 3 38 0
 17 96 0 40 0 34 0 59 0 12 1 47 0 93 0 50 0 39 0 97 0 19 0 54 0
    11 0 29 0 70 2 87 0 47 0
 13 59 0 96 0 47 1 64 0 18 0 30 0 37 0 36 1 69 0 78 1 47 1 86 0
    88 0
 15 66 0 45 1 96 1 17 0 91 0  4 0 22 0  5 2 47 0 38 0 80 0  7 1
    38 1 33 0 52 0
 12 84 6 60 1 33 1 92 0 38 0  6 0 43 3 13 2 18 0 51 0 50 4 68 0
;
```

The following PROC GLIMMIX statements fit a standard Poisson regression model with random intercepts by maximum likelihood. The marginal likelihood of the data is approximated by adaptive quadrature (METHOD=QUAD).

```
proc glimmix data=counts method=quad;
   class sub;
   model y = x / link=log s dist=poisson;
   random int / subject=sub;
run;
```

Output 52.14.1 displays various informational items about the model and the estimation process.

Output 52.14.1 Poisson: Model and Optimization Information

The GLIMMIX Procedure

Model Information	
Data Set	WORK.COUNTS
Response Variable	y
Response Distribution	Poisson
Link Function	Log
Variance Function	Default
Variance Matrix Blocked By	sub
Estimation Technique	Maximum Likelihood
Likelihood Approximation	Gauss-Hermite Quadrature
Degrees of Freedom Method	Containment

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	3
Lower Boundaries	1
Upper Boundaries	0
Fixed Effects	Not Profiled
Starting From	GLM estimates
Quadrature Points	5

Iteration History

Iteration	Restarts	Evaluations	Objective Function	Change	Max Gradient
0	**0**	4	862.57645728	.	366.7105
1	**0**	5	862.43893582	0.13752147	22.36158
2	**0**	6	854.49131023	7.94762559	28.70814
3	**0**	2	854.47983504	0.01147519	6.036114
4	**0**	4	854.47396189	0.00587315	4.238363
5	**0**	4	854.47006558	0.00389631	0.332454
6	**0**	3	854.47006484	0.00000074	0.003104

The "Model Information" table shows that the parameters are estimated by ML with quadrature. Using the starting values for fixed effects and covariance parameters that the GLIMMIX procedure generates by default, the procedure determined that five quadrature nodes provide a sufficiently accurate approximation of the marginal log likelihood ("Optimization Information" table). The iterative estimation process converges after nine iterations.

The table of conditional fit statistics displays the sum of the independent contributions to the conditional –2 log likelihood (854.47) and the Pearson statistics for the conditional distribution (Output 52.14.2).

Output 52.14.2 Poisson: Fit Statistics and Estimates

Fit Statistics	
-2 Log Likelihood	854.47
AIC (smaller is better)	860.47
AICC (smaller is better)	860.54
BIC (smaller is better)	864.67
CAIC (smaller is better)	867.67
HQIC (smaller is better)	861.81

Fit Statistics for Conditional Distribution	
-2 log L(y | r. effects)	777.90
Pearson Chi-Square	649.58
Pearson Chi-Square / DF	1.97

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	sub	1.1959	0.4334

Solutions for Fixed Effects

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	-1.4947	0.2745	29	-5.45	<.0001
x	0.01207	0.002387	299	5.06	<.0001

The departure of the scaled Pearson statistic from 1.0 is fairly pronounced in this case (1.97). If one deems it too far from 1.0, however, the conclusion has to be that the conditional variation is not properly specified. This could be due to an incorrect variance function, for example. The "Solutions for Fixed Effects" table shows the estimates of the slope and intercept in this model along with their standard errors and tests of significance. Note that the slope in this model is highly significant. The variance of the random subject-specific intercepts is estimated as 1.1959.

To fit the generalized Poisson distribution to these data, you cannot draw on the built-in distributions. Instead, the variance function and the log likelihood are computed directly with PROC GLIMMIX programming statements. The CLASS, MODEL, and RANDOM statements in the following PROC GLIMMIX program are as before, except for the omission of the DIST= option in the MODEL statement:

```
proc glimmix data=counts method=quad;
   class sub;
   model y = x / link=log s;
   random int / subject=sub;
   xi = (1 - 1/exp(_phi_));
   _variance_ = _mu_ / (1-xi)/(1-xi);
   if (_mu_=.) or (_linp_ = .) then _logl_ = .;
   else do;
      mustar = _mu_ - xi*(_mu_ - y);
      if (mustar < 1E-12) or (_mu_*(1-xi) < 1e-12) then
         _logl_ = -1E20;
      else do;
         _logl_ = log(_mu_*(1-xi)) + (y-1)*log(mustar) -
```

```
                    mustar - lgamma(y+1);
        end;
      end;
   run;
```

The assignments to the variables xi and the reserved symbols _VARIANCE_ and _LOGL_ define the variance function and the log likelihood. Because the scale parameter of the generalized Poisson distribution has the range $0 < \xi < 1$, and the scale parameter _PHI_ in the GLIMMIX procedure is bounded only from below (by 0), a reparameterization is applied so that $\phi = 0 \Leftrightarrow \xi = 0$ and ξ approaches 1 as ϕ increases. The statements preceding the calculation of the actual log likelihood are intended to prevent floating-point exceptions and to trap missing values.

Output 52.14.3 displays information about the model and estimation process. The "Model Information" table shows that the distribution is not a built-in distribution and echoes the expression for the user-specified variance function. As in the case of the Poisson model, the GLIMMIX procedure determines that five quadrature points are sufficient for accurate estimation of the marginal log likelihood at the starting values. The estimation process converges after 11 iterations.

Output 52.14.3 Generalized Poisson: Model, Optimization, and Iteration Information

The GLIMMIX Procedure

Model Information	
Data Set	WORK.COUNTS
Response Variable	y
Response Distribution	User specified
Link Function	Log
Variance Function	_mu_ / (1-xi)/(1-xi)
Variance Matrix Blocked By	sub
Estimation Technique	Maximum Likelihood
Likelihood Approximation	Gauss-Hermite Quadrature
Degrees of Freedom Method	Containment

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	4
Lower Boundaries	2
Upper Boundaries	0
Fixed Effects	Not Profiled
Starting From	GLM estimates
Quadrature Points	5

Output 52.14.3 *continued*

Iteration History					
Iteration	Restarts	Evaluations	Objective Function	Change	Max Gradient
0	0	4	716.12976769	.	161.1184
1	0	5	716.07585953	0.05390816	11.88788
2	0	4	714.27148068	1.80437884	36.09657
3	0	2	711.02643265	3.24504804	108.4615
4	0	2	710.26952196	0.75691069	216.9822
5	0	2	709.96824991	0.30127205	96.2775
6	0	3	709.8419071	0.12634280	19.07487
7	0	3	709.83122731	0.01067980	0.649164
8	0	3	709.83047646	0.00075085	2.127665
9	0	3	709.83046461	0.00001185	0.383319
10	0	3	709.83046436	0.00000025	0.010279

The achieved –2 log likelihood is lower than in the Poisson model (compare "Fit Statistics" tables in Output 52.14.4 and Output 52.14.1). The scaled Pearson statistic is now less than 1.0. The fixed slope estimate remains significant at the 5% level, but the test statistics are not as large as in the Poisson model, partly because the generalized Poisson model permits more variation.

Output 52.14.4 Generalized Poisson: Fit Statistics and Estimates

Fit Statistics	
-2 Log Likelihood	709.83
AIC (smaller is better)	717.83
AICC (smaller is better)	717.95
BIC (smaller is better)	723.44
CAIC (smaller is better)	727.44
HQIC (smaller is better)	719.62

Fit Statistics for Conditional Distribution	
-2 log L(y | r. effects)	665.56
Pearson Chi-Square	241.42
Pearson Chi-Square / DF	0.73

Covariance Parameter Estimates			
Cov Parm	Subject	Estimate	Standard Error
Intercept	sub	0.5135	0.2400
Scale		0.6401	0.09718

Solutions for Fixed Effects					
Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	-0.7264	0.2749	29	-2.64	0.0131
x	0.003742	0.003537	299	1.06	0.2910

Based on the large difference in the –2 log likelihoods between the Poisson and generalized Poisson models,

you conclude that a mixed model based on the latter provides a better fit to these data. From the "Covariance Parameter Estimates" table in Output 52.14.4, you can see that the estimate of the scale parameter is $\widehat{\phi} = 0.6401$ and is considerably larger than 0, taking into account its standard error. The hypothesis $H{:}\phi = 0$, which articulates that a Poisson model fits the data as well as the generalized Poisson model, can be formally tested with a likelihood ratio test. Adding the following statement to the previous PROC GLIMMIX run compares the model to one in which the variance of the random intercepts (the first covariance parameter) is not constrained and the scale parameter is fixed at zero:

```
covtest 'H: phi = 0' . 0 / est;
```

This COVTEST statement produces Output 52.14.5.

Output 52.14.5 Likelihood Ratio Test for Poisson Assumption

Tests of Covariance Parameters Based on the Likelihood

Label	DF	-2 Log Like	ChiSq	Pr > ChiSq	Estimates H0 Est1	Est2	Note
H:phi = 0	1	854.47	144.64	<.0001	1.1959	1.11E-12	MI

MI: P-value based on a mixture of chi-squares.

Note that the –2 Log Like reported in Output 52.14.5 agrees with the value reported in the "Fit Statistics" table for the Poisson model (Output 52.14.2) and that the estimate of the random intercept under the null hypothesis agrees with the "Covariance Parameter Estimates" table in Output 52.14.2. Because the null hypothesis places the parameter ϕ (or ξ) on the boundary of the parameter space, a mixture correction is applied in the p-value calculation. Because of the magnitude of the likelihood ratio statistic (144.64), this correction has no effect on the displayed p-value.

Example 52.15: Comparing Multiple B-Splines

(View the complete code for this example (gmxex15.sas) in the example repository.)

This example uses simulated data to demonstrate the use of the nonpositional syntax (see the section "Positional and Nonpositional Syntax for Contrast Coefficients" on page 3973 for details) in combination with the EFFECT statement to produce interesting predictions and comparisons in models containing fixed spline effects. Consider the data in the following DATA step. Each of the 100 observations for the continuous response variable y is associated with one of two groups.

```
data spline;
   input group y @@;
   x = _n_;
   datalines;
 1    -.020 1    0.199 2    -1.36 1    -.026
 2    -.397 1    0.065 2    -.861 1    0.251
 1    0.253 2    -.460 2    0.195 2    -.108
 1    0.379 1    0.971 1    0.712 2    0.811
 2    0.574 2    0.755 1    0.316 2    0.961
 2    1.088 2    0.607 2    0.959 1    0.653
 1    0.629 2    1.237 2    0.734 2    0.299
```

```
 2    1.002 2    1.201 1    1.520 1    1.105
 1    1.329 1    1.580 2    1.098 1    1.613
 2    1.052 2    1.108 2    1.257 2    2.005
 2    1.726 2    1.179 2    1.338 1    1.707
 2    2.105 2    1.828 2    1.368 1    2.252
 1    1.984 2    1.867 1    2.771 1    2.052
 2    1.522 2    2.200 1    2.562 1    2.517
 1    2.769 1    2.534 2    1.969 1    2.460
 1    2.873 1    2.678 1    3.135 2    1.705
 1    2.893 1    3.023 1    3.050 2    2.273
 2    2.549 1    2.836 2    2.375 2    1.841
 1    3.727 1    3.806 1    3.269 1    3.533
 1    2.948 2    1.954 2    2.326 2    2.017
 1    3.744 2    2.431 2    2.040 1    3.995
 2    1.996 2    2.028 2    2.321 2    2.479
 2    2.337 1    4.516 2    2.326 2    2.144
 2    2.474 2    2.221 1    4.867 2    2.453
 1    5.253 2    3.024 2    2.403 1    5.498
;
```

The following statements produce a scatter plot of the response variable by group (Output 52.15.1):

```
proc sgplot data=spline;
   scatter y=y x=x / group=group name="data";
   keylegend "data"  / title="Group";
run;
```

Output 52.15.1 Scatter Plot of Observed Data by Group

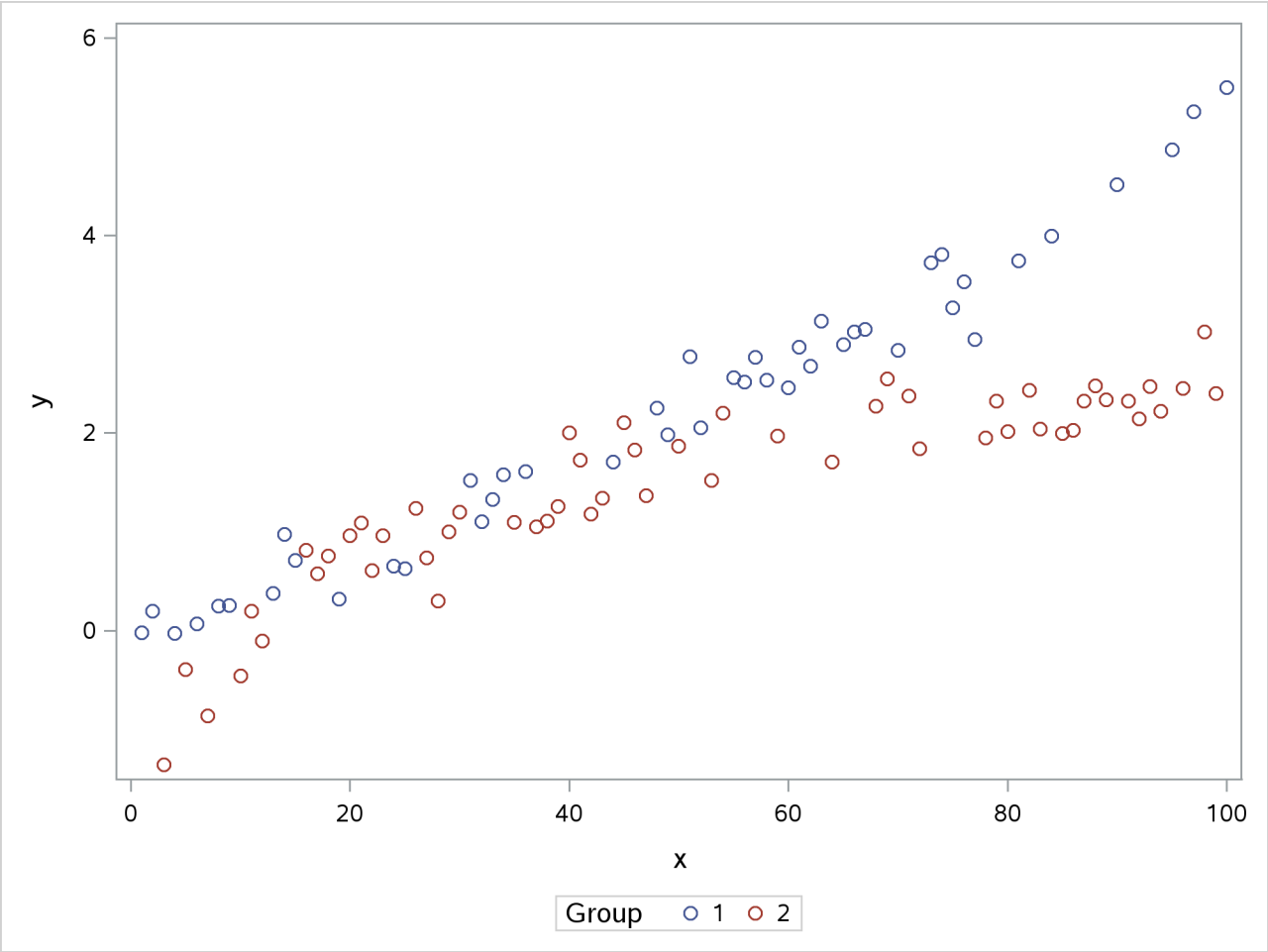

The trends in the two groups exhibit curvature, but the type of curvature is not the same in the groups. Also, there appear to be ranges of x values where the groups are similar and areas where the point scatters separate. To model the trends in the two groups separately and with flexibility, you might want to allow for some smooth trends in x that vary by group. Consider the following PROC GLIMMIX statements:

```
proc glimmix data=spline outdesign=x;
   class group;
   effect spl = spline(x);
   model y = group spl*group / s noint;
   output out=gmxout pred=p;
run;
```

The EFFECT statement defines a constructed effect named spl by expanding the x into a spline with seven columns. The group main effect creates separate intercepts for the groups, and the interaction of the group variable with the spline effect creates separate trends. The NOINT option suppresses the intercept. This is not necessary and is done here only for convenience of interpretation. The OUTPUT statement computes predicted values.

The "Parameter Estimates" table contains the estimates of the group-specific "intercepts," the spline coefficients varied by group, and the residual variance ("Scale," Output 52.15.2).

Output 52.15.2 Parameter Estimates in Two-Group Spline Model

The GLIMMIX Procedure

Parameter Estimates							
Effect	spl	group	Estimate	Standard Error	DF	t Value	Pr > |t|
group		1	9.7027	3.1342	86	3.10	0.0026
group		2	6.3062	2.6299	86	2.40	0.0187
spl*group	1	1	-11.1786	3.7008	86	-3.02	0.0033
spl*group	1	2	-20.1946	3.9765	86	-5.08	<.0001
spl*group	2	1	-9.5327	3.2576	86	-2.93	0.0044
spl*group	2	2	-5.8565	2.7906	86	-2.10	0.0388
spl*group	3	1	-8.9612	3.0718	86	-2.92	0.0045
spl*group	3	2	-5.5567	2.5717	86	-2.16	0.0335
spl*group	4	1	-7.2615	3.2437	86	-2.24	0.0278
spl*group	4	2	-4.3678	2.7247	86	-1.60	0.1126
spl*group	5	1	-6.4462	2.9617	86	-2.18	0.0323
spl*group	5	2	-4.0380	2.4589	86	-1.64	0.1042
spl*group	6	1	-4.6382	3.7095	86	-1.25	0.2146
spl*group	6	2	-4.3029	3.0479	86	-1.41	0.1616
spl*group	7	1	0	.	.	.	.
spl*group	7	2	0	.	.	.	.
Scale			0.07352	0.01121	.	.	.

Because the B-spline coefficients for an observation sum to 1 and the model contains group-specific constants, the last spline coefficient in each group is zero. In other words, you can achieve exactly the same fit with the MODEL statement

```
model y = spl*group / noint;
```

or

```
model y = spl*group;
```

The following statements graph the observed and fitted values in the two groups (Output 52.15.3):

```
proc sgplot data=gmxout;
   series  y=p x=x / group=group name="fit";
   scatter y=y x=x / group=group;
   keylegend "fit" / title="Group";
run;
```

Output 52.15.3 Observed and Predicted Values by Group

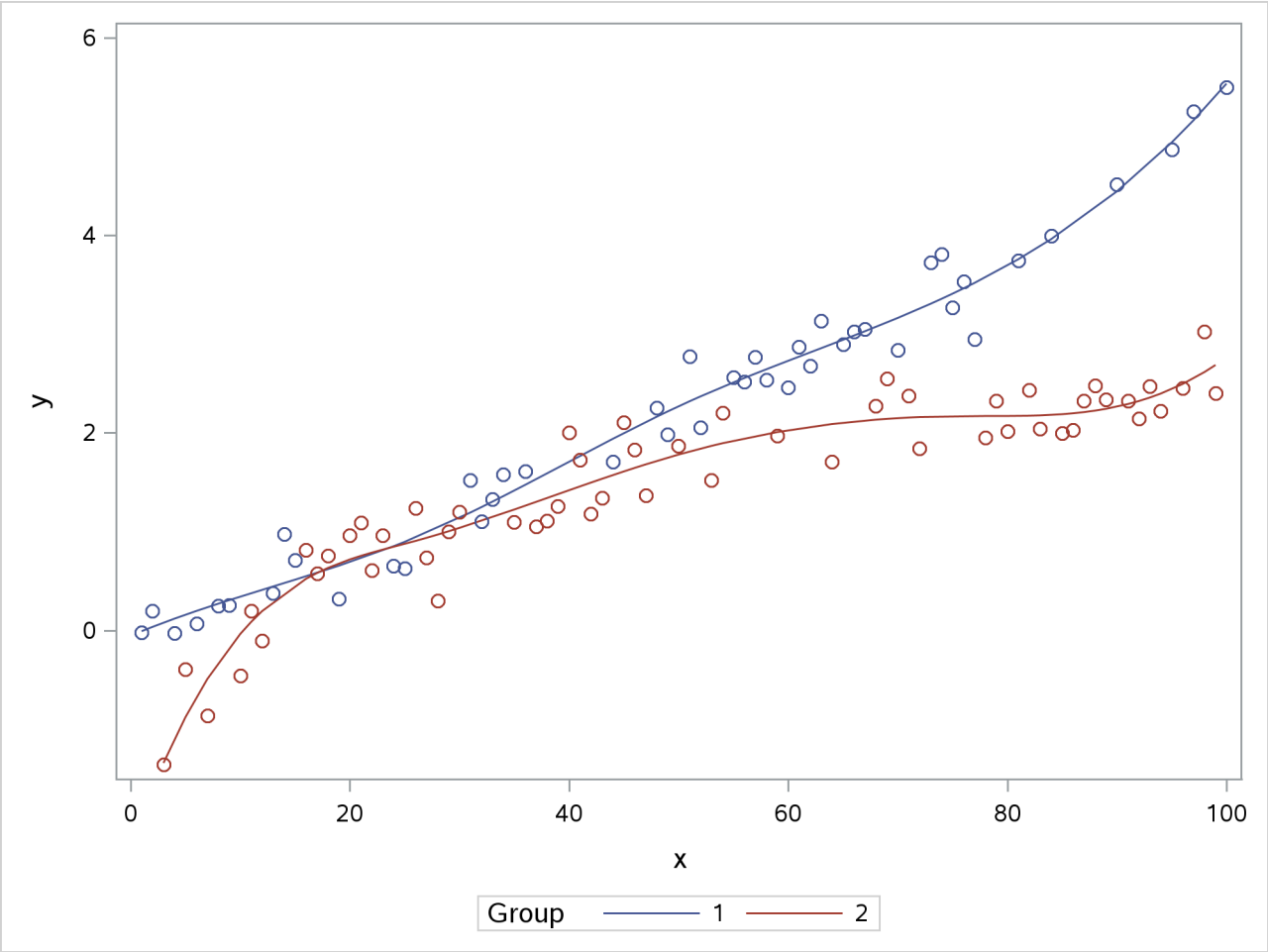

Suppose that you are interested in estimating the mean response at particular values of x and in performing comparisons of predicted values. The following program uses ESTIMATE statements with nonpositional syntax to accomplish this:

```
proc glimmix data=spline;
   class group;
   effect spl = spline(x);
   model y = group spl*group / s noint;
   estimate 'Group 1, x=20' group 1    group*spl [1,1 20] / e;
   estimate 'Group 2, x=20' group 0  1 group*spl [1,2 20];
   estimate 'Diff at x=20 ' group 1 -1 group*spl [1,1 20] [-1,2 20];
run;
```

The first ESTIMATE statement predicts the mean response at $x = 20$ in group 1. The E option requests the coefficient vector for this linear combination of the parameter estimates. The coefficient for the group effect is entered with positional (standard) syntax. The coefficients for the group*spl effect are formed based on nonpositional syntax. Because this effect comprises the interaction of a standard effect (group) with a constructed effect, the values and levels for the standard effect must precede those for the constructed effect. A similar statement produces the predicted mean at $x = 20$ in group 2.

The GLIMMIX procedure interprets the syntax

```
group*spl [1,2 20]
```

as follows: construct the spline basis at $x = 20$ as appropriate for group 2; then multiply the resulting coefficients for these columns of the **L** matrix with 1.

The final ESTIMATE statement represents the difference between the predicted values; it is a group comparison at $x = 20$.

Output 52.15.4 Coefficients from First ESTIMATE Statement

The GLIMMIX Procedure

Coefficients for Estimate Group 1, x=20			
Effect	**spl**	**group**	**Row1**
group		1	1
group		2	
spl*group	1	1	0.0021
spl*group	1	2	
spl*group	2	1	0.3035
spl*group	2	2	
spl*group	3	1	0.619
spl*group	3	2	
spl*group	4	1	0.0754
spl*group	4	2	
spl*group	5	1	
spl*group	5	2	
spl*group	6	1	
spl*group	6	2	
spl*group	7	1	
spl*group	7	2	

The "Coefficients" table shows how the value 20 supplied in the ESTIMATE statement was expanded into the appropriate spline basis (Output 52.15.4). There is no significant difference between the group means at $x = 20$ ($p = 0.8346$, Output 52.15.5).

Output 52.15.5 Results from ESTIMATE Statements

Estimates					
Label	**Estimate**	**Standard Error**	**DF**	**t Value**	**Pr > |t|**
Group 1, x=20	0.6915	0.09546	86	7.24	<.0001
Group 2, x=20	0.7175	0.07953	86	9.02	<.0001
Diff at x=20	-0.02602	0.1243	86	-0.21	0.8346

The group comparisons you can achieve in this way are comparable to slices of interaction effects with classification effects. There are, however, no preset number of levels at which to perform the comparisons because x is continuous. If you add further x values for the comparisons, a multiplicity correction is in order to control the familywise Type I error. The following statements compare the groups at values $x = 0, 5, 10, \ldots, 80$ and compute simulation-based step-down-adjusted p-values. The results appear in

Output 52.15.6. (The numeric results for simulation-based *p*-value adjustments depend slightly on the value of the random number seed.)

```
ods select Estimates;
proc glimmix data=spline;
   class group;
   effect spl = spline(x);
   model y = group spl*group / s;
   estimate 'Diff at x= 0' group 1 -1 group*spl [1,1  0] [-1,2  0],
            'Diff at x= 5' group 1 -1 group*spl [1,1  5] [-1,2  5],
            'Diff at x=10' group 1 -1 group*spl [1,1 10] [-1,2 10],
            'Diff at x=15' group 1 -1 group*spl [1,1 15] [-1,2 15],
            'Diff at x=20' group 1 -1 group*spl [1,1 20] [-1,2 20],
            'Diff at x=25' group 1 -1 group*spl [1,1 25] [-1,2 25],
            'Diff at x=30' group 1 -1 group*spl [1,1 30] [-1,2 30],
            'Diff at x=35' group 1 -1 group*spl [1,1 35] [-1,2 35],
            'Diff at x=40' group 1 -1 group*spl [1,1 40] [-1,2 40],
            'Diff at x=45' group 1 -1 group*spl [1,1 45] [-1,2 45],
            'Diff at x=50' group 1 -1 group*spl [1,1 50] [-1,2 50],
            'Diff at x=55' group 1 -1 group*spl [1,1 55] [-1,2 55],
            'Diff at x=60' group 1 -1 group*spl [1,1 60] [-1,2 60],
            'Diff at x=65' group 1 -1 group*spl [1,1 65] [-1,2 65],
            'Diff at x=70' group 1 -1 group*spl [1,1 70] [-1,2 70],
            'Diff at x=75' group 1 -1 group*spl [1,1 75] [-1,2 75],
            'Diff at x=80' group 1 -1 group*spl [1,1 80] [-1,2 80] /
           adjust=sim(seed=1) stepdown;
run;
```

Output 52.15.6 Estimates with Multiplicity Adjustments

The GLIMMIX Procedure

Estimates
Adjustment for Multiplicity: Holm-Simulated

Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Adj P
Diff at x= 0	1.8809	0.3640	86	5.17	<.0001	<.0001
Diff at x= 5	1.0376	0.1759	86	5.90	<.0001	<.0001
Diff at x=10	0.3778	0.1540	86	2.45	0.0162	0.0551
Diff at x=15	0.05822	0.1481	86	0.39	0.6952	0.9102
Diff at x=20	-0.02602	0.1243	86	-0.21	0.8346	0.9557
Diff at x=25	0.02014	0.1312	86	0.15	0.8783	0.9557
Diff at x=30	0.1023	0.1378	86	0.74	0.4600	0.7426
Diff at x=35	0.1924	0.1236	86	1.56	0.1231	0.2965
Diff at x=40	0.2883	0.1114	86	2.59	0.0113	0.0453
Diff at x=45	0.3877	0.1195	86	3.24	0.0017	0.0085
Diff at x=50	0.4885	0.1308	86	3.74	0.0003	0.0025
Diff at x=55	0.5903	0.1231	86	4.79	<.0001	<.0001
Diff at x=60	0.7031	0.1125	86	6.25	<.0001	<.0001
Diff at x=65	0.8401	0.1203	86	6.99	<.0001	<.0001
Diff at x=70	1.0147	0.1348	86	7.52	<.0001	<.0001
Diff at x=75	1.2400	0.1326	86	9.35	<.0001	<.0001
Diff at x=80	1.5237	0.1281	86	11.89	<.0001	<.0001

There are significant differences at the low end and high end of the x range. Notice that without the multiplicity adjustment you would have concluded at the 0.05 level that the groups are significantly different at $x = 10$. At the 0.05 level, the groups separate significantly for $x < 10$ and $x > 40$.

Example 52.16: Diallel Experiment with Multimember Random Effects

(View the complete code for this example (gmxex16.sas) in the example repository.)

Cockerham and Weir (1977) apply variance component models in the analysis of reciprocal crosses. In these experiments, it is of interest to separate genetically determined variation from variation determined by parentage. This example analyzes the data for the diallel experiment in Cockerham and Weir (1977, Appendix C). A diallel is a mating design that consists of all possible crosses of a set of parental lines. It includes reciprocal crossings, but not self-crossings.

The basic model for a cross is $Y_{ijk} = \beta + \alpha_{ij} + \epsilon_{ijk}$, where Y_{ijk} is the observation for offspring k from maternal parent i and paternal parent j. The various models in Cockerham and Weir (1977) are different decompositions of the term α_{ij}, the total effect that is due to the parents. Their "bio model" (model (c)) decomposes α_{ij} into

$$\alpha_{ij} = \eta_i + \eta_j + \mu_i + \phi_j + (\eta\eta)_{ij} + \kappa_{ij}$$

where η_i and η_j are contributions of the female and male parents, respectively. The term $(\eta\eta)_{ij}$ captures the interaction between maternal and paternal effects. In contrast to usual interaction effects, this term must obey a symmetry because of the reciprocals: $(\eta\eta)_{ij} = (\eta\eta)_{ji}$. The terms μ_i and ϕ_j in the decomposition are extranuclear maternal and paternal effects, and the remaining interactions are captured by the κ_{ij} term.

The following DATA step creates a SAS data set for the diallel example in Appendix C of Cockerham and Weir (1977):

```
data diallel;
   label time = 'Flowering time in days';
   do p = 1 to 8;
      do m = 1 to 8;
         if (m ne p) then do;
            sym = trim(left(min(m,p))) || ',' || trim(left(max(m,p)));
            do block = 1 to 2;
               input time @@;
               output;
            end;
         end;
      end;
   end;
   datalines;
14.4 16.2 27.2 30.8 17.2 27.0 18.3 20.2 16.2 16.8 18.6 14.4 16.4 16.0
15.4 16.5 14.8 14.6 18.6 18.6 15.2 15.3 17.0 15.2 14.4 14.8 10.8 13.2
31.8 30.4 21.0 23.0 24.6 25.4 19.2 20.0 29.8 28.4 12.8 14.2 13.0 14.4
16.2 17.8 11.4 13.0 16.8 16.3 12.4 14.2 16.8 14.8 12.6 12.2  9.6 11.2
14.6 18.8 12.2 13.6 15.2 15.4 15.2 13.8 18.0 16.0 10.4 12.2 13.4 20.0
20.2 23.4 14.2 14.0 18.6 14.8 22.2 17.0 14.3 17.3  9.0 10.2 11.8 12.8
14.0 16.6 12.2  9.2 13.6 16.2 13.8 14.4 15.6 15.6 15.6 11.0 13.0  9.8
15.2 17.2 10.0 11.6 17.0 18.2 20.8 20.8 20.0 17.4 17.0 12.6 13.0  9.8
```

```
;
```

The observations represent mean flowering times of *Nicotiana rustica* (Aztec tobacco) from crosses of inbred varieties grown in two blocks. The variables p and m identify the eight paternal and maternal lines, respectively. The variable sym is used to model the interaction between the parents, subject to the symmetry condition $(\eta\eta)_{ij} = (\eta\eta)_{ji}$. For example, the first two observations, 14.4 and 16.2 days, represent the observations from blocks 1 and 2 where paternal line 1 was crossed with maternal line 2.

The following PROC GLIMMIX statements fit the "bio model" in Cockerham and Weir (1977):

```
proc glimmix data=diallel outdesign(z)=zmat;
   class block sym p m;
   effect line = mm(p m);
   model  time = block;
   random line sym p m p*m;
run;
```

The EFFECT statement defines the nuclear parental contributions as a multimember effect based on the CLASS variables p and m. Each observation has two nonzero entries in the design matrix for the effect that identifies the paternal and maternal lines. The terms in the RANDOM statement model the variance components as follows: line $\rightarrow \sigma^2_n$, sym $\rightarrow \sigma^2_{(\eta\eta)}$, p $\rightarrow \sigma^2_\phi$, m $\rightarrow \sigma^2_\mu$, p*m $\rightarrow \sigma^2_\kappa$. The OUTDESIGN= option in the PROC GLIMMIX statement writes the $\mathbf{Z}$ matrix to the SAS data set zmat. The EFFECT statement alleviates the need for complex coding, as in Section 2.3 of Saxton (2004).

Output 52.16.1 displays the "Class Level Information" table of the diallel model. Because the interaction terms are symmetric, there are only $8 \times 7/2 = 28$ levels for the 8 lines. The estimates of the variance components and the residual variance in Output 52.16.1 agree with the results in Table 7 of Cockerham and Weir (1977).

Output 52.16.1 Class Levels and Covariance Parameter Estimates in Diallel Example

The GLIMMIX Procedure

Class Level Information

Class	Levels	Values
block	2	1 2
sym	28	1,2 1,3 1,4 1,5 1,6 1,7 1,8 2,3 2,4 2,5 2,6 2,7 2,8 3,4 3,5 3,6 3,7 3,8 4,5 4,6 4,7 4,8 5,6 5,7 5,8 6,7 6,8 7,8
p	8	1 2 3 4 5 6 7 8
m	8	1 2 3 4 5 6 7 8

Covariance Parameter Estimates

Cov Parm	Estimate	Standard Error
line	5.1047	4.0021
sym	2.3856	1.9025
p	3.3080	3.4053
m	1.9134	2.9891
p*m	4.0196	1.8323
Residual	3.6225	0.6908

The following statements print the $\mathbf{Z}$ matrix columns that correspond to the multimember line effect for the first 10 observations in block 1 (Output 52.16.2). For each observation there are two nonzero entries, and

their column index corresponds to the index of the paternal and maternal line.

```
proc print data=zmat(where=(block=1) obs=10);
   var p m time _z1-_z8;
run;
```

Output 52.16.2 **Z** Matrix for Line Effect of the First 10 Observations in Block 1

Obs	p	m	time	_Z1	_Z2	_Z3	_Z4	_Z5	_Z6	_Z7	_Z8
1	1	2	14.4	1	1	0	0	0	0	0	0
3	1	3	27.2	1	0	1	0	0	0	0	0
5	1	4	17.2	1	0	0	1	0	0	0	0
7	1	5	18.3	1	0	0	0	1	0	0	0
9	1	6	16.2	1	0	0	0	0	1	0	0
11	1	7	18.6	1	0	0	0	0	0	1	0
13	1	8	16.4	1	0	0	0	0	0	0	1
15	2	1	15.4	1	1	0	0	0	0	0	0
17	2	3	14.8	0	1	1	0	0	0	0	0
19	2	4	18.6	0	1	0	1	0	0	0	0

Example 52.17: Linear Inference Based on Summary Data

(View the complete code for this example (gmxex17.sas) in the example repository.)

The GLIMMIX procedure has facilities for multiplicity-adjusted inference through the ADJUST= and STEPDOWN options in the ESTIMATE, LSMEANS, and LSMESTIMATE statements. You can employ these facilities to test linear hypotheses among parameters even in situations where the quantities were obtained outside the GLIMMIX procedure. This example demonstrates the process. The basic idea is to prepare a data set containing the estimates of interest and a data set containing their covariance matrix. These are then passed to the GLIMMIX procedure, preventing updating of the parameters, essentially moving directly into the post-processing stage as if estimates with this covariance matrix had been produced by the GLIMMIX procedure.

The final documentation example in Chapter 88, "The NLIN Procedure," discusses a nonlinear first-order compartment pharmacokinetic model for theophylline concentration. The data are derived by collapsing and averaging the subject-specific data from Pinheiro and Bates (1995) in a particular—yet unimportant—way that leads to two groups for comparisons. The following DATA step creates these data:

```
data theop;
   input time dose conc @@;
   if (dose = 4) then group=1; else group=2;
   datalines;
 0.00   4  0.1633  0.25   4   2.045
 0.27   4     4.4  0.30   4    7.37
 0.35   4    1.89  0.37   4    2.89
 0.50   4    3.96  0.57   4    6.57
 0.58   4     6.9  0.60   4     4.6
 0.63   4    9.03  0.77   4    5.22
 1.00   4    7.82  1.02   4   7.305
 1.05   4    7.14  1.07   4     8.6
```

```
  1.12   4    10.5  2.00   4    9.72
  2.02   4    7.93  2.05   4    7.83
  2.13   4    8.38  3.50   4    7.54
  3.52   4    9.75  3.53   4    5.66
  3.55   4   10.21  3.62   4     7.5
  3.82   4    8.58  5.02   4   6.275
  5.05   4    9.18  5.07   4    8.57
  5.08   4     6.2  5.10   4    8.36
  7.02   4    5.78  7.03   4    7.47
  7.07   4   5.945  7.08   4    8.02
  7.17   4    4.24  8.80   4    4.11
  9.00   4     4.9  9.02   4    5.33
  9.03   4    6.11  9.05   4    6.89
  9.38   4    7.14 11.60   4    3.16
 11.98   4    4.19 12.05   4    4.57
 12.10   4    5.68 12.12   4    5.94
 12.15   4     3.7 23.70   4    2.42
 24.15   4    1.17 24.17   4    1.05
 24.37   4    3.28 24.43   4    1.12
 24.65   4    1.15  0.00   5   0.025
  0.25   5    2.92  0.27   5   1.505
  0.30   5    2.02  0.50   5   4.795
  0.52   5    5.53  0.58   5    3.08
  0.98   5   7.655  1.00   5   9.855
  1.02   5    5.02  1.15   5    6.44
  1.92   5    8.33  1.98   5    6.81
  2.02   5  7.8233  2.03   5    6.32
  3.48   5    7.09  3.50   5   7.795
  3.53   5    6.59  3.57   5    5.53
  3.60   5    5.87  5.00   5     5.8
  5.02   5  6.2867  5.05   5    5.88
  6.98   5    5.25  7.00   5    4.02
  7.02   5    7.09  7.03   5   4.925
  7.15   5    4.73  9.00   5    4.47
  9.03   5    3.62  9.07   5    4.57
  9.10   5     5.9  9.22   5    3.46
 12.00   5    3.69 12.05   5    3.53
 12.10   5    2.89 12.12   5    2.69
 23.85   5    0.92 24.08   5    0.86
 24.12   5    1.25 24.22   5    1.15
 24.30   5     0.9 24.35   5    1.57
;
```

In terms of two fixed treatment groups, the nonlinear model for these data can be written as

$$C_{it} = \frac{Dk_{e_i}k_{a_i}}{Cl_i(k_{a_i} - k_{e_i})}[\exp(-k_{e_i}t) - \exp(-k_{a_i}t)] + \epsilon_{it}$$

where C_{it} is the observed concentration in group i at time t, D is the dose of theophylline, k_{e_i} is the elimination rate constant in group i, k_{a_i} is the absorption rate in group i, Cl_i is the clearance in group i, and ϵ_{it} denotes the model error. Because the rates and the clearance must be positive, you can parameterize the

model in terms of log rates and the log clearance:

$$
\begin{aligned}
Cl_i &= \exp\{\beta_{1i}\} \\
k_{a_i} &= \exp\{\beta_{2i}\} \\
k_{e_i} &= \exp\{\beta_{3i}\}
\end{aligned}
$$

In this parameterization the model contains six parameters, and the rates and clearance vary by group. The following PROC NLIN statements fit the model and obtain the group-specific parameter estimates:

```
proc nlin data=theop outest=cov;
   parms beta1_1=-3.22 beta2_1=0.47 beta3_1=-2.45
         beta1_2=-3.22 beta2_2=0.47 beta3_2=-2.45;
   if (group=1) then do;
      cl   = exp(beta1_1);
      ka   = exp(beta2_1);
      ke   = exp(beta3_1);
   end; else do;
      cl   = exp(beta1_2);
      ka   = exp(beta2_2);
      ke   = exp(beta3_2);
   end;
   mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
   model conc = mean;
   ods output ParameterEstimates=ests;
run;
```

The conditional programming statements determine the clearance, elimination, and absorption rates depending on the value of the group variable. The OUTEST= option in the PROC NLIN statement saves estimates and their covariance matrix to the data set cov. The ODS OUTPUT statement saves the "Parameter Estimates" table to the data set ests.

Output 52.17.1 displays the analysis of variance table and the parameter estimates from this NLIN run. Note that the confidence levels in the "Parameter Estimates" table are based on 92 degrees of freedom, corresponding to the residual degrees of freedom in the analysis of variance table.

Output 52.17.1 Analysis of Variance and Parameter Estimates for Nonlinear Model

The NLIN Procedure

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	6	3247.9	541.3	358.56	<.0001
Error	92	138.9	1.5097		
Uncorrected Total	98	3386.8			

Output 52.17.1 *continued*

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1_1	-3.5671	0.0864	-3.7387	-3.3956
beta2_1	0.4421	0.1349	0.1742	0.7101
beta3_1	-2.6230	0.1265	-2.8742	-2.3718
beta1_2	-3.0111	0.1061	-3.2219	-2.8003
beta2_2	0.3977	0.1987	0.00305	0.7924
beta3_2	-2.4442	0.1618	-2.7655	-2.1229

The following DATA step extracts the part of the cov data set that contains the covariance matrix of the parameter estimates in Output 52.17.1 and renames the variables as Col1–Col6. Output 52.17.2 shows the result of the DATA step.

```
data covb;
   set cov(where=(_type_='COVB'));
   rename beta1_1=col1 beta2_1=col2 beta3_1=col3
          beta1_2=col4 beta2_2=col5 beta3_2=col6;
   row = _n_;
   Parm = 1;
   keep parm row beta:;
run;

proc print data=covb;
run;
```

Output 52.17.2 Covariance Matrix of NLIN Parameter Estimates

Obs	col1	col2	col3	col4	col5	col6	row	Parm
1	0.007462	-0.005222	0.010234	0.000000	0.000000	0.000000	1	1
2	-0.005222	0.018197	-0.010590	0.000000	0.000000	0.000000	2	1
3	0.010234	-0.010590	0.015999	0.000000	0.000000	0.000000	3	1
4	0.000000	0.000000	0.000000	0.011261	-0.009096	0.015785	4	1
5	0.000000	0.000000	0.000000	-0.009096	0.039487	-0.019996	5	1
6	0.000000	0.000000	0.000000	0.015785	-0.019996	0.026172	6	1

The reason for this transformation of the data is to use the resulting data set to define a covariance structure in PROC GLIMMIX. The following statements reconstitute a model in which the parameter estimates from PROC NLIN are the observations and in which the covariance matrix of the "observations" matches the covariance matrix of the NLIN parameter estimates:

```
proc glimmix data=ests order=data;
   class Parameter;
   model Estimate = Parameter / noint df=92 s;
   random _residual_ / type=lin(1) ldata=covb v;
   parms (1) / noiter;
   lsmeans parameter / cl;
   lsmestimate Parameter
               'beta1 eq. across groups' 1 0 0 -1,
```

```
                 'beta2 eq. across groups' 0 1 0  0 -1,
                 'beta3 eq. across groups' 0 0 1  0  0  -1 /
                 adjust=bon stepdown ftest(label='Homogeneity');
run;
```

In other words, you are using PROC GLIMMIX to set up a linear statistical model

$$\mathbf{Y} = \mathbf{I}\boldsymbol{\alpha} + \boldsymbol{\epsilon}$$
$$\boldsymbol{\epsilon} \sim (\mathbf{0}, \mathbf{A})$$

where the covariance matrix $\mathbf{A}$ is given by

$$\mathbf{A} = \begin{bmatrix} 0.007 & -0.005 & 0.010 & 0 & 0 & 0 \\ -0.005 & 0.018 & -0.011 & 0 & 0 & 0 \\ 0.010 & -0.011 & 0.016 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.011 & -0.009 & 0.016 \\ 0 & 0 & 0 & -0.009 & 0.039 & -0.019 \\ 0 & 0 & 0 & 0.016 & -0.019 & 0.026 \end{bmatrix}$$

The generalized least squares estimate for $\boldsymbol{\alpha}$ in this saturated model reproduces the observations:

$$\begin{aligned} \widehat{\boldsymbol{\alpha}} &= \left(\mathbf{I}'\mathbf{A}^{-1}\mathbf{I}\right)^{-1}\mathbf{I}'\mathbf{A}^{-1}\mathbf{y} \\ &= \left(\mathbf{A}^{-1}\right)^{-1}\mathbf{A}^{-1}\mathbf{y} \\ &= \mathbf{y} \end{aligned}$$

The ORDER=DATA option in the PROC GLIMMIX statement requests that the sort order of the Parameter variable be identical to the order in which it appeared in the "Parameter Estimates" table of the NLIN procedure (Output 52.17.1). The MODEL statement uses the Estimate and Parameter variables from that table to form a model in which the $\mathbf{X}$ matrix is the identity; hence the NOINT option. The DF=92 option sets the degrees of freedom equal to the value used in the NLIN procedure. The RANDOM statement specifies a linear covariance structure with a single component and supplies the values for the structure through the LDATA= data set. This structure models the covariance matrix as $\mathrm{Var}[\mathbf{Y}] = \theta\mathbf{A}$, where the $\mathbf{A}$ matrix is given previously. Essentially, the TYPE=LIN(1) structure forces an unstructured covariance matrix onto the data. To make this work, the parameter θ is held fixed at 1 in the PARMS statement.

Output 52.17.3 displays the parameter estimates and least squares means for this model. Note that estimates and least squares means are identical, since the $\mathbf{X}$ matrix is the identity. Also, the confidence limits agree with the values reported by PROC NLIN (see Output 52.17.1).

Output 52.17.3 Parameter Estimates and LS-Means from Summary Data

The GLIMMIX Procedure

Solutions for Fixed Effects

Effect	Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|
Parameter	beta1_1	-3.5671	0.08638	92	-41.29	<.0001
Parameter	beta2_1	0.4421	0.1349	92	3.28	0.0015
Parameter	beta3_1	-2.6230	0.1265	92	-20.74	<.0001
Parameter	beta1_2	-3.0111	0.1061	92	-28.37	<.0001
Parameter	beta2_2	0.3977	0.1987	92	2.00	0.0483
Parameter	beta3_2	-2.4442	0.1618	92	-15.11	<.0001

Parameter Least Squares Means

Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
beta1_1	-3.5671	0.08638	92	-41.29	<.0001	0.05	-3.7387	-3.3956
beta2_1	0.4421	0.1349	92	3.28	0.0015	0.05	0.1742	0.7101
beta3_1	-2.6230	0.1265	92	-20.74	<.0001	0.05	-2.8742	-2.3718
beta1_2	-3.0111	0.1061	92	-28.37	<.0001	0.05	-3.2219	-2.8003
beta2_2	0.3977	0.1987	92	2.00	0.0483	0.05	0.003050	0.7924
beta3_2	-2.4442	0.1618	92	-15.11	<.0001	0.05	-2.7655	-2.1229

The (marginal) covariance matrix of the data is shown in Output 52.17.4 to confirm that it matches the **A** matrix given earlier.

Output 52.17.4 R-Side Covariance Matrix

Estimated V Matrix for Subject 1

Row	Col1	Col2	Col3	Col4	Col5	Col6
1	0.007462	-0.00522	0.01023			
2	-0.00522	0.01820	-0.01059			
3	0.01023	-0.01059	0.01600			
4				0.01126	-0.00910	0.01579
5				-0.00910	0.03949	-0.02000
6				0.01579	-0.02000	0.02617

The LSMESTIMATE statement specifies three linear functions. These set equal the β parameters from the groups. The step-down Bonferroni adjustment requests a multiplicity adjustment for the family of three tests. The FTEST option requests a joint test of the three estimable functions; it is a global test of parameter homogeneity across groups.

Output 52.17.5 displays the result from the LSMESTIMATE statement. The joint test is highly significant ($F = 30.52$, $p < 0.0001$). From the p-values associated with the individual rows of the estimates, you can see that the lack of homogeneity is due to group differences for β_1, the log clearance.

Output 52.17.5 Test of Parameter Homogeneity across Groups

Least Squares Means Estimates
Adjustment for Multiplicity: Holm

Effect	Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Adj P
Parameter	beta1 eq. across groups	-0.5560	0.1368	92	-4.06	0.0001	0.0003
Parameter	beta2 eq. across groups	0.04443	0.2402	92	0.18	0.8537	0.8537
Parameter	beta3 eq. across groups	-0.1788	0.2054	92	-0.87	0.3862	0.7725

F Test for Least Squares Means Estimates

Label	Num DF	Den DF	F Value	Pr > F
Homogeneity	3	92	30.52	<.0001

An alternative method of setting up this model is given by the following statements, where the data set pdata contains the covariance parameters:

```
random _residual_ / type=un;
parms / pdata=pdata noiter
```

The following DATA step creates an appropriate PDATA= data set from the data set covb, which was constructed earlier:

```
data pdata;
   set covb;
   array col{6};
   do i=1 to _n_;
      estimate = col{i};
      output;
   end;
   keep estimate;
run;
```

Example 52.18: Weighted Multilevel Model for Survey Data

(View the complete code for this example (gmxex18.sas) in the example repository.)

Multilevel models are a useful tool for analyzing survey data from multistage sampling designs. In multistage designs, the first-stage clusters (primary sampling units) are selected by using a probability sample from a list of first-stage units. In the second stage, second-stage clusters are selected by using a probability sample from a list of second-stage clusters for every first-stage cluster in the sample. The third and subsequent stages of clusters are selected similarly. You can use multilevel models to analyze data from multistage designs in which each stage of sampling corresponds to one level of random effects in the model. The characteristics of the units at each stage become the explanatory variables at that level. If units are drawn with unequal selection probabilities at each stage, then the unweighted estimators from standard multilevel models can be biased. Pfeffermann et al. (1998) and Rabe-Hesketh and Skrondal (2006) discuss using sampling weights to reduce this bias.

To learn how to fit a two-level model, consider a two-stage design simulation that is similar to the simulation discussed in Rabe-Hesketh and Skrondal (2006).

First, you generate a finite population of 500 level-2 units and 50 level-1 units from a two-level random intercept probit model with dichotomous responses and linear predictor,

$$\eta_{ij} = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2ij} + \gamma_i$$

where $\beta_0 = \beta_1 = \beta_2 = 1$, $\gamma_i \sim N(0, \sigma^2)$, and $\sigma^2 = 1$. The between-cluster covariate x_1i and within-cluster covariate x_2ij are generated from a Bernoulli distribution with probability 0.5.

Then, you sample subsets of level-2 units and level-1 units from a two-stage sampling design similar to the one discussed in Rabe-Hesketh and Skrondal (2006). The generated sample contains a total of 7,530 observations. The inverse selection probabilities are used as weights. To reduce the bias in the variance parameter estimator, you scale the level-1 weights by using what Pfeffermann et al. (1998) and Rabe-Hesketh and Skrondal (2006) refer to as "Method 2."

The first 30 observations of the data set are shown in Output 52.18.1. In this data set, y is the dichotomous response variable, and x1 and x2 are the between-cluster and within-cluster covariates, respectively. The variables id and w2 identify the first-stage clusters (level-2 units) and their weights, respectively. The observation units are the second-stage clusters (level-1 units), and their weights are scaled using "Method 2" in Pfeffermann et al. (1998) and Rabe-Hesketh and Skrondal (2006) and stored in sw1. The first-stage sampling weights, w2, range from 1.3359 to 4.0513, with an average of 1.6965. The second-stage sampling weights, w1, range from 1.8571 to 2.000, with an average of 1.9588. The scaled weights, sw1, range from 0.9657 to 1.0126, with an average of 1.000.

Output 52.18.1 Simulated Two-Stage Sampling Survey Data (First 30 Observations)

Obs	x1	x2	y	w2	id	sw1
1	1	0	1	1.33594	1	1.00286
2	1	0	1	1.33594	1	1.00286
3	1	1	1	1.33594	1	1.00286
4	1	1	1	1.33594	1	1.00286
5	1	0	1	1.33594	1	1.00286
6	1	1	1	1.33594	1	1.00286
7	1	1	1	1.33594	1	1.00286
8	1	1	1	1.33594	1	1.00286
9	1	0	1	1.33594	1	1.00286
10	1	0	1	1.33594	1	1.00286
11	1	1	1	1.33594	1	1.00286
12	1	0	1	1.33594	1	1.00286
13	1	1	1	1.33594	1	1.00286
14	1	0	1	1.33594	1	1.00286
15	1	0	1	1.33594	1	0.99667
16	1	1	1	1.33594	1	0.99667
17	1	0	1	1.33594	1	0.99667
18	1	0	1	1.33594	1	0.99667
19	1	0	1	1.33594	1	0.99667
20	1	1	1	1.33594	1	0.99667
21	1	0	1	1.33594	1	0.99667
22	1	0	1	1.33594	1	0.99667
23	1	0	1	1.33594	1	0.99667
24	1	0	1	1.33594	1	0.99667
25	1	1	1	1.33594	1	0.99667
26	1	1	1	1.33594	1	0.99667
27	1	0	1	1.33594	2	0.99667
28	1	0	1	1.33594	2	0.99667
29	1	1	1	1.33594	2	0.99667
30	1	1	1	1.33594	2	0.99667

The following statements fit a two-level model without using weights at any level:

```
proc glimmix data=dws method=quadrature;
   class id;
   model y = x1 x2 / dist=binomial link=probit solution;
   random int  / subject=id;
run;
```

The parameter estimates are shown in Output 52.18.2. The estimated fixed-effects parameters are all close to their corresponding true values, but the estimated covariance parameter is almost half of its true value.

Output 52.18.2 Analysis of Two-Stage Sampling Survey Data

The GLIMMIX Procedure

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	id	0.5612	0.08777

Solutions for Fixed Effects

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	1.0300	0.07608	293	13.54	<.0001
x1	1.0809	0.1186	7234	9.11	<.0001
x2	0.9934	0.06344	7234	15.66	<.0001

You can incorporate the unequal weights at both levels into the model by using the OBSWEIGHT= and WEIGHT= options as follows:

```
proc glimmix data=dws method=quadrature empirical=classical;
   class id;
   model y = x1 x2 / dist=binomial link=probit obsweight=sw1 solution;
   random int  / subject=id weight=w2;
run;
```

To fit a weighted multilevel model, you should use METHOD=QUAD. The EMPIRICAL=CLASSICAL option in the PROC GLIMMIX statement instructs PROC GLIMMIX to compute the empirical (sandwich) variance estimators for the fixed effect and the variance. The empirical variance estimators are recommended for the inference about fixed effects and variance estimated by pseudo-likelihood.

The OBSWEIGHT= option in the MODEL statement specifies the weight variable for the observation level. The WEIGHT= option in the RANDOM statement specifies the weight variable for the level that is specified by the SUBJECT= option.

The parameter estimates are shown in Output 52.18.3.

Output 52.18.3 Analysis of Two-Stage Sampling Survey Data Using Weights

The GLIMMIX Procedure

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	id	1.0106	0.2508

Solutions for Fixed Effects

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	1.0322	0.1265	497.3	8.16	<.0001
x1	1.1182	0.2146	12275	5.21	<.0001
x2	1.0074	0.07901	12275	12.75	<.0001

Note that the estimated variance component for the random id effect is much closer to the true value of 1 in the weighted analysis. When you use the raw level-2 weights and the scaled level-1 weights, the multilevel model reduces the bias in the variance parameter estimate. However, the standard errors for the weighted analysis are larger than those for the unweighted analysis.

Example 52.19: Quadrature Method for Multilevel Model

(View the complete code for this example (gmxex19.sas) in the example repository.)

In many fields, data are observed on nested units at multiple levels. For example, in an educational study, classes of students from multiple schools might be assigned to one of two programs. At the end of the program, students take an evaluation test for which they receive a grade of Pass or Fail. Such a study has a three-level structure:

1. Students are the level-1 units.
2. Classes are the level-2 clusters.
3. Schools are the level-3 clusters.

Students are nested within classes, which are further nested within schools.

The following DATA step simulates such data for 500 students from five classes that are selected from 10 schools. The observations of test results (Grade) are simulated from a logistic model, with the variable Program as a fixed effect and the variables School and Class as random effects. The random effects for School and Class are simulated with variances 16.0 and 4.0, respectively.

```
data test;
   do school = 1 to 10;
      schef = rannor(1234)*4;
      do class = 1 to 5;
         clsef = rannor(2345)*2;
         program = ranbin(12345,1,.5);
         do student = 1 to 10;
            eta = 3 + program + schef + clsef ;
            p = 1/(1+exp(-eta));
            grade = ranbin(23456,1,p);
            output;
         end;
      end;
   end;
run;
```

The following statements use a three-point adaptive quadrature to estimate this three-level model:

```
proc glimmix data=test method = quad(qpoints=3);
   class school class program;
   model grade = program/s dist=binomial link=logit solution;
   random int /subject = school;
   random int /subject = class(school);
run;
```

Output 52.19.1 shows the model information. Note that Gauss-Hermite quadrature is used for likelihood approximation.

Output 52.19.1 Model Information

The GLIMMIX Procedure

Model Information	
Data Set	WORK.TEST
Response Variable	grade
Response Distribution	Binomial
Link Function	Logit
Variance Function	Default
Variance Matrix Blocked By	school
Estimation Technique	Maximum Likelihood
Likelihood Approximation	Gauss-Hermite Quadrature
Degrees of Freedom Method	Containment

The covariance parameter estimates and the solutions for fixed effects are shown in Output 52.19.2.

Output 52.19.2 Parameter Estimates

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	school	3.7342	3.1391
Intercept	class(school)	6.2835	2.5338

Solutions for Fixed Effects

Effect	program	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		2.6014	0.9512	9	2.73	0.0230
program	0	-0.2056	0.9009	450	-0.23	0.8195
program	1	0	.	.	.	.

For this model, the number of random effects within each school is $(1 + 5) = 6$: one for school random intercept and five for random intercepts for classes that are nested within the school. This means that the marginal log-likelihood computation requires the numerical evaluation of a six-dimensional integral. With the number of quadrature points set to three by the QPOINTS=3 suboption, this numerical evaluation computes $3^6 = 729$ conditional log likelihoods for each observation on each pass through the data.

The number of conditional log likelihoods increases exponentially with the number of random effects. For example, suppose that you increase the number of classes from five to ten. Then the number of conditional log-likelihood evaluations becomes $3^{(1+10)} = 177{,}147$ for each observation, where 1 is the number of random school intercepts and 10 is the number of random intercepts for classes that are nested within each school. In many applications, it is not uncommon to have more than 10 nested units within a higher-level unit. As the number of nested units increases, the number of nested random effects increases, driving the exponential growth of the computational requirement.

To address the intimidating computational demand for dealing with such multilevel models, Pinheiro and Chao (2006) propose a multilevel adaptive quadrature algorithm. You can use the FASTQUAD suboption in

the METHOD=QUAD option to invoke this algorithm. The following DATA step simulates the data set for 10 schools:

```
data test;
   do school = 1 to 10;
      schef = rannor(1234)*4;
      do class = 1 to 10;
         clsef = rannor(2345)*2;
         program = ranbin(12345,1,.5);
         do student = 1 to 10;
            eta = 3 + program + schef + clsef ;
            p = 1/(1+exp(-eta));
            grade = ranbin(23456,1,p);
            output;
         end;
      end;
   end;
run;
```

The following statements use the FASTQUAD suboption to request multilevel quadrature estimation for the model:

```
proc glimmix data=test method = quad(qpoints=3 fastquad);
   class school class program;
   model grade = program/s dist=binomial link=logit solution;
   random int /subject = school;
   random int /subject = class(school);
run;
```

The "Model Information" table in Output 52.19.3 shows that multilevel Gauss-Hermite quadrature is used for likelihood approximation.

Output 52.19.3 Model Information

The GLIMMIX Procedure

Model Information	
Data Set	WORK.TEST
Response Variable	grade
Response Distribution	Binomial
Link Function	Logit
Variance Function	Default
Variance Matrix Blocked By	school
Estimation Technique	Maximum Likelihood
Likelihood Approximation	Multilevel Gauss-Hermite Quadrature
Degrees of Freedom Method	Containment

Output 52.19.4 displays the parameter estimates.

Output 52.19.4 Parameter Estimates

Covariance Parameter Estimates

Cov Parm	Subject	Estimate	Standard Error
Intercept	school	13.4910	8.5572
Intercept	class(school)	2.9375	1.1371

Solutions for Fixed Effects

Effect	program	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		4.9024	1.4353	9	3.42	0.0077
program	0	0.03780	0.6157	900	0.06	0.9511
program	1	0	.	.	.	.

In multilevel quadrature, the number of random effects within each school is $(1 + 1) = 2$: one for the school random intercept and one for the class random intercept. That is, the number of class random intercepts does not grow with the number of classes. With three quadrature points, this leads to $3^{(1+1)} = 9$ evaluations of conditional log likelihoods for any number of classes in this example. In addition to reducing computational demand, multilevel adaptive quadrature also reduces memory usage. Both the computational efficiency and the memory-usage efficiency that the FASTQUAD suboption affords enables you to handle much larger multilevel models.

Example 52.20: Comparison of Predictive Margins in Multicenter Study

(View the complete code for this example (gmxex20.sas) in the example repository.)

It is often useful to compare the average response rates of treatment groups in clinical trials. This example shows how to compute predictive margins to account for covariates distributions in such a comparison.

Consider a multicenter study that investigates the performance of two treatments. In this study, ten treatment centers are randomly selected for inclusion. At each center, patients are randomly assigned to treatment *A* or treatment *B*. One of the study goals is to compare the response rates of the treatments.

The data set multicenter, created in the following DATA step, has five variables. The variable trt identifies the two treatments. The variable marker takes a value of 1 if the patient is biomarker-positive, and 0 if the patient is biomarker-negative. The response variable is 1 if the patient responds to the treatment, and 0 if the patient does not.

```
data multicenter;
   input center trt$ age marker response @@;
   datalines;
 1 A  15  0  0     1 A  28  0  0
 1 A  60  0  1     1 A  68  1  1
 1 A  23  1  0     1 A  33  1  1
 1 A  30  1  0     1 A  73  1  1
 1 A  15  1  0     1 A  34  1  0
 1 A  15  1  0     1 A  68  1  1
 1 B  53  0  1     1 B  62  0  1
 1 B  15  0  0     1 B  28  1  0
```

1	B	27	1	0	1	B	45	1	0
1	B	56	1	1	1	B	24	1	0
1	B	42	1	0	1	B	61	1	0
1	B	15	1	0	1	B	67	1	1
2	A	28	0	1	2	A	43	0	1
2	A	52	0	1	2	A	49	1	1
2	A	59	1	1	2	A	32	1	1
2	A	50	1	1	2	A	41	1	1
2	A	21	1	0	2	A	62	1	1
2	A	77	1	1	2	A	70	1	1
2	B	79	0	1	2	B	49	0	1
2	B	73	0	1	2	B	73	1	1
2	B	78	1	1	2	B	61	1	1
2	B	78	1	1	2	B	54	1	1
2	B	51	1	1	2	B	50	1	1
2	B	17	1	0	2	B	15	1	0
3	A	20	0	0	3	A	18	0	1
3	A	32	0	1	3	A	55	1	1
3	A	51	1	1	3	A	58	1	1
3	A	36	1	1	3	A	15	1	0
3	A	21	1	0	3	A	24	1	1
3	A	40	1	1	3	A	28	1	0
3	B	54	0	1	3	B	64	0	1
3	B	15	0	0	3	B	15	1	0
3	B	76	1	1	3	B	39	1	1
3	B	48	1	0	3	B	34	1	0
3	B	17	1	0	3	B	37	1	0
3	B	35	1	0	3	B	30	1	1
4	A	74	0	1	4	A	78	0	1
4	A	15	1	0	4	A	41	1	1
4	A	25	1	0	4	A	31	1	0
4	A	22	1	0	4	A	15	1	0
4	B	77	0	1	4	B	33	0	0
4	B	21	1	0	4	B	15	1	0
4	B	70	1	1	4	B	54	1	0
4	B	15	1	0	4	B	15	1	0
5	A	21	0	1	5	A	72	0	1
5	A	28	1	1	5	A	58	1	1
5	A	39	1	1	5	A	41	1	1
5	A	35	1	0	5	A	59	1	1
5	B	73	0	1	5	B	15	0	0
5	B	46	1	1	5	B	59	1	1
5	B	20	1	0	5	B	66	1	1
5	B	24	1	0	5	B	15	1	0
6	A	25	0	0	6	A	15	0	0
6	A	63	1	1	6	A	15	1	0
6	A	61	1	1	6	A	73	1	1
6	A	35	1	0	6	A	76	1	1
6	B	76	0	1	6	B	15	0	0
6	B	15	1	0	6	B	15	1	0
6	B	15	1	0	6	B	47	1	0
6	B	77	1	1	6	B	17	1	0
7	A	27	0	0	7	A	18	0	0
7	A	18	0	1	7	A	62	1	1

```
 7 A  30  1  0     7 A  25  1  0
 7 A  40  1  0     7 A  18  1  0
 7 A  16  1  0     7 A  17  1  0
 7 A  43  1  1     7 A  24  1  0
 7 B  64  0  1     7 B  29  0  0
 7 B  23  0  0     7 B  20  1  0
 7 B  36  1  0     7 B  46  1  0
 7 B  19  1  0     7 B  41  1  0
 7 B  23  1  0     7 B  40  1  1
 7 B  37  1  1     7 B  22  1  0
 8 A  51  0  1     8 A  85  0  1
 8 A  16  1  1     8 A  37  1  1
 8 A  28  1  0     8 A  24  1  0
 8 A  25  1  0     8 A  17  1  0
 8 B  54  0  1     8 B  21  0  1
 8 B  23  1  0     8 B  20  1  0
 8 B  19  1  0     8 B  18  1  0
 8 B  21  1  0     8 B  22  1  0
 9 A  22  0  0     9 A  23  0  0
 9 A  33  1  0     9 A  68  1  1
 9 A  42  1  1     9 A  51  1  1
 9 A  16  1  0     9 A  52  1  1
 9 B  77  0  1     9 B  16  0  0
 9 B  42  1  0     9 B  59  1  1
 9 B  28  1  0     9 B  33  1  0
 9 B  31  1  0     9 B  20  1  0
10 A  30  0  1    10 A  24  0  1
10 A  18  1  0    10 A  16  1  0
10 A  27  1  1    10 A  29  1  1
10 A  40  1  1    10 A  64  1  1
10 B  32  0  1    10 B  25  0  0
10 B  19  1  0    10 B  22  1  0
10 B  23  1  0    10 B  38  1  0
10 B  18  1  1    10 B  23  1  0
;
```

For patient j at center i, the logistic regression model is

$$
\begin{aligned}
\log\left\{\frac{\pi_{ij}}{1-\pi_{ij}}\right\} &= \beta_0 + \beta_A I(\text{trt} = \text{A}) + \beta_B I(\text{trt} = \text{B}) \\
&\quad + \beta_+ I(\text{marker} = 1) + \beta_- I(\text{marker} = 0) \\
&\quad + \beta_{A+} I(\text{trt} = \text{A}, \text{marker} = 1) + \beta_{A-} I(\text{trt} = \text{A}, \text{marker} = 0) \\
&\quad + \beta_{B+} I(\text{trt} = \text{B}, \text{marker} = 1) + \beta_{B-} I(\text{trt} = \text{B}, \text{marker} = 0) \\
&\quad + \beta_1 \text{Age}_{ij} + \gamma_i
\end{aligned}
$$

where π_{ij} is the probability that patient j at center i has responded to the treatment; γ_i is the random effect for center i; I is the indicator function; β_A and β_B are the effects of treatments A and B, respectively; β_+ and β_- are the effects of a positive biomarker and a negative biomarker, respectively; and finally, β_{A+}, β_{A-}, β_{B+}, and β_{B-} are the interaction effects of trt and marker.

Given this model, the predictive margin for treatment A is

$$\frac{1}{N}\sum_{i=1}^{10}\sum_{j=1}^{n_i} g^{-1}(\hat{\beta}_0 + \hat{\beta}_A + \hat{\beta}_+ I(\text{marker} = 1) + \hat{\beta}_- I(\text{marker} = 0) \\ + \hat{\beta}_1 \text{Age}_{ij} + \hat{\beta}_{A+} I(\text{marker} = 1) + \hat{\beta}_{A-} I(\text{marker} = 0))$$

where n_i is the number of patients at treatment center i, $N = \sum_{i=1}^{10} n_i$ is the number of patients in the study, and $g^{-1}(\cdot)$ is the inverse logistic link function. The predictive margin for treatment A is the average predicted response rate if all patients receive treatment A.

Similarly, the predictive margin for treatment B is

$$\frac{1}{N}\sum_{i=1}^{6}\sum_{j=1}^{n_i} g^{-1}(\hat{\beta}_0 + \hat{\beta}_B + \hat{\beta}_+ I(\text{marker} = 1) + \hat{\beta}_- I(\text{marker} = 0) \\ + \hat{\beta}_1 \text{Age}_{ij} + \hat{\beta}_{B+} I(\text{marker} = 1) + \hat{\beta}_{B-} I(\text{marker} = 0))$$

This is the average predicted response rate if all patients receive treatment B.

The following statements fit the multicenter data to the logistic regression model:

```
proc glimmix data=multicenter;
   class center trt marker;
   model response = trt|marker age/s dist=binary link=logit;
   random  intercept/ subject=center;
   margins trt/ diff;
   margins trt*marker/ sliceby=marker slicediff;
run;
```

The first MARGINS statement requests predictive margins for the two treatment groups. The DIFF option compares average treatment response rates that control for the age and biomarker distributions.

Output 52.20.1 shows the predictive margins for treatment A and treatment B.

Output 52.20.1 Treatment Predictive Margins

The GLIMMIX Procedure

trt Margins

trt	Estimate	Standard Error	DF	t Value	Pr > |t|
A	0.4481	0.05349	178	8.38	<.0001
B	0.6296	0.04276	178	14.72	<.0001

Output 52.20.2 shows the test of the difference between the two treatment predictive margins.

Output 52.20.2 Difference of Treatment Margins

Differences of trt Margins

trt	_trt	Estimate	Standard Error	DF	t Value	Pr > |t|
A	B	-0.1815	0.04689	178	-3.87	0.0002

The degrees of freedom in Output 52.20.1 and Output 52.20.2 are the same as those in the "Type III Tests of Fixed Effect" table (not shown). Based on the *p*-value, you would conclude that at the 0.05 level the average response rates of treatment *A* and treatment *B* are significantly different.

The second MARGINS statement requests predictive margins for the trt*marker interaction. The results are shown in Output 52.20.3.

Output 52.20.3 trt*marker Predictive Margins

trt*marker Margins						
trt	marker	Estimate	Standard Error	DF	t Value	Pr > |t|
A	0	0.3256	0.08589	178	3.79	0.0002
A	1	0.4847	0.05322	178	9.11	<.0001
B	0	0.4913	0.08996	178	5.46	<.0001
B	1	0.6612	0.04152	178	15.93	<.0001

The four predictive margins in Output 52.20.3 are plotted in Output 52.20.4.

Output 52.20.4 Predictive Margins for trt*marker

To test the significance of the treatment difference within each biomarker group, you can use the SLICEBY option, which slices the trt*marker interaction by the marker effect. The SLICEDIFF option then compares the sliced predictive margins for the biomarker-positive group and the biomarker-negative group separately.

Output 52.20.5 shows the test of the treatment margin difference for the biomarker-negative group.

Output 52.20.5 Treatment Margins Difference in Biomarker-Negative Group

Differences of trt*marker Margins Sliced by marker							
Slice	trt	_trt	Estimate	Standard Error	DF	t Value	Pr > |t|
marker 0	A	B	-0.1657	0.1101	178	-1.50	0.1343

Output 52.20.6 shows the test of the treatment margin difference for the biomarker-positive group.

Output 52.20.6 Treatment Margins Difference in Biomarker-Positive Group

Differences of trt*marker Margins Sliced by marker							
Slice	trt	_trt	Estimate	Standard Error	DF	t Value	Pr > |t|
marker 1	A	B	-0.1765	0.05013	178	-3.52	0.0005

Output 52.20.6 shows that the average response rates are significantly different between treatment *A* and treatment *B* in the biomarker-positive group (p-value of 0.0005) whereas the average response rates are not significantly different in the biomark-negative group (p-value of 0.13, Output 52.20.5).

References

Abramowitz, M., and Stegun, I. A., eds. (1972). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* 10th printing. New York: Dover.

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Bahadur, R. R. (1961). "A Representation of the Joint Distribution of Responses to n Dichotomous Items." In *Studies in Item Analysis and Prediction*, edited by H. Solomon, 158–168. Stanford, CA: Stanford University Press.

Beale, E. M. L. (1972). "A Derivation of Conjugate Gradients." In *Numerical Methods for Nonlinear Optimization*, edited by F. A. Lootsma, 39–43. London: Academic Press.

Bell, R. M., and McCaffrey, D. F. (2002). "Bias Reduction in Standard Errors for Linear Regression with Multi-stage Samples." *Survey Methodology* 28:169–181.

Bickel, P. J., and Doksum, K. A. (1977). *Mathematical Statistics*. San Francisco: Holden-Day.

Booth, J. G., and Hobert, J. P. (1998). "Standard Errors of Prediction in Generalized Linear Mixed Models." *Journal of the American Statistical Association* 93:262–272.

Bozdogan, H. (1987). "Model Selection and Akaike's Information Criterion (AIC): The General Theory and Its Analytical Extensions." *Psychometrika* 52:345–370.

Breslow, N. E., and Clayton, D. G. (1993). "Approximate Inference in Generalized Linear Mixed Models." *Journal of the American Statistical Association* 88:9–25.

Breslow, N. E., and Lin, X. (1995). "Bias Correction in Generalised Linear Mixed Models with a Single Component of Dispersion." *Biometrika* 81:81–91.

Brinkman, N. D. (1981). "Ethanol Fuel: A Single-Cylinder Engine Study of Efficiency and Exhaust Emissions." *Society of Automotive Engineers Transactions* 90:1410–1424.

Brown, H., and Prescott, R. (1999). *Applied Mixed Models in Medicine*. New York: John Wiley & Sons.

Burdick, R. K., and Graybill, F. A. (1992). *Confidence Intervals on Variance Components*. New York: Marcel Dekker.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Cameron, A. C., and Trivedi, P. K. (1998). *Regression Analysis of Count Data*. Cambridge: Cambridge University Press.

Chang, I.-M., Gelman, R., and Pagano, M. (1982). "Corrected Group Prognostic Curves and Summary Statistics." *Journal of Chronic Diseases* 35:669–674.

Clayton, D. G., and Kaldor, J. (1987). "Empirical Bayes Estimates of Age-Standardized Relative Risks for Use in Disease Mapping." *Biometrics* 43:671–681.

Cleveland, W. S., and Grosse, E. (1991). "Computational Methods for Local Regression." *Statistics and Computing* 1:47–62.

Cockerham, C. C., and Weir, B. S. (1977). "Quadratic Analyses of Reciprocal Crosses." *Biometrics* 33:187–203.

Davis, A. W. (1977). "A Differential Equation Approach to Linear Combinations of Independent Chi-Squares." *Journal of the American Statistical Association* 72:212–214.

De Boor, C. (2001). *A Practical Guide to Splines*. Rev. ed. New York: Springer-Verlag.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981). "An Adaptive Nonlinear Least-Squares Algorithm." *ACM Transactions on Mathematical Software* 7:348–368.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Dennis, J. E., and Schnabel, R. B. (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Englewood Cliffs, NJ: Prentice-Hall.

Diggle, P. J., Liang, K.-Y., and Zeger, S. L. (1994). *Analysis of Longitudinal Data*. Oxford: Clarendon Press.

Dunnett, C. W. (1980). "Pairwise Multiple Comparisons in the Unequal Variance Case." *Journal of the American Statistical Association* 75:796–800.

Edwards, D., and Berry, J. J. (1987). "The Efficiency of Simulation-Based Multiple Comparisons." *Biometrics* 43:913–928.

Eilers, P. H. C., and Marx, B. D. (1996). "Flexible Smoothing with B-Splines and Penalties." *Statistical Science* 11:89–121. With discussion.

Eskow, E., and Schnabel, R. B. (1991). "Algorithm 695: Software for a New Modified Cholesky Factorization." *ACM Transactions on Mathematical Software* 17:306–312.

Evans, G. (1993). *Practical Numerical Integration*. New York: John Wiley & Sons.

Fai, A. H. T., and Cornelius, P. L. (1996). "Approximate F-Tests of Multiple Degree of Freedom Hypotheses in Generalized Least Squares Analyses of Unbalanced Split-Plot Experiments." *Journal of Statistical Computation and Simulation* 54:363–378.

Fay, M. P., and Graubard, B. I. (2001). "Small-Sample Adjustments for Wald-Type Tests Using Sandwich Estimators." *Biometrics* 57:1198–1206.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). "Beta Regression for Modelling Rates and Proportions." *Journal of Applied Statistics* 31:799–815.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Friedman, J. H., Bentley, J. L., and Finkel, R. A. (1977). "An Algorithm for Finding Best Matches in Logarithmic Expected Time." *ACM Transactions on Mathematical Software* 3:209–226.

Fuller, W. A. (1976). *Introduction to Statistical Time Series*. New York: John Wiley & Sons.

Games, P. A., and Howell, J. F. (1976). "Pairwise Multiple Comparison Procedures with Unequal *n*'s and/or Variances: A Monte Carlo Study." *Journal of Educational Statistics* 1:113–125.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Giesbrecht, F. G., and Burns, J. C. (1985). "Two-Stage Analysis Based on a Mixed Model: Large-Sample Asymptotic Theory and Small-Sample Simulation Results." *Biometrics* 41:477–486.

Gilliland, D., and Schabenberger, O. (2001). "Limits on Pairwise Association for Equi-correlated Binary Variables." *Journal of Applied Statistical Sciences* 10:279–285.

Gilmour, A. R., Anderson, R. D., and Rae, A. L. (1987). "Variance Components on an Underlying Scale for Ordered Multiple Threshold Categorical Data Using a Generalized Linear Mixed Model." *Journal of Animal Breeding and Genetics* 104:149–155.

Golub, G. H., and Welsch, J. H. (1969). "Calculation of Gaussian Quadrature Rules." *Mathematical Computing* 23:221–230.

Goodnight, J. H. (1978a). *Computing MIVQUE0 Estimates of Variance Components*. Technical Report R-105, SAS Institute Inc., Cary, NC.

Goodnight, J. H. (1978b). *Tests of Hypotheses in Fixed-Effects Linear Models*. Technical Report R-101, SAS Institute Inc., Cary, NC.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Goodnight, J. H., and Hemmerle, W. J. (1979). "A Simplified Algorithm for the W-Transformation in Variance Component Estimation." *Technometrics* 21:265–267.

Gotway, C. A., and Stroup, W. W. (1997). "A Generalized Linear Model Approach to Spatial Data and Prediction." *Journal of Agricultural, Biological, and Environmental Statistics* 2:157–187.

Guirguis, G. H., and Tobias, R. D. (2004). "On the Computation of the Distribution for the Analysis of Means." *Communications in Statistics—Simulation and Computation* 33:861–888.

Hand, D. J., Daly, F., Lunn, A. D., McConway, K. J., and Ostrowski, E. (1994). *A Handbook of Small Data Sets*. London: Chapman & Hall.

Handcock, M. S., and Stein, M. L. (1993). "A Bayesian Analysis of Kriging." *Technometrics* 35:403–410.

Handcock, M. S., and Wallis, J. R. (1994). "An Approach to Statistical Spatial-Temporal Modeling of Meteorological Fields." *Journal of the American Statistical Association* 89:368–390. With discussion.

Hannan, E. J., and Quinn, B. G. (1979). "The Determination of the Order of an Autoregression." *Journal of the Royal Statistical Society, Series B* 41:190–195.

Harville, D. A., and Jeske, D. R. (1992). "Mean Squared Error of Estimation or Prediction under a General Linear Model." *Journal of the American Statistical Association* 87:724–731.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Hemmerle, W. J., and Hartley, H. O. (1973). "Computing Maximum Likelihood Estimates for the Mixed AOV Model Using the W-Transformation." *Technometrics* 15:819–831.

Henderson, C. R. (1984). *Applications of Linear Models in Animal Breeding*. Guelph, ON: University of Guelph.

Hinkley, D. V. (1977). "Jackknifing in Unbalanced Situations." *Technometrics* 19:285–292.

Hirotsu, C., and Srivastava, M. (2000). "Simultaneous Confidence Intervals Based on One-Sided Max t Test." *Statistics and Probability Letters* 49:25–37.

Holm, S. (1979). "A Simple Sequentially Rejective Multiple Test Procedure." *Scandinavian Journal of Statistics* 6:65–70.

Hsu, J. C. (1992). "The Factor Analytic Approach to Simultaneous Inference in the General Linear Model." *Journal of Computational and Graphical Statistics* 1:151–168.

Hsu, J. C. (1996). *Multiple Comparisons: Theory and Methods*. London: Chapman & Hall.

Hsu, J. C., and Peruggia, M. (1994). "Graphical Representation of Tukey's Multiple Comparison Method." *Journal of Computational and Graphical Statistics* 3:143–161.

Huber, P. J. (1967). "The Behavior of Maximum Likelihood Estimates under Nonstandard Conditions." *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* 1:221–233.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Huynh, H., and Feldt, L. S. (1970). "Conditions Under Which Mean Square Ratios in Repeated Measurements Designs Have Exact F-Distributions." *Journal of the American Statistical Association* 65:1582–1589.

Jennrich, R. I., and Schluchter, M. D. (1986). "Unbalanced Repeated-Measures Models with Structured Covariance Matrices." *Biometrics* 42:805–820.

Joe, H., and Zhu, R. (2005). "Generalized Poisson Distribution: The Property of Mixture of Poisson and Comparison with Negative Binomial Distribution." *Biometrical Journal* 47:219–229.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1994). *Continuous Univariate Distributions.* 2nd ed. Vol. 1. New York: John Wiley & Sons.

Kackar, R. N., and Harville, D. A. (1984). "Approximations for Standard Errors of Estimators of Fixed and Random Effects in Mixed Linear Models." *Journal of the American Statistical Association* 79:853–862.

Kahaner, D., Moler, C. B., and Nash, S. (1989). *Numerical Methods and Software*. Englewood Cliffs, NJ: Prentice-Hall.

Karim, M. R., and Zeger, S. L. (1992). "Generalized Linear Models with Random Effects; Salamander Mating Revisited." *Biometrics* 48:631–644.

Kass, R. E., and Steffey, D. (1989). "Approximate Bayesian Inference in Conditionally Independent Hierarchical Models (Parametric Empirical Bayes Models)." *Journal of the American Statistical Association* 84:717–726.

Kauermann, G., and Carroll, R. J. (2001). "A Note on the Efficiency of Sandwich Covariance Estimation." *Journal of the American Statistical Association* 96:1387–1396.

Kenward, M. G. (1987). "A Method for Comparing Profiles of Repeated Measurements." *Journal of the Royal Statistical Society, Series C* 36:296–308.

Kenward, M. G., and Roger, J. H. (1997). "Small Sample Inference for Fixed Effects from Restricted Maximum Likelihood." *Biometrics* 53:983–997.

Kenward, M. G., and Roger, J. H. (2009). "An Improved Approximation to the Precision of Fixed Effects from Restricted Maximum Likelihood." *Computational Statistics and Data Analysis* 53:2583–2595.

Koch, G. G., Carr, G. J., Amara, I. A., Stokes, M. E., and Uryniak, T. J. (1990). "Categorical Data Analysis." In *Statistical Methodology in the Pharmaceutical Sciences*, edited by D. A. Berry, 391–475. New York: Marcel Dekker.

Kramer, C. Y. (1956). "Extension of Multiple Range Tests to Group Means with Unequal Numbers of Replications." *Biometrics* 12:307–310.

Lane, P. W., and Nelder, J. A. (1982). "Analysis of Covariance and Standardization as Instances of Prediction." *Biometrics* 38:613–621.

Lange, K. (1999). *Numerical Analysis for Statisticians*. New York: Springer-Verlag.

Liang, K.-Y., and Zeger, S. L. (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika* 73:13–22.

Lin, X., and Breslow, N. E. (1996). "Bias Correction in Generalized Linear Mixed Models with Multiple Components of Dispersion." *Journal of the American Statistical Association* 91:1007–1016.

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and Schabenberger, O. (2006). *SAS for Mixed Models*. 2nd ed. Cary, NC: SAS Institute Inc.

Long, J. S., and Ervin, L. H. (2000). "Using Heteroscedasticity Consistent Standard Errors in the Linear Regression Model." *American Statistician* 54:217–224.

Macchiavelli, R. E., and Arnold, S. F. (1994). "Variable Order Ante-dependence Models." *Communications in Statistics—Theory and Methods* 23:2683–2699.

MacKinnon, J. G., and White, H. (1985). "Some Heteroskedasticity-Consistent Covariance Matrix Estimators with Improved Finite Sample Properties." *Journal of Econometrics* 29:305–325.

Mancl, L. A., and DeRouen, T. A. (2001). "A Covariance Estimator for GEE with Improved Small-Sample Properties." *Biometrics* 57:126–134.

Matérn, B. (1986). *Spatial Variation*. 2nd ed. New York: Springer-Verlag.

McCullagh, P. (1980). "Regression Models for Ordinal Data." *Journal of the Royal Statistical Society, Series B* 42:109–142.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

McLean, R. A., and Sanders, W. L. (1988). "Approximating Degrees of Freedom for Standard Errors in Mixed Linear Models." In *Proceedings of the Statistical Computing Section*, 50–59. Alexandria, VA: American Statistical Association.

McLean, R. A., Sanders, W. L., and Stroup, W. W. (1991). "A Unified Approach to Mixed Linear Models." *American Statistician* 45:54–64.

Milliken, G. A., and Johnson, D. E. (1992). *Designed Experiments.* Vol. 1 of Analysis of Messy Data. Reprint edition. New York: Chapman & Hall.

Moré, J. J. (1978). "The Levenberg-Marquardt Algorithm: Implementation and Theory." In *Lecture Notes in Mathematics*, vol. 30, edited by G. A. Watson, 105–116. Berlin: Springer-Verlag.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Morel, J. G. (1989). "Logistic Regression under Complex Survey Designs." *Survey Methodology* 15:203–223.

Morel, J. G., Bokossa, M. C., and Neerchal, N. K. (2003). "Small Sample Correction for the Variance of GEE Estimators." *Biometrical Journal* 4:395–409.

Moriguchi, S., ed. (1976). *Statistical Method for Quality Control*. Tokyo: Japan Standards Association. In Japanese.

Mosteller, F., and Tukey, J. W. (1977). *Data Analysis and Regression*. Reading, MA: Addison-Wesley.

Murray, D. M., Varnell, S. P., and Blitstein, J. L. (2004). "Design and Analysis of Group-Randomized Trials: A Review of Recent Methodological Developments." *American Journal of Public Health* 94:423–432.

National Institute of Standards and Technology (1998). "Statistical Reference Data Sets." Accessed June 6, 2011. http://www.itl.nist.gov/div898/strd/general/dataarchive.html.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Nelson, P. R. (1982). "Exact Critical Points for the Analysis of Means." *Communications in Statistics—Theory and Methods* 11:699–709.

Nelson, P. R. (1991). "Numerical Evaluation of Multivariate Normal Integrals with Correlations $\rho_{lj} = -\alpha_l \alpha_j$." In *Frontiers of Statistical Scientific Theory and Industrial Applications: Proceedings of the ICOSCO I Conference*, edited by A. Öztürk and E. C. van der Meulen, 97–114. Columbus, OH: American Sciences Press.

Nelson, P. R. (1993). "Additional Uses for the Analysis of Means and Extended Tables of Critical Values." *Technometrics* 35:61–71.

Ott, E. R. (1967). "Analysis of Means: A Graphical Procedure." *Industrial Quality Control* 24:101–109. Reprinted in *Journal of Quality Technology* 15 (1983): 10–18.

Patel, H. I. (1991). "Analysis of Incomplete Data from a Clinical Trial with Repeated Measurements." *Biometrika* 78:609–619.

Pawitan, Y. (2001). *In All Likelihood: Statistical Modelling and Inference Using Likelihood*. Oxford: Clarendon Press.

Pfeffermann, D., Skinner, C. J., Holmes, D. J., Goldstein, H., and Rasbash, J. (1998). "Weighting for Unequal Selection Probabilities in Multilevel Models." *Journal of the Royal Statistical Society, Series B* 60:23–40.

Pinheiro, J. C., and Bates, D. M. (1995). "Approximations to the Log-Likelihood Function in the Nonlinear Mixed-Effects Model." *Journal of Computational and Graphical Statistics* 4:12–35.

Pinheiro, J. C., and Chao, E. C. (2006). "Efficient Laplacian and Adaptive Gaussian Quadrature Algorithms for Multilevel Generalized Linear Mixed Models." *Journal of Computational and Graphical Statistics* 15:58–81.

Polak, E. (1971). *Computational Methods in Optimization*. New York: Academic Press.

Pothoff, R. F., and Roy, S. N. (1964). "A Generalized Multivariate Analysis of Variance Model Useful Especially for Growth Curve Problems." *Biometrika* 51:313–326.

Powell, M. J. D. (1977). "Restart Procedures for the Conjugate Gradient Method." *Mathematical Programming* 12:241–254.

Prasad, N. G. N., and Rao, J. N. K. (1990). "The Estimation of Mean Squared Error of Small-Area Estimators." *Journal of the American Statistical Association* 85:163–171.

Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.

Rabe-Hesketh, S., and Skrondal, A. (2006). "Multilevel Modelling of Complex Survey Data." *Journal of the Royal Statistical Society, Series A* 169:805–827.

Raudenbush, S. M., Yang, M.-L., and Yosef, M. (2000). "Maximum Likelihood for Generalized Linear Models with Nested Random Effects via Higher-Order, Multivariate Laplace Approximation." *Journal of Computational and Graphical Statistics* 9:141–157.

Royen, T. (1989). "Generalized Maximum Range Tests for Pairwise Comparisons of Several Populations." *Biometrical Journal* 31:905–929.

Ruppert, D., Wand, M. P., and Carroll, R. J. (2003). *Semiparametric Regression*. Cambridge: Cambridge University Press.

Saxton, A., ed. (2004). *Genetic Analysis of Complex Traits Using SAS*. Cary, NC: SAS Institute Inc.

Schabenberger, O., and Gregoire, T. G. (1996). "Population-Averaged and Subject-Specific Approaches for Clustered Categorical Data." *Journal of Statistical Computation and Simulation* 54:231–253.

Schabenberger, O., Gregoire, T. G., and Kong, F. (2000). "Collections of Simple Effects and Their Relationship to Main Effects and Interactions in Factorials." *American Statistician* 54:210–214.

Schabenberger, O., and Pierce, F. J. (2002). *Contemporary Statistical Models for the Plant and Soil Sciences*. Boca Raton, FL: CRC Press.

Schall, R. (1991). "Estimation in Generalized Linear Models with Random Effects." *Biometrika* 78:719–727.

Schluchter, M. D., and Elashoff, J. D. (1990). "Small-Sample Adjustments to Tests with Unbalanced Repeated Measures Assuming Several Covariance Structures." *Journal of Statistical Computation and Simulation* 37:69–87.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Self, S. G., and Liang, K.-Y. (1987). "Asymptotic Properties of Maximum Likelihood Estimators and Likelihood Ratio Tests under Nonstandard Conditions." *Journal of the American Statistical Association* 82:605–610.

Shaffer, J. P. (1986). "Modified Sequentially Rejective Multiple Test Procedures." *Journal of the American Statistical Association* 81:826–831.

Shapiro, A. (1988). "Towards a Unified Theory of Inequality Constrained Testing in Multivariate Analysis." *International Statistical Review* 56:49–62.

Shun, Z. (1997). "Another Look at the Salamander Mating Data: A Modified Laplace Approximation Approach." *Journal of the American Statistical Association* 92:341–349.

Shun, Z., and McCullagh, P. (1995). "Laplace Approximation of High Dimensional Integrals." *Journal of the Royal Statistical Society, Series B* 57:749–760.

Silvapulle, M. J., and Sen, P. K. (2004). *Constrained Statistical Inference: Order, Inequality, and Shape Constraints*. New York: John Wiley & Sons.

Silvapulle, M. J., and Silvapulle, P. (1995). "A Score Test against One-Sided Alternatives." *Journal of the American Statistical Association* 90:342–349.

Simpson, S. L., Edwards, L. J., Muller, K. E., Sen, P. K., and Styner, M. A. (2010). "A Linear Exponent AR(1) Family of Correlation Structures." *Statistics in Medicine* 29:1825–1838.

Stenstrom, F. H. (1940). "The Growth of Snapdragons, Stocks, Cinerarias, and Carnations on Six Iowa Soils." Master's thesis, Iowa State College.

Stram, D. O., and Lee, J. W. (1994). "Variance Components Testing in the Longitudinal Mixed Effects Model." *Biometrics* 50:1171–1177.

Stram, D. O., and Lee, J. W. (1995). "Correction to 'Variance Components Testing in the Longitudinal Mixed Effects Model'." *Biometrics* 51:1196.

Tamhane, A. C. (1979). "A Comparison of Procedures for Multiple Comparisons of Means with Unequal Variances." *Journal of the American Statistical Association* 74:471–480.

Thall, P. F., and Vail, S. C. (1990). "Some Covariance Models for Longitudinal Count Data with Overdispersion." *Biometrics* 46:657–671.

Verbeke, G., and Molenberghs, G. (2000). *Linear Mixed Models for Longitudinal Data*. New York: Springer.

Verbeke, G., and Molenberghs, G. (2003). "The Use of Score Tests for Inference on Variance Components." *Biometrics* 59:254–262.

Vonesh, E. F. (1996). "A Note on Laplace's Approximation for Nonlinear Mixed-Effects Models." *Biometrika* 83:447–452.

Vonesh, E. F., and Chinchilli, V. M. (1997). *Linear and Nonlinear Models for the Analysis of Repeated Measurements*. New York: Marcel Dekker.

Vonesh, E. F., Chinchilli, V. M., and Pu, K. (1996). "Goodness-of-Fit in Generalized Nonlinear Mixed-Effects Models." *Biometrics* 52:572–587.

Wedderburn, R. W. M. (1974). "Quasi-likelihood Functions, Generalized Linear Models, and the Gauss-Newton Method." *Biometrika* 61:439–447.

Westfall, P. H. (1997). "Multiple Testing of General Contrasts Using Logical Constraints and Correlations." *Journal of the American Statistical Association* 92:299–306.

Westfall, P. H., and Tobias, R. D. (2007). "Multiple Testing of General Contrasts: Truncated Closure and the Extended Shaffer-Royen Method." *Journal of the American Statistical Association* 478:487–494.

Westfall, P. H., Tobias, R. D., Rom, D., Wolfinger, R. D., and Hochberg, Y. (1999). *Multiple Comparisons and Multiple Tests Using the SAS System*. Cary, NC: SAS Institute Inc.

Westfall, P. H., and Young, S. S. (1993). *Resampling-Based Multiple Testing: Examples and Methods for* p-*Value Adjustment*. New York: John Wiley & Sons.

White, H. (1980). "A Heteroskedasticity-Consistent Covariance Matrix Estimator and a Direct Test for Heteroskedasticity." *Econometrica* 48:817–838.

White, H. (1982). "Maximum Likelihood Estimation of Misspecified Models." *Econometrica* 50:1–25.

Whittle, P. (1954). "On Stationary Processes in the Plane." *Biometrika* 41:434–449.

Winer, B. J. (1971). *Statistical Principles in Experimental Design*. 2nd ed. New York: McGraw-Hill.

Wolfinger, R. D. (1993). "Laplace's Approximation for Nonlinear Mixed Models." *Biometrika* 80:791–795.

Wolfinger, R. D., and O'Connell, M. A. (1993). "Generalized Linear Mixed Models: A Pseudo-likelihood Approach." *Journal of Statistical Computation and Simulation* 48:233–243.

Wolfinger, R. D., Tobias, R. D., and Sall, J. (1994). "Computing Gaussian Likelihoods and Their Derivatives for General Linear Mixed Models." *SIAM Journal on Scientific Computing* 15:1294–1310.

Zeger, S. L., and Liang, K.-Y. (1986). "Longitudinal Data Analysis for Discrete and Continuous Outcomes." *Biometrics* 42:121–130.

Chapter 53
The GLM Procedure

Contents

Overview: GLM Procedure . . . **4145**
- PROC GLM Features . . . 4145
- PROC GLM Contrasted with Other SAS Procedures . . . 4146

Getting Started: GLM Procedure . . . **4147**
- PROC GLM for Unbalanced ANOVA . . . 4147
- PROC GLM for Quadratic Least Squares Regression . . . 4150

Syntax: GLM Procedure . . . **4156**
- PROC GLM Statement . . . 4158
- ABSORB Statement . . . 4165
- BY Statement . . . 4165
- CLASS Statement . . . 4166
- CODE Statement . . . 4167
- CONTRAST Statement . . . 4168
- ESTIMATE Statement . . . 4170
- FREQ Statement . . . 4171
- ID Statement . . . 4171
- LSMEANS Statement . . . 4172
- MANOVA Statement . . . 4181
- MEANS Statement . . . 4186
- MODEL Statement . . . 4192
- OUTPUT Statement . . . 4196
- RANDOM Statement . . . 4199
- REPEATED Statement . . . 4200
- STORE Statement . . . 4204
- TEST Statement . . . 4205
- WEIGHT Statement . . . 4206

Details: GLM Procedure . . . **4206**
- Statistical Assumptions for Using PROC GLM . . . 4206
- Specification of Effects . . . 4207
- Using PROC GLM Interactively . . . 4210
- Parameterization of PROC GLM Models . . . 4210
- Hypothesis Testing in PROC GLM . . . 4215
- Effect Size Measures for F Tests in GLM . . . 4220
- Absorption . . . 4224
- Specification of ESTIMATE Expressions . . . 4227
- Comparing Groups . . . 4228

Means versus LS-Means 4229
Multiple Comparisons 4231
Simple Effects 4241
Homogeneity of Variance in One-Way Models 4243
Weighted Means 4244
Construction of Least Squares Means 4244
Multivariate Analysis of Variance 4247
Repeated Measures Analysis of Variance 4249
Random-Effects Analysis 4257
Missing Values 4260
Computational Resources 4261
Computational Method 4263
Output Data Sets 4264
Displayed Output 4266
ODS Table Names 4267
ODS Graphics 4270
Examples: GLM Procedure **4272**
Example 53.1: Randomized Complete Blocks with Means Comparisons and Contrasts 4273
Example 53.2: Regression with Mileage Data 4278
Example 53.3: Unbalanced ANOVA for Two-Way Design with Interaction 4281
Example 53.4: Analysis of Covariance 4286
Example 53.5: Three-Way Analysis of Variance with Contrasts 4292
Example 53.6: Multivariate Analysis of Variance 4296
Example 53.7: Repeated Measures Analysis of Variance 4303
Example 53.8: Mixed Model Analysis of Variance with the RANDOM Statement . . . 4307
Example 53.9: Analyzing a Doubly Multivariate Repeated Measures Design 4310
Example 53.10: Testing for Equal Group Variances 4315
Example 53.11: Analysis of a Screening Design 4320
References **4324**

Overview: GLM Procedure

The GLM procedure uses the method of least squares to fit general linear models. Among the statistical methods available in PROC GLM are regression, analysis of variance, analysis of covariance, multivariate analysis of variance, and partial correlation.

PROC GLM analyzes data within the framework of general linear models. PROC GLM handles models relating one or several continuous dependent variables to one or several independent variables. The independent variables can be either *classification* variables, which divide the observations into discrete groups, or *continuous* variables. Thus, the GLM procedure can be used for many different analyses, including the following:

- simple regression
- multiple regression
- analysis of variance (ANOVA), especially for unbalanced data
- analysis of covariance
- response surface models
- weighted regression
- polynomial regression
- partial correlation
- multivariate analysis of variance (MANOVA)
- repeated measures analysis of variance

PROC GLM Features

The following list summarizes the features in PROC GLM:

- PROC GLM enables you to specify any degree of interaction (crossed effects) and nested effects. It also provides for polynomial, continuous-by-class, and continuous-nesting-class effects.
- Through the concept of estimability, the GLM procedure can provide tests of hypotheses for the effects of a linear model regardless of the number of missing cells or the extent of confounding. PROC GLM displays the sum of squares (SS) associated with each hypothesis tested and, upon request, the form of the estimable functions employed in the test. PROC GLM can produce the general form of all estimable functions.
- The REPEATED statement enables you to specify effects in the model that represent repeated measurements on the same experimental unit for the same response, providing both univariate and multivariate tests of hypotheses.

- The RANDOM statement enables you to specify random effects in the model; expected mean squares are produced for each Type I, Type II, Type III, Type IV, and contrast mean square used in the analysis. Upon request, F tests that use appropriate mean squares or linear combinations of mean squares as error terms are performed.
- The ESTIMATE statement enables you to specify an $\mathbf{L}$ vector for estimating a linear function of the parameters $\mathbf{L}\boldsymbol{\beta}$.
- The CONTRAST statement enables you to specify a contrast vector or matrix for testing the hypothesis that $\mathbf{L}\boldsymbol{\beta} = 0$. When specified, the contrasts are also incorporated into analyses that use the MANOVA and REPEATED statements.
- The MANOVA statement enables you to specify both the hypothesis effects and the error effect to use for a multivariate analysis of variance.
- PROC GLM can create an output data set containing the input data set in addition to predicted values, residuals, and other diagnostic measures.
- PROC GLM can be used interactively. After you specify and fit a model, you can execute a variety of statements without recomputing the model parameters or sums of squares.
- For analysis involving multiple dependent variables but not the MANOVA or REPEATED statements, a missing value in one dependent variable does not eliminate the observation from the analysis for other dependent variables. PROC GLM automatically groups together those variables that have the same pattern of missing values within the data set or within a BY group. This ensures that the analysis for each dependent variable brings into use all possible observations.
- The GLM procedure automatically produces graphs as part of its ODS output. For general information about ODS Graphics, see the section "ODS Graphics" on page 4270 and Chapter 24, "Statistical Graphics Using ODS."

PROC GLM Contrasted with Other SAS Procedures

As described previously, PROC GLM can be used for many different analyses and has many special features not available in other SAS procedures. However, for some types of analyses, other procedures are available. As discussed in the sections "PROC GLM for Unbalanced ANOVA" on page 4147 and "PROC GLM for Quadratic Least Squares Regression" on page 4150, sometimes these other procedures are more efficient than PROC GLM. The following procedures perform some of the same analyses as PROC GLM:

ANOVA — performs analysis of variance for balanced designs. The ANOVA procedure is generally more efficient than PROC GLM for these designs.

MIXED — fits mixed linear models by incorporating covariance structures in the model fitting process. Its RANDOM and REPEATED statements are similar to those in PROC GLM but offer different functionalities.

NESTED — performs analysis of variance and estimates variance components for nested random models. The NESTED procedure is generally more efficient than PROC GLM for these models.

NPAR1WAY performs nonparametric one-way analysis of rank scores. This can also be done using the RANK procedure and PROC GLM.

REG performs simple linear regression. The REG procedure allows several MODEL statements and gives additional regression diagnostics, especially for detection of collinearity.

RSREG performs quadratic response surface regression, and canonical and ridge analysis. The RSREG procedure is generally recommended for data from a response surface experiment.

TTEST compares the means of two groups of observations. Also, tests for equality of variances for the two groups are available. The TTEST procedure is usually more efficient than PROC GLM for this type of data.

VARCOMP estimates variance components for a general linear model.

Getting Started: GLM Procedure

PROC GLM for Unbalanced ANOVA

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glmunbal.sas.)

Analysis of variance, or ANOVA, typically refers to partitioning the variation in a variable's values into variation between and within several groups or classes of observations. The GLM procedure can perform simple or complicated ANOVA for balanced or unbalanced data.

This example discusses the analysis of variance for the unbalanced 2×2 data shown in Table 53.1. The experimental design is a full factorial, in which each level of one treatment factor occurs at each level of the other treatment factor. Note that there is only one value for the cell with A='A2' and B='B2'. Since one cell contains a different number of values from the other cells in the table, this is an unbalanced design.

Table 53.1 Unbalanced Two-Way Data

	A1	**A2**
B1	12, 14	20, 18
B2	11, 9	17

The following statements read the data into a SAS data set and then invoke PROC GLM to produce the analysis.

```
title 'Analysis of Unbalanced 2-by-2 Factorial';
data exp;
   input A $ B $ Y @@;
   datalines;
A1 B1 12 A1 B1 14     A1 B2 11 A1 B2 9
A2 B1 20 A2 B1 18     A2 B2 17
;
```

```
proc glm data=exp;
   class A B;
   model Y=A B A*B;
run;
```

Both treatments are listed in the CLASS statement because they are classification variables. A*B denotes the interaction of the A effect and the B effect. The results are shown in Figure 53.1 and Figure 53.2.

Figure 53.1 Class Level Information

Analysis of Unbalanced 2-by-2 Factorial

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
A	2	A1 A2
B	2	B1 B2

Number of Observations Read	7
Number of Observations Used	7

Figure 53.1 displays information about the classes as well as the number of observations in the data set. Figure 53.2 shows the ANOVA table, simple statistics, and tests of effects.

Figure 53.2 ANOVA Table and Tests of Effects

Analysis of Unbalanced 2-by-2 Factorial

The GLM Procedure

Dependent Variable: Y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	91.71428571	30.57142857	15.29	0.0253
Error	3	6.00000000	2.00000000		
Corrected Total	6	97.71428571			

R-Square	Coeff Var	Root MSE	Y Mean
0.938596	9.801480	1.414214	14.42857

Source	DF	Type I SS	Mean Square	F Value	Pr > F
A	1	80.04761905	80.04761905	40.02	0.0080
B	1	11.26666667	11.26666667	5.63	0.0982
A*B	1	0.40000000	0.40000000	0.20	0.6850

Source	DF	Type III SS	Mean Square	F Value	Pr > F
A	1	67.60000000	67.60000000	33.80	0.0101
B	1	10.00000000	10.00000000	5.00	0.1114
A*B	1	0.40000000	0.40000000	0.20	0.6850

The degrees of freedom can be used to check your data. The Model degrees of freedom for a 2×2 factorial design with interaction are $(ab - 1)$, where a is the number of levels of A and b is the number of levels of B; in this case, $(2 \times 2 - 1) = 3$. The Corrected Total degrees of freedom are always one less than the number of observations used in the analysis; in this case, 7 – 1 = 6.

The overall F test is significant ($F = 15.29$, $p = 0.0253$), indicating strong evidence that the means for the four different A×B cells are different. You can further analyze this difference by examining the individual tests for each effect.

Four types of estimable functions of parameters are available for testing hypotheses in PROC GLM. For data with no missing cells, the Type III and Type IV estimable functions are the same and test the same hypotheses that would be tested if the data were balanced. Type I and Type III sums of squares are typically not equal when the data are unbalanced; Type III sums of squares are preferred in testing effects in unbalanced cases because they test a function of the underlying parameters that is independent of the number of observations per treatment combination.

According to a significance level of 5% ($\alpha = 0.05$), the A*B interaction is not significant ($F = 0.20$, $p = 0.6850$). This indicates that the effect of A does not depend on the level of B and vice versa. Therefore, the tests for the individual effects are valid, showing a significant A effect ($F = 33.80$, $p = 0.0101$) but no significant B effect ($F = 5.00$, $p = 0.1114$).

If ODS Graphics is enabled, GLM also displays by default an interaction plot for this analysis. The following statements, which are the same as in the previous analysis but with ODS Graphics enabled, additionally produce Figure 53.3.

```
ods graphics on;
proc glm data=exp;
   class A B;
   model Y=A B A*B;
run;
ods graphics off;
```

Figure 53.3 Plot of Y by A and B

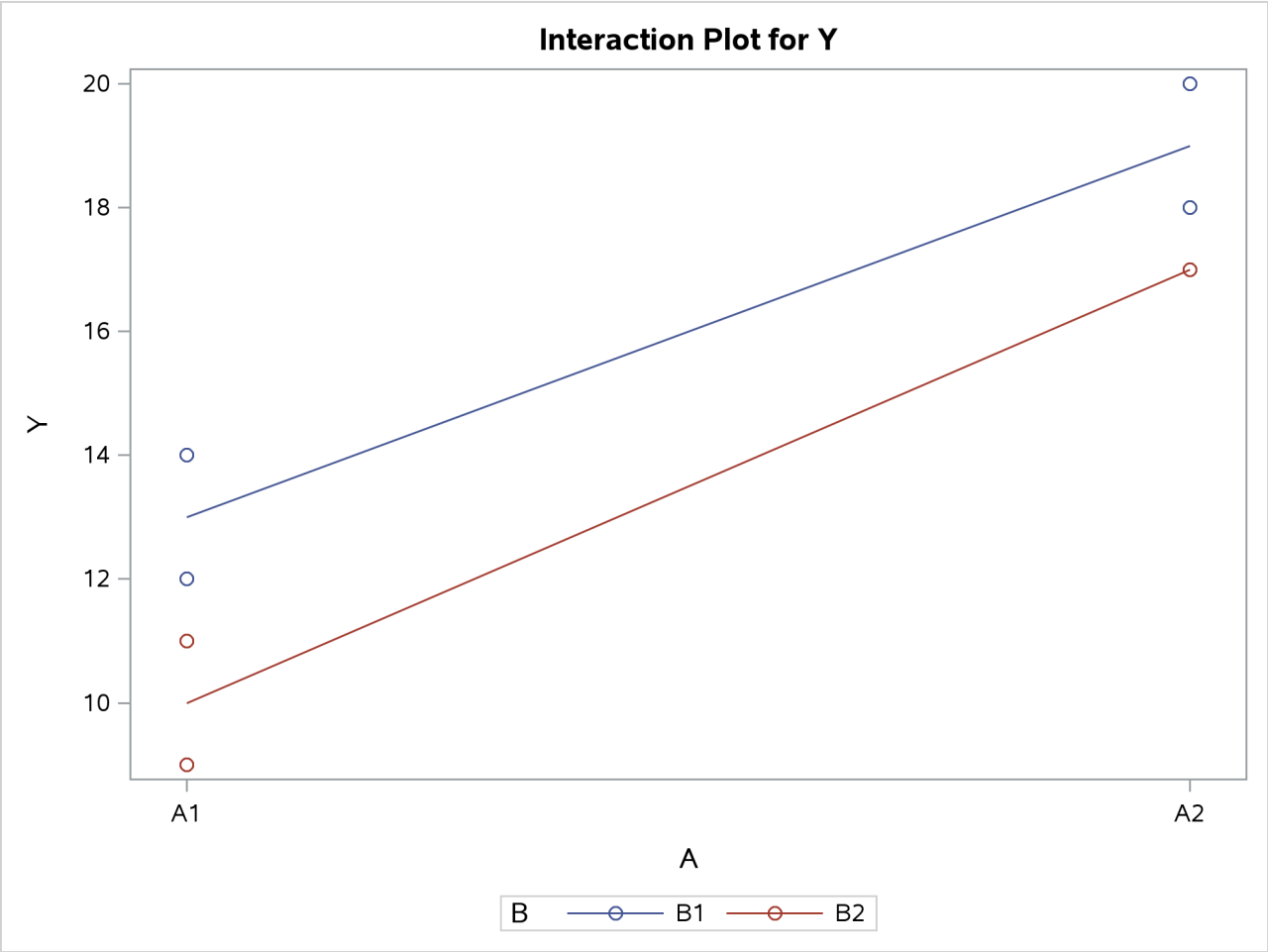

The insignificance of the A*B interaction is reflected in the fact that two lines in Figure 53.3 are nearly parallel. For more information about the graphics that GLM can produce, see the section "ODS Graphics" on page 4270.

PROC GLM for Quadratic Least Squares Regression

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glmin2.sas`.)

In polynomial regression, the values of a dependent variable (also called a response variable) are described or predicted in terms of polynomial terms involving one or more independent or explanatory variables. An example of quadratic regression in PROC GLM follows. These data are taken from Draper and Smith (1966, p. 57). Thirteen specimens of 90/10 Cu-Ni alloys are tested in a corrosion-wheel setup in order to examine corrosion. Each specimen has a certain iron content. The wheel is rotated in salt seawater at 30 ft/sec for 60 days. Weight loss is used to quantify the corrosion. The fe variable represents the iron content, and the loss variable denotes the weight loss in milligrams/square decimeter/day in the following DATA step.

```
title 'Regression in PROC GLM';
data iron;
   input fe loss @@;
   datalines;
0.01 127.6   0.48 124.0   0.71 110.8   0.95 103.9
1.19 101.5   0.01 130.1   0.48 122.0   1.44  92.3
0.71 113.1   1.96  83.7   0.01 128.0   1.44  91.4
1.96  86.2
;
```

The SGSCATTER procedure is used in the following statements to request a scatter plot of the response variable versus the independent variable.

```
proc sgscatter data=iron;
   plot loss*fe;
run;
ods graphics off;
```

The plot in Figure 53.4 displays a strong negative relationship between iron content and corrosion resistance, but it is not clear whether there is curvature in this relationship.

Figure 53.4 Plot of Observed Corrosion Resistance by Iron Content

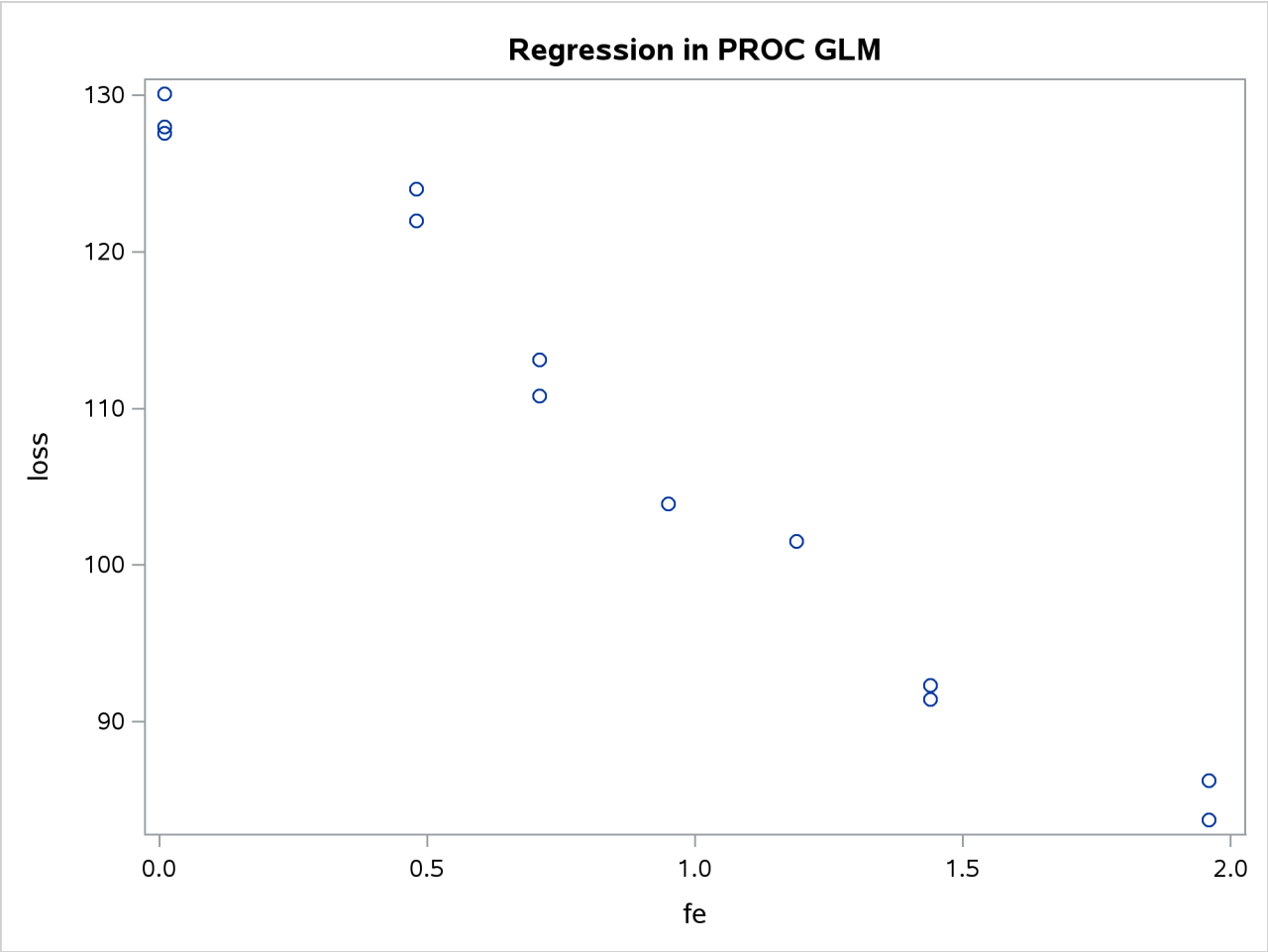

The following statements fit a quadratic regression model to the data. This enables you to estimate the linear relationship between iron content and corrosion resistance and to test for the presence of a quadratic component. The intercept is automatically fit unless the NOINT option is specified.

```
proc glm data=iron;
   model loss=fe fe*fe;
run;
```

The CLASS statement is omitted because a regression line is being fitted. Unlike PROC REG, PROC GLM allows polynomial terms in the MODEL statement.

PROC GLM first displays preliminary information, shown in Figure 53.5, telling you that the GLM procedure has been invoked and stating the number of observations in the data set. If the model involves classification variables, they are also listed here, along with their levels.

Figure 53.5 Data Information

Regression in PROC GLM

The GLM Procedure

Number of Observations Read	13
Number of Observations Used	13

Figure 53.6 shows the overall ANOVA table and some simple statistics. The degrees of freedom can be used to check that the model is correct and that the data have been read correctly. The Model degrees of freedom for a regression is the number of parameters in the model minus 1. You are fitting a model with three parameters in this case,

$$\text{loss} \quad = \quad \beta_0 + \beta_1 \times (\text{fe}) + \beta_2 \times (\text{fe})^2 + error$$

so the degrees of freedom are $3 - 1 = 2$. The Corrected Total degrees of freedom are always one less than the number of observations used in the analysis.

Figure 53.6 ANOVA Table

Regression in PROC GLM

The GLM Procedure

Dependent Variable: loss

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	3296.530589	1648.265295	164.68	<.0001
Error	10	100.086334	10.008633		
Corrected Total	12	3396.616923			

R-Square	Coeff Var	Root MSE	loss Mean
0.970534	2.907348	3.163642	108.8154

The R square indicates that the model accounts for 97% of the variation in LOSS. The coefficient of variation

(Coeff Var), Root MSE (Mean Square for Error), and mean of the dependent variable are also listed.

The overall F test is significant ($F = 164.68$, $p < 0.0001$), indicating that the model as a whole accounts for a significant amount of the variation in LOSS. Thus, it is appropriate to proceed to testing the effects.

Figure 53.7 contains tests of effects and parameter estimates. The latter are displayed by default when the model contains only continuous variables.

Figure 53.7 Tests of Effects and Parameter Estimates

Source	DF	Type I SS	Mean Square	F Value	Pr > F
fe	1	3293.766690	3293.766690	329.09	<.0001
fe*fe	1	2.763899	2.763899	0.28	0.6107

Source	DF	Type III SS	Mean Square	F Value	Pr > F
fe	1	356.7572421	356.7572421	35.64	0.0001
fe*fe	1	2.7638994	2.7638994	0.28	0.6107

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	130.3199337	1.77096213	73.59	<.0001
fe	-26.2203900	4.39177557	-5.97	0.0001
fe*fe	1.1552018	2.19828568	0.53	0.6107

The t tests provided are equivalent to the Type III F tests. The quadratic term is not significant ($p = 0.6107$) and thus can be removed from the model; the linear term is significant ($p < 0.0001$). This suggests that there is indeed a straight-line relationship between loss and fe.

Finally, if ODS Graphics is enabled, PROC GLM also displays by default a scatter plot of the original data, as in Figure 53.4, with the quadratic fit overlaid. The following statements, which are the same as the previous analysis but with ODS Graphics enabled, additionally produce Figure 53.8.

```
ods graphics on;
proc glm data=iron;
   model loss=fe fe*fe;
run;
ods graphics off;
```

Figure 53.8 Plot of Observed and Fit Corrosion Resistance by Iron Content, Quadratic Model

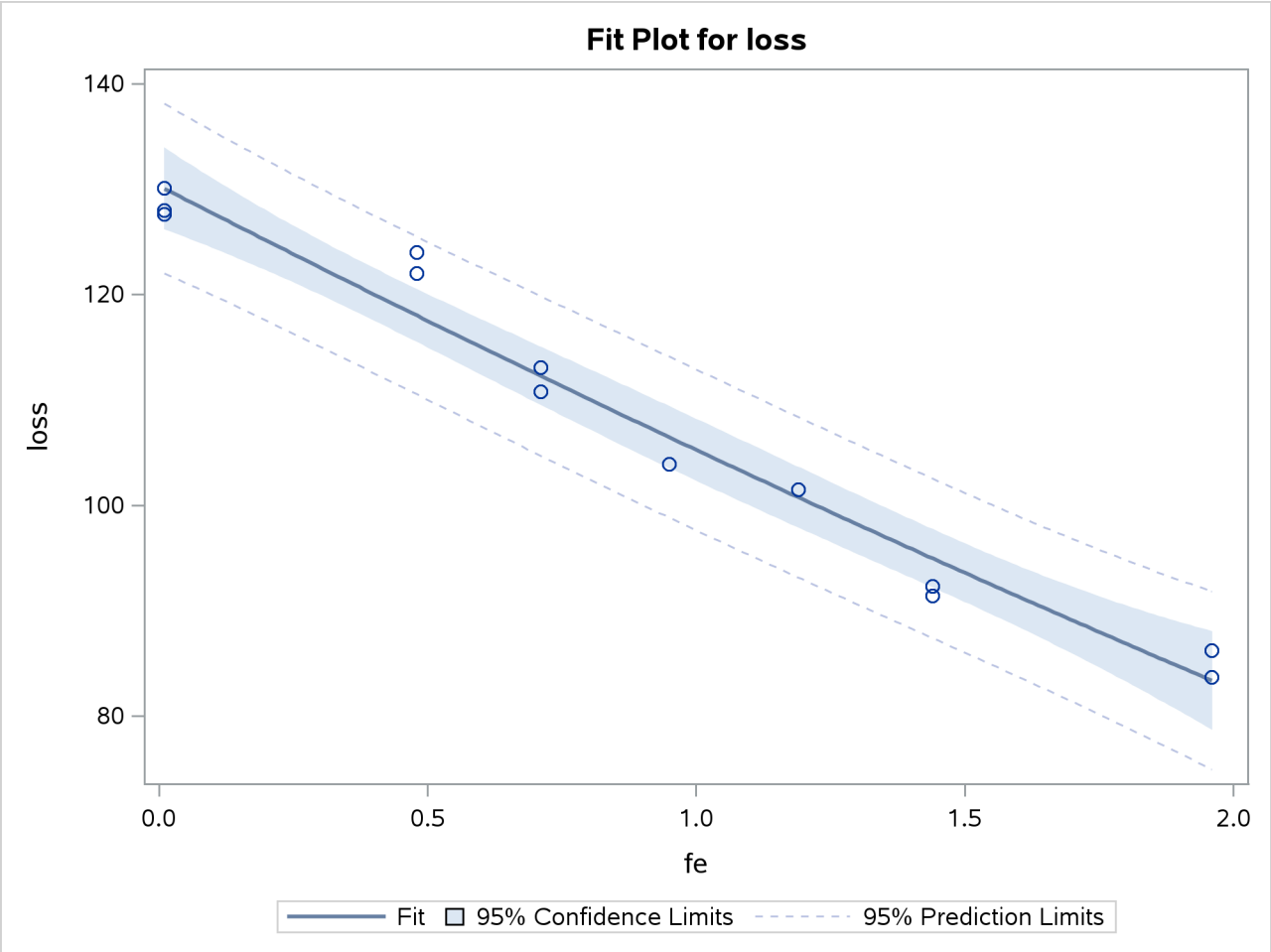

The insignificance of the quadratic term in the model is reflected in the fact that the fit is nearly linear.

Fitting the model without the quadratic term provides more accurate estimates for β_0 and β_1. PROC GLM allows only one MODEL statement per invocation of the procedure, so the PROC GLM statement must be issued again. The following statements are used to fit the linear model.

```
proc glm data=iron;
   model loss=fe;
run;
```

Figure 53.9 displays the output produced by these statements. The linear term is still significant ($F = 352.27$, $p < 0.0001$). The estimated model is now

$$\text{loss} = 129.79 - 24.02 \times \text{fe}$$

Figure 53.9 Linear Model Output

Regression in PROC GLM

The GLM Procedure

Dependent Variable: loss

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	3293.766690	3293.766690	352.27	<.0001
Error	11	102.850233	9.350021		
Corrected Total	12	3396.616923			

R-Square	Coeff Var	Root MSE	loss Mean
0.969720	2.810063	3.057780	108.8154

Source	DF	Type I SS	Mean Square	F Value	Pr > F
fe	1	3293.766690	3293.766690	352.27	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
fe	1	3293.766690	3293.766690	352.27	<.0001

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	129.7865993	1.40273671	92.52	<.0001
fe	-24.0198934	1.27976715	-18.77	<.0001

Syntax: GLM Procedure

The following statements are available in the GLM procedure:

```
PROC GLM <options> ;
    CLASS variable <(REF= option)> ... <variable <(REF= option)>> </ global-options> ;
    MODEL dependent-variables = independent-effects </ options> ;
    ABSORB variables ;
    BY variables ;
    CODE <options> ;
    FREQ variable ;
    ID variables ;
    WEIGHT variable ;
    CONTRAST 'label' effect values <... effect values> </ options> ;
    ESTIMATE 'label' effect values <... effect values> </ options> ;
    LSMEANS effects </ options> ;
    MANOVA <test-options> </ detail-options> ;
    MEANS effects </ options> ;
    OUTPUT <OUT=SAS-data-set> keyword=names <... keyword=names> </ option> ;
    RANDOM effects </ options> ;
    REPEATED factor-specification </ options> ;
    STORE <OUT= >item-store-name </ LABEL='label'> ;
    TEST <H=effects> E=effect </ options> ;
```

Although there are numerous statements and options available in PROC GLM, many applications use only a few of them. Often you can find the features you need by looking at an example or by quickly scanning through this section.

To use PROC GLM, the PROC GLM and MODEL statements are required. You can specify only one MODEL statement (in contrast to the REG procedure, for example, which allows several MODEL statements in the same PROC REG run). If your model contains classification effects, the classification variables must be listed in a CLASS statement, and the CLASS statement must appear before the MODEL statement. In addition, if you use a CONTRAST statement in combination with a MANOVA, RANDOM, REPEATED, or TEST statement, the CONTRAST statement must be entered first in order for the contrast to be included in the MANOVA, RANDOM, REPEATED, or TEST analysis.

Table 53.2 summarizes the positional requirements for the statements in the GLM procedure.

Table 53.2 Positional Requirements for PROC GLM Statements

Statement	Must Precede...	Must Follow...
ABSORB	First RUN statement	
BY	First RUN statement	
CLASS	MODEL statement	
CONTRAST	MANOVA, REPEATED, or RANDOM statement	MODEL statement
ESTIMATE		MODEL statement

Table 53.2 *continued*

Statement	Must Precede...	Must Follow...
FREQ	First RUN statement	
ID	First RUN statement	
LSMEANS		MODEL statement
MANOVA		CONTRAST or MODEL statement
MEANS		MODEL statement
MODEL	CONTRAST, ESTIMATE, LSMEANS, or MEANS statement	CLASS statement
OUTPUT		MODEL statement
RANDOM		CONTRAST or MODEL statement
REPEATED		CONTRAST, MODEL, or TEST statement
TEST	MANOVA or REPEATED statement	MODEL statement
WEIGHT	First RUN statement	

Table 53.3 summarizes the function of each statement (other than the PROC statement) in the GLM procedure.

Table 53.3 Statements in the GLM Procedure

Statement	Description
ABSORB	Absorbs classification effects in a model
BY	Specifies variables to define subgroups for the analysis
CLASS	Declares classification variables
CODE	Requests that the procedure write SAS DATA step code to a file or catalog entry for computing predicted values according to the fitted model
CONTRAST	Constructs and tests linear functions of the parameters
ESTIMATE	Estimates linear functions of the parameters
FREQ	Specifies a frequency variable
ID	Identifies observations on output
LSMEANS	Computes least squares (marginal) means
MANOVA	Performs a multivariate analysis of variance
MEANS	Computes and optionally compares arithmetic means
MODEL	Defines the model to be fit
OUTPUT	Requests an output data set containing diagnostics for each observation
RANDOM	Declares certain effects to be random and computes expected mean squares

Table 53.3 *continued*

Statement	Description
REPEATED	Performs multivariate and univariate repeated measures analysis of variance
STORE	Requests that the procedure save the context and results of the statistical analysis into an item store
TEST	Constructs tests that use the sums of squares for effects and the error term you specify
WEIGHT	Specifies a variable for weighting observations

The rest of this section provides detailed syntax information for each of these statements, beginning with the PROC GLM statement. The remaining statements are covered in alphabetical order.

The STORE and CODE statements are also used by many other procedures. A summary description of functionality and syntax for these statements is also shown after the PROC GLM statement in alphabetical order, but you can find full documentation about them in the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

PROC GLM Statement

PROC GLM < *options* > **;**

The PROC GLM statement invokes the GLM procedure. Table 53.4 summarizes the *options* available in the PROC GLM statement.

Table 53.4 PROC GLM Statement Options

Option	Description
ALPHA=	Specifies the level of significance for confidence intervals
DATA=	Names the SAS data set used by the GLM procedure
MANOVA	Requests the multivariate mode of eliminating observations with missing values
MULTIPASS	Requests that the input data set be reread when necessary, instead of using a utility file
NAMELEN=	Specifies the length of effect names
NOPRINT	Suppresses the normal display of results
ORDER=	Specifies the order in which to sort classification variables
OUTSTAT=	Names an output data set for information and statistics on each model effect
PLOTS	Controls the plots produced through ODS Graphics

You can specify the following *options* in the PROC GLM statement.

ALPHA=*p*

specifies the level of significance p for $100(1-p)\%$ confidence intervals. The value must be between 0 and 1; the default value of $p = 0.05$ results in 95% intervals. This value is used as the default confidence level for limits computed by the following *options*.

Statement	Options
LSMEANS	CL
MEANS	CLM CLDIFF
MODEL	CLI CLM CLPARM
OUTPUT	UCL= LCL= UCLM= LCLM=

You can override the default in each of these cases by specifying the ALPHA= option for each statement individually.

DATA=*SAS-data-set*

names the SAS data set used by the GLM procedure. By default, PROC GLM uses the most recently created SAS data set.

MANOVA

requests the multivariate mode of eliminating observations with missing values. If any of the dependent variables have missing values, the procedure eliminates that observation from the analysis. The MANOVA option is useful if you use PROC GLM in interactive mode and plan to perform a multivariate analysis.

MULTIPASS

requests that PROC GLM reread the input data set when necessary, instead of writing the necessary values of dependent variables to a utility file. This option decreases disk space usage at the expense of increased execution times, and is useful only in rare situations where disk space is at an absolute premium.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters long, where *n* is a value between 20 and 200 characters. The default length is 20 characters.

NOPRINT

suppresses the normal display of results. The NOPRINT option is useful when you want only to create one or more output data sets with the procedure. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This ordering determines which parameters in the model correspond to each level in the data, so the ORDER= option can be useful when you specify the CONTRAST or ESTIMATE statement.

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUTSTAT=*SAS-data-set*

names an output data set that contains sums of squares, degrees of freedom, *F* statistics, and probability levels for each effect in the model, as well as for each CONTRAST that uses the overall residual or error mean square (MSE) as the denominator in constructing the *F* statistic. If you use the CANONICAL option in the MANOVA statement and do not use an M= specification in the MANOVA statement, the data set also contains results of the canonical analysis.

See the section "Output Data Sets" on page 4264 for more information.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the plots produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses from around the *plot-request*. For example:

```
PLOTS=NONE
PLOTS=(DIAGNOSTICS RESIDUALS)
PLOTS(UNPACK)=RESIDUALS
PLOT=MEANPLOT(CLBAND)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc glm data=iron;
   model loss=fe fe*fe;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, then PROC GLM produces a default set of plots, which might be different for different models, as discussed in the following.

- If you specify a one-way analysis of variance model that has just one CLASS variable, the GLM procedure produces a grouped box plot of the response values versus the CLASS levels. For an example of the box plot, see the section "One-Way Layout with Means Comparisons" on page 1119 in Chapter 29, "The ANOVA Procedure."
- If you specify a two-way analysis of variance model that has just two CLASS variables, the GLM procedure produces an interaction plot of the response values, where horizontal position represents one CLASS variable and marker style represents the other, and where predicted response values connected by lines represent the two-way analysis. For an example of the interaction plot, see the section "PROC GLM for Unbalanced ANOVA" on page 4147.
- If you specify a model that has two CLASS variables, and one variable is nested within the other, then the GLM procedure produces a nested box plot of the response values, where horizontal position represents one CLASS variable nested within the other CLASS variable.
- If you specify a model that has a single continuous predictor, the GLM procedure produces a fit plot of the response values versus the covariate values, where a curve represents the fitted relationship and a band represents the confidence limits for individual mean values. For an example of the fit plot, see the section "PROC GLM for Quadratic Least Squares Regression" on page 4150.
- If you specify a model that has two continuous predictors and no CLASS variables, the GLM procedure produces a contour fit plot, overlaying a scatter plot of the data and a contour plot of the predicted surface.
- If you specify an analysis of covariance model that has one or two CLASS variables and one continuous variable, the GLM procedure produces an analysis of covariance plot of the response values versus the covariate values, where lines represent the fitted relationship within each classification level. For an example of the analysis of covariance plot, see Example 53.4.
- If you specify an LSMEANS statement with the PDIFF option, the GLM procedure produces a plot appropriate for the type of LS-means comparison. For PDIFF=ALL (which is the default if you specify only PDIFF), the procedure produces a diffogram, which displays all pairwise LS-means differences and their significance. The display is also known as a "mean-mean scatter plot" (Hsu 1996). For PDIFF=CONTROL, the procedure produces a display of each noncontrol LS-mean compared to the control LS-mean, with two-sided confidence intervals for the comparison. For PDIFF=CONTROLL and PDIFF=CONTROLU a similar display is produced, but with one-sided confidence intervals. Finally, for the PDIFF=ANOM option, the procedure produces an analysis-of-means plot, which compares each LS-mean to the average LS-mean.
- If you specify a MEANS statement, the GLM procedure produces a grouped box plot of the response values versus the effect for which means are being calculated.

The *global-plot-options* include the following:

MAXPOINTS=NONE | *number*

suppresses plots that contain elements that require processing of more than *number* points. The default is MAXPOINTS=5000. This limit is ignored if you specify MAXPOINTS=NONE.

ONLY
suppresses the default plots. Only plots that you specifically request are displayed.

UNPACKPANEL
UNPACK
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACKPANEL to get each plot in a separate panel. You can specify PLOTS(UNPACKPANEL) to just unpack the default plots. You can also specify UNPACKPANEL as a suboption with DIAGNOSTICS and RESIDUALS.

The following individual plots and plot options are available. If you specify only one *plot*, then you can omit the parentheses.

ALL
produces all appropriate plots. You can specify other options with ALL; for example, to request all plots and unpack just the residuals, specify: PLOTS=(ALL RESIDUALS(UNPACK)).

ANCOVAPLOT< (CLM CLI LIMITS) >
modifies the analysis of covariance plot produced by default when you have an analysis of covariance model, with one or two CLASS variables and one continuous variable. By default the plot does not show confidence limits around the predicted values. The PLOTS=ANCOVAPLOT(CLM) option adds limits for the expected predicted values, and PLOTS=ANCOVAPLOT(CLI) adds limits for new predictions. Use PLOTS=ANCOVAPLOT(LIMITS) to add both kinds of limits.

ANOMPLOT
requests an analysis-of-means display, in which least squares means are compared against an average least squares mean (Ott 1967; Nelson 1982, 1991, 1993). LS-mean ANOM plots are produced only if you also specify PDIFF=ANOM or ADJUST=NELSON in the LSMEANS statement, and in this case they are produced by default.

BOXPLOT< (NPANELPOS=*n*) >
modifies the plot produced by default for the model effect in a one-way analysis of variance model, or for an effect specified in the MEANS statement. Suppose the effect has m levels. By default, or if you specify PLOTS=BOXPLOT(NPANELPOS=0), all m levels of the effect are displayed in a single plot. Specifying a nonzero value of n will result in P panels, where P is the integer part of $m/n + 1$. If $n > 0$, then the levels will be approximately balanced across the P panels; whereas if $n < 0$, precisely $|n|$ levels will be displayed on each panel except possibly the last.

CONTOURFIT< (OBS=*obs-options*) >
modifies the contour fit plot produced by default when you have a model involving only two continuous predictors. The plot displays a contour plot of the predicted surface overlaid with a scatter plot of the observed data. You can use the following *obs-options* to control how the observations are displayed:

GRADIENT
specifies that observations are displayed as circles colored by the observed response. The same color gradient is used to display the fitted surface and the observations. Observations where the predicted response is close to the observed response have similar colors: the greater the contrast between the color of an observation and the surface, the larger the residual is at that point.

NONE
suppresses the observations.

OUTLINE
specifies that observations are displayed as circles with a border but with a completely transparent fill.

OUTLINEGRADIENT
is the same as OBS=GRADIENT except that a border is shown around each observation. This option is useful to identify the location of observations where the residuals are small, since at these points the color of the observations and the color of the surface are indistinguishable. OBS=OUTLINEGRADIENT is the default if you do not specify any *obs-options*.

CONTROLPLOT
requests a display in which least squares means are compared against a reference level. LS-mean control plots are produced only when you specify PDIFF=CONTROL or ADJUST=DUNNETT in the LSMEANS statement, and in this case they are produced by default.

DIAGNOSTICS< (LABEL UNPACK) >
requests that a panel of summary diagnostics for the fit be displayed. The panel displays scatter plots of residuals, studentized residuals, and observed responses by predicted values; studentized residuals by leverage; Cook's D by observation; a Q-Q plot of residuals; a residual histogram; and a residual-fit spread plot. The LABEL option displays labels on observations satisfying RSTUDENT > 2, LEVERAGE $> 2p/n$, and on the Cook's D plot, COOKSD $> 4/n$, where n is the number of observations used in fitting the model, and p is the number of parameters in the model. The label is the first ID variable if the ID statement is specified; otherwise, it is the observation number. The UNPACK option unpanels the diagnostic display and produces the series of individual plots that form the paneled display.

DIFFPLOT< (ABS NOABS CENTER NOLINES) >
modifies the plot produced by an LSMEANS statement with the PDIFF=ALL option (or just PDIFF, since ALL is the default argument). The ABS and NOABS options determine the positioning of the line segments in the plot. When the ABS option is in effect, and this is the default, all line segments are shown on the same side of the reference line. The NOABS option separates comparisons according to the sign of the difference. The CENTER option marks the center point for each comparison. This point corresponds to the intersection of two least squares means. The NOLINES option suppresses the display of the line segments that represent the confidence bounds for the differences of the least squares means. The NOLINES option implies the CENTER option. The default is to draw line segments in the upper portion of the plot area without marking the center point.

FITPLOT< (NOCLM NOCLI NOLIMITS) >

modifies the fit plot produced by default when you have a model with a single continuous predictor. By default the plot includes confidence limits for both the expected predicted values and individual new predictions. The PLOTS=FITPLOT(NOCLM) option removes the limits on the expected values and the PLOTS=FITPLOT(NOCLI) option removes the limits on new predictions. The PLOTS=FITPLOT(NOLIMITS) option removes both kinds of confidence limits.

INTPLOT< (CLM CLI LIMITS) >

modifies the interaction plot that is produced by default when you have a two-way analysis of variance model that has just two CLASS variables. By default, the plot does not show confidence limits around the predicted values. The PLOTS=INTPLOT(CLM) option adds limits for the expected predicted values and PLOTS=INTPLOT(CLI) adds limits for new predictions. Use PLOTS=INTPLOT(LIMITS) to add both kinds of limits.

LINESPLOT< (WSCALE=*wfactor* HSCALE=*hfactor*) >

modifies the dimensions of the means comparison plot that is produced by the LINES option in the LSMEANS and MEANS statements. The default dimensions of the plot vary according to aspects such as the number of groups, the number of CLASS variables in the effect, and the number of parallel lines needed to represent comparisons. You can change the defaults by specifying the following options:

HSCALE=*hfactor*

scales the default height of the plot by *hfactor*, which must be a positive number. For example, specifying HSCALE=2 makes the plot twice as high as it would be by default.

WSCALE=*wfactor*

scales the default width of the plot by *wfactor*, which must be a positive number. For example, specifying WSCALE=2 makes the plot twice as wide as it would be by default.

MEANPLOT< (CL CLBAND CONNECT ASCENDING DESCENDING) >

modifies the grouped box plot that is produced by an LSMEANS statement. Upper and lower confidence limits are plotted when the CL option is specified. When the CLBAND option is in effect, confidence limits are shown as bands and the means are connected. By default, means are not joined by lines. You can achieve that effect by specifying the CONNECT option. Means are displayed in the same order in which they appear in the “Means” table. You can change that order for plotting by specifying the ASCENDING and DESCENDING options.

NESTPLOT< (BOXWIDTH=*value* CLUSTERWIDTH=*value* LABELOUTLIER) >

modifies the box plot that is produced by default when you have two CLASS variables and one is nested within the other. By default, the plot does not label the outliers. The PLOTS=NESTPLOT(BOXWIDTH=) option controls the width of the boxes, and the PLOTS=NESTPLOT(CLUSTERWIDTH=) option controls how tightly the boxes are clustered together. You can specify the BOXWIDTH= and CLUSTERWIDTH= *values* as numbers between 0 and 1; the defaults are 0.8.

NONE
suppresses all graphics.

RESIDUALS< (SMOOTH UNPACK) >
displays scatter plots of the residuals against each continuous covariate. The SMOOTH option overlays a Loess smooth on each residual plot. Note that if a WEIGHT variable is specified, then it is not used to weight the smoother. For more information, see Chapter 78, "The LOESS Procedure." The UNPACK option unpanels the residual display and produces a series of individual plots that form the paneled display.

ABSORB Statement

ABSORB *variables* **;**

Absorption is a computational technique that provides a large reduction in time and memory requirements for certain types of models. The *variables* are one or more variables in the input data set.

For a main-effect variable that does not participate in interactions, you can absorb the effect by naming it in an ABSORB statement. This means that the effect can be adjusted out before the construction and solution of the rest of the model. This is particularly useful when the effect has a large number of levels.

Several variables can be specified, in which case each one is assumed to be nested in the preceding variable in the ABSORB statement.

NOTE: When you use the ABSORB statement, the data set (or each BY group, if a BY statement appears) must be sorted by the variables in the ABSORB statement. The GLM procedure cannot produce predicted values or least squares means (LS-means) or create an output data set of diagnostic values if an ABSORB statement is used. If the ABSORB statement is used, it must appear before the first RUN statement; otherwise, it is ignored.

When you use an ABSORB statement and also use the INT option in the MODEL statement, the procedure ignores the option but computes the uncorrected total sum of squares (SS) instead of the corrected total sums of squares.

See the section "Absorption" on page 4224 for more information.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC GLM to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the GLM procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Since sorting the data changes the order in which PROC GLM reads observations, the sort order for the levels of the classification variables might be affected if you also specify ORDER=DATA in the PROC GLM statement. This, in turn, affects specifications in the CONTRAST and ESTIMATE statements.

If you specify the BY statement, it must appear before the first RUN statement; otherwise, it is ignored. When you use a BY statement, the interactive features of PROC GLM are disabled.

When both the BY and ABSORB statements are used, observations must be sorted first by the variables in the BY statement, and then by the variables in the ABSORB statement.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(REF=** *option*) > . . . < *variable* < **(REF=** *option*) > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC GLM statement.

The GLM procedure displays a table that summarizes the CLASS variables and their levels. You can use this table to check the ordering of levels and, hence, of the corresponding parameters for main effects. If you need to check the ordering of parameters for interaction effects, use the E option in the MODEL, CONTRAST, ESTIMATE, or LSMEANS statement. See the section "Parameterization of PROC GLM Models" on page 4210 for more information.

You can specify the following REF= option to indicate how the levels of an individual classification variable are to be ordered by enclosing it in parentheses after the variable name:

REF=*'level'* | **FIRST** | **LAST**
specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC GLM's singular parameterization. You can specify the *level* of the variable to use as the reference level; specify a value that corresponds to the formatted value of the variable if a format is assigned. Alternatively, you can specify REF=FIRST to designate that the first ordered level serve as the reference, or REF=LAST to designate that the last ordered level serve as the reference. To specify that REF=FIRST or REF=LAST be used for all classification variables, use the REF= *global-option* after the slash (/) in the CLASS statement.

You can specify the following *global-options* in the CLASS statement after a slash (/):

REF=FIRST | **LAST**
specifies a level of all classification variables to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC GLM's singular parameterization. Specify REF=FIRST to designate that the first ordered level for each classification variable serve as the reference. Specify REF=LAST to designate that the last ordered level serve as the reference. This option applies to all the variables specified in the CLASS statement. To specify different reference levels for different classification variables, use REF= options for individual variables.

TRUNCATE
specifies that class levels be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 53.5 summarizes the *options* available in the CODE statement.

Table 53.5 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

CONTRAST Statement

CONTRAST *'label' effect values < . . . effect values > < / options >* **;**

The CONTRAST statement enables you to perform custom hypothesis tests by specifying an $\mathbf{L}$ vector or matrix for testing the univariate hypothesis $\mathbf{L}\boldsymbol{\beta} = 0$ or the multivariate hypothesis $\mathbf{LBM} = 0$. Thus, to use this feature you must be familiar with the details of the model parameterization that PROC GLM uses. For more information, see the section "Parameterization of PROC GLM Models" on page 4210. All of the elements of the $\mathbf{L}$ vector might be given, or if only certain portions of the $\mathbf{L}$ vector are given, the remaining elements are constructed by PROC GLM from the context (in a manner similar to rule 4 discussed in the section "Construction of Least Squares Means" on page 4244).

There is no limit to the number of CONTRAST statements you can specify, but they must appear after the MODEL statement. In addition, if you use a CONTRAST statement and a MANOVA, REPEATED, or TEST statement, appropriate tests for contrasts are carried out as part of the MANOVA, REPEATED, or TEST analysis. If you use a CONTRAST statement and a RANDOM statement, the expected mean square of the contrast is displayed. As a result of these additional analyses, the CONTRAST statement must appear before the MANOVA, REPEATED, RANDOM, or TEST statement.

In the CONTRAST statement,

label identifies the contrast on the output. A label is required for every contrast specified. Labels must be enclosed in quotes.

effect identifies an effect that appears in the MODEL statement, or the INTERCEPT effect. The INTERCEPT effect can be used when an intercept is fitted in the model. You do not need to include all effects that are in the MODEL statement.

values are constants that are elements of the $\mathbf{L}$ vector associated with the effect.

You can specify the following *options* in the CONTRAST statement after a slash (/).

E

displays the entire $\mathbf{L}$ vector. This option is useful in confirming the ordering of parameters for specifying $\mathbf{L}$.

E=*effect*

specifies an error term, which must be one of the effects in the model. The procedure uses this effect as the denominator in F tests in univariate analysis. In addition, if you use a MANOVA or REPEATED statement, the procedure uses the effect specified by the E= option as the basis of the $\mathbf{E}$ matrix. By default, the procedure uses the overall residual or error mean square (MSE) as an error term.

ETYPE=*n*

specifies the type (1, 2, 3, or 4, corresponding to a Type I, II, III, or IV test, respectively) of the E= effect. If the E= option is specified and the ETYPE= option is not, the procedure uses the highest type computed in the analysis.

SINGULAR=*number*

tunes the estimability checking. If $\mathrm{ABS}(\mathbf{L} - \mathbf{LH}) > C \times \mathit{number}$ for any row in the contrast, then $\mathbf{L}$ is declared nonestimable. $\mathbf{H}$ is the $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$ matrix, and C is $\mathrm{ABS}(\mathbf{L})$ except for rows where $\mathbf{L}$ is zero, and then it is 1. The default value for the SINGULAR= option is 10^{-4}. Values for the SINGULAR= option must be between 0 and 1.

As stated previously, the CONTRAST statement enables you to perform custom hypothesis tests. If the hypothesis is testable in the univariate case, $\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = 0)$ is computed as

$$(\mathbf{Lb})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{Lb})$$

where $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y}$. This is the sum of squares displayed on the analysis-of-variance table.

For multivariate testable hypotheses, the usual multivariate tests are performed using

$$\mathbf{H} = \mathbf{M}'(\mathbf{LB})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{LB})\mathbf{M}$$

where $\mathbf{B} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y}$ and $\mathbf{Y}$ is the matrix of multivariate responses or dependent variables. The degrees of freedom associated with the hypothesis are equal to the row rank of $\mathbf{L}$. The sum of squares computed in this situation is equivalent to the sum of squares computed using an $\mathbf{L}$ matrix with any row deleted that is a linear combination of previous rows.

Multiple-degrees-of-freedom hypotheses can be specified by separating the rows of the $\mathbf{L}$ matrix with commas.

For example, for the model

```
proc glm;
   class A B;
   model Y=A B;
run;
```

with A at 5 levels and B at 2 levels, the parameter vector is

$$(\mu \;\; \alpha_1 \;\; \alpha_2 \;\; \alpha_3 \;\; \alpha_4 \;\; \alpha_5 \;\; \beta_1 \;\; \beta_2)$$

To test the hypothesis that the pooled A linear and A quadratic effect is zero, you can use the following $\mathbf{L}$ matrix:

$$\mathbf{L} = \begin{bmatrix} 0 & -2 & -1 & 0 & 1 & 2 & 0 & 0 \\ 0 & 2 & -1 & -2 & -1 & 2 & 0 & 0 \end{bmatrix}$$

The corresponding CONTRAST statement is

```
contrast 'A LINEAR & QUADRATIC'
         a -2 -1  0  1  2,
         a  2 -1 -2 -1  2;
```

If the first level of A is a control level and you want a test of control versus others, you can use this statement:

```
contrast 'CONTROL VS OTHERS'  a -1 0.25 0.25 0.25 0.25;
```

See the following discussion of the ESTIMATE statement and the section "Specification of ESTIMATE Expressions" on page 4227 for rules on specification, construction, distribution, and estimability in the CONTRAST statement.

ESTIMATE Statement

ESTIMATE *'label' effect values* < . . . *effect values* > < / *options* > **;**

The ESTIMATE statement enables you to estimate linear functions of the parameters by multiplying the vector $\mathbf{L}$ by the parameter estimate vector $\mathbf{b}$, resulting in $\mathbf{Lb}$. All of the elements of the $\mathbf{L}$ vector might be given, or, if only certain portions of the $\mathbf{L}$ vector are given, the remaining elements are constructed by PROC GLM from the context (in a manner similar to rule 4 discussed in the section "Construction of Least Squares Means" on page 4244).

The linear function is checked for estimability. The estimate $\mathbf{Lb}$, where $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y}$, is displayed along with its associated standard error, $\sqrt{\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'s^2}$, and t test. If you specify the CLPARM option in the MODEL statement (see page 4193), confidence limits for the true value are also displayed.

There is no limit to the number of ESTIMATE statements that you can specify, but they must appear after the MODEL statement. In the ESTIMATE statement,

label identifies the estimate on the output. A label is required for every contrast specified. Labels must be enclosed in quotes.

effect identifies an effect that appears in the MODEL statement, or the INTERCEPT effect. The INTERCEPT effect can be used as an effect when an intercept is fitted in the model. You do not need to include all effects that are in the MODEL statement.

values are constants that are the elements of the $\mathbf{L}$ vector associated with the preceding effect. For example,

```
estimate 'A1 VS A2' A  1 -1;
```

forms an estimate that is the difference between the parameters estimated for the first and second levels of the CLASS variable A.

You can specify the following *options* in the ESTIMATE statement after a slash (/):

DIVISOR=*number*

specifies a value by which to divide all coefficients so that fractional coefficients can be entered as integer numerators. For example, you can use

```
estimate '1/3(A1+A2) - 2/3A3' a 1 1 -2 / divisor=3;
```

instead of

```
estimate '1/3(A1+A2) - 2/3A3' a 0.33333 0.33333 -0.66667;
```

E

displays the entire $\mathbf{L}$ vector. This option is useful in confirming the ordering of parameters for specifying $\mathbf{L}$.

SINGULAR=*number*

tunes the estimability checking. If $\mathrm{ABS}(\mathbf{L} - \mathbf{LH}) > C \times \mathit{number}$, then the $\mathbf{L}$ vector is declared nonestimable. $\mathbf{H}$ is the $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$ matrix, and C is $\mathrm{ABS}(\mathbf{L})$ except for rows where $\mathbf{L}$ is zero, and then it is 1. The default value for the SINGULAR= option is 10^{-4}. Values for the SINGULAR= option must be between 0 and 1.

See also the section "Specification of ESTIMATE Expressions" on page 4227.

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a variable that provides frequencies for each observation in the DATA= data set. Specifically, if n is the value of the FREQ variable for a given observation, then that observation is used n times.

The analysis produced using a FREQ statement reflects the expanded number of observations. For example, means and total degrees of freedom reflect the expanded number of observations. You can produce the same analysis (without the FREQ statement) by first creating a new data set that contains the expanded number of observations. For example, if the value of the FREQ variable is 5 for the first observation, the first 5 observations in the new data set are identical. Each observation in the old data set is replicated n_i times in the new data set, where n_i is the value of the FREQ variable for that observation.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

If you specify the FREQ statement, it must appear before the first RUN statement or it is ignored.

ID Statement

ID *variables* **;**

When predicted values are requested as a MODEL statement option, values of the variables given in the ID statement are displayed beside each observed, predicted, and residual value for identification. Although there are no restrictions on the length of ID variables, PROC GLM might truncate the number of values listed in order to display them on one line. The GLM procedure displays a maximum of five ID variables.

If you specify the ID statement, it must appear before the first RUN statement or it is ignored.

LSMEANS Statement

LSMEANS *effects* < / *options* > **;**

Table 53.6 summarizes the *options* available in the LSMEANS statement.

Table 53.6 LSMEANS Statement Options

Option	Description
ADJUST=	Requests a multiple comparison adjustment
ALPHA=	Specifies the level of significance
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
AT	Enables you to modify the values of the covariates
BYLEVEL	Processes the OM data set by each level of the LS-mean effect
CL	Requests confidence limits
COV	Includes variances and covariances of the LS-means
E	Displays the coefficients of the linear functions
E=	Specifies an effect in the model to use as an error term
ETYPE=	Specifies the type of the E= effect
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
NOPRINT	Suppresses the normal display of results
OBSMARGINS	Specifies a potentially different weighting scheme
OUT=	Creates an output data set
PDIFF	Requests that *p*-values for differences
PLOT=	Requests graphics related to least squares means
SLICE=	Specifies effects within which to test for differences
SINGULAR=	Tunes the estimability checking
STDERR	Produces the standard error
TDIFF	Produces the *t* values

Least squares means (LS-means) are computed for each *effect* listed in the LSMEANS statement. You can specify only classification effects in the LSMEANS statement—that is, effects that contain only classification variables. You can also specify options to perform multiple comparisons. In contrast to the MEANS statement, the LSMEANS statement performs multiple comparisons on interactions as well as main effects.

LS-means are *predicted population margins*; that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs. Each LS-mean is computed as $\mathbf{L}'\mathbf{b}$ for a certain column vector $\mathbf{L}$, where $\mathbf{b}$ is the vector of parameter estimates—that is, the solution of the normal equations. For further information, see the section "Construction of Least Squares Means" on page 4244.

Multiple effects can be specified in one LSMEANS statement, or multiple LSMEANS statements can be used, but they must all appear after the MODEL statement. For example:

```
proc glm;
   class A B;
   model Y=A B A*B;
   lsmeans A B A*B;
run;
```

LS-means are displayed for each level of the A, B, and A*B effects.

You can specify the following *options* in the LSMEANS statement after a slash (/):

ADJUST=BON
ADJUST=DUNNETT
ADJUST=NELSON
ADJUST=SCHEFFE
ADJUST=SIDAK
ADJUST=SIMULATE <(*simoptions*)>
ADJUST=SMM | GT2
ADJUST=TUKEY
ADJUST=T

requests a multiple comparison adjustment for the p-values and confidence limits for the differences of LS-means. The ADJUST= option modifies the results of the TDIFF and PDIFF options; thus, if you omit the TDIFF or PDIFF option then the ADJUST= option implies PDIFF. By default, PROC GLM analyzes all pairwise differences. If you specify ADJUST=DUNNETT, PROC GLM analyzes all differences with a control level. If you specify the ADJUST=NELSON option, PROC GLM analyzes all differences with the average LS-mean. The default is ADJUST=T, which really signifies no adjustment for multiple comparisons.

The BON (Bonferroni) and SIDAK adjustments involve correction factors described in the section "Multiple Comparisons" on page 4231 and in Chapter 86, "The MULTTEST Procedure." When you specify ADJUST=TUKEY and your data are unbalanced, PROC GLM uses the approximation described in Kramer (1956) and identifies the adjustment as "Tukey-Kramer" in the results. Similarly, when you specify either ADJUST=DUNNETT or the ADJUST=NELSON option and the LS-means are correlated, PROC GLM uses the factor-analytic covariance approximation described in Hsu (1992) and identifies the adjustment in the results as "Dunnett-Hsu" or "Nelson-Hsu," respectively. The preceding references also describe the SCHEFFE and SMM adjustments.

The SIMULATE adjustment computes the adjusted p-values from the simulated distribution of the maximum or maximum absolute value of a multivariate t random vector. The simulation estimates q, the true $(1-\alpha)$ quantile, where $1-\alpha$ is the confidence coefficient. The default α is the value of the ALPHA= option in the PROC GLM statement or 0.05 if that option is not specified. You can change this value with the ALPHA= option in the LSMEANS statement.

The number of samples for the SIMULATE adjustment is set so that the tail area for the simulated q is within a certain *accuracy radius* γ of $1-\alpha$ with an *accuracy confidence* of $100(1-\epsilon)\%$. In equation form,

$$P(|F(\hat{q}) - (1-\alpha)| \leq \gamma) \quad = \quad 1-\epsilon$$

where $\hat{q}$ is the simulated q and F is the true distribution function of the maximum; see Edwards and Berry (1987) for details. By default, $\gamma = 0.005$ and $\epsilon = 0.01$, so that the tail area of $\hat{q}$ is within 0.005 of 0.95 with 99% confidence.

You can specify the following *simoptions* in parentheses after the ADJUST=SIMULATE option.

ACC=*value*
specifies the target accuracy radius γ of a $100(1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. The default value is ACC=0.005. Note that, if you also specify the CVADJUST *simoption*, then the actual accuracy radius will probably be substantially less than this target.

CVADJUST
specifies that the quantile should be estimated by the control variate adjustment method of Hsu and Nelson (1998) instead of simply as the quantile of the simulated sample. Specifying the CVADJUST option typically has the effect of significantly reducing the accuracy radius γ of a $100 \times (1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. The control-variate-adjusted quantile estimate takes roughly twice as long to compute, but it is typically much more accurate than the sample quantile.

EPS=*value*
specifies the value ϵ for a $100 \times (1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. The default value for the accuracy confidence is 99%, corresponding to EPS=0.01.

NSAMP=*n*
specifies the sample size for the simulation. By default, *n* is set based on the values of the target accuracy radius γ and accuracy confidence $100 \times (1-\epsilon)\%$ for an interval for the true probability content of the estimated $(1-\alpha)$ quantile. With the default values for γ, ϵ, and α (0.005, 0.01, and 0.05, respectively), NSAMP=12604 by default.

REPORT
specifies that a report on the simulation should be displayed, including a listing of the parameters, such as γ, ϵ, and α, as well as an analysis of various methods for estimating or approximating the quantile.

SEED=*number*
specifies an integer used to start the pseudo-random number generator for the simulation. If you do not specify a seed, or specify a value less than or equal to zero, the seed is by default generated from reading the time of day from the computer's clock.

THREADS
specifies that the computational work for the simulation be divided into parallel threads, where the number of threads is the value of the SAS system option CPUCOUNT=. For large simulations (as specified directly using the NSAMP= *simoption* or indirectly using the ACC= or EPS= *simoptions*), parallel processing can markedly speed up the computation of adjusted *p*-values and confidence intervals. However, because the parallel processing has different pseudo-random number streams, the precise results are different from the default ones, which are computed in sequence rather than in parallel. This option overrides the SAS system option THREADS | NOTHREADS.

NOTHREADS

specifies that the computational work for the simulation be performed in sequence rather than in parallel. NOTHREADS is the default. This option overrides the SAS system option THREADS | NOTHREADS.

ALPHA=*p*

specifies the level of significance *p* for $100(1-p)\%$ confidence intervals. This option is useful only if you also specify the CL option, and, optionally, the PDIFF option. By default, *p* is equal to the value of the ALPHA= option in the PROC GLM statement or 0.05 if that option is not specified, This value is used to set the endpoints for confidence intervals for the individual means as well as for differences between means.

ASC

ASCENDING

sorts LS-means in ascending order in results that you display by specifying the LINES option.

AT *variable* **=** *value*

AT (*variable-list***)= (***value-list***)**

AT MEANS

enables you to modify the values of the covariates used in computing

LS-means. By default, all covariate effects are set equal to their mean values for computation of standard LS-means. The AT option enables you to set the covariates to whatever values you consider interesting. For more information, see the section "Setting Covariate Values" on page 4245.

BYLEVEL

requests that PROC GLM process the OM data set by each level of the LS-mean effect in question. For more details, see the entry for the OM option in this section.

CL

requests confidence limits for the individual LS-means. If you specify the PDIFF option, confidence limits for differences between means are produced as well. You can control the confidence level with the ALPHA= option. Note that, if you specify an ADJUST= option, the confidence limits for the differences are adjusted for multiple inference but the confidence intervals for individual means are **not** adjusted.

COV

includes variances and covariances of the LS-means in the output data set specified in the OUT= option in the LSMEANS statement. Note that this is the covariance matrix for the LS-means themselves, not the covariance matrix for the differences between the LS-means, which is used in the PDIFF computations. If you omit the OUT= option, the COV option has no effect. When you specify the COV option, you can specify only one effect in the LSMEANS statement.

E

displays the coefficients of the linear functions used to compute the LS-means.

E=*effect*

specifies an effect in the model to use as an error term. The procedure uses the mean square for the *effect* as the error mean square when calculating estimated standard errors (requested with the STDERR option) and probabilities (requested with the STDERR, PDIFF, or TDIFF option). Unless you specify STDERR, PDIFF or TDIFF, the E= option is ignored. By default, if you specify the STDERR, PDIFF, or TDIFF option and do not specify the E= option, the procedure uses the error mean square for calculating standard errors and probabilities.

ETYPE=*n*

specifies the type (1, 2, 3, or 4, corresponding to a Type I, II, III, or IV test, respectively) of the E= effect. If you specify the E= option but not the ETYPE= option, the highest type computed in the analysis is used. If you omit the E= option, the ETYPE= option has no effect.

LINES

presents results of comparisons between all pairs of LS-means (as specified by the PDIFF=ALL option) by listing the LS-means in descending order and indicating nonsignificant subsets by line segments beside the corresponding LS-means. When all differences have the same variance, these comparison lines are guaranteed to accurately reflect the inferences that are based on the corresponding tests, which compare the respective *p*-values to the value of the ALPHA= option (0.05 by default). However, equal variances are rarely the case for differences between LS-means. If the variances are not all the same, then the comparison lines might be conservative, in the sense that if you base your inferences on the lines alone, you will detect fewer significant differences than the tests indicate. If there are any such differences, PROC GLM appends to the table a note that lists the pairs of LS-means that are inferred to be significantly different by the tests but not by the comparison lines. Note that in many cases, even though the variances are unbalanced, they are near enough that the comparison lines actually do reflect the test inferences accurately.

If ODS Graphics is enabled, then the LINES display is produced in graphical form by default. To display it in tabular form, use the LINESTABLE option.

LINESTABLE

presents the results that are described in the LINES option as a table.

NOPRINT

suppresses the normal display of results from the LSMEANS statement. This option is useful when an output data set is created with the OUT= option in the LSMEANS statement.

OBSMARGINS

OM

specifies a potentially different weighting scheme for computing LS-means coefficients. The standard LS-means have equal coefficients across classification effects; however, the OM option changes these coefficients to be proportional to those found in the input data set. For more information, see the section "Changing the Weighting Scheme" on page 4246.

The BYLEVEL option modifies the observed-margins LS-means. Instead of computing the margins across the entire data set, the procedure computes separate margins for each level of the LS-mean effect in question. The resulting LS-means are actually equal to raw means in this case. If you specify the BYLEVEL option, it disables the AT option.

OUT=*SAS-data-set*

creates an output data set that contains the values, standard errors, and, optionally, the covariances (see the COV option) of the LS-means.

For more information, see the section "Output Data Sets" on page 4264.

PDIFF< *=difftype* >

DIFF< *=difftype* >

requests that *p*-values for differences of the LS-means be produced. The optional *difftype* specifies which differences to display. Possible values for *difftype* are ALL, CONTROL, CONTROLL, CONTROLU, and ANOM. The ALL value requests all pairwise differences, and it is the default. The CONTROL value requests the differences with a control that, by default, is the first level of each of the specified LS-mean effects. The ANOM value requests differences between each LS-mean and the average LS-mean, as in the *analysis of means* (Ott 1967). The average is computed as a weighted mean of the LS-means, the weights being inversely proportional to the variances. Note that the ANOM procedure in SAS/QC software implements both tables and graphics for the analysis of means with a variety of response types. For one-way designs, the PDIFF=ANOM computations are equivalent to the results of PROC ANOM. See the section "Analysis of Means: Comparing Each Treatments to the Average" on page 4237 for more details.

To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the keyword CONTROL. For example, if the effects A, B, and C are CLASS variables, each having two levels, '1' and '2', the following LSMEANS statement specifies the '1' '2' level of A*B and the '2' '1' level of B*C as controls:

```
lsmeans A*B B*C / pdiff=control('1' '2', '2' '1');
```

For multiple-effect situations such as this one, the ordering of the list is significant, and you should check the output to make sure that the controls are correct.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*.

- **CONTROLL** tests whether the noncontrol levels are less than the control; you declare a noncontrol level to be significantly less than the control if the associated upper confidence limit for the noncontrol level minus the control is less than zero, and you ignore the associated lower confidence limits (which are set to minus infinity).
- **CONTROLU** tests whether the noncontrol levels are greater than the control; you declare a noncontrol level to be significantly greater than the control if the associated lower confidence limit for the noncontrol level minus the control is greater than zero, and you ignore the associated upper confidence limits (which are set to infinity).

The default multiple comparisons adjustment for each *difftype* is shown in the following table.

difftype	**Default ADJUST=**
Not specified	T
ALL	TUKEY
CONTROL **CONTROLL** **CONTROLU**	DUNNETT
ANOM	NELSON

If no *difftype* is specified, the default for the ADJUST= option is T (that is, no adjustment); for PDIFF=ALL, ADJUST=TUKEY is the default; for PDIFF=CONTROL, PDIFF=CONTROLL, or PDIFF=CONTROLU, the default value for the ADJUST= option is DUNNETT. For PDIFF=ANOM, ADJUST=NELSON is the default. If there is a conflict between the PDIFF= and ADJUST= options, the ADJUST= option takes precedence.

For example, in order to compute one-sided confidence limits for differences with a control, adjusted according to Dunnett's procedure, the following statements are equivalent:

```
lsmeans Treatment / pdiff=controll cl;
lsmeans Treatment / pdiff=controll cl adjust=dunnett;
```

PLOT | **PLOTS**< =*plot-request*< (*options*) > >

PLOT | **PLOTS**< =(*plot-request*< (*options*) > < ... *plot-request*< (*options*) > >) >

requests that graphics related to least squares means be produced via ODS Graphics, provided that ODS Graphics is enabled and the *plot-request* does not conflict with other options in the LSMEANS statement. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

The available options and suboptions are as follows:

ALL

requests that the default plots that correspond to this LSMEANS statement be produced. The default plot depends on the options in the statement.

ANOMPLOT

ANOM

requests an analysis-of-means display in which least squares means are compared to an average least squares mean. Least squares mean ANOM plots are produced only for those model effects that are listed in LSMEANS statements and have options that do not contradict with the display. For example, the following statements produce analysis-of-mean plots for effects A and C:

```
lsmeans A / diff=anom plot=anom;
lsmeans B / diff      plot=anom;
lsmeans C /           plot=anom;
```

The PDIFF option in the second LSMEANS statement implies all pairwise differences.

CONTROLPLOT

CONTROL

requests a display in which least squares means are visually compared against a reference level. These plots are produced only for statements with options that are compatible with control differences. For example, the following statements produce control plots for effects A and C:

```
lsmeans A / diff=control('1') plot=control;
lsmeans B / diff              plot=control;
lsmeans C                     plot=control;
```

The PDIFF option in the second LSMEANS statement implies all pairwise differences.

DIFFPLOT< (*diffplot-options*) >

DIFFOGRAM< (*diffplot-options*) >

requests a display of all pairwise least squares mean differences and their significance. The display is also known as a "mean-mean scatter plot" when it is based on arithmetic means (Hsu 1996; Hsu and Peruggia 1994). For each comparison a line segment, centered at the LS-means in the pair, is drawn. The length of the segment corresponds to the projected width of a confidence interval for the least squares mean difference. Segments that fail to cross the 45-degree reference line correspond to significant least squares mean differences.

LS-mean difference plots are produced only for statements with options that are compatible with the display. For example, the following statements request differences against a control level for the A effect, all pairwise differences for the B effect, and the least squares means for the C effect:

```
lsmeans A / diff=control('1') plot=diffplot;
lsmeans B / diff              plot=diffplot;
lsmeans C                     plot=diffplot;
```

The PDIFF= type in the first statement is incompatible with a display of all pairwise differences.

You can specify the following *diffplot-options*:

ABS

determines the positioning of the line segments in the plot. This is the default *diffplot-options*. When the ABS option is in effect, all line segments are shown on the same side of the reference line.

NOABS

determines the positioning of the line segments in the plot. The NOABS option separates comparisons according to the sign of the difference.

CENTER

marks the center point for each comparison. This point corresponds to the intersection of two least squares means.

NOLINES
suppresses the display of the line segments that represent the confidence bounds for the differences of the least squares means. The NOLINES option implies the CENTER option. The default is to draw line segments in the upper portion of the plot area without marking the center point.

LINESPLOT< (WSCALE=*wfactor* HSCALE=*hfactor*) >
modifies the dimensions of the LS-means comparison plot that is produced by the LINES option. The default dimensions of the plot vary depending on aspects such as the number of groups, the number of CLASS variables in the effect, and the number of parallel lines that are needed to represent comparisons. You can change the defaults by specifying the following options:

HSCALE=*hfactor*
scales the default height of the plot by *hfactor*, which must be a positive number. For example, specifying HSCALE=2 makes the plot twice as high as it would be by default.

WSCALE=*wfactor*
scales the default width of the plot by *wfactor*, which must be a positive number. For example, specifying WSCALE=2 makes the plot twice as wide as it would be by default.

MEANPLOT< (*meanplot-options*) >
requests displays of the least squares means.

The following *meanplot-options* control the display of the least squares means.

ASCENDING
displays the least squares means in ascending order. This option has no effect if means are displayed in separate plots.

CL
displays upper and lower confidence limits for the least squares means. By default, 95% limits are drawn. You can change the confidence level with the ALPHA= option. Confidence limits are drawn by default if the CL option is specified in the LSMEANS statement.

CLBAND
displays confidence limits as bands. This option implies the JOIN option.

DESCENDING
displays the least squares means in descending order. This option has no effect if means are displayed in separate plots.

ILINK
requests that means (and confidence limits) be displayed on the inverse linked scale.

JOIN
CONNECT
connects the least squares means with lines. This option is implied by the CLBAND option.

NONE
requests that no plots be produced.

When LS-mean calculations are adjusted for multiplicity by using the ADJUST= option, the plots are adjusted accordingly.

SLICE=*fixed-effect* | (*fixed-effects*)
specifies effects within which to test for differences between interaction LS-mean effects. This can produce what are known as tests of simple effects (Winer 1971). For example, suppose that A*B is significant and you want to test for the effect of A within each level of B. The appropriate LSMEANS statement is

```
lsmeans A*B / slice=B;
```

This statement tests for the simple main effects of A for B, which are calculated by extracting the appropriate rows from the coefficient matrix for the A*B LS-means and using them to form an F test as performed by the CONTRAST statement.

SINGULAR=*number*
tunes the estimability checking. If $\mathrm{ABS}(\mathbf{L} - \mathbf{LH}) > C \times \mathit{number}$ for any row, then $\mathbf{L}$ is declared nonestimable. $\mathbf{H}$ is the $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$ matrix, and C is $\mathrm{ABS}(\mathbf{L})$ except for rows where $\mathbf{L}$ is zero, and then it is 1. The default value for the SINGULAR= option is 10^{-4}. Values for the SINGULAR= option must be between 0 and 1.

STDERR
produces the standard error of the LS-means and the probability level for the hypothesis H_0: LS-mean $= 0$.

TDIFF
produces the t values for all hypotheses H_0: LS-mean(i) = LS-mean(j) and the corresponding probabilities.

MANOVA Statement

MANOVA < *test-options* > < / *detail-options* > **;**

If the MODEL statement includes more than one dependent variable, you can perform multivariate analysis of variance with the MANOVA statement. The *test-options* define which effects to test, while the *detail-options* specify how to execute the tests and what results to display. Table 53.7 summarizes the *options* available in the MANOVA statement.

Table 53.7 MANOVA Statement Options

Option	Description
Test Options	
H=	Specifies hypothesis effects
E=	Specifies the error effect
M=	Specifies a transformation matrix for the dependent variables

Table 53.7 *continued*

Option	Description
MNAMES=	Provides names for the transformed variables
PREFIX=	Alternatively identifies the transformed variables
Detail Options	
CANONICAL	Displays a canonical analysis of the **H** and **E** matrices
ETYPE=	Specifies the type of the **E** matrix
HTYPE=	Specifies the type of the **H** matrix
MSTAT=	Specifies the method of evaluating the multivariate test statistics
ORTH	Orthogonalizes the rows of the transformation matrix
PRINTE	Displays the error SSCP matrix **E**
PRINTH	Displays the hypothesis SSCP matrix **H**
SUMMARY	Produces analysis-of-variance tables for each dependent variable

When a MANOVA statement appears before the first RUN statement, PROC GLM enters a multivariate mode with respect to the handling of missing values; in addition to observations with missing independent variables, observations with *any* missing dependent variables are excluded from the analysis. If you want to use this mode of handling missing values and do not need any multivariate analyses, specify the MANOVA option in the PROC GLM statement.

If you use both the CONTRAST and MANOVA statements, the MANOVA statement must appear after the CONTRAST statement.

Test Options

The following *options* can be specified in the MANOVA statement as *test-options* in order to define which multivariate tests to perform.

H=*effects* **| INTERCEPT | _ALL_**

specifies effects in the preceding model to use as hypothesis matrices. For each **H** matrix (the SSCP matrix associated with an effect), the H= specification displays the characteristic roots and vectors of $\mathbf{E}^{-1}\mathbf{H}$ (where **E** is the matrix associated with the error effect), along with the Hotelling-Lawley trace, Pillai's trace, Wilks' lambda, and Roy's greatest root. By default, these statistics are tested with approximations based on the F distribution. To test them with exact (but computationally intensive) calculations, use the MSTAT=EXACT option.

Use the keyword INTERCEPT to produce tests for the intercept. To produce tests for all effects listed in the MODEL statement, use the keyword _ALL_ in place of a list of effects.

For background and further details, see the section "Multivariate Analysis of Variance" on page 4247.

E=*effect*

specifies the error effect. If you omit the E= specification, the GLM procedure uses the error SSCP (residual) matrix from the analysis.

M=*equation,. . . ,equation* | (*row-of-matrix,. . . ,row-of-matrix*)

specifies a transformation matrix for the dependent variables listed in the MODEL statement. The equations in the M= specification are of the form

$$\begin{aligned} c_1 \times \textit{dependent-variable} \quad &\pm \quad c_2 \times \textit{dependent-variable} \\ \cdots \quad &\pm \quad c_n \times \textit{dependent-variable} \end{aligned}$$

where the c_i values are coefficients for the various *dependent-variables*. If the value of a given c_i is 1, it can be omitted; in other words $1 \times Y$ is the same as Y. Equations should involve two or more dependent variables. For sample syntax, see the section "Examples" on page 4184.

Alternatively, you can input the transformation matrix directly by entering the elements of the matrix with commas separating the rows and parentheses surrounding the matrix. When this alternate form of input is used, the number of elements in each row must equal the number of dependent variables. Although these combinations actually represent the columns of the **M** matrix, they are displayed by rows.

When you include an M= specification, the analysis requested in the MANOVA statement is carried out for the variables defined by the equations in the specification, not the original dependent variables. If you omit the M= option, the analysis is performed for the original dependent variables in the MODEL statement.

If an M= specification is included without either the MNAMES= or PREFIX= option, the variables are labeled MVAR1, MVAR2, and so forth, by default. For further information, see the section "Multivariate Analysis of Variance" on page 4247.

MNAMES=*names*

provides names for the variables defined by the equations in the M= specification. Names in the list correspond to the M= equations or to the rows of the **M** matrix (as it is entered).

PREFIX=*name*

is an alternative means of identifying the transformed variables defined by the M= specification. For example, if you specify PREFIX=DIFF, the transformed variables are labeled DIFF1, DIFF2, and so forth.

Detail Options

You can specify the following *options* in the MANOVA statement after a slash (/) as *detail-options*.

CANONICAL

displays a canonical analysis of the **H** and **E** matrices (transformed by the **M** matrix, if specified) instead of the default display of characteristic roots and vectors.

ETYPE=*n*

specifies the type (1, 2, 3, or 4, corresponding to a Type I, II, III, or IV test, respectively) of the **E** matrix, the SSCP matrix associated with the E= effect. You need this option if you use the E= specification to specify an error effect other than residual error and you want to specify the type of sums of squares used for the effect. If you specify ETYPE=*n*, the corresponding test must have been performed in the MODEL statement, either by options SS*n*, E*n*, or the default Type I and Type III tests. By default, the procedure uses an ETYPE= value corresponding to the highest type (largest *n*) used in the analysis.

HTYPE=*n*

specifies the type (1, 2, 3, or 4, corresponding to a Type I, II, III, or IV test, respectively) of the **H** matrix. See the ETYPE= option for more details.

MSTAT=FAPPROX | EXACT

specifies the method of evaluating the multivariate test statistics. The default is MSTAT=FAPPROX, which specifies that the multivariate tests are evaluated using the usual approximations based on the F distribution, as discussed in the section "Multivariate Tests" in Chapter 4, "Introduction to Regression Procedures." Alternatively, you can specify MSTAT=EXACT to compute exact p-values for three of the four tests (Wilks' lambda, the Hotelling-Lawley trace, and Roy's greatest root) and an improved F approximation for the fourth (Pillai's trace). While MSTAT=EXACT provides better control of the significance probability for the tests, especially for Roy's greatest root, computations for the exact p-values can be appreciably more demanding, and are in fact infeasible for large problems (many dependent variables). Thus, although MSTAT=EXACT is more accurate for most data, it is not the default method. For more information about the results of MSTAT=EXACT, see the section "Multivariate Analysis of Variance" on page 4247.

ORTH

requests that the transformation matrix in the M= specification of the MANOVA statement be orthonormalized by rows before the analysis.

PRINTE

displays the error SSCP matrix **E**. If the **E** matrix is the error SSCP (residual) matrix from the analysis, the partial correlations of the dependent variables given the independent variables are also produced.

For example, the statement

```
manova / printe;
```

displays the error SSCP matrix and the partial correlation matrix computed from the error SSCP matrix.

PRINTH

displays the hypothesis SSCP matrix **H** associated with each effect specified by the H= specification.

SUMMARY

produces analysis-of-variance tables for each dependent variable. When no **M** matrix is specified, a table is displayed for each original dependent variable from the MODEL statement; with an **M** matrix other than the identity, a table is displayed for each transformed variable defined by the **M** matrix.

Examples

The following statements provide several examples of using a MANOVA statement.

```
proc glm;
   class A B;
   model Y1-Y5=A B(A) / nouni;
   manova h=A e=B(A) / printh printe htype=1 etype=1;
   manova h=B(A) / printe;
   manova h=A e=B(A) m=Y1-Y2,Y2-Y3,Y3-Y4,Y4-Y5
          prefix=diff;
   manova h=A e=B(A) m=(1 -1  0  0  0,
```

```
                          0  1 -1  0  0,
                          0  0  1 -1  0,
                          0  0  0  1 -1) prefix=diff;
run;
```

Since this MODEL statement requests no options for type of sums of squares, the procedure uses Type I and Type III sums of squares. The first MANOVA statement specifies A as the hypothesis effect and B(A) as the error effect. As a result of the PRINTH option, the procedure displays the hypothesis SSCP matrix associated with the A effect; and, as a result of the PRINTE option, the procedure displays the error SSCP matrix associated with the B(A) effect. The option HTYPE=1 specifies a Type I **H** matrix, and the option ETYPE=1 specifies a Type I **E** matrix.

The second MANOVA statement specifies B(A) as the hypothesis effect. Since no error effect is specified, PROC GLM uses the error SSCP matrix from the analysis as the **E** matrix. The PRINTE option displays this **E** matrix. Since the **E** matrix is the error SSCP matrix from the analysis, the partial correlation matrix computed from this matrix is also produced.

The third MANOVA statement requests the same analysis as the first MANOVA statement, but the analysis is carried out for variables transformed to be successive differences between the original dependent variables. The option PREFIX=DIFF labels the transformed variables as DIFF1, DIFF2, DIFF3, and DIFF4.

Finally, the fourth MANOVA statement has the identical effect as the third, but it uses an alternative form of the M= specification. Instead of specifying a set of equations, the fourth MANOVA statement specifies rows of a matrix of coefficients for the five dependent variables.

As a second example of the use of the M= specification, consider the following:

```
proc glm;
   class group;
   model dose1-dose4=group / nouni;
   manova h = group
          m = -3*dose1 -   dose2 +   dose3 + 3*dose4,
                 dose1 -   dose2 -   dose3 +   dose4,
                -dose1 + 3*dose2 - 3*dose3 +   dose4
          mnames = Linear Quadratic Cubic
          / printe;
run;
```

The M= specification gives a transformation of the dependent variables dose1 through dose4 into orthogonal polynomial components, and the MNAMES= option labels the transformed variables LINEAR, QUADRATIC, and CUBIC, respectively. Since the PRINTE option is specified and the default residual matrix is used as an error term, the partial correlation matrix of the orthogonal polynomial components is also produced.

MEANS Statement

MEANS *effects* < / *options* > **;**

Within each group corresponding to each effect specified in the MEANS statement, PROC GLM computes the arithmetic means and standard deviations of all continuous variables in the model (both dependent and independent). You can specify only classification effects in the MEANS statement—that is, effects that contain only classification variables.

Note that the arithmetic means are not adjusted for other effects in the model; for adjusted means, see the section "LSMEANS Statement" on page 4172.

If you use a WEIGHT statement, PROC GLM computes weighted means; see the section "Weighted Means" on page 4244.

You can also specify options to perform multiple comparisons. However, the MEANS statement performs multiple comparisons only for main-effect means; for multiple comparisons of interaction means, see the section "LSMEANS Statement" on page 4172.

You can use any number of MEANS statements, provided that they appear after the MODEL statement. For example, suppose A and B each have two levels. Then, if you use the statements

```
proc glm;
   class A B;
   model Y=A B A*B;
   means A B / tukey;
   means A*B;
run;
```

the means, standard deviations, and Tukey's multiple comparisons tests are displayed for each level of the main effects A and B, and just the means and standard deviations are displayed for each of the four combinations of levels for A*B. Since multiple comparisons tests apply only to main effects, the single MEANS statement

```
means A B A*B / tukey;
```

produces the same results.

PROC GLM does not compute means for interaction effects containing continuous variables. Thus, if you have the model

```
class A;
model Y=A X A*X;
```

then the effects X and A*X cannot be used in the MEANS statement. However, if you specify the effect A in the means statement

```
means A;
```

then PROC GLM, by default, displays within-A arithmetic means of both Y and X. You can use the DEPONLY option to display means of only the dependent variables.

```
means A / deponly;
```

If you use a WEIGHT statement, PROC GLM computes weighted means and estimates their variance as inversely proportional to the corresponding sum of weights (see the section "Weighted Means" on page 4244). However, note that the statistical interpretation of multiple comparison tests for weighted means is not well understood. See the section "Multiple Comparisons" on page 4231 for formulas. Table 53.8 summarizes the *options* available in the MEANS statement.

Table 53.8 MEANS Statement Options

Option	Description
Modify Output	
DEPONLY	Displays only means for the dependent variables
Perform Multiple Comparison Tests	
BON	Performs Bonferroni *t* tests
DUNCAN	Performs Duncan's multiple range test
DUNNETT	Performs Dunnett's two-tailed *t* test
DUNNETTL	Performs Dunnett's lower one-tailed *t* test
DUNNETTU	Performs Dunnett's upper one-tailed *t* test
GABRIEL	Performs Gabriel's multiple-comparison procedure
REGWQ	Performs the Ryan-Einot-Gabriel-Welsch multiple range test
SCHEFFE	Performs Scheffé's multiple-comparison procedure
SIDAK	Performs pairwise *t* tests on differences between means
SMM or GT2	Performs pairwise comparisons based on the studentized maximum modulus and Sidak's uncorrelated-*t* inequality
SNK	Performs the Student-Newman-Keuls multiple range test
T or LSD	Performs pairwise *t* tests
TUKEY	Performs Tukey's studentized range test (HSD)
WALLER	Performs the Waller-Duncan *k*-ratio *t* test
Specify Additional Details for Multiple Comparison Tests	
ALPHA=	Specifies the level of significance
CLDIFF	Presents confidence intervals for all pairwise differences between means
CLM	Presents results as intervals for the mean of each level of the variables
E=	Specifies the error mean square used in the multiple comparisons
ETYPE=	Specifies the type of mean square for the error effect
HTYPE=	Specifies the MS type for the hypothesis MS
KRATIO=	Specifies the Type 1/Type 2 error seriousness ratio
LINES	Displays the means in descending order and indicates nonsignificant subsets by line segments
LINESTABLE	Displays the results of the LINES option as a table
NOSORT	Prevents the means from being sorted into descending order
Test for Homogeneity of Variances	
HOVTEST	Requests a homogeneity of variance test
Compensate for Heterogeneous Variances	
WELCH	Requests the variance-weighted one-way ANOVA of Welch (1951)

The *options* available in the MEANS statement are described in the following list.

ALPHA=

ALPHA=*p* specifies the level of significance for comparisons among the means. By default, *p* is equal to the value of the ALPHA= option in the PROC GLM statement or 0.05 if that option is not specified. You can specify any value greater than 0 and less than 1.

BON

performs Bonferroni *t* tests of differences between means for all main-effect means in the MEANS statement. See the CLDIFF and LINES options for a discussion of how the procedure displays results.

CLDIFF

presents results of the BON, GABRIEL, SCHEFFE, SIDAK, SMM, GT2, T, LSD, and TUKEY options as confidence intervals for all pairwise differences between means, and the results of the DUNNETT, DUNNETTU, and DUNNETTL options as confidence intervals for differences with the control. The CLDIFF option is the default for unequal cell sizes unless the DUNCAN, REGWQ, SNK, or WALLER option is specified.

CLM

presents results of the BON, GABRIEL, SCHEFFE, SIDAK, SMM, T, and LSD options as intervals for the mean of each level of the variables specified in the MEANS statement. For all options except GABRIEL, the intervals are confidence intervals for the true means. For the GABRIEL option, they are *comparison intervals* for comparing means pairwise: in this case, if the intervals corresponding to two means overlap, then the difference between them is insignificant according to Gabriel's method.

DEPONLY

displays only means for the dependent variables. By default, PROC GLM produces means for all continuous variables, including continuous independent variables.

DUNCAN

performs Duncan's multiple range test on all main-effect means given in the MEANS statement. See the LINES option for a discussion of how the procedure displays results.

DUNNETT <(*formatted-control-values*)>

performs Dunnett's two-tailed *t* test, testing if any treatments are significantly different from a single control for all main-effect means in the MEANS statement.

To specify which level of the effect is the control, enclose the formatted value in quotes and parentheses after the *keyword*. If more than one effect is specified in the MEANS statement, you can use a list of control values within the parentheses. By default, the first level of the effect is used as the control. For example:

```
means A  / dunnett('CONTROL');
```

where CONTROL is the formatted control value of A. As another example:

```
means A B C / dunnett('CNTLA' 'CNTLB' 'CNTLC');
```

where CNTLA, CNTLB, and CNTLC are the formatted control values for A, B, and C, respectively.

DUNNETTL < (*formatted-control-value*) >
performs Dunnett's one-tailed *t* test, testing if any treatment is significantly less than the control. Control level information is specified as described for the DUNNETT option.

DUNNETTU < (*formatted-control-value*) >
performs Dunnett's one-tailed *t* test, testing if any treatment is significantly greater than the control. Control level information is specified as described for the DUNNETT option.

E=*effect*
specifies the error mean square used in the multiple comparisons. By default, PROC GLM uses the overall residual or error mean square (MS). The effect specified with the E= option must be a term in the model; otherwise, the procedure uses the residual MS.

ETYPE=*n*
specifies the type of mean square for the error effect. When you specify E=*effect*, you might need to indicate which type (1, 2, 3, or 4) of MS is to be used. The *n* value must be one of the types specified in or implied by the MODEL statement. The default MS type is the highest type used in the analysis.

GABRIEL
performs Gabriel's multiple-comparison procedure on all main-effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

GT2
See the SMM option.

HOVTEST
HOVTEST=BARTLETT
HOVTEST=BF
HOVTEST=LEVENE < (**TYPE= ABS** | **SQUARE**) >
HOVTEST=OBRIEN < (**W=***number*) >
requests a homogeneity of variance test for the groups defined by the MEANS effect. You can optionally specify a particular test; if you do not specify a test, Levene's test (Levene 1960) with TYPE=SQUARE is computed. Note that this option is ignored unless your MODEL statement specifies a simple one-way model.

The HOVTEST=BARTLETT option specifies Bartlett's test (Bartlett 1937), a modification of the normal-theory likelihood ratio test.

The HOVTEST=BF option specifies Brown and Forsythe's variation of Levene's test (Brown and Forsythe 1974).

The HOVTEST=LEVENE option specifies Levene's test (Levene 1960), which is widely considered to be the standard homogeneity of variance test. You can use the TYPE= option in parentheses to specify whether to use the absolute residuals (TYPE=ABS) or the squared residuals (TYPE=SQUARE) in Levene's test. TYPE=SQUARE is the default.

The HOVTEST=OBRIEN option specifies O'Brien's test (O'Brien 1979), which is basically a modification of HOVTEST=LEVENE(TYPE=SQUARE). You can use the W= option in parentheses to tune the variable to match the suspected kurtosis of the underlying distribution. By default, W=0.5, as suggested by O'Brien (1979, 1981).

See the section "Homogeneity of Variance in One-Way Models" on page 4243 for more details on these methods. Example 53.10 illustrates the use of the HOVTEST and WELCH options in the MEANS statement in testing for equal group variances and adjusting for unequal group variances in a one-way ANOVA.

HTYPE=*n*

specifies the MS type for the hypothesis MS. The HTYPE= option is needed only when the WALLER option is specified. The default HTYPE= value is the highest type used in the model.

KRATIO=*value*

specifies the Type 1/Type 2 error seriousness ratio for the Waller-Duncan test. Reasonable values for the KRATIO= option are 50, 100, 500, which roughly correspond for the two-level case to ALPHA levels of 0.1, 0.05, and 0.01, respectively. By default, the procedure uses the value of 100.

LINES

presents results of the BON, DUNCAN, GABRIEL, REGWQ, SCHEFFE, SIDAK, SMM, GT2, SNK, T, LSD, TUKEY, and WALLER options by listing the means in descending order and indicating nonsignificant subsets by line segments beside the corresponding means. The LINES option is appropriate for equal cell sizes, for which it is the default. The LINES option is also the default if the DUNCAN, REGWQ, SNK, or WALLER option is specified, or if there are only two cells of unequal size. The LINES option cannot be used in combination with the DUNNETT, DUNNETTL, or DUNNETTU option.

If ODS Graphics is enabled then the LINES display is produced in graphical form by default. To display it in tabular form, use the LINESTABLE option.

In addition, the procedure has a restriction that no more than 24 overlapping groups of means can exist. If a mean belongs to more than 24 groups, the procedure issues an error message. You can either reduce the number of levels of the variable or use a multiple comparison test that allows the CLDIFF option rather than the LINES option.

NOTE: If the cell sizes are unequal, the harmonic mean of the cell sizes is used to compute the critical ranges. This approach is reasonable if the cell sizes are not too different, but it can lead to liberal tests if the cell sizes are highly disparate. In this case, you should not use the LINES option for displaying multiple comparisons results; use the TUKEY and CLDIFF options instead.

LINESTABLE

requests that the results of the LINES option be presented as a table.

LSD

See the T option.

NOSORT

prevents the means from being sorted into descending order when the CLDIFF or CLM option is specified.

REGWQ

performs the Ryan-Einot-Gabriel-Welsch multiple range test on all main-effect means in the MEANS statement. See the LINES option for a discussion of how the procedure displays results.

SCHEFFE

erforms Scheffé's multiple-comparison procedure on all main-effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

SIDAK

performs pairwise *t* tests on differences between means with levels adjusted according to Sidak's inequality for all main-effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

SMM

GT2

performs pairwise comparisons based on the studentized maximum modulus and Sidak's uncorrelated-*t* inequality, yielding Hochberg's GT2 method when sample sizes are unequal, for all main-effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

SNK

performs the Student-Newman-Keuls multiple range test on all main-effect means in the MEANS statement. See the LINES option for discussions of how the procedure displays results.

T

LSD

performs pairwise *t* tests, equivalent to Fisher's least significant difference test in the case of equal cell sizes, for all main-effect means in the MEANS statement. See the CLDIFF and LINES options for discussions of how the procedure displays results.

TUKEY

performs Tukey's studentized range test (HSD) on all main-effect means in the MEANS statement. (When the group sizes are different, this is the Tukey-Kramer test.) See the CLDIFF and LINES options for discussions of how the procedure displays results.

WALLER

performs the Waller-Duncan *k*-ratio *t* test on all main-effect means in the MEANS statement. See the KRATIO= and HTYPE= options for information about controlling details of the test, and the LINES option for a discussion of how the procedure displays results.

WELCH

requests the variance-weighted one-way ANOVA of Welch (1951). This alternative to the usual analysis of variance for a one-way model is robust to the assumption of equal within-group variances. This option is ignored unless your MODEL statement specifies a simple one-way model.

Note that using the WELCH option merely produces one additional table consisting of Welch's ANOVA. It does not affect all of the other tests displayed by the GLM procedure, which still require the assumption of equal variance for exact validity.

See the section "Homogeneity of Variance in One-Way Models" on page 4243 for more details on Welch's ANOVA. Example 53.10 illustrates the use of the HOVTEST and WELCH options in the MEANS statement in testing for equal group variances and adjusting for unequal group variances in a one-way ANOVA.

MODEL Statement

MODEL *dependent-variables* **=** *independent-effects* < / *options* > **;**

The MODEL statement names the dependent variables and independent effects. The syntax of effects is described in the section "Specification of Effects" on page 4207. For any model effect involving classification variables (interactions as well as main effects), the number of levels cannot exceed 32,767. If no independent effects are specified, only an intercept term is fit. You can specify only one MODEL statement (in contrast to the REG procedure, for example, which allows several MODEL statements in the same PROC REG run).

Table 53.9 summarizes the *options* available in the MODEL statement.

Table 53.9 MODEL Statement Options

Option	Description
Produce Effect Size Information	
EFFECTSIZE	Adds measures of effect size to each analysis of variance table
Produce Tests for the Intercept	
INTERCEPT	Produces the hypothesis tests associated with the intercept
Omit the Intercept Parameter from Model	
NOINT	Omits the intercept parameter from the model
Produce Parameter Estimates	
SOLUTION	Produces parameter estimates
Produce Tolerance Analysis	
TOLERANCE	Displays the tolerances used in the SWEEP routine
Suppress Univariate Tests and Output	
NOUNI	Suppresses the display of univariate statistics
Display Estimable Functions	
E	Displays the general form of all estimable functions
E1	Displays the Type I estimable functions and computes sums of squares
E2	Displays the Type II estimable functions and computes sums of squares
E3	Displays the Type III estimable functions and computes sums of squares
E4	Displays the Type IV estimable functions computes sums of squares
ALIASING	Specifies that the estimable functions should be displayed as an *aliasing structure*
Control Hypothesis Tests Performed	
SS1	Displays Type I sum of squares
SS2	Displays Type II sum of squares
SS3	Displays Type III sum of squares
SS4	Displays Type IV sum of squares
Produce Confidence Intervals	
ALPHA=	Specifies the level of significance
CLI	Produces confidence limits for individual predicted values
CLM	Produces confidence limits for a mean predicted value
CLPARM	Produces confidence limits for the parameter estimates

Table 53.9 *continued*

Option	Description
Display Predicted and Residual Values	
P	Displays observed, predicted, and residual values
Display Intermediate Calculations	
INVERSE	Displays the augmented inverse (or generalized inverse) $\mathbf{X}'\mathbf{X}$ matrix
XPX	Displays the augmented $\mathbf{X}'\mathbf{X}$ crossproducts matrix
Tune Sensitivity	
SINGULAR=	Tunes the sensitivity of the regression routine to linear dependencies
ZETA=	Tunes the sensitivity of the check for estimability for Type III and Type IV functions

The *options* available in the MODEL statement are described in the following list.

ALIASING

specifies that the estimable functions should be displayed as an *aliasing structure*, for which each row says which linear combination of the parameters is estimated by each estimable function; also, this option adds a column of the same information to the table of parameter estimates, giving for each parameter the expected value of the estimate associated with that parameter. This option is most useful in fractional factorial experiments that can be analyzed without a CLASS statement.

ALPHA=*p*

specifies the level of significance p for $100(1-p)\%$ confidence intervals. By default, p is equal to the value of the ALPHA= option in the PROC GLM statement, or 0.05 if that option is not specified. You can use values between 0 and 1.

CLI

produces confidence limits for individual predicted values for each observation. The CLI option is ignored if the CLM option is also specified.

CLM

produces confidence limits for a mean predicted value for each observation.

CLPARM

produces confidence limits for the parameter estimates (if the SOLUTION option is also specified) and for the results of all ESTIMATE statements.

E

displays the general form of all estimable functions. This is useful for determining the order of parameters when you are writing CONTRAST and ESTIMATE statements.

E1

displays the Type I estimable functions for each effect in the model and computes the corresponding sums of squares.

E2

displays the Type II estimable functions for each effect in the model and computes the corresponding sums of squares.

E3

displays the Type III estimable functions for each effect in the model and computes the corresponding sums of squares.

E4

displays the Type IV estimable functions for each effect in the model and computes the corresponding sums of squares.

EFFECTSIZE

adds measures of effect size to each analysis of variance table displayed by the procedure, except for those displayed by the TEST option in the RANDOM statement, by CONTRAST and TEST statements with the E= option, and by MANOVA and REPEATED statements. The effect size measures include the intraclass correlation and both estimates and confidence intervals for the noncentrality for the F test, as well as for the semipartial and partial correlation ratios. For more information about the computation and interpretation of these measures, see the section "Effect Size Measures for F Tests in GLM" on page 4220.

INTERCEPT

INT

produces the hypothesis tests associated with the intercept as an effect in the model. By default, the procedure includes the intercept in the model but does not display associated tests of hypotheses. Except for producing the uncorrected total sum of squares instead of the corrected total sum of squares, the INT option is ignored when you use an ABSORB statement.

INVERSE

I

displays the augmented inverse (or generalized inverse) $\mathbf{X}'\mathbf{X}$ matrix:

$$\begin{bmatrix} (\mathbf{X}'\mathbf{X})^- & (\mathbf{X}'\mathbf{X})^-\mathbf{X}'\mathbf{y} \\ \mathbf{y}'\mathbf{X}(\mathbf{X}'\mathbf{X})^- & \mathbf{y}'\mathbf{y} - \mathbf{y}'\mathbf{X}(\mathbf{X}'\mathbf{X})^-\mathbf{X}'\mathbf{y} \end{bmatrix}$$

The upper-left corner is the generalized inverse of $\mathbf{X}'\mathbf{X}$, the upper-right corner is the parameter estimates, and the lower-right corner is the error sum of squares.

NOINT

omits the intercept parameter from the model. The NOINT option is ignored when you use an ABSORB statement.

NOUNI

suppresses the display of univariate statistics. You typically use the NOUNI option with a multivariate or repeated measures analysis of variance when you do not need the standard univariate results. The NOUNI option in a MODEL statement does not affect the univariate output produced by the REPEATED statement.

P

displays observed, predicted, and residual values for each observation that does not contain missing values for independent variables. The Durbin-Watson statistic is also displayed when the P option is specified. The PRESS statistic is also produced if either the CLM or CLI option is specified.

SINGULAR=*number*

tunes the sensitivity of the regression routine to linear dependencies in the design. If a diagonal pivot element is less than $C \times number$ as PROC GLM sweeps the $\mathbf{X}'\mathbf{X}$ matrix, the associated design column is declared to be linearly dependent with previous columns, and the associated parameter is zeroed.

The C value adjusts the check to the relative scale of the variable. The C value is equal to the corrected sum of squares for the variable, unless the corrected sum of squares is 0, in which case C is 1. If you specify the NOINT option but not the ABSORB statement, PROC GLM uses the uncorrected sum of squares instead.

The default value of the SINGULAR= option, which is approximately 10^{-9} on most machines, might be too small, but this value is necessary in order to handle the high-degree polynomials used in the literature to compare regression routines.

SOLUTION

produces a solution to the normal equations (parameter estimates). PROC GLM displays a solution by default when your model involves no classification variables, so you need this option only if you want to see the solution for models with classification effects.

SS1

displays the sum of squares associated with Type I estimable functions for each effect. These are also displayed by default.

SS2

displays the sum of squares associated with Type II estimable functions for each effect.

SS3

displays the sum of squares associated with Type III estimable functions for each effect. These are also displayed by default.

SS4

displays the sum of squares associated with Type IV estimable functions for each effect.

TOLERANCE

displays the tolerances used in the SWEEP routine. The tolerances are of the form C/USS or C/CSS, as described in the discussion of the SINGULAR= option. The tolerance value for the intercept is not divided by its uncorrected sum of squares.

XPX

displays the augmented $\mathbf{X}'\mathbf{X}$ crossproducts matrix:

$$\begin{bmatrix} \mathbf{X}'\mathbf{X} & \mathbf{X}'\mathbf{y} \\ \mathbf{y}'\mathbf{X} & \mathbf{y}'\mathbf{y} \end{bmatrix}$$

ZETA=*value*

tunes the sensitivity of the check for estimability for Type III and Type IV functions. Any element in the estimable function basis with an absolute value less than the ZETA= option is set to zero. The default value for the ZETA= option is 10^{-8}.

Although it is possible to generate data for which this absolute check can be defeated, the check suffices in most practical examples. Additional research is needed in order to make this check relative rather than absolute.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > *keyword***=***names* < . . . *keyword***=***names* > < *option* > **;**

The OUTPUT statement creates a new SAS data set that saves diagnostic measures calculated after fitting the model. At least one specification of the form *keyword***=***names* is required.

All the variables in the original data set are included in the new data set, along with variables created in the OUTPUT statement. These new variables contain the values of a variety of diagnostic measures that are calculated for each observation in the data set. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

Details on the specifications in the OUTPUT statement follow.

*keyword***=***names*

specifies the statistics to include in the output data set and provides names to the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), an equal sign, and the variable or variables to contain the statistic.

In the output data set, the first variable listed after a *keyword* in the OUTPUT statement contains that statistic for the first dependent variable listed in the MODEL statement; the second variable contains the statistic for the second dependent variable in the MODEL statement, and so on. The list of variables following the equal sign can be shorter than the list of dependent variables in the MODEL statement. In this case, the procedure creates the new names in order of the dependent variables in the MODEL statement. See the section "Examples" on page 4198.

The *keywords* allowed and the statistics they represent are as follows:

COOKD

Cook's *D* influence statistic

COVRATIO

standard influence of observation on covariance of parameter estimates

DFFITS

standard influence of observation on predicted value

H

leverage, $h_i = x_i(\mathbf{X}'\mathbf{X})^{-1}x_i'$

LCL

lower bound of a 100(1 - p)% confidence interval for an individual prediction. The p-level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the ALPHA= option in the PROC GLM statement. If neither of these options is set, then p = 0.05 by default, resulting in the lower bound for a 95% confidence interval. The interval also depends on the variance of the error, as well as the variance of the parameter estimates. For the corresponding upper bound, see the UCL keyword.

LCLM

lower bound of a 100(1 - p)% confidence interval for the expected value (mean) of the predicted value. The p-level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the ALPHA= option in the PROC GLM statement. If neither of these options is set, then p = 0.05 by default, resulting in the lower bound for a 95% confidence interval. For the corresponding upper bound, see the UCLM keyword.

PREDICTED | P

predicted values

PRESS

residual for the ith observation that results from dropping it and predicting it on the basis of all other observations. This is the residual divided by $(1 - h_i)$, where h_i is the leverage, defined previously.

RESIDUAL | R

residuals, calculated as ACTUAL – PREDICTED

RSTUDENT

a studentized residual with the current observation deleted

STDI

standard error of the individual predicted value

STDP

standard error of the mean predicted value

STDR

standard error of the residual

STUDENT

studentized residuals, the residual divided by its standard error

UCL

upper bound of a 100(1 - p)% confidence interval for an individual prediction. The p-level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the ALPHA= option in the PROC GLM statement. If neither of these options is set, then p = 0.05 by default, resulting in the upper bound for a 95% confidence interval. The interval also depends on the variance of the error, as well as the variance of the parameter estimates. For the corresponding lower bound, see the LCL keyword.

UCLM

upper bound of a 100(1 - p)% confidence interval for the expected value (mean) of the predicted value. The p-level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the ALPHA= option in the PROC GLM statement. If neither of these options is set, then $p = 0.05$ by default, resulting in the upper bound for a 95% confidence interval. For the corresponding lower bound, see the LCLM keyword.

OUT=*SAS-data-set*

gives the name of the new data set. By default, the procedure uses the DATA*n* convention to name the new data set.

The following *option* is available in the OUTPUT statement and is specified after a slash (/):

ALPHA=*p*

specifies the level of significance p for $100(1 - p)\%$ confidence intervals. By default, p is equal to the value of the ALPHA= option in the PROC GLM statement or 0.05 if that option is not specified. You can use values between 0 and 1.

See Chapter 4, "Introduction to Regression Procedures," and the section "Influence Statistics" on page 8810 in Chapter 105, "The REG Procedure," for details on the calculation of these statistics.

Examples

The following statements show the syntax for creating an output data set with a single dependent variable.

```
proc glm;
   class a b;
   model y=a b a*b;
   output out=new p=yhat r=resid stdr=eresid;
run;
```

These statements create an output data set named new. In addition to all the variables from the original data set, new contains the variable yhat, with values that are predicted values of the dependent variable y; the variable resid, with values that are the residual values of y; and the variable eresid, with values that are the standard errors of the residuals.

The following statements show a situation with five dependent variables.

```
proc glm;
   by group;
   class a;
   model y1-y5=a x(a);
   output out=pout predicted=py1-py5;
run;
```

The data set pout contains five new variables, py1 through py5. The values of py1 are the predicted values of y1; the values of py2 are the predicted values of y2; and so on.

For more information about the data set produced by the OUTPUT statement, see the section "Output Data Sets" on page 4264.

RANDOM Statement

RANDOM *effects* < / *options* > **;**

When some model effects are random (that is, assumed to be sampled from a normal population of effects), you can specify these effects in the RANDOM statement in order to compute the expected values of mean squares for various model effects and contrasts and, optionally, to perform random effects analysis of variance tests. You can use as many RANDOM statements as you want, provided that they appear after the MODEL statement. If you use a CONTRAST statement with a RANDOM statement and you want to obtain the expected mean squares for the contrast hypothesis, you must enter the CONTRAST statement before the RANDOM statement.

NOTE: PROC GLM uses only the information pertaining to expected mean squares when you specify the TEST option in the RANDOM statement and, even then, only in the extra *F* tests produced by the RANDOM statement. Other features in the GLM procedure—including the results of the LSMEANS and ESTIMATE statements—assume that all effects are fixed, so that all tests and estimability checks for these statements are based on a fixed-effects model, even when you use a RANDOM statement. Therefore, you should use the MIXED procedure to compute tests involving these features that take the random effects into account; see the section "PROC GLM versus PROC MIXED for Random-Effects Analysis" on page 4257 and Chapter 84, "The MIXED Procedure," for more information.

When you use the RANDOM statement, by default the GLM procedure produces the Type III expected mean squares for model effects and for contrasts specified before the RANDOM statement in the program statements. In order to obtain expected values for other types of mean squares, you need to specify which types of mean squares are of interest in the MODEL statement. See the section "Computing Type I, II, and IV Expected Mean Squares" on page 4259 for more information.

The list of effects in the RANDOM statement should contain one or more of the pure classification effects specified in the MODEL statement (that is, main effects, crossed effects, or nested effects involving only classification variables). The coefficients corresponding to each effect specified are assumed to be normally and independently distributed with common variance. Levels in different effects are assumed to be independent.

You can specify the following *options* in the RANDOM statement after a slash (/):

Q

displays all quadratic forms in the fixed effects that appear in the expected mean squares. For some designs, such as large mixed-level factorials, the Q option might generate a substantial amount of output.

TEST

performs hypothesis tests for each effect specified in the model, using appropriate error terms as determined by the expected mean squares.

CAUTION: PROC GLM does not automatically declare interactions to be random when the effects in the interaction are declared random. For example,

```
random a b / test;
```

does not produce the same expected mean squares or tests as

```
random a b a*b / test;
```

To ensure correct tests, you need to list all random interactions and random main effects in the RANDOM statement.

See the section "Random-Effects Analysis" on page 4257 for more information about the calculation of expected mean squares and tests and on the similarities and differences between the GLM and MIXED procedures. See Chapter 5, "Introduction to Analysis of Variance Procedures," and Chapter 84, "The MIXED Procedure," for more information about random effects.

REPEATED Statement

REPEATED *factor-specification* < / *options* > **;**

When values of the dependent variables in the MODEL statement represent repeated measurements on the same experimental unit, the REPEATED statement enables you to test hypotheses about the measurement factors (often called *within-subject factors*) as well as the interactions of within-subject factors with independent variables in the MODEL statement (often called *between-subject factors*). The REPEATED statement provides multivariate and univariate tests as well as hypothesis tests for a variety of single-degree-of-freedom contrasts. There is no limit to the number of within-subject factors that can be specified.

The REPEATED statement is typically used for handling repeated measures designs with one repeated response variable. Usually, the variables on the left-hand side of the equation in the MODEL statement represent one repeated response variable. This does not mean that only one factor can be listed in the REPEATED statement. For example, one repeated response variable (hemoglobin count) might be measured 12 times (implying variables Y1 to Y12 on the left-hand side of the equal sign in the MODEL statement), with the associated within-subject factors treatment and time (implying two factors listed in the REPEATED statement). See the section "Examples" on page 4204 for an example of how PROC GLM handles this case.

Designs with two or more repeated response variables can, however, be handled with the IDENTITY transformation; see the description of this transformation in the following section, and see Example 53.9 for an example of analyzing a doubly multivariate repeated measures design.

When a REPEATED statement appears, the GLM procedure enters a multivariate mode of handling missing values. If any values for variables corresponding to each combination of the within-subject factors are missing, the observation is excluded from the analysis.

If you use a CONTRAST or TEST statement with a REPEATED statement, you must enter the CONTRAST or TEST statement before the REPEATED statement.

The simplest form of the REPEATED statement requires only a *factor-name*. With two repeated factors, you must specify the *factor-name* and number of levels (*levels*) for each factor. Optionally, you can specify the actual values for the levels (*level-values*), a *transformation* that defines single-degree-of-freedom contrasts, and *options* for additional analyses and output. When you specify more than one within-subject factor, the *factor-names* (and associated level and transformation information) must be separated by a comma in the REPEATED statement.

These terms are described in the following section, "Syntax Details."

Syntax Details

Table 53.10 summarizes the *options* available in the REPEATED statement.

Table 53.10 REPEATED Statement Options

Option	Description
CANONICAL	Performs a canonical analysis of the **H** and **E** matrices
HTYPE=	Specifies the type of the **H** matrix used for analysis
MEAN	Generates the overall arithmetic means of the within-subject variables
MSTAT=	Specifies the method of evaluating the test statistics
NOM	Displays only univariate analyses results
NOU	Displays only multivariate analyses results
PRINTE	Displays **E** and partial correlation matrices for multivariate tests, and prints sphericity tests
PRINTH	Displays the **H** matrices for multivariate tests
PRINTM	Displays the transformation matrices that define tested contrasts
PRINTRV	Displays characteristic roots and vectors for multivariate tests
SUMMARY	Produces analysis-of-variance tables for univariate tests
UEPSDEF=	Specifies the F test adjustment for univariate tests

You can specify the following terms in the REPEATED statement.

factor-specification

The *factor-specification* for the REPEATED statement can include any number of individual factor specifications, separated by commas, of the following form:

factor-name levels < **(***level-values***)** > < *transformation* >

where

factor-name names a factor to be associated with the dependent variables. The name should not be the same as any variable name that already exists in the data set being analyzed and should conform to the usual conventions of SAS variable names.

When specifying more than one factor, list the dependent variables in the MODEL statement so that the within-subject factors defined in the REPEATED statement are nested; that is, the first factor defined in the REPEATED statement should be the one with values that change least frequently.

levels gives the number of levels associated with the factor being defined. When there is only one within-subject factor, the number of levels is equal to the number of dependent variables. In this case, *levels* is optional. When more than one within-subject factor is defined, however, *levels* is required, and the product of the number of levels of all the factors must equal the number of dependent variables in the MODEL statement.

(*level-values*) gives values that correspond to levels of a repeated-measures factor. These values are used to label output and as spacings for constructing orthogonal polynomial

contrasts if you specify a POLYNOMIAL transformation. The number of values specified must correspond to the number of levels for that factor in the REPEATED statement. Enclose the *level-values* in parentheses.

The following *transformation* keywords define single-degree-of-freedom contrasts for factors specified in the REPEATED statement. Since the number of contrasts generated is always one less than the number of levels of the factor, you have some control over which contrast is omitted from the analysis by which transformation you select. The only exception is the IDENTITY transformation; this transformation is not composed of contrasts and has the same degrees of freedom as the factor has levels. By default, the procedure uses the CONTRAST transformation.

CONTRAST< (*ordinal-reference-level*) >
generates contrasts between levels of the factor and a reference level. By default, the procedure uses the last level as the reference level; you can optionally specify a reference level in parentheses after the keyword CONTRAST. The reference level corresponds to the ordinal value of the level rather than the level value specified. For example, to generate contrasts between the first level of a factor and the other levels, use

```
contrast(1)
```

HELMERT
generates contrasts between each level of the factor and the mean of subsequent levels.

IDENTITY
generates an identity transformation corresponding to the associated factor. This transformation is *not* composed of contrasts; it has n degrees of freedom for an n-level factor, instead of $n - 1$. This can be used for doubly multivariate repeated measures.

MEAN< (*ordinal-reference-level*) >
generates contrasts between levels of the factor and the mean of all other levels of the factor. Specifying a reference level eliminates the contrast between that level and the mean. Without a reference level, the contrast involving the last level is omitted. See the CONTRAST transformation for an example.

POLYNOMIAL
generates orthogonal polynomial contrasts. Level values, if provided, are used as spacings in the construction of the polynomials; otherwise, equal spacing is assumed.

PROFILE
generates contrasts between adjacent levels of the factor.

You can specify the following *options* in the REPEATED statement after a slash (/).

CANONICAL
performs a canonical analysis of the **H** and **E** matrices corresponding to the transformed variables specified in the REPEATED statement.

HTYPE=*n*

specifies the type of the **H** matrix used in the multivariate tests and the type of sums of squares used in the univariate tests. See the HTYPE= option in the specifications for the MANOVA statement for further details.

MEAN

generates the overall arithmetic means of the within-subject variables.

MSTAT=FAPPROX | EXACT

specifies the method of evaluating the test statistics for the multivariate analysis. The default is MSTAT=FAPPROX, which specifies that the multivariate tests are evaluated using the usual approximations based on the F distribution, as discussed in the section "Multivariate Tests" in Chapter 4, "Introduction to Regression Procedures." Alternatively, you can specify MSTAT=EXACT to compute exact p-values for three of the four tests (Wilks' lambda, the Hotelling-Lawley trace, and Roy's greatest root) and an improved F approximation for the fourth (Pillai's trace). While MSTAT=EXACT provides better control of the significance probability for the tests, especially for Roy's greatest root, computations for the exact p-values can be appreciably more demanding, and are in fact infeasible for large problems (many dependent variables). Thus, although MSTAT=EXACT is more accurate for most data, it is not the default method. For more information about the results of MSTAT=EXACT, see the section "Multivariate Analysis of Variance" on page 4247.

NOM

displays only the results of the univariate analyses.

NOU

displays only the results of the multivariate analyses.

PRINTE

displays the **E** matrix for each combination of within-subject factors, as well as partial correlation matrices for both the original dependent variables and the variables defined by the transformations specified in the REPEATED statement. In addition, the PRINTE option provides sphericity tests for each set of transformed variables. If the requested transformations are not orthogonal, the PRINTE option also provides a sphericity test for a set of orthogonal contrasts.

PRINTH

displays the **H** (SSCP) matrix associated with each multivariate test.

PRINTM

displays the transformation matrices that define the contrasts in the analysis. PROC GLM always displays the **M** matrix so that the transformed variables are defined by the rows, not the columns, of the displayed **M** matrix. In other words, PROC GLM actually displays $\mathbf{M}'$.

PRINTRV

displays the characteristic roots and vectors for each multivariate test.

SUMMARY

produces analysis-of-variance tables for each contrast defined by the within-subject factors. Along with tests for the effects of the independent variables specified in the MODEL statement, a term labeled MEAN tests the hypothesis that the overall mean of the contrast is zero.

UEPSDEF=*unbiased-epsilon-definition*

specifies the type of adjustment for the univariate F test that is displayed in addition to the Greenhouse-Geisser adjustment. The default is UEPSDEF=HFL, corresponding to the corrected form of the Huynh-Feldt adjustment (Huynh and Feldt 1976; Lecoutre 1991). Other alternatives are UEPSDEF=HF, the uncorrected Huynh-Feldt adjustment (the only available method in previous releases of SAS/STAT software), and UEPSDEF=CM, the adjustment of Chi et al. (2012). See the section "Hypothesis Testing in Repeated Measures Analysis" on page 4251 for details about these adjustments.

Examples

When specifying more than one factor, list the dependent variables in the MODEL statement so that the within-subject factors defined in the REPEATED statement are nested; that is, the first factor defined in the REPEATED statement should be the one with values that change least frequently. For example, assume that three treatments are administered at each of four times, for a total of twelve dependent variables on each experimental unit. If the variables are listed in the MODEL statement as Y1 through Y12, then the REPEATED statement in

```
proc glm;
   class group;
   model Y1-Y12=group / nouni;
   repeated trt 3, time 4;
run;
```

implies the following structure:

	Dependent Variables											
	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10	Y11	Y12
Value of trt	1	1	1	1	2	2	2	2	3	3	3	3
Value of time	1	2	3	4	1	2	3	4	1	2	3	4

The REPEATED statement always produces a table like the preceding one. For more information, see the section "Repeated Measures Analysis of Variance" on page 4249.

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

TEST Statement

TEST H=*effects* **E=***effect* < / *options* > **;**

By default, for each sum of squares in the analysis an F value is computed that uses the residual MS as an error term. Use a TEST statement to request additional F tests that use other effects as error terms. You need a TEST statement when a nonstandard error structure (as in a split-plot design) exists. Note, however, that this might not be appropriate if the design is unbalanced, since in most unbalanced designs with nonstandard error structures, mean squares are not necessarily independent with equal expectations under the null hypothesis.

CAUTION: The GLM procedure does not check any of the assumptions underlying the F statistic. When you specify a TEST statement, you assume sole responsibility for the validity of the F statistic produced. To help validate a test, you can use the RANDOM statement and inspect the expected mean squares, or you can use the TEST option of the RANDOM statement.

You can use as many TEST statements as you want, provided that they appear after the MODEL statement.

You can specify the following terms in the TEST statement.

H=*effects*
specifies which effects in the preceding model are to be used as hypothesis (numerator) effects. The H= specification is required.

E=*effect*
specifies one, and only one, effect to use as the error (denominator) term. The E= specification is required.

By default, the sum of squares type for all hypothesis sum of squares and error sum of squares is the highest type computed in the model. If the hypothesis type or error type is to be another type that was computed in the model, you should specify one or both of the following *options* after a slash (/).

ETYPE=*n*
specifies the type of sum of squares to use for the error term. The type must be a type computed in the model (*n*=1, 2, 3, or 4).

HTYPE=*n*
specifies the type of sum of squares to use for the hypothesis. The type must be a type computed in the model (*n*=1, 2, 3, or 4).

This example illustrates the TEST statement with a split-plot model:

```
proc glm;
   class a b c;
   model y=a  b(a) c a*c b*c(a);
   test h=a e=b(a)/ htype=1 etype=1;
   test h=c a*c e=b*c(a) / htype=1 etype=1;
run;
```

WEIGHT Statement

WEIGHT *variable* **;**

When a WEIGHT statement is used, a weighted residual sum of squares

$$\sum_i w_i (y_i - \hat{y}_i)^2$$

is minimized, where w_i is the value of the variable specified in the WEIGHT statement, y_i is the observed value of the response variable, and $\hat{y}_i$ is the predicted value of the response variable.

If you specify the WEIGHT statement, it must appear before the first RUN statement or it is ignored.

An observation is used in the analysis only if the value of the WEIGHT statement variable is nonmissing and greater than zero.

The WEIGHT statement has no effect on degrees of freedom or number of observations, but it is used by the MEANS statement when calculating means and performing multiple comparison tests (as described in the section "MEANS Statement" on page 4186).

The normal equations used when a WEIGHT statement is present are

$$\mathbf{X}'\mathbf{W}\mathbf{X}\boldsymbol{\beta} = \mathbf{X}'\mathbf{W}\mathbf{Y}$$

where $\mathbf{W}$ is a diagonal matrix consisting of the values of the variable specified in the WEIGHT statement.

If the weights for the observations are proportional to the reciprocals of the error variances, then the weighted least squares estimates are best linear unbiased estimators (BLUE).

Details: GLM Procedure

Statistical Assumptions for Using PROC GLM

The basic statistical assumption underlying the least squares approach to general linear modeling is that the observed values of each dependent variable can be written as the sum of two parts: a fixed component $x'\beta$, which is a linear function of the independent coefficients, and a random noise, or error, component ϵ:

$$y = x'\boldsymbol{\beta} + \epsilon$$

The independent coefficients x are constructed from the model effects as described in the section "Parameterization of PROC GLM Models" on page 4210. Further, the errors for different observations are assumed to be uncorrelated with identical variances. Thus, this model can be written

$$E(Y) = \mathbf{X}\beta, \qquad \mathrm{Var}(Y) = \sigma^2 I$$

where Y is the vector of dependent variable values, $\mathbf{X}$ is the matrix of independent coefficients, I is the identity matrix, and σ^2 is the common variance for the errors. For multiple dependent variables, the model is

similar except that the errors for different dependent variables within the same observation are not assumed to be uncorrelated. This yields a multivariate linear model of the form

$$E(Y) \quad = \quad \mathbf{X}B, \qquad \mathrm{Var}(\mathrm{vec}(Y)) \quad = \quad \Sigma \otimes I$$

where Y and B are now matrices, with one column for each dependent variable, $\mathrm{vec}(Y)$ strings Y out by rows, and $\otimes$ indicates the Kronecker matrix product.

Under the assumptions thus far discussed, the least squares approach provides estimates of the linear parameters that are unbiased and have minimum variance among linear estimators. Under the further assumption that the errors have a normal (or Gaussian) distribution, the least squares estimates are the maximum likelihood estimates and their distribution is known. All of the significance levels ("p values") and confidence limits calculated by the GLM procedure require this assumption of normality in order to be exactly valid, although they are good approximations in many other cases.

Specification of Effects

Each term in a model, called an *effect*, is a variable or combination of variables. Effects are specified with a special notation that uses variable names and operators. There are two kinds of variables: *classification* (or *CLASS*) *variables* and *continuous variables*. There are two primary operators: *crossing* and *nesting*. A third operator, the *bar operator*, is used to simplify effect specification.

In an analysis-of-variance model, independent variables must be variables that identify classification levels. In the SAS System, these are called *classification* (or *class*) *variables* and are declared in the CLASS statement. (They can also be called *categorical*, *qualitative*, *discrete*, or *nominal variables*.) Classification variables can be either *numeric* or *character*. The values of a classification variable are called *levels*. For example, the classification variable Sex has the levels "male" and "female."

In a model, an independent variable that is not declared in the CLASS statement is assumed to be continuous. Continuous variables, which must be numeric, are used for response variables and covariates. For example, the heights and weights of subjects are continuous variables.

Types of Effects

There are seven different types of effects used in the GLM procedure. In the following list, assume that A, B, C, D, and E are CLASS variables and that X1, X2, and Y are continuous variables:

- Regressor effects are specified by writing continuous variables by themselves: X1 X2.
- Polynomial effects are specified by joining two or more continuous variables with asterisks: X1*X1 X1*X2.
- Main effects are specified by writing CLASS variables by themselves: A B C.
- Crossed effects (interactions) are specified by joining classification variables with asterisks: A*B B*C A*B*C.
- Nested effects are specified by following a main effect or crossed effect with a classification variable or list of classification variables enclosed in parentheses. The main effect or crossed effect is nested within the effects listed in parentheses: B(A) C(B*A) D*E(C*B*A). In this example, B(A) is read "B nested within A."

- Continuous-by-class effects are written by joining continuous variables and classification variables with asterisks: X1*A.
- Continuous-nesting-class effects consist of continuous variables followed by a classification variable interaction enclosed in parentheses: X1(A) X1*X2(A*B).

One example of the general form of an effect involving several variables is

X1*X2*A*B*C(D*E)

This example contains crossed continuous terms by crossed classification terms nested within more than one classification variable. The continuous list comes first, followed by the crossed list, followed by the nesting list in parentheses. Note that asterisks can appear within the nested list but not immediately before the left parenthesis. For details on how the design matrix and parameters are defined with respect to the effects specified in this section, see the section "Parameterization of PROC GLM Models" on page 4210.

The MODEL statement and several other statements use these effects. Some examples of MODEL statements that use various kinds of effects are shown in the following table; a, b, and c represent classification variables, and y, y1, y2, x, and z represent continuous variables.

Specification	Type of Model
`model y=x;`	Simple regression
`model y=x z;`	Multiple regression
`model y=x x*x;`	Polynomial regression
`model y1 y2=x z;`	Multivariate regression
`model y=a;`	One-way ANOVA
`model y=a b c;`	Main-effects ANOVA
`model y=a b a*b;`	Factorial ANOVA with interaction
`model y=a b(a) c(b a);`	Nested ANOVA
`model y1 y2=a b;`	Multivariate analysis of variance (MANOVA)
`model y=a x;`	Analysis of covariance
`model y=a x(a);`	Separate-slopes regression
`model y=a x x*a;`	Homogeneity-of-slopes regression

The Bar Operator

You can shorten the specification of a large factorial model by using the bar operator. For example, two ways of writing the model for a full three-way factorial model follow:

```
model Y = A B C   A*B A*C B*C   A*B*C;

model Y = A|B|C;
```

When the bar (|) is used, the right and left sides become effects, and the cross of them becomes an effect. Multiple bars are permitted. The expressions are expanded from left to right, using rules 2–4 given in Searle (1971, p. 390).

- Multiple bars are evaluated from left to right. For instance, A|B|C is evaluated as follows:

```
A | B | C   →   { A | B } | C
            →   { A  B  A*B } | C
            →   A  B  A*B  C  A*C  B*C  A*B*C
```

- Crossed and nested groups of variables are combined. For example, A(B) | C(D) generates A*C(B D), among other terms.
- Duplicate variables are removed. For example, A(C) | B(C) generates A*B(C C), among other terms, and the extra C is removed.
- Effects are discarded if a variable occurs on both the crossed and nested parts of an effect. For instance, A(B) | B(D E) generates A*B(B D E), but this effect is eliminated immediately.

You can also specify the maximum number of variables involved in any effect that results from bar evaluation by specifying that maximum number, preceded by an @ sign, at the end of the bar effect. For example, the specification A | B | C@2 would result in only those effects that contain 2 or fewer variables: in this case, A B A*B C A*C and B*C.

More examples of using the bar and at operators follow:

A | C(B)	is equivalent to	A C(B) A*C(B)
A(B) | C(B)	is equivalent to	A(B) C(B) A*C(B)
A(B) | B(D E)	is equivalent to	A(B) B(D E)
A | B(A) | C	is equivalent to	A B(A) C A*C B*C(A)
A | B(A) | C@2	is equivalent to	A B(A) C A*C
A | B | C | D@2	is equivalent to	A B A*B C A*C B*C D A*D B*D C*D
A*B(C*D)	is equivalent to	A*B(C D)

Using PROC GLM Interactively

You can use the GLM procedure interactively. After you specify a model with a MODEL statement and run PROC GLM with a RUN statement, you can execute a variety of statements without reinvoking PROC GLM.

The section "Syntax: GLM Procedure" on page 4156 describes which statements can be used interactively. These interactive statements can be executed singly or in groups by following the single statement or group of statements with a RUN statement. Note that the MODEL statement cannot be repeated; PROC GLM allows only one MODEL statement.

If you use PROC GLM interactively, you can end the GLM procedure with a DATA step, another PROC step, an ENDSAS statement, or a QUIT statement.

When you are using PROC GLM interactively, additional RUN statements do not end the procedure but tell PROC GLM to execute additional statements.

When you specify a WHERE statement with PROC GLM, it should appear before the first RUN statement. The WHERE statement enables you to select only certain observations for analysis without using a subsetting DATA step. For example, **`where group ne 5`** omits observations with GROUP=5 from the analysis. See *SAS DATA Step Statements: Reference* for details on this statement.

When you specify a BY statement with PROC GLM, interactive processing is not possible; that is, once the first RUN statement is encountered, processing proceeds for each BY group in the data set, and no further statements are accepted by the procedure.

Interactivity is also disabled when there are different patterns of missing values among the dependent variables. For details, see the section "Missing Values" on page 4260.

Parameterization of PROC GLM Models

The GLM procedure constructs a linear model according to the specifications in the MODEL statement. Each effect generates one or more columns in a design matrix $\mathbf{X}$. This section shows precisely how $\mathbf{X}$ is built.

Intercept

All models include a column of 1s by default to estimate an intercept parameter μ. You can use the NOINT option to suppress the intercept.

Regression Effects

Regression effects (covariates) have the values of the variables copied into the design matrix directly. Polynomial terms are multiplied out and then installed in $\mathbf{X}$.

Main Effects

If a classification variable has *m* levels, PROC GLM generates *m* columns in the design matrix for its main effect. Each column is an indicator variable for one of the levels of the classification variable. The default order of the columns is the sort order of the values of their levels; this order can be controlled with the ORDER= option in the PROC GLM statement, as shown in the following table.

Data		**Design Matrix**					
			A		B		
A	B	μ	A1	A2	B1	B2	B3
1	1	1	1	0	1	0	0
1	2	1	1	0	0	1	0
1	3	1	1	0	0	0	1
2	1	1	0	1	1	0	0
2	2	1	0	1	0	1	0
2	3	1	0	1	0	0	1

There are more columns for these effects than there are degrees of freedom for them; in other words, PROC GLM is using an over-parameterized model.

Crossed Effects

First, PROC GLM reorders the terms to correspond to the order of the variables in the CLASS statement; thus, B*A becomes A*B if A precedes B in the CLASS statement. Then, PROC GLM generates columns for all combinations of levels that occur in the data. The order of the columns is such that the rightmost variables in the cross index faster than the leftmost variables. No columns are generated corresponding to combinations of levels that do not occur in the data.

Data		**Design Matrix**											
			A		B			A*B					
A	B	μ	A1	A2	B1	B2	B3	A1B1	A1B2	A1B3	A2B1	A2B2	A2B3
1	1	1	1	0	1	0	0	1	0	0	0	0	0
1	2	1	1	0	0	1	0	0	1	0	0	0	0
1	3	1	1	0	0	0	1	0	0	1	0	0	0
2	1	1	0	1	1	0	0	0	0	0	1	0	0
2	2	1	0	1	0	1	0	0	0	0	0	1	0
2	3	1	0	1	0	0	1	0	0	0	0	0	1

In this matrix, main-effects columns are not linearly independent of crossed-effect columns; in fact, the column space for the crossed effects contains the space of the main effect.

Nested Effects

Nested effects are generated in the same manner as crossed effects. Hence, the design columns generated by the following statements are the same (but the ordering of the columns is different):

```
model y=a b(a);
```
(B nested within A)

```
model y=a a*b;
```
(omitted main effect for B)

The nesting operator in PROC GLM is more a notational convenience than an operation distinct from crossing. Nested effects are characterized by the property that the nested variables never appear as main effects. The order of the variables within nesting parentheses is made to correspond to the order of these variables in the CLASS statement. The order of the columns is such that variables outside the parentheses index faster than those inside the parentheses, and the rightmost nested variables index faster than the leftmost variables.

Data		**Design Matrix**								
			A		B(A)					
A	B	μ	A1	A2	B1A1	B2A1	B3A1	B1A2	B2A2	B3A2
1	1	1	1	0	1	0	0	0	0	0
1	2	1	1	0	0	1	0	0	0	0
1	3	1	1	0	0	0	1	0	0	0
2	1	1	0	1	0	0	0	1	0	0
2	2	1	0	1	0	0	0	0	1	0
2	3	1	0	1	0	0	0	0	0	1

Continuous-Nesting-Class Effects

When a continuous variable nests with a classification variable, the design columns are constructed by multiplying the continuous values into the design columns for the class effect.

Data		**Design Matrix**				
			A		X(A)	
X	A	μ	A1	A2	X(A1)	X(A2)
21	1	1	1	0	21	0
24	1	1	1	0	24	0
22	1	1	1	0	22	0
28	2	1	0	1	0	28
19	2	1	0	1	0	19
23	2	1	0	1	0	23

This model estimates a separate slope for X within each level of A.

Continuous-by-Class Effects

Continuous-by-class effects generate the same design columns as continuous-nesting-class effects. The two models differ by the presence of the continuous variable as a regressor by itself, in addition to being a contributor to X*A.

Data		**Design Matrix**					
				A		X*A	
X	A	μ	X	A1	A2	X*A1	X*A2
21	1	1	21	1	0	21	0
24	1	1	24	1	0	24	0
22	1	1	22	1	0	22	0
28	2	1	28	0	1	0	28
19	2	1	19	0	1	0	19
23	2	1	23	0	1	0	23

Continuous-by-class effects are used to test the homogeneity of slopes. If the continuous-by-class effect is nonsignificant, the effect can be removed so that the response with respect to X is the same for all levels of the classification variables.

General Effects

An example that combines all the effects is

```
X1*X2*A*B*C(D E)
```

The continuous list comes first, followed by the crossed list, followed by the nested list in parentheses.

The sequencing of parameters is important to learn if you use the CONTRAST or ESTIMATE statement to compute or test some linear function of the parameter estimates.

Effects might be retitled by PROC GLM to correspond to ordering rules. For example, B*A(E D) might be retitled A*B(D E) to satisfy the following:

- Classification variables that occur outside parentheses (crossed effects) are sorted in the order in which they appear in the CLASS statement.
- Variables within parentheses (nested effects) are sorted in the order in which they appear in a CLASS statement.

The sequencing of the parameters generated by an effect can be described by which variables have their levels indexed faster:

- Variables in the crossed part index faster than variables in the nested list.
- Within a crossed or nested list, variables to the right index faster than variables to the left.

For example, suppose a model includes four effects—A, B, C, and D—each having two levels, 1 and 2. If the CLASS statement is

```
class A B C D;
```

then the order of the parameters for the effect B*A(C D), which is retitled A*B(C D), is as follows.

$$
\begin{array}{c}
A_1B_1C_1D_1 \\
A_1B_2C_1D_1 \\
A_2B_1C_1D_1 \\
A_2B_2C_1D_1 \\
A_1B_1C_1D_2 \\
A_1B_2C_1D_2 \\
A_2B_1C_1D_2 \\
A_2B_2C_1D_2 \\
A_1B_1C_2D_1 \\
A_1B_2C_2D_1 \\
A_2B_1C_2D_1 \\
A_2B_2C_2D_1 \\
A_1B_1C_2D_2 \\
A_1B_2C_2D_2 \\
A_2B_1C_2D_2 \\
A_2B_2C_2D_2
\end{array}
$$

Note that first the crossed effects B and A are sorted in the order in which they appear in the CLASS statement so that A precedes B in the parameter list. Then, for each combination of the nested effects in turn, combinations of A and B appear. The B effect changes fastest because it is rightmost in the (renamed) cross list. Then A changes next fastest. The D effect changes next fastest, and C is the slowest since it is leftmost in the nested list.

When numeric classification variables are used, their levels are sorted by their character format, which might not correspond to their numeric sort sequence. Therefore, it is advisable to include a format for numeric classification variables or to use the ORDER=INTERNAL option in the PROC GLM statement to ensure that levels are sorted by their internal values.

Degrees of Freedom

For models with classification (categorical) effects, there are more design columns constructed than there are degrees of freedom for the effect. Thus, there are linear dependencies among the columns. In this event, the parameters are not jointly estimable; there is an infinite number of least squares solutions. The GLM procedure uses a generalized g_2-inverse to obtain values for the estimates; see the section "Computational Method" on page 4263 for more details. The solution values are not produced unless the SOLUTION option is specified in the MODEL statement. The solution has the characteristic that estimates are zero whenever the design column for that parameter is a linear combination of previous columns. (Strictly termed, the solution values should not be called estimates, since the parameters might not be formally estimable.) With this full parameterization, hypothesis tests are constructed to test linear functions of the parameters that are estimable.

Other procedures (such as the CATMOD procedure) reparameterize models to full rank by using certain restrictions on the parameters. PROC GLM does not reparameterize, making the hypotheses that are commonly tested more understandable. See Goodnight (1978a) for additional reasons for not reparameterizing.

PROC GLM does not actually construct the entire design matrix $\mathbf{X}$; rather, a row x_i of $\mathbf{X}$ is constructed for each observation in the data set and used to accumulate the crossproduct matrix $\mathbf{X}'\mathbf{X} = \sum_i x_i' x_i$.

Hypothesis Testing in PROC GLM

See Chapter 16, "The Four Types of Estimable Functions," for a complete discussion of the four standard types of hypothesis tests.

Example

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glmest.sas.)

To illustrate the four types of tests and the principles upon which they are based, consider a two-way design with interaction based on the following data:

		B 1	B 2
A	1	23.5 23.7	28.7
	2	8.9	5.6 8.9
	3	10.3 12.5	13.6 14.6

Invoke PROC GLM and specify all the estimable functions options to examine what the GLM procedure can test. The following statements produce the summary ANOVA table displayed in Figure 53.10.

```
data example;
   input a b y @@;
   datalines;
1 1 23.5  1 1 23.7  1 2 28.7  2 1  8.9  2 2  5.6
2 2  8.9  3 1 10.3  3 1 12.5  3 2 13.6  3 2 14.6
;

proc glm;
   class a b;
   model y=a b a*b / e e1 e2 e3 e4;
run;
```

Figure 53.10 Summary ANOVA Table from PROC GLM

The GLM Procedure

Dependent Variable: y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	520.4760000	104.0952000	49.66	0.0011
Error	4	8.3850000	2.0962500		
Corrected Total	9	528.8610000			

R-Square	Coeff Var	Root MSE	y Mean
0.984145	9.633022	1.447843	15.03000

The following sections show the general form of estimable functions and discuss the four standard tests, their properties, and abbreviated output for the two-way crossed example.

Estimability

Figure 53.11 is the general form of estimable functions for the example. In order to be testable, a hypothesis must be able to fit within the framework displayed here.

Figure 53.11 General Form of Estimable Functions

The GLM Procedure

General Form of Estimable Functions		
Effect		**Coefficients**
Intercept		L1
a	**1**	L2
a	**2**	L3
a	**3**	L1-L2-L3
b	**1**	L5
b	**2**	L1-L5
a*b	**1 1**	L7
a*b	**1 2**	L2-L7
a*b	**2 1**	L9
a*b	**2 2**	L3-L9
a*b	**3 1**	L5-L7-L9
a*b	**3 2**	L1-L2-L3-L5+L7+L9

If a hypothesis is estimable, the **L**s in the preceding scheme can be set to values that match the hypothesis. All the standard tests in PROC GLM can be shown in the preceding format, with some of the **L**s zeroed and some set to functions of other **L**s.

The following sections show how many of the hypotheses can be tested by comparing the model sum-of-squares regression from one model to a submodel. The notation used is

$$\mathrm{SS}(B\ \mathit{effects}|A\ \mathit{effects}) = \mathrm{SS}(B\ \mathit{effects}, A\ \mathit{effects}) - \mathrm{SS}(A\ \mathit{effects})$$

where SS(A *effects*) denotes the regression model sum of squares for the model consisting of A *effects*. This notation is equivalent to the reduction notation defined by Searle (1971) and summarized in Chapter 16, "The Four Types of Estimable Functions."

Type I Tests

Type I sums of squares (SS), also called *sequential sums of squares*, are the incremental improvement in error sums of squares as each effect is added to the model. They can be computed by fitting the model in steps and recording the difference in error sum of squares at each step.

Source	Type I SS
A	$\mathrm{SS}(A \mid \mu)$
B	$\mathrm{SS}(B \mid \mu, A)$
$A * B$	$\mathrm{SS}(A * B \mid \mu, A, B)$

Type I sums of squares are displayed by default because they are easy to obtain and can be used in various hand calculations to produce sum of squares values for a series of different models. Nelder (1994) and others have argued that Type I and II sums are essentially the only appropriate ones for testing ANOVA effects; however, see also the discussion of Nelder's article, especially Rodriguez, Tobias, and Wolfinger (1995) and Searle (1995).

The Type I hypotheses have these properties:

- Type I sum of squares for all effects add up to the model sum of squares. None of the other sum of squares types have this property, except in special cases.
- Type I hypotheses can be derived from rows of the Forward-Dolittle transformation of $\mathbf{X}'\mathbf{X}$ (a transformation that reduces $\mathbf{X}'\mathbf{X}$ to an upper triangular matrix by row operations).
- Type I sum of squares are statistically independent of each other under the usual assumption that the true residual errors are independent and identically normally distributed (see page 4206).
- Type I hypotheses depend on the order in which effects are specified in the MODEL statement.
- Type I hypotheses are uncontaminated by parameters corresponding to effects that precede the effect being tested; however, the hypotheses usually involve parameters for effects following the tested effect in the model. For example, in the model

  ```
  Y=A B;
  ```

  the Type I hypothesis for B does not involve A parameters, but the Type I hypothesis for A does involve B parameters.
- Type I hypotheses are functions of the cell counts for unbalanced data; the hypotheses are not usually the same hypotheses that are tested if the data are balanced.
- Type I sums of squares are useful for polynomial models where you want to know the contribution of a term as though it had been made orthogonal to preceding effects. Thus, in polynomial models, Type I sums of squares correspond to tests of the orthogonal polynomial effects.

The Type I estimable functions and associated tests for the example are shown in Figure 53.12.

Figure 53.12 Type I Estimable Functions and Tests

Type I Estimable Functions				
		Coefficients		
Effect		**a**	**b**	**a*b**
Intercept		0	0	0
a	**1**	L2	0	0
a	**2**	L3	0	0
a	**3**	-L2-L3	0	0
b	**1**	0.1667*L2-0.1667*L3	L5	0
b	**2**	-0.1667*L2+0.1667*L3	-L5	0
a*b	**1 1**	0.6667*L2	0.2857*L5	L7
a*b	**1 2**	0.3333*L2	-0.2857*L5	-L7
a*b	**2 1**	0.3333*L3	0.2857*L5	L9
a*b	**2 2**	0.6667*L3	-0.2857*L5	-L9
a*b	**3 1**	-0.5*L2-0.5*L3	0.4286*L5	-L7-L9
a*b	**3 2**	-0.5*L2-0.5*L3	-0.4286*L5	L7+L9

Source	DF	Type I SS	Mean Square	F Value	Pr > F
a	2	494.0310000	247.0155000	117.84	0.0003
b	1	10.7142857	10.7142857	5.11	0.0866
a*b	2	15.7307143	7.8653571	3.75	0.1209

Type II Tests

The Type II tests can also be calculated by comparing the error sums of squares (SS) for subset models. The Type II SS are the reduction in error SS due to adding the term after all other terms have been added to the model except terms that contain the effect being tested. An effect is contained in another effect if it can be derived by deleting variables from the latter effect. For example, A and B are both contained in A*B. For this model, the Type II SS are given by the reduced sums of squares as shown in the following table.

Source	Type II SS
A	$\mathrm{SS}(A \mid \mu, B)$
B	$\mathrm{SS}(B \mid \mu, A)$
$A * B$	$\mathrm{SS}(A * B \mid \mu, A, B)$

Type II SS have these properties:

- Type II SS do not necessarily sum to the model SS.
- The hypothesis for an effect does not involve parameters of other effects except for containing effects (which it must involve to be estimable).
- Type II SS are invariant to the ordering of effects in the model.
- For unbalanced designs, Type II hypotheses for effects that are contained in other effects are not usually the same hypotheses that are tested if the data are balanced. The hypotheses are generally functions of the cell counts.

The Type II estimable functions and associated tests for the example are shown in Figure 53.13.

Figure 53.13 Type II Estimable Functions and Tests

Type II Estimable Functions				
		Coefficients		
Effect		**a**	**b**	**a*b**
Intercept		0	0	0
a	**1**	L2	0	0
a	**2**	L3	0	0
a	**3**	-L2-L3	0	0
b	**1**	0	L5	0
b	**2**	0	-L5	0
a*b	**1 1**	0.619*L2+0.0476*L3	0.2857*L5	L7
a*b	**1 2**	0.381*L2-0.0476*L3	-0.2857*L5	-L7
a*b	**2 1**	-0.0476*L2+0.381*L3	0.2857*L5	L9
a*b	**2 2**	0.0476*L2+0.619*L3	-0.2857*L5	-L9
a*b	**3 1**	-0.5714*L2-0.4286*L3	0.4286*L5	-L7-L9
a*b	**3 2**	-0.4286*L2-0.5714*L3	-0.4286*L5	L7+L9

Source	DF	Type II SS	Mean Square	F Value	Pr > F
a	2	499.1202857	249.5601429	119.05	0.0003
b	1	10.7142857	10.7142857	5.11	0.0866
a*b	2	15.7307143	7.8653571	3.75	0.1209

Type III and Type IV Tests

Type III and Type IV sums of squares (SS), sometimes referred to as *partial sums of squares*, are considered by many to be the most desirable; see Searle (1987, Section 4.6). Using PROC GLM's singular parameterization, these SS cannot, in general, be computed by comparing model SS from different models. However, they can sometimes be computed by reduction for methods that reparameterize to full rank, when such a reparameterization effectively imposes Type III linear constraints on the parameters. In PROC GLM, they are computed by constructing a hypothesis matrix $\mathbf{L}$ and then computing the SS associated with the hypothesis $\mathbf{L}\boldsymbol{\beta} = 0$. As long as there are no missing cells in the design, Type III and Type IV SS are the same.

These are properties of Type III and Type IV SS:

- The hypothesis for an effect does not involve parameters of other effects except for containing effects (which it must involve to be estimable).
- The hypotheses to be tested are invariant to the ordering of effects in the model.
- The hypotheses are the same hypotheses that are tested if there are no missing cells. They are not functions of cell counts.
- The SS do not generally add up to the model SS and, in some cases, can exceed the model SS.

The SS are constructed from the general form of estimable functions. Type III and Type IV tests are different only if the design has missing cells. In this case, the Type III tests have an orthogonality property, while the

Type IV tests have a balancing property. These properties are discussed in Chapter 16, "The Four Types of Estimable Functions." For this example, since the data contain observations for all pairs of levels of A and B, Type IV tests are identical to the Type III tests that are shown in Figure 53.14. (This combines tables from several pages of output.)

Figure 53.14 Type III Estimable Functions and Tests

Type III Estimable Functions				
		Coefficients		
Effect		**a**	**b**	**a*b**
Intercept		0	0	0
a	**1**	L2	0	0
a	**2**	L3	0	0
a	**3**	-L2-L3	0	0
b	**1**	0	L5	0
b	**2**	0	-L5	0
a*b	**1 1**	0.5*L2	0.3333*L5	L7
a*b	**1 2**	0.5*L2	-0.3333*L5	-L7
a*b	**2 1**	0.5*L3	0.3333*L5	L9
a*b	**2 2**	0.5*L3	-0.3333*L5	-L9
a*b	**3 1**	-0.5*L2-0.5*L3	0.3333*L5	-L7-L9
a*b	**3 2**	-0.5*L2-0.5*L3	-0.3333*L5	L7+L9

Source	DF	Type III SS	Mean Square	F Value	Pr > F
a	2	479.1078571	239.5539286	114.28	0.0003
b	1	9.4556250	9.4556250	4.51	0.1009
a*b	2	15.7307143	7.8653571	3.75	0.1209

Effect Size Measures for F Tests in GLM

A significant F test in a linear model indicates that the effect of the term or contrast being tested might be real. The next thing you want to know is, How big is the effect? Various measures have been devised to give answers to this question that are comparable over different experimental designs. If you specify the EFFECTSIZE option in the MODEL statement, then GLM adds to each ANOVA table estimates and confidence intervals for three different measures of effect size:

- the noncentrality parameter for the F test
- the proportion of total variation accounted for (also known as the semipartial correlation ratio or the squared semipartial correlation)
- the proportion of partial variation accounted for (also known as the full partial correlation ratio or the squared full partial correlation)

The adjectives "semipartial" and "full partial" might seem strange. They refer to how other effects are "partialed out" of the dependent variable and the effect being tested. For "semipartial" statistics, all other effects are partialed out of the effect in question, but not the dependent variable. This measures the (adjusted)

effect as a proportion of the total variation in the dependent variable. On the other hand, for "full partial" statistics, all other effects are partialed out of *both* the dependent variable and the effect in question. This measures the (adjusted) effect as a proportion of only the dependent variation remaining after partialing, or in other words the partial variation. Details about the computation and interpretation of these estimates and confidence intervals are discussed in the remainder of this section.

The noncentrality parameter is directly related to the true distribution of the F statistic when the effect being tested has a non-null effect. The uniformly minimum variance unbiased estimate for the noncentrality is

$$\mathrm{NC}_{\mathrm{UMVUE}} = \frac{\mathrm{DF}(\mathrm{DFE}-2)\mathrm{FValue}}{\mathrm{DFE}} - \mathrm{DF}$$

where FValue is the observed value of the F statistic for the test and DF and DFE are the numerator and denominator degrees of freedom for the test, respectively. An alternative estimate that can be slightly biased but has a somewhat lower expected mean square error is

$$\mathrm{NC}_{\mathrm{minMSE}} = \frac{\mathrm{DF}(\mathrm{DFE}-4)\mathrm{FValue}}{\mathrm{DFE}} - \frac{\mathrm{DF}(\mathrm{DFE}-4)}{\mathrm{DFE}-2}$$

(See Perlman and Rasmussen (1975), cited in Johnson, Kotz, and Balakrishnan (1994).) A $p \times 100\%$ lower confidence bound for the noncentrality is given by the value of NC for which probf(FValue,DF,DFE,NC) = p, where probf() is the cumulative probability function for the non-central F distribution. This result can be used to form a $(1-\alpha) \times 100\%$ confidence interval for the noncentrality.

The partial proportion of variation accounted for by the effect being tested is easiest to define by its natural sample estimate,

$$\hat{\eta}^2_{partial} = \frac{\mathrm{SS}}{\mathrm{SS}+\mathrm{SSE}}$$

where SSE is the sample error sum of squares. Note that $\hat{\eta}^2_{partial}$ is actually sometimes denoted $R^2_{partial}$ or just R square, but in this context the R square notation is reserved for the $\hat{\eta}^2$ corresponding to the overall model, which is just the familiar R square for the model. $\hat{\eta}^2_{partial}$ is actually a biased estimate of the true $\eta^2_{partial}$; an alternative that is approximately unbiased is given by

$$\omega^2_{partial} = \frac{\mathrm{SS}-\mathrm{DF}\times\mathrm{MSE}}{\mathrm{SS}+(\mathrm{N}-\mathrm{DF})\mathrm{MSE}}$$

where MSE = SSE/DFE is the sample mean square for error and N is the number of observations. The true $\eta^2_{partial}$ is related to the true noncentrality parameter NC by the formula

$$\eta^2_{partial} = \frac{\mathrm{NC}}{\mathrm{NC}+\mathrm{N}}$$

This fact can be employed to transform a confidence interval for NC into one for $\eta^2_{partial}$. Note that some authors (Steiger and Fouladi 1997; Fidler and Thompson 2001; Smithson 2003) have published slightly different confidence intervals for $\eta^2_{partial}$, based on a slightly different formula for the relationship between $\eta^2_{partial}$ and NC, apparently due to Cohen (1988). Cohen's formula appears to be approximately correct for

random predictor values (Maxwell 2000), but the one given previously is correct if the predictor values are assumed fixed, as is standard for the GLM procedure.

Finally, the proportion of total variation accounted for by the effect being tested is again easiest to define by its natural sample estimate, which is known as the *(semipartial)* $\hat{\eta}^2$ statistic,

$$\hat{\eta}^2 = \frac{\mathrm{SS}}{\mathrm{SS}_{total}}$$

where SS_{total} is the total sample (corrected) sum of squares, and SS is the observed sum of squares due to the effect being tested. As with $\hat{\eta}^2_{partial}$, $\hat{\eta}^2$ is actually a biased estimate of the true η^2; an alternative that is approximately unbiased is the *(semipartial)* ω^2 statistic

$$\omega^2 = \frac{\mathrm{SS} - \mathrm{DF} \times \mathrm{MSE}}{\mathrm{SS}_{total} + \mathrm{MSE}}$$

where MSE = SSE/DFE is the sample mean square for error. Whereas $\eta^2_{partial}$ depends only on the noncentrality for its associated F test, the presence of the total sum of squares in the previous formulas indicates that η^2 depends on the noncentralities for all effects in the model. An exact confidence interval is not available, but if you write the formula for $\hat{\eta}^2$ as

$$\hat{\eta}^2 = \frac{\mathrm{SS}}{\mathrm{SS} + (\mathrm{SS}_{total} - \mathrm{SS})}$$

then a conservative confidence interval can be constructed as for $\eta^2_{partial}$, treating $\mathrm{SS}_{total} - \mathrm{SS}$ as the SSE and N – DF – 1 as the DFE (Smithson 2004). This confidence interval is conservative in the sense that it implies values of the true η^2 that are smaller than they should be.

Estimates and confidence intervals for effect sizes require some care in interpretation. For example, while the true proportions of total and partial variation accounted for are nonnegative quantities, their estimates might be less than zero. Also, confidence intervals for effect sizes are not directly related to the corresponding estimates. In particular, it is possible for the estimate to lie outside the confidence interval.

As for interpreting the actual values of effect size measures, the approximately unbiased ω^2 estimates are usually preferred for point estimates. Some authors have proposed certain ranges as indicating "small," "medium," and "large" effects (Cohen 1988), but general benchmarks like this depend on the nature of the data and the typical signal-to-noise ratio; they should not be expected to apply across various disciplines. For example, while an ω^2 value of 10% might be viewed as "large" for psychometric data, it can be a relatively small effect for industrial experimentation. Whatever the standard, confidence intervals for true effect sizes typically span more than one category, indicating that in small experiments, it can be difficult to make firm statements about the size of effects.

Example

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glmde2.sas`.)

The data for this example are similar to data analyzed in Steiger and Fouladi (1997); Fidler and Thompson (2001); Smithson (2003). Consider the following hypothetical design, testing 28 men and 28 women on seven different tasks.

```
data Test;
   do Task = 1 to 7;
      do Gender = 'M','F';
         do i = 1 to 4;
            input Response @@;
            output;
         end;
      end;
   end;
   datalines;
7.1 2.8 3.9 3.7     6.5 6.5 6.5 6.6
7.1 5.5 4.8 2.6     3.6 5.4 5.6 4.5
7.2 4.6 4.9 4.6     3.3 5.4 2.8 1.5
5.6 6.2 5.4 6.5     5.6 2.7 3.8 2.3
2.2 5.4 5.6 8.4     1.2 2.0 4.3 4.6
9.1 4.5 7.6 4.9     4.3 7.7 6.5 7.7
4.5 3.8 5.9 6.1     1.7 2.5 4.3 2.7
;
```

This is a balanced two-way design with four replicates per cell. The following statements analyze this data. Since this is a balanced design, you can use the SS1 option in the MODEL statement to display only the Type I sums of squares.

```
proc glm data=Test;
   class Gender Task;
   model Response = Gender|Task / ss1;
run;
```

The analysis of variance results are shown in Figure 53.15.

Figure 53.15 Two-Way Analysis of Variance

The GLM Procedure

Dependent Variable: Response

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Gender	1	14.40285714	14.40285714	6.00	0.0185
Task	6	38.15964286	6.35994048	2.65	0.0285
Gender*Task	6	35.99964286	5.99994048	2.50	0.0369

You can see that the two main effects as well as their interaction are all significant. Suppose you want to compare the main effect of Gender with the interaction between Gender and Task. The sums of squares for the interaction are more than twice as large, but it's not clear how experimental variability might affect this. The following statements perform the same analysis as before, but add the EFFECTSIZE option to the MODEL statement; also, with ALPHA=0.1 option displays 90% confidence intervals, ensuring that inferences based on the *p*-values at the 0.05 levels will agree with the lower confidence limit.

```
proc glm data=Test;
   class Gender Task;
   model Response = Gender|Task / ss1 effectsize alpha=0.1;
run;
```

The Type I analysis of variance results with added effect size information are shown in Figure 53.16.

Figure 53.16 Two-Way Analysis of Variance with Effect Sizes

The GLM Procedure

Dependent Variable: Response

						Noncentrality Parameter			
Source	DF	Type I SS	Mean Square	F Value	Pr > F	Min Var Unbiased Estimate	Low MSE Estimate	90% Confidence Limits	
Gender	1	14.40285714	14.40285714	6.00	0.0185	4.72	4.48	0.521	17.1
Task	6	38.15964286	6.35994048	2.65	0.0285	9.14	8.69	0.870	27.3
Gender*Task	6	35.99964286	5.99994048	2.50	0.0369	8.29	7.87	0.463	25.9

	Total Variation Accounted For				Partial Variation Accounted For			
Source	Semipartial Eta-Square	Semipartial Omega-Square	Conservative 90% Confidence Limits		Partial Eta-Square	Partial Omega-Square	90% Confidence Limits	
Gender	0.0761	0.0626	0.0019	0.2030	0.1250	0.0820	0.0092	0.2342
Task	0.2015	0.1239	0.0000	0.2772	0.2746	0.1502	0.0153	0.3277
Gender*Task	0.1901	0.1126	0.0000	0.2639	0.2632	0.1385	0.0082	0.3160

The estimated effect sizes for Gender and the interaction all tell pretty much the same story: the effect of the interaction is appreciably greater than the effect of Gender. However, the confidence intervals suggest that this inference should be treated with some caution, since the lower confidence bound for the Gender effect is greater than the lower confidence bound for the interaction in all three cases. Follow-up testing is probably in order, using the estimated effect sizes in this preliminary study to design a large enough sample to distinguish the sizes of the effects.

Absorption

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glmabsrb.sas.)

Absorption is a computational technique used to reduce computing resource needs in certain cases. The classic use of absorption occurs when a blocking factor with a large number of levels is a term in the model.

For example, the statements

```
proc glm;
   absorb herd;
   class a b;
   model y=a b a*b;
run;
```

are equivalent to

```
proc glm;
   class herd a b;
   model y=herd a b a*b;
run;
```

The exception to the previous statements is that the Type II, Type III, or Type IV SS for HERD are not computed when HERD is absorbed.

The algorithm for absorbing variables is similar to the one used by the NESTED procedure for computing a nested analysis of variance. As each new row of $[X|Y]$ (corresponding to the nonabsorbed independent effects and the dependent variables) is constructed, it is adjusted for the absorbed effects in a Type I fashion. The efficiency of the absorption technique is due to the fact that this adjustment can be done in one pass of the data and without solving any linear equations, assuming that the data have been sorted by the absorbed variables.

Several effects can be absorbed at one time. For example, these statements

```
proc glm;
   absorb herd cow;
   class a b;
   model y=a b a*b;
run;
```

are equivalent to

```
proc glm;
   class herd cow a b;
   model y=herd cow(herd) a b a*b;
run;
```

When you use absorption, the size of the $\mathbf{X}'\mathbf{X}$ matrix is a function only of the effects in the MODEL statement. The effects being absorbed do not contribute to the size of the $\mathbf{X}'\mathbf{X}$ matrix.

For the preceding example, a and b can be absorbed:

```
proc glm;
   absorb a b;
   class herd cow;
   model y=herd cow(herd);
run;
```

Although the sources of variation in the results are listed as

```
a b(a) herd cow(herd)
```

all types of estimable functions for herd and cow(herd) are free of a, b, and a*b parameters.

To illustrate the savings in computing by using the ABSORB statement, PROC GLM is run on generated data with 1147 degrees of freedom in the model with the following statements.

```
data a;
   do herd=1 to 40;
      do cow=1 to 30;
         do treatment=1 to 3;
            do rep=1 to 2;
               y = herd/5 + cow/10 + treatment + rannor(1);
```

```
                output;
             end;
          end;
       end;
    end;
run;

proc glm data=a;
   class herd cow treatment;
   model y=herd cow(herd) treatment;
run;
```

This analysis would have required over 6 megabytes of memory for the $\mathbf{X}'\mathbf{X}$ matrix had PROC GLM solved it directly. However, in the following statements, the GLM procedure needs only a 4×4 matrix for the intercept and treatment because the other effects are absorbed.

```
proc glm data=a;
   absorb herd cow;
   class treatment;
   model y = treatment;
run;
```

These statements produce the results shown in Figure 53.17.

Figure 53.17 Absorption of Effects

The GLM Procedure

Class Level Information		
Class	Levels	Values
treatment	3	1 2 3

Number of Observations Read	7200
Number of Observations Used	7200

The GLM Procedure

Dependent Variable: y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1201	49465.40242	41.18685	41.57	<.0001
Error	5998	5942.23647	0.99070		
Corrected Total	7199	55407.63889			

R-Square	Coeff Var	Root MSE	y Mean
0.892754	13.04236	0.995341	7.631598

Source	DF	Type I SS	Mean Square	F Value	Pr > F
herd	39	38549.18655	988.44068	997.72	<.0001
cow(herd)	1160	6320.18141	5.44843	5.50	<.0001
treatment	2	4596.03446	2298.01723	2319.58	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
treatment	2	4596.034455	2298.017228	2319.58	<.0001

Specification of ESTIMATE Expressions

Consider the model

$$E(Y) \quad = \quad \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

The corresponding MODEL statement for PROC GLM is

```
model y=x1 x2 x3;
```

To estimate the difference between the parameters for x_1 and x_2,

$$\beta_1 - \beta_2 \quad = \quad \begin{pmatrix} 0 & 1 & -1 & 0 \end{pmatrix} \boldsymbol{\beta}, \text{ where } \boldsymbol{\beta} = \begin{pmatrix} \beta_0 & \beta_1 & \beta_2 & \beta_3 \end{pmatrix}'$$

you can use the following ESTIMATE statement:

```
estimate 'B1-B2'  x1 1  x2 -1;
```

To predict y at $x_1 = 1$, $x_2 = 0$, and $x_3 = -2$, you can estimate

$$\beta_0 + \beta_1 - 2\beta_3 \quad = \quad \begin{pmatrix} 1 & 1 & 0 & -2 \end{pmatrix} \boldsymbol{\beta}$$

with the following ESTIMATE statement:

```
estimate 'B0+B1-2B3' intercept 1 x1 1 x3 -2;
```

Now consider models involving classification variables such as

```
model y=A B A*B;
```

with the associated parameters:

$$\begin{pmatrix} \mu & \alpha_1 & \alpha_2 & \alpha_3 & \beta_1 & \beta_2 & \gamma_{11} & \gamma_{12} & \gamma_{21} & \gamma_{22} & \gamma_{31} & \gamma_{32} \end{pmatrix}$$

The LS-mean for the first level of A is $\mathbf{L}\boldsymbol{\beta}$, where

$$\mathbf{L} = \begin{pmatrix} 1 & | & 1 & 0 & 0 & | & 0.5 & 0.5 & | & 0.5 & 0.5 & 0 & 0 & 0 & 0 \end{pmatrix}$$

You can estimate this with the following ESTIMATE statement:

```
estimate 'LS-mean(A1)' intercept 1 A 1 B 0.5 0.5 A*B 0.5 0.5;
```

Note in this statement that only one element of $\mathbf{L}$ is specified following the A effect, even though A has three levels. Whenever the list of constants following an effect name is shorter than the effect's number of levels, zeros are used as the remaining constants. (If the list of constants is longer than the number of levels for the effect, the extra constants are ignored, and a warning message is displayed.)

To estimate the A linear effect in the preceding model, assuming equally spaced levels for A, you can use the following $\mathbf{L}$:

$$\mathbf{L} = \begin{pmatrix} 0 & | & -1 & 0 & 1 & | & 0 & 0 & | & -0.5 & -0.5 & 0 & 0 & 0.5 & 0.5 \end{pmatrix}$$

The ESTIMATE statement for this $\mathbf{L}$ is written as

```
estimate 'A Linear' A -1 0 1;
```

If you do not specify the elements of $\mathbf{L}$ for an effect that contains a specified effect, then the elements of the specified effect are equally distributed over the corresponding levels of the higher-order effect. In addition, if you specify the intercept in an ESTIMATE or CONTRAST statement, it is distributed over all classification effects that are not contained by any other specified effect.

The distribution of lower-order coefficients to higher-order effect coefficients follows the same general rules as in the LSMEANS statement, and it is similar to that used to construct Type IV tests. In the previous example, the –1 associated with α_1 is divided by the number n_{1j} of γ_{1j} parameters; then each γ_{1j} coefficient is set to $-1/n_{1j}$. The 1 associated with α_3 is distributed among the γ_{3j} parameters in a similar fashion. In the event that an unspecified effect contains several specified effects, only that specified effect with the most factors in common with the unspecified effect is used for distribution of coefficients to the higher-order effect.

Numerous syntactical expressions for the ESTIMATE statement were considered, including many that involved specifying the effect and level information associated with each coefficient. For models involving higher-level effects, the requirement of specifying level information can lead to very bulky specifications. Consequently, the simpler form of the ESTIMATE statement described earlier was implemented.

The syntax of this ESTIMATE statement puts a burden on you to know a priori the order of the parameter list associated with each effect. You can use the ORDER= option in the PROC GLM statement to ensure that the levels of the classification effects are sorted appropriately.

NOTE: If you use the ESTIMATE statement with unspecified effects, use the E option to make sure that the actual $\mathbf{L}$ constructed by the preceding rules is the one you intended.

A Check for Estimability

Each $\mathbf{L}$ is checked for estimability using the relationship $\mathbf{L} = \mathbf{LH}$, where $\mathbf{H} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$. The $\mathbf{L}$ vector is declared nonestimable, if for any i

$$\mathrm{ABS}(\mathbf{L}_i - (\mathbf{LH})_i) > \begin{cases} \epsilon & \text{if } \mathbf{L}_i = 0 \text{ or} \\ \epsilon \times \mathrm{ABS}(\mathbf{L}_i) & \text{otherwise} \end{cases}$$

where $\epsilon = 10^{-4}$ by default; you can change this with the SINGULAR= option. Continued fractions (like 1/3) should be specified to at least six decimal places, or the DIVISOR parameter should be used.

Comparing Groups

An important task in analyzing data with classification effects is to estimate the typical response for each level of a given effect; often, you also want to compare these estimates to determine which levels are equivalent in terms of the response. You can perform this task in two ways with the GLM procedure: with direct, arithmetic group means; and with so-called *least squares means* (LS-means).

Means versus LS-Means

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glmde4.sas.)

Computing and comparing arithmetic means—either simple or weighted within-group averages of the input data—is a familiar and well-studied statistical process. This is the right approach to summarizing and comparing groups for one-way and balanced designs. However, in unbalanced designs with more than one effect, the arithmetic mean for a group might not accurately reflect the "typical" response for that group, since it does not take other effects into account.

For example, the following analysis of an unbalanced two-way design produces the ANOVA, means, and LS-means shown in Figure 53.18, Figure 53.19, and Figure 53.20.

```
data twoway;
   input Treatment Block y @@;
   datalines;
1 1 17   1 1 28   1 1 19   1 1 21   1 1 19
1 2 43   1 2 30   1 2 39   1 2 44   1 2 44
1 3 16
2 1 21   2 1 21   2 1 24   2 1 25
2 2 39   2 2 45   2 2 42   2 2 47
2 3 19   2 3 22   2 3 16
3 1 22   3 1 30   3 1 33   3 1 31
3 2 46
3 3 26   3 3 31   3 3 26   3 3 33   3 3 29   3 3 25
;

title "Unbalanced Two-way Design";
ods select ModelANOVA Means LSMeans;

proc glm data=twoway;
   class Treatment Block;
   model y = Treatment|Block;
   means Treatment;
   lsmeans Treatment;
run;

ods select all;
```

Figure 53.18 ANOVA Results for Unbalanced Two-Way Design

Unbalanced Two-way Design

The GLM Procedure

Dependent Variable: y

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Treatment	2	8.060606	4.030303	0.24	0.7888
Block	2	2621.864124	1310.932062	77.95	<.0001
Treatment*Block	4	32.684361	8.171090	0.49	0.7460

Figure 53.18 *continued*

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Treatment	2	266.130682	133.065341	7.91	0.0023
Block	2	1883.729465	941.864732	56.00	<.0001
Treatment*Block	4	32.684361	8.171090	0.49	0.7460

Figure 53.19 Treatment Means for Unbalanced Two-Way Design

Unbalanced Two-way Design

The GLM Procedure

		y	
Level of Treatment	N	Mean	Std Dev
1	11	29.0909091	11.5104695
2	11	29.1818182	11.5569735
3	11	30.1818182	6.3058414

Figure 53.20 Treatment LS-means for Unbalanced Two-Way Design

Unbalanced Two-way Design

The GLM Procedure
Least Squares Means

Treatment	y LSMEAN
1	25.6000000
2	28.3333333
3	34.4444444

No matter how you look at them, these data exhibit a strong effect due to the blocks (F test $p < 0.0001$) and no significant interaction between treatments and blocks (F test $p > 0.7$). But the lack of balance affects how the treatment effect is interpreted: in a main-effects-only model, there are no significant differences between the treatment means themselves (Type I F test $p > 0.7$), but there are highly significant differences between the treatment means corrected for the block effects (Type III F test $p < 0.01$).

LS-means are, in effect, within-group means appropriately adjusted for the other effects in the model. More precisely, they estimate the marginal means for a balanced population (as opposed to the unbalanced design). For this reason, they are also called *estimated population marginal means* by Searle, Speed, and Milliken (1980). In the same way that the Type I F test assesses differences between the arithmetic treatment means (when the treatment effect comes first in the model), the Type III F test assesses differences between the LS-means. Accordingly, for the unbalanced two-way design, the discrepancy between the Type I and Type III tests is reflected in the arithmetic treatment means and treatment LS-means, as shown in Figure 53.19 and Figure 53.20. See the section "Construction of Least Squares Means" on page 4244 for more on LS-means.

Note that, while the arithmetic means are always uncorrelated (under the usual assumptions for analysis of variance; see page 4206), the LS-means might not be. This fact complicates the problem of multiple comparisons for LS-means; see the following section.

Multiple Comparisons

When comparing more than two means, an ANOVA F test tells you whether the means are significantly different from each other, but it does not tell you which means differ from which other means. Multiple-comparison procedures (MCPs), also called *mean separation tests*, give you more detailed information about the differences among the means. The goal in multiple comparisons is to compare the average effects of three or more "treatments" (for example, drugs, groups of subjects) to decide which treatments are better, which ones are worse, and by how much, while controlling the probability of making an incorrect decision. A variety of multiple-comparison methods are available with the MEANS and LSMEANS statement in the GLM procedure.

The following classification is due to Hsu (1996). Multiple-comparison procedures can be categorized in two ways: by the comparisons they make and by the strength of inference they provide. With respect to which comparisons are made, the GLM procedure offers two types:

- comparisons between all pairs of means
- comparisons between a control and all other means

The strength of inference says what can be inferred about the structure of the means when a test is significant; it is related to what type of error rate the MCP controls. MCPs available in the GLM procedure provide one of the following types of inference, in order from weakest to strongest:

- Individual: differences between means, unadjusted for multiplicity
- Inhomogeneity: means are different
- Inequalities: which means are different
- Intervals: simultaneous confidence intervals for mean differences

Methods that control only individual error rates are not true MCPs at all. Methods that yield the strongest level of inference, simultaneous confidence intervals, are usually preferred, since they enable you not only to say which means are different but also to put confidence bounds on *how much* they differ, making it easier to assess the practical significance of a difference. They are also less likely to lead nonstatisticians to the invalid conclusion that nonsignificantly different sample means imply equal population means. Interval MCPs are available for both arithmetic means and LS-means via the MEANS and LSMEANS statements, respectively.[1]

Table 53.12 and Table 53.13 display MCPs available in PROC GLM for all pairwise comparisons and comparisons with a control, respectively, along with associated strength of inference and the syntax (when applicable) for both the MEANS and the LSMEANS statements.

[1] The Duncan-Waller method does not fit into the preceding scheme, since it is based on the Bayes risk rather than any particular error rate.

Table 53.12 Multiple-Comparison Procedures for All Pairwise Comparisons

Method	Strength of Inference	Syntax	
		MEANS	LSMEANS
Student's *t*	Individual	T	PDIFF ADJUST=T
Duncan	Individual	DUNCAN	
Student-Newman-Keuls	Inhomogeneity	SNK	
REGWQ	Inequalities	REGWQ	
Tukey-Kramer	Intervals	TUKEY	PDIFF ADJUST=TUKEY
Bonferroni	Intervals	BON	PDIFF ADJUST=BON
Sidak	Intervals	SIDAK	PDIFF ADJUST=SIDAK
Scheffé	Intervals	SCHEFFE	PDIFF ADJUST=SCHEFFE
SMM	Intervals	SMM	PDIFF ADJUST=SMM
Gabriel	Intervals	GABRIEL	
Simulation	Intervals		PDIFF ADJUST=SIMULATE

Table 53.13 Multiple-Comparison Procedures for Comparisons with a Control

Method	Strength of Inference	Syntax	
		MEANS	LSMEANS
Student's *t*	Individual		PDIFF=CONTROL ADJUST=T
Dunnett	Intervals	DUNNETT	PDIFF=CONTROL ADJUST=DUNNETT
Bonferroni	Intervals		PDIFF=CONTROL ADJUST=BON
Sidak	Intervals		PDIFF=CONTROL ADJUST=SIDAK
Scheffé	Intervals		PDIFF=CONTROL ADJUST=SCHEFFE
SMM	Intervals		PDIFF=CONTROL ADJUST=SMM
Simulation	Intervals		PDIFF=CONTROL ADJUST=SIMULATE

NOTE: One-sided Dunnett's tests are also available from the MEANS statement with the DUNNETTL and DUNNETTU options and from the LSMEANS statement with PDIFF=CONTROLL and PDIFF=CONTROLU.

A note concerning the ODS tables for the results of the PDIFF or TDIFF options in the LSMEANS statement: The *p*/*t*-values for differences are displayed in columns of the LSMeans table for PDIFF/TDIFF=CONTROL or PDIFF/TDIFF=ANOM, and for PDIFF/TDIFF=ALL when there are only two LS-means. Otherwise (for PDIFF/TDIFF=ALL when there are more than two LS-means), the *p*/*t*-values for differences are displayed in a separate table called Diff.

Details of these multiple comparison methods are given in the following sections.

Pairwise Comparisons

All the methods discussed in this section depend on the standardized pairwise differences $t_{ij} = (\bar{y}_i - \bar{y}_j)/\hat{\sigma}_{ij}$, where the parts of this expression are defined as follows:

- i and j are the indices of two groups

- $\bar{y}_i$ and $\bar{y}_j$ are the means or LS-means for groups i and j
- $\hat{\sigma}_{ij}$ is the square root of the estimated variance of $\bar{y}_i - \bar{y}_j$. For simple arithmetic means, $\hat{\sigma}^2_{ij} = s^2(1/n_i + 1/n_j)$, where n_i and n_j are the sizes of groups i and j, respectively, and s^2 is the mean square for error, with ν degrees of freedom. For weighted arithmetic means, $\hat{\sigma}^2_{ij} = s^2(1/w_i + 1/w_j)$, where w_i and w_j are the sums of the weights in groups i and j, respectively. Finally, for LS-means defined by the linear combinations $\mathbf{l}'_i\mathbf{b}$ and $\mathbf{l}'_j\mathbf{b}$ of the parameter estimates, $\hat{\sigma}^2_{ij} = s^2\mathbf{l}'_i(\mathbf{X}'\mathbf{X})^{-}\mathbf{l}_j$.

Furthermore, all of the methods are discussed in terms of significance tests of the form

$$|t_{ij}| \quad \geq \quad c(\alpha)$$

where $c(\alpha)$ is some constant depending on the significance level. Such tests can be inverted to form confidence intervals of the form

$$(\bar{y}_i - \bar{y}_j) - \hat{\sigma}_{ij}c(\alpha) \quad \leq \quad \mu_i - \mu_j \quad \leq \quad (\bar{y}_i - \bar{y}_j) + \hat{\sigma}_{ij}c(\alpha)$$

The simplest approach to multiple comparisons is to do a t test on every pair of means (the T option in the MEANS statement, ADJUST=T in the LSMEANS statement). For the ith and jth means, you can reject the null hypothesis that the population means are equal if

$$|t_{ij}| \quad \geq \quad t(\alpha;\nu)$$

where α is the significance level, ν is the number of error degrees of freedom, and $t(\alpha;\nu)$ is the two-tailed critical value from a Student's t distribution. If the cell sizes are all equal to, say, n, the preceding formula can be rearranged to give

$$|\bar{y}_i - \bar{y}_j| \quad \geq \quad t(\alpha;\nu)s\sqrt{\frac{2}{n}}$$

the value of the right-hand side being Fisher's least significant difference (LSD).

There is a problem with repeated t tests, however. Suppose there are 10 means and each t test is performed at the 0.05 level. There are $10(10-1)/2 = 45$ pairs of means to compare, each with a 0.05 probability of a type 1 error (a false rejection of the null hypothesis). The chance of making at least one type 1 error is much higher than 0.05. It is difficult to calculate the exact probability, but you can derive a pessimistic approximation by assuming that the comparisons are independent, giving an upper bound to the probability of making at least one type 1 error (the experimentwise error rate) of

$$1 - (1 - 0.05)^{45} \quad = \quad 0.90$$

The actual probability is somewhat less than 0.90, but as the number of means increases, the chance of making at least one type 1 error approaches 1.

If you decide to control the individual type 1 error rates for each comparison, you are controlling the individual or comparisonwise error rate. On the other hand, if you want to control the overall type 1 error rate for all the comparisons, you are controlling the experimentwise error rate. It is up to you to decide whether to control the comparisonwise error rate or the experimentwise error rate, but there are many situations in which

the experimentwise error rate should be held to a small value. Statistical methods for comparing three or more means while controlling the probability of making at least one type 1 error are called *multiple-comparison procedures*.

It has been suggested that the experimentwise error rate can be held to the α level by performing the overall ANOVA F test at the α level and making further comparisons only if the F test is significant, as in Fisher's protected LSD. This assertion is false if there are more than three means (Einot and Gabriel 1975). Consider again the situation with 10 means. Suppose that one population mean differs from the others by such a sufficiently large amount that the power (probability of correctly rejecting the null hypothesis) of the F test is near 1 but that all the other population means are equal to each other. There will be $9(9-1)/2 = 36$ t tests of true null hypotheses, with an upper limit of 0.84 on the probability of at least one type 1 error. Thus, you must distinguish between the experimentwise error rate under the complete null hypothesis, in which all population means are equal, and the experimentwise error rate under a partial null hypothesis, in which some means are equal but others differ. The following abbreviations are used in the discussion:

CER	comparisonwise error rate
EERC	experimentwise error rate under the complete null hypothesis
MEER	maximum experimentwise error rate under any complete or partial null hypothesis

These error rates are associated with the different strengths of inference discussed on page 4231: individual tests control the CER; tests for inhomogeneity of means control the EERC; tests that yield confidence inequalities or confidence intervals control the MEER. A preliminary F test controls the EERC but not the MEER.

You can control the MEER at the α level by setting the CER to a sufficiently small value. The Bonferroni inequality (Miller 1981) has been widely used for this purpose. If

$$\text{CER} = \frac{\alpha}{c}$$

where c is the total number of comparisons, then the MEER is less than α. Bonferroni t tests (the BON option in the MEANS statement, ADJUST=BON in the LSMEANS statement) with $\text{MEER} < \alpha$ declare two means to be significantly different if

$$|t_{ij}| \geq t(\epsilon; \nu)$$

where

$$\epsilon = \frac{2\alpha}{k(k-1)}$$

for comparison of k means.

Šidák (1967) has provided a tighter bound, showing that

$$\text{CER} = 1 - (1-\alpha)^{1/c}$$

also ensures that $\mathrm{MEER} \leq \alpha$ for any set of c comparisons. A Sidak t test (Games 1977), provided by the SIDAK option, is thus given by

$$|t_{ij}| \quad \geq \quad t(\epsilon; \nu)$$

where

$$\epsilon \quad = \quad 1 - (1 - \alpha)^{\frac{2}{k(k-1)}}$$

for comparison of k means.

You can use the Bonferroni additive inequality and the Sidak multiplicative inequality to control the MEER for any set of contrasts or other hypothesis tests, not just pairwise comparisons. The Bonferroni inequality can provide simultaneous inferences in any statistical application requiring tests of more than one hypothesis. Other methods discussed in this section for pairwise comparisons can also be adapted for general contrasts (Miller 1981).

Scheffé (1953, 1959) proposes another method to control the MEER for any set of contrasts or other linear hypotheses in the analysis of linear models, including pairwise comparisons, obtained with the SCHEFFE option. Two means are declared significantly different if

$$|t_{ij}| \quad \geq \quad \sqrt{\mathrm{DF} \cdot F(\alpha; \mathrm{DF}, \nu)}$$

where $F(\alpha; \mathrm{DF}, \nu)$ is the α-level critical value of an F distribution with DF numerator degrees of freedom and ν denominator degrees of freedom. The value of DF is $k - 1$ for the MEANS statement, but in other statements the precise definition depends on context. For the LSMEANS statement, DF is the rank of the contrast matrix $\mathbf{L}$ for LS-means differences. In more general contexts—for example, the ESTIMATE or LSMESTIMATE statements in PROC GLIMMIX—DF is the rank of the contrast covariance matrix $\mathbf{L}\mathrm{Cov}(\mathbf{b})\mathbf{L}'$.

Scheffé's test is compatible with the overall ANOVA F test in that Scheffé's method never declares a contrast significant if the overall F test is nonsignificant. Most other multiple-comparison methods can find significant contrasts when the overall F test is nonsignificant and, therefore, suffer a loss of power when used with a preliminary F test.

Scheffé's method might be more powerful than the Bonferroni or Sidak method if the number of comparisons is large relative to the number of means. For pairwise comparisons, Sidak t tests are generally more powerful.

Tukey (1952, 1953) proposes a test designed specifically for pairwise comparisons based on the studentized range, sometimes called the "honestly significant difference test," that controls the MEER when the sample sizes are equal. Tukey (1953) and Kramer (1956) independently propose a modification for unequal cell sizes. The Tukey or Tukey-Kramer method is provided by the TUKEY option in the MEANS statement and the ADJUST=TUKEY option in the LSMEANS statement. This method has fared extremely well in Monte Carlo studies (Dunnett 1980). In addition, Hayter (1984) gives a proof that the Tukey-Kramer procedure controls the MEER for means comparisons, and Hayter (1989) describes the extent to which the Tukey-Kramer procedure has been proven to control the MEER for LS-means comparisons. The Tukey-Kramer method is more powerful than the Bonferroni, Sidak, or Scheffé method for pairwise comparisons. Two means are considered significantly different by the Tukey-Kramer criterion if

$$|t_{ij}| \quad \geq \quad q(\alpha; k, \nu)/\sqrt{2}$$

where $q(\alpha; k, \nu)$ is the α-level critical value of a studentized range distribution of k independent normal random variables with ν degrees of freedom.

Hochberg (1974) devised a method (the GT2 or SMM option) similar to Tukey's, but it uses the studentized maximum modulus instead of the studentized range and employs the uncorrelated t inequality of Šidák (1967). It is proven to hold the MEER at a level not exceeding α with unequal sample sizes. It is generally less powerful than the Tukey-Kramer method and always less powerful than Tukey's test for equal cell sizes. Two means are declared significantly different if

$$|t_{ij}| \quad \geq \quad m(\alpha; c, \nu)$$

where $m(\alpha; c, \nu)$ is the α-level critical value of the studentized maximum modulus distribution of c independent normal random variables with ν degrees of freedom and $c = k(k-1)/2$.

Gabriel (1978) proposes another method (the GABRIEL option) based on the studentized maximum modulus. This method is applicable only to arithmetic means. It rejects if

$$\frac{|\bar{y}_i - \bar{y}_j|}{s\left(\frac{1}{\sqrt{2n_i}} + \frac{1}{\sqrt{2n_j}}\right)} \quad \geq \quad m(\alpha; k, \nu)$$

For equal cell sizes, Gabriel's test is equivalent to Hochberg's GT2 method. For unequal cell sizes, Gabriel's method is more powerful than GT2 but might become liberal with highly disparate cell sizes (see also Dunnett (1980)). Gabriel's test is the only method for unequal sample sizes that lends itself to a graphical representation as intervals around the means. Assuming $\bar{y}_i > \bar{y}_j$, you can rewrite the preceding inequality as

$$\bar{y}_i - m(\alpha; k, \nu)\frac{s}{\sqrt{2n_i}} \quad \geq \quad \bar{y}_j + m(\alpha; k, \nu)\frac{s}{\sqrt{2n_j}}$$

The expression on the left does not depend on j, nor does the expression on the right depend on i. Hence, you can form what Gabriel calls an (l, u)-interval around each sample mean and declare two means to be significantly different if their (l, u)-intervals do not overlap. See Hsu (1996, section 5.2.1.1) for a discussion of other methods of graphically representing all pairwise comparisons.

Comparing All Treatments to a Control

One special case of means comparison is that in which the only comparisons that need to be tested are between a set of new treatments and a single control. In this case, you can achieve better power by using a method that is restricted to test only comparisons to the single control mean. Dunnett (1955) proposes a test for this situation that declares a mean significantly different from the control if

$$|t_{i0}| \quad \geq \quad d(\alpha; k, \nu, \rho_1, \ldots, \rho_{k-1})$$

where $\bar{y}_0$ is the control mean and $d(\alpha; k, \nu, \rho_1, \ldots, \rho_{k-1})$ is the critical value of the "many-to-one t statistic" (Miller 1981; Krishnaiah and Armitage 1966) for k means to be compared to a control, with ν error degrees of freedom and correlations $\rho_1, \ldots, \rho_{k-1}$, $\rho_i = n_i/(n_0 + n_i)$. The correlation terms arise because each of the treatment means is being compared to the same control. Dunnett's test holds the MEER to a level not exceeding the stated α.

Analysis of Means: Comparing Each Treatments to the Average

Analysis of means (ANOM) refers to a technique for comparing group means and displaying the comparisons graphically so that you can easily see which ones are different. Means are judged as different if they are significantly different from the overall average, with significance adjusted for multiplicity. The overall average is computed as a weighted mean of the LS-means, the weights being inversely proportional to the variances. If you use the PDIFF=ANOM option in the LSMEANS statement, the procedure will display the p-values (adjusted for multiplicity, by default) for tests of the differences between each LS-mean and the average LS-mean. The ANOM procedure in SAS/QC software displays both tables and graphics for the analysis of means with a variety of response types. For one-way designs, confidence intervals for PDIFF=ANOM comparisons are equivalent to the results of PROC ANOM. The difference is that PROC GLM directly displays the confidence intervals for the differences, while the graphical output of PROC ANOM displays them as decision limits around the overall mean.

If the LS-means being compared are uncorrelated, exact adjusted p-values and critical values for confidence limits can be computed; see Nelson (1982, 1991, 1993) and Guirguis and Tobias (2004). For correlated LS-means, an approach similar to that of Hsu (1992) is employed, using a factor-analytic approximation of the correlation between the LS-means to derive approximate "effective sample sizes" for which exact critical values are computed. Note that computing the exact adjusted p-values and critical values for unbalanced designs can be computationally intensive. A simulation-based approach, as specified by the ADJUST=SIM option, while nondeterministic, might provide inferences that are accurate enough in much less time. See the section "Approximate and Simulation-Based Methods" on page 4237 for more details.

Approximate and Simulation-Based Methods

Tukey's, Dunnett's, and Nelson's tests are all based on the same general quantile calculation:

$$q^t(\alpha, \nu, R) \quad = \quad \{q \ni P(\max(|t_1|, \ldots, |t_n|) > q) = \alpha\}$$

where the t_i have a joint multivariate t distribution with ν degrees of freedom and correlation matrix R. In general, evaluating $q^t(\alpha, \nu, R)$ requires repeated numerical calculation of an $(n+1)$-fold integral. This is usually intractable, but the problem reduces to a feasible 2-fold integral when R has a certain symmetry in the case of Tukey's test, and a *factor analytic structure* (Hsu 1992) in the case of Dunnett's and Nelson's tests. The R matrix has the required symmetry for exact computation of Tukey's test in the following two cases:

- The t_is are studentized differences between $k(k-1)/2$ pairs of k uncorrelated means with equal variances—that is, equal sample sizes.
- The t_is are studentized differences between $k(k-1)/2$ pairs of k LS-means from a *variance-balanced* design (for example, a balanced incomplete block design).

See Hsu (1992, 1996) for more information. The R matrix has the factor analytic structure for exact computation of Dunnett's and Nelson's tests in the following two cases:

- if the t_is are studentized differences between $k-1$ means and a control mean, all uncorrelated. (Dunnett's one-sided methods depend on a similar probability calculation, without the absolute values.) Note that it is not required that the variances of the means (that is, the sample sizes) be equal.
- if the t_is are studentized differences between $k-1$ LS-means and a control LS-mean from either a *variance-balanced* design, or a design in which the other factors are *orthogonal* to the treatment factor (for example, a randomized block design with proportional cell frequencies)

However, other important situations that do **not** result in a correlation matrix R that has the structure for exact computation are the following:

- all pairwise differences with unequal sample sizes
- differences between LS-means in many unbalanced designs

In these situations, exact calculation of $q^t(\alpha, \nu, R)$ is intractable in general. Most of the preceding methods can be viewed as using various approximations for $q^t(\alpha, \nu, R)$. When the sample sizes are unequal, the Tukey-Kramer test is equivalent to another approximation. For comparisons with a control when the correlation R does not have a factor analytic structure, Hsu (1992) suggests approximating R with a matrix R^* that does have such a structure and correspondingly approximating $q^t(\alpha, \nu, R)$ with $q^t(\alpha, \nu, R^*)$. When you request Dunnett's or Nelson's test for LS-means (the PDIFF=CONTROL and ADJUST=DUNNETT options or the PDIFF=ANOM and ADJUST=NELSON options, respectively), the GLM procedure automatically uses Hsu's approximation when appropriate.

Finally, Edwards and Berry (1987) suggest calculating $q^t(\alpha, \nu, R)$ by simulation. Multivariate t vectors are sampled from a distribution with the appropriate ν and R parameters, and Edwards and Berry (1987) suggest estimating $q^t(\alpha, \nu, R)$ by $\hat{q}$, the αth percentile of the observed values of $\max(|t_1|, \ldots, |t_n|)$. Sufficient samples are generated for the true $P(\max(|t_1|, \ldots, |t_n|) > \hat{q})$ to be within a certain accuracy radius γ of α with accuracy confidence $100(1 - \epsilon)$. You can approximate $q^t(\alpha, \nu, R)$ by simulation for comparisons between LS-means by specifying ADJUST=SIM (with any PDIFF= type). By default, $\gamma = 0.005$ and $\epsilon = 0.01$, so that the tail area of $\hat{q}$ is within 0.005 of α with 99% confidence. You can use the ACC= and EPS= options with ADJUST=SIM to reset γ and ϵ, or you can use the NSAMP= option to set the sample size directly. You can also control the random number sequence with the SEED= option.

Hsu and Nelson (1998) suggest a more accurate simulation method for estimating $q^t(\alpha, \nu, R)$, using a control variate adjustment technique. The same independent, standardized normal variates that are used to generate multivariate t vectors from a distribution with the appropriate ν and R parameters are also used to generate multivariate t vectors from a distribution for which the exact value of $q^t(\alpha, \nu, R)$ is known. $\max(|t_1|, \ldots, |t_n|)$ for the second sample is used as a control variate for adjusting the quantile estimate based on the first sample; see Hsu and Nelson (1998) for more details. The control variate adjustment has the drawback that it takes somewhat longer than the crude technique of Edwards and Berry (1987), but it typically yields an estimate that is many times more accurate. In most cases, if you are using ADJUST=SIM, then you should specify ADJUST=SIM(CVADJUST). You can also specify ADJUST=SIM(CVADJUST REPORT) to display a summary of the simulation that includes, among other things, the actual accuracy radius γ, which should be substantially smaller than the target accuracy radius (0.005 by default).

Multiple-Stage Tests

You can use all of the methods discussed so far to obtain simultaneous confidence intervals (Miller 1981). By sacrificing the facility for simultaneous estimation, you can obtain simultaneous tests with greater power by using multiple-stage tests (MSTs). MSTs come in both step-up and step-down varieties (Welsch 1977). The step-down methods, which have been more widely used, are available in SAS/STAT software.

Step-down MSTs first test the homogeneity of all the means at a level γ_k. If the test results in a rejection, then each subset of $k - 1$ means is tested at level γ_{k-1}; otherwise, the procedure stops. In general, if the hypothesis of homogeneity of a set of p means is rejected at the γ_p level, then each subset of $p - 1$ means is tested at the γ_{p-1} level; otherwise, the set of p means is considered not to differ significantly and none of its subsets are tested. The many varieties of MSTs that have been proposed differ in the levels γ_p and the

statistics on which the subset tests are based. Clearly, the EERC of a step-down MST is not greater than γ_k, and the CER is not greater than γ_2, but the MEER is a complicated function of γ_p, $p = 2, \ldots, k$.

With unequal cell sizes, PROC GLM uses the harmonic mean of the cell sizes as the common sample size. However, since the resulting operating characteristics can be undesirable, MSTs are recommended only for the balanced case. When the sample sizes are equal, using the range statistic enables you to arrange the means in ascending or descending order and test only contiguous subsets. But if you specify the F statistic, this shortcut cannot be taken. For this reason, only range-based MSTs are implemented. It is common practice to report the results of an MST by writing the means in such an order and drawing lines parallel to the list of means spanning the homogeneous subsets. This form of presentation is also convenient for pairwise comparisons with equal cell sizes.

The best-known MSTs are the Duncan (the DUNCAN option) and Student-Newman-Keuls (the SNK option) methods (Miller 1981). Both use the studentized range statistic and, hence, are called *multiple range tests*. Duncan's method is often called the "new" multiple range test despite the fact that it is one of the oldest MSTs in current use.

The Duncan and SNK methods differ in the γ_p values used. For Duncan's method, they are

$$\gamma_p = 1 - (1 - \alpha)^{p-1}$$

whereas the SNK method uses

$$\gamma_p = \alpha$$

Duncan's method controls the CER at the α level. Its operating characteristics appear similar to those of Fisher's unprotected LSD or repeated t tests at level α (Petrinovich and Hardyck 1969). Since repeated t tests are easier to compute, easier to explain, and applicable to unequal sample sizes, Duncan's method is not recommended. Several published studies (for example, Carmer and Swanson (1973)) have claimed that Duncan's method is superior to Tukey's because of greater power without considering that the greater power of Duncan's method is due to its higher type 1 error rate (Einot and Gabriel 1975).

The SNK method holds the EERC to the α level but does not control the MEER (Einot and Gabriel 1975). Consider ten population means that occur in five pairs such that means within a pair are equal, but there are large differences between pairs. If you make the usual sampling assumptions and also assume that the sample sizes are very large, all subset homogeneity hypotheses for three or more means are rejected. The SNK method then comes down to five independent tests, one for each pair, each at the α level. Letting α be 0.05, the probability of at least one false rejection is

$$1 - (1 - 0.05)^5 = 0.23$$

As the number of means increases, the MEER approaches 1. Therefore, the SNK method cannot be recommended.

A variety of MSTs that control the MEER have been proposed, but these methods are not as well known as those of Duncan and SNK. One approach (Ryan 1959, 1960; Einot and Gabriel 1975; Welsch 1977) sets

$$\gamma_p = \begin{cases} 1 - (1 - \alpha)^{p/k} & \text{for } p < k - 1 \\ \alpha & \text{for } p \geq k - 1 \end{cases}$$

You can use range statistics, leading to what is called the REGWQ method, after the authors' initials. If you assume that the sample means have been arranged in descending order from $\bar{y}_1$ through $\bar{y}_k$, the homogeneity of means $\bar{y}_i, \ldots, \bar{y}_j, i < j$, is rejected by REGWQ if

$$\bar{y}_i - \bar{y}_j \quad \geq \quad q(\gamma_p; p, \nu)\frac{s}{\sqrt{n}}$$

where $p = j - i + 1$ and the summations are over $u = i, \ldots, j$ (Einot and Gabriel 1975). To ensure that the MEER is controlled, the current implementation checks whether $q(\gamma_p; p, \nu)$ is monotonically increasing in p. If not, then a set of critical values that are increasing in p is substituted instead.

REGWQ appears to be the most powerful step-down MST in the current literature (for example, Ramsey 1978). Use of a preliminary F test decreases the power of all the other multiple-comparison methods discussed previously except for Scheffé's test.

Bayesian Approach

Waller and Duncan (1969) and Duncan (1975) take an approach to multiple comparisons that differs from all the methods previously discussed in minimizing the Bayes risk under additive loss rather than controlling type 1 error rates. For each pair of population means μ_i and μ_j, null (H_0^{ij}) and alternative (H_a^{ij}) hypotheses are defined:

$$\begin{aligned} H_0^{ij}: &\quad \mu_i - \mu_j \leq 0 \\ H_a^{ij}: &\quad \mu_i - \mu_j > 0 \end{aligned}$$

For any i, j pair, let d_0 indicate a decision in favor of H_0^{ij} and d_a indicate a decision in favor of H_a^{ij}, and let $\delta = \mu_i - \mu_j$. The loss function for the decision on the i, j pair is

$$L(d_0 \mid \delta) \quad = \quad \begin{cases} 0 & \text{if } \delta \leq 0 \\ \delta & \text{if } \delta > 0 \end{cases}$$

$$L(d_a \mid \delta) \quad = \quad \begin{cases} -k\delta & \text{if } \delta \leq 0 \\ 0 & \text{if } \delta > 0 \end{cases}$$

where k represents a constant that you specify rather than the number of means. The loss for the joint decision involving all pairs of means is the sum of the losses for each individual decision. The population means are assumed to have a normal prior distribution with unknown variance, the logarithm of the variance of the means having a uniform prior distribution. For the i, j pair, the null hypothesis is rejected if

$$\bar{y}_i - \bar{y}_j \quad \geq \quad t_B s\sqrt{\frac{2}{n}}$$

where t_B is the Bayesian t value (Waller and Kemp 1976) depending on k, the F statistic for the one-way ANOVA, and the degrees of freedom for F. The value of t_B is a decreasing function of F, so the Waller-Duncan test (specified by the WALLER option) becomes more liberal as F increases.

Recommendations

In summary, if you are interested in several individual comparisons and are not concerned about the effects of multiple inferences, you can use repeated *t* tests or Fisher's unprotected LSD. If you are interested in all pairwise comparisons or all comparisons with a control, you should use Tukey's or Dunnett's test, respectively, in order to make the strongest possible inferences. If you have weaker inferential requirements and, in particular, if you do not want confidence intervals for the mean differences, you should use the REGWQ method. Finally, if you agree with the Bayesian approach and Waller and Duncan's assumptions, you should use the Waller-Duncan test.

Interpretation of Multiple Comparisons

When you interpret multiple comparisons, remember that failure to reject the hypothesis that two or more means are equal should not lead you to conclude that the population means are, in fact, equal. Failure to reject the null hypothesis implies only that the difference between population means, if any, is not large enough to be detected with the given sample size. A related point is that nonsignificance is nontransitive: that is, given three sample means, the largest and smallest might be significantly different from each other, while neither is significantly different from the middle one. Nontransitive results of this type occur frequently in multiple comparisons.

Multiple comparisons can also lead to counterintuitive results when the cell sizes are unequal. Consider four cells labeled A, B, C, and D, with sample means in the order A>B>C>D. If A and D each have two observations, and B and C each have 10,000 observations, then the difference between B and C might be significant, while the difference between A and D is not.

Simple Effects

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glmde5.sas.)

Suppose you use the following statements to fit a full factorial model to a two-way design:

```
data twoway;
   input A B Y @@;
   datalines;
1 1 10.6   1 1 11.0   1 1 10.6   1 1 11.3
1 2 -0.2   1 2  1.3   1 2 -0.2   1 2  0.2
1 3  0.1   1 3  0.4   1 3 -0.4   1 3  1.0
2 1 19.7   2 1 19.3   2 1 18.5   2 1 20.4
2 2 -0.2   2 2  0.5   2 2  0.8   2 2 -0.4
2 3 -0.9   2 3 -0.1   2 3 -0.2   2 3 -1.7
3 1 29.7   3 1 29.6   3 1 29.0   3 1 30.2
3 2  1.5   3 2  0.2   3 2 -1.5   3 2  1.3
3 3  0.2   3 3  0.4   3 3 -0.4   3 3 -2.2
;

proc glm data=twoway;
   class A B;
   model Y = A B A*B;
run;
```

Partial results for the analysis of variance are shown in Figure 53.21. The Type I and Type III results are the same because this is a balanced design.

Figure 53.21 Two-Way Design with Significant Interaction

The GLM Procedure

Dependent Variable: Y

Source	DF	Type I SS	Mean Square	F Value	Pr > F
A	2	219.905000	109.952500	165.11	<.0001
B	2	3206.101667	1603.050833	2407.25	<.0001
A*B	4	487.103333	121.775833	182.87	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
A	2	219.905000	109.952500	165.11	<.0001
B	2	3206.101667	1603.050833	2407.25	<.0001
A*B	4	487.103333	121.775833	182.87	<.0001

The interaction A*B is significant, indicating that the effect of A depends on the level of B. In some cases, you might be interested in looking at the differences between predicted values across A for different levels of B. Winer (1971) calls this the *simple effects* of A. You can compute simple effects with the LSMEANS statement by specifying the SLICE= option. In this case, since the GLM procedure is interactive, you can compute the simple effects of A by submitting the following statements after the preceding statements.

```
    lsmeans A*B / slice=B;
run;
```

The results are shown Figure 53.22. Note that A has a significant effect for B=1 but not for B=2 and B=3.

Figure 53.22 Interaction LS-Means and Simple Effects

The GLM Procedure
Least Squares Means

A	B	Y LSMEAN
1	1	10.8750000
1	2	0.2750000
1	3	0.2750000
2	1	19.4750000
2	2	0.1750000
2	3	-0.7250000
3	1	29.6250000
3	2	0.3750000
3	3	-0.5000000

The GLM Procedure
Least Squares Means

A*B Effect Sliced by B for Y					
B	DF	Sum of Squares	Mean Square	F Value	Pr > F
1	2	704.726667	352.363333	529.13	<.0001
2	2	0.080000	0.040000	0.06	0.9418
3	2	2.201667	1.100833	1.65	0.2103

Homogeneity of Variance in One-Way Models

One of the usual assumptions in using the GLM procedure is that the underlying errors are all uncorrelated with homogeneous variances (see page 4206). You can test this assumption in PROC GLM by using the HOVTEST option in the MEANS statement, requesting a *homogeneity of variance* test. This section discusses the computational details behind these tests. Note that the GLM procedure allows homogeneity of variance testing for simple one-way models only. Homogeneity of variance testing for more complex models is a subject of current research.

Bartlett (1937) proposes a test for equal variances that is a modification of the normal-theory likelihood ratio test (the HOVTEST=BARTLETT option). While Bartlett's test has accurate Type I error rates and optimal power when the underlying distribution of the data is normal, it can be very inaccurate if that distribution is even slightly nonnormal (Box 1953). Therefore, Bartlett's test is not recommended for routine use.

An approach that leads to tests that are much more robust to the underlying distribution is to transform the original values of the dependent variable to derive a *dispersion variable* and then to perform analysis of variance on this variable. The significance level for the test of homogeneity of variance is the p-value for the ANOVA F test on the dispersion variable. All of the homogeneity of variance tests available in PROC GLM except Bartlett's use this approach.

Levene's test (Levene 1960) is widely considered to be the standard homogeneity of variance test (the HOVTEST=LEVENE option). Levene's test is of the dispersion-variable-ANOVA form discussed previously, where the dispersion variable is either of the following:

$$\begin{aligned} z_{ij}^2 &= (y_{ij} - \bar{y}_i)^2 \quad \text{(TYPE=SQUARE, the default)} \\ z_{ij} &= |y_{ij} - \bar{y}_i| \quad \text{(TYPE=ABS)} \end{aligned}$$

O'Brien (1979) proposes a test (HOVTEST=OBRIEN) that is basically a modification of Levene's z_{ij}^2, using the dispersion variable

$$z_{ij}^W = \frac{(W + n_i - 2)n_i(y_{ij} - \bar{y}_i)^2 - W(n_i - 1)\sigma_i^2}{(n_i - 1)(n_i - 2)}$$

where n_i is the size of the ith group and σ_i^2 is its sample variance. You can use the W= option in parentheses to tune O'Brien's z_{ij}^W dispersion variable to match the suspected kurtosis of the underlying distribution. The choice of the value of the W= option is rarely critical. By default, W=0.5, as suggested by O'Brien (1979, 1981).

Finally, Brown and Forsythe (1974) suggest using the absolute deviations from the group *medians*:

$$z_{ij}^{\mathrm{BF}} = |y_{ij} - m_i|$$

where m_i is the median of the ith group. You can use the HOVTEST=BF option to specify this test.

Simulation results (Conover, Johnson, and Johnson 1981; Olejnik and Algina 1987) show that, while all of these ANOVA-based tests are reasonably robust to the underlying distribution, the Brown-Forsythe test seems best at providing power to detect variance differences while protecting the Type I error probability. However, since the within-group medians are required for the Brown-Forsythe test, it can be resource intensive if there are very many groups or if some groups are very large.

If one of these tests rejects the assumption of homogeneity of variance, you should use Welch's ANOVA instead of the usual ANOVA to test for differences between group means. However, this conclusion holds only if you use one of the robust homogeneity of variance tests (that is, not for HOVTEST=BARTLETT); even then, any homogeneity of variance test has too little power to be relied upon to always detect when Welch's ANOVA is appropriate. Unless the group variances are extremely different or the number of groups is large, the usual ANOVA test is relatively robust when the groups are all about the same size. As Box (1953) notes, "To make the preliminary test on variances is rather like putting to sea in a rowing boat to find out whether conditions are sufficiently calm for an ocean liner to leave port!"

Example 53.10 illustrates the use of the HOVTEST and WELCH options in the MEANS statement in testing for equal group variances and adjusting for unequal group variances in a one-way ANOVA.

Weighted Means

If you specify a WEIGHT statement and one or more of the multiple comparisons options, the variance of the difference between weighted group means for group i and j is computed as

$$\text{MSE} \times \left(\frac{1}{w_i} + \frac{1}{w_j} \right)$$

where w_i is the sum of the weights for the observations in group i.

Construction of Least Squares Means

To construct a least squares mean (LS-mean) for a particular level of a particular effect, construct a row vector $\mathbf{L}$ according to the following rules and use it in an ESTIMATE statement to compute the value of the LS-mean:

1. Set all L_i that correspond to covariates (continuous variables) to their mean value.
2. Consider effects that are contained by the particular effect. (For a definition of *containing*, see Chapter 16, "The Four Types of Estimable Functions.") Set the L_i that correspond to levels associated with the particular level equal to 1. Set all other L_i in these effects equal to 0.
3. Consider the particular effect. Set the L_i that correspond to the particular level equal to 1. Set the L_i that correspond to other levels equal to 0.
4. Consider the effects that contain the particular effect. If these effects are not nested within the particular effect, then set the L_i that correspond to the particular level to $1/k$, where k is the number of such columns. If these effects are nested within the particular effect, then set the L_i that correspond to the particular level to $1/(k_1 k_2)$, where k_1 is the number of nested levels within this combination of nested effects and k_2 is the number of such combinations. For L_i that correspond to other levels, use 0.
5. Consider other effects that are not yet considered. For each effect that has no nested factors, set all L_i that correspond to this effect to $1/j$, where j is the number of levels in the effect. For each effect that has nested factors, set all L_i that correspond to this effect to $1/(j_1 j_2)$, where j_1 is the number of nested levels within a particular combination of nested effects and j_2 is the number of such combinations.

The consequence of these rules is that the sum of the Xs within any classification effect is 1. This set of Xs forms a linear combination of the parameters that is checked for estimability before it is evaluated.

For example, consider the following model:

```
proc glm;
   class A B C;
   model Y=A B A*B C Z;
   lsmeans A B A*B C;
run;
```

Assume A has 3 levels, B has 2 levels, and C has 2 levels, and assume that every combination of levels of A and B exists in the data. Assume also that Z is a continuous variable with an average of 12.5. Then the least squares means are computed by the following linear combinations of the parameter estimates:

		A			B		A*B						C		
	μ	1	2	3	1	2	11	12	21	22	31	32	1	2	Z
LSM( )	1	1/3	1/3	1/3	1/2	1/2	1/6	1/6	1/6	1/6	1/6	1/6	1/2	1/2	12.5
LSM(A1)	1	1	0	0	1/2	1/2	1/2	1/2	0	0	0	0	1/2	1/2	12.5
LSM(A2)	1	0	1	0	1/2	1/2	0	0	1/2	1/2	0	0	1/2	1/2	12.5
LSM(A3)	1	0	0	1	1/2	1/2	0	0	0	0	1/2	1/2	1/2	1/2	12.5
LSM(B1)	1	1/3	1/3	1/3	1	0	1/3	0	1/3	0	1/3	0	1/2	1/2	12.5
LSM(B2)	1	1/3	1/3	1/3	0	1	0	1/3	0	1/3	0	1/3	1/2	1/2	12.5
LSM(AB11)	1	1	0	0	1	0	1	0	0	0	0	0	1/2	1/2	12.5
LSM(AB12)	1	1	0	0	0	1	0	1	0	0	0	0	1/2	1/2	12.5
LSM(AB21)	1	0	1	0	1	0	0	0	1	0	0	0	1/2	1/2	12.5
LSM(AB22)	1	0	1	0	0	1	0	0	0	1	0	0	1/2	1/2	12.5
LSM(AB31)	1	0	0	1	1	0	0	0	0	0	1	0	1/2	1/2	12.5
LSM(AB32)	1	0	0	1	0	1	0	0	0	0	0	1	1/2	1/2	12.5
LSM(C1)	1	1/3	1/3	1/3	1/2	1/2	1/6	1/6	1/6	1/6	1/6	1/6	1	0	12.5
LSM(C2)	1	1/3	1/3	1/3	1/2	1/2	1/6	1/6	1/6	1/6	1/6	1/6	0	1	12.5

Setting Covariate Values

By default, all covariate effects are set equal to their mean values for computation of standard LS-means. The AT option in the LSMEANS statement enables you to set the covariates to whatever values you consider interesting.

If any effect contains two or more covariates, the AT option sets the effect equal to the product of the individual means rather than the mean of the product (as with standard LS-means calculations). The AT MEANS option leaves covariates equal to their mean values (as with standard LS-means) and incorporates this adjustment to crossproducts of covariates.

For example, the following statements specify a model with a classification variable A and two continuous variables, x1 and x2:

```
class A;
model y = A x1 x2 x1*x2;
```

The coefficients for the continuous effects with various AT specifications are shown in the following table.

Syntax	x1	x2	x1*x2
`lsmeans A;`	$\overline{x_1}$	$\overline{x_2}$	$\overline{x_1 x_2}$
`lsmeans A / at means;`	$\overline{x_1}$	$\overline{x_2}$	$\overline{x_1} \cdot \overline{x_2}$
`lsmeans A / at x1=1.2;`	1.2	$\overline{x_2}$	$1.2 \cdot \overline{x_2}$
`lsmeans A / at (x1 x2)=(1.2 0.3);`	1.2	0.3	$1.2 \cdot 0.3$

For the first two LSMEANS statements, the A LS-mean coefficient for x1 is $\overline{x_1}$ (the mean of x1) and for x2 is $\overline{x_2}$ (the mean of x2). However, the coefficient for x1*x2 is $\overline{x_1 x_2}$ for the first LSMEANS statement, but it is $\overline{x_1} \cdot \overline{x_2}$ for the second LSMEANS statement. The third LSMEANS statement sets the coefficient for x1 equal to 1.2 and leaves the coefficient for x2 at $\overline{x_2}$, and the final LSMEANS statement sets these values to 1.2 and 0.3, respectively.

The covariate means used in computing the LS-means are affected by the AT option in the LSMEANS statement as follows:

- If you use a WEIGHT statement:
  - If you do not specify the AT option, unweighted covariate means are used for the covariate coefficients.
  - If you specify the AT option, weighted covariate means are used for the covariate coefficients for which no explicit AT values are specified (thus for all the covariate means if you specify AT MEANS).
- If any observations have missing dependent variable values:
  - If you do not specify the AT option, these observations are not included in computing the covariate means.
  - If you specify the AT option, these observations are included unless they form a missing cell (a combination of CLASS variables all of whose responses are missing).

These conditions are summarized in Table 53.14. You can use the E option in conjunction with the AT option to verify that the modified LS-means coefficients are the ones you want.

Table 53.14 Treatment of Covariate Means with and without the AT Option

	Without AT	With AT
WEIGHT statement	Weights not used	Weights used
Observations with missing dependent variable values	Observations not included	Observations included

The AT option is disabled if you specify the BYLEVEL option, in which case the coefficients for the covariates are set equal to their means within each level of the LS-mean effect in question.

Changing the Weighting Scheme

The standard LS-means have equal coefficients across classification effects; however, the OM option in the LSMEANS statement changes these coefficients to be proportional to those found in the input data set. This

adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in the original data set.

In computing the observed margins, PROC GLM uses all observations for which there are no missing independent variables, including those for which there are missing dependent variables. Also, if there is a WEIGHT variable, PROC GLM uses weighted margins to construct the LS-means coefficients. If the analysis data set is balanced or if you specify a simple one-way model, the LS-means will be unchanged by the OM option.

The BYLEVEL option modifies the observed-margins LS-means. Instead of computing the margins across the entire data set, PROC GLM computes separate margins for each level of the LS-mean effect in question. The resulting LS-means are actually equal to raw means in this case. The BYLEVEL option disables the AT option if it is specified.

Note that the MIXED procedure implements a more versatile form of the OM option, enabling you to specifying an alternative data set over which to compute observed margins. If you use the BYLEVEL option, too, then this data set is effectively the "population" over which the population marginal means are computed. For more information, see Chapter 84, "The MIXED Procedure."

You might want to use the E option in conjunction with either the OM or BYLEVEL option to verify that the modified LS-means coefficients are the ones you want. It is possible that the modified LS-means are not estimable when the standard ones are, or vice versa.

Estimability of LS-Means

LS-means are defined as certain linear combinations of the parameters. As such, it is possible for them to be inestimable. In fact, it is possible for a pair of LS-means to be both inestimable but their difference estimable. When this happens, only the entries that correspond to the estimable difference are computed and displayed in the Diffs table. If ADJUST=SIMULATE is specified when there are inestimable LS-means differences, adjusted results for all differences are displayed as missing.

Multivariate Analysis of Variance

If you fit several dependent variables to the same effects, you might want to make joint tests involving parameters of several dependent variables. Suppose you have p dependent variables, k parameters for each dependent variable, and n observations. The models can be collected into one equation:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\mathbf{Y}$ is $n \times p$, $\mathbf{X}$ is $n \times k$, $\boldsymbol{\beta}$ is $k \times p$, and $\boldsymbol{\epsilon}$ is $n \times p$. Each of the p models can be estimated and tested separately. However, you might also want to consider the joint distribution and test the p models simultaneously.

For multivariate tests, you need to make some assumptions about the errors. With p dependent variables, there are $n \times p$ errors that are independent across observations but not across dependent variables. Assume

$$\mathrm{vec}(\boldsymbol{\epsilon}) \sim N(\mathbf{0}, \mathbf{I}_n \otimes \boldsymbol{\Sigma})$$

where $\mathrm{vec}(\boldsymbol{\epsilon})$ strings $\boldsymbol{\epsilon}$ out by rows, $\otimes$ denotes Kronecker product multiplication, and $\boldsymbol{\Sigma}$ is $p \times p$. $\boldsymbol{\Sigma}$ can be estimated by

$$\mathbf{S} = \frac{\mathbf{e}'\mathbf{e}}{n-r} = \frac{(\mathbf{Y}-\mathbf{X}\mathbf{b})'(\mathbf{Y}-\mathbf{X}\mathbf{b})}{n-r}$$

where $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y}$, r is the rank of the $\mathbf{X}$ matrix, and $\mathbf{e}$ is the matrix of residuals.

If $\mathbf{S}$ is scaled to unit diagonals, the values in $\mathbf{S}$ are called *partial correlations of the Ys adjusting for the Xs*. This matrix can be displayed by PROC GLM if PRINTE is specified as a MANOVA option.

The multivariate general linear hypothesis is written

$$\mathbf{L}\boldsymbol{\beta}\mathbf{M} = 0$$

You can form hypotheses for linear combinations across columns, as well as across rows of $\boldsymbol{\beta}$.

The MANOVA statement of the GLM procedure tests special cases where $\mathbf{L}$ corresponds to Type I, Type II, Type III, or Type IV tests, and $\mathbf{M}$ is the $p \times p$ identity matrix. These tests are joint tests that the given type of hypothesis holds for all dependent variables in the model, and they are often sufficient to test all hypotheses of interest.

Finally, when these special cases are not appropriate, you can specify your own $\mathbf{L}$ and $\mathbf{M}$ matrices by using the CONTRAST statement before the MANOVA statement and the M= specification in the MANOVA statement, respectively. Another alternative is to use a REPEATED statement, which automatically generates a variety of $\mathbf{M}$ matrices useful in repeated measures analysis of variance. See the section "REPEATED Statement" on page 4200 and the section "Repeated Measures Analysis of Variance" on page 4249 for more information.

One useful way to think of a MANOVA analysis with an $\mathbf{M}$ matrix other than the identity is as an analysis of a set of transformed variables defined by the columns of the $\mathbf{M}$ matrix. You should note, however, that PROC GLM always displays the $\mathbf{M}$ matrix in such a way that the transformed variables are defined by the rows, not the columns, of the displayed $\mathbf{M}$ matrix.

All multivariate tests carried out by the GLM procedure first construct the matrices $\mathbf{H}$ and $\mathbf{E}$ corresponding to the numerator and denominator, respectively, of a univariate F test:

$$\begin{aligned}
\mathbf{H} &= \mathbf{M}'(\mathbf{L}\mathbf{b})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{L}\mathbf{b})\mathbf{M} \\
\mathbf{E} &= \mathbf{M}'(\mathbf{Y}'\mathbf{Y} - \mathbf{b}'(\mathbf{X}'\mathbf{X})\mathbf{b})\mathbf{M}
\end{aligned}$$

The diagonal elements of $\mathbf{H}$ and $\mathbf{E}$ correspond to the hypothesis and error SS for univariate tests. When the $\mathbf{M}$ matrix is the identity matrix (the default), these tests are for the original dependent variables on the left side of the MODEL statement. When an $\mathbf{M}$ matrix other than the identity is specified, the tests are for transformed variables defined by the columns of the $\mathbf{M}$ matrix. These tests can be studied by requesting the SUMMARY option, which produces univariate analyses for each original or transformed variable.

Four multivariate test statistics, all functions of the eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$ (or $(\mathbf{E}+\mathbf{H})^{-1}\mathbf{H}$), are constructed:

- Wilks' lambda = $\det(\mathbf{E})/\det(\mathbf{H}+\mathbf{E})$
- Pillai's trace = $\mathrm{trace}(\mathbf{H}(\mathbf{H}+\mathbf{E})^{-1})$
- Hotelling-Lawley trace = $\mathrm{trace}(\mathbf{E}^{-1}\mathbf{H})$
- Roy's greatest root = λ, largest eigenvalue of $\mathbf{E}^{-1}\mathbf{H}$

By default, all four are reported with p-values based on F approximations, as discussed in the "Multivariate Tests" section in Chapter 4, "Introduction to Regression Procedures." Alternatively, if you specify

MSTAT=EXACT in the associated MANOVA or REPEATED statement, *p*-values for three of the four tests are computed exactly (Wilks' lambda, the Hotelling-Lawley trace, and Roy's greatest root), and the *p*-values for the fourth (Pillai's trace) are based on an *F* approximation that is more accurate than the default. See the "Multivariate Tests" section in Chapter 4, "Introduction to Regression Procedures," for more details on the exact calculations.

Repeated Measures Analysis of Variance

When several measurements are taken on the same experimental unit (person, plant, machine, and so on), the measurements tend to be correlated with each other. When the measurements represent qualitatively different things, such as weight, length, and width, this correlation is best taken into account by use of multivariate methods, such as multivariate analysis of variance. When the measurements can be thought of as responses to levels of an experimental factor of interest, such as time, treatment, or dose, the correlation can be taken into account by performing a repeated measures analysis of variance.

PROC GLM provides both univariate and multivariate tests for repeated measures for one response. For an overall reference on univariate repeated measures, see Winer (1971). The multivariate approach is covered in Cole and Grizzle (1966). For a discussion of the relative merits of the two approaches, see LaTour and Miniard (1983).

Another approach to analysis of repeated measures is via general mixed models. This approach can handle balanced as well as unbalanced or missing within-subject data, and it offers more options for modeling the within-subject covariance. The main drawback of the mixed models approach is that it generally requires iteration and, thus, might be less computationally efficient. For further details on this approach, see Chapter 84, "The MIXED Procedure," and Wolfinger and Chang (1995).

Organization of Data for Repeated Measure Analysis

In order to deal efficiently with the correlation of repeated measures, the GLM procedure uses the multivariate method of specifying the model, even if only a univariate analysis is desired. In some cases, data might already be entered in the univariate mode, with each repeated measure listed as a separate observation along with a variable that represents the experimental unit (subject) on which measurement is taken. Consider the following data set Old:

```
data Old;
   input Subject Group Time y;
   datalines;
 1 1 1 15
 1 1 2 19
 1 1 3 25
 2 1 1 21
 2 1 2 18
 2 1 3 17
 1 2 1 14
 1 2 2 12
 1 2 3 16
 2 2 1 11
 2 2 2 20
 2 2 3 21
```

```
   ... more lines ...

10 3 1 14
10 3 2 18
10 3 3 16
;
```

There are three observations for each subject, corresponding to measurements taken at times 1, 2, and 3. These data could be analyzed using the following statements:

```
proc glm data=Old;
   class Group Subject Time;
   model y=Group Subject(Group) Time Group*Time;
   test h=Group e=Subject(Group);
run;
```

However, this analysis assumes subjects' measurements are uncorrelated across time. A repeated measures analysis does not make this assumption. It uses the following data set New:

```
data New;
   input Group y1 y2 y3;
   datalines;
1  15 19 25
1  21 18 17
2  14 12 16
2  11 20 21
2  24 15 12

   ... more lines ...

3  14 18 16
;
```

In the data set New, the three measurements for a subject are all in one observation. For example, the measurements for subject 1 for times 1, 2, and 3 are 15, 19, and 25, respectively. For these data, the statements for a repeated measures analysis (assuming default options) are

```
proc glm data=New;
   class Group;
   model y1-y3 = Group / nouni;
   repeated Time;
run;
```

To convert the univariate form of repeated measures data to the multivariate form, you can use a program like the following:

```
proc sort data=Old;
   by Group Subject;
run;

data New(keep=y1-y3 Group);
   array yy(3)  y1-y3;
   do Time = 1 to 3;
      set Old;
      by Group Subject;
```

```
      yy(Time) = y;
      if last.Subject then return;
   end;
run;
```

Alternatively, you could use PROC TRANSPOSE to achieve the same results with a program like this one:

```
proc sort data=Old;
   by Group Subject;
run;

proc transpose out=New(rename=(_1=y1 _2=y2 _3=y3));
   by Group Subject;
   id Time;
run;
```

See the discussions in *SAS Programmers Guide: Essentials* for more information about rearrangement of data sets.

Hypothesis Testing in Repeated Measures Analysis

In repeated measures analysis of variance, the effects of interest are as follows:

- between-subject effects (such as GROUP in the previous example)
- within-subject effects (such as TIME in the previous example)
- interactions between the two types of effects (such as GROUP*TIME in the previous example)

Repeated measures analyses are distinguished from MANOVA because of interest in testing hypotheses about the within-subject effects and the within-subject-by-between-subject interactions.

For tests that involve only between-subjects effects, both the multivariate and univariate approaches give rise to the same tests. These tests are provided for all effects in the MODEL statement, as well as for any CONTRASTs specified. The ANOVA table for these tests is labeled "Tests of Hypotheses for Between Subjects Effects" in the PROC GLM results. These tests are constructed by first adding together the dependent variables in the model. Then an analysis of variance is performed on the sum divided by the square root of the number of dependent variables. For example, the statements

```
model y1-y3=group;
repeated time;
```

give a one-way analysis of variance that uses $(Y1 + Y2 + Y3)/\sqrt{3}$ as the dependent variable for performing tests of hypothesis on the between-subject effect GROUP. Tests for between-subject effects are equivalent to tests of the hypothesis $\mathbf{L}\boldsymbol{\beta}\mathbf{M} = 0$, where $\mathbf{M}$ is simply a vector of 1s.

For within-subject effects and for within-subject-by-between-subject interaction effects, the univariate and multivariate approaches yield different tests. These tests are provided for the within-subject effects and for the interactions between these effects and the other effects in the MODEL statement, as well as for any CONTRASTs specified. The univariate tests are displayed in a table labeled "Univariate Tests of Hypotheses for Within Subject Effects." Results for multivariate tests are displayed in a table labeled "Repeated Measures Analysis of Variance."

The multivariate tests provided for within-subjects effects and interactions involving these effects are Wilks' lambda, Pillai's trace, Hotelling-Lawley trace, and Roy's greatest root. For further details on these four statistics, see the "Multivariate Tests" section in Chapter 4, "Introduction to Regression Procedures." As an example, the statements

```
model y1-y3=group;
repeated time;
```

produce multivariate tests for the within-subject effect TIME and the interaction TIME*GROUP.

The multivariate tests for within-subject effects are produced by testing the hypothesis $\mathbf{L}\boldsymbol{\beta}\mathbf{M} = 0$, where the $\mathbf{L}$ matrix is the usual matrix corresponding to the Type I, Type II, Type III, or Type IV hypotheses test, and the $\mathbf{M}$ matrix is one of several matrices depending on the transformation that you specify in the REPEATED statement. These multivariate tests require that the column rank of $\mathbf{M}$ be less than or equal to the number of error degrees of freedom. Besides that, the only assumption required for valid tests is that the dependent variables in the model have a multivariate normal distribution with a common covariance matrix across the between-subject effects.

The univariate tests for within-subject effects and interactions involving these effects require some assumptions for the probabilities provided by the ordinary F tests to be correct. Specifically, these tests require certain patterns of covariance matrices, known as Type H covariances (Huynh and Feldt 1970). Data with these patterns in the covariance matrices are said to satisfy the Huynh-Feldt condition. You can test this assumption (and the Huynh-Feldt condition) by applying a sphericity test (Anderson 1958) to any set of variables defined by an orthogonal contrast transformation. Such a set of variables is known as a set of orthogonal components. When you use the PRINTE option in the REPEATED statement, this sphericity test is applied both to the transformed variables defined by the REPEATED statement and to a set of orthogonal components if the specified transformation is not orthogonal. It is the test applied to the orthogonal components that is important in determining whether your data have a Type H covariance structure. When there are only two levels of the within-subject effect, there is only one transformed variable, and a sphericity test is not needed. The sphericity test is labeled "Test for Sphericity" in the output.

If your data satisfy the preceding assumptions, use the usual F tests to test univariate hypotheses for the within-subject effects and associated interactions.

If your data do not satisfy the assumption of Type H covariance, an adjustment to numerator and denominator degrees of freedom can be used. Several such adjustments, based on a degrees-of-freedom adjustment factor known as ϵ (epsilon) (Box 1954), are provided in PROC GLM. All these adjustments estimate ϵ and then multiply the numerator and denominator degrees of freedom by this estimate before determining significance levels for the F tests. Significance levels associated with the adjusted tests are labeled "Adj Pr > F" in the output. Two such adjustments are displayed. One is the maximum likelihood estimate of Box's ϵ factor, which is known to be conservative, possibly very much so. The other adjustment is intended to be unbiased although possibly at the cost of being liberal. The first adjustment is labeled as the "Greenhouse-Geisser Epsilon." It has the form

$$\hat{\epsilon}_{\mathrm{GG}} = \frac{\mathrm{trace}^2(\mathbf{E})/b}{\mathrm{trace}(\mathbf{E}^2)}$$

where $\mathbf{E}$ is the error matrix for the corresponding multivariate test and b is the degrees of freedom for the hypothesis being tested. $\hat{\epsilon}_{\mathrm{GG}}$ was initially proposed for use in data analysis by Greenhouse and Geisser (1959). Significance levels associated with F tests thus adjusted are labeled "G-G" in the output.

Huynh and Feldt (1976) showed that $\hat{\epsilon}_{\mathrm{GG}}$ tends to be biased downward (that is, conservative), especially for small samples. Alternative estimates have been proposed to overcome this conservative bias, and there are several options for which estimate to display along with $\hat{\epsilon}_{\mathrm{GG}}$.

- Huynh and Feldt (1976) proposed an estimate of Box's epsilon, constructed using estimators of its numerator and denominator that are intended to be unbiased. The Huynh-Feldt epsilon has the form of a modification of the Greenhouse-Geisser epsilon,

  $$\hat{\epsilon}_{\mathrm{HF}} = \frac{nb\hat{\epsilon}_{\mathrm{GG}} - 2}{b(\mathrm{DFE} - b\hat{\epsilon}_{\mathrm{GG}})}$$

  where n is the number of subjects and DFE is the degrees of freedom for error. The numerator of this estimate is precisely unbiased only when there are no between-subject effects, but $\hat{\epsilon}_{\mathrm{HF}}$ is still often employed even with nontrivial between-subject models; it was the only unbiased epsilon alternative in SAS/STAT releases before SAS/STAT 9.22. The Huynh-Feldt epsilon is no longer the default, but you can request it and its corresponding F test by using the UEPSDEF=HF option in the REPEATED statement. The estimate is labeled "Huynh-Feldt Epsilon" in the PROC GLM output, and the significance levels associated with adjusted F tests are labeled "H-F."

- Lecoutre (1991) gave the unbiased form of the numerator of Box's epsilon when there is one between-subject effect. The correct form of Huynh and Feldt's idea in this case is

  $$\hat{\epsilon}_{\mathrm{HFL}} = \frac{(\mathrm{DFE}+1)b\hat{\epsilon}_{\mathrm{GG}} - 2}{b(\mathrm{DFE} - b\hat{\epsilon}_{\mathrm{GG}})}$$

  More recently, Gribbin (2007) showed that $\hat{\epsilon}_{\mathrm{HFL}}$ applies to general between-subject models, and Chi et al. (2012) showed that it extends even to situations where the number of error degrees of freedom is less than the column rank of the within-subject contrast matrix. Thus, the Lecoutre correction of the Huynh-Feldt epsilon is displayed by default along with the Greenhouse-Geisser epsilon; you can also explicitly request it by using the UEPSDEF=HFL option in the REPEATED statement. The estimate is labeled "Huynh-Feldt-Lecoutre Epsilon" in the PROC GLM output, and the significance levels associated with adjusted F tests are labeled "H-F-L."

- Finally, Chi et al. (2012) suggest that Box's epsilon might be better estimated by replacing the reciprocal of an unbiased form of the denominator with an approximately unbiased form of the reciprocal itself. The resulting estimator can be written as a multiple of the corrected Huynh-Feldt epsilon $\hat{\epsilon}_{\mathrm{HFL}}$,

  $$\hat{\epsilon}_{\mathrm{CM}} = \hat{\epsilon}_{\mathrm{HFL}}(\nu_a - 2)(\nu_a - 4)/\nu_a^2$$

  where $\nu_a = (\mathrm{DFE}-1) + \mathrm{DFE}(\mathrm{DFE}-1)/2$. Simulations indicate that $\hat{\epsilon}_{\mathrm{CM}}$ does a good job of providing accurate p-values without being either too conservative or too liberal. Over a wide range of cases, it is never much worse than any other alternative epsilon and often much better. You can request that the Chi-Muller epsilon estimate and its corresponding F test be displayed by using the UEPSDEF=CM option in the REPEATED statement. The estimate is labeled "Chi-Muller Epsilon" in the PROC GLM output, and the significance levels associated with adjusted F tests are labeled "C-M."

Although ϵ must be in the range of 0 to 1, the three approximately unbiased estimators can be outside this range. When any of these estimators is greater than 1, a value of 1 is used in all calculations for

probabilities—in other words, the probabilities are not adjusted. Additionally, if $\hat{\epsilon}_{\mathrm{CM}} < 1/b$, then the degrees of freedom are adjusted by $1/b$ instead of $\hat{\epsilon}_{\mathrm{CM}}$.

In summary, if your data do not meet the assumptions, use adjusted F tests. However, when you strongly suspect that your data might not have Type H covariance, all these univariate tests should be interpreted cautiously. In such cases, you should consider using the multivariate tests instead.

The univariate sums of squares for hypotheses involving within-subject effects can be easily calculated from the **H** and **E** matrices corresponding to the multivariate tests described in the section "Multivariate Analysis of Variance" on page 4247. If the **M** matrix is orthogonal, the univariate sums of squares is calculated as the trace (sum of diagonal elements) of the appropriate **H** matrix; if it is not orthogonal, PROC GLM calculates the trace of the **H** matrix that results from an orthogonal **M** matrix transformation. The appropriate error term for the univariate F tests is constructed in a similar way from the error SSCP matrix and is labeled Error(*factorname*), where *factorname* indicates the **M** matrix that is used in the transformation.

When the design specifies more than one repeated measures factor, PROC GLM computes the **M** matrix for a given effect as the direct (Kronecker) product of the **M** matrices defined by the REPEATED statement if the factor is involved in the effect or as a vector of 1s if the factor is not involved. The test for the main effect of a repeated measures factor is constructed using an **L** matrix that corresponds to a test that the mean of the observation is zero. Thus, the main effect test for repeated measures is a test that the means of the variables defined by the **M** matrix are all equal to zero, while interactions involving repeated measures effects are tests that the between-subjects factors involved in the interaction have no effect on the means of the transformed variables defined by the **M** matrix. In addition, you can specify other **L** matrices to test hypotheses of interest by using the CONTRAST statement, since hypotheses defined by CONTRAST statements are also tested in the REPEATED analysis. To see which combinations of the original variables the transformed variables represent, you can specify the PRINTM option in the REPEATED statement. This option displays the transpose of **M**, which is labeled as M in the PROC GLM results. The tests produced are the same for any choice of transformation (**M**) matrix specified in the REPEATED statement; however, depending on the nature of the repeated measurements being studied, a particular choice of transformation matrix, coupled with the CANONICAL or SUMMARY option, can provide additional insight into the data being studied.

Transformations Used in Repeated Measures Analysis of Variance

As mentioned in the specifications of the REPEATED statement, several different **M** matrices can be generated automatically, based on the transformation that you specify in the REPEATED statement. Remember that both the univariate and multivariate tests that PROC GLM performs are unaffected by the choice of transformation; the choice of transformation is important only when you are trying to study the nature of a repeated measures effect, particularly with the CANONICAL and SUMMARY options. If one of these matrices does not meet your needs for a particular analysis, you might want to use the M= option in the MANOVA statement to perform the tests of interest.

The following sections describe the transformations available in the REPEATED statement, provide an example of the **M** matrix that is produced, and give guidelines for the use of the transformation. As in the PROC GLM output, the displayed matrix is labeled M. This is the $\mathbf{M}'$ matrix.

CONTRAST Transformation

This is the default transformation used by the REPEATED statement. It is useful when one level of the repeated measures effect can be thought of as a control level against which the others are compared. For example, if five drugs are administered to each of several animals and the first drug is a control or placebo, the statements

```
proc glm;
   model d1-d5= / nouni;
   repeated drug 5 contrast(1) / summary printm;
run;
```

produce the following **M** matrix:

$$\mathbf{M} = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

When you examine the analysis of variance tables produced by the SUMMARY option, you can tell which of the drugs differed significantly from the placebo.

POLYNOMIAL Transformation

This transformation is useful when the levels of the repeated measure represent quantitative values of a treatment, such as dose or time. If the levels are unequally spaced, *level values* can be specified in parentheses after the number of levels in the REPEATED statement. For example, if five levels of a drug corresponding to 1, 2, 5, 10, and 20 milligrams are administered to different treatment groups, represented by the variable group, the statements

```
proc glm;
   class group;
   model r1-r5=group / nouni;
   repeated dose 5 (1 2 5 10 20) polynomial / summary printm;
run;
```

produce the following **M** matrix:

$$\mathbf{M} = \begin{bmatrix} -0.4250 & -0.3606 & -0.1674 & 0.1545 & 0.7984 \\ 0.4349 & 0.2073 & -0.3252 & -0.7116 & 0.3946 \\ -0.4331 & 0.1366 & 0.7253 & -0.5108 & 0.0821 \\ 0.4926 & -0.7800 & 0.3743 & -0.0936 & 0.0066 \end{bmatrix}$$

The SUMMARY option in this example provides univariate ANOVAs for the variables defined by the rows of this **M** matrix. In this case, they represent the linear, quadratic, cubic, and quartic trends for dose and are labeled dose_1, dose_2, dose_3, and dose_4, respectively.

HELMERT Transformation

Since the Helmert transformation compares a level of a repeated measure to the mean of subsequent levels, it is useful when interest lies in the point at which responses cease to change. For example, if four levels of a repeated measures factor represent responses to treatments administered over time to males and females, the statements

```
proc glm;
   class sex;
   model resp1-resp4=sex / nouni;
   repeated trtmnt 4 helmert / canon printm;
```

```
run;
```

produce the following **M** matrix:

$$\mathbf{M} = \begin{bmatrix} 1 & -0.33333 & -0.33333 & -0.33333 \\ 0 & 1 & -0.50000 & -0.50000 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

MEAN Transformation

This transformation can be useful in the same types of situations in which the CONTRAST transformation is useful. If you substitute the following statement for the REPEATED statement shown in the CONTRAST Transformation section,

```
repeated drug 5 mean / printm;
```

the following **M** matrix is produced:

$$\mathbf{M} = \begin{bmatrix} 1 & -0.25 & -0.25 & -0.25 & -0.25 \\ -0.25 & 1 & -0.25 & -0.25 & -0.25 \\ -0.25 & -0.25 & 1 & -0.25 & -0.25 \\ -0.25 & -0.25 & -0.25 & 1 & -0.25 \end{bmatrix}$$

As with the CONTRAST transformation, if you want to omit a level other than the last, you can specify it in parentheses after the keyword MEAN in the REPEATED statement.

PROFILE Transformation

When a repeated measure represents a series of factors administered over time, but a polynomial response is unreasonable, a profile transformation might prove useful. As an example, consider a training program in which four different methods are employed to teach students at several different schools. The repeated measure is the score on tests administered after each of the methods is completed. The statements

```
proc glm;
   class school;
   model t1-t4=school / nouni;
   repeated method 4 profile / summary nom printm;
run;
```

produce the following **M** matrix:

$$\mathbf{M} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

To determine the point at which an improvement in test scores takes place, you can examine the analyses of variance for the transformed variables representing the differences between adjacent tests. These analyses are requested by the SUMMARY option in the REPEATED statement, and the variables are labeled METHOD.1, METHOD.2, and METHOD.3.

Random-Effects Analysis

When some model effects are random (that is, assumed to be sampled from a normal population of effects), you can specify these effects in the RANDOM statement in order to compute the expected values of mean squares for various model effects and contrasts and, optionally, to perform random-effects analysis of variance tests.

PROC GLM versus PROC MIXED for Random-Effects Analysis

Other SAS procedures that can be used to analyze models with random effects include the MIXED and VARCOMP procedures. Note that, for these procedures, the random-effects specification is an integral part of the model, affecting how both random and fixed effects are fit; for PROC GLM, the random effects are treated in a *post hoc* fashion after the complete fixed-effect model is fit. This distinction affects other features in the GLM procedure, such as the results of the LSMEANS and ESTIMATE statements. These features assume that all effects are fixed, so that all tests and estimability checks for these statements are based on a fixed-effects model, even when you use a RANDOM statement. Standard errors for estimates and LS-means based on the fixed-effects model might be significantly smaller than those based on a true random-effects model; in fact, some functions that are estimable under a true random-effects model might not even be estimable under the fixed-effects model. Therefore, you should use the MIXED procedure to compute tests involving these features that take the random effects into account; see Chapter 84, "The MIXED Procedure," for more information.

Note that, for balanced data, the test statistics computed when you specify the TEST option in the RANDOM statement have an exact F distribution only when the design is balanced; for unbalanced designs, the p values for the F tests are approximate. For balanced data, the values obtained by PROC GLM and PROC MIXED agree; for unbalanced data, they usually do not.

Computation of Expected Mean Squares for Random Effects

The RANDOM statement in PROC GLM declares one or more effects in the model to be random rather than fixed. By default, PROC GLM displays the coefficients of the expected mean squares for all terms in the model. In addition, when you specify the TEST option in the RANDOM statement, the procedure determines what tests are appropriate and provides F ratios and probabilities for these tests.

The expected mean squares are computed as follows. Consider the model

$$Y = X_0\boldsymbol{\beta}_0 + X_1\boldsymbol{\beta}_1 + \cdots + X_k\boldsymbol{\beta}_k + \boldsymbol{\epsilon}$$

where $\boldsymbol{\beta}_0$ represents the fixed effects and $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\epsilon}$ represent the random effects. Random effects are assumed to be normally and independently distributed. For any $\mathbf{L}$ in the row space of $\mathbf{X} = (X_0 \mid X_1 \mid X_2 \mid \cdots \mid X_k)$, the expected value of the sum of squares for $\mathbf{L}\beta$ is

$$E(\mathrm{SS}_L) = \boldsymbol{\beta}_0'\mathbf{C}_0'\mathbf{C}_0\boldsymbol{\beta}_0 + \mathrm{SSQ}(\mathbf{C}_1)\sigma_1^2 + \mathrm{SSQ}(\mathbf{C}_2)\sigma_2^2 + \cdots + \mathrm{SSQ}(\mathbf{C}_k)\sigma_k^2 + \mathrm{rank}(\mathbf{L})\sigma_\epsilon^2$$

where $\mathbf{C}$ is of the same dimensions as $\mathbf{L}$ and is partitioned as the $\mathbf{X}$ matrix. In other words,

$$\mathbf{C} = (\mathbf{C}_0 \mid \mathbf{C}_1 \mid \cdots \mid \mathbf{C}_k)$$

Furthermore, $\mathbf{C} = \mathbf{ML}$, where $\mathbf{M}$ is the inverse of the lower triangular Cholesky decomposition matrix of $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'$. $\mathrm{SSQ}(\mathbf{A})$ is defined as $\mathrm{tr}(\mathbf{A}'\mathbf{A})$.

For the model in the following MODEL statement

```
model Y=A B(A) C A*C;
random B(A);
```

with B(A) declared as random, the expected mean square of each effect is displayed as

$$\text{Var(Error)} + constant \times \text{Var(B(A))} + Q(\text{A}, \text{C}, \text{A} * \text{C})$$

If any fixed effects appear in the expected mean square of an effect, the letter Q followed by the list of fixed effects in the expected value is displayed. The actual numeric values of the quadratic form ($\mathbf{Q}$ matrix) can be displayed using the Q option.

To determine appropriate means squares for testing the effects in the model, the TEST option in the RANDOM statement performs the following steps:

1. First, it forms a matrix of coefficients of the expected mean squares of those effects that were declared to be random.

2. Next, for each effect in the model, it determines the combination of these expected mean squares that produce an expectation that includes all the terms in the expected mean square of the effect of interest except the one corresponding to the effect of interest. For example, if the expected mean square of an effect A*B is

   $$\text{Var(Error)} + 3 \times \text{Var(A)} + \text{Var(A} * \text{B)}$$

   PROC GLM determines the combination of other expected mean squares in the model that has expectation

   $$\text{Var(Error)} + 3 \times \text{Var(A)}$$

3. If the preceding criterion is met by the expected mean square of a single effect in the model (as is often the case in balanced designs), the F test is formed directly. In this case, the mean square of the effect of interest is used as the numerator, the mean square of the single effect with an expected mean square that satisfies the criterion is used as the denominator, and the degrees of freedom for the test are simply the usual model degrees of freedom.

4. When more than one mean square must be combined to achieve the appropriate expectation, an approximation is employed to determine the appropriate degrees of freedom (Satterthwaite 1946). When effects other than the effect of interest are listed after the Q in the output, tests of hypotheses involving the effect of interest are not valid unless all other fixed effects involved in it are assumed to be zero. When tests such as these are performed by using the TEST option in the RANDOM statement, a note is displayed reminding you that further assumptions are necessary for the validity of these tests. Remember that although the tests are not valid unless these assumptions are made, this does not provide a basis for these assumptions to be true. The particulars of a given experiment must be examined to determine whether the assumption is reasonable.

For further theoretical discussion, see Goodnight and Speed (1978), Milliken and Johnson (1984, Chapters 22 and 23), and Hocking (1985).

Sum-to-Zero Assumptions

The formulation and parameterization of the expected mean squares for random effects in mixed models are ongoing items of controversy in the statistical literature. Confusion arises over whether or not to assume that terms involving fixed effects sum to zero. Cornfield and Tukey (1956); Winer (1971), and others assume that they do sum to zero; Searle (1971); Hocking (1973), and others (including PROC GLM) do not.

Different assumptions about these sum-to-zero constraints can lead to different expected mean squares for certain terms, and hence to different F and p values.

For arguments in favor of not assuming that terms involving fixed effects sum to zero, see Section 9.7 of Searle (1971) and Sections 1 and 4 of McLean, Sanders, and Stroup (1991). Other references are Hartley and Searle (1969) and Searle, Casella, and McCulloch (1992).

Computing Type I, II, and IV Expected Mean Squares

When you use the RANDOM statement, by default the GLM procedure produces the Type III expected mean squares for model effects and for contrasts specified before the RANDOM statement. In order to obtain expected values for other types of mean squares, you need to specify which types of mean squares are of interest in the MODEL statement. For example, in order to obtain the Type IV expected mean squares for effects in the RANDOM and CONTRAST statements, specify the SS4 option in the MODEL statement. If you want both Type III and Type IV expected mean squares, specify both the SS3 and SS4 options in the MODEL statement. Since the estimable function basis is not automatically calculated for Type I and Type II SS, the E1 (for Type I) or E2 (for Type II) option must be specified in the MODEL statement in order for the RANDOM statement to produce the expected mean squares for the Type I or Type II sums of squares. Note that it is important to list the fixed effects first in the MODEL statement when requesting the Type I expected mean squares.

For example, suppose you have a two-way design with factors A and B in which the main effect for B and the interaction are random. In order to compute the Type III expected mean squares (in addition to the fixed-effect analysis), you can use the following statements:

```
proc glm;
   class A B;
   model Y = A B A*B;
   random B A*B;
run;
```

Suppose you use the SS4 option in the MODEL statement, as follows:

```
proc glm;
   class A B;
   model Y = A B A*B / ss4;
   random B A*B;
run;
```

Then only the Type IV expected mean squares are computed (as well as the Type IV fixed-effect tests). For the Type I expected mean squares, you can use the following statements:

```
proc glm;
   class A B;
   model Y = A B A*B / e1;
   random B A*B;
```

```
run;
```

For each of these cases, in order to perform random-effect analysis of variance tests for each effect specified in the model, you need to specify the TEST option in the RANDOM statement, as follows:

```
proc glm;
   class A B;
   model Y = A B A*B;
   random B A*B / test;
run;
```

The GLM procedure automatically determines the appropriate error term for each test, based on the expected mean squares.

Missing Values

For an analysis involving one dependent variable, PROC GLM uses an observation if values are nonmissing for that dependent variable and all the independent variables.

For an analysis involving multiple dependent variables without the MANOVA or REPEATED statement, or without the MANOVA option in the PROC GLM statement, a missing value in one dependent variable does not eliminate the observation from the analysis of other nonmissing dependent variables. On the other hand, for an analysis with the MANOVA or REPEATED statement, or with the MANOVA option in the PROC GLM statement, PROC GLM uses an observation if values are nonmissing for all dependent variables and all the variables used in independent effects.

During processing, the GLM procedure groups the dependent variables by their pattern of missing values across observations so that sums and crossproducts can be collected in the most efficient manner.

If your data have different patterns of missing values among the dependent variables, interactivity is disabled. This can occur when some of the variables in your data set have missing values and either of the following conditions obtain:

- You do not use the MANOVA option in the PROC GLM statement.
- You do not use a MANOVA or REPEATED statement before the first RUN statement.

Note that the REG procedure handles missing values differently in this case; see Chapter 105, “The REG Procedure,” for more information.

Computational Resources

Memory

For large problems, most of the memory resources are required for holding the $\mathbf{X}'\mathbf{X}$ matrix of the sums and crossproducts. The section "Parameterization of PROC GLM Models" on page 4210 describes how columns of the $\mathbf{X}$ matrix are allocated for various types of effects. For each level that occurs in the data for a combination of classification variables in a given effect, a row and a column for $\mathbf{X}'\mathbf{X}$ are needed.

The following example illustrates the calculation. Suppose A has 20 levels, B has 4 levels, and C has 3 levels. Then consider the model

```
proc glm;
   class A B C;
   model Y1 Y2 Y3=A B A*B C A*C B*C A*B*C X1 X2;
run;
```

The $\mathbf{X}'\mathbf{X}$ matrix (bordered by $\mathbf{X}'\mathbf{Y}$ and $\mathbf{Y}'\mathbf{Y}$) can have as many as 425 rows and columns:

1	for the intercept term
20	for A
4	for B
80	for A ∗ B
3	for C
60	for A ∗ C
12	for B ∗ C
240	for A ∗ B ∗ C
2	for X1 and X2 (continuous variables)
3	for Y1, Y2, and Y3 (dependent variables)

The matrix has 425 rows and columns only if all combinations of levels occur for each effect in the model. For m rows and columns, $8m^2$ bytes are needed for crossproducts. In this case, $8 \cdot 425^2 = 1{,}445{,}000$ bytes, or about $1{,}445{,}000/1{,}024\ 1.4M$.

The required memory grows as the square of the number of columns of $\mathbf{X}$; most of the memory is for the A*B*C interaction. Without A*B*C, you have 185 columns and need 268K for $\mathbf{X}'\mathbf{X}$. Without either A*B*C or A*B, you need 86K. If A is recoded to have 10 levels, then the full model has only 220 columns and requires 378K.

The second time that a large amount of memory is needed is when Type III, Type IV, or contrast sums of squares are being calculated. This memory requirement is a function of the number of degrees of freedom of the model being analyzed and the maximum degrees of freedom for any single source. Let Rank equal the sum of the model degrees of freedom, MaxDF be the maximum number of degrees of freedom for any single source, and N_y be the number of dependent variables in the model. Then the memory requirement in bytes is

8 times

$$
\begin{aligned}
N_y \times \text{Rank} \quad &+ \quad (\text{Rank} \times (\text{Rank} + 1))\,/2 \\
&+ \quad \text{MaxDF} \times \text{Rank} \\
&+ \quad (\text{MaxDF} \times (\text{MaxDF} + 1))\,/2 \\
&+ \quad \text{MaxDF} \times N_y
\end{aligned}
$$

The first two components of this formula are for the estimable model coefficients and their variance; the rest correspond to $\mathbf{L}$, $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'$, and $\mathbf{Lb}$ in the computation of $\text{SS}(\mathbf{L}\boldsymbol{\beta} = 0) = (\mathbf{Lb})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{Lb})$. If the operating system enables SAS to run parallel computational threads on multiple CPUs, then GLM will attempt to allocate another $8 \times \text{Rank} \times \text{Rank}$ bytes in order to perform these calculations in parallel. If this much memory is not available, then the estimability calculations are performed in a single thread.

Unfortunately, these quantities are not available when the $\mathbf{X}'\mathbf{X}$ matrix is being constructed, so PROC GLM might occasionally request additional memory even after you have increased the memory allocation available to the program.

If you have a large model that exceeds the memory capacity of your computer, these are your options:

- Eliminate terms, especially high-level interactions.
- Reduce the number of levels for variables with many levels.
- Use the ABSORB statement for parts of the model that are large.
- Use the REPEATED statement for repeated measures variables.
- Use PROC ANOVA or PROC REG rather than PROC GLM, if your design allows.

A related limitation is that for any model effect involving classification variables (interactions as well as main effects), the number of levels cannot exceed 32,767. This is because GLM internally indexes effect levels with signed short (16-bit) integers, for which the maximum value is $2^{15} - 1 = 32{,}767$.

CPU Time

Typically, if the GLM procedure requires a lot of CPU time, it will be for one of several reasons. Suppose that the input data has n rows (observations) and the model has E effects that together produce a design matrix $\mathbf{X}$ with m columns. Then if m or n is relatively large, the procedure might spend a lot of time in any of the following areas:

- collecting the sums of squares and crossproducts
- solving the normal equations
- computing the Type III tests

The time required for collecting sums and crossproducts is difficult to calculate because it is a complicated function of the model. The worst case occurs if all columns are continuous variables, involving $nm^2/2$ multiplications and additions. If the columns are levels of a classification, then only m sums might be needed, but a significant amount of time might be spent in look-up operations. Solving the normal equations requires time for approximately $m^3/2$ multiplications and additions, and the number of operations required to compute the Type III tests is also proportional to both E and m^3.

Suppose that you know that Type IV sums of squares are appropriate for the model you are analyzing (for example, if your design has no missing cells). You can specify the SS4 option in your MODEL statement, which saves CPU time by requesting the Type IV sums of squares instead of the more computationally burdensome Type III sums of squares. This proves especially useful if you have a factor in your model that has many levels and is involved in several interactions.

If the operating system enables SAS to run parallel computational threads on multiple CPUs, then both the solution of the normal equations and the computation of Type III tests can take advantage of this to reduce the computational time for large models. In solving the normal equations, the fundamental row sweep operations (Goodnight 1979) are performed in parallel. In computing the Type III tests, both the orthogonalization for the estimable functions and the sums of squares calculation have been parallelized (if there is sufficient memory).

The reduction in computational time due to parallel processing depends on the size of the model, the number of processors, and the parallel architecture of the operating system. If the model is large enough that the overwhelming proportion of CPU time for the procedure is accounted for in solving the normal equations and/or computing the Type III tests, then you can expect a reduction in computational time approximately inversely proportional to the number of CPUs. However, as you increase the number of processors, the efficiency of this scaling can be reduced by several effects. One mitigating factor is a purely mathematical one known as "Amdahl's law," which is related to the fact that only part of the processing time for the procedure can be parallelized. Even taking Amdahl's law into account, the parallelization efficiency can be reduced by cache effects related to how fast the multiple processors can access memory. See Cohen (2002) for a discussion of these issues. For additional information about parallel processing in SAS, see the chapter on "Support for Parallel Processing" in *SAS Programmers Guide: Essentials*.

Computational Method

Let $\mathbf{X}$ represent the $n \times p$ design matrix and $\mathbf{Y}$ the $n \times 1$ vector of dependent variables. (See the section "Parameterization of PROC GLM Models" on page 4210 for information about how $\mathbf{X}$ is formed from your model specification.)

The normal equations $\mathbf{X}'\mathbf{X}\boldsymbol{\beta} = \mathbf{X}'\mathbf{Y}$ are solved using a modified sweep routine that produces a generalized inverse $(\mathbf{X}'\mathbf{X})^-$ and a solution $\mathbf{b} = (\mathbf{X}'\mathbf{X})^-\mathbf{X}'\mathbf{y}$. The modification is that rows and columns corresponding to diagonal elements that are found during sweeping to be zero (or within the expected level of numerical error of zero) are zeroed out. The $(\mathbf{X}'\mathbf{X})^-$ produced by this procedure satisfies the following two equations:

$$
\begin{aligned}
(\mathbf{X}'\mathbf{X})\ \ (\mathbf{X}'\mathbf{X})^-(\mathbf{X}'\mathbf{X}) &= (\mathbf{X}'\mathbf{X}) \\
(\mathbf{X}'\mathbf{X})^-(\mathbf{X}'\mathbf{X})\ \ (\mathbf{X}'\mathbf{X})^- &= (\mathbf{X}'\mathbf{X})^-
\end{aligned}
$$

Pringle and Rayner (1971) call a generalized inverse with these characteristics a g_2-inverse, and this is the term usually used in SAS documentation and output. Urquhart (1968) uses the term *reflexive g-inverse* to

emphasize that $(\mathbf{X}'\mathbf{X})^-$ is a generalized inverse of $\mathbf{X}'\mathbf{X}$ in the same way that $\mathbf{X}'\mathbf{X}$ is a generalized inverse of $(\mathbf{X}'\mathbf{X})^-$. Note that a g_2-inverse is not necessarily unique: if $\mathbf{X}'\mathbf{X}$ is singular, then sweeping the matrix in a different order will result in a different g_2-inverse that also satisfies the two preceding equations.

For each effect in the model, a matrix $\mathbf{L}$ is computed such that the rows of $\mathbf{L}$ are estimable. Tests of the hypothesis $\mathbf{L}\boldsymbol{\beta} = 0$ are then made by first computing

$$\mathrm{SS}(\mathbf{L}\boldsymbol{\beta} = 0) = (\mathbf{L}\mathbf{b})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^-\mathbf{L}')^{-1}(\mathbf{L}\mathbf{b})$$

and then computing the associated F value by using the mean squared error.

Output Data Sets

OUT= Data Set Created by the OUTPUT Statement

The OUTPUT statement produces an output data set that contains the following:

- all original data from the SAS data set input to PROC GLM
- the new variables corresponding to the diagnostic measures specified with statistics keywords in the OUTPUT statement (PREDICTED=, RESIDUAL=, and so on)

With multiple dependent variables, a name can be specified for any of the diagnostic measures for each of the dependent variables in the order in which they occur in the MODEL statement.

For example, suppose that the input data set A contains the variables y1, y2, y3, x1, and x2. Then you can use the following statements:

```
proc glm data=A;
   model y1 y2 y3=x1;
   output out=out p=y1hat y2hat y3hat
                  r=y1resid lclm=y1lcl uclm=y1ucl;
run;
```

The output data set out contains y1, y2, y3, x1, x2, y1hat, y2hat, y3hat, y1resid, y1lcl, and y1ucl. The variable x2 is output even though it is not used by PROC GLM. Although predicted values are generated for all three dependent variables, residuals are output for only the first dependent variable.

When any independent variable in the analysis (including all class variables) is missing for an observation, then all new variables that correspond to diagnostic measures are missing for the observation in the output data set.

When a dependent variable in the analysis is missing for an observation, then some new variables that correspond to diagnostic measures are missing for the observation in the output data set, and some are still available. Specifically, in this case, the new variables that correspond to COOKD, COVRATIO, DFFITS, PRESS, R, RSTUDENT, STDR, and STUDENT are missing in the output data set. The variables corresponding to H, LCL, LCLM, P, STDI, STDP, UCL, and UCLM are not missing.

OUT= Data Set Created by the LSMEANS Statement

The OUT= option in the LSMEANS statement produces an output data set that contains the following:

- the unformatted values of each classification variable specified in any effect in the LSMEANS statement
- a new variable, LSMEAN, which contains the LS-mean for the specified levels of the classification variables
- a new variable, STDERR, which contains the standard error of the LS-mean

The variances and covariances among the LS-means are also output when the COV option is specified along with the OUT= option. In this case, only one effect can be specified in the LSMEANS statement, and the following variables are included in the output data set:

- new variables, COV1, COV2, . . . , COV*n*, where *n* is the number of levels of the effect specified in the LSMEANS statement. These variables contain the covariances of each LS-mean with every other LS-mean.
- a new variable, NUMBER, which provides an index for each observation to identify the covariances that correspond to that observation. The covariances for the observation with NUMBER equal to *n* can be found in the variable COV*n*.

OUTSTAT= Data Set

The OUTSTAT= option in the PROC GLM statement produces an output data set that contains the following:

- the BY variables, if any
- _TYPE_, a new character variable. _TYPE_ can take the values ‘SS1’, ‘SS2’, ‘SS3’, ‘SS4’, or ‘CONTRAST’, corresponding to the various types of sums of squares generated, or the values ‘CANCORR’, ‘STRUCTUR’, or ‘SCORE’, if a canonical analysis is performed through the MANOVA statement and no M= matrix is specified.
- _SOURCE_, a new character variable. For each observation in the data set, _SOURCE_ contains the name of the model effect or contrast label from which the corresponding statistics are generated.
- _NAME_, a new character variable. For each observation in the data set, _NAME_ contains the name of one of the dependent variables in the model or, in the case of canonical statistics, the name of one of the canonical variables (CAN1, CAN2, and so forth).
- four new numeric variables: SS, DF, F, and PROB, containing sums of squares, degrees of freedom, *F* values, and probabilities, respectively, for each model or contrast sum of squares generated in the analysis. For observations resulting from canonical analyses, these variables have missing values.
- if there is more than one dependent variable, then variables with the same names as the dependent variables represent the following:
  - for _TYPE_=SS1, SS2, SS3, SS4, or CONTRAST, the crossproducts of the hypothesis matrices

  - for _TYPE_=CANCORR, canonical correlations for each variable
  - for _TYPE_=STRUCTUR, coefficients of the total structure matrix
  - for _TYPE_=SCORE, raw canonical score coefficients

The output data set can be used to perform special hypothesis tests (for example, with the IML procedure in SAS/IML software), to reformat output, to produce canonical variates (through the SCORE procedure), or to rotate structure matrices (through the FACTOR procedure).

Displayed Output

The GLM procedure produces the following output by default:

- The overall analysis-of-variance table breaks down the Total Sum of Squares for the dependent variable into the portion attributed to the Model and the portion attributed to Error.
- The Mean Square term is the Sum of Squares divided by the degrees of freedom (DF).
- The Mean Square for Error is an estimate of σ^2, the variance of the true errors.
- The F Value is the ratio produced by dividing the Mean Square for the Model by the Mean Square for Error. It tests how well the model as a whole (adjusted for the mean) accounts for the dependent variable's behavior. An F test is a joint test to determine that all parameters except the intercept are zero.
- A small significance probability, Pr > F, indicates that some linear function of the parameters is significantly different from zero.
- R-Square, R^2, measures how much variation in the dependent variable can be accounted for by the model. R square, which can range from 0 to 1, is the ratio of the sum of squares for the model to the corrected total sum of squares. In general, the larger the value of R square, the better the model's fit.
- Coeff Var, the coefficient of variation, which describes the amount of variation in the population, is 100 times the standard deviation estimate of the dependent variable, Root MSE (Mean Square for Error), divided by the Mean. The coefficient of variation is often a preferred measure because it is unitless.
- Root MSE estimates the standard deviation of the dependent variable (or equivalently, the error term) and equals the square root of the Mean Square for Error.
- Mean is the sample mean of the dependent variable.

These tests are used primarily in analysis-of-variance applications:

- The Type I SS (sum of squares) measures incremental sums of squares for the model as each variable is added.
- The Type III SS is the sum of squares for a balanced test of each effect, adjusted for every other effect.

These items are used primarily in regression applications:

- The Estimates for the model Parameters (the intercept and the coefficients)
- t Value is the Student's *t* value for testing the null hypothesis that the parameter (if it is estimable) equals zero.
- The significance level, Pr > |t|, is the probability of getting a larger value of *t* if the parameter is truly equal to zero. A very small value for this probability leads to the conclusion that the independent variable contributes significantly to the model.
- The Standard Error is the square root of the estimated variance of the estimate of the true value of the parameter.

Other portions of output are discussed in the following examples.

ODS Table Names

PROC GLM assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 53.15. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 53.15 ODS Tables Produced by PROC GLM

ODS Table Name	Description	Statement / Option
Aliasing	Type 1,2,3,4 aliasing structure	MODEL / (E1 E2 E3 or E4) and ALIASING
AltErrContrasts	ANOVA table for contrasts with alternative error	CONTRAST / E=
AltErrTests	ANOVA table for tests with alternative error	TEST / E=
Bartlett	Bartlett's homogeneity of variance test	MEANS / HOVTEST=BARTLETT
CanAnalysis	Canonical analysis	(MANOVA or REPEATED) / CANONICAL
CanCoef	Canonical coefficients	(MANOVA or REPEATED) / CANONICAL
CanStructure	Canonical structure	(MANOVA or REPEATED) / CANONICAL
CharStruct	Characteristic roots and vectors	(MANOVA / not CANONICAL) or (REPEATED / PRINTRV)
ClassLevels	Classification variable levels	CLASS statement
CLDiffs	Multiple comparisons of pairwise differences	MEANS / CLDIFF or DUNNETT or (Unequal cells and not LINES)
CLDiffsInfo	Information for multiple comparisons of pairwise differences	MEANS / CLDIFF or DUNNETT or (Unequal cells and not LINES)

Table 53.15 *continued*

ODS Table Name	Description	Statement / Option
CLMeans	Multiple comparisons of means with confidence/comparison interval	MEANS / CLM
CLMeansInfo	Information for multiple comparison of means with confidence/comparison interval	MEANS / CLM
ContrastCoef	**L** matrix for contrast or estimate	CONTRAST / E or ESTIMATE / E
Contrasts	ANOVA table for contrasts	CONTRAST statement
DependentInfo	Simultaneously analyzed dependent variables	Default when there are multiple dependent variables with different patterns of missing values
Diff	PDiff matrix of least squares means	LSMEANS / PDIFF=ALL and more than two LS-means
Epsilons	Greenhouse-Geisser and Huynh-Feldt epsilons	REPEATED statement
ErrorSSCP	Error SSCP matrix	(MANOVA or REPEATED) / PRINTE
EstFunc	Type 1,2,3,4 estimable functions	MODEL / (E1 E2 E3 or E4)
Estimates	Estimate statement results	ESTIMATE statement
ExpectedMeanSquares	Expected mean squares	RANDOM statement
FitStatistics	R-Square, Coeff Var, Root MSE, and dependent mean	Default
GAliasing	General form of aliasing structure	MODEL / E and ALIASING
GEstFunc	General form of estimable functions	MODEL / E
HOVFTest	Homogeneity of variance ANOVA	MEANS / HOVTEST
HypothesisSSCP	Hypothesis SSCP matrix	(MANOVA or REPEATED) / PRINTH
InvXPX	inv($\mathbf{X}'\mathbf{X}$) matrix	MODEL / INVERSE
LSMeanCL	Confidence interval for LS-means	LSMEANS / CL
LSMeanCoef	Coefficients of least squares means	LSMEANS / E
LSMeanDiffCL	Confidence interval for LS-mean differences	LSMEANS / PDIFF and CL
LSMeans	Least squares means	LSMEANS statement
LSMLines	Least squares means comparison lines	LSMEANS / PDIFF=ALL LINES
MANOVATransform	Multivariate transformation matrix	MANOVA / M=
MatrixRepresentation	**X** matrix element representation	As needed for other options

Table 53.15 *continued*

ODS Table Name	Description	Statement / Option
MCLines	Multiple comparisons LINES output	MEANS / LINES or ((DUNCAN or WALLER or SNK or REGWQ) and not (CLDIFF or CLM)) or (Equal cells and not CLDIFF)
MCLinesInfo	Information for multiple comparison LINES output	MEANS / LINES or ((DUNCAN or WALLER or SNK or REGWQ) and not (CLDIFF or CLM)) or (Equal cells and not CLDIFF)
MCLinesRange	Ranges for multiple range MC tests	MEANS / LINES or ((DUNCAN or WALLER or SNK or REGWQ) and not (CLDIFF or CLM)) or (Equal cells and not CLDIFF)
Means	Group means	MEANS statement
ModelANOVA	ANOVA for model terms	Default
MultStat	Multivariate tests	MANOVA statement
NObs	Number of observations	Default
OverallANOVA	Overall ANOVA	Default
OverallEffectSize	Effect size measures for overall ANOVA	MODEL / EFFECTSIZE
ParameterEstimates	Estimated linear model coefficients	MODEL / SOLUTION
PartialCorr	Partial correlation matrix	(MANOVA or REPEATED) / PRINTE
PredictedInfo	Predicted values info	MODEL / P or CLM or CLI
PredictedValues	Predicted values	MODEL / P or CLM or CLI
QForm	Quadratic form for expected mean squares	RANDOM / Q
RandomModelANOVA	Random-effect tests	RANDOM / TEST
RepeatedLevelInfo	Correspondence between dependents and repeated measures levels	REPEATED statement
RepeatedTransform	Repeated measures transformation matrix	REPEATED / PRINTM
SimDetails	Details of difference quantile simulation	LSMEANS / ADJUST=SIMULATE(REPORT)
SimResults	Evaluation of difference quantile simulation	LSMEANS / ADJUST=SIMULATE(REPORT)
SlicedANOVA	Sliced-effect ANOVA table	LSMEANS / SLICE
Sphericity	Sphericity tests	REPEATED / PRINTE
Tests	Summary ANOVA for specified MANOVA H= effects	MANOVA / H= SUMMARY
Tolerances	$\mathbf{X}'\mathbf{X}$ tolerances	MODEL / TOLERANCE
Welch	Welch's ANOVA	MEANS / WELCH
XPX	$\mathbf{X}'\mathbf{X}$ matrix	MODEL / XPX

With the PDIFF or TDIFF option in the LSMEANS statement, the *p*/*t*-values for differences are displayed in columns of the LSMeans table for PDIFF/TDIFF=CONTROL or PDIFF/TDIFF=ANOM, and for PDIFF/TDIFF=ALL when there are only two LS-means. Otherwise (for PDIFF/TDIFF=ALL when there are more than two LS-means), the *p*/*t*-values for differences are displayed in a separate table called Diff.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, then for particular models the GLM procedure produces default graphics as follows:

- If you specify a one-way analysis of variance model, with just one CLASS variable, the GLM procedure produces a grouped box plot of the response values versus the CLASS levels. For an example of the box plot, see the section "One-Way Layout with Means Comparisons" on page 1119 in Chapter 29, "The ANOVA Procedure." Outliers are labeled by the observation number within the entire data set, including observations that are not used in the analysis. Therefore, the labels of the outliers in this box plot might differ from those in the box plot that is produced by the MEANS statement.

- If you specify a two-way analysis of variance model, with just two CLASS variables, the GLM procedure produces an interaction plot of the response values, with horizontal position representing one CLASS variable and marker style representing the other, and with predicted response values connected by lines that represent the two-way analysis. For an example of the interaction plot, see the section "PROC GLM for Unbalanced ANOVA" on page 4147.

- If you specify a model that has two CLASS variables, and one variable is nested within the other, then the GLM procedure produces a nested box plot of the response values, where horizontal position represents one CLASS variable nested within the other CLASS variable.

- If you specify a model that has a single continuous predictor, the GLM procedure produces a fit plot of the response values versus the covariate values, with a curve representing the fitted relationship. For an example of the fit plot, see the section "PROC GLM for Quadratic Least Squares Regression" on page 4150.

- If you specify a model that has a two continuous predictors and no CLASS variables, the GLM procedure produces a panel of fit plots as in the single predictor case, with a plot of the response values versus one of the covariates at each of several values of the other covariate.

- If you specify an analysis of covariance model that has one or two CLASS variables and one continuous variable, the GLM procedure produces an analysis-of-covariance plot of the response values versus the

covariate values, with lines representing the fitted relationship within each classification level. For an example of the analysis of covariance plot, see Example 53.4.

- If you specify an LSMEANS statement with the PDIFF option, the GLM procedure produces a plot appropriate for the type of LS-means comparison. For PDIFF=ALL (which is the default if you specify only PDIFF), the procedure produces a diffogram, which displays all pairwise LS-means differences and their significance. The display is also known as a "mean-mean scatter plot" (Hsu 1996). For PDIFF=CONTROL, the procedure produces a display of each noncontrol LS-mean compared to the control LS-mean, with two-sided confidence intervals for the comparison. For PDIFF=CONTROLL and PDIFF=CONTROLU, a similar display is produced, but with one-sided confidence intervals. Finally, for the PDIFF=ANOM option, the procedure produces an analysis-of-means plot, which compares each LS-mean to the average LS-mean.
- If you specify a MEANS statement, the GLM procedure produces a grouped box plot of the response values versus the effect for which means are being calculated. Outliers are labeled by the observation number within only the data that have no missing values, thus excluding observations that are not used in the analysis. Therefore, the labels of the outliers in this plot might differ from the labels of outliers in the box plot that is the default for one-way models.

In addition to the default graphics mentioned previously, you can request plots that help you diagnose the quality of the fitted model.

- The PLOTS=DIAGNOSTICS option in the PROC GLM statement requests that a panel of summary diagnostics for the fit be displayed. The panel displays scatter plots of residuals, absolute residuals, studentized residuals, and observed responses by predicted values; studentized residuals by leverage; Cook's D by observation; a Q-Q plot of residuals; a residual histogram; and a residual-fit spread plot.
- The PLOTS=RESIDUALS option in the PROC GLM statement requests scatter plots of the residuals against each continuous covariate.

ODS Graph Names

PROC GLM assigns a name to each graph that it creates using ODS. You can use these names to refer to the graphs when you use ODS. The names are listed in Table 53.16.

ODS Graphics must be enabled before requesting plots. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Table 53.16 Graphs Produced by PROC GLM

ODS Graph Name	Plot Description	Option
ANCOVAPlot	Analysis-of-covariance plot	Analysis-of-covariance model
ANOMPlot	Plot of LS-mean differences against average LS-mean	LSMEANS / PDIFF=ANOM
BoxPlot	Box plot of group means	One-way ANOVA model or MEANS statement
ContourFit	Plot of predicted response surface	Two-predictor response surface model

Table 53.16 *continued*

ODS Graph Name	Plot Description	Option
ControlPlot	Plot of LS-mean differences against a control level	LSMEANS / PDIFF=CONTROL
DiagnosticsPanel	Panel of summary diagnostics for the fit	PLOTS=DIAGNOSTICS
CooksDPlot	Cook's *D* plot	PLOTS=DIAGNOSTICS(UNPACK)
ObservedByPredicted	Observed by predicted	PLOTS=DIAGNOSTICS(UNPACK)
QQPlot	Residual Q-Q plot	PLOTS=DIAGNOSTICS(UNPACK)
ResidualByPredicted	Residual by predicted values	PLOTS=DIAGNOSTICS(UNPACK)
ResidualHistogram	Residual histogram	PLOTS=DIAGNOSTICS(UNPACK)
RFPlot	RF plot	PLOTS=DIAGNOSTICS(UNPACK)
RStudentByPredicted	Studentized residuals by predicted	PLOTS=DIAGNOSTICS(UNPACK)
RStudentByLeverage	RStudent by hat diagonals	PLOTS=DIAGNOSTICS(UNPACK)
DiffPlot	Plot of LS-mean pairwise differences	LSMEANS / PDIFF
FitPlot	Plot of predicted response by predictor	Model with one continuous predictor
IntPlot	Interaction plot	Two-way ANOVA model
LinesPlot	Plot of mean or LS-mean comparison lines	Lines comparison options in LSMEANS or MEANS
NestPlot	Nested box plot	Model in which one classification predictor is nested within the other
ResidualPlots	Plots of the residuals against each continuous covariate	PLOTS=RESIDUALS

Examples: GLM Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 53.1: Randomized Complete Blocks with Means Comparisons and Contrasts

(View the complete code for this example (glmex1.sas) in the example repository.)

This example, reported by Stenstrom (1940), analyzes an experiment to investigate how snapdragons grow in various soils. To eliminate the effect of local fertility variations, the experiment is run in blocks, with each soil type sampled in each block. Since these data are balanced, the Type I and Type III SS are the same and are equal to the traditional ANOVA SS.

First, the standard analysis is shown, followed by an analysis that uses the SOLUTION option and includes MEANS and CONTRAST statements. The ORDER=DATA option in the second PROC GLM statement is used so that the ordering of coefficients in the CONTRAST statement can correspond to the ordering in the input data. The SOLUTION option requests a display of the parameter estimates, which are produced by default only if there are no CLASS variables. A MEANS statement is used to request a table of the means with two multiple-comparison procedures requested. In experiments with focused treatment questions, CONTRAST statements are preferable to general means comparison methods. The following statements produce Output 53.1.1 through Output 53.1.4.

```
title 'Balanced Data from Randomized Complete Block';
data plants;
   input Type $ @;
   do Block = 1 to 3;
      input StemLength @;
      output;
   end;
   datalines;
Clarion  32.7 32.3 31.5
Clinton  32.1 29.7 29.1
Knox     35.7 35.9 33.1
O'Neill  36.0 34.2 31.2
Compost  31.8 28.0 29.2
Wabash   38.2 37.8 31.9
Webster  32.5 31.1 29.7
;

proc glm;
   class Block Type;
   model StemLength = Block Type;
run;

proc glm order=data;
   class Block Type;
   model StemLength = Block Type / solution;

   /*----------------------------------clrn-cltn-knox-onel-cpst-wbsh-wstr */
   contrast 'Compost vs. others'  Type   -1   -1   -1   -1    6   -1   -1;
   contrast 'River soils vs. non' Type   -1   -1   -1   -1    0    5   -1,
                                  Type   -1    4   -1   -1    0    0   -1;
   contrast 'Glacial vs. drift'   Type   -1    0    1    1    0    0   -1;
   contrast 'Clarion vs. Webster' Type   -1    0    0    0    0    0    1;
   contrast "Knox vs. O'Neill"    Type    0    0    1   -1    0    0    0;
```

```
run;

   means Type / waller regwq;
run;
```

Output 53.1.1 Analysis of Variance for Randomized Complete Blocks

Balanced Data from Randomized Complete Block

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
Block	3	1 2 3
Type	7	Clarion Clinton Compost Knox O'Neill Wabash Webster

Number of Observations Read	21
Number of Observations Used	21

Balanced Data from Randomized Complete Block

The GLM Procedure

Dependent Variable: StemLength

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	8	142.1885714	17.7735714	10.80	0.0002
Error	12	19.7428571	1.6452381		
Corrected Total	20	161.9314286			

R-Square	Coeff Var	Root MSE	StemLength Mean
0.878079	3.939745	1.282668	32.55714

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Block	2	39.0371429	19.5185714	11.86	0.0014
Type	6	103.1514286	17.1919048	10.45	0.0004

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Block	2	39.0371429	19.5185714	11.86	0.0014
Type	6	103.1514286	17.1919048	10.45	0.0004

This analysis shows that the stem length is significantly different for the different soil types. In addition, there are significant differences in stem length among the three blocks in the experiment.

The GLM procedure is invoked again, this time with the ORDER=DATA option. This enables you to write accurate contrast statements more easily because you know the order SAS is using for the levels of the variable Type. The standard analysis is displayed again, this time including the tests for contrasts that you specified as well as the estimated parameters. These additional results are shown in Output 53.1.2.

Output 53.1.2 Contrasts and Solutions

Balanced Data from Randomized Complete Block

The GLM Procedure

Dependent Variable: StemLength

Contrast	DF	Contrast SS	Mean Square	F Value	Pr > F
Compost vs. others	1	29.24198413	29.24198413	17.77	0.0012
River soils vs. non	2	48.24694444	24.12347222	14.66	0.0006
Glacial vs. drift	1	22.14083333	22.14083333	13.46	0.0032
Clarion vs. Webster	1	1.70666667	1.70666667	1.04	0.3285
Knox vs. O'Neill	1	1.81500000	1.81500000	1.10	0.3143

Parameter		Estimate		Standard Error	t Value	Pr > |t|
Intercept		29.35714286	B	0.83970354	34.96	<.0001
Block	**1**	3.32857143	B	0.68561507	4.85	0.0004
Block	**2**	1.90000000	B	0.68561507	2.77	0.0169
Block	**3**	0.00000000	B	.	.	.
Type	**Clarion**	1.06666667	B	1.04729432	1.02	0.3285
Type	**Clinton**	-0.80000000	B	1.04729432	-0.76	0.4597
Type	**Knox**	3.80000000	B	1.04729432	3.63	0.0035
Type	**O'Neill**	2.70000000	B	1.04729432	2.58	0.0242
Type	**Compost**	-1.43333333	B	1.04729432	-1.37	0.1962
Type	**Wabash**	4.86666667	B	1.04729432	4.65	0.0006
Type	**Webster**	0.00000000	B	.	.	.

Note: The X'X matrix has been found to be singular, and a generalized inverse was used to solve the normal equations. Terms whose estimates are followed by the letter 'B' are not uniquely estimable.

The contrast label, degrees of freedom, sum of squares, Mean Square, F Value, and Pr > F are shown for each contrast requested. In this example, the contrast results indicate the following inferences, at the 5% significance level:

- The stem length of plants grown in compost soil is significantly different from the average stem length of plants grown in other soils.
- The stem length of plants grown in river soils is significantly different from the average stem length of those grown in nonriver soils.
- The average stem length of plants grown in glacial soils (Clarion and Webster types) is significantly different from the average stem length of those grown in drift soils (Knox and O'Neill types).
- Stem lengths for Clarion and Webster types are not significantly different.
- Stem lengths for Knox and O'Neill types are not significantly different.

In addition to the estimates for the parameters of the model, the results of *t* tests about the parameters are also displayed. The 'B' following the parameter estimates indicates that the estimates are biased and do not represent a unique solution to the normal equations.

Output 53.1.3 Waller-Duncan tests

Balanced Data from Randomized Complete Block

The GLM Procedure

Waller-Duncan K-ratio t Test for StemLength

Note: This test minimizes the Bayes risk under additive loss and certain other assumptions.

Kratio	100
Error Degrees of Freedom	12
Error Mean Square	1.645238
F Value	10.45
Critical Value of t	2.12034
Minimum Significant Difference	2.2206

StemLength Waller Grouping for Means of Type (Alpha = 0.05)

Means covered by the same bar are not significantly different.

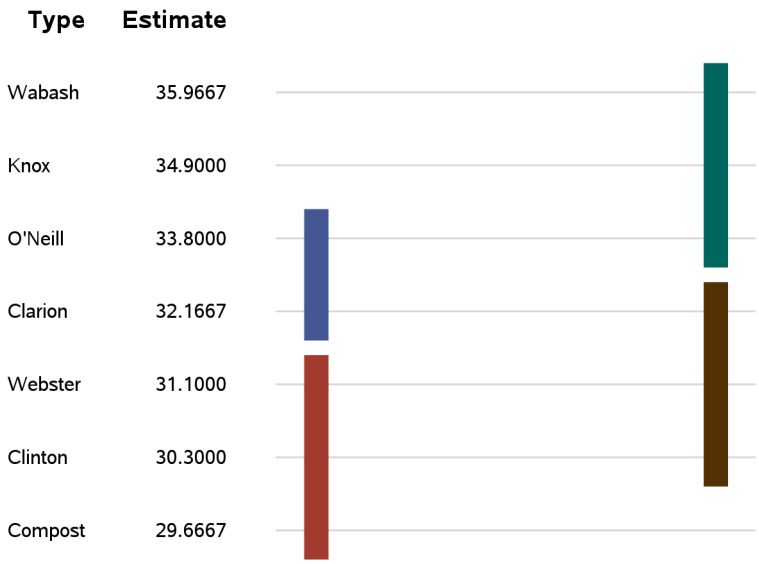

Output 53.1.4 Ryan-Einot-Gabriel-Welsch Multiple Range Test

Balanced Data from Randomized Complete Block

The GLM Procedure

Ryan-Einot-Gabriel-Welsch Multiple Range Test for StemLength

Note: This test controls the Type I experimentwise error rate.

Alpha	0.05
Error Degrees of Freedom	12
Error Mean Square	1.645238

Number of Means	**2**	**3**	**4**	**5**	**6**	**7**
Critical Range	2.9875528	3.2837322	3.4395625	3.5402383	3.5402383	3.6653133

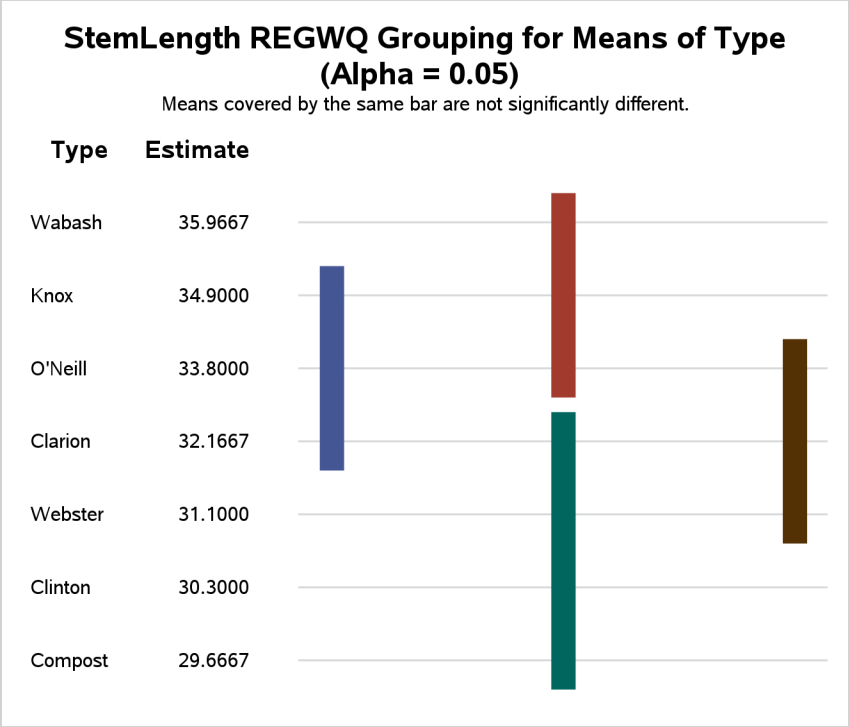

The final two pages of output (Output 53.1.3 and Output 53.1.4) present results of the Waller-Duncan and REGWQ multiple-comparison procedures. For each test, notes and information pertinent to the test are given in the output. In the plot of the comparisons, the Type means are arranged from highest to lowest. Means covered by the same bar are not significantly different. For this example, while some pairs of means are significantly different, there are no clear equivalence classes among the different soils.

For an alternative method of analyzing and displaying mean differences, see Example 53.3.

Example 53.2: Regression with Mileage Data

(View the complete code for this example (glmex2.sas) in the example repository.)

A car is tested for gas mileage at various speeds to determine at what speed the car achieves the highest gas mileage. A quadratic model is fit to the experimental data. The following statements produce Output 53.2.1 through Output 53.2.4.

```
title 'Gasoline Mileage Experiment';
data mileage;
   input mph mpg @@;
   datalines;
20 15.4
30 20.2
40 25.7
50 26.2  50 26.6  50 27.4
55   .
60 24.8
;

ods graphics on;
proc glm;
   model mpg=mph mph*mph / p clm;
run;
ods graphics off;
```

Output 53.2.1 Standard Regression Analysis

Gasoline Mileage Experiment

The GLM Procedure

Number of Observations Read	8
Number of Observations Used	7

Gasoline Mileage Experiment

The GLM Procedure

Dependent Variable: mpg

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	111.8086183	55.9043091	77.96	0.0006
Error	4	2.8685246	0.7171311		
Corrected Total	6	114.6771429			

R-Square	Coeff Var	Root MSE	mpg Mean
0.974986	3.564553	0.846836	23.75714

Source	DF	Type I SS	Mean Square	F Value	Pr > F
mph	1	85.64464286	85.64464286	119.43	0.0004
mph*mph	1	26.16397541	26.16397541	36.48	0.0038

Output 53.2.1 *continued*

Source	DF	Type III SS	Mean Square	F Value	Pr > F
mph	1	41.01171219	41.01171219	57.19	0.0016
mph*mph	1	26.16397541	26.16397541	36.48	0.0038

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	-5.985245902	3.18522249	-1.88	0.1334
mph	1.305245902	0.17259876	7.56	0.0016
mph*mph	-0.013098361	0.00216852	-6.04	0.0038

The overall F statistic is significant. The tests of mph and mph*mph in the Type I sums of squares show that both the linear and quadratic terms in the regression model are significant. The model fits well, with an R square of 0.97. The table of parameter estimates indicates that the estimated regression equation is

$$\text{mpg} = -5.9852 + 1.3052 \times \text{mph} - 0.0131 \times \text{mph}^2$$

Output 53.2.2 Results of Requesting the P and CLM Options

Observation	Observed	Predicted	Residual	95% Confidence Limits for Mean Predicted Value	
1	15.40000000	14.88032787	0.51967213	12.69701317	17.06364257
2	20.20000000	21.38360656	-1.18360656	20.01727192	22.74994119
3	25.70000000	25.26721311	0.43278689	23.87460041	26.65982582
4	26.20000000	26.53114754	-0.33114754	25.44573423	27.61656085
5	26.60000000	26.53114754	0.06885246	25.44573423	27.61656085
6	27.40000000	26.53114754	0.86885246	25.44573423	27.61656085
7 *	.	26.18073770	.	24.88679308	27.47468233
8	24.80000000	25.17540984	-0.37540984	23.05954977	27.29126990

The P and CLM options in the MODEL statement produce the table shown in Output 53.2.2. For each observation, the observed, predicted, and residual values are shown. In addition, the 95% confidence limits for a mean predicted value are shown for each observation. Note that the observation with a missing value for mph is not used in the analysis, but predicted and confidence limit values are shown.

Output 53.2.3 Additional Results of Requesting the P and CLM Options

Sum of Residuals	-0.00000000
Sum of Squared Residuals	2.86852459
Sum of Squared Residuals - Error SS	-0.00000000
PRESS Statistic	23.18107335
First Order Autocorrelation	-0.54376613
Durbin-Watson D	2.94425592

The last portion of the output listing, shown in Output 53.2.3, gives some additional information about the residuals. The Press statistic gives the sum of squares of predicted residual errors, as described in

Chapter 4, "Introduction to Regression Procedures." The First Order Autocorrelation and the Durbin-Watson *D* statistic, which measures first-order autocorrelation, are also given.

Output 53.2.4 Plot of Mileage Data

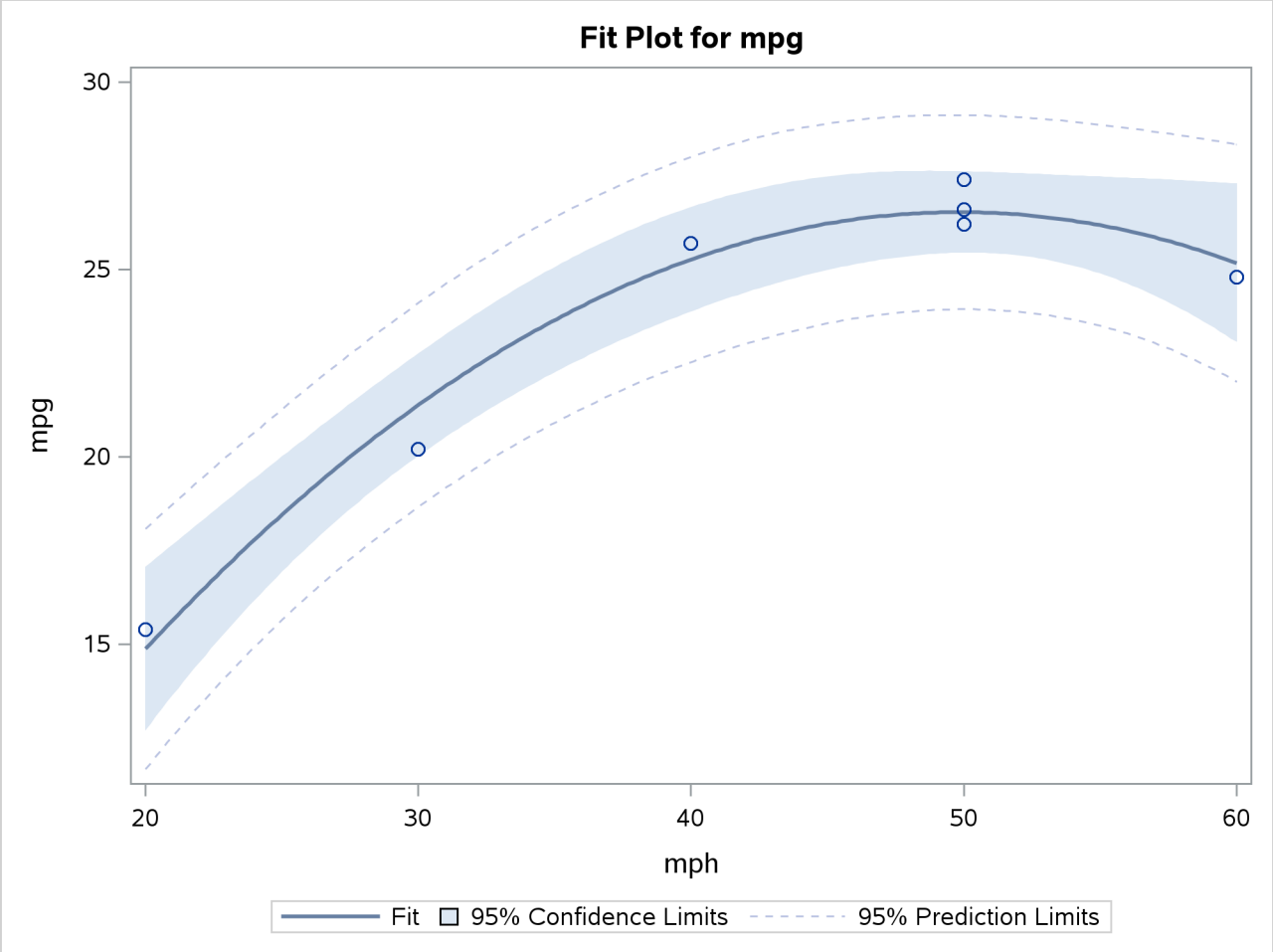

Finally, the ODS GRAPHICS ON command in the previous statements enables ODS Graphics, which in this case produces the plot shown in Output 53.2.4 of the actual and predicted values for the data, as well as a band representing the confidence limits for individual predictions. The quadratic relationship between mpg and mph is evident.

Example 53.3: Unbalanced ANOVA for Two-Way Design with Interaction

(View the complete code for this example (glmex3.sas) in the example repository.)

This example uses data from Kutner (1974, p. 98) to illustrate a two-way analysis of variance. The original data source is Afifi and Azen (1972, p. 166). These statements produce Output 53.3.1 and Output 53.3.2.

```
title 'Unbalanced Two-Way Analysis of Variance';
data a;
   input drug disease @;
   do i=1 to 6;
      input y @;
      output;
   end;
   datalines;
1 1 42 44 36 13 19 22
1 2 33  . 26  . 33 21
1 3 31 -3  . 25 25 24
2 1 28  . 23 34 42 13
2 2  . 34 33 31  . 36
2 3  3 26 28 32  4 16
3 1  .  .  1 29  . 19
3 2  . 11  9  7  1 -6
3 3 21  1  .  9  3  .
4 1 24  .  9 22 -2 15
4 2 27 12 12 -5 16 15
4 3 22  7 25  5 12  .
;

proc glm;
   class drug disease;
   model y=drug disease drug*disease / ss1 ss2 ss3 ss4;
run;
```

Output 53.3.1 Classes and Levels for Unbalanced Two-Way Design

Unbalanced Two-Way Analysis of Variance

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
drug	4	1 2 3 4
disease	3	1 2 3

Number of Observations Read	72
Number of Observations Used	58

Output 53.3.2 Analysis of Variance for Unbalanced Two-Way Design

Unbalanced Two-Way Analysis of Variance

The GLM Procedure

Dependent Variable: y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	11	4259.338506	387.212591	3.51	0.0013
Error	46	5080.816667	110.452536		
Corrected Total	57	9340.155172			

R-Square	Coeff Var	Root MSE	y Mean
0.456024	55.66750	10.50964	18.87931

Source	DF	Type I SS	Mean Square	F Value	Pr > F
drug	3	3133.238506	1044.412835	9.46	<.0001
disease	2	418.833741	209.416870	1.90	0.1617
drug*disease	6	707.266259	117.877710	1.07	0.3958

Source	DF	Type II SS	Mean Square	F Value	Pr > F
drug	3	3063.432863	1021.144288	9.25	<.0001
disease	2	418.833741	209.416870	1.90	0.1617
drug*disease	6	707.266259	117.877710	1.07	0.3958

Source	DF	Type III SS	Mean Square	F Value	Pr > F
drug	3	2997.471860	999.157287	9.05	<.0001
disease	2	415.873046	207.936523	1.88	0.1637
drug*disease	6	707.266259	117.877710	1.07	0.3958

Source	DF	Type IV SS	Mean Square	F Value	Pr > F
drug	3	2997.471860	999.157287	9.05	<.0001
disease	2	415.873046	207.936523	1.88	0.1637
drug*disease	6	707.266259	117.877710	1.07	0.3958

Note the differences among the four types of sums of squares. The Type I sum of squares for drug essentially tests for differences between the expected values of the arithmetic mean response for different drugs, unadjusted for the effect of disease. By contrast, the Type II sum of squares for drug measures the differences between arithmetic means for each drug after adjusting for disease. The Type III sum of squares measures the differences between predicted drug means over a balanced drug×disease population—that is, between the LS-means for drug. Finally, the Type IV sum of squares is the same as the Type III sum of squares in this case, since there are data for every drug-by-disease combination.

No matter which sum of squares you prefer to use, this analysis shows a significant difference among the four drugs, while the disease effect and the drug-by-disease interaction are not significant. As the previous discussion indicates, Type III sums of squares correspond to differences between LS-means, so you can follow up the Type III tests with a multiple-comparison analysis of the drug LS-means. Since the GLM procedure is interactive, you can accomplish this by submitting the following statements after the previous ones that performed the ANOVA.

```
   lsmeans drug / pdiff=all adjust=tukey;
run;
```

Both the LS-means themselves and a matrix of adjusted p-values for pairwise differences between them are displayed; see Output 53.3.3 and Output 53.3.4.

Output 53.3.3 LS-Means for Unbalanced ANOVA

Unbalanced Two-Way Analysis of Variance

The GLM Procedure
Least Squares Means
Adjustment for Multiple Comparisons: Tukey-Kramer

drug	y LSMEAN	LSMEAN Number
1	25.9944444	1
2	26.5555556	2
3	9.7444444	3
4	13.5444444	4

Output 53.3.4 Adjusted p-Values for Pairwise LS-Mean Differences

Least Squares Means for effect drug
Pr > |t| for H0: LSMean(i)=LSMean(j)

Dependent Variable: y

i/j	1	2	3	4
1		0.9989	0.0016	0.0107
2	0.9989		0.0011	0.0071
3	0.0016	0.0011		0.7870
4	0.0107	0.0071	0.7870	

The multiple-comparison analysis shows that drugs 1 and 2 have very similar effects, and that drugs 3 and 4 are also insignificantly different from each other. Evidently, the main contribution to the significant drug effect is the difference between the 1/2 pair and the 3/4 pair.

If ODS Graphics is enabled for the previous analysis, GLM also displays three additional plots by default:

- an interaction plot for the effects of disease and drug
- a mean plot of the drug LS-means
- a plot of the adjusted pairwise differences and their significance levels

The following statements reproduce the previous analysis with ODS Graphics enabled. Additionally, the PLOTS=MEANPLOT(CL) option specifies that confidence limits for the LS-means should also be displayed in the mean plot. The graphical results are shown in Output 53.3.5 through Output 53.3.7.

```
ods graphics on;
proc glm plot=meanplot(cl);
   class drug disease;
   model y=drug disease drug*disease;
   lsmeans drug / pdiff=all adjust=tukey;
run;
ods graphics off;
```

Output 53.3.5 Plot of Response by Drug and Disease

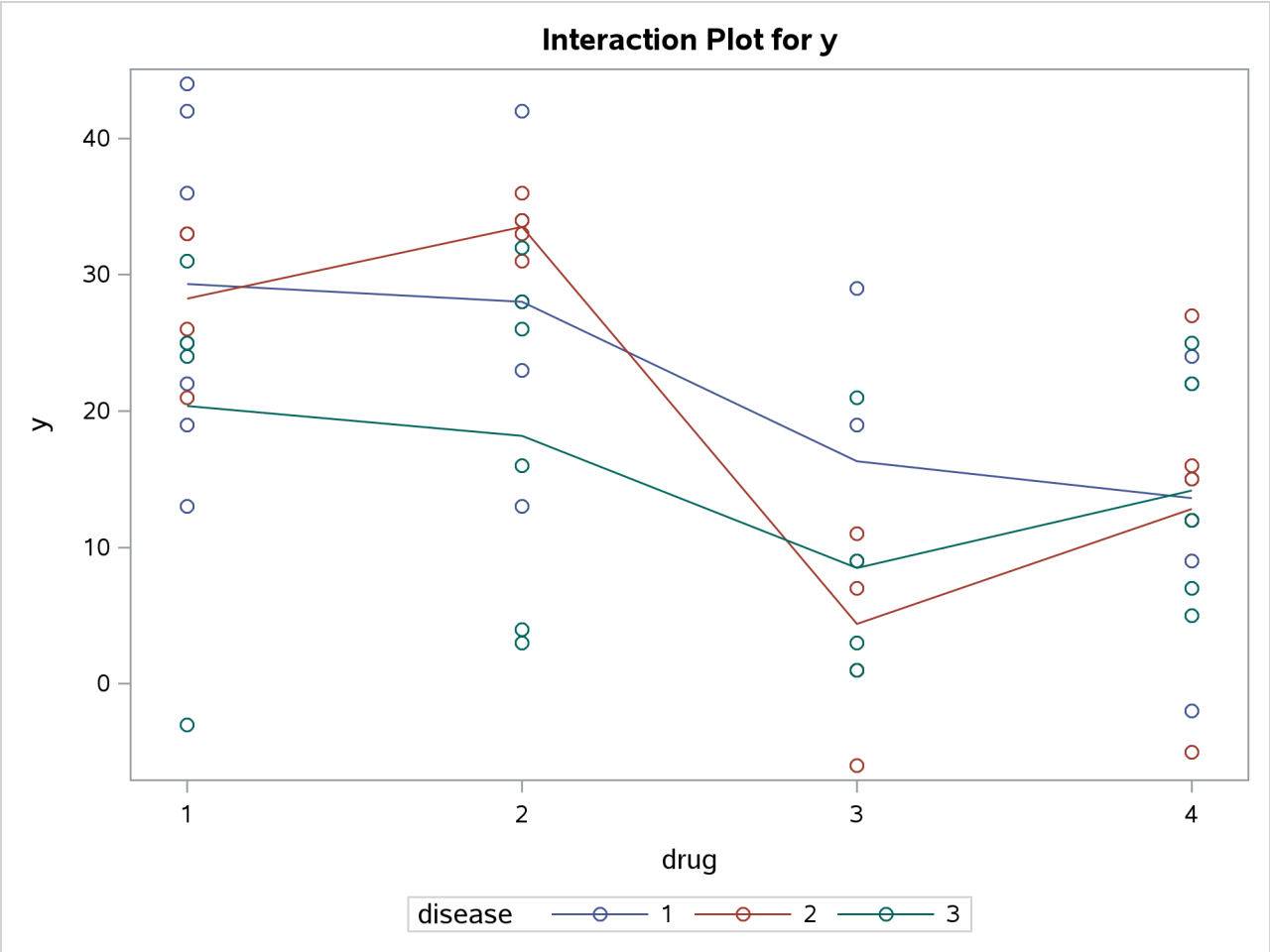

Output 53.3.6 Plot of Response LS-Means for Drug

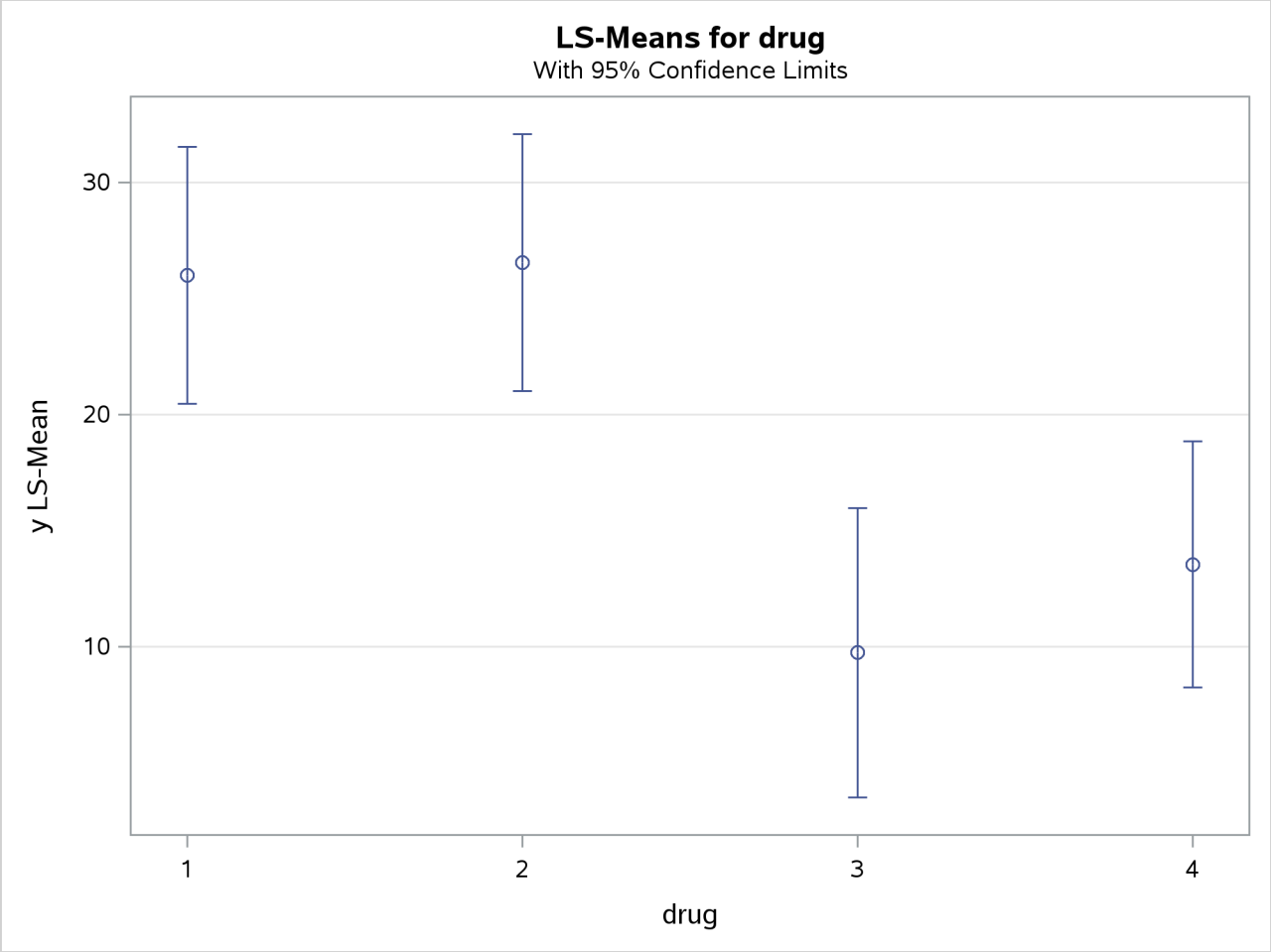

Output 53.3.7 Plot of Response LS-Mean Differences for Drug

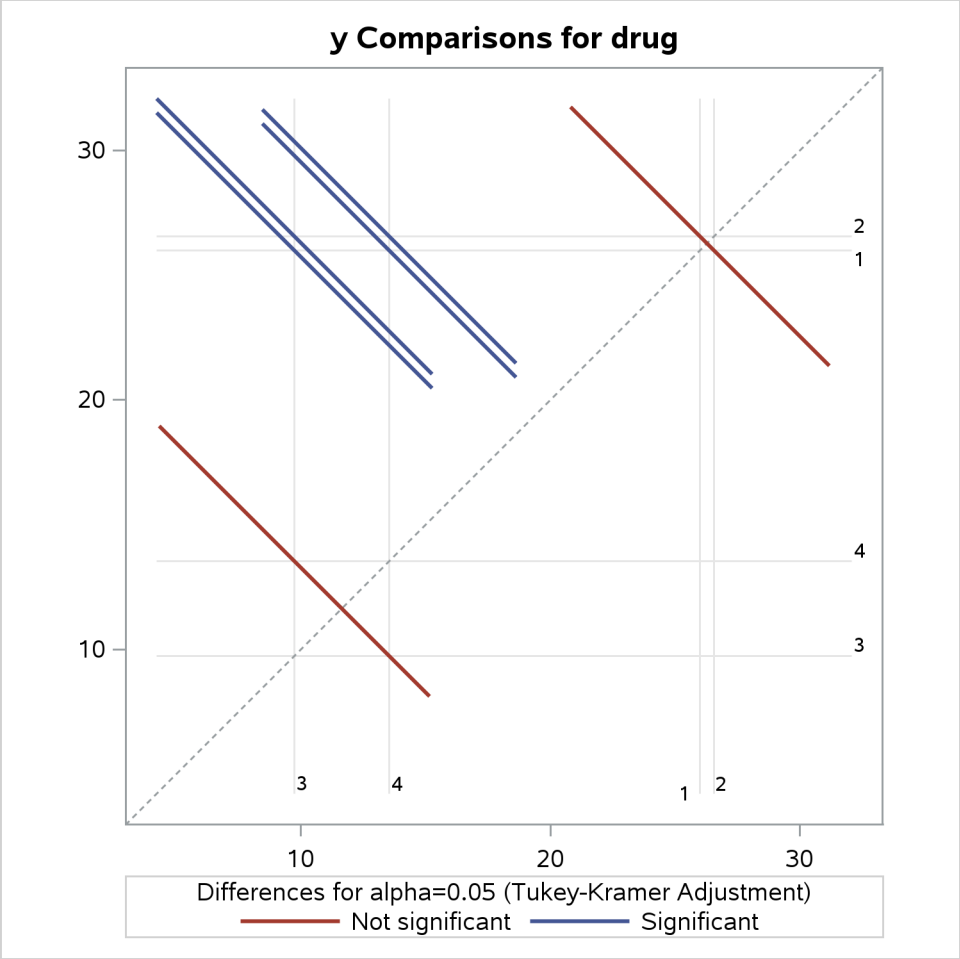

The significance of the drug differences is difficult to discern in the original data, as displayed in Output 53.3.5, but the plot of just the LS-means and their individual confidence limits in Output 53.3.6 makes it clearer. Finally, Output 53.3.7 indicates conclusively that the significance of the effect of drug is due to the difference between the two drug pairs (1, 2) and (3, 4).

Example 53.4: Analysis of Covariance

(View the complete code for this example (glmex4.sas) in the example repository.)

Analysis of covariance combines some of the features of both regression and analysis of variance. Typically, a continuous variable (the covariate) is introduced into the model of an analysis-of-variance experiment.

Data in the following example are selected from a larger experiment on the use of drugs in the treatment of leprosy (Snedecor and Cochran 1967, p. 422).

Variables in the study are as follows:

Drug	two antibiotics (A and D) and a control (F)
PreTreatment	a pretreatment score of leprosy bacilli
PostTreatment	a posttreatment score of leprosy bacilli

Ten patients are selected for each treatment (Drug), and six sites on each patient are measured for leprosy bacilli.

The covariate (a pretreatment score) is included in the model for increased precision in determining the effect of drug treatments on the posttreatment count of bacilli.

The following statements create the data set, perform a parallel-slopes analysis of covariance with PROC GLM, and compute Drug LS-means. These statements produce Output 53.4.1 and Output 53.4.2.

```
data DrugTest;
   input Drug $ PreTreatment PostTreatment @@;
   datalines;
A 11  6   A  8  0   A  5  2   A 14  8   A 19 11
A  6  4   A 10 13   A  6  1   A 11  8   A  3  0
D  6  0   D  6  2   D  7  3   D  8  1   D 18 18
D  8  4   D 19 14   D  8  9   D  5  1   D 15  9
F 16 13   F 13 10   F 11 18   F  9  5   F 21 23
F 16 12   F 12  5   F 12 16   F  7  1   F 12 20
;

proc glm data=DrugTest;
   class Drug;
   model PostTreatment = Drug PreTreatment / solution;
   lsmeans Drug / stderr pdiff cov out=adjmeans;
run;

proc print data=adjmeans;
run;
```

Output 53.4.1 Classes and Levels

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
Drug	3	A D F

Number of Observations Read	30
Number of Observations Used	30

Output 53.4.2 Overall Analysis of Variance

The GLM Procedure

Dependent Variable: PostTreatment

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	871.497403	290.499134	18.10	<.0001
Error	26	417.202597	16.046254		
Corrected Total	29	1288.700000			

Output 53.4.2 *continued*

R-Square	Coeff Var	Root MSE	PostTreatment Mean
0.676261	50.70604	4.005778	7.900000

This model assumes that the slopes relating posttreatment scores to pretreatment scores are parallel for all drugs. You can check this assumption by including the class-by-covariate interaction, Drug*PreTreatment, in the model and examining the ANOVA test for the significance of this effect. This extra test is omitted in this example, but it is insignificant, justifying the equal-slopes assumption.

In Output 53.4.3, the Type I SS for Drug (293.6) gives the between-drug sums of squares that are obtained for the analysis-of-variance model PostTreatment=Drug. This measures the difference between arithmetic means of posttreatment scores for different drugs, disregarding the covariate. The Type III SS for Drug (68.5537) gives the Drug sum of squares adjusted for the covariate. This measures the differences between Drug LS-means, controlling for the covariate. The Type I test is highly significant ($p = 0.001$), but the Type III test is not. This indicates that, while there is a statistically significant difference between the arithmetic drug means, this difference is reduced to below the level of background noise when you take the pretreatment scores into account. From the table of parameter estimates, you can derive the least squares predictive formula model for estimating posttreatment score based on pretreatment score and drug:

$$\text{post} = \begin{cases} (-0.435 + -3.446) & + \quad 0.987 \cdot \text{pre}, \quad \text{if Drug=A} \\ (-0.435 + -3.337) & + \quad 0.987 \cdot \text{pre}, \quad \text{if Drug=D} \\ -0.435 & + \quad 0.987 \cdot \text{pre}, \quad \text{if Drug=F} \end{cases}$$

Output 53.4.3 Tests and Parameter Estimates

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Drug	2	293.6000000	146.8000000	9.15	0.0010
PreTreatment	1	577.8974030	577.8974030	36.01	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Drug	2	68.5537106	34.2768553	2.14	0.1384
PreTreatment	1	577.8974030	577.8974030	36.01	<.0001

Parameter		Estimate		Standard Error	t Value	Pr > |t|
Intercept		-0.434671164	B	2.47135356	-0.18	0.8617
Drug	A	-3.446138280	B	1.88678065	-1.83	0.0793
Drug	D	-3.337166948	B	1.85386642	-1.80	0.0835
Drug	F	0.000000000	B	.	.	.
PreTreatment		0.987183811		0.16449757	6.00	<.0001

Output 53.4.4 displays the LS-means, which are, in a sense, the means adjusted for the covariate. The STDERR option in the LSMEANS statement causes the standard error of the LS-means and the probability of getting a larger t value under the hypothesis H_0: LS-mean $= 0$ to be included in this table as well. Specifying the PDIFF option causes all probability values for the hypothesis H_0: LS-mean(i) $=$ LS-mean(j) to be displayed, where the indexes i and j are numbered treatment levels.

Output 53.4.4 LS-Means

The GLM Procedure
Least Squares Means

Drug	PostTreatment LSMEAN	Standard Error	Pr > |t|	LSMEAN Number
A	6.7149635	1.2884943	<.0001	1
D	6.8239348	1.2724690	<.0001	2
F	10.1611017	1.3159234	<.0001	3

Least Squares Means for effect Drug
Pr > |t| for H0: LSMean(i)=LSMean(j)

Dependent Variable: PostTreatment

i/j	1	2	3
1		0.9521	0.0793
2	0.9521		0.0835
3	0.0793	0.0835	

The OUT= and COV options in the LSMEANS statement create a data set of the estimates, their standard errors, and the variances and covariances of the LS-means, which is displayed in Output 53.4.5.

Output 53.4.5 LS-Means Output Data Set

Obs	_NAME_	Drug	LSMEAN	STDERR	NUMBER	COV1	COV2	COV3
1	PostTreatment	A	6.7150	1.28849	1	1.66022	0.02844	-0.08403
2	PostTreatment	D	6.8239	1.27247	2	0.02844	1.61918	-0.04299
3	PostTreatment	F	10.1611	1.31592	3	-0.08403	-0.04299	1.73165

The new graphical features of PROC GLM enable you to visualize the fitted analysis of covariance model. The following statements enable ODS Graphics by specifying the ODS GRAPHICS statement and then fit an analysis-of-covariance model with LS-means for Drug.

```
ods graphics on;

proc glm data=DrugTest plot=meanplot(cl);
   class Drug;
   model PostTreatment = Drug PreTreatment;
   lsmeans Drug / pdiff;
run;

ods graphics off;
```

With graphics enabled, the GLM procedure output includes an analysis-of-covariance plot, as in Output 53.4.6. The LSMEANS statement produces a plot of the LS-means; the SAS statements previously shown use the PLOTS=MEANPLOT(CL) option to add confidence limits for the individual LS-means, shown in Output 53.4.7. If you also specify the PDIFF option in the LSMEANS statement, the output also includes a plot appropriate for the type of LS-mean differences computed. In this case, the default is to compare all LS-means with each other pairwise, so the plot is a "diffogram" or "mean-mean scatter plot" (Hsu 1996), as in Output 53.4.8. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the GLM procedure, see the section

"ODS Graphics" on page 4270.

Output 53.4.6 Analysis of Covariance Plot of PostTreatment Score by Drug and PreTreatment Score

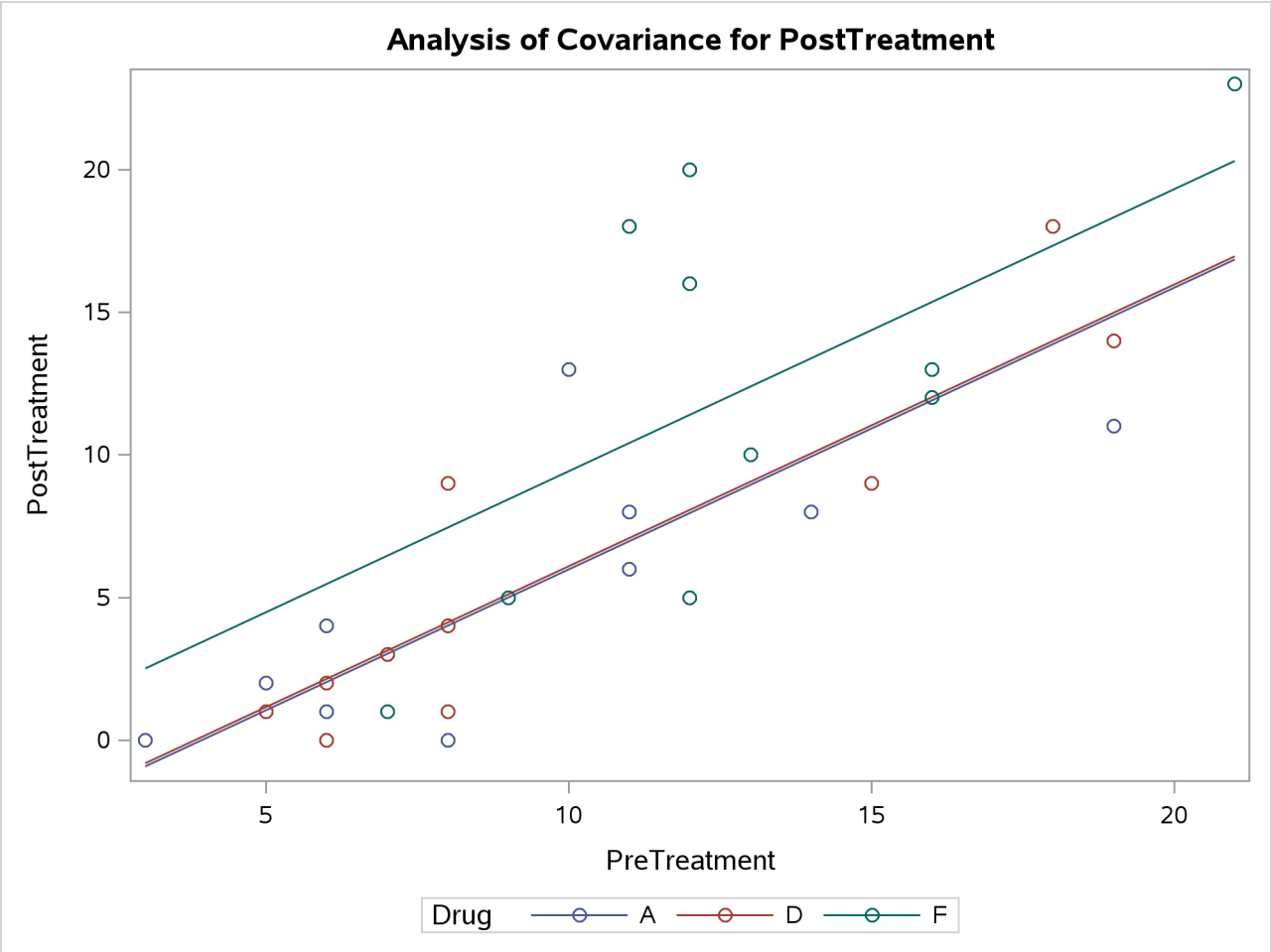

Output 53.4.7 LS-Means for PostTreatment Score by Drug

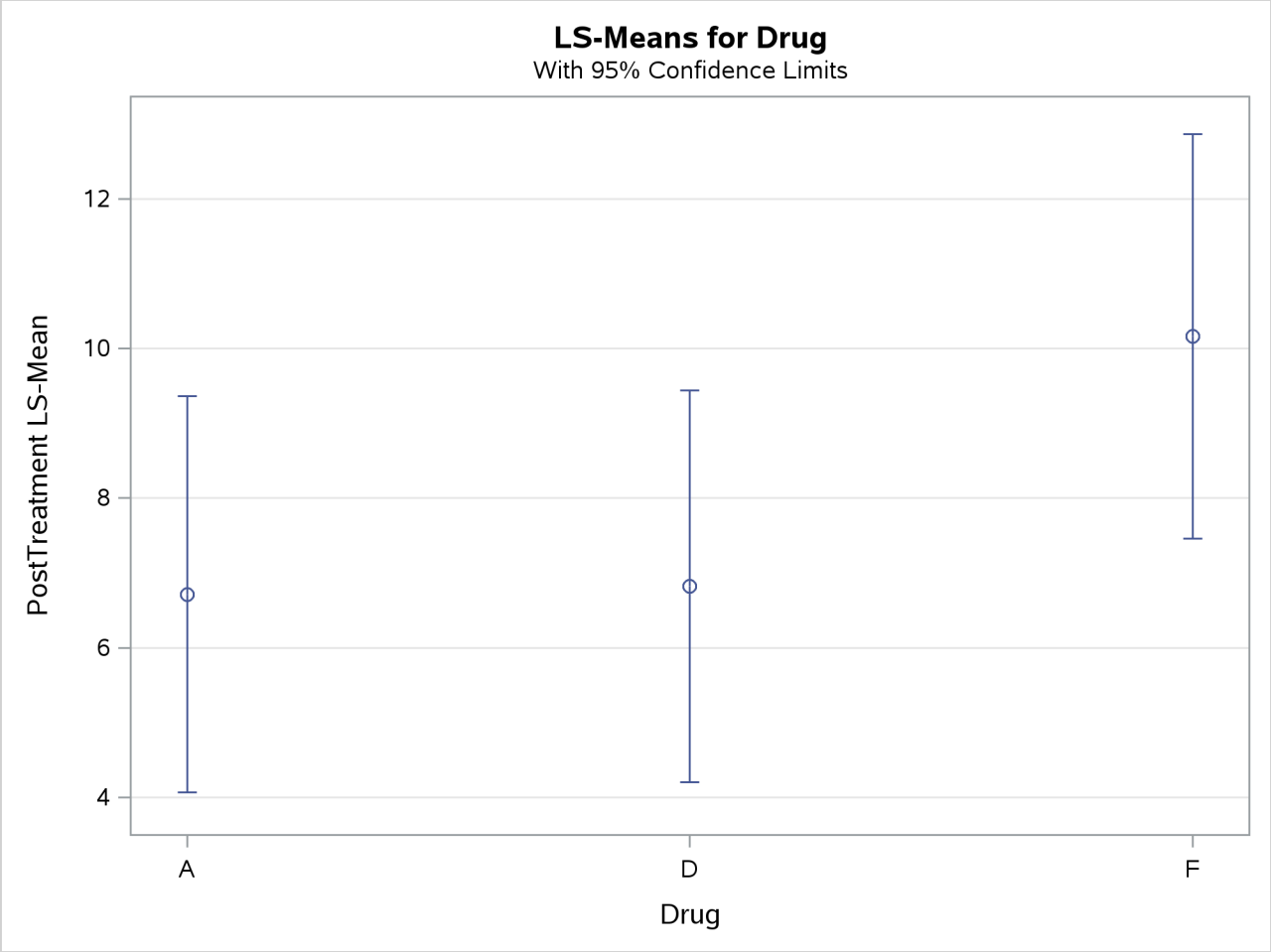

Output 53.4.8 Plot of Differences between Drug LS-Means for PostTreatment Scores

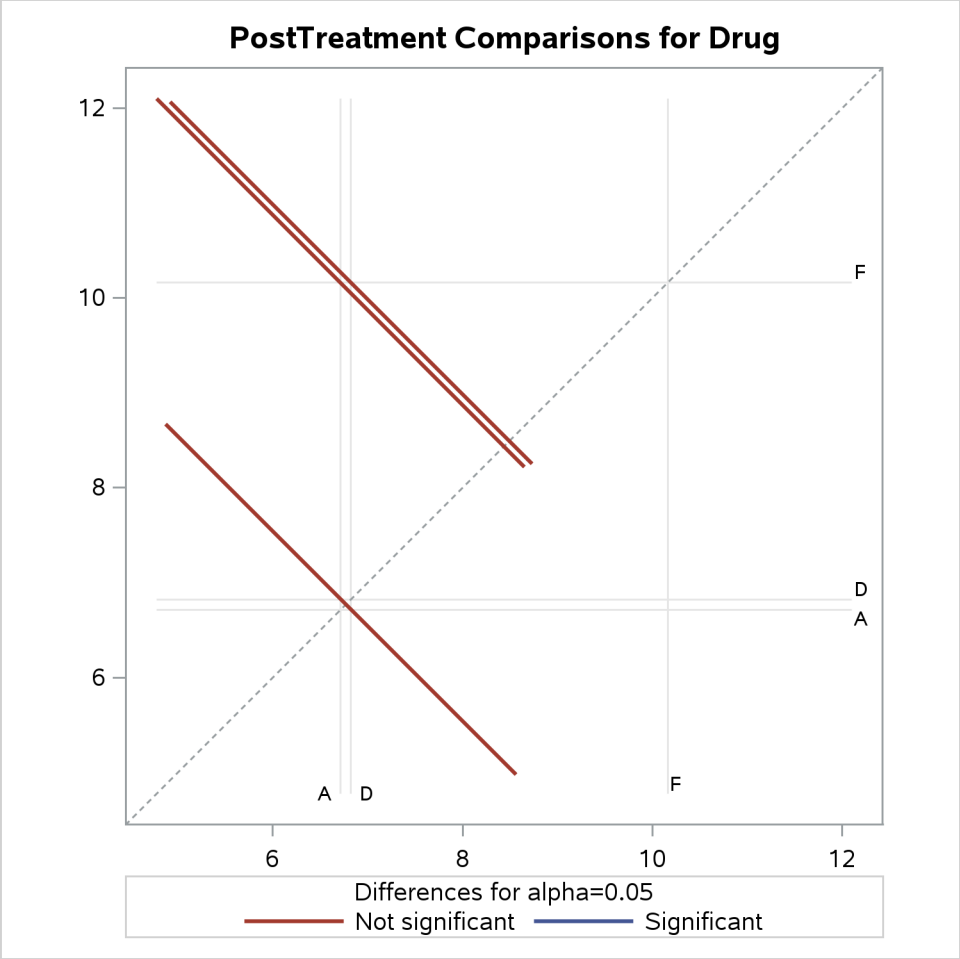

The analysis of covariance plot Output 53.4.6 makes it clear that the control (drug F) has higher posttreatment scores across the range of pretreatment scores, while the fitted models for the two antibiotics (drugs A and D) nearly coincide. Similarly, while the diffogram Output 53.4.8 indicates that none of the LS-mean differences are significant at the 5% level, the difference between the LS-means for the two antibiotics is much closer to zero than the differences between either one and the control.

Example 53.5: Three-Way Analysis of Variance with Contrasts

(View the complete code for this example (glmex5.sas) in the example repository.)

This example uses data from Cochran and Cox (1957, p. 176) to illustrate the analysis of a three-way factorial design with replication, including the use of the CONTRAST statement with interactions, the OUTSTAT= data set, and the SLICE= option in the LSMEANS statement.

The object of the study is to determine the effects of electric current on denervated muscle. The variables are as follows:

Rep	the replicate number, 1 or 2
Time	the length of time the current is applied to the muscle, ranging from 1 to 4
Current	the level of electric current applied, ranging from 1 to 4
Number	the number of treatments per day, ranging from 1 to 3
MuscleWeight	the weight of the denervated muscle

The following statements produce Output 53.5.1 through Output 53.5.4.

```
data muscles;
   do Rep=1 to 2;
      do Time=1 to 4;
         do Current=1 to 4;
            do Number=1 to 3;
               input MuscleWeight @@;
               output;
            end;
         end;
      end;
   end;
   datalines;
72 74 69 61 61 65 62 65 70 85 76 61
67 52 62 60 55 59 64 65 64 67 72 60
57 66 72 72 43 43 63 66 72 56 75 92
57 56 78 60 63 58 61 79 68 73 86 71
46 74 58 60 64 52 71 64 71 53 65 66
44 58 54 57 55 51 62 61 79 60 78 82
53 50 61 56 57 56 56 56 71 56 58 69
46 55 64 56 55 57 64 66 62 59 58 88
;

proc glm outstat=summary;
   class Rep Current Time Number;
   model MuscleWeight = Rep Current|Time|Number;
   contrast 'Time in Current 3'
      Time 1 0 0 -1 Current*Time 0 0 0 0 0 0 0 0 1 0 0 -1,
      Time 0 1 0 -1 Current*Time 0 0 0 0 0 0 0 0 0 1 0 -1,
      Time 0 0 1 -1 Current*Time 0 0 0 0 0 0 0 0 0 0 1 -1;
   contrast 'Current 1 versus 2' Current 1 -1;
   lsmeans Current*Time / slice=Current;
run;

proc print data=summary;
run;
```

The first CONTRAST statement examines the effects of Time within level 3 of Current. This is also called the *simple effect* of Time within Current*Time. Note that, since there are three degrees of freedom, it is necessary to specify three rows in the CONTRAST statement, separated by commas. Since the parameterization that PROC GLM uses is determined in part by the ordering of the variables in the CLASS statement, Current is specified before Time so that the Time parameters are nested within the Current*Time parameters; thus, the Current*Time contrast coefficients in each row are simply the Time coefficients of that row within the appropriate level of Current.

The second CONTRAST statement isolates a single-degree-of-freedom effect corresponding to the difference between the first two levels of Current. You can use such a contrast in a large experiment where certain preplanned comparisons are important, but you want to take advantage of the additional error degrees of freedom available when all levels of the factors are considered.

The LSMEANS statement with the SLICE= option is an alternative way to test for the simple effect of Time within Current*Time. In addition to listing the LS-means for each current strength and length of time, it gives a table of F tests for differences between the LS-means across Time within each Current level. In some cases, this can be a way to disentangle a complex interaction.

Output 53.5.1 Overall Analysis

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
Rep	2	1 2
Current	4	1 2 3 4
Time	4	1 2 3 4
Number	3	1 2 3

Number of Observations Read	96
Number of Observations Used	96

The GLM Procedure

Dependent Variable: MuscleWeight

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	48	5782.916667	120.477431	1.77	0.0261
Error	47	3199.489583	68.074246		
Corrected Total	95	8982.406250			

R-Square	Coeff Var	Root MSE	MuscleWeight Mean
0.643805	13.05105	8.250712	63.21875

The output, shown in Output 53.5.2 and Output 53.5.3, indicates that the main effects for Rep, Current, and Number are significant (with p-values of 0.0045, <0.0001, and 0.0461, respectively), but the main effect for Time is not significant, indicating that, in general, it does not matter how long the current is applied. None of the interaction terms are significant, nor are the contrasts significant. Notice that the row in the sliced ANOVA table corresponding to level 3 of current matches the “Time in Current 3” contrast.

Output 53.5.2 Individual Effects and Contrasts

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Rep	1	605.010417	605.010417	8.89	0.0045
Current	3	2145.447917	715.149306	10.51	<.0001
Time	3	223.114583	74.371528	1.09	0.3616
Current*Time	9	298.677083	33.186343	0.49	0.8756
Number	2	447.437500	223.718750	3.29	0.0461
Current*Number	6	644.395833	107.399306	1.58	0.1747
Time*Number	6	367.979167	61.329861	0.90	0.5023
Current*Time*Number	18	1050.854167	58.380787	0.86	0.6276

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Rep	1	605.010417	605.010417	8.89	0.0045
Current	3	2145.447917	715.149306	10.51	<.0001
Time	3	223.114583	74.371528	1.09	0.3616
Current*Time	9	298.677083	33.186343	0.49	0.8756
Number	2	447.437500	223.718750	3.29	0.0461
Current*Number	6	644.395833	107.399306	1.58	0.1747
Time*Number	6	367.979167	61.329861	0.90	0.5023
Current*Time*Number	18	1050.854167	58.380787	0.86	0.6276

Contrast	DF	Contrast SS	Mean Square	F Value	Pr > F
Time in Current 3	3	34.83333333	11.61111111	0.17	0.9157
Current 1 versus 2	1	99.18750000	99.18750000	1.46	0.2334

Output 53.5.3 Simple Effects of Time

The GLM Procedure
Least Squares Means

Current*Time Effect Sliced by Current for MuscleWeight					
Current	DF	Sum of Squares	Mean Square	F Value	Pr > F
1	3	271.458333	90.486111	1.33	0.2761
2	3	120.666667	40.222222	0.59	0.6241
3	3	34.833333	11.611111	0.17	0.9157
4	3	94.833333	31.611111	0.46	0.7085

The SS, F statistics, and p-values can be stored in an OUTSTAT= data set, as shown in Output 53.5.4.

Output 53.5.4 Contents of the OUTSTAT= Data Set

Obs	_NAME_	_SOURCE_	_TYPE_	DF	SS	F	PROB
1	MuscleWeight	ERROR	ERROR	47	3199.49	.	.
2	MuscleWeight	Rep	SS1	1	605.01	8.8875	0.00454
3	MuscleWeight	Current	SS1	3	2145.45	10.5054	0.00002
4	MuscleWeight	Time	SS1	3	223.11	1.0925	0.36159
5	MuscleWeight	Current*Time	SS1	9	298.68	0.4875	0.87562
6	MuscleWeight	Number	SS1	2	447.44	3.2864	0.04614
7	MuscleWeight	Current*Number	SS1	6	644.40	1.5777	0.17468
8	MuscleWeight	Time*Number	SS1	6	367.98	0.9009	0.50231
9	MuscleWeight	Current*Time*Number	SS1	18	1050.85	0.8576	0.62757
10	MuscleWeight	Rep	SS3	1	605.01	8.8875	0.00454
11	MuscleWeight	Current	SS3	3	2145.45	10.5054	0.00002
12	MuscleWeight	Time	SS3	3	223.11	1.0925	0.36159
13	MuscleWeight	Current*Time	SS3	9	298.68	0.4875	0.87562
14	MuscleWeight	Number	SS3	2	447.44	3.2864	0.04614
15	MuscleWeight	Current*Number	SS3	6	644.40	1.5777	0.17468
16	MuscleWeight	Time*Number	SS3	6	367.98	0.9009	0.50231
17	MuscleWeight	Current*Time*Number	SS3	18	1050.85	0.8576	0.62757
18	MuscleWeight	Time in Current 3	CONTRAST	3	34.83	0.1706	0.91574
19	MuscleWeight	Current 1 versus 2	CONTRAST	1	99.19	1.4570	0.23344

Example 53.6: Multivariate Analysis of Variance

(View the complete code for this example (glmex6.sas) in the example repository.)

This example employs multivariate analysis of variance (MANOVA) to measure differences in the chemical characteristics of ancient pottery found at four kiln sites in Great Britain. The data are from Tubb, Parker, and Nickless (1980), as reported in Hand et al. (1994).

For each of 26 samples of pottery, the percentages of oxides of five metals are measured. The following statements create the data set and invoke the GLM procedure to perform a one-way MANOVA. Additionally, it is of interest to know whether the pottery from one site in Wales (Llanederyn) differs from the samples from other sites; a CONTRAST statement is used to test this hypothesis.

```
title "Romano-British Pottery";
data pottery;
   input Site $12. Al Fe Mg Ca Na;
   datalines;
Llanederyn   14.4 7.00 4.30 0.15 0.51
Llanederyn   13.8 7.08 3.43 0.12 0.17
Llanederyn   14.6 7.09 3.88 0.13 0.20
Llanederyn   11.5 6.37 5.64 0.16 0.14
Llanederyn   13.8 7.06 5.34 0.20 0.20
Llanederyn   10.9 6.26 3.47 0.17 0.22
Llanederyn   10.1 4.26 4.26 0.20 0.18
Llanederyn   11.6 5.78 5.91 0.18 0.16
Llanederyn   11.1 5.49 4.52 0.29 0.30
Llanederyn   13.4 6.92 7.23 0.28 0.20
```

```
Llanederyn   12.4 6.13 5.69 0.22 0.54
Llanederyn   13.1 6.64 5.51 0.31 0.24
Llanederyn   12.7 6.69 4.45 0.20 0.22
Llanederyn   12.5 6.44 3.94 0.22 0.23
Caldicot     11.8 5.44 3.94 0.30 0.04
Caldicot     11.6 5.39 3.77 0.29 0.06
IslandThorns 18.3 1.28 0.67 0.03 0.03
IslandThorns 15.8 2.39 0.63 0.01 0.04
IslandThorns 18.0 1.50 0.67 0.01 0.06
IslandThorns 18.0 1.88 0.68 0.01 0.04
IslandThorns 20.8 1.51 0.72 0.07 0.10
AshleyRails  17.7 1.12 0.56 0.06 0.06
AshleyRails  18.3 1.14 0.67 0.06 0.05
AshleyRails  16.7 0.92 0.53 0.01 0.05
AshleyRails  14.8 2.74 0.67 0.03 0.05
AshleyRails  19.1 1.64 0.60 0.10 0.03
;

proc glm data=pottery;
   class Site;
   model Al Fe Mg Ca Na = Site;
   contrast 'Llanederyn vs. the rest' Site 1 1 1 -3;
   manova h=_all_ / printe printh;
run;
```

After the summary information, displayed in Output 53.6.1, PROC GLM produces the univariate analyses for each of the dependent variables, as shown in Output 53.6.2 through Output 53.6.6. These analyses show that sites are significantly different for all oxides individually. You can suppress these univariate analyses by specifying the NOUNI option in the MODEL statement.

Output 53.6.1 Summary Information about Groups

Romano-British Pottery

The GLM Procedure

Class Level Information

Class	Levels	Values
Site	4	AshleyRails Caldicot IslandThorns Llanederyn

Number of Observations Read	26
Number of Observations Used	26

Output 53.6.2 Univariate Analysis of Variance for Aluminum Oxide

Romano-British Pottery

The GLM Procedure

Dependent Variable: Al

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	175.6103187	58.5367729	26.67	<.0001
Error	22	48.2881429	2.1949156		
Corrected Total	25	223.8984615			

R-Square	Coeff Var	Root MSE	Al Mean
0.784330	10.22284	1.481525	14.49231

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Site	3	175.6103187	58.5367729	26.67	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Site	3	175.6103187	58.5367729	26.67	<.0001

Contrast	DF	Contrast SS	Mean Square	F Value	Pr > F
Llanederyn vs. the rest	1	58.58336640	58.58336640	26.69	<.0001

Output 53.6.3 Univariate Analysis of Variance for Iron Oxide

Romano-British Pottery

The GLM Procedure

Dependent Variable: Fe

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	134.2216158	44.7405386	89.88	<.0001
Error	22	10.9508457	0.4977657		
Corrected Total	25	145.1724615			

R-Square	Coeff Var	Root MSE	Fe Mean
0.924567	15.79171	0.705525	4.467692

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Site	3	134.2216158	44.7405386	89.88	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Site	3	134.2216158	44.7405386	89.88	<.0001

Contrast	DF	Contrast SS	Mean Square	F Value	Pr > F
Llanederyn vs. the rest	1	71.15144132	71.15144132	142.94	<.0001

Output 53.6.4 Univariate Analysis of Variance for Calcium Oxide

Romano-British Pottery

The GLM Procedure

Dependent Variable: Ca

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	0.20470275	0.06823425	29.16	<.0001
Error	22	0.05148571	0.00234026		
Corrected Total	25	0.25618846			

R-Square	Coeff Var	Root MSE	Ca Mean
0.799032	33.01265	0.048376	0.146538

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Site	3	0.20470275	0.06823425	29.16	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Site	3	0.20470275	0.06823425	29.16	<.0001

Contrast	DF	Contrast SS	Mean Square	F Value	Pr > F
Llanederyn vs. the rest	1	0.03531688	0.03531688	15.09	0.0008

Output 53.6.5 Univariate Analysis of Variance for Magnesium Oxide

Romano-British Pottery

The GLM Procedure

Dependent Variable: Mg

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	103.3505270	34.4501757	49.12	<.0001
Error	22	15.4296114	0.7013460		
Corrected Total	25	118.7801385			

R-Square	Coeff Var	Root MSE	Mg Mean
0.870099	26.65777	0.837464	3.141538

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Site	3	103.3505270	34.4501757	49.12	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Site	3	103.3505270	34.4501757	49.12	<.0001

Contrast	DF	Contrast SS	Mean Square	F Value	Pr > F
Llanederyn vs. the rest	1	56.59349339	56.59349339	80.69	<.0001

Output 53.6.6 Univariate Analysis of Variance for Sodium Oxide

Romano-British Pottery

The GLM Procedure

Dependent Variable: Na

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	0.25824560	0.08608187	9.50	0.0003
Error	22	0.19929286	0.00905877		
Corrected Total	25	0.45753846			

R-Square	Coeff Var	Root MSE	Na Mean
0.564424	60.06350	0.095178	0.158462

Source	DF	Type I SS	Mean Square	F Value	Pr > F
Site	3	0.25824560	0.08608187	9.50	0.0003

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Site	3	0.25824560	0.08608187	9.50	0.0003

Contrast	DF	Contrast SS	Mean Square	F Value	Pr > F
Llanederyn vs. the rest	1	0.23344446	0.23344446	25.77	<.0001

The PRINTE option in the MANOVA statement displays the elements of the error matrix, also called the Error Sums of Squares and Crossproducts matrix. (See Output 53.6.7.) The diagonal elements of this matrix are the error sums of squares from the corresponding univariate analyses.

The PRINTE option also displays the partial correlation matrix associated with the E matrix. In this example, none of the oxides are very strongly correlated; the strongest correlation ($r = 0.488$) is between magnesium oxide and calcium oxide.

Output 53.6.7 Error SSCP Matrix and Partial Correlations

Romano-British Pottery

The GLM Procedure
Multivariate Analysis of Variance

E = Error SSCP Matrix					
	Al	Fe	Mg	Ca	Na
Al	48.288142857	7.0800714286	0.6080142857	0.1064714286	0.5889571429
Fe	7.0800714286	10.950845714	0.5270571429	-0.155194286	0.0667585714
Mg	0.6080142857	0.5270571429	15.429611429	0.4353771429	0.0276157143
Ca	0.1064714286	-0.155194286	0.4353771429	0.0514857143	0.0100785714
Na	0.5889571429	0.0667585714	0.0276157143	0.0100785714	0.1992928571

Output 53.6.7 *continued*

Partial Correlation Coefficients from the Error SSCP Matrix / Prob > |r|					
DF = 22	**Al**	**Fe**	**Mg**	**Ca**	**Na**
Al	1.000000	0.307889 0.1529	0.022275 0.9196	0.067526 0.7595	0.189853 0.3856
Fe	0.307889 0.1529	1.000000	0.040547 0.8543	-0.206685 0.3440	0.045189 0.8378
Mg	0.022275 0.9196	0.040547 0.8543	1.000000	0.488478 0.0180	0.015748 0.9431
Ca	0.067526 0.7595	-0.206685 0.3440	0.488478 0.0180	1.000000	0.099497 0.6515
Na	0.189853 0.3856	0.045189 0.8378	0.015748 0.9431	0.099497 0.6515	1.000000

The PRINTH option produces the SSCP matrix for the hypotheses being tested (Site and the contrast); see Output 53.6.8 and Output 53.6.9. Since the Type III SS are the highest-level SS produced by PROC GLM by default, and since the HTYPE= option is not specified, the SSCP matrix for Site gives the Type III $\mathbf{H}$ matrix. The diagonal elements of this matrix are the model sums of squares from the corresponding univariate analyses.

Four multivariate tests are computed, all based on the characteristic roots and vectors of $\mathbf{E}^{-1}\mathbf{H}$. These roots and vectors are displayed along with the tests. All four tests can be transformed to variates that have F distributions under the null hypothesis. Note that the four tests all give the same results for the contrast, since it has only one degree of freedom. In this case, the multivariate analysis matches the univariate results: there is an overall difference between the chemical composition of samples from different sites, and the samples from Llanederyn are different from the average of the other sites.

Output 53.6.8 Hypothesis SSCP Matrix and Multivariate Tests for Overall Site Effect

Romano-British Pottery

The GLM Procedure
Multivariate Analysis of Variance

H = Type III SSCP Matrix for Site					
	Al	**Fe**	**Mg**	**Ca**	**Na**
Al	175.61031868	-149.295533	-130.8097066	-5.889163736	-5.372264835
Fe	-149.295533	134.22161582	117.74503516	4.8217865934	5.3259491209
Mg	-130.8097066	117.74503516	103.35052703	4.2091613187	4.7105458242
Ca	-5.889163736	4.8217865934	4.2091613187	0.2047027473	0.154782967
Na	-5.372264835	5.3259491209	4.7105458242	0.154782967	0.2582456044

Output 53.6.8 *continued*

Characteristic Roots and Vectors of: E Inverse * H, where
H = Type III SSCP Matrix for Site
E = Error SSCP Matrix

Characteristic Vector V'EV=1

Characteristic Root	Percent	Al	Fe	Mg	Ca	Na
34.1611140	96.39	0.09562211	-0.26330469	-0.05305978	-1.87982100	-0.47071123
1.2500994	3.53	0.02651891	-0.01239715	0.17564390	-4.25929785	1.23727668
0.0275396	0.08	0.09082220	0.13159869	0.03508901	-0.15701602	-1.39364544
0.0000000	0.00	0.03673984	-0.15129712	0.20455529	0.54624873	-0.17402107
0.0000000	0.00	0.06862324	0.03056912	-0.10662399	2.51151978	1.23668841

MANOVA Test Criteria and F Approximations for the Hypothesis of No Overall Site Effect
H = Type III SSCP Matrix for Site
E = Error SSCP Matrix

S=3 M=0.5 N=8

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.01230091	13.09	15	50.091	<.0001
Pillai's Trace	1.55393619	4.30	15	60	<.0001
Hotelling-Lawley Trace	35.43875302	40.59	15	29.13	<.0001
Roy's Greatest Root	34.16111399	136.64	5	20	<.0001

NOTE: F Statistic for Roy's Greatest Root is an upper bound.

Output 53.6.9 Hypothesis SSCP Matrix and Multivariate Tests for Differences between Llanederyn and the Other Sites

H = Contrast SSCP Matrix for Llanederyn vs. the rest

	Al	Fe	Mg	Ca	Na
Al	58.583366402	-64.56230291	-57.57983466	-1.438395503	-3.698102513
Fe	-64.56230291	71.151441323	63.456352116	1.5851961376	4.0755256878
Mg	-57.57983466	63.456352116	56.593493386	1.4137558201	3.6347541005
Ca	-1.438395503	1.5851961376	1.4137558201	0.0353168783	0.0907993915
Na	-3.698102513	4.0755256878	3.6347541005	0.0907993915	0.2334444577

Characteristic Roots and Vectors of: E Inverse * H, where
H = Contrast SSCP Matrix for Llanederyn vs. the rest
E = Error SSCP Matrix

Characteristic Vector V'EV=1

Characteristic Root	Percent	Al	Fe	Mg	Ca	Na
16.1251646	100.00	-0.08883488	0.25458141	0.08723574	0.98158668	0.71925759
0.0000000	0.00	-0.00503538	0.03825743	-0.17632854	5.16256699	-0.01022754
0.0000000	0.00	0.00162771	-0.08885364	-0.01774069	-0.83096817	2.17644566
0.0000000	0.00	0.04450136	-0.15722494	0.22156791	0.00000000	0.00000000
0.0000000	0.00	0.11939206	0.10833549	0.00000000	0.00000000	0.00000000

Output 53.6.9 *continued*

MANOVA Test Criteria and Exact F Statistics for the Hypothesis of No Overall Llanederyn vs. the rest Effect
H = Contrast SSCP Matrix for Llanederyn vs. the rest
E = Error SSCP Matrix

S=1 M=1.5 N=8

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.05839360	58.05	5	18	<.0001
Pillai's Trace	0.94160640	58.05	5	18	<.0001
Hotelling-Lawley Trace	16.12516462	58.05	5	18	<.0001
Roy's Greatest Root	16.12516462	58.05	5	18	<.0001

Example 53.7: Repeated Measures Analysis of Variance

(View the complete code for this example (glmex7.sas) in the example repository.)

This example uses data from Cole and Grizzle (1966) to illustrate a commonly occurring repeated measures ANOVA design. Sixteen dogs are randomly assigned to four groups. (One animal is removed from the analysis due to a missing value for one dependent variable.) Dogs in each group receive either morphine or trimethaphan (variable Drug) and have either depleted or intact histamine levels (variable Depleted) before receiving the drugs. The dependent variable is the blood concentration of histamine at 0, 1, 3, and 5 minutes after injection of the drug. Logarithms are applied to these concentrations to minimize correlation between the mean and the variance of the data.

The following SAS statements perform both univariate and multivariate repeated measures analyses and produce Output 53.7.1 through Output 53.7.7.

```
data dogs;
   input Drug $12. Depleted $ Histamine0 Histamine1
         Histamine3 Histamine5;
   LogHistamine0=log(Histamine0);
   LogHistamine1=log(Histamine1);
   LogHistamine3=log(Histamine3);
   LogHistamine5=log(Histamine5);
   datalines;
Morphine      N  .04  .20  .10  .08
Morphine      N  .02  .06  .02  .02
Morphine      N  .07 1.40  .48  .24
Morphine      N  .17  .57  .35  .24
Morphine      Y  .10  .09  .13  .14
Morphine      Y  .12  .11  .10   .
Morphine      Y  .07  .07  .06  .07
Morphine      Y  .05  .07  .06  .07
Trimethaphan  N  .03  .62  .31  .22
Trimethaphan  N  .03 1.05  .73  .60
Trimethaphan  N  .07  .83 1.07  .80
Trimethaphan  N  .09 3.13 2.06 1.23
Trimethaphan  Y  .10  .09  .09  .08
Trimethaphan  Y  .08  .09  .09  .10
Trimethaphan  Y  .13  .10  .12  .12
Trimethaphan  Y  .06  .05  .05  .05
```

```
;

proc glm;
   class Drug Depleted;
   model LogHistamine0--LogHistamine5 =
         Drug Depleted Drug*Depleted / nouni;
   repeated Time 4 (0 1 3 5) polynomial / summary printe;
run;
```

The NOUNI option in the MODEL statement suppresses the individual ANOVA tables for the original dependent variables. These analyses are usually of no interest in a repeated measures analysis. The POLYNOMIAL option in the REPEATED statement indicates that the transformation used to implement the repeated measures analysis is an orthogonal polynomial transformation, and the SUMMARY option requests that the univariate analyses for the orthogonal polynomial contrast variables be displayed. The parenthetical numbers (0 1 3 5) determine the spacing of the orthogonal polynomials used in the analysis.

Output 53.7.1 Summary Information about Groups

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
Drug	2	Morphine Trimethaphan
Depleted	2	N Y

Number of Observations Read	16
Number of Observations Used	15

The "Repeated Measures Level Information" table gives information about the repeated measures effect; it is displayed in Output 53.7.2. In this example, the within-subject (within-dog) effect is Time, which has the levels 0, 1, 3, and 5.

Output 53.7.2 Repeated Measures Levels

The GLM Procedure
Repeated Measures Analysis of Variance

Repeated Measures Level Information				
Dependent Variable	**LogHistamine0**	**LogHistamine1**	**LogHistamine3**	**LogHistamine5**
Level of Time	0	1	3	5

The multivariate analyses for within-subject effects and related interactions are displayed in Output 53.7.3. For the example, the first table displayed shows that the TIME effect is significant. In addition, the Time*Drug*Depleted interaction is significant, as shown in the fourth table. This means that the effect of Time on the blood concentration of histamine is different for the four Drug*Depleted combinations studied.

Output 53.7.3 Multivariate Tests of Within-Subject Effects

MANOVA Test Criteria and Exact F Statistics for the Hypothesis of no Time Effect
H = Type III SSCP Matrix for Time
E = Error SSCP Matrix

S=1 M=0.5 N=3.5

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.11097706	24.03	3	9	0.0001
Pillai's Trace	0.88902294	24.03	3	9	0.0001
Hotelling-Lawley Trace	8.01087137	24.03	3	9	0.0001
Roy's Greatest Root	8.01087137	24.03	3	9	0.0001

MANOVA Test Criteria and Exact F Statistics for the Hypothesis of no Time*Drug Effect
H = Type III SSCP Matrix for Time*Drug
E = Error SSCP Matrix

S=1 M=0.5 N=3.5

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.34155984	5.78	3	9	0.0175
Pillai's Trace	0.65844016	5.78	3	9	0.0175
Hotelling-Lawley Trace	1.92774470	5.78	3	9	0.0175
Roy's Greatest Root	1.92774470	5.78	3	9	0.0175

MANOVA Test Criteria and Exact F Statistics for the Hypothesis of no Time*Depleted Effect
H = Type III SSCP Matrix for Time*Depleted
E = Error SSCP Matrix

S=1 M=0.5 N=3.5

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.12339988	21.31	3	9	0.0002
Pillai's Trace	0.87660012	21.31	3	9	0.0002
Hotelling-Lawley Trace	7.10373567	21.31	3	9	0.0002
Roy's Greatest Root	7.10373567	21.31	3	9	0.0002

MANOVA Test Criteria and Exact F Statistics for the Hypothesis of no Time*Drug*Depleted Effect
H = Type III SSCP Matrix for Time*Drug*Depleted
E = Error SSCP Matrix

S=1 M=0.5 N=3.5

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.19383010	12.48	3	9	0.0015
Pillai's Trace	0.80616990	12.48	3	9	0.0015
Hotelling-Lawley Trace	4.15915732	12.48	3	9	0.0015
Roy's Greatest Root	4.15915732	12.48	3	9	0.0015

Output 53.7.4 displays tests of hypotheses for between-subject (between-dog) effects. This section tests the hypotheses that the different Drugs, Depleteds, and their interactions have no effects on the dependent variables, while ignoring the within-dog effects. From this analysis, there is a significant between-dog effect for Depleted (p-value=0.0229). The interaction and the main effect for Drug are not significant (p-values=0.1734 and 0.1281, respectively).

Output 53.7.4 Tests of Between-Subject Effects

The GLM Procedure
Repeated Measures Analysis of Variance
Tests of Hypotheses for Between Subjects Effects

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Drug	1	5.99336243	5.99336243	2.71	0.1281
Depleted	1	15.44840703	15.44840703	6.98	0.0229
Drug*Depleted	1	4.69087508	4.69087508	2.12	0.1734
Error	11	24.34683348	2.21334850		

Univariate analyses for within-subject (within-dog) effects and related interactions are displayed in Output 53.7.6. The results for this example are the same as for the multivariate analyses; this is not always the case. In addition, before the univariate analyses are used to make conclusions about the data, the result of the sphericity test (requested with the PRINTE option in the REPEATED statement and displayed in Output 53.7.5) should be examined. If the sphericity test is rejected, consider using the adjusted G-G or H-F-L probabilities. See the section "Repeated Measures Analysis of Variance" on page 4249 for more information.

Output 53.7.5 Sphericity Test

Sphericity Tests

Variables	DF	Mauchly's Criterion	Chi-Square	Pr > ChiSq
Transformed Variates	5	0.1752641	16.930873	0.0046
Orthogonal Components	5	0.1752641	16.930873	0.0046

Output 53.7.6 Univariate Tests of Within-Subject Effects

The GLM Procedure
Repeated Measures Analysis of Variance
Univariate Tests of Hypotheses for Within Subject Effects

						Adj Pr > F	
Source	DF	Type III SS	Mean Square	F Value	Pr > F	G - G	H-F-L
Time	3	12.05898677	4.01966226	53.44	<.0001	<.0001	<.0001
Time*Drug	3	1.84429514	0.61476505	8.17	0.0003	0.0039	0.0023
Time*Depleted	3	12.08978557	4.02992852	53.57	<.0001	<.0001	<.0001
Time*Drug*Depleted	3	2.93077939	0.97692646	12.99	<.0001	0.0005	0.0002
Error(Time)	33	2.48238887	0.07522391				

Greenhouse-Geisser Epsilon	0.5694
Huynh-Feldt-Lecoutre Epsilon	0.6636

Output 53.7.7 is produced by the SUMMARY option in the REPEATED statement. If the POLYNOMIAL option is not used, a similar table is displayed using the default CONTRAST transformation. The linear, quadratic, and cubic trends for Time, labeled as 'Time_1', 'Time_2', and 'Time_3', are displayed, and in each case, the Source labeled 'Mean' gives a test for the respective trend.

Output 53.7.7 Tests of Between-Subject Effects for Transformed Variables

The GLM Procedure
Repeated Measures Analysis of Variance
Analysis of Variance of Contrast Variables

Time_N represents the nth degree polynomial contrast for Time

Contrast Variable: Time_1

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Mean	1	2.00963483	2.00963483	34.99	0.0001
Drug	1	1.18069076	1.18069076	20.56	0.0009
Depleted	1	1.36172504	1.36172504	23.71	0.0005
Drug*Depleted	1	2.04346848	2.04346848	35.58	<.0001
Error	11	0.63171161	0.05742833		

Contrast Variable: Time_2

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Mean	1	5.40988418	5.40988418	57.15	<.0001
Drug	1	0.59173192	0.59173192	6.25	0.0295
Depleted	1	5.94945506	5.94945506	62.86	<.0001
Drug*Depleted	1	0.67031587	0.67031587	7.08	0.0221
Error	11	1.04118707	0.09465337		

Contrast Variable: Time_3

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Mean	1	4.63946776	4.63946776	63.04	<.0001
Drug	1	0.07187246	0.07187246	0.98	0.3443
Depleted	1	4.77860547	4.77860547	64.94	<.0001
Drug*Depleted	1	0.21699504	0.21699504	2.95	0.1139
Error	11	0.80949018	0.07359002		

Example 53.8: Mixed Model Analysis of Variance with the RANDOM Statement

(View the complete code for this example (glmex8.sas) in the example repository.)

Milliken and Johnson (1984) present an example of an unbalanced mixed model. Three machines, which are considered as a fixed effect, and six employees, which are considered a random effect, are studied. Each employee operates each machine for either one, two, or three different times. The dependent variable is an overall rating, which takes into account the number and quality of components produced.

The following statements form the data set and perform a mixed model analysis of variance by requesting the TEST option in the RANDOM statement. Note that the machine*person interaction is declared as a random effect; in general, when an interaction involves a random effect, it too should be declared as random. The results of the analysis are shown in Output 53.8.1 through Output 53.8.4.

```
data machine;
   input machine person rating @@;
   datalines;
1 1 52.0  1 2 51.8  1 2 52.8  1 3 60.0  1 4 51.1  1 4 52.3  1 5 50.9
1 5 51.8  1 5 51.4  1 6 46.4  1 6 44.8  1 6 49.2  2 1 64.0  2 2 59.7
2 2 60.0  2 2 59.0  2 3 68.6  2 3 65.8  2 4 63.2  2 4 62.8  2 4 62.2
2 5 64.8  2 5 65.0  2 6 43.7  2 6 44.2  2 6 43.0  3 1 67.5  3 1 67.2
3 1 66.9  3 2 61.5  3 2 61.7  3 2 62.3  3 3 70.8  3 3 70.6  3 3 71.0
3 4 64.1  3 4 66.2  3 4 64.0  3 5 72.1  3 5 72.0  3 5 71.1  3 6 62.0
3 6 61.4  3 6 60.5
;

proc glm data=machine;
   class machine person;
   model rating=machine person machine*person;
   random person machine*person / test;
run;
```

The TEST option in the RANDOM statement requests that PROC GLM determine the appropriate F tests based on person and machine*person being treated as random effects. As you can see in Output 53.8.4, this requires that a linear combination of mean squares be constructed to test both the machine and person hypotheses; thus, F tests that use Satterthwaite approximations are needed.

Output 53.8.1 Summary Information about Groups

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
machine	3	1 2 3
person	6	1 2 3 4 5 6

Number of Observations Read	44
Number of Observations Used	44

Output 53.8.2 Fixed-Effect Model Analysis of Variance

The GLM Procedure

Dependent Variable: rating

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	17	3061.743333	180.102549	206.41	<.0001
Error	26	22.686667	0.872564		
Corrected Total	43	3084.430000			

R-Square	Coeff Var	Root MSE	rating Mean
0.992645	1.560754	0.934111	59.85000

Output 53.8.2 *continued*

Source	DF	Type I SS	Mean Square	F Value	Pr > F
machine	2	1648.664722	824.332361	944.72	<.0001
person	5	1008.763583	201.752717	231.22	<.0001
machine*person	10	404.315028	40.431503	46.34	<.0001

Source	DF	Type III SS	Mean Square	F Value	Pr > F
machine	2	1238.197626	619.098813	709.52	<.0001
person	5	1011.053834	202.210767	231.74	<.0001
machine*person	10	404.315028	40.431503	46.34	<.0001

Output 53.8.3 Expected Values of Type III Mean Squares

Source	Type III Expected Mean Square
machine	Var(Error) + 2.137 Var(machine*person) + Q(machine)
person	Var(Error) + 2.2408 Var(machine*person) + 6.7224 Var(person)
machine*person	Var(Error) + 2.3162 Var(machine*person)

Output 53.8.4 Mixed Model Analysis of Variance

The GLM Procedure
Tests of Hypotheses for Mixed Model Analysis of Variance

Dependent Variable: rating

Source	DF	Type III SS	Mean Square	F Value	Pr > F
machine	2	1238.197626	619.098813	16.57	0.0007
Error	10.036	375.057436	37.370384		
Error: 0.9226*MS(machine*person) + 0.0774*MS(Error)					

Source	DF	Type III SS	Mean Square	F Value	Pr > F
person	5	1011.053834	202.210767	5.17	0.0133
Error	10.015	392.005726	39.143708		
Error: 0.9674*MS(machine*person) + 0.0326*MS(Error)					

Source	DF	Type III SS	Mean Square	F Value	Pr > F
machine*person	10	404.315028	40.431503	46.34	<.0001
Error: MS(Error)	26	22.686667	0.872564		

Note that you can also use the MIXED procedure to analyze mixed models. The following statements use PROC MIXED to reproduce the mixed model analysis of variance; the relevant part of the PROC MIXED results is shown in Output 53.8.5.

```
proc mixed data=machine method=type3;
   class machine person;
   model rating = machine;
   random person machine*person;
run;
```

Output 53.8.5 PROC MIXED Mixed Model Analysis of Variance (Partial Output)

The Mixed Procedure

Type 3 Analysis of Variance

Source	DF	Sum of Squares	Mean Square	Expected Mean Square	Error Term	Error DF
machine	2	1238.197626	619.098813	Var(Residual) + 2.137 Var(machine*person) + Q(machine)	0.9226 MS(machine*person) + 0.0774 MS(Residual)	10.036
person	5	1011.053834	202.210767	Var(Residual) + 2.2408 Var(machine*person) + 6.7224 Var(person)	0.9674 MS(machine*person) + 0.0326 MS(Residual)	10.015
machine*person	10	404.315028	40.431503	Var(Residual) + 2.3162 Var(machine*person)	MS(Residual)	26
Residual	26	22.686667	0.872564	Var(Residual)	.	.

Type 3 Analysis of Variance

Source	F Value	Pr > F
machine	16.57	0.0007
person	5.17	0.0133
machine*person	46.34	<.0001
Residual	.	.

The advantage of PROC MIXED is that it offers more versatility for mixed models; the disadvantage is that it can be less computationally efficient for large data sets. See Chapter 84, "The MIXED Procedure," for more details.

Example 53.9: Analyzing a Doubly Multivariate Repeated Measures Design

(View the complete code for this example (glmex9.sas) in the example repository.)

This example shows how to analyze a doubly multivariate repeated measures design by using PROC GLM with an IDENTITY factor in the REPEATED statement. Note that this differs from previous releases of PROC GLM, in which you had to use a MANOVA statement to get a doubly repeated measures analysis.

Two responses, Y1 and Y2, are each measured three times for each subject (pretreatment, posttreatment, and in a later follow-up). Each subject receives one of three treatments; A, B, or the control. In PROC GLM, you use a REPEATED factor of type IDENTITY to identify the different responses and another repeated factor to identify the different measurement times. The repeated measures analysis includes multivariate tests for time and treatment main effects, as well as their interactions, across responses. The following statements produce Output 53.9.1 through Output 53.9.3.

```
options ls=96;
data Trial;
   input Treatment $ Repetition PreY1 PostY1 FollowY1
                                PreY2 PostY2 FollowY2;
   datalines;
A        1  3  13  9  0  0  9
A        2  0  14 10  6  6  3
A        3  4   6 17  8  2  6
A        4  7   7 13  7  6  4
A        5  3  12 11  6 12  6
A        6 10  14  8 13  3  8
B        1  9  11 17  8 11 27
B        2  4  16 13  9  3 26
B        3  8  10  9 12  0 18
B        4  5   9 13  3  0 14
B        5  0  15 11  3  0 25
B        6  4  11 14  4  2  9
Control  1 10  12 15  4  3  7
Control  2  2   8 12  8  7 20
Control  3  4   9 10  2  0 10
Control  4 10   8  8  5  8 14
Control  5 11  11 11  1  0 11
Control  6  1   5 15  8  9 10
;

proc glm data=Trial;
   class Treatment;
   model PreY1 PostY1 FollowY1
         PreY2 PostY2 FollowY2 = Treatment / nouni;
   repeated Response 2 identity, Time 3;
run;
```

Output 53.9.1 A Doubly Multivariate Repeated Measures Design

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
Treatment	3	A B Control

Number of Observations Read	18
Number of Observations Used	18

The levels of the repeated factors are displayed in Output 53.9.2. Note that RESPONSE is 1 for all the Y1 measurements and 2 for all the Y2 measurements, while the three levels of Time identify the pretreatment, posttreatment, and follow-up measurements within each response. The multivariate tests for within-subject effects are displayed in Output 53.9.3.

Output 53.9.2 Repeated Factor Levels

The GLM Procedure
Repeated Measures Analysis of Variance

Repeated Measures Level Information						
Dependent Variable	PreY1	PostY1	FollowY1	PreY2	PostY2	FollowY2
Level of Response	1	1	1	2	2	2
Level of Time	1	2	3	1	2	3

Output 53.9.3 Within-Subject Tests

MANOVA Test Criteria and Exact F Statistics for the Hypothesis of no Response Effect
H = Type III SSCP Matrix for Response
E = Error SSCP Matrix

S=1 M=0 N=6

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.02165587	316.24	2	14	<.0001
Pillai's Trace	0.97834413	316.24	2	14	<.0001
Hotelling-Lawley Trace	45.17686368	316.24	2	14	<.0001
Roy's Greatest Root	45.17686368	316.24	2	14	<.0001

MANOVA Test Criteria and F Approximations for the Hypothesis of no Response*Treatment Effect
H = Type III SSCP Matrix for Response*Treatment
E = Error SSCP Matrix

S=2 M=-0.5 N=6

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.72215797	1.24	4	28	0.3178
Pillai's Trace	0.27937444	1.22	4	30	0.3240
Hotelling-Lawley Trace	0.38261660	1.31	4	15.818	0.3074
Roy's Greatest Root	0.37698780	2.83	2	15	0.0908

NOTE: F Statistic for Roy's Greatest Root is an upper bound.

NOTE: F Statistic for Wilks' Lambda is exact.

MANOVA Test Criteria and Exact F Statistics for the Hypothesis of no Response*Time Effect
H = Type III SSCP Matrix for Response*Time
E = Error SSCP Matrix

S=1 M=1 N=5

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.14071380	18.32	4	12	<.0001
Pillai's Trace	0.85928620	18.32	4	12	<.0001
Hotelling-Lawley Trace	6.10662362	18.32	4	12	<.0001
Roy's Greatest Root	6.10662362	18.32	4	12	<.0001

Output 53.9.3 *continued*

MANOVA Test Criteria and F Approximations for the Hypothesis of no Response*Time*Treatment Effect
H = Type III SSCP Matrix for Response*Time*Treatment
E = Error SSCP Matrix

S=2 M=0.5 N=5

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.22861451	3.27	8	24	0.0115
Pillai's Trace	0.96538785	3.03	8	26	0.0151
Hotelling-Lawley Trace	2.52557514	3.64	8	15	0.0149
Roy's Greatest Root	2.12651905	6.91	4	13	0.0033

NOTE: F Statistic for Roy's Greatest Root is an upper bound.
NOTE: F Statistic for Wilks' Lambda is exact.

The table for Response*Treatment tests for an overall treatment effect across the two responses; likewise, the tables for Response*Time and Response*Treatment*Time test for time and the treatment-by-time interaction, respectively. In this case, there is a strong main effect for time and possibly for the interaction, but not for treatment.

In previous releases (before the IDENTITY transformation was introduced), in order to perform a doubly repeated measures analysis, you had to use a MANOVA statement with a customized transformation matrix M. You might still want to use this approach to see details of the analysis, such as the univariate ANOVA for each transformed variate. The following statements demonstrate this approach by using the MANOVA statement to test for the overall main effect of time and specifying the SUMMARY option.

```
proc glm data=Trial;
   class Treatment;
   model PreY1 PostY1 FollowY1
         PreY2 PostY2 FollowY2 = Treatment / nouni;
   manova  h=intercept  m=prey1 - posty1,
                        prey1 - followy1,
                        prey2 - posty2,
                        prey2 - followy2 / summary;
run;
```

The M matrix used to perform the test for time effects is displayed in Output 53.9.4, while the results of the multivariate test are given in Output 53.9.5. Note that the test results are the same as for the Response*Time effect in Output 53.9.3.

Output 53.9.4 M Matrix to Test for Time Effect (Repeated Measure)

The GLM Procedure
Multivariate Analysis of Variance

M Matrix Describing Transformed Variables

	PreY1	PostY1	FollowY1	PreY2	PostY2	FollowY2
MVAR1	1	-1	0	0	0	0
MVAR2	1	0	-1	0	0	0
MVAR3	0	0	0	1	-1	0
MVAR4	0	0	0	1	0	-1

Output 53.9.5 Tests for Time Effect (Repeated Measure)

The GLM Procedure
Multivariate Analysis of Variance

Characteristic Roots and Vectors of: E Inverse * H, where
H = Type III SSCP Matrix for Intercept
E = Error SSCP Matrix

Variables have been transformed by the M Matrix

Characteristic Vector V'EV=1

Characteristic Root	Percent	MVAR1	MVAR2	MVAR3	MVAR4
6.10662362	100.00	-0.00157729	0.04081620	-0.04210209	0.03519437
0.00000000	0.00	0.00796367	0.00493217	0.05185236	0.00377940
0.00000000	0.00	-0.03534089	-0.01502146	-0.00283074	0.04259372
0.00000000	0.00	-0.05672137	0.04500208	0.00000000	0.00000000

MANOVA Test Criteria and Exact F Statistics for the Hypothesis of No Overall Intercept Effect
on the Variables Defined by the M Matrix Transformation
H = Type III SSCP Matrix for Intercept
E = Error SSCP Matrix

S=1 M=1 N=5

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.14071380	18.32	4	12	<.0001
Pillai's Trace	0.85928620	18.32	4	12	<.0001
Hotelling-Lawley Trace	6.10662362	18.32	4	12	<.0001
Roy's Greatest Root	6.10662362	18.32	4	12	<.0001

The SUMMARY option in the MANOVA statement creates an ANOVA table for each transformed variable as defined by the M matrix. MVAR1 and MVAR2 contrast the pretreatment measurement for Y1 with the posttreatment and follow-up measurements for Y1, respectively; MVAR3 and MVAR4 are the same contrasts for Y2. Output 53.9.6 displays these univariate ANOVA tables and shows that the contrasts are all strongly significant except for the pre-versus-post difference for Y2.

Output 53.9.6 Summary Output for the Test for Time Effect

The GLM Procedure
Multivariate Analysis of Variance

Dependent Variable: MVAR1

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Intercept	1	512.0000000	512.0000000	22.65	0.0003
Error	15	339.0000000	22.6000000		

The GLM Procedure
Multivariate Analysis of Variance

Dependent Variable: MVAR2

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Intercept	1	813.3888889	813.3888889	32.87	<.0001
Error	15	371.1666667	24.7444444		

Output 53.9.6 *continued*

The GLM Procedure
Multivariate Analysis of Variance

Dependent Variable: MVAR3

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Intercept	1	68.0555556	68.0555556	3.49	0.0814
Error	15	292.5000000	19.5000000		

The GLM Procedure
Multivariate Analysis of Variance

Dependent Variable: MVAR4

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Intercept	1	800.0000000	800.0000000	26.43	0.0001
Error	15	454.0000000	30.2666667		

Example 53.10: Testing for Equal Group Variances

(View the complete code for this example (glmex10.sas) in the example repository.)

This example demonstrates how you can test for equal group variances in a one-way design. The data come from the University of Pennsylvania Smell Identification Test (UPSIT), reported in O'Brien and Heft (1995). The study is undertaken to explore how age and gender are related to sense of smell. A total of 180 subjects 20 to 89 years old are exposed to 40 different odors: for each odor, subjects are asked to choose which of four words best describes the odor. The Freeman-Tukey modified arcsine transformation (Bishop, Fienberg, and Holland 1975) is applied to the proportion of correctly identified odors to arrive at an olfactory index. For the following analysis, subjects are divided into five age groups:

$$\text{agegroup} = \begin{cases} 1 & \text{if} \quad \text{age} \le 25 \\ 2 & \text{if} \quad 25 < \text{age} \le 40 \\ 3 & \text{if} \quad 40 < \text{age} \le 55 \\ 4 & \text{if} \quad 55 < \text{age} \le 70 \\ 5 & \text{if} \quad 70 < \text{age} \end{cases}$$

The following statements create a data set named upsit, containing the age group and olfactory index for each subject.

```
data upsit;
   input agegroup smell @@;
   datalines;
1 1.381  1 1.322  1 1.162  1 1.275  1 1.381  1 1.275  1 1.322
1 1.492  1 1.322  1 1.381  1 1.162  1 1.013  1 1.322  1 1.322
1 1.275  1 1.492  1 1.322  1 1.322  1 1.492  1 1.322  1 1.381
1 1.234  1 1.162  1 1.381  1 1.381  1 1.381  1 1.322  1 1.381
1 1.322  1 1.381  1 1.275  1 1.492  1 1.275  1 1.322  1 1.275
1 1.381  1 1.234  1 1.105
```

```
2 1.234  2 1.234  2 1.381  2 1.322  2 1.492  2 1.234  2 1.381
2 1.381  2 1.492  2 1.492  2 1.275  2 1.492  2 1.381  2 1.492
2 1.322  2 1.275  2 1.275  2 1.275  2 1.322  2 1.492  2 1.381
2 1.322  2 1.492  2 1.196  2 1.322  2 1.275  2 1.234  2 1.322
2 1.098  2 1.322  2 1.381  2 1.275  2 1.492  2 1.492  2 1.381
2 1.196
3 1.381  3 1.381  3 1.492  3 1.492  3 1.492  3 1.098  3 1.492
3 1.381  3 1.234  3 1.234  3 1.129  3 1.069  3 1.234  3 1.322
3 1.275  3 1.230  3 1.234  3 1.234  3 1.322  3 1.322  3 1.381
4 1.322  4 1.381  4 1.381  4 1.322  4 1.234  4 1.234  4 1.234
4 1.381  4 1.322  4 1.275  4 1.275  4 1.492  4 1.234  4 1.098
4 1.322  4 1.129  4 0.687  4 1.322  4 1.322  4 1.234  4 1.129
4 1.492  4 0.810  4 1.234  4 1.381  4 1.040  4 1.381  4 1.381
4 1.129  4 1.492  4 1.129  4 1.098  4 1.275  4 1.322  4 1.234
4 1.196  4 1.234  4 0.585  4 0.785  4 1.275  4 1.322  4 0.712
4 0.810
5 1.322  5 1.234  5 1.381  5 1.275  5 1.275  5 1.322  5 1.162
5 0.909  5 0.502  5 1.234  5 1.322  5 1.196  5 0.859  5 1.196
5 1.381  5 1.322  5 1.234  5 1.275  5 1.162  5 1.162  5 0.585
5 1.013  5 0.960  5 0.662  5 1.129  5 0.531  5 1.162  5 0.737
5 1.098  5 1.162  5 1.040  5 0.558  5 0.960  5 1.098  5 0.884
5 1.162  5 1.098  5 0.859  5 1.275  5 1.162  5 0.785  5 0.859
;
```

Older people are more at risk for problems with their sense of smell, and this should be reflected in significant differences in the mean of the olfactory index across the different age groups. However, many older people also have an excellent sense of smell, which implies that the older age groups should have greater variability. In order to test this hypothesis and to compute a one-way ANOVA for the olfactory index that is robust to the possibility of unequal group variances, you can use the HOVTEST and WELCH options in the MEANS statement for the GLM procedure, as shown in the following statements.

```
proc glm data=upsit;
   class agegroup;
   model smell = agegroup;
   means agegroup / hovtest welch;
run;
```

Output 53.10.1, Output 53.10.2, and Output 53.10.3 display the usual ANOVA test for equal age group means, Levene's test for equal age group variances, and Welch's test for equal age group means, respectively. The hypotheses of age effects for mean and variance of the olfactory index are both confirmed.

Output 53.10.1 Usual ANOVA Test for Age Group Differences in Mean Olfactory Index

The GLM Procedure

Dependent Variable: smell

Source	DF	Type III SS	Mean Square	F Value	Pr > F
agegroup	4	2.13878141	0.53469535	16.65	<.0001

Output 53.10.2 Levene's Test for Age Group Differences in Olfactory Variability

The GLM Procedure

Levene's Test for Homogeneity of smell Variance ANOVA of Squared Deviations from Group Means					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
agegroup	4	0.0799	0.0200	6.35	<.0001
Error	175	0.5503	0.00314		

Output 53.10.3 Welch's Test for Age Group Differences in Mean Olfactory Index

Welch's ANOVA for smell			
Source	DF	F Value	Pr > F
agegroup	4.0000	13.72	<.0001
Error	78.7489		

As discussed in "Homogeneity of Variance in One-Way Models" on page 4243, Levene's test or any other test for homogeneity of variance should not be used as a diagnostic for the assumption of equal group variances that underlies the usual analysis of variance. However, graphical diagnostics can be a useful informal tool for monitoring whether your data meet the assumptions of a GLM analysis. The following statements perform a one-way ANOVA as before, but with ODS Graphics enabled. In addition to the box plot that is produced by default, the PLOTS=DIAGNOSTICS option requests a panel of summary diagnostics for the fit. These additional plots are shown in Output 53.10.4 and Output 53.10.5.

```
ods graphics on;
proc glm data=upsit plot=diagnostics;
   class agegroup;
   model smell = agegroup;
run;
ods graphics off;
```

Output 53.10.4 Box Plot of Olfactory Index by Age Group

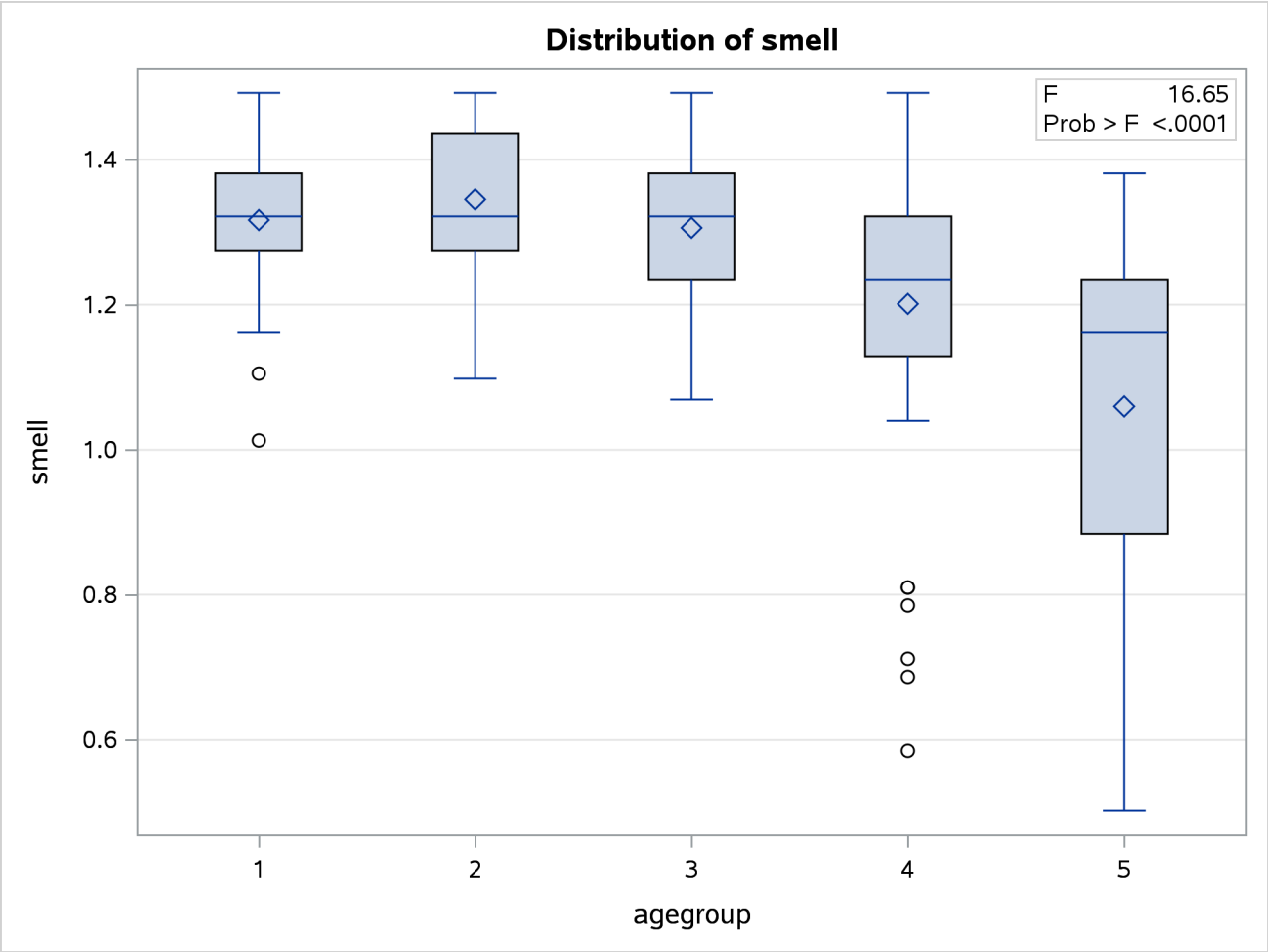

Output 53.10.5 Diagnostics for One-Way ANOVA of Olfactory Index by Age Group

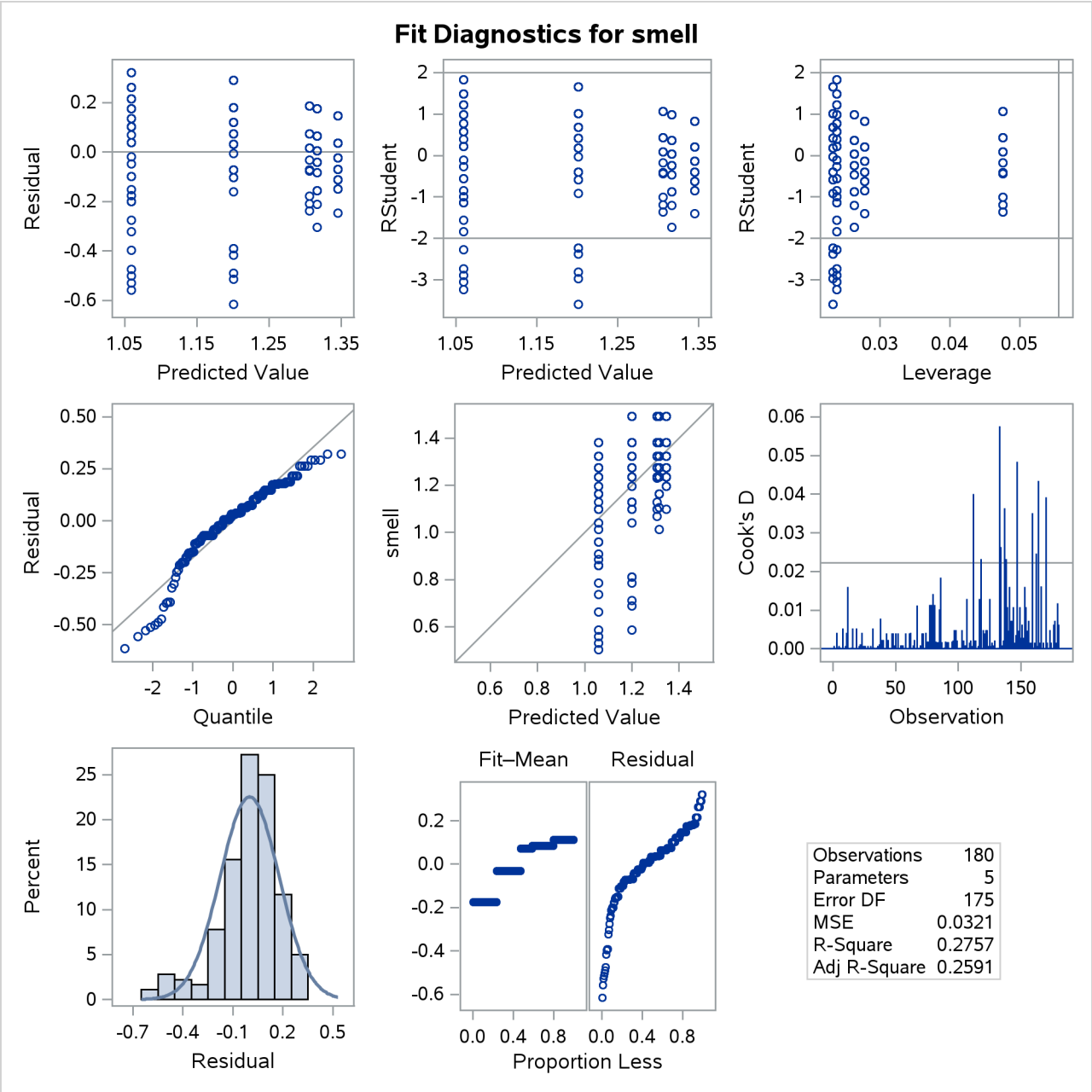

Output 53.10.4 clearly shows different degrees of variability for olfactory index within different age groups, with the variability generally rising with age. Likewise, several of the plots in the diagnostics panel shown in Output 53.10.5 indicate a relationship between olfactory variability and mean olfactory index. Also, note that the plot of Cook's D statistic indicates that observations in the higher, more variable age groups are overly influential on the analysis of group means. The overall inference from these plots is that an assumption of equal group variances is probably untenable and that the analysis of the group means should thus take this into account.

Example 53.11: Analysis of a Screening Design

(View the complete code for this example (glmex11.sas) in the example repository.)

Yin and Jillie (1987) describe an experiment performed on a nitride etch process for a single wafer plasma etcher. The experiment is run using four factors: cathode power (power), gas flow (flow), reactor chamber pressure (pressure), and electrode gap (gap). Of interest are the main effects and interaction effects of the factors on the nitride etch rate (rate). The following statements create a SAS data set named HalfFraction, containing the factor settings and the observed etch rate for each of eight experimental runs.

```
data HalfFraction;
   input power flow pressure gap rate;
   datalines;
0.8   4.5 125 275     550
0.8   4.5 200 325     650
0.8 550.0 125 325     642
0.8 550.0 200 275     601
1.2   4.5 125 325     749
1.2   4.5 200 275    1052
1.2 550.0 125 275    1075
1.2 550.0 200 325     729
;
```

Notice that each of the factors has just two values. This is a common experimental design when the intent is to screen from the many factors that *might* affect the response the few that actually *do*. Since there are $2^4 = 16$ different possible settings of four two-level factors, this design with only eight runs is called a "half fraction." The eight runs are chosen specifically to provide unambiguous information on main effects at the cost of confounding interaction effects with each other.

One way to analyze these data is simply to use PROC GLM to compute an analysis of variance, including both main effects and interactions in the model. The following statements demonstrate this approach.

```
proc glm data=HalfFraction;
   class power flow pressure gap;
   model rate=power|flow|pressure|gap@2;
run;
```

The "`@2`" notation in the MODEL statement includes all main effects and two-factor interactions between the factors. The output is shown in Output 53.11.1.

Output 53.11.1 Analysis of Variance for Nitride Etch Process Half Fraction

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
power	2	0.8 1.2
flow	2	4.5 550
pressure	2	125 200
gap	2	275 325

Number of Observations Read	8
Number of Observations Used	8

Output 53.11.1 *continued*

The GLM Procedure

Dependent Variable: rate

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	280848.0000	40121.1429	.	.
Error	0	0.0000	.		
Corrected Total	7	280848.0000			

R-Square	Coeff Var	Root MSE	rate Mean
1.000000	.	.	756.0000

Source	DF	Type I SS	Mean Square	F Value	Pr > F
power	1	168780.5000	168780.5000	.	.
flow	1	264.5000	264.5000	.	.
power*flow	1	200.0000	200.0000	.	.
pressure	1	32.0000	32.0000	.	.
power*pressure	1	1300.5000	1300.5000	.	.
flow*pressure	1	78012.5000	78012.5000	.	.
gap	1	32258.0000	32258.0000	.	.
power*gap	0	0.0000	.	.	.
flow*gap	0	0.0000	.	.	.
pressure*gap	0	0.0000	.	.	.

Source	DF	Type III SS	Mean Square	F Value	Pr > F
power	1	168780.5000	168780.5000	.	.
flow	1	264.5000	264.5000	.	.
power*flow	0	0.0000	.	.	.
pressure	1	32.0000	32.0000	.	.
power*pressure	0	0.0000	.	.	.
flow*pressure	0	0.0000	.	.	.
gap	1	32258.0000	32258.0000	.	.
power*gap	0	0.0000	.	.	.
flow*gap	0	0.0000	.	.	.
pressure*gap	0	0.0000	.	.	.

Notice that there are no error degrees of freedom. This is because there are 10 effects in the model (4 main effects plus 6 interactions) but only 8 observations in the data set. This is another cost of using a fractional design: not only is it impossible to estimate all the main effects and interactions, but there is also no information left to estimate the underlying error rate in order to measure the significance of the effects that are estimable.

Another thing to notice in Output 53.11.1 is the difference between the Type I and Type III ANOVA tables. The rows corresponding to main effects in each are the same, but no Type III interaction tests are estimable, while some Type I interaction tests are estimable. This indicates that there is *aliasing* in the design: some interactions are completely confounded with each other.

In order to analyze this confounding, you should examine the aliasing structure of the design by using the ALIASING option in the MODEL statement. Before doing so, however, it is advisable to *code* the design,

replacing low and high levels of each factor with the values –1 and +1, respectively. This puts each factor on an equal footing in the model and makes the aliasing structure much more interpretable. The following statements code the data, creating a new data set named Coded.

```
data Coded; set HalfFraction;
   power    = -1*(power   =0.80) + 1*(power   =1.20);
   flow     = -1*(flow    =4.50) + 1*(flow    =550 );
   pressure = -1*(pressure=125 ) + 1*(pressure=200 );
   gap      = -1*(gap     =275 ) + 1*(gap     =325 );
run;
```

The following statements use the GLM procedure to reanalyze the coded design, displaying the parameter estimates as well as the functions of the parameters that they each estimate.

```
proc glm data=Coded;
   model rate=power|flow|pressure|gap@2 / solution aliasing;
run;
```

The parameter estimates table is shown in Output 53.11.2.

Output 53.11.2 Parameter Estimates and Aliases for Nitride Etch Process Half Fraction

The GLM Procedure

Dependent Variable: rate

Parameter	Estimate		Standard Error	t Value	Pr > |t|	Expected Value
Intercept	756.0000000		.	.	.	Intercept
power	145.2500000		.	.	.	power
flow	5.7500000		.	.	.	flow
power*flow	-5.0000000	B	.	.	.	power*flow + pressure*gap
pressure	2.0000000		.	.	.	pressure
power*pressure	-12.7500000	B	.	.	.	power*pressure + flow*gap
flow*pressure	-98.7500000	B	.	.	.	flow*pressure + power*gap
gap	-63.5000000		.	.	.	gap
power*gap	0.0000000	B	.	.	.	
flow*gap	0.0000000	B	.	.	.	
pressure*gap	0.0000000	B	.	.	.	

In the "Expected Value" column, notice that, while each of the main effects is unambiguously estimated by its associated term in the model, the expected values of the interaction estimates are more complicated. For example, the relatively large effect (–98.75) corresponding to flow*pressure actually estimates the combined effect of flow*pressure and power*gap. Without further information, it is impossible to disentangle these aliased interactions; however, since the main effects of both power and gap are large and those for flow and pressure are small, it is reasonable to suspect that power*gap is the more "active" of the two interactions.

Fortunately, eight more runs are available for this experiment (the other half fraction). The following statements create a data set containing these extra runs and add it to the previous eight, resulting in a full $2^4 = 16$ run replicate. Then PROC GLM displays the analysis of variance again.

```
data OtherHalf;
   input power flow pressure gap rate;
   datalines;
0.8   4.5 125 325     669
0.8   4.5 200 275     604
0.8 550.0 125 275     633
0.8 550.0 200 325     635
1.2   4.5 125 275    1037
1.2   4.5 200 325     868
1.2 550.0 125 325     860
1.2 550.0 200 275    1063
;
data FullRep;
   set HalfFraction OtherHalf;
run;

proc glm data=FullRep;
   class power flow pressure gap;
   model rate=power|flow|pressure|gap@2;
run;
```

The results are displayed in Output 53.11.3.

Output 53.11.3 Analysis of Variance for Nitride Etch Process Full Replicate

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
power	2	0.8 1.2
flow	2	4.5 550
pressure	2	125 200
gap	2	275 325

Number of Observations Read	16
Number of Observations Used	16

The GLM Procedure

Dependent Variable: rate

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	10	521234.1250	52123.4125	25.58	0.0011
Error	5	10186.8125	2037.3625		
Corrected Total	15	531420.9375			

R-Square	Coeff Var	Root MSE	rate Mean
0.980831	5.816175	45.13715	776.0625

Output 53.11.3 *continued*

Source	DF	Type I SS	Mean Square	F Value	Pr > F
power	1	374850.0625	374850.0625	183.99	<.0001
flow	1	217.5625	217.5625	0.11	0.7571
power*flow	1	18.0625	18.0625	0.01	0.9286
pressure	1	10.5625	10.5625	0.01	0.9454
power*pressure	1	1.5625	1.5625	0.00	0.9790
flow*pressure	1	7700.0625	7700.0625	3.78	0.1095
gap	1	41310.5625	41310.5625	20.28	0.0064
power*gap	1	94402.5625	94402.5625	46.34	0.0010
flow*gap	1	2475.0625	2475.0625	1.21	0.3206
pressure*gap	1	248.0625	248.0625	0.12	0.7414

Source	DF	Type III SS	Mean Square	F Value	Pr > F
power	1	374850.0625	374850.0625	183.99	<.0001
flow	1	217.5625	217.5625	0.11	0.7571
power*flow	1	18.0625	18.0625	0.01	0.9286
pressure	1	10.5625	10.5625	0.01	0.9454
power*pressure	1	1.5625	1.5625	0.00	0.9790
flow*pressure	1	7700.0625	7700.0625	3.78	0.1095
gap	1	41310.5625	41310.5625	20.28	0.0064
power*gap	1	94402.5625	94402.5625	46.34	0.0010
flow*gap	1	2475.0625	2475.0625	1.21	0.3206
pressure*gap	1	248.0625	248.0625	0.12	0.7414

With 16 runs, the analysis of variance tells the whole story: all effects are estimable and there are five degrees of freedom left over to estimate the underlying error. The main effects of power and gap and their interaction are all significant, and no other effects are. Notice that the Type I and Type III ANOVA tables are the same; this is because the design is orthogonal and all effects are estimable.

This example illustrates the use of the GLM procedure for the model analysis of a screening experiment. Typically, there is much more involved in performing an experiment of this type, from selecting the design points to be studied to graphically assessing significant effects, optimizing the final model, and performing subsequent experimentation. Specialized tools for this are available in SAS/QC software, in particular the FACTEX and OPTEX procedures. See *SAS/QC User's Guide* for more information.

References

Afifi, A. A., and Azen, S. P. (1972). *Statistical Analysis: A Computer-Oriented Approach*. New York: Academic Press.

Anderson, T. W. (1958). *An Introduction to Multivariate Statistical Analysis*. New York: John Wiley & Sons.

Bartlett, M. S. (1937). "Properties of Sufficiency and Statistical Tests." *Proceedings of the Royal Society of London, Series A* 160:268–282.

Begun, J. M., and Gabriel, K. R. (1981). "Closure of the Newman-Keuls Multiple Comparisons Procedure." *Journal of the American Statistical Association* 76:374.

Belsley, D. A., Kuh, E., and Welsch, R. E. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. New York: John Wiley & Sons.

Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1975). *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, MA: MIT Press.

Box, G. E. P. (1953). "Non-normality and Tests on Variances." *Biometrika* 40:318–335.

Box, G. E. P. (1954). "Some Theorems on Quadratic Forms Applied in the Study of Analysis of Variance Problems, Part 2: Effects of Inequality of Variance and of Correlation between Errors in the Two-Way Classification." *Annals of Mathematical Statistics* 25:484–498.

Brown, M. B., and Forsythe, A. B. (1974). "Robust Tests for Equality of Variances." *Journal of the American Statistical Association* 69:364–367.

Carmer, S. G., and Swanson, M. R. (1973). "Evaluation of Ten Pairwise Multiple Comparison Procedures by Monte Carlo Methods." *Journal of the American Statistical Association* 68:66–74.

Chi, Y.-Y., Gribbin, M. J., Lamers, Y., Gregory, J. F., III, and Muller, K. E. (2012). "Global Hypothesis Testing for High-Dimensional Repeated Measures Outcomes." *Statistics in Medicine* 31:724–742.

Cochran, W. G., and Cox, G. M. (1957). *Experimental Designs*. 2nd ed. New York: John Wiley & Sons.

Cohen, J. (1988). *Statistical Power Analysis for the Behavioral Sciences*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Cohen, R. (2002). "SAS Meets Big Iron: High Performance Computing in SAS Analytical Procedures." In *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi27/p246-27.pdf.

Cole, J. W. L., and Grizzle, J. E. (1966). "Applications of Multivariate Analysis of Variance to Repeated Measures Experiments." *Biometrics* 22:810–828.

Conover, W. J., Johnson, M. E., and Johnson, M. M. (1981). "A Comparative Study of Tests for Homogeneity of Variances, with Applications to the Outer Continental Shelf Bidding Data." *Technometrics* 23:351–361.

Cornfield, J., and Tukey, J. W. (1956). "Average Values of Mean Squares in Factorials." *Annals of Mathematical Statistics* 27:907–949.

Draper, N. R., and Smith, H. (1966). *Applied Regression Analysis*. New York: John Wiley & Sons.

Duncan, D. B. (1975). "*t* Tests and Intervals for Comparisons Suggested by the Data." *Biometrics* 31:339–359.

Dunnett, C. W. (1955). "A Multiple Comparisons Procedure for Comparing Several Treatments with a Control." *Journal of the American Statistical Association* 50:1096–1121.

Dunnett, C. W. (1980). "Pairwise Multiple Comparisons in the Homogeneous Variance, Unequal Sample Size Case." *Journal of the American Statistical Association* 75:789–795.

Edwards, D., and Berry, J. J. (1987). "The Efficiency of Simulation-Based Multiple Comparisons." *Biometrics* 43:913–928.

Einot, I., and Gabriel, K. R. (1975). "A Study of the Powers of Several Methods of Multiple Comparisons." *Journal of the American Statistical Association* 70:351.

Fidler, F., and Thompson, B. (2001). "Computing Correct Confidence Intervals for ANOVA Fixed- and Random-Effects Effect Sizes." *Educational and Psychological Measurement* 61:575–604.

Freund, R. J., Littell, R. C., and Spector, P. C. (1986). *SAS System for Linear Models*. 1986 ed. Cary, NC: SAS Institute Inc.

Gabriel, K. R. (1978). "A Simple Method of Multiple Comparisons of Means." *Journal of the American Statistical Association* 73:724–729.

Games, P. A. (1977). "An Improved *t* Table for Simultaneous Control on *g* Contrasts." *Journal of the American Statistical Association* 72:531–534.

Goodnight, J. H. (1976). "General Linear Models Procedure." In *Proceedings of the First Annual SAS Users Group International Conference*, 1–39. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI76/Sugi-76-02%20Goodnight.pdf.

Goodnight, J. H. (1978a). *The SWEEP Operator: Its Importance in Statistical Computing*. Technical Report R-106, SAS Institute Inc., Cary, NC.

Goodnight, J. H. (1978b). *Tests of Hypotheses in Fixed-Effects Linear Models*. Technical Report R-101, SAS Institute Inc., Cary, NC.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Goodnight, J. H., and Harvey, W. R. (1978). *Least-Squares Means in the Fixed-Effects General Linear Models*. Technical Report R-103, SAS Institute Inc., Cary, NC.

Goodnight, J. H., and Speed, F. M. (1978). *Computing Expected Mean Squares*. Technical Report R-102, SAS Institute Inc., Cary, NC.

Graybill, F. A. (1961). *An Introduction to Linear Statistical Models.* Vol. 1. New York: McGraw-Hill.

Greenhouse, S. W., and Geisser, S. (1959). "On Methods in the Analysis of Profile Data." *Psychometrika* 32:95–112.

Gribbin, M. J. (2007). "Better Power Methods for the Univariate Approach to Repeated Measures." Ph.D. diss., Department of Biostatistics, University of North Carolina at Chapel Hill.

Guirguis, G. H., and Tobias, R. D. (2004). "On the Computation of the Distribution for the Analysis of Means." *Communications in Statistics—Simulation and Computation* 33:861–888.

Hand, D. J., Daly, F., Lunn, A. D., McConway, K. J., and Ostrowski, E. (1994). *A Handbook of Small Data Sets*. London: Chapman & Hall.

Hartley, H. O., and Searle, S. R. (1969). "On Interaction Variance Components in Mixed Models." *Biometrics* 25:573–576.

Harvey, W. R. (1975). *Least-Squares Analysis of Data with Unequal Subclass Numbers*. Technical Report ARS H-4, Agriculture Research Service, U.S. Department of Agriculture.

Hayter, A. J. (1984). "A Proof of the Conjecture That the Tukey-Kramer Method Is Conservative." *Annals of Statistics* 12:61–75.

Hayter, A. J. (1989). "Pairwise Comparisons of Generally Correlated Means." *Journal of the American Statistical Association* 84:208–213.

Heck, D. L. (1960). "Charts of Some Upper Percentage Points of the Distribution of the Largest Characteristic Root." *Annals of Mathematical Statistics* 31:625–642.

Hochberg, Y. (1974). "Some Conservative Generalizations of the T-Method in Simultaneous Inference." *Journal of Multivariate Analysis* 4:224–234.

Hocking, R. R. (1973). "A Discussion of the Two-Way Mixed Model." *American Statistician* 27:148–152.

Hocking, R. R. (1976). "The Analysis and Selection of Variables in a Linear Regression." *Biometrics* 32:1–50.

Hocking, R. R. (1985). *The Analysis of Linear Models*. Monterey, CA: Brooks/Cole.

Hsu, J. C. (1992). "The Factor Analytic Approach to Simultaneous Inference in the General Linear Model." *Journal of Computational and Graphical Statistics* 1:151–168.

Hsu, J. C. (1996). *Multiple Comparisons: Theory and Methods*. London: Chapman & Hall.

Hsu, J. C., and Nelson, B. L. (1998). "Multiple Comparisons in the General Linear Model." *Journal of Computational and Graphical Statistics* 7:23–41.

Hsu, J. C., and Peruggia, M. (1994). "Graphical Representation of Tukey's Multiple Comparison Method." *Journal of Computational and Graphical Statistics* 3:143–161.

Huynh, H., and Feldt, L. S. (1970). "Conditions Under Which Mean Square Ratios in Repeated Measurements Designs Have Exact F-Distributions." *Journal of the American Statistical Association* 65:1582–1589.

Huynh, H., and Feldt, L. S. (1976). "Estimation of the Box Correction for Degrees of Freedom from Sample Data in the Randomized Block and Split Plot Designs." *Journal of Educational Statistics* 1:69–82.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1994). *Continuous Univariate Distributions.* 2nd ed. Vol. 1. New York: John Wiley & Sons.

Kennedy, W. J., Jr., and Gentle, J. E. (1980). *Statistical Computing*. New York: Marcel Dekker.

Kramer, C. Y. (1956). "Extension of Multiple Range Tests to Group Means with Unequal Numbers of Replications." *Biometrics* 12:307–310.

Krishnaiah, P. R., and Armitage, J. V. (1966). "Tables for Multivariate *t* Distribution." *Sankhyā, Series B* 28:31–56.

Kutner, M. H. (1974). "Hypothesis Testing in Linear Models (Eisenhart Model)." *American Statistician* 28:98–100.

LaTour, S. A., and Miniard, P. W. (1983). "The Misuse of Repeated Measures Analysis in Marketing Research." *Journal of Marketing Research* 20:45–57.

Lecoutre, B. (1991). "A Correction for the Epsilon Approximate Test with Repeated Measures Design with Two or More Independent Groups." *Journal of Educational Statistics* 16:371–372.

Levene, H. (1960). "Robust Tests for the Equality of Variance." In *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, edited by I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow, and H. B. Mann, 278–292. Palo Alto, CA: Stanford University Press.

Marcus, R., Peritz, E., and Gabriel, K. R. (1976). "On Closed Testing Procedures with Special Reference to Ordered Analysis of Variance." *Biometrika* 63:655–660.

Maxwell, S. E. (2000). "Sample Size and Multiple Regression Analysis." *Psychological Methods* 5:434–458.

McLean, R. A., Sanders, W. L., and Stroup, W. W. (1991). "A Unified Approach to Mixed Linear Models." *American Statistician* 45:54–64.

Miller, R. G., Jr. (1981). *Simultaneous Statistical Inference*. New York: Springer-Verlag.

Milliken, G. A., and Johnson, D. E. (1984). *Designed Experiments.* Vol. 1 of Analysis of Messy Data. Belmont, CA: Lifetime Learning Publications.

Morrison, D. F. (1976). *Multivariate Statistical Methods*. 2nd ed. New York: McGraw-Hill.

Nelder, J. A. (1994). "The Statistics of Linear Models: Back to Basics." *Statistics and Computing* 4:221–234.

Nelson, P. R. (1982). "Exact Critical Points for the Analysis of Means." *Communications in Statistics—Theory and Methods* 11:699–709.

Nelson, P. R. (1991). "Numerical Evaluation of Multivariate Normal Integrals with Correlations $\rho_{lj} = -\alpha_l \alpha_j$." In *Frontiers of Statistical Scientific Theory and Industrial Applications: Proceedings of the ICOSCO I Conference*, edited by A. Öztürk and E. C. van der Meulen, 97–114. Columbus, OH: American Sciences Press.

Nelson, P. R. (1993). "Additional Uses for the Analysis of Means and Extended Tables of Critical Values." *Technometrics* 35:61–71.

O'Brien, R. G. (1979). "A General ANOVA Method for Robust Tests of Additive Models for Variances." *Journal of the American Statistical Association* 74:877–880.

O'Brien, R. G. (1981). "A Simple Test for Variance Effects in Experimental Designs." *Psychological Bulletin* 89:570–574.

O'Brien, R. G., and Heft, M. W. (1995). "New Discrimination Indexes and Models for Studying Sensory Functioning in Aging." *Journal of Applied Statistics* 22:9–27.

Olejnik, S. F., and Algina, J. (1987). "Type I Error Rates and Power Estimates of Selected Parametric and Non-parametric Tests of Scale." *Journal of Educational Statistics* 12:45–61.

Ott, E. R. (1967). "Analysis of Means: A Graphical Procedure." *Industrial Quality Control* 24:101–109. Reprinted in *Journal of Quality Technology* 15 (1983): 10–18.

Perlman, M. D., and Rasmussen, U. A. (1975). "Some Remarks on Estimating a Noncentrality Parameter." *Communications in Statistics—Theory and Methods* 4:455–468.

Petrinovich, L. F., and Hardyck, C. D. (1969). "Error Rates for Multiple Comparison Methods: Some Evidence Concerning the Frequency of Erroneous Conclusions." *Psychological Bulletin* 71:43–54.

Pillai, K. C. S. (1960). *Statistical Table for Tests of Multivariate Hypotheses*. Manila: University of Philippines Statistical Center.

Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.

Ramsey, P. H. (1978). "Power Differences between Pairwise Multiple Comparisons." *Journal of the American Statistical Association* 73:479–485.

Rao, C. R. (1965). *Linear Statistical Inference and Its Applications*. New York: John Wiley & Sons.

Rodriguez, R. N., Tobias, R. D., and Wolfinger, R. D. (1995). "Comments on J. A. Nelder, 'The Statistics of Linear Models: Back to Basics'." *Statistics and Computing* 5:97–101.

Ryan, T. A. (1959). "Multiple Comparisons in Psychological Research." *Psychological Bulletin* 56:26–47.

Ryan, T. A. (1960). "Significance Tests for Multiple Comparison of Proportions, Variances, and Other Statistics." *Psychological Bulletin* 57:318–328.

Satterthwaite, F. E. (1946). "An Approximate Distribution of Estimates of Variance Components." *Biometrics Bulletin* 2:110–114.

Schatzoff, M. (1966). "Exact Distributions of Wilks's Likelihood Ratio Criterion." *Biometrika* 53:347–358.

Scheffé, H. (1953). "A Method for Judging All Contrasts in the Analysis of Variance." *Biometrika* 40:87–104.

Scheffé, H. (1959). *The Analysis of Variance*. New York: John Wiley & Sons.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Searle, S. R. (1987). *Linear Models for Unbalanced Data*. New York: John Wiley & Sons.

Searle, S. R. (1995). "Comments on J. A. Nelder, 'The Statistics of Linear Models: Back to Basics'." *Statistics and Computing* 5:103–107.

Searle, S. R., Casella, G., and McCulloch, C. E. (1992). *Variance Components*. New York: John Wiley & Sons.

Searle, S. R., Speed, F. M., and Milliken, G. A. (1980). "Population Marginal Means in the Linear Model: An Alternative to Least Squares Means." *American Statistician* 34:216–221.

Šidák, Z. (1967). "Rectangular Confidence Regions for the Means of Multivariate Normal Distributions." *Journal of the American Statistical Association* 62:626–633.

Smithson, M. (2003). *Confidence Intervals*. Thousand Oaks, CA: Sage Publications.

Smithson, M. (2004). Personal communication.

Snedecor, G. W., and Cochran, W. G. (1967). *Statistical Methods*. 6th ed. Ames: Iowa State University Press.

Steel, R. G. D., and Torrie, J. H. (1960). *Principles and Procedures of Statistics*. New York: McGraw-Hill.

Steiger, J. H., and Fouladi, R. T. (1997). "Noncentrality Interval Estimation and the Evaluation of Statistical Models." In *What If There Were No Significance Tests?*, edited by L. Harlow, S. Mulaik, and J. H. Steiger, 222–257. Hillsdale, NJ: Lawrence Erlbaum Associates.

Stenstrom, F. H. (1940). "The Growth of Snapdragons, Stocks, Cinerarias, and Carnations on Six Iowa Soils." Master's thesis, Iowa State College.

Tubb, A., Parker, A. J., and Nickless, G. (1980). "The Analysis of Romano-British Pottery by Atomic Absorption Spectrophotometry." *Archaeometry* 22:153–171.

Tukey, J. W. (1952). "Allowances for Various Types of Error Rates." Invited address to Blacksburg meeting of Institute of Mathematical Studies.

Tukey, J. W. (1953). "The Problem of Multiple Comparisons." In *Multiple Comparisons, 1948–1983*, edited by H. I. Braun, vol. 8 of *The Collected Works of John W. Tukey* (published 1994), 1–300. London: Chapman & Hall. Unpublished manuscript.

Urquhart, N. S. (1968). "Computation of Generalized Inverse Matrices Which Satisfy Specific Conditions." *SIAM Review* 10:216–218.

Waller, R. A., and Duncan, D. B. (1969). "A Bayes Rule for the Symmetric Multiple Comparison Problem." *Journal of the American Statistical Association* 64:1484–1499.

Waller, R. A., and Duncan, D. B. (1972). "Corrigenda to 'A Bayes Rule for the Symmetric Multiple Comparison Problem'." *Journal of the American Statistical Association* 67:253–255.

Waller, R. A., and Kemp, K. E. (1976). "Computations of Bayesian *t*-Values for Multiple Comparisons." *Journal of Statistical Computation and Simulation* 75:169–172.

Welch, B. L. (1951). "On the Comparison of Several Mean Values: An Alternative Approach." *Biometrika* 38:330–336.

Welsch, R. E. (1977). "Stepwise Multiple Comparison Procedures." *Journal of the American Statistical Association* 72:566–575.

Westfall, P. H., and Young, S. S. (1993). *Resampling-Based Multiple Testing: Examples and Methods for* p-*Value Adjustment*. New York: John Wiley & Sons.

Winer, B. J. (1971). *Statistical Principles in Experimental Design*. 2nd ed. New York: McGraw-Hill.

Wolfinger, R. D., and Chang, M. (1995). "Comparing the SAS GLM and MIXED Procedures for Repeated Measures." In *Proceedings of the Twentieth Annual SAS Users Group Conference*, 1172–1182. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI95/Sugi-95-198%20Wolfinger%20Chang.pdf.

Yin, G. Z., and Jillie, D. W. (1987). "Orthogonal Design for Process Optimization and Its Application in Plasma Etching." *Solid State Technology* 30:127–132.

Chapter 54
The GLMMOD Procedure

Contents

Overview: GLMMOD Procedure **4332**
Getting Started: GLMMOD Procedure **4332**
A One-Way Design 4332
Syntax: GLMMOD Procedure **4336**
PROC GLMMOD Statement 4336
BY Statement 4338
CLASS Statement 4338
FREQ and WEIGHT Statements 4339
MODEL Statement 4339
Details: GLMMOD Procedure **4339**
Displayed Output 4339
Missing Values 4340
OUTPARM= Data Set 4340
OUTDESIGN= Data Set 4341
ODS Table Names 4341
Examples: GLMMOD Procedure **4342**
Example 54.1: A Two-Way Design 4342
Example 54.2: Factorial Screening 4348
References **4350**

Overview: GLMMOD Procedure

The GLMMOD procedure constructs the design matrix for a general linear model; it essentially constitutes the model-building front end for the GLM procedure. You can use the GLMMOD procedure in conjunction with other SAS/STAT software regression procedures or with SAS/IML software to obtain specialized analyses for general linear models that you cannot obtain with the GLM procedure.

While some of the regression procedures in SAS/STAT software provide for general linear effects modeling with classification variables and interaction or polynomial effects, many others do not. For such procedures, you must specify the model directly in terms of distinct variables. For example, if you want to use the REG procedure to fit a polynomial model, you must first create the crossproduct and power terms as new variables, usually in a DATA step. Alternatively, you can use the GLMMOD procedure to create a data set that contains the design matrix for a model as specified using the effects modeling facilities of the GLM procedure.

Note that the TRANSREG procedure provides alternative methods to construct design matrices for full-rank and less-than-full-rank models, polynomials, and splines. See Chapter 126, "The TRANSREG Procedure," for more information.

Getting Started: GLMMOD Procedure

A One-Way Design

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glmmodin.sas.)

A one-way analysis of variance considers one treatment factor with two or more treatment levels. This example employs PROC GLMMOD together with PROC REG to perform a one-way analysis of variance to study the effect of bacteria on the nitrogen content of red clover plants. The treatment factor is bacteria strain, and it has six levels. Red clover plants are inoculated with the treatments, and nitrogen content is later measured in milligrams. The data are derived from an experiment by Erdman (1946) and are analyzed in Chapters 7 and 8 of Steel and Torrie (1980). PROC GLMMOD is used to create the design matrix. The following DATA step creates the SAS data set Clover.

```
title 'Nitrogen Content of Red Clover Plants';
data Clover;
   input Strain $ Nitrogen @@;
   datalines;
3DOK1  19.4 3DOK1  32.6 3DOK1  27.0 3DOK1  32.1 3DOK1  33.0
3DOK5  17.7 3DOK5  24.8 3DOK5  27.9 3DOK5  25.2 3DOK5  24.3
3DOK4  17.0 3DOK4  19.4 3DOK4   9.1 3DOK4  11.9 3DOK4  15.8
3DOK7  20.7 3DOK7  21.0 3DOK7  20.5 3DOK7  18.8 3DOK7  18.6
3DOK13 14.3 3DOK13 14.4 3DOK13 11.8 3DOK13 11.6 3DOK13 14.2
COMPOS 17.3 COMPOS 19.4 COMPOS 19.1 COMPOS 16.9 COMPOS 20.8
;
```

The variable Strain contains the treatment levels, and the variable Nitrogen contains the response. The following statements produce the design matrix:

```
proc glmmod data=Clover;
   class Strain;
   model Nitrogen = Strain;
run;
```

The classification variable, or treatment factor, is specified in the CLASS statement. The MODEL statement defines the response and independent variables. The design matrix produced corresponds to the model

$$Y_{i,j} = \mu + \alpha_i + \epsilon_{i,j}$$

where $i = 1, \ldots, 6$ and $j = 1, \ldots, 5$.

Figure 54.1 and Figure 54.2 display the output produced by these statements. Figure 54.1 displays information about the data set, which is useful for checking your data.

Figure 54.1 Class Level Information and Parameter Definitions

Nitrogen Content of Red Clover Plants

The GLMMOD Procedure

Class Level Information		
Class	**Levels**	**Values**
Strain	6	3DOK1 3DOK13 3DOK4 3DOK5 3DOK7 COMPOS

Number of Observations Read	30
Number of Observations Used	30

Parameter Definitions		
Column Number	**Name of Associated Effect**	**CLASS Variable Values**
		Strain
1	Intercept	
2	Strain	3DOK1
3	Strain	3DOK13
4	Strain	3DOK4
5	Strain	3DOK5
6	Strain	3DOK7
7	Strain	COMPOS

The design matrix, shown in Figure 54.2, consists of seven columns: one for the mean and six for the treatment levels. The vector of responses, Nitrogen, is also displayed.

Figure 54.2 Design Matrix

Design Points								
		Column Number						
Observation Number	Nitrogen	1	2	3	4	5	6	7
1	19.4	1	1	0	0	0	0	0
2	32.6	1	1	0	0	0	0	0
3	27.0	1	1	0	0	0	0	0
4	32.1	1	1	0	0	0	0	0
5	33.0	1	1	0	0	0	0	0
6	17.7	1	0	0	0	1	0	0
7	24.8	1	0	0	0	1	0	0
8	27.9	1	0	0	0	1	0	0
9	25.2	1	0	0	0	1	0	0
10	24.3	1	0	0	0	1	0	0
11	17.0	1	0	0	1	0	0	0
12	19.4	1	0	0	1	0	0	0
13	9.1	1	0	0	1	0	0	0
14	11.9	1	0	0	1	0	0	0
15	15.8	1	0	0	1	0	0	0
16	20.7	1	0	0	0	0	1	0
17	21.0	1	0	0	0	0	1	0
18	20.5	1	0	0	0	0	1	0
19	18.8	1	0	0	0	0	1	0
20	18.6	1	0	0	0	0	1	0
21	14.3	1	0	1	0	0	0	0
22	14.4	1	0	1	0	0	0	0
23	11.8	1	0	1	0	0	0	0
24	11.6	1	0	1	0	0	0	0
25	14.2	1	0	1	0	0	0	0
26	17.3	1	0	0	0	0	0	1
27	19.4	1	0	0	0	0	0	1
28	19.1	1	0	0	0	0	0	1
29	16.9	1	0	0	0	0	0	1
30	20.8	1	0	0	0	0	0	1

Usually, you will find PROC GLMMOD most useful for the data sets it can create rather than for its displayed output. For example, the following statements use PROC GLMMOD to save the design matrix for the clover study to the data set CloverDesign instead of displaying it.

```
proc glmmod data=Clover outdesign=CloverDesign noprint;
   class Strain;
   model Nitrogen = Strain;
run;
```

Now you can use the REG procedure to analyze the data, as the following statements demonstrate:

```
proc reg data=CloverDesign;
   model Nitrogen = Col2-Col7;
run;
```

The results are shown in Figure 54.3.

Figure 54.3 Regression Analysis Using the REG Procedure

Nitrogen Content of Red Clover Plants

The REG Procedure
Model: MODEL1
Dependent Variable: Nitrogen

Number of Observations Read	30
Number of Observations Used	30

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	847.04667	169.40933	14.37	<.0001
Error	24	282.92800	11.78867		
Corrected Total	29	1129.97467			

Root MSE	3.43346	**R-Square**	0.7496
Dependent Mean	19.88667	**Adj R-Sq**	0.6975
Coeff Var	17.26515		

Note: Model is not full rank. Least-squares solutions for the parameters are not unique. Some statistics will be misleading. A reported DF of 0 or B means that the estimate is biased.

Note: The following parameters have been set to 0, since the variables are a linear combination of other variables as shown.

Col7 =	Intercept - Col2 - Col3 - Col4 - Col5 - Col6

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	Intercept	B	18.70000	1.53549	12.18	<.0001
Col2	Strain 3DOK1	B	10.12000	2.17151	4.66	<.0001
Col3	Strain 3DOK13	B	-5.44000	2.17151	-2.51	0.0194
Col4	Strain 3DOK4	B	-4.06000	2.17151	-1.87	0.0738
Col5	Strain 3DOK5	B	5.28000	2.17151	2.43	0.0229
Col6	Strain 3DOK7	B	1.22000	2.17151	0.56	0.5794
Col7	Strain COMPOS	0	0	.	.	.

Syntax: GLMMOD Procedure

The following statements are available in the GLMMOD procedure.

```
PROC GLMMOD <options> ;
    BY variables ;
    CLASS variables ;
    FREQ variable ;
    MODEL dependents = independents </ options> ;
    WEIGHT variable ;
```

The PROC GLMMOD and MODEL statements are required. If classification effects are used, the classification variables must be declared in a CLASS statement, and the CLASS statement must appear before the MODEL statement.

PROC GLMMOD Statement

```
PROC GLMMOD <options> ;
```

The PROC GLMMOD statement invokes the GLMMOD procedure. Table 54.1 summarizes the options available in the PROC GLMMOD statement.

Table 54.1 PROC GLMMOD Statement Options

Statement	Description
DATA=	Specifies the SAS data set to be used
NAMELEN=	Specifies the maximum length for an effect name
NOPRINT	Suppresses the normal display of results
OUTPARM=	Names an output data set describing the design matrix columns
OUTDESIGN=	Names an output data set to contain the columns of the design matrix
PREFIX=	Specifies a prefix to use in naming the columns of the design matrix
ZEROBASED	Modifies the numbering for the columns of the design matrix

It has the following options:

DATA=*SAS-data-set*

specifies the SAS data set to be used by the GLMMOD procedure. If you do not specify the DATA= option, the most recently created SAS data set is used.

NAMELEN=*n*

specifies the maximum length for an effect name. Effect names are listed in the table of parameter definitions and stored in the EFFNAME variable in the OUTPARM= data set. By default, $n = 20$. You can specify $20 < n \leq 200$ if 20 characters are not enough to distinguish between effects, which might be the case if the model includes a high-order interaction between variables with relatively long, similar names.

NOPRINT

suppresses the normal display of results. This option is generally useful only when one or more output data sets are being produced by the GLMMOD procedure. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUTPARM=*SAS-data-set*

names an output data set to contain the information regarding the association between model effects and design matrix columns.

OUTDESIGN=*SAS-data-set*

names an output data set to contain the columns of the design matrix.

PREFIX=*name*

specifies a prefix to use in naming the columns of the design matrix in the OUTDESIGN= data set. The default prefix is Col and the column name is formed by appending the column number to the prefix, so that by default the columns are named Col1, Col2, and so on. If you specify the ZEROBASED option, the column numbering starts at zero, so that with the default value of PREFIX= the columns of the design matrix in the OUTDESIGN= data set are named Col0, Col1, and so on.

ZEROBASED

specifies that the numbering for the columns of the design matrix in the OUTDESIGN= data set begin at 0. By default it begins at 1, so that with the default value of PREFIX= the columns of the design matrix in the OUTDESIGN= data set are named Col1, Col2, and so on. If you use the ZEROBASED option, the column names are instead Col0, Col1, and so on.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC GLMMOD to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the GLMMOD procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / **TRUNCATE** > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC GLMMOD statement.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

FREQ and WEIGHT Statements

FREQ *variable* **;**

WEIGHT *variable* **;**

FREQ and WEIGHT variables are transferred to the output data sets without change.

MODEL Statement

MODEL *dependents* **=** *independents* < / *options* > **;**

The MODEL statement names the dependent variables and independent effects. For the syntax of effects, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

You can specify the following option in the MODEL statement after a slash (/):

NOINT
requests that the intercept parameter not be included in the model.

Details: GLMMOD Procedure

Displayed Output

For each pass of the data (that is, for each BY group and for each pass required by the pattern of missing values for the dependent variables), the GLMMOD procedure displays the definitions of the columns of the design matrix along with the following:

- the number of the column
- the name of the associated effect
- the values that the classification variables take for this level of the effect

The design matrix itself is also displayed, along with the following:

- the observation number
- the dependent variable values

- the FREQ and WEIGHT values, if any
- the columns of the design matrix

Missing Values

If some variables have missing values for some observations, then PROC GLMMOD handles missing values in the same way as PROC GLM; see the section "Missing Values" on page 4260 in Chapter 53, "The GLM Procedure," for further details.

OUTPARM= Data Set

An output data set containing information regarding the association between model effects and design matrix columns is created whenever you specify the OUTPARM= option in the PROC GLMMOD statement. The OUTPARM= data set contains an observation for each column of the design matrix with the following variables:

- a numeric variable, _COLNUM_, identifying the number of the column of the design matrix corresponding to this observation
- a character variable, EFFNAME, containing the name of the effect that generates the column of the design matrix corresponding to this observation
- the CLASS variables, with the values they have for the column corresponding to this observation, or blanks if they are not involved with the effect associated with this column

If there are BY-group variables or if the pattern of missing values for the dependent variables requires it, the single data set defines several design matrices. In this case, for each of these design matrices, the OUTPARM= data set also contains the following:

- the current values of the BY variables, if you specify a BY statement
- a numeric variable, _YPASS_, containing the current pass of the data, if the pattern of missing values for the dependent variables requires multiple passes

OUTDESIGN= Data Set

An output data set containing the design matrix is created whenever you specify the OUTDESIGN= option in the PROC GLMMOD statement. The OUTDESIGN= data set contains an observation for each observation in the DATA= data set, with the following variables:

- the dependent variables
- the FREQ variable, if any
- the WEIGHT variable, if any
- a variable for each column of the design matrix, with names COL1, COL2, and so forth

If there are BY-group variables or if the pattern of missing values for the dependent variables requires it, the single data set contains several design matrices. In this case, for each of these, the OUTDESIGN= data set also contains the following:

- the current values of the BY variables, if you specify a BY statement
- a numeric variable, _YPASS_, containing the current pass of the data, if the pattern of missing values for the dependent variables requires multiple passes

ODS Table Names

PROC GLMMOD assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 54.2 ODS Tables Produced by PROC GLMMOD

ODS Table Name	Description	Statement
ClassLevels	Table of class levels	CLASS statement
DependentInfo	Simultaneously analyzed dependent variables	Default when there are multiple dependent variables
DesignPoints	Design matrix	Default
NObs	Number of observations	Default
Parameters	Parameters and associated column numbers	Default

Examples: GLMMOD Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 54.1: A Two-Way Design

(View the complete code for this example (glmmodx1.sas) in the example repository.)

The following program uses the GLMMOD procedure to produce the design matrix for a two-way design. The two classification factors have seven and three levels, respectively, so the design matrix contains $1 + 7 + 3 + 21 = 32$ columns in all. Output 54.1.1, Output 54.1.2, and Output 54.1.3 display the output produced by the following statements.

```
data Plants;
   input Type $ @;
   do Block=1 to 3;
      input StemLength @;
      output;
   end;
   datalines;
Clarion  32.7 32.3 31.5
Clinton  32.1 29.7 29.1
Knox     35.7 35.9 33.1
O'Neill  36.0 34.2 31.2
Compost  31.8 28.0 29.2
Wabash   38.2 37.8 31.9
Webster  32.5 31.1 29.7
;

proc glmmod data=Plants outparm=Parm outdesign=Design;
   class Type Block;
   model StemLength = Type|Block;
run;

proc print data=Parm;
run;

proc print data=Design;
run;
```

Output 54.1.1 A Two-Way Design

The GLMMOD Procedure

Class Level Information

Class	Levels	Values
Type	7	Clarion Clinton Compost Knox O'Neill Wabash Webster
Block	3	1 2 3

Output 54.1.1 *continued*

Number of Observations Read	21
Number of Observations Used	21

Parameter Definitions

Column Number	Name of Associated Effect	CLASS Variable Values: Type	CLASS Variable Values: Block
1	Intercept		
2	Type	Clarion	
3	Type	Clinton	
4	Type	Compost	
5	Type	Knox	
6	Type	O'Neill	
7	Type	Wabash	
8	Type	Webster	
9	Block		1
10	Block		2
11	Block		3
12	Type*Block	Clarion	1
13	Type*Block	Clarion	2
14	Type*Block	Clarion	3
15	Type*Block	Clinton	1
16	Type*Block	Clinton	2
17	Type*Block	Clinton	3
18	Type*Block	Compost	1
19	Type*Block	Compost	2
20	Type*Block	Compost	3
21	Type*Block	Knox	1
22	Type*Block	Knox	2
23	Type*Block	Knox	3
24	Type*Block	O'Neill	1
25	Type*Block	O'Neill	2
26	Type*Block	O'Neill	3
27	Type*Block	Wabash	1
28	Type*Block	Wabash	2
29	Type*Block	Wabash	3
30	Type*Block	Webster	1
31	Type*Block	Webster	2
32	Type*Block	Webster	3

Output 54.1.1 *continued*

Design Points

Observation Number	StemLength
1	32.7
2	32.3
3	31.5
4	32.1
5	29.7
6	29.1
7	35.7
8	35.9
9	33.1
10	36.0
11	34.2
12	31.2
13	31.8
14	28.0
15	29.2
16	38.2
17	37.8
18	31.9
19	32.5
20	31.1
21	29.7

Output 54.1.1 *continued*

	Design Points																															
	Column Number																															
Observation Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
1	1	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
8	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
9	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
11	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
12	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
13	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
15	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
16	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
17	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
18	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
19	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
20	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
21	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Output 54.1.2 The OUTPARM= Data Set

Obs	_COLNUM_	EFFNAME	Type	Block
1	1	Intercept		
2	2	Type	Clarion	
3	3	Type	Clinton	
4	4	Type	Compost	
5	5	Type	Knox	
6	6	Type	O'Neill	
7	7	Type	Wabash	
8	8	Type	Webster	
9	9	Block		1
10	10	Block		2
11	11	Block		3
12	12	Type*Block	Clarion	1
13	13	Type*Block	Clarion	2
14	14	Type*Block	Clarion	3
15	15	Type*Block	Clinton	1
16	16	Type*Block	Clinton	2
17	17	Type*Block	Clinton	3
18	18	Type*Block	Compost	1
19	19	Type*Block	Compost	2
20	20	Type*Block	Compost	3
21	21	Type*Block	Knox	1
22	22	Type*Block	Knox	2
23	23	Type*Block	Knox	3
24	24	Type*Block	O'Neill	1
25	25	Type*Block	O'Neill	2
26	26	Type*Block	O'Neill	3
27	27	Type*Block	Wabash	1
28	28	Type*Block	Wabash	2
29	29	Type*Block	Wabash	3
30	30	Type*Block	Webster	1
31	31	Type*Block	Webster	2
32	32	Type*Block	Webster	3

Output 54.1.3 The OUTDESIGN= Data Set

Obs	StemLength	Col1	Col2	Col3	Col4	Col5	Col6	Col7	Col8	Col9	Col10	Col11	Col12	Col13	Col14	Col15	Col16
1	32.7	1	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0
2	32.3	1	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0
3	31.5	1	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0
4	32.1	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0
5	29.7	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1
6	29.1	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
7	35.7	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
8	35.9	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
9	33.1	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
10	36.0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0
11	34.2	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0
12	31.2	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
13	31.8	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0
14	28.0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0
15	29.2	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0
16	38.2	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0
17	37.8	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
18	31.9	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
19	32.5	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
20	31.1	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0
21	29.7	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0

Obs	Col17	Col18	Col19	Col20	Col21	Col22	Col23	Col24	Col25	Col26	Col27	Col28	Col29	Col30	Col31	Col32
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
13	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Example 54.2: Factorial Screening

(View the complete code for this example (glmmodx2.sas) in the example repository.)

Screening experiments are undertaken to select from among the many possible factors that might affect a response the few that actually do, either simply (main effects) or in conjunction with other factors (interactions). One method of selecting significant factors is forward model selection, in which the model is built by successively adding the most statistically significant effects. Forward selection is an option in the REG procedure, but the REG procedure does not allow you to specify interactions directly (as the GLM procedure does, for example). You can use the GLMMOD procedure to create the screening model for a design and then use the REG procedure on the results to perform the screening.

The following statements create the SAS data set Screening, which contains the results of a screening experiment:

```
title 'PROC GLMMOD and PROC REG for Forward Selection Screening';
data Screening;
   input a b c d e y;
   datalines;
-1 -1 -1 -1  1  -6.688
-1 -1 -1  1 -1 -10.664
-1 -1  1 -1 -1  -1.459
-1 -1  1  1  1   2.042
-1  1 -1 -1 -1  -8.561
-1  1 -1  1  1  -7.095
-1  1  1 -1  1   0.553
-1  1  1  1 -1  -2.352
 1 -1 -1 -1 -1  -4.802
 1 -1 -1  1  1   5.705
 1 -1  1 -1  1  14.639
 1 -1  1  1 -1   2.151
 1  1 -1 -1  1   5.884
 1  1 -1  1 -1  -3.317
 1  1  1 -1 -1   4.048
 1  1  1  1  1  15.248
;
```

The data set contains a single dependent variable (y) and five independent factors (a, b, c, d, and e). The design is a half-fraction of the full 2^5 factorial, the precise half-fraction having been chosen to provide uncorrelated estimates of all main effects and two-factor interactions.

The following statements use the GLMMOD procedure to create a design matrix data set containing all the main effects and two-factor interactions for the preceding screening design.

```
ods output DesignPoints = DesignMatrix;
proc glmmod data=Screening;
   model y = a|b|c|d|e@2;
run;
```

Notice that the preceding statements use ODS to create the design matrix data set, instead of the OUTDESIGN= option in the PROC GLMMOD statement. The results are equivalent, but the columns of the data set produced by ODS have names that are directly related to the names of their corresponding effects.

Finally, the following statements use the REG procedure to perform forward model selection for the screening design. Two MODEL statements are used, one without the selection options (which produces the regression analysis for the full model) and one with the selection options. Output 54.2.1 and Output 54.2.2 show the results of the PROC REG analysis.

```
proc reg data=DesignMatrix;
   model y = a--d_e;
   model y = a--d_e / selection = forward
                      details   = summary
                      slentry   = 0.05;
run;
```

Output 54.2.1 PROC REG Full Model Fit

PROC GLMMOD and PROC REG for Forward Selection Screening

The REG Procedure
Model: MODEL1
Dependent Variable: y

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	15	861.48436	57.43229	.	.
Error	0	0	.		
Corrected Total	15	861.48436			

Root MSE	.	**R-Square**	1.0000
Dependent Mean	0.33325	**Adj R-Sq**	.
Coeff Var	.		

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	Intercept	1	0.33325	.	.	.
a		1	4.61125	.	.	.
b		1	0.21775	.	.	.
a_b	a*b	1	0.30350	.	.	.
c		1	4.02550	.	.	.
a_c	a*c	1	0.05150	.	.	.
b_c	b*c	1	-0.20225	.	.	.
d		1	-0.11850	.	.	.
a_d	a*d	1	0.12075	.	.	.
b_d	b*d	1	0.18850	.	.	.
c_d	c*d	1	0.03200	.	.	.
e		1	3.45275	.	.	.
a_e	a*e	1	1.97175	.	.	.
b_e	b*e	1	-0.35625	.	.	.
c_e	c*e	1	0.30900	.	.	.
d_e	d*e	1	0.30750	.	.	.

Output 54.2.2 PROC REG Screening Results

Summary of Forward Selection								
Step	Variable Entered	Label	Number Vars In	Partial R-Square	Model R-Square	C(p)	F Value	Pr > F
1	a		1	0.3949	0.3949	.	9.14	0.0091
2	c		2	0.3010	0.6959	.	12.87	0.0033
3	e		3	0.2214	0.9173	.	32.13	0.0001
4	a_e	a*e	4	0.0722	0.9895	.	75.66	<.0001

The full model has 16 parameters (the intercept + 5 main effects + 10 interactions). These are all estimable, but since there are only 16 observations in the design, there are no degrees of freedom left to estimate error; consequently, there is no way to use the full model to test for the statistical significance of effects. However, the forward selection method chooses only four effects for the model: the main effects of factors a, c, and e, and the interaction between a and e. Using this reduced model enables you to estimate the underlying level of noise, although note that the selection method biases this estimate somewhat.

References

Erdman, L. W. (1946). "Studies to Determine If Antibiosis Occurs among Rhizobia." *Journal of the American Society of Agronomy* 38:251–258.

Steel, R. G. D., and Torrie, J. H. (1980). *Principles and Procedures of Statistics*. 2nd ed. New York: McGraw-Hill.

Chapter 55
The GLMPOWER Procedure

Contents

Overview: GLMPOWER Procedure **4352**
Getting Started: GLMPOWER Procedure **4354**
Simple Two-Way ANOVA 4354
Incorporating Contrasts, Unbalanced Designs, and Multiple Means Scenarios 4357
Syntax: GLMPOWER Procedure **4359**
PROC GLMPOWER Statement 4360
BY Statement 4361
CLASS Statement 4362
CONTRAST Statement 4362
MANOVA Statement 4363
MODEL Statement 4365
PLOT Statement 4365
POWER Statement 4370
REPEATED Statement 4377
WEIGHT Statement 4380
Details: GLMPOWER Procedure **4381**
Specifying Value Lists in the POWER Statement 4381
Number-Lists 4381
Name-Lists 4381
Keyword-Lists 4382
Sample Size Adjustment Options 4382
Error and Information Output 4382
Displayed Output 4383
ODS Table Names 4384
Computational Methods and Formulas 4385
Contrasts in Fixed-Effect Univariate Models 4385
Adjustments for Covariates in Univariate Models 4387
Contrasts in Fixed-Effect Multivariate Models 4387
ODS Graphics 4395
Examples: GLMPOWER Procedure **4396**
Example 55.1: One-Way ANOVA 4396
Example 55.2: Two-Way ANOVA with Covariate 4401
Example 55.3: Repeated Measures ANOVA 4407
References **4411**

Overview: GLMPOWER Procedure

Power and sample size analysis optimizes the resource usage and design of a study, improving chances of conclusive results with maximum efficiency. The GLMPOWER procedure performs prospective power and sample size analysis for linear models, with a variety of goals:

- determining the sample size required to get a significant result with adequate probability (power)
- characterizing the power of a study to detect a meaningful effect
- conducting what-if analyses to assess sensitivity of the power or required sample size to other factors

Here *prospective* indicates that the analysis pertains to planning for a future study. This is in contrast to *retrospective* analysis for a past study, which is not supported by this procedure.

The statistical analyses that are covered include Type III F tests and contrasts of fixed effects in univariate and multivariate linear models. For univariate models, you can specify covariates, which can be continuous or categorical. For multivariate models, you can choose among Wilks' likelihood ratio, Hotelling-Lawley trace, and Pillai's trace F tests for multivariate analysis of variance (MANOVA) and among uncorrected, Greenhouse-Geisser, Huynh-Feldt, and Box conservative F tests for the univariate approach to repeated measures. Tests and contrasts that involve random effects are not supported. For power and sample size analyses in a variety of other statistical situations, see Chapter 96, "The POWER Procedure."

Input for PROC GLMPOWER includes the following components, which are considered in study planning:

- design (including subject profiles and their allocation weights)
- statistical model and test
- between-subject contrasts of class effects
- within-subject contrasts (for multivariate models)
- significance level (alpha)
- surmised response means for subject profiles (often called "cell means")
- surmised variability (and correlation for multivariate models)
- power
- sample size

In order to identify power or sample size as the result parameter, you designate it by a missing value in the input. The procedure calculates this result value over one or more scenarios of input values for all other components.

You specify the design and the cell means by using an *exemplary data set*, a data set of artificial values that is constructed to represent the intended sampling design and the surmised response means in the underlying population. You specify the model and between-subject contrasts by using MODEL and CONTRAST statements similar to those in the GLM, ANOVA, and MIXED procedures. For multivariate models, you

specify the within-subject contrasts by using MANOVA and REPEATED statements similar to those in the GLM and MIXED procedures. You specify the remaining parameters by using the POWER statement, which is similar to analysis statements in the POWER procedure.

In addition to tabular results, PROC GLMPOWER produces graphs. You can produce the most common types of plots easily with default settings and use a variety of options for more customized graphics. For example, you can control the choice of axis variables, axis ranges, number of plotted points, mapping of graphical features (such as color, line style, symbol, and panel) to analysis parameters, and legend appearance.

If ODS Graphics is enabled, then PROC GLMPOWER uses ODS Graphics to create graphs; otherwise, traditional graphs are produced.

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

For specific information about the statistical graphics and options available with the GLMPOWER procedure, see the PLOT statement and the section "ODS Graphics" on page 4395.

The GLMPOWER procedure is one of several tools available in SAS/STAT software for power and sample size analysis. PROC POWER covers a variety of other analyses such as *t* tests, equivalence tests, confidence intervals, binomial proportions, multiple regression, one-way ANOVA, survival analysis, logistic regression, and the Wilcoxon rank-sum test. For more information about the POWER procedure, see Chapter 96, "The POWER Procedure." SAS Studio provides a user interface and implements many of the analyses that are supported in the procedures. For more information about the power and sample size steps available in SAS Studio flows, see the steps whose names end in "Statistical Power" and are documented in the "Generating Statistics" section of *SAS Studio: Working with Flows*.

The following sections of this chapter describe how to use PROC GLMPOWER and discuss the underlying statistical methodology. The section "Getting Started: GLMPOWER Procedure" on page 4354 introduces PROC GLMPOWER with examples of power computation for a two-way analysis of variance. The section "Syntax: GLMPOWER Procedure" on page 4359 describes the syntax of the procedure. The section "Details: GLMPOWER Procedure" on page 4381 summarizes the methods employed by PROC GLMPOWER and provides details on several special topics. The section "Examples: GLMPOWER Procedure" on page 4396 illustrates the use of the GLMPOWER procedure with several applications.

For an overview of methodology and SAS tools for power and sample size analysis, see Chapter 19, "Introduction to Power and Sample Size Analysis." For more discussion and examples for linear models, see Castelloe and O'Brien (2001); O'Brien and Shieh (1992); Muller and Benignus (1992); O'Brien and Muller (1993). For additional discussion of general power and sample size concepts, see O'Brien and Castelloe (2007); Castelloe (2000); Muller and Benignus (1992); Lenth (2001).

Getting Started: GLMPOWER Procedure

Simple Two-Way ANOVA

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/gmpgs1.sas.)

This example demonstrates how to use PROC GLMPOWER to compute and plot power for each effect test in a two-way analysis of variance (ANOVA).

Suppose you are planning an experiment to study the effect of light exposure at three levels on the growth of two varieties of flowers. The planned data analysis is a two-way ANOVA with flower height (measured at two weeks) as the response and a model consisting of the effects of light exposure, flower variety, and their interaction. You want to calculate the power of each effect test for a balanced design with a total of 60 specimens (10 for each combination of exposure and variety) with $\alpha = 0.05$ for each test.

As a first step, create an *exemplary data set* describing your conjectures about the underlying population means. You believe that the mean flower height for each combination of variety and exposure level (that is, for each design profile, or for each *cell* in the design) roughly follows Table 55.1.

Table 55.1 Mean Flower Height (in cm) by Variety and Exposure

	Exposure		
Variety	**1**	**2**	**3**
1	14	16	21
2	10	15	16

The following statements create a data set named Exemplary containing these cell means.

```
data Exemplary;
   do Variety = 1 to 2;
      do Exposure = 1 to 3;
         input Height @@;
         output;
      end;
   end;
   datalines;
      14 16 21
      10 15 16
;
```

You also conjecture that the error standard deviation is about 5 cm.

Use the DATA= option in the PROC GLMPOWER statement to specify Exemplary as the exemplary data set. Identify the classification variables (Variety and Exposure) by using the CLASS statement. Specify the model by using the MODEL statement. Use the POWER statement to specify power as the result parameter and provide values for the other analysis parameters, error standard deviation and total sample size. The following SAS statements perform the power analysis:

```
proc glmpower data=Exemplary;
   class Variety Exposure;
   model Height = Variety | Exposure;
   power
      stddev = 5
      ntotal = 60
      power  = .;
run;
```

The MODEL statement defines the full model including both main effects and the interaction. The POWER= option in the POWER statement identifies power as the result parameter with a missing value (POWER=.). The STDDEV= option specifies an error standard deviation of 5, and the NTOTAL= option specifies a total sample size of 60. The default value for the ALPHA= option sets the significance level to $\alpha = 0.05$.

Figure 55.1 shows the output.

Figure 55.1 Sample Size Analysis for Two-Way ANOVA

The GLMPOWER Procedure

Fixed Scenario Elements	
Dependent Variable	Height
Error Standard Deviation	5
Total Sample Size	60
Alpha	0.05
Error Degrees of Freedom	54

Computed Power			
Index	**Source**	**Test DF**	**Power**
1	Variety	1	0.718
2	Exposure	2	0.957
3	Variety*Exposure	2	0.191

The power is about 0.72 for the test of the Variety effect. In other words, there is a probability of 0.72 that the test of the Variety effect will produce a significant result (given the assumptions for the means and error standard deviation). The power is 0.96 for the test of the Exposure effect and 0.19 for the interaction test.

Now, suppose you want to account for some of your uncertainty in conjecturing the true error standard deviation by evaluating the power at reasonable low and high values, 4 and 6.5. You also want to plot power for sample sizes between 30 and 90. The following statements perform the analysis:

```
ods graphics on;

proc glmpower data=Exemplary;
   class Variety Exposure;
   model Height = Variety | Exposure;
   power
      stddev = 4 6.5
      ntotal = 60
      power  = .;
   plot x=n min=30 max=90;
```

```
run;

ods graphics off;
```

The PLOT statement with the X=N option requests a plot with sample size on the X axis. (The result parameter—in this case, power—is always plotted on the other axis.) The MIN= and MAX= options in the PLOT statement specify the sample size range. The ODS GRAPHICS ON statement enables ODS Graphics.

Figure 55.2 shows the output, and Figure 55.3 shows the plot.

Figure 55.2 Sample Size Analysis for Two-Way ANOVA with Input Ranges

The GLMPOWER Procedure

Fixed Scenario Elements	
Dependent Variable	Height
Total Sample Size	60
Alpha	0.05
Error Degrees of Freedom	54

Computed Power				
Index	**Source**	**Std Dev**	**Test DF**	**Power**
1	Variety	4.0	1	0.887
2	Variety	6.5	1	0.496
3	Exposure	4.0	2	0.996
4	Exposure	6.5	2	0.793
5	Variety*Exposure	4.0	2	0.280
6	Variety*Exposure	6.5	2	0.130

Figure 55.2 reveals that the power ranges from about 0.130 to 0.996 for the different effect tests and scenarios for standard deviation, with a sample size of 60. In Figure 55.3, the line style identifies the effect test, and the plotting symbol identifies the standard deviation. The locations of the plotting symbols identify actual computed powers; the curves are linear interpolations of these points. Note that the computed points in the plot occur at sample size multiples of 6, because there are 6 cells in the design (and by default, sample sizes are rounded to produce integer cell sizes).

Figure 55.3 Plot of Power versus Sample Size for Two-Way ANOVA with Input Ranges

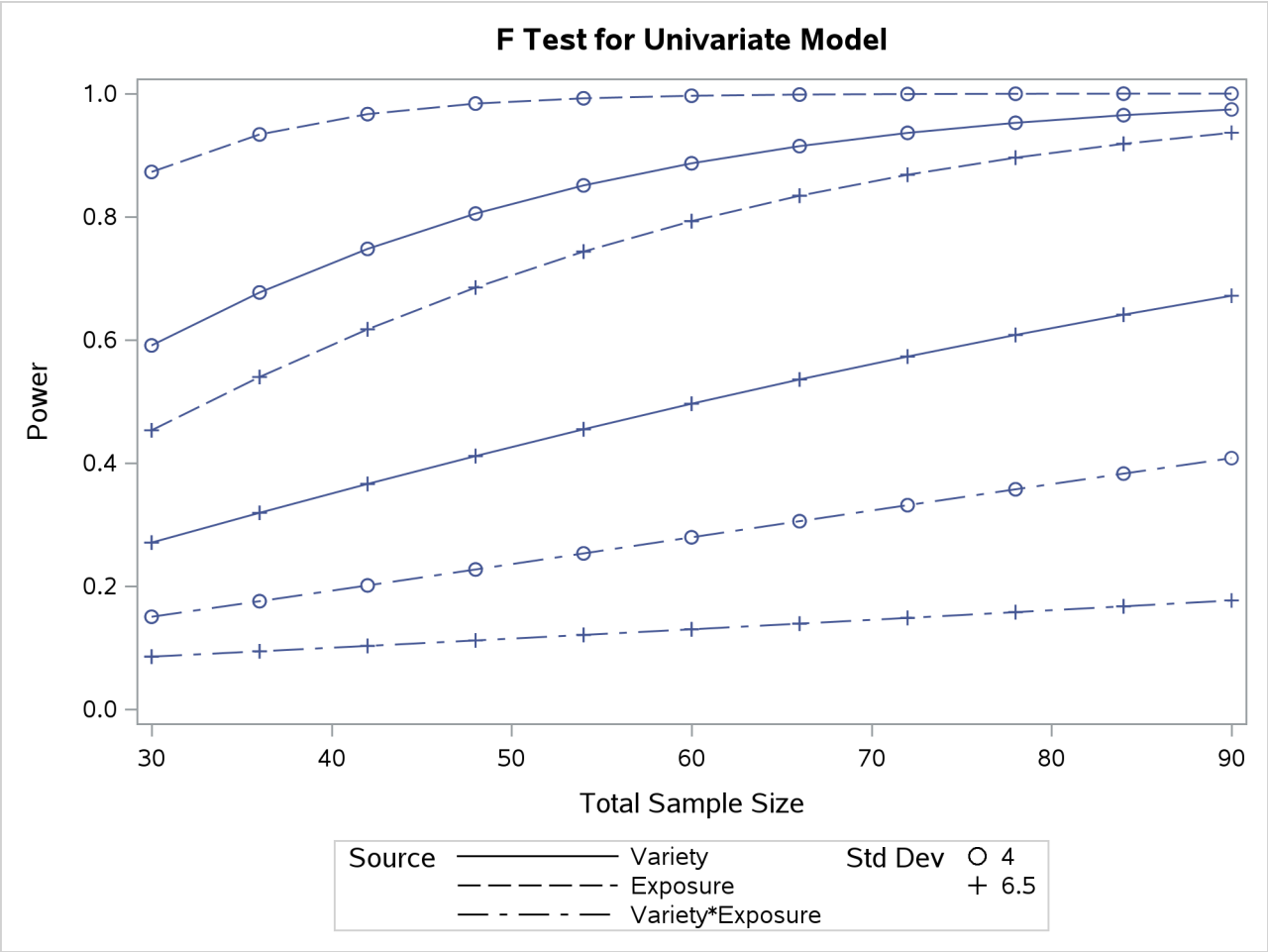

Incorporating Contrasts, Unbalanced Designs, and Multiple Means Scenarios

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/gmpgs2.sas`.)

Suppose you want to compute power for the two-way ANOVA described in the section "Simple Two-Way ANOVA" on page 4354, but you want to additionally perform the following tasks:

- try an unbalanced sample size allocation with respect to Exposure, using twice as many samples for levels 2 and 3 as for level 1
- consider an additional, less optimistic scenario for the cell means, shown in Table 55.2
- test a contrast of Exposure comparing levels 1 and 3

Table 55.2 Additional Cell Means Scenario

	Exposure		
Variety	**1**	**2**	**3**
1	15	16	20
2	11	14	15

To specify the unbalanced design and the additional cell means scenario, you can add two new variables to the exemplary data set (Weight for the sample size weights, and HeightNew for the new cell means scenario). Change the name of the original cell means scenario to HeightOrig. The following statements define the exemplary data set:

```
data Exemplary;
   input Variety $ Exposure $ HeightOrig HeightNew Weight;
   datalines;
         1   1   14  15  1
         1   2   16  16  2
         1   3   21  20  2
         2   1   10  11  1
         2   2   15  14  2
         2   3   16  15  2
;
```

In PROC GLMPOWER, specify the name of the weight variable by using the WEIGHT statement, and specify the name of the cell means variables as dependent variables in the MODEL statement. Use the CONTRAST statement to specify the contrast as you would in PROC GLM. The following statements perform the sample size analysis.

```
proc glmpower data=Exemplary;
   class Variety Exposure;
   model HeightOrig HeightNew = Variety | Exposure;
   weight Weight;
   contrast 'Exposure=1 vs Exposure=3' Exposure 1 0 -1;
   power
      stddev = 5
      ntotal = 60
      power  = .;
run;
```

Figure 55.4 shows the output.

Figure 55.4 Sample Size Analysis for More Complex Two-Way ANOVA

The GLMPOWER Procedure

Fixed Scenario Elements	
Weight Variable	Weight
Error Standard Deviation	5
Total Sample Size	60
Alpha	0.05
Error Degrees of Freedom	54

Figure 55.4 *continued*

Computed Power					
Index	Dependent	Type	Source	Test DF	Power
1	HeightOrig	Effect	Variety	1	0.672
2	HeightOrig	Effect	Exposure	2	0.911
3	HeightOrig	Effect	Variety*Exposure	2	0.217
4	HeightOrig	Contrast	Exposure=1 vs Exposure=3	1	0.951
5	HeightNew	Effect	Variety	1	0.754
6	HeightNew	Effect	Exposure	2	0.633
7	HeightNew	Effect	Variety*Exposure	2	0.137
8	HeightNew	Contrast	Exposure=1 vs Exposure=3	1	0.705

The power of the contrast of Exposure levels 1 and 3 is about 0.95 for the original cell means scenario (HeightOrig) and only 0.71 for the new one (HeightNew). The power is higher for the test of Variety, but lower for the tests of Exposure and of Variety*Exposure for the new cell means scenario compared to the original one. Note also for the HeightOrig scenario that the power for the unbalanced design (Figure 55.4) compared to the balanced design (Figure 55.1) is slightly lower for the tests of Variety and Exposure, but slightly higher for the test of Variety*Exposure.

Syntax: GLMPOWER Procedure

The following statements are available in the GLMPOWER procedure:

```
PROC GLMPOWER <options> ;
    BY variables ;
    CLASS variables ;
    CONTRAST 'label' effect values <... effect values> </ options> ;
    MANOVA 'label' <test-options> </ detail-options> ;
    MODEL dependent-variables = independent-effects ;
    PLOT <plot-options> </ graph-options> ;
    POWER <options> ;
    REPEATED factor-specification ;
    WEIGHT variable ;
```

The PROC GLMPOWER statement, the MODEL statement, and the POWER statement are required. If your model contains classification effects, the classification variables must be listed in a CLASS statement, and the CLASS statement must appear before the MODEL statement. In addition, CONTRAST and POWER statements must appear after the MODEL statement. PLOT statements must appear after the POWER statement that defines the analysis for the plot.

If you specify one or more MANOVA or REPEATED statements, then the model is assumed to be multivariate. Otherwise, a univariate model is assumed, in which case multiple dependent variables represent cell means scenarios for a single response.

You can use multiple CONTRAST, MANOVA, REPEATED, POWER, and PLOT statements. Each CONTRAST statement defines a separate between-subject contrast. Each MANOVA or REPEATED statement

defines a separate within-subject contrast for a multivariate model. Each POWER statement produces a separate analysis and uses the information that is contained in the CLASS, MODEL, WEIGHT, CONTRAST, MANOVA, and REPEATED statements. Each PLOT statement refers to the previous POWER statement and generates a separate graph (or set of graphs).

Table 55.3 summarizes the basic functions of each statement in PROC GLMPOWER. The syntax of each statement in Table 55.3 is described in the following pages.

Table 55.3 Statements in the GLMPOWER Procedure

Statement	Description
PROC GLMPOWER	Invokes procedure and specifies exemplary data set
BY	Specifies variables to define subgroups for the analysis
CLASS	Declares classification variables
CONTRAST	Defines between-subject linear tests of model parameters
MANOVA	Defines within-subject linear tests of model parameters for multivariate models, in terms of contrast matrix coefficients
MODEL	Defines model and specifies dependent variables; for univariate models, multiple dependent variables represent cell means scenarios for a single response
PLOT	Displays graphs for preceding POWER statement
POWER	Identifies parameter to solve for and provides one or more scenarios for values of other analysis parameters
REPEATED	Defines within-subject linear tests of model parameters for multivariate models, in terms of common repeated measures transformations of the dependent variables
WEIGHT	Specifies variable for allocating sample sizes to different subject profiles

PROC GLMPOWER Statement

PROC GLMPOWER < *options* > **;**

The PROC GLMPOWER statement invokes the GLMPOWER procedure. You can specify the following options.

DATA=*SAS-data-set*
names a SAS data set to be used as the exemplary data set, which is an artificial data set constructed to represent the intended sampling design and the conjectured response means for the underlying population.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PLOTONLY
specifies that only graphical results from the PLOT statement be produced.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC GLMPOWER to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the GLMPOWER procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Because sorting the data changes the order in which PROC GLMPOWER reads observations, the sort order for the levels of the classification variables might be affected if you have also specified ORDER=DATA in the PROC GLMPOWER statement. This, in turn, affects specifications in CONTRAST statements.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* **;**

The CLASS statement names the classification variables to be used in the analysis. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

CONTRAST Statement

CONTRAST *'label' effect values < . . . effect values > < / options > ;*

The CONTRAST statement enables you to define custom Type III hypothesis tests by specifying an $\mathbf{L}$ vector or matrix for testing either the hypothesis $\mathbf{L}\boldsymbol{\beta} = 0$ (for univariate models) or the hypothesis $\mathbf{LBM} = 0$ (for multivariate models). The $\mathbf{L}$ matrix consists of one or more between-subject contrasts.

To use this feature, you must be familiar with the details of the model parameterization that PROC GLM uses. For more information, see the section "Parameterization of PROC GLM Models" on page 4210 in Chapter 53, "The GLM Procedure." All the elements of the $\mathbf{L}$ matrix can be given, or if only certain portions of the $\mathbf{L}$ matrix are given, PROC GLMPOWER constructs the remaining elements from the context (in a manner similar to that in rule 4 in the section "Construction of Least Squares Means" on page 4244 in Chapter 53, "The GLM Procedure").

There is no limit to the number of CONTRAST statements that you can specify, but they must appear after the MODEL statement. Each power analysis includes tests for all CONTRAST statements.

You can specify the following arguments:

label identifies the contrast in the output. A label is required for every contrast that is specified. Labels must be enclosed in single or double quotation marks.

effect identifies an effect that appears in the MODEL statement, or the INTERCEPT effect. You do not need to include all effects that appear in the MODEL statement.

values are constants that are elements of the $\mathbf{L}$ matrix associated with the effect.

You can specify the following option in the CONTRAST statement after a slash (/):

SINGULAR=*number*

tunes the estimability checking. If $\mathrm{ABS}(\mathbf{L} - \mathbf{LH}) > C \times$ *number* for any row in the contrast, then $\mathbf{L}$ is declared nonestimable. $\mathbf{H}$ is the $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$ matrix, and C is $\mathrm{ABS}(\mathbf{L})$ except for rows where $\mathbf{L}$ is zero, and then it is 1. The default value for the SINGULAR= option is 10^{-4}. Values for the SINGULAR= option must be between 0 and 1.

As stated previously, the CONTRAST statement enables you to define custom hypothesis tests. If the hypothesis is testable in a univariate model, then the hypothesis sum of squares, $\mathrm{SS}(H_0\colon \mathbf{L}\boldsymbol{\beta} = 0)$, is computed as

$$(\mathbf{Lb})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{Lb})$$

where $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y}$.

For testable hypotheses in a multivariate model, the usual multivariate tests are defined by using

$$\mathbf{H} = \mathbf{M}'(\mathbf{LB})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{LB})\mathbf{M}$$

where $\mathbf{B} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y}$ and $\mathbf{Y}$ is the matrix of multivariate responses or dependent variables.

The degrees of freedom associated with the hypothesis are equal to the row rank of $\mathbf{L}$. The sum of squares computed in this situation is equivalent to the sum of squares computed using an $\mathbf{L}$ matrix with any row deleted that is a linear combination of previous rows.

Multiple-degrees-of-freedom hypotheses can be specified by separating the rows of the $\mathbf{L}$ matrix with commas.

MANOVA Statement

MANOVA *'label'* < *test-options* > < / *detail-options* > **;**

If the MODEL statement includes more than one dependent variable, you can indicate a multivariate model and define transformations of dependent variables by using the MANOVA statement.

The MANOVA statement enables you to define custom Type III hypothesis tests by specifying an $\mathbf{M}$ vector or matrix for testing the hypothesis $\mathbf{L}\boldsymbol{\beta}\mathbf{M} = 0$. The $\mathbf{L}$ matrix consists of one or more between-subject contrasts that involve the model effects, and the $\mathbf{M}$ matrix consists of one or more within-subject contrasts.

To use this feature, you must be familiar with the details of multivariate model and contrast parameterizations that are used in PROC GLM. For more information, see the sections "Multivariate Analysis of Variance" on page 4247 and "Repeated Measures Analysis of Variance" on page 4249 in Chapter 53, "The GLM Procedure." For information about the power and sample size computational methods and formulas, see the section "Contrasts in Fixed-Effect Multivariate Models" on page 4387.

You can use either the MANOVA statement or the REPEATED statement with any of the tests for multivariate models that are supported in the MTEST= option in the POWER statement. For handling repeated measures on the same experimental unit, you would usually use the REPEATED statement instead of the MANOVA statement. But you can use the MANOVA statement in repeated measures situations, in addition to situations where you have clusters or multiple outcome variables. The differences between the MANOVA and REPEATED statements are as follows:

- You can use the MANOVA statement to construct any **M** matrix, but you must specify the coefficients explicitly (except for the default identity matrix).
- You can use the REPEATED statement to specify commonly used contrasts by using keywords rather than coefficients, but you are limited to only those forms of the **M** matrix.

There is no limit to the number of MANOVA statements that you can specify. Each power analysis includes tests for all MANOVA statements.

The *label* identifies the dependent variable transformation in the output. The label serves the same purpose as the *factor-name* in the REPEATED statement, enabling you to use the MANOVA statement for tests of within-subject effects and within-subject-by-between-subject interactions. A label is required for every transformation that is specified. Labels must be enclosed in single or double quotation marks.

Test Options

You can specify the following *test-option* in the MANOVA statement:

M=*equation, . . . , equation* | **(***row-of-matrix, . . . , row-of-matrix***)**
specifies a transformation matrix for the dependent variables that are listed in the MODEL statement.

The equations in the M= specification are of the form

$$\begin{aligned} c_1 \times \textit{dependent-variable} \quad & \pm \quad c_2 \times \textit{dependent-variable} \\ \cdots \quad & \pm \quad c_n \times \textit{dependent-variable} \end{aligned}$$

where the c_i values are coefficients of the various *dependent-variables*. If the value of a given c_i is 1, it can be omitted; in other words, $1 \times Y$ is the same as Y. Equations should involve two or more dependent variables.

Alternatively, you can input the transformation matrix directly by entering the elements of the matrix, using commas to separate the rows and parentheses to surround the matrix. When you use this alternate form of input, the number of elements in each row must equal the number of dependent variables. Although these combinations actually represent the columns of the **M** matrix, they are displayed by rows.

When you include an M= specification, the tests are based on the variables that are defined by the equations in the specification, not the original dependent variables. If you omit the M= option, the tests are based on the original dependent variables in the MODEL statement. Omitting the M= option is equivalent to specifying an identity matrix, as in the following example, which assumes three dependent variables:

```
MANOVA 'Identity' M=(1 0 0,
                    0 1 0,
                    0 0 1);
```

For more examples of the M= option, see the section "Examples" on page 4184 in Chapter 53, "The GLM Procedure." The syntax and functionality of the M= option in PROC GLM are the same as in PROC GLMPOWER.

Detail Options

You can specify the following *detail-option* in the MANOVA statement after a slash (/):

ORTH
requests that the transformation matrix in the M= specification of the MANOVA statement be orthogonalized by rows before the analysis.

MODEL Statement

MODEL *dependent-variables* **=** *independent-effects* **;**

The MODEL statement names the dependent variables and independent effects. If one or more MANOVA or REPEATED statements are specified, then multiple dependent variables define a multivariate model. In the absence of the MANOVA and REPEATED statements, a univariate model is assumed, and multiple dependent variables represent different scenarios for the cell means.

The *independent-effects* can involve classification variables, continuous variables, or both. You can include main effects and interactions by using the effects notation of PROC GLM; for more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure." For any model effect that involves classification variables (main effects and interactions), the number of levels cannot exceed 32,767. If no independent effects are specified, only an intercept term is fit. You can specify only one MODEL statement, and it must appear before the POWER statement if the EFFECTS= option is specified in the POWER statement.

For a univariate model, you can account for covariates without specifying them explicitly in the model by using the NCOVARIATES= option and either the CORRXY= or PROPVARREDUCTION= option in the POWER statement. For a multivariate model, you must explicitly specify any covariates in the MODEL statement.

The values of dependent variables in the exemplary data set (the data set named by the DATA= option in the PROC GLMPOWER statement) are surmised response means across subject profiles. For a univariate model, multiple dependent variables correspond to multiple scenarios for these cell means.

The MODEL statement is required. You can specify only one MODEL statement.

PLOT Statement

PLOT < *plot-options* > < / *graph-options* > **;**

The PLOT statement produces a graph or set of graphs for the sample size analysis defined by the previous POWER statement. The *plot-options* define the plot characteristics, and the *graph-options* are like those in SAS/GRAPH software. If ODS Graphics is enabled, then the PLOT statement uses ODS Graphics to create graphs. For example:

```
ods graphics on;

proc glmpower data=Exemplary;
   class Variety Exposure;
   model Height = Variety | Exposure;
```

```
   power
      stddev = 4 6.5
      ntotal = 60
      power  = .;
   plot x=n min=30 max=90;
run;

ods graphics off;
```

Otherwise, traditional graphics are produced. For example:

```
ods graphics off;

proc glmpower data=Exemplary;
   class Variety Exposure;
   model Height = Variety | Exposure;
   power
      stddev = 4 6.5
      ntotal = 60
      power  = .;
   plot x=n min=30 max=90;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

Table 55.4 summarizes the options available in the PLOT statement.

Table 55.4 PLOT Statement Options

Option	Description
Plot Options	
INTERPOL=	Specifies the type of curve to draw
KEY=	Specifies the style of key for the plot
MARKERS=	Specifies the locations for plotting symbols
MAX=	Specifies the maximum of the range of values
MIN=	Specifies the minimum of the range of values
NPOINTS=	Specifies the number of values
STEP=	Specifies the increment between values
VARY	Specifies how plot features should be linked to varying analysis parameters
X=	Specifies a plot with the requested type of parameter on the X axis
XOPTS=	Specifies plot characteristics pertaining to the X axis
Y=	Specifies a plot with the requested type of parameter on the Y axis
YOPTS=	Specifies plot characteristics pertaining to the Y axis
Graph Options	
DESCRIPTION=	Specifies a descriptive string
NAME=	Specifies a name for the catalog entry for the plot

Options

You can specify the following *plot-options* in the PLOT statement.

INTERPOL=JOIN | NONE

specifies the type of curve to draw through the computed points. The INTERPOL=JOIN option connects computed points with straight lines. The INTERPOL=NONE option leaves computed points unconnected.

KEY=BYCURVE < (*bycurve-options*) >

KEY=BYFEATURE < (*byfeature-options*) >

KEY=ONCURVES

specifies the style of key (or "legend") for the plot. The default is KEY=BYFEATURE, which specifies a key with a column of entries for each plot feature (line style, color, and/or symbol). Each entry shows the mapping between a value of the feature and the value(s) of the analysis parameter(s) linked to that feature. The KEY=BYCURVE option specifies a key with each row identifying a distinct curve in the plot. The KEY=ONCURVES option places a curve-specific label adjacent to each curve.

You can specify the following *byfeature-options* in parentheses after the KEY=BYCURVE option.

NUMBERS=OFF | ON

specifies how the key should identify curves. If NUMBERS=OFF, then the key includes symbol, color, and line style samples to identify the curves. If NUMBERS=ON, then the key includes numbers matching numeric labels placed adjacent to the curves. The default is NUMBERS=ON.

POS=BOTTOM | INSET

specifies the position of the key. The POS=BOTTOM option places the key below the X axis. The POS=INSET option places the key inside the plotting region and attempts to choose the least crowded corner. The default is POS=BOTTOM.

You can specify the following *byfeature-options* in parentheses after KEY=BYFEATURE option.

POS=BOTTOM | INSET

specifies the position of the key. The POS=BOTTOM option places the key below the X axis. The POS=INSET option places the key inside the plotting region and attempts to choose the least crowded corner. The default is POS=BOTTOM.

MARKERS=ANALYSIS | COMPUTED | NICE | NONE

specifies the locations for plotting symbols.

The MARKERS=ANALYSIS option places plotting symbols at locations that correspond to the values of the relevant input parameter from the POWER statement preceding the PLOT statement.

The MARKERS=COMPUTED option (the default) places plotting symbols at the locations of actual computed points from the sample size analysis.

The MARKERS=NICE option places plotting symbols at tick mark locations (corresponding to the argument axis).

The MARKERS=NONE option disables plotting symbols.

MAX=*number* | **DATAMAX**

specifies the maximum of the range of values for the parameter associated with the "argument" axis (the axis that is *not* representing the parameter being solved for). The default is DATAMAX, which specifies the maximum value that occurs for this parameter in the POWER statement that precedes the PLOT statement.

MIN=*number* | **DATAMIN**

specifies the minimum of the range of values for the parameter associated with the "argument" axis (the axis that is *not* representing the parameter being solved for). The default is DATAMIN, which specifies the minimum value that occurs for this parameter in the POWER statement that precedes the PLOT statement.

NPOINTS=*number*

NPTS=*number*

specifies the number of values for the parameter associated with the "argument" axis (the axis that is *not* representing the parameter being solved for). You cannot use the NPOINTS= and STEP= options simultaneously. The default value for typical situations is 20.

STEP=*number*

specifies the increment between values of the parameter associated with the "argument" axis (the axis that is *not* representing the parameter being solved for). You cannot use the STEP= and NPOINTS= options simultaneously. By default, the NPOINTS= option is used instead of the STEP= option.

VARY (*feature* **<BY** *parameter-list* **>** **<, . . . ,** *feature* **<BY** *parameter-list* **>>** **)**

specifies how plot features should be linked to varying analysis parameters. Available *features* are COLOR, LINESTYLE, PANEL, and SYMBOL. A "panel" refers to a separate plot with a heading identifying the subset of values represented in the plot.

The *parameter-list* is a list of one or more names, separated by spaces. Each name must match the name of an analysis option used in the POWER statement preceding the PLOT statement, *or* one of the following keywords:

- SOURCE, which represents the model effects and contrasts in a univariate model and the between-subject effects and contrasts in a multivariate model
- DEPENDENT, which represents the cell means scenarios in a univariate model or the dependent variable transformations in a multivariate model

If the name represents an analysis option that is specified in the POWER statement, then it must be the *primary* name for the analysis option—that is, the one that is listed first in the syntax description.

If you omit the < BY *parameter-list* > portion for a feature, then one or more multivalued parameters from the analysis are automatically selected for you.

X=N | **POWER**

specifies a plot with the requested type of parameter on the X axis and the parameter being solved for on the Y axis. When X=N, sample size is assigned to the X axis. When X=POWER, power is assigned to the X axis. You cannot use the X= and Y= options simultaneously. The default is X=POWER, unless the result parameter is power, in which case the default is X=N.

XOPTS= (*x-options* **)**
specifies plot characteristics pertaining to the X axis.

You can specify the following *x-options* in parentheses.

CROSSREF=NO | YES
specifies whether the reference lines defined by the REF= *x-option* should be crossed with a reference line on the Y axis that indicates the solution point on the curve.

REF=*number-list*
specifies locations for reference lines extending from the X axis across the entire plotting region. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

Y=N | POWER
specifies a plot with the requested type of parameter on the Y axis and the parameter being solved for on the X axis. When Y=N, sample size is assigned to the Y axis. When Y=POWER, power is assigned to the Y axis. You cannot use the Y= and X= options simultaneously. By default, the X= option is used instead of the Y= option.

YOPTS= (*y-options* **)**
specifies plot characteristics pertaining to the Y axis.

You can specify the following *y-options* in parentheses.

CROSSREF=NO | YES
specifies whether the reference lines defined by the REF= *y-option* should be crossed with a reference line on the X axis that indicates the solution point on the curve.

REF=*number-list*
specifies locations for reference lines extending from the Y axis across the entire plotting region. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

You can specify the following *graph-options* in the PLOT statement after a slash (/).

DESCRIPTION='*string* **'**
specifies a descriptive string of up to 40 characters that appears in the "Description" field of the graphics catalog. The description does not appear on the plots. By default, PROC GLMPOWER assigns a description either of the form "*Y* versus *X*" (for a single-panel plot) or of the form "*Y* versus *X* (*S*)," where *Y* is the parameter on the Y axis, *X* is the parameter on the X axis, and *S* is a description of the subset represented on the current panel of a multipanel plot.

NAME='*string* **'**
specifies a name of up to eight characters for the catalog entry for the plot. The default name is PLOT*n*, where *n* is the number of the plot statement within the current invocation of PROC GLMPOWER. If the name duplicates the name of an existing entry, SAS/GRAPH software adds a number to the duplicate name to create a unique entry—for example, PLOT11 and PLOT12 for the second and third panels of a multipanel plot generated in the first PLOT statement in an invocation of PROC GLMPOWER.

POWER Statement

POWER < *options* > **;**

The POWER statement performs power and sample size analyses for the Type III F tests that are specified in the MTEST= option, for each effect in the model that is defined by the MODEL statement, and for the contrasts that are defined by all CONTRAST, MANOVA, and REPEATED statements. The POWER statement must appear after the MODEL statement if the EFFECTS= option is used in the POWER statement.

For information about the power and sample size computational methods and formulas, see the section "Computational Methods and Formulas" on page 4385.

Summary of Options

Table 55.5 summarizes the options available in the POWER statement.

Table 55.5 POWER Statement Options

Option	Description
Specify Test Statistic	
MTEST=	Specifies the test statistic for a multivariate model
UEPSDEF=	Specifies the form of the Huynh-Feldt epsilon for MTEST=HF
Specify Analysis Information	
ALPHA=	Specifies the level of significance of each test
EFFECTS=	Specifies the model effects
Specify Covariates for a Univariate Model	
CORRXY=	Specifies multiple correlation (ρ) between covariates and response
NCOVARIATES=	Specifies additional degrees of freedom due to covariates
PROPVARREDUCTION=	Specifies proportional variance reduction (r) due to covariates
Specify Variability	
CORRMAT=	Specifies the correlation matrix of the dependent variables in a multivariate model
CORRS=	Specifies the correlations among the dependent variables in a multivariate model
COVMAT=	Specifies the covariance matrix of the dependent variables in a multivariate model
MATRIX=	Defines a matrix or vector
SQRTVAR=	Specifies the vector of error standard deviations for each dependent variable in a multivariate model
STDDEV=	Specifies the common error standard deviation
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NTOTAL=	Specifies the sample size
Specify Power	
POWER=	Specifies power

Table 55.5 *continued*

Option	Description
Choose Computational Method	
METHOD=	Specifies the computational method for multivariate tests
Control Ordering in Output	
DEPENDENT	Specifies the location of the Dependent or Transformation column in the output
OUTPUTORDER=	Controls ordering in output

Table 55.6 summarizes the valid result parameters.

Table 55.6 Summary of Result Parameters in the POWER Statement

Solve for	Syntax
Power	POWER = .
Sample size	NTOTAL = .

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of each test. The default is 0.05,which corresponds to the usual 0.05 × 100% = 5% level of significance. Note that this is a test-wise significance level with the same value for all tests, not incorporating any corrections for multiple testing. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

CORRMAT=*name-list*

specifies the correlation matrix of the dependent variables in a multivariate model, by using labels that are specified in the MATRIX= option. The corresponding matrices that are defined in the MATRIX= option must have either a lower triangular form that *includes* the diagonal of 1's or a linear exponent autoregressive (LEAR) correlation structure. The matrix must be positive definite. You can use the CORRMAT= option only when you have a multivariate model—that is, in the presence of one or more MANOVA or REPEATED statements. For information about specifying the *name-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

CORRS=*name-list*

specifies the correlations among the dependent variables in a multivariate model, by using labels that are specified in the MATRIX= option. The corresponding matrices that are defined in the MATRIX= option must have either a lower triangular form that *excludes* the diagonal of 1's or a LEAR correlation structure. The matrix must be positive definite. You can use the CORRS= option only when you have a multivariate model—that is, in the presence of one or more MANOVA or REPEATED statements. For information about specifying the *name-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

COVMAT=*name-list*

specifies the covariance matrix of the dependent variables in a multivariate model, by using labels that are specified in the MATRIX= option. The corresponding matrices that are defined in the MATRIX= option must have a lower triangular form that includes the diagonal of error variances. The matrix must be positive definite. You can use the COVMAT= option only when you have a multivariate model—that is, in the presence of one or more MANOVA or REPEATED statements. For information about specifying the *name-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

CORRXY=*number-list*

specifies the multiple correlation (ρ) between all covariates and the response for a univariate model. The error standard deviation that is given by the STDDEV= option is consequently reduced by multiplying it by a factor of $(1-\rho^2)^{\frac{1}{2}}$, provided that the number of covariates (as determined by the NCOVARIATES= option) is greater than 0. You cannot use the CORRXY= and PROPVARREDUCTION= options simultaneously. You cannot use the CORRXY= option when you have a multivariate model—that is, in the presence of a MANOVA or REPEATED statement. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

DEPENDENT

specifies the location of the Dependent column (for a univariate model) or the Transformation column (for a multivariate model) in the output when you specify the OUTPUTORDER=REVERSE option or OUTPUTORDER=SYNTAX option, according to its relative position in the POWER statement.

EFFECTS < = < (effect . . . effect) > >

specifies the model effects to include in the power analysis. By default, or if the EFFECTS keyword is specified without the equal sign (=), all model effects are included. Specify EFFECTS=() to exclude all model effect tests from the power analysis. You can include main effects and interactions by using the effects notation of PROC GLM; for more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure." The MODEL statement must appear before the POWER statement if the EFFECTS option is used.

MATRIX('*label***')=***matrix-specification*

defines a matrix or vector that you can use along with the CORRMAT=, CORRS=, COVMAT=, and SQRTVAR= options when you have a multivariate model.

The *matrix-specification* can have one of the following three forms:

1. Raw values

(*values* **)**

specifies the values of a matrix or vector in one of the following forms:

- a matrix in lower triangular form, for use with the CORRMAT= or COVMAT= option
- a matrix in strictly lower triangular form, for use with the CORRS= option
- a vector, for use with the SQRTVAR= option

A matrix in lower triangular form contains the diagonal the and values below it. For example, you can represent a 3×3 correlation matrix for use with the CORRMAT= option as follows:

```
MATRIX ('MyCorrMat') = (  1
                         0.5    1
                         0.2  0.5  1)
```

You can represent the same correlation matrix in strictly lower triangular form for use with the CORRS= option as follows:

```
MATRIX ('MyCorrs') = (0.5
                      0.2  0.5)
```

An example of a vector for use with the SQRTVAR= option is as follows:

```
MATRIX ('MySqrtVar') = (3.2  4.5  3.7)
```

2. Linear exponent autoregressive (LEAR) correlation structure

LEAR (*base-corr*, *corr-decay* <, *nlevels* <, *level-values* >> **)**

specifies a LEAR correlation structure for use with the CORRMAT= or CORRS= option. The LEAR structure is useful for characterizing exponentially decaying within-subject correlation that decays at a rate slower or faster than AR(1). Special cases include compound symmetry, first-order autoregressive (AR(1)), and first-order moving average correlation structures.

The LEAR correlation structure is related to the spatial covariance structures in PROC MIXED and is discussed in Simpson et al. (2010).

The *base-corr* (ρ) is the correlation between variables whose *level-values* are one unit apart, and it must satisfy $0 \leq \rho < 1$. The *corr-decay* (δ) is the correlation decay rate, which must be nonnegative. The default value for the number of levels, *nlevels* (n), is the number of dependent variables. The n *level-values*, denoted $\{l_1, \ldots, l_n\}$, must be distinct. The default *level-values* are $\{1, \ldots, n\}$. Let d_{jk} denote the distance between levels j and k, ($d_{jk} = \mid l_j - l_k \mid$). Let $d_{\min} = \min(d_{jk} : j \neq k)$ and $d_{\max} = \max(d_{jk} : j \neq k)$. The (i, j) element of the correlation matrix according to the LEAR model is defined as

$$\rho_{jk} = \begin{cases} 1 & \text{if } j = k \\ \rho^{d_{\min} + \delta[(d_{jk} - d_{\min})/(d_{\max} - d_{\min})]} & \text{if } j \neq k \text{ and } d_{\min} \neq d_{\max} \\ \rho & \text{if } j \neq k \text{ and } d_{\min} = d_{\max} \end{cases}$$

Compound symmetry is the special case $\delta = 0$. AR(1) is the special case $\delta = d_{\max} - d_{\min}$. As $\delta \to \infty$, the model approaches the first-order moving average model.

3. Kronecker product

'matrix-name' @ *'matrix-name'* < @ ... @ *'matrix-name'* >

specifies a direct (Kronecker) product of two or more matrices for use with the CORRMAT=, CORRS=, or COVMAT= option. This form is useful when you have more than one type of distinction among dependent variables. For example, suppose you have a three-level repeated measurement factor with correlation that is 0.4 for neighboring measurements and that decays slightly more slowly than AR(1)

across more distant measurements. You also have four clusters that you believe satisfy compound symmetry with a correlation of 0.3. Your level values are the same as the default for the LEAR model. You can specify this correlation structure as follows:

```
MATRIX ('RepMeasures') = LEAR (0.4, 1.5, 3)
MATRIX ('Clusters') = LEAR (0.3, 0, 4)
MATRIX ('FullCorr') = 'RepMeasures' @ 'Clusters'
CORRMAT = 'FullCorr'
```

You can use the MATRIX= option only when you have a multivariate model—that is, in the presence of one or more MANOVA or REPEATED statements.

METHOD=MULLERPETERSON | MP

METHOD=OBRIENSHIEH | OS

specifies the power computation method for the multivariate tests (MTEST=HLT, MTEST=PT, and MTEST=WILKS). METHOD=OBRIENSHIEH (the default) is based on O'Brien and Shieh (1992), and METHOD=MULLERPETERSON is based on Muller and Peterson (1984). For information about the associated power and sample size computational methods and formulas, see the section "Multivariate Tests" on page 4389.

If the dependent variable transformation consists of a single contrast ($r_M = 1$), then the two methods are identical and compute exact power. If $r_M > 1$ but the model effect or between-subject contrast has only one degree of freedom ($r_L = 1$), then METHOD=OBRIENSHIEH computes exact results and METHOD=MULLERPETERSON computes approximate results. If $r_M > 1$ and $r_L > 1$, then both methods compute approximate results.

You can use the METHOD= option only when you have a multivariate model—that is, in the presence of one or more MANOVA or REPEATED statements.

MTEST=*test-list*

specifies the form of the F test for a multivariate model. Seven keywords are available, as discussed in the following paragraphs: BOX, GG, HF, HLT, PT, UNCORR, and WILKS. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

Three of these tests are multivariate, corresponding to the default MSTAT=FAPPROX option in the MANOVA and REPEATED statements in PROC GLM:

- MTEST=HLT (the default) is the Hotelling-Lawley trace
- MTEST=PT is Pillai's trace
- MTEST=WILKS is Wilks' lambda

For more information about these multivariate tests, see the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures." For information about the associated power and sample size computational methods and formulas, see the section "Multivariate Tests" on page 4389.

The other four tests are univariate, corresponding to the univariate approach to repeated measures in the REPEATED statement in PROC GLM:

- MTEST=UNCORR is the uncorrected univariate F test, assuming sphericity ($\varepsilon = 1$).

- MTEST=GG is the F test with the Greenhouse-Geisser adjustment, estimating ε by its maximum likelihood estimate $\hat{\varepsilon}$.
- MTEST=HF is the F test with the Huynh-Feldt adjustment as specified by the UEPSDEF= option, using an approximately unbiased estimate $\tilde{\varepsilon}$.
- MTEST=BOX is the F test with Box's conservative adjustment, estimating sphericity by its smallest possible value $1/r_M$, where r_M is the number of within-subject contrasts in the dependent variable transformation.

For more information about these univariate tests, see the section "Hypothesis Testing in Repeated Measures Analysis" on page 4251 in Chapter 53, "The GLM Procedure." For information about the associated power and sample size computational methods and formulas, see the section "Univariate Tests" on page 4392.

These tests are all of the form $\mathbf{L}\boldsymbol{\beta}\mathbf{M} = 0$, where $\mathbf{L}$ is a between-subject contrast, $\boldsymbol{\beta}$ is the matrix of model parameters, and $\mathbf{M}$ is a within-subject contrast.

You can use the MTEST= option only when you have a multivariate model—that is, in the presence of one or more MANOVA or REPEATED statements.

NCOVARIATES=*number-list*

NCOVARIATE=*number-list*

NCOVS=*number-list*

NCOV=*number-list*

specifies the number of additional degrees of freedom to accommodate covariate effects—both class and continuous—not listed in the MODEL statement, for a univariate model. The error degrees of freedom are consequently reduced by the value of the NCOVARIATES= option, and the error standard deviation (whose unadjusted value is provided with the STDDEV= option) is reduced according to the value of the CORRXY= or PROPVARREDUCTION= option. You cannot use the NCOVARIATES= option when you have a multivariate model—that is, in the presence of a MANOVA or REPEATED statement. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 4382 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). The term "sample size" here refers to the number of independent sampling units. Values for the sample size for a univariate model must be no smaller than the model degrees of freedom (counting the covariates if present). The minimum required sample size for a multivariate model depends on the analysis and computational method; for more information, see the section "Contrasts in Fixed-Effect Multivariate Models" on page 4387. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding options:

- DEPENDENT
- EFFECTS=
- weight variable (from the WEIGHT statement)
- ALPHA=
- NCOVARIATES=
- CORRXY=
- PROPVARREDUCTION=
- STDDEV=
- NTOTAL=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the POWER statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding options are specified in the POWER statement.

POWER=*number-list*

specifies the desired power of each test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability (for example, 0.9) rather than a percentage. Note that this is a test-wise power with the same value for all tests, without any correction for multiple testing. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

PROPVARREDUCTION=*number-list*

PVRED=*number-list*

specifies the proportional reduction (r) in total R square incurred by the covariates—in other words, the amount of additional variation explained by the covariates—for a univariate model. The error standard deviation that is given by the STDDEV= option is consequently reduced by multiplying it by a factor of $(1-r)^{\frac{1}{2}}$, provided that the number of covariates (as determined by the NCOVARIATES= option) is greater than 0. You cannot use the PROPVARREDUCTION= and CORRXY= options simultaneously. You cannot use the PROPVARREDUCTION= option when you have a multivariate model—that is, in the presence of a MANOVA or REPEATED statement. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

SQRTVAR=*name-list*

specifies the vector of standard deviations—that is, the square roots of the variances—of the dependent variables in a multivariate model, by using labels that are specified using the MATRIX= option. The standard deviation values must be positive. You can use the SQRTVAR= option only when you have a multivariate model—that is, in the presence of one or more MANOVA or REPEATED statements. For information about specifying the *name-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

STDDEV=*number-list*
specifies the error standard deviation, or root MSE. For a multivariate model, each value in the *number-list* is taken to be a common value for all dependent variables. If covariates are specified by using the NCOVARIATES= option, then the STDDEV= option denotes the error standard deviation before accounting for these covariates. For information about specifying the *number-list*, see the section "Specifying Value Lists in the POWER Statement" on page 4381.

UEPSDEF=*unbiased-epsilon-definition*
specifies the type of adjustment for MTEST=HF. The default is UEPSDEF=HFL, which corresponds to the corrected form of the Huynh-Feldt adjustment (Huynh and Feldt 1976; Lecoutre 1991). Other alternatives are UEPSDEF=HF; the uncorrected Huynh-Feldt adjustment; and UEPSDEF=CM, the adjustment of Chi et al. (2012). For more information about these adjustments, see the section "Hypothesis Testing in Repeated Measures Analysis" on page 4251 in Chapter 53, "The GLM Procedure." You can use the UEPSDEF= option only when you have a multivariate model—that is, in the presence of one or more MANOVA or REPEATED statements. For information about the associated power and sample size computational methods and formulas, see the section "Univariate Tests" on page 4392.

Restrictions on Option Combinations

To specify the variability in a multivariate model, choose one of the following parameterizations:

- covariance matrix (using the MATRIX= and COVMAT= options)
- standard deviations and correlations (using the MATRIX=, SQRTVAR=, and CORRS= options)
- common standard deviation and correlations (using the STDDEV=, MATRIX=, and CORRS= options)
- standard deviations and correlation matrix (using the MATRIX=, SQRTVAR=, and CORRMAT= options)
- common standard deviation and correlation matrix (using the STDDEV=, MATRIX=, and CORRMAT= options)

For the relationship between covariates and response in a univariate model, specify either the multiple correlation (by using the CORRXY= option) or the proportional reduction in total R square (by using the PROPVARREDUCTION= option).

REPEATED Statement

REPEATED *factor-specification* **;**

If the MODEL statement includes more than one dependent variable, you can indicate a multivariate model and define transformations of dependent variables by using the REPEATED statement.

The REPEATED statement enables you to define custom Type III hypothesis tests by choosing from among several transformations of the dependent variables: contrast, Helmert, identity, mean, polynomial, and profile. You can specify a transformation for each repeated factor (often called a *within-subject factor*), and each combination of repeated factors produces an $\mathbf{M}$ vector or matrix for testing the hypothesis $\mathbf{L}\boldsymbol{\beta}\mathbf{M} = 0$. The $\mathbf{L}$ matrix consists of one or more between-subject contrasts that involve the model effects, and the $\mathbf{M}$ matrix

consists of one or more within-subject contrasts that involve the repeated factors. There is no limit to the number of repeated factors that you can specify.

Usually, the variables on the left side of the equation in the MODEL statement represent one repeated response variable. This does not mean that you are limited to listing only one factor in the REPEATED statement. For example, one repeated response variable (wellness rating) might be measured six times (implying variables Y1 to Y6 on the left side of the equal sign in the MODEL statement), with the associated within-subject factors rater and time (implying two factors listed in the REPEATED statement). However, designs that have two or more repeated response variables can be handled by using the IDENTITY transformation.

To use this feature, you must be familiar with the details of multivariate model and contrast parameterizations that PROC GLM uses. For more information, see the sections "Repeated Measures Analysis of Variance" on page 4249 and "Multivariate Analysis of Variance" on page 4247 in Chapter 53, "The GLM Procedure." For information about the power and sample size computational methods and formulas, see the section "Contrasts in Fixed-Effect Multivariate Models" on page 4387.

If you specify one or more REPEATED statements, then a "Mean(Dep)" transformation is added to the power analysis. This transformation is the mean of the dependent variables, the same transformation that is used implicitly in the "Tests of Hypotheses for Between Subjects Effects" table in PROC GLM. In addition, the Intercept model effect is included in the power analysis. If the REPEATED statement is not specified, then tests that involve the Intercept are excluded from the power analysis.

You can use either the REPEATED statement or the MANOVA statement along with any of the tests for multivariate models that are supported in the MTEST= option in the POWER statement. The REPEATED statement is usually used for handling repeated measurements on the same experimental unit, but you can also use the REPEATED statement for other situations, such as clusters or multiple outcome variables. The differences between the REPEATED and MANOVA statements are as follows:

- You can use the REPEATED statement to specify commonly used contrasts by using keywords rather than coefficients, but you are limited to only those forms of the M matrix.
- You can use the MANOVA statement to construct any M matrix, but you must specify the coefficients explicitly (except for the default identity matrix).

There is no limit to the number of REPEATED statements that you can specify. Each power analysis includes tests for all REPEATED statements and also (if you specify at least one REPEATED statement) the extra "Mean(Dep)" transformation that was previously mentioned.

The simplest form of the REPEATED statement requires only a *factor-name*. When you have two or more repeated factors, you must specify the *factor-name* and number of levels (*levels*) for each factor. Optionally, you can specify the actual values for the levels (*level-values*) and a *transformation* that defines single-degree-of-freedom contrasts. When you specify more than one within-subject factor, the *factor-names* (and associated level and transformation information) must be separated by a comma in the REPEATED statement.

The *factor-specification* for the REPEATED statement can include any number of individual factor specifications, separated by commas, of the following form:

factor-name levels < **(***level-values***)** > < *transformation* >

where

factor-name names a factor to be associated with the dependent variables. The name should not be the same as any variable name that already exists in the data set being analyzed and should conform to the usual conventions of SAS variable names.

When you specify more than one factor, list the dependent variables in the MODEL statement so that the within-subject factors that you define in the REPEATED statement are nested; that is, the first factor you define in the REPEATED statement should be the one whose values change least frequently.

levels gives the number of levels associated with the factor being defined. When there is only one within-subject factor, the number of levels is equal to the number of dependent variables. In this case, *levels* is optional. When more than one within-subject factor is defined, however, *levels* is required, and the product of the number of levels of all the factors must equal the number of dependent variables in the MODEL statement.

(*level-values*) gives values that correspond to levels of a repeated measures factor. These values are used as spacings for constructing orthogonal polynomial contrasts if you specify a POLYNOMIAL transformation. The number of values that you specify must correspond to the number of levels for that factor in the REPEATED statement. Enclose the *level-values* in parentheses.

The following *transformation* keywords define single-degree-of-freedom contrasts for factors that you specify in the REPEATED statement. Because the number of contrasts that are generated is always one less than the number of levels of the factor, you have some control over which contrast is omitted from the analysis by which transformation you select. The only exception is the IDENTITY transformation, which is not composed of contrasts and has the same degrees of freedom as the factor has levels. By default, PROC GLMPOWER uses the CONTRAST transformation.

CONTRAST< (*ordinal-reference-level*) >
generates contrasts between levels of the factor and a reference level. By default, PROC GLMPOWER uses the last level as the reference level; you can optionally specify a reference level in parentheses after the keyword CONTRAST. The reference level corresponds to the ordinal value of the level rather than the level value that is specified. For example, to generate contrasts between the first level of a factor and the other levels, specify CONTRAST(1).

HELMERT
generates contrasts between each level of the factor and the mean of subsequent levels.

IDENTITY
generates an identity transformation that corresponds to the associated factor. This transformation is *not* composed of contrasts; it has n degrees of freedom for an n-level factor, instead of $n - 1$ degrees of freedom.

MEAN< (*ordinal-reference-level*) >
generates contrasts between levels of the factor and the mean of all other levels of the factor. Specifying a reference level eliminates the contrast between that level and the mean. When no reference level is specified, the contrast that involves the last level is omitted. For an example, see the CONTRAST transformation.

POLYNOMIAL
generates orthogonal polynomial contrasts. Level values, if provided, are used as spacings in the construction of the polynomials; otherwise, equal spacing is assumed.

PROFILE
generates contrasts between adjacent levels of the factor.

Examples

When you specify more than one factor, list the dependent variables in the MODEL statement so that the within-subject factors that you define in the REPEATED statement are nested; that is, the first factor you define in the REPEATED statement should be the one whose values change least frequently. For example, assume that two raters submit a wellness rating at each of three times, for a total of six dependent variables for each subject. Consider the following statements:

```
proc glm;
   class treatment;
   model Y1-Y6 = treatment;
   repeated rater 2, time 3;
run;
```

The variables are listed in the MODEL statement as Y1 through Y6, so the REPEATED statement in the preceding statements implies the following structure:

	Dependent Variables					
	Y1	Y2	Y3	Y4	Y5	Y6
Value of rater	1	1	1	2	2	2
Value of time	1	2	3	1	2	3

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement names a variable that provides a profile weight ("cell weight") for each observation in the exemplary data set specified by the DATA= option in the PROC GLMPOWER statement.

If the WEIGHT statement is not used, then a balanced design is assumed with default cell weights of 1.

Details: GLMPOWER Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/gmpdet.sas.)

Specifying Value Lists in the POWER Statement

To specify one or more scenarios for an analysis parameter (or set of parameters) in the POWER statement, you provide a list of values for the option that corresponds to the parameter(s). To identify the parameter you want to solve for, you place a missing value in the appropriate list.

There are three basic types of such lists: *number-lists*, *name-lists*, and *keyword-lists*. Scenarios for scalar-valued parameters, such as power, are represented by a *number-list*. Scenarios for named parameters, such as correlation matrices, are represented by a *name-list*. Some parameters, such as the test statistic for a multivariate model, have values that are represented by one or more keywords in a *keyword-list*.

Number-Lists

A *number-list* can be one of two things: a series of one or more numbers expressed in the form of one or more DOLISTs, or a missing value indicator (.).

The DOLIST format is the same as in the DATA step. For example, you can specify four scenarios (30, 50, 70, and 100) for a total sample size in either of the following ways:

```
NTOTAL = 30 50 70 100
NTOTAL = 30 to 70 by 20 100
```

A missing value identifies a parameter as the result parameter; it is valid only with options representing parameters you can solve for in a given analysis. For example, you can request a solution for NTOTAL as follows:

```
NTOTAL = .
```

Name-Lists

A *name-list* is a list of one or more names that are enclosed in single or double quotation marks and separated by spaces. For example, you can specify two scenarios for the correlation matrix in a multivariate model as follows:

```
CORRMAT = "Corr A" "Corr B"
```

Keyword-Lists

A *keyword-list* is a list of one or more keywords, separated by spaces. For example, you can specify both the multivariate Hotelling-Lawley trace and uncorrected univariate F test for a multivariate model as follows:

```
MTEST = HLT UNCORR
```

Sample Size Adjustment Options

By default, PROC GLMPOWER rounds sample sizes conservatively (down in the input, up in the output) so that all total sizes *and* sample sizes for individual design profiles are integers. This is generally considered conservative because it selects the closest realistic design providing *at most* the power of the (possibly fractional) input or mathematically optimized design. In addition, all design profile sizes are adjusted to be multiples of their corresponding weights. If a design profile is present more than once in the exemplary data set, then the weights for that design profile are summed. For example, if a particular design profile is present twice in the exemplary data set with weight values 2 and 6, then all sample sizes for this design profile become multiples of 2 + 6 = 8.

With the NFRACTIONAL option, sample size input is not rounded, and sample size output is reported in two versions, a raw "fractional" version and a "ceiling" version rounded up to the nearest integer.

Whenever an input sample size is adjusted, both the original ("nominal") and adjusted ("actual") sample sizes are reported. Whenever computed output sample sizes are adjusted, both the original input ("nominal") power and the achieved ("actual") power at the adjusted sample size are reported.

Error and Information Output

The Error column in the main output table explains reasons for missing results and flags numerical results that are bounds rather than exact answers.

The Info column provides further information about Error entries, warnings about any boundary conditions detected, and notes about any adjustments to input. Note that the Info column is hidden by default in the main output. You can view it by using the ODS OUTPUT statement to save the output as a data set and the PRINT procedure. For example, the following SAS statements print both the Error and Info columns for a power computation in a one-way ANOVA:

```
data MyExemp;
   input A $ Y1 Y2;
   datalines;
         1   10 11
         2   12 11
         3   15 11
;

proc glmpower data=MyExemp;
   class A;
   model Y1 Y2 = A;
   power
```

```
      stddev = 2
      ntotal = 3 10
      power  = .;
   ods output output=Power;
run;

proc print noobs data=Power;
   var NominalNTotal NTotal Dependent Power Error Info;
run;
```

The output is shown in Figure 55.5.

Figure 55.5 Error and Information Columns

NominalNTotal	NTotal	Dependent	Power	Error	Info
3	3	Y1	.	Invalid input	Error DF=0
10	9	Y1	0.557		Input N adjusted
3	3	Y2	.	Invalid input	Error DF=0 / No effect
10	9	Y2	0.050		Input N adjusted / No effect

The sample size of 3 specified with the NTOTAL= option causes an "Invalid input" message in the Error column and an "Error DF=0" message in the Info column, because a sample size of 3 is so small that there are no degrees of freedom left for the error term. The sample size of 10 causes an "Input N adjusted" message in the Info column, because it is rounded down to 9 to produce integer group sizes of 3 per cell. The cell means scenario represented by the dependent variable Y2 causes a "No effect" message to appear in the Info column, because the means in this scenario are all equal.

Displayed Output

If you use the PLOTONLY option in the PROC GLMPOWER statement, the procedure displays only graphical output. Otherwise, the displayed output of the GLMPOWER procedure includes the following:

- the "Fixed Scenario Elements" table, which shows all applicable single-valued analysis parameters, in the following order: the dependent variable that represents the cell means scenario (for a univariate model) or the dependent variable transformation (for a multivariate model), the source of the test (that is, the model effect or between-subject contrast), the weight variable, parameters that are input explicitly, parameters that are supplied with defaults, and ancillary results
- an output table that shows the following when applicable (in order): the index of the scenario, the dependent variable that represents the cell means scenario (for a univariate model) or the dependent variable transformation (for a multivariate model), the type of the test, the source of the test (that is, the model effect or between-subject contrast), all multivalued input, ancillary results, the primary computed result, and error descriptions
- plots (if requested)

The exception to these ordering conventions is that the DEPENDENT and EFFECTS= options can be used along with the OUTPUTORDER=SYNTAX or OUTPUTORDER=REVERSE option in the POWER statement to specify the relative location of the output for dependent variable and type and source of test.

Ancillary results include the following:

- Actual Power, the achieved power, if it differs from the input (Nominal) power value
- Actual Alpha, the achieved significance level, if it differs from the input (Nominal) alpha value
- fractional sample size, if the NFRACTIONAL option is used in the POWER statement
- test or numerator degrees of freedom in the test's critical value
- error or denominator degrees of freedom in the test's critical value
- Effect, the combination of the within-subject Transformation contrast and between-subject Source test or contrast in a multivariate model

If sample size is the result parameter and the NFRACTIONAL option is used in the POWER statement, then both "Fractional" and "Ceiling" sample size results are displayed. Fractional sample sizes correspond to the "Nominal" values of power. Ceiling sample sizes are simply the fractional sample sizes rounded up to the nearest integer; they correspond to "Actual" values of power.

The noncentrality parameter is computed and stored in a hidden column called Noncentrality in the "Output" table. If a univariate test for a multivariate model is specified (that is, one of MTEST=BOX, MTEST=GG, MTEST=HF, or MTEST=UNCORR), then the numerator and denominator degrees of freedom that are used in the noncentral F approximation of the test statistic distribution are computed and stored in hidden columns called NumNCDF and DenNCDF, respectively, in the "Output" table. These are the only tests for which the degrees of freedom in the noncentral F approximation of the test statistic are different from those in the critical value.

ODS Table Names

PROC GLMPOWER assigns a name to each table that it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 55.7. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 55.7 ODS Tables Produced by PROC GLMPOWER

ODS Table Name	Description	Statement
FixedElements	Factoid with single-valued analysis parameters	Default
Output	All input and computed analysis parameters, error messages, and information messages for each scenario	Default
PlotContent	Data contained in plots, including analysis parameters and indices identifying plot features. (**NOTE:** This table is saved as a data set and not displayed in PROC GLMPOWER output.)	PLOT

Computational Methods and Formulas

This section describes the approaches that PROC GLMPOWER uses to compute power and sample size.

Contrasts in Fixed-Effect Univariate Models

The univariate linear model has the form

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\mathbf{y}$ is the $N \times 1$ vector of responses, $\mathbf{X}$ is the $N \times k$ design matrix, $\boldsymbol{\beta}$ is the $k \times 1$ vector of model parameters that correspond to the columns of $\mathbf{X}$, and $\boldsymbol{\epsilon}$ is an $N \times 1$ vector of errors with

$$\epsilon_1, \ldots, \epsilon_N \sim \mathrm{N}(0, \sigma^2) \quad \text{(iid)}$$

In PROC GLMPOWER, the model parameters $\boldsymbol{\beta}$ are not specified directly, but rather indirectly as $\mathbf{y}^\star$, which represents either conjectured response means or typical response values for each design profile. The $\mathbf{y}^\star$ values are manifested as the dependent variable in the MODEL statement. The vector $\boldsymbol{\beta}$ is obtained from $\mathbf{y}^\star$ according to the least squares equation,

$$\boldsymbol{\beta} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y}^\star$$

Note that, in general, there is not a one-to-one mapping between $\mathbf{y}^\star$ and $\boldsymbol{\beta}$. Many different scenarios for $\mathbf{y}^\star$ might lead to the same $\boldsymbol{\beta}$. If you specify $\mathbf{y}^\star$ with the intention of representing cell means, keep in mind that PROC GLMPOWER allows scenarios that are *not* valid cell means according to the model that is specified in the MODEL statement. For example, if $\mathbf{y}^\star$ exhibits an interaction effect but the corresponding interaction term is left out of the model, then the cell means ($\mathbf{X}\boldsymbol{\beta}$) that are derived from $\boldsymbol{\beta}$ differ from $\mathbf{y}^\star$. In particular, the cell means that are derived in this way are the projection of $\mathbf{y}^\star$ onto the model space.

It is convenient in power analysis to parameterize the design matrix $\mathbf{X}$ in three parts, $\{\ddot{\mathbf{X}}, \mathbf{w}, N\}$, defined as follows:

1. The $q \times k$ essence design matrix $\ddot{\mathbf{X}}$ is the collection of unique rows of $\mathbf{X}$. Its rows are sometimes referred to as "design profiles." Here, $q \leq N$ is defined simply as the number of unique rows of $\mathbf{X}$.
2. The $q \times 1$ weight vector $\mathbf{w}$ reveals the relative proportions of design profiles, and $\mathbf{W} = \mathrm{diag}(\mathbf{w})$. Row i of $\ddot{\mathbf{X}}$ is to be included in the design w_i times for every w_j times that row j is included. The weights are assumed to be standardized (that is, they sum up to 1).
3. The total sample size is N. This is the number of rows in $\mathbf{X}$. If you gather $Nw_i = n_i$ copies of the ith row of $\ddot{\mathbf{X}}$, for $i = 1, \ldots, q$, then you end up with $\mathbf{X}$.

The preceding quantities are derived from PROC GLMPOWER syntax as follows:

- Values for $\ddot{\mathbf{X}}$, $\mathbf{y}^\star$, and $\mathbf{w}$ are specified in the exemplary data set (from using the DATA= option in the PROC GLMPOWER statement), and the corresponding variables are identified in the CLASS, MODEL, and WEIGHT statements.
- N is specified in the NTOTAL= option in the POWER statement.

It is useful to express the crossproduct matrix $\mathbf{X}'\mathbf{X}$ in terms of these three parts,

$$\mathbf{X}'\mathbf{X} = N\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}}$$

because this expression factors out the portion (N) that depends on sample size and the portion ($\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}}$) that depends only on the design structure.

A general linear hypothesis for the univariate model has the form

$$\begin{aligned} H_0{:}\mathbf{L}\boldsymbol{\beta} &= \boldsymbol{\theta}_0 \\ H_A{:}\mathbf{L}\boldsymbol{\beta} &\neq \boldsymbol{\theta}_0 \end{aligned}$$

where $\mathbf{L}$ is an $1 \times k$ contrast matrix with rank r_L and $\boldsymbol{\theta}_0$ is the null value (usually just a vector of zeros).

Note that model effect tests are just contrasts that use special forms of $\mathbf{L}$. Thus, this scheme covers both effect tests (which are specified in the MODEL statement and the EFFECTS= option in the POWER statement) and custom contrasts (which are specified in the CONTRAST statement).

The model degrees of freedom $\mathrm{DF_M}$ are equal to the rank of $\mathbf{X}$, denoted r_X. The error degrees of freedom $\mathrm{DF_E}$ are equal to $N - r_X$. The sample size N must be at least $\mathrm{DF_M}$ plus the number of covariates.

The test statistic is

$$F = \frac{\left(\frac{\mathrm{SS_H}}{r_L}\right)}{\hat{\sigma}^2}$$

where

$$\begin{aligned} \mathrm{SS_H} &= \frac{1}{N}\left(\mathbf{L}\hat{\boldsymbol{\beta}} - \boldsymbol{\theta}_0\right)' \left(\mathbf{L}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{L}'\right)^{-1}\left(\mathbf{L}\hat{\boldsymbol{\beta}} - \boldsymbol{\theta}_0\right) \\ \hat{\boldsymbol{\beta}} &= (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y} \\ \hat{\sigma}^2 &= \frac{1}{\mathrm{DF_E}}\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right)'\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right) \end{aligned}$$

Under H_0, $F \sim F(r_L, \mathrm{DF_E})$. Under H_A, F is distributed as $F(r_L, \mathrm{DF_E}, \lambda)$ with noncentrality

$$\lambda = N\left(\mathbf{L}\boldsymbol{\beta} - \boldsymbol{\theta}_0\right)'\left(\mathbf{L}\left(\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}}\right)^{-1}\mathbf{L}'\right)^{-1}\left(\mathbf{L}\boldsymbol{\beta} - \boldsymbol{\theta}_0\right)\sigma^{-2}$$

The value of σ is specified in the STDDEV= option in the POWER statement.

Muller and Peterson (1984) give the exact power of the test as

$$\mathrm{power} = P\left(F(r_L, \mathrm{DF_E}, \lambda) \geq F_{1-\alpha}(r_L, \mathrm{DF_E})\right)$$

The value of α is specified in the ALPHA= option in the POWER statement.

Sample size is computed by inverting the power equation.

See Muller and Benignus (1992) and O'Brien and Shieh (1992) for additional discussion.

Adjustments for Covariates in Univariate Models

If you specify covariates in a univariate model (whether continuous or categorical), then two adjustments are made in order to compute approximate power in the presence of the covariates. Let n_ν denote the number of covariates (counting dummy variables for categorical covariates individually) as specified in the NCOVARIATES= option in the POWER statement. In other words, n_ν is the total degrees of freedom used by the covariates. The adjustments are as follows:

1. The error degrees of freedom decrease by n_ν.
2. The error standard deviation σ shrinks by a factor of $(1-\rho^2)^{\frac{1}{2}}$ (if the CORRXY= option is used to specify the correlation ρ between covariates and response) or $(1-r)^{\frac{1}{2}}$ (if the PROPVARREDUCTION= option is used to specify the proportional reduction in total R^2 incurred by the covariates). Let $\sigma^\star$ represent the updated value of σ.

As a result of these changes, the power is computed as

$$\text{power} = P\left(F(r_L, \mathrm{DF_E} - n_\nu, \lambda^\star) \geq F_{1-\alpha}(r_L, N - r_x - n_\nu)\right)$$

where $\lambda^\star$ is calculated using $\sigma^\star$ rather than σ:

$$\lambda^\star = N\,(\mathbf{L}\boldsymbol{\beta} - \boldsymbol{\theta}_0)'\left(\mathbf{L}\left(\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}}\right)^{-1}\mathbf{L}'\right)^{-1}(\mathbf{L}\boldsymbol{\beta} - \boldsymbol{\theta}_0)\,(\sigma^\star)^{-2}$$

Contrasts in Fixed-Effect Multivariate Models

The multivariate model has the form

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\mathbf{Y}$ is the $N \times p$ vector of responses, for $p > 1$; $\mathbf{X}$ is the $N \times k$ design matrix; $\boldsymbol{\beta}$ is the $k \times p$ matrix of model parameters that correspond to the columns of $\mathbf{X}$ and $\mathbf{Y}$; and $\boldsymbol{\epsilon}$ is an $N \times p$ vector of errors, where

$$\epsilon_1, \ldots, \epsilon_N \sim \mathrm{N}(0, \boldsymbol{\Sigma}) \quad \text{(iid)}$$

In PROC GLMPOWER, the model parameters $\boldsymbol{\beta}$ are not specified directly, but rather indirectly as $\mathbf{Y}^\star$, which represents either conjectured response means or typical response values for each design profile. The $\mathbf{Y}^\star$ values are manifested as the collection of dependent variables in the MODEL statement. The matrix $\boldsymbol{\beta}$ is obtained from $\mathbf{Y}^\star$ according to the least squares equation,

$$\boldsymbol{\beta} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y}^\star$$

Note that, in general, there is not a one-to-one mapping between $\mathbf{Y}^\star$ and $\boldsymbol{\beta}$. Many different scenarios for $\mathbf{Y}^\star$ might lead to the same $\boldsymbol{\beta}$. If you specify $\mathbf{Y}^\star$ with the intention of representing cell means, keep in mind that PROC GLMPOWER allows scenarios that are *not* valid cell means according to the model that is specified in the MODEL statement. For example, if $\mathbf{Y}^\star$ exhibits an interaction effect but the corresponding interaction term is left out of the model, then the cell means ($\mathbf{X}\boldsymbol{\beta}$) that are derived from $\boldsymbol{\beta}$ differ from $\mathbf{Y}^\star$. In particular, the cell means that are derived in this way are the projection of $\mathbf{Y}^\star$ onto the model space.

It is convenient in power analysis to parameterize the design matrix $\mathbf{X}$ in three parts, $\{\ddot{\mathbf{X}}, \mathbf{W}, N\}$, defined as follows:

1. The $q \times k$ essence design matrix $\ddot{\mathbf{X}}$ is the collection of unique rows of $\mathbf{X}$. Its rows are sometimes referred to as "design profiles." Here, $q \leq N$ is defined simply as the number of unique rows of $\mathbf{X}$.
2. The $q \times 1$ weight vector $\mathbf{w}$ reveals the relative proportions of design profiles, and $\mathbf{W} = \mathrm{diag}(\mathbf{w})$. Row i of $\ddot{\mathbf{X}}$ is to be included in the design w_i times for every w_j times that row j is included. The weights are assumed to be standardized (that is, they sum up to 1).
3. The total sample size is N. This is the number of rows in $\mathbf{X}$. If you gather $Nw_i = n_i$ copies of the ith row of $\ddot{\mathbf{X}}$, for $i = 1, \ldots, q$, then you end up with $\mathbf{X}$.

The preceding quantities are derived from PROC GLMPOWER syntax as follows:

- Values for $\ddot{\mathbf{X}}$, $\mathbf{Y}^{\star}$, and $\mathbf{w}$ are specified in the exemplary data set (from using the DATA= option in the PROC GLMPOWER statement), and the corresponding variables are identified in the CLASS, MODEL, and WEIGHT statements.
- N is specified in the NTOTAL= option in the POWER statement.

It is useful to express the crossproduct matrix $\mathbf{X}'\mathbf{X}$ in terms of these three parts,

$$\mathbf{X}'\mathbf{X} = N\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}}$$

because this expression factors out the portion (N) that depends on sample size and the portion ($\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}}$) that depends only on the design structure.

A general linear hypothesis for the univariate model has the form

$$H_0{:}\mathbf{L}\boldsymbol{\beta}\mathbf{M} = \boldsymbol{\theta}_0$$
$$H_A{:}\mathbf{L}\boldsymbol{\beta}\mathbf{M} \neq \boldsymbol{\theta}_0$$

where $\mathbf{L}$ is an $l \times k$ between-subject contrast matrix with rank r_L, $\mathbf{M}$ is a $p \times m$ within-subject contrast matrix with rank r_M, and $\boldsymbol{\theta}_0$ is an $\mathrm{l} \times m$ null contrast matrix (usually just a matrix of zeros).

Note that model effect tests are just between-subject contrasts that use special forms of $\mathbf{L}$, combined with an $\mathbf{M}$ that is the $p \times 1$ mean transformation vector of the dependent variables (a vector of values all equal to $1/p$). Thus, this scheme covers both effect tests (which are specified in the MODEL statement and the EFFECTS= option in the POWER statement) and custom between-subject contrasts (which are specified in the CONTRAST statement).

The $\mathbf{M}$ matrix is often referred to as the dependent variable transformation and is specified in the MANOVA or REPEATED statement.

The model degrees of freedom $\mathrm{DF_M}$ are equal to the rank of $\mathbf{X}$, denoted r_X. The error degrees of freedom $\mathrm{DF_E}$ are equal to $N - r_X$.

The hypothesis sum of squares $\mathrm{SS_H}$ in the univariate model generalizes to the hypothesis SSCP matrix in the multivariate model,

$$\mathbf{H} = \left(\mathbf{L}\hat{\boldsymbol{\beta}}\mathbf{M} - \boldsymbol{\theta}_0\right)' \left(\mathbf{L}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{L}'\right)^{-1} \left(\mathbf{L}\hat{\boldsymbol{\beta}}\mathbf{M} - \boldsymbol{\theta}_0\right)$$

The error sum of squares $\hat{\sigma}^2(N - r_X)$ in the univariate model generalizes to the error SSCP matrix in the multivariate model,

$$\mathbf{E} = (N - r_X)\mathbf{M}'\hat{\boldsymbol{\Sigma}}\mathbf{M}$$

where

$$\hat{\mathbf{\Sigma}} = \left(\mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right)' \left(\mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right) / (N - r_X)$$

and

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y}$$

The population counterpart of $\mathbf{H}/N$ is

$$\mathbf{H}^{\star} = (\mathbf{L}\boldsymbol{\beta}\mathbf{M} - \boldsymbol{\theta}_0)' \left(\mathbf{L}\left(\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}}\right)^{-}\mathbf{L}'\right)^{-1} (\mathbf{L}\boldsymbol{\beta}\mathbf{M} - \boldsymbol{\theta}_0)$$

and the population counterpart of $\mathbf{E}/N$ is

$$\mathbf{E}^{\star} = \mathbf{M}'\mathbf{\Sigma}\mathbf{M}$$

The elements of $\mathbf{\Sigma}$ are specified in the MATRIX= and STDDEV= options and identified in the CORRMAT=, CORRS=, COVMAT=, and SQRTVAR= options in the POWER statement.

The power and sample size computations for all the tests that are supported in the MTEST= option in the POWER statement are based on $\mathbf{H}^{\star}$ and $\mathbf{E}^{\star}$. The following two subsections cover the computational methods and formulas for the multivariate and univariate tests that are supported in the MTEST= and UEPSDEF= options in the POWER statement.

Multivariate Tests

Power computations for multivariate tests are based on O'Brien and Shieh (1992) (for METHOD=OBRIENSHIEH) and Muller and Peterson (1984) (for METHOD=MULLERPETERSON).

Let $s = \min(r_L, r_M)$, the smaller of the between-subject and within-subject contrast degrees of freedom. Critical value computations assume that under H_0, the test statistic F is distributed as $F(r_L r_M, \nu_2)$, where $\nu_2 = (N - r_X) - r_M + 1$ if $s = 1$ but depends on the choice of test if $s > 1$. Power computations assume that under H_A, F is distributed as $F(r_L r_M, \nu_2, \lambda)$, where the noncentrality λ depends on r_L, r_M, the choice of test, and the power computation method.

Formulas for the test statistic F, denominator degrees of freedom ν_2, and noncentrality λ for all combinations of dimensions, tests, and methods are given in the following subsections.

The power in each case is computed as

$$\text{power} = P\left(F(r_L r_M, \nu_2, \lambda) \geq F_{1-\alpha}(r_L r_M, \nu_2)\right)$$

Computed power is exact for some cases and approximate for others. Sample size is computed by inverting the power equation.

Let $\mathbf{\Delta} = \mathbf{E}^{-1}\mathbf{H}$, and define $\boldsymbol{\phi}$ as the $s \times 1$ vector of ordered positive eigenvalues of $\mathbf{\Delta}$, $\boldsymbol{\phi} = \{\phi_1, \ldots, \phi_s\}$, where $\phi_1 \geq \cdots \geq \phi_s > 0$. The population equivalent is

$$\begin{aligned}\mathbf{\Delta}^{\star} &= \mathbf{E}^{\star -1}\mathbf{H}^{\star} \\ &= \left(\mathbf{M}'\mathbf{\Sigma}\mathbf{M}\right)^{-1} (\mathbf{L}\boldsymbol{\beta}\mathbf{M} - \boldsymbol{\theta}_0)' \left(\mathbf{L}\left(\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}}\right)^{-1}\mathbf{L}'\right)^{-1} (\mathbf{L}\boldsymbol{\beta}\mathbf{M} - \boldsymbol{\theta}_0)\end{aligned}$$

where $\boldsymbol{\phi}^{\star}$ is the $s \times 1$ vector of ordered positive eigenvalues of $\mathbf{\Delta}^{\star}$, $\boldsymbol{\phi}^{\star} = \{\phi_1^{\star}, \ldots, \phi_s^{\star}\}$ for $\phi_1^{\star} \geq \cdots \geq \phi_s^{\star} > 0$.

Case 1: $s = 1$

When $s = 1$, all three multivariate tests (MTEST=HLT, MTEST=PT, and MTEST=WILKS) are equivalent. The test statistic is $F = \phi_1 \nu_2/(r_L r_M)$, where $\nu_2 = (N - r_X) - r_M + 1$.

When the dependent variable transformation has a single degree of freedom ($r_M = 1$), METHOD=OBRIENSHIEH and METHOD=MULLERPETERSON are the same, computing exact power by using noncentrality $\lambda = N\mathbf{\Delta}^\star$. The sample size must satisfy $N \geq r_X + 1$.

When the dependent variable transformation has more than one degree of freedom but the between-subject contrast has a single degree of freedom ($r_M > 1, r_L = 1$), METHOD=OBRIENSHIEH computes exact power by using noncentrality $\lambda = N\phi_1^\star$, and METHOD=MULLERPETERSON computes approximate power by using

$$\lambda = \frac{(N - r_X) - r_M + 1}{(N - r_X)} N\phi_1^\star$$

The sample size must satisfy $N \geq r_X + r_M$.

Case 2: $s > 1$

When both the dependent variable transformation and the between-subject contrast have more than one degree of freedom ($s > 1$), METHOD=OBRIENSHIEH computes the noncentrality as $\lambda = N\lambda^\star$, where $\lambda^\star$ is the primary noncentrality. The form of $\lambda^\star$ depends on the choice of test statistic.

METHOD=MULLERPETERSON computes the noncentrality as $\lambda = \nu_2 \lambda^{(\mathrm{MP})\star}$, where $\lambda^{(\mathrm{MP})\star}$ has the same form as $\lambda^\star$ except that $\boldsymbol{\phi}^\star$ is replaced by

$$\boldsymbol{\phi}^{(\mathrm{MP})\star} = \frac{N}{(N - r_X)} \boldsymbol{\phi}^\star$$

Computed power is approximate for both methods when $s > 1$.

Hotelling-Lawley Trace (MTEST=HLT) When $s > 1$

If $N > r_X + r_M + 1$, then the denominator degrees of freedom for the Hotelling-Lawley trace are $\nu_2 = \nu_{2a}$,

$$\nu_{2a} = 4 + (r_L r_M + 2)g$$

where

$$g = \frac{(N - r_X)^2 - (N - r_X)(2r_M + 3) + r_M(r_M + 3)}{(N - r_X)(r_L + r_M + 1) - (r_L + 2r_M + r_M^2 - 1)}$$

which is the same as $\nu_2^{(T_2)}$ in O'Brien and Shieh (1992) and is due to McKeon (1974).

If $N \leq r_X + r_M + 1$, then $\nu_2 = \nu_{2b}$,

$$\nu_{2b} = s((N - r_X) - r_M - 1) + 2$$

which is the same as both $\nu_2^{(T_1)}$ in O'Brien and Shieh (1992) and ν_2 in Muller and Peterson (1984) and is due to Pillai and Samson (1959).

The primary noncentrality is

$$\lambda^\star = \sum_{i=1}^{s} \phi_i^\star$$

The sample size must satisfy

$$N \geq r_X + r_M + 1 - 1/s$$

If $N > r_X + r_M + 1$, then the test statistic is

$$F = \frac{U/\nu_1}{c/\nu_{2a}}$$

where

$$\begin{aligned} U &= \mathrm{trace}(\mathbf{E}^{-1}\mathbf{H}) \\ &= \sum_{i=1}^{s} \phi_i \end{aligned}$$

and

$$c = \frac{2 + (r_L r_M + 2)g}{N - r_X - r_M - 1}$$

If $N \leq r_X + r_M + 1$, then the test statistic is

$$F = \frac{U/\nu_1}{s/\nu_{2b}}$$

Pillai's Trace (MTEST=PT) When $s > 1$

The denominator degrees of freedom for Pillai's trace are

$$\nu_2 = s((N - r_X) + s - r_M)$$

The primary noncentrality is

$$\lambda^\star = s\left(\frac{\sum_{i=1}^{s} \frac{\phi_i^\star}{1+\phi_i^\star}}{s - \sum_{i=1}^{s} \frac{\phi_i^\star}{1+\phi_i^\star}}\right)$$

The sample size must satisfy

$$N \geq r_X + r_M + 1/s - s$$

The test statistic is

$$F = \frac{V/\nu_1}{(s - V)/\nu_2}$$

where

$$\begin{aligned} V &= \mathrm{trace}\left(\mathbf{H}(\mathbf{H} + \mathbf{E})^{-1}\right) \\ &= \sum_{i=1}^{s} \frac{\phi_i}{1 + \phi_i} \end{aligned}$$

Wilks' Lambda (MTEST=WILKS) When $s > 1$

The denominator degrees of freedom for Wilks' lambda are

$$\nu_2 = t\,[(N - r_X) - 0.5(r_M - r_L + 1)] - 0.5(r_L r_M - 2)$$

where

$$t = \begin{cases} 1 & \text{if } r_L r_M \le 3 \\ \left[\frac{(r_L r_M)^2 - 4}{r_L^2 + r_M^2 - 5}\right]^{\frac{1}{2}} & \text{if } r_L r_M \ge 4 \end{cases}$$

The primary noncentrality is

$$\lambda^\star = t\left[\left(\prod_{i=1}^{s}\left[(1+\phi_i^\star)^{-1}\right]\right)^{-\frac{1}{t}} - 1\right]$$

The sample size must satisfy

$$N \ge (1 + 0.5(r_L r_M - 2))/t + r_X + (r_M - r_L + 1)/2$$

The test statistic is

$$F = \frac{(1 - \Lambda^{1/t})/\nu_1}{\Lambda^{1/t}/\nu_2}$$

where

$$\begin{aligned}\Lambda &= \det(\mathbf{E})/\det(\mathbf{H} + \mathbf{E}) \\ &= \prod_{i=1}^{s}\left[(1+\phi_i)^{-1}\right]\end{aligned}$$

Univariate Tests

Power computations for univariate tests are based on Muller et al. (2007) and Muller and Barton (1989).

The test statistic is

$$F = \frac{\mathrm{trace}(\mathbf{H})/r_L}{\mathrm{trace}(\mathbf{E})/(N - r_X)}$$

Critical value computations assume that under H_0, F is distributed as $F(\nu_1, \nu_2)$, where ν_1 and ν_2 depend on the choice of test.

The four tests for the univariate approach to repeated measures differ in their assumptions about the sphericity ε of $\mathbf{E}^\star$,

$$\varepsilon = \frac{\mathrm{trace}^2(\mathbf{E}^\star)}{r_M\,\mathrm{trace}(\mathbf{E}^{\star 2})}$$

Power computations assume that under H_A, F is distributed as $F(\nu_1^\star, \nu_2^\star, \lambda)$.

Formulas for ν_1 and ν_2 for each test and formulas for $\nu_1^\star$, $\nu_2^\star$, and λ are given in the following subsections.

The power in each case is approximated as

$$\text{power} = P\left(F(\nu_1^\star, \nu_2^\star, \lambda) \geq F_{1-\alpha}(\nu_1, \nu_2)\right)$$

Sample size is computed by inverting the power equation.

The sample size must be large enough to yield $\nu_1 > 0$, $\nu_1^\star > 0$, $\nu_2 \geq 1$, and $\nu_2^\star \geq 1$.

Because these univariate tests are biased, the achieved significance level might differ from the nominal significance level. The actual alpha is computed in the same way as the power, except that the noncentrality parameter λ is set to 0.

Define $\boldsymbol{\phi}^{(\mathrm{E})}$ as the vector of ordered eigenvalues of $\mathbf{E}^\star$, $\boldsymbol{\phi}^{(\mathrm{E})} = \{\phi_1^{(\mathrm{E})}, \ldots, \phi_{r_M}^{(\mathrm{E})}\}$, where $\phi_1^{(\mathrm{E})} \geq \cdots \geq \phi_{r_M}^{(\mathrm{E})}$, and define $\boldsymbol{\gamma}_j^{(\mathrm{E})}$ as the jth eigenvector of $\mathbf{E}^\star$. Critical values and power computations are based on the following intermediate parameters:

$$\omega_{*j} = N\left(\boldsymbol{\gamma}_j^{(\mathrm{E})}\right)' \mathbf{H}^\star \boldsymbol{\gamma}_j^{(\mathrm{E})} / \phi_j^{(\mathrm{E})}$$

$$S_{t1} = \sum_{j=1}^{r_M} \phi_j^{(\mathrm{E})}$$

$$S_{t2} = \sum_{j=1}^{r_M} \phi_j^{(\mathrm{E})} \omega_{*j}$$

$$S_{t3} = \sum_{j=1}^{r_M} \left(\phi_j^{(\mathrm{E})}\right)^2$$

$$S_{t4} = \sum_{j=1}^{r_M} \left(\phi_j^{(\mathrm{E})}\right)^2 \omega_{*j}$$

$$R_{*1} = \frac{r_L S_{t3} + 2S_{t4}}{r_L S_{t1} + 2S_{t2}}$$

$$R_{*2} = \frac{S_{t3}}{S_{t1}}$$

$$\mathrm{E}(t_1) = 2(N - r_X)S_{t3} + (N - r_X)^2 S_{t1}^2$$

$$\mathrm{E}(t_2) = (N - r_X)((N - r_X) + 2)S_{t3} + 2(N - r_X) \sum_{j_1=2}^{r_M} \sum_{j_2=1}^{j_1-1} \phi_{j_1}^{(\mathrm{E})} \phi_{j_2}^{(\mathrm{E})}$$

The degrees of freedom and noncentrality in the noncentral F approximation of the test statistic are computed as follows:

$$\nu_1^\star = \frac{r_L S_{t1}}{R_{*1}}$$

$$\nu_2^\star = \frac{(N - r_X)S_{t1}}{R_{*2}}$$

$$\lambda = \frac{S_{t2}}{R_{*1}}$$

Uncorrected Test

The uncorrected test assumes sphericity $\varepsilon = 1$, in which case the null F distribution is exact, with the following degrees of freedom:

$$\nu_1 = r_L r_M$$
$$\nu_2 = r_M(N - r_X)$$

Greenhouse-Geisser Adjustment (MTEST=UNCORR)

The Greenhouse-Geisser adjustment to the uncorrected test reduces degrees of freedom by the MLE $\hat{\varepsilon}$ of the sphericity,

$$\hat{\varepsilon} = \frac{\mathrm{trace}^2(\mathbf{E})}{r_M \mathrm{trace}(\mathbf{E}^2)}$$

An approximation for the expected value of $\hat{\varepsilon}$ is used to compute the degrees of freedom for the null F distribution,

$$\nu_1 = r_L r_M \mathrm{E}(\hat{\varepsilon})$$
$$\nu_2 = r_M(N - r_X)\mathrm{E}(\hat{\varepsilon})$$

where

$$\mathrm{E}(\hat{\varepsilon}) = \frac{\mathrm{E}(t_1)}{r_M \mathrm{E}(t_2)}$$

Huynh-Feldt Adjustments (MTEST=HF)

The Huynh-Feldt adjustment reduces degrees of freedom by a nearly unbiased estimate $\tilde{\varepsilon}$ of the sphericity,

$$\tilde{\varepsilon} = \begin{cases} \frac{N r_M \hat{\varepsilon} - 2}{r_M[(N - r_X) - r_M \hat{\varepsilon}]} & \text{if UEPSDEF=HF} \\ \frac{(N - r_X + 1) r_M \hat{\varepsilon} - 2}{r_M[(N - r_X) - r_M \hat{\varepsilon}]} & \text{if UEPSDEF=HFL} \\ \left(\frac{(\nu_a - 2)(\nu_a - 4)}{\nu_a^2}\right)\left(\frac{(N - r_X + 1) r_M \hat{\varepsilon} - 2}{r_M[(N - r_X) - r_M \hat{\varepsilon}]}\right) & \text{if UEPSDEF=CM} \end{cases}$$

where

$$\nu_a = (N - r_X - 1) + (N - r_X)(N - r_X - 1)/2$$

The value of $\tilde{\varepsilon}$ is truncated if necessary to be at least $1/r_M$ and at most 1.

An approximation for the expected value of $\tilde{\varepsilon}$ is used to compute the degrees of freedom for the null F distribution,

$$\nu_1 = r_L r_M \mathrm{E_t}(\tilde{\varepsilon})$$
$$\nu_2 = r_M(N - r_X)\mathrm{E_t}(\tilde{\varepsilon})$$

where

$$\mathrm{E_t}(\tilde{\varepsilon}) = \min(\max(\mathrm{E}(\tilde{\varepsilon}), 1/r_M), 1)$$

and

$$\mathrm{E}(\tilde{\varepsilon}) = \begin{cases} \frac{N\mathrm{E}(t_1)-2\mathrm{E}(t_2)}{r_M[(N-r_X)\mathrm{E}(t_2)-\mathrm{E}(t_1)]} & \text{if UEPSDEF=HF} \\ \frac{(N-r_X+1)\mathrm{E}(t_1)-2\mathrm{E}(t_2)}{r_M[(N-r_X)\mathrm{E}(t_2)-\mathrm{E}(t_1)]} & \text{if UEPSDEF=HFL} \\ \left(\frac{(\nu_a-2)(\nu_a-4)}{\nu_a^2}\right)\left(\frac{(N-r_X+1)\mathrm{E}(t_1)-2\mathrm{E}(t_2)}{r_M[(N-r_X)\mathrm{E}(t_2)-\mathrm{E}(t_1)]}\right) & \text{if UEPSDEF=CM} \end{cases}$$

Box Conservative Test (MTEST=BOX)

The Box conservative test assumes the worst case for sphericity, $\varepsilon = 1/r_M$, leading to the following degrees of freedom for the null F distribution:

$$\nu_1 = r_L$$
$$\nu_2 = (N - r_X)$$

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is not enabled, then PROC GLMPOWER creates traditional graphics.

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC GLMPOWER generates are listed in Table 55.8, along with the required statements and options.

Table 55.8 Graphs Produced by PROC GLMPOWER

ODS Graph Name	Plot Description	Option
PowerPlot	Plot with power and sample size on the axes	PLOT
PowerAbort	Empty plot that shows an error message when a plot could not be produced	PLOT

Examples: GLMPOWER Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 55.1: One-Way ANOVA

(View the complete code for this example (gmpex01.sas) in the example repository.)

This example deals with the same situation as in Example 96.1 in Chapter 96, "The POWER Procedure."

Hocking (1985, p. 109) describes a study of the effectiveness of electrolytes in reducing lactic acid buildup for long-distance runners. You are planning a similar study in which you will allocate five different fluids to runners on a 10-mile course and measure lactic acid buildup immediately after the race. The fluids consist of water and two commercial electrolyte drinks, EZDure and LactoZap, each prepared at two concentrations, low (EZD1 and LZ1) and high (EZD2 and LZ2).

You conjecture that the standard deviation of lactic acid measurements given any particular fluid is about 3.75, and that the expected lactic acid values will correspond roughly to Table 55.9. You are least familiar with the LZ1 drink and hence decide to consider a range of reasonable values for that mean.

Table 55.9 Mean Lactic Acid Buildup by Fluid

Water	EZD1	EZD2	LZ1	LZ2
35.6	33.7	30.2	29 or 28	25.9

You are interested in four different comparisons, shown in Table 55.10 with appropriate contrast coefficients.

Table 55.10 Planned Comparisons

	Contrast Coefficients				
Comparison	**Water**	**EZD1**	**EZD2**	**LZ1**	**LZ2**
Water versus electrolytes	4	-1	-1	-1	-1
EZD versus LZ	0	1	1	-1	-1
EZD1 versus EZD2	0	1	-1	0	0
LZ1 versus LZ2	0	0	0	1	-1

For each of these contrasts you want to determine the sample size required to achieve a power of 0.9 for detecting an effect with magnitude in accord with Table 55.9. You are not yet attempting to choose a single sample size for the study, but rather checking the range of sample sizes needed for individual contrasts. You plan to test each contrast at $\alpha = 0.025$. In the interests of reducing costs, you will provide twice as many runners with water as with any of the electrolytes; that is, you will use a sample size weighting scheme of 2:1:1:1:1.

Before calling PROC GLMPOWER, you need to create the *exemplary data set* to specify means and weights for the design profiles:

```
data Fluids;
   input Fluid $ LacticAcid1 LacticAcid2 CellWgt;
   datalines;
         Water      35.6        35.6        2
         EZD1       33.7        33.7        1
         EZD2       30.2        30.2        1
         LZ1        29          28          1
         LZ2        25.9        25.9        1
;
```

The variable LacticAcid1 represents the cell means scenario with the larger LZ1 mean (29), and LacticAcid2 represents the scenario with the smaller LZ1 mean (28). The variable CellWgt contains the sample size allocation weights.

Use the DATA= option in the PROC GLMPOWER statement to specify Fluids as the exemplary data set. The following statements perform the sample size analysis:

```
proc glmpower data=Fluids;
   class Fluid;
   model LacticAcid1 LacticAcid2 = Fluid;
   weight CellWgt;
   contrast "Water vs. others" Fluid  -1 -1 -1 -1 4;
   contrast "EZD vs. LZ"       Fluid   1  1 -1 -1 0;
   contrast "EZD1 vs. EZD2"    Fluid   1 -1  0  0 0;
   contrast "LZ1 vs. LZ2"      Fluid   0  0  1 -1 0;
   power
      stddev = 3.75
      alpha  = 0.025
      ntotal = .
      power  = 0.9;
run;
```

The CLASS statement identifies Fluid as a classification variable. The MODEL statement specifies the model and the two cell means scenarios LacticAcid1 and LacticAcid2. The WEIGHT statement identifies CellWgt as the weight variable. The CONTRAST statement specifies the contrasts. Since PROC GLMPOWER by default processes class levels in order of formatted values, the contrast coefficients correspond to the following order: EZD1, EZD2, LZ1, LZ2, Water. (**NOTE:** You could use the ORDER=DATA option in the PROC GLMPOWER statement to achieve the same ordering as in Table 55.10 instead.) The POWER statement specifies total sample size as the result parameter and provides values for the other analysis parameters (error standard deviation, alpha, and power).

Output 55.1.1 displays the results.

Output 55.1.1 Sample Sizes for One-Way ANOVA Contrasts

The GLMPOWER Procedure

Fixed Scenario Elements	
Weight Variable	CellWgt
Alpha	0.025
Error Standard Deviation	3.75
Nominal Power	0.9

Output 55.1.1 *continued*

Computed N Total							
Index	Dependent	Type	Source	Test DF	Error DF	Actual Power	N Total
1	LacticAcid1	Effect	Fluid	4	25	0.958	30
2	LacticAcid1	Contrast	Water vs. others	1	25	0.947	30
3	LacticAcid1	Contrast	EZD vs. LZ	1	55	0.929	60
4	LacticAcid1	Contrast	EZD1 vs. EZD2	1	169	0.901	174
5	LacticAcid1	Contrast	LZ1 vs. LZ2	1	217	0.902	222
6	LacticAcid2	Effect	Fluid	4	25	0.972	30
7	LacticAcid2	Contrast	Water vs. others	1	19	0.901	24
8	LacticAcid2	Contrast	EZD vs. LZ	1	43	0.922	48
9	LacticAcid2	Contrast	EZD1 vs. EZD2	1	169	0.901	174
10	LacticAcid2	Contrast	LZ1 vs. LZ2	1	475	0.902	480

The sample sizes range from 24 for the comparison of water versus electrolytes to 480 for the comparison of LZ1 versus LZ2, both assuming the smaller LZ1 mean. The sample size for the latter comparison is relatively large because the small mean difference of 28 - 25.9 = 2.1 is hard to detect. PROC GLMPOWER also includes the effect test for Fluid. Note that, in this case, it is equivalent to TEST=OVERALL_F in the ONEWAYANOVA statement of PROC POWER, since there is only one effect in the model.

The Nominal Power of 0.9 in the "Fixed Scenario Elements" table in Output 55.1.1 represents the input target power, and the Actual Power column in the "Computed N Total" table is the power at the sample size (N Total) adjusted to achieve the specified sample weighting. Note that all of the sample sizes are rounded up to multiples of 6 to preserve integer group sizes (since the group weights add up to 6). You can use the NFRACTIONAL option in the POWER statement to compute raw fractional sample sizes.

Suppose you want to plot the required sample size for the range of power values from 0.5 to 0.95. First, define the analysis by specifying the same statements as before, but add the PLOTONLY option to the PROC GLMPOWER statement to disable the nongraphical results. Next, specify the PLOT statement with X=POWER to request a plot with power on the X axis. (The result parameter—here sample size—is always plotted on the other axis.) Use the MIN= and MAX= options in the PLOT statement to specify the power range. The following statements produce the plot:

```
ods graphics on;

proc glmpower data=Fluids plotonly;
   class Fluid;
   model LacticAcid1 LacticAcid2 = Fluid;
   weight CellWgt;
   contrast "Water vs. others" Fluid  -1 -1 -1 -1 4;
   contrast "EZD vs. LZ"       Fluid   1  1 -1 -1 0;
   contrast "EZD1 vs. EZD2"    Fluid   1 -1  0  0 0;
   contrast "LZ1 vs. LZ2"      Fluid   0  0  1 -1 0;
   power
      stddev = 3.75
      alpha  = 0.025
      ntotal = .
      power  = 0.9;
   plot x=power min=.5 max=.95;
run;
```

Output 55.1.2 Plot of Sample Size versus Power for One-Way ANOVA Contrasts

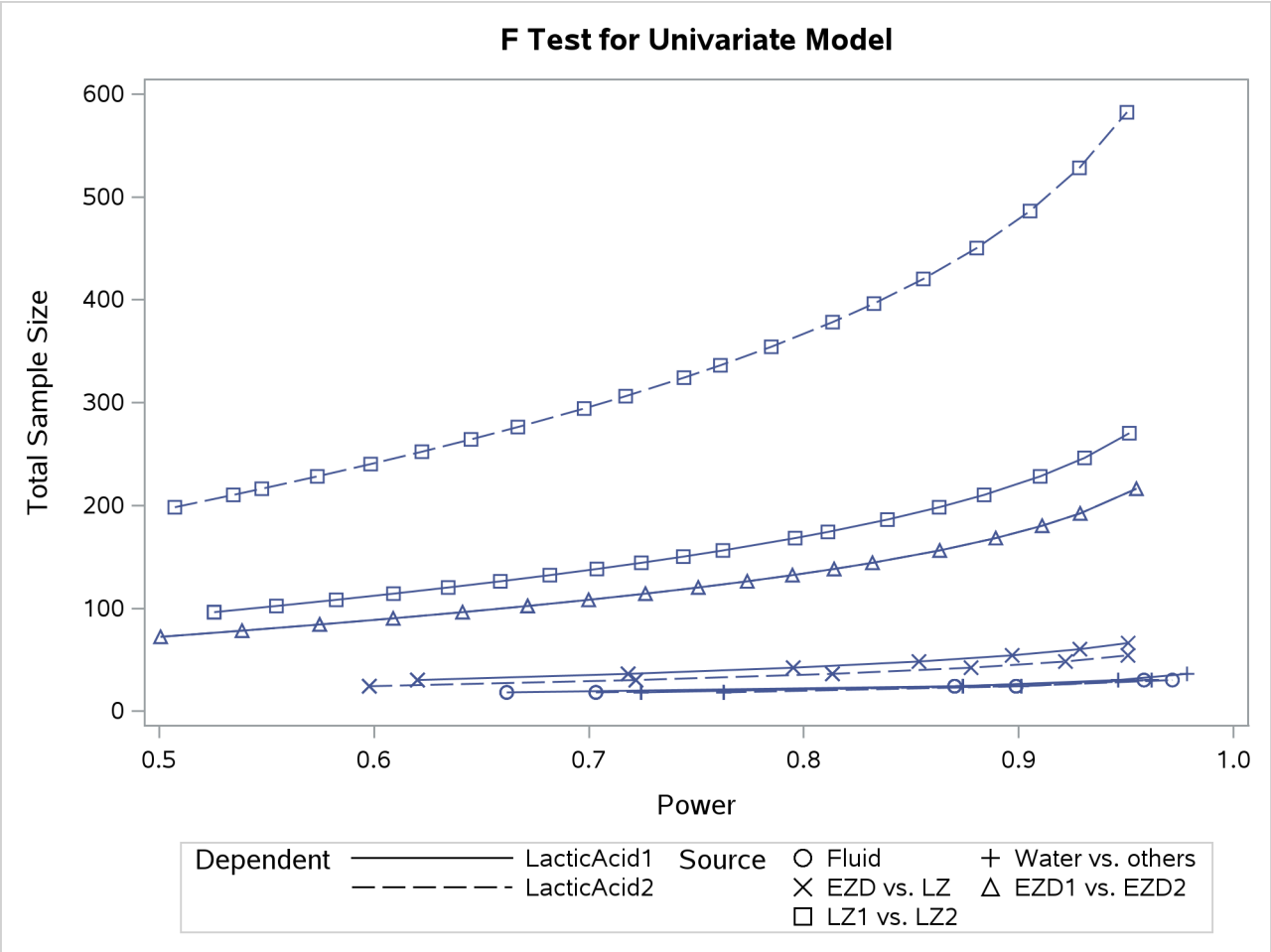

In Output 55.1.2, the line style identifies the cell means scenario, and the plotting symbol identifies the test. The plotting symbol locations identify actual computed powers; the curves are linear interpolations of these points. The plot shows that the required sample size is highest for the test of LZ1 versus LZ2, which was previously found to require the most resources.

Note that some of the plotted points in Output 55.1.2 are unevenly spaced. This is because the plotted points are the *rounded* sample size results at their corresponding *actual* power levels. The range specified with the MIN= and MAX= values in the PLOT statement corresponds to *nominal* power levels. In some cases, actual power is substantially higher than nominal power. To obtain plots with evenly spaced points (but with *fractional* sample sizes at the computed points), you can use the NFRACTIONAL option in the POWER statement preceding the PLOT statement.

Finally, suppose you want to plot the power for the range of sample sizes you will likely consider for the study (the range of 24 to 480 that achieves 0.9 power for different comparisons). In the POWER statement, identify power as the result (POWER=.), and specify any total sample size value (say, NTOTAL=100). Specify the PLOT statement with X=N to request a plot with sample size on the X axis.

The following statements produce the plot:

```
proc glmpower data=Fluids plotonly;
   class Fluid;
   model LacticAcid1 LacticAcid2 = Fluid;
   weight CellWgt;
   contrast "Water vs. others" Fluid  -1 -1 -1 -1 4;
   contrast "EZD vs. LZ"       Fluid   1  1 -1 -1 0;
   contrast "EZD1 vs. EZD2"    Fluid   1 -1  0  0 0;
   contrast "LZ1 vs. LZ2"      Fluid   0  0  1 -1 0;
   power
      stddev = 3.75
      alpha  = 0.025
      ntotal = 24
      power  = .;
   plot x=n min=24 max=480;
run;

ods graphics off;
```

Note that the value 100 specified with the NTOTAL=100 option is not used. It is overridden in the plot by the MIN= and MAX= options in the PLOT statement, and the PLOTONLY option in the PROC GLMPOWER statement disables nongraphical results. But the NTOTAL= option (along with a value) is still needed in the POWER statement as a placeholder, to identify the desired parameterization for sample size.

See Output 55.1.3 for the plot.

Output 55.1.3 Plot of Power versus Sample Size for One-Way ANOVA Contrasts

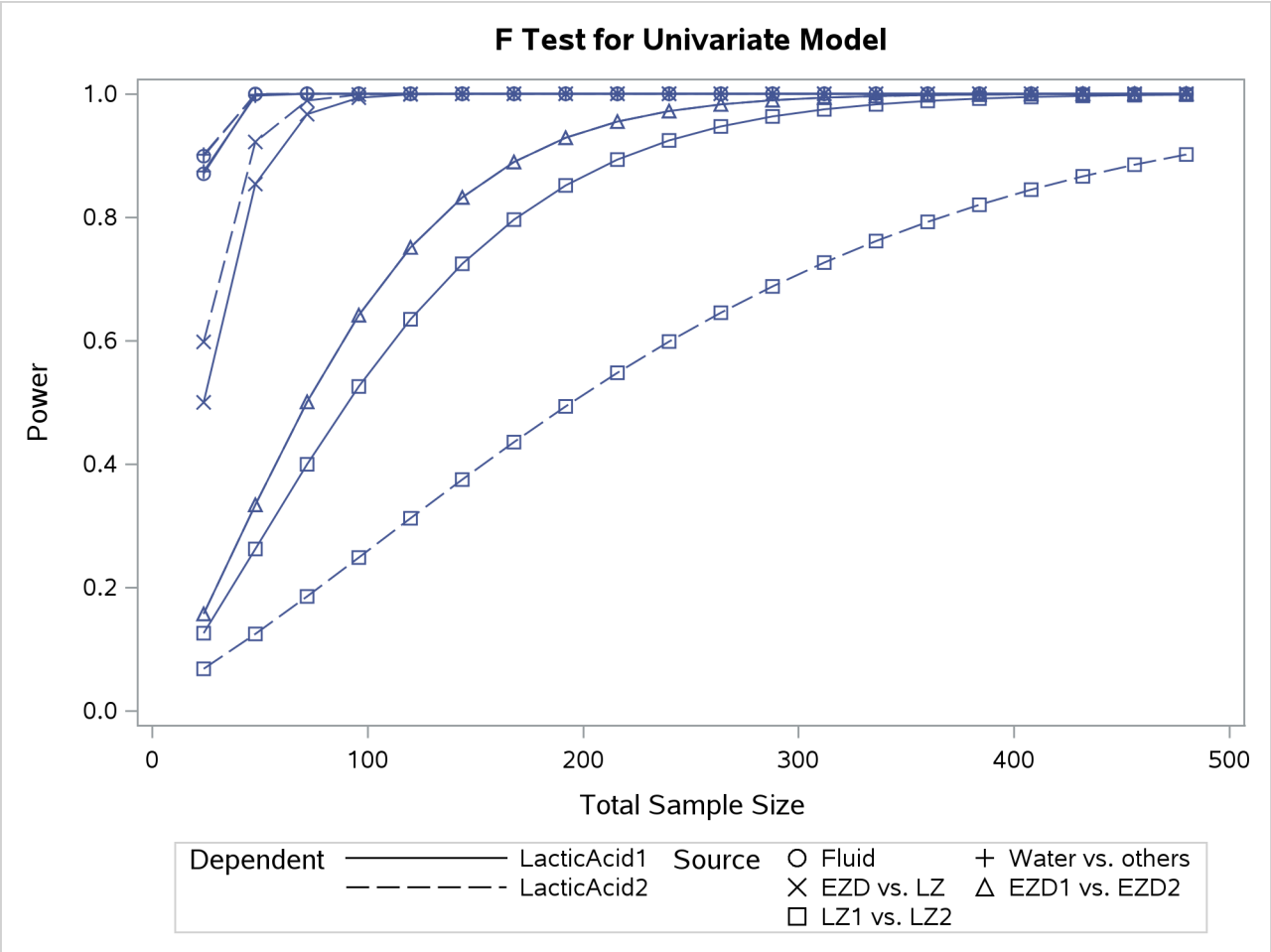

Although Output 55.1.2 and Output 55.1.3 surface essentially the same computations for practical power ranges, they each provide a different quick visual assessment. Output 55.1.2 reveals the range of required sample sizes for powers of interest, and Output 55.1.3 reveals the range of achieved powers for sample sizes of interest.

Example 55.2: Two-Way ANOVA with Covariate

(View the complete code for this example (gmpex02.sas) in the example repository.)

Suppose you can enhance the planned study discussed in Example 55.1 in two ways:

- incorporate results from races at two different altitudes (“high” and “low”)
- measure the body mass index of each runner before the race

This is equivalent to adding a second fixed effect and a continuous covariate to your model.

Since lactic acid buildup is more pronounced at higher altitudes, you will include altitude as a factor in the model along with fluid, extending the one-way ANOVA to a two-way ANOVA. In doing so, you expect to

lower the residual standard deviation from about 3.75 to 3.5 (in addition to generalizing the study results). You assume there is negligible interaction between fluid and altitude and plan to use a main-effects-only model. You conjecture that the mean lactic acid buildup follows Table 55.11.

Table 55.11 Mean Lactic Acid Buildup by Fluid and Altitude

	Fluid				
Altitude	**Water**	**EZD1**	**EZD2**	**LZ1**	**LZ2**
High	36.9	35.0	31.5	30	27.1
Low	34.3	32.4	28.9	27	24.7

By including a measurement of body mass index as a covariate in the study, you hope to further reduce the error variability. The extent of this reduction in variability is commonly expressed in two alternative ways: (1) the correlation between the covariates and the response or (2) the proportional reduction in total R square incurred by the covariates. You prefer the former and guess that the correlation between body mass index and lactic acid buildup is between 0.2 and 0.3. You specify these estimates with the NCOVARIATES= and CORRXY= options in the POWER statement. The covariate is not included in the MODEL statement.

You are interested in the same four fluid comparisons as in Example 55.1, shown in Table 55.10, except this time you want to marginalize over the effect of altitude.

For each of these contrasts, you want to determine the sample size required to achieve a power of 0.9 to detect an effect with magnitude according to Table 55.11. You are not yet attempting to choose a single sample size for the study, but rather checking the range of sample sizes needed by individual contrasts. You plan to test each contrast at $\alpha = 0.025$. You will provide twice as many runners with water as with any of the electrolytes, and you predict that you can study approximately two-thirds as many runners at high altitude than at low altitude. The resulting planned sample size weighting scheme is shown in Table 55.12. Since the scheme is only approximate, you use the NFRACTIONAL option in the POWER statement to disable the rounding of sample sizes up to integers satisfying the weights exactly.

Table 55.12 Approximate Sample Size Allocation Weights

	Fluid				
Altitude	**Water**	**EZD1**	**EZD2**	**LZ1**	**LZ2**
High	4	2	2	2	2
Low	6	3	3	3	3

First, you create the exemplary data set to specify means and weights for the design profiles:

```
data Fluids2;
   input Altitude $ Fluid $ LacticAcid CellWgt;
   datalines;
         High       Water      36.9       4
         High       EZD1       35.0       2
         High       EZD2       31.5       2
         High       LZ1        30         2
         High       LZ2        27.1       2
         Low        Water      34.3       6
         Low        EZD1       32.4       3
```

```
          Low        EZD2       28.9       3
          Low        LZ1        27         3
          Low        LZ2        24.7       3
;
```

The variables Altitude, Fluid, and LacticAcid specify the factors and cell means in Table 55.11. The variable CellWgt contains the sample size allocation weights in Table 55.12.

Use the DATA= option in the PROC GLMPOWER statement to specify Fluids2 as the exemplary data set. The following statements perform the sample size analysis:

```
proc glmpower data=Fluids2;
   class Altitude Fluid;
   model LacticAcid = Altitude Fluid;
   weight CellWgt;
   contrast "Water vs. others" Fluid  -1 -1 -1 -1 4;
   contrast "EZD vs. LZ"       Fluid   1  1 -1 -1 0;
   contrast "EZD1 vs. EZD2"    Fluid   1 -1  0  0 0;
   contrast "LZ1 vs. LZ2"      Fluid   0  0  1 -1 0;
   power
      nfractional
      stddev      = 3.5
      ncovariates = 1
      corrxy      = 0.2 0.3 0
      alpha       = 0.025
      ntotal      = .
      power       = 0.9;
run;
```

The CLASS statement identifies Altitude and Fluid as classification variables. The MODEL statement specifies the model, and the WEIGHT statement identifies CellWgt as the weight variable. The CONTRAST statement specifies the contrasts in Table 55.10. As in Example 55.1, the order of the contrast coefficients corresponds to the formatted class levels (EZD1, EZD2, LZ1, LZ2, Water). The POWER statement specifies total sample size as the result parameter and provides values for the other analysis parameters. The NCOVARIATES= option specifies the single covariate (body mass index), and the CORRXY= option specifies the two scenarios for its correlation with lactic acid buildup (0.2 and 0.3). Output 55.2.1 displays the results.

Output 55.2.1 Sample Sizes for Two-Way ANOVA Contrasts

The GLMPOWER Procedure

Fixed Scenario Elements	
Dependent Variable	LacticAcid
Weight Variable	CellWgt
Alpha	0.025
Number of Covariates	1
Std Dev Without Covariate Adjustment	3.5
Nominal Power	0.9

Output 55.2.1 *continued*

Computed Ceiling N Total									
Index	Type	Source	Corr XY	Adj Std Dev	Test DF	Error DF	Fractional N Total	Actual Power	Ceiling N Total
1	Effect	Altitude	0.2	3.43	1	84	90.418451	0.902	91
2	Effect	Altitude	0.3	3.34	1	79	85.862649	0.901	86
3	Effect	Altitude	0.0	3.50	1	88	94.063984	0.903	95
4	Effect	Fluid	0.2	3.43	4	16	22.446173	0.912	23
5	Effect	Fluid	0.3	3.34	4	15	21.687544	0.908	22
6	Effect	Fluid	0.0	3.50	4	17	23.055716	0.919	24
7	Contrast	Water vs. others	0.2	3.43	1	15	21.720195	0.905	22
8	Contrast	Water vs. others	0.3	3.34	1	14	20.848805	0.903	21
9	Contrast	Water vs. others	0.0	3.50	1	16	22.422381	0.910	23
10	Contrast	EZD vs. LZ	0.2	3.43	1	35	41.657424	0.903	42
11	Contrast	EZD vs. LZ	0.3	3.34	1	33	39.674037	0.903	40
12	Contrast	EZD vs. LZ	0.0	3.50	1	37	43.246415	0.906	44
13	Contrast	EZD1 vs. EZD2	0.2	3.43	1	139	145.613657	0.901	146
14	Contrast	EZD1 vs. EZD2	0.3	3.34	1	132	138.173983	0.902	139
15	Contrast	EZD1 vs. EZD2	0.0	3.50	1	145	151.565917	0.901	152
16	Contrast	LZ1 vs. LZ2	0.2	3.43	1	268	274.055008	0.901	275
17	Contrast	LZ1 vs. LZ2	0.3	3.34	1	253	259.919126	0.900	260
18	Contrast	LZ1 vs. LZ2	0.0	3.50	1	279	285.363976	0.901	286

The sample sizes in Output 55.2.1 range from 21 for the comparison of water versus electrolytes (assuming a correlation of 0.3 between body mass and lactic acid buildup) to 275 for the comparison of LZ1 versus LZ2 (assuming a correlation of 0.2). PROC GLMPOWER also includes the effect tests for Altitude and Fluid. Note that the required sample sizes for this study are lower than those for the study in Example 55.1.

Note that the error standard deviation has been reduced from 3.5 to 3.43 (when correlation is 0.2) or 3.34 (when correlation is 0.3) in the approximation of the effect of the body mass index covariate. The error degrees of freedom has also been automatically adjusted, lowered by 1 (the number of covariates).

Suppose you want to plot the required sample size for the range of power values from 0.5 to 0.95. First, define the analysis by specifying the same statements as before, but add the PLOTONLY option to the PROC GLMPOWER statement to disable the nongraphical results. Next, specify the PLOT statement with X=POWER to request a plot with power on the X axis. Sample size is automatically placed on the Y axis. Use the MIN= and MAX= options in the PLOT statement to specify the power range. The following statements produce the plot:

```
ods graphics on;

proc glmpower data=Fluids2 plotonly;
   class Altitude Fluid;
   model LacticAcid = Altitude Fluid;
   weight CellWgt;
   contrast "Water vs. others" Fluid  -1 -1 -1 -1 4;
   contrast "EZD vs. LZ"       Fluid   1  1 -1 -1 0;
   contrast "EZD1 vs. EZD2"    Fluid   1 -1  0  0 0;
   contrast "LZ1 vs. LZ2"      Fluid   0  0  1 -1 0;
   power
```

```
        nfractional
        stddev      = 3.5
        ncovariates = 1
        corrxy      = 0.2 0.3 0
        alpha       = 0.025
        ntotal      = .
        power       = 0.9;
    plot x=power min=.5 max=.95;
  run;
```

See Output 55.2.2 for the resulting plot.

Output 55.2.2 Plot of Sample Size versus Power for Two-Way ANOVA Contrasts

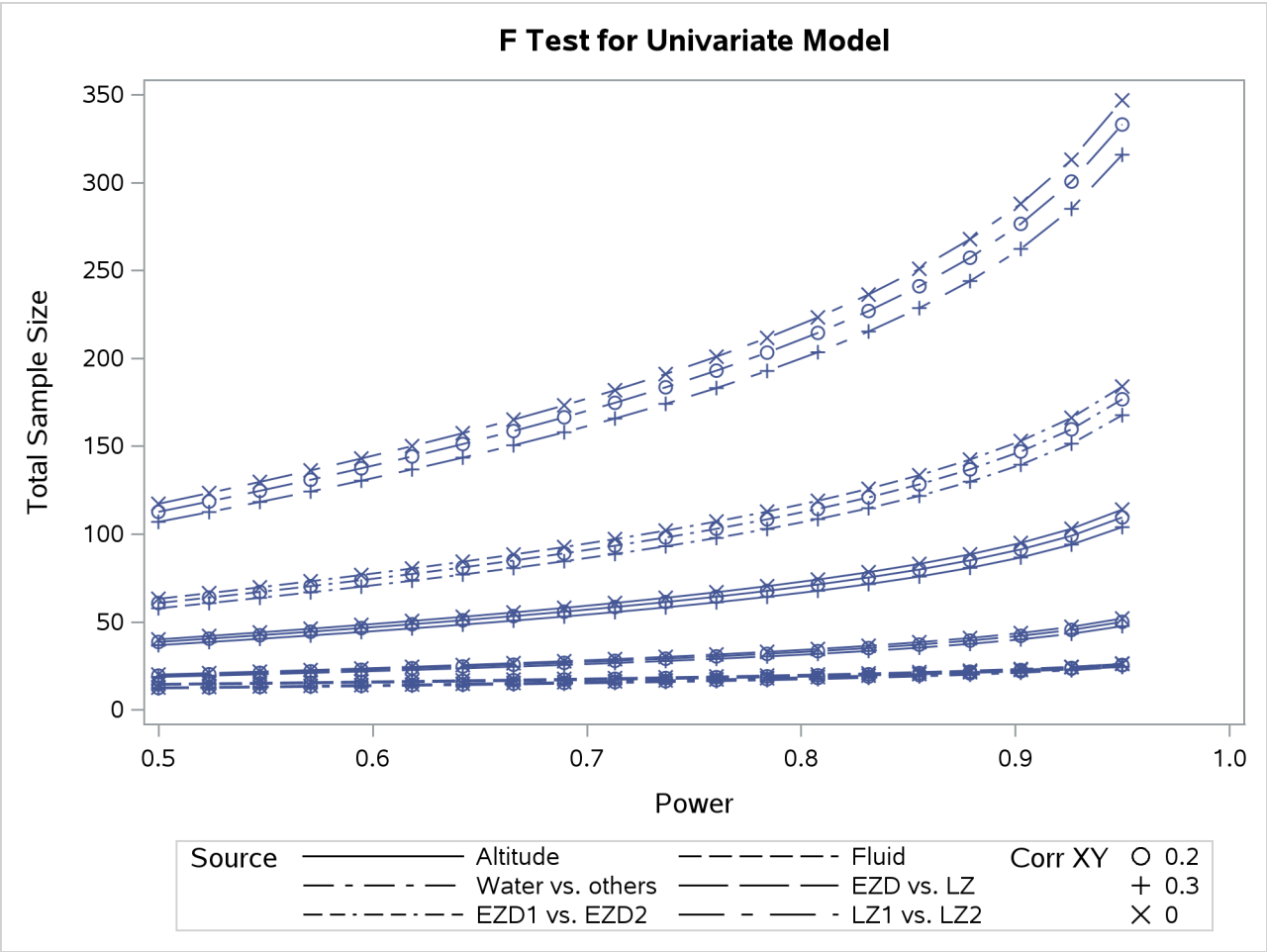

In Output 55.1.2, the line style identifies the test, and the plotting symbol identifies the scenario for the correlation between covariate and response. The plotting symbol locations identify actual computed powers; the curves are linear interpolations of these points. As in Example 55.1, the required sample size is highest for the test of LZ1 versus LZ2.

Finally, suppose you want to plot the power for the range of sample sizes you will likely consider for the study (the range of 21 to 275 that achieves 0.9 power for different comparisons). In the POWER statement, identify power as the result (POWER=.), and specify NTOTAL=21. Specify the PLOT statement with X=N to request a plot with sample size on the X axis.

The following statements produce the plot:

```
proc glmpower data=Fluids2 plotonly;
   class Altitude Fluid;
   model LacticAcid = Altitude Fluid;
   weight CellWgt;
   contrast "Water vs. others" Fluid  -1 -1 -1 -1 4;
   contrast "EZD vs. LZ"       Fluid   1  1 -1 -1 0;
   contrast "EZD1 vs. EZD2"    Fluid   1 -1  0  0 0;
   contrast "LZ1 vs. LZ2"      Fluid   0  0  1 -1 0;
   power
      nfractional
      stddev      = 3.5
      ncovariates = 1
      corrxy      = 0.2 0.3 0
      alpha       = 0.025
      ntotal      = 21
      power       = .;
   plot x=n min=21 max=275;
run;

ods graphics off;
```

The MAX=275 option in the PLOT statement sets the maximum sample size value. The MIN= option automatically defaults to the value of 21 from the NTOTAL= option in the POWER statement.

See Output 55.2.3 for the plot.

Output 55.2.3 Plot of Power versus Sample Size for Two-Way ANOVA Contrasts

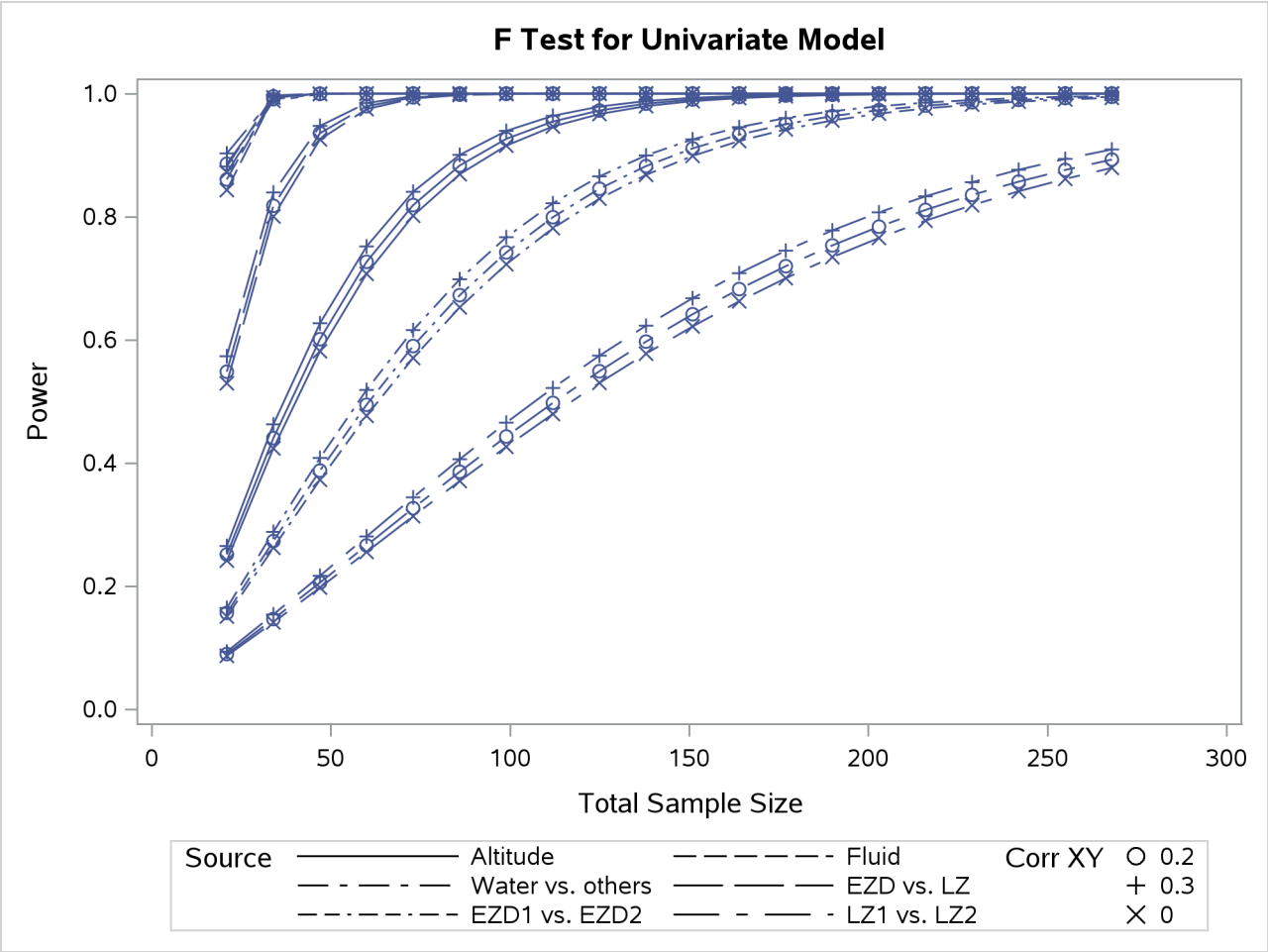

Although Output 55.2.2 and Output 55.2.3 surface essentially the same computations for practical power ranges, they each provide a different quick visual assessment. Output 55.2.2 reveals the range of required sample sizes for powers of interest, and Output 55.2.3 reveals the range of powers achieved for sample sizes of interest.

Example 55.3: Repeated Measures ANOVA

(View the complete code for this example (gmpex03.sas) in the example repository.)

Logan, Baron, and Kohout (1995) and Guo et al. (2013) study the effect of a dental intervention on the memory of pain after root canal therapy. The intervention is a sensory focus strategy, in which patients are instructed to pay attention only to the physical sensations in their mouth during the root canal procedure.

Suppose you are interested in the long-term effects of this sensory focus intervention, because avoidance behavior has been shown to build along with memory of pain. You are planning a study to compare sensory focus to standard of care over a period of a year, asking patients to self-report their memory of pain immediately after the procedure and then again at 1 week, 6 months, and 12 months. You use a scale from 0 (no pain remembered) to 5 (maximum pain remembered).

The between-subject factor in your model is treatment, with two levels (sensory focus versus standard of care), and you allocate each treatment equally for a balanced design. The within-subject factor is time, with four levels (0, 1, 26, and 52 weeks).

You want to determine the number of patients who are needed in order to achieve a power of 0.9 at significance level $\alpha = 0.01$ for the test of the interaction between time and treatment, where the contrast over time contains all pairwise comparisons. You also want to generate a plot of power versus sample size that covers the power range of 0.05 to 0.99.

The default Hotelling-Lawley F test is appropriate for this study, especially because it is the same as the Wald test in PROC MIXED with the DDFM=KR Kenward-Roger degrees-of-freedom method and an unstructured covariance model.

You conjecture that the mean memory of pain for each treatment follows the information in Table 55.13.

Table 55.13 Mean Memory of Pain by Treatment

	Time Since Root Canal Therapy			
Treatment	**Later on Same Day**	**1 Week**	**6 Months**	**12 Months**
Sensory Focus	2.40	2.38	2.05	1.90
Standard of Care	2.40	2.39	2.36	2.30

The following statements create a data set named Pain that is to contain these means over treatment and time:

```
data Pain;
   input Treatment $ PainMem0 PainMem1Wk PainMem6Mo PainMem12Mo;
   datalines;
      SensoryFocus    2.40  2.38  2.05  1.90
      StandardOfCare  2.40  2.39  2.36  2.30
;
```

The variable Treatment specifies the two treatments. The four variables PainMem0, PainMem1Wk, PainMem6Mo, and PainMem12Mo specify the mean memory of pain scores in Table 55.13.

To characterize the variability, you must specify a set of parameters that defines the entire covariance matrix of the residuals. You conjecture that the error standard deviation is the same at all four time points, with a value somewhere between 0.92 and 1.04, and you account for your uncertainty by including both the lower and upper ends of this range in the sample size analysis. You believe that the correlation has a linear exponent autoregressive (LEAR) structure, with a correlation of about 0.6 between measurements one week apart and a decay rate of about 0.8 over one-week intervals. The correlation matrix that contains these LEAR parameters, rounded to three decimal places, is shown in Table 55.14.

Table 55.14 Conjectured Correlation Matrix

	0	**1**	**26**	**52**
0	1	0.6	0.491	0.399
1	0.6	1	0.495	0.402
26	0.491	0.495	1	0.491
52	0.399	0.402	0.491	1

Use the DATA= option in the PROC GLMPOWER statement to specify Pain as the exemplary data set. Specify the between- and within-subject factors and the model by using the CLASS, MODEL, and REPEATED statements just as you would in PROC GLM for the repeated measures data analysis. Use the POWER statement to indicate sample size as the result parameter and specify the other analysis parameters, and use the PLOT statement to generate the power curves. The following statements perform the sample size analysis:

```
ods graphics on;

proc glmpower data=Pain;
   class Treatment;
   model PainMem0 PainMem1Wk PainMem6Mo PainMem12Mo = Treatment;
   repeated Time contrast;
   power
      mtest = hlt
      alpha = 0.01
      power = .9
      ntotal = .
      stddev = 0.92 1.04
      matrix ("PainCorr") = lear(0.6, 0.8, 4, 0 1 26 52)
      corrmat = "PainCorr";
   plot y=power min=0.05 max=0.99 yopts=(ref=0.9)
      vary (linestyle by stddev, symbol by dependent source);
run;
ods graphics off;
```

The STDDEV= option specifies the two scenarios for the common residual standard deviation, 0.92 and 1.04. The MATRIX= option defines the LEAR correlation structure, and the CORRMAT= option specifies it as the correlation matrix of the residuals. The Y=POWER option in the PLOT statement requests a plot that has power on the Y axis. (The result parameter—in this case, total sample size—is always plotted on the other axis.) The MIN= and MAX= options in the PLOT statement specify the power range. The YOPTS=(REF=) option adds a reference line at the target power value of 0.9. The VARY option specifies that the line style vary by the residual standard deviation and that the plotting symbol vary by the combination of within-subject and between-subject effects. The ODS GRAPHICS ON statement enables ODS Graphics.

Output 55.3.1 shows the output, and Output 55.3.2 shows the plot.

Output 55.3.1 Sample Size Analysis for Repeated Measures

The GLMPOWER Procedure
F Test for Multivariate Model

Fixed Scenario Elements	
Wilks/HLT/PT Method	O'Brien-Shieh
F Test	Hotelling-Lawley Trace
Alpha	0.01
Correlation Matrix	PainCorr
Nominal Power	0.9

Output 55.3.1 *continued*

Computed N Total								
Index	Transformation	Source	Std Dev	Effect	Num DF	Den DF	Actual Power	N Total
1	Time	Intercept	0.92	Time	3	176	0.900	180
2	Time	Intercept	1.04	Time	3	226	0.903	230
3	Time	Treatment	0.92	Time*Treatment	3	346	0.901	350
4	Time	Treatment	1.04	Time*Treatment	3	442	0.901	446
5	Mean(Dep)	Intercept	0.92	Intercept	1	4	0.960	6
6	Mean(Dep)	Intercept	1.04	Intercept	1	4	0.907	6
7	Mean(Dep)	Treatment	0.92	Treatment	1	950	0.900	952
8	Mean(Dep)	Treatment	1.04	Treatment	1	1214	0.900	1216

Output 55.3.1 reveals that the required sample size to achieve a power of 0.9 for the test of the Time*Treatment interaction is 350 for the error standard deviation of 0.92 and 446 for the error standard deviation of 1.04.

Output 55.3.2 Plot of Power versus Sample Size for Repeated Measures Analysis

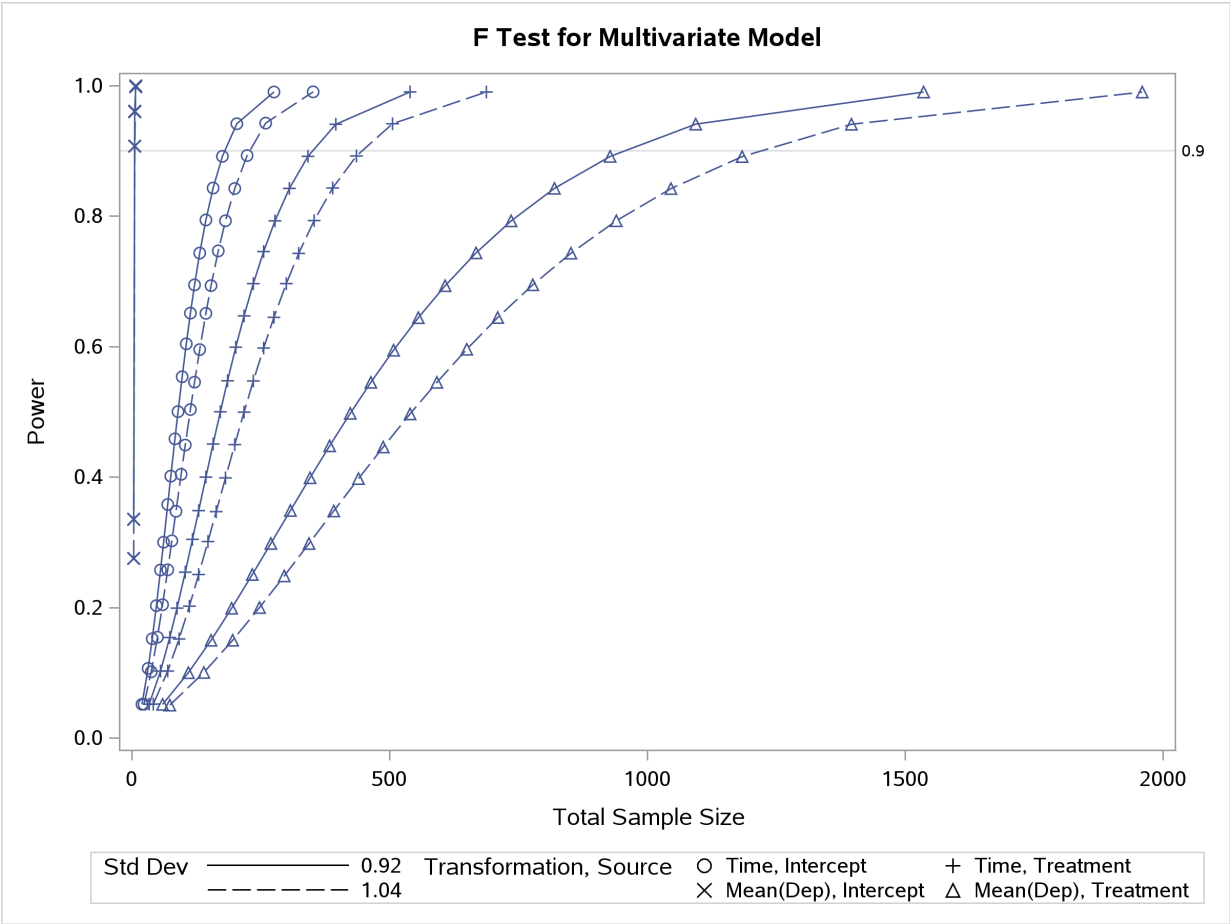

References

Castelloe, J. M. (2000). "Sample Size Computations and Power Analysis with the SAS System." In *Proceedings of the Twenty-Fifth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi25/25/st/25p265.pdf.

Castelloe, J. M., and O'Brien, R. G. (2001). "Power and Sample Size Determination for Linear Models." In *Proceedings of the Twenty-Sixth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi26/p240-26.pdf.

Chi, Y.-Y., Gribbin, M. J., Lamers, Y., Gregory, J. F., III, and Muller, K. E. (2012). "Global Hypothesis Testing for High-Dimensional Repeated Measures Outcomes." *Statistics in Medicine* 31:724–742.

Guo, Y., Logan, H. L., Glueck, D. H., and Muller, K. E. (2013). "Selecting a Sample Size for Studies with Repeated Measures." *BMC Medical Research Methodology* 13:100. https://doi.org/10.1186/1471-2288-13-100.

Hocking, R. R. (1985). *The Analysis of Linear Models*. Monterey, CA: Brooks/Cole.

Huynh, H., and Feldt, L. S. (1976). "Estimation of the Box Correction for Degrees of Freedom from Sample Data in the Randomized Block and Split Plot Designs." *Journal of Educational Statistics* 1:69–82.

Lecoutre, B. (1991). "A Correction for the Epsilon Approximate Test with Repeated Measures Design with Two or More Independent Groups." *Journal of Educational Statistics* 16:371–372.

Lenth, R. V. (2001). "Some Practical Guidelines for Effective Sample Size Determination." *American Statistician* 55:187–193.

Logan, H. L., Baron, R. S., and Kohout, F. (1995). "Sensory Focus as Therapeutic Treatments for Acute Pain." *Psychosomatic Medicine* 57:475–484.

McKeon, J. J. (1974). "F Approximations to the Distribution of Hotelling's T_0^2." *Biometrika* 61:381–383.

Muller, K. E., and Barton, C. N. (1989). "Approximate Power for Repeated-Measures ANOVA Lacking Sphericity." *Journal of the American Statistical Association* 84:549–555. Also see "Correction to 'Approximate Power for Repeated-Measures ANOVA Lacking Sphericity,'" *Journal of the American Statistical Association* 86 (1991): 255–256.

Muller, K. E., and Benignus, V. A. (1992). "Increasing Scientific Power with Statistical Power." *Neurotoxicology and Teratology* 14:211–219.

Muller, K. E., Edwards, L. J., Simpson, S. L., and Taylor, D. J. (2007). "Statistical Tests with Accurate Size and Power for Balanced Linear Mixed Models." *Statistics in Medicine* 26:3639–3660.

Muller, K. E., and Peterson, B. L. (1984). "Practical Methods for Computing Power in Testing the Multivariate General Linear Hypothesis." *Computational Statistics and Data Analysis* 2:143–158.

O'Brien, R. G., and Castelloe, J. M. (2007). "Sample-Size Analysis for Traditional Hypothesis Testing: Concepts and Issues." In *Pharmaceutical Statistics Using SAS: A Practical Guide*, edited by A. Dmitrienko, C. Chuang-Stein, and R. D'Agostino, 237–271. Cary, NC: SAS Institute Inc.

O'Brien, R. G., and Muller, K. E. (1993). "Unified Power Analysis for t-Tests through Multivariate Hypotheses." In *Applied Analysis of Variance in Behavioral Science*, edited by L. K. Edwards, 297–344. New York: Marcel Dekker.

O'Brien, R. G., and Shieh, G. (1992). "Pragmatic, Unifying Algorithm Gives Power Probabilities for Common F Tests of the Multivariate General Linear Hypothesis." Poster presented at the American Statistical Association Meetings, Statistical Computing Section, Boston.

Pillai, K. C. S., and Samson, P., Jr. (1959). "On Hotelling's Generalization of T^2." *Biometrika* 46:160–168.

Simpson, S. L., Edwards, L. J., Muller, K. E., Sen, P. K., and Styner, M. A. (2010). "A Linear Exponent AR(1) Family of Correlation Structures." *Statistics in Medicine* 29:1825–1838.

Chapter 56
The GLMSELECT Procedure

Contents

Overview: GLMSELECT Procedure **4414**
Features 4415
Getting Started: GLMSELECT Procedure **4416**
Syntax: GLMSELECT Procedure **4423**
PROC GLMSELECT Statement 4424
BY Statement 4432
CLASS Statement 4432
CODE Statement 4436
EFFECT Statement 4436
FREQ Statement 4438
MODEL Statement 4438
MODELAVERAGE Statement 4450
OUTPUT Statement 4453
PARTITION Statement 4455
PERFORMANCE Statement 4456
SCORE Statement 4456
STORE Statement 4457
WEIGHT Statement 4457
Details: GLMSELECT Procedure **4458**
Model-Selection Methods 4458
Full Model Fitted (NONE) 4458
Forward Selection (FORWARD) 4458
Backward Elimination (BACKWARD) 4461
Stepwise Selection (STEPWISE) 4462
Least Angle Regression (LAR) 4463
Lasso Selection (LASSO) 4464
Adaptive LASSO Selection 4465
Elastic Net Selection (ELASTICNET) 4465
Group LASSO Selection (GROUPLASSO) 4467
Model Selection Issues 4468
Criteria Used in Model Selection Methods 4469
CLASS Variable Parameterization and the SPLIT Option 4471
Macro Variables Containing Selected Models 4472
Using the STORE Statement 4475
Building the SSCP Matrix 4475
Model Averaging 4476

Using Validation and Test Data 4478
Cross Validation 4480
External Cross Validation 4481
Screening 4484
Displayed Output 4486
ODS Table Names 4490
ODS Graphics 4491
Examples: GLMSELECT Procedure **4498**
Example 56.1: Modeling Baseball Salaries Using Performance Statistics 4499
Example 56.2: Using Validation and Cross Validation 4510
Example 56.3: Scatter Plot Smoothing by Selecting Spline Functions 4525
Example 56.4: Multimember Effects and the Design Matrix 4533
Example 56.5: Model Averaging 4539
Example 56.6: Elastic Net and External Cross Validation 4548
Example 56.7: LASSO with Screening 4561
Example 56.8: Group LASSO Selection 4566
References **4573**

Overview: GLMSELECT Procedure

The GLMSELECT procedure performs effect selection in the framework of general linear models. A variety of model selection methods are available, including the LASSO method of Tibshirani (1996) and the related LAR method of Efron et al. (2004). The procedure offers extensive capabilities for customizing the selection with a wide variety of selection and stopping criteria, from traditional and computationally efficient significance-level-based criteria to more computationally intensive validation-based criteria. The procedure also provides graphical summaries of the selection search.

The GLMSELECT procedure compares most closely to REG and GLM. The REG procedure supports a variety of model-selection methods but does not support a CLASS statement. The GLM procedure supports a CLASS statement but does not include effect selection methods. The GLMSELECT procedure fills this gap. GLMSELECT focuses on the standard independently and identically distributed general linear model for univariate responses and offers great flexibility for and insight into the model selection algorithm. GLMSELECT provides results (displayed tables, output data sets, and macro variables) that make it easy to take the selected model and explore it in more detail in a subsequent procedure such as REG or GLM.

Features

The main features of the GLMSELECT procedure are as follows:

- **Model Specification**
  - supports different parameterizations for classification effects
  - supports any degree of interaction (crossed effects) and nested effects
  - supports hierarchy among effects
  - supports partitioning of data into training, validation, and testing roles
  - supports constructed effects including spline and multimember effects
- **Selection Control**
  - provides multiple effect selection methods
  - enables selection from a very large number of effects (tens of thousands)
  - offers selection of individual levels of classification effects
  - provides effect selection based on a variety of selection criteria
  - provides stopping rules based on a variety of model evaluation criteria
  - provides leave-one-out, k-fold cross validation, and k-fold external cross validation
  - supports data resampling and model averaging
- **Display and Output**
  - produces graphical representation of selection process
  - produces output data sets containing predicted values and residuals
  - produces an output data set containing the design matrix
  - produces macro variables containing selected models
  - supports parallel processing of BY groups
  - supports multiple SCORE statements

The GLMSELECT procedure supports the following effect selection methods. For more information about these methods, see the section "Model-Selection Methods" on page 4458.

forward selection	starts with no effects in the model and adds effects.
backward elimination	starts with all effects in the model and deletes effects.
stepwise regression	is similar to forward selection except that effects already in the model do not necessarily stay there.
least angle regression (LAR)	is similar to forward selection in that it starts with no effects in the model and adds effects. The parameter estimates at any step are "shrunk" when compared to the corresponding least squares estimates.

LASSO	adds and deletes parameters based on a version of ordinary least squares where the sum of the absolute regression coefficients is constrained.
elastic net	is an extension of LASSO that estimates parameters based on a version of ordinary least squares in which both the sum of the absolute regression coefficients and the sum of the squared regression coefficients are constrained.
group LASSO	is a variant of LASSO that estimates parameters based on a version of ordinary least squares in which the sum of the Euclidean norms of a group of regression coefficients is constrained.

PROC GLMSELECT also supports hybrid versions of the LAR and LASSO methods. They use LAR and LASSO to select the model but then estimate the regression coefficients by ordinary weighted least squares.

The GLMSELECT procedure is intended primarily as a model selection procedure and does not include regression diagnostics or other postselection facilities such as hypothesis testing, testing of contrasts, and LS-means analyses. The intention is that you use PROC GLMSELECT to select a model or a set of candidate models. Further investigation of these models can be done by using these models in existing regression procedures.

Getting Started: GLMSELECT Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glsgs.sas.)

The Sashelp.Baseball data set contains salary and performance information for Major League Baseball players who played at least one game in both the 1986 and 1987 seasons, excluding pitchers. The salaries (*Sports Illustrated,* April 20, 1987) are for the 1987 season and the performance measures are from 1986 (Collier Books, *The 1987 Baseball Encyclopedia Update*). The following step displays in Figure 56.1 the variables in the data set:

```
proc contents varnum data=sashelp.baseball;
   ods select position;
run;
```

Suppose you want to investigate whether you can model the players' salaries for the 1987 season based on performance measures for the previous season. The aim is to obtain a parsimonious model that does not overfit this particular data, making it useful for prediction. This example shows how you can use PROC GLMSELECT as a starting point for such an analysis. Since the variation of salaries is much greater for the higher salaries, it is appropriate to apply a log transformation to the salaries before doing the model selection.

The following code selects a model with the default settings:

```
ods graphics on;
proc glmselect data=sashelp.baseball plots=all;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / details=all stats=all;
run;
```

```
ods graphics off;
```

PROC GLMSELECT performs effect selection where effects can contain classification variables that you specify in a CLASS statement. The "Class Level Information" table shown in Figure 56.2 lists the levels of the classification variables Division and League.

Figure 56.1 Sashelp.Baseball Data Set

The CONTENTS Procedure

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	Name	Char	18	Player's Name
2	Team	Char	14	Team at the End of 1986
3	nAtBat	Num	8	Times at Bat in 1986
4	nHits	Num	8	Hits in 1986
5	nHome	Num	8	Home Runs in 1986
6	nRuns	Num	8	Runs in 1986
7	nRBI	Num	8	RBIs in 1986
8	nBB	Num	8	Walks in 1986
9	YrMajor	Num	8	Years in the Major Leagues
10	CrAtBat	Num	8	Career Times at Bat
11	CrHits	Num	8	Career Hits
12	CrHome	Num	8	Career Home Runs
13	CrRuns	Num	8	Career Runs
14	CrRbi	Num	8	Career RBIs
15	CrBB	Num	8	Career Walks
16	League	Char	8	League at the End of 1986
17	Division	Char	8	Division at the End of 1986
18	Position	Char	8	Position(s) in 1986
19	nOuts	Num	8	Put Outs in 1986
20	nAssts	Num	8	Assists in 1986
21	nError	Num	8	Errors in 1986
22	Salary	Num	8	1987 Salary in $ Thousands
23	Div	Char	16	League and Division
24	logSalary	Num	8	Log Salary

Figure 56.2 Class Level Information

The GLMSELECT Procedure

Class Level Information		
Class	**Levels**	**Values**
League	2	American National
Division	2	East West

When you specify effects that contain classification variables, the number of parameters is usually larger than the number of effects. The "Dimensions" table in Figure 56.3 shows the number of effects and the number of parameters considered.

Figure 56.3 Dimensions

Dimensions	
Number of Effects	19
Number of Parameters	21

Figure 56.4 Model Information

The GLMSELECT Procedure

Data Set	SASHELP.BASEBALL
Dependent Variable	logSalary
Selection Method	Stepwise
Select Criterion	SBC
Stop Criterion	SBC
Effect Hierarchy Enforced	None

You find details of the default search settings in the "Model Information" table shown in Figure 56.4. The default selection method is a variant of the traditional stepwise selection where the decisions about what effects to add or drop at any step and when to terminate the selection are both based on the Schwarz Bayesian information criterion (SBC). The effect in the current model whose removal yields the maximal decrease in the SBC statistic is dropped provided this lowers the SBC value. Once no decrease in the SBC value can be obtained by dropping an effect in the model, the effect whose addition to the model yields the lowest SBC statistic is added and the whole process is repeated. The method terminates when dropping or adding any effect increases the SBC statistic.

Figure 56.5 Candidates for Entry at Step Two

Best 10 Entry Candidates		
Rank	**Effect**	**SBC**
1	**nHits**	-252.5794
2	**nAtBat**	-241.5789
3	**nRuns**	-240.1010
4	**nRBI**	-232.2880
5	**nBB**	-223.3741
6	**nHome**	-208.0565
7	**nOuts**	-205.8107
8	**Division**	-194.4688
9	**CrBB**	-191.5141
10	**nAssts**	-190.9425

The DETAILS=ALL option requests details of each step of the selection process. The "Best 10 Entry Candidates" table at each step shows the candidates for inclusion or removal at that step ranked from best to worst in terms of the selection criterion, which in this example is the SBC statistic. By default only the 10 best candidates are shown. Figure 56.5 shows the candidate table at step two.

To help in the interpretation of the selection process, you can use graphics supported by PROC GLMSELECT. ODS Graphics must be enabled before requesting plots. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." With ODS Graphics enabled, the PLOTS=ALL option

together with the DETAILS=STEPS option in the MODEL statement produces a needle plot view of the "Candidates" tables. The plot corresponding to the "Candidates" table at step two is shown in Figure 56.6. You can see that adding the effect `'nHits'` yields the smallest SBC value, and so this effect is added at step two.

Figure 56.6 Needle Plot of Entry Candidates at Step Two

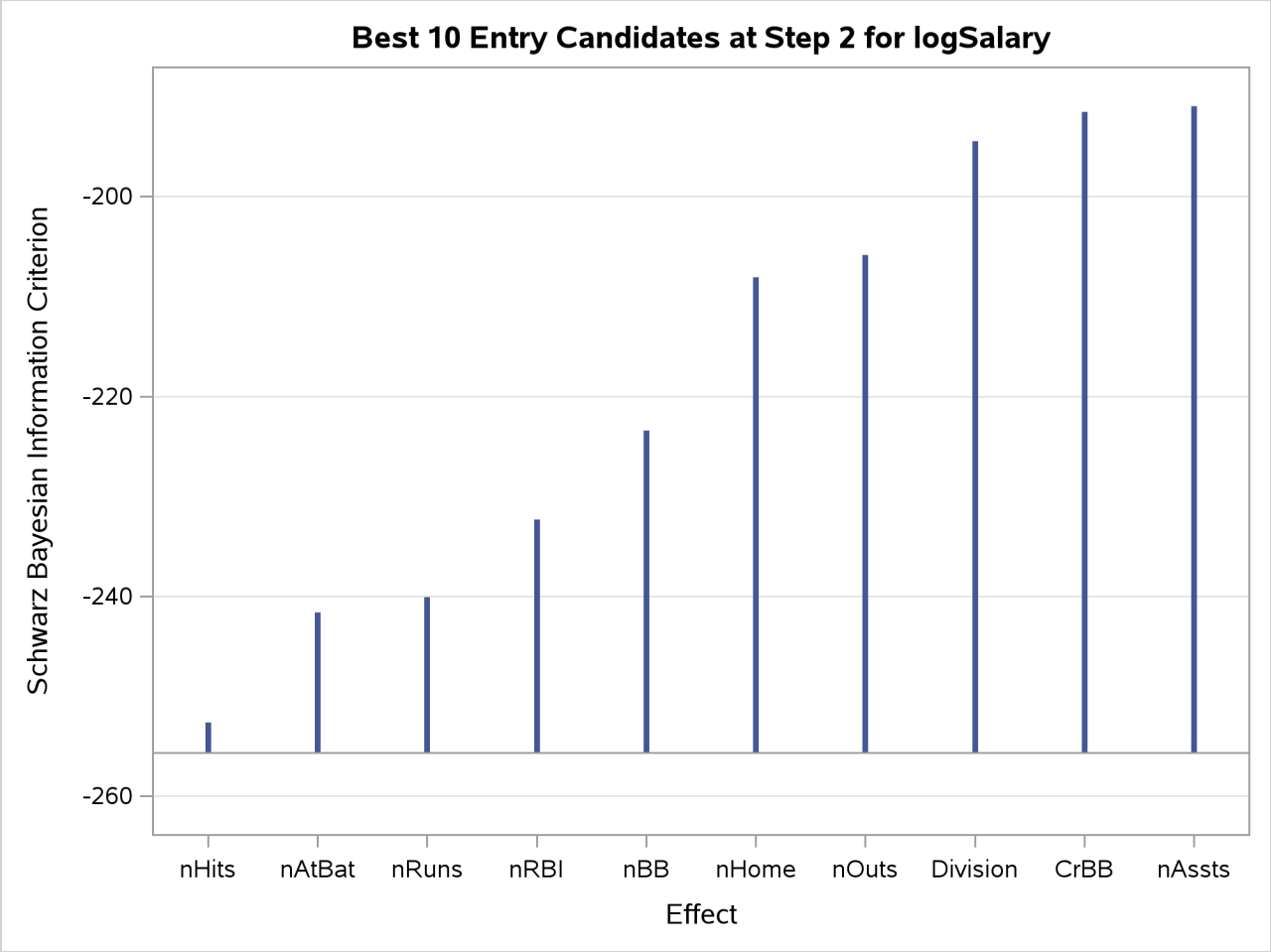

The "Stepwise Selection Summary" table in Figure 56.7 shows the effect that was added or dropped at each step of the selection process together with fit statistics for the model at each step. The STATS=ALL option in the MODEL statement requests that all the available fit statistics are displayed. See the section "Criteria Used in Model Selection Methods" on page 4469 for descriptions and formulas. The criterion panel in Figure 56.8 provides a graphical view of the progression of these fit criteria as the selection process evolves. Note that none of these criteria has a local optimum before step five.

Figure 56.7 Selection Summary Table

The GLMSELECT Procedure

Stepwise Selection Summary											
Step	Effect Entered	Effect Removed	Number Effects In	Number Parms In	Model R-Square	Adjusted R-Square	AIC	AICC	BIC	CP	SBC
0	Intercept		1	1	0.0000	0.0000	204.2238	204.2699	-60.6397	375.9275	-57.2041
1	CrRuns		2	2	0.4187	0.4165	63.5391	63.6318	-200.7872	111.2315	-194.3166
2	nHits		3	3	0.5440	0.5405	1.7041	1.8592	-261.8807	33.4438	-252.5794
3	YrMajor		4	4	0.5705	0.5655	-12.0208	-11.7873	-275.3333	18.5870	-262.7322
4		CrRuns	3	3	0.5614	0.5581	-8.5517	-8.3967	-271.9095	22.3357	-262.8353
5	nBB		4	4	0.5818	0.5770*	-19.0690*	-18.8356*	-282.1700*	11.3524*	-269.7804*
* Optimal Value of Criterion											

Stepwise Selection Summary						
Step	Effect Entered	Effect Removed	PRESS	ASE	F Value	Pr > F
0	Intercept		208.7381	0.7877	0.00	1.0000
1	CrRuns		123.9195	0.4578	188.01	<.0001
2	nHits		97.6368	0.3592	71.42	<.0001
3	YrMajor		92.2998	0.3383	15.96	<.0001
4		CrRuns	93.1482	0.3454	5.44	0.0204
5	nBB		89.5434*	0.3294	12.62	0.0005
* Optimal Value of Criterion						

The stop reason and stop details tables in Figure 56.9 gives details of why the selection process terminated. This table shows that at step five the best add candidate, `'Division'`, and the best drop candidate, `'nBB'`, yield models with SBC values of –268.6094 and –262.8353, respectively. Both of these values are larger than the current SBC value of –269.7804, and so the selection process stops at the model at step five.

Figure 56.8 Criterion Panel

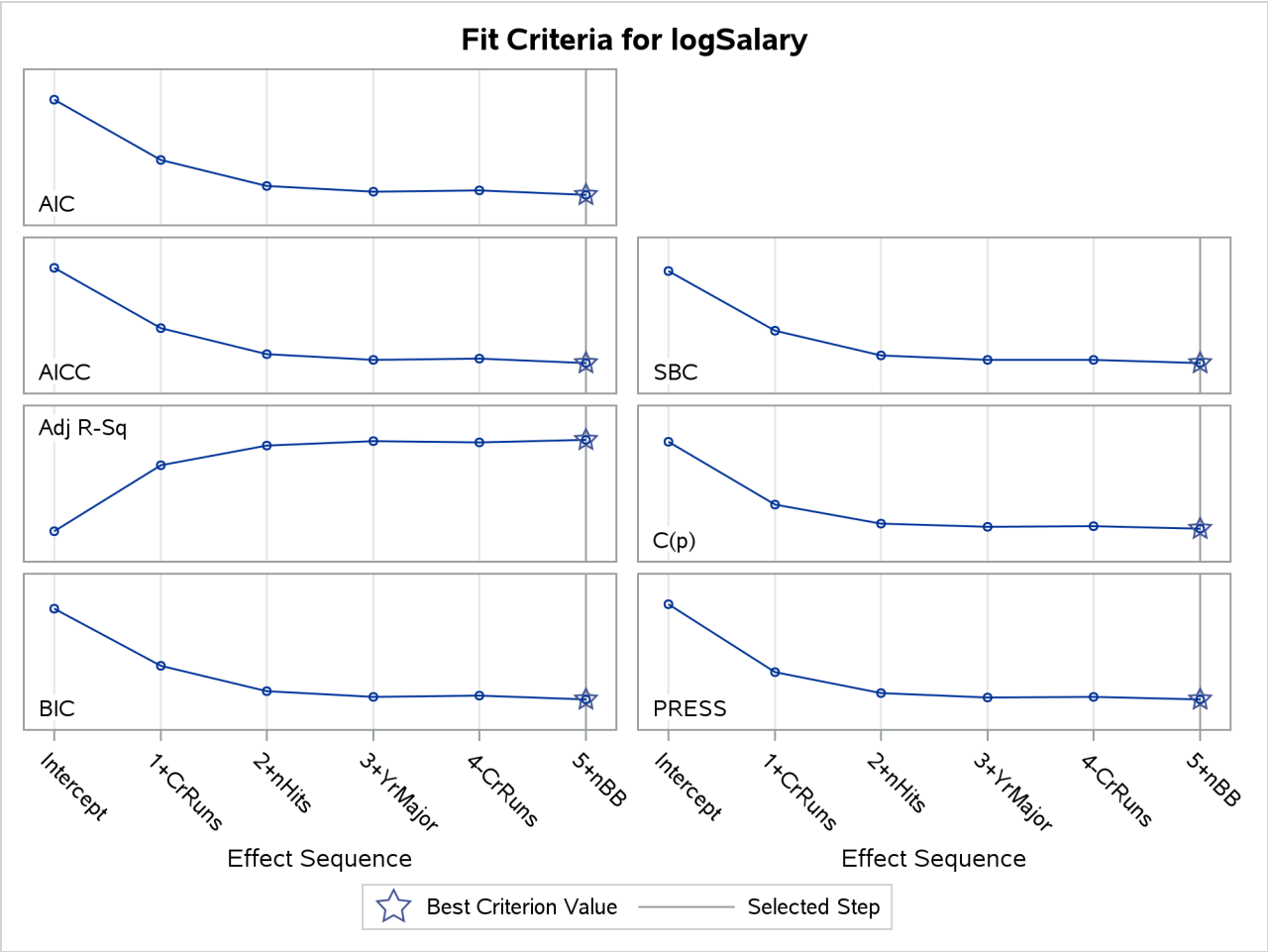

Figure 56.9 Stopping Details

Selection stopped at a local minimum of the SBC criterion.

Stop Details				
Candidate For	**Effect**	**Candidate SBC**		**Compare SBC**
Entry	Division	-268.6094	>	-269.7804
Removal	nBB	-262.8353	>	-269.7804

The coefficient panel in Figure 56.10 enables you to visualize the selection process. In this plot, standardized coefficients of all the effects selected at some step of the stepwise method are plotted as a function of the step number. This enables you to assess the relative importance of the effects selected at any step of the selection process as well as providing information as to when effects entered the model. The lower plot in the panel shows how the criterion used to choose the selected model changes as effects enter or leave the model.

Figure 56.10 Coefficient Progression

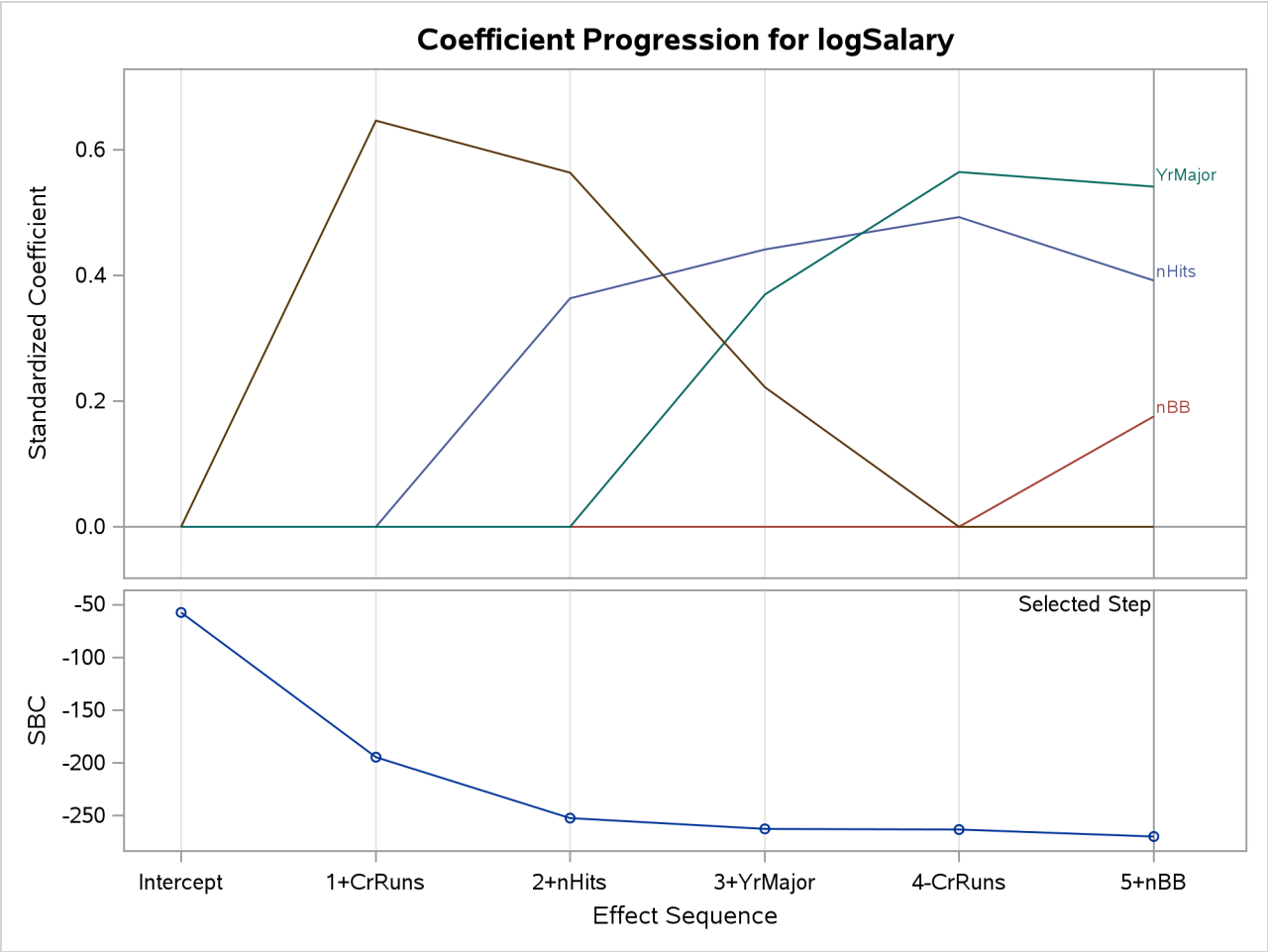

The selected effects, analysis of variance, fit statistics, and parameter estimates tables shown in Figure 56.11 give details of the selected model.

Figure 56.11 Details of the Selected Model

The GLMSELECT Procedure
Selected Model

The selected model is the model at the last step (Step 5).

Effects: Intercept nHits nBB YrMajor

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	3	120.52553	40.17518	120.12
Error	259	86.62820	0.33447	
Corrected Total	262	207.15373		

Figure 56.11 *continued*

Root MSE	0.57834
Dependent Mean	5.92722
R-Square	0.5818
Adj R-Sq	0.5770
AIC	-19.06903
AICC	-18.83557
BIC	-282.17004
C(p)	11.35235
PRESS	89.54336
SBC	-269.78041
ASE	0.32938

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	4.013911	0.111290	36.07
nHits	1	0.007929	0.000994	7.98
nBB	1	0.007280	0.002049	3.55
YrMajor	1	0.100663	0.007551	13.33

PROC GLMSELECT provides you with the flexibility to use several selection methods and many fit criteria for selecting effects that enter or leave the model. You can also specify criteria to determine when to stop the selection process and to choose among the models at each step of the selection process. You can find continued exploration of the baseball data that uses a variety of these methods in Example 56.1.

Syntax: GLMSELECT Procedure

The following statements are available in the GLMSELECT procedure:

```
PROC GLMSELECT <options> ;
   BY variables ;
   CLASS variable <(v-options)> <variable <(v-options ...)>> </ v-options> <options> ;
   CODE <options> ;
   EFFECT name = effect-type (variables </ options>) ;
   FREQ variable ;
   MODEL variable = <effects> </ options> ;
   MODELAVERAGE <options> ;
   OUTPUT <OUT=SAS-data-set> <keyword<=name>> <... keyword<=name>> ;
   PARTITION <options> ;
   PERFORMANCE <options> ;
   SCORE <DATA=SAS-data-set> <OUT=SAS-data-set> ;
   STORE <OUT=>item-store-name </ LABEL='label'> ;
   WEIGHT variable ;
```

All statements other than the MODEL statement are optional and multiple SCORE statements can be used. CLASS and EFFECT statements, if present, must precede the MODEL statement.

The STORE and CODE statements are also used by many other procedures. A summary description of functionality and syntax for these statements is also shown after the PROC GLMSELECT statement in alphabetical order, but you can find full documentation about them in the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

PROC GLMSELECT Statement

PROC GLMSELECT < *options* > **;**

The PROC GLMSELECT statement invokes the GLMSELECT procedure. Table 56.1 summarizes the *options* available in the PROC GLMSELECT statement.

Table 56.1 PROC GLMSELECT Statement Options

Option	Description
Data Set Options	
DATA=	Names a data set to use for the regression
MAXMACRO=	Sets the maximum number of macro variables produced
TESTDATA=	Names a data set that contains test data
VALDATA=	Names a data set that contains validation data
ODS Graphics Options	
PLOTS=	Produces ODS graphical displays
Other Options	
OUTDESIGN=	Requests a data set that contains the design matrix
NAMELEN=	Sets the length of effect names in tables and output data sets
NOPRINT	Suppresses displayed output including plots
SEED=	Sets the seed used for pseudo-random number generation

Following are explanations of the *options* that you can specify in the PROC GLMSELECT statement (in alphabetical order).

DATA=*SAS-data-set*

names the SAS data set to be used by PROC GLMSELECT. If the DATA= option is not specified, PROC GLMSELECT uses the most recently created SAS data set. If the named data set contains a variable named _ROLE_, then this variable is used to assign observations for training, validation, and testing roles. See the section "Using Validation and Test Data" on page 4478 for details on using the _ROLE_ variable.

MAXMACRO=*n*

specifies the maximum number of macro variables with selected effects to create. By default, MAXMACRO=100. PROC GLMSELECT saves the list of selected effects in a macro variable, &_GLSIND. Say your input effect list consists of x1-x10. Then &_GLSIND would be set to x1 x3 x4 x10 if, for example, the first, third, fourth, and tenth effects were selected for the model. This list can be used, for example, in the model statement of a subsequent procedure. If you specify the OUTDESIGN= option in the PROC GLMSELECT statement, then PROC GLMSELECT saves the list of columns in the design matrix in a macro variable named &_GLSMOD.

With BY processing, one macro variable is created for each BY group, and the macro variables are indexed by the BY group number. The MAXMACRO= option can be used to either limit or increase the number of these macro variables when you are processing data sets with many BY groups.

With no BY processing, PROC GLMSELECT creates the following:

_GLSIND	selected effects
_GLSIND1	selected effects
_GLSMOD	design matrix columns
_GLSMOD1	design matrix columns
_GLSNUMBYS	number of BY groups
_GLSNUMMACROBYS	number of _GLSIND*i* macro variables actually made

With BY processing, PROC GLMSELECT creates the following:

_GLSIND	selected effects for BY group 1
_GLSIND1	selected effects for BY group 1
_GLSIND2	selected effects for BY group 2
.	
.	
.	
_GLSIND*m*	selected effects for BY group *m*, where a number is substituted for *m*
_GLSMOD	design matrix columns for BY group 1
_GLSMOD1	design matrix columns for BY group 1
_GLSMOD2	design matrix columns for BY group 2
.	
.	
.	
_GLSMOD*m*	design matrix columns for BY group *m*, where a number is substituted for *m*
_GLSNUMBYS	*n*, the number of BY groups
_GLSNUMMACROBYS	the number *m* of _GLSIND*i* macro variables actually made. This value can be less than _GLSNUMBYS = *n*, and it is less than or equal to the MAXMACRO= value.

See the section "Macro Variables Containing Selected Models" on page 4472 for further details.

NOPRINT

suppresses all displayed output including plots.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters long, where *n* is a value between 20 and 200 characters. The default length is 20 characters.

OUTDESIGN <(*options***)><=***SAS-data-set***>**

creates a data set that contains the design matrix. By default, the GLMSELECT procedure includes in the OUTDESIGN data set the **X** matrix corresponding to the parameters in the selected model. Two schemes for naming the columns of the design matrix are available. In the first scheme, names of the parameters are constructed from the parameter labels that appear in the ParameterEstimates table. This

naming scheme is the default when you do not request BY processing and is not available when you do use BY processing. In the second scheme, the design matrix column names consist of a prefix followed by an index. The default naming prefix is _X.

You can specify the following *options* in parentheses to control the contents of the OUTDESIGN data set:

ADDINPUTVARS
requests that all variables in the input data set be included in the OUTDESIGN= data set.

FULLMODEL
specifies that parameters corresponding to all the effects specified in the MODEL statement be included in the OUTDESIGN= data set. By default, only parameters corresponding to the selected model are included.

NAMES
produces a table associating columns in the OUTDESIGN data set with the labels of the parameters they represent.

PREFIX< =prefix >
requests that the design matrix column names consist of a prefix followed by an index. The default naming prefix is _X. You can optionally specify a different prefix.

PARMLABELSTYLE=*options*
specifies how parameter names and labels are constructed for nested and crossed effects.

The following *options* are available:

INTERLACED < (SEPARATOR=*quoted-string***) >**
forms parameter names and labels by positioning levels of classification variables and constructed effects adjacent to the associated variable or constructed effect name and using " * " as the delimiter for both crossed and nested effects. This style of naming parameters and labels is used in the TRANSREG procedure. You can request truncation of the classification variable names used in forming the parameter names and labels by using the CPREFIX= and LPREFIX= options in the CLASS statement. You can use the SEPARATOR= suboption to change the delimiter between the crossed variables in the effect. PARMLABELSTYLE=INTERLACED is not supported if you specify the SPLIT option in an EFFECT statement or a CLASS statement. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender male * City Beijing
City London * Age
```

SEPARATE
specifies that in forming parameter names and labels, the effect name appears before the levels associated with the classification variables and constructed effects in the effect. You can control the length of the effect name by using the NAMELEN= option in the PROC GLMSELECT statement. In forming parameter labels, the first level that is displayed is positioned so that it starts at the same offset in every parameter label—this enables you to easily distinguish the effect name from the levels when the parameter labels are displayed in a column in the "Parameter

Estimates" table. This style of labeling is used in the GLM procedure and is the default if you do not specify the PARMLABELSTYLE option. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender*City male Beijing
Age*City    London
```

SEPARATECOMPACT

requests the same parameter naming and labeling scheme as PARMLABELSTYLE=SEPARATE except that the first level in the parameter label is separated from the effect name by a single blank. This style of labeling is used in the PLS procedure. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender*City male Beijing
Age*City London
```

PLOTS < (*global-plot-options*) > < = *plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < = (*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=all
plots=coefficients(unpack)
plots(unpack)=(criteria candidates)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc glmselect plots=all;
   model y = x1-x100;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

Global Plot Options

The *global-options* apply to all plots generated by the GLMSELECT procedure, unless it is altered by a *specific-plot-option*.

ENDSTEP=*n*
specifies that the step ranges shown on the horizontal axes of plots terminates at specified step. By default, the step range shown terminates at the final step of the selection process. If you specify the ENDSTEP= option as both a global plot option and a specific plot option, then the ENDSTEP= value on the specific plot is used.

LOGP | LOGPVALUE
displays the natural logarithm of the entry and removal significance levels when the SELECT=SL option is specified.

MAXSTEPLABEL=*n*
specifies the maximum number of characters beyond which labels of effects on plots are truncated.

MAXPARMLABEL= *n*
specifies the maximum number of characters beyond which parameter labels on plots are truncated.

STARTSTEP=*n*
specifies that the step ranges shown on the horizontal axes of plots start at the specified step. By default, the step range shown starts at the initial step of the selection process. If you specify the STARTSTEP= option both as a global plot option and a specific plot option, then the STARTSTEP= value on the specific plot is used.

STEPAXIS=EFFECT | NORMB | NUMBER
specifies the horizontal axis to be used on the plots, where this axis represents the sequence of entering or departing effects.

STEPAXIS=EFFECT
requests that each step be labeled by a prefix followed by the name of the effect that enters or leaves at that step. The prefix consists of the step number, followed by a "+" sign or a "-" sign, depending on whether the effect enters (+) or leaves (-) at that step.

STEPAXIS=NORMB
is valid only with the LAR, LASSO, and elastic net selection methods and requests that the horizontal axis value at step i be the L1 norm of the parameters at step i, normalized by the L1 norm of the parameters at the final step.

STEPAXIS=NUMBER
requests that each step be labeled by the step number.

UNPACK
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACK to see each plot individually. You can also specify UNPACK as a suboption of the CRITERIA and COEFFICIENTS options.

Specific Plot Options

The following listing describes the specific plots and their *options*.

ALL

requests that all default plots be produced. Note that candidate plots are produced only if you specify DETAILS=STEPS or DETAILS=ALL in the MODEL statement.

ASE | ASEPLOT < (*aseplot-option*) >

plots the progression of the average square error on the training data, and the test and validation data whenever these data are provided with the TESTDATA= and VALDATA= options or are produced by using a PARTITION statement. You can specify the following *aseplot-option*:

STEPAXIS=EFFECT | NORMB | NUMBER

specifies the horizontal axis to be used.

CANDIDATES | CANDIDATESPLOT < (*candidatesplot-options*) >

produces a needle plot of the SELECT= criterion values for the candidates for entry or removal at each step of the selection process, ordered from best to worst. Candidates plots are not available if you specify SELECTION=NONE, SELECTION=LAR, SELECTION=LASSO, SELECTION=ELASTICNET, or SELECTION=GROUPLASSO in the MODEL statement, or if you have not specified DETAILS=ALL or DETAILS=STEPS in the MODEL statement. The following *candidatesplot-options* are available:

LOGP | LOGPVALUE

displays the natural logarithm of the entry and removal significance levels when the SELECT=SL option is specified.

SHOW=*number*

specifies the maximum number of candidates displayed at each step. The default is SHOW=10.

COEFFICIENTS | COEFFICIENTPANEL < (*coefficientPanel-options*) >

plots a panel of two plots. The upper plot shows the progression of the parameter values as the selection process proceeds. The lower plot shows the progression of the CHOOSE= criterion. If no CHOOSE= criterion is in effect, then the AICC criterion is displayed. The following *coefficientPanel-options* are available:

LABELGAP=*percentage*

specifies the percentage of the vertical axis range that forms the minimum gap between successive parameter labels at the final step of the coefficient progression plot. If the values of more than one parameter at the final step are closer than this gap, then the labels on all but one of these parameters is suppressed. The default value is LABELGAP=5.

LOGP | LOGPVALUE

displays the natural logarithm of the entry and removal significance levels when the SELECT=SL option is specified.

STEPAXIS=EFFECT | NORMB | NUMBER

specifies the horizontal axis to be used.

UNPACK | UNPACKPANEL
displays the coefficient progression and the CHOOSE= criterion progression in separate plots.

CRITERIA | CRITERIONPANEL < (*criterionPanel-options*) >
plots a panel of model fit criteria. The criteria that are displayed are ADJRSQ, AIC, AICC, and SBC, as well as any other criteria that are named in the CHOOSE=, SELECT=, STOP=, or STATS= option in the MODEL statement. The following *criterionPanel-options* are available:

STEPAXIS=EFFECT | NORMB | NUMBER
specifies the horizontal axis to be used.

UNPACK | UNPACKPANEL
displays each criterion progression on a separate plot.

EFFECTSELECTPCT < (*effectSelectPct-options*) >
requests a bar chart whose bars correspond to effects that are selected in at least one sample when you use the MODELAVERAGE statement. The length of a bar corresponds to the percentage of samples where the selected model contains the effect the bar represents. The EFFECTSELECTPCT option is ignored if you do not specify a MODELAVERAGE statement. The following *effectSelectPct-options* are available:

MINPCT=*percent*
specifies that effects that appear in fewer than the specified percentage of the sample selected models not be included in the plot. By default, effects that are shown in the EffectSelectPct table are displayed.

ORDER=ASCENDING | DESCENDING | MODEL
specifies the ordering of the effects in the bar chart. ORDER=MODEL specifies that effects appear in the order in which they appear in the MODEL statement. ORDER=ASCENDING | DESCENDING specifies that the effects are shown in ascending or descending order of the number of samples in which the effects appear in the selected model. The default is ORDER=DESCENDING.

NONE
suppresses all plots.

PARMDIST < (*parmDist-options*) >
produces a panel that shows histograms and box plots of the parameter estimate values across samples when you use a MODELAVERAGE statement. There is a histogram and box plot for each parameter that appears in the AvgParmEst table. The PARMDIST option is ignored if you do not specify a MODELAVERAGE statement. The following *parmDist-options* are available:

MINPCT=*percent*
specifies that distributions be shown only for parameters whose estimates are nonzero in at least the specified percentage of the selected models. By default, distributions are shown for the all parameters that appear in the AvgParmEst table.

ORDER=ASCENDING | DESCENDING | MODEL
specifies the ordering of the parameters in the panels. ORDER=MODEL specifies that parameters be shown in the order in which the corresponding effects appear in the MODEL statement. ORDER=ASCENDING | DESCENDING specifies that the parameters be shown in an ascending or descending order of the number of samples in which the parameter estimate is nonzero. The default is ORDER=DESCENDING.

NOBOXPLOTS
suppress the box plots.

PLOTSPERPANEL=*number*
specifies the maximum number of parameter distributions that appear in a panel. If the number of relevant parameters is greater than *number*, then multiple panels are produced. Valid values are 1–16 with 9 as the default.

UNPACK
specifies that the distribution for each relevant parameter be shown in a separate plot.

SEED=*number*
specifies an integer used to start the pseudo-random number generator for resampling the data, random cross validation, and random partitioning of data for training, testing, and validation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer's clock.

TESTDATA=*SAS-data-set*
names a SAS data set containing test data. This data set must contain all the variables specified in the MODEL statement. Furthermore, when a BY statement is used and the TESTDATA=data set contains any of the BY variables, then the TESTDATA= data set must also contain all the BY variables sorted in the order of the BY variables. In this case, only the test data for a specific BY group is used with the corresponding BY group in the analysis data. If the TESTDATA= data set contains none of the BY variables, then the entire TESTDATA = data set is used with each BY group of the analysis data.

If you specify a TESTDATA=data set, then you cannot also reserve observations for testing by using a PARTITION statement.

VALDATA=*SAS-data-set*
names a SAS data set containing validation data. This data set must contain all the variables specified in the MODEL statement. Furthermore, when a BY statement is used and the VALDATA=data set contains any of the BY variables, then the VALDATA= data set must also contain all the BY variables sorted in the order of the BY variables. In this case, only the validation data for a specific BY group are used with the corresponding BY group in the analysis data. If the VALDATA= data set contains none of the BY variables, then the entire VALDATA = data set is used with each BY group of the analysis data.

If you specify a VALDATA=data set, then you cannot also reserve observations for validation by using a PARTITION statement.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC GLMSELECT to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the GLMSELECT procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(***v-options***)** > . . . < *variable* < **(***v-options***)** > > < / *options* > **;**

The CLASS statement names the classification variables to be used in the analysis. The CLASS statement must precede the MODEL statement.

Table 56.2 summarizes the *options* available in the CLASS statement.

Table 56.2 CLASS Statement Options

Option	Description
CPREFIX=	Specifies maximum number of CLASS variable name characters
DELIMITER=	Specifies the delimiter
DESCENDING	Reverses the sort order
LPREFIX=	Labels design variables
MISSING	Allows for missing values
ORDER=	Specifies the sort order
PARAM=	Specifies the parameterization method
REF=	Specifies the reference level
SHOW	Requests a table for each CLASS variable
SPLIT	Splits CLASS variables into independent effects

The following *options* can be specified after a slash (/):

DELIMITER=*quoted character*
specifies the delimiter that is used between levels of classification variables in building parameter names and lists of class level values. The default if you do not specify DELIMITER= is a space. This option is useful if the levels of a classification variable contain embedded blanks.

SHOW | SHOWCODING
requests a table for each classification variable that shows the coding used for that variable.

You can specify various *v-options* for each variable by enclosing them in parentheses after the variable name. You can also specify global *v-options* for the CLASS statement by placing them after a slash (/). Global *v-options* are applied to all the variables specified in the CLASS statement. If you specify more than one CLASS statement, the global *v-options* specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *v-options* override the global *v-options* except for the PARAM=GLM option. The global PARAM=GLM option overrides all individual PARAM= options.

The following *v-options* are available:

CPREFIX=*n*
specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable. The CPREFIX= applies only when you specify the PARMLABELSTYLE=INTERLACED option in the PROC GLMSELECT statement.

DESCENDING
DESC
reverses the sort order of the classification variable.

LPREFIX=*n*
specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable. The LPREFIX= applies only when you specify the PARMLABELSTYLE=INTERLACED option in the PROC GLMSELECT statement.

MISSING
allows missing value ('.' for a numeric variable and blanks for a character variables) as a valid value for the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so the ORDER= option might be useful when you use the CONTRAST or ESTIMATE statement. If ORDER=FORMATTED for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC GLMSELECT interprets values of the ORDER= option.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For FORMATTED and INTERNAL, the sort order is machine dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. Design matrix columns are created from CLASS variables according to the following coding schemes. If the PARAM= option is not specified with any individual CLASS variable, by default, PARAM=GLM. Otherwise, the default is PARAM=EFFECT. If PARAM=ORTHPOLY or PARAM=POLY, and the CLASS levels are numeric, then the ORDER= option in the CLASS statement is ignored, and the internal, unformatted values are used. See the section "CLASS Variable Parameterization and the SPLIT Option" on page 4471 for further details.

EFFECT specifies effect coding.

GLM specifies less-than-full-rank, reference-cell coding; this option can be used only as a global option.

ORDINAL
THERMOMETER specifies the cumulative parameterization for an ordinal CLASS variable.

POLYNOMIAL
POLY specifies polynomial coding.

REFERENCE
REF specifies reference-cell coding.

ORTHEFFECT orthogonalizes PARAM=EFFECT.

ORTHORDINAL
ORTHOTHERM orthogonalizes PARAM=ORDINAL.

ORTHPOLY orthogonalizes PARAM=POLYNOMIAL.

ORTHREF orthogonalizes PARAM=REFERENCE.

The EFFECT, POLYNOMIAL, REFERENCE, and ORDINAL schemes and their orthogonal parameterizations are full rank. The REF= option in the CLASS statement determines the reference level for the EFFECT and REFERENCE schemes and their orthogonal parameterizations.

REF=*'level'* | *keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For an individual (but not a global) variable REF= option, you can specify the *level* of the variable to use as the reference level. For a global or individual variable REF= option, you can use one of the following *keywords*. The default is REF=LAST.

FIRST designates the first-ordered level as reference.

LAST designates the last-ordered level as reference.

SPLIT

splits the columns of the design matrix that correspond to any effect that contains a split classification variable so that they can enter or leave a model independently of the other design columns for that effect. For example, suppose a variable named temp has three levels with values `'hot'`, `'warm'`, and `'cold'`, and a variable named sex has two levels with values `'M'` and `'F'` are used in a PROC GLMSELECT job as follows:

```
proc glmselect;
   class temp sex/split;
   model depVar = sex sex*temp;
run;
```

The two effects named in the MODEL statement are split into eight independent effects. The effect `'sex'` is split into two effects labeled `'sex_M'` and `'sex_F'`. The effect `'sex*temp'` is split into six effects labeled `'sex_M*temp_hot'`, `'sex_F*temp_hot'`, `'sex_M*temp_warm'`, `'sex_F*temp_warm'`, `'sex_M*temp_cold'`, and `'sex_F*temp_cold'`. Thus the previous PROC GLMSELECT step is equivalent to the following step:

```
proc glmselect;
   model depVar =  sex_M sex_F sex_M*temp_hot  sex_F*temp_hot
                               sex_M*temp_warm sex_F*temp_warm
                               sex_M*temp_cold sex_F*temp_cold;
run;
```

The split option can be used on individual classification variables. For example, consider the following PROC GLMSELECT step:

```
proc glmselect;
   class temp(split) sex;
   model depVar = sex sex*temp;
run;
```

In this case the effect `'sex'` is not split and the effect `'sex*temp'` is split into three effects labeled `'sex*temp_hot'`, `'sex*temp_warm'`, and `'sex*temp_cold'`. Furthermore each of these three split effects now has two parameters that correspond to the two levels of `'sex'`, and the PROC GLMSELECT step is equivalent to the following:

```
proc glmselect;
   class sex;
   model depVar = sex sex*temp_hot sex*temp_warm sex*temp_cold;
run;
```

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 56.3 summarizes the *options* available in the CODE statement.

Table 56.3 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) **;**

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 56.4 summarizes the *options* available in the EFFECT statement.

Table 56.4 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

The variable that is specified in the FREQ statement identifies a variable in the input data set that contains the frequency of occurrence of each observation. PROC GLMSELECT treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If it is not an integer, the frequency value is truncated to an integer. If it is less than 1 or missing, the observation is not used.

MODEL Statement

MODEL *dependent* **=** < *effects* > / < *options* > **;**

The MODEL statement names the dependent variable and the explanatory effects, including covariates, main effects, constructed effects, interactions, and nested effects; for more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure." If you omit the explanatory effects, the procedure fits an intercept-only model.

After the keyword MODEL, the dependent (response) variable is specified, followed by an equal sign. The explanatory effects follow the equal sign.

Table 56.5 summarizes the *options* available in the MODEL statement.

Table 56.5 MODEL Statement Options

Option	Description
CVDETAILS<=>	Specifies the details to be displayed when you use cross validation
CVMETHOD=	Specifies how subsets for cross validation are formed
DETAILS=	Specifies the details to be displayed
FUZZ=	Specifies the tolerance range for criterion comparisons
HIERARCHY=	Specifies the hierarchy of effects to impose
NOINT	Specifies models without an explicit intercept
ORDERSELECT	Displays parameter estimates in the order in which the parameters first entered the model
SELECTION=	Specifies the model selection method
SHOWPVALUES	Displays *p*-values in "ANOVA" and "Parameter Estimates" tables
STATS=	Specifies additional statistics to be displayed
STB<=>	Adds standardized coefficients to "Parameter Estimates" tables

You can specify the following *options* in the MODEL statement after a slash (/).

CVDETAILS < =ALL | COEFFS | CVPRESS >

specifies the details that are produced when cross validation is requested as the CHOOSE=, SELECT=, or STOP= criterion in the MODEL statement. If n-fold cross validation is being used, then the training data are subdivided into n parts, and at each step of the selection process, models are obtained on each of the n subsets of the data obtained by omitting one of these parts. CVDETAILS=COEFFS requests that the parameter estimates obtained for each of these n subsets be included in the parameter estimates table. CVDETAILS=CVPRESS requests a table containing the predicted residual sum of squares of each of these models scored on the omitted subset. CVDETAILS=ALL requests both CVDETAILS=COEFFS and CVDETAILS=CVPRESS. If DETAILS=STEPS or DETAILS=ALL has been specified in the MODEL statement, then the specified cross validation details are produced for every step of the selection process. If you specify the CVDETAILS option without a value, then by default CVDETAILS=CVPRESS.

CVMETHOD=BLOCK< (*n*) > | RANDOM< (*n*) > | SPLIT< (*n*) > | INDEX (*variable*)

specifies how the training data are subdivided into n parts when you request n-fold cross validation by using any of the CHOOSE=CV, SELECT=CV, and STOP=CV suboptions of the SELECTION= option in the MODEL statement.

- **BLOCK** requests that parts be formed of n blocks of consecutive training observations.
- **SPLIT** requests that the ith part consist of training observations $i, i+n, i+2n, \ldots$.
- **RANDOM** assigns each training observation randomly to one of the n parts.
- **INDEX(*variable*)** assigns observations to parts based on the formatted value of the named variable. This input data set variable is treated as a classification variable and the number of parts n is the number of distinct levels of this variable. By optionally naming this variable in a CLASS statement, you can use the CLASS statement options ORDER= and MISSING to control how the levelization of this variable is done.

The value n defaults to 5 when CVMETHOD=BLOCK, SPLIT, or RANDOM. If you omit the CVMETHOD= option, then by default CVMETHOD=RANDOM(5).

DETAILS=*level* | STEPS < (*step-options*) >

specifies the level of detail to be produced, where *level* can be ALL, STEPS, or SUMMARY. By default, DETAILS=SUMMARY.

The DETAILS=ALL option produces the following:

- entry and removal statistics for each variable selected in the model building process
- ANOVA, fit statistics, and parameter estimates
- entry and removal statistics for the top 10 candidates for inclusion or exclusion at each step
- a selection summary table

The DETAILS=SUMMARY option produces only the selection summary table.

The option DETAILS=STEPS < (*step options*) > provides the step information and the selection summary table. The following *options* can be specified within parentheses after the DETAILS=STEPS option:

ALL
requests ANOVA, fit statistics, parameter estimates, and entry or removal statistics for the top 10 candidates for inclusion or exclusion at each selection step.

ANOVA
requests ANOVA at each selection step.

FITSTATISTICS | FITSTATS | FIT
requests fit statistics at each selection step. The default set of statistics includes all the statistics named in the CHOOSE=, SELECT=, and STOP= suboptions specified in the MODEL statement SELECTION= option, but you can request additional statistics by specifying the STATS= option in the MODEL statement.

PARAMETERESTIMATES | PARMEST
requests parameter estimates at each selection step.

CANDIDATES < (SHOW= ALL | *n*) >
requests entry or removal statistics for the best *n* candidate effects for inclusion or exclusion at each step. If you specify SHOW=ALL, then all candidates are shown. If SHOW= is not specified. then the best 10 candidates are shown. The entry or removal statistic is the statistic named in the SELECT= option that is specified in the SELECTION= option in the MODEL statement.

FUZZ=*value*
specifies the tolerance range for criterion comparisons. Criterion values that differ by less than the tolerance are regarded as equal. If you specify FUZZ=0, then the comparisons are based on simple equality. The default is 10^{-12}.

HIERARCHY=NONE | SINGLE | SINGLECLASS
HIER=NONE | SINGLE | SINGLECLASS
specifies whether and how the model hierarchy requirement is applied. This option also controls whether a single effect or multiple effects are allowed to enter or leave the model in one step. You can specify that only classification effects, or both classification and continuous effects, be subject to the hierarchy requirement. The HIERARCHY= option is ignored unless you also specify one of the following *options*: SELECTION=FORWARD, SELECTION=BACKWARD, or SELECTION=STEPWISE.

Model hierarchy refers to the requirement that for any term to be in the model, all model effects contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model.

You can specify the following values:

NONE
specifies that model hierarchy not be maintained. Any single effect can enter or leave the model at any given step of the selection process.

SINGLE
specifies that only one effect enter or leave the model at one time, subject to the model hierarchy requirement. For example, suppose that the model contains the main effects A and B and the interaction A*B. In the first step of the selection process, either A or B can enter the model. In the second step, the other main effect can enter the model. The interaction effect can enter the model only when both main effects have already entered. Also, before A or B can be removed from the model, the A*B interaction must first be removed. All effects (CLASS and interval) are subject to the hierarchy requirement.

SINGLECLASS
is the same as HIERARCHY=SINGLE except that only CLASS effects are subject to the hierarchy requirement.

By default, HIERARCHY=NONE.

NOINT
suppresses the intercept term that is otherwise included in the model.

ORDERSELECT
specifies that for the selected model, effects be displayed in the order in which they first entered the model. If you do not specify the ORDERSELECT option, then effects in the selected model are displayed in the order in which they appeared in the MODEL statement.

SELECTION=*method* **<(***method-options***)>**
specifies the method used to select the model, optionally followed by parentheses enclosing options applicable to the specified method. The default if the SELECTION= option is omitted is SELECTION=STEPWISE.

You can specify the following *methods*, which are explained in detail in the section "Model-Selection Methods" on page 4458:

NONE specifies no model selection.

FORWARD specifies forward selection. This method starts with no effects in the model and adds effects.

BACKWARD specifies backward elimination. This method starts with all effects in the model and deletes effects.

STEPWISE specifies stepwise regression. This is similar to the forward selection method except that effects already in the model do not necessarily stay there.

LAR specifies least angle regression. This method, like forward selection, starts with no effects in the model and adds effects. The parameter estimates at any step are "shrunk" when compared to the corresponding least squares estimates. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.

LASSO specifies the LASSO method, which adds and deletes parameters based on a version of ordinary least squares where the sum of the absolute regression coefficients is constrained. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.

ELASTICNET specifies the elastic net method, an extension of LASSO that estimates parameters based on a version of ordinary least squares in which both the sum of the absolute regression coefficients and the sum of the squared regression coefficients are constrained. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.

GROUPLASSO specifies the group LASSO method, a variant of LASSO that estimates parameters based on a version of ordinary least squares in which the sum of the Euclidean norms of a group of regression coefficients is constrained.

Table 56.6 lists the applicable *method-options* for each of these methods.

Table 56.6 Applicable SELECTION= Options by Method

Option	FORWARD	BACKWARD	STEPWISE	LAR	LASSO	ELASTICNET	GROUPLASSO
STOP=	x	x	x	x	x	x	x
CHOOSE=	x	x	x	x	x	x	x
STEPS=	x	x	x	x	x	x	x
MAXSTEP=	x	x	x	x	x	x	x
SELECT=	x	x	x				
INCLUDE=	x	x	x				
SLENTRY=	x		x	x	x	x	
SLSTAY=		x	x		x	x	
DROP=			x				
ADAPTIVE					x		
LSCOEFFS				x	x		
L1=					x	x	x
L1CHOICE=					x	x	
L2=						x	
L2STEPS=						x	
L2LOW=						x	
L2HIGH=						x	
L2SEARCH=						x	
ENSCALE						x	
RHO=							x
SCREEN=					x	x	
SHOWL2SEARCH						x	
SHOWSTEPL1					x		x
TOL=							x

The syntax of the *method-options* follows. Note that, as described in Table 56.6, not all selection *method-options* are available for every SELECTION= method.

ADAPTIVE < (< **GAMMA=***nonnegative number* > < **INEST=***SAS-data-set* >) >

requests that adaptive weights be applied to each of the coefficients in the LASSO method. You use the optional INEST= option to name the SAS data set that contains estimates that are used to form the adaptive weights for all the parameters in the model. If you do not specify an INEST= data set, then ordinary least squares estimates of the parameters in the model are used in forming the adaptive weights. You use the GAMMA= option to specify the power transformation that is applied to the parameters in forming the adaptive weights. By default, GAMMA=1.

CHOOSE=*criterion*

specifies the criterion for choosing the model. The specified *criterion* is evaluated at each step of the selection process, and the model that yields the best value of the criterion is chosen. If the optimal value of the criterion occurs for models at more than one step, then the model that has the smallest number of parameters is chosen. If you do not specify the CHOOSE= option, then the model at the final step in the selection process is selected.

The criteria that you can specify in the CHOOSE= option are shown in Table 56.7. For more information about these criteria, see the section "Criteria Used in Model Selection Methods" on page 4469.

Table 56.7 Criteria for the CHOOSE= Option

Criterion	**Criterion**
ADJRSQ	Adjusted R-square statistic
AIC	Akaike's information criterion
AICC	Corrected Akaike's information criterion
BIC	Sawa Bayesian information criterion
CP	Mallows' C_p statistic
CV	Predicted residual sum of square with k-fold cross validation
CVEX	Predicted residual sum of square with k-fold external cross validation
PRESS	Predicted residual sum of squares
SBC	Schwarz Bayesian information criterion
VALIDATE	Average square error for the validation data

For ADJRSQ, the chosen value is the largest one; for all other criteria, the smallest value is chosen. You can use the CHOOSE=VALIDATE option only if you have specified a VALDATA= data set in the PROC GLMSELECT statement or if you have reserved part of the input data for validation by using either a PARTITION statement or a _ROLE_ variable in the input data. The PRESS criterion is not available for SELECTION=ELASTICNET. The CHOOSE=PRESS option cannot be used with SELECTION=LAR or SELECTION=LASSO unless the LSCOEFFS suboption is also specified. The BIC, CP, CV, and PRESS criteria are not available for SELECTION=GROUPLASSO.

DROP=BEFOREADD | COMPETITIVE

specifies when effects are eligible to be dropped in the STEPWISE method. You can specify the following values:

BEFOREADD requests that currently in the model be examined to see if any meet the requirements to be removed from the model. If so, the effect that gives the best value of the removal criterion is dropped from the model and the stepwise method proceeds to the next step. Only when no effect currently in the model meets the requirement to be removed from the model are any effects added to the model.

COMPETITIVE requests that the SELECT= criterion be evaluated for all models in which an effect currently in the model is dropped or an effect not yet in the model is added. The effect whose removal or addition to the model yields the maximum improvement to the SELECT= criterion is dropped or added. You can specify DROP=COMPETITIVE only if the SELECT= criterion is not SL.

By default, DROP=BEFOREADD. If SELECT=SL, PROC GLMSELECT uses the traditional stepwise method as implemented in PROC REG.

ENSCALE

requests that the solution to SELECTION=ELASTICNET be scaled to offset bias because of the double shrinkage inherent in the elastic net method (Zou and Hastie 2005). This option applies only when SELECTION=ELASTICNET. The default is not to rescale the solution: this is the so-called naive elastic net.

INCLUDE=*n*

forces the first *n* effects listed in the MODEL statement to be included in all models. The selection methods are performed on the other effects in the MODEL statement. The INCLUDE= option is available only when SELECTION=FORWARD, SELECTION=STEPWISE, and SELECTION=BACKWARD.

L1=*value*

specifies the LASSO regularization or constraint parameter that is used when SELECTION=LASSO, SELECTION=ELASTICNET or SELECTION=GROUPLASSO. This option is available only when you specify the STOP=L1 option with SELECTION=LASSO, SELECTION=ELASTICNET, or SELECTION=GROUPLASSO.

L1CHOICE=NORM | RATIO | VALUE

specifies both the criterion used in the L1=*value* option and the criterion used in aggregating the results of *k*-fold external cross validation for computing the CVEXPRESS statistic. This option is available only when you specify SELECTION=LASSO or SELECTION=ELASTICNET. You can specify the following values:

NORM indicates that the value specified in the L1=*value* option corresponds to the sum of the absolute values of the coefficients (the so-called L1 norm), and the *k*-fold external cross validation aggregation is based on the L1 norms.

RATIO indicates that the value specified in the L1=*value* option corresponds to the ratio obtained by scaling the value of the LASSO regularization parameter to lie in the interval [0,1], and the *k*-fold external cross validation aggregation is based on the scaled ratios.

VALUE indicates that the value specified in the L1=*value* option corresponds to the actual value of the LASSO regularization parameter, and the *k*-fold external cross validation aggregation is based on the actual LASSO regularization parameters.

By default, L1CHOICE=RATIO. If you specify SELECTION=GROUPLASSO, L1CHOICE=RATIO is used.

L2=*value*

specifies the ridge regularization parameter that is used when SELECTION=ELASTICNET. The L2= option is available only when SELECTION=ELASTICNET. If you specify the L2= option, then the value that you specify is used in defining the elastic net method, and the L2HIGH=, L2LOW=, L2SEARCH=, and L2STEPS= options are ignored. If you do not specify the L2 = option with SELECTION=ELASTICNET, then PROC GLMSELECT searches for the suitable value of L2 according to the L2HIGH=, L2LOW=, L2SEARCH=, and L2STEPS= options.

L2HIGH=*value*

specifies the highest value used in the search of the ridge regression parameter L2 when SELECTION=ELASTICNET. If you specify the L2= option, then the L2HIGH= option is ignored. By default, L2HIGH=1.

L2LOW=*value*

specifies the lowest value used in the search of the ridge regression parameter L2 when SELECTION=ELASTICNET. If you specify the L2= option, then the L2LOW= option is ignored. By default, L2LOW=0.

L2SEARCH=GOLDEN | GRID

specifies the approach for the search of the ridge regression parameter L2 for the SELECTION=ELASTICNET option. You can specify the following values:

GOLDEN requests a golden section search of L2 in the range [low, high], where low and high are specified by the L2LOW= and L2HIGH= options, respectively.

GRID requests a log scale grid search of L2 in the range [low, high], where low and high are specified by the L2LOW= and L2HIGH= options, respectively. If L2LOW=0, then the log scale grid search for L2 is in the range $[10^{-8}, \text{high}]$ plus 0.

If you specify the L2= option with SELECTION=ELASTICNET, then the L2SEARCH= option is ignored. By default, L2SEARCH=GRID.

L2STEPS=*n*

specifies the number of steps in the search of the ridge regression parameter L2 when SELECTION=ELASTICNET. If you specify the L2 = option, then the L2STEPS= option is ignored. By default, L2STEPS=50.

LSCOEFFS

requests a hybrid version of the LAR or LASSO method, in which the sequence of models is determined by the LAR or LASSO method but the coefficients of the parameters for the model at any step are determined by using ordinary least squares.

MAXSTEP=*n*

specifies the maximum number of selection steps that are performed. The default value of *n* is the number of effects in the model statement for the forward, backward, and LAR methods, two times the number of effects for the stepwise, LASSO, and elastic net methods and 100 for the group LASSO method.

SELECT=*criterion*

specifies the criterion that PROC GLMSELECT uses to determine the order in which effects enter or leave at each step of the specified selection method. The SELECT= option is not valid with the LAR, LASSO, elastic net, and group LASSO methods. The criteria that you can specify with the SELECT= option are ADJRSQ, AIC, AICC, BIC, CP, CV, PRESS, RSQUARE, SBC, SL, and VALIDATE. For more information about these criteria, see the section "Criteria Used in Model Selection Methods" on page 4469. The default value of the SELECT= criterion is SELECT=SBC. You can use SELECT=SL to request the traditional approach, in which effects enter and leave the model based on the significance level. For other SELECT= criteria, the effect that is selected to enter or leave at a step of the selection process is the effect whose addition to or removal from the current model gives the maximum improvement in the specified criterion.

RHO=*value*

specifies the value of ρ that determines the sequence of regularization parameters $[\rho, \rho^2, \rho^3, \rho^4, \ldots]$ used in SELECTION=GROUPLASSO. By default, RHO=0.9.

SCREEN=NONE | SASVI | SIS< (*sis-options***) >**

specifies which screening method to apply. This option is ignored unless you also specify SELECTION=LASSO or SELECTION=ELASTICNET. In addition, the SCREEN= option is ignored when the SSCP matrix is computed in full. For more information about the SSCP matrix, see "Building the SSCP Matrix" on page 4475.

You can specify the following values:

NONE

specifies no screening.

SASVI

uses the SASVI safe screening technique of Liu et al. (2014). The resulting solution is identical to the one that results from specifying SCREEN=NONE, but it can often be several times faster to obtain. For more information about the SASVI technique, see the section "Safe Screening via SCREEN=SASVI" on page 4485.

SIS< (*sis-options***) >**

uses the sure independence screening (SIS) technique of Fan and Lv (2008). Because SIS is a heuristic screening rule, the resulting solution is not necessarily identical to the one that results from specifying SCREEN=NONE. However, it can be much faster to obtain when the number of potential predictors is very large.

You can specify the following *sis-options*:

KEEPNUM=*n*

keeps *n* effects that have the largest screening statistic values. The kept effects are then processed by the method that is specified in the SELECTION= option for model selection.

KEEPRATIO=*p*

specifies a value in the range 0 to 1 for *p*, and keeps (*p**100)% effects that have the largest screening statistic values. The kept effects are then processed by the method that is specified in the SELECTION= option for model selection.

If you do not specify any *sis-options*, KEEPNUM=100 by default. If you specify both KEEPNUM=*n* and KEEPRATIO=*value*, KEEPRATIO=*value* is used.

By default, SCREEN=NONE, which means that no screening is done. In this case, the LASSO or elastic net method is performed on all the potential predictors.

SHOWL2SEARCH

displays an elastic net parameter summary table, which shows the chosen values for the LASSO regularization parameter L1 and the ridge regularization parameter L2 (in the ratio scaling). This option is available only when you specify SELECTION=ELASTICNET. You can provide the corresponding L1 and L2 values back to PROC GLMSELECT by specifying SELECTION=ELASTICNET(STOP=L1 L1CHOICE=RATIO L1=*L1 value* L2=*L2 value*).

SHOWSTEPL1

adds to the selection summary table a column that shows the value of the LASSO regularization parameter L1 for each step of the LASSO selection process, with the selected step identified. This option is available only when you specify SELECTION=LASSO or SELECTION=GROUPLASSO. You can provide the corresponding L1 value back to PROC GLMSELECT by specifying SELECTION=LASSO(STOP=L1 L1=*value*) or SELECTION=GROUPLASSO(STOP=L1 L1=*value*).

SLENTRY=*value*

SLE=*value*

specifies the significance level for entry, which is used when the STOP=SL or SELECT=SL option is specified. The default is 0.50 when SELECTION=FORWARD, SELECTION=LAR, SELECTION=LASSO or SELECTION=ELASTICNET. The default is 0.15 when SELECTION=STEPWISE.

SLSTAY=*value*

SLS=*value*

specifies the significance level for staying in the model, which is used when the STOP=SL or SELECT=SL option is specified. The default is 0.10 when SELECTION=BACKWARD, SELECTION=LAR, SELECTION=LASSO or SELECTION=ELASTICNET. The default is 0.15 when SELECTION=STEPWISE.

STEPS=*n*

specifies the number of selection steps to be done. If the STEPS= option is specified, the STOP= and MAXSTEP= options are ignored.

STOP=*n*

STOP=*criterion*

specifies when PROC GLMSELECT is to stop the selection process. If the STEPS= option is specified, then the STOP= option is ignored. If the STOP=option does not cause the selection process to stop before the maximum number of steps for the selection method, then the selection process terminates at the maximum number of steps.

If you do not specify the STOP= option but do specify the SELECT= option, then the criterion named in the SELECT=option is also used as the STOP= criterion. If you do not specify either the STOP= or SELECT= option, then by default STOP=SBC.

If you specify STOP=*n*, then PROC GLMSELECT stops selection at the first step for which the selected model has *n* effects.

The nonnumeric arguments that you can specify in the STOP= option are shown in Table 56.8. For more information about these criteria, see the section "Criteria Used in Model Selection Methods" on page 4469.

Table 56.8 Nonnumeric Criteria for the STOP= Option

Option	Criteria
NONE	
ADJRSQ	Adjusted R-square statistic
AIC	Akaike's information criterion
AICC	Corrected Akaike's information criterion
BIC	Sawa Bayesian information criterion
CP	Mallows' C_p statistic
CV	Predicted residual sum of square with k-fold cross validation
L1	The LASSO regularization or constraint parameter
PRESS	Predicted residual sum of squares
SBC	Schwarz Bayesian information criterion
SL	Significance level
VALIDATE	Average square error for the validation data

When you use the SL criterion, selection stops at the step where the significance level for entry of all the effects not yet in the model is greater than the SLE= value for addition steps in the forward and stepwise methods and where the significance level for removal of any effect in the current model is smaller than the SLS= value in the backward and stepwise methods. When you use the ADJRSQ criterion, selection stops at the step where the next step would yield a model that has a smaller value of the adjusted R-square statistic; for all other criteria, selection stops at the step where the next step would yield a model that has a larger value of the criteria. You can use the VALIDATE option only if you have specified a VALDATA= data set in the PROC GLMSELECT statement or if you have reserved part of the input data for validation by using either a PARTITION statement or a _ROLE_ variable in the input data.

The L1 criterion is available only when SELECTION=LASSO or SELECTION=ELASTICNET. When you use the L1 criterion, selection stops at the step where the LASSO regularization parameter is equal to the value specified by the L1=*value* option. The PRESS criterion is not available for SELECTION=ELASTICNET. The STOP=PRESS option cannot be used with SELECTION=LAR or SELECTION=LASSO unless the LSCOEFFS suboption is also specified. The BIC, CP, CV, PRESS, and SL criteria are not available for SELECTION=GROUPLASSO.

STAT | STATS=*name*

STATS=(*names***)**

specifies which model fit statistics are displayed in the fit summary table and fit statistics tables. If you omit the STATS= option, the default set of statistics that are displayed in these tables includes all the criteria specified in any of the CHOOSE=, SELECT=, and STOP= options specified in the MODEL statement SELECTION= option.

You can specify the following statistics:

ADJRSQ specifies the adjusted R-square statistic.

AIC specifies Akaike's information criterion.

AICC specifies corrected Akaike's information criterion.

ASE specifies the average square errors for the training, test, and validation data. The ASE statistics for the test and validation data are reported only if you specify TESTDATA= or VALDATA= in the PROC GLMSELECT statement or if you have reserved part of the input data for testing or validation by using either a PARTITION statement or a _ROLE_ variable in the input data.

BIC specifies the Sawa Bayesian information criterion.

CP specifies the Mallows' C_p statistic.

FVALUE specifies the F statistic for entering or departing effects.

PRESS specifies the predicted residual sum of squares statistic.

RSQUARE specifies the R-square statistic.

SBC specifies the Schwarz Bayesian information criterion.

SL specifies the significance level of the F statistic for entering or departing effects.

The statistics ADJRSQ, AIC, AICC, FVALUE, RSQUARE, SBC, and SL can be computed with little computation cost. However, computing BIC, CP, CVPRESS, PRESS, and ASE for test and validation data can hurt performance when these statistics are not used in any of the CHOOSE=, SELECT=, and STOP= options that you specify in the SELECTION= option in the MODEL statement.

SHOWPVALUES
SHOWPVALS

displays p-values in the "ANOVA" and "Parameter Estimates" tables. These p-values are generally liberal because they are not adjusted for the fact that the terms in the model have been selected.

STB< =ALL | X | XY >

displays standardized regression coefficients in the "Parameter Estimates" tables. You can optionally request either or both regressor standardized and standardized regression coefficients. A regressor standardized regression coefficient is computed by dividing a parameter estimate by the sample standard deviation of the regressor. A standardized regression coefficient is computed by dividing a parameter estimate by the ratio of the sample standard deviation of the dependent variable to the sample standard deviation of the corresponding regressor. Specify STB=X to produce a column that contains the regressor standardized regression coefficients. Specify STB=XY to produce a column that contains the standardized regression coefficients. Specify STB=ALL to produce both regressor standardized and standardized regression coefficients. This option is applicable only when SELECTION=ELASTICNET, GROUPLASSO, LAR, or LASSO. If you specify STB without a value, then by default STB=XY.

TOL=*value*

specifies the tolerance value that determines the optimization precision in SELECTION= GROUPLASSO. The default *value* is 10^{-6}.

MODELAVERAGE Statement

MODELAVERAGE < *options* > **;**

The MODELAVERAGE statement requests that model selection be repeated on resampled subsets of the input data. An average model is produced by averaging the parameter estimates of the selected models that are obtained for each resampled subset of the input data.

Table 56.9 summarizes the *options* available in the MODELAVERAGE statement.

Table 56.9 MODELAVERAGE Statement Options

Option	Description
ALPHA=	Specifies lower and upper quantiles of the sample parameter
DETAILS	Displays model selection details
NSAMPLES=	Specifies the number of samples used for the refit averaging
REFIT	Performs a second round of model averaging
SAMPLING=	Specifies how to generate the samples taken from the training data
SUBSET	Uses only a subset of the selected models in forming the average model
TABLES	Controls the displayed tables

The following *options* are available:

ALPHA=α

controls which lower and upper quantiles of the sample parameter estimates are displayed. The ALPHA= option also controls which quantiles of the predicted values are added to the output data set when the LOWER= and UPPER= options are specified in the OUTPUT statement. The lower and upper quantiles used are $\alpha/2$ and $1-\alpha/2$, respectively. The value specified must lie in the interval $[0, 1]$. The default value is ALPHA=0.5.

DETAILS

requests that model selection details be displayed for each sample of the data. The level of detail shown is controlled by the DETAILS= option in the MODEL statement.

NSAMPLES=*n*

specifies the number of samples to be used. The default value is NSAMPLES=100.

REFIT < (*refit-options*) >

requests that a second round of model averaging, referred to as the refit averaging, be performed. Usually, the initial round of model averaging produces a model that contains a large number of effects. You can use the refit option to obtain a more parsimonious model. For each data sample in the refit, a least squares model is fit with no effect selection. The effects that are used in the refit depend on the results of the initial round of model averaging. If you do not specify any *refit-options*, then effects that are selected in at least twenty percent of the samples in the initial round of model averaging are used in the refit model average. The following *refit-options* are available:

BEST=*n*
specifies that the *n* most frequently selected effects in the initial round of model averaging be used in the refit averaging.

MINPCT=*percent*
specifies that the effects that are selected at least the specified percentage of times in the initial round of model averaging be used in the refit averaging.

NSAMPLES=*n*
specifies the number of samples to be used for the refit averaging. The default value is the number of samples used in the initial round of model averaging.

SAMPLING=SRS | URS < (*sampling-options***) >**
specifies how the samples of the usable observations in the training data are generated. SAMPLING=SRS specifies simple random sampling in which samples are generated by randomly drawing without replacement. SAMPLING=URS specifies unrestricted random sampling in which samples are generated by randomly drawing with replacement. Model averaging with samples drawn without replacement corresponds to the bootstrap methodology. The default is SAMPLING=URS. If you specify a frequency variable by using a FREQ statement, then the *i*th observation is sampled f_i times, where f_i is the frequency of the *i*th observation.

You can specify one of the following *sampling-options*:

PERCENT=*percent*
specifies the percentage of the training data that is used in each sample. The default value is 75% for SAMPLING=SRS and 100% for SAMPLING=URS.

SIZE=*n*
specifies the sum of frequencies in each sample.

SUBSET(*subset-options***)**
specifies that only a subset of the selected models be used in forming the average model and producing predicted values. The following *subset-options* are available:

BEST=*n*
specifies that only the best *n* models be used, where the model ranking criterion used is the frequency score. See the section "Model Selection Frequencies and Frequency Scores" on page 4477 for the definition of the frequency score. If multiple models with the same frequency score correspond to the *n*th best model, then all these tied models are used in forming the average model and producing predicted values.

MINMODELFREQ=*freq*
specifies that only models that are selected at least *freq* times be used in forming the average model and producing predicted values.

TABLES < (ONLY) > < =*table-request* **< (***options***) > >**
TABLES < (ONLY) > < = (*table-request* **< (***options***) > < . . .** *table-request* **< (***options***) > >) >**
controls the displayed output that is produced in the initial round of model averaging. By default, the following tables are produced:

EFFECTSELECTPCT displays the percentage of times that effects appear in the selected models.

MODELSELECTFREQ displays the frequency with which models are selected.

AVGPARMEST displays the mean, standard deviation, and quantiles of the parameter estimates of the parameters that appear in the selected models.

When you specify only one *table-request*, you can omit the outer parentheses. Here are some examples:

```
tables=none
tables=(all parmest(minpct=10))
tables(only)=effectselectpct(order=model minpct=15)
```

The following *table-request* options are available:

ALL

requests that all model averaging output tables be produced. You can specify other options with ALL; for example, to request all tables and to require that effects are displayed in decreasing order of selection frequency in the EffectSelectPct table, specify TABLES=(ALL EFFECTSELECTPCT(ORDER=DESCENDING)).

EFFECTSELECTPCT < (*effectSelectPct-options*) >

specifies how the effects in the EffectSelectPct table are displayed. The following *effectSelectPct-options* are available:

ALL

specifies that effects that appear in the selected model for any sample be displayed.

MINPCT=*percent*

specifies that the effects displayed must appear in the selected model for at least the specified percentage of the samples. By default, this table includes effects that appear in at least twenty percent of the selected models. The MINPCT= option is ignored if you also specify the ALL option as a *effectSelectPct-option*.

ORDER=ASCENDING | DESCENDING | MODEL

specifies the order in which the effects are displayed. ORDER=MODEL specifies that effects be displayed in the order in which they appear in the MODEL statement. ORDER= ASCENDING | DESCENDING specifies that the effects be displayed in ascending or descending order of their selection frequency.

MODELSELECTFREQ < (*modelSelectFreq-options*) >

specifies how the models in the ModelSelectFreq table are displayed. The following *modelSelectFreq-options* are available:

ALL

specifies that all selected models be displayed in the ModelSelectFreq table.

BEST=*n*
specifies that only the best *n* models be displayed, where the model ranking criterion used is the frequency score. See the section "Model Selection Frequencies and Frequency Scores" on page 4477 for the definition of the frequency score. The default value is BEST=20. The BEST= option is ignored if you also specify the ALL option as a *modelSelectFreq-option*.

ONLY
suppresses the default output. If you specify the ONLY option within parentheses after the TABLES option, then only the tables specifically requested are produced.

PARMEST < (*parmEst-options***) >**
specifies how the parameters in the AvgParmEst table are displayed. The following *parmEst-options* are available:

ALL
specifies that parameters that are nonzero in the selected model for any sample be displayed.

MINPCT=*percent*
specifies that the parameters displayed must have nonzero estimates in the selected model for at least the specified percentage of the samples. By default, this table includes parameters that appear in at least twenty percent of the selected models. The MINPCT= option is ignored if you also specify the ALL option as a *parmEst* option.

NONZEROPARMS
specifies that for each parameter, the sample that is used to compute the estimate mean, standard deviation, and quantiles consist of just the nonzero values of that parameter in the selected models. If you do not specify the NONZEROPARMS option, then parameters that do not appear in a selected model are assigned the value zero in that model and these zero values are retained when computing the estimate means, standard deviations, and quantiles.

ORDER=ASCENDING | DESCENDING | MODEL
specifies the order in which the effects are displayed. ORDER=MODEL specifies that effects are displayed in the order in which they appear in the MODEL statement. ORDER=ASCENDING | DESCENDING specifies that the effects are displayed in ascending or descending order of their selection frequency.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *keyword*< **=***name* > > . . . < *keyword*< **=***name* > > **;**

The OUTPUT statement creates a new SAS data set that saves diagnostic measures calculated for the selected model. If you do not specify a *keyword*, then the only diagnostic included is the predicted response.

All the variables in the original data set are included in the new data set, along with variables created in the OUTPUT statement. These new variables contain the values of a variety of statistics and diagnostic measures that are calculated for each observation in the data set. If you specify a BY statement, then a variable _BY_ that indexes the BY groups is included. For each observation, the value of _BY_ is the index of the BY group to which this observation belongs. This variable is useful for matching BY groups with macro variables that

PROC GLMSELECT creates. See the section "Macro Variables Containing Selected Models" on page 4472 for details.

If you have requested n-fold cross validation by requesting CHOOSE=CV, SELECT=CV, or STOP=CV in the MODEL statement, then a variable _CVINDEX_ is included in the output data set. For each observation used for model training the value of _CVINDEX_ is i if that observation is omitted in forming the ith subset of the training data. See the CVMETHOD= for additional details. The value of _CVINDEX_ is 0 for all observations in the input data set that are not used for model training.

If you have partitioned the input data with a PARTITION statement, then a character variable _ROLE_ is included in the output data set. For each observation the value of _ROLE_ is as follows:

ROLE	Observation Role
TEST	testing
TRAIN	training
VALIDATE	validation

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

Details on the specifications in the OUTPUT statement follow.

keyword< **=***name*>

specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), followed optionally by an equal sign, and a variable to contain the statistic.

If you specify *keyword*=*name*, the new variable that contains the requested statistic has the specified name. If you omit the optional =*name* after a *keyword*, then the new variable name is formed by using a prefix of one or more characters that identify the statistic. For residuals and predicted values, the prefix is followed by an underscore (_), followed by the dependent variable name.

The *keywords* allowed and the statistics they represent are as follows:

PREDICTED | PRED | P predicted values. The prefix for the default name is p.

RESIDUAL | RESID | R residual, calculated as ACTUAL – PREDICTED. The prefix for the default name is r.

When you also use the MODELAVERAGE statement, the following *keywords* and the statistics that they represent are also available:

LOWER the $100(\alpha/2)$th percentile of the sample predicted values. By default, $\alpha = 0.5$, which yields the 25th percentile. You can change the value of α by using the ALPHA= option in the MODELAVERAGE statement. The default name is *LOWER*.

MEDIAN median of the sample predicted values. The default name is *median*.

SAMPLEFREQ | SF sample frequencies. For the ith sample, a column that contains the frequencies used for that sample is added. The name of this column is formed by appending an index i to the name that you specify. If you do not specify a name, then the default prefix is sampleFreq.

SAMPLEPRED | SP sample predictions. For the ith sample, a column that contains the predicted values produced by the model selected for that sample is added. The name of this

column is formed by appending an index i to the name that you specify. If you do not specify a name, then the default prefix is samplePred.

STANDARDDEVIATION | STDDEV standard deviation of the sample predicted values. The default name is *stdDev*.

UPPER the $100(1-\alpha/2)$th percentile of the sample predicted values. By default, $\alpha = 0.5$, which yields the 75th percentile. You can change the value of α by using the ALPHA= option in the MODELAVERAGE statement. The default name is *UPPER*.

OUT=*SAS data set*
specifies the name of the new data set. By default, the procedure uses the DATA*n* convention to name the new data set.

PARTITION Statement

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. Either you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations for each role.

An alternative to using a PARTITION statement is to provide a variable named _ROLE_ in the input data set to define roles of observations in the input data. If you specify a PARTITION statement then the _ROLE_ variable if present in the input data set is ignored. If you do not use a PARTITION statement and the input data do not contain a variable named _ROLE_, then all observations in the input data set are assigned to model training.

The following mutually exclusive *options* are available:

ROLEVAR | ROLE=*variable* **(< TEST=***'value'* **> < TRAIN=***'value'* **> < VALIDATE=***'value'* **>)**
names the variable in the input data set whose values are used to assign roles to each observation. The formatted values of this variable that are used to assign observations roles are specified in the TEST=, TRAIN=, and VALIDATE= suboptions. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboptions are assigned to training. If you specify a TESTDATA= data set in the PROC GLMSELECT statement, then you cannot also specify the TEST= suboption in the PARTITION statement. If you specify a VALDATA= data set in the PROC GLMSELECT statement, then you cannot also specify the VALIDATE= suboption in the PARTITION statement.

FRACTION(< TEST=*fraction* **> < VALIDATE=***fraction* **>)**
requests that specified proportions of the observations in the input data set be randomly assigned training and validation roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and the VALIDATE= suboptions, then the sum of the specified fractions must be less than one and the remaining fraction of the observations are assigned to the training role. If you specify a TESTDATA= data set in the PROC GLMSELECT statement, then you cannot also specify the TEST= suboption in the PARTITION statement. If you specify a VALDATA= data set in the PROC GLMSELECT statement, then you cannot also specify the VALIDATE= suboption in the PARTITION statement.

PERFORMANCE Statement

PERFORMANCE < *options* > **;**

The PERFORMANCE statement is used to change default options that affect the performance of PROC GLMSELECT and to request tables that show the performance options in effect and timing details.

The following *options* are available:

DETAILS
requests the PerfSettings table that shows the performance settings in effect and the Timing table that provides a broad timing breakdown of the PROC GLMSELECT step.

BUILDSSCP=FULL | INCREMENTAL
specifies whether the SSCP matrix is built incrementally as the selection process progresses or whether the SCCP matrix for the full model is built at the outset. Building the SSCP matrix incrementally can significantly reduce the memory required and the time taken to perform model selection in cases where the number of parameters in the selected model is much smaller than the number of parameters in the full model, but it can hurt performance in other cases since it requires at least one pass through the model training data at each step. If you use backward selection or no selection, or if the BIC or CP statistics are required in the selection process, then the BUILDSSCP=INCREMENTAL option is ignored. In other cases, BUILDSSCP=INCREMENTAL is used by default if the number of effects is greater than 100. See the section "Building the SSCP Matrix" on page 4475 for further details.

SCORE Statement

SCORE < **DATA=***SAS-data-set* > < **OUT=***SAS-data-set* > < *keyword*< **=***name* > > . . . < *keyword*< **=***name* > >
;

The SCORE statement creates a new SAS data set containing predicted values and optionally residuals for data in a new data set that you name. If you do not specify a DATA= data set, then the input data are scored. If you have multiple data sets to predict, you can specify multiple SCORE statements. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

When a BY statement is used, the score data set must either contain all the BY variables sorted in the order of the BY variables or contain none of the BY variables. If the score data set contains all of the BY variables, then the model selected for a given BY group is used to score just the matching observations in the score data set. If the score data set contains none of the BY variables, then the entire score data set is scored for each BY group.

All observations in the score data set are retained in the output data set. However, only those observations that contain nonmissing values for all the continuous regressors in the selected model and whose levels of the classification variables appearing in effects of the selected model are represented in the corresponding classification variables in the procedure's input data set are scored. All the variables in the input data set are included in the output data set, along with variables containing predicted values and optionally residuals.

Details on the specifications in the SCORE statement follow:

DATA=*SAS data set*
names the data set to be scored. If you omit this option, then the input data set named in the DATA= option in the PROC GLMSELECT statement is scored.

keyword< **=***name* >
specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), followed optionally by an equal sign, and a variable to contain the statistic.

If you specify *keyword***=***name*, the new variable that contains the requested statistic has the specified name. If you omit the optional **=***name* after a *keyword*, then the new variable name is formed by using a prefix of one or more characters that identify the statistic, followed by an underscore (_), followed by the dependent variable name.

The *keywords* allowed and the statistics they represent are as follows:

PREDICTED | PRED | P predicted values. The prefix for the default name is p.

RESIDUAL | RESID | R residual, calculated as ACTUAL – PREDICTED. The prefix for the default name is r.

OUT=*SAS data set*
gives the name of the new output data set. By default, the procedure uses the DATAn convention to name the new data set.

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

A WEIGHT statement names a variable in the input data set with values that are relative weights for a weighted least squares fit. If the weight value is proportional to the reciprocal of the variance for each observation, then the weighted estimates are the best linear unbiased estimates (BLUE).

Values of the weight variable must be nonnegative. If an observation's weight is zero, the observation is deleted from the analysis. If a weight is negative or missing, it is set to zero, and the observation is excluded from the analysis. A more complete description of the WEIGHT statement can be found in Chapter 53, "The GLM Procedure."

Details: GLMSELECT Procedure

Model-Selection Methods

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/d-g/glsdt.sas.)

The model selection methods implemented in PROC GLMSELECT are specified with the SELECTION= option in the MODEL statement.

Full Model Fitted (NONE)

The complete model specified in the MODEL statement is used to fit the model and no effect selection is done. You request this by specifying SELECTION=NONE in the MODEL statement.

Forward Selection (FORWARD)

The forward selection technique begins with just the intercept and then sequentially adds the effect that most improves the fit. The process terminates when no significant improvement can be obtained by adding any effect.

In the traditional implementation of forward selection, the statistic used to gauge improvement in fit is an F statistic that reflects an effect's contribution to the model if it is included. At each step, the effect that yields the most significant F statistic is added. Note that because effects can contribute different degrees of freedom to the model, it is necessary to compare the p-values corresponding to these F statistics.

More precisely, if the current model has p parameters excluding the intercept, and if you denote its residual sum of squares by RSS_p and you add an effect with k degrees of freedom and denote the residual sum of squares of the resulting model by RSS_{p+k}, then the F statistic for entry with k numerator degrees of freedom and $n-(p+k)-1$ denominator degrees of freedom is given by

$$F = \frac{(\mathrm{RSS}_p - \mathrm{RSS}_{p+k})/k}{\mathrm{RSS}_{p+k}/(n-(p+k)-1)}$$

where n is number of observations used in the analysis.

The process stops when the significance level for adding any effect is greater than some specified entry significance level. A well-known problem with this methodology is that these F statistics do not follow an F distribution (Draper, Guttman, and Kanemasu 1971). Hence these p-values cannot reliably be interpreted as probabilities. Various ways to approximate this distribution are described by Miller (2002). Another issue when you use significance levels of entering effects as a stopping criterion arises because the entry significance level is an a priori specification that does not depend on the data. Thus, the same entry significance level can result in overfitting for some data and underfitting for other data.

One approach to address the critical problem of when to stop the selection process is to assess the quality of the models produced by the forward selection method and choose the model from this sequence that "best" balances goodness of fit against model complexity. PROC GLMSELECT supports several criteria that you can use for this purpose. These criteria fall into two groups—information criteria and criteria based on out-of-sample prediction performance.

You use the CHOOSE= option of forward selection to specify the criterion for selecting one model from the sequence of models produced. If you do not specify a CHOOSE= criterion, then the model at the final step is the selected model.

For example, if you specify

```
selection=forward(select=SL choose=AIC SLE=0.2)
```

then forward selection terminates at the step where no effect can be added at the 0.2 significance level. However, the selected model is the first one with the minimal value of Akaike's information criterion. Note that in some cases this minimal value might occur at a step much earlier that the final step, while in other cases the AIC criterion might start increasing only if more steps are done (that is, a larger value of SLE is used). If what you are interested in is minimizing AIC, then too many steps are done in the former case and too few in the latter case. To address this issue, PROC GLMSELECT enables you to specify a stopping criterion with the STOP= option. With a stopping criterion specified, forward selection continues until a local extremum of the stopping criterion in the sequence of models generated is reached. You can also specify STOP= number, which causes forward selection to continue until there are the specified number of effects in the model.

For example, if you specify

```
selection=forward(select=SL stop=AIC)
```

then forward selection terminates at the step where the effect to be added at the next step would produce a model with an AIC statistic larger than the AIC statistic of the current model. Note that in most cases, provided that the entry significance level is large enough that the local extremum of the named criterion occurs before the final step, specifying

```
selection=forward(select=SL choose=CRITERION)
```

or

```
selection=forward(select=SL stop=CRITERION)
```

selects the same model, but more steps are done in the former case. In some cases there might be a better local extremum that cannot be reached if you specify the STOP= option but can be found if you use the CHOOSE= option. Also, you can use the CHOOSE= option in preference to the STOP= option if you want examine how the named criterion behaves as you move beyond the step where the first local minimum of this criterion occurs.

Note that you can specify both the CHOOSE= and STOP= options. You might want to consider models generated by forward selection that have at most some fixed number of effects but select from within this set based on a criterion you specify. For example, specifying

```
selection=forward(stop=20 choose=ADJRSQ)
```

requests that forward selection continue until there are 20 effects in the final model and chooses among the sequence of models the one that has the largest value of the adjusted R-square statistic. You can also combine these options to select a model where one of two conditions is met. For example,

```
selection=forward(stop=AICC choose=PRESS)
```

chooses whatever occurs first between a local minimum of the predicted residual sum of squares (PRESS) and a local minimum of corrected Akaike's information criterion (AICC).

It is important to keep in mind that forward selection bases the decision about what effect to add at any step by considering models that differ by one effect from the current model. This search paradigm cannot guarantee reaching a "best" subset model. Furthermore, the add decision is greedy in the sense that the effect deemed most significant is the effect that is added. However, if your goal is to find a model that is best in terms of some selection criterion other than the significance level of the entering effect, then even this one step choice might not be optimal. For example, the effect you would add to get a model with the smallest value of the PRESS statistic at the next step is not necessarily the same effect that has the most significant entry F statistic. PROC GLMSELECT enables you to specify the criterion to optimize at each step by using the SELECT= option. For example,

```
selection=forward(select=CP)
```

requests that at each step the effect that is added be the one that gives a model with the smallest value of the Mallows' C_p statistic. Note that in the case where all effects are variables (that is, effects with one degree of freedom and no hierarchy), using ADJRSQ, AIC, AICC, BIC, CP, RSQUARE, or SBC as the selection criterion for forward selection produces the same sequence of additions. However, if the degrees of freedom contributed by different effects are not constant, or if an out-of-sample prediction-based criterion is used, then different sequences of additions might be obtained.

You can use SELECT= together with CHOOSE= and STOP=. If you specify only the SELECT= criterion, then this criterion is also used as the stopping criterion. In the previous example where only the selection criterion is specified, not only do effects enter based on the Mallows' C_p statistic, but the selection terminates when the $C(p)$ statistic first increases.

You can find discussion and references to studies about criteria for variable selection in Burnham and Anderson (2002), along with some cautions and recommendations.

Examples of Forward Selection Specifications

```
selection=forward
```

adds effects that at each step give the lowest value of the SBC statistic and stops at the step where adding any effect would increase the SBC statistic.

```
selection=forward(select=SL)
```

adds effects based on significance level and stops when all candidate effects for entry at a step have a significance level greater than the default entry significance level of 0.50.

```
selection=forward(select=SL stop=validation)
```

adds effects based on significance level and stops at a step where adding any effect increases the error sum of squares computed on the validation data.

```
selection=forward(select=AIC)
```

adds effects that at each step give the lowest value of the AIC statistic and stops at the step where adding any effect would increase the AIC statistic.

```
selection=forward(select=ADJRSQ stop=SL SLE=0.2)
```

adds effects that at each step give the largest value of the adjusted R-square statistic and stops at the step where the significance level corresponding to the addition of this effect is greater than 0.2.

Backward Elimination (BACKWARD)

The backward elimination technique starts from the full model including all independent effects. Then effects are deleted one by one until a stopping condition is satisfied. At each step, the effect showing the smallest contribution to the model is deleted. In traditional implementations of backward elimination, the contribution of an effect to the model is assessed by using an F statistic. At any step, the predictor producing the least significant F statistic is dropped and the process continues until all effects remaining in the model have F statistics significant at a stay significance level (SLS).

More precisely, if the current model has p parameters excluding the intercept, and if you denote its residual sum of squares by RSS_p and you drop an effect with k degrees of freedom and denote the residual sum of squares of the resulting model by RSS_{p-k}, then the F statistic for removal with k numerator degrees of freedom and $n - p - 1$ denominator degrees of freedom is given by

$$F = \frac{(\mathrm{RSS}_{p-k} - \mathrm{RSS}_p)/k}{\mathrm{RSS}_p/(n - p - 1)}$$

where n is number of observations used in the analysis.

Just as with forward selection, you can change the criterion used to assess effect contributions with the SELECT= option. You can also specify a stopping criterion with the STOP= option and use a CHOOSE= option to provide a criterion used to select among the sequence of models produced. See the discussion in the section "Forward Selection (FORWARD)" on page 4458 for additional details.

Examples of Backward Selection Specifications

```
selection=backward
```

removes effects that at each step produce the largest value of the Schwarz Bayesian information criterion (SBC) statistic and stops at the step where removing any effect increases the SBC statistic.

```
selection=backward(stop=press)
```

removes effects based on the SBC statistic and stops at the step where removing any effect increases the predicted residual sum of squares (PRESS).

```
selection=backward(select=SL)
```

removes effects based on significance level and stops when all candidate effects for removal at a step have a significance level less than the default stay significance level of 0.10.

```
selection=backward(select=SL choose=validate SLS=0.1)
```

removes effects based on significance level and stops when all effects in the model are significant at the 0.1 level. Finally, from the sequence of models generated, choose the one that gives the smallest average square error when scored on the validation data.

Stepwise Selection (STEPWISE)

The stepwise method is a modification of the forward selection technique that differs in that effects already in the model do not necessarily stay there.

In the traditional implementation of stepwise selection method, the same entry and removal F statistics for the forward selection and backward elimination methods are used to assess contributions of effects as they are added to or removed from a model. If at a step of the stepwise method, any effect in the model is not significant at the SLSTAY= level, then the least significant of these effects is removed from the model and the algorithm proceeds to the next step. This ensures that no effect can be added to a model while some effect currently in the model is not deemed significant. Only after all necessary deletions have been accomplished can another effect be added to the model. In this case the effect whose addition yields the most significant F value is added to the model and the algorithm proceeds to the next step. The stepwise process ends when none of the effects outside the model has an F statistic significant at the SLENTRY= level and every effect in the model is significant at the SLSTAY= level. In some cases, neither of these two conditions for stopping is met and the sequence of models cycles. In this case, the stepwise method terminates at the end of the second cycle.

Just as with forward selection and backward elimination, you can change the criterion used to assess effect contributions, with the SELECT= option. You can also specify a stopping criterion with the STOP= option and use a CHOOSE= option to provide a criterion used to select among the sequence of models produced. See the discussion in the section "Forward Selection (FORWARD)" on page 4458 for additional details.

For selection criteria other than significance level, PROC GLMSELECT optionally supports a further modification in the stepwise method. In the standard stepwise method, no effect can enter the model if removing any effect currently in the model would yield an improved value of the selection criterion. In the modification, you can use the DROP=COMPETITIVE option to specify that addition and deletion of effects should be treated competitively. The selection criterion is evaluated for all models obtained by deleting an effect from the current model or by adding an effect to this model. The action that most improves the selection criterion is the action taken.

Examples of Stepwise Selection Specifications

```
selection=stepwise
```

requests stepwise selection based on the SBC criterion. First, if removing any effect yields a model with a lower SBC statistic than the current model, then the effect producing the smallest SBC statistic is removed. When removing any effect increases the SBC statistic, then provided that adding some effect lowers the SBC statistic, the effect producing the model with the lowest SBC is added.

```
selection=stepwise(select=SL)
```

requests the traditional stepwise method. First, if the removal of any effect yields an F statistic that is not significant at the default stay level of 0.15, then the effect whose removal produces the least significant F statistic is removed and the algorithm proceeds to the next step. Otherwise the effect whose addition yields the most significant F statistic is added, provided that it is significant at the default entry level of 0.15.

```
selection=stepwise(select=SL stop=SBC)
```

is the traditional stepwise method, where effects enter and leave based on significance levels, but with the following extra check: If any effect to be added or removed yields a model whose SBC statistic is greater than the SBC statistic of the current model, then the stepwise method terminates at the current model. Note that in this case, the entry and stay significance levels still play a role as they determine whether an effect is

deleted from or added to the model. This might result in the selection terminating before a local minimum of the SBC criterion is found.

```
selection=stepwise(select=SL SLE=0.1 SLS=0.08 choose=AIC)
```

selects effects to enter or drop as in the previous example except that the significance level for entry is now 0.1 and the significance level to stay is 0.08. From the sequence of models produced, the selected model is chosen to yield the minimum AIC statistic.

```
selection=stepwise(select=AICC drop=COMPETITIVE)
```

requests stepwise selection based on the AICC criterion with steps treated competitively. At any step, evaluate the AICC statistics corresponding to the removal of any effect in the current model or the addition of any effect to the current model. Choose the addition or removal that produced this minimum value, provided that this minimum is lower than the AICC statistic of the current model.

```
selection=stepwise(select=SBC drop=COMPETITIVE stop=VALIDATE)
```

requests stepwise selection based on the SBC criterion with steps treated competitively and where stopping is based on the average square error over the validation data. At any step, SBC statistics corresponding to the removal of any effect from the current model or the addition of any effect to the current model are evaluated. The addition or removal that produces the minimum SBC value is made. The average square error on the validation data for the model with this addition or removal is evaluated. If this average square error is greater than the average square error on the validation data prior to this addition or deletion, then the algorithm terminates at this prior model.

Least Angle Regression (LAR)

Least angle regression was introduced by Efron et al. (2004). Not only does this algorithm provide a selection method in its own right, but with one additional modification it can be used to efficiently produce LASSO solutions. Just like the forward selection method, the LAR algorithm produces a sequence of regression models where one parameter is added at each step, terminating at the full least squares solution when all parameters have entered the model.

The algorithm starts by centering the covariates and response, and scaling the covariates so that they all have the same corrected sum of squares. Initially all coefficients are zero, as is the predicted response. The predictor that is most correlated with the current residual is determined and a step is taken in the direction of this predictor. The length of this step determines the coefficient of this predictor and is chosen so that some other predictor and the current predicted response have the same correlation with the current residual. At this point, the predicted response moves in the direction that is equiangular between these two predictors. Moving in this direction ensures that these two predictors continue to have a common correlation with the current residual. The predicted response moves in this direction until a third predictor has the same correlation with the current residual as the two predictors already in the model. A new direction is determined that is equiangular between these three predictors and the predicted response moves in this direction until a fourth predictor joins the set having the same correlation with the current residual. This process continues until all predictors are in the model.

As with other selection methods, the issue of when to stop the selection process is crucial. You can specify a criterion to use to choose among the models at each step with the CHOOSE= option. You can also specify a stopping criterion with the STOP= option. See the section "Criteria Used in Model Selection Methods" on page 4469 for details and Table 56.10 for the formulas for evaluating these criteria. These formulas use

the approximation that at step k of the LAR algorithm, the model has k degrees of freedom. See Efron et al. (2004) for a detailed discussion of this so-called simple approximation.

A modification of LAR selection suggested in Efron et al. (2004) uses the LAR algorithm to select the set of covariates in the model at any step, but uses ordinary least squares regression with just these covariates to obtain the regression coefficients. You can request this hybrid method by specifying the LSCOEFFS suboption of SELECTION=LAR.

Lasso Selection (LASSO)

LASSO (least absolute shrinkage and selection operator) selection arises from a constrained form of ordinary least squares regression where the sum of the absolute values of the regression coefficients is constrained to be smaller than a specified parameter. More precisely let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$ denote the matrix of covariates and let $\mathbf{y}$ denote the response, where the $\mathbf{x}_i$s have been centered and scaled to have unit standard deviation and mean zero, and $\mathbf{y}$ has mean zero. Then for a given parameter t, the LASSO regression coefficients $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the constrained optimization problem

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \qquad \sum_{j=1}^{m} |\beta_j| \le t$$

Provided that the LASSO parameter t is small enough, some of the regression coefficients will be exactly zero. Hence, you can view the LASSO as selecting a subset of the regression coefficients for each LASSO parameter. By increasing the LASSO parameter in discrete steps, you obtain a sequence of regression coefficients where the nonzero coefficients at each step correspond to selected parameters.

Early implementations (Tibshirani 1996) of LASSO selection used quadratic programming techniques to solve the constrained least squares problem for each LASSO parameter of interest. Later Osborne, Presnell, and Turlach (2000) developed a "homotopy method" that generates the LASSO solutions for all values of t. Efron et al. (2004) derived a variant of their algorithm for least angle regression that can be used to obtain a sequence of LASSO solutions from which all other LASSO solutions can be obtained by linear interpolation. This algorithm for SELECTION=LASSO is used in PROC GLMSELECT. It can be viewed as a stepwise procedure with a single addition to or deletion from the set of nonzero regression coefficients at any step.

As with the other selection methods that PROC GLMSELECT supports, you can specify a criterion to choose among the models at each step of the LASSO algorithm by using the CHOOSE= option. You can also specify a stopping criterion by using the STOP= option. For more information, see the discussion in the section "Forward Selection (FORWARD)" on page 4458. The model degrees of freedom that PROC GLMSELECT uses at any step of the LASSO are simply the number of nonzero regression coefficients in the model at that step. Efron et al. (2004) cite empirical evidence for doing this but do not give any mathematical justification for this choice.

A modification of LASSO selection that is suggested in Efron et al. (2004) uses the LASSO algorithm to select the set of covariates in the model at any step, but it uses ordinary least squares regression with just these covariates to obtain the regression coefficients. You can request this hybrid method by specifying the LSCOEFFS suboption of the SELECTION=LASSO option.

Adaptive LASSO Selection

Adaptive LASSO selection is a modification of LASSO selection; in adaptive LASSO selection, weights are applied to each of the parameters in forming the LASSO constraint (Zou 2006). More precisely, suppose that the response $\mathbf{y}$ has mean zero and the regressors $\mathbf{x}$ are scaled to have mean zero and common standard deviation. Furthermore, suppose you can find a suitable estimator $\hat{\boldsymbol{\beta}}$ of the parameters in the true model and you define a weight vector by $w = 1/|\hat{\boldsymbol{\beta}}|^{\gamma}$, where $\gamma \geq 0$. Then the adaptive LASSO regression coefficients $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the constrained optimization problem

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{m} |w_j \beta_j| \leq t$$

You can specify $\hat{\boldsymbol{\beta}}$ by using the INEST=suboption of the SELECTION=LASSO option in the MODEL statement. The INEST= data set has the same structure as the OUTEST= data set that is produced by several SAS/STAT procedures, including the REG and LOGISTIC procedures. The INEST= data set must contain all explanatory variables in the MODEL statement. It must also contain an intercept variable named Intercept unless you specify the NOINT option in the MODEL statement. If BY processing is used, the INEST= data set must also include the BY variables, and there must be one observation for each BY group. If the INEST= data set also contains the _TYPE_ variable, only observations whose _TYPE_ value is `'PARMS'` are used.

If you do not specify an INEST= data set, then PROC GLMSELECT uses the solution to the unconstrained least squares problem as the estimator $\hat{\boldsymbol{\beta}}$. This is appropriate unless collinearity is a concern. If the regressors are collinear or nearly collinear, then Zou (2006) suggests using a ridge regression estimate to form the adaptive weights.

Elastic Net Selection (ELASTICNET)

The elastic net method bridges the LASSO method and ridge regression. It balances having a parsimonious model with borrowing strength from correlated regressors, by solving the least squares regression problem with constraints on both the sum of the absolute coefficients and the sum of the squared coefficients. More specifically, the elastic net coefficients $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the constrained optimization problem

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \quad \sum_{j=1}^{m} |\beta_j| \leq t_1, \sum_{j=1}^{m} \beta_j^2 \leq t_2$$

The method can be written as the equivalent Lagrangian form

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda_1 \sum_{j=1}^{m} |\beta_j| + \lambda_2 \sum_{j=1}^{m} \beta_j^2$$

If t_1 is set to a very large value or, equivalently, if λ_1 is set to 0, then the elastic net method reduces to ridge regression. If t_2 is set to a very large value or, equivalently, if λ_2 is set to 0, then the elastic net method

reduces to LASSO. If t_1 and t_2 are both large or, equivalently, if λ_1 and λ_2 are both set to 0, then the elastic net method reduces to ordinary least squares regression.

As stated by Zou and Hastie (2005), the elastic net method can overcome the limitations of LASSO in the following three scenarios:

- In the case where you have more parameters than observations, $m > n$, the LASSO method selects at most n variables before it saturates, because of the nature of the convex optimization problem. This can be a defect for a variable selection method. By contrast, the elastic net method can select more than n variables in this case because of the ridge regression regularization.
- If there is a group of variables that have high pairwise correlations, then whereas LASSO tends to select only one variable from that group, the elastic net method can select more than one variable.
- In the $n > m$ case, if there are high correlations between predictors, it has been empirically observed that the prediction performance of LASSO is dominated by ridge regression. In this case, the elastic net method can achieve better prediction performance by using ridge regression regularization.

An elastic net fit is achieved by building on LASSO estimation, in the following sense. Let $\tilde{\mathbf{X}}$ be a matrix obtained by augmenting $\mathbf{X}$ with a scaled identity matrix,

$$\tilde{\mathbf{X}} = [\mathbf{X}; \sqrt{\lambda_2} I]$$

Let $\tilde{\mathbf{y}}$ be a vector correspondingly obtained by augmenting the response $\mathbf{y}$ with m 0's,

$$\tilde{\mathbf{y}} = [\mathbf{y}; \mathbf{0}]$$

Then the Lagrangian form of the elastic net optimization problem can be reformulated as

$$\min ||\tilde{\mathbf{y}} - \tilde{\mathbf{X}}\boldsymbol{\beta}||^2 + \lambda_1 \sum_{j=1}^{m} |\beta_j|$$

In other words, you can solve the elastic net method in the same way as LASSO by using this augmented design matrix $\tilde{\mathbf{X}}$ and response $\tilde{\mathbf{y}}$. Therefore, for given λ_2, the coefficients of the elastic net fit follow the same piecewise linear path as LASSO. Zou and Hastie (2005) suggest rescaling the coefficients by $1 + \lambda_2$ to deal with the double amount of shrinkage in the elastic net fit, and such rescaling is applied when you specify the ENSCALE option in the MODEL statement.

If you have a good estimate of λ_2, you can specify the value in the L2= option. If you do not specify a value for λ_2, then by default PROC GLMSELECT searches for a value between 0 and 1 that is optimal according to the current CHOOSE= criterion. Figure 56.12 illustrates the estimation of the ridge regression parameter λ_2 (L2). Meanwhile, if you do not specify the CHOOSE= option, then the model at the final step in the selection process is selected for each λ_2 (L2), and the criterion value shown in Figure 56.12 is the one at the final step that corresponds to the specified STOP= option (STOP=SBC by default).

Figure 56.12 Estimation of the Ridge Regression Parameter λ_2 (L2) in the Elastic Net Method

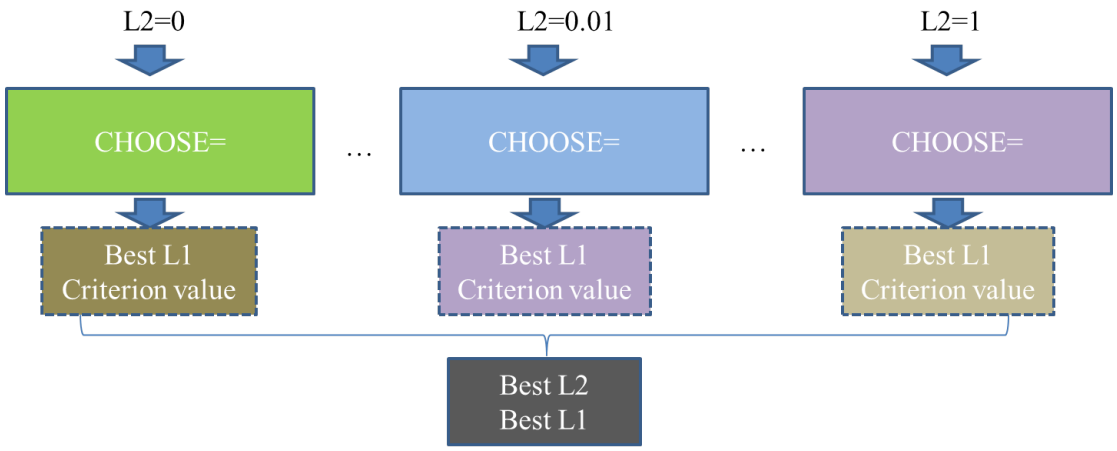

Note that when you specify the L2SEARCH=GOLDEN, it is assumed that the criterion curve that corresponds to the CHOOSE= option with respect to λ_2 is a smooth and bowl-shaped curve. However, this assumption is not checked and validated. Hence, the default value for the L2SEARCH= option is set to GRID.

Group LASSO Selection (GROUPLASSO)

The group LASSO method proposed by Yuan and Lin (2006) is a variant of LASSO that is specifically designed for linear models defined in terms of effects that have multiple degrees of freedom, such as the main effects of CLASS variables, interactions between CLASS variables, and effects defined using an EFFECT statement.

Recall that LASSO selection depends on solving a constrained least squares problem of the form

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \qquad \sum_{j=1}^{m} |\beta_j| \leq t$$

In this formulation, you can include or exclude individual parameters from the model independently, subject only to the overall constraint. In contrast, the group LASSO method uses a constraint that forces all parameters that correspond to the same effect to be included or excluded simultaneously. For a model that has k effects, let β_{G_j} be the group of linear coefficients that correspond to effect j in the model. Then group LASSO depends on solving a constrained optimization problem of the form

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \qquad \sum_{j=1}^{k} \sqrt{|G_j|}||\beta_{G_j}|| \leq t$$

where $|G_j|$ is the number of parameters that correspond to effect j, and $||\beta_{G_j}||$ denotes the Euclidean norm of the parameters β_{G_j}. That is, instead of constraining the sum of the absolute value of individual parameters, group LASSO constrains the Euclidean norm of groups of parameters, where groups are defined by effects.

You can write the group LASSO method in the equivalent Lagrangian form

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda \sum_{j=1}^{k} \sqrt{|G_j|}||\beta_{G_j}||$$

The weight $\sqrt{|G_j|}$, as suggested by Yuan and Lin (2006), should take the size of the group into consideration in group LASSO.

Unlike LASSO, group LASSO does not allow a piecewise linear constant solution path as generated by a LAR algorithm. Instead, the method that Nesterov (2013) proposes is adopted to solve the Lagrangian form of the group LASSO problem that corresponds to a prespecified regularization parameter, λ. Nesterov's method is known to have an optimal convergence rate for first-order black box optimization. Because the optimal λ is usually unknown, a sequence of regularization parameters, $\rho, \rho^2, \rho^3, \ldots$, is employed, where ρ is a positive value less than 1. You can specify ρ by using the RHO= suboption of the SELECTION= option in the MODEL statement; by default, RHO=0.9. In the ith step of group LASSO selection, the value used for λ is ρ^i. If you want the solution that corresponds to a prespecified λ, you can specify the value of λ by using the L1= option together with STOP=L1.

Another unique feature of the group LASSO method is that it does not necessarily add or remove precisely one effect at each step of the process. This is different from the forward, stepwise, backward, LAR, LASSO, and elastic net selection methods.

Model Selection Issues

Many authors caution against the use of "automatic variable selection" methods and describe pitfalls that plague many such methods. For example, Harrell (2001) states that "stepwise variable selection has been a very popular technique for many years, but if this procedure had just been proposed as a statistical method, it would most likely be rejected because it violates every principle of statistical estimation and hypothesis testing." He lists and discusses several of these issues and cites a variety of studies that highlight these problems. He also notes that many of these issues are not restricted to stepwise selection but affect forward selection and backward elimination, as well as methods based on all-subset selection.

In their introductory chapter, Burnham and Anderson (2002) discuss many issues involved in model selection. They also strongly warn against "data dredging," which they describe as "the process of analyzing data with few or no a priori questions, by subjectively and iteratively searching the data for patterns and 'significance.'" However, Burnham and Anderson also discuss the desirability of finding parsimonious models. They note that using "full models" that contain many insignificant predictors might avoid some of the inferential problems arising in models with automatically selected variables but will lead to overfitting the particular sample data and produce a model that performs poorly in predicting data not used in training the model.

One problem in the traditional implementations of forward, backward, and stepwise selection methods is that they are based on sequential testing with specified entry (SLE) and stay (SLS) significance levels. However, it is known that the "F-to-enter" and "F-to-delete" statistics do not follow an F distribution (Draper, Guttman, and Kanemasu 1971). Hence the SLE and SLS values cannot reliably be viewed as probabilities. One way to address this difficulty is to replace hypothesis testing as a means of selecting a model with information criteria or out-of-sample prediction criteria. While Harrell (2001) points out that information criteria were developed for comparing only prespecified models, Burnham and Anderson (2002) note that AIC criteria have routinely been used for several decades for performing model selection in time series analysis.

Problems also arise when the selected model is interpreted as if it were prespecified. There is a "selection bias" in the parameter estimates that is discussed in detail in Miller (2002). This bias occurs because a parameter is more likely to be selected if it is above its expected value than if it is below its expected value. Furthermore, because multiple comparisons are made in obtaining the selected model, the p-values obtained

for the selected model are not valid. When a single best model is selected, inference is conditional on that model.

Model averaging approaches provide a way to make more stable inferences based on a set of models. PROC GLMSELECT provides support for model averaging by averaging models that are selected on resampled data. Other approaches for performing model averaging are presented in Burnham and Anderson (2002), and Bayesian approaches are discussed in Raftery, Madigan, and Hoeting (1997).

Despite these difficulties, careful and informed use of variable selection methods still has its place in modern data analysis. For example, Foster and Stine (2004) use a modified version of stepwise selection to build a predictive model for bankruptcy from over 67,000 possible predictors and show that this yields a model whose predictions compare favorably with other recently developed data mining tools. In particular, when the goal is prediction rather than estimation or hypothesis testing, variable selection with careful use of validation to limit both under and over fitting is often a useful starting point of model development.

Criteria Used in Model Selection Methods

PROC GLMSELECT supports a variety of fit statistics that you can specify as criteria for the CHOOSE=, SELECT=, and STOP= options in the MODEL statement. The following statistics are available:

ADJRSQ	adjusted R-square statistic (Darlington 1968; Judge et al. 1985)
AIC	Akaike's information criterion (Darlington 1968; Judge et al. 1985)
AICC	corrected Akaike's information criterion (Hurvich and Tsai 1989)
BIC	Sawa Bayesian information criterion (Sawa 1978; Judge et al. 1985)
CP	Mallows' C_p statistic (Mallows 1973; Hocking 1976)
PRESS	predicted residual sum of squares statistic
SBC	Schwarz Bayesian information criterion (Schwarz 1978; Judge et al. 1985)
SL	significance level of the F statistic used to assess an effect's contribution to the fit when it is added to or removed from a model
VALIDATE	average square error over the validation data

Table 56.10 provides formulas and definitions for the fit statistics.

Table 56.10 Formulas and Definitions for Model Fit Summary Statistics

Statistic	Definition or Formula
n	Number of observations
p	Number of parameters including the intercept
$\hat{\sigma}^2$	Estimate of pure error variance from fitting the full model
SST	Total sum of squares corrected for the mean for the dependent variable
SSE	Error sum of squares
ASE	$\frac{\text{SSE}}{n}$

Table 56.10 *continued*

Statistic	Definition or Formula
MSE	$\frac{\text{SSE}}{n-p}$
R^2	$1-\frac{\text{SSE}}{\text{SST}}$
ADJRSQ	$1-\frac{(n-1)(1-R^2)}{n-p}$
AIC	$n\log\left(\frac{\text{SSE}}{n}\right)+2p+n+2$
AICC	$n\log\left(\frac{\text{SSE}}{n}\right)+\frac{n(n+p)}{n-p-2}$
BIC	$n\log\left(\frac{\text{SSE}}{n}\right)+2(p+2)q-2q^2$ where $q=\frac{n\hat{\sigma}^2}{\text{SSE}}$
CP (C_p)	$\frac{\text{SSE}}{\hat{\sigma}^2}+2p-n$
PRESS	$\sum_{i=1}^{n}\frac{r_i^2}{(1-h_i)^2}$ where r_i = residual at observation i and h_i = leverage of observation i = $\mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-}\mathbf{x}_i'$
RMSE	$\sqrt{\text{MSE}}$
SBC	$n\log\left(\frac{\text{SSE}}{n}\right)+p\log(n)$

Formulas for AIC and AICC

In the context of linear regression, several different versions of the formulas for AIC and AICC appear in the statistics literature. However, for a fixed number of observations, these different versions differ by additive and positive multiplicative constants.

PROC GLMSLECT now uses the definitions of AIC and AICC found in Hurvich and Tsai (1989):

$$\text{AIC}=n\log\left(\frac{\text{SSE}}{n}\right)+2p+n+2$$

and

$$\text{AICC}=\text{AIC}+\frac{2(p+1)(p+2)}{n-p-2}$$

Hurvich and Tsai (1989) show that the formula for AICC can also be written as

$$\text{AICC}=n\log\left(\frac{\text{SSE}}{n}\right)+\frac{n(n+p)}{n-p-2}$$

CLASS Variable Parameterization and the SPLIT Option

The GLMSELECT procedure supports nonsingular parameterizations for classification effects. A variety of these nonsingular parameterizations are available. You use the PARAM= option in the CLASS statement to specify the parameterization. See the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics," for details.

PROC GLMSELECT also supports the ability to split classification effects. You can use the SPLIT option in the CLASS statement to request that the columns of the design matrix that correspond to any effect that contains a split classification variable can be selected to enter or leave a model independently of the other design columns of that effect. The following statements illustrate the use of SPLIT option together with other features of the CLASS statement:

```
data codingExample;
   drop i;
   do i=1 to 1000;
      c1 = 1 + mod(i,6);
      if      i < 50  then c2 = 'very low ';
      else if i < 250 then c2 = 'low';
      else if i < 500 then c2 = 'medium';
      else if i < 800 then c2 = 'high';
      else                 c2 = 'very high';
      x1 = ranuni(1);
      x2 = ranuni(1);
      y = x1 + 10*(c1=3) +5*(c1=5) +rannor(1);
      output;
   end;
run;
proc glmselect data=codingExample;
   class c1(param=ref split) c2(param=ordinal order=data) /
          delimiter = ',' showcoding;
   model y = c1 c2 x1 x2/orderselect;
run;
```

The "Class Level Information" table shown in Figure 56.13 is produced by default whenever you specify a CLASS statement.

Figure 56.13 Class Level Information

The GLMSELECT Procedure

Class Level Information		
Class	**Levels**	**Values**
c1	6	* 1,2,3,4,5,6
c2	5	very low,low,medium,high,very high
* Associated Parameters Split		

Note that because the levels of the variable c2 contain embedded blanks, the DELIMITER=`','` option has been specified. The SHOWCODING option requests the display of the "Class Level Coding" table shown in Figure 56.14. An ordinal parameterization is used for c2 because its levels have a natural order. Furthermore, because these levels appear in their natural order in the data, you can preserve this order by specifying the ORDER=DATA option.

Figure 56.14 Class Level Coding

Class Level Coding					
	Design Variables				
c1 Level	1	2	3	4	5
1	1	0	0	0	0
2	0	1	0	0	0
3	0	0	1	0	0
4	0	0	0	1	0
5	0	0	0	0	1
6	0	0	0	0	0

The SPLIT option has been specified for the classification variable c1. This permits the parameters associated with the effect c1 to enter or leave the model individually. The "Parameter Estimates" table in Figure 56.15 shows that for this example the parameters that correspond to only levels 3 and 5 of c1 are in the selected model. Finally, note that the ORDERSELECT option in the MODEL statement specifies that the parameters are displayed in the order in which they first entered the model.

Figure 56.15 Parameter Estimates

Parameter Estimates				
Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	-0.216680	0.068650	-3.16
c1_3	1	10.160900	0.087898	115.60
c1_5	1	5.018015	0.087885	57.10
x1	1	1.315468	0.109772	11.98

Macro Variables Containing Selected Models

Often you might want to perform postselection analysis by using other SAS procedures. To facilitate this, PROC GLMSELECT saves the list of selected effects in a macro variable. This list does not explicitly include the intercept so that you can use it in the MODEL statement of other SAS/STAT regression procedures.

The following table describes the macro variables that PROC GLMSELECT creates. Note that when BY processing is used, one macro variable, indexed by the BY group number, is created for each BY group.

Macro Variable	Description
No BY Processing	
_GLSIND1	Selected model
BY Processing	
_GLSNUMBYS	Number of BY groups
_GLSIND1	Selected model for BY group 1
_GLSIND2	Selected model for BY group 2
...	

You can use the macro variable _GLSIND as a synonym for _GLSIND1. If you do not use BY processing, _GLSNUMBYS is still defined and has the value 1.

To aid in associating indexed macro variables with the appropriate observations when BY processing is used, PROC GLMSELECT creates a variable _BY_ in the output data set specified in an OUTPUT statement (see the section "OUTPUT Statement" on page 4453) that tags observations with an index that matches the index of the appropriate macro variable.

The following statements create a data set with two BY groups and run PROC GLMSELECT to select a model for each BY group.

```
data one(drop=i j);
   array x{5} x1-x5;
   do i=1 to 1000;
      classVar = mod(i,4)+1;
      do j=1 to 5;
         x{j} = ranuni(1);
      end;
      if i<400 then do;
         byVar = 'group 1';
         y     = 3*classVar+7*x2+5*x2*x5+rannor(1);
      end;
      else do;
         byVar = 'group 2';
         y     = 2*classVar+x5+rannor(1);
      end;
      output;
   end;
run;

proc glmselect data=one;
   by     byVar;
   class  classVar;
   model  y = classVar x1|x2|x3|x4|x5 @2 /
                  selection=stepwise(stop=aicc);
   output out=glmselectOutput;
run;
```

The preceding PROC GLMSELECT step produces three macro variables:

Macro Variable	Value	Description
_GLSNUMBYS	2	Number of BY groups
_GLSIND1	classVar x2 x2*x5	Selected model for the first BY group
_GLSIND2	classVar x5	Selected model for the second BY group

You can now leverage these macro variables and the output data set created by PROC GLMSELECT to perform postselection analyses that match the selected models with the appropriate BY-group observations. For example, the following statements create and run a macro that uses PROC GLM to perform LSMeans analyses.

```
%macro LSMeansAnalysis;
   %do i=1 %to &_GLSNUMBYS;
      title1  "Analysis Using the Selected Model for BY group number &i";
      title2 "Selected Effects: &&_GLSIND&i";

      ods select LSMeans;
      proc glm data=glmselectOutput(where = (_BY_ = &i));
         class classVar;
         model y = &&_GLSIND&i;
         lsmeans classVar;
      run;quit;
   %end;
%mend;
%LSMeansAnalysis;
```

The LSMeans analysis output from PROC GLM is shown in Output 56.16.

Figure 56.16 LS-Means Analyses for Selected Models

Analysis Using the Selected Model for BY group number 1
Selected Effects: classVar x2 x2*x5

The GLM Procedure
Least Squares Means

classVar	y LSMEAN
1	7.8832052
2	10.9528618
3	13.9412216
4	16.7929355

Analysis Using the Selected Model for BY group number 2
Selected Effects: classVar x5

The GLM Procedure
Least Squares Means

classVar	y LSMEAN
1	2.46805014
2	4.52102826
3	6.53369479
4	8.49354763

Using the STORE Statement

The preceding section shows how you can use macro variables to facilitate performing postselection analysis by using other SAS procedures. An alternative approach is to use the STORE statement to save the results of the PROC GLMSELECT step in an *item store*. You can then use the PLM procedure to obtain a rich set of postselection analyses. The following statements show how you can use this approach to obtain the same LSMeans analyses as shown in section "Macro Variables Containing Selected Models" on page 4472:

```
proc glmselect data=one;
   by     byVar;
   class  classVar;
   model  y = classVar x1|x2|x3|x4|x5 @2 /
                  selection=stepwise(stop=aicc);
   store out=glmselectStore;
run;

proc plm source=glmselectStore;
   lsmeans classVar;
run;
```

The LSMeans analysis output for the first BY group is shown in Figure 56.17.

Figure 56.17 LS-Means Analysis Produced by PROC PLM

The PLM Procedure

classVar Least Squares Means

classVar	Estimate	Standard Error	DF	t Value	Pr > |t|
1	7.8832	0.1050	393	75.11	<.0001
2	10.9529	0.1043	393	104.99	<.0001
3	13.9412	0.1043	393	133.70	<.0001
4	16.7929	0.1042	393	161.09	<.0001

Building the SSCP Matrix

Traditional implementations of FORWARD and STEPWISE selection methods start by computing the augmented crossproduct matrix for all the specified effects. This initial crossproduct matrix is updated as effects enter or leave the current model by sweeping the columns corresponding to the parameters of the entering or departing effects. Building the starting crossproduct matrix can be done with a single pass through the data and requires $O(m^2)$ storage and $O(nm^2)$ work, where n is the number of observations and m is the number of parameters. If k selection steps are done, then the total work sweeping effects in and out of the model is $O(km^2)$. When $n >> m$, the work required is dominated by the time spent forming the crossproduct matrix. However, when m is large (tens of thousands), just storing the crossproduct matrix becomes intractable even though the number of selected parameters might be small. Note also that when interactions of classification effects are considered, the number of parameters considered can be large, even though the number of effects considered is much smaller.

When the number of selected parameters is smaller than the total number of parameters, it turns out that many of the crossproducts are not needed in the selection process. Let $\mathbf{y}$ denote the dependent variable, and suppose at some step of the selection process that $\mathbf{X}$ denotes the $n \times p$ design matrix columns corresponding to the currently selected model. Let $\mathbf{Z} = \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{m-p}$ denote the design matrix columns corresponding to the $m - p$ effects not yet in the model. Then in order to compute the reduction in the residual sum of squares when $\mathbf{z}_j$ is added to the model, the only additional crossproducts needed are $\mathbf{z}'_j y$, $\mathbf{z}'_j \mathbf{X}$, and $\mathbf{z}'_j \mathbf{z}_j$. Note that it is not necessary to compute any of $\mathbf{z}'_j \mathbf{z}_i$ with $i \neq j$ and if $p << m$, and this yields a substantial saving in both memory required and computational work. Note, however, that this strategy does require a pass through the data at any step where adding an effect to the model is considered.

PROC GLMSELECT supports both of these strategies for building the crossproduct matrix. You can choose which of these strategies to use by specifying the BUILDSSCP=FULL or BUILDSSCP=INCREMENTAL option in the PERFORMANCE statement. If you request BACKWARD selection, then the full SSCP matrix is required. Similarly, if you request the BIC or CP criterion as the SELECT=, CHOOSE=, or STOP= criterion, or if you request the display of one or both of these criteria with the STATS=BIC, STATS=CP, or STATS=ALL option, then the full model needs to be computed. If you do not specify the BUILDSSCP= option, then PROC GLMSELECT switches to the incremental strategy if the number of effects is greater than one hundred. This default strategy is designed to give good performance when the number of selected parameters is less than about 20% of the total number of parameters. Hence if you choose options that you know will cause the selected model to contain a significantly higher percentage of the total number of candidate parameters, then you should consider specifying BUILDSSCP=FULL. Conversely, if you specify fewer than 100 effects in the MODEL statement but many of these effects have a large number of associated parameters, then specifying BUILDSSCP=INCREMENTAL might result in improved performance.

Model Averaging

As discussed in the section "Model Selection Issues" on page 4468, some well-known issues arise in performing model selection for inference and prediction. One approach to address these issues is to use resampled data as a proxy for multiple samples that are drawn from some conceptual probability distribution. A model is selected for each resampled set of data, and a predictive model is built by averaging the predictions of these selected models. You can perform this method of model averaging by using the MODELAVERAGE statement. Resampling-based methods, in which samples are obtained by drawing with replacement from your data, fall under the umbrella of the widely studied methodology known as the bootstrap (Efron and Tibshirani 1993). For use of the bootstrap in the context of variable selection, see Breiman (1992).

By default, when the average is formed, models that are selected in multiple samples receive more weight than infrequently selected models. Alternatively, you can start by fitting a prespecified set of models on your data, then use information-theoretic approaches to assign a weight to each model in building a weighted average model. You can find a detailed discussion of this methodology in Burnham and Anderson (2002), in addition to some comparisons of this approach with bootstrap-based methods.

In the linear model context, the average prediction that you obtain from a set of models is the same as the prediction that you obtain with the single model whose parameter estimates are the averages of the corresponding estimates of the set of models. Hence, you can regard model averaging as a selection method that selects this average model. To show this, denote by $\beta^{(i)}$ the parameter estimates for the sample i where $\beta_j^{(i)} = 0$ if parameter j is not in the selected model for sample i. Then the predicted values $\hat{y}^{(i)}$ for average model i are given by

$$\hat{y}^{(i)} = \mathbf{X}\boldsymbol{\beta}^{(i)}$$

where $\mathbf{X}$ is the design matrix of the data to be scored. Forming averages gives

$$\hat{y}^{(*)} = \frac{1}{N}\sum_{i=1}^{N}\hat{y}^{(i)} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{X}\boldsymbol{\beta}^{(i)} = \mathbf{X}\left(\frac{1}{N}\sum_{i=1}^{N}\beta^{(i)}\right) = \mathbf{X}\beta^{(*)}$$

where for parameter j, $\beta_j^{(*)} = \frac{1}{N}\sum_{i=1}^{N}\beta_j^{(i)}$.

You can see that if a parameter estimate is nonzero for just a few of the sample models, then averaging the estimates for this parameter shrinks this estimate towards zero. It is this shrinkage that ameliorates the bias that a parameter is more likely to be selected if it is above its expected value rather than below it. This reduction in bias often produces improved predictions on new data that you obtain with the average model. However, the average model is not parsimonious since it has nonzero estimates for any parameter that is selected in any sample.

One resampling-based approach for obtaining a parsimonious model is to use the number of times that regressors are selected as an indication of importance and then to fit a new model that uses just the regressors that you deem to be most important. This approach is not without risk. One possible problem is that you might have several regressors that, for purposes of prediction, can be used as surrogates for one another. In this case it is possible that none of these regressors individually appears in a large enough percentage of the sample models to be deemed important, even though every model contains at least one of them. Despite such potential problems, this strategy is often successful. You can implement this approach by using the REFIT option in the MODELAVERAGE statement. By default, the REFIT option performs a second round of model averaging, where a fixed model that consists of the effects that are selected in a least twenty percent of the samples in the initial round of model averaging is used. The average model is obtained by averaging the ordinary least squares estimates obtained for each sample in the refit. Note that the default selection frequency cutoff of twenty percent is merely a heuristic guideline that often produces reasonable models.

Another approach to obtaining a parsimonious average model is to form the average of just the frequently selected models. You can implement this strategy by using the SUBSET option in the MODELAVERAGE statement. However, in situations where there are many irrelevant regressors, it is often the case that most of the selected models are selected just once. In such situations, having a way to order the models that are selected with the same frequency is desirable. The following section discusses a way to do this.

Model Selection Frequencies and Frequency Scores

The model frequency score orders models by their selection frequency, but also uses effect selection frequencies to order different models that are selected with the same frequency. Let $\mathbf{x}_i$ denote the ith effect and let f_i denote the selection fraction for this effect. f_i is computed as the number of samples whose selected model contains effect $\mathbf{x}_i$ divided by the number of samples. Suppose the jth model that consists of the K effects $\mathbf{x}_{j_1}, \mathbf{x}_{j_2}, \ldots, \mathbf{x}_{j_K}$ is selected m_j times. Then the model *frequency score*, s_j, for this model is computed as the sum of the model selection frequency and the average selection fraction for this model; that is,

$$s_j = m_j + \frac{\sum_{k=1}^{K} f_{j_k}}{K}$$

When you use the BEST=b suboption of the SUBSET option in the MODELAVERAGE statement, then the average model is formed from the b models with the largest model frequency scores.

Using Validation and Test Data

When you have sufficient data, you can subdivide your data into three parts called the training, validation, and test data. During the selection process, models are fit on the training data, and the prediction error for the models so obtained is found by using the validation data. This prediction error on the validation data can be used to decide when to terminate the selection process or to decide what effects to include as the selection process proceeds. Finally, once a selected model has been obtained, the test set can be used to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases you might want to use only training and test data. For example, you might decide to use an information criterion to decide what effects to include and when to terminate the selection process. In this case no validation data are required, but test data can still be useful in assessing the predictive performance of the selected model. In other cases you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) note that it is difficult to give a general rule on how many observations you should assign to each role. They note that a typical split might be 50% for training and 25% each for validation and testing.

PROC GLMSELECT provides several methods for partitioning data into training, validation, and test data. You can provide data for each role in separate data sets that you specify with the DATA=, TESTDATA=, and VALDATA= options in the PROC GLMSELECT procedure. An alternative method is to use a PARTITION statement to logically subdivide the DATA= data set into separate roles. You can name the fractions of the data that you want to reserve as test data and validation data. For example, specifying

```
proc glmselect data=inData;
   partition fraction(test=0.25 validate=0.25);
   ...
run;
```

randomly subdivides the inData data set, reserving 50% for training and 25% each for validation and testing.

In some cases you might need to exercise more control over the partitioning of the input data set. You can do this by naming a variable in the input data set as well as a formatted value of that variable that correspond to each role. For example, specifying

```
proc glmselect data=inData;
   partition roleVar=group(test='group 1' train='group 2')
   ...
run;
```

assigns all roles observations in the inData data set based on the value of the variable named group in that data set. Observations where the value of group is 'group 1' are assigned for testing, and those with value 'group 2' are assigned to training. All other observations are ignored.

You can also combine the use of the PARTITION statement with named data sets for specifying data roles. For example,

```
proc glmselect data=inData testData=inTest;
   partition fraction(validate=0.4);
   ...
run;
```

reserves 40% of the inData data set for validation and uses the remaining 60% for training. Data for testing is supplied in the inTest data set. Note that in this case, because you have supplied a TESTDATA= data set, you cannot reserve additional observations for testing with the PARTITION statement.

When you use a PARTITION statement, the output data set created with an OUTPUT statement contains a character variable _ROLE_ whose values 'TRAIN', 'TEST', and 'VALIDATE' indicate the role of each observation. _ROLE_ is blank for observations that were not assigned to any of these three roles. When the input data set specified in the DATA= option in the PROC GLMSELECT statement contains an _ROLE_ variable and no PARTITION statement is used, and TESTDATA= and VALDATA= are not specified, then the _ROLE_ variable is used to define the roles of each observation. This is useful when you want to rerun PROC GLMSELECT but use the same data partitioning as in a previous PROC GLMSELECT step. For example, the following statements use the same data for testing and training in both PROC GLMSELECT steps:

```
proc glmselect data=inData;
   partition fraction(test=0.5);
   model y=x1-x10/selection=forward;
   output out=outDataForward;
run;

proc glmselect data=outDataForward;
   model y=x1-x10/selection=backward;
run;
```

When you have reserved observations for training, validation, and testing, a model fit on the training data is scored on the validation and test data, and the average squared error, denoted by ASE, is computed separately for each of these subsets. The ASE for each data role is the error sum of squares for observations in that role divided by the number of observations in that role.

Using the Validation ASE as the STOP= Criterion

If you have provided observations for validation, then you can specify STOP=VALIDATE as a suboption of the SELECTION= option in the MODEL statement. At step k of the selection process, the best candidate effect to enter or leave the current model is determined. Note that here "best candidate" means the effect that gives the best value of the SELECT= criterion that need not be based on the validation data. The validation ASE for the model with this candidate effect added is computed. If this validation ASE is greater than the validation ASE for the model at step k, then the selection process terminates at step k.

Using the Validation ASE as the CHOOSE= Criterion

When you specify the CHOOSE=VALIDATE suboption of the SELECTION= option in the MODEL statement, the validation ASE is computed for the models at each step of the selection process. The model at the first step yielding the smallest validation ASE is selected.

Using the Validation ASE as the SELECT= Criterion

You request the validation ASE as the selection criterion by specifying the SELECT=VALIDATE suboption of the SELECTION= option in the MODEL statement. At step k of the selection process, the validation ASE is computed for each model where a candidate for entry is added or candidate for removal is dropped. The selected candidate for entry or removal is the one that yields a model with the minimal validation ASE.

Cross Validation

Deciding when to stop a selection method is a crucial issue in performing effect selection. Predictive performance of candidate models on data not used in fitting the model is one approach supported by PROC GLMSELECT for addressing this problem (see the section "Using Validation and Test Data" on page 4478). However, in some cases, you might not have sufficient data to create a sizable training set and a validation set that represent the predictive population well. In these cases, cross validation is an attractive alternative for estimating prediction error.

In k-fold cross validation, the data are split into k roughly equal-sized parts. One of these parts is held out for validation, and the model is fit on the remaining $k-1$ parts. This fitted model is used to compute the predicted residual sum of squares on the omitted part, and this process is repeated for each of k parts. The sum of the k predicted residual sum of squares so obtained is the estimate of the prediction error that is denoted by CVPRESS. Note that computing the CVPRESS statistic for k-fold cross validation requires fitting k different models, and so the work and memory requirements increase linearly with the number of cross validation folds.

You can use the CVMETHOD= option in the MODEL statement to specify the method for splitting the data into k parts. CVMETHOD=BLOCK(k) requests that the k parts be made of blocks of floor(n/k) or floor$(n/k)+1$ successive observations, where n is the number of observations. CVMETHOD=SPLIT(k) requests that parts consist of observations $\{1, k+1, 2k+1, 3k+1, \ldots\}$, $\{2, k+2, 2k+2, 3k+2, \ldots\}$, . . . , $\{k, 2k, 3k, \ldots\}$. CVMETHOD=RANDOM(k) partitions the data into random subsets each with roughly floor(n/k) observations. Finally, you can use the formatted value of an input data set variable to define the parts by specifying CVMETHOD=*variable*. This last partitioning method is useful in cases where you need to exercise extra control over how the data are partitioned by taking into account factors such as important but rare observations that you want to "spread out" across the various parts.

You can request details of the CVPRESS computations by specifying the CVDETAILS= option in the MODEL statement. When you use cross validation, the output data set created with an OUTPUT statement contains an integer-valued variable, _CVINDEX_, whose values indicate the subset to which an observation is assigned.

The widely used special case of n-fold cross validation when you have n observations is known as *leave-one-out* cross validation. In this case, each omitted part consists of one observation, and CVPRESS statistic can be efficiently obtained without refitting the model n times. In this case, the CVPRESS statistic is denoted simply by PRESS and is given by

$$\text{PRESS} = \sum_{i=1}^{n} \left(\frac{r_i}{1-h_i} \right)^2$$

where r_i is the residual and h_i is the leverage of the ith observation. You can request *leave-one-out* cross validation by specifying PRESS instead of CV with the options SELECT=, CHOOSE=, and STOP= in the MODEL statement. For example, if the number of observations in the data set is 100, then the following two PROC GLMSELECT steps are mathematically equivalent, but the second step is computed much more efficiently:

```
proc glmselect;
   model y=x1-x10/selection=forward(stop=CV) cvMethod=split(100);
run;
```

```
proc glmselect;
   model y=x1-x10/selection=forward(stop=PRESS);
run;
```

Hastie, Tibshirani, and Friedman (2001) include a discussion about choosing the cross validation fold. They note that as an estimator of true prediction error, cross validation tends to have decreasing bias but increasing variance as the number of folds increases. They recommend five- or tenfold cross validation as a good compromise. By default, PROC GLMSELECT uses CVMETHOD=RANDOM(5) for cross validation.

Using Cross Validation as the STOP= Criterion

You request cross validation as the stopping criterion by specifying the STOP=CV suboption of the SELECTION= option in the MODEL statement. At step k of the selection process, the best candidate effect to enter or leave the current model is determined. Note that here "best candidate" means the effect that gives the best value of the SELECT= criterion that need not be the CV criterion. The CVPRESS score for the model with this candidate effect added or removed is determined. If this CVPRESS score is greater than the CVPRESS score for the model at step k, then the selection process terminates at step k.

Using Cross Validation as the CHOOSE= Criterion

When you specify the CHOOSE=CV suboption of the SELECTION= option in the MODEL statement, the CVPRESS score is computed for the models at each step of the selection process. The model at the first step that yields the smallest CVPRESS score is selected.

Using Cross Validation as the SELECT= Criterion

You request cross validation as the selection criterion by specifying the SELECT=CV suboption of the SELECTION= option in the MODEL statement. At step k of the selection process, the CVPRESS score is computed for each model in which a candidate for entry is added or a candidate for removal is dropped. The selected candidate for entry or removal is the one that yields a model that has the minimal CVPRESS score. At each step of the selection process, this requires forming the CVPRESS statistic for all possible candidate models at the next step. Because forming the CVPRESS statistic for k-fold requires fitting k models, using cross validation as the selection criterion is computationally very demanding compared to using other selection criteria.

External Cross Validation

The method of cross validation that is discussed in the previous section judges models by their performance with respect to ordinary least squares. An alternative to ordinary least squares is to use the penalized regression that is defined by the LASSO or elastic net method. This method is called *external cross validation*, and you can specify external cross validation with CHOOSE=CVEX, new in SAS/STAT 13.1. CHOOSE=CVEX applies only when SELECTION=LASSO or SELECTION=ELASTICNET.

To understand how k-fold external cross validation works, first recall how k-fold cross validation works, as shown in Figure 56.18. The first column in this figure illustrates dividing the training samples into four folds, the second column illustrates the same training samples with reduced numbers of variables at a given step, and the third column illustrates applying ordinary least squares to compute the CVPRESS statistic. For the SELECTION=LASSO and SELECTION=ELASTICNET options, the CVPRESS statistic that is computed

by k-fold cross validation uses an ordinary least squares fit, and hence it does not directly depend on the coefficients obtained by the penalized least squares regression.

Figure 56.18 Applying k-fold Cross Validation to Computing the CVPRESS Statistic

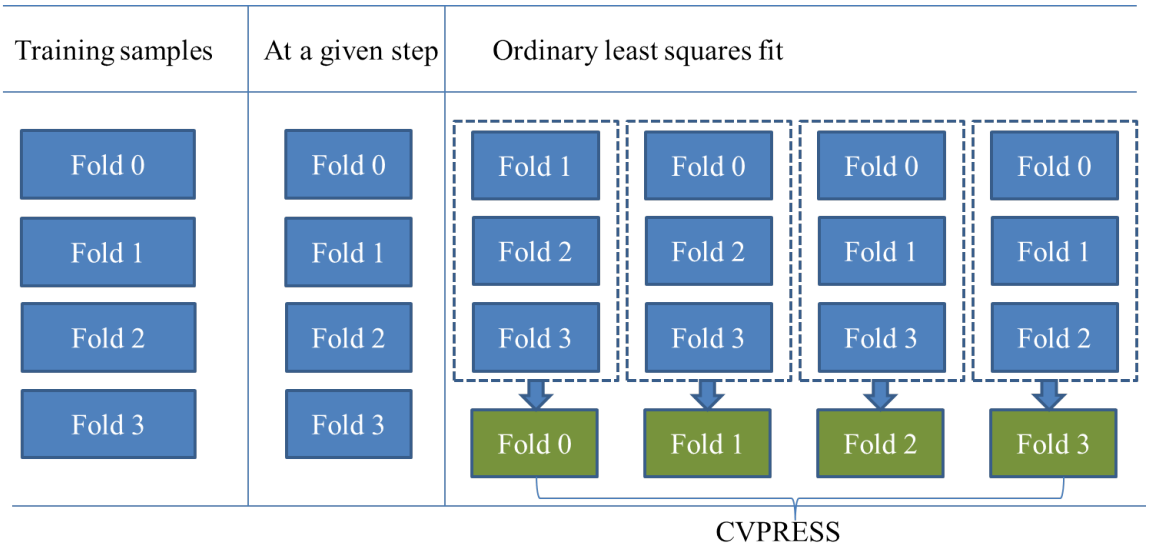

If you want a statistic that is directly based on the coefficients obtained by a penalized least squares regression, you can specify CHOOSE=CVEX to use k-fold external cross validation. External cross validation directly applies the coefficients obtained by a penalized least squares regression to computing the predicted residual sum of squares. Figure 56.19 depicts k-fold external cross validation.

Figure 56.19 k-fold External Cross Validation

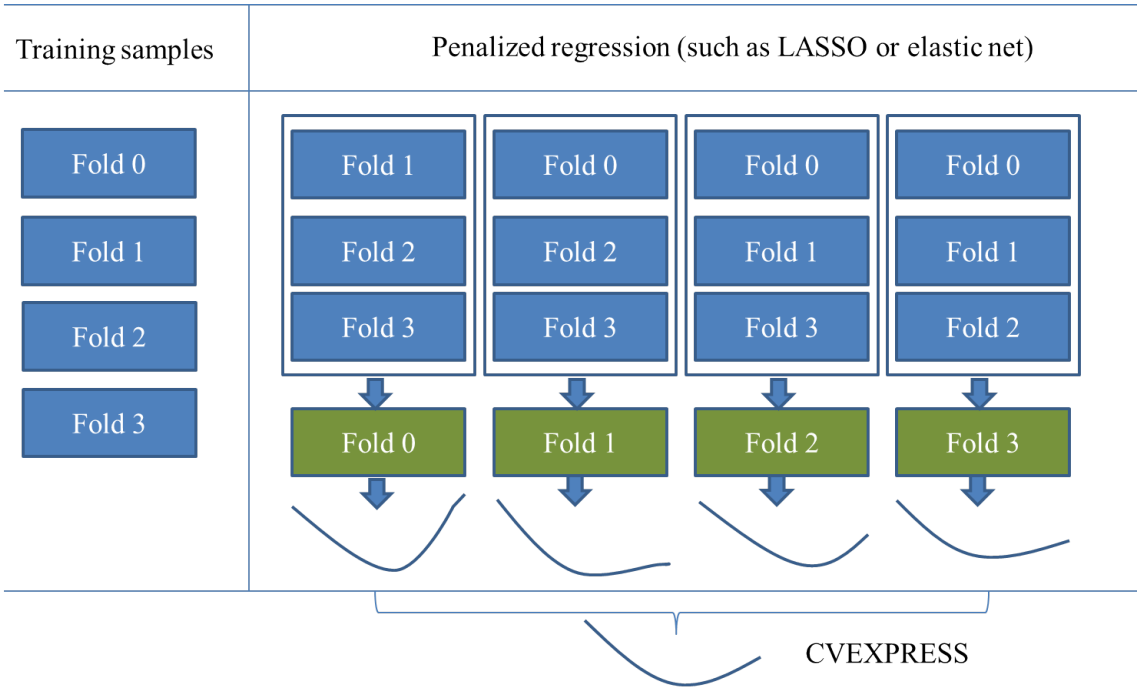

In k-fold external cross validation, the data are split into k approximately equal-sized parts, as illustrated in the first column of Figure 56.19. One of these parts is held out for validation, and the model is fit on the remaining $k-1$ parts by the LASSO method or the elastic net method. This fitted model is used to compute the predicted residual sum of squares on the omitted part, and this process is repeated for each of the k parts. More specifically, for the ith model fit ($i = 1, 2, \ldots, k$), let $\mathbf{X}_{-i}$ denote the held-out part of the matrix of covariates and $\mathbf{y}_{-i}$ denote the corresponding response, and let $\mathbf{X}_i$ denote the remaining $k-1$ parts of the matrix of covariates and $\mathbf{y}_i$ denote the corresponding response. The LASSO method is applied on $\mathbf{X}_i$ and $\mathbf{y}_i$ to solve the optimization problem

$$\min ||\mathbf{y}_i - \mathbf{X}_i \boldsymbol{\beta}||^2 + \lambda_1 \sum_{j=1}^{m} |\beta_j|$$

Note that, as discussed in the section "Elastic Net Selection (ELASTICNET)" on page 4465, the elastic net method can be solved in the same way as LASSO, by augmenting the design matrix $\mathbf{X}_i$ and the response $\mathbf{y}_i$.

Following the piecewise linear solution path of LASSO, the coefficients

$$\boldsymbol{\beta}^{i1}, \boldsymbol{\beta}^{i2}, \ldots, \boldsymbol{\beta}^{ip}, \ldots$$

are computed to correspond to the LASSO regularization parameter

$$\lambda_1^{i1}, \lambda_1^{i2}, \ldots, \lambda_1^{ip}, \ldots$$

Based on the computed coefficients, the predicted residual sum of squares is computed on the held-out part $\mathbf{X}_{-i}$ and $\mathbf{y}_{-i}$ as

$$r^{ip} = ||\mathbf{y}_{-i} - \mathbf{X}_{-i} \boldsymbol{\beta}^{ip}||^2$$

The preceding process can be summarized as

$$(\mathbf{X}_i, \mathbf{y}_i) \rightarrow (\lambda_1^{ip}, \boldsymbol{\beta}^{ip}) \rightarrow (\lambda_1^{ip}, r^{ip}), i = 1, 2, \ldots, k, p = 1, 2, \ldots$$

For the illustration in Figure 56.19, the results (λ_1^{ip}, r^{ip}) correspond to four curves. The knots $\lambda_1^{i1}, \lambda_1^{i2}, \ldots, \lambda_1^{ip}, \ldots$ are usually different among different model fits $i = 1, 2, \ldots, k$. To merge the results of the k model fits for computing the CVEXPRESS statistic, perform the following three steps:

1. Identify distinct knots among all $\lambda_1^{ip}, i = 1, 2, \ldots, k, p = 1, 2, \ldots$
2. Apply interpolation to compute the predicted residual sum of squares at all the knots. Here the interpolation is done in a closed quadratic function by using the piecewise linear solutions of LASSO.
3. Add the predicted residual sum of squares of k model fits according to the identified knots, and the sum of the k predicted residual sum of squares so obtained is the estimate of the prediction error that is denoted by CVEXPRESS.

The bottom curve in Figure 56.19 illustrates the curve between the CVEXPRESS statistics and the value of λ_1 (L1). Note that computing the CVEXPRESS statistic for k-fold external cross validation requires fitting k different LASSO or elastic net models, and so the work and memory requirements increase linearly with the number of cross validation folds.

In addition to characterizing the piecewise linear solutions of the coefficients $\boldsymbol{\beta}^{i1}, \boldsymbol{\beta}^{i2}, \ldots, \boldsymbol{\beta}^{ip}, \ldots$ by the LASSO regularization parameters $\lambda_1^{i1}, \lambda_1^{i2}, \ldots, \lambda_1^{ip}, \ldots$, you can also characterize the solutions by the sum

of the absolute values of the coefficients or the scaled regularization parameter. For a detailed discussion of the different options, see the L1CHOICE= option in the MODEL statement.

Like k-fold cross validation, you can use the CVMETHOD= option in the MODEL statement to specify the method for splitting the data into k parts in k-fold external cross validation. CVMETHOD=BLOCK(k) requests that the k parts be made of blocks of floor(n/k) or floor(n/k) + 1 successive observations, where n is the number of observations. CVMETHOD=SPLIT(k) requests that parts consist of observations $\{1, k+1, 2k+1, 3k+1, \ldots\}, \{2, k+2, 2k+2, 3k+2, \ldots\}, \ldots, \{k, 2k, 3k, \ldots\}$. CVMETHOD=RANDOM($k$) partitions the data into random subsets, each with approximately floor(n/k) observations. Finally, you can use the formatted value of an input data set variable to define the parts by specifying CVMETHOD=*variable*. This last partitioning method is useful in cases where you need to exercise extra control over how the data are partitioned by taking into account factors such as important but rare observations that you want to "spread out" across the various parts. By default, PROC GLMSELECT uses CVMETHOD=RANDOM(5) for external cross validation.

For the elastic net method, if the ridge regression parameter λ_2 is not specified by the L2= option and you use k-fold external cross validation for the CHOOSE= option, then the optimal λ_2 is searched over an interval (see Figure 56.12 for an illustration) and it is set to the value that achieves the minimum CVEXPRESS statistic. You can use the L2SEARCH=, L2LOW=, L2HIGH=, and L2STEPS= options to control the search of λ_2 (L2).

Difference between Cross Validation and External Cross Validation

If you specify SELECTION=LASSO or SELECTION=ELASTICNET, the penalized model is fit only once using the same training samples in k-fold cross validation, whereas the penalized model is fit k times by using different training samples in k-fold external cross validation. External cross validation also requires identifying the knots that result from the different solution paths. The CVPRESS statistic that is computed in k-fold cross validation is based on ordinary least squares regression, whereas the CVEXPRESS statistic that is computed in k-fold external cross validation is based on the penalized regression.

Using External Cross Validation as the CHOOSE= Criterion

When you specify the CHOOSE=CVEX suboption of the SELECTION= option in the MODEL statement, the CVEXPRESS statistics are computed for the models at each step of the selection process. The model at the first step that has the smallest CVEXPRESS score is selected.

Screening

Model selection from a very large number of effects is computationally demanding. For example, in analyzing microarray data (where each dot in the array corresponds to a regressor), it is not uncommon to have 35,000 such regressors. Large regression problems also arise when you want to consider all possible interactions of your main effects as candidates for inclusion in a selected model. For an example that uses this approach to build a predictive model for bankruptcy, see Foster and Stine (2004).

In recent years, there has been a resurgence of interest in combining variable selection methods with a screening approach that reduces the large number of regressors to a much smaller subset from which the model selection is performed. There are two categories of screening methods:

- safe screening methods, by which the resulting solution is exactly the same as the solution when no screening is performed
- heuristic screening methods, by which the resulting solution is not necessarily the same as the solution when no screening is performed

The heuristic screening approaches are usually much faster than the safe screening methods, but they are not guaranteed to reproduce the true LASSO or elastic net solution.

Safe Screening via SCREEN=SASVI

The safe screening approaches are developed mainly for the LASSO method and its extensions, which solve a well-defined convex optimization problem. For these methods, safe screening works as follows:

1. Given a solution $\boldsymbol{\beta}_1$ that corresponds to a regularization parameter λ_1, safe screening approaches aim to identify the effects that are guaranteed to have zero coefficients in the solution $\boldsymbol{\beta}_2$, which corresponds to the regularization parameter λ_2 ($0 < \lambda_2 < \lambda_1$).
2. The computation of $\boldsymbol{\beta}_2$ can exclude such inactive effects, thus saving computation cost.

The idea of safe screening was pioneered by El Ghaoui, Viallon, and Rabbani (2012), and improved subsequently by other researchers (Liu et al. 2014; Wang et al. 2013; Xiang, Xu, and Ramadge 2011).

If you specify SCREEN=SASVI in the model statement, PROC GLMSELECT uses the SASVI technique of Liu et al. (2014) to speed up LAR-type LASSO. The computation cost can usually be reduced while the solution is the same when you specify SCREEN=NONE.

Heuristic Screening via SCREEN=SIS

Heuristic screening approaches (Fan and Lv 2008; Tibshirani et al. 2012) use a screening statistic that is inexpensive to compute in order to eliminate regressors that are unlikely to be selected. For linear regression, you can use the magnitude of the correlation between each individual regressor and the response as such a screening statistic. The square of the correlation between a regressor and the response is the R-square value for the univariate regression of the response on this regressor. Hence, screening by the magnitude of the pairwise correlations is equivalent to fitting univariate models to do the screening.

The SIS (sure independence screening) approach proposed by Fan and Lv (2008) is a well-known heuristic screening approach that applies to model selection methods such as forward selection, backward selection, LASSO, and so on. When you specify SCREEN=SIS in the MODEL statement, PROC GLMSELECT first chooses only the subset of regressors whose screening statistics are among a specified number or percentage of the largest screening statistic values. When you specify SCREEN=SIS, PROC GLMSELECT uses the screening statistic that is the magnitude of the correlation between each individual regressor and the response. Then it performs model selection for the response from this screened subset of the original regressors.

Displayed Output

The following sections describe the displayed output produced by PROC GLMSELECT. The output is organized into various tables, which are discussed in order of appearance. Note that the contents of a table might change depending on the options that you specify.

Model Information

The "Model Information" table displays basic information about the data sets and the settings used to control effect selection. These settings include the following:

- the selection method
- the criteria used to select effects, stop the selection, and choose the selected model
- the effect hierarchy enforced

The ODS name of the "Model Information" table is ModelInfo.

Performance Settings

The "Performance Settings" table displays settings that affect performance. These settings include whether threading is enabled and the number of CPUs available as well as the method used to build the crossproduct matrices. This table is displayed only if you specify the DETAILS option in the PERFORMANCE statement. The ODS name of the "Performance Settings" table is PerfSettings.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used. If you use a PARTITION statement, the table also displays the number of observations used for each data role. If you specify TESTDATA= or VALDATA= data sets in the PROC GLMSELECT statement, then "Number of Observations" tables are also produced for these data sets. The ODS name of the "Number of Observations" table is NObs.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. The ODS name of the "Class Level Information" table is ClassLevelInfo.

Class Level Coding

The "Class Level Coding" table shows the coding used for variables specified in the CLASS statement. The ODS name of the "Class Level Coding" table is ClassLevelCoding.

Dimensions

The "Dimensions" table displays information about the number of effects and the number of parameters from which the selected model is chosen. If you use split classification variables, then this table also includes the number of effects after splitting is taken into account. The ODS name of the "Dimensions" table is Dimensions.

Candidates

The "Candidates" table displays the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. The effects are displayed in sorted order from best to worst of the selection criterion. You request this table with the DETAILS= option in the MODEL statement. The ODS name of the "Candidates" table is Candidates.

Selection Summary

The "Selection Summary" table displays details about the sequence of steps of the selection process. For each step, the effect that was entered or dropped is displayed along with the statistics used to select the effect, stop the selection, and choose the selected model. You can request that additional statistics be displayed with the STATS= option in the MODEL statement. For all criteria that you can use for model selection, the steps at which the optimal values of these criteria occur are also indicated. The ODS name of the "Selection Summary" table is SelectionSummary.

Stop Reason

The "Stop Reason" table displays the reason why the selection stopped. To facilitate programmatic use of this table, an integer code is assigned to each reason and is included if you output this table by using an ODS OUTPUT statement. The reasons and their associated codes follow:

Code	Stop Reason
1	maximum number of steps done
2	specified number of steps done
3	specified number of effects in model
4	stopping criterion at local optimum
5	model is an exact fit
6	all entering effects are linearly dependent on those in the model
7	all effects are in the model
8	all effects have been dropped
9	requested full least squares fit completed
10	stepwise selection is cycling
11	dropping any effect does not improve the selection criterion
12	no effects are significant at the specified SLE or SLS levels
13	adding or dropping any effect does not improve the selection criterion
14	all remaining effects are required

The ODS name of the "Stop Reason" table is StopReason.

Stop Details

The "Stop Details" table compares the optimal value of the stopping criterion at the final model with how it would change if the best candidate effect were to enter or leave the model. The ODS name of the "Stop Details" table is StopDetails.

Selected Effects

The "Selected Effects" table displays a string containing the list of effects in the selected model. The ODS name of the "Selected Effects" table is SelectedEffects.

ANOVA

The "ANOVA" table displays an analysis of variance for the selected model. This table includes the following:

- the Source of the variation, Model for the fitted regression, Error for the residual error, and C Total for the total variation after correcting for the mean. The Uncorrected Total Variation is produced when the NOINT option is used.
- the degrees of freedom (DF) associated with the source
- the Sum of Squares for the term
- the Mean Square, the sum of squares divided by the degrees of freedom
- the F Value for testing the hypothesis that all parameters are zero except for the intercept. This is formed by dividing the mean square for Model by the mean square for Error.
- the Prob>F, the probability of getting a greater F statistic than that observed if the hypothesis is true. Note that these p-values are displayed only if you specify the "SHOWPVALUES" option in the MODEL statement. These p-values are generally liberal because they are not adjusted for the fact that the terms in the model have been selected.

You can request "ANOVA" tables for the models at each step of the selection process with the DETAILS= option in the MODEL statement. The ODS name of the "ANOVA" table is ANOVA.

Fit Statistics

The "Fit Statistics" table displays fit statistics for the selected model. The statistics displayed include the following:

- Root MSE, an estimate of the standard deviation of the error term. It is calculated as the square root of the mean square error.
- Dep Mean, the sample mean of the dependent variable
- R-square, a measure between 0 and 1 that indicates the portion of the (corrected) total variation attributed to the fit rather than left to residual error. It is calculated as SS(Model) divided by SS(Total). It is also called the *coefficient of determination*. It is the square of the multiple correlation—in other words, the square of the correlation between the dependent variable and the predicted values.

- Adj R-Sq, the adjusted R^2, a version of R^2 that has been adjusted for degrees of freedom. It is calculated as

$$\bar{R}^2 = 1 - \frac{(n-i)(1-R^2)}{n-p}$$

  where i is equal to 1 if there is an intercept and 0 otherwise, n is the number of observations used to fit the model, and p is the number of parameters in the model.

- fit criteria AIC, AICC, BIC, CP, and PRESS if they are used in the selection process or are requested with the STATS= option. See the section "Criteria Used in Model Selection Methods" on page 4469 for details and Table 56.10 for the formulas for evaluating these criteria.

- the CVPRESS statistic when cross validation is used in the selection process. See the section "Cross Validation" on page 4480 for details.

- the average square errors (ASE) on the training, validation, and test data. See the section "Using Validation and Test Data" on page 4478 for details.

You can request "Fit Statistics" tables for the models at each step of the selection process with the DETAILS= option in the MODEL statement. The ODS name of the "Fit Statistics" table is FitStatistics.

Cross Validation Details

The "Cross Validation Details" table displays the following:

- the fold number
- the number of observations used for fitting
- the number of observations omitted
- the predicted residual sum of squares on the omitted observations

You can request this table with the CVDETAILS= option in the MODEL statement whenever cross validation is used in the selection process. This table is displayed for the selected model, but you can request this table at each step of the selection process by using the DETAILS= option in the MODEL statement. The ODS name of the "Cross Validation Details" table is CVDetails.

Parameter Estimates

The "Parameter Estimates" table displays the parameters in the selected model and their estimates. The information displayed for each parameter in the selected model includes the following:

- the parameter label that includes the effect name and level information for effects containing classification variables
- the degrees of freedom (DF) for the parameter. There is one degree of freedom unless the model is not full rank.

- the parameter estimate
- the standard error, which is the estimate of the standard deviation of the parameter estimate
- T for H0: Parameter=0, the *t* test that the parameter is zero. This is computed as the parameter estimate divided by the standard error.
- the Prob > |T|, the probability that a *t* statistic would obtain a greater absolute value than that observed given that the true parameter is zero. This is the two-tailed significance probability. Note that these *p*-values are displayed only if you specify the SHOWPVALUES option in the MODEL statement. These *p*-values are generally liberal because they are not adjusted for the fact that the terms in the model have been selected.

If cross validation is used in the selection process, then you can request that estimates of the parameters for each cross validation fold be included in the "Parameter Estimates" table by using the CVDETAILS= option in the MODEL statement. You can request "Parameter Estimates" tables for the models at each step of the selection process with the DETAILS= option in the MODEL statement. The ODS name of the "Parameter Estimates" table is ParameterEstimates.

Score Information

For each SCORE statement, the "Score Information" table displays the names of the score input and output data sets, and the number of observations that were read and successfully scored. The ODS name of the "Score Information" table is ScoreInfo.

Timing Breakdown

The "Timing Breakdown" table displays a broad breakdown of where time was spent in the PROC GLMSELECT step. This table is displayed only if you specify the DETAILS option in the PERFORMANCE statement. The ODS name of the "Timing Breakdown" table is Timing.

ODS Table Names

PROC GLMSELECT assigns a name to each table it creates. You can use these names to reference the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 56.11.

For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 56.11 ODS Tables Produced by PROC GLMSELECT

ODS Table Name	Description	Statement	Option
ANOVA	Selected model ANOVA table	MODEL	Default
AvgParmEst	Average parameter estimates	MODELAVERAGE	Default
BSplineDetails	B-spline basis details	EFFECT	DETAILS
Candidates	Entry/removal effect ranking	MODEL	DETAILS=
ClassLevelCoding	Classification variable coding	CLASS	SHOWCODING
ClassLevelInfo	Classification variable levels	CLASS	Default

Table 56.11 *continued*

ODS Table Name	Description	Statement	Option
CollectionLevelInfo	Levels of collection effects	EFFECT	DETAILS
CVDetails	Cross validation PRESS by fold	MODEL	CVDETAILS=
Dimensions	Number of effects and parameters	MODEL	Default
EffectSelectPct	Effect selection percentages	MODELAVERAGE	Default
FitStatistics	Selected model fit statistics	MODEL	Default
MMLevelInfo	Levels of multimember effects	EFFECT	DETAILS
ModelAvgInfo	Model averaging information	MODELAVERAGE	Default
ModelInfo	Model information	MODEL	Default
ModelSelectFreq	Model selection frequencies	MODELAVERAGE	Default
NObs	Number of observations	MODEL	Default
ParameterNames	Labels for column names in the design matrix	PROC	OUTDESIGN(names)
ParameterEstimates	Selected model parameter estimates	MODEL	Default
PerfSettings	Performance settings	PERFORMANCE	DETAILS
PolynomialDetails	Polynomial details	EFFECT	DETAILS
PolynomialScaling	Polynomial scaling	EFFECT	DETAILS
RefitAvgParmEst	Refit average parameter estimates	MODELAVERAGE	REFIT
ScoreInfo	Score request information	SCORE	Default
SelectedEffects	List of selected effects	MODEL	Default
SelectionSummary	Selection summary	MODEL	Default
SplineKnots	Knot and boundary knot values	EFFECT	DETAILS
StopDetails	Stopping criterion details	MODEL	Default
StopReason	Reason why selection stopped	MODEL	Default
Timing	Timing details	PERFORMANCE	DETAILS
TPFSplineDetails	Truncated power function spline basis details	EFFECT	DETAILS

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You must also specify the PLOTS= option in the PROC GLMSELECT statement.

The following sections describe the ODS graphical displays produced by PROC GLMSELECT. The examples use the Sashelp.Baseball data set that is described in the section "Getting Started: GLMSELECT Procedure" on page 4416.

ODS Graph Names

PROC GLMSELECT assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 56.12.

Table 56.12 Graphs Produced by PROC GLMSELECT

ODS Graph Name	Plot Description	PLOTS Option
AdjRSqPlot	Adjusted R-square by step	CRITERIA(UNPACK)
AICCPlot	Corrected Akaike's information criterion by step	CRITERIA(UNPACK)
AICPlot	Akaike's information criterion by step	CRITERIA(UNPACK)
ASEPlot	Average square errors by step	ASE
BICPlot	Sawa's Bayesian information criterion by step	CRITERIA(UNPACK)
CandidatesPlot	SELECT= criterion by effect	CANDIDATES
ChooseCriterionPlot	CHOOSE= criterion by step	COEFFICIENTS(UNPACK)
CoefficientPanel	Coefficients and CHOOSE= criterion by step	COEFFICIENTS
CoefficientPlot	Coefficients by step	COEFFICIENTS(UNPACK)
CPPlot	Mallows' C_p by step	CRITERIA(UNPACK)
CriterionPanel	Fit criteria by step	CRITERIA
CVPRESSPlot	Cross validation predicted RSS by step	CRITERIA(UNPACK)
EffectSelectPctPlot	Resampling effect selection percentages	EFFECTSELECTPCT
ParmDistPanel	Resampling parameter estimate distributions	PARMDIST
PRESSPlot	Predicted RSS by step	CRITERIA(UNPACK)
RefitParmDistPanel	Refit resampling parameter estimate distributions	PARMDIST
SBCPlot	Schwarz Bayesian information criterion by step	CRITERIA(UNPACK)
ValidateASEPlot	Average square error on validation data by step	CRITERIA(UNPACK)

Candidates Plot

You request the "Candidates Plot" by specifying the PLOTS=CANDIDATES option in the PROC GLMSELECT statement and the DETAILS=STEPS option in the MODEL statement. This plot shows the values of selection criterion for the candidate effects for entry or removal, sorted from best to worst from left to right across the plot. The leftmost candidate displayed is the effect selected for entry or removal at that step. You can use this plot to see at what steps the decision about which effect to add or drop is clear-cut. See Figure 56.6 for an example.

Coefficient Panel

When you specify the PLOTS=COEFFICIENTS option in the PROC GLMSELECT statement, PROC GLMSELECT produces a panel of two plots showing how the standardized coefficients and the criterion used to choose the final model evolve as the selection progresses. The following statements provide an example:

```
ods graphics on;

proc glmselect data=sashelp.baseball plots=coefficients;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor|yrMajor crAtBat|crAtBat crHits|crHits
                     crHome|crHome crRuns|crRuns crRbi|crRbi
                     crBB|crBB league division nOuts nAssts nError /
                     selection=forward(stop=AICC CHOOSE=SBC);
run;
```

Figure 56.20 shows the requested graphic. The upper plot in the panel displays the standardized coefficients as a function of the step number. You can request standardized coefficients in the parameter estimates tables by specifying the STB option in the MODEL statement, but this option is not required to produce this plot. To help in tracing the changes in a parameter, the standardized coefficients for each parameter are connected by lines. Coefficients corresponding to effects that are not in the selected model at a step are zero and hence not observable. For example, consider the parameter CrAtBat*CrAtBat in Output 56.20. Because CrAtBat*CrAtBat enters the model at step 2, the line that represents this parameter starts rising from zero at step 1 when CrRuns enters the model. Parameters that are nonzero at the final step of the selection are labeled if their magnitudes are greater than 1% of the range of the magnitudes of all the nonzero parameters at this step. To avoid collision, labels corresponding to parameters with similar values at the final step might get suppressed. You can control when this label collision avoidance occurs by using the LABELGAP= suboption of the PLOTS=COEFFICIENTS option.

Figure 56.20 Coefficient Panel

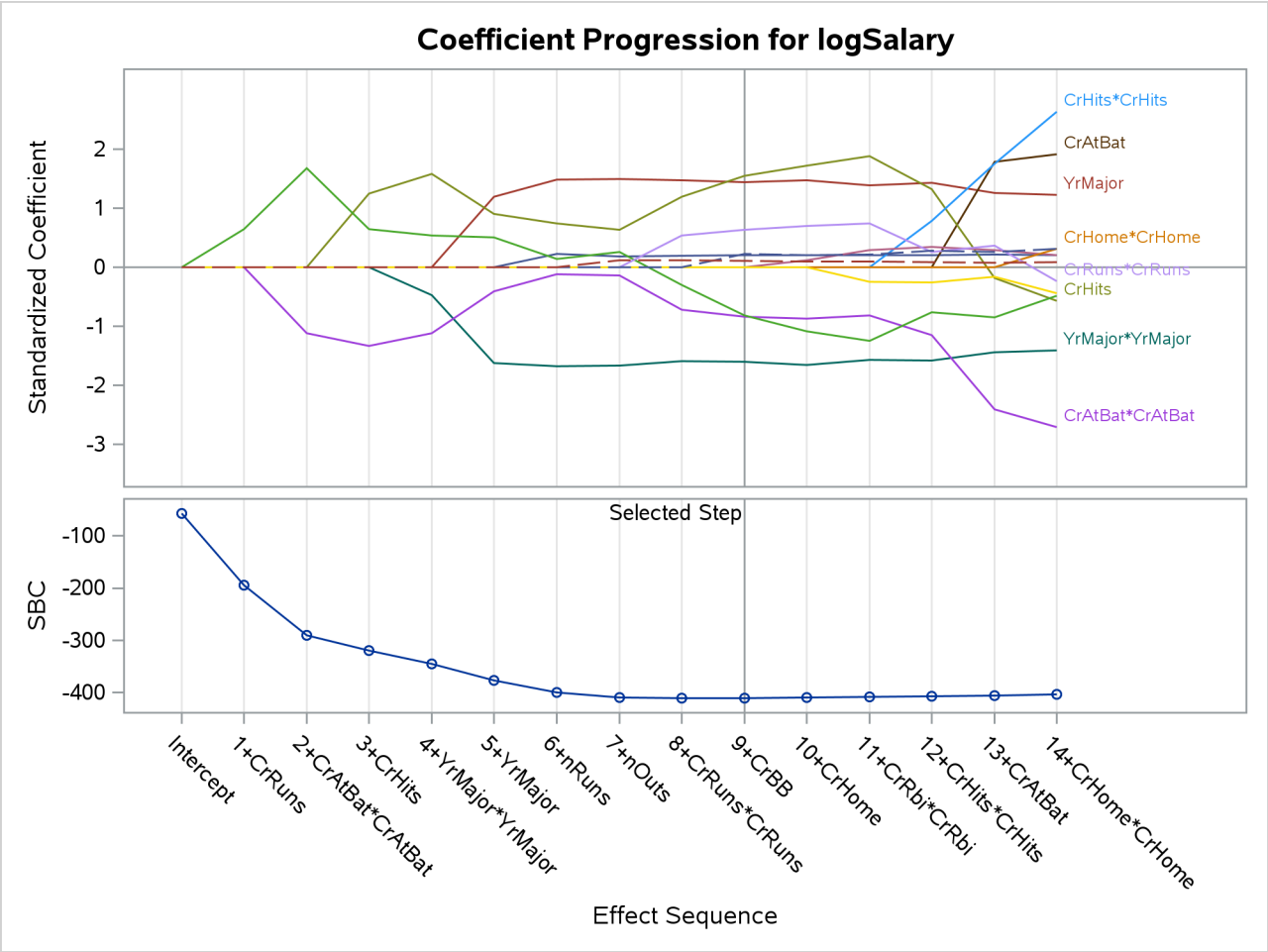

The lower plot in the panel shows how the criterion used to choose among the examined models progresses. The selected step occurs at the optimal value of this criterion. In this example, this criterion is the SBC criterion and it achieves its minimal value at step 9 of the forward selection.

In some cases, particularly when the final step contains a large number of parameters, you might be interested in using this plot only to discern if and when the parameters in the model are essential unchanged beyond a certain step. In such cases, you might want to suppress the labeling of the parameters and use a numeric axis on the horizontal axis of the plot. You can do this using the STEPAXIS= and MAXPARMLABEL= suboptions of the PLOTS=CRITERIA option. The following statements provide an example:

```
proc glmselect data=sashelp.baseball
     plots(unpack maxparmlabel=0 stepaxis=number)=coefficients;

   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor|yrMajor crAtBat|crAtBat crHits|crHits
                     crHome|crHome crRuns|crRuns crRbi|crRbi
                     crBB|crBB league division nOuts nAssts nError /
                     selection=forward(stop=none);
run;
```

The UNPACK = option requests that the plots of the coefficients and CHOOSE= criterion be shown in separate plots. The STEPAXIS=NUMBER option requests a numeric horizontal axis showing step number, and the MAXPAMLABEL=0 option suppresses the labels for the parameters. The "Coefficient Plot" is shown in Figure 56.21. You can see that the standardized coefficients do not vary greatly after step 16.

Figure 56.21 Coefficient Plot

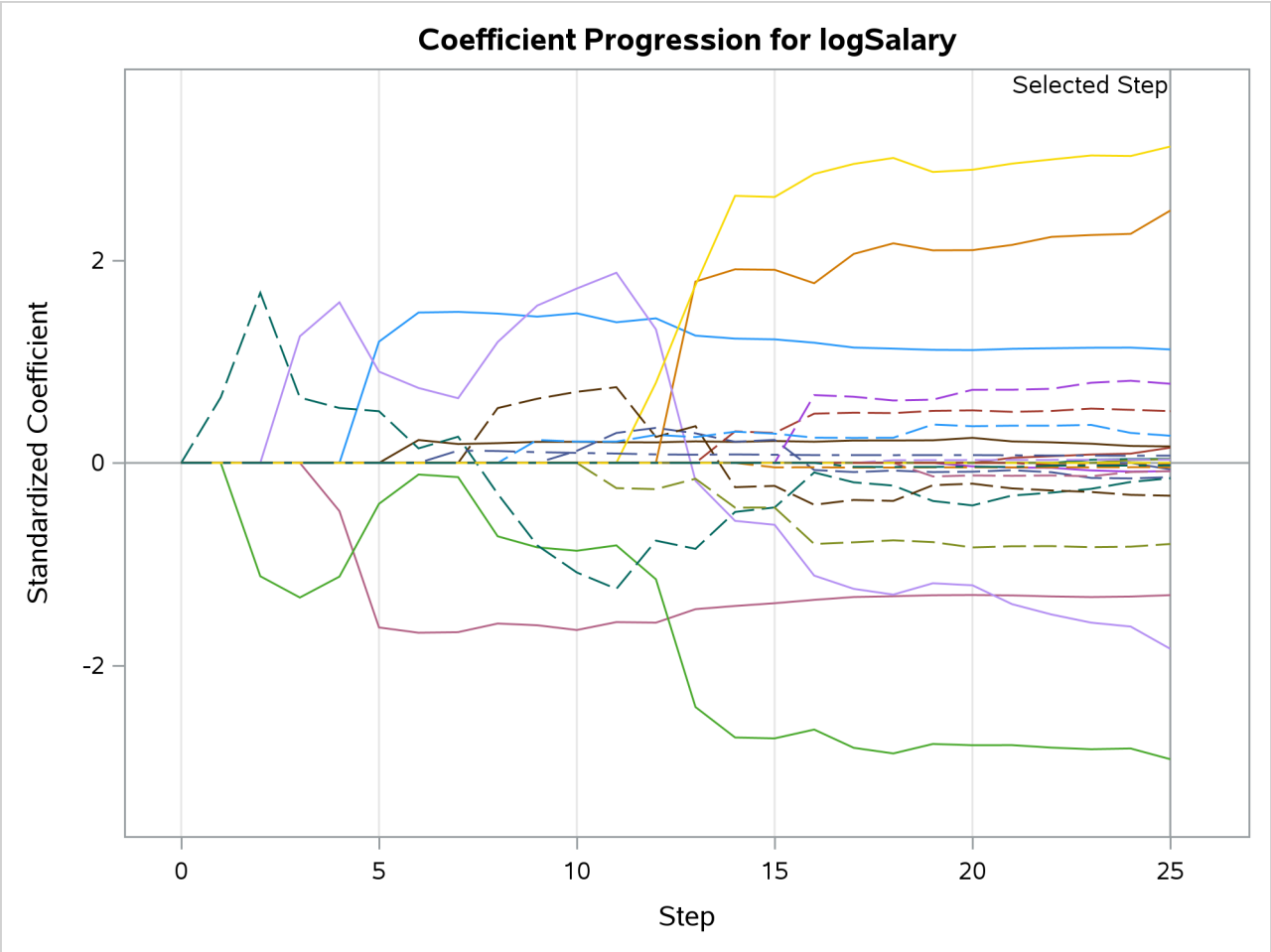

Criterion Panel

You request the criterion panel by specifying the PLOTS=CRITERIA option in the PROC GLMSELECT statement. This panel displays the progression of the ADJRSQ, AIC, AICC, and SBC criteria, as well as any other criteria that are named in the CHOOSE=, SELECT=, STOP=, or STATS= option in the MODEL statement.

The following statements provide an example:

```
proc glmselect data=sashelp.baseball plots=criteria;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor|yrMajor crAtBat|crAtBat crHits|crHits
                     crHome|crHome crRuns|crRuns crRbi|crRbi
```

```
                        crBB|crBB league division nOuts nAssts nError /
                        selection=forward(steps=15 choose=AICC)
                        stats=PRESS;
run;
```

Figure 56.22 shows the requested criterion panel. Note that the PRESS criterion is included in the panel because it is named in the STATS= option in the MODEL statement. The selected step is displayed as a vertical reference line on the plot of each criterion, and the legend indicates which of these criteria is used to make the selection. If the selection terminates for a reason other than optimizing a criterion displayed on this plot, then the legend will not report a reason for the selected step. The optimal value of each criterion is indicated with the "Star" marker. Note that it is possible that a better value of a criterion might have been reached had more steps of the selection process been done.

Figure 56.22 Criterion Panel

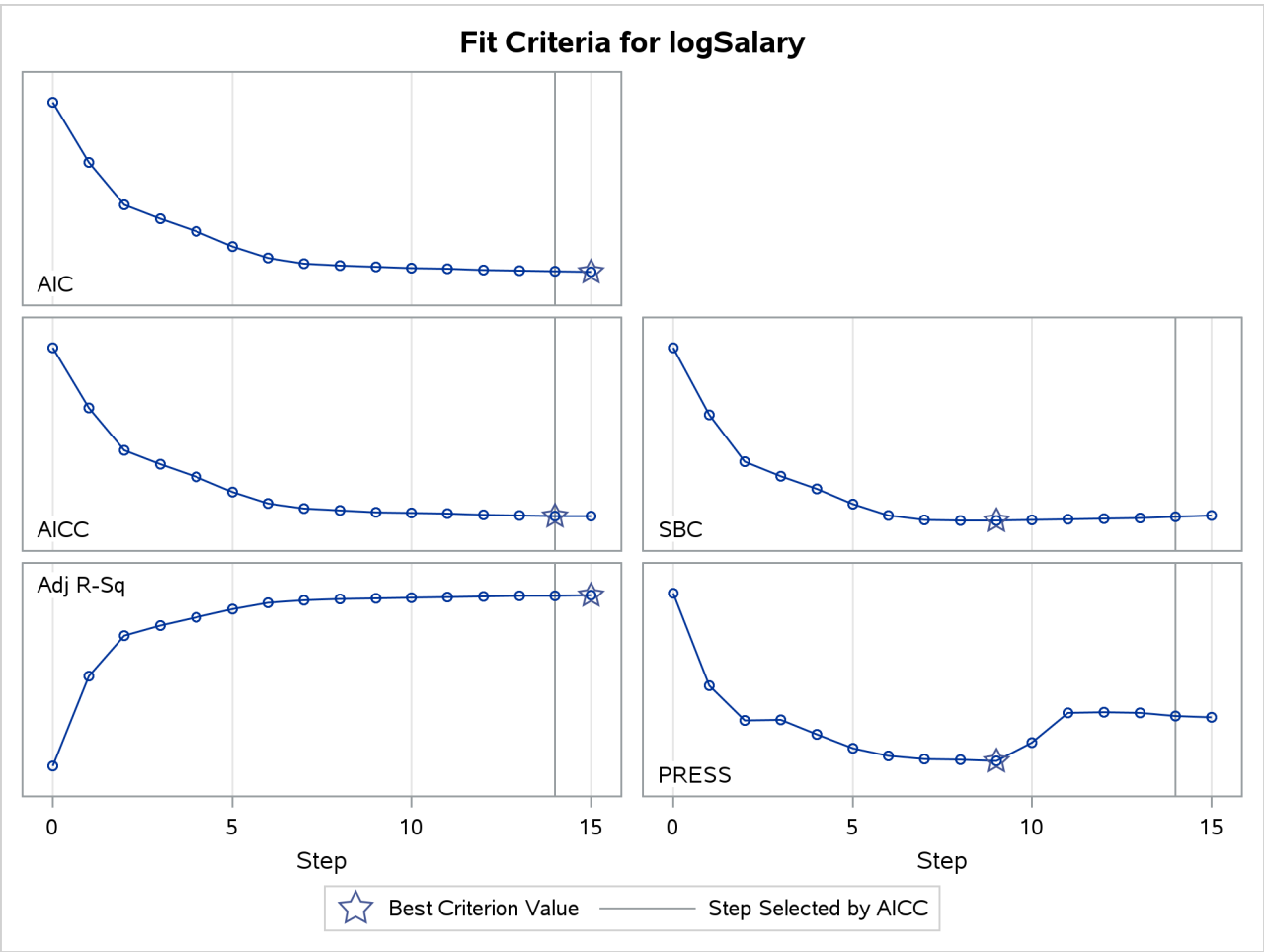

Average Square Error Plot

You request the average square error plot by specifying the PLOTS=ASE option in the PROC GLMSELECT statement. This plot shows the progression of the average square error (ASE) evaluated separately on the training data, and the test and validation data whenever these data are provided with the TESTDATA= and VALDATA= options or are produced by using a PARTITION statement. You use the plot to detect when overfitting the training data occurs. The ASE decreases monotonically on the training data as parameters are added to a model. However, the average square error on test and validation data typically starts increasing when overfitting occurs. See Output 56.1.9 and Output 56.2.6 for examples.

Examining Specific Step Ranges

The coefficient panel, criterion panel, and average square error plot display information for all the steps examined in the selection process. In some cases, you might want to focus attention on just a particular step range. For example, it is hard to discern the variation in the criteria displayed in Figure 56.22 near the selected step because the variation in these criteria in the steps close to the selected step is small relative to the variation across all steps. You can request a range of steps to display using the STARTSTEP= and ENDSTEP= suboptions of the PLOTS= option. You can specify these options as both global and specific plot options, with the specific options taking precedence if both are specified. The following statements provide an example:

```
proc glmselect data=sashelp.baseball plots=criteria(startstep=10 endstep=16);
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor|yrMajor crAtBat|crAtBat crHits|crHits
                     crHome|crHome crRuns|crRuns crRbi|crRbi
                     crBB|crBB league division nOuts nAssts nError /
                     selection=forward(stop=none choose=AICC);
run;

ods graphics off;
```

Figure 56.23 shows the progression of the fit criteria between steps 10 and 16. Note that if the optimal value of a criterion does not occur in this specified step range, then no optimal marker appears for that criterion. The plot of the SBC criterion in Figure 56.23 is one such case.

Figure 56.23 Criterion Panel for Specified Step Range

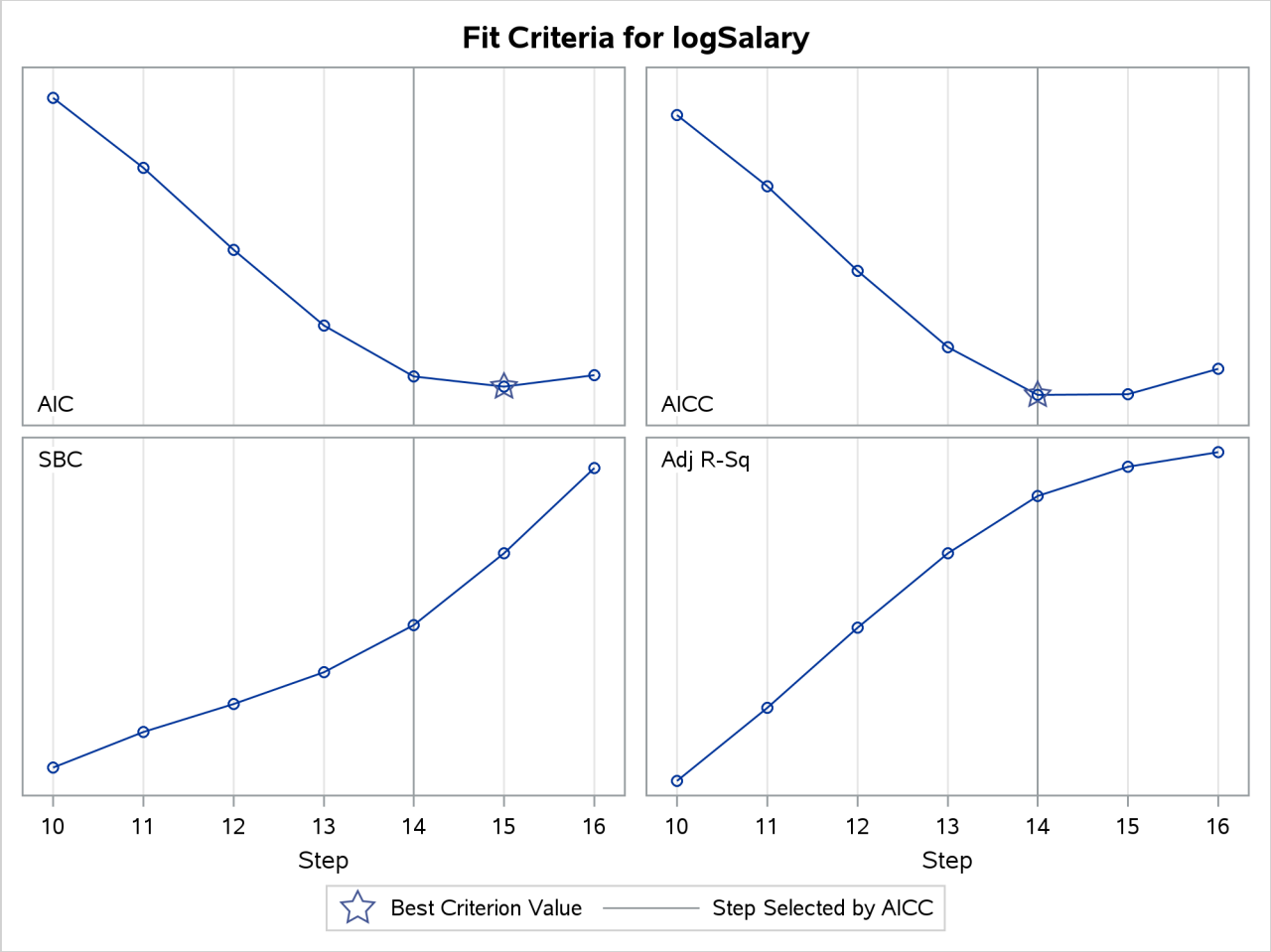

Examples: GLMSELECT Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 56.1: Modeling Baseball Salaries Using Performance Statistics

(View the complete code for this example (glsex1.sas) in the example repository.)

This example continues the investigation of the baseball data set introduced in the section "Getting Started: GLMSELECT Procedure" on page 4416. In that example, the default stepwise selection method based on the SBC criterion was used to select a model. In this example, model selection that uses other information criteria and out-of-sample prediction criteria is explored.

PROC GLMSELECT provides several selection algorithms that you can customize by specifying criteria for selecting effects, stopping the selection process, and choosing a model from the sequence of models at each step. For more details on the criteria available, see the section "Criteria Used in Model Selection Methods" on page 4469. The SELECT=SL suboption of the SELECTION= option in the MODEL statement in the following code requests the traditional hypothesis test-based stepwise selection approach, where effects in the model that are not significant at the stay significance level (SLS) are candidates for removal and effects not yet in the model whose addition is significant at the entry significance level (SLE) are candidates for addition to the model.

```
ods graphics on;

proc glmselect data=sashelp.baseball plot=CriterionPanel;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=stepwise(select=SL) stats=all;
run;
```

The default SLE and SLS values of 0.15 might not be appropriate for these data. One way to investigate alternative ways to stop the selection process is to assess the sequence of models in terms of model fit statistics. The STATS=ALL option in the MODEL statement requests that all model fit statistics for assessing the sequence of models of the selection process be displayed. To help in the interpretation of the selection process, you can use graphics supported by PROC GLMSELECT. ODS Graphics must be enabled before requesting plots. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." With ODS Graphics enabled, the PLOTS=CRITERIONPANEL option in the PROC GLMSELECT statement produces the criterion panel shown in Output 56.1.1.

Output 56.1.1 Criterion Panel

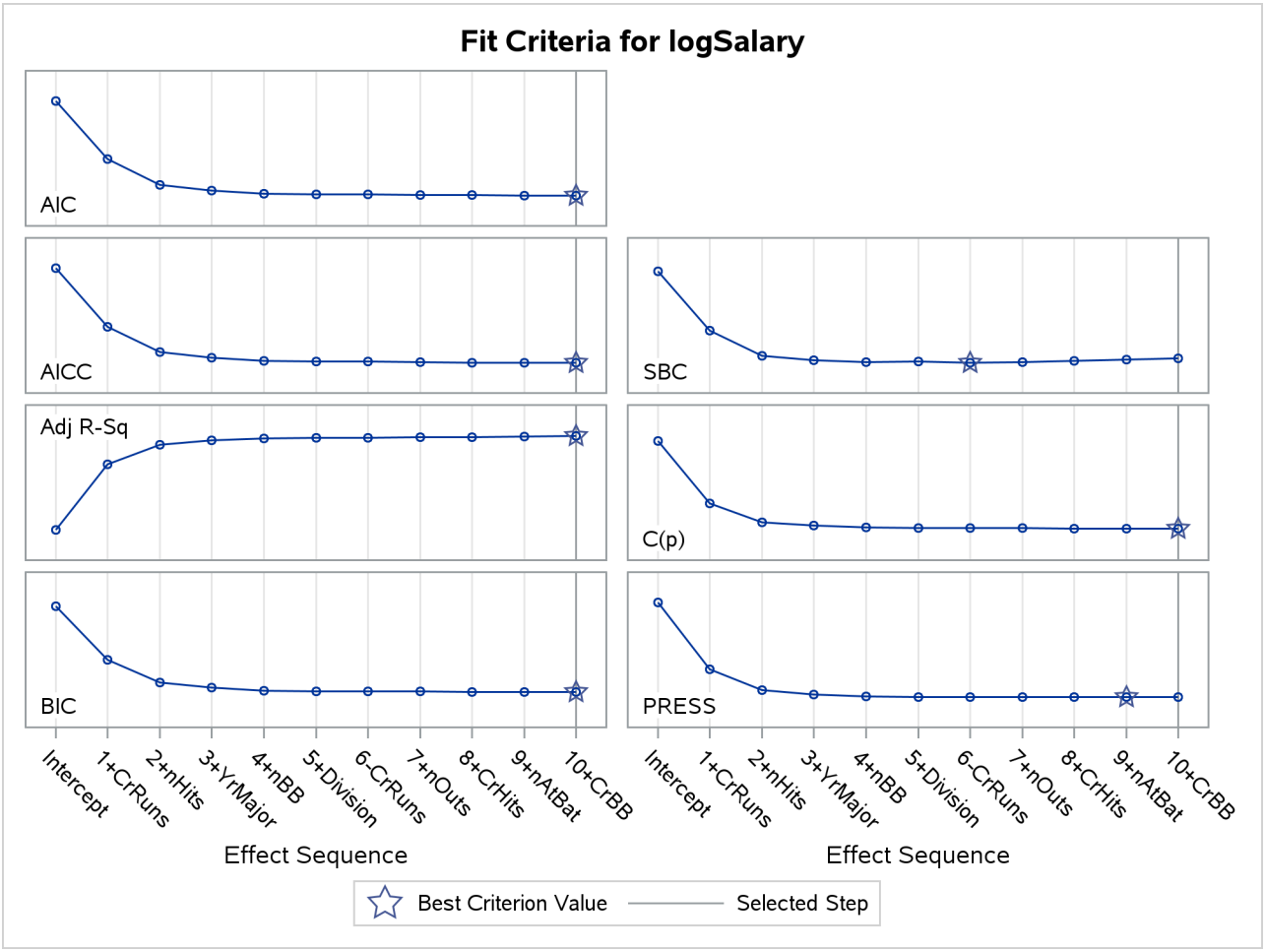

You can see in Output 56.1.1 that this stepwise selection process would stop at an earlier step if you use the Schwarz Bayesian information criterion (SBC) or predicted residual sum of squares (PRESS) to assess the selected models as stepwise selection progresses. You can use the CHOOSE= suboption of the SELECTION= option in the MODEL statement to specify the criterion you want to use to select among the evaluated models. The following statements use the PRESS statistic to choose among the models evaluated during the stepwise selection.

```
proc glmselect data=sashelp.baseball;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=stepwise(select=SL choose=PRESS);
run;
```

Note that the selected model is the model at step 9. By default, PROC GLMSELECT displays the selected model, ANOVA and fit statistics, and parameter estimates for the selected model. These are shown in Output 56.1.2.

Output 56.1.2 Details of Selected Model

The GLMSELECT Procedure
Selected Model

The selected model, based on PRESS, is the model at Step 9.

Effects: Intercept nAtBat nHits nBB YrMajor CrHits Division nOuts

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	7	124.67715	17.81102	55.07
Error	255	82.47658	0.32344	
Corrected Total	262	207.15373		

Root MSE	0.56872
Dependent Mean	5.92722
R-Square	0.6019
Adj R-Sq	0.5909
AIC	-23.98522
AICC	-23.27376
PRESS	88.55275
SBC	-260.40799

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	4.176133	0.150539	27.74
nAtBat	1	-0.001468	0.000946	-1.55
nHits	1	0.011078	0.002983	3.71
nBB	1	0.007226	0.002115	3.42
YrMajor	1	0.070056	0.018911	3.70
CrHits	1	0.000247	0.000143	1.72
Division East	1	0.143082	0.070972	2.02
Division West	0	0	.	.
nOuts	1	0.000241	0.000134	1.81

Even though the model that is chosen to give the smallest value of the PRESS statistic is the model at step 9, the stepwise selection process continues to the step where the stopping condition based on entry and stay significance levels is met. If you use the PRESS statistic as the stopping criterion, the stepwise selection process stops at step 9. This ability to stop at the first extremum of the criterion you specify can significantly reduce the amount of computation done, especially in the cases where you are selecting from a large number of effects. The following statements request stopping based on the PRESS statistic. The stop reason and stop details tables are shown in Output 56.1.3.

```
proc glmselect data=sashelp.baseball plot=Coefficients;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=stepwise(select=SL stop=PRESS);
```

```
run;
```

Output 56.1.3 Stopping Based on PRESS Statistic

The GLMSELECT Procedure

Selection stopped at a local minimum of the PRESS criterion.

Stop Details				
Candidate For	**Effect**	**Candidate PRESS**		**Compare PRESS**
Entry	CrBB	88.6321	>	88.5528
Removal	nAtBat	88.6866	>	88.5528

The PLOTS=COEFFICIENTS specification in the PROC GLMSELECT statement requests a plot that enables you to visualize the selection process.

Output 56.1.4 Coefficient Progression Plot

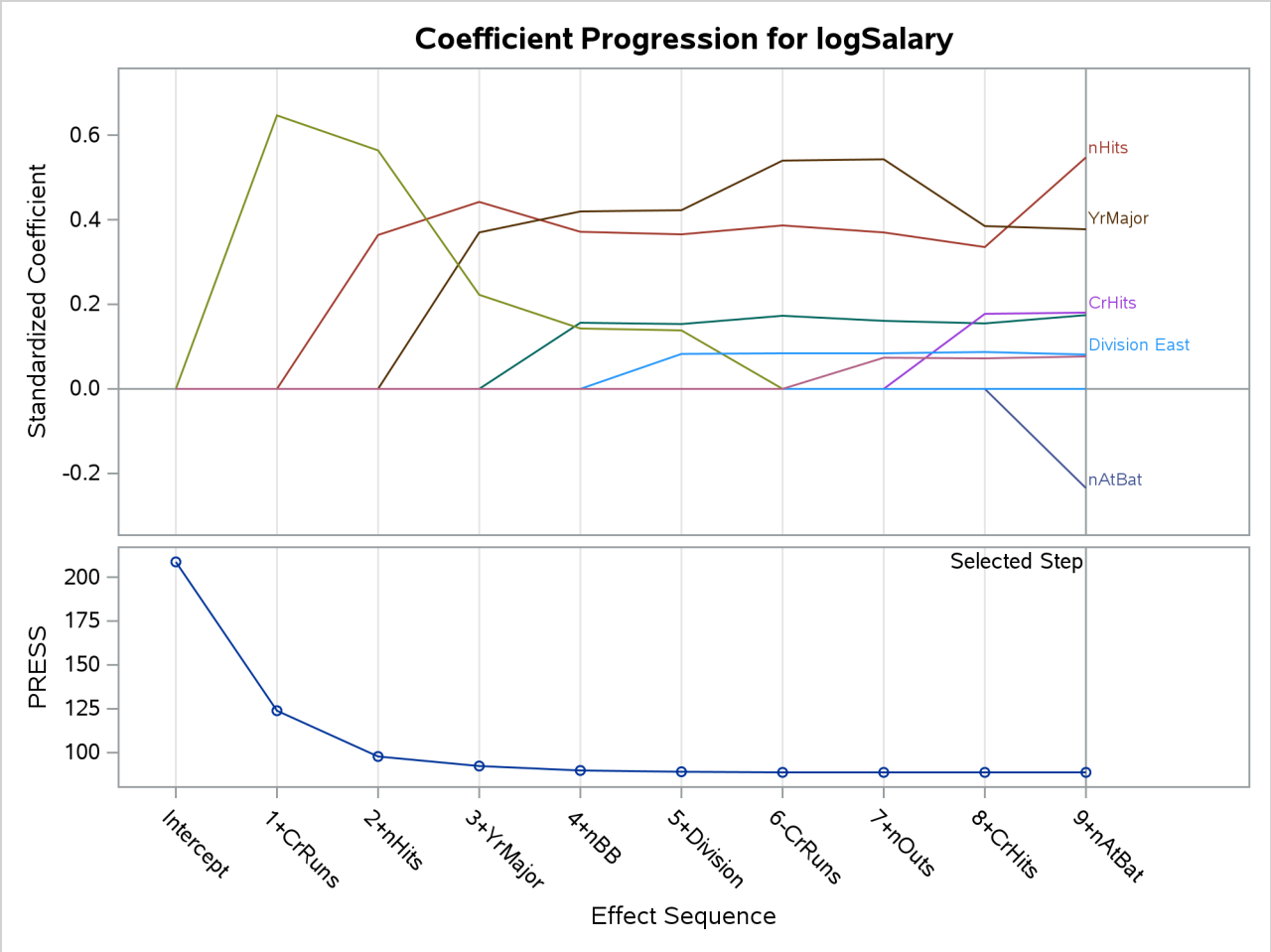

Output 56.1.4 shows the standardized coefficients of all the effects selected at some step of the stepwise method plotted as a function of the step number. This enables you to assess the relative importance of the effects selected at any step of the selection process as well as providing information as to when effects entered

the model. The lower plot in the panel shows how the criterion used to choose the selected model changes as effects enter or leave the model.

Model selection is often done in order to obtain a parsimonious model that can be used for prediction on new data. An ever-present danger is that of selecting a model that overfits the "training" data used in the fitting process, yielding a model with poor predictive performance. Using cross validation is one way to assess the predictive performance of the model. Using k-fold cross validation, the training data are subdivided into k parts, and at each step of the selection process, models are obtained on each of the k subsets of the data obtained by omitting one of these parts. The cross validation predicted residual sum of squares, denoted CV PRESS, is obtained by summing the squares of the residuals when each of these submodels is scored on the data omitted in fitting the submodel. Note that the PRESS statistic corresponds to the special case of "leave-one-out" cross validation.

In the preceding example, the PRESS statistic was used to choose among models that were chosen based on entry and stay significance levels. In the following statements, the SELECT=CVPRESS suboption of the SELECTION= option in the MODEL statement requests that the CV PRESS statistic itself be used as the selection criterion. The DROP=COMPETITIVE suboption requests that additions and deletions be considered simultaneously when deciding whether to add or remove an effect. At any step, the CV PRESS statistic for all models obtained by deleting one effect from the model or adding one effect to the model is computed. Among these models, the one yielding the smallest value of the CV PRESS statistic is selected and the process is repeated from this model. The stepwise selection terminates if all additions or deletions increase the CV PRESS statistic. The CVMETHOD=SPLIT(5) option requests five-fold cross validation with the five subsets consisting of observations $\{1, 6, 11, \ldots\}$, $\{2, 7, 12, \ldots\}$, and so on.

```
proc glmselect data=sashelp.baseball plot=Candidates;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=stepwise(select=CV drop=competitive)
                     cvMethod=split(5);
run;
```

The selection summary table is shown in Output 56.1.5. By comparing Output 56.1.5 and Output 56.7 you can see that the sequence of models produced is different from the sequence when the stepwise selection is based on the SBC statistic.

Output 56.1.5 Stepwise Selection Based on Cross Validation

The GLMSELECT Procedure

Stepwise Selection Summary					
Step	Effect Entered	Effect Removed	Number Effects In	Number Parms In	CV PRESS
0	Intercept		1	1	208.9638
1	CrRuns		2	2	122.5755
2	nHits		3	3	96.3949
3	YrMajor		4	4	92.2117
4	nBB		5	5	89.5242
5		CrRuns	4	4	88.6917
6	League		5	5	88.0417
7	nError		6	6	87.3170
8	Division		7	7	87.2147
9	nHome		8	8	87.0960*
* Optimal Value of Criterion					

If you have sufficient data, another way you can assess the predictive performance of your model is to reserve part of your data for testing your model. You score the model obtained using the training data on the test data and assess the predictive performance on these data that had no role in the selection process. You can also reserve part of your data to validate the model you obtain in the training process. Note that the validation data are not used in obtaining the coefficients of the model, but they are used to decide when to stop the selection process to limit overfitting.

PROC GLMSELECT enables you to partition your data into disjoint subsets for training validation and testing roles. This partitioning can be done by using random proportions of the data, or you can designate a variable in your data set that defines which observations to use for each role. See the section "PARTITION Statement" on page 4455 for more details.

The following statements randomly partition the baseball data set, using 50% for training, 30% for validation, and 20% for testing. The model selected at each step is scored on the validation data, and the average residual sums of squares (ASE) is evaluated. The model yielding the lowest ASE on the validation data is selected. The ASE on the test data is also evaluated, but these data play no role in the selection process. Note that a seed for the pseudo-random number generator is specified in the PROC GLMSELECT statement.

```
proc glmselect data=sashelp.baseball plots=(CriterionPanel ASE) seed=1;
   partition fraction(validate=0.3 test=0.2);
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=forward(choose=validate stop=10);
run;
```

Output 56.1.6 Number of Observations Table

The GLMSELECT Procedure

Number of Observations Read	322
Number of Observations Used	263
Number of Observations Used for Training	132
Number of Observations Used for Validation	80
Number of Observations Used for Testing	51

Output 56.1.6 shows the number of observation table. You can see that of the 263 observations that were used in the analysis, 132 (50.2%) observations were used for model training, 80 (30.4%) for model validation, and 51 (19.4%) for model testing.

Output 56.1.7 Selection Summary and Stop Reason

The GLMSELECT Procedure

Forward Selection Summary

Step	Effect Entered	Number Effects In	Number Parms In	SBC	ASE	Validation ASE	Test ASE
0	Intercept	1	1	-30.8531	0.7628	0.7843	0.8818
1	CrRuns	2	2	-93.9367	0.4558	0.4947	0.4210
2	nHits	3	3	-126.2647	0.3439	0.3248	0.4697
3	YrMajor	4	4	-128.7570	0.3252	0.2920*	0.4614
4	nBB	5	5	-132.2409*	0.3052	0.3065	0.4297
5	Division	6	6	-130.7794	0.2974	0.3050	0.4218
6	nOuts	7	7	-128.5897	0.2914	0.3028	0.4186
7	nRBI	8	8	-125.7825	0.2868	0.3097	0.4489
8	nHome	9	9	-124.7709	0.2786	0.3383	0.4533
9	nAtBat	10	10	-121.3767	0.2754	0.3337	0.4580

*** Optimal Value of Criterion**

Selection stopped at the first model containing the specified number of effects (10).

Output 56.1.7 shows the selection summary table and the stop reason. The forward selection stops at step 9 since the model at this step contains 10 effects, and so it satisfies the stopping criterion requested with the STOP=10 suboption. However, the selected model is the model at step 3, where the validation ASE, the CHOOSE= criterion, achieves its minimum.

Output 56.1.8 Criterion Panel

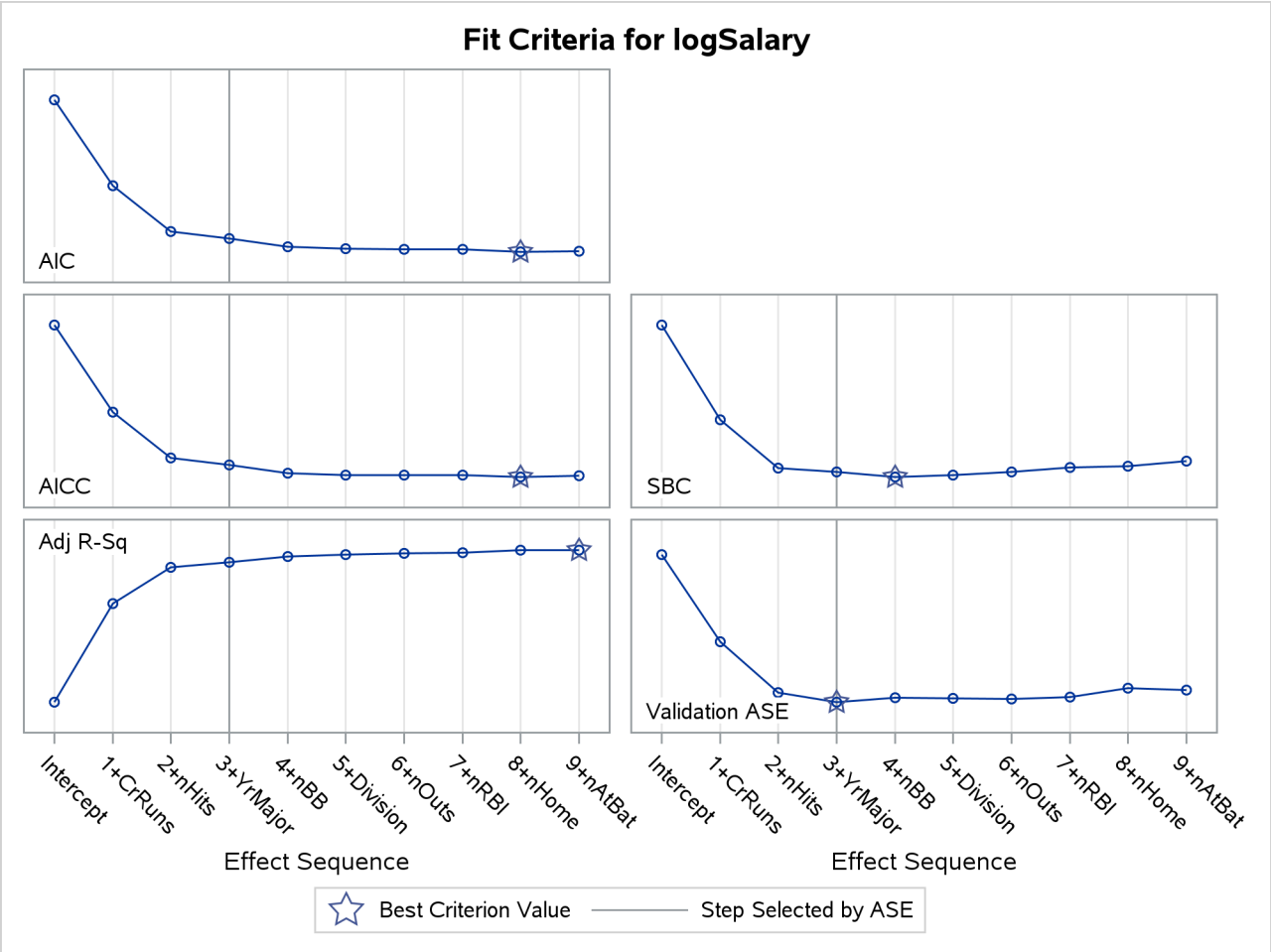

The criterion panel in Output 56.1.8 shows how the various criteria evolved as the stepwise selection method proceeded. Note that other than the ASE evaluated on the validation data, these criteria are evaluated on the training data.

Output 56.1.9 Average Square Errors by Role

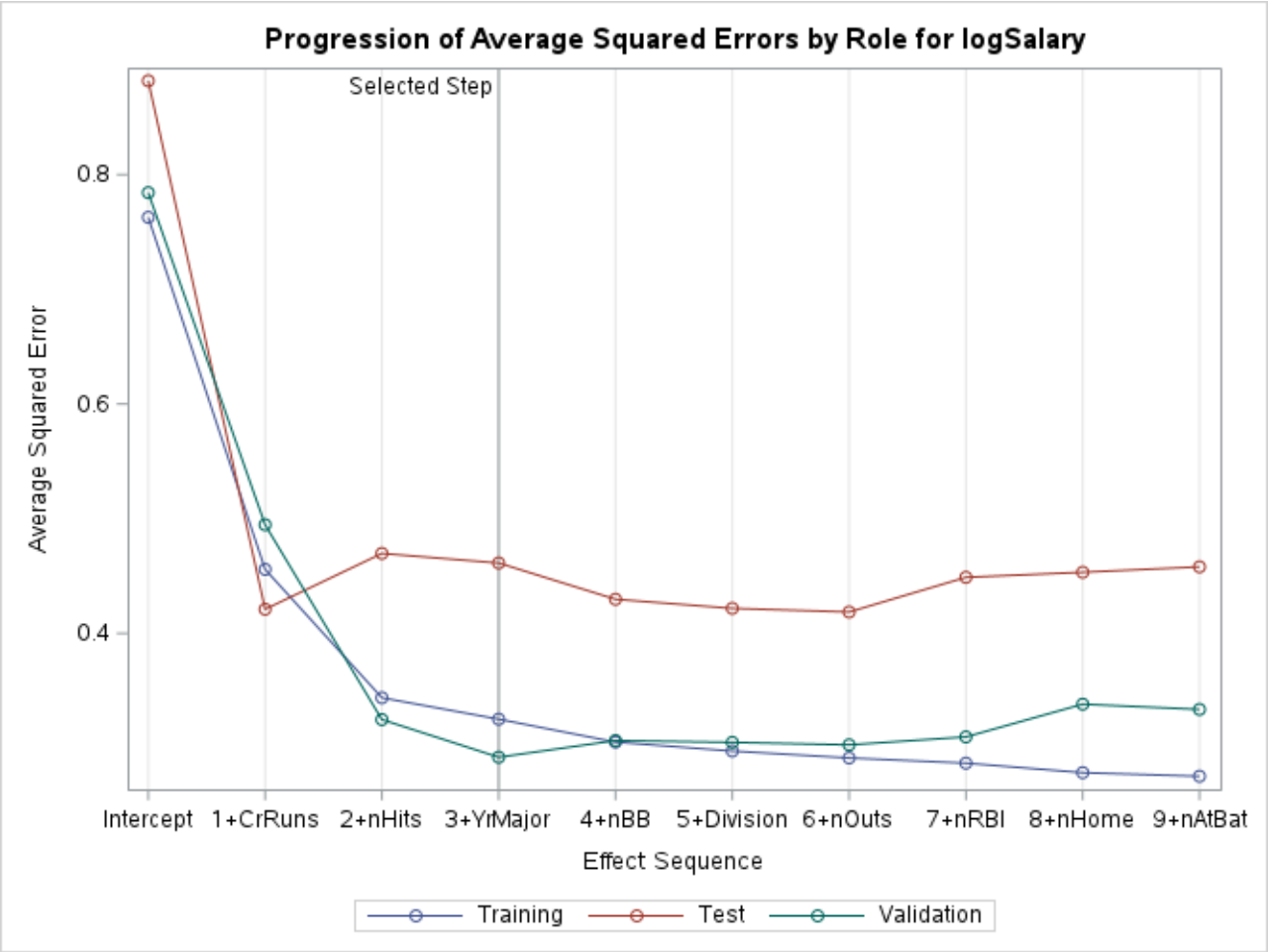

Finally, the ASE plot in Output 56.1.9 shows how the average square error evolves on the training, validation, and test data. Note that while the ASE on the training data continued decreasing as the selection steps proceeded, the ASE on the test and validation data behave more erratically.

LASSO selection, pioneered by Tibshirani (1996), is a constrained least squares method that can be viewed as a stepwise-like method where effects enter and leave the model sequentially. You can find additional details about the LASSO method in the section "Lasso Selection (LASSO)" on page 4464. Note that when classification effects are used with LASSO, the design matrix columns for all effects containing classification variables can enter or leave the model individually. The following statements perform LASSO selection for the baseball data. The LASSO selection summary table is shown in Output 56.1.10.

```
proc glmselect data=sashelp.baseball plot=CriterionPanel ;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError
                   / selection=LASSO(choose=CP steps=20);
run;
```

```
ods graphics off;
```

Output 56.1.10 Selection Summary for LASSO Selection

The GLMSELECT Procedure

LASSO Selection Summary				
Step	Effect Entered	Effect Removed	Number Effects In	CP
0	Intercept		1	375.9275
1	CrRuns		2	328.6492
2	CrHits		3	239.5392
3	nHits		4	134.0374
4	nBB		5	111.6638
5	CrRbi		6	81.7296
6	YrMajor		7	75.0428
7	nRBI		8	30.4494
8	Division_East		9	29.9913
9	nOuts		10	25.1656
10		CrRuns	9	18.7295
11		CrRbi	8	15.1683
12	nError		9	16.6233
13	nHome		10	16.3741
14	League_American		11	14.8794
15		nRBI	10	8.8477*
16	CrBB		11	9.2242
17	CrRuns		12	10.7608
18	nAtBat		13	11.6266
19	nAssts		14	11.8572
20	CrAtBat		15	13.4020
* Optimal Value of Criterion				

Selection stopped at the specified number of steps (20).

Note that effects enter and leave sequentially. In this example, the STEPS= suboption of the SELECTION= option specifies that 20 steps of LASSO selection be done. You can see how the various model fit statistics evolved in Output 56.1.11.

Output 56.1.11 Criterion Panel

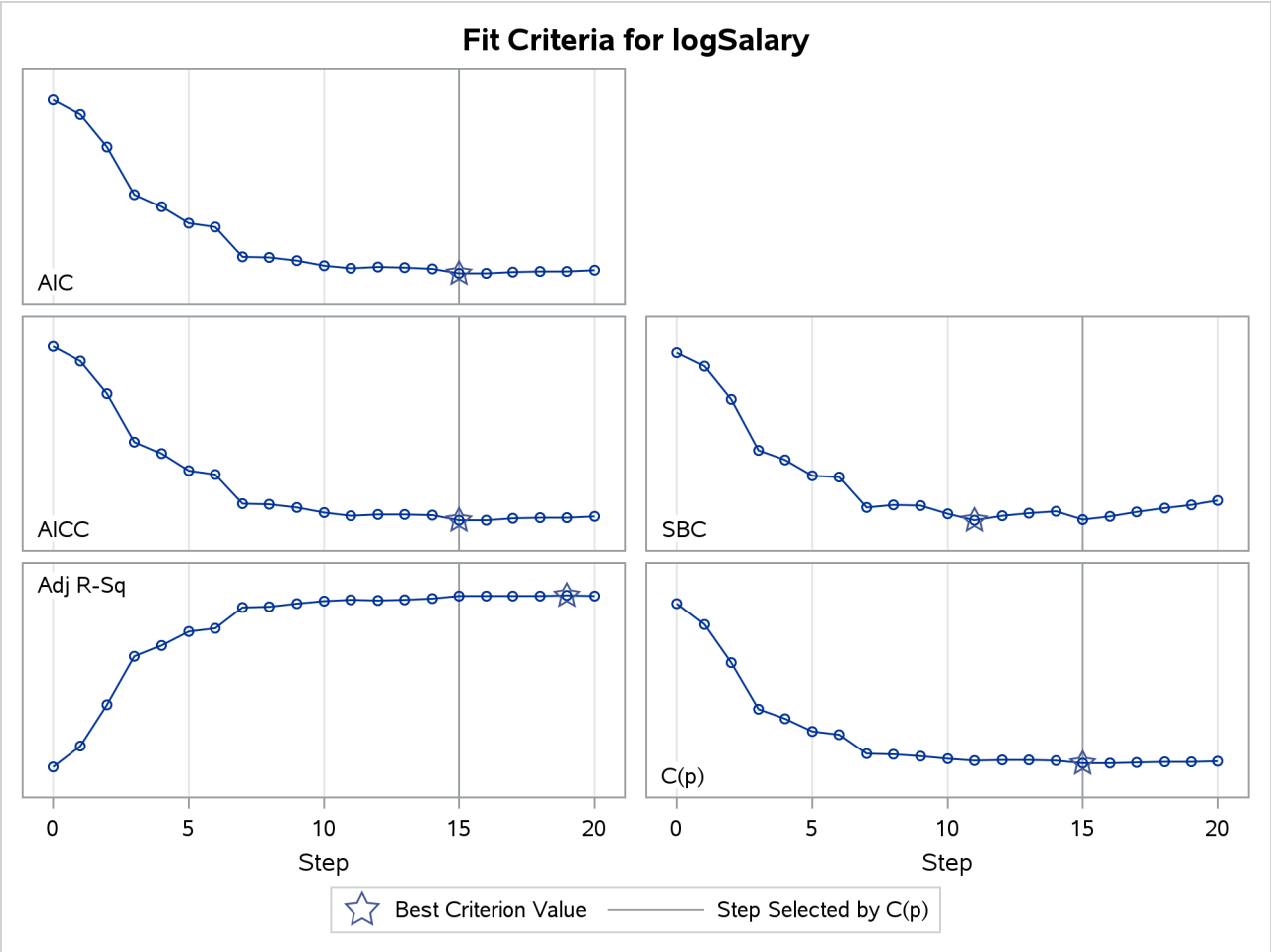

The CHOOSE=CP suboption specifies that the selected model be the model at step 15 that yields the optimal value of Mallows' C_p statistic. Details of this selected model are shown in Output 56.1.12.

Output 56.1.12 Selected Model

The GLMSELECT Procedure
Selected Model

The selected model, based on C(p), is the model at Step 15.

Effects: Intercept nHits nHome nBB YrMajor CrHits League_American Division_East nOuts nError

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	9	125.24302	13.91589	42.98
Error	253	81.91071	0.32376	
Corrected Total	262	207.15373		

Output 56.1.12 *continued*

Root MSE	0.56900
Dependent Mean	5.92722
R-Square	0.6046
Adj R-Sq	0.5905
AIC	-21.79589
AICC	-20.74409
BIC	-283.91417
C(p)	8.84767
SBC	-251.07435

Parameter Estimates

Parameter	DF	Estimate
Intercept	1	4.204236
nHits	1	0.006942
nHome	1	0.002785
nBB	1	0.005727
YrMajor	1	0.067054
CrHits	1	0.000249
League_American	1	-0.079607
Division_East	1	0.134723
nOuts	1	0.000183
nError	1	-0.007213

Example 56.2: Using Validation and Cross Validation

(View the complete code for this example (glsex2.sas) in the example repository.)

This example shows how you can use both test set and cross validation to monitor and control variable selection. It also demonstrates the use of split classification variables.

The following statements produce analysis and test data sets. Note that the same statements are used to generate the observations that are randomly assigned for analysis and test roles in the ratio of approximately two to one.

```
data analysisData testData;
   drop i j c3Num;
   length c3 $ 7;

   array x{20} x1-x20;

   do i=1 to 1500;
      do j=1 to 20;
         x{j} = ranuni(1);
      end;

      c1 = 1 + mod(i,8);
      c2 = ranbin(1,3,.6);
```

```
      if      i < 50   then do; c3 = 'tiny';     c3Num=1;end;
      else if i < 250  then do; c3 = 'small';    c3Num=1;end;
      else if i < 600  then do; c3 = 'average';  c3Num=2;end;
      else if i < 1200 then do; c3 = 'big';      c3Num=3;end;
      else                  do; c3 = 'huge';     c3Num=5;end;

      y = 10 + x1 + 2*x5 + 3*x10 + 4*x20  + 3*x1*x7 + 8*x6*x7
             + 5*(c1=3)*c3Num + 8*(c1=7)  + 5*rannor(1);

      if ranuni(1) < 2/3 then output analysisData;
                         else output testData;
   end;
run;
```

Suppose you suspect that the dependent variable depends on both main effects and two-way interactions. You can use the following statements to select a model:

```
ods graphics on;

proc glmselect data=analysisData testdata=testData
               seed=1 plots(stepAxis=number)=(criterionPanel ASEPlot);
   partition fraction(validate=0.5);
   class c1 c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2
          / selection=stepwise(choose = validate
                               select = sl)
             hierarchy=single stb;
run;
```

Note that a TESTDATA= data set is named in the PROC GLMSELECT statement and that a PARTITION statement is used to randomly assign half the observations in the analysis data set for model validation and the rest for model training. You find details about the number of observations used for each role in the number of observations tables shown in Output 56.2.1.

Output 56.2.1 Number of Observations Tables

The GLMSELECT Procedure

Observation Profile for Analysis Data	
Number of Observations Read	1010
Number of Observations Used	1010
Number of Observations Used for Training	510
Number of Observations Used for Validation	500

The “Class Level Information” and “Dimensions” tables are shown in Output 56.2.2. The “Dimensions” table shows that at each step of the selection process, 278 effects are considered as candidates for entry or removal. Since several of these effects have multilevel classification variables as members, there are 661 parameters.

Output 56.2.2 Class Level Information and Problem Dimensions

Class Level Information		
Class	**Levels**	**Values**
c1	8	1 2 3 4 5 6 7 8
c2	4	0 1 2 3
c3	5	tiny small average big huge

Dimensions	
Number of Effects	278
Number of Parameters	661

The model statement options request stepwise selection with the default entry and stay significance levels used for both selecting entering and departing effects and stopping the selection method. The CHOOSE=VALIDATE suboption specifies that the selected model is chosen to minimize the predicted residual sum of squares when the models at each step are scored on the observations reserved for validation. The HIERARCHY=SINGLE option specifies that interactions can enter the model only if the corresponding main effects are already in the model, and that main effects cannot be dropped from the model if an interaction with such an effect is in the model. These settings are listed in the model information table shown in Output 56.2.3.

Output 56.2.3 Model Information

The GLMSELECT Procedure

Data Set	WORK.ANALYSISDATA
Test Data Set	WORK.TESTDATA
Dependent Variable	y
Selection Method	Stepwise
Select Criterion	Significance Level
Stop Criterion	Significance Level
Choose Criterion	Validation ASE
Entry Significance Level (SLE)	0.15
Stay Significance Level (SLS)	0.15
Effect Hierarchy Enforced	Single
Random Number Seed	1

The stop reason and stop details tables are shown in Output 56.2.4. Note that because the STOP= suboption of the SELECTION= option was not explicitly specified, the stopping criterion used is the selection criterion, namely significance level.

Output 56.2.4 Stop Details

Selection stopped because the candidate for entry has SLE > 0.15 and the candidate for removal has SLS < 0.15.

Stop Details					
Candidate For	**Effect**	**Candidate Significance**		**Compare Significance**	
Entry	x2*x5	0.1742	>	0.1500	(SLE)
Removal	x5*x10	0.0534	<	0.1500	(SLS)

The criterion panel in Output 56.2.5 shows how the various fit criteria evolved as the stepwise selection method proceeded. Note that other than the ASE evaluated on the validation data, these criteria are evaluated on the training data. You see that the minimum of the validation ASE occurs at step 9, and hence the model at this step is selected.

Output 56.2.5 Criterion Panel

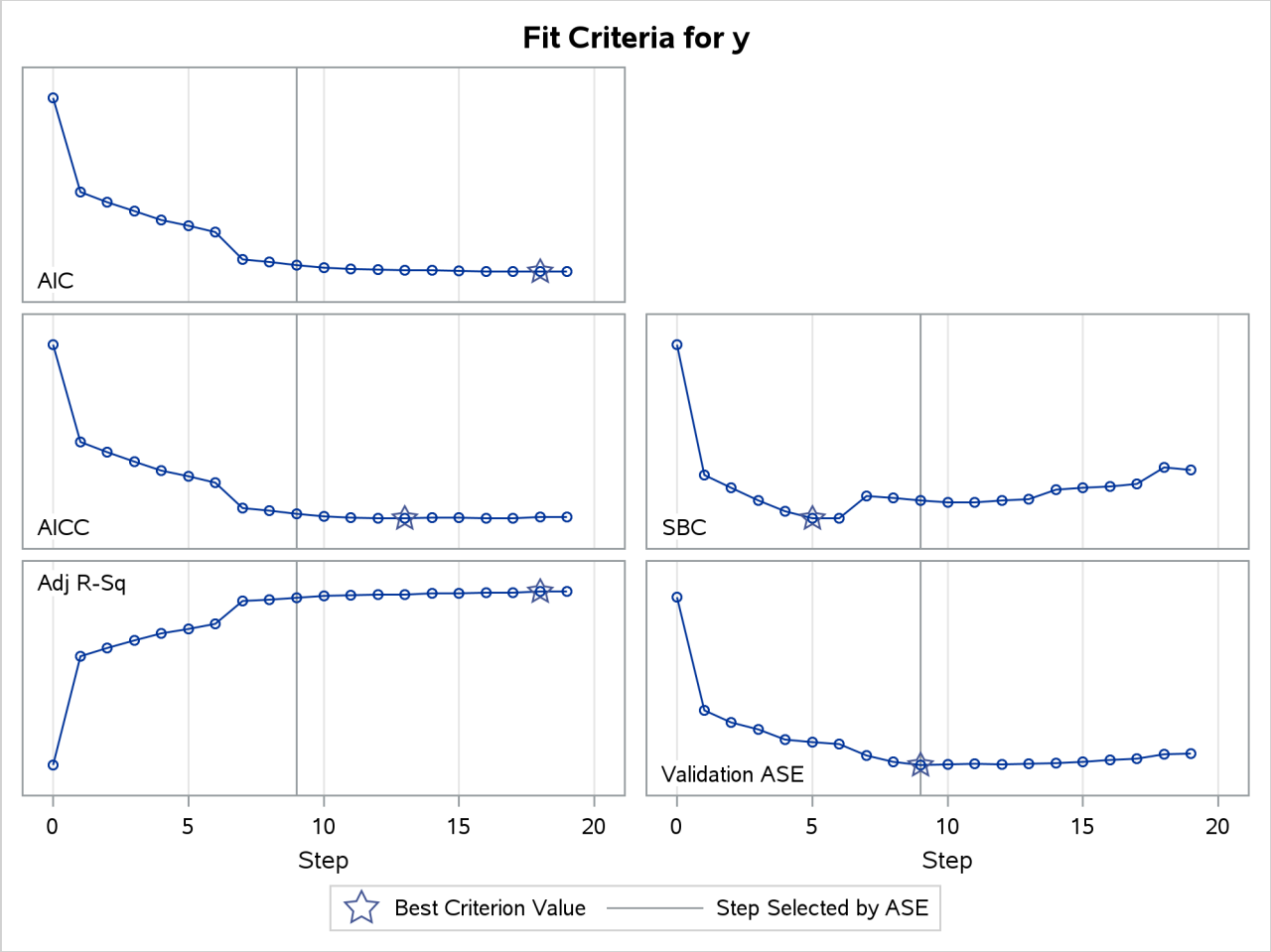

Output 56.2.6 shows how the average squared error (ASE) evolved on the training, validation, and test data. Note that while the ASE on the training data decreases monotonically, the errors on both the validation and test data start increasing beyond step 9. This indicates that models after step 9 are beginning to overfit the training data.

Output 56.2.6 Average Squared Errors

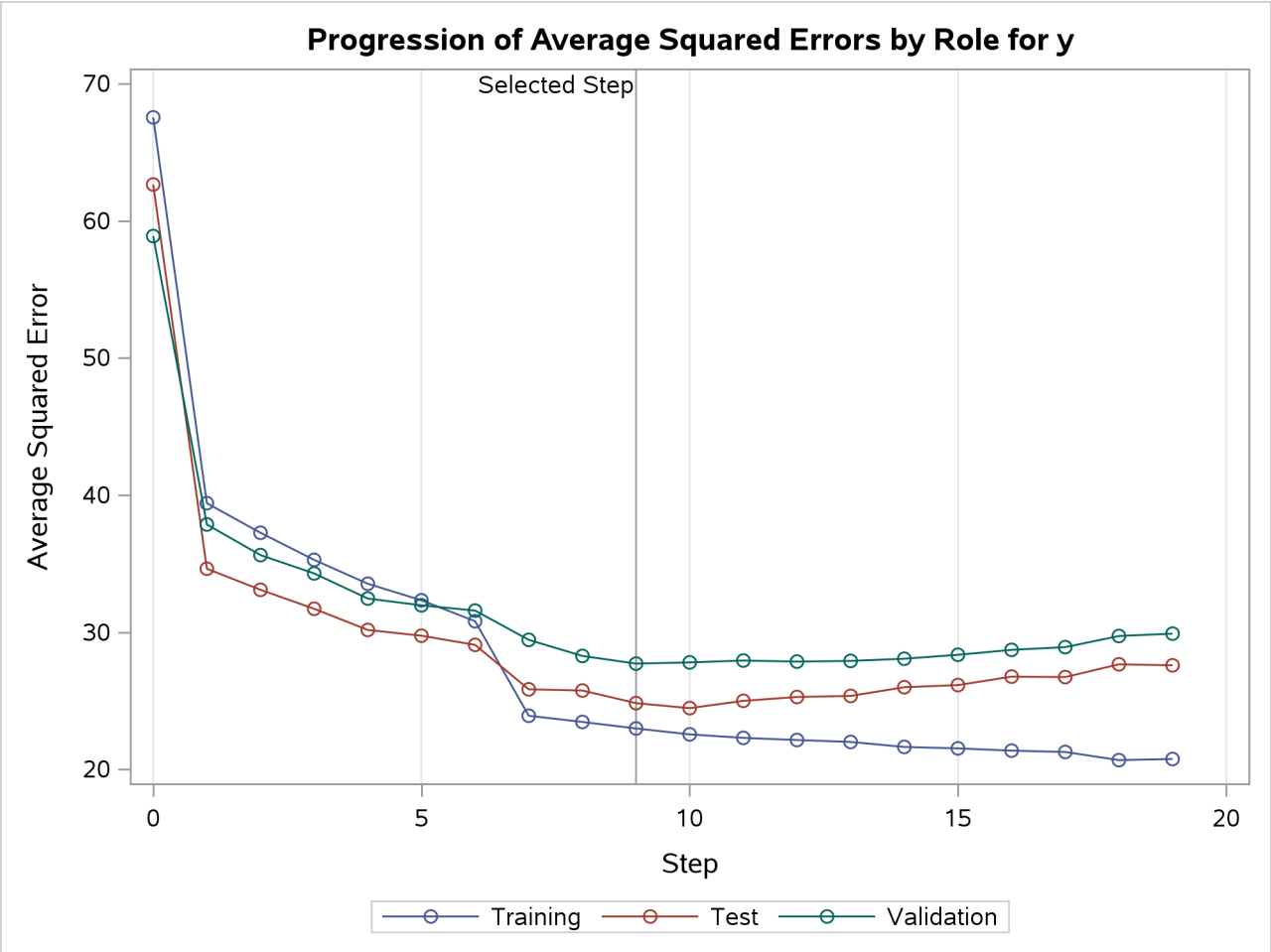

Output 56.2.7 shows the selected effects, analysis of variance, and fit statistics tables for the selected model. Output 56.2.8 shows the parameter estimates table.

Output 56.2.7 Selected Model Details

The GLMSELECT Procedure
Selected Model

The selected model, based on Validation ASE, is the model at Step 9.

Effects:	Intercept c1 c3 c1*c3 x1 x5 x6 x7 x10 x20

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	44	22723	516.43621	20.49
Error	465	11722	25.20856	
Corrected Total	509	34445		

Output 56.2.7 *continued*

Root MSE	5.02081
Dependent Mean	21.09705
R-Square	0.6597
Adj R-Sq	0.6275
AIC	2200.75319
AICC	2210.09228
SBC	1879.30167
ASE (Train)	22.98427
ASE (Validate)	27.71105
ASE (Test)	24.82947

Output 56.2.8 Parameter Estimates

Parameter Estimates						
Parameter		DF	Estimate	Standardized Estimate	Standard Error	t Value
Intercept		1	6.867831	0	1.524446	4.51
c1	1	1	0.226602	0.008272	2.022069	0.11
c1	2	1	-1.189623	-0.048587	1.687644	-0.70
c1	3	1	25.968930	1.080808	1.693593	15.33
c1	4	1	1.431767	0.054892	1.903011	0.75
c1	5	1	1.972622	0.073854	1.664189	1.19
c1	6	1	-0.094796	-0.004063	1.898700	-0.05
c1	7	1	5.971432	0.250037	1.846102	3.23
c1	8	0	0	0	.	.
c3	tiny	1	-2.919282	-0.072169	2.756295	-1.06
c3	small	1	-4.635843	-0.184338	2.218541	-2.09
c3	average	1	0.736805	0.038247	1.793059	0.41
c3	big	1	-1.078463	-0.063580	1.518927	-0.71
c3	huge	0	0	0	.	.
c1*c3	1 tiny	1	-2.449964	-0.018632	4.829146	-0.51
c1*c3	1 small	1	5.265031	0.069078	3.470382	1.52
c1*c3	1 average	1	-3.489735	-0.064365	2.850381	-1.22
c1*c3	1 big	1	0.725263	0.017929	2.516502	0.29
c1*c3	1 huge	0	0	0	.	.
c1*c3	2 tiny	1	5.455122	0.050760	4.209507	1.30
c1*c3	2 small	1	7.439196	0.131499	2.982411	2.49
c1*c3	2 average	1	-0.739606	-0.014705	2.568876	-0.29
c1*c3	2 big	1	3.179351	0.078598	2.247611	1.41
c1*c3	2 huge	0	0	0	.	.
c1*c3	3 tiny	1	-19.266847	-0.230989	3.784029	-5.09
c1*c3	3 small	1	-15.578909	-0.204399	3.266216	-4.77
c1*c3	3 average	1	-18.119398	-0.395770	2.529578	-7.16
c1*c3	3 big	1	-10.650012	-0.279796	2.205331	-4.83
c1*c3	3 huge	0	0	0	.	.
c1*c3	4 tiny	0	0	0	.	.
c1*c3	4 small	1	4.432753	0.047581	3.677008	1.21
c1*c3	4 average	1	-3.976295	-0.091632	2.625564	-1.51
c1*c3	4 big	1	-1.306998	-0.033003	2.401064	-0.54
c1*c3	4 huge	0	0	0	.	.
c1*c3	5 tiny	1	6.714186	0.062475	4.199457	1.60
c1*c3	5 small	1	1.565637	0.022165	3.182856	0.49
c1*c3	5 average	1	-4.286085	-0.068668	2.749142	-1.56
c1*c3	5 big	1	-2.046468	-0.045949	2.282735	-0.90
c1*c3	5 huge	0	0	0	.	.
c1*c3	6 tiny	1	5.135111	0.039052	4.754845	1.08
c1*c3	6 small	1	4.442898	0.081945	3.079524	1.44
c1*c3	6 average	1	-2.287870	-0.056559	2.601384	-0.88
c1*c3	6 big	1	1.598086	0.043542	2.354326	0.68
c1*c3	6 huge	0	0	0	.	.
c1*c3	7 tiny	1	1.108451	0.010314	4.267509	0.26
c1*c3	7 small	1	7.441059	0.119214	3.135404	2.37
c1*c3	7 average	1	1.796483	0.038106	2.630570	0.68

Output 56.2.8 *continued*

Parameter Estimates						
Parameter		DF	Estimate	Standardized Estimate	Standard Error	t Value
c1*c3	7 big	1	3.324160	0.095173	2.303369	1.44
c1*c3	7 huge	0	0	0	.	.
c1*c3	8 tiny	0	0	0	.	.
c1*c3	8 small	0	0	0	.	.
c1*c3	8 average	0	0	0	.	.
c1*c3	8 big	0	0	0	.	.
c1*c3	8 huge	0	0	0	.	.
x1		1	2.713527	0.091530	0.836942	3.24
x5		1	2.810341	0.098303	0.816290	3.44
x6		1	4.837022	0.167394	0.810402	5.97
x7		1	5.844394	0.207035	0.793775	7.36
x10		1	2.463916	0.087712	0.794599	3.10
x20		1	4.385924	0.156155	0.787766	5.57

The magnitudes of the standardized estimates and the *t* statistics of the parameters of the effect `'c1'` reveal that only levels `'3'` and `'7'` of this effect contribute appreciably to the model. This suggests that a more parsimonious model with similar or better predictive power might be obtained if parameters corresponding to the levels of `'c1'` are allowed to enter or leave the model independently. You request this with the SPLIT option in the CLASS statement as shown in the following statements:

```
proc glmselect data=analysisData testdata=testData
               seed=1 plots(stepAxis=number)=all;
   partition fraction(validate=0.5);
   class c1(split) c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2
           / selection=stepwise(stop   = validate
                                select = sl)
             hierarchy=single;
   output out=outData;
run;
```

The "Class Level Information" and "Dimensions" tables are shown in Output 56.2.9. The "Dimensions" table shows that while the model statement specifies 278 effects, after splitting the parameters corresponding to the levels of c1, there are 439 split effects that are considered for entry or removal at each step of the selection process. Note that the total number of parameters considered is not affected by the split option.

Output 56.2.9 Class Level Information and Problem Dimensions

The GLMSELECT Procedure

Class Level Information		
Class	Levels	Values
c1	8	* 1 2 3 4 5 6 7 8
c2	4	0 1 2 3
c3	5	tiny small average big huge
* Associated Parameters Split		

Output 56.2.9 *continued*

Dimensions	
Number of Effects	278
Number of Effects after Splits	439
Number of Parameters	661

The stop reason and stop details tables are shown in Output 56.2.10. Since the validation ASE is specified as the stopping criterion, the selection stops at step 11, where the validation ASE achieves a local minimum and the model at this step is the selected model.

Output 56.2.10 Stop Details

Selection stopped at a local minimum of the residual sum of squares of the validation data.

Stop Details				
Candidate For	**Effect**	**Candidate Validation ASE**		**Compare Validation ASE**
Entry	x18	25.9851	>	25.7462
Removal	x6*x7	25.7611	>	25.7462

You find details of the selected model in Output 56.2.11. The list of selected effects confirms that parameters corresponding to levels `'3'` and `'7'` only of c1 are in the selected model. Notice that the selected model with classification variable c1 split contains 18 parameters, whereas the selected model without splitting c1 has 45 parameters. Furthermore, by comparing the fit statistics in Output 56.2.7 and Output 56.2.11, you see that this more parsimonious model has smaller prediction errors on both the validation and test data.

Output 56.2.11 Details of the Selected Model

The GLMSELECT Procedure
Selected Model

The selected model is the model at the last step (Step 11).

Effects: Intercept c1_3 c1_7 c3 c1_3*c3 x1 x5 x6 x7 x6*x7 x10 x20

Analysis of Variance				
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**
Model	17	22111	1300.63200	51.88
Error	492	12334	25.06998	
Corrected Total	509	34445		

Output 56.2.11 *continued*

Root MSE	5.00699
Dependent Mean	21.09705
R-Square	0.6419
Adj R-Sq	0.6295
AIC	2172.72685
AICC	2174.27787
SBC	1736.94624
ASE (Train)	24.18515
ASE (Validate)	25.74617
ASE (Test)	22.57297

When you use a PARTITION statement to subdivide the analysis data set, an output data set created with the OUTPUT statement contains a variable named _ROLE_ that shows the role each observation was assigned to. See the section "OUTPUT Statement" on page 4453 and the section "Using Validation and Test Data" on page 4478 for additional details.

The following statements use PROC PRINT to produce Output 56.2.12, which shows the first five observations of the outData data set.

```
proc print data=outData(obs=5);
run;
```

Output 56.2.12 Output Data Set with _ROLE_ Variable

Obs	c3	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12
1	tiny	0.18496	0.97009	0.39982	0.25940	0.92160	0.96928	0.54298	0.53169	0.04979	0.06657	0.81932	0.52387
2	tiny	0.47579	0.84499	0.63452	0.59036	0.58258	0.37701	0.72836	0.50660	0.93121	0.92912	0.58966	0.29722
3	tiny	0.51132	0.43320	0.17611	0.66504	0.40482	0.12455	0.45349	0.19955	0.57484	0.73847	0.43981	0.04937
4	tiny	0.42071	0.07174	0.35849	0.71143	0.18985	0.14797	0.56184	0.27011	0.32520	0.56918	0.04259	0.43921
5	tiny	0.42137	0.03798	0.27081	0.42773	0.82010	0.84345	0.87691	0.26722	0.30602	0.39705	0.34905	0.76593

Obs	x13	x14	x15	x16	x17	x18	x19	x20	c1	c2	y	_ROLE_	p_y
1	0.85339	0.06718	0.95702	0.29719	0.27261	0.68993	0.97676	0.22651	2	1	11.4391	VALIDATE	18.5069
2	0.39104	0.47243	0.67953	0.16809	0.16653	0.87110	0.29879	0.93464	3	1	31.4596	TRAIN	26.2188
3	0.52238	0.34337	0.02271	0.71289	0.93706	0.44599	0.94694	0.71290	4	3	16.4294	VALIDATE	17.0979
4	0.91744	0.52584	0.73182	0.90522	0.57600	0.18794	0.33133	0.69887	5	3	15.4815	VALIDATE	16.1567
5	0.54340	0.61257	0.55291	0.73591	0.37186	0.64565	0.55718	0.87504	6	2	26.0023	TRAIN	24.6358

Cross validation is often used to assess the predictive performance of a model, especially for when you do not have enough observations for test set validation. See the section "Cross Validation" on page 4480 for further details. The following statements provide an example where cross validation is used as the CHOOSE= criterion.

```
proc glmselect data=analysisData testdata=testData
               plots(stepAxis=number)=(criterionPanel ASEPlot);
   class c1(split) c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2
           / selection = stepwise(choose = cv
```

```
                                        select = sl)
                stats     = press
                cvMethod  = split(5)
                cvDetails = all
                hierarchy = single;
     output out=outData;
   run;
```

The CVMETHOD=SPLIT(5) option in the MODEL statement requests five-fold cross validation with the five subsets consisting of observations $\{1, 6, 11, \ldots\}$, $\{2, 7, 12, \ldots\}$, and so on. The STATS=PRESS option requests that the leave-one-out cross validation predicted residual sum of squares (PRESS) also be computed and displayed at each step, even though this statistic is not used in the selection process.

Output 56.2.13 shows how several fit statistics evolved as the selection process progressed. The five-fold CV PRESS statistic achieves its minimum at step 19. Note that this gives a larger model than was selected when the stopping criterion was determined using validation data. Furthermore, you see that the PRESS statistic has not achieved its minimum within 25 steps, so an even larger model would have been selected based on leave-one-out cross validation.

Output 56.2.13 Criterion Panel

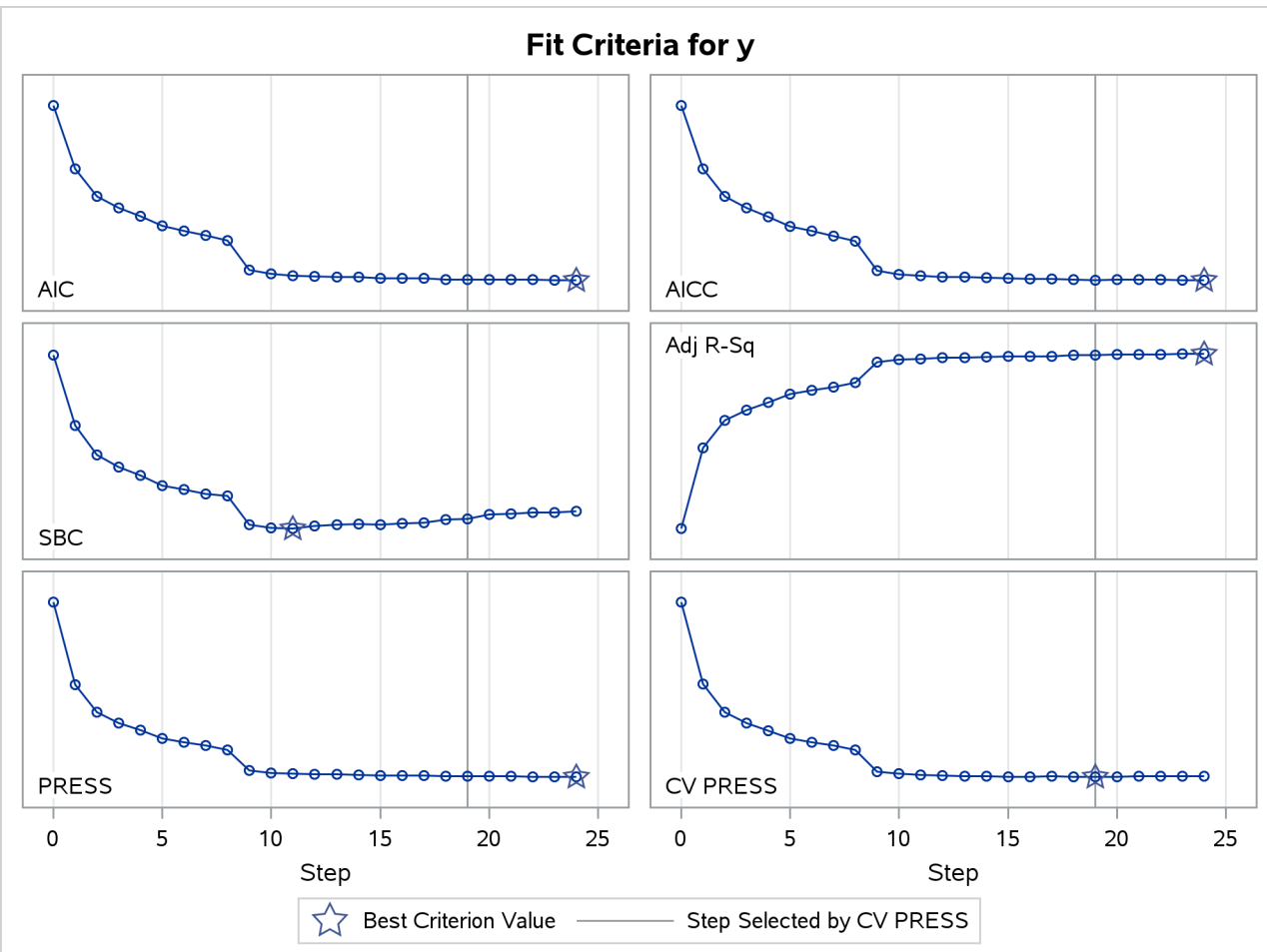

Output 56.2.14 shows how the average squared error compares on the test and training data. Note that the

ASE error on the test data achieves a local minimum at step 11 and is already slowly increasing at step 19, which corresponds to the selected model.

Output 56.2.14 Average Squared Error Plot

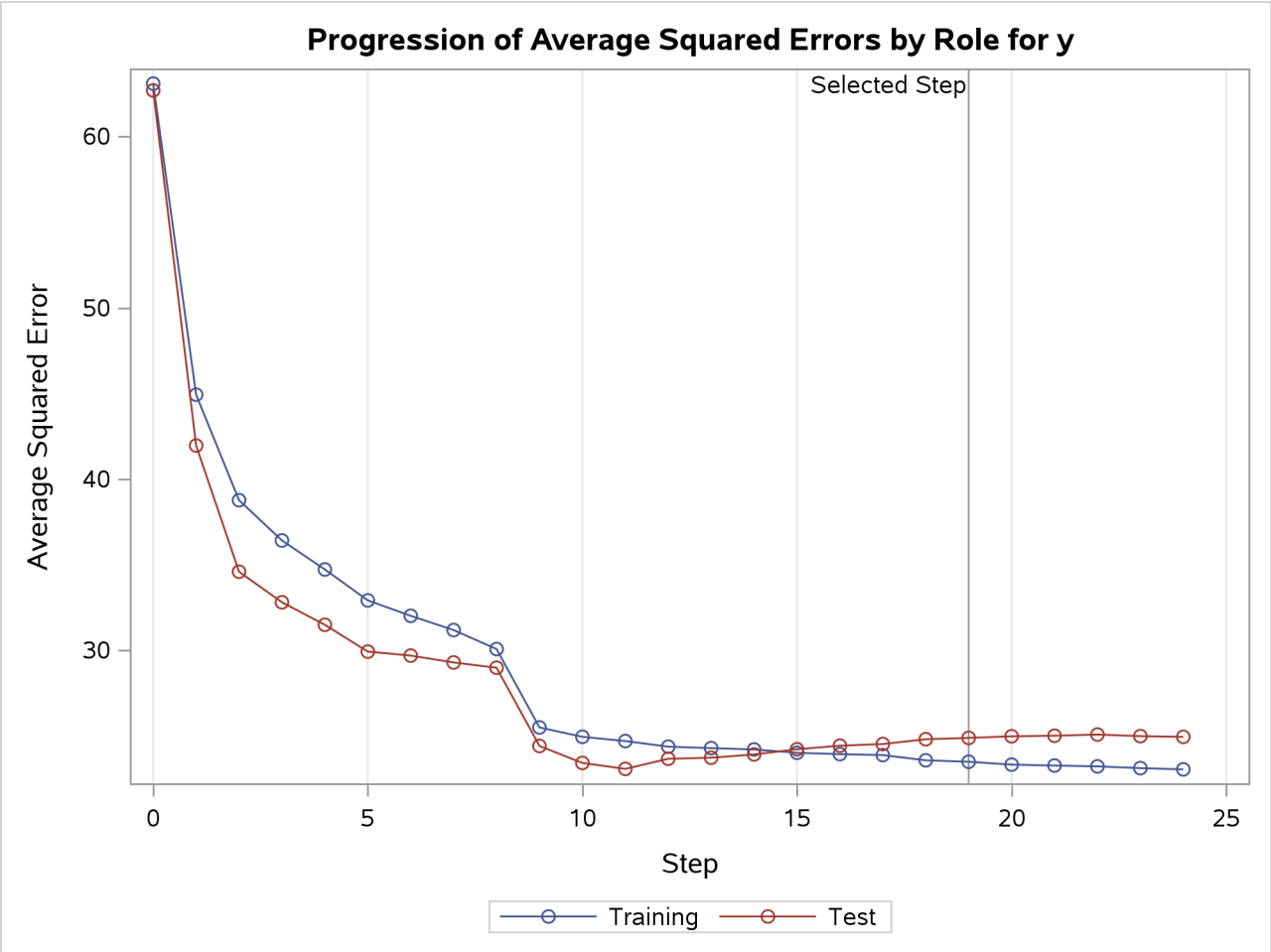

Output 56.2.15 Breakdown of CV Press Statistic by Fold

Cross Validation Details			
	Observations		
Index	Fitted	Left Out	CV PRESS
1	808	202	5031.4484
2	808	202	4209.3671
3	808	202	5616.2516
4	808	202	5020.6697
5	808	202	5415.7655
Total			25293.5024

The CVDETAILS=ALL option in the MODEL statement requests the "Cross Validation Details" table in Output 56.2.15 and the cross validation parameter estimates that are included in the "Parameter Estimates" table in Output 56.2.16. For each cross validation index, the predicted residual sum of squares on the

observations omitted is shown in the "Cross Validation Details" table and the parameter estimates of the corresponding model are included in the "Parameter Estimates" table. By default, these details are shown for the selected model, but you can request this information at every step with the DETAILS= option in the MODEL statement. You use the _CVINDEX_ variable in the output data set shown in Output 56.2.17 to find out which observations in the analysis data are omitted for each cross validation fold.

Output 56.2.16 Cross Validation Parameter Estimates

Parameter Estimates					
	Cross Validation Estimates				
Parameter	**1**	**2**	**3**	**4**	**5**
Intercept	10.7617	10.1200	9.0254	13.4164	12.3352
c1_3	28.2715	27.2977	27.0696	28.6835	27.8070
c1_7	7.6530	7.6445	7.9257	7.4217	7.6862
c3 tiny	-3.1103	-4.4041	-5.1793	-8.4131	-7.2096
c3 small	2.2039	1.5447	1.0121	-0.3998	1.4927
c3 average	0.3021	-1.3939	-1.2201	-3.3407	-2.1467
c3 big	-0.9621	-1.2439	-1.6092	-3.7666	-3.4389
c3 huge	0	0	0	0	0
c1_3*c3 tiny	-21.9104	-21.7840	-22.0173	-22.6066	-21.9791
c1_3*c3 small	-20.8196	-20.2725	-19.5850	-20.4515	-20.7586
c1_3*c3 average	-16.8500	-15.1509	-15.0134	-15.3851	-13.4339
c1_3*c3 big	-12.7212	-12.1554	-12.0354	-12.3282	-13.0174
c1_3*c3 huge	0	0	0	0	0
x1	0.9238	1.7286	2.5976	-0.2488	1.2093
x1*c3 tiny	-1.5819	-1.1748	-3.2523	-1.7016	-2.7624
x1*c3 small	-3.7669	-3.2984	-2.9755	-1.8738	-4.0167
x1*c3 average	2.2253	2.4489	1.5675	4.0948	2.0159
x1*c3 big	0.9222	0.5330	0.7960	2.6061	1.2694
x1*c3 huge	0	0	0	0	0
x5	-1.3562	0.5639	0.3022	-0.4700	-2.5063
x6	-0.9165	-3.2944	-1.2163	-2.2063	-0.5696
x7	5.2295	5.3015	6.2526	4.1770	5.8364
x6*x7	6.4211	7.5644	6.1182	7.0020	5.8730
x10	1.9591	1.4932	0.7196	0.6504	-0.3989
x5*x10	3.6058	1.7274	4.3447	2.4388	3.8967
x15	-0.0079	0.6896	1.6811	0.0136	0.1799
x15*c1_3	-3.5022	-2.7963	-2.6003	-4.2355	-4.7546
x7*x15	-5.1438	-5.8878	-5.9465	-3.6155	-5.3337
x18	-2.1347	-1.5656	-2.4226	-4.0592	-1.4985
x18*c3 tiny	2.2988	1.1931	2.6491	6.1615	5.6204
x18*c3 small	4.6033	3.2359	4.4183	5.5923	1.7270
x18*c3 average	-2.3712	-2.5392	-0.6361	-1.1729	-1.6481
x18*c3 big	2.3160	1.4654	2.7683	3.0487	2.5768
x18*c3 huge	0	0	0	0	0
x6*x18	3.0716	4.2036	4.1354	4.9196	2.7165
x20	4.1229	4.5773	4.5774	4.6555	4.2655

The following statements display the first eight observations in the outData data set.

```
proc print data=outData(obs=8);
run;
```

Output 56.2.17 First Eight Observations in the Output Data Set

Obs	c3	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12
1	tiny	0.18496	0.97009	0.39982	0.25940	0.92160	0.96928	0.54298	0.53169	0.04979	0.06657	0.81932	0.52387
2	tiny	0.47579	0.84499	0.63452	0.59036	0.58258	0.37701	0.72836	0.50660	0.93121	0.92912	0.58966	0.29722
3	tiny	0.51132	0.43320	0.17611	0.66504	0.40482	0.12455	0.45349	0.19955	0.57484	0.73847	0.43981	0.04937
4	tiny	0.42071	0.07174	0.35849	0.71143	0.18985	0.14797	0.56184	0.27011	0.32520	0.56918	0.04259	0.43921
5	tiny	0.42137	0.03798	0.27081	0.42773	0.82010	0.84345	0.87691	0.26722	0.30602	0.39705	0.34905	0.76593
6	tiny	0.81722	0.65822	0.02947	0.85339	0.36285	0.37732	0.51054	0.71194	0.37533	0.22954	0.68621	0.55243
7	tiny	0.19480	0.81673	0.08548	0.18376	0.33264	0.70558	0.92761	0.29642	0.22404	0.14719	0.59064	0.46326
8	tiny	0.04403	0.51697	0.68884	0.45333	0.83565	0.29745	0.40325	0.95684	0.42194	0.78079	0.33106	0.17210

Obs	x13	x14	x15	x16	x17	x18	x19	x20	c1	c2	y	_CVINDEX_	p_y
1	0.85339	0.06718	0.95702	0.29719	0.27261	0.68993	0.97676	0.22651	2	1	11.4391	1	18.1474
2	0.39104	0.47243	0.67953	0.16809	0.16653	0.87110	0.29879	0.93464	3	1	31.4596	2	24.7930
3	0.52238	0.34337	0.02271	0.71289	0.93706	0.44599	0.94694	0.71290	4	3	16.4294	3	16.5752
4	0.91744	0.52584	0.73182	0.90522	0.57600	0.18794	0.33133	0.69887	5	3	15.4815	4	14.7605
5	0.54340	0.61257	0.55291	0.73591	0.37186	0.64565	0.55718	0.87504	6	2	26.0023	5	24.7479
6	0.58182	0.17472	0.04610	0.64380	0.64545	0.09317	0.62008	0.07845	7	1	16.6503	1	21.4444
7	0.41860	0.25631	0.23045	0.08034	0.43559	0.67020	0.42272	0.49827	1	1	14.0342	2	20.9661
8	0.91056	0.26897	0.95602	0.13720	0.27190	0.55692	0.65825	0.68465	2	3	14.9830	3	17.5644

This example demonstrates the usefulness of effect selection when you suspect that interactions of effects are needed to explain the variation in your dependent variable. Ideally, a priori knowledge should be used to decide what interactions to allow, but in some cases this information might not be available. Simply fitting a least squares model allowing all interactions produces a model that overfits your data and generalizes very poorly.

The following statements use forward selection with selection based on the SBC criterion, which is the default selection criterion. At each step, the effect whose addition to the model yields the smallest SBC value is added. The STOP=NONE suboption specifies that this process continue even when the SBC statistic grows whenever an effect is added, and so it terminates at a full least squares model. The BUILDSSCP=FULL option is specified in a PERFORMANCE statement, since building the SSCP matrix incrementally is counterproductive in this case. See the section "BUILDSSCP=FULL | INCREMENTAL" on page 4456 for details. Note that if all you are interested in is a full least squares model, then it is much more efficient to simply specify SELECTION=NONE in the MODEL statement.

However, in this example the aim is to add effects in roughly increasing order of explanatory power.

```
proc glmselect data=analysisData testdata=testData plots=ASEPlot;
   class c1 c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2
           / selection=forward(stop=none)
             hierarchy=single;
   performance buildSSCP = full;
run;
```

```
ods graphics off;
```

The ASE plot shown in Output 56.2.18 clearly demonstrates the danger in overfitting the training data. As more insignificant effects are added to the model, the growth in test set ASE shows how the predictions produced by the resulting models worsen. This decline is particularly rapid in the latter stages of the forward selection, because the use of the SBC criterion results in insignificant effects with lots of parameters being added after insignificant effects with fewer parameters.

Output 56.2.18 Average Squared Error Plot

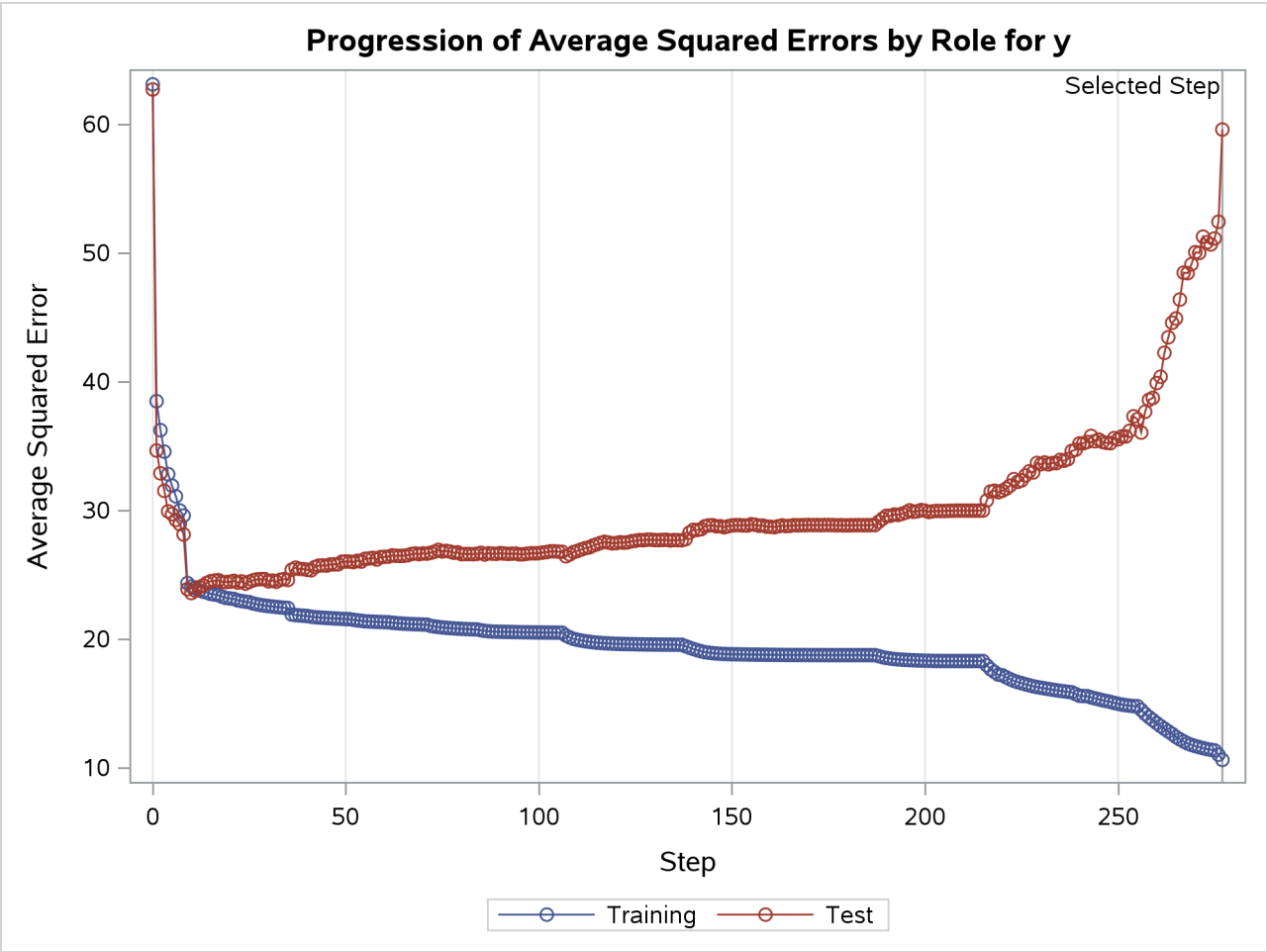

Example 56.3: Scatter Plot Smoothing by Selecting Spline Functions

(View the complete code for this example (glsex3.sas) in the example repository.)

This example shows how you can use model selection to perform scatter plot smoothing. It illustrates how you can use the experimental EFFECT statement to generate a large collection of B-spline basis functions from which a subset is selected to fit scatter plot data.

The data for this example come from a set of benchmarks developed by Donoho and Johnstone (1994) that have become popular in the statistics literature. The particular benchmark used is the "Bumps" functions to which random noise has been added to create the test data. The following DATA step, extracted from Sarle (2001), creates the data. The constants are chosen so that the noise-free data have a standard deviation of 7. The standard deviation of the noise is $\sqrt{5}$, yielding bumpsNoise with a signal-to-noise ratio of 3.13 ($7/\sqrt{5}$).

```
%let random=12345;

data DoJoBumps;
   keep x bumps bumpsWithNoise;

   pi = arcos(-1);

   do n=1 to 2048;
      x=(2*n-1)/4096;
      link compute;
      bumpsWithNoise=bumps+rannor(&random)*sqrt(5);
      output;
   end;
   stop;

compute:
   array t(11) _temporary_ (.1 .13 .15 .23 .25 .4 .44 .65 .76 .78 .81);
   array b(11) _temporary_ (   4    5    3   4   5 4.2 2.1 4.3  3.1  5.1  4.2);
   array w(11) _temporary_ (.005 .005 .006 .01 .01 .03 .01 .01 .005 .008 .005);

   bumps=0;
   do i=1 to 11;
      bumps=bumps+b[i]*(1+abs((x-t[i])/w[i]))**-4;
   end;
   bumps=bumps*10.528514619;
   return;
run;
```

The following statements use the SGPLOT procedure to produce the plot in Output 56.3.1. The plot shows the bumps function superimposed on the function with added noise.

```
proc sgplot data=DoJoBumps;
   yaxis display=(nolabel);
   series  x=x y=bumpsWithNoise/lineattrs=(color=black);
   series  x=x y=bumps/lineattrs=(color=red);
run;
```

Output 56.3.1 Donoho-Johnstone Bumps Function

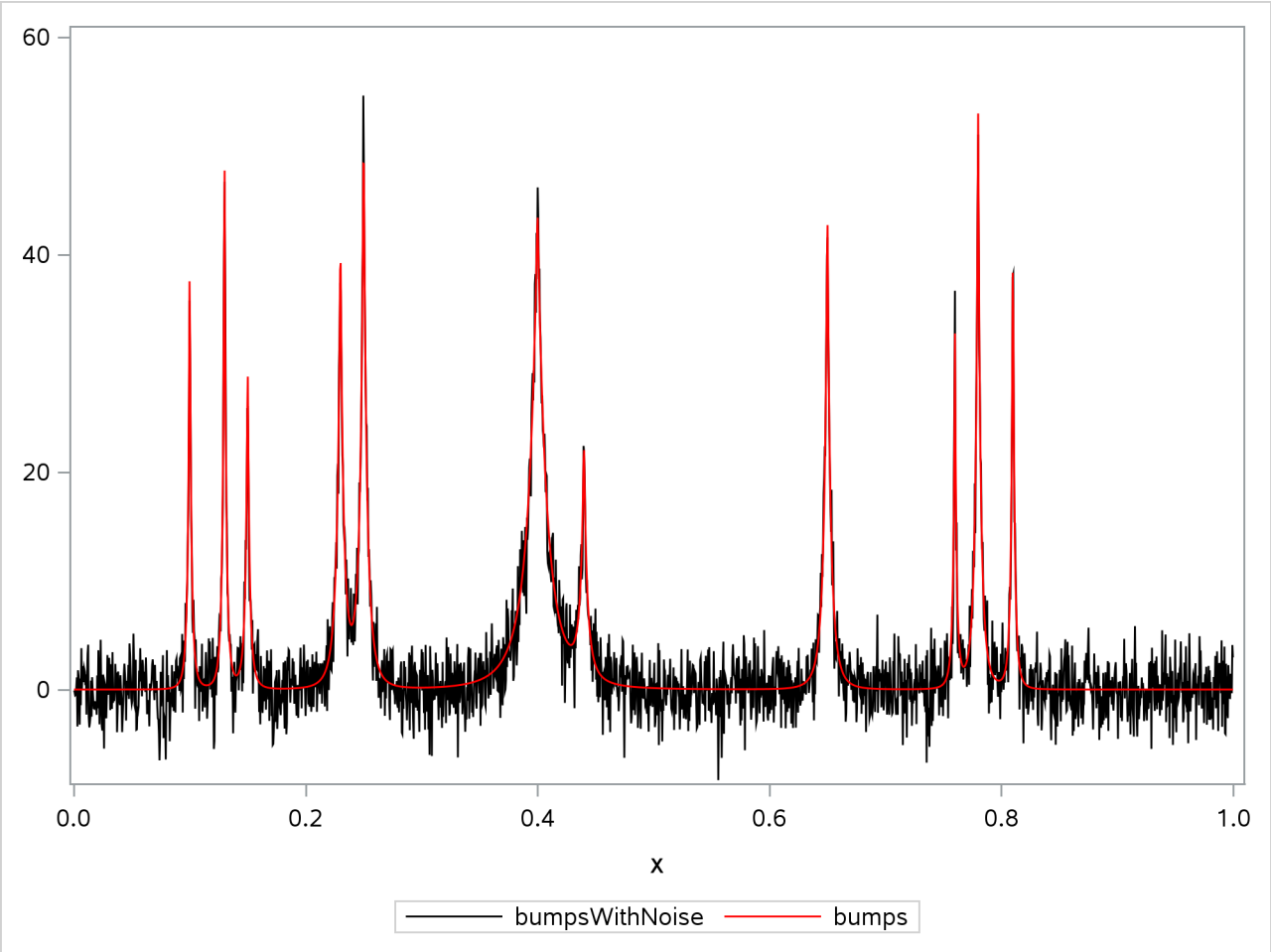

Suppose you want to smooth the noisy data to recover the underlying function. This problem is studied by Sarle (2001), who shows how neural nets can be used to perform the smoothing. The following statements use the LOESS statement in the SGPLOT procedure to show a loess fit superimposed on the noisy data (Output 56.3.2). (See Chapter 78, "The LOESS Procedure," for information about the loess method.)

```
proc sgplot data=DoJoBumps;
   yaxis display=(nolabel);
   series x=x y=bumps;
   loess  x=x y=bumpsWithNoise / lineattrs=(color=red) nomarkers;
run;
```

The algorithm selects a smoothing parameter that is small enough to enable bumps to be resolved. Because there is a single smoothing parameter that controls the number of points for all local fits, the loess method undersmooths the function in the intervals between the bumps.

Output 56.3.2 Loess Fit

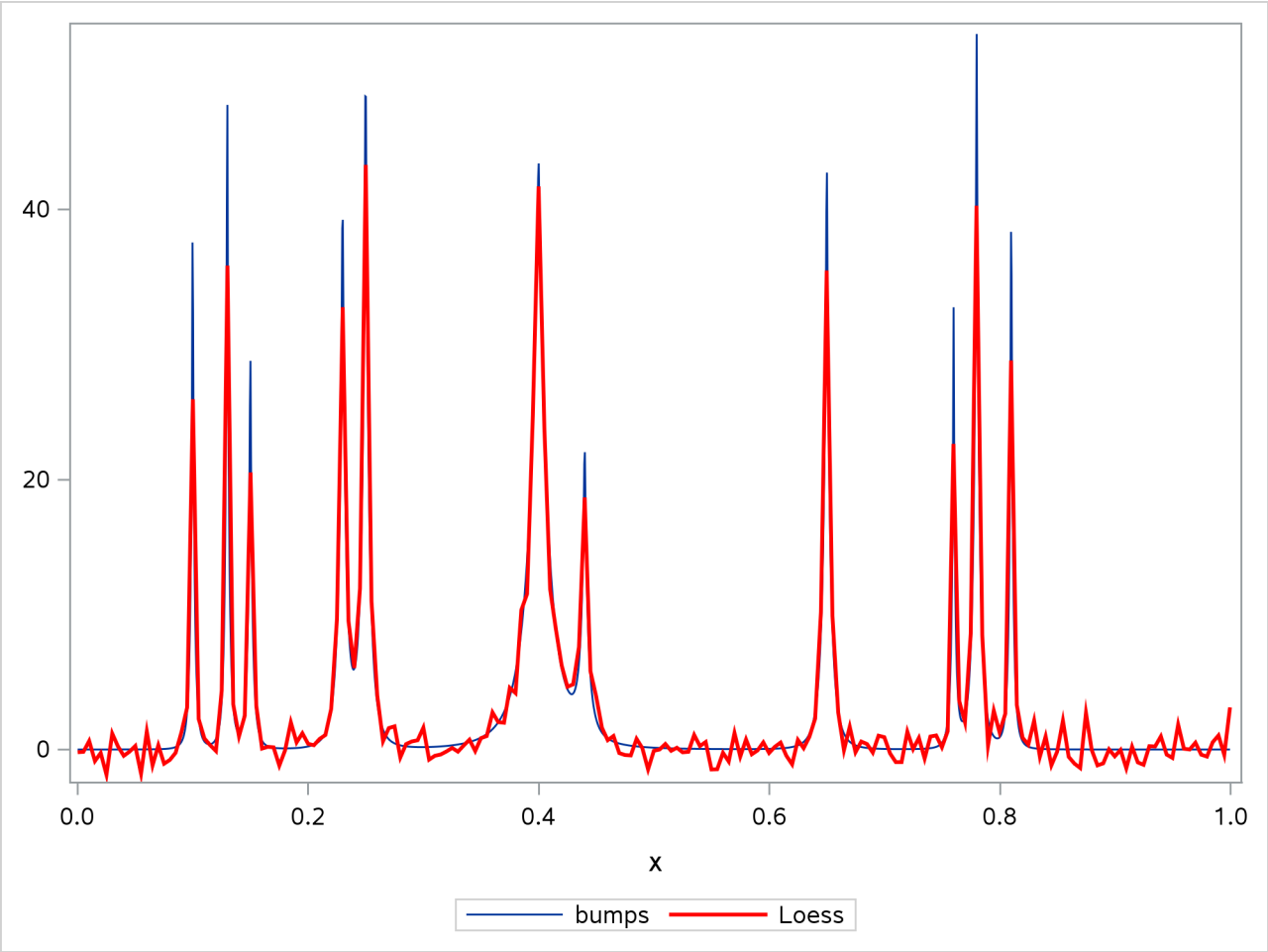

Another approach to doing nonparametric fitting is to approximate the unknown underlying function as a linear combination of a set of basis functions. Once you specify the basis functions, then you can use least squares regression to obtain the coefficients of the linear combination. A problem with this approach is that for most data, you do not know a priori what set of basis functions to use. You need to supply a sufficiently rich set to enable the features in the data to be approximated. However, if you use too rich a set of functions, then this approach yields a fit that undersmooths the data and captures spurious features in the noise.

The penalized B-spline method (Eilers and Marx 1996) uses a basis of B-splines (see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics") corresponding to a large number of equally spaced knots as the set of approximating functions. To control the potential overfitting, their algorithm modifies the least squares objective function to include a penalty term that grows with the complexity of the fit.

The following statements use the PBSPLINE statement in the SGPLOT procedure to show a penalized B-spline fit superimposed on the noisy data (Output 56.3.3). See Chapter 126, "The TRANSREG Procedure," for details about the implementation of the penalized B-spline method.

```
proc sgplot data=DoJoBumps;
   yaxis display=(nolabel);
   series    x=x y=bumps;
   pbspline  x=x y=bumpsWithNoise /
               lineattrs=(color=red) nomarkers;
run;
```

As in the case of loess fitting, you see undersmoothing in the intervals between the bumps because there is only a single smoothing parameter that controls the overall smoothness of the fit.

Output 56.3.3 Penalized B-spline Fit

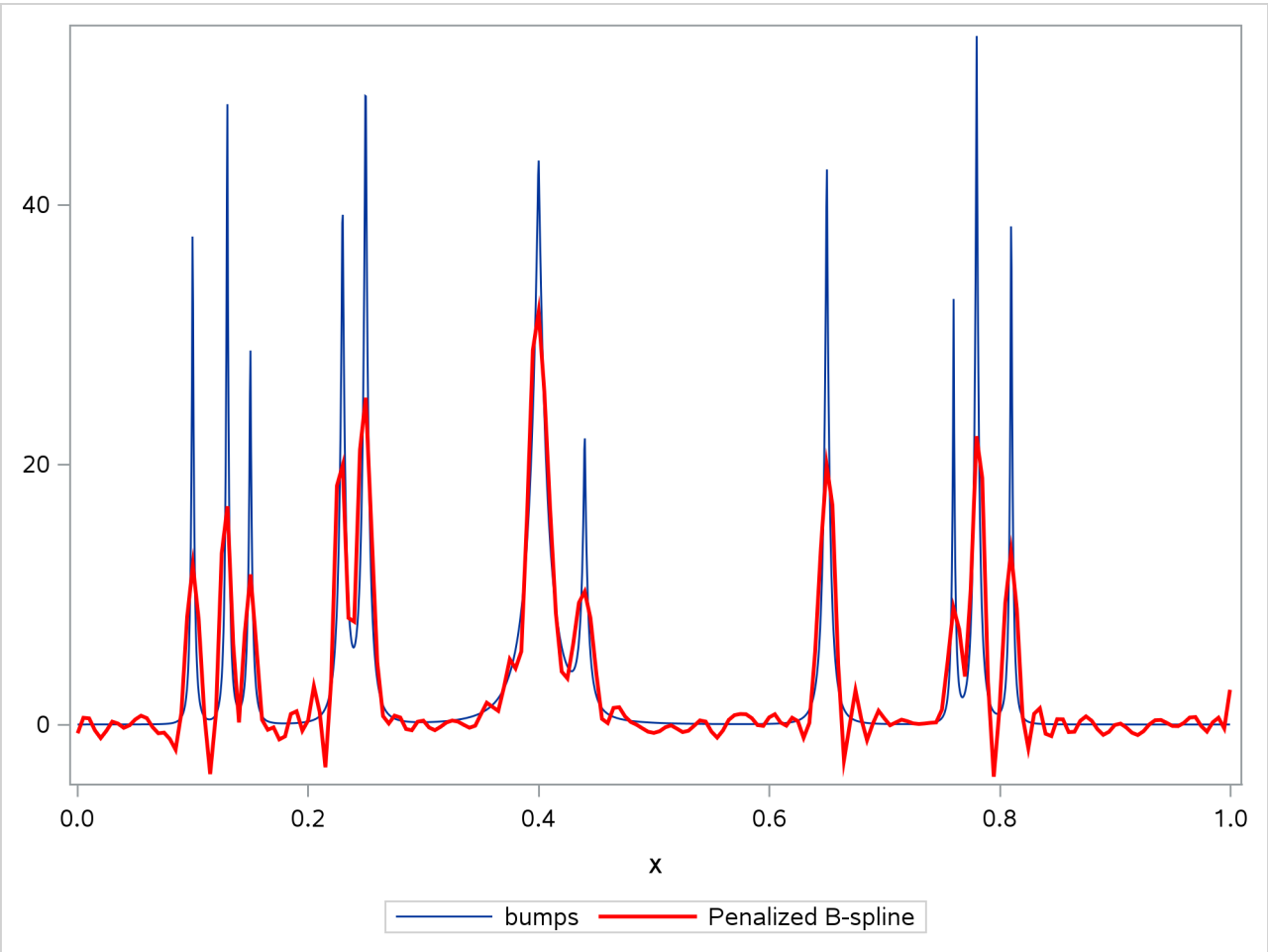

An alternative to using a smoothness penalty to control the overfitting is to use variable selection to obtain an appropriate subset of the basis functions. In order to be able to represent features in the data that occur at multiple scales, it is useful to select from B-spline functions defined on just a few knots to capture large scale features of the data as well as B-spline functions defined on many knots to capture fine details of the data. The following statements show how you can use PROC GLMSELECT to implement this strategy:

```
proc glmselect data=dojoBumps;
   effect spl = spline(x / knotmethod=multiscale(endscale=8)
                           split details);
   model bumpsWithNoise=spl;
   output out=out1 p=pBumps;
run;

proc sgplot data=out1;
   yaxis display=(nolabel);
   series    x=x y=bumps;
   series    x=x y=pBumps / lineattrs=(color=red);
run;
```

The KNOTMETHOD=MULTISCALE suboption of the **EFFECT spl = SPLINE** statement provides a convenient way to generate B-spline basis functions at multiple scales. The ENDSCALE=8 option requests that the finest scale use B-splines defined on 2^8 equally spaced knots in the interval $[0, 1]$. Because the cubic B-splines are nonzero over five adjacent knots, at the finest scale, the support of each B-spline basis function is an interval of length about 0.02 (5/256), enabling the bumps in the underlying data to be resolved. The default value is ENDSCALE=7. At this scale you will still be able to capture the bumps, but with less sharp resolution. For these data, using a value of ENDSCALE= greater than eight provides unneeded resolution, making it more likely that basis functions that fit spurious features in the noise are selected.

Output 56.3.4 shows the model information table. Since no options are specified in the MODEL statement, PROC GLMSELECT uses the stepwise method with selection and stopping based on the SBC criterion.

Output 56.3.4 Model Settings

The GLMSELECT Procedure

Data Set	WORK.DOJOBUMPS
Dependent Variable	bumpsWithNoise
Selection Method	Stepwise
Select Criterion	SBC
Stop Criterion	SBC
Effect Hierarchy Enforced	None

The DETAILS suboption in the EFFECT statement requests the display of spline knots and spline basis tables. These tables contain information about knots and basis functions at all scales. The results for scale four are shown in Output 56.3.5 and Output 56.3.6.

Output 56.3.5 Spline Knots

Knots for Spline Effect spl				
Knot Number	Scale	Scale Knot Number	Boundary	x
40	4	1	*	-0.11735
41	4	2	*	-0.05855
42	4	3	*	0.00024414
43	4	4		0.05904
44	4	5		0.11783
45	4	6		0.17663
46	4	7		0.23542
47	4	8		0.29422
48	4	9		0.35301
49	4	10		0.41181
50	4	11		0.47060
51	4	12		0.52940
52	4	13		0.58819
53	4	14		0.64699
54	4	15		0.70578
55	4	16		0.76458
56	4	17		0.82337
57	4	18		0.88217
58	4	19		0.94096
59	4	20	*	0.99976
60	4	21	*	1.05855
61	4	22	*	1.11735

Output 56.3.6 Spline Details

Basis Details for Spline Effect spl					
Column	Scale	Scale Column	Support		Support Knots
32	4	1	-0.11735	0.05904	1-4
33	4	2	-0.11735	0.11783	1-5
34	4	3	-0.05855	0.17663	2-6
35	4	4	0.00024414	0.23542	3-7
36	4	5	0.05904	0.29422	4-8
37	4	6	0.11783	0.35301	5-9
38	4	7	0.17663	0.41181	6-10
39	4	8	0.23542	0.47060	7-11
40	4	9	0.29422	0.52940	8-12
41	4	10	0.35301	0.58819	9-13
42	4	11	0.41181	0.64699	10-14
43	4	12	0.47060	0.70578	11-15
44	4	13	0.52940	0.76458	12-16
45	4	14	0.58819	0.82337	13-17
46	4	15	0.64699	0.88217	14-18
47	4	16	0.70578	0.94096	15-19
48	4	17	0.76458	0.99976	16-20
49	4	18	0.82337	1.05855	17-21
50	4	19	0.88217	1.11735	18-22
51	4	20	0.94096	1.11735	19-22

The Dimensions table in Output 56.3.7 shows that at each step of the selection process, 548 effects are considered as candidates for entry or removal. Note that although the MODEL statement specifies a single constructed effect spl, the SPLIT suboption causes each of the parameters in this constructed effect to be treated as an individual effect.

Output 56.3.7 Dimensions

Dimensions	
Number of Effects	548
Number of Parameters	548

Output 56.3.8 shows the parameter estimates for the selected model. You can see that the selected model contains 31 B-spline basis functions and that all the selected B-spline basis functions are from scales four though eight. For example, the first basis function listed in the parameter estimates table is spl_S4:9—the ninth B-spline function at scale 4. You see from Output 56.3.6 that this function is nonzero on the interval $(0.29, 0.52)$.

Output 56.3.8 Parameter Estimates

Parameter Estimates				
Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	-0.009039	0.077412	-0.12
spl_S4:9	1	7.070207	0.586990	12.04
spl_S5:10	1	5.323121	1.199824	4.44
spl_S6:17	1	5.222808	1.728910	3.02
spl_S6:28	1	24.562103	1.490639	16.48
spl_S6:44	1	4.930829	1.243552	3.97
spl_S6:52	1	-7.046308	2.487700	-2.83
spl_S7:86	1	9.592742	2.626471	3.65
spl_S7:106	1	16.268550	3.334015	4.88
spl_S8:27	1	10.626586	1.752152	6.06
spl_S8:28	1	27.882444	2.004520	13.91
spl_S8:29	1	-6.129939	1.752151	-3.50
spl_S8:33	1	5.855648	1.766912	3.31
spl_S8:34	1	-11.782303	2.092484	-5.63
spl_S8:35	1	38.705178	2.092486	18.50
spl_S8:36	1	13.823256	1.766916	7.82
spl_S8:40	1	15.975124	1.691679	9.44
spl_S8:41	1	14.898716	1.691679	8.81
spl_S8:61	1	37.441965	2.084375	17.96
spl_S8:66	1	47.484506	1.883409	25.21
spl_S8:67	1	16.811502	1.910358	8.80
spl_S8:104	1	11.098484	1.958676	5.67
spl_S8:105	1	26.704556	2.042735	13.07
spl_S8:115	1	21.102920	1.576185	13.39
spl_S8:169	1	36.572294	2.914521	12.55
spl_S8:197	1	20.869716	1.882529	11.09
spl_S8:198	1	16.210987	2.693183	6.02
spl_S8:200	1	13.113942	3.458187	3.79
spl_S8:202	1	38.463549	2.462314	15.62
spl_S8:203	1	34.164644	1.757908	19.43
spl_S8:209	1	-22.645471	3.598587	-6.29
spl_S8:210	1	29.024741	2.557567	11.35

The OUTPUT statement captures the predicted values in a data set named out1, and Output 56.3.9 shows a fit plot produced by PROC SGPLOT.

Output 56.3.9 Fit by Selecting B-splines

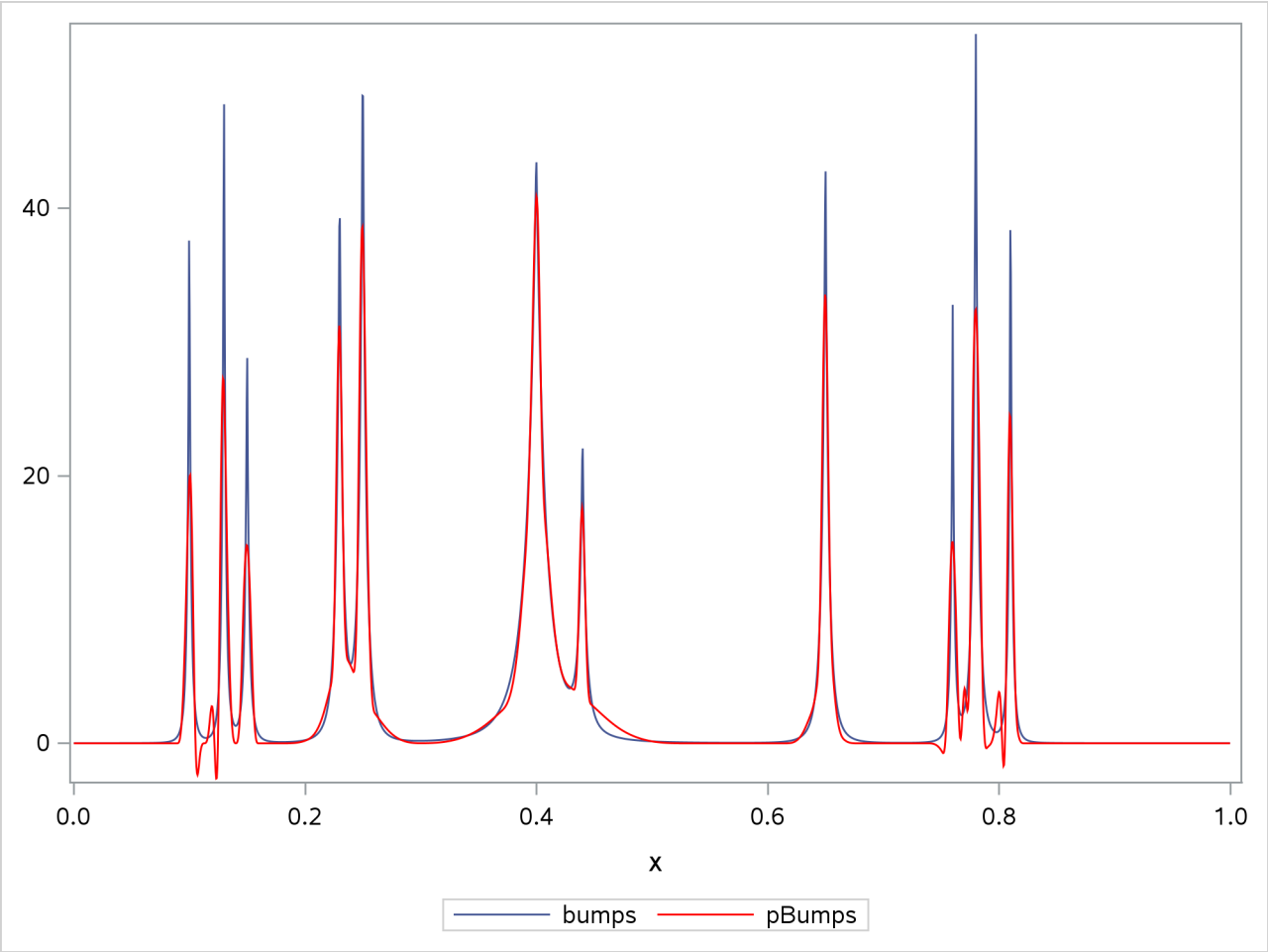

Example 56.4: Multimember Effects and the Design Matrix

(View the complete code for this example (glsex4.sas) in the example repository.)

This example shows how you can use multimember effects to build predictive models. It also demonstrates several features of the OUTDESIGN= option in the PROC GLMSELECT statement.

The simulated data for this example describe a two-week summer tennis camp. The tennis ability of each camper was assessed and ratings were assigned at the beginning and end of the camp. The camp consisted of supervised group instruction in the mornings with a number of different options in the afternoons. Campers could elect to participate in unsupervised practice and play. Some campers paid for one or more individual lessons from 30 to 90 minutes in length, focusing on forehand and backhand strokes and volleying. The aim of this example is to build a predictive model for the rating improvement of each camper based on the times the camper spent doing each activity and several other variables, including the age, gender, and initial rating of the camper.

The following statements produce the TennisCamp data set:

```
data TennisCamp;
   length forehandCoach $6 backhandCoach $6 volleyCoach $6 gender $1;
   input forehandCoach backhandCoach volleyCoach tLessons tPractice tPlay
         gender inRating nPastCamps age tForehand tBackhand tVolley
         improvement;
   label forehandCoach = "Forehand lesson coach"
         backhandCoach = "Backhand lesson coach"
         volleyCoach   = "Volley   lesson coach"
         tForehand     = "time (1/2 hours) of forehand lesson"
         tBackhand     = "time (1/2 hours) of backhand lesson"
         tVolley       = "time (1/2 hours) of volley lesson"
         tLessons      = "time (1/2 hours) of all lessons"
         tPractice     = "total practice time (hours)"
         tPlay         = "total play time (hours)"
         nPastCamps    = "Number of previous camps attended"
         age           = "age (years)"
         inRating      = "Rating at camp start"
         improvement   = "Rating improvement at end of camp";
   datalines;
.        .        Tom     1   30   19   f   44   0   13   0   0   1    6
Greg     .        .       2   12   33   f   48   2   15   2   0   0   14
.        .        Mike    2   12   24   m   53   0   15   0   0   2   13
.        Mike     .       1   12   28   f   48   0   13   0   1   0   11
.        Bruna    .       2   13   34   f   57   0   16   0   2   0   12

   ... more lines ...

.        .        .       0   12   38   m   47   1   15   0   0   0    8
Greg     Tom      Tom     6    3   41   m   48   2   15   2   1   3   19
.        Greg     Mike    5   30   16   m   52   0   13   0   2   3   18
;
```

A multimember effect (see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics") is appropriate for modeling the effect of coaches on the campers' improvement, because campers might have worked with multiple coaches. Furthermore, since the time a coach spent with each camper varies, it is appropriate to use these times to weight each coach's contribution in the multimember effect. It is also important not to exclude campers from the analysis if they did not receive any individual instruction. You can accomplish all these goals by using a multimember effect defined as follows:

```
class forehandCoach backhandCoach volleyCoach;
effect coach = MM(forehandCoach backhandCoach volleyCoach/ noeffect
                  weight=(tForehand tBackhand tVolley));
```

Based on similar previous studies, it is known that the time spent practicing should not be included linearly, because there are diminishing returns and perhaps even counterproductive effects beyond about 25 hours. A spline effect with a single knot at 25 provides flexibility in modeling effect of practice time.

The following statements use PROC GLMSELECT to select effects for the model.

```
proc glmselect data=TennisCamp outdesign=designCamp;
   class forehandCoach backhandCoach volleyCoach gender;

   effect coach    = mm(forehandCoach backhandCoach volleyCoach / noeffect
```

```
                        details weight=(tForehand tBackhand tVolley));
    effect practice = spline(tPractice/knotmethod=list(25) details);

    model improvement = coach practice tLessons tPlay age gender
                        inRating nPastCamps;
  run;
```

Output 56.4.1 shows the class level and MM level information. The levels of the constructed MM effect are the union of the levels of its constituent classification variables. The MM level information is not displayed by default—you request this table by specifying the DETAILS suboption in the relevant EFFECT statement.

Output 56.4.1 Levels of MM EFFECT Coach

The GLMSELECT Procedure

Class Level Information		
Class	**Levels**	**Values**
forehandCoach	5	Bruna Elaine Greg Mike Tom
backhandCoach	5	Bruna Elaine Greg Mike Tom
volleyCoach	5	Andy Bruna Greg Mike Tom
gender	2	f m

The GLMSELECT Procedure

Level Details for MM Effect coach	
Levels	**Values**
6	Andy Bruna Elaine Greg Mike Tom

Output 56.4.2 shows the parameter estimates for the selected model. You can see that the constructed multimember effect coach and the spline effect practice are both included in the selected model. All coaches provided benefit (all the parameters of the multimember effect coach are positive), with Greg and Mike being the most effective.

Output 56.4.2 Parameter Estimates

Parameter Estimates					
Parameter		**DF**	**Estimate**	**Standard Error**	**t Value**
Intercept		1	0.379873	0.513431	0.74
coach	**Andy**	1	1.444370	0.318078	4.54
coach	**Bruna**	1	1.446063	0.110179	13.12
coach	**Elaine**	1	1.312290	0.281877	4.66
coach	**Greg**	1	3.042828	0.112256	27.11
coach	**Mike**	1	2.840728	0.121166	23.45
coach	**Tom**	1	1.248946	0.115266	10.84
practice	**1**	1	2.538938	1.015772	2.50
practice	**2**	1	3.837684	1.104557	3.47
practice	**3**	1	2.574775	0.930816	2.77
practice	**4**	1	-0.034747	0.717967	-0.05
practice	**5**	0	0	.	.
tPlay		1	0.139409	0.023043	6.05

Suppose you want to examine regression diagnostics for the selected model. PROC GLMSELECT does not support such diagnostics, so you might want to use the REG procedure to produce these diagnostics. You can overcome the difficulty that PROC REG does not support CLASS and EFFECT statements by using the OUTDESIGN= option in the PROC GLMSELECT statement to obtain the design matrix that you can use as an input data set for further analysis with other SAS procedures.

The following statements use PROC PRINT to produce Output 56.4.3, which shows the first five observations of the design matrix designCamp.

```
proc print data=designCamp(obs=5);
run;
```

Output 56.4.3 First Five Observations of the designCamp Data Set

Obs	Intercept	coach_Andy	coach_Bruna	coach_Elaine	coach_Greg	coach_Mike	coach_Tom	tPlay
1	1	0	0	0	0	0	1	19
2	1	0	0	0	2	0	0	33
3	1	0	0	0	0	2	0	24
4	1	0	0	0	0	1	0	28
5	1	0	2	0	0	0	0	34

Obs	practice_1	practice_2	practice_3	practice_4	practice_5	improvement
1	0.00000	0.00136	0.05077	0.58344	0.36443	6
2	0.20633	0.50413	0.25014	0.03940	0.00000	14
3	0.20633	0.50413	0.25014	0.03940	0.00000	13
4	0.20633	0.50413	0.25014	0.03940	0.00000	11
5	0.16228	0.49279	0.29088	0.05405	0.00000	12

To facilitate specifying the columns of the design matrix corresponding to the selected model, you can use the macro variable named _GLSMOD that PROC GLMSELECT creates whenever you specify the OUTDESIGN= option. The following statements use PROC REG to produce a panel of regression diagnostics corresponding to the model selected by PROC GLMSELECT.

```
ods graphics on;

proc reg data=designCamp;
   model improvement = &_GLSMOD;
quit;

ods graphics off;
```

The regression diagnostics shown in Output 56.4.4 indicate a reasonable model. However, they also reveal the presence of one large outlier and several influential observations that you might want to investigate.

Output 56.4.4 Fit Diagnostics

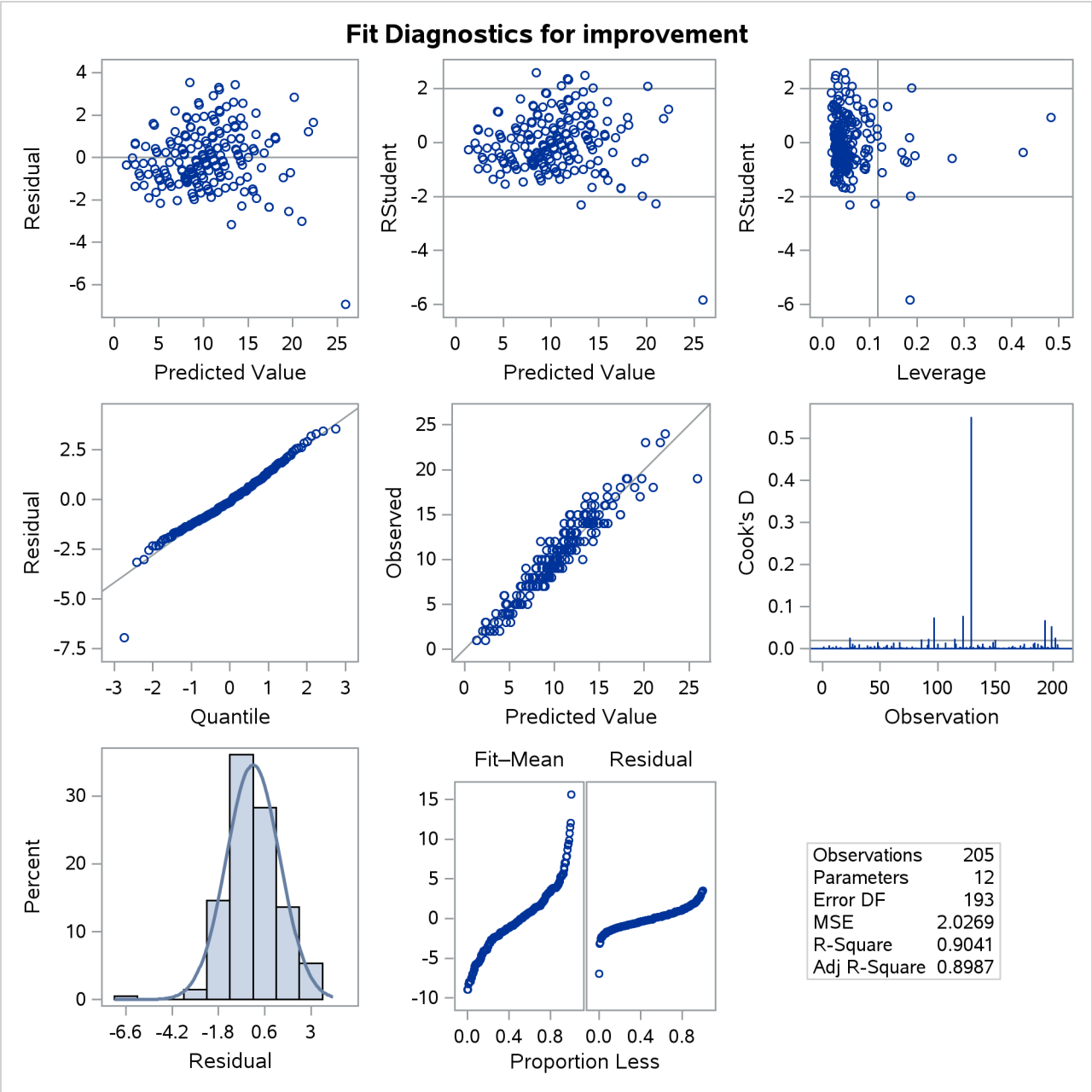

Sometimes you might want to use subsets of the columns of the design matrix. In such cases, it might be convenient to produce a design matrix with generic names for the columns. You might also want a design matrix containing the columns corresponding to the full model that you specify in the MODEL statement. By default, the design matrix includes only the columns that correspond to effects in the selected model. The following statements show how to do this.

```
proc glmselect data=TennisCamp
     outdesign(fullmodel prefix=parm names)=designCampGeneric;
   class forehandCoach backhandCoach volleyCoach gender;
```

```
   effect coach    = mm(forehandCoach backhandCoach volleyCoach / noeffect
                        details weight=(tForehand tBackhand tVolley));
   effect practice = spline(tPractice/knotmethod=list(25) details);

   model improvement = coach practice tLessons tPlay age gender
                       inRating nPastCamps;
run;
```

The PREFIX=parm suboption of the OUTDESIGN= option specifies that columns in the design matrix be given the prefix parm with a trailing index. The NAMES suboption requests the table in Output 56.4.5 that associates descriptive labels with the names of columns in the design matrix. Finally, the FULLMODEL suboption specifies that the design matrix include columns corresponding to all effects specified in the MODEL statement.

Output 56.4.5 Descriptive Names of Design Matrix Columns

The GLMSELECT Procedure
Selected Model

Parameter Names	
Name	**Parameter**
parm1	Intercept
parm2	coach Andy
parm3	coach Bruna
parm4	coach Elaine
parm5	coach Greg
parm6	coach Mike
parm7	coach Tom
parm8	practice 1
parm9	practice 2
parm10	practice 3
parm11	practice 4
parm12	practice 5
parm13	tLessons
parm14	tPlay
parm15	age
parm16	gender f
parm17	gender m
parm18	inRating
parm19	nPastCamps

The following statements produce Output 56.4.6, displaying the first five observations of the designCampGeneric data set:

```
proc print data=designCampGeneric(obs=5);
run;
```

Output 56.4.6 First Five Observations of designCampGeneric Data Set

Obs	parm1	parm2	parm3	parm4	parm5	parm6	parm7	parm8	parm9	parm10	parm11	parm12
1	1	0	0	0	0	0	1	0.00000	0.00136	0.05077	0.58344	0.36443
2	1	0	0	0	2	0	0	0.20633	0.50413	0.25014	0.03940	0.00000
3	1	0	0	0	0	2	0	0.20633	0.50413	0.25014	0.03940	0.00000
4	1	0	0	0	0	1	0	0.20633	0.50413	0.25014	0.03940	0.00000
5	1	0	2	0	0	0	0	0.16228	0.49279	0.29088	0.05405	0.00000

Obs	parm13	parm14	parm15	parm16	parm17	parm18	parm19	improvement
1	1	19	13	1	0	44	0	6
2	2	33	15	1	0	48	2	14
3	2	24	15	0	1	53	0	13
4	1	28	13	1	0	48	0	11
5	2	34	16	1	0	57	0	12

Example 56.5: Model Averaging

(View the complete code for this example (glsex5.sas) in the example repository.)

This example shows how you can combine variable selection methods with model averaging to build parsimonious predictive models. This example uses simulated data that consist of observations from the model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + N(0, \sigma^2)$$

where $\mathbf{X}$ is drawn from a multivariate normal distribution $N(0, \mathbf{V})$ with $V_{i,j} = \rho^{|i-j|}$ where $0 < \rho < 1$. This setup has been widely studied in investigations of variable selection methods. For examples, see Breiman (1992); Tibshirani (1996); Zou (2006).

The following statements define a macro that uses the SIMNORMAL procedure to generate the regressors. This macro prepares a TYPE=CORR data set that specifies the desired pairwise correlations. This data set is used as the input data for PROC SIMNORMAL which produces the sampled regressors in an output data set named Regressors.

```
%macro makeRegressorData(nObs=100,nVars=8,rho=0.5,seed=1);
   data varCorr;
      drop i j;
      array x{&nVars};
      length  _NAME_ $8   _TYPE_ $8;
      _NAME_ = '';

      _TYPE_ = 'MEAN';
      do j=1 to &nVars; x{j}=0; end;
      output;

      _TYPE_ = 'STD';
      do j=1 to &nVars; x{j}=1;end;
      output;

      _TYPE_ = 'N';
```

```
      do j=1 to &nVars; x{j}=10000;end;
      output;

      _TYPE_ = 'CORR';
      do i=1 to &nVars;
         _NAME_="x" || trim(left(i));
         do j= 1 to &nVars;
            x{j}=&rho**(abs(i-j));
         end;
         output;
      end;
   run;

   proc simnormal data=varCorr(type=corr) out=Regressors
               numReal=&nObs seed=&seed;
      var x1-x&nVars;
   run;
%mend;
```

The following statements use the %makeRegressorData macro to generate a sample of 100 observations with 10 regressors, where $E(\mathbf{X}_i\mathbf{X}_j) = 0.5^{|i-j|}$, the true coefficients are $\boldsymbol{\beta}' = (3, 1.5, 0, 0, 2, 0, 0, 0, 0, 0)$, and $\sigma = 3$.

```
%makeRegressorData(nObs=100,nVars=10,rho=0.5);

data simData;
   set regressors;
   yTrue = 3*x1 + 1.5*x2 + 2*x5;
   y     = yTrue + 3*rannor(2);
run;
```

The adaptive LASSO algorithm (see "Adaptive LASSO Selection" on page 4465) is a modification of the standard LASSO algorithm in which weights are applied to each of the parameters in forming the LASSO constraint. Zou (2006) shows that the adaptive LASSO has theoretical advantages over the standard LASSO. Furthermore, simulation studies show that the adaptive LASSO tends to perform better than the standard LASSO in selecting the correct regressors, particularly in high signal-to-noise ratio cases. The following statements fit an adaptive LASSO model to the simData data:

```
proc glmselect data=simData;
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=sbc);
run;
```

The selected model and parameter estimates are shown in Output 56.5.1

Output 56.5.1 Model Selected by Adaptive Lasso

The GLMSELECT Procedure
Selected Model

Effects:	Intercept x1 x2 x5

Output 56.5.1 *continued*

Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	-0.243219
x1	1	3.246129
x2	1	1.310514
x5	1	2.132416

You see that the selected model contains only the relevant regressors x1, x2, and x5. You might want to investigate how frequently the adaptive LASSO method selects just the relevant regressors and how stable the corresponding parameter estimates are. In a simulation study, you can do this by drawing new samples and repeating this process many times. What can you do when you only have a single sample of the data available? One approach is to repeatedly draw subsamples from the data that you have, and to fit models for each of these samples. You can then form the average model and use this model for prediction. You can also examine how frequently models are selected, and you can use the frequency of effect selection as a measure of effect importance.

The following statements show how you can use the MODELAVERAGE statement to perform such an analysis:

```
ods graphics on;

proc glmselect data=simData seed=3 plots=(EffectSelectPct ParmDistribution);
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=SBC);
   modelAverage tables=(EffectSelectPct(all) ParmEst(all));
run;
```

The ModelAverageInfo table in Output 56.5.2 shows that the default sampling method is the bootstrap approach of drawing 100 samples with replacement, where the sampling percentage of 100 means that each sample has the same sum of frequencies as the input data. You can use the SAMPLING= and NSAMPLES= options in the MODELAVERAGE statement to modify the sampling method and the number of samples used.

Output 56.5.2 Model Averaging Information

The GLMSELECT Procedure

Model Averaging Information	
Sampling Method	Unrestricted (with replacement)
Sample Percentage	100
Number of Samples	100

Output 56.5.3 shows the percentage of samples where each effect is in the selected model. The ALL option of the EFFECTSELECTPCT request in the TABLES= option specifies that effects that appear in any selected model are shown. (By default, the "Effect Selection Percentage" table displays only those effects that are selected at least 20 percent of the time.)

Output 56.5.3 Effect Selection Percentages

Effect Selection Percentage	
Effect	Selection Percentage
x1	100.0
x2	99.00
x3	33.00
x4	7.00
x5	100.0
x6	14.00
x7	25.00
x8	9.00
x9	7.00
x10	16.00

The EFFECTSELECTPCT request in the PLOTS= option in the PROC GLMSELECT statement produces the bar chart shown in Output 56.5.4, which graphically displays the information in the EffectSelectPct table.

Output 56.5.4 Effect Selection Percentages

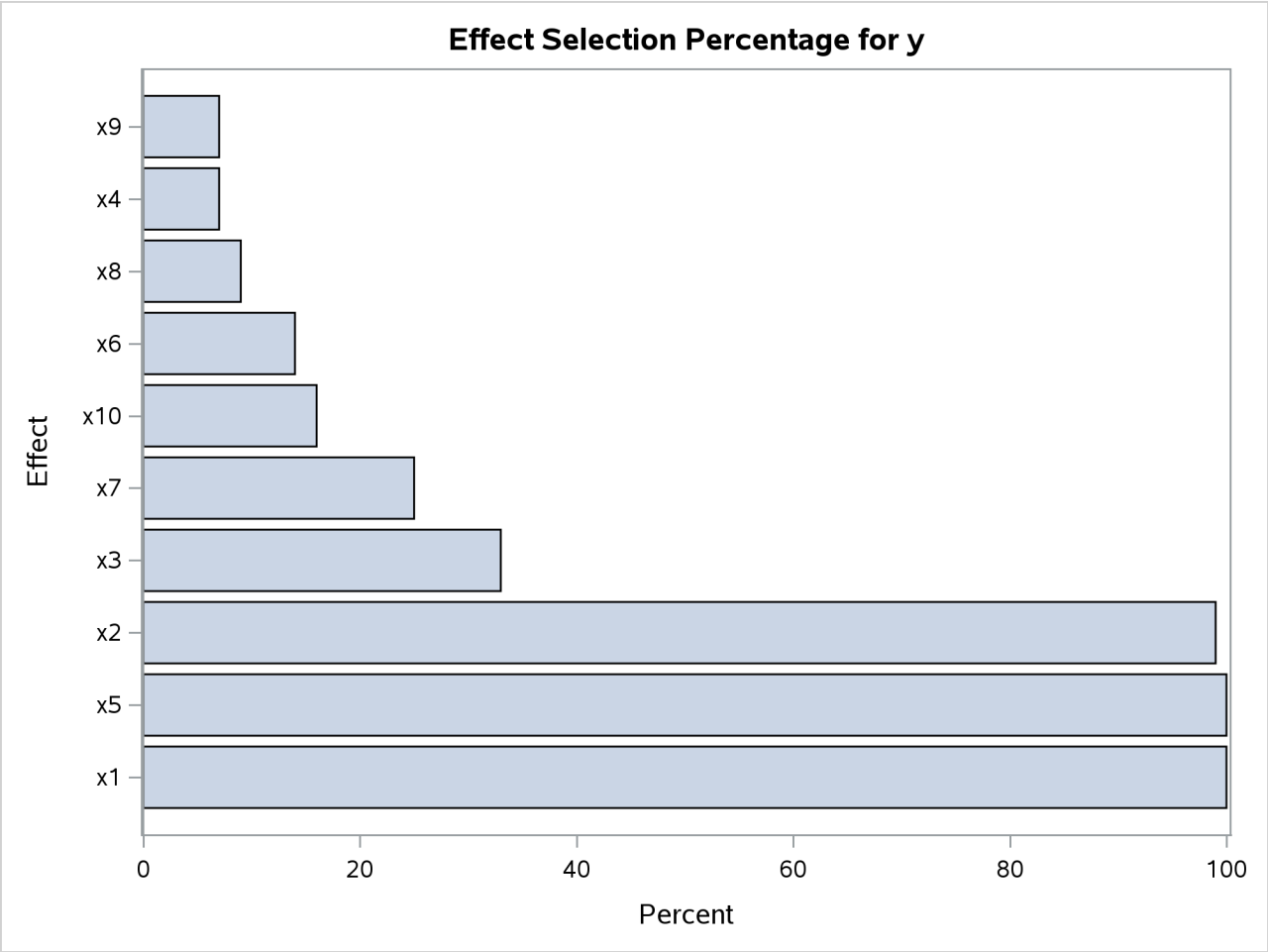

Output 56.5.5 shows the frequencies with which models get selected. By default, only the "best" 20 models are shown. See the section "Model Selection Frequencies and Frequency Scores" on page 4477 for details about how these models are ordered.

Output 56.5.5 Model Selection Frequency

Model Selection Frequency				
Times Selected	Selection Percentage	Number of Effects	Frequency Score	Effects in Model
44	44.00	4	45.00	Intercept x1 x2 x5
9	9.00	5	9.86	Intercept x1 x2 x3 x5
8	8.00	6	8.76	Intercept x1 x2 x3 x5 x7
4	4.00	5	4.82	Intercept x1 x2 x5 x8
4	4.00	7	4.67	Intercept x1 x2 x3 x5 x6 x7
3	3.00	5	3.85	Intercept x1 x2 x5 x7
2	2.00	5	2.83	Intercept x1 x2 x5 x10
2	2.00	5	2.81	Intercept x1 x2 x4 x5
2	2.00	6	2.74	Intercept x1 x2 x3 x5 x6
2	2.00	7	2.66	Intercept x1 x2 x3 x5 x6 x10
1	1.00	5	1.83	Intercept x1 x2 x5 x6
1	1.00	4	1.81	Intercept x1 x5 x7
1	1.00	5	1.81	Intercept x1 x2 x5 x9
1	1.00	6	1.75	Intercept x1 x2 x3 x5 x10
1	1.00	6	1.74	Intercept x1 x2 x3 x5 x8
1	1.00	6	1.73	Intercept x1 x2 x5 x6 x7
1	1.00	6	1.72	Intercept x1 x2 x5 x7 x9
1	1.00	6	1.71	Intercept x1 x2 x5 x8 x10
1	1.00	6	1.70	Intercept x1 x2 x4 x5 x10
1	1.00	7	1.68	Intercept x1 x2 x3 x5 x7 x10

You can see that the most frequently selected model is the model that contains just the true underlying regressors. Although this model is selected in 44% of the samples, most of the selected models contain at least one irrelevant regressor. This is not surprising because even though the true model has just a few large effects in this example, the regressors have nontrivial pairwise correlations.

When your goal is simply to obtain a predictive model, then a good strategy is to average the predictions that you get from all the selected models. In the linear model context, this corresponds to using the model whose parameter estimates are the averages of the estimates that you get for each sample, where if a parameter is not in a selected model, the corresponding estimate is defined to be zero. This has the effect of shrinking the estimates of infrequently selected parameters towards zero.

Output 56.5.6 shows the average parameter estimates. The ALL option of the PARMEST request in the TABLES= option specifies that all parameters that are nonzero in any selected model are shown. (By default, the "Average Parameter Estimates" table displays only those parameters that are nonzero in at least 20 percent of the selected models.)

Output 56.5.6 Average Parameter Estimates

Average Parameter Estimates							
					Estimate Quantiles		
Parameter	Number Non-zero	Non-zero Percentage	Mean Estimate	Standard Deviation	25%	Median	75%
Intercept	100	100.00	-0.271262	0.308146	-0.489061	-0.249163	-0.058233
x1	100	100.00	3.196392	0.377771	2.951551	3.189078	3.446055
x2	99	99.00	1.439966	0.416054	1.209781	1.484064	1.710275
x3	33	33.00	-0.264831	0.412148	-0.536449	0	0
x4	7	7.00	-0.037810	0.142932	0	0	0
x5	100	100.00	2.253196	0.397032	2.036261	2.242240	2.489068
x6	14	14.00	-0.083823	0.261641	0	0	0
x7	25	25.00	0.184656	0.372813	0	0	0.143317
x8	9	9.00	0.060438	0.206621	0	0	0
x9	7	7.00	-0.043307	0.239940	0	0	0
x10	16	16.00	0.083411	0.199573	0	0	0

The average estimate for a parameter is computed by dividing the sum of the estimate values for that parameter in each sample by the total number of samples. This corresponds to using zero as the estimate value for the parameter in those samples where the parameter does not appear in the selected model. Similarly, these zero estimates are included in the computation of the estimated standard deviation and quantiles that are displayed in the AvgParmEst table. If you want see the estimates that you get if you do not use zero for nonselected parameters, you can specify the NONZEROPARMS suboption of the PARMEST request in the TABLES=option.

The PARMDISTRIBUTION request in the PLOTS= option in the PROC GLMSELECT statement requests the panel in Output 56.5.7, which shows the distribution of the estimates for each parameter in the average model. For each parameter in the average model, a histogram and box plot of the nonzero values of the estimates are shown. You can use this plot to assess how the selected estimates vary across the samples.

Output 56.5.7 Effect Selection Percentages

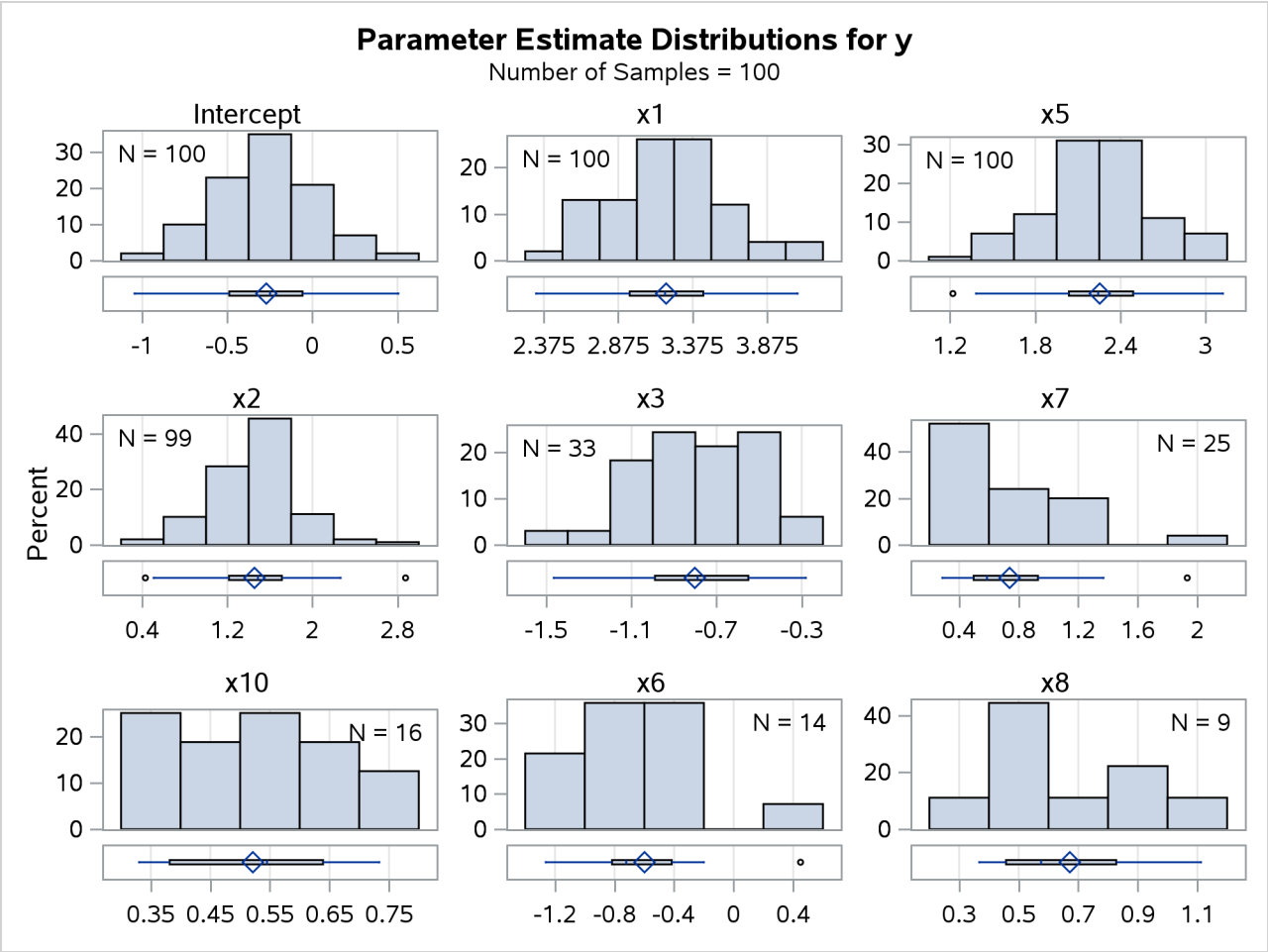

You can obtain details about the model selection for each sample by specifying the DETAILS option in the MODELAVERAGE statement. You can use an OUTPUT statement to output the mean predicted value and standard deviation, quantiles of the predicted values, as well as the individual sample frequencies and predicted values for each sample. The following statements show how you do this:

```
proc glmselect data=simData seed=3;
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=SBC);
   modelAverage details;
   output out=simOut sampleFreq=sf samplePred=sp
                     p=p stddev=stddev lower=q25 upper=q75 median;
run;
```

Output 56.5.8 shows the selection summary and parameter estimates for sample 1 of the model averaging. Note that you can obtain all the model selection output, including graphs, for each sample.

Output 56.5.8 Selection Details for Sample 1

The GLMSELECT Procedure
Sample 1

LASSO Selection Summary				
Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	374.8287
1	x1		2	243.4087
2	x5		3	227.5991
3	x2		4	225.7356*
4	x7		5	229.9135
5	x3		6	233.3660
6	x6		7	237.7447
7	x10		8	235.2171
8	x4		9	238.8085
9	x9		10	239.8353
10	x8		11	244.4236
* Optimal Value of Criterion				

Parameter Estimates		
Parameter	DF	Estimate
Intercept	1	-0.092885
x1	1	4.079938
x2	1	0.505697
x5	1	1.473929

The following statements display the subset of the variables in the first five observations of the output data set, as shown in Output 56.5.9.

```
proc print data=simOut(obs=5);
   var p stddev q25 median q75 sf1-sf3 sp1-sp3;
run;
```

Output 56.5.9 Part of the Output Data Set

Obs	p	stddev	q25	median	q75	sf1	sf2	sf3	sp1	sp2	sp3
1	10.3569	0.82219	9.95992	10.3878	10.9194	1	0	1	10.1378	11.2104	11.0124
2	-5.5453	0.64544	-6.05563	-5.6455	-5.0829	1	1	1	-4.7517	-6.7191	-6.4413
3	6.5066	0.75289	6.05984	6.5077	6.9099	3	2	0	6.0838	7.4880	6.3466
4	-1.7527	0.85168	-2.26638	-1.8123	-1.3312	1	1	2	-2.1891	-1.4887	-1.7083
5	-7.5840	1.20687	-8.44679	-7.5716	-6.7386	3	1	1	-6.7051	-9.0558	-6.7949

By default, the LOWER and UPPER options in the OUTPUT statement produce the lower and upper quartiles of the sample predicted values. You can change the quantiles produced by using the ALPHA= option in the MODELAVERAGE statement. The variables sf1–sf100 contain the sample frequencies used for each sample, and the variables sp1–sp100 hold the corresponding predicted values. Even if you do not specify the DETAILS option in the MODELAVERAGE statement, you can use the sample frequencies in the output data set to reproduce the selection results for any particular sample. For example, the following statements recover the selection for sample 1:

```
proc glmselect data=simOut;
   freq sf1;
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=SBC);
run;
```

The average model is not parsimonious—it includes shrunken estimates of infrequently selected parameters which often correspond to irrelevant regressors. It is tempting to ignore the estimates of infrequently selected parameters by setting their estimate values to zero in the average model. However, this can lead to a poorly performing model. Even though a parameter might occur in only one selected model, it might be a very important term in that model. Ignoring its estimate but including some of the estimates of the other parameters in this model leads to biased predictions. One scenario where this might occur is when the data contains two highly correlated regressors which are both strongly correlated with the response.

You can obtain a parsimonious model by using the frequency of effect selection as a measure of effect importance and refitting a model that contains just the effects that you deem important. In this example, Output 56.5.3 shows that the effects x1, x2, and x5 all get selected at least 99 percent of the time, whereas all other effects get selected less than 34 percent of the time. This large gap suggests that using 35% as the selection cutoff for this data will produce a parsimonious model that retains good predictive performance. You can use the REFIT option to implement this strategy. The REFIT option requests a second round of model averaging, where you use the MINPCT= suboption to specify the minimum percentage of times an effect must be selected in the initial set of samples to be included in the second round of model averaging. The average model is obtained by averaging the ordinary least squares estimates obtained for each sample. The following statements show how you do this:

```
proc glmselect data=simData seed=3 plots=(ParmDistribution);
   model y=x1-x10/selection=LASSO(adaptive stop=none choose=SBC);
   modelAverage refit(minpct=35 nsamples=1000) alpha=0.1;
run;

ods graphics off;
```

The NSAMPLES=1000 suboption of the REFIT option specifies that 1,000 samples be used in the refit, and the MINPCT=35 suboption specifies the cutoff for inclusion in the refit. The ALPHA=0.1 option specifies that the 5th and 95th percentiles of the estimates be displayed. Output 56.5.10 shows the effects that are used in performing the refit and the resulting average parameter estimates.

Output 56.5.10 Refit Average Parameter Estimates

The GLMSELECT Procedure
Refit Model Averaging Results

Effects: Intercept x1 x2 x5

Average Parameter Estimates					
			Estimate Quantiles		
Parameter	Mean Estimate	Standard Deviation	5%	Median	95%
Intercept	-0.243514	0.315207	-0.762462	-0.230630	0.271510
x1	3.226252	0.299443	2.737843	3.226758	3.708131
x2	1.453584	0.308062	0.947059	1.454635	1.968231
x5	2.226044	0.345185	1.627491	2.228189	2.780034

Output 56.5.11 displays the distributions of the estimates that are obtained for each parameter in the refit model. Because the distributions are approximately normal and a large number of samples are used, it is reasonable to interpret the range between the 5th and 95th percentiles of each estimate as an approximate 90% confidence interval for that estimate.

Output 56.5.11 Effect Selection Percentages

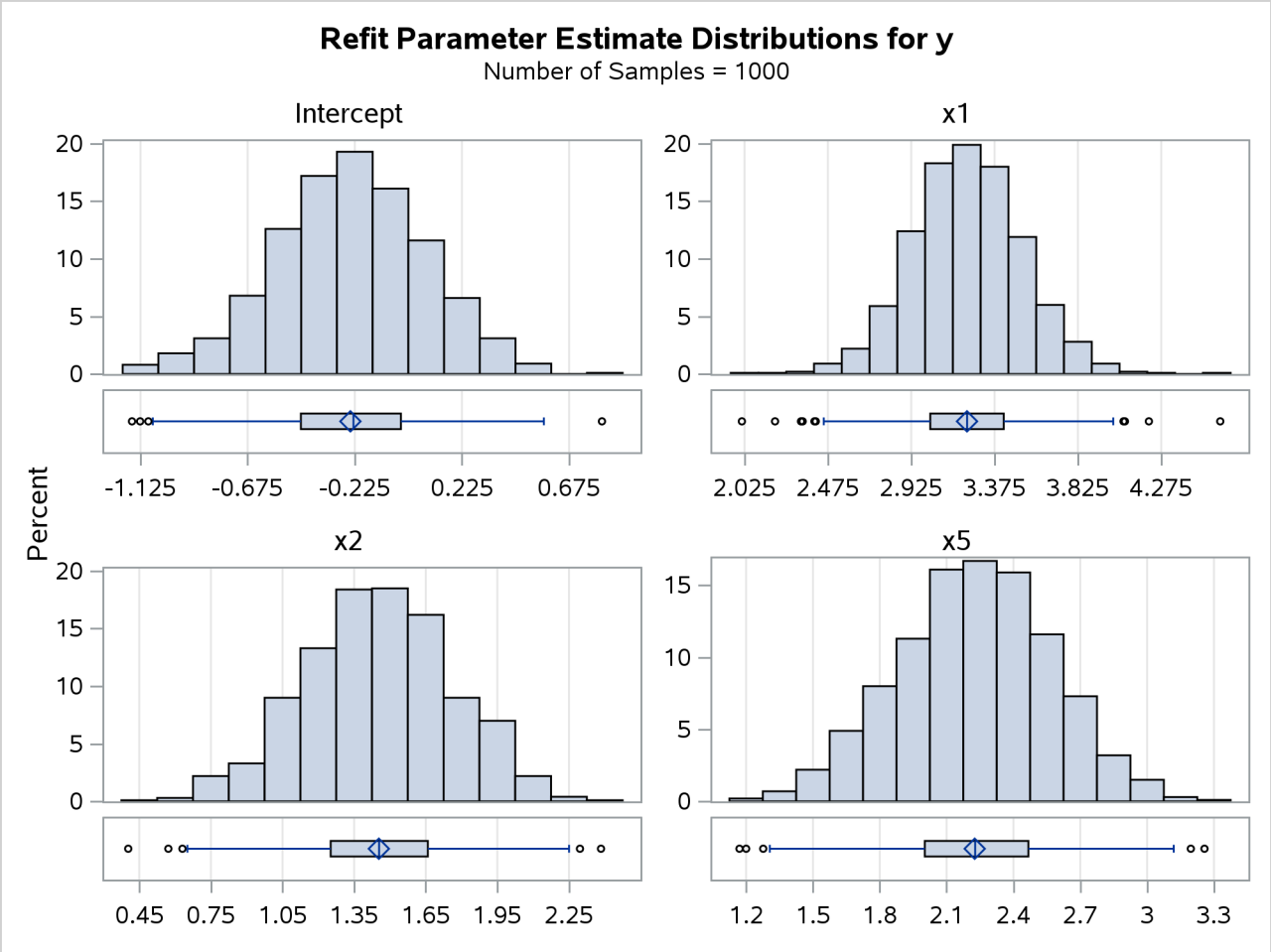

Example 56.6: Elastic Net and External Cross Validation

(View the complete code for this example (glsex6.sas) in the example repository.)

This example shows how to use the elastic net method for model selection and compares it with the LASSO method. The example also uses *k*-fold external cross validation as a criterion in the CHOOSE= option to choose the best model based on the penalized regression fit.

This example uses a microarray data set called the leukemia (LEU) data set (Golub et al. 1999), which is used in the paper by Zou and Hastie (2005) to demonstrate the performance of the elastic net method in comparison with that of LASSO. The LEU data set consists of 7,129 genes and 72 samples, and 38 samples are used as training samples. Among the 38 training samples, 27 are type 1 leukemia (acute lymphoblastic leukemia) and 11 are type 2 leukemia (acute myeloid leukemia). The goal is to construct a diagnostic

rule based on the expression level of those 7,219 genes to predict the type of leukemia. The remaining 34 samples are used as the validation data for picking the appropriate model or as the test data for testing the performance of the selected model. The training and validation data sets are available from Sashelp.Leutrain and Sashelp.Leutest.

To have a basis for comparison, use the following statements to apply the LASSO method for model selection:

```
ods graphics on;
proc glmselect data=sashelp.Leutrain valdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=LASSO(steps=120  choose=validate);
run;
```

The details about the number of observations for training and validation data are given in Output 56.6.1 and Output 56.6.2, respectively.

Output 56.6.1 Number of Training Observations Table

The GLMSELECT Procedure

Observation Profile for Analysis Data	
Number of Observations Read	38
Number of Observations Used	38
Number of Observations Used for Training	38

Output 56.6.2 Number of Validation Observations Table

Observation Profile for Validation Data	
Number of Observations Read	34
Number of Observations Used	34

Output 56.6.3 shows the "LASSO Selection Summary" table. Although the STEPS= suboption of the SELECTION= option specifies that 120 steps of LASSO selection be performed, the LASSO method terminates at step 81 because the selected model is a perfect fit and the number of effects that can be selected by LASSO is bounded by the number of training samples. Note that a VALDATA= data set is named in the PROC GLMSELECT statement and that the minimum of the validation ASE occurs at step 68. Hence the model at this step is selected, resulting in 35 selected effects. Output 56.6.4 shows the standardized coefficients of all the effects selected at some step of the LASSO method, plotted as a function of the step number.

Output 56.6.3 LASSO Selection Summary Table

The GLMSELECT Procedure

LASSO Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Validation ASE
0	Intercept		1	0.8227	1.0287
1	x4847		2	0.7884	0.9787
2	x3320		3	0.7610	0.9507
3	x2020		4	0.4935	0.7061
4	x5039		5	0.2838	0.5527
5	x1249		6	0.2790	0.5518
6	x2242		7	0.2255	0.5513
			.	.	.
			.	.	.
			.	.	.
62	x2534		35	0.0016	0.1554
63		x3320	34	0.0016	0.1557
64	x5631		35	0.0013	0.1532
65		x6021	34	0.0012	0.1534
66		x1745	33	0.0012	0.1535
67	x6376		34	0.0012	0.1535
68	x4831		35	0.0010	0.1531*
69		x6184	34	0.0009	0.1539
70	x3820		35	0.0006	0.1588
71	x3171		36	0.0005	0.1615
72		x6376	35	0.0004	0.1640
73	x5142		36	0.0004	0.1659
74	x1745		37	0.0003	0.1688
75	x3605		38	0.0001	0.1821
76		x573	37	0.0001	0.1825
77	x3221		38	0.0001	0.1842
78		x1745	37	0.0001	0.1842
79	x6163		38	0.0000	0.1918
80		x4831	37	0.0000	0.1920
81	x1745		38	0.0000	0.1940

* Optimal Value of Criterion

Output 56.6.4 LASSO Coefficient Progression Plot

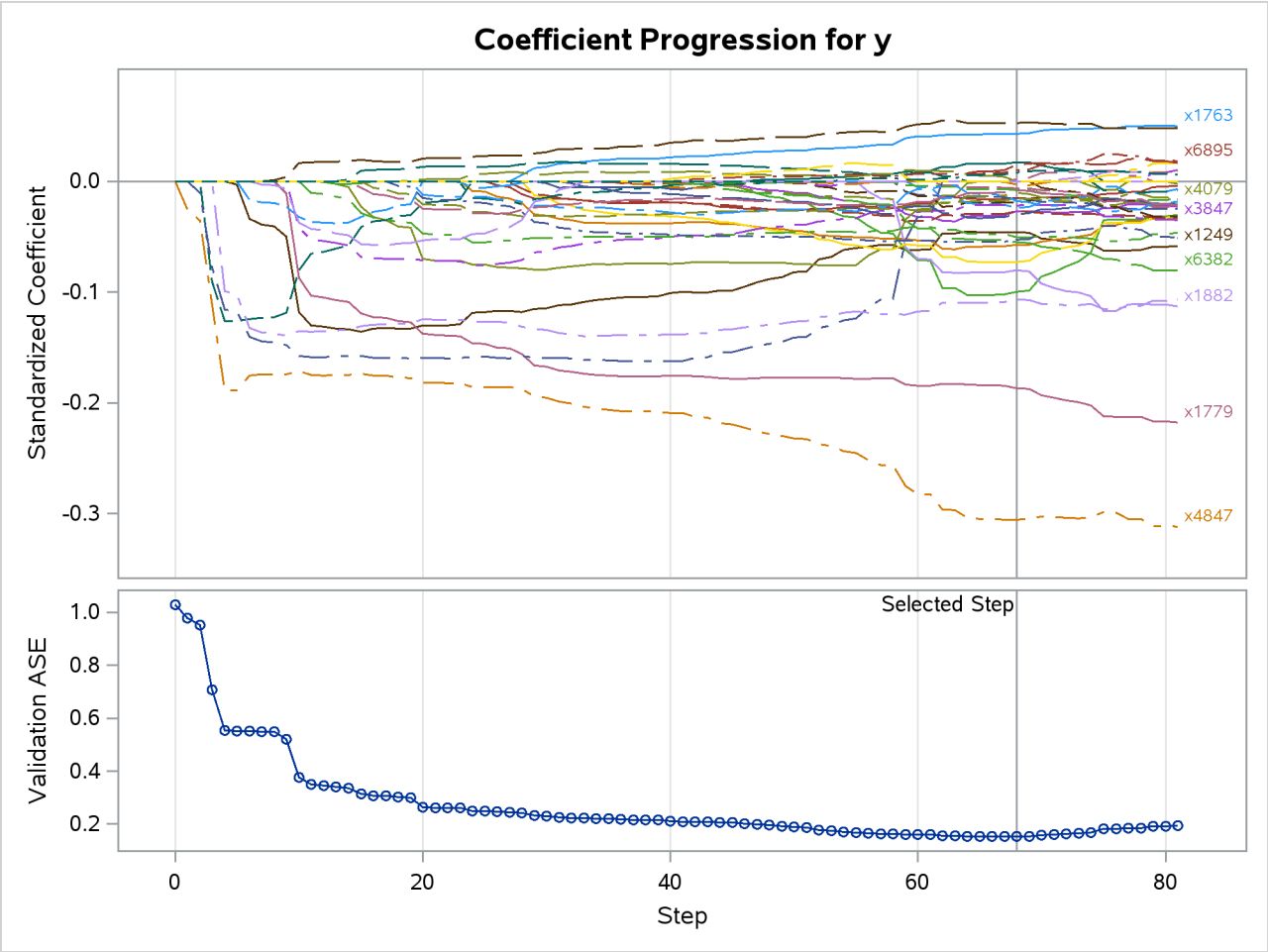

As discussed earlier, the number of effects that can be selected by the LASSO method is upper-bounded by the number of training samples. However, this is not a restriction for the elastic net method, which incorporates an additional ridge regression penalty. You can use the following statements to apply the elastic net method for model selection:

```
proc glmselect data=sashelp.Leutrain valdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=elasticnet(steps=120 L2=0.001 choose=validate);
run;
```

The L2= suboption of the SELECTION= option in the MODEL statement specifies the value of the ridge regression parameter. Output 56.6.5 shows the "Elastic Net Selection Summary" table. As in the example for LASSO, the STEPS= suboption of the SELECTION= option specifies that 120 steps of elastic net selection be performed. However, in contrast to the early termination in the LASSO method at step 81, the elastic net method runs until the specified 120 steps are performed. With the same VALDATA= data set named in the PROC GLMSELECT statement as in the LASSO example, the minimum of the validation ASE occurs at step 105, and hence the model at this step is selected, resulting in 54 selected effects. Compared with the LASSO method, the elastic net method can select more variables, and the number of selected variables is not restricted by the number of samples (38 for this example). Output 56.6.6 shows the standardized coefficients

of all the effects that are selected at some step of the elastic net method, plotted as a function of the step number.

Output 56.6.5 Elastic Net Selection Summary Table

The GLMSELECT Procedure

Elastic Net Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Validation ASE
0	Intercept		1	0.8227	1.0287
1	x4847		2	0.7885	0.9790
2	x3320		3	0.7611	0.9510
3	x2020		4	0.4939	0.7065
4	x5039		5	0.2845	0.5533
5	x1249		6	0.2799	0.5525
6	x2242		7	0.2259	0.5517
			.	.	.
			.	.	.
			.	.	.
100	x259		53	0.0000	0.1207
101	x1781		54	0.0000	0.1207
102		x7065	53	0.0000	0.1198
103	x4079		54	0.0000	0.1193
104		x6163	53	0.0000	0.1193
105	x6539		54	0.0000	0.1192*
106	x6071		55	0.0000	0.1192
107	x1829		56	0.0000	0.1192
108		x5598	55	0.0000	0.1192
109	x6169		56	0.0000	0.1192
110	x1529		57	0.0000	0.1194
111	x1306		58	0.0000	0.1194
112	x5954		59	0.0000	0.1202
113	x4271		60	0.0000	0.1203
114	x3312		61	0.0000	0.1207
115	x461		62	0.0000	0.1213
116	x4697		63	0.0000	0.1228
117		x259	62	0.0000	0.1231
118	x4186		63	0.0000	0.1252
119		x4271	62	0.0000	0.1258
120	x4939		63	0.0000	0.1261

* Optimal Value of Criterion

Output 56.6.6 Elastic Net Coefficient Progression Plot

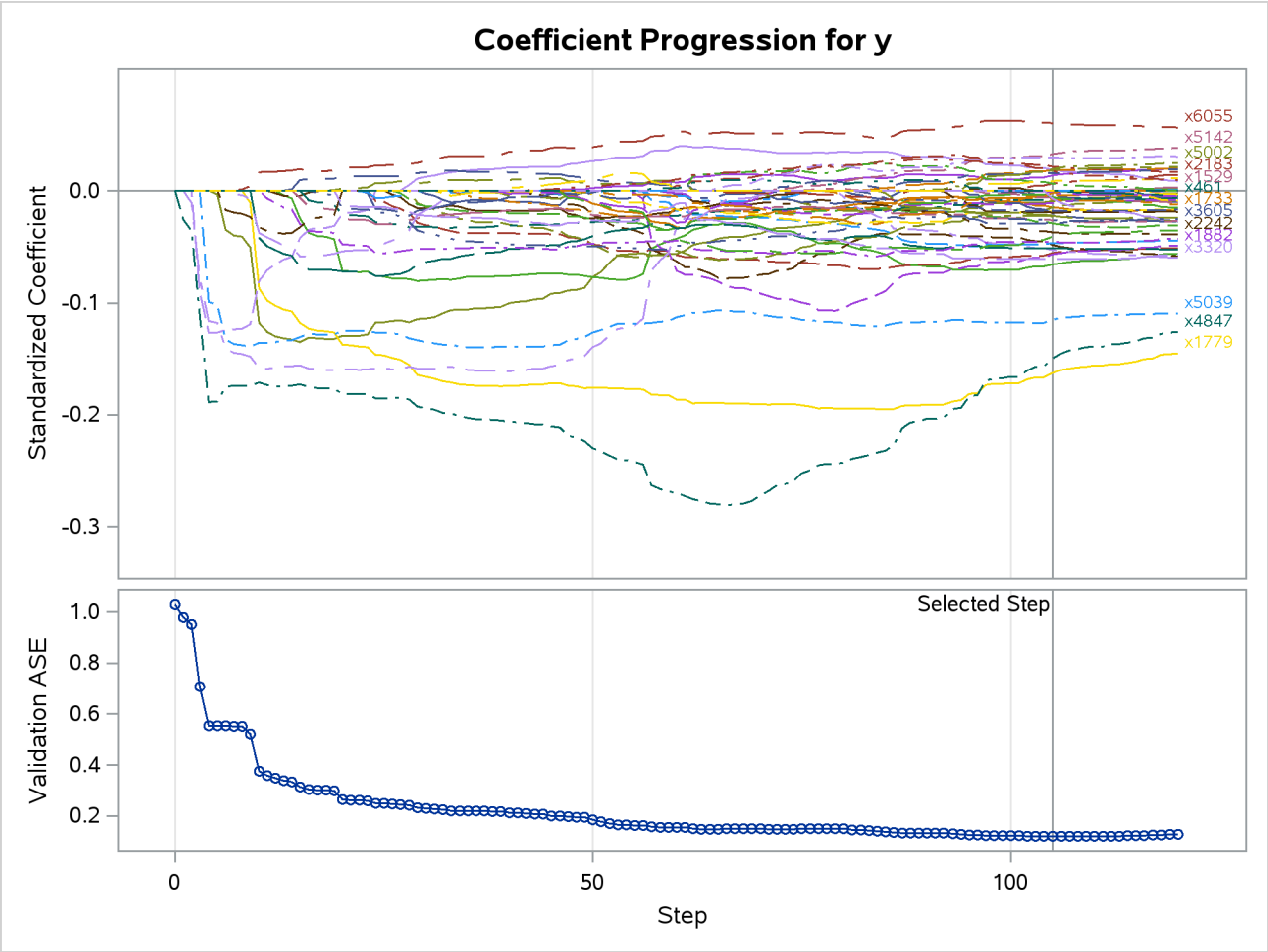

When you have a good estimate of the ridge regression parameter, you can specify the L2= suboption of the SELECTION= option in the MODEL statement. However, when you do not have a good estimate of the ridge regression parameter, you can leave the L2= option unspecified. In this case, PROC GLMSELECT estimates this regression parameter based on the criterion specified in the CHOOSE= option.

If you have a validation data set, you can use the following statements to apply the elastic net method of model selection:

```
proc glmselect data=sashelp.Leutrain valdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=elasticnet(steps=120 choose=validate);
run;
```

PROC GLMSELECT tries a series of candidate values for the ridge regression parameter, which you can control by using the L2HIGH=, L2LOW=, and L2SEARCH= options. The ridge regression parameter is set to the value that achieves the minimum validation ASE (see Figure 56.12 for an illustration). Output 56.6.7 shows the "Elastic Net Selection Summary" table. Compared to Output 56.6.5, where a ridge regression parameter is specified by the L2= option, this table shows that a slightly lower validation ASE is achieved as the value of the ridge regression parameter is optimized on the validation data set.

Output 56.6.7 Elastic Net Selection Summary Table

The GLMSELECT Procedure

Elastic Net Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Validation ASE
0	Intercept		1	0.8227	1.0287
1	x4847		2	0.7884	0.9787
2	x3320		3	0.7610	0.9508
3	x2020		4	0.4936	0.7061
4	x5039		5	0.2839	0.5527
5	x1249		6	0.2790	0.5519
6	x2242		7	0.2256	0.5513
			.	.	.
			.	.	.
			.	.	.
110	x3605		49	0.0000	0.1252
111	x1975		50	0.0000	0.1241
112	x6757		51	0.0000	0.1224
113		x3221	50	0.0000	0.1206
114	x2183		51	0.0000	0.1205
115	x259		52	0.0000	0.1202
116	x157		53	0.0000	0.1201
117	x1781		54	0.0000	0.1197
118		x7065	53	0.0000	0.1194
119	x4079		54	0.0000	0.1187
120	x6539		55	0.0000	0.1186*

* Optimal Value of Criterion

Output 56.6.8 shows the standardized coefficients of all the effects that are selected at some step of the elastic net method, plotted as a function of the step number.

Output 56.6.8 Elastic Net Coefficient Progression Plot

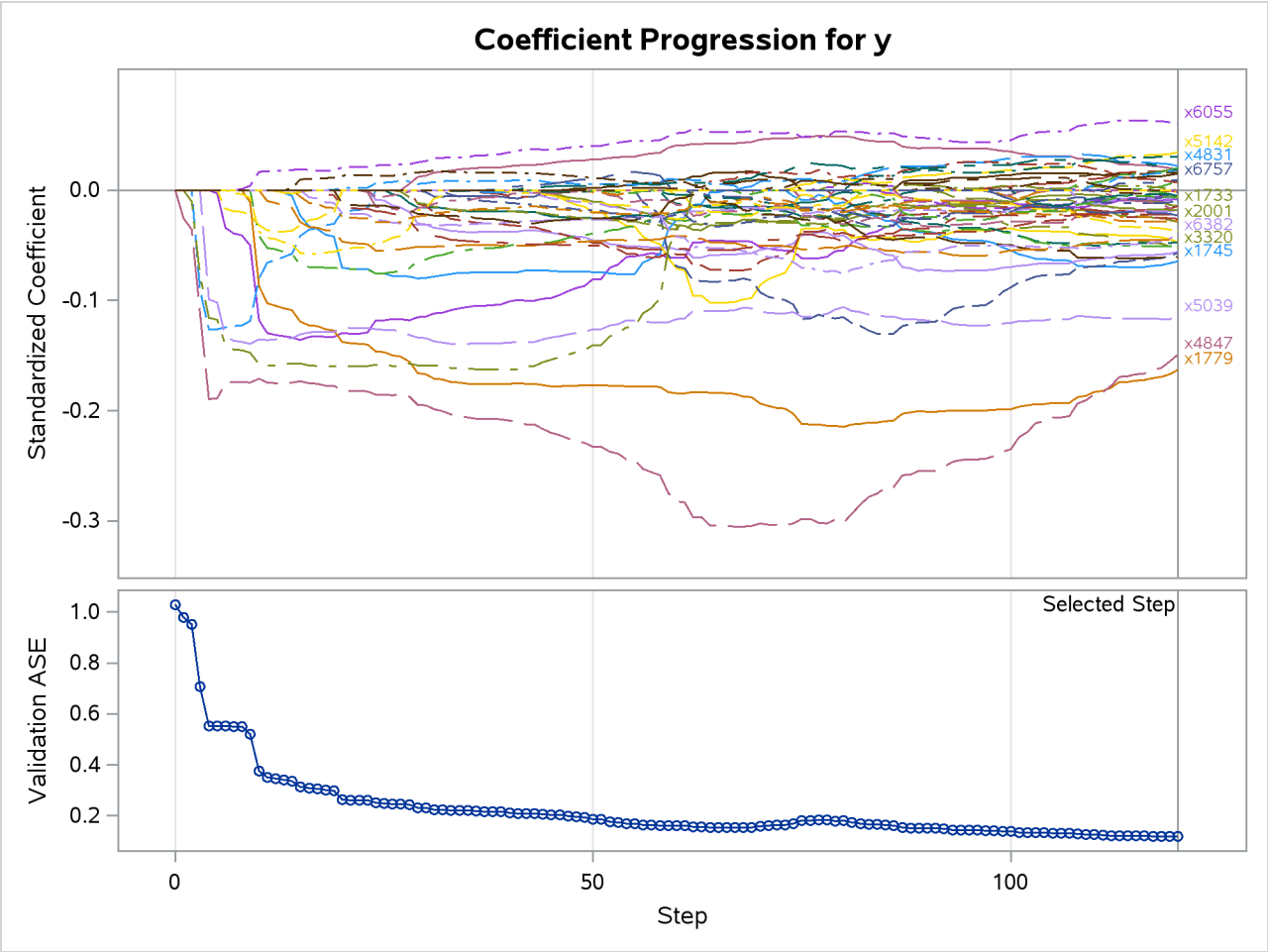

When validation data are not available, you can use other criteria supported by the CHOOSE= option for estimating the value of the ridge regression parameter. Criteria such as AIC, AICC, and BIC are defined for the training samples only, and they tend to set the value of the ridge regression parameter to 0 when the number of training samples is less than the number of variables. In this case, the elastic net method reduces to the LASSO method. As an alternative to AIC, AICC, and BIC, you can use either *k*-fold cross validation or *k*-fold external cross validation in the CHOOSE= option, where the criterion is computed for held-out data.

If you want to use *k*-fold cross validation for selecting the model and selecting the appropriate value for the ridge regression parameter, you can use the following statements to apply the elastic net method of model selection:

```
proc glmselect data=sashelp.Leutrain testdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=elasticnet(steps=120 choose=cv)cvmethod=split(4);
run;
```

Output 56.6.9 Elastic Net Selection Summary Table

The GLMSELECT Procedure

Elastic Net Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Test ASE	CV PRESS
0	Intercept		1	0.8227	1.0287	31.4370
1	x4847		2	0.7884	0.9788	10.6957
2	x3320		3	0.7610	0.9508	10.2065
3	x2020		4	0.4936	0.7062	7.5478
4	x5039		5	0.2840	0.5528	5.7695
5	x1249		6	0.2792	0.5520	5.0187
6	x2242		7	0.2256	0.5514	5.4050
			.	.	.	.
			.	.	.	.
			.	.	.	.
80		x573	37	0.0001	0.1732	24.1821
81	x5766		38	0.0001	0.1732	24.1821
82	x3221		39	0.0001	0.1706	24.1821
83	x6376		40	0.0001	0.1648	24.1821
84	x6163		41	0.0001	0.1642	24.1821
85		x4079	40	0.0000	0.1643	1.5316*
86		x464	39	0.0000	0.1641	2.3192
87	x573		40	0.0000	0.1639	2.3192
88	x1987		41	0.0000	0.1603	2.3192
89	x3605		42	0.0000	0.1555	2.3192
90		x3221	41	0.0000	0.1533	2.3192
91	x6021		42	0.0000	0.1498	2.3192
92		x2793	41	0.0000	0.1504	2.9494
93	x4820		42	0.0000	0.1502	2.9494
94	x3320		43	0.0000	0.1464	2.9494
95	x4399		44	0.0000	0.1430	2.9487
96	x2121		45	0.0000	0.1415	2.9488
97	x4142		46	0.0000	0.1388	2.9488
98		x3171	45	0.0000	0.1372	2.9488
99	x4461		46	0.0000	0.1339	2.9488
100	x4697		47	0.0000	0.1335	2.9488
101		x3605	46	0.0000	0.1328	2.9488
102	x3221		47	0.0000	0.1327	3.0515
103	x5002		48	0.0000	0.1319	3.0515
104		x4697	47	0.0000	0.1319	3.0515
105		x3820	46	0.0000	0.1314	3.0515
106	x4052		47	0.0000	0.1277	3.0515
107	x6184		48	0.0000	0.1268	3.0515
108	x3605		49	0.0000	0.1254	3.0515
109		x3221	48	0.0000	0.1251	2.9488
110	x1975		49	0.0000	0.1248	2.9488
111	x3221		50	0.0000	0.1242	3.0164

* Optimal Value of Criterion

Output 56.6.9 *continued*

The GLMSELECT Procedure

Elastic Net Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Test ASE	CV PRESS
112	x6757		51	0.0000	0.1233	3.0164
113		x3221	50	0.0000	0.1207	2.9489
114	x2183		51	0.0000	0.1206	2.9805
115	x259		52	0.0000	0.1204	2.9805
116	x157		53	0.0000	0.1202	2.9805
117	x1781		54	0.0000	0.1198	2.9805
118		x7065	53	0.0000	0.1195	250.8558
119	x4079		54	0.0000	0.1188	250.8558
120	x6539		55	0.0000	0.1186	250.8558

* Optimal Value of Criterion

A TESTDATA= data set is named in the PROC GLMSELECT statement; this is the same as the validation data set used earlier. Because the L2= option is not specified, PROC GLMSELECT tries a series of candidate values for the ridge regression parameter according to the CHOOSE=CV option, and the ridge regression parameter is set to the value that achieves the minimal CVPRESS score. Output 56.6.9 shows the "Elastic Net Selection Summary" table, which corresponds to the ridge regression parameter selected by k-fold cross validation. The elastic net method achieves the smallest CVPRESS score at step 92. Hence the model at this step is selected, resulting in 41 selected effects.

Output 56.6.10 shows the standardized coefficients of all the effects selected at some step of the elastic net method, plotted as a function of the step number. Note that k-fold cross validation uses a least squares fit to compute the CVPRESS score. Thus the criterion does not directly depend on the penalized regression used in the elastic net method, and the CVPRESS curve in Output 56.6.10 has abrupt jumps.

Output 56.6.10 Elastic Net Coefficient Progression Plot

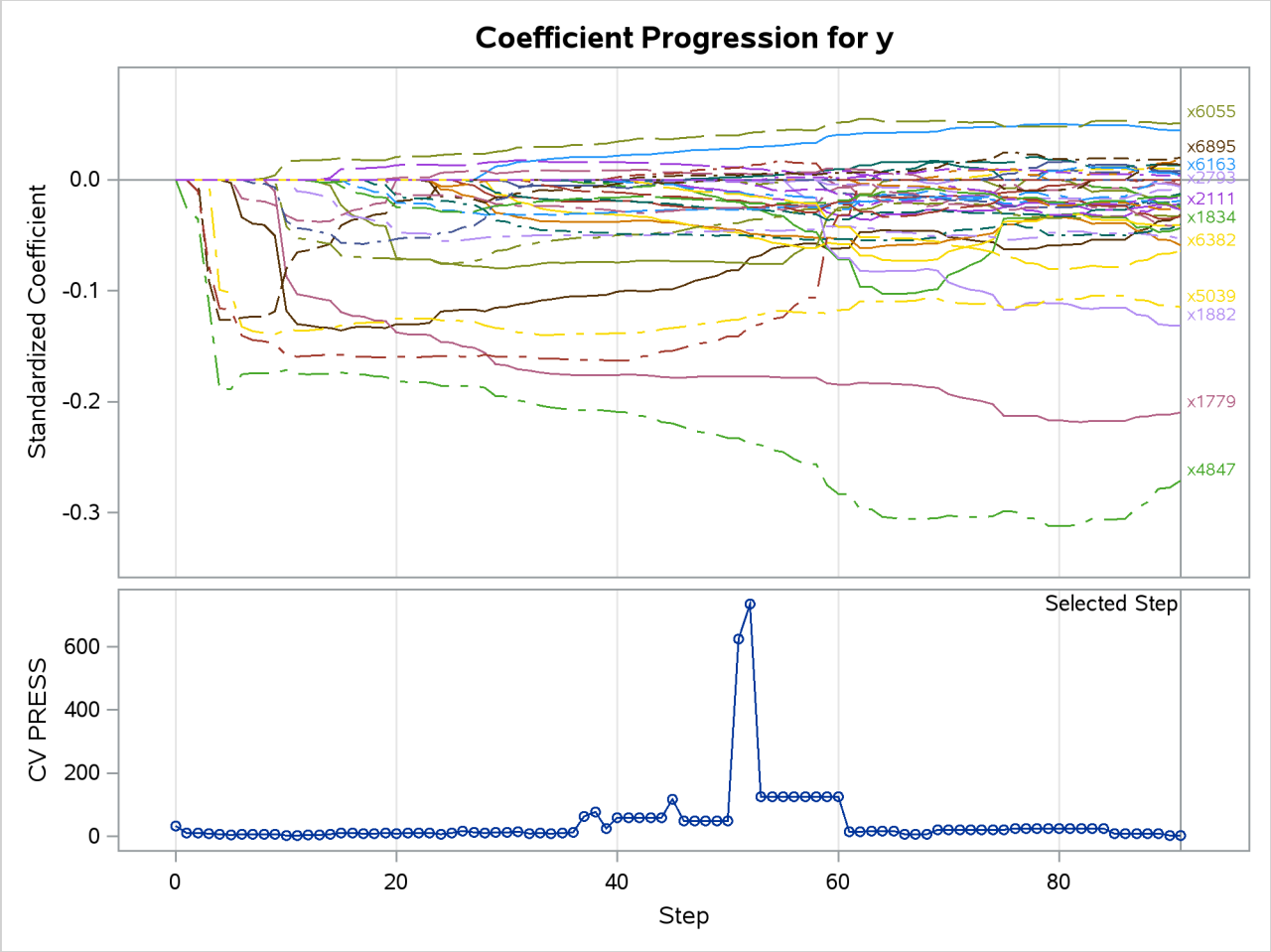

If you want to perform the selection of both the ridge regression parameter and the selected variables in the model based on a criterion that directly depends on the regression used in the elastic net method, you can use k-fold external cross validation by specifying the following statements:

```
proc glmselect data=sashelp.Leutrain testdata=sashelp.Leutest
               plots=coefficients;
   model y = x1-x7129/
         selection=elasticnet(steps=120 choose=cvex)cvmethod=split(4);
run;
```

Output 56.6.11 Elastic Net Coefficient Progression Plot

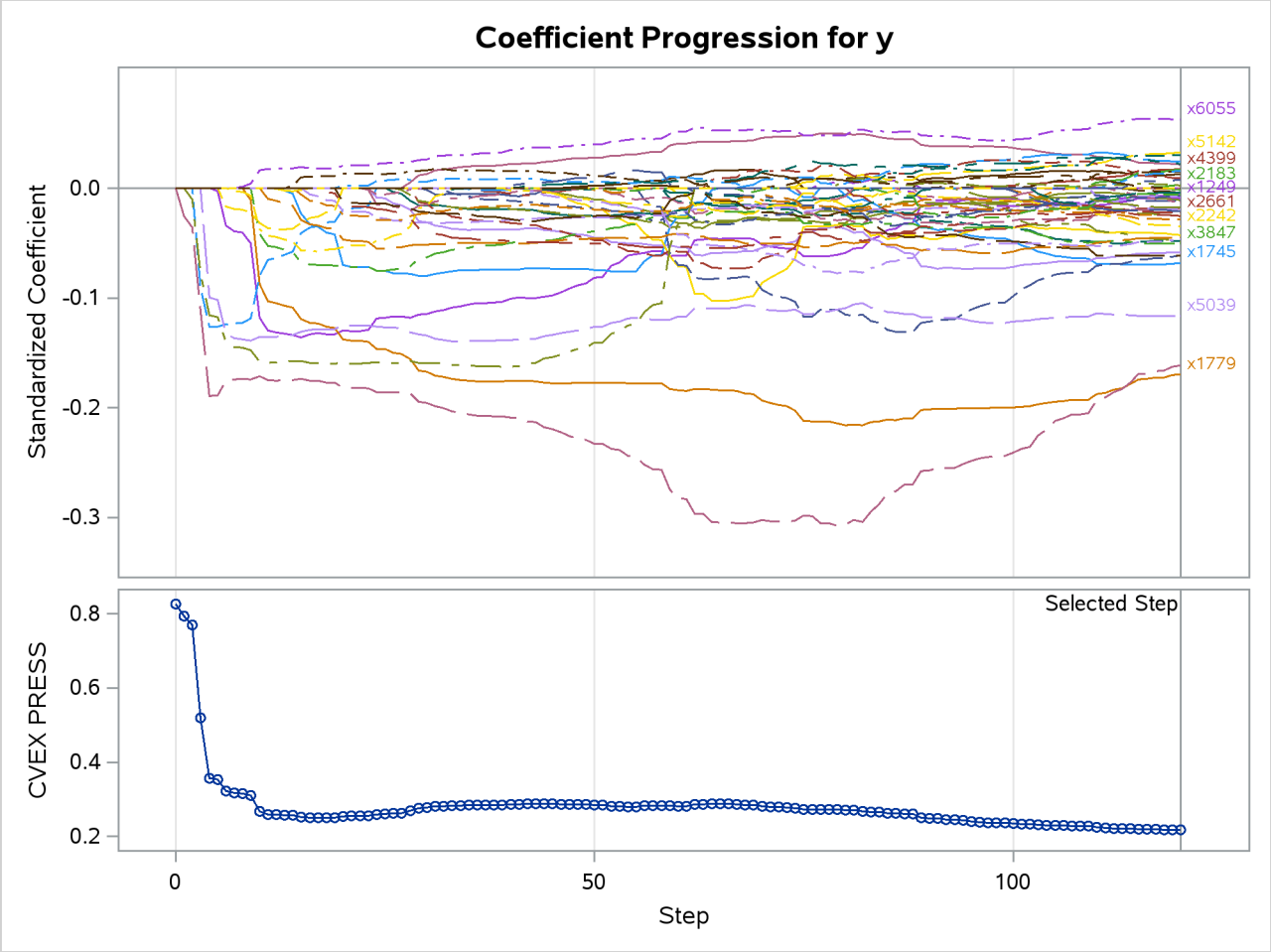

As earlier, a TESTDATA= data set is named in the PROC GLMSELECT statement to test the prediction accuracy of the model on the training samples. Because the L2= option is not specified, PROC GLMSELECT tries a series of candidate values for the ridge regression parameter. Output 56.6.11 shows the standardized coefficients of all the effects selected at some step of the elastic net method, plotted as a function of the step number, and also the curve of the CVEXPRESS statistic as a function of the step number. A comparison between the criterion curves in Output 56.6.10 and Output 56.6.11 shows that the CVEXPRESS statistic is smoother than the CVPRESS statistic. Note that the CVEXPRESS statistic is based on a penalized model, whereas the CVPRESS statistic is based on an ordinary least squares model. Output 56.6.12 shows the "Elastic Net Selection Summary" table, which corresponds to the ridge regression parameter selected by k-fold external cross validation. The elastic net method achieves the smallest CVEXPRESS score at step 120, and hence the model at this step is selected, resulting in 53 selected effects.

Output 56.6.12 Elastic Net Selection Summary Table

The GLMSELECT Procedure

Elastic Net Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	ASE	Test ASE	CVEX PRESS
0	Intercept		1	0.8227	1.0287	0.8265
1	x4847		2	0.7884	0.9787	0.7940
2	x3320		3	0.7610	0.9507	0.7698
3	x2020		4	0.4936	0.7061	0.5195
4	x5039		5	0.2839	0.5527	0.3554
5	x1249		6	0.2790	0.5519	0.3526
6	x2242		7	0.2255	0.5513	0.3218
7	x6539		8	0.2161	0.5498	0.3166
8	x6055		9	0.2128	0.5488	0.3148
9	x1779		10	0.1968	0.5205	0.3096
10	x3847		11	0.1065	0.3765	0.2671
11	x4951		12	0.0891	0.3498	0.2586
12	x1834		13	0.0868	0.3458	0.2579
13	x1745		14	0.0837	0.3397	0.2570
14	x6021		15	0.0809	0.3355	0.2561
15	x6362		16	0.0667	0.3144	0.2510
16	x2238		17	0.0621	0.3075	0.2498
			.	.	.	.
			.	.	.	.
			.	.	.	.
100	x2121		45	0.0000	0.1402	0.2350
101	x6021		46	0.0000	0.1386	0.2334
102		x3171	45	0.0000	0.1377	0.2324
103	x4461		46	0.0000	0.1350	0.2302
104	x4697		47	0.0000	0.1338	0.2299
105		x3605	46	0.0000	0.1331	0.2293
106	x3221		47	0.0000	0.1330	0.2289
107	x5002		48	0.0000	0.1320	0.2282
108		x3820	47	0.0000	0.1319	0.2281
109		x4697	46	0.0000	0.1315	0.2278
110	x4052		47	0.0000	0.1276	0.2235
111	x6184		48	0.0000	0.1266	0.2229
112	x3605		49	0.0000	0.1252	0.2215
113	x1975		50	0.0000	0.1240	0.2212
114	x6757		51	0.0000	0.1222	0.2200
115		x3221	50	0.0000	0.1206	0.2185
116	x2183		51	0.0000	0.1205	0.2185
117	x259		52	0.0000	0.1202	0.2182
118	x157		53	0.0000	0.1201	0.2181
119	x1781		54	0.0000	0.1197	0.2177
120		x7065	53	0.0000	0.1194	0.2177*

* Optimal Value of Criterion

Example 56.7: LASSO with Screening

(View the complete code for this example (glsex7.sas) in the example repository.)

This example shows how you can use the SCREEN= option to speed up model selection when you have a large number of regressors. In order to demonstrate the efficiency in screening model selection, this example uses simulated data in which the response y depends systematically on a relatively small subset of a much larger set of regressors, which is described in Table 56.13.

Table 56.13 Complete Set of Regressors

Regressor Name	Type	Number of Levels	In True Model
xIn1–xIn5	Continuous		Yes
xOut1–xOut1000	Continuous		No
cIn1–cIn2	Classification	2–5	Yes
cOut1–cOut1000	Classification	2–5	No

The labels In and Out, which are part of the regressor names, make it easy to identify whether the selected model succeeds or fails in capturing the true underlying model.

The following DATA step generates the data:

```
%let nObs      = 5000;
%let nContIn   = 5;
%let nContOut  = 1000;
%let nClassIn  = 2;
%let nClassOut = 1000;
%let maxLevs   = 5;
%let noiseScale= 1;

 data ex7Data;
   array xIn{&nContIn};
   array xOut{&nContOut};
   array cIn{&nClassIn};
   array cOut{&nClassOut};

   drop i j sign nLevs xBeta;

   do i=1 to &nObs;
      sign  = -1;
      xBeta = 0;
      do j=1 to dim(xIn);
         xIn{j} = ranuni(1);
         xBeta  = xBeta + sqrt(j)*sign*xIn{j};
         sign   = -sign;
      end;
      do j=1 to dim(xOut);
         xOut{j} = ranuni(1);
      end;

      do j=1 to dim(cIn);
```

```
         nLevs  = 2 + mod(j,&maxlevs-1);
         cIn{j} = 1+int(ranuni(1)*nLevs);
         xBeta  = xBeta + j*sign*(cIn{j}-nLevs/2);
         sign   = -sign;
      end;

      do j=1 to dim(cOut);
         nLevs  = 2 + mod(j,&maxlevs-1);
         cOut{j} = 1+int(ranuni(1)*nLevs);
      end;

      y = xBeta + &noiseScale*rannor(1);

      output;
   end;
run;
```

The following statements apply the LASSO method for model selection but do not specify any screening technique:

```
proc glmselect data=ex7Data;
   class c:;
   model y = x: c:/
         selection=lasso;
run;
```

Output 56.7.1 and Output 56.7.2 show the number of observations and the number of effects, respectively. You can see that 5,000 observations are used for training and the number of effects after splits is 4,513.

Output 56.7.1 Number of Training Observations Table

The GLMSELECT Procedure

Number of Observations Read	5000
Number of Observations Used	5000

Output 56.7.2 Dimensions

Dimensions	
Number of Effects	2008
Number of Effects after Splits	4513
Number of Parameters	4513

Output 56.7.3 shows the "LASSO Selection Summary" table. You can see that the selected model contains all the predictive effects and successfully excludes all the noise effects.

Output 56.7.3 LASSO Selection Summary Table

The GLMSELECT Procedure

Without the SCREEN= option

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	10413.6418
1	cIn2_1		2	10335.5423
2	cIn2_4		3	7237.0701
3	cIn1_1		4	7186.9642
4	cIn1_3		5	6753.6076
5	xIn5		6	6609.0322
6	cIn2_3		7	6510.2639
7	xIn4		8	5797.1607
8	xIn3		9	5487.4432
9	xIn2		10	2894.0166
10	xIn1		11	310.9340*

* Optimal Value of Criterion

You can improve the computational performance of the LASSO method by using the SCREEN=SASVI option as follows:

```
proc glmselect data=ex7Data;
   class c:;
   model y = x: c:/
         selection=lasso(screen=sasvi);
run;
```

Output 56.7.4 shows that the safe screening approach SASVI is used.

Output 56.7.4 Screening Information

The GLMSELECT Procedure

Screening Information	
Screening Method	SASVI (safe screening)

Output 56.7.5 shows the "LASSO Selection Summary" table, which is exactly the same as Output 56.7.3. If you compare the "Parameter Estimates" tables that correspond to Output 56.7.3 and Output 56.7.5, you can see that the parameter estimates are exactly the same, because SASVI is a safe screening approach that guarantees the same solution, usually in less time. PROC GLMSELECT runs about twice as fast when SCREEN=SASVI as it runs when no screening is specified.

Output 56.7.5 LASSO Selection Summary Table

The GLMSELECT Procedure

SCREEN=SASVI

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	10413.6418
1	cIn2_1		2	10335.5423
2	cIn2_4		3	7237.0701
3	cIn1_1		4	7186.9642
4	cIn1_3		5	6753.6076
5	xIn5		6	6609.0322
6	cIn2_3		7	6510.2639
7	xIn4		8	5797.1607
8	xIn3		9	5487.4432
9	xIn2		10	2894.0166
10	xIn1		11	310.9340*

* Optimal Value of Criterion

As an alternative to screening by SASVI, you can specify SCREEN=SIS for sure independence screening as follows:

```
proc glmselect data=ex7Data;
   class c:;
   model y = x: c:/
         selection=lasso(screen=sis(keepnum=15));
run;
```

Output 56.7.6 shows that sure independence screening is used and that the number of effects to be kept before applying LASSO for model selection is 15. The last two rows of Output 56.7.6 show the actual number of kept effects and the corresponding ratio with the total number of effects.

Output 56.7.6 Screening Information

The GLMSELECT Procedure

Screening Information	
Screening Method	SIS (sure independence screening)
Screening Keep Number	15
Actual Kept Number	15
Actual Kept Ratio	0.003

Output 56.7.7 shows the absolute correlation between the effects and the response that corresponds to the leading 20 effects. Because you specified KEEPNUM=15, only the leading 15 effects are kept for model selection by LASSO. In this example, you can see that all the predictive effects are among the top 15. Note that if the screening stage does not keep an effect, that effect has no chance of being selected in the final model.

Output 56.7.7 LASSO Selection Summary Table

The GLMSELECT Procedure

SCREEN=SIS

Effect Screening by Correlation		
Rank	**Effect**	**Maximum Absolute Correlation**
1	cIn2_1	0.621286
2	cIn2_4	0.611998
3	cIn1_1	0.261449
4	cIn1_3	0.259374
5	cIn2_3	0.215103
6	xIn5	0.212433
7	xIn4	0.201704
8	cIn2_2	0.195087
9	xIn3	0.174133
10	xIn2	0.170705
11	xIn1	0.083742
12	cOut447_4	0.061958
13	cOut238_1	0.052981
14	cOut538_4	0.052165
15	cOut747_5	0.051390
16	xOut345	0.050114*
17	cOut511_5	0.047524*
18	cOut630_2	0.047440*
19	cOut672_1	0.045937*
20	cOut672_2	0.045937*
* Screened Out Effect		

Output 56.7.8 shows the "LASSO Selection Summary" table. As in Output 56.7.3, all the predictive effects enter into the model. However, one noise effect also enters into the model, because SCREEN=SIS does not necessarily guarantee the same solution as the solution that results when SCREEN=NONE. For this example, PROC GLMSELECT runs only slightly faster when SCREEN=SIS than it does when SCREEN=SASVI, although it runs about twice as fast as it does when SCREEN=NONE. However, for problems that have more predictors or that use much more computationally intense CHOOSE= criterion, sure independence screening (SIS) can run faster by orders of magnitude.

Output 56.7.8 LASSO Selection Summary Table

The GLMSELECT Procedure

SCREEN=SIS

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	10413.6418
1	cIn2_1		2	10335.5423
2	cIn2_4		3	7237.0701
3	cIn1_1		4	7186.9642
4	cIn1_3		5	6753.6076
5	xIn5		6	6609.0322
6	cIn2_3		7	6510.2639
7	xIn4		8	5797.1607
8	xIn3		9	5487.4432
9	xIn2		10	2894.0166
10	xIn1		11	245.9693
11	cOut747_5		12	238.4427*

* Optimal Value of Criterion

Example 56.8: Group LASSO Selection

(View the complete code for this example (glsex8.sas) in the example repository.)

This example shows how you can use the group LASSO method for model selection. This example treats the parameters that correspond to the same spline and CLASS variable as a group and also uses a collection effect to group otherwise unrelated parameters. The following DATA step generates the data:

```
%macro makeRegressorData(data=,nObs=500,nCont=5,nClass=5,nLev=3);
   data &data;
      drop i j;
      %if &nCont>0  %then %do; array x{&nCont}  x1-x&nCont; %end;
      %if &nClass>0 %then %do; array c{&nClass} c1-c&nClass;%end;
      do i = 1 to &nObs;
         %if &nCont>0 %then %do;
            do j= 1 to &nCont;
               x{j} = rannor(1);
            end;
         %end;
         %if &nClass > 0 %then %do;
            do j=1 to &nClass;
               if      mod(j,3) = 0 then c{j} = ranbin(1,&nLev,.6);
               else if mod(j,3) = 1 then c{j} = ranbin(1,&nLev,.5);
               else if mod(j,3) = 2 then c{j} = ranbin(1,&nLev,.4);
            end;
         %end;
         output;
      end;
```

```
   run;
%mend;

%makeRegressorData(data=traindata,nObs=500,nCont=5,nClass=5,nLev=3);
```

In the generated data, there are five continuous effects, x1–x5, and five CLASS variables, c1–c5. The following DATA step generates a response, y, that is dependent on x1–x4 and c1:

```
%macro AddDepVar(data=,modelRHS =,errorStd = 1);
   data &data;
      set &data;
      y = &modelRHS + &errorStd * rannor(1);
   run;
%mend;

%AddDepVar(data    = traindata,
           modelRHS= x1 +
                     0.1*x2 - 0.1*x3 - 0.01* x4 -
                     c1,
           errorStd= 1);
```

The effects x1 and c1 have a stronger influence on the response y than x2 and x3, and x4 has the weakest influence. For the continuous effect x1, a spline effect is constructed using the cubic B-spline basis (see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics") as follows:

```
effect s1=spline(x1)
```

Models that contain spline effects are particularly good candidates for group LASSO selection, because the individual parameters in a spline effect have no meaning by themselves. You need them all to be in the model together in order to define the correct spline term in the model.

To have a basis for comparison, first use the following statements to apply LASSO to model selection:

```
ods graphics on;

proc glmselect data=traindata plots=coefficients;
  class c1-c5/split;
  effect s1=spline(x1/split);
  model y = s1 x2-x5 c:/
        selection=lasso(steps=20 choose=sbc);
run;
```

In LASSO selection, effects that have multiple parameters are split into their constituent parameters. In this problem, the spline effect has seven parameters by default and each of the five CLASS effects has four parameters. So together with the intercept and four continuous main effects, your model has a total of 32 parameters, as shown in the "Dimensions" table in Output 56.8.1. Note that the number of effects in the table is 17, because the spline effect s1 is counted as 7 effects.

Output 56.8.1 Dimensions

The GLMSELECT Procedure

Dimensions	
Number of Effects	17
Number of Effects after Splits	32
Number of Parameters	32

Output 56.8.2 shows the "LASSO Selection Summary" table, and Output 56.8.3 shows the standardized coefficients of all the effects selected at some step of the LASSO method, plotted as a function of the step number. Because you specified CHOOSE=SBC to pick the best model, the SBC value for the model at each step is also displayed in Output 56.8.3, with the minimum occurring at step 12. Hence the model at this step is selected, resulting in 13 selected effects. The "Parameter Estimates" table in Figure 56.8.4 shows that x2 and x3 as well as parts of the spline effect and parts of three CLASS effects are selected for the final model. So this selection misses the small true effect of x4 and erroneously includes c3 and c4 because of noise.

Output 56.8.2 LASSO Selection Summary Table

The GLMSELECT Procedure

LASSO Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	491.1109
1	s1:5		2	410.2008
2	s1:2		3	386.7278
3	c1_3		4	388.1524
4	s1:6		5	385.0045
5	c1_0		6	216.9169
6	c1_2		7	84.3459
7	x2		8	82.8547
8	s1:3		9	52.3145
9	c4_0		10	46.8144
10	c3_1		11	46.6228
11	x3		12	46.2866
12	c3_2		13	42.7335*
13	c5_3		14	42.8659
14	c2_0		15	47.1151
15	s1:7		16	52.5826
16	x5		17	58.3346
17	c3_0		18	63.8812
18	c2_3		19	69.4954
19	x4		20	75.6385
20	s1:1		21	79.8007

* Optimal Value of Criterion

Output 56.8.3 LASSO Coefficient Progression Plot

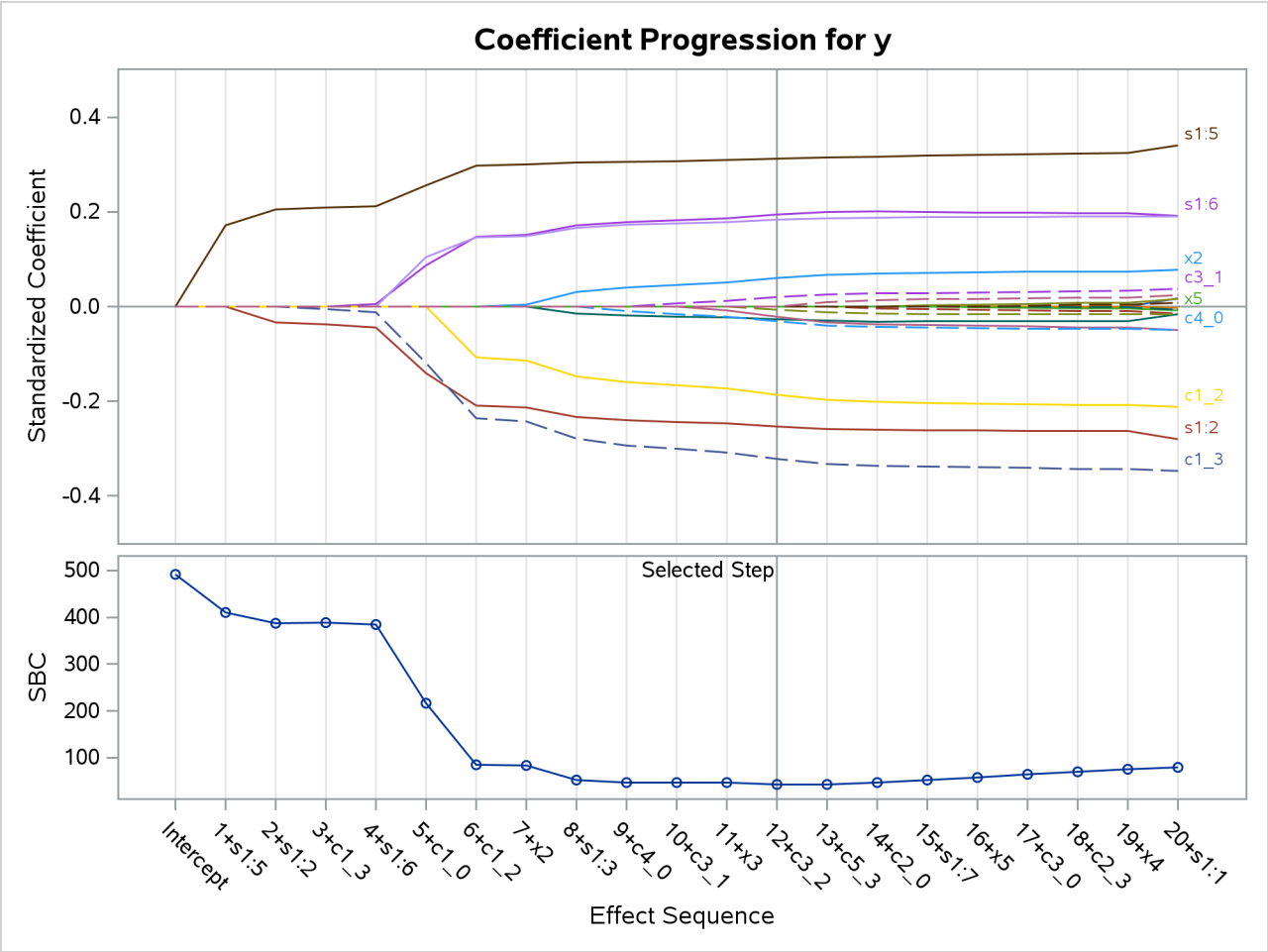

Output 56.8.4 LASSO Parameter Estimates

Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	-1.636852
s1:2	1	-3.829173
s1:3	1	-0.192499
s1:5	1	2.259207
s1:6	1	3.040642
x2	1	0.098525
x3	1	-0.033892
c1_0	1	0.852990
c1_2	1	-0.635575
c1_3	1	-1.643039
c3_1	1	0.070937
c3_2	1	-0.022502
c4_0	1	-0.149375

As opposed to what happens in LASSO selection, spline effects and classification effects are not split by default in group LASSO selection. In addition, you can use a collection effect to construct a group of three of the continuous effects, as shown in the following statements:

```
proc glmselect data=traindata plots=coefficients;
  class c1-c5;
  effect s1=spline(x1);
  effect s2=collection(x2 x3 x4);
  model y = s1 s2 x5 c:/
        selection=grouplasso(steps=20 choose=sbc rho=0.8);
run;
```

Because spline and classification effects are not split by default in group LASSO selection, this model contains nine effects: s1, s2, x5, c1–c5, and the intercept. The number of parameters is the same as before, as shown in Output 56.8.5.

Output 56.8.5 Dimensions

The GLMSELECT Procedure

Dimensions	
Number of Effects	9
Number of Parameters	32

The RHO=0.8 option specifies the value of ρ for determining the regularization parameter $\lambda = \rho^i$ used in the ith step of the group LASSO selection process. If you use a larger value for ρ, you can expect the plot shown in Output 56.8.7 to exhibit a finer coefficient progression.

Output 56.8.6 shows the "Group LASSO Selection Summary" table, and Output 56.8.7 shows the standardized coefficients of all the effects selected at some step of the group LASSO method, plotted as a function of the step number. As you can see in this plot, the CHOOSE=SBC option selects the model at step 10 as the one that has the minimum value of the SBC statistic. The resulting model contains the four effects, including all the true ones, and 15 parameters, as shown in the "Parameter Estimates" table in Output 56.8.8.

Unlike LASSO, which adds or drops an effect at each step, the group LASSO method can add or drop more than one effect. Indeed, you can see in Output 56.8.7 that steps 11 and 16 each added two effects to the model. Simple selection breaks down because group LASSO does not admit a piecewise linear constant solution path, as regular LASSO does.

Output 56.8.6 Group LASSO Selection Summary Table

The GLMSELECT Procedure

Group LASSO Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	491.1109
1	s1		2	457.0038
2	c1		3	400.1165
3			3	302.8975
4			3	228.2883
5			3	173.3913
6			3	134.3346
7			3	107.2038
8			3	88.6178
9	s2		4	91.9567
10			4	79.7022*
11	c3 c4		6	119.8826
12			6	111.5926
13			6	105.8915
14			6	101.9617
15	c5		7	123.6787
16	x5 c2		9	152.2387
17			9	150.4210
18			9	149.1847
19			9	148.3496
20			9	147.7893

* Optimal Value of Criterion

Output 56.8.7 LASSO Coefficient Progression Plot

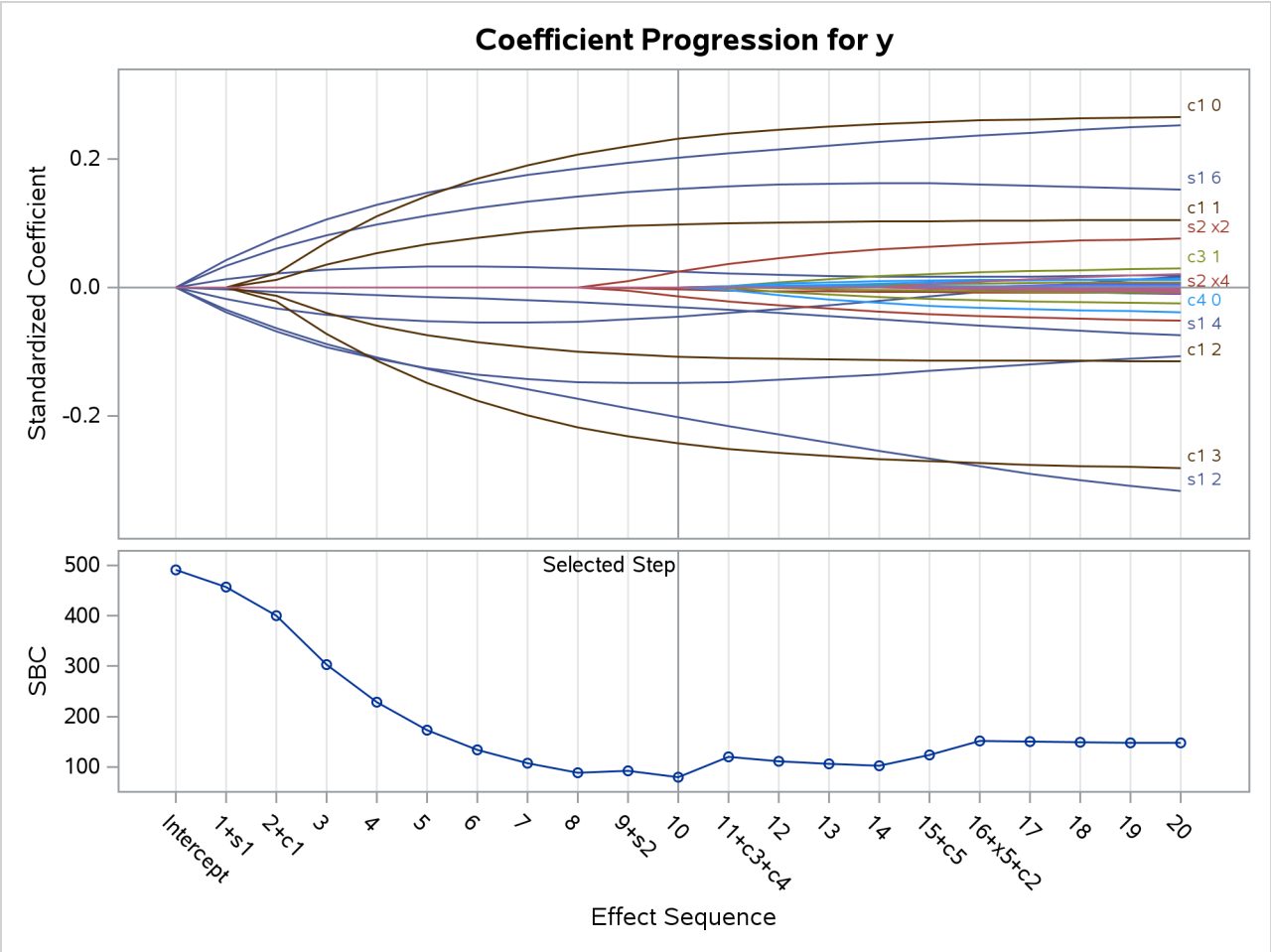

Output 56.8.8 Group LASSO Parameter Estimates

Parameter Estimates		
Parameter		**Estimate**
Intercept		-1.439483
s1	**1**	-5.320998
s1	**2**	-3.058403
s1	**3**	-1.090183
s1	**4**	-0.240067
s1	**5**	1.458828
s1	**6**	2.404019
s1	**7**	3.023258
s2	**x2**	0.040036
s2	**x3**	-0.022261
s2	**x4**	-0.005458
c1	**0**	1.075888
c1	**1**	0.326289
c1	**2**	-0.366125
c1	**3**	-1.240907

References

Breiman, L. (1992). "The Little Bootstrap and Other Methods for Dimensionality Selection in Regression: X-Fixed Prediction Error." *Journal of the American Statistical Association* 87:738–754.

Burnham, K. P., and Anderson, D. R. (2002). *Model Selection and Multimodel Inference*. 2nd ed. New York: Springer-Verlag.

Darlington, R. B. (1968). "Multiple Regression in Psychological Research and Practice." *Psychological Bulletin* 69:161–182.

Donoho, D. L., and Johnstone, I. M. (1994). "Ideal Spatial Adaptation via Wavelet Shrinkage." *Biometrika* 81:425–455.

Draper, N. R., Guttman, I., and Kanemasu, H. (1971). "The Distribution of Certain Regression Statistics." *Biometrika* 58:295–298.

Efron, B., Hastie, T. J., Johnstone, I. M., and Tibshirani, R. (2004). "Least Angle Regression." *Annals of Statistics* 32:407–499. With discussion.

Efron, B., and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. New York: Chapman & Hall.

Eilers, P. H. C., and Marx, B. D. (1996). "Flexible Smoothing with *B*-Splines and Penalties." *Statistical Science* 11:89–121. With discussion.

El Ghaoui, L., Viallon, V., and Rabbani, T. (2012). "Safe Feature Elimination for the Lasso and Sparse Supervised Learning Problems." *Pacific Journal of Optimization* 8:667–698. Special issue on conic optimization.

Fan, J., and Lv, J. (2008). "Sure Independence Screening for Ultrahigh Dimensional Feature Space." *Journal of the Royal Statistical Society, Series B* 70:849–911.

Foster, D. P., and Stine, R. A. (2004). "Variable Selection in Data Mining: Building a Predictive Model for Bankruptcy." *Journal of the American Statistical Association* 99:303–313.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., et al. (1999). "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression." *Science* 286:531–537.

Harrell, F. E., Jr. (2001). *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. New York: Springer.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Hocking, R. R. (1976). "The Analysis and Selection of Variables in a Linear Regression." *Biometrics* 32:1–50.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Judge, G. G., Griffiths, W. E., Hill, R. C., Lütkepohl, H., and Lee, T.-C. (1985). *The Theory and Practice of Econometrics*. 2nd ed. New York: John Wiley & Sons.

Liu, J., Zhao, Z., Wang, J., and Ye, J. (2014). "Safe Screening with Variational Inequalities and Its Application to Lasso." In *JMLR Workshop and Conference Proceedings, Vol. 32: Proceedings of the Thirty-First International Conference on Machine Learning, Second Cycle*. http://proceedings.mlr.press/v32/liuc14.pdf.

Mallows, C. L. (1967). "Choosing a Subset Regression." Bell Telephone Laboratories.

Mallows, C. L. (1973). "Some Comments on C_P." *Technometrics* 15:661–675.

Miller, A. J. (2002). *Subset Selection in Regression*. Vol. 95 of Monographs on Statistics and Applied Probability. 2nd ed. Boca Raton, FL: Chapman & Hall/CRC.

Nesterov, Y. (2013). "Gradient Methods for Minimizing Composite Objective Function." *Mathematical Programming* 140:125–161.

Osborne, M. R., Presnell, B., and Turlach, B. A. (2000). "A New Approach to Variable Selection in Least Squares Problems." *IMA Journal of Numerical Analysis* 20:389–404.

Raftery, A. E., Madigan, D., and Hoeting, J. A. (1997). "Bayesian Model Averaging for Linear Regression Models." *Journal of the American Statistical Association* 92:179–191.

Reichler, J. L., ed. (1987). *The 1987 Baseball Encyclopedia Update*. New York: Macmillan.

Sarle, W. S. (2001). "Donoho-Johnstone Benchmarks: Neural Net Results." Accessed March 27, 2007. ftp://ftp.sas.com/pub/neural/dojo/dojo.html.

Sawa, T. (1978). "Information Criteria for Discriminating among Alternative Regression Models." *Econometrica* 46:1273–1282.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B* 58:267–288.

Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J., and Tibshirani, R. (2012). "Strong Rules for Discarding Predictors in Lasso-Type Problems." *Journal of the Royal Statistical Society, Series B* 74:245–266.

Time Inc. (1987). "What They Make." *Sports Illustrated* (April 20): 54–81.

Wang, J., Zhou, J., Wonka, P., and Ye, J. (2013). "Lasso Screening Rules via Dual Polytope Projection." In *Advances in Neural Information Processing Systems 26*, edited by C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, 1070–1078. La Jolla, CA: Neural Information Processing Systems Foundation.

Xiang, Z. J., Xu, H., and Ramadge, P. J. (2011). "Learning Sparse Representations of High Dimensional Data on Large Scale Dictionaries." In *Advances in Neural Information Processing Systems 24*, edited by J. Shawe-Taylor, R. S. Zemel, P. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, 900–908. La Jolla, CA: Neural Information Processing Systems Foundation.

Yuan, M., and Lin, L. (2006). "Model Selection and Estimation in Regression with Grouped Variables." *Journal of the Royal Statistical Society, Series B* 68:49–67.

Zou, H. (2006). "The Adaptive Lasso and Its Oracle Properties." *Journal of the American Statistical Association* 101:1418–1429.

Zou, H., and Hastie, T. (2005). "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society, Series B* 67:301–320.

Chapter 57
The HPCANDISC Procedure

Contents

Overview: HPCANDISC Procedure **4578**
PROC HPCANDISC Features 4579
PROC HPCANDISC Compared with PROC CANDISC 4580
Getting Started: HPCANDISC Procedure **4580**
Syntax: HPCANDISC Procedure **4585**
PROC HPCANDISC Statement 4585
BY Statement 4589
CLASS Statement 4589
FREQ Statement 4589
ID Statement 4590
PERFORMANCE Statement 4590
VAR Statement 4590
WEIGHT Statement 4590
Details: HPCANDISC Procedure **4591**
Missing Values 4591
Computational Method 4591
General Formulas 4591
Multithreading 4592
Output Data Sets 4593
OUT= Data Set 4593
OUTSTAT= Data Set 4593
Displayed Output 4595
ODS Table Names 4598
Examples: HPCANDISC Procedure **4599**
Example 57.1: Analyzing Iris Data with PROC HPCANDISC 4599
References **4604**

Overview: HPCANDISC Procedure

The HPCANDISC procedure is a high-performance procedure that performs canonical discriminant analysis. It is a high-performance version of the CANDISC procedure in SAS/STAT software.

Canonical discriminant analysis is a dimension-reduction technique related to principal component analysis and canonical correlation. The methodology that is used in deriving the canonical coefficients parallels that of a one-way multivariate analysis of variance (MANOVA). MANOVA tests for equality of the mean vector across class levels. Canonical discriminant analysis finds linear combinations of the quantitative variables that provide maximal separation between classes or groups. Given a classification variable and several quantitative variables, the HPCANDISC procedure derives *canonical variables*, which are linear combinations of the quantitative variables that summarize between-class variation in much the same way that principal components summarize total variation.

The HPCANDISC procedure performs a canonical discriminant analysis, computes squared Mahalanobis distances between class means, and performs both univariate and one-way multivariate analyses of variance. Two output data sets can be produced: one that contains the canonical coefficients and another that contains, among other things, scored canonical variables. You can rotate the canonical coefficients output data set by using the FACTOR procedure. It is customary to standardize the canonical coefficients so that the canonical variables have means that are equal to 0 and pooled within-class variances that are equal to 1. PROC HPCANDISC displays both standardized and unstandardized canonical coefficients. Correlations between the canonical variables and the original variables in addition to the class means for the canonical variables are also displayed; these correlations, sometimes known as loadings, are called *canonical structures*.

When you have two or more groups of observations that have measurements on several quantitative variables, canonical discriminant analysis derives a linear combination of the variables that has the highest possible multiple correlation with the groups. This maximal multiple correlation is called the *first canonical correlation*. The coefficients of the linear combination are the *canonical coefficients* or *canonical weights*. The variable that is defined by the linear combination is the *first canonical variable* or *canonical component*. The second canonical correlation is obtained by finding the linear combination uncorrelated with the first canonical variable that has the highest possible multiple correlation with the groups. The process of extracting canonical variables can be repeated until the number of canonical variables equals the number of original variables or the number of classes minus one, whichever is smaller.

The first canonical correlation is at least as large as the multiple correlation between the groups and any of the original variables. If the original variables have high within-group correlations, the first canonical correlation can be large even if all the multiple correlations are small. In other words, the first canonical variable can show substantial differences between the classes, even if none of the original variables do. Canonical variables are sometimes called *discriminant functions*, but this usage is ambiguous because the DISCRIM procedure produces very different functions for classification that are also called discriminant functions.

For each canonical correlation, PROC HPCANDISC tests the hypothesis that it and all smaller canonical correlations are zero in the population. An F approximation (Rao 1973; Kshirsagar 1972) is used that gives better small-sample results than the usual chi-square approximation. The variables should have an approximate multivariate normal distribution within each class, with a common covariance matrix in order for the probability levels to be valid.

Canonical discriminant analysis is equivalent to canonical correlation analysis between the quantitative variables and a set of dummy variables coded from the CLASS variable. Performing canonical discriminant

analysis is also equivalent to performing the following steps:

1. Transform the variables so that the pooled within-class covariance matrix is an identity matrix.
2. Compute class means on the transformed variables.
3. Perform a principal component analysis on the means, weighting each mean by the number of observations in the class. The eigenvalues are equal to the ratio of between-class variation to within-class variation in the direction of each principal component.
4. Back-transform the principal components into the space of the original variables to obtain the canonical variables.

An interesting property of the canonical variables is that they are uncorrelated whether the correlation is calculated from the total sample or from the pooled within-class correlations. However, the canonical coefficients are not orthogonal, so the canonical variables do not represent perpendicular directions through the space of the original variables.

PROC HPCANDISC Features

The main features of the HPCANDISC procedure are as follows:

- performs a canonical discriminant analysis, computes squared Mahalanobis distances between class means, and performs both univariate and multivariate one-way analyses of variance
- is highly multithreaded during calculations of the within-class sum-of-squares-and-crossproducts (SSCP) matrix and the canonical variable scores
- supports a FREQ statement for grouped analysis
- supports a WEIGHT statement for weighted analysis
- displays both standardized and unstandardized canonical coefficients
- displays correlations between the canonical variables and the original variables
- displays class means for the canonical variables
- produces two output data sets: one that contains the canonical coefficients and another that contains scored canonical variables

PROC HPCANDISC Compared with PROC CANDISC

The HPCANDISC procedure and the CANDISC procedure in SAS/STAT have the following similarities and differences:

- All the statements that are available in PROC CANDISC are supported in the HPCANDISC procedure.
- As input, PROC CANDISC can accept ordinary SAS data set and other types of special SAS data sets. In the HPCANDISC procedure, only the ordinary SAS data set (raw data) can be used as input.
- The HPCANDISC procedure supports an ID statement that is not available in PROC CANDISC.
- By default, PROC HPCANDISC performs computations on multiple threads. The CANDISC procedure executes on a single thread.

Getting Started: HPCANDISC Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpcangs.sas.)

The data in this example are measurements of 159 fish caught in Finland's Lake Laengelmaevesi; this data set is available from the Puranen. For each of the seven species (bream, roach, whitefish, parkki, perch, pike, and smelt), the weight, length, height, and width of each fish are tallied. Three different length measurements are recorded: from the nose of the fish to the beginning of its tail, from the nose to the notch of its tail, and from the nose to the end of its tail. The height and width are recorded as percentages of the third length variable. The fish data set is available from the Sashelp library.

The following step uses PROC HPCANDISC to find the three canonical variables that best separate the species of fish in the Sashelp.Fish data and create the output data set outcan. When the NCAN=3 option is specified, only the first three canonical variables are displayed. The ID statement adds the variable Species from the input data set to the output data set. The ODS EXCLUDE statement excludes the canonical structure tables and most of the canonical coefficient tables in order to obtain a more compact set of results. The TEMPLATE and SGRENDER procedures create a plot of the first two canonical variables. The following statements produce Figure 57.1 through Figure 57.6:

```
title 'Fish Measurement Data';

proc hpcandisc data=sashelp.fish ncan=3 out=outcan;
   ods exclude tstruc bstruc pstruc tcoef pcoef;
   id Species;
   class Species;
   var Weight Length1 Length2 Length3 Height Width;
run;

proc template;
   define statgraph scatter;
      begingraph;
```

```
            entrytitle 'Fish Measurement Data';
            layout overlayequated / equatetype=fit
               xaxisopts=(label='Canonical Variable 1')
               yaxisopts=(label='Canonical Variable 2');
               scatterplot x=Can1 y=Can2 / group=species name='fish';
               layout gridded / autoalign=(topright);
                  discretelegend 'fish' / border=false opaque=false;
               endlayout;
            endlayout;
         endgraph;
      end;
   run;

   proc sgrender data=outcan template=scatter;
   run;
```

PROC HPCANDISC begins by displaying performance information, data access information, and summary information about the variables in the analysis, as shown in Figure 57.1.

The "Performance Information" table shows the procedure executes in single-machine mode; that is, the data reside and the computation is conducted on the machine where the SAS session executes. This run of the HPCANDISC procedure took place on a multicore machine that had four CPUs; one computational thread was spawned per CPU.

The "Data Access Information" table shows that the input data set and the output data set are both accessed with the V9 (base) engine on the client machine where the MVA SAS session executes.

The summary information includes the number of observations, the number of quantitative variables in the analysis (specified using the VAR statement), and the number of class levels in the classification variable (specified using the CLASS statement). The value and frequency of each class level are also displayed.

Figure 57.1 Fish Data: Performance, Data Access, and Summary Information

Fish Measurement Data

The HPCANDISC Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Data Access Information			
Data	**Engine**	**Role**	**Path**
SASHELP.FISH	V9	Input	On Client
WORK.OUTCAN	V9	Output	On Client

Total Sample Size	158	**DF Total**	157
Variables	6	**DF Within Classes**	151
Class Levels	7	**DF Between Classes**	6

Number of Observations Read	159
Number of Observations Used	158

Figure 57.1 *continued*

Class Level Information			
Species	**Frequency**	**Weight**	**Proportion**
Bream	34	34.00000	0.21519
Parkki	11	11.00000	0.06962
Perch	56	56.00000	0.35443
Pike	17	17.00000	0.10759
Roach	20	20.00000	0.12658
Smelt	14	14.00000	0.08861
Whitefish	6	6.00000	0.03797

Figure 57.2 displays the "Multivariate Statistics and F Approximations" table. PROC HPCANDISC performs a one-way multivariate analysis of variance (one-way MANOVA) and provides four multivariate tests of the hypothesis that the class mean vectors are equal. These tests indicate that not all the mean vectors are equal ($p < 0.0001$).

Figure 57.2 Fish Data: MANOVA and Multivariate Tests

Fish Measurement Data

The HPCANDISC Procedure

Multivariate Statistics and F Approximations S=6 M=-0.5 N=72					
Statistic	**Value**	**F Value**	**Num DF**	**Den DF**	**Pr > F**
Wilks' Lambda	0.000363	90.71	36	643.89	<.0001
Pillai's Trace	3.104651	26.99	36	906	<.0001
Hotelling-Lawley Trace	52.057997	209.24	36	413.64	<.0001
Roy's Greatest Root	39.134998	984.90	6	151	<.0001
NOTE: F Statistic for Roy's Greatest Root is an upper bound.					

Figure 57.3 displays the "Canonical Correlations" table. The first canonical correlation is the greatest possible multiple correlation with the classes that you can achieve by using a linear combination of the quantitative variables. The first canonical correlation, displayed in the table, is 0.987463. The figure shows a likelihood ratio test of the hypothesis that the current canonical correlation and all smaller ones are zero. The first line is equivalent to Wilks' lambda multivariate test.

Figure 57.3 Fish Data: Canonical Correlations

Fish Measurement Data

The HPCANDISC Procedure

	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq) Eigenvalue	Difference	Proportion	Cumulative
1	0.987463	0.986671	0.001989	0.975084	39.1350	29.3859	0.7518	0.7518
2	0.952349	0.950095	0.007425	0.906969	9.7491	7.3786	0.1873	0.9390
3	0.838637	0.832518	0.023678	0.703313	2.3706	1.7016	0.0455	0.9846
4	0.633094	0.623649	0.047821	0.400809	0.6689	0.5346	0.0128	0.9974
5	0.344157	0.334170	0.070356	0.118444	0.1344	0.1343	0.0026	1.0000
6	0.005701	.	0.079806	0.000033	0.0000		0.0000	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero					
	Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
1	0.00036325	90.71	36	643.89	<.0001
2	0.01457896	46.46	25	547.58	<.0001
3	0.15671134	23.61	16	452.79	<.0001
4	0.52820347	12.09	9	362.78	<.0001
5	0.88152702	4.88	4	300	0.0008
6	0.99996749	0.00	1	151	0.9442

Figure 57.4 displays the "Raw Canonical Coefficients" table. The first canonical variable, Can1, shows that the linear combination of the centered variables Can1 = –0.0006 × Weight – 0.33 × Length1 2.49 × Length2 + 2.60 × Length3 + 1.12 × Height – 1.45 × Width separates the species most effectively.

Figure 57.4 Fish Data: Raw Canonical Coefficients

Fish Measurement Data

The HPCANDISC Procedure

Raw Canonical Coefficients			
Variable	**Can1**	**Can2**	**Can3**
Weight	-0.00064851	-0.00523	-0.00560
Length1	-0.32944	-0.62660	-2.93432
Length2	-2.48613	-0.69025	4.04504
Length3	2.59565	1.80318	-1.13926
Height	1.12198	-0.71475	0.28320
Width	-1.44639	-0.90703	0.74149

Figure 57.5 displays the "Class Means on Canonical Variables" table. PROC HPCANDISC computes the means of the canonical variables for each class. The first canonical variable is the linear combination of the variables Weight, Length1, Length2, Length3, Height, and Width that provides the greatest difference (in terms of a univariate F test) between the class means. The second canonical variable provides the greatest difference between class means while being uncorrelated with the first canonical variable.

Figure 57.5 Fish Data: Class Means for Canonical Variables

Class Means on Canonical Variables			
Species	**Can1**	**Can2**	**Can3**
Bream	10.94142	0.52078	0.23497
Parkki	2.58904	-2.54722	-0.49326
Perch	-4.47181	-1.70823	1.29281
Pike	-4.89689	8.22141	-0.16469
Roach	-0.35837	0.08734	-1.10056
Smelt	-4.09137	-2.35806	-4.03836
Whitefish	-0.39542	-0.42072	1.06459

Figure 57.6 displays a plot of the first two canonical variables, which shows that Can1 discriminates among three groups: (1) bream; (2) whitefish, roach, and parkki; and (3) smelt, pike, and perch. Can2 best discriminates between pike and the other species.

Figure 57.6 Fish Data: Plot of First Two Canonical Variables

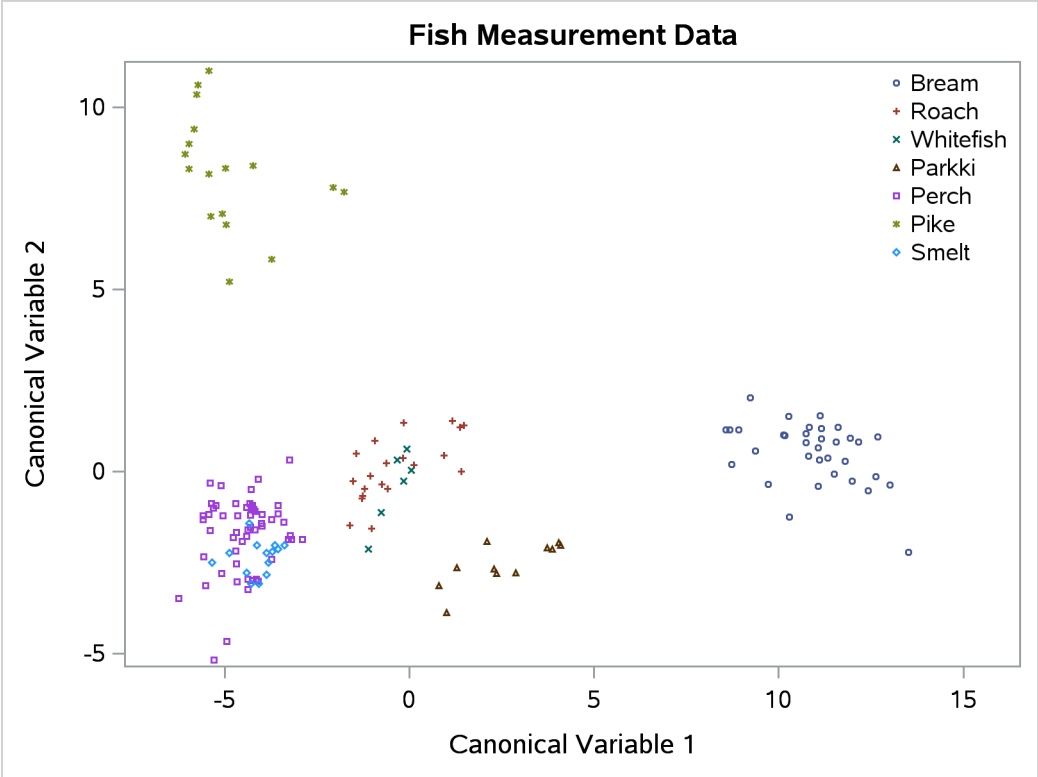

Syntax: HPCANDISC Procedure

The following statements are available in the HPCANDISC procedure:

PROC HPCANDISC < *options* > ;
  BY *variables* ;
  CLASS *variable* ;
  FREQ *variable* ;
  ID *variables* ;
  PERFORMANCE *performance-options* ;
  VAR *variables* ;
  WEIGHT *variable* ;

The PROC HPCANDISC statement and a single CLASS statement are required. All other statements are optional.

PROC HPCANDISC Statement

PROC HPCANDISC < *options* > ;

The PROC HPCANDISC statement invokes the HPCANDISC procedure. Optionally, it also identifies input and output data sets, specifies the analyses performed, and controls displayed output. Table 57.1 summarizes the options available in the PROC HPCANDISC statement.

Table 57.1 PROC HPCANDISC Statement Options

Option	Description
Specify Data Sets	
DATA=	Specifies the input data set
OUT=	Specifies the output data set that contains canonical scores
OUTSTAT=	Specifies the output statistics data set
Specify Details of Analysis	
NCAN=	Specifies the number of canonical variables
PREFIX=	Specifies a prefix for naming the canonical variables
SINGULAR=	Specifies the singularity criterion
Control Displayed Output	
ALL	Displays all output
ANOVA	Displays univariate statistics
BCORR	Displays between correlations
BCOV	Displays between covariances
BSSCP	Displays between SSCPs
DISTANCE	Displays squared Mahalanobis distances
NOPRINT	Suppresses all displayed output
PCORR	Displays pooled correlations
PCOV	Displays pooled covariances

Table 57.1 *continued*

Option	Description
PSSCP	Displays pooled SSCPs
SHORT	Suppresses some displayed output
SIMPLE	Displays simple descriptive statistics
STDMEAN	Displays standardized class means
TCORR	Displays total correlations
TCOV	Displays total covariances
TSSCP	Displays total SSCPs
WCORR	Displays within correlations
WCOV	Displays within covariances
WSSCP	Displays within SSCPs

The following list provides details about these *options*.

ALL
activates all the display options.

ANOVA
displays univariate statistics for testing the hypothesis that the class means are equal in the population for each variable.

BCORR
displays between-class correlations.

BCOV
displays between-class covariances. The between-class covariance matrix equals the between-class SSCP matrix divided by $n(c-1)/c$, where n is the number of observations and c is the number of classes. The between-class covariances should be interpreted in comparison with the total-sample and within-class covariances, not as formal estimates of population parameters.

BSSCP
displays the between-class SSCP matrix.

DATA=*SAS-data-set*
specifies the data set to be analyzed. The data set can only be an ordinary SAS data set (raw data). If you omit the DATA= option, PROC HPCANDISC uses the most recently created SAS data set.

DISTANCE
MAHALANOBIS
displays squared Mahalanobis distances between the group means, the F statistics, and the corresponding probabilities of greater squared Mahalanobis distances between the group means.

NCAN=*n*

specifies the number of canonical variables to be computed. The value of *n* must be less than or equal to the number of variables. If you specify NCAN=0, PROC HPCANDISC displays the canonical correlations but not the canonical coefficients, structures, or means. A negative value suppresses the canonical analysis entirely. Let v be the number of variables in the VAR statement, and let c be the number of classes. If you omit the NCAN= option, only $\min(v, c-1)$ canonical variables are generated; if you also specify an OUT= output data set, v canonical variables are generated, and the last $v-(c-1)$ canonical variables have missing values.

NOPRINT

suppresses the normal display of results. This option temporarily disables the Output Delivery System (ODS). For more information about ODS, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*

creates an output SAS data set to contain observationwise canonical variable scores. The variables in the input data set are *not* included in the output data set to avoid data duplication for large data sets; however, variables that are specified in the ID statement are included.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For more information about OUT= data sets, see the section "Output Data Sets" on page 4593.

OUTSTAT=*SAS-data-set*

creates a TYPE=CORR output SAS data set to contain various statistics, including class means, standard deviations, correlations, canonical correlations, canonical structures, canonical coefficients, and means of canonical variables for each class level.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PCORR

displays pooled within-class correlations (partial correlations based on the pooled within-class covariances).

PCOV

displays pooled within-class covariances.

PREFIX=*name*

specifies a prefix for naming the canonical variables. By default, the names are Can1, Can2, Can3, and so on. If you specify PREFIX=Abc, the components are named Abc1, Abc2, and so on. The number of characters in the prefix plus the number of digits required to designate the canonical variables should not exceed 32. The prefix is truncated if the combined length exceeds 32.

PSSCP

displays the pooled within-class corrected SSCP matrix.

SHORT
suppresses the display of canonical structures, canonical coefficients, and class means on canonical variables; only tables of canonical correlations and multivariate test statistics are displayed.

SIMPLE
displays simple descriptive statistics for the total sample and within each class.

SINGULAR=*p*
specifies the criterion for determining the singularity of the total-sample correlation matrix and the pooled within-class covariance matrix, where $0 < p < 1$. The default is SINGULAR=1E–8.

Let **S** be the total-sample correlation matrix. If the R square for predicting a quantitative variable in the VAR statement from the variables that precede it exceeds $1 - p$, then **S** is considered singular. If **S** is singular, the probability levels for the multivariate test statistics and canonical correlations are adjusted for the number of variables whose R square exceeds $1 - p$.

If **S** is considered singular and the inverse of **S** (squared Mahalanobis distances) is required, a quasi inverse is used instead. For more information, see the section "Quasi-inverse" on page 2745.

STDMEAN
displays total-sample and pooled within-class standardized class means.

TCORR
displays total-sample correlations.

TCOV
displays total-sample covariances.

TSSCP
displays the total-sample corrected SSCP matrix.

WCORR
displays within-class correlations for each class level.

WCOV
displays within-class covariances for each class level.

WSSCP
displays the within-class corrected SSCP matrix for each class level.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPCANDISC to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPCANDISC procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* **;**

The values of the CLASS variable define the groups for analysis. Class levels are determined by the formatted values of the CLASS variable. The CLASS variable can be numeric or character. A CLASS statement is required.

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. SAS high-performance analytics procedures that support the FREQ statement treat each observation as if it appeared f times, where f is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

The total number of observations is considered to be equal to the sum of the FREQ variable when the procedure determines degrees of freedom for significance probabilities.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are transferred to output data sets created by SAS high-performance analytics procedures, provided that the output data set produces one (or more) records per input observation.

For information about the common ID statement in SAS high-performance analytics procedures, see the section "ID Statement" on page 576.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of the HPCANDISC procedure.

The PERFORMANCE statement is documented further in the section "PERFORMANCE Statement" on page 569.

VAR Statement

VAR *variables* **;**

You specify the quantitative variables to include in the analysis by using a VAR statement. If you do not use a VAR statement, the analysis includes all numeric variables that are not listed in other statements.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations that are used in the analysis are assigned a weight of 1.

The WEIGHT statement does not alter the degrees of freedom.

Details: HPCANDISC Procedure

Missing Values

If an observation has a missing value for any of the quantitative variables, it is omitted from the analysis. If an observation has a missing CLASS value but is otherwise complete, PROC HPCANDISC does not use it in computing the canonical correlations and coefficients; however, canonical variable scores are computed for that observation for the OUT= data set.

Computational Method

General Formulas

Canonical discriminant analysis is equivalent to canonical correlation analysis between the quantitative variables and a set of dummy variables coded from the CLASS variable. In the following notation, the dummy variables are denoted by $\mathbf{y}$ and the quantitative variables are denoted by $\mathbf{x}$. The total sample covariance matrix for the $\mathbf{x}$ and $\mathbf{y}$ variables is

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_{xx} & \mathbf{S}_{xy} \\ \mathbf{S}_{yx} & \mathbf{S}_{yy} \end{bmatrix}$$

When c is the number of groups, n_t is the number of observations in group t, and $\mathbf{S}_t$ is the sample covariance matrix for the $\mathbf{x}$ variables in group t, the within-class pooled covariance matrix for the $\mathbf{x}$ variables is

$$\mathbf{S}_p = \frac{1}{\sum n_t - c} \sum (n_t - 1)\mathbf{S}_t$$

The canonical correlations, ρ_i, are the square roots of the eigenvalues, λ_i, of the following matrix. The corresponding eigenvectors are $\mathbf{v}_i$.

$$\mathbf{S}_p{}^{-1/2}\mathbf{S}_{xy}\mathbf{S}_{yy}{}^{-1}\mathbf{S}_{yx}\mathbf{S}_p{}^{-1/2}$$

Let $\mathbf{V}$ be the matrix that contains the eigenvectors $\mathbf{v}_i$ that correspond to nonzero eigenvalues as columns. The raw canonical coefficients are calculated as follows:

$$\mathbf{R} = \mathbf{S}_p{}^{-1/2}\mathbf{V}$$

The pooled within-class standardized canonical coefficients are

$$\mathbf{P} = \mathrm{diag}(\mathbf{S}_p)^{1/2}\mathbf{R}$$

The total sample standardized canonical coefficients are

$$\mathbf{T} = \mathrm{diag}(\mathbf{S}_{xx})^{1/2}\mathbf{R}$$

Let $\mathbf{X}_c$ be the matrix that contains the centered $\mathbf{x}$ variables as columns. The canonical scores can be calculated by any of the following:

$$\mathbf{X}_c\,\mathbf{R}$$

$$\mathbf{X}_c \operatorname{diag}(\mathbf{S}_p)^{-1/2}\mathbf{P}$$

$$\mathbf{X}_c \operatorname{diag}(\mathbf{S}_{xx})^{-1/2}\mathbf{T}$$

For the multivariate tests based on $\mathbf{E}^{-1}\mathbf{H}$,

$$\mathbf{E} = (n-1)(\mathbf{S}_{yy} - \mathbf{S}_{yx}\mathbf{S}_{xx}^{-1}\mathbf{S}_{xy})$$

$$\mathbf{H} = (n-1)\mathbf{S}_{yx}\mathbf{S}_{xx}^{-1}\mathbf{S}_{xy}$$

where n is the total number of observations.

Multithreading

Threading is the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading is the concurrent execution of threads. When multithreading is possible, you can realize substantial performance gains compared to the performance that you get from sequential (single-threaded) execution.

The number of threads that the HPCANDISC procedure spawns is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the CPU count by using the CPUCOUNT= SAS system option. For example, if you specify the following statement, PROC HPCANDISC schedules threads as if it were executing on a system that had four CPUs, regardless of the actual CPU count:

```
options cpucount=4;
```

- You can specify the NTHREADS= option in the PERFORMANCE statement to determine the number of threads. This specification overrides the system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads per machine is displayed in the “Performance Information” table, which is part of the default output. The HPCANDISC procedure allocates one thread per CPU.

The tasks that are multithreaded by the HPCANDISC procedure are primarily defined by dividing the data processed on a single machine among the threads; that is, PROC HPCANDISC implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and you are running on four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following:

- variable levelization
- formation of the crossproducts matrix
- canonical variable scoring of observations

Output Data Sets

OUT= Data Set

Many SAS procedures add the variables from the input data set when an observationwise output data set is created. The assumption of high-performance analytics procedures is that the input data sets can be large and can contain many variables. For performance reasons, the OUT= data set contains the following:

- new variables that are explicitly created for the OUT= data set
- variables that are listed in the ID statement

The new variables that are created for the OUT= data set contain the canonical variable scores. You determine the number of new variables by using the NCAN= option. The names of the new variables are formed as they are for the PREFIX= option. The new variables have means equal to 0 and pooled within-class variances equal to 1.

OUTSTAT= Data Set

The OUTSTAT= data set is similar to the TYPE=CORR data set that the CORR procedure produces but contains many results in addition to those produced by PROC CORR.

The OUTSTAT= data set is TYPE=CORR, and it contains the following variables:

- the BY variables, if any
- the CLASS variable
- _TYPE_, a character variable of length 8 that identifies the type of statistic
- _NAME_, a character variable of length 32 that identifies the row of the matrix or the name of the canonical variable
- the quantitative variables (those in the VAR statement, or if there is no VAR statement, all numeric variables not listed in any other statement)

The observations, as identified by the variable _TYPE_, have the following _TYPE_ values:

TYPE	Contents
N	number of observations for the total sample (CLASS variable missing) and within each class (CLASS variable present)
SUMWGT	sum of weights for the total sample (CLASS variable missing) and within each class (CLASS variable present) if a WEIGHT statement is specified
MEAN	means for the total sample (CLASS variable missing) and within each class (CLASS variable present)
STDMEAN	total-standardized class means
PSTDMEAN	pooled within-class standardized class means

STD	standard deviations for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PSTD	pooled within-class standard deviations
BSTD	between-class standard deviations
RSQUARED	univariate R squares

The following kinds of observations are identified by the combination of the variables _TYPE_ and _NAME_. When the _TYPE_ variable has one of the following values, the _NAME_ variable identifies the row of the matrix:

TYPE	**Contents**
CSSCP	corrected SSCP matrix for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PSSCP	pooled within-class corrected SSCP matrix
BSSCP	between-class SSCP matrix
COV	covariance matrix for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PCOV	pooled within-class covariance matrix
BCOV	between-class covariance matrix
CORR	correlation matrix for the total sample (CLASS variable missing) and within each class (CLASS variable present)
PCORR	pooled within-class correlation matrix
BCORR	between-class correlation matrix

When the _TYPE_ variable has one of the following values, the _NAME_ variable identifies the canonical variable:

TYPE	**Contents**
CANCORR	canonical correlations
STRUCTUR	canonical structure
BSTRUCT	between canonical structure
PSTRUCT	pooled within-class canonical structure
SCORE	total-sample standardized canonical coefficients
PSCORE	pooled within-class standardized canonical coefficients
RAWSCORE	raw canonical coefficients
CANMEAN	means of the canonical variables for each class

You can use this data set in PROC SCORE to get scores on the canonical variables for new data by using one of the following forms:

```
* The CLASS variable C is numeric;
proc score data=NewData score=Coef(where=(c = .  )) out=Scores;
run;

* The CLASS variable C is character;
proc score data=NewData score=Coef(where=(c = ' ')) out=Scores;
run;
```

The WHERE clause excludes the within-class means and standard deviations. PROC SCORE standardizes the new data by subtracting the original variable means that are stored in the _TYPE_='MEAN' observations and dividing by the original variable standard deviations from the _TYPE_='STD' observations. Then PROC SCORE multiplies the standardized variables by the coefficients from the _TYPE_='SCORE' observations to get the canonical scores.

Displayed Output

By default, the HPCANDISC procedure begins by displaying the output along with the following:

- The "Performance Information" table, which is produced by default. It displays the number of threads used for single-machine mode.
- The "Data Access Information" table, which is produced by default. For the input and output data sets, it displays the libref and data set name, the engine used to access the data, the role (input or output) of the data set, and the path that data followed to reach the computation.
- Summary information about the variables in the analysis that displays the total sample size, the number of quantitative variables, the number of class levels, and the number of degrees of freedom
- The "Number of Observations" table, which displays the number of observations read from the input data set and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used.
- The "Class Level Information" table, which displays, for each level of the classification variable, the frequency sum, weight sum, and proportion of the total sample

The optional output from PROC HPCANDISC includes the following:

- Within-class SSCP matrices for each group
- Pooled within-class SSCP matrix
- Between-class SSCP matrix
- Total-sample SSCP matrix
- Within-class covariance matrices for each group
- Pooled within-class covariance matrix

- Between-class covariance matrix, equal to the between-class SSCP matrix divided by $n(c-1)/c$, where n is the number of observations and c is the number of classes
- Total-sample covariance matrix
- Within-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the within-class population correlation coefficients are zero
- Pooled within-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the partial population correlation coefficients are zero
- Between-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the between-class population correlation coefficients are zero
- Total-sample correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the total population correlation coefficients are zero
- Simple statistics, including N (the number of observations), sum, mean, variance, and standard deviation for the total sample and within each class
- Total-sample standardized class means, obtained by subtracting the grand mean from each class mean and dividing by the total sample standard deviation
- Pooled within-class standardized class means, obtained by subtracting the grand mean from each class mean and dividing by the pooled within-class standard deviation
- Pairwise squared distances between groups
- Univariate test statistics, including total-sample standard deviations, pooled within-class standard deviations, between-class standard deviations, R square, $R^2/(1-R^2)$, F, and $\mathrm{Pr} > F$ (univariate F values and probability levels for one-way analyses of variance)
- The "Timing" table, which displays the elapsed time for each main task of the procedure, if you specify the DETAILS option in the PERFORMANCE statement

By default, PROC HPCANDISC displays these statistics:

- Multivariate statistics and F approximations, including Wilks' lambda, Pillai's trace, Hotelling-Lawley trace, and Roy's greatest root with F approximations, numerator and denominator degrees of freedom (Num DF and Den DF), and probability values ($\mathrm{Pr} > F$). Each of these four multivariate statistics tests the hypothesis that the class means are equal in the population. For more information, see the section "Multivariate Tests" on page 98.
- Canonical correlations
- Adjusted canonical correlations (Lawley 1959). These are asymptotically less biased than the raw correlations and can be negative. The adjusted canonical correlations might not be computable and are displayed as missing values if two canonical correlations are nearly equal or if some are close to zero. A missing value is also displayed if an adjusted canonical correlation is larger than a previous adjusted canonical correlation.
- Approximate standard error of the canonical correlations

- Squared canonical correlations
- Eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$. Each eigenvalue is equal to $\rho^2/(1-\rho^2)$, where ρ^2 is the corresponding squared canonical correlation and can be interpreted as the ratio of between-class variation to pooled within-class variation for the corresponding canonical variable. The table includes eigenvalues, differences between successive eigenvalues, the proportion of the sum of the eigenvalues, and the cumulative proportion.
- Likelihood ratio for the hypothesis that the current canonical correlation and all smaller ones are zero in the population. The likelihood ratio for the hypothesis that all canonical correlations equal zero is Wilks' lambda.
- Approximate F statistic based on Rao's approximation to the distribution of the likelihood ratio (Rao 1973, p. 556; Kshirsagar 1972, p. 326)
- Numerator degrees of freedom (Num DF), denominator degrees of freedom (Den DF), and $\Pr > F$, the probability level associated with the F statistic

You can suppress the following statistics by specifying the SHORT option:

- Total canonical structure, giving total-sample correlations between the canonical variables and the original variables
- Between canonical structure, giving between-class correlations between the canonical variables and the original variables
- Pooled within canonical structure, giving pooled within-class correlations between the canonical variables and the original variables
- Total-sample standardized canonical coefficients, standardized to give canonical variables that have zero mean and unit pooled within-class variance when applied to the total-sample standardized variables
- Pooled within-class standardized canonical coefficients, standardized to give canonical variables that have zero mean and unit pooled within-class variance when applied to the pooled within-class standardized variables
- Raw canonical coefficients, standardized to give canonical variables that have zero mean and unit pooled within-class variance when applied to the centered variables
- Class means on the canonical variables

ODS Table Names

PROC HPCANDISC assigns a name to each table that it creates. You can use these names to reference the ODS table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 57.2. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 57.2 ODS Tables Produced by PROC HPCANDISC

Table Name	Description	Required Statement / Option
ANOVA	Univariate statistics	ANOVA
AveRSquare	Average R square	ANOVA
BCorr	Between-class correlations	BCORR
BCov	Between-class covariances	BCOV
BSSCP	Between-class SSCP matrix	BSSCP
BStruc	Between canonical structure	Default
CanCorr	Canonical correlations	Default
CanonicalMeans	Class means on canonical variables	Default
ClassLevels	Class level information	Default
Counts	Number of observations, variables, class levels, *df*	Default
DataAccessInfo	Information about modes of data access	Default
Dist	Squared distances	DISTANCE
DistFValues	*F* statistics based on squared distances	DISTANCE
DistProb	Probabilities for *F* statistics from squared distances	DISTANCE
MultStat	MANOVA	Default
NObs	Number of observations	Default
PCoef	Pooled standard canonical coefficients	Default
PCorr	Pooled within-class correlations	PCORR
PCov	Pooled within-class covariances	PCOV
PerformanceInfo	Information about the high-performance computing environment	Default
PSSCP	Pooled within-class SSCP matrix	PSSCP
PStdMeans	Pooled standardized class means	STDMEAN
PStruc	Pooled within canonical structure	Default
RCoef	Raw canonical coefficients	Default
SimpleStatistics	Simple statistics	SIMPLE
TCoef	Total-sample standard canonical coefficients	Default
TCorr	Total-sample correlations	TCORR
TCov	Total-sample covariances	TCOV
Timing	Absolute and relative times for tasks performed by the procedure	PERFORMANCE DETAILS
TSSCP	Total-sample SSCP matrix	TSSCP
TStdMeans	Total standardized class means	STDMEAN
TStruc	Total canonical structure	Default

Table 57.2 *continued*

Table Name	Description	Required Statement / Option
WCorr	Within-class correlations	WCORR
WCov	Within-class covariances	WCOV
WSSCP	Within-class SSCP matrices	WSSCP

Examples: HPCANDISC Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 57.1: Analyzing Iris Data with PROC HPCANDISC

(View the complete code for this example (hpcanex1.sas) in the example repository.)

The iris data that were published by Fisher (1936) have been widely used for examples in discriminant analysis and cluster analysis. The sepal length, sepal width, petal length, and petal width are measured in millimeters in 50 iris specimens from each of three species: *Iris setosa*, *I. versicolor*, and *I. virginica*. The iris data set is available from the Sashelp library.

This example is a canonical discriminant analysis that creates an output data set that contains scores on the canonical variables and plots the canonical variables. The ID statement is specified to add the variable Species from the input data set to the output data set.

The following statements produce Output 57.1.1 through Output 57.1.6:

```
title 'Fisher (1936) Iris Data';

proc hpcandisc data=sashelp.iris out=outcan distance anova;
   id Species;
   class Species;
   var SepalLength SepalWidth PetalLength PetalWidth;
run;
```

Output 57.1.1 displays performance information, data access information, and summary information about the observations and the classes in the data set.

Output 57.1.1 Iris Data: Performance, Data Access, and Summary Information

Fisher (1936) Iris Data

The HPCANDISC Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Output 57.1.1 *continued*

Data Access Information			
Data	**Engine**	**Role**	**Path**
SASHELP.IRIS	V9	Input	On Client
WORK.OUTCAN	V9	Output	On Client

Total Sample Size	150	**DF Total**	149
Variables	4	**DF Within Classes**	147
Class Levels	3	**DF Between Classes**	2

Number of Observations Read	150
Number of Observations Used	150

Class Level Information			
Species	**Frequency**	**Weight**	**Proportion**
Setosa	50	50.00000	0.33333
Versicolor	50	50.00000	0.33333
Virginica	50	50.00000	0.33333

Output 57.1.2 shows results from the DISTANCE option in the PROC HPCANDISC statement, which display squared Mahalanobis distances between class means.

Output 57.1.2 Iris Data: Squared Mahalanobis Distances and Distance Statistics

Fisher (1936) Iris Data

The HPCANDISC Procedure

Squared Distance to Species			
From Species	**Setosa**	**Versicolor**	**Virginica**
Setosa	0	89.86419	179.38471
Versicolor	89.86419	0	17.20107
Virginica	179.38471	17.20107	0

F Statistics, Num DF=4, Den DF=144 for Squared Distance to Species			
From Species	**Setosa**	**Versicolor**	**Virginica**
Setosa	0	550.18889	1098.27375
Versicolor	550.18889	0	105.31265
Virginica	1098.27375	105.31265	0

Prob > Mahalanobis Distance for Squared Distance to Species			
From Species	**Setosa**	**Versicolor**	**Virginica**
Setosa	1.0000	<.0001	<.0001
Versicolor	<.0001	1.0000	<.0001
Virginica	<.0001	<.0001	1.0000

Output 57.1.3 displays univariate and multivariate statistics. The ANOVA option uses univariate statistics to test the hypothesis that the class means are equal. The resulting R-square values range from 0.4008 for SepalWidth to 0.9414 for PetalLength, and each variable is significant at the 0.0001 level. The multivariate test for differences between the class levels (which is displayed by default) is also significant at the 0.0001 level; you would expect this from the highly significant univariate test results.

Output 57.1.3 Iris Data: Univariate and Multivariate Statistics

Fisher (1936) Iris Data

The HPCANDISC Procedure

Univariate Test Statistics

F Statistics, Num DF=2, Den DF=147

Variable	Label	Total Standard Deviation	Pooled Standard Deviation	Between Standard Deviation	R-Square	R-Square / (1-Rsq)	F Value	Pr > F
SepalLength	Sepal Length (mm)	8.28066	5.14789	7.95061	0.6187	1.6226	119.26	<.0001
SepalWidth	Sepal Width (mm)	4.35866	3.39688	3.36822	0.4008	0.6688	49.16	<.0001
PetalLength	Petal Length (mm)	17.65298	4.30334	20.90700	0.9414	16.0566	1180.16	<.0001
PetalWidth	Petal Width (mm)	7.62238	2.04650	8.96735	0.9289	13.0613	960.01	<.0001

Average R-Square	
Unweighted	0.7224358
Weighted by Variance	0.8689444

Multivariate Statistics and F Approximations

S=2 M=0.5 N=71

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.023439	199.15	8	288	<.0001
Pillai's Trace	1.191899	53.47	8	290	<.0001
Hotelling-Lawley Trace	32.477320	582.20	8	203.4	<.0001
Roy's Greatest Root	32.191929	1166.96	4	145	<.0001

NOTE: F Statistic for Roy's Greatest Root is an upper bound.

NOTE: F Statistic for Wilks' Lambda is exact.

Output 57.1.4 displays canonical correlations and eigenvalues. The R square between Can1 and the CLASS variable, 0.969872, is much larger than the corresponding R square for Can2, 0.222027.

Output 57.1.4 Iris Data: Canonical Correlations and Eigenvalues

Fisher (1936) Iris Data

The HPCANDISC Procedure

					Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq)			
	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalue	Difference	Proportion	Cumulative
1	0.984821	0.984508	0.002468	0.969872	32.1919	31.9065	0.9912	0.9912
2	0.471197	0.461445	0.063734	0.222027	0.2854		0.0088	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero					
	Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
1	0.02343863	199.15	8	288	<.0001
2	0.77797337	13.79	3	145	<.0001

Output 57.1.5 displays correlations between canonical and original variables.

Output 57.1.5 Iris Data: Correlations between Canonical and Original Variables

Fisher (1936) Iris Data

The HPCANDISC Procedure

Total Canonical Structure			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	0.79189	0.21759
SepalWidth	Sepal Width (mm)	-0.53076	0.75799
PetalLength	Petal Length (mm)	0.98495	0.04604
PetalWidth	Petal Width (mm)	0.97281	0.22290

Between Canonical Structure			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	0.99147	0.13035
SepalWidth	Sepal Width (mm)	-0.82566	0.56417
PetalLength	Petal Length (mm)	0.99975	0.02236
PetalWidth	Petal Width (mm)	0.99404	0.10898

Pooled Within Canonical Structure			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	0.22260	0.31081
SepalWidth	Sepal Width (mm)	-0.11901	0.86368
PetalLength	Petal Length (mm)	0.70607	0.16770
PetalWidth	Petal Width (mm)	0.63318	0.73724

Output 57.1.6 displays canonical coefficients. The raw canonical coefficients for the first canonical variable, Can1, show that the class levels differ most widely on the linear combination of the centered variables: $-0.0829378 \times \text{SepalLength} - 0.153447 \times \text{SepalWidth} + 0.220121 \times \text{PetalLength} + 0.281046 \times \text{PetalWidth}$.

Output 57.1.6 Iris Data: Canonical Coefficients

Fisher (1936) Iris Data

The HPCANDISC Procedure

Total-Sample Standardized Canonical Coefficients			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	-0.68678	0.01996
SepalWidth	Sepal Width (mm)	-0.66883	0.94344
PetalLength	Petal Length (mm)	3.88580	-1.64512
PetalWidth	Petal Width (mm)	2.14224	2.16414

Pooled Within-Class Standardized Canonical Coefficients			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	-0.42695	0.01241
SepalWidth	Sepal Width (mm)	-0.52124	0.73526
PetalLength	Petal Length (mm)	0.94726	-0.40104
PetalWidth	Petal Width (mm)	0.57516	0.58104

Raw Canonical Coefficients			
Variable	**Label**	**Can1**	**Can2**
SepalLength	Sepal Length (mm)	-0.08294	0.00241
SepalWidth	Sepal Width (mm)	-0.15345	0.21645
PetalLength	Petal Length (mm)	0.22012	-0.09319
PetalWidth	Petal Width (mm)	0.28105	0.28392

Output 57.1.7 displays class means on canonical variables.

Output 57.1.7 Iris Data: Canonical Means

Class Means on Canonical Variables		
Species	**Can1**	**Can2**
Setosa	-7.60760	0.21513
Versicolor	1.82505	-0.72790
Virginica	5.78255	0.51277

The TEMPLATE and SGRENDER procedures are used to create a plot of the first two canonical variables. The following statements produce Output 57.1.8:

```
proc template;
   define statgraph scatter;
      begingraph;
         entrytitle 'Fisher (1936) Iris Data';
         layout overlayequated / equatetype=fit
            xaxisopts=(label='Canonical Variable 1')
            yaxisopts=(label='Canonical Variable 2');
            scatterplot x=Can1 y=Can2 / group=species name='iris';
            layout gridded / autoalign=(topleft);
               discretelegend 'iris' / border=false opaque=false;
```

```
            endlayout;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=outcan template=scatter;
run;
```

Output 57.1.8 Iris Data: Plot of First Two Canonical Variables

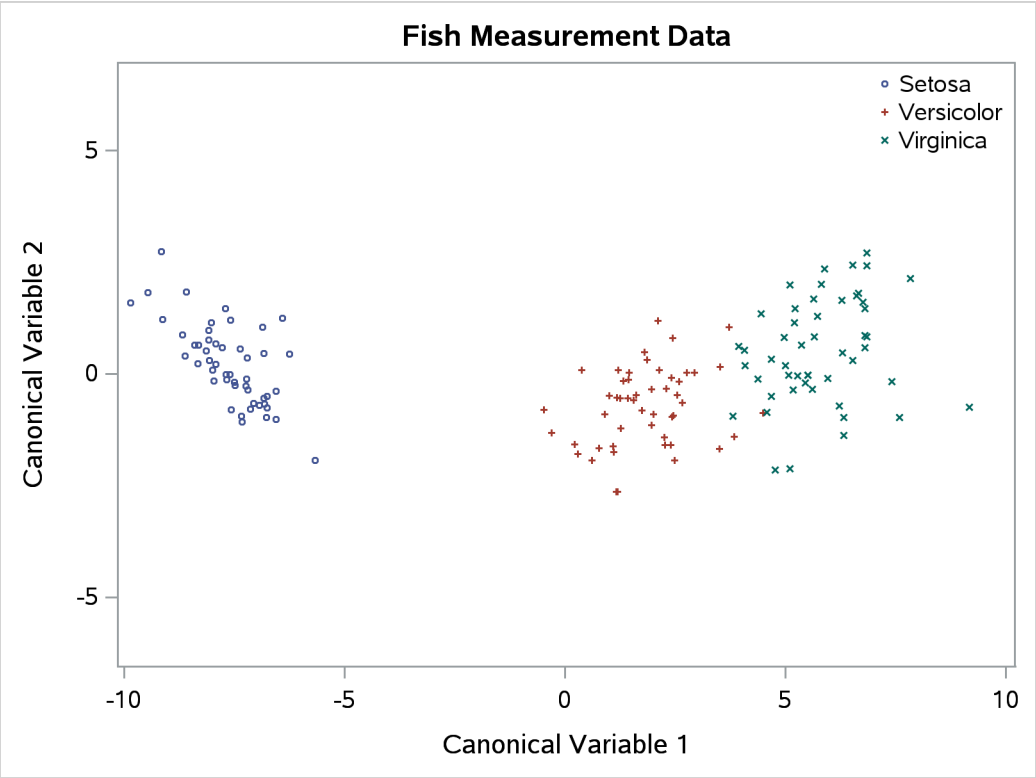

The plot of canonical variables in Output 57.1.8 shows that of the two canonical variables, Can1 has more discriminatory power.

References

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Kshirsagar, A. M. (1972). *Multivariate Analysis*. New York: Marcel Dekker.

Lawley, D. N. (1959). "Tests of Significance in Canonical Analysis." *Biometrika* 46:59–66.

Puranen, J. (1917). "Fish Catch data set (1917)." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Rao, C. R. (1973). *Linear Statistical Inference and Its Applications*. 2nd ed. New York: John Wiley & Sons.

Chapter 58
The HPFMM Procedure

Contents

Overview: HPFMM Procedure . . . **4608**
- Basic Features . . . 4609
- PROC HPFMM Contrasted with PROC FMM . . . 4610
- Assumptions . . . 4610
- Notation for the Finite Mixture Model . . . 4611
  - Homogeneous Mixtures . . . 4611
  - Special Mixtures . . . 4612

Getting Started: HPFMM Procedure . . . **4612**
- Mixture Modeling for Binomial Overdispersion: “Student,” Pearson, Beer, and Yeast . 4612
- Modeling Zero-Inflation: Is It Better to Fish Poorly or Not to Have Fished at All? . . . 4618
- Looking for Multiple Modes: Are Galaxies Clustered? . . . 4626
  - Comparison with Roeder’s Method . . . 4633

Syntax: HPFMM Procedure . . . **4636**
- PROC HPFMM Statement . . . 4636
- BAYES Statement . . . 4648
- BY Statement . . . 4657
- CLASS Statement . . . 4658
- FREQ Statement . . . 4658
- ID Statement . . . 4658
- MODEL Statement . . . 4659
  - Response Variable Options . . . 4660
  - Model Options . . . 4662
- OUTPUT Statement . . . 4667
- PERFORMANCE Statement . . . 4669
- PROBMODEL Statement . . . 4670
- RESTRICT Statement . . . 4672
- WEIGHT Statement . . . 4674

Details: HPFMM Procedure . . . **4674**
- A Gentle Introduction to Finite Mixture Models . . . 4674
  - The Form of the Finite Mixture Model . . . 4674
  - Mixture Models Contrasted with Mixing and Mixed Models: Untangling the Terminology Web . . . 4675
  - Overdispersion . . . 4676
- Log-Likelihood Functions for Response Distributions . . . 4677
- Bayesian Analysis . . . 4684
  - Conjugate Sampling . . . 4684

Metropolis-Hastings Algorithm 4684
Latent Variables via Data Augmentation 4685
Prior Distributions 4686
Parameterization of Model Effects 4687
Computational Method 4688
Multithreading 4688
Choosing an Optimization Algorithm 4689
First- or Second-Order Algorithms 4689
Algorithm Descriptions 4690
Output Data Set 4692
Default Output 4693
Performance Information 4693
Model Information 4693
Class Level Information 4693
Number of Observations 4693
Response Profile 4694
Default Output for Maximum Likelihood 4694
Default Output for Bayes Estimation 4696
ODS Table Names 4697
ODS Graphics 4699
Examples: HPFMM Procedure **4700**
Example 58.1: Modeling Mixing Probabilities: All Mice Are Equal, but Some Mice Are More Equal Than Others 4700
Example 58.2: The Usefulness of Custom Starting Values: When Do Cows Eat? . . . 4709
Example 58.3: Enforcing Homogeneity Constraints: Count and Dispersion—It Is All Over! 4718
References **4723**

Overview: HPFMM Procedure

The HPFMM procedure is a high-performance counterpart of the FMM procedure that fits statistical models to data for which the distribution of the response is a finite mixture of univariate distributions—that is, each response comes from one of several random univariate distributions that have unknown probabilities. You can use PROC HPFMM to model the component distributions in addition to the mixing probabilities. For more precise definitions and a discussion of similar but distinct modeling methodologies, see the section "A Gentle Introduction to Finite Mixture Models" on page 4674.

The HPFMM procedure is designed to fit finite mixtures of regression models or finite mixtures of generalized linear models in which the covariates and regression structure can be the same across components or can be different. You can fit finite mixture models by maximum likelihood or Bayesian methods. Note that classical statistical models are a special case of the finite mixture models in which the distribution of the data has only a single component.

Basic Features

The HPFMM procedure estimates the parameters in univariate finite mixture models and produces various statistics to evaluate parameters and model fit. The following list summarizes some basic features of the HPFMM procedure:

- maximum likelihood estimation for all models
- Markov chain Monte Carlo estimation for many models, including zero-inflated Poisson models
- many built-in link and distribution functions for modeling, including the beta, shifted *t*, Weibull, beta-binomial, and generalized Poisson distributions, in addition to many standard members of the exponential family of distributions
- specialized built-in mixture models such as the binomial cluster model (Morel and Nagaraj 1993; Morel and Neerchal 1997; Neerchal and Morel 1998)
- acceptance of multiple MODEL statements to build mixture models in which the model effects, distributions, or link functions vary across mixture components
- model-building syntax using CLASS and effect-based MODEL statements familiar from many other SAS/STAT procedures (for example, the GLM, GLIMMIX, and MIXED procedures)
- evaluation of sequences of mixture models when you specify ranges for the number of components
- simple syntax to impose linear equality and inequality constraints among parameters
- ability to model regression and classification effects in the mixing probabilities through the PROB-MODEL statement
- ability to incorporate full or partially known component membership into the analysis through the PARTIAL= option in the PROC HPFMM statement
- OUTPUT statement that produces a SAS data set with important statistics for interpreting mixture models, such as component log likelihoods and prior and posterior probabilities
- ability to add zero-inflation to any model
- output data set with posterior parameter values for the Markov chain
- multithreading optimization and Monte Carlo sampling

The HPFMM procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the HPFMM procedure, see the PLOTS options in the PROC HPFMM statement.

Because the HPFMM procedure is a high-performance analytical procedure, it also exploits all the available cores and concurrent threads.

PROC HPFMM Contrasted with PROC FMM

For general contrasts between SAS high-performance analytical procedures and other SAS procedures, see the section "Common Features of SAS High-Performance Statistical Procedures" on page 572.

The HPFMM procedure is somewhat distinct from other high-performance analytical procedures in being very nearly a twin of its counterpart, PROC FMM. You can fit many of the same kinds of models and get the same kinds of tabular, graphical, and data set results from PROC HPFMM as from PROC FMM.

PROC HPFMM and PROC FMM have several differences because of their respective underlying technology:

- The ORDER option that specifies the sort order for the levels of CLASS variables is not available in the PROC statement of the HPFMM procedure. Instead the HPFMM procedure makes this option available in the CLASS statement.
- The CLASS statement in the HPFMM procedure provides many more options than the CLASS statement in the FMM procedure.
- The PERFORMANCE statement in the HPFMM procedure includes a superset of the options that are available in the PERFORMANCE statement in the FMM procedure.
- The NOVAR option in the OUTPUT statement in the FMM procedure is not available in the OUTPUT statement of the HPFMM procedure.

The OUTPUT statement in PROC HPFMM produces observationwise statistics. However, as is customary for SAS high-performance analytical procedures, PROC HPFMM's OUTPUT statement does not by default include the input and BY variables in the output data set. This is to avoid data duplication for large data sets. In order to include any input or BY variables in the output data set, you must list these variables in the ID statement. Furthermore, PROC HPFMM's OUTPUT statement includes the predicted values of the response variable if you do not specify any output statistics.

In contrast, when you request that the posterior sample be saved to a SAS data by specifying the OUTPOST= option in the BAYES statement, PROC HPFMM includes the BY variables in the data set.

Assumptions

The HPFMM procedure makes the following assumptions in fitting statistical models:

- The number of components k in the finite mixture is known a priori and is not a parameter to be estimated.
- The parameters of the components are distinct a priori.
- The observations are uncorrelated.

Notation for the Finite Mixture Model

The general expression for the finite mixture model fitted with the HPFMM procedure is as follows:

$$f(y) = \sum_{j=1}^{k} \pi_j(\mathbf{z}, \boldsymbol{\alpha}_j) p_j(y; \mathbf{x}_j' \boldsymbol{\beta}_j, \phi_j)$$

The number of components in the mixture is denoted as k. The mixture probabilities π_j can depend on regressor variables $\mathbf{z}$ and parameters $\boldsymbol{\alpha}_j$. By default, the HPFMM procedure models these probabilities using a logit transform if $k = 2$ and as a generalized logit model if $k > 2$. The component distributions p_j can also depend on regressor variables in $\mathbf{x}_j$, regression parameters $\boldsymbol{\beta}_j$, and possibly scale parameters ϕ_j. Notice that the component distributions p_j are indexed by j since the distributions might belong to different families. For example, in a two-component model, you might model one component as a normal (Gaussian) variable and the second component as a variable with a t distribution with low degrees of freedom to manage overdispersion.

The mixture probabilities π_j satisfy $\pi_j \geq 0$, for all j, and

$$\sum_{j=1}^{k} \pi_j(\mathbf{z}, \boldsymbol{\alpha}_j) = 1$$

Homogeneous Mixtures

If the component distributions are of the same distributional form, the mixture is called homogeneous. In most applications of homogeneous mixtures, the mixing probabilities do not depend on regression parameters. The general model then simplifies to

$$f(y) = \sum_{j=1}^{k} \pi_j p(y; \mathbf{x}' \boldsymbol{\beta}_j, \phi_j)$$

Since the component distributions depend on regression parameters $\boldsymbol{\beta}_j$, this model is known as a homogeneous regression mixture. A homogeneous regression mixture assumes that the regression effects are the same across the components, although the HPFMM procedure does not impose such a restriction. If the component distributions do not contain regression effects, the model

$$f(y) = \sum_{j=1}^{k} \pi_j p(y; \mu_j, \phi_j)$$

is *the* homogeneous mixture model. A classical case is the estimation of a continuous density as a k-component mixture of normal distributions.

Special Mixtures

The HPFMM procedure enables you to fit several special mixture models. The Morel-Neerchal binomial cluster model (Morel and Nagaraj 1993; Morel and Neerchal 1997; Neerchal and Morel 1998) is a mixture of binomial distributions in which the success probabilities depend on the mixing probabilities.

Zero-inflated count models are obtained as two-component mixtures where one component is a classical count model—such as the Poisson or negative binomial model—and the other component is a distribution that is concentrated at zero. If the nondegenerate part of this special mixture is a zero-truncated model, the resulting two-component mixture is known as a hurdle model (Cameron and Trivedi 1998).

Getting Started: HPFMM Procedure

Mixture Modeling for Binomial Overdispersion: "Student," Pearson, Beer, and Yeast

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpfmmgs1.sas`.)

The following example demonstrates how you can model a complicated, two-component binomial mixture distribution, either with maximum likelihood or with Bayesian methods, with a few simple PROC HPFMM statements.

William Sealy Gosset, a chemist at the Arthur Guinness Son and Company brewery in Dublin, joined the statistical laboratory of Karl Pearson in 1906–1907 to study statistics. At first Gosset—who published all but one paper under the pseudonym "Student" because his employer forbade publications by employees after a co-worker had disclosed trade secrets—worked on the Poisson limit to the binomial distribution, using haemacytometer yeast cell counts. Gosset's interest in studying small-sample (and limit) problems was motivated by the small sample sizes he typically saw in his work at the brewery.

Subsequently, Gosset's yeast count data have been examined and revisited by many authors. In 1915, Karl Pearson undertook his own examination and realized that the variability in "Student's" data exceeded that consistent with a Poisson distribution. Pearson (1915) bemoans the fact that if this were so, "it is certainly most unfortunate that such material should have been selected to illustrate Poisson's limit to the binomial."

Using a count of Gosset's yeast cell counts on the 400 squares of a haemacytometer (Table 58.1), Pearson argues that a mixture process would explain the heterogeneity (beyond the Poisson).

Table 58.1 "Student's" Yeast Cell Counts

Number of Cells	0	1	2	3	4	5
Frequency	213	128	37	18	3	1

Pearson fits various models to these data, chief among them a mixture of two binomial series

$$\nu_1(p_1 + q_1)^\theta + \nu_2(p_2 + q_2)^\theta$$

where θ is real-valued and thus the binomial series expands to

$$(p+q)^{\theta} = \sum_{k=0}^{\infty} \frac{\Gamma(\theta+1)}{\Gamma(k+1)\Gamma(\theta-k+1)} p^k q^{\theta-k}$$

Pearson's fitted model has $\theta = 4.89997$, $\nu_1 = 356.986$, $\nu_2 = 43.014$ (corresponding to a mixing proportion of $356.986/(43.014 + 356.986) = 0.892$), and estimated success probabilities in the binomial components of 0.1017 and 0.4514, respectively. The success probabilities indicate that although the data have about a 90% chance of coming from a distribution with small success probability of about 0.1, there is a 10% chance of coming from a distribution with a much larger success probability of about 0.45.

If θ is an integer, the binomial series is the cumulative mass function of a binomial random variable. The value of θ suggests that a suitable model for these data could also be constructed as a two-component mixture of binomial random variables as follows:

$$f(y) = \pi \,\mathrm{binomial}(5, \mu_1) + (1-\pi)\,\mathrm{binomial}(5, \mu_2)$$

The binomial sample size n=5 is suggested by Pearson's estimate of $\theta = 4.89997$ and the fact that the largest cell count in Table 58.1 is 5.

The following DATA step creates a SAS data set from the data in Table 58.1.

```
data yeast;
   input count f;
   n = 5;
   datalines;
   0     213
   1     128
   2      37
   3      18
   4       3
   5       1
;
```

The two-component binomial model is fit with the HPFMM procedure with the following statements:

```
proc hpfmm data=yeast;
   model count/n =  / k=2;
   freq f;
run;
```

Because the events/trials syntax is used in the MODEL statement, PROC HPFMM defaults to the binomial distribution. The K=2 option specifies that the number of components is fixed and known to be two. The FREQ statement indicates that the data are grouped; for example, the first observation represents 213 squares on the haemacytometer where no yeast cells were found.

The "Model Information" and "Number of Observations" tables in Figure 58.1 convey that the fitted model is a two-component homogeneous binomial mixture with a logit link function. The mixture is *homogeneous* because there are no model effects in the MODEL statement and because both component distributions belong to the same distributional family. By default, PROC HPFMM estimates the model parameters by maximum likelihood.

Although only six observations are read from the data set, the data represent 400 observations (squares on the haemacytometer). Since a constant binomial sample size of 5 is assumed, the data represent 273 successes (finding a yeast cell) out of 2,000 Bernoulli trials.

Figure 58.1 Model Information for Yeast Cell Model

The HPFMM Procedure

Model Information	
Data Set	WORK.YEAST
Response Variable (Events)	count
Response Variable (Trials)	n
Frequency Variable	f
Type of Model	Homogeneous Mixture
Distribution	Binomial
Components	2
Link Function	Logit
Estimation Method	Maximum Likelihood

Number of Observations Read	6
Number of Observations Used	6
Sum of Frequencies Read	400
Sum of Frequencies Used	400
Number of Events	273
Number of Trials	2000

The estimated intercepts (on the logit scale) for the two binomial means are –2.2316 and –0.2974, respectively. These values correspond to binomial success probabilities of 0.09695 and 0.4262, respectively (Figure 58.2). The two components mix with probabilities 0.8799 and $1 - 0.8799 = 0.1201$. These values are generally close to the values found by Pearson (1915) using infinite binomial series instead of binomial mass functions.

Figure 58.2 Maximum Likelihood Estimates

Parameter Estimates for Binomial Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|	Inverse Linked Estimate
1	Intercept	-2.2316	0.1522	-14.66	<.0001	0.09695
2	Intercept	-0.2974	0.3655	-0.81	0.4158	0.4262

Parameter Estimates for Mixing Probabilities

		Linked Scale			
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.8799	1.9913	0.5725	3.48	0.0005
2	0.1201	-1.9913			

To obtain fitted values and other observationwise statistics under the stipulated two-component model, you can add the OUTPUT statement to the previous PROC HPFMM run. The following statements request componentwise predicted values and the posterior probabilities:

```
proc hpfmm data=yeast;
   model count/n =  / k=2;
   freq f;
   id f n;
```

```
   output out=hpfmmout pred(components) posterior;
run;
data hpfmmout;
   set hpfmmout;
   PredCount_1 = post_1 * f;
   PredCount_2 = post_2 * f;
run;
proc print data=hpfmmout;
run;
```

The DATA step following the PROC HPFMM step computes the predicted cell counts in each component (Figure 58.3). Note that the The predicted means in the components, 0.48476 and 2.13099, are close to the values determined by Pearson (0.4983 and 2.2118), as are the predicted cell counts.

Figure 58.3 Predicted Cell Counts

Obs	f	n	Pred_1	Pred_2	Post_1	Post_2	PredCount_1	PredCount_2
1	213	5	0.096951	0.42620	0.98606	0.01394	210.030	2.9698
2	128	5	0.096951	0.42620	0.91089	0.08911	116.594	11.4058
3	37	5	0.096951	0.42620	0.59638	0.40362	22.066	14.9341
4	18	5	0.096951	0.42620	0.17598	0.82402	3.168	14.8323
5	3	5	0.096951	0.42620	0.02994	0.97006	0.090	2.9102
6	1	5	0.096951	0.42620	0.00444	0.99556	0.004	0.9956

Gosset, who was interested in small-sample statistical problems, investigated the use of prior knowledge in mathematical-statistical analysis—for example, deriving the sampling distribution of the correlation coefficient after having assumed a uniform prior distribution for the coefficient in the population (Aldrich 1997). Pearson also was not opposed to using prior information, especially uniform priors that reflect "equal distribution of ignorance." Fisher, on the other hand, would not have any of it: the best estimator in his opinion is obtained by a criterion that is absolutely independent of prior assumptions about probabilities of particular values. He objected to the insinuation that his derivations in the work on the correlation were deduced from Bayes theorem (Fisher 1921).

The preceding analysis of the yeast cell count data uses maximum likelihood methods that are free of prior assumptions. The following analysis takes instead a Bayesian approach, assuming a beta prior distribution for the binomial success probabilities and a uniform prior distribution for the mixing probabilities. The changes from the previous run of PROC HPFMM are the addition of the ODS GRAPHICS, PERFORMANCE, and BAYES statements and the SEED=12345 option.

```
ods graphics on;
proc hpfmm data=yeast seed=12345;
   model count/n = / k=2;
   freq f;
   performance nthreads=2;
   bayes;
run;
ods graphics off;
```

When ODS Graphics is enabled, PROC HPFMM produces diagnostic trace plots for the posterior samples. Bayesian analyses are sensitive to the random number seed and thread count; the SEED= and NTHREADS= options in the PERFORMANCE statement ensure consistent results for the purposes of this example. The SEED=12345 option in the PROC HPFMM statement determines the random number seed for the random

number generator that the analysis used. The NTHREADS=2 option in the PERFORMANCE statement sets the number of threads to be used by the procedure to two. The BAYES statement requests a Bayesian analysis.

The "Bayes Information" table in Figure 58.4 provides basic information about the Markov chain Monte Carlo sampler. Because the model is a homogeneous mixture, the HPFMM procedure applies an efficient conjugate sampling algorithm with a posterior sample size of 10,000 samples after a burn-in size of 2,000 samples. The "Prior Distributions" table displays the prior distribution for each parameter along with its mean and variance and the initial value in the chain. Notice that in this situation all three prior distributions reduce to a uniform distribution on (0, 1).

Figure 58.4 Basic Information about MCMC Sampler

The HPFMM Procedure

Bayes Information	
Sampling Algorithm	Conjugate
Data Augmentation	Latent Variable
Initial Values of Chain	Data Based
Burn-In Size	2000
MC Sample Size	10000
MC Thinning	1
Parameters in Sampling	3
Mean Function Parameters	2
Scale Parameters	0
Mixing Prob Parameters	1

Prior Distributions

Component	Parameter	Distribution	Mean	Variance	Initial Value
1	**Success Probability**	Beta(1, 1)	0.5000	0.08333	0.1365
2	**Success Probability**	Beta(1, 1)	0.5000	0.08333	0.1365
1	**Probability**	Dirichlet(1, 1)	0.5000	0.08333	0.6180

The HPFMM procedure produces a log note for this model, indicating that the sampled quantities are not the linear predictors on the logit scale, but are the actual population parameters (on the data scale):

```
NOTE: Bayesian results for this model (no regressor variables,
      non-identity link) are displayed on the data scale, not the
      linked scale. You can obtain results on the linked (=linear)
      scale by requesting a Metropolis-Hastings sampling algorithm.
```

The trace panel for the success probability in the first binomial component is shown in Figure 58.5. Note that the first component in this Bayesian analysis corresponds to the second component in the MLE analysis. The graphics in this panel can be used to diagnose the convergence of the Markov chain. If the chain has not converged, inferences cannot be made based on quantities derived from the chain. You generally look for the following:

- a smooth unimodal distribution of the posterior estimates in the density plot displayed on the lower right

- good mixing of the posterior samples in the trace plot at the top of the panel (good mixing is indicated when the trace traverses the support of the distribution and appears to have reached a stationary distribution)

Figure 58.5 Trace Panel for Success Probability in First Component

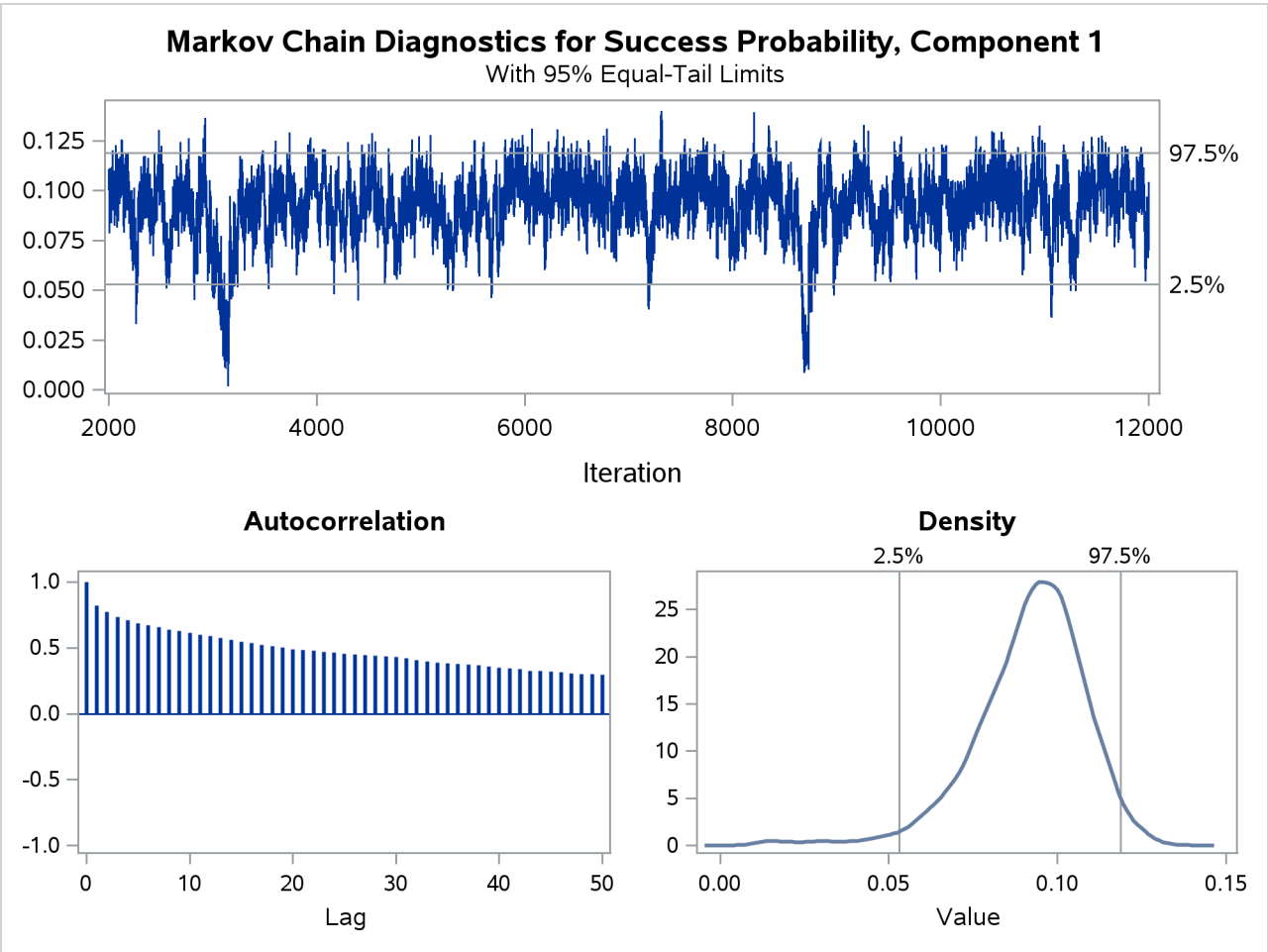

The autocorrelation plot in Figure 58.5 shows fairly high and sustained autocorrelation among the posterior estimates. While this is generally not a problem, you can affect the degree of autocorrelation among the posterior estimates by running a longer chain and thinning the posterior estimates; see the NMC= and THIN= options in the BAYES statement.

Both the trace plot and the density plot in Figure 58.5 are indications of successful convergence.

Figure 58.6 reports selected results that summarize the 10,000 posterior samples. The arithmetic means of the success probabilities in the two components are 0.0917 and 0.3974, respectively. The posterior mean of the mixing probability is 0.8312. These values are similar to the maximum likelihood parameter estimates in Figure 58.2 (after swapping components).

Figure 58.6 Summaries for Posterior Estimates

Posterior Summaries							
					Percentiles		
Component	Parameter	N	Mean	Standard Deviation	25	50	75
1	Success Probability	10000	0.0917	0.0168	0.0830	0.0938	0.1027
2	Success Probability	10000	0.3974	0.0871	0.3379	0.3957	0.4556
1	Probability	10000	0.8312	0.1045	0.7986	0.8578	0.8984

Posterior Intervals						
			Equal-Tail			
Component	Parameter	Alpha	Interval		HPD Interval	
1	Success Probability	0.050	0.0530	0.1187	0.0585	0.1212
2	Success Probability	0.050	0.2272	0.5722	0.2343	0.5780
1	Probability	0.050	0.5464	0.9454	0.6325	0.9642

Note that the standard errors in Figure 58.2 are not comparable to those in Figure 58.6, since the standard errors for the MLEs are expressed on the logit scale and the Bayes estimates are expressed on the data scale. You can add the METROPOLIS option in the BAYES statement to sample the quantities on the logit scale.

The "Posterior Intervals" table in Figure 58.6 displays 95% credible intervals (equal-tail intervals and intervals of highest posterior density). It can be concluded that the component with the higher success probability contributes less than 40% to the process.

Modeling Zero-Inflation: Is It Better to Fish Poorly or Not to Have Fished at All?

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpfmmgs2.sas`.)

The following example shows how you can use PROC HPFMM to model data with more zero values than expected.

Many count data show an excess of zeros relative to the frequency of zeros expected under a reference model. An excess of zeros leads to overdispersion since the process is more variable than a standard count data model. Different mechanisms can lead to excess zeros. For example, suppose that the data are generated from two processes with different distribution functions—one process generates the zero counts, and the other process generates nonzero counts. In the vernacular of Cameron and Trivedi (1998), such a model is called a *hurdle* model. With a certain probability—the probability of a nonzero count—a hurdle is crossed, and events are being generated. Hurdle models are useful, for example, to model the number of doctor visits per year. Once the decision to see a doctor has been made—the hurdle has been overcome—a certain number of visits follow.

Hurdle models are closely related to zero-inflated models. Both can be expressed as two-component mixtures in which one component has a degenerate distribution at zero and the other component is a count model. In a hurdle model, the count model follows a zero-truncated distribution. In a zero-inflated model, the count

model has a nonzero probability of generating zeros. Formally, a zero-inflated model can be written as

$$\Pr(Y = y) = \pi p_1 + (1 - \pi) p_2(y, \mu)$$
$$p_1 = \begin{cases} 1 & y = 0 \\ 0 & \text{otherwise} \end{cases}$$

where $p_2(y, \mu)$ is a standard count model with mean μ and support $y \in \{0, 1, 2, \ldots\}$.

The following data illustrates the use of a zero-inflated model. In a survey of park attendees, randomly selected individuals were asked about the number of fish they caught in the last six months. Along with that count, the gender and age of each sampled individual was recorded. The following DATA step displays the data for the analysis:

```
data catch;
   input gender $ age count @@;
   datalines;
  F   54  18   M   37   0   F   48  12   M   27   0
  M   55   0   M   32   0   F   49  12   F   45  11
  M   39   0   F   34   1   F   50   0   M   52   4
  M   33   0   M   32   0   F   23   1   F   17   0
  F   44   5   M   44   0   F   26   0   F   30   0
  F   38   0   F   38   0   F   52  18   M   23   1
  F   23   0   M   32   0   F   33   3   M   26   0
  F   46   8   M   45   5   M   51  10   F   48   5
  F   31   2   F   25   1   M   22   0   M   41   0
  M   19   0   M   23   0   M   31   1   M   17   0
  F   21   0   F   44   7   M   28   0   M   47   3
  M   23   0   F   29   3   F   24   0   M   34   1
  F   19   0   F   35   2   M   39   0   M   43   6
;
```

At first glance, the prevalence of zeros in the DATA set is apparent. Many park attendees did not catch any fish. These zero counts are made up of two populations: attendees who do not fish and attendees who fish poorly. A zero-inflation mechanism thus appears reasonable for this application since a zero count can be produced by two separate distributions.

The following statements fit a standard Poisson regression model to these data. A common intercept is assumed for men and women, and the regression slope varies with gender.

```
proc hpfmm data=catch;
   class gender;
   model count = gender*age / dist=Poisson;
run;
```

Figure 58.7 displays information about the model and data set. The "Model Information" table conveys that the model is a single-component Poisson model (a Poisson GLM) and that parameters are estimated by maximum likelihood. There are two levels in the CLASS variable gender, with females preceding males.

Figure 58.7 Model Information and Class Levels in Poisson Regression

The HPFMM Procedure

Model Information	
Data Set	WORK.CATCH
Response Variable	count
Type of Model	Generalized Linear (GLM)
Distribution	Poisson
Components	1
Link Function	Log
Estimation Method	Maximum Likelihood

Class Level Information		
Class	**Levels**	**Values**
gender	2	F M

Number of Observations Read	52
Number of Observations Used	52

The "Fit Statistics" and "Parameter Estimates" tables from the maximum likelihood estimation of the Poisson GLM are shown in Figure 58.8. If the model is not overdispersed, the Pearson statistic should roughly equal the number of observations in the data set minus the number of parameters. With n=52, there is evidence of overdispersion in these data.

Figure 58.8 Fit Results in Poisson Regression

Fit Statistics	
-2 Log Likelihood	182.7
AIC (Smaller is Better)	188.7
AICC (Smaller is Better)	189.2
BIC (Smaller is Better)	194.6
Pearson Statistic	85.9573

Parameter Estimates for Poisson Model					
Effect	**gender**	**Estimate**	**Standard Error**	**z Value**	**Pr > |z|**
Intercept		-3.9811	0.5439	-7.32	<.0001
age*gender	F	0.1278	0.01149	11.12	<.0001
age*gender	M	0.1044	0.01224	8.53	<.0001

Suppose that the cause of overdispersion is zero-inflation of the count data. The following statements fit a zero-inflated Poisson model.

```
proc hpfmm data=catch;
   class gender;
   model count = gender*age / dist=Poisson ;
   model       +            / dist=Constant;
run;
```

There are two MODEL statements, one for each component of the mixture. Because the distributions are different for the components, you cannot specify the mixture model with a single MODEL statement. The first MODEL statement identifies the response variable for the model (count) and defines a Poisson model with intercept and gender-specific slopes. The second MODEL statement uses the continuation operator ("+") and adds a model with a degenerate distribution by using DIST=CONSTANT. Because the mass of the constant is placed by default at zero, the second MODEL statement adds a zero-inflation component to the model. It is sufficient to specify the response variable in one of the MODEL statements; you use the "=" sign in that statement to separate the response variable from the model effects.

Figure 58.9 displays the "Model Information" and "Optimization Information" tables for this run of the HPFMM procedure. The model is now identified as a zero-inflated Poisson (ZIP) model with two components, and the parameters continue to be estimated by maximum likelihood. The "Optimization Information" table shows that there are four parameters in the optimization (compared to three parameters in the Poisson GLM model). The four parameters correspond to three parameters in the mean function (intercept and two gender-specific slopes) and the mixing probability.

Figure 58.9 Model and Optimization Information in the ZIP Model

The HPFMM Procedure

Model Information	
Data Set	WORK.CATCH
Response Variable	count
Type of Model	Zero-inflated Poisson
Components	2
Estimation Method	Maximum Likelihood

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	4
Mean Function Parameters	3
Scale Parameters	0
Mixing Prob Parameters	1

Results from fitting the ZIP model by maximum likelihood are shown in Figure 58.10. The –2 log likelihood and the information criteria suggest a much-improved fit over the single-component Poisson model (compare Figure 58.10 to Figure 58.8). The Pearson statistic is reduced by factor 2 compared to the Poisson model and suggests a better fit than the standard Poisson model.

Figure 58.10 Maximum Likelihood Results for the ZIP model

Fit Statistics	
-2 Log Likelihood	145.6
AIC (Smaller is Better)	153.6
AICC (Smaller is Better)	154.5
BIC (Smaller is Better)	161.4
Pearson Statistic	43.4467
Effective Parameters	4
Effective Components	2

Figure 58.10 *continued*

Parameter Estimates for Poisson Model						
Component	Effect	gender	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept		-3.5215	0.6448	-5.46	<.0001
1	age*gender	F	0.1216	0.01344	9.04	<.0001
1	age*gender	M	0.1056	0.01394	7.58	<.0001

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.6972	0.8342	0.4768	1.75	0.0802
2	0.3028	-0.8342			

The number of effective parameters and components shown in Figure 58.8 equals the values from Figure 58.9. This is not always the case because components can collapse (for example, when the mixing probability approaches zero or when two components have identical parameter estimates). In this example, both components and all four parameters are identifiable. The Poisson regression and the zero process mix, with a probability of approximately 0.6972 attributed to the Poisson component.

The HPFMM procedure enables you to fit some mixture models by Bayesian techniques. The following statements add the BAYES statement to the previous PROC HPFMM statements:

```
proc hpfmm data=catch seed=12345;
   class gender;
   model count = gender*age / dist=Poisson;
   model       +            / dist=constant;
   performance nthreads=2;
   bayes;
run;
```

The "Model Information" table indicates that the model parameters are estimated by Markov chain Monte Carlo techniques, and it displays the random number seed (Figure 58.11). This is useful if you did not specify a seed to identify the seed value that reproduces the current analysis. The "Bayes Information" table provides basic information about the Monte Carlo sampling scheme. The sampling method uses a data augmentation scheme to impute component membership and then the Gamerman (1997) algorithm to sample the component-specific parameters. The 2,000 burn-in samples are followed by 10,000 Monte Carlo samples without thinning.

Figure 58.11 Model, Bayes, and Prior Information in the ZIP Model

The HPFMM Procedure

Model Information	
Data Set	WORK.CATCH
Response Variable	count
Type of Model	Zero-inflated Poisson
Components	2
Estimation Method	Markov Chain Monte Carlo
Random Number Seed	12345

Figure 58.11 *continued*

Bayes Information	
Sampling Algorithm	Gamerman
Data Augmentation	Latent Variable
Initial Values of Chain	ML Estimates
Burn-In Size	2000
MC Sample Size	10000
MC Thinning	1
Parameters in Sampling	4
Mean Function Parameters	3
Scale Parameters	0
Mixing Prob Parameters	1

Prior Distributions

Component	Effect	gender	Distribution	Mean	Variance	Initial Value
1	**Intercept**		Normal(0, 1000)	0	1000.00	-3.5215
1	**age*gender**	F	Normal(0, 1000)	0	1000.00	0.1216
1	**age*gender**	M	Normal(0, 1000)	0	1000.00	0.1056
1	**Probability**		Dirichlet(1, 1)	0.5000	0.08333	0.6972

The "Prior Distributions" table identifies the prior distributions, their parameters for the sampled quantities, and their initial values. The prior distribution of parameters associated with model effects is a normal distribution with mean 0 and variance 1,000. The prior distribution for the mixing probability is a Dirichlet(1,1), which is identical to a uniform distribution (Figure 58.11). Since the second mixture component is a degeneracy at zero with no associated parameters, it does not appear in the "Prior Distributions" table in Figure 58.11.

Figure 58.12 displays descriptive statistics about the 10,000 posterior samples. Recall from Figure 58.10 that the maximum likelihood estimates were –3.5215, 0.1216, 0.1056, and 0.6972, respectively. With this choice of prior, the means of the posterior samples are generally close to the MLEs in this example. The "Posterior Intervals" table displays 95% intervals of equal-tail probability and 95% intervals of highest posterior density (HPD) intervals.

Figure 58.12 Posterior Summaries and Intervals in the ZIP Model

Posterior Summaries

Component	Effect	gender	N	Mean	Standard Deviation	Percentiles 25	50	75
1	**Intercept**		10000	-3.5456	0.6463	-3.9822	-3.5286	-3.1082
1	**age*gender**	F	10000	0.1219	0.0135	0.1129	0.1216	0.1310
1	**age*gender**	M	10000	0.1057	0.0140	0.0962	0.1053	0.1148
1	**Probability**		10000	0.6923	0.0950	0.6280	0.6960	0.7589

Figure 58.12 *continued*

Posterior Intervals							
Component	Effect	gender	Alpha	Equal-Tail Interval		HPD Interval	
1	Intercept		0.050	-4.8615	-2.3246	-4.8979	-2.3808
1	age*gender	F	0.050	0.0955	0.1490	0.0963	0.1495
1	age*gender	M	0.050	0.0784	0.1338	0.0786	0.1339
1	Probability		0.050	0.4999	0.8683	0.5139	0.8799

You can generate trace plots for the posterior parameter estimates by enabling ODS Graphics:

```
ods graphics on;
ods select TADPanel;
proc hpfmm data=catch seed=12345;
   class gender;
   model count = gender*age / dist=Poisson;
   model       +            / dist=constant;
   performance nthreads=2;
   bayes;
run;
ods graphics off;
```

A separate trace panel is produced for each sampled parameter, and the panels for the gender-specific slopes are shown in Figure 58.13. There is good mixing in the chains: the modest autocorrelation that diminishes after about 10 successive samples. By default, the HPFMM procedure transfers the credible intervals for each parameter from the “Posterior Intervals” table to the trace plot and the density plot in the trace panel.

Figure 58.13 Trace Panels for Gender-Specific Slopes

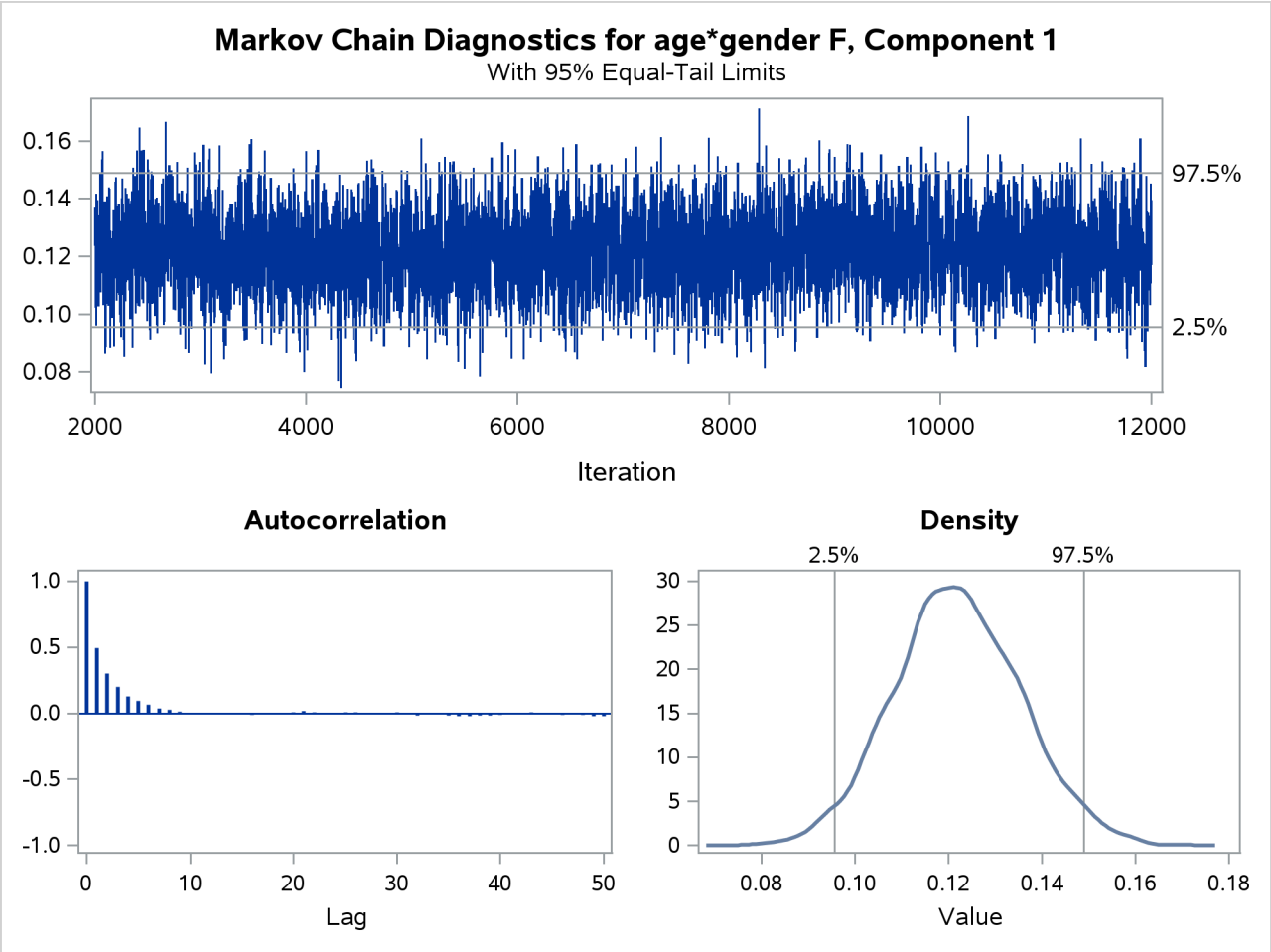

Figure 58.13 *continued*

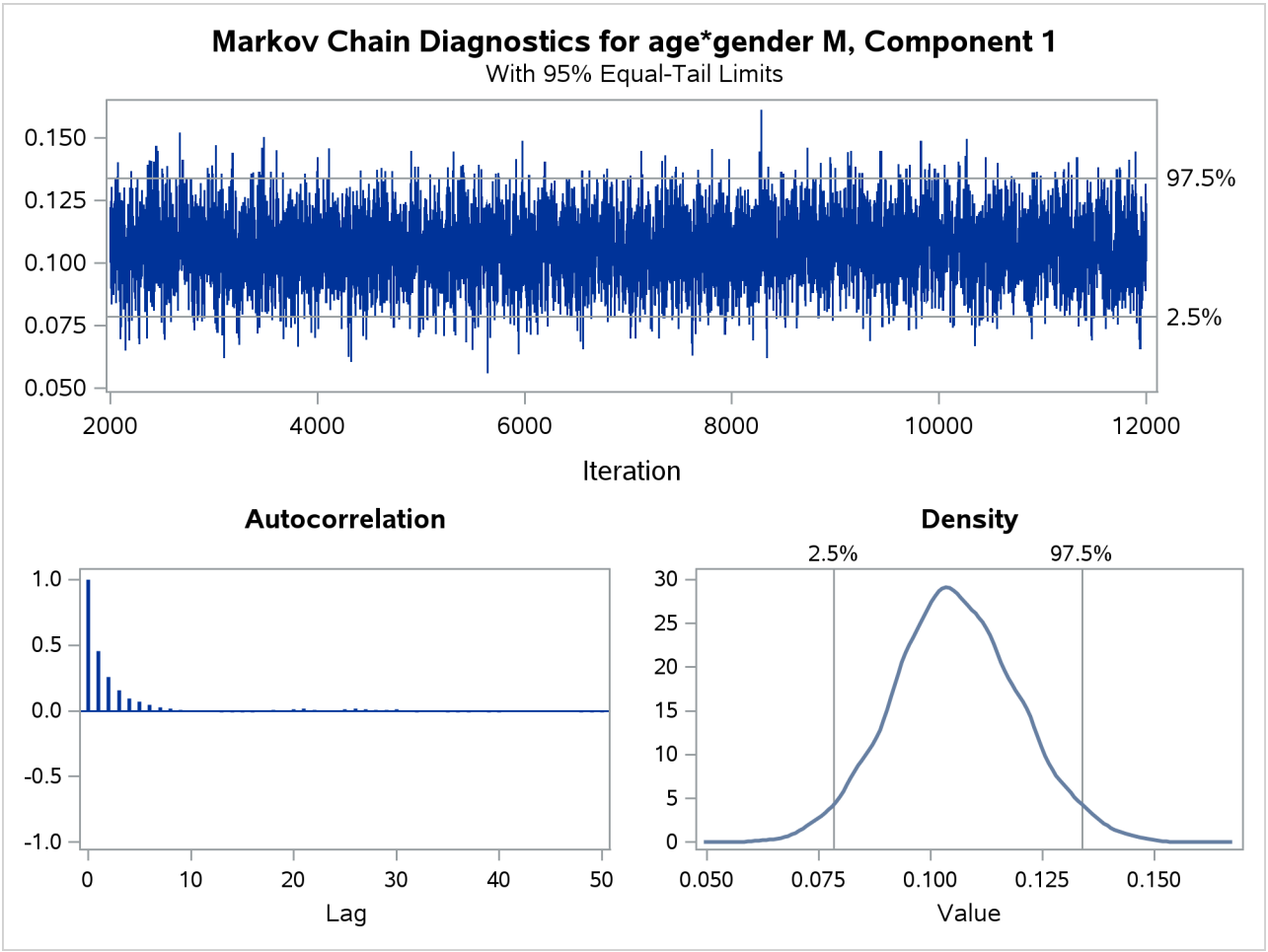

Looking for Multiple Modes: Are Galaxies Clustered?

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpfmmgs3.sas`.)

Mixture modeling is essentially a generalized form of one-dimensional cluster analysis. The following example shows how you can use PROC HPFMM to explore the number and nature of Gaussian clusters in univariate data.

Roeder (1990) presents data from the Corona Borealis sky survey with the velocities of 82 galaxies in a narrow slice of the sky. Cosmological theory suggests that the observed velocity of each galaxy is proportional to its distance from the observer. Thus, the presence of multiple modes in the density of these velocities could indicate a clustering of the galaxies at different distances.

The following DATA step recreates the data set in Roeder (1990). The computed variable v represents the measured velocity in thousands of kilometers per second.

```
title "HPFMM Analysis of Galaxies Data";
data galaxies;
   input velocity @@;
   v = velocity / 1000;
   datalines;
9172  9350  9483  9558  9775  10227 10406 16084 16170 18419
18552 18600 18927 19052 19070 19330 19343 19349 19440 19473
19529 19541 19547 19663 19846 19856 19863 19914 19918 19973
19989 20166 20175 20179 20196 20215 20221 20415 20629 20795
20821 20846 20875 20986 21137 21492 21701 21814 21921 21960
22185 22209 22242 22249 22314 22374 22495 22746 22747 22888
22914 23206 23241 23263 23484 23538 23542 23666 23706 23711
24129 24285 24289 24366 24717 24990 25633 26960 26995 32065
32789 34279
;
```

Analysis of potentially multimodal data is a natural application of finite mixture models. In this case, the modeling is complicated by the question of the variance for each of the components. Using identical variances for each component could obscure underlying structure, but the additional flexibility granted by component-specific variances might introduce spurious features.

You can use PROC HPFMM to prepare analyses for equal and unequal variances and use one of the available fit statistics to compare the resulting models. You can use the model selection facility to explore models with varying numbers of mixture components—say, from three to seven as investigated in Roeder (1990). The following statements select the best unequal-variance model using Akaike's information criterion (AIC), which has a built-in penalty for model complexity:

```
title2 "Three to Seven Components, Unequal Variances";
ods graphics on;
proc hpfmm data=galaxies criterion=AIC;
   model v = / kmin=3 kmax=7;
   ods exclude IterHistory OptInfo ComponentInfo;
run;
```

The KMIN= and KMAX= options indicate the smallest and largest number of components to consider. The ODS GRAPHICS and ODS SELECT statements request a density plot. The output for unequal variances is shown in Figure 58.14 and Figure 58.15.

Figure 58.14 Model Selection for Galaxy Data Assuming Unequal Variances

HPFMM Analysis of Galaxies Data
Three to Seven Components, Unequal Variances

The HPFMM Procedure

Model Information	
Data Set	WORK.GALAXIES
Response Variable	v
Type of Model	Homogeneous Mixture
Distribution	Normal
Min Components	3
Max Components	7
Link Function	Identity
Estimation Method	Maximum Likelihood

Component Evaluation for Mixture Models

Model ID	Number of Components Total	Number of Components Eff.	Number of Parameters Total	Number of Parameters Eff.	-2 Log L	AIC	AICC	BIC	Pearson	Max Gradient
1	3	3	8	8	406.96	422.96	424.94	442.22	82.00	0.000024
2	4	4	11	11	406.96	428.96	432.74	455.44	82.00	0.00011
3	5	5	14	14	406.96	434.96	441.23	468.66	82.00	0.000054
4	6	6	17	17	406.96	440.96	450.53	481.88	82.00	0.000011
5	7	7	20	20	406.96	446.96	460.73	495.10	82.00	0.00011

The model with 3 components (ID=1) was selected as 'best' based on the AIC statistic.

Fit Statistics	
-2 Log Likelihood	407.0
AIC (Smaller is Better)	423.0
AICC (Smaller is Better)	424.9
BIC (Smaller is Better)	442.2
Pearson Statistic	82.0001
Effective Parameters	8
Effective Components	3

Parameter Estimates for Normal Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	9.7101	0.1597	60.80	<.0001
2	Intercept	33.0444	0.5322	62.09	<.0001
3	Intercept	21.4039	0.2597	82.41	<.0001
1	Variance	0.1785	0.09542		
2	Variance	0.8496	0.6937		
3	Variance	4.8567	0.8098		

Figure 58.14 *continued*

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.0854	-2.3308	0.3959	-5.89	<.0001
2	0.0366	-3.1781	0.5893	-5.39	<.0001
3	0.8781	0			

Figure 58.15 Density Plot for Best (Three-Component) Model Assuming Unequal Variances

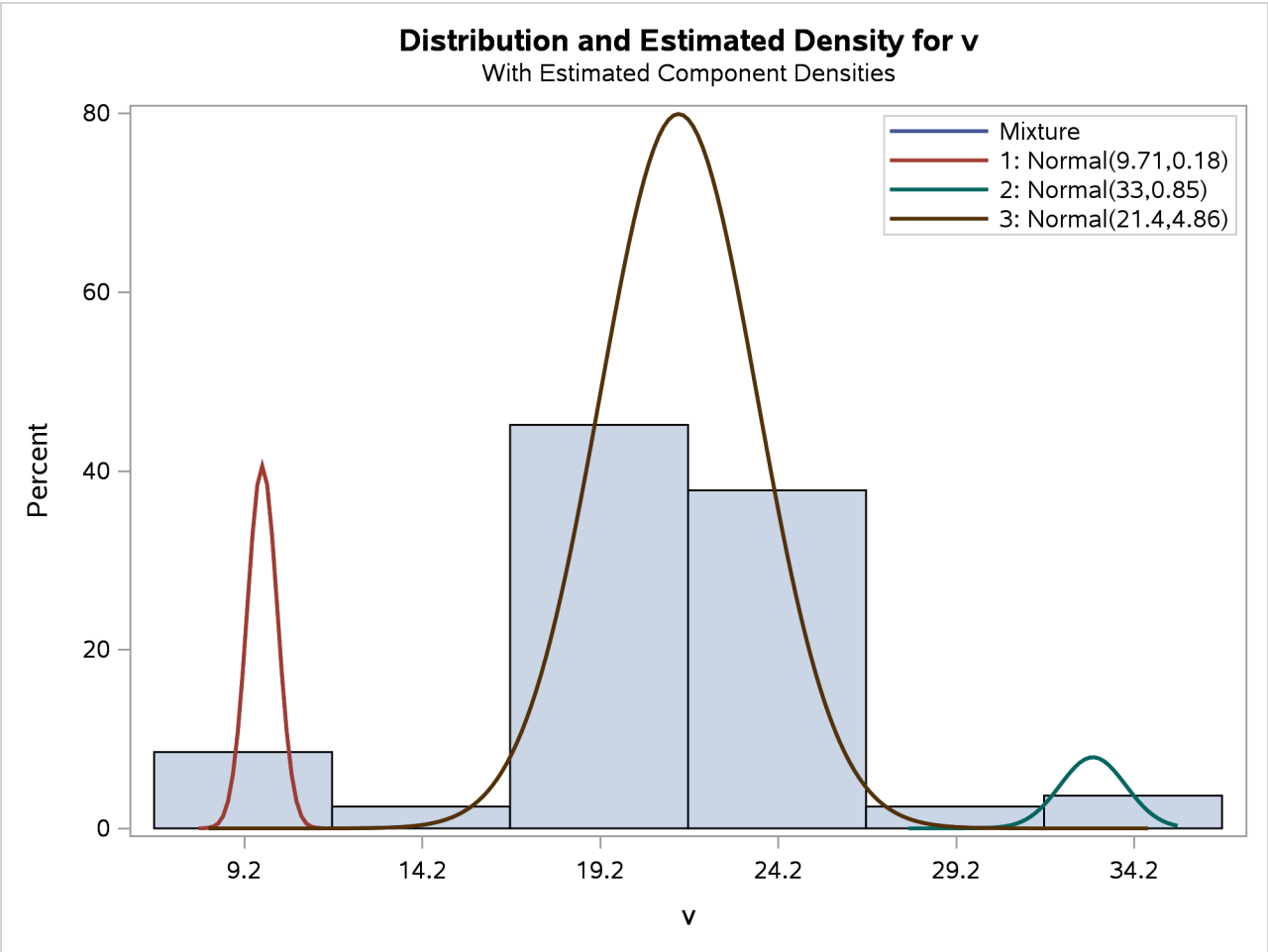

Figure 58.16 Criterion Panel Plot for Model Selection Assuming Unequal Variances

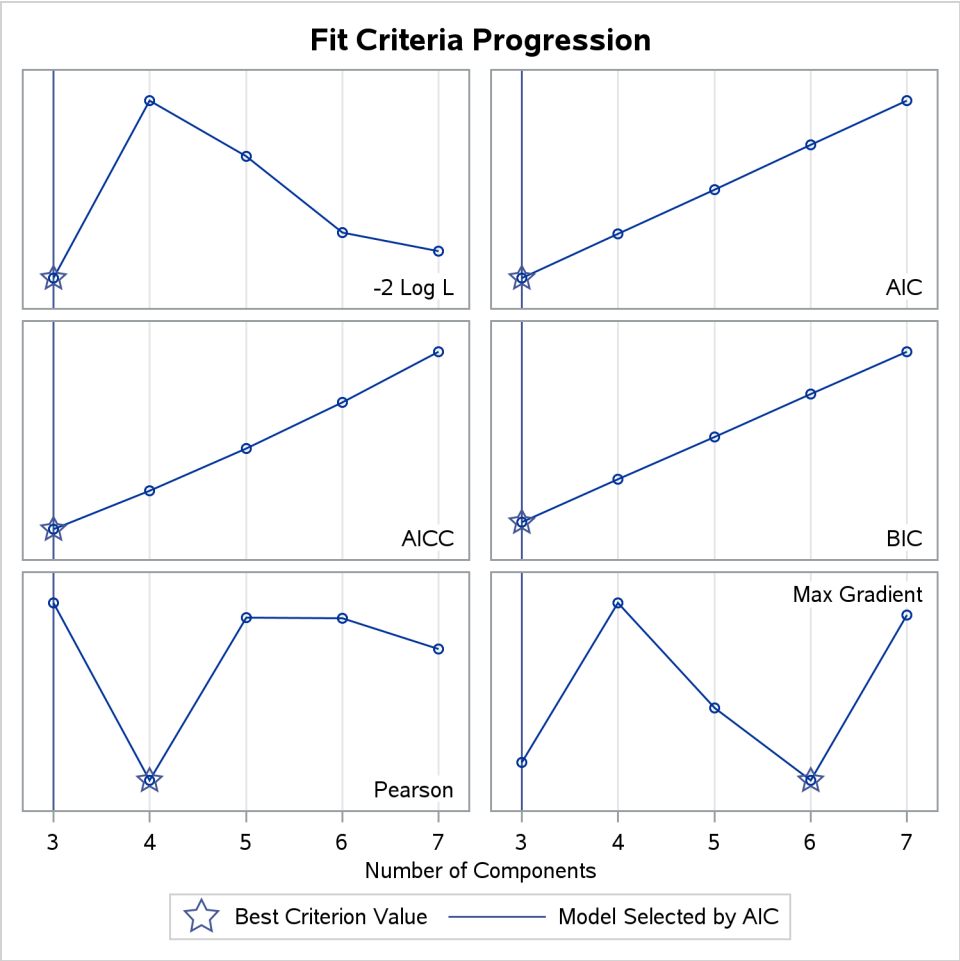

This example uses the AIC for model selection. Figure 58.16 shows the AIC and other model fit criteria for each of the fitted models.

To require that the separate components have identical variances, add the EQUATE=SCALE option in the MODEL statement:

```
title2 "Three to Seven Components, Equal Variances";
proc hpfmm data=galaxies criterion=AIC gconv=0;
   model v = / kmin=3 kmax=7  equate=scale;
run;
```

The GCONV= convergence criterion is turned off in this PROC HPFMM run to avoid the early stoppage of the iterations when the relative gradient changes little between iterations. Turning the criterion off usually ensures that convergence is achieved with a small absolute gradient of the objective function.

The output for equal variances is shown in Figure 58.17 and Figure 58.18.

Figure 58.17 Model Selection for Galaxy Data Assuming Equal Variances

HPFMM Analysis of Galaxies Data
Three to Seven Components, Equal Variances

The HPFMM Procedure

Model Information	
Data Set	WORK.GALAXIES
Response Variable	v
Type of Model	Homogeneous Mixture
Distribution	Normal
Min Components	3
Max Components	7
Link Function	Identity
Estimation Method	Maximum Likelihood

Component Evaluation for Mixture Models

Model ID	Number of Components Total	Number of Components Eff.	Number of Parameters Total	Number of Parameters Eff.	-2 Log L	AIC	AICC	BIC	Pearson	Max Gradient
1	**3**	**3**	**6**	**6**	478.74	490.74	491.86	505.18	82.00	1.197E-6
2	**4**	**4**	**8**	**8**	416.49	432.49	434.47	451.75	82.00	2.958E-7
3	**5**	**5**	**10**	**10**	416.49	436.49	439.59	460.56	82.00	4.313E-6
4	**6**	**6**	**12**	**12**	416.49	440.49	445.02	469.37	82.00	6.294E-6
5	**7**	**7**	**14**	**14**	416.49	444.49	450.76	478.19	82.00	4.882E-6

The model with 4 components (ID=2) was selected as 'best' based on the AIC statistic.

Fit Statistics	
-2 Log Likelihood	416.5
AIC (Smaller is Better)	432.5
AICC (Smaller is Better)	434.5
BIC (Smaller is Better)	451.7
Pearson Statistic	82.0000
Effective Parameters	8
Effective Components	4

Parameter Estimates for Normal Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	**Intercept**	23.5058	0.3460	67.93	<.0001
2	**Intercept**	33.0440	0.7610	43.42	<.0001
3	**Intercept**	20.0086	0.3029	66.06	<.0001
4	**Intercept**	9.7103	0.4981	19.50	<.0001
1	**Variance**	1.7354	0.3905		
2	**Variance**	1.7354	0.3905		
3	**Variance**	1.7354	0.3905		
4	**Variance**	1.7354	0.3905		

Figure 58.17 *continued*

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.3503	1.4118	0.4497	3.14	0.0017
2	0.0366	-0.8473	0.6901	-1.23	0.2195
3	0.5277	1.8216	0.4205	4.33	<.0001
4	0.0854	0			

Figure 58.18 Density Plot for Best (Six-Component) Model Assuming Equal Variances

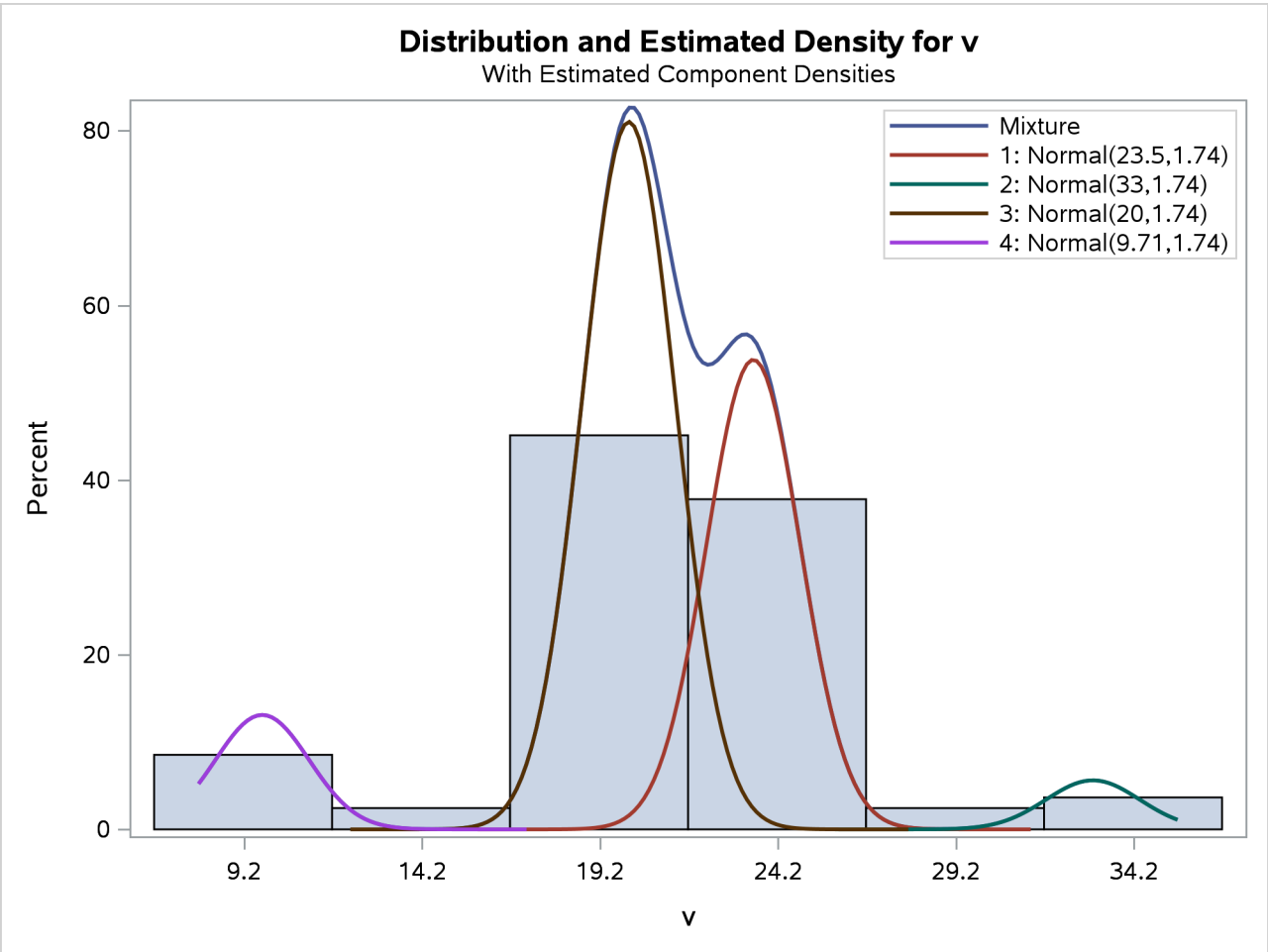

Not surprisingly, the two variance specifications produce different optimal models. The unequal variance specification favors a three-component model while the equal variance specification favors a four-component model. Comparison of the AIC fit statistics, 423.0 and 432.5, indicates that the three-component, unequal variance model provides the best overall fit.

Comparison with Roeder's Method

It is important to note that Roeder's original analysis proceeds in a different manner than the finite mixture modeling presented here. The technique presented by Roeder first develops a "best" range of scale parameters based on a specific criterion. Roeder then uses fixed scale parameters taken from this range to develop optimal equal-scale Gaussian mixture models.

You can reproduce Roeder's point estimate for the density by specifying a five-component Gaussian mixture. In addition, use the EQUATE=SCALE option in the MODEL statement and a RESTRICT statement fixing the first component's scale parameter at 0.9025 (Roeder's $h = 0.95$, scale$= h^2$). The combination of these options produces a mixture of five Gaussian components, each with variance 0.9025. The following statements conduct this analysis:

```
title2 "Five Components, Equal Variances = 0.9025";
proc hpfmm data=galaxies;
   model v = / K=5 equate=scale;
   restrict int 0 (scale 1) = 0.9025;
run;
ods graphics off;
```

The output is shown in Figure 58.19 and Figure 58.20.

Figure 58.19 Reproduction of Roeder's Five-Component Analysis of Galaxy Data

HPFMM Analysis of Galaxies Data
Five Components, Equal Variances = 0.9025

The HPFMM Procedure

Model Information	
Data Set	WORK.GALAXIES
Response Variable	v
Type of Model	Homogeneous Mixture
Distribution	Normal
Components	5
Link Function	Identity
Estimation Method	Maximum Likelihood

Fit Statistics	
-2 Log Likelihood	412.2
AIC (Smaller is Better)	430.2
AICC (Smaller is Better)	432.7
BIC (Smaller is Better)	451.9
Pearson Statistic	82.5549
Effective Parameters	9
Effective Components	5

Linear Constraints at Solution	
k = 1	**Constraint Active**
Variance = 0.90	Yes

Figure 58.19 *continued*

Parameter Estimates for Normal Model					
Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	26.3266	0.7778	33.85	<.0001
2	Intercept	33.0443	0.5485	60.25	<.0001
3	Intercept	9.7101	0.3591	27.04	<.0001
4	Intercept	23.0295	0.2294	100.38	<.0001
5	Intercept	19.7187	0.1784	110.55	<.0001
1	Variance	0.9025	0		
2	Variance	0.9025	0		
3	Variance	0.9025	0		
4	Variance	0.9025	0		
5	Variance	0.9025	0		

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.0397	-2.4739	0.7084	-3.49	0.0005
2	0.0366	-2.5544	0.6016	-4.25	<.0001
3	0.0854	-1.7071	0.4141	-4.12	<.0001
4	0.3678	-0.2466	0.2699	-0.91	0.3609
5	0.4706	0			

Figure 58.20 Density Plot for Roeder's Analysis

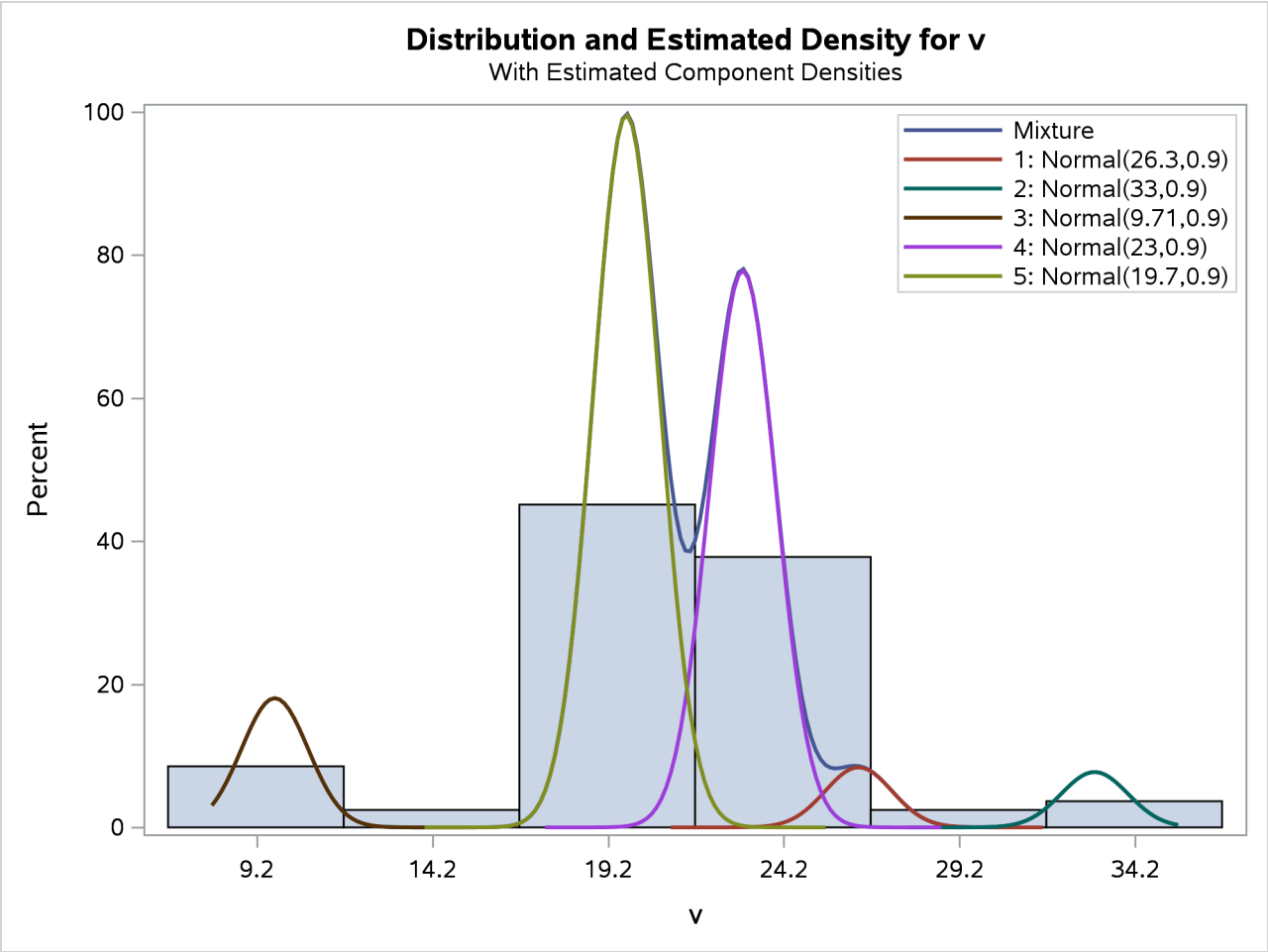

Syntax: HPFMM Procedure

The following statements are available in the HPFMM procedure:

```
PROC HPFMM < options > ;
    BAYES bayes-options ;
    BY variables ;
    CLASS variables ;
    FREQ variable ;
    ID variables ;
    MODEL response< (response-options) > = < effects > < / model-options > ;
    MODEL events/trials = < effects > < / model-options > ;
    MODEL               + < effects > < / model-options > ;
    OUTPUT < OUT=SAS-data-set >
            < keyword< (keyword-options) > < =name > > . . .
            < keyword< (keyword-options) > < =name > > < / options > ;
    PERFORMANCE performance-options ;
    PROBMODEL < effects > < / probmodel-options > ;
    RESTRICT < 'label' > constraint-specification < , . . . , constraint-specification >
              < operator < value > > < / option > ;
    WEIGHT variable ;
```

The PROC HPFMM statement and at least one MODEL statement is required. The CLASS, RESTRICT and MODEL statements can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statements. The RESTRICT statements must appear after the MODEL statements.

PROC HPFMM Statement

```
PROC HPFMM < options > ;
```

The PROC HPFMM statement invokes the HPFMM procedure. Table 58.2 summarizes the *options* available in the PROC HPFMM statement. These and other *options* in the PROC HPFMM statement are then described fully in alphabetical order.

Table 58.2 PROC HPFMM Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
EXCLUSION=	Specifies how the procedure responds to support violations in the data
NAMELEN=	Specifies the length of effect names
SEED=	Specifies the random number seed for analyses that require random number draws
Displayed Output	
COMPONENTINFO	Displays information about the mixture components

Table 58.2 *continued*

Option	Description
CORR	Displays the asymptotic correlation matrix of the maximum likelihood parameter estimates or the empirical correlation matrix of the Bayesian posterior estimates
COV	Displays the asymptotic covariance matrix of the maximum likelihood parameter estimates or the empirical covariance matrix of the Bayesian posterior estimates
COVI	Displays the inverse of the covariance matrix of the parameter estimates
FITDETAILS	Displays fit information for all examined models
ITDETAILS	Adds estimates and gradients to the "Iteration History" table
NOCLPRINT	Suppresses the "Class Level Information" table completely or partially
NOITPRINT	Suppresses the "Iteration History Information" table
NOPRINT	Suppresses tabular and graphical output
PARMSTYLE=	Specifies how parameters are displayed in ODS tables
PLOTS	Produces ODS statistical graphics
Computational Options	
CRITERION=	Specifies the criterion used in model selection
NOCENTER	Prevents centering and scaling of the regressor variables
PARTIAL=	Specifies a variable that defines a partial classification
Options Related to Optimization	
ABSCONV=	Tunes an absolute function convergence criterion
ABSFCONV=	Tunes an absolute function difference convergence criterion
ABSGCONV=	Tunes the absolute gradient convergence criterion
FCONV=	Specifies a relative function convergence criterion that is based on a relative change of the function value
FCONV2=	Specifies a relative function convergence criterion that is based on a predicted reduction of the objective function
GCONV=	Tunes the relative gradient convergence criterion
MAXITER=	Specifies the maximum number of iterations in any optimization
MAXFUNC=	Specifies the maximum number of function evaluations in any optimization
MAXTIME=	Specifies the upper limit of CPU time in seconds for any optimization
MINITER=	Specifies the minimum number of iterations in any optimization
TECHNIQUE=	Selects the optimization technique
Singularity Tolerances	
INVALIDLOGL=	Tunes the value assigned to an invalid component log likelihood
SINGCHOL=	Tunes singularity for Cholesky decompositions
SINGRES=	Tunes singularity for the residual variance
SINGULAR=	Tunes general singularity criterion

You can specify the following *options* in the PROC HPFMM statement.

ABSCONV=*r*

ABSTOL=*r*

specifies an absolute function convergence criterion. For minimization, the termination criterion is $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of *r* is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

ABSFCONV=*r*

ABSFTOL=*r*

specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, the termination criterion is a small change of the function value in successive iterations:

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}(k)$ is defined as the vertex with the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default value is *r*=0.

ABSGCONV=*r*

ABSGTOL=*r*

specifies an absolute gradient convergence criterion. The termination criterion is a small maximum absolute gradient element:

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $g_j(\cdot)$ is the gradient of the objective function with respect to the *j*th parameter. This criterion is not used by the NMSIMP technique. The default value is *r*=1E–5.

COMPONENTINFO

COMPINFO

CINFO

produces a table with additional details about the fitted model components.

COV

produces the covariance matrix of the parameter estimates. For maximum likelihood estimation, this matrix is based on the inverse (projected) Hessian matrix. For Bayesian estimation, it is the empirical covariance matrix of the posterior estimates. The covariance matrix is shown for all parameters, even if they did not participate in the optimization or sampling.

COVI

produces the inverse of the covariance matrix of the parameter estimates. For maximum likelihood estimation, the covariance matrix is based on the inverse (projected) Hessian matrix. For Bayesian estimation, it is the empirical covariance matrix of the posterior estimates. This matrix is then inverted by sweeping, and rows and columns that correspond to linear dependencies or singularities are zeroed.

CORR

produces the correlation matrix of the parameter estimates. For maximum likelihood estimation this matrix is based on the inverse (projected) Hessian matrix. For Bayesian estimation, it is based on the empirical covariance matrix of the posterior estimates.

CRITERION=*keyword*

CRIT=*keyword*

specifies the criterion by which the HPFMM procedure ranks models when multiple models are evaluated during maximum likelihood estimation. You can choose from the following *keywords* to rank models:

AIC	based on Akaike's information criterion
AICC	based on the bias-corrected AIC criterion
BIC	based on the Bayesian information criterion
GRADIENT	based on the largest element of the gradient (in absolute value)
LOGL | LL	based on the mixture log likelihood
PEARSON	based on the Pearson statistic

The default is CRITERION=BIC.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC HPFMM. The default is the most recently created data set.

EXCLUSION=NONE | ANY | ALL

EXCLUDE=NONE | ANY | ALL

specifies how the HPFMM procedure handles support violations of observations. For example, in a mixture of two Poisson variables, negative response values are not possible. However, in a mixture of a Poisson and a normal variable, negative values are possible, and their likelihood contribution to the Poisson component is zero. An observation that violates the support of one component distribution of the model might be a valid response with respect to one or more other component distributions. This requires some nuanced handling of support violations in mixture models.

The default exclusion technique, EXCLUSION=ALL, removes an observation from the analysis only if it violates the support of all component distributions. The other extreme, EXCLUSION=NONE, permits an observation into the analysis regardless of support violations. EXCLUSION=ANY removes observations from the analysis if the response violates the support of any component distributions. In the single-component case, EXCLUSION=ALL and EXCLUSION=ANY are identical.

FCONV=*r*

FTOL=*r*

specifies a relative function convergence criterion that is based on the relative change of the function value. For all techniques except NMSIMP, PROC HPFMM terminates when there is a small relative change of the function value in successive iterations:

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex with

the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default is $r = 10^{-\mathrm{FDIGITS}}$, where FDIGITS is by default $-\log_{10}\{\epsilon\}$, and ϵ is the machine precision.

FCONV2=*r*

FTOL2=*r*

specifies a relative function convergence criterion that is based on the predicted reduction of the objective function. For all techniques except NMSIMP, the termination criterion is a small predicted reduction

$$df^{(k)} \approx f(\boldsymbol{\theta}^{(k)}) - f(\boldsymbol{\theta}^{(k)} + \mathbf{s}^{(k)})$$

of the objective function. The predicted reduction

$$\begin{aligned} df^{(k)} &= -\mathbf{g}^{(k)\prime}\mathbf{s}^{(k)} - \frac{1}{2}\mathbf{s}^{(k)\prime}\mathbf{H}^{(k)}\mathbf{s}^{(k)} \\ &= -\frac{1}{2}\mathbf{s}^{(k)\prime}\mathbf{g}^{(k)} \\ &\le r \end{aligned}$$

is computed by approximating the objective function f by the first two terms of the Taylor series and substituting the Newton step:

$$\mathbf{s}^{(k)} = -[\mathbf{H}^{(k)}]^{-1}\mathbf{g}^{(k)}$$

For the NMSIMP technique, the termination criterion is a small standard deviation of the function values of the $n+1$ simplex vertices $\boldsymbol{\theta}_l^{(k)}$, $l = 0, \ldots, n$,

$$\sqrt{\frac{1}{n+1}\sum_l \left[f(\boldsymbol{\theta}_l^{(k)}) - \overline{f}(\boldsymbol{\theta}^{(k)})\right]^2} \le r$$

where $\overline{f}(\boldsymbol{\theta}^{(k)}) = \frac{1}{n+1}\sum_l f(\boldsymbol{\theta}_l^{(k)})$. If there are n_{act} boundary constraints active at $\boldsymbol{\theta}^{(k)}$, the mean and standard deviation are computed only for the $n+1-n_{act}$ unconstrained vertices.

The default value is r = 1E–6 for the NMSIMP technique and r = 0 otherwise.

FITDETAILS

requests that the "Optimization Information," "Iteration History," and "Fit Statistics" tables be produced for all optimizations when models with different number of components are evaluated. For example, the following statements fit a binomial regression model with up to three components and produces fit and optimization information for all three:

```
proc hpfmm fitdetails;
   model y/n  = x / kmax=3;
run;
```

Without the FITDETAILS option, only the "Fit Statistics" table for the selected model is displayed.

In Bayesian estimation, the FITDETAILS option displays the following tables for each model that the procedure fits: "Bayes Information," "Iteration History," "Prior Information," "Fit Statistics," "Posterior Summaries," "Posterior Intervals," and any requested diagnostics tables. The "Iteration History" table appears only if the BAYES statement includes the INITIAL=MLE option.

Without the FITDETAILS option, these tables are listed only for the selected model.

GCONV=*r*

GTOL=*r*

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, the termination criterion is a small normalized predicted function reduction:

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the CONGRA technique (where a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \, |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The default value is r=1E–8.

HESSIAN

displays the Hessian matrix of the model. This option is not available for Bayesian estimation.

INVALIDLOGL=*r*

specifies the value assumed by the HPFMM procedure if a log likelihood cannot be computed (for example, because the value of the response variable falls outside of the response distribution's support). The default value is –1E20.

ITDETAILS

adds parameter estimates and gradients to the "Iteration History" table. If the HPFMM procedure centers or scales the model variables (or both), the parameter estimates and gradients reported during the iteration refer to that scale. You can suppress centering and scaling with the NOCENTER option.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, and NEWRAP: 125
- QUANEW and DBLDOG: 500
- CONGRA: 1000
- NMSIMP: 3000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that is specified by the MAXFUNC= option. You can choose the optimization technique with the TECHNIQUE= option.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, and NEWRAP: 50
- QUANEW and DBLDOG: 200
- CONGRA: 400
- NMSIMP: 1000

These default values also apply when *n* is specified as a missing value. You can choose the optimization technique with the TECHNIQUE= option.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The time is checked only at the end of each iteration. Therefore, the actual run time might be longer than the specified time. By default, CPU time is not limited.

MINITER=*n*

MINIT=*n*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

NAMELEN=*number*

specifies the length to which long effect names are shortened. The default and minimum value is 20.

NOCENTER

requests that regressor variables not be centered or scaled. By default the HPFMM procedure centers and scales columns of the **X** matrix if the models contain intercepts. If NOINT options in MODEL statements are in effect, the columns of **X** are scaled but not centered. Centering and scaling can help with the stability of estimation and sampling algorithms. The HPFMM procedure does not produce a table of the centered and scaled coefficients and provides no user control over the type of centering and scaling that is applied. The NOCENTER option turns any centering and scaling off and processes the raw values of the continuous variables.

NOCLPRINT<=*number***>**

suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying a *number* helps to reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOITPRINT

suppresses the display of the "Iteration History Information" table.

NOPRINT

suppresses the normal display of tabular and graphical results. The NOPRINT option is useful when you want to create only one or more output data sets with the procedure. This option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

PARMSTYLE=EFFECT | LABEL

specifies the display style for parameters and effects. The HPFMM procedure can display parameters in two styles:

- The EFFECT style (which is used by the MIXED and GLIMMIX procedure, for example) identifies a parameter with an "Effect" column and adds separate columns for the CLASS variables in the model.
- The LABEL style creates one column, named Parameter, that combines the relevant information about a parameter into a single column. If your model contains multiple CLASS variables, the LABEL style might use space more economically.

The EFFECT style is the default for models that contain effects; otherwise the LABEL style is used (for example, in homogeneous mixtures). You can change the display style with the PARMSTYLE= option. Regardless of the display style, ODS output data sets that contain information about parameter estimates contain columns for both styles.

PARTIAL=*variable*

MEMBERSHIP=*variable*

specifies a variable in the input data set that identifies component membership. You can specify missing values for observations whose component membership is undetermined; this is known as a partial classification (McLachlan and Peel 2000, p. 75). For observations with known membership, the likelihood contribution is no longer a mixture. If observation i is known to be a member of component m, then its log likelihood contribution is

$$\log\left\{\pi_m(\mathbf{z},\boldsymbol{\alpha}_m)\ \ p_m(y;\mathbf{x}_m'\boldsymbol{\beta}_m,\phi_m)\right\}$$

Otherwise, if membership is undetermined, it is

$$\log\left\{\sum_{j=1}^{k}\ \pi_j(\mathbf{z},\boldsymbol{\alpha}_j)\,p_j(y;\mathbf{x}_j'\boldsymbol{\beta}_j,\phi_j)\right\}$$

The variable specified in the PARTIAL= option can be numeric or character. In case of a character variable, the variable must appear in the CLASS statement. If the PARTIAL= variable appears in the CLASS statement, the membership assignment is made based on the levelized values of the variable, as shown in the "Class Level Information" table. Invalid values of the PARTIAL= variable are ignored.

In a model in which label switching is a problem, the switching can sometimes be avoided by assigning just a few observations to categories. For example, in a three-component model, switches might be prevented by assigning the observation with the smallest response value to the first component and the observation with the largest response value to the last component.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the plots produced through ODS Graphics.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc hpfmm data=yeast seed=12345;
   model count/n = / k=2;
   freq f;
   performance nthreads=2;
   bayes;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

Global Plot Options

The *global-plot-options* apply to all relevant plots generated by the HPFMM procedure. The *global-plot-options* supported by the HPFMM procedure are as follows:

UNPACKPANEL
UNPACK
displays each graph separately. (By default, some graphs can appear together in a single panel.)

ONLY
produces only the specified plots. This option is useful if you do not want the procedure to generate all default graphics, but only the ones specified.

Specific Plot Options

The following listing describes the specific plots and their *options*.

ALL
requests that all plots appropriate for the analysis be produced.

NONE
requests that no ODS graphics be produced.

DENSITY < (*density-options*) >
requests a plot of the data histogram and mixture density function. This graphic is a default graphic in models without effects in the MODEL statements and is available only in these models. Furthermore, all distributions involved in the mixture must be continuous. You can specify the following *density-options* to modify the plot:

CUMULATIVE

CDF

displays the histogram and densities in cumulative form.

NBINS=*n*

BINS=*n*

specifies the number of bins in the histogram; *n* is greater than or equal to 0. By default, the HPFMM procedure computes a suitable bin width and number of bins, based on the range of the response and the number of usable observations. The option has no effect for binary data.

NOCOMPONENTS

NOCOMP

suppresses the component densities from the plot. If the component densities are displayed, they are scaled so that their sum equals the mixture density at any point on the graph. In single-component models, this option has no effect.

NODENSITY

NODENS

suppresses the computation of the mixture density (and the component densities if the COMPONENTS suboption is specified). If you specify the NOHISTOGRAM and the NODENSITY option, no graphic is produced.

NOLABEL

suppresses the component identification with labels. By default, the HPFMM procedure labels component densities in the legend of the plot. If you do not specify a model label with the LABEL= option in the MODEL statement, an identifying label is constructed from the parameter estimates that are associated with the component. In this case the parameter values are not necessarily the mean and variance of the distribution; the values used to identify the densities on the plot are chosen to simplify linking between graphical and tabular results.

NOHISTOGRAM

NOHIST

suppresses the computation of the histogram of the raw values. If you specify the NOHISTOGRAM and the NODENSITY option, no graphic is produced.

NPOINTS=*n*

N=*n*

specifies the number of values used to compute the density functions; *n* is greater than or equal to 0. The default is N=200.

WIDTH=*value*

BINWIDTH=*value*

specifies the bin width for the histogram. The *value* is specified in units of the response variable and must be positive. The option has no effect for binary data.

TRACE < (*tadpanel-options*) >
requests a trace panel with posterior diagnostics for a Bayesian analysis. If a BAYES statement is present, the trace panel plots are generated by default, one for each sampled parameter. You can specify the following *tadpanel-options* to modify the graphic:

BOX
BOXPLOT
replaces the autocorrelation plot with a box plot of the posterior sample.

SMOOTH=NONE | MEAN | SPLINE
adds a reference estimate to the trace plot. By default, SMOOTH=NONE. SMOOTH=MEAN uses the arithmetic mean of the trace as the reference. SMOOTH=SPLINE adds a penalized B-spline.

REFERENCE= *reference-style*
adds vertical reference lines to the density plot, trace plot, and box plot. The available options for the *reference-style* are:

NONE	suppresses the reference lines
EQT	requests equal-tail intervals
HPD	requests intervals of highest posterior density. The level for the credible or HPD intervals is chosen based on the "Posterior Interval Statistics" table.
PERCENTILES	(or PERC) for percentiles. Up to three percentiles can be displayed, as based on the "Posterior Summary Statistics" table.

The default is REFERENCE=EQT.

UNPACK
unpacks the panel graphic and displays its elements as separate plots.

CRITERIONPANEL < (*critpanel-options*) >
requests a plot for comparing the model fit criteria for different numbers of components. This plot is available only if you also specify the KMAX option in at least one MODEL statement. The plot includes different criteria, depending on whether you are using maximum likelihood or Bayesian estimation. You can specify the following *critpanel-option* to modify the plot:

UNPACK
unpacks the panel plot and displays its elements as separate plots, one for each fit criterion.

SEED=*n*
determines the random number seed for analyses that depend on a random number stream. If you do not specify a seed or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. The largest possible value for the seed is $2^{31} - 1$. The seed value is reported in the "Model Information" table.

You can use the SYSRANDOM and SYSRANEND macro variables after a PROC HPFMM run to query the initial and final seed values. However, using the final seed value as the starting seed for a subsequent analysis does not continue the random number stream where the previous analysis left off.

The SYSRANEND macro variable provides a mechanism to pass on seed values to ensure that the sequence of random numbers is the same every time you run an entire program.

Analyses that use the same (nonzero) seed are not completely reproducible if they are executed with a different number of threads since the random number streams in separate threads are independent. You can control the number of threads used by the HPFMM procedure with system options or through the PERFORMANCE statement in the HPFMM procedure.

SINGCHOL=*number*
tunes the singularity criterion in Cholesky decompositions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGRES=*number*
sets the tolerance for which the residual variance or scale parameter is considered to be zero. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGULAR=*number*
tunes the general singularity criterion applied by the HPFMM procedure in sweeps and inversions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

TECHNIQUE=*keyword*
TECH=*keyword*
specifies the optimization technique to obtain maximum likelihood estimates. You can choose from the following techniques by specifying the appropriate *keyword*:

CONGRA	performs a conjugate-gradient optimization.
DBLDOG	performs a version of double-dogleg optimization.
NEWRAP	performs a Newton-Raphson optimization combining a line-search algorithm with ridging.
NMSIMP	performs a Nelder-Mead simplex optimization.
NONE	performs no optimization.
NRRIDG	performs a Newton-Raphson optimization with ridging.
QUANEW	performs a dual quasi-Newton optimization.
TRUREG	performs a trust-region optimization.

The default is TECH=QUANEW.

For more details about these optimization methods, see the section "Choosing an Optimization Algorithm" on page 4689.

ZEROPROB=*number*
tunes the threshold (a value between 0 and 1) below which the HPFMM procedure considers a component mixing probability to be zero. This affects the calculation of the number of effective components. The default is the square root of the machine epsilon; this is approximately 1E–8 on most computers.

BAYES Statement

BAYES *bayes-options* **;**

The BAYES statement requests that the parameters of the model be estimated by Markov chain Monte Carlo sampling techniques. The HPFMM procedure can estimate by maximum likelihood the parameters of all models supported by the procedure. Bayes estimation, on the other hand, is available for only a subset of these models.

In Bayesian analysis, it is essential to examine the convergence of the Markov chains before you proceed with posterior inference. With ODS Graphics turned on, the HPFMM procedure produces graphs at the end of the procedure output; these graphs enable you to visually examine the convergence of the chain. Inferences cannot be made if the Markov chain has not converged.

The output produced for a Bayesian analysis is markedly different from that for a frequentist (maximum likelihood) analysis for the following reasons:

- Parameter estimates do not have the same interpretation in the two analyses. Parameters are fixed unknown constants in the frequentist context and random variables in a Bayesian analysis.
- The results of a Bayesian analysis are summarized through chain diagnostics and posterior summary statistics and intervals.
- The HPFMM procedure samples the mixing probabilities in Bayesian models directly, rather than mapping them onto a logistic (or other) scale.

The HPFMM procedure applies highly specialized sampling algorithms in Bayesian models. For single-component models without effects, a conjugate sampling algorithm is used where possible. For models in the exponential family that contain effects, the sampling algorithm is based on Gamerman (1997). For the normal and *t* distributions, a conjugate sampler is the default sampling algorithm for models with and without effects. In multi-component models, the sampling algorithm is based on latent variable sampling through data augmentation (Frühwirth-Schnatter 2006) and the Gamerman or conjugate sampler. Because of this specialization, the options for controlling the prior distributions of the parameters are limited.

Table 58.3 summarizes the *bayes-options* available in the BAYES statement. The full assortment of options is then described in alphabetical order.

Table 58.3 BAYES Statement Options

Option	Description
Options Related to Sampling	
INITIAL=	Specifies how to construct initial values
NBI=	Specifies the number of burn-in samples
NMC=	Specifies the number of samples after burn-in
METROPOLIS	Forces a Metropolis-Hastings sampling algorithm even if conjugate sampling is possible
OUTPOST=	Generates a data set that contains the posterior estimates
THIN=	Controls the thinning of the Markov chain

Table 58.3 *continued*

Option	Description
Specification of Prior Information	
MIXPRIORPARMS	Specifies the prior parameters for the Dirichlet distribution of the mixing probabilities
BETAPRIORPARMS=	Specifies the parameters of the normal prior distribution for individual parameters in the $\boldsymbol{\beta}$ vector
MUPRIORPARMS=	Specifies the parameters of the prior distribution for the means in homogeneous mixtures without effects
PHIPRIORPARMS=	Specifies the parameters of the inverse gamma prior distribution for the scale parameters in homogeneous mixtures
PRIOROPTIONS	Specifies additional options used in the determination of the prior distribution
Posterior Summary Statistics and Convergence Diagnostics	
DIAGNOSTICS=	Displays convergence diagnostics for the Markov chain
STATISTICS	Displays posterior summary information for the Markov chain
Other Options	
ESTIMATE=	Specifies which estimate is used for the computation of OUTPUT statistics and graphics
TIMEINC=	Specifies the time interval to report on sampling progress (in seconds)

You can specify the following *bayes-options* in the BAYES statement.

BETAPRIORPARMS=*pair-specification*

BETAPRIORPARMS(*pair-specification* **...** *pair-specification***)**

specifies the parameters for the normal prior distribution of the parameters that are associated with model effects (βs). The *pair-specification* is of the form (a, b), and the values a and b are the mean and variance of the normal distribution, respectively. This option overrides the PRIOROPTIONS option.

The form of the BETAPRIORPARMS with an equal sign and a single pair is used to specify one pair of prior parameters that applies to all components in the mixture. In the following example, the two intercepts and the two regression coefficients all have a $N(0, 100)$ prior distribution:

```
proc hpfmm;
   model y = x / k=2;
   bayes betapriorparms=(0,100);
run;
```

You can also provide a list of pairs to specify different sets of prior parameters for the various regression parameters and components. For example:

```
proc hpfmm;
   model y = x/ k=2;
   bayes betapriorparms( (0,10) (0,20) (.,.) (3,100) );
```

```
run;
```

The simple linear regression in the first component has a $N(0, 10)$ prior for the intercept and a $N(0, 20)$ prior for the slope. The prior for the intercept in the second component uses the HPFMM default, whereas the prior for the slope is $N(3, 100)$.

DIAGNOSTICS=ALL | NONE | (*keyword-list*)

DIAG=ALL | NONE | (*keyword-list*)

controls the computation of diagnostics for the posterior chain. You can request all posterior diagnostics by specifying DIAGNOSTICS=ALL or suppress the computation of posterior diagnostics by specifying DIAGNOSTICS=NONE. The following *keywords* enable you to select subsets of posterior diagnostics; the default is DIAGNOSTICS=(AUTOCORR).

AUTOCORR < (LAGS= *numeric-list*) >

computes for each sampled parameter the autocorrelations of lags specified in the LAGS= list. Elements in the list are truncated to integers, and repeated values are removed. If the LAGS= option is not specified, autocorrelations are computed by default for lags 1, 5, 10, and 50. See the section "Autocorrelations" on page 174 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

ESS

computes an estimate of the effective sample size (Kass et al. 1998), the correlation time, and the efficiency of the chain for each parameter. See the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

GEWEKE < (*geweke-options*) >

computes the Geweke spectral density diagnostics (Geweke 1992), which are essentially a two-sample t test between the first f_1 portion and the last f_2 portion of the chain. The default is $f_1 = 0.1$ and $f_2 = 0.5$, but you can choose other fractions by using the following *geweke-options*:

FRAC1=*value*

specifies the fraction f_1 for the first window.

FRAC2=*value*

specifies the fraction f_2 for the second window.

See the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

HEIDELBERGER < (*Heidel-options*) >

HEIDEL < (*Heidel-options*) >

computes the Heidelberger and Welch diagnostic (which consists of a stationarity test and a half-width test) for each variable. The stationary diagnostic test tests the null hypothesis that the posterior samples are generated from a stationary process. If the stationarity test is passed, a half-width test is then carried out. See the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

These diagnostics are not performed by default. You can specify the DIAGNOSTICS=HEIDELBERGER option to request these diagnostics, and you can also specify suboptions, such as DIAGNOSTICS=HEIDELBERGER(EPS=0.05), as follows:

SALPHA=*value*

specifies the α level ($0 < \alpha < 1$) for the stationarity test. By default, SALPHA=0.05.

HALPHA=*value*

specifies the α level ($0 < \alpha < 1$) for the half-width test. By default, HALPHA=0.05.

EPS=*value*

specifies a small positive number ϵ such that if the half-width is less than ϵ times the sample mean of the retaining iterates, the half-width test is passed. By default, EPS=0.1.

MCERROR

MCSE

computes an estimate of the Monte Carlo standard error for each sampled parameter. See the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

MAXLAG=*n*

specifies the largest lag used in computing the effective sample size and the Monte Carlo standard error. Specifying this option implies the ESS and MCERROR options. The default is MAXLAG=250.

RAFTERY <(*Raftery-options***)>**

RL <(*Raftery-options***)>**

computes the Raftery and Lewis diagnostics, which evaluate the accuracy of the estimated quantile ($\hat{\theta}_Q$ for a given $Q \in (0, 1)$) of a chain. $\hat{\theta}_Q$ can achieve any degree of accuracy when the chain is allowed to run for a long time. The algorithm stops when the estimated probability $\hat{P}_Q = \Pr(\theta \leq \hat{\theta}_Q)$ reaches within $\pm$R of the value Q with probability S; that is, $\Pr(Q - R \leq \hat{P}_Q \leq Q + R) = S$. See the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The *Raftery-options* enable you to specify Q, R, S, and a precision level ϵ for a stationary test.

These diagnostics are not performed by default. You can specify the DIAGNOSTICS=RAFERTY option to request these diagnostics, and you can also specify suboptions, such as DIAGNOSTICS=RAFERTY(QUANTILE=0.05), as follows:

QUANTILE=*value*

Q=*value*

specifies the order (a value between 0 and 1) of the quantile of interest. By default, QUANTILE=0.025.

ACCURACY=*value*

R=*value*

specifies a small positive number as the margin of error for measuring the accuracy of estimation of the quantile. By default, ACCURACY=0.005.

PROB=*value*
S=*value*

specifies the probability of attaining the accuracy of the estimation of the quantile. By default, PROB=0.95.

EPS=*value*

specifies the tolerance level (a small positive number between 0 and 1) for the stationary test. By default, EPS=0.001.

MIXPRIORPARMS=K
MIXPRIORPARMS(*value-list***)**

specifies the parameters used in constructing the Dirichlet prior distribution for the mixing parameters. If you specify MIXPRIORPARMS=K, the parameters of the k-dimensional Dirichlet distribution are a vector that contains the number of components in the model (k), whatever that might be. You can specify an explicit list of parameters in *value-list*. If the MIXPRIORPARMS option is not specified, the default Dirichlet parameter vector is a vector of length k of ones. This results in a uniform prior over the unit simplex; for k=2, this is the uniform distribution. See the section "Prior Distributions" on page 4686 for the distribution function of the Dirichlet as used by the HPFMM procedure.

ESTIMATE=MEAN | MAP

determines which overall estimate is used, based on the posterior sample, in the computation of OUTPUT statistics and certain ODS graphics. By default, the arithmetic average of the (thinned) posterior sample is used. If you specify ESTIMATE=MAP, the parameter vector is used that corresponds to the maximum log posterior density in the posterior sample. In any event, a message is written to the SAS log if postprocessing results depend on a summary estimate of the posterior sample.

INITIAL=DATA | MLE | MODE | RANDOM

determines how initial values for the Markov chain are obtained. The default when a conjugate sampler is used is INITIAL=DATA, in which case the HPFMM procedure uses the same algorithm to obtain data-dependent starting values as it uses for maximum likelihood estimation. If no conjugate sampler is available or if you use the METROPOLIS option to explicitly request that it not be used, then the default is INITIAL=MLE, in which case the maximum likelihood estimates are used as the initial values. If the maximum likelihood optimization fails, the HPFMM procedure switches to the default INITIAL=DATA.

The options INITIAL=MODE and INITIAL=RANDOM use the mode and random draws from the prior distribution, respectively, to obtain initial values. If the mode does not exist or if it falls on the boundary of the parameter space, the prior mean is used instead.

METROPOLIS

requests that the HPFMM procedure use the Metropolis-Hastings sampling algorithm based on Gamerman (1997), even in situations where a conjugate sampler is available.

MUPRIORPARMS=*pair-specification*

MUPRIORPARMS(*pair-specification* **...** *pair-specification***)**

specifies the parameters for the means in homogeneous mixtures without regression coefficients. The *pair-specification* is of the form (a, b), where a and b are the two parameters of the prior distribution, optionally delimited with a comma. The actual distribution of the parameter is implied by the distribution selected in the MODEL statement. For example, it is a normal distribution for a mixture of normals, a gamma distribution for a mixture of Poisson variables, a beta distribution for a mixture of binary variables, and an inverse gamma distribution for a mixture of exponential variables. This option overrides the PRIOROPTIONS option.

The parameters correspond as follows:

Beta:	The parameters correspond to the α and β parameters of the beta prior distribution such that its mean is $\mu = \alpha/(\alpha + \beta)$ and its variance is $\mu(1 - \mu)/(\alpha + \beta + 1)$.
Normal:	The parameters correspond to the mean and variance of the normal prior distribution.
Gamma:	The parameters correspond to the α and β parameters of the gamma prior distribution such that its mean is α/β and its variance is α/β^2.
Inverse gamma:	The parameters correspond to the α and β parameters of the inverse gamma prior distribution such that its mean is $\mu = \beta/(\alpha - 1)$ and its variance is $\mu^2/(\alpha - 2)$.

The two techniques for specifying the prior parameters with the MUPRIORPARMS option are as follows:

- Specify an equal sign and a single pair of values:

```
proc hpfmm seed=12345;
   model y = / k=2;
   bayes mupriorparms=(0,50);
run;
```

- Specify a list of parameter pairs within parentheses:

```
proc hpfmm seed=12345;
   model y = / k=2;
   bayes mupriorparms( (.,.) (1.4,10.5));
run;
```

If you specify an invalid value (outside of the parameter space for the prior distribution), the HPFMM procedure chooses the default value and writes a message to the SAS log. If you want to use the default values for a particular parameter, you can also specify missing values in the *pair-specification*. For example, the preceding list specification assigns default values for the first component and uses the values 1.4 and 10.5 for the mean and variance of the normal prior distribution in the second component. The first example assigns a $N(0, 50)$ prior distribution to the means in both components.

NBI=*n*

specifies the number of burn-in samples. During the burn-in phase, chains are not saved. The default is NBI=2000.

NMC=*n*

SAMPLE=*n*

specifies the number of Monte Carlo samples after the burn-in. Samples after the burn-in phase are saved unless they are thinned with the THIN= option. The default is NMC=10000.

OUTPOST< (*outpost-options***) >=***data-set*

requests that the posterior sample be saved to a SAS data set. In addition to variables that contain log likelihood and log posterior values, the OUTPOST data set contains variables for the parameters. The variable names for the parameters are generic (Parm_1, Parm_2, . . ., Parm_p). The labels of the parameters are descriptive and correspond to the "Parameter Mapping" table that is produced when the OUTPOST= option is in effect.

You can specify the following *outpost-options* in parentheses:

LOGPRIOR

adds the value of the log prior distribution to the data set.

NONSINGULAR | NONSING | COMPRESS

eliminates parameters that correspond to singular columns in the design matrix (and were not sampled) from the posterior data set. This is the default.

SINGULAR | SING

adds columns of zeros to the data set in positions that correspond to singularities in the model or to parameters that were not sampled for other reasons. By default, these columns of zeros are not written to the posterior data set.

PHIPRIORPARMS=*pair-specification*

PHIPRIORPARMS(*pair-specification* **...** *pair-specification***)**

specifies the parameters for the inverse gamma prior distribution of the scale parameters (ϕ's) in the model. The *pair-specification* is of the form (a, b), and the values are chosen such that the prior distribution has mean $\mu = b/(a-1)$ and variance $\mu^2/(a-2)$.

The form of the PHIPRIORPARMS with an equal sign and a single pair is used to specify one pair of prior parameters that applies to all components in the mixture. For example:

```
proc hpfmm seed=12345;
   model y = / k=2;
   bayes phipriorparms=(2.001,1.001);
run;
```

The form with a list of pairs is used to specify different prior parameters for the scale parameters in different components. For example:

```
proc hpfmm seed=12345;
   model y = / k=2;
   bayes phipriorparms( (.,1.001)  (3.001,2.001) );
```

```
run;
```

If you specify an invalid value (outside of the parameter space for the prior distribution), the HPFMM procedure chooses the default value and writes a message to the SAS log. If you want to use the default values for a particular parameter, you can also specify missing values in the *pair-specification*. For example, the preceding list specification assigns default values for the first component a prior parameter and uses the value 1.001 for the b prior parameter. The second pair assigns 3.001 and 2.001 for the a and b prior parameters, respectively.

PRIOROPTIONS <=>(*prior-options*)

PRIOROPTS <=>(*prior-options*)

specifies options related to the construction of the prior distribution and the choice of their parameters. Some *prior-options* apply only in particular models. The BETAPRIORPARMS= and MUPRIORPARMS= options override this option.

You can specify the following *prior-options*:

CONDITIONAL | COND

chooses a conditional prior specification for the homogeneous normal and t distribution response components. The default prior specification in these models is an independence prior where the mean of the hth component has prior $\mu_h \sim N(a, b)$. The conditional prior is characterized by $\mu_h \sim N(a, \sigma_h^2/b)$.

DEPENDENT | DEP

chooses a data-dependent prior for the homogeneous models without effects. The prior parameters a and b are chosen as follows, based on the distribution in the MODEL statement:

Binary and binomial: $a = \bar{y}/(1-\bar{y})$, b=1, and the prior distribution for the success probability is $\mathrm{beta}(a, b)$.

Poisson: $a = 1$, $b = 1/\bar{y}$, and the prior distribution for μ is $\mathrm{gamma}(a, b)$. See Frühwirth-Schnatter (2006, p. 280) and Viallefont, Richardson, and Greene (2002).

Exponential: $a = 3$, $b = 2\bar{y}$, and the prior distribution for μ is inverse gamma with parameters a and b.

Normal and t: Under the default independence prior, the prior distribution for μ is $N(\bar{y}, fs^2)$ where f is the variance factor from the VAR= option and

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

Under the default conditional prior specification, the prior for μ_h is $N(a, \sigma_h^2/b)$ where $a = \bar{y}$ and $b = 2.6/(\max\{y\} - \min\{y\})$. The prior for the scale parameter is inverse gamma with parameters 1.28 and $0.36s^2$. For further details, see Raftery (1996) and Frühwirth-Schnatter (2006, p. 179).

VAR=*f*

specifies the variance for normal prior distributions. The default is VAR=1000. This factor is used, for example, in determining the prior variance of regression coefficients or in determining the prior variance of means in homogeneous mixtures of t or normal distributions (unless a data-dependent prior is used).

MLE< =*r* **>**

specifies that the prior distribution for regression variables be based on a multivariate normal distribution centered at the MLEs and whose dispersion is a multiple r of the asymptotic MLE covariance matrix. The default is MLE=10. In other words, if you specify PRIOROPTIONS(MLE), the HPFMM procedure chooses the prior distribution for the regression variables as $N(\widehat{\beta}, 10\mathrm{Var}[\widehat{\beta}])$ where $\widehat{\beta}$ is the vector of maximum likelihood estimates. The prior for the scale parameter is inverse gamma with parameters 1.28 and $0.36s^2$ where

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

For further details, see Raftery (1996) and Frühwirth-Schnatter (2006, p. 179). If you specify PRIOROPTIONS(MLE) for the regression parameters, then the data-dependent prior is used for the scale parameter; see the PRIOROPTIONS(DEPENDENT) option above.

The MLE option is not available for mixture models in which the parameters are estimated directly on the data scale, such as homogeneous mixture models or mixtures of distributions without model effects for which a conjugate sampler is available. By using the METROPOLIS option, you can always force the HPFMM procedure to abandon a conjugate sampler in favor of a Metropolis-Hastings sampling algorithm to which the MLE option applies.

STATISTICS < (*global-options***) > = ALL | NONE |** *keyword* **| (***keyword-list***)**

SUMMARIES < (*global-options***) > = ALL | NONE |** *keyword* **| (***keyword-list***)**

controls the number of posterior statistics produced. Specifying STATISTICS=ALL is equivalent to specifying STATISTICS=(SUMMARY INTERVAL). To suppress the computation of posterior statistics, specify STATISTICS=NONE. The default is STATISTICS=(SUMMARY INTERVAL). See the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

The *global-options* include the following:

ALPHA=*numeric-list*

controls the coverage levels of the equal-tail credible intervals and the credible intervals of highest posterior density (HPD) credible intervals. The ALPHA= values must be between 0 and 1. Each ALPHA= value produces a pair of $100(1-\alpha)\%$ equal-tail and HPD credible intervals for each sampled parameter. The default is ALPHA=0.05, which results in 95% credible intervals for the parameters.

PERCENT=*numeric-list*

requests the percentile points of the posterior samples. The values in *numeric-list* must be between 0 and 100. The default is PERCENT=(25 50 75), which yields for each parameter the 25th, 50th, and 75th percentiles, respectively.

The list of *keywords* includes the following:

SUMMARY
produces the means, standard deviations, and percentile points for the posterior samples. The default is to produce the 25th, 50th, and 75th percentiles; you can modify this list with the global PERCENT= option.

INTERVAL
produces equal-tail and HPD credible intervals. The default is to produce the 95% equal-tail credible intervals and 95% HPD credible intervals, but you can use the ALPHA= *global-option* to request credible intervals for any probabilities.

THIN=*n*

THINNING=*n*
controls the thinning of the Markov chain after the burn-in. Only one in every k samples is used when THIN=k, and if NBI=n_0 and NMC=n, the number of samples kept is

$$\left[\frac{n_0 + n}{k}\right] - \left[\frac{n_0}{k}\right]$$

where $[a]$ represents the integer part of the number a. The default is THIN=1—that is, all samples are kept after the burn-in phase.

TIMEINC=*n*
specifies a time interval in seconds to report progress during the burn-in and sampling phase. The time interval is approximate, since the minimum time interval in which the HPFMM procedure can respond depends on the multithreading configuration.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPFMM to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPFMM procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < *(options)* >. . . < *variable* < *(options)* > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. The CLASS statement must precede the MODEL statement.

The CLASS statement for SAS high-performance analytical procedures is documented in the section "CLASS Statement" on page 572. The HPFMM procedure also supports the following *global-option* in the CLASS statement:

UPCASE

uppercases the values of character-valued CLASS variables before levelizing them. For example, if the UPCASE option is in effect and a CLASS variable can take the values 'a,' 'A,' and 'b,' then 'a' and 'A' represent the same level and the CLASS variable is treated as having only two values: 'A' and 'B.'

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence for each observation. SAS high-performance analytical procedures that support the FREQ statement treat each observation as if it appeared f times, where f is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are transferred to output data sets created by SAS high-performance analytical procedures, provided that the output data set produces one or more records per input observation.

For more information about the common ID statement in SAS high-performance analytical procedures, see the section "ID Statement" on page 576.

MODEL Statement

MODEL *response* < **(***response-options***)** > **=** < *effects* > < / *model-options* > **;**

MODEL *events/trials* **=** < *effects* > < / *model-options* > **;**

MODEL **+** < *effects* > < / *model-options* > **;**

The MODEL statement defines elements of the mixture model, such as the model effects, the distribution, and the link function. At least one MODEL statement is required. You can specify more than one MODEL statement. Each MODEL statement identifies one or more components of a mixture. For example, if components differ in their distributions, link functions, or regressor variables, then you can use separate MODEL statements to define the components. If the finite mixture model is homogeneous—in the sense that all components share the same regressors, distribution, and link function—then you can specify the mixture model with a single MODEL statement by using the K= option.

An intercept is included in each model by default. It can be removed with the NOINT option.

The dependent variable can be specified by using either the *response* syntax or the *events/trials* syntax. The *events/trials* syntax is specific to models for binomial-type data. A binomial(n, π) variable is the sum of n independent Bernoulli trials with event probability π. Each Bernoulli trial results in either an event or a nonevent (with probability $1 - \pi$). The value of the second variable, *trials*, gives the number n of Bernoulli trials. The value of the first variable, *events*, is the number of events out of n. The values of both *events* and (*trials*–*events*) must be nonnegative, and the value of *trials* must be positive. Other distributions that allow the *events/trials* syntax are the beta-binomial distribution and the binomial cluster model.

If the *events/trials* syntax is used, the HPFMM procedure defaults to the binomial distribution. If you use the *response* syntax, the procedure defaults to the normal distribution unless the response variable is a character variable or listed in the CLASS statement.

The HPFMM procedure supports a continuation-style syntax in MODEL statements. Since a mixture has only one response variable, it is sufficient to specify the response variable in one MODEL statement. Other MODEL statements can use the continuation symbol "+" before the specification of effects. For example, the following statements fit a three-component binomial mixture model:

```
class A;
model y/n = x / k=2;
model     + A;
```

The first MODEL statement uses the "=" sign to separate response from effect information and specifies the response variable by using the *events/trials* syntax. This determines the distribution as binomial. This MODEL statement adds two components to the mixture models with different intercepts and regression slopes. The second MODEL statement adds another component to the mixture where the mean is a function of the classification main effect for variable A. The response is also binomial; it is a continuation from the previous MODEL statement.

There are two sets of options in the MODEL statement. The *response-options* determine how the HPFMM procedure models probabilities for binary data. The *model-options* control other aspects of model formation and inference. Table 58.4 summarizes the *response-options* and *model-options* available in the MODEL statement. These are subsequently discussed in detail in alphabetical order by option category.

Table 58.4 Summary of MODEL Statement Options

Option	Description
Response Variable Options	
DESCENDING	Reverses the order of response categories
EVENT=	Specifies the event category in binary models
ORDER=	Specifies the sort order for the response variable
REFERENCE=	Specifies the reference category in categorical models
Model Building	
DIST=	Specifies the response distribution
LINK=	Specifies the link function
K=	Specifies the number of mixture components
KMAX=	Specifies the maximum number of mixture components
KMIN=	Specifies the minimum number of mixture components
KRESTART	Requests that the starting values for each analysis be determined separately instead of sequentially
NOINT	Excludes fixed-effect intercept from model
OFFSET=	Specifies the offset variable for linear predictor
Statistical Computations and Output	
ALPHA=α	Determines the confidence level $(1-\alpha)$
CL	Displays confidence limits for fixed-effects parameter estimates
EQUATE=	Imposes simple equality constraints on parameters in this model
LABEL=	Identifies the model
PARMS	Provides starting values for the parameters in this model

Response Variable Options

Response variable options determine how the HPFMM procedure models probabilities for binary data.

You can specify the following *response-options* by enclosing them in parentheses after the *response* variable. The default is ORDER=FORMATTED.

DESCENDING

DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC HPFMM orders the response categories according to the ORDER= option and then reverses that order.

EVENT='*category*' | *keyword*

specifies the event category for the binary response model. PROC HPFMM models the probability of the event category. You can specify the value (formatted, if a format is applied) of the event category in quotes, or you can specify one of the following *keywords*:

FIRST
designates the first ordered category as the event. This is the default.

LAST
designates the last ordered category as the event.

ORDER=*order-type*
specifies the sort order for the levels of the response variable. You can specify the following values for *order-type*:

DATA
sorts the levels by order of appearance in the input data set.

FORMATTED
sorts the levels by external formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value.

FREQ
sorts the levels by descending frequency count; levels with the most observations come first in the order.

INTERNAL
sorts the levels by unformatted value.

FREQDATA
sorts the levels by order of descending frequency count, and within counts by order of appearance in the input data set when counts are tied.

FREQFORMATTED
sorts the levels by order of descending frequency count, and within counts by formatted value (as above) when counts are tied.

FREQINTERNAL
sorts the levels by order of descending frequency count, and within counts by unformatted value when counts are tied.

When ORDER=FORMATTED (the default) for numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC HPFMM run or in the DATA step that created the data set), the levels are ordered by their internal (numeric) value. If you specify the ORDER= option in the MODEL statement and the ORDER= option in the CLASS statement, the former takes precedence.

By default, ORDER=FORMATTED. For the FORMATTED and INTERNAL values, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | *keyword*

REF='*category*' | *keyword*

specifies the reference category for categorical models. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotes, or you can specify one of the following *keywords*:

FIRST

designates the first ordered category as the reference category.

LAST

designates the last ordered category as the reference category. This is the default.

Model Options

ALPHA=*number*

requests that confidence intervals be constructed for each of the parameters with confidence level $1 - number$. The value of *number* must be between 0 and 1; the default is 0.05.

CL

requests that confidence limits be constructed for each of the parameter estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

DISTRIBUTION=*keyword*

DIST=*keyword*

specifies the probability distribution for a mixture component.

If you specify the DIST= option and you do not specify a link function with the LINK= option, a default link function is chosen according to Table 58.5. If you do not specify a distribution, the HPFMM procedure defaults to the normal distribution for continuous response variables and to the binary distribution for classification or character variables, unless the *events/trial* syntax is used in the MODEL statement. If you choose the *events/trial* syntax, the HPFMM procedure defaults to the binomial distribution.

Table 58.5 lists *keywords* that you can specify for the DISTRIBUTION= option and the corresponding default link functions. For generalized linear models with these distributions, you can find expressions for the log-likelihood functions in the section "Log-Likelihood Functions for Response Distributions" on page 4677.

Table 58.5 Keyword Values of the DIST= Option

keyword	Alias	Distribution	Default Link Function
BETA		Beta	Logit
BETABINOMIAL	**BETABIN**	Beta-binomial	Logit
BINARY	**BERNOULLI**	Binary	Logit
BINOMIAL	**BIN**	Binomial	Logit
BINOMCLUSTER	**BINOMCLUS**	Binomial cluster	Logit
CONSTANT <(*c*)>	**DEGENERATE** <(*c*)>	Degenerate	N/A

Table 58.5 *continued*

DIST=	Alias	Distribution	Default Link Function
EXPONENTIAL	**EXPO**	Exponential	Log
FOLDEDNORMAL	**FNORMAL**	Folded normal	Identity
GAMMA	**GAM**	Gamma	Log
GAUSSIAN	**NORMAL**	Normal	Identity
GENPOISSON	**GPOISSON**	Generalized Poisson	Log
GEOMETRIC	**GEOM**	Geometric	Log
INVGAUSS	**IGAUSSIAN**, **IG**	Inverse Gaussian	Inverse squared (power(–2))
LOGNORMAL	**LOGN**	Lognormal	Identity
NEGBINOMIAL	**NEGBIN**, **NB**	Negative binomial	Log
POISSON	**POI**	Poisson	Log
T $<(\nu)>$	**STUDENT** $<(\nu)>$	t	Identity
TRUNCEXPO $<(a,b)>$	**TEXPO** $<(a,b)>$	Truncated exponential	Log
TRUNCLOGN $<(a,b)>$	**TLOGN** $<(a,b)>$	Lognormal	Identity
TRUNCNEGBIN	**TNEGBIN**, **TNB**	Negative binomial	Log
TRUNCNORMAL $<(a,b)>$	**TNORMAL** $<(a,b)>$	Truncated normal	Identity
TRUNCPOISSON	**TPOISSON**, **TPOI**	Truncated Poisson	Log
UNIFORM $<(a,b)>$	**UNIF** $<(a,b)>$	Uniform	N/A
WEIBULL		Weibull	Log

Note that the PROC HPFMM default link for the gamma or exponential distribution is not the canonical link (the reciprocal link).

The binomial cluster model is a two-component model described in Morel and Nagaraj (1993); Morel and Neerchal (1997); Neerchal and Morel (1998). See Example 58.1 for an application of the binomial cluster model in a teratological experiment.

If the *events/trials* syntax is used, the default distribution is the binomial and only the following choices are available: DIST=BINOMIAL, DIST=BETABINOMIAL, and DIST=BINOMCLUSTER. The *trials* variable is ignored for all other distributions. This enables you to fit models in which some components have a binomial or binomial-like distribution. For example, suppose that variable n is a binomial denominator and variable logn is its logarithm. Then the following statements model a two-component mixture of a binomial and Poisson count model:

```
model y/n  = ;
model      +  / dist=Poisson offset=logn;
```

The OFFSET= option is used in the second MODEL statement to specify that the Poisson counts refer to different base counts, since the trial variable n is ignored in the second model.

If DIST=BINOMIAL is specified without the *events/trials* syntax, then n=1 is used for the default number of trials.

DIST=TRUNCNEGBIN and DIST=TRUNCPOISSON are zero-truncated versions of DIST=NEGBINOMIAL and DIST=POISSON, respectively—that is, only the value of 0 is excluded from the support.

For DIST=TRUNCEXPO, DIST=TRUNCLOGN, and DIST=TRUNCNORMAL, you must specify the lower (*a*) and upper (*b*) truncation points of the distribution. For example:

DIST=TRUNCEXPO<(*a*,*b*)>

DIST=TRUNCLOGN<(*a*,*b*)>

DIST=TRUNCNORMAL<(*a*,*b*)>

Each of these distributions is the conditional version of its corresponding nontruncated distribution that is confined to the support $[a, b]$ (inclusive). You can specify a missing value (.) for either *a* or *b* to truncate only on the other side; that is, *a*=. indicates a right-truncated distribution, and *b*=. indicates a left-truncated distribution.

For several distribution specifications you can provide additional optional parameters to further define the distribution. These optional parameters are listed in the following:

CONSTANT<(*c*)> The number *c* specifies the value where the mass is concentrated. The default is DIST=CONSTANT(0), so you can add zero-inflation to any model by adding a MODEL statement with DIST=CONSTANT.

T<(ν)> The number ν specifies the degrees of freedom for the (shifted) t distribution. The default is DIST=T(3); this leads to a heavy-tailed distribution for which the variance is defined. See the section "Log-Likelihood Functions for Response Distributions" on page 4677 for the density function of the shifted t_ν distribution.

UNIFORM<(*a*,*b*)> The values *a* and *b* define the support of the uniform distribution, $a < b$. By default, $a = 0$ and $b = 1$.

EQUATE=MEAN | SCALE | NONE | EFFECTS(*effect-list***)**

specifies simple sets of parameter constraints across the components in a MODEL statement; the default is EQUATE=NONE. This option is available only for maximum likelihood estimation. If you specify EQUATE=MEAN, the parameters that determine the mean are reduced to a single set that is applicable to all components in the MODEL statement. If you specify EQUATE=SCALE, a single parameter represents the common scale for all components in the MODEL statement. The EFFECTS option enables you to force the parameters for the chosen model effects to be equal across components; however, the number of parameters is unaffected.

For example, the following statements fit a two-component multiple regression model in which the coefficients for variable logd vary by component and the intercepts and coefficients for variable dose are the same for the two components:

```
proc hpfmm;
   model num = dose logd / equate=effects(int dose) k=2;
run;
```

To fix all coefficients across the two components, you can write the MODEL statement as

```
model num = dose logd / equate=effects(int dose logd) k=2;
```

or

```
model num = dose logd / equate=mean k=2;
```

If you restrict all parameters in a *k*-component MODEL statement to be equal, the HPFMM procedure reduces the model to *k*=1.

K=*n*

NUMBER=*n*

specifies the number of components the MODEL statement contributes to the overall mixture. For the binomial cluster model, this option is not available, since this model is a two-component model by definition.

KMAX=*n*

specifies the maximum number of components the MODEL statement contributes to the overall mixture.

If the maximum number of components in the mixture, as determined by all KMAX= options, is larger than the minimum number of components, the HPFMM procedure fits all possible models and displays summary fit information for the sequence of evaluated models. The "best" model according to the CRITERION= option in the PROC HPFMM statement is then chosen, and the remaining output and analyses performed by PROC HPFMM pertain to this "best" model.

When you use MCMC methods to estimate the parameters of a mixture, you need to ensure that the chain for a given value of *k* has converged; otherwise, comparisons among models that have varying numbers of components might not be meaningful. You can use the FITDETAILS option to display summary and diagnostic information for the MCMC chains from each model.

If you specify the KMIN= option but not the KMAX= option, then the default value for the KMAX= option is the value of the KMIN= option (unless KMIN=0, in which case the KMAX= option is set to 1).

KMIN=*n*

specifies the minimum number of components that the MODEL statement contributes to the overall mixture. When you use MCMC methods to estimate the parameters of a mixture, you need to ensure that the chain for a given value of *k* has converged; otherwise, comparisons among models that have varying numbers of components might not be meaningful.

KRESTART

requests that the starting values for each analysis (that is, for each unique number of components as determined by the KMIN= and KMAX= options) be determined separately, in the same way as if no other analyses were performed. If you do not specify the KRESTART option, then the starting values for each analysis are based on results from the previous analysis with one less component.

LABEL='*label*'
specifies an optional label for the model that is used to identify the model in printed output, on graphics, and in data sets created from ODS tables.

LINK=*keyword*
specifies the link function in the model. The *keywords* and expressions for the associated link functions are shown in Table 58.6.

Table 58.6 Link Functions in MODEL Statement of PROC HPFMM

LINK=	Alias	Link Function	$g(\mu) = \eta =$
CLOGLOG	**CLL**	Complementary log-log	$\log(-\log(1-\mu))$
IDENTITY	**ID**	Identity	μ
LOG		Log	$\log(\mu)$
LOGIT		Logit	$\log(\mu/(1-\mu))$
LOGLOG		Log-log	$-\log(-\log(\mu))$
PROBIT	**NORMIT**	Probit	$\Phi^{-1}(\mu)$
POWER(λ)	**POW**(λ)	Power with exponent λ=*number*	$\begin{cases} \mu^{\lambda} & \text{if } \lambda \neq 0 \\ \log(\mu) & \text{if } \lambda = 0 \end{cases}$
POWERMINUS2		Power with exponent –2	$1/\mu^2$
RECIPROCAL	**INVERSE**	Reciprocal	$1/\mu$

The default link functions for the various distributions are shown in Table 58.5.

NOINT
requests that no intercept be included in the model. An intercept is included by default, unless the distribution is DIST=CONSTANT or DIST=UNIFORM.

OFFSET=*variable*
specifies the offset variable function for the linear predictor in the model. An offset variable can be thought of as a regressor variable whose regression coefficient is known to be 1. For example, you can use an offset in a Poisson model when counts have been obtained in time intervals of different lengths. With a log link function, you can model the counts as Poisson variables with the logarithm of the time interval as the offset variable.

PARAMETERS(*parameter-specification***)**
PARMS(*parameter-specification***)**
specifies starting values for the model parameters. If no PARMS option is given, the HPFMM procedure determines starting values by a data-dependent algorithm. To determine initial values for the Markov chain with Bayes estimation, see also the INITIAL= option in the BAYES statement. The specification of the parameters takes the following form: parameters in the mean function precede the scale parameters, and parameters for different components are separated by commas.

The following statements specify starting parameters for a two-component normal model. The initial values for the intercepts are 1 and –3; the initial values for the variances are 0.5 and 4.

```
proc hpfmm;
   model y = / k=2 parms(1 0.5, -3 4);
run;
```

You can specify missing values for parameters whose starting values are to be determined by the default method. Only values for parameters that participate in the optimization are specified. The values for model effects are specified on the linear (linked) scale.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< *keyword*< (*keyword-options*) > < =*name* > > . . .
< *keyword*< (*keyword-options*) > < =*name* > > < / *options* > ;

The OUTPUT statement creates a data set that contains observationwise statistics that are computed after fitting the model. The variables in the input data set are *not* included in the output data set to avoid data duplication for large data sets; however, variables specified in the ID statement are included.

The output statistics are computed based on the parameter estimates of the converged model if the parameters are estimated by maximum likelihood. If a Bayesian analysis is performed, the output statistics are computed based on the arithmetic mean in the posterior sample. You can change to the maximum posterior estimate with the ESTIMATE=MAP option in the BAYES statement.

You can specify the following syntax elements in the OUTPUT statement before the slash (/).

OUT=*SAS-data-set*
specifies the name of the output data set. If the OUT= option is omitted, the procedure uses the DATA*n* convention to name the output data set.

keyword< (*keyword-options*) > < =*name* >
specifies a statistic to include in the output data set and optionally assigns the variable the name *name*. If you do not provide a name, the HPFMM procedure assigns a default name based on the type of statistic requested. If you provide a name for a statistic that leads to multiple output statistics, the name is modified to index the associated component number.

You can use the *keyword-options* to control which type of a particular statistic is computed. The following are valid values for *keyword* and *keyword-options*:

PREDICTED< (**COMPONENT** | **OVERALL**) >

PRED< (**COMPONENT** | **OVERALL**) >

MEAN< (**COMPONENT** | **OVERALL**) >
requests predicted values (predicted means) for the response variable. The predictions in the output data set are mapped onto the data scale in all cases except for a binomial or binary response with *events/trials* syntax and when PREDTYPE=COUNT has not been specified. In that case the predictions are predicted success probabilities.

The default is to compute the predicted value for the mixture (OVERALL). You can request predictions for the means of the component distributions by adding the COMPONENT suboption in parentheses. The predicted values for some distributions are not identical to the parameter

modeled as μ. For example, in the lognormal distribution the predicted mean is $\exp\{\mu + 0.5\phi\}$ where μ and ϕ are the parameters of an underlying normal process; see the section "Log-Likelihood Functions for Response Distributions" on page 4677 for details.

RESIDUAL< (COMPONENT | OVERALL) >
RESID< (COMPONENT | OVERALL) >

requests residuals for the response or residuals in the component distributions. Only "raw" residuals on the data scale are computed (observed minus predicted).

VARIANCE< (COMPONENT | OVERALL) >
VAR< (COMPONENT | OVERALL) >

requests variances for the mixture or the component distributions.

LOGLIKE< (COMPONENT | OVERALL) >
LOGL< (COMPONENT | OVERALL) >

requests values of the log-likelihood function for the mixture or the components. For observations used in the analysis, the overall computed value is the observations' contribution to the log likelihood; if a FREQ statement is present, the frequency is accounted for in the computed value. In other words, if all observations in the input data set have been used in the analysis, adding the value of the log-likelihood contributions in the OUTPUT data set produces the negative of the final objective function value in the "Iteration History" table. By default, the log-likelihood contribution to the mixture is computed. You can request the individual mixture component contributions with the COMPONENT suboption.

MIXPROBS< (COMPONENT | MAX) >
MIXPROB< (COMPONENT | MAX) >
PRIOR< (COMPONENT | MAX) >
MIXWEIGHTS< (COMPONENT | MAX) >

requests that the prior weights $\pi_j(\mathbf{z}, \boldsymbol{\alpha}_j)$ be added to the OUTPUT data set. By default, the probabilities are output for all components. You can limit the output to a single statistic, the largest mixing probability, with the MAX suboption.

NOTE: The keyword "prior" is used here because of long-standing practice to refer to the mixing probabilities as prior weights. This must not be confused with the prior distribution and its parameters in a Bayesian analysis.

POSTERIOR< (COMPONENT | MAX) >
POST< (COMPONENT | MAX) >
PROB< (COMPONENT | MAX) >

requests that the posterior weights

$$\frac{\pi_j(\mathbf{z}, \boldsymbol{\alpha}_j)\, p_j(y; \mathbf{x}_j'\boldsymbol{\beta}_j, \phi_j)}{\sum_{j=1}^{k} \pi_j(\mathbf{z}, \boldsymbol{\alpha}_j)\, p_j(y; \mathbf{x}_j'\boldsymbol{\beta}_j, \phi_j)}$$

be added to the OUTPUT data set. By default, the probabilities are output for all components. You can limit the output to a single statistic, the largest posterior probability, with the MAX suboption.

NOTE: The keyword "posterior" is used here because of long-standing practice to refer to these probabilities as posterior probabilities. This must not be confused with the posterior distribution in a Bayesian analysis.

LINP
XBETA
requests that the linear predictors for the models be added to the OUTPUT data set.

CLASS | CATEGORY | GROUP
adds the estimated component membership to the OUTPUT data set. An observation is associated with the component that has the highest posterior probability.

MAXPOST | MAXPROB
adds the highest posterior probability to the OUTPUT data set.

A *keyword* can appear multiple times. For example, the following OUTPUT statement requests predicted values for the mixture in addition to the predicted means in the individual components:

```
output out=hpfmmout pred=MixtureMean pred(component)=CompMean;
```

In a three-component model, this produces four variables in the hpfmmout data set: MixtureMean, CompMean_1, CompMean_2, and CompMean_3.

You can specify the following *options* in the OUTPUT statement after a slash (/).

ALLSTATS
requests that all statistics are computed. If you do not use a *keyword* to assign a name, the HPFMM procedure uses the default name.

PREDTYPE=PROB | COUNT
specifies the type of predicted values that are produced for a binomial or binary response with *events/trials* syntax. If PREDTYPE=PROB, the predicted values are success probabilities. If PREDTYPE=COUNT, the predicted values are success counts. The default is PREDTYPE=PROB.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of the HPFMM procedure.

The PERFORMANCE statement is documented further in the section "PERFORMANCE Statement" on page 569.

PROBMODEL Statement

PROBMODEL < *effects* > < / *probmodel-options* > **;**

The PROBMODEL statement defines the model effects for the mixing probabilities and their link function and starting values. Model effects (other than the implied intercept) are not supported with Bayesian estimation. By default, the HPFMM procedure models mixing probabilities on the logit scale for two-component models and as generalized logit models in situations with more than two components. The PROBMODEL statement is not required.

The generalized logit model with k categories has a common vector of regressor or design variables, $\mathbf{z}$, $k-1$ parameter vectors that vary with category, and one linear predictor whose value is constant. The constant linear predictor is assigned by the HPFMM procedure to the last component in the model, and its value is zero ($\boldsymbol{\alpha}_k = \mathbf{0}$). The probability of observing category $1 \leq j \leq k$ is then

$$\pi_j(\mathbf{z}, \boldsymbol{\alpha}_j) = \frac{\exp\{\mathbf{z}'\boldsymbol{\alpha}_j\}}{\sum_{i=1}^{k} \exp\{\mathbf{z}'\boldsymbol{\alpha}_i\}}$$

For k=2, the generalized logit model reduces to a model with the logit link (a logistic model); hence the attribute *generalized* logit.

By default, an intercept is included in the model for the mixing probabilities. If you suppress the intercept with the NOINT option, you must specify at least one effect in the statement.

You can specify the following *probmodel-options* in the PROBMODEL statement after the slash (/):

ALPHA=*number*

requests that confidence intervals that have the confidence level $1 - \mathit{number}$ be constructed for the parameters in the probability model. The value of *number* must be between 0 and 1; the default is 0.05. If the probability model is simple—that is, it does not contain any effects—the confidence intervals are produced for the estimated parameters (on the logit scale) and for the mixing probabilities. This option has no effect when you perform Bayesian estimation. You can modify credible interval settings by specifying the STATISTICS(ALPHA=) option in the BAYES statement.

CL

requests that confidence limits be constructed for each of the parameter estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

LINK=*keyword*

specifies the link function in the model for the mixing probabilities. The default is a logit link for models with two components. For models with more than two components, only the generalized logit link is available. The *keywords* and expressions for the associated link functions for two-component models are shown in Table 58.7.

Table 58.7 Link Functions in the PROBMODEL Statement

LINK=	Link Function	$g(\mu) = \eta =$
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$
LOGIT	Logit	$\log(\mu/(1-\mu))$
LOGLOG	Log-log	$-\log(-\log(\mu))$
PROBIT | NORMIT	Probit	$\Phi^{-1}(\mu)$

NOINT

requests that no intercept be included in the model for the mixing probabilities. An intercept is included by default. If you suppress the intercept with the NOINT option, you must specify at least one other effect for the mixing probabilities—since an empty probability model is not meaningful.

PARAMETERS(*parameter-specification***)**

PARMS(*parameter-specification***)**

specifies starting values for the parameters. The specification of the parameters takes the following form: parameters in the mean function appear in a list, and parameters for different components are separated by commas. Starting values are given on the linked scale, not in terms of probabilities. Also, you need to specify starting values for each of the first *k–1* components in a *k*-component model. The linear predictor for the last component is always assumed to be zero.

The following statements specify a three-component mixture of multiple regression models. The PROBMODEL statement does not list any effects, a standard "intercept-only" generalized logit model is used to model the mixing probabilities.

```
proc hpfmm;
   model y = x1 x2 / k=3;
   probmodel  / parms(2, 1);
run;
```

There are three linear predictors in the model for the mixing probabilities, α_1, α_2, and α_3. With starting values of $\alpha_1 = 2$, $\alpha_2 = 1$, and $\alpha_3 = 0$, this leads to initial mixing probabilities of

$$\pi_1 = \frac{e^2}{e^2 + e^1 + e^0} = 0.24$$

$$\pi_2 = \frac{e^1}{e^2 + e^1 + e^0} = 0.66$$

$$\pi_3 = \frac{e^0}{e^2 + e^1 + e^0} = 0.1$$

You can specify missing values for parameters whose starting values are to be determined by the default method.

RESTRICT Statement

RESTRICT < *'label'* > *constraint-specification* < **,** . . . **,** *constraint-specification* >
< *operator* < *value* > > < / *option* > **;**

The RESTRICT statement enables you to specify linear equality or inequality constraints among the parameters of a mixture model. These restrictions are incorporated into the maximum likelihood analysis. The RESTRICT statement is not available for a Bayesian analysis with the HPFMM procedure.

Following are reasons why you might want to place constraints and restrictions on the model parameters:

- to fix a parameter at a particular value
- to equate parameters in different components in a mixture
- to impose order conditions on the parameters in a model
- to specify contrasts among the parameters that the fitted model should honor

A restriction is composed of a left-hand side and a right-hand side, separated by an operator. If the operator and right-hand side are not specified, the restriction is assumed to be an equality constraint against zero. If the right-hand side is not specified, the value is assumed to be zero.

An individual *constraint-specification* is written in (nearly) the same form as estimable linear functions are specified in the ESTIMATE statement of the GLM, MIXED, or GLIMMIX procedure. The *constraint-specification* takes the form

model-effect value-list < . . . *model-effect value-list* > < **(SCALE=** *value***)** >

At least one *model-effect* must be specified followed by one or more values in the *value-list*. The values in the list correspond to the multipliers of the corresponding parameter that is associated with the position in the model effect. If you specify more values in the *value-list* than the *model-effect* occupies in the model design matrix, the extra coefficients are ignored.

To specify restrictions for effects in specific components in the model, separate the *constraint-specification* by commas. The following statements provide an example:

```
proc hpfmm;
   class A;
   model y/n = A x /  k = 2;
   restrict A 1 0 -1;
   restrict x 2, x -1 >= 0.5;
run;
```

The linear predictors for this two-component model can be written as

$$\eta_1 = \beta_{10} + \alpha_{11} A_1 + \cdots + \alpha_{1a} A_a + x\beta_{11}$$
$$\eta_2 = \beta_{20} + \alpha_{21} A_1 + \cdots + \alpha_{2a} A_a + x\beta_{21}$$

where A_k is the binary variable associated with the kth level of A.

The first RESTRICT statement applies only to the first component and specifies that the parameter estimates that are associated with the first and third level of the A effect are identical. In terms of the linear predictor, the restriction can be written as

$$\alpha_{11} - \alpha_{13} = 0$$

Now suppose that A has only two levels. Then the HPFMM procedure ignores the value –1 in the first RESTRICT statement and imposes the restriction

$$\alpha_{11} = 0$$

on the fitted model.

The second RESTRICT statement involves parameters in two different components of the model. In terms of the linear predictors, the restriction can be written as

$$2\beta_{11} - \beta_{21} \geq \frac{1}{2}$$

When restrictions are specified explicitly through the RESTRICT statement or implied through the EQUATE=EFFECTS option in the MODEL statement, the HPFMM procedure lists all restrictions after the model fit in a table of linear constraints and indicates whether a particular constraint is active at the converged solution.

The following operators can be specified to separate the left- and right-hand sides of the restriction: **=**, **>**, **<**, **>=**, **<=**.

Some distributions involve scale parameters (the parameter ϕ in the expressions of the log likelihood) and you can also use the *constraint-specification* to involve a component's scale parameter in a constraint. To this end, assign a value to the keyword SCALE, separated from the model effects and value lists with parentheses. The following statements fit a two-component normal model and restrict the component variances to be equal:

```
proc hpfmm;
   model y = / k=2;
   restrict int 0 (scale  1),
            int 0 (scale -1);
run;
```

The intercept specification is necessary because each *constraint-specification* requires at least one model effect. The zero coefficient ensures that the intercepts are not involved in the restriction. Instead, the RESTRICT statement leads to $\phi_1 - \phi_2 = 0$.

You can specify the following *option* in the RESTRICT statement after a slash (/).

DIVISOR=*value*

specifies a *value* by which all coefficients on the right-hand side and left-hand side of the restriction are divided.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement is used to perform a weighted analysis. Consult the section "Log-Likelihood Functions for Response Distributions" on page 4677 for expressions on how weight variables are included in the log-likelihood functions. Because the probability structure of a mixture model is different from that of a classical statistical model, the presence of a weight variable in a mixture model *cannot* be interpreted as altering the variance of an observation.

Observations with nonpositive or missing weights are not included in the PROC HPFMM analysis. If a WEIGHT statement is not included, all observations used in the analysis are assigned a weight of 1.

Details: HPFMM Procedure

A Gentle Introduction to Finite Mixture Models

The Form of the Finite Mixture Model

Suppose that you observe realizations of a random variable Y, the distribution of which depends on an unobservable random variable S that has a discrete distribution. S can occupy one of k states, the number of which might be unknown but is at least known to be finite. Since S is not observable, it is frequently referred to as a latent variable.

Let π_j denote the probability that S takes on state j. Conditional on $S = j$, the distribution of the response Y is assumed to be $f_j(y; \alpha_j, \boldsymbol{\beta}_j | S = j)$. In other words, each distinct state j of the random variable S leads to a particular distributional form f_j and set of parameters $\{\alpha_j, \boldsymbol{\beta}_j\}$ for Y.

Let $\{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$ denote the collection of α_j and $\boldsymbol{\beta}_j$ parameters across all $j = 1$ to k. The marginal distribution of Y is obtained by summing the joint distribution of Y and S over the states in the support of S:

$$\begin{aligned} f(y; \boldsymbol{\alpha}, \boldsymbol{\beta}) &= \sum_{j=1}^{k} \Pr(S = j)\, f(y; \alpha_j, \boldsymbol{\beta}_j | S = j) \\ &= \sum_{j=1}^{k} \pi_j\, f(y; \alpha_j, \boldsymbol{\beta}_j | S = j) \end{aligned}$$

This is a mixture of distributions, and the π_j are called the mixture (or prior) probabilities. Because the number of states k of the latent variable S is finite, the entire model is termed a finite mixture (of distributions) model.

The finite mixture model can be expressed in a more general form by representing $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ in terms of regressor variables and parameters with optional additional scale parameters for $\boldsymbol{\beta}$. The section "Notation for the Finite Mixture Model" on page 4611 develops this in detail.

Mixture Models Contrasted with Mixing and Mixed Models: Untangling the Terminology Web

Statistical terminology can have its limitations. The terms mixture, mixing, and mixed models are sometimes used interchangeably, causing confusion. Even worse, the terms arise in related situations. One application needs to be eliminated from the discussion in this documentation: mixture experiments, where design factors are the proportions with which components contribute to a blend, are not mixture models and do not fall under the purview of the HPFMM procedure. However, the data from a mixture experiment might be analyzed with a mixture model, a mixing model, or a mixed model, besides other types of statistical models.

Suppose that you observe realizations of random variable Y and assume that Y follows some distribution $f(y;\boldsymbol{\alpha},\boldsymbol{\beta})$ that depends on parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Furthermore, suppose that the model is found to be deficient in the sense that the variability implied by the fitted model is less than the observed variability in the data, a condition known as overdispersion (see the section "Overdispersion" on page 4676). To tackle the problem the statistical model needs to be modified to allow for more variability. Clearly, one way of doing this is to introduce additional random variables into the process. Mixture, mixing, and mixed models are simply different ways of adding such random variables. The section "The Form of the Finite Mixture Model" on page 4674 explains how mixture models add a discrete state variable S. The following two subsections explain how mixing and mixed models instead assume variation for a natural parameter or in the mean function.

Mixing Models

Suppose that the model is modified to allow for some random quantity U, which might be one of the parameters of the model or a quantity related to the parameters. Now there are two distributions to cope with: the conditional distribution of the response given the random effect U,

$$f(y;\boldsymbol{\alpha},\boldsymbol{\beta}|u)$$

and the marginal distribution of the data. If U is continuous, the marginal distribution is obtained by integration:

$$f(y;\boldsymbol{\alpha},\boldsymbol{\beta}) = \int f(y;\boldsymbol{\alpha},\boldsymbol{\beta}|u)\, f(u)\, du$$

Otherwise, it is obtained by summation over the support of U:

$$f(y;\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum_u \Pr(U = u)\, f(y;\boldsymbol{\alpha},\boldsymbol{\beta}|u)$$

The important entity for statistical estimation is the marginal distribution $f(y;\boldsymbol{\alpha},\boldsymbol{\beta})$; the conditional distribution is often important for model description, genesis, and interpretation.

In a mixing model the marginal distribution is known and is typically of a well-known form. For example, if $Y|n$ has a binomial(n,μ) distribution and n follows a Poisson distribution, then the marginal distribution of Y is Poisson. The preceding operation is called *mixing* a binomial distribution with a Poisson distribution. Similarly, when mixing a Poisson(λ) distribution with a gamma(a,b) distribution for λ, a negative binomial distribution results as the marginal distribution. Other important mixing models involve mixing a binomial(n,μ) random variable with a beta(a,b) distribution for the binomial success probability μ. This results in a distribution known as the beta-binomial.

The finite mixtures have in common with the mixing models the introduction of random effects into the model to vary some or all of the parameters at random.

Mixed Models

The difference between a mixing and a mixed model is that the conditional distribution is not that important in the mixing model. It matters to motivate the overdispersed reference model and to arrive at the marginal distribution. Inferences with respect to the conditional distribution, such as predicting the random variable U, are not performed in mixing models. In a mixed model the random variable U typically follows a continuous distribution—almost always a normal distribution. The random effects usually do not model the natural parameters of the distribution; instead, they are involved in linear predictors that relate to the conditional mean. For example, a linear mixed model is a model in which the response and the random effects are normally distributed, and the random effects enter the conditional mean function linearly:

$$\begin{aligned}
\mathbf{Y} &= \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{U} + \boldsymbol{\epsilon} \\
\mathbf{U} &\sim N(\mathbf{0}, \mathbf{G}) \\
\boldsymbol{\epsilon} &\sim N(\mathbf{0}, \mathbf{R}) \\
\mathrm{Cov}[\mathbf{U}, \boldsymbol{\epsilon}] &= \mathbf{0}
\end{aligned}$$

The conditional and marginal distributions are then

$$\begin{aligned}
\mathbf{Y}|\mathbf{U} &\sim N(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{U} + \boldsymbol{\epsilon}, \mathbf{R}) \\
\mathbf{Y} &\sim N(\mathbf{X}\boldsymbol{\beta}, \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R})
\end{aligned}$$

For this model, because of the linearity in the mean and the normality of the random effects, you could also refer to mixing the normal vector $\mathbf{Y}$ with the normal vector $\mathbf{U}$, since the marginal distribution is known. The linear mixed model can be fit with the MIXED procedure. When the conditional distribution is not normal and the random effects are normal, the marginal distribution does not have a closed form. In this class of mixed models, called generalized linear mixed models, model approximations and numerical integration methods are commonly used in model fitting; see for example, those models fit by the GLIMMIX and NLMIXED procedures. Chapter 6, "Introduction to Mixed Modeling Procedures," contains details about the various classes of mixed models and about the relevant SAS/STAT procedures.

The previous expression for the marginal variance in the linear mixed model, $\mathrm{var}[\mathbf{Y}] = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$, emphasizes again that the variability in the marginal distribution of a model that contains random effects exceeds the variability in a model without the random effects ($\mathbf{R}$).

The finite mixtures have in common with the mixed models that the marginal distribution is not necessarily a well-known model, but is expressed through a formal integration over the random-effects distribution. In contrast to the mixed models, in particular those involving nonnormal distributions or nonlinear elements, this integration is rather trivial; it reduces to a weighted and finite sum of densities or mass functions.

Overdispersion

Overdispersion is the condition by which the data are more dispersed than is permissible under a reference model. Overdispersion arises only if the variability a model can capture is limited (for example, because of a functional relationship between mean and variance). For example, a model for normal data can never be overdispersed in this sense, although the reasons that lead to overdispersion also negatively affect a misspecified model for normal data. For example, omitted variables increase the residual variance estimate because variability that should have been modeled through changes in the mean is now "picked up" as error variability.

Overdispersion is important because an overdispersed model can lead to misleading inferences and conclusions. However, diagnosing and remedying overdispersion is complicated. In order to handle it appropriately,

the source of overdispersion must be identified. For example, overdispersion can arise from any of the following conditions alone or in combination:

- omitted variables and model effects
- omitted random effects (a source of random variation is not being modeled or is modeled as a fixed effect)
- correlation among the observations
- incorrect distributional assumptions
- incorrectly specified mean-variance relationships
- outliers in the data

As discussed in the previous section, introducing randomness into a system increases its variability. Mixture, mixed, and mixing models have thus been popular in modeling data that appear overdispersed. Finite mixture models are particularly powerful in this regard, because even low-order mixtures of basic, symmetric distributions (such as two- or three-component mixtures of normal or t distributions) enable you to model data with multiple modes, heavy tails, and skewness. In addition, the latent variable S provides a natural way to accommodate omitted, unobservable variables into the model.

One approach to remedy overdispersion is to apply simple modifications of the variance function of the reference model. For example, with binomial-type data this approach replaces the variance of the binomial count variable $Y \sim \text{Binomial}(n, \mu)$, $\text{Var}[Y] = n \times \mu(1-\mu)$ with a scaled version, $\phi n \times \mu(1-\mu)$, where ϕ is called an overdispersion parameter, $\phi > 0$.

In addressing overdispersion problems, it is important to tackle the problem at its root. A missing scale factor on the variance function is hardly ever the root cause of overdispersion; it is only the easiest remedy.

Log-Likelihood Functions for Response Distributions

The HPFMM procedure calculates the log likelihood that corresponds to a particular response distribution according to the following formulas. The response distribution is the distribution specified (or chosen by default) through the DIST= option in the MODEL statement. The parameterizations used for log-likelihood functions of these distributions were chosen to facilitate expressions in terms of mean parameters that are modeled through an (inverse) link functions and in terms of scale parameters. These are not necessarily the parameterizations in which parameters of prior distributions are specified in a Bayesian analysis of homogeneous mixtures. See the section "Prior Distributions" on page 4686 for details about the parameterizations of prior distributions.

The HPFMM procedure includes all constant terms in the computation of densities or mass functions. In the expressions that follow, l denotes the log-likelihood function, ϕ denotes a general scale parameter, μ_i is the "mean", and w_i is a weight from the use of a WEIGHT statement.

For some distributions (for example, the Weibull distribution) μ_i is not the mean of the distribution. The parameter μ_i is the quantity that is modeled as $g^{-1}(\mathbf{x}'\boldsymbol{\beta})$, where $g^{-1}(\cdot)$ is the inverse link function and the $\mathbf{x}$ vector is constructed based on the effects in the MODEL statement. Situations in which the parameter μ does not represent the mean of the distribution are explicitly mentioned in the list that follows.

The parameter ϕ is frequently labeled as a "Scale" parameter in output from the HPFMM procedure. It is not necessarily the scale parameter of the particular distribution.

Beta(μ, ϕ)

$$\begin{aligned} l(\mu_i, \phi; y_i, w_i) &= \log\left\{\frac{\Gamma(\phi/w_i)}{\Gamma(\mu_i\phi/w_i)\Gamma((1-\mu_i)\phi/w_i)}\right\} \\ &+ (\mu_i\phi/w_i - 1)\log\{y_i\} \\ &+ ((1-\mu_i)\phi/w_i - 1)\log\{1-y_i\} \end{aligned}$$

This parameterization of the beta distribution is due to Ferrari and Cribari-Neto (2004) and has properties $\mathrm{E}[Y] = \mu$, $\mathrm{Var}[Y] = \mu(1-\mu)/(1+\phi)$, $\phi > 0$.

Beta-binomial$(n; \mu, \phi)$

$$\begin{aligned} \phi &= (1-\rho^2)/\rho^2 \\ l(\mu_i, \rho; y_i) &= \log\{\Gamma(n_i+1)\} - \log\{\Gamma(y_i+1)\} \\ &- \log\{\Gamma(n_i - y_i + 1)\} \\ &+ \log\{\Gamma(\phi)\} - \log\{\Gamma(n_i+\phi)\} + \log\{\Gamma(y_i+\phi\mu_i)\} \\ &+ \log\{\Gamma(n_i - y_i + \phi(1-\mu_i))\} - \log\{\Gamma(\phi\mu_i)\} \\ &- \log\{\Gamma(\phi(1-\mu_i))\} \\ l(\mu_i, \rho; y_i, w_i) &= w_i l(\mu_i, \rho; y_i) \end{aligned}$$

where y_i and n_i are the *events* and *trials* in the *events/trials* syntax and $0 < \mu < 1$. This parameterization of the beta-binomial model presents the distribution as a special case of the Dirichlet-Multinomial distribution—see, for example, Neerchal and Morel (1998). In this parameterization, $\mathrm{E}[Y] = n\mu$ and $\mathrm{Var}[Y] = n\mu(1-\mu)(1+(n-1)/(\phi+1))$, $0 \le \rho \le 1$. The HPFMM procedure models the parameter ϕ and labels it "Scale" on the procedure output. For other parameterizations of the beta-binomial model, see Griffiths (1973) or Williams (1975).

Binomial$(n; \mu)$

$$\begin{aligned} l(\mu_i; y_i) &= y_i\log\{\mu_i\} + (n_i - y_i)\log\{1-\mu_i\} \\ &+ \log\{\Gamma(n_i+1)\} - \log\{\Gamma(y_i+1)\} \\ &- \log\{\Gamma(n_i - y_i + 1)\} \\ l(\mu_i; y_i, w_i) &= w_i\, l(\mu_i; y_i) \end{aligned}$$

where y_i and n_i are the *events* and *trials* in the *events/trials* syntax and $0 < \mu < 1$. In this parameterization $\mathrm{E}[Y] = n\mu$, $\mathrm{Var}[Y] = n\mu(1-\mu)$.

Binomial cluster$(n; \mu, \pi)$

$$\begin{aligned} z &= \log\{\Gamma(n_i+1)\} - \log\{\Gamma(y_i+1)\} - \log\{\Gamma(n_i - y_i + 1)\} \\ \mu_i^* &= (1-\mu_i)\pi \\ l(\mu_i, \pi; y_i) &= \log\{\pi\} + z + y_i\log\{\mu_i^* + \mu_i\} \\ &+ (n_i - y_i)\log\{1 - \mu_i^* - \mu_i\} \\ &+ \log\{1-\pi\} + z + y_i\log\{\mu_i^*\} \\ &+ (n_i - y_i)\log\{1-\mu_i^*\} \\ l(\mu_i, \pi; y_i, w_i) &= w_i l(\mu_i, \pi; y_i) \end{aligned}$$

In this parameterization, $\mathrm{E}[Y] = n\pi$ and $\mathrm{Var}[Y] = n\pi(1-\pi)\left\{1+\mu^2(n-1)\right\}$. The binomial cluster model is a two-component mixture of a binomial$(n, \mu^* + \mu)$ and a binomial(n, μ^*) random variable. This mixture is unusual in that it fixes the number of components and because the mixing probability π appears in the moments of the mixture components. For further details, see Morel and Nagaraj (1993); Morel and Neerchal (1997); Neerchal and Morel (1998) and Example 58.1 in this chapter. The expressions for the mean and variance in the binomial cluster model are identical to those of the beta-binomial model shown previously, with $\pi_{bc} = \mu_{bb}$, $\mu_{bc} = \rho_{bb}$.

The HPFMM procedure models the parameter μ through the MODEL statement and the parameter π through the PROBMODEL statement.

Constant(c)

$$l(y_i) = \begin{cases} 0 & y_i = c \\ -1\mathrm{E}20 & y_i \neq c \end{cases}$$

The extreme value when $y_i \neq c$ is chosen so that $\exp\{l(y_i)\}$ yields a likelihood of zero. You can change this value with the INVALIDLOGL= option in the PROC HPFMM statement. The constant distribution is useful for modeling overdispersion due to zero-inflation (or inflation of the process at support c).

The DIST=CONSTANT distribution is useful for modeling an inflated probability of observing a particular value (zero, by default) in data from other discrete distributions, as demonstrated in "Modeling Zero-Inflation: Is It Better to Fish Poorly or Not to Have Fished at All?" on page 4618. While it is syntactically valid to mix a constant distribution with a continuous distribution, such as DIST=LOGNORMAL, such a mixture is not mathematically appropriate, because the constant log-likelihood is the log of a probability, while a continuous log-likelihood is the log of a probability density function. If you want to mix a constant distribution with a continuous distribution, you could model the constant as a very narrow continuous distribution, such as DIST=UNIFORM($c-\delta$, $c+\delta$) for a small value δ. However, using PROC HPFMM to analyze such mixtures is sensitive to numerical inaccuracy and ultimately unnecessary. Instead, the following approach is mathematically equivalent and more numerically stable:

1. Estimate the mixing probability $\mathrm{P}(Y = c)$ as the proportion of observations in the data set such that $|y_i - c| < \epsilon$.
2. Estimate the parameters of the continuous distribution from the observations for which $|y_i - c| \geq \epsilon$.

Exponential(μ)

$$l(\mu_i; y_i, w_i) = \begin{cases} -\log\{\mu_i\} - y_i/\mu_i & w_i = 1 \\ w_i \log\left\{\frac{w_i y_i}{\mu_i}\right\} - \frac{w_i y_i}{\mu_i} - \log\{y_i \Gamma(w_i)\} & w_i \neq 1 \end{cases}$$

In this parameterization, $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \mu^2$.

Folded normal(μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\log\{2\pi\} - \frac{1}{2}\log\{\phi/w_i\} + \log\left\{\exp\left\{\frac{-w_i(y_i-\mu_i)^2}{2\phi}\right\} + \exp\left\{\frac{-w_i(y_i+\mu_i)^2}{2\phi}\right\}\right\}$$

If X has a normal distribution with mean μ and variance ϕ, then $Y = |X|$ has a folded normal distribution and log-likelihood function $l(\mu, \phi; y, w)$ for $y \geq 0$. The folded normal distribution arises, for example, when normally distributed measurements are observed, but their signs are not observed. The mean and variance of the folded normal in terms of the underlying $\mathrm{N}(\mu, \phi)$ distribution are

$$\begin{aligned}\mathrm{E}[Y] =& \frac{1}{\sqrt{2\pi\phi}} \exp\left\{-\frac{\mu^2}{2/\phi}\right\} + \mu\left(1 - 2\Phi\left(-\mu/\sqrt{\phi}\right)\right)\\ \mathrm{Var}[Y] =& \phi + \mu^2 - \mathrm{E}[Y]^2\end{aligned}$$

The HPFMM procedure models the folded normal distribution through the mean μ and variance ϕ of the underlying normal distribution. When the HPFMM procedure computes output statistics for the response variable (for example when you use the OUTPUT statement), the mean and variance of the response Y are reported. Similarly, the fit statistics apply to the distribution of $Y = |X|$, not the distribution of X. When you model a folded normal variable, the response input variable should be positive; the HPFMM procedure treats negative values of Y as a support violation.

Gamma(μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = w_i\phi \log\left\{\frac{w_i y_i \phi}{\mu_i}\right\} - \frac{w_i y_i \phi}{\mu_i} - \log\{y_i\} - \log\{\Gamma(w_i\phi)\}$$

In this parameterization, $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \mu^2/\phi$, respectively, where $\phi > 0$. This parameterization of the gamma distribution differs from that in the GLIMMIX procedure, which expresses the log-likelihood function in terms of $1/\phi$ in order to achieve a variance function suitable for mixed model analysis.

Geometric(μ)

$$\begin{aligned}l(\mu_i; y_i, w_i) = y_i \log\left\{\frac{\mu_i}{w_i}\right\} - (y_i + w_i)\log\left\{1 + \frac{\mu_i}{w_i}\right\}\\ + \log\left\{\frac{\Gamma(y_i + w_i)}{\Gamma(w_i)\Gamma(y_i + 1)}\right\}\end{aligned}$$

In this parameterization, $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \mu + \mu^2$. The geometric distribution is a special case of the negative binomial distribution with $\phi = 1$.

Generalized Poisson(μ, ϕ)

$$\begin{aligned}\xi_i &= 1 - \exp\{-\phi\}/w_i\\ \mu_i^* &= \mu_i - \xi(\mu_i - y_i)\\ l(\mu_i^*, \xi_i; y_i, w_i) &= \log\{\mu_i^* - \xi_i y_i\} + (y_i - 1)\log\{\mu_i^*\}\\ &\quad - \mu_i^* - \log\{\Gamma(y_i + 1)\}\end{aligned}$$

In this parameterization, $\mathrm{E}[Y] = \mu$, $\mathrm{Var}[Y] = \mu/(1 - \xi)^2$, and $\phi \geq 0$. The HPFMM procedure models the mean μ through the effects in the MODEL statement and applies a log link by default. The generalized Poisson distribution provides an overdispersed alternative to the Poisson distribution; $\phi = \xi_i = 0$ produces the mass function of a regular Poisson random variable. For details about the generalized Poisson distribution and a comparison with the negative binomial distribution, see Joe and Zhu (2005).

Inverse Gaussian(μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{y_i \phi \mu_i^2} + \log\left\{\frac{\phi y_i^3}{w_i}\right\} + \log\{2\pi\}\right]$$

The variance is $\mathrm{Var}[Y] = \phi\mu^3$, $\phi > 0$.

Lognormal(μ, ϕ)

$$z_i = \log\{y_i\} - \mu_i$$
$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left(2\log\{y_i\} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\} + \frac{w_i z_i^2}{\phi}\right)$$

If $X = \log\{Y\}$ has a normal distribution with mean μ and variance ϕ, then Y has the log-likelihood function $l(\mu_i, \phi; y_i, w_i)$. The HPFMM procedure models the lognormal distribution and not the "shortcut" version you can obtain by taking the logarithm of a random variable and modeling that as normally distributed. The two approaches are not equivalent, and the approach taken by PROC HPFMM is the actual lognormal distribution. Although the lognormal model is a member of the exponential family of distributions, it is not in the "natural" exponential family because it cannot be written in canonical form.

In terms of the parameters μ and ϕ of the underlying normal process for X, the mean and variance of Y are $\mathrm{E}[Y] = \exp\{\mu\}\sqrt{\omega}$ and $\mathrm{Var}[Y] = \exp\{2\mu\}\omega(\omega - 1)$, respectively, where $\omega = \exp\{\phi\}$. When you request predicted values with the OUTPUT statement, the HPFMM procedure computes $\mathrm{E}[Y]$ and not μ.

Negative binomial(μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = y_i \log\left\{\frac{\phi\mu_i}{w_i}\right\} - (y_i + w_i/\phi)\log\left\{1 + \frac{\phi\mu_i}{w_i}\right\} + \log\left\{\frac{\Gamma(y_i + w_i/\phi)}{\Gamma(w_i/\phi)\Gamma(y_i + 1)}\right\}$$

The variance is $\mathrm{Var}[Y] = \mu + \phi\mu^2$, $\phi > 0$.

For a given ϕ, the negative binomial distribution is a member of the exponential family. The parameter ϕ is related to the scale of the data because it is part of the variance function. However, it cannot be factored from the variance, as is the case with the ϕ parameter in many other distributions.

Normal(μ, ϕ)

$$l(\mu_i, \phi; y_i, w_i) = -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{\phi} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\}\right]$$

The mean and variance are $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \phi$, respectively, where $\phi > 0$.

Poisson(μ)

$$l(\mu_i; y_i, w_i) = w_i(y_i \log\{\mu_i\} - \mu_i - \log\{\Gamma(y_i + 1)\})$$

The mean and variance are $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y] = \mu$, respectively.

(Shifted) T$(\nu; \mu, \phi)$

$$
\begin{aligned}
z_i &= -0.5 \log\{\phi/w_i\} + \log\{\Gamma(0.5(\nu+1)\} \\
&\quad - \log\{\Gamma(0.5\nu)\} - 0.5 \times \log\{\pi\nu\} \\
l(\mu_i, \phi; y_i, w_i) &= -\left(\frac{\nu+1}{2}\right) \log\left\{1 + \frac{w_i}{\nu}\frac{(y_i-\mu_i)^2}{\phi}\right\} + z_i
\end{aligned}
$$

$\nu > 0, \phi > 0$. In this parameterization, $\mathrm{E}[Y] = \mu$ when $\nu > 1$ and $\mathrm{Var}[Y] = \phi\nu/(\nu-2)$ when $\nu > 2$. Note that this form of the t distribution is not a noncentral distribution, but that of a shifted central t random variable.

Truncated Exponential$(\mu; a, b)$

$$
\begin{aligned}
l(\mu_i; a, b, y_i, w_i) &= w_i \log\left\{\frac{w_i y_i}{\mu_i}\right\} - \frac{w_i y_i}{\mu_i} - \log\{y_i \Gamma(w_i)\} \\
&\quad - \log\left[\frac{\gamma\left(w_i, \frac{w_i b}{\mu_i}\right)}{\Gamma(w_i)} - \frac{\gamma\left(w_i, \frac{w_i a}{\mu_i}\right)}{\Gamma(w_i)}\right]
\end{aligned}
$$

where

$$
\gamma(c_1, c_2) = \int_0^{c_2} t^{c_1-1} \exp(-t)\mathrm{d}t
$$

is the lower incomplete gamma function. The mean and variance are

$$
\begin{aligned}
\mathrm{E}[Y] &= \frac{(a+\mu_i)\exp(-a/\mu_i) - (b+\mu_i)\exp(-b/\mu_i)}{\exp(-a/\mu_i) - \exp(-b/\mu_i)} \\
\mathrm{Var}[Y] &= \frac{(a^2 + 2a\mu_i + 2\mu_i^2)\exp(-a/\mu_i) - (b^2 + 2b\mu_i + 2\mu_i^2)\exp(-b/\mu_i)}{\exp(-a/\mu_i) - \exp(-b/\mu_i)} \\
&\quad - (\mathrm{E}[Y])^2
\end{aligned}
$$

Truncated Lognormal$(\mu, \phi; a, b)$

$$
\begin{aligned}
z_i &= \log\{y_i\} - \mu_i \\
l(\mu_i, \phi; a, b, y_i, w_i) &= -\frac{1}{2}\left(2\log\{y_i\} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\} + \frac{w_i z_i^2}{\phi}\right) \\
&\quad - \log\left\{\Phi\left[\sqrt{w_i/\phi}(\log b - \mu_i)\right] - \Phi\left[\sqrt{w_i/\phi}(\log a - \mu_i)\right]\right\}
\end{aligned}
$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. The mean and variance are

$$
\begin{aligned}
\mathrm{E}[Y] &= \exp(\mu_i + 0.5\phi)\frac{\Phi\left(\sqrt{\phi} - \frac{\log a - \mu_i}{\sqrt{\phi}}\right) - \Phi\left(\sqrt{\phi} - \frac{\log b - \mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{\log b - \mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{\log a - \mu_i}{\sqrt{\phi}}\right)} \\
\mathrm{Var}[Y] &= \exp(2\mu_i + 2\phi)\frac{\Phi\left(2\sqrt{\phi} - \frac{\log a - \mu_i}{\sqrt{\phi}}\right) - \Phi\left(2\sqrt{\phi} - \frac{\log b - \mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{\log b - \mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{\log a - \mu_i}{\sqrt{\phi}}\right)} - (\mathrm{E}[Y])^2
\end{aligned}
$$

Truncated Negative binomial(μ, ϕ)

$$
\begin{aligned}
l(\mu_i, \phi; y_i, w_i) = {} & y_i \log\left\{\frac{\phi\mu_i}{w_i}\right\} - (y_i + w_i/\phi)\log\left\{1 + \frac{\phi\mu_i}{w_i}\right\} \\
& + \log\left\{\frac{\Gamma(y_i + w_i/\phi)}{\Gamma(w_i/\phi)\Gamma(y_i + 1)}\right\} \\
& - \log\left\{1 - \left(\frac{\phi\mu_i}{w_i} + 1\right)^{-w_i/\phi}\right\}
\end{aligned}
$$

The mean and variance are

$$
\begin{aligned}
\mathrm{E}[Y] &= \mu_i \left\{1 - (\phi\mu_i + 1)^{-1/\phi}\right\}^{-1} \\
\mathrm{Var}[Y] &= (1 + \phi\mu_i + \mu_i)\mathrm{E}[Y] - (\mathrm{E}[Y])^2
\end{aligned}
$$

Truncated Normal$(\mu, \phi; a, b)$

$$
\begin{aligned}
l(\mu_i, \phi; a, b, y_i, w_i) = {} & -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{\phi} + \log\left\{\frac{\phi}{w_i}\right\} + \log\{2\pi\}\right] \\
& - \log\left\{\Phi\left[\sqrt{w_i/\phi}(b - \mu_i)\right] - \Phi\left[\sqrt{w_i/\phi}(a - \mu_i)\right]\right\}
\end{aligned}
$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. The mean and variance are

$$
\begin{aligned}
\mathrm{E}[Y] &= \mu_i + \sqrt{\phi}\frac{\mathrm{phi}\left(\frac{a-\mu_i}{\sqrt{\phi}}\right) - \mathrm{phi}\left(\frac{b-\mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{b-\mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{a-\mu_i}{\sqrt{\phi}}\right)} \\
\mathrm{Var}[Y] &= \phi\left[1 + \frac{\frac{a-\mu_i}{\sqrt{\phi}}\mathrm{phi}\left(\frac{a-\mu_i}{\sqrt{\phi}}\right) - \frac{b-\mu_i}{\sqrt{\phi}}\mathrm{phi}\left(\frac{b-\mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{b-\mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{a-\mu_i}{\sqrt{\phi}}\right)}\right. \\
&\quad \left. - \left\{\frac{\mathrm{phi}\left(\frac{a-\mu_i}{\sqrt{\phi}}\right) - \mathrm{phi}\left(\frac{b-\mu_i}{\sqrt{\phi}}\right)}{\Phi\left(\frac{b-\mu_i}{\sqrt{\phi}}\right) - \Phi\left(\frac{a-\mu_i}{\sqrt{\phi}}\right)}\right\}^2\right]
\end{aligned}
$$

where $\mathrm{phi}(\cdot)$ is the probability density function of the standard normal distribution.

Truncated Poisson(μ)

$$
l(\mu_i; y_i, w_i) = w_i(y_i \log\{\mu_i\} - \log\{\exp(\mu_i) - 1\} - \log\{\Gamma(y_i + 1)\})
$$

The mean and variance are

$$
\begin{aligned}
\mathrm{E}[Y] &= \frac{\mu}{1 - \exp(-\mu_i)} \\
\mathrm{Var}[Y] &= \frac{\mu_i\left[1 - \exp(-\mu_i) - \mu_i \exp(-\mu_i)\right]}{[1 - \exp(-\mu_i)]^2}
\end{aligned}
$$

Uniform(a, b)

$$
l(\mu_i; y_i, w_i) = -\log\{b - a\}
$$

The mean and variance are $\mathrm{E}[Y] = 0.5(a + b)$ and $\mathrm{Var}[Y] = (b - a)^2/12$, respectively.

Weibull(μ, ϕ)

$$l(\mu_i, \phi; y_i) = -w_i \left\{ \frac{\phi - 1}{\phi} \log\left(\frac{y_i}{\mu_i}\right) + \log(\mu_i \phi) + \exp\left[\log\left(\frac{y_i}{\mu_i}\right)/\phi\right] \right\}$$

In this particular parameterization of the two-parameter Weibull distribution, the mean and variance of the random variable Y are $\mathrm{E}[Y] = \mu\Gamma(1+\phi)$ and $\mathrm{Var}[Y] = \mu^2 \left\{\Gamma(1+2\phi) - \Gamma^2(1+\phi)\right\}$, respectively.

Bayesian Analysis

Conjugate Sampling

The HPFMM procedure uses Bayesian analysis via a conjugate Gibbs sampler if the model belongs to a small class of mixture models for which a conjugate sampler is available. See the section "Gibbs Sampler" on page 157 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a general discussion of Gibbs sampling. Table 58.8 summarizes the models for which conjugate and Metropolis-Hastings samplers are available.

Table 58.8 Availability of Conjugate and Metropolis Samplers in PROC HPFMM

Effects (exclusive of intercept)	Distributions	Available Samplers
No	Normal or T	Conjugate or Metropolis-Hastings
Yes	Normal or T	Conjugate or Metropolis-Hastings
No	Binomial, binary, Poisson, exponential	Conjugate or Metropolis-Hastings
Yes	Binomial, binary, Poisson, exponential	Metropolis-Hastings only

The conjugate sampler enjoys greater efficiency than the Metropolis-Hastings sampler and has the advantage of sampling in terms of the natural parameters of the distribution.

You can always switch to the Metropolis-Hastings sampling algorithm in any model by adding the METROPOLIS option in the BAYES statement.

Metropolis-Hastings Algorithm

If Metropolis-Hastings is the only sampler available for the specified model (see Table 58.8) or if the METROPOLIS option is specified in the BAYES statement, PROC HPFMM uses the Metropolis-Hastings approach of Gamerman (1997). See the section "Metropolis and Metropolis-Hastings Algorithms" on page 156 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a general discussion of the Metropolis-Hastings algorithm.

The Gamerman (1997) algorithm derives a specific density that is used to generate proposals for the component-specific parameters $\boldsymbol{\beta}_j$. The form of this proposal density is multivariate normal, with mean $\mathbf{m}_j$ and covariance matrix $\mathbf{C}_j$ derived as follows.

Suppose $\boldsymbol{\beta}_j$ is the vector of model coefficients in the jth component and suppose that $\boldsymbol{\beta}_j$ has prior distribution $N(\mathbf{a}, \mathbf{R})$. Consider a generalized linear model (GLM) with link function $g(\mu) = \eta = \mathbf{x}'\boldsymbol{\beta}$ and variance function $a(\mu)$. The pseudo-response and weight in the GLM for a weighted least squares step are

$$\begin{aligned} y^* =& \eta + (y - \mu)/\frac{\partial \mu}{\partial \eta} \\ w =& \frac{\partial \mu}{\partial \eta}/a(\mu) \end{aligned}$$

If the model contains offsets or FREQ or WEIGHT statements, or if a *trials* variable is involved, suitable adjustments are made to these quantities.

In each component, $j = 1, \ldots, k$, form an adjusted cross-product matrix with a "pseudo" border

$$\begin{bmatrix} \mathbf{X}_j'\mathbf{W}_j\mathbf{X}_j + \mathbf{R}^{-1} & \mathbf{X}_j'\mathbf{W}_j\mathbf{y}_j^* + \mathbf{R}^{-1}\mathbf{a} \\ \mathbf{y}_j^{*\prime}\mathbf{W}_j\mathbf{X}_j + \mathbf{a}'\mathbf{R}^{-1} & c \end{bmatrix}$$

where $\mathbf{W}_j$ is a diagonal matrix formed from the pseudo-weights w, $\mathbf{y}^*$ is a vector of pseudo-responses, and c is arbitrary. This is basically a system of normal equations with ridging, and the degree of ridging is governed by the precision and mean of the normal prior distribution of the coefficients. Sweeping on the leading partition leads to

$$\begin{aligned} \mathbf{C}_j =& \left(\mathbf{X}_j'\mathbf{W}_j\mathbf{X}_j + \mathbf{R}^{-1}\right)^- \\ \mathbf{m}_j =& \mathbf{C}_j\left(\mathbf{X}_j'\mathbf{W}_j\mathbf{y}_j^* + \mathbf{R}^{-1}\mathbf{a}\right) \end{aligned}$$

where the generalized inverse is a reflexive, g_2-inverse (see the section "Linear Model Theory" on page 55 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," for details).

PROC HPFMM then generates a proposed parameter vector from the resulting multivariate normal distribution, and then accepts or rejects this proposal according to the appropriate Metropolis-Hastings thresholds.

Latent Variables via Data Augmentation

In order to fit finite Bayesian mixture models, the HPFMM procedure treats the mixture model as a missing data problem and introduces an assignment variable **S** as in Dempster, Laird, and Rubin (1977). Since **S** is not observable, it is frequently referred to as a latent variable. The unobservable variable **S** assigns an observation to a component in the mixture model. The number of states, k, might be unknown, but it is known to be finite. Conditioning on the latent variable **S**, the component memberships of each observation is assumed to be known, and Bayesian estimation is straightforward for each component in the finite mixture model. That is, conditional on $S = j$, the distribution of the response is now assumed to be $f(y; \alpha_j, \beta_j | S = j)$. In other words, each distinct state of the random variable **S** leads to a distinct set of parameters. The parameters in each component individually are then updated using a conjugate Gibbs sampler (where available) or a Metropolis-Hastings sampling algorithm.

The HPFMM procedure assumes that the random variable **S** has a discrete multinomial distribution with probability π_j of belonging to a component j; it can occupy one of k states. The distribution for the latent variable **S** is

$$f(S_i = j | \pi_1, \ldots, \pi_k) = \text{multinomial}(1, \pi_1, \ldots, \pi_k)$$

where $f(\cdot|\cdot)$ denotes a conditional probability density. The parameters in the density π_j denote the probability that S takes on state j.

The HPFMM procedure assumes a conjugate Dirichlet prior distribution on the mixture proportions π_j written as:

$$p(\boldsymbol{\pi}) \quad = \quad \text{Dirichlet}(a_1, \ldots, a_k)$$

where $p(\cdot)$ indicates a prior distribution.

Using Bayes' theorem, the likelihood function and prior distributions determine a conditionally conjugate posterior distribution of $\mathbf{S}$ and $\boldsymbol{\pi}$ from the multinomial distribution and Dirichlet distribution, respectively.

Prior Distributions

The following list displays the parameterization of prior distributions for situations in which the HPFMM procedure uses a conjugate sampler in mixture models without model effects and certain basic distributions (binary, binomial, exponential, Poisson, normal, and t). You specify the parameters a and b in the formulas below in the MUPRIORPARMS= and PHIPRIORPARMS= options in the BAYES statement in these models.

Beta(a, b)

$$f(y) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\, y^{a-1}\,(1-y)^{b-1}$$

where $a > 0$, $b > 0$. In this parameterization, the mean and variance of the distribution are $\mu = a/(a+b)$ and $\mu(1-\mu)/(a+b+1)$, respectively. The beta distribution is the prior distribution for the success probability in binary and binomial distributions when conjugate sampling is used.

Dirichlet$(a_1, \ldots, a_k)$

$$f(\mathbf{y}) = \frac{\Gamma\left(\sum_{i=1}^{k} a_i\right)}{\prod_{i=1}^{k}\,\Gamma(a_i)}\, y_1^{a_1-1}\,\cdots\, y_k^{a_k-1}$$

where $\sum_{i=1}^{k} y_i = 1$ and the parameters $a_i > 0$. If any a_i were zero, an improper density would result. The Dirichlet density is the prior distribution for the mixture probabilities. You can affect the choice of the a_i through the MIXPRIORPARMS option in the BAYES statement. If k=2, the Dirichlet is the same as the beta(a, b) distribution.

Gamma(a, b)

$$f(y) = \frac{b^a}{\Gamma(a)}\, y^{a-1}\, \exp\{-by\}$$

where $a > 0$, $b > 0$. In this parameterization, the mean and variance of the distribution are $\mu = a/b$ and μ/b, respectively. The gamma distribution is the prior distribution for the mean parameter of the Poisson distribution when conjugate sampling is used.

Inverse gamma(a, b)

$$f(y) = \frac{b^a}{\Gamma(a)}\, y^{-a-1}\, \exp\{-b/y\}$$

where $a > 0$, $b > 0$. In this parameterization, the mean and variance of the distribution are $\mu = b/(a-1)$ if $a > 1$ and $\mu^2/(a-2)$ if $a > 2$, respectively. The inverse gamma distribution is the prior distribution for the mean parameter of the exponential distribution when conjugate sampling is used. It is also the prior distribution for the scale parameter ϕ in all models.

Multinomial$(1, \pi_1, \ldots, \pi_k)$

$$f(\mathbf{y}) = \frac{1}{y_1! \cdots y_k!} \pi_1^{y_1} \cdots \pi_k^{y_k}$$

where $\sum_{j=1}^{k} y_j = n$, $y_j \geq 0$, $\sum_{j=1}^{k} \pi_j = 1$, and n is the number of observations included in the analysis. The multinomial density is the prior distribution for the mixture proportions. The mean and variance of Y_j are $\mu_j = \pi_j$ and $\mu_j(1-\mu_j)$, respectively.

Normal(a, b)

$$f(y) = \frac{a}{\sqrt{2\pi b}} \exp\left\{-\frac{1}{2}\frac{(y-a)^2}{b}\right\}$$

where $b > 0$. The mean and variance of the distribution are $\mu = a$ and b, respectively. The normal distribution is the prior distribution for the mean parameter of the normal and t distribution when conjugate sampling is used.

When a MODEL statement contains effects or if you specify the METROPOLIS option, the prior distribution for the regression parameters is multivariate normal, and you can specify the means and variances of the parameters in the BETAPRIORPARMS= option in the BAYES statement.

Parameterization of Model Effects

PROC HPFMM constructs a finite mixture model according to the specifications in the CLASS, MODEL, and PROBMODEL statements. Each effect in the MODEL statement generates one or more columns in the matrix **X** for that model. The same **X** matrix applies to all components that are associated with the MODEL statement. Each effect in the PROBMODEL statement generates one or more columns in the matrix **Z** from which the linear predictors in the model for the mixture probability models is formed. The same **Z** matrix applies to all components.

The formation of effects from continuous and classification variables in the HPFMM procedure follows the same general rules and techniques as for other linear modeling procedures. For information about constructing the model effects, see the section "Parameterization of Model Effects" on page 412.

Computational Method

Multithreading

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to sequential (single-threaded) execution.

The number of threads spawned by the HPFMM procedure is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the CPU count with the CPUCOUNT= SAS system option. For example, if you specify the following statements, the HPFMM procedure schedules threads as if it were executing on a system that had four CPUs, regardless of the actual CPU count:

  ```
  options cpucount=4;
  ```

- You can specify the NTHREADS= option in the PERFORMANCE statement to determine the number of threads. This specification overrides the system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads is displayed in the "Performance Information" table, which is part of the default output. The HPFMM procedure allocates one thread per CPU.

The tasks that are multithreaded by the HPFMM procedure are primarily defined by dividing the data processed on a single machine among the threads—that is, the HPFMM procedure implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and you are running with four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. This operations include the following:

- variable levelization
- effect levelization
- formation of the crossproducts matrix
- objective function, gradient, and Hessian evaluations
- scoring of observations

In addition, operations on matrices such as sweeps might be multithreaded if the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Choosing an Optimization Algorithm

First- or Second-Order Algorithms

The factors that affect how you choose a particular optimization technique for a particular problem are complex. Occasionally, you might benefit from trying several algorithms.

For many optimization problems, computing the gradient takes more computer time than computing the function value. Computing the Hessian sometimes takes *much* more computer time and memory than computing the gradient, especially when there are many decision variables. Unfortunately, optimization techniques that do not use some kind of Hessian approximation usually require many more iterations than techniques that do use a Hessian matrix; as a result, the total run time of these techniques is often longer. Techniques that do not use the Hessian also tend to be less reliable. For example, they can terminate more easily at stationary points than at global optima.

Table 58.9 shows which derivatives are required for each optimization technique.

Table 58.9 Derivatives Required

Algorithm	First-Order	Second-Order
TRUREG	x	x
NEWRAP	x	x
NRRIDG	x	x
QUANEW	x	-
DBLDOG	x	-
CONGRA	x	-
LEVMAR	x	-
NMSIMP	-	-

The second-derivative methods (TRUREG, NEWRAP, and NRRIDG) are best for small problems for which the Hessian matrix is not expensive to compute. Sometimes the NRRIDG algorithm can be faster than the TRUREG algorithm, but TRUREG can be more stable. The NRRIDG algorithm requires only one matrix with $p(p+1)/2$ double words; TRUREG and NEWRAP require two such matrices. Here, p denotes the number of parameters in the optimization.

The first-derivative methods QUANEW and DBLDOG are best for medium-sized problems for which the objective function and the gradient are much faster to evaluate than the Hessian. In general, the QUANEW and DBLDOG algorithms require more iterations than TRUREG, NRRIDG, and NEWRAP, but each iteration can be much faster. The QUANEW and DBLDOG algorithms require only the gradient to update an approximate Hessian, and they require slightly less memory than TRUREG or NEWRAP.

The first-derivative method CONGRA is best for large problems for which the objective function and the gradient can be computed much faster than the Hessian and for which too much memory is required to store the (approximate) Hessian. In general, the CONGRA algorithm requires more iterations than QUANEW or DBLDOG, but each iteration can be much faster. Because CONGRA requires only a factor of p double-word memory, many large applications can be solved only by CONGRA.

The no-derivative method NMSIMP is best for small problems for which derivatives are not continuous or are very difficult to compute.

The LEVMAR method is appropriate only for least squares optimization problems.

Each optimization method uses one or more convergence criteria that determine when it has converged. An algorithm is considered to have converged when any one of the convergence criteria is satisfied. For example, under the default settings, the QUANEW algorithm converges if ABSGCONV <1E−5, FCONV $< 2 \times \epsilon$, or GCONV <1E−8.

By default, the HPFMM procedure applies the NRRIDG algorithm because it can take advantage of multi-threading in Hessian computations and inversions. If the number of parameters becomes large, specifying TECHNIQUE=QUANEW (which is a first-order method with good overall properties) is recommended.

Algorithm Descriptions

The following subsections provide details about each optimization technique and follow the same order as Table 58.9.

Trust Region Optimization (TRUREG)

The trust region method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function $f(\boldsymbol{\psi})$ have continuous first- and second-order derivatives inside the feasible region.

The trust region method iteratively optimizes a quadratic approximation to the nonlinear objective function within a hyperelliptic trust region that has radius Δ. The radius constrains the step size that corresponds to the quality of the quadratic approximation. The trust region method is implemented based on Dennis, Gay, and Welsch (1981); Gay (1983); Moré and Sorensen (1983).

The trust region method performs well for small- to medium-sized problems, and it does not need many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the quasi-Newton or conjugate gradient algorithms might be more efficient.

Newton-Raphson Optimization with Line Search (NEWRAP)

The NEWRAP technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

If second-order derivatives are computed efficiently and precisely, the NEWRAP method can perform well for medium-sized to large problems, and it does not need many function, gradient, and Hessian calls.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. Otherwise, a combination of ridging and line search is performed to compute successful steps. If the Hessian is not positive-definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive-definite (Eskow and Schnabel 1991).

In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation (LIS=2).

Newton-Raphson Ridge Optimization (NRRIDG)

The NRRIDG technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. If at least one of these two conditions is not satisfied, a multiple of the identity matrix is added to the Hessian matrix.

The NRRIDG method performs well for small- to medium-sized problems, and it does not require many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the quasi-Newton or conjugate gradient algorithms might be more efficient.

Because the NRRIDG technique uses an orthogonal decomposition of the approximate Hessian, each iteration of NRRIDG can be slower than an iteration of the NEWRAP technique, which works with a Cholesky decomposition. However, NRRIDG usually requires fewer iterations than NEWRAP.

Quasi-Newton Optimization (QUANEW)

The (dual) quasi-Newton method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$, and it does not need to compute second-order derivatives because they are approximated. It works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. However, in general it requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which compute second-order derivatives. QUANEW is the default optimization algorithm because it provides an appropriate balance between the speed and stability that are required for most nonlinear mixed model applications.

The QUANEW technique that is implemented by the HPFMM procedure is the dual quasi-Newton algorithm, which updates the Cholesky factor of an approximate Hessian.

In each iteration, a line search is performed along the search direction to find an approximate optimum. The line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions (Fletcher 1987). One of the Goldstein conditions can be violated if the feasible region defines an upper limit of the step size. Violating the left-side Goldstein condition can affect the positive-definiteness of the quasi-Newton update. In that case, either the update is skipped or the iterations are restarted by using an identity matrix, resulting in the steepest descent or ascent search direction.

The QUANEW algorithm uses its own line-search technique.

Double-Dogleg Optimization (DBLDOG)

The double-dogleg optimization method combines the ideas of the quasi-Newton and trust region methods. In each iteration, the double-dogleg algorithm computes the step $\mathbf{s}^{(k)}$ as the linear combination of the steepest descent or ascent search direction $\mathbf{s}_1^{(k)}$ and a quasi-Newton search direction $\mathbf{s}_2^{(k)}$:

$$\mathbf{s}^{(k)} = \alpha_1 \mathbf{s}_1^{(k)} + \alpha_2 \mathbf{s}_2^{(k)}$$

The step is requested to remain within a prespecified trust region radius (Fletcher 1987, p. 107). Thus, the DBLDOG subroutine uses the dual quasi-Newton update but does not perform a line search.

The double-dogleg optimization technique works well for medium-sized to moderately large optimization problems, where the objective function and the gradient are much faster to compute than the Hessian. The implementation is based on Dennis and Mei (1979); Gay (1983), but it is extended for dealing with boundary and linear constraints. The DBLDOG technique generally requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which require second-order derivatives; however, each of the DBLDOG iterations is computationally cheap. Furthermore, the DBLDOG technique requires only gradient calls for the update of the Cholesky factor of an approximate Hessian.

Conjugate Gradient Optimization (CONGRA)

Second-order derivatives are not required by the CONGRA algorithm and are not even approximated. The CONGRA algorithm can be expensive in function and gradient calls, but it requires only $O(p)$ memory

for unconstrained optimization. In general, the algorithm must perform many iterations to obtain a precise solution, but each of the CONGRA iterations is computationally cheap.

The CONGRA algorithm should be used for optimization problems that have large p. For the unconstrained or boundary-constrained case, the CONGRA algorithm requires only $O(p)$ bytes of working memory, whereas all other optimization methods require order $O(p^2)$ bytes of working memory. During p successive iterations, uninterrupted by restarts or changes in the working set, the CONGRA algorithm computes a cycle of p conjugate search directions. In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit for the step size. Other line-search algorithms can be specified with the LIS= option.

Levenberg-Marquardt Optimization (LEVMAR)

The LEVMAR algorithm performs a highly stable optimization; however, for large problems, it consumes more memory and takes longer than the other techniques. The Levenberg-Marquardt optimization technique is a slightly improved variant of the Moré (1978) implementation.

Nelder-Mead Simplex Optimization (NMSIMP)

The Nelder-Mead simplex method does not use any derivatives and does not assume that the objective function has continuous derivatives. The objective function itself needs to be continuous. This technique is quite expensive in the number of function calls, and it might be unable to generate precise results for $p \gg 40$.

The original Nelder-Mead simplex algorithm is implemented and extended to boundary constraints. This algorithm does not compute the objective for infeasible points, but it changes the shape of the simplex by adapting to the nonlinearities of the objective function. This adaptation contributes to an increased speed of convergence. NMSIMP uses a special termination criterion.

Output Data Set

Many procedures in SAS software add the variables from the input data set when an observationwise output data set is created. The assumption of high-performance analytical procedures is that the input data sets can be large and contain many variables. For performance reasons, the output data set contains the following:

- those variables explicitly created by the statement
- variables listed in the ID statement

This enables you to add output data set information that is necessary for subsequent SQL joins without copying the entire input data set to the output data set. For more information about output data sets see the section “Output Data Sets” on page 568.

Default Output

The following sections describe the output that PROC HPFMM produces by default. The output is organized into various tables, which are discussed in the order of appearance for maximum likelihood and Bayes estimation, respectively.

Performance Information

The "Performance Information" table is produced by default. The table displays the number of threads used.

Model Information

The "Model Information" table displays basic information about the model, such as the response variable, frequency variable, link function, and the model category that the HPFMM procedure determined based on your input and options. The "Model Information" table is one of a few tables that are produced irrespective of estimation technique. Most other tables are specific to Bayes or maximum likelihood estimation.

If the analysis depends on generated random numbers, the "Model Information" table also displays the random number seed used to initialize the random number generators. If you repeat the analysis and pass this seed value in the SEED= option in the PROC HPFMM statement, an identical stream of random numbers results.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels with the ORDER= option in the CLASS statement. You can suppress the "Class Level Information" table completely or partially with the NOCLPRINT= option in the PROC HPFMM statement.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used. If the *events/trials* syntax is used for the response, the table also displays the number of events and trials used in the analysis.

Note that the number of observations "used" in the analysis is not unambiguous in a mixture model. An observation that is "unusable" for one component distribution (because the response value is outside of the support of the distribution) might still be usable in the mixture model when the response value is in the support of another component distribution. You can affect the way in which PROC HPFMM handles exclusion of observations due to support violations with the EXCLUSION= option in the PROC HPFMM statement.

Response Profile

For binary data, the "Response Profile" table displays the ordered value from which the HPFMM procedure determines the probability being modeled as an event for binary data. For each response category level, the frequency used in the analysis is reported.

Default Output for Maximum Likelihood

Optimization Information

The "Optimization Information" table displays basic information about the optimization setup to determine the maximum likelihood estimates, such as the optimization technique, the parameters that participate in the optimization, and the number of threads used for the calculations. This table is not produced during model selection—that is, if the KMAX= option is specified in the MODEL statement.

Iteration History

The "Iteration History" table displays for each iteration of the optimization the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element. The objective function used in the optimization in the HPFMM procedure is the negative of the mixture log likelihood; consequently, PROC HPFMM performs a minimization. This table is not produced if you specify the KMAX= option in the MODEL statement. If you wish to see the "Iteration History" table in this setting, you must also specify the FITDETAILS option in the PROC HPFMM statement.

Convergence Status

The convergence status table is a small ODS table that follows the "Iteration History" table in the default output. In the listing, it appears as a message that identifies whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the "Convergence Status" table to an output data set, a numeric Status variable is added that allows you to assess convergence programmatically. The values of the Status variable encode the following:

0	Convergence was achieved or an optimization was not performed (because of TECHNIQUE=NONE).
1	The objective function could not be improved.
2	Convergence was not achieved because of a user interrupt or because a limit was exceeded, such as the maximum number of iterations or the maximum number of function evaluations. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC HPFMM statement.
3	Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Fit Statistics

The "Fit Statistics" table displays a variety of fit measures based on the mixture log likelihood in addition to the Pearson statistic. All statistics are presented in "smaller is better" form. If you are fitting a single-component normal, gamma, or inverse Gaussian model, the table also contains the unscaled Pearson statistic. If you are fitting a mixture model or the model has been fitted under restrictions, the table also contains the number of effective components and the number of effective parameters.

The calculation of the information criteria uses the following formulas, where p denotes the number of effective parameters, n denotes the number of observations used (or the sum of the frequencies used if a FREQ statement is present), and l is the log likelihood of the mixture evaluated at the converged estimates:

$$\mathrm{AIC} = -2l + 2p$$

$$\mathrm{AICC} = \begin{cases} -2l + 2pn/(n-p-1) & n > p+2 \\ -2l + 2p(p+2) & \text{otherwise} \end{cases}$$

$$\mathrm{BIC} = -2l + p\log(n)$$

The Pearson statistic is computed simply as

$$\text{Pearson statistic} = \sum_{i=1}^{n} f_i \frac{(y_i - \widehat{\mu}_i)^2}{\widehat{\mathrm{Var}}[Y_i]}$$

where n denotes the number of observations used in the analysis, f_i is the frequency associated with the ith observation (or 1 if no frequency is specified), μ_i is the mean of the mixture, and the denominator is the variance of the ith observation in the mixture. Note that the mean and variance in this expression are not those of the component distributions, but the mean and variance of the mixture:

$$\mu_i = \mathrm{E}[Y_i] = \sum_{j=1}^{k} \pi_{ij}\mu_{ij}$$

$$\mathrm{Var}[Y_i] = -\mu_i^2 + \sum_{j=1}^{k} \pi_{ij}\left(\sigma_{ij}^2 + \mu_{ij}^2\right)$$

where μ_{ij} and σ_{ij}^2 are the mean and variance, respectively, for observation i in the jth component distribution and π_{ij} is the mixing probability for observation i in component j.

The unscaled Pearson statistic is computed with the same expression as the Pearson statistic with n, f_i, and μ_i as previously defined, but the scale parameter ϕ is set to 1 in the $\widehat{\mathrm{Var}}[Y_i]$ expression.

The number of effective components and the number of effective parameters are determined by examining the converged solution for the parameters that are associated with model effects and the mixing probabilities. For example, if a component has an estimated mixing probability of zero, the values of its parameter estimates are immaterial. You might argue that all parameters should be counted towards the penalty in the information criteria. But a component with zero mixing probability in a k-component model effectively reduces the model to a $(k-1)$-component model. A situation of an overfit model, for which a parameter penalty needs to be taken when calculating the information criteria, is a different situation; here the mixing probability might be small, possibly close to zero.

Parameter Estimates

The parameter estimates, their estimated (asymptotic) standard errors, and p-values for the hypothesis that the parameter is zero are presented in the "Parameter Estimates" table. A separate table is produced for each MODEL statement, and the components that are associated with a MODEL statement are identified with an overall component count variable that counts across MODEL statements. If you assign a label to a model with the LABEL= option in the MODEL statement, the label appears in the title of the "Parameter Estimates" table. Otherwise, the internal label generated by the HPFMM procedure is used.

If the MODEL statement does not contain effects and the link function is not the identity, the inversely linked estimate is also displayed in the table. For many distributions, the inverse linked estimate is the estimated mean on the data scale. For example, in a binomial or binary model, it represents the estimated probability of an event. For some distributions (for example, the Weibull distribution), the inverse linked estimate is not the component distribution mean.

If you request confidence intervals with the CL or ALPHA= option in the MODEL statement, confidence limits are produced for the estimate on the linear scale. If the inverse linked estimate is displayed, confidence intervals for that estimate are also produced by inversely linking the confidence bounds on the linear scale.

Mixing Probabilities

If you fit a model with more than one component, the table of mixing probabilities is produced. If there are no effects in the PROBMODEL statement or if there is no PROBMODEL statement, the parameters are reported on the linear scale and as mixing probabilities. If model effects are present, only the linear parameters (on the scale of the logit, generalized logit, probit, and so on) are displayed.

Default Output for Bayes Estimation

Bayes Information

This table provides basic information about the sampling algorithm. The HPFMM procedure uses either a conjugate sampler or a Metropolis-Hastings sampling algorithm based on Gamerman (1997). The table reveals, for example, how many model parameters are sampled, how many parameters associated with mixing probabilities are sampled, and how many threads are used to perform multithreaded analysis.

Prior Distributions

The "Prior Distributions" table lists for each sampled parameter the prior distribution and its parameters. The mean and variance (if they exist) for those values of the parameters are also displayed, along with the initial value for the parameter in the Markov chain. The Component column in this table identifies the mixture component to which a particular parameter belongs. You can control how the HPFMM procedure determines initial values with the INITIAL= option in the BAYES statement.

Bayesian Fit Statistics

The "Bayesian Fit Statistics" table shows three measures based on the posterior sample. The "Average -2 Log Likelihood" is derived from the average mixture log-likelihood for the data, where the average is taken over the posterior sample. The deviance information criterion (DIC) is a Bayesian measure of model fit and the effective number of parameters (p_D) is a penalization term used in the computation of the DIC. See the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a detailed discussion of the DIC and p_D.

Posterior Summaries

The arithmetic mean, standard deviation, and percentiles of the posterior distribution of the parameter estimates are displayed in the "Posterior Summaries" table. By default, the HPFMM procedure computes the 25th, 50th (median), and 75th percentiles of the sampling distribution. You can modify the percentiles through suboptions of the STATISTICS option in the BAYES statement. If a parameter corresponds to a singularity in the design and was removed from sampling for that purpose, it is also displayed in the table of posterior summaries (and in other tables that relate to output from the BAYES statement). The posterior sample size for such a parameter is shown as N = 0.

Posterior Intervals

The table of "Posterior Intervals" displays equal-tail intervals and intervals of highest posterior density for each parameter. By default, intervals are computed for an α-level of 0.05, which corresponds to 95% intervals. You can modify this confidence level by providing one or more α values in the ALPHA= suboption of the STATISTICS option in the BAYES statement. The computation of these intervals is detailed in section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Posterior Autocorrelations

Autocorrelations for the posterior estimates are computed by default for autocorrelation lags 1, 5, 10, and 50, provided that a sufficient number of posterior samples is available. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for the computation of posterior autocorrelations and their utility in diagnosing convergence of Markov chains. You can modify the list of lags for which posterior autocorrelations are calculated with the AUTOCORR suboption of the DIAGNOSTICS= option in the BAYES statement.

ODS Table Names

Each table created by PROC HPFMM has a name associated with it, and you must use this name to reference the table when you use ODS statements. These names are listed in Table 58.10.

Table 58.10 ODS Tables Produced by PROC HPFMM

Table Name	Description	Required Statement / Option
Autocorr	Autocorrelation among posterior estimates	BAYES
BayesInfo	Basic information about Bayesian estimation	BAYES
ClassLevels	Level information from the CLASS statement	CLASS
CompDescription	Component description in models with varying number of components	KMAX= in MODEL with ML estimation
CompEvaluation	Comparison of mixture models with varying number of components	KMAX= in MODEL with ML estimation
CompInfo	Component information	COMPONENTINFO option in PROC HPFMM statement
ConvergenceStatus	Status of optimization at conclusion of optimization	Default output
Constraints	Linear equality and inequality constraints	RESTRICT statement or EQUATE=EFFECTS option in MODEL statement
Corr	Asymptotic correlation matrix of parameter estimates (ML) or empirical correlation matrix of the Bayesian posterior estimates	CORR option in PROC HPFMM statement

Table 58.10 *continued*

Table Name	Description	Required Statement / Option
Cov	Asymptotic covariance matrix of parameter estimates (ML) or empirical covariance matrix of the Bayesian posterior estimates	COV option in PROC HPFMM statement
CovI	Inverse of the covariance matrix of the parameter estimates	COVI option in PROC HPFMM statement
ESS	Effective sample size	DIAG=ESS option in BAYES statement
FitStatistics	Fit statistics	Default output
Geweke	Geweke diagnostics (Geweke 1992) for Markov chain	DIAG=GEWEKE option in BAYES statement
Hessian	Hessian matrix from the maximum likelihood optimization, evaluated at the converged estimates	HESSIAN
IterHistory	Iteration history	Default output for ML estimation
MCSE	Monte Carlo standard errors	DIAG=MCERROR in BAYES statement
MixingProbs	Solutions for the parameter estimates associated with effects in PROBMODEL statements	Default output for ML estimation if number of components is greater than 1
ModelInfo	Model information	Default output
NObs	Number of observations read and used, number of trials and events	Default output
OptInfo	Optimization information	Default output for ML estimation
ParameterEstimates	Solutions for the parameter estimates associated with effects in MODEL statements	Default output for ML estimation
ParameterMap	Mapping of parameter names to OUTPOST= data set	OUTPOST= option in BAYES statement
PriorInfo	Prior distributions and initial value of Markov chain	BAYES
PostSummaries	Summary statistics for posterior estimates	BAYES
PostIntervals	Equal-tail and highest posterior density intervals for posterior estimates	BAYES

Table 58.10 *continued*

Table Name	Description	Required Statement / Option
ResponseProfile	Response categories and category modeled	Default output in models with binary response
Timing	Absolute and relative times for tasks performed by the procedure	PERFORMANCE statement with DETAILS option

ODS Graphics

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC HPFMM generates are listed in Table 58.11, along with the required statements and options.

Table 58.11 Graphs Produced by PROC HPFMM

ODS Graph Name	Plot Description	Option
TADPanel	Panel of diagnostic graphics to assess convergence of Markov chains	BAYES
DensityPlot	Histogram and density with component distributions	Default plot for homogeneous mixtures
CriterionPanel	Panel of plots showing progression of model fit criteria for mixtures with different numbers of components	KMIN= and KMAX= options in MODEL statement

Examples: HPFMM Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 58.1: Modeling Mixing Probabilities: All Mice Are Equal, but Some Mice Are More Equal Than Others

(View the complete code for this example (hpfmmex1.sas) in the example repository.)

This example demonstrates how you can model the means and mixture proportions separately in a binomial cluster model. It also compares the binomial cluster model to the beta-binomial model.

In a typical teratological experiment, the offspring of animals that were exposed to a toxin during pregnancy are studied for malformation. If you count the number of malformed offspring in a litter of size n, then this count is typically not binomially distributed. The responses of the offspring from the same litter are not independent; hence their sum does not constitute a binomial random variable. Relative to a binomial model, data from teratological experiments exhibit overdispersion because ignoring positive correlation among the responses tends to overstate the precision of the parameter estimates. Overdispersion mechanisms are briefly discussed in the section "Overdispersion" on page 4676.

In this application, the focus is on mixtures and models that involve a mixing mechanism. The mixing approach (Williams 1975; Haseman and Kupper 1979) supposes that the binomial success probability is a random variable that follows a $\text{beta}(\alpha, \beta)$ distribution:

$$
\begin{aligned}
Y|\mu &\sim \text{Binomial}(n, \mu) \\
\mu &\sim \text{Beta}(\alpha, \beta) \\
Y &\sim \text{Beta-binomial}(n, \mu, \phi) \\
\text{E}[Y] &= n\pi \\
\text{Var}[Y] &= n\pi(1-\pi)\left\{1 + \mu^2(n-1)\right\}
\end{aligned}
$$

If $\mu = 0$, then the beta-binomial distribution reduces to a standard binomial model with success probability π. The parameterization of the beta-binomial distribution used by the HPFMM procedure is based on Neerchal and Morel (1998), see the section "Log-Likelihood Functions for Response Distributions" on page 4677 for details.

Morel and Nagaraj (1993); Morel and Neerchal (1997); Neerchal and Morel (1998) propose a different model to capture dependency within binomial clusters. Their model is a two-component mixture that gives rise to the same mean and variance function as the beta-binomial model. The genesis is different, however. In the binomial cluster model of Morel and Neerchal, suppose there is a cluster of n Bernoulli outcomes with success probability π. The number of responses in the cluster decomposes into $N \leq n$ outcomes that all respond with either "success" or "failure"; the important aspect is that they all respond identically. The remaining $n - N$ Bernoulli outcomes respond independently, so the sum of successes in this group is a binomial$(n - N, \pi)$ random variable. Denote the probability with which cluster members fall into the group of identical respondents as μ. Then $1 - \mu$ is the probability that a response belongs to the group of independent Bernoulli outcomes.

It is easy to see how this process of dividing the individual Bernoulli outcomes creates clustering. The binomial cluster model can be written as the two-component mixture

$$\Pr(Y = y) = \pi \Pr(U = y) + (1 - \pi) \Pr(V = y)$$

where $U \sim \text{Binomial}(n, \mu^* + \mu)$, $V \sim \text{Binomial}(n, \mu^*)$, and $\mu^* = (1 - \mu)\pi$. This mixture model is somewhat unusual because the mixing probability π appears as a parameter in the component distributions. The two probabilities involved, π and μ, have the following interpretation: π is the unconditional probability of success for any observation, and μ is the probability with which the Bernoulli observations respond identically. The complement of this probability, $1 - \mu$, is the probability with which the Bernoulli outcomes respond independently. If $\mu = 0$, then the two-component mixture reduces to a standard Binomial model with success probability π. Since both π and μ are involved in the success probabilities of the two Binomial variables in the mixture, you can affect these binomial means by specifying effects in the PROBMODEL statement (for the πs) or the MODEL statement (for the μs). In a "straight" two-component Binomial mixture,

$$\pi \text{Binomial}(n, \mu_1) + (1 - \pi)\text{Binomial}(n, \mu_2)$$

you would vary the success probabilities μ_1 and μ_2 through the MODEL statement.

With the HPFMM procedure, you can fit the beta-binomial model by specifying DIST=BETABIN and the binomial cluster model by specifying DIST=BINOMCLUS in the MODEL statement.

Morel and Neerchal (1997) report data from a completely randomized design that studies the teratogenicity of phenytoin in 81 pregnant mice. The treatment structure of the experiment is an augmented factorial. In addition to an untreated control, mice received 60 mg/kg of phenytoin (PHT), 100 mg/kg of trichloropropene oxide (TCPO), and their combination. The design was augmented with a control group that was treated with water. As in Morel and Neerchal (1997), the two control groups are combined here into a single group.

The following DATA step creates the data for this analysis as displayed in Table 1 of Morel and Neerchal (1997). The second DATA step creates continuous variables x1–x3 to match the parameterization of these authors.

```
data ossi;
   length tx $8;
   input tx$ n @@;
   do i=1 to n;
      input y m @@;
      output;
   end;
   drop i;
   datalines;
Control  18 8 8 9  9  7  9 0  5 3  3 5 8 9 10 5 8 5 8 1 6 0 5
            8 8 9 10  5  5 4  7 9 10 6 6 3  5
Control  17 8 9 7 10 10 10 1  6 6  6 1 9 8  9 6 7 5 5 7 9
            2 5 5  6  2  8 1  8 0  2 7 8 5  7
PHT      19 1 9 4  9  3  7 4  7 0  7 0 4 1  8 1 7 2 7 2 8 1 7
            0 2 3 10  3  7 2  7 0  8 0 8 1 10 1 1
TCPO     16 0 5 7 10  4  4 8 11 6 10 6 9 3  4 2 8 0 6 0 9
            3 6 2  9  7  9 1 10 8  8 6 9
PHT+TCPO 11 2 2 0  7  1  8 7  8 0 10 0 4 0  6 0 7 6 6 1 6 1 7
;
```

```
data ossi;
   set ossi;
   array xx{3} x1-x3;
   do i=1 to 3; xx{i}=0; end;
   pht  = 0;
   tcpo = 0;
   if (tx='TCPO') then do;
      xx{1} = 1;
      tcpo  = 100;
   end; else if (tx='PHT') then do;
      xx{2} = 1;
      pht   = 60;
   end; else if (tx='PHT+TCPO') then do;
      pht  = 60;
      tcpo = 100;
      xx{1} = 1; xx{2} = 1; xx{3}=1;
   end;
run;
```

The HPFMM procedure models the mean parameters μ through the MODEL statement and the mixing proportions π through the PROBMODEL statement. In the binomial cluster model, you can place a regression structure on either set of probabilities, and the regression structure does not need to be the same. In the following statements, the unconditional probability of ossification is modeled as a two-way factorial, whereas the intralitter effect—the propensity to group within a cluster—is assumed to be constant:

```
proc hpfmm data=ossi;
   class pht tcpo;
   model y/m = / dist=binomcluster;
   probmodel pht tcpo pht*tcpo;
run;
```

The CLASS statement declares the PHT and TCPO variables as classification variables. They affect the analysis through their levels, not through their numeric values. The MODEL statement declares the distribution of the data to follow a binomial cluster model. The HPFMM procedure then automatically assumes that the model is a two-component mixture. An intercept is included by default. The PROBMODEL statement declares the effect structure for the mixing probabilities. The unconditional probability of ossification of a fetus depends on the main effects and the interaction in the factorial.

The "Model Information" table displays important details about the model fit with the HPFMM procedure (Output 58.1.1). Although no K= option was specified in the MODEL statement, the HPFMM procedure recognizes the model as a two-component model. The "Class Level Information" table displays the levels and values of the PHT and TCPO variables. Eighty-one observations are read from the data and are used in the analysis. These observations comprise 287 events and 585 total outcomes.

Output 58.1.1 Model Information in Binomial Cluster Model with Constant Clustering Probability

The HPFMM Procedure

Model Information	
Data Set	WORK.OSSI
Response Variable (Events)	y
Response Variable (Trials)	m
Type of Model	Binomial Cluster
Distribution	Binomial Cluster
Components	2
Link Function	Logit
Estimation Method	Maximum Likelihood

Class Level Information		
Class	**Levels**	**Values**
pht	2	0 60
tcpo	2	0 100

Number of Observations Read	81
Number of Observations Used	81
Number of Events	287
Number of Trials	585

The "Optimization Information" table in Output 58.1.2 gives details about the maximum likelihood optimization. By default, the HPFMM procedure uses a quasi-Newton algorithm. The model contains five parameters, four of which are part of the model for the mixing probabilities. The fifth parameter is the intercept in the model for μ.

Output 58.1.2 Optimization in Binomial Cluster Model with Constant Clustering Probability

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	5
Mean Function Parameters	1
Scale Parameters	0
Mixing Prob Parameters	4

Output 58.1.2 *continued*

Iteration History				
Iteration	Evaluations	Objective Function	Change	Max Gradient
0	5	174.92723892		43.78769
1	2	154.13180744	20.79543149	11.2346
2	3	153.26693611	0.86487133	6.888215
3	2	152.84974281	0.41719329	3.541977
4	3	152.61756033	0.23218248	2.783556
5	3	152.54795303	0.06960730	1.146807
6	3	152.52684929	0.02110374	0.034367
7	3	152.52671214	0.00013715	0.011511
8	3	152.52670799	0.00000415	0.000202
9	3	152.52670799	0.00000000	4.001E-6

Convergence criterion (GCONV=1E-8) satisfied.

Fit Statistics	
-2 Log Likelihood	305.1
AIC (Smaller is Better)	315.1
AICC (Smaller is Better)	315.9
BIC (Smaller is Better)	327.0
Pearson Statistic	89.2077
Effective Parameters	5
Effective Components	2

After nine iterations, the iterative optimization converges. The –2 log likelihood at the converged solution is 305.1, and the Pearson statistic is 89.2077. The HPFMM procedure computes the Pearson statistic as a general goodness-of-fit measure that expresses the closeness of the fitted model to the data.

The estimates of the parameters in the conditional probability μ and in the unconditional probability π are given in Output 58.1.3. The intercept estimate in the model for μ is 0.3356. Since the default link in the binomial cluster model is the logit link, the estimate of the conditional probability is

$$\widehat{\mu} = \frac{1}{1 + \exp\{-0.3356\}} = 0.5831$$

This value is displayed in the "Inverse Linked Estimate" column. There is greater than a 50% chance that the individual fetuses in a litter provide the same response. The clustering tendency is substantial.

Output 58.1.3 Parameter Estimates in Binomial Cluster Model with Constant Clustering Probability

Parameter Estimates for Binomial Cluster Model						
Component	Effect	Estimate	Standard Error	z Value	Pr > |z|	Inverse Linked Estimate
1	Intercept	0.3356	0.1714	1.96	0.0503	0.5831

Output 58.1.3 *continued*

Parameter Estimates for Mixing Probabilities							
Component	Effect	pht	tcpo	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept			-1.2194	0.4690	-2.60	0.0093
1	pht	0		0.9129	0.5608	1.63	0.1036
1	pht	60		0	.	.	.
1	tcpo		0	0.3295	0.5534	0.60	0.5516
1	tcpo		100	0	.	.	.
1	pht*tcpo	0	0	0.6162	0.6678	0.92	0.3561
1	pht*tcpo	0	100	0	.	.	.
1	pht*tcpo	60	0	0	.	.	.
1	pht*tcpo	60	100	0	.	.	.

The "Mixing Probabilities" table displays the estimates of the parameters in the model for π on the logit scale (Output 58.1.3). Table 58.12 constructs the estimates of the unconditional probabilities of ossification.

Table 58.12 Estimates of Ossification Probabilities

PHT	TCPO	$\widehat{\eta}$	$\widehat{\pi}$
0	0	$-1.2194 + 0.9129 + 0.3295 + 0.6162 = 0.6392$	0.6546
60	0	$-1.2194 + 0.3295 = -0.8899$	0.2911
0	100	$-1.2194 + 0.9129 = -0.3065$	0.4240
60	100	-1.2194	0.2280

Morel and Neerchal (1997) considered a model in which the intralitter effects also depend on the treatments. This model is fit with the HPFMM procedure with the following statements:

```
proc hpfmm data=ossi;
   class pht tcpo;
   model y/m = pht tcpo pht*tcpo / dist=binomcluster;
   probmodel   pht tcpo pht*tcpo;
run;
```

The –2 log likelihood of this model is much reduced compared to the previous model with constant conditional probability (compare 287.8 in Output 58.1.4 with 305.1 in Output 58.1.2). The likelihood-ratio statistic of 17.3 is significant, $\Pr(\chi^2_3 > 17.3 = 0.0006)$. Varying the conditional probabilities by treatment improved the model fit significantly.

Output 58.1.4 Fit Statistics and Parameter Estimates in Binomial Cluster Model

The HPFMM Procedure

Fit Statistics	
-2 Log Likelihood	287.8
AIC (Smaller is Better)	303.8
AICC (Smaller is Better)	305.8
BIC (Smaller is Better)	323.0
Pearson Statistic	85.5998
Effective Parameters	8
Effective Components	2

Parameter Estimates for Binomial Cluster Model

Component	Effect	pht	tcpo	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept			1.8213	0.5889	3.09	0.0020
1	pht	0		-1.4962	0.6630	-2.26	0.0240
1	pht	60		0	.	.	.
1	tcpo		0	-3.1828	1.1261	-2.83	0.0047
1	tcpo		100	0	.	.	.
1	pht*tcpo	0	0	3.3736	1.1953	2.82	0.0048
1	pht*tcpo	0	100	0	.	.	.
1	pht*tcpo	60	0	0	.	.	.
1	pht*tcpo	60	100	0	.	.	.

Parameter Estimates for Mixing Probabilities

Component	Effect	pht	tcpo	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept			-0.7394	0.5395	-1.37	0.1705
1	pht	0		0.4351	0.6203	0.70	0.4830
1	pht	60		0	.	.	.
1	tcpo		0	-0.5342	0.5893	-0.91	0.3646
1	tcpo		100	0	.	.	.
1	pht*tcpo	0	0	1.4055	0.7080	1.99	0.0471
1	pht*tcpo	0	100	0	.	.	.
1	pht*tcpo	60	0	0	.	.	.
1	pht*tcpo	60	100	0	.	.	.

Table 58.13 computes the conditional probabilities in the four treatment groups. Recall that the previous model estimated a constant clustering probability of 0.5831.

Table 58.13 Estimates of Clustering Probabilities

PHT	TCPO	$\widehat{\eta}$	$\widehat{\mu}$
0	0	$1.8213 - 1.4962 - 3.1828 + 3.3736 = 0.5159$	0.6262
60	0	$1.8213 - 3.1828 = -1.3615$	0.2040
0	100	$1.8213 - 1.4962 = 0.3251$	0.5806
60	100	1.8213	0.8607

The presence of phenytoin alone reduces the probability of response clustering within the litter. The presence of trichloropropene oxide alone does not have a strong effect on the clustering. The simultaneous presence of both agents substantially increases the probability of clustering.

The following statements fit the binomial cluster model in the parameterization of Morel and Neerchal (1997).

```
proc hpfmm data=ossi;
   model y/m = x1-x3 / dist=binomcluster;
   probmodel   x1-x3;
run;
```

The model fit is the same as in the previous model (compare the "Fit Statistics" tables in Output 58.1.5 and Output 58.1.4). The parameter estimates change due to the reparameterization of the treatment effects and match the results in Table III of Morel and Neerchal (1997).

Output 58.1.5 Fit Statistics and Estimates (Morel and Neerchal Parameterization)

The HPFMM Procedure

Fit Statistics	
-2 Log Likelihood	287.8
AIC (Smaller is Better)	303.8
AICC (Smaller is Better)	305.8
BIC (Smaller is Better)	323.0
Pearson Statistic	85.5999
Effective Parameters	8
Effective Components	2

Parameter Estimates for Binomial Cluster Model

Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	0.5159	0.2603	1.98	0.0475
1	x1	-0.1908	0.4006	-0.48	0.6339
1	x2	-1.8774	0.9946	-1.89	0.0591
1	x3	3.3736	1.1953	2.82	0.0048

Parameter Estimates for Mixing Probabilities

Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	0.5669	0.2455	2.31	0.0209
1	x1	-0.8712	0.3924	-2.22	0.0264
1	x2	-1.8405	0.3413	-5.39	<.0001
1	x3	1.4055	0.7080	1.99	0.0471

The following sets of statements fit the binomial and beta-binomial models, respectively, as single-component mixtures in the parameterization akin to the first binomial cluster model. Note that the model effects that affect the underlying Bernoulli success probabilities are specified in the MODEL statement, in contrast to the binomial cluster model.

```
proc hpfmm data=ossi;
   model y/m = x1-x3 / dist=binomial;
run;
```

```
proc hpfmm data=ossi;
   model y/m = x1-x3 / dist=betabinomial;
run;
```

The Pearson statistic for the beta-binomial model (Output 58.1.6) indicates a much better fit compared to the single-component binomial model (Output 58.1.7). This is not surprising since these data are obviously overdispersed relative to a binomial model because the Bernoulli outcomes are not independent. The difference between the binomial cluster and the beta-binomial model lies in the mechanism by which the correlations are induced:

- a mixing mechanism in the beta-binomial model that leads to a common shared random effect among all offspring in a cluster
- a mixture specification in the binomial cluster model that divides the offspring in a litter into identical and independent responders

Output 58.1.6 Fit Statistics in Binomial Model

The HPFMM Procedure

Fit Statistics	
-2 Log Likelihood	401.8
AIC (Smaller is Better)	409.8
AICC (Smaller is Better)	410.3
BIC (Smaller is Better)	419.4
Pearson Statistic	252.1

Output 58.1.7 Fit Statistics in Beta-Binomial Model

The HPFMM Procedure

Fit Statistics	
-2 Log Likelihood	306.6
AIC (Smaller is Better)	316.6
AICC (Smaller is Better)	317.4
BIC (Smaller is Better)	328.5
Pearson Statistic	87.5379

Example 58.2: The Usefulness of Custom Starting Values: When Do Cows Eat?

(View the complete code for this example (hpfmmex2.sas) in the example repository.)

This example with a mixture of normal and Weibull distributions illustrates the benefits of specifying starting values for some of the components.

The data for this example were generously provided by Dr. Luciano A. Gonzalez of the Lethbridge Research Center of Agriculture and Agri-Food Canada and his collaborator, Dr. Bert Tolkamp, from the Scottish Agricultural College.

The outcome variable of interest is the logarithm of a time interval between consecutive visits by cattle to feeders. The intervals fall into three categories:

- short breaks within meals—such as when an animal stops eating for a moment and resumes shortly thereafter
- somewhat longer breaks when eating is interrupted to go have a drink of water
- long breaks between meals

Modeling such time interval data is important to understand the feeding behavior and biology of the animals and to derive other biological parameters such as the probability of an animal to stop eating after it has consumed a certain amount of a given food. Because there are three distinct biological categories, data of this nature are frequently modeled as three-component mixtures. The point at which the second and third components cross over is used to separate feeding events into meals.

The original data set comprises 141,414 observations of log feeding intervals. For the purpose of presentation in this document, where space is limited, the data have been rounded to precision 0.05 and grouped by frequency. The following DATA step displays the modified data used in this example. A comparison with the raw data and the results obtained in a full analysis of the original data show that the grouping does not alter the presentation or conclusions in a way that matters for the purpose of this example.

```
data cattle;
   input LogInt Count @@;
   datalines;
 0.70   195  1.10   233  1.40   355  1.60   563
 1.80   822  1.95   926  2.10  1018  2.20  1712
 2.30  3190  2.40  2212  2.50  1692  2.55  1558
 2.65  1622  2.70  1637  2.75  1568  2.85  1599
 2.90  1575  2.95  1526  3.00  1537  3.05  1561
 3.10  1555  3.15  1427  3.20  2852  3.25  1396
 3.30  1343  3.35  2473  3.40  1310  3.45  2453
 3.50  1168  3.55  2300  3.60  2174  3.65  2050
 3.70  1926  3.75  1849  3.80  1687  3.85  2416
 3.90  1449  3.95  2095  4.00  1278  4.05  1864
 4.10  1672  4.15  2104  4.20  1443  4.25  1341
 4.30  1685  4.35  1445  4.40  1369  4.45  1284
 4.50  1523  4.55  1367  4.60  1027  4.65  1491
 4.70  1057  4.75  1155  4.80  1095  4.85  1019
```

```
  4.90  1158  4.95  1088  5.00  1075  5.05   912
  5.10  1073  5.15   803  5.20   924  5.25   916
  5.30   784  5.35   751  5.40   766  5.45   833
  5.50   748  5.55   725  5.60   674  5.65   690
  5.70   659  5.75   695  5.80   529  5.85   639
  5.90   580  5.95   557  6.00   524  6.05   473
  6.10   538  6.15   444  6.20   456  6.25   453
  6.30   374  6.35   406  6.40   409  6.45   371
  6.50   320  6.55   334  6.60   353  6.65   305
  6.70   302  6.75   301  6.80   263  6.85   218
  6.90   255  6.95   240  7.00   219  7.05   202
  7.10   192  7.15   180  7.20   162  7.25   126
  7.30   148  7.35   173  7.40   142  7.45   163
  7.50   152  7.55   149  7.60   139  7.65   161
  7.70   174  7.75   179  7.80   188  7.85   239
  7.90   225  7.95   213  8.00   235  8.05   256
  8.10   272  8.15   290  8.20   320  8.25   355
  8.30   307  8.35   311  8.40   317  8.45   335
  8.50   369  8.55   365  8.60   365  8.65   396
  8.70   419  8.75   467  8.80   468  8.85   515
  8.90   558  8.95   623  9.00   712  9.05   716
  9.10   829  9.15   803  9.20   834  9.25   856
  9.30   838  9.35   842  9.40   826  9.45   834
  9.50   798  9.55   801  9.60   780  9.65   849
  9.70   779  9.75   737  9.80   683  9.85   686
  9.90   626  9.95   582 10.00   522 10.05   450
 10.10   443 10.15   375 10.20   342 10.25   285
 10.30   254 10.35   231 10.40   195 10.45   186
 10.50   143 10.55   100 10.60    73 10.65    49
 10.70    28 10.75    36 10.80    16 10.85     9
 10.90     5 10.95     6 11.00     4 11.05     1
 11.15     1 11.25     4 11.30     2 11.35     5
 11.40     4 11.45     3 11.50     1
;
```

If you scan the columns for the Count variable in the DATA step, the prevalence of values between 2 and 5 units of LogInt is apparent, as is a long right tail. To explore these data graphically, the following statements produce a histogram of the data and a kernel density estimate of the density of the LogInt variable.

```
ods graphics on;
proc kde data=cattle;
   univar LogInt / bwm=4;
   freq count;
run;
```

Output 58.2.1 Histogram and Kernel Density for LogInt

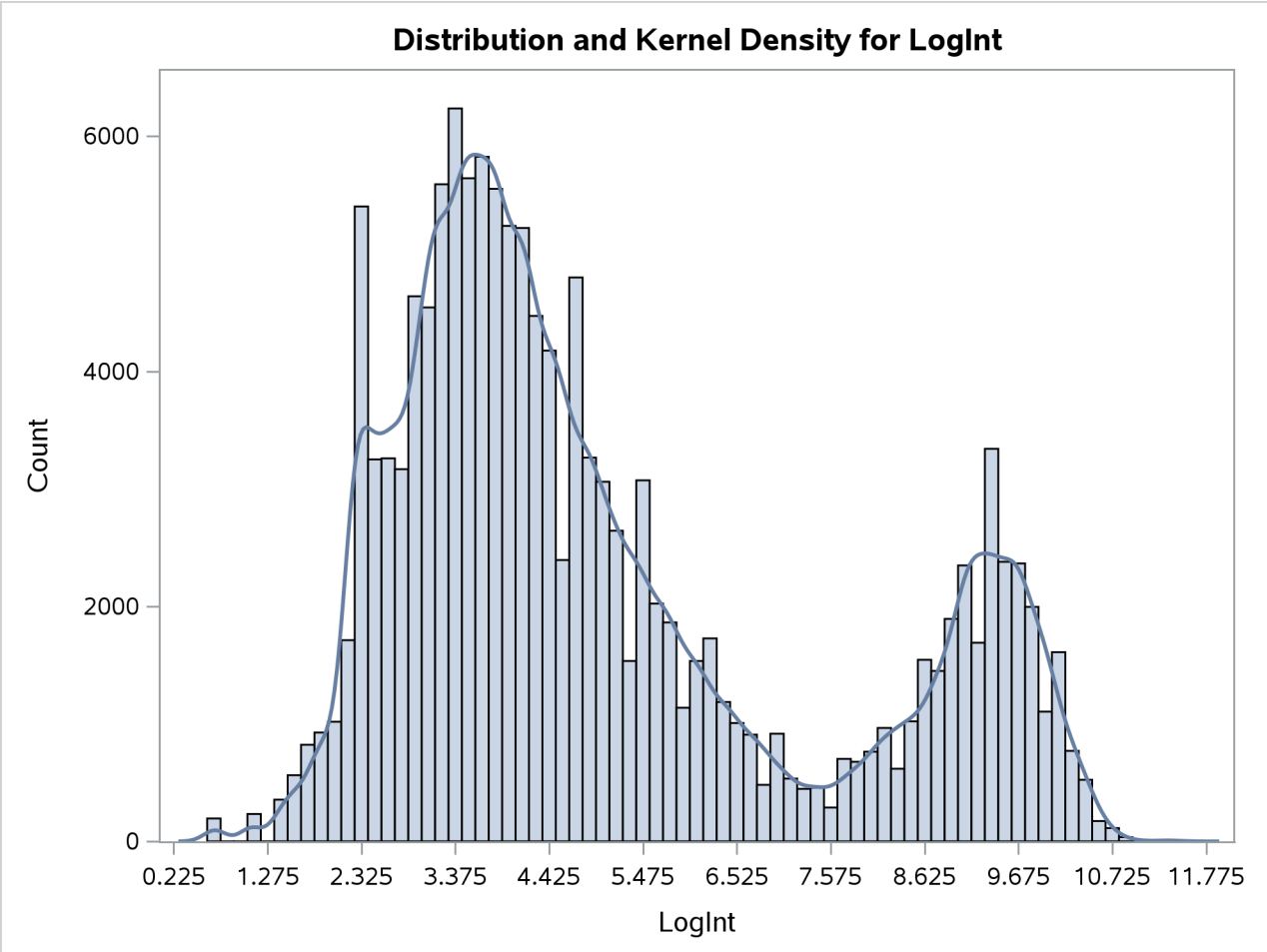

Two modes are clearly visible in Output 58.2.1. Given the biological background, one would expect that three components contribute to the mixture. The histogram would suggest either a two-component mixture with modes near 4 and 9, or a three-component mixture with modes near 3, 5, and 9.

Following Dr. Gonzalez' suggestion, the process is modeled as a three-component mixture of two normal distributions and a Weibull distribution. The Weibull distribution is chosen because it can have long left and right tails and it is popular in modeling data that relate to time intervals.

```
proc hpfmm data=cattle gconv=0;
   model LogInt = / dist=normal k=2 parms(3 1, 5 1);
   model        + / dist=weibull;
   freq count;
run;
```

The GCONV= convergence criterion is turned off in this PROC HPFMM run to avoid the early stoppage of the iterations when the relative gradient changes little between iterations. Turning the criterion off usually ensures that convergence is achieved with a small absolute gradient of the objective function. The PARMS option in the first MODEL statement provides starting values for the means and variances for the parameters of the normal distributions. The means for the two components are started at $\mu = 3$ and $\mu = 5$, respectively. Specifying starting values is generally not necessary. However, the choice of starting values can play an

important role in modeling finite mixture models; the importance of the choice of starting values in this example is discussed further below.

The "Model Information" table shows that the model is a three-component mixture and that the HPFMM procedure considers the estimation of a density to be the purpose of modeling. The procedure draws this conclusion from the absence of effects in the MODEL statements. There are 187 observations in the data set, but these actually represent 141,414 measurements (Output 58.2.2).

Output 58.2.2 Model Information and Number of Observations

The HPFMM Procedure

Model Information	
Data Set	WORK.CATTLE
Response Variable	LogInt
Frequency Variable	Count
Type of Model	Density Estimation
Components	3
Estimation Method	Maximum Likelihood

Number of Observations Read	187
Number of Observations Used	187
Sum of Frequencies Read	141414
Sum of Frequencies Used	141414

There are eight parameters in the optimization: the means and variances of the two normal distributions, the μ and ϕ parameter of the Weibull distribution, and the two mixing probabilities (Output 58.2.3). At the converged solution, the –2 log likelihood is 563,153 and all parameters and components are effective—that is, the model is not overspecified in the sense that components have collapsed during the model fitting. The Pearson statistic is close to the number of observations in the data set, indicating a good fit.

Output 58.2.3 Optimization Information and Fit Statistics

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	8
Mean Function Parameters	3
Scale Parameters	3
Mixing Prob Parameters	2
Lower Boundaries	3
Upper Boundaries	0

Fit Statistics	
-2 Log Likelihood	563153
AIC (Smaller is Better)	563169
AICC (Smaller is Better)	563169
BIC (Smaller is Better)	563248
Pearson Statistic	141458
Effective Parameters	8
Effective Components	3

Output 58.2.4 displays the parameter estimates for the three models and for the mixing probabilities. The order in which the "Parameter Estimates" tables appear in the output corresponds to the order in which the MODEL statements were specified.

Output 58.2.4 Optimization Information and Fit Statistics

Parameter Estimates for Normal Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	3.3415	0.01260	265.16	<.0001
2	Intercept	4.8940	0.05447	89.84	<.0001
1	Variance	0.6718	0.01287		
2	Variance	1.4497	0.05247		

Parameter Estimates for Weibull Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|	Inverse Linked Estimate
3	Intercept	2.2531	0.000506	4452.11	<.0001	9.5174
3	Scale	0.06848	0.000427			

Parameter Estimates for Mixing Probabilities

Component	Mixing Probability	Linked Scale GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.4545	0.8106	0.03409	23.78	<.0001
2	0.3435	0.5305	0.04640	11.43	<.0001
3	0.2021	0			

The estimated means of the two normal components are 3.3415 and 4.8940, respectively. Note that the means are displayed here as Intercept. The inverse linked estimate is not produced because the default link for the normal distribution is the identity link; hence the Estimate column represents the means of the component distributions. The parameter estimates in the Weibull model are $\widehat{\beta}_0 = 2.2531$, $\widehat{\phi} = 0.06848$, and $\widehat{\mu} = \exp\{\widehat{\beta}_0\} = 9.5174$. In the Weibull distribution, the μ parameter does not estimate the mean of the distribution, the maximum likelihood estimate of the distribution's mean is $\widehat{\mu}\Gamma(\widehat{\phi} + 1) = 9.1828$.

The estimated mixing probabilities are $\widehat{\pi}_1 = 0.4545$, $\widehat{\pi}_2 = 0.3435$, and $\widehat{\pi_3} = 0.2021$. In other words, the estimated distribution of log feeding intervals is a 45:35:20 mixture of an $\mathrm{N}(3.3415, 0.6718)$, a $\mathrm{N}(4.8940, 1.4497)$, and a $\mathrm{Weibull}(9.5174, 0.06848)$ distribution.

You can obtain a graphical display of the observed and estimated distribution of these data by enabling ODS Graphics. The PLOTS option in the PROC HPFMM statement modifies the default density plot by adding the densities of the mixture components:

```
ods select DensityPlot;
proc hpfmm data=cattle gconv=0;
   model LogInt = / dist=normal k=2 parms(3 1, 5 1);
   model        + / dist=weibull;
   freq count;
run;
```

Output 58.2.5 Observed and Estimated Densities in the Three-Component Model

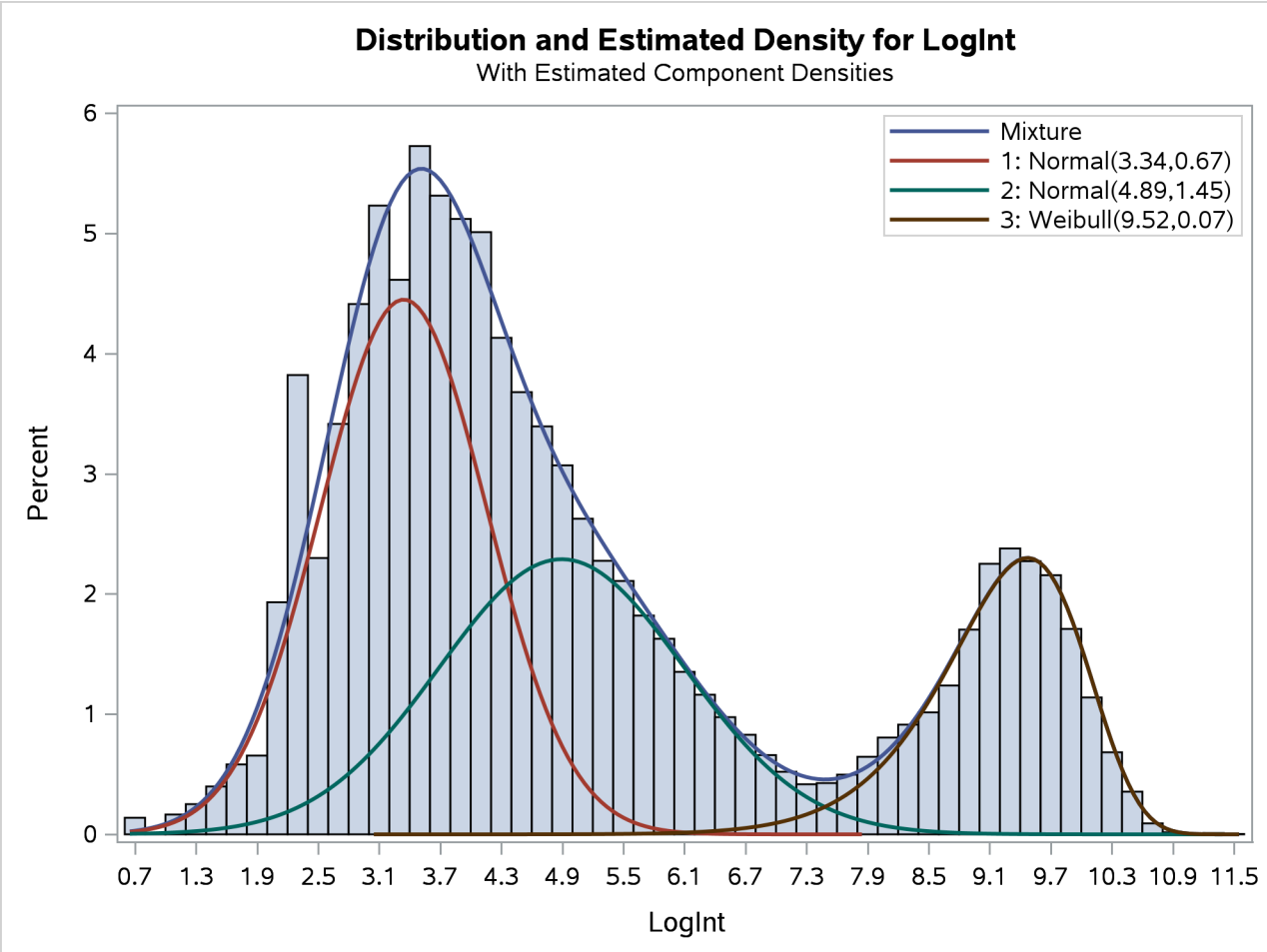

The estimated mixture density matches the histogram of the observed data closely (Output 58.2.5). The component densities are displayed in such a way that, at each point in the support of the LogInt variable, their sum combines to the overall mixture density. The three components in the mixtures are well separated.

The excellent quality of the fit is even more evident when the distributions are displayed cumulatively by adding the CUMULATIVE option in the DENSITY option (Output 58.2.6):

```
ods select DensityPlot;
proc hpfmm data=cattle plot=density(cumulative) gconv=0;
   model LogInt = / dist=normal k=2 parms(3 1, 5 1);
   model        + / dist=weibull;
   freq count;
run;
```

The component cumulative distribution functions are again scaled so that their sum produces the overall mixture cumulative distribution function. Because of this scaling, the percentage reached at the maximum value of LogInt corresponds to the mixing probabilities in Output 58.2.4.

Output 58.2.6 Observed and Estimated Cumulative Densities in the Three-Component Model

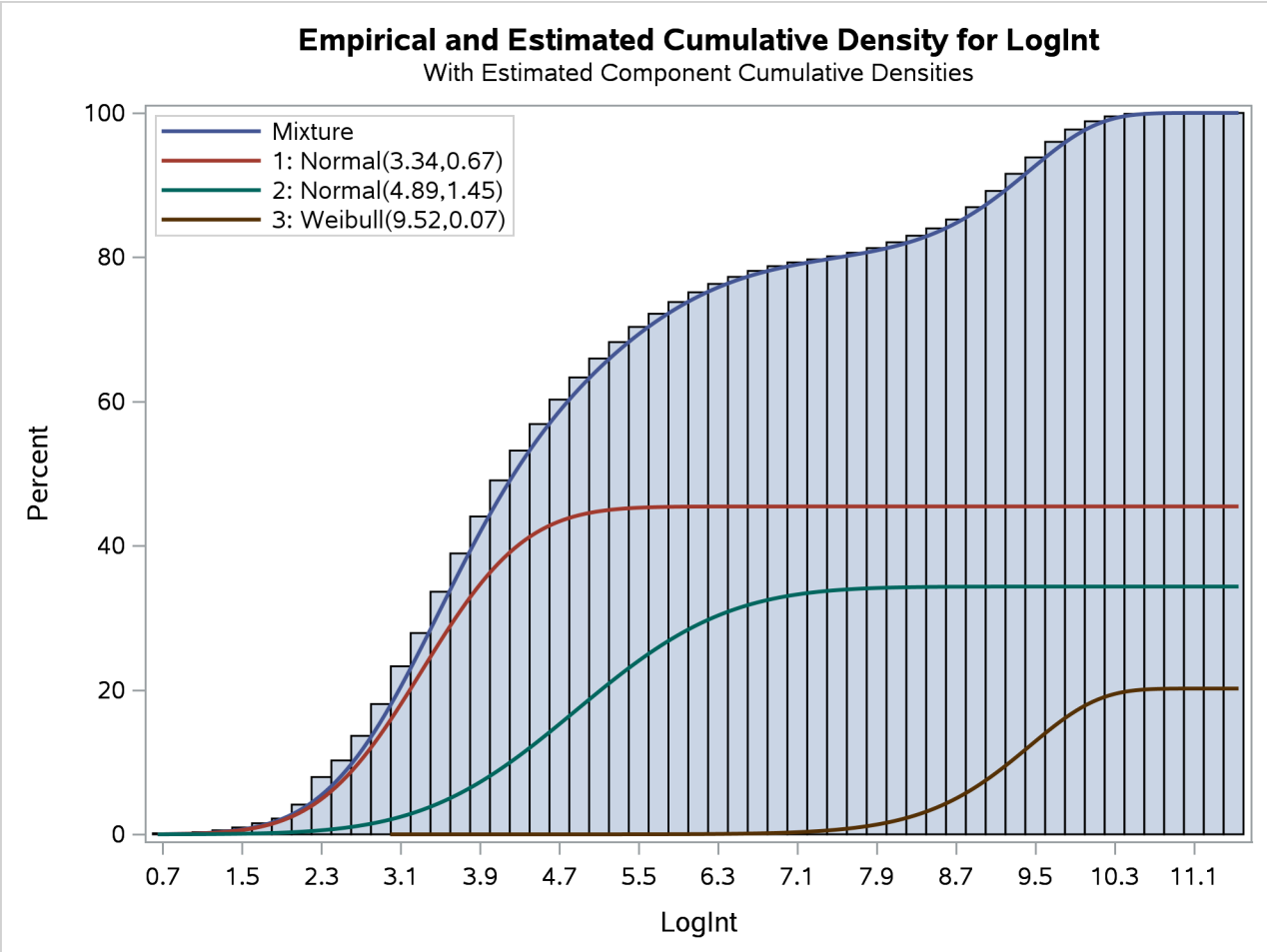

The importance of starting values for the parameter estimates was mentioned previously. Suppose that different starting values are selected for the three components (for example, the default starting values).

```
proc hpfmm data=cattle gconv=0;
   model LogInt = / dist=normal k=2;
   model        + / dist=weibull;
   freq count;
run;
ods graphics off;
```

The fit statistics and parameter estimates from this run are displayed in Output 58.2.7, and the density plot is shown in Output 58.2.8.

Output 58.2.7 Fit Statistics and Parameter Estimates

The HPFMM Procedure

Fit Statistics	
-2 Log Likelihood	564431
AIC (Smaller is Better)	564447
AICC (Smaller is Better)	564447
BIC (Smaller is Better)	564526
Pearson Statistic	141228
Effective Parameters	8
Effective Components	3

Parameter Estimates for Normal Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	4.9106	0.02604	188.56	<.0001
2	Intercept	9.2883	0.005031	1846.28	<.0001
1	Variance	1.7410	0.02753		
2	Variance	0.4158	0.005086		

Parameter Estimates for Weibull Model

Component	Parameter	Estimate	Standard Error	z Value	Pr > |z|	Inverse Linked Estimate
3	Intercept	1.2908	0.002790	462.71	<.0001	3.6358
3	Scale	0.2093	0.001311			

Parameter Estimates for Mixing Probabilities

		Linked Scale			
Component	Mixing Probability	GLogit(Prob)	Standard Error	z Value	Pr > |z|
1	0.3745	-0.1505	0.03678	-4.09	<.0001
2	0.1902	-0.8280	0.01922	-43.08	<.0001
3	0.4353	0			

All components are active; no collapsing of components occurred. However, a closer look at the "Parameter Estimates" tables in Output 58.2.7 shows an important difference from the tables in Output 58.2.4. The means of the two normal distributions are now 4.9106 and 9.2883. Previously, the means were 3.3415 and 4.8940. The "position" of the Weibull distribution has moved from right to left, and the third component is now modeled by a symmetric normal distribution (Output 58.2.8). The mixture probabilities have also changed—in particular, for the first and third component.

Output 58.2.8 Three-Component Model with Default Starting Values

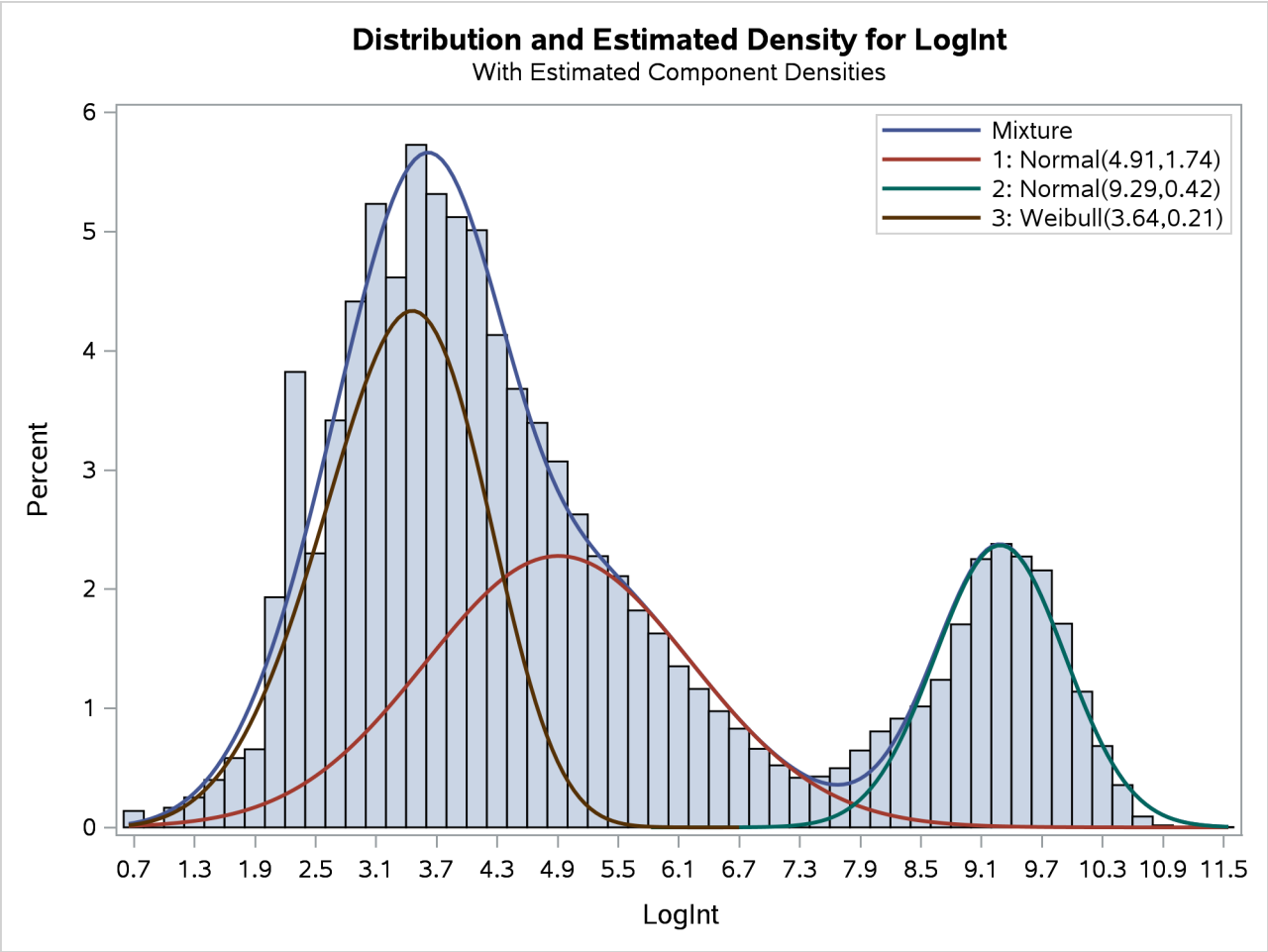

Such switching is not uncommon in mixture modeling. As judged by the information criteria, the model in which the Weibull distribution is the component with the smallest mean does not fit the data as well as the first model in which the specification of the starting values guided the optimization towards placing the normal distributions first. The converged solution found in the last HPFMM run represents a local minimum of the log-likelihood surface. There are other local minima—for example, when components are removed from the model, which is tantamount to estimating the associated mixture probabilities as zero.

Example 58.3: Enforcing Homogeneity Constraints: Count and Dispersion—It Is All Over!

(View the complete code for this example (hpfmmex3.sas) in the example repository.)

The following example demonstrates how you can use either the EQUATE= option in the MODEL statement or the RESTRICT statement to impose homogeneity constraints on chosen model effects.

The data for this example were presented by Margolin, Kaplan, and Zeiger (1981) and analyzed by various authors applying a number of techniques. The following DATA step shows the number of revertant salmonella colonies (variable num) at six levels of quinoline dosing (variable dose). There are three replicate plates at each dose of quinoline.

```
data assay;
   label dose = 'Dose of quinoline (microg/plate)'
         num  = 'Observed number of colonies';
   input dose @;
   logd = log(dose+10);
   do i=1 to 3; input num@; output; end;
   datalines;
   0  15 21 29
  10  16 18 21
  33  16 26 33
 100  27 41 60
 333  33 38 41
1000  20 27 42
;
```

The basic notion is that the data are overdispersed relative to a Poisson distribution in which the logarithm of the mean count is modeled as a linear regression in dose (in $\mu g/\text{plate}$) and in the derived variable $\log\{\text{dose} + 10\}$ (Lawless 1987). The log of the expected count of revertants is thus

$$\beta_0 + \beta_1 \text{dose} + \beta_2 \log\{\text{dose} + 10\}$$

The following statements fit a standard Poisson regression model to these data:

```
proc hpfmm data=assay;
   model num = dose logd / dist=Poisson;
run;
```

The Pearson statistic for this model is rather large compared to the number of degrees of freedom ($18-3 = 15$). The ratio $46.2707/15 = 3.08$ indicates an overdispersion problem in the Poisson model (Output 58.3.1).

Output 58.3.1 Result of Fitting Poisson Regression Models

The HPFMM Procedure

Number of Observations Read	18
Number of Observations Used	18

Output 58.3.1 *continued*

Fit Statistics	
-2 Log Likelihood	136.3
AIC (Smaller is Better)	142.3
AICC (Smaller is Better)	144.0
BIC (Smaller is Better)	144.9
Pearson Statistic	46.2707

Parameter Estimates for Poisson Model				
Effect	Estimate	Standard Error	z Value	Pr > |z|
Intercept	2.1728	0.2184	9.95	<.0001
dose	-0.00101	0.000245	-4.13	<.0001
logd	0.3198	0.05700	5.61	<.0001

Breslow (1984) accounts for overdispersion by including a random effect in the predictor for the log rate and applying a quasi-likelihood technique to estimate the parameters. Wang et al. (1996) examine these data using mixtures of Poisson regression models. They fit several two- and three-component Poisson regression mixtures. Examining the log likelihoods, AIC, and BIC criteria, they eventually settle on a two-component model in which the intercepts vary by category and the regression coefficients are the same. This mixture model can be written as

$$
\begin{aligned}
f(y) =& \pi \frac{1}{y!}\lambda_1^y \exp\{-\lambda_1\} + (1-\pi)\frac{1}{y!}\lambda_2^y \exp\{-\lambda_2\} \\
\lambda_1 =& \exp\{\beta_{01} + \beta_1 \text{dose} + \beta_2 \log[\text{dose} + 10]\} \\
\lambda_2 =& \exp\{\beta_{02} + \beta_1 \text{dose} + \beta_2 \log[\text{dose} + 10]\}
\end{aligned}
$$

This model is fit with the HPFMM procedure with the following statements:

```
proc hpfmm data=assay;
   model num = dose logd / dist=Poisson k=2
                           equate=effects(dose logd);
run;
```

The EQUATE= option in the MODEL statement places constraints on the optimization and makes the coefficients for dose and logd homogeneous across components in the model. Output 58.3.2 displays the "Fit Statistics" and parameter estimates in the mixture. The Pearson statistic is drastically reduced compared to the Poisson regression model in Output 58.3.1. With $18 - 5 = 13$ degrees of freedom, the ratio of the Pearson and the degrees of freedom is now $16.1573/13 = 1.2429$. Note that the effective number of parameters was used to compute the degrees of freedom, not the total number of parameters, because of the equality constraints.

Output 58.3.2 Result for Two-Component Poisson Regression Mixture

The HPFMM Procedure

Fit Statistics	
-2 Log Likelihood	121.8
AIC (Smaller is Better)	131.8
AICC (Smaller is Better)	136.8
BIC (Smaller is Better)	136.3
Pearson Statistic	16.1573
Effective Parameters	5
Effective Components	2

Parameter Estimates for Poisson Model

Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	1.9097	0.2654	7.20	<.0001
1	dose	-0.00126	0.000273	-4.62	<.0001
1	logd	0.3639	0.06602	5.51	<.0001
2	Intercept	2.4770	0.2731	9.07	<.0001
2	dose	-0.00126	0.000273	-4.62	<.0001
2	logd	0.3639	0.06602	5.51	<.0001

Parameter Estimates for Mixing Probabilities

		Linked Scale			
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.8173	1.4984	0.6875	2.18	0.0293
2	0.1827	-1.4984			

You could also have used RESTRICT statements to impose the homogeneity constraints on the model fit, as shown in the following statements:

```
proc hpfmm data=assay;
   model num = dose logd / dist=Poisson k=2;
   restrict 'common dose' dose 1, dose -1;
   restrict 'common logd' logd 1, logd -1;
run;
```

The first RESTRICT statement equates the coefficients for the dose variable in the two components, and the second RESTRICT statement accomplishes the same for the coefficients of the logd variable. If the right-hand side of a restriction is not specified, PROC HPFMM defaults to equating the left-hand side of the restriction to zero. The "Linear Constraints" table in Output 58.3.3 shows that both linear equality constraints are active. The parameter estimates match the previous HPFMM run.

Output 58.3.3 Result for Two-Component Mixture with RESTRICT Statements

The HPFMM Procedure

Linear Constraints at Solution					
Label	k = 1	k = 2			Constraint Active
common dose	dose	- dose	=	0	Yes
common logd	logd	- logd	=	0	Yes

Parameter Estimates for Poisson Model					
Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	**Intercept**	1.9097	0.2654	7.20	<.0001
1	**dose**	-0.00126	0.000273	-4.62	<.0001
1	**logd**	0.3639	0.06602	5.51	<.0001
2	**Intercept**	2.4770	0.2731	9.07	<.0001
2	**dose**	-0.00126	0.000273	-4.62	<.0001
2	**logd**	0.3639	0.06602	5.51	<.0001

Parameter Estimates for Mixing Probabilities					
			Linked Scale		
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.8173	1.4984	0.6875	2.18	0.0293
2	0.1827	-1.4984			

Wang et al. (1996) note that observation 12 with a revertant colony count of 60 is comparably high. The following statements remove the observation from the analysis and fit their selected model:

```
proc hpfmm data=assay(where=(num ne 60));
   model num = dose logd / dist=Poisson k=2
                           equate=effects(dose logd);
run;
```

Output 58.3.4 Result for Two-Component Model without Outlier

The HPFMM Procedure

Fit Statistics	
-2 Log Likelihood	111.5
AIC (Smaller is Better)	121.5
AICC (Smaller is Better)	126.9
BIC (Smaller is Better)	125.6
Pearson Statistic	16.5987
Effective Parameters	5
Effective Components	2

Output 58.3.4 *continued*

Parameter Estimates for Poisson Model					
Component	Effect	Estimate	Standard Error	z Value	Pr > |z|
1	Intercept	2.2272	0.3022	7.37	<.0001
1	dose	-0.00065	0.000445	-1.46	0.1440
1	logd	0.2432	0.1045	2.33	0.0199
2	Intercept	2.5477	0.3331	7.65	<.0001
2	dose	-0.00065	0.000445	-1.46	0.1440
2	logd	0.2432	0.1045	2.33	0.0199

Parameter Estimates for Mixing Probabilities					
		Linked Scale			
Component	Mixing Probability	Logit(Prob)	Standard Error	z Value	Pr > |z|
1	0.5777	0.3134	1.7261	0.18	0.8559
2	0.4223	-0.3134			

The ratio of Pearson Statistic over degrees of freedom (12) is only slightly worse than in the previous model; the loss of 5% of the observations carries a price (Output 58.3.4). The parameter estimates for the two intercepts are now fairly close. If the intercepts were identical, then the two-component model would collapse to the Poisson regression model:

```
proc hpfmm data=assay(where=(num ne 60));
   model num = dose logd / dist=Poisson;
run;
```

Output 58.3.5 Result of Fitting Poisson Regression Model without Outlier

The HPFMM Procedure

Number of Observations Read	17
Number of Observations Used	17

Fit Statistics	
-2 Log Likelihood	114.1
AIC (Smaller is Better)	120.1
AICC (Smaller is Better)	121.9
BIC (Smaller is Better)	122.5
Pearson Statistic	27.8008

Parameter Estimates for Poisson Model				
Effect	Estimate	Standard Error	z Value	Pr > |z|
Intercept	2.3164	0.2244	10.32	<.0001
dose	-0.00072	0.000258	-2.78	0.0055
logd	0.2603	0.05996	4.34	<.0001

Compared to the same model applied to the full data, the Pearson statistic is much reduced (compare 46.2707 in Output 58.3.1 to 27.8008 in Output 58.3.5). The outlier—or *over*count, if you will—induces at least some

of the *over*dispersion.

References

Aldrich, J. (1997). "R. A. Fisher and the Making of Maximum Likelihood, 1912–1922." *Statistical Science* 12:162–176.

Breslow, N. E. (1984). "Extra-Poisson Variation in Log-Linear Models." *Journal of the Royal Statistical Society, Series C* 33:38–44.

Cameron, A. C., and Trivedi, P. K. (1998). *Regression Analysis of Count Data*. Cambridge: Cambridge University Press.

Celeux, G., Forbes, F., Robert, C. P., and Titterington, D. M. (2006). "Deviance Information Criteria for Missing Data Models." *Bayesian Analysis* 1:651–674.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society, Series B* 39:1–38.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981). "An Adaptive Nonlinear Least-Squares Algorithm." *ACM Transactions on Mathematical Software* 7:348–368.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Eskow, E., and Schnabel, R. B. (1991). "Algorithm 695: Software for a New Modified Cholesky Factorization." *ACM Transactions on Mathematical Software* 17:306–312.

Everitt, B. S., and Hand, D. J. (1981). *Finite Mixture Distributions*. London: Chapman & Hall.

Ferrari, S. L. P., and Cribari-Neto, F. (2004). "Beta Regression for Modelling Rates and Proportions." *Journal of Applied Statistics* 31:799–815.

Fisher, R. A. (1921). "On the 'Probable Error' of a Coefficient of Correlation Deduced from a Small Sample." *Metron* 1:3–32.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Frühwirth-Schnatter, S. (2006). *Finite Mixture and Markov Switching Models*. New York: Springer.

Gamerman, D. (1997). "Sampling from the Posterior Distribution in Generalized Linear Models." *Statistics and Computing* 7:57–68.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Geweke, J. (1992). "Evaluating the Accuracy of Sampling-Based Approaches to Calculating Posterior Moments." In *Bayesian Statistics*, vol. 4, edited by J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, 169–193. Oxford: Clarendon Press.

Griffiths, D. A. (1973). "Maximum Likelihood Estimation for the Beta-Binomial Distribution and an Application to the Household Distribution of the Total Number of Cases of a Disease." *Biometrics* 29:637–648.

Haseman, J. K., and Kupper, L. L. (1979). "Analysis of Dichotomous Response Data from Certain Toxicological Experiments." *Biometrics* 35:281–293.

Joe, H., and Zhu, R. (2005). "Generalized Poisson Distribution: The Property of Mixture of Poisson and Comparison with Negative Binomial Distribution." *Biometrical Journal* 47:219–229.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998). "Markov Chain Monte Carlo in Practice: A Roundtable Discussion." *American Statistician* 52:93–100.

Lawless, J. F. (1987). "Negative Binomial and Mixed Poisson Regression." *Canadian Journal of Statistics* 15:209–225.

Margolin, B. H., Kaplan, N. L., and Zeiger, E. (1981). "Statistical Analysis of the Ames Salmonella Microsome Test." *Proceedings of the National Academy of Sciences* 76:3779–3783.

McLachlan, G. J., and Peel, D. (2000). *Finite Mixture Models*. New York: John Wiley & Sons.

Moré, J. J. (1978). "The Levenberg-Marquardt Algorithm: Implementation and Theory." In *Lecture Notes in Mathematics*, vol. 30, edited by G. A. Watson, 105–116. Berlin: Springer-Verlag.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Morel, J. G., and Nagaraj, N. K. (1993). "A Finite Mixture Distribution for Modelling Multinomial Extra Variation." *Biometrika* 80:363–371.

Morel, J. G., and Neerchal, N. K. (1997). "Clustered Binary Logistic Regression in Teratology Data Using a Finite Mixture Distribution." *Statistics in Medicine* 16:2843–2853.

Neerchal, N. K., and Morel, J. G. (1998). "Large Cluster Results for Two Parametric Multinomial Extra Variation Models." *Journal of the American Statistical Association* 93:1078–1087.

Pearson, K. (1915). "On Certain Types of Compound Frequency Distributions in Which the Components Can Be Individually Described by Binomial Series." *Biometrika* 11:139–144.

Raftery, A. E. (1996). "Hypothesis Testing and Model Selection." In *Markov Chain Monte Carlo in Practice*, edited by W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, 163–188. London: Chapman & Hall.

Richardson, S. (2002). "Discussion of Spiegelhalter et al." *Journal of the Royal Statistical Society, Series B* 64:631.

Roeder, K. (1990). "Density Estimation with Confidence Sets Exemplified by Superclusters and Voids in the Galaxies." *Journal of the American Statistical Association* 85:617–624.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and Van der Linde, A. (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society, Series B* 64:583–616. With discussion.

Titterington, D. M., Smith, A. F. M., and Makov, U. E. (1985). *Statistical Analysis of Finite Mixture Distributions*. New York: John Wiley & Sons.

Viallefont, V., Richardson, S., and Greene, P. J. (2002). "Bayesian Analysis of Poisson Mixtures." *Journal of Nonparametric Statistics* 14:181–202.

Wang, P., Puterman, M. L., Cockburn, I., and Le, N. (1996). "Mixed Poisson Regression Models with Covariate Dependent Rates." *Biometrics* 52:381–400.

Williams, D. A. (1975). "The Analysis of Binary Responses from Toxicological Experiments Involving Reproduction and Teratogenicity." *Biometrics* 31:949–952.

Chapter 59
The HPGENSELECT Procedure

Contents

Overview: HPGENSELECT Procedure **4728**
PROC HPGENSELECT Features 4728
PROC HPGENSELECT Contrasted with PROC GENMOD 4729
Getting Started: HPGENSELECT Procedure **4729**
Syntax: HPGENSELECT Procedure **4735**
PROC HPGENSELECT Statement 4735
BY Statement 4742
CLASS Statement 4742
CODE Statement 4743
FREQ Statement 4743
ID Statement 4744
MODEL Statement 4744
OUTPUT Statement 4750
PARTITION Statement 4753
PERFORMANCE Statement 4754
RESTRICT Statement 4754
SELECTION Statement 4756
WEIGHT Statement 4758
ZEROMODEL Statement 4758
Details: HPGENSELECT Procedure **4760**
Missing Values 4760
Exponential Family Distributions 4760
Response Distributions 4761
Response Probability Distribution Functions 4762
Log-Likelihood Functions 4765
The LASSO Method of Model Selection 4770
Using Validation and Test Data 4772
Computational Method: Multithreading 4773
Choosing an Optimization Algorithm 4774
First- or Second-Order Algorithms 4774
Algorithm Descriptions 4775
Displayed Output 4777
ODS Table Names 4781
Examples: HPGENSELECT Procedure **4783**
Example 59.1: Model Selection 4783
Example 59.2: Modeling Binomial Data 4786

Example 59.3: Tweedie Model 4790
Example 59.4: Model Selection by the LASSO Method 4792
References **4796**

Overview: HPGENSELECT Procedure

The HPGENSELECT procedure is a high-performance procedure that provides model fitting and model building for generalized linear models. It fits models for standard distributions in the exponential family, such as the normal, Poisson, and Tweedie distributions. In addition, PROC HPGENSELECT fits multinomial models for ordinal and nominal responses, and it fits zero-inflated Poisson and negative binomial models for count data. For all these models, the HPGENSELECT procedure provides forward, backward, and stepwise variable selection.

PROC HPGENSELECT Features

The HPGENSELECT procedure does the following:

- estimates the parameters of a generalized linear regression model by using maximum likelihood techniques
- provides model-building syntax in the CLASS statement and the effect-based MODEL statement, which are familiar from SAS/STAT procedures (in particular, the GLM, GENMOD, LOGISTIC, GLIMMIX, and MIXED procedures)
- enables you to split classification effects into individual components by using the SPLIT option in the CLASS statement
- permits any degree of interaction effects that involve classification and continuous variables
- provides multiple link functions
- provides models for zero-inflated count data
- provides cumulative link modeling for ordinal data and generalized logit modeling for unordered multinomial data
- enables model building (variable selection) through the SELECTION statement
- provides a WEIGHT statement for weighted analysis
- provides a FREQ statement for grouped analysis
- provides an OUTPUT statement to produce a data set that has predicted values and other observation-wise statistics

Because the HPGENSELECT procedure is a high-performance analytical procedure, it also exploits all the available cores and concurrent threads.

PROC HPGENSELECT Contrasted with PROC GENMOD

This section contrasts the HPGENSELECT procedure with the GENMOD procedure in SAS/STAT software.

The CLASS statement in the HPGENSELECT procedure permits two parameterizations: GLM parameterization and a reference parameterization. In contrast to the LOGISTIC, GENMOD, and other procedures that permit multiple parameterizations, the HPGENSELECT procedure does not mix parameterizations across the variables in the CLASS statement. In other words, all classification variables have the same parameterization, and this parameterization is either GLM parameterization or reference parameterization. The CLASS statement also enables you to split an effect that involves a classification variable into multiple effects that correspond to individual levels of the classification variable.

The default optimization technique used by the HPGENSELECT procedure is a modification of the Newton-Raphson algorithm with a ridged Hessian. You can choose different optimization techniques (including first-order methods that do not require a crossproducts matrix or Hessian) by specifying the TECHNIQUE= option in the PROC HPGENSELECT statement.

As in the GENMOD procedure, the default parameterization of CLASS variables in the HPGENSELECT procedure is GLM parameterization. You can change the parameterization by specifying the PARAM= option in the CLASS statement.

The GENMOD procedure offers a wide variety of postfitting analyses, such as contrasts, estimates, tests of model effects, and least squares means. The HPGENSELECT procedure is limited in postfitting functionality because it is primarily designed for large-data tasks, such as predictive model building, model fitting, and scoring.

By default, the HPGENSELECT procedure performs computations in multiple threads. The GENMOD procedure executes in a single thread.

Getting Started: HPGENSELECT Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpgengs1.sas.)

This example illustrates how you can use PROC HPGENSELECT to perform Poisson regression for count data. The following DATA step contains 100 observations for a count response variable (Y), a continuous variable (Total) to be used in a later analysis, and five categorical variables (C1–C5), each of which has four numerical levels:

```
data getStarted;
  input C1-C5 Y Total;
  datalines;
0 3 1 1 3 2 28.361
2 3 0 3 1 2 39.831
1 3 2 2 2 1 17.133
1 2 0 0 3 2 12.769
0 2 1 0 1 1 29.464
0 2 1 0 2 1 4.152
1 2 1 0 1 0 0.000
```

```
0 2 1 1 2 1 20.199
1 2 0 0 1 0 0.000
0 1 1 3 3 2 53.376
2 2 2 2 1 1 31.923
0 3 2 0 3 2 37.987
2 2 2 0 0 1 1.082
0 2 0 2 0 1 6.323
1 3 0 0 0 0 0.000
1 2 1 2 3 2 4.217
0 1 2 3 1 1 26.084
1 1 0 0 1 0 0.000
1 3 2 2 2 0 0.000
2 1 3 1 1 2 52.640
1 3 0 1 2 1 3.257
2 0 2 3 0 5 88.066
2 2 2 1 0 1 15.196
3 1 3 1 0 1 11.955
3 1 3 1 2 3 91.790
3 1 1 2 3 7 232.417
3 1 1 1 0 1 2.124
3 1 0 0 0 2 32.762
3 1 2 3 0 1 25.415
2 2 0 1 2 1 42.753
3 3 2 2 3 1 23.854
2 0 0 2 3 2 49.438
1 0 0 2 3 4 105.449
0 0 2 3 0 6 101.536
0 3 1 0 0 0 0.000
3 0 1 0 1 1 5.937
2 0 0 0 3 2 53.952
1 0 1 0 3 2 23.686
1 1 3 1 1 1 0.287
2 1 3 0 3 7 281.551
1 3 2 1 1 0 0.000
2 1 0 0 1 0 0.000
0 0 1 1 2 3 93.009
0 1 0 1 0 2 25.055
1 2 2 2 3 1 1.691
0 3 2 3 1 1 10.719
3 3 0 3 3 1 19.279
2 0 0 2 1 2 40.802
2 2 3 0 3 3 72.924
0 2 0 3 0 1 10.216
3 0 1 2 2 2 87.773
2 1 2 3 1 0 0.000
3 2 0 3 1 0 0.000
3 0 3 0 0 2 62.016
1 3 2 2 1 3 36.355
2 3 2 0 3 1 23.190
1 0 1 2 1 1 11.784
2 1 2 2 2 5 204.527
3 0 1 1 2 5 115.937
0 1 1 3 2 1 44.028
2 2 1 3 1 4 52.247
```

```
1 1 0 0 1 1 17.621
3 3 1 2 1 2 10.706
2 2 0 2 3 3 81.506
0 1 0 0 2 2 81.835
0 1 2 0 1 2 20.647
3 2 2 2 0 1 3.110
2 2 3 0 0 1 13.679
1 2 2 3 2 1 6.486
3 3 2 2 1 2 30.025
0 0 3 1 3 6 202.172
3 2 3 1 2 3 44.221
0 3 0 0 0 1 27.645
3 3 3 0 3 2 22.470
2 3 2 0 2 0 0.000
1 3 0 2 0 1 1.628
1 3 1 0 2 0 0.000
3 2 3 3 0 1 20.684
3 1 0 2 0 4 108.000
0 1 2 2 1 1 4.615
0 2 3 2 2 1 12.461
0 3 2 0 1 3 53.798
2 1 1 2 0 1 36.320
1 0 3 0 0 0 0.000
0 0 3 2 0 1 19.902
0 2 3 1 0 0 0.000
2 2 2 1 3 2 31.815
3 3 3 0 0 0 0.000
2 2 1 3 3 2 17.915
0 2 3 2 3 2 69.315
1 3 1 2 1 0 0.000
3 0 1 1 1 4 94.050
2 1 1 1 3 6 242.266
0 2 0 3 2 1 40.885
2 0 1 1 2 2 74.708
2 2 2 2 3 2 50.734
1 0 2 2 1 3 35.950
1 3 3 1 1 1 2.777
3 1 2 1 3 5 118.065
0 3 2 1 2 0 0.000
;
```

The following statements fit a log-linked Poisson model to these data by using classification effects for variables C1–C5:

```
proc hpgenselect data=getStarted;
   class C1-C5;
   model Y = C1-C5 /  Distribution=Poisson Link=Log;
run;
```

The default output from this analysis is presented in Figure 59.1 through Figure 59.8.

The "Performance Information" table in Figure 59.1 shows that the procedure executed in single-machine mode (that is, on the server where SAS is installed). When high-performance procedures run in single-machine mode, they use concurrently scheduled threads. In this case, four threads were used.

Figure 59.1 Performance Information

The HPGENSELECT Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Figure 59.2 displays the "Model Information" table. The variable Y is an integer-valued variable that is modeled by using a Poisson probability distribution, and the mean of Y is modeled by using a log link function. The HPGENSELECT procedure uses a Newton-Raphson algorithm to fit the model. The CLASS variables C1–C5 are parameterized by using GLM parameterization, which is the default.

Figure 59.2 Model Information

Model Information	
Data Source	WORK.GETSTARTED
Response Variable	Y
Class Parameterization	GLM
Distribution	Poisson
Link Function	Log
Optimization Technique	Newton-Raphson with Ridging

Each of the CLASS variables C1–C5 has four unique formatted levels, which are displayed in the "Class Level Information" table in Figure 59.3.

Figure 59.3 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
C1	4	0 1 2 3
C2	4	0 1 2 3
C3	4	0 1 2 3
C4	4	0 1 2 3
C5	4	0 1 2 3

Figure 59.4 displays the "Number of Observations" table. All 100 observations in the data set are used in the analysis.

Figure 59.4 Number of Observations

Number of Observations Read	100
Number of Observations Used	100

Figure 59.5 displays the "Dimensions" table for this model. This table summarizes some important sizes of various model components. For example, it shows that there are 21 columns in the design matrix $\mathbf{X}$: one column for the intercept and 20 columns for the effects that are associated with the classification variables C1–C5. However, the rank of the crossproducts matrix is only 16. Because the classification variables C1–C5 use GLM parameterization and because the model contains an intercept, there is one singularity in

the crossproducts matrix of the model for each classification variable. Consequently, only 16 parameters enter the optimization.

Figure 59.5 Dimensions in Poisson Regression

Dimensions	
Number of Effects	6
Number of Parameters	16
Columns in X	21

Figure 59.6 displays the final convergence status of the Newton-Raphson algorithm. The GCONV= relative convergence criterion is satisfied.

Figure 59.6 Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

The "Fit Statistics" table is shown in Figure 59.7. The –2 log likelihood at the converged estimates is 290.16169. You can use this value to compare the model to nested model alternatives by means of a likelihood-ratio test. To compare models that are not nested, information criteria such as AIC (Akaike's information criterion), AICC (Akaike's bias-corrected information criterion), and BIC (Schwarz Bayesian information criterion) are used. These criteria penalize the –2 log likelihood for the number of parameters.

Figure 59.7 Fit Statistics

Fit Statistics	
-2 Log Likelihood	290.16
AIC (smaller is better)	322.16
AICC (smaller is better)	328.72
BIC (smaller is better)	363.84
Pearson Chi-Square	77.7694
Pearson Chi-Square/DF	0.9258

The "Parameter Estimates" table in Figure 59.8 shows that many parameters have fairly large p-values, indicating that one or more of the model effects might not be necessary.

Figure 59.8 Parameter Estimates

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	0.881903	0.382730	5.3095	0.0212
C1 0	1	-0.196002	0.211482	0.8590	0.3540
C1 1	1	-0.605161	0.263508	5.2742	0.0216
C1 2	1	-0.068458	0.210776	0.1055	0.7453
C1 3	0	0	.	.	.
C2 0	1	0.961117	0.255485	14.1521	0.0002
C2 1	1	0.708188	0.246768	8.2360	0.0041
C2 2	1	0.161741	0.266365	0.3687	0.5437
C2 3	0	0	.	.	.
C3 0	1	-0.227016	0.252561	0.8079	0.3687
C3 1	1	-0.094775	0.229519	0.1705	0.6797
C3 2	1	0.044801	0.238127	0.0354	0.8508
C3 3	0	0	.	.	.
C4 0	1	-0.280476	0.263589	1.1322	0.2873
C4 1	1	0.028157	0.249652	0.0127	0.9102
C4 2	1	0.047803	0.240378	0.0395	0.8424
C4 3	0	0	.	.	.
C5 0	1	-0.817936	0.219901	13.8351	0.0002
C5 1	1	-0.710596	0.206265	11.8684	0.0006
C5 2	1	-0.602080	0.217724	7.6471	0.0057
C5 3	0	0	.	.	.

Syntax: HPGENSELECT Procedure

The following statements are available in the HPGENSELECT procedure:

```
PROC HPGENSELECT <options> ;
    BY variables ;
    CLASS variable <(options)>... <variable <(options)>> </global-options> ;
    CODE <options> ;
    FREQ variable ;
    ID variables ;
    MODEL response<(response-options)> = <effects> </model-options> ;
    MODEL events/trials<(response-options)> = <effects> </model-options> ;
    OUTPUT <OUT=SAS-data-set>
            <keyword <=name>>...
            <keyword <=name>> </options> ;
    PARTITION <partition-options> ;
    PERFORMANCE performance-options ;
    RESTRICT <'label'> constraint-specification <, ..., constraint-specification>
              <operator <value>> </option> ;
    SELECTION selection-options ;
    WEIGHT variable ;
    ZEROMODEL <effects></zeromodel-options> ;
```

The PROC HPGENSELECT statement and at least one MODEL statement are required. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statements.

PROC HPGENSELECT Statement

```
PROC HPGENSELECT <options> ;
```

The PROC HPGENSELECT statement invokes the procedure. Table 59.1 summarizes the available options in the PROC HPGENSELECT statement by function. The options are then described fully in alphabetical order.

Table 59.1 PROC HPGENSELECT Statement Options

Option	Description
Basic Options	
ALPHA=	Specifies a global significance level
DATA=	Specifies the input data set
NAMELEN=	Limits the length of effect names

Table 59.1 *continued*

Option	Description
Output Options	
CORR	Displays the "Parameter Estimates Correlation Matrix" table
COV	Displays the "Parameter Estimates Covariance Matrix" table
ITDETAILS	Displays the "Iteration History" table when no model selection is performed
ITSELECT	Displays the summarized "Iteration History" table when model selection is performed
ITSUMMARY	Displays the summarized "Iteration History" table when no model selection is performed
NOPRINT	Suppresses ODS output
NOCLPRINT	Limits or suppresses the display of classification variable levels
NOSTDERR	Suppresses computation of the covariance matrix and standard errors
Optimization Options	
ABSCONV=	Tunes the absolute function convergence criterion
ABSFCONV=	Tunes the absolute function difference convergence criterion
ABSGCONV=	Tunes the absolute gradient convergence criterion
FCONV=	Tunes the relative function difference convergence criterion
GCONV=	Tunes the relative gradient convergence criterion
INEST=	Specifies the SAS data set that contains starting values, bounds, and constraints for single parameters
LASSORHO=	Specifies the base scaling factor for the LASSO method
LASSOSTEPS=	Specifies the maximum number of steps for the LASSO method
MAXITER=	Chooses the maximum number of iterations in any optimization
MAXFUNC=	Specifies the maximum number of function evaluations in any optimization
MAXTIME=	Specifies the upper limit of CPU time (in seconds) for any optimization
MINITER=	Specifies the minimum number of iterations in any optimization
NORMALIZE=	Specifies whether the objective function is normalized during optimization
OUTEST	Adds parameter names to the "Parameter Estimates" table
TECHNIQUE=	Selects the optimization technique
XCONV=	Tunes the relative parameter difference convergence criterion
Tolerance Options	
LASSOTOL=	Specifies the convergence criterion for the LASSO method
SINGCHOL=	Tunes the singularity criterion for Cholesky decompositions
SINGSWEEP=	Tunes the singularity criterion for the sweep operator
SINGULAR=	Tunes the general singularity criterion
User-Defined Format Options	
FMTLIBXML=	Specifies the file reference for a format stream

You can specify the following *options* in the PROC HPGENSELECT statement.

ABSCONV=*r*

ABSTOL=*r*

specifies an absolute function convergence criterion. For minimization, termination requires $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The

default value of *r* is the negative square root of the largest double-precision value, which serves only as a protection against overflow.

ABSFCONV=*r* < *n* >

ABSFTOL=*r*< *n* >

specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, termination requires a small change of the function value in successive iterations:

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}(k)$ is defined as the vertex that has the lowest function value and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated. The default value is $r = 0$.

ABSGCONV=*r* < *n* >

ABSGTOL=*r*< *n* >

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small:

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $g_j(\cdot)$ is the gradient of the objective function with respect to the *j*th parameter. This criterion is not used by the NMSIMP technique. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated. The default value is $r = 1E{-}8$.

ALPHA=*number*

specifies a global significance level for the construction of confidence intervals. The confidence level is 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05. You can override this global significance level by specifying the ALPHA= option in the MODEL statement or the ALPHA= option in the OUTPUT statement.

CORR

creates the "Parameter Estimates Correlation Matrix" table. The correlation matrix is computed by normalizing the covariance matrix $\boldsymbol{\Sigma}$. That is, if σ_{ij} is an element of $\boldsymbol{\Sigma}$, then the corresponding element of the correlation matrix is $\sigma_{ij}/\sigma_i\sigma_j$, where $\sigma_i = \sqrt{\sigma_{ii}}$.

COV

creates the "Parameter Estimates Covariance Matrix" table. The covariance matrix is computed as the inverse of the negative of the matrix of second derivatives of the log-likelihood function with respect to the model parameters (the Hessian matrix).

DATA=*SAS-data-set*

names the input SAS data set for PROC HPGENSELECT to use. The default is the most recently created data set.

FCONV=*r*< *n* >

FTOL=*r*< *n* >

specifies a relative function difference convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations:

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex that has the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex.

The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated. The default value is $r = 2 \times \epsilon$, where ϵ is the machine precision.

FMTLIBXML=*file-ref*

specifies the file reference for the XML stream that contains the user-defined format definitions. For information about how to generate an XML stream for your formats, see the section "Working with Formats" on page 568.

GCONV=*r*< *n* >

GTOL=*r*< *n* >

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, termination requires that the normalized predicted function reduction be small:

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the CONGRA technique (where a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \, |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated. The default value is *r*=1E–8.

INEST=*SAS-data-set*

names the SAS data set that contains starting values for the parameters. Your data set must include the _TYPE_ variable, a character variable in which the value 'PARMS' indicates the observation that contains your starting values. The data set also includes a numeric variable for each parameter for which you are specifying a starting value; the name of this numeric variable is the parameter name. You can obtain parameter names by specifying the OUTEST option and by using the ODS OUTPUT statement to output the "Parameter Estimates" table into a data set; the parameter name is contained in

the ParmName variable in this data set. If you do not specify a starting value for a parameter, it is set to 0. PROC HPGENSELECT uses only the first observation for which _TYPE_=PARMS, and it ignores BY variables. You can also specify single-parameter equality constraints by using a value of 'EQ' for the variable _TYPE_ to indicate the observation that contains your equality constraints, and similarly by using values for _TYPE_ of 'UB' for upper bounds and 'LB' for lower bounds on parameters.

ITDETAILS

adds to the "Iteration History" table the current values of the parameter estimates and their gradients. These quantities are reported only for parameters that participate in the optimization. This option is not available when you perform model selection.

ITSELECT

generates the "Iteration History" table when you perform a model selection.

ITSUMMARY

generates the "Iteration History" table. This option is not available when you perform model selection.

LASSORHO=*r*

specifies the base scaling factor that is applied to the maximum regularization parameter for the LASSO model selection method. The regularization parameter for step i is mr^i, where m is the maximum regularization parameter.

LASSOSTEPS=*n*

specifies the maximum number of steps for LASSO model selection.

LASSOTOL=*r*

specifies the convergence tolerance for the optimization algorithm that solves for the LASSO parameter estimates at each step of LASSO model selection.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, NEWRAP: $n = 125$
- QUANEW, DBLDOG: $n = 500$
- CONGRA: $n = 1{,}000$
- NMSIMP: $n = 3{,}000$

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed n. You can choose the optimization technique by specifying the TECHNIQUE= option.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, NEWRAP: $n = 50$

- QUANEW, DBLDOG: $n = 200$
- CONGRA: $n = 400$
- NMSIMP: $n = 1{,}000$

These default values also apply when *n* is specified as a missing value. You can choose the optimization technique by specifying the TECHNIQUE= option.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The default value is the largest floating-point double representation of your computer. The time specified by this option is checked only once at the end of each iteration. Therefore, the actual running time can be longer than *r*.

MINITER=*n*

MINIT=*n*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms might behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

NAMELEN=*number*

specifies the length to which long effect names are shortened. The default and minimum value is 20.

NOCLPRINT< =*number* **>**

suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying a *number* helps to reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOPRINT

suppresses the generation of ODS output.

NORMALIZE=YES | NO

specifies whether to normalize the objective function during optimization by the reciprocal of the frequency count of observations that are used in the analysis. This option affects the values that are reported in the "Iteration History" table. The results that are reported in the "Fit Statistics" are always displayed for the nonnormalized log-likelihood function. By default, NORMALIZE = NO.

NOSTDERR

suppresses the computation of the covariance matrix and the standard errors of the regression coefficients. When the model contains many variables (thousands), the inversion of the Hessian matrix to derive the covariance matrix and the standard errors of the regression coefficients can be time-consuming.

OUTEST

adds a column for the ParmName variable to the "Parameter Estimates" table. This column is not displayed, but you can use it to create a data set that you can specify in an INEST= option by first using the ODS OUTPUT statement to output the "Parameter Estimates" table and then submitting the following statements:

```
proc transpose data=parameterestimates out=inest label=_TYPE_;
   label Estimate=PARMS;
   var Estimate;
   id ParmName;
run;
```

SINGCHOL=*number*

tunes the singularity criterion in Cholesky decompositions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGSWEEP=*number*

tunes the singularity criterion for sweep operations. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGULAR=*number*

tunes the general singularity criterion that is applied in sweeps and inversions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

TECHNIQUE=*keyword*

TECH=*keyword*

specifies the optimization technique for obtaining maximum likelihood estimates. You can choose from the following techniques by specifying the appropriate *keyword*:

CONGRA	performs a conjugate-gradient optimization.
DBLDOG	performs a version of double-dogleg optimization.
NEWRAP	performs a Newton-Raphson optimization with line search.
NMSIMP	performs a Nelder-Mead simplex optimization.
NONE	performs no optimization.
NRRIDG	performs a Newton-Raphson optimization with ridging.
QUANEW	performs a dual quasi-Newton optimization.
TRUREG	performs a trust-region optimization

The default value is TECHNIQUE=NRRIDG, except for the Tweedie distribution, for which the default value is TECHNIQUE=QUANEW.

For more information, see the section "Choosing an Optimization Algorithm" on page 4774.

XCONV=*r*< *n* >

XTOL=*r*< *n* >

specifies the relative parameter convergence criterion. The termination criterion and the default value depend on the technique, as follows:

- For all techniques except NMSIMP, termination requires a small relative parameter change in subsequent iterations:

$$\frac{\max_j |\psi_j^{(k)} - \psi_j^{(k-1)}|}{\max(|\psi_j^{(k)}|, |\psi_j^{(k-1)}|)} \leq r$$

By default, XCONV = 0.

- For the NMSIMP technique, the same formula is used, but $\psi_j^{(k)}$ is defined as the vertex with the lowest function value and $\psi_j^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default value is r = 1E–8.

The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPGENSELECT to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPGENSELECT procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) >. . . < *variable* < (*options*) > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. The CLASS statement must precede the MODEL statement. You can list the response variable for binary and multinomial models in the CLASS statement, but this is not necessary.

The CLASS statement is documented in the section "CLASS Statement" on page 572.

The HPGENSELECT procedure additionally supports the following *global-option* in the CLASS statement:

UPCASE

uppercases the values of character-valued CLASS variables before levelizing them. For example, if the UPCASE option is in effect and a CLASS variable can take the values 'a', 'A', and 'b', then 'a' and 'A' represent the same level and the CLASS variable is treated as having only two values: 'A' and 'B'.

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 59.2 summarizes the *options* available in the CODE statement.

Table 59.2 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence for each observation. PROC HPGENSELECT treats each observation as if it appeared f times, where the frequency value f is the value of the FREQ variable for the observation. If f is not an integer, then f is truncated to an integer. If f is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are to be transferred to the output data set that is specified in the OUTPUT statement.

For more information, see the section "ID Statement" on page 576.

MODEL Statement

MODEL *response* < **(***response-options***)** > **=** < *effects* > < / *model-options* > **;**

MODEL *events* / *trials* **=** < *effects* > < / *model-options* > **;**

The MODEL statement defines the statistical model in terms of a *response* variable (the target) or an *events/trials* specification. You can also specify model effects that are constructed from variables in the input data set, and you can specify options. An intercept is included in the model by default. You can remove the intercept by specifying the NOINT option.

You can specify a single *response* variable that contains your interval, binary, ordinal, or nominal response values. When you have binomial data, you can specify the *events/trials* form of the response, where one variable contains the number of positive responses (or events) and another variable contains the number of trials. The values of both *events* and (*trials* – *events*) must be nonnegative, and the value of *trials* must be positive. If you specify a single *response* variable that is in a CLASS statement, then the response is assumed to be either binary or multinomial, depending on the number of levels.

For information about constructing the model effects, see the section "Parameterization of Model Effects" on page 412.

There are two sets of options in the MODEL statement. The *response-options* determine how the HPGENSELECT procedure models probabilities for binary and multinomial data. The *model-options* control other aspects of model formation and inference. Table 59.3 summarizes these options.

Table 59.3 MODEL Statement Options

Option	Description
Response Variable Options for Binary and Multinomial Models	
DESCENDING	Reverses the response categories
EVENT=	Specifies the event category
ORDER=	Specifies the sort order
REF=	Specifies the reference category

Table 59.3 *continued*

Option	Description
Model Options	
ALPHA=	Specifies the confidence level for confidence limits
CL	Requests confidence limits
DISPERSION | PHI=	Specifies a fixed dispersion parameter
DISTRIBUTION | DIST=	Specifies the response distribution
INCLUDE=	Includes effects in all models for model selection
INITIALPHI=	Specifies a starting value of the dispersion parameter
LINK=	Specifies the link function
NOCENTER	Requests that continuous main effects not be centered and scaled
NOINT	Suppresses the intercept
OFFSET=	Specifies the offset variable
SAMPLEFRAC=	Specifies the fraction of the data to be used to compute starting values for the Tweedie distribution
START=	Includes effects in the initial model for model selection

Response Variable Options

Response variable options determine how the HPGENSELECT procedure models probabilities for binary and multinomial data.

You can specify the following *response-options* by enclosing them in parentheses after the *response* or *trials* variable.

DESCENDING
DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC HPGENSELECT orders the response categories according to the ORDER= option and then reverses that order.

EVENT='*category*' | FIRST | LAST

specifies the event category for the binary response model. PROC HPGENSELECT models the probability of the event category. The EVENT= option has no effect when there are more than two response categories.

You can specify the event *category* (formatted, if a format is applied) in quotes, or you can specify one of the following:

FIRST

designates the first ordered category as the event. This is the default.

LAST

designates the last ordered category as the event.

For example, the following statements specify that observations that have a formatted value of '1' represent events in the data. The probability modeled by the HPGENSELECT procedure is thus the probability that the variable def takes on the (formatted) value '1'.

```
proc hpgenselect data=MyData;
   class A B C;
   model def(event ='1') = A B C x1 x2 x3;
run;
```

ORDER=DATA | FORMATTED | INTERNAL

ORDER=FREQ | FREQDATA | FREQFORMATTED | FREQINTERNAL

specifies the sort order for the levels of the *response* variable. When ORDER=FORMATTED (the default) for numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC HPGENSELECT run or in the DATA step that created the data set), the levels are ordered by their internal (numeric) value. Table 59.4 shows the interpretation of the ORDER= option.

Table 59.4 Sort Order

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count (levels that have the most observations come first in the order)
FREQDATA	Order of descending frequency count; within counts by order of appearance in the input data set when counts are tied
FREQFORMATTED	Order of descending frequency count; within counts by formatted value when counts are tied
FREQINTERNAL	Order of descending frequency count; within counts by unformatted value when counts are tied
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For the FORMATTED and INTERNAL orders, the sort order is machine-dependent.

For more information about sort order, see the chapter about the SORT procedure in *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Language Reference: Concepts*.

REF='*category*' | FIRST | LAST

specifies the reference category for the generalized logit model and the binary response model. For the generalized logit model, each logit contrasts a nonreference category with the reference category. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the reference *category* (formatted if a format is applied) in quotes, or you can specify one of the following:

FIRST
designates the first ordered category as the reference

LAST
designates the last ordered category as the reference. This is the default.

Model Options

ALPHA=*number*
requests that confidence intervals for each of the parameters that are requested by the CL option be constructed with confidence level 1−*number*. The value of *number* must be between 0 and 1; the default is 0.05.

CL
requests that confidence limits be constructed for each of the parameter estimates. The confidence level is 0.95 by default; this can be changed by specifying the ALPHA= option.

DISPERSION=*number*
specifies a fixed dispersion parameter for those distributions that have a dispersion parameter. The dispersion parameter used in all computations is fixed at *number*, and not estimated.

DISTRIBUTION=*keyword*
specifies the response distribution for the model. The *keywords* and the associated distributions are shown in Table 59.5.

Table 59.5 Built-In Distribution Functions

DISTRIBUTION=	Distribution Function
BINARY	Binary
BINOMIAL	Binary or binomial
GAMMA	Gamma
INVERSEGAUSSIAN | IG	Inverse Gaussian
MULTINOMIAL | MULT	Multinomial
NEGATIVEBINOMIAL | NB	Negative binomial
NORMAL | GAUSSIAN	Normal
POISSON	Poisson
TWEEDIE< (*Tweedie-options*) >	Tweedie
ZINB	Zero-inflated negative binomial
ZIP	Zero-inflated Poisson

When DISTRIBUTION=TWEEDIE, you can specify the following *Tweedie-options*:

INITIALP=
specifies a starting value for iterative estimation of the Tweedie power parameter.

OPTMETHOD=*Tweedie-optimization-option*
requests an optimization method for iterative estimation of the Tweedie model parameters. You can specify the following *Tweedie-optimization-options*:

EQL
requests that extended quasi-likelihood be used for a sample of the data, followed by extended quasi-likelihood for the full data. This is equivalent to the TWEEDIEEQL *Tweedie-option*.

EQLLHOOD
requests that extended quasi-likelihood be used for a sample of the data, followed by Tweedie log likelihood for the full data. This is the default method.

FINALLHOOD
requests a four-stage approach to estimating the Tweedie model parameters. The four stages are as follows:

1. extended quasi-likelihood for a sample of the data
2. Tweedie log likelihood for a sample of the data
3. extended quasi-likelihood for the full data
4. Tweedie log likelihood for the full data

LHOOD
requests that Tweedie log likelihood be used for a sample of the data, followed by Tweedie log likelihood for the full data.

P=
requests a fixed Tweedie power parameter.

TWEEDIEEQL | EQL
requests that extended quasi-likelihood be used instead of Tweedie log likelihood in parameter estimation.

If you do not specify a link function with the LINK= option, a default link function is used. The default link function for each distribution is shown in Table 59.6. For the binary and multinomial distributions, only the link functions shown in Table 59.6 are available. For the other distributions, you can use any link function shown in Table 59.7 by specifying the LINK= option. Other commonly used link functions for each distribution are shown in Table 59.6.

Table 59.6 Default and Commonly Used Link Functions

DISTRIBUTION=	Default Link Function	Other Commonly Used Link Functions
BINARY	Logit	Probit, complementary log-log, log-log
BINOMIAL	Logit	Probit, complementary log-log, log-log
GAMMA	Reciprocal	Log
INVERSEGAUSSIAN | IG	Reciprocal square	Log
MULTINOMIAL | MULT	Logit (ordinal)	Probit, complementary log-log, log-log

Table 59.6 *continued*

DISTRIBUTION=	Default Link Function	Other Commonly Used Link Functions
MULTINOMIAL | MULT	Generalized logit (nominal)	
NEGATIVEBINOMIAL | NB	Log	
NORMAL | GAUSSIAN	Identity	Log
POISSON	Log	
TWEEDIE	Log	
ZINB	Log	
ZIP	Log	

INCLUDE=*n*

INCLUDE=*single-effect*

INCLUDE=(*effects***)**

forces effects to be included in all models. If you specify INCLUDE=*n*, then the first *n* effects that are listed in the MODEL statement are included in all models. If you specify INCLUDE=*single-effect* or if you specify a list of effects within parentheses, then the specified effects are forced into all models. The effects that you specify in this option must be explanatory effects that are specified in the MODEL statement before the slash (/).

INITIALPHI=*number*

specifies a starting value for iterative maximum likelihood estimation of the dispersion parameter for distributions that have a dispersion parameter.

LINK=*keyword*

specifies the link function for the model. The *keywords* and the associated link functions are shown in Table 59.7. Default and commonly used link functions for the available distributions are shown in Table 59.6.

Table 59.7 Built-In Link Functions

LINK=	Link Function	$g(\mu) = \eta =$
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$
GLOGIT | GENLOGIT	Generalized logit	
IDENTITY | ID	Identity	μ
INV | RECIP	Reciprocal	$\frac{1}{\mu}$
INV2	Reciprocal square	$\frac{1}{\mu^2}$
LOG	Logarithm	$\log(\mu)$
LOGIT	Logit	$\log(\mu/(1-\mu))$
LOGLOG	Log-log	$-\log(-\log(\mu))$
PROBIT	Probit	$\Phi^{-1}(\mu)$

$\Phi^{-1}(\cdot)$ denotes the quantile function of the standard normal distribution.

If a multinomial response variable has more than two categories, the HPGENSELECT procedure

fits a model by using a cumulative link function that is based on the specified link. However, if you specify LINK=GLOGIT, the procedure assumes a generalized logit model for nominal (unordered) data, regardless of the number of response categories.

NOCENTER
requests that continuous main effects not be centered and scaled internally. (Continuous main effects are centered and scaled by default to aid in computing maximum likelihood estimates.) Parameter estimates and related statistics are always reported on the original scale.

NOINT
requests that no intercept be included in the model. (An intercept is included by default.) The NOINT option is not available in multinomial models.

OFFSET=*variable*
specifies a *variable* to be used as an offset to the linear predictor. An offset plays the role of an effect whose coefficient is known to be 1. The offset variable cannot appear in the CLASS statement or elsewhere in the MODEL statement. Observations that have missing values for the offset variable are excluded from the analysis.

SAMPLEFRAC=*number*
specifies a fraction of the data to be used to determine starting values for iterative estimation of the parameters of a Tweedie model. The sampled data are used in an extended quasi-likelihood estimation of the model parameters. The estimated parameters are then used as starting values in a full maximum likelihood estimation of the model parameters that uses all of the data.

START=*n*
START=*single-effect*
START=(*effects***)**
begins the selection process from the designated initial model for the FORWARD and STEPWISE selection methods. If you specify START=*n*, then the starting model includes the first *n* effects that are listed in the MODEL statement. If you specify START=*single-effect* or if you specify a list of effects within parentheses, then the starting model includes those specified effects. The effects that you specify in the START= option must be explanatory effects that are specified in the MODEL statement before the slash (/). The START= option is not available when you specify METHOD=BACKWARD in the SELECTION statement.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< *keyword* < =*name* > >. . . < *keyword* < =*name* > > < / *options* > **;**

The OUTPUT statement creates a data set that contains observationwise statistics that are computed after the model is fitted. The variables in the input data set are *not* included in the output data set to avoid data duplication for large data sets; however, variables that are specified in the ID statement are included.

The computation of the output statistics is based on the final parameter estimates. If the model fit does not converge, missing values are produced for the quantities that depend on the estimates.

When there are more than two response levels for multinomial data, values are computed only for variables that are named by the LINP and PREDICTED keywords; the other variables have missing values. These

statistics are computed for every response category, and the automatic variable _LEVEL_ identifies the response category on which the computed values are based. If you also specify the OBSCAT option, then the observationwise statistics are computed only for the observed response category, as indicated by the value of the _LEVEL_ variable.

For observations in which only the response variable is missing, values of the XBETA and PREDICTED statistics are computed even though these observations do not affect the model fit. For zero-inflated models, ZBETA and PZERO are also computed. This practice enables predicted mean values or predicted probabilities to be computed for new observations.

You can specify the following syntax elements in the OUTPUT statement before the slash (/).

OUT=*SAS-data-set*

DATA=*SAS-data-set*

specifies the name of the output data set. If the OUT= (or DATA=) option is omitted, the procedure uses the DATA*n* convention to name the output data set.

keyword **<=***name***>**

specifies a statistic to include in the output data set and optionally assigns a *name* to the variable. If you do not provide a *name*, the HPGENSELECT procedure assigns a default name based on the type of statistic requested.

You can specify the following *keywords* for adding statistics to the OUTPUT data set:

ADJPEARSON<=*name***>**

ADJPEARS<=*name***>**

STDRESCHI<=*name***>**

requests the Pearson residual, adjusted to have unit variance. The adjusted Pearson residual is defined for the *i*th observation as $\frac{y_i-\mu_i}{\sqrt{\phi \mathrm{V}(\mu_i)(1-h_i)}}$, where $\mathrm{V}(\mu)$ is the response distribution variance function and h_i is the leverage. The leverage h_i of the *i*th observation is defined as the *i*th diagonal element of the hat matrix

$$\mathbf{H} = \mathbf{W}^{\frac{1}{2}}\mathbf{X}(\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}^{\frac{1}{2}}$$

where $\mathbf{W}$ is the diagonal matrix whose *i*th diagonal is $w_{ei} = \frac{w_i}{\phi \mathrm{V}(\mu_i)(g'(\mu_i))^2}$, and w_i is a prior weight specified in a WEIGHT statement or 1 if no WEIGHT statement is specified. For the negative binomial, $\phi \mathrm{V}(\mu_i)$ in the denominator is replaced with the distribution variance, in both the definition of the leverage and the adjusted residual.

This statistic is not computed for multinomial models, nor is it computed for zero-modified models.

If you do not specify a *name*, PROC HPGENSELECT assigns Adjusted_Pearson as the *name*.

LINP<=*name***>**

XBETA<=*name***>**

requests the linear predictor $\eta = \mathbf{x}'\boldsymbol{\beta}$.

If you do not specify a *name*, PROC HPGENSELECT assigns Xbeta as the *name*.

LOWER<=name>

requests a lower confidence limit for the predicted value. This statistic is not computed for generalized logit multinomial models or zero-modified models.

If you do not specify a *name*, PROC HPGENSELECT assigns Lower as the *name*.

PEARSON<=*name*>

PEARS<=*name*>

RESCHI<=*name*>

requests the Pearson residual, $\frac{y-\mu}{\sqrt{\mathrm{V}(\mu)}}$, where μ is the estimate of the predicted response mean and $\mathrm{V}(\mu)$ is the response distribution variance function. For the negative binomial defined in the section "Negative Binomial Distribution" on page 4763 and the zero-inflated models defined in the sections "Zero-Inflated Poisson Distribution" on page 4765 and "Zero-Inflated Negative Binomial Distribution" on page 4765, the distribution variance is used in place of $\mathrm{V}(\mu)$.

This statistic is not computed for multinomial models.

If you do not specify a *name*, PROC HPGENSELECT assigns Pearson as the *name*.

PREDICTED<=*name*>

PRED<=*name*>

P<=*name*>

requests predicted values for the response variable.

If you do not specify a *name*, PROC HPGENSELECT assigns Pred as the *name*.

PZERO<=*name*>

requests zero-inflation probabilities for zero-inflated models.

If you do not specify a *name*, PROC HPGENSELECT assigns Pzero as the *name*.

RESIDUAL<=*name*>

RESID<=*name*>

R<=*name*>

requests the raw residual, $y - \mu$, where μ is the estimate of the predicted mean. This statistic is not computed for multinomial models.

If you do not specify a *name*, PROC HPGENSELECT assigns Residual as the *name*.

ROLE<=*name*>

requests a numeric variable that indicates the role played by each observation in fitting the model. Table 59.8 shows the interpretation of this variable for each observation.

Table 59.8 Role Interpretation

Value	Observation Role
0	Not used
1	Training
2	Validation
3	Testing

If you do not partition the input data by specifying a PARTITION statement, then the role variable value is 1 for observations that are used in fitting the model and 0 for observations that have at least one missing or invalid value for the response, regressors, frequency, or weight variable.

If you do not specify a *name*, PROC HPGENSELECT assigns Role as the *name*.

UPPER< =*name*>
requests an upper confidence limit for the predicted value. This statistic is not computed for generalized logit multinomial models or zero-modified models.

If you do not specify a *name*, PROC HPGENSELECT assigns Upper as the *name*.

ZBETA< =*name*>
requests the linear predictor for the zeros model in zero-modified models: $\kappa = \mathbf{z}'\boldsymbol{\gamma}$.

If you do not specify a *name*, PROC HPGENSELECT assigns Zbeta as the *name*.

You can specify the following *options* in the OUTPUT statement after the slash (/):

ALPHA=*number*
specifies the significance level for the construction of confidence intervals in the OUTPUT data set. The confidence level is $1 - \mathit{number}$.

OBSCAT
requests (for multinomial models) that observationwise statistics be produced only for the response level. If the OBSCAT option is not specified and the response variable has J levels, then the following outputs are created: for cumulative link models, $J - 1$ records are output for every observation in the input data that corresponds to the $J - 1$ lower-ordered response categories; for generalized logit models, J records are output that correspond to all J response categories.

PARTITION Statement

PARTITION < *partition-option* > **;**

The PARTITION statement specifies how observations in the input data set are to be logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section "Using Validation and Test Data" on page 4772. You can either designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or specify proportions to use for random assignment of observations for each role.

You can specify one of the following mutually exclusive *partition-options*:

ROLEVAR | ROLE=*variable*(< TEST='*value*' > < TRAIN='*value*' > < VALIDATE='*value*' >)
names the variable in the input data set whose values are used to assign roles to each observation. The TEST=, TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to assign observations roles. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboption are assigned to training.

FRACTION(< TEST=*fraction***> < VALIDATE=***fraction***> < SEED=***number***>)**
randomly assigns specified proportions of the observations in the input data set to the roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and the VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. The SEED= option specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify a seed, or if you specify a *number* less than or equal to 0, the seed is generated by reading the time of day from the computer's clock.

PERFORMANCE Statement

PERFORMANCE < *performance-options*> **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of the HPGENSELECT procedure.

The PERFORMANCE statement is documented in the section "PERFORMANCE Statement" on page 569.

RESTRICT Statement

RESTRICT < *'label'*> *constraint-specification* < **,** . . . **,** *constraint-specification*>
< *operator* < *value*> > < / *option*> **;**

The RESTRICT statement enables you to specify linear equality or inequality constraints among the parameters of a model. These restrictions are incorporated into the maximum likelihood analysis.

Following are reasons why you might want to place constraints and restrictions on the model parameters:

- to fix a parameter at a particular value
- to equate parameters in a model
- to impose order conditions on the parameters in a model
- to specify contrasts among the parameters that the fitted model should honor

A restriction is composed of a left-hand side and a right-hand side, separated by an operator. If you do not specify the operator and right-hand side, the restriction is assumed to be an equality constraint against zero. If you do not specify the right-hand side, the value is assumed to be zero.

You write an individual *constraint-specification* in (nearly) the same form as you specify estimable linear functions in the ESTIMATE statement of the GLM, MIXED, or GLIMMIX procedure. The *constraint-specification* takes the form

model-effect value-list < . . . *model-effect value-list* >

You must specify at least one *model-effect*, followed by one or more values in the *value-list*. The values in the list correspond to the multipliers of the corresponding parameter that is associated with the position in the model effect. If you specify more values in the *value-list* than the *model-effect* occupies in the model design matrix, the extra coefficients are ignored.

The following statements provide an example. Here, A is a CLASS variable that has three levels.

```
proc hpgenselect;
   class A;
   model y/n = A x /  dist=binomial;
   restrict A 1 0 -1;
   restrict x 2 >= 0.5;
run;
```

The linear predictor for this model can be written as

$$\eta = \beta_0 + \beta_1 A_1 + \beta_2 A_2 + \beta_3 A_3 + x\beta_4$$

where A_k is the binary variable associated with the kth level of A.

The first RESTRICT statement specifies that the parameter estimates that are associated with the first and third levels of the A effect be identical. In terms of the linear predictor, the restriction can be written as

$$\beta_1 - \beta_3 = 0$$

Because, in the default GLM parameterization, $\beta_3 = 0$, the RESTRICT statement has the effect of setting $\beta_1 = 0$.

The second RESTRICT statement involves the regression parameter associated with the variable x and specifies that the parameter estimate satisfy $\beta_4 \geq 0.25$. In terms of the linear predictor, the restriction can be written as

$$2\beta_4 \geq \frac{1}{2}$$

PROC HPGENSELECT applies both of these restrictions when it computes the maximum likelihood estimates of the regression parameters of the model.

Zero-inflated models contain two components: a model for the mean of the underlying distribution and a model for the zero-inflation probability. To specify restrictions for effects in specific components of the model, separate the *constraint-specifications* by commas. The following statements provide an example:

```
proc hpgenselect data=b itdetails itselect cov;
   class  C;
   model B = C / dist=ZIP;
   zeromodel X;
   restrict Intercept 0, X 1 = 0;
run;
```

In this example, the model for the mean has a single regressor, which is specified by the CLASS variable C. The model for the zero-inflation probability has a continuous regressor X. The RESTRICT statement specifies that the parameter estimate associated with X be constrained to be 0. The *Intercept 0 constraint-specification* serves as a placeholder and has no effect on the model for the mean. You must include this *model-effect value-list* pair in order to specify constraints on the zero-inflation part of the model. You can use any *model-effect* in the model for the mean in place of *Intercept*. For example, the following statement has the same effect, because C is in the model for the mean:

```
restrict C 0, X 1 = 0;
```

The generalized logit model for a nominal multinomial response consists of a regression model for each nonreference level of the response variable. To specify restrictions for effects in specific components of the model, you specify a *constraint-specification* for each component to which you want to apply constraints. You specify the *constraint-specifications* in the sort order of the response variable and separate them with commas. You must specify a null *constraint-specification* with a *value-list* set to zero for each component model that has a lower response variable sort order than the one to which you want to apply constraints. The following statements provide an example. In this example, a generalized logit regression model is fit to the categorical response variable Y, with four levels. The generalized logit model consists of a regression model with a CLASS regressor Visit and a continuous regressor Lage for each level of the response variable Y. The RESTRICT statements constrain the model to have identical values of the estimated regression coefficient for Lage for all three nonreference categories of Y; that is, a common-slopes model is fit. In the second RESTRICT statement, the *constraint-specification* of *Lage 0* is necessary as a placeholder and does not affect the regression coefficient of Lage for the first level of Y.

```
proc hpgenselect data=thallMult_hgen7809;
   class Visit / Param=Ref;
   model Y=Visit Lage/dist=Multinomial link=Glogit;
   restrict Lage 1 , Lage -1;
   restrict Lage 0 , Lage  1, Lage -1;
run;
```

You can use following operators to separate the left- and right-hand sides of the restriction: **=**, **>**, **<**, **>=**, **<=**.

Some distributions involve a dispersion parameter (the parameter ϕ in the expressions for the log likelihood), and in the case of the Tweedie distribution, a power parameter. You cannot use the RESTRICT statement to constrain either of these parameters. Instead, you can use the MODEL statement options PHI= to set the dispersion to a fixed value and P= to set the Tweedie power parameter to a fixed value.

You can specify the following *option* after a slash (/):

DIVISOR=*value*
specifies a *value* by which all coefficients on the right-hand and left-hand sides of the restriction are divided.

SELECTION Statement

SELECTION < *options* > **;**

The SELECTION statement performs model selection by examining whether effects should be added to or removed from the model according to rules that are defined by model selection methods. The statement is fully documented in the section "SELECTION Statement" on page 576.

The HPGENSELECT procedure supports the following effect-selection methods in the SELECTION statement:

METHOD=NONE results in no model selection. This method fits the full model.

METHOD=BACKWARD performs backward elimination. This method starts with all effects in the model and deletes effects.

METHOD=FORWARD performs forward selection. This method starts with no effects in the model and adds effects.

METHOD=LASSO performs model selection by the group LASSO method. This method adds and removes effects by using a sequence of LASSO steps.

METHOD=STEPWISE performs stepwise regression. This method is similar to the FORWARD method except that effects already in the model do not necessarily stay there.

For methods other than LASSO, the only effect-selection criterion that the HPGENSELECT procedure supports is SELECT=SL, in which effects enter and leave the model based on an evaluation of the significance level. To determine the level of significance for each candidate effect, PROC HPGENSELECT calculates an approximate chi-square test statistic. The SELECT= option is not supported by the LASSO method.

You can specify the following criteria in the CHOOSE= option:

AIC specifies Akaike's information criterion (Akaike 1974).

AICC specifies a small-sample bias-corrected version of Akaike's information criterion as promoted in Hurvich and Tsai (1989) and Burnham and Anderson (1998), among others.

BIC | SBC specifies the Schwarz Bayesian criterion (Schwarz 1978).

VALIDATE specifies the average squared error (ASE) that is computed from validation data, if you specify validation data by using a PARTITION statement. For more information about ASE and partitioned data, see the section "Using Validation and Test Data" on page 4772

You can specify the following criteria in the STOP= option:

SL specifies the significance level of the test.

AIC specifies Akaike's information criterion (Akaike 1974).

AICC specifies a small-sample bias-corrected version of Akaike's information criterion as promoted in Hurvich and Tsai (1989) and Burnham and Anderson (1998), among others.

BIC | SBC specifies the Schwarz Bayesian criterion (Schwarz 1978).

If you specify METHOD=LASSO and you do not specify either the CHOOSE= or STOP= option, then the model in the last LASSO step is chosen as the selected model.

The calculation of the information criteria uses the following formulas, where p denotes the number of effective parameters in the candidate model, f denotes the number of frequencies used, and l is the log likelihood evaluated at the converged estimates:

$$\mathrm{AIC} = -2l + 2p$$

$$\mathrm{AICC} = \begin{cases} -2l + 2pf/(f-p-1) & \text{when } f > p+2 \\ -2l + 2p(p+2) & \text{otherwise} \end{cases}$$

$$\mathrm{BIC} = -2l + p\log(f)$$

If you specify the PARTITION statement, then the AIC, AICC, BIC, and SL statistics are computed on the training data set; otherwise they are computed on the full data set.

When you specify one of the following DETAILS= options in the SELECTION statement, the HPGENSELECT procedure produces the indicated tables:

DETAILS=STEPS produces a table of selection details that displays fit statistics for the model at each step of the selection process and the approximate log *p*-value. The summary table that results from the DETAILS=SUMMARY option is also produced. This option has no effect when you use the LASSO method.

DETAILS=SUMMARY produces a summary table that shows which effect is added or removed at each step along with the *p*-value. The summary table is produced by default if you do not specify the DETAILS= option. This option has no effect when you use the LASSO method.

DETAILS=ALL for methods other than LASSO, produces all the tables that are produced when DETAILS=STEPS and also produces a table that displays the effect that is added or removed at each step along with the *p*-value, chi-square statistic, and fit statistics for the model. For the LASSO method, it produces a table that displays the effects that are added or removed at each step; the scaling factor that is multiplied by the maximum regularization parameter to determine the LASSO regularization parameter for the corresponding step; and the AIC, AICC, and BIC fit statistics.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, then all observations used in the analysis are assigned a weight of 1.

ZEROMODEL Statement

ZEROMODEL < *effects* > < / *zeromodel-options* > **;**

The ZEROMODEL statement defines the statistical model for zero inflation probability in terms of model effects that are constructed from variables in the input data set. An intercept term is always included in the model.

You can specify the following *zeromodel-options*.

INCLUDE=*n*

INCLUDE=*single-effect*

INCLUDE=(*effects***)**

forces effects to be included in all models for zero inflation for all selection methods. If you specify INCLUDE=*n*, then the first *n* effects that are listed in the ZEROMODEL statement are included in all models. If you specify INCLUDE=*single-effect* or if you specify a list of effects within parentheses, then the specified effects are forced into all models. The effects that you specify in the INCLUDE= option must be explanatory effects that are specified in the ZEROMODEL statement before the slash (/).

LINK=*keyword*

specifies the link function for the zero inflation probability. The *keywords* and the associated link functions are shown in Table 59.9.

Table 59.9 Built-In Link Functions for Zero Inflation Probability

LINK=	Link Function	$g(\mu) = \eta =$
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$
LOGIT	Logit	$\log(\mu/(1-\mu))$
LOGLOG	Log-log	$-\log(-\log(\mu))$
PROBIT	Probit	$\Phi^{-1}(\mu)$

$\Phi^{-1}(\cdot)$ denotes the quantile function of the standard normal distribution.

START=*n*

START=*single-effect*

START=(*effects***)**

begins the selection process from the designated initial zero inflation model for the FORWARD and STEPWISE selection methods. If you specify START=*n*, then the starting model includes the first *n* effects that are listed in the ZEROMODEL statement. If you specify START=*single-effect* or if you specify a list of effects within parentheses, then the starting model includes those specified effects. The effects that you specify in the START= option must be explanatory effects that are specified in the ZEROMODEL statement before the slash (/). The START= option is not available when you specify METHOD=BACKWARD in the SELECTION statement.

Details: HPGENSELECT Procedure

Missing Values

Any observation that has missing values for the response, frequency, weight, offset, or explanatory variables is excluded from the analysis; however, missing values are valid for response and explanatory variables that are specified in the MISSING option in the CLASS statement. Observations that have a nonpositive weight or a frequency less than 1 are also excluded.

The estimated linear predictor and the fitted probabilities are not computed for any observation that has missing offset or explanatory variable values. However, if only the response value is missing, the linear predictor and the fitted probabilities can be computed and output to a data set by using the OUTPUT statement.

Exponential Family Distributions

Many of the probability distributions that the HPGENSELECT procedure fits are members of an exponential family of distributions, which have probability distributions that are expressed as follows for some functions b and c that determine the specific distribution:

$$f(y) = \exp\left\{\frac{y\theta - b(\theta)}{\phi} + c(y,\phi)\right\}$$

For fixed ϕ, this is a one-parameter exponential family of distributions. The response variable can be discrete or continuous, so $f(y)$ represents either a probability mass function or a probability density function. A more useful parameterization of generalized linear models is by the mean and variance of the distribution:

$$\begin{aligned} \mathrm{E}(Y) &= b'(\theta) \\ \mathrm{Var}(Y) &= b''(\theta)\phi \end{aligned}$$

In generalized linear models, the mean μ of the response distribution is related to linear regression parameters through a link function,

$$g(\mu_i) = \mathbf{x}_i'\boldsymbol{\beta}$$

for the ith observation, where $\mathbf{x}_i$ is a fixed known vector of explanatory variables and $\boldsymbol{\beta}$ is a vector of regression parameters. The HPGENSELECT procedure parameterizes models in terms of the regression parameters $\boldsymbol{\beta}$ and either the dispersion parameter ϕ or a parameter that is related to ϕ, depending on the model. For exponential family models, the distribution variance is $\mathrm{Var}(Y) = \phi\mathrm{V}(\mu)$ where $\mathrm{V}(\mu)$ is a variance function that depends only on μ.

The zero-inflated models and the multinomial models are not exponential family models, but they are closely related models that are useful and are included in the HPGENSELECT procedure.

Response Distributions

The response distribution is the probability distribution of the response (target) variable. The HPGENSELECT procedure can fit data for the following distributions:

- binary distribution
- binomial distribution
- gamma distribution
- inverse Gaussian distribution
- multinomial distribution (ordinal and nominal)
- negative binomial distribution
- normal (Gaussian) distribution
- Poisson distribution
- Tweedie distribution
- zero-inflated negative binomial distribution
- zero-inflated Poisson distribution

Expressions for the probability distributions (probability density functions for continuous variables or probability mass functions for discrete variables) are shown in the section "Response Probability Distribution Functions" on page 4762. The expressions for the log-likelihood functions of these distributions are given in the section "Log-Likelihood Functions" on page 4765.

The binary (or Bernoulli) distribution is the elementary distribution of a discrete random variable that can take on two values that have probabilities p and $1 - p$. Suppose the random variable is denoted Y and

$$\Pr(Y = 1) = p$$
$$\Pr(Y = 0) = 1 - p$$

The value that is associated with probability p is often termed the *event* or "success"; the complementary event is termed the *non-event* or "failure." A Bernoulli experiment is a random draw from a binary distribution and generates events with probability p.

If $Y_1, \ldots, Y_n$ are n independent Bernoulli random variables, then their sum follows a binomial distribution. In other words, if $Y_i = 1$ denotes an event (success) in the ith Bernoulli trial, a binomial random variable is the number of events (successes) in n independent Bernoulli trials. If you use the events/trials syntax in the MODEL statement and you specify the DISTRIBUTION=BINOMIAL option, the HPGENSELECT procedure fits the model as if the data had arisen from a binomial distribution. For example, the following statements fit a binomial regression model that has regressors x1 and x2. The variables e and t represent the events and trials, respectively, for the binomial distribution:

```
proc hpgenselect;
   model e/t = x1 x2 / distribution=Binomial;
run;
```

If the events/trials syntax is used, then both variables must be numeric and the value of the events variable cannot be less than 0 or exceed the value of the trials variable. A "Response Profile" table is not produced for binomial data, because the response variable is not subject to levelization.

The multinomial distribution is a generalization of the binary distribution and allows for more than two outcome categories. Because there are more than two possible outcomes for the multinomial distribution, the terminology of "successes," "failures," "events," and "non-events" no longer applies. For multinomial data, these outcomes are generically referred to as "categories" or levels.

Whenever the HPGENSELECT procedure determines that the response variable is listed in a CLASS statement and has more than two levels (unless the events/trials syntax is used), the procedure fits the model as if the data had arisen from a multinomial distribution. By default, it is then assumed that the response categories are ordered and a cumulative link model is fit by applying the default or specified link function. If the response categories are unordered, then you should fit a generalized logit model by choosing LINK=GLOGIT in the MODEL statement.

If the response variable is not listed in a CLASS statement and a response distribution is not specified in a DISTRIBUTION= option, then a normal distribution that uses the default or specified link function is assumed.

Response Probability Distribution Functions

Binary Distribution

$$
\begin{aligned}
f(y) &= \begin{cases} p & \text{for } y = 1 \\ 1-p & \text{for } y = 0 \end{cases} \\
\mathrm{E}(Y) &= p \\
\mathrm{Var}(Y) &= p(1-p)
\end{aligned}
$$

Binomial Distribution

$$
\begin{aligned}
f(y) &= \binom{n}{r} \mu^r (1-\mu)^{n-r} \quad \text{for } y = \frac{r}{n},\ r = 0, 1, 2, \ldots, n \\
\mathrm{E}(Y) &= \mu \\
\mathrm{Var}(Y) &= \frac{\mu(1-\mu)}{n}
\end{aligned}
$$

Gamma Distribution

$$
\begin{aligned}
f(y) &= \frac{1}{\Gamma(\nu)y}\left(\frac{y\nu}{\mu}\right)^{\nu}\exp\left(-\frac{y\nu}{\mu}\right) \quad \text{for } 0 < y < \infty \\
\phi &= \frac{1}{\nu} \\
\mathrm{E}(Y) &= \mu \\
\mathrm{Var}(Y) &= \frac{\mu^2}{\nu}
\end{aligned}
$$

For the gamma distribution, $\nu = \frac{1}{\phi}$ is the estimated dispersion parameter that is displayed in the output. The parameter ν is also sometimes called the gamma index parameter.

Inverse Gaussian Distribution

$$
\begin{aligned}
f(y) &= \frac{1}{\sqrt{2\pi y^3}\sigma}\exp\left[-\frac{1}{2y}\left(\frac{y-\mu}{\mu\sigma}\right)^2\right] \quad \text{for } 0 < y < \infty \\
\phi &= \sigma^2 \\
\mathrm{Var}(Y) &= \phi\mu^3
\end{aligned}
$$

Multinomial Distribution

$$
f(y_1, y_2, \ldots, y_k) = \frac{m!}{y_1!y_2!\cdots y_k!}p_1^{y_1}p_2^{y_2}\cdots p_k^{y_k}
$$

Negative Binomial Distribution

$$
\begin{aligned}
f(y) &= \frac{\Gamma(y+1/k)}{\Gamma(y+1)\Gamma(1/k)}\frac{(k\mu)^y}{(1+k\mu)^{y+1/k}} \quad \text{for } y = 0, 1, 2, \ldots \\
\phi &= k \\
\mathrm{E}(Y) &= \mu \\
\mathrm{Var}(Y) &= \mu + \phi\mu^2
\end{aligned}
$$

For the negative binomial distribution, k is the estimated dispersion parameter that is displayed in the output.

Normal Distribution

$$
\begin{aligned}
f(y) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{y-\mu}{\sigma}\right)^2\right] \quad \text{for } -\infty < y < \infty \\
\phi &= \sigma^2 \\
\mathrm{E}(Y) &= \mu \\
\mathrm{Var}(Y) &= \phi
\end{aligned}
$$

Poisson Distribution

$$
\begin{aligned}
f(y) &= \frac{\mu^y \mathrm{e}^{-\mu}}{y!} \quad \text{for } y = 0, 1, 2, \ldots \\
\mathrm{E}(Y) &= \mu \\
\mathrm{Var}(Y) &= \mu
\end{aligned}
$$

Tweedie Distribution

The Tweedie model is a generalized linear model from the exponential family. The Tweedie distribution is characterized by three parameters: the mean parameter μ, the dispersion ϕ, and the power p. The variance of the distribution is $\phi\mu^p$. For values of p in the range $1 < p < 2$, a Tweedie random variable can be represented as a Poisson sum of gamma distributed random variables. That is,

$$
Y = \sum_{i=1}^{N} Y_i
$$

where N has a Poisson distribution that has mean $\lambda = \frac{\mu^{2-p}}{\phi(2-p)}$ and the Y_is have independent, identical gamma distributions, each of which has an expected value $\mathrm{E}(Y_i) = \phi(2-p)\mu^{p-1}$ and an index parameter $\nu_i = \frac{2-p}{p-1}$.

In this case, Y has a discrete mass at 0, $\Pr(Y = 0) = \Pr(N = 0) = \exp(-\lambda)$, and the probability density of Y $f(y)$ is represented by an infinite series for $y > 0$. The HPGENSELECT procedure restricts the power parameter to satisfy $1.1 <= p$ for numerical stability in model fitting. The Tweedie distribution does not have a general closed form representation for all values of p. It can be characterized in terms of the distribution mean parameter μ, dispersion parameter ϕ, and power parameter p. For more information about the Tweedie distribution, see Frees (2010).

The distribution mean and variance are given by:

$$
\begin{aligned}
\mathrm{E}(Y) &= \mu \\
\mathrm{Var}(Y) &= \phi\mu^p
\end{aligned}
$$

Zero-Inflated Negative Binomial Distribution

$$
\begin{aligned}
f(y) &= \begin{cases} \omega + (1-\omega)(1+k\lambda)^{-\frac{1}{k}} & \text{for } y = 0 \\ (1-\omega)\frac{\Gamma(y+1/k)}{\Gamma(y+1)\Gamma(1/k)}\frac{(k\lambda)^y}{(1+k\lambda)^{y+1/k}} & \text{for } y = 1, 2, \ldots \end{cases} \\
\phi &= k \\
\mu = \mathrm{E}(Y) &= (1-\omega)\lambda \\
\mathrm{Var}(Y) &= (1-\omega)\lambda(1+\omega\lambda+k\lambda) \\
&= \mu + \left(\frac{\omega}{1-\omega} + \frac{k}{1-\omega}\right)\mu^2
\end{aligned}
$$

For the zero-inflated negative binomial distribution, k is the estimated dispersion parameter that is displayed in the output.

Zero-Inflated Poisson Distribution

$$
\begin{aligned}
f(y) &= \begin{cases} \omega + (1-\omega)\mathrm{e}^{-\lambda} & \text{for } y = 0 \\ (1-\omega)\frac{\lambda^y \mathrm{e}^{-\lambda}}{y!} & \text{for } y = 1, 2, \ldots \end{cases} \\
\mu = \mathrm{E}(Y) &= (1-\omega)\lambda \\
\mathrm{Var}(Y) &= (1-\omega)\lambda(1+\omega\lambda) \\
&= \mu + \frac{\omega}{1-\omega}\mu^2
\end{aligned}
$$

Log-Likelihood Functions

The HPGENSELECT procedure forms the log-likelihood functions of the various models as

$$L(\boldsymbol{\mu}; \mathbf{y}) = \sum_{i=1}^{n} f_i \, l(\mu_i; y_i, w_i)$$

where $l(\mu_i; y_i, w_i)$ is the log-likelihood contribution of the ith observation that has weight w_i, and f_i is the value of the frequency variable. For the determination of w_i and f_i, see the WEIGHT and FREQ statements. The individual log likelihood contributions for the various distributions are as follows.

In the following, the mean parameter μ_i for each observation i is related to the regression parameters $\boldsymbol{\beta}_i$ through the linear predictor $\eta_i = \mathbf{x}_i'\boldsymbol{\beta}$ by

$$\mu_i = g^{-1}(\eta_i)$$

where g is the link function.

There are two link functions and linear predictors that are associated with zero-inflated Poisson and zero-inflated negative binomial distributions: one for the zero-inflation probability ω, and another for the parameter

λ, which is the Poisson or negative binomial mean if there is no zero-inflation. Each of these parameters is related to regression parameters through an individual link function,

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\kappa_i &= \mathbf{z}_i'\boldsymbol{\gamma} \\
\lambda_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
\omega_i(\boldsymbol{\gamma}) &= h^{-1}(\kappa_i)
\end{aligned}$$

where h is one of the following link functions that are associated with binary data: complementary log-log, log-log, logit, or probit. These link functions are also shown in Table 59.9.

Binary Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith binary observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i) &= y_i \log\{\mu_i\} + (1 - y_i)\log\{1 - \mu_i\}
\end{aligned}$$

Here, μ_i is the probability of an event, and the variable y_i takes on the value 1 for an event and the value 0 for a non-event. The inverse link function $g^{-1}(\cdot)$ maps from the scale of the linear predictor η_i to the scale of the mean. For example, for the logit link (the default),

$$\mu_i(\boldsymbol{\beta}) = \frac{\exp\{\eta_i\}}{1 + \exp\{\eta_i\}}$$

You can control which binary outcome in your data is modeled as the event by specifying the *response-options* in the MODEL statement, and you can choose the link function by specifying the LINK= option in the MODEL statement.

If a WEIGHT statement is specified and w_i denotes the weight for the current observation, the log-likelihood function is computed as

$$l(\mu_i(\boldsymbol{\beta}); y_i, w_i) = w_i l(\mu_i(\boldsymbol{\beta}); y_i)$$

Binomial Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith binomial observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i, w_i) &= w_i\left(y_i \log\{\mu_i\} + (n_i - y_i)\log\{1 - \mu_i\}\right) \\
&\quad + w_i\left(\log\{\Gamma(n_i + 1)\} - \log\{\Gamma(y_i + 1)\} - \log\{\Gamma(n_i - y_i + 1)\}\right)
\end{aligned}$$

where y_i and n_i are the values of the events and trials of the ith observation, respectively. μ_i measures the probability of events (successes) in the underlying Bernoulli distribution whose aggregate follows the binomial distribution.

Gamma Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i, w_i) &= \frac{w_i}{\phi}\log\left(\frac{w_i y_i}{\phi\mu_i}\right) - \frac{w_i y_i}{\phi\mu_i} - \log(y_i) - \log\left(\Gamma\left(\frac{w_i}{\phi}\right)\right)
\end{aligned}$$

For the gamma distribution, $\nu = \frac{1}{\phi}$ is the estimated dispersion parameter that is displayed in the output.

Inverse Gaussian Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i, w_i) &= -\frac{1}{2}\left[\frac{w_i(y_i-\mu_i)^2}{y_i\mu_i^2\phi} + \log\left(\frac{\phi y_i^3}{w_i}\right) + \log(2\pi)\right]
\end{aligned}$$

where ϕ is the dispersion parameter.

Multinomial Distribution

The multinomial distribution that is modeled by the HPGENSELECT procedure is a generalization of the binary distribution; it is the distribution of a single draw from a discrete distribution with J possible values. The log-likelihood function for the ith observation is

$$l(\boldsymbol{\mu}_i; \mathbf{y}_i, w_i) = w_i \sum_{j=1}^{J} y_{ij}\log\{\mu_{ij}\}$$

In this expression, J denotes the number of response categories (the number of possible outcomes) and μ_{ij} is the probability that the ith observation takes on the response value that is associated with category j. The category probabilities must satisfy

$$\sum_{j=1}^{J}\mu_j = 1$$

and the constraint is satisfied by modeling $J-1$ categories. In models that have ordered response categories, the probabilities are expressed in cumulative form, so that the last category is redundant. In generalized logit models (multinomial models that have unordered categories), one category is chosen as the reference category and the linear predictor in the reference category is set to 0.

Negative Binomial Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i, w_i) &= y_i \log\left(\frac{k\mu_i}{w_i}\right) - (y_i + w_i/k)\log\left(1 + \frac{k\mu_i}{w_i}\right) + \log\left(\frac{\Gamma(y_i + w_i/k)}{\Gamma(y_i+1)\Gamma(w_i/k)}\right)
\end{aligned}$$

where k is the negative binomial dispersion parameter that is displayed in the output.

Normal Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i, w_i) &= -\frac{1}{2}\left[\frac{w_i(y_i - \mu_i)^2}{\phi} + \log\left(\frac{\phi}{w_i}\right) + \log(2\pi)\right]
\end{aligned}$$

where ϕ is the dispersion parameter.

Poisson Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i, w_i) &= w_i[y_i \log(\mu_i) - \mu_i - \log(y_i!)]
\end{aligned}$$

Tweedie Distribution

The Tweedie distribution does not in general have a closed form log-likelihood function in terms of the mean, dispersion, and power parameters. The form of the log likelihood is

$$L(\boldsymbol{\mu}; \mathbf{y}) = \sum_{i=1}^{n} f_i\, l(\mu_i; y_i, w_i)$$

where

$$l(\mu_i, y_i, w_i) = \log(f(y_i; \mu_i, p, \frac{\phi}{w_i}))$$

and $f(y, \mu, p, \phi)$ is the Tweedie probability distribution, which is described in the section "Tweedie Distribution" on page 4764. Evaluation of the Tweedie log likelihood for model fitting is performed numerically as described in Dunn and Smyth (2005, 2008).

Quasi-likelihood

The extended quasi-likelihood (EQL) is constructed according to the definition of McCullagh and Nelder (1989, Chapter 9) as

$$Q_p(\mathbf{y}, \boldsymbol{\mu}, \phi, p) = \sum_i q(y_i, \mu_i, \phi, p)$$

where the contribution from an observation is

$$q(y_i, \mu_i, \phi, p) = -0.5 \log(2\pi \frac{\phi}{w_i} y_i^p) - w_i \left(\frac{y_i^{2-p} - (2-p) y_i \mu_i^{1-p} + (1-p)\mu_i^{2-p}}{(1-p)(1-p)} \right) / \phi$$

where $1 < p < 2$. This EQL is used in computing initial values for the iterative maximization of the Tweedie log likelihood, as specified using the OPTMETHOD= Tweedie option in Table 59.5. If you specify the OPTMETHOD=EQL *Tweedie-optimization-option* in Table 59.5, then the parameter estimates are computed by using the EQL instead of the log likelihood.

Zero-Inflated Negative Binomial Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\lambda_i(\boldsymbol{\beta}), \omega_i(\boldsymbol{\gamma}); y_i)$ for the ith observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i' \boldsymbol{\beta} \\
\kappa_i &= \mathbf{z}_i' \boldsymbol{\gamma} \\
\lambda_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
\omega_i(\boldsymbol{\gamma}) &= h^{-1}(\kappa_i) \\
l(\mu_i(\boldsymbol{\beta}), \omega_i(\boldsymbol{\gamma}); y_i, w_i) &= \begin{cases} \log[\omega_i + (1-\omega_i)(1 + \frac{k}{w_i}\lambda)^{-\frac{w_i}{k}}] & y_i = 0 \\ \\ \log(1-\omega_i) + y_i \log\left(\frac{k\lambda}{w_i}\right) & \\ -(y_i + \frac{w_i}{k}) \log\left(1 + \frac{k\lambda}{w_i}\right) & \\ + \log\left(\frac{\Gamma(y_i + \frac{w_i}{k})}{\Gamma(y_i+1)\Gamma(\frac{w_i}{k})}\right) & y_i > 0 \end{cases}
\end{aligned}$$

where k is the zero-inflated negative binomial dispersion parameter that is displayed in the output.

Zero-Inflated Poisson Distribution

The HPGENSELECT procedure computes the log-likelihood function $l(\lambda_i(\boldsymbol{\beta}), \omega_i(\boldsymbol{\gamma}); y_i)$ for the ith observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i' \boldsymbol{\beta} \\
\kappa_i &= \mathbf{z}_i' \boldsymbol{\gamma} \\
\lambda_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
\omega_i(\boldsymbol{\gamma}) &= h^{-1}(\kappa_i) \\
l(\mu_i(\boldsymbol{\beta}), \omega_i(\boldsymbol{\gamma}); y_i, w_i) &= \begin{cases} w_i \log[\omega_i + (1-\omega_i)\exp(-\lambda_i)] & y_i = 0 \\ \\ w_i[\log(1-\omega_i) + y_i \log(\lambda_i) - \lambda_i - \log(y_i!)] & y_i > 0 \end{cases}
\end{aligned}$$

The LASSO Method of Model Selection

LASSO Selection

The HPGENSELECT procedure implements the group LASSO method, which is described in the section "Group LASSO Selection" on page 4770. This section provides some background about the LASSO method that you need in order to understand the group LASSO method.

LASSO (least absolute shrinkage and selection operator) selection arises from a constrained form of ordinary least squares regression in which the sum of the absolute values of the regression coefficients is constrained to be smaller than a specified parameter. More precisely, let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m)$ denote the matrix of covariates, and let $\mathbf{y}$ denote the response. Then for a given parameter t, the LASSO regression coefficients $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the constrained least squares problem

$$\min ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \qquad \text{subject to} \qquad \sum_{j=1}^{m} |\beta_j| \le t$$

For generalized linear models, the LASSO regression coefficients $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_m)$ are the solution to the constrained optimization problem

$$\min\{-L(\boldsymbol{\mu}; \mathbf{y})\} \qquad \text{subject to} \qquad \sum_{j=1}^{m} |\beta_j| \le t$$

where L is the log-likelihood function defined in the section "Log-Likelihood Functions" on page 4765.

Provided that the LASSO parameter t is small enough, some of the regression coefficients will be exactly zero. Hence, you can view the LASSO method as selecting a subset of the regression coefficients for each LASSO parameter. By increasing the LASSO parameter in discrete steps, you obtain a sequence of regression coefficients for which the nonzero coefficients at each step correspond to selected parameters. For more information about the LASSO method, see, for example, Hastie, Tibshirani, and Friedman (2009).

Group LASSO Selection

The group LASSO method, proposed by Yuan and Lin (2006), is a variant of LASSO that is specifically designed for models defined in terms of effects that have multiple degrees of freedom, such as the main effects of CLASS variables, and interactions between CLASS variables. If all effects in the model are continuous, then the group LASSO method is the same as the LASSO method.

Recall that LASSO selection depends on solving a constrained optimization problem of the form

$$\min\{-L(\boldsymbol{\mu}; \mathbf{y})\} \qquad \text{subject to} \qquad \sum_{j=1}^{m} |\beta_j| \le t$$

In this formulation, individual parameters can be included or excluded from the model independently, subject only to the overall constraint. In contrast, the group LASSO method uses a constraint that forces all parameters corresponding to the same effect to be included or excluded simultaneously. For a model that

has k effects, let β_{G_j} be the group of linear coefficients that correspond to effect j in the model. Then group LASSO depends on solving a constrained optimization problem of the form

$$\min\{-L(\boldsymbol{\mu};\mathbf{y})\} \qquad \text{subject to} \qquad \sum_{j=1}^{k}\sqrt{|G_j|}||\beta_{G_j}|| \le t$$

where $|G_j|$ is the number of parameters that correspond to effect j, and $||\beta_{G_j}||$ denotes the Euclidean norm of the parameters β_{G_j},

$$||\beta_{G_j}|| = \sqrt{\sum_{i=1}^{G_j}\beta_i^2}$$

That is, instead of constraining the sum of the absolute value of individual parameters, group LASSO constrains the Euclidean norm of groups of parameters, where groups are defined by effects.

You can write the group LASSO method in the equivalent Lagrangian form, which is an example of a penalized log-likelihood function:

$$\min\{-L(\boldsymbol{\mu};\mathbf{y})\} + \lambda\sum_{j=1}^{k}\sqrt{|G_j|}||\beta_{G_j}||$$

The weight $\sqrt{|G_j|}$ was suggested by Yuan and Lin (2006) in order to take the size of the group into consideration in group LASSO.

Unlike LASSO for linear models, group LASSO does not allow a piecewise linear constant solution path as generated by a LAR algorithm. Instead, the method proposed by Nesterov (2013) is adopted to solve the Lagrangian form of the group LASSO problem that corresponds to a prespecified regularization parameter λ. Nesterov's method is known to have an optimal convergence rate for first-order black box optimization. The optimal regularization parameter is usually unknown, so the procedure uses a sequence of values. To create this sequence, the procedure first uses the data to determine a maximum regularization parameter m. This is the smallest regularization parameter that produces a model in which all regression parameters are equal to zero. The procedure then uses the sequence of regularization parameters $m\rho, m\rho^2, m\rho^3, \ldots$, where ρ is a positive value less than 1. You can specify ρ by using the LASSORHO= option in the PROC HPGENSELECT statement; the default value is $\rho = 0.8$. In the ith step of group LASSO selection, the value that is used for λ is $m\rho^i$.

A unique feature of the group LASSO method is that it does not necessarily add or remove precisely one effect at each step of the process. This is different from the forward, stepwise, and backward selection methods.

As with the other selection methods that PROC HPGENSELECT supports, you can specify a criterion to choose among the models at each step of the group LASSO algorithm by using the CHOOSE= option in the SELECTION statement. You can also specify a stopping criterion by using the STOP= option in the SELECTION statement. If you do not specify either the CHOOSE= or STOP= option, the model at the last LASSO step is chosen as the selected model, and parameter estimates are reported for this model. If you request an output data set by using an OUTPUT statement, these parameter estimates are used to compute predicted values in the output data set.

For more information, see the discussion in the section "SELECTION Statement" on page 576.

The model degrees of freedom that PROC HPGENSELECT uses at any step of the LASSO are simply the number of nonzero regression coefficients in the model at that step. Efron et al. (2004) cite empirical evidence for doing this but do not give any mathematical justification for this choice.

Some distributions involve a dispersion parameter (the parameter ϕ in the expressions for the log likelihood), and in the case of the Tweedie distribution, a power parameter. These parameters are not estimated by the LASSO optimization algorithm, and are set to either the default value or a value that you specify. You can use the MODEL statement options PHI= to set the dispersion to a fixed value and P= to set the Tweedie power parameter to a fixed value.

Using Validation and Test Data

When you have sufficient data, you can divide your data into three parts, which are called the training, validation, and test data. For a single model fit or during the model selection process, models are fit and selected based on the training data. After a model has been fit, the validation and test sets can be used to assess how the selected model generalizes on data that played no role in selecting the model. For example, Hastie, Tibshirani, and Friedman (2009) advocate using validation data in the model selection process to determine which effects to include in each step and when to terminate the selection process. PROC HPGENSELECT does not currently use validation data in this way, so the validation and test data subsets are equivalent.

You can use validation and test data to score data that were not used in fitting the model. Statistics in an output data set that is created by an OUTPUT statement are computed for validation and test data, using the model fit based on the training data. You can use the ROLE option in an OUTPUT statement to add a variable (named Role by default) to an output data set to indicate the role played by each observation.

You use a PARTITION statement to logically divide the DATA= data set into separate roles. You can specify the fractions of the data that you want to reserve as test data and validation data. For example, the following statements randomly divide the inData data set, reserving 50% for training and 25% each for validation and testing:

```
proc hpgenselect data=inData;
  partition fraction(test=0.25 validate=0.25);
  ...
run;
```

In some cases you might need to exercise more control over the partitioning of the input data set. You can do this by naming both a variable in the input data set and a formatted value of that variable that corresponds to each role. For example, the following statements assign roles to the observations in the inData data set based on the value of the variable group in that data set. Observations whose value of Group is `'group 1'` are assigned for testing, and those whose value is `'group 2'` are assigned to training. All other observations are ignored.

```
proc hpgenselect data=inData;
  partition roleVar=Group(test='group 1' train='group 2')
  ...
run;
```

When you have reserved observations for training, validation, and testing, a model that is fit on the training data is scored on the validation and test data, and fit statistics, including the average squared error (ASE),

are computed separately for each of these subsets. The ASE for each data role is the sum of the squared differences between the responses and the predictions for observations in that role divided by the number of observations in that role.

Computational Method: Multithreading

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to sequential (single-threaded) execution.

The number of threads spawned by the HPGENSELECT procedure is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the number of CPUs in the CPUCOUNT= SAS system option. For example, if you specify the following statement, the HPGENSELECT procedure determines threading as if it executed on a system that has four CPUs, regardless of the actual CPU count:

  ```
  options cpucount=4;
  ```

- You can specify the NTHREADS= option in the PERFORMANCE statement to control the number of threads. This specification overrides the CPUCOUNT= system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads per machine is displayed in the “Dimensions” table, which is part of the default output. The HPGENSELECT procedure allocates one thread per CPU by default.

The tasks that are multithreaded by the HPGENSELECT procedure are primarily defined by dividing the data that are processed on a single machine among the threads—that is, the HPGENSELECT procedure implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and PROC HPGENSELECT is running with four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following:

- variable levelization
- effect levelization
- formation of the initial crossproducts matrix
- formation of approximate Hessian matrices for candidate evaluation during model selection
- objective function calculation
- gradient calculation
- Hessian calculation
- scoring of observations

In addition, operations on matrices such as sweeps can be multithreaded provided that the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Choosing an Optimization Algorithm

First- or Second-Order Algorithms

The factors that affect how you choose an optimization technique for a particular problem are complex. Although the default method works well for most problems, you might occasionally benefit from trying several different algorithms.

For many optimization problems, computing the gradient takes more computer time than computing the function value. Computing the Hessian sometimes takes *much* more computer time and memory than computing the gradient, especially when there are many decision variables. Unfortunately, optimization techniques that do not use some kind of Hessian approximation usually require many more iterations than techniques that do use a Hessian matrix; as a result, the total run time of these techniques is often longer. Techniques that do not use the Hessian also tend to be less reliable. For example, they can terminate more easily at stationary points than at global optima.

Table 59.10 shows which derivatives are required for each optimization technique.

Table 59.10 Derivatives Required

Algorithm	First-Order	Second-Order
TRUREG	x	x
NEWRAP	x	x
NRRIDG	x	x
QUANEW	x	-
DBLDOG	x	-
CONGRA	x	-
NMSIMP	-	-

The second-derivative methods TRUREG, NEWRAP, and NRRIDG are best for small problems for which the Hessian matrix is not expensive to compute. Sometimes the NRRIDG algorithm can be faster than the TRUREG algorithm, but TRUREG can be more stable. The NRRIDG algorithm requires only one matrix with $p(p+1)/2$ double words; TRUREG and NEWRAP require two such matrices. Here, p denotes the number of parameters in the optimization.

The first-derivative methods QUANEW and DBLDOG are best for medium-sized problems for which the objective function and the gradient can be evaluated much faster than the Hessian. In general, the QUANEW and DBLDOG algorithms require more iterations than TRUREG, NRRIDG, and NEWRAP, but each iteration can be much faster. The QUANEW and DBLDOG algorithms require only the gradient to update an approximate Hessian, and they require slightly less memory than TRUREG or NEWRAP.

The first-derivative method CONGRA is best for large problems for which the objective function and the gradient can be computed much faster than the Hessian and for which too much memory is required to store the (approximate) Hessian. In general, the CONGRA algorithm requires more iterations than QUANEW or

DBLDOG, but each iteration can be much faster. Because CONGRA requires only a factor of p double-word memory, many large applications can be solved only by CONGRA.

The no-derivative method NMSIMP is best for small problems for which derivatives are not continuous or are very difficult to compute.

Each optimization method uses one or more convergence criteria that determine when it has converged. An algorithm is considered to have converged when any one of the convergence criteria is satisfied. For example, under the default settings, the QUANEW algorithm converges if ABSGCONV <1E–5, FCONV $< 2 \times \epsilon$, or GCONV <1E–8.

By default, the HPGENSELECT procedure applies the NRRIDG algorithm because it can take advantage of multithreading in Hessian computations and inversions. If the number of parameters becomes large, specifying the TECHNIQUE=QUANEW option (which is a first-order method with good overall properties), is recommended.

Algorithm Descriptions

The following subsections provide details about each optimization technique and follow the same order as Table 59.10.

Trust Region Optimization (TRUREG)

The trust region method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function $f(\boldsymbol{\psi})$ have continuous first- and second-order derivatives inside the feasible region.

The trust region method iteratively optimizes a quadratic approximation to the nonlinear objective function within a hyperelliptic trust region with radius Δ that constrains the step size that corresponds to the quality of the quadratic approximation. The trust region method is implemented based on Dennis, Gay, and Welsch (1981); Gay (1983); Moré and Sorensen (1983).

The trust region method performs well for small- to medium-sized problems, and it does not need many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the dual quasi-Newton or conjugate gradient algorithms might be more efficient.

Newton-Raphson Optimization with Line Search (NEWRAP)

The NEWRAP technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

If second-order derivatives are computed efficiently and precisely, the NEWRAP method can perform well for medium-sized to large problems, and it does not need many function, gradient, and Hessian calls.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. Otherwise, a combination of ridging and line search is performed to compute successful steps. If the Hessian is not positive-definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive-definite (Eskow and Schnabel 1991).

In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The line-search method uses quadratic interpolation and cubic extrapolation.

Newton-Raphson Ridge Optimization (NRRIDG)

The NRRIDG technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. If at least one of these two conditions is not satisfied, a multiple of the identity matrix is added to the Hessian matrix.

Because the NRRIDG technique uses an orthogonal decomposition of the approximate Hessian, each iteration of NRRIDG can be slower than an iteration of the NEWRAP technique, which works with a Cholesky decomposition. However, NRRIDG usually requires fewer iterations than NEWRAP.

The NRRIDG method performs well for small- to medium-sized problems, and it does not require many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the dual quasi-Newton or conjugate gradient algorithms might be more efficient.

Quasi-Newton Optimization (QUANEW)

The dual quasi-Newton method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$, and it does not need to compute second-order derivatives because they are approximated. It works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. However, in general the QUANEW technique requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which compute second-order derivatives. The QUANEW technique provides an appropriate balance between the speed and stability that are required for most generalized linear model applications.

The QUANEW technique that is implemented by the HPGENSELECT procedure is the dual quasi-Newton algorithm, which updates the Cholesky factor of an approximate Hessian.

In each iteration, a line search is performed along the search direction to find an approximate optimum. The line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions (Fletcher 1987). One of the Goldstein conditions can be violated if the feasible region defines an upper limit of the step size. Violating the left-side Goldstein condition can affect the positive-definiteness of the quasi-Newton update. In that case, either the update is skipped or the iterations are restarted by using an identity matrix, resulting in the steepest descent or ascent search direction.

Double-Dogleg Optimization (DBLDOG)

The double-dogleg optimization method combines the ideas of the quasi-Newton and trust region methods. In each iteration, the double-dogleg algorithm computes the step $\mathbf{s}^{(k)}$ as the linear combination of the steepest descent or ascent search direction $\mathbf{s}_1^{(k)}$ and a quasi-Newton search direction $\mathbf{s}_2^{(k)}$:

$$\mathbf{s}^{(k)} = \alpha_1 \mathbf{s}_1^{(k)} + \alpha_2 \mathbf{s}_2^{(k)}$$

The step is requested to remain within a prespecified trust region radius (Fletcher 1987, p. 107). Thus, the DBLDOG subroutine uses the dual quasi-Newton update but does not perform a line search.

The double-dogleg optimization technique works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. The implementation is based on Dennis and Mei (1979); Gay (1983), but it is extended for dealing with boundary and linear constraints. The DBLDOG technique generally requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which require second-order derivatives; however, each of the DBLDOG iterations is computationally cheap. Furthermore, the DBLDOG technique requires only gradient calls for the update of the Cholesky factor of an approximate Hessian.

Conjugate Gradient Optimization (CONGRA)

Second-order derivatives are not required by the CONGRA algorithm and are not even approximated. The CONGRA algorithm can be expensive in function and gradient calls, but it requires only $O(p)$ memory for unconstrained optimization. In general, the algorithm must perform many iterations to obtain a precise solution, but each of the CONGRA iterations is computationally cheap.

The CONGRA algorithm should be used for optimization problems that have large p. For the unconstrained or boundary-constrained case, the CONGRA algorithm requires only $O(p)$ bytes of working memory, whereas all other optimization methods require order $O(p^2)$ bytes of working memory. During p successive iterations, uninterrupted by restarts or changes in the working set, the CONGRA algorithm computes a cycle of p conjugate search directions. In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit for the step size.

Nelder-Mead Simplex Optimization (NMSIMP)

The Nelder-Mead simplex method does not use any derivatives and does not assume that the objective function has continuous derivatives. The objective function itself needs to be continuous. This technique is quite expensive in the number of function calls, and it might be unable to generate precise results for $p \gg 40$.

The original Nelder-Mead simplex algorithm is implemented and extended to boundary constraints. This algorithm does not compute the objective for infeasible points, but it changes the shape of the simplex adapting to the nonlinearities of the objective function. This change contributes to an increased speed of convergence and uses a special termination criterion.

Displayed Output

The following sections describe the output that PROC HPGENSELECT produces by default. The output is organized into various tables, which are discussed in the order of their appearance.

Performance Information

The “Performance Information” table is produced by default, and displays the number of threads used for computation.

If you specify the DETAILS option in the PERFORMANCE statement, the procedure also produces a “Timing” table in which elapsed times (absolute and relative) for the main tasks of the procedure are displayed.

Model Information

The “Model Information” table displays basic information about the model, such as the response variable, frequency variable, link function, and the model category that the HPGENSELECT procedure determined based on your input and options. The “Model Information” table also displays the distribution of the data that is assumed by the HPGENSELECT procedure. For information about how the procedure determines the response distribution, see the section “Response Distributions” on page 4761.

Class Level Information

The "Class Level Information" table lists the levels of every variable that is specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress the "Class Level Information" table completely or partially by specifying the NOCLPRINT= option in the PROC HPGENSELECT statement.

If the classification variables use reference parameterization, the "Class Level Information" table also displays the reference value for each variable.

Number of Observations

The "Number of Observations" table displays the number of observations that are read from the input data set and the number of observations that are used in the analysis. If a FREQ statement is present, the sum of the frequencies read and used is displayed. If the events/trials syntax is used, the number of events and trials is also displayed. If a PARTITION statement is specified, the table displays the values for each role.

Response Profile

The "Response Profile" table displays the ordered value from which the HPGENSELECT procedure determines the probability being modeled as an event in binary models and the ordering of categories in multinomial models. For each response category level, the frequency that is used in the analysis is reported. You can affect the ordering of the response values by specifying *response-options* in the MODEL statement. For binary and generalized logit models, the note that follows the "Response Profile" table indicates which outcome is modeled as the event in binary models and which value serves as the reference category.

The "Response Profile" table is not produced for binomial data. You can find information about the number of events and trials in the "Number of Observations" table. If a PARTITION statement is specified, the table displays the values for each role.

Entry and Removal Candidates

When you specify the DETAILS=ALL or DETAILS=STEPS option in the SELECTION statement, the HPGENSELECT procedure produces "Entry Candidates" and "Removal Candidates" tables that display the effect names and the values of the criterion that is used to select entering or departing effects at each step of the selection process. The effects are displayed in sorted order from best to worst of the selection criterion.

Selection Information

When you specify the SELECTION statement, the HPGENSELECT procedure produces by default a series of tables that have information about the model selection. The "Selection Information" table informs you about the model selection method, selection and stop criteria, and other parameters that govern the selection. You can suppress this table by specifying DETAILS=NONE in the SELECTION statement.

Selection Summary

When you specify the SELECTION statement, the HPGENSELECT procedure produces the "Selection Summary" table, which contains information about which effects were entered into or removed from the model at the steps of the model selection process. The p-value for the score chi-square test that led to the removal or entry decision is also displayed. You can request further details about the model selection steps by specifying DETAILS=STEPS or DETAILS=ALL in the SELECTION statement. You can suppress the display of the "Selection Summary" table by specifying DETAILS=NONE in the SELECTION statement.

Selection Details

When you specify the DETAILS=ALL option in the SELECTION statement, the HPGENSELECT procedure produces the "Selection Details" table, which contains information about which effects were entered into or removed from the model at the steps of the model selection process. When you specify METHOD=FORWARD, BACKWARD, or STEPWISE, the p-value and the chi-square test statistic that led to the removal or entry decision are also displayed. Fit statistics for the model at the steps are also displayed. When you specify METHOD=LASSO, fit statistics for the model at the steps are displayed.

Stop Reason

When you specify the SELECTION statement, the HPGENSELECT procedure produces a simple table that tells you why model selection stopped.

Selection Reason

When you specify the SELECTION statement, the HPGENSELECT procedure produces a simple table that tells you why the final model was selected.

Selected Effects

When you specify the SELECTION statement, the HPGENSELECT procedure produces a simple table that tells you which effects were selected to be included in the final model.

Iteration History

For each iteration of the optimization, the "Iteration History" table displays the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element. The objective function used in the optimization in the HPGENSELECT procedure is normalized by default to enable comparisons across data sets that have different sampling intensity. You can control normalization by specifying the NORMALIZE= option in the PROC HPGENSELECT statement.

If you specify the ITDETAILS option in the PROC HPGENSELECT statement, information about the parameter estimates and gradients in the course of the optimization is added to the "Iteration History" table. To generate the history from a model selection process, specify the ITSELECT option.

Convergence Status

The convergence status table is a small ODS table that follows the "Iteration History" table in the default output. In the listing it appears as a message that indicates whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the convergence status table to an output data set, a numeric Status variable is added that enables you to programmatically assess convergence. The values of the Status variable encode the following:

0 Convergence was achieved, or an optimization was not performed because TECHNIQUE=NONE is specified.

1 The objective function could not be improved.

2 Convergence was not achieved because of a user interrupt or because a limit (such as the maximum number of iterations or the maximum number of function evaluations) was reached. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC HPGENSELECT statement.

3 Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Dimensions

The "Dimensions" table displays size measures that are derived from the model and the environment. It displays the number of effects in the model, the number of columns in the design matrix, and the number of parameters for which maximum likelihood estimates are computed.

Optimization Stage Details

The "Optimization Stage Details" table displays the optimization stages that are used to fit Tweedie models. The type of optimization, the percentage of observations used, and the number of observations used are displayed for each stage.

Fit Statistics

The "Fit Statistics" table displays a variety of likelihood-based measures of fit. All statistics are presented in "smaller is better" form.

The calculation of the information criteria uses the following formulas, where p denotes the number of effective parameters, f denotes the number of frequencies used, and l is the log likelihood evaluated at the converged estimates:

$$\mathrm{AIC} = -2l + 2p$$

$$\mathrm{AICC} = \begin{cases} -2l + 2pf/(f-p-1) & \text{when } f > p+2 \\ -2l + 2p(p+2) & \text{otherwise} \end{cases}$$

$$\mathrm{BIC} = -2l + p\log(f)$$

If no FREQ statement is given, f equals n, the number of observations used.

If a PARTITION statement is specified, the table displays the values for each role. In addition, the average squared error (ASE) is computed separately for each role. The ASE for each data role is the sum of the

squared differences between the responses and the predictions for observations in that role divided by the number of observations in that role.

The values displayed in the "Fit Statistics" table are not based on a normalized log-likelihood function.

Parameter Estimates

The "Parameter Estimates" table displays the parameter estimates, their estimated (asymptotic) standard errors, chi-square statistics, and *p*-values for the hypothesis that the parameter is 0.

If you request confidence intervals by specifying the CL option in the MODEL statement, confidence limits for regression parameters are produced for the estimate on the linear scale. Confidence limits for the dispersion parameter of those distributions that possess a dispersion parameter are produced on the log scale, because the dispersion must be greater than 0. Similarly, confidence limits for the power parameter of the Tweedie distribution are produced on the log scale.

Parameter Estimates Correlation Matrix

When you specify the CORR option in the PROC HPGENSELECT statement, the correlation matrix of the parameter estimates is displayed.

Parameter Estimates Covariance Matrix

When you specify the COV option in the PROC HPGENSELECT statement, the covariance matrix of the parameter estimates is displayed. The covariance matrix is computed as the inverse of the negative of the matrix of second derivatives of the log-likelihood function with respect to the model parameters (the Hessian matrix), evaluated at the parameter estimates.

Zero-Inflation Parameter Estimates

The parameter estimates for zero-inflation probability in zero-inflated models, their estimated (asymptotic) standard errors, chi-square statistics, and *p*-values for the hypothesis that the parameter is 0 are presented in the "Parameter Estimates" table. If you request confidence intervals by specifying the CL option in the MODEL statement, confidence limits for regression parameters are produced for the estimate on the linear scale.

ODS Table Names

Each table created by the HPGENSELECT procedure has a name that is associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 59.11.

Table 59.11 ODS Tables Produced by PROC HPGENSELECT

Table Name	Description	Required Statement and Option
ClassLevels	Level information from the CLASS statement	CLASS
ConvergenceStatus	Status of optimization at conclusion of optimization	Default output

Table 59.11 *continued*

Table Name	Description	Required Statement / Option
CorrelationMatrix	Correlation matrix of parameter estimates	PROC HPGENSELECT CORR
CovarianceMatrix	Covariance matrix of parameter estimates	PROC HPGENSELECT COV
Dimensions	Model dimensions	Default output
EntryCandidates	Candidates for entry at step	SELECTION DETAILS=ALL | STEPS
FitStatistics	Fit statistics	Default output
IterHistory	Iteration history	PROC HPGENSELECT ITDETAILS or PROC HPGENSELECT ITSUMMARY or PROC HPGENSELECT ITSELECT
LassoSelectionDetails	Details about model selection by LASSO, including fit statistics by step	SELECTION DETAILS=ALL
ModelInfo	Information about the modeling environment	Default output
NObs	Number of observations read and used, and number of events and trials, if applicable	Default output
OptimizationStages	Optimization stages that are used to fit Tweedie models	MODEL DISTRIBUTION=TWEEDIE
ParameterEstimates	Solutions for the parameter estimates that are associated with effects in MODEL statements	Default output
PerformanceInfo	Information about the high-performance computing environment	Default output
Regularization	Maximum regularization parameter used in penalized log likelihood for LASSO model selection and regularization parameter of the chosen model	SELECTION METHOD=LASSO
RemovalCandidates	Candidates for removal at step	SELECTION DETAILS=ALL | STEPS
ResponseProfile	Response categories and the category that is modeled in models for binary and multinomial data	Default output
SelectedEffects	List of effects that are selected to be included in model	SELECTION
SelectionDetails	Details about model selection, including fit statistics by step	SELECTION DETAILS=ALL

Table 59.11 *continued*

Table Name	Description	Required Statement / Option
SelectionInfo	Information about the settings for model selection	SELECTION
SelectionReason	Reason why the particular model was selected	SELECTION
SelectionSummary	Summary information about model selection steps	SELECTION
StopReason	Reason for termination of model selection	SELECTION
Timing	Absolute and relative times for tasks performed by the procedure	PERFORMANCE DETAILS
ZeroParameterEstimates	Solutions for the parameter estimates that are associated with effects in ZEROMODEL statements	ZEROMODEL

Examples: HPGENSELECT Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 59.1: Model Selection

(View the complete code for this example (hpgenex1.sas) in the example repository.)

The following HPGENSELECT statements examine the same data that is used in the section "Getting Started: HPGENSELECT Procedure" on page 4729, but they request model selection via the forward selection technique. Model effects are added in the order of their significance until no more effects make a significant improvement of the current model. The DETAILS=ALL option in the SELECTION statement requests that all tables that are related to model selection be produced.

The data set getStarted is shown in the section "Getting Started: HPGENSELECT Procedure" on page 4729. It contains 100 observations on a count response variable (Y), a continuous variable (Total) to be used in Example 59.3, and five categorical variables (C1–C5), each of which has four numerical levels.

A log-linked Poisson regression model is specified by using classification effects for variables C1–C5. The following statements request model selection by using the forward selection method:

```
proc hpgenselect data=getStarted;
   class C1-C5;
   model Y = C1-C5 / Distribution=Poisson;
   selection method=forward details=all;
run;
```

The model selection tables are shown in Output 59.1.1 through Output 59.1.3.

The "Selection Information" table in Output 59.1.1 summarizes the settings for the model selection. Effects are added to the model only if they produce a significant improvement as judged by comparing the p-value of a score test to the entry significance level (SLE), which is 0.05 by default. The forward selection stops when no effect outside the model meets this criterion.

Output 59.1.1 Selection Information

The HPGENSELECT Procedure

Selection Information	
Selection Method	Forward
Select Criterion	Significance Level
Stop Criterion	Significance Level
Effect Hierarchy Enforced	None
Entry Significance Level (SLE)	0.05
Stop Horizon	1

The "Selection Summary" table in Output 59.1.2 shows the effects that were added to the model and their significance level. Step 0 refers to the null model that contains only an intercept. In the next step, effect C2 made the most significant contribution to the model among the candidate effects ($p < 0.0001$). In step 2, the most significant contribution when adding an effect to a model that contains the intercept and C2 was made by C5. In step 3, the variable C1 ($p = 0.0496$) was added. In the subsequent step, no effect could be added to the model that would produce a p-value less than 0.05, so variable selection stops.

Output 59.1.2 Selection Summary Information

The HPGENSELECT Procedure

Selection Summary			
Step	Effect Entered	Number Effects In	p Value
0	**Intercept**	1	.
1	**C2**	2	<.0001
2	**C5**	3	<.0001
3	**C1**	4	0.0496

Selection stopped because no candidate for entry is significant at the 0.05 level.

Selected Effects: Intercept C1 C2 C5

The DETAILS=ALL option produces the "Selection Details" table, which provides fit statistics and the value of the score test chi-square statistic at each step.

Output 59.1.3 Selection Details

Selection Details								
Step	Description	Effects In Model	Chi-Square	Pr > ChiSq	-2 LogL	AIC	AICC	BIC
0	Initial Model	1			350.193	352.193	352.234	354.798
1	C2 entered	2	25.7340	<.0001	324.611	332.611	333.032	343.032
2	C5 entered	3	23.0291	<.0001	303.580	317.580	318.798	335.817
3	C1 entered	4	7.8328	0.0496	295.263	315.263	317.735	341.315

Output 59.1.4 displays information about the selected model. Notice that the –2 log likelihood value in the "Fit Statistics" table is larger than the value for the full model in Figure 59.7. This is expected because the selected model contains only a subset of the parameters. Because the selected model is more parsimonious than the full model, the information criteria AIC, AICC and BIC are smaller than in the full model, indicating a better fit.

Output 59.1.4 Fit Statistics

Fit Statistics	
-2 Log Likelihood	295.26
AIC (smaller is better)	315.26
AICC (smaller is better)	317.74
BIC (smaller is better)	341.31
Pearson Chi-Square	85.0656
Pearson Chi-Square/DF	0.9452

The parameter estimates of the selected model are given in Output 59.1.5. Notice that the effects are listed in the "Parameter Estimates" table in the order in which they were specified in the MODEL statement and not in the order in which they were added to the model.

Output 59.1.5 Parameter Estimates

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	0.775498	0.242561	10.2216	0.0014
C1 0	1	-0.211240	0.207209	1.0393	0.3080
C1 1	1	-0.685575	0.255713	7.1879	0.0073
C1 2	1	-0.127612	0.203663	0.3926	0.5309
C1 3	0	0	.	.	.
C2 0	1	0.958378	0.239731	15.9817	<.0001
C2 1	1	0.738529	0.237098	9.7024	0.0018
C2 2	1	0.211075	0.255791	0.6809	0.4093
C2 3	0	0	.	.	.
C5 0	1	-0.825545	0.214054	14.8743	0.0001
C5 1	1	-0.697611	0.202607	11.8555	0.0006
C5 2	1	-0.566706	0.213961	7.0153	0.0081
C5 3	0	0	.	.	.

Example 59.2: Modeling Binomial Data

(View the complete code for this example (hpgenex2.sas) in the example repository.)

If $Y_1, \ldots, Y_n$ are independent binary (Bernoulli) random variables that have common success probability π, then their sum is a binomial random variable. In other words, a binomial random variable that has parameters n and π can be generated as the sum of n Bernoulli(π) random experiments. The HPGENSELECT procedure uses a special syntax to express data in binomial form: the *events/trials* syntax.

Consider the following data, taken from Cox and Snell (1989, pp. 10–11), of the number, r, of ingots not ready for rolling, out of n tested, for a number of combinations of heating time and soaking time.

```
data Ingots;
   input Heat Soak r n @@;
   Obsnum= _n_;
   datalines;
7 1.0 0 10  14 1.0 0 31  27 1.0 1 56  51 1.0 3 13
7 1.7 0 17  14 1.7 0 43  27 1.7 4 44  51 1.7 0  1
7 2.2 0  7  14 2.2 2 33  27 2.2 0 21  51 2.2 0  1
7 2.8 0 12  14 2.8 0 31  27 2.8 1 22  51 4.0 0  1
7 4.0 0  9  14 4.0 0 19  27 4.0 1 16
;
```

If each test is carried out independently and if for a particular combination of heating and soaking time there is a constant probability that the tested ingot is not ready for rolling, then the random variable r follows a Binomial(n, π) distribution, where the success probability π is a function of heating and soaking time.

The following statements show the use of the events/trials syntax to model the binomial response. The *events* variable in this situation is r (the number of ingots not ready for rolling), and the *trials* variable is n (the number of ingots tested). The dependency of the probability of not being ready for rolling is modeled as a function of heating time, soaking time, and their interaction. The OUTPUT statement stores the linear predictors and the predicted probabilities in the Out data set along with the ID variable.

```
proc hpgenselect data=Ingots;
   model r/n = Heat Soak Heat*Soak / dist=Binomial;
   id Obsnum;
   output out=Out xbeta predicted=Pred;
run;
```

The "Performance Information" table in Output 59.2.1 shows that the procedure executes in single-machine mode.

Output 59.2.1 Performance Information

The HPGENSELECT Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

The "Model Information" table shows that the data are modeled as binomially distributed with a logit link function (Output 59.2.2). This is the default link function in the HPGENSELECT procedure for binary and

binomial data. The procedure uses a ridged Newton-Raphson algorithm to estimate the parameters of the model.

Output 59.2.2 Model Information and Number of Observations

Model Information	
Data Source	WORK.INGOTS
Response Variable (Events)	r
Response Variable (Trials)	n
Distribution	Binomial
Link Function	Logit
Optimization Technique	Newton-Raphson with Ridging

Number of Observations Read	19
Number of Observations Used	19
Number of Events	12
Number of Trials	387

The second table in Output 59.2.2 shows that all 19 observations in the data set were used in the analysis and that the total number of events and trials equal 12 and 387, respectively. These are the sums of the variables r and n across all observations.

Output 59.2.3 displays the "Dimensions" table for the model. There are four columns in the design matrix of the model (the **X** matrix); they correspond to the intercept, the Heat effect, the Soak effect, and the interaction of the Heat and Soak effects. The model is nonsingular, because the rank of the crossproducts matrix equals the number of columns in **X**. All parameters are estimable and participate in the optimization.

Output 59.2.3 Dimensions in Binomial Logistic Regression

Dimensions	
Number of Effects	4
Number of Parameters	4
Columns in X	4

Output 59.2.4 displays the "Fit Statistics" table for this run. Evaluated at the converged estimates, –2 times the value of the log-likelihood function equals 27.9569. Further fit statistics are also given, all of them in "smaller is better" form. The AIC, AICC, and BIC criteria are used to compare non-nested models and to penalize the model fit for the number of observations and parameters. The –2 log-likelihood value can be used to compare nested models by way of a likelihood ratio test.

Output 59.2.4 Fit Statistics

Fit Statistics	
-2 Log Likelihood	27.9569
AIC (smaller is better)	35.9569
AICC (smaller is better)	38.8140
BIC (smaller is better)	39.7346
Pearson Chi-Square	13.4350
Pearson Chi-Square/DF	0.8957

The "Parameter Estimates" table in Output 59.2.5 displays the estimates and standard errors of the model effects.

Output 59.2.5 Parameter Estimates

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	-5.990191	1.666622	12.9183	0.0003
Heat	1	0.096339	0.047067	4.1896	0.0407
Soak	1	0.299574	0.755068	0.1574	0.6916
Heat*Soak	1	-0.008840	0.025319	0.1219	0.7270

You can construct the prediction equation of the model from the "Parameter Estimates" table. For example, an observation with Heat equal to 14 and Soak equal to 1.7 has linear predictor

$$\widehat{\eta} = -5.9902 + 0.09634 \times 14 + 0.2996 \times 1.7 - 0.00884 \times 14 \times 7 = -4.34256$$

The probability that an ingot with these characteristics is not ready for rolling is

$$\widehat{\pi} = \frac{1}{1 + \exp\{-(-4.34256)\}} = 0.01284$$

The OUTPUT statement computes these linear predictors and probabilities and stores them in the Out data set. This data set also contains the ID variable, which is used by the following statements to attach the covariates to these statistics. Output 59.2.6 shows the probability that an ingot with Heat equal to 14 and Soak equal to 1.7 is not ready for rolling.

```
data Out;
   merge Out Ingots;
   by Obsnum;
proc print data=Out;
   where Heat=14 & Soak=1.7;
run;
```

Output 59.2.6 Predicted Probability for Heat=14 and Soak=1.7

Obs	Obsnum	Pred	Xbeta	Heat	Soak	r	n
6	6	0.012836	-4.34256	14	1.7	0	43

Binomial data are a form of grouped binary data where "successes" in the underlying Bernoulli trials are totaled. You can thus expand data for which you use the events/trials syntax and fit them with techniques for binary data.

The following DATA step expands the Ingots data set (which has 12 events in 387 trials) into a binary data set that has 387 observations.

```
data Ingots_binary;
   set Ingots;
   do i=1 to n;
     if i <= r then Y=1; else Y = 0;
     output;
```

```
   end;
run;
```

The following HPGENSELECT statements fit the model by using Heat effect, Soak effect, and their interaction to the binary data set. The **event='1'** response-variable option in the MODEL statement ensures that the HPGENSELECT procedure models the probability that the variable Y takes on the value '1'.

```
proc hpgenselect data=Ingots_binary;
   model Y(event='1') = Heat Soak Heat*Soak / dist=Binary;
run;
```

Output 59.2.7 displays the "Performance Information," "Model Information," "Number of Observations," and the "Response Profile" tables. The data are now modeled as binary (Bernoulli distributed) by using a logit link function. The "Response Profile" table shows that the binary response breaks down into 375 observations where Y equals 0 and 12 observations where Y equals 1.

Output 59.2.7 Model Information in Binary Model

The HPGENSELECT Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.INGOTS_BINARY	V9	Input	On Client

Model Information	
Data Source	WORK.INGOTS_BINARY
Response Variable	Y
Distribution	Binary
Link Function	Logit
Optimization Technique	Newton-Raphson with Ridging

Number of Observations Read	387
Number of Observations Used	387

Response Profile		
Ordered Value	**Y**	**Total Frequency**
1	0	375
2	1	12

You are modeling the probability that Y='1'.

Output 59.2.8 displays the parameter estimates. These results match those in Output 59.2.5.

Output 59.2.8 Parameter Estimates

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	-5.990191	1.666622	12.9183	0.0003
Heat	1	0.096339	0.047067	4.1896	0.0407
Soak	1	0.299574	0.755068	0.1574	0.6916
Heat*Soak	1	-0.008840	0.025319	0.1219	0.7270

Example 59.3: Tweedie Model

(View the complete code for this example (hpgenex3.sas) in the example repository.)

The following HPGENSELECT statements examine the data set getStarted used in the section "Getting Started: HPGENSELECT Procedure" on page 4729, but they request that a Tweedie model be fit by using the continuous variable Total as the response instead of the count variable Y. The following statements fit a log-linked Tweedie model to these data by using classification effects for variables C1–C5. In an insurance underwriting context, Y represents the total number of claims in each category that is defined by C1–C5, and Total represents the total cost of the claims (that is, the sum of costs for individual claims). The CODE statement requests that a text file named "Scoring Parameters.txt" be created. This file contains a SAS program that contains information from the model that allows scoring of a new data set based on the parameter estimates from the current model.

```
proc hpgenselect data=getStarted;
   class C1-C5;
   model Total = C1-C5 / Distribution=Tweedie Link=Log;
   code File='ScoringParameters.txt';
run;
```

The "Optimizations Stage Details" table in Output 59.3.1 shows the stages used in computing the maximum likelihood estimates of the parameters of the Tweedie model. Stage 1 uses quasi-likelihood and all of the data to compute starting values for stage 2, which uses all of the data and the Tweedie log likelihood to compute the final estimates.

Output 59.3.1 Optimization Stage Details

The HPGENSELECT Procedure

Optimization Stage Details			
Optimization Stage	Optimization Type	Sampling Percentage	Observations Used
1	Quasilikelihood	100.00	100
2	Full Likelihood	100.00	100

The "Parameter Estimates" table in Output 59.3.2 shows the resulting regression model parameter estimates, the estimated Tweedie dispersion parameter, and the estimated Tweedie power.

Output 59.3.2 Parameter Estimates

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	3.888904	0.435325	79.8044	<.0001
C1 0	1	-0.072400	0.240613	0.0905	0.7635
C1 1	1	-1.358456	0.324363	17.5400	<.0001
C1 2	1	0.154711	0.237394	0.4247	0.5146
C1 3	0	0	.	.	.
C2 0	1	1.350591	0.289897	21.7050	<.0001
C2 1	1	1.159242	0.275459	17.7106	<.0001
C2 2	1	0.033921	0.303204	0.0125	0.9109
C2 3	0	0	.	.	.
C3 0	1	-0.217763	0.272474	0.6387	0.4242
C3 1	1	-0.289425	0.259751	1.2415	0.2652
C3 2	1	-0.131961	0.276723	0.2274	0.6335
C3 3	0	0	.	.	.
C4 0	1	-0.258069	0.288840	0.7983	0.3716
C4 1	1	-0.057042	0.287566	0.0393	0.8428
C4 2	1	0.219697	0.272064	0.6521	0.4194
C4 3	0	0	.	.	.
C5 0	1	-1.314657	0.257806	26.0038	<.0001
C5 1	1	-0.996980	0.236881	17.7138	<.0001
C5 2	1	-0.481185	0.235614	4.1708	0.0411
C5 3	0	0	.	.	.
Dispersion	1	5.296966	0.773401	.	.
Power	1	1.425625	0.048981	.	.

Now suppose you want to compute predicted values for some different data. If $\mathbf{x}$ is a vector of explanatory variables that might not be in the original data and $\hat{\boldsymbol{\beta}}$ is the vector of estimated regression parameters from the model, then $\mu = g^{-1}(\mathbf{x}'\hat{\boldsymbol{\beta}})$ is the predicted value of the mean, where g is the log link function in this case. The following data contain new values of the regression variables C1–C5, from which you can compute predicted values based on information in the SAS program that is created by the CODE statement. This is called *scoring* the new data set.

```
data ScoringData;
   input C1-C5;
   datalines;
3 3 1 0 2
1 1 2 2 0
3 2 2 2 0
1 1 2 3 2
1 1 2 3 3
3 1 1 0 1
0 2 1 0 0
2 1 3 1 3
3 2 3 2 0
3 0 2 0 1
;
```

The following SAS DATA step creates the new data set Scores, which contains a variable P_Total that represents the predicted values of Total, along with the variables C1–C5. The resulting data are shown in Output 59.3.3.

```
data Scores;
   set ScoringData;
   %inc 'ScoringParameters.txt';
run;
proc print data=Scores;
run;
```

Output 59.3.3 Predicted Values for Scoring Data

Obs	C1	C2	C3	C4	C5	P_Total
1	3	3	1	0	2	17.465
2	1	1	2	2	0	11.737
3	3	2	2	2	0	14.819
4	1	1	2	3	2	21.683
5	1	1	2	3	3	35.083
6	3	1	1	0	1	33.237
7	0	2	1	0	0	7.303
8	2	1	3	1	3	171.711
9	3	2	3	2	0	16.909
10	3	0	2	0	1	47.110

Example 59.4: Model Selection by the LASSO Method

(View the complete code for this example (hpgenex4.sas) in the example repository.)

This example shows how you can use PROC HPGENSELECT to perform model selection among Poisson regression models by using the LASSO method. The following DATA step generates the data for this example. There are 1,000,000 observations in the data set, and the response yPoisson is a Poisson variable with a mean that depends on 20 of the 100 regressors.

```
%let nObs      = 1000000;
%let nContIn   = 20;
%let nContOut  = 80;
%let Seed      = 12345;

data ex4Data;
array xIn{&nContIn};
array xOut{&nContOut};

drop i j sign xBeta expXbeta;

seed = &Seed;
do i=1 to &nObs;
   sign  = -1;
       xBeta = 0;
       do j=1 to dim(xIn);
           call ranuni(seed,xIn[j]);
```

```
            xBeta  = xBeta + j*sign*xIn{j};
            sign   = -sign;
        end;

        do j=1 to dim(xOut);
           call ranuni(seed,xOut[j]);
        end;

        call ranuni(seed,xSubtle);
        call ranuni(seed,xTiny);

        xBeta = xBeta + 0.1*xSubtle + 0.05*xTiny;
        expXbeta = exp(xBeta/20);
        call ranpoi(seed,expXbeta,yPoisson);
        output;
end;
run;
```

The following statements use PROC HPGENSELECT to select a model by using the LASSO method and only the first 10,000 observations:

```
proc hpgenselect data=ex4Data(Obs=10000);
   model yPoisson = x: / dist=Poisson;
   selection method=Lasso(choose=SBC) details=all;
   performance details;
run;
```

Output 59.4.1 shows the "Performance Information" table. This shows that the HPGENSELECT procedure executed in single-machine mode on four threads because the client machine has four CPUs. You can select a certain number of threads on any machine involved in the computations by using the NTHREADS= option in the PERFORMANCE statement.

Output 59.4.1 Performance Information

The HPGENSELECT Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Output 59.4.2 shows the models fit by maximizing the penalized log likelihoods for a sequence of regularization parameters. For each step in the sequence, Output 59.4.2 shows you the effects that are added or removed and the fit statistics AIC, AICC, and BIC (SBC) for each model. The variable Lambda contains the scaling factor that is multiplied by the maximum regularization parameter to obtain the regularization parameter used in that step. Unlike other methods, such as forward selection, LASSO selection includes zero or more effects in each step.

Output 59.4.2 Selection Details

Selection Details						
Step	Description	Effects In Model	Lambda	AIC	AICC	BIC
0	Initial Model	1	1	39592.399	39592.399	39599.609
1	xIn17 entered	5	0.8	38533.060	38533.066	38569.111
	xIn18 entered	5	0.8	38533.060	38533.066	38569.111
	xIn19 entered	5	0.8	38533.060	38533.066	38569.111
	xIn20 entered	5	0.8	38533.060	38533.066	38569.111
2	xIn14 entered	8	0.64	36731.498	36731.513	36789.181
	xIn15 entered	8	0.64	36731.498	36731.513	36789.181
	xIn16 entered	8	0.64	36731.498	36731.513	36789.181
3	xIn11 entered	11	0.512	34887.979	34888.005	34967.292
	xIn12 entered	11	0.512	34887.979	34888.005	34967.292
	xIn13 entered	11	0.512	34887.979	34888.005	34967.292
4	xIn8 entered	12	0.4096	33321.737	33321.769	33408.262
5	xIn7 entered	15	0.3277	32043.428	32043.476	32151.583
	xIn9 entered	15	0.3277	32043.428	32043.476	32151.583
	xIn10 entered	15	0.3277	32043.428	32043.476	32151.583
6	xIn6 entered	16	0.2621	31135.411	31135.465	31250.776
7	xIn5 entered	17	0.2097	30494.512	30494.573	30617.088
8	xIn4 entered	18	0.1678	30062.397	30062.465	30192.183
9	xIn3 entered	19	0.1342	29761.384	29761.460	29898.380
10	xIn2 entered	20	0.1074	29563.603	29563.687	29707.810
11		20	0.0859	29432.716	29432.800	29576.922
12		20	0.0687	29348.856	29348.940	29493.063
13	xOut8 entered	22	0.055	29297.961	29298.062	29456.588
	xOut66 entered	22	0.055	29297.961	29298.062	29456.588
14	xIn1 entered	26	0.044	29265.882	29266.022	29453.350*
	xOut34 entered	26	0.044	29265.882	29266.022	29453.350*
	xOut37 entered	26	0.044	29265.882	29266.022	29453.350*
	xOut65 entered	26	0.044	29265.882	29266.022	29453.350*
15	xOut7 entered	32	0.0352	29247.136	29247.348	29477.867
	xOut22 entered	32	0.0352	29247.136	29247.348	29477.867
	xOut29 entered	32	0.0352	29247.136	29247.348	29477.867
	xOut51 entered	32	0.0352	29247.136	29247.348	29477.867
	xOut57 entered	32	0.0352	29247.136	29247.348	29477.867
	xOut60 entered	32	0.0352	29247.136	29247.348	29477.867
16	xOut1 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut6 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut14 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut18 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut27 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut30 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut33 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut53 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut56 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut59 entered	44	0.0281	29244.461	29244.859	29561.716
	xOut62 entered	44	0.0281	29244.461	29244.859	29561.716
	xTiny entered	44	0.0281	29244.461	29244.859	29561.716
17	xOut11 entered	56	0.0225	29245.665	29246.307	29649.444

Output 59.4.2 *continued*

Selection Details						
Step	Description	Effects In Model	Lambda	AIC	AICC	BIC
	xOut24 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut35 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut36 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut43 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut45 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut49 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut54 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut69 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut75 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut76 entered	56	0.0225	29245.665	29246.307	29649.444
	xOut79 entered	56	0.0225	29245.665	29246.307	29649.444
18	xOut12 entered	63	0.018	29241.974	29242.786	29696.225
	xOut19 entered	63	0.018	29241.974	29242.786	29696.225
	xOut28 entered	63	0.018	29241.974	29242.786	29696.225
	xOut48 entered	63	0.018	29241.974	29242.786	29696.225
	xOut61 entered	63	0.018	29241.974	29242.786	29696.225
	xOut78 entered	63	0.018	29241.974	29242.786	29696.225
	xSubtle entered	63	0.018	29241.974	29242.786	29696.225
19	xOut4 entered	69	0.0144	29241.231	29242.203	29738.744
	xOut39 entered	69	0.0144	29241.231	29242.203	29738.744
	xOut40 entered	69	0.0144	29241.231	29242.203	29738.744
	xOut44 entered	69	0.0144	29241.231	29242.203	29738.744
	xOut67 entered	69	0.0144	29241.231	29242.203	29738.744
	xOut71 entered	69	0.0144	29241.231	29242.203	29738.744
20	xOut2 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut3 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut16 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut21 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut31 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut41 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut46 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut50 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut72 entered	79	0.0115	29251.884	29253.158	29821.500
	xOut77 entered	79	0.0115	29251.884	29253.158	29821.500

*** Optimal Value of Criterion**

The model in step 14 had the smallest Schwarz Bayesian criterion (BIC in Output 59.4.2), and it was chosen as the final model because the CHOOSE=SBC option was specified in the SELECTION statement. Output 59.4.3 shows the parameter estimates for the selected model. You can see that the LASSO selection in which the final model was chosen based on the SBC criterion retains all 20 of the true effects but also keeps several extraneous effects.

Output 59.4.3 Parameter Estimates for the Selected Model

Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	-0.008605
xIn1	1	-0.004242
xIn2	1	0.076690
xIn3	1	-0.141401
xIn4	1	0.160843
xIn5	1	-0.219377
xIn6	1	0.268402
xIn7	1	-0.336859
xIn8	1	0.384260
xIn9	1	-0.407838
xIn10	1	0.397287
xIn11	1	-0.516073
xIn12	1	0.559396
xIn13	1	-0.547200
xIn14	1	0.631820
xIn15	1	-0.692216
xIn16	1	0.783101
xIn17	1	-0.790773
xIn18	1	0.875372
xIn19	1	-0.837447
xIn20	1	0.954752
xOut8	1	0.013643
xOut34	1	0.002167
xOut37	1	0.001344
xOut65	1	-0.007790
xOut66	1	0.015540

Output 59.4.4 shows timing information for the PROC HPGENSELECT run. This table is produced when you specify the DETAILS option in the PERFORMANCE statement. You can see that, in this case, the majority of time is spent in performing model selection.

Output 59.4.4 Timing

Procedure Task Timing		
Task	**Seconds**	**Percent**
Reading and Levelizing Data	0.08	0.07%
Candidate model fit	0.06	0.05%
Performing Model Selection	112.89	99.88%

References

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Cox, D. R., and Snell, E. J. (1989). *The Analysis of Binary Data*. 2nd ed. London: Chapman & Hall.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981). "An Adaptive Nonlinear Least-Squares Algorithm." *ACM Transactions on Mathematical Software* 7:348–368.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Dunn, P. K., and Smyth, G. K. (2005). "Series Evaluation of Tweedie Exponential Dispersion Model Densities." *Statistics and Computing* 15:267–280.

Dunn, P. K., and Smyth, G. K. (2008). "Series Evaluation of Tweedie Exponential Dispersion Model Densities by Fourier Inversion." *Statistics and Computing* 18:73–86.

Efron, B., Hastie, T. J., Johnstone, I. M., and Tibshirani, R. (2004). "Least Angle Regression." *Annals of Statistics* 32:407–499. With discussion.

Eskow, E., and Schnabel, R. B. (1991). "Algorithm 695: Software for a New Modified Cholesky Factorization." *ACM Transactions on Mathematical Software* 17:306–312.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Frees, E. W. (2010). *Regression Modeling with Actuarial and Financial Applications*. Cambridge: Cambridge University Press.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer-Verlag.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Nesterov, Y. (2013). "Gradient Methods for Minimizing Composite Objective Function." *Mathematical Programming* 140:125–161.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Yuan, M., and Lin, L. (2006). "Model Selection and Estimation in Regression with Grouped Variables." *Journal of the Royal Statistical Society, Series B* 68:49–67.

Chapter 60
The HPLMIXED Procedure

Contents

Overview: HPLMIXED Procedure **4800**
PROC HPLMIXED Features 4801
Notation for the Mixed Model 4802
PROC HPLMIXED Contrasted with Other SAS Procedures 4803
Getting Started: HPLMIXED Procedure **4803**
Mixed Model Analysis of Covariance with Many Groups 4803
Syntax: HPLMIXED Procedure **4806**
PROC HPLMIXED Statement 4807
CLASS Statement 4812
ID Statement 4812
MODEL Statement 4813
OUTPUT Statement 4814
PARMS Statement 4815
PERFORMANCE Statement 4817
RANDOM Statement 4817
REPEATED Statement 4823
Details: HPLMIXED Procedure **4824**
Linear Mixed Models Theory 4824
Matrix Notation 4824
Formulation of the Mixed Model 4825
Estimating Covariance Parameters in the Mixed Model 4830
Estimating Fixed and Random Effects in the Mixed Model 4831
Statistical Properties 4832
Computational Method 4833
Multithreading 4833
Displayed Output 4834
Performance Information 4834
Model Information 4834
Class Level Information 4834
Dimensions 4834
Number of Observations 4834
Optimization Information 4834
Iteration History 4835
Convergence Status 4835
Covariance Parameter Estimates 4836
Fit Statistics 4836

Timing Information 4836
ODS Table Names 4837
Examples: HPLMIXED Procedure **4837**
Example 60.1: Computing BLUPs for a Large Number of Subjects 4838
References **4840**

Overview: HPLMIXED Procedure

The HPLMIXED procedure fits a variety of mixed linear models to data and enables you to use these fitted models to make statistical inferences about the data. A *mixed linear model* is a generalization of the standard linear model used in the GLM procedure in SAS/STAT software; the generalization is that the data are permitted to exhibit correlation and nonconstant variability. Therefore, the mixed linear model provides you with the flexibility of modeling not only the means of your data (as in the standard linear model) but also their variances and covariances.

The primary assumptions underlying the analyses performed by PROC HPLMIXED are as follows:

- The data are normally distributed (Gaussian).
- The means (expected values) of the data are linear in terms of a certain set of parameters.
- The variances and covariances of the data are in terms of a different set of parameters, and they exhibit a structure that matches one of those available in PROC HPLMIXED.

Because Gaussian data can be modeled entirely in terms of their means and variances/covariances, the two sets of parameters in a mixed linear model actually specify the complete probability distribution of the data. The parameters of the mean model are referred to as *fixed-effects parameters*, and the parameters of the variance-covariance model are referred to as *covariance parameters*.

The fixed-effects parameters are associated with known explanatory variables, as in the standard linear model. These variables can be either qualitative (as in the traditional analysis of variance) or quantitative (as in standard linear regression). However, the covariance parameters are what distinguishes the mixed linear model from the standard linear model.

The need for covariance parameters arises quite frequently in applications; the following scenarios are the most typical:

- The experimental units on which the data are measured can be grouped into clusters, and the data from a common cluster are correlated. This scenario can be generalized to include one set of clusters nested within another. For example, if students are the experimental unit, they can be clustered into classes, which in turn can be clustered into schools. Each level of this hierarchy can introduce an additional source of variability and correlation.
- Repeated measurements are taken on the same experimental unit, and these repeated measurements are correlated or exhibit variability that changes. This scenario occurs in longitudinal studies, where repeated measurements are taken over time. Alternatively, the repeated measures could be spatial or multivariate in nature.

PROC HPLMIXED provides a variety of covariance structures to handle these two scenarios. The most common covariance structures arise from the use of *random-effects parameters*, which are additional unknown random variables that are assumed to affect the variability of the data. The variances of the random-effects parameters, commonly known as *variance components*, become the covariance parameters for this particular structure. Traditional mixed linear models contain both fixed- and random-effects parameters; in fact, it is the combination of these two types of effects that led to the name *mixed model*. PROC HPLMIXED fits not only these traditional variance component models but also numerous other covariance structures.

PROC HPLMIXED fits the structure you select to the data by using the method of *restricted maximum likelihood (REML)*, also known as *residual maximum likelihood*. It is here that the Gaussian assumption for the data is exploited.

PROC HPLMIXED Features

PROC HPLMIXED provides easy accessibility to numerous mixed linear models that are useful in many common statistical analyses.

Here are some basic features of PROC HPLMIXED:

- covariance structures, including variance components, compound symmetry, unstructured, AR(1), Toeplitz, and factor analytic
- MODEL, RANDOM, and REPEATED statements for model specification as in the HPLMIXED procedure
- appropriate standard errors, t tests, and F tests for all specified estimable linear combinations of fixed and random effects
- a subject effect that enables blocking
- REML and ML (maximum likelihood) estimation methods implemented with a variety of optimization algorithms
- capacity to handle unbalanced data
- special dense and sparse algorithms that take advantage of multicore computing environment

PROC HPLMIXED uses the Output Delivery System (ODS), a SAS subsystem that provides capabilities for displaying and controlling the output from SAS procedures. ODS enables you to convert any output from PROC HPLMIXED into a SAS data set. See the section "ODS Table Names" on page 4837.

Notation for the Mixed Model

This section introduces the mathematical notation used throughout this chapter to describe the mixed linear model and assumes familiarity with basic matrix algebra (for an overview, see Searle 1982). A more detailed description of the mixed model is contained in the section "Linear Mixed Models Theory" on page 4824.

A statistical model is a mathematical description of how data are generated. The standard linear model, as used by the GLM procedure, is one of the most common statistical models:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

In this expression, $\mathbf{y}$ represents a vector of observed data, $\boldsymbol{\beta}$ is an unknown vector of fixed-effects parameters with a known design matrix $\mathbf{X}$, and $\boldsymbol{\epsilon}$ is an unknown random error vector that models the statistical noise around $\mathbf{X}\boldsymbol{\beta}$. The focus of the standard linear model is to model the mean of $\mathbf{y}$ by using the fixed-effects parameters $\boldsymbol{\beta}$. The residual errors $\boldsymbol{\epsilon}$ are assumed to be independent and identically distributed Gaussian random variables with mean 0 and variance σ^2.

The mixed model generalizes the standard linear model as follows:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

Here, $\boldsymbol{\gamma}$ is an unknown vector of random-effects parameters with a known design matrix $\mathbf{Z}$, and $\boldsymbol{\epsilon}$ is an unknown random error vector whose elements are no longer required to be independent and homogeneous.

To further develop this notion of variance modeling, assume that $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$ are Gaussian random variables that are uncorrelated, have expectations $\mathbf{0}$, and have variances $\mathbf{G}$ and $\mathbf{R}$, respectively. The variance of $\mathbf{y}$ is thus

$$\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$$

Note that when $\mathbf{R} = \sigma^2\mathbf{I}$ and $\mathbf{Z} = \mathbf{0}$, the mixed model reduces to the standard linear model.

You can model the variance of the data $\mathbf{y}$ by specifying the structure of $\mathbf{Z}$, $\mathbf{G}$, and $\mathbf{R}$. The model matrix $\mathbf{Z}$ is set up in the same fashion as $\mathbf{X}$, the model matrix for the fixed-effects parameters. For $\mathbf{G}$ and $\mathbf{R}$, you must select some covariance structure. Possible covariance structures include the following:

- variance components
- compound symmetry (common covariance plus diagonal)
- unstructured (general covariance)
- autoregressive
- spatial
- general linear
- factor analytic

By appropriately defining the model matrices $\mathbf{X}$ and $\mathbf{Z}$ in addition to the covariance structure matrices $\mathbf{G}$ and $\mathbf{R}$, you can perform numerous mixed model analyses.

PROC HPLMIXED Contrasted with Other SAS Procedures

The RANDOM and REPEATED statements of the HPLMIXED procedure follow the convention of the same statements in the MIXED procedure in SAS/STAT software. For information about how these statements differ from RANDOM and REPEATED statements in the MIXED procedure, see the documentation for the MIXED procedure in the *SAS/STAT User's Guide*.

The GLIMMIX procedure in SAS/STAT software fits generalized linear mixed models. Linear mixed models—where the data are normally distributed, given the random effects—are in the class of generalized linear mixed models. Therefore, PROC GLIMMIX accommodates nonnormal data with random effects.

Generalized linear mixed models have intrinsically nonlinear features because a nonlinear mapping (the link function) connects the conditional mean of the data (given the random effects) to the explanatory variables. The NLMIXED procedure also accommodates nonlinear structures in the conditional mean, but places no restrictions on the nature of the nonlinearity.

The HPMIXED procedure in SAS/STAT software is also termed a "high-performance" procedure, but it does not follow the general pattern of high-performance analytical procedures. The HPMIXED procedure does not take advantage of multicore computing environment; it derives high performance from applying sparse techniques to solving the mixed model equations. The HPMIXED procedure fits a small subset of the statistical models you can fit with the MIXED or HPLMIXED procedures and is particularly suited for problems in which the $[\mathbf{XZ}]'[\mathbf{XZ}]$ crossproducts matrix is sparse.

The HPLMIXED procedure employs algorithms that are specialized for multicore computing environment. The HPLMIXED procedure does not support BY processing.

Getting Started: HPLMIXED Procedure

Mixed Model Analysis of Covariance with Many Groups

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hplmxgs.sas.)

Suppose you are an educational researcher who studies how student scores on math tests change over time. Students are tested four times, and you want to estimate the overall rise or fall, accounting for correlation between test response behaviors of students in the same neighborhood and school. One way to model this correlation is by using a random-effects analysis of covariance, where the scores for students from the same neighborhood and school are all assumed to share the same quadratic mean test response function, the parameters of this response function being random. The following statements simulate a data set with this structure:

```
data SchoolSample;
   do SchoolID = 1 to 300;
      do nID = 1 to 25;
         Neighborhood = (SchoolID-1)*5 + nId;
         bInt   = 5*ranuni(1);
         bTime  = 5*ranuni(1);
```

```
            bTime2 =   ranuni(1);
            do sID = 1 to 2;
               do Time = 1 to 4;
                  Math = bInt + bTime*Time + bTime2*Time*Time + rannor(2);
                  output;
                  end;
               end;
            end;
         end;
run;
```

In these data, there are 300 schools and about 1,500 neighborhoods; neighborhoods are associated with more than one school and vice versa. The following statements use PROC HPLMIXED to fit a mixed analysis of covariance model to these data.

```
proc hplmixed data=SchoolSample;
   class Neighborhood SchoolID;
   model Math = Time Time*Time / solution;
   random   int Time Time*Time / sub=Neighborhood(SchoolID) type=un;
run;
```

This model fits a quadratic mean response model with an unstructured covariance matrix to model the covariance between the random parameters of the response model. With 7,500 neighborhood/school combinations, this model can be computationally daunting to fit, but PROC HPLMIXED finishes quickly and displays the results shown in Figure 60.1.

Figure 60.1 Mixed Model Analysis of Covariance

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.SCHOOLSAMPLE	V9	Input	On Client

Model Information	
Data Set	WORK.SCHOOLSAMPLE
Dependent Variable	Math
Covariance Structure	Unstructured
Subject Effect	Neighborho(SchoolID)
Estimation Method	Restricted Maximum Likelihood
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

Class Level Information		
Class	**Levels**	**Values**
Neighborhood	1520	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...
SchoolID	300	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...

Dimensions	
Covariance Parameters	7
Columns in X	3
Columns in Z per Subject	3
Subjects	7500
Max Obs per Subject	8

Number of Observations Read	60000
Number of Observations Used	60000
Number of Observations Not Used	0

Optimization Information	
Optimization Technique	Newton-Raphson with Ridging
Parameters in Optimization	6
Lower Boundaries	3
Upper Boundaries	0
Starting Values From	Data

Convergence criterion (ABSGCONV=0.00001) satisfied.

Covariance Parameter Estimates		
Cov Parm	**Subject**	**Estimate**
UN(1,1)	Neighborho(SchoolID)	2.0902
UN(2,1)	Neighborho(SchoolID)	0.000349
UN(2,2)	Neighborho(SchoolID)	2.0517
UN(3,1)	Neighborho(SchoolID)	0.01448
UN(3,2)	Neighborho(SchoolID)	0.01599
UN(3,3)	Neighborho(SchoolID)	0.08047
Residual		1.0083

Fit Statistics	
-2 Res Log Likelihood	225642
AIC (Smaller is Better)	225656
AICC (Smaller is Better)	225656
BIC (Smaller is Better)	225704

Solution for Fixed Effects

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	2.5070	0.02828	6E4	88.66	<.0001
Time	2.5124	0.02659	6E4	94.48	<.0001
Time*Time	0.5010	0.005247	6E4	95.48	<.0001

Syntax: HPLMIXED Procedure

The following statements are available in PROC HPLMIXED.

```
PROC HPLMIXED <options> ;
    CLASS variables ;
    ID variables ;
    MODEL dependent = <fixed-effects> </ options> ;
    OUTPUT <OUT=SAS-data-set> <keyword <=name>>...<keyword <=name>> </ options> ;
    RANDOM random-effects </ options> ;
    REPEATED repeated-effect </ options> ;
    PARMS <(value-list) ... > </ options> ;
    PERFORMANCE <options> ;
```

Items within angle brackets (< >) are optional. The RANDOM statement can appear multiple times. Other statements can appear only once.

The PROC HPLMIXED and MODEL statements are required, and the MODEL statement must appear after the CLASS statement if a CLASS statement is included. The RANDOM statement must follow the MODEL statement.

Table 60.1 summarizes the basic functions and important options of the PROC HPLMIXED statements. The syntax of each statement in Table 60.1 is described in the following sections in alphabetical order after the description of the PROC HPLMIXED statement.

Table 60.1 Summary of PROC HPLMIXED Statements

Statement	Description	Important Options
PROC HPLMIXED	Invokes the procedure	DATA= specifies the input data set; METHOD= specifies the estimation method.
CLASS	Declares qualitative variables that create indicator variables in **X** and **Z** matrices.	None
ID	Lists additional variables to be included in predicted values tables	None
MODEL	Specifies dependent variable and fixed effects, setting up **X**	S requests a solution for fixed-effects parameters.
RANDOM	Specifies random effects, setting up **Z** and **G**	SUBJECT= creates block-diagonality; TYPE= specifies the covariance structure; S requests a solution for the random effects.
REPEATED	Sets up **R**	SUBJECT= creates block-diagonality; TYPE= specifies the covariance structure.

Table 60.1 *continued*

Statement	Description	Important Options
OUTPUT	Creates a data set that contains observationwise statistics	ALLSTATS requests that all statistics be computed.
PARMS	Specifies a grid of initial values for the covariance parameters	HOLD= and NOITER hold the covariance parameters or their ratios constant; PARMSDATA= reads the initial values from a SAS data set.

PROC HPLMIXED Statement

PROC HPLMIXED < *options* > **;**

The PROC HPLMIXED statement invokes the procedure. Table 60.2 summarizes important options in the PROC HPLMIXED statement by function. These and other options in the PROC HPLMIXED statement are then described fully in alphabetical order.

Table 60.2 PROC HPLMIXED Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
METHOD=	Specifies the estimation method
NAMELEN=	Limits the length of effect names
BLUP	Computes the best linear unbiased prediction
Options Related to Output	
NOCLPRINT	Suppresses the "Class Level Information" table completely or in parts
MAXCLPRINT=	Specifies the maximum levels of CLASS variables to print
RANKS	Displays the rank of design matrix **X**
Optimization Options	
ABSCONV=	Tunes an absolute function convergence criterion
ABSFCONV=	Tunes an absolute function difference convergence criterion
ABSGCONV=	Tunes the absolute gradient convergence criterion
FCONV=	Tunes the relative function convergence criterion
GCONV=	Tunes the relative gradient convergence criterion
MAXITER=	Chooses the maximum number of iterations in any optimization
MAXFUNC=	Specifies the maximum number of function evaluations in any optimization
MAXTIME=	Specifies the upper limit on seconds of CPU time for any optimization

Table 60.2 *continued*

Option	Description
MINITER=	Specifies the minimum number of iterations in any optimization
TECHNIQUE=	Selects the optimization technique
XCONV=	Tunes the relative parameter convergence criterion

You can specify the following *options* in the PROC HPLMIXED statement.

ABSCONV=*r*

specifies an absolute function convergence criterion. For minimization, termination requires $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of r is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

ABSFCONV=*r*

specifies an absolute function difference convergence criterion. For all techniques except Nelder–Mead simplex (NMSIMP), termination requires a small change of the function value in successive iterations:

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex with the lowest function value and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default value is $r = 0$.

ABSGCONV=*r*

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small:

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the j parameter. This criterion is not used by the NMSIMP technique. The default value is r=1E–5.

BLUP< *(suboptions)* >

requests that best linear unbiased predictions (BLUPs) for the random effects be displayed. To use this option, you must also use the PARMS statement to specify fixed values for the covariance parameters, which means that the NOITER option in the PARMS statement will be implied by the BLUP option. Also, the iterations in the ODS Table IterHistory will refer to iterations used to compute the BLUP rather than optimization iterations.

You can specify the following *suboptions*:

ITPRINT=*number* specifies that the iteration history be displayed after every *number* of iterations. The default value is 10, which means the procedure displays the iteration history for every 10 iterations.

MAXITER=*number* specifies the maximum number of iterations allowed. The default value is the number of parameters in the BLUP option plus 2.

TOL=*number* specifies the tolerance value. The default value is the square root of machine precision.

DATA=*SAS-data-set*

names the SAS data set to be used as the input data set. The default is the most recently created data set.

FCONV=*r*

specifies a relative function convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex with the lowest function value and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default is $r = 10^{-\mathrm{FDIGITS}}$, where FDIGITS is $-\log_{10}(\epsilon)$ and ϵ is the machine precision.

GCONV=*r*

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, termination requires that the normalized predicted function reduction be small,

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the CONGRA technique (where a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \, |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The default value is *r*=1E–8.

MAXCLPRINT=*number*

specifies the maximum levels of CLASS variables to print in the ODS table "ClassLevels." The default value is 20. MAXCLPRINT=0 enables you to print all levels of each CLASS variable. However, the option NOCLPRINT takes precedence over MAXCLPRINT.

MAXFUNC=*n*

specifies the maximum number *n* of function calls in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, NEWRAP: 125
- QUANEW, DBLDOG: 500
- CONGRA: 1,000
- NMSIMP: 3,000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed *n*. You can choose the optimization technique with the TECHNIQUE= option.

MAXITER=*n*

specifies the maximum number *n* of iterations in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, NEWRAP: 50
- QUANEW, DBLDOG: 200
- CONGRA: 400
- NMSIMP: 1,000

These default values also apply when *n* is specified as a missing value. You can choose the optimization technique with the TECHNIQUE= option.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The default value is the largest floating-point double representation of your computer. The time specified by the MAXTIME= option is checked only once at the end of each iteration. Therefore, the actual running time can be longer than *r*.

METHOD=REML

METHOD=ML

specifies the estimation method for the covariance parameters. METHOD=REML performs residual (restricted) maximum likelihood; it is the default method. METHOD=ML performs maximum likelihood.

MINITER=*n*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

NAMELEN=*number*
specifies the length to which long effect names are shortened. The minimum value is 20, which is also the default.

NOCLPRINT<=*number***>**
suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying a *number* helps to reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOPRINT
suppresses the generation of ODS output.

RANKS
displays the rank of design matrix **X**.

SINGCHOL=*number*
tunes the singularity criterion in Cholesky decompositions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGSWEEP=*number*
tunes the singularity criterion for sweep operations. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGULAR=*number*
tunes the general singularity criterion applied by the HPLMIXED procedure in sweeps and inversions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

TECHNIQUE=*keyword*
specifies the optimization technique for obtaining maximum likelihood estimates. You can specify any of the following *keywords*:

CONGRA	performs a conjugate-gradient optimization.
DBLDOG	performs a version of double-dogleg optimization.
NEWRAP	performs a Newton-Raphson optimization combining a line-search algorithm with ridging.
NMSIMP	performs a Nelder-Mead simplex optimization.
NONE	performs no optimization.
NRRIDG	performs a Newton-Raphson optimization with ridging.
QUANEW	performs a dual quasi-Newton optimization.
TRUREG	performs a trust-region optimization.

The default value is TECHNIQUE=NRRIDG.

XCONV=*r*

specifies the relative parameter convergence criterion:

- For all techniques except NMSIMP, termination requires a small relative parameter change in subsequent iterations:

$$\frac{\max_j |\psi_j^{(k)} - \psi_j^{(k-1)}|}{\max(|\psi_j^{(k)}|, |\psi_j^{(k-1)}|)} \leq r$$

- For the NMSIMP technique, the same formula is used, but $\psi_j^{(k)}$ is defined as the vertex with the lowest function value and $\psi_j^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default value is r = 1E–8 for the NMSIMP technique and r = 0 otherwise.

CLASS Statement

CLASS *variables* **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. These variables enter the analysis not through their values, but through levels to which the unique values are mapped. For more information about these mappings, see the section "Levelization of Classification Variables" on page 409.

If a CLASS statement is specified, it must precede the MODEL statement in high-performance analytical procedures that support a MODEL statement.

Levels of classification variables are ordered by their external formatted values, except for numeric variables with no explicit format, which are ordered by their unformatted (internal) values.

ID Statement

ID *variables* **;**

The ID statement specifies which variables from the input data set are to be included in the OUT= data set that is created by the OUTPUT statement. If you do not specify an ID statement, then no variables from the input data set are included in the OUT= data set.

MODEL Statement

MODEL *dependent* **=** < *fixed-effects* >< / *options* > **;**

The MODEL statement names a single dependent variable and the fixed effects, which determine the **X** matrix of the mixed model. (For more information, see the section "Parameterization of Model Effects" on page 412. The MODEL statement is required.

An intercept is included in the fixed-effects model by default. If no fixed effects are specified, only this intercept term is fit. The intercept can be removed by using the NOINT option.

Table 60.3 summarizes options in the MODEL statement. These are subsequently discussed in detail in alphabetical order.

Table 60.3 Summary of Important MODEL Statement Options

Option	Description
Model Building	
NOINT	Excludes the fixed-effect intercept from model
Statistical Computations	
ALPHA=α	Determines the confidence level $(1-\alpha)$ for fixed effects
DDFM=	Specifies the method for computing denominator degrees of freedom
Statistical Output	
CL	Displays confidence limits for fixed-effects parameter estimates
SOLUTION	Displays fixed-effects parameter estimates

You can specify the following *options* in the MODEL statement after a slash (/).

ALPHA=*number*

sets the confidence level to be $1-$*number* for each confidence interval of the fixed-effects parameters. The value of *number* must be between 0 and 1; the default is 0.05.

CL

requests that t-type confidence limits be constructed for each of the fixed-effects parameter estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

DDFM=NONE | RESIDUAL

specifies the method for computing the denominator degrees of freedom for the tests of fixed effects.

The DDFM=RESIDUAL option performs all tests by using the residual degrees of freedom, $n - \mathrm{rank}(\mathbf{X})$, where n is the number of observations used. It is the default degrees-of-freedom method.

DDFM=NONE specifies that no denominator degrees of freedom be applied. PROC HPLMIXED then essentially assumes that infinite degrees of freedom are available in the calculation of p-values. The p-values for t tests are then identical to p-values that are derived from the standard normal distribution. In the case of F tests, the p-values equal those of chi-square tests determined as follows: if F_{obs} is the

observed value of the F test with l numerator degrees of freedom, then

$$p = \Pr\{F_{l,\infty} > F_{\text{obs}}\} = \Pr\{\chi^2_l > lF_{\text{obs}}\}$$

NOINT

requests that no intercept be included in the model. (An intercept is included by default.)

SOLUTION

S

requests that a solution for the fixed-effects parameters be produced. Using notation from the section "Linear Mixed Models Theory" on page 4824, the fixed-effects parameter estimates are $\widehat{\boldsymbol{\beta}}$ and their approximate standard errors are the square roots of the diagonal elements of $(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}$.

Along with the estimates and their approximate standard errors, a t statistic is computed as the estimate divided by its standard error. The Pr > |t| column contains the two-tailed p-value that corresponds to the t statistic and associated degrees of freedom. You can use the CL option to request confidence intervals for all of the parameters; they are constructed around the estimate by using a radius that is the product of the standard error times a percentage point from the t distribution.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*> < *keyword* < **=***name*> >. . . < *keyword* < **=***name*> > < / *options*> **;**

The OUTPUT statement creates a data set that contains predicted values and residual diagnostics, which are computed after the model is fit. The variables in the input data set are *not* included in the output data set, in order to avoid data duplication for large data sets; however, variables that are specified in the ID statement are included. By default, only predicted values are included in the output data set.

For example, suppose that the data set Scores contains the variables Score, Machine, and Person. The following statements fit a model that has fixed machine and random person effects and save the predicted and residual values to the data set IgausOut:

```
proc hplmixed data = Scores;
   class machine person score;
   model score = machine;
   random person;
   output out=igausout pred=p resid=r;
run;
```

You can specify the following syntax element in the OUTPUT statement:

OUT=*SAS-data-set*

specifies the name of the output data set. If the OUT= option is omitted, PROC HPLMIXED uses the DATA*n* convention to name the output data set.

A *keyword* can appear multiple times in the OUTPUT statement. Table 60.4 lists the keywords and the default names that PROC HPLMIXED assigns if you do not specify a *name*. In this table, y denotes the response variable.

Table 60.4 Keywords for Output Statistics

Keyword	Description	Expression	Name
PRED	Linear predictor	$\widehat{\eta} = \mathbf{x}'\widehat{\boldsymbol{\beta}} + \mathbf{z}'\widehat{\boldsymbol{\gamma}}$	Pred
PREDPA	Marginal linear predictor	$\widehat{\eta}_m = \mathbf{x}'\widehat{\boldsymbol{\beta}}$	PredPA
RESIDUAL	Residual	$r = y - \widehat{\eta}$	Resid
RESIDUALPA	Marginal residual	$r_m = y - \widehat{\eta}_m$	ResidPA

The marginal linear predictor and marginal residual are also referred to as the predicted population average (PREDPA) and residual population average (RESIDUALPA), respectively. You can use the following shortcuts to request statistics: PRED for PREDICTED and RESID for RESIDUAL.

You can specify the following *option* in the OUTPUT statement after a slash (/):

ALLSTATS
requests that all statistics be computed. If you do not use a *keyword* to assign a name, PROC HPLMIXED uses the default name.

PARMS Statement

PARMS < *(value-list)*. . . > < / *options* > **;**

The PARMS statement specifies initial values for the covariance parameters, or it requests a grid search over several values of these parameters. You must specify the values in the order in which they appear in the "Covariance Parameter Estimates" table.

The *value-list* specification can take any of several forms:

m	a single value
$m_1, m_2, \ldots, m_n$	several values
m to n	a sequence in which m equals the starting value, n equals the ending value, and the increment equals 1
m to n by i	a sequence in which m equals the starting value, n equals the ending value, and the increment equals i
m_1, m_2 to m_3	mixed values and sequences

You can use the PARMS statement to input known parameters.

If you specify more than one set of initial values, PROC HPLMIXED performs a grid search of the likelihood surface and uses the best point on the grid for subsequent analysis. Specifying a large number of grid points can result in long computing times.

The results from the PARMS statement are the values of the parameters on the specified grid (denoted by CovP1 through CovP*n*), the residual variance (possibly estimated) for models with a residual variance parameter, and various functions of the likelihood.

If there are multiple PARMS statements, the first one is used and the rest are ignored.

You can specify the following *options* in the PARMS statement after a slash (/).

HOLD=*all*

EQCONS=*all*

specifies that all parameter values be held to equal the specified values.

For example, the following statement constrains all covariance parameters to equal 5, 3, 2, and 3:

```
parms (5) (3) (2) (3) / hold=all;
```

LOWERB=*value-list*

enables you to specify lower boundary constraints on the covariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that PROC HPLMIXED uses for the covariance parameters, and each number corresponds to the lower boundary constraint. A missing value instructs PROC HPLMIXED to use its default constraint. If you do not specify numbers for all of the covariance parameters, PROC HPLMIXED assumes the remaining ones are missing.

This option is useful when you want to constrain the **G** matrix to be positive definite in order to avoid the more computationally intensive algorithms that would be required when **G** becomes singular. The corresponding statements for a random coefficients model are as follows:

```
proc hplmixed;
   class person;
   model y = time;
   random int time / type=fa0(2) sub=person;
   parms / lowerb=1e-4,.,1e-4;
run;
```

The TYPE=FA0(2) structure specifies a Cholesky root parameterization for the 2×2 unstructured blocks in **G**. This parameterization ensures that the **G** matrix is nonnegative definite, and the PARMS statement then ensures that it is positive definite by constraining the two diagonal terms to be greater than or equal to 1E–4.

NOITER

requests that no optimization iterations be performed and that PROC HPLMIXED use the best value from the grid search to perform inferences. By default, iterations begin at the best value from the PARMS grid search. The NOITER option will be implied by the specification of the BLUP option in the HPLMIXED statement.

PARMSDATA=*SAS-data-set*

PDATA=*SAS-data-set*

reads in covariance parameter values from a SAS data set. The data set should contain the Est or Covp1 through Covp*n* variables.

UPPERB=*value-list*

enables you to specify upper boundary constraints on the covariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that PROC HPLMIXED uses for the covariance parameters, and each number corresponds to the upper boundary constraint. A missing value instructs PROC HPLMIXED to use its default constraint. If you do not specify numbers for all of the covariance parameters, PROC HPLMIXED assumes that the remaining ones are missing.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of HPLMIXED procedure.

The PERFORMANCE statement for SAS high-performance analytical procedures is documented in the section "PERFORMANCE Statement" on page 569.

RANDOM Statement

RANDOM *random-effects* < / *options* > **;**

The RANDOM statement defines the random effects that constitute the $\boldsymbol{\gamma}$ vector in the mixed model. You can use this statement to specify traditional variance component models and to specify random coefficients. The random effects can be classification or continuous, and multiple RANDOM statements are possible.

Using notation from the section "Linear Mixed Models Theory" on page 4824, the purpose of the RANDOM statement is to define the $\mathbf{Z}$ matrix of the mixed model, the random effects in the $\boldsymbol{\gamma}$ vector, and the structure of $\mathbf{G}$. The $\mathbf{Z}$ matrix is constructed exactly as the $\mathbf{X}$ matrix for the fixed effects is constructed, and the $\mathbf{G}$ matrix is constructed to correspond with the effects that constitute $\mathbf{Z}$. The structure of $\mathbf{G}$ is defined by using the TYPE= option.

You can specify INTERCEPT (or INT) as a random effect to indicate the intercept. PROC HPLMIXED does not include the intercept in the RANDOM statement by default as it does in the MODEL statement.

Table 60.5 summarizes important options in the RANDOM statement. All options are subsequently discussed in alphabetical order.

Table 60.5 Summary of Important RANDOM Statement Options

Option	Description
Construction of Covariance Structure	
SUBJECT=	Identifies the subjects in the model
TYPE=	Specifies the covariance structure
Statistical Output	
ALPHA=α	Determines the confidence level $(1-\alpha)$
CL	Requests confidence limits for predictors of random effects
SOLUTION	Displays solutions $\widehat{\boldsymbol{\gamma}}$ of the random effects

You can specify the following *options* in the RANDOM statement after a slash (/).

ALPHA=*number*

sets the confidence level to be 1−*number* for each confidence interval of the random-effects estimates. The value of *number* must be between 0 and 1; the default is 0.05.

CL

requests that *t*-type confidence limits be constructed for each of the random-effect estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

SOLUTION

S

requests that the solution for the random-effects parameters be produced. Using notation from the section "Linear Mixed Models Theory" on page 4824, these estimates are the empirical best linear unbiased predictors (EBLUPs), $\widehat{\boldsymbol{\gamma}} = \widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{V}}^{-1}(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})$. They can be useful for comparing the random effects from different experimental units and can also be treated as residuals in performing diagnostics for your mixed model.

The numbers displayed in the SE Pred column of the "Solution for Random Effects" table are not the standard errors of the $\widehat{\boldsymbol{\gamma}}$ displayed in the Estimate column; rather, they are the standard errors of predictions $\widehat{\boldsymbol{\gamma}}_i - \boldsymbol{\gamma}_i$, where $\widehat{\boldsymbol{\gamma}}_i$ is the *i*th EBLUP and $\boldsymbol{\gamma}_i$ is the *i*th random-effect parameter.

SUBJECT=*effect*

SUB=*effect*

identifies the subjects in your mixed model. Complete independence is assumed across subjects; thus, for the RANDOM statement, the SUBJECT= option produces a block-diagonal structure in $\mathbf{G}$ with identical blocks. In fact, specifying a subject effect is equivalent to nesting all other effects in the RANDOM statement within the subject effect.

When you specify the SUBJECT= option and a classification random effect, computations are usually much quicker if the levels of the random effect are duplicated within each level of the SUBJECT= effect.

TYPE=*covariance-structure*

specifies the covariance structure of $\mathbf{G}$. Valid values for *covariance-structure* and their descriptions are listed in Table 60.6. Although a variety of structures are available, most applications call for either TYPE=VC or TYPE=UN. The TYPE=VC (variance components) option is the default structure, and it models a different variance component for each random effect.

The TYPE=UN (unstructured) option is useful for correlated random coefficient models. For example, the following statement specifies a random intercept-slope model that has different variances for the intercept and slope and a covariance between them:

```
random intercept age / type=un subject=person;
```

You can also use TYPE=FA0(2) here to request a $\mathbf{G}$ estimate that is constrained to be nonnegative definite.

If you are constructing your own columns of $\mathbf{Z}$ with continuous variables, you can use the TYPE=TOEP(1) structure to group them together to have a common variance component. If you want to have different covariance structures in different parts of $\mathbf{G}$, you must use multiple RANDOM statements with different TYPE= options.

Table 60.6 Covariance Structures

Structure	Description	Parms	(i, j) element
ANTE(1)	Antedependence	$2t-1$	$\sigma_i\sigma_j \prod_{k=i}^{j-1} \rho_k$
AR(1)	Autoregressive(1)	2	$\sigma^2\rho^{\lvert i-j\rvert}$
ARH(1)	Heterogeneous AR(1)	$t+1$	$\sigma_i\sigma_j\rho^{\lvert i-j\rvert}$
ARMA(1,1)	Autoregressive moving average(1,1)	3	$\sigma^2[\gamma\rho^{\lvert i-j\rvert-1}1(i \neq j)+1(i=j)]$
CS	Compound symmetry	2	$\sigma_1+\sigma^2 1(i=j)$
CSH	Heterogeneous compound symmetry	$t+1$	$\sigma_i\sigma_j[\rho 1(i \neq j)+1(i=j)]$
FA(q)	Factor analytic	$\frac{q}{2}(2t-q+1)+t$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}+\sigma_i^2 1(i=j)$
FA0(q)	No diagonal FA	$\frac{q}{2}(2t-q+1)$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}$
FA1(q)	Equal diagonal FA	$\frac{q}{2}(2t-q+1)+1$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}+\sigma^2 1(i=j)$
HF	Huynh-Feldt	$t+1$	$(\sigma_i^2+\sigma_j^2)/2+\lambda 1(i \neq j)$
SIMPLE	An alias for VC	q	$\sigma_k^2 1(i=j)$ for the kth effect
TOEP	Toeplitz	t	$\sigma_{\lvert i-j\rvert+1}$
TOEP(q)	Banded Toeplitz	q	$\sigma_{\lvert i-j\rvert+1}1(\lvert i-j\rvert<q)$
TOEPH	Heterogeneous TOEP	$2t-1$	$\sigma_i\sigma_j\rho_{\lvert i-j\rvert}$
TOEPH(q)	Banded heterogeneous TOEP	$t+q-1$	$\sigma_i\sigma_j\rho_{\lvert i-j\rvert}1(\lvert i-j\rvert<q)$
UN	Unstructured	$t(t+1)/2$	σ_{ij}
UN(q)	Banded	$\frac{q}{2}(2t-q+1)$	$\sigma_{ij}1(\lvert i-j\rvert<q)$
UNR	Unstructured correlation	$t(t+1)/2$	$\sigma_i\sigma_j\rho_{\max(i,j)\min(i,j)}$
UNR(q)	Banded correlations	$\frac{q}{2}(2t-q+1)$	$\sigma_i\sigma_j\rho_{\max(i,j)\min(i,j)}$
VC	Variance components	q	$\sigma_k^2 1(i=j)$ for the kth effect

In Table 60.6, the Parms column represents the number of covariance parameters in the structure, t is the overall dimension of the covariance matrix, and $1(A)$ equals 1 when A is true and 0 otherwise. For example, $1(i=j)$ equals 1 when $i=j$ and 0 otherwise, and $1(\lvert i-j\rvert<q)$ equals 1 when $\lvert i-j\rvert<q$ and 0 otherwise. For the TYPE=TOEPH structures, $\rho_0=1$; for the TYPE=UNR structures, $\rho_{ii}=1$ for all i.

Table 60.7 lists some examples of the structures in Table 60.6.

Table 60.7 Covariance Structure Examples

Description	Structure	Example
Variance components	VC (default)	$\begin{bmatrix} \sigma_B^2 & 0 & 0 & 0 \\ 0 & \sigma_B^2 & 0 & 0 \\ 0 & 0 & \sigma_{AB}^2 & 0 \\ 0 & 0 & 0 & \sigma_{AB}^2 \end{bmatrix}$

Table 60.7 *continued*

Description	Structure	Example
Compound symmetry	CS	$\begin{bmatrix} \sigma^2+\sigma_1 & \sigma_1 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma^2+\sigma_1 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma^2+\sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma_1 & \sigma^2+\sigma_1 \end{bmatrix}$
Unstructured	UN	$\begin{bmatrix} \sigma_1^2 & \sigma_{21} & \sigma_{31} & \sigma_{41} \\ \sigma_{21} & \sigma_2^2 & \sigma_{32} & \sigma_{42} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 & \sigma_{43} \\ \sigma_{41} & \sigma_{42} & \sigma_{43} & \sigma_4^2 \end{bmatrix}$
Banded main diagonal	UN(1)	$\begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & 0 \\ 0 & 0 & 0 & \sigma_4^2 \end{bmatrix}$
First-order autoregressive	AR(1)	$\sigma^2 \begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 \\ \rho & 1 & \rho & \rho^2 \\ \rho^2 & \rho & 1 & \rho \\ \rho^3 & \rho^2 & \rho & 1 \end{bmatrix}$
Toeplitz	TOEP	$\begin{bmatrix} \sigma^2 & \sigma_1 & \sigma_2 & \sigma_3 \\ \sigma_1 & \sigma^2 & \sigma_1 & \sigma_2 \\ \sigma_2 & \sigma_1 & \sigma^2 & \sigma_1 \\ \sigma_3 & \sigma_2 & \sigma_1 & \sigma^2 \end{bmatrix}$
Toeplitz with two bands	TOEP(2)	$\begin{bmatrix} \sigma^2 & \sigma_1 & 0 & 0 \\ \sigma_1 & \sigma^2 & \sigma_1 & 0 \\ 0 & \sigma_1 & \sigma^2 & \sigma_1 \\ 0 & 0 & \sigma_1 & \sigma^2 \end{bmatrix}$
Heterogeneous autoregressive(1)	ARH(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho^2 & \sigma_1\sigma_4\rho^3 \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho^2 \\ \sigma_3\sigma_1\rho^2 & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho^3 & \sigma_4\sigma_2\rho^2 & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$
First-order autoregressive moving average	ARMA(1,1)	$\sigma^2 \begin{bmatrix} 1 & \gamma & \gamma\rho & \gamma\rho^2 \\ \gamma & 1 & \gamma & \gamma\rho \\ \gamma\rho & \gamma & 1 & \gamma \\ \gamma\rho^2 & \gamma\rho & \gamma & 1 \end{bmatrix}$
Heterogeneous compound symmetry	CSH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho & \sigma_1\sigma_4\rho \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho \\ \sigma_3\sigma_1\rho & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho & \sigma_4\sigma_2\rho & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$
First-order factor analytic	FA(1)	$\begin{bmatrix} \lambda_1^2+d_1 & \lambda_1\lambda_2 & \lambda_1\lambda_3 & \lambda_1\lambda_4 \\ \lambda_2\lambda_1 & \lambda_2^2+d_2 & \lambda_2\lambda_3 & \lambda_2\lambda_4 \\ \lambda_3\lambda_1 & \lambda_3\lambda_2 & \lambda_3^2+d_3 & \lambda_3\lambda_4 \\ \lambda_4\lambda_1 & \lambda_4\lambda_2 & \lambda_4\lambda_3 & \lambda_4^2+d_4 \end{bmatrix}$

Table 60.7 *continued*

Description	Structure	Example
Huynh-Feldt	HF	$\begin{bmatrix} \sigma_1^2 & \frac{\sigma_1^2+\sigma_2^2}{2}-\lambda & \frac{\sigma_1^2+\sigma_3^2}{2}-\lambda \\ \frac{\sigma_2^2+\sigma_1^2}{2}-\lambda & \sigma_2^2 & \frac{\sigma_2^2+\sigma_3^2}{2}-\lambda \\ \frac{\sigma_3^2+\sigma_1^2}{2}-\lambda & \frac{\sigma_3^2+\sigma_2^2}{2}-\lambda & \sigma_3^2 \end{bmatrix}$
First-order antedependence	ANTE(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_1 & \sigma_1\sigma_3\rho_1\rho_2 \\ \sigma_2\sigma_1\rho_1 & \sigma_2^2 & \sigma_2\sigma_3\rho_2 \\ \sigma_3\sigma_1\rho_2\rho_1 & \sigma_3\sigma_2\rho_2 & \sigma_3^2 \end{bmatrix}$
Heterogeneous Toeplitz	TOEPH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_1 & \sigma_1\sigma_3\rho_2 & \sigma_1\sigma_4\rho_3 \\ \sigma_2\sigma_1\rho_1 & \sigma_2^2 & \sigma_2\sigma_3\rho_1 & \sigma_2\sigma_4\rho_2 \\ \sigma_3\sigma_1\rho_2 & \sigma_3\sigma_2\rho_1 & \sigma_3^2 & \sigma_3\sigma_4\rho_1 \\ \sigma_4\sigma_1\rho_3 & \sigma_4\sigma_2\rho_2 & \sigma_4\sigma_3\rho_1 & \sigma_4^2 \end{bmatrix}$
Unstructured correlations	UNR	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_{21} & \sigma_1\sigma_3\rho_{31} & \sigma_1\sigma_4\rho_{41} \\ \sigma_2\sigma_1\rho_{21} & \sigma_2^2 & \sigma_2\sigma_3\rho_{32} & \sigma_2\sigma_4\rho_{42} \\ \sigma_3\sigma_1\rho_{31} & \sigma_3\sigma_2\rho_{32} & \sigma_3^2 & \sigma_3\sigma_4\rho_{43} \\ \sigma_4\sigma_1\rho_{41} & \sigma_4\sigma_2\rho_{42} & \sigma_4\sigma_3\rho_{43} & \sigma_4^2 \end{bmatrix}$

The following list provides some further information about these covariance structures:

TYPE=ANTE(1) specifies the first-order antedependence structure (Kenward 1987; Patel 1991; Macchiavelli and Arnold 1994). In Table 60.6, σ_i^2 is the i variance parameter, and ρ_k is the k autocorrelation parameter that satisfies $|\rho_k| < 1$.

TYPE=AR(1) specifies a first-order autoregressive structure. PROC HPLMIXED imposes the constraint $|\rho| < 1$ for stationarity.

TYPE=ARH(1) specifies a heterogeneous first-order autoregressive structure. As with TYPE=AR(1), PROC HPLMIXED imposes the constraint $|\rho| < 1$ for stationarity.

TYPE=ARMA(1,1) specifies the first-order autoregressive moving average structure. In Table 60.6, ρ is the autoregressive parameter, γ models a moving average component, and σ^2 is the residual variance. In the notation of Fuller (1976, p. 68), $\rho = \theta_1$ and

$$\gamma = \frac{(1 + b_1\theta_1)(\theta_1 + b_1)}{1 + b_1^2 + 2b_1\theta_1}$$

The example in Table 60.7 and $|b_1| < 1$ imply that

$$b_1 = \frac{\beta - \sqrt{\beta^2 - 4\alpha^2}}{2\alpha}$$

where $\alpha = \gamma - \rho$ and $\beta = 1 + \rho^2 - 2\gamma\rho$. PROC HPLMIXED imposes the constraints $|\rho| < 1$ and $|\gamma| < 1$ for stationarity, although the resulting covariance matrix is not positive definite for some values of ρ and γ in this region. When the estimated value of ρ becomes negative, the computed covariance is multiplied by $\cos(\pi d_{ij})$ to account for the negativity.

TYPE=CS specifies the compound-symmetry structure, which has constant variance and constant covariance.

TYPE=CSH specifies the heterogeneous compound-symmetry structure. This structure has a different variance parameter for each diagonal element, and it uses the square roots of these parameters in the off-diagonal entries. In Table 60.6, σ_i^2 is the i variance parameter, and ρ is the correlation parameter that satisfies $|\rho| < 1$.

TYPE=FA(q) specifies the factor-analytic structure with q factors (Jennrich and Schluchter 1986). This structure is of the form $\mathbf{\Lambda}\mathbf{\Lambda}' + \mathbf{D}$, where $\mathbf{\Lambda}$ is a $t \times q$ rectangular matrix and $\mathbf{D}$ is a $t \times t$ diagonal matrix with t different parameters. When $q > 1$, the elements of $\mathbf{\Lambda}$ in its upper right corner (that is, the elements in the i row and j column for $j > i$) are set to zero to fix the rotation of the structure.

TYPE=FA0(q) is similar to the FA(q) structure except that no diagonal matrix $\mathbf{D}$ is included. When $q < t$ (that is, when the number of factors is less than the dimension of the matrix), this structure is nonnegative definite but not of full rank. In this situation, you can use this structure for approximating an unstructured $\mathbf{G}$ matrix in the RANDOM statement. When $q = t$, you can use this structure to constrain $\mathbf{G}$ to be nonnegative definite in the RANDOM statement.

TYPE=FA1(q) is similar to the TYPE=FA(q) structure except that all of the elements in $\mathbf{D}$ are constrained to be equal. This offers a useful and more parsimonious alternative to the full factor-analytic structure.

TYPE=HF specifies the Huynh-Feldt covariance structure (Huynh and Feldt 1970). This structure is similar to the TYPE=CSH structure in that it has the same number of parameters and heterogeneity along the main diagonal. However, it constructs the off-diagonal elements by taking arithmetic means rather than geometric means.

You can perform a likelihood ratio test of the Huynh-Feldt conditions by running PROC HPLMIXED twice, once with TYPE=HF and once with TYPE=UN, and then subtracting their respective values of -2 times the maximized likelihood.

If PROC HPLMIXED does not converge under your Huynh-Feldt model, you can specify your own starting values with the PARMS statement. The default MIVQUE(0) starting values can sometimes be poor for this structure. A good choice for starting values is often the parameter estimates that correspond to an initial fit that uses TYPE=CS.

TYPE=SIMPLE is an alias for TYPE=VC.

TYPE=TOEP<(q)> specifies a banded Toeplitz structure. This can be viewed as a moving average structure with order equal to $q - 1$. The TYPE=TOEP option is a full Toeplitz matrix, which can be viewed as an autoregressive structure with order equal to the dimension of the matrix. The specification TYPE=TOEP(1) is the same as $\sigma^2 I$, where I is an identity matrix, and it can be useful for specifying the same variance component for several effects.

TYPE=TOEPH<(q)> specifies a heterogeneous banded Toeplitz structure. In Table 60.6, σ_i^2 is the i variance parameter and ρ_j is the j correlation parameter that satisfies $|\rho_j| < 1$. If you specify the order parameter q, then PROC HPLMIXED estimates only the first q bands of the matrix, setting all higher bands equal to 0. The option TOEPH(1) is equivalent to both the TYPE=UN(1) and TYPE=UNR(1) options.

TYPE=UN<(*q*)> specifies a completely general (unstructured) covariance matrix that is parameterized directly in terms of variances and covariances. The variances are constrained to be nonnegative, and the covariances are unconstrained. This structure is not constrained to be nonnegative definite in order to avoid nonlinear constraints. However, you can use the TYPE=FA0 structure if you want this constraint to be imposed by a Cholesky factorization. If you specify the order parameter *q*, then PROC HPLMIXED estimates only the first *q* bands of the matrix, setting all higher bands equal to 0.

TYPE=UNR<(*q*)> specifies a completely general (unstructured) covariance matrix that is parameterized in terms of variances and correlations. This structure fits the same model as the TYPE=UN(*q*) option but with a different parameterization. The *i* variance parameter is σ_i^2. The parameter ρ_{jk} is the correlation between the *j* and *k* measurements; it satisfies $|\rho_{jk}| < 1$. If you specify the order parameter *r*, then PROC HPLMIXED estimates only the first *q* bands of the matrix, setting all higher bands equal to zero.

TYPE=VC specifies standard variance components. This is the default structure for both the RANDOM and REPEATED statements. In the RANDOM statement, a distinct variance component is assigned to each effect.

Jennrich and Schluchter (1986) provide general information about the use of covariance structures, and Wolfinger (1996) presents details about many of the heterogeneous structures.

REPEATED Statement

REPEATED *repeated-effect* < / *options* > **;**

The REPEATED statement specifies the $\mathbf{R}$ matrix in the mixed model. If no REPEATED statement is specified, $\mathbf{R}$ is assumed to be equal to $\sigma^2\mathbf{I}$. You can use the REPEATED statement only with the BLUP option. The statement is ignored when no BLUP option is specified.

The *repeated-effect* is required. The *repeated-effect* must contain only classification variables. Make sure that the levels of the *repeated-effect* are different for each observation within a subject; otherwise, PROC HPLMIXED constructs identical rows in $\mathbf{R}$ that correspond to the observations with the same level. This results in a singular $\mathbf{R}$ matrix and an infinite likelihood.

Table 60.8 summarizes important options in the REPEATED statement. All options are subsequently discussed in alphabetical order.

Table 60.8 Summary of Important REPEATED Statement Options

Option	Description
Construction of Covariance Structure	
SUBJECT=	Identifies the subjects in the R-side model
TYPE=	Specifies the R-side covariance structure

You can specify the following *options* in the REPEATED statement after a slash (/).

SUBJECT=*effect*

SUB=*effect*

identifies the subjects in your mixed model. Complete independence is assumed across subjects; therefore, the SUBJECT= option produces a block-diagonal structure in $\mathbf{R}$ with identical blocks. The blocks of $\mathbf{R}$ correspond to observations that share the same level of SUBJECT= effect. These blocks are sorted according to this effect as well.

If you want to model nonzero covariance among all of the observations in your SAS data set, specify SUBJECT=Dummy_Intercept to treat the data as if they are all from one subject. You need to create this Dummy_Intercept variable in the data set. However, be aware that in this case PROC HPLMIXED manipulates an $\mathbf{R}$ matrix with dimensions equal to the number of observations.

TYPE=*covariance-structure*

specifies the covariance structure of the $\mathbf{R}$ matrix. The SUBJECT= option defines the blocks of $\mathbf{R}$, and the TYPE= option specifies the structure of these blocks. The default structure is VC. You can specify any of the covariance structures that are described in the TYPE= option in the RANDOM statement.

Details: HPLMIXED Procedure

Linear Mixed Models Theory

This section provides an overview of a likelihood-based approach to linear mixed models. This approach simplifies and unifies many common statistical analyses, including those that involve repeated measures, random effects, and random coefficients. The basic assumption is that the data are linearly related to unobserved multivariate normal random variables. For extensions to nonlinear and nonnormal situations, see the documentation of the GLIMMIX and NLMIXED procedures in the *SAS/STAT User's Guide*. Additional theory and examples are provided in Littell et al. (2006); Verbeke and Molenberghs (1997, 2000); and Burdick and Graybill (1992).

Matrix Notation

Suppose that you observe n data points $y_1, \ldots, y_n$ and that you want to explain them by using n values for each of p explanatory variables $x_{11}, \ldots, x_{1p}, x_{21}, \ldots, x_{2p}, \ldots, x_{n1}, \ldots, x_{np}$. The x_{ij} values can be either regression-type continuous variables or dummy variables that indicate class membership. The standard linear model for this setup is

$$y_i = \sum_{j=1}^{p} x_{ij}\beta_j + \epsilon_i \quad i = 1, \ldots, n$$

where $\beta_1, \ldots, \beta_p$ are unknown fixed-effects parameters to be estimated and $\epsilon_1, \ldots, \epsilon_n$ are unknown independent and identically distributed normal (Gaussian) random variables with mean 0 and variance σ^2.

The preceding equations can be written simultaneously by using vectors and a matrix, as follows:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

For convenience, simplicity, and extendability, this entire system is written as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\mathbf{y}$ denotes the vector of observed y_i's, $\mathbf{X}$ is the known matrix of x_{ij}'s, $\boldsymbol{\beta}$ is the unknown fixed-effects parameter vector, and $\boldsymbol{\epsilon}$ is the unobserved vector of independent and identically distributed Gaussian random errors.

In addition to denoting data, random variables, and explanatory variables in the preceding fashion, the subsequent development makes use of basic matrix operators such as transpose ($'$), inverse ($^{-1}$), generalized inverse ($^{-}$), determinant ($|\cdot|$), and matrix multiplication. See Searle (1982) for details about these and other matrix techniques.

Formulation of the Mixed Model

The previous general linear model is certainly a useful one (Searle 1971), and it is the one fitted by the GLM procedure. However, many times the distributional assumption about ϵ is too restrictive. The mixed model extends the general linear model by allowing a more flexible specification of the covariance matrix of ϵ. In other words, it allows for both correlation and heterogeneous variances, although you still assume normality.

The mixed model is written as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

where everything is the same as in the general linear model except for the addition of the known design matrix, $\mathbf{Z}$, and the vector of unknown random-effects parameters, $\boldsymbol{\gamma}$. The matrix $\mathbf{Z}$ can contain either continuous or dummy variables, just like $\mathbf{X}$. The name *mixed model* comes from the fact that the model contains both fixed-effects parameters, $\boldsymbol{\beta}$, and random-effects parameters, $\boldsymbol{\gamma}$. See Henderson (1990) and Searle, Casella, and McCulloch (1992) for historical developments of the mixed model.

A key assumption in the foregoing analysis is that $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$ are normally distributed with

$$\mathrm{E}\begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\epsilon} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

$$\mathrm{Var}\begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\epsilon} \end{bmatrix} = \begin{bmatrix} \mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{bmatrix}$$

Therefore, the variance of $\mathbf{y}$ is $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$. You can model $\mathbf{V}$ by setting up the random-effects design matrix $\mathbf{Z}$ and by specifying covariance structures for $\mathbf{G}$ and $\mathbf{R}$.

Note that this is a general specification of the mixed model, in contrast to many texts and articles that discuss only simple random effects. Simple random effects are a special case of the general specification with $\mathbf{Z}$ containing dummy variables, $\mathbf{G}$ containing variance components in a diagonal structure, and $\mathbf{R} = \sigma^2\mathbf{I}_n$, where $\mathbf{I}_n$ denotes the $n \times n$ identity matrix. The general linear model is a further special case with $\mathbf{Z} = \mathbf{0}$ and $\mathbf{R} = \sigma^2\mathbf{I}_n$.

The following two examples illustrate the most common formulations of the general linear mixed model.

Example: Growth Curve with Compound Symmetry

Suppose that you have three growth curve measurements for s individuals and that you want to fit an overall linear trend in time. Your $\mathbf{X}$ matrix is as follows:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ \vdots & \vdots \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}$$

The first column (coded entirely with 1s) fits an intercept, and the second column (coded with series of $1, 2, 3$) fits a slope. Here, $n = 3s$ and $p = 2$.

Suppose further that you want to introduce a common correlation among the observations from a single individual, with correlation being the same for all individuals. One way of setting this up in the general mixed model is to eliminate the $\mathbf{Z}$ and $\mathbf{G}$ matrices and let the $\mathbf{R}$ matrix be block-diagonal with blocks corresponding to the individuals and with each block having the *compound-symmetry* structure. This structure has two unknown parameters, one modeling a common covariance and the other modeling a residual variance. The form for $\mathbf{R}$ would then be

$$\mathbf{R} = \begin{bmatrix} \sigma_1^2 + \sigma^2 & \sigma_1^2 & \sigma_1^2 & & & & \\ \sigma_1^2 & \sigma_1^2 + \sigma^2 & \sigma_1^2 & & & & \\ \sigma_1^2 & \sigma_1^2 & \sigma_1^2 + \sigma^2 & & & & \\ & & & \ddots & & & \\ & & & & \sigma_1^2 + \sigma^2 & \sigma_1^2 & \sigma_1^2 \\ & & & & \sigma_1^2 & \sigma_1^2 + \sigma^2 & \sigma_1^2 \\ & & & & \sigma_1^2 & \sigma_1^2 & \sigma_1^2 + \sigma^2 \end{bmatrix}$$

where blanks denote zeros. There are $3s$ rows and columns altogether, and the common correlation is $\sigma_1^2/(\sigma_1^2 + \sigma^2)$.

The following PROC HPLMIXED statements fit this model:

```
proc hplmixed;
   class indiv;
   model y = time;
   repeated morder/ type=cs subject=indiv;
run;
```

Here, INDIV is a classification variable that indexes individuals. The MODEL statement fits a straight line for TIME ; the intercept is fit by default just as in PROC GLM. The REPEATED statement models the $\mathbf{R}$ matrix: TYPE=CS specifies the compound symmetry structure, and SUBJECT=INDIV specifies the blocks of $\mathbf{R}$, and MORDER is the repeated effect that records the order of the measurements for each individual.

An alternative way of specifying the common intra-individual correlation is to let

$$\mathbf{Z} = \begin{bmatrix} 1 & & & \\ 1 & & & \\ 1 & & & \\ & 1 & & \\ & 1 & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \\ & & & 1 \\ & & & 1 \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_1^2 & & \\ & & \ddots & \\ & & & \sigma_1^2 \end{bmatrix}$$

and $\mathbf{R} = \sigma^2 \mathbf{I}_n$. The $\mathbf{Z}$ matrix has $3s$ rows and s columns, and $\mathbf{G}$ is $s \times s$.

You can set up this model in PROC HPLMIXED in two different but equivalent ways:

```
proc hplmixed;
   class indiv;
   model y = time;
   random indiv;
run;

proc hplmixed;
   class indiv;
   model y = time;
   random intercept / subject=indiv;
run;
```

Both of these specifications fit the same model as the previous one that used the REPEATED statement. However, the RANDOM specifications constrain the correlation to be positive, whereas the REPEATED specification leaves the correlation unconstrained.

Example: Split-Plot Design

The split-plot design involves two experimental treatment factors, A and B, and two different sizes of experimental units to which they are applied (Winer 1971; Snedecor and Cochran 1980; Milliken and Johnson 1992; Steel, Torrie, and Dickey 1997). The levels of A are randomly assigned to the larger-sized experimental units, called *whole plots*, whereas the levels of B are assigned to the smaller-sized experimental units, the *subplots*. The subplots are assumed to be nested within the whole plots, so that a whole plot consists of a cluster of subplots and a level of A is applied to the entire cluster.

Such an arrangement is often necessary by nature of the experiment; the classical example is the application of fertilizer to large plots of land and different crop varieties planted in subdivisions of the large plots. For this example, fertilizer is the whole-plot factor A and variety is the subplot factor B.

The first example is a split-plot design for which the whole plots are arranged in a randomized block design. The appropriate PROC HPLMIXED statements are as follows:

```
proc hplmixed;
   class a b block;
   model y = a b a*b;
   random block a*block;
run;
```

Here

$$\mathbf{R} = \sigma^2 \mathbf{I}_{24}$$

and **X**, **Z**, and **G** have the following form:

$$\mathbf{X} = \begin{bmatrix}
1 & 1 & & & 1 & & 1 & & & & & \\
1 & 1 & & & & 1 & & 1 & & & & \\
1 & & 1 & & 1 & & & & 1 & & & \\
1 & & 1 & & & 1 & & & & 1 & & \\
1 & & & 1 & 1 & & & & & & 1 & \\
1 & & & 1 & & 1 & & & & & & 1 \\
\vdots & & \vdots & & \vdots & & & & & \vdots & & \\
1 & 1 & & & 1 & & 1 & & & & & \\
1 & 1 & & & & 1 & & 1 & & & & \\
1 & & 1 & & 1 & & & & 1 & & & \\
1 & & 1 & & & 1 & & & & 1 & & \\
1 & & & 1 & 1 & & & & & & 1 & \\
1 & & & 1 & & 1 & & & & & & 1
\end{bmatrix}$$

$$
\mathbf{Z} = \begin{bmatrix}
1 & & & & 1 & & & &

$$
\mathbf{Z} = \begin{bmatrix}
1 & 1 & & & & & & & & & & & & & & & & & & & & & & \\
1 & 1 & & & & & & & & & & & & & & & & & & & & & & \\
1 & & 1 & & & & & & & & & & & & & & & & & & & & & \\
1 & & 1 & & & & & & & & & & & & & & & & & & & & & \\
1 & & & 1 & & & & & & & & & & & & & & & & & & & & \\
1 & & & 1 & & & & & & & & & & & & & & & & & & & & \\
 & & & & 1 & 1 & & & & & & & & & & & & & & & & & & \\
 & & & & 1 & 1 & & & & & & & & & & & & & & & & & & \\
 & & & & 1 & & 1 & & & & & & & & & & & & & & & & & \\
 & & & & 1 & & 1 & & & & & & & & & & & & & & & & & \\
 & & & & 1 & & & 1 & & & & & & & & & & & & & & & & \\
 & & & & 1 & & & 1 & & & & & & & & & & & & & & & & \\
 & & & & & & & & 1 & 1 & & & & & & & & & & & & & & \\
 & & & & & & & & 1 & 1 & & & & & & & & & & & & & & \\
 & & & & & & & & 1 & & 1 & & & & & & & & & & & & & \\
 & & & & & & & & 1 & & 1 & & & & & & & & & & & & & \\
 & & & & & & & & 1 & & & 1 & & & & & & & & & & & & \\
 & & & & & & & & 1 & & & 1 & & & & & & & & & & & & \\
 & & & & & & & & & & & & 1 & 1 & & & & & & & & & & \\
 & & & & & & & & & & & & 1 & 1 & & & & & & & & & & \\
 & & & & & & & & & & & & 1 & & 1 & & & & & & & & & \\
 & & & & & & & & & & & & 1 & & 1 & & & & & & & & & \\
 & & & & & & & & & & & & 1 & & & 1 & & & & & & & & \\
 & & & & & & & & & & & & 1 & & & 1 & & & & & & & & \\
 & & & & & & & & & & & & & & & & 1 & & & & & & & \\
 & & & & & & & & & & & & & & & & 1 & & & & & & & \\
 & & & & & & & & & & & & & & & & & 1 & & & & & & \\
 & & & & & & & & & & & & & & & & & 1 & & & & & & \\
 & & & & & & & & & & & & & & & & & & 1 & & & & & \\
 & & & & & & & & & & & & & & & & & & 1 & & & & & \\
 & & & & & & & & & & & & & & & & & & & 1 & & & & \\
 & & & & & & & & & & & & & & & & & & & 1 & & & & \\
 & & & & & & & & & & & & & & & & & & & & 1 & & & \\
 & & & & & & & & & & & & & & & & & & & & 1 & & & \\
 & & & & & & & & & & & & & & & & & & & & & 1 & & \\
 & & & & & & & & & & & & & & & & & & & & & 1 & & \\
 & & & & & & & & & & & & & & & & & & & & & & 1 & \\
 & & & & & & & & & & & & & & & & & & & & & & 1 &
\end{bmatrix}
$$

$$
\mathbf{G} = \begin{bmatrix}
\sigma_B^2 & & & & & & & & & & & & & & & & & & & & & & & \\
 & \sigma_{AB}^2 & & & & & & & & & & & & & & & & & & & & & & \\
 & & \sigma_{AB}^2 & & & & & & & & & & & & & & & & & & & & & \\
 & & & \sigma_{AB}^2 & & & & & & & & & & & & & & & & & & & & \\
 & & & & \ddots & & & & & & & & & & & & & & & & & & & \\
 & & & & & \sigma_B^2 & & & & & & & & & & & & & & & & & & \\
 & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & & & & & \\
 & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & & & & \\
 & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & & & \\
 & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & & \\
 & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & \\
 & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & \\
 & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & \\
 & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & \\
 & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & \\
 & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & \\
 & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & \\
 & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & \\
 & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & \\
 & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & \\
 & & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & \\
 & & & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & \\
 & & & & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & \\
 & & & & & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2
\end{bmatrix}
$$

Estimating Covariance Parameters in the Mixed Model

Estimation is more difficult in the mixed model than in the general linear model. Not only do you have $\boldsymbol{\beta}$ as in the general linear model, but you also have unknown parameters in $\boldsymbol{\gamma}$, $\mathbf{G}$, and $\mathbf{R}$. Least squares is no longer the best method. Generalized least squares (GLS) is more appropriate, minimizing

$$(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'\mathbf{V}^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

However, GLS requires knowledge of $\mathbf{V}$ and therefore knowledge of $\mathbf{G}$ and $\mathbf{R}$. Lacking such information, one approach is to use an *estimated* GLS, in which you insert some reasonable estimate for $\mathbf{V}$ into the minimization problem. The goal thus becomes finding a reasonable estimate of $\mathbf{G}$ and $\mathbf{R}$.

In many situations, the best approach is to use likelihood-based methods, exploiting the assumption that $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$ are normally distributed (Hartley and Rao 1967; Patterson and Thompson 1971; Harville 1977; Laird and Ware 1982; Jennrich and Schluchter 1986). PROC HPLMIXED implements two likelihood-based methods: maximum likelihood (ML) and restricted (residual) maximum likelihood (REML). A favorable theoretical property of ML and REML is that they accommodate data that are missing at random (Rubin 1976; Little 1995).

PROC HPLMIXED constructs an objective function associated with ML or REML and maximizes it over all unknown parameters. Using calculus, it is possible to reduce this maximization problem to one over only the parameters in $\mathbf{G}$ and $\mathbf{R}$. The corresponding log-likelihood functions are as follows:

$$\begin{aligned} \text{ML}: \quad & l(\mathbf{G},\mathbf{R}) = -\frac{1}{2}\log|\mathbf{V}| - \frac{1}{2}\mathbf{r}'\mathbf{V}^{-1}\mathbf{r} - \frac{n}{2}\log(2\pi) \\ \text{REML}: \quad & l_R(\mathbf{G},\mathbf{R}) = -\frac{1}{2}\log|\mathbf{V}| - \frac{1}{2}\log|\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}| - \frac{1}{2}\mathbf{r}'\mathbf{V}^{-1}\mathbf{r} - \frac{n-p}{2}\log(2\pi) \end{aligned}$$

where $\mathbf{r} = \mathbf{y} - \mathbf{X}(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{y}$ and p is the rank of $\mathbf{X}$. By default, PROC HPLMIXED actually minimizes a normalized form of -2 times these functions by using a ridge-stabilized Newton-Raphson algorithm by default. Lindstrom and Bates (1988) provide reasons for preferring Newton-Raphson to the expectation-maximum (EM) algorithm described in Dempster, Laird, and Rubin (1977) and Laird, Lange, and Stram (1987), in addition to analytical details for implementing a QR-decomposition approach to the problem. Wolfinger, Tobias, and Sall (1994) present the sweep-based algorithms that are implemented in PROC HPLMIXED. You can change the optimization technique with the TECHNIQUE= option in the PROC HPLMIXED statement.

One advantage of using the Newton-Raphson algorithm is that the second derivative matrix of the objective function evaluated at the optima is available upon completion. Denoting this matrix $\mathbf{H}$, the asymptotic theory of maximum likelihood (Serfling 1980) shows that $2\mathbf{H}^{-1}$ is an asymptotic variance-covariance matrix of the estimated parameters of $\mathbf{G}$ and $\mathbf{R}$. Thus, tests and confidence intervals based on asymptotic normality can be obtained. However, these can be unreliable in small samples, especially for parameters such as variance components that have sampling distributions that tend to be skewed to the right.

If a residual variance σ^2 is a part of your mixed model, it can usually be *profiled* out of the likelihood. This means solving analytically for the optimal σ^2 and plugging this expression back into the likelihood formula (Wolfinger, Tobias, and Sall 1994). This reduces the number of optimization parameters by 1 and can improve convergence properties. PROC HPLMIXED profiles the residual variance out of the log likelihood.

Estimating Fixed and Random Effects in the Mixed Model

ML and REML methods provide estimates of $\mathbf{G}$ and $\mathbf{R}$, which are denoted $\widehat{\mathbf{G}}$ and $\widehat{\mathbf{R}}$, respectively. To obtain estimates of $\boldsymbol{\beta}$ and predicted values of $\boldsymbol{\gamma}$, the standard method is to solve the *mixed model equations* (Henderson 1984):

$$\begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} + \widehat{\mathbf{G}}^{-1} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \end{bmatrix}$$

The solutions can also be written as

$$\begin{aligned} \widehat{\boldsymbol{\beta}} &= (\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{y} \\ \widehat{\boldsymbol{\gamma}} &= \widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{V}}^{-1}(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}) \end{aligned}$$

and have connections with empirical Bayes estimators (Laird and Ware 1982; Carlin and Louis 1996). Note that the $\boldsymbol{\gamma}$ are random variables and not parameters (unknown constants) in the model. Technically, determining values for $\boldsymbol{\gamma}$ from the data is thus a prediction task, whereas determining values for $\boldsymbol{\beta}$ is an estimation task.

The mixed model equations are extended normal equations. The preceding expression assumes that $\widehat{\mathbf{G}}$ is nonsingular. For the extreme case where the eigenvalues of $\widehat{\mathbf{G}}$ are very large, $\widehat{\mathbf{G}}^{-1}$ contributes very little to the equations and $\widehat{\boldsymbol{\gamma}}$ is close to what it would be if $\boldsymbol{\gamma}$ actually contained fixed-effects parameters. On the other hand, when the eigenvalues of $\widehat{\mathbf{G}}$ are very small, $\widehat{\mathbf{G}}^{-1}$ dominates the equations and $\widehat{\boldsymbol{\gamma}}$ is close to 0. For intermediate cases, $\widehat{\mathbf{G}}^{-1}$ can be viewed as shrinking the fixed-effects estimates of $\boldsymbol{\gamma}$ toward 0 (Robinson 1991).

If $\widehat{\mathbf{G}}$ is singular, then the mixed model equations are modified (Henderson 1984) as follows:

$$\begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}\widehat{\mathbf{G}} \\ \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}\widehat{\mathbf{G}} + \mathbf{G} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\tau}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \\ \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \end{bmatrix}$$

Denote the generalized inverses of the nonsingular $\widehat{\mathbf{G}}$ and singular $\widehat{\mathbf{G}}$ forms of the mixed model equations by $\mathbf{C}$ and $\mathbf{M}$, respectively. In the nonsingular case, the solution $\widehat{\boldsymbol{\gamma}}$ estimates the random effects directly. But in the singular case, the estimates of random effects are achieved through a back-transformation $\widehat{\boldsymbol{\gamma}} = \widehat{\mathbf{G}}\widehat{\boldsymbol{\tau}}$ where $\widehat{\boldsymbol{\tau}}$ is the solution to the modified mixed model equations. Similarly, while in the nonsingular case $\mathbf{C}$ itself is the estimated covariance matrix for $(\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\gamma}})$, in the singular case the covariance estimate for $(\widehat{\boldsymbol{\beta}}, \widehat{\mathbf{G}}\widehat{\boldsymbol{\tau}})$ is given by $\mathbf{PMP}$ where

$$\mathbf{P} = \begin{bmatrix} \mathbf{I} & \\ & \widehat{\mathbf{G}} \end{bmatrix}$$

An example of when the singular form of the equations is necessary is when a variance component estimate falls on the boundary constraint of 0.

Statistical Properties

If $\mathbf{G}$ and $\mathbf{R}$ are known, $\widehat{\boldsymbol{\beta}}$ is the best linear unbiased estimator (BLUE) of $\boldsymbol{\beta}$, and $\widehat{\boldsymbol{\gamma}}$ is the best linear unbiased predictor (BLUP) of $\boldsymbol{\gamma}$ (Searle 1971; Harville 1988, 1990; Robinson 1991; McLean, Sanders, and Stroup 1991). Here, "best" means minimum mean squared error. The covariance matrix of $(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}, \widehat{\boldsymbol{\gamma}} - \boldsymbol{\gamma})$ is

$$\mathbf{C} = \begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{R}^{-1}\mathbf{Z} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z} + \mathbf{G}^{-1} \end{bmatrix}^{-}$$

where $^{-}$ denotes a generalized inverse (Searle 1971).

However, $\mathbf{G}$ and $\mathbf{R}$ are usually unknown and are estimated by using one of the aforementioned methods. These estimates, $\widehat{\mathbf{G}}$ and $\widehat{\mathbf{R}}$, are therefore simply substituted into the preceding expression to obtain

$$\widehat{\mathbf{C}} = \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} + \widehat{\mathbf{G}}^{-1} \end{bmatrix}^{-}$$

as the approximate variance-covariance matrix of $(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}, \widehat{\boldsymbol{\gamma}} - \boldsymbol{\gamma})$. In this case, the BLUE and BLUP acronyms no longer apply, but the word *empirical* is often added to indicate such an approximation. The appropriate acronyms thus become EBLUE and EBLUP.

McLean and Sanders (1988) show that $\widehat{\mathbf{C}}$ can also be written as

$$\widehat{\mathbf{C}} = \begin{bmatrix} \widehat{\mathbf{C}}_{11} & \widehat{\mathbf{C}}_{21}' \\ \widehat{\mathbf{C}}_{21} & \widehat{\mathbf{C}}_{22} \end{bmatrix}$$

where

$$\begin{aligned} \widehat{\mathbf{C}}_{11} &= (\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-} \\ \widehat{\mathbf{C}}_{21} &= -\widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{V}}^{-1}\mathbf{X}\widehat{\mathbf{C}}_{11} \\ \widehat{\mathbf{C}}_{22} &= (\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} + \widehat{\mathbf{G}}^{-1})^{-1} - \widehat{\mathbf{C}}_{21}\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{Z}\widehat{\mathbf{G}} \end{aligned}$$

Note that $\widehat{\mathbf{C}}_{11}$ is the familiar estimated generalized least squares formula for the variance-covariance matrix of $\widehat{\boldsymbol{\beta}}$.

Computational Method

Multithreading

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to sequential (single-threaded) execution.

The number of threads spawned by the HPLMIXED procedure is determined by the number of cores on a machine and can be controlled in the following ways:

You can specify the NTHREADS= option in the PERFORMANCE statement to determine the number of threads. This specification overrides the system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads is displayed in the “Performance Information” table, which is part of the default output. The HPLMIXED procedure allocates two threads per core.

The tasks multithreaded by the HPLMIXED procedure are primarily defined by dividing the data processed on a single machine among the threads—that is, the HPLMIXED procedure implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and you are running with four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following:

- variable levelization
- effect levelization
- formation of the crossproducts matrix
- the log-likelihood computation

In addition, operations on matrices such as sweeps might be multithreaded if the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Displayed Output

The following sections describe the output produced by PROC HPLMIXED. The output is organized into various tables, which are discussed in the order of their appearance.

Performance Information

The "Performance Information" table is produced by default. The table displays the number of threads used.

If you specify the DETAILS option in the PERFORMANCE statement, PROC HPLMIXED also produces a "Timing" table that displays elapsed times (absolute and relative) for the main tasks of the procedure.

Model Information

The "Model Information" table describes the model, some of the variables it involves, and the method used in fitting it. The "Model Information" table also has a row labeled Fixed Effects SE Method. This row describes the method used to compute the approximate standard errors for the fixed-effects parameter estimates and related functions of them.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement.

Dimensions

The "Dimensions" table lists the sizes of relevant matrices. This table can be useful in determining the requirements for CPU time and memory.

Number of Observations

The "Number of Observations" table shows the number of observations read from the data set and the number of observations used in fitting the model.

Optimization Information

The "Optimization Information" table displays important details about the optimization process.

The number of parameters that are updated in the optimization equals the number of parameters in this table minus the number of equality constraints. The number of constraints is displayed if you fix covariance parameters with the HOLD= option in the PARMS statement. The HPLMIXED procedure also lists the number of upper and lower boundary constraints. PROC HPLMIXED might impose boundary constraints for certain parameters, such as variance components and correlation parameters. If you specify the HOLD= option in the PARMS statement, covariance parameters have an upper and lower boundary equal to the parameter value.

Iteration History

The "Iteration History" table describes the optimization of the restricted log likelihood or log likelihood. The function to be minimized (the *objective function*) is $-2l$ for ML and $-2l_R$ for REML; the column name of the objective function in the "Iteration History" table is "-2 Log Like" for ML and "-2 Res Log Like" for REML. The minimization is performed by using a ridge-stabilized Newton-Raphson algorithm, and the rows of this table describe the iterations that this algorithm takes in order to minimize the objective function.

The Evaluations column of the "Iteration History" table tells how many times the objective function is evaluated during each iteration.

The Criterion column of the "Iteration History" table is, by default, a relative Hessian convergence quantity given by

$$\frac{\mathbf{g}_k' \mathbf{H}_k^{-1} \mathbf{g}_k}{|f_k|}$$

where f_k is the value of the objective function at iteration k, $\mathbf{g}_k$ is the gradient (first derivative) of f_k, and $\mathbf{H}_k$ is the Hessian (second derivative) of f_k. If $\mathbf{H}_k$ is singular, then PROC HPLMIXED uses the following relative quantity:

$$\frac{\mathbf{g}_k' \mathbf{g}_k}{|f_k|}$$

To prevent division by $|f_k|$, specify the ABSGCONV option in the PROC HPLMIXED statement. To use a relative function or gradient criterion, specify the FCONV or GCONV option, respectively.

The Hessian criterion is considered superior to function and gradient criteria because it measures orthogonality rather than lack of progress (Bates et al. 1987). Provided that the initial estimate is feasible and the maximum number of iterations is not exceeded, the Newton-Raphson algorithm is considered to have converged when the criterion is less than the tolerance specified with the FCONV or GCONV option in the PROC HPLMIXED statement. The default tolerance is 1E–8. If convergence is not achieved, PROC HPLMIXED displays the estimates of the parameters at the last iteration.

A convergence criterion that is missing indicates that a boundary constraint has been dropped; it is usually not a cause for concern.

Convergence Status

The "Convergence Status" table displays the status of the iterative estimation process at the end of the optimization. The status appears as a message in the listing, and this message is repeated in the log. The ODS object "ConvergenceStatus" also contains several nonprinting columns that can be helpful in checking the success of the iterative process, in particular during batch processing. The Status variable takes on the value 0 for a successful convergence (even if the Hessian matrix might not be positive definite). The values 1 and 2 of the Status variable indicate lack of convergence and infeasible initial parameter values, respectively. The variable pdG can be used to check whether the **G** matrix is positive definite.

For models that are not fit iteratively, such as models without random effects or when the NOITER option is in effect, the "Convergence Status" is not produced.

Covariance Parameter Estimates

The "Covariance Parameter Estimates" table contains the estimates of the parameters in **G** and **R**. (See the section "Estimating Covariance Parameters in the Mixed Model" on page 4830.) Their values are labeled in the table along with Subject information if applicable. The estimates are displayed in the Estimate column and are the results of either the REML or the ML estimation method.

Fit Statistics

The "Fit Statistics" table provides some statistics about the estimated mixed model. Expressions for -2 times the log likelihood are provided in the section "Estimating Covariance Parameters in the Mixed Model" on page 4830. If the log likelihood is an extremely large number, then PROC HPLMIXED has deemed the estimated **V** matrix to be singular. In this case, all subsequent results should be viewed with caution.

In addition, the "Fit Statistics" table lists three information criteria: AIC, AICC, and BIC. All these criteria are in smaller-is-better form and are described in Table 60.9.

Table 60.9 Information Criteria

Criterion	Formula	Reference
AIC	$-2\ell + 2d$	Akaike (1974)
AICC	$-2\ell + 2dn^*/(n^* - d - 1)$	Hurvich and Tsai (1989) Burnham and Anderson (1998)
BIC	$-2\ell + d \log n$ for $n > 0$	Schwarz (1978)

Here ℓ denotes the maximum value of the (possibly restricted) log likelihood; d is the dimension of the model; and n equals the number of effective subjects as displayed in the "Dimensions" table, unless this value equals 1, in which case n equals the number of levels of the first random effect specified in the first RANDOM statement or the number of levels of the interaction of the first random effect with noncommon subject effect specified in the first RANDOM statement. If the number of effective subjects equals 1 and you have no RANDOM statements, then n equals the number of valid observations for maximum likelihood estimation and $n - p$ for restricted maximum likelihood estimation, where p equals the rank of **X**. For AICC (a finite-sample corrected version of AIC), n^* equals the number of valid observations for maximum likelihood estimation and $n - p$ equals the number of valid observations for restricted maximum likelihood estimation, unless this number is less than $d + 2$, in which case it equals $d + 2$. When $n = 0$, the value of the BIC is -2ℓ. For restricted likelihood estimation, d equals q, the effective number of estimated covariance parameters. For maximum likelihood estimation, d equals $q + p$.

Timing Information

If you specify the DETAILS option in the PERFORMANCE statement, the procedure also produces a "Timing" table in which the elapsed time for each main task of the procedure is displayed.

ODS Table Names

Each table created by PROC HPLMIXED has a name associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 60.10.

Table 60.10 ODS Tables Produced by PROC HPLMIXED

Table Name	Description	Required Statement / Option
ClassLevels	Level information from the CLASS statement	Default output
ConvergenceStatus	Convergence status	Default output
CovParms	Estimated covariance parameters	Default output
Dimensions	Dimensions of the model	Default output
FitStatistics	Fit statistics	Default output
IterHistory	Iteration history	Default output
ModelInfo	Model information	Default output
NObs	Number of observations read and used	Default output
OptInfo	Optimization information	Default output
ParmSearch	Parameter search values	PARMS
PerformanceInfo	Information about high-performance computing environment	Default output
Ranks	Rank of designed matrix X	PROC HPLMIXED RANKS
SolutionF	Fixed-effects solution vector	MODEL / S
SolutionR	Random-effects solution vector	RANDOM / S
Timing	Timing breakdown by task	DETAILS option in the PERFORMANCE statement

Examples: HPLMIXED Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 60.1: Computing BLUPs for a Large Number of Subjects

(View the complete code for this example (hplmxex1.sas) in the example repository.)

Suppose you are using health measurements on patients treated by each medical center to monitor the performance of those centers. Different measurements within each patient are correlated, and there is enough data to fit the parameters of an unstructured covariance model for this correlation. In fact, long experience with historical data provides you with values for the covariance model that are essentially known, and the task is to apply these known values in order to compute best linear unbiased predictors (BLUPs) of the random effect of medical center. You can use these BLUPs to determine the best and worst performing medical centers, adjusting for other factors, on a weekly basis. Another reason why you want to do this with fixed values for the covariance parameters is to make the week-to-week BLUPs more comparable.

Although you cannot use the REPEATED statement in PROC HPLMIXED to fit models, you can use it to compute BLUPs for such models with known values of the variance parameters. For illustration, the following statements create a simulated data set of a given week's worth of patient health measurements across 100 different medical centers. Measurements at three different times are simulated for each patient, and each center has about 50 patients. The simulated model includes a fixed gender effect, a random effect due to center, and covariance between different measurements on the same patient.

```
%let NCenter  = 100;
%let NPatient = %eval(&NCenter*50);
%let NTime    = 3;
%let SigmaC   = 2.0;
%let SigmaP   = 4.0;
%let SigmaE   = 8.0;
%let Seed     = 12345;

data WeekSim;
   keep Gender Center Patient Time Measurement;
   array PGender{&NPatient};
   array PCenter{&NPatient};
   array PEffect{&NPatient};
   array CEffect{&NCenter};
   array GEffect{2};

   do Center = 1 to &NCenter;
      CEffect{Center} = sqrt(&SigmaC)*rannor(&Seed);
      end;

   GEffect{1} = 10*ranuni(&Seed);
   GEffect{2} = 10*ranuni(&Seed);

   do Patient = 1 to &NPatient;
      PGender{Patient} = 1 + int(2       *ranuni(&Seed));
      PCenter{Patient} = 1 + int(&NCenter*ranuni(&Seed));
      PEffect{Patient} = sqrt(&SigmaP)*rannor(&Seed);
      end;

   do Patient = 1 to &NPatient;
      Gender = PGender{Patient};
      Center = PCenter{Patient};
```

```
            Mean = 1 + GEffect{Gender} + CEffect{Center} + PEffect{Patient};
            do Time = 1 to &nTime;
               Measurement = Mean + sqrt(&SigmaE)*rannor(&Seed);
            output;
            end;
         end;
      run;
```

Suppose that the known values for the covariance parameters are

$$\begin{aligned} \text{Var(Center)} &= 1.7564 \\ \text{Cov(Patient)} &= \begin{bmatrix} 11.4555 & 3.6883 & 4.5951 \\ 3.6883 & 11.2071 & 3.6311 \\ 4.5951 & 3.6311 & 12.1050 \end{bmatrix} \end{aligned}$$

Incidentally, these are not precisely the same estimates you would get if you estimated these parameters based on the preceding data (for example, with the HPLMIXED procedure).

The following statements use PROC HPLMIXED to compute the BLUPs for the random medical center effect.

```
ods exclude all;
proc hplmixed data=WeekSim blup;
   class Gender Center Patient Time;
   model Measurement = Gender;
   random   Center / s;
   repeated Time   / sub=Patient type=un;
   parms   1.7564
          11.4555
           3.6883 11.2071
           4.5951  3.6311 12.1050;
   ods output SolutionR=BLUPs;
run;
ods exclude none;

proc sort data=BLUPs;
   by Estimate;
run;

data BLUPs; set BLUPs;
   Rank = _N_;
run;

proc print data=BLUPs;
   where ((Rank <= 5) | (Rank >= 96));
   var Center Estimate;
run;
```

Three parts of the PROC HPLMIXED syntax are required in order to compute BLUPs for this model: the BLUP option in the HPLMIXED statement, the REPEATED statement, and the PARMS statement with fixed values for all parameters. The resulting values of the best and worst performing medical centers for this week are shown in Output 60.1.1. Apparently, for this week's data, medical center 54 had the most decreasing effect, and medical center 48 the most increasing effect, on patient measurements overall.

Output 60.1.1 Highest and Lowest Medical Center BLUPs

Obs	Center	Estimate
1	54	-2.9369
2	7	-2.4614
3	50	-2.2467
4	51	-2.2281
5	93	-2.1644
96	26	2.1603
97	99	2.2718
98	44	2.4222
99	60	2.6089
100	48	2.6443

References

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Bates, D. M., Lindstrom, M. J., Wahba, G., and Yandell, B. S. (1987). "GCVPACK—Routines for Generalized Cross Validation." *Communications in Statistics—Simulation and Computation* 16:263–297.

Burdick, R. K., and Graybill, F. A. (1992). *Confidence Intervals on Variance Components*. New York: Marcel Dekker.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Carlin, B. P., and Louis, T. A. (1996). *Bayes and Empirical Bayes Methods for Data Analysis*. London: Chapman & Hall.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society, Series B* 39:1–38.

Fuller, W. A. (1976). *Introduction to Statistical Time Series*. New York: John Wiley & Sons.

Hartley, H. O., and Rao, J. N. K. (1967). "Maximum-Likelihood Estimation for the Mixed Analysis of Variance Model." *Biometrika* 54:93–108.

Harville, D. A. (1977). "Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems." *Journal of the American Statistical Association* 72:320–338.

Harville, D. A. (1988). "Mixed-Model Methodology: Theoretical Justifications and Future Directions." In *Proceedings of the Statistical Computing Section*, 41–49. Alexandria, VA: American Statistical Association.

Harville, D. A. (1990). "BLUP (Best Linear Unbiased Prediction), and Beyond." In *Advances in Statistical Methods for Genetic Improvement of Livestock*, edited by D. Gianola and K. Hammond, 239–276. Vol. 18 of Advanced Series in Agricultural Sciences. Berlin: Springer-Verlag.

Henderson, C. R. (1984). *Applications of Linear Models in Animal Breeding*. Guelph, ON: University of Guelph.

Henderson, C. R. (1990). "Statistical Method in Animal Improvement: Historical Overview." In *Advances in Statistical Methods for Genetic Improvement of Livestock*, 1–14. New York: Springer-Verlag.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Huynh, H., and Feldt, L. S. (1970). "Conditions Under Which Mean Square Ratios in Repeated Measurements Designs Have Exact F-Distributions." *Journal of the American Statistical Association* 65:1582–1589.

Jennrich, R. I., and Schluchter, M. D. (1986). "Unbalanced Repeated-Measures Models with Structured Covariance Matrices." *Biometrics* 42:805–820.

Kenward, M. G. (1987). "A Method for Comparing Profiles of Repeated Measurements." *Journal of the Royal Statistical Society, Series C* 36:296–308.

Laird, N. M., Lange, N. T., and Stram, D. O. (1987). "Maximum Likelihood Computations with Repeated Measures: Application of the EM Algorithm." *Journal of the American Statistical Association* 82:97–105.

Laird, N. M., and Ware, J. H. (1982). "Random-Effects Models for Longitudinal Data." *Biometrics* 38:963–974.

Lindstrom, M. J., and Bates, D. M. (1988). "Newton-Raphson and EM Algorithms for Linear Mixed-Effects Models for Repeated-Measures Data." *Journal of the American Statistical Association* 83:1014–1022.

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and Schabenberger, O. (2006). *SAS for Mixed Models*. 2nd ed. Cary, NC: SAS Institute Inc.

Little, R. J. A. (1995). "Modeling the Drop-Out Mechanism in Repeated-Measures Studies." *Journal of the American Statistical Association* 90:1112–1121.

Macchiavelli, R. E., and Arnold, S. F. (1994). "Variable Order Ante-dependence Models." *Communications in Statistics—Theory and Methods* 23:2683–2699.

McLean, R. A., and Sanders, W. L. (1988). "Approximating Degrees of Freedom for Standard Errors in Mixed Linear Models." In *Proceedings of the Statistical Computing Section*, 50–59. Alexandria, VA: American Statistical Association.

McLean, R. A., Sanders, W. L., and Stroup, W. W. (1991). "A Unified Approach to Mixed Linear Models." *American Statistician* 45:54–64.

Milliken, G. A., and Johnson, D. E. (1992). *Designed Experiments.* Vol. 1 of Analysis of Messy Data. Reprint edition. New York: Chapman & Hall.

Patel, H. I. (1991). "Analysis of Incomplete Data from a Clinical Trial with Repeated Measurements." *Biometrika* 78:609–619.

Patterson, H. D., and Thompson, R. (1971). "Recovery of Inter-block Information When Block Sizes Are Unequal." *Biometrika* 58:545–554.

Robinson, G. K. (1991). "That BLUP Is a Good Thing: The Estimation of Random Effects." *Statistical Science* 6:15–51.

Rubin, D. B. (1976). "Inference and Missing Data." *Biometrika* 63:581–592.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Searle, S. R. (1982). *Matrix Algebra Useful for Statisticians*. New York: John Wiley & Sons.

Searle, S. R., Casella, G., and McCulloch, C. E. (1992). *Variance Components*. New York: John Wiley & Sons.

Serfling, R. J. (1980). *Approximation Theorems of Mathematical Statistics*. New York: John Wiley & Sons.

Snedecor, G. W., and Cochran, W. G. (1980). *Statistical Methods*. 7th ed. Ames: Iowa State University Press.

Steel, R. G. D., Torrie, J. H., and Dickey, D. A. (1997). *Principles and Procedures of Statistics: A Biometrical Approach*. 3rd ed. New York: McGraw-Hill.

Verbeke, G., and Molenberghs, G., eds. (1997). *Linear Mixed Models in Practice: A SAS-Oriented Approach*. New York: Springer.

Verbeke, G., and Molenberghs, G. (2000). *Linear Mixed Models for Longitudinal Data*. New York: Springer.

Winer, B. J. (1971). *Statistical Principles in Experimental Design*. 2nd ed. New York: McGraw-Hill.

Wolfinger, R. D. (1996). "Heterogeneous Variance-Covariance Structures for Repeated Measures." *Journal of Agricultural, Biological, and Environmental Statistics* 1:205–230.

Wolfinger, R. D., Tobias, R. D., and Sall, J. (1994). "Computing Gaussian Likelihoods and Their Derivatives for General Linear Mixed Models." *SIAM Journal on Scientific Computing* 15:1294–1310.

Chapter 61
The HPLOGISTIC Procedure

Contents

Overview: HPLOGISTIC Procedure **4844**
- PROC HPLOGISTIC Features 4844
- PROC HPLOGISTIC Contrasted with Other SAS Procedures 4845

Getting Started: HPLOGISTIC Procedure **4845**
- Binary Logistic Regression 4845

Syntax: HPLOGISTIC Procedure **4851**
- PROC HPLOGISTIC Statement 4852
- BY Statement 4857
- CLASS Statement 4858
- CODE Statement 4858
- FREQ Statement 4859
- ID Statement 4859
- MODEL Statement 4859
- OUTPUT Statement 4866
- PARTITION Statement 4868
- PERFORMANCE Statement 4868
- SELECTION Statement 4869
- WEIGHT Statement 4870

Details: HPLOGISTIC Procedure **4870**
- Missing Values 4870
- Response Distributions 4871
- Log-Likelihood Functions 4872
- Existence of Maximum Likelihood Estimates 4873
- Using Validation and Test Data 4874
- Model Fit and Assessment Statistics 4876
- The Hosmer-Lemeshow Goodness-of-Fit Test 4880
- Computational Method: Multithreading 4881
- Choosing an Optimization Algorithm 4881
  - First- or Second-Order Algorithms 4881
  - Algorithm Descriptions 4883
- Displayed Output 4885
- ODS Table Names 4889

Examples: HPLOGISTIC Procedure **4891**
- Example 61.1: Model Selection 4891
- Example 61.2: Modeling Binomial Data 4894
- Example 61.3: Ordinal Logistic Regression 4901

Example 61.4: Partitioning Data 4904
References **4906**

Overview: HPLOGISTIC Procedure

The HPLOGISTIC procedure is a high-performance statistical procedure that fits logistic regression models for binary, binomial, and multinomial data on the SAS appliance.

The HPLOGISTIC procedure fits logistic regression models in the broader sense; the procedure permits several link functions and can handle ordinal and nominal data with more than two response categories (multinomial data).

PROC HPLOGISTIC Features

The HPLOGISTIC procedure estimates the parameters of a logistic regression model by using maximum likelihood techniques. It also does the following:

- provides model-building syntax with the CLASS and effect-based MODEL statements, which are familiar from SAS/STAT analytic procedures (in particular, the GLM, LOGISTIC, GLIMMIX, and MIXED procedures)
- provides response-variable options as in the LOGISTIC procedure
- performs maximum likelihood estimation
- provides multiple link functions
- provides cumulative link models for ordinal data and generalized logit modeling for unordered multinomial data
- enables model building (variable selection) through the SELECTION statement
- provides a WEIGHT statement for weighted analysis
- provides a FREQ statement for grouped analysis
- provides an OUTPUT statement to produce a data set that contains predicted probabilities and other observationwise statistics

Because the HPLOGISTIC procedure is a high-performance statistical procedure, it also exploits all the available cores and concurrent threads.

PROC HPLOGISTIC Contrasted with Other SAS Procedures

For general contrasts, see the section "Common Features of SAS High-Performance Statistical Procedures" on page 572. The following remarks contrast the HPLOGISTIC procedure with the LOGISTIC procedure in SAS/STAT software.

The CLASS statement in the HPLOGISTIC procedure permits two parameterizations: the GLM parameterization and a reference parameterization. In contrast to the LOGISTIC, GENMOD, and other procedures that permit multiple parameterizations, the HPLOGISTIC procedure does not mix parameterizations across the variables in the CLASS statement. In other words, all classification variables have the same parameterization, and this parameterization is either the GLM or reference parameterization.

The default parameterization of CLASS variables in the HPLOGISTIC procedure is the GLM parameterization. The LOGISTIC procedure uses the EFFECT parameterization for the CLASS variables by default. In either procedure, you can change the parameterization with the PARAM= option in the CLASS statement.

The default optimization technique used by the LOGISTIC procedure is Fisher scoring; the HPLOGISTIC procedure uses by default a modification of the Newton-Raphson algorithm with a ridged Hessian. You can choose different optimization techniques, including first-order methods that do not require a crossproducts matrix or Hessian, with the TECHNIQUE= option in the PROC HPLOGISTIC statement.

The LOGISTIC procedure offers a wide variety of postfitting analyses, such as contrasts, estimates, tests of model effects, least squares means, and odds ratios. The HPLOGISTIC procedure is limited in postfitting functionality, since with large data sets the focus is primarily on model fitting and scoring.

By default, PROC HPLOGISTIC performs computations in multiple threads. The LOGISTIC procedure executes in a single thread.

Getting Started: HPLOGISTIC Procedure

Binary Logistic Regression

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hploggs1.sas.)

The following DATA step contains 100 observations on a dichotomous response variable (y), a character variable (C), and 10 continuous variables (x1–x10):

```
data getStarted;
   input C$ y x1-x10;
   datalines;
   D  0  10.2  6  1.6  38  15  2.4  20  0.8  8.5  3.9
   F  1  12.2  6  2.6  42  61  1.5  10  0.6  8.5  0.7
   D  1   7.7  1  2.1  38  61    1  90  0.6  7.5  5.2
   J  1  10.9  7  3.5  46  42  0.3   0  0.2    6  3.6
   E  0  17.3  6  3.8  26  47  0.9  10  0.4  1.5  4.7
   A  0  18.7  4  1.8   2  34  1.7  80    1  9.5  2.2
   B  0   7.2  1  0.3  48  61  1.1  10  0.8  3.5    4
```

D	0	0.1	3	2.4	0	65	1.6	70	0.8	3.5	0.7
H	1	2.4	4	0.7	38	22	0.2	20	0	3	4.2
J	0	15.6	7	1.4	0	98	0.3	0	1	5	5.2
J	0	11.1	3	2.4	42	55	2.2	60	0.6	4.5	0.7
F	0	4	6	0.9	4	36	2.1	30	0.8	9	4.6
A	0	6.2	2	1.8	14	79	1.1	70	0.2	0	5.1
H	0	3.7	3	0.8	12	66	1.3	40	0.4	0.5	3.3
A	1	9.2	3	2.3	48	51	2.3	50	0	6	5.4
G	0	14	3	2	18	12	2.2	0	0	3	3.4
E	1	19.5	6	3.7	26	81	0.1	30	0.6	5	4.8
C	0	11	3	2.8	38	9	1.7	50	0.8	6.5	0.9
I	0	15.3	7	2.2	20	98	2.7	100	0.4	7	0.8
H	1	7.4	4	0.5	28	65	1.3	60	0.2	9.5	5.4
F	0	11.4	2	1.4	42	12	2.4	10	0.4	1	4.5
C	1	19.4	1	0.4	42	4	2.4	10	0	6.5	0.1
G	0	5.9	4	2.6	12	57	0.8	50	0.4	2	5.8
G	1	15.8	6	3.7	34	8	1.3	90	0.6	2.5	5.7
I	0	10	3	1.9	16	80	3	90	0.4	9.5	1.9
E	0	15.7	1	2.7	32	25	1.7	20	0.2	8.5	6
G	0	11	5	2.9	48	53	0.1	50	1	3.5	1.2
J	1	16.8	0	0.9	14	86	1.4	40	0.8	9	5
D	1	11	4	3.2	48	63	2.8	90	0.6	0	2.2
J	1	4.8	7	3.6	24	1	2.2	20	1	8.5	0.5
J	1	10.4	5	2	42	56	1	20	0	3.5	4.2
G	0	12.7	7	3.6	8	56	2.1	70	1	4.5	1.5
G	0	6.8	1	3.2	30	27	0.6	0	0.8	2	5.6
E	0	8.8	0	3.2	2	67	0.7	10	0.4	1	5
I	1	0.2	0	2.9	10	41	2.3	60	0.2	9	0.3
J	1	4.6	7	3.9	50	61	2.1	50	0.4	3	4.9
J	1	2.3	2	3.2	36	98	0.1	40	0.6	4.5	4.3
I	0	10.8	3	2.7	28	58	0.8	80	0.8	3	6
B	0	9.3	2	3.3	44	44	0.3	50	0.8	5.5	0.4
F	0	9.2	6	0.6	4	64	0.1	0	0.6	4.5	3.9
D	0	7.4	0	2.9	14	0	0.2	30	0.8	7.5	4.5
G	0	18.3	3	3.1	8	60	0.3	60	0.2	7	1.9
F	0	5.3	4	0.2	48	63	2.3	80	0.2	8	5.2
C	0	2.6	5	2.2	24	4	1.3	20	0	2	1.4
F	0	13.8	4	3.6	4	7	1.1	10	0.4	3.5	1.9
B	1	12.4	6	1.7	30	44	1.1	60	0.2	6	1.5
I	0	1.3	1	1.3	8	53	1.1	70	0.6	7	0.8
F	0	18.2	7	1.7	26	92	2.2	30	1	8.5	4.8
J	0	5.2	2	2.2	18	12	1.4	90	0.8	4	4.9
G	1	9.4	2	0.8	22	86	0.4	30	0.4	1	5.9
J	1	10.4	2	1.7	26	31	2.4	10	0.2	7	1.6
J	0	13	1	1.8	14	11	2.3	50	0.6	5.5	2.6
A	0	17.9	4	3.1	46	58	2.6	90	0.6	1.5	3.2
D	1	19.4	6	3	20	50	2.8	100	0.2	9	1.2
I	0	19.6	3	3.6	22	19	1.2	0	0.6	5	4.1
I	1	6	2	1.5	30	30	2.2	20	0.4	8.5	5.3
G	0	13.8	1	2.7	0	52	2.4	20	0.8	6	2
B	0	14.3	4	2.9	30	11	0.6	90	0.6	0.5	4.9
E	0	15.6	0	0.4	38	79	0.4	80	0.4	1	3.3
D	0	14	2	1	22	61	3	90	0.6	2	0.1
C	1	9.4	5	0.4	12	53	1.7	40	0	3	1.1

```
   H  0  13.2  1  1.6  40  15  0.7  40  0.2    9  5.5
   A  0  13.5  5  2.4  18  89  1.6  20  0.4  9.5  4.7
   E  0   2.6  4  2.3  38   6  0.8  20  0.4    5  5.3
   E  0  12.4  3  1.3  26   8  2.8  10  0.8    6  5.8
   D  0   7.6  2  0.9  44  89  1.3  50  0.8    6  0.4
   I  0  12.7  1  2.3  42   6  2.4  10  0.4    1    3
   C  1  10.7  4  3.2  28  23  2.2  90  0.8  5.5  2.8
   H  0  10.1  2  2.3  10  62  0.9  50  0.4  2.5  3.7
   C  1  16.6  1  0.5  12  88  0.1  20  0.6  5.5  1.8
   I  1   0.2  3  2.2   8  71  1.7  80  0.4  0.5  5.5
   C  0  10.8  4  3.5  30  70  2.3  60  0.4  4.5  5.9
   F  0   7.1  4    3  14  63  2.4  70    0    7  3.1
   D  0  16.5  1  3.3  30  80  1.6  40    0  3.5  2.7
   H  0  17.1  7  2.1  30  45  1.5  60  0.6  0.5  2.8
   D  0   4.3  1  1.5  24  44    0  70    0    5  0.5
   H  0    15  2  0.2  14  87  1.8  50    0  4.5  4.7
   G  0  19.7  3  1.9  36  99  1.5  10  0.6    3  1.7
   H  1   2.8  6  0.6  34  21    2  60    1    9  4.7
   G  0  16.6  3  3.3  46   1  1.4  70  0.6  1.5  5.3
   E  0  11.7  5  2.7  48   4  0.9  60  0.8  4.5  1.6
   F  0  15.6  3  0.2   4  79  0.5   0  0.8  1.5  2.9
   C  1   5.3  6  1.4   8  64    2  80  0.4    9  4.2
   B  1   8.1  7  1.7  40  36  1.4  60  0.6    6  3.9
   I  0  14.8  2  3.2   8  37  0.4  10    0  4.5    3
   D  0   7.4  4    3  12   3  0.6  60  0.6    7  0.7
   D  0   4.8  3  2.3  44  41  1.9  60  0.2    3  3.1
   A  0   4.5  0  0.2   4  48  1.7  80  0.8    9  4.2
   D  0   6.9  6  3.3  14  92  0.5  40  0.4  7.5    5
   B  0   4.7  4  0.9  14  99  2.4  80    1  0.5  0.7
   I  1   7.5  4  2.1  20  79  0.4  40  0.4  2.5  0.7
   C  0   6.1  0  1.4  38  18  2.3  60  0.8  4.5  0.7
   C  0  18.3  1    1  26  98  2.7  20    1  8.5  0.5
   F  0  16.4  7  1.2  32  94  2.9  40  0.4  5.5  2.1
   I  0   9.4  2  2.3  32  42  0.2  70  0.4  8.5  0.3
   F  1  17.9  4  1.3  32  42    2  40  0.2    1  5.4
   H  0  14.9  3  1.6  36  74  2.6  60  0.2    1  2.3
   C  0  12.7  0  2.6   0  88  1.1  80  0.8  0.5  2.1
   F  0   5.4  4  1.5   2   1  1.8  70  0.4  5.5  3.6
   J  1  12.1  4  1.8  20  59  1.3  60  0.4    3  3.8
;
```

The following statements fit a logistic model to these data by using a classification effect for variable C and 10 regressor effects for x1–x10:

```
proc hplogistic data=getStarted;
   class C;
   model y = C x1-x10;
run;
```

The default output from this analysis is presented in Figure 61.1 through Figure 61.11.

The "Performance Information" table in Figure 61.1 shows that the procedure executes in single-machine mode—that is, the model is fit on the machine where the SAS session executes. This run of the HPLOGISTIC procedure was performed on a multicore machine with the same number of CPUs as there are threads; that is, one computational thread was spawned per CPU.

Figure 61.1 Performance Information

The HPLOGISTIC Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Figure 61.2 displays the "Model Information" table. The HPLOGISTIC procedure uses a Newton-Raphson algorithm to model a binary distribution for the variable y with a logit link function. The CLASS variable C is parameterized using the GLM parameterization, which is the default.

Figure 61.2 Model Information

Model Information	
Data Source	WORK.GETSTARTED
Response Variable	y
Class Parameterization	GLM
Distribution	Binary
Link Function	Logit
Optimization Technique	Newton-Raphson with Ridging

The CLASS variable C has 10 unique formatted levels, and these are displayed in the "Class Level Information" table in Figure 61.3.

Figure 61.3 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
C	10	A B C D E F G H I J

Figure 61.4 displays the "Number of Observations" table. All 100 observations in the data set are used in the analysis.

Figure 61.4 Number of Observations

Number of Observations Read	100
Number of Observations Used	100

The "Response Profile" table in Figure 61.5 is produced by default for binary and multinomial response variables. It shows the breakdown of the response variable levels by frequency. By default for binary data, the HPLOGISTIC procedure models the probability of the event with the lower-ordered value in the "Response Profile" table—this is indicated by the note that follows the table. In this example, the values represented by y = '0' are modeled as the "successes" in the Bernoulli experiments.

Figure 61.5 Response Profile

Response Profile		
Ordered Value	y	Total Frequency
1	0	69
2	1	31

You are modeling the probability that y='0'.

You can use the response-variable options in the MODEL statement to affect which value of the response variable is modeled.

Figure 61.6 displays the "Dimensions" table for this model. This table summarizes some important sizes of various model components. For example, it shows that there are 21 columns in the design matrix **X**, which correspond to one column for the intercept, 10 columns for the effect associated with the classification variable C, and one column each for the continuous variables x1–x10. However, the rank of the crossproducts matrix is only 20. Because the classification variable C uses GLM parameterization and because the model contains an intercept, there is one singularity in the crossproducts matrix of the model. Consequently, only 20 parameters enter the optimization.

Figure 61.6 Dimensions in Binomial Logistic Regression

Dimensions	
Columns in X	21
Number of Effects	12
Max Effect Columns	10
Rank of Cross-product Matrix	20
Parameters in Optimization	20

The "Iteration History" table is shown in Figure 61.7. The Newton-Raphson algorithm with ridging converged after four iterations, not counting the initial setup iteration.

Figure 61.7 Iteration History

Iteration History				
Iteration	Evaluations	Objective Function	Change	Max Gradient
0	4	0.4493546916	.	0.410972
1	2	0.4436453992	0.00570929	0.081339
2	2	0.4435038109	0.00014159	0.003302
3	2	0.4435035933	0.00000022	5.623E-6
4	2	0.4435035933	0.00000000	1.59E-11

Figure 61.8 displays the final convergence status of the Newton-Raphson algorithm. The GCONV= relative convergence criterion is satisfied.

Figure 61.8 Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

The "Fit Statistics" table is shown in Figure 61.9. The –2 log likelihood at the converged estimates is 88.7007. You can use this value to compare the model to nested model alternatives by means of a likelihood-ratio test. To compare models that are not nested, information criteria such as AIC (Akaike's information criterion), AICC (Akaike's bias-corrected information criterion), and BIC (Schwarz' Bayesian information criterion) are used. These criteria penalize the –2 log likelihood for the number of parameters. Because of the large number of parameters relative to the number of observations, the discrepancy between the –2 log likelihood and, say, AIC, is substantial in this case.

Figure 61.9 Fit Statistics

Fit Statistics	
-2 Log Likelihood	88.7007
AIC (smaller is better)	128.70
AICC (smaller is better)	139.33
BIC (smaller is better)	180.80

Figure 61.10 shows the global test for the null hypothesis that all model effects jointly do not affect the probability of success of the binary response. The test is significant (p-value = 0.0135). One or more of the model effects thus significantly affects the probability of observing an event.

Figure 61.10 Null Test

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	35.1194	19	0.0135

However, a look at the "Parameter Estimates" table in Figure 61.11 shows that many parameters have fairly large p-values, indicating that one or more of the model effects might not be necessary.

Figure 61.11 Parameter Estimates

Parameter Estimates					
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	1.2101	1.7507	Infty	0.69	0.4894
C A	3.4341	1.6131	Infty	2.13	0.0333
C B	2.1638	1.4271	Infty	1.52	0.1295
C C	0.6552	1.0810	Infty	0.61	0.5445
C D	2.4945	1.1094	Infty	2.25	0.0245
C E	3.2449	1.4321	Infty	2.27	0.0235
C F	3.6054	1.3070	Infty	2.76	0.0058
C G	2.0841	1.1898	Infty	1.75	0.0798
C H	2.9368	1.2939	Infty	2.27	0.0232
C I	1.3785	1.0319	Infty	1.34	0.1816
C J	0	.	.	.	.
x1	0.03218	0.05710	Infty	0.56	0.5730
x2	-0.3677	0.1538	Infty	-2.39	0.0168
x3	0.3146	0.3574	Infty	0.88	0.3787
x4	-0.05196	0.02443	Infty	-2.13	0.0334
x5	-0.00683	0.01056	Infty	-0.65	0.5177
x6	0.2539	0.3785	Infty	0.67	0.5024
x7	-0.00723	0.01073	Infty	-0.67	0.5004
x8	2.5370	0.9942	Infty	2.55	0.0107
x9	-0.1675	0.1068	Infty	-1.57	0.1168
x10	-0.2222	0.1577	Infty	-1.41	0.1590

Syntax: HPLOGISTIC Procedure

The following statements are available in the HPLOGISTIC procedure:

PROC HPLOGISTIC < *options* > ;
- **BY** *variables* ;
- **CLASS** *variable* < (*options*) >. . . < *variable* < (*options*) > > < / *global-options* > ;
- **CODE** < *options* > ;
- **FREQ** *variable* ;
- **ID** *variables* ;
- **MODEL** *response*< (*response-options*) > = < *effects* > < / *model-options* > ;
- **MODEL** *events/trials*< (*response-options*) > = < *effects* > < / *model-options* > ;
- **OUTPUT** < **OUT=***SAS-data-set* > < *keyword* < =*name* > >. . . < *keyword* < =*name* > > < / *options* > ;
- **PARTITION** *partition-options* ;
- **PERFORMANCE** *performance-options* ;
- **SELECTION** *selection-options* ;
- **WEIGHT** *variable* ;

The PROC HPLOGISTIC statement and at least one MODEL statement is required. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statements.

PROC HPLOGISTIC Statement

PROC HPLOGISTIC < *options* > **;**

The PROC HPLOGISTIC statement invokes the procedure. Table 61.1 summarizes the available options in the PROC HPLOGISTIC statement by function. The options are then described fully in alphabetical order.

Table 61.1 PROC HPLOGISTIC Statement Options

Option	Description
Basic Options	
ALPHA=	Specifies a global significance level
DATA=	Specifies the input data set
NAMELEN=	Limits the length of effect names
Options Related to Output	
ITDETAILS	Adds detail information to "Iteration History" table
ITSELECT	Displays the "Iteration History" table with model selection
NOPRINT	Suppresses ODS output
NOCLPRINT	Limits or suppresses the display of class levels
NOITPRINT	Suppresses generation of the iteration history table
NOSTDERR	Suppresses computation of covariance matrix and standard errors
Options Related to Optimization	
ABSCONV=	Tunes the absolute function convergence criterion
ABSFCONV=	Tunes the absolute function difference convergence criterion
ABSGCONV=	Tunes the absolute gradient convergence criterion
FCONV=	Tunes the relative function difference convergence criterion
GCONV=	Tunes the relative gradient convergence criterion
INEST=	Specifies the SAS data set that contains the starting values
MAXITER=	Chooses the maximum number of iterations in any optimization
MAXFUNC=	Specifies the maximum number of function evaluations in any optimization
MAXTIME=	Specifies the upper limit of CPU time (in seconds) for any optimization
MINITER=	Specifies the minimum number of iterations in any optimization
NORMALIZE=	Specifies whether the objective function is normalized during optimization
OUTEST	Adds parameter name to the "Parameter Estimates" table
TECHNIQUE=	Selects the optimization technique
Tolerances	
SINGCHOL=	Tunes the singularity criterion for Cholesky decompositions
SINGSWEEP=	Tunes the singularity criterion for the sweep operator
SINGULAR=	Tunes the general singularity criterion
User-Defined Formats	
FMTLIBXML=	Specifies the file reference for a format stream

You can specify the following options in the PROC HPLOGISTIC statement.

ABSCONV=*r*
ABSTOL=*r*

specifies an absolute function convergence criterion. For minimization, termination requires $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of *r* is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

ABSFCONV=*r*
ABSFTOL=*r*

specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, termination requires a small change of the function value in successive iterations:

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}(k)$ is defined as the vertex with the lowest function value and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default value is $r = 0$.

ABSGCONV=*r*
ABSGTOL=*r*

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small:

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $g_j(\cdot)$ is the gradient of the objective function with respect to the *j*th parameter. This criterion is not used by the NMSIMP technique. The default value is *r*=1E–5.

ALPHA=*number*

specifies a global significance level for the construction of confidence intervals. The confidence level is 1–*number*. The value of *number* must be between 0 and 1; the default is 0.05. You can override the global specification with the ALPHA= option in the MODEL statement.

DATA=*SAS-data-set*

names the input SAS data set for PROC HPLOGISTIC to use. The default is the most recently created data set.

FCONV=*r*
FTOL=*r*

specifies a relative function difference convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex with the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default value is r=2 $\times \epsilon$ where ϵ is the machine precision.

FMTLIBXML=*file-ref*

specifies the file reference for the XML stream that contains the user-defined format definitions. For more information about how to generate a XML stream for your formats, see the section "Working with Formats" on page 568.

GCONV=*r*

GTOL=*r*

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, termination requires that the normalized predicted function reduction be small,

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the CONGRA technique (where a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \; |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The default value is *r*=1E–8.

INEST=*SAS-data-set*

names the TYPE=EST SAS data set that contains starting values for the parameters.

Your data set must include the _TYPE_ variable, a character variable in which the value 'PARMS' indicates the observation that contains your starting values. The data set also includes a numeric variable for each parameter for which you are specifying a starting value; the name of this numeric variable is the "parameter name." You can obtain parameter names by specifying the OUTEST option and by using the ODS OUTPUT statement to output the "Parameter Estimates" table into a data set; the parameter name is contained in the ParmName variable in this data set. If you do not specify a starting value for a parameter, it is set to 0. PROC HPLOGISTIC uses only the first observation for which _TYPE_=PARMS, and it ignores BY variables. For more information about TYPE=EST data sets, see Chapter A, "Special SAS Data Sets."

If you specify TECH=NONE or MAXITER=0, then the values in the INEST= data set are used as the parameter estimates, but the null model is still computed at the optimum value for the intercepts. If you specify TECH=NONE or MAXITER=0 and you specify a null model in the MODEL statement, then the null model is computed at the starting values for the intercept parameters.

ITDETAILS

adds to the "Iteration History" table the current values of the parameter estimates and their gradients. These quantities are reported only for parameters that participate in the optimization.

ITSELECT

generates the "Iteration History" table when you perform a model selection.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number *n* of function calls in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, NEWRAP: 125
- QUANEW, DBLDOG: 500
- CONGRA: 1,000
- NMSIMP: 3,000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that is specified by the MAXFUNC= option. You can choose the optimization technique with the TECHNIQUE= option.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number *n* of iterations in the optimization process. The default values are as follows, depending on the optimization technique:

- TRUREG, NRRIDG, NEWRAP: 50
- QUANEW, DBLDOG: 200
- CONGRA: 400
- NMSIMP: 1,000

These default values also apply when *n* is specified as a missing value. You can choose the optimization technique with the TECHNIQUE= option.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The default value is the largest floating-point double representation of your computer. The time specified by the MAXTIME= option is checked only once at the end of each iteration. Therefore, the actual running time can be longer than that specified by the MAXTIME= option.

MINITER=*n*

MINIT=*n*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

NAMELEN=*number*
specifies the length to which long effect names are shortened. The default and minimum value is 20.

NOCLPRINT<=*number***>**
suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying a *number* helps to reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOITPRINT
suppresses the generation of the "Iteration History" table.

NOPRINT
suppresses the generation of ODS output.

NORMALIZE=YES | NO
specifies whether the objective function should be normalized during the optimization by the reciprocal of the used frequency count. The default is to normalize the objective function. This option affects the values reported in the "Iteration History" table. The results reported in the "Fit Statistics" are always displayed for the nonnormalized log-likelihood function.

NOSTDERR
suppresses the computation of the covariance matrix and the standard errors of the logistic regression coefficients. When the model contains many variables (thousands), the inversion of the Hessian matrix to derive the covariance matrix and the standard errors of the regression coefficients can be time-consuming.

OUTEST
adds a column for the ParmName variable to the "Parameter Estimates" table. This column is not displayed, but you can use it to create a data set that you can specify in an INEST= option by first using the ODS OUTPUT statement to output the "Parameter Estimates" table and then submitting the following statements:

```
proc transpose data=parameterestimates out=inest(type=EST) label=_TYPE_;
   label Estimate=PARMS;
   var Estimate;
   id ParmName;
run;
```

SINGCHOL=*number*
tunes the singularity criterion in Cholesky decompositions. The default is 1E7 times the machine epsilon; this product is approximately 1E–9 on most computers.

SINGSWEEP=*number*
tunes the singularity criterion for sweep operations. The default is 1E7 times the machine epsilon; this product is approximately 1E–9 on most computers.

SINGULAR=*number*

tunes the general singularity criterion applied by the HPLOGISTIC procedure in sweeps and inversions. The default is 1E7 times the machine epsilon; this product is approximately 1E–9 on most computers.

TECHNIQUE=*keyword*

TECH=*keyword*

specifies the optimization technique for obtaining maximum likelihood estimates. You can choose from the following techniques by specifying the appropriate *keyword*:

CONGRA	performs a conjugate-gradient optimization.
DBLDOG	performs a version of double-dogleg optimization.
NEWRAP	performs a Newton-Raphson optimization with line search.
NMSIMP	performs a Nelder-Mead simplex optimization.
NONE	performs no optimization.
NRRIDG	performs a Newton-Raphson optimization with ridging.
QUANEW	performs a dual quasi-Newton optimization.
TRUREG	performs a trust-region optimization

The default value is TECHNIQUE=NRRIDG.

For more information, see the section "Choosing an Optimization Algorithm" on page 4881.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPLOGISTIC to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPLOGISTIC procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) >. . . < *variable* < (*options*) > > < / *global-options* > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. The CLASS statement must precede the MODEL statement. You can list the response variable for binary and multinomial models in the CLASS statement, but this is not necessary.

The CLASS statement for high-performance statistical procedures is documented in the section "CLASS Statement" on page 572.

The HPLOGISTIC procedure does not support the SPLIT option in the CLASS statement. The HPLOGISTIC procedure additionally supports the following global-option in the CLASS statement:

UPCASE

uppercases the values of character-valued CLASS variables before levelizing them. For example, if the UPCASE option is in effect and a CLASS variable can take the values 'a', 'A', and 'b', then 'a' and 'A' represent the same level and the CLASS variable is treated as having only two values: 'A' and 'B'.

CODE Statement

CODE < *options* > ;

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 61.2 summarizes the *options* available in the CODE statement.

Table 61.2 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence for each observation. High-performance statistical procedures that support the FREQ statement treat each observation as if it appeared f times, where the frequency value f is the value of the FREQ variable for the observation. If f is not an integer, then f is truncated to an integer. If f is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are to be transferred to output data sets created by high-performance statistical procedures, provided that the output data set produces one (or more) records per input observation.

For documentation about the common ID statement in high-performance statistical procedures, see the section "ID Statement" on page 576.

MODEL Statement

MODEL *response* < **(***response-options***)** > **=** < *effects* > < / *model-options* > **;**

MODEL *events* / *trials* < **(***response-options***)** > **=** < *effects* > < / *model-options* > **;**

The MODEL statement defines the statistical model in terms of a response variable (the target) or an *events/trials* specification, model effects constructed from variables in the input data set, and options. An intercept is included in the model by default. You can remove the intercept with the NOINT option.

You can specify a single *response* variable that contains your binary, ordinal, or nominal response values. When you have binomial data, you can specify the *events/trials* form of the response, where one variable contains the number of positive responses (or events) and another variable contains the number of trials. Note that the values of both *events* and (*trials* – *events*) must be nonnegative and the value of *trials* must be positive.

For information about constructing the model effects, see the section "Parameterization of Model Effects" on page 412.

There are two sets of options in the MODEL statement. The *response-options* determine how the HPLOGISTIC procedure models probabilities for binary data. The *model-options* control other aspects of model formation and inference. Table 61.3 summarizes these options.

Table 61.3 MODEL Statement Options

Option	Description
Response Variable Options	
DESCENDING	Reverses the response categories
EVENT=	Specifies the event category
ORDER=	Specifies the sort order
REF=	Specifies the reference category
Model Options	
ALPHA=	Specifies the confidence level for confidence limits
ASSOCIATION	Requests association statistics
CL	Requests confidence limits
CTABLE	Requests classification statistics
CUTPOINT=	Specifies a cutpoint for binary classification
DDFM=	Specifies the degrees-of-freedom method
INCLUDE=	Includes effects in all models for model selection
LACKFIT	Requests the Hosmer and Lemeshow goodness-of-fit test
LINK=	Specifies the link function
NOCHECK	Suppresses checking for infinite parameters
NOINT	Suppresses the intercept
OFFSET=	Specifies the offset variable
PRIOR=	Specifies prior probabilities
RSQUARE	Requests a generalized coefficient of determination
START=	Includes effects in the initial model for model selection
STB	Displays standardized estimates

Response Variable Options

Response variable options determine how the HPLOGISTIC procedure models probabilities for binary and multinomial data.

You can specify the following *response-options* by enclosing them in parentheses after the *response* or *trials* variable.

DESCENDING

DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC HPLOGISTIC orders the response categories according to the ORDER= option and then reverses that order.

EVENT='*category*' | FIRST | LAST

specifies the event category for the binary response model. PROC HPLOGISTIC models the probability of the event category. The EVENT= option has no effect when there are more than two response categories.

You can specify the value (formatted, if a format is applied) of the event category in quotes, or you can specify one of the following:

FIRST
designates the first ordered category as the event. This is the default.

LAST
designates the last ordered category as the event.

For example, the following statements specify that observations with formatted value '1' represent events in the data. The probability modeled by the HPLOGISTIC procedure is thus the probability that the variable def takes on the (formatted) value '1'.

```
proc hplogistic data=MyData;
   class A B C;
   model def(event ='1') = A B C x1 x2 x3;
run;
```

ORDER=DATA | FORMATTED | INTERNAL

ORDER=FREQ | FREQDATA | FREQFORMATTED | FREQINTERNAL

specifies the sort order for the levels of the *response* variable. When ORDER=FORMATTED (the default) for numeric variables for which you have supplied no explicit format (that is, for which there is no corresponding FORMAT statement in the current PROC HPLOGISTIC run or in the DATA step that created the data set), the levels are ordered by their internal (numeric) value. Table 61.4 shows the interpretation of the ORDER= option.

Table 61.4 Sort Order

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count (levels with the most observations come first in the order)
FREQDATA	Order of descending frequency count; by order of appearance in the input data set when counts are tied
FREQFORMATTED	Order of descending frequency count; by formatted value (as above) when counts are tied
FREQINTERNAL	Order of descending frequency count; by unformatted value when counts are tied
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For the FORMATTED and INTERNAL orders, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Language Reference: Concepts*.

REF='*category*' | **FIRST** | **LAST**

specifies the reference category for the generalized logit model and the binary response model. For the generalized logit model, each logit contrasts a nonreference category with the reference category. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotes, or you can specify one of the following:

FIRST

designates the first ordered category as the reference

LAST

designates the last ordered category as the reference. This is the default.

Model Options

ALPHA=*number*

requests that confidence intervals for each of the parameters be constructed with the confidence level 1–*number*. The value of *number* must be between 0 and 1. By default, *number* is equal to the value of the ALPHA= option in the PROC HPLOGISTIC statement, or 0.05 if you do not specify that option.

ASSOCIATION

displays measures of association between predicted probabilities and observed responses for binary or binomial response models. These measures assess the predictive ability of the model. The displayed statistics are the concordance index *c* (the area under the ROC curve, AUC), Somers' *D* statistic (Gini's coefficient), the Goodman-Kruskal gamma statistic, and Kendall's tau-*a* statistic. For more information, see the section "Association Statistics" on page 4879.

CL

requests that confidence limits be constructed for each of the parameter estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

CTABLE< =*SAS-data-set* **>**

OUTROC< =*SAS-data-set* **>**

displays a table for binary or binomial response models that contains the frequencies of observations that are correctly and incorrectly classified as events and nonevents, the sensitivity, the 1–specificity, the positive and negative predictive values, and the correct classification rate. For more information, see the section "Classification Table and ROC Curves" on page 4877.

Classification is carried out by initially binning the predicted probabilities as discussed in the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880. The PRIOR= option does not change the reported predicted probabilities.

Because the number of cutpoints can be very large, you can store the table in an output data set. If you specify a PARTITION statement, then the statistics are computed by their roles, and a Role variable indicates to which partition the computations belong.

CUTPOINT=*value*

specifies a value between 0 and 1 used for classifying observations when you have a binary or binomial response variable. If the predicted probability of an observation equals or exceeds the cutpoint, the observation is classified as an event; otherwise it is classified as a nonevent. This option affects computation of the misclassification rate and the true positive and true negative fractions in the "Partition Fit Statistics" table. By default, CUTPOINT=0.5.

DDFM=RESIDUAL | NONE

specifies how degrees of freedom for statistical inference be determined in the "Parameter Estimates Table."

The HPLOGISTIC procedure always displays the statistical tests and confidence intervals in the "Parameter Estimates" tables in terms of a t test and a two-sided probability from a t distribution. With the DDFM= option, you can control the degrees of freedom of this t distribution and thereby switch between small-sample inference and large-sample inference based on the normal or chi-square distribution.

The default is DDFM=NONE, which leads to z-based statistical tests and confidence intervals. The HPLOGISTIC procedure then displays the degrees of freedom in the DF column as Infty, the p-values are identical to those from a Wald chi-square test, and the square of the t value equals the Wald chi-square statistic.

If you specify DDFM=RESIDUAL, the degrees of freedom are finite and determined by the number of usable frequencies (observations) minus the number of nonredundant model parameters. This leads to t-based statistical tests and confidence intervals. If the number of frequencies is large relative to the number of parameters, the inferences from the two degrees-of-freedom methods are almost identical.

INCLUDE=*n*

INCLUDE=*single-effect*

INCLUDE=(*effects***)**

forces effects to be included in all models. If you specify INCLUDE=*n*, then the first *n* effects that are listed in the MODEL statement are included in all models. If you specify INCLUDE=*single-effect* or if you specify a list of effects within parentheses, then the specified effects are forced into all models. The effects that you specify in the INCLUDE= option must be explanatory effects that are specified in the MODEL statement before the slash (/).

LACKFIT< (DFREDUCE=*r* **NGROUPS=***G***) >**

performs the Hosmer and Lemeshow goodness-of-fit test (Hosmer and Lemeshow 2000) for binary response models.

The subjects are divided into at most G groups of roughly the same size, based on the percentiles of the estimated probabilities. You can specify G as any integer greater than or equal to 5; by default, G=10. Let the actual number of groups created be g. The discrepancies between the observed and expected number of observations in these g groups are summarized by the Pearson chi-square statistic, which is then compared to a chi-square distribution with g–r degrees of freedom. You can specify a nonnegative integer r that satisfies $g–r \geq 1$; by default, r=2.

A small p-value suggests that the fitted model is not an adequate model. For more information, see the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880.

LINK=*keyword*

specifies the link function for the model. The *keywords* and the associated link functions are shown in Table 61.5.

Table 61.5 Built-in Link Functions of the HPLOGISTIC Procedure

LINK=	Link Function	$g(\mu) = \eta =$
CLOGLOG | CLL	Complementary log-log	$\log(-\log(1-\mu))$
GLOGIT | GENLOGIT	Generalized logit	
LOGIT	Logit	$\log(\mu/(1-\mu))$
LOGLOG	Log-log	$-\log(-\log(\mu))$
PROBIT	Probit	$\Phi^{-1}(\mu)$

For the probit and cumulative probit links, $\Phi^{-1}(\cdot)$ denotes the quantile function of the standard normal distribution.

If the response variable has more than two categories, the HPLOGISTIC procedure fits a model with a cumulative link function based on the specified link. However, if you specify LINK=GLOGIT, the procedure assumes a generalized logit model for nominal (unordered) data, regardless of the number of response categories.

NOCHECK

disables the checking process that determines whether maximum likelihood estimates of the regression parameters exist. For more information, see the section "Existence of Maximum Likelihood Estimates" on page 4873.

NOINT

requests that no intercept be included in the model. An intercept is included by default. The NOINT option is not available in multinomial models.

OFFSET=*variable*

specifies a *variable* to be used as an offset to the linear predictor. An offset plays the role of an effect whose coefficient is known to be 1. The offset variable cannot appear in the CLASS statement or elsewhere in the MODEL statement. Observations with missing values for the offset variable are excluded from the analysis.

PRIOR=*SAS-data-set*

PRIOR=*number*

PEVENT=*number*

PRIOR=ALLDATA

specifies prior probabilities (prevalences) that are used for computing posterior predicted probabilities. When you know what percentage of the population has a rare event and you oversample that rare event, specifying the prior probabilities as the prevalence of events in your population enables you to produce posterior probabilities that reflect the population, not the data.

You can specify your priors in a SAS data set in which a _PRIOR_ column contains the prior probabilities. For events/trials MODEL statement syntax, this data set should also include an _OUTCOME_ variable that contains the values EVENT and NONEVENT; for single-trial syntax, this data set should include the response variable that contains the unformatted response categories. Each row of the data

set contains a unique response variable level and its prior. For binary and binomial response models, you can instead specify the probability of an event as *number*. If you also specify a PARTITION statement, you can specify PRIOR=ALLDATA to compute the prevalences as the observed proportions of the response levels gathered across all the roles.

If your response Y takes values $i = 1, \ldots, k$ that have observed empirical probabilities $\mathrm{OldPrior}_i = \frac{n_i}{n}$, you specify priors Prior_i, and your model predicted probabilities are $\hat{p}_i$, then the posterior predicted probabilities Post_i are computed as

$$\mathrm{Post}_i = \frac{\hat{p}_i \frac{\mathrm{Prior}_i}{\mathrm{OldPrior}_i}}{\sum_{j=1}^{k} \hat{p}_j \frac{\mathrm{Prior}_j}{\mathrm{OldPrior}_j}}$$

The POST= option in the OUTPUT statement writes the posterior to the output data set. If your priors are identical to the empirical probabilities, then the posteriors are identical to the model-predicted probabilities.

The priors do not affect the model-fitting process, but modify the "Classification" table statistics PPV, NPV, and Percent Correct as described in the section "Classification Table and ROC Curves" on page 4877.

RSQUARE

R2

requests a generalized coefficient of determination (R square, R^2) and a scaled version thereof for the fitted model. The results are added to the "Fit Statistics" table. For more information about the computation of these measures, see the section "Generalized Coefficient of Determination" on page 4876.

START=*n*

START=*single-effect*

START=(*effects***)**

begins the selection process from the designated initial model for the FORWARD and STEPWISE selection methods. If you specify START=*n*, then the starting model includes the first *n* effects that are listed in the MODEL statement. If you specify START=*single-effect* or if you specify a list of effects within parentheses, then the starting model includes those specified effects. The effects that you specify in the START= option must be explanatory effects that are specified in the MODEL statement before the slash (/). The START= option is not available when you specify METHOD=BACKWARD in the SELECTION statement.

STB

displays the standardized estimates for the parameters in the "Parameter Estimates" table. The standardized estimate of β_i is given by $\widehat{\beta}_i/(s/s_i)$, where s_i is the total sample standard deviation for the *i*th explanatory variable and

$$s = \begin{cases} \pi/\sqrt{3} & \text{LOGIT and GLOGIT links} \\ 1 & \text{PROBIT link} \\ \pi/\sqrt{6} & \text{CLOGLOG and LOGLOG links} \end{cases}$$

The sample standard deviations for parameters that are associated with CLASS variables are computed using their codings. The standardized estimates are not computed for the intercept parameters.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< **COPYVARS=***(variables)* >
< *keyword* < *=name* > >. . . < *keyword* < *=name* > > < / *options* > **;**

The OUTPUT statement creates a data set that contains observationwise statistics that PROC HPLOGISTIC computes after fitting the model. The variables in the input data set are *not* included in the output data set, in order to avoid data duplication for large data sets; however, variables that you specify in the ID statement or COPYVAR= option are included.

The output statistics are computed based on the final parameter estimates. If the optimization does not converge, then the output data set is not created.

When there are more than two response levels, values are computed only for variables that are named by the XBETA, POST, and PRED keywords; the other variables have missing values. These statistics are computed for every response category, and the automatic variable _LEVEL_ identifies the response category upon which the computed values are based. That is, every observation generates several rows in the output data set. If you also specify the OBSCAT option, then the observationwise statistics are computed only for the observed response category, as indicated by the value of the _LEVEL_ variable.

For observations in which only the response variable is missing, values of the XBETA, POST, and PRED statistics are computed even though these observations do not affect the model fit. This enables, for instance, predicted probabilities to be computed for new observations.

You can specify the following syntax elements in the OUTPUT statement before the slash (/).

OUT=*SAS-data-set*

DATA=*SAS-data-set*

specifies the name of the output data set. If the OUT= (or DATA=) option is omitted, the procedure uses the DATA*n* convention to name the output data set.

COPYVAR=*variable*

COPYVARS=(*variables***)**

transfers one or more *variables* from the input data set to the output data set. Variables named in an ID statement are also copied from the input data set to the output data set.

keyword **< =***name* **>**

specifies a statistic to include in the output data set and optionally names the variable *name*. If you do not provide a *name*, the HPLOGISTIC procedure assigns a default name based on the type of statistic requested.

The following are valid *keywords* for adding statistics to the OUTPUT data set:

LINP | XBETA

requests the linear predictor $\eta = \mathbf{x}'\boldsymbol{\beta}$. The default name is Xbeta.

PEARSON | PEARS | RESCHI

requests the Pearson residual, $\frac{\sqrt{wn}(y/n-\mu)}{\sqrt{\mu(1-\mu)}}$, where μ is the estimate of the predicted event probability, w is the weight of the observation, and n is the number of binomial trials (n=1 for binary observations). The default name is Pearson. This statistic is not computed for multinomial models.

POSTERIOR | POST

requests a numeric variable that contains the posterior predicted probability of each observation that is used in fitting the model. The default name is _POST_. If you do not specify the PRIOR option in the MODEL statement, then this value is the same as the predicted probability.

PREDICTED | PRED | P

requests predicted values (predicted probabilities of events) for the response variable. The default name is Pred.

RESIDUAL | RESID | R

requests the raw residual, $y - \mu$, where μ is the estimate of the predicted event probability. The default name is Residual. This statistic is not computed for multinomial models.

ROLE

requests a numeric variable that indicates the role played by each observation in fitting the model. The default name is _ROLE_. Table 61.6 shows how this variable is interpreted for each observation.

Table 61.6 Role Interpretation

Value	Observation Role
0	Not used
1	Training
2	Validation
3	Testing

If you do not partition the input data by specifying a PARTITION statement, then the role variable value is 1 for observations that are used in fitting the model and 0 for observations that have at least one missing or invalid value for the response, regressors, frequency or weight variables.

You can specify the following *options* in the OUTPUT statement after the slash (/):

ALLSTATS

adds all available statistics to the output data set.

OBSCAT

requests (for multinomial models) that observationwise statistics be produced for the response level only. If you do not specify the OBSCAT option and the response variable has J levels, then the following outputs are created: for cumulative link models, $J - 1$ records are output for every observation in the input data that corresponds to the $J - 1$ lower-ordered response categories; for generalized logit models, J records are output that correspond to all J response categories.

PARTITION Statement

PARTITION *partition-options* **;**

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. For more information, see the section "Using Validation and Test Data" on page 4874. Either you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations for each role.

You must specify one and only one of the following *partition-options*:

ROLEVAR | ROLE=*variable***(< TEST='***value***' > < TRAIN='***value***' > < VALIDATE='***value***' >)**
names the variable in the input data set whose values are used to assign roles to each observation. The formatted values of this variable that are used to assign observations roles are specified in the TEST=, TRAIN=, and VALIDATE= suboptions. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboption are assigned to training.

FRACTION(< TEST=*fraction***> < VALIDATE=***fraction***> < SEED=***number***>)**
randomly assigns specified proportions of the observations in the input data set to the roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and the VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role. The SEED= option specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify a seed, or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer's clock.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of the HPLOGISTIC procedure.

The PERFORMANCE statement for high-performance statistical procedures is documented in the section "PERFORMANCE Statement" on page 569.

SELECTION Statement

SELECTION < *options* > **;**

The SELECTION statement performs model selection by examining whether effects should be added to or removed from the model according to rules that are defined by model selection methods. The statement is fully documented in the section "SELECTION Statement" on page 576.

The HPLOGISTIC procedure supports the following effect-selection methods in the SELECTION statement:

METHOD=NONE	results in no model selection. This method fits the full model.
METHOD=FORWARD	performs forward selection. This method starts with no effects in the model and adds effects.
METHOD=BACKWARD	performs backward elimination. This method starts with all effects in the model and deletes effects.
METHOD=BACKWARD(FAST)	performs fast backward elimination when SELECT=SL. This method starts with all effects in the model and deletes effects without refitting the model.
METHOD=STEPWISE	performs stepwise selection. This method is similar to the FORWARD method except that effects already in the model do not necessarily stay there.

The default criterion for the SELECT=, CHOOSE=, and STOP= options in the SELECTION statement is the significance level (SL), where effects enter and leave the model based on the significance level of an approximate chi-square test statistic. You can specify the following criteria in the SELECT=, CHOOSE=, and STOP= options:

AIC	uses Akaike's information criterion (Akaike 1974).
AICC	uses a small-sample bias corrected version of Akaike's information criterion, as promoted in Hurvich and Tsai (1989) and Burnham and Anderson (1998), for example.
BIC | SBC	uses Schwarz' Bayesian criterion (Schwarz 1978).
SL	uses the significance level of the score test as the criterion (not available for a CHOOSE= option).
VALIDATE	uses the average square error (ASE) that is computed on the VALIDATE partition as the criterion (not available for a SELECT= option).

For more information, see the section "Information Criteria" on page 4876. If you specify the PARTITION statement, then the AIC, AICC, BIC, and SL statistics are computed on the training data set; otherwise they are computed on the full data set.

NOTE: If you use the fast backward elimination method, then the –2 log likelihood, AIC, AICC, and BIC statistics are approximated at each step where the model is not refit, and hence they do not match the values that are computed when that model is fit outside the selection routine. Similarly, if you specify SELECT=AIC, AICC, or BIC, the selection criteria are estimated (Lawless and Singhal 1978), and hence they do not match the values that are computed when that model is fit outside the selection routine.

NOTE: The default model hierarchy method is HIERARCHY=NONE for the stepwise, forward, and fast backward selection methods. The backward elimination method always uses the HIERARCHY=SINGLE method.

When you specify the DETAILS= option in the SELECTION statement, the HPLOGISTIC procedure produces the following:

DETAILS=SUMMARY	produces a summary table that shows the effect that is added or removed at each step along with the *p*-value and the SELECT=, CHOOSE=, and STOP= criteria. The summary table is produced by default if the DETAILS= option is not specified.
DETAILS=STEPS	produces a detailed listing of all candidates at each step and their ranking in terms of the selection criterion for entry into or removal from the model.
DETAILS=ALL	produces the preceding two tables and a table of selection details, which displays fit statistics for the model at each step of the selection process and an approximate chi-square score or likelihood ratio statistic.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations with nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, then all observations used in the analysis are assigned a weight of 1.

Details: HPLOGISTIC Procedure

Missing Values

Any observation with missing values for the response, frequency, weight, offset, or explanatory variables is excluded from the analysis; however, missing values are valid for response and explanatory variables that are specified with the MISSING option in the CLASS statement. Observations with a nonpositive weight or with a frequency less than 1 are also excluded.

The estimated linear predictor and the fitted probabilities are not computed for any observation that has missing offset or explanatory variable values. However, if only the response value is missing, the linear predictor and the fitted probabilities can be computed and output to a data set by using the OUTPUT statement.

Response Distributions

The response distribution is the probability distribution of the response (target) variable. The HPLOGISTIC procedure can fit data for the following distributions:

- binary distribution
- binomial distribution
- multinomial distribution

The expressions for the log-likelihood functions of these distributions are given in the next section.

The binary (or Bernoulli) distribution is the elementary distribution of a discrete random variable that can take on two values with probabilities p and $1 - p$. Suppose the random variable is denoted Y and

$$\Pr(Y = 1) = p$$
$$\Pr(Y = 0) = 1 - p$$

The value associated with probability p is often termed the *event* or "success"; the complementary event is termed the *non-event* or "failure." A Bernoulli experiment is a random draw from a binary distribution and generates events with probability p.

If $Y_1, \cdots, Y_n$ are n independent Bernoulli random variables, then their sum follows a binomial distribution. In other words, if $Y_i = 1$ denotes an event (success) in the ith Bernoulli trial, a binomial random variable is the number of events (successes) in n independent Bernoulli trials. If you use the events/trials syntax in the MODEL statement, the HPLOGISTIC procedure fits the model as if the data had arisen from a binomial distribution. For example, the following statements fit a binomial regression model with regressors x1 and x2. The variables e and t represent the events and trials for the binomial distribution:

```
proc hplogistic;
   model e/t = x1 x2;
run;
```

If the events/trials syntax is used, then both variables must be numeric and the value of the events variable cannot be less than 0 or exceed the value of the trials variable. A "Response Profile" table is not produced for binomial data.

The multinomial distribution is a generalization of the binary distribution and allows for more than two outcome categories. Because there are more than two possible outcomes for the multinomial distribution, the terminology of "successes," "failures," "events," and "non-events" no longer applies. With multinomial data, these outcomes are generically referred to as "categories" or levels.

Whenever the HPLOGISTIC procedure determines that the response variable has more than two levels (unless the events/trials syntax is used), the procedure fits the model as if the data had arisen from a multinomial distribution. By default, it is then assumed that the response categories are ordered and a cumulative link model is fit by applying the default or specified link function. If the response categories are unordered, then you should fit a generalized logit model by choosing LINK=GLOGIT in the MODEL statement.

Log-Likelihood Functions

The HPLOGISTIC procedure forms the log-likelihood functions of the various models as

$$L(\boldsymbol{\mu};\mathbf{y}) = \sum_{i=1}^{n} f_i\, l(\mu_i; y_i, w_i)$$

where $l(\mu_i; y_i, w_i)$ is the log-likelihood contribution of the ith observation with weight w_i and f_i is the value of the frequency variable. For the determination of w_i and f_i, see the WEIGHT and FREQ statements. The individual log-likelihood contributions for the various distributions are as follows.

Binary Distribution

The HPLOGISTIC procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith binary observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i) &= y_i \log\{\mu_i\} + (1 - y_i)\log\{1 - \mu_i\}
\end{aligned}$$

Here, μ_i is the probability of an event, and the variable y_i takes on the value 1 for an event and the value 0 for a non-event. The inverse link function $g^{-1}(\cdot)$ maps from the scale of the linear predictor η_i to the scale of the mean. For example, for the logit link (the default),

$$\mu_i(\boldsymbol{\beta}) = \frac{\exp\{\eta_i\}}{1 + \exp\{\eta_i\}}$$

You can control which binary outcome in your data is modeled as the event with the *response-options* in the MODEL statement, and you can choose the link function with the LINK= option in the MODEL statement.

If a WEIGHT statement is given and w_i denotes the weight for the current observation, the log-likelihood function is computed as

$$l(\mu_i(\boldsymbol{\beta}); y_i, w_i) = w_i\, l(\mu_i(\boldsymbol{\beta}); y_i)$$

Binomial Distribution

The HPLOGISTIC procedure computes the log-likelihood function $l(\mu_i(\boldsymbol{\beta}); y_i)$ for the ith binomial observation as

$$\begin{aligned}
\eta_i &= \mathbf{x}_i'\boldsymbol{\beta} \\
\mu_i(\boldsymbol{\beta}) &= g^{-1}(\eta_i) \\
l(\mu_i(\boldsymbol{\beta}); y_i, w_i) &= w_i\,(y_i \log\{\mu_i\} + (n_i - y_i)\log\{1 - \mu_i\}) \\
&\quad + w_i\,(\log\{\Gamma(n_i + 1)\} - \log\{\Gamma(y_i + 1)\} - \log\{\Gamma(n_i - y_i + 1)\})
\end{aligned}$$

where y_i and n_i are the values of the events and trials of the ith observation, respectively. μ_i measures the probability of events (successes) in the underlying Bernoulli distribution whose aggregate follows the binomial distribution.

Multinomial Distribution

The multinomial distribution modeled by the HPLOGISTIC procedure is a generalization of the binary distribution; it is the distribution of a single draw from a discrete distribution with J possible values. The log-likelihood function for the ith observation is thus deceptively simple:

$$l(\boldsymbol{\mu}_i; \mathbf{y}_i, w_i) = w_i \sum_{j=1}^{J} y_{ij} \log\{\mu_{ij}\}$$

In this expression, J denotes the number of response categories (the number of possible outcomes) and μ_{ij} is the probability that the ith observation takes on the response value associated with category j. The category probabilities must satisfy

$$\sum_{j=1}^{J} \mu_j = 1$$

and the constraint is satisfied by modeling $J - 1$ categories. In models that have ordered response categories, the probabilities are expressed in cumulative form, so that the last category is redundant. In generalized logit models (multinomial models that have unordered categories), one category is chosen as the reference category and the linear predictor in the reference category is set to zero. For more information, see the REF= *response-option* in the MODEL statement.

Existence of Maximum Likelihood Estimates

The likelihood equation for a logistic regression model does not always have a finite solution. Sometimes there is a nonunique maximum on the boundary of the parameter space, at infinity. The existence, finiteness, and uniqueness of maximum likelihood estimates for the logistic regression model depend on the patterns of data points in the observation space (Albert and Anderson 1984; Santner and Duffy 1986).

Consider a binary response model. Let Y_j be the response of the jth subject, and let $\mathbf{x}_j$ be the vector of explanatory variables (including the constant 1 that is associated with the intercept). There are three mutually exclusive and exhaustive types of data configurations: complete separation, quasi-complete separation, and overlap.

Complete Separation There is a complete separation of data points if there exists a vector $\mathbf{b}$ that correctly allocates all observations to their response groups; that is,

$$\begin{cases} \mathbf{b}'\mathbf{x}_j > 0 & Y_j = 1 \\ \mathbf{b}'\mathbf{x}_j < 0 & Y_j = 2 \end{cases}$$

This configuration produces nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, the log likelihood diminishes to 0, and the dispersion matrix becomes unbounded.

Quasi-complete Separation The data are not completely separable, but there is a vector $\mathbf{b}$ such that

$$\begin{cases} \mathbf{b}'\mathbf{x}_j \geq 0 & Y_j = 1 \\ \mathbf{b}'\mathbf{x}_j \leq 0 & Y_j = 2 \end{cases}$$

and equality holds for at least one subject in each response group. This configuration also yields nonunique infinite estimates. If the iterative process of maximizing the likelihood function is

allowed to continue, the dispersion matrix becomes unbounded and the log likelihood diminishes to a nonzero constant.

Overlap If neither complete nor quasi-complete separation exists in the sample points, there is an overlap of sample points. In this configuration, the maximum likelihood estimates exist and are unique.

The HPLOGISTIC procedure uses a simple empirical approach to recognize the data configurations that lead to infinite parameter estimates. The basis of this approach is that any convergence method of maximizing the log likelihood must yield a solution that indicates complete separation, if such a solution exists. Upon convergence, if the predicted response equals the observed response for every observation, there is a complete separation of data points.

If the data are not completely separated, if an observation is identified to have an extremely large probability (≥ 0.95) of predicting the observed response, and if there have been at least eight iterations, then there are two possible situations. First, there is overlap in the data set, the observation is an atypical observation of its own group, and the iterative process stopped when a maximum was reached. Second, there is quasi-complete separation in the data set, and the asymptotic dispersion matrix is unbounded. If any of the diagonal elements of the dispersion matrix for the standardized observation vector (all explanatory variables standardized to zero mean and unit variance) exceeds 5,000, quasi-complete separation is declared. If either complete separation or quasi-complete separation is detected, a note is displayed in the procedure output.

Checking for quasi-complete separation is less foolproof than checking for complete separation. If neither type of separation is discovered and your parameter estimates have large standard errors, then this indicates that your data might be separable. The NOCHECK option in the MODEL statement turns off the process of checking for infinite parameter estimates.

Using Validation and Test Data

When you have sufficient data, you can subdivide your data into three parts called the training, validation, and test data. During the selection process, models are fit on the training data, and the prediction errors for the models so obtained are found by using the validation data. This prediction error on the validation data can be used to decide when to terminate the selection process and to decide which model. Finally, after a selected model has been obtained, the test set can be used to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases you might want to use only training and test data. For example, you might decide to use an information criterion to decide which effects to include and when to terminate the selection process. In this case no validation data are required, but test data can still be useful in assessing the predictive performance of the selected model. In other cases you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) note that it is difficult to provide a general rule for how many observations you should assign to each role. They note that a typical split might be 50% for training and 25% each for validation and testing.

You use a PARTITION statement to logically subdivide the DATA= data set into separate roles. You can specify the fractions of the data that you want to reserve as test data and validation data. For example, the following statements randomly subdivide the inData data set, reserving 50% for training and 25% each for validation and testing:

```
proc hplogistic data=inData;
  partition fraction(test=0.25 validate=0.25);
  ...
run;
```

You can specify the SEED= option in the PARTITION statement to create the same partition data sets.

In some cases you might need to exercise more control over the partitioning of the input data set. You can do this by naming both a variable in the input data set and a formatted value of that variable for each role. For example, the following statements assign roles to the observations in the inData data set that are based on the value of the variable Group in that data set. Observations whose value of Group is `'Group 1'` are assigned for testing, and those whose value is `'Group 2'` are assigned to training. All other observations are ignored.

```
proc hplogistic data=inData;
  partition roleVar=Group(test='Group 1' train='Group 2')
  ...
run;
```

When you have reserved observations for training, validation, and testing, a model that is fit on the training data is scored on the validation and test data, and statistics are computed separately for each of these subsets. For more information, see the section "Partition Fit Statistics" on page 4875. For an illustration, see Example 61.4.

Using the Validation Statistic as the CHOOSE= Criterion

When you specify the CHOOSE=VALIDATE suboption of the METHOD= option in the SELECTION statement, the ASE is computed on the validation data for the models at each step of the selection process. The smallest model at any step that yields the smallest validation ASE is selected.

Using the Validation Statistic as the STOP= Criterion

When you specify the STOP=VALIDATE suboption of the METHOD= option in the SELECTION statement, the ASE is computed on the validation data for the models at each step of the selection process. At step k of the selection process, the best candidate effect to enter or leave the current model is determined and the validation ASE for this new model is computed. If this validation ASE is greater than the validation ASE for the model at step k, then the selection process terminates at step k.

Partition Fit Statistics

Specifying a PARTITION statement modifies the display of many tables by adding separate rows or columns for the training, validation, and test data sets. A "Partition Fit Statistics" table is also produced, and it displays the following statistics, which are useful for assessing the model and which should be very similar for the different roles when the training data are representative of the other data sets: average square error, misclassification rate, R^2, max-rescaled R^2, and McFadden's R^2. Binary and binomial response models also display the following statistics: area under the ROC curve, the Hosmer-Lemeshow test p-value, difference of means, Somers' D statistic, and the true positive and negative fractions. Polytomous response models also display the true fraction for each response level. For more information, see the sections "Model Fit and Assessment Statistics" on page 4876 and "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880.

Model Fit and Assessment Statistics

Information Criteria

The calculation of the information criteria uses the following formulas, where p denotes the number of effective parameters in the candidate model, F denotes the sum of frequencies used, and l is the log likelihood evaluated at the converged estimates:

$$\mathrm{AIC} = -2l + 2p$$

$$\mathrm{AICC} = \begin{cases} -2l + 2pF/(F-p-1) & \text{when } F > p+2 \\ -2l + 2p(p+2) & \text{otherwise} \end{cases}$$

$$\mathrm{BIC} = -2l + p\log(F)$$

If you do not specify a FREQ statement, F equals n, the number of observations used.

Generalized Coefficient of Determination

The goal of a coefficient of determination, also known as an R-square measure, is to express the agreement between a stipulated model and the data in terms of variation in the data that is explained by the model. In linear models, the R-square measure is based on residual sums of squares; because these are additive, a measure bounded between 0 and 1 is easily derived.

In more general models where parameters are estimated by the maximum likelihood principle, Cox and Snell (1989, pp. 208–209) and Magee (1990) proposed the following generalization of the coefficient of determination:

$$R^2 = 1 - \left\{ \frac{L(\mathbf{0})}{L(\widehat{\boldsymbol{\beta}})} \right\}^{\frac{2}{n}}$$

Here, $L(\mathbf{0})$ is the likelihood of the intercept-only model, $L(\widehat{\boldsymbol{\beta}})$ is the likelihood of the specified model, and n denotes the number of observations used in the analysis. This number is adjusted for frequencies if a FREQ statement is present and is based on the trials variable for binomial models.

As discussed in Nagelkerke (1991), this generalized R-square measure has properties similar to the coefficient of determination in linear models. If the model effects do not contribute to the analysis, $L(\widehat{\boldsymbol{\beta}})$ approaches $L(\mathbf{0})$ and R^2 approaches zero.

However, R^2 does not have an upper limit of 1. Nagelkerke suggested a rescaled generalized coefficient of determination, R^2_N, which achieves an upper limit of 1 by dividing R^2 by its maximum value:

$$R^2_{\max} = 1 - \{L(\mathbf{0})\}^{\frac{2}{n}}$$

$$R^2_N = \frac{R^2}{R^2_{\max}}$$

Another measure from McFadden (1974) is also bounded by 0 and 1:

$$R^2_M = 1 - \left(\frac{\log L(\widehat{\boldsymbol{\beta}})}{\log L(\mathbf{0})} \right)$$

If you specify the RSQUARE option in the MODEL statement, the HPLOGISTIC procedure computes R^2 and R^2_N. All three measures are computed for each data role when you specify a PARTITION statement.

These measures are most useful for comparing competing models that are not necessarily nested—that is, models that cannot be reduced to one another by simple constraints on the parameter space. Larger values of the measures indicate better models.

Classification Table and ROC Curves

For binary response data, the response Y is either an event or a nonevent; let the response Y take the value 1 for an event and 2 for a nonevent. From the fitted model, a predicted event probability $\hat{\pi}_i$ can be computed for each observation i. If the predicted event probability equals or exceeds some cutpoint value $z \in [0, 1]$, the observation is classified as an event; otherwise, it is classified as a nonevent. Suppose n_1 of n individuals experience an event, such as a disease, and the remaining $n_2 = n - n_1$ individuals are nonevents. The 2×2 decision matrix in Table 61.7 is obtained by cross-classifying the observed and predicted responses, where n_{ij} is the total number of observations that are observed to have Y=i and are classified into j. In this table, let $Y = 1$ denote an observed event and $Y = 2$ denote a nonevent, and let $D = 1$ indicate that the observation is classified as an event and $D = 2$ denote that the observation is classified as a nonevent.

Table 61.7 Decision Matrix

	$D = 1$ ($\hat{\pi} \geq z$)	$D = 2$ ($\hat{\pi} < z$)	**Total**
$Y = 1$ (event)	n_{11}	n_{12}	n_1
$Y = 2$ (nonevent)	n_{21}	n_{22}	n_2

In the decision matrix, the *number of true positives*, n_{11}, is the number of event observations that are correctly classified as events; the *number of false positives*, n_{21}, is the number of nonevent observations that are incorrectly classified as events; the *number of false negatives*, n_{12}, is the number of event observations that are incorrectly classified as nonevents; and the *number of true negatives*, n_{22}, is the number of nonevent observations that are correctly classified as nonevents. The statistics in Table 61.8 are computed from the preceding decision matrix.

Table 61.8 Statistics from the Decision Matrix with Cutpoint z

Statistic	**Equation**	**OUTROC Column**
Cutpoint	z	ProbLevel
Number of true positives	n_{11}	TruePos
Number of true negatives	n_{22}	TrueNeg
Number of false positives	n_{21}	FalsePos
Number of false negatives	n_{12}	FalseNeg
Sensitivity	n_{11}/n_1	TPF (true positive fraction)
1–specificity	n_{21}/n_2	FPF (false positive fraction)
Correct classification rate	$(n_{11} + n_{22})/n$	PercentCorrect (PC)
Misclassification rate	1 – PC	
Positive predictive value	$n_{11}/(n_{11} + n_{21})$	PPV
Negative predictive value	$n_{22}/(n_{12} + n_{22})$	NPV

The accuracy of the classification is measured by its ability to predict events and nonevents correctly. *Sensitivity* (TPF, true positive fraction) is the proportion of event responses that are predicted to be events. *Specificity* (1–FPF, true negative fraction) is the proportion of nonevent responses that are predicted to be nonevents.

You can also measure accuracy by how well the classification predicts the response. The *positive predictive value* (PPV) is the proportion of observations classified as events that are correctly classified. The *negative predictive value* (NPV) is the proportion of observations classified as nonevents that are correctly classified. The *correct classification rate* (PC) is the proportion of observations that are correctly classified.

If you also specify a PRIOR= option, then PROC HPLOGISTIC uses Bayes' theorem to modify the PPV, NPV, and PC as follows. Results of the classification are represented by two conditional probabilities: sensitivity, $\Pr(D=1|Y=1) = \frac{n_{11}}{n_1}$, and one minus the specificity, $\Pr(D=1|Y=2) = \frac{n_{21}}{n_2}$.

If the prevalence of the disease in the population $\Pr(Y=1)$ is provided by the value of the PRIOR= option, then the PPV, NPV, and PC are given by Fleiss (1981, pp. 4–5) as follows:

$$\begin{aligned}
\text{PPV} &= \Pr(Y=1|D=1) = \frac{\Pr(Y=1)\Pr(D=1|Y=1)}{\Pr(D=1|Y=2) + \Pr(Y=1)[\Pr(D=1|Y=1) - \Pr(D=1|Y=2)]} \\
\text{NPV} &= \Pr(Y=2|D=2) = \frac{[1-\Pr(D=1|Y=2)][1-\Pr(Y=1)]}{1-\Pr(D=1|Y=2) - \Pr(Y=1)[\Pr(D=1|Y=1) - \Pr(D=1|Y=2)]} \\
\text{PC} &= \Pr(Y=1|D=1) + \Pr(Y=2|D=2) \\
&= \Pr(D=1|Y=1)\Pr(Y=1) + \Pr(D=2|Y=2)[1-\Pr(Y=1)]
\end{aligned}$$

If you do not specify the PRIOR= option, then PROC HPLOGISTIC uses the sample proportion of diseased individuals; that is, $\Pr(Y=1) = n_1/n$. In such a case, the preceding values reduce to those in Table 61.8. Note that for a stratified sampling situation in which n_1 and n_2 are chosen a priori, n_1/n is not a desirable estimate of $\Pr(Y=1)$, so you should specify a PRIOR= option.

PROC HPLOGISTIC constructs the data for a receiver operating characteristic (ROC) curve by initially binning the predicted probabilities as discussed in the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880, then moving the cutpoint from 0 to 1 along the bin boundaries (so that the cutpoints correspond to the predicted probabilities), and then selecting those cutpoints where a change in the decision matrix occurs. The CTABLE option produces a table that includes these cutpoints and the statistics in Table 61.8 that correspond to each cutpoint. You can output this table to a SAS data set by specifying the CTABLE= option (see Table 61.8 for the column names), and you can display the ROC curve by using the SGPLOT procedure as shown in Example 61.2.

The area under the ROC curve (AUC), as determined by the trapezoidal rule, is given by the concordance index c, which is described in the section "Association Statistics" on page 4879.

For more information about the topics in this section, see Pepe (2003).

The "Partition Fit Statistics" table displays the misclassification rate, true positive fraction, true negative fraction, and AUC according to their roles. If you have a polytomous response, then instead of classifying according to a cutpoint, PROC HPLOGISTIC classifies the observation into the lowest response level (which has the largest predicted probability for that observation) and similarly computes a true response-level fraction.

Association Statistics

If you specify the ASSOCIATION option in the MODEL statement, PROC HPLOGISTIC displays measures of association between predicted probabilities and observed responses for binary or binomial response models. These measures assess the predictive ability of a model.

Of the n pairs of observations in the data set with different responses, let n_c be the number of pairs where the observation that has the lower-ordered response value has a lower predicted probability, let n_d be the number of pairs where the observation that has the lower-ordered response value has a higher predicted probability, and let $n_t = n - n_c - n_d$ be the rest. Let N be the sum of observation frequencies in the data. Then the following statistics are reported:

$$\begin{aligned}
\text{concordance index } c \text{ (AUC)} &= (n_c + 0.5n_t)/n \\
\text{Somers' } D \text{ (Gini coefficient)} &= (n_c - n_d)/n \\
\text{Goodman-Kruskal gamma} &= (n_c - n_d)/(n_c + n_d) \\
\text{Kendall's tau-}a &= (n_c - n_d)/(0.5N(N-1))
\end{aligned}$$

Classification of the pairs is carried out by initially binning the predicted probabilities as discussed in the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880. The concordance index, c, is an estimate of the AUC, which is the area under the receiver operating characteristic (ROC) curve. If there are no ties, then Somers' D (Gini's coefficient) = $2c$–1.

If you specify a PARTITION statement, then PROC HPLOGISTIC displays the AUC and Somers' D in the "Association" and "Partition Fit Statistics" tables according to their roles.

Average Square Error

The average square error (ASE) is the average of the squared differences between the responses and the predictions. When you have a discrete number of response levels, the ASE is modified as shown in Table 61.9 (Brier 1950; Murphy 1973); it is also called the Brier score or Brier reliability.

Table 61.9 Average Square Error Computations

Response Type	ASE (Brier Score)
Polytomous	$\frac{1}{F}\sum_i f_i \sum_j (y_{ij} - \widehat{\pi}_{ij})^2$
Binary	$\frac{1}{F}\sum_i f_i (y_i(1-\widehat{\pi}_i)^2 + (1-y_i)\widehat{\pi}_i^2)$
Binomial	$\frac{1}{F}\sum_i f_i (r_i/t_i - \widehat{\pi}_i)^2$

In Table 61.9, $F = \sum_i f_i$, r_i is the number of events, t_i is the number of trials in binomial response models, and y_i=1 for events and 0 for nonevents in binary response models. For polytomous response models, y_{ij}=1 if the ith observation has response level j, and π_{ij} is the model-predicted probability of response level j for observation i.

Mean Difference

For a binary response model, write the mean of the model-predicted probabilities of event (Y=1) observations as $\overline{X}_1 = \frac{\sum_{i=1}^{n}(\pi_i|y_i=1)}{n_1}$ and of the nonevent (Y=2) observations as $\overline{X}_2 = \frac{\sum_{i=1}^{n}(\pi_i|y_i=2)}{n_1}$. The mean difference, or more precisely the difference of means, is $\overline{X}_1 - \overline{X}_2$, which Tjur (2009) relates to other R-square measures and calls the *coefficient of discrimination* because it is a measure of the model's ability to distinguish between the event and nonevent distributions. The difference of means is also the d' or Δm statistic (with unit standard error) that is discussed in the signal detection literature (McNicol 2005).

The Hosmer-Lemeshow Goodness-of-Fit Test

To evaluate the fit of the model, Hosmer and Lemeshow (2000) proposed a statistic that they show, through simulation, is distributed as chi-square when there is no replication in any of the subpopulations. This goodness-of-fit test is available only for binary response models.

The unit interval is partitioned into 2,000 equal-sized bins, and each observation i is placed into the bin that contains its estimated event probability. This effectively sorts the observations in increasing order of their estimated event probability.

The observations (and frequencies) are further combined into G groups. By default G = 10, but you can specify $G \geq 5$ with the NGROUPS= suboption of the LACKFIT option in the MODEL statement. Let F be the total frequency. The target frequency for each group is $T = \lfloor F/G + 0.5 \rfloor$, which is the integer part of $F/G + 0.5$. Load the first group (g_j, $j = 1$) with the first of the 2,000 bins that has nonzero frequency f_1, and let the next nonzero bin have a frequency of f. PROC HPLOGISTIC performs the following steps for each nonzero bin to create the groups:

1. If $j = G$, then add this bin to group g_j.
2. Otherwise, if $f_j < T$ and $f_j + \lfloor f/2 \rfloor \leq T$, then add this bin to group g_j.
3. Otherwise, start loading the next group (g_{j+1}) with $f_{j+1} = f$, and set $j = j + 1$.

If the final group g_j has frequency $f_j < T/2$, then add these observations to the preceding group. The total number of groups actually created, g, can be less than G.

The Hosmer-Lemeshow goodness-of-fit statistic is obtained by calculating the Pearson chi-square statistic from the $2 \times g$ table of observed and expected frequencies. The statistic is written

$$\chi^2_{HL} = \sum_{j=1}^{g} \frac{(O_j - F_j\bar{\pi}_j)^2}{F_j\bar{\pi}_j(1-\bar{\pi}_j)}$$

where, for the jth group g_j, $F_j = \sum_{i \in g_j} f_i$ is the total frequency of subjects, O_j is the total frequency of event outcomes, and $\bar{\pi}_j = \sum_{i \in g_j} f_i \widehat{p}_i / F_j$ is the average estimated predicted probability of an event outcome. Let ϵ be the square root of the machine epsilon divided by 4,000, which is about 2.5E–12. Any $\bar{\pi}_j < \epsilon$ is set to ϵ; similarly, any $\bar{\pi}_j > 1 - \epsilon$ is set to $1 - \epsilon$.

The Hosmer-Lemeshow statistic is compared to a chi-square distribution with $g - r$ degrees of freedom. You can specify r with the DFREDUCE= suboption of the LACKFIT option in the MODEL statement. By default, $r = 2$, and to compute the Hosmer-Lemeshow statistic you must have $g - r \geq 1$. Large values of χ^2_{HL} (and small p-values) indicate a lack of fit of the model.

Computational Method: Multithreading

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to sequential (single-threaded) execution.

The number of threads spawned by the HPLOGISTIC procedure is determined by the number of CPUs on a machine and can be controlled by specifying the NTHREADS= option in the PERFORMANCE statement. This specification overrides the system option. Specify NTHREADS=1 to force single-threaded execution. The number of threads per machine is displayed in the "Dimensions" table, which is part of the default output. The HPLOGISTIC procedure allocates one thread per CPU by default.

The tasks that are multithreaded by the HPLOGISTIC procedure are primarily defined by dividing the data processed on a single machine among the threads—that is, the HPLOGISTIC procedure implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and you are running with four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following:

- variable levelization
- effect levelization
- formation of the initial crossproducts matrix
- formation of approximate Hessian matrices for candidate evaluation during model selection
- objective function calculation
- gradient calculation
- Hessian calculation
- scoring of observations
- summarization of data for the Hosmer-Lemeshow test and association statistics

In addition, operations on matrices such as sweeps can be multithreaded provided that the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Choosing an Optimization Algorithm

First- or Second-Order Algorithms

The factors that go into choosing a particular optimization technique for a particular problem are complex. Trial and error can be involved.

For many optimization problems, computing the gradient takes more computer time than computing the function value. Computing the Hessian sometimes takes *much* more computer time and memory than computing the gradient, especially when there are many decision variables. Unfortunately, optimization techniques that do not use some kind of Hessian approximation usually require many more iterations than techniques that do use a Hessian matrix, and, as a result the total run time of these techniques is often longer. Techniques that do not use the Hessian also tend to be less reliable. For example, they can terminate more easily at stationary points than at global optima.

Table 61.10 shows which derivatives are required for each optimization technique.

Table 61.10 Derivatives Required

Algorithm	First-Order	Second-Order
TRUREG	x	x
NEWRAP	x	x
NRRIDG	x	x
QUANEW	x	-
DBLDOG	x	-
CONGRA	x	-
NMSIMP	-	-

The second-derivative methods TRUREG, NEWRAP, and NRRIDG are best for small problems for which the Hessian matrix is not expensive to compute. Sometimes the NRRIDG algorithm can be faster than the TRUREG algorithm, but TRUREG can be more stable. The NRRIDG algorithm requires only one matrix with $p(p+1)/2$ double words; TRUREG and NEWRAP require two such matrices. Here, p denotes the number of parameters in the optimization.

The first-derivative methods QUANEW and DBLDOG are best for medium-sized problems for which the objective function and the gradient can be evaluated much faster than the Hessian. In general, the QUANEW and DBLDOG algorithms require more iterations than TRUREG, NRRIDG, and NEWRAP, but each iteration can be much faster. The QUANEW and DBLDOG algorithms require only the gradient to update an approximate Hessian, and they require slightly less memory than TRUREG or NEWRAP.

The first-derivative method CONGRA is best for large problems for which the objective function and the gradient can be computed much faster than the Hessian and for which too much memory is required to store the (approximate) Hessian. In general, the CONGRA algorithm requires more iterations than QUANEW or DBLDOG, but each iteration can be much faster. Because CONGRA requires only a factor of p double-word memory, many large applications can be solved only by CONGRA.

The no-derivative method NMSIMP is best for small problems for which derivatives are not continuous or are very difficult to compute.

Each optimization method uses one or more convergence criteria that determine when it has converged. An algorithm is considered to have converged when any one of the convergence criteria is satisfied. For example, under the default settings, the QUANEW algorithm converges if ABSGCONV $<$ 1E–5, FCONV $< 2 \times \epsilon$, or GCONV $<$ 1E–8.

By default, the HPLOGISTIC procedure applies the NRRIDG algorithm because it can take advantage of multithreading in Hessian computations and inversions. If the number of parameters becomes large, specifying the TECHNIQUE=QUANEW option, which is a first-order method with good overall properties, is recommended.

Algorithm Descriptions

The following subsections provide details about each optimization technique and follow the same order as Table 61.10.

Trust Region Optimization (TRUREG)

The trust region method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function $f(\boldsymbol{\psi})$ have continuous first- and second-order derivatives inside the feasible region.

The trust region method iteratively optimizes a quadratic approximation to the nonlinear objective function within a hyperelliptic trust region with radius Δ that constrains the step size that corresponds to the quality of the quadratic approximation. The trust region method is implemented based on Dennis, Gay, and Welsch (1981), Gay (1983), and Moré and Sorensen (1983).

The trust region method performs well for small- to medium-sized problems, and it does not need many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the dual quasi-Newton or conjugate gradient algorithms might be more efficient.

Newton-Raphson Optimization with Line Search (NEWRAP)

The NEWRAP technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region. If second-order derivatives are computed efficiently and precisely, the NEWRAP method can perform well for medium-sized to large problems, and it does not need many function, gradient, and Hessian calls.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. Otherwise, a combination of ridging and line search is performed to compute successful steps. If the Hessian is not positive-definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive-definite (Eskow and Schnabel 1991).

In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The line-search method uses quadratic interpolation and cubic extrapolation.

Newton-Raphson Ridge Optimization (NRRIDG)

The NRRIDG technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. If at least one of these two conditions is not satisfied, a multiple of the identity matrix is added to the Hessian matrix.

Because the NRRIDG technique uses an orthogonal decomposition of the approximate Hessian, each iteration of NRRIDG can be slower than that of the NEWRAP technique, which works with a Cholesky decomposition. However, NRRIDG usually requires fewer iterations than NEWRAP.

The NRRIDG method performs well for small- to medium-sized problems, and it does not require many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the dual quasi-Newton or conjugate gradient algorithms might be more efficient.

Quasi-Newton Optimization (QUANEW)

The dual quasi-Newton method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$, and it does not need to compute second-order derivatives because they are approximated. It works well for medium-sized to moderately large optimization

problems, where the objective function and the gradient can be computed much faster than the Hessian. However, in general the QUANEW technique requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which compute second-order derivatives. The QUANEW technique provides an appropriate balance between the speed and stability required for most nonlinear mixed model applications.

The QUANEW technique implemented by the HPLOGISTIC procedure is the dual quasi-Newton algorithm, which updates the Cholesky factor of an approximate Hessian.

In each iteration, a line search is performed along the search direction to find an approximate optimum. The line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions (Fletcher 1987). One of the Goldstein conditions can be violated if the feasible region defines an upper limit of the step size. Violating the left-side Goldstein condition can affect the positive-definiteness of the quasi-Newton update. In that case, either the update is skipped or the iterations are restarted with an identity matrix, resulting in the steepest descent or ascent search direction.

Double-Dogleg Optimization (DBLDOG)

The double-dogleg optimization method combines the ideas of the quasi-Newton and trust region methods. In each iteration, the double-dogleg algorithm computes the step $\mathrm{s}^{(k)}$ as the linear combination of the steepest descent or ascent search direction $\mathrm{s}_1^{(k)}$ and a quasi-Newton search direction $\mathrm{s}_2^{(k)}$:

$$\mathrm{s}^{(k)} = \alpha_1 \mathrm{s}_1^{(k)} + \alpha_2 \mathrm{s}_2^{(k)}$$

The step is requested to remain within a prespecified trust region radius (Fletcher 1987, p. 107). Thus, the DBLDOG subroutine uses the dual quasi-Newton update but does not perform a line search.

The double-dogleg optimization technique works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. The implementation is based on Dennis and Mei (1979) and Gay (1983), but it is extended for dealing with boundary and linear constraints. The DBLDOG technique generally requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which require second-order derivatives; however, each of the DBLDOG iterations is computationally cheap. Furthermore, the DBLDOG technique requires only gradient calls for the update of the Cholesky factor of an approximate Hessian.

Conjugate Gradient Optimization (CONGRA)

Second-order derivatives are not required by the CONGRA algorithm and are not even approximated. The CONGRA algorithm can be expensive in function and gradient calls, but it requires only $O(p)$ memory for unconstrained optimization. In general, many iterations are required to obtain a precise solution, but each of the CONGRA iterations is computationally cheap.

The CONGRA subroutine should be used for optimization problems with large p. For the unconstrained or boundary-constrained case, CONGRA requires only $O(p)$ bytes of working memory, whereas all other optimization methods require order $O(p^2)$ bytes of working memory. During p successive iterations, uninterrupted by restarts or changes in the working set, the conjugate gradient algorithm computes a cycle of p conjugate search directions. In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit for the step size.

Nelder-Mead Simplex Optimization (NMSIMP)

The Nelder-Mead simplex method does not use any derivatives and does not assume that the objective function has continuous derivatives. The objective function itself needs to be continuous. This technique is quite expensive in the number of function calls, and it might be unable to generate precise results for $p \gg 40$.

The original Nelder-Mead simplex algorithm is implemented and extended to boundary constraints. This algorithm does not compute the objective for infeasible points, but it changes the shape of the simplex adapting to the nonlinearities of the objective function. This change contributes to an increased speed of convergence and uses a special termination criterion.

Displayed Output

The following sections describe the output that PROC HPLOGISTIC produces. The output is organized into various tables, which are discussed in the order of appearance.

Performance Information

The "Performance Information" table is produced by default, and displays the number of threads used for single-machine mode.

If you specify the DETAILS option in the PERFORMANCE statement, the procedure also produces a "Timing" table in which elapsed time (absolute and relative) for the main tasks of the procedure are displayed.

Model Information

The "Model Information" table displays basic information about the model, such as the response variable, frequency variable, link function, and the model category the HPLOGISTIC procedure determined based on your input and options. The "Model Information" table also displays the distribution of the data that is assumed by the HPLOGISTIC procedure. See the section "Response Distributions" on page 4871 for how the procedure determines the response distribution.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels with the ORDER= option in the CLASS statement. You can suppress the "Class Level Information" table completely or partially with the NOCLPRINT= option in the PROC HPLOGISTIC statement.

If the classification variables use reference parameterization, the "Class Level Information" table also displays the reference value for each variable.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set and the number of observations used in the analysis. If a FREQ statement is present, the sum of the frequencies read and used is displayed. If the events/trials syntax is used, the number of events and trials is also displayed. If you specify a PARTITION statement, the table displays the values for each role.

Response Profile

The "Response Profile" table displays the ordered value from which the HPLOGISTIC procedure determines the probability being modeled as an event in binary models and the ordering of categories in multinomial models. For each response category level, the frequency used in the analysis is reported. You can affect the ordering of the response values with the *response-options* in the MODEL statement. For binary and generalized logit models, the note that follows the "Response Profile" table indicates which outcome is modeled as the event in binary models and which value serves as the reference category.

The "Response Profile" table is not produced for binomial data. You can find information about the number of events and trials in the "Number of Observations" table. If you specify a PARTITION statement, the table displays the values for each role.

Selection Information

When you specify the SELECTION statement, the HPLOGISTIC procedure produces by default a series of tables with information about the model selection. The "Selection Information" table informs you about the model selection method, selection and stop criteria, and other parameters that govern the selection. You can suppress this table by specifying DETAILS=NONE in the SELECTION statement.

Selection Summary

When you specify the SELECTION statement, the HPLOGISTIC procedure produces the "Selection Summary" table with information about which effects were entered into or removed from the model at the steps of the model selection process. The statistic that led to the removal or entry decision is also displayed. You can request further details about the model selection steps by specifying DETAILS=STEPS or DETAILS=ALL in the SELECTION statement. You can suppress the display of the "Selection Summary" table by specifying DETAILS=NONE in the SELECTION statement.

Stop Reason

When you specify the SELECTION statement, the HPLOGISTIC procedure produces a simple table that tells you why model selection stopped.

Selection Reason

When you specify the SELECTION statement, the HPLOGISTIC procedure produces a simple table that tells you why the final model was selected.

Selected Effects

When you specify the SELECTION statement, the HPLOGISTIC procedure produces a simple table that tells you which effects were selected into the final model.

Candidate Entry and Removal Details

When you specify the DETAILS=ALL or DETAILS=STEPS option in the SELECTION statement, the HPLOGISTIC procedure produces the "Candidate Entry and Removal Details" table, which displays the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. For each step, the effects are displayed in sorted order from best to worst of the selection criterion.

Selection Details

When you specify the DETAILS=ALL option in the SELECTION statement, the HPLOGISTIC procedure produces the "Selection Details" table, which contains information about which effects were entered into or removed from the model at the steps of the model selection process. If you specify SELECT=AIC, AICC, or BIC then the likelihood ratio chi-square statistic is displayed along with the estimated selection criteria; otherwise the score or Wald chi-square statistic is displayed. Fit statistics computed at each step are also displayed.

Iteration History

For each iteration of the optimization, the "Iteration History" table displays the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration and the absolute value of the largest (projected) gradient element. The objective function used in the optimization in the HPLOGISTIC procedure is normalized by default to enable comparisons across data sets with different sampling intensity. You can control normalization with the NORMALIZE= option in the PROC HPLOGISTIC statement.

If you specify the ITDETAILS option in the PROC HPLOGISTIC statement, information about the parameter estimates and gradients in the course of the optimization is added to the "Iteration History" table.

The "Iteration History" table is displayed by default unless you specify the NOITPRINT option or perform a model selection. To generate the history from a model selection process, specify the ITSELECT or ITDETAILS option.

Convergence Status

The convergence status table is a small ODS table that follows the "Iteration History" table in the default output. In the listing it appears as a message that indicates whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the convergence status table to an output data set, a numeric Status variable is added that enables you to assess convergence programmatically. The values of the Status variable encode the following:

0 Convergence was achieved, or an optimization was not performed (because TECHNIQUE=NONE is specified).

1 The objective function could not be improved.

2 Convergence was not achieved because of a user interrupt or because a limit was exceeded, such as the maximum number of iterations or the maximum number of function evaluations. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC HPLOGISTIC statement.

3 Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Dimensions

The "Dimensions" table displays size measures that are derived from the model and the environment. For example, it displays the number of columns in the design matrix, the rank of the matrix, the largest number of design columns associated with an effect, and the number of threads.

Fit Statistics

The "Fit Statistics" table displays a variety of likelihood-based measures of fit. All statistics are presented in "smaller is better" form. If you specify a PARTITION statement, the table displays the values for each role. The values displayed in the "Fit Statistics" table are not based on a normalized log-likelihood function. For more information, see the section "Information Criteria" on page 4876.

Partition Fit Statistics

If you specify the PARTITION statement, the "Partition Fit Statistics" table displays statistics for comparing the training, validation, and testing results. For more information about the statistics displayed in this table, see the sections "Partition Fit Statistics" on page 4875, "Model Fit and Assessment Statistics" on page 4876, and "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880.

Global Tests

The "Global Tests" table provides a statistical test for the hypothesis of whether the final model provides a better fit than a model without effects (an "intercept-only" model).

If you specify the NOINT option in the MODEL statement, the reference model is one where the linear predictor is 0 for all observations.

Partition for the Hosmer and Lemeshow Test

If you specify the LACKFIT option in the MODEL statement, the "Partition for the Hosmer and Lemeshow Test" table displays the grouping used in the Hosmer-Lemeshow test. If you specify a PARTITION statement, a table is displayed for each role. For more information, see the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880. For examples of using this partition, see Hosmer and Lemeshow (2000).

Hosmer and Lemeshow Goodness-of-Fit Test

If you specify the LACKFIT option in the MODEL statement, the "Hosmer and Lemeshow Goodness-of-Fit Test" table provides a test of the fit of the model; small p-values reject the null hypothesis that the fitted model is adequate. If you specify a PARTITION statement, a row is displayed for each role. For more information, see the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880.

Association Statistics

If you specify the ASSOCIATION option in the MODEL statement, the "Association Statistics" table displays the concordance index c (the area under the ROC curve, AUC), Somers' D statistic (Gini's coefficient), Goodman-Kruskal's gamma statistic, and Kendall's tau-a statistic. If you also specify a PARTITION statement, a row is displayed for each role. For more information, see the section "Association Statistics" on page 4879.

Classification Table

The "Classification" table is displayed if you specify the CTABLE option without specifying an output data set. If you also specify a PARTITION statement, a table is displayed for each role. For more information, see the section "Classification Table and ROC Curves" on page 4877.

Parameter Estimates

The parameter estimates, their estimated (asymptotic) standard errors, and p-values for the hypothesis that the parameter is 0 are presented in the "Parameter Estimates" table. If you request confidence intervals by using the CL or ALPHA= option in the MODEL statement, confidence limits are produced for the estimate on the linear scale.

By default, a normal z statistic is used to test the parameter estimates and is displayed in the "t Value" column with DF='Infty'. The square of the z statistic is a chi-square, so these p-values are identical to those from a Wald chi-square test. You can specify the DDFM=RESIDUAL option in the MODEL statement to obtain small-sample t tests.

ODS Table Names

Each table created by the HPLOGISTIC procedure has a name associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 61.11.

Table 61.11 ODS Tables Produced by PROC HPLOGISTIC

Table Name	Description	Statement	Option
Association	Association of predicted probabilities and observed responses	MODEL	ASSOCIATION
CandidateDetails	Details about candidates for entry into or removal from the model	SELECTION	DETAILS=STEP
Classification	Classification table	MODEL	CTABLE
ClassLevels	Level information from the CLASS statement	CLASS	Default
ConvergenceStatus	Status of optimization at conclusion of optimization	PROC HPLOGISTIC	Default
DataAccessInfo	Information about modes of data access	PROC HPLOGISTIC	Default
Dimensions	Model dimensions	PROC HPLOGISTIC	Default
FitStatistics	Fit statistics	PROC HPLOGISTIC	Default
GlobalTests	Test of the model versus the null model	PROC HPLOGISTIC	Default
IterHistory	Iteration history	PROC HPLOGISTIC	Default or ITSELECT
LackFitChiSq	Hosmer-Lemeshow chi-square test results	MODEL	LACKFIT

Table 61.11 *continued*

Table Name	Description	Statement	Option
LackFitPartition	Partition for the Hosmer-Lemeshow test	MODEL	LACKFIT
ModelInfo	Information about the modeling environment	PROC HPLOGISTIC	Default
NObs	Number of observations read and used, and number of events and trials, if applicable	PROC HPLOGISTIC	Default
ParameterEstimates	Solutions for the parameter estimates associated with effects in MODEL statements	PROC HPLOGISTIC	Default
PartFitStats	Fit statistics for the data roles	PARTITION statement	Default
PerformanceInfo	Information about the high-performance computing environment	PROC HPLOGISTIC	Default
ResponseProfile	Response categories and category modeled in models for binary and multinomial data	PROC HPLOGISTIC	Default
SelectedEffects	List of effects selected into model	SELECTION	Default
SelectionDetails	Details about model selection, including fit statistics by step	SELECTION	DETAILS=ALL
SelectionInfo	Information about the settings for model selection	SELECTION	Default
SelectionReason	Reason why the particular model was selected	SELECTION	Default
SelectionSummary	Summary information about model selection steps	SELECTION	Default
StopReason	Reason for termination of model selection	SELECTION	Default
Timing	Absolute and relative times for tasks performed by the procedure	PERFORMANCE	DETAILS

Examples: HPLOGISTIC Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 61.1: Model Selection

(View the complete code for this example (hplogex1.sas) in the example repository.)

The following HPLOGISTIC statements examine the same data as in the section "Getting Started: HPLOGISTIC Procedure" on page 4845, but they request model selection via the forward selection technique. Model effects are added in the order of their significance until no more effects make a significant improvement of the current model. The DETAILS=ALL option in the SELECTION statement requests that all tables related to model selection be produced.

```
proc hplogistic data=getStarted;
   class C;
   model y = C x1-x10;
   selection method=forward details=all;
run;
```

The model selection tables are shown in Output 61.1.1 through Output 61.1.4.

The "Selection Information" table in Output 61.1.1 summarizes the settings for the model selection. Effects are added to the model only if they produce a significant improvement as judged by comparing the p-value of a score test to the entry significance level (SLE), which is 0.05 by default. The forward selection stops when no effect outside the model meets this criterion.

Output 61.1.1 Selection Information

The HPLOGISTIC Procedure

Selection Information	
Selection Method	Forward
Select Criterion	Significance Level
Stop Criterion	Significance Level
Effect Hierarchy Enforced	None
Entry Significance Level (SLE)	0.05
Stop Horizon	1

The "Selection Summary" table in Output 61.1.2 shows the effects that were added to the model and their significance level. Step 0 refers to the null model that contains only an intercept. In the next step, effect x8 made the most significant contribution to the model among the candidate effects ($p = 0.0381$). In step 2 the most significant contribution when adding an effect to a model that contains the intercept and x8 was made by x2. In the subsequent step no effect could be added to the model that would produce a p-value less than 0.05, so variable selection stops.

Output 61.1.2 Selection Summary Information

Selection Summary			
Step	Effect Entered	Number Effects In	p Value
0	Intercept	1	.
1	x8	2	0.0381
2	x2	3	0.0255

Selection stopped because no candidate for entry is significant at the 0.05 level.

Selected Effects: Intercept x2 x8

The DETAILS=ALL option requests further detail information about the steps of the model selection. The "Candidate Details" table in Output 61.1.3 list all candidates for each step in the order of significance of their score tests. The effect with smallest p-value less than the SLE level of 0.05 is added in each step.

Output 61.1.3 Candidate Details

Candidate Entry and Removal Details				
Step	Rank	Effect	Candidate For	p Value
1	1	x8	Entry	0.0381
	2	x2	Entry	0.0458
	3	x4	Entry	0.0557
	4	x9	Entry	0.1631
	5	C	Entry	0.1858
	6	x1	Entry	0.2715
	7	x10	Entry	0.4434
	8	x5	Entry	0.7666
	9	x3	Entry	0.8006
	10	x7	Entry	0.8663
	11	x6	Entry	0.9626
2	1	x2	Entry	0.0255
	2	x4	Entry	0.0721
	3	x9	Entry	0.1080
	4	C	Entry	0.1241
	5	x1	Entry	0.2778
	6	x10	Entry	0.5250
	7	x5	Entry	0.6993
	8	x7	Entry	0.7103
	9	x3	Entry	0.8743
	10	x6	Entry	0.9577

The DETAILS=ALL option also produces the "Selection Details" table, which provides fit statistics and the value of the score test chi-square statistic at each step.

Output 61.1.4 Selection Details

Selection Details								
Step	Effect Entered	Number Effects In	Chi-Square	Pr > ChiSq	-2 LogL	AIC	AICC	BIC
0	Initial Model	1			123.82	125.82	125.86	128.43
1	x8	2	4.2986	0.0381	119.46	123.46	123.59	128.67
2	x2	3	4.9882	0.0255	114.40	120.40	120.65	128.21

Output 61.1.5 displays information about the selected model. Notice that the –2 log likelihood value in the "Fit Statistics" table is larger than the value for the full model in Figure 61.9. This is expected because the selected model contains only a subset of the parameters. Because the selected model is more parsimonious than the full model, the discrepancy between the –2 log likelihood and the information criteria is less severe than previously noted.

Output 61.1.5 Fit Statistics and Null Test

Fit Statistics	
-2 Log Likelihood	114.40
AIC (smaller is better)	120.40
AICC (smaller is better)	120.65
BIC (smaller is better)	128.21

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	9.4237	2	0.0090

The parameter estimates of the selected model are given in Output 61.1.6. Notice that the effects are listed in the "Parameter Estimates" table in the order in which they were specified in the MODEL statement and not in the order in which they were added to the model.

Output 61.1.6 Parameter Estimates

Parameter Estimates					
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	0.8584	0.5503	Infty	1.56	0.1188
x2	-0.2502	0.1146	Infty	-2.18	0.0290
x8	1.7840	0.7908	Infty	2.26	0.0241

You can construct the prediction equation for this model from the parameter estimates as follows. The estimated linear predictor for an observation is

$$\widehat{\eta} = 0.8584 - 0.2503 \times x_2 + 1.7840 \times x_8$$

and the predicted probability that variable y takes on the value 0 is

$$\widehat{\Pr}(Y = 0) = \frac{1}{1 + \exp\{-\widehat{\eta}\}}$$

Example 61.2: Modeling Binomial Data

(View the complete code for this example (hplogex2.sas) in the example repository.)

If $Y_1, \cdots, Y_n$ are independent binary (Bernoulli) random variables with common success probability π, then their sum is a binomial random variable. In other words, a binomial random variable with parameters n and π can be generated as the sum of n Bernoulli(π) random experiments. The HPLOGISTIC procedure uses a special syntax to express data in binomial form, the *events/trials* syntax.

Consider the following data, taken from Cox and Snell (1989, pp. 10–11), of the number, r, of ingots not ready for rolling, out of n tested, for a number of combinations of heating time and soaking time. If each test is carried out independently and if for a particular combination of heating and soaking time there is a constant probability that the tested ingot is not ready for rolling, then the random variable r follows a Binomial(n, π) distribution, where the success probability π is a function of heating and soaking time.

```
data Ingots;
   input Heat Soak r n @@;
   Obsnum= _n_;
   datalines;
7 1.0 0 10  14 1.0 0 31  27 1.0 1 56  51 1.0 3 13
7 1.7 0 17  14 1.7 0 43  27 1.7 4 44  51 1.7 0  1
7 2.2 0  7  14 2.2 2 33  27 2.2 0 21  51 2.2 0  1
7 2.8 0 12  14 2.8 0 31  27 2.8 1 22  51 4.0 0  1
7 4.0 0  9  14 4.0 0 19  27 4.0 1 16
;
```

The following statements show the use of the events/trials syntax to model the binomial response. The *events* variable in this situation is r, the number of ingots not ready for rolling, and the *trials* variable is n, the number of ingots tested. The dependency of the probability of not being ready for rolling is modeled as a function of heating time, soaking time, and their interaction. The ASSOCIATION option displays ordinal measures of association between the observed responses and predicted probabilities. The CTABLE= option stores statistics that are used for evaluating the predictive power of the model in the Roc data set. The LACKFIT option produces the Hosmer and Lemeshow goodness-of-fit test for binary response models. The OUTPUT statement stores the linear predictors and the predicted probabilities in the Out data set along with the ID variable.

```
proc hplogistic data=Ingots;
   model r/n = Heat Soak Heat*Soak / association ctable=Roc lackfit;
   id Obsnum;
   output out=Out xbeta predicted=Pred;
run;
```

The "Performance Information" table in Output 61.2.1 shows that the procedure executes in single-machine mode. The example is executed on a single machine with the same number of cores as the number of threads used; that is, one computational thread was spawned per CPU.

Output 61.2.1 Performance Information

The HPLOGISTIC Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

The "Model Information" table shows that the data are modeled as binomially distributed with a logit link function (Output 61.2.2). This is the default link function in the HPLOGISTIC procedure for binary and binomial data. The procedure estimates the parameters of the model by a Newton-Raphson algorithm.

Output 61.2.2 Model Information and Number of Observations

Model Information	
Data Source	WORK.INGOTS
Response Variable (Events)	r
Response Variable (Trials)	n
Distribution	Binomial
Link Function	Logit
Optimization Technique	Newton-Raphson with Ridging

Number of Observations Read	19
Number of Observations Used	19
Number of Events	12
Number of Trials	387

The second table in Output 61.2.2 shows that all 19 observations in the data set were used in the analysis, and that the total number of events and trials equal 12 and 387, respectively. These are the sums of the variables r and n across all observations.

Output 61.2.3 displays the "Iteration History" and convergence status tables for this run. The HPLOGISTIC procedure converged after four iterations (not counting the initial setup iteration) and meets the GCONV= convergence criterion.

Output 61.2.3 Iteration History and Convergence Status

Iteration History				
Iteration	**Evaluations**	**Objective Function**	**Change**	**Max Gradient**
0	4	0.7676329445	.	6.378002
1	2	0.7365832479	0.03104970	0.754902
2	2	0.7357086248	0.00087462	0.023623
3	2	0.7357075299	0.00000109	0.00003
4	2	0.7357075299	0.00000000	5.42E-11

Convergence criterion (GCONV=1E-8) satisfied.

Output 61.2.4 displays the "Dimensions" table for the model. There are four columns in the design matrix of the model (the **X** matrix); they correspond to the intercept, the Heat effect, the Soak effect, and the interaction of the Heat and Soak effects. The model is nonsingular, since the rank of the crossproducts matrix equals the number of columns in **X**. All parameters are estimable and participate in the optimization.

Output 61.2.4 Dimensions in Binomial Logistic Regression

Dimensions	
Columns in X	4
Number of Effects	4
Max Effect Columns	1
Rank of Cross-product Matrix	4
Parameters in Optimization	4

Output 61.2.5 displays the "Fit Statistics" table for this run. Evaluated at the converged estimates, –2 times the value of the log-likelihood function equals 27.9569. Further fit statistics are also given, all of them in "smaller is better" form. The AIC, AICC, and BIC criteria are used to compare non-nested models and to penalize the model fit for the number of observations and parameters. The –2 log-likelihood value can be used to compare nested models by way of a likelihood ratio test.

Output 61.2.5 Fit Statistics

Fit Statistics	
-2 Log Likelihood	27.9569
AIC (smaller is better)	35.9569
AICC (smaller is better)	38.8140
BIC (smaller is better)	39.7346

Output 61.2.6 shows the test of the global hypothesis that the effects jointly do not impact the probability of ingot readiness. The chi-square test statistic can be obtained by comparing the –2 log-likelihood value of the model with covariates to the value in the intercept-only model. The test is significant with a p-value of 0.0082. One or more of the effects in the model have a significant impact on the probability of ingot readiness.

Output 61.2.6 Null Test

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	11.7663	3	0.0082

Output 61.2.7 shows the tables that are produced when you specify the LACKFIT option in the MODEL statement. The first table displays the partition that PROC HPLOGISTIC uses to compute the chi-square test that is displayed in the second table; the large p-value does not reject the adequacy of the model fit. For more information, see the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880.

Output 61.2.7 Hosmer-Lemeshow Goodness-of-Fit Test

Partition for the Hosmer and Lemeshow Test					
	Events		Nonevents		
Group	Observed	Expected	Observed	Expected	Total
1	0	0.24	34	33.76	34
2	0	0.47	43	42.53	43
3	0	0.66	52	51.34	52
4	2	0.46	31	32.54	33
5	0	0.48	31	30.52	31
6	0	0.36	19	18.64	19
7	1	1.94	55	54.06	56
8	4	1.59	40	42.41	44
9	1	1.63	42	41.37	43
10	4	4.17	28	27.83	32

Hosmer and Lemeshow Goodness-of-Fit Test		
Chi-Square	DF	Pr > ChiSq
11.9771	8	0.1522

Output 61.2.8 displays the "Association Statistics" table, which is produced when you specify the ASSOCIATION option in the MODEL statement. The table contains four measures of association for assessing the predictive ability of a model. For more information, see the section "Association Statistics" on page 4879.

Output 61.2.8 Association of Observed Responses and Predicted Probabilities

Association Statistics			
Concordance Index	Somers' D	Gamma	Tau-a
0.770556	0.541111	0.585759	0.032601

The "Parameter Estimates" table in Output 61.2.9 displays the estimates and standard errors of the model effects.

Output 61.2.9 Parameter Estimates

Parameter Estimates					
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	-5.9902	1.6666	Infty	-3.59	0.0003
Heat	0.09634	0.04707	Infty	2.05	0.0407
Soak	0.2996	0.7551	Infty	0.40	0.6916
Heat*Soak	-0.00884	0.02532	Infty	-0.35	0.7270

You can construct the prediction equation of the model from the "Parameter Estimates" table. For example, an observation with Heat equal to 14 and Soak equal to 1.7 has linear predictor

$$\widehat{\eta} = -5.9902 + 0.09634 \times 14 + 0.2996 \times 1.7 - 0.00884 \times 14 \times 7 = -4.34256$$

The probability that an ingot with these characteristics is not ready for rolling is

$$\widehat{\pi} = \frac{1}{1 + \exp\{-(-4.34256)\}} = 0.01284$$

The OUTPUT statement computes these linear predictors and probabilities and stores them in the Out data set. This data set also contains the ID variable, which is used by the following statements to attach the covariates to these statistics. Output 61.2.10 shows the probability that an ingot with Heat equal to 14 and Soak equal to 1.7 is not ready for rolling.

```
data Out;
   merge Out Ingots;
   by Obsnum;
proc print data=Out;
   where Heat=14 & Soak=1.7;
run;
```

Output 61.2.10 Predicted Probability for Heat=14 and Soak=1.7

Obs	Obsnum	Pred	Xbeta	Heat	Soak	r	n
6	6	0.012836	-4.34256	14	1.7	0	43

The CTABLE=ROC option computes statistics for binary response models (based on classifying observations according to whether their predicted probabilities exceed certain values) and stores the results in the Roc data set. For more information, see the section "Classification Table and ROC Curves" on page 4877. You can use this data set to display the ROC curve by using the SGPLOT procedure as follows:

```
proc sgplot data=Roc aspect=1 noautolegend;
   title 'ROC Curve';
   xaxis values=(0 to 1 by 0.25) grid offsetmin=.05 offsetmax=.05;
   yaxis values=(0 to 1 by 0.25) grid offsetmin=.05 offsetmax=.05;
   lineparm x=0 y=0 slope=1 / lineattrs=(color=ligr);
   series x=FPF y=TPF;
   inset 'Area under the curve=0.7706' / position=bottomright;
run;
```

Output 61.2.11 Receiver Operating Characteristics Curve

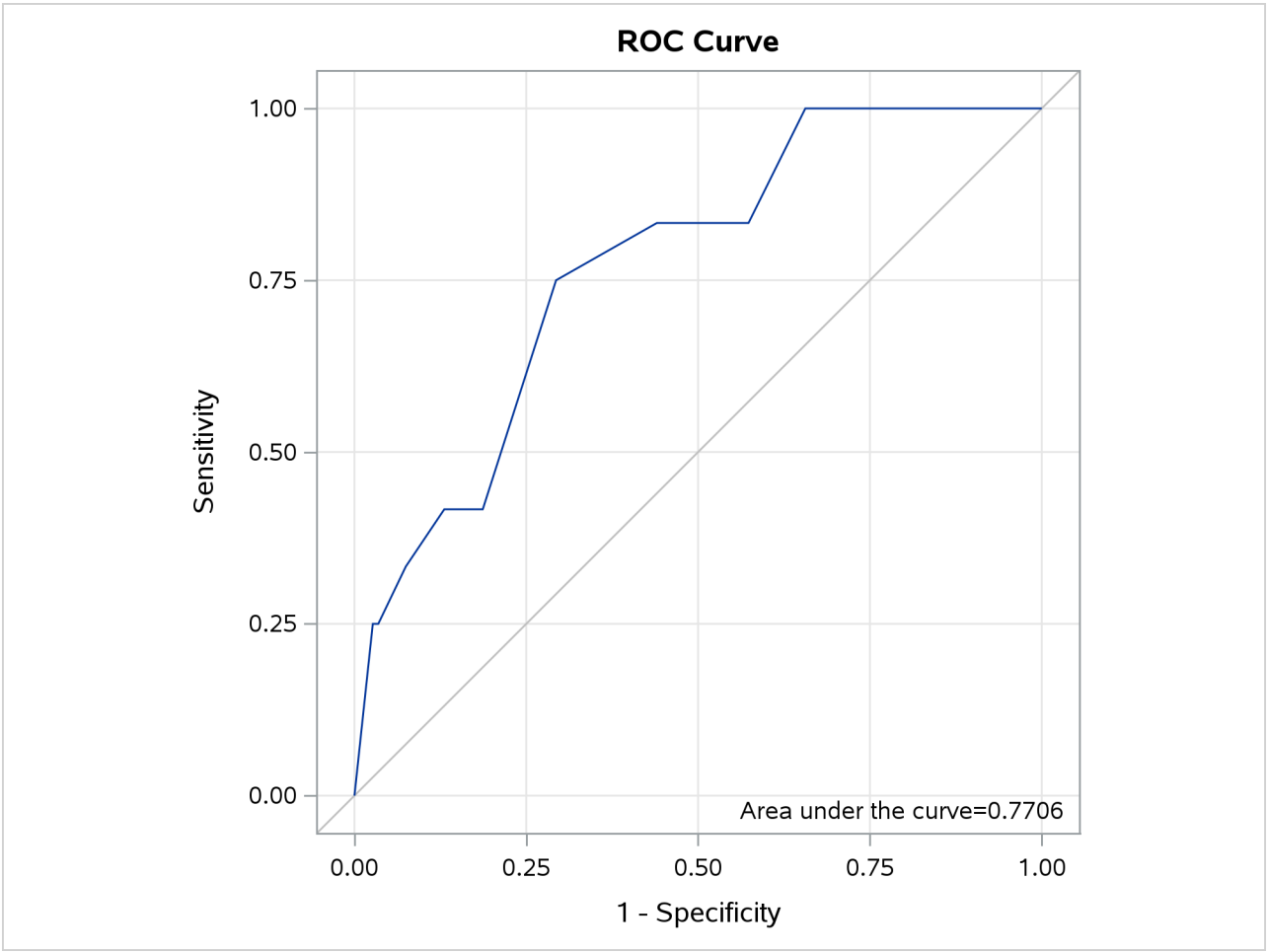

Binomial data are a form of grouped binary data where "successes" in the underlying Bernoulli trials are totaled. You can thus unwind data for which you use the events/trials syntax and fit it with techniques for binary data.

The following DATA step expands the Ingots data set with 12 events in 387 trials into a binary data set with 387 observations.

```
data Ingots_binary;
   set Ingots;
   do i=1 to n;
     if i <= r then y=1; else y = 0;
     output;
   end;
run;
```

The following HPLOGISTIC statements fit the model with Heat effect, Soak effect, and their interaction to the binary data set. The **event='1'** response-variable option in the MODEL statement ensures that the HPLOGISTIC procedure models the probability that the variable y takes on the value ('1').

```
proc hplogistic data=Ingots_binary;
   model y(event='1') = Heat Soak Heat*Soak;
run;
```

Output 61.2.12 displays the "Performance Information", "Model Information," "Number of Observations," and the "Response Profile" tables. The data are now modeled as binary (Bernoulli distributed) with a logit link function. The "Response Profile" table shows that the binary response breaks down into 375 observations where y equals zero and 12 observations where y equals 1.

Output 61.2.12 Model Information in Binary Model

The HPLOGISTIC Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.INGOTS_BINARY	V9	Input	On Client

Model Information	
Data Source	WORK.INGOTS_BINARY
Response Variable	y
Distribution	Binary
Link Function	Logit
Optimization Technique	Newton-Raphson with Ridging

Number of Observations Read	387
Number of Observations Used	387

Response Profile		
Ordered Value	**y**	**Total Frequency**
1	0	375
2	1	12

You are modeling the probability that y='1'.

Output 61.2.13 displays the result for the test of the global null hypothesis and the parameter estimates. These results match those in Output 61.2.6 and Output 61.2.9.

Output 61.2.13 Null Test and Parameter Estimates

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	11.7663	3	0.0082

Output 61.2.13 *continued*

Parameter Estimates					
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	-5.9902	1.6666	Infty	-3.59	0.0003
Heat	0.09634	0.04707	Infty	2.05	0.0407
Soak	0.2996	0.7551	Infty	0.40	0.6916
Heat*Soak	-0.00884	0.02532	Infty	-0.35	0.7270

Example 61.3: Ordinal Logistic Regression

(View the complete code for this example (hplogex3.sas) in the example repository.)

Consider a study of the effects of various cheese additives on taste. Researchers tested four cheese additives and obtained 52 response ratings for each additive. Each response was measured on a scale of nine categories ranging from strong dislike (1) to excellent taste (9). The data, given in McCullagh and Nelder (1989, p. 175) in the form of a two-way frequency table of additive by rating, are saved in the data set Cheese by using the following program. The variable y contains the response rating. The variable Additive specifies the cheese additive (1, 2, 3, or 4). The variable freq gives the frequency with which each additive received each rating.

```
data Cheese;
   do Additive = 1 to 4;
      do y = 1 to 9;
         input freq @@;
         output;
      end;
   end;
   label y='Taste Rating';
   datalines;
0  0  1  7  8  8 19  8  1
6  9 12 11  7  6  1  0  0
1  1  6  8 23  7  5  1  0
0  0  0  1  3  7 14 16 11
;
```

The response variable y is ordinally scaled. A cumulative logit model is used to investigate the effects of the cheese additives on taste. The following statements invoke PROC HPLOGISTIC to fit this model with y as the response variable and three indicator variables as explanatory variables, with the fourth additive as the reference level. With this parameterization, each Additive parameter compares an additive to the fourth additive.

```
proc hplogistic data=Cheese;
   freq freq;
   class Additive(ref='4') / param=ref ;
   model y=Additive;
   title 'Multiple Response Cheese Tasting Experiment';
run;
```

Results from the logistic analysis are shown in Output 61.3.1 through Output 61.3.3.

The "Response Profile" table in Output 61.3.1 shows that the strong dislike (y=1) end of the rating scale is associated with lower Ordered Values in the "Response Profile" table; hence the probability of disliking the additives is modeled.

Output 61.3.1 Proportional Odds Model Regression Analysis

Multiple Response Cheese Tasting Experiment

The HPLOGISTIC Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.CHEESE	V9	Input	On Client

Model Information	
Data Source	WORK.CHEESE
Response Variable	y
Frequency Variable	freq
Class Parameterization	Reference
Distribution	Multinomial
Link Function	Cumulative Logit
Optimization Technique	Newton-Raphson with Ridging

Class Level Information			
Class	**Levels**	**Reference Value**	**Values**
Additive	4	4	1 2 3 4

Number of Observations Read	36
Number of Observations Used	28
Sum of Frequencies Read	208
Sum of Frequencies Used	208

Response Profile		
Ordered Value	**y**	**Total Frequency**
1	1	7
2	2	10
3	3	19
4	4	27
5	5	41
6	6	28
7	7	39
8	8	25
9	9	12

You are modeling the probabilities of levels of y having lower Ordered Values in the Response Profile Table.

Output 61.3.2 Proportional Odds Model Regression Analysis

Iteration History				
Iteration	Evaluations	Objective Function	Change	Max Gradient
0	4	2.0668312595	.	0.137412
1	2	1.7319560317	0.33487523	0.062757
2	2	1.7105150048	0.02144103	0.008919
3	2	1.7099716191	0.00054339	0.00035
4	2	1.7099709251	0.00000069	6.981E-7
5	2	1.7099709251	0.00000000	2.98E-12

Convergence criterion (GCONV=1E-8) satisfied.

Dimensions	
Columns in X	11
Number of Effects	2
Max Effect Columns	8
Rank of Cross-product Matrix	11
Parameters in Optimization	11

Fit Statistics	
-2 Log Likelihood	711.35
AIC (smaller is better)	733.35
AICC (smaller is better)	734.69
BIC (smaller is better)	770.06

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	148.4539	3	<.0001

The positive value (1.6128) for the parameter estimate for Additive=1 in Output 61.3.3 indicates a tendency toward the lower-numbered categories of the first cheese additive relative to the fourth. In other words, the fourth additive tastes better than the first additive. Similarly, the second and third additives are both less favorable than the fourth additive. The relative magnitudes of these slope estimates imply the preference ordering: fourth, first, third, second.

Output 61.3.3 Proportional Odds Model Regression Analysis

Parameter Estimates						
Parameter	Taste Rating	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	1	-7.0802	0.5640	Infty	-12.55	<.0001
Intercept	2	-6.0250	0.4764	Infty	-12.65	<.0001
Intercept	3	-4.9254	0.4257	Infty	-11.57	<.0001
Intercept	4	-3.8568	0.3880	Infty	-9.94	<.0001
Intercept	5	-2.5206	0.3453	Infty	-7.30	<.0001
Intercept	6	-1.5685	0.3122	Infty	-5.02	<.0001
Intercept	7	-0.06688	0.2738	Infty	-0.24	0.8071
Intercept	8	1.4930	0.3357	Infty	4.45	<.0001
Additive 1		1.6128	0.3805	Infty	4.24	<.0001
Additive 2		4.9646	0.4767	Infty	10.41	<.0001
Additive 3		3.3227	0.4218	Infty	7.88	<.0001

Example 61.4: Partitioning Data

(View the complete code for this example (hplogex4.sas) in the example repository.)

The Sashelp.JunkMail data set comes from a study that classifies whether an email is junk email (coded as 1) or not (coded as 0). The data were collected by Hewlett-Packard Labs and donated by George Forman. The data set, which is specified in the following program, contains 4,601 observations, with 2 binary variables and 57 continuous explanatory variables. The response variable, Class, is a binary indicator of whether an email is considered spam or not. The partitioning variable, Test, is a binary indicator that is used to divide the data into training and testing sets. The 57 explanatory variables are continuous variables that represent frequencies of some common words and characters and lengths of uninterrupted sequences of capital letters in emails.

In the following program, the PARTITION statement divides the data into two parts. The training data have a Test value of 0 and contain about two-thirds of the data; the rest of the data are used to evaluate the fit. A forward selection method selects the best model based on the training observations.

```
proc hplogistic data=Sashelp.JunkMail;
   model Class(event='1')=Make Address All _3d Our Over Remove Internet Order
         Mail Receive Will People Report Addresses Free Business Email You
         Credit Your Font _000 Money HP HPL George _650 Lab Labs Telnet _857
         Data _415 _85 Technology _1999 Parts PM Direct CS Meeting Original
         Project RE Edu Table Conference Semicolon Paren Bracket Exclamation
         Dollar Pound CapAvg CapLong CapTotal;
   partition rolevar=Test(train='0' test='1');
   selection method=forward;
run;
```

Selected results from the analysis are shown in Output 61.4.1 through Output 61.4.3.

The “Number of Observations” and “Response Profile” tables in Output 61.4.1 are divided into training and testing columns.

Output 61.4.1 Partitioned Counts

The HPLOGISTIC Procedure

Number of Observations			
Description	**Total**	**Training**	**Testing**
Number of Observations Read	4601	3065	1536
Number of Observations Used	4601	3065	1536

Response Profile				
Ordered Value	**0 - Not Junk, 1 - Junk**	**Total Frequency**	**Training**	**Testing**
1	0	2788	1847	941
2	1	1813	1218	595

You are modeling the probability that Class='1'.

The standard likelihood-based fit statistics for the selected model are displayed in the "Fit Statistics" table, with a column for each of the training and testing subsets.

Output 61.4.2 Partitioned Fit Statistics

Fit Statistics		
Description	**Training**	**Testing**
-2 Log Likelihood	1202.18	813.03
AIC (smaller is better)	1262.18	873.03
AICC (smaller is better)	1262.80	874.27
BIC (smaller is better)	1443.02	1033.14

More fit statistics are displayed in the "Partition Fit Statistics" table shown in Output 61.4.3. These statistics are computed for both the training and testing data and should be very similar between the two groups when the training data are representative of the testing data. The statistics include the likelihood-based R-square statistics, as well as several prediction-based statistics that are described in the sections "Model Fit and Assessment Statistics" on page 4876 and "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 4880. For this model, the values of the statistics seem similar between the two disjoint subsets.

Output 61.4.3 More Partitioned Fit Statistics

Partition Fit Statistics		
Statistic	**Training**	**Testing**
Area under the ROCC	0.9769	0.9653
Average Square Error	0.05467	0.06351
Hosmer-Lemeshow Test	3.74E-49	0
Misclassification Error	0.07145	0.07878
R-Square	0.6139	0.5533
Max-rescaled R-Square	0.8305	0.7508
McFadden's R-Square	0.7081	0.6035
Mean Difference	0.7596	0.7393
Somers' D	0.9538	0.9307
True Negative Fraction	0.9556	0.9416
True Positive Fraction	0.8875	0.8891

If you want to display the "Partition Fit Statistics" table without partitioning your data set, you must identify all your data as training data. One way to do this is to define the fractions for the other roles to be zero:

```
proc hplogistic data=Sashelp.JunkMail;
   model Class(event='1')= Our Over Remove Internet Order Will
         Free Business You Your Font _000 Money HP George Parts
         Meeting RE Edu Semicolon Exclamation Dollar CapAvg
         CapLong;
   partition fraction(test=0 validation=0);
run;
```

Another way is to specify a constant variable as the training role:

```
data JunkMail;
   set Sashelp.JunkMail;
   Role=0;
run;
proc hplogistic data=JunkMail;
   model Class(event='1')= Our Over Remove Internet Order Will
         Free Business You Your Font _000 Money HP George Parts
         Meeting RE Edu Semicolon Exclamation Dollar CapAvg
         CapLong;
   partition role=Role(train='0');
run;
```

The resulting "Partition Fit Statistics" table is shown in Output 61.4.4.

Output 61.4.4 All Data Are Training Data

The HPLOGISTIC Procedure

Partition Fit Statistics	
Statistic	**Training**
Area under the ROCC	0.9724
Average Square Error	0.05910
Hosmer-Lemeshow Test	854E-220
Misclassification Error	0.07324
R-Square	0.5932
Max-rescaled R-Square	0.8034
McFadden's R-Square	0.6708
Mean Difference	0.7325
Somers' D	0.9448
True Negative Fraction	0.9570
True Positive Fraction	0.8803

References

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Albert, A., and Anderson, J. A. (1984). "On the Existence of Maximum Likelihood Estimates in Logistic Regression Models." *Biometrika* 71:1–10.

Brier, G. W. (1950). "Verification of Forecasts Expressed in Terms of Probability." *Monthly Weather Review* 78:1–3.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Cox, D. R., and Snell, E. J. (1989). *The Analysis of Binary Data*. 2nd ed. London: Chapman & Hall.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981). "An Adaptive Nonlinear Least-Squares Algorithm." *ACM Transactions on Mathematical Software* 7:348–368.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Eskow, E., and Schnabel, R. B. (1991). "Algorithm 695: Software for a New Modified Cholesky Factorization." *ACM Transactions on Mathematical Software* 17:306–312.

Fleiss, J. L. (1981). *Statistical Methods for Rates and Proportions*. 2nd ed. New York: John Wiley & Sons.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Hosmer, D. W., Jr., and Lemeshow, S. (2000). *Applied Logistic Regression*. 2nd ed. New York: John Wiley & Sons.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Lawless, J. F., and Singhal, K. (1978). "Efficient Screening of Nonnormal Regression Models." *Biometrics* 34:318–327.

Magee, L. (1990). "R^2 Measures Based on Wald and Likelihood Ratio Joint Significant Tests." *American Statistician* 44:250–253.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

McFadden, D. (1974). "Conditional Logit Analysis of Qualitative Choice Behavior." In *Frontiers in Econometrics*, edited by P. Zarembka, 105–142. New York: Academic Press.

McNicol, D. (2005). *A Primer of Signal Detection Theory*. Mahwah, NJ: Lawrence Erlbaum Associates.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Murphy, A. H. (1973). "A New Vector Partition of the Probability Score." *Journal of Applied Meteorology* 12:595–600.

Nagelkerke, N. J. D. (1991). "A Note on a General Definition of the Coefficient of Determination." *Biometrika* 78:691–692.

Pepe, M. S. (2003). *The Statistical Evaluation of Medical Tests for Classification and Prediction*. New York: Oxford University Press.

Santner, T. J., and Duffy, D. E. (1986). "A Note on A. Albert and J. A. Anderson's Conditions for the Existence of Maximum Likelihood Estimates in Logistic Regression Models." *Biometrika* 73:755–758.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Tjur, T. (2009). "Coefficients of Determination in Logistic Regression Models—a New Proposal: The Coefficient of Discrimination." *American Statistician* 63:366–372.

Chapter 62
The HPMIXED Procedure

Contents

Overview: HPMIXED Procedure . . . **4910**
- Basic Features . . . 4910
- Assumptions and Notation . . . 4911
- Computational Approach . . . 4912
- The HPMIXED Procedure Contrasted with the MIXED Procedure . . . 4913

Getting Started: HPMIXED Procedure . . . **4914**
- Mixed Model with Large Number of Fixed and Random Effects . . . 4914

Syntax: HPMIXED Procedure . . . **4916**
- PROC HPMIXED Statement . . . 4918
- BY Statement . . . 4923
- CLASS Statement . . . 4924
- CONTRAST Statement . . . 4925
- EFFECT Statement . . . 4928
- ESTIMATE Statement . . . 4929
- ID Statement . . . 4932
- LSMEANS Statement . . . 4932
- MODEL Statement . . . 4935
- NLOPTIONS Statement . . . 4936
- OUTPUT Statement . . . 4936
- PARMS Statement . . . 4939
- RANDOM Statement . . . 4942
- REPEATED Statement . . . 4947
- TEST Statement . . . 4949
- WEIGHT Statement . . . 4950

Details: HPMIXED Procedure . . . **4950**
- Model Assumptions . . . 4950
- Computing and Maximizing the Likelihood . . . 4951
- Computing Starting Values by EM-REML . . . 4952
- Sparse Matrix Techniques . . . 4953
- Hypothesis Tests for Fixed Effects . . . 4955
- Default Output . . . 4955
- ODS Table Names . . . 4957

Examples: HPMIXED Procedure . . . **4958**
- Example 62.1: Ranking Many Random-Effect Coefficients . . . 4958
- Example 62.2: Comparing Results from PROC HPMIXED and PROC MIXED . . . 4962
- Example 62.3: Using PROC GLIMMIX for Further Analysis of PROC HPMIXED Fit 4966

Example 62.4: Mixed Model Analysis of Microarray Data 4968
Example 62.5: Repeated Measures 4972
References **4976**

Overview: HPMIXED Procedure

The HPMIXED procedure uses a number of specialized high-performance techniques to fit linear mixed models with variance component structure. The HPMIXED procedure is specifically designed to cope with estimation problems involving a large number of fixed effects, a large number of random effects, or a large number of observations.

The HPMIXED procedure complements the MIXED procedure and other SAS/STAT procedures for mixed modeling. On the one hand, the models supported by the HPMIXED procedure are a subset of the models that you can fit with the MIXED procedure, and the confirmatory inferences available in the HPMIXED procedure are also a subset of the general analyses available with the MIXED procedure. On the other hand, the HPMIXED procedure can have considerably better performance than other SAS/STAT mixed modeling tools, in terms of memory requirements and computational speed.

A mixed model can be large in a number of ways, not all of which are suited for the specialized algorithms and storage techniques implemented in the HPMIXED procedure. The following are examples of linear mixed modeling problems for which the HPMIXED procedure has been specifically designed:

- linear mixed models with thousands of levels for the fixed and/or random effects
- linear mixed models with hierarchically nested fixed and/or random effects, possibly with hundreds or thousands of levels at each level of the hierarchy

Basic Features

The HPMIXED procedure enables you to specify a linear mixed model with variance component structure, to estimate the covariance parameters by restricted maximum likelihood, and to perform confirmatory inference in such models. The HPMIXED procedure fits the specified linear mixed model and produces appropriate statistics.

The following are some of the basic features of the HPMIXED procedure:

- capacity to handle large linear mixed model problems for balanced or unbalanced data
- MIXED-type MODEL and RANDOM statements for model specification and CONTRAST, ESTIMATE, LSMEANS, and TEST statements for inferences
- estimate covariance parameters by restricted maximum likelihood (REML)
- output statistics by using the OUTPUT statement

- computation of appropriate standard errors for all specified estimable linear combinations of fixed and random effects, and corresponding t and F tests
- subject and group effects that enable blocking and heterogeneity, respectively
- NLOPTIONS statement, which enables you to exercise control over the numerical optimization

The HPMIXED procedure uses the Output Delivery System (ODS), a SAS subsystem that provides capabilities for displaying and controlling the output from SAS procedures. ODS enables you to convert any of the output from the HPMIXED procedure into a SAS data set. See the section "ODS Table Names" on page 4957 and Chapter 23, "Using the Output Delivery System," for further information about using ODS with the HPMIXED procedure.

Assumptions and Notation

The linear mixed models fit by the HPMIXED procedure can be represented as linear statistical models in the following form:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

$$\boldsymbol{\gamma} \sim N(\mathbf{0}, \mathbf{G})$$

$$\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2\mathbf{I})$$

$$\mathrm{Cov}[\boldsymbol{\gamma}, \boldsymbol{\epsilon}] = \mathbf{0}$$

The symbols in these expressions denote the following:

$\mathbf{y}$	the $(n \times 1)$ vector of responses
$\mathbf{X}$	the $(n \times k)$ design matrix for the fixed effects
$\boldsymbol{\beta}$	the $(k \times 1)$ vector of fixed-effects parameters
$\mathbf{Z}$	the $(n \times q)$ design matrix for the random effects
$\boldsymbol{\gamma}$	the $(q \times 1)$ vector of random effects
$\boldsymbol{\epsilon}$	the $(n \times 1)$ vector of unobservable residual errors

As is customary for statistical models in the linear mixed model family, the random effects are assumed normally distributed. The same holds for the residual errors and these are furthermore distributed independently of the random effects. As a consequence, these assumptions imply that the response vector $\mathbf{y}$ has a multivariate normal distribution.

Further assumptions, implicit in the preceding expression, are as follows:

- The conditional mean of the data—given the random effects—is linear in the fixed effects and the random effects.
- The marginal mean of the data is linear in the fixed-effects parameters.

Computational Approach

The computational methods to efficiently solve large mixed model problems with the HPMIXED procedure rely on a combination of several techniques, including sparse matrix storage, specialized solving of sparse linear systems, and dedicated nonlinear optimization.

Sparse Storage and Computation

One of the fundamental computational tasks in analyzing a linear mixed model is solving the mixed model equations

$$\begin{bmatrix} \mathbf{X}'\mathbf{X} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{Z}'\mathbf{Z} + \sigma^2\mathbf{G}^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{y} \\ \mathbf{Z}'\mathbf{y} \end{bmatrix}$$

where $\mathbf{G}$ denotes the variance matrix of the random effects. The mixed model crossproduct matrix

$$\begin{bmatrix} \mathbf{X}'\mathbf{X} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{Z}'\mathbf{Z} + \sigma^2\mathbf{G}^{-1} \end{bmatrix}$$

is a key component of these equations, and it often has many zero values (George and Liu 1981). Sparse storage techniques can result in significant savings in both memory and CPU resources. The HPMIXED procedure draws on sparse matrix representation and storage where appropriate or necessary.

Conjugate Gradient Algorithm and Iteration-on-Data Technology

Solving the mixed model equations is a critical component of linear mixed model analysis. The two main components of the preconditioned conjugate gradient (PCCG) algorithm are preconditioning and matrix-vector product computing (Shewchuk 1994). The algorithm is guaranteed to converge to the solution within n_e iterations, where n_e is equal to the number of distinct eigenvalues of the mixed model equations. This simple yet powerful algorithm can be easily implemented with an iteration-on-data (IOD) technique (Tsuruta, Misztal, and Stranden 2001) that can yield significant savings of memory resources.

The combination of the PCCG algorithm and iteration on data makes it possible to efficiently compute best linear unbiased predictors (BLUPs) for the random effects in mixed models with large mixed model equations.

Average Information Algorithm

The HPMIXED procedure estimates covariance parameters by restricted maximum likelihood. The default optimization method is a quasi-Newton algorithm. When the Hessian or information matrix is required, the HPMIXED procedure takes advantage of the computational simplifications that are available by *averaging information* (AI). The AI algorithm (Johnson and Thompson 1995; Gilmour, Thompson, and Cullis 1995) replaces the second derivative matrix with the average of the observed and expected information matrices. The computationally intensive trace terms in these information matrices cancel upon averaging. Coarsely, the AI algorithm can be viewed as a hybrid of a Newton-Raphson approach and Fisher scoring.

The HPMIXED Procedure Contrasted with the MIXED Procedure

The HPMIXED procedure is designed to solve large mixed model problems by using sparse matrix techniques. A mixed model can be large in many ways: a large number of observations, a large number of columns in the **X** matrix, a large number of columns in the **Z** matrix, and a large number of covariance parameters. The aim of the HPMIXED procedure is parameter estimation, inference, and prediction in linear mixed models with large **X** and/or **Z** matrices and many observations, but with relatively few covariance parameters.

The models that you can fit with the HPMIXED procedure and the available postprocessing analyses are a subset of the models and analyses available with the MIXED procedure. With the HPMIXED procedure you can model only G-side random effects with variance component structure or an unstructured covariance matrix in a Cholesky parameterization. R-side random effects and direct modeling of their covariance structures are not supported.

The MIXED and HPMIXED procedures offer different balances for computing performance and statistical generality. To some extent the generality of the MIXED procedure means that it cannot serve as a high-performance computing tool for all of the model-data scenarios that it can potentially handle. For example, although efficient sparse algorithms are available to estimate variance components in large linear mixed models, the computational configuration changes profoundly when, for example, Kenward-Roger degree-of-freedom adjustments are requested.

On the other hand, the HPMIXED procedure can handle only a small subset of the models that PROC MIXED can fit. Invariably, some features of high-performance sparse computing methods might be surprising at first. For example, the best computational path depends on the model and the data, so that in models with a singular $\mathbf{X}'\mathbf{X}$ matrix, the order in which singularities are detected and accounted for can change from one data set to the next.

The following is a list of features available in the MIXED procedure, but *not* available in the HPMIXED procedure:

- a variety of covariance structures by using the TYPE= option in the RANDOM statement
- automatic Type III tests of fixed effects. You request tests of fixed effects in the HPMIXED procedure with the TEST statement.
- ODS statistical graphics
- advanced degree-of-freedom adjustments available by using the DDFM= option
- maximum likelihood or method-of-moments estimation for the covariance parameters
- a PRIOR statement for a sampling-based Bayesian analysis

Getting Started: HPMIXED Procedure

Mixed Model with Large Number of Fixed and Random Effects

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpmgs1.sas.)

In animal breeding, it is common to model genetic and environmental effects with a random effect for the animal. When there are many animals being studied, this can lead to very large mixed model equations to be solved. In this example we present an analysis of simulated data with this structure.

Suppose you have 3,000 animals from five different genetic species raised on 100 different farms. The following DATA step simulates 40,000 observations of milk yield (Yield) from a linear mixed model with variables Species and Farm in the fixed-effect model and Animal as a random effect. The random effect due to Animal is simulated with a variance of 4.0, while the residual error variance is 8.0. These variance component values reflect the fact that variation in milk yield is typically genetically controlled to be no more than 33% (4/(4+8)).

```
data Sim;
   keep Species Farm Animal Yield;
   array AnimalEffect{3000};
   array AnimalFarm{3000};
   array AnimalSpecies{3000};
   do i = 1 to dim(AnimalEffect);
      AnimalEffect{i}  = sqrt(4.0)*rannor(12345);
      AnimalFarm{i}    = 1 + int(100*ranuni(12345));
      AnimalSpecies{i} = 1 + int(5*ranuni(12345));
   end;
   do i = 1 to 40000;
      Animal  = 1 + int(3000*ranuni(12345));
      Species = AnimalSpecies{Animal};
      Farm    = AnimalFarm{Animal};
      Yield   = 1 + Species + Farm/10 + AnimalEffect{Animal}
                  + sqrt(8.0)*rannor(12345);
      output;
   end;
run;
```

A simple linear mixed model analysis is performed by using the following SAS statements:

```
proc hpmixed data=Sim;
   class Species Farm Animal;
   model Yield = Species Species*Farm;
   random Animal;
   test Species*Farm;
   contrast 'Species1 = Species2 = Species3'
      Species 1 0 -1,
      Species 0 1 -1;
run;
```

Selected results from the preceding SAS statements are shown in Figure 62.1 through Figure 62.4.

The "Class Level Information" table in Figure 62.1 shows that the three model effects have 5, 100, and 3,000 levels, respectively. Only a portion of the levels are displayed by default. The "Dimensions" table shows that the model contains a single G-side covariance parameter and a single R-side covariance parameter. R-side covariance parameters are those associated with the covariance matrix $\mathbf{R}$ in the conditional distribution, given the random effects. In the case of the HPMIXED procedure this matrix is simply $\mathbf{R} = \sigma^2\mathbf{I}$ and the single R-side covariance parameter corresponds to the residual variance. The G-side parameter is the variance of the random Animal effect; the $\mathbf{G}$ matrix is a diagonal $(3{,}000 \times 3{,}000)$ matrix with the common variance on the diagonal.

Figure 62.1 Class Levels and Dimensions

The HPMIXED Procedure

Class Level Information		
Class	**Levels**	**Values**
Species	5	1 2 3 4 5
Farm	100	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...
Animal	3000	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...

Dimensions	
G-side Cov. Parameters	1
R-side Cov. Parameters	1
Columns in X	506
Columns in Z	3000
Subjects (Blocks in V)	1

Taking into account the intercept as well as the number of levels of the Species and Species*Farm effects, the $\mathbf{X}$ matrix for this problem has 506 columns, so that the mixed model equations

$$\begin{bmatrix} \mathbf{X}'\mathbf{X} & \mathbf{X}'\mathbf{Z} \\ \mathbf{Z}'\mathbf{X} & \mathbf{Z}'\mathbf{Z} + \sigma^2\mathbf{G}^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{y} \\ \mathbf{Z}'\mathbf{y} \end{bmatrix}$$

have 3,506 rows and columns. This is a substantial computational problem: simply storing a single copy of this matrix in dense format requires nearly 50 megabytes of memory. The sparse matrix techniques of PROC HPMIXED use a small fraction of this amount of memory and a similarly small fraction of the CPU time required to solve the equations with dense techniques. For more information about sparse versus dense techniques, see the section "Sparse Matrix Techniques" on page 4953.

Figure 62.2 displays the covariance parameter estimates at convergence of the REML algorithm. The variance component estimate for animal effect is $\widehat{\sigma}_a^2 = 3.9889$ and for residual $\widehat{\sigma}^2 = 7.9623$. These estimates are close to the simulated values (4.0 and 8.0).

Figure 62.2 Estimates of Variance Components

Covariance Parameter Estimates	
Cov Parm	**Estimate**
Animal	3.9889
Residual	7.9623

The TEST statement requests a Type III test of the fixed effect in the model. By default, the HPMIXED procedure does not compute Type III tests, because they can be computationally demanding. The tests of the Species*Farm effect is highly significant. That indicates animals of a genetic species perform differently in different environments.

Figure 62.3 Type III Tests of Fixed Effect

Type III Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
Species*Farm	495	39500	11.72	<.0001

You can use the CONTRAST or ESTIMATE statement to test custom linear hypotheses involving the fixed and/or random effects. The CONTRAST statement in the preceding program tests the null hypothesis that there are no differences among the first three genetic species. Results from this analysis are shown in Figure 62.4. The small p-value indicates that there are significant differences among the first three genetics species.

Figure 62.4 Result of CONTRAST Statement

Contrasts				
Label	Num DF	Den DF	F Value	Pr > F
Species1 = Species2 = Species3	2	39500	92.93	<.0001

Syntax: HPMIXED Procedure

The following statements are available in the HPMIXED procedure:

```
PROC HPMIXED <options> ;
    BY variables ;
    CLASS variable <(REF= option)> ... <variable <(REF= option)>> </ global-options> ;
    EFFECT name=effect-type (variables </ options>) ;
    ID variables ;
    MODEL dependent = <fixed-effects> </ options> ;
    RANDOM random-effects </ options> ;
    REPEATED repeated-effect </ options> ;
    PARMS <(value-list)... > </ options> ;
    TEST fixed-effects </ options> ;
    CONTRAST 'label' contrast-specification <, contrast-specification> <, ... > </ options> ;
    ESTIMATE 'label' contrast-specification <(divisor=n)>
              <, 'label' contrast-specification <(divisor=n)>> <, ... > </ options> ;
    LSMEANS fixed-effects </ options> ;
    NLOPTIONS <options> ;
    OUTPUT <OUT=SAS-data-set>
            <keyword<(keyword-options)> <=name>> ...
            <keyword<(keyword-options)> <=name>> </ options> ;
    WEIGHT variable ;
```

Items within angle brackets (< >) are optional. The CONTRAST, ESTIMATE, LSMEANS, RANDOM, and TEST statements can appear multiple times; all other statements can appear only once.

The PROC HPMIXED and MODEL statements are required, and the MODEL statement must appear after the CLASS statement if these statements are included. The BY, CLASS, MODEL, ID, OUTPUT, TEST, RANDOM, REPEATED and WEIGHT statements are described in full after the PROC HPMIXED statement in alphabetical order. The EFFECT, is shared with many other procedures. Summary descriptions of functionality and syntax for this statement is also given after the PROC HPMIXED statement in alphabetical order, but you can find full documentation on it in Chapter 20, "Shared Concepts and Topics."

Table 62.1 summarizes the basic functions and important options of each PROC HPMIXED statement.

Table 62.1 Summary of PROC HPMIXED Statements

Statement	Description	Options
PROC HPMIXED	Invokes the procedure	DATA= specifies input data set, METHOD= specifies estimation method
BY	Performs multiple PROC HPMIXED analyses in one invocation	None
CLASS	Declares qualitative variables that create indicator variables in design matrices	None
ID	Lists additional variables to be included in predicted values tables	None

Table 62.1 *continued*

Statement	Description	Options
MODEL	Specifies dependent variable and fixed effects, setting up **X**	S requests solution for fixed-effects parameters, DDFM= specifies denominator degrees of freedom method
RANDOM	Specifies random effects, setting up **Z** and **G**	SUBJECT= creates block-diagonality, TYPE= specifies covariance structure, S requests solution for random-effects parameters
REPEATED	Sets up **R**	SUBJECT= creates block-diagonality, TYPE= specifies covariance structure, R= displays estimated blocks of **R**, GROUP= enables between-subject heterogeneity
PARMS	Specifies a grid of initial values for the covariance parameters	HOLD= and NOITER hold the covariance parameters or their ratios constant, PARMSDATA= reads the initial values from a SAS data set
CONTRAST	Constructs custom hypothesis tests	E displays the **L** matrix coefficients
ESTIMATE	Constructs custom scalar estimates	CL produces confidence limits
LSMEANS	Computes least squares means for classification fixed effects	DIFF computes differences of the least squares means, CL produces confidence limits, SLICE= tests simple effects
WEIGHT	Specifies a variable by which to weight **R**	None

PROC HPMIXED Statement

PROC HPMIXED < *options* > **;**

The PROC HPMIXED statement invokes the HPMIXED procedure. Table 62.2 summarizes the *options* available in the PROC HPMIXED statement. These and other *options* in the PROC HPMIXED statement are then described fully in alphabetical order.

Table 62.2 PROC HPMIXED Statement Options

Option	Description
Basic Options	
DATA=	Specifies input data set
METHOD=	Specifies the estimation method
NOPROFILE	Includes scale parameter in optimization
ORDER=	Determines the sort order of CLASS variables
BLUP	Computes BLUP/BLUE only

Table 62.2 *continued*

Option	Description
Displayed Output	
IC=	Displays a table of information criteria
ITDETAILS	Displays estimates and gradients added to "Iteration History"
MAXCLPRINT=	Specifies the maximum levels of CLASS variables to print
LOGNOTE	Writes periodic status notes to the log
MMEQ	Displays mixed model equations
NOCLPRINT	Suppresses "Class Level Information" completely or in parts
NOITPRINT	Suppresses "Iteration History" table
RANKS	Displays a table of ranks of matrices **X**, (**XZ**), and **MMEQ**
SIMPLE	Displays "Descriptive Statistics" table
Singularity Tolerances	
SINGCHOL=	Tunes singularity for Cholesky decompositions
SINGRES=	Tunes singularity for the residual variance
SINGULAR=	Tunes general singularity criterion

You can specify the following *options*.

BLUP< *(suboptions)* >**=***SAS-data-set*

creates a data set that contains the BLUE and BLUP solutions.The covariance parameters are assumed to be known and given by PARMS statement. All hypothesis testing is ignored. The statements TEST, ESTIMATE, CONTRAST, LSMEANS, and OUTPUT are all ignored. This option is designed for users who need BLUP solutions for random effects with many levels, up to tens of millions.

You can specify the following *suboptions*:

ITPRINT=*number* specifies that the iteration history be displayed after every *number* of iterations. This *suboption* applies only for iterative solving methods (IOC or IOD). The default value is 10, which means the procedure displays the iteration history for every 10 iterations.

MAXITER=*number* specifies the maximum number of iterations allowed. This applies only for iterative solving methods (IOC or IOD). The default value is the number of parameters in the BLUE/BLUP plus two.

METHOD=**DIRECT** | **IOC** | **IOD** specifies the method used to solve for BLUP solutions. METHOD=DIRECT requires storing mixed model equations (MMEQ) in memory and computing the Cholesky decomposition of MMEQ. This method is the most accurate, but it is the most inefficient in terms of speed and memory. METHOD=IOD does not build mixed model equations; instead it iterates on data to solve for the solutions. This method is most efficient in terms of memory. METHOD=IOC requires storing mixed model equations in memory and iterates on MMEQ to solve for the solutions. This method is the most efficient in terms of speed. The default method is IOC.

TOL=*number* specifies the tolerance value. This *suboption* applies only for iterative solving methods (IOC or IOD). The default value is the square root of machine precision.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC HPMIXED. The default is the most recently created data set.

INFOCRIT=NONE | PQ | Q

IC=NONE | PQ | Q

determines the computation of information criteria in the "Fit Statistics" table. The criteria are all in smaller-is-better form, and are described in Table 62.3.

Table 62.3 Information Criteria

Criteria	Formula	Reference
AIC	$-2\ell + 2d$	Akaike (1974)
AICC	$-2\ell + 2dn^*/(n^* - d - 1)$ for $n^* \geq d + 2$ $-2\ell + 2d(d + 2)$ for $n^* < d + 2$	Hurvich and Tsai (1989) and Burnham and Anderson (1998)
HQIC	$-2\ell + 2d \log(\log(n))$ for $n > 1$	Hannan and Quinn (1979)
BIC	$-2\ell + d \log(n)$ for $n > 0$	Schwarz (1978)
CAIC	$-2\ell + d(\log(n) + 1)$ for $n > 0$	Bozdogan (1987)

Here ℓ denotes the maximum value of the restricted log likelihood, d is the dimension of the model, and n, n^* reflect the size of the data. When $n \leq 1$, the value of the HQIC criterion is -2ℓ. When n=0, the values of the BIC and CAIC criteria are undefined.

The quantities d, n, and n^* depend on the model and IC= option.

- models without random effects:
  The IC=Q and IC=PQ options have no effect on the computation.
  - d equals the number of parameters in the optimization whose solutions do not fall on the boundary or are otherwise constrained.
  - n equals the number of used observations minus rank($\mathbf{X}$).
  - n^* equals n, unless $n < d + 2$, in which case $n^* = d + 2$.
- models with random effects:
  - d equals the number of parameters in the optimization whose solutions do not fall on the boundary or are otherwise constrained. If IC=PQ, this value is incremented by rank($\mathbf{X}$).
  - n equals the effective number of subjects as displayed in the "Dimensions" table, unless this value equals 1, in which case n equals the number of levels of the first random effect specified. The IC=Q and IC=PQ options have no effect.
  - n^* equals n, unless $n < d + 2$, in which case $n^* = d + 2$. The IC=Q and IC=PQ options have no effect.

The IC=NONE option suppresses the "Fit Statistics" table. IC=Q is the default.

ITDETAILS

displays the parameter values at each iteration and enables the writing of notes to the SAS log pertaining to "infinite likelihood" and "singularities" during optimization iterations.

LOGNOTE

writes to the log periodic notes that describe the current status of computations. This option is designed for use with analyses that require extensive CPU resources.

MAXCLPRINT=*number*

specifies the maximum levels of CLASS variables to print in the ODS table ClassLevels. The default value is 20. MAXCLPRINT=0 enables you to print all levels of each CLASS variable. However, the option NOCLPRINT takes precedence over MAXCLPRINT.

METHOD=

specifies the estimation method for the covariance parameters. The REML specification performs residual (restricted) maximum likelihood, and it is currently the only available method. This option is therefore currently redundant for PROC HPMIXED, but it is included for consistency with other mixed model procedures in SAS/STAT software.

MMEQ

displays coefficients of the mixed model equations. These are

$$\begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}+\widehat{\mathbf{G}}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \end{bmatrix}$$

assuming $\widehat{\mathbf{G}}$ is nonsingular. If $\widehat{\mathbf{G}}$ is singular, PROC HPMIXED produces the following coefficients

$$\begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}\widehat{\mathbf{G}} \\ \widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}\widehat{\mathbf{G}}+\widehat{\mathbf{G}} \end{bmatrix} \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \\ \widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \end{bmatrix}$$

See the section "Model and Assumptions" on page 4950 for further information about these equations.

NAMELEN=*number*

specifies the length to which long effect names are shortened. The default and minimum value is 20.

NLPRINT

requests that optimization-related output options specified in the NLOPTIONS statement override corresponding options in the PROC HPMIXED statement. When you specify NLPRINT, the ITDETAILS and NOITPRINT options in the PROC HPMIXED statement are ignored and the following six options in the NLOPTIONS statement are enabled: NOPRINT, PHISTORY, PSUMMARY, PALL, PLONG, and PHISTPARMS.

The syntax and options of the NLOPTIONS statement are described in the section "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics."

NOCLPRINT< =*number*>

suppresses the display of the "Class Level Information" table if you do not specify *number*. If you do specify *number*, only levels with totals that are less than *number* are listed in the table.

NOFIT

suppresses fitting of the model. When the NOFIT option is in effect, PROC HPMIXED produces the "Model Information," "Class Level Information," "Number of Observations," "Dimensions," and "Descriptive Statistics" tables. These can be helpful in gauging the computational effort required to fit the model.

NOINFO

suppresses the display of the "Model Information," "Number of Observations," and "Dimensions" tables.

NOITPRINT

suppresses the display of the "Iteration History" table.

NOPRINT

suppresses the normal display of results. The NOPRINT option is useful when you want only to create one or more output data sets with the procedure by using the OUTPUT statement. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

NOPROFILE

includes the residual variance as one of the covariance parameters in the optimization iterations. This option applies only to models that have a residual variance parameter. By default, this parameter is profiled out of the optimization iterations, except when you have specified the HOLD= option in the PARMS statement.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

RANKS
displays the ranks of design matrices **X** and (**XZ**) and the coefficient matrix of the mixed model equations (**MMEQ**).

SIMPLE
displays the mean, standard deviation, coefficient of variation, minimum, and maximum for each variable used in PROC HPMIXED that is not a classification variable.

SINGCHOL=*number*
tunes the singularity criterion in Cholesky decompositions. The default is 1E6 times the machine epsilon; this product is approximately 1E–10 on most computers.

SINGRES=*number*
sets the tolerance for which the residual variance is considered to be zero. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGULAR=*number*
tunes the general singularity criterion applied by the HPMIXED procedure in divisions and inversions. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

UPDATE
is an alias for the LOGNOTE option.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPMIXED to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPMIXED procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Since sorting the data changes the order in which PROC HPMIXED reads observations, the sort order for the levels of the CLASS variable might be affected if you have specified ORDER=DATA in the PROC HPMIXED statement. This, in turn, affects specifications in the CONTRAST and ESTIMATE statements.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(REF=** *option***)** > . . . < *variable* < **(REF=** *option***)** > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

NOTE: Prior to SAS 9, class levels were determined by using no more than the first 16 characters of the formatted values. To revert to this previous behavior, you can use the TRUNCATE option in the CLASS statement.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC HPMIXED statement.

You can specify the following REF= option to indicate how the levels of an individual classification variable are to be ordered by enclosing it in parentheses after the variable name:

REF=*'level'* **| FIRST | LAST**

specifies a level of the classification variable to be put at the end of the list of levels. (In procedures that solve mixed model equations by sequentially sweeping rows and columns, this level thus corresponds to the reference level in the usual interpretation of the estimates of a singular parameterization. However, since PROC HPMIXED does not necessarily solve mixed model equations in the original order, this interpretation of the specified REF= level does not apply for this procedure.) You can specify the *level* of the variable to use as the reference level; specify a value that corresponds to the formatted value of the variable if a format is assigned. Alternatively, you can specify REF=FIRST to designate that the first ordered level serve as the reference, or REF=LAST to designate that the last ordered level serve as the reference. To specify that REF=FIRST or REF=LAST be used for all classification variables, use the REF= *global-option* after the slash (/) in the CLASS statement.

You can specify the following *global-options* in the CLASS statement after a slash (/):

REF=FIRST | LAST

specifies a level of all classification variables to be put at the end of the list of levels. (In procedures that solve mixed model equations by sequentially sweeping rows and columns, this level thus corresponds to the reference level in the usual interpretation of the estimates of a singular parameterization. However, since PROC HPMIXED does not necessarily solve mixed model equations in the original order, this interpretation of the specified REF= level does not apply for this procedure.) Specify REF=FIRST to designate that the first ordered level for each classification variable serve as the reference. Specify REF=LAST to designate that the last ordered level serve as the reference. This option applies to all the variables specified in the CLASS statement. To specify different reference levels for different classification variables, use REF= options for individual variables.

TRUNCATE
specifies that class levels be determined by using only up to the first 16 characters of the formatted values of CLASS variables. When formatted values are longer than 16 characters, you can use this option to revert to the levels as determined in releases prior to SAS 9.

The "Class Levels Information" table displays the number of levels for each CLASS variable. If a CLASS variable is a string, then because some string values can have spaces at the beginning or end (or both), those spaces are taken into account when the number of distinctive values is counted. However, those spaces are eliminated when the data are used in further analysis.

CONTRAST Statement

CONTRAST *'label' contrast-specification* < **,** *contrast-specification* > < **,** . . . > < / *options* > **;**

The CONTRAST statement provides a mechanism for obtaining custom hypothesis tests. It is patterned after the CONTRAST statement in PROC MIXED and enables you to select an appropriate inference space (McLean, Sanders, and Stroup 1991).

You can test the hypothesis $\mathbf{L}'\boldsymbol{\phi} = \mathbf{0}$, where $\mathbf{L}' = [\mathbf{K}'\ \mathbf{M}']$ and $\boldsymbol{\phi}' = [\boldsymbol{\beta}'\ \boldsymbol{\gamma}']$, in several inference spaces. The inference space corresponds to the choice of $\mathbf{M}$. When $\mathbf{M} = \mathbf{0}$, your inferences apply to the entire population from which the random effects are sampled; this is known as the *broad* inference space. When all elements of $\mathbf{M}$ are nonzero, your inferences apply only to the observed levels of the random effects. This is known as the *narrow* inference space, and you can also choose it by specifying all of the random effects as fixed. The GLM procedure uses the narrow inference space. Finally, by zeroing portions of $\mathbf{M}$ corresponding to selected main effects and interactions, you can choose *intermediate* inference spaces. The broad inference space is usually the most appropriate, and it is used when you do not specify any random effects in the CONTRAST statement.

In the CONTRAST statement,

label identifies the contrast in the table. A label is required for every contrast specified. Labels can be up to 20 characters and must be enclosed in single quotes.

contrast-specification identifies the fixed effects and random effects and their coefficients from which the $\mathbf{L}$ matrix is formed. The syntax representation of a *contrast-specification* is
< fixed-effect values . . . > < | random-effect values . . . >

fixed-effect identifies an effect that appears in the MODEL statement. The keyword INTERCEPT can be used as an effect when an intercept is fitted in the model. You do not need to include all effects that are in the MODEL statement.

random-effect identifies an effect that appears in the RANDOM statement. The first random effect must follow a vertical bar (|); however, random effects do not have to be specified.

values are constants that are elements of the $\mathbf{L}$ matrix associated with the fixed and random effects.

The rows of $\mathbf{L}'$ are specified in order and are separated by commas. The rows of the $\mathbf{K}'$ component of $\mathbf{L}'$ are specified on the left side of the vertical bars (|). These rows test the fixed effects and are, therefore, checked for estimability. The rows of the $\mathbf{M}'$ component of $\mathbf{L}'$ are specified on the right side of the vertical bars. They test the random effects, and no estimability checking is necessary.

If PROC HPMIXED finds the fixed-effects portion of the specified contrast to be nonestimable (see the SINGULAR= option on page 4927), then it displays missing values for the test statistics and a note in the log.

If the elements of **L** are not specified for an effect that contains a specified effect, then the elements of the specified effect are automatically "filled in" over the levels of the higher-order effect. This feature is designed to preserve estimability for cases where there are complex higher-order effects. The coefficients for the higher-order effect are determined by equitably distributing the coefficients of the lower-level effect as in the construction of least squares means. In addition, if the intercept is specified, it is distributed over all classification effects that are not contained by any other specified effect. If an effect is not specified and does not contain any specified effects, then all of its coefficients in **L** are set to 0. You can override this behavior by specifying coefficients for the higher-order effect.

If too many values are specified for an effect, the extra ones are ignored; if too few are specified, the remaining ones are set to 0. If no random effects are specified, the vertical bar can be omitted; otherwise, it must be present. If a SUBJECT effect is used in the RANDOM statement, then the coefficients specified for the effects in the RANDOM statement are equitably distributed across the levels of the SUBJECT effect. You can use the E option to see exactly what **L** matrix is used.

The SUBJECT and GROUP options in the CONTRAST statement are useful for the case where a SUBJECT= or GROUP= variable appears in the RANDOM statement, and you want to contrast different subjects or groups. By default, CONTRAST statement coefficients about random effects are distributed equally across subjects and groups.

PROC HPMIXED handles missing level combinations of CLASS variables similarly to the way PROC GLM does. Both procedures delete fixed-effects parameters corresponding to missing levels in order to preserve estimability. However, PROC HPMIXED does not delete missing level combinations for random-effects parameters because linear combinations of the random-effects parameters are always estimable. These conventions can affect the way you specify your CONTRAST coefficients.

The CONTRAST statement computes the statistic

$$F = \frac{\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}' \mathbf{L}(\mathbf{L}'\widehat{\mathbf{C}}\mathbf{L})^{-1}\mathbf{L}' \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}}{r}$$

where $r = \mathrm{rank}(\mathbf{L}'\widehat{\mathbf{C}}\mathbf{L})$ and approximates its distribution with an F distribution. In this expression, $\widehat{\mathbf{C}}$ is an estimate of the generalized inverse of the coefficient matrix in the mixed model equations.

The numerator degree of freedom in the F approximation is $r = \mathrm{rank}(\mathbf{L}'\widehat{\mathbf{C}}\mathbf{L})$, and the denominator degree of freedom is taken from the "Type III Tests of Fixed Effects" table and corresponds to the final effect you list in the CONTRAST statement. You can change the denominator degrees of freedom by using the DF= option.

You can specify the following *options* in the CONTRAST statement after a slash (/).

CHISQ

requests that χ^2 tests be performed in addition to any F tests. A χ^2 statistic equals its corresponding F statistic times the associate numerator degree of freedom, and this same degree of freedom is used to compute the p-value for the χ^2 test. This p-value will always be less than that for the F test, as it effectively corresponds to an F test with infinite denominator degrees of freedom.

DF=*number*

specifies the denominator degrees of freedom for the F test. The default is the denominator degrees of freedom taken from the "Type III Tests of Fixed Effects" table and corresponds to the final effect you list in the CONTRAST statement.

E

requests that the **L** matrix coefficients for the contrast be displayed. The name of this "L Matrix Coefficients" table is "Coef."

GROUP *coeffs*

sets up random-effect contrasts between different groups when a GROUP= variable appears in the RANDOM statement. By default, CONTRAST statement coefficients about random effects are distributed equally across groups. If you enter a multi-row contrast, you can also enter multiple rows for the GROUP coefficients. If the number of GROUP coefficients is less than the number of contrasts in the CONTRAST statement, the HPMIXED procedure cycles through the GROUP coefficients. For example, the following two statements are equivalent:

```
contrast 'Trt @ x=0.4 and 0.5' trt 1 -1  0 | x 0.4,
                               trt 1  0 -1 | x 0.4,
                               trt 1 -1  0 | x 0.5,
                               trt 1  0 -1 | x 0.5 /
                group 1 -1, 1 0 -1, 1 -1, 1 0 -1;

contrast 'Trt @ x=0.4 and 0.5' trt 1 -1  0 | x 0.4,
                               trt 1  0 -1 | x 0.4,
                               trt 1 -1  0 | x 0.5,
                               trt 1  0 -1 | x 0.5 /
                group 1 -1, 1 0 -1;
```

SINGULAR=*number*

tunes the estimability checking. If **v** is a vector, define ABS(**v**) to be the largest absolute value of the element of **v** with the largest absolute value. If ABS($\mathbf{K}' - \mathbf{K}'\mathbf{T}$) is greater than c**number* for any row of $\mathbf{K}'$ in the contrast, then **K** is declared nonestimable. Here **T** is the Hermite form matrix $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$, and c is ABS($\mathbf{K}'$) except when it equals 0, and then c is 1. The value for *number* must be between 0 and 1; the default is 1E–4.

SUBJECT *coeffs*

sets up random-effect contrasts between different subjects when a SUBJECT= variable appears in the RANDOM statement. By default, CONTRAST statement coefficients about random effects are distributed equally across subjects. Listing subject coefficients for multiple row CONTRASTS follows the same rules as for GROUP coefficients.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 62.4 summarizes the *options* available in the EFFECT statement.

Table 62.4 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.

Table 62.4 *continued*

Option	Description
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

The HPMIXED procedure does not support the SPLIT or SEPARATED option in spline effects and poly effects.

ESTIMATE Statement

ESTIMATE *'label' contrast-specification* <(*divisor*=*n*)>
<, *'label' contrast-specification* <(*divisor*=*n*)>> <, . . . > </ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. As in the CONTRAST statement, the basic element of the ESTIMATE statement is the *contrast-specification*, which consists of MODEL and RANDOM effects and their coefficients. Specifically, a *contrast-specification* takes the form

< *fixed-effect values* . . . > < | *random-effect values* . . . >

Based on the *contrast-specifications* in your ESTIMATE statement, PROC HPMIXED constructs the matrix $\mathbf{L}' = [\mathbf{K}' \ \mathbf{M}']$, as in the CONTRAST statement, where $\mathbf{K}$ is associated with the fixed effects and $\mathbf{M}$ is associated with the G-side random effects.

PROC HPMIXED then produces for each row $\mathbf{l}$ of $\mathbf{L}'$ an approximate t test of the hypothesis $H: \mathbf{l}\boldsymbol{\phi} = 0$, where $\boldsymbol{\phi} = [\boldsymbol{\beta}' \ \boldsymbol{\gamma}']'$. Results from all ESTIMATE statement are combined in the "Estimates" ODS table.

Note that multi-row estimates are permitted. Unlike the CONTRAST statement, you need to specify a *'label'* for every row of the multi-row estimate, since PROC HPMIXED produces one test per row.

PROC HPMIXED selects the degrees of freedom to match those displayed in the "Type III Tests of Fixed Effects" table for the final effect you list in the ESTIMATE statement. You can modify the degrees of freedom by using the DF= option. If you select DDFM=NONE and do not modify the degrees of freedom by using the DF= option, PROC HPMIXED uses infinite degrees of freedom, essentially computing approximate z tests.

If PROC HPMIXED finds the fixed-effects portion of the specified estimate to be nonestimable, then it displays "Non-est" for the estimate entry.

The construction of the **L** matrix for an ESTIMATE statement follows the same rules as listed under the CONTRAST statement.

Table 62.5 summarizes the *options* available in the ESTIMATE statement.

Table 62.5 ESTIMATE Statement Options

Option	Description
ALPHA=	Specifies the confidence level
CL	Constructs t-type confidence limits
DF=	Specifies the degrees of freedom
DIVISOR=	Specifies values to divide the coefficients
E	Displays the matrix coefficients
GROUP	Sets up random-effect contrasts between groups
SINGULAR=	Tunes the estimability checking
SUBJECT	Sets up random-effect estimates between subjects

You can specify the following *options* in the ESTIMATE statement after a slash (/).

ALPHA=*number*

requests that a t-type confidence interval be constructed with confidence level $1 - \mathit{number}$. The value of *number* must be between 0 and 1 exclusively; the default is 0.05. If DDFM=NONE and you do not specify degrees of freedom with the DF= option, PROC HPMIXED uses infinite degrees of freedom, essentially computing a z interval.

CL

requests that t-type confidence limits be constructed. If DDFM=NONE and you do not specify degrees of freedom with the DF= option, PROC HPMIXED uses infinite degrees of freedom, essentially computing a z interval. The confidence level is 0.95 by default.

DF=*number*

specifies the degrees of freedom for the t-test. The default is the denominator degrees of freedom taken from the "Type III Tests of Fixed Effects" table and corresponds to the final effect you list in the ESTIMATE statement.

DIVISOR=*value-list*

specifies a list of values by which to divide the coefficients so that fractional coefficients can be entered as integer numerators. If you do not specify *value-list*, a default value of 1.0 is assumed. Missing values in the *value-list* are converted to 1.0.

If the number of elements in *value-list* exceeds the number of rows of the estimate, the extra values are ignored. If the number of elements in *value-list* is less than the number of rows of the estimate, the last value in *value-list* is copied forward.

If you specify a row-specific divisor as part of the specification of the estimate row, this value multiplies the corresponding divisor implied by the *value-list*. For example, the following statement divides the coefficients in the first row by 8, and the coefficients in the third and fourth row by 3:

```
estimate 'One vs. two'   A 2 -2  (divisor=2),
         'One vs. three' A 1  0 -1         ,
         'One vs. four'  A 3  0  0 -3      ,
         'One vs. five'  A 1  0  0  0  -1  / divisor=4,.,3;
```

E

requests that the matrix coefficients be displayed. For ODS purposes, the name of this "L Matrix Coefficients" table is "Coef."

GROUP *coeffs*

sets up random-effect contrasts between different groups when a GROUP= variable appears in the RANDOM statement. By default, ESTIMATE statement coefficients about random effects are distributed equally across groups. If you enter a multi-row estimate, you can also enter multiple rows for the GROUP coefficients. If the number of GROUP coefficients is less than the number of contrasts in the ESTIMATE statement, the HPMIXED procedure cycles through the GROUP coefficients. For example, the following two statements are equivalent:

```
estimate 'Trt 1 vs 2 @ x=0.4' trt 1 -1  0 | x 0.4,
         'Trt 1 vs 3 @ x=0.4' trt 1  0 -1 | x 0.4,
         'Trt 1 vs 2 @ x=0.5' trt 1 -1  0 | x 0.5,
         'Trt 1 vs 3 @ x=0.5' trt 1  0 -1 | x 0.5 /
                group 1 -1, 1 0 -1, 1 -1, 1 0 -1;

estimate 'Trt 1 vs 2 @ x=0.4' trt 1 -1  0 | x 0.4,
         'Trt 1 vs 3 @ x=0.4' trt 1  0 -1 | x 0.4,
         'Trt 1 vs 2 @ x=0.5' trt 1 -1  0 | x 0.5,
         'Trt 1 vs 3 @ x=0.5' trt 1  0 -1 | x 0.5 /
                group 1 -1, 1 0 -1;
```

SINGULAR=*number*

tunes the estimability checking as documented for the SINGULAR= in the CONTRAST statement.

SUBJECT *coeffs*

sets up random-effect estimates between different subjects when a SUBJECT= variable appears in the RANDOM statement. By default, ESTIMATE statement coefficients about random effects are distributed equally across subjects. Listing subject coefficients for an ESTIMATE statement with multiple rows follows the same rules as for GROUP coefficients.

ID Statement

ID *variables* **;**

The ID statement specifies which variables from the input data set are to be included in the OUT= data sets from the OUTPUT statement. If you do not specify an ID statement, then all variables are included in these data sets. Otherwise, only the variables you list in the ID statement are included. Specifying an ID statement with no variables prevents any variables from being included in these data sets.

LSMEANS Statement

LSMEANS *fixed-effects* < / *options* > **;**

The LSMEANS statement computes least squares means (LS-means) of fixed effects. As in the GLM procedure, LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as classification and subclassification arithmetic means are to balanced designs. The **L** matrix constructed to compute them is the same as the **L** matrix formed in PROC GLM; however, the standard errors are adjusted for the covariance parameters in the model.

Each LS-mean is computed as $\mathbf{L}'\widehat{\boldsymbol{\beta}}$, where **L** is the coefficient matrix associated with the least squares mean and $\widehat{\boldsymbol{\beta}}$ is the estimate of the fixed-effects parameter vector. The approximate standard errors for the LS-mean is computed as the square root of $\mathbf{L}'(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}\mathbf{L}$.

LS-means can be computed for any effect in the MODEL statement that involves CLASS variables. You can specify multiple effects in one LSMEANS statement or in multiple LSMEANS statements, and all LSMEANS statements must appear after the MODEL statement. As in the ESTIMATE statement, the **L** matrix is tested for estimability, and if this test fails, PROC HPMIXED displays "Non-est" for the LS-means entries.

Assuming the LS-mean is estimable, PROC HPMIXED constructs an approximate t test to test the null hypothesis that the associated population quantity equals zero. By default, the denominator degrees of freedom for this test are the same as those displayed for the effect in the "Type III Tests of Fixed Effects" table (see the section "TEST Statement" on page 4949).

Table 62.6 summarizes the *options* available in the LSMEANS statement.

Table 62.6 LSMEANS Statement Options

Option	Description
ALPHA=	Specifies the confidence level
CL	Constructs t-type confidence limits

Table 62.6 *continued*

Option	Description
CORR	Displays the estimated correlation matrix
COV	Displays the estimated covariance matrix
DF=	Specifies the degrees of freedom
DIFF or PDIFF	Displays the differences of the LS-means
E	Displays the matrix coefficients for LSMEANS effects
SINGULAR=	Tunes the estimability checking
SLICE=	Partitions interaction LSMEANS effects

You can specify the following *options* in the LSMEANS statement after a slash (/).

ALPHA=*number*

requests that a t-type confidence interval be constructed for each of the LS-means with confidence level $1 - \mathit{number}$. The value of *number* must be between 0 and 1; the default is 0.05.

CL

requests that t-type confidence limits be constructed for each of the LS-means. If DDFM=NONE, then PROC HPMIXED uses infinite degrees of freedom for this test, essentially computing a z interval. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

CORR

displays the estimated correlation matrix of the least squares means as part of the "Least Squares Means" table.

COV

displays the estimated covariance matrix of the least squares means as part of the "Least Squares Means" table.

DF=*number*

specifies the degrees of freedom for the t test and confidence limits. The default is the denominator degrees of freedom taken from the "Type III Tests of Fixed Effects" table corresponding to the LS-means effect. For these DDFM= methods, degrees of freedom are determined separately for each test; see the DDFM= option on page 4935 for more information.

DIFF<=*difftype***>**

PDIFF<=*difftype***>**

requests that differences of the LS-means be displayed. You can specify the following values for the optional *difftype*.

ALL requests all pairwise differences; it is the default.

ANOM requests differences between each LS-mean and the average LS-mean, as in the analysis of means (Ott 1967). The average is computed as a weighted mean of the LS-means, with the weights being inversely proportional to the diagonal entries of the $\mathbf{L}\left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{L}'$ matrix. When a WEIGHT statement is specified, then the preceding matrix is replaced with $\mathbf{L}\left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-}\mathbf{L}'$ where $\mathbf{W}$ is the diagonal matrix that contains the weights. If LS-means are nonestimable, this design-based weighted

mean is replaced with an equally weighted mean. Note that the ANOM procedure in SAS/QC software implements both tables and graphics for the analysis of means with a variety of response types. For one-way designs and normally distributed data, the DIFF=ANOM computations are equivalent to the results of PROC ANOM.

CONTROL requests differences with a control; by default, the control is the first level of each of the specified LSMEANS effects. To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the CONTROL keyword. For example, if the effects A, B, and C are classification variables, each having two levels, 1 and 2, the following LSMEANS statement specifies the (1,2) level of A*B and the (2,1) level of B*C as controls:

```
lsmeans A*B B*C / diff=control('1' '2', '2' '1');
```

For multiple effects, the results depend upon the order of the list, and so you should check the output to make sure that the controls are correct.

CONTROL produces two-tailed tests and confidence limits.

CONTROLL requests one-tailed results and tests whether the noncontrol levels are significantly smaller than the control. The upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing.

CONTROLU requests one-tailed results and tests whether the noncontrol levels are significantly larger than the control. The upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

The differences of the LS-means are displayed in a table titled "Differences of Least Squares Means." The table name is "Diffs."

E

requests that the matrix coefficients for all LSMEANS effects be displayed. The name of this "Matrix Coefficients" table is "Coef."

PDIFF

is the same as the DIFF option. See the description of the DIFF option on page 4933.

SINGULAR=*number*

tunes the estimability checking as documented for the SINGULAR= in the CONTRAST statement.

SLICE=*fixed-effect* | **(***fixed-effects***)**

specifies effects by which to partition interaction LSMEANS effects. This can produce what are known as tests of simple effects (Winer 1971). For example, suppose that A*B is significant, and you want to test the effect of A for each level of B. The appropriate LSMEANS statement is

```
lsmeans A*B / slice=B;
```

This statement tests for the simple main effects of A for B, which are calculated by extracting the appropriate rows from the coefficient matrix for the A*B LS-means and by using them to form an F test.

The SLICE= option produces F tests that test the simultaneous equality of cell means at a fixed level of the slice effect (Schabenberger, Gregoire, and Kong 2000).

The SLICE= option produces a table titled "Tests of Effect Slices." The table name is "Slices."

MODEL Statement

MODEL *dependent* **=** < *fixed-effects* > < / *options* > **;**

The MODEL statement names a single dependent variable and the fixed effects, which determine the **X** matrix of the mixed model. The specification of effects is the same as in the GLM procedure; however, unlike PROC GLM, you do not specify random effects in the MODEL statement. The MODEL statement is required.

An intercept is included in the fixed-effects model by default. If no fixed effects are specified, only this intercept term is fit. The intercept can be removed by using the NOINT option.

You can specify the following *options* in the MODEL statement after a slash (/).

ALPHA=*number*
requests that a t-type confidence interval be constructed for each of the fixed-effects parameters with confidence level $1 - \mathit{number}$. The value of *number* must be between 0 and 1; the default is 0.05.

CL
requests that t-type confidence limits be constructed for each of the fixed-effects parameter estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

DDF=*value-list*
enables you to specify your own denominator degrees of freedom for the fixed effects. The *value-list* specification is a list of numbers or missing values (.) separated by commas. The degrees of freedom should be listed in the order in which the effects appear in the "Type III Tests of Fixed Effects" table. If you want to retain the default degrees of freedom for a particular effect, use a missing value for its location in the list. For example, the following statement assigns 3 denominator degrees of freedom to A and 4.7 to A*B, while those for B remain the same:

```
model Y = A B A*B / ddf=3,.,4.7;
```

DDFM=RESIDUAL | NONE
specifies the method for computing the denominator degrees of freedom for the tests of fixed effects resulting from the MODEL, CONTRAST, ESTIMATE, LSMEANS, and TEST statements.

The DDFM=RESIDUAL option performs all tests by using the residual degrees of freedom, $n - \mathrm{rank}(\mathbf{X})$, where n is the number of observations used. It is the default degrees of freedom method.

DDFM=NONE specifies that no denominator degrees of freedom be applied. PROC HPMIXED then essentially assumes that infinite degrees of freedom are available in the calculation of p-values. The p-values for t tests are then identical to p-values derived from the standard normal distribution. In the case of F tests, the p-values equal those of chi-square tests determined as follows: if F_{obs} is the observed value of the F test with l numerator degrees of freedom, then

$$p = \Pr\{F_{l,\infty} > F_{obs}\} = \Pr\{\chi^2_l > lF_{obs}\}$$

NOINT
requests that no intercept be included in the model. An intercept is included by default.

SOLUTION | S
requests that a solution for the fixed-effects parameters be produced. Using notation from the section "Model Assumptions" on page 4950, the fixed-effects parameter estimates are $\widehat{\boldsymbol{\beta}}$ and their approximate standard errors are the square roots of the diagonal elements of $(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}$.

Along with the estimates and their approximate standard errors, a t statistic is computed as the estimate divided by its standard error. The degree of freedom for this t statistic matches the one appearing in the "Type III Tests of Fixed Effects" table under the effect containing the parameter. The "Pr > |t|" column contains the two-tailed p-value corresponding to the t statistic and associated degrees of freedom.

ZETA=*number*
tunes the sensitivity in forming Type III functions. Any element in the estimable function basis with an absolute value less than *number* is set to 0. The default is 1E–8.

NLOPTIONS Statement

NLOPTIONS < *options* > **;**

For more information about the NLOPTIONS, see the section "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics."

If you choose TECH=NEWRAP, then the default value of LSPRECISION is 0.4 in the HPMIXED procedure.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< *keyword*< **(***keyword-options***)** > < **=***name* > > . . .
< *keyword*< **(***keyword-options***)** > < **=***name* > > < / *options* > **;**

The OUTPUT statement creates a data set that contains predicted values and residual diagnostics, computed after fitting the model. By default, all variables in the original data set are included in the output data set.

You can use the ID statement to select a subset of the variables from the input data set to be added to the output data set.

For example, suppose that the data set Scores contains the variables score, machine, and person. The following statements fit a model with fixed machine and random person effects and save the predicted and residual values to the data set igausout:

```
proc hpmixed data = Scores;
   class machine person score;
   model score = machine;
   random person;
   output out=igausout pred=p resid=r;
run;
```

You can specify the following *options* in the OUTPUT statement before the slash (/).

OUT=*SAS data set*
specifies the name of the output data set. If the OUT= option is omitted, the procedure uses the DATA*n* convention to name the output data set.

keyword < (*keyword-options*) >< =*name* >
specifies a statistic to include in the output data set and optionally assigns the variable the name name. You can use the *keyword-options* to control which type of a particular statistic to compute. The *keyword-options* can take on the following values:

BLUP uses the predictors of the random effects in computing the statistic.

NOBLUP does not use the predictors of the random effects in computing the statistic.

The default is to compute statistics by using BLUPs. For example, the following two OUTPUT statements are equivalent:

```
output out=out1  pred=predicted lcl=lower;
output out=out1  pred(blup)=predicted lcl(blup)=lower;
```

If a particular combination of *keyword* and *keyword-options* is not supported, the statistic is not computed and a message is produced in the SAS log.

A *keyword* can appear multiple times in the OUTPUT statement. Table 62.7 lists the *keywords* and the default names assigned by the HPMIXED procedure if you do not specify a *name*. In this table, y denotes the response variable.

Table 62.7 Keywords for Output Statistics

Keyword	Options	Description	Expression	Name
PREDICTED	**BLUP**	Linear predictor	$\widehat{\eta} = \mathbf{x}'\widehat{\boldsymbol{\beta}} + \mathbf{z}'\widehat{\boldsymbol{\gamma}}$	Pred
	NOBLUP	Marginal linear predictor	$\widehat{\eta}_m = \mathbf{x}'\widehat{\boldsymbol{\beta}}$	PredPA
STDERR	**BLUP**	Standard deviation of linear predictor	$\sqrt{\mathrm{Var}[\widehat{\eta} - \mathbf{z}'\boldsymbol{\gamma}]}$	StdErr
	NOBLUP	Standard deviation of marginal linear predictor	$\sqrt{\mathrm{Var}[\widehat{\eta}_m]}$	StdErrPA
RESIDUAL	**BLUP**	Residual	$r = y - \widehat{\eta}$	Resid
	NOBLUP	Marginal residual	$r_m = y - \widehat{\eta}_m$	ResidPA
PEARSON	**BLUP**	Pearson-type residual	$r/\sqrt{\widehat{\mathrm{Var}}[y\|\boldsymbol{\gamma}]}$	Pearson
	NOBLUP	Marginal Pearson-type residual	$r_m/\sqrt{\widehat{\mathrm{Var}}[y]}$	PearsonPA
STUDENT	**BLUP**	Studentized residual	$r/\sqrt{\widehat{\mathrm{Var}}[r]}$	Student
	NOBLUP	Studentized marginal residual	$r_m/\sqrt{\widehat{\mathrm{Var}}[r_m]}$	StudentPA
LCL	**BLUP**	Lower prediction limit for linear predictor		LCL
	NOBLUP	Lower confidence limit for marginal linear predictor		LCLPA

Table 62.7 (continued)

Keyword	Options	Description	Expression	Name
UCL	**BLUP**	Upper prediction limit for linear predictor		UCL
	NOBLUP	Upper confidence limit for marginal linear predictor		UCLPA
VARIANCE	**BLUP**	Conditional variance of response variable	$\widehat{\mathrm{Var}}[y\|\boldsymbol{\gamma}]$	Variance
	NOBLUP	Marginal variance of response variable	$\widehat{\mathrm{Var}}[y]$	VariancePA

You can use the following shortcuts to request statistics: PRED for PREDICTED, STD for STDERR, RESID for RESIDUAL, VAR for VARIANCE.

You can specify the following *options* of the OUTPUT statement after the slash (/).

ALLSTATS
requests that all statistics are computed. If you do not use a *keyword* to assign a name, the HPMIXED procedure uses the default name.

ALPHA=*number*
determines the coverage probability for two-sided confidence and prediction intervals. The coverage probability is computed as $1 - \mathit{number}$. The value of *number* must be between 0 and 1 inclusively; the default is 0.05.

NOMISS
requests that records from the input data set be written to the output data only for those observations that were used in the analysis. By default, the HPMIXED procedure produces output statistics for all observations in the input data set.

NOUNIQUE
requests that names not be made unique in the case of naming conflicts. By default, the HPMIXED procedure avoids naming conflicts by assigning a unique name to each output variable. If you specify the NOUNIQUE option, variables with conflicting names are not renamed. In that case, the first variable added to the output data set takes precedence.

NOVAR
requests that variables from the input data set not be added to the output data set. This option ignores ID statement but does not apply to variables listed in a BY statement.

PARMS Statement

PARMS < (*value-list*)... > < / *options* > ;

The PARMS statement specifies initial values for the covariance parameters, or it requests a grid search over several values of these parameters. You must specify the values in the order in which they appear in the "Covariance Parameter Estimates" table.

The *value-list* specification can take any of several forms:

m	a single value
$m_1, m_2, \ldots, m_n$	several values
m to n	a sequence where m equals the starting value, n equals the ending value, and the increment equals 1
m to n by i	a sequence where m equals the starting value, n equals the ending value, and the increment equals i
m_1, m_2 to m_3	mixed values and sequences

You can use the PARMS statement to input known parameters. Suppose the three variance components are known to be 2, 1, and 3. The SAS statements to fix the variance components at these values are as follows:

```
proc hpmixed noprofile;
   class Family Gender;
   model Height = Gender;
   random Family Family*Gender;
   parms (2) (1) (3) / noiter;
run;
```

The NOPROFILE option in the PROC HPMIXED statement suppresses profiling the residual variance parameter during its calculations, thereby enabling its value to be held at 3 as specified in the PARMS statement.

If you specify more than one set of initial values, PROC HPMIXED performs a grid search of the likelihood surface and uses the best point on the grid for subsequent analysis. Specifying a large number of grid points can result in long computing times. The grid search feature is also useful for exploring the likelihood surface.

The results from the PARMS statement are the values of the parameters on the specified grid (denoted by CovP1–CovP*n*), the residual variance (possibly estimated) for models with a residual variance parameter, and various functions of the likelihood.

The name of the "Parameter Search" table is ParmSearch.

You can specify the following *options* in the PARMS statement after a slash (/).

HOLD< =ALL | *value-list* >

EQCONS< =ALL | *value-list* >

specifies which parameter values PROC HPMIXED should hold equal to the specified values. If you do not specify *value-list* or simply specify HOLD=ALL, then all covariance parameters are held equal to the specified values. For example, the following statement constrains the first and third covariance parameters to equal 5 and 2, respectively:

```
parms (5) (3) (2) (3) / hold=1,3;
```

Specifying the HOLD= option implies the NOPROFILE option in the PROC HPMIXED statement.

LOWERB=*value-list*

enables you to specify lower boundary constraints on the covariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that PROC HPMIXED uses for the covariance parameters, and each number corresponds to the lower boundary constraint. A missing value instructs PROC HPMIXED to use its default constraint, and if you do not specify numbers for all the covariance parameters, PROC MIXED assumes that the remaining ones are missing.

NOITER

requests that no optimization iterations be performed and that PROC HPMIXED use the best value from the grid search to perform inferences. By default, iterations begin at the best value from the PARMS grid search. This option is ignored when you specify the HOLD= option.

If a residual variance is profiled, the parameter estimates can change from the initial values that you provide as the residual variance is recomputed. To prevent an update of the residual variance, combine the NOITER option with the NOPROFILE option in the PROC HPMIXED statements, as in the following program:

```
proc hpmixed noprofile;
   class A B C rep mp sp;
   model y = A | B | C;
   random rep mp sp;
   parms (180) (200) (170) (1000) / noiter;
run;
```

Specifying the NOITER option in the PARMS statement has the same effect as specifying TECHNIQUE=NONE in the NLOPTIONS statement.

Notice that the NOITER option can be useful if you want to obtain the starting values HPMIXED computes. The following statements produce the starting values:

```
proc hpmixed noprofile;
   class A B;
   model y = A;
   random int / subject=B;
   parms / noiter;
run;
```

PARMSDATA=*SAS-data-set*

PDATA=*SAS data set*

reads in covariance parameter values from a SAS data set. The data set should contain the numerical variable ESTIMATE or the numerical variables Covp1–Covp$q$, where q denotes the number of covariance parameters.

If the PARMSDATA= data set contains multiple sets of covariance parameters, the HPMIXED procedure evaluates the initial objective function for each set and commences the optimization step by

using the set with the lowest function value as the starting values. For example, the following SAS statements request that the objective function be evaluated for three sets of initial values:

```
data data_covp;
   input covp1-covp4;
   datalines;
180 200 170 1000
170 190 160  900
160 180 150  800
;
proc hpmixed;
   class A B C rep;
   model yield = A;
   random rep B C;
   parms / pdata=data_covp;
run;
```

Each set comprises four covariance parameters.

The order of the observations in a data set with the numerical variable Estimate corresponds to the order of the covariance parameters in the "Covariance Parameter Estimates" table.

The PARMSDATA= data set must contain at least one set of covariance parameters with no missing values.

If the HPMIXED procedure is processing the input data set in BY groups, you can add the BY variables to the PARMSDATA= data set. If this data set is sorted by the BY variables, the HPMIXED procedure matches the covariance parameter values to the current BY group. If the PARMSDATA= data set does not contain all BY variables, the data set is processed in its entirety for every BY group and a message is written to the log. This enables you to provide a single set of starting values across BY groups, as in the following statements:

```
data data_covp;
   input covp1-covp4;
   datalines;
180 200 170 1000
;
proc hpmixed;
   class A B C rep;
   model yield = A;
   random rep B C;
   parms / pdata=data_covp;
   by year;
run;
```

The same set of starting values is used for each value of the year variable.

UPPERB=*value-list*

enables you to specify upper boundary constraints on the covariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that PROC HPMIXED uses for the covariance parameters, and each number corresponds to the upper boundary constraint. A missing value instructs PROC HPMIXED to use its default constraint, and if you do not specify numbers for all of the covariance parameters, PROC HPMIXED assumes that the remaining ones are missing.

RANDOM Statement

RANDOM *random-effects* < / *options* > **;**

The RANDOM statement defines the random effects in the mixed model. It can be used to specify traditional variance component models (as in the VARCOMP procedure) and to specify random coefficients. The random effects can be classification or continuous. Multiple RANDOM statements are possible. Random effects specified in a RANDOM statement could be correlated with each other for certain types of covariance structures (see the TYPE= option on page 4943). It is, however, assumed that random effects specified using different RANDOM statements are not correlated.

Using notation from the section "Model Assumptions" on page 4950, the purpose of the RANDOM statement is to define the $\mathbf{Z}$ matrix of the mixed model, the random effects in the $\boldsymbol{\gamma}$ vector, and the structure of $\mathbf{G}$. The $\mathbf{Z}$ matrix is constructed exactly like the $\mathbf{X}$ matrix for the fixed effects, and the $\mathbf{G}$ matrix is constructed to correspond to the effects constituting $\mathbf{Z}$. The structure of $\mathbf{G}$ is defined by using the TYPE= option described on page 4943.

You can specify INTERCEPT (or INT) as a random effect. PROC HPMIXED does not include the intercept in the RANDOM statement by default, as it does in the MODEL statement.

You can specify the following *options* in the RANDOM statement after a slash (/).

ALPHA=*number*

requests that a t-type confidence interval with confidence level $1 - \mathit{number}$ be constructed for the predictors of random effects in this statement. The value of *number* must be between 0 and 1 exclusively; the default is 0.05. Specifying the ALPHA= option implies the CL option.

CL

requests that t-type confidence limits be constructed for each of the predictors of random effects in this statement. The confidence level is 0.95 by default; this can be changed with the ALPHA= option. The CL option implies the SOLUTION option.

GROUP=*effect*

defines an effect specifying heterogeneity in the covariance structure of $\mathbf{G}$. All observations having the same level of the group effect have the same covariance parameters. Each new level of the group effect produces a new set of covariance parameters with the same structure as the original group. You should exercise caution in defining the group effect, because strange covariance patterns can result from its misuse. Also, the group effect can greatly increase the number of estimated covariance parameters, which can adversely affect the optimization process.

Continuous variables are permitted as arguments to the GROUP= option. PROC HPMIXED does not sort by the values of the continuous variable; rather, it considers the data to be from a new

group whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of groups and also prevents the production of a large "Class Levels Information" table.

NOFULLZ

eliminates the columns in $\mathbf{Z}$ corresponding to missing levels of random effects involving CLASS variables. By default, these columns are included in $\mathbf{Z}$. It is sufficient to specify the NOFULLZ option in any RANDOM statement.

SOLUTION

requests that the solution for the random-effects parameters be produced. Using notation from the section "Model Assumptions" on page 4950, these estimates are the empirical best linear unbiased predictors (BLUPs) $\widehat{\boldsymbol{\gamma}} = \widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{V}}^{-1}(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})$. They can be useful for comparing the random effects from different experimental units and can also be treated as residuals in performing diagnostics for your mixed model.

The numbers displayed in the SE Pred column of the "Solution for Random Effects" table are not the standard errors of the $\widehat{\boldsymbol{\gamma}}$ displayed in the Estimate column; rather, they are the standard errors of predictions $\widehat{\boldsymbol{\gamma}}_i - \boldsymbol{\gamma}_i$, where $\widehat{\boldsymbol{\gamma}}_i$ is the ith BLUP and $\boldsymbol{\gamma}_i$ is the ith random-effect parameter.

SUBJECT=*effect*

identifies the subjects in your mixed model. Complete independence is assumed across subjects; thus, for the RANDOM statement, the SUBJECT= option produces a block-diagonal structure in $\mathbf{G}$ with identical blocks. The $\mathbf{Z}$ matrix is modified to accommodate this block-diagonality. In fact, specifying a subject effect is equivalent to nesting all other effects in the RANDOM statement within the subject effect.

Continuous variables are permitted as arguments to the SUBJECT= option. PROC HPMIXED does not sort by the values of the continuous variable; rather, it considers the data to be from a new subject whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of subjects and also prevents the production of a large "Class Levels Information" table.

TYPE=*covariance-structure*

specifies the structure of the covariance matrix $\mathbf{G}$ for random effects. The default structure is VC.

If you want different covariance structures in different parts of $\mathbf{G}$, you must use multiple RANDOM statements with different TYPE= options.

Valid values for *covariance-structure* are listed in Table 62.8. Examples are shown in Table 62.9.

Table 62.8 Covariance Structures

Structure	Description	Parameters	(i, j) element
AR(1)	Autoregressive(1)	2	$\sigma^2\rho^{\lvert i-j\rvert}$
CHOL	Cholesky root	$t(t+1)/2$	l_{ij}
CS	Compound symmetry (CS)	2	$\sigma_1 + \sigma^2 1(i = j)$
CSH	Heterogeneous CS	$t+1$	$\sigma_i\sigma_j[\rho 1(i \neq j) + 1(i = j)]$
TOEP(1)	Toeplitz(1)	1	σ^2

Table 62.8 *continued*

Structure	Description	Parms	(i, j) element
UC	Uniform correlation (UC)	2	$\sigma^2[\rho 1(i \neq j) + 1(i = j)]$
UCH	Heterogeneous UC	$t + 1$	$\sigma_i \sigma_j [\rho 1(i \neq j) + 1(i = j)]$
UN	Unstructured	$t(t + 1)/2$	σ_{ij}
VC	Variance components	q	$\sigma_k^2 1(i = j)$ and *i*,*j* correspond to *k*th effect

In Table 62.8, t is the overall dimension of the covariance matrix, and $1(A)$ equals 1 when A is true and 0 otherwise. For example, $1(i = j)$ equals 1 when $i = j$ and equals 0 otherwise. TYPE=UCH is the same as TYPE=CSH.

Table 62.9 lists some examples of the structures in Table 62.8.

Table 62.9 Covariance Structure Examples

Description	Structure	Example
First-order autoregressive	AR(1)	$\sigma^2 \begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 \\ \rho & 1 & \rho & \rho^2 \\ \rho^2 & \rho & 1 & \rho \\ \rho^3 & \rho^2 & \rho & 1 \end{bmatrix}$
Cholesky root	CHOL	$\begin{bmatrix} l_{11} & 0 & 0 & 0 \\ l_{21} & l_{22} & 0 & 0 \\ l_{31} & l_{32} & l_{33} & 0 \\ l_{41} & l_{42} & l_{43} & l_{44} \end{bmatrix} \begin{bmatrix} l_{11} & l_{21} & l_{31} & l_{41} \\ 0 & l_{22} & l_{32} & l_{42} \\ 0 & 0 & l_{33} & l_{43} \\ 0 & 0 & 0 & l_{44} \end{bmatrix}$
Compound symmetry	CS	$\begin{bmatrix} \sigma^2 + \sigma_1 & \sigma_1 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma^2 + \sigma_1 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma^2 + \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma_1 & \sigma^2 + \sigma_1 \end{bmatrix}$
Banded Toeplitz	TOEP(1)	$\begin{bmatrix} \sigma^2 & 0 & 0 & 0 \\ 0 & \sigma^2 & 0 & 0 \\ 0 & 0 & \sigma^2 & 0 \\ 0 & 0 & 0 & \sigma^2 \end{bmatrix}$
Uniform correlation	UC	$\sigma^2 \begin{bmatrix} 1 & \rho & \rho & \rho \\ \rho & 1 & \rho & \rho \\ \rho & \rho & 1 & \rho \\ \rho & \rho & \rho & 1 \end{bmatrix}$
Heterogeneous UC	UCH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho & \sigma_1\sigma_4\rho \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho \\ \sigma_3\sigma_1\rho & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho & \sigma_4\sigma_2\rho^2 & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$

Table 62.9 *continued*

Description	Structure	Example
Unstructured	UN	$\begin{bmatrix} \sigma_1^2 & \sigma_{21} & \sigma_{31} & \sigma_{41} \\ \sigma_{21} & \sigma_2^2 & \sigma_{32} & \sigma_{42} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 & \sigma_{34} \\ \sigma_{41} & \sigma_{42} & \sigma_{43} & \sigma_4^2 \end{bmatrix}$
Variance components	VC (default)	$\begin{bmatrix} \sigma_A^2 & 0 & 0 & 0 \\ 0 & \sigma_A^2 & 0 & 0 \\ 0 & 0 & \sigma_B^2 & 0 \\ 0 & 0 & 0 & \sigma_B^2 \end{bmatrix}$

The variances and covariances in the formulas that follow in the TYPE= option descriptions are expressed in terms of generic random variables ξ_i and ξ_j. They represent random effects for which the **G** matrices are constructed.

The following list provides some further information about these *covariance-structures*:

AR(1) specifies a first-order autoregressive structure,

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \rho^{|i-j|}$$

The values i and j are derived for the ith and jth observations, respectively. For example, in the following statements the values correspond to the class levels for the time effect of the ith and jth observation within a particular subject:

```
proc hpmixed;
   class time patient;
   model y = x x*x;
   random time / sub=patient type=ar(1);
run;
```

PROC HPMIXED imposes the constraint $|\rho| < 1$ for stationarity.

CHOL specifies an unstructured variance-covariance matrix parameterized through its Cholesky root. All diagonal values are constrained to be positive. This parameterization guarantees a positive definite covariance matrix. For example, a 2×2 unstructured covariance matrix can be written as

$$\mathrm{Var}[\boldsymbol{\xi}] = \begin{bmatrix} \sigma_1^2 & \sigma_{21} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix}$$

Without imposing constraints on the three parameters, there is no guarantee that the estimated variance matrix is positive definite. Even if σ_1^2 and σ_2^2 are nonzero, a large value for σ_{21} can lead to a negative eigenvalue of $\mathrm{Var}[\boldsymbol{\xi}]$. The Cholesky root of a positive definite matrix $\mathbf{A}$ is a lower triangular matrix $\mathbf{L}$ such that $\mathbf{LL}' = \mathbf{A}$. The Cholesky root of the above 2×2 matrix can be written as

$$\mathbf{L} = \begin{bmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{bmatrix}$$

The elements of the unstructured variance matrix are then simply $\sigma_1^2 = l_{11}^2$, $\sigma_{21} = l_{21}l_{11}$, and $\sigma_2^2 = l_{21}^2 + l_{22}^2$. Similar operations yield the generalization to covariance matrices of higher orders.

For example, the following statements model the covariance matrix of each subject as an unstructured matrix:

```
proc hpmixed;
   class sub;
   model y = x;
   random  time / sub=patient type=chol;
run;
```

The HPMIXED procedure constrains the diagonal elements of the Cholesky root to be positive. This guarantees that the structure is positive definite.

CS specifies the compound-symmetry structure, which has constant variance and constant covariance

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 + \sigma_1 & i = j \\ \sigma_1 & i \neq j \end{cases}$$

Under compound-symmetry, the $\mathbf{G}$ matrix is of form $\sigma^2\mathbf{I} + \sigma_1\mathbf{J}$. The variance parameter σ^2 is constrained to be positive, and the covariance parameter σ_1 is constrained to be greater than $-\sigma^2/t$ where t is the dimension of the structure. This guarantees the structure is positive definite. The compound-symmetry structure arises naturally with nested random effects, such as when a subsampling error is nested within an experimental error.

CSH specifies the heterogeneous compound-symmetry structure. This structure has a different variance parameter for each diagonal element, and it uses the square roots of these parameters in the off-diagonal entries. In Table 62.8, σ_i^2 is the ith variance parameter that satisfies $\sigma_i^2 > 0$, and ρ is the correlation parameter that satisfies $\rho > -1/(t-1)$, where t is the dimension of the structure. This guarantees that the structure is positive definite.

TOEP(1) specifies a Toeplitz structure with one band. It is the same as $\sigma^2 I$, where I is an identity matrix, and it can be useful for specifying the same variance component for several effects.

UC specifies the uniform correlation structure, which has constant variance and constant correlation

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \begin{cases} \sigma^2 & i = j \\ \sigma^2\rho & i \neq j \end{cases}$$

Under uniform correlation, the $\mathbf{G}$ matrix is of form $\sigma^2[(1-\rho)\mathbf{I} + \rho\mathbf{J}]$. The variance σ^2 is constrained to be positive, and the correlation ρ is constrained to be greater than $-1/(t-1)$, where t is the dimension of the structure. This guarantees the structure is positive definite. This structure is equivalent to the compound-symmetry structure with a better numerical property in terms of optimization.

The uniform correlation structure arises frequently in agriculture and animal sciences.

UCH specifies the heterogeneous uniform correlation structure. This structure has a different variance parameter for each diagonal element, and it uses the square roots of these parameters in the off-diagonal entries. In Table 62.8, σ_i^2 is the ith variance parameter that satisfies $\sigma_i^2 > 0$, and ρ is the correlation parameter that satisfies $\rho > -1/(t-1)$, where t is the dimension of the structure. This guarantees that the structure is positive definite.

UN specifies a completely general (unstructured) covariance matrix parameterized directly in terms of variances and covariances. The variances are constrained to be positive, and the covariances are unconstrained. In addition, this structure is internally constrained to be positive definite.

VC specifies standard variance components and is the default structure for the RANDOM and REPEATED statements. In the RANDOM statement, a distinct variance component is assigned to each effect. In the REPEATED statement, this structure is usually used only with the GROUP= option to specify a heterogeneous variance model.

REPEATED Statement

REPEATED *repeated-effect* < / *options* > **;**

The REPEATED statement defines the repeated effect and the residual covariance structure in the mixed model. The residual variance-covariance matrix is denoted as **R**. The *repeated-effect* is required and consists entirely of classification variables. The levels of the *repeated-effect* must be different for each observation within a subject in order to avoid the singular **R** matrix. The SUBJECT= option is required. The data set must be grouped by subject effect.

Table 62.10 summarizes the *options* available in the REPEATED statement.

Table 62.10 Summary of REPEATED Statement Options

Option	Description
Construction of Covariance Structure	
GROUP=	Defines an effect that specifies heterogeneity in the residual covariance structure
SUBJECT=	Identifies the subjects in the residual covariance structure
TYPE=	Specifies the residual covariance structure (the default is VC)
Statistical Output	
R=	Displays blocks of the estimated **R** matrix
RC=	Display the Cholesky root (lower) of blocks of the estimated **R** matrix
RCI=	Displays the inverse Cholesky root (lower) of blocks of the estimated **R** matrix
RCORR=	Displays the correlation matrix that corresponds to blocks of the estimated **R** matrix
RI=	Displays the inverse of blocks of the estimated **R** matrix

You can specify the following *options* in the REPEATED statement after a slash (/).

GROUP=*effect*

GRP=*effect*

defines an effect that specifies heterogeneity in the residual covariance structure. All observations that have the same level of the GROUP effect have the same covariance parameters. Each new level of the GROUP effect produces a new set of covariance parameters with the same structure as the original group. You should exercise caution in defining the GROUP effect, because strange covariance patterns can result with its misuse. Also, the GROUP effect can greatly increase the number of estimated covariance parameters, which can adversely affect the optimization process.

Continuous variables are permitted as arguments to the GROUP= option. PROC HPMIXED does not sort by the values of the continuous variable; rather, it considers the data to be from a new subject or group whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of subjects or groups and also prevents the production of a large "Class Level Information" table.

R< **=***value-list*>

requests that blocks of the estimated **R** matrix be displayed. The first block determined by the SUBJECT= effect is the default displayed block.

The *value-list* indicates the subjects for which blocks of **R** are to be displayed. For example, the following statement displays block matrices for the first, third, and fifth persons:

```
repeated time / type=un subject=person r=1,3,5;
```

See the PARMS statement for the possible forms of *value-list*.

RC< **=***value-list*>

displays the Cholesky root of blocks of the estimated **R** matrix. The *value-list* specification is the same as for the R= option.

RCI< **=***value-list*>

displays the inverse Cholesky root of blocks of the estimated **R** matrix. The *value-list* specification is the same as for the R= option.

RCORR< **=***value-list*>

displays the correlation matrix that corresponds to blocks of the estimated **R** matrix. The *value-list* specification is the same as for the R= option.

RI< **=***value-list*>

produces the inverse of blocks of the estimated **R** matrix. The *value-list* specification is the same as for the R= option.

SUBJECT=*effect*

SUB=*effect*

identifies the subjects in your mixed model. Complete independence is assumed across subjects; therefore, the SUBJECT= option produces a block-diagonal structure in **R** with identical blocks. The SUBJECT= option is required. The data set must be grouped by SUBJECT= effect. When the SUBJECT= effect consists entirely of classification variables, the blocks of **R** correspond to

observations that share the same level of that effect. These blocks are sorted according to this effect as well.

Continuous variables are permitted as arguments to the SUBJECT= option. PROC HPMIXED does not sort by the values of the continuous variable; rather, it considers the data to be from a new subject or group whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of subjects or groups and also prevents the production of a large "Class Level Information" table.

If you want to model nonzero covariance among all of the observations in your data, specify SUBJECT=INTERCEPT to treat the data as if they are all from one subject. However, be aware that in this case PROC HPMIXED manipulates an **R** matrix with dimensions equal to the number of observations.

TYPE=*covariance-structure*
specifies the structure of the residual variance-covariance matrix **R**. The SUBJECT= option defines the blocks of **R**, and the TYPE= option specifies the structure of these blocks. PROC HPMIXED supports the following structures: TYPE=AR(1), TYPE=CHOL, TYPE=UN, and TYPE=VC. The default structure is VC. See the description in the section "RANDOM Statement" on page 4942 for more information about these covariance structure types.

TEST Statement

TEST *fixed-effects* < / *options* > **;**

The TEST statement performs a hypothesis test on the fixed effects. You can specify multiple effects in one TEST statement or in multiple TEST statements, and all TEST statements must appear after the MODEL statement.

You can specify the following *options* after a slash (/):

CHISQ
performs χ^2 tests in addition to any F tests. A χ^2 statistic equals its corresponding F statistic times the associated numerator degrees of freedom, and PROC HPMIXED uses the same degrees of freedom to compute the p-value for the χ^2 test. This p-value is always less than that for the F test, because a χ^2 test effectively corresponds to an F test that has infinite denominator degrees of freedom.

DF=*number*
specifies the denominator degrees of freedom for the F test. The default is the residual degrees of freedom that you define by specifying the DDFM=RESIDUAL option.

E
displays matrix coefficients that are associated with test types for specified effects.

E3
EIII
displays Type III matrix coefficients if a Type III test is performed.

HTYPE=*value-list*

indicates the type of hypothesis test to perform on the specified effects. The only valid value for entries in the *value-list* is 3, corresponding to a Type III test. The default value is 3. The ODS table name is Tests3 for the Type III test.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement replaces $\mathbf{R}$ with $\mathbf{W}^{-1/2}\mathbf{R}\mathbf{W}^{-1/2}$, where $\mathbf{W}$ is a diagonal matrix containing the weights. Observations with nonpositive or missing weights are not included in the resulting PROC HPMIXED analysis. If you omit the WEIGHT statement, all observations that you use in the analysis are assigned a weight of 1.

If a computation in PROC HPMIXED involves $\mathbf{R}$, then the WEIGHT statement replaces $\mathbf{R}$ with $\mathbf{W}^{-1/2}\mathbf{R}\mathbf{W}^{-1/2}$. For example, the covariance matrix $\mathbf{V}$ for the observations usually has the form $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$; therefore, when you specify the WEIGHT statement, this becomes $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{W}^{-1/2}\mathbf{R}\mathbf{W}^{-1/2}$.

Details: HPMIXED Procedure

Model Assumptions

The following sections provide an overview of the approach used by the HPMIXED procedure for likelihood-based analysis of linear mixed models with sparse matrix technique. Additional theory and examples are provided in Littell et al. (1996); Verbeke and Molenberghs (1997, 2000); Brown and Prescott (1999).

The HPMIXED procedure fits models generally of the form

$$\mathrm{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

Models of this form contain both fixed-effects parameters, $\boldsymbol{\beta}$, and random-effects parameters, $\boldsymbol{\gamma}$; hence, they are called *mixed models*. See Henderson (1990) and Searle, Casella, and McCulloch (1992) for historical developments of the mixed model. Note that the matrix $\mathbf{Z}$ can contain either continuous or dummy variables, just like $\mathbf{X}$.

So far this is the same general form of model fit by the MIXED procedure. The difference between the models handled by the two procedures lies in the assumptions about the distributions of $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$. For both procedures a key assumption is that $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$ are normally distributed with

$$\begin{aligned} \mathrm{E}\begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\epsilon} \end{bmatrix} &= \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix} \\ \mathrm{Var}\begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\epsilon} \end{bmatrix} &= \begin{bmatrix} \mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{bmatrix} \end{aligned}$$

The two procedures differ in their assumptions about the variance matrices $\mathbf{G}$ and $\mathbf{R}$ for $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$, respectively. The MIXED procedure allows a variety of different structures for both $\mathbf{G}$ and $\mathbf{R}$; while in HPMIXED

procedure, $\mathbf{R}$ is always assumed to be of the form $\mathbf{R} = \mathbf{I}\sigma^2$, and the structures available for modeling $\mathbf{G}$ are only a small subset of the structures offered by the MIXED procedure.

Estimates of fixed effects and predictions for random effects are obtained by solving the so-called *mixed model equations*:

$$\begin{bmatrix} \mathbf{X}'\mathbf{X}/\sigma^2 & \mathbf{X}'\mathbf{Z}/\sigma^2 \\ \mathbf{Z}'\mathbf{X}/\sigma^2 & \mathbf{Z}'\mathbf{Z}/\sigma^2 + \mathbf{G}^{-1} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{y}/\sigma^2 \\ \mathbf{Z}'\mathbf{y}/\sigma^2 \end{bmatrix}$$

Let $\mathbf{C}$ denote the coefficient matrix of the mixed model equations:

$$\mathbf{C} = \begin{bmatrix} \mathbf{X}'\mathbf{X}/\sigma^2 & \mathbf{X}'\mathbf{Z}/\sigma^2 \\ \mathbf{Z}'\mathbf{X}/\sigma^2 & \mathbf{Z}'\mathbf{Z}/\sigma^2 + \mathbf{G}^{-1} \end{bmatrix}$$

Under the assumptions given previously for the moments of $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$, the variance of $\mathbf{y}$ is $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{I}\sigma^2$. You can model $\mathbf{V}$ by setting up the random-effects design matrix $\mathbf{Z}$ and by specifying covariance structures for $\mathbf{G}$. Let $\boldsymbol{\theta}$ be a vector of all unknown parameters in $\mathbf{G}$. Then the general form of the restricted likelihood function for the mixed models that the HPMIXED procedure can fit is

$$L(\boldsymbol{\theta}, \sigma^2) = -2\log l = (n-p)\log(2\pi) + \log|\mathbf{C}| + \log|\mathbf{G}| + n\log(\sigma^2) + \mathbf{y}'\mathbf{P}\mathbf{y}$$

where

$$\mathbf{P} = \mathbf{V}^{-1} - \mathbf{V}^{-1}\mathbf{X}(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}^{-1}$$

and p is the rank of $\mathbf{X}$. The HPMIXED procedure minimizes $L(\boldsymbol{\theta}, \sigma^2)$ over all unknown parameters in $\boldsymbol{\theta}$ and σ^2 by using nonlinear optimization algorithms.

Computing and Maximizing the Likelihood

In computing the restricted likelihood function given previously, the determinants of the matrices $\mathbf{C}$ and $\mathbf{G}$ can be obtained effectively by using Cholesky decomposition. The quadratic term $\mathbf{y}'\mathbf{P}\mathbf{y}$ can be expressed in terms of solutions of mixed model equations as follows:

$$\mathbf{y}'\mathbf{P}\mathbf{y} = \frac{1}{\sigma^2}\left(\mathbf{y}'\mathbf{y} - \left[\widehat{\boldsymbol{\beta}}', \widehat{\boldsymbol{\gamma}}'\right]\begin{bmatrix} \mathbf{X}'\mathbf{y} \\ \mathbf{Z}'\mathbf{y} \end{bmatrix}\right)$$

By default, the HPMIXED procedure profiles out the residual variance σ^2 from the parameter vector $\boldsymbol{\theta}$. Let $\boldsymbol{\theta}^*$ be the new parameter vector such that $\theta_i^* = \theta_i/\sigma^2$. The profiled objective function becomes

$$L(\boldsymbol{\theta}^*, \sigma^2) = (n-p)\log(2\pi) + \log|\mathbf{C}^*| + \log|\mathbf{G}^*| - (r_C - r_G - n)\log(\sigma^2) + (n-p)$$

where $\mathbf{C}^* = \mathbf{C}\sigma^2$ and $\mathbf{G}^* = \mathbf{G}\sigma^2$ are the profiled versions of $\mathbf{C}$ and $\mathbf{G}$, r_C and r_G are the ranks of $\mathbf{C}$ and $\mathbf{G}$. Minimizing analytically for σ^2 yields

$$\widehat{\sigma}^2 = \frac{1}{n-p}\left(\mathbf{y}'\mathbf{y} - \left[\widehat{\boldsymbol{\beta}}', \widehat{\boldsymbol{\gamma}}'\right]\begin{bmatrix} \mathbf{X}'\mathbf{y} \\ \mathbf{Z}'\mathbf{y} \end{bmatrix}\right)$$

Optimizing the likelihood calls for derivatives with respect to the parameters. The first and second derivatives of the log-likelihood function L with respect to scalar variance components θ_i and θ_j are

$$\frac{\partial L}{\partial \theta_i} = \mathrm{tr}\left(\frac{\partial \mathbf{V}}{\partial \theta_i}\mathbf{P}\right) - \mathbf{y}'\mathbf{P}\frac{\partial \mathbf{V}}{\partial \theta_i}\mathbf{P}\mathbf{y}$$

and

$$\frac{\partial^2 L}{\partial\theta_i\theta_j} = -\mathrm{tr}\left(\frac{\partial\mathbf{V}}{\partial\theta_i}\mathbf{P}\frac{\partial\mathbf{V}}{\partial\theta_j}\mathbf{P}\right) + 2\mathbf{y}'\mathbf{P}\frac{\partial\mathbf{V}}{\partial\theta_i}\mathbf{P}\frac{\partial\mathbf{V}}{\partial\theta_j}\mathbf{P}\mathbf{y}$$

The default quasi-Newton method of optimization for the HPMIXED procedure requires only first derivatives of the log likelihood, and these are readily derived by solving the mixed model equations. For example, when $\mathbf{G} = \mathbf{I}\sigma_a$, the first derivative of the log likelihood with respect to the parameter σ_a^2 can be computed as follows:

$$\frac{\partial L}{\partial\sigma_a^2} = \frac{q}{\sigma_a^2} - \frac{\mathrm{tr}(\mathbf{C}^{aa})}{\sigma_a^4} - \frac{\widehat{\boldsymbol{\gamma}}'\widehat{\boldsymbol{\gamma}}}{\sigma_a^4}$$

where q is the size of $\boldsymbol{\gamma}$ vector and $\mathbf{C}^{aa}$ is the part of the g-inverse of the mixed model equation coefficient matrix $\mathbf{C}$ corresponding to the random effect $\boldsymbol{\gamma}$.

The second derivative of the log likelihood needs to be computed only if you specify certain nondefault optimization techniques in the NLOPTIONS statement, namely TECH=NEWRAP, TECH=NRRIDG, or TECH=TRUREG; see "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics," for more information about optimization techniques. For these second-derivative-based optimization techniques, the HPMIXED procedure does not actually use the true second derivative matrix, or *observed information matrix*, as defined earlier. Instead, it uses an alternative matrix that is more efficient to compute for large problems and that can be more stable. This alternative is called the *average information* matrix, and it is defined as follows. The expected value of the second derivative is

$$\mathbf{E}(\frac{\partial^2 L}{\partial\theta_i\theta_j}) = \mathrm{tr}\left(\frac{\partial\mathbf{V}}{\partial\theta_i}\mathbf{P}\frac{\partial\mathbf{V}}{\partial\theta_j}\mathbf{P}\right)$$

It is this trace that is computationally inefficient to evaluate. But if you average the expected information matrix defined by this formula with the observed information matrix defined by the preceding formula for the true second derivative, then the trace term cancels, leaving just a quadratic expression in $\mathbf{y}$. This quadratic expression defines the average information (Johnson and Thompson 1995) with respect to θ_i and θ_j:

$$\mathrm{AI}(\theta_i, \theta_j) = \mathbf{y}'\mathbf{P}\frac{\partial\mathbf{V}}{\partial\theta_i}\mathbf{P}\frac{\partial\mathbf{V}}{\partial\theta_j}\mathbf{P}\mathbf{y}$$

Computing Starting Values by EM-REML

The EM-REML algorithm (Dempster, Laird, and Rubin 1977) iteratively alternates between an expectation step and a maximization step to maximize the restricted log likelihood. The algorithm is based on augmenting the observed data $\mathbf{y}$ with the unobservable random effects $\boldsymbol{\gamma}$, leading to a simplified form for the log likelihood. For example, if $\mathbf{G} = \mathbf{I}\sigma_a^2$ then given the realized values $\tilde{\boldsymbol{\gamma}}$ of the unobservable random effects $\boldsymbol{\gamma}$, the REML estimate of σ_a^2 satisfies

$$\widehat{\sigma}_a^2 = \frac{\tilde{\boldsymbol{\gamma}}'\tilde{\boldsymbol{\gamma}}}{q - \sigma^2/\sigma_a^2\mathrm{tr}(\mathbf{C}^{aa})}$$

This corresponds to the maximization step of EM-REML. However, the true realized values $\tilde{\boldsymbol{\gamma}}$ are unknown in practice. The expectation step of EM-REML replaces them with the conditional expected values $\widehat{\boldsymbol{\gamma}}$ of the

random effects, given the observed data $\mathbf{y}$ and initial values for the parameters. The new estimate of σ_a^2 is used in turn to recalculate the conditional expected values, and the iteration is repeated until convergence.

It is well known that EM-REML is generally more robust against a poor choice of starting values than general nonlinear optimization methods such as Newton-Raphson, though it tends to converge slowly as it approaches the optimum. The Newton-Raphson method, on the other hand, converges much faster when it has a good set of starting values. The HPMIXED procedure, thus, employs a scheme that uses EM-REML initially in order to get good starting values, and after a few iterations, when the decrease in log likelihood has significantly slowed down, switching to a more general nonlinear optimization technique (by default, quasi-Newton).

Sparse Matrix Techniques

A key component of the HPMIXED procedure is the use of sparse matrix techniques for computing and optimizing the likelihood expression given in the section "Model Assumptions" on page 4950. There are two aspects to sparse matrix techniques, namely, sparse matrix storage and sparse matrix computations. Typically, computer programs represent an $N \times M$ matrix in a dense form as an array of size NM, making row-wise and column-wise arithmetic operations particularly efficient to compute. However, if many of these NM numbers are zeros, then correspondingly many of these operations are unnecessary or trivial. Sparse matrix techniques exploit this fact by representing a matrix not as a complete array, but as a set of nonzero elements and their location (row and column) within the matrix. Sparse matrix techniques are more efficient if there are enough zero-element operations in the dense form to make the extra time required to find and operate on matrix elements in the sparse form worthwhile.

The following discussion illustrates sparse techniques. Let the symmetric matrix $\mathbf{C}$ be the matrix of mixed model equations of size 5×5.

$$\mathbf{C} = \begin{bmatrix} 8.0 & 0 & 0 & 2.0 & 0 \\ 0 & 4.0 & 3.0 & 0 & 0 \\ 0 & 3.0 & 5.0 & 0 & 0 \\ 2.0 & 0 & 0 & 7.0 & 0 \\ 0 & 0 & 0 & 0 & 9.0 \end{bmatrix}$$

There are 15 elements in the upper triangle of $\mathbf{C}$, though eight of them are zeros. The row and column indices and the values of seven nonzero elements are listed as follows:

i	1	1	2	2	3	4	5
j	1	4	2	3	3	4	5
C_{ij}	8.0	2.0	4.0	3.0	5.0	7.0	9.0

The most elegant scheme to store these seven elements is to store them in a hash table with row and column indices as a hash key. However, this scheme is not efficient as the number of non-zero elements gets very large. The classical and widely used scheme, and the one the HPMIXED procedure employs, is the $(\mathbf{i}_c, \mathbf{j}_c, c)$ format, in which the nonzero elements are stored contiguously row by row in the vector c. To identify the individual nonzero elements in each row, you need to know the column index of an element. These column indices are stored in the vector $\mathbf{j}_c$; that is, if $c(k) = C_{ij}$, then $\mathbf{j}_c(k) = j$. To identify the individual rows, you need to know where each row starts and ends. These row starting positions are stored in the vector $\mathbf{i}_c$. For instance, if C_{ij} is the first nonzero element in the row i and $c(k) = C_{ij}$, then $\mathbf{i}_c(i) = k$. The row i ending

position is one less than $\mathbf{i}_c(i+1)$. Thus, the number of nonzero elements in the row i is $\mathbf{i}_c(i+1) - \mathbf{i}_c(i)$, these elements in the row i are stored consecutively starting from the position $k_i = \mathbf{i}_c(i)$

$$c(k_i), c(k_i+1), c(k_i+2), \ldots, c(k_{i+1}-1)$$

and the corresponding columns indices are stored consecutively in

$$\mathbf{j}_c(k_i), \mathbf{j}_c(k_i+1), \mathbf{j}_c(k_i+2), \ldots, \mathbf{j}_c(k_{i+1}-1)$$

For example, the seven nonzero elements in matrix $\mathbf{C}$ are stored in $(\mathbf{i}_c, \mathbf{j}_c, c)$ format as

$\mathbf{i}_c$	1	3	5	6	7	8	
$\mathbf{j}_c$	1	4	2	3	3	4	5
c	8.0	2.0	4.0	3.0	5.0	7.0	9.0

Note that since matrices are stored row by row in the $(\mathbf{i}_c, \mathbf{j}_c, c)$ format, row-wise operations can be performed efficiently but it is inefficient to retrieve elements column-wise. Thus, this representation will be inefficient for matrix computations requiring column-wise operations. Fortunately, the likelihood calculations for mixed models can usually avoid column-wise operations.

In mixed models, sparse matrices typically arise from a large number of levels for fixed effects and/or random effects. If a linear model contains one or more large CLASS effects, then the mixed model equations are usually very sparse. Storing zeros in mixed model equations not only requires significantly more memory but also results in longer execution time and larger rounding error. As an illustration, the example in the "Getting Started: HPMIXED Procedure" on page 4914 has 3,506 mixed model equations. Storing just the upper triangle of these equations in a dense form requires $(1 + 3{,}506) \times 3{,}506/2 = 6{,}147{,}771$ elements. However, there are only 60,944 nonzero elements—less than 1% of what dense storage requires.

Note that as the density of the mixed model equations increases, the advantage of sparse matrix techniques decreases. For instance, a classical regression model typically has a dense coefficient matrix, though the dimension of the matrix is relatively small.

The HPMIXED procedure employs sparse matrix techniques to store the nonzero elements in the mixed model equations and to compute a sparse Cholesky decomposition of these equations. A reordering of the mixed model equations is required in order to keep the minimum memory consumption during the factorization. This reordering process results in a different g-inverse from what is produced by most other SAS/STAT procedures, for which the g-inverse is defined by sequential sweeping in the order defined by the model. If mixed model equations are singular, this different g-inverse produces a different solution of mixed model equations. However, estimable functions and tests based on them are invariant to the choice of g-inverse, and are thus the same for the HPMIXED procedure as for other procedures.

Hypothesis Tests for Fixed Effects

Unlike most other SAS/STAT procedures for analyzing general linear models, the HPMIXED procedure does not by default provide F tests for the fixed effects. This is because, for the large mixed model problems that the HPMIXED procedure is designed to address, such tests are often computationally prohibitive to compute. The computation of Type III tests first constructs the Hermite matrix of the mixed model coefficient matrix $\mathbf{C}$ and then forms the $\mathbf{L}$ coefficient matrix to obtain the F value as follows:

$$F = \frac{\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}' \mathbf{L}'(\mathbf{L}\widehat{\mathbf{C}}^{-1}\mathbf{L}')^{-1}\mathbf{L} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}}{r}$$

where $r = \mathrm{rank}(\mathbf{L}\widehat{\mathbf{C}}^{-1}\mathbf{L}')$. The coefficient matrix $\mathbf{L}$ corresponding to fixed effects with many levels can be very large and dense, making them very difficult to work with. At the same time, Type III tests for effects with many levels are relatively unlikely to be statistically useful.

For this reason, you must use the TEST statement in PROC HPMIXED to specifically ask for Type III tests for any effects for which you want to compute them. An example of this is given in the section "Getting Started: HPMIXED Procedure" on page 4914.

Default Output

The following sections describe the output PROC HPMIXED produces by default. This output is organized into various tables, and they are discussed in order of appearance.

Model Information

The "Model Information" table describes the model, some of the variables it involves, and the method used in fitting it. It also lists the method for computing the degrees of freedom.

The name of the "Model Information" table is ModelInfo.

Class Level Information

The "Class Level Information" table lists the first 20 levels of every variable specified in the CLASS statement. You should check this information to make sure the data are correct. You can adjust the order of the CLASS variable levels with the ORDER= option in the PROC HPMIXED statement. The name of the "Class Level Information" table is ClassLevels.

Dimensions

The "Dimensions" table lists the sizes of relevant matrices. This table can be useful in determining CPU time and memory requirements. The name of the "Dimensions" table is "Dimensions."

Number of Observations

The "Number of Observations" table shows the number of observations read from the data set and the number of observations used in fitting the model.

Descriptive Statistics

The "Descriptive Statistics" table lists simple statistics such as means and standard deviations for the dependent variable, for each covariate in the MODEL statement, and for the weight variable in the WEIGHT statement.

Iteration History

The "Iteration History" table describes the optimization of the residual log likelihood. The function to be minimized (the *objective function*) is $-2l$.

The name of the "Iteration History" table is IterHistory.

Covariance Parameter Estimates

The "Covariance Parameter Estimates" table contains the estimates of the parameters in **G** and **R**. Their values are labeled in the "Cov Parm" table along with Subject and Group information if applicable. The estimates are displayed in the Estimate column.

The name of the "Covariance Parameter Estimates" table is CovParms.

Convergence Status

The "Convergence Status" table contains a status message that describes the reason the optimization terminated. The message is also written to the log. The name of the "Convergence Status" table is ConvergenceStatus. You can query the nonprinting numeric variable Status to check for a successful optimization. This is useful in batch processing, or when processing BY groups, such as in simulations. Successful optimizations are indicated by the value 0 for the Status variable.

Fit Statistics

The "Fit Statistics" table provides some statistics about the estimated mixed model.

In addition, the "Fit Statistics" table lists three information criteria: AIC, AICC, and BIC, all in smaller-is-better form. Expressions for these criteria are described under the IC= option on page 4920.

The name of the "Model Fitting Information" table is FitStatistics.

ODS Table Names

Each table created by PROC HPMIXED has a name associated with it, and you must use this name to reference the table when using ODS statements. These names are listed in Table 62.11.

Table 62.11 ODS Tables Produced by PROC HPMIXED

Table Name	Description	Required Statement / Option
CholR	Cholesky root of blocks of the estimated **R** matrix	REPEATED / RC
ClassLevels	Level information from the CLASS statement	Default output
Coef	**L** matrix coefficients	E option in TEST, CONTRAST, ESTIMATE, or LSMEANS
Contrasts	Results from the CONTRAST statements	CONTRAST
ConvergenceStatus	Convergence status	Default
CovParms	Estimated covariance parameters	Default output
Diffs	Differences of LS-means	LSMEANS / DIFF (or PDIFF)
Dimensions	Dimensions of the model	Default output
Estimates	Results from ESTIMATE statements	ESTIMATE
FitStatistics	Fit statistics	Default
InvCholR	Inverse Cholesky root of blocks of the estimated **R** matrix	REPEATED / RCI=
InvR	Inverse of blocks of the estimated **R** matrix	REPEATED / RI=
IterHistory	Iteration history	Default output
LSMeans	LS-means	LSMEANS
MMEq	Mixed model equations	PROC HPMIXED MMEQ
ModelInfo	Model information	Default output
NObs	Number of observations read and used	Default output
OptInfo	Optimization information	Default output
OverallANOVA	ANOVA table for model without random effect	Default output for fixed models
ParameterEstimates	Fixed-effects solution	MODEL / SOLUTION
ParmSearch	Parameter search values	PARMS
Ranks	Ranks of matrices **X**, (**XZ**), and **MMEQ**	PROC HPMIXED RANKS
R	Blocks of the estimated **R** matrix	REPEATED / R=
RCorr	Correlation matrix from blocks of the estimated **R** matrix	REPEATED / RCORR=
SimpleStatistics	Descriptive statistics for dependent variable and covariate variables	PROC HPMIXED / SIMPLE
Slices	Tests of LS-means slices	LSMEANS / SLICE=

Table 62.11 *continued*

Table Name	Description	Required Statement / Option
SolutionR	Random-effect solution vector	RANDOM / SOLUTION
Tests3	Type III tests of fixed effects	TEST

Examples: HPMIXED Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 62.1: Ranking Many Random-Effect Coefficients

(View the complete code for this example (hpmex1.sas) in the example repository.)

In analyzing models with random effects that have many levels, a frequent goal is to estimate and rank the predicted values of the coefficients corresponding to these levels. For example, in mixed models for animal breeding, the predicted coefficient of the random effect for each animal is referred to as the *estimated breeding value* (EBV) and animals with relatively high EBVs are chosen for breeding. This example demonstrates the use of the HPMIXED procedure for computing EBVs and their precision. Although other mixed modeling tools in SAS/STAT can potentially compute EBVs, PROC HPMIXED is particularly suited for the large, sparse matrix calculations involved. The typical performance of the HPMIXED procedure and other tools for this problem is also discussed.

The data for this problem are generated by simulation. Suppose you are considering analyzing EBVs for animals on 15 farms, with about 100 animals of 5 different species on each farm. The following DATA step simulates data with this structure, where about 40 observations of the response variable Yield are made per animal:

```
%let NFarm = 15;
%let NAnimal = %eval(&NFarm*100);
data Sim;
   keep Species Farm Animal Yield;
   array BV{&NAnimal};
   array AnimalSpecies{&NAnimal};
   array AnimalFarm{&NAnimal};
   do i = 1 to &NAnimal;
      BV           {i} = sqrt(4.0)*rannor(12345);
      AnimalSpecies{i} = 1 + int(   5  *ranuni(12345));
      AnimalFarm   {i} = 1 + int(&NFarm*ranuni(12345));
   end;
   do i = 1 to 40*&NAnimal;
      Animal  = 1 + int(&NAnimal*ranuni(12345));
      Species = AnimalSpecies{Animal};
      Farm    = AnimalFarm   {Animal};
      Yield   = 1 + Species
```

```
                    + Farm
                    + BV{Animal}
                    + sqrt(8.0)*rannor(12345);
      output;
   end;
run;
```

In this simulation, the true breeding value for each animal (BV1–BV1500) has a variance component of 4.0, while the level of background variance is 8.0.

In this type of experiment, the effect of Species and the interaction between Species and Farm are typically modeled as fixed effects, while the effect of Animal is modeled as a random effect. The following statements use the HPMIXED procedure to compute predictions for the Animal random effect and save them to the data set EBV. This data set is then sorted and the 10 animals with the highest EBVs are displayed.

```
ods exclude all;
proc hpmixed data=Sim;
   class Species Farm Animal;
   model Yield = Species Farm*Species;
   random Animal/cl;
   ods output SolutionR=EBV;
run;
ods exclude none;

proc sort data=EBV;
   by descending estimate;
run;
proc print data=EBV(obs=10) noobs;
   var Animal Estimate StdErrPred Lower Upper;
run;
```

The preceding statements close the ODS all destinations for the duration of the PROC HPMIXED run. This avoids displaying the long random-effects solution table, since only the top few EBVs are of interest. Output 62.1.1 displays the EBVs of the top 10 animals, along with their precision and confidence bounds.

Output 62.1.1 Estimated Breeding Values: Top 10 Animals

Animal	Estimate	StdErrPred	Lower	Upper
1294	5.9703	0.6317	4.7321	7.2085
1219	5.0081	0.6396	3.7544	6.2618
1054	4.9452	0.5874	3.7939	6.0966
758	4.9340	0.6196	3.7195	6.1485
986	4.9329	0.5767	3.8025	6.0633
1150	4.7444	0.5806	3.6064	5.8824
962	4.6651	0.5794	3.5294	5.8008
225	4.5294	0.6137	3.3266	5.7322
1252	4.5012	0.5686	3.3868	5.6157
1033	4.4971	0.6080	3.3054	5.6889

Notice that animal 1294 is ranked as the top animal based on its EBV, but the precision of this estimate, as measured by the standard error of prediction, is lower than that of other animals.

You can also use PROC MIXED and PROC GLIMMIX to compute EBVs, but the performance of these

general mixed modeling procedures for this specialized kind of data and model is quite different from that of PROC HPMIXED. The MIXED and GLIMMIX procedures are engineered to have good performance properties across a broad class of models and analyses, a class much broader than what PROC HPMIXED can handle. The HPMIXED procedure, on the other hand, can have better performance, in terms of both memory and run time, for certain specialized models and analyses, of which the current example is one.

For this example, an equivalent PROC GLIMMIX approach can take twice as long to complete, and PROC MIXED three times as long. Precise relative timings are not feasible, since those of the MIXED and GLIMMIX procedures are sensitive to the speed of disk access for writing to and reading from the utility file that holds the underlying matrices. But the results on any system would be similar: for the limited class of models to which it applies, the sparse matrix representation that the HPMIXED procedure employs should provide better computational performance than a dense representation, in terms of both run time and memory use.

Moreover, for a given analysis, if the size of the problem is increased in such a way that the underlying matrices become sparser, the relative performance of PROC HPMIXED gets even better. As an illustration of this, Output 62.1.2 shows relative performance of the three procedures for simulated data as the number of farms increases. For this plot, each additional farm adds 500 levels of the Animal random effect to the model—a substantial number.

Output 62.1.2 Comparing Mixed Model Tools for Increasingly Sparse Problems

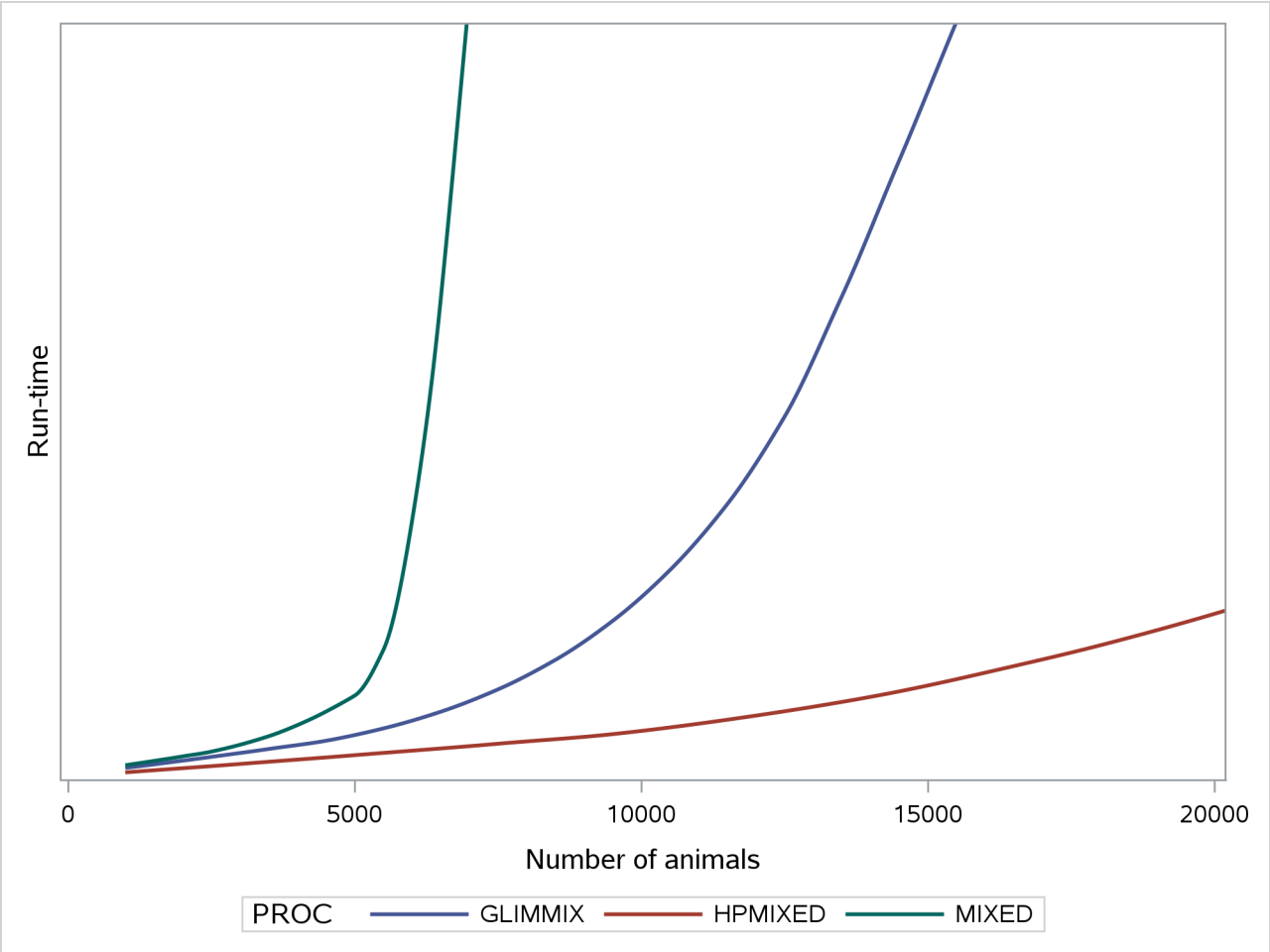

The vertical axis in Output 62.1.2 measures run time, but the units are omitted: relative performance is what counts, and that is expected to be fairly invariant to machine architecture. The output shows that while the performance of the MIXED and GLIMMIX procedures is relatively competitive with PROC HPMIXED for up to 3000 or 4000 animals, both procedures' relative performance decreases as the number of animals increases into the tens of thousands.

As a caveat, note that PROC HPMIXED can be *inefficient* relative to PROC MIXED and PROC GLIMMIX for models and data that are not sparse, because it can take many times longer to invert a large, dense matrix by sparse techniques. For example, Output 62.1.3 shows relative performance of the three procedures for simulated data like the preceding, but where the fixed part of the model consists of an increasing number of continuous covariates and is thus dense.

Output 62.1.3 Comparing Mixed Model Tools for Increasingly Dense Problems

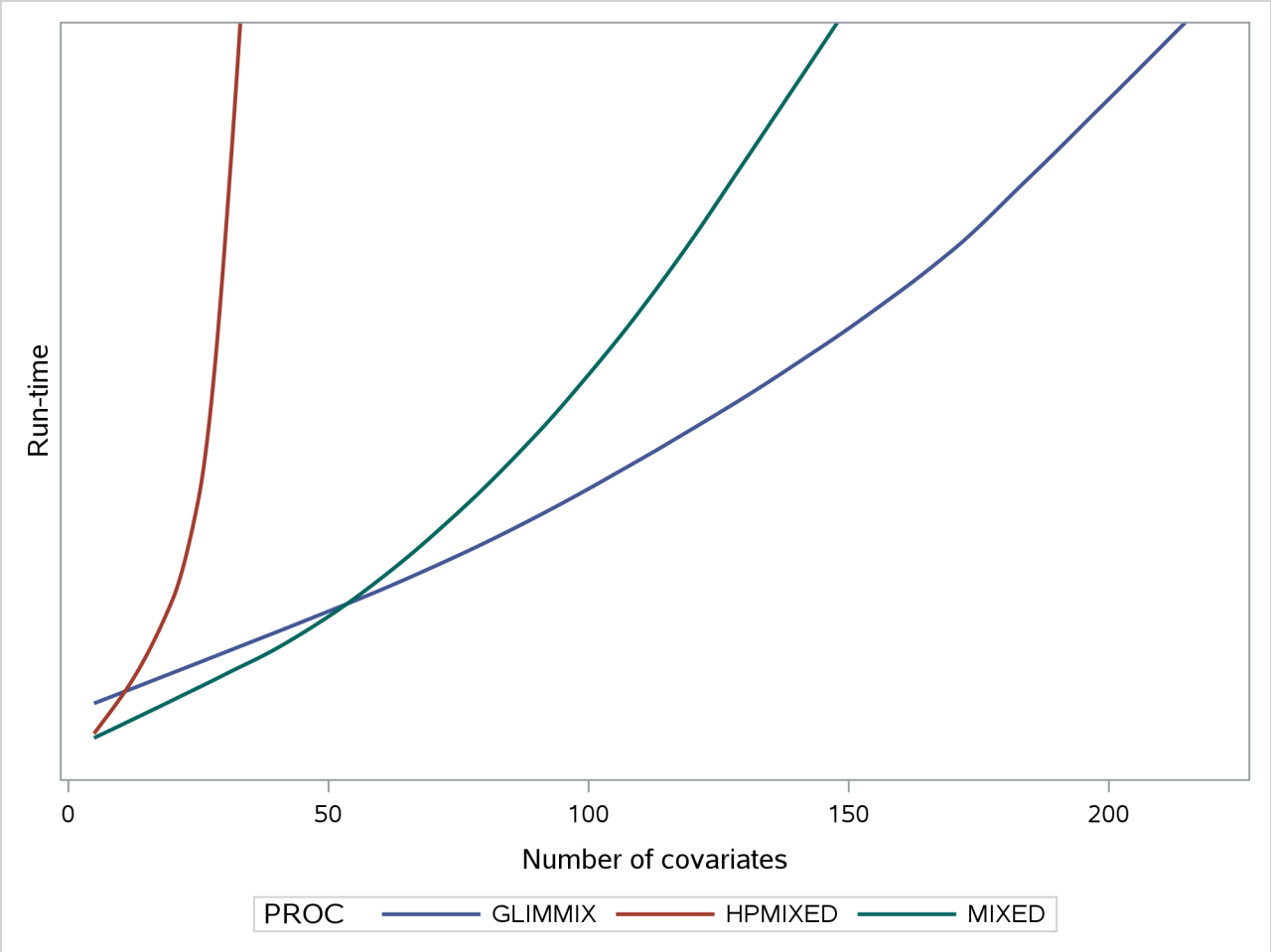

As before, the HPMIXED procedure is more efficient than the MIXED and GLIMMIX procedures for few covariates, but when the fixed-effect calculations dominate the run time, PROC HPMIXED rapidly becomes relatively inefficient as the size of the dense fixed-effect matrix increases. Also note that while PROC MIXED is more efficient than PROC GLIMMIX for small to moderate numbers of covariates, PROC GLIMMIX has the best performance as the number of covariates get very large.

Example 62.2: Comparing Results from PROC HPMIXED and PROC MIXED

(View the complete code for this example (hpmex2.sas) in the example repository.)

This example revisits the mixed model problem from the section "Getting Started: MIXED Procedure" on page 6766, in Chapter 84, "The MIXED Procedure," with the data set shown in the following statements:

```
data heights;
   input Family Gender$ Height @@;
   datalines;
1 F 67   1 F 66   1 F 64   1 M 71   1 M 72   2 F 63
2 F 63   2 F 67   2 M 69   2 M 68   2 M 70   3 F 63
3 M 64   4 F 67   4 F 66   4 M 67   4 M 67   4 M 69
;
```

The response variable Height measures the heights (in inches) of 18 individuals. The individuals are classified according to Family and Gender. The following statements fit a mixed model with random effects for Family and the Family*Gender interaction with the MIXED procedure:

```
proc mixed;
   class Family Gender;
   model Height = Gender / s;
   random Family Family*Gender / s;
run;
```

The "Iteration History" and "Fit Statistics" tables for the optimization in PROC MIXED are shown in Output 62.2.1. The MIXED procedure converges after six iterations and achieves a –2 restricted log likelihood of 71.02246.

Output 62.2.1 Iteration History and Fit Statistics: MIXED Procedure

The Mixed Procedure

Iteration History			
Iteration	**Evaluations**	**-2 Res Log Like**	**Criterion**
0	1	74.11074833	
1	2	71.51614003	0.01441208
2	1	71.13845990	0.00412226
3	1	71.03613556	0.00058188
4	1	71.02281757	0.00001689
5	1	71.02245904	0.00000002
6	1	71.02245869	0.00000000

Fit Statistics	
-2 Res Log Likelihood	71.0
AIC (Smaller is Better)	77.0
AICC (Smaller is Better)	79.0
BIC (Smaller is Better)	75.2

Output 62.2.2 displays the covariance parameter estimates and the solutions for the fixed and random effects. Because the fixed-effect model contains a classification effect (Gender) and an intercept, the $\mathbf{X}'\mathbf{X}$ matrix is singular. Only two fixed-effect parameters can be estimated in this model. The MIXED procedure, relying

on a sweep operation in the order in which effects enter the model, determines that the last column of the $\mathbf{X}'\mathbf{X}$ matrix is a linear function of previous columns. Consequently, the coefficient for the second level of the Gender variable is zero.

Output 62.2.2 Parameter Estimates and Solutions: MIXED Procedure

Covariance Parameter Estimates

Cov Parm	Estimate
Family	2.4010
Family*Gender	1.7657
Residual	2.1668

Solution for Fixed Effects

Effect	Gender	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		68.2114	1.1477	3	59.43	<.0001
Gender	F	-3.3621	1.1923	3	-2.82	0.0667
Gender	M	0	.	.	.	.

Solution for Random Effects

Effect	Gender	Family	Estimate	Std Err Pred	DF	t Value	Pr > |t|
Family		1	1.2680	1.1201	10	1.13	0.2840
Family		2	0.08980	1.1121	10	0.08	0.9372
Family		3	-1.6660	1.1712	10	-1.42	0.1853
Family		4	0.3082	1.1201	10	0.28	0.7888
Family*Gender	F	1	-0.3198	1.0810	10	-0.30	0.7734
Family*Gender	M	1	1.2523	1.0933	10	1.15	0.2787
Family*Gender	F	2	-0.4299	1.0774	10	-0.40	0.6983
Family*Gender	M	2	0.4959	1.0774	10	0.46	0.6551
Family*Gender	F	3	-0.08229	1.1409	10	-0.07	0.9439
Family*Gender	M	3	-1.1429	1.1409	10	-1.00	0.3401
Family*Gender	F	4	0.8320	1.0933	10	0.76	0.4642
Family*Gender	M	4	-0.6053	1.0810	10	-0.56	0.5878

The "Type 3 Tests of Fixed Effects" table in Output 62.2.3 is produced by the MIXED procedure by default.

Output 62.2.3 Test of Gender Effect

Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
Gender	1	3	7.95	0.0667

The same linear mixed model is fit with the HPMIXED procedure with the following statements:

```
proc hpmixed;
   class Family Gender;
   model Height = Gender / s;
   random Family Family*Gender / s;
```

```
   test gender;
run;
```

Output 62.2.4 displays the "Iteration History" and "Fit Statistics" tables. The HPMIXED procedure, with its default quasi-Newton algorithm, achieves the same –2 restricted log likelihood as the MIXED procedure (71.02246; see Output 62.2.1).

Output 62.2.4 Iteration History and Fit Statistics: HPMIXED Procedure

The HPMIXED Procedure

Iteration History

Iteration	Evaluations	Objective Function	Change	Max Gradient
0	4	71.023177956	.	0.034074
1	3	71.022519936	0.00065802	0.007839
2	3	71.022477283	0.00004265	0.004674
3	2	71.0224587	0.00001858	0.000168
4	2	71.022458689	0.00000001	3.28E-6

Fit Statistics

-2 Res Log Likelihood	71.02246
AIC (Smaller is Better)	77.02246
AICC (Smaller is Better)	79.02246
BIC (Smaller is Better)	75.18134
CAIC (Smaller is Better)	78.18134
HQIC (Smaller is Better)	72.98226

Output 62.2.5 displays the results that correspond to those in Output 62.2.2 in the MIXED procedure.

Output 62.2.5 Parameter Estimates and Solutions: HPMIXED Procedure

Covariance Parameter Estimates

Cov Parm	Estimate
Family	2.4010
Family*Gender	1.7657
Residual	2.1668

Solution for Fixed Effects

Effect	Gender	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		0	.	.	.	.
Gender	F	64.8493	1.1477	16	56.50	<.0001
Gender	M	68.2114	1.1477	16	59.43	<.0001

Output 62.2.5 *continued*

Solution for Random Effects							
Effect	Gender	Family	Estimate	Std Err Pred	DF	t Value	Pr > |t|
Family		1	1.2680	1.1201	16	1.13	0.2743
Family		2	0.08980	1.1121	16	0.08	0.9366
Family		3	-1.6660	1.1712	16	-1.42	0.1741
Family		4	0.3082	1.1201	16	0.28	0.7867
Family*Gender	F	1	-0.3198	1.0810	16	-0.30	0.7712
Family*Gender	M	1	1.2523	1.0933	16	1.15	0.2689
Family*Gender	F	2	-0.4299	1.0774	16	-0.40	0.6951
Family*Gender	M	2	0.4959	1.0774	16	0.46	0.6515
Family*Gender	F	3	-0.08229	1.1409	16	-0.07	0.9434
Family*Gender	M	3	-1.1429	1.1409	16	-1.00	0.3314
Family*Gender	F	4	0.8320	1.0933	16	0.76	0.4577
Family*Gender	M	4	-0.6053	1.0810	16	-0.56	0.5832

A number of points are noteworthy in comparing the results from the procedures. The covariance parameter estimates are the same, yet the solutions for the fixed effects differ. In fact, both solutions are correct. Solving a sparse system of linear equations requires reordering of the mixed model equations to minimize memory consumption in the factorization process. As a consequence, the order in which singularities are detected can differ from the order in which effects enter the model. Mathematically, the two sets of solutions simply correspond to different choices for the generalized inverse in solving a singular linear system. See the sections "Generalized Inverse Matrices" on page 47 and "Linear Model Theory" on page 55, in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," for more information about the role and importance of generalized inverses in linear model analysis.

Although the two sets of solutions for the fixed effects correspond to different choices of generalized inverses, many important results are invariant to the choice of the g-inverse. For example, the solutions for the random effects in Output 62.2.5 and Output 62.2.2 are identical. Also, the test for the Gender effect yields the same F value in both analyses (compare Output 62.2.6 and Output 62.2.3). However, note that the p-values associated with both F tests and t tests differ between the two procedures. This is due to their different default methods for computing the degrees of freedom. For this model, the HPMIXED procedure use the residual method to determine the denominator degrees of freedom for tests of fixed effects, whereas the MIXED procedure uses the containment method. The containment method is order-dependent, and thus not available in the HPMIXED procedure.

Output 62.2.6 Parameter Estimates and Solutions: HPMIXED Procedure

Type III Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
Gender	1	16	7.95	0.0123

Example 62.3: Using PROC GLIMMIX for Further Analysis of PROC HPMIXED Fit

(View the complete code for this example (hpmex3.sas) in the example repository.)

The HPMIXED procedure handles only a subset of the analyses of the GLIMMIX procedure. However, you can use the HPMIXED procedure to accelerate your GLIMMIX procedure analyses for large problems. The idea is to use PROC HPMIXED to maximize the likelihood and produce parameter estimates more quickly than PROC GLIMMIX, and then to pass these parameter estimates to PROC GLIMMIX for some further analysis that is not available within PROC HPMIXED.

This example revisits the mixed model problem from the section "Getting Started: HPMIXED Procedure" on page 4914 to illustrate how to obtain the covariance estimates from the HPMIXED procedure and, in turn, how to use these estimates in PROC GLIMMIX's PARMS statement. The following statements again simulate data from animals of different species on different farms:

```
data Sim;
   keep Species Farm Animal Yield;
   array AnimalEffect{3000};
   array AnimalSpecies{3000};
   array AnimalFarm{3000};
   do i = 1 to 3000;
      AnimalEffect{i} = sqrt(4.0)*rannor(12345);
      AnimalSpecies{i} = 1 + int(5*ranuni(12345));
      AnimalFarm{i}    = 1 + int(10*ranuni(12345));
   end;
   do i = 1 to 40000;
      Animal  = 1 + int(3000*ranuni(12345));
      Species = AnimalSpecies{Animal};
      Farm    = AnimalFarm{Animal};
      Yield   = 1 + Species + int(Farm/2) + AnimalEffect{Animal}
                  + sqrt(8.0)*rannor(12345);
      output;
   end;
run;
```

Note that in the preceding DATA step program, certain pairs of farms are simulated to have the same effect on yield. Suppose that your goal is to determine which farms are significantly different. While the HPMIXED procedure has an LSMEANS statement, it has no options for multiple comparisons. The following statements first use the HPMIXED procedure to obtain the covariance estimates, saving them in the SAS data set HPMEstimate. Then the GLIMMIX procedure is executed with the PARMS statement to initialize the parameter values from the data set HPMEstimate and with the HOLD= and NOITER options to prevent further optimization iterations. The LSMEANS statement is used in PROC GLIMMIX to perform multiple comparisons of the LS-means for farms, and the results are displayed as a so-called diffogram.

```
proc hpmixed data=Sim;
   class Species Farm Animal;
   model Yield = Farm|Species;
   random Animal;
   test Species Species*Farm;
   ods output CovParms=HPMEstimate;
run;
```

```
ods graphics on;
proc glimmix data=Sim;
   class Species Farm Animal;
   model Yield = Farm|Species;
   random int/sub=Animal;
   parms /pdata=HPMEstimate hold=1,2 noiter;
   lsmeans Farm / pdiff=all plot=diffplot;
run;
```

The iteration histories for the two procedures are shown in Output 62.3.1 and Output 62.3.2. Whereas PROC HPMIXED requires several iterations in order to converge, PROC GLIMMIX "converges" to the same value in one step, with no iteration since the options HOLD= and NOITER are used.

Output 62.3.1 Iteration History for the HPMIXED Procedure

The HPMIXED Procedure

Iteration History				
Iteration	Evaluations	Objective Function	Change	Max Gradient
0	4	202516.66891	.	0.841954
1	6	202516.66887	0.00004385	0.000641
2	1	202516.66887	-0.00000000	0.000641

Output 62.3.2 Iteration History for the GLIMMIX Procedure

The GLIMMIX Procedure

Iteration History					
Iteration	Restarts	Evaluations	Objective Function	Change	Max Gradient
0	0	4	202516.66887	.	0

The graphical multiple-comparisons analysis for the LS-means of farms is shown in Output 62.3.3. It confirms the pairwise equalities between farm effects with which the data were simulated.

Output 62.3.3 LS-Means Plot of Pairwise Farm Differences

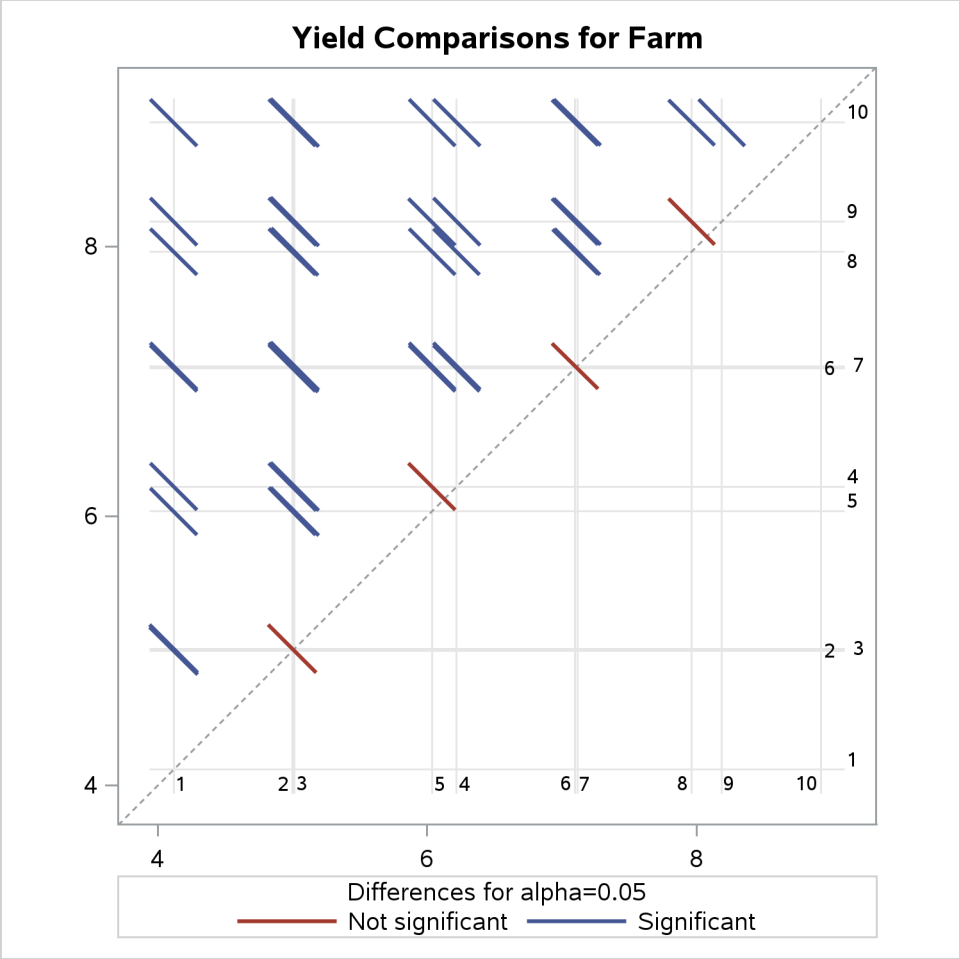

For more information about the interpretation of the LS-means difference plot, see the section "ODS Graphics" on page 3990, in Chapter 52, "The GLIMMIX Procedure."

Example 62.4: Mixed Model Analysis of Microarray Data

(View the complete code for this example (hpmex4.sas) in the example repository.)

Microarray experiments are an advanced genomic technique used in the discovery of new treatments for diseases. Microarray analysis allows for the detection of tens of thousands of genes in a single DNA sample. A microarray is a glass slide or membrane that has been spotted or "arrayed" with DNA fragments or oligonucleotides representing specific genes. The response of the gene detected by a spot is proportional to the intensity of fluorescence associated with that spot. These gene responses can indicate associations with disease conditions, but they can also be affected by systematic biases and different treatments such as sex and genotypes. Statistical models for microarray data attempt to assess the significance and magnitude of gene effects across treatments while adjusting for these systematic biases and to evaluate the significance of differences between treatments.

There are two statistical approaches frequently used in mixed model analysis for microarray data. The first approach is to fit multiple gene-specific models to data normalized for systematic biases (Wolfinger et al. 2001; Gibson and Wolfinger 2004). This approach is based on assuming that the biases are independent from the gene effects. If this assumption is untenable, then a second approach fits a single model that combines both the systematic biases and the gene effects (Kerr, Martin, and Churchill 2000; Churchill 2002; Littell et al. 2006). When the number of genes is very large, several hundreds to tens of thousands, this is an analysis for which the sparse matrix approach implemented in the HPMIXED procedure is well suited.

The following SAS statements simulate a microarray experiment with a so-called loop design structure, which is commonly used in such studies. There are 500 genes, each gene occurs in 6 arrays, and each array has 2 dyes.

```
%let narray  = 6;
%let ndye    = 2;
%let nrow    = 4;
%let ngene   = 500;
%let ntrt    = 6;
%let npin    = 4;
%let ndip    = 4;
%let no      = %eval(&ndye*&nrow*&ngene);
%let tno     = %eval(&narray*&no);

data microarray;
   keep Gene MArray Dye Trt Pin Dip log2i;
   array PinDist{&tno};
   array DipDist{&tno};
   array GeneDist{&tno};

   array ArrayEffect{&narray};
   array ArrayGeneEffect{%eval(&narray*&ngene)};
   array ArrayDipEffect{%eval(&narray*&ndip)};
   array ArrayPinEffect{%eval(&narray*&npin)};

   do i = 1 to &tno;
      PinDist{i}  = 1 + int(&npin*ranuni(12345));
      DipDist{i}  = 1 + int(&ndip*ranuni(12345));
      GeneDist{i} = 1 + int(&ngene*ranuni(12345));
   end;

   igene = 0;
   idip = 0;
   ipin = 0;
   do i = 1 to &narray;
      ArrayEffect{i} = sqrt(0.014)*rannor(12345);
      do j = 1 to &ngene;
         igene = igene+1;
         ArrayGeneEffect{igene} = sqrt(0.0017)*rannor(12345);
      end;
      do j = 1 to &ndip;
         idip = idip + 1;
         ArrayDipEffect{idip} = sqrt(0.0033)*rannor(12345);
      end;
```

```
      do j = 1 to &npin;
         ipin = ipin + 1;
         ArrayPinEffect{ipin} = sqrt(0.037)*rannor(12345);
      end;
   end;

   i = 0;
   do MArray = 1 to &narray;
      do Dye = 1 to &ndye;
         do Row = 1 to &nrow;
            do k = 1 to &ngene;
               if MArray=1 and Dye = 1 then do;
                  Trt = 0;
                  trtc = 0;
               end;
               else do;
                  if trtc >= &no then trtc = 0;
                  if trtc = 0 then do;
                     Trt = Trt + 1;
                     if Trt >= &ntrt then do;
                        Trt = 0;
                        trtc = 0;
                     end;
                  end;
                  trtc = trtc + 1;
               end;
               i = i + 1;
               Pin = PinDist{i};
               Dip = DipDist{i};
               Gene = GeneDist{i};
               a   = ArrayEffect{MArray};
               ag  = ArrayGeneEffect{(MArray-1)*&ngene+Gene};
               ad  = ArrayDipEffect{(MArray-1)*&ndip+Dip};
               ap  = ArrayPinEffect{(MArray-1)*&npin+Pin};
               log2i   = 1 +
                        + Dye
                        + Trt
                        + Gene/1000.0
                        + Dye*Gene/1000.0
                        + Trt*Gene/1000.0
                        + Pin
                        + a
                        + ag
                        + ad
                        + ap
                        + sqrt(0.02)*rannor(12345);
               output;
            end;
         end;
      end;
   end;
run;
```

A linear mixed model for fitting the log intensity data Y_{ijkmnr} from such a design is described by

Littell et al. (2006) as follows:

Y_{ijkmnr}	$=$		**Fixed Effects**
		μ	Overall mean
	$+$	λ_i	Gene
	$+$	τ_j	Treatment
	$+$	δ_k	Dye
	$+$	$(\tau\lambda)_{ij}$	Treatment-by-gene
	$+$	$(\delta\lambda)_{ik}$	Dye-by-gene
	$+$	p_r	Pin
			Random Effects
	$+$	a_m	Microarray
	$+$	$(a\lambda)_{im}$	Microarray-by-gene
	$+$	$d(a)_{mn}$	Dip-within-microarray
	$+$	$(ap)_{mr}$	Microarray-by-pin
	$+$	e_{ijkmnr}	Residual noise

You can use the HPMIXED procedure with the following statements to fit this model:

```
proc hpmixed data=microarray;
   class marray dye trt gene pin dip;
   model log2i = dye trt gene dye*gene trt*gene pin;
   random marray marray*gene dip(marray) pin*marray;
   test trt;
run;
```

The "Dimensions" table shown in Output 62.4.1 indicates that this is a very large model, with 4,512 columns in the **X** matrix and 3,054 columns in the **Z** matrix. It will be computationally very inefficient to fit this model by using dense matrix methods; the sparse matrix approach of PROC HPMIXED is of critical importance.

Output 62.4.1 Mixed Model Dimensions

The HPMIXED Procedure

Dimensions	
G-side Cov. Parameters	4
R-side Cov. Parameters	1
Columns in X	4513
Columns in Z	3054
Subjects (Blocks in V)	1

The p-value in Output 62.4.2 indicates that there are significant differences between treatments.

Output 62.4.2 Type III Tests of Fixed Effects

Type III Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
Trt	5	20497	370006	<.0001

Example 62.5: Repeated Measures

(View the complete code for this example (hpmex5.sas) in the example repository.)

The following data are from Pothoff and Roy (1964) and consist of growth measurements for 11 girls and 16 boys at ages 8, 10, 12, and 14. Some of the observations are suspect (for example, the third observation for person 20); however, all of the data are used here for comparison purposes.

The analysis strategy employs a linear growth curve model for the boys and girls in addition to a variance-covariance model that incorporates correlations for all of the observations that arise from the same person. The PROC HPMIXED statements to fit an unstructured variance matrix are as follows:

```
data pr;
   input Person Gender $ y1 y2 y3 y4;
   y=y1; Time=1; Age=8;  output;
   y=y2; Time=2; Age=10; output;
   y=y3; Time=3; Age=12; output;
   y=y4; Time=4; Age=14; output;
   drop y1-y4;
   datalines;
 1   F   21.0    20.0    21.5    23.0
 2   F   21.0    21.5    24.0    25.5
 3   F   20.5    24.0    24.5    26.0
 4   F   23.5    24.5    25.0    26.5
 5   F   21.5    23.0    22.5    23.5
 6   F   20.0    21.0    21.0    22.5
 7   F   21.5    22.5    23.0    25.0
 8   F   23.0    23.0    23.5    24.0
 9   F   20.0    21.0    22.0    21.5
10   F   16.5    19.0    19.0    19.5
11   F   24.5    25.0    28.0    28.0
12   M   26.0    25.0    29.0    31.0
13   M   21.5    22.5    23.0    26.5
14   M   23.0    22.5    24.0    27.5
15   M   25.5    27.5    26.5    27.0
16   M   20.0    23.5    22.5    26.0
17   M   24.5    25.5    27.0    28.5
18   M   22.0    22.0    24.5    26.5
19   M   24.0    21.5    24.5    25.5
20   M   23.0    20.5    31.0    26.0
21   M   27.5    28.0    31.0    31.5
22   M   23.0    23.0    23.5    25.0
23   M   21.5    23.5    24.0    28.0
24   M   17.0    24.5    26.0    29.5
25   M   22.5    25.5    25.5    26.0
26   M   23.0    24.5    26.0    30.0
27   M   22.0    21.5    23.5    25.0
;

proc hpmixed data=pr;
   class Person Gender Time;
   model y = Gender Age Gender*Age;
   test Gender Age Gender*Age;
```

```
   repeated Time / type=un subject=Person r;
run;
```

The MODEL statement first lists the dependent variable Y. The fixed effects are then listed after the equal sign. The variable Gender requests a different intercept for the girls and boys, Age models an overall linear growth trend, and Gender*Age makes the slopes different over time. It is actually not necessary to specify Age separately, but doing so enables PROC HPMIXED to carry out a test for heterogeneous slopes.

The REPEATED statement contains a *repeated-effect* Time. The TYPE=UN option models the covariance as an unstructured block for each SUBJECT=Person. Each of the 27 subjects has a maximum of four observations. Therefore, the **R** matrix is block diagonal with 27 blocks, each block consisting of identical 4×4 unstructured matrices. The 10 parameters of these unstructured blocks make up the covariance parameters estimated by restricted maximum likelihood. The R= option requests that the first block of **R** be displayed.

The results from this analysis are shown in Output 62.5.1 through Output 62.5.5.

Output 62.5.1 Repeated Measures Analysis

The HPMIXED Procedure

Dimensions	
G-side Cov. Parameters	0
R-side Cov. Parameters	10
Columns in X	6
Columns in Z per Subject	0
Subjects (Blocks in V)	27

In Output 62.5.1, the 10 covariance parameters result from the 4×4 unstructured blocks of **R**. There is no **Z** matrix for this model.

Output 62.5.2 Repeated Measures Analysis (*continued*)

Number of Observations Read	108
Number of Observations Used	108

Output 62.5.2 *continued*

Iteration History				
Iteration	Evaluations	Objective Function	Change	Max Gradient
0	4	483.55903028	.	18.65974
1	4	446.6618154	36.89721488	14.63195
2	5	430.2967104	16.36510500	10.93182
3	5	427.86149052	2.43521988	12.34361
4	2	426.16528163	1.69620890	8.094057
5	3	425.56874743	0.59653420	3.517822
6	2	424.91919206	0.64955537	2.492626
7	3	424.731766	0.18742606	2.110784
8	3	424.66856966	0.06319634	1.417574
9	2	424.63858357	0.02998609	1.468348
10	2	424.60787324	0.03071033	1.174872
11	2	424.5593949	0.04847834	0.601039
12	3	424.55305379	0.00634111	0.316659
13	2	424.54886941	0.00418438	0.170275
14	3	424.54696194	0.00190747	0.072622
15	3	424.5468178	0.00014413	0.019582
16	3	424.54680027	0.00001753	0.001888
17	3	424.5468002	0.00000007	0.000235

Convergence criterion (GCONV=1E-8) satisfied.

The 17 quasi-Newton iterations are used to find the maximum likelihood estimates (Output 62.5.2).

Output 62.5.3 Repeated Measures Analysis (*continued*)

Estimated R Matrix for Person 1				
Row	Col1	Col2	Col3	Col4
1	5.4252	2.7092	3.8411	2.7151
2	2.7092	4.1906	2.9745	3.3137
3	3.8411	2.9745	6.2632	4.1332
4	2.7151	3.3137	4.1332	4.9862

The 4×4 matrix in Output 62.5.3 is the estimated unstructured covariance matrix. It is the estimate of the first block of **R**, and the other 26 blocks all have the same estimate.

Output 62.5.4 Repeated Measures Analysis (*continued*)

Covariance Parameter Estimates		
Cov Parm	**Subject**	**Estimate**
UN(1,1)	Person	5.4252
UN(2,1)	Person	2.7092
UN(2,2)	Person	4.1906
UN(3,1)	Person	3.8411
UN(3,2)	Person	2.9745
UN(3,3)	Person	6.2632
UN(4,1)	Person	2.7151
UN(4,2)	Person	3.3137
UN(4,3)	Person	4.1332
UN(4,4)	Person	4.9862

The "Covariance Parameter Estimates" table in Output 62.5.4 lists the 10 estimated covariance parameters in order; note their correspondence to the first block of **R** displayed in Output 62.5.3. The parameter estimates are labeled according to their location in the block in the Cov Parm column, and all of these estimates are associated with Person as the subject effect.

Output 62.5.5 Repeated Measures Analysis (*continued*)

Type III Tests of Fixed Effects				
Effect	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Gender	1	104	1.08	0.3011
Age	1	104	102.35	<.0001
Age*Gender	1	104	7.40	0.0076

The "Type III Tests of Fixed Effects" table in Output 62.5.5 displays Type III tests for all of the fixed effects. These tests are partial in the sense that they account for all of the other fixed effects in the model.

Since the different levels of the repeated effect represent different years, it is natural to try fitting a time series model to the data within each subject. To obtain time series structures in **R**, you can replace TYPE=UN with TYPE=AR(1) to obtain the first-order autoregressive covariance matrices. For example, the statements to fit an AR(1) structure are as follows:

```
proc hpmixed data=pr;
   class Person Gender Time;
   model y = Gender Age Gender*Age;
   repeated Time / type=ar(1) sub=Person r;
run;
```

The estimated AR(1) structure covariance matrix of the first block of **R** is shown in Output 62.5.6

Output 62.5.6 Repeated Measures Analysis

The HPMIXED Procedure

Estimated R Matrix for Person 1				
Row	**Col1**	**Col2**	**Col3**	**Col4**
1	5.2144	3.2563	2.0335	1.2699
2	3.2563	5.2144	3.2563	2.0335
3	2.0335	3.2563	5.2144	3.2563
4	1.2699	2.0335	3.2563	5.2144

References

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Bozdogan, H. (1987). "Model Selection and Akaike's Information Criterion (AIC): The General Theory and Its Analytical Extensions." *Psychometrika* 52:345–370.

Brown, H., and Prescott, R. (1999). *Applied Mixed Models in Medicine*. New York: John Wiley & Sons.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Churchill, G. A. (2002). "Fundamentals of Experimental Design for cDNA Microarray." *Nature Genetics* 32:490–495.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society, Series B* 39:1–38.

George, J. A., and Liu, J. W. (1981). *Computer Solutions of Large Sparse Positive Definite Systems*. Englewood Cliffs, NJ: Prentice-Hall.

Gibson, G., and Wolfinger, R. D. (2004). "Gene Expression Profiling Using Mixed Models." In *Genetic Analysis of Complex Traits Using SAS*, edited by A. M. Saxton, 251–278. Cary, NC: SAS Institute Inc.

Gilmour, A. R., Thompson, R., and Cullis, B. R. (1995). "Average Information REML: An Efficient Algorithm for Variance Parameter Estimation in Linear Mixed Models." *Biometrics* 51:1440–1450.

Hannan, E. J., and Quinn, B. G. (1979). "The Determination of the Order of an Autoregression." *Journal of the Royal Statistical Society, Series B* 41:190–195.

Henderson, C. R. (1990). "Statistical Method in Animal Improvement: Historical Overview." In *Advances in Statistical Methods for Genetic Improvement of Livestock*, 1–14. New York: Springer-Verlag.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Johnson, D. L., and Thompson, R. (1995). "Restricted Maximum Likelihood Estimation of Variance Components for Univariate Animal Models Using Sparse Matrix Techniques and Average Information." *Journal of Dairy Science* 78:449–456.

Kerr, M. K., Martin, M., and Churchill, G. A. (2000). "Analysis of Variance for Gene Expression Microarray Data." *Journal of Computational Biology* 7:819–837.

Littell, R. C., Milliken, G. A., Stroup, W. W., and Wolfinger, R. D. (1996). *SAS System for Mixed Models*. Cary, NC: SAS Institute Inc.

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and Schabenberger, O. (2006). *SAS for Mixed Models*. 2nd ed. Cary, NC: SAS Institute Inc.

McLean, R. A., Sanders, W. L., and Stroup, W. W. (1991). "A Unified Approach to Mixed Linear Models." *American Statistician* 45:54–64.

Ott, E. R. (1967). "Analysis of Means: A Graphical Procedure." *Industrial Quality Control* 24:101–109. Reprinted in *Journal of Quality Technology* 15 (1983): 10–18.

Pothoff, R. F., and Roy, S. N. (1964). "A Generalized Multivariate Analysis of Variance Model Useful Especially for Growth Curve Problems." *Biometrika* 51:313–326.

Schabenberger, O., Gregoire, T. G., and Kong, F. (2000). "Collections of Simple Effects and Their Relationship to Main Effects and Interactions in Factorials." *American Statistician* 54:210–214.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Searle, S. R., Casella, G., and McCulloch, C. E. (1992). *Variance Components*. New York: John Wiley & Sons.

Shewchuk, J. R. (1994). *An Introduction to the Conjugate Gradient Method without the Agonizing Pain*. Technical report, Carnegie Mellon University.

Tsuruta, S., Misztal, I., and Stranden, I. (2001). "Use of the Preconditioned Conjugate Gradient Algorithm as a Generic Solver for Mixed-Model Equations in Animal Breeding Applications." *Journal of Animal Science* 79:1166–1172.

Verbeke, G., and Molenberghs, G., eds. (1997). *Linear Mixed Models in Practice: A SAS-Oriented Approach*. New York: Springer.

Verbeke, G., and Molenberghs, G. (2000). *Linear Mixed Models for Longitudinal Data*. New York: Springer.

Winer, B. J. (1971). *Statistical Principles in Experimental Design*. 2nd ed. New York: McGraw-Hill.

Wolfinger, R. D., Gibson, G., Wolfinger, E., Bennett, L., Hamadeh, H., Bushel, P., Afshari, C., and Paules, R. S. (2001). "Assessing Gene Significance from cDNA Microarray Expression Data via Mixed Models." *Journal of Computational Biology* 8:625–637.

Chapter 63
The HPNLMOD Procedure

Contents

Overview: HPNLMOD Procedure **4980**
- PROC HPNLMOD Features 4980
- PROC HPNLMOD Contrasted with the NLIN and NLMIXED Procedures 4980

Getting Started: HPNLMOD Procedure **4981**
- Least Squares Model 4981
- Binomial Model 4982

Syntax: HPNLMOD Procedure **4983**
- PROC HPNLMOD Statement 4983
- BOUNDS Statement 4988
- BY Statement 4989
- ESTIMATE Statement 4989
- MODEL Statement 4990
- PARAMETERS Statement 4990
- PERFORMANCE Statement 4993
- PREDICT Statement 4993
- RESTRICT Statement 4994
- Programming Statements 4995

Details: HPNLMOD Procedure **4998**
- Least Squares Estimation 4998
- Built-In Log-Likelihood Functions 4998
- Computational Method 5000
- Choosing an Optimization Algorithm 5001
- Displayed Output 5005
- ODS Table Names 5008

Examples: HPNLMOD Procedure **5009**
- Example 63.1: Segmented Model 5009

References **5012**

Overview: HPNLMOD Procedure

The HPNLMOD procedure is a high-performance procedure that uses either nonlinear least squares or maximum likelihood to fit nonlinear regression models. PROC HPNLMOD enables you to specify the model by using SAS programming statements, which give you greater flexibility in modeling the relationship between the response variable and independent (regressor) variables than do SAS procedures that use a more structured MODEL statement.

PROC HPNLMOD Features

The HPNLMOD procedure does the following:

- reads input data in parallel and writes output data in parallel when the data source is the appliance database
- is highly multithreaded during all phases of analytic execution
- computes analytical derivatives of user-provided expressions for more robust parameter estimations
- evaluates user-provided expressions and their confidence limits by using the ESTIMATE and PREDICT statements
- estimates parameters without specifying a particular distribution function by using the least squares method
- estimates parameters by using the maximum likelihood method when either a built-in distribution function is specified or a likelihood function is provided

Because the HPNLMOD procedure is a high-performance analytical procedure, it also exploits all the available cores and concurrent threads,

PROC HPNLMOD Contrasted with the NLIN and NLMIXED Procedures

Like the NLIN procedure, the HPNLMOD procedure estimates parameters by using least squares minimization for models that are specified by SAS programming statements. However, PROC HPNLMOD can also perform maximum likelihood estimation when information about the response variable's distribution is provided. PROC HPNLMOD also has a RESTRICT statement for specifying restrictions on parameter estimates that are more general than those that are available in PROC NLIN. Because the HPNLMOD and NLIN procedures use different optimization techniques, the available options that control the estimation process and resulting parameter estimates can differ between these procedures when equivalent models and data are analyzed.

Although it does not support the specification of random effects, PROC HPNLMOD is similar to PROC NLMIXED. Both procedures perform maximum likelihood estimation by using the same programming syntax and set of distributions to specify the model's mean term. In addition, both PROC HPNLMOD and PROC NLMIXED use the same optimization techniques and options. However, PROC NLMIXED does not support least squares parameter estimation.

Getting Started: HPNLMOD Procedure

The most common use of the HPNLMOD procedure is to estimate the parameters in a model in which the response variable is a nonlinear function of one or more of the parameters.

Least Squares Model

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpnlngs1.sas.)

The Michaelis-Menten model of enzyme kinetics (Ratkowsky 1990, p. 59) relates a substrate's concentration to its catalyzed reaction rate. The Michaelis-Menten model can be analyzed using a least squares estimation because it does not specify how the reaction rate is distributed around its predicted value. The relationship between reaction rate and substrate concentration is

$$f(\mathbf{x}, \boldsymbol{\theta}) = \frac{\theta_1 x_i}{\theta_2 + x_i}, \quad \text{for } i = 1, 2, \ldots, n$$

where x_i represents the concentration for n trials and $f(\mathbf{x}, \boldsymbol{\theta})$ is the reaction rate. The vector $\boldsymbol{\theta}$ contains the rate parameters.

For this model, which has experimental measurements of reaction rate and concentration stored in the enzyme data set, the following SAS statements estimate the parameters θ_1 and θ_2:

```
proc hpnlmod data=enzyme;
   parms theta1=0 theta2=0;
   model rate ~ residual(theta1*conc / (theta2 + conc));
run;
```

The least squares estimation performed by PROC HPNLMOD for this enzyme kinetics problem produces the analysis of variance table that is displayed in Figure 63.1. The table displays the degrees of freedom, sums of squares, and mean squares along with the model F test.

Figure 63.1 Nonlinear Least Squares Analysis of Variance

The HPNLMOD Procedure

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	290116	145058	88537.2	<.0001
Error	12	19.6606	1.6384		
Uncorrected Total	14	290135			

An intercept was not specified for this model.

Finally, Figure 63.2 displays the parameter estimates, standard errors, t statistics, and 95% confidence intervals for θ_1 and θ_2.

Figure 63.2 Parameter Estimates and Approximate 95% Confidence Intervals

Parameter Estimates							
Parameter	Estimate	Standard Error	DF	t Value	Approx Pr > |t|	Approximate 95% Confidence Limits	
theta1	158.1	0.6737	1	234.67	<.0001	156.6	159.6
theta2	0.0741	0.00313	1	23.69	<.0001	0.0673	0.0809

In the enzyme kinetics model, no information was supplied about the distribution of the reaction rate around the model's mean value. Therefore, the residual model distribution was specified to perform a least squares parameter fit.

Binomial Model

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpnlngs2.sas.)

In Example 88.3 cancer remission is modeled by expressing the maximum likelihood function for a binary distribution as a nonlinear least squares optimization. The following statements show an equivalent formulation of this model that uses PROC HPNLMOD and specifies the binary distribution explicitly:

```
proc hpnlmod data=remiss corr;
   parms int=-10 a=-2 b=-1 c=6;
   linp = int + a*cell + b*li + c*temp;
   p = probnorm(linp);
   model remiss ~ binary(1-p);
run;
```

This binary distribution model displays information about the quality of the estimation that is different from the information displayed in the section "Least Squares Model" on page 4981. No analysis of variance table is produced for this model; fit statistics that are based on the value of the likelihood function are displayed in Figure 63.3.

Figure 63.3 Nonlinear Likelihood Function Statistics

The HPNLMOD Procedure

Fit Statistics	
-2 Log Likelihood	21.9002
AIC (smaller is better)	29.9002
AICC (smaller is better)	31.7183
BIC (smaller is better)	35.0835

Parameter estimates for the binary distribution model that uses the same quantities as are used in the section "Least Squares Model" on page 4981 are displayed in Figure 63.4.

Figure 63.4 Parameter Estimates and Approximate 95% Confidence Intervals

Parameter Estimates							
Parameter	Estimate	Standard Error	DF	t Value	Approx Pr > |t|	Approximate 95% Confidence Limits	
int	-36.7548	32.3607	1	-1.14	0.2660	-103.2	29.6439
a	-5.6298	4.6376	1	-1.21	0.2353	-15.1454	3.8858
b	-2.2513	0.9790	1	-2.30	0.0294	-4.2599	-0.2426
c	45.1815	34.9095	1	1.29	0.2065	-26.4469	116.8

Syntax: HPNLMOD Procedure

The following statements are available in the HPNLMOD procedure:

PROC HPNLMOD < *options* > ;
  BOUNDS *constraint* < ,. . . ,*constraint* > ;
  BY *variables* ;
  ESTIMATE '*label*' *expression* < *options* > ;
  MODEL *dependent-variable* ∼ *distribution* ;
  PARAMETERS < *parameter-specification* > < ,. . . , *parameter-specification* > < / *options* > ;
  PERFORMANCE < *performance-options* > ;
  PREDICT '*label*' *expression* < *options* > ;
  RESTRICT *restriction1* < , *restriction2* . . . > ;
  Programming Statements ;

The PROC HPNLMOD statement and exactly one MODEL statement are required.

PROC HPNLMOD Statement

PROC HPNLMOD < *options* > ;

The PROC HPNLMOD statement invokes the procedure. Table 63.1 summarizes important options in the PROC HPNLMOD statement by function. These and other options in the PROC HPNLMOD statement are then described fully in alphabetical order.

Table 63.1 PROC HPNLMOD Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
OUT=	Specifies the output data set
Output Options	
CORR	Specifies the correlation matrix
COV	Specifies the covariance matrix

Table 63.1 *continued*

Option	Description
ECORR	Specifies the correlation matrix of additional estimates
ECOV	Specifies the covariance matrix of additional estimates
DF	Specifies the default degrees of freedom
NOPRINT	Suppresses ODS output
NOITPRINT	Suppresses output about iterations within the optimization process
Optimization Options	
ABSCONV=	Tunes an absolute function convergence criterion
ABSFCONV=	Tunes an absolute difference function convergence criterion
ABSGCONV=	Tunes the absolute gradient convergence criterion
FCONV=	Tunes the relative function convergence criterion
GCONV=	Tunes the relative gradient convergence criterion
MAXITER=	Chooses the maximum number of iterations in any optimization
MAXFUNC=	Specifies the maximum number of function evaluations in any optimization
MAXTIME=	Specifies the upper limit seconds of CPU time for any optimization
MINITER=	Specifies the minimum number of iterations in any optimization
TECHNIQUE=	Selects the optimization technique
Tolerance Options	
SINGULAR=	Tunes the general singularity criterion
User-Defined Format Options	
FMTLIBXML=	Specifies a file reference for a format stream
XMLFORMAT=	Specifies a file name for a format stream

You can specify the following *options* in the PROC HPNLMOD statement.

ABSCONV=*r*

ABSTOL=*r*

specifies an absolute function convergence criterion. For minimization, termination requires $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of r is the negative square root of the largest double-precision value, which serves only as a protection against overflow.

ABSFCONV=*r* **<** *n* **>**

ABSFTOL=*r***<** *n* **>**

specifies an absolute difference function convergence criterion. For all techniques except the Nelder-Mead simplex (NMSIMP) technique, termination requires a small change of the function value in successive iterations:

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}(k)$ is defined as the vertex

that has the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex. The default value is $r = 0$. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

ABSGCONV=*r* **<** *n* **>**

ABSGTOL=*r***<** *n* **>**

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small:

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $g_j(\cdot)$ is the gradient of the objective function with respect to the jth parameter. This criterion is not used by the NMSIMP technique. The default value is r =1E−5. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

ALPHA=*α*

specifies the level of significance α that is used in the construction of $100(1-\alpha)\%$ confidence intervals. The value must be strictly between 0 and 1; the default value of $\alpha = 0.05$ results in 95% intervals. This value is used as the default confidence level for limits that are computed in the "Parameter Estimates" table and is used in the LOWER and UPPER options in the PREDICT statement.

CORR

requests the approximate correlation matrix for the parameter estimates.

COV

requests the approximate covariance matrix for the parameter estimates.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC HPNLMOD. The default is the most recently created data set.

DF=*n*

specifies the default number of degrees of freedom to use in the calculation of p-values and confidence limits for additional parameter estimates.

ECORR

requests the approximate correlation matrix for all expressions that are specified in ESTIMATE statements.

ECOV

requests the approximate covariance matrix for all expressions that are specified in ESTIMATE statements.

FCONV=*r***<** *n* **>**

FTOL=*r***<** *n* **>**

specifies a relative function convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations:

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex that has the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex. The default is $r = 10^{-\mathrm{FDIGITS}}$, where FDIGITS is by default $-\log_{10}\{\epsilon\}$ and ϵ is the machine precision. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

FMTLIBXML=*file-ref*

specifies the file reference for the XML stream that contains the user-defined format definitions. For information about how to generate a XML stream for your formats, see the section "Working with Formats" on page 568.

GCONV=*r*< *n* >

GTOL=*r*< *n* >

specifies a relative gradient convergence criterion. For all techniques except the conjugate gradient (CONGRA) and NMSIMP techniques, termination requires that the normalized predicted function reduction be small:

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

Here, $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the CONGRA technique (where a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \, |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The default value is r =1E−8. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values are as follows, depending on the optimization technique (which you specify in the TECHNIQUE= option):

- TRUREG, NRRIDG, NEWRAP: $n = 125$
- QUANEW, DBLDOG: $n = 500$
- CONGRA: $n = 1{,}000$
- NMSIMP: $n = 3{,}000$

Optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed *n*.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on the optimization technique (which you specify in the TECHNIQUE= option):

- TRUREG, NRRIDG, NEWRAP: $n = 50$
- QUANEW, DBLDOG: $n = 200$
- CONGRA: $n = 400$
- NMSIMP: $n = 1{,}000$

These default values also apply when *n* is specified as a missing value.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The default value is the largest floating-point double representation of your computer. This time that is specified by *r* is checked only once at the end of each iteration. Therefore, the actual running time can be longer than *r*.

MINITER=*n*

MINIT=*n*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

NOITPRINT

suppresses the display of the "Iteration History" table.

NOPRINT

suppresses the generation of ODS output.

OUT=*SAS-data-set*

names the SAS data set to be created when one or more PREDICT statements are specified. A single OUT= data set is created to contain all predicted values when more than one PREDICT statement is specified. An error message is produced if a PREDICT statement is specified and an OUT= data set is not specified. For more information about output data sets in SAS high-performance analytical procedures, see the section "Output Data Sets" on page 568.

SINGULAR=*number*

tunes the general singularity criterion that is applied in sweeps and inversions. The default is 1E4 times the machine epsilon; this product is approximately 1E−12 on most computers.

TECHNIQUE=*keyword*

TECH=*keyword*

specifies the optimization technique for obtaining maximum likelihood estimates. You can choose from the following techniques by specifying the appropriate *keyword*:

CONGRA performs a conjugate-gradient optimization.

DBLDOG performs a version of double-dogleg optimization.

LEVMAR performs a Levenberg-Marquardt optimization.

NEWRAP performs a Newton-Raphson optimization that combines a line-search algorithm with ridging.

NMSIMP performs a Nelder-Mead simplex optimization.

NONE performs no optimization.

NRRIDG performs a Newton-Raphson optimization with ridging.

QUANEW performs a quasi-Newton optimization.

TRUREG performs a trust-region optimization.

The default value is TECHNIQUE=LEVMAR for least squares regression models and TECHNIQUE=NRRIDG for models where the distribution is specified.

XMLFORMAT=*filename*

specifies the file name for the XML stream that contains the user-defined format definitions. For information about how to generate a XML stream for your formats, see the section "Working with Formats" on page 568.

BOUNDS Statement

BOUNDS *constraint* < , *constraint* . . . > **;**

where *constraint* represents

< *number operator* > *parameter-list* < *operator number* >

Boundary constraints are specified in a BOUNDS statement. One- or two-sided boundary constraints are allowed. Elements in a list of boundary constraints are separated by commas. For example:

```
bounds 0 <= a1-a9 X <= 1, -1 <= c2-c5;
bounds b1-b10 y >= 0;
```

You can specify more than one BOUNDS statement. If you specify more than one lower (or upper) bound for the same parameter, the maximum (or minimum) of these is taken.

If the maximum l_j of all lower bounds is larger than the minimum of all upper bounds u_j for the same parameter θ_j, the boundary constraint is replaced by $\theta_j := l_j := \min(u_j)$, which is defined by the minimum of all upper bounds specified for θ_j.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPNLMOD to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPNLMOD procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

ESTIMATE Statement

ESTIMATE '*label*' *expression* < *options* > **;**

The ESTIMATE statement enables you to compute an additional estimate that is a function of the parameter values. You must provide a quoted string to identify the estimate and then provide a valid SAS expression. Multiple ESTIMATE statements are permitted, and results from all ESTIMATE statements are listed in a common table. PROC HPNLMOD computes approximate standard errors for the estimates by using the delta method (Billingsley 1986). It uses these standard errors to compute corresponding t statistics, p-values, and confidence limits.

The ECOV option in the PROC HPNLMOD statement produces a table that contains the approximate covariance matrix of all the additional estimates you specify. The ECORR option produces the corresponding correlation matrix.

You can specify the following *options* in the ESTIMATE statement:

ALPHA=α
specifies the alpha level to be used to compute confidence limits. The default value corresponds to the ALPHA= option in the PROC HPNLMOD statement.

DF=*d*

specifies the degrees of freedom to be used to compute *p*-values and confidence limits. The default value corresponds to the DF= option in the PROC HPNLMOD statement.

MODEL Statement

MODEL *dependent-variable* ∼ *distribution* **;**

The MODEL statement is the mechanism for either using a distribution specification to specify the distribution of the data or using the RESIDUAL distribution to specify a predicted value. You must specify a single dependent variable from the input data set, a tilde (∼), and then a distribution along with its parameters. You can specify the following values for *distribution*:

RESIDUAL(*m*) or **LS**(*m*) specifies no particular distribution. Instead the sum of squares of the differences between *m* and the dependent variable is minimized.

NORMAL(*m*, *v*) specifies a normal (Gaussian) distribution that has mean *m* and variance *v*.

BINARY(*p*) specifies a binary (Bernoulli) distribution that has probability *p*.

BINOMIAL(*n*, *p*) specifies a binomial distribution that has count *n* and probability *p*.

GAMMA(*a*, *b*) specifies a gamma distribution that has shape *a* and scale *b*.

NEGBIN(*n*, *p*) specifies a negative binomial distribution that has count *n* and probability *p*.

POISSON(*m*) specifies a Poisson distribution that has mean *m*.

GENERAL(*ll*) specifies a general log-likelihood function that you construct by using SAS programming statements.

The MODEL statement must follow any SAS programming statements that you specify for computing parameters of the preceding distributions. For information about the built-in log-likelihood functions, see the section "Built-In Log-Likelihood Functions" on page 4998 .

PARAMETERS Statement

PARAMETERS < *parameter-specification* > < *,. . . , parameter-specification* > < / *options* > **;**

PARMS < *parameter-specification* > < *,. . . , parameter-specification* > < / *options* > **;**

The purpose of the PARAMETERS statement is to provide starting values for the HPNLMOD procedure. You can provide values that define a single point in the parameter space or that define a set of points. For more information about *parameter-specification*, see the section "Assigning Starting Values by Using Parameter Specification" on page 4991.

You can specify the following *options* in the PARAMETERS statement after the slash (/).

BEST=*i* > 0

specifies the maximum number of parameter grid points and the corresponding objective function values to display in the "Parameters" table. If you specify this option, the parameter grid points are listed in ascending order of objective function value. By default, all parameter grid points are displayed.

PDATA=*SAS-data-set*

DATA=*SAS-data-set*

specifies the data set that provides parameter starting values.

START=*value*

DEFSTART=*value*

specifies a default starting value for all parameters.

There are four methods available for providing starting values to the optimization process. In descending order of precedence, the methods are as follows:

1. Specify values directly in the PARAMETERS statement.
2. Specify values in the PDATA= data set option.
3. Specify a single value for all parameters by using the START= option.
4. Use the default value 1.0.

The names that are assigned to parameters must be valid SAS names and must not coincide with names of variables in the input data set (see the DATA= option in the PROC HPNLMOD statement). Parameters that are assigned starting values through the PARAMETERS statement can be omitted from the estimation if the expression in the MODEL statement does not depend on them.

Assigning Starting Values by Using Parameter Specification

A *parameter-specification* has the following general form, where *name* identifies the parameter and *value-list* provides the set of starting values for the parameter:

name **=** *value-list*

Often the *value-list* contains only a single value, but more general and flexible list specifications such as the following are possible:

m	a single value
m1, *m2*, . . . , *mn*	several values
m TO *n*	a sequence in which *m* equals the starting value, *n* equals the ending value, and the increment equals 1
m TO *n* BY *i*	a sequence in which *m* equals the starting value, *n* equals the ending value, and the increment is *i*
m1, *m2* TO *m3*	mixed values and sequences

When you specify more than one value for a parameter, PROC HPNLMOD sorts the values in ascending sequence and removes duplicate values from the parameter list before forming the grid for the parameter search. If you specify several values for each parameter, PROC HPNLMOD evaluates the model at each point on the grid. The iterations then commence from the point on the grid that yields the smallest objective function value.

For example, the following PARAMETERS statement specifies five parameters and sets their possible starting values as shown in the following table:

```
parms  b0 = 0
       b1 = 4 to 8
       b2 = 0 to .6 by .2
       b3 = 1, 10, 100
       b4 = 0, .5, 1 to 4;
```

Possible Starting Values				
B0	B1	B2	B3	B4
0	4	0.0	1	0.0
	5	0.2	10	0.5
	6	0.4	100	1.0
	7	0.6		2.0
	8			3.0
				4.0

The objective function values are calculated for each of the $1 \times 5 \times 4 \times 3 \times 6 = 360$ combinations of possible starting values.

If you specify a starting value by using a *parameter-specification*, any starting values that are provided for this parameter through the PDATA= data set are ignored. The *parameter-specification* overrides the information in the PDATA= data set.

Assigning Starting Values from a SAS Data Set

The PDATA= option in the PARAMETERS statement enables you to assign starting values for parameters by using a SAS data set. The data set must contain at least two variables: a character variable named Parameter or Parm that identifies the parameter, and a numeric variable named Estimate or Est that contains the starting values. For example, the PDATA= option enables you to use the contents of the "ParameterEstimates" table from one PROC HPNLMOD run to supply starting values for a subsequent run, as follows:

```
proc hpnlmod data=d(obs=30);
   parameters alpha=100 beta=3 gamma=4;
   Switch = 1/(1+gamma*exp(beta*log(dose)));
   model y ~ residual(alpha*Switch);
   ods output ParameterEstimates=pest;
run;

proc hpnlmod data=d;
   parameters / pdata=pest;
   Switch = 1/(1+gamma*exp(beta*log(dose)));
   model y ~ residual(alpha*Switch);
```

```
run;
```

You can specify multiple values for a parameter in the PDATA= data set, and the parameters can appear in any order. The starting values are collected by parameter and arranged in ascending order, and duplicate values are removed. The parameter names in the PDATA= data set are not case sensitive. For example, the following DATA step defines starting values for three parameters and a starting grid with $1 \times 3 \times 1 = 3$ points:

```
data test;
   input Parameter $ Estimate;
   datalines;
alpha  100
 BETA   4
 beta   4.1
beta    4.2
beta    4.1
 gamma 30
;
```

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing, and requests detailed results about the performance characteristics of the procedure.

The PERFORMANCE statement is documented further in the section "PERFORMANCE Statement" on page 569.

PREDICT Statement

PREDICT '*label*' *expression* < *options* > **;**

PREDICT '*label*' **MEAN** < *options* > **;**

The PREDICT statement enables you to construct predictions of an expression across all of the observations in the input data set. Multiple PREDICT statements are permitted. Results for all PREDICT statements are placed in the output data set that you specify in the OUT= option in the PROC HPNLMOD statement. For more information, see the section "Output Data Sets" on page 568.

You must specify the following arguments:

'*label*'
: identifies the predicted expression.

expression | **MEAN**
: provides the predicted value.You can specify the predicted value either by using a SAS programming expression that involves the input data set variables and parameters or by using the keyword MEAN. If you specify the keyword MEAN, the predicted mean value for the distribution specified in the MODEL statement is used. Predicted values are computed using the final parameter estimates. Standard errors of prediction are computed using the delta method (Billingsley 1986; Cox 1998).

You can also specify the following *options*:

ALPHA=α
specifies the alpha level to be used to compute confidence limits. The default value corresponds to the ALPHA= option in the PROC HPNLMOD statement.

DF=*d*
specifies the degrees of freedom to be used to compute confidence limits. The default value corresponds to the DF= option in the PROC HPNLMOD statement.

LOWER=*name*
specifies a variable that contains the lower confidence limit of the predicted value.

PRED=*name*
specifies a variable that contains the predicted value.

PROBT=*name*
specifies a variable that contains the p-value of the predicted value.

STDERR=*name*
specifies a variable that contains the standard error of the predicted value.

TVALUE=*name*
specifies a variable that contains the t statistic for the predicted value.

UPPER=*name*
specifies a variable that contains the upper confidence limit of the predicted value.

RESTRICT Statement

RESTRICT *restriction1* < , *restriction2* . . . > **;**

The RESTRICT statement imposes linear restrictions on the model's parameters estimates. You can specify any number of RESTRICT statements.

Each *restriction* is written as an expression, optionally followed by an equality operator (=) or an inequality operator (<, >, <=, >=), followed by a second expression as follows:

expression < *operator expression* >

The *operator* can be =, <, >, <= , or >=. The operator and second expression are optional. When they are omitted, the *operator* defaults to = and the second *expression* defaults to the value 0.

Restriction expressions can be composed of parameter names, arithmetic operators, functions, and constants. Comparison operators (such as = or <) and logical operators (such as &) cannot be used in RESTRICT statement expressions. Parameters that are named in restriction expressions must be among the parameters that are estimated by the model. Restriction expressions cannot refer to other variables that are defined in the program or the DATA= data set. The restriction expressions must be linear functions of the parameters.

The following example illustrates how to use the RESTRICT statement to impose a linear constraint on parameters:

```
 proc hpnlmod;
    parms alpha beta;
    f = (x/alpha + beta)**2
    model y ~ residual(f);
    restrict beta < 2*(alpha + constant('pi'));
run;
```

The preceding RESTRICT statement represents the following model constraint:

$$\beta < 2(\alpha + \pi)$$

Programming Statements

Programming statements define the arguments of the MODEL, ESTIMATE, and PREDICT statements in PROC HPNLMOD. Most of the programming statements that can be used in the SAS DATA step can also be used in the HPNLMOD procedure. See *SAS Language Reference: Concepts* for a description of SAS programming statements. The following are valid programming statements:

ABORT;
CALL *name* **[(** *expression* **[,** *expression* **. . .])];**
DELETE;
DO[*variable* **=** *expression*
[TO *expression***] [BY** *expression***]**
[, *expression* **[TO** *expression***] [BY** *expression* **] . . .]**
]
[WHILE *expression* **] [UNTIL** *expression* **] ;**
END;
GOTO *statement_label***;**
IF *expression***;**
IF *expression* **THEN** *program_statement***;**
ELSE *program_statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement_label***;**
PUT [*variable***] [=] [. . .];**
RETURN;
SELECT[(*expression***)];**
STOP;
SUBSTR(*variable***,** *index***,** *length* **)=** *expression***;**
WHEN (*expression***)** *program_statement***;**
OTHERWISE *program_statement***;**

For the most part, the SAS programming statements work the same as they do in the SAS DATA step, as documented in *SAS Language Reference: Concepts*. However, they differ as follows:

- The ABORT statement does not allow any arguments.
- The DO statement does not allow a character index variable. Thus, the first of the following statements is supported, but the second is not:

```
do i = 1,2,3;
```

```
do i = 'A','B','C';
```

- In contrast to other procedures that share PROC HPNLMOD's programming syntax, PROC HPNLMOD does not support the LAG function. Because observations are not processed sequentially when high-performance analytical procedures perform the parameter optimization, information for computing lagged values is not available.
- The PUT statement, used mostly for program debugging in PROC HPNLMOD, supports only some of the features of the DATA step PUT statement, and it has some new features that the DATA step PUT statement does not have:
  - The PROC HPNLMOD PUT statement does not support line pointers, factored lists, iteration factors, overprinting, _INFILE_, the colon (:) format modifier, or "$".
  - The PROC HPNLMOD PUT statement supports expressions, but the expression must be enclosed in parentheses. For example, the following statement displays the square root of x:

    ```
    put  (sqrt(x));
    ```

  - The PROC HPNLMOD PUT statement supports the item _PDV_, which displays a formatted listing of all variables in the program. For example, the following statement displays a much more readable listing of the variables than the _ALL_ print item:

    ```
    put _pdv_;
    ```

- The WHEN and OTHERWISE statements enable you to specify more than one programming statement. That is, DO/END groups are not necessary for multiple WHEN statements. For example, the following syntax is valid:

```
select;
   when (exp1) stmt1;
               stmt2;
   when (exp2) stmt3;
               stmt4;
end;
```

When you code your programming statements, avoid defining variables that begin with an underscore (_) because they might conflict with internal variables that are created by PROC HPNLMOD. The MODEL statement must come after any SAS programming statements that define or modify terms that are used to specify the model.

Special Variable _WEIGHT_

Defining the special variable _WEIGHT_ in your PROC HPNLMOD programming statements enables you either to request weighted nonlinear least squares estimation of parameters or to request that each observation in the input data set be weighted by a value.

You can define the _WEIGHT_ variable as in the following statement:

```
_weight_ = expression;
```

When you specify this statement, the expression on the right-hand side of the assignment statement is evaluated for each observation in the data set. You can simply provide a variable name instead of an expression. PROC HPNLMOD uses the value of the _WEIGHT_ variable in two ways, depending on which type of model you request:

- When you do not specify a particular distribution, PROC HPNLMOD performs the weighted nonlinear least squares method by using the value of the _WEIGHT_ expression for each observation. These values multiplied by $1/\sigma^2$ are taken as inverse elements of the diagonal variance-covariance matrix of the dependent variable. The larger the _WEIGHT_ value, the more importance the observation is given.
- When you specify either a built-in distribution function or a general likelihood function, PROC HPNLMOD performs the maximum likelihood method by weighting each observation in the input data set by the value of the _WEIGHT_ expression.

Note that the _WEIGHT_ expression can be a function of the estimated parameters. For estimation purposes, the derivative of the _WEIGHT_ expression with respect to the parameters is not included in the gradient and Hessian of the loss function. This is usually the desired approach for iteratively reweighted least squares estimation. When the _WEIGHT_ expression is a function of the parameters, the gradient and Hessian that are used can lead to poor convergence or nonconvergence of the requested estimation. To have the derivative of the _WEIGHT_ expression with respect to the parameters included in the gradient and Hessian of the loss function, do not use the _WEIGHT_ programming statement. Instead, redefine the model as

$$(y - f(x, \beta)) \times \sqrt{\mathrm{wgt}(\beta)}$$

where y is the original dependent variable, $f(x, \beta)$ is the nonlinear model, and $\mathrm{wgt}(\beta)$ is the weight that is a function of the parameters.

Details: HPNLMOD Procedure

Least Squares Estimation

Models that are estimated by PROC HPNLMOD can be represented by using the equations

$$\begin{aligned} \mathbf{Y} &= \mathbf{f}(\boldsymbol{\beta}; \mathbf{z}_1, \ldots, \mathbf{z}_k) + \boldsymbol{\epsilon} \\ \mathrm{E}[\boldsymbol{\epsilon}] &= \mathbf{0} \\ \mathrm{Var}[\boldsymbol{\epsilon}] &= \sigma^2 \mathbf{I} \end{aligned}$$

where

$\mathbf{Y}$	is the $(n \times 1)$ vector of observed responses
$\mathbf{f}$	is the nonlinear prediction function of parameters and regressor variables
$\boldsymbol{\beta}$	is the vector of model parameters to be estimated
$\mathbf{z}_1, \ldots, \mathbf{z}_k$	are the $(n \times 1)$ vectors for each of the k regressor variables
$\boldsymbol{\epsilon}$	is the $(n \times 1)$ vector of residuals
σ^2	is the variance of the residuals

In these models, the distribution of the residuals is not specified and the model parameters are estimated using the least squares method. For the standard errors and confidence limits in the "ParameterEstimates" table to apply, the errors are assumed to be homoscedastic, uncorrelated, and have zero mean.

Built-In Log-Likelihood Functions

For models in which the distribution of model errors is specified, the HPNLMOD procedure estimates parameters by maximizing the value of a log-likelihood function for the specified distribution. The log-likelihood functions used by PROC HPNLMOD for the supported error distributions are as follows:

$Y \sim \mathrm{normal}(m, v)$

$$\begin{aligned} l(m, v; y) &= -\frac{1}{2}\left(\log\{2\pi\} + \frac{(y-m)^2}{v} + \log\{v\}\right) \\ \mathrm{E}[Y] &= m \\ \mathrm{Var}[Y] &= v \\ v &> 0 \end{aligned}$$

$Y \sim \text{binary}(p)$

$$
\begin{aligned}
l_1(p;y) &= \begin{cases} y \ \log\{p\} & y > 0 \\ 0 & \text{otherwise} \end{cases} \\
l_2(p;y) &= \begin{cases} (1-y) \ \log\{1-p\} & y < 1 \\ 0 & \text{otherwise} \end{cases} \\
l(p;y) &= l_1(p;y) + l_2(p;y) \\
\text{E}[Y] &= p \\
\text{Var}[Y] &= p\,(1-p) \\
0 &< p < 1
\end{aligned}
$$

$Y \sim \text{binomial}(n, p)$

$$
\begin{aligned}
l_c &= \log\{\Gamma(n+1)\} - \log\{\Gamma(y+1)\} - \log\{\Gamma(n-y+1)\} \\
l_1(n,p;y) &= \begin{cases} y \ \log\{p\} & y > 0 \\ 0 & \text{otherwise} \end{cases} \\
l_2(n,p;y) &= \begin{cases} (n-y) \ \log\{1-p\} & n-y > 0 \\ 0 & \text{otherwise} \end{cases} \\
l(n,p;y) &= l_c + l_1(n,p;y) + l_2(n,p;y) \\
\text{E}[Y] &= n\,p \\
\text{Var}[Y] &= n\,p\,(1-p) \\
0 &< p < 1
\end{aligned}
$$

$Y \sim \text{gamma}(a, b)$

$$
\begin{aligned}
l(a,b;y) &= -a\log\{b\} - \log\{\Gamma(a)\} + (a-1)\log\{y\} - y/b \\
\text{E}[Y] &= ab \\
\text{Var}[Y] &= ab^2 \\
a &> 0 \\
b &> 0
\end{aligned}
$$

This parameterization of the gamma distribution differs from the parameterization that the GLIMMIX and GENMOD procedures use. The scale parameter in PROC HPNLMOD is expressed as the inverse of the scale parameter that PROC GLIMMIX and PROC GENMOD use. The PROC HPNLMOD parameter represents the scale of the magnitude of the residuals. The scale parameter in PROC GLIMMIX can be estimated by using the following statements:

```
proc glimmix;
   model y = x / dist=gamma s;
run;
```

The following statements show how to use PROC HPNLMOD to estimate the equivalent scale parameter:

```
proc hpnlmod;
   parms b0=1 b1=0 scale=14;
   linp = b0 + b1*x;
   mu   = exp(linp);
   b    = mu*scale;
   model y ~ gamma(1/scale,b);
run;
```

$Y \sim \text{negbin}(n, p)$

$$
\begin{aligned}
l(n, p; y) &= \log\{\Gamma(n+y)\} - \log\{\Gamma(n)\} - \log\{\Gamma(y+1)\} \\
&\quad + n\log\{p\} + y\log\{1-p\} \\
\mathrm{E}[Y] &= n\left(\frac{1-p}{p}\right) \\
\mathrm{Var}[Y] &= n\left(\frac{1-p}{p^2}\right) \\
n &\geq 0 \\
0 &< p < 1
\end{aligned}
$$

The parameter n can be real-numbered; it does not have to be integer-valued.

$Y \sim \text{Poisson}(m)$

$$
\begin{aligned}
l(m; y) &= y\log\{m\} - m - \log\{\Gamma(y+1)\} \\
\mathrm{E}[Y] &= m \\
\mathrm{Var}[Y] &= m \\
m &> 0
\end{aligned}
$$

Computational Method

Multithreading

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to sequential (single-threaded) execution.

The number of threads that the HPNLMOD procedure spawns is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the CPU count by using the CPUCOUNT= SAS system option. For example, if you specify the following statement, the HPNLMOD procedure determines threading as if it executed on a system that has four CPUs, regardless of the actual CPU count:

```
options cpucount=4;
```

- You can specify the NTHREADS= option in the PERFORMANCE statement to determine the number of threads. This specification overrides the CPUCOUNT= system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads used is displayed in the "Performance Information" table, which is part of the default output. The HPNLMOD procedure allocates one thread per CPU.

The HPNLMOD procedure divides the data that are processed on a single machine among the threads—that is, the HPNLMOD procedure implements multithreading by distributing computations across the data. For example, if the input data set has 1,000 observations and PROC HPNLMOD is running with four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following:

- calculation of objective function values for the initial parameter grid
- objective function calculation
- gradient calculation
- Hessian calculation
- scoring of observations

In addition, operations on matrices such as sweeps might be multithreaded, provided that the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Choosing an Optimization Algorithm

First- or Second-Order Algorithms

The factors that affect how you choose a particular optimization technique for a particular problem are complex. Occasionally, you might benefit from trying several algorithms.

For many optimization problems, computing the gradient takes more computer time than computing the function value. Computing the Hessian sometimes takes *much* more computer time and memory than computing the gradient, especially when there are many decision variables. Unfortunately, optimization techniques that do not use some kind of Hessian approximation usually require many more iterations than techniques that do use a Hessian matrix; as a result, the total run time of these techniques is often longer. Techniques that do not use the Hessian also tend to be less reliable. For example, they can terminate more easily at stationary points than at global optima.

Table 63.3 shows which derivatives are required for each optimization technique.

Table 63.3 Derivatives Required

Algorithm	First-Order	Second-Order
TRUREG	x	x
NEWRAP	x	x
NRRIDG	x	x
QUANEW	x	-
DBLDOG	x	-
CONGRA	x	-
LEVMAR	x	-
NMSIMP	-	-

The second-derivative methods (TRUREG, NEWRAP, and NRRIDG) are best for small problems for which the Hessian matrix is not expensive to compute. Sometimes the NRRIDG algorithm can be faster than the TRUREG algorithm, but TRUREG can be more stable. The NRRIDG algorithm requires only one matrix with $p(p+1)/2$ double words; TRUREG and NEWRAP require two such matrices. Here, p denotes the number of parameters in the optimization.

The first-derivative methods QUANEW and DBLDOG are best for medium-sized problems for which the objective function and the gradient are much faster to evaluate than the Hessian. In general, the QUANEW and DBLDOG algorithms require more iterations than TRUREG, NRRIDG, and NEWRAP, but each iteration can be much faster. The QUANEW and DBLDOG algorithms require only the gradient to update an approximate Hessian, and they require slightly less memory than TRUREG or NEWRAP.

The first-derivative method CONGRA is best for large problems for which the objective function and the gradient can be computed much faster than the Hessian and for which too much memory is required to store the (approximate) Hessian. In general, the CONGRA algorithm requires more iterations than QUANEW or DBLDOG, but each iteration can be much faster. Because CONGRA requires only a factor of p double-word memory, many large applications can be solved only by CONGRA.

The no-derivative method NMSIMP is best for small problems for which derivatives are not continuous or are very difficult to compute.

The LEVMAR method is appropriate only for least squares optimization problems.

Each optimization method uses one or more convergence criteria that determine when it has converged. An algorithm is considered to have converged when any one of the convergence criteria is satisfied. For example, under the default settings, the QUANEW algorithm converges if ABSGCONV <1E−5, FCONV $< 2 \times \epsilon$, or GCONV <1E−8.

By default, the HPNLMOD procedure applies the NRRIDG algorithm because it can take advantage of multithreading in Hessian computations and inversions. If the number of parameters becomes large, specifying TECHNIQUE=QUANEW (which is a first-order method with good overall properties) is recommended.

Algorithm Descriptions

The following subsections provide details about each optimization technique and follow the same order as Table 63.3.

Trust Region Optimization (TRUREG)

The trust region method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function $f(\boldsymbol{\psi})$ have continuous first- and second-order derivatives inside the feasible region.

The trust region method iteratively optimizes a quadratic approximation to the nonlinear objective function within a hyperelliptic trust region that has radius Δ. The radius constrains the step size that corresponds to the quality of the quadratic approximation. The trust region method is implemented based on Dennis, Gay, and Welsch (1981); Gay (1983); Moré and Sorensen (1983).

The trust region method performs well for small- to medium-sized problems, and it does not need many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the quasi-Newton or conjugate gradient algorithms might be more efficient.

Newton-Raphson Optimization with Line Search (NEWRAP)

The NEWRAP technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

If second-order derivatives are computed efficiently and precisely, the NEWRAP method can perform well for medium-sized to large problems, and it does not need many function, gradient, and Hessian calls.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. Otherwise, a combination of ridging and line search is performed to compute successful steps. If the Hessian is not positive-definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive-definite (Eskow and Schnabel 1991).

In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation (LIS=2).

Newton-Raphson Ridge Optimization (NRRIDG)

The NRRIDG technique uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\psi}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

This algorithm uses a pure Newton step when the Hessian is positive-definite and when the Newton step reduces the value of the objective function successfully. If at least one of these two conditions is not satisfied, a multiple of the identity matrix is added to the Hessian matrix.

The NRRIDG method performs well for small- to medium-sized problems, and it does not require many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the quasi-Newton or conjugate gradient algorithms might be more efficient.

Because the NRRIDG technique uses an orthogonal decomposition of the approximate Hessian, each iteration of NRRIDG can be slower than an iteration of the NEWRAP technique, which works with a Cholesky decomposition. However, NRRIDG usually requires fewer iterations than NEWRAP.

Quasi-Newton Optimization (QUANEW)

The (dual) quasi-Newton method uses the gradient $\mathbf{g}(\boldsymbol{\psi}^{(k)})$, and it does not need to compute second-order derivatives because they are approximated. It works well for medium-sized to moderately large optimization problems, where the objective function and the gradient can be computed much faster than the Hessian. However, in general it requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which compute second-order derivatives. QUANEW is the default optimization algorithm because it provides an appropriate balance between the speed and stability that are required for most nonlinear mixed model applications.

The QUANEW technique that is implemented by the HPNLMOD procedure is the dual quasi-Newton algorithm, which updates the Cholesky factor of an approximate Hessian.

In each iteration, a line search is performed along the search direction to find an approximate optimum. The line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions (Fletcher 1987). One of the Goldstein conditions can be violated if the feasible region defines an upper limit of the step size. Violating the left-side Goldstein condition can affect the positive-definiteness of the quasi-Newton update. In that case, either the update is skipped or the iterations are restarted by using an identity matrix, resulting in the steepest descent or ascent search direction.

The QUANEW algorithm uses its own line-search technique.

Double-Dogleg Optimization (DBLDOG)

The double-dogleg optimization method combines the ideas of the quasi-Newton and trust region methods. In each iteration, the double-dogleg algorithm computes the step $\mathbf{s}^{(k)}$ as the linear combination of the steepest descent or ascent search direction $\mathbf{s}_1^{(k)}$ and a quasi-Newton search direction $\mathbf{s}_2^{(k)}$:

$$\mathbf{s}^{(k)} = \alpha_1 \mathbf{s}_1^{(k)} + \alpha_2 \mathbf{s}_2^{(k)}$$

The step is requested to remain within a prespecified trust region radius (Fletcher 1987, p. 107). Thus, the DBLDOG subroutine uses the dual quasi-Newton update but does not perform a line search.

The double-dogleg optimization technique works well for medium-sized to moderately large optimization problems, where the objective function and the gradient are much faster to compute than the Hessian. The implementation is based on Dennis and Mei (1979); Gay (1983), but it is extended for dealing with boundary and linear constraints. The DBLDOG technique generally requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which require second-order derivatives; however, each of the DBLDOG iterations is computationally cheap. Furthermore, the DBLDOG technique requires only gradient calls for the update of the Cholesky factor of an approximate Hessian.

Conjugate Gradient Optimization (CONGRA)

Second-order derivatives are not required by the CONGRA algorithm and are not even approximated. The CONGRA algorithm can be expensive in function and gradient calls, but it requires only $O(p)$ memory for unconstrained optimization. In general, the algorithm must perform many iterations to obtain a precise solution, but each of the CONGRA iterations is computationally cheap.

The CONGRA algorithm should be used for optimization problems that have large p. For the unconstrained or boundary-constrained case, the CONGRA algorithm requires only $O(p)$ bytes of working memory, whereas all other optimization methods require order $O(p^2)$ bytes of working memory. During p successive iterations, uninterrupted by restarts or changes in the working set, the CONGRA algorithm computes a cycle of p conjugate search directions. In each iteration, a line search is performed along the search direction to find an

approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α that satisfies the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit for the step size. Other line-search algorithms can be specified with the LIS= option.

Levenberg-Marquardt Optimization (LEVMAR)

The LEVMAR algorithm performs a highly stable optimization; however, for large problems, it consumes more memory and takes longer than the other techniques. The Levenberg-Marquardt optimization technique is a slightly improved variant of the Moré (1978) implementation.

Nelder-Mead Simplex Optimization (NMSIMP)

The Nelder-Mead simplex method does not use any derivatives and does not assume that the objective function has continuous derivatives. The objective function itself needs to be continuous. This technique is quite expensive in the number of function calls, and it might be unable to generate precise results for $p \gg 40$.

The original Nelder-Mead simplex algorithm is implemented and extended to boundary constraints. This algorithm does not compute the objective for infeasible points, but it changes the shape of the simplex by adapting to the nonlinearities of the objective function. This adaptation contributes to an increased speed of convergence. NMSIMP uses a special termination criterion.

Displayed Output

The following sections describe the output that PROC HPNLMOD produces by default. The output is organized into various tables, which are discussed in the order of their appearance.

Performance Information

The “Performance Information” table is produced by default. It displays the number of threads used.

If you specify the DETAILS option in the PERFORMANCE statement, the procedure also produces a “Timing” table in which elapsed times (absolute and relative) for the main tasks of the procedure are displayed.

Specifications

The “Specifications” table displays basic information about the model such as the data source, the dependent variable, the distribution being modeled, and the optimization technique.

Number of Observations

The “Number of Observations” table displays the number of observations that are read from the input data set and the number of observations that are used in the analysis.

Dimensions

The "Dimensions" table displays the number of parameters that are estimated in the model and the number of upper and lower bounds that are imposed on the parameters.

Parameters

The "Parameters" table displays the initial values of parameters that are used to start the estimation process. You can limit this information by specifying the BEST= option in the PARAMETERS statement when you specify a large number of initial parameter value combinations. The parameter combinations and their corresponding objective function values are listed in increasing order of objective function value.

Iteration History

For each iteration of the optimization, the "Iteration History" table displays the number of function evaluations (including gradient and Hessian evaluations), the value of the objective function, the change in the objective function from the previous iteration, and the absolute value of the largest (projected) gradient element.

Convergence Status

The convergence status table is a small ODS table that follows the "Iteration History" table in the default output. In the listing it appears as a message that identifies whether the optimization succeeded and which convergence criterion was met. If the optimization fails, the message indicates the reason for the failure. If you save the convergence status table to an output data set, a numeric Status variable is added that enables you to programmatically assess convergence. The values of the Status variable encode the following:

0	Convergence was achieved or an optimization was not performed because TECHNIQUE=NONE.
1	The objective function could not be improved.
2	Convergence was not achieved because of a user interrupt or because a limit (such as the maximum number of iterations or the maximum number of function evaluations) was reached. To modify these limits, see the MAXITER=, MAXFUNC=, and MAXTIME= options in the PROC HPNLMOD statement.
3	Optimization failed to converge because function or derivative evaluations failed at the starting values or during the iterations or because a feasible point that satisfies the parameter constraints could not be found in the parameter space.

Linear Constraints

The "Linear Constraints" table summarizes the linear constraints that are applied to the model by using the RESTRICT statements. All the constraints that are specified in the model are listed in the "Linear Constraints" table, together with information about whether each constraint represents an inequality or equality condition and whether that constraint is active for the final parameter estimates.

Fit Statistics

The "Fit Statistics" table displays a variety of measures of fit, depending on whether the model was estimated using least squares or maximum likelihood. In both cases, smaller values of the fit statistics indicate better fit.

For least squares estimations, the "Fit Statistics" table displays the sum of squares of errors and the variance of errors.

For maximum likelihood estimations, the table uses the following formulas to display information criteria, where p denotes the number of effective parameters, n denotes the number of observations used, and l is the log likelihood that is evaluated at the converged estimates:

$$\mathrm{AIC} = -2l + 2p$$

$$\mathrm{AICC} = \begin{cases} -2l + 2pn/(n-p-1) & f > p+2 \\ -2l + 2p(p+2) & \text{otherwise} \end{cases}$$

$$\mathrm{BIC} = -2l + p\log(f)$$

The information criteria values that are displayed in the "Fit Statistics" table are not based on a normalized log-likelihood function.

ANOVA

The "Analysis of Variance" table is displayed only for least squares estimations. The ANOVA table displays the number of degrees of freedom and the sum of squares that are attributed to the model, the error, and the total. The ANOVA table also reports the variance of the model and the errors, the F statistic, and its probability for the model.

Parameter Estimates

The "Parameter Estimates" table displays the parameter estimates, their estimated (asymptotic) standard errors t statistics, and associated p-values for the hypothesis that the parameter is 0. Confidence limits are displayed for each parameter and are based on the value of the ALPHA= option in the PROC HPNLMOD statement.

Additional Estimates

The "Additional Estimates" table displays the same information as the "Parameter Estimates" table for the expressions that appear in the optional ESTIMATE statements. The table is generated when one or more ESTIMATE statements are specified. Because a separate ALPHA= option can be specified for each ESTIMATE statement, the "Additional Estimates" table also includes a column that indicates each confidence interval's corresponding significance level.

Covariance

The "Covariance" table appears when the COV option is specified in the PROC HPNLMOD statement. The "Covariance" table displays a matrix of covariances between each pair of estimated parameters.

Correlation

The "Correlation" table appears when the CORR option is specified in the PROC HPNLMOD statement. The "Correlation" table displays the correlation matrix for the estimated parameters.

Additional Estimates Covariance

The "Covariance of Additional Estimates" table appears when the ECOV option is specified in the PROC HPNLMOD statement. The "Covariance of Additional Estimates" table displays a matrix of covariances between each pair of expressions that are specified in ESTIMATE statements.

Additional Estimates Correlation

The "Correlation of Additional Estimates" table appears when the ECORR option is specified in the PROC HPNLMOD statement. The "Correlation of Additional Estimates" table displays the correlation matrix for the expressions that are specified in ESTIMATE statements.

Procedure Task Timing

If you specify the DETAILS option in the PERFORMANCE statement, the procedure also produces a "Procedure Task Timing" table in which elapsed times (absolute and relative) for the main tasks of the procedure are displayed.

ODS Table Names

Each table that is created by the HPNLMOD procedure has a name associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 63.4.

Table 63.4 ODS Tables Produced by PROC HPNLMOD

Table Name	Description	Required Statement and Option
AdditionalEstimates	Functions of estimated parameters and their associated statistics	ESTIMATE statement
ANOVA	Least squares analysis of variance information	RESIDUAL option in the MODEL statement
Constraints	Information about the model's linear constraints	RESTRICT statement
ConvergenceStatus	Optimization success and convergence information	Default output
CorrB	Parameter correlation matrix	CORR option in the PROC HPNLMOD statement
CovB	Parameter covariance matrix	COV option in the PROC HPNLMOD statement
Dimensions	Number of parameters and their bounds	Default output

Table 63.4 *continued*

Table Name	Description	Required Statement and Option
ECorrB	Additional estimates' correlation matrix	ECORR option in the PROC HPNLMOD statement
ECovB	Additional estimates' covariance matrix	ECOV option in the PROC HPNLMOD statement
FitStatistics	Statistics about the quality of the fit	Default output
IterHistory	Optimizer iteration information	Default output
NObs	Number of observations read and used	Default output
ParameterEstimates	Parameter estimates and associated statistics	Default output
Parameters	Initial parameter values	Default output
PerformanceInfo	Information about high-performance computing environment	Default output
Specifications	Basic model characteristics	Default output

Examples: HPNLMOD Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 63.1: Segmented Model

(View the complete code for this example (hpnlnex1.sas) in the example repository.)

Suppose you are interested in fitting a model that consists of two segments that connect in a smooth fashion. For example, the following model states that the mean of Y is a quadratic function in x for values of x less than x_0 and that the mean of Y is constant for values of x greater than x_0:

$$\mathrm{E}[Y|x] = \begin{cases} \alpha + \beta x + \gamma x^2 & \text{if } x < x_0 \\ c & \text{if } x \geq x_0 \end{cases}$$

In this model equation, α, β, and γ are the coefficients of the quadratic segment, and c is the plateau of the mean function. The HPNLMOD procedure can fit such a segmented model even when the join point, x_0, is unknown.

Suppose you also want to impose conditions on the two segments of the model. First, the curve should be continuous—that is, the quadratic and the plateau section need to meet at x_0. Second, the curve should be smooth—that is, the first derivative of the two segments with respect to x needs to coincide at x_0.

The continuity condition requires that

$$c = \mathrm{E}[Y|x_0] = \alpha + \beta x_0 + \gamma x_0^2$$

The smoothness condition requires that

$$\frac{\partial \mathrm{E}[Y|x_0]}{\partial x} = \beta + 2\gamma x_0 \equiv 0$$

If you solve for x_0 and substitute into the expression for c, the two conditions jointly imply that

$$x_0 = -\beta/2\gamma$$
$$c = \alpha - \beta^2/4\gamma$$

Although there are five unknowns, the model contains only three independent parameters. The continuity and smoothness restrictions together completely determine two parameters, given the other three.

The following DATA step creates the SAS data set for this example:

```
data a;
   input y x @@;
   datalines;
.46 1  .47  2 .57  3 .61  4 .62  5 .68  6 .69  7
.78 8  .70  9 .74 10 .77 11 .78 12 .74 13 .80 13
.80 15 .78 16
 ;
```

The following PROC HPNLMOD statements fit this segmented model:

```
proc hpnlmod data=a out=b;
   parms alpha=.45 beta=.05 gamma=-.0025;

   x0 = -.5*beta / gamma;

   if (x < x0) then
        yp = alpha + beta*x  + gamma*x*x;
   else
        yp = alpha + beta*x0 + gamma*x0*x0;

   model y ~ residual(yp);

   estimate 'join point' -beta/2/gamma;
   estimate 'plateau value c' alpha - beta**2/(4*gamma);
   predict 'predicted' yp pred=yp;
   predict 'response' y pred=y;
   predict 'x' x pred=x;
run;
```

The parameters of the model are α, β, and γ. They are represented in the PROC HPNLMOD statements by the variables alpha, beta, and gamma, respectively. In order to model the two segments, a conditional statement assigns the appropriate expression to the mean function, depending on the value of x_0. The ESTIMATE statements compute the values of x_0 and c. The PREDICT statement computes predicted values for plotting and saves them to data set b.

The results from fitting this model are shown in Output 63.1.1 through Output 63.1.3. The iterative optimization converges after six iterations (Output 63.1.1). Output 63.1.2 shows the estimated parameters. Output 63.1.3 indicates that the join point is 12.7477 and the plateau value is 0.7775.

Output 63.1.1 Nonlinear Least Squares Iterative Phase

Quadratic Model with Plateau

The HPNLMOD Procedure

Iteration History

Iteration	Evaluations	Objective Function	Change	Max Gradient
0	5	0.0035144531		7.184063
1	2	0.0007352716	0.00277918	2.145337
2	2	0.0006292751	0.00010600	0.032551
3	2	0.0006291261	0.00000015	0.002952
4	2	0.0006291244	0.00000000	0.000238
5	2	0.0006291244	0.00000000	0.000023
6	2	0.0006291244	0.00000000	2.313E-6

Convergence criterion (GCONV=1E-8) satisfied.

Output 63.1.2 Least Squares Analysis for the Quadratic Model

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	0.1769	0.0884	114.22	<.0001
Error	13	0.0101	0.000774		
Corrected Total	15	0.1869			

Parameter Estimates

Parameter	Estimate	Standard Error	DF	t Value	Approx Pr > |t|	Approximate 95% Confidence Limits	
alpha	0.3921	0.0267	1	14.70	<.0001	0.3345	0.4497
beta	0.0605	0.00842	1	7.18	<.0001	0.0423	0.0787
gamma	-0.00237	0.000551	1	-4.30	0.0009	-0.00356	-0.00118

Output 63.1.3 Additional Estimates for the Quadratic Model

Additional Estimates

Label	Estimate	Standard Error	DF	t Value	Approx Pr > |t|	Alpha	Approximate Confidence Limits	
join point	12.7477	1.2781	1	9.97	<.0001	0.05	9.9864	15.5089
plateau value c	0.7775	0.0123	1	63.11	<.0001	0.05	0.7509	0.8041

The following statements produce a graph of the observed and predicted values along with reference lines for

the join point and plateau estimates (Output 63.1.4):

```
proc sgplot data=b noautolegend;
   yaxis label='Observed or Predicted';
   refline 0.7775  / axis=y label="Plateau"    labelpos=min;
   refline 12.7477 / axis=x label="Join point" labelpos=min;
   scatter y=y  x=x;
   series  y=yp x=x;
run;
```

Output 63.1.4 Observed and Predicted Values for the Quadratic Model

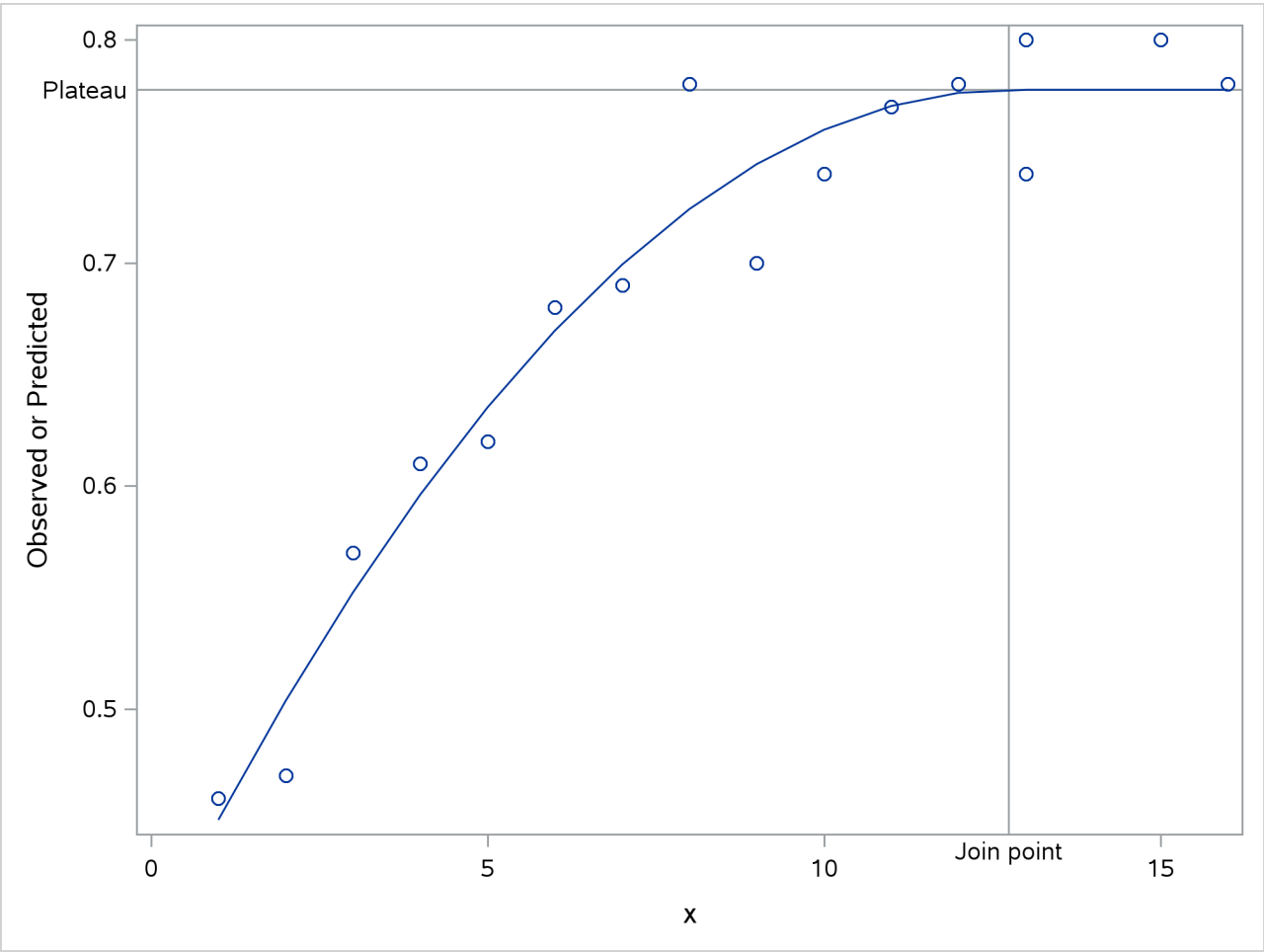

References

Billingsley, P. (1986). *Probability and Measure*. 2nd ed. New York: John Wiley & Sons.

Cox, C. (1998). "Delta Method." In *Encyclopedia of Biostatistics*, edited by P. Armitage and T. Colton. New York: John Wiley & Sons.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981). "An Adaptive Nonlinear Least-Squares Algorithm." *ACM Transactions on Mathematical Software* 7:348–368.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Eskow, E., and Schnabel, R. B. (1991). "Algorithm 695: Software for a New Modified Cholesky Factorization." *ACM Transactions on Mathematical Software* 17:306–312.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Moré, J. J. (1978). "The Levenberg-Marquardt Algorithm: Implementation and Theory." In *Lecture Notes in Mathematics*, vol. 30, edited by G. A. Watson, 105–116. Berlin: Springer-Verlag.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Ratkowsky, D. (1990). *Handbook of Nonlinear Regression Models*. New York: Marcel Dekker.

Chapter 64
The HPPLS Procedure

Contents

Overview: HPPLS Procedure . . . **5016**
PROC HPPLS Features . . . 5017
PROC HPPLS Contrasted with PROC PLS . . . 5017
Getting Started: HPPLS Procedure . . . **5018**
Spectrometric Calibration . . . 5018
Fitting a PLS Model . . . 5020
Selecting the Number of Factors by Test Set Validation . . . 5021
Predicting New Observations . . . 5024
Syntax: HPPLS Procedure . . . **5026**
PROC HPPLS Statement . . . 5026
BY Statement . . . 5029
CLASS Statement . . . 5030
ID Statement . . . 5030
MODEL Statement . . . 5030
OUTPUT Statement . . . 5031
PARTITION Statement . . . 5033
PERFORMANCE Statement . . . 5034
Details: HPPLS Procedure . . . **5034**
Regression Methods . . . 5034
Partial Least Squares . . . 5034
SIMPLS . . . 5035
Principal Components Regression . . . 5036
Reduced Rank Regression . . . 5036
Relationships between Methods . . . 5036
Test Set Validation . . . 5039
Centering and Scaling . . . 5040
Missing Values . . . 5041
Computational Method . . . 5041
Multithreading . . . 5041
Output Data Set . . . 5042
Displayed Output . . . 5042
Performance Information . . . 5042
Data Access Information . . . 5042
Centering and Scaling Information . . . 5042
Model Information . . . 5042
Number of Observations . . . 5043

Class Level Information 5043
Dimensions 5043
Test Set Validation 5043
Percent Variation Accounted for by Extracted Factors 5043
Model Details 5043
Parameter Estimates 5044
Timing Information 5044
ODS Table Names 5044
Example: HPPLS Procedure **5045**
Example 64.1: Choosing a PLS Model by Test Set Validation 5045
References **5049**

Overview: HPPLS Procedure

The HPPLS procedure is a high-performance version of the PLS procedure in SAS/STAT software, which fits models by using any one of a number of linear predictive methods, including *partial least squares* (PLS). Ordinary least squares regression, as implemented in SAS/STAT procedures such as the GLM and REG procedures, has the single goal of minimizing sample response prediction error, and it seeks linear functions of the predictors that explain as much variation in each response as possible. The HPPLS procedure implements techniques that have the additional goal of accounting for variation in the predictors, under the assumption that directions in the predictor space that are well sampled should provide better prediction for *new* observations when the predictors are highly correlated. All the techniques that the HPPLS procedure implements work by extracting successive linear combinations of the predictors, called *factors* (also called *components*, *latent vectors*, or *latent variables*), which optimally address one or both of these two goals: explaining response variation and explaining predictor variation. In particular, the method of partial least squares balances the two objectives by seeking factors that explain both response and predictor variation.

The name "partial least squares" also applies to a more general statistical method that is *not* implemented in this procedure. The partial least squares method was originally developed in the 1960s by the econometrician Herman Wold (1966) for modeling "paths" of causal relation between any number of "blocks" of variables. However, the HPPLS procedure fits only *predictive* partial least squares models that have one "block" of predictors and one "block" of responses. If you are interested in fitting more general path models, you should consider using the CALIS procedure.

PROC HPPLS Features

The main features of the HPPLS procedure are as follows:

- supports GLM and reference parameterization for classification effects
- permits any degree of interaction effects that involve classification and continuous variables
- supports partitioning of data into training and testing roles
- supports test set validation to choose the number of extracted factors, where the model is fit to only part of the available data (the training set) and the fit is evaluated over the other part of the data (the test set)
- produces an output data set that contains predicted values and other observationwise statistics

The HPPLS procedure implements the following techniques:

- principal components regression, which extracts factors to explain as much predictor sample variation as possible
- reduced rank regression, which extracts factors to explain as much response variation as possible. This technique, also known as (maximum) redundancy analysis, differs from multivariate linear regression only when there are multiple responses.
- partial least squares regression, which balances the two objectives of explaining response variation and explaining predictor variation. Two different formulations for partial least squares are available: the original predictive method of Wold (1966) and the straightforward implementation of a statistically inspired modification of the partial least squares (SIMPLS) method of De Jong (1993).

Because the HPPLS procedure is a high-performance analytical procedure, it also exploits all the available cores and concurrent threads.

PROC HPPLS Contrasted with PROC PLS

The HPPLS procedure and the PLS procedure have the following similarities and differences:

- All the general factor extraction methods that are available in PROC PLS are supported by PROC HPPLS.
- The RLGW algorithm, which is available in PROC PLS to compute extracted PLS factors, is not supported by PROC HPPLS.
- PROC PLS can specify various methods to be used for cross validation. PROC HPPLS supports test set validation only by using the PARTITION statement.

- The CLASS statement in PROC HPPLS permits two parameterizations: the GLM-type parameterization and a reference parameterization. The HPPLS procedure does not mix parameterizations across the variables in the CLASS statement. In other words, all classification variables are in the same parameterization, and this parameterization is either the GLM or reference parameterization. In PROC PLS, only the GLM-type parameterization is supported.
- The HPPLS procedure does not support the EFFECT statement, the MISSING= option, the VARSCALE option, and the PLOTS option that are available in PROC PLS.
- The syntax of the OUTPUT statement in the HPPLS procedure is different from the syntax of the OUTPUT statement in PROC PLS. In the HPPLS procedure, you do not need to provide a prefix in the OUTPUT statement. A default prefix is used if you do not provide one. If you do not specify any output statistics in the OUTPUT statement in PROC HPPLS, the output data set includes the predicted values for response variables. Furthermore, although the OUTPUT statement in the PLS procedure includes the input and BY variables in the output data by default, PROC HPPLS does not include them by default so that it can avoid data duplication for large data sets. In order to include any input or BY variables in the output data set, you must list these variables in the ID statement.
- By default, PROC HPPLS performs computations on multiple threads. The PLS procedure executes on a single thread.

Getting Started: HPPLS Procedure

Spectrometric Calibration

The example in this section illustrates basic features of the HPPLS procedure. The data are reported in Umetrics (1995); the original source is Lindberg, Persson, and Wold (1983). Suppose you are researching pollution in the Baltic Sea and you want to use the spectra of samples of seawater to determine the amounts of three compounds present in seawater samples: lignin sulfonate (ls: pulp industry pollution), humic acids (ha: natural forest products), and optical whitener from detergent (dt). Spectrometric calibration is a type of problem in which partial least squares can be very effective. The predictors are the spectra emission intensities at different frequencies in a sample spectrum, and the responses are the amounts of various chemicals in the sample.

For the purposes of calibrating the model, samples that have known compositions are used. The calibration data consist of 16 samples of known concentrations of ls, ha, and dt, with spectra based on 27 frequencies (or, equivalently, wavelengths). The following statements create a SAS data set named Sample for these data. In order to demonstrate the use of test set validation, the data set contains a variable Role that is used to assign observations to the training and testing roles. In this case, the training role has nine samples and the testing role has seven samples.

```
data Sample;
   input obsnam $ v1-v27 ls ha dt Role $5. @@;
   datalines;
EM1   2766 2610 3306 3630 3600 3438 3213 3051 2907 2844 2796
      2787 2760 2754 2670 2520 2310 2100 1917 1755 1602 1467
```

```
      1353 1260 1167 1101 1017         3.0110  0.0000   0.00 TRAIN
EM2   1492 1419 1369 1158  958  887  905  929  920  887  800
       710  617  535  451  368  296  241  190  157  128  106
        89   70   65   56   50         0.0000  0.4005   0.00 TEST
EM3   2450 2379 2400 2055 1689 1355 1109  908  750  673  644
       640  630  618  571  512  440  368  305  247  196  156
       120   98   80   61   50         0.0000  0.0000  90.63 TRAIN
EM4   2751 2883 3492 3570 3282 2937 2634 2370 2187 2070 2007
      1974 1950 1890 1824 1680 1527 1350 1206 1080  984  888
       810  732  669  630  582         1.4820  0.1580  40.00 TEST
EM5   2652 2691 3225 3285 3033 2784 2520 2340 2235 2148 2094
      2049 2007 1917 1800 1650 1464 1299 1140 1020  909  810
       726  657  594  549  507         1.1160  0.4104  30.45 TEST
EM6   3993 4722 6147 6720 6531 5970 5382 4842 4470 4200 4077
      4008 3948 3864 3663 3390 3090 2787 2481 2241 2028 1830
      1680 1533 1440 1314 1227         3.3970  0.3032  50.82 TRAIN
EM7   4032 4350 5430 5763 5490 4974 4452 3990 3690 3474 3357
      3300 3213 3147 3000 2772 2490 2220 1980 1779 1599 1440
      1320 1200 1119 1032  957         2.4280  0.2981  70.59 TRAIN
EM8   4530 5190 6910 7580 7510 6930 6150 5490 4990 4670 4490
      4370 4300 4210 4000 3770 3420 3060 2760 2490 2230 2060
      1860 1700 1590 1490 1380         4.0240  0.1153  89.39 TRAIN
EM9   4077 4410 5460 5857 5607 5097 4605 4170 3864 3708 3588
      3537 3480 3330 3192 2910 2610 2325 2064 1830 1638 1476
      1350 1236 1122 1044  963         2.2750  0.5040  81.75 TEST
EM10  3450 3432 3969 4020 3678 3237 2814 2487 2205 2061 2001
      1965 1947 1890 1776 1635 1452 1278 1128  981  867  753
       663  600  552  507  468         0.9588  0.1450 101.10 TRAIN
EM11  4989 5301 6807 7425 7155 6525 5784 5166 4695 4380 4197
      4131 4077 3972 3777 3531 3168 2835 2517 2244 2004 1809
      1620 1470 1359 1266 1167         3.1900  0.2530 120.00 TRAIN
EM12  5340 5790 7590 8390 8310 7670 6890 6190 5700 5380 5200
      5110 5040 4900 4700 4390 3970 3540 3170 2810 2490 2240
      2060 1870 1700 1590 1470         4.1320  0.5691 117.70 TEST
EM13  3162 3477 4365 4650 4470 4107 3717 3432 3228 3093 3009
      2964 2916 2838 2694 2490 2253 2013 1788 1599 1431 1305
      1194 1077  990  927  855         2.1600  0.4360  27.59 TRAIN
EM14  4380 4695 6018 6510 6342 5760 5151 4596 4200 3948 3807
      3720 3672 3567 3438 3171 2880 2571 2280 2046 1857 1680
      1548 1413 1314 1200 1119         3.0940  0.2471  61.71 TRAIN
EM15  4587 4200 5040 5289 4965 4449 3939 3507 3174 2970 2850
      2814 2748 2670 2529 2328 2088 1851 1641 1431 1284 1134
      1020  918  840  756  714         1.6040  0.2856 108.80 TEST
EM16  4017 4725 6090 6570 6354 5895 5346 4911 4611 4422 4314
      4287 4224 4110 3915 3600 3240 2913 2598 2325 2088 1917
      1734 1587 1452 1356 1257         3.1620  0.7012  60.00 TEST
;
```

Fitting a PLS Model

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpplsgs1.sas.)

To isolate a few underlying spectral factors that provide a good predictive model, you can fit a PLS model to the 16 samples by using the following SAS statements:

```
proc hppls data=sample;
   model ls ha dt = v1-v27;
run;
```

By default, the HPPLS procedure extracts at most 15 factors. The default output from this analysis is presented in Figure 64.1 through Figure 64.3.

Figure 64.1 displays the "Performance Information," "Data Access Information," and "Model Information" tables.

The "Performance Information" table shows that PROC HPPLS executes in single-machine mode—that is, the model is fit on the machine where the SAS session executes. This run of the HPPLS procedure was performed on a multicore machine that has four CPUs; one computational thread was spawned per CPU.

The "Data Access Information" table shows that the input data set is accessed with the V9 (base) engine on the client machine where the MVA SAS session executes.

The "Model Information" table identifies the data source and shows that the factor extraction method is partial least squares regression (which is the default) and that the nonlinear iterative partial least squares (NIPALS) algorithm (which is also the default) is used to compute extracted PLS factors.

Figure 64.1 Performance, Data Access, and Model Information

The HPPLS Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.SAMPLE	V9	Input	On Client

Model Information	
Data Source	WORK.SAMPLE
Factor Extraction Method	Partial Least Squares
PLS Algorithm	NIPALS
Validation Method	None

Figure 64.2 displays the "Number of Observations" and "Dimensions" tables. The "Number of Observations" table shows that all 16 of the sample observations in the input data are used in the analysis because all samples contain complete data. The "Dimensions" table shows the number of dependent variables, the number of effects, the number of predictor parameters, and the number of factors to extract.

Figure 64.2 Number of Observations and Dimensions

Number of Observations Read	16
Number of Observations Used	16

Dimensions	
Number of Response Variables	3
Number of Effects	27
Number of Predictor Parameters	27
Number of Factors	15

Figure 64.3 lists the amount of variation, both individual and cumulative, that is accounted for by each of the 15 factors. All the variation in both the predictors and the responses is accounted for by only 15 factors because there are only 16 sample observations. More important, almost all the variation is accounted for with even fewer factors—one or two for the predictors and three to eight for the responses.

Figure 64.3 PLS Variation Summary

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		**Dependent Variables**	
Number of Extracted Factors	**Current**	**Total**	**Current**	**Total**
1	97.46068	97.46068	41.91546	41.91546
2	2.18296	99.64365	24.24355	66.15900
3	0.17806	99.82170	24.53393	90.69293
4	0.11973	99.94143	3.78978	94.48271
5	0.04146	99.98289	1.00454	95.48725
6	0.01058	99.99347	2.28084	97.76809
7	0.00168	99.99515	1.16935	98.93744
8	0.00097586	99.99613	0.50410	99.44153
9	0.00142	99.99755	0.12292	99.56446
10	0.00097037	99.99852	0.11027	99.67472
11	0.00032725	99.99884	0.15227	99.82699
12	0.00029338	99.99914	0.12907	99.95606
13	0.00024792	99.99939	0.03121	99.98727
14	0.00042742	99.99981	0.00651	99.99378
15	0.00018639	100.00000	0.00622	100.00000

Selecting the Number of Factors by Test Set Validation

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpplsgs1.sas`.)

A PLS model is not complete until you choose the number of factors. You can choose the number of factors by using test set validation, in which the data set is divided into two groups called the training data and test data. You fit the model to the training data, and then you check the capability of the model to predict responses for the test data. The predicted residual sum of squares (PRESS) statistic is based on the residuals that are generated by this process.

To select the number of extracted factors by test set validation, you use the PARTITION statement to specify how observations in the input data set are logically divided into two subsets for model training and testing. For example, you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, as in the following SAS statements:

```
proc hppls data=sample;
   model ls ha dt = v1-v27;
   partition roleVar = Role(train='TRAIN' test='TEST');
run;
```

The resulting output is shown in Figure 64.4 through Figure 64.6.

Figure 64.4 Model Information and Number of Observations with Test Set Validation

The HPPLS Procedure

Model Information	
Data Source	WORK.SAMPLE
Factor Extraction Method	Partial Least Squares
PLS Algorithm	NIPALS
Validation Method	Test Set Validation

Number of Observations Read	16
Number of Observations Used	16
Number of Observations Used for Training	9
Number of Observations Used for Testing	7

Figure 64.5 Test-Set-Validated PRESS Statistics for Number of Factors

The HPPLS Procedure

Test Set Validation for the Number of Extracted Factors	
Number of Extracted Factors	**Root Mean PRESS**
0	1.426362
1	1.276694
2	1.181752
3	0.656999
4	0.43457
5	0.420916
6	0.585031
7	0.576586
8	0.563935
9	0.563935

Minimum Root Mean PRESS	0.420916
Minimizing Number of Factors	5

Figure 64.6 PLS Variation Summary for Test-Set-Validated Model

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	95.92495	95.92495	37.27071	37.27071
2	3.86407	99.78903	32.38167	69.65238
3	0.10170	99.89073	20.76882	90.42120
4	0.08979	99.98052	4.66666	95.08787
5	0.01142	99.99194	3.88184	98.96971

In Figure 64.4, the "Model Information" table indicates that test set validation is used. The "Number of Observations" table shows that nine sample observations are assigned for training roles and seven are assigned for testing roles.

Figure 64.5 provides details about the results from test set validation. These results show that the absolute minimum PRESS is achieved with five extracted factors. Notice, however, that this is not much smaller than the PRESS for three factors. By using the CVTEST option, you can perform a statistical model comparison that is suggested by Van der Voet (1994) to test whether this difference is significant, as shown in the following SAS statements:

```
proc hppls data=sample cvtest(pval=0.15 seed=12345);
   model ls ha dt = v1-v27;
   partition roleVar = Role(train='TRAIN' test='TEST');
run;
```

The model comparison test is based on a rerandomization of the data. By default, the seed for this randomization is based on the system clock, but it is specified here. The resulting output is presented in Figure 64.7 through Figure 64.9.

Figure 64.7 Model Information with Model Comparison Test

The HPPLS Procedure

Model Information	
Data Source	WORK.SAMPLE
Factor Extraction Method	Partial Least Squares
PLS Algorithm	NIPALS
Validation Method	Test Set Validation
Validation Testing Criterion	Prob T**2 > 0.15
Number of Random Permutations	1000
Random Number Seed for Permutation	12345

Figure 64.8 Testing Test Set Validation for Number of Factors

The HPPLS Procedure

Test Set Validation for the Number of Extracted Factors

Number of Extracted Factors	Root Mean PRESS	T**2	Prob > T**2
0	1.426362	5.191629	0.0430
1	1.276694	6.174825	<.0001
2	1.181752	4.60203	0.0580
3	0.656999	3.09999	0.5020
4	0.43457	4.980227	0.0790
5	0.420916	0	1.0000
6	0.585031	2.05496	0.7700
7	0.576586	3.009172	0.4530
8	0.563935	2.416635	0.7420
9	0.563935	2.416635	0.7370

Minimum Root Mean PRESS	0.420916
Minimizing Number of Factors	5
Smallest Number of Factors with p > 0.15	3

Figure 64.9 PLS Variation Summary for Tested Test-Set-Validated Model

Percent Variation Accounted for by Partial Least Squares Factors

	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	95.92495	95.92495	37.27071	37.27071
2	3.86407	99.78903	32.38167	69.65238
3	0.10170	99.89073	20.76882	90.42120

The "Model Information" table in Figure 64.7 displays information about the options that are used in the model comparison test. In Figure 64.8, the p-value in comparing the test-set validated residuals from models that have five and three factors indicates that the difference between the two models is insignificant; therefore, the model with fewer factors is preferred. The variation summary in Figure 64.9 shows that more than 99% of the predictor variation and more than 90% of the response variation are accounted for by the three factors.

Predicting New Observations

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpplsgs1.sas`.)

Now that you have chosen a two-factor PLS model for predicting pollutant concentrations that are based on sample spectra, suppose that you have two new samples. The following SAS statements create a data set that contains the spectra for the new samples:

```
data newobs;
   input obsnam $ v1-v27 @@;
   datalines;
EM17  3933 4518 5637 6006 5721 5187 4641 4149 3789
      3579 3447 3381 3327 3234 3078 2832 2571 2274
      2040 1818 1629 1470 1350 1245 1134 1050  987
EM25  2904 2997 3255 3150 2922 2778 2700 2646 2571
      2487 2370 2250 2127 2052 1713 1419 1200  984
       795  648  525  426  351  291  240  204  162
;
```

You can apply the PLS model to these samples to estimate pollutant concentration by appending the new samples to the original 16 and specifying that the predicted values for all 18 be output to a data set, as shown in the following statements:

```
data all;
   set sample newobs;
run;

proc hppls data=all nfac=2;
   model ls ha dt = v1-v27;
   partition roleVar = Role(train='TRAIN' test='TEST');
   output out=result pred=p;
   id obsnam;
run;

proc print data=result;
   where (obsnam in ('EM17','EM25'));
   var obsnam p_ls p_ha p_dt;
run;
```

The ID statement lists the variable obsnam from the input data set that is transferred to the output data set. The new observations are not used in calculating the PLS model because they have no response values. Their predicted concentrations are shown in Figure 64.10.

Figure 64.10 Predicted Concentrations for New Observations

Obs	obsnam	p_ls	p_ha	p_dt
17	EM17	2.63326	0.22343	80.2027
18	EM25	0.69865	0.14308	98.9937

Syntax: HPPLS Procedure

The following statements are available in the HPPLS procedure:

```
PROC HPPLS <options> ;
   BY variables ;
   CLASS variable <(options)>... <variable <(options)>> </global-options> ;
   MODEL response-variables = predictor-effects </options> ;
   OUTPUT <OUT=SAS-data-set>
            <keyword <=prefix>>... <keyword <=prefix>>  ;
   PARTITION <partition-options> ;
   PERFORMANCE <performance-options> ;
   ID variables ;
```

The PROC HPPLS statement and a single MODEL statement are required. All other statements are optional. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statement. The following sections describe the PROC HPPLS statement and then describe the other statements in alphabetical order.

PROC HPPLS Statement

```
PROC HPPLS <options> ;
```

The PROC HPPLS statement invokes the HPPLS procedure. Table 64.1 summarizes the options available in the PROC HPPLS statement.

Table 64.1 PROC HPPLS Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
NAMELEN=	Limits the length of effect names
Model Fitting Options	
CVTEST	Requests that van der Voet's (1994) randomization-based model comparison test be performed
METHOD=	Specifies the general factor extraction method to be used
NFAC=	Specifies the number of factors to extract
NOCENTER	Suppresses centering of the responses and predictors before fitting
NOCVSTDIZE	Suppresses re-centering and rescaling of the responses and predictors when cross-validating
NOSCALE	Suppresses scaling of the responses and predictors before fitting
Output Options	
CENSCALE	Displays the centering and scaling information
DETAILS	Displays the details of the fitted model
NOCLPRINT	Limits or suppresses the display of class levels

Table 64.1 *continued*

Option	Description
NOPRINT	Suppresses ODS output
VARSS	Displays the amount of variation accounted for in each response and predictor

The following list provides details about these *options*.

CENSCALE
lists the centering and scaling information for each response and predictor.

CVTEST < (*cvtest-options*) >
requests that van der Voet's (1994) randomization-based model comparison test be performed to test models that have different numbers of extracted factors against the model that minimizes the predicted residual sum of squares. For more information, see the section "Test Set Validation" on page 5039. You can also specify the following *cvtest-options* in parentheses after the CVTEST option:

PVAL=*n*
specifies the cutoff probability for declaring an insignificant difference. By default, PVAL=0.10.

STAT=PRESS | T2
specifies the test statistic for the model comparison. You can specify either T2, for Hotelling's T^2 statistic, or PRESS, for the predicted residual sum of squares. By default, STAT=T2.

NSAMP=*n*
specifies the number of randomizations to perform. By default, NSAMP=1000.

SEED=*n*
specifies the seed value for the random number stream. If you do not specify a seed, or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer's clock.

Analyses that use the same (nonzero) seed are not completely reproducible if they are executed on a different number of threads because the random number streams in separate threads are independent. You can control the number of threads on which the HPPLS procedure executes by using SAS system options or by using the PERFORMANCE statement in the HPPLS procedure.

DATA=*SAS-data-set*
names the input SAS data set to be used by PROC HPPLS. The default is the most recently created data set.

DETAILS
lists the details of the fitted model for each successive factor. The listed details are different for different extraction methods. For more information, see the section "Displayed Output" on page 5042.

METHOD=PLS< (*PLS-options*) > | SIMPLS | PCR | RRR
specifies the general factor extraction method to be used. You can specify the following values:

PCR
requests principal components regression.

PLS< (*PLS-options*) >
requests partial least squares. You can also specify the following optional *PLS-options* in parentheses after METHOD=PLS:

ALGORITHM=NIPALS | SVD | EIG
names the specific algorithm used to compute extracted PLS factors. NIPALS requests the usual iterative NIPALS algorithm, SVD bases the extraction on the singular value decomposition of $\mathbf{X}'\mathbf{Y}$, and EIG bases the extraction on the eigenvalue decomposition of $\mathbf{Y}'\mathbf{X}\mathbf{X}'\mathbf{Y}$. ALGORITHM=SVD is the most accurate but least efficient approach. By default, ALGORITHM=NIPALS.

EPSILON=*n*
specifies the convergence criterion for the NIPALS algorithm. By default, EPSILON=10^{-12}.

MAXITER=*n*
specifies the maximum number of iterations for the NIPALS algorithm. By default, MAXITER=200.

RRR
requests reduced rank regression.

SIMPLS
requests the straightforward implementation of a statistically inspired modification of the partial least squares (SIMPLS) method of De Jong (1993).

By default, METHOD=PLS.

NAMELEN=*number*
specifies the length to which long effect names are shortened. By default, NAMELEN=20. If you specify a value less than 20 for *number*, the default is used.

NFAC=*n*
specifies the number of factors to extract. The default is $\min\{15, p, N\}$, where p is the number of predictors (or the number of dependent variables when METHOD=RRR) and N is the number of runs (observations). You probably do not need to extract this many factors for most applications. Extracting too many factors can lead to an overfit model (one that matches the training data too well), sacrificing predictive ability. Thus, if you use the default, you should also either specify the PARTITION statement to select the appropriate number of factors for the final model or consider the analysis to be preliminary and examine the results to determine the appropriate number of factors for a subsequent analysis.

NOCENTER
suppresses centering of the responses and predictors before fitting. This option is useful if the analysis variables are already centered and scaled. For more information, see the section "Centering and Scaling" on page 5040.

NOCLPRINT< =*number* >
suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed only for variables whose number of levels is less than *number*. Specifying a *number* helps to reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOCVSTDIZE
suppresses re-centering and rescaling of the responses and predictors before each model is fit in the cross validation. For more information, see the section "Centering and Scaling" on page 5040.

NOPRINT
suppresses the normal display of results. This option is useful when you want only the output statistics saved in a data set. This option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

NOSCALE
suppresses scaling of the responses and predictors before fitting. This option is useful if the analysis variables are already centered and scaled. For more information, see the section "Centering and Scaling" on page 5040.

VARSS
lists, in addition to the average response and predictor sum of squares accounted for by each successive factor, the amount of variation accounted for in each response and predictor.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPPLS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPPLS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) >. . . < *variable* < (*options*) > > < / *global-options* > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. The CLASS statement must precede the MODEL statement.

The CLASS statement for SAS high-performance statistical procedures is documented in the section "CLASS Statement" on page 572. The HPPLS procedure also supports the following *global-option* in the CLASS statement:

UPCASE

uppercases the values of character-valued CLASS variables before levelizing them. For example, if the UPCASE option is in effect and a CLASS variable can take the values 'a', 'A', and 'b', then 'a' and 'A' represent the same level and that CLASS variable is treated as having only two values: 'A' and 'B'.

ID Statement

ID *variables* ;

The ID statement lists one or more variables from the input data set to be transferred to output data sets that are created by SAS high-performance analytical procedures, provided that the output data set produces one (or more) records per input observation.

For information about the common ID statement in SAS high-performance analytical procedures, see the section "ID Statement" on page 576.

MODEL Statement

MODEL *response-variables* = *predictor-effects* < / *options* > ;

The MODEL statement names the responses and the predictors, which determine the **Y** and **X** matrices of the model, respectively. You can simply list the names of the predictor variables as the model effects, but you can also specify other types of effects, including polynomial effects and interactions. For information about constructing the model effects, see the section "Parameterization of Model Effects" on page 412.

The MODEL statement is required. You can specify only one MODEL statement.

You can specify the following *options* in the MODEL statement after a slash (/).

INTERCEPT
overrides the default, in which the responses and predictors are centered. When responses and predictors are centered, no intercept is required in the model.

SOLUTION
lists the coefficients of the final predictive model for the responses. The coefficients for predicting the centered and scaled responses that are based on the centered and scaled predictors are displayed, in addition to the coefficients for predicting the raw responses that are based on the raw predictors.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< *keyword* < =*prefix* > >. . . < *keyword* < =*prefix* > > **;**

The OUTPUT statement creates a data set that contains observationwise statistics, which are computed after fitting the model. If you do not specify any *keyword*, then only the predicted values for responses are included.

The variables in the input data set are *not* included in the output data set in order to avoid data duplication for large data sets; however, variables specified in the ID statement are included.

You can specify the following syntax elements in the OUTPUT statement:

OUT=*SAS-data-set*

DATA=*SAS-data-set*
specifies the name of the output data set. If the OUT= (or DATA=) option is omitted, the procedure uses the DATA*n* convention to name the output data set.

keyword **< =***prefix* **>**
specifies a statistic to include in the output data set and optionally a *prefix* for naming the output variables. If you do not provide a *prefix*, the HPPLS procedure assigns a default prefix based on the type of statistic requested. For example, for response variables y1 and y2, a specification of PREDICTED produces two predicted value variables Pred_y1 and Pred_y2.

You can specify the following *keywords* for adding statistics to the OUTPUT data set:

H
requests the approximate leverage. The default prefix is H.

PREDICTED

PRED

P
requests predicted values for each response. The default prefix is Pred.

PRESS
requests approximate predicted residuals for each response. The default prefix is PRESS.

ROLE

requests numeric values that indicate the role played by each observation in fitting the model. The default prefix is _ROLE_. Table 64.2 shows the interpretation of this variable for each observation.

Table 64.2 Role Interpretation

Value	Observation Role
0	Not used
1	Training
2	Testing

If you do not partition the input data by using a PARTITION statement, then the role variable value is 1 for observations that are used in fitting the model, and 0 for observations that have at least one missing or invalid value for the responses or predictors.

STDX

requests standardized (centered and scaled) predictor values for each predictor. The default prefix is StdX.

STDXSSE

requests the sum of squares of residuals for standardized predictors. The default prefix is StdXSSE.

STDY

requests standardized (centered and scaled) response values for each response. The default prefix is StdY.

STDYSSE

requests the sum of squares of residuals for standardized responses. The default prefix is StdYSSE.

TSQUARE

T2

requests scaled sum of squares of score values. The default prefix is TSquare.

XRESIDUAL

XRESID

XR

requests residuals for each predictor. The default prefix is XResid.

XSCORE

requests extracted factors (X-scores, latent vectors, latent variables, T) for each selected model factor. The default prefix is XScore.

YRESIDUAL
YRESID
YR
requests residuals for each response. The default prefix is YResid.

YSCORE
requests extracted responses (Y-scores, U) for each selected model factor. The default prefix is YScore.

According to the *keyword* specified, the output variables that contain the requested statistic are named as follows:

- The *keywords* XRESIDUAL and STDX define an output variable for each predictor, so the variables that correspond to each predictor are named by appending a number (which starts from 1) to the prefix. For each defined variable, a label is also generated automatically; the label contains the prefix of the variable and the name of the predictor. For example, if the model has three predictors, then a specification of XRESIDUAL=XR produces the variables XR1, XR2, and XR3.
- The *keywords* PREDICTED, YRESIDUAL, STDY, and PRESS define an output variable for each response, so the variables that correspond to each response are named by appending the name of the response variable to the prefix. For example, if the model has response variables y1 and y2, then a specification of PREDICTED=P produces the variables P_y1 and P_y2.
- The *keywords* XSCORE and YSCORE define an output variable for each selected model factor, so the variables that correspond to each successive factor are named by appending the factor number to the prefix. For example, if the model has three selected factors, then a specification of XSCORE=T produces the variables T1, T2, and T3.
- The *keywords* H, TSQUARE, STDXSSE, STDYSSE, and ROLE each define a single output variable, so the variable name matches the prefix.

PARTITION Statement

PARTITION < *partition-options* > **;**

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training and testing. Either you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations for each role.

You can specify one (but not both) of the following *partition-options*:

ROLEVAR | ROLE=*variable* **(< TEST='***value***' > < TRAIN='***value***' >)**
names the variable in the input data set whose values are used to assign roles to each observation. The formatted values of this variable that are used to assign observations roles are specified in the TEST= and TRAIN= suboptions. If you specify only the TEST= suboption, then all observations whose role is not determined by the TEST= suboption are assigned to training. If you specify only the TRAIN= suboption, then all observations whose role is not determined by the TRAIN= suboption are assigned to testing.

FRACTION(<TEST=*fraction*> <SEED=*n*>)

requests that specified proportions of the observations in the input data set be randomly assigned training and testing roles. You specify the proportions for testing by using the TEST= suboption; the specified fraction must be less than 1 and the remaining fraction of the observations are assigned to the training role. If you do not specify the TEST= suboption, the default fraction is 0.5. The SEED= suboption specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training and testing. If you do not specify a seed, or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer's clock.

Because *fraction* is a per-observation probability (which means that any particular observation has a probability of *fraction* of being assigned the testing role), using the FRACTION option can cause different numbers of observations to be assigned training and testing roles. Different partitions can be observed when the number of threads changes.

PERFORMANCE Statement

PERFORMANCE <*performance-options*> **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of the HPPLS procedure.

The PERFORMANCE statement is documented further in the section "PERFORMANCE Statement" on page 569.

Details: HPPLS Procedure

Regression Methods

All the predictive methods that PROC HPPLS implements work essentially by finding linear combinations of the predictors (factors) to use to predict the responses linearly. The methods differ only in how the factors are derived, as explained in the following sections.

Partial Least Squares

Partial least squares (PLS) works by extracting one factor at a time. Let $\mathbf{X} = \mathbf{X}_0$ be the centered and scaled matrix of predictors, and let $\mathbf{Y} = \mathbf{Y}_0$ be the centered and scaled matrix of response values. The PLS method starts with a linear combination $\mathbf{t} = \mathbf{X}_0\mathbf{w}$ of the predictors, where $\mathbf{t}$ is called a *score vector* and $\mathbf{w}$ is its associated *weight vector*. The PLS method predicts both $\mathbf{X}_0$ and $\mathbf{Y}_0$ by regression on $\mathbf{t}$:

$$\begin{aligned} \hat{\mathbf{X}}_0 &= \mathbf{t}\mathbf{p}', \quad \text{where} \quad \mathbf{p}' = (\mathbf{t}'\mathbf{t})^{-1}\mathbf{t}'\mathbf{X}_0 \\ \hat{\mathbf{Y}}_0 &= \mathbf{t}\mathbf{c}', \quad \text{where} \quad \mathbf{c}' = (\mathbf{t}'\mathbf{t})^{-1}\mathbf{t}'\mathbf{Y}_0 \end{aligned}$$

The vectors $\mathbf{p}$ and $\mathbf{c}$ are called the X- and Y-*loadings*, respectively.

The specific linear combination $\mathbf{t} = \mathbf{X}_0\mathbf{w}$ is the one that has maximum covariance $\mathbf{t}'\mathbf{u}$ with some response linear combination $\mathbf{u} = \mathbf{Y}_0\mathbf{q}$. Another characterization is that the X-weight, $\mathbf{w}$, and the Y-weight, $\mathbf{q}$, are

proportional to the first left and right singular vectors, respectively, of the covariance matrix $\mathbf{X}_0'\mathbf{Y}_0$ or, equivalently, the first eigenvectors of $\mathbf{X}_0'\mathbf{Y}_0\mathbf{Y}_0'\mathbf{X}_0$ and $\mathbf{Y}_0'\mathbf{X}_0\mathbf{X}_0'\mathbf{Y}_0$, respectively.

This accounts for how the first PLS factor is extracted. The second factor is extracted in the same way by replacing $\mathbf{X}_0$ and $\mathbf{Y}_0$ with the X- and Y-residuals from the first factor:

$$\begin{aligned}\mathbf{X}_1 &= \mathbf{X}_0 - \hat{\mathbf{X}}_0 \\ \mathbf{Y}_1 &= \mathbf{Y}_0 - \hat{\mathbf{Y}}_0\end{aligned}$$

These residuals are also called the *deflated* $\mathbf{X}$ and $\mathbf{Y}$ blocks. The process of extracting a score vector and deflating the data matrices is repeated for as many extracted factors as are wanted.

SIMPLS

Note that each extracted PLS factor is defined in terms of different X-variables $\mathbf{X}_i$. This leads to difficulties in comparing different scores, weights, and so on. The SIMPLS method of De Jong (1993) overcomes these difficulties by computing each score $\mathbf{t}_i = \mathbf{X}\mathbf{r}_i$ in terms of the original (centered and scaled) predictors $\mathbf{X}$. The SIMPLS X-weight vectors r_i are similar to the eigenvectors of $\mathbf{S}\mathbf{S}' = \mathbf{X}'\mathbf{Y}\mathbf{Y}'\mathbf{X}$, but they satisfy a different orthogonality condition. The $\mathbf{r}_1$ vector is just the first eigenvector $\mathbf{e}_1$ (so that the first SIMPLS score is the same as the first PLS score). However, the second eigenvector maximizes

$$\mathbf{e}_1'\mathbf{S}\mathbf{S}'\mathbf{e}_2 \text{ subject to } \mathbf{e}_1'\mathbf{e}_2 = 0$$

whereas the second SIMPLS weight $\mathbf{r}_2$ maximizes

$$\mathbf{r}_1' S S' \mathbf{r}_2 \text{ subject to } \mathbf{r}_1'\mathbf{X}'\mathbf{X}\mathbf{r}_2 = \mathbf{t}_1'\mathbf{t}_2 = 0$$

The SIMPLS scores are identical to the PLS scores for one response but slightly different for more than one response; see De Jong (1993) for details. The X- and Y-loadings are defined as in PLS, but because the scores are all defined in terms of $\mathbf{X}$, it is easy to compute the overall model coefficients $\mathbf{B}$:

$$\begin{aligned}\hat{\mathbf{Y}} &= \sum_i \mathbf{t}_i \mathbf{c}_i' \\ &= \sum_i \mathbf{X}\mathbf{r}_i \mathbf{c}_i' \\ &= \mathbf{X}\mathbf{B}, \text{ where } \mathbf{B} = \mathbf{R}\mathbf{C}'\end{aligned}$$

Principal Components Regression

Like the SIMPLS method, principal component regression (PCR) defines all the scores in terms of the original (centered and scaled) predictors $\mathbf{X}$. However, unlike both the PLS and SIMPLS methods, the PCR method chooses the X-weights and X-scores without regard to the response data. The X-scores are chosen to explain as much variation in $\mathbf{X}$ as possible; equivalently, the X-weights for the PCR method are the eigenvectors of the predictor covariance matrix $\mathbf{X}'\mathbf{X}$. Again, the X- and Y-loadings are defined as in PLS; but, as in SIMPLS, it is easy to compute overall model coefficients for the original (centered and scaled) responses $\mathbf{Y}$ in terms of the original predictors $\mathbf{X}$.

Reduced Rank Regression

As discussed in the preceding sections, partial least squares depends on selecting factors $\mathbf{t} = \mathbf{X}\mathbf{w}$ of the predictors and $\mathbf{u} = \mathbf{Y}\mathbf{q}$ of the responses that have maximum covariance, whereas principal components regression effectively ignores $\mathbf{u}$ and selects $\mathbf{t}$ to have maximum variance, subject to orthogonality constraints. In contrast, reduced rank regression selects $\mathbf{u}$ to account for as much variation in the *predicted* responses as possible, effectively ignoring the predictors for the purposes of factor extraction. In reduced rank regression, the Y-weights $\mathbf{q}_i$ are the eigenvectors of the covariance matrix $\hat{\mathbf{Y}}'_{\mathrm{LS}}\hat{\mathbf{Y}}_{\mathrm{LS}}$ of the responses that are predicted by ordinary least squares regression, and the X-scores are the projections of the Y-scores $\mathbf{Y}\mathbf{q}_i$ onto the X space.

Relationships between Methods

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpplsgs2.sas.)

When you develop a predictive model, it is important to consider not only the explanatory power of the model for current responses, but also how well sampled the predictive functions are, because the sampling affects how well the model can extrapolate to future observations. All the techniques that the HPPLS procedure implements work by extracting successive factors (linear combinations of the predictors) that optimally address one or both of these two goals: explaining response variation and explaining predictor variation. In particular, principal components regression selects factors that explain as much predictor variation as possible, reduced rank regression selects factors that explain as much response variation as possible, and partial least squares balances the two objectives, seeking factors that explain both response and predictor variation.

To see the relationships between these methods, consider how each one extracts a single factor from the following artificial data set, which consists of two predictors and one response:

```
data data;
   input x1 x2 y;
   datalines;
    3.37651  2.30716        0.75615
    0.74193 -0.88845        1.15285
    4.18747  2.17373        1.42392
    0.96097  0.57301        0.27433
   -1.11161 -0.75225       -0.25410
   -1.38029 -1.31343       -0.04728
    1.28153 -0.13751        1.00341
   -1.39242 -2.03615        0.45518
    0.63741  0.06183        0.40699
   -2.52533 -1.23726       -0.91080
    2.44277  3.61077       -0.82590
```

```
;

proc hppls data=data nfac=1 method=rrr;
   model y = x1 x2;
run;

proc hppls data=data nfac=1 method=pcr;
   model y = x1 x2;
run;

proc hppls data=data nfac=1 method=pls;
   model y = x1 x2;
run;
```

The amount of model and response variation that are explained by the first factor for each method is shown in Figure 64.11 through Figure 64.13.

Figure 64.11 Variation Explained by the First Reduced Rank Regression Factor

The HPPLS Procedure

Percent Variation Accounted for by Reduced Rank Regression Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	15.06605	15.06605	100.00000	100.00000

Figure 64.12 Variation Explained by the First Principal Components Regression Factor

The HPPLS Procedure

Percent Variation Accounted for by Principal Components				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	92.99959	92.99959	9.37874	9.37874

Figure 64.13 Variation Explained by the First Partial Least Squares Regression Factor

The HPPLS Procedure

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	88.53567	88.53567	26.53038	26.53038

Notice that although the first reduced rank regression factor explains *all* of the response variation, it accounts for only about 15% of the predictor variation. In contrast, the first principal component regression factor accounts for most of the predictor variation (93%) but only 9% of the response variation. The first partial least squares factor accounts for only slightly less predictor variation than principal components but about three times as much response variation.

Figure 64.14 illustrates how partial least squares balances the goals of explaining response and predictor variation in this case.

Figure 64.14 Depiction of the First Factors for Three Different Regression Methods

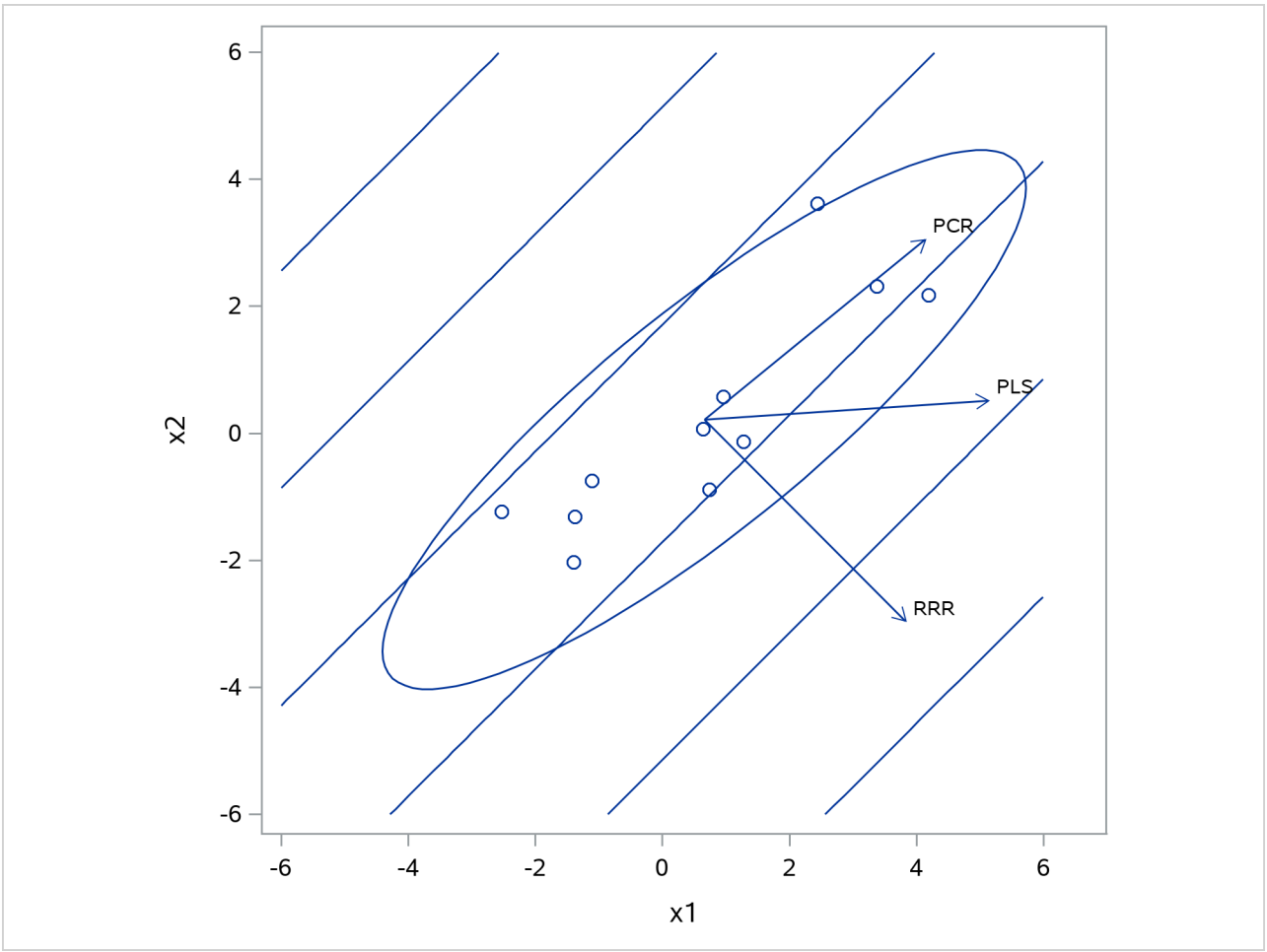

The ellipse shows the general shape of the 11 observations in the predictor space, with the contours of increasing y overlaid. Also shown are the directions of the first factor for each of the three methods. Notice that although the predictors vary most in the x1 = x2 direction, the response changes most in the orthogonal x1 = –x2 direction. This explains why the first principal component accounts for little variation in the response and why the first reduced rank regression factor accounts for little variation in the predictors. The direction of the first partial least squares factor represents a compromise between the other two directions.

Test Set Validation

None of the regression methods that the HPPLS procedure implements fit the observed data any better than ordinary least squares (OLS) regression; in fact, all the methods approach OLS as more factors are extracted. Basing the model on more extracted factors improves the model fit to the observed data, but extracting too many factors can cause *overfitting*—that is, tailoring the model too much to the current data to the detriment of future predictions. So the crucial point is that when there are many predictors, OLS can *overfit* the observed data; biased regression methods that use fewer extracted factors can provide better predictability of *future* observations. However, as the preceding observations imply, the quality of the observed data fit cannot be used to choose the number of factors to extract; the number of extracted factors must be chosen on the basis of how well the model fits observations that are not involved in the modeling procedure itself.

The method of choosing the number of extracted factors that PROC HPPLS implements is called *test set validation*. When you have sufficient data, you can subdivide your data into two parts: training data and test data. During the validation process, the model is fit on the training data, and the predicted residual sum of squares (PRESS) for models that have different numbers of extracted factors is found by using the test data. The number of factors chosen is usually the one that minimizes PRESS.

You use a PARTITION statement to logically subdivide the DATA= data set into separate roles. You can name the fractions of the data that you want to reserve as training data and test data. For example, the following statements randomly subdivide the inData data set, reserving 50% each for training and testing:

```
proc hppls data=inData;
   partition fraction(test=0.5);
   ...
run;
```

In some cases you might need to exercise more control over the partitioning of the input data set. You can do this by naming both a variable in the input data set and a formatted value of that variable for each role. For example, the following statements assign roles to the observations in the inData data set based on the value of the variable Group in that data set. Observations whose value of Group is `'group 1'` are assigned for training, and those whose value is `'group 2'` are assigned to testing. All other observations are ignored.

```
proc hppls data=inData;
   partition roleVar=Group(train='group 1' test='group 2')
   ...
run;
```

By default, the number of extracted factors is chosen to be the one that minimizes PRESS. However, models that have fewer factors often have PRESS statistics that are only marginally larger than the absolute minimum. To address this, Van der Voet (1994) proposed a statistical test for comparing the predicted residuals from different models; when you apply van der Voet's test, the number of factors chosen is the fewest while still producing residuals that are insignificantly larger than the residuals of the model that has a minimum PRESS.

To see how van der Voet's test works, let $R_{i,jk}$ be the jth predicted residual for response k for the model that has i extracted factors. Then, the PRESS statistic is $\sum_{jk} R^2_{i,jk}$. Also, let $i_{\min}$ be the number of factors for which PRESS is minimized. The critical value for van der Voet's test is based on the differences between squared predicted residuals:

$$D_{i,jk} = R^2_{i,jk} - R^2_{i_{\min},jk}$$

One alternative for the critical value is $C_i = \sum_{jk} D_{i,jk}$, which is simply the difference between the PRESS statistics for i and $i_{\min}$ factors; alternatively, van der Voet suggests Hotelling's T^2 statistic $C_i = \mathbf{d}'_{i,.}\mathbf{S}_i^{-1}\mathbf{d}_{i,.}$, where $\mathbf{d}_{i,.}$ is the sum of the vectors $\mathbf{d}_{i,j} = \{D_{i,j1}, \ldots, D_{i,jN_y}\}'$ and $\mathbf{S}_i$ is the sum of squares and crossproducts matrix,

$$\mathbf{S}_i = \sum_j \mathbf{d}_{i,j}\mathbf{d}'_{i,j}$$

Virtually, the significance level for van der Voet's test is obtained by comparing C_i with the distribution of values that result from randomly exchanging $R^2_{i,jk}$ and $R^2_{i_{\min},jk}$. In practice, a Monte Carlo sample of such values is simulated and the significance level is approximated as the proportion of simulated critical values that are greater than C_i. If you apply van der Voet's test by specifying the CVTEST option, then, by default, the number of extracted factors that are chosen is the least number of factors that have an approximate significance level that is greater than 0.10.

Centering and Scaling

By default, the predictors and the responses are centered and scaled to have mean 0 and standard deviation 1. Centering the predictors and the responses ensures that the criterion for choosing successive factors is based on how much *variation* they explain in either the predictors or the responses or in both. (For more information about how different methods explain variation, see the section "Regression Methods" on page 5034.) Without centering, both the mean variable value and the variation around that mean are involved in selecting factors. Scaling serves to place all predictors and responses on an equal footing relative to their variation in the data. For example, if Time and Temp are two of the predictors, then scaling says that a change of std(Time) in Time is approximately equivalent to a change of std(Temp) in Temp.

Usually, both the predictors and responses should be centered and scaled. However, if their values already represent variation around a nominal or target value, then you can use the NOCENTER option in the PROC HPPLS statement to suppress centering. Likewise, if the predictors or responses are already all on comparable scales, then you can use the NOSCALE option to suppress scaling.

If the predictors involve crossproduct terms, PROC HPPLS does not standardize the variables before it standardizes the crossproduct. That is, if the ith values of two predictors are denoted x_i^1 and x_i^2, then the default standardized ith value of the crossproduct is

$$\frac{x_i^1 x_i^2 - \mathrm{mean}_j(x_j^1 x_j^2)}{\mathrm{std}_j(x_j^1 x_j^2)}$$

When test set validation is performed for the number of effects, some practitioners disagree as to whether the training data should be retransformed. By default, PROC HPPLS does retransform the training data, but you can suppress this behavior by specifying the NOCVSTDIZE option in the PROC HPPLS statement.

Missing Values

Observations that have any missing independent variables (including all classification variables) are excluded from the analysis, and no predictions are computed for such observations. Observations that have no missing independent variables but do have missing dependent variables are also excluded from the analysis, but predictions are computed. If you use the PARTITION statement and specify the ROLEVAR= option, observations that contain the missing ROLEVAR= variable are excluded from the analysis, but predictions are computed for them.

Computational Method

Multithreading

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to sequential (single-threaded) execution.

The number of threads that the HPPLS procedure spawns is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the CPU count by using the CPUCOUNT= SAS system option. For example, if you specify the following statements, the HPPLS procedure schedules threads as if it executed on a system that has four CPUs, regardless of the actual CPU count.

```
options cpucount=4;
```

- You can specify the NTHREADS= option in the PERFORMANCE statement to determine the number of threads. This specification overrides the system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads per machine is displayed in the “Performance Information” table, which is part of the default output. The HPPLS procedure allocates one thread per CPU.

The tasks that the HPPLS procedure multithreads are primarily defined by dividing the data that are processed on a single machine among the threads—that is, the HPPLS procedure implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and PROC HPPLS runs on four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following:

- variable levelization
- effect levelization
- formation of the crossproducts matrix
- computation of loadings, weights, scores, generalized inverse, and residual sums of squares

- scoring of observations

In addition, operations on matrices such as sweeps might be multithreaded if the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Output Data Set

When an observationwise output data set is created, many procedures in SAS software add the variables from the input data set to the output data set. High-performance statistical procedures assume that the input data sets can be large and contain many variables. For performance reasons, the output data set contains only the following:

- variables that are explicitly created by the statement
- variables that are listed in the ID statement

Displayed Output

The following sections describe the output that PROC HPPLS produces. The output is organized into various tables, which are discussed in the order of their appearance.

Performance Information

The "Performance Information" table is produced by default. It displays the number of threads used for single-machine mode.

Data Access Information

The "Data Access Information" table is produced by default. For the input and output data sets, it displays the libref and data set name, the engine used to access the data, the role (input or output) of the data set, and the path that data followed to reach the computation.

Centering and Scaling Information

If you specify the CENSCALE option in the PROC HPPLS statement, the HPPLS procedure produces "Model Effect Centering and Scaling" and "Dependent Variable Centering and Scaling" tables, which display the centering and scaling information for each response and predictor.

Model Information

The "Model Information" table displays basic information about the model, such as the input data set, the factor extraction method, the validation method, and the type of parameterization used for classification variables that are named in the CLASS statement. If you use the PARTITION statement, the table also displays the random number seed for partition, the validation testing criterion, the number of random permutations, and the random number seed for permutation, depending on whether you specify the FRACTION option in the PARTITION statement and the CVTEST option in the PROC HPPLS statement.

Number of Observations

The "Number of Observations" table displays the number of observations that are read from the input data set and the number of observations that are used in the analysis. If you use a PARTITION statement, the table also displays the number of observations that are used for each data role.

Class Level Information

The "Class Level Information" table lists the levels of every variable that is specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress the "Class Level Information" table completely or partially by specifying the NOCLPRINT= option in the PROC HPPLS statement.

If the classification variables are in the reference parameterization, the "Class Level Information" table also displays the reference value for each variable.

Dimensions

The "Dimensions" table displays information about the number of response variables, the number of effects, and the number of predictor parameters. It also displays the number of factors to extract.

Test Set Validation

If you use the PARTITION statement to perform a test set validation for choosing the number of extracted factors, the HPPLS procedure produces a "Test Set Validation Residual Summary" table to display a residual summary of the validation for each number of factors. It also produces a "Test Set Validation Results" table to display information about the optimal number of factors.

Percent Variation Accounted for by Extracted Factors

By default, the HPPLS procedure produces the "Percent Variation Accounted for by Extracted Factors" table to display just the amount of predictor variation and response variation that are accounted for by each factor. If you specify the VARSS option in the PROC HPPLS statement, the HPPLS procedure also produces the "Model Effect Percent Variation Accounted for by Extracted Factors" table and the "Dependent Variable Percent Variation Accounted for by Extracted Factors" table to display the amount of variation that is accounted for in each response and predictor, in addition to the average response and predictor sum of squares that are accounted for by each successive factor.

Model Details

If you specify the DETAILS option in the PROC HPPLS statement, the HPPLS procedure produces tables to display details about the fitted model for each successive factor. These tables include the following:

- "Model Effect Loadings" table, which displays the predictor loadings
- "Model Effect Weights" table, which displays predictor weights
- "Dependent Variable Weights" table, which displays the response weights

- "Coded Regression Coefficients" tables, which display the coded regression coefficients, if you specify METHOD=SIMPLS, METHOD=PCR, or METHOD=RRR in the PROC HPPLS statement.

Parameter Estimates

If you specify the SOLUTION option in the MODEL statement, the HPPLS procedure produces a "Parameter Estimates" table to display the coefficients of the final predictive model for the responses. The coefficients for predicting the centered and scaled responses based on the centered and scaled predictors are displayed, in addition to the coefficients for predicting the raw responses based on the raw predictors.

Timing Information

If you specify the DETAILS option in the PERFORMANCE statement, the HPPLS procedure produces a "Timing" table, which displays the elapsed time (absolute and relative) of each main task of the procedure.

ODS Table Names

PROC HPPLS assigns a name to each table that it creates. You can use these names to refer to the ODS table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 64.3. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 64.3 ODS Tables Produced by PROC HPPLS

Table Name	Description	Required Statement and Option
CVResults	Results of test set validation	PARTITION statement
CenScaleParms	Parameter estimates for centered and scaled data	SOLUTION option in MODEL statement
ClassLevels	Level information from the CLASS statement	CLASS statement
CodedCoef	Coded regression coefficients	DETAILS option in PROC HPPLS statement
DataAccessInfo	Information about modes of data access	Default output
Dimensions	Model dimensions	Default output
ModelInfo	Model information	Default output
NObs	Number of observations read and used	Default output
ParameterEstimates	Parameter estimates for raw data	SOLUTION option in MODEL statement
PercentVariation	Predictor and response variation that are accounted for by each factor	Default output
PerformanceInfo	Information about the high-performance computing environment	Default output
ResidualSummary	Residual summary from test set validation	PARTITION statement

Table 64.3 *continued*

Table Name	Description	Required Statement and Option
Timing	Absolute and relative times of tasks that are performed by the procedure	DETAILS option in PERFORMANCE statement
XEffectCenScale	Centering and scaling information for predictor effects	CENSCALE option in PROC HPPLS statement
XLoadings	Loadings for predictor effects	DETAILS option in PROC HPPLS statement
XPercentVariation	Variation that is accounted for by each factor for predictor effects	VARSS option in PROC HPPLS statement
XWeights	Weights for predictor effects	DETAILS option in PROC HPPLS statement
YPercentVariation	Variation that is accounted for by each factor for responses	VARSS option in PROC HPPLS statement
YVariableCenScale	Centering and scaling information for responses	CENSCALE option in PROC HPPLS statement
YWeights	Weights for responses	DETAILS option in PROC HPPLS statement

Example: HPPLS Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 64.1: Choosing a PLS Model by Test Set Validation

(View the complete code for this example (hpplsex1.sas) in the example repository.)

This example demonstrates issues in spectrometric calibration. The data (Umetrics 1995) consist of spectrographic readings on 33 samples that contain known concentrations of two amino acids, tyrosine and tryptophan. The spectra are measured at 30 frequencies across the overall range of frequencies. For example, Output 64.1.1 shows the observed spectra for three samples: one with only tryptophan, one with only tyrosine, and one with a mixture of the two, all at a total concentration of 10^{-6}.

Output 64.1.1 Spectra for Three Samples of Tyrosine and Tryptophan

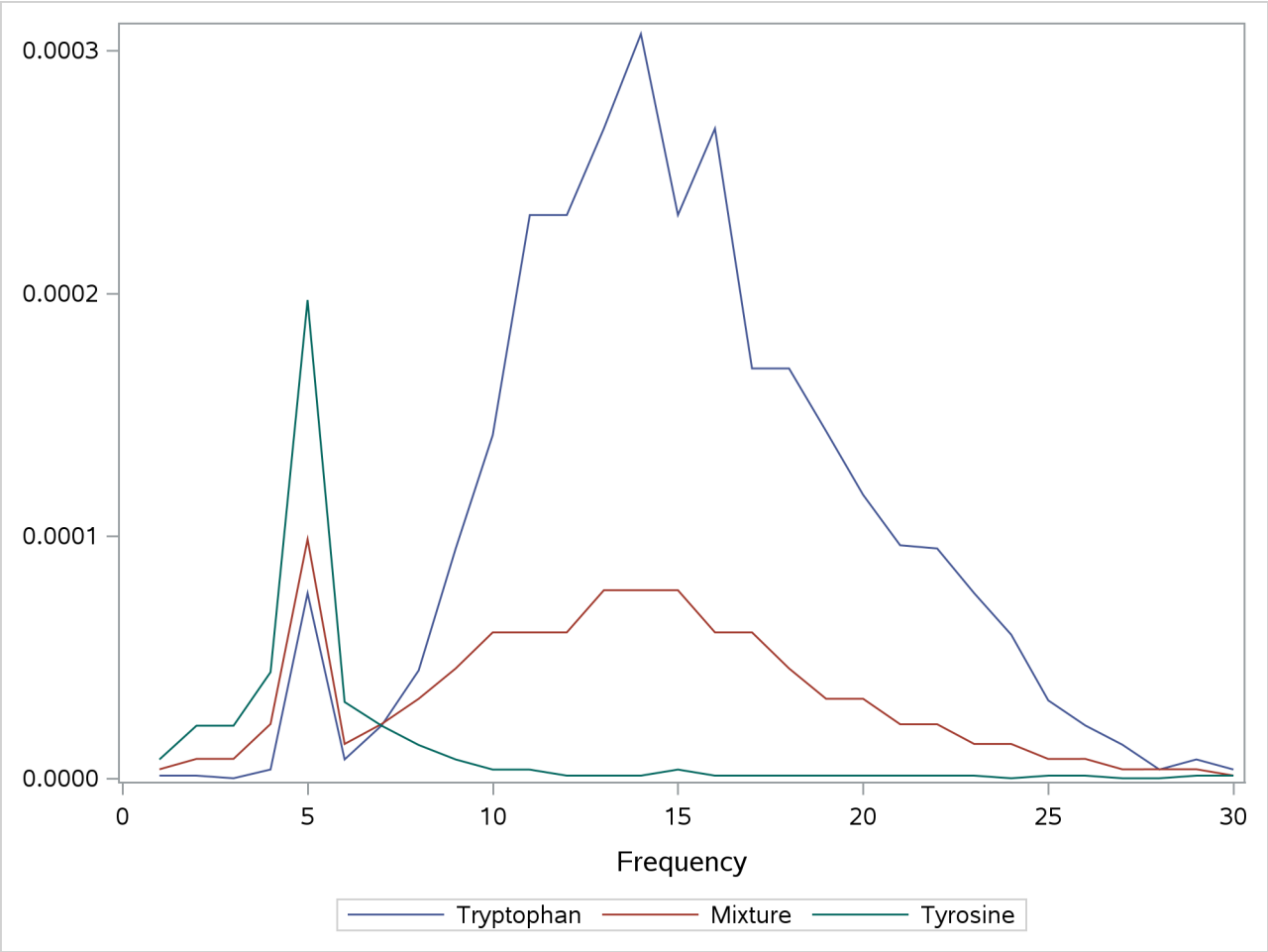

Of the 33 samples, 18 are used as a training set and 15 as a test set. The data originally appear in McAvoy et al. (1989).

These data were created in a lab, where the concentrations are fixed in order to provide a wide range of applicability for the model. You want to use a linear function of the logarithms of the spectra to predict the logarithms of tyrosine and tryptophan concentration, in addition to the logarithm of the total concentration. Actually, because zeros are possible in both the responses and the predictors, slightly different transformations are used. The following statements create a SAS data set named ex1Data for these data. The data set also contains a variable Role that is used to assign samples to the training and testing roles.

```
data ex1Data;
   input obsnam $ Role : $5. tot tyr f1-f30 @@;
   try = tot - tyr;
   if (tyr) then tyr_log = log10(tyr); else tyr_log = -8;
   if (try) then try_log = log10(try); else try_log = -8;
   tot_log = log10(tot);
   datalines;
17mix35 TRAIN 0.00003 0
 -6.215 -5.809 -5.114 -3.963 -2.897 -2.269 -1.675 -1.235
 -0.900 -0.659 -0.497 -0.395 -0.335 -0.315 -0.333 -0.377
```

```
 -0.453 -0.549 -0.658 -0.797 -0.878 -0.954 -1.060 -1.266
 -1.520 -1.804 -2.044 -2.269 -2.496 -2.714
19mix35 TRAIN 0.00003 3E-7
 -5.516 -5.294 -4.823 -3.858 -2.827 -2.249 -1.683 -1.218
 -0.907 -0.658 -0.501 -0.400 -0.345 -0.323 -0.342 -0.387
 -0.461 -0.554 -0.665 -0.803 -0.887 -0.960 -1.072 -1.272
 -1.541 -1.814 -2.058 -2.289 -2.496 -2.712
21mix35 TRAIN 0.00003 7.5E-7
 -5.519 -5.294 -4.501 -3.863 -2.827 -2.280 -1.716 -1.262
 -0.939 -0.694 -0.536 -0.444 -0.384 -0.369 -0.377 -0.421
 -0.495 -0.596 -0.706 -0.824 -0.917 -0.988 -1.103 -1.294
 -1.565 -1.841 -2.084 -2.320 -2.521 -2.729
23mix35 TRAIN 0.00003 1.5E-6

   ... more lines ...

26tyro5 TEST 0.00001 0.00001
 -3.037 -2.696 -2.464 -2.321 -2.239 -2.444 -2.602 -2.823
 -3.144 -3.396 -3.742 -4.063 -4.398 -4.699 -4.893 -5.138
 -5.140 -5.461 -5.463 -5.945 -5.461 -5.138 -5.140 -5.138
 -5.138 -5.463 -5.461 -5.461 -5.461 -5.461
tyro2   TEST 0.0001 0.0001
 -1.081 -0.710 -0.470 -0.337 -0.327 -0.433 -0.602 -0.841
 -1.119 -1.423 -1.750 -2.121 -2.449 -2.818 -3.110 -3.467
 -3.781 -4.029 -4.241 -4.366 -4.501 -4.366 -4.501 -4.501
 -4.668 -4.668 -4.865 -4.865 -5.109 -5.111
;
```

The following statements fit a PLS model that has 10 factors.

```
proc hppls data=ex1Data nfac=10;
   model tot_log tyr_log try_log = f1-f30;
run;
```

The "Model Information" table in Output 64.1.2 shows that no validation method is used. The "Number of Observations" table confirms that all 33 sample observations are used in the analysis.

The table in Output 64.1.3 indicates that only four or five factors are required to explain almost all of the variation in both the predictors and the responses.

Output 64.1.2 Model Information and Number of Observations

The HPPLS Procedure

Model Information	
Data Source	WORK.EX1DATA
Factor Extraction Method	Partial Least Squares
PLS Algorithm	NIPALS
Validation Method	None

Number of Observations Read	33
Number of Observations Used	33

Output 64.1.3 Amount of Variation Explained

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	77.67903	77.67903	47.80217	47.80217
2	20.62719	98.30622	38.96826	86.77043
3	1.00143	99.30766	6.89262	93.66305
4	0.24930	99.55696	1.77222	95.43528
5	0.13077	99.68773	1.71762	97.15290
6	0.08970	99.77742	0.58619	97.73909
7	0.05684	99.83426	0.29079	98.02988
8	0.06730	99.90156	0.13857	98.16845
9	0.01521	99.91676	0.68214	98.85059
10	0.02627	99.94304	0.14388	98.99447

In order to choose the optimal number of PLS factors, you can explore how well models that are based on data in training roles and have different numbers of factors fit the data in testing roles. To do so, you can use the PARTITION statement to assign observations to training and testing roles based on the values of the input variable named Role.

```
proc hppls data=ex1Data nfac=10 cvtest(stat=press seed=12345);
   model tot_log tyr_log try_log = f1-f30;
   partition roleVar = Role(train='TRAIN' test='TEST');
run;
```

Output 64.1.4 shows the "Model Information" table and the "Number of Observations" table. The "Model Information" table indicates that test set validation is used and displays information about the options that are used in the model comparison test. The "Number of Observations" table confirms that there are 18 observations for the training role and 15 for the testing role.

Output 64.1.5 displays the results of the test set validation. They indicate that although five PLS factors produce the minimum predicted residual sum of squares, the residuals for four factors are insignificantly different from the residuals for five factors. Thus, the smaller model is preferred.

Output 64.1.4 Model Information and Number of Observations with Test Set Validation

The HPPLS Procedure

Model Information	
Data Source	WORK.EX1DATA
Factor Extraction Method	Partial Least Squares
PLS Algorithm	NIPALS
Validation Method	Test Set Validation
Validation Testing Criterion	Prob PRESS > 0.1
Number of Random Permutations	1000
Random Number Seed for Permutation	12345

Output 64.1.4 *continued*

Number of Observations Read	33
Number of Observations Used	33
Number of Observations Used for Training	18
Number of Observations Used for Testing	15

Output 64.1.5 Test Set Validation for the Number of PLS Factors

The HPPLS Procedure

Test Set Validation for the Number of Extracted Factors

Number of Extracted Factors	Root Mean PRESS	Prob > PRESS
0	3.056797	<.0001
1	2.630561	<.0001
2	1.00706	0.0080
3	0.664603	<.0001
4	0.521578	0.3690
5	0.500034	1.0000
6	0.513561	0.4790
7	0.501431	0.6930
8	1.055791	0.1510
9	1.435085	0.1050
10	1.720389	0.0280

Minimum Root Mean PRESS	0.500034
Minimizing Number of Factors	5
Smallest Number of Factors with p > 0.1	4

Percent Variation Accounted for by Partial Least Squares Factors

	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	81.16545	81.16545	48.33854	48.33854
2	16.81131	97.97676	32.54654	80.88508
3	1.76391	99.74067	11.44380	92.32888
4	0.19507	99.93574	3.83631	96.16519

References

De Jong, S. (1993). "SIMPLS: An Alternative Approach to Partial Least Squares Regression." *Chemometrics and Intelligent Laboratory Systems* 18:251–263.

De Jong, S., and Kiers, H. (1992). "Principal Covariates Regression." *Chemometrics and Intelligent Laboratory Systems* 14:155–164.

Dijkstra, T. K. (1983). "Some Comments on Maximum Likelihood and Partial Least Squares Methods." *Journal of Econometrics* 22:67–90.

Dijkstra, T. K. (1985). *Latent Variables in Linear Stochastic Models: Reflections on Maximum Likelihood and Partial Least Squares Methods*. 2nd ed. Amsterdam: Sociometric Research Foundation.

Frank, I., and Friedman, J. (1993). "A Statistical View of Some Chemometrics Regression Tools." *Technometrics* 35:109–135.

Geladi, P., and Kowalski, B. (1986). "Partial Least-Squares Regression: A Tutorial." *Analytica Chimica Acta* 185:1–17.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. New York: Macmillan.

Helland, I. S. (1988). "On the Structure of Partial Least Squares Regression." *Communications in Statistics—Simulation and Computation* 17:581–607.

Hoerl, A. E., and Kennard, R. W. (1970). "Ridge Regression: Biased Estimation for Non-orthogonal Problems." *Technometrics* 12:55–67.

Lindberg, W., Persson, J.-A., and Wold, S. (1983). "Partial Least-Squares Method for Spectrofluorimetric Analysis of Mixtures of Humic Acid and Ligninsulfonate." *Analytical Chemistry* 55:643–648.

McAvoy, T. J., Wang, N. S., Naidu, S., Bhat, N., Gunter, J., and Simmons, M. (1989). "Interpreting Biosensor Data via Backpropagation." *International Joint Conference on Neural Networks* 1:227–233.

Naes, T., and Martens, H. (1985). "Comparison of Prediction Methods for Multicollinear Data." *Communications in Statistics—Simulation and Computation* 14:545–576.

Ränner, S., Lindgren, F., Geladi, P., and Wold, S. (1994). "A PLS Kernel Algorithm for Data Sets with Many Variables and Fewer Objects." *Journal of Chemometrics* 8:111–125.

Sarle, W. S. (1994). "Neural Networks and Statistical Models." In *Proceedings of the Nineteenth Annual SAS Users Group International Conference*, 1538–1550. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI94/Sugi-94-255%20Sarle.pdf.

Shao, J. (1993). "Linear Model Selection by Cross-Validation." *Journal of the American Statistical Association* 88:486–494.

Tobias, R. D. (1995). "An Introduction to Partial Least Squares Regression." In *Proceedings of the Twentieth Annual SAS Users Group International Conference*, 1250–1257. Cary, NC: SAS Institute Inc. http://support.sas.com/rnd/app/stat/papers/pls.pdf.

Umetrics (1995). *Multivariate Analysis.* Three-day course. Winchester, MA: Umetrics.

Van den Wollenberg, A. L. (1977). "Redundancy Analysis: An Alternative to Canonical Correlation Analysis." *Psychometrika* 42:207–219.

Van der Voet, H. (1994). "Comparing the Predictive Accuracy of Models Using a Simple Randomization Test." *Chemometrics and Intelligent Laboratory Systems* 25:313–323.

Wold, H. (1966). "Estimation of Principal Components and Related Models by Iterative Least Squares." In *Multivariate Analysis*, edited by P. R. Krishnaiah, 391–420. New York: Academic Press.

Chapter 65
The HPPRINCOMP Procedure

Contents

Overview: HPPRINCOMP Procedure **5054**
- PROC HPPRINCOMP Features 5054
- PROC HPPRINCOMP Contrasted with PROC PRINCOMP 5055

Getting Started: HPPRINCOMP Procedure **5056**

Syntax: HPPRINCOMP Procedure **5060**
- PROC HPPRINCOMP Statement 5060
- BY Statement 5064
- CODE Statement 5064
- FREQ Statement 5065
- ID Statement 5065
- OUTPUT Statement 5066
- PARTIAL Statement 5067
- PERFORMANCE Statement 5067
- VAR Statement 5068
- WEIGHT Statement 5068

Details: HPPRINCOMP Procedure **5068**
- Computing Principal Components 5068
  - Eigenvalue Decomposition 5068
  - NIPALS 5069
  - ITERGS 5069
- Missing Values 5069
- Output Data Sets 5069
  - OUT= Data Set 5070
  - OUTSTAT= Data Set 5070
- Computational Method 5072
  - Multithreading 5072
- Displayed Output 5073
  - Performance Information 5073
  - Data Access Information 5073
  - Model Information 5073
  - Number of Observations 5073
  - Number of Variables 5073
  - Simple Statistics 5073
  - Centering and Scaling Information 5073
  - Explained Variation of Variables 5074
  - Correlation Matrix 5074

Regression Statistics 5074
Regression Coefficients 5074
Partial Correlation Matrix 5074
Total Variance 5074
Eigenvalues 5074
Eigenvectors 5074
Loadings 5075
Timing Information 5075
ODS Table Names 5075
Examples: HPPRINCOMP Procedure **5076**
Example 65.1: Analyzing Mean Temperatures of US Cities 5076
Example 65.2: Extracting Principal Components with NIPALS 5078
References **5080**

Overview: HPPRINCOMP Procedure

The HPPRINCOMP procedure is a high-performance procedure that performs principal component analysis. It is a high-performance version of the PRINCOMP procedure in SAS/STAT software, but it provides additional iterative methods to calculate the principal components.

Principal component analysis is a multivariate technique for examining relationships among several quantitative variables, providing an optimal way of reducing dimensionality by projecting the data onto a lower-dimensional orthogonal subspace that explains as much variation in those variables as possible. The choice between using factor analysis and using principal component analysis depends in part on your research objectives. You should use the HPPRINCOMP procedure if you are interested in summarizing data and detecting linear relationships. You can use principal component analysis to reduce the number of variables in regression, clustering, and so on.

PROC HPPRINCOMP Features

The main features of the HPPRINCOMP procedure are as follows:

- supports a PARTIAL statement for analyzing a partial correlation or covariance matrix
- supports a FREQ statement for grouped analysis
- supports a WEIGHT statement for weighted analysis
- produces an output data set that contains principal component scores and other observationwise statistics
- produces an output data set that contains means, standard deviations, number of observations, correlations or covariances, eigenvalues, and eigenvectors

The HPPRINCOMP procedure implements the following algorithms:

- eigenvalue decomposition, which uses the correlation or covariance of the data matrix and calculates all the principal components simultaneously
- nonlinear iterative partial least squares (NIPALS), which uses the data matrix and extracts the principal components successively
- the iterative method based on Gram-Schmidt orthogonalization (ITERGS) of Andrecut (2009), which uses the data matrix and extracts the principal components successively. The algorithm applies reorthogonalization correction to both the scores and the loadings at each iteration step.

Because the HPPRINCOMP procedure is a high-performance analytical procedure, it also exploits all the available cores and concurrent threads.

PROC HPPRINCOMP Contrasted with PROC PRINCOMP

The HPPRINCOMP procedure and the PRINCOMP procedure in SAS/STAT have the following similarities and differences:

- All statements that are available in PROC PRINCOMP are supported by the HPPRINCOMP procedure.
- The HPPRINCOMP procedure supports the OUTPUT statement, which is not available in PROC PRINCOMP.
- The HPPRINCOMP procedure can specify various methods to be used for calculating the principal components by using the METHOD= option, which is not available in PROC PRINCOMP.
- PROC PRINCOMP can accept ordinary SAS data sets and other types of special SAS data sets as input. The HPPRINCOMP procedure can accept only ordinary SAS data sets (raw data) as input.
- The HPPRINCOMP procedure does not support the PLOTS option that is available in PROC PRINCOMP.
- By default, PROC HPPRINCOMP performs computations on multiple threads. The PRINCOMP procedure executes on a single thread.

Getting Started: HPPRINCOMP Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpprigs.sas.)

The following data provide crime rates per 100,000 people in seven categories for each of the 50 US states in 1977:

```
title 'Crime Rates per 100,000 Population by State';

data Crime;
   input State $1-15 Murder Rape Robbery Assault
         Burglary Larceny Auto_Theft;
   datalines;
Alabama        14.2 25.2  96.8 278.3 1135.5 1881.9 280.7
Alaska         10.8 51.6  96.8 284.0 1331.7 3369.8 753.3
Arizona         9.5 34.2 138.2 312.3 2346.1 4467.4 439.5
Arkansas        8.8 27.6  83.2 203.4  972.6 1862.1 183.4
California     11.5 49.4 287.0 358.0 2139.4 3499.8 663.5
Colorado        6.3 42.0 170.7 292.9 1935.2 3903.2 477.1
Connecticut     4.2 16.8 129.5 131.8 1346.0 2620.7 593.2
Delaware        6.0 24.9 157.0 194.2 1682.6 3678.4 467.0
Florida        10.2 39.6 187.9 449.1 1859.9 3840.5 351.4
Georgia        11.7 31.1 140.5 256.5 1351.1 2170.2 297.9
Hawaii          7.2 25.5 128.0  64.1 1911.5 3920.4 489.4
Idaho           5.5 19.4  39.6 172.5 1050.8 2599.6 237.6
Illinois        9.9 21.8 211.3 209.0 1085.0 2828.5 528.6
Indiana         7.4 26.5 123.2 153.5 1086.2 2498.7 377.4
Iowa            2.3 10.6  41.2  89.8  812.5 2685.1 219.9
Kansas          6.6 22.0 100.7 180.5 1270.4 2739.3 244.3
Kentucky       10.1 19.1  81.1 123.3  872.2 1662.1 245.4
Louisiana      15.5 30.9 142.9 335.5 1165.5 2469.9 337.7
Maine           2.4 13.5  38.7 170.0 1253.1 2350.7 246.9
Maryland        8.0 34.8 292.1 358.9 1400.0 3177.7 428.5
Massachusetts   3.1 20.8 169.1 231.6 1532.2 2311.3 1140.1
Michigan        9.3 38.9 261.9 274.6 1522.7 3159.0 545.5
Minnesota       2.7 19.5  85.9  85.8 1134.7 2559.3 343.1
Mississippi    14.3 19.6  65.7 189.1  915.6 1239.9 144.4
Missouri        9.6 28.3 189.0 233.5 1318.3 2424.2 378.4
Montana         5.4 16.7  39.2 156.8  804.9 2773.2 309.2
Nebraska        3.9 18.1  64.7 112.7  760.0 2316.1 249.1
Nevada         15.8 49.1 323.1 355.0 2453.1 4212.6 559.2
New Hampshire   3.2 10.7  23.2  76.0 1041.7 2343.9 293.4
New Jersey      5.6 21.0 180.4 185.1 1435.8 2774.5 511.5
New Mexico      8.8 39.1 109.6 343.4 1418.7 3008.6 259.5
New York       10.7 29.4 472.6 319.1 1728.0 2782.0 745.8
North Carolina 10.6 17.0  61.3 318.3 1154.1 2037.8 192.1
North Dakota    0.9  9.0  13.3  43.8  446.1 1843.0 144.7
Ohio            7.8 27.3 190.5 181.1 1216.0 2696.8 400.4
Oklahoma        8.6 29.2  73.8 205.0 1288.2 2228.1 326.8
Oregon          4.9 39.9 124.1 286.9 1636.4 3506.1 388.9
Pennsylvania    5.6 19.0 130.3 128.0  877.5 1624.1 333.2
```

```
Rhode Island    3.6 10.5  86.5 201.0 1489.5 2844.1 791.4
South Carolina 11.9 33.0 105.9 485.3 1613.6 2342.4 245.1
South Dakota    2.0 13.5  17.9 155.7  570.5 1704.4 147.5
Tennessee      10.1 29.7 145.8 203.9 1259.7 1776.5 314.0
Texas          13.3 33.8 152.4 208.2 1603.1 2988.7 397.6
Utah            3.5 20.3  68.8 147.3 1171.6 3004.6 334.5
Vermont         1.4 15.9  30.8 101.2 1348.2 2201.0 265.2
Virginia        9.0 23.3  92.1 165.7  986.2 2521.2 226.7
Washington      4.3 39.6 106.2 224.8 1605.6 3386.9 360.3
West Virginia   6.0 13.2  42.2     .  597.4 1341.7 163.3
Wisconsin       2.8 12.9  52.2  63.7  846.9 2614.2 220.7
Wyoming          .  21.9  39.7 173.9  811.6 2772.2 282.0
;
```

The following statements invoke the HPPRINCOMP procedure, which requests a principal component analysis of the data and produces Figure 65.1 through Figure 65.4:

```
proc hpprincomp data=Crime;
run;
```

Figure 65.1 displays the "Performance Information," "Data Access Information," "Model Information," "Number of Observations," "Number of Variables," and "Simple Statistics" tables.

The "Performance Information" table shows the procedure executes in single-machine mode—that is, the data reside and the computation is performed on the machine where the SAS session executes. This run of the HPPRINCOMP procedure took place on a multicore machine with four CPUs; one computational thread was spawned per CPU.

The "Data Access Information" table shows that the input data set is accessed with the V9 (base) engine on the client machine where the MVA SAS session executes.

The "Model Information" table identifies the data source and shows that the principal component extraction method is eigenvalue decomposition, which is the default.

The "Number of Observations" table shows that of the 50 observations in the input data, only 48 observations are used in the analysis because some observations have incomplete data.

The "Number of Variables" table indicates that there are seven variables to be analyzed and seven principal components to be computed. By default, if the VAR statement is omitted, all numeric variables that are not listed in other statements are used in the analysis.

The "Simple Statistics" table displays the mean and standard deviation of the analysis variables.

Figure 65.1 Performance Information and Simple Statistics

Crime Rates per 100,000 Population by State

The HPPRINCOMP Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.CRIME	V9	Input	On Client

Figure 65.1 *continued*

Model Information	
Data Source	WORK.CRIME
Component Extraction Method	Eigenvalue Decomposition

Number of Observations Read	50
Number of Observations Used	48

Number of Variables	7
Number of Principal Components	7

Simple Statistics

Variable	Mean	Standard Deviation
Murder	7.51667	3.93059
Rape	26.07500	10.81304
Robbery	127.55625	88.49374
Assault	214.58750	100.64360
Burglary	1316.37917	423.31261
Larceny	2696.88542	714.75023
Auto_Theft	383.97917	194.37033

Figure 65.2 displays the "Correlation Matrix" table. By default, the PROC HPPRINCOMP statement requests that principal components be computed from the correlation matrix, so the total variance is equal to the number of variables, 7.

Figure 65.2 Correlation Matrix Table

Correlation Matrix

Variable	Murder	Rape	Robbery	Assault	Burglary	Larceny	Auto_Theft
Murder	1.0000	0.6000	0.4768	0.6485	0.3778	0.0925	0.0555
Rape	0.6000	1.0000	0.5817	0.7316	0.7038	0.6009	0.3282
Robbery	0.4768	0.5817	1.0000	0.5452	0.6200	0.4371	0.5787
Assault	0.6485	0.7316	0.5452	1.0000	0.6082	0.3791	0.2520
Burglary	0.3778	0.7038	0.6200	0.6082	1.0000	0.7932	0.5390
Larceny	0.0925	0.6009	0.4371	0.3791	0.7932	1.0000	0.4246
Auto_Theft	0.0555	0.3282	0.5787	0.2520	0.5390	0.4246	1.0000

Figure 65.3 displays the "Eigenvalues" table. The first principal component accounts for about 57.8% of the total variance, the second principal component accounts for about 18.1%, and the third principal component accounts for about 10.7%. Note that the eigenvalues sum to the total variance.

The eigenvalues indicate that two or three components provide a good summary of the data: two components account for 76% of the total variance, and three components account for 87%. Subsequent components account for less than 5% each.

Figure 65.3 Eigenvalues Table

Eigenvalues of the Correlation Matrix				
	Eigenvalue	**Difference**	**Proportion**	**Cumulative**
1	4.045824	2.781795	0.5780	0.5780
2	1.264030	0.516529	0.1806	0.7586
3	0.747500	0.421175	0.1068	0.8653
4	0.326325	0.061119	0.0466	0.9120
5	0.265207	0.036843	0.0379	0.9498
6	0.228364	0.105613	0.0326	0.9825
7	0.122750		0.0175	1.0000

Figure 65.4 displays the "Eigenvectors" table. From the eigenvectors matrix, you can represent the first principal component, Prin1, as a linear combination of the original variables:

$$\begin{aligned}\text{Prin1} &= 0.302888 \times (\text{Murder}) \\ &+ 0.434103 \times (\text{Rape}) \\ &+ 0.397055 \times (\text{Robbery}) \\ &\;\;\vdots \\ &+ 0.288343 \times (\text{Auto_Theft})\end{aligned}$$

Similarly, the second principal component, Prin2, is

$$\begin{aligned}\text{Prin2} &= -0.618929 \times (\text{Murder}) \\ &- 0.170526 \times (\text{Rape}) \\ &+ 0.047125 \times (\text{Robbery}) \\ &\;\;\vdots \\ &+ 0.504003 \times (\text{Auto_Theft})\end{aligned}$$

where the variables are standardized.

Figure 65.4 Eigenvectors Table

Eigenvectors							
Variable	**Prin1**	**Prin2**	**Prin3**	**Prin4**	**Prin5**	**Prin6**	**Prin7**
Murder	0.30289	-0.61893	0.17353	-0.23308	0.54896	0.26371	0.26428
Rape	0.43410	-0.17053	-0.23539	0.06540	0.18075	-0.78232	-0.27946
Robbery	0.39705	0.04713	0.49208	-0.57470	-0.50808	-0.09452	-0.02497
Assault	0.39622	-0.35142	-0.05343	0.61743	-0.51525	0.17395	0.19921
Burglary	0.44164	0.20861	-0.22454	-0.02750	0.11273	0.52340	-0.65085
Larceny	0.35634	0.40570	-0.53681	-0.23231	0.02172	0.04085	0.60346
Auto_Theft	0.28834	0.50400	0.57524	0.41853	0.35939	-0.06024	0.15487

The first component is a measure of the overall crime rate, because the first eigenvector shows approximately equal loadings on all variables. The second eigenvector has high positive loadings on the variables Auto_Theft and Larceny and high negative loadings on the variables Murder and Assault. There is also a small positive loading on the variable Burglary and a small negative loading on the variable Rape. This component seems to measure the preponderance of property crime compared to violent crime. The interpretation of the third component is not obvious.

Syntax: HPPRINCOMP Procedure

The following statements are available in the HPPRINCOMP procedure:

```
PROC HPPRINCOMP <options> ;
    BY variables ;
    CODE <options> ;
    FREQ variable ;
    ID variables ;
    OUTPUT <OUT=SAS-data-set>
            <keyword <=prefix>>...<keyword <=prefix>> ;
    PARTIAL variables ;
    PERFORMANCE performance-options ;
    VAR variables ;
    WEIGHT variable ;
```

The rest of this section provides detailed syntax information for each of the preceding statements, beginning with the PROC HPPRINCOMP statement. The remaining statements are described in alphabetical order.

PROC HPPRINCOMP Statement

PROC HPPRINCOMP *<options>* **;**

The PROC HPPRINCOMP statement invokes the HPPRINCOMP procedure. Optionally, it also identifies the input and output data sets, specifies the analyses to be performed, and controls displayed output. Table 65.1 summarizes the options available in the PROC HPPRINCOMP statement.

Table 65.1 PROC HPPRINCOMP Statement Options

Option	Description
Specify Data Sets	
DATA=	Specifies the name of the input data set
OUT=	Specifies the name of the output data set
OUTSTAT=	Specifies the name of the output data set that contains various statistics
Specify Details of Analysis	
COV	Computes the principal components from the covariance matrix
METHOD=	Specifies the principal component extraction method to be used

Table 65.1 *continued*

Option	Description
N=	Specifies the number of principal components to be computed
NOINT	Omits the intercept from the model
PREFIX=	Specifies a prefix for naming the principal components
PARPREFIX=	Specifies a prefix for naming the residual variables
SINGULAR=	Specifies the singularity criterion
STD	Standardizes the principal component scores
VARDEF=	Specifies the divisor used in calculating variances and standard deviations
Suppress the Display of Output	
NOPRINT	Suppresses the display of all output

The following list provides details about these *options*.

COVARIANCE
COV

computes the principal components from the covariance matrix. If you omit the COV option, the correlation matrix is analyzed. The COV option causes variables that have large variances to be more strongly associated with components that have large eigenvalues, and it causes variables that have small variances to be more strongly associated with components that have small eigenvalues. You should not specify the COV option unless the units in which the variables are measured are comparable or the variables are standardized in some way.

DATA=*SAS-data-set*

specifies the SAS data set to be analyzed. The data set can only be an ordinary SAS data set (raw data). If you omit the DATA= option, the HPPRINCOMP procedure uses the most recently created SAS data set.

METHOD=EIG | ITERGS<(*iter-options***)> | NIPALS<(***iter-options***)>**

specifies the principal component extraction method to be used. You can specify the following values:

EIG

requests eigenvalue decomposition.

ITERGS<(*iter-options***)>**

requests the iterative method based on Gram-Schmidt orthogonalization (ITERGS) of Andrecut (2009). You can also specify the following optional *iter-options* in parentheses after METHOD=ITERGS:

EPSILON=*n*

specifies the convergence criterion for the iterative method. By default, EPSILON=1E–12.

MAXITER=*n*
specifies the maximum number of iterations for the iterative method. By default, MAXITER=5000.

NOCENTER
suppresses centering of the numeric variables to be analyzed. This option is useful if the analysis variables are already centered and scaled.

NOSCALE
suppresses scaling of the numeric variables to be analyzed. This option is useful if the analysis variables are already centered and scaled.

NIPALS< (*iter-options***) >**
requests the nonlinear iterative partial least squares (NIPALS) method. You can also specify the optional *iter-options* in parentheses after METHOD=NIPALS.

By default, METHOD=EIG. If you specify METHOD=NIPALS or METHOD=ITERGS, the following options in the PROC HPPRINCOMP statement are ignored: COV, NOINT, OUT=, OUTSTAT=, PARPREFIX=, SINGULAR=, and STD.

N=*number*
specifies the number of principal components to be computed. The default is the number of variables. The value of the N= option must be an integer greater than or equal to 0.

NOINT
omits the intercept from the model. In other words, the NOINT option requests that the covariance or correlation matrix not be corrected for the mean. When you specify the NOINT option in the HPPRINCOMP procedure, the covariance matrix and, hence, the standard deviations are not corrected for the mean. If you want to obtain the standard deviations corrected for the mean, you can obtain them by using a procedure such as PROC MEANS.

If you use the NOINT option and also create an OUTSTAT= data set, the data set is TYPE=UCORR or TYPE=UCOV rather than TYPE=CORR or TYPE=COV.

NOPRINT
suppresses the display of all output. This option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*
creates an output SAS data set to contain observationwise principal component scores. To avoid data duplication when you have large data sets, the variables in the input data set are *not* included in the output data set; however, variables that are specified in the ID statement are included.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For more information about OUT= data sets, see the section "Output Data Sets" on page 5069.

OUTSTAT=*SAS-data-set*

creates an output SAS data set to contain means, standard deviations, number of observations, correlations or covariances, eigenvalues, and eigenvectors. If you specify the COV option, the data set is TYPE=COV or TYPE=UCOV, depending on the NOINT option, and it contains covariances; otherwise, the data set is TYPE=CORR or TYPE=UCORR, depending on the NOINT option, and it contains correlations. If you specify the PARTIAL statement, the OUTSTAT= data set also contains R squares.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For more information about OUTSTAT= data sets, see the section "Output Data Sets" on page 5069.

PREFIX=*name*

specifies a prefix for naming the principal components. By default, the *names* are Prin1, Prin2, . . . , Prin*n*. If you specify PREFIX=Abc, the components are named Abc1, Abc2, Abc3, and so on. The number of characters in the prefix plus the number of digits required to designate the variables should not exceed the current name length that is defined by the VALIDVARNAME= system option.

PARPREFIX=*name*

PPREFIX=*name*

RPREFIX=*name*

specifies a prefix for naming the residual variables in the OUT= data set and the OUTSTAT= data set. By default, the prefix is R_. The number of characters in the prefix plus the maximum length of the variable names should not exceed the current name length that is defined by the VALIDVARNAME= system option.

SINGULAR=*p*

SING=*p*

specifies the singularity criterion, where $0 < p < 1$. If a variable in a PARTIAL statement has an R square as large as $1 - p$ when predicted from the variables listed before it in the statement, the variable is assigned a standardized coefficient of 0. By default, SINGULAR=1E–8.

STANDARD

STD

standardizes the principal component scores in the OUT= data set to unit variance. If you omit the STANDARD option, the scores have a variance equal to the corresponding eigenvalue. Note that the STANDARD option has no effect on the eigenvalues themselves.

VARDEF=DF | N | WDF | WEIGHT | WGT

specifies the divisor to be used in calculating variances and standard deviations. By default, VARDEF=DF. The following table displays the values and associated divisors:

Value	Divisor	Formula	
DF	Error degrees of freedom	$n - i$	(before partialing)
		$n - p - i$	(after partialing)
N	Number of observations	n	
WEIGHT | WGT	Sum of weights	$\sum_{j=1}^{n} w_j$	

Table 65.2 *continued*

Value	Divisor	Formula	
WDF	Sum of weights minus one	$\left(\sum_{j=1}^{n} w_j\right) - i$	(before partialing)
		$\left(\sum_{j=1}^{n} w_j\right) - p - i$	(after partialing)

In the formulas for VARDEF=DF and VARDEF=WDF, p is the number of degrees of freedom of the variables in the PARTIAL statement, and i is 0 if the NOINT option is specified and 1 otherwise.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPPRINCOMP to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPPRINCOMP procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CODE Statement

CODE < *options* > **;**

The CODE statement enables you to write SAS DATA step code for computing the principal component scores either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

The CODE statement is not supported when the PARTIAL statement is specified. If you specify more than one CODE statement, only the last one specified is used.

Table 65.3 summarizes the *options* available in the CODE statement.

Table 65.3 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the eigenvectors
GROUP=	Specifies the group identifier for array names and statement labels
LINESIZE=	Specifies the line size of the generated code

For more information about the syntax of the CODE statement, see the section "CODE Statement" in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence of each observation. SAS high-performance analytics procedures that support the FREQ statement treat each observation as if it appeared f times, where f is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

The FREQ statement is not supported if you specify METHOD=NIPALS or METHOD=ITERGS in the PROC HPPRINCOMP statement.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are transferred to output data sets created by SAS high-performance analytics procedures, provided that the output data set produces one (or more) records per input observation.

For information about the common ID statement in SAS high-performance analytics procedures, see the section "ID Statement" on page 576.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< *keyword* < **=***prefix* > >. . . < *keyword* < **=***prefix* > > **;**

The OUTPUT statement creates a data set that contains observationwise statistics, which are computed after PROC HPPRINCOMP fits the model. If you do not specify a *keyword*, then only the principal component scores are included.

The OUTPUT statement causes the OUT= option in the PROC HPPRINCOMP statement to be ignored.

The variables in the input data set are *not* included in the output data set, in order to avoid data duplication for large data sets; however, variables that you specify in the ID statement are included.

You can specify the following syntax elements:

OUT=*SAS-data-set*

DATA=*SAS-data-set*

specifies the name of the output data set. If you omit this option, the procedure uses the DATA*n* convention to name the output data set.

keyword **< =***prefix* **>**

specifies a statistic to include in the output data set and optionally a *prefix* for naming the output variables. If you do not provide a *prefix*, the HPPRINCOMP procedure assigns a default prefix based on the type of statistic requested. For example, for the VAR variables x1 and x2, RESIDUAL produces two residual value variables, R_x1 and R_x2.

You can specify the following *keywords* to add statistics to the OUTPUT data set:

H

requests the approximate leverage. The default prefix is H.

STD

requests standardized (centered and scaled) VAR variable values for each VAR variable. The default prefix is Std.

STDSSE

requests the sum of squares of residuals for standardized VAR variables. The default prefix is StdSSE.

TSQUARE

T2

requests scaled sum of squares of score values. The default prefix is TSquare.

RESIDUAL

RESID

R

requests residuals for each VAR variable. The default prefix is R.

SCORE
requests principal component scores for each principal component. The default prefix is Score.

If you specify METHOD=EIG, the only valid *keywords* are RESIDUAL (if you also specify the PARTIAL statement) and SCORE. Other *keywords* are ignored.

The output variables that contain the requested statistic are named as follows, according to the *keyword* that you specify:

- The *keywords* RESIDUAL and STD define an output variable for each VAR variable, so the variables that correspond to each VAR variable are named by appending the name of the VAR variable to the prefix. For example, if the model has the VAR variables x1 and x2, then RESIDUAL=R produces the variables R_x1 and R_x2.
- The *keyword* SCORE defines an output variable for each principal component, so the variables that correspond to each successive component are named by appending the component number to the prefix. For example, if the model has three principal components, then SCORE=T produces the variables T1, T2, and T3.
- The *keywords* H, STDSSE, and TSQUARE each define a single output variable, so the variable name matches the prefix.

PARTIAL Statement

PARTIAL *variables* **;**

If you want to analyze a partial correlation or covariance matrix, specify the names of the numeric variables to be partialed out in the PARTIAL statement. The HPPRINCOMP procedure computes the principal components of the residuals from the prediction of the VAR variables by the PARTIAL variables. If you request an OUT= or OUTSTAT= data set, the residual variables are named by prefixing either the characters R_ (by default) or the string specified in the PARPREFIX= option to the VAR variables.

The PARTIAL statement is not supported if you specify METHOD=NIPALS or METHOD=ITERGS in the PROC HPPRINCOMP statement.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of the HPPRINCOMP procedure.

The PERFORMANCE statement is documented further in the section "PERFORMANCE Statement" on page 569.

VAR Statement

VAR *variables* **;**

The VAR statement lists the numeric variables to be analyzed. If you omit the VAR statement, all numeric variables that are not specified in other statements are analyzed.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If you do not specify a WEIGHT statement, all observations that are used in the analysis are assigned a weight of 1.

The WEIGHT statement is not supported if you specify METHOD=NIPALS or METHOD=ITERGS in the PROC HPPRINCOMP statement.

Details: HPPRINCOMP Procedure

Computing Principal Components

The HPPRINCOMP procedure implements several algorithms to calculate principal components: eigenvalue decomposition, NIPALS, and ITERGS of Andrecut (2009). Eigenvalue decomposition is more efficient when you want to calculate all principal components, whereas the NIPALS method is faster if you want to extract only the first few principal components. For high-dimensional data sets, the NIPALS method is more efficient, whereas it gets expensive for eigenvalue decomposition to calculate all the components simultaneously.

Eigenvalue Decomposition

Let $\mathbf{X}$ be a centered and scaled data matrix that has k numerical variables. The eigenvalue decomposition method bases the component extraction on the eigenvalue decomposition of the covariance matrix $\mathbf{X}'\mathbf{X}$, which extracts all the k principal components simultaneously. Each principal component is a linear combination of the original variables, and each component is orthogonal, with coefficients equal to the eigenvectors of the covariance matrix $\mathbf{X}'\mathbf{X}$. The eigenvectors are usually normalized to have unit length. The principal components are sorted by descending order of the eigenvalues, which are equal to the variances of the components.

NIPALS

The nonlinear iterative partial least squares (NIPALS) method extracts the principal components successively based on the data matrix $\mathbf{X}$. The NIPALS method starts by calculating the loadings, $\mathbf{p}$, as $\mathbf{p}' = (\mathbf{t}'\mathbf{t})^{-1}\mathbf{t}'\mathbf{X}$, where $\mathbf{t}$ is the score vector. It then calculates an improved score vector, $\mathbf{t} = \mathbf{X}\mathbf{p}$. The method iteratively computes the improved $\mathbf{p}$ and $\mathbf{t}$ until convergence is reached.

This process accounts for how the first principal component is extracted. The second component is extracted in the same way, by replacing $\mathbf{X}$ with the residual from the first component: $\mathbf{E} = \mathbf{X} - \mathbf{t}\mathbf{p}'$.

For large data matrices or matrices that have a high degree of column collinearity, the NIPALS method suffers from loss of orthogonality because of the machine-precision errors that accumulate at each iteration step. In practice, the NIPALS method is used to extract only the first few principal components.

ITERGS

The iterative method based on Gram-Schmidt orthogonalization (ITERGS) of Andrecut (2009) overcomes the issue of loss of orthogonality in the NIPALS method by applying Gram-Schmidt reorthogonalization correction to both the loadings and the scores at each iteration step:

$$\begin{aligned} \mathbf{p}_c &= \mathbf{p} - \mathbf{P}_k\mathbf{P}_k'\mathbf{p} \\ \mathbf{t}_c &= \mathbf{t} - \mathbf{T}_k\mathbf{T}_k'\mathbf{t} \end{aligned}$$

Here, $\mathbf{p}_c$ and $\mathbf{t}_c$ are the corrected loading vector and score vector, respectively. $\mathbf{P}_k$ is the matrix that is formed by using the first k loadings. $\mathbf{T}_k$ is the matrix that is formed by using the first k scores.

The ITERGS method stabilizes the iterative process at the cost of increased computational effort.

Missing Values

Observations that have missing values for any variable in the VAR, PARTIAL, FREQ, or WEIGHT statement are omitted from the analysis and are given missing values for principal component scores in the OUT= data set.

Output Data Sets

When an observationwise output data set is created, many SAS procedures add the variables from the input data set to the output data set. High-performance statistical procedures assume that the input data sets can be large and can contain many variables. For performance reasons, the output data set contains only the following:

- variables that are explicitly created by the statement
- variables that are listed in the ID statement

OUT= Data Set

The new variables that are created for the OUT= data set contain the principal component scores. The N= option determines the number of new variables. The names of the new variables are formed by concatenating the value given by the PREFIX= option (or Prin if PREFIX= is omitted) to the numbers 1, 2, 3, and so on. The new variables have mean 0 and a variance equal to the corresponding eigenvalue, unless you specify the STANDARD option to standardize the scores to unit variance. Also, if you specify the COV option, PROC HPPRINCOMP computes the principal component scores from the corrected or uncorrected (if the NOINT option is specified) variables rather than from the standardized variables.

If you use a PARTIAL statement, the OUT= data set also contains the residuals from predicting the VAR variables from the PARTIAL variables.

OUTSTAT= Data Set

The OUTSTAT= data set is similar to the TYPE=CORR data set that the CORR procedure produces. The following table relates the TYPE= value for the OUTSTAT= data set to the options that are specified in the PROC HPPRINCOMP statement:

Options	TYPE=
(Default)	CORR
COV	COV
NOINT	UCORR
COV NOINT	UCOV

Note that the default (neither the COV nor NOINT option) produces a TYPE=CORR data set.

The new data set contains the following variables:

- the BY variables, if any
- two new variables, _TYPE_ and _NAME_, both character variables
- the variables that are analyzed (that is, those in the VAR statement); or, if there is no VAR statement, all numeric variables not listed in any other statement; or, if there is a PARTIAL statement, the residual variables as described in the section "OUT= Data Set"

Each observation in the new data set contains some type of statistic, as indicated by the _TYPE_ variable. The values of the _TYPE_ variable are as follows:

TYPE	Contents
MEAN	mean of each variable. If you specify the PARTIAL statement, this observation is omitted.
STD	standard deviations. If you specify the COV option, this observation is omitted, so the SCORE procedure does not standardize the variables before computing scores. If you use the PARTIAL statement, the standard deviation of a variable is computed as its root mean squared error as predicted from the PARTIAL variables.
USTD	uncorrected standard deviations. When you specify the NOINT option in the PROC HPPRINCOMP statement, the OUTSTAT= data set contains standard deviations not corrected for the mean. However, if you also specify the COV option in the PROC HPPRINCOMP statement, this observation is omitted.

N — number of observations on which the analysis is based. This value is the same for each variable. If you specify the PARTIAL statement and the value of the VARDEF= option is DF or unspecified, then the number of observations is decremented by the degrees of freedom for the PARTIAL variables.

SUMWGT — the sum of the weights of the observations. This value is the same for each variable. If you specify the PARTIAL statement and VARDEF=WDF, then the sum of the weights is decremented by the degrees of freedom for the PARTIAL variables. This observation is output only if the value is different from that in the observation for which _TYPE_='N'.

CORR — correlations between each variable and the variable specified by the _NAME_ variable. The number of observations for which _TYPE_='CORR' is equal to the number of variables being analyzed. If you specify the COV option, no _TYPE_='CORR' observations are produced. If you use the PARTIAL statement, the partial correlations, not the raw correlations, are output.

UCORR — uncorrected correlation matrix. When you specify the NOINT option without the COV option in the PROC HPPRINCOMP statement, the OUTSTAT= data set contains a matrix of correlations not corrected for the means. However, if you also specify the COV option in the PROC HPPRINCOMP statement, this observation is omitted.

COV — covariances between each variable and the variable specified by the _NAME_ variable. _TYPE_='COV' observations are produced only if you specify the COV option. If you use the PARTIAL statement, the partial covariances, not the raw covariances, are output.

UCOV — uncorrected covariance matrix. When you specify the NOINT and COV options in the PROC HPPRINCOMP statement, the OUTSTAT= data set contains a matrix of covariances not corrected for the means.

EIGENVAL — eigenvalues. If the N= option requests less than the maximum number of principal components, only the specified number of eigenvalues are produced, with missing values filling out the observation.

SCORE — eigenvectors. The _NAME_ variable contains the name of the corresponding principal component as constructed from the PREFIX= option. The number of observations for which _TYPE_='SCORE' equals the number of principal components computed. The eigenvectors have unit length unless you specify the STD option, in which case the unit-length eigenvectors are divided by the square roots of the eigenvalues to produce scores that have unit standard deviations.

To obtain the principal component scores, if the COV option is not specified, these coefficients should be multiplied by the standardized data. For the COV option, these coefficients should be multiplied by the centered data. To center and standardize the data, you should use means that are obtained from the observation for which _TYPE_='MEAN' and standard deviations that are obtained from the observation for which _TYPE_='STD'.

USCORE — scoring coefficients to be applied without subtracting the mean from the raw variables. Observations for which _TYPE_='USCORE' are produced when you specify the NOINT option in the PROC HPPRINCOMP statement.

To obtain the principal component scores, these coefficients should be multiplied by the data that are standardized by the uncorrected standard deviations obtained from the observation for which _TYPE_='USTD'.

RSQUARED — R squares for each VAR variable as predicted by the PARTIAL variables.

B	regression coefficients for each VAR variable as predicted by the PARTIAL variables. This observation is produced only if you specify the COV option.
STB	standardized regression coefficients for each VAR variable as predicted by the PARTIAL variables. If you specify the COV option, this observation is omitted.

You can use the data set in the SCORE procedure to compute principal component scores, or you can use it as input to the FACTOR procedure and specify METHOD=SCORE to rotate the components. If you use the PARTIAL statement, the scoring coefficients should be applied to the residuals, not to the original variables.

Computational Method

Multithreading

Threading is the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading is the concurrent execution of threads. When multithreading is possible, you can realize substantial performance gains compared to the performance that you get from sequential (single-threaded) execution.

The number of threads that the HPPRINCOMP procedure spawns is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the CPU count by using the CPUCOUNT= SAS system option. For example, if you specify the following statements, the HPPRINCOMP procedure schedules threads as if it were executing on a system that had four CPUs, regardless of the actual CPU count:

```
options cpucount=4;
```

- You can specify the NTHREADS= option in the PERFORMANCE statement to determine the number of threads. This specification overrides the system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads per machine is displayed in the “Performance Information” table, which is part of the default output. The HPPRINCOMP procedure allocates one thread per CPU.

The tasks that are multithreaded by the HPPRINCOMP procedure are primarily defined by dividing the data processed on a single machine among the threads; that is, PROC HPPRINCOMP implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and you are running on four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. Those operations include the following:

- formation of the crossproducts matrix
- computation of loadings, scores, and residual sums of squares
- principal component scoring of observations

Displayed Output

The following sections describe the output that PROC HPPRINCOMP produces. The output is organized into various tables, which are discussed in order of appearance.

Performance Information

The "Performance Information" table is produced by default. It displays the number of threads used for single-machine mode.

Data Access Information

The "Data Access Information" table is produced by default. For the input and output data sets, it displays the libref and data set name, the engine used to access the data, the role (input or output) of the data set, and the path that data followed to reach the computation.

Model Information

The "Model Information" table displays basic information about the model, including the input data set and the principal component extraction method that is used in the analysis.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used.

Number of Variables

The "Number of Variables" table displays the number of VAR variables, the number of PARTIAL variables, and the number of principal components to be extracted.

Simple Statistics

If you specify METHOD=EIG, the HPPRINCOMP procedure produces a "Simple Statistics" table that displays the mean and standard deviation (std) for each variable. If you specify the NOINT option, the uncorrected standard deviation (ustd) is displayed.

Centering and Scaling Information

If you specify METHOD=NIPALS or METHOD=ITERGS, the HPPRINCOMP procedure produces a "Centering and Scaling Information" table that displays the centering and scaling information for each variable.

Explained Variation of Variables

If you specify METHOD=NIPALS or METHOD=ITERGS, the HPPRINCOMP procedure produces an "Explained Variation of Variables" table that displays the fraction of variation that is accounted for in each variable by each successive principal component.

Correlation Matrix

If you specify METHOD=EIG, the HPPRINCOMP procedure produces a "Correlation Matrix" table that displays the correlation or, if you specify the COV option, the covariance matrix.

Regression Statistics

When you specify the PARTIAL statement, the HPPRINCOMP procedure produces a "Regression Statistics" table that displays the R square and root mean squared error (RMSE) for each VAR variable as predicted by the PARTIAL variables.

Regression Coefficients

When you specify the PARTIAL statement, the HPPRINCOMP procedure produces a "Regression Coefficients" table that displays standardized regression coefficients or, if you specify the COV option, regression coefficients for predicting the VAR variables from the PARTIAL variables.

Partial Correlation Matrix

When you specify the PARTIAL statement, the HPPRINCOMP procedure produces a "Partial Correlation Matrix" table that displays the partial correlation matrix or, if you specify the COV option, the partial covariance matrix.

Total Variance

If you specify METHOD=EIG and the COV option, the HPPRINCOMP procedure produces a simple table that displays the total variance.

Eigenvalues

The "Eigenvalues" table displays eigenvalues of the correlation or covariance matrix (if you specify METHOD=EIG) or eigenvalues of the data matrix (if you specify METHOD=NIPALS or METHOD=ITERGS), along with the difference between successive eigenvalues, the proportion of variance explained by each eigenvalue, and the cumulative proportion of variance explained.

Eigenvectors

If you specify METHOD=EIG, the HPPRINCOMP procedure produces an "Eigenvectors" table that displays the eigenvectors.

Loadings

If you specify METHOD=NIPALS or METHOD=ITERGS, the HPPRINCOMP procedure produces a "Loadings" table that displays the loadings.

Timing Information

If you specify the DETAILS option in the PERFORMANCE statement, the HPPRINCOMP procedure produces a "Timing" table that displays the elapsed time of each main task of the procedure.

ODS Table Names

PROC HPPRINCOMP assigns a name to each table that it creates. You can use these names to reference the ODS table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 65.4. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 65.4 ODS Tables Produced by PROC HPPRINCOMP

Table Name	Description	Required Statement / Option
CenScaleInfo	Centering and scaling information	METHOD=NIPALS | ITERGS
Corr	Correlation matrix	METHOD=EIG
Cov	Covariance matrix	METHOD=EIG and COV
DataAccessInfo	Information about modes of data access	Default output
Eigenvalues	Eigenvalues	Default output
Eigenvectors	Eigenvectors	METHOD=EIG
Loadings	Loadings	METHOD=NIPALS | ITERGS
ModelInfo	Model information	Default output
NObs	Number of observations read and used	Default output
NVars	Number of variables, partial variables, and principal components	Default output
ParCorr	Partial correlation matrix	PARTIAL statement
ParCov	Uncorrected partial covariance matrix	PARTIAL statement and COV
PerformanceInfo	Information about the high-performance computing environment	Default output
RegCoef	Regression coefficients	PARTIAL statement and COV
RSquareRMSE	Regression statistics: R-squares and RMSEs	PARTIAL statement
SimpleStatistics	Simple statistics	METHOD=EIG
StdRegCoef	Standardized regression coefficients	PARTIAL statement
Timing	Absolute and relative times of tasks that are performed by the procedure	DETAILS option in PERFORMANCE statement
TotalVariance	Total variance	METHOD=EIG and COV
Variation	Explained variation of variables	METHOD=NIPALS | ITERGS

Examples: HPPRINCOMP Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 65.1: Analyzing Mean Temperatures of US Cities

(View the complete code for this example (hppriex1.sas) in the example repository.)

This example analyzes mean daily temperatures of selected US cities in January and July. The following statements create the Temperature data set:

```
data Temperature;
   length Cityid $ 2;
   title 'Mean Temperature in January and July for Selected Cities ';
   input City $1-15 January July;
   Cityid = substr(City,1,2);
   datalines;
Mobile          51.2 81.6
Phoenix         51.2 91.2
Little Rock     39.5 81.4
Sacramento      45.1 75.2
Denver          29.9 73.0

   ... more lines ...

Cheyenne        26.6 69.1
;
```

The following statements invoke the HPPRINCOMP procedure, which requests a principal component analysis of the Temperature data set and outputs the scores to the Scores data set (OUT= Scores). The Cityid variable in the ID statement is also included in the output data set.

```
title 'Mean Temperature in January and July for Selected Cities';
proc hpprincomp data=Temperature cov out=Scores;
   var July January;
   id Cityid;
run;
```

Output 65.1.1 displays the PROC HPPRINCOMP output. The standard deviation of January (11.712) is higher than the standard deviation of July (5.128). The COV option in the PROC HPPRINCOMP statement requests that the principal components be computed from the covariance matrix. The total variance is 163.474. The first principal component accounts for about 94% of the total variance, and the second principal component accounts for only about 6%. The eigenvalues sum to the total variance.

Note that January receives a higher loading on Prin1 because it has a higher standard deviation than July. Also note that the HPPRINCOMP procedure calculates the scores by using the centered variables rather than the standardized variables.

Output 65.1.1 Results of Principal Component Analysis

Mean Temperature in January and July for Selected Cities

The HPPRINCOMP Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.TEMPERATURE	V9	Input	On Client
WORK.SCORES	V9	Output	On Client

Model Information	
Data Source	WORK.TEMPERATURE
Component Extraction Method	Eigenvalue Decomposition

Number of Observations Read	64
Number of Observations Used	64

Number of Variables	2
Number of Principal Components	2

Simple Statistics		
Variable	**Mean**	**Standard Deviation**
July	75.60781	5.12762
January	32.09531	11.71243

Covariance Matrix		
Variable	**July**	**January**
July	26.29248	46.82829
January	46.82829	137.18109

Total Variance	163.47356647

Eigenvalues of the Covariance Matrix				
	Eigenvalue	**Difference**	**Proportion**	**Cumulative**
1	154.310607	145.147647	0.9439	0.9439
2	9.162960		0.0561	1.0000

Eigenvectors		
Variable	**Prin1**	**Prin2**
July	0.34353	0.93914
January	0.93914	-0.34353

Example 65.2: Extracting Principal Components with NIPALS

(View the complete code for this example (hppriex2.sas) in the example repository.)

This example demonstrates the NIPALS method in PROC HPPRINCOMP, which extracts principal components successively. The data that this example uses are from the Getting Started section; they provide crime rates per 100,000 people in seven categories for each of the 50 US states in 1977. The following DATA step generates the data:

```
data Crime;
   title 'Crime Rates per 100,000 Population by State';
   input State $1-15 Murder Rape Robbery Assault
         Burglary Larceny Auto_Theft;
   datalines;
Alabama        14.2 25.2  96.8 278.3 1135.5 1881.9 280.7
Alaska         10.8 51.6  96.8 284.0 1331.7 3369.8 753.3
Arizona         9.5 34.2 138.2 312.3 2346.1 4467.4 439.5
Arkansas        8.8 27.6  83.2 203.4  972.6 1862.1 183.4
California     11.5 49.4 287.0 358.0 2139.4 3499.8 663.5

   ... more lines ...

Wisconsin       2.8 12.9  52.2  63.7  846.9 2614.2 220.7
Wyoming          .  21.9  39.7 173.9  811.6 2772.2 282.0
;
```

The following statements use PROC HPPRINCOMP to extract principal components by using the NIPALS method:

```
proc hpprincomp data=Crime method=nipals;
run;
```

Output 65.2.1 displays the PROC HPPRINCOMP output. The “Model Information” table shows that the NIPALS method is used to extract principal components. The “Explained Variation of Variables” table lists the fraction of variation that is accounted for in each variable by each of the seven principal components. All the variation in each variable is accounted for by seven principal components because there are only seven variables. The eigenvalues indicate that two or three components provide a good summary of the data: two components account for 76% of the total variance, and three components account for 87%. Subsequent components account for less than 5% each.

Note that in the Getting Started section, the principal components are extracted from the same data by using the eigenvalue decomposition method; the “Eigenvalues” table generated there matches the one generated by the NIPALS method. Also, the eigenvectors in the “Eigenvectors” table match the loading factors in the “Loadings” table.

Output 65.2.1 Results of Principal Component Analysis Using NIPALS

Crime Rates per 100,000 Population by State

The HPPRINCOMP Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.CRIME	V9	Input	On Client

Model Information	
Data Source	WORK.CRIME
Component Extraction Method	NIPALS

Number of Observations Read	50
Number of Observations Used	48

Number of Variables	7
Number of Principal Components	7

Centering and Scaling Information		
Variable	**Subtracted off**	**Divided by**
Murder	7.51667	3.93059
Rape	26.07500	10.81304
Robbery	127.55625	88.49374
Assault	214.58750	100.64360
Burglary	1316.37917	423.31261
Larceny	2696.88542	714.75023
Auto_Theft	383.97917	194.37033

Explained Variation of Variables							
Variable	**Prin1**	**Prin2**	**Prin3**	**Prin4**	**Prin5**	**Prin6**	**Prin7**
Murder	0.37117	0.85539	0.87790	0.89562	0.97555	0.99143	1.00000
Rape	0.76242	0.79917	0.84059	0.84199	0.85065	0.99041	1.00000
Robbery	0.63783	0.64064	0.82164	0.92942	0.99788	0.99992	1.00000
Assault	0.63517	0.79127	0.79341	0.91781	0.98822	0.99513	1.00000
Burglary	0.78913	0.84414	0.88183	0.88207	0.88544	0.94800	1.00000
Larceny	0.51373	0.72178	0.93718	0.95479	0.95492	0.95530	1.00000
Auto_Theft	0.33638	0.65746	0.90481	0.96197	0.99623	0.99706	1.00000

Output 65.2.1 *continued*

Eigenvalues of the Data Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	4.045824	2.781795	0.5780	0.5780
2	1.264030	0.516529	0.1806	0.7586
3	0.747500	0.421175	0.1068	0.8653
4	0.326325	0.061119	0.0466	0.9120
5	0.265207	0.036843	0.0379	0.9498
6	0.228364	0.105613	0.0326	0.9825
7	0.122750		0.0175	1.0000

Loadings							
Variable	**Prin1**	**Prin2**	**Prin3**	**Prin4**	**Prin5**	**Prin6**	**Prin7**
Murder	0.30289	-0.61893	0.17353	-0.23308	0.54896	-0.26371	-0.26428
Rape	0.43410	-0.17053	-0.23539	0.06540	0.18075	0.78232	0.27946
Robbery	0.39705	0.04713	0.49208	-0.57470	-0.50808	0.09452	0.02497
Assault	0.39622	-0.35142	-0.05343	0.61744	-0.51525	-0.17395	-0.19921
Burglary	0.44164	0.20861	-0.22454	-0.02750	0.11273	-0.52340	0.65085
Larceny	0.35634	0.40570	-0.53681	-0.23231	0.02172	-0.04085	-0.60346
Auto_Theft	0.28834	0.50400	0.57524	0.41853	0.35939	0.06024	-0.15487

References

Andrecut, M. (2009). "Parallel GPU Implementation of Iterative PCA Algorithms." *Journal of Computational Biology* 16:1593–1599.

Cooley, W. W., and Lohnes, P. R. (1971). *Multivariate Data Analysis*. New York: John Wiley & Sons.

Gnanadesikan, R. (1977). *Methods for Statistical Data Analysis of Multivariate Observations*. New York: John Wiley & Sons.

Hotelling, H. (1933). "Analysis of a Complex of Statistical Variables into Principal Components." *Journal of Educational Psychology* 24:417–441, 498–520.

Kshirsagar, A. M. (1972). *Multivariate Analysis*. New York: Marcel Dekker.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Morrison, D. F. (1976). *Multivariate Statistical Methods*. 2nd ed. New York: McGraw-Hill.

Pearson, K. (1901). "On Lines and Planes of Closest Fit to Systems of Points in Space." *Philosophical Magazine* 6:559–572.

Rao, C. R. (1964). "The Use and Interpretation of Principal Component Analysis in Applied Research." *Sankhyā, Series A* 26:329–358.

Wold, S., Esbensen, K., and Geladi, P. (1987). "Principal Component Analysis." *Chemometrics and Intelligent Laboratory Systems* 2:37–52.

Chapter 66
The HPQUANTSELECT Procedure

Contents

Overview: HPQUANTSELECT Procedure **5082**
PROC HPQUANTSELECT Features 5084
PROC HPQUANTSELECT Contrasted with Other SAS Procedures 5085
Getting Started: HPQUANTSELECT Procedure **5086**
Syntax: HPQUANTSELECT Procedure **5095**
PROC HPQUANTSELECT Statement 5095
BY Statement 5097
CLASS Statement 5097
CODE Statement 5098
ID Statement 5098
MODEL Statement 5099
OUTPUT Statement 5101
PARTITION Statement 5103
PERFORMANCE Statement 5103
SELECTION Statement 5104
WEIGHT Statement 5105
Details: HPQUANTSELECT Procedure **5105**
Quantile Regression 5105
Linear Model with iid Errors 5106
Linear-in-Parameter Model with Non-iid Settings 5107
More Statistics for Parameter Estimates 5108
Criteria Used in Model Selection 5108
Quasi-likelihood Information Criteria 5109
Statistical Tests for Significance Level 5110
Diagnostic Statistics 5112
Classification Variables and the SPLIT Option 5112
Macro Variables That Contain Selected Effects 5113
Using Validation and Test Data 5115
Using the Validation ACL as the STOP= Criterion 5116
Using the Validation ACL as the CHOOSE= Criterion 5116
Using the Validation ACL as the SELECT= Criterion 5116
Computational Method 5116
Multithreading 5116
Output Data Set 5117
Displayed Output 5118
Performance Information 5118

Data Access Information 5118
Model Information 5118
Selection Information 5118
Number of Observations 5118
Class Level Information 5118
Dimensions 5119
Entry and Removal Candidates 5119
Selection Summary 5119
Stop Reason 5119
Selection Reason 5120
Selected Effects 5120
Fit Statistics 5120
Parameter Estimates 5120
Timing Information 5121
ODS Table Names 5121
Examples: HPQUANTSELECT Procedure **5122**
Example 66.1: Simulation Study 5122
Example 66.2: Growth Charts for Body Mass Index 5125
References **5128**

Overview: HPQUANTSELECT Procedure

Quantile regression is a systematic statistical methodology for modeling conditional quantile functions of a response variable on explanatory covariate effects. Although modern quantile regression was introduced by Koenker and Bassett (1978), simple quantile regression that uses only the intercept as the explanatory effect has been practiced for much longer, because quantile is no more than a generalized notion for terms such as percentile, decile, quintile, and quartile. A conditional quantile of a response variable at quantile level τ denotes the value below which the proportion of the conditional response population is τ. Unlike linear regression, which exclusively focuses on the conditional mean, quantile regression can anatomize the entire response distribution and examine how the covariate effects influence the shape of the response distribution over the entire range of quantile levels $[0, 1]$. Therefore, quantile regression provides a more comprehensive view of the regression relationship. Figure 66.1 shows an example of quantile regression that creates growth charts for the men's body mass index (BMI) as quantile curves. Each entry in the legend shows the quantile level for the corresponding quantile curve. For example, the curve whose quantile level $\tau = 0.85$ corresponds to the 85th conditional percentile. For more information about the BMI example, see "Example 66.2: Growth Charts for Body Mass Index" on page 5125.

Figure 66.1 Growth Chart for Body Mass Index

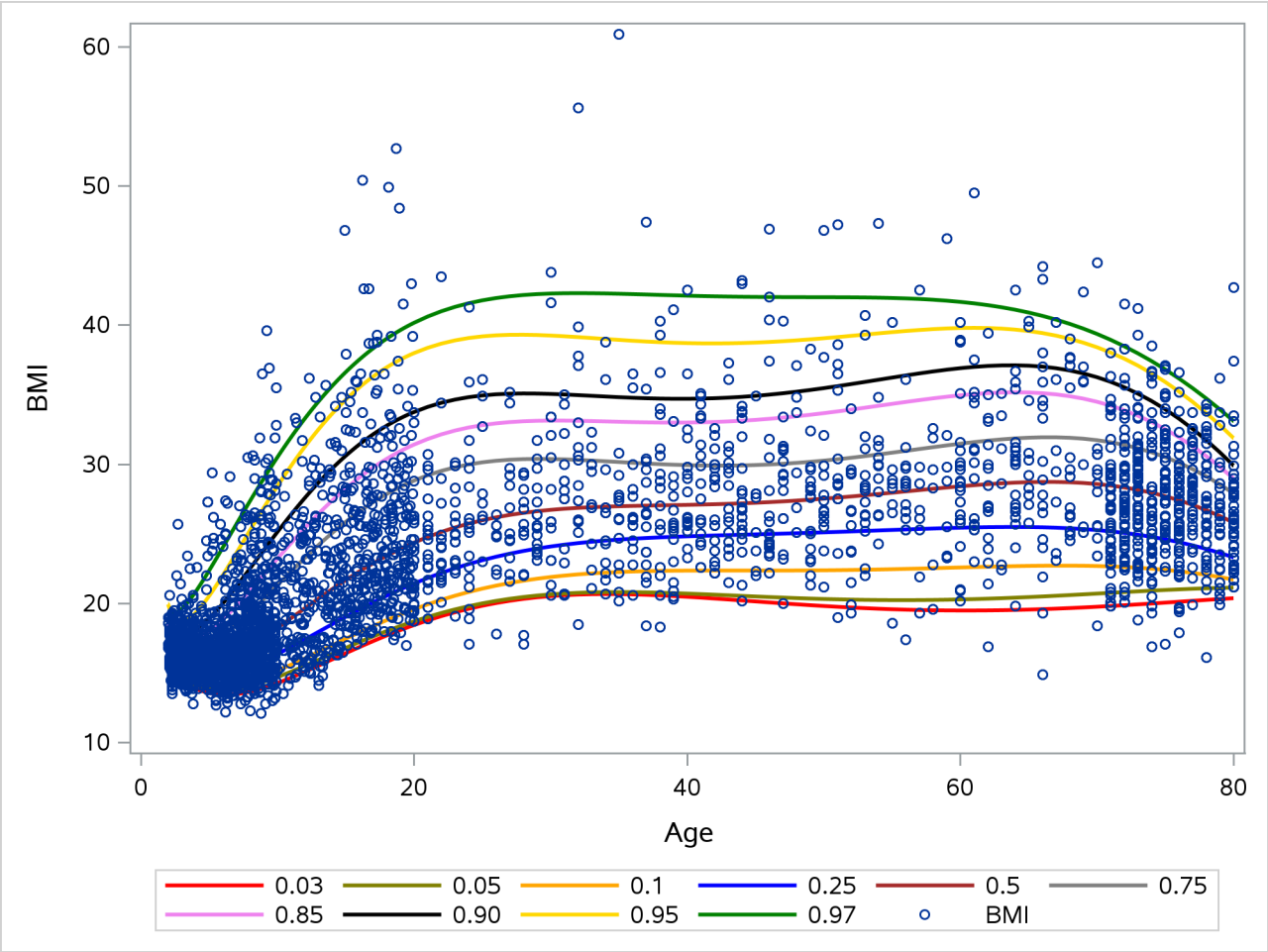

The HPQUANTSELECT procedure is a high-performance procedure that fits and performs effect selection for quantile regression analysis. PROC HPQUANTSELECT supports continuous variables, CLASS variables, and the interactions of these variables. PROC HPQUANTSELECT supports statistical inferences on quantile regression models with or without the assumption of independently and identically distributed (iid) errors. PROC HPQUANTSELECT also offers extensive capabilities for customizing the effect selection by using a wide variety of selection and stopping criteria.

PROC HPQUANTSELECT Features

The main features of the HPQUANTSELECT procedure are as follows:

- **Model specification**
  - supports quantile regression for single or multiple quantile levels
  - supports GLM and reference cell parameterization for classification effects
  - supports any degree of interaction (crossed effects) and nested effects
  - supports statistical inferences with or without iid errors assumption
  - supports hierarchy among effects
  - supports partitioning of data into training, validation, and testing roles
  - supports a CODE statement to write SAS DATA step code to a file or catalog entry for computing predicted quantiles
  - supports a WEIGHT statement for a weighted analysis
- **Selection control**
  - provides multiple effect-selection methods
  - offers selection of individual levels of classification effects
  - provides effect selection based on a variety of selection criteria
  - provides stopping rules based on a variety of model evaluation criteria
- **Display and output**
  - produces output data sets that contain predicted values, residuals, standardized errors, and confidence limits of predicted values

The HPQUANTSELECT procedure supports the following effect-selection methods. For more information about these methods, see the section "Methods" on page 582.

forward selection — starts with no effects in the model and adds effects.

backward elimination — starts with all effects in the model and deletes effects.

stepwise selection — is similar to the forward selection method except that effects already in the model do not necessarily stay there.

PROC HPQUANTSELECT Contrasted with Other SAS Procedures

For general contrasts between SAS High-Performance Analytics procedures and other SAS procedures, see the section "Common Features of SAS High-Performance Statistical Procedures" on page 572. The following remarks contrast the HPQUANTSELECT procedure with the QUANTSELECT and QUANTREG procedures in SAS/STAT.

The major functional differences between the HPQUANTSELECT and QUANTSELECT procedures are as follows:

- The HPQUANTSELECT procedure uses an interior point algorithm for model fitting. The QUANTSELECT procedure uses a flexible simplex algorithm for model fitting.
- The HPQUANTSELECT procedure can output confidence limits for parameter estimates.
- The HPQUANTSELECT procedure does not support the LASSO and adaptive LASSO effect-selection methods.
- The HPQUANTSELECT procedure does not support the TESTDATA= and VALDATA= options in its PROC statement.
- The HPQUANTSELECT procedure does not support graphical summaries for the effect selection processes.

Both the HPQUANTSELECT and QUANTSELECT procedures support the forward, backward, and stepwise effect-selection methods and the ability to use separate validation and test data via the PARTITION statement. For more information about the QUANTSELECT procedure, see Chapter 104, "The QUANTSELECT Procedure."

The major functional differences between the HPQUANTSELECT and QUANTREG procedures are as follows:

- The QUANTREG procedure does not support any effect-selection methods. It does not output the following fit statistics: AIC, AICC, SBS, VALIDATE, TEST, R1, and adjusted R1. And it does not support the PARTITION statement.
- The QUANTREG procedure provides three algorithms for fitting quantile regression models: simplex algorithm, interior point algorithm, and smoothing algorithm. The HPQUANTSELECT procedure supports only the interior point algorithm.
- The QUANTREG procedure supports the rank test, which is not available for the HPQUANTSELECT procedure. Both the QUANTREG and HPQUANTSELECT procedures support the Wald test and the likelihood test.
- The QUANTREG procedure supports two methods of estimating the covariance matrix of the parameter estimates: an asymptotic method and a bootstrap method. The HPQUANTSELECT procedure supports only the asymptotic method.

For more information about the QUANTREG procedure, see Chapter 103, "The QUANTREG Procedure."

The HPQUANTSELECT procedure is also different from the QUANTSELECT and QUANTREG procedures in the following respects:

- The HPQUANTSELECT procedure supports the CODE statement, which is not available in the QUANTSELECT and QUANTREG procedures.
- The HPQUANTSELECT procedure does not support quantile process regression, whereas the QUANTREG procedure does support quantile process regression. The QUANTSELECT procedure supports effect selection for quantile process regression.
- The HPQUANTSELECT procedure does not support the EFFECT statement, which provides constructed effects such as polynomial effects, spline effects, collection effects, and multimember classification effects.
- The HPQUANTSELECT procedure can output confidence limits for mean predicted quantiles; this functionality is not available in the QUANTSELECT and QUANTREG procedures.

In addition to having many similarities to the QUANTSELECT and QUANTREG procedures, the HPQUANTSELECT procedure also compares closely to the HPREG procedure. PROC HPREG is a high-performance procedure that performs effect selection in the framework of general linear models. The HPQUANTSELECT procedure inherits most of its syntax from the HPREG and QUANTREG procedures. The HPQUANTSELECT procedure provides results that are similar to those of the HPREG and QUANTSELECT procedures.

Getting Started: HPQUANTSELECT Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpqtrgs.sas`.)

The following example is modeled on the example in the section "Getting Started: QUANTSELECT Procedure" in the *SAS/STAT User's Guide*. The Sashelp.baseball data set contains salary and performance information for Major League Baseball (MLB) players, excluding pitchers, who played in at least one game in both the 1986 and 1987 seasons. The salaries (Time Inc. 1987) are for the 1987 season, and the performance measures are for the 1986 season (Reichler 1987).

The following statements display the variables in the data set. Figure 66.2 shows the results.

```
proc contents varnum data=sashelp.baseball;
   ods select position;
run;
```

Figure 66.2 Sashelp.Baseball Data Set

The CONTENTS Procedure

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	Name	Char	18	Player's Name
2	Team	Char	14	Team at the End of 1986
3	nAtBat	Num	8	Times at Bat in 1986
4	nHits	Num	8	Hits in 1986
5	nHome	Num	8	Home Runs in 1986
6	nRuns	Num	8	Runs in 1986
7	nRBI	Num	8	RBIs in 1986
8	nBB	Num	8	Walks in 1986
9	YrMajor	Num	8	Years in the Major Leagues
10	CrAtBat	Num	8	Career Times at Bat
11	CrHits	Num	8	Career Hits
12	CrHome	Num	8	Career Home Runs
13	CrRuns	Num	8	Career Runs
14	CrRbi	Num	8	Career RBIs
15	CrBB	Num	8	Career Walks
16	League	Char	8	League at the End of 1986
17	Division	Char	8	Division at the End of 1986
18	Position	Char	8	Position(s) in 1986
19	nOuts	Num	8	Put Outs in 1986
20	nAssts	Num	8	Assists in 1986
21	nError	Num	8	Errors in 1986
22	Salary	Num	8	1987 Salary in $ Thousands
23	Div	Char	16	League and Division
24	logSalary	Num	8	Log Salary

Suppose you want to investigate how the MLB players' salaries for the 1987 season depend on performance measures for the players' previous season and MLB career. You might worry that some players who are outliers could dominate your least squares analysis. To address this concern, you can use the following statements to obtain a median regression model, which is equivalent to the 50th conditional percentile or the quantile regression model at quantile level 0.5:

```
proc hpquantselect data=sashelp.baseball;
   class league division;
   model Salary = nAtBat nHits nHome nRuns nRBI nBB
                  yrMajor crAtBat crHits crHome crRuns crRbi
                  crBB league division nOuts nAssts nError
         / clb;
run;
```

The CLB option in the MODEL statement requests 95% confidence limits for the parameter estimates. If you do not use the SELECTION statement, the HPQUANTSELECT procedure fits the full model that is specified by the MODEL statement without any effect selection.

Figure 66.3 Performance, Data Access, and Model Information

The HPQUANTSELECT Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Data Access Information			
Data	**Engine**	**Role**	**Path**
SASHELP.BASEBALL	V9	Input	On Client

Model Information	
Data Source	SASHELP.BASEBALL
Dependent Variable	Salary
Class Parameterization	GLM
Optimization Algorithm	Interior

Figure 66.3 displays the "Performance Information," "Data Access Information," and "Model Information" tables.

The "Performance Information" table shows that the HPQUANTSELECT procedure executes in client mode—that is, the model is fit on the machine where the SAS session executes. This step was performed on a multicore machine that contained four CPUs; one computational thread was spawned per CPU.

The "Data Access Information" table shows that the input data set is accessed with the V9 (base) engine on the client machine.

The "Model Information" table identifies the data source and response and shows that the CLASS variables are parameterized in the GLM parameterization, which is the default.

Figure 66.4 Number of Observations, Class Level Information, and Dimensions Tables

Number of Observations Read	322
Number of Observations Used	263

Class Level Information		
Class	**Levels**	**Values**
League	2	American National
Division	2	East West

Dimensions	
Number of Effects	19
Number of Parameters	21

Figure 66.4 displays the "Number of Observations," "Class Level Information," and "Dimensions" tables.

The "Number of Observations" table shows that, of the 322 observations, PROC HPQUANTSELECT uses only 263 observations for model fitting and ignores 59 incomplete observations.

The "Class Level Information" table shows level information for two CLASS effects that the CLASS statement identifies: League and Division. League has two levels: American League and National League.

Division also has two levels: East Division and West Division.

The "Dimensions" table shows that the MODEL statement identifies 19 effects for model fitting besides the intercept effect. Because the 19 effects include two CLASS effects and each level of a CLASS effect corresponds to a parameter, the 19 effects contain a total of 21 parameters.

Figure 66.5 Fit Statistics

The HPQUANTSELECT Procedure

Quantile Level = 0.5

Fit Statistics	
Objective Function	25977
R1	0.40584
Adj R1	0.36200
AIC	2453.81587
AICC	2456.94344
SBC	2521.68680
ACL	98.77118

Figure 66.5 displays the "Fit Statistics" table, which shows the values of model fitting criteria for the fitted median model. For more information about model fitting criteria for quantile regression, see the section "Details: HPQUANTSELECT Procedure" on page 5105.

Figure 66.6 Parameter Estimates

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	-67.75322	39.95908	-146.46197	10.95553	-1.70	0.0912
nAtBat	1	-1.57112	0.44700	-2.45160	-0.69064	-3.51	0.0005
nHits	1	8.82192	1.94990	4.98114	12.66270	4.52	<.0001
nHome	1	-5.91757	4.91015	-15.58926	3.75412	-1.21	0.2293
nRuns	1	-5.17076	2.14914	-9.40400	-0.93753	-2.41	0.0169
nRBI	1	0.77547	2.15469	-3.46870	5.01963	0.36	0.7192
nBB	1	5.28866	1.67603	1.98732	8.59000	3.16	0.0018
YrMajor	1	6.61877	6.61798	-6.41690	19.65444	1.00	0.3182
CrAtBat	1	-0.04463	0.15485	-0.34964	0.26038	-0.29	0.7734
CrHits	1	0.07896	0.73594	-1.37064	1.52857	0.11	0.9146
CrHome	1	3.78231	1.90065	0.03854	7.52607	1.99	0.0477
CrRuns	1	1.23105	0.77137	-0.28833	2.75044	1.60	0.1118
CrRbi	1	-0.70695	0.76888	-2.22144	0.80754	-0.92	0.3588
CrBB	1	-0.68911	0.41382	-1.50423	0.12601	-1.67	0.0971
League American	1	-34.39136	24.37175	-82.39724	13.61451	-1.41	0.1595
League National	0	0	.	.	.	.	.
Division East	1	60.30856	27.28730	6.55984	114.05728	2.21	0.0280
Division West	0	0	.	.	.	.	.
nOuts	1	0.23273	0.12110	-0.00581	0.47126	1.92	0.0558
nAssts	1	0.09824	0.18888	-0.27381	0.47029	0.52	0.6035
nError	1	-0.81574	3.51436	-7.73810	6.10661	-0.23	0.8166

Figure 66.6 displays the "Parameter Estimates" table, which shows the parameter estimates of the fitted median model. You can see that, of the 19 effective parameters whose degrees of freedom are not zero, the fitted model contains 13 insignificant parameters whose 95% confidence intervals cover zeros. Because more than half of the 19 effective parameters are insignificant, you might worry that the model is overfitted.

It is well known that both overfitting and underfitting harm the prediction performance of a model. You can prevent overfitting and underfitting by using a good effect-selection technique. The following statements apply the forward selection method and the SL (significance level) criterion to choose a parsimonious model for the Sashelp.baseball data set:

```
proc hpquantselect data=sashelp.baseball;
   class league division;
   model Salary = nAtBat nHits nHome nRuns nRBI nBB
                  yrMajor crAtBat crHits crHome crRuns crRbi
                  crBB league division nOuts nAssts nError
         / clb;
   selection method=forward(select=sl sle=0.1);
run;
```

The SLE=0.1 option in the SELECTION statement specifies the significance level for entry. A candidate effect can enter the model at a certain selection step only if the following conditions are met:

- Its p-value is the smallest among all the valid candidate effects.
- Its p-value is smaller than 0.1 (the significance level for entry).

For more information about using significance levels in effect selection, see the section "Statistical Tests for Significance Level" on page 5110.

Figure 66.7 Selection Information

The HPQUANTSELECT Procedure

Selection Information	
Selection Method	Forward
Select Criterion	Significance Level
Stop Criterion	Significance Level
Effect Hierarchy Enforced	None
Entry Significance Level (SLE)	0.1
Stop Horizon	1

Figure 66.7 displays the "Selection Information" table. The "Selection Information" provides details about the method and criteria used to perform the model selection. The requested selection method is the forward selection method where the decisions about what effects to add at any step and when to terminate the selection are both based on the significance level criterion.

Figure 66.8 Selection Summary

The HPQUANTSELECT Procedure

Quantile Level = 0.5

Selection Summary			
Step	Effect Entered	Number Effects In	p Value
0	Intercept	1	.
1	CrHome	2	<.0001
2	nHits	3	<.0001
3	CrHits	4	<.0001
4	nOuts	5	0.0185
5	nAtBat	6	0.0182
6	Division	7	0.0118
7	nBB	8	0.0647
8	nRuns	9	0.0558

Figure 66.8 displays the "Selection Summary" table. Each row in the "Selection Summary" table shows the effect that enters the model at the corresponding step of the effect selection process together with its p-value for adding the effect into the model at that step.

Figure 66.9 Stopping and Selection Reasons

Selection stopped because no candidate for entry is significant at the 0.1 level.

The model at step 8 is selected.

The HPQUANTSELECT Procedure

Quantile Level = 0.5
Selected Model

Selected Effects: Intercept nAtBat nHits nRuns nBB CrHits CrHome Division nOuts

Figure 66.9 displays the "Stop Reason," "Selection Reason," and "Selected Effects" tables. The "Stop Reason" and "Selection Reason" tables indicate that effect selection stopped because no candidate for entry was significant at the 0.1 level after step 8. The "Selected Effects" table lists the effects that are included in the selected model.

Figure 66.10 Details of the Selected Model

Fit Statistics	
Objective Function	26568
R1	0.39232
Adj R1	0.37318
AIC	2445.64547
AICC	2446.35693
SBC	2477.79485
ACL	101.01768

Figure 66.10 *continued*

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	-130.65536	33.04546	-195.73336	-65.57735	-3.95	<.0001
nAtBat	1	-1.20522	0.41690	-2.02624	-0.38419	-2.89	0.0042
nHits	1	7.76667	1.81045	4.20127	11.33207	4.29	<.0001
nRuns	1	-3.92180	1.87567	-7.61566	-0.22795	-2.09	0.0375
nBB	1	3.92049	1.04341	1.86565	5.97532	3.76	0.0002
CrHits	1	0.17697	0.06856	0.04195	0.31198	2.58	0.0104
CrHome	1	1.66939	0.73083	0.23012	3.10866	2.28	0.0232
Division East	1	65.14327	24.48038	16.93289	113.35364	2.66	0.0083
Division West	0	0	.	.	.	.	.
nOuts	1	0.23719	0.11403	0.01261	0.46176	2.08	0.0385

The "Fit Statistics" and "Parameter Estimates" tables in Figure 66.10 give details of the final selected model. You can see that all nine effective parameters (excluding Division West) are significant at the 5% significance level, corresponding to the 95% confidence limits.

Like the sample median, a median regression model is robust to extreme observations, because it depends only on a small middle subset of all the observations in the data set. However, it is less representative of the entire conditional distribution of the response variable. You might want to further investigate the Sashelp.baseball data set at other quantile levels. The following statements select quantile regression models at the quantile levels 0.1 and 0.9, which correspond to the 10% and 90% conditional percentiles of the players' salaries:

```
proc hpquantselect data=sashelp.baseball alpha=0.1;
   class league division;
   model Salary = nAtBat nHits nHome nRuns nRBI nBB
                  yrMajor crAtBat crHits crHome crRuns crRbi
                  crBB league division nOuts nAssts nError
         / quantile=0.1 0.9 clb;
   selection method=backward(select=sl sls=0.1);
run;
```

The ALPHA=0.1 option in the PROC statement sets the significance level to 0.1. Combined with the CLB option in the MODEL statement, the ALPHA=0.1 option requests 90% confidence limits for parameter estimates. The QUANTILE= option in the MODEL statement specifies two quantile levels, 0.1 and 0.9, for fitting quantile regression models. The METHOD=BACKWARD option in the SELECTION statement specifies the backward elimination method of effect selection.

Figure 66.11 Parameter Estimates at Quantile Level 0.1

The HPQUANTSELECT Procedure

Quantile Level = 0.1
Selected Model

Selected Effects:	Intercept nAtBat nHits nBB CrRuns CrBB Division nAssts

Figure 66.11 *continued*

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	90% Confidence Limits		t Value	Pr > |t|
Intercept	1	4.75224	18.94983	-26.53111	36.03558	0.25	0.8022
nAtBat	1	-0.73670	0.17958	-1.03316	-0.44024	-4.10	<.0001
nHits	1	2.69396	0.63510	1.64551	3.74241	4.24	<.0001
nBB	1	1.81807	0.40595	1.14791	2.48823	4.48	<.0001
CrRuns	1	0.65476	0.09560	0.49694	0.81259	6.85	<.0001
CrBB	1	-0.44622	0.16254	-0.71454	-0.17790	-2.75	0.0065
Division East	1	28.35406	10.68922	10.70774	46.00038	2.65	0.0085
Division West	0	0	.	.	.	.	.
nAssts	1	0.14958	0.05897	0.05223	0.24694	2.54	0.0118

Figure 66.11 displays the "Selected Effects" and "Parameter Estimates" tables at quantile level 0.1.

Figure 66.12 Parameter Estimates at Quantile Level 0.9

The HPQUANTSELECT Procedure

Quantile Level = 0.9
Selected Model

Selected Effects:	Intercept nHits nBB CrAtBat CrHits CrHome CrRbi League Division nOuts

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	90% Confidence Limits		t Value	Pr > |t|
Intercept	1	20.39804	58.17164	-75.63745	116.43353	0.35	0.7261
nHits	1	2.30897	0.55640	1.39042	3.22752	4.15	<.0001
nBB	1	3.09799	1.44414	0.71386	5.48213	2.15	0.0329
CrAtBat	1	-0.44914	0.14651	-0.69101	-0.20727	-3.07	0.0024
CrHits	1	2.48064	0.51725	1.62672	3.33457	4.80	<.0001
CrHome	1	6.29896	1.37134	4.03502	8.56291	4.59	<.0001
CrRbi	1	-2.12293	0.76546	-3.38662	-0.85923	-2.77	0.0060
League American	1	-103.28955	33.68480	-158.89975	-47.67935	-3.07	0.0024
League National	0	0	.	.	.	.	.
Division East	1	107.46694	50.82797	23.55512	191.37876	2.11	0.0355
Division West	0	0	.	.	.	.	.
nOuts	1	0.39766	0.12820	0.18601	0.60931	3.10	0.0021

Figure 66.11 displays the "Selected Effects" and "Parameter Estimates" tables at quantile level 0.9.

You might want to compute the 90th percentile predictions for players' salaries and find out which players were overpaid based on the quantile regression model at quantile level 0.9. The following statements repeat the backward elimination method at quantile level 0.9, compute and sort the overpaid players' salaries, and output the observations for the top 10 overpaid players in the Sashelp.baseball data set:

```
proc hpquantselect data=sashelp.baseball alpha=0.1;
   id Name;
   class league division;
   model Salary = nAtBat nHits nHome nRuns nRBI nBB
                  yrMajor crAtBat crHits crHome crRuns crRbi
                  crBB league division nOuts nAssts nError
         / quantile=0.9 clb;
   selection method=backward(select=sl sls=0.1);
   output out=BaseballOverpaid copyvar=Salary r=Overpaid
          p=PredictedSalary lclm uclm;
run;

proc sort data=BaseballOverpaid;
   by descending Overpaid;
run;

proc print data=BaseballOverpaid(obs=10);
   var Name Salary Overpaid PredictedSalary lclm uclm;
run;
```

The ID statement adds the variable Name from the input data set to the BaseballOverpaid data set that is produced by the OUTPUT statement. The LCLM and UCLM options, respectively, request lower and upper bounds of $100(1-\alpha)\%$ confidence intervals for the expected conditional quantile predictions of players' salaries at quantile level 0.9.

Figure 66.13 Top 10 Overpaid Baseball Players at Quantile Level 0.9

Obs	Name	Salary	Overpaid	PredictedSalary	LCLM	UCLM
1	Smith, Ozzie	1940.0	1084.54	855.46	611.09	1099.83
2	Wiggins, Alan	700.0	220.74	479.26	345.93	612.60
3	Murray, Eddie	2460.0	213.10	2246.90	1982.06	2511.74
4	Strawberry, Darryl	1220.0	187.64	1032.36	914.99	1149.72
5	Gibson, Kirk	1300.0	177.24	1122.76	1011.97	1233.56
6	Trevino, Alex	512.5	149.70	362.80	273.17	452.43
7	Ramirez, Rafael	875.0	141.09	733.91	599.52	868.31
8	Romero, Ed	375.0	128.71	246.29	144.88	347.70
9	Mattingly, Don	1975.0	124.82	1850.18	1557.81	2142.55
10	Puhl, Terry	900.0	104.34	795.66	625.51	965.81

Output 66.13 shows the information about the top 10 overpaid players according to the final selected quantile regression model at quantile level 0.9. Ozzie Smith is in first place. This might be because, although Smith was known for his defensive brilliance, the model weights offensive performance measures much more than defensive performance measures.

Syntax: HPQUANTSELECT Procedure

The following statements are available in the HPQUANTSELECT procedure:

```
PROC HPQUANTSELECT <options> ;
    BY variables ;
    CLASS variable <(options)>... <variable <(options)>> </global-options> ;
    CODE <options> ;
    ID variables ;
    MODEL dependent = <effects> </model-options> ;
    OUTPUT <OUT=SAS-data-set>
            <keyword <=name>>...
            <keyword <=name>> </options> ;
    PARTITION <partition-options> ;
    PERFORMANCE performance-options ;
    SELECTION selection-options ;
    WEIGHT variable ;
```

The PROC HPQUANTSELECT statement and a single MODEL statement are required. All other statements are optional. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statement.

The rest of this section provides detailed syntax information about each of the preceding statements, beginning with the PROC HPQUANTSELECT statement. The remaining statements are described in alphabetical order.

PROC HPQUANTSELECT Statement

```
PROC HPQUANTSELECT <options> ;
```

The PROC HPQUANTSELECT statement invokes the HPQUANTSELECT procedure. Table 66.1 summarizes the options in the PROC HPQUANTSELECT statement by function.

Table 66.1 PROC HPQUANTSELECT Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
MAXMACRO=	Specifies the maximum number of macro variables to produce
NAMELEN=	Limits the length of effect names
Options Related to Output	
NOCLPRINT	Limits or suppresses the display of CLASS levels
NOPRINT	Suppresses ODS output
User-Defined Formats	
FMTLIBXML=	Specifies a file reference for a format stream

Table 66.1 *continued*

Option	Description
Other Options	
ALPHA=	Sets the significance level to use for the construction of confidence intervals
SEED=	Sets the seed used for pseudorandom number generation

The following list describes these *options* in alphabetical order:

ALPHA=*number*

sets the significance level to use for the construction of confidence intervals. The value must be between 0 and 1; the default value of 0.05 results in 95% intervals. This option affects the STDP, LCLM, and UCLM keywords in the OUTPUT statement and the CLB option in the MODEL statement.

DATA=*SAS-data-set*

names the input SAS data set to be used by PROC HPQUANTSELECT. The default is the most recently created data set.

FMTLIBXML=*file-ref*

specifies the file reference for the XML stream that contains the user-defined format definitions. For more information about how to generate an XML stream for your formats, see the section "Working with Formats" on page 568.

MAXMACRO=*n*

specifies the total maximum number of macro variables to produce. Each macro variable contains selected effects for a selected model. For more information about the macro variables, see the section "Macro Variables That Contain Selected Effects" on page 5113. By default, MAXMACRO=100.

NAMELEN=*number*

specifies the length to which long effect names are to be shortened. The default and minimum value is 20.

NOCLPRINT< =*number* **>**

suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying *number* helps reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOPRINT

suppresses the generation of ODS output.

SEED=*number*

specifies an integer to be used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify a seed, or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer's clock.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPQUANTSELECT to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPQUANTSELECT procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(** *options* **)** >. . . < *variable* < **(** *options* **)** > > < / *global-option* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. The CLASS statement must precede the MODEL statement.

The CLASS statement for SAS high-performance analytical procedures is documented in the section "CLASS Statement" on page 572. The HPQUANTSELECT procedure also supports the following *global-option* in the CLASS statement:

UPCASE

uppercases the values of character-valued CLASS variables before levelizing them. For example, if you specify the UPCASE option and a CLASS variable can take the values "a," "A," and "b," then "a" and "A" represent the same level and the CLASS variable is treated as having only two values: "A" and "B."

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 66.2 summarizes the *options* available in the CODE statement.

Table 66.2 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are transferred to output data sets created by SAS High-Performance Analytics procedures, provided that the output data set produces one (or more) records per input observation.

For information about the common ID statement in SAS high-performance analytical procedures, see the section "ID Statement" on page 576.

MODEL Statement

MODEL *dependent*=< *effects* > / < *options* > **;**

The MODEL statement names the dependent variable and the explanatory effects, including covariates, main effects, interactions, and nested effects. If you omit the explanatory effects, PROC HPQUANTSELECT fits an intercept-only model.

After the keyword MODEL, the dependent (response) variable is specified, followed by an equal sign. The explanatory effects follow the equal sign. For information about constructing the model effects, see the section "Parameterization of Model Effects" on page 412.

You can specify the following *options* in the MODEL statement after a slash (/):

CLB

requests the $100(1-\alpha)\%$ upper and lower confidence limits for the parameter estimates. By default, the 95% limits are computed; you can use the ALPHA= option in the PROC HPQUANTSELECT statement to change the α level.

INCLUDE=*n*

INCLUDE=*single-effect*

INCLUDE=(*effects***)**

forces effects to be included in all models. If you specify INCLUDE=*n*, then the first *n* effects that are listed in the MODEL statement are included in all models. If you specify INCLUDE=*single-effect* or if you specify a list of effects within parentheses, then the specified effects are forced into all models. The effects that you specify in the INCLUDE= option must be explanatory effects that are defined in the MODEL statement.

NOINT

suppresses the intercept term that is otherwise included in the model.

ORDERSELECT

specifies that, for the selected model, effects be displayed in the order in which they first entered the model. If you do not specify this option, then effects in the selected model are displayed in the order in which they appear in the MODEL statement.

QUANTILES=*number-list*

QUANTILE=*number-list*

specifies the quantile levels for the quantile regression. You can specify any number of quantile levels in $(0, 1)$. If you do not specify this option, the HPQUANTSELECT procedure performs median regression effect selection that corresponds to QUANTILE=0.5.

SPARSITY(< BF | HS > < IID >)

specifies the suboptions for estimating the sparsity function. You can specify the Bofinger method by using the BF suboption or the Hall-Sheather method by using the HS suboption. By default, the Hall-Sheather method is used. You can also specify the IID suboption to assume that the quantile regression errors satisfy the independently and identically distributed (iid) assumption. Let f_i and F_i, respectively, denote the probability density function and the cumulative distribution function of the ith error for $i = 1, \ldots, n$. The iid assumption means that there exist f and F such that $f = f_1 = \cdots = f_n$ and $F = F_1 = \cdots = F_n$. If you specify the IID option, the covariance matrix of the parameter estimates,

$\omega^2(\tau, F)(\mathbf{X}'\mathbf{X})^-$, is adopted for computing the confidence limits and the Wald statistics, where $\omega^2(\tau, F) = \tau(1-\tau)/f^2(F^{-1}(\tau))$. By default, the covariance matrix of the parameter estimates is non-iid and takes the sandwich form: $n^{-2}\tau(1-\tau)\mathbf{H}_n^-(\mathbf{X}'\mathbf{X})\mathbf{H}_n^-$, where $\mathbf{H}_n = n^{-1}\sum_{i=1}^n f_i(F_i^{-1}(\tau))\mathbf{x}_i\mathbf{x}_i'$. For more information, see the section "Details: HPQUANTSELECT Procedure" on page 5105.

START=*n*

START=*single-effect*

START=(*effects***)**

begins the effect-selection process in the forward and stepwise selection methods from the initial model that you designate. If you specify START=*n*, then the starting model consists of the first *n* effects listed in the MODEL statement. If you specify START=*single-effect* or if you specify a list of effects within parentheses, then the starting model consists of these specified effects. The effects that you specify in the START= option must be explanatory effects defined in the MODEL statement. The START= option is not available when you specify METHOD=BACKWARD in the SELECTION statement.

STB

produces standardized regression coefficients. A standardized regression coefficient is computed by dividing a parameter estimate by the ratio of the sample standard deviation of the dependent variable to the sample standard deviation of the regressor. If you use the INCLUDE= option to force some effects to be in the model, then the QTRSELECT procedure computes the sample standard deviation against all the effects that are forced in, as follows. Let $\mathbf{X}_1$ denote the design submatrix of p_1 regressors that consists of all the effects that are forced in, and let $\mathbf{z}$ denote the dependent variable or any regressor. Then the sample standard deviation of $\mathbf{z}$ is computed as

$$s_z = \sqrt{\frac{\mathbf{z}'\left[\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1{}'\mathbf{X}_1)^{-1}\mathbf{X}_1{}'\right]\mathbf{z}}{n - p_1}}$$

TOL

produces tolerance values for the estimates. Tolerance for a parameter is defined as $1 - R^2$, where R^2 is obtained from the ordinary least squares regression of the parameter on all other parameters in the model.

VIF

produces variance inflation factors in the parameter estimates table. Variance inflation is the reciprocal of tolerance.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< **COPYVARS=(***variables***)** >
< *keyword* < **=***name* > >. . . < *keyword* < **=***name* > > **;**

The OUTPUT statement creates a data set that contains observationwise statistics, which are computed after the final selected model is fit. To avoid data duplication for large data sets, the variables in the input data set are *not* included in the output data set; however, variables that are specified in the ID statement or COPYVARS= option are included.

The output statistics are computed based on the parameter estimates for the selected model. If you specify multiple quantile levels by using the QUANTILE= option in the MODEL statement, then for each appropriate keyword that is specified in the OUTPUT statement, one variable is generated for each specified quantile level. These variables appear in the sorted order of the specified quantile levels. For example, the following statements generate the Out data set, which contains the two predicted quantile variables p1 and p2:

```
proc hpquantselect data=one;
   model y = x1-x4 /quantile=0.5 0.3;
   output out=out pred=p;
run;
```

The variable p1 is for quantile level 0.3, and the variable p2 is for quantile level 0.5, because the sorted quantile levels are (0.3 0.5), not (0.5 0.3).

You can specify the following options in the OUTPUT statement:

OUT=*SAS-data-set*

DATA=*SAS-data-set*

specifies the name of the output data set. If you omit the OUT= (or DATA=) option, PROC HPQUANTSELECT uses the DATA*n* convention to name the output data set.

COPYVAR=*variable*

COPYVARS=(*variables***)**

transfers one or more *variables* from the input data set to the output data set. Variables that you name in an ID statement are also copied from the input data set to the output data set.

keyword **< =***name* **>**

specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), followed optionally by an equal sign and a variable to contain the statistic.

If you specify *keyword***=***name*, the new variable that contains the requested statistic has the specified name. If you omit the optional **=***name* after a *keyword*, then a default name is used.

You can specify the following *keywords* to request statistics that are available with for selection methods:

LCLM

requests the lower bound of a $100(1-\alpha)\%$ confidence interval for the expected quantile of the dependent variable. The default variable name is LCLM.

PREDICTED

PRED

P

requests predicted values for the response variable. The default variable name is Pred.

RESIDUAL

RESID

R

requests the residual, calculated as ACTUAL – PREDICTED. The default variable name is Residual.

ROLE

requests a numeric variable that indicates the role that each observation plays in fitting the model. The default variable name is _ROLE_. For each observation, the interpretation of this variable is shown in Table 66.3.

Table 66.3 Role Interpretation

Value	Observation Role
0	Not used
1	Training
2	Validation
3	Testing

If you do not partition the input data by using a PARTITION statement, then the role variable value is 1 for observations that are used in fitting the model, and 0 for observations that have at least one missing or invalid value for the response, regressor, or weight variables.

You can use the following statements to display the values of the _ROLE_ variable in the output data set:

```
proc format;
   value role 0 = 'Not Used'
              1 = 'Training'
              2 = 'Validation'
              3 = 'Testing';
run;

proc freq;
   tables _role_;
   format _role_ role.;
run;
```

STDP
requests standard error of the mean predicted quantiles. The default variable name is STDP.

UCLM
requests the upper bound of a $100(1-\alpha)\%$ confidence interval for the expected quantile of the dependent variable. The default variable name is UCLM.

PARTITION Statement

PARTITION *partition-options* **;**

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. Either you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations for each role.

You can specify either of the following mutually exclusive *partition-options*:

FRACTION(< TEST=*fraction* **> < VALIDATE=***fraction* **>)**
randomly assigns the specified proportions of observations in the input data set to testing, validation, and training roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role.

ROLEVAR | ROLE=*variable***(< TEST='***value***' > < TRAIN='***value***' > < VALIDATE='***value***' >)**
names the variable in the input data set whose values are used to assign roles to each observation. Use the TEST=, TRAIN=, and VALIDATE= suboptions to specify the formatted values of this variable that are used to assign observation roles are specified in the TEST=, TRAIN=, and VALIDATE= suboptions. If you do not specify the TRAIN= suboption, then all observations whose roles are not determined by the TEST= and VALIDATE= suboptions are assigned to training.

To create an output data set variable that indicates the role assignment for either *partition-option*, specify the ROLE=*variable* option in the OUTPUT statement.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of a SAS high-performance analytical procedure.

The PERFORMANCE statement for SAS high-performance analytical procedures is documented in the section "PERFORMANCE Statement" on page 569.

SELECTION Statement

SELECTION < *options* > **;**

The SELECTION statement performs variable selection. The statement is fully documented in the section "SELECTION Statement" on page 576.

The HPQUANTSELECT procedure supports the following suboptions in the METHOD= option in the SELECTION statement to specify the corresponding effect selection methods:

NONE specifies no model selection.

FORWARD specifies the forward selection method, which starts with no effects in the model and adds effects.

BACKWARD specifies the backward elimination method, which starts with all effects in the model and deletes effects.

STEPWISE specifies the stepwise regression method, which is similar to the forward selection method except that effects already in the model do not necessarily stay there.

By default, the METHOD=STEPWISE option is used in the SELECTION statement. If you do not use the SELECTION statement, the HPQUANTSELECT procedure fits the full model that is specified by the MODEL statement; this is equivalent to specifying the METHOD=NONE option in the SELECTION statement. For information about all the selection criteria that are used in PROC HPQUANTSELECT, see the section "Criteria Used in Model Selection" on page 5108.

The DETAILS=ALL and DETAILS=STEPS options produce "Fit Statistics" and "Parameter Estimates" tables, which provide information about the model that is selected at each step of the selection process.

SHOWLASTSTEP
SLS

displays the last stop horizon step in the "Selection Summary" table, and includes this step for choosing the final model.

TEST=*value*

specifies a method to compute significance levels for the selection process. Table 66.4 summarizes these methods.

Table 66.4 Options for Significance Levels

Value of TEST=	Method
LR1	Likelihood ratio test Type I
LR2	Likelihood ratio test Type II
WALD	Wald score test

By default, PROC HPQUANTSELECT uses the Wald score to compute significance levels. If you specify the IID suboption in the SPARSITY option of the MODEL statement, the Wald score test uses the iid form of the covariance matrix to compute the Wald score and the associated significance levels. Otherwise, the non-iid form of the covariance matrix is used. The sparsity functions for both Type I

and Type II likelihood ratio tests are estimated under the iid assumption no matter whether you specify the IID suboption.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations that have nonpositive or missing weights are not included in the analysis. If you do not use a WEIGHT statement, all observations that are used in the analysis are assigned a weight of 1.

The HPQUANTSELECT procedure mainly uses each valid weight as the scale factor of its relevant observation. Let $\mathbf{X}$ denote the design matrix before using weights, and let $\mathbf{W}$ denote the diagonal matrix whose diagonal elements are the weights. Then the HPQUANTSELECT procedure mainly uses the weighted design matrix $\mathbf{WX}$ for its computation. For example, the weighted version of $\mathbf{X}'\mathbf{X}$ is $\mathbf{X}'\mathbf{WWX}$ but not $\mathbf{X}'\mathbf{WX}$. The only exception is in computing the standardized parameter estimates, which computation uses the weighted standard deviation of the dependent variable and all regressors. These standard deviations are based effectively on the diagonal of $\mathbf{X}'\mathbf{WX}$.

Details: HPQUANTSELECT Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpqtrdt.sas`.)

Quantile Regression

This section describes the basic concepts and notations for quantile regression and quantile regression model selection.

Let $\{(y_i, \mathbf{x}_i) : i = 1, \ldots, n\}$ denote a data set of observations, where y_i are responses and $\mathbf{x}_i$ are regressors. Koenker and Bassett (1978) define the *regression quantile* at quantile level $\tau \in (0, 1)$ as any solution to the minimization problem

$$\min_{\boldsymbol{\beta} \in \mathbf{R}^p} \sum_{i=1}^{n} \rho_\tau \left(y_i - \mathbf{x}_i' \boldsymbol{\beta} \right)$$

where $\rho_\tau(r) = \tau r^+ + (1 - \tau) r^-$ is a check loss function in which $r^+ = \max(r, 0)$ and $r^- = \max(-r, 0)$.

If you specify weights $w_i, i = 1, \ldots, n$, in the WEIGHT statement, then weighted quantile regression is carried out by solving

$$\min_{\boldsymbol{\beta} \in \mathbf{R}^p} \sum_{i=1}^{n} \rho_\tau \left(w_i (y_i - \mathbf{x}_i' \boldsymbol{\beta}) \right)$$

The HPQUANTSELECT procedure fits a quantile regression model by using a predictor-corrector interior point algorithm, which was originally designed to solve support vector machine classifiers for large data sets (Gertz and Griffin 2005, 2010).

Linear Model with iid Errors

You can specify the SPARSITY(IID) option in the MODEL statement to assume that the distribution of Y_i conditional on $\mathbf{x}_i$ follows the linear model

$$Y_i = \mathbf{x}_i'\boldsymbol{\beta} + \epsilon_i$$

where ϵ_i for $i = 1, \ldots, n$ are iid in the distribution function F. Let $f = F'$ denote the density function of F. Further assume that $f(F^{-1}(\tau)) > 0$ in a neighborhood of τ. Then, under some mild conditions, Koenker and Bassett (1982) prove that the asymptotic distribution of the quantile regression estimates is

$$\sqrt{n}(\hat{\boldsymbol{\beta}}(\tau) - \boldsymbol{\beta}(\tau)) \rightarrow N(0, \omega^2(\tau, F)\boldsymbol{\Omega}^{-1})$$

where $\omega^2(\tau, F) = \tau(1-\tau)/f^2(F^{-1}(\tau))$ and $\boldsymbol{\Omega} = \lim_{n\rightarrow\infty} n^{-1}\sum \mathbf{x}_i\mathbf{x}_i'$. The reciprocal of the density function, $s(\tau) = 1/f(F^{-1}(\tau))$, is called the *sparsity function*.

Accordingly, the covariance matrix of $\hat{\boldsymbol{\beta}}(\tau)$ can be estimated as

$$\hat{\Sigma}(\tau) = \tau(1-\tau)\hat{s}^2(\tau)(\mathbf{X}'\mathbf{X})^-$$

where $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)'$ is the design matrix and $\hat{s}(\tau)$ is an estimate of $s(\tau)$. Under the iid assumption, the algorithm for computing $\hat{s}(\tau)$ is as follows:

1. Fit a quantile regression model and compute the residuals. Each residual $r_i = y_i - \mathbf{x}_i'\hat{\boldsymbol{\beta}}(\tau)$ can be viewed as an estimated realization of the corresponding error ϵ_i.
2. Compute the quantile level bandwidth h_n. The HPQUANTSELECT procedure provides two bandwidth methods:
   - The Bofinger bandwidth is an optimizer of mean squared error for standard density estimation:
     $$h_n = n^{-1/5}(4.5v^2(\tau))^{1/5}$$
   - The Hall-Sheather bandwidth is based on Edgeworth expansions for studentized quantiles,
     $$h_n = n^{-1/3}z_\alpha^{2/3}(1.5v(\tau))^{1/3}$$
     z_α satisfies $T(z_\alpha, df) = 1 - \alpha/2$ for the construction of $1-\alpha$ confidence intervals, where T is the cumulative distribution function for the t distribution and df is the residual degrees of freedom.

   The quantity

   $$v(\tau) = \frac{s(\tau)}{s^{(2)}(\tau)} = \frac{f^2}{2(f^{(1)}/f)^2 + [(f^{(1)}/f)^2 - f^{(2)}/f]}$$

   is not sensitive to f and can be estimated by assuming f is Gaussian as

   $$\hat{v}(\tau) = \frac{\exp(-q^2)}{2\pi(q^2+1)}$$

   where $q = \Phi^{-1}(\tau)$.

3. Compute residual quantiles $\hat{F}^{-1}(\tau_0)$ and $\hat{F}^{-1}(\tau_1)$ as follows:

   a) Set $\tau_0 = \max(0, \tau - h_n)$ and $\tau_1 = \min(1, \tau + h_n)$.

   b) Use the equation

$$\hat{F}^{-1}(t) = \begin{cases} r_{(1)} & \text{if } t \in [0, 0.5/n) \\ \frac{0.5+(nt-i)}{n} r_{(i+1)} + \frac{0.5-(nt-i)}{n} r_{(i)} & \text{if } t \in [(i-0.5)/n, (i+0.5)/n) \\ r_{(n)} & \text{if } t \in [(n-0.5)/n, 1] \end{cases}$$

   where $r_{(i)}$ is the ith smallest residual.

   c) If $\hat{F}^{-1}(\tau_0) = \hat{F}^{-1}(\tau_1)$, find i that satisfies $r_{(i)} < \hat{F}^{-1}(\tau_0)$ and $r_{(i+1)} \geq \hat{F}^{-1}(\tau_0)$. If such an i exists, reset $\tau_0 = (i - 0.5)/n$ so that $\hat{F}^{-1}(\tau_0) = r_{(i)}$. Also find j that satisfies $r_{(j)} > \hat{F}^{-1}(\tau_1)$ and $r_{(j-1)} \leq \hat{F}^{-1}(\tau_1)$. If such a j exists, reset $\tau_1 = (j - 0.5)/n$ so that $\hat{F}^{-1}(\tau_1) = r_{(j)}$.

4. Estimate the sparsity function $s(\tau)$ as

$$\hat{s}(\tau) = \frac{\hat{F}^{-1}(\tau_1) - \hat{F}^{-1}(\tau_0)}{\tau_1 - \tau_0}$$

Linear-in-Parameter Model with Non-iid Settings

The general form of a linear quantile regression model is

$$Q_Y(\tau|\mathbf{x}) = \mathbf{x}'\boldsymbol{\beta}(\tau)$$

where the iid assumption is not necessary. Under some regularity conditions, the asymptotic distribution of the general form of quantile regression estimates is

$$\sqrt{n}(\hat{\boldsymbol{\beta}}(\tau) - \boldsymbol{\beta}(\tau)) \to N(0, \tau(1-\tau)\mathbf{H}_n^{-}\boldsymbol{\Omega}\mathbf{H}_n^{-})$$

where $\mathbf{H}_n = \lim_{n\to\infty} n^{-1} \sum \mathbf{x}_i \mathbf{x}_i' f_i(F_i^{-1}(\tau))$.

Accordingly, the covariance matrix of $\hat{\boldsymbol{\beta}}(\tau)$ can be estimated as

$$\hat{\Sigma}(\tau) = n^{-2}\tau(1-\tau)\hat{\mathbf{H}}_n^{-}(\mathbf{X}'\mathbf{X})\hat{\mathbf{H}}_n^{-}$$

where $\hat{\mathbf{H}}_n = n^{-1} \sum \left(\mathbf{x}_i \mathbf{x}_i' / \hat{s}_i(\tau)\right)$.

The sparsity function of the ith observation, $\hat{s}_i(\tau)$, can be estimated as

$$\hat{s}_i(\tau) = \frac{\hat{F}_i^{-1}(\tau + h_n) - \hat{F}_i^{-1}(\tau - h_n)}{2h_n}$$

where $\hat{F}_i^{-1}(\tau \pm h_n) = \mathbf{x}_i'\hat{\boldsymbol{\beta}}(\tau \pm h_n)$ are the ith predicted quantile values at quantile levels $(\tau \pm h_n)$.

More Statistics for Parameter Estimates

Let COV(·) denote the covariance function matrix of a random vector. Then, the sparsity-function estimates of $\mathrm{COV}\left(\hat{\boldsymbol{\beta}}(\tau)\right)$ is

$$\widehat{\mathrm{COV}}\left(\hat{\boldsymbol{\beta}}(\tau)\right) = \begin{cases} \omega^2(\tau, F)\boldsymbol{\Omega}^{-1}/n & \text{for a linear model with iid errors} \\ \tau(1-\tau)\mathbf{H}_n^{-}\boldsymbol{\Omega}\mathbf{H}_n^{-}/n & \text{for a linear-in-parameter model with non-iid settings} \end{cases}$$

where $\hat{\boldsymbol{\beta}}(\tau) = \left(\hat{\beta}_1(\tau), \ldots, \hat{\beta}_p(\tau)\right)$ is the vector of the parameter estimates.

If you specify the CLB option in the MODEL statement, PROC HPQUANTSELECT outputs the standard error, confidence limits, t value, and Pr > $|t|$ probability for each $\hat{\beta}_j(\tau)$ in the parameter estimates table. Table 66.5 summarizes these statistics for $\hat{\beta}_j(\tau)$.

Table 66.5 More Statistics for $\hat{\beta}_j(\tau)$

Statistic	Definition
Standard error: $\hat{\sigma}_j$	$\sqrt{\widehat{\mathrm{COV}}\left(\hat{\boldsymbol{\beta}}(\tau)\right)_{jj}}$
$(1-\alpha)\%$ confidence limits	$\left(\hat{\beta}_j(\tau) \pm t_{1,1-\frac{\alpha}{2}}\hat{\sigma}_j\right)$
t value	$\hat{\beta}_j(\tau)/\hat{\sigma}_j$
Pr > $\lvert t\rvert$ probability	p-value of the t value

Here $\mathrm{COV}\left(\hat{\boldsymbol{\beta}}(\tau)\right)_{jj}$ is the (j, j) element of $\mathrm{COV}\left(\hat{\boldsymbol{\beta}}(\tau)\right)$, and $t_{1,1-\frac{\alpha}{2}}$ denotes the $\left(1-\frac{\alpha}{2}\right)$-level student's t score with 1 degree of freedom.

Criteria Used in Model Selection

The HPQUANTSELECT procedure supports the following fit statistics that you can use as criteria for the CHOOSE=, SELECT=, and STOP= options in the SELECTION statement:

ADJR1	specifies the adjusted R1 statistic.
AIC	specifies Akaike's information criterion (Akaike 1969; Koenker 2005).
AICC	specifies the corrected Akaike's information criterion (Hurvich and Tsai 1989).
BIC | SBC	specifies the Schwarz Bayesian information criterion (Schwarz 1978; Koenker 2005).
R1	specifies the R1 statistic (Koenker and Machado 1999). The R1 statistic is not valid for the STOP= or CHOOSE= option.
SL	specifies the significance level that is used to assess an effect's contribution to the fit when it is added to or removed from a model. SL is not valid for the CHOOSE= option.
VALIDATE	specifies the average check loss over the validation data.

Quasi-likelihood Information Criteria

Given the quantile level τ, assume that the distribution of Y_i conditional on $\mathbf{x}_i$ follows the linear model

$$Y_i = \mathbf{x}_i'\boldsymbol{\beta} + \epsilon_i$$

where ϵ_i for $i = 1, \ldots, n$ are iid in distribution F. Further assume that F is an asymmetric Laplace distribution whose density function is

$$f_\tau(r) = \frac{\tau(1-\tau)}{\sigma} \exp\left(-\frac{\rho_\tau(r)}{\sigma}\right)$$

where σ is the scale parameter. Then, the negative log-likelihood function is

$$l_\tau(\boldsymbol{\beta}, \sigma) = n\log(\sigma) + \sigma^{-1}\sum_{i=1}^{n}\rho_\tau(y_i - \mathbf{x}_i'\boldsymbol{\beta}) - n\log(\tau(1-\tau))$$

Under these settings, the maximum likelihood estimate (MLE) of $\boldsymbol{\beta}$ is the same as the relevant level-τ quantile regression solution $\hat{\boldsymbol{\beta}}(\tau)$, and the MLE for σ is

$$\hat{\sigma}(\tau) = n^{-1}\sum_{i=1}^{n}\rho_\tau\left(y_i - \mathbf{x}_i'\hat{\boldsymbol{\beta}}(\tau)\right)$$

where $\hat{\sigma}(\tau)$ equals the level-τ average check loss $\mathrm{ACL}(\tau)$ for the quantile regression solution.

Because the general form of Akaike's information criterion (AIC) is $\mathrm{AIC} = (-2l + 2p)$, the quasi-likelihood AIC for quantile regression is

$$\mathrm{AIC}(\tau) = 2n\ln(\mathrm{ACL}(\tau)) + 2p$$

where p is the degrees of freedom for the fitted model.

Similarly, the quasi-likelihood AICC (corrected AIC) and SBC (Schwarz Bayesian information criterion) can be formulated as follows:

$$\mathrm{AICC}(\tau) = 2n\ln(\mathrm{ACL}(\tau)) + \frac{2pn}{n-p-1}$$

$$\mathrm{SBC}(\tau) = 2n\ln(\mathrm{ACL}(\tau)) + p\ln(n)$$

In fact, the quasi-likelihood AIC, AICC, and SBC are fairly robust, and you can use them to select effects for data sets without the iid assumption in asymmetric Laplace distribution. For a simulation study that applies SBC for effect selection, see "Example 66.1: Simulation Study" on page 5122. The study generates a data set by using a naive instrumental model (Chernozhukov and Hansen 2008).

Statistical Tests for Significance Level

The HPQUANTSELECT procedure supports the significance level (SL) criterion for effect selection. Consider the general form of a linear quantile regression model:

$$Q_Y(\tau|\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1'\boldsymbol{\beta}_1(\tau) + \mathbf{x}_2'\boldsymbol{\beta}_2(\tau)$$

At each step of an effect-selection process, a candidate effect can be represented as $\mathbf{x}_2$, and the significance level of the candidate effect can be calculated by testing the null hypothesis: $H_0 : \boldsymbol{\beta}_2(\tau) = \mathbf{0}$.

When you use SL as a criterion for effect selection, you can further use the TEST= option in the SELECTION statement to specify a statistical test method to compute the significance-level values as follows:

- The TEST=WALD option specifies the Wald test. Let $\hat{\boldsymbol{\beta}}(\tau) = \left(\hat{\boldsymbol{\beta}}_1'(\tau), \hat{\boldsymbol{\beta}}_2'(\tau)\right)'$ be the parameter estimates for the extended model, and denote the estimated covariance matrix of $\hat{\boldsymbol{\beta}}(\tau)$ as

  $$\hat{\Sigma}(\tau) = \begin{bmatrix} \hat{\Sigma}_{11}(\tau) & \hat{\Sigma}_{12}(\tau) \\ \hat{\Sigma}_{21}(\tau) & \hat{\Sigma}_{22}(\tau) \end{bmatrix}$$

  where $\hat{\Sigma}_{22}(\tau)$ is the covariance matrix for $\hat{\boldsymbol{\beta}}_2(\tau)$. Then the Wald test score is defined as

  $$\hat{\boldsymbol{\beta}}_2'(\tau)\hat{\Sigma}_{22}^{-1}(\tau)\hat{\boldsymbol{\beta}}_2(\tau)$$

  If you specify the SPARSITY(IID) option in the MODEL statement, $\hat{\Sigma}(\tau)$ is estimated under the iid errors assumption. Otherwise, $\hat{\Sigma}(\tau)$ is estimated by using non-iid settings. For more information about the linear model with iid errors and non-iid settings, see the section "Quantile Regression" on page 5105.

- The TEST=LR1 or TEST=LR2 option specifies the Type I or Type II quasi-likelihood ratio test, respectively. Under the iid assumption, Koenker and Machado (1999) propose two types of quasi-likelihood ratio tests for quantile regression, where the error distribution is flexible but not limited to the asymmetric Laplace distribution. The Type I test score, LR1, is defined as

  $$\frac{2(D_1(\tau) - D_2(\tau))}{\tau(1-\tau)\hat{s}}$$

  where $D_1(\tau) = \sum \rho_\tau\left(y_i - \mathbf{x}_{1i}\hat{\boldsymbol{\beta}}_{1_1}(\tau)\right)$ is the sum of check losses for the reduced model, $D_2(\tau) = \sum \rho_\tau\left(y_i - \mathbf{x}_{1i}\hat{\boldsymbol{\beta}}_{1_2}(\tau) - \mathbf{x}_{2i}\hat{\boldsymbol{\beta}}_2(\tau)\right)$ is the sum of check losses for the extended model, and $\hat{s}$ is the estimated sparsity function. The Type II test score, LR2, is defined as

  $$\frac{2D_2(\tau)\left(\log(D_1(\tau)) - \log(D_2(\tau))\right)}{\tau(1-\tau)\hat{s}}$$

Under the null hypothesis that the reduced model is the true model, the Wald score, LR1 score, and LR2 score all follow a χ^2 distribution with degrees of freedom $df = df_2 - df_1$, where df_1 and df_2 are the degrees of freedom for the reduced model and the extended model, respectively .

When you use SL as a criterion for effect selection, the algorithm for estimating sparsity function depends on whether an effect is being considered as an add or a drop candidate. For testing an add candidate effect, the

sparsity function, which is $s(\tau)$ under the iid error assumption or $s_i(\tau)$ for non-iid settings, is estimated on the reduced model that does not include the add candidate effect. For testing a drop candidate effect, the sparsity function is estimated on the extended model that does not exclude the drop candidate effect. Then, these estimated sparsity function values are used to compute LR1 or LR2 and the covariance matrix of the parameter estimates for the extended model. However, for the model that is selected at each step, the sparsity function for estimating standard errors and confidence limits of the parameter estimates is estimated on that model itself, but not on the model that was selected at the preceding step.

Because the null hypotheses usually do not hold, the SLENTRY and SLSTAY values cannot reliably be viewed as probabilities. One way to address this difficulty is to replace hypothesis testing as a means of selecting a model with information criteria or out-of-sample prediction criteria.

Table 66.6 provides formulas and definitions for these fit statistics.

Table 66.6 Formulas and Definitions for Model Fit Summary Statistics for Single Quantile Effect Selection

Statistic	Definition or Formula
n	Number of observations
p	Number of parameters, including the intercept
$r_i(\tau)$	Residual for the ith observation; $r_i(\tau) = y_i - \mathbf{x}_i\hat{\boldsymbol{\beta}}(\tau)$
$D(\tau)$	Total sum of check losses; $D(\tau) = \sum_{i=1}^{n} \rho_\tau(r_i)$. $D(\tau)$ is labeled as Objective Function in the "Fit Statistics" table.
$D_0(\tau)$	Total sum of check losses for intercept-only model if the intercept is a forced-in effect; otherwise for empty model.
$\mathrm{ACL}(\tau)$	Average check loss; $\mathrm{ACL}(\tau) = \dfrac{D(\tau)}{n}$
$\mathrm{R1}(\tau)$	Counterpart of linear regression R square for quantile regression; $\mathrm{R1}(\tau) = 1 - \dfrac{D(\tau)}{D_0(\tau)}$
$\mathrm{ADJR1}(\tau)$	Adjusted R1; $1 - \dfrac{(n-1)D(\tau)}{(n-p)D_0(\tau)}$ if intercept is a forced-in effect; otherwise $1 - \dfrac{nD(\tau)}{(n-p)D_0(\tau)}$.
$\mathrm{AIC}(\tau)$	$2n\ln(\mathrm{ACL}(\tau)) + 2p$
$\mathrm{AICC}(\tau)$	$2n\ln(\mathrm{ACL}(\tau)) + \dfrac{2pn}{n-p-1}$
$\mathrm{SBC}(\tau)$	$2n\ln(\mathrm{ACL}(\tau)) + p\ln(n)$

The $\mathrm{ADJR1}(\tau)$ criterion is equivalent to the generalized approximate cross validation (GACV) criterion for quantile regression (Yuan 2006). The GACV criterion is defined as

$$\mathrm{GACV}(\tau) = D(\tau)/(n-p)$$

which is proportional to $1 - \mathrm{ADJR1}(\tau)$.

Diagnostic Statistics

This section gathers the formulas for the statistics available in the OUTPUT statement. All the statistics available in the OUTPUT statement are conditional on the selected model and do not take into account the variability that is introduced by doing model selection.

The model to be fit is $Q_Y(\tau|\mathbf{x}) = \mathbf{x}'\boldsymbol{\beta}(\tau)$, and the parameter estimate $\hat{\boldsymbol{\beta}}(\tau)$ is the solution that minimizes $\sum_{i=1}^{n} \rho_\tau \left(y_i - \mathbf{x}_i'\boldsymbol{\beta}\right)$. The subscript i is for the ith observation. The subscript j is for the jth-smallest quantile level among all the specified QUANTILE= levels in the MODEL statement. $\hat{\Sigma}(\tau)$ denotes the covariance estimation for $\hat{\boldsymbol{\beta}}(\tau)$.

The ALPHA= option in the PROC HPQUANTSELECT statement sets the α value for the confidence limit statistics. The degrees of freedom for $t_{\frac{\alpha}{2}}$ are $n - p$.

Table 66.7 contains the diagnostic statistics and their formulas. Each statistic is computed for each observation.

Table 66.7 Formulas and Definitions for Diagnostic Statistics

MODEL Option or Statistic	Formula
PREDj	$\widehat{y}_{ji} = \mathbf{x}_i'\hat{\boldsymbol{\beta}}(\tau_j)$
RESj	$y_i - \widehat{y}_{ji}$
STDPj	$\sqrt{\mathbf{x}_i'\hat{\Sigma}(\tau_j)\mathbf{x}_i}$
LCLMj	$\widehat{y}_{ji} - t_{\frac{\alpha}{2}}\mathrm{STDP}j_i$
UCLMj	$\widehat{y}_{ji} + t_{\frac{\alpha}{2}}\mathrm{STDP}j_i$

Classification Variables and the SPLIT Option

PROC HPQUANTSELECT supports the ability to split classification variables when you do model selection. You use the SPLIT option in the CLASS statement to specify that the columns of the design matrix that correspond to effects that contain a split classification variable can enter or leave a model independently of the other design columns of that effect. The following statements illustrate the use of the SPLIT option:

```
data splitExample;
   length C2 $6;
   drop i;
   do i=1 to 1000;
     C1 = 1 + mod(i,6);
     if      i < 250 then C2 = 'Low';
     else if i < 500 then C2 = 'Medium';
     else                 C2 = 'High';
     x1 = ranuni(1);
     x2 = ranuni(1);
     y = x1+3*(C2 ='low')  + 10*(C1=3) +5*(C1=5) + rannor(1);
     output;
   end;
run;
```

```
proc hpquantselect data=splitExample;
   class C1(split) C2(order=data);
   model y = C1 C2 x1 x2/orderselect clb;
   selection method=forward;
run;
```

The "Class Levels" table in Figure 66.14 is produced by default whenever you specify a CLASS statement.

Figure 66.14 Class Levels

The HPQUANTSELECT Procedure

Class Level Information		
Class	**Levels**	**Values**
C1	6	* 1 2 3 4 5 6
C2	3	Low Medium High

*** Associated Parameters Split**

The SPLIT option has been specified for the classification variable C1. This permits the parameters that are associated with the effect C1 to enter or leave the model individually. The "Parameter Estimates" table in Figure 66.15 shows that for this example the parameters that correspond to only levels 3 and 5 of C1 are in the selected model. Finally, note that the ORDERSELECT option in the MODEL statement displays the parameters in the order in which they first entered the model.

Figure 66.15 Parameter Estimates

Parameter Estimates							
Parameter	**DF**	**Estimate**	**Standard Error**	**95% Confidence Limits**		**t Value**	**Pr > |t|**
Intercept	1	-0.21596	0.09024	-0.39304	-0.03887	-2.39	0.0169
C1_3	1	10.08952	0.09852	9.89619	10.28285	102.41	<.0001
C1_5	1	5.04115	0.10835	4.82854	5.25376	46.53	<.0001
x1	1	1.29863	0.14014	1.02363	1.57363	9.27	<.0001

Macro Variables That Contain Selected Effects

PROC HPQUANTSELECT saves the list of selected effects in a macro variable for each selected model so that you can use other SAS procedures to perform postselection analyses. This list does not explicitly include the intercept so that you can use it in the MODEL statement of other SAS/STAT regression procedures.

When multiple quantile levels or BY processing are used, one macro variable, indexed by the quantile level order and the BY group number (as shown in Table 66.8), is created for each quantile level and BY group combination.

Table 66.8 Macro Variables Created for Subsequent Processing

Macro Variable	Description
Single Quantile Level and No BY Processing	
_HPQRSIND	Selected model
Multiple Quantile Levels and No BY Processing	
_HPQRSINDT1	Selected model for the first quantile level
_HPQRSINDT2	Selected model for the second quantile level
...	
Single Quantile Level and BY Processing	
_HPQRSIND1	Selected model for BY group 1
_HPQRSIND2	Selected model for BY group 2
...	
Multiple Quantile Levels and BY Processing	
_HPQRSIND1T1	Selected model for the first quantile level and BY group 1
_HPQRSIND1T2	Selected model for the second quantile level and BY group 1
...	
_HPQRSIND2T1	Selected model for the first quantile level and BY group 2
_HPQRSIND2T2	Selected model for the second quantile level and BY group 2
...	

The macro variables _HPQRSIND, _HPQRSINDT1, _HPQRSIND1, and _HPQRSIND1T1 are all synonyms.

The following statements generate a simulation data set, use PROC HPQUANTSELECT to select a median regression model, and print the macro variables for the selected model:

```
%let seed=321;
%let p=20;
%let n=3000;

data analysisData;
   array x{&p} x1-x&p;
   do i=1 to &n;
      do j=1 to &p;
         x{j} = ranuni(&seed);
      end;
      e  = ranuni(&seed);
      y  = x1 + x2 + x3 + e;
      output;
   end;
run;

proc hpquantselect data=analysisData;
   model y = x1-x&p;
   selection method=forward;
run;

%put _HPQRSIND    = &_hpqrsind;
%put _HPQRSIND1   = &_hpqrsind1;
%put _HPQRSIND1T1 = &_hpqrsind1t1;
```

The following statements use PROC HPREG to fit a linear regression model on the effects that are selected by the HPQUANTSELECT procedure:

```
proc hpreg data=analysisData;
   model y = &_hpqrsind;
run;
```

Using Validation and Test Data

When you have sufficient data, you can subdivide your data into three parts, called the training, validation, and test data. During the selection process, models are fit on the training data, and the prediction error for the models is found by using the validation data. This prediction error on the validation data can be used to decide when to terminate the selection process or what effects to include as the selection process proceeds. Finally, after you have obtained a selected model, you can use the test set to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases, you might want to use only training and test data. For example, you might decide to use an information criterion to decide what effects to include and when to terminate the selection process. In this case, no validation data are required, but test data can still be useful in assessing the predictive performance of the selected model. In other cases, you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) state that it is difficult to give a general rule for how many observations you should assign to each role. They note that a typical split might be 50% for training and 25% each for validation and testing.

You use a PARTITION statement to logically subdivide the DATA= data set into separate roles. You can name the fractions of the data that you want to reserve as test data and validation data. For example, the following statements randomly subdivide the inData data set, reserving 50% for training and 25% each for validation and testing:

```
proc hpquantselect data=inData;
  partition fraction(test=0.25 validate=0.25);
  ...
run;
```

In some cases, you might need to exercise more control over the partitioning of the input data set. You can do this by naming both a variable in the input data set and a formatted value of that variable that correspond to each role. For example, the following statements assign roles to the observations in the inData data set based on the value of the variable Group in that data set. Observations in which the value of Group is `'group 1'` are assigned for testing, and those whose value is `'group 2'` are assigned to training. All other observations are ignored.

```
proc hpquantselect data=inData;
  partition roleVar=Group(test='group 1' train='group 2')
  ...
run;
```

After you reserve observations for training, validation, and testing, a model fit on the training data is scored on the validation and test data, and the average check loss (ACL) is computed separately for each of these subsets. The ACL for each data role is the error sum of squares for observations in that role divided by the number of observations in that role.

Using the Validation ACL as the STOP= Criterion

If you have provided observations for validation, then you can specify STOP=VALIDATE as a suboption of the METHOD= option in the SELECTION statement. At step *k* of the selection process, the best candidate effect to enter or leave the current model is determined. Here "best candidate" means the effect that gives the best value of the SELECT= criterion; this criterion need not be based on the validation data. The validation ACL is computed for the model in which this candidate effect is added or removed. If this validation ACL is greater than the validation ACL for the model at step *k*, then the selection process terminates at step *k*.

Using the Validation ACL as the CHOOSE= Criterion

When you specify the CHOOSE=VALIDATE suboption of the METHOD= option in the SELECTION statement, the validation ACL is computed for the models at each step of the selection process. The smallest model at any step that yields the smallest validation ACL is selected.

Using the Validation ACL as the SELECT= Criterion

You request the validation ACL as the selection criterion by specifying the SELECT=VALIDATE suboption of the METHOD= option in the SELECTION statement. At step *k* of the selection process, the validation ACL is computed for each model in which a candidate for entry is added or a candidate for removal is dropped. The selected candidate for entry or removal is the one that yields a model that has the minimal validation ACL.

Computational Method

Multithreading

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to those that occur in sequential (single-threaded) execution.

The number of threads that the HPQUANTSELECT procedure spawns is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the CPU count by using the CPUCOUNT= SAS system option. For example, if you specify the following statement, the HPQUANTSELECT procedure schedules threads as if it were executing on a system that had at most four CPUs:

```
options cpucount=4;
```

- You can specify the NTHREADS= option in the PERFORMANCE statement to determine the number of threads. This specification overrides the system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads is displayed in the "Performance Information" table, which is part of the default output. The HPQUANTSELECT procedure allocates one thread per CPU.

PROC HPQUANTSELECT divides the data processing on a single machine among the threads—that is, the HPQUANTSELECT procedure implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and you are running on four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. These operations include the following:

- variable levelization
- effect levelization
- formation of the crossproducts matrix
- quantile regression model fitting
- estimation of covariance matrix for parameter estimates
- evaluation of predicted residual sums of check losses on validation and test data
- scoring of observations

In addition, operations on matrices such as sweeps might be multithreaded if the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Output Data Set

Many SAS procedures add the variables from the input data set when an observationwise output data set is created. The assumption of high-performance analytical procedures is that the input data sets can be large and contain many variables. For performance reasons, the output data set contains the following:

- variables that are explicitly created by the statement
- variables that are listed in the ID statement or specified by using the COPYVAR= option
- distribution keys or hash keys that are transferred from the input data set

The high-performance analytical procedures enable you to add output data set information that is necessary for subsequent SQL joins without copying the entire input data set to the output data set.

Displayed Output

The following sections describe the output that PROC HPQUANTSELECT produces. The output is organized into various tables, which are discussed in their order of appearance.

Performance Information

The "Performance Information" table is produced by default and displays information about the single-machine mode and the number of threads.

Data Access Information

The "Data Access Information" table is produced by default. For the input and output data sets, it displays the libref and data set name, the engine used to access the data, the role (input or output) of the data set, and path that data followed to the reach the computation.

Model Information

The "Model Information" table displays basic information about the model, such as the response variable, the weight variable, and the type of parameterization that is used for classification variables named in the CLASS statement.

Selection Information

When you specify the SELECTION statement, by default the HPQUANTSELECT procedure produces a series of tables that contain information about the model selection. The "Selection Information" table informs you about the model selection method; select, stop, and choose criteria; and other parameters that govern the selection. You can suppress this table by specifying DETAILS=NONE in the SELECTION statement.

Number of Observations

The "Number of Observations" table displays the number of observations that are read from the input data set and the number of observations that are used in the analysis. If you use a PARTITION statement, the table also displays the number of observations that are used for each data role.

Class Level Information

The "Class Level Information" table lists the levels of every variable that is specified in the CLASS statement. You should check this information to ensure that the data are correct. You can adjust the order of the CLASS variable levels by specifying the ORDER= option in the CLASS statement. You can suppress the "Class Level Information" table completely or partially by specifying the NOCLPRINT= option in the PROC HPQUANTSELECT statement.

If the classification variables are in the reference parameterization, the "Class Level Information" table also displays the reference value for each variable. This table also indicates which, if any, of the classification variables are split by using the SPLIT option in the CLASS statement.

Dimensions

The "Dimensions" table displays information about the number of effects and the number of parameters from which the selected model is chosen. If you use split classification variables, then this table also includes the number of effects after splitting is taken into account.

Entry and Removal Candidates

When you specify the DETAILS=ALL or DETAILS=STEPS option in the SELECTION statement, the HPQUANTSELECT procedure produces "Entry Candidates" and "Removal Candidates" tables that display the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. The effects are displayed in order from best to worst of the selection criterion.

Selection Summary

When you specify the SELECTION statement, the HPQUANTSELECT procedure produces the "Selection Summary" table, which contains information about the sequence of steps of the selection process. For each step, the effect that was entered or dropped is displayed along with the statistics used to select the effect, stop the selection, and choose the selected model. For all criteria that you can use for model selection, the steps at which the optimal values of these criteria occur are also indicated.

You can suppress the "Selection Summary" table by specifying DETAILS=NONE in the SELECTION statement.

Stop Reason

The "Stop Reason" table displays the reason why the selection stopped. To facilitate programmatic use of this table, an integer code is assigned to each reason and is included if you output this table by using an ODS OUTPUT statement. The reasons and their associated codes follow:

Code	Stop Reason
1	All eligible effects are in the model.
2	All eligible effects have been removed.
3	Specified maximum number of steps have been done.
4	The model contains the specified maximum number of effects.
5	The model contains the specified minimum number of effects (for backward selection).
6	The stopping criterion is at a local optimum.
7	No suitable add or drop candidate could be found.
8	Adding or dropping any effect does not improve the selection criterion.
9	No candidate meets the appropriate SLE or SLS significance level.
10	Stepwise selection is cycling.
11	The model is an exact fit.
12	Dropping an effect would result in an empty model.

You can suppress the "Stop Reason" table by specifying DETAILS=NONE in the SELECTION statement.

Selection Reason

When you specify the SELECTION statement, the HPQUANTSELECT procedure produces a simple table that contains text that explains why the final model was selected.

You can suppress the "Selection Reason" table by specifying DETAILS=NONE in the SELECTION statement.

Selected Effects

When you specify the SELECTION statement, the HPQUANTSELECT procedure produces a simple table that contains text that lists the selected effects in the final model.

Fit Statistics

The "Fit Statistics" table displays fit statistics for the selected model. The statistics include the following:

- Objective Function, total sum of check losses. Objective Function is denoted as $D(\tau)$ in Table 66.6.
- R1, a measure between 0 and 1 that indicates the portion of the (corrected) total check losses attributed to the fit rather than left to residuals. It is calculated as $1 - \frac{D(\tau)}{D_0(\tau)}$. It is the quantile regression counterpart of the linear regression R square.
- Adj R1, the adjusted R1, a version of R1 that has been adjusted for degrees of freedom. It is calculated as

  $$\text{Adj R1} = 1 - \frac{n-i}{n-p}(1 - \text{R1})$$

  where $i = 1$ if the intercept is forced in and $i = 0$ otherwise , n is the number of observations used to fit the model, and p is the number of parameters in the model.
- the fit criteria AIC, AICC, and SBC if they are used in the selection process. For the formulas to evaluate these criteria, see Table 66.6.
- the average check loss (ACL) on the training, validation, and test data

You can request the "Fit Statistics" tables for the model at each step of the selection process by specifying the DETAILS= option in the SELECTION statement.

Parameter Estimates

The "Parameter Estimates" table displays the parameters in the selected model and their estimates. The information that is displayed for each parameter in the selected model includes the following:

- the parameter label, which includes the effect name and level information for effects that contain classification variables
- the degrees of freedom (DF) for the parameter. There is one degree of freedom unless the model is not full rank.

- the parameter estimate
- the standardized parameter estimate. PROC HPQUANTSELECT outputs the standardized parameter estimate only if you specify the STB option in the MODEL statement.

If you specify the CLB option in the MODEL statement, PROC HPQUANTSELECT also outputs the following information:

- the standard error, which is the estimate of the standard deviation of the parameter estimate
- the $100(1-\alpha)\%$ confidence limits for the parameter estimate
- t value, the t test that the parameter is 0. This is computed as the parameter estimate divided by the standard error.
- the Pr > $|t|$, the probability that a t statistic would obtain a greater absolute value than that observed when the true parameter is 0. This is the two-tailed significance probability.

For more information about standard errors, confidence limits, t values, and the Pr > $|t|$ probability, see the section "More Statistics for Parameter Estimates" on page 5108.

You can request "Parameter Estimates" tables for the model at each step of the selection process by specifying the DETAILS= option in the SELECTION statement.

Timing Information

If you specify the DETAILS option in the PERFORMANCE statement, PROC HPQUANTSELECT also produces a "Timing" table that displays the elapsed time for each main task of PROC HPQUANTSELECT.

ODS Table Names

Each table that the HPQUANTSELECT procedure creates has a name associated with it. You must use this name to refer to the table when you use ODS statements. These names are listed in Table 66.9.

Table 66.9 ODS Tables Produced by PROC HPQUANTSELECT

Table Name	Description	Required Statement / Option
ClassLevels	Level information from the CLASS statement	CLASS
DataAccessInfo	Information about modes of data access	Default output
Dimensions	Model dimensions	Default output
EntryCandidates	Candidates for entry at step	SELECTION DETAILS=ALL | STEPS
FitStatistics	Fit statistics	Default output
ModelInfo	Information about the modeling environment	Default output
NObs	Number of observations read and used	Default output

Table 66.9 *continued*

Table Name	Description	Required Statement / Option
ParameterEstimates	Solutions for the parameter estimates associated with effects in the MODEL statement	Default output
PerformanceInfo	Information about high-performance computing environment	Default output
RemovalCandidates	Candidates for removal at step	SELECTION DETAILS=ALL | STEPS
SelectedEffects	List of selected effects	SELECTION
SelectionInfo	Information about selection settings	Default output
SelectionReason	Reason for selecting the final model	SELECTION
SelectionSummary	Summary information about the model selection steps	SELECTION
StopReason	Reason selection was terminated	SELECTION
Timing	Timing breakdown by task	PERFORMANCE DETAILS

Examples: HPQUANTSELECT Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 66.1: Simulation Study

(View the complete code for this example (hpqtrex1.sas) in the example repository.)

This example is based on "Example 104.1: Simulation Study" on page 8693. This simulation study shows how you can use the forward selection method to select quantile regression models for single quantile levels. The following statements simulate a data set from a naive instrumental model (Chernozhukov and Hansen 2008):

```
%let seed=321;
%let p=20;
%let n=3000;

data analysisData;
   array x{&p} x1-x&p;
   do i=1 to &n;
      U  = ranuni(&seed);
      x1 = ranuni(&seed);
      x2 = ranexp(&seed);
      x3 = abs(rannor(&seed));
      y  = x1*(U-0.1) + x2*(U*U-0.25) + x3*(exp(U)-exp(0.9));
      do j=4 to &p;
         x{j} = ranuni(&seed);
```

```
      end;
      output;
   end;
run;
```

Variable U in the data set indicates the true quantile level of the response y conditional on $\mathbf{x} = (x_1, \ldots, x_p)$.

Let $Q_Y(\tau|\mathbf{x}) = \mathbf{x}\boldsymbol{\beta}(\tau)$ denote the underlying quantile regression model, where $\boldsymbol{\beta}(\tau) = (\beta_1(\tau), \ldots, \beta_p(\tau))'$. Then, the true parameter functions are

$$\begin{aligned}
\beta_1(\tau) &= \tau - 0.1 \\
\beta_2(\tau) &= \tau^2 - 0.25 \\
\beta_3(\tau) &= \exp(\tau) - \exp(0.9) \\
\beta_4(\tau) &= \cdots = \beta_p(\tau) = 0
\end{aligned}$$

It is easy to see that, at $\tau = 0.1$, only $\beta_2(0.1) = -0.24$ and $\beta_3(0.1) = \exp(0.1) - \exp(0.9) \approx -1.354432$ are nonzero parameters. Therefore, an effective effect-selection method should select x2 and x3 and drop all the other effects in this data set at $\tau = 0.1$. By the same rationale, x1 and x3 should be selected at $\tau = 0.5$ with $\beta_1(0.5) = 0.4$ and $\beta_3(0.5) \approx -0.810882$, and x1 and x2 should be selected at $\tau = 0.9$ with $\beta_1(0.9) = 0.8$ and $\beta_2(0.9) = 0.56$.

The following statements use PROC HPQUANTSELECT with the forward selection method. The STB option and the CLB option in the MODEL statement request the standardized parameter estimates and the confidence limits of parameter estimates, respectively.

```
proc hpquantselect data=analysisData;
   model y= x1-x&p / quantile=0.1 0.5 0.9 stb clb;
   selection method=forward;
   output out=out p=pred;
run;
```

Output 66.1.1 shows that, by default, the CHOOSE= and STOP= options are both set to SBC.

Output 66.1.1 Model Information

The HPQUANTSELECT Procedure

Selection Information	
Selection Method	Forward
Select Criterion	SBC
Stop Criterion	SBC
Effect Hierarchy Enforced	None
Stop Horizon	3

Output 66.1.2, Output 66.1.3, and Output 66.1.4 display the selected effects and the parameter estimates for $\tau = 0.1$, $\tau = 0.5$, and $\tau = 0.9$, respectively. You can see that the forward selection method correctly selects active effects for all three quantile levels.

Output 66.1.2 Parameter Estimates at $\tau = 0.1$

The HPQUANTSELECT Procedure

Quantile Level = 0.1
Selected Model

Selected Effects: Intercept x2 x3

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	0.01179	0	0.01192	-0.01158	0.03516	0.99	0.3225
x2	1	-0.22871	-0.21829	0.00946	-0.24725	-0.21017	-24.19	<.0001
x3	1	-1.37991	-0.78452	0.01556	-1.41042	-1.34939	-88.67	<.0001

Output 66.1.3 Parameter Estimates at $\tau = 0.5$

The HPQUANTSELECT Procedure

Quantile Level = 0.5
Selected Model

Selected Effects: Intercept x1 x3

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	0.01178	0	0.03418	-0.05524	0.07879	0.34	0.7304
x1	1	0.42584	0.11879	0.06237	0.30355	0.54814	6.83	<.0001
x3	1	-0.86332	-0.49082	0.04765	-0.95674	-0.76989	-18.12	<.0001

Output 66.1.4 Parameter Estimates at $\tau = 0.9$

The HPQUANTSELECT Procedure

Quantile Level = 0.9
Selected Model

Selected Effects: Intercept x1 x2

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	-0.00774	0	0.03292	-0.07228	0.05680	-0.24	0.8142
x1	1	0.78294	0.21841	0.05134	0.68228	0.88360	15.25	<.0001
x2	1	0.57644	0.55018	0.03422	0.50935	0.64354	16.85	<.0001

Example 66.2: Growth Charts for Body Mass Index

(View the complete code for this example (hpqtrex2.sas) in the example repository.)

This example is modeled on the example in the section "Getting Started: QUANTSELECT Procedure" on page 8639. It highlights the use of the HPQUANTSELECT procedure for multiple-level quantile regression by creating growth charts for men's body mass index (BMI).

BMI, which is defined as the ratio of weight (kg) to squared height (m^2), is a standard measure for categorizing individuals as overweight or underweight. The percentiles of BMI for specified ages are of particular interest. This example draws smooth BMI quantile curves conditional on Age, which can serve as BMI growth charts in medical diagnosis to identify BMI percentiles for subjects.

The BMIMen data set is from the 1999–2000 and 2001–2002 survey results for men that are published by the National Center for Health Statistics. It contains the two variables BMI and Age with 3,264 observations.

```
data bmimen;
   input BMI Age @@;
   SqrtAge = sqrt(Age);
   InveAge = 1/Age;
   LogBMI  = log(BMI);
   datalines;
18.6  2.0 17.1  2.0 19.0  2.0 16.8  2.0 19.0  2.1  15.5   2.1
16.7  2.1 16.1  2.1 18.0  2.1 17.8  2.1 18.3  2.1  16.9   2.1
15.9  2.1 20.6  2.1 16.7  2.1 15.4  2.1 15.9  2.1  17.7   2.1

   ... more lines ...

29.0 80.0 24.1 80.0 26.6 80.0 24.2 80.0 22.7 80.0  28.4  80.0
26.3 80.0 25.6 80.0 24.8 80.0 28.6 80.0 25.7 80.0  25.8  80.0
22.5 80.0 25.1 80.0 27.0 80.0 27.9 80.0 28.5 80.0  21.7  80.0
33.5 80.0 26.1 80.0 28.4 80.0 22.7 80.0 28.0 80.0  42.7  80.0
;
```

The logarithm of BMI is used as the response. (Although this approach does not improve the quantile regression fit, it helps with statistical inference.) The following statements fit quantile regression models for the BMIMen data set at 10 quantile levels:

```
%let quantile=0.03 0.05 0.1 0.25 0.5 0.75 0.85 0.90 0.95 0.97;
%let nq=10;

proc hpquantselect data=BMIMen;
   model logBMI = InveAge SqrtAge Age SqrtAge*Age Age*Age Age*Age*Age
         / quantile=&quantile;
   code file='bmicode.sas';
   output out=Bmiout copyvars=(BMI Age) pred=P_LogBMI;
run;
```

The CODE statement enables you to write a SAS DATA step to compute quantile predictions of the fitted model. The OUTPUT statement outputs the mean predicted quantiles for the 10 specified quantile levels. The PRED= option in the OUTPUT statement specifies the variable names for the quantile predictions. For examples, p1 is for quantile level 0.03, and p2 is for quantile level 0.05.

The following statements define and apply a SAS macro function to create a quantile curves plot for the BMIMen data set:

```
%let BMIcolor=red olive orange blue brown gray violet black gold green;

%macro plotBMI;
   data BmiPred;
      set Bmiout;
      %do j=1 %to &nq;
         predBMI&j = exp(P_LogBMI&j);
      %end;
      label %do j=1 %to &nq;
               predBMI&j=%qscan(&quantile,&j,%str( ))
            %end;;
   run;

   proc sort data=BmiPred;
      by Age;
   run;

   proc sgplot data=BmiPred;
      %do j=1 %to &nq;
         series y=predBMI&j x=Age/lineattrs=(thickness=2
         color=%qscan(&BMIcolor,&j,%str( )));
      %end;
      scatter y=BMI x=Age/markerattrs=(size=5);
   run;
%mend;

%plotBMI;
```

Figure 66.2.1 shows the BMI quantile curves, which can serve as BMI growth charts. For example, the percentiles of any observations (small blue circles) that are located between the top 0.95 quantile (gold) curve and the 0.97 quantile (green) curve are between the 95th percentile and the 97th percentile. By using this rule, you can measure the percentile range for any observations of interest.

Output 66.2.1 Growth Chart for Body Mass Index

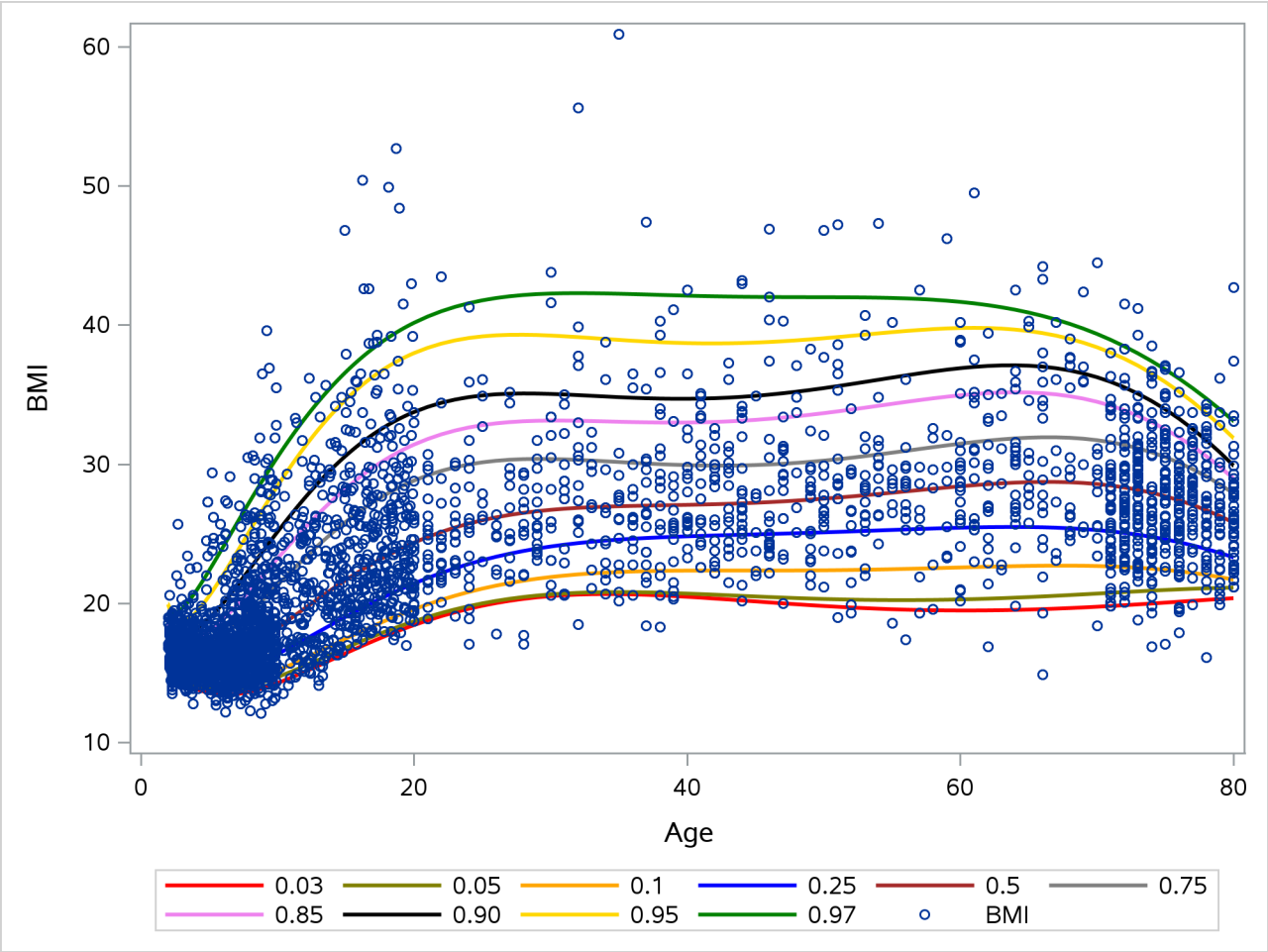

Other than using the OUTPUT statement, you can also calculate quantile predictions by using the CODE statement. The following statements show how to use the SAS DATA step and the SAS file *bmicode*, which the CODE statement requests, to calculate quantile predictions for the BMIMen data set:

```
data Newmen;
   set BMIMen;
   %inc bmicode;
run;
```

The SET statement in the SAS DATA step specifies a data set for computing quantile predictions. This is usually a new data set that you want to score. This example uses the BMIMen data set again, so the quantile predictions in the Newmen data set are identical to those in the Bmiout data set. The following statements compare the Bmiout data set with the Newmen data set:

```
proc compare data=Bmiout compare=Newmen criterion=0.00001;
run;
```

References

Akaike, H. (1969). "Fitting Autoregressive Models for Prediction." *Annals of the Institute of Statistical Mathematics* 21:243–247.

Chernozhukov, V., and Hansen, C. (2008). "Instrumental Variable Quantile Regression: A Robust Inference Approach." *Journal of Econometrics* 142:379–398.

Collier Books (1987). *The 1987 Baseball Encyclopedia Update*. New York: Macmillan.

Gertz, E. M., and Griffin, J. D. (2005). *Support Vector Machine Classifiers for Large Data Sets*. Technical Report ANL/MCS-TM-289, Mathematics and Computer Science Division, Argonne National Laboratory.

Gertz, E. M., and Griffin, J. D. (2010). "Using an Iterative Linear Solver in an Interior-Point Method for Generating Support Vector Machines." *Computational Optimization and Applications* 47:431–453.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Koenker, R. (2005). *Quantile Regression*. New York: Cambridge University Press.

Koenker, R., and Bassett, G. W. (1978). "Regression Quantiles." *Econometrica* 46:33–50.

Koenker, R., and Bassett, G. W. (1982). "Tests of Linear Hypotheses and l_1 Estimation." *Econometrica* 50:1577–1583.

Koenker, R., and Machado, A. F. (1999). "Goodness of Fit and Related Inference Processes for Quantile Regression." *Journal of the American Statistical Association* 94:1296–1310.

Reichler, J. L., ed. (1987). *The 1987 Baseball Encyclopedia Update*. New York: Macmillan.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Time Inc. (1987). "What They Make." *Sports Illustrated* (April 20): 54–81.

Yuan, M. (2006). "GACV for Quantile Smoothing Splines." *Computational Statistics and Data Analysis* 50:813–829. http://econpapers.repec.org/article/eeecsdana/v_3a50_3ay_3a2006_3ai_3a3_3ap_3a813-829.htm.

Chapter 67
The HPREG Procedure

Contents

Overview: HPREG Procedure . . . **5130**
- PROC HPREG Features . . . 5130
- PROC HPREG Contrasted with Other SAS Procedures . . . 5131

Getting Started: HPREG Procedure . . . **5132**

Syntax: HPREG Procedure . . . **5138**
- PROC HPREG Statement . . . 5138
- BY Statement . . . 5139
- CLASS Statement . . . 5140
- CODE Statement . . . 5140
- FREQ Statement . . . 5141
- ID Statement . . . 5141
- MODEL Statement . . . 5141
- OUTPUT Statement . . . 5143
- PARTITION Statement . . . 5145
- PERFORMANCE Statement . . . 5145
- SELECTION Statement . . . 5146
- WEIGHT Statement . . . 5148

Details: HPREG Procedure . . . **5148**
- Criteria Used in Model Selection . . . 5148
- Diagnostic Statistics . . . 5150
- Classification Variables and the SPLIT Option . . . 5151
- Using Validation and Test Data . . . 5152
- Computational Method . . . 5154
- Output Data Set . . . 5155
- Screening . . . 5155
- Displayed Output . . . 5156
- ODS Table Names . . . 5161

Examples: HPREG Procedure . . . **5162**
- Example 67.1: Model Selection with Validation . . . 5162
- Example 67.2: Forward-Swap Selection . . . 5168
- Example 67.3: Forward Selection with Screening . . . 5171

References . . . **5177**

Overview: HPREG Procedure

The HPREG procedure is a high-performance procedure that fits and performs model selection for ordinary linear least squares models. The models supported are standard independently and identically distributed general linear models, which can contain main effects that consist of both continuous and classification variables and interaction effects of these variables. The procedure offers extensive capabilities for customizing the model selection with a wide variety of selection and stopping criteria, from traditional and computationally efficient significance-level-based criteria to more computationally intensive validation-based criteria. PROC HPREG also provides a variety of regression diagnostics that are conditional on the selected model.

PROC HPREG Features

The main features of the HPREG procedure are as follows:

- **Model specification**
  - supports GLM and reference parameterization for classification effects
  - supports any degree of interaction (crossed effects) and nested effects
  - supports hierarchy among effects
  - supports partitioning of data into training, validation, and testing roles
  - supports a FREQ statement for grouped analysis
  - supports a WEIGHT statement for weighted analysis
- **Selection control**
  - provides multiple effect-selection methods
  - enables selection from a very large number of effects (tens of thousands)
  - offers selection of individual levels of classification effects
  - provides effect selection based on a variety of selection criteria
  - provides stopping rules based on a variety of model evaluation criteria
  - supports stopping and selection rules based on external validation and leave-one-out cross validation
- **Display and output**
  - produces output data sets that contain predicted values, residuals, studentized residuals, confidence limits, and influence statistics

The HPREG procedure supports the following effect selection methods. For a more detailed description of these methods, see the section "Methods" on page 582.

- Forward selection starts with no effects in the model and adds effects.

- Backward elimination starts with all effects in the model and deletes effects.
- Stepwise regression is similar to forward selection except that effects already in the model do not necessarily stay there.
- Forward-swap selection is a modification of forward selection. Before any addition step, PROC HPREG makes all pairwise swaps of effects in and out of the current model that improve the selection criterion. When the selection criterion is R square, this method coincides with the MAXR method in the REG procedure in SAS/STAT software.
- Least angle regression, like forward selection, starts with no effects in the model and adds effects. The parameter estimates at any step are "shrunk" when compared to the corresponding least squares estimates.
- Lasso adds and deletes parameters based on a version of ordinary least squares in which the sum of the absolute regression coefficients is constrained. PROC HPREG also supports adaptive lasso selection where weights are applied to each of the parameters in forming the lasso constraint.

Hybrid versions of LAR and LASSO methods are also supported. They use LAR or LASSO to select the model, but then estimate the regression coefficients by ordinary weighted least squares.

Because the HPREG procedure is a high-performance analytical procedure, it also exploits all the available cores and concurrent threads.

PROC HPREG Contrasted with Other SAS Procedures

For general contrasts between SAS high-performance statistical procedures and other SAS procedures, see the section "Common Features of SAS High-Performance Statistical Procedures" on page 572. The following remarks contrast the HPREG procedure with the GLMSELECT, GLM, and REG procedures in SAS/STAT software.

A major functional difference between the HPREG and REG procedures is that the HPREG procedure enables you to specify general linear models that include classification variables. In this respect it is similar to the GLM and GLMSELECT procedures. In terms of the supported model selection methods, the HPREG procedure most resembles the GLMSELECT procedure. Like the GLMSELECT procedure but different from the REG procedure, the HPREG procedure supports the LAR and LASSO methods, the ability to use external validation data and cross validation as selection criteria, and extensive options to customize the selection process. The HPREG procedure does not support the MAXR and MINR methods that are available in the REG procedure. Nor does the HPREG procedure include any support for the all-subset-based methods that you can find in the REG procedure.

The CLASS statement in the HPREG procedure permits two parameterizations: the GLM-type parameterization and a reference parameterization. In contrast to the GLMSELECT, GENMOD, LOGISTIC, and other procedures that permit multiple parameterizations, the HPREG procedure does not mix parameterizations across the variables in the CLASS statement. In other words, all classification variables are in the same parameterization, and this parameterization is either the GLM or reference parameterization.

Like the REG procedure but different from the GLMSELECT procedure, the HPREG procedure does not perform model selection by default. If you request model selection by using the SELECTION statement

then the default selection method is stepwise selection based on the SBC criterion. This default matches the default method used in PROC GLMSELECT.

As with the REG procedure but not supported with the GLMSELECT procedure, you can request observation-wise residual and influence diagnostics in the OUTPUT statement and variance inflation and tolerance statistics for the parameter estimates. If the fitted model has been obtained by performing model selection, then these statistics are conditional on the selected model and do not take the variability introduced by the selection process into account.

Getting Started: HPREG Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpreggs.sas.)

The following example is closely modeled on the example in the section "Getting Started: GLMSELECT Procedure" in the *SAS/STAT User's Guide*.

The Sashelp.Baseball data set contains salary and performance information for Major League Baseball players who played at least one game in both the 1986 and 1987 seasons, excluding pitchers. The salaries (*Sports Illustrated,* April 20, 1987) are for the 1987 season and the performance measures are from 1986 (Collier Books, *The 1987 Baseball Encyclopedia Update*). The following step displays in Figure 67.1 the variables in the data set:

```
proc contents varnum data=sashelp.baseball;
   ods select position;
run;
```

Suppose you want to investigate whether you can model the players' salaries for the 1987 season based on performance measures for the previous season. The aim is to obtain a parsimonious model that does not overfit this particular data, making it useful for prediction. This example shows how you can use PROC HPREG as a starting point for such an analysis. Since the variation of salaries is much greater for the higher salaries, it is appropriate to apply a log transformation to the salaries before doing the model selection.

The following statements select a model with the default settings for stepwise selection:

```
proc hpreg data=sashelp.baseball;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError;
   selection method=stepwise;
run;
```

The default output from this analysis is presented in Figure 67.2 through Figure 67.6.

Figure 67.1 Sashelp.Baseball Data Set

The CONTENTS Procedure

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	Name	Char	18	Player's Name
2	Team	Char	14	Team at the End of 1986
3	nAtBat	Num	8	Times at Bat in 1986
4	nHits	Num	8	Hits in 1986
5	nHome	Num	8	Home Runs in 1986
6	nRuns	Num	8	Runs in 1986
7	nRBI	Num	8	RBIs in 1986
8	nBB	Num	8	Walks in 1986
9	YrMajor	Num	8	Years in the Major Leagues
10	CrAtBat	Num	8	Career Times at Bat
11	CrHits	Num	8	Career Hits
12	CrHome	Num	8	Career Home Runs
13	CrRuns	Num	8	Career Runs
14	CrRbi	Num	8	Career RBIs
15	CrBB	Num	8	Career Walks
16	League	Char	8	League at the End of 1986
17	Division	Char	8	Division at the End of 1986
18	Position	Char	8	Position(s) in 1986
19	nOuts	Num	8	Put Outs in 1986
20	nAssts	Num	8	Assists in 1986
21	nError	Num	8	Errors in 1986
22	Salary	Num	8	1987 Salary in $ Thousands
23	Div	Char	16	League and Division
24	logSalary	Num	8	Log Salary

Figure 67.2 Performance, Data Access, Model, and Selection Information

The HPREG Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

Data Access Information			
Data	**Engine**	**Role**	**Path**
SASHELP.BASEBALL	V9	Input	On Client

Model Information	
Data Source	SASHELP.BASEBALL
Dependent Variable	logSalary
Class Parameterization	GLM

Figure 67.2 *continued*

Selection Information	
Selection Method	Stepwise
Select Criterion	SBC
Stop Criterion	SBC
Effect Hierarchy Enforced	None
Stop Horizon	3

Figure 67.2 displays the "Performance Information," "Data Access Information," "Model Information," and "Selection Information" tables. The "Performance Information" table shows that procedure executes in single-machine mode—that is, the model is fit on the machine where the SAS session executes. This run of the HPREG procedure was performed on a multicore machine with four CPUs; one computational thread was spawned per CPU.

The "Data Access Information" table shows that the input data set is accessed with the V9 (base) engine on the client machine.

The "Model Information" table identifies the data source and response and shows that the CLASS variables are parameterized in the GLM parameterization, which is the default.

The "Selection Information" provides details about the method and criteria used to perform the model selection. The requested selection method is a variant of the traditional stepwise selection where the decisions about what effects to add or drop at any step and when to terminate the selection are both based on the Schwarz Bayesian information criterion (SBC). The effect in the current model whose removal yields the maximal decrease in the SBC statistic is dropped provided this lowers the SBC value. When no further decrease in the SBC value can be obtained by dropping an effect in the model, the effect whose addition to the model yields the lowest SBC statistic is added and the whole process is repeated. The method terminates when dropping or adding any effect increases the SBC statistic.

Figure 67.3 displays the "Number of Observations," "Class Levels," and "Dimensions" tables. The "Number of Observations" table shows that of the 322 observations in the input data, only 263 observations are used in the analysis because there are observations with incomplete data. The "Class Level Information" table lists the levels of the classification variables "division" and "league." When you specify effects that contain classification variables, the number of parameters is usually larger than the number of effects. The "Dimensions" table shows the number of effects and the number of parameters considered.

Figure 67.3 Number of Observations, Class Levels, and Dimensions

Number of Observations Read	322
Number of Observations Used	263

Class Level Information		
Class	**Levels**	**Values**
League	2	American National
Division	2	East West

Dimensions	
Number of Effects	19
Number of Parameters	21

The "Stepwise Selection Summary" table in Figure 67.4 shows the effect that was added or dropped at each step of the selection process together with fit statistics for the model at each step. In this case, both selection and stopping are based on the SBC statistic.

Figure 67.4 Selection Summary Table

The HPREG Procedure

Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	SBC
0	Intercept		1	-57.2041
1	CrRuns		2	-194.3166
2	nHits		3	-252.5794
3	YrMajor		4	-262.7322
4		CrRuns	3	-262.8353
5	nBB		4	-269.7804*

*** Optimal Value of Criterion**

Figure 67.5 displays the "Stop Reason," "Selection Reason," and "Selected Effects" tables. Note that these tables are displayed without any titles. The "Stop Reason" table indicates that selection stopped because adding or removing any effect would worsen the SBC value that is used as the selection criterion. In this case, because no CHOOSE= criterion is specified in the SELECTION statement, the final model is the selected model; this is indicated in the "Selection Reason" table. The "Selected Effects" table lists the effects in the selected model.

Figure 67.5 Stopping and Selection Reasons

Stepwise selection stopped because adding or removing an effect does not improve the SBC criterion.

The model at step 5 is selected.

Selected Effects:	Intercept nHits nBB YrMajor

The "Analysis of Variance," "Fit Statistics," and "Parameter Estimates" tables shown in Figure 67.6 give details of the selected model.

Figure 67.6 Details of the Selected Model

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	120.52553	40.17518	120.12	<.0001
Error	259	86.62820	0.33447		
Corrected Total	262	207.15373			

Figure 67.6 *continued*

Root MSE	0.57834
R-Square	0.58182
Adj R-Sq	0.57697
AIC	-19.06903
AICC	-18.83557
SBC	-269.78041
ASE	0.32938

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	4.013911	0.111290	36.07	<.0001
nHits	1	0.007929	0.000994	7.98	<.0001
nBB	1	0.007280	0.002049	3.55	0.0005
YrMajor	1	0.100663	0.007551	13.33	<.0001

You might want to examine regression diagnostics for the selected model to investigate whether collinearity among the selected parameters or the presence of outlying or high leverage observations might be impacting the fit produced. The following statements include some options and statements to obtain these diagnostics:

```
proc hpreg data=sashelp.baseball;
  id name;
  class league division;
  model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                    yrMajor crAtBat crHits crHome crRuns crRbi
                    crBB league division nOuts nAssts nError / vif clb;
  selection method=stepwise;
  output out=baseballOut p=predictedLogSalary r h cookd rstudent;
run;
```

The VIF and CLB options in the MODEL statement request variance inflation factors and 95% confidence limits for the parameter estimates. Figure 67.7 shows the "Parameter Estimates" with these requested statistics. The variance inflation factors (VIF) measure the inflation in the variances of the parameter estimates due to collinearities that exist among the regressor (independent) variables. Although there are no formal criteria for deciding whether a VIF is large enough to affect the predicted values, the VIF values for the selected effects in this example are small enough to indicate that there are no collinearity issues among the selected regressors.

Figure 67.7 Parameter Estimates with Additional Statistics

The HPREG Procedure

Selected Model

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Variance Inflation	95% Confidence Limits	
Intercept	1	4.013911	0.111290	36.07	<.0001	0	3.79476	4.23306
nHits	1	0.007929	0.000994	7.98	<.0001	1.49642	0.00597	0.00989
nBB	1	0.007280	0.002049	3.55	0.0005	1.52109	0.00325	0.01131
YrMajor	1	0.100663	0.007551	13.33	<.0001	1.02488	0.08579	0.11553

By default, high-performance statistical procedures do not include all variables from the input data set in output data sets. The ID statement specifies that the variable name in the input data set be added as an identification variable in the baseballOut data set that is produced by the OUTPUT statement. In addition to this variable, the OUTPUT statement requests that predicted values, raw residuals, leverage values, Cook's D statistics, and studentized residuals be added in the output data set. Note that default names are used for these statistics except for the predicted values for which a specified name, predictedLogSalary, is supplied. The following statements use PROC PRINT to display the first five observations of this output data set:

```
proc print data=baseballOut(obs=5);
run;
```

Figure 67.8 First 5 Observations of the baseballOut Data Set

Obs	Name	predictedLogSalary	Residual	H	COOKD	RSTUDENT
1	Allanson, Andy	4.73980	.	0.016087	.	.
2	Ashby, Alan	6.34935	-0.18603	0.012645	.000335535	-0.32316
3	Davis, Alan	5.89993	0.27385	0.019909	.001161794	0.47759
4	Dawson, Andre	6.50852	-0.29392	0.011060	.000730178	-0.51031
5	Galarraga, Andres	5.12344	-0.60711	0.009684	.002720358	-1.05510

Syntax: HPREG Procedure

The following statements are available in the HPREG procedure:

```
PROC HPREG <options> ;
    BY variables ;
    CODE <options> ;
    CLASS variable <(options)>... <variable <(options)>> </global-options> ;
    MODEL dependent = <effects> </model-options> ;
    OUTPUT <OUT=SAS-data-set>
            <keyword <=name>>...
            <keyword <=name>> </options> ;
    PARTITION <partition-options> ;
    PERFORMANCE <performance-options> ;
    SELECTION options ;
    FREQ variable ;
    ID variables ;
    WEIGHT variable ;
```

The PROC HPREG statement and a single MODEL statement are required. All other statements are optional. The CLASS statement can appear multiple times. If a CLASS statement is specified, it must precede the MODEL statement.

PROC HPREG Statement

```
PROC HPREG <options> ;
```

The PROC HPREG statement invokes the procedure. Table 67.1 summarizes the options in the PROC HPREG statement by function.

Table 67.1 PROC HPREG Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
NAMELEN=	Limits the length of effect names
Options Related to Output	
NOPRINT	Suppresses ODS output
NOCLPRINT	Limits or suppresses the display of class levels
User-Defined Formats	
FMTLIBXML=	Specifies a file reference for a format stream
Other Options	
ALPHA=	Sets the significance level used for the construction of confidence intervals
SEED=	Sets the seed used for pseudorandom number generation

Following are explanations of the *options* that you can specify in the PROC HPREG statement (in alphabetical order):

ALPHA=*number*
sets the significance level used for the construction of confidence intervals. The value must be between 0 and 1; the default value of 0.05 results in 95% intervals. This option affects the OUTPUT statement keywords LCL, LCLM, UCL, and UCLM, and the CLB option in the MODEL statement.

DATA=*SAS-data-set*
names the input SAS data set to be used by PROC HPREG. The default is the most recently created data set.

FMTLIBXML=*file-ref*
specifies the file reference for the XML stream that contains the user-defined format definitions. See the section "Working with Formats" on page 568 for details about how to generate a XML stream for your formats.

NAMELEN=*number*
specifies the length to which long effect names are shortened. The default and minimum value is 20.

NOCLPRINT< =*number* **>**
suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, the values of the classification variables are displayed for only those variables whose number of levels is less than *number*. Specifying a *number* helps to reduce the size of the "Class Level Information" table if some classification variables have a large number of levels.

NOPRINT
suppresses the generation of ODS output.

SEED=*number*
specifies an integer used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify a seed, or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer's clock.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC HPREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the HPREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.

- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) >. . . < *variable* < (*options*) > > < / *global-options* > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. The CLASS statement must precede the MODEL statement.

The CLASS statement for SAS high-performance statistical procedures is documented in the section "CLASS Statement" on page 572. The HPREG procedure also supports the following *global-option* in the CLASS statement:

UPCASE

uppercases the values of character-valued CLASS variables before levelizing them. For example, if the UPCASE option is in effect and a CLASS variable can take the values 'a', 'A', and 'b', then 'a' and 'A' represent the same level and the CLASS variable is treated as having only two values: 'A' and 'B'.

CODE Statement

CODE < *options* > ;

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 67.2 summarizes the *options* available in the CODE statement.

Table 67.2 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a numeric variable in the data set that contains the frequency of occurrence for each observation. SAS high-performance statistical procedures that support the FREQ statement treat each observation as if it appeared f times, where f is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the input data set that are transferred to output data sets created by SAS high-performance statistical procedures, provided that the output data set produces one (or more) records per input observation.

For documentation on the common ID statement in SAS high-performance statistical procedures, see the section "ID Statement" on page 576.

MODEL Statement

MODEL *dependent=< effects> / < options>* **;**

The MODEL statement names the dependent variable and the explanatory effects, including covariates, main effects, interactions, and nested effects. If you omit the explanatory effects, the procedure fits an intercept-only model.

After the keyword MODEL, the dependent (response) variable is specified, followed by an equal sign. The explanatory effects follow the equal sign. For information about constructing the model effects, see the section "Parameterization of Model Effects" on page 412.

You can specify the following *options* in the MODEL statement after a slash (/):

CLB

requests the $100(1-\alpha)\%$ upper and lower confidence limits for the parameter estimates. By default, the 95% limits are computed; the ALPHA= option in the PROC HPREG statement can be used to change the α level. The CLB option is not supported when you request METHOD=LAR or METHOD=LASSO in the SELECTION statement.

INCLUDE=*n*

INCLUDE=*single-effect*

INCLUDE=(*effects***)**

forces effects to be included in all models. If you specify INCLUDE=*n*, then the first *n* effects listed in the MODEL statement are included in all models. If you specify INCLUDE=*single-effect* or if you specify a list of effects within parentheses, then the specified effects are forced into all models. The effects that you specify in the INCLUDE= option must be explanatory effects defined in the MODEL statement before the slash (/). The INCLUDE= option is not available when you specify METHOD=LAR or METHOD=LASSO in the SELECTION statement.

NOINT

suppresses the intercept term that is otherwise included in the model.

ORDERSELECT

specifies that, for the selected model, effects be displayed in the order in which they first entered the model. If you do not specify the ORDERSELECT option, then effects in the selected model are displayed in the order in which they appear in the MODEL statement.

START=*n*

START=*single-effect*

START=(*effects***)**

is used to begin the selection process in the FORWARD, FORWARDSWAP, and STEPWISE selection methods from the initial model that you designate. If you specify START=*n*, then the starting model consists of the first *n* effects listed in the MODEL statement. If you specify START=*single-effect* or if you specify a list of effects within parentheses, then the starting model consists of these specified effects. The effects that you specify in the START= option must be explanatory effects defined in the MODEL statement before the slash (/). The START= option is not available when you specify METHOD=BACKWARD, METHOD=LAR, or METHOD=LASSO in the SELECTION statement.

STB

produces standardized regression coefficients. A standardized regression coefficient is computed by dividing a parameter estimate by the ratio of the sample standard deviation of the dependent variable to the sample standard deviation of the regressor.

TOL

produces tolerance values for the estimates. Tolerance for a parameter is defined as $1 - R^2$, where R^2 is obtained from the regression of the parameter on all other parameters in the model. The TOL option is not supported when you request METHOD=LAR or METHOD=LASSO in the SELECTION statement.

VIF

produces variance inflation factors with the parameter estimates. Variance inflation is the reciprocal of tolerance. The VIF option is not supported when you request METHOD=LAR or METHOD=LASSO in the SELECTION statement.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* >
< **COPYVARS=***(variables)* >
< *keyword* < *=name* > >. . . < *keyword* < *=name* > > **;**

The OUTPUT statement creates a data set that contains observationwise statistics, which are computed after fitting the model. The variables in the input data set are *not* included in the output data set to avoid data duplication for large data sets; however, variables specified in the ID statement or COPYVARS= option are included.

The output statistics are computed based on the parameter estimates for the selected model.

You can specify the following syntax elements in the OUTPUT statement:

OUT=*SAS-data-set*

DATA=*SAS-data-set*

specifies the name of the output data set. If the OUT= (or DATA=) option is omitted, the procedure uses the DATA*n* convention to name the output data set.

COPYVAR=*variable*

COPYVARS=(*variables***)**

transfers one or more *variables* from the input data set to the output data set. Variables named in an ID statement are also copied from the input data set to the output data set.

keyword **<** *=name* **>**

specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. Specify a keyword for each desired statistic (see the following list of keywords), followed optionally by an equal sign and a variable to contain the statistic.

If you specify *keyword=name*, the new variable that contains the requested statistic has the specified name. If you omit the optional *=name* after a *keyword*, then a default name is used.

The following are valid values for *keyword* to request statistics that are available with all selection methods:

PREDICTED

PRED

P

requests predicted values for the response variable. The default name is Pred.

RESIDUAL

RESID

R

requests the residual, calculated as ACTUAL–PREDICTED. The default name is Residual.

ROLE

requests a numeric variable that indicates the role played by each observation in fitting the model. The default name is _ROLE_. For each observation the interpretation of this variable is shown in Table 67.3:

Table 67.3 Role Interpretation

Value	Observation Role
0	Not used
1	Training
2	Validation
3	Testing

If you do not partition the input data by using a PARTITION statement, then the role variable value is 1 for observations used in fitting the model, and 0 for observations that have at least one missing or invalid value for the response, regressors, frequency or weight variables.

In addition to the preceding statistics, you can also use the *keywords* listed in Table 67.4 in the OUTPUT statement to obtain additional statistics. These statistics are not available if you use METHOD=LAR or METHOD=LASSO in the SELECTION statement, unless you also specify the LSCOEFFS option. See the section "Diagnostic Statistics" on page 5150 for computational formulas. All the statistics available in the OUTPUT statement are conditional on the selected model and do not take into account the variability introduced by doing model selection.

Table 67.4 Keywords for OUTPUT Statement

Keyword	Description
COOKD	Cook's D influence statistic
COVRATIO	Standard influence of observation on covariance of betas
DFFIT	Standard influence of observation on predicted value
H	Leverage, $\mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-}\mathbf{x}_i'$
LCL	Lower bound of a $100(1-\alpha)\%$ confidence interval for an individual prediction. This includes the variance of the error, as well as the variance of the parameter estimates.
LCLM	Lower bound of a $100(1-\alpha)\%$ confidence interval for the expected value (mean) of the dependent variable
PRESS	ith residual divided by $(1-h)$, where h is the leverage, and where the model has been refit without the ith observation
RSTUDENT	A studentized residual with the current observation deleted
STDI	Standard error of the individual predicted value
STDP	Standard error of the mean predicted value
STDR	Standard error of the residual
STUDENT	Studentized residuals, which are the residuals divided by their standard errors

Table 67.4 *continued*

Keyword	Description
UCL	Upper bound of a $100(1-\alpha)\%$ confidence interval for an individual prediction
UCLM	Upper bound of a $100(1-\alpha)\%$ confidence interval for the expected value (mean) of the dependent variable

PARTITION Statement

PARTITION < *partition-options* > **;**

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. Either you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations for each role.

The following mutually exclusive *partition-options* are available:

ROLEVAR | ROLE=*variable***(< TEST='***value***' > < TRAIN='***value***' > < VALIDATE='***value***' >)**
names the variable in the input data set whose values are used to assign roles to each observation. The formatted values of this variable that are used to assign observations roles are specified in the TEST=, TRAIN=, and VALIDATE= suboptions. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboptions are assigned to training.

FRACTION(< TEST=*fraction* > < **VALIDATE=***fraction* >)**
requests that specified proportions of the observations in the input data set be randomly assigned training and validation roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and the VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing, and requests detailed results about the performance characteristics of the HPREG procedure.

The PERFORMANCE statement is documented further in the section "PERFORMANCE Statement" on page 569.

SELECTION Statement

SELECTION < *options* > **;**

The SELECTION statement performs variable selection. All *options* except the SCREEN option are fully documented in the section "SELECTION Statement" on page 576. The SCREEN option is described in the following section. The remainder of this section describes specific information about how PROC HPREG implements the METHOD= option and the DETAILS= option.

The HPREG procedure supports the following values of the METHOD= option in the SELECTION statement:

NONE specifies no model selection.

FORWARD specifies the forward selection method, which starts with no effects in the model and adds effects.

BACKWARD specifies the backward elimination method, which starts with all effects in the model and deletes effects.

STEPWISE specifies the stepwise regression method, which is similar to the forward selection method except that effects already in the model do not necessarily stay there.

FORWARDSWAP specifies the forward-swap selection method, which is an extension of the forward selection method. Before any addition step, PROC HPREG makes all pairwise swaps of effects in and out of the current model that improve the selection criterion. When the selection criterion is R square, this method is the same as the MAXR method in the REG procedure in SAS/STAT software.

LAR specifies the least angle regression method. Like forward selection, this method starts with no effects in the model and adds effects. The parameter estimates at any step are "shrunk" when compared to the corresponding least squares estimates. If the model contains classification variables, then these classification variables are split. For more information, see the SPLIT option in the CLASS statement.

LASSO specifies the lasso method, which adds and deletes parameters based on a version of ordinary least squares in which the sum of the absolute regression coefficients is constrained. If the model contains classification variables, then these classification variables are split. For more information, see the option in the SPLIT CLASS statement.

The DETAILS=ALL and DETAILS=STEPS options produce the "ANOVA," "Fit Statistics," and "Parameter Estimates" tables, which provide information about the model that is selected at each step of the selection process.

In addition to other options, which are fully documented in the section "SELECTION Statement" on page 576, PROC HPREG also supports a SCREEN option, which has the following syntax:

SCREEN < **(***global-screen-options***)** > < **=***screen-options* >

You can specify following *global-screen-options*:

DETAILS=NONE | SUMMARY | ALL

specifies the level of detail to be produced about the screening process. You can specify the following values:

NONE suppresses all tables that provide details of the screening process.

ALL produces the following output and shows model selection details at each stage of the screening process:

- a screening table that shows the correlations that are used to obtain the screened effects for the first two stages of the screening process
- a screened effects table that lists the effects that are chosen at each stage of the screening process

SUMMARY produces the following output and shows details about the model selection only for the final stage of the screening process:

- a screening table that shows the correlations that are used to obtain the screened effects for the first two stages of the screening process
- a screened effects table that lists the effects that are chosen at each stage of the screening process

By default, DETAILS=SUMMARY.

SINGLESTAGE

screens effects and selects a model only once. If you do not specify this option, then screening is done in multiple stages.

MULTISTAGE

performs multiple stages, each of which contains a screening and a model selection step.

You can specify the following *screen-options* after an = sign:

SCREEN=*n1* **<** *n2* **>**

specifies the number of effects to be chosen at the first two stages of the screening process. If you specify only *n1*, then *n1* is used for both the first and second stages. If you specify both *n1* and *n2*, then *n1* is used at the first stage and *n2* is used at the second stage. At the first stage, effects are ranked in decreasing order of the magnitude of their pairwise correlations with the response, and the first *n1* effects are used in the selection process at that stage. At the second stage, effects are ranked in decreasing order of the magnitude of their pairwise correlations with the residuals obtained at the first stage, and the first *n2* effects are used in the selection process at that stage.

SCREEN=PERCENT(*p1* **<** *p2* **>)**

specifies the percentage of effects in the MODEL statement to be chosen at the first two stages of the screening process. If you specify only *p1*, then *p1* is used for both the first and second stages. If you specify *p1* and *p2*, then *p1* is used at the first stage and *p2* is used at the second stage.

SCREEN=CUTOFF(*c1* < *c2* >)

specifies the minimum value of the screening statistic that effects must have in order to be chosen at the first two stages of the screening process. If you specify only *c1*, then *c1* is used for both the first and second stages. If you specify both *c1* and *c2*, then *c1* is used at the first stage and *c2* is used at the second stage. At the first stage, any effect whose absolute pairwise correlation with the response is less than the first-stage cutoff is not used in the selection process at that stage. At the second stage, any effect whose absolute pairwise correlation with the residuals obtained from the first stage is less than the second-stage cutoff is not used in the selection process at that stage.

If you do not specify any *screen-options*, SCREEN=PERCENT(10) by default.

For a classification effect that has multiple degrees of freedom, pairwise correlations with the response at the first stage and the first stage residuals at the second stage are computed separately for each dummy variable that corresponds to the levels of the classification variables in the effect. The largest magnitude of these correlations is used as a proxy for the correlation statistic for that effect.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations with nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, all observations used in the analysis are assigned a weight of 1.

Details: HPREG Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpregdt.sas`.)

Criteria Used in Model Selection

The HPREG procedure supports a variety of fit statistics that you can specify as criteria for the CHOOSE=, SELECT=, and STOP= options in the SELECTION statement. The following statistics are available:

ADJRSQ	Adjusted R-square statistic (Darlington 1968; Judge et al. 1985)
AIC	Akaike's information criterion (Akaike 1969; Judge et al. 1985)
AICC	Corrected Akaike's information criterion (Hurvich and Tsai 1989)
BIC | SBC	Schwarz Bayesian information criterion (Schwarz 1978; Judge et al. 1985)
CP	Mallows C_p statistic (Mallows 1973; Hocking 1976)
PRESS	Predicted residual sum of squares statistic
RSQUARE	R-square statistic (Darlington 1968; Judge et al. 1985)

SL	Significance used to assess an effect's contribution to the fit when it is added to or removed from a model
VALIDATE	Average square error over the validation data

When you use SL as a criterion for effect selection, the definition depends on whether an effect is being considered as a drop or an add candidate. If the current model has p parameters excluding the intercept, and if you denote its residual sum of squares by RSS_p and you add an effect with k degrees of freedom and denote the residual sum of squares of the resulting model by RSS_{p+k}, then the F statistic for entry with k numerator degrees of freedom and $n-(p+k)-1$ denominator degrees of freedom is given by

$$F = \frac{(\mathrm{RSS}_p - \mathrm{RSS}_{p+k})/k}{\mathrm{RSS}_{p+k}/(n-(p+k)-1)}$$

where n is number of observations used in the analysis. The significance level for entry is the p-value of this F statistic, and is deemed significant if it is smaller than the SLENTRY limit. Among several such add candidates, the effect with the smallest p-value (most significant) is deemed best.

If you drop an effect with k degrees of freedom and denote the residual sum of squares of the resulting model by RSS_{p-k}, then the F statistic for removal with k numerator degrees of freedom and $n-p-k$ denominator degrees of freedom is given by

$$F = \frac{(\mathrm{RSS}_{p-k} - \mathrm{RSS}_p)/k}{\mathrm{RSS}_p/(n-p-k)}$$

where n is number of observations used in the analysis. The significance level for removal is the p-value of this F statistic, and the effect is deemed not significant if this p-value is larger than the SLSTAY limit. Among several such removal candidates, the effect with the largest p-value (least significant) is deemed the best removal candidate.

It is known that the "F-to-enter" and "F-to-delete" statistics do not follow an F distribution (Draper, Guttman, and Kanemasu 1971).. Hence the SLENTRY and SLSTAY values cannot reliably be viewed as probabilities. One way to address this difficulty is to replace hypothesis testing as a means of selecting a model with information criteria or out-of-sample prediction criteria. While Harrell (2001) points out that information criteria were developed for comparing only prespecified models, Burnham and Anderson (2002) note that AIC criteria have routinely been used for several decades for performing model selection in time series analysis.

Table 67.5 provides formulas and definitions for these fit statistics.

Table 67.5 Formulas and Definitions for Model Fit Summary Statistics

Statistic	Definition or Formula
n	Number of observations
p	Number of parameters including the intercept
$\hat{\sigma}^2$	Estimate of pure error variance from fitting the full model
SST	Total sum of squares corrected for the mean for the dependent variable
SSE	Error sum of squares
ASE	$\frac{\mathrm{SSE}}{n}$

Table 67.5 *continued*

Statistic	Definition or Formula
MSE	$\frac{\text{SSE}}{n-p}$
R^2	$1-\frac{\text{SSE}}{\text{SST}}$
ADJRSQ	$1-\frac{(n-1)(1-R^2)}{n-p}$
AIC	$n\ln\left(\frac{\text{SSE}}{n}\right)+2p$
AICC	$1+\ln\left(\frac{\text{SSE}}{n}\right)+\frac{2(p+1)}{n-p-2}$
CP (C_p)	$\frac{\text{SSE}}{\hat{\sigma}^2}+2p-n$
PRESS	$\sum_{i=1}^{n}\frac{r_i^2}{(1-h_i)^2}$ where r_i = residual at observation i and h_i = leverage of observation $i = \mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-}\mathbf{x}_i'$
RMSE	$\sqrt{\text{MSE}}$
SBC	$n\ln\left(\frac{\text{SSE}}{n}\right)+p\ln(n)$

Diagnostic Statistics

This section gathers the formulas for the statistics available in the OUTPUT statement. All the statistics available in the OUTPUT statement are conditional on the selected model and do not take into account the variability introduced by doing model selection.

The model to be fit is $\mathbf{Y}=\mathbf{X}\boldsymbol{\beta}+\boldsymbol{\epsilon}$, and the parameter estimate is denoted by $\mathbf{b}=(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y}$. The subscript i denotes values for the ith observation, and the parenthetical subscript (i) means that the statistic is computed by using all observations except the ith observation.

The ALPHA= option in the PROC HPREG statement is used to set the α value for the confidence limit statistics.

Table 67.6 contains the diagnostic statistics and their formulas. Each statistic is computed for each observation.

Table 67.6 Formulas and Definitions for Diagnostic Statistics

MODEL Option or Statistic	Formula
PRED ($\widehat{\mathbf{Y}}_i$)	$\mathbf{X}_i\mathbf{b}$
RES (r_i)	$\mathbf{Y}_i-\widehat{\mathbf{Y}}_i$
H (h_i)	$\mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-}\mathbf{x}_i'$
STDP	$\sqrt{h_i\widehat{\sigma}^2}$

Table 67.6 *continued*

STDI	$\sqrt{(1+h_i)\widehat{\sigma}^2}$
STDR	$\sqrt{(1-h_i)\widehat{\sigma}^2}$
LCL	$\widehat{Y}_i - t_{\frac{\alpha}{2}}\text{STDI}$
LCLM	$\widehat{Y}_i - t_{\frac{\alpha}{2}}\text{STDP}$
UCL	$\widehat{Y}_i + t_{\frac{\alpha}{2}}\text{STDI}$
UCLM	$\widehat{Y}_i + t_{\frac{\alpha}{2}}\text{STDP}$
STUDENT	$\frac{r_i}{\text{STDR}_i}$
RSTUDENT	$\frac{r_i}{\hat{\sigma}_{(i)}\sqrt{1-h_i}}$
COOKD	$\frac{1}{p}\text{STUDENT}^2\frac{\text{STDP}^2}{\text{STDR}^2}$
COVRATIO	$\frac{\det(\hat{\sigma}^2_{(i)}(\mathbf{x}'_{(i)}\mathbf{x}_{(i)})^{-1}}{\det(\hat{\sigma}^2(\mathbf{X}'\mathbf{X})^{-1})}$
DFFITS	$\frac{(\widehat{\mathbf{Y}}_i - \widehat{\mathbf{Y}}_{(i)})}{(\hat{\sigma}_{(i)}\sqrt{h_i})}$
PRESS(predr$_i$)	$\frac{r_i}{1-h_i}$

Classification Variables and the SPLIT Option

PROC HPREG supports the ability to split classification variables when doing model selection. You use the SPLIT option in the CLASS statement to specify that the columns of the design matrix that correspond to effects that contain a split classification variable can enter or leave a model independently of the other design columns of that effect. The following statements illustrate the use of SPLIT option:

```
data splitExample;
   length c2 $6;
   drop i;
   do i=1 to 1000;
      c1 = 1 + mod(i,6);
      if      i < 250 then c2 = 'low';
      else if i < 500 then c2 = 'medium';
      else                 c2 = 'high';
      x1 = ranuni(1);
      x2 = ranuni(1);
      y = x1+3*(c2 ='low')  + 10*(c1=3) +5*(c1=5) + rannor(1);
      output;
   end;
run;

proc hpreg data=splitExample;
   class c1(split) c2(order=data);
   model y = c1 c2 x1 x2/orderselect;
   selection method=forward;
```

```
run;
```

The "Class Levels" table shown in Figure 67.9 is produced by default whenever you specify a CLASS statement.

Figure 67.9 Class Levels

The HPREG Procedure

Class Level Information		
Class	**Levels**	**Values**
c1	6	* 1 2 3 4 5 6
c2	3	low medium high

*** Associated Parameters Split**

The SPLIT option has been specified for the classification variable c1. This permits the parameters associated with the effect c1 to enter or leave the model individually. The "Parameter Estimates" table in Figure 67.10 shows that for this example the parameters that correspond to only levels 3 and 5 of c1 are in the selected model. Finally, note that the ORDERSELECT option in the MODEL statement specifies that the parameters be displayed in the order in which they first entered the model.

Figure 67.10 Parameter Estimates

Parameter Estimates					
Parameter	**DF**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	-0.308111	0.075387	-4.09	<.0001
c1_3	1	10.161702	0.087601	116.00	<.0001
c1_5	1	5.018407	0.087587	57.30	<.0001
c2 low	1	3.139941	0.078495	40.00	<.0001
c2 medium	1	0.221539	0.078364	2.83	0.0048
c2 high	0	0	.	.	.
x1	1	1.317420	0.109510	12.03	<.0001

Using Validation and Test Data

When you have sufficient data, you can subdivide your data into three parts called the training, validation, and test data. During the selection process, models are fit on the training data, and the prediction error for the models so obtained is found by using the validation data. This prediction error on the validation data can be used to decide when to terminate the selection process or to decide what effects to include as the selection process proceeds. Finally, after a selected model has been obtained, the test set can be used to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases you might want to use only training and test data. For example, you might decide to use an information criterion to decide what effects to include and when to terminate the selection process. In this case no validation data are required, but test data can still be useful in assessing the predictive performance of the selected model. In other cases you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) note that it is

difficult to give a general rule for how many observations you should assign to each role. They note that a typical split might be 50% for training and 25% each for validation and testing.

You use a PARTITION statement to logically subdivide the DATA= data set into separate roles. You can name the fractions of the data that you want to reserve as test data and validation data. For example, the following statements randomly subdivide the "inData" data set, reserving 50% for training and 25% each for validation and testing:

```
proc hpreg data=inData;
   partition fraction(test=0.25 validate=0.25);
   ...
run;
```

In some cases you might need to exercise more control over the partitioning of the input data set. You can do this by naming both a variable in the input data set and also a formatted value of that variable that correspond to each role. For example, the following statements assign roles to the observations in the "inData" data set based on the value of the variable group in that data set. Observations where the value of group is 'group 1' are assigned for testing, and those with value 'group 2' are assigned to training. All other observations are ignored.

```
proc hpreg data=inData;
   partition roleVar=group(test='group 1' train='group 2')
   ...
run;
```

When you have reserved observations for training, validation, and testing, a model fit on the training data is scored on the validation and test data, and the average squared error (ASE) is computed separately for each of these subsets. The ASE for each data role is the error sum of squares for observations in that role divided by the number of observations in that role.

Using the Validation ASE as the STOP= Criterion

If you have provided observations for validation, then you can specify STOP=VALIDATE as a suboption of the METHOD= option in the SELECTION statement. At step k of the selection process, the best candidate effect to enter or leave the current model is determined. Here "best candidate" means the effect that gives the best value of the SELECT= criterion; this criterion need not be based on the validation data. The validation ASE for the model with this candidate effect added or removed is computed. If this validation ASE is greater than the validation ASE for the model at step k, then the selection process terminates at step k.

Using the Validation ASE as the CHOOSE= Criterion

When you specify the CHOOSE=VALIDATE suboption of the METHOD= option in the SELECTION statement, the validation ASE is computed for the models at each step of the selection process. The smallest model at any step that yields the smallest validation ASE is selected.

Using the Validation ASE as the SELECT= Criterion

You request the validation ASE as the selection criterion by specifying the SELECT=VALIDATE suboption of the METHOD= option in the SELECTION statement. At step *k* of the selection process, the validation ASE is computed for each model in which a candidate for entry is added or candidate for removal is dropped. The selected candidate for entry or removal is the one that yields a model with the minimal validation ASE. This method is computationally very expensive because validation statistics need to be computed for every candidate at every step; it should be used only with small data sets or models with a small number of regressors.

Computational Method

Multithreading

Threading refers to the organization of computational work into multiple tasks (processing units that can be scheduled by the operating system). A task is associated with a thread. Multithreading refers to the concurrent execution of threads. When multithreading is possible, substantial performance gains can be realized compared to sequential (single-threaded) execution.

The number of threads spawned by the HPREG procedure is determined by the number of CPUs on a machine and can be controlled in the following ways:

- You can specify the CPU count with the CPUCOUNT= SAS system option. For example, if you specify the following statements, the HPREG procedure schedules threads as if it executes on a system with four CPUs, regardless of the actual CPU count.

```
options cpucount=4;
```

- You can specify the NTHREADS= option in the PERFORMANCE statement to determine the number of threads. This specification overrides the system option. Specify NTHREADS=1 to force single-threaded execution.

The number of threads per machine is displayed in the “Performance Information” table, which is part of the default output. The HPREG procedure allocates one thread per CPU.

The tasks multithreaded by the HPREG procedures are primarily defined by dividing the data processed on a single machine among the threads—that is, the HPREG procedure implements multithreading through a data-parallel model. For example, if the input data set has 1,000 observations and you are running with four threads, then 250 observations are associated with each thread. All operations that require access to the data are then multithreaded. This operations include the following:

- variable levelization
- effect levelization
- formation of the crossproducts matrix
- evaluation of predicted residual sums of squares on validation and test data

- scoring of observations

In addition, operations on matrices such as sweeps might be multithreaded if the matrices are of sufficient size to realize performance benefits from managing multiple threads for the particular matrix operation.

Output Data Set

Many procedures in SAS software add the variables from the input data set when an observationwise output data set is created. The assumption of high-performance statistical procedures is that the input data sets can be large and contain many variables. For performance reasons, the output data set contains the following:

- those variables explicitly created by the statement
- variables listed in the ID statement
- distribution keys or hash keys that are transferred from the input data set

This enables you to add output data set information that is necessary for subsequent SQL joins without copying the entire input data set to the output data set. For more information see the section "Output Data Sets" on page 568.

Screening

Model selection from a very large number of effects is computationally demanding. For example, in analyzing microarray data, where each dot in the array corresponds to a regressor, having 35,000 such regressors is not uncommon. Another source of such large regression problems arises when you want to consider all possible two-way interactions of your main effects as candidates for inclusion in a selected model. See Foster and Stine (2004) for an example that uses this approach to build a predictive model for bankruptcy.

In recent years, there has been a resurgence of interest in combining variable selection methods with an initial screening step that reduces the large number of regressors to a much smaller subset from which the final model is chosen. You can find theoretical underpinnings of this approach in Fan and Lv (2008). See El Ghaoui, Viallon, and Rabbani (2012) and Tibshirani et al. (2012) for examples where screening has also been incorporated in the context of penalized regression methods (such as lasso) for performing model selection.

Screening uses a screening statistic that is inexpensive to compute in order to eliminate from consideration regressors that are unlikely to be selected if you included them in variable selection. For linear regression, you can use the magnitude of the correlation between each individual regressor and the response as such a screening statistic. The square of the correlation between a regressor that has one degree of freedom and the response is the R-square value for the univariate regression for the response with this regressor. Hence, screening by the magnitude of the pairwise correlations is equivalent to fitting univariate models to do the screening.

The first stage of the screening method chooses only the subset of regressors whose screening statistic is larger than a specified cutoff value or by choosing those regressors whose screening statistics are among a

specified number or percentage of the largest screening statistic values. Then you perform model selection for the response from this screened subset of the original regressors.

One problem with this approach is that a regressor that is pairwise (marginally) uncorrelated or has very small correlation with the response can nevertheless be an important predictor, but it would be eliminated in the screening. You can address this problem by switching to a multistage approach. The first stage consists of screening the regressors and selecting the model for the response from the screened subset. The second stage repeats the first stage except that you use the residuals from the first stage as the response variable in this second stage. You can iterate this process by using the residuals from the previous stage as the response for the next stage. The final stage forms the union of all the screened regressors from the first stage with all the selected regressors at the subsequent stages and selects a model for the original response variable from this union.

Experimentation has shown that there is little benefit in practice in using more than one stage where the response is the residual from the previous stage. Hence, PROC HPREG implements a three-stage process by default. However, if you specify the SINGLESTAGE suboption in the SCREEN option in the SELECTION statement, then only the first screening stage is performed.

Displayed Output

The following sections describe the output produced by PROC HPREG. The output is organized into various tables, which are discussed in the order of appearance.

Performance Information

The “Performance Information” table is produced by default. It displays the number of threads used.

Data Access Information

The “Data Access Information” table is produced by default. For the input and output data sets, it displays the libref and data set name, the engine used to access the data, the role (input or output) of the data set, and path that data followed to the reach the computation.

Model Information

The “Model Information” table displays basic information about the model, such as the response variable, frequency variable, weight variable, and the type of parameterization used for classification variables named in the CLASS statement.

Selection Information

When you specify the SELECTION statement, the HPREG procedure produces by default a series of tables with information about the model selection. The “Selection Information” table informs you about the model selection method; select, stop, and choose criteria; and other parameters that govern the selection. You can suppress this table by specifying DETAILS=NONE in the SELECTION statement.

Screening Information

When you specify the SCREEN option in the SELECTION statement, the "Screening Information" table informs you about the number of screening stages used and informs you about the method and values that are used to determine how many screened effects are chosen at each screening stage.

Screening

When you specify the DETAILS=ALL suboption of the SCREEN option in the SELECTION statement, the "Screening" table displays the model effects and their screening statistic values in descending order of the screening statistic values.

Screened Effects

When you specify the SCREEN option in the SELECTION statement, the "Screened Effects" table displays a list of the screened model effects at each stage of the screening process.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set and the number of observations used in the analysis. If you specify a FREQ statement, the table also displays the sum of frequencies read and used. If you use a PARTITION statement, the table also displays the number of observations used for each data role.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. You should check this information to make sure that the data are correct. You can adjust the order of the CLASS variable levels with the ORDER= option in the CLASS statement. You can suppress the "Class Level Information" table completely or partially with the NOCLPRINT= option in the PROC HPREG statement.

If the classification variables are in the reference parameterization, the "Class Level Information" table also displays the reference value for each variable. The "Class Level Information" table also indicates which, if any, of the classification variables are split by using the SPLIT option in the CLASS statement.

Dimensions

The "Dimensions" table displays information about the number of effects and the number of parameters from which the selected model is chosen. If you use split classification variables, then this table also includes the number of effects after splitting is taken into account.

Entry and Removal Candidates

When you specify the DETAILS=ALL or DETAILS=STEPS option in the SELECTION statement, the HPREG procedure produces "Entry Candidates" and "Removal Candidates" tables that display the effect names and values of the criterion used to select entering or departing effects at each step of the selection process. The effects are displayed in sorted order from best to worst of the selection criterion.

Selection Summary

When you specify the SELECTION statement, the HPREG procedure produces the "Selection Summary" table with information about the sequence of steps of the selection process. For each step, the effect that was entered or dropped is displayed along with the statistics used to select the effect, stop the selection, and choose the selected model. For all criteria that you can use for model selection, the steps at which the optimal values of these criteria occur are also indicated.

The display of the "Selection Summary" table can be suppressed by specifying DETAILS=NONE in the SELECTION statement.

Stop Reason

The "Stop Reason" table displays the reason why the selection stopped. To facilitate programmatic use of this table, an integer code is assigned to each reason and is included if you output this table by using an ODS OUTPUT statement. The reasons and their associated codes follow:

Code	Stop Reason
1	All eligible effects are in the model.
2	All eligible effects have been removed.
3	Specified maximum number of steps done.
4	The model contains the specified maximum number of effects.
5	The model contains the specified minimum number of effects (for backward selection).
6	The stopping criterion is at a local optimum.
7	No suitable add or drop candidate could be found.
8	Adding or dropping any effect does not improve the selection criterion.
9	No candidate meets the appropriate SLE or SLS significance level.
10	Stepwise selection is cycling.
11	The model is an exact fit.
12	Dropping an effect would result in an empty model.

The display of the "Stop Reason" table can be suppressed by specifying DETAILS=NONE in the SELECTION statement.

Selection Reason

When you specify the SELECTION statement, the HPREG procedure produces a simple table that contains text informing you about the reason why the final model was selected.

The display of the "Selection Reason" table can be suppressed by specifying DETAILS=NONE in the SELECTION statement.

Selected Effects

When you specify the SELECTION statement, the HPREG procedure produces a simple table that contains text informing you about which effects were selected into the final model.

ANOVA

The "ANOVA" table displays an analysis of variance for the selected model. This table includes the following:

- the Source of the variation, Model for the fitted regression, Error for the residual error, and C Total for the total variation after correcting for the mean. The Uncorrected Total Variation is produced when the NOINT option is used.
- the degrees of freedom (DF) associated with the source
- the Sum of Squares for the term
- the Mean Square, the sum of squares divided by the degrees of freedom
- the F Value for testing the hypothesis that all parameters are 0 except for the intercept. This is formed by dividing the mean square for Model by the mean square for Error.
- the Prob>F, the probability of getting a greater F statistic than that observed if the hypothesis is true. When you do model selection, these p-values are generally liberal because they are not adjusted for the fact that the terms in the model have been selected.

You can request "ANOVA" tables for the model at each step of the selection process with the DETAILS= option in the SELECTION statement.

Fit Statistics

The "Fit Statistics" table displays fit statistics for the selected model. The statistics displayed include the following:

- Root MSE, an estimate of the standard deviation of the error term. It is calculated as the square root of the mean square error.
- R-square, a measure between 0 and 1 that indicates the portion of the (corrected) total variation attributed to the fit rather than left to residual error. It is calculated as SS(Model) divided by SS(Total). It is also called the *coefficient of determination*. It is the square of the multiple correlation—in other words, the square of the correlation between the dependent variable and the predicted values.
- Adj R-Sq, the adjusted R-square, a version of R-square that has been adjusted for degrees of freedom. It is calculated as

$$\bar{R}^2 = 1 - \frac{(n-i)(1-R^2)}{n-p}$$

  where i is equal to 1 if there is an intercept and 0 otherwise, n is the number of observations used to fit the model, and p is the number of parameters in the model.

- fit criteria AIC, AICC, BIC, CP, and PRESS if they are used in the selection process. See Table 67.5 for the formulas for evaluating these criteria.
- the average square errors (ASE) on the training, validation, and test data.

You can request "Fit Statistics" tables for the model at each step of the selection process with the DETAILS= option in the SELECTION statement.

Parameter Estimates

The "Parameter Estimates" table displays the parameters in the selected model and their estimates. The information displayed for each parameter in the selected model includes the following:

- the parameter label that includes the effect name and level information for effects that contain classification variables
- the degrees of freedom (DF) for the parameter. There is one degree of freedom unless the model is not full rank.
- the parameter estimate
- the standard error, which is the estimate of the standard deviation of the parameter estimate
- t Value, the t test that the parameter is 0. This is computed as the parameter estimate divided by the standard error.
- the Pr > |t|, the probability that a t statistic would obtain a greater absolute value than that observed given that the true parameter is 0. This is the two-tailed significance probability.

  When you do model selection, these p-values are generally liberal because they are not adjusted for the fact that the terms in the model have been selected.

You can request "Parameter Estimates" tables for the model at each step of the selection process with the DETAILS= option in the SELECTION statement.

Timing Information

If you specify the DETAILS option in the PERFORMANCE statement, the procedure also produces a "Timing" table in which elapsed time (absolute and relative) for the main tasks of the procedure are displayed.

ODS Table Names

Each table created by the HPREG procedure has a name associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 67.7.

Table 67.7 ODS Tables Produced by PROC HPREG

Table Name	Description	Required Statement / Option
ANOVA	Selected model ANOVA table	Default output
Candidates	Swap candidates at step	SELECTION DETAILS=ALL|STEPS
ClassLevels	Level information from the CLASS statement	CLASS
DataAccessInfo	Information about modes of data access	Default output
Dimensions	Model dimensions	Default output
EntryCandidates	Candidates for entry at step	SELECTION DETAILS=ALL|STEPS
FitStatistics	Fit statistics	Default output
ModelInfo	Information about the modeling environment	Default output
NObs	Number of observations read and used	Default output
ParameterEstimates	Solutions for the parameter estimates associated with effects in MODEL statement	Default output
PerformanceInfo	Information about high-performance computing environment	Default output
RemovalCandidates	Candidates for removal at step	SELECTION DETAILS=ALL|STEPS
ScreenedEffects	List of screened effects	SELECTION SCREEN
ScreeningInfo	Information about the screening method	SELECTION SCREEN
Screening	Screening statistic values for model effects	SELECTION SCREEN(DETAILS=ALL)
SelectedEffects	List of selected effects	SELECTION
SelectionInfo	Information about selection settings	Default output
SelectionReason	Reason for selecting the final model	SELECTION
SelectionSummary	Summary information about the model selection steps	SELECTION
StopReason	Reason selection was terminated	SELECTION
Timing	Timing breakdown by task	SELECTION DETAILS

Examples: HPREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 67.1: Model Selection with Validation

(View the complete code for this example (hpregex1.sas) in the example repository.)

This example is based on the example "Using Validation and Cross Validation" in the documentation for the GLMSELECT procedure in the *SAS/STAT User's Guide*. This example shows how you can use validation data to monitor and control variable selection. It also demonstrates the use of split classification variables.

The following DATA step produces analysis data that contains a variable that you can use to assign observations to the training, validation, and testing roles. In this case, each role has 5,000 observations.

```
data analysisData;
   drop i j c3Num;
   length c3$ 7;

   array x{20} x1-x20;

   do i=1 to 15000;
      do j=1 to 20;
         x{j} = ranuni(1);
      end;

      c1 = 1 + mod(i,8);
      c2 = ranbin(1,3,.6);

      if      i < 50   then do; c3 = 'tiny';     c3Num=1;end;
      else if i < 250  then do; c3 = 'small';    c3Num=1;end;
      else if i < 600  then do; c3 = 'average';  c3Num=2;end;
      else if i < 1200 then do; c3 = 'big';      c3Num=3;end;
      else                  do; c3 = 'huge';     c3Num=5;end;

      yTrue = 10 + x1 + 2*x5 + 3*x10 + 4*x20  + 3*x1*x7 + 8*x6*x7
                 + 5*(c1=3)*c3Num + 8*(c1=7);

      error = 5*rannor(1);

      y = yTrue + error;

           if mod(i,3)=1 then Role = 'TRAIN';
      else if mod(i,3)=2 then Role = 'VAL';
      else                    Role = 'TEST';

      output;
   end;
```

```
run;
```

By construction, the true model consists of main effects x1, x5, x10, x20, and c1 and interaction effects x1*x7, x6*x7, and c1*c3. Furthermore, you can see that only levels 3 and 7 of the classification variable c1 are systematically related to the response.

Because the error term for each observation is five times a value drawn from a standard normal distribution, the expected error variance is 25. For the data in each role, you can compute an estimate of this error variance by forming the average square error (ASE) for the observations in the role. Output 67.1.1 shows the ASE for each role that you can compute with the following statements:

```
proc summary data=analysisData;
   class role;
   ways 1;
   var error;
   output out=ASE uss=uss n=n;
run;
data ASE; set ASE;
   OracleASE = uss / n;
   label OracleASE = 'Oracle ASE';
   keep Role OracleASE;
run;
proc print data=ASE label noobs;
run;


proc print data=ASE label noobs;
run;
```

Output 67.1.1 Oracle ASE Values by Role

Role	Oracle ASE
TEST	25.5784
TRAIN	25.4008
VAL	25.8993

The ASE values shown Output 67.1.1 are labeled as "Oracle ASE" because you need to know the true underlying model if you want to compute these values from the response and underlying regressors. In a modeling context, a good predictive model produces values that are close to these oracle values. An overfit model produces a smaller ASE on the training data but higher values on the validation and test data. An underfit model exhibits higher values for all data roles.

Suppose you suspect that the dependent variable depends on both main effects and two-way interactions. You can use the following statements to select a model:

```
proc hpreg data=analysisData;
   partition roleVar=role(train='TRAIN' validate='VAL' test='TEST');
   class c1 c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2 /stb;
   selection method = stepwise(select=sl sle=0.1 sls=0.15 choose=validate)
                      hierarchy=single details=steps;
```

```
run;
```

A PARTITION statement assigns observations to training, validation, and testing roles based on the values of the input variable named role. The SELECTION statement requests STEPWISE selection based on significance level where the SLE and SLS values are set to use the defaults of PROC REG. The CHOOSE=VALIDATE option selects the model that yields the smallest ASE value on the validation data.

The "Number Of Observation" table in Output 67.1.2 confirms that there are 5,000 observations for each data role. The "Dimensions" table shows that the selection is from 278 effects with a total of 661 parameters.

Output 67.1.2 Number of Observations, Class Levels, and Dimensions

The HPREG Procedure

Number of Observations Read	15000
Number of Observations Used	15000
Number of Observations Used for Training	5000
Number of Observations Used for Validation	5000
Number of Observations Used for Testing	5000

Class Level Information

Class	Levels	Values
c1	8	1 2 3 4 5 6 7 8
c2	4	0 1 2 3
c3	5	tiny small average big huge

Dimensions	
Number of Effects	278
Number of Parameters	661

Output 67.1.3 shows the "Selection Summary" table. You see that 18 steps are done, at which point all effects in the model are significant at the SLS value of 0.15 and all the remaining effects if added individually would not be significant at the SLE significance level of 0.1. However, because you have specified the CHOOSE=VALIDATE option, the model at step 18 is not used as the selected model. Instead the model at step 10 (where the validation ASE achieves a local minimum value) is selected. The "Stop Reason," "Selection Reason," and "Selected Effects" in Output 67.1.4 provide this information.

Output 67.1.3 Selection Summary

The HPREG Procedure

Selection Summary

Step	Effect Entered	Number Effects In	Validation ASE	p Value
0	**Intercept**	1	98.3895	1.0000
1	**c1**	2	34.8572	<.0001
2	**x7**	3	32.5531	<.0001
3	**x6**	4	31.0646	<.0001
4	**x20**	5	29.7078	<.0001
5	**x6*x7**	6	29.2210	<.0001
6	**x10**	7	28.6683	<.0001
7	**x1**	8	28.3250	<.0001
8	**x5**	9	27.9766	<.0001
9	**c3**	10	27.8288	<.0001
10	**c1*c3**	11	25.9701*	<.0001
11	**x10*c1**	12	26.0696	0.0109
12	**x4**	13	26.1594	0.0128
13	**x4*x10**	14	26.1814	0.0035
14	**x20*c1**	15	26.3294	0.0156
15	**x1*c3**	16	26.3945	0.0244
16	**x1*x7**	17	26.3632	0.0270
17	**x7*x10**	18	26.4120	0.0313
18	**x1*x20**	19	26.4330	0.0871

*** Optimal Value of Criterion**

Output 67.1.4 Stopping and Selection Reasons

Selection stopped because all candidates for removal are significant at the 0.15 level and no candidate for entry is significant at the 0.1 level.

The model at step 10 is selected where Validation ASE is 25.9701.

Selected Effects: Intercept c1 c3 c1*c3 x1 x5 x6 x7 x6*x7 x10 x20

You can see that the selected effects include all the main effects in the true model and two of the three true interaction terms. Furthermore, the selected model does not include any variables that are not in the true model. Note that these statements are not true of the larger model at the final step of the selection process.

Output 67.1.5 shows the fit statistics of the selected model. You can see that the ASE values on the training, validation, and test data are all similar, which is indicative of a reasonable predictive model. In this case where the true model is known, you can see that all three ASE values are close to oracle values for the true model, as shown in Output 67.1.1.

Output 67.1.5 Fit Statistics for the Selected Model

Root MSE	5.03976
R-Square	0.74483
Adj R-Sq	0.74246
AIC	21222
AICC	21223
SBC	16527
ASE (Train)	25.16041
ASE (Validate)	25.97010
ASE (Test)	25.83436

Because you specified the DETAILS=STEPS option in the SELECTION statement, you can see the "Fit Statistics" for the model at each step of the selection process. Output 67.1.6 shows these fit statistics for final model at step 18. You see that for this model, the ASE value on the training data is smaller than the ASE values on the validation and test data. This is indicative an overfit model that might not generalize well to new data. You see the ASE values on the validation and test data are now worse in comparison to the oracle values than the values for the selected model at step 10.

Output 67.1.6 Fit Statistics for the Model at Step 18

Root MSE	5.01386
R-Square	0.74862
Adj R-Sq	0.74510
AIC	21194
AICC	21196
SBC	16648
ASE (Train)	24.78688
ASE (Validate)	26.43304
ASE (Test)	26.07078

Output 67.1.7 shows part of the "Parameter Estimates" table for the selected model at step 10 that includes the estimates for the main effect c1. Because the STB option is specified in the MODEL statement, this table includes standardized estimates.

Output 67.1.7 Part of the Parameter Estimates Table for the Selected Model

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	9.479114	0	0.422843	22.42	<.0001
c1 1	1	0.279417	0.009306	0.297405	0.94	0.3475
c1 2	1	0.615589	0.020502	0.297332	2.07	0.0385
c1 3	1	25.678601	0.855233	0.297280	86.38	<.0001
c1 4	1	0.420360	0.014000	0.297283	1.41	0.1574
c1 5	1	0.473986	0.015786	0.297265	1.59	0.1109
c1 6	1	0.394044	0.013124	0.297299	1.33	0.1851
c1 7	1	8.469793	0.282089	0.297345	28.48	<.0001
c1 8	0	0	0	.	.	.

The magnitudes of the standardized estimates and the *t* statistics of the parameters of the effect c1 reveal that only levels 3 and 7 of this effect contribute appreciably to the model. This suggests that a more parsimonious model with similar or better predictive power might be obtained if parameters that correspond to the levels of c1 can enter or leave the model independently. You request this with the SPLIT option in the CLASS statement as shown in the following statements:

```
proc hpreg data=analysisData;
   partition roleVar=role(train='TRAIN' validate='VAL' test='TEST');
   class c1(split) c2 c3(order=data);
   model y =  c1|c2|c3|x1|x2|x3|x4|x5|x5|x6|x7|x8|x9|x10
             |x11|x12|x13|x14|x15|x16|x17|x18|x19|x20 @2 /stb;
   selection method = stepwise(select=sl sle=0.1 sls=0.15 choose=validate)
                      hierarchy=single details=steps;
run;
```

Output 67.1.8 shows the "Dimensions" table. You can see that because the columns in the design matrix that correspond to levels of c1 are treated as separate effects, the selection is now from 439 effects, even though the number of parameters is unchanged.

Output 67.1.8 Dimensions with c1 Split

The HPREG Procedure

Dimensions	
Number of Effects	278
Number of Effects after Splits	439
Number of Parameters	661

Output 67.1.9 shows the selected effects. You can see that as anticipated the selected model now depends on only levels 3 and 7 of c1.

Output 67.1.9 Selected Effects with c1 Split

Selected Effects:	Intercept c1_3 c1_7 c3 c1_3*c3 x1 x5 x6 x7 x6*x7 x10 x20

Finally, the fit statistics for the selected model are shown Output 67.1.10.

Output 67.1.10 Fit Statistics for the Selected Model with c1 Split

Root MSE	5.04060
R-Square	0.74325
Adj R-Sq	0.74238
AIC	21195
AICC	21195
SBC	16311
ASE (Train)	25.31622
ASE (Validate)	25.98055
ASE (Test)	25.76059

If you compare the ASE values for this model in Output 67.1.10 with the oracle values in Output 67.1.1 and the values for the model without splitting c1 in Output 67.1.5, you see that this more parsimonious model

produces the best predictive performance on the test data of all the models considered in this example.

Example 67.2: Forward-Swap Selection

(View the complete code for this example (hpregex2.sas) in the example repository.)

This example highlights the use of the forward-swap selection method, which is a generalization of the maximum R-square improvement (MAXR) method that is available in the REG procedure in SAS/STAT software. This example also demonstrates the use of the INCLUDE and START options.

The following DATA step produces the simulated data in which the response y depends on six main effects and three 2-way interactions from a set of 20 regressors.

```
data ex3Data;
   array x{20};
   do i=1 to 10000;
      do j=1 to 20;
         x{j} = ranuni(1);
      end;
      y = 3*x1 + 7*x2 -5*x3 + 5*x1*x3 +
          4*x2*x13 + x7 + x11 -x13  + x1*x4 + rannor(1);
      output;
   end;
run;
```

Suppose you want to find the best model of each size in a range of sizes for predicting the response y. You can use the forward-swap selection method to produce good models of each size without the computational expense of examining all possible models of each size. In this example, the criterion used to evaluate the models of each size is the model R square. With this criterion, the forward-swap method coincides with the MAXR method that is available in the REG procedure in SAS/STAT software. The model of a given size for which no pairwise swap of an effect in the model with any candidate effect improves the R-square value is deemed to be the best model of that size.

Suppose that you have prior knowledge that the regressors x1, x2, and x3 are needed in modeling the response y. Suppose that you also believe that some of the two-way interactions of these variables are likely to be important in predicting y and that some other two-way interactions might also be needed. You can use this prior information by specifying the selection process shown in the following statements:

```
proc hpreg data=ex3Data;
   model y = x1|x2|x3|x4|x5|x6|x7|x8|x9|x10|X11|
             x12|x13|x14|x5|x16|x7|x18|x19|x20@2 /
                 include=(x1 x2 x3) start=(x1*x2 x1*x3 x2*x3);
   selection method=forwardswap(select=rsquare maxef=15 choose=sbc)
             details=all;
run;
```

The MODEL statement specifies that all main effects and two-way interactions are candidates for selection. The INCLUDE= option specifies that the effects x1, x2, and x3 must appear in all models that are examined. The START= option specifies that all the two-way interactions of these variables should be used in the initial model that is considered but that these interactions are eligible for removal during the forward-swap selection.

The “Selection Summary” table is shown in Output 67.2.1.

Output 67.2.1 Selection Summary

The HPREG Procedure

Selection Summary					
Step	Effect Entered	Effect Removed	Number Effects In	SBC	Model R-Square
0	Intercept		1		
	x1		2		
	x2		3		
	x1*x2		4		
	x3		5		
	x1*x3		6		
	x2*x3		7	3307.6836	0.8837
1	x2*x13		8	1892.8403	0.8992
2	x7*x11	x1*x2	8	618.9298	0.9112
3	x1*x4	x2*x3	8	405.3751	0.9131
4	x13		9	213.6140	0.9148
5	x7		10	180.4457	0.9152
6	x11	x7*x11	10	1.4039*	0.9167
7	x10*x11		11	2.3393	0.9168
8	x3*x7		12	4.5000	0.9168
9	x6*x7		13	10.0589	0.9169
10	x3*x6		14	13.1113	0.9169
11	x5*x20		15	19.4612	0.9169
12	x13*x20	x3*x6	15	18.3678	0.9169
13	x5*x5	x6*x7	15	12.1398	0.9170*

*** Optimal Value of Criterion**

You see that starting from the model with an intercept and the effects specified in the INCLUDE= and START= options at step 0, the forward-swap selection method adds the effect x2*x13 at step one, because this yields the maximum improvement in R square that can be obtained by adding a single effect. The forward-swap selection method now evaluates whether any effect swap yields a better eight-effect model (one with a higher R-square value). Because you specified the DETAILS=ALL option in the SELECTION statement, at each step where a swap is made you obtain a "Candidates" table that shows the R-square values for the evaluated swaps. Output 67.2.2 shows the "Candidates" for step 2. By default, only the best 10 swaps are displayed.

Output 67.2.2 Swap Candidates at Step 2

Best 10 Candidates			
Rank	Effect Dropped	Effect Added	R-Square
1	x1*x2	x7*x11	0.9112
2	x2*x3	x7*x11	0.9112
3	x1*x2	x7	0.9065
4	x2*x3	x7	0.9065
5	x1*x2	x7*x7	0.9060
6	x2*x3	x7*x7	0.9060
7	x1*x2	x4*x7	0.9060
8	x2*x3	x4*x7	0.9060
9	x1*x2	x11	0.9058
10	x2*x3	x11	0.9058

You see that the best swap adds x7*x11 and drops x1*x2. This yields an eight-effect model whose R-square value (0.9112) is larger than the R-square value (0.8992) of the eight-effect model at step 1. Hence this swap is made at step 2. At step 3, an even better eight-effect model than the model at step 2 is obtained by dropping x2*x3 and adding x1*x4. No additional swap improves the R-square value, and so the model at step 3 is deemed to be the best eight-effect model. Although this is the best eight-effect model that can be found by this method given the starting model, it is not guaranteed that this model that has the highest R-square value among all possible models that consist of seven effects and an intercept.

Because the DETAILS=ALL option is specified in the SELECTION statement, details for the model at each step of the selection process are displayed. Output 67.2.3 provides details of the model at step 3.

Output 67.2.3 Model Details at Step 3

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	108630	15519	15000.3	<.0001
Error	9992	10337	1.03455		
Corrected Total	9999	118967			

Root MSE	1.01713
R-Square	0.91311
Adj R-Sq	0.91305
AIC	10350
AICC	10350
SBC	405.37511
ASE	1.03373

Output 67.2.3 *continued*

Parameter Estimates					
Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	0.012095	0.045712	0.26	0.7913
x1	1	3.087078	0.076390	40.41	<.0001
x2	1	7.775180	0.046815	166.08	<.0001
x3	1	-4.957140	0.070995	-69.82	<.0001
x1*x3	1	4.910115	0.122503	40.08	<.0001
x1*x4	1	0.890436	0.060523	14.71	<.0001
x7*x11	1	1.708469	0.045939	37.19	<.0001
x2*x13	1	2.584078	0.061506	42.01	<.0001

The forward-swap method continues to find the best nine-effect model, best 10-effect model, and so on until it obtains the best 15-effect model. At this point the selection terminates because you specified the MAXEF=15 option in the SELECTION statement. The R-square value increases at each step of the selection process. However, because you specified the CHOOSE=SBC criterion in the SELECTION statement, the final model selected is the model at step 6.

Example 67.3: Forward Selection with Screening

(View the complete code for this example (hpregex3.sas) in the example repository.)

This example shows how you can use the SCREEN option in the SELECTION statement to greatly speed up model selection from a large number of regressors. In order to demonstrate the efficacy of model selection with screening, this example uses simulated data in which the response y depends systematically on a relatively small subset of a much larger set of regressors, which is described in Table 67.8.

Table 67.8 Complete Set of Regressors

Regressor Name	Type	Number of Levels	In True Model
xIn1–xIn25	Continuous		Yes
xWeakIn1–xWeakIn2	Continuous		Yes
xOut1–xOut500	Continuous		No
cIn1–cIn5	Classification	From two to five	Yes
cOut1–cOut500	Classification	From two to five	No

The labels In and Out, which are part of the variable names, make it easy to identify whether the selected model succeeds or fails in capturing the true underlying model. The regressors that are labeled xWeakIn1 and xWeakIn2 are predictive, but their influence is substantially smaller than the influence of the other regressors in the true model.

The following DATA step generates the data:

```
%let nObs      = 50000;
%let nContIn   = 25;
%let nContOut  = 500;
%let nClassIn  = 5;
%let nClassOut = 500;
%let maxLevs   = 5;
%let noiseScale= 1;

data ex4Data;
   array xIn{&nContIn};
   array xOut{&nContOut};
   array cIn{&nClassIn};
   array cOut{&nClassOut};

   drop i j sign nLevs xBeta;

   do i=1 to &nObs;
      sign  = -1;
      xBeta = 0;
      do j=1 to dim(xIn);
         xIn{j} = ranuni(1);
         xBeta  = xBeta + j*sign*xIn{j};
         sign   = -sign;
      end;
      do j=1 to dim(xOut);
         xOut{j} = ranuni(1);
      end;

      xWeakIn1 = ranuni(1);
      xWeakin2 = ranuni(1);

      xBeta  = xBeta + 0.1*xWeakIn1+ 0.1*xWeakIn2;

      do j=1 to dim(cIn);
         nLevs  = 2 + mod(j,&maxlevs-1);
         cIn{j} = 1+int(ranuni(1)*nLevs);
         xBeta  = xBeta + j*sign*(cIn{j}-nLevs/2);
         sign   = -sign;
      end;

      do j=1 to dim(cOut);
         nLevs  = 2 + mod(j,&maxlevs-1);
         cOut{j} = 1+int(ranuni(1)*nLevs);
      end;

      y = xBeta + &noiseScale*rannor(1);

      output;
   end;
run;
```

When you have insufficient prior knowledge of what effects need to be included in a parsimonious predictive model, a reasonable starting point is to use model selection to build a such a model. In such cases, you might

want to consider a large number of possible model effects, even though you know that a successful model that generalizes well for predicting unseen data depends on a relatively small number of effects. In such cases, you can dramatically reduce the computational task by including screening in the model selection process. The following statements show how you do this:

```
proc hpreg data=ex4Data;
   class c: ;
   model y = x: c: ;
   selection method=forward screen(details=all)=100 20;
   performance details;
run;
```

The ordered pair of integers that is specified in the SCREEN option in the SELECTION statement requests that screening be used to reduce the set of regressors to 100 regressors at the first screening stage and to 20 regressors at the second screening stage. This information is reflected in the "Screening Information" table shown in Output 67.3.1.

Output 67.3.1 Screening Information

The HPREG Procedure

Screening Information	
Screening Stages	Multiple
Screening Criterion	Maximum Absolute Correlation
Stage 1 Number of Screened Effects	100
Stage 2 Number of Screened Effects	20

The "Number Of Observations" table in Output 67.3.2 confirms that the data contain 50,000 observations and the "Dimensions" table shows that the selection is from 1,033 effects that have a total of 2,295 parameters.

Output 67.3.2 Number of Observations and Dimensions

Number of Observations Read	50000
Number of Observations Used	50000

Dimensions	
Number of Effects	1033
Number of Parameters	2295

Because you specified the DETAILS=ALL suboption of the SCREEN option, you obtain the "Screening" table in Output 67.3.3, which shows how the screened subset of 100 effects is obtained at the first screening stage. For display purposes, some ranks in this table have been suppressed.

Output 67.3.3 First Stage Screening Details

Effect Screening for Response		
Rank	Effect	Maximum Absolute Correlation
1	xIn25	0.31785
2	xIn24	0.30697
3	xIn23	0.29734
.	.	.
.	.	.
98	xOut338	0.00932
99	cOut363	0.00922
100	cOut194	0.00920
101	xOut125	0.00919*
102	xOut220	0.00916*
103	cOut310	0.00916*
104	cOut49	0.00915*
105	cOut11	0.00915*

The "Screened Effects" table shown in Output 67.3.4 lists the effects from which a model is selected at the first screening stage.

Output 67.3.4 First Stage Screened Effects

Screened Effects:	xIn25 xIn24 xIn23 xIn22 xIn21 xIn20 xIn19 xIn18 xIn17 xIn16 xIn15 xIn14 xIn13 cIn5 xIn12 cIn3 xIn11 xIn10 xIn8 xIn9 xIn7 cIn4 cIn2 xIn6 xIn5 xIn4 xIn3 cIn1 xIn2 cOut498 cOut110 cOut450 cOut441 cOut272 xOut82 cOut45 cOut6 cOut281 cOut134 cOut15 xOut310 xOut252 xOut485 xOut365 cOut138 cOut123 cOut337 cOut195 cOut423 cOut283 cOut62 cOut114 xOut489 cOut14 cOut158 cOut437 xOut64 cOut301 cOut311 cOut187 cOut431 cOut464 cOut388 cOut213 cOut46 xOut329 cOut403 cOut305 cOut171 cOut85 cOut99 cOut249 xOut267 cOut455 cOut457 cOut271 cOut78 xOut93 cOut259 cOut417 cOut258 cOut326 cOut291 cOut263 cOut107 cOut402 cOut17 cOut237 cOut129 cOut198 cOut58 cOut428 cOut135 cOut206 cOut139 cOut113 cOut486 xOut338 cOut363 cOut194

You see that the magnitude of the pairwise correlations of effects xIn1, xWeakIn1, and xWeakIn2 with response are too small for those effects to be included as candidates for selection at the first screening stage.

The first stage continues with forward selection from the screened effects that are shown in Output 67.3.4. The effects in the selected model at this stage are shown in Output 67.3.5.

Output 67.3.5 First Stage Selected Effects

Selected Effects:	Intercept xIn2 xIn3 xIn4 xIn5 xIn6 xIn7 xIn8 xIn9 xIn10 xIn11 xIn12 xIn13 xIn14 xIn15 xIn16 xIn17 xIn18 xIn19 xIn20 xIn21 xIn22 xIn23 xIn24 xIn25 cIn1 cIn2 cIn3 cIn4 cIn5

You see that the selected model at this stage includes only effects that are systematically related to the response. If you had requested that only a single-stage screening method be used by specifying the SINGLESTAGE suboption of the SCREEN option, then the selected model at this stage would have been the final selected model. However, multistage screening is used in this example. The second stage repeats the steps of the first stage except that the modeled response is the residuals from the selected model at the first stage.

Output 67.3.6 shows the screening details at the second stage. You see that 20 effects are chosen by screening at this stage as specified. Because the selected effects from the first stage are orthogonal to the residuals at

the first stage, none of these effects are in the screened subset. Furthermore, you see that although the effects xIn1, xWeakIn1, and xWeakIn2 are weakly correlated with y, they are the most strongly correlated effects with the residuals from the first stage.

Output 67.3.6 Second Stage Screening Details

Screening Stage 2: Residual Fit

Effect Screening for Stage 1 Residuals		
Rank	**Effect**	**Maximum Absolute Correlation**
1	**xIn1**	0.27373
2	**xWeakIn1**	0.02352
3	**xWeakin2**	0.02132
4	**cOut295**	0.01524
5	**cOut35**	0.01443
6	**cOut323**	0.01417
7	**cOut202**	0.01406
8	**xOut6**	0.01401
9	**cOut154**	0.01263
10	**cOut54**	0.01160
11	**cOut181**	0.01159
12	**cOut115**	0.01150
13	**cOut403**	0.01144
14	**xOut332**	0.01142
15	**xOut409**	0.01141
16	**cOut267**	0.01137
17	**cOut374**	0.01132
18	**cOut254**	0.01128
19	**xOut204**	0.01121
20	**cOut147**	0.01120
21	**xOut113**	0.01116*
22	**xOut427**	0.01115*
23	**cOut259**	0.01111*
24	**cOut170**	0.01106*
25	**cOut107**	0.01102*

*** Screened Out**

Screened Effects:	xIn1 xWeakIn1 xWeakin2 cOut295 cOut35 cOut323 cOut202 xOut6 cOut154 cOut54 cOut181 cOut115 cOut403 xOut332 xOut409 cOut267 cOut374 cOut254 xOut204 cOut147

Output 67.3.7 shows the selected effects at the second screening stage. You see that the selected effects are precisely the remaining effects that are systematically predictive of y but that were not in the screened subset at the first screening stage.

Output 67.3.7 Second Stage Selected Effects

Selected Effects:	Intercept xIn1 xOut6 xWeakIn1 xWeakin2

In the third and final screening stage, model selection is performed from the union of the screened effects from the first stage (which are shown in Output 67.3.4) and the selected effects from the second stage (which are shown in Output 67.3.7). The selected effects from this final stage are shown in Output 67.3.8.

Output 67.3.8 Final Stage Selected Effects

Selected Effects:	Intercept xIn1 xIn2 xIn3 xIn4 xIn5 xIn6 xIn7 xIn8 xIn9 xIn10 xIn11 xIn12 xIn13 xIn14 xIn15 xIn16 xIn17 xIn18 xIn19 xIn20 xIn21 xIn22 xIn23 xIn24 xIn25 xOut6 xWeakIn1 xWeakin2 cIn1 cIn2 cIn3 cIn4 cIn5

You see that the final selected model contains all the true underlying model effects and just one noise effect (xOut6). Because you specified the DETAILS option in the PERFORMANCE statement, the "Timing" table shown in Output 67.3.9 is displayed.

Output 67.3.9 Timing for Model Selection with Screening

Procedure Task Timing		
Task	**Seconds**	**Percent**
Reading and Levelizing Data	14.39	79.43%
Loading Design Matrix	1.66	9.15%
Computing Moments	0.49	2.72%
Computing Cross Products Matrix	1.22	6.71%
Performing Model Selection	0.36	2.00%

You see that even though the selected model was obtained by selecting from thousands of effects, screening enabled the entire modeling task to be completed in about 10 seconds. You can perform the same model selection without screening as shown in the following statements:

```
proc hpreg data=ex4Data;
   class c: ;
   model y = x: c: ;
   selection method=forward;
   performance details;
run;
```

In this case, the model that is selected without screening is identical to model that is obtained with screening. However, there is no guarantee that you will get identical selected models. Output 67.3.10 shows the "Timing" table for the model selection without screening.

Output 67.3.10 Timing for Model Selection without Screening

Procedure Task Timing		
Task	**Seconds**	**Percent**
Reading and Levelizing Data	8.69	5.16%
Loading Design Matrix	0.88	0.52%
Computing Moments	0.19	0.11%
Computing Cross Products Matrix	20.14	11.96%
Performing Model Selection	138.50	82.25%

You see that the model selection without screening took about 83 seconds, which is substantially slower than the approximately 10 seconds it took when screening was included in the selection process.

References

Akaike, H. (1969). "Fitting Autoregressive Models for Prediction." *Annals of the Institute of Statistical Mathematics* 21:243–247.

Burnham, K. P., and Anderson, D. R. (2002). *Model Selection and Multimodel Inference*. 2nd ed. New York: Springer-Verlag.

Collier Books (1987). *The 1987 Baseball Encyclopedia Update*. New York: Macmillan.

Darlington, R. B. (1968). "Multiple Regression in Psychological Research and Practice." *Psychological Bulletin* 69:161–182.

Draper, N. R., Guttman, I., and Kanemasu, H. (1971). "The Distribution of Certain Regression Statistics." *Biometrika* 58:295–298.

El Ghaoui, L., Viallon, V., and Rabbani, T. (2012). "Safe Feature Elimination for the Lasso and Sparse Supervised Learning Problems." *Pacific Journal of Optimization* 8:667–698. Special issue on conic optimization.

Fan, J., and Lv, J. (2008). "Sure Independence Screening for Ultrahigh Dimensional Feature Space." *Journal of the Royal Statistical Society, Series B* 70:849–911.

Foster, D. P., and Stine, R. A. (2004). "Variable Selection in Data Mining: Building a Predictive Model for Bankruptcy." *Journal of the American Statistical Association* 99:303–313.

Harrell, F. E., Jr. (2001). *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. New York: Springer.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Hocking, R. R. (1976). "The Analysis and Selection of Variables in a Linear Regression." *Biometrics* 32:1–50.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Judge, G. G., Griffiths, W. E., Hill, R. C., Lütkepohl, H., and Lee, T.-C. (1985). *The Theory and Practice of Econometrics*. 2nd ed. New York: John Wiley & Sons.

Mallows, C. L. (1973). "Some Comments on C_P." *Technometrics* 15:661–675.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J., and Tibshirani, R. (2012). "Strong Rules for Discarding Predictors in Lasso-Type Problems." *Journal of the Royal Statistical Society, Series B* 74:245–266.

Time Inc. (1987). "What They Make." *Sports Illustrated* (April 20): 54–81.

Chapter 68
The HPSPLIT Procedure

Contents

Overview: HPSPLIT Procedure **5180**
PROC HPSPLIT Features 5181
Getting Started: HPSPLIT Procedure **5182**
Syntax: HPSPLIT Procedure **5189**
PROC HPSPLIT Statement 5189
CLASS Statement 5198
CODE Statement 5200
GROW Statement 5200
ID Statement 5202
MODEL Statement 5202
OUTPUT Statement 5203
PARTITION Statement 5203
PERFORMANCE Statement 5205
PRUNE Statement 5205
RULES Statement 5206
WEIGHT Statement 5207
Details: HPSPLIT Procedure **5207**
Building a Decision Tree 5207
Splitting Criteria 5210
Splitting Strategy 5213
Pruning 5213
Memory Considerations 5218
Primary and Surrogate Splitting Rules 5218
Handling Missing Values 5219
Unknown Values of Categorical Predictors 5220
Scoring 5220
Measures of Model Fit 5221
Variable Importance 5225
ODS Table Names 5226
ODS Graphics 5227
SAS Enterprise Miner Syntax and Notes 5227
Examples: HPSPLIT Procedure **5229**
Example 68.1: Building a Classification Tree for a Binary Outcome 5230
Example 68.2: Cost-Complexity Pruning with Cross Validation 5240
Example 68.3: Creating a Regression Tree 5245
Example 68.4: Creating a Binary Classification Tree with Validation Data 5248

Example 68.5: Assessing Variable Importance 5257
References **5258**

Overview: HPSPLIT Procedure

The HPSPLIT procedure is a high-performance procedure that builds tree-based statistical models for classification and regression. The procedure produces classification trees, which model a categorical response, and regression trees, which model a continuous response. Both types of trees are referred to as decision trees because the model is expressed as a series of if-then statements.

The predictor variables for tree models can be categorical or continuous. The model is based on a partition of the predictor space into nonoverlapping segments, which correspond to the terminal nodes or leaves of the tree. The partitioning is done recursively, starting with the root node, which contains all the data, and ending with the terminal nodes. At each step of the recursion, the parent node is split into child nodes through selection of a predictor variable and a split value that minimize the variability in the response across the child nodes.

Tree models are built from training data for which the response values are known, and these models are subsequently used to score (classify or predict) response values for new data. For classification trees, the most frequent response level of the training observations in a leaf is used to classify observations in that leaf. For regression trees, the average response of the training observations in a leaf is used to predict the response for observations in that leaf. The splitting rules that define the leaves provide the information that is needed to score new data.

The process of building a decision tree begins with growing a large, full tree. Various measures, such as the Gini index, entropy, and residual sum of squares, are used to assess candidate splits for each node. The full tree can overfit the training data, resulting in a model that does not adequately generalize to new data. To prevent overfitting, the full tree is pruned back to a smaller subtree that balances the goals of fitting training data and predicting new data. Two commonly applied approaches for finding the best subtree are cost-complexity pruning (Breiman et al. 1984) and C4.5 pruning (Quinlan 1993). For more information, see the section "Building a Decision Tree" on page 5207.

SAS/STAT software provides many different methods of regression and classification. Compared with other methods, an advantage of tree models is that they are easy to interpret and visualize, especially when the tree is small. Tree-based methods scale well to large data, and they offer various methods of handling missing values, including surrogate splits.

However, tree models have limitations. Regression tree models fit response surfaces that are constant over rectangular regions of the predictor space, and so they often lack the flexibility needed to capture smooth relationships between the predictor variables and the response. Another limitation of tree models is that small changes in the data can lead to very different splits, and this undermines the interpretability of the model (Hastie, Tibshirani, and Friedman 2009; Kuhn and Johnson 2013).

PROC HPSPLIT Features

The main features of the HPSPLIT procedure are as follows:

- provides a variety of methods of splitting nodes, including criteria based on impurity (entropy, Gini index, residual sum of squares) and criteria based on statistical tests (chi-square, F test, CHAID, FastCHAID)
- provides a computationally efficient strategy for generating candidate splits
- provides the cost-complexity, C4.5, and reduced-error methods of pruning trees
- supports the use of cross validation and validation data for selecting the best subtree
- provides various methods of handling missing values, including surrogate rules
- creates tree diagrams, plots for cost-complexity analysis, and plots of ROC curves
- computes statistics for assessing model fit, including model-based (resubstitution) statistics and cross validation statistics
- computes measures of variable importance
- produces a file that contains SAS DATA step code for scoring new data
- produces a file that contains node rules
- provides an output data set with leaf assignments and predicted values for observations

The HPSPLIT procedure uses ODS Graphics to create plots as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the HPSPLIT procedure, see the PLOTS options in the PROC HPSPLIT statement and the section "ODS Graphics" on page 5227.

Because the HPSPLIT procedure is a high-performance analytical procedure, it also exploits all available cores and concurrent threads.

Getting Started: HPSPLIT Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/hpsplgs1.sas.)

This example explains basic features of the HPSPLIT procedure for building a classification tree. The data are measurements of 13 chemical attributes for 178 samples of wine. Each wine is derived from one of three cultivars that are grown in the same area of Italy, and the goal of the analysis is a model that classifies samples into cultivar groups. The data are available from the UCI Irvine Machine Learning Repository; see Lichman (2013).[1]

The following statements create a data set named Wine that contains the measurements:

```
%let url = http://archive.ics.uci.edu/ml/machine-learning-databases;
data Wine;
   infile "&url/wine/wine.data" url delimiter=',';
   input Cultivar Alcohol Malic Ash Alkan Mg TotPhen
         Flav NFPhen Cyanins Color Hue ODRatio Proline;
   label Cultivar = "Cultivar"
         Alcohol  = "Alcohol"
         Malic    = "Malic Acid"
         Ash      = "Ash"
         Alkan    = "Alkalinity of Ash"
         Mg       = "Magnesium"
         TotPhen  = "Total Phenols"
         Flav     = "Flavonoids"
         NFPhen   = "Nonflavonoid Phenols"
         Cyanins  = "Proanthocyanins"
         Color    = "Color Intensity"
         Hue      = "Hue"
         ODRatio  = "OD280/OD315 of Diluted Wines"
         Proline  = "Proline";
run;

proc print data=Wine(obs=10); run;
```

Figure 68.1 lists the first 10 observations of Wine.

[1]Disclaimer: SAS may reference other websites or content or resources for use at Customer's sole discretion. SAS has no control over any websites or resources that are provided by companies or persons other than SAS. Customer acknowledges and agrees that SAS is not responsible for the availability or use of any such external sites or resources, and does not endorse any advertising, products, or other materials on or available from such websites or resources. Customer acknowledges and agrees that SAS is not liable for any loss or damage that may be incurred by Customer or its end users as a result of the availability or use of those external sites or resources, or as a result of any reliance placed by Customer or its end users on the completeness, accuracy, or existence of any advertising, products, or other materials on, or available from, such websites or resources.

Figure 68.1 Partial Listing of Wine

Obs	Cultivar	Alcohol	Malic	Ash	Alkan	Mg	TotPhen	Flav	NFPhen	Cyanins	Color	Hue	ODRatio	Proline
1	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
2	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
3	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
4	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
5	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735
6	1	14.20	1.76	2.45	15.2	112	3.27	3.39	0.34	1.97	6.75	1.05	2.85	1450
7	1	14.39	1.87	2.45	14.6	96	2.50	2.52	0.30	1.98	5.25	1.02	3.58	1290
8	1	14.06	2.15	2.61	17.6	121	2.60	2.51	0.31	1.25	5.05	1.06	3.58	1295
9	1	14.83	1.64	2.17	14.0	97	2.80	2.98	0.29	1.98	5.20	1.08	2.85	1045
10	1	13.86	1.35	2.27	16.0	98	2.98	3.15	0.22	1.85	7.22	1.01	3.55	1045

The variable Cultivar is a nominal categorical variable with levels 1, 2, and 3, and the 13 attribute variables are continuous.

The following statements use the HPSPLIT procedure to create a classification tree:

```
ods graphics on;

proc hpsplit data=Wine seed=15533;
   class Cultivar;
   model Cultivar = Alcohol Malic Ash Alkan Mg TotPhen Flav
                    NFPhen Cyanins Color Hue ODRatio Proline;
   grow entropy;
   prune costcomplexity;
run;
```

The MODEL statement specifies Cultivar as the response variable and the variables to the right of the equal sign as the predictor variables. The inclusion of Cultivar in the CLASS statement designates it as a categorical response variable and requests a classification tree. All the predictor variables are treated as continuous variables because none are included in the CLASS statement.

The GROW and PRUNE statements control two fundamental aspects of building classification and regression trees: growing and pruning. You use the GROW statement to specify the criterion for recursively splitting parent nodes into child nodes as the tree is grown. For classification trees, the default criterion is entropy; see the section "Splitting Criteria" on page 5210.

By default, the growth process continues until the tree reaches a maximum depth of 10 (you can specify a different limit by using the MAXDEPTH= option). The result is often a large tree that overfits the data and is likely to perform poorly in predicting future data. A recommended strategy for avoiding this problem is to prune the tree to a smaller subtree that minimizes prediction error. You use the PRUNE statement to specify the method of pruning. The default method is cost complexity; see the section "Pruning" on page 5213.

The default output includes the four informational tables that are shown in Figures 68.2 through 68.5.

The "Performance Information" table in Figure 68.2 shows that the procedure ran locally and that four threads were used. The default number of threads is the number of processors in the computer.

Figure 68.2 Performance Information

The HPSPLIT Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	16

The "Data Access Information" table in Figure 68.3 shows that the input data set is accessed using the V9 (base) engine on the client machine where the MVA SAS session executes. This table includes similar information about output data sets that you can request by using the OUTPUT statement.

Figure 68.3 Data Access Information

Data Access Information			
Data	**Engine**	**Role**	**Path**
WORK.WINE	V9	Input	On Client

The "Model Information" table in Figure 68.4 provides information about the model and the methods that are used to grow and prune the tree.

Figure 68.4 Model Information

Model Information	
Split Criterion Used	Entropy
Pruning Method	Cost-Complexity
Subtree Evaluation Criterion	Cost-Complexity
Number of Branches	2
Maximum Tree Depth Requested	10
Maximum Tree Depth Achieved	4
Tree Depth	4
Number of Leaves Before Pruning	8
Number of Leaves After Pruning	8

The "Observation Information" table in Figure 68.5 provides the numbers of observations that are read and used.

Figure 68.5 Observation Information

Number of Observations Read	178
Number of Observations Used	178

These numbers are the same in this example because there are no missing values in the predictor variables. By default, observations that have missing values are not used. However, the HPSPLIT procedure provides methods for incorporating missing values in the analysis, as explained in the sections "Handling Missing Values" on page 5219 and "Primary and Surrogate Splitting Rules" on page 5218.

The plot in Figure 68.6 is a tool for selecting the tuning parameter for cost-complexity pruning. The parameter, indicated on the lower horizontal axis, indexes a sequence of progressively smaller subtrees that are nested within the large tree. The parameter value 0 corresponds to the large tree, and positive values control

the trade-off between complexity (number of leaves) and fit to the training data, as measured by average misclassification rate.

Figure 68.6 Misclassification Rate as a Function of Cost-Complexity Parameter

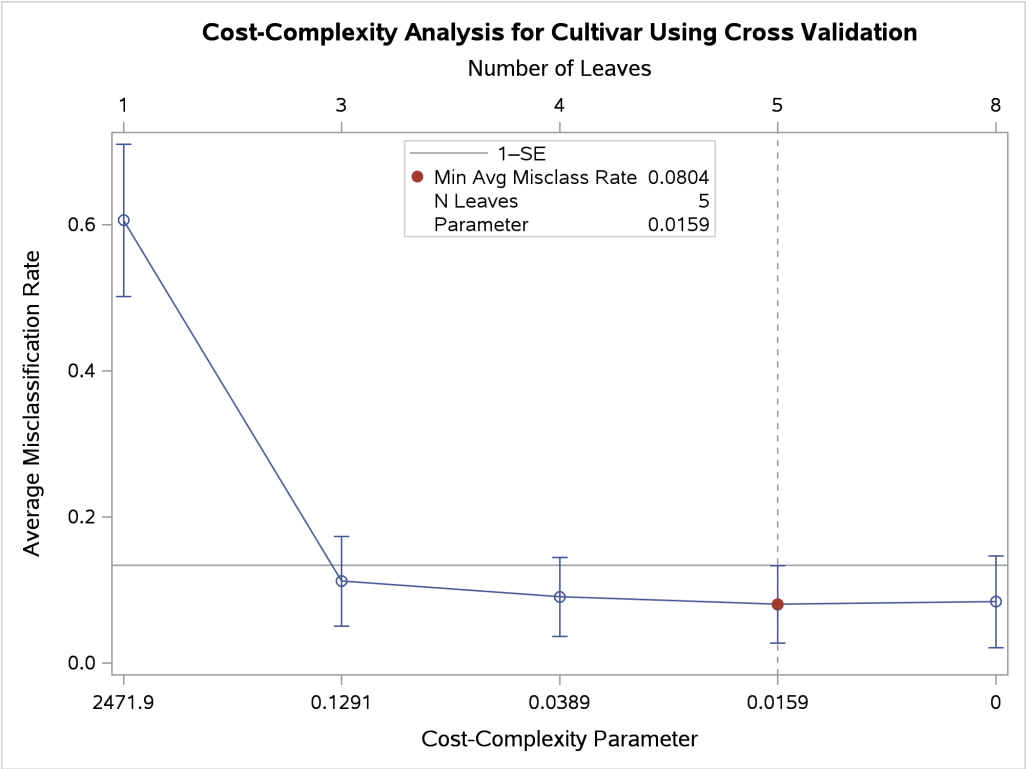

Figure 68.6 shows that PROC HPSPLIT selects the parameter value 0.0159 because it minimizes the average misclassification rate, which is obtained by 10-fold cross validation. However, the average misclassification rates for several other parameter choices are nearly the same.

Breiman's 1-SE rule chooses the parameter that corresponds to the smallest subtree for which the misclassification rate is less than one standard error above the minimum misclassification rate (Breiman et al. 1984). This parameter choice (0.1291) corresponds to a tree that has only three leaves.

NOTE: The estimates of misclassification rate and their standard errors depend on the random assignment of observations to the folds in the cross validation; the random assignment is determined by the SEED= option.

The following statements build a classification tree by growing a large tree and applying cost-complexity pruning (also known as weakest-link pruning) to obtain a tree that has three leaves:

```
proc hpsplit data=Wine seed=15533;
   class Cultivar;
   model Cultivar = Alcohol Malic Ash Alkan Mg TotPhen Flav
                    NFPhen Cyanins Color Hue ODRatio Proline;
   prune costcomplexity(leaves=3);
run;
```

The tree diagram in Figure 68.7, which is produced by default when ODS Graphics is enabled, provides an overview of the tree as a classifier.

Figure 68.7 Overview Diagram of Final Tree

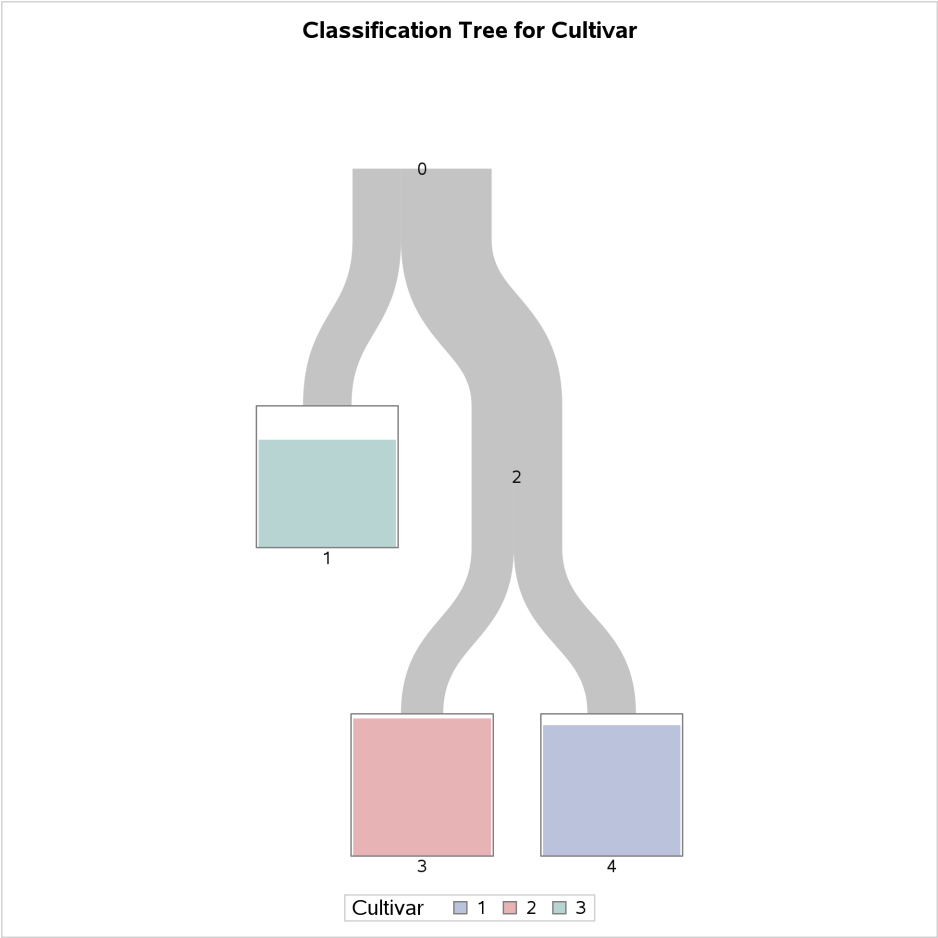

The tree is constructed starting with all the observations in the root node (labeled 0). This node is split into a leaf node (1) and an internal node (2), which is further split into two leaf nodes (3 and 4).

The color of the bar in each leaf node indicates the most frequent level of Cultivar among the observations in that node; this is also the classification level assigned to all observations in that node. The height of the bar indicates the proportion of observations in the node that have the most frequent level. The width of the link between parent and child nodes is proportional to the number of observations in the child node.

The diagram reveals that splitting on just two of the attributes is sufficient to differentiate the three cultivars, and a tree model that has only three leaves provides a high degree of accuracy for classification.

The diagram in Figure 68.8 provides more detail about the nodes and the splits.

Figure 68.8 Detailed Tree Diagram

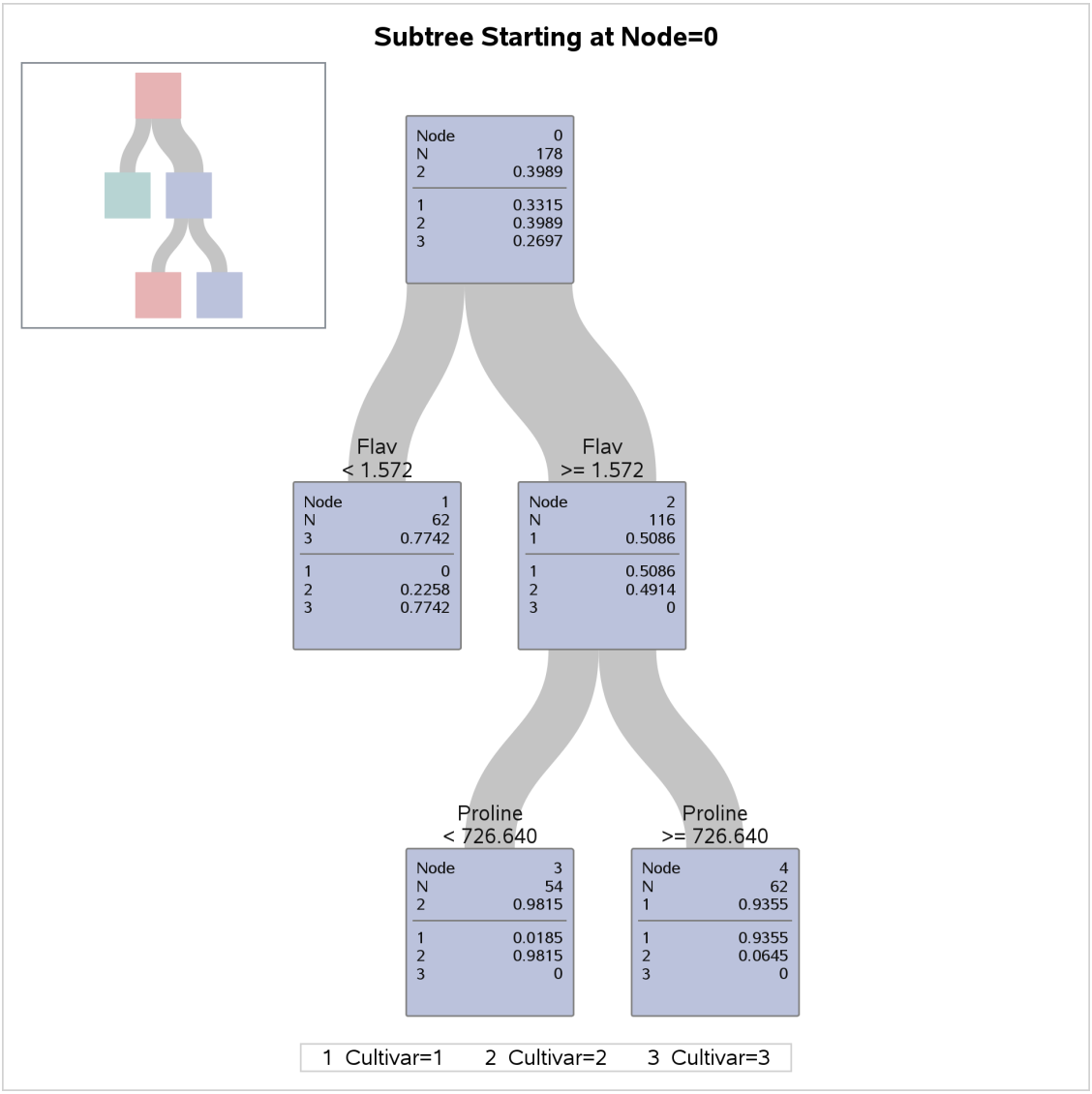

There are 178 samples in the root node (node 0). The table below the line in the box for node 0 provides the proportion of samples for each level of Cultivar, and the level that has the highest proportion is also given above the line. These samples are divided into 62 samples for which Flav < 1.572 (node 1) and 116 samples for which Flav ≥ 1.572 (node 2).

The variable Flav and the split point 1.572 are chosen to maximally decrease the impurity of the root node as measured by the entropy criterion. There are no samples for which Cultivar=1 in node 1, and there are no samples for which Cultivar=3 in node 2.

The samples in node 2 were further divided into 54 samples for which Proline < 726.640 (node 3) and 62 samples for which Proline ≥ 726.640 (node 4).

The classification tree yields simple rules for predicting the wine cultivar. For example, a sample for which Flav ≥ 1.572 and Proline < 726.640 is predicted to be from the third cultivar (Cultivar=3).

The diagram in Figure 68.8 happens to show the entire tree that was created by the preceding statements, but in general this diagram shows a subtree of the entire tree that begins with the root node and has a depth of four levels. You can use the PLOTS=ZOOMEDTREE option in the PROC HPSPLIT statement to request diagrams that begin with other nodes and have specified depths.

The confusion matrix in Figure 68.9 evaluates the accuracy of the fitted tree for classifying the training data that were used to build the tree model.

Figure 68.9 Confusion Matrix

The HPSPLIT Procedure

Model-Based Confusion Matrix

	Predicted			
Actual	**1**	**2**	**3**	**Error Rate**
1	58	1	0	0.0169
2	4	53	14	0.2535
3	0	0	48	0.0000

The values in the off-diagonal entries of the matrices show how many times the fitted model misclassified a sample. For example, among the 59 samples for which Cultivar=1, only one sample was misclassified, and it was incorrectly classified as Cultivar=2.

The "Tree Performance" table in Figure 68.10 displays various fit statistics for the tree model.

Figure 68.10 Fit Statistics

Model-Based Fit Statistics for Selected Tree

N Leaves	ASE	Mis-class	Entropy	Gini	RSS
3	0.0583	0.1067	0.4290	0.1749	31.1243

For instance, the misclassification rate is the proportion of the 178 wine samples that were misclassified: (1 + 4 + 14)/178 = 0.1067.

You can use the CVMODELFIT option in the PROC HPSPLIT statement to request a model assessment that is based on cross validation; this is done independently of the cross validation used to assess pruning parameters. For more information about fit statistics, see the section "Measures of Model Fit" on page 5221.

NOTE: Additional fit statistics are available in the case of classification trees for binary responses. See "Example 68.1: Building a Classification Tree for a Binary Outcome" on page 5230.

Syntax: HPSPLIT Procedure

The following statements and options are available in the HPSPLIT procedure:

PROC HPSPLIT < *options* > ;
  CLASS *variable*. . . < /*options* > ;
  CODE FILE=*filename* ;
  GROW *criterion* < / *options* > ;
  ID *variables* ;
  MODEL *response* < (*response-options*) > = *variable* < *variable*. . . > ;
  OUTPUT *output-options* ;
  PARTITION < *partition-options* > ;
  PERFORMANCE *performance-options* ;
  PRUNE *prune-method* < (*prune-options*) > ;
  RULES FILE=*filename* ;
  WEIGHT *variable* < / **UNWEIGHTDF** > ;

The PROC HPSPLIT statement and the MODEL statement are required. If any variables are character or to be treated as categorical, at least one CLASS statement is required. Variables that appear after the equal sign (=) in the MODEL statement are explanatory variables that model the *response* variable. By default, all *variables* that appear in the MODEL statement are treated as continuous variables. CLASS statements cause a *variable* to be treated as categorical. Specifying a *variable* in a CLASS statement but not in a MODEL statement causes the variable to be ignored and a warning to be issued.

The following sections describe the PROC HPSPLIT statement and then describe the other statements in alphabetical order.

PROC HPSPLIT Statement

PROC HPSPLIT < *options* > ;

The PROC HPSPLIT statement invokes the procedure. Table 68.1 summarizes the options in the PROC HPSPLIT statement.

Table 68.1 PROC HPSPLIT Statement Options

Option	Description
Basic Options	
CVCC	Requests a table of the results of cost-complexity pruning based on cross validation
CVMETHOD=	Specifies the cross validation method to use
CVMODELFIT	Requests model assessment and a confusion matrix with cross validation
DATA=	Specifies the predictor data set
INTERVALBINS=	Specifies the number of bins for continuous variables
MINVARIANCE=	Sets the minimum variance for a regression tree leaf to be split
NODES	Requests a table that describes the nodes of the final tree

Table 68.1 *continued*

Option	Description
NOPRINT	Suppresses ODS output
NSURROGATES=	Specifies the number of surrogate rules to create
PLOTS=	Specifies options for plots
SEED=	Specifies the random number seed to use for cross validation
SPLITONCE	Specifies that variables should be split only once per branch
Splitting Options	
ASSIGNMISSING=	Specifies how to handle missing values in a predictor variable
LEVTHRESH1=	Specifies the maximum number of computations to perform in an exhaustive search for a categorical predictor
LEVTHRESH2=	Specifies the number of computations to perform before the splitter uses the fastest greedy search
MAXBRANCH=	Specifies the maximum number of leaves per node
MAXDEPTH=	Specifies the maximum tree depth
MINCATSIZE=	Specifies the number of observations per level in order for the level to be considered for splitting
MINLEAFSIZE=	Specifies the minimum number of observations per leaf

You can specify the following *options*.

ASSIGNMISSING=BRANCH | NONE | POPULAR | SIMILAR

specifies how PROC HPSPLIT creates a default splitting rule to handle missing values, unknown levels, and levels that have fewer observations than you specify in the MINCATSIZE= option. An unknown level is a level of a categorical predictor that does not exist in the training data but is encountered during scoring.

Both the ASSIGNMISSING= and NSURROGATES= options affect training and scoring. See the NSURROGATES= option for the definition of surrogate rules.

During training, the primary splitting rule is created first, along with the default splitting rule (controlled by the ASSIGNMISSING= option). If you request surrogate rules (by using the NSURROGATES= option), they are created after the primary and default splits are made. When the splitting rules have been created, the rules are used as described in the following list for assigning the training data by using the new splitting rules, and splitting rule creation continues on the new children.

Observation assignment during the training phase and during scoring proceeds as follows:

1. The primary splitting rule is applied if the rule's variable is not missing. Otherwise:
2. The first surrogate rule (with the largest agreement, described in the section "Primary and Surrogate Splitting Rules" on page 5218) is applied if the first surrogate rule's variable is not missing. Otherwise:
3. Each subsequent (ordered by agreement) splitting rule is applied as described in the first surrogate item in this list. If all of the surrogate rules' variables are missing, then:
4. The default splitting rule is used.

Because there is always a default splitting rule, all data can be scored, even if the primary rule and all surrogate rules cannot be used on a particular observation.

You can specify one of the following:

BRANCH
specifies that PROC HPSPLIT create a special child (branch) for the default rule and assign to that child missing values, unknown levels, and levels that have fewer observations than you specify in the MINCATSIZE= option. You need to have missing values in the training data set for this to work. If no missing values or levels that have fewer observations than you specify in the MINCATSIZE= option are available at the split in the training data set, then missing values are assigned using ASSIGNMISSING=POPULAR.

NONE
specifies that observations that have any missing variables be excluded from the analysis. In addition, the default rule (to handle unknown levels, which include missing values, because they are excluded from the analysis, and levels that have fewer observations than you specify in the MINCATSIZE= option) is created using ASSIGNMISSING=POPULAR.

POPULAR
specifies that missing values be assigned to the most popular (largest) child.

SIMILAR
specifies that missing values be assigned to the child that they are most similar to (using the chi-square for categorical responses). For more information about this option and the similarity measurement, see the section "Handling Missing Values" on page 5219. If no missing values or levels that have fewer observations than you specify in the MINCATSIZE= option are available at the split in the training data set, then missing values are assigned using ASSIGNMISSING=POPULAR. If you specify a continuous response, then missing values are assigned using ASSIGNMISSING=POPULAR.

By default, ASSIGNMISSING=NONE.

CVCC
CVCOSTCOMPLEXITY
requests a table of the results of cost-complexity pruning based on cross validation. For each fold in the cross validation, the table provides the penalty parameter, the number of leaves, and the average ASE. In addition, the table provides the minimum and maximum ASE, and the minimum, median, and maximum number of leaves. The number of leaves is the floor of the median. You can use the PLOTS=CVCC option to request a plot of the information in this table.

CVMETHOD=NONE | RANDOM < (*k*) >
requests the cross validation method to be performed.

You can specify one of the following:

NONE
suppresses cross validation.

RANDOM < (*k*) >
assigns each training observation randomly to one of the *k* folds (with a probability of 1/*k* for any given fold) for cross validation. The default value of *k* is 10.

Cross validation applies to the CVMODELFIT option and cost-complexity pruning. In *k*-fold cross validation, the training set is divided into *k* folds; *k* trees are then built using all but the one fold. For example, the first tree uses all of the training set except for the observations in the first fold. The second uses all of the training set except for the observations in the second fold, and so on.

The holdout fold for each tree is used to calculate the ASE of that tree. The average ASE across the *k* trees is the cross validation error for that set of trees. This process is repeated for each value of the parameter that is cross validated. The parameter that has the minimum cross validated error is used as the best parameter value. A final tree is then grown using this final parameter value.

NOTE: Attempting to use a PARTITION statement along with cross validation results in an error.

By default, CVMETHOD=RANDOM(10) when you perform cost-complexity pruning with no PARTITION statement.

CVMODELFIT
requests model assessment with cross validation. When you specify this option, the procedure does a cross validation of the final model parameters and produces a table that describes the cross validation error measures of the parameters. The table contains various summary statistics of the ASE, the number of leaves, and, for a categorical response, the misclassification rate across the *k* trees grown. This option also requests a table that contains the cross validation confusion matrix.

If cost complexity is cross validated (the default if you use cost-complexity pruning without a validation set), the assessment is a completely separate cross validation of the final tree penalty parameter, using a different seed for fold assignment.

NOTE: The assessment is not on a holdout sample but instead on *k*-fold cross validation. This is not a second run of the procedure but instead is done automatically and internally.

DATA=*SAS-data-set*
names the predictor SAS data set to be used by PROC HPSPLIT. The default is the most recently created data set.

INTERVALBINS=*number*
specifies the *number* of bins for continuous variables. PROC HPSPLIT bins continuous predictors to a fixed bin size. This option controls the number of bins and thereby also the size of the bins. For more information about interval variable binning, see the section "Details: HPSPLIT Procedure" on page 5207.

By default, INTERVALBINS=100.

LEVTHRESH1=*number*

applies only to categorical predictor variables and specifies the limit for the number of computations in an exhaustive search for the optimal partition of the levels of a particular variable. The splitter first evaluates the number of computations that are needed for an exhaustive search. If this number exceeds the limit, then the splitter falls back to a faster heuristic algorithm. You can use the LEVTHRESH2= option to specify the limit for the number of computations in this faster algorithm.

By default, LEVTHRESH1=500000.

LEVTHRESH2=*number*

applies to categorical predictor variables and continuous predictor variables with multiway splits. This option does not apply to continuous predictor variables with binary splits.

For a categorical predictor variable, the splitter first evaluates the number of computations that are needed for an exhaustive search. If this number exceeds the limit that you specify in the LEVTHRESH1= option, then the splitter falls back to a faster heuristic algorithm. You can use the LEVTHRESH2= option to specify the limit for the number of computations in this faster algorithm. If this number of computations exceeds the limit that you specify in the LEVTHRESH2= option, then the splitter falls back to an even faster algorithm.

For a continuous predictor variable, the splitter first tries to perform an exhaustive search for the optimal split values. The splitter first evaluates the number of computations that are needed for an exhaustive search. If this number exceeds the limit that you specify in the LEVTHRESH2= option, the splitter falls back to a faster heuristic algorithm.

By default, LEVTHRESH2=1000000.

MAXBRANCH=*b*

specifies the maximum number of children per node in the tree. PROC HPSPLIT tries to create this number of children unless it is impossible (for example, if a split variable does not have enough levels). By default, MAXBRANCH=2.

MAXDEPTH=*number*

specifies the maximum depth of the tree to be grown. The default is set using the following equation, where b is the value for the MAXBRANCH= option:

$$\text{MaxDepth} = \left\lceil \frac{10}{\log_2(b)} \right\rceil$$

MINCATSIZE=*number*

specifies the *number* of observations that a categorical variable level must have in order to be considered in the split. Predictor variable levels that have fewer observations than *number* receive the nonsurrogate missing value assignment for that split. See the NSURROGATES= option for the definition of surrogate rules.

By default, MINCATSIZE=1.

MINLEAFSIZE=*number*

LEAFSIZE=

specifies the minimum *number* of observations that each child of a split must contain in the training data set in order for the split to be considered.

By default, MINLEAFSIZE=1.

MINVARIANCE=*value*

specifies the minimum variance for a regression tree leaf to be eligible for splitting. That is, leaves whose variance is less than *value* are not split any further.

By default, MINVARIANCE=1E–8.

The variance at some leaf λ with weight N_λ (number of observations) is calculated using

$$\text{Var} = \frac{\sum_i^\lambda y_i^2}{N_\lambda} - \bar{y}_\lambda^2$$

where y_i is the response value at observation i and $\bar{y}_\lambda$ is the average value of the response within leaf λ.

NODES< =DETAIL | SUMMARY >

requests a table that contains the descriptions of the paths from each leaf to the root.

You can specify the following values:

DETAIL

prints all nodes and leaves, including the path from each node or leaf to the root of the tree.

SUMMARY

prints all nodes and leaves, without including the path to the root of the tree.

If you do not specify any value, then the HPSPLIT procedure prints all leaf nodes, including the path from each leaf to the root of the tree.

NOPRINT

suppresses the generation of ODS output.

NSURROGATES=*number*

specifies the *number* of surrogate rules to create for each splitting rule. Surrogate rules are backup splitting rules that are used when the variable that corresponds to the primary splitting rule is missing. By default, NSURROGATES=0.

Both the ASSIGNMISSING= option and NSURROGATES= options affect training and scoring.

During training, the primary splitting rule is created first, along with the default splitting rule (controlled by the ASSIGNMISSING= option). If you request surrogate rules (by specifying the NSURROGATES= option), they are created after the primary and default splits are created. When the splitting rules have been created, the rules are used as described in the following list for assigning the training data by using the new splitting rules, and splitting rule creation continues on the new children.

Observation assignment during the training phase and during scoring proceeds as follows:

1. The primary splitting rule is applied if the primary rule's variable is not missing. Otherwise:

2. The first surrogate rule (with the largest agreement, described in the section "Primary and Surrogate Splitting Rules" on page 5218) is applied if the first surrogate rule's variable is not missing. Otherwise:
3. Each subsequent (ordered by agreement) splitting rule is applied as described in the first surrogate item in the list. If all of the surrogate rules' variables are missing, then:
4. The default splitting rule is used.

Because there is always a default splitting rule, all data can be scored, even if the primary rule and all surrogate rules cannot be used on a particular observation.

As auxiliary rules, surrogate rules are used in the following ways:

- Surrogate rules are used in generating the tree.
- Surrogate rules therefore also affect the final tree metrics.
- Surrogate rules are used in the SAS code output from the CODE statement.
- Surrogate rules are used in scoring the predictor data set by using the OUTPUT statement.
- Surrogate rules are *not* written out in the "Node Rules" file, output from the RULES statement.

PLOTS < (*global-plot-option*) > < = *plot-request* < (*options*) > >
PLOTS < (*global-plot-option*) > < = (*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >
controls the plots that are produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around it. Some examples follow.

You can specify the following *global-plot-option*:

ONLY
suppresses the default plots. Only plots that you specifically request are displayed.

You can specify the following *plot-requests*:

ALL
produces all appropriate plots.

CVCC< (ASE) >
produces a plot that is used to determine the tuning parameter for cost-complexity pruning when cross validation is used. This plot is disabled when you specify the LEAVES= option in the PRUNE statement or when you are not using cross validation of cost-complexity pruning. By default, this plot displays the average misclassification rate when the response variable is a classification variable and displays the average square error (ASE) when the response variable is a continuous variable. You can specify the following suboption:

ASE
displays the ASE when the response variable is a classification variable.

NONE
suppresses the default plots. Only plots that you specifically request are displayed.

PRUNEUNTIL
produces a plot of the metric that is used to select the final subtree. This option is enabled by default except in the following cases:

- You request cross validation of cost-complexity pruning.
- You specify the LEAVES= option in the PRUNE statement when you use the cost-complexity pruning method.
- You specify the OFF option in the PRUNE statement. Note that specifying the PRUNEUNTIL option has no effect in this case.

ROC
produces the receiver operating characteristic (ROC) curve. This option is enabled by default.

WHOLETREE < (*whole-tree-options*) >
produces a plot to visualize the entire finished (grown and pruned) tree. This option is enabled by default.

You can specify the following *whole-tree-options*:

LINKSTYLE=*link-style*
specifies the style of links between nodes and leaves in the tree. You can specify the following *link-styles*:

CURVED
requests curved links between the nodes and their children.

ORTHOGONAL
requests that links go straight down partway from a node to its children, create a horizontal line at the base of the vertical line, and then go straight down from that line to each child.

STRAIGHT
requests that links go straight from the nodes to their children.

By default, LINKSTYLE=CURVED.

LINKWIDTH=*link-width*
specifies the width of links between nodes and leaves in the tree. You can specify the following *link-widths*:

CONSTANT
requests that all links have the same thickness.

PROPORTIONAL
requests that links have a thickness proportional to the total number of observations that go between the node and each child.

By default, LINKWIDTH=PROPORTIONAL.

NOLEGEND
turns off the legend.

ZOOMEDTREE < (*zoomed-tree-options*) >
produces a plot to visualize a portion of the finished (grown and pruned) tree. This option is enabled by default.

You can specify the following *zoomed-tree-options*:

DEPTH=*depth*
requests that PROC HPSPLIT create plots that are specified by the ZOOMEDTREE option for each *node-id* down to *depth*.

FRACTIONPRECISION=*integer*
requests that the fraction of total observations that are displayed in the decision tree node have a number of digits after the decimal equal to *integer*.

By default, FRACTIONPRECISION=4.

LINKSTYLE=*link-style*
specifies the style of links between nodes and leaves in the tree. You can specify the following *link-styles*:

CURVED
requests curved links between the nodes and their children.

ORTHOGONAL
requests that links go straight down partway from a node to its children, create a horizontal line at the base of the vertical line, and then go straight down from that line to each child.

STRAIGHT
requests that links go straight from the nodes to their children.

By default, LINKSTYLE=CURVED.

LINKWIDTH=*link-width*
specifies the width of links between nodes and leaves in the tree. You can specify the following *link-widths*:

CONSTANT
requests that all links have the same thickness.

PROPORTIONAL
requests that links have a thickness proportional to the total number of observations that go between the node and each child.

By default, LINKWIDTH=PROPORTIONAL.

NODES=('*node-id*' < '*node-id*' < . . . > >**)**
requests plots for the subtree that is rooted at nodes at *node-id*. The default node ID is '0', the root of the entire tree. The values of *node-id* are alphanumeric strings that are displayed within the nodes in the plot that is created by the WHOLETREE option. PROC HPSPLIT creates one plot that is specified by the ZOOMEDTREE option for each *node-id* that you specify. The DEPTH=, LINKSTYLE=, and LINKWIDTH= option values apply to each plot that the ZOOMEDTREE option creates; if you specify any of these option values more than once, then only the last value specified is used.

The nodes in the decision tree are named using a base 62 convention, in order to improve the visual quality of the plots. That is, the first 10 nodes are named '0' through '9'. After '9', the next 26 nodes are named 'A' through 'Z'. The following 26 nodes are named 'a' through 'z'. After these first 62 nodes, a second digit is added, with the names starting at '10'. In a similar fashion, the name after '19' is '1A', the name after '1Z' is '1a', and the name after '1z' is '20'.

A quick conversion is the mapping 'A'=10 to 'Z'=35 and 'a'=36 to 'z'=61. Thus, for example, a node named '1F' would equal 1*62 + 15 = 77 (where 15 is the value of 'F').

NOLEGEND
turns off the legend.

PREDICTORPRECISION=*integer*
requests that the split value that is displayed above the decision tree node have a number of digits after the decimal equal to *integer*.

By default, PREDICTORPRECISION=3.

SEED=*number*
specifies the initial seed for random number generation for cross validation. The value of *number* must be an integer. The default seed is based on the date and time.

SPLITONCE
specifies that variables be split only once on a branch. However, a variable can be used more than once across branches. That is, a variable cannot be split more than once on the path from any leaf to the root node.

CLASS Statement

CLASS *variable* < **(***var-options***)** > < *variable* < **(***var-options***)** > < . . . > > < / *class-options* > **;**

The CLASS statement causes *variable* to be treated as a categorical variable in the analysis. These variables enter the analysis not through their values but through levels to which the unique values are mapped. For more information about these mappings, see the section "Levelization of Classification Variables" on page 409.

Multiple CLASS statements are supported.

If you use the CLASS statement, it must appear before the MODEL statement.

NOTE: All class levels are padded or truncated to 32 characters.

By default, PROC HPSPLIT treats *variable*s as categorical variables whose order is specified by the ORDER= option.

You can specify the following *var-options*:

DESC
DESCENDING
reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC HPSPLIT orders the categories according to the ORDER= option and then reverses that order.

ORDER=*ordering*
specifies the sort order for the levels of classification variables. By default, ORDER=FORMATTED except for numeric CLASS variables that have no specified format, for which ORDER=INTERNAL is the default. You can specify the following values:

FORMATTED orders values in ascending order of the formatted value.

INTERNAL orders values in ascending order of the unformatted value.

You can specify the following *class-options*:

DESC
DESCENDING
reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC HPSPLIT orders the categories according to the ORDER= option and then reverse that order.

ORDER=*ordering*
specifies the sort order for the levels of classification variables. By default, ORDER=FORMATTED except for numeric CLASS variables that have no specified format, for which ORDER=INTERNAL is the default. You can specify the following values:

FORMATTED orders values in ascending order of the formatted value.

INTERNAL orders values in ascending order of the unformatted value.

UPCASE
uppercases the values of character-valued CLASS variables before levelizing them. For example, if the UPCASE option is in effect and a CLASS variable can take the values ‘a’, ‘A’, and ‘b’, then ‘a’ and ‘A’ represent the same level and the CLASS variable is treated as having only two values: ‘A’ and ‘B’.

CODE Statement

CODE FILE=*filename* **;**

The CODE statement converts the final tree into SAS DATA step code that can be used for scoring. The code is written to the file that is specified by *filename*.

If you do not specify a CODE statement, no SAS DATA step code is output.

GROW Statement

GROW *criterion* < **(***options***)** > **;**

The GROW statement specifies the *criterion* by which to grow the tree. For more information, see the section "Splitting Criteria" on page 5210. For categorical responses, the available *criteria* are CHAID, CHISQUARE, ENTROPY, FASTCHAID, and GINI, and the default *criterion* is ENTROPY. For continuous responses, the available *criteria* are CHAID, FTEST, and RSS, and the default *criterion* is RSS.

For either categorical or continuous responses, you can specify the following *criterion*:

CHAID < **(***options***)** >

For categorical predictors, CHAID uses values of a chi-square statistic (in the case of a classification tree) or an F statistic (in the case of a regression tree) to merge similar levels until the number of children in the proposed split reaches the number that you specify in the MAXBRANCH= option. The p-values for the final split determine the variable on which to split.

For continuous predictors, CHAID chooses the best single split until the number of children in the proposed split reaches the value that you specify in the MAXBRANCH= option.

You can specify the following *options*:

ALPHA=*value*

specifies the maximum p-value for a split to be considered.

By default, ALPHA=0.3.

BONFERRONI

requests a Bonferroni adjustment to the p-value for a variable after the split has been determined.

By default, no adjustment is made.

For categorical responses only, you can specify the following *criteria*:

CHISQUARE < **(***options***)** >

uses a chi-square statistic to split each variable and then uses the p-values that correspond to the resulting splits to determine the splitting variable.

You can specify the following *options*:

ALPHA=*value*
specifies the maximum p-value for a split to be considered.

By default, ALPHA=0.3.

BONFERRONI
requests a Bonferroni adjustment to the p-value for a variable after the split has been determined.

By default, no adjustment is made.

ENTROPY
uses the gain in information (decrease in entropy) to split each variable and then to determine the split.

FASTCHAID < (*options***) >**
uses a Kolmogorov-Smirnov splitter to determine splits for each variable. The FastCHAID *criterion* follows a recursive method similar to that of Friedman (1977) after ordering the levels according to the response variable. The *criterion* then selects the split variable as the variable that has the smallest p-value.

You can specify the following *options*:

ALPHA=*value*
specifies the maximum p-value for a split to be considered.

By default, ALPHA=0.3.

BONFERRONI
requests a Bonferroni adjustment to the p-value for a variable after the split has been determined.

By default, no adjustment is made.

MINDIST=*number*
specifies the minimum Kolmogorov-Smirnov distance for a candidate split.

By default, MINDIST=0.01.

GINI
uses the decrease in the Gini index to split each variable and then to determine the split.

IGR
uses the entropy metric to split each variable and then uses the information gain ratio to determine the split.

The default *criterion* for categorical responses is ENTROPY.

For continuous responses only, you can specify the following *criteria*:

FTEST < (*options***) >**
uses an F statistic to split each variable and then uses the resulting p-value to determine the split variable.

You can specify the following *options*:

ALPHA=*value*
specifies the maximum *p*-value for a split to be considered.

By default, ALPHA=0.3.

BONFERRONI
requests a Bonferroni adjustment to the *p*-value for a variable after the split has been determined.

By default, no adjustment is made.

RSS
VARIANCE
uses the change in response variance to split each variable and then to determine the split.

The default *criterion* for continuous responses is RSS.

ID Statement

ID *variables* **;**

The ID statement lists one or more variables from the predictor data set that are to be transferred to the output data set that you specify in the OUTPUT statement.

For more information, see the section "ID Statement" on page 576.

MODEL Statement

MODEL *response* < **(***response-option***)** > **=** *variable* < *variable. . .* > **;**

The MODEL statement causes PROC HPSPLIT to create a tree model by using *response* as the response variable and *variable* as a predictor. By default, *variable* is treated as a continuous predictor if it is a numeric variable, or as a categorical variable if the variable also appears in the CLASS statement. The order of the variables in the MODEL statement can affect which split is chosen for a node in the case of equally performing splits.

NOTE: Specifying a character variable in a MODEL statement without previously declaring it in a CLASS statement results in an error.

You can specify the following *response-option*:

EVENT='*category***'**
specifies the event level for a binary categorical response variable. PROC HPSPLIT associates this level with the event of interest (sometimes referred to as the positive outcome) for the purpose of computing sensitivity, specificity, and area under the curve (AUC) and creating receiver operating characteristic (ROC) curves. You can specify the value (formatted if a format is applied) of the event category in quotation marks.

The default level is the first level of the response variable as specified by the ORDER= option in the CLASS statement.

NOTE: The EVENT= option has no effect in the case of categorical response variables that have more than two levels, and it has no effect in the case of continuous response variables.

OUTPUT Statement

OUTPUT OUT=*SAS-data-set* **;**

The OUTPUT statement creates a data set that contains one observation for each observation in the input data set. The OUT= data set contains the following:

- the response variable
- any variables that you specify by using the ID statement
- the observation's assigned leaf number
- the observation's assigned node number

You can use the leaf number in conjunction with the RULES statement to examine the tree in more detail. You can use the node number in conjunction with the table that is produced using the NODES option to investigate the tree.

In addition, for regression trees the OUT= data set contains the following:

- the prediction for this observation
- the average value within the observation's assigned leaf in the validation partition, if you specify a validation partition in the PARTITION statement

In addition, for classification trees the OUT= data set contains the following for each response variable level:

- the fraction of training partition weight within the assigned leaf for the observation
- the fraction of validation partition weight within the assigned leaf for the observation, if you specify a validation partition in the PARTITION statement

PARTITION Statement

PARTITION < *partition-options* > **;**

The PARTITION statement specifies how observations in the predictor data set are logically partitioned into disjoint subsets for model testing, training, and validation. Either you can designate a variable in the predictor data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations to each role.

NOTE: Attempting to use a PARTITION statement along with cross validation results in an error.

You can specify only one of the following *partition-options*:

FRACTION(TEST=*fraction* **VALIDATE=***fraction* **< SEED=***number* **>)**

requests that specified proportions of the observations in the predictor data set be randomly assigned to testing, training, and validation roles. You specify the proportions for testing and validation by using the VALIDATE= and the TEST= suboptions. The SEED= suboption sets the seed. Because *fraction* is a per-observation probability, setting *fraction* too low can result in an empty or nearly empty testing or validation set. If you omit either the TEST= or VALIDATE= suboption, then the partition statement is ignored.

Using the FRACTION option can cause different numbers of observations to be selected for the validation set because this option specifies a per-observation probability. Different partitions can be observed when the number of nodes or threads changes or when PROC HPSPLIT runs in alongside-the-database mode.

The following PARTITION statement shows how to use a probability of choosing a particular observation for the validation set:

```
partition fraction(validate=0.1 seed=1234);
```

In this example, any particular observation has a probability of 10% of being selected for the validation set. All nonselected records are in the training set. The SEED= suboption specifies the seed that is used for the random number generator. If you omit this suboption, the seed is obtained by generating the time of day from the computer's clock.

ROLEVAR=*variable* **(< TEST='***value***' > TRAIN='***value***' VALIDATE='***value***')**

names the *variable* in the predictor data set whose values are used to assign roles to each observation. You specify the formatted values of this *variable*, which are used to assign observation roles, in the TRAIN= and VALIDATE= suboptions. You can also assign observations to the test role by specifying the TEST= suboption. You must specify the TRAIN= suboption and either the VALID= or TEST= suboption.

In the following example, the ROLEVAR= option specifies _PARTIND_ as the *variable* in the predictor data set that is used to select the data set:

```
partition rolevar=_partind_(TRAIN='1' VALIDATE='0');
```

The TEST=, TRAIN=, and VALIDATE= suboptions provide the values that indicate whether an observation is in the testing, training, or validation sets, respectively. Observations in which the variable is missing or a value corresponds to none of the arguments are ignored. Formatting and normalization are performed before comparison, so you should specify numeric variable values as formatted values, as in the preceding example.

PERFORMANCE Statement

PERFORMANCE < *performance-options* > **;**

The PERFORMANCE statement defines performance parameters for multithreaded computing and requests detailed results about the performance characteristics of PROC HPSPLIT.

This statement is documented further in the section "PERFORMANCE Statement" (Chapter 2, *SAS Enterprise Miner: High-Performance Procedures*).

PRUNE Statement

PRUNE *prune-method* < **(***prune-options***)** > **;**

The PRUNE statement specifies the pruning method and related options. You can specify the following *prune-methods*:

C45 < **(***prune-option***)** >

requests C4.5 pruning (Quinlan 1993), which is based on the upper confidence limit for the error rate. For more information, see the section "Pruning" on page 5213. This pruning method is available only for classification trees (categorical responses). PROC HPSPLIT uses the error rate from the validation set, if one is available.

PROC HPSPLIT generates a subtree selection plot during pruning. The subtree selection plot shows the minimum change in prediction error as a function of the number of leaves in the subtree.

You can specify the following *prune-option*:

CONFIDENCE=*confidence-level*

specifies the pruning confidence level, which must be a positive number in the range of [0, 1]. The default confidence level is 0.25.

COSTCOMPLEXITY < **(***prune-options***)** >

CC < **(***prune-options***)** >

requests cost-complexity pruning (Breiman et al. 1984; Quinlan 1987; Zhang and Singer 2010). You can specify this pruning method for both classification trees and regression trees (continuous response). This is the default pruning method.

If you specify a validation set by using a PARTITION statement, PROC HPSPLIT uses the validation set for subtree selection. If you specify the number of leaves by using the LEAVES= option, the procedure selects the subtree that has the specified *number* of leaves, or if no subtree with exactly that *number* of leaves is available, it selects a subtree with fewer leaves. By default, if you do not specify a validation set by using a PARTITION statement and you do not use the LEAVES= option, the procedure uses k-fold cross validation for subtree selection.

For cost-complexity pruning, PROC HPSPLIT generates either a cost-complexity analysis plot based on cross validation or a cost-complexity pruning plot that shows the error metric for the training and validation sets when a validation set exists. The error metric is misclassification rate for classification trees and average square error (ASE) for regression trees.

You can specify the following *prune-option*:

LEAVES=*number* | **ALL**
specifies the subtree that has the requested *number* of leaves, or if no subtree with exactly that *number* of leaves is available, specifies a subtree with fewer leaves. When LEAVES=ALL, the largest tree is selected.

OFF
turns off pruning completely. No pruning is performed, and no pruning plots are generated.

REDUCEDERROR < (*prune-options*) >

REP < (*prune-options*) >
requests reduced-error pruning (Quinlan 1986). Reduced-error pruning has two stages: subtree sequence generation and subtree selection. For reduced-error pruning, if you specify a validation set by using a PARTITION statement, the validation set is used for subtree sequence generation and for subtree selection. Otherwise, the training set is used. For more information, see the section "Pruning" on page 5213.

For reduced-error pruning, PROC HPSPLIT generates a pruning plot that shows the requested error metric as a function of the number of leaves on the subtree.

You can specify the following *prune-options*:

LEAVES=*number* | **ALL**
specifies the subtree that has the requested *number* of leaves, or if no subtree with exactly that *number* of leaves is available, specifies a subtree with fewer leaves. When LEAVES=ALL, the largest tree is selected.

METRIC=ASE | **MISC**
specifies the metric for reduced-error pruning. Average square error (ASE) is the default metric. You can specify ASE for both classification trees and regression trees (continuous response). You can specify MISC only for classification trees (categorical response).

RULES Statement

RULES FILE=*filename* **;**

The RULES statement specifies a file for saving the rules that define the leaves for the final tree.

These rules can offer more details than are present in the tree plots and table that are produced using the NODES option in the PROC HPSPLIT statement. For example, if you have a large number of levels, the RULES file can provide extra space for those levels.

WEIGHT Statement

WEIGHT *variable* < / **UNWEIGHTDF** > **;**

The *variable* in the WEIGHT statement is used as a weight to perform a weighted analysis of the data. Observations with nonpositive or missing weights are not included in the analysis. If a WEIGHT statement is not included, then all observations used in the analysis are assigned a weight of 1.

You can specify the following:

UNWEIGHTDF
uses the unweighted calculation for the F statistic rather than the weighted calculation for the F statistic. For more information, see the section "Splitting Criteria" on page 5210.

Details: HPSPLIT Procedure

Building a Decision Tree

Algorithms for building a decision tree use the training data to split the predictor space (the set of all possible combinations of values of the predictor variables) into nonoverlapping regions. These regions correspond to the terminal nodes of the tree, which are also known as leaves.

Each region is described by a set of rules, and these rules are used to assign a new observation to a particular region. In the case of a classification tree, the predicted value for this observation is the most commonly occurring level of the response variable in that region; in the case of a regression tree, the predicted value is the mean of the values of the response variables in that region.

The splitting is done by recursive partitioning, starting with all the observations, which are represented by the node at the top of the tree. The algorithm splits this parent node into two or more child nodes in such a way that the responses within each child region are as similar as possible. The splitting process is then repeated for each of the child nodes, and the recursion continues until a stopping criterion is satisfied and the tree is fully built.

At each step, the split is determined by finding a best predictor variable and a best cutpoint (or set of cutpoints) that assign the observations in the parent node to the child nodes. The sections "Splitting Criteria" on page 5210 and "Splitting Strategy" on page 5213 provide details about the splitting methods available in the HPSPLIT procedure.

To illustrate the process, consider the first two splits for the classification tree in "Example 68.4: Creating a Binary Classification Tree with Validation Data" on page 5248, which is shown in Figure 68.11.

Figure 68.11 First Two Splits for Hmeq Data Set

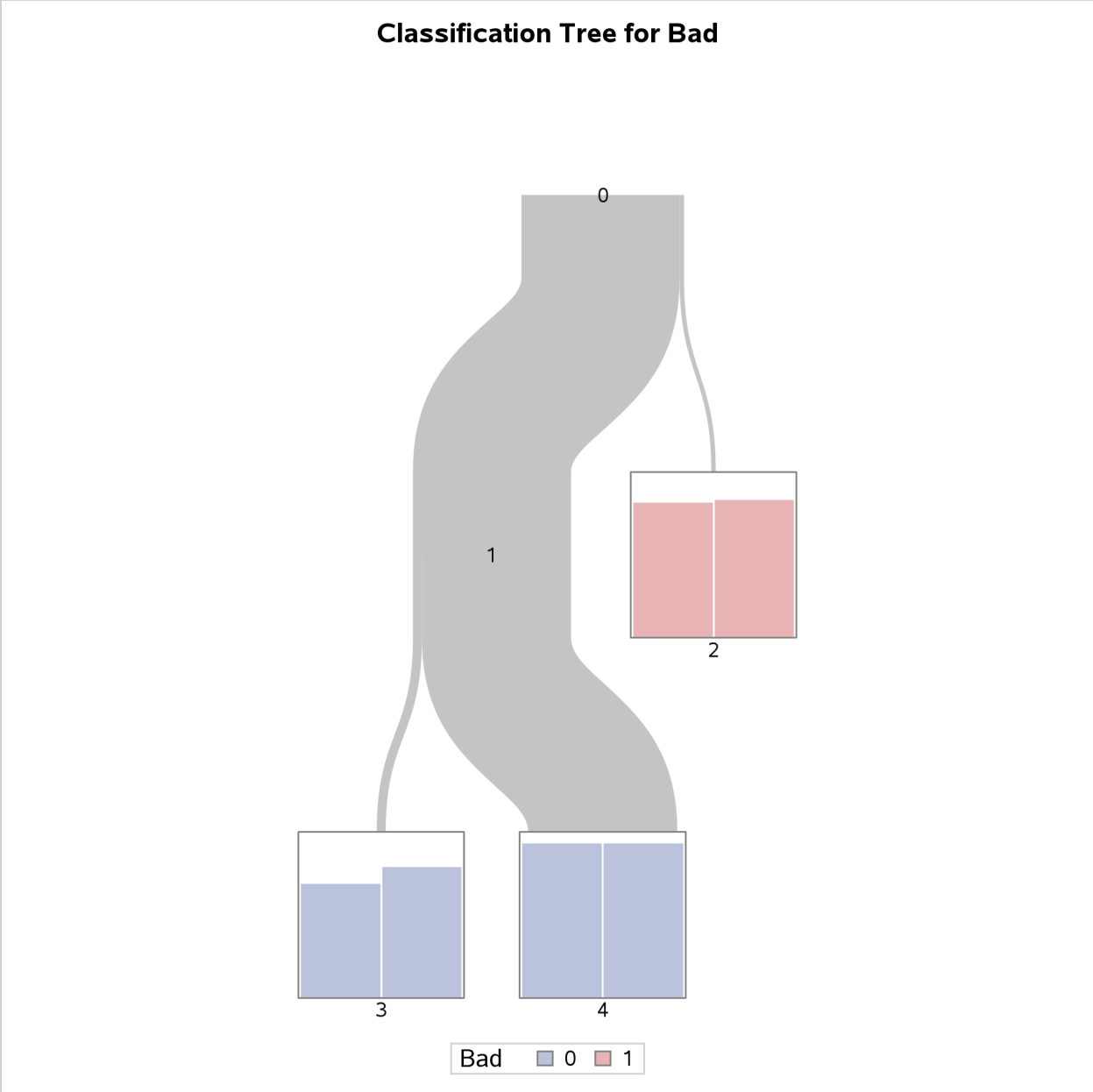

All 2,352 observations in the data are initially assigned to Node 0 at the top of the tree, which represents the entire predictor space. The algorithm then splits this space into two nonoverlapping regions, represented by Node 1 and Node 2. Observations with Debtinc < 48.8434 or with missing values of Debtinc are assigned to Node 1, and observations with Debtinc > 48.8434 are assigned to Node 2. Here the best variable, Debtinc, and the best cutpoint, 48.8434, are chosen to maximize the reduction in entropy as a measure of node impurity.

Next, the algorithm splits the region that is represented by Node 1 into two nonoverlapping regions, represented by Node 3 and Node 4. Observations with values of the categorical predictor variable Delinq equal to 2, 3, 4, 5, 6, or 7 are assigned to Node 3, and observations with Delinq equal to 0, 1, 8, 9, 10, 11, 12, 14, 15, or missing are assigned to Node 4. Note that the levels of Delinq in Node 4 are not adjacent to each other. Here the best variable, Delinq, and the values 2, 3, 4, 5, 6, and 7 are chosen to maximize the reduction in entropy.

Figure 68.12 Scatter Plot of the Predictor Space for the First Split

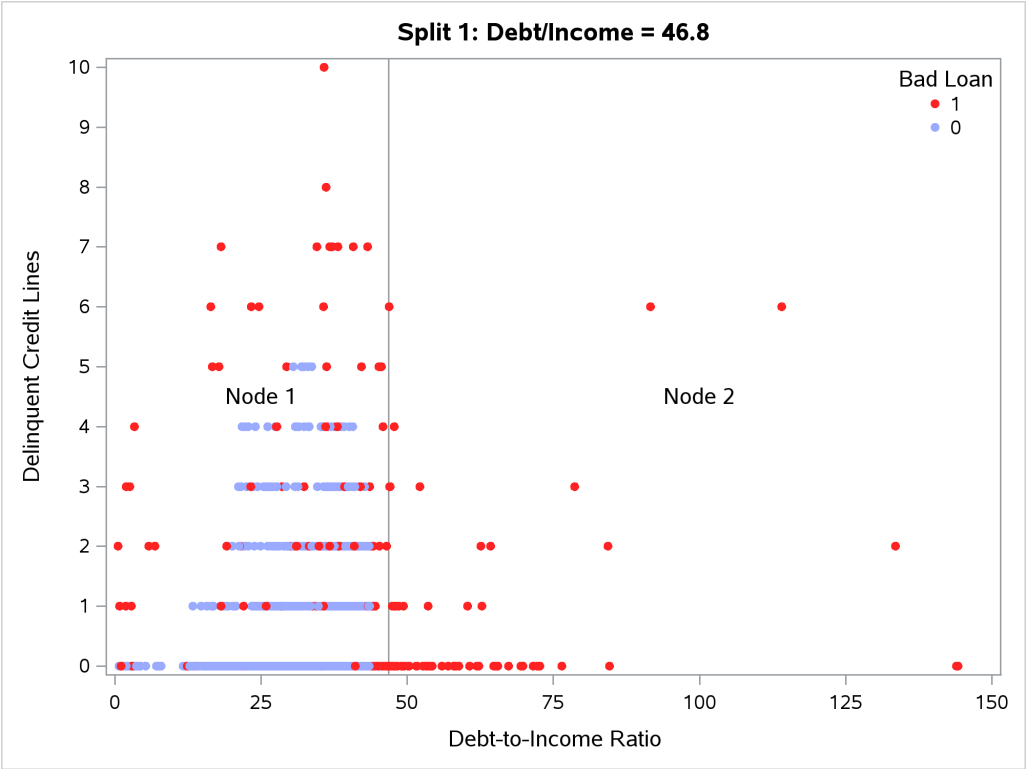

Figure 68.13 Scatter Plot of the Predictor Space for the Second Split

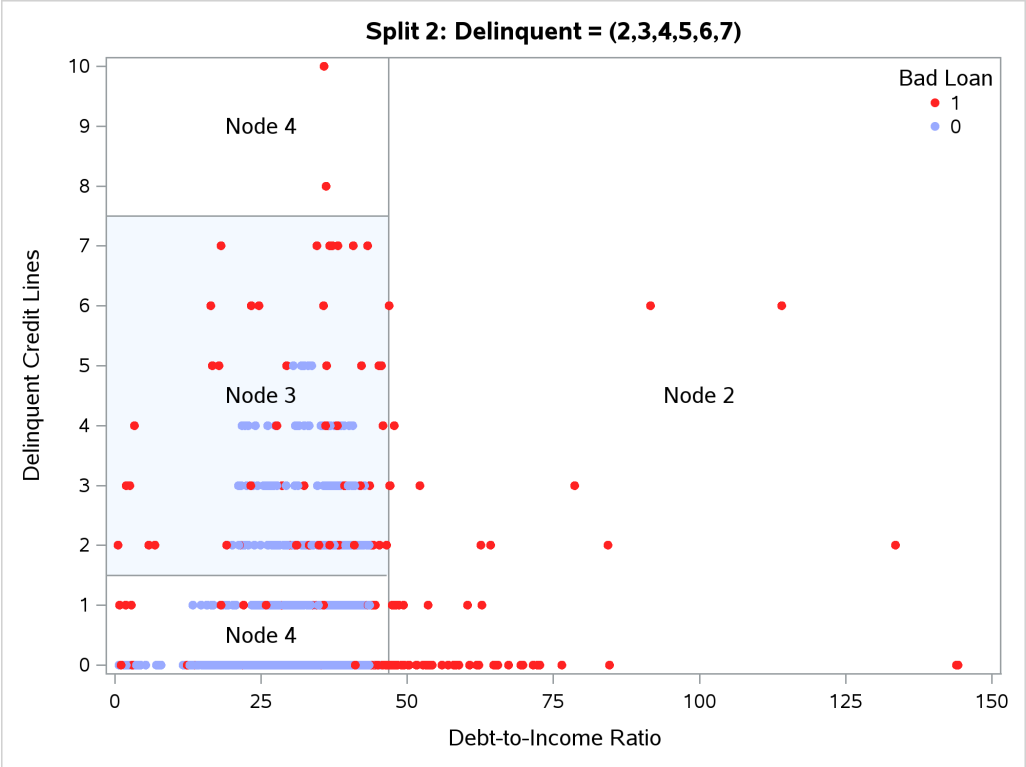

The tree that is defined by these two splits has three leaf (terminal) nodes, which are Nodes 2, 3, and 4 in Figure 68.13. Figure 68.12 and Figure 68.13 present scatter plots of the predictor space for these two splits one at a time. Notice in Figure 68.12 that the first split in Debt-to-Income Ratio divides the entire predictor space into Node 1 and Node 2, represented by two rectangular regions that have different ratios of events to nonevents for the response variable. In a similar way, notice in Figure 68.13 that the second split in Delinquent Credit Lines divides Node 1 into Node 3 and 4, which have different ratios of events to nonevents for the response variable. Also note that several observations have the same values for Debtinc or Delinq, giving the perception of a scatter plot that has fewer observations than there actually are.

For example, Node 4 has a very high proportion of observations for which Bad is equal to 0. In contrast, Bad is equal to 1 for all the observations in Node 2.

This example illustrates recursive binary splitting in which each parent node is split into two child nodes. By default, the HPSPLIT procedure finds two-way splits when building regression trees, and it finds k-way splits when building classification trees, where k is the number of levels of the categorical response variable. You can use the MAXBRANCH= option to specify the k-way splits of your tree.

Splitting Criteria

The goal of recursive partitioning, as described in the section "Building a Decision Tree" on page 5207, is to subdivide the predictor space in such a way that the response values for the observations in the terminal nodes are as similar as possible. The HPSPLIT procedure provides two types of criteria for splitting a parent node τ: criteria that maximize a decrease in node impurity, as defined by an impurity function, and criteria that are defined by a statistical test. You select the criterion by specifying an option in the GROW statement.

Criteria Based on Impurity

The entropy, Gini index, and RSS criteria decrease impurity. The impurity of a parent node τ is defined as $i(\tau)$, a nonnegative number that is equal to zero for a pure node—in other words, a node for which all the observations have the same value of the response variable. Parent nodes for which the observations have very different values of the response variable have a large impurity.

The HPSPLIT procedure selects the best splitting variable and the best cutoff value to produce the highest reduction in impurity,

$$\Delta i(s,\tau) = i(\tau) - \sum_{b=1}^{B} p(\tau_b|\tau) i(\tau_b)$$

where τ_b denotes the bth child node, $p(\tau_b|\tau)$ is the sum of the weights of observations in τ that are assigned to τ_b divided by the sum of the weights of observations in τ, and B is the number of branches after splitting τ.

If you specify a variable in the WEIGHT statement, then the weight of an observation is the value of the weight variable for that observation. If you omit the WEIGHT statement, then the weight of each observation is equal to 1. In this case, the sum of weights of observations is equal to the number of observations.

The impurity reduction criteria available for classification trees are based on different impurity functions $i(\tau)$ as follows:

- Entropy criterion (default)

  The entropy impurity of node τ is defined as

  $$i(\tau) = -\sum_{j=1}^{J} p_j \log_2 p_j$$

  where p_j is the sum of the weights of observations that have the jth response value divided by the sum of weights of all observations.

- Gini index criterion

  The Gini index criterion defines $i(\tau)$ as the Gini index that corresponds to the ASE of a class response and is given by

  $$i(\tau) = 1 - \sum_{j=1}^{J} p_j^2$$

  For more information, see Hastie, Tibshirani, and Friedman (2009).

The impurity reduction criterion available for regression trees is as follows:

- RSS criterion (default)

  The RSS criterion, also referred to as the ANOVA criterion, defines $i(\tau)$ as the residual sum of squares,

  $$i(\tau) = \frac{1}{N_w(\tau)} \sum_{i=1}^{N(\tau)} (Y_i - \overline{Y}_w)^2$$

  where $N(\tau)$ is the number of observations in τ, $N_w(\tau)$ is the sum of weights of all observations in τ, Y_i is the response value of observation i, $\overline{Y}_w$ is the weighted average response of the observations in τ, and w_i is the weight of observation i.

  The weighted average response is defined as

  $$\overline{Y}_w = \frac{\sum_{i=1}^{N(\tau)} w_i Y_i}{\sum_{i=1}^{N(\tau)} w_i}$$

Criteria Based on Statistical Test

The chi-square, F-test, CHAID, and FastCHAID criteria are defined by statistical tests. These criteria calculate the worth of a split by testing for a significant difference in the response variable across the branches defined by a split. The worth is defined as $-\log(p)$, where p is the p-value of the test. You can adjust the p-values for these criteria by specifying the BONFERRONI option in the GROW statement.

The criteria based on statistical tests compute the worth of a split as follows:

- Chi-square criterion

  For categorical response variables, the worth is based on the p-value for the Pearson chi-square test that compares the frequencies of the levels of the response across the child nodes.

- F-test criterion

  For continuous response variables, the worth is based on the F test for the null hypothesis that the means of the response values are identical across the child nodes. The test statistic is

$$F = \frac{\mathrm{SS}_{\mathrm{between}}/(B-1)}{\mathrm{SS}_{\mathrm{within}}/(N_w(\tau)-B)}$$

  where

$$\mathrm{SS}_{\mathrm{between}} = \sum_{b=1}^{B} N_w(\tau_b)(\overline{Y}_w(\tau_b) - \overline{Y}_w(\tau))^2$$

$$\mathrm{SS}_{\mathrm{within}} = \sum_{b=1}^{B} \sum_{i=1}^{N(\tau_b)} w_i(Y_{bi} - \overline{Y}_w(\tau_b))^2$$

  If you specify the UNWEIGHTDF option in the WEIGHT statement, then the degrees of freedom for the F test are not weighted. The $\mathrm{SS}_{\mathrm{between}}$ and $\mathrm{SS}_{\mathrm{within}}$ formulas are the same when the degrees of freedom are either weighted or unweighted. The F statistic with unweighted degrees of freedom is

$$F = \frac{\mathrm{SS}_{\mathrm{between}}/(B-1)}{\mathrm{SS}_{\mathrm{within}}/(N(\tau)-B)}$$

Available for both categorical and continuous response variables:

- CHAID criterion

  For categorical and continuous response variables, CHAID is an approach first described by Kass (1980) that regards every possible split as representing a test. CHAID tests the hypothesis of no association between the values of the response (target) and the branches of a node. The Bonferroni adjusted probability is defined as $m\alpha$, where α is the significance level of a test and m is the number of independent tests.

  For categorical response variables, the HPSPLIT procedure also provides the FastCHAID criterion, which is a special case of CHAID. FastCHAID is faster to compute because it sorts the possible splits according to the response variable.

Splitting Strategy

When you are building a tree, it is computationally intensive to calculate the node purity or the node worth for all possible split points of every predictor variable. For this reason, the HPSPLIT procedure implements a strategy that combines three different methods of generating candidate splits. The exhaustive method computes the split criterion for all the levels of a predictor variable. The greedy method, which is based on the CHAID algorithm, finds candidates for splits by recursively halving the data. The fast-sort method computes splits in order of the proportions of levels of a categorical response or the average of a continuous response.

By default, PROC HPSPLIT first tries to find candidates for splits by using the exhaustive method. If the number of computations exceeds the number that you specify in the LEVTHRESH1= or LEVTHRESH2= option, the procedure switches to the greedy algorithm. If the number of computations for a specified predictor runs out with the greedy algorithm, the procedure switches to the fast-sort method.

For categorical predictor variables, the exhaustive method tries to group levels in every possible combination. If the number of computations exceeds the threshold before this method finishes, the HPSPLIT procedure switches to the greedy algorithm, which groups levels by adding levels one at a time to the number of groups that you specify in the MAXBRANCH= option. If the number of computations exceeds the threshold before the greedy algorithm finishes, the procedure sorts the split points by the proportion of a categorical response in the predictor levels or by the average of a continuous response. Use the LEVTHRESH1= option to specify the number of computations for nominal predictors.

Continuous predictor variables are first binned in equidistant intervals as specified by the INTERVALBINS= option. Binning reduces the number of computations when you are searching for candidate splits. Use the option LEVTHRESH2= to specify the threshold for the number of computations before switching from exhaustive to fast-sort method. The greedy method is not available for interval predictor variables.

By default, a variable can be used more than once in a branch of a decision tree as long as the split is on a different value of that variable. You can specify the SPLITONCE option in the PROC HPSPLIT statement to request that variables be used only once per branch.

Pruning

The HPSPLIT procedure creates a classification or regression tree by first growing a tree as described in the section "Splitting Criteria" on page 5210. This usually results in a large tree that provides a good fit to the training data. The problem with this tree is its potential for overfitting the data: the tree can be tailored too specifically to the training data and not generalize well to new data. The solution is to find a smaller subtree that results in a low error rate on holdout or validation data.

It is often prohibitively expensive to evaluate the error on all possible subtrees of the full tree. A more practical strategy is to focus on a sequence of nested trees obtained by successively pruning leaves from the tree. Figure 68.14 shows an example of pruning in which the leaves (or terminal nodes) of Node 3 (Nodes 4 and 5) are removed to create a nested subtree of the full tree. In the nested subtree, Node 3 is now a leaf that contains all the observations that were previously in Nodes 4 and 5. This process is repeated until only the root node remains.

Figure 68.14 Tree and Pruned Subtree

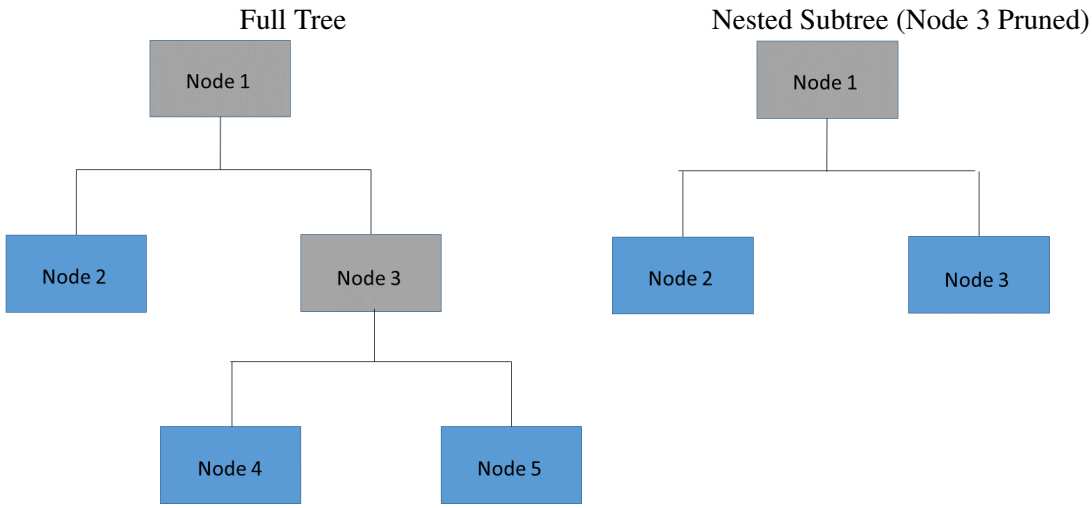

Many different methods have been proposed for pruning in this manner. These methods address both how to select which nodes to prune to create the sequence of subtrees and how then to select the optimal subtree from this sequence as the final tree. You can use the PRUNE statement in PROC HPSPLIT to specify which pruning method to apply and related options. Several well-known pruning methods, described in this section, are available, and you can override the final selected tree based on your preferences or domain knowledge.

Cost-Complexity Pruning

Cost-complexity pruning is a widely used pruning method that was originally proposed by Breiman et al. (1984). You can request cost-complexity pruning for either a categorical or continuous response variable by specifying

```
prune costcomplexity;
```

This algorithm is based on making a trade-off between the complexity (size) of a tree and the error rate to help prevent overfitting. Thus large trees with a low error rate are penalized in favor of smaller trees. The cost complexity of a tree T is defined as

$$\mathrm{CC}(T) = \mathrm{R}(T) + \alpha|T|$$

where $\mathrm{R}(T)$ represents its error rate, $|T|$ represents the number of leaves on T, and the complexity parameter α represents the cost of each leaf. For a categorical response variable, the misclassification rate is used for the error rate, $\mathrm{R}(T)$; for a continuous response variable, the residual sum of squares (RSS), also called the sum of square errors (SSE), is used for the error rate. Note that only the training data are used to evaluate cost complexity.

Breiman et al. (1984) show that for each value of α, there is a subtree of T that minimizes cost complexity. When $\alpha = 0$, this is the full tree, T_0. As α increases, the corresponding subtree becomes progressively smaller, and the subtrees are in fact nested. Then, at some value of α, the root node has the minimal cost complexity for any α greater than or equal to that value. Because there are a finite number of possible

subtrees, each subtree corresponds to an interval of values of α; that is,

$$
\begin{aligned}
[0, \alpha_1) &= \text{interval where } T_0 \text{ (the full tree) has minimal cost complexity} \\
[\alpha_1, \alpha_2) &= \text{interval where } T_1 \text{ has minimal cost complexity} \\
&\vdots \\
[\alpha_m, \infty) &= \text{interval where } T_m \text{ (the root node) has minimal cost complexity}
\end{aligned}
$$

PROC HPSPLIT uses weakest-link pruning, as described by Breiman et al. (1984), to create the sequence of $\alpha_1, \ldots, \alpha_m$ values and the corresponding sequence of nested subtrees, $T_1, \ldots, T_m$.

Finding the optimal subtree from this sequence is then a question of determining the optimal value of the complexity parameter α. This is performed either by using the validation partition, when you use the PARTITION statement to reserve a validation holdout sample, or by using cross validation. In the first case, the subtree in the pruning sequence that has the lowest validation error rate is selected as the final tree. When there is no validation partition, k-fold cross validation can be applied to cost-complexity pruning to select a subtree that generalizes well and does not overfit the training data (Breiman et al. 1984; Zhang and Singer 2010). The algorithm proceeds as follows after creating the sequence of subtrees and α values by using the entire set of training data as described earlier:

1. Randomly divide the training observations into k approximately equal-sized parts, or folds.
2. Define a sequence of β values as the geometric mean of the endpoints of the $[\alpha_i, \alpha_{i+1})$ intervals (that is, $\beta_i = \sqrt{\alpha_i \times \alpha_{i+1}}$) to represent the intervals.
3. For each of the k folds, hold out the current fold for validation and use the remaining $k - 1$ folds for the training data in the following steps:
   a) Grow a tree as done using the full training data set with the same splitting criterion.
   b) Using the $\beta_1, \ldots, \beta_m$ values that are calculated in step 2, create a sequence of subtrees for each β_i as described in the pruning steps given earlier, but now using β_i as a fixed value for α and minimizing the cost complexity, CC(T), to select a subtree at each pruning step.
   c) For each β_i, set T_{ij} to be the subtree that has the minimum cost complexity from the sequence for the jth fold.
   d) Calculate the error $\mathrm{ASE}(T_{ij})$ for each T_{ij} by using the current (jth) fold (the one omitted from the training).
4. Now the error rate can be averaged across folds, $\overline{\mathrm{ASE}}_i = \frac{1}{k} \sum_{j=1}^{k} \mathrm{ASE}(T_{ij})$, and the β_i that has the smallest $\overline{\mathrm{ASE}}_i$ is selected. The tree T_i from pruning the complete training data that corresponds to the selected β_i is the final selected subtree.

The HPSPLIT procedure provides two plots that you can use to tune and evaluate the pruning process: the cost-complexity analysis plot and the cost-complexity pruning plot.

When performing cost-complexity pruning with cross validation (that is, no PARTITION statement is specified), you should examine the cost-complexity analysis plot that is created by default. This plot displays $\overline{\mathrm{ASE}}_i$ as a function of the complexity parameters β_i, and it uses a vertical reference line to indicate the β_i

that minimizes $\overline{\mathrm{ASE}}_i$. You can use this plot to examine alternative choices for β_i. For example, you might prefer to select a smaller tree that has only a slightly higher error rate. The plot also gives you the information that you need to implement the 1-SE rule developed by Breiman et al. (1984).

You can use the LEAVES= option in the PRUNE statement to select a tree that has a specified number of leaves. This is illustrated in "Example 68.2: Cost-Complexity Pruning with Cross Validation" on page 5240.

When you specify validation data by using the PARTITION statement, the cost-complexity pruning plot displays the error rate R(T) as a function of the number of leaves $|T|$ for both the training and validation data. This plot indicates the final selected tree, the tree with the minimum R(T) for the validation data, by using a vertical reference line. Like the cost-complexity analysis plot that is produced when you perform cross validation, this plot can help you identify a smaller tree that has only a slightly higher validation error rate. You could then use the LEAVES= option in a subsequent run of PROC HPSPLIT to obtain the final selected tree that has the number of leaves that you specify. See Output 68.4.7 in "Example 68.4: Creating a Binary Classification Tree with Validation Data" on page 5248 for an example of this plot.

C4.5 Pruning

Quinlan (1987) first introduced pessimistic pruning as a pruning method for classification trees in which the estimate of the true error rate is increased by using a statistical correction in order to prevent overfitting. C4.5 pruning (Quinlan 1993) then evolved from pessimistic pruning to employ an even more pessimistic (that is, higher) estimate of the true error rate. An advantage of methods like pessimistic and C4.5 pruning is that they enable you to use all the data for training instead of requiring a holdout sample. In C4.5 pruning, the upper confidence limit of the true error rate based on the binomial distribution is used to estimate the error rate. The C4.5 algorithm variant that PROC HPSPLIT implements uses the beta distribution in place of the binomial distribution for estimating the upper confidence limit. This pruning method is available only for categorical response variables and is implemented when you specify

```
prune C45;
```

The C4.5 pruning method follows these steps:

1. Grow a tree from the training data set, and call this full, unpruned tree T_0.
2. Solve the following equation for p_l, the adjusted prediction error rate for leaf l, for each leaf in the tree:

   $$\alpha = 1 - \frac{\Gamma(N_l + 1)}{\Gamma(F_l + 1)\,\Gamma(N_l - F_l)} \int_0^{p_l} \nu^{F_l} (1 - \nu)^{N_l - F_l + 1}\, \mathrm{d}\nu$$

   Here the confidence level α is specified in the CONFIDENCE= option, F_l is the number of failures (misclassified observations) at leaf l, N_l is the number of observations at leaf l, and the function $\Gamma(x)$ is defined as

   $$\Gamma(x) = \int_0^{\infty} \nu^{x-1} e^{-\nu} \mathrm{d}\nu$$

3. Given these values of p_l, use the formula for the prediction error E of a tree T to calculate E_0 for the full tree T_0 before pruning:

   $$E = \sum_{l \in T} N_l\, p_l$$

4. Consider all nodes that have only leaves as children for pruning in tree T_0.

5. Calculate the prediction error for each possible subtree that is created by replacing a node with a leaf by using the equations from steps 2 and 3, and prune the node that creates the subtree that has the largest decrease (or smallest increase) in prediction error from tree T_0. Let this be the next subtree in the sequence, T_1.

6. Repeat steps 2–5 with subtree T_1 from step 5 as the new "full" tree to use for creating the next subtree in the sequence, T_2. Continue until only the root node remains, represented by T_m.

The change in error is calculated between each pair of consecutive subtrees, $\Delta_i = E_i - E_{i-1}$. For the first change Δ_i that is greater than 0, for $i = 1, \ldots, m$, subtree T_{i-1} is selected as the final subtree. Note that for subtree selection, the change in error Δ_i is calculated on the validation partition when it exists; otherwise the training data are used.

Reduced-Error Pruning

Quinlan's reduced-error pruning (1987) performs pruning and subtree selection based on minimizing the error rate in the validation partition at each pruning step and then in the overall subtree sequence. This is usually based on the misclassification rate for a categorical response variable, but the average square error (ASE) can also be used. For a continuous response, the error is measured by the ASE. To implement reduced-error pruning, you can use the following PRUNE statement:

```
prune reducederror;
```

This pruning algorithm is implemented as follows, starting with the full tree, T_0:

1. Consider all nodes that have only leaves as children for pruning in tree T_0.

2. For each subtree that is created by replacing a node from step 1 with a leaf, calculate the error rate by using the validation data when available (otherwise use the training data).

3. Replace the node that has the smallest error rate with a leaf, and let T_1 be the subtree.

4. Repeat steps 1–3 with subtree T_1 from step 3 as the new "full" tree to use for creating the next subtree in the sequence, T_2. Continue until only the root node remains, represented by T_m.

This algorithm creates a sequence of subtrees from the largest tree, T_0, to the root node, T_m. The subtree that has the smallest validation error is then selected as the final subtree. Note that using this method without validation data results in the largest tree being selected. Also note that pruning could be stopped as soon as the error starts to increase in the validation data as originally described by Quinlan; continuing to prune to create a subtree sequence back to the root node enables you to select a smaller tree that still has an acceptable error rate, as discussed in the next section.

User Specification of Subtree

There might be situations in which you want to select a different tree from the one selected by default but using cost-complexity or reduced-error pruning to create the sequence of subtrees. For example, maybe there is a subtree that has a slightly larger error but is smaller, and thus simpler, than the subtree with the minimum ASE that was selected using reduced-error pruning. In that case, you can override the selected subtree and instead select the subtree with *n* leaves that was created using cost-complexity or reduced-error pruning, where *n* is specified in the LEAVES= option in the PRUNE statement. You can implement the 1-SE rule for cost-complexity pruning described by Breiman et al. (1984) by using this option, as shown in "Example 68.2: Cost-Complexity Pruning with Cross Validation" on page 5240. Alternatively, you might want to select the largest tree that is created. In this case, you have two options: you can specify LEAVES=ALL in the PRUNE statement to still see the statistics for the sequence of subtrees created according to the specified pruning error measure, but the tree with no pruning performed is selected as the final subtree; or you can specify

```
prune off;
```

to select the largest tree with no pruning performed, meaning that statistics are not calculated and plots are not created on a sequence of subtrees.

Memory Considerations

The HPSPLIT procedure is designed for high-performance computing. As a result, it does not create utility files but rather stores all the data in memory. Data sets that have a large number of predictor variables and a large number of response levels can cause PROC HPSPLIT to run out of memory. One way to overcome this problem is to give SAS more memory to use. Another way is to use fewer threads, which reduces the memory that is required. You can use the NTHREADS= option in the PERFORMANCE statement to specify the number of threads. For more information, see the section "PERFORMANCE Statement" (Chapter 2, *SAS Enterprise Miner: High-Performance Procedures*).

Primary and Surrogate Splitting Rules

The HPSPLIT procedure calculates primary and surrogate splitting rules for assigning the observations in a node to a branch. Both types of splitting rules use the value of a single predictor variable to assign an observation to a branch.

A primary splitting rule is always calculated by default, and it provides for the assignment of observations when the predictor variable is missing, even when there are no missing values in the training data. This allows for the possibility of missing values when you are scoring new data.

In addition, you can request one or more surrogate splitting rules by using the NSURROGATES= option. The purpose of a surrogate rule is to handle the assignment of observations by using an alternative variable that has similar predictive ability and has nonmissing values in observations where the primary predictor is missing. Surrogate rules enable you to make better use of the data. The HPSPLIT procedure uses the method of Breiman et al. (1984) to determine surrogate rules. By default, NSURROGATES=0. If you request one or more surrogate rules, the last column of the "Variable Importance" table shows the number of times that a variable is used as a surrogate. If a variable is used as a surrogate, you can see exactly how it is used in the SAS DATA step code that is generated when you specify the CODE statement.

When you request more than one surrogate rule, the rule that is applied to an observation is the first rule for which the surrogate variable is nonmissing. For example, consider the set of rules in Table 68.2 for a particular node in a decision tree where X, Y, and Z are three continuous predictors.

Table 68.2 Example of Splitting Rules

Rule	Description
Primary	If (NOT MISSING(X)) then (if(X < 1) then (assign to branch 2) else (assign to branch 1))
Surrogate 1	Else if (MISSING(X) and NOT MISSING(Y)) then (if(Y < 0) then (assign to branch 1) else (assign to branch 2))
Surrogate 2	Else if (MISSING(X) and MISSING(Y) and NOT MISSING(Z)) then (if(Z < 100) then (assign to branch 2) else (assign to branch 1))
Default	Else assign to branch 2

Note that the default rule ensures that observations that have missing values are always assigned to a branch regardless of whether you request surrogate rules.

PROC HPSPLIT selects a surrogate rule whose alternative predictor variable has the largest agreement with the predictor variable for the primary split. The measure of agreement between primary and surrogate splitting rules is the proportion of nonmissing observations (that is, observations without a missing value in the predictor variables) in the within-node training sample that the two rules assign to the same branch.

Handling Missing Values

Observations for which the response variable is missing are omitted from the analysis. The HPSPLIT procedure provides various methods of handling missing values of predictor variables.

By default, observations for which predictor variables are missing are omitted from the analysis. This behavior is common to other statistical modeling procedures in SAS/STAT software. Alternatively, you can use the ASSIGNMISSING= option to request different methods of dealing with missing values of predictor variables.

The ASSIGNMISSING=BRANCH option creates an extra branch for missing values. You determine the maximum number of branches by using the MAXBRANCH= option, which specifies the maximum number of branches per node in the tree. By default, MAXBRANCH=2. The ASSIGNMISSING=BRANCH option has no effect if there are no missing values in the training data set for a particular split. However, even if this is not the case, there could be missing values in the data set that is used for scoring, so this option is used to assign missing values that could be encountered in the future. For more information, see the section "Scoring" on page 5220.

The ASSIGNMISSING=POPULAR option assigns missing values to the most popular node of a split. If two or more branches have the same maximum number of observations, then the missing values are assigned to the branch that has the lowest node index.

The ASSIGNMISSING=SIMILAR option assigns missing values to the most similar node. Similarity is calculated using a chi-square test for categorical response variables and an F test for continuous response variables. The ASSIGNMISSING=SIMILAR option has no effect if there are no missing values in the

training data set for a certain split. To handle this case, PROC HPSPLIT assigns future missing values to the most popular node of a split.

Unknown Values of Categorical Predictors

Unseen and infrequent levels of categorical predictors are referred to as unknown values, and they present problems for building decision trees in two situations. In the first situation, categorical predictors in validation or scoring data contain levels that do not occur in the training data. Unseen levels of this type are treated as missing values by splitting rules. In the second situation, categorical predictors in training data contain levels that occur very rarely and might be considered as outliers.

You can use the MINCATSIZE= option to specify the minimum number of occurrences that are required for a level to be used in splitting. By default, MINCATSIZE=1. By specifying a number greater than 1, you can filter out rarely occurring levels.

Note that for continuous predictors, the splitting rules always provide for the assignment of values regardless of whether they are unseen or infrequent. For example, a decision tree might have a branch based on the range "(Age > 20) AND (Age < 60)". Here "Age" is a continuous predictor. Any value of "Age" that is not in this range can be assigned to another branch of that decision tree.

Scoring

After you create a tree model, you can apply it to the training data for model diagnosis or to new data in order to make predictions. The process of applying a model to a data set is called *scoring*. You can use scoring to improve or deploy your model. There are two approaches to using PROC HPSPLIT to score a data set.

With the first approach, you can use the OUTPUT statement to score the training data. Usually, the purpose of scoring a training data set is to diagnose the model. The training data set is the data set that you specify by using the DATA= option. When scoring the training data, PROC HPSPLIT creates an output data set that contains one observation for each observation in the training data. You can specify the output data set by using the OUT= option in the OUTPUT statement.

In the following example, the input data set is scored after the tree model has been created:

```
proc hpsplit data=Sampsio.Hmeq;
   class Bad Delinq Derog Job Ninq Reason;
   model Bad = Delinq Derog Job Ninq Reason;
   output out=scored;
run;
```

With the second approach to scoring, the HPSPLIT procedure generates SAS DATA step code that you can use to score new data. Usually, the purpose of scoring new data is to make predictions. PROC HPSPLIT generates SAS DATA step code when you specify the CODE statement. For more information, see the section "Creating Score Code and Scoring New Data" in "Example 68.4: Creating a Binary Classification Tree with Validation Data" on page 5248.

With either approach, scoring a data set creates a data set that contains the new variables shown in Table 68.3.

Table 68.3 Scoring Variables

Variable	Description
Leaf	Leaf number to which the observation is assigned
Node	Node number to which the observation is assigned

In addition, for classification trees, the scored data set also contains new variables with the prefix "P_" and can contain new variables with the prefix "V_". There is one "P_" variable that corresponds to each level of the response variable. For all observations in the same leaf, these variables represent the proportion of the training observations in that leaf that have that particular response level. Similarly, if validation data are used, the variables that have the prefix "V_" represent the proportion of the validation observations in a leaf that have the corresponding response. For example, if the name of the categorical response variable is Color and it has two levels, `'Blue'` and `'Green'`, then the scored data set contains the variables P_ColorBlue and P_ColorGreen, which provide the proportions of training data in this leaf that have the response levels `'Blue'` and `'Green'`. If you use validation data, the HPSPLIT procedure creates two more new variables, V_ColorBlue and V_ColorGreen, which provide the proportions of validation data in this leaf that have the response levels `'Blue'` and `'Green'`.

For regression trees, the scored data set contains a new variable with the prefix "P_" and can contain another new variable with the prefix "V_". For all observations in the same leaf, the new variable that has the prefix "P_" represents the average value of the response variable for those observations. If you use validation data, the new variable with the prefix "V_" represents the average value of the response variable for the validation observations in the same leaf. For example, if the name of the continuous response variable is logSalary, then the scored data set contains a new variable, P_logSalary, to represent the average value of the response variable logSalary in the training data for observations on the same leaf. If you use validation data, the HPSPLIT procedure creates another new variable, V_logSalary, which provides the average value of the response variable for validation observations on the same leaf.

Measures of Model Fit

Various measures of model fit have been proposed in the data mining literature. The HPSPLIT procedure measures model fit based on a number of metrics for classification trees and regression trees.

If you specify a variable in the WEIGHT statement, then the weight of an observation is the value of the weight variable for that observation. If no WEIGHT statement is specified, then the weight of each observation is equal to one. In this case, the sum of weights of observations is equal to the number of observations.

Measures of Model Fit for Classification Trees

The HPSPLIT procedure measures model fit based on the following metrics for classification tree: entropy, Gini index, misclassification rate (Misc), residual sum of squares (RSS), average square error (ASE, also known as the Brier score), sensitivity, specificity, area under the curve (AUC), and confusion matrix.

Entropy for Classification Trees

Entropy for classifications tree is defined as

$$\text{Entropy} = -\sum_{\lambda} \frac{N_{w\lambda}}{N_{w0}} \sum_{\tau} \frac{N^{\lambda}_{w\tau}}{N_{w\lambda}} \log_2 \left(\frac{N^{\lambda}_{w\tau}}{N_{w\lambda}} \right)$$

where

- λ is a leaf
- $N_{w\lambda}$ is the sum of weights of observations on leaf λ
- N_{w0} is the total sum of weights of observations in the entire data set
- τ is a level of the response variable
- $N^{\lambda}_{w\tau}$ is the sum of weights of observations on leaf λ that have response level τ

Gini Index for Classification Trees

The Gini index for classification trees is defined as

$$\text{Gini} = \sum_{\lambda} \frac{N_{w\lambda}}{N_{w0}} \sum_{\tau} \frac{N^{\lambda}_{w\tau}}{N_{w\lambda}} \left(1 - \frac{N^{\lambda}_{w\tau}}{N_{w\lambda}} \right)$$

Misclassification Rate for Classification Trees

Misclassification (Misc) comes from the number of incorrectly predicted observations. It is defined as

$$\text{Misc} = \frac{1}{N_{w0}} \sum \begin{cases} 0 & \text{if prediction is correct} \\ w_i & \text{otherwise} \end{cases}$$

Residual Sum of Squares for Classification Trees

The residual sum of squares (RSS) for classification trees is defined as

$$\text{RSS} = \sum_{\lambda} \sum_{\Phi} N^{\lambda}_{w\Phi} \left[\sum_{\tau \neq \Phi} \left(P^{\lambda}_{w\tau} \right)^2 + \left(1 - P^{\lambda}_{w\Phi} \right)^2 \right]$$

where

- Φ is the actual response level
- N^{λ}_{Φ} is the number of observations on leaf λ that have response level Φ
- $P^{\lambda}_{w\tau}$ is the weighted posterior probability for response level τ on leaf λ,

  $$P_{w\lambda} = \frac{N^{\lambda}_{w\tau}}{N_{w\lambda}}$$

- $P^{\lambda}_{w\Phi}$ is the weighted posterior probability for the actual response level Φ on leaf λ,

  $$P_{w\Phi} = \frac{N^{\Phi}_{w\tau}}{N_{w\lambda}}$$

Average Square Error for Classification Trees

The average square error (ASE) is also known as the Brier score for classification trees. It is defined as

$$\mathrm{ASE} = \frac{\mathrm{RSS}}{N_{w0} N_T}$$

where N_T is the number of levels for the response variable.

Sensitivity for Binary Classification Trees

Sensitivity is the probability of predicting an event for the response variable when the actual state is an event. For example, if the event is "an individual is sick," then sensitivity is the probability of predicting that an individual is sick given that the individual is actually sick. For binary classification trees, it is defined as

$$\mathrm{Sensitivity} = \frac{\mathrm{TP_w}}{P_w}$$

where

- TP is the sum of weights of true positives (predicting that an individual is sick)
- P is the sum of weights of positive observations (sick individuals)

Specificity for Binary Classification Trees

Specificity is the probability of predicting a nonevent for the response variable when the actual state is a nonevent. For example, if the event is "an individual is sick," then specificity is the probability of predicting that an individual is not sick given the fact that the individual is actually not sick. For a binary classification tree, specificity is defined as

$$\mathrm{Specificity} = \frac{\mathrm{TN_w}}{N_w}$$

where

- TN is the sum of weights of true negatives (predicting that an individual is not sick)
- N is the sum of weights of negative observations (healthy individuals)

Area under the Curve for Binary Classification Trees

Area under the curve (AUC) is defined as the area under the receiver operating characteristic (ROC) curve. PROC HPSPLIT uses sensitivity as the Y axis and 1 – specificity as the X axis to draw the ROC curve. AUC is calculated by trapezoidal rule integration,

$$\mathrm{AUC} = \frac{1}{2} \sum_{\lambda} ((x_\lambda - x_{\lambda-1})(y_\lambda + y_{\lambda-1}))$$

where

- y_λ is the sensitivity value at leaf λ
- x_λ is the 1 – specificity value at leaf λ

NOTE: For a binary response, the event level that is used for calculating sensitivity, specificity, and AUC is specified in the EVENT= option in the MODEL statement.

Confusion Matrix for Classification Trees

A confusion matrix is also known as a contingency table. It contains information about actual values and predicted values from a classification tree. A confusion matrix has N_T rows and N_T columns, where each row corresponds to the actual response level and each column corresponds to the predicted response level. The values in the matrix represent the number of observations that have the actual response represented in the row and the predicted response represented in the column. The error rate per actual response level is also reported,

$$\text{ErrorRate} = \frac{N_{ww}}{N_{w\Phi}}$$

where

- N_{ww} is the sum of weights of wrong predictions
- $N_{w\Phi}$ is the sum of weights of observations that have response level Φ

Measures of Model Fit for Regression Trees

The HPSPLIT procedure measures model fit for regression trees based on RSS and ASE.

Residual Sum of Squares for Regression Trees

The residual sum of squares (RSS) for regression trees is defined as

$$\text{RSS} = \sum_{\lambda} \sum_{i \in \lambda} w_i \left(y_i - \hat{y}_{\lambda}^{T} \right)^2$$

where

- i is an observation on leaf λ
- y_i is the predicted value of the response variable of observation i
- $\hat{y}_{\lambda}^{T}$ is the actual value of the response variable on leaf λ

Average Square Error for Regression Trees

The average square error (ASE) for regression trees is defined as

$$\text{ASE} = \frac{\text{RSS}}{N_{w0}}$$

Variable Importance

A training data set can contain a large number of predictors. Some predictors are useful for predicting the response variable, and others are not. You can use the HPSPLIT procedure to select the most useful predictors based on variable importance (Breiman et al. 1984). (See "Example 68.5: Assessing Variable Importance" on page 5257.) Variable importance is an indication of which predictors are most useful for predicting the response variable. Various measures of variable importance have been proposed in the data mining literature.

The most important variables might not be the ones near the top of the tree. PROC HPSPLIT measures variable importance based on the following metrics: count, surrogate count, RSS, and relative importance. The count-based variable importance simply counts the number of times in the tree that a particular variable is used in a split. Similarly, the surrogate count tallies the number of times that a variable is used in a surrogate splitting rule.

The RSS-based metric measures variable importance based on the change of RSS when a split is found at a node. The change is

$$\Delta_d = \mathrm{RSS}_d - \sum_i \mathrm{RSS}_i^d$$

where

- d denotes the node
- i denotes the index of a child that this node has
- RSS_d is the RSS if the node is treated as a leaf
- RSS_i^d is the RSS of the node after it has been split

If the change in RSS is negative (which is possible when you use the validation set), then the change is set to 0.

If surrogate rules are in effect, they are also credited with a portion of the change in RSS. The credit is proportional to the agreement between the primary and surrogate splitting rules at the node. The agreement at node d, κ_d, is defined as

$$\kappa_d = \sum_i \frac{N_i}{N_d}$$

where

- N_d is the number of nonmissing observations
- N_i is the number of observations that were assigned to i by both the primary and surrogate rules

The change in RSS from the surrogate rules is defined as

$$\Delta_d = \kappa_d \left(\mathrm{RSS}_d - \sum_i \mathrm{RSS}_i^d \right)$$

The RSS-based importance is then defined as

$$\sqrt{\sum_{d=1}^{D} \Delta_d}$$

where D is the total number of nodes.

The relative importance metric is a number between 0 and 1. It is calculated in two steps. First, PROC HPSPLIT finds the maximum RSS-based variable importance. Then, for each variable, it calculates the relative variable importance as the RSS-based importance of this variable divided by the maximum RSS-based importance among all the variables.

The RSS and relative importance are calculated from the training set. They are calculated again from the validation set if one exists.

ODS Table Names

Each table that the HPSPLIT procedure creates has a name associated with it, and you must use this name to refer to the table when you use ODS statements. These names are listed in Table 68.4.

Table 68.4 ODS Tables Produced by PROC HPSPLIT

Table Name	Description	Required Statement / Option
ConfusionMatrix	One or more confusion matrices	Default output for categorical response variable
CrossValidatedModel	Model assessment with cross validation	PROC / CVMODELFIT option
CrossValidationValues	Results for cost-complexity pruning based on cross validation	PROC / CVCC option
CVModelConfusionMatrix	Cross validation confusion matrix	PROC / CVMODELFIT option
DataAccessInfo	Information about modes of data access	Default output
ModelInfo	Information about the modeling environment	Default output
NodeTable	Node information	PROC / NODES option
NObs	Observation information	Default output
PerformanceInfo	Information about high-performance computing environment	Default output
Timing	Absolute and relative times for tasks performed by the procedure	PERFORMANCE / DETAILS option
TreePerformance	Fit statistics for the selected tree	Default output
VarImportance	Variable importance	Default output

ODS Graphics

You can refer to every graph that is produced through ODS Graphics by name. The names of the graphs that PROC HPSPLIT generates are listed in Table 68.5, along with the relevant PLOTS= options.

Table 68.5 Graphs Produced by PROC HPSPLIT

ODS Graph Name	Plot Description	PLOTS Option
CrossValidationASEPlot	Cross validation cost-complexity ASE plot	CVCC
PruningPlot	Plot of the error sum of squares, misclassification rate, or cost complexity when it is used for final tree selection	PRUNEUNTIL
ROCPlot	Plot of receiver operating characteristic (ROC) curve	ROC
WholeTreePlot	Overview plot of final tree	WHOLETREE
ZoomedTreePlot	Detailed plot of portion of final tree	ZOOMEDTREE

SAS Enterprise Miner Syntax and Notes

In addition to the syntax that is described in the CLASS and MODEL statement sections, PROC HPSPLIT supports SAS Enterprise Miner INPUT/TARGET syntax that many Enterprise Miner users are familiar with. The INPUT/TARGET syntax cannot be used together with the CLASS/MODEL syntax of SAS/STAT. Doing so is an error.

Enterprise Miner style syntax has one TARGET statement and one or more INPUT statements. If you use the Enterprise Miner syntax, then the PROC HPSPLIT statement, the TARGET statement, and the INPUT statement are required. Depending on the options in those statements, specified variables can be interval or nominal. By default, numeric INPUT variables are treated as interval (or continuous) predictors, and character INPUT variables are treated as nominal (or categorical) predictors.

INPUT *variables* < / *option* > **;**
TARGET *variable* < / *option* > **;**

INPUT Statement

INPUT *variables* < / *option* > **;**

The INPUT statement specifies predictor *variables* for the decision tree or regression tree. The value of *variable* can be a range such as “var_1–var_1000” or the special “_ALL_” value to include all variables in the data set. As with CLASS variables, all nominal INPUT variables are padded or truncated to 32 characters.

It is an error to use an INPUT statement with a MODEL or CLASS statement.

You can specify the following *option*:

LEVEL=INT | NOM
specifies whether the specified predictor *variables* are interval or nominal.

INT
treats all numeric *variables* as interval predictors.

NOM
treats all *variables* as nominal predictors.

By default, numeric *variables* are treated as interval predictors, and character *variables* are treated as nominal predictors. Specifying LEVEL=NOM forces all *variables* in that statement to be treated as nominal. PROC HPSPLIT ignores the LEVEL=INT option for character variables.

TARGET Statement

TARGET *variable* < / *options* > **;**

The TARGET statement names the *variable* whose values PROC HPSPLIT tries to predict. Missing values in the target are ignored except during scoring.

It is an error to use a TARGET statement with a MODEL or CLASS statement.

You can specify the following *options*:

LEVEL=INT | NOM
specifies whether the specified response *variable* is interval or nominal.

INT
treats the response as an interval variable and creates a regression tree.

NOM
treats the response as a nominal variable and creates a decision tree.

By default, LEVEL=NOM, and PROC HPSPLIT creates a decision tree (nominal response).

ORDER=*ordering*
ensures that the response values are levelized in the specified order. You can specify the following values:

ASC | ASCENDING levelizes response values in ascending order.

DESC | DESCENDING levelizes response values in descending order.

FMTASC | ASCFORMATTED levelizes response values in ascending order of the formatted value.

FMTDESC | DESFORMATTED levelizes response values in descending order of the formatted value.

By default, ORDER=DESC.

Example Classification Tree Syntax for SAS/STAT and SAS Enterprise Miner

The following two programs are equivalent. The first is based on the syntax in the section "Syntax: HPSPLIT Procedure" on page 5189, and the second is SAS Enterprise Miner syntax.

```
proc hpsplit data=sashelp.cars;
   class enginesize model;
   model enginesize = mpg_highway model;
run;

proc hpsplit data=sashelp.cars;
   input mpg_highway model;
   target enginesize;
run;
```

Example Regression Tree Syntax for SAS/STAT and SAS Enterprise Miner

The following two programs are equivalent. The first is based on the syntax in the section "Syntax: HPSPLIT Procedure" on page 5189, and the second is SAS Enterprise Miner syntax.

```
proc hpsplit data=sashelp.cars;
   class model;
   model enginesize = mpg_highway model;
run;

proc hpsplit data=sashelp.cars;
   input mpg_highway model;
   target enginesize / level=int;
run;
```

Note for SAS Enterprise Miner Users

NOTE: The RSS splitting criterion is also known as the variance splitting criterion.

Examples: HPSPLIT Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 68.1: Building a Classification Tree for a Binary Outcome

(View the complete code for this example (hpsplex1.sas) in the example repository.)

This example illustrates how you can use the HPSPLIT procedure to build and assess a classification tree for a binary outcome. Overfitting is avoided by cost-complexity pruning, and the selection of the pruning parameter is based on cross validation.

The training data in this example come from the Lichen Air Quality Surveys (Geiser and Neitlich 2007), which were conducted in western Oregon and Washington between 1994 and 2001. Both the training data and the test data in this example were provided by Richard Cutler, Department of Mathematics and Statistics, Utah State University.

The LAQ data set consists of 30 measurements of environmental conditions, such as temperature, elevation, and moisture, at 840 sites. These variables are treated as predictors for the response variable LobaOreg, which is coded as 1 if the lichen species *Lobaria oregana* was present at the site and 0 otherwise.

```
proc print data=sampsio.LAQ(obs=5);
   var LobaOreg MinMinTemp Aconif PrecipAve Elevation ReserveStatus;
run;
```

Output 68.1.1 lists the first five observations for six of the variables in LAQ.

Output 68.1.1 Partial Listing of LAQ

Obs	LobaOreg	MinMinTemp	Aconif	PrecipAve	Elevation	ReserveStatus
1	0	-5.970	44.897	89.623	1567	Matrix
2	0	-6.430	81.585	91.231	1673	Reserve
3	1	-0.893	229.330	154.610	685	Reserve
4	0	-7.476	45.875	110.330	1971	Reserve
5	0	-5.992	81.679	98.739	1597	Reserve

The following statements invoke the HPSPLIT procedure to create a classification tree for LobaOreg:

```
ods graphics on;

proc hpsplit data=sampsio.LAQ seed=123;
   class LobaOreg ReserveStatus;
   model LobaOreg (event='1') =
      Aconif DegreeDays TransAspect Slope Elevation PctBroadLeafCov
      PctConifCov PctVegCov TreeBiomass EvapoTransAve EvapoTransDiff
      MoistIndexAve MoistIndexDiff PrecipAve PrecipDiff RelHumidAve
      RelHumidDiff PotGlobRadAve PotGlobRadDiff AveTempAve AveTempDiff
      DayTempAve DayTempDiff MinMinTemp MaxMaxTemp AmbVapPressAve
      AmbVapPressDiff SatVapPressAve SatVapPressDiff ReserveStatus;
   grow entropy;
   prune costcomplexity;
run;
```

In the case of binary outcomes, the EVENT= option is used to explicitly control the level of the response variable that represents the event of interest for computing the area under the curve (AUC), sensitivity, specificity, and values of the receiver operating characteristic (ROC) curves.

NOTE: These fit statistics do not apply to categorical response variables that have more than two levels, so the EVENT= option does not apply in that situation. Likewise, this option does not apply to continuous response variables.

The GROW statement specifies the entropy criterion for splitting the observations during the process of recursive partitioning that results in a large initial tree. The PRUNE statement requests cost-complexity pruning to select a smaller subtree that avoids overfitting the data.

The plot in Output 68.1.2 displays estimates of the misclassification rate for a series of progressively smaller subtrees of the large tree. The subtrees are indexed by a cost-complexity parameter (also called a pruning parameter or a tuning parameter). The plot provides a tool for selecting the parameter that results in the smallest misclassification rate.

Output 68.1.2 Misclassification Rate as a Function of Cost-Complexity Parameter

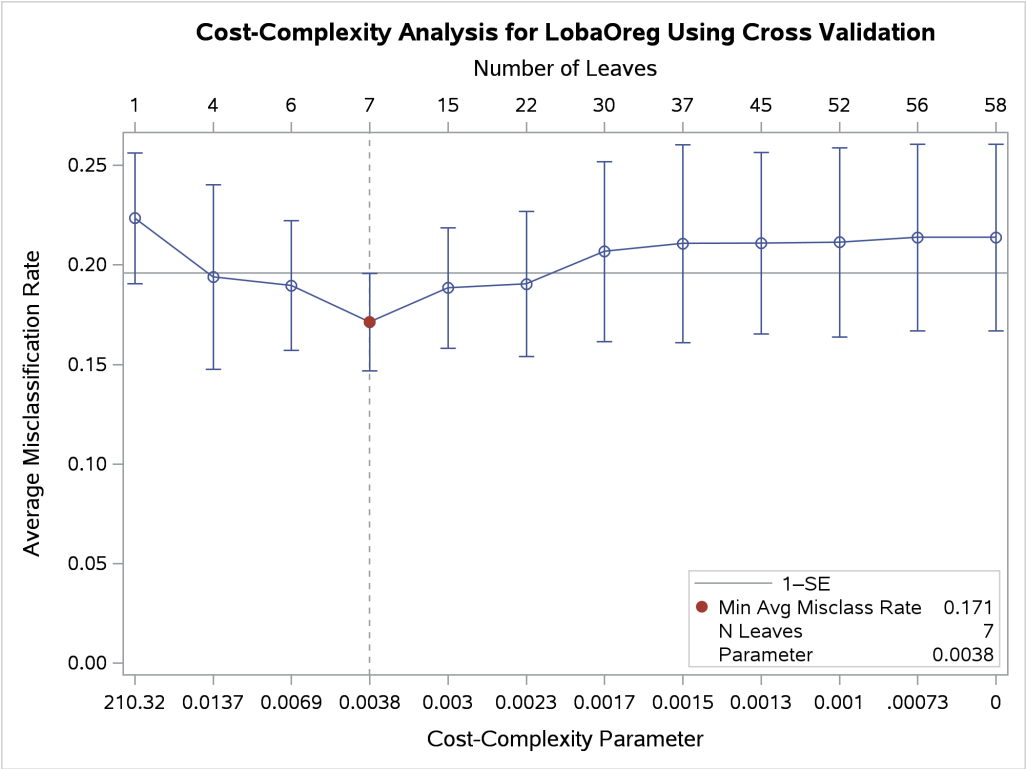

The misclassification rate is estimated by 10-fold cross validation; for computational details, see the section "Cost-Complexity Pruning" on page 5214. The subtree size (number of leaves) that corresponds to each parameter is indicated on the upper horizontal axis. The parameter value 0 corresponds to the fully grown tree, which has 58 leaves.

By default, PROC HPSPLIT selects the parameter that minimizes the average misclassification rate, as indicated by the vertical reference line and the dot in Output 68.1.2. Here the minimum rate occurs at a parameter value of 0.0038, which corresponds to a subtree that has seven leaves. However, the rates for two smaller subtrees, one that has four leaves and one that has six leaves, are within one standard error of the minimum rate.

In general, the plot in Output 68.1.2 often shows parameter choices that correspond to smaller subtrees for which the misclassification rates are nearly the same as the minimum misclassification rate. A common

approach for choosing the parameter is the 1-SE rule of Breiman et al. (1984), which selects the smallest subtree for which the misclassification rate is less than the minimum rate plus one standard error. In this case, the 1-SE rule selects a subtree that has only four leaves, but this is such a small subtree that the subtree with six leaves and a parameter value of 0.0069 seems preferable.

NOTE: The misclassification rates and their standard errors are estimates that depend on the random allocation of the observations to the 10 folds. You obtain different estimates if you specify a different SEED= value. Likewise, the estimates differ if you request a different number of folds by using the CVMETHOD= option.

The following statements rerun the analysis and request a tree with six leaves:

```
proc hpsplit data=sampsio.LAQ cvmodelfit seed=123;
   class LobaOreg ReserveStatus;
   model LobaOreg (event='1') =
      Aconif DegreeDays TransAspect Slope Elevation PctBroadLeafCov
      PctConifCov PctVegCov TreeBiomass EvapoTransAve EvapoTransDiff
      MoistIndexAve MoistIndexDiff PrecipAve PrecipDiff RelHumidAve
      RelHumidDiff PotGlobRadAve PotGlobRadDiff AveTempAve AveTempDiff
      DayTempAve DayTempDiff MinMinTemp MaxMaxTemp AmbVapPressAve
      AmbVapPressDiff SatVapPressAve SatVapPressDiff ReserveStatus;
   grow entropy;
   prune costcomplexity(leaves=6);
   code file='trescore.sas';
run;
```

The tree diagram in Output 68.1.3 provides an overview of the entire tree.

Output 68.1.3 Overview of Fitted Tree

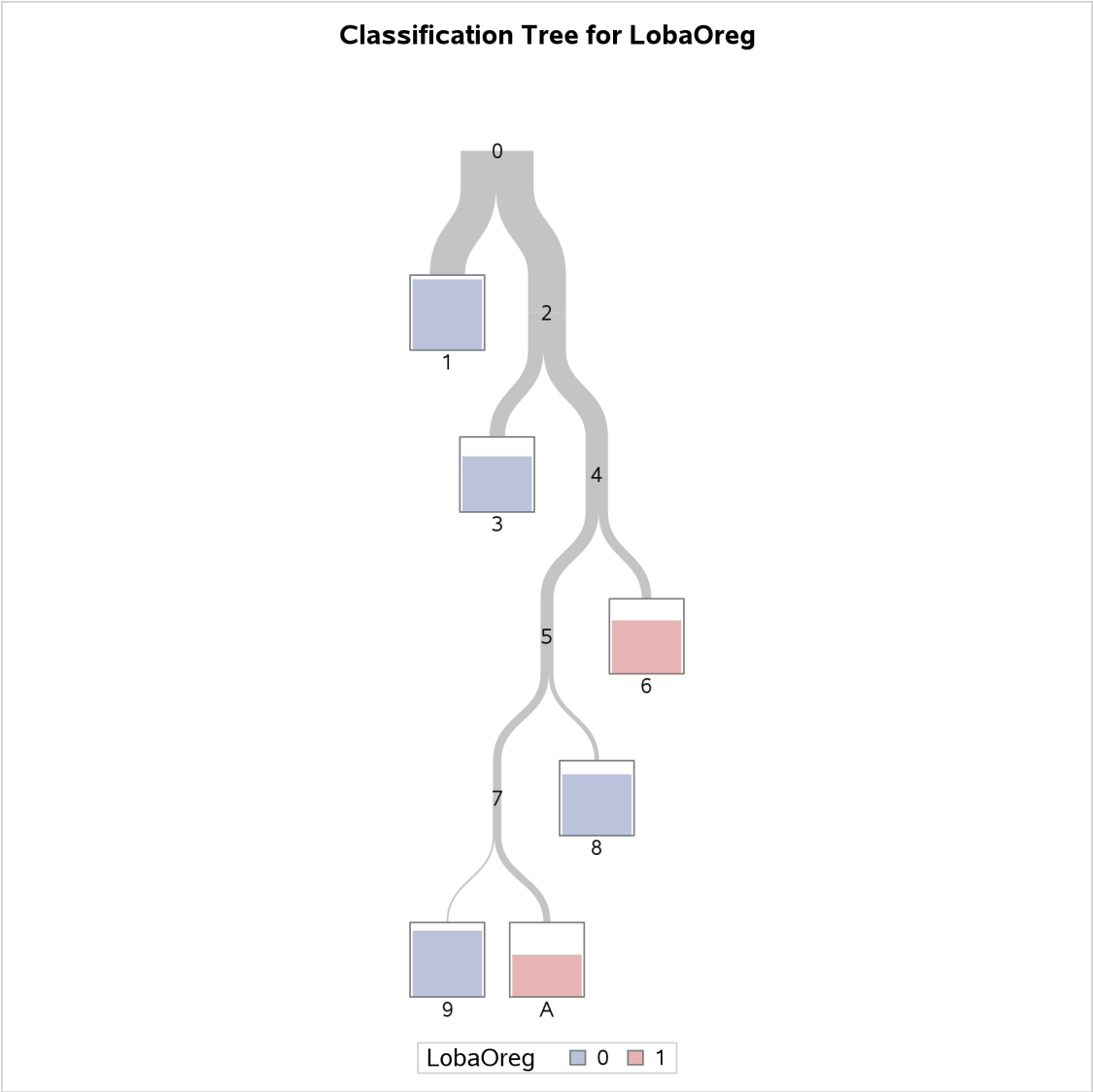

The color of the bar in each leaf node indicates the most frequent level of LobaOreg and represents the classification level assigned to all observations in that node. The height of the bar indicates the proportion of observations (sites) in the node that have the most frequent level.

The diagram in Output 68.1.4 provides more detail about the nodes and splits in the first four levels of the tree. It reveals a model that is highly interpretable.

Output 68.1.4 First Four Levels of Fitted Tree

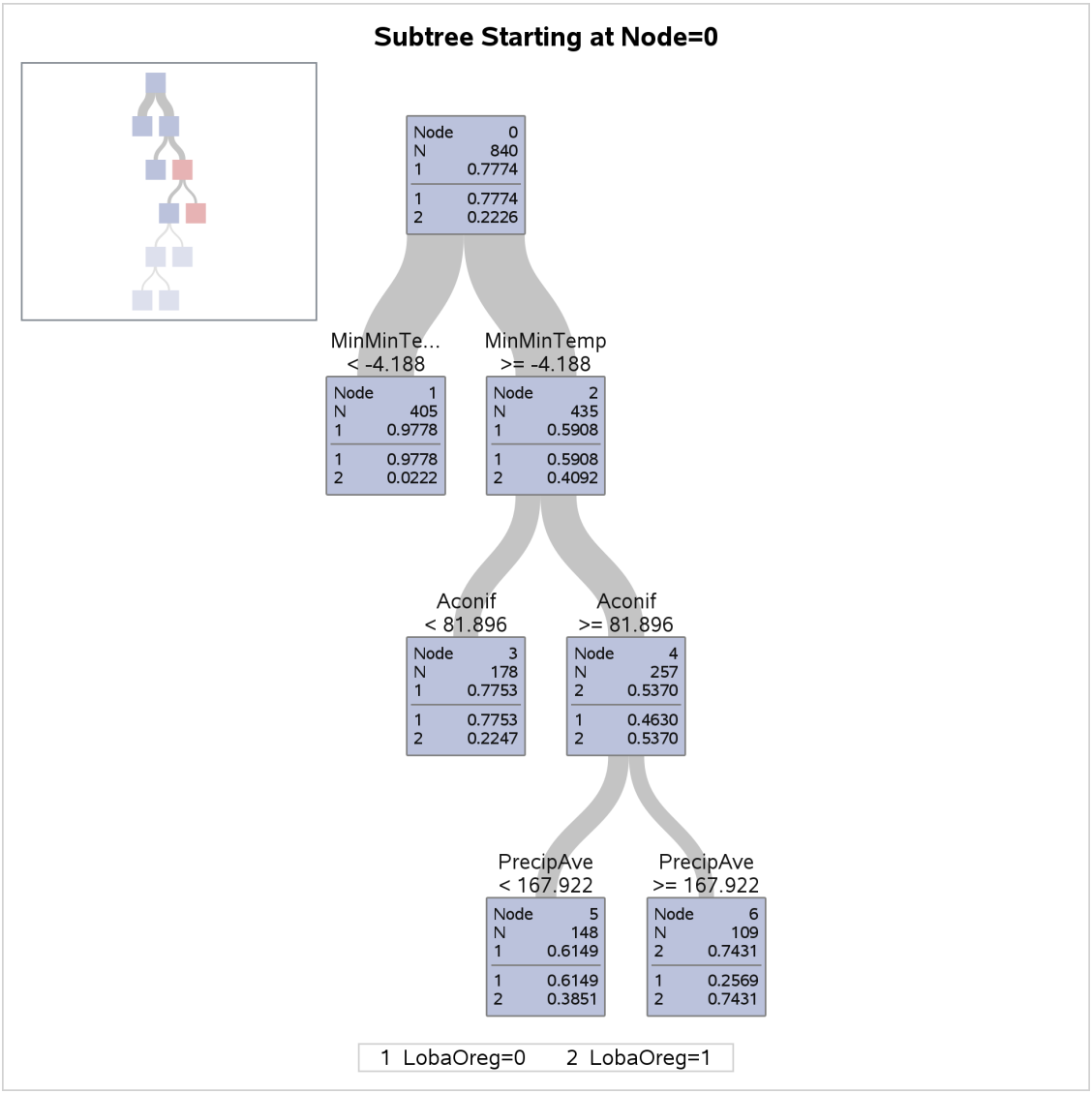

The first split is based on the variable MinMinTemp, which is the minimum of the 12 monthly minimum temperatures at each site. There are 405 sites whose values of MinMinTemp are less than –4.188 (node 1), and the species *Lobaria oregana* is present at only around 2% of them. Apparently this species does not grow well when the temperature can get very low.

The 435 sites for which MinMinTemp $\geq$ –4.188 (node 2) are further subdivided based on the variable Aconif, which is the average age of the dominant conifer at the site. *Lobaria oregana* is present at 53.7% of the 257 sites for which MinMinTemp $\geq$ –4.188 and Aconif $\geq$ 81.896 years. The cutoff of 81.896 years is notable because coniferous forests in the Pacific Northwest begin to exhibit old-forest characteristics at approximately 80 years of age (Old-Growth Definition Task Group 1986), and *Lobaria oregana* is a lichen species that is associated with old forests.

The 257 sites for which Aconif $\geq$ 81.896 are further subdivided on the basis of PrecipAve (average monthly precipitation) with a cutoff value 167.922 mm. *Lobaria oregana* was present at 74.31% of the 109 sites

for which MinMinTemp $\geq$ –4.188, Aconif $\geq$ 81.896 years, and PrecipAve $\geq$ 167.922 mm. Contrast this occupancy percentage with the 2.22% for the sites for which MinMinTemp $<$ –4.188.

In summary, based on the first three splits, *Lobaria oregana* is most likely to be found at sites for which MinMinTemp $\geq$ –4.188, Aconif $\geq$ 81.896, and PrecipAve $\geq$ 167.922.

The following statements use the PLOTS=ZOOMEDTREE option to request a detailed diagram of the subtree that begins at node 4:

```
proc hpsplit data=sampsio.LAQ cvmodelfit seed=123
   plots=zoomedtree(nodes=('4') depth=4);
   class LobaOreg ReserveStatus;
   model LobaOreg (event='1') =
      Aconif DegreeDays TransAspect Slope Elevation PctBroadLeafCov
      PctConifCov PctVegCov TreeBiomass EvapoTransAve EvapoTransDiff
      MoistIndexAve MoistIndexDiff PrecipAve PrecipDiff RelHumidAve
      RelHumidDiff PotGlobRadAve PotGlobRadDiff AveTempAve AveTempDiff
      DayTempAve DayTempDiff MinMinTemp MaxMaxTemp AmbVapPressAve
      AmbVapPressDiff SatVapPressAve SatVapPressDiff ReserveStatus;
   grow entropy;
   prune costcomplexity(leaves=6);
   code file='trescore.sas';
run;
```

This subtree, shown in Output 68.1.5, provides details for the portion of the entire tree that is not shown in Output 68.1.4.

Output 68.1.5 Bottom Four Levels of Fitted Tree

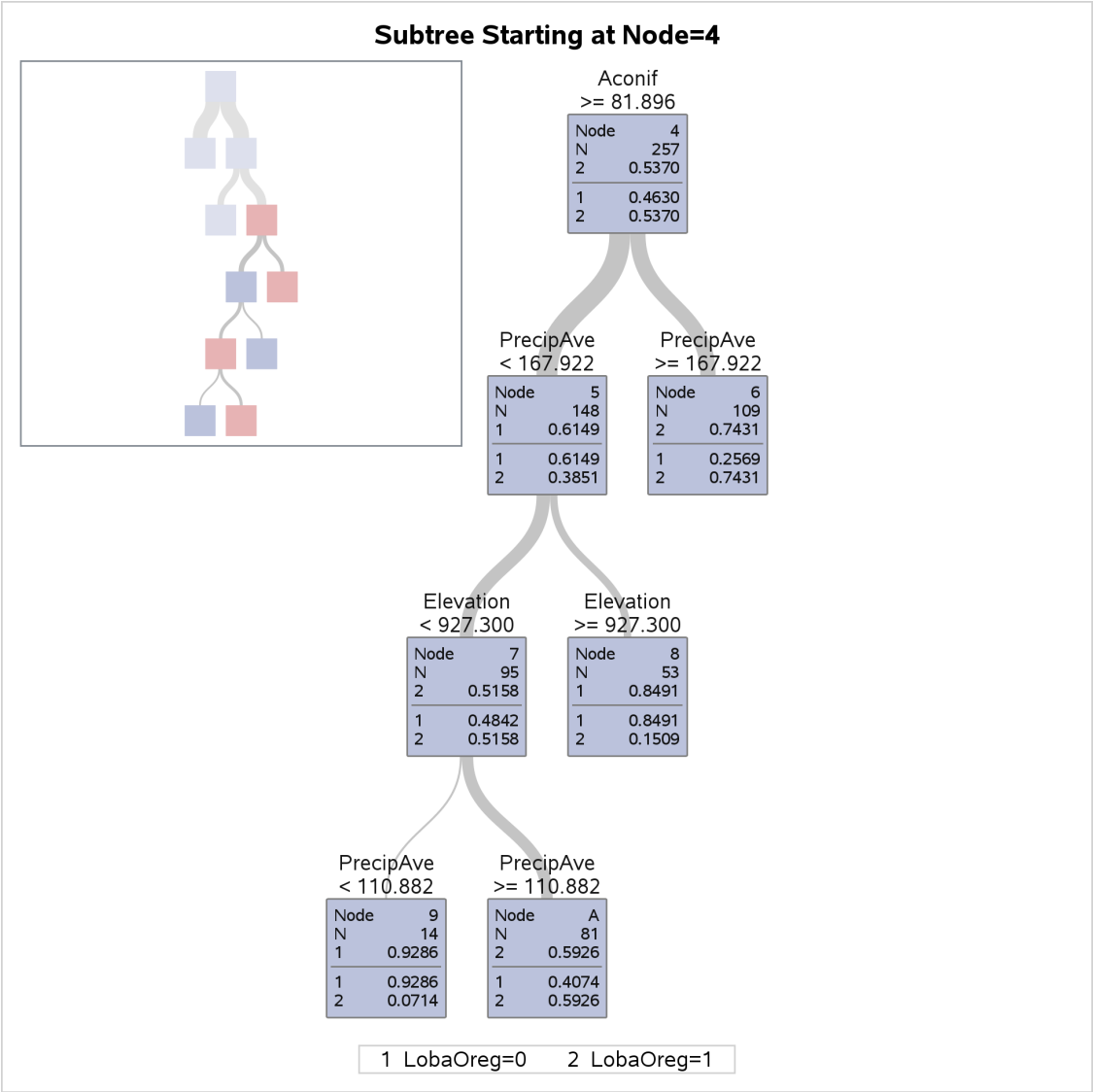

NOTE: You can use the NODES= option to request a table that describes the path from each leaf in the fitted tree back to the root node.

The next three displays evaluate the accuracy of the selected classification tree. Output 68.1.6 provides two confusion matrices.

Output 68.1.6 Confusion Matrices for Classification of LAQ

The HPSPLIT Procedure

Confusion Matrices				
		Predicted		
	Actual	0	1	Error Rate
Model Based	**0**	592	61	0.0934
	1	58	129	0.3102
Cross Validation	**0**	571	82	0.1256
	1	81	106	0.4332

The model-based confusion matrix is sometimes referred to as a resubstitution confusion matrix, because it results from applying the fitted model to the training data. The cross validation confusion matrix is produced when you specify the CVMODELFIT option. It is based on a 10-fold cross validation that is done independently of the 10-fold cross validation that is used to estimate ASEs for pruning parameters.

The table in Output 68.1.7 provides fit statistics for the selected classification tree.

Output 68.1.7 Fit Statistics for Classification of LAQ

Fit Statistics for Selected Tree									
	N Leaves	ASE	Mis-class	Sensitivity	Specificity	Entropy	Gini	RSS	AUC
Model Based	6	0.1046	0.1417	0.6898	0.9066	0.4825	0.2093	175.8	0.8805
Cross Validation	6	0.1236	0.1914	0.5668	0.8744				

Two sets of fit statistics are provided. The first is based on the fitted model, and the second (requested by the CVMODELFIT option) is based on 10-fold cross validation.

The model-based misclassification rate is low (14.2%), but the corresponding sensitivity, which measures the prediction accuracy at sites where the species is present, is only 69%. Good overall prediction accuracy but poor prediction of a particular level can occur when the data are not well balanced (in this case there are three times as many sites where the species is absent as there are sites where the species is present).

The cross validation misclassification rate is higher (18.1%) than the model-based rate, and the cross validation sensitivity is 64%, which is lower than the model-based sensitivity. Model-based error rates tend to be optimistic.

Output 68.1.8 displays an ROC curve, which is produced only for binary outcomes.

Output 68.1.8 ROC Curve for Classification of LAQ

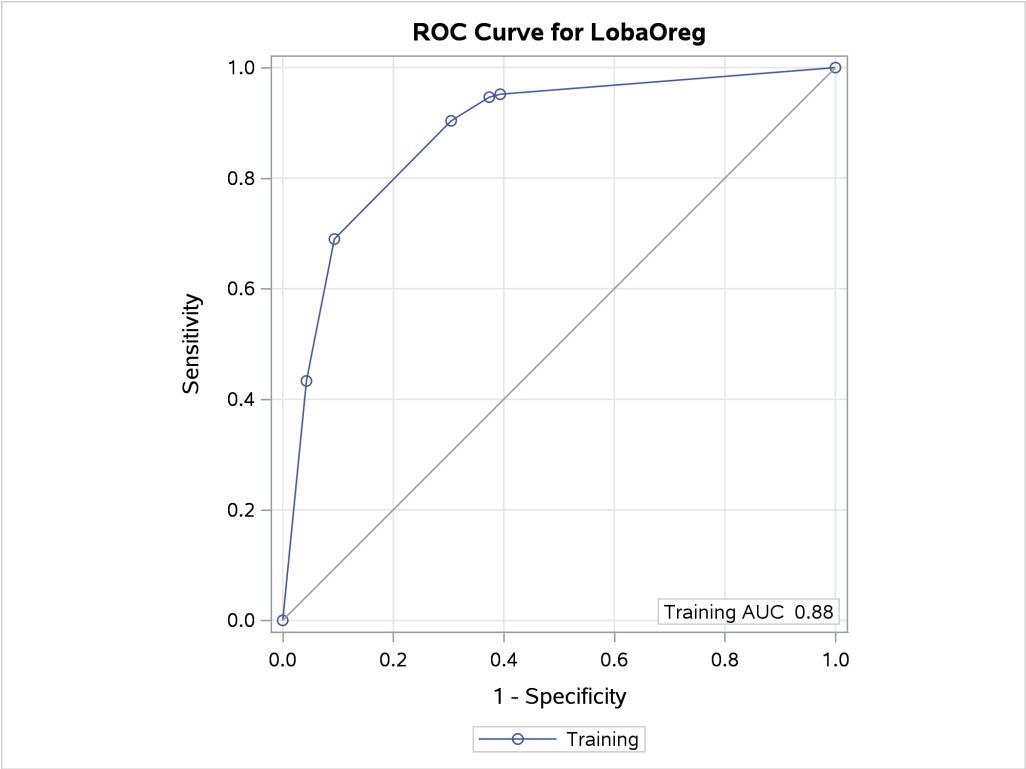

The AUC statistic and the values of the ROC curve are computed from the training data. When you specify a validation data set by using the PARTITION statement, the plot displays an additional ROC curve and AUC statistic, whose values are computed from the validation data.

NOTE: In this example, the computations of the sensitivity, specificity, AUC, and values of the ROC curve depend on defining LobaOreg=1 as the event of interest by using the EVENT= option in the MODEL statement.

Another way to assess the predictive accuracy of the fitted tree is to apply it to an independent test data set. In 2003, data on the presence of *Lobaria oregana* were collected at 300 sites as part of the Survey and Manage Program within the Northwest Forest Plan, the conservation plan for the northern spotted owl (Molina et al. 2003). These sites were in approximately the same area where the earlier Lichen Air Quality Surveys were conducted, with similar (but not identical) sampling protocols. The 2003 surveys are sometimes called the Pilot Random Grid surveys (Edwards et al. 2005). *Lobaria oregana* was detected at 26.7% (80) of the sites, an occupancy rate comparable to that of the Lichen Air Quality Surveys (187 / 840 = 22.3%).

The CODE statement in the preceding PROC HPSPLIT step requests a file named *trescore.sas* that contains SAS DATA step code for the rules that define the fitted tree model. You can use this code to classify the observations in the PRG data set as follows:

```
data lichenpred(keep=Actual Predicted);
   set sampsio.PRG end=eof;
   %include "trescore.sas";
   Actual    = LobaOreg;
   Predicted = (P_LobaOreg1 >= 0.5);
run;
```

The variables P_LobaOreg1 and P_LobaOreg0 contain the predicted probabilities that *Lobaria oregana* is present or absent, respectively. The value of the expression **(P_LobaOreg1 >= 0.5)** is 1 when P_LobaOreg1 $\geq$ P_LobaOreg0.

The following statements use these probabilities to produce a confusion matrix:

```
title "Confusion Matrix Based on Cutoff Value of 0.5";
proc freq data=lichenpred;
   tables Actual*Predicted / norow nocol nopct;
run;
```

The matrix is shown in Output 68.1.9.

Output 68.1.9 Confusion Matrix Based on Cutoff Value of 0.5

Confusion Matrix Based on Cutoff Value of 0.5

The FREQ Procedure

Frequency	Table of Actual by Predicted		
		Predicted	
Actual	0	1	Total
0	191	29	220
1	40	40	80
Total	231	69	300

The misclassification rate is (40 + 29) / 300 = 0.23, which is higher than the rates in Output 68.1.7. The sensitivity is 50%, which is less than the cross validation sensitivity in Output 68.1.7. The specificity is 86.8%.

When the sensitivity is much smaller than the specificity (which is common for highly unbalanced data), you can change the probability cutoff to a smaller value, increasing the sensitivity at the expense of the specificity and the overall accuracy. The following statements produce the confusion matrix for a cutoff value of 0.1:

```
data lichenpred(keep=Actual Predicted);
   set sampsio.PRG end=eof;
   %include "trescore.sas";
   Actual    = LobaOreg;
   Predicted = (P_LobaOreg1 >= 0.1);
run;

title "Confusion Matrix Based on Cutoff Value of 0.1";
proc freq data=lichenpred;
   tables Actual*Predicted / norow nocol nopct;
run;
```

The matrix is shown in Output 68.1.10.

Output 68.1.10 Confusion Matrix Based on Cutoff Value of 0.1

Confusion Matrix Based on Cutoff Value of 0.1

The FREQ Procedure

Frequency	Table of Actual by Predicted		
		Predicted	
Actual	0	1	Total
0	160	60	220
1	35	45	80
Total	195	105	300

Based on the cutoff value of 0.1, the sensitivity is 56.3% and the specificity is 72.7%.

Example 68.2: Cost-Complexity Pruning with Cross Validation

(View the complete code for this example (hpsplex2.sas) in the example repository.)

In this example, data were collected to study the damage to pine forests from mountain pine beetle attacks in the Sawtooth National Recreation Area (SNRA) in Idaho (Cutler et al. 2003). (The data in this example were provided by Richard Cutler, Department of Mathematics and Statistics, Utah State University.) A classification tree is applied to classify various types of vegetation in the area based on data from satellite images. This classification can then be used to track how the pine beetle infestation is progressing through the forest. Data from 699 points in the SNRA are included in the sample.

This example creates a classification tree to predict the response variable Type, which contains the 10 vegetation classes represented in the data: `'Agriculture'`, `'Dirt'`, `'DougFir'`, `'Grass'`, `'GreenLP'`, `'RedTop'`, `'Road'`, `'Sagebrush'`, `'Shadow'`, and `'Water'`. The predictor variables include the following:

- the spectral intensities on four bands of the satellite imagery: Blue, Green, Red, and NearInfrared
- Elevation
- NDVI, a function of Red and NearInfrared
- "Tasseled cap transformations" of the intensities on the four bands of imagery: SoilBrightness, Greenness, Yellowness, and NoneSuch

The first step in the analysis is to run PROC HPSPLIT to identify the best subtree model:

```
ods graphics on;

proc hpsplit data=sampsio.snra cvmethod=random(10) seed=123 intervalbins=500;
   class Type;
   grow gini;
   model Type = Blue Green Red NearInfrared NDVI Elevation
                SoilBrightness Greenness Yellowness NoneSuch;
   prune costcomplexity;
```

```
run;
```

You grow the tree by using the Gini index criterion, specified in the GROW statement, to create splits. This is a relatively small data set, so in order to use all the data to train the model, you apply cross validation with 10 folds, as specified in the CVMETHOD= option, to the cost-complexity pruning for subtree selection. An alternative would be to partition the data into training and validation sets. The SEED= option ensures that results remain the same in each run of the procedure. Different seeds can produce different trees because the cross validation fold assignments vary. When you do not specify the SEED= option, the seed is assigned based on the time.

By default, PROC HPSPLIT creates a plot of the estimated misclassification rate at each complexity parameter value in the sequence, as displayed in Output 68.2.1.

Output 68.2.1 Misclassification Rate as a Function of Cost-Complexity Parameter

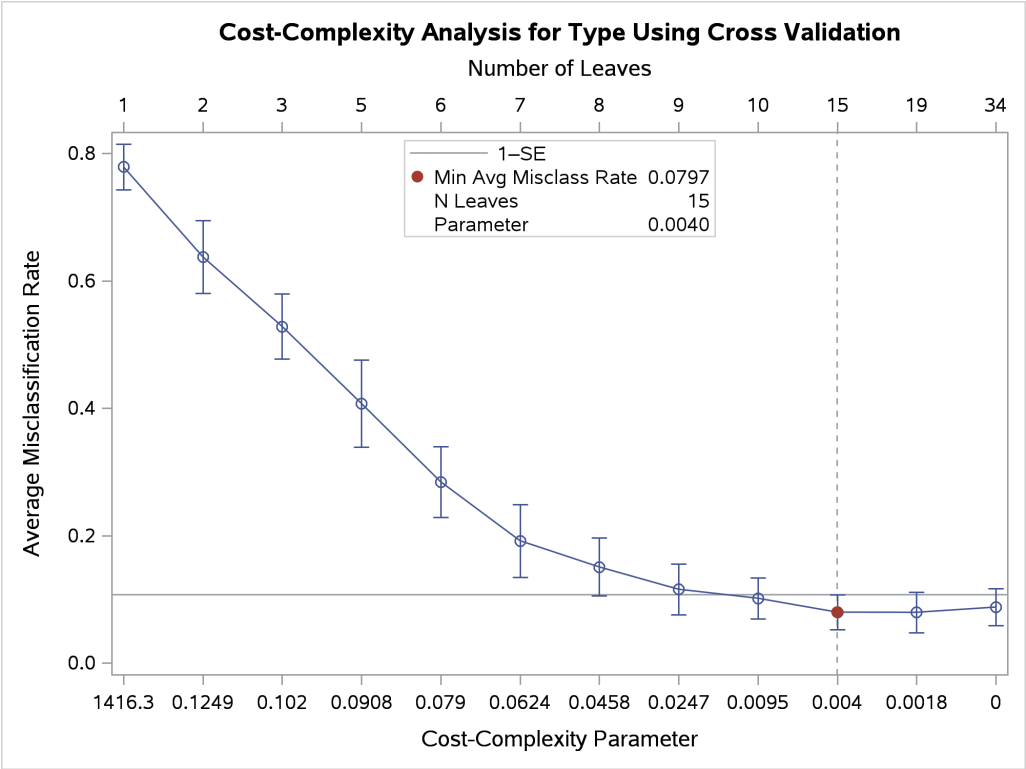

The ends of the error bars correspond to the misclassification rate plus or minus one standard error (SE) at each of the complexity pruning parameter values. A vertical reference line is drawn at the complexity parameter that has the lowest misclassification rate, and the subtree of the corresponding size for that complexity parameter is selected as the final tree. In this case, the 15-leaf tree is selected as the final tree. The horizontal reference line represents the minimum misclassification rate plus one standard error.

Often, you would apply the 1-SE rule (Breiman et al. 1984) when you are pruning via the cost-complexity method to potentially select a smaller tree that has only a slightly higher error rate than the minimum error. Selecting the smallest tree that has a misclassification rate below the horizontal reference line is in effect implementing the 1-SE rule. The subtree that has 10 leaves would be selected according to this rule, so you can run PROC HPSPLIT again as follows to override the subtree that was automatically selected in the first run:

```
proc hpsplit data=sampsio.snra plots=zoomedtree(node=7) seed=123 cvmodelfit
   intervalbins=500;
   class Type;
   grow gini;
   model Type = Blue Green Red NearInfrared NDVI Elevation
                SoilBrightness Greenness Yellowness NoneSuch;
   prune costcomplexity (leaves=10);
run;
```

This code includes specification of the LEAVES=10 option in the PRUNE statement to select this smaller subtree that performs almost equivalently to the subtree with 15 leaves from the earlier run. Specifying ZOOMEDTREE(NODE=7) in the PLOTS= option requests that the ODS graph ZoomedTreePlot displays the tree rooted at node 7 instead of at the root node. The CVMODELFIT option requests fit statistics for the final model by using cross validation as well as the cross validation confusion matrix.

Output 68.2.2 provides an overview of the final tree that has 10 leaves as requested.

Output 68.2.2 Diagram of 10-Leaf Tree Selected Using 1-SE Rule

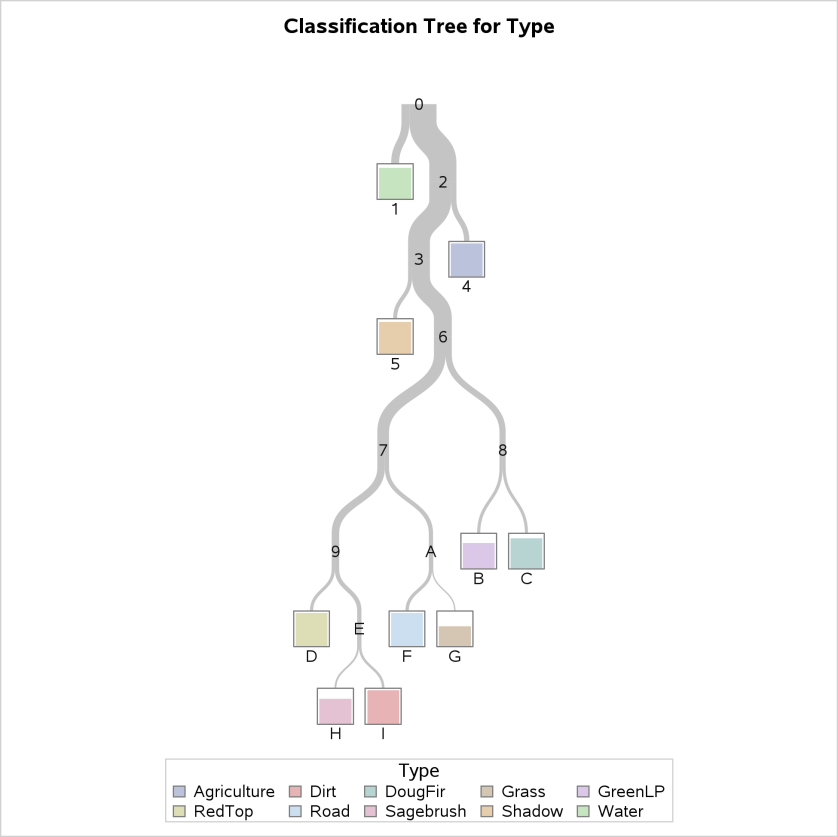

It turns out that there is exactly one leaf in the classification tree that corresponds to each of the 10 vegetation classes; this does not usually occur. The leaf color indicates the most frequently observed response among observations in that leaf, which is then the predicted response for all observations in that leaf. The height of the bars in the nodes represents the proportion of observations that have that particular response. For example, in node D, all observations have the value of `'RedTop'` for the response variable Type, whereas in node G, it appears that slightly over half of the observations have the value `'Grass'`.

Output 68.2.3 shows more details about a portion of the final tree, including splitting variables and values.

Output 68.2.3 Detailed Diagram of 10-Leaf Tree

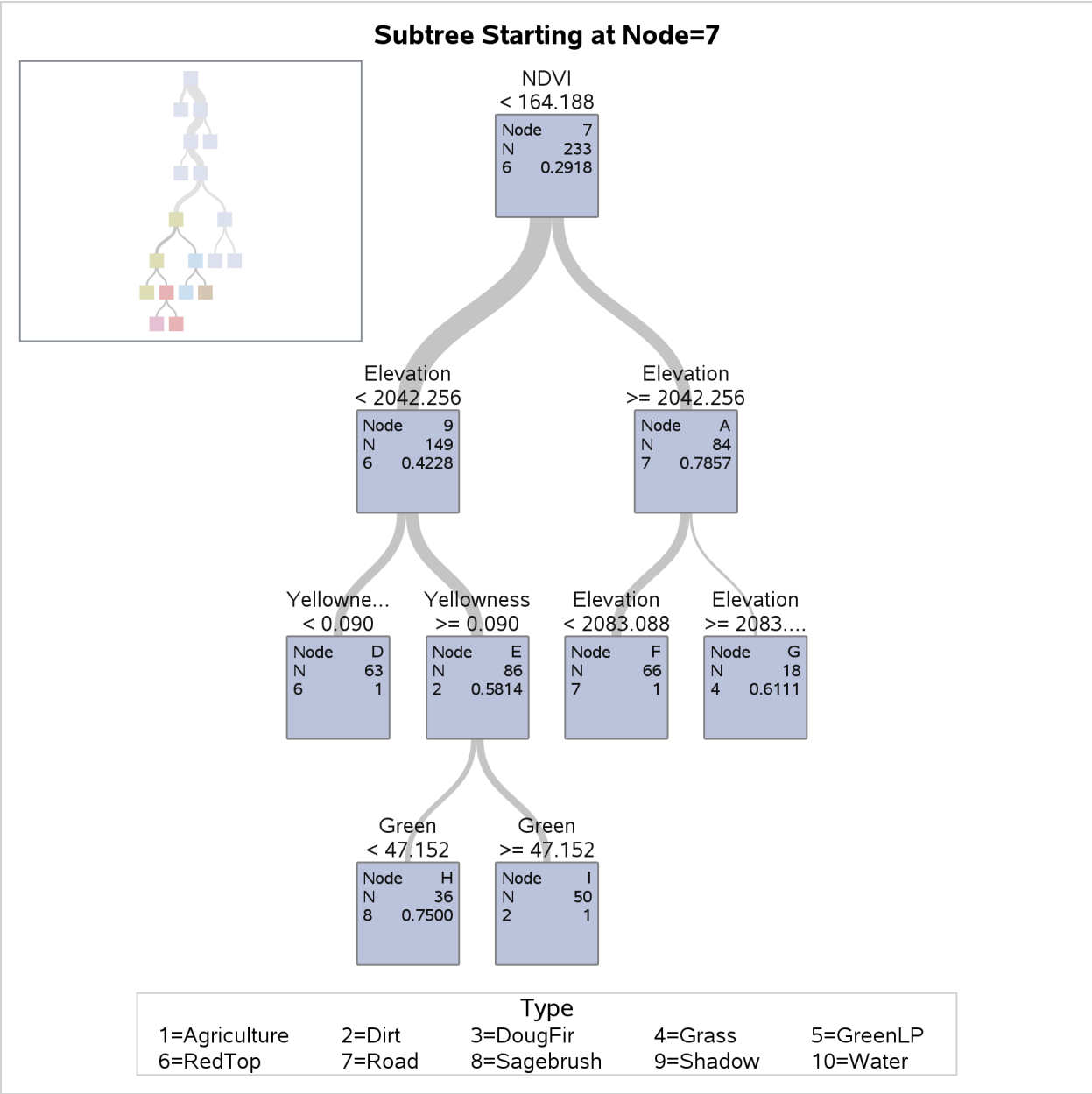

As requested, the detailed tree diagram is displayed for the portion of the tree rooted at node 7 so that you can view the splits made at the bottom of the tree. You can see that several splits are made on the variable Elevation. The vegetation type most common in node G, whose observations have an elevation of at least 2083.0880, is `'Grass'`.

Confusion matrices are displayed in Output 68.2.4.

Output 68.2.4 Confusion Matrices for SNRA Data

The HPSPLIT Procedure

Confusion Matrices												
		Predicted										
	Actual	Agriculture	Dirt	DougFir	Grass	GreenLP	RedTop	Road	Sagebrush	Shadow	Water	Error Rate
Model Based	Agriculture	105	0	0	0	1	0	0	0	0	0	0.0094
	Dirt	0	50	0	0	1	0	0	0	0	2	0.0566
	DougFir	0	0	55	0	11	0	0	0	0	0	0.1667
	Grass	1	0	0	11	1	0	0	2	0	0	0.2667
	GreenLP	0	0	5	0	49	0	0	1	0	0	0.1091
	RedTop	0	0	0	5	0	63	0	0	0	0	0.0735
	Road	0	0	0	0	0	0	66	0	0	2	0.0294
	Sagebrush	0	0	0	2	0	0	0	27	0	0	0.0690
	Shadow	0	0	0	0	0	0	0	2	76	6	0.0952
	Water	0	0	0	0	0	0	0	4	3	148	0.0452
Cross Validation	Agriculture	104	0	0	0	2	0	0	0	0	0	0.0189
	Dirt	0	49	0	0	1	0	0	1	0	2	0.0755
	DougFir	1	0	55	0	10	0	0	0	0	0	0.1667
	Grass	2	0	0	11	0	0	0	2	0	0	0.2667
	GreenLP	0	0	16	0	37	0	0	1	1	0	0.3273
	RedTop	0	1	0	4	1	62	0	0	0	0	0.0882
	Road	0	0	0	0	0	2	64	0	0	2	0.0588
	Sagebrush	0	1	0	2	0	0	0	26	0	0	0.1034
	Shadow	0	0	0	0	0	2	0	2	74	6	0.1190
	Water	0	0	0	0	0	3	0	1	3	148	0.0452

This table contains two matrices—one for the training data that uses the final tree and one that uses the cross validation folds—requested by the CVMODELFIT option in the PROC HPSPLIT statement. The values on the diagonal of each confusion matrix are the number of observations that are correctly classified for each of the 10 vegetation types. For the model-based matrix, you can see that the only nonzero value in the `'RedTop'` column is in the `'RedTop'` row. This is consistent with what is shown in Output 68.2.2, where the bar in node D with a predicted response of `'RedTop'` is the full height of the box representing the leaf, indicating that all observations on that leaf are correctly classified. You can also see from the matrix that the `'DougFir'` and `'GreenLP'` vegetation types are hard to distinguish; 11 of the 66 observations that have an actual response of `'DougFir'` are incorrectly assigned the response of `'GreenLP'`, corresponding to the 0.1667 error rate reported for `'DougFir'`.

Fit statistics are shown in Output 68.2.5.

Output 68.2.5 Fit Statistics for SNRA Data

Fit Statistics for Selected Tree						
	N Leaves	ASE	Mis-class	Entropy	Gini	RSS
Model Based	10	0.0120	0.0701	0.3597	0.1197	83.6514
Cross Validation	10	0.0166	0.0974			

You can see from this table that the subtree with 10 leaves fits the training data very accurately, with 93%

of the observations classified correctly. Because no validation data are present in this analysis, you get a better indication of how well the model fits and will generalize to new data by looking at the cross validation statistics, also requested by the CVMODELFIT option, that are included in the table. The misclassification rate that is averaged across the 10 folds is higher than the training misclassification rate for the final tree, suggesting that this model is slightly overfitting the training data.

Example 68.3: Creating a Regression Tree

(View the complete code for this example (hpsplex3.sas) in the example repository.)

This example performs an analysis similar to the one in the "Getting Started" section of Chapter 67, "The HPREG Procedure," where a linear regression model is fit. You can alternatively fit a regression tree to predict the salaries of Major League Baseball players based on their performance measures from the previous season by using almost identical code. Regression trees are piecewise constant models that, for relatively small data sets such as this, provide succinct summaries of how the predictors determine the predictions. These models are usually easier to interpret than linear regression models. The Sashelp.Baseball data set contains salary and performance information for Major League Baseball players (excluding pitchers) who played at least one game in both the 1986 and 1987 seasons (Time Inc. 1987). The following statements create a regression tree model:

```
ods graphics on;

proc hpsplit data=sashelp.baseball seed=123;
   class league division;
   model logSalary = nAtBat nHits nHome nRuns nRBI nBB
                     yrMajor crAtBat crHits crHome crRuns crRbi
                     crBB league division nOuts nAssts nError;
   output out=hpsplout;
run;
```

By default, the tree is grown using the RSS criterion, and cost-complexity pruning with 10-fold cross validation is performed. The OUTPUT statement requests generation of the data set hpsplout, which contains the predicted salary from the tree model for each observation.

Much of the output for a regression tree is identical to the output for a classification tree. Tables and plots where there are differences are displayed and discussed on the following pages.

Output 68.3.1 displays the full regression tree.

Output 68.3.1 Overview Diagram of Regression Tree

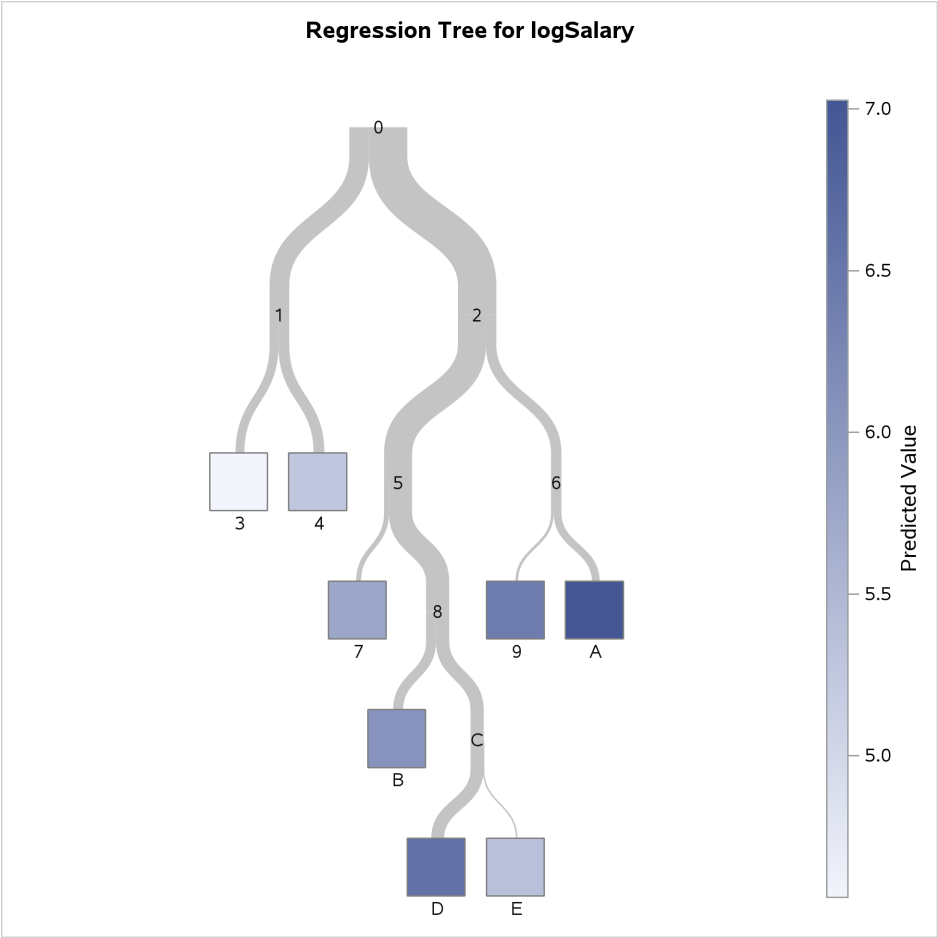

You can see from this diagram that the final selected tree has eight leaves. For a regression tree, the shade of the leaves represents the predicted response value, which is the average observed logSalary for the observations in that leaf. Node 3 has the lowest predicted response value, indicated by the lightest shade of blue, and Node A has the highest, indicated by the dark shade.

Output 68.3.2 shows details of the tree.

Output 68.3.2 Detailed Diagram of Regression Tree

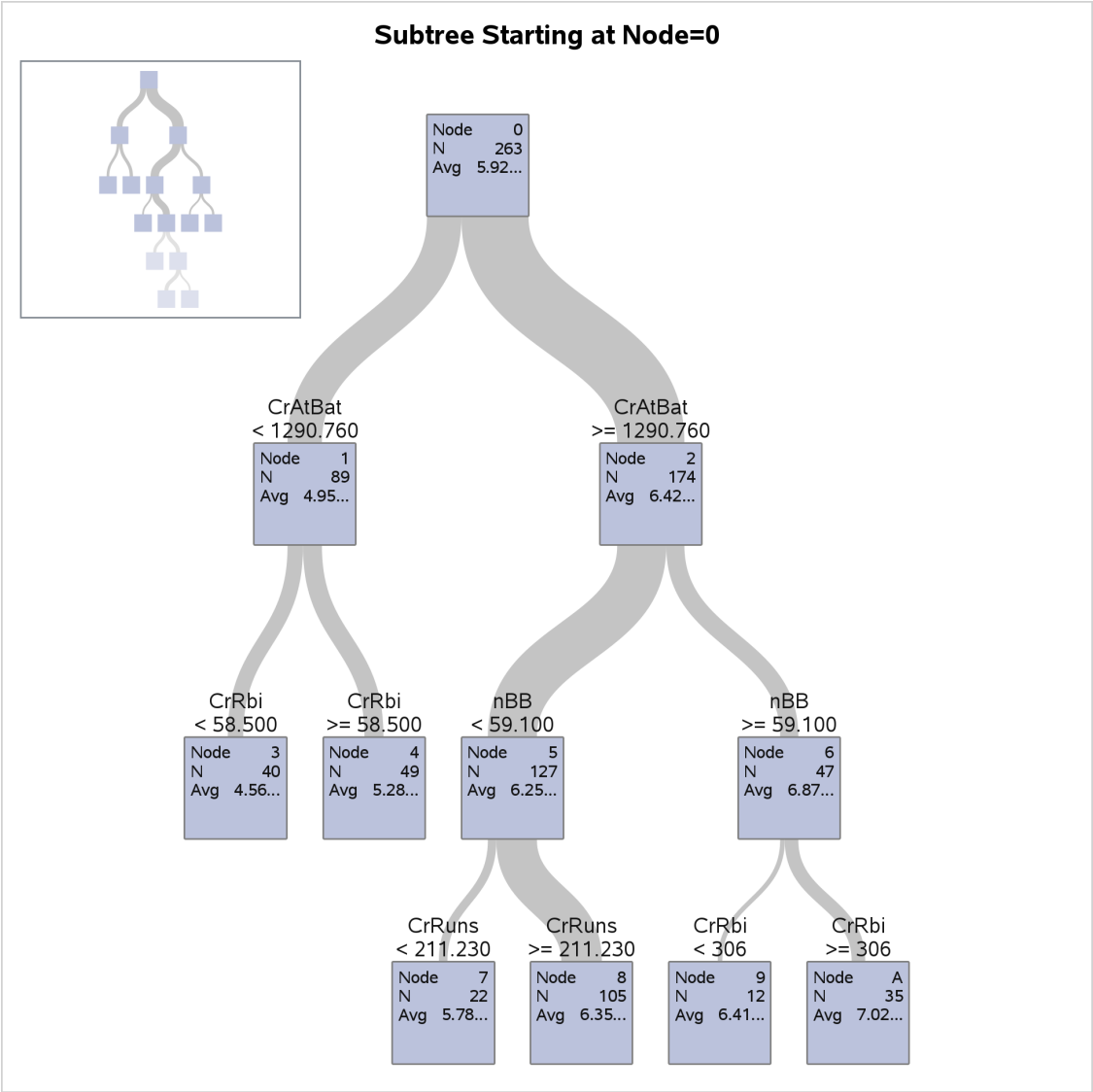

As with a classification tree, you can see split variables and values for a portion of the tree in this view. You can also see the exact predicted response value, which is the average observed response, in each node.

The confusion matrix is omitted from the output when you are modeling a regression tree because it is relevant only for a categorical response.

Output 68.3.3 displays fit statistics for the final regression tree.

Output 68.3.3 Regression Tree Performance

The HPSPLIT Procedure

Model-Based Fit Statistics for Selected Tree		
N Leaves	ASE	RSS
6	0.1681	44.2142

Note that this table contains different statistics from those included for a classification tree. The ASE and RSS are reported here to help you assess the model fit. You could also use the CVMODELFIT option in the PROC HPSPLIT statement to obtain the cross validated fit statistics, as with a classification tree.

Output 68.3.4 shows the hpsplout data set that is created by using the OUTPUT statement and contains the first 10 observations of the predicted log-transformed salaries for each player in Sashelp.Baseball based on the regression tree model.

Output 68.3.4 Scored Predictor Data Set

Obs	logSalary	_Node_	_Leaf_	P_logSalary
1	.	3	0	4.56042
2	6.16331	9	4	6.08107
3	6.17379	6	2	6.87289
4	6.21461	10	5	6.56218
5	4.51634	3	0	4.56042
6	6.62007	10	5	6.56218
7	4.24850	3	0	4.56042
8	4.60517	3	0	4.56042
9	4.31749	3	0	4.56042
10	7.00307	6	2	6.87289

The variable P_logSalary contains the predicted salaries on the log scale. Note that all observations in the same leaf have the same predicted response. The OUT= data set can contain additional variables from the DATA= data set if you specify them in the ID statement.

Example 68.4: Creating a Binary Classification Tree with Validation Data

(View the complete code for this example (hpsplex4.sas) in the example repository.)

A common use of classification trees is to predict the likelihood that a mortgage applicant will default on a loan. The data set Hmeq, which is in the Sampsio library that SAS provides, contains observations for 5,960 mortgage applicants. A variable named Bad indicates whether the applicant paid or defaulted on the loan that was given.

This example uses Hmeq to build a tree model that is used to score the data and can be used to score data about new applicants. Table 68.6 describes the variables in Hmeq.

Table 68.6 Variables in the Home Equity (Hmeq) Data Set

Variable	Role	Level	Description
Bad	Response	Binary	1 = applicant defaulted on the loan or is seriously delinquent 0 = applicant paid the loan
CLAge	Predictor	Interval	Age of oldest credit line in months
CLNo	Predictor	Interval	Number of credit lines
DebtInc	Predictor	Interval	Debt-to-income ratio
Delinq	Predictor	Interval	Number of delinquent credit lines
Derog	Predictor	Interval	Number of major derogatory reports
Job	Predictor	Nominal	Occupational category
Loan	Predictor	Interval	Requested loan amount
MortDue	Predictor	Interval	Amount due on existing mortgage
nInq	Predictor	Interval	Number of recent credit inquiries
Reason	Predictor	Binary	`'DebtCon'` = debt consolidation `'HomeImp'` = home improvement
Value	Predictor	Interval	Value of current property
YoJ	Predictor	Interval	Years at present job

The response variable for the tree model is Bad, a CLASS variable that has two values (0 indicates payment of loan, and 1 indicates default). The other variables are predictor variables for the model. The following statements display the first 10 observations of the data set:

```
/* Convert variable names to mixed case */
data hmeq;
   length Bad Loan MortDue Value 8 Reason Job $7
          YoJ Derog Delinq CLAge nInq CLNo DebtInc 8;
   set sampsio.hmeq;
run;

proc print data=hmeq(obs=10); run;
```

Output 68.4.1 shows the partial listing of Hmeq.

Output 68.4.1 Partial Listing of the Hmeq Data

Obs	Bad	Loan	MortDue	Value	Reason	Job	YoJ	Derog	Delinq	CLAge	nInq	CLNo	DebtInc
1	1	1100	25860	39025	HomeImp	Other	10.5	0	0	94.367	1	9	.
2	1	1300	70053	68400	HomeImp	Other	7.0	0	2	121.833	0	14	.
3	1	1500	13500	16700	HomeImp	Other	4.0	0	0	149.467	1	10	.
4	1	1500	.	.			.	.	.	.	.	.	.
5	0	1700	97800	112000	HomeImp	Office	3.0	0	0	93.333	0	14	.
6	1	1700	30548	40320	HomeImp	Other	9.0	0	0	101.466	1	8	37.1136
7	1	1800	48649	57037	HomeImp	Other	5.0	3	2	77.100	1	17	.
8	1	1800	28502	43034	HomeImp	Other	11.0	0	0	88.766	0	8	36.8849
9	1	2000	32700	46740	HomeImp	Other	3.0	0	2	216.933	1	12	.
10	1	2000	.	62250	HomeImp	Sales	16.0	0	0	115.800	0	13	.

The following statements use the HPSPLIT procedure to create a classification tree:

```
ods graphics on;

proc hpsplit data=hmeq maxdepth=5;
   class Bad Delinq Derog Job nInq Reason;
   model Bad(event='1') = Delinq Derog Job nInq Reason CLAge CLNo
               DebtInc Loan MortDue Value YoJ;
   prune costcomplexity;
   partition fraction(validate=0.3 seed=123);
   code file='hpsplexc.sas';
   rules file='rules.txt';
run;
```

The MAXDEPTH= option specifies the maximum depth of the tree to be grown.

Specifying Bad to the left of the equal sign in the MODEL statement indicates that it is the response variable. Because Bad is a binary response, the EVENT= option specifies that the calculations for sensitivity, specificity, AUC, and ROC be based on the level Bad=1.

By default, the entropy metric is used to grow the tree. The PRUNE statement requests cost-complexity pruning.

The PARTITION statement specifies that the observations in Hmeq be logically partitioned into disjoint subsets for model training and validation. Observations are randomly selected for the validation subset with the probability 0.3; the remaining observations are selected for the training subset.

The FILE= option in the CODE statement requests that SAS DATA step score code be saved to a file named *hpsplexc.sas*, and the FILE= option in the RULES statement requests that the node rules be saved to a file named *rules.txt*.

The tree diagram in Output 68.4.2 provides an overview of the full tree.

Output 68.4.2 Overview Diagram of Final Tree

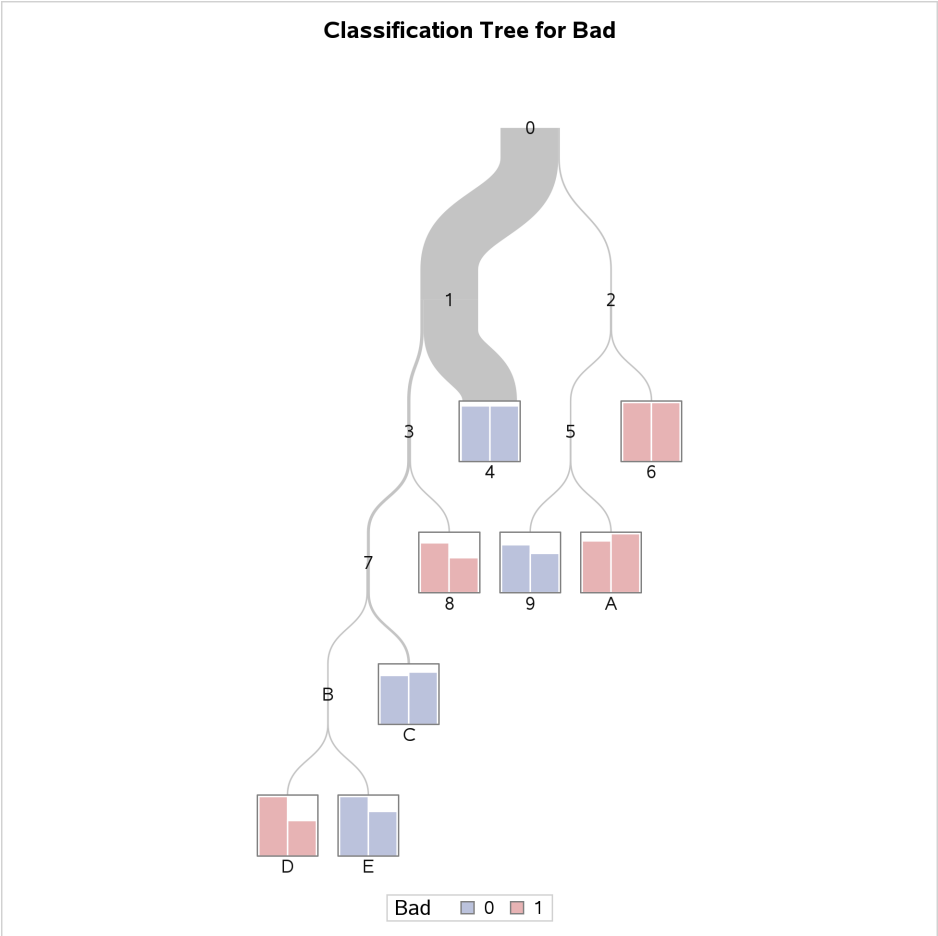

You can see from this diagram that the observations in terminal nodes 4, 9, C, and E are assigned a prediction of Bad=0 and those in terminal nodes 6, 8, A, and D are assigned a prediction of Bad=1. You can easily see that node 4 contains the most observations, as indicated by thickness of the link from its parent node.

The tree diagram in Output 68.4.3 is a detailed view of the top portion of the tree. You can use the PLOTS= option in the PROC HPSPLIT statement to control which nodes are displayed.

Output 68.4.3 Detailed Tree Diagram

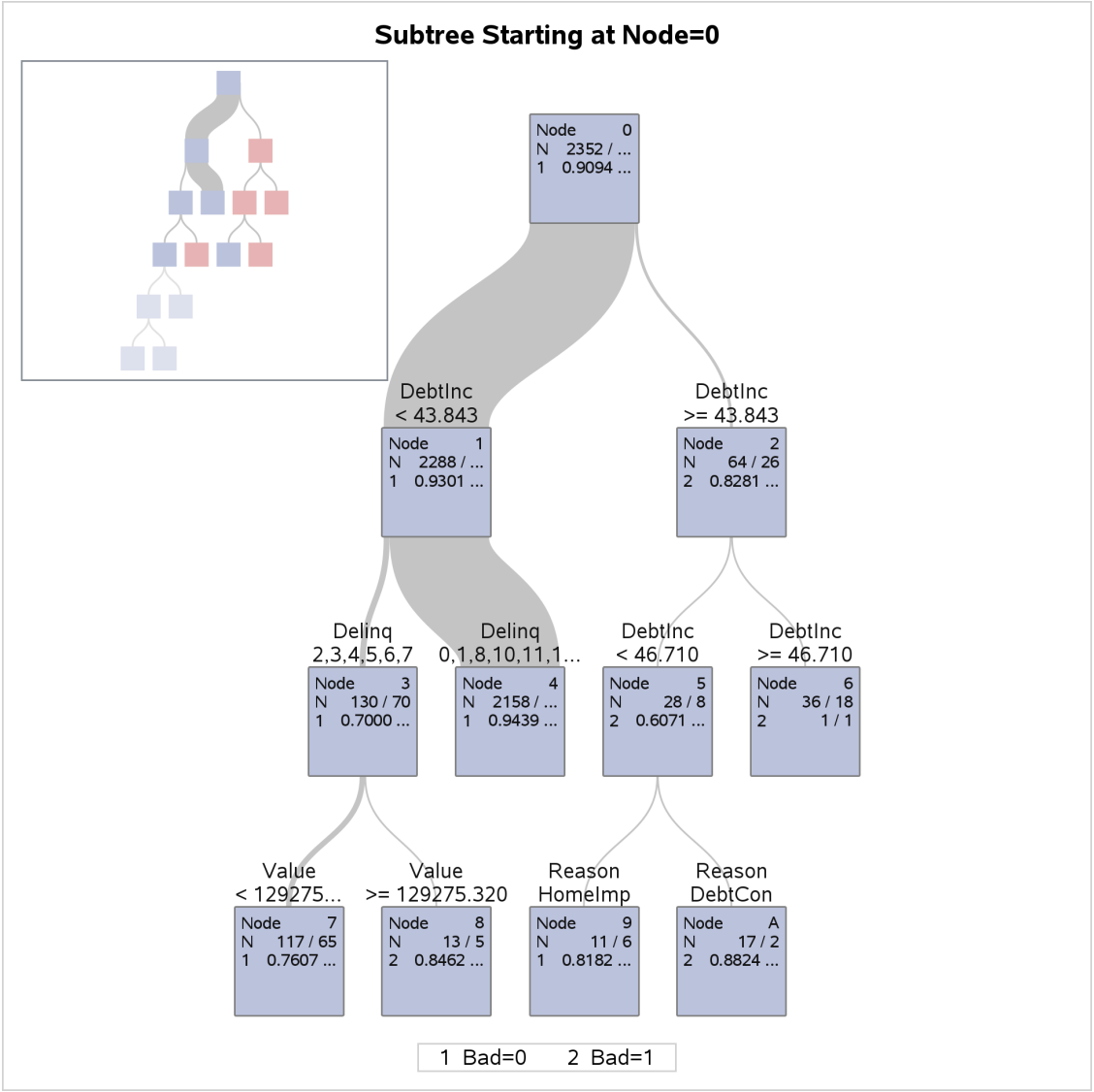

By default, this view provides detailed splitting information about the first three levels of the tree, including the splitting variable and splitting values. The splitting rule above each node determines which observations from the parent node are included in the node. The first row of the table inside the node provides the node identifier. The second row of the table provides the number of training and validation observations, separated by a slash. The third row shows the predicted response for observations in that node if classification was occurring at that point, along with the proportion of training and validation observations with that observed response. Note that the legend shows what actual value of the response variable is represented by the value shown in the node. For example, in node 6, all 36 observations in the training data and all 18 observations in the validation data have an observed response value of Bad=1, as indicated by the value 2 shown on the third line.

Because the response is categorical, the confusion matrices for the training data and for the validation data are displayed in the table shown in Output 68.4.4.

Output 68.4.4 Training and Validation Confusion Matrices

The HPSPLIT Procedure

Confusion Matrices				
		Predicted		
	Actual	0	1	Error Rate
Training	0	2135	4	0.0019
	1	140	73	0.6573
Validation	0	921	4	0.0043
	1	61	26	0.7011

For the training data, there are 73 observations where the model correctly predicted Bad=1 and 2,135 observations where the model correctly predicted Bad=0. The values in the off-diagonal entries of the matrices show how many times the model misclassified observations.

When you are modeling a binary response variable, a plot of the ROC curve is displayed as shown in Output 68.4.5.

Output 68.4.5 ROC Plot

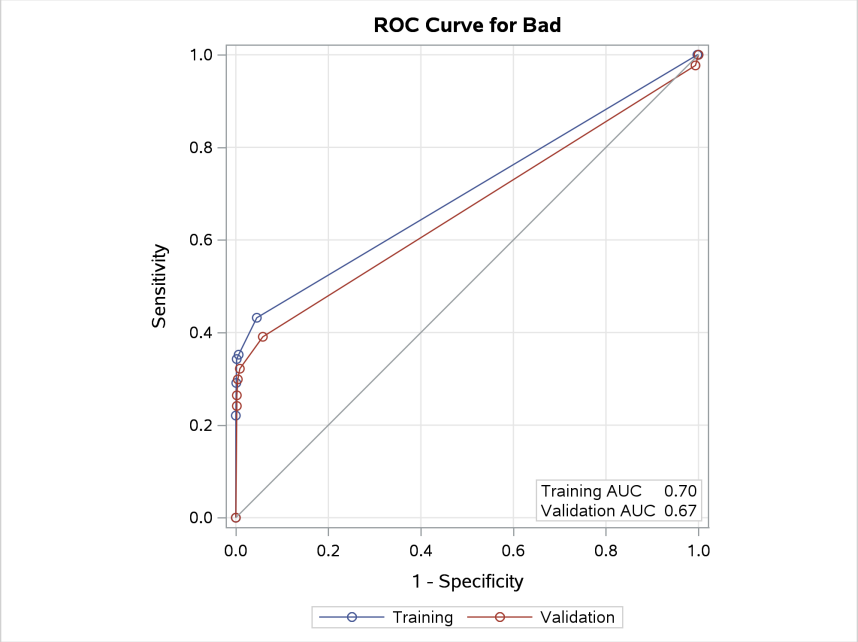

This plot summarizes the performance of the tree model in terms of sensitivity and specificity.

Output 68.4.6 displays the "Tree Performance" table, which provides several impurity measures and fit statistics for the final tree.

Output 68.4.6 Tree Performance

Fit Statistics for Selected Tree									
	N Leaves	ASE	Mis-class	Sensitivity	Specificity	Entropy	Gini	RSS	AUC
Training	8	0.0567	0.0612	0.3427	0.9981	0.3244	0.1135	266.9	0.7016
Validation	8	0.0599	0.0642	0.2989	0.9957	0.3286	0.1160	121.2	0.6659

The last three columns of this table contain statistics related to the ROC curve plotted in Output 68.4.5. This model does not classify the observations with Bad=1 (the event of interest) very well, as you can see in the confusion matrix and from the low sensitivity reported in this table. This is often the case with a relatively rare event. The AUC measures the area under the ROC curve. A model that fits the data perfectly would have an AUC of 1. These three columns are included only for a binary response variable.

Output 68.4.7 displays the pruning plot.

Output 68.4.7 Pruning Plot

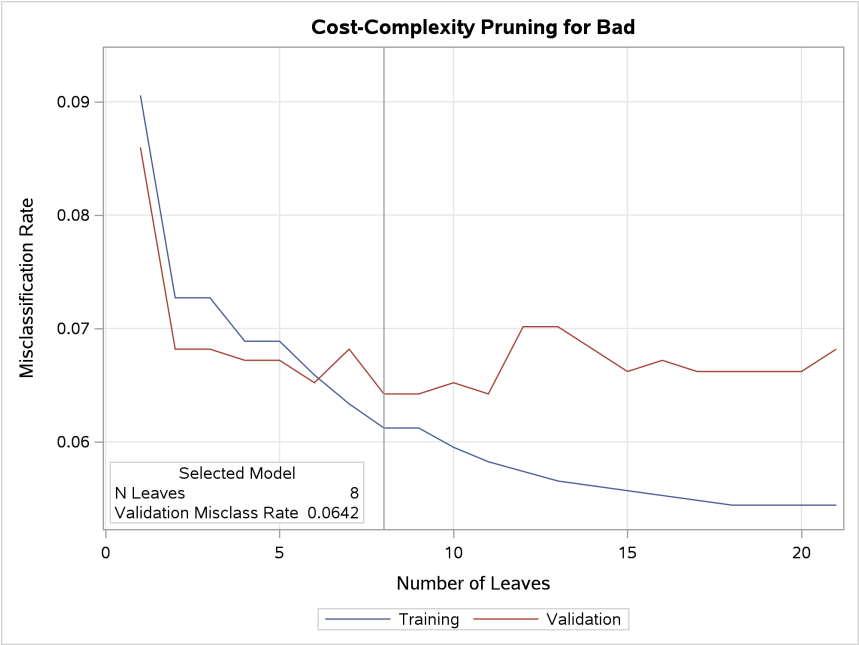

This plot displays misclassification rates for the training and validation data as the tree is pruned. The tree with eight leaves is selected as the final tree because it has the lowest misclassification rate for the validation data.

Creating Score Code and Scoring New Data

In addition to seeing information about the tree model, you might be interested in applying a model to predict the response variable in other data sets where the response is unknown. The following statements show how you can use the score code file *hpsplexc.sas*, created by the FILE= option in the CODE statement, to score the data in Hmeq and save the results in a SAS data set named Scored.

```
data scored;
   set hmeq;
   %include 'hpsplexc.sas';
run;
```

Output 68.4.8 shows a partial listing of Scored.

Output 68.4.8 Partial Listing of the Scored Hmeq Data

Obs	Bad	Loan	MortDue	Value	Reason	Job	YoJ	Derog	Delinq	CLAge	nInq	CLNo	DebtInc
1	1	1100	25860	39025	HomeImp	Other	10.5	0	0	94.367	1	9	.
2	1	1300	70053	68400	HomeImp	Other	7.0	0	2	121.833	0	14	.
3	1	1500	13500	16700	HomeImp	Other	4.0	0	0	149.467	1	10	.
4	1	1500	.	.			.	.	.	.	.	.	.
5	0	1700	97800	112000	HomeImp	Office	3.0	0	0	93.333	0	14	.
6	1	1700	30548	40320	HomeImp	Other	9.0	0	0	101.466	1	8	37.1136
7	1	1800	48649	57037	HomeImp	Other	5.0	3	2	77.100	1	17	.
8	1	1800	28502	43034	HomeImp	Other	11.0	0	0	88.766	0	8	36.8849
9	1	2000	32700	46740	HomeImp	Other	3.0	0	2	216.933	1	12	.
10	1	2000	.	62250	HomeImp	Sales	16.0	0	0	115.800	0	13	.

Obs	_Node_	_Leaf_	_WARN_	P_Bad0	P_Bad1	V_Bad0	V_Bad1
1	4	0		0.94393	0.05607	0.94432	0.05568
2	12	5		0.83168	0.16832	0.88462	0.11538
3	4	0		0.94393	0.05607	0.94432	0.05568
4	4	0		0.94393	0.05607	0.94432	0.05568
5	4	0		0.94393	0.05607	0.94432	0.05568
6	4	0		0.94393	0.05607	0.94432	0.05568
7	12	5		0.83168	0.16832	0.88462	0.11538
8	4	0		0.94393	0.05607	0.94432	0.05568
9	12	5		0.83168	0.16832	0.88462	0.11538
10	4	0		0.94393	0.05607	0.94432	0.05568

The data set contains the 13 original variables and the 7 new variables that are created by the score code. The variable P_Bad1 is the proportion of training observations on this leaf for which Bad=1, and this variable can be interpreted as the probability of default. The variable V_Bad1 is the proportion of validation observations on this leaf for which Bad=1. Similar variables are created for the other response value, Bad=0. Also included in the scored data set are the node and leaf assignments, which are shown in the variables _Node_ and _Leaf_, respectively. Note that these same variables are included in the OUT= data set when you specify that option in the OUTPUT statement. For more information about these variables, see the section "Scoring" on page 5220.

You can use the preceding statements to score new data by including the new data set in place of Hmeq in the SET statement. The new data set must contain the same variables as the data that are used to build the tree model, but not the unknown response variable that you are now trying to predict.

Creating a Node Rules Description of a Tree

When you specify the FILE= option in the RULES statement, a file that contains the node rules is created. In this example, PROC HPSPLIT saves the node rules of the tree model to a file named *rules.txt*; a partial listing of this file is shown in Output 68.4.9.

Output 68.4.9 Rules File

```
*-------------------------------------------------------------*
NODE = 4
*-------------------------------------------------------------*
MISSING(Delinq) OR (Delinq IS ONE OF 0, 1, 8, 10, 11, 12, 13, 15)
AND MISSING(DebtInc) OR (DebtInc < 43.843383)
    PREDICTED VALUE IS 0
    PREDICTED 0 = 0.9439( 2037/2158)
    PREDICTED 1 = 0.05607( 121/2158)
*-------------------------------------------------------------*
NODE = 10
*-------------------------------------------------------------*
MISSING(Reason) OR (Reason IS DebtCon)
AND (DebtInc < 46.7104)
AND (DebtInc >= 43.843383)
    PREDICTED VALUE IS 1
    PREDICTED 0 = 0.1176( 2/17)
    PREDICTED 1 = 0.8824( 15/17)
*-------------------------------------------------------------*
NODE = 13
*-------------------------------------------------------------*
MISSING(CLNo) OR (CLNo < 30.08)
AND (Delinq IS ONE OF 4, 5, 6, 7)
AND MISSING(Value) OR (Value < 129275.32)
AND (Delinq IS ONE OF 2, 3, 4, 5, 6, 7)
AND MISSING(DebtInc) OR (DebtInc < 43.843383)
    PREDICTED VALUE IS 1
    PREDICTED 0 = 0( 0/11)
    PREDICTED 1 = 1( 11/11)
*-------------------------------------------------------------*
NODE = 14
*-------------------------------------------------------------*
(CLNo >= 30.08)
AND (Delinq IS ONE OF 4, 5, 6, 7)
AND MISSING(Value) OR (Value < 129275.32)
AND (Delinq IS ONE OF 2, 3, 4, 5, 6, 7)
AND MISSING(DebtInc) OR (DebtInc < 43.843383)
    PREDICTED VALUE IS 0
    PREDICTED 0 = 1( 5/5)
    PREDICTED 1 = 0( 0/5)
```

In this listing, the predicted value and the fraction of observations for each level of the response variable are displayed for each terminal node. The nodes are not numbered consecutively because only terminal nodes (leaves) are included. The splits that lead to each leaf are shown above the predicted value and fractions.

Example 68.5: Assessing Variable Importance

(View the complete code for this example (hpsplex5.sas) in the example repository.)

This example creates a classification tree model to determine important variables (parameters) during the manufacture of a semiconductor device. Some of the variables that are involved in the manufacturing process are as follows: gTemp is the growth temperature of substrate, aTemp is the anneal temperature, Rot is rotation speed, Dopant is the atom that is used during device growth, and Usable indicates whether the device is usable.

The following statements create a data set named MBE_Data, which contains measurements for 20 devices:

```
data MBE_Data;
   label gTemp  = 'Growth Temperature of Substrate';
   label aTemp  = 'Anneal Temperature';
   label Rot    = 'Rotation Speed';
   label Dopant = 'Dopant Atom';
   label Usable = 'Experiment Could Be Performed';

   input gTemp aTemp Rot Dopant $ 39-40 Usable $ 47-54;
   datalines;
   384.614    633.172    1.01933      C       Unusable
   363.874    512.942    0.72057      C       Unusable
   397.395    671.179    0.90419      C       Unusable
   389.962    653.940    1.01417      C       Unusable
   387.763    612.545    1.00417      C       Unusable
   394.206    617.021    1.07188      Si      Usable
   387.135    616.035    0.94740      Si      Usable
   428.783    745.345    0.99087      Si      Unusable
   399.365    600.932    1.23307      Si      Unusable
   455.502    648.821    1.01703      Si      Unusable
   387.362    697.589    1.01623      Ge      Usable
   408.872    640.406    0.94543      Ge      Usable
   407.734    628.196    1.05137      Ge      Usable
   417.343    612.328    1.03960      Ge      Usable
   482.539    669.392    0.84249      Ge      Unusable
   367.116    564.246    0.99642      Sn      Unusable
   398.594    733.839    1.08744      Sn      Unusable
   378.032    619.561    1.06137      Sn      Usable
   357.544    606.871    0.85205      Sn      Unusable
   384.578    635.858    1.12215      Sn      Unusable
   ;
```

The following statements create the tree model:

```
proc hpsplit data=MBE_Data maxdepth=6;
   class Usable Dopant;
   model Usable = gTemp aTemp Rot Dopant;
   prune none;
run;
```

Output 68.5.1 shows the “Variable Importance” table.

Output 68.5.1 Variable Importance

The HPSPLIT Procedure

Variable Importance				
		Training		
Variable	Variable Label	Relative	Importance	Count
gTemp	Growth Temperature of Substrate	1.0000	2.1022	2
Dopant	Dopant Atom	0.7522	1.5811	1
aTemp	Anneal Temperature	0.6228	1.3093	1
Rot	Rotation Speed	0.3250	0.6831	1

This table shows that the predictor gTemp has the largest value. This means that the growth temperature of substrate is the most important consideration in determining the usability of the sample.

References

Breiman, L., Friedman, J., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Belmont, CA: Wadsworth.

Cutler, R., Brown, L., Powell, J., Bentz, B., and Cutler, A. (2003). "Identifying 'Redtops': Classification of Satellite Imagery for Tracking Mountain Pine Beetle Progression through a Pine Forest." In *Security and Infrastructure Protection: Proceedings of the 35th Symposium on the Interface*, edited by A. Braverman, T. Hesterberg, M. Minnotte, J. Symanzik, and Y. Said. Fairfax Station, VA: Interface Foundation of North America Inc.

Edwards, T. C., Jr., Cutler, D. R., Zimmermann, N. E., Geiser, L., and Alegria, J. (2005). "Model-Based Stratifications for Enhancing the Detection of Rare Ecological Events." *Ecology* 86:1081–1090.

Friedman, J. H. (1977). "A Recursive Partitioning Decision Rule for Nonparametric Classification." *IEEE Transactions on Computers* 26:404–408.

Geiser, L. H., and Neitlich, P. N. (2007). "Air Pollution and Climate Gradients in Western Oregon and Washington Indicated by Epiphytic Macrolichens." *Environmental Pollution* 145:203–218.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer-Verlag.

Kass, G. V. (1980). "An Exploratory Technique for Investigating Large Quantities of Categorical Data." *Journal of the Royal Statistical Society, Series C* 29:119–127.

Kuhn, M., and Johnson, K. (2013). *Applied Predictive Modeling*. New York: Springer.

Lichman, M. (2013). "UCI Machine Learning Repository." School of Information and Computer Sciences, University of California, Irvine. http://archive.ics.uci.edu/ml.

Molina, R., McKenzie, D., Lesher, R., Ford, J., Alegria, J., and Cutler, R. (2003). *Strategic Survey Framework for the Northwest Forest Plan Survey and Manage Program*. General Technical Report PNW-GTR-573, US Department of Agriculture, Forest Service, Pacific Northwest Research Station.

Old-Growth Definition Task Group (1986). *Interim Definitions for Old-Growth Douglas-Fir and Mixed-Conifer Forests in the Pacific Northwest and California*. Research Note PNW-447, US Department of Agriculture, Forest Service, Pacific Northwest Research Station.

Quinlan, J. R. (1986). "Induction of Decision Trees." *Machine Learning* 1:81–106.

Quinlan, J. R. (1987). "Simplifying Decision Trees." *International Journal of Man-Machine Studies* 27:221–234.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. San Francisco: Morgan Kaufmann.

Time Inc. (1987). "What They Make." *Sports Illustrated* (April 20): 54–81.

Zhang, H., and Singer, B. H. (2010). *Recursive Partitioning and Applications*. 2nd ed. New York: Springer.

Chapter 69
The ICLIFETEST Procedure

Contents

Overview: ICLIFETEST Procedure **5262**
- Features 5262

Getting Started: ICLIFETEST Procedure **5263**

Syntax: ICLIFETEST Procedure **5268**
- PROC ICLIFETEST Statement 5269
- BY Statement 5276
- FREQ Statement 5277
- STRATA Statement 5277
- TEST Statement 5278
- TIME Statement 5281

Details: ICLIFETEST Procedure **5282**
- Statistical Methods 5282
- Missing Values 5296
- Output Data Sets 5296
- Displayed Output 5297
- ODS Table Names 5300
- ODS Graphics 5301

Examples: ICLIFETEST Procedure **5302**
- Example 69.1: Analyzing Data with Observations below a Limit of Detection 5302
- Example 69.2: Controlling the Plotting of Survival Estimates 5304
- Example 69.3: Plotting Kernel-Smoothed Hazard Functions 5306
- Example 69.4: Outputting Scores for Permutation Tests 5307

References **5308**

Overview: ICLIFETEST Procedure

The ICLIFETEST procedure performs nonparametric survival analysis for interval-censored data. You can use the ICLIFETEST procedure to compute nonparametric estimates of the survival functions and to examine the equality of the survival functions through statistical tests.

The ICLIFETEST procedure compares most closely to the LIFETEST procedure. The two procedures share the same analytic objectives: estimating and summarizing subjects' survival experiences and comparing them systematically. The distinction between these procedures lies in the types of data that they are designed to handle. The ICLIFETEST procedure is intended primarily for handling interval-censored data, whereas the LIFETEST procedure deals exclusively with right-censored data. You can use the ICLIFETEST procedure to analyze data that are left-censored, interval-censored, or right-censored. However, if the data to be analyzed contain only exact or right-censored observations, it is recommended that you use the LIFETEST procedure because it provides specialized methods for dealing with right-censored data.

Features

The main features of the ICLIFETEST procedure are as follows:

- **Nonparametric estimation**
  - uses the efficient EMICM algorithm (Wellner and Zhan 1997) to estimate survival functions by default
  - supports Turnbull's algorithm (Turnbull 1976) and the iterative convex minorant (ICM) algorithm (Groeneboom and Wellner 1992)
  - computes standard errors of the survival estimates by using a multiple imputation method or a bootstrap method
  - supports several transformation-based confidence intervals
  - produces survival plots
- **Survival comparisons**
  - provides the weighted generalized log-rank test
  - supports a variety of weight functions for testing early or late differences
  - supports a stratified test for survival differences within predefined populations
  - supports a trend test for ordered alternatives
  - supports multiple-comparison functionalities

The ICLIFETEST procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Getting Started: ICLIFETEST Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/iclftgs1.sas.)

This example illustrates the use of the ICLIFETEST procedure to estimate the survival function and test for equality of survival functions by using interval-censored data from a breast cancer study.

The data consist of observations on 94 subjects from a retrospective study that compares the risks of breast cosmetic deterioration after tumorectomy (Finkelstein 1986). There are two treatment groups: patients who receive radiation alone (TRT=RT) and patients who receive radiation plus chemotherapy (TRT=RCT). Patients are followed up every four to six months, leading to interval-censored observations of deterioration times. Of the 94 observation times, 38 are right-censored and the remaining 56 are censored into intervals of finite length.

The following statements create a SAS data set named RT for the group that receives radiation alone. The variable lTime provides the last follow-up time at which cosmetic deterioration has not occurred for the patient, and the variable rTime provides the last follow-up time immediately after the event. Note that for the ICLIFETEST procedure to recognize the observations as right-censored, their right bounds must be represented by missing values in the input data set.

```
data RT;
   input lTime rTime @@;
   trt = ' RT';
   datalines;
45  .  25 37  37  .
 6 10  46  .   0  5
 0  7  26 40  18  .
46  .  46  .  24  .
46  .  27 34  36  .
 7 16  36 44   5 11
17  .  46  .  19 35
 7 14  36 48  17 25
37 44  37  .  24  .
 0  8  40  .  32  .
 4 11  17 25  33  .
15  .  46  .  19 26
11 15  11 18  37  .
22  .  38  .  34  .
46  .   5 12  36  .
46  .
;
```

The following statements create a SAS data set named RCT for patients who receive both radiation and chemotherapy:

```
data RCT;
   input lTime rTime @@;
   trt = 'RCT';
   datalines;
 8 12   0  5  30 34
 0 22   5  8  13  .
```

```
24 31  12 20  10 17
17 27  11  .   8 21
17 23  33 40   4  9
24 30  31  .  11  .
16 24  13 39  14 19
13  .  19 32   4  8
11 13  34  .  34  .
16 20  13  .  30 36
18 25  16 24  18 24
17 26  35  .  16 60
32  .  15 22  35 39
23  .  11 17  21  .
44 48  22 32  11 20
14 17  10 35  48  .
;
```

The following statements combine the data sets RT and RCT into a single data set named BCS that is to be analyzed by the ICLIFETEST procedure:

```
data BCS;
   set RT RCT;
run;
```

Suppose you want to gain insights into the incidence rate of cosmetic deterioration over time after the two treatments. From the perspective of survival analysis, the tasks are to estimate the survival probabilities for the two treatment groups and to test for a systematic difference between the groups.

The following statements invoke the ICLIFETEST procedure to estimate the survival functions for both treatment groups:

```
ods graphics on;
proc iclifetest plots=(survival logsurv) data=BCS impute(seed=1234);
   strata trt;
   time (lTime, rTime);
run;
```

In the TIME statement, the variables that represent the interval boundaries, lTime and rTime, are enclosed in parentheses and separated by a comma. Because the treatment indicator variable, Trt, is specified in the STRATA statement, PROC ICLIFETEST conducts the analysis separately for each treatment group. When you specify the keywords SURVIVAL and LOGSURV in the PLOTS= option, the procedure plots the estimated survival functions and the negative of the log transformations of the estimates. You can specify an integer seed for the random number generator that is used in creating imputed data sets for calculating standard errors of the survival estimates. If the SEED= option is not specified, a random seed is obtained from the computer's clock. For more information about the IMPUTE option, see the section "IMPUTE <(*options*)>" on page 5271.

Figure 69.1 displays the nonparametric survival estimates for the radiation group (TRT=RT).

Figure 69.1 Nonparametric Survival Estimates

The ICLIFETEST Procedure

Stratum 1: trt = RT

Nonparametric Survival Estimates				
		Probability Estimate		
Time Interval		Failure	Survival	Imputation Standard Error
0	4	0.0000	1.0000	0.0000
5	6	0.0463	0.9537	0.0354
7	7	0.0797	0.9203	0.0458
8	11	0.1684	0.8316	0.0580
12	24	0.2391	0.7609	0.0629
25	33	0.3318	0.6682	0.0706
34	38	0.4136	0.5864	0.0739
40	46	0.5344	0.4656	0.0758
48	Inf	1.0000	0.0000	0.0000

Note that because interval censoring is present, the nonparametric estimates of failure probability and survival probability are computed for a set of nonoverlapping intervals. These estimates are constant within each interval. For example, consider the time interval (8, 11]. The estimated failure probability is 0.1684 between 8 and 11. The interval (11, 12], which is not displayed, is an interval for which the survival estimate cannot be uniquely determined; such intervals are referred to as Turnbull intervals. The failure probability increases to 0.2391 at 12 and remains constant until 24. The table in Figure 69.1 also contains standard errors for the estimates. For a plot of the survival probabilities, see Figure 69.4.

Figure 69.2 displays estimated quartiles of the deterioration time and corresponding 95% confidence limits for the group that receives radiation only.

Figure 69.2 Quartile Estimates

Quartile Estimates				
			95% Confidence Interval	
Percentile	Point Estimate	Transform	[Lower	Upper)
75	.	LOGLOG	.	.
50	40	LOGLOG	34	48
25	25	LOGLOG	8	34

Because the estimated failure distribution is undefined inside the Turnbull intervals, the quartiles of deterioration time and their confidence intervals are obtained by imputing probabilities within the intervals. This approach is ad hoc in nature, so the results should be interpreted with caution. For more information, see the section "Quartile Estimation" on page 5287.

The ICLIFETEST procedure also produces a summary of the frequencies of various types of censoring (Figure 69.3).

Figure 69.3 Summary of Various Censoring Types

Number of Censored and Uncensored Values						
			Type of Censoring			
Stratum ID	trt	Total	Left	Interval	Right	Uncensored
1	RT	46	3 (6.5%)	18 (39.1%)	25 (54.3%)	0 (0.0%)
2	RCT	48	2 (4.2%)	33 (68.8%)	13 (27.1%)	0 (0.0%)
Total		94	5 (5.3%)	51 (54.3%)	38 (40.4%)	0 (0.0%)

The categories are exact times, left-censoring, right-censoring, and interval-censoring. The percentage of each category within a stratum is also calculated.

Figure 69.4 displays the estimated survival functions for the two treatments.

Figure 69.4 Nonparametric Survival Estimates by Treatment

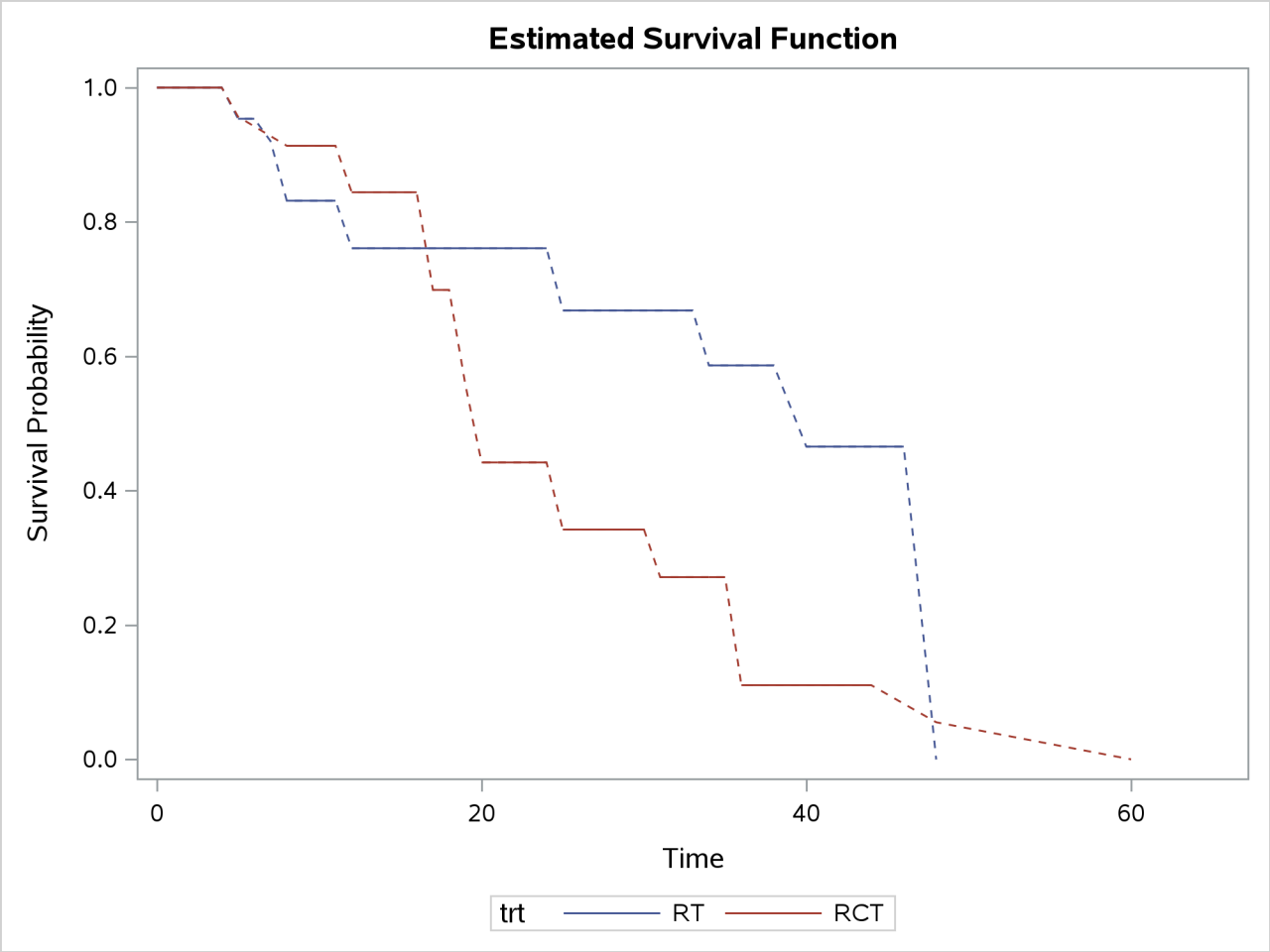

Clearly, the group that receives radiation alone tends to survive longer before experiencing cosmetic deterioration than the group that receives both radiation and chemotherapy. As shown in Figure 69.1, the estimated survival probabilities are undetermined within the Turnbull intervals. For ease of visualization, dashed lines are plotted across the Turnbull intervals for which the estimates are not defined. You can suppress plotting of

the dashed lines by specifying the NODASH option. For more information about options for customizing the survival plots, see the section "SURVIVAL <(CL | FAILURE | NODASH | STRATA | TEST)>" on page 5275.

Figure 69.5 displays log transformations of the survival estimates over time.

Figure 69.5 Negative Log of Nonparametric Survival Estimates

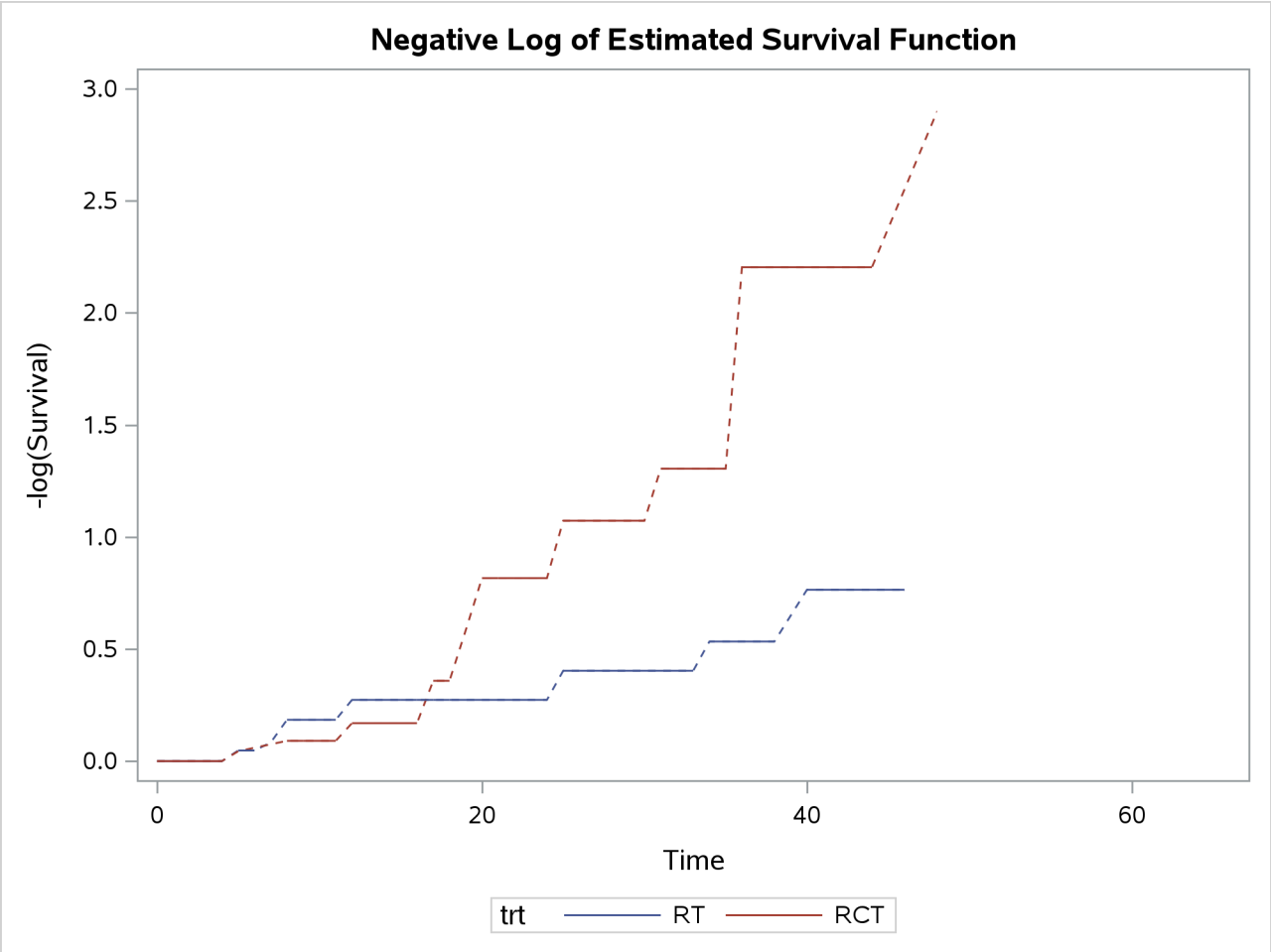

The following statements formally test whether patients in the two treatment groups have the same survival rate. By default, the generalized log-rank test with constant weights over time is used. For more information about alternative weight functions, see the section "Generalized Log-Rank Statistic" on page 5290. You can specify an integer seed for the random number generator that is used in creating imputed data sets for obtaining standard errors of the survival estimates and for calculating the generalized log-rank test. If the SEED= option is not specified, a random seed is obtained from the computer's clock.

```
proc iclifetest data=BCS impute(seed=1234);
   time (lTime, rTime);
   test trt;
run;
```

Figure 69.6 displays results of the generalized log-rank test.

Figure 69.6 Generalized Log-Rank Statistics

The ICLIFETEST Procedure

Test of Equality over Group			
Weight	Chi-Square	DF	Pr > Chi-Square
SUN	7.3349	1	0.0068

The chi-square statistic for the generalized log-rank test is 7.3349. The p-value of 0.0068 is computed using the chi-square distribution with one degree of freedom.

Syntax: ICLIFETEST Procedure

The following statements are available in the ICLIFETEST procedure:

PROC ICLIFETEST < *options* > **;**
  BY *variables* **;**
  FREQ *variable* **;**
  STRATA *variables* **;**
  TEST *variable* < */options* > **;**
  TIME (*variable,variable*) **;**

At a minimum, you must provide the PROC and TIME statements, and you must specify the left and right boundaries of the intervals in the TIME statements. For instance, the following statements compute a nonparametric estimate of the survival function:

```
proc iclifetest;
   time (Left, Right);
run;
```

You use the STRATA statement to obtain separate survival estimates for different groups of the input data. The ICLIFETEST procedure produces an individual survival estimate for each group that is formed by one or more variables that you specify in the STRATA statement.

You use the TEST statement to test whether the underlying survival functions are the same between the groups. When you specify both the TEST and STRATA statements, PROC ICLIFETEST produces a stratified test in which the comparisons are conditional on the strata. The variables that you specify in the STRATA statement must be different from the variables that you specify in the TEST statement. Note that this setup is different from that of the LIFETEST procedure, in which survival comparisons are handled implicitly by the STRATA statement.

The rest of this section provides detailed syntax information for each statement, beginning with the PROC ICLIFETEST statement. The remaining statements are covered in alphabetical order.

PROC ICLIFETEST Statement

PROC ICLIFETEST < *options* > **;**

The PROC ICLIFETEST statement invokes the ICLIFETEST procedure. Table 69.1 summarizes the options available in the PROC ICLIFETEST statement.

Table 69.1 Options Available in the PROC ICLIFETEST Statement

Option	Description
Input and Output Data Sets	
DATA=	Specifies the input SAS data set
OUTSURV=	Names an output data set to contain survival estimates and confidence limits
Nonparametric Estimation	
BOOTSTRAP	Requests that the simple bootstrap method be used for computing standard errors of the survival estimates
IMPUTE	Specifies details for multiple imputations used for computing standard errors and estimating covariance matrix of the generalized log-rank statistic
ITHISTORY	Displays the iteration history of nonparametric estimation
MAXITER=	Set the maximum number of iterations allowed for the estimation method
METHOD=	Specifies the method to estimate the survival function
TOLLIKE=	Sets the log-likelihood convergence criterion
Confidence Limits for Survival Function	
ALPHA=	Sets the level for confidence interval estimation
CONFTYPE=	Specifies the transformation applied to the survival function for computing confidence limits
ODS Graphics	
MAXTIME=	Specifies the maximum time value for plotting
PLOTS=	Specifies plots to display
Control Output	
NOPRINT	Suppresses the display of printed output
NOSUMMARY	Suppresses the display of summary of censored and uncensored values
SHOWTI	Displays the Turnbull intervals
Miscellaneous	
ALPHAQT=	Sets the level for confidence intervals for survival time quartiles
MISSING	Treats missing values as a valid stratum or group for the variables specified in the STRATA and TEST statements
SINGULAR=	Sets the tolerance for testing singularity of the covariance matrix of test statistics

If no options are specified, PROC ICLIFETEST computes and displays a nonparametric estimate of the survival function. If ODS Graphics is enabled, a plot of the estimated survival function is also displayed.

You can specify the following options.

ALPHA=α
specifies the level α for the $100(1-\alpha)\%$ confidence intervals for the survival functions. For example, ALPHA=0.05 requests 95% confidence limits. By default, ALPHA=0.05.

ALPHAQT=α
specifies the level α for the $100(1-\alpha)\%$ confidence intervals for the quartiles of the survival time. For example, ALPHAQT=0.05 requests a 95% confidence interval. By default, ALPHAQT=0.05.

BOOTSTRAP <(*options*)>
BOOT <(*options*)>
performs simple bootstrap sampling and computes bootstrap standard errors of nonparametric survival estimates. You can specify the following suboptions to control how the bootstrap samples are generated:

NBOOT
specifies the number of bootstrap samples to be generated. The default value is 1,000.

SEED
specifies the seed to generate bootstrap samples. The default seed is selected randomly.

CONFTYPE=ASINSQRT | LINEAR | LOG | LOGLOG
specifies the transformation to be applied to $S(t)$ to obtain pointwise confidence intervals for the survival function in addition to the confidence intervals for the quartiles of the survival times. You can specify the following keywords:

ASINSQRT specifies the arcsine–square root transformation,

$$g(x) = \sin^{-1}(\sqrt{x})$$

LOGLOG specifies the log–log transformation,

$$g(x) = \log(-\log(x))$$

This is also referred to as the log cumulative hazard transformation, because it applies the log transformation to the cumulative hazard function. Collett (1994) and Lachin (2000) call it the complementary log-log transformation.

LINEAR specifies the identity transformation,

$$g(x) = x$$

LOG specifies the log transformation,

$$g(x) = \log(x)$$

LOGIT specifies the logit transformation,

$$g(x) = \log\left(\frac{x}{1-x}\right)$$

By default, CONFTYPE=LOGLOG.

DATA=*SAS-data-set*
names the SAS data set that PROC ICLIFETEST reads. By default, the most recently created SAS data set is used.

IMPUTE <(*options***)>**
IM <(*options***)>**
specifies details for multiple imputations. You can specify the following two suboptions to control how the samples are generated:

NIMSE
specifies the number of imputation samples to be generated for computing standard errors of the survival estimates. The default value is 1,000.

NIMTEST
specifies the number of imputation samples to be generated for estimating the covariance matrix of the generalized log-rank statistic. The default value is 1,000.

SEED
specifies the seed to generate imputation data sets. The default seed is selected randomly.

ITHISTORY
prints the iteration history of the nonparametric estimation, including the log likelihood, the probability estimate that is associated with each Turnbull interval, and whether the EM or ICM method is used for each iteration.

MAXITER=*n*
MAXIT=*n*
specifies the maximum number of iterations for estimating the survival function. The default value depends on the estimation method as follows:

- EMICM method: 200
- ICM method: 200
- Turnbull method: 500

Note that you specify the method with the METHOD= option.

MAXTIME=*value*
specifies the maximum value of the time variable that is allowed in plots so that outlying points do not determine the scale of the time axis of the plots. This option affects only the way in which plots are displayed and has no effect on any calculations.

METHOD=TURNBULL | ICM | EMICM
specifies the method to be used for computing survival function estimates. You can specify the following:

TURNBULL

EM

requests that the Turnbull method be used to estimate the survival function.

EMICM

requests that the EMICM algorithm be used to estimate the survival function.

ICM

requests that the iterative convex minorant algorithm be used to estimate the survival function.

By default, METHOD=EMICM.

MISSING

allows missing values to be a stratum level or a valid group for the variables that are specified in the STRATA and TEST statements.

NOPRINT

suppresses the display of output. This option is useful when only an output data set is needed. It temporarily disables the Output Delivery System (ODS); for more information about ODS, see Chapter 23, "Using the Output Delivery System."

NOSUMMARY

suppresses the summary table of the number of censored and uncensored values.

OUTSURV=*SAS-data-set*

OUTS=*SAS-data-set*

creates an output SAS data set to contain the estimates of the survival function and corresponding confidence limits for all the strata. For more information about the contents of the OUTSURV= data set, see the section "OUTSURV= Data Set" on page 5296.

PLOTS<(*global-plot-options***)>=***plot-request* **<(***options***)>**

PLOTS<(*global-plot-options***)>=(***plot-request* **<(***options***)> <...** *plot-request* **<(***options***)>>)**

controls the plots that are produced by using ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around it. Here are some examples:

```
plots=none
plots=(survival(cl) logsurv)
plots(only)=hazard
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc iclifetest plots=survival;
   time (Left,Right);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, then by default PROC ICLIFETEST produces a plot of the estimated survival functions.

You can specify the following *global-plot-option*:

ONLY
specifies that only the specified plots in the list be produced; otherwise, the default survival function plot is also displayed.

You can specify the following *plot-requests* and their options:

ALL
produces all appropriate plots. Specifying PLOTS=ALL is equivalent to specifying PLOTS=(SURVIVAL LOGSURV LOGLOGS HAZARD).

HAZARD < (*hazard-options*) >

H < *hazard-options* >
plots the kernel-smoothed hazard functions. You can specify the following *hazard-options*.

BANDWIDTH=*value* | *numeric-list* | **RANGE(***lower,upper***)**

BW=
specifies the bandwidth for kernel-smoothed estimation of the hazard function. You can specify one of the following:

value
sets the bandwidth to the given *value*.

numeric-list
selects the bandwidth from the specified *numeric-list* that minimizes the cross validation pseudo-likelihood.

RANGE(*lower,upper***)**
selects from the interval (*lower, upper*) the bandwidth that minimizes a cross validation pseudo-likelihood function. PROC ICLIFETEST uses the golden section search algorithm to find the minimum. If the interval contains more than one local minimum, there is no guarantee that the local minimum that the algorithm finds is also the global minimum.

For more information about the cross validation pseudo-likelihood function, see the section "Kernel-Smoothed Estimation" on page 5288. By default, BANDWIDTH= RANGE(0.1b,0.25b), where $b = g_u - g_l$; g_l is the left boundary of the first Turnbull interval; and g_u is the right boundary of the last Turnbull interval if it is finite or the left boundary of that interval multiplied by *1.1* otherwise.

SAMPLING=LEAVEONE | RANDOM

specifies how to partition the data set to form cross validation groups. You can specify the following values:

LEAVEONE

partitions the data set into leave-one-out subsets.

RANDOM

randomly assigns observations to cross validation groups with equal probabilities.

By default, SAMPLING=RANDOM.

GRIDL=*number*

specifies the lower grid limit for the kernel-smoothed estimate. The default value is the time origin.

GRIDU=*number*

specifies the upper grid limit for the kernel-smoothed estimate. The default value is the maximum input boundary value.

KERNEL=*kernel-option*

specifies the kernel to be used. You can specify the following values:

BIWEIGHT

BW

uses the following kernel: $K_{BW}(x) = \frac{15}{16}(1 - x^2)^2, \qquad -1 \le x \le 1$

EPANECHNIKOV

E

uses the following kernel: $K_E(x) = \frac{3}{4}(1 - x^2), \qquad -1 \le x \le 1$

UNIFORM

U

uses the following kernel: $K_U(x) = \frac{1}{2}, \qquad -1 \le x \le 1$

By default, KERNEL=EPANECHNIKOV.

CVFOLD=*number*

specifies the number of cross validation groups. This option is applicable only when SAMPLING=RANDOM. The default *number* is either 5 or the sample size, whichever value is less.

CVGRID=*number*

specifies the number of grid points to use in determining the cross validation pseudo-likelihood. By default, CVGRID=51.

HGRID=*number*
specifies the number of grid points for discretizing the smoothed hazard function. By default, HGRID=101.

LOGLOGS
LLS
plots the log of the negative log of the estimated survival function versus the log of time.

LOGSURV
LS
plots the negative log of the estimated survival function versus time.

NONE
suppresses all plots.

SURVIVAL < (CL | FAILURE | NODASH | STRATA | TEST) >
S < (CL | FAILURE | STRATA | TEST) >
plots the estimated survival function. You can customize the display by specifying the following values:

CL
displays pointwise confidence limits for the survival function.

FAILURE
F
changes all the displays for survival function to those for the failure function. For example, if you specify both the FAILURE and CL options, the plot displays the failure curves in addition to pointwise confidence limits for the failure function.

NODASH
suppresses the plotting of dashed lines as linear interpolations for the Turnbull intervals.

STRATA=INDIVIDUAL | OVERLAY | PANEL
specifies how to display the survival or failure curves for multiple strata. This option has no effect if there is only one stratum. You can specify one of the following values:

INDIVIDUAL
UNPACK
requests that a separate plot be displayed for each stratum.

OVERLAY
requests that the survival or failure curves for the strata be overlaid on the same plot.

PANEL
requests that separate plots for the strata be organized into panels of two or four plots, depending on the number of strata.

By default, STRATA=OVERLAY.

TEST
displays the p-value for a K-sample test that is specified in the TEST statement.

SHOWTI
presents the survival estimates in terms of Turnbull intervals.

SINGULAR=*value*
sets the tolerance for testing the singularity of the covariance matrix of test statistics.

TOLLIKE=*value*
sets the log-likelihood convergence criterion for the estimation algorithm. The default *value* depends on the estimation method as follows:

- EMICM method: 10^{-10}
- ICM method: 10^{-10}
- Turnbull method: 10^{-8}

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC ICLIFETEST to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the ICLIFETEST procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a variable that provides frequencies for each observation in the DATA= data set. Specifically, if n is the value of the FREQ variable for a given observation, then that observation is used n times.

The analysis that is produced by using a FREQ statement reflects the expanded number of observations. You can produce the same analysis (without the FREQ statement) by first creating a new data set that contains the expanded number of observations. For example, if the value of the FREQ variable is 5 for the first observation, the first five observations in the new data set are identical. Each observation in the old data set is replicated n_i times in the new data set, where n_i is the value of the FREQ variable for that observation.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

STRATA Statement

STRATA *variables* **;**

The STRATA statement identifies the variables that determine the stratum levels. You can specify more than one variable in the STRATA statement. By default, strata are formed according to the nonmissing values of these variables. However, missing values are treated as a valid stratum level if you specify the MISSING option in the PROC ICLIFETEST statement.

The specification of several STRATA *variables*, such as

```
strata A B C;
```

is equivalent to the A*B*C syntax of the TABLES statement in the FREQ procedure. The number of stratum levels usually grows very rapidly along with the number of STRATA variables, so you must be cautious when specifying the list of STRATA variables.

If you specify the STRATA statement without specifying a TEST statement, the ICLIFETEST procedure produces an individual survival estimate for each stratum. On the other hand, if you specify both the STRATA and TEST statements, the ICLIFETEST procedure produces a stratified test in which the comparisons are made among the groups that are formed by the levels of the TEST variable within strata. Also, individual survival estimates are computed for all the groups that are formed by the levels of the TEST variable within a stratum.

In contrast with the STRATA statement in the LIFETEST procedure, the STRATA statement in PROC ICLIFETEST does not support the options for specifying various weight functions, stratified tests, or trend tests and to make multiple-comparison adjustments for paired differences. Instead, PROC ICLIFETEST provides these options in the TEST statement.

TEST Statement

TEST *variable* < /*options* > **;**

In the preceding syntax, variable is a variable whose values determine the groups to be tested. The values for variable can be formatted or unformatted. If variable is a character or numeric variable, then the groups are defined by the unique values of the TEST variable. You can specify only one variable in the TEST statement.

When you are comparing more than two survival curves, a generalized log-rank test tells you whether the curves are significantly different from each other, but it does not identify which pairs of curves are different. A multiple-comparison adjustment of the p-values for the paired comparisons retains the same overall probability of a Type I error as the K-sample test. Two types of paired comparisons can be made: comparisons between all pairs of curves and comparisons between a control curve and all other curves. You use the DIFF= option to specify the comparison type, and you use the ADJUST= option to select a method of multiple-comparison adjustments.

Compared with the TEST statement in the LIFETEST procedure, the TEST statement in PROC ICLIFETEST is designed for comparing survival between predefined groups. Unlike the LIFETEST procedure, PROC ICLIFETEST does not support a similar test for detecting association with multiple covariates.

Table 69.2 summarizes the *options* available in the TEST statement.

Table 69.2 Options Available in the TEST Statement

Option	Description
Homogeneity Tests	
NODETAIL	Suppresses printing the test statistic and covariance matrix
NOTEST	Suppresses all tests
OUTSCORE=	Names an output data set to contain the scores derived from the permutation form of the generalized log-rank test
TREND	Requests a trend test
WEIGHT=	Specifies tests that correspond to various weight functions
Multiple Comparisons	
ADJUST=	Requests a multiple-comparison adjustment
DIFF=	Specifies the type of differences to consider

You can specify the following *options* in the TEST statement after a slash (/).

ADJUST=*method*

specifies the multiple-comparison method to use for adjusting the p-values of the paired tests. For mathematical details, see the section "Multiple-Comparison Adjustments" on page 5293; also see Westfall et al. (1999). You can specify the following adjustment *methods*:

BONFERRONI

BON

applies the Bonferroni correction to the raw p-values.

DUNNETT

performs Dunnett's two-tailed comparisons of the control group to all other groups. PROC ICLIFETEST uses the factor-analytic covariance approximation that is described in Hsu (1992) and identifies the adjustment in the results as "Dunnett-Hsu." ADJUST=DUNNETT is incompatible with DIFF=ALL.

SCHEFFE

performs Scheffé's multiple-comparison adjustment.

SIDAK

applies the Šidák correction to the raw p-values.

SMM

GTE

performs the paired comparisons based on the studentized maximum modulus test.

TUKEY

performs the paired comparisons based on Tukey's studentized range test. PROC ICLIFETEST uses the approximation that is described in Kramer (1956) and identifies the adjustment as "Tukey-Kramer" in the results. ADJUST=TUKEY is incompatible with DIFF=CONTROL.

SIMULATE < (*simulate-options*) >

computes the adjusted p-values from the simulated distribution of the maximum or maximum absolute value of a multivariate normal random vector. The simulation estimates q, the true $(1-\alpha)$ quantile, where α is the value of the ALPHA= *simulate-option*.

The number of samples for the simulation adjustment is set so that the tail area for the simulated q is within a certain accuracy radius γ of $1-\alpha$, with an accuracy confidence of $100(1-\epsilon)\%$. In equation form,

$$\Pr(|F(\hat{q}) - (1-\alpha)| \leq \gamma) \quad = \quad 1-\epsilon$$

where $\hat{q}$ is the simulated q and F is the true distribution function of the maximum; for more information, see Edwards and Berry (1987). By default, $\gamma = 0.005$ and $\epsilon = 0.01$, so the tail area of $\hat{q}$ is within 0.005 of 0.95 with 99% confidence.

You can specify the following *simulate-options*:

ACC=*value*

specifies the target accuracy radius γ of a $100(1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. By default, ACC=0.005.

ALPHA=*value*

specifies the value α for estimating the $(1-\alpha)$ quantile. The default *value* is the ALPHA= value in the PROC ICLIFETEST statement, or 0.05 if that option is not specified.

EPS=*value*

specifies the value ϵ for a $100(1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. The default *value* for the accuracy confidence is 99%, corresponding to EPS=0.01.

NSAMP=*n*
specifies the sample size for the simulation. By default, *n* is set based on the values of the target accuracy radius γ and accuracy confidence $100(1-\epsilon)\%$ for an interval for the true probability content of the estimated $(1-\alpha)$ quantile. With the default values for γ, ϵ, and α (0.005, 0.01, and 0.05, respectively), by default NSAMP=12604.

REPORT
specifies that a report of the simulation be displayed, including a listing of the parameters, such as γ, ϵ, and α, in addition to an analysis of various methods of estimating or approximating the quantile.

SEED=*number*
specifies an integer used to start the pseudorandom number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to 0, the seed is generated by default from reading the time of day from the computer's clock.

DIFF=ALL | CONTROL<(*'string'* < . . . , *'string'*>)>
specifies which pairs of survival curves to consider for the multiple comparisons. You can specify the following values:

ALL
requests all paired comparisons.

CONTROL <(*'string'* < . . . *'string'*>)>
requests comparisons of the control survival curve with all other survival curves. To specify the control curve, you specify the quoted strings of formatted values that represent the curve in parentheses. For example, if CELL=LARGE identifies the control group, you specify

```
DIFF=CONTROL('large')
```

If more than one variable is used to identify the curves (for example, if CELL=LARGE and SEX=F represent the control), you specify

```
DIFF=CONTROL('large' 'F')
```

The order of the quoted strings should correspond to the order of the TEST variable. If no *string* is specified as the control, the first group value is used.

By default, DIFF=ALL unless you specify ADJUST= DUNNETT, in which case DIFF=CONTROL.

NODETAIL
suppresses the display of the generalized log-rank statistics and the corresponding covariance matrices. If you specify the TREND option, the display of the scores for computing the trend test is suppressed.

NOTEST
suppresses the K-sample tests, stratified tests, and trend tests.

OUTSCORE=*SAS-data-set*

OUTSC=*SAS-data-set*

creates an output SAS data set to contain subject scores that are derived from a permutation form of the generalized log-rank statistic. For more information about the contents of the OUTSCORE= data set, see the section "OUTSCORE= Data Set" on page 5297.

TREND

computes the trend test for the null hypothesis that the survival rates are the same for the groups versus an ordered alternative. If the TEST variable is numeric, the unformatted values of the variable are used as the scores; otherwise, the scores are $1, 2, \ldots,$ in the order of the strata. For more information, see the section "Trend Tests" on page 5295.

WEIGHT=*test-request* | **(***test-request* < . . . *test-request*>**)**

requests weights to be applied to the generalized log-rank statistics to perform various tests. For more information about the various weight functions that the ICLIFETEST procedure supports, see the section "Generalized Log-Rank Statistic" on page 5290. You can specify the following *test-requests*:

FAY	specifies the test that uses Fay's weights (Fay 1999).
FINKELSTEIN	specifies the test that uses Finkelstein's weights (Finkelstein 1986).
FLEMING(ρ1, ρ2)	specifies the family of tests in Harrington and Fleming (1982), where ρ1 and ρ2 are nonnegative numbers. FLEMING(ρ1,ρ2) corresponds to the Fleming-Harrington G^{ρ} family (Fleming and Harrington 1981) when ρ2=0. Alternatively, you can specify a test in this family as FLEMING(ρ) with one argument. When ρ=0, the test becomes the log-rank test. When ρ=1, the test should be very close to the Peto-Peto test.
NONE	suppresses all comparison tests. Specifying WEIGHT=NONE is equivalent to specifying NOTEST.
SUN	specifies the test that uses Sun's weights (Sun 1996).

TIME Statement

TIME (*variable, variable***) ;**

The TIME statement names the variables that represent the interval-censored observations. The variables should be numeric. For example, the following statement identifies Left as the lower bound and Right as the upper bound of an observed interval:

```
time (Left, Right);
```

Observations for which Left < Right and both values are nonnegative are used. If the value for Left is missing or 0 and the value for Right is not missing, then the observation is treated as left-censored. If the Right value is missing and the value for Left is not missing, then the observation is treated as right-censored. Observations for which both variables are missing are not analyzed.

Details: ICLIFETEST Procedure

Statistical Methods

Nonparametric Estimation of the Survival Function

Suppose the event times for a total of n subjects, $T_1, T_2, \ldots, T_n$, are independent random variables with an underlying cumulative distribution function $F(t)$. Denote the corresponding survival function as $S(t)$. Interval-censoring occurs when some or all T_i's cannot be observed directly but are known to be within the interval $(L_i, R_i], L_i \leq R_i$.

The observed intervals might or might not overlap. It they do not overlap, then you can usually use conventional methods for right-censored data, with minor modifications. On the other hand, if some intervals overlap, you need special algorithms to compute an unbiased estimate of the underlying survival function.

To characterize the nonparametric estimate of the survival function, Peto (1973) and Turnbull (1976) show that the estimate can jump only at the right endpoint of a set of nonoverlapping intervals (also known as Turnbull intervals), $\{I_j = (q_j, p_j], j = 1, \ldots, m\}$. A simple algorithm for finding these intervals is to order all the boundary values $\{L_i, R_i, i = 1, \ldots, n\}$ with labels of L and R attached and then pick up the intervals that have L as the left boundary and R as the right boundary. For example, suppose that the data set contains only three intervals, $(1, 3]$, $(2, 4]$, and $(5, 6]$. The ordered values are $1(L), 2(L), 3(R), 4(R), 5(L), 6(R)$. Then the Turnbull intervals are $(2, 3]$ and $(5, 6]$.

For the exact observation $L_i = R_i = t$, Ng (2002) suggests that it be represented by the interval $(t - \epsilon, t)$ for a positive small value ϵ. If $R_j = t$ for an observation $(L_j, R_j]$ $(L_j < R_j)$, then the observation is represented by $(L_j + \epsilon, R_j - \epsilon)$.

Define $\theta_j = P(T \in I_j)$, $j = 1, \ldots, m$. Given the data, the survival function, $S(t)$, can be determined only up to equivalence classes t, which are complements of the Turnbull intervals. $S(t)$ is undefined if t is within some I_j. The likelihood function for $\boldsymbol{\theta} = \{\theta_j, j = 1, \ldots, m\}$ is then

$$L(\boldsymbol{\theta}) = \prod_{i=1}^{n} \Big(\sum_{j=1}^{m} \alpha_{ij} \theta_j \Big)$$

where α_{ij} is 1 if I_j is contained in $(L_i, R_i]$ and 0 otherwise.

Denote the maximum likelihood estimate for $\hat{\boldsymbol{\theta}}$ as $\hat{\boldsymbol{\theta}} = \{\hat{\theta}_j, j = 1, \ldots, m\}$. The survival function can then be estimated as

$$\hat{S}(t) = \sum_{k: p_k > t} \hat{\theta}_k, \quad t \notin \text{any } I_j, j = 1, \ldots, m$$

Estimation Algorithms

Peto (1973) suggests maximizing this likelihood function by using a Newton-Raphson algorithm subject to the constraint $\sum_{i=j}^{m} \theta_j = 1$. This approach has been implemented in the %ICE macro. Although feasible, the optimization becomes less stable as the dimension of $\boldsymbol{\theta}$ increases.

Treating interval-censored data as missing data, Turnbull (1976) derives a self-consistent equation for estimating the θ_j's:

$$\theta_j = \frac{1}{n}\sum_{i=1}^{n}\mu_{ij}(\boldsymbol{\theta}) = \frac{1}{n}\sum_{i=1}^{n}\frac{\alpha_{ij}\theta_j}{\sum_{j=1}^{m}\alpha_{ij}\theta_j}$$

where $\mu_{ij}(\boldsymbol{\theta})$ is the expected probability that the event T_i occurs within I_j for the ith subject, given the observed data.

The algorithm is an expectation-maximization (EM) algorithm in the sense that it iteratively updates $\boldsymbol{\theta}$ and $\mu_{ij}(\boldsymbol{\theta})$. Convergence is declared if, for a chosen number $\epsilon > 0$,

$$\sum_{j=1}^{m}|\hat{\theta}_j^{(l)} - \hat{\theta}_j^{(l-1)}| < \epsilon$$

where $\hat{\theta}_j^{(l)}$ denotes the updated value for θ_j after the lth iteration.

An alternative criterion is to declare convergence when increments of the likelihood are small:

$$|L(\hat{\theta}_j^{(l)}; j = 1, \ldots, m) - L(\hat{\theta}_j^{(l-1)}; j = 1, \ldots, m)| < \epsilon$$

There is no guarantee that the converged values constitute a maximum likelihood estimate (MLE). Gentleman and Geyer (1994) introduced the Kuhn-Tucker conditions based on constrained programming as a check of whether the algorithm converges to a legitimate MLE. These conditions state that a sufficient and necessary condition for the estimate to be a MLE is that the Lagrange multipliers $\gamma_j = n - c_j$ are nonnegative for all the θ_j's that are estimated to be zero, where c_j is the derivative of the log-likelihood function with respect to θ_j:

$$c_j = \frac{\partial \log(L)}{\partial \theta_j} = \sum_{i=1}^{n}\frac{\alpha_{ij}}{\sum_{j=1}^{m}\alpha_{ij}\theta_j}$$

You can use Turnbull's method by specifying METHOD=TURNBULL in the ICLIFETEST statement. The Lagrange multipliers are displayed in the "Nonparametric Survival Estimates" table.

Groeneboom and Wellner (1992) propose using the iterative convex minorant (ICM) algorithm to estimate the underlying survival function as an alternative to Turnbull's method. Define $\beta_j = F(p_j)$, $j = 1, \ldots, m$ as the cumulative probability at the right boundary of the jth Turnbull interval: $\beta_j = \sum_{k=1}^{j}\theta_k$. It follows that $\beta_m = 1$. Denote $\beta_0 = 0$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{m-1})'$. You can rewrite the likelihood function as

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n}\sum_{j=1}^{m}\alpha_{ij}(\beta_j - \beta_{j-1})$$

Maximizing the likelihood with respect to the θ_j's is equivalent to maximizing it with respect to the β_j's. Because the β_j's are naturally ordered, the optimization is subject to the following constraint:

$$C = \{\mathbf{x} = (\beta_1, \ldots, \beta_{m-1}) : 0 \le \beta_1 \le \cdots \le \beta_{m-1} \le 1\}$$

Denote the log-likelihood function as $l(\boldsymbol{\beta})$. Suppose its maximum occurs at $\hat{\boldsymbol{\beta}}$. Mathematically, it can be proved that $\hat{\boldsymbol{\beta}}$ equals the maximizer of the following quadratic function:

$$g^*(\mathbf{x}|\mathbf{y}, \mathbf{W}) = -\frac{1}{2}(\mathbf{x} - \mathbf{y})'\mathbf{W}(\mathbf{x} - \mathbf{y})$$

where $\mathbf{y} = \hat{\boldsymbol{\beta}} + \mathbf{W}^{-1}\nabla l(\hat{\boldsymbol{\beta}})$, $\nabla l(\cdot)$ denotes the derivatives of $l(\cdot)$ with respect to $\boldsymbol{\beta}$, and $\mathbf{W}$ is a positive definite matrix of size $(m-1) \times (m-1)$ (Groeneboom and Wellner 1992).

An iterative algorithm is needed to determine $\hat{\boldsymbol{\beta}}$. For the lth iteration, the algorithm updates the quantity

$$\mathbf{y}^{(l)} = \hat{\boldsymbol{\beta}}^{(l-1)} - \mathbf{W}^{-1}(\hat{\boldsymbol{\beta}}^{(l-1)})\nabla l(\hat{\boldsymbol{\beta}}^{(l-1)})$$

where $\hat{\boldsymbol{\beta}}^{(l-1)}$ is the parameter estimate from the previous iteration and $\mathbf{W}(\hat{\boldsymbol{\beta}}^{(l-1)}) = \mathrm{diag}(w_j, j = 1, \ldots, m-1)$ is a positive definite diagonal matrix that depends on $\hat{\boldsymbol{\beta}}^{(l-1)}$.

A convenient choice for $\mathbf{W}(\boldsymbol{\beta})$ is the negative of the second-order derivative of the log-likelihood function $l(\boldsymbol{\beta})$:

$$w_j = w_j(\boldsymbol{\beta}) = -\frac{\partial^2}{\partial \beta_j^2} l(\boldsymbol{\beta})$$

Given $\mathbf{y} = \mathbf{y}^{(l)} = (y_1^{(l)}, \ldots, y_{m-1}^{(l)})'$ and $\mathbf{W} = \mathbf{W}(\hat{\boldsymbol{\beta}}^{(l-1)})$, the parameter estimate for the lth iteration $\hat{\boldsymbol{\beta}}^{(l)}$ maximizes the quadratic function $g^*(\mathbf{x}|\mathbf{y}, \mathbf{W})$.

Define the cumulative sum diagram $\{P_k, k = 0, \ldots, m-1\}$ as a set of m points in the plane, where $P_0 = (0, 0)$ and

$$P_k = \Big(\sum_{i=1}^{k} w_i, \sum_{i=1}^{k} w_i y_i^{(l)}\Big)$$

Technically, $\hat{\boldsymbol{\beta}}^{(l)}$ equals the left derivative of the convex minorant, or in other words, the largest convex function below the diagram $\{P_k, k = 0, \ldots, m-1\}$. This optimization problem can be solved by the pool-adjacent-violators algorithm (Groeneboom and Wellner 1992).

Occasionally, the ICM step might not increase the likelihood. Jongbloed (1998) suggests conducting a line search to ensure that positive increments are always achieved. Alternatively, you can switch to the EM step, exploiting the fact that the EM iteration never decreases the likelihood, and then resume iterations of the ICM algorithm after the EM step. As with Turnbull's method, convergence can be determined based on the closeness of two consecutive sets of parameter values or likelihood values. You can use the ICM algorithm by specifying METHOD=ICM in the PROC ICLIFETEST statement.

As its name suggests, the EMICM algorithm combines the self-consistent EM algorithm and the ICM algorithm by alternating the two different steps in its iterations. Wellner and Zhan (1997) show that the converged values of the EMICM algorithm always constitute an MLE if it exists and is unique. The ICLIFETEST procedure uses the EMICM algorithm as the default.

Variance Estimation of the Survival Estimator

Peto (1973) and Turnbull (1976) suggest estimating the variances of the survival estimates by inverting the Hessian matrix, which is obtained by twice differentiating the log-likelihood function. This method can become less stable when the number of θ_j's increase as n increases. Simulations have shown that the confidence limits based on variances estimated with this method tend to have conservative coverage probabilities that are greater than the nominal level (Goodall, Dunn, and Babiker 2004).

Sun (2001b) proposes using two resampling techniques, simple bootstrap and multiple imputation, to estimate the variance of the survival estimator. The undefined regions that the Turnbull intervals represent create a special challenge using the bootstrap method. Because each bootstrap sample could have a different set of Turnbull intervals, some time points to evaluate the variances based on the original Turnbull intervals might be located within the intervals in a bootstrap sample, with the result that their survival probabilities become unknown. A simple ad hoc solution is to shrink the Turnbull interval to its right boundary and modify the survival estimates into a right continuous function:

$$\hat{S}_m(t) = \sum_{j:p_j>t} \hat{\theta}_j$$

Let M denote the number of resampling data sets. Let $A_1^k, \ldots, A_n^k$ denote the n independent samples from the original data with replacement, $k = 1, \ldots, M$. Let $\hat{S}_m^k(t)$ be the modified estimate of the survival function computed from the kth resampling data set. Then you can estimate the variance of $\hat{S}(t)$ by the sample variance as

$$\hat{\sigma}_b^2(t) = \frac{1}{M-1} \sum_{k=1}^{M} [\hat{S}_m^k(t) - \bar{S}_m(t)]^2$$

where

$$\bar{S}_m(t) = \frac{\sum_{k=1}^{M} \hat{S}_m^k(t)}{M}$$

The method of multiple imputations exploits the fact that interval-censored data reduce to right-censored data when all interval observations of finite length shrink to single points. Suppose that each finite interval has been converted to one of the p_j values it contains. For this right-censored data set, you can estimate the variance of the survival estimates via the well-known Greenwood formula as

$$\hat{\sigma}_G^2(t) = \hat{S}_{KM}^2(t) \sum_{q_j<t} \frac{d_j}{n_j(n_j - d_j)}$$

where d_j is the number of events at time p_j and n_j is the number of subjects at risk just prior to p_j, and $\hat{S}_{KM}(t)$ is the Kaplan-Meier estimator of the survival function,

$$\hat{S}_{KM}(t) = \prod_{q_j<t} \frac{n_j - d_j}{n_j}$$

Essentially, multiple imputation is used to account for the uncertainty of ranking overlapping intervals. The kth imputed data set is obtained by substituting every interval-censored observation of finite length with an exact event time randomly drawn from the conditional survival function:

$$\hat{S}_i(t) = \frac{\hat{S}_m(t) - \hat{S}_m(R_i+)}{\hat{S}_m(L_i) - \hat{S}_m(R_i+)}, \quad t \in (L_i, R_i]$$

Because $\hat{S}_m(t)$ only jumps at the p_j, this is a discrete function.

Denote the Kaplan-Meier estimate of each imputed data set as $\hat{S}^k_{KM}(t)$. The variance of $\hat{S}(t)$ is estimated by

$$\hat{\sigma}^2_I(t) = \hat{S}^2(t) \sum_{q_j < t} \frac{d'_j}{n'_j(n'_j - d'_j)} + \frac{1}{M-1} \sum_{k=1}^{M} [\hat{S}^k_{KM}(t) - \bar{S}_{KM}(t)]$$

where

$$\bar{S}_{KM}(t) = \frac{1}{M} \sum_{k=1}^{M} \hat{S}^k_{KM}(t)$$

and

$$d'_j = \sum_{i=1}^{n} \frac{\alpha_{ij}[\hat{S}(p_{j-1}) - \hat{S}(p_j)]}{\sum_{j=1}^{m} \alpha_{ij}[\hat{S}(p_{j-1}) - \hat{S}(p_j)]}$$

and

$$n'_j = \sum_{k=j}^{m} d'_j$$

Note that the first term in the formula for $\hat{\sigma}^2_I(t)$ mimics the Greenwood formula but uses expected numbers of deaths and subjects. The second term is the sample variance of the Kaplan-Meier estimates of imputed data sets, which accounts for between-imputation contributions.

Pointwise Confidence Limits of the Survival Function

Pointwise confidence limits can be computed for the survival function given the estimated standard errors. Let α be specified by the ALPHA= option. Let $z_{\alpha/2}$ be the critical value for the standard normal distribution. That is, $\Phi(-z_{\alpha/2}) = \alpha/2$, where Φ is the cumulative distribution function of the standard normal random variable.

Constructing the confidence limits for the survival function $S(t)$ as $\hat{S}(t) \pm z_{\alpha/2}\hat{\sigma}[\hat{S}(t)]$ might result in an estimate that exceeds the range [0,1] at extreme values of t. This problem can be avoided by applying a transformation to $S(t)$ so that the range is unrestricted. In addition, certain transformed confidence intervals for $S(t)$ perform better than the usual linear confidence intervals (Borgan and Liestøl 1990). You can use the CONFTYPE= option to set one of the following transformations: the log-log function (Kalbfleisch and Prentice 1980), the arcsine–square root function (Nair 1984), the logit function (Meeker and Escobar 1998), the log function, and the linear function.

Let g denote the transformation that is being applied to the survival function $S(t)$. Using the delta method, you estimate the standard error of $g(\hat{S}(t))$ by

$$\tau(t) = \hat{\sigma}\left[g(\hat{S}(t))\right] = g'\left(\hat{S}(t)\right)\hat{\sigma}[\hat{S}(t)]$$

where g' is the first derivative of the function g. The $100(1-\alpha)\%$ confidence interval for $S(t)$ is given by

$$g^{-1}\left\{g[\hat{S}(t)] \pm z_{\frac{\alpha}{2}} g'[\hat{S}(t)]\hat{\sigma}[\hat{S}(t)]\right\}$$

where g^{-1} is the inverse function of g. The choices for the transformation g are as follows:

- arcsine–square root transformation: The estimated variance of $\sin^{-1}\left(\sqrt{\hat{S}(t)}\right)$ is $\hat{\tau}^2(t) = \frac{\hat{\sigma}^2[\hat{S}(t)]}{4\hat{S}(t)[1-\hat{S}(t)]}$. The $100(1-\alpha)\%$ confidence interval for $S(t)$ is given by

$$\sin^2\left\{\max\left[0, \sin^{-1}(\sqrt{\hat{S}(t)}) - z_{\frac{\alpha}{2}}\hat{\tau}(t)\right]\right\} \le S(t) \le \sin^2\left\{\min\left[\frac{\pi}{2}, \sin^{-1}(\sqrt{\hat{S}(t)}) + z_{\frac{\alpha}{2}}\hat{\tau}(t)\right]\right\}$$

- linear transformation: This is the same as the identity transformation. The $100(1-\alpha)\%$ confidence interval for $S(t)$ is given by

$$\hat{S}(t) - z_{\frac{\alpha}{2}}\hat{\sigma}\left[\hat{S}(t)\right] \le S(t) \le \hat{S}(t) + z_{\frac{\alpha}{2}}\hat{\sigma}\left[\hat{S}(t)\right]$$

- log transformation: The estimated variance of $\log(\hat{S}(t))$ is $\hat{\tau}^2(t) = \frac{\hat{\sigma}^2(\hat{S}(t))}{\hat{S}^2(t)}$. The $100(1-\alpha)\%$ confidence interval for $S(t)$ is given by

$$\hat{S}(t)\exp\left(-z_{\frac{\alpha}{2}}\hat{\tau}(t)\right) \le S(t) \le \hat{S}(t)\exp\left(z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)$$

- log-log transformation: The estimated variance of $\log(-\log(\hat{S}(t))$ is $\hat{\tau}^2(t) = \frac{\hat{\sigma}^2[\hat{S}(t)]}{[\hat{S}(t)\log(\hat{S}(t))]^2}$. The $100(1-\alpha)\%$ confidence interval for $S(t)$ is given by

$$\left[\hat{S}(t)\right]^{\exp\left(z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)} \le S(t) \le \left[\hat{S}(t)\right]^{\exp\left(-z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)}$$

- logit transformation: The estimated variance of $\log\left(\frac{\hat{S}(t)}{1-\hat{S}(t)}\right)$ is

$$\hat{\tau}^2(t) = \frac{\hat{\sigma}^2(\hat{S}(t))}{\hat{S}^2(t)[1-\hat{S}(t)]^2}$$

  The $100(1-\alpha)\%$ confidence limits for $S(t)$ are given by

$$\frac{\hat{S}(t)}{\hat{S}(t) + \left[1-\hat{S}(t)\right]\exp\left(z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)} \le S(t) \le \frac{\hat{S}(t)}{\hat{S}(t) + \left[1-\hat{S}(t)\right]\exp\left(-z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)}$$

Quartile Estimation

The first quartile (25th percentile) of the survival time is the time beyond which 75% of the subjects in the population under study are expected to survive. For interval-censored data, it is problematic to define point estimators of the quartiles based on the survival estimate $\hat{S}(t)$ because of its undefined regions of Turnbull intervals. To overcome this problem, you need to impute survival probabilities within the Turnbull intervals. The previously defined estimator $\hat{S}_m(t)$ achieves this by placing all the estimated probabilities at the right boundary of the interval. The first quartile is estimated by

$$q_{.25} = \min\{t_j | \hat{S}_m(t_j) < 0.75\}$$

If $\hat{S}_m(t)$ is exactly equal to 0.75 from t_j to t_{j+1}, the first quartile is taken to be $(t_j + t_{j+1})/2$. If $\hat{S}_m(t)$ is greater than 0.75 for all values of t, the first quartile cannot be estimated and is represented by a missing value in the printed output.

The general formula for estimating the 100p percentile point is

$$q_p = \min\{t_j | \hat{S}_m(t_j) < 1 - p\}$$

The second quartile (the median) and the third quartile of survival times correspond to $p = 0.5$ and $p = 0.75$, respectively.

Brookmeyer and Crowley (1982) constructed the confidence interval for the median survival time based on the confidence interval for the survival function $S(t)$. The methodology is generalized to construct the confidence interval for the 100p percentile based on a g-transformed confidence interval for $S(t)$ (Klein and Moeschberger 1997). You can use the CONFTYPE= option to specify the g-transformation. The $100(1-\alpha)\%$ confidence interval for the first quantile survival time is the set of all points t that satisfy

$$\left| \frac{g(\hat{S}_m(t)) - g(1 - 0.25)}{g'(\hat{S}_m(t))\hat{\sigma}(\hat{S}(t))} \right| \le z_{1-\frac{\alpha}{2}}$$

where $g'(x)$ is the first derivative of $g(x)$ and $z_{1-\frac{\alpha}{2}}$ is the $100(1 - \frac{\alpha}{2})$ percentile of the standard normal distribution.

Kernel-Smoothed Estimation

After you obtain the survival estimate $\hat{S}(t)$, you can construct a discrete estimator for the cumulative hazard function. First, you compute the jumps of the discrete function as

$$\hat{\lambda}_j = \frac{c_j \hat{\theta}_j}{\sum_{k=j}^{m} c_k \hat{\theta}_k}, \quad j = 1, \ldots, m$$

where the c_j's have been defined previously for calculating the Lagrange multiplier statistic.

Essentially, the numerator and denominator estimate the number of failures and the number at risks that are associated with the Turnbull intervals. Thus these quantities estimate the increments of the cumulative hazard function over the Turnbull intervals.

The estimator of the cumulative hazard function is

$$\hat{\lambda}(t) = \sum_{k:p_k<t} \hat{\lambda}_k, \quad t \notin \text{any } I_j$$

Like $\hat{S}(t)$, $\hat{\lambda}(t)$ is undefined if t is located within some Turnbull interval I_j. To facilitate applying the kernel-smoothed methods, you need to reformulate the estimator so that it has only point masses. An ad hoc approach would be to place all the mass for a Turnbull interval at the right boundary. The kernel-based estimate of the hazard function is computed as

$$\tilde{h}(t, b) = -\frac{1}{b} \sum_{j=1}^{m} K\Big(\frac{t - p_j}{b}\Big) \hat{\lambda}_j$$

where $K(\cdot)$ is a kernel function and $b > 0$ is the bandwidth. You can estimate the cumulative hazard function by integrating $\tilde{h}(t, b)$ with respect to t.

Practically, an upper limit t_D is usually imposed so that the kernel-smoothed estimate is defined on $(0, t_D)$. The ICLIFETEST procedure sets the value depending on whether the right boundary of the last Turnbull interval is finite or not: $t_D = p_m$ if $p_m < \infty$ and $t_D = 1.2 * q_m$ otherwise.

Typical choices of kernel function are as follows:

- uniform kernel:

$$K_U(x) = \frac{1}{2}, \quad -1 \le x \le 1$$

- Epanechnikov kernel:

$$K_E(x) = \frac{3}{4}(1 - x^2), \quad -1 \le x \le 1$$

- biweight kernel:

$$K_{BW}(x) = \frac{15}{16}(1 - x^2)^2, \quad -1 \le x \le 1$$

For $t < b$, the symmetric kernels $K()$ are replaced by the corresponding asymmetric kernels of Gasser and Müller (1979). Let $q = \frac{t}{b}$. The modified kernels are as follows:

- uniform kernel:

$$K_{U,q}(x) = \frac{4(1 + q^3)}{(1 + q)^4} + \frac{6(1 - q)}{(1 + q)^3}x, \quad -1 \le x \le q$$

- Epanechnikov kernel:

$$K_{E,q}(x) = K_E(x)\frac{64(2 - 4q + 6q^2 - 3q^3) + 240(1 - q)^2 x}{(1 + q)^4(19 - 18q + 3q^2)}, \quad -1 \le x \le q$$

- biweight kernel:

$$K_{BW,q}(x) = K_{BW}(x)\frac{64(8 - 24q + 48q^2 - 45q^3 + 15q^4) + 1120(1 - q)^3 x}{(1 + q)^5(81 - 168q + 126q^2 - 40q^3 + 5q^4)}, \quad -1 \le x \le q$$

For $t_D - b \le t \le t_D$, let $q = \frac{t_D - t}{b}$. The asymmetric kernels for $t < b$ are used, with x replaced by $-x$.

The bandwidth parameter b controls how much "smoothness" you want to have in the kernel-smoothed estimate. For right-censored data, a commonly accepted method of choosing an optimal bandwidth is to use the mean integrated square error(MISE) as an objective criteria. This measure becomes difficult to adapt to interval-censored data because it no longer has a closed-form mathematical formula.

Pan (2000) proposes using a V-fold cross validation likelihood as a criterion for choosing the optimal bandwidth for the kernel-smoothed estimate of the survival function. The ICLIFETEST procedure implements this approach for smoothing the hazard function. Computing such a criterion entails a cross validation type procedure. First, the original data $\mathcal{D}$ are partitioned into V almost balanced subsets $\mathcal{D}^{(v)}$, $v = 1, \ldots, V$. Denote the kernel-smoothed estimate of the leave-one-subset-out data $\mathcal{D} - \mathcal{D}^{(v)}$ as $\hat{h}^{*(-v)}(t; b)$. The optimal bandwidth is defined as the one that maximizes the cross validation likelihood:

$$b_0 = \text{argmax}_b \sum_{v=1}^{V} L(\hat{h}^{*(-v)}(t; b)|\mathcal{D}^{(v)})$$

Comparison of Survival between Groups

If the TEST statement is specified, the ICLIFETEST procedure compares the K groups formed by the levels of the TEST variable using a generalized log-rank test. Let $S_k(t)$ be the underlying survival function of the kth group, $k = 1, \ldots, K$. The null and alternative hypotheses to be tested are

$$H_0\colon S_1(t) = S_2(t) = \cdots = S_K(t) \text{ for all } t$$

versus

$$H_1\colon \text{at least one of the } S_k(t)\text{'s is different for some } t$$

Let N_k denote the number of subjects in group k, and let n denote the total number of subjects ($n = N_1 + \cdots + N_K$).

Generalized Log-Rank Statistic

For the ith subject, let $\mathbf{z}_i = (z_{i1}, \ldots, z_{iK})'$ be a vector of K indicators that represent whether or not the subject belongs to the kth group. Denote $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_K)'$, where β_k represents the treatment effect for the kth group. Suppose that a model is specified and the survival function for the ith subject can be written as

$$S(t|\mathbf{z}_i) = S(t|\mathbf{z}_i'\boldsymbol{\beta}, \boldsymbol{\gamma})$$

where $\boldsymbol{\gamma}$ denotes the nuisance parameters.

It follows that the likelihood function is

$$L = \prod_{i=1}^{n} [S(L_i|\mathbf{z}_i'\boldsymbol{\beta}, \boldsymbol{\gamma}) - S(R_i|\mathbf{z}_i'\boldsymbol{\beta}, \boldsymbol{\gamma})]$$

where (L_i, R_i) denotes the interval observation for the ith subject.

Testing whether or not the survival functions are equal across the K groups is equivalent to testing whether all the β_j's are zero. It is natural to consider a score test based on the specified model (Finkelstein 1986).

The score statistics for $\boldsymbol{\beta}$ are derived as the first-order derivatives of the log-likelihood function evaluated at $\boldsymbol{\beta} = \mathbf{0}$ and $\hat{\boldsymbol{\gamma}}$.

$$\mathbf{U} = (U_1, \ldots, U_K)' = \frac{\partial \log(L)}{\partial \boldsymbol{\beta}}\Big|_{\boldsymbol{\beta}=\mathbf{0}, \hat{\boldsymbol{\gamma}}}$$

where $\hat{\boldsymbol{\gamma}}$ denotes the maximum likelihood estimate for the $\boldsymbol{\gamma}$, given that $\boldsymbol{\beta} = \mathbf{0}$.

Under the null hypothesis that $\boldsymbol{\beta} = \mathbf{0}$, all K groups share the same survival function $S(t)$. It is typical to leave $S(t)$ unspecified and obtain a nonparametric maximum likelihood estimate $\hat{S}(t)$ using, for instance, Turnbull's method. In this case, $\boldsymbol{\gamma}$ represents all the parameters to be estimated in order to determine $\hat{S}(t)$.

Suppose the given data generates m Turnbull intervals as $\{I_j = (q_j, p_j], j = 1, \ldots, m\}$. Denote the probability estimate at the right end point of the jth interval by $\hat{\theta}_j$. The nonparametric survival estimate is $\hat{S}(t) = \sum_{k:p_k>t} \hat{\theta}_k$ for $t \notin$ any I_j.

Under the null hypothesis, Fay (1999) showed that the score statistics can be written in the form of a weighted log-rank test as

$$U_k = \sum_{j=1}^{m} U_{kj} = \sum_{j=1}^{m} v_j \left(d_{kj}' - \frac{n_{kj}'}{n_j'} d_j' \right)$$

where

$$v_j = \frac{[\hat{S}(p_j) - \hat{S}'(p_{j-1})][\hat{S}(p_{j-1}) - \hat{S}'(p_j)]}{\hat{S}(p_j)[\hat{S}(p_{j-1}) - \hat{S}(p_j)]}$$

and $S'(t)$ denotes the derivative of $S(t)$ with respect to $\boldsymbol{\beta}$.

d'_{kj} estimates the expected number of events within I_j for the kth group, and it is computed as

$$d'_{kj} = \sum_{i=1}^{n} z_{ik} \frac{\alpha_{ij}\hat{\theta}_j}{\sum_{l=1}^{m} \alpha_{il}\hat{\theta}_l}$$

d'_j is an estimate for the expected number of events within I_j for the whole sample, and it is computed as

$$d'_j = \sum_{k=1}^{K} d'_{kj}$$

Similarly, n'_{kj} estimates the expected number of subjects at risk before entering I_j for the kth group, and can be estimated by $n'_{kj} = \sum_{l=j}^{m} d'_{kl}$. n'_j is an estimate of the expected number of subjects at risk before entering I_j for all the groups: $n'_j = \sum_{k=1}^{K} n'_{kj}$.

Assuming different survival models gives rise to different weight functions v_j (Fay 1999). For example, Finkelstein's score test (1986) is derived assuming a proportional hazards model; Fay's test (1996) is based on a proportional odds model.

The choices of weight function are given in Table 69.3.

Table 69.3 Weight Functions for Various Tests

Test	v_j
Sun (1996)	1.0
Fay (1999)	$\hat{S}(p_{j-1})$
Finkelstein (1986)	$\frac{\hat{S}(p_{j-1})[\log \hat{S}(p_{j-1}) - \log \hat{S}(p_j)]}{\hat{S}(p_{j-1}) - \hat{S}(p_j)}$
Harrington-Fleming (p,q)	$[\hat{S}(p_{j-1})]^p[1 - \hat{S}(p_{j-1})]^q, p \geq 0, q \geq 0$

Variance Estimation of the Generalized Log-Rank Statistic

Sun (2001a) proposed the use of multiple imputation to estimate the variance-covariance matrix of the generalized log-rank statistic **U**. This approach is similar to the multiple imputation method as presented in "Variance Estimation of the Survival Estimator" on page 5285. Both methods impute right-censored data from interval-censored data and analyze the imputed data sets by using standard statistical techniques. Huang, Lee, and Yu (2008) suggested improving the performance of the generalized log-rank test by slightly modifying the variance calculation.

Suppose the given data generate m Turnbull intervals as $\{I_j = (q_j, p_j], j = 1, \ldots, m\}$. Denote the probability estimate for the jth interval as $\hat{\theta}_j$, and denote the nonparametric survival estimate as $\hat{S}(t) =$

$\sum_{k:p_k>t} \hat{\theta}_k$ for $t \notin$ any I_j.

In order to generate an imputed data set, you need to randomly generate a survival time for every subject of the sample. For the ith subject, a random time T_i^* is generated randomly based on the following discrete survival function:

$$\hat{S}_i(T_i^* = p_j) = \frac{\hat{S}(q_j) - \hat{S}(R_i+)}{\hat{S}(L_i) - \hat{S}(R_i+)}, \quad p_j \in (L_i, R_i], \quad j = 1, \ldots, m$$

where $(L_i, R_i]$ denotes the interval observation for the subject.

For the hth imputed data set ($h = 1, \ldots, H$), let d_{kj}^h and n_{kj}^h denote the numbers of failures and subjects at risk by counting the imputed T_i^*'s for group k. Let d_j^h and n_j^h denote the corresponding pooled numbers.

You can perform the standard weighted log-rank test for right-censored data on each of the imputed data sets (Huang, Lee, and Yu 2008). The test statistic is

$$\mathbf{U}^h = (U_1^h, \ldots, U_K^h)'$$

where

$$U_k^h = \sum_{j=1}^{m} v_j \left(d_{kj}^h - \frac{n_{kj}^h}{n_j^h} d_j^h \right)$$

Its variance-covariance matrix is estimated by the Greenwood formula as

$$\mathbf{V}^h = \mathbf{V}_1^h + \cdots + \mathbf{V}_m^h$$

where

$$(\mathbf{V}_j^h)_{l_1 l_2} = \begin{cases} v_j^2 n_{l_1 j}^h (n_j^h - n_{l_1 j}^h) d_j^h (n_j^h - d_j^h)(n_j^h)^{-2}(n_j^h - 1)^{-1} & \text{when } l_1 = l_2 \\ -v_j^2 n_{l_1 j}^h n_{l_2 j}^h d_j^h (n_j^h - d_j^h)(n_j^h)^{-2}(n_j^h - 1)^{-1} & \text{when } l_1 \neq l_2 \end{cases}$$

After analyzing each imputed data set, you can estimate the variance-covariance matrix of $\mathbf{U}$ by pooling the results as

$$\hat{\mathbf{V}} = \frac{1}{H} \sum_{h=1}^{H} \mathbf{V}^h - \frac{1}{H-1} \sum_{h=1}^{H} [\mathbf{U}^h - \bar{\mathbf{U}}][\mathbf{U}^h - \bar{\mathbf{U}}]'$$

where

$$\bar{\mathbf{U}} = \frac{1}{H} \sum_{h=1}^{H} \mathbf{U}^h$$

The overall test statistic is formed as $\mathbf{U}'\mathbf{V}^-\mathbf{U}$, where $\mathbf{V}^-$ is the generalized inverse of $\mathbf{V}$. Under the null hypothesis, the statistic has a chi-squared distribution with degrees of freedom equal to the rank of $\mathbf{V}$. By default, the ICLIFETEST procedure perform 1000 imputations. You can change the number of imputations by the IMPUTE option in the PROC ICLIFETEST statement.

Stratified Tests

Suppose the generalized log-rank test is to be stratified on the M levels that are formed from the variables that you specify in the STRATA statement. Based only on the data of the sth stratum ($s = 1, \ldots, M$), let $\mathbf{U}_{(s)}$ be the test statistic for the sth stratum and let $V_{(s)}$ be the corresponding covariance matrix as constructed in the section "Variance Estimation of the Generalized Log-Rank Statistic" on page 5291. First, sum over the stratum-specific estimates as follows:

$$\mathbf{U}. = \sum_{s=1}^{M} \mathbf{U}_{(s)}$$

$$\mathbf{V}. = \sum_{s=1}^{M} \mathbf{V}_{(s)}$$

Then construct the global test statistic as

$$\mathbf{U}.'\mathbf{V}.^{-}\mathbf{U}.$$

Under the null hypothesis, the test statistic has a chi-squared distribution with degrees of freedom equal to the rank of $\mathbf{V}.$. The ICLIFETEST procedure performs the stratified test only when the groups to be compared are balanced across all the strata.

Multiple-Comparison Adjustments

When you have more than two groups, a generalized log-rank test tells you whether the survival curves are significantly different from each other, but it does not identify which pairs of curves are different. Pairwise comparisons can be performed based on the generalized log-rank statistic and the corresponding variance-covariance matrix. However, reporting all pairwise comparisons is problematic because the overall Type I error rate would be inflated. A multiple-comparison adjustment of the p-values for the paired comparisons retains the same overall probability of a Type I error as the K-sample test.

The ICLIFETEST procedure supports two types of paired comparisons: comparisons between all pairs of curves and comparisons between a control curve and all other curves. You use the DIFF= option to specify the comparison type, and you use the ADJUST= option to select a method of multiple-comparison adjustments.

Let χ^2_r denote a chi-square random variable with r degrees of freedom. Denote ϕ and Φ as the density function and the cumulative distribution function of a standard normal distribution, respectively. Let m be the number of comparisons; that is,

$$m = \begin{cases} \frac{k(k-1)}{2} & \text{DIFF} = \text{ALL} \\ k-1 & \text{DIFF} = \text{CONTROL} \end{cases}$$

For a two-sided test that compares the survival of the jth group with that of lth group, $1 \le j \ne l \le r$, the test statistic is

$$z^2_{jl} = \frac{(U_j - U_l)^2}{V_{jj} + V_{ll} - 2V_{jl}}$$

and the raw p-value is

$$p = \Pr(\chi^2_1 > z^2_{jl})$$

For multiple comparisons of more than two groups ($r > 2$), adjusted p-values are computed as follows:

- Bonferroni adjustment:

$$p = \min\{1, m\Pr(\chi_1^2 > z_{jl}^2)\}$$

- Dunnett-Hsu adjustment: With the first group defined as the control, there are $r-1$ comparisons to be made. Let $\mathbf{C} = (c_{ij})$ be the $(r-1) \times r$ matrix of contrasts that represents the $r-1$ comparisons; that is,

$$c_{ij} = \begin{cases} 1 & i = 1, \ldots, r-1, j = 2, \ldots, r \\ -1 & j = i+1, i = 2, \ldots, r \\ 0 & \text{otherwise} \end{cases}$$

Let $\mathbf{\Sigma} \equiv (\sigma_{ij})$ and $\mathbf{R} \equiv (r_{ij})$ be covariance and correlation matrices of $\mathbf{Cv}$, respectively; that is,

$$\mathbf{\Sigma} = \mathbf{CVC}'$$

and

$$r_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}}$$

The factor-analytic covariance approximation of Hsu (1992) is to find $\lambda_1, \ldots, \lambda_{r-1}$ such that

$$\mathbf{R} = \mathbf{D} + \boldsymbol{\lambda}\boldsymbol{\lambda}'$$

where $\mathbf{D}$ is a diagonal matrix whose jth diagonal element is $1 - \lambda_j$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_{r-1})'$. The adjusted p-value is

$$p = 1 - \int_{-\infty}^{\infty} \phi(y) \prod_{i=1}^{r-1} \left[\Phi\left(\frac{\lambda_i y + z_{jl}}{\sqrt{1-\lambda_i^2}} \right) - \Phi\left(\frac{\lambda_i y - z_{jl}}{\sqrt{1-\lambda_i^2}} \right) \right] dy$$

This value can be obtained in a DATA step as

$$p = \text{PROBMC("DUNNETT2"}, z_{ij}, ., ., r-1, \lambda_1, \ldots, \lambda_{r-1}).$$

- Scheffé adjustment:

$$p = \Pr(\chi_{r-1}^2 > z_{jl}^2)$$

- Šidák adjustment:

$$p = 1 - \{1 - \Pr(\chi_1^2 > z_{jl}^2)\}^m$$

- SMM adjustment:

$$p = 1 - [2\Phi(z_{jl}) - 1]^m$$

This can also be evaluated in a DATA step as

$$p = 1 - \text{PROBMC("MAXMOD"}, z_{jl}, ., ., m).$$

- Tukey adjustment:

$$p = 1 - \int_{-\infty}^{\infty} r\phi(y)[\Phi(y) - \Phi(y - \sqrt{2}z_{jl})]^{r-1}\, dy$$

This can be evaluated in a DATA step as

$$p = 1 - \text{PROBMC("RANGE"}, \sqrt{2}z_{jl}, ., ., r).$$

Trend Tests

Trend tests for right-censored data (Klein and Moeschberger 1997, Section 7.4) can be extended to interval-censored data in a straightforward way. Such tests are specifically designed to detect ordered alternatives as

$$H_1\colon S_1(t) \geq S_2(t) \geq \cdots \geq S_K(t), t \leq \tau, \text{ with at least one inequality}$$

or

$$H_2\colon S_1(t) \leq S_2(t) \leq \cdots \leq S_K(t), t \leq \tau, \text{ with at least one inequality}$$

Let $a_1 < a_2 < \cdots < a_K$ be a sequence of scores associated with the K samples. Let $\mathbf{U} = (U_1, \ldots, U_K)$ be the generalized log-rank statistic and $\mathbf{V} = (V_{jl})$ be the corresponding covariance matrix of size $K \times K$ as constructed in the section "Variance Estimation of the Generalized Log-Rank Statistic" on page 5291. The trend test statistic and its variance are given by $\sum_{j=1}^{K} a_j U_j$ and $\sum_{j=1}^{K}\sum_{l=1}^{K} a_j a_l V_{jl}$, respectively. Under the null hypothesis that there is no trend, the following z-score has, asymptotically, a standard normal distribution:

$$Z = \frac{\sum_{j=1}^{K} a_j U_j}{\sqrt{\{\sum_{j=1}^{K}\sum_{l=1}^{K} a_j a_l V_{jl}\}}}$$

The ICLIFETEST procedure provides both one-tail and two-tail p-values for the test.

Scores for Permutation Tests

The weighted log-rank statistic can also be expressed as

$$\mathbf{U} = \sum_{i=1}^{n} \mathbf{z}_i c_i$$

where c_i is the score from the ith subject and follows the form

$$c_i = \frac{\hat{S}'(L_i) - \hat{S}'(R_i)}{\hat{S}(L_i) - \hat{S}(R_i)}$$

where $S'(\cdot)$ denotes the derivative of $S(\cdot)$ with respect to $\boldsymbol{\beta}$, which is evaluated at $\boldsymbol{\beta} = \mathbf{0}$.

As presented in Table 69.4, Fay (1999) derives the forms of scores for three weight functions. Under the assumption that censoring is independent of the grouping of subjects, these derived scores can be used by permutation tests.

Table 69.4 Scores for Different Weight Functions

Test Weight	v_j
Sun (1996)	$-\frac{\hat{S}(L_i)\hat{\lambda}(L_i)-\hat{S}(R_i)\hat{\lambda}(R_i)}{\hat{S}(L_i)-\hat{S}(R_i)}$
Fay (1999)	$\hat{S}(L_i) + \hat{S}(R_i) - 1$
Finkelstein (1986)	$\frac{\hat{S}(L_i)\log[\hat{S}(L_i)]-\hat{S}(R_i)\log[\hat{S}(R_i)]}{\hat{S}(L_i)-\hat{S}(R_i)}$

You can output scores to a designated SAS data set by specifying the OUTSCORE= option in the TEST

statement.

Missing Values

Observations that have a missing value for both the left boundary and the right boundary are not used in the analysis. If a STRATA or FREQ variable value is missing, the observation is not used. However, you can treat missing values as an independent stratum if you specify the MISSING option in the PROC ICLIFETEST statement. If any variable that you specify in the TEST statement has a missing value, that observation is not used in the calculation of test statistics unless you specify the MISSING option in the PROC ICLIFETEST statement.

Output Data Sets

OUTSURV= Data Set

You can specify the OUTSURV= option in the PROC ICLIFETEST statement to create an output data set that contains the following columns:

- any variables that are specified using the BY statement
- any variables that are specified using the STRATA statement
- any variable that is specified using the TEST statement
- LeftBoundary and RightBoundary, the two boundary variables that represent a Turnbull interval
- SurvProb, a variable that contains the survival function estimates at RightBoundary
- FailProb, a variable that contains the failure probability estimates at RightBoundary
- IMStderr, a variable that contains the standard errors that are estimated by using multiple imputations
- BTStderr, a variable that contains the bootstrap standard errors
- ConfType, a variable that contains the name of the transformation applied to the survival time in the computation of confidence intervals
- LagrangeMult, a variable that contains the Lagrange multiplier statistics for checking the Kuhn-Tucker conditions
- SurvProb_LCL, a variable that contains the lower limits of the pointwise confidence intervals for the survival function, using imputation standard errors
- SurvProb_UCL, a variable that contains the upper limits of the pointwise confidence intervals for the survival function, using imputation standard errors
- SurvProb_LCL_BT, a variable that contains the lower limits of the pointwise confidence intervals for the survival function, using bootstrap standard errors
- SurvProb_UCL_BT, a variable that contains the upper limits of the pointwise confidence intervals for the survival function, using bootstrap standard errors

OUTSCORE= Data Set

You can specify the OUTSCORE= option in the PROC ICLIFETEST statement to create an output data set that contains the following columns:

- any variables that are specified using the BY statement
- any variables that are specified using the STRATA statement
- any variable that is specified using the TEST statement
- Score, a variable that contains the scores for each subject
- ScoreType, a variable that contains the name of the weight function for which the scores are calculated

Displayed Output

If you use the NOPRINT option in the PROC ICLIFETEST statement, the procedure does not display any output.

Data and Method Information

The "Data and Method Information" table is displayed each time PROC ICLIFETEST is called. The table displays the following:

- input SAS data set name
- input left and right boundaries
- FREQ variable name
- method used to compute survival estimates
- number of observations read
- number of observations used
- number of bootstrap samples
- number of multiple imputation for calculating standard errors of the survival estimates
- number of multiple imputation for estimating covariance matrix of the generalized log-rank statistics (if the TEST statement is specified)
- seed for multiple imputation
- seed for bootstrap (if the BOOTSTRAP option is specified in the PROC ICLIFETEST statement)

The ODS name of this table is DataInfo.

Nonparametric Survival Estimates

The "Nonparametric Survival Estimates" table is displayed by default. The table displays the following:

- left boundary of time intervals
- right boundary of time intervals
- estimate of survival probability
- estimate of failure probability
- standard error of the survival estimate by the multiple imputation method
- bootstrapped standard error of the survival estimate (if the BOOTSTRAP option is specified in the PROC ICLIFETEST statement)
- Lagrange multiplier (if METHOD=TURNBULL|EM is specified in the PROC ICLIFETEST statement)

The ODS name of this table is ProbabilityEstimates.

Estimates at Right Boundaries of Turnbull Intervals

The "Estimates at Right Boundaries of Turnbull Intervals" table is displayed when you specify the SHOWTI option. The table displays the following:

- left boundary of Turnbull intervals
- right boundary of Turnbull intervals
- estimate of survival probability at the right boundary
- estimate of failure probability at the right boundary
- standard error of the survival estimate by the multiple imputation method
- bootstrapped standard error of the survival estimate (if the BOOTSTRAP option is specified in the PROC ICLIFETEST statement)
- Lagrange multiplier (if METHOD=TURNBULL is specified in the PROC ICLIFETEST statement)

The ODS name of this table is TurnbullIntervals.

Iteration History of Probability Estimates Associated with Turnbull Intervals

The "Iteration History of Probability Estimates Associated with Turnbull Intervals" table is displayed when you specify the ITHISTORY option. The table displays the following:

- iteration index
- type of updating method (EM or ICM) for the current iteration
- nonparametric log likelihood at each iteration
- probability estimate that is associated with each Turnbull interval

The ODS name of this table is IterHistory.

Summary of Censored and Uncensored Values

The "Summary of Censored and Uncensored Values" table is displayed by default. The table displays the following:

- STRATA variables, if there are any, and their values
- index of strata that are formed by the specified STRATA variables
- TEST variable, if there is one, and its values
- index of groups that are formed by the specified TEST variable
- frequency of exact observations
- frequency of left censorings
- frequency of interval censorings
- frequency of right censorings
- percentage of each type of censoring within a stratum

The ODS name of this table is CensoredSummary.

Quartile Estimates

The "Quartiles Estimates" table is displayed if nonparametric estimation finds an MLE successfully. The table displays the following:

- point estimates of the quartiles of the survival times
- lower and upper confidence limits for the quartiles

The ODS name of this table is Quartiles.

Generalized Log-Rank Statistics

The "Rank Statistics" table contains the test statistics of the nonparametric K-sample tests. The ODS name of this table is HomStats.

Covariance Matrix for the Generalized Log-Rank Statistics

The "Covariance Matrix for the Log-Rank Statistics" table is displayed together with the generalized log-rank statistics. The ODS name of this table is HomCov.

Test of Equality over Group

The "Test of Equality over Group" table is displayed if an unstratified K-sample test is performed. The table contains the chi-square statistics, degrees of freedom, and p-value of the generalized log-rank test. The ODS name of this table is HomTests.

Stratified Test of Equality over Group

The "Stratified Test of Equality over Group" table is displayed if a stratified test is carried out. The table contains the chi-square statistics, degrees of freedom, and p-values of the stratified tests. The ODS name of this table is HomTests2.

Scores for Trend Test

The "Scores for Trend Test" table is displayed if the TREND option is specified in the TEST statement. The table contains the set of scores that are used to construct the trend test. The ODS name of this table is TrendScores.

Trend Test

The "Trend Tests" table is displayed if the TREND option is specified in the TEST statement. The table contains the results of the trend test. The ODS name of this table is TrendTest.

Pairwise Group Comparisons

The "Pairwise Group Comparisons" table is displayed if a multiple-comparison adjustment method is specified. The table contains the chi-square statistics and the raw and adjusted p-values of the paired comparisons. The ODS name of this table is SurvDiff.

ODS Table Names

PROC ICLIFETEST assigns a name to each table that it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 69.5. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 69.5 ODS Tables Produced by PROC ICLIFETEST

ODS Table Name	Description	Statement / Option
CensoredSummary	Number of censored and uncensored observations	Default
ConvergenceStatus	Convergence status	Default
DataInfo	Data and methods information	Default
HomCov	Covariance matrix for the generalized log-rank statistics	TEST
HomStats	Generalized log-rank statistics	TEST
HomTests	Results of K-sample tests	TEST | TEST / WEIGHT=
IterHistory	Iteration history of nonparametric estimation	ITHISTORY

Table 69.5 *continued*

ODS Table Name	Description	Statement / Option
ProbabilityEstimates	Nonparametric survival estimates	Default
Quartiles	Quartile estimates	Default
SurvivalDiff	Adjustments for multiple comparisons	TEST / ADJUST= | TEST / DIFF=
TrendScores	Scores for trend test	TEST / TREND
TrendTest	Trend test results	TEST / TREND
TurnbullIntervals	Turnbull intervals	SHOWTI

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

The survival plot is produced by default; other graphs are produced by specifying the PLOTS= option in the PROC ICLIFETEST statement. You can refer by name to every graph produced through ODS Graphics. The names of the graphs that PROC ICLIFETEST generates are listed in Table 69.6, along with the required keywords for the PLOTS= option.

Table 69.6 Graphs Produced by PROC ICLIFETEST

ODS Graph Name	Plot Description	PLOTS=Option
FailurePlot	Estimated failure function	SURVIVAL(FAILURE)
LogNegLogSurvivalPlot	Log of the negative log of the estimated survival function	LOGLOGS
NegLogSurvivalPlot	Negative log of the estimated survival function	LOGSURV
BandwidthGrid	Cross validation pseudo-log likelihood by bandwidth	HAZARD(BW=*numeric-list*) HAZARD(BW=RANGE(*lower,upper*))
SmoothedHazard	Kernel-smoothed hazard estimate	HAZARD
SurvivalPlot	Estimated survival function	Default or SURVIVAL
SurvivalPlot	Estimated survival function with pointwise confidence limits	SURVIVAL(CL)
SurvivalPlot	Estimated survival function with K-sample test p-value	SURVIVAL(TEST)

Examples: ICLIFETEST Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The following examples illustrate some of the capabilities of the ICLIFETEST procedure. They are not intended to represent definitive analyses of the data sets that are presented here. See the references for guidance on complete analysis of interval-censored data by appropriate statistical methods.

Example 69.1: Analyzing Data with Observations below a Limit of Detection

(View the complete code for this example (iclftex1.sas) in the example repository.)

Data that have certain values below a limit of detection (LOD) are frequently encountered by toxicologists and environmental scientists. Such data are usually analyzed by imputing the unobserved values by LOD/2 or LOD/$\sqrt{2}$. This type of practice often raises the question of whether the population distributions can be estimated without bias. Gillespie et al. (2010) propose using a reverse Kaplan-Meier estimator, or equivalently, Turnbull's method (1976) by treating the unobserved data as left-censored. When the assumption of independent censoring holds, these estimators can unbiasedly estimate the population distribution functions.

The following hypothetical data have two values, 3 and 10, that are below the limit of detection:

```
data temp;
   input C1 C2;
   datalines;
  .     3
  4     4
  6     6
  8     8
  .     10
  12    12
;
```

The following statements invoke PROC ICLIFETEST to estimate the population distribution function by using Turnbull's method:

```
proc iclifetest data=temp method=turnbull plots=survival(failure)
                impute(seed=1234);
   time (c1,c2);
run;
```

Specifying the PLOTS=SURVIVAL(FAILURE) option requests a failure probability plot. Results are shown in Output 69.1.1. Note that because the first Turnbull interval is $(0, 3)$, the failure probability function is undefined within that interval.

Output 69.1.1 Failure Probability Plot for Fictitious Nondetection Data

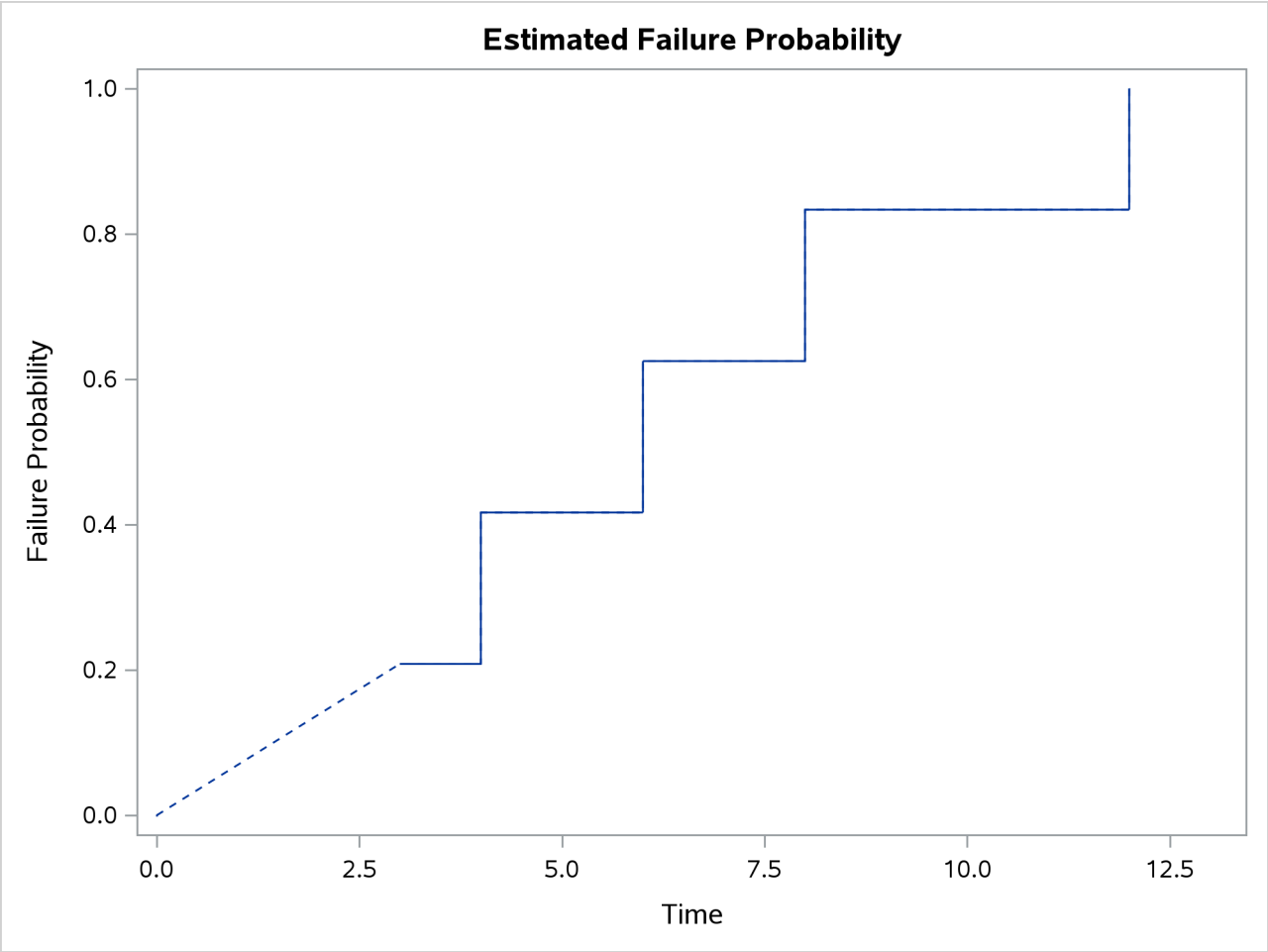

Output 69.1.2 presents the estimated failure probability, with standard errors that are estimated by the method of multiple imputations.

Output 69.1.2 Cumulative Probability Estimates

The ICLIFETEST Procedure

Nonparametric Survival Estimates					
		Probability Estimate			
Time Interval		Failure	Survival	Imputation Standard Error	Lagrange Multiplier
3	4	0.2083	0.7917	0.1811	0.0000
4	6	0.4167	0.5833	0.2179	0.0000
6	8	0.6250	0.3750	0.2099	0.0000
8	12	0.8333	0.1667	0.1521	0.0000
12	Inf	1.0000	0.0000	0.0000	0.0000

Example 69.2: Controlling the Plotting of Survival Estimates

(View the complete code for this example (iclftex2.sas) in the example repository.)

Recall the breast cancer data in the example in the section "Getting Started: ICLIFETEST Procedure" on page 5263. The following statements compute complementary log-log transformed pointwise confidence limits for the two treatment groups and plot them:

```
proc iclifetest data=BCS plots=survival(cl) impute(seed=1234);
   strata trt;
   time (lTime, rTime);
run;
```

The two areas overlap a lot before the 20th month. After that, the radiation-only group tends to have higher survival rate than the radiation-plus-chemotherapy group. That is, patients who receive only radiation tend to survive longer to experience cosmetic deterioration than those who receive both treatments.

Output 69.2.1 Nonparametric Survival Estimates for Breast Cancer Data

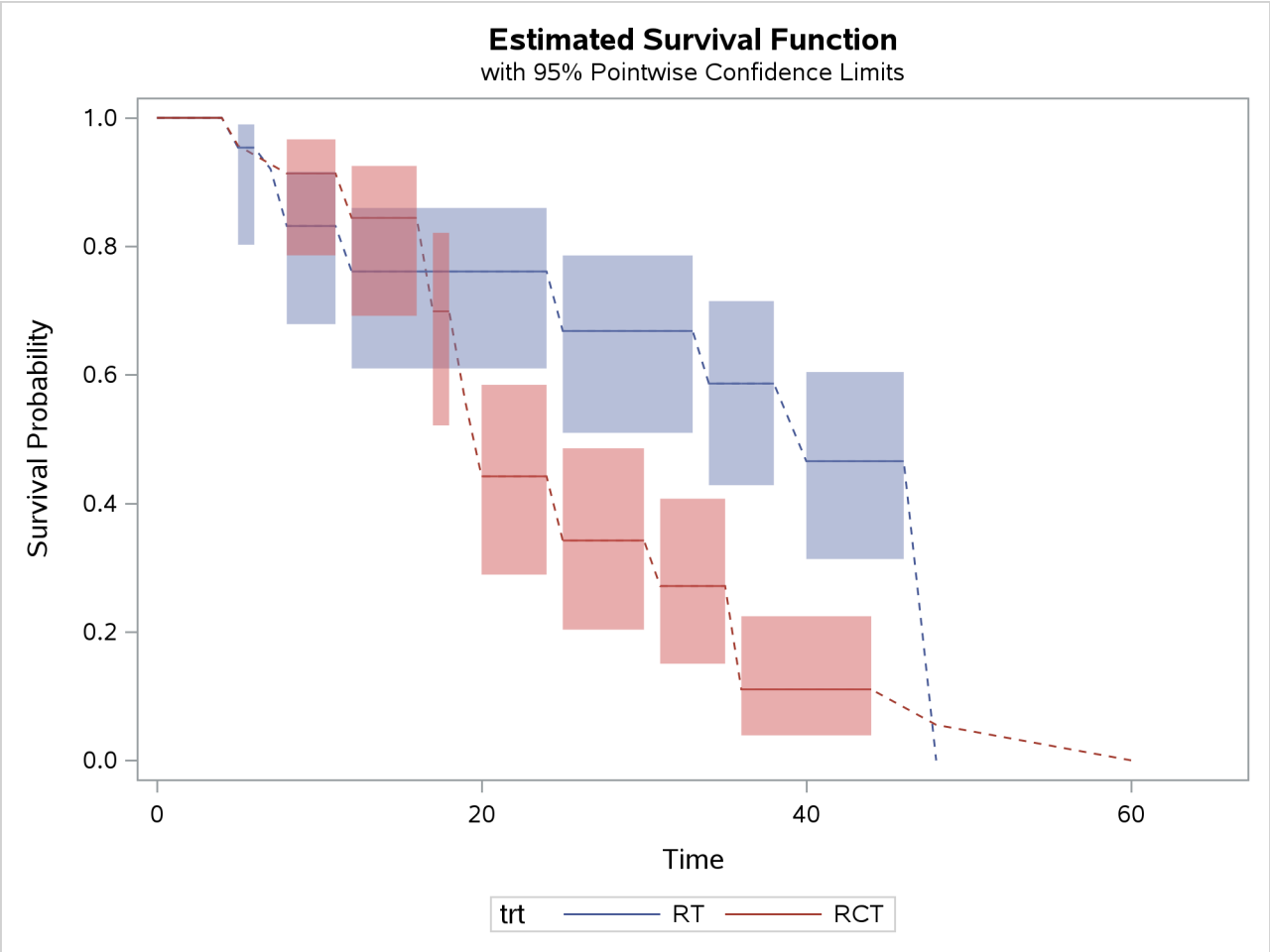

The dashed lines visually connect the survival estimates across Turnbull intervals for which the estimates are not defined. You can choose not to display the dashed lines by specifying the NODASH option as follows:

```
proc iclifetest data=BCS plots=survival(cl nodash) impute(seed=1234);
   strata trt;
   time (lTime, rTime);
run;
```

In Output 69.2.2, the survival estimates are plotted only for the time intervals for which estimates can be obtained.

Output 69.2.2 Nonparametric Survival Estimates

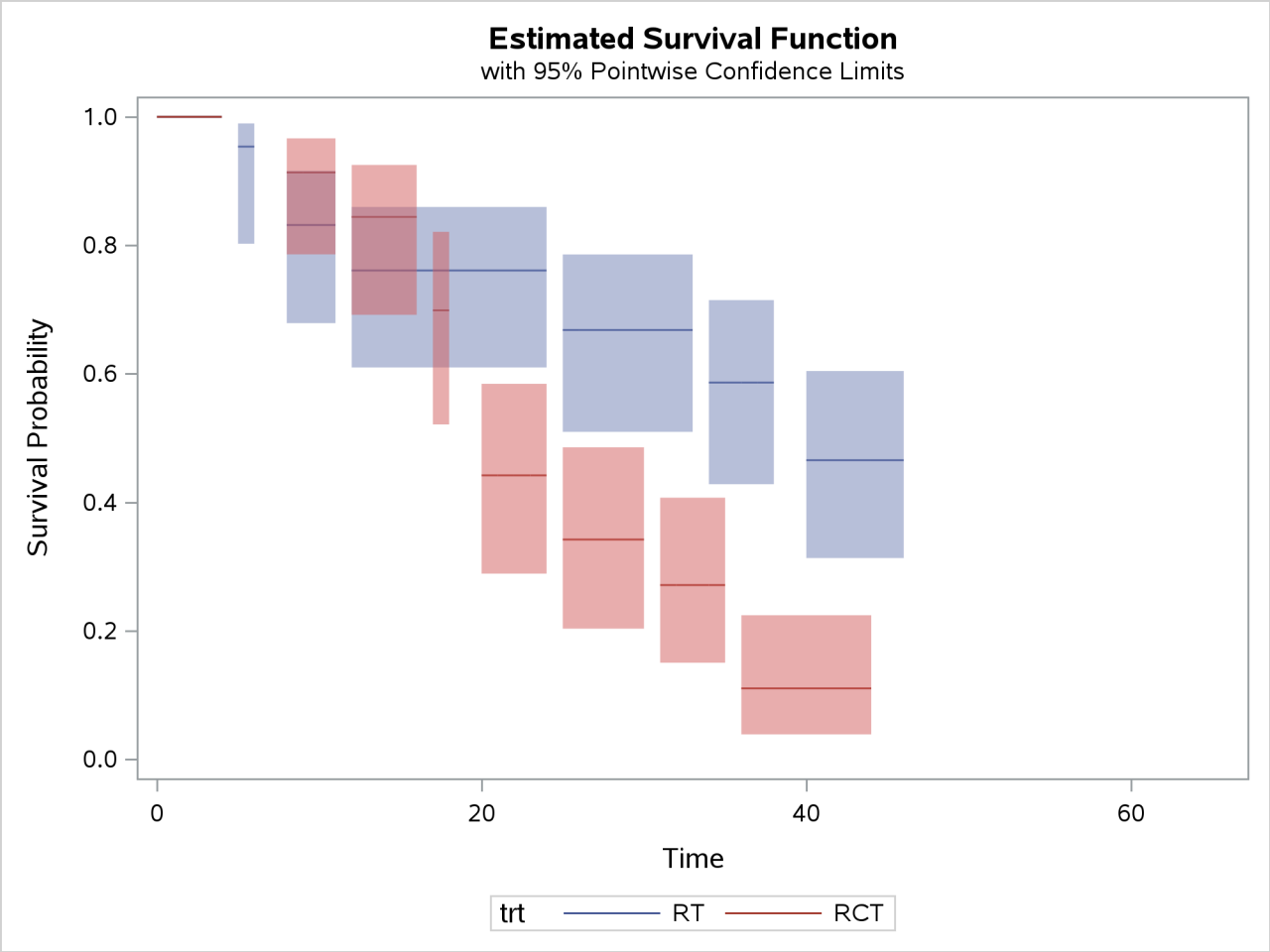

If the overlaid plot becomes too busy, then it might be necessary to plot the survival estimates separately. You can request a paneled display of individual survival estimate by specifying the STRATA=PANEL option as follows:

```
proc iclifetest data=BCS plots=survival(cl strata=panel) impute(seed=1234);
   strata trt;
   time (lTime, rTime);
run;
```

Now the output survival estimates for different treatments are plotted in different panels, as shown in Output 69.2.3.

Output 69.2.3 Paneled Survival Plot

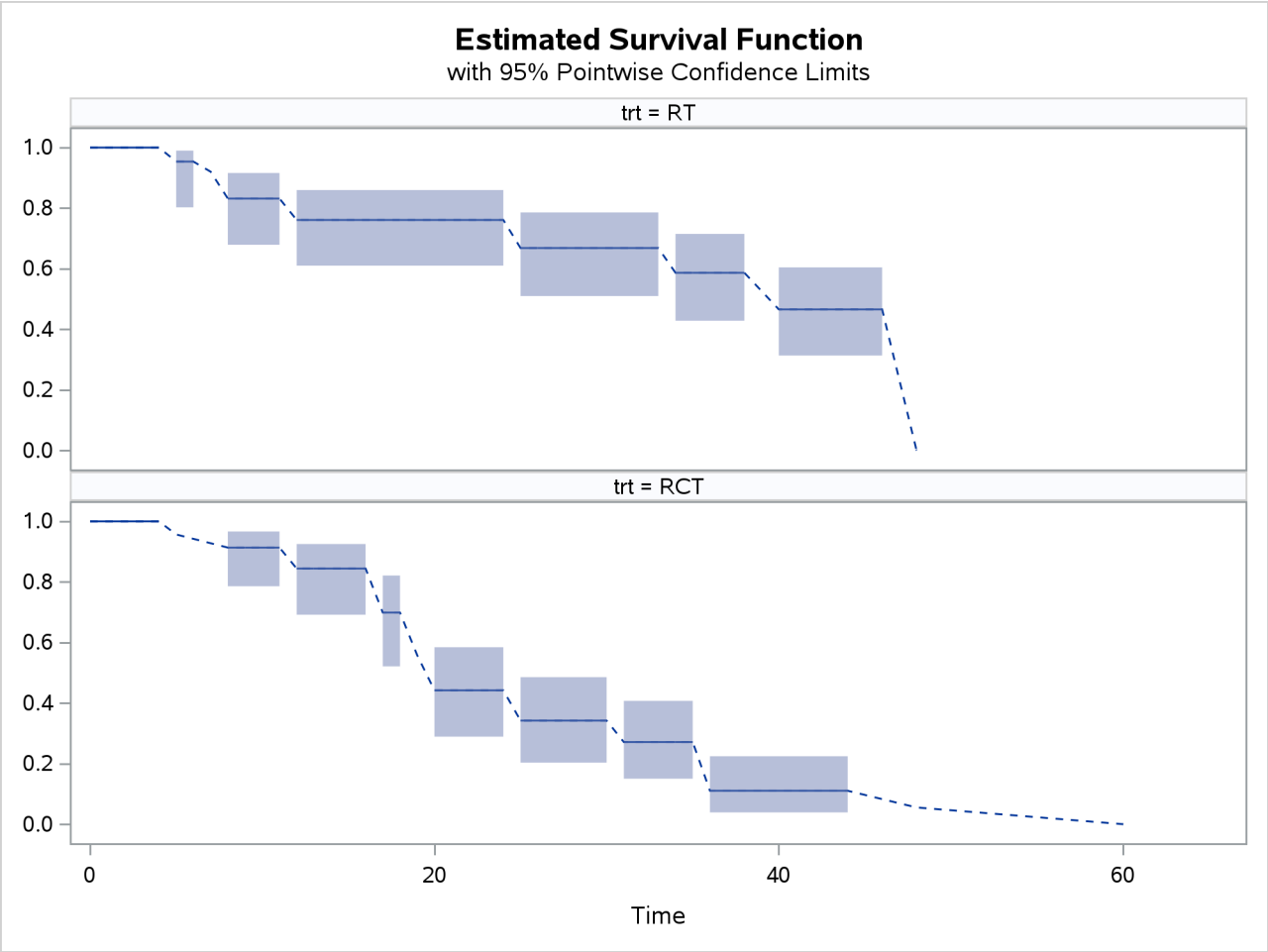

Example 69.3: Plotting Kernel-Smoothed Hazard Functions

(View the complete code for this example (iclftex3.sas) in the example repository.)

The following statements compute the kernel-smoothed hazard functions of the two treatment groups for the breast cancer data:

```
proc iclifetest data=BCS plots=hazard(bandwidth=15) impute(seed=1234);
   strata trt;
   time (lTime, rTime);
run;
```

A bandwidth of 15 is specified. By default, the ICLIFETEST procedure uses the Epanechnikov kernel.

Output 69.3.1 Kernel-Smoothed Hazards for Breast Cancer Data

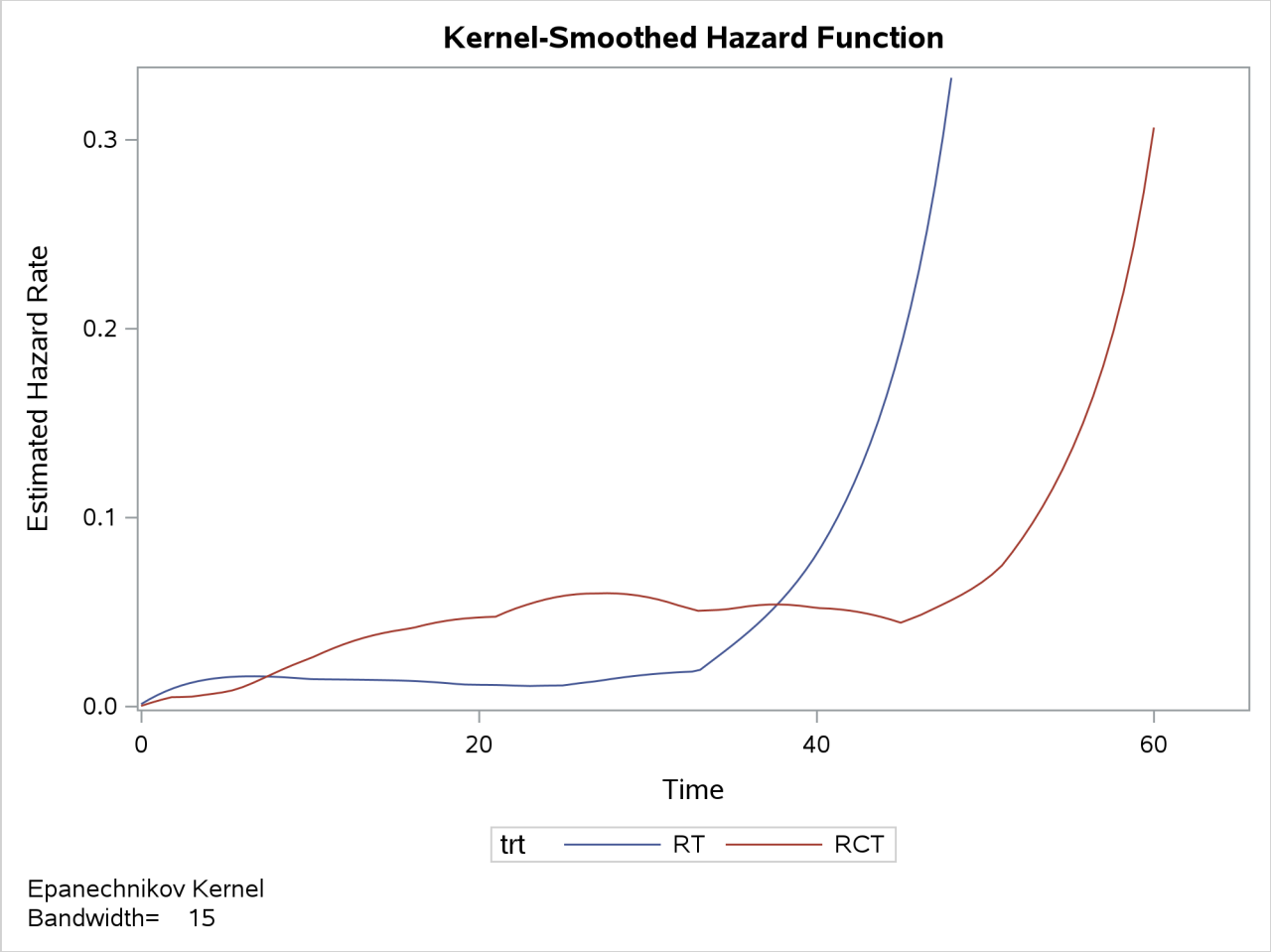

Example 69.4: Outputting Scores for Permutation Tests

(View the complete code for this example (iclftex4.sas) in the example repository.)

Recall the breast cancer data in the example in the section "Getting Started: ICLIFETEST Procedure" on page 5263. The following statements perform the generalized log-rank test to compare the survival distributions between two treatments by using Finkelstein's weights and save the corresponding scores to a SAS data set named Out:

```
proc iclifetest data=bcs impute(seed=123);
   time (ltime,rtime);
   test trt/weight=finkelstein outscore=out;
run;
```

Fay (1999) describes how to perform permutation tests by using these generated scores. PROC NPAR1WAY can do such tests and compute *p*-values based on normal theory approximation or Monte Carlo simulation. You need to specify the input data and specify SCORES=DATA, as follows:

```
proc npar1way data=out scores=data;
   class trt;
   var score;
   exact scores=data / mc seed=1234;
run;
```

The variable Trt is specified in the CLASS statement so that permutations are done for the groups formed by different levels of the variable. The MC option performs a Monte Carlo version of the permutation test and computes the *p*-values by using Monte Carlo samples. Output 69.4.1 shows the results from the test based on normal theory approximation.

Output 69.4.1 Asymptotic Permutation Test

The NPAR1WAY Procedure

Data Scores Two-Sample Test			
Statistic (S)	Z	Pr < Z	Pr > |Z|
-9.9442	-2.6839	0.0036	0.0073

Output 69.4.2 displays the *p*-values that are calculated from the Monte Carlo samples.

Output 69.4.2 Monte Carlo Permutation Test

Monte Carlo Estimates for the Exact Test					
Probability	Estimate	99% Confidence Limits		Samples	Seed
Pr <= S	0.0033	0.0018	0.0048	10000	1234
Pr >= |S - Mean|	0.0064	0.0043	0.0085		

Output 69.4.3 displays the results of the generalized log-rank statistics from using Finkelstein's weights. As you can see, it matches the two-sample test statistic of the permutation test.

Output 69.4.3 Generalized Log-Rank Statistics Using Finkelstein's Weights

The ICLIFETEST Procedure

Generalized Log-Rank Statistics	
trt	Log-Rank
RT	-9.94418
RCT	9.944182

References

Borgan, Ø., and Liestøl, K. (1990). "A Note on Confidence Interval and Bands for the Survival Curves Based on Transformations." *Scandinavian Journal of Statistics* 18:35–41.

Brookmeyer, R., and Crowley, J. (1982). "A Confidence Interval for the Median Survival Time." *Biometrics* 38:29–41.

Collett, D. (1994). *Modelling Survival Data in Medical Research*. London: Chapman & Hall.

Edwards, D., and Berry, J. J. (1987). "The Efficiency of Simulation-Based Multiple Comparisons." *Biometrics* 43:913–928.

Fay, M. P. (1996). "Rank Invariant Tests for Interval Censored Data under the Grouped Continuous Model." *Biometrics* 52:811–822.

Fay, M. P. (1999). "Comparing Several Score Tests for Interval Censored Data." *Statistics in Medicine* 18:273–285.

Finkelstein, D. M. (1986). "A Proportional Hazards Model for Interval-Censored Failure Time Data." *Biometrics* 42:845–854.

Fleming, T. R., and Harrington, D. P. (1981). "A Class of Hypothesis Tests for One and Two Samples of Censored Survival Data." *Communications in Statistics—Theory and Methods* 10:763–794.

Gasser, T., and Müller, H. G. (1979). "Kernel Estimation of Regression Functions." In *Smoothing Techniques for Curve Estimation: Proceedings of a Workshop Held in Heidelberg, April 2–4, 1979*, edited by T. Gasser and M. Rosenblatt, 23–68. Vol. 757 of Lecture Notes in Mathematics. Berlin: Springer-Verlag.

Gentleman, R., and Geyer, C. J. (1994). "Maximum Likelihood for Interval Censored Data: Consistency and Computation." *Biometrika* 81:618–623.

Gillespie, B. W., Chen, Q., Reichert, H., Franzblau, A., Hedgeman, E., Lepkowski, J., Adriaens, P., Demond, A., Luksemburg, W., and Garabrant, D. H. (2010). "Estimating Population Distributions When Some Data Are Below a Limit of Detection by Using a Reverse Kaplan-Meier Estimator." *Epidemiology* 21, Suppl. 4:S64–S70.

Goodall, R. L., Dunn, D. T., and Babiker, A. G. (2004). "Interval-Censored Survival Time Data: Confidence Intervals for the Non-parametric Survivor Function." *Statistics in Medicine* 23:1131–1145. http://dx.doi.org/10.1002/sim.1682.

Groeneboom, P., and Wellner, J. A. (1992). *Information Bounds and Nonparametric Maximum Likelihood Estimation*. Basel: Birkhäuser.

Harrington, D. P., and Fleming, T. R. (1982). "A Class of Rank Test Procedures for Censored Survival Data." *Biometrika* 69:133–143.

Hsu, J. C. (1992). "The Factor Analytic Approach to Simultaneous Inference in the General Linear Model." *Journal of Computational and Graphical Statistics* 1:151–168.

Huang, J., Lee, C., and Yu, Q. (2008). "A Generalized Log-Rank Test for Interval-Censored Failure Time Data via Multiple Imputation." *Statistics in Medicine* 27:3217–3226. http://dx.doi.org/10.1002/sim.3211.

Jongbloed, G. (1998). "The Iterative Convex Minorant Algorithm for Nonparametric Estimation." *Journal of Computational and Graphical Statistics* 7:310–321. https://www.jstor.org/stable/1390706.

Kalbfleisch, J. D., and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley & Sons.

Klein, J. P., and Moeschberger, M. L. (1997). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: Springer-Verlag.

Kramer, C. Y. (1956). "Extension of Multiple Range Tests to Group Means with Unequal Numbers of Replications." *Biometrics* 12:307–310.

Lachin, J. M. (2000). *Biostatistical Methods: The Assessment of Relative Risks*. New York: John Wiley & Sons.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Nair, V. N. (1984). "Confidence Bands for Survival Functions with Censored Data: A Comparative Study." *Technometrics* 26:265–275.

Ng, M. P. (2002). "A Modification of Peto's Nonparametric Estimation of Survival Curves for Interval-Censored Data." *Biometrics* 58:439–442.

Pan, W. (2000). "Smooth Estimation of the Survival Function for Interval Censored Data." *Statistics in Medicine* 19:2611–2624.

Peto, R. (1973). "Experimental Survival Curves for Interval-Censored Data." *Journal of the Royal Statistical Society, Series C* 22:86–91.

Sun, J. (1996). "A Nonparametric Test for Interval-Censored Failure Time Data with Application to AIDS Studies." *Statistics in Medicine* 15:1387–1395.

Sun, J. (2001a). "Nonparametric Test for Doubly Interval-Censored Failure Time Data." *Lifetime Data Analysis* 7:363–375.

Sun, J. (2001b). "Variance Estimation of a Survival Function for Interval-Censored Survival Data." *Statistics in Medicine* 20:1249–1257.

Turnbull, B. W. (1976). "The Empirical Distribution Function with Arbitrarily Grouped, Censored, and Truncated Data." *Journal of the Royal Statistical Society, Series B* 38:290–295.

Wellner, J. A., and Zhan, Y. (1997). "A Hybrid Algorithm for Computation of the Nonparametric Maximum Likelihood Estimator from Censored Data." *Journal of the American Statistical Association* 92:945–959.

Westfall, P. H., Tobias, R. D., Rom, D., Wolfinger, R. D., and Hochberg, Y. (1999). *Multiple Comparisons and Multiple Tests Using the SAS System*. Cary, NC: SAS Institute Inc.

Chapter 70
The ICPHREG Procedure

Contents

Overview: ICPHREG Procedure . . . **5312**
- Comparison with the PHREG Procedure . . . 5312

Getting Started: ICPHREG Procedure . . . **5313**

Syntax: ICPHREG Procedure . . . **5318**
- PROC ICPHREG Statement . . . 5319
- BASELINE Statement . . . 5326
- BY Statement . . . 5329
- CLASS Statement . . . 5330
- FREQ Statement . . . 5333
- HAZARDRATIO Statement . . . 5333
- ID Statement . . . 5335
- MODEL Statement . . . 5336
- OUTPUT Statement . . . 5339
- STRATA Statement . . . 5340
- TEST Statement . . . 5341

Details: ICPHREG Procedure . . . **5341**
- Model and Likelihood . . . 5341
- Baseline Parameterization . . . 5343
- Specification of Effects . . . 5344
- Computational Details . . . 5345
- Predicted Values . . . 5347
- Hazard Ratios . . . 5348
- Left-Truncation of Failure Times . . . 5351
- Residuals and Diagnostic Statistics . . . 5352
- EM Algorithm and Extensions . . . 5353
- Input and Output Data Sets . . . 5358
- Missing Values . . . 5358
- Displayed Output . . . 5358
- ODS Table Names . . . 5360
- ODS Graphics . . . 5361

Examples: ICPHREG Procedure . . . **5362**
- Example 70.1: Fitting Cubic Spline Models . . . 5362
- Example 70.2: Plotting Predicted Survival and Cumulative Hazard Functions . . . 5365
- Example 70.3: Fitting Stratified Weibull Models . . . 5368
- Example 70.4: Fitting Semiparametric Proportional Hazards Models . . . 5371
- Example 70.5: Fitting Semiparametric Models with Time-Dependent Covariates . . . 5377

References . . . **5379**

Overview: ICPHREG Procedure

The ICPHREG procedure is designed to fit proportional hazards regression models to interval-censored data. It can also fit proportional hazards regression models to failure time data that are uncensored, right censored, or left censored. The survival time of each member of a population is assumed to follow its own hazard function, $\lambda_i(t)$, which is expressed as

$$\lambda_i(t) = \lambda(t; \mathbf{Z}_i) = \lambda_0(t) \exp(\mathbf{Z}_i' \boldsymbol{\beta})$$

where $\lambda_0(t)$ is the baseline hazard function, $\mathbf{Z}_i$ is the vector of explanatory variables for the ith individual, and $\boldsymbol{\beta}$ is the vector of unknown regression coefficients that is associated with the explanatory variables. The vector $\boldsymbol{\beta}$ is assumed to be the same for all individuals.

The ICPHREG procedure enables you to use a variety of configurations with respect to the baseline function to fit a proportional hazards model; these configurations include a piecewise constant model (Friedman 1982), a cubic spline model (Royston and Parmar 2002), and a semiparametric model (Finkelstein 1986). To estimate the regression coefficients and the baseline parameters, the ICPHREG procedure maximizes the full likelihood instead of the Cox partial likelihood. Standard errors of the estimates are obtained by inverting the observed information matrix, which is derived from the full likelihood.

The ICPHREG procedure also enables you to do the following: include an offset variable in the model, weight the observations in the input data, test linear hypotheses about the regression coefficients, compute customized hazard ratios, and estimate and plot the survival function and the cumulative hazard function for a new set of covariates.

The ICPHREG procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Comparison with the PHREG Procedure

The ICPHREG procedure compares most closely to the PHREG procedure. Both procedures can fit proportional hazards models. They differ in the types of censored data that they are designed to handle and the forms of the baseline function. Table 70.1 matches the procedures with the types of censored data they can analyze and the form of the baseline function.

Table 70.1 Proportional Hazards Modeling

Data Type	Baseline	PROC ICPHREG	PROC PHREG
Interval-censored	Piecewise constant	Y	
	Cubic splines	Y	
	Unspecified	Y	
Right-censored	Piecewise constant	Y	Y
	Cubic splines	Y	
	Unspecified		Y

The PHREG procedure deals exclusively with right-censored data, and it mainly adopts a *semiparametric* approach by leaving the baseline hazard function unspecified. The ICPHREG procedure is specifically

designed to handle interval-censored data and offers different options to parameterize the baseline hazard function. You can use the ICPHREG procedure to analyze data that are left-censored, interval-censored, or right-censored. However, if the data to be analyzed contain only exact or right-censored observations, it is recommended that you use the PHREG procedure because it provides specialized methods for dealing with right-censored data. For more information about PROC PHREG, see Chapter 92, "The PHREG Procedure."

Both the LIFEREG procedure and the ICPHREG procedure can handle interval-censored data. The LIFEREG procedure focuses on parametric analysis that uses accelerated failure time models, and it can fit only a proportional hazards model that assumes a Weibull baseline hazard function.

Getting Started: ICPHREG Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/icphggs1.sas.)

This example demonstrates how you can fit a proportional hazards model on an interval-censored data set. By default, PROC ICPHREG uses a piecewise constant baseline hazard to fit the model.

The AIDS data (Larder, Darby, and Richman 1989) consist of observations from 31 patients who were followed up for the development of drug resistance to zidovudine. The following DATA step creates the SAS data set HIV:

```
data hiv;
   input Left Right Stage Dose CdLow CdHigh;
   if (Left=0) then Left=.;
   if (Right>=26) then Right=.;
   datalines;
0 16 0 0 0 1
15 26 0 0 0 1
12 26 0 0 0 1
17 26 0 0 0 1
13 26 0 0 0 1
0 24 0 0 1 0
6 26 0 1 1 0
0 15 0 1 1 0
14 26 0 1 1 0
12 26 0 1 1 0
13 26 0 1 0 1
12 26 0 1 1 0
12 26 0 1 1 0
0 18 0 1 0 1
0 14 0 1 0 1
0 17 0 1 1 0
0 15 0 1 1 0
3 26 1 0 0 1
4 26 1 0 0 1
1 11 1 0 0 1
13 19 1 0 0 1
0 6 1 0 0 1
0 11 1 1 0 0
6 26 1 1 0 0
```

```
0 6 1 1 0 0
2 12 1 1 0 0
1 17 1 1 1 0
0 14 1 1 0 0
0 25 1 1 0 1
2 11 1 1 0 0
0 14 1 1 0 0
;
```

The data set HIV contains the variables Left and Right, which are the starting time and ending time, both in months since the start of study; the variable Stage, which indicates the stage of disease (early (0) or late (1)); the variable Dose, a binary variable that indicates whether the dose is low (0) or high (1); the variable CdLow, which indicates whether the CD4 lymphocyte count is less than 100; and the variable CdHigh, which indicates that a count greater than or equal to 400 is recorded.

The following statements use PROC ICPHREG to fit a proportional hazards model to these data:

```
proc icphreg data=hiv;
   class Stage Dose / desc;
   model (Left, Right) = Stage Dose;
run;
```

The CLASS statement specifies that the variables Stage and Dose are classification variables. The DESC option sets the lower formatted value as the reference level for each CLASS variable. The MODEL statement specifies that the observed intervals are formed by Left and Right.

By default, the preceding statements produce information about the input data and the fitted model, as shown in Figure 70.1.

Figure 70.1 Model and Data Information from the ICPHREG Procedure

The ICPHREG Procedure

Model Information	
Data Set	WORK.HIV
Left Boundary	Left
Right Boundary	Right
Baseline Hazard	Piecewise Constant

Number of Observations Read	31
Number of Observations Used	31
Right Censored Observations	13
Interval Censored Observations	5
Left Censored Observations	13

Figure 70.1 shows 13 left-censored observations, 13 right-censored observations, and 5 interval-censored observations.

Figure 70.2 displays the “Class Level Information” table, which identifies the levels of the classification variables that are used in the model.

Figure 70.2 CLASS Variables Information from the ICPHREG Procedure

Class Level Information		
Name	**Levels**	**Values**
Stage	2	1 0
Dose	2	1 0

By default, PROC ICPHREG uses a baseline hazard that is partitioned into five disjoint intervals to fit a proportional hazards model. Figure 70.3 displays details about this partition.

Figure 70.3 Interval Partition

Constant Hazard Time Intervals		
Interval		
[Lower	**Upper)**	**Hazard Parameter**
0	5.5	Haz1
5.5	8	Haz2
8	12.5	Haz3
12.5	17	Haz4
17	Infty	Haz5

PROC ICPHREG determines the break points so that each time interval contains approximately an equal number of imputed middle points and boundary values in the input data set after excluding the right-censored observations. For more information about this method, see the section "Choosing Break Points" on page 5346. You can supply your own partition by using the INTERVALS= option in the MODEL statement.

The "Fit Statistics" table, shown in Figure 70.4, contains several statistics that summarize how well the model fits the data. These statistics are helpful in judging the adequacy of a model and in comparing it with other models under consideration.

Figure 70.4 Model Fit Statistics from the ICPHREG Procedure

Fit Statistics	
-2 Log Likelihood	21.813
AIC (Smaller is Better)	31.813
AICC (Smaller is Better)	34.213
BIC (Smaller is Better)	38.983

The table of parameter estimates is displayed in Figure 70.5. The columns display the parameter name, the degrees of freedom that are associated with the parameter, the estimated parameter value, the standard error of the parameter estimate, the confidence limits, the Wald chi-square statistic, and the associated p-value for testing the significance of the parameter. If a parameter has been fixed during the optimization process, or if a column of the Hessian matrix that corresponds to that parameter is found to linearly depend on columns that correspond to proceeding model parameters, PROC ICPHREG assigns zero degrees of freedom to that parameter and displays a value of zero for its standard error.

Figure 70.5 Model Parameter Estimates from the ICPHREG Procedure

Analysis of Maximum Likelihood Parameter Estimates									
Effect	Stage	Dose	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Haz1			0	0.0000					
Haz2			1	0.0167	0.0205	0.0000	0.0568		
Haz3			0	0.0000					
Haz4			1	0.0842	0.0655	0.0000	0.2126		
Haz5			1	2.5641	366.4263	0.0000	720.7464		
Stage	1		1	2.9597	0.9358	1.1255	4.7939	10.00	0.0016
Stage	0		0	0.0000					
Dose		1	1	1.6229	0.8410	-0.0255	3.2713	3.72	0.0537
Dose		0	0	0.0000					

Two types of parameters are present in Figure 70.5: the hazard parameters (Haz1, Haz2, . . . , Haz5) and the regression coefficients for the covariates. PROC ICPHREG does not display the chi-square statistic and associated *p*-value for the hazard parameters.

Two of the hazard parameters are constrained at 0, a sign of overparameterization that results from too many hazard parameters in the model. For more information about how the constraints are constructed, see the section "NOPOLISH" on page 5339. You can use fewer break points to fit the model by using the NINTERVAL= option or the INTERVALS= option. For example, the following statements request a model that has exactly two hazard parameters by specifying one break point at 10:

```
proc icphreg data=hiv ithistory;
   class Stage Dose / desc;
   model (Left, Right) = Stage Dose / basehaz=pch(intervals=(10));
run;
```

The table of parameter estimates is displayed in Figure 70.6. None of the hazard parameters are constrained.

Figure 70.6 Model Parameter Estimates from the ICPHREG Procedure

The ICPHREG Procedure

Analysis of Maximum Likelihood Parameter Estimates									
Effect	Stage	Dose	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Haz1			1	0.0042	0.0051	0.0000	0.0142		
Haz2			1	0.0590	0.0360	0.0000	0.1296		
Stage	1		1	2.0810	0.7298	0.6506	3.5114	8.13	0.0044
Stage	0		0	0.0000					
Dose		1	1	1.0907	0.6766	-0.2354	2.4167	2.60	0.1069
Dose		0	0	0.0000					

The ITHISTORY option outputs the iteration history of the fitting algorithm, which is shown in Figure 70.7. This option also produces the gradient and Hessian of the likelihood function at the last evaluation. In Figure 70.7, all values of the gradient are close to zero.

Figure 70.7 Iteration History from the ICPHREG Procedure

Likelihood Optimization Iteration History												
					Parameter Values				Gradient Values			
Iteration	Evaluations	-2 Log Likelihood	Change	Max Gradient	Stage1	Dose1	Haz1	Haz2	Stage1	Dose1	Haz1	Haz2
0	2	47.8895	.	74.9948	0	0	0.1245	0.0741	-1.8827	3.6394	74.9948	5.0972
1	10	39.1668	-8.7227	47.5072	0.1816	0.3012	0.0559	0.0848	-3.6580	0.5057	47.5072	-1.3723
2	6	37.6893	-1.4775	32.3796	0.2453	0.3347	0.0440	0.0944	-3.9790	-0.2883	32.3796	-1.7277
3	3	35.3576	-2.3317	50.9346	0.4701	0.4206	0.0185	0.1168	-4.6140	-2.0073	-50.9346	-3.9286
4	3	32.5990	-2.7586	6.4037	0.8039	0.5075	0.0200	0.1003	-2.8566	-0.7001	6.4037	0.3183
5	3	30.1026	-2.4964	94.9226	1.3800	0.7032	0.00785	0.0839	-2.1446	-1.3702	-94.9226	-6.0631
6	3	29.0224	-1.0802	23.8304	1.7623	0.8588	0.00724	0.0699	-0.3881	-0.2204	23.8304	-0.6369
7	3	28.8561	-0.1663	103.7	2.0115	1.0376	0.00384	0.0622	-0.5201	-0.4868	-103.7	-2.6791
8	3	28.7697	-0.0863	1.9794	2.0740	1.0858	0.00418	0.0593	-0.0194	-0.0145	-1.9794	-0.0585
9	3	28.7696	-0.00013	0.00309	2.0799	1.0898	0.00416	0.0590	-0.00170	-0.00121	0.00309	-0.00077
10	3	28.7696	-3.01E-6	0.000803	2.0809	1.0906	0.00416	0.0590	-0.00015	-0.00012	-0.00080	-0.00004
11	3	28.7696	-2.5E-8	6.991E-6	2.0810	1.0907	0.00416	0.0590	-6.99E-6	-5.67E-6	-6.93E-6	-1.38E-7
12	2	28.7696	0	6.991E-6	2.0810	1.0907	0.00416	0.0590	-6.99E-6	-5.67E-6	-6.93E-6	-1.38E-7

Last Evaluation of the Negative of the Gradient			
Haz1	Haz2	Stage1	Dose1
-6.93E-6	-1.38E-7	-6.99E-6	-5.67E-6

Last Evaluation of the Negative of the Hessian				
	Haz1	Haz2	Stage1	Dose1
Haz1	139133	4906.9	605.6	626.1
Haz2	4906.9	1967.0	46.4218	78.7404
Stage1	605.6	46.4218	5.2554	2.5003
Dose1	626.1	78.7404	2.5003	7.2471

One reason for fitting a proportional hazards model is to evaluate the hazard ratios between various disease groups. You can request customized hazard ratios by using the HAZARDRATIO statement, as follows:

```
proc icphreg data=hiv;
   class Stage / desc;
   model (Left, Right) = Stage / basehaz=pch(intervals=(10));
   hazardratio Stage;
run;
```

Figure 70.8 shows the estimated hazard ratio between the values 1 and 0 of the Stage variable and the corresponding 95% confidence limits.

Figure 70.8 Hazard Ratio Estimate between Stage Values 1 and 0

The ICPHREG Procedure

Hazard Ratios for Stage			
Description	Point Estimate	95% Wald Confidence Limits	
Stage 1 vs 0	5.624	1.734	18.241

The estimate of 5.624 indicates that patients who have Stage 1 disease tend to have a much higher risk of developing AIDS than those who have Stage 0. However, the confidence limits are wide because of the small sample size.

Syntax: ICPHREG Procedure

The following statements are available in the ICPHREG procedure:

```
PROC ICPHREG <options> ;
   BASELINE <OUT=SAS-data-set> <COVARIATES=SAS-data-set><TIMELIST=list>
            <keyword=name ... keyword=name> </ options> ;
   BY variables ;
   CLASS variable <(options)> ... <variable <(options)>> </ global-options> ;
   FREQ variable </ option> ;
   HAZARDRATIO <'label'> variable </ options> ;
   ID variable ;
   MODEL (t1, t2)= effects </ options> ;
   MODEL response <* censor (list)> = effects </ options> ;
   OUTPUT <OUT=SAS-data-set> <keyword=name ... keyword=name> ;
   TEST <model-effects> </ options> ;
   STRATA variable ;
```

The MODEL statement is required, and only one MODEL statement is allowed. If multiple MODEL statements are present, only the last one is used. You can specify main effects and interaction terms in the MODEL statement, as in the GLM procedure. The CLASS statement, if present, must precede the MODEL statement. The BASELINE, HAZARDRATIO, and OUTPUT statements, if present, must follow the MODEL statement. If multiple BASELINE, OUTPUT, and STRATA statements are present, only the last one is used.

The following sections describe the PROC ICPHREG statement and then describe the other statements in alphabetical order.

PROC ICPHREG Statement

PROC ICPHREG < *options* > **;**

The PROC ICPHREG statement invokes the ICPHREG procedure. Table 70.2 summarizes the *options* available in the PROC ICPHREG statement.

Table 70.2 PROC ICPHREG Statement Options

Option	Description
ALPHA=	Specifies the level for confidence limits
DATA=	Names the SAS data set to be analyzed
ITHISTORY	Displays the iteration history, final gradient, and second derivative matrix
NAMELEN=	Specifies the length of effect names
NLOPTIONS	Specifies optimization parameters for fitting the specified model
NOPRINT	Suppresses all displayed output
NOTHREADS	Requests a single-threaded mode for the computation
PLOTS=	Controls the plots that are produced through ODS Graphics
SINGULAR=	Specifies the singularity tolerance
THREADS=	Specifies the number of threads for the computation

You can specify the following *options* in the PROC ICPHREG statement.

ALPHA=*number*

specifies the α level for $100(1-\alpha)\%$ confidence limits. The *number* must be between 0 and 1; the default value is 0.05, which results in 95% intervals. This value is used as the default level for confidence limits that are computed by the BASELINE, HAZARDRATIO, and MODEL statements. You can override this default by specifying the ALPHA= option in these statements.

DATA=*SAS-data-set*

names the SAS data set that contains the data to be analyzed. If you omit this option, the procedure uses the most recently created SAS data set.

ITHISTORY

displays the iteration history for computing maximum likelihood estimates, the final evaluation of the gradient, and the final evaluation of the negative of the second derivative matrix (that is, the negative of the Hessian).

NAMELEN=*n*

specifies the maximum length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 200. By default, NAMELEN=20.

NLOPTIONS(*options***)**

specifies options for the nonlinear optimization methods that are used for fitting the specified model. You can specify the following *options*:

ABSCONV=*r*

ABSTOL=*r*

specifies an absolute function convergence criterion by which minimization stops when $f(\boldsymbol{\psi}^{(k)}) \leq r$, where $\boldsymbol{\psi}$ is the vector of parameters in the optimization and $f(\cdot)$ is the objective function. The default value of *r* is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

ABSFCONV=*r*

ABSFTOL=*r*

specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, termination requires a small change of the function value in successive iterations,

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex that has the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex. By default, ABSFCONV=0.

ABSGCONV=*r*

ABSGTOL=*r*

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small,

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the *j*th parameter. This criterion is not used by the NMSIMP technique. The default value is *r* = 1E–5.

FCONV=*r*

FTOL=*r*

specifies a relative function convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. The same formula is used for the NMSIMP technique, but $\boldsymbol{\psi}^{(k)}$ is defined as the vertex that has the lowest function value, and $\boldsymbol{\psi}^{(k-1)}$ is defined as the vertex that has the highest function value in the simplex. The default is $r = 10^{-\mathrm{FDIGITS}}$, where FDIGITS is by default $-\log_{10}\{\epsilon\}$ and ϵ is the machine precision.

GCONV=*r*

GTOL=*r*

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, termination requires the normalized predicted function reduction to be small,

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the CONGRA technique (in which a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \; |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the NMSIMP technique. The default value is r = 1E–8.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. The default values are as follows, depending on the optimization technique (which you can specify in the TECHNIQUE= option):

- TRUREG, NRRIDG, and NEWRAP: 125
- QUANEW and DBLDOG: 500
- CONGRA: 1000
- NMSIMP: 3000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed *n*.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on the optimization technique (which you can specify in the TECHNIQUE= option):

- TRUREG, NRRIDG, and NEWRAP: 50
- QUANEW and DBLDOG: 200
- CONGRA: 400
- NMSIMP: 1000

These default values also apply when *n* is specified as a missing value.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The time is checked only at the end of each iteration. Therefore, the actual run time might be longer than *r*. By default, CPU time is not limited.

MINITER=*n*

MINIT=*n*

specifies the minimum number of iterations. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations. By default, MINITER=0.

TECHNIQUE=*keyword*

TECH=*keyword*

specifies the optimization technique to obtain maximum likelihood estimates. You can choose from the following techniques:

CONGRA	performs a conjugate-gradient optimization.
DBLDOG	performs a version of double-dogleg optimization.
EM	fits the model by using the expectation-maximization (EM) algorithm.
EMICM	fits the model by using the EMICM algorithm, which is an extension of the EM algorithm that also uses the iterative convex minorant (ICM) algorithm.
NEWRAP	performs a Newton-Raphson optimization that combines a line-search algorithm with ridging.
NMSIMP	performs a Nelder-Mead simplex optimization.
NONE	performs no optimization.
NRRIDG	performs a Newton-Raphson optimization with ridging.
QUANEW	performs a dual quasi-Newton optimization.
TRUREG	performs a trust-region optimization.

For models other than the semiparametric model, the default is TECHNIQUE=NEWRAP. For the semiparametric model, only the EM and EMICM methods are supported and the default optimization method is EMICM.

For more information about these optimization methods, see the section "Choosing an Optimization Algorithm" on page 545 in Chapter 20, "Shared Concepts and Topics."

For more information about the EM and EMICM methods, see the section "EM Algorithm and Extensions" on page 5353.

NOPRINT

suppresses all displayed output. This option temporarily disables the Output Delivery System (ODS); For more information, see Chapter 23, "Using the Output Delivery System."

NOTHREADS

forces single-threaded execution of the analytic computations. This option overrides the SAS system option THREADS | NOTHREADS. Specifying this option is equivalent to specifying the THREADS=1 option.

PLOTS< (*global-plot-options*) > = *plot-request*

PLOTS< (*global-plot-options*) > = (*plot-request* < . . . < *plot-request* > >)

specifies plots to be created using ODS Graphics. You can request plots of survival functions and cumulative hazard functions. Also, many of the observation statistics in the output data set can be plotted using this option. You are not required to create an output data set in order to produce a plot. When you specify only one *plot-request*, you can omit the parentheses around it.

You can specify the following *global-plot-options*:

CL

displays the pointwise confidence limits for the plot.

OVERLAY < =*overlay-option* >

specifies how to overlay the functions that are plotted for the covariate sets. You can specify the following *overlay-options*:

BYGROUP

GROUP

overlays onto the same plot all functions that are plotted for the covariate sets and have the same GROUP= value in the COVARIATES= data set.

INDIVIDUAL

IND

displays a separate plot for each covariate set.

By default, OVERLAY=BYGROUP if the GROUP= option is specified in the BASELINE statement or if the COVARIATES= data set contains the _GROUP_ variable; otherwise, by default, OVERLAY=INDIVIDUAL.

UNPACK

displays multiple plots individually. The default is to display related multiple plots in a panel. The UNPACK option works for INTERVAL, RESDEV, and RESLAG plots only. See the section "OUTPUT Statement" on page 5339 for definitions of the statistics specified with these *plot-requests*.

TIMERANGE=(< *min* > < ,*max* >)

TIMERANGE=< *min* > < ,*max* >

RANGE=(< *min* > < ,*max* >)

RANGE=< *min* > < ,*max* >

specifies the range of values on the time axis to clip the display. The *min* and *max* values are the lower and upper bounds of the range. By default, *min* is 0 and *max* is the largest boundary value.

You can specify the following *plot-requests*:

CUMHAZ
plots the estimated cumulative hazard function for each set of covariates in the data set that you specify in the COVARIATES= option in the BASELINE statement. If the COVARIATES= data set is not specified, the estimated cumulative hazard function is plotted for the reference set of covariates, which consists of reference levels for the CLASS variables and average values for the continuous variables.

HAZARD
plots the estimated hazard function for each set of covariates in the data set that you specify in the COVARIATES= option in the BASELINE statement. If the COVARIATES= data set is not specified, the estimated hazard function is plotted for the reference set of covariates, which consists of reference levels for the CLASS variables and average values for the continuous variables.

INTERVAL
plots the observed interval length as a function of observation number.

NONE
suppresses all the plots in the procedure. Specifying this option is equivalent to disabling ODS Graphics for the entire procedure.

RESDEV< (*options*) >
plots deviance residuals. You can specify the following *options*:

INDEX
plots deviance residuals as a function of the observation number.

XBETA
plots deviance residuals as a function of the linear predictor.

If you do not specify an *option*, deviance residuals are plotted as a function of the observation number.

RESLAG< (*options*) >
plots Lagakos residuals. You can specify the following *options*:

INDEX
plots Lagakos residuals as a function of the observation number.

XBETA
plots Lagakos residuals as a function of the linear predictor.

If you do not specify an *option*, deviance residuals are plotted as a function of observation number.

SURVIVAL
S
SURV
SUR
plots the estimated survival function for each set of covariates in the data set that is specified in the COVARIATES= option in the BASELINE statement. If the COVARIATES= data set is not specified, the estimated survival function is plotted for the reference set of covariates, which

consists of reference levels for the CLASS variables and average values for the continuous variables.

Each observation in the data set that is specified in the COVARIATES= option in the BASELINE statement provides a set of covariates for which a plot is produced for each *plot-request*. You can use the ROWID= option in the BASELINE statement to specify a variable in the COVARIATES= data set for identifying the functions that are plotted for the covariate sets. If the ROWID= option is not specified, the plots are identified by the covariate values if there is only a single covariate or by the observation numbers of the COVARIATES= data set if the model has two or more covariates. If the COVARIATES= data set is not specified, a reference set of covariates that consists of the reference levels for the CLASS variables and the average values for the continuous variables is used. When plotting more than one function, you can use the OVERLAY= option to group the functions. When you specify only one *plot-request*, you can omit the parentheses around the plot request. Here are some examples:

```
plots=survival
plots=(survival cumhaz)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc icphreg plots(cl)=survival;
   model (Left, Right)=X1-X5;
   baseline covariates=One;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

SINGULAR=*number*

EPSILON=*number*

specifies the tolerance for testing the singularity of the $\mathbf{Z}'\mathbf{Z}$ matrix that is formed from the design matrix $\mathbf{Z}$ and for testing the singularity of the Hessian matrix upon convergence of the optimization algorithm. Appropriately, the test requires that a pivot be at least this number times the original diagonal value. By default, *number* is 10^7 times the machine epsilon. On most machines, the default *number* is approximately 10^{-9}.

THREADS=*n*

NTHREADS=*n*

specifies the number of threads for analytic computations and overrides the SAS system option THREADS | NOTHREADS. If you do not specify the THREADS= option or if you specify THREADS=0, the number of threads is determined based on the data size and the number of CPUs on the host on which the analytic computations execute.

BASELINE Statement

BASELINE <**OUT=***SAS-data-set*> <**COVARIATES=***SAS-data-set*> <**TIMELIST=***list*> <*keyword*=*name* . . . *keyword*=*name*> <*/ options*> **;**

The BASELINE statement creates a SAS data set (named by the OUT= option) that contains the predicted values at specified times that partition the time axis for every set of covariates in the COVARIATES= data set. If the COVARIATES= data set is not specified, PROC ICPHREG uses a reference set of covariates that consists of the reference levels for the CLASS variables and the average values for the continuous variables.

Table 70.3 summarizes the options that you can specify in the BASELINE statement.

Table 70.3 BASELINE Statement Options

Option	Description
Data Set and Time List Options	
OUT=	Specifies the output BASELINE data set
COVARIATES=	Specifies the SAS data set that contains the explanatory variables
TIMELIST=	Specifies a list of time points for computing the predicted values
Keyword Options for Variables	
CUMHAZ=	Specifies the cumulative hazard function estimate
HAZARD=	Specifies the hazard function estimate
LOGLOGS=	Specifies the log of the negative log of the survival function
LOGSURV=	Specifies the log of the survival function
LOWERCUMHAZ=	Specifies the lower pointwise confidence limit for the cumulative hazard function
LOWER=	Specifies the lower pointwise confidence limit for the survival function
STDCUMHAZ=	Specifies the estimated standard error of the cumulative hazard function
STDERR=	Specifies the standard error of the survival function
STDXBETA=	Specifies the estimated standard error of the linear predictor estimator
SURVIVAL=	Specifies the survival function estimate
UPPERCUMHAZ=	Specifies the upper pointwise confidence limit for the cumulative hazard function
UPPER=	Specifies the upper pointwise confidence limit for the survival function
XBETA=	Specifies the estimate of the linear predictor $\mathbf{z}'\boldsymbol{\beta}$
Other Options	
ALPHA=	Specifies the level of the confidence interval for the survival function
CLTYPE=	Specifies the transformation that is used to compute confidence limits for the survival function
GROUP=	Names a variable whose values identify or group predicted survival or cumulative hazard functions in plots
ROWID=	Names the variable in the COVARIATES= data set for identifying the predicted survival or cumulative hazard functions in plots

You can specify the following options in the BASELINE statement.

OUT=*SAS-data-set*
names the output data set. If you omit the OUT= option, the data set is created and given a default name by using the DATA*n* convention. For more information, see the section "OUT= Output Data Set in the BASELINE Statement" on page 5358.

COVARIATES=*SAS-data-set*
names the SAS data set that contains the sets of explanatory variable values for which the functions of interest are estimated. All variables in the COVARIATES= data set are copied to the OUT= data set. Thus, any variable in the COVARIATES= data set can be used to identify the covariate sets in the OUT= data set.

TIMELIST=*list*
specifies a list of time points at which the predicted values are computed. The following specifications are equivalent:

```
timelist=5,20 to 50 by 10
timelist=5 20 30 40 50
```

If you do not specify this option, predicted values are computed at all the times that partition the time axis.

*keyword***=***name*
specifies the statistics to be included in the OUT= data set and assigns names to the variables that contain these statistics. Specify a *keyword* for each desired statistic, an equal sign, and the name of the variable for the statistic. You can specify the following *keywords*:

CUMHAZ=*name*
specifies the cumulative hazard function estimate. Specifying CUMHAZ=_ALL_ is equivalent to specifying CUMHAZ=CumHaz, STDCUMHAZ=StdErrCumHaz, LOWERCUMHAZ=LowerCumHaz, and UPPERCUMHAZ=UpperCumHaz.

HAZARD=*name*
specifies the hazard function estimate.

LOGLOGS=*name*

CLOGLOGS=*name*
specifies the log of the negative log of the estimated survival function.

LOGSURV=*name*
specifies the log of the estimated survival function.

LOWER=*name*

L=*name*

LOWERSDF=*name*
specifies the lower pointwise confidence limit for the survival function. The confidence level is determined by the ALPHA= option.

LOWERCUMHAZ=*name*
specifies the lower pointwise confidence limit for the cumulative hazard function. The confidence level is determined by the ALPHA= option.

STDERR=*name*
STDSDF=*name*
specifies the standard error of the survival function estimator.

STDCUMHAZ=*name*
specifies the estimated standard error of the cumulative hazard function estimator.

STDXBETA=*name*
specifies the estimated standard error of the linear predictor estimator.

SURVIVAL=*name*
SDF=*name*
specifies the estimated survival function ($S(t) = [S_0(t)]^{\exp(\mathbf{z}'\boldsymbol{\beta})}$). Specifying SURVIVAL=_ALL_ is equivalent to specifying SURVIVAL=Survival, STDERR=StdErrSurvival, LOWER=LowerSurvival, and UPPER=UpperSurvival.

UPPER=*name*
U=*name*
UPPERSDF=*name*
specifies the upper pointwise confidence limit for the survival function. The confidence level is determined by the ALPHA= option.

UPPERCUMHAZ=*name*
specifies the upper pointwise confidence limit for the cumulative hazard function. The confidence level is determined by the ALPHA= option.

XBETA=*name*
specifies the estimate of the linear predictor $\mathbf{z}'\boldsymbol{\beta}$. If there is an offset, it is added to the predictor.

You can specify the following *options* after a slash (/).

ALPHA=*value*
specifies the level of the confidence interval for the survival function. The *value* must be between 0 and 1. The default is the value of the ALPHA= option in the PROC ICPHREG statement, or 0.05 if that option is not specified.

CLTYPE=*method*
CITYPE=*method*
TYPE=*method*
CLTRANSFORM=*method*
TRANSFORM=*method*
specifies the transformation that is used to compute the confidence limits for $S(t, \mathbf{z})$, which is the survival function for a subject that has a fixed covariate vector $\mathbf{z}$ at event time t. You can specify the following *methods*:

LOG
uses normal theory approximation to compute the confidence limits for $\log(S(t,\mathbf{z}))$. The confidence limits for $S(t,\mathbf{z})$ are obtained by back-transforming the confidence limits for $\log(S(t,\mathbf{z}))$.

LOGLOG
uses normal theory approximation to compute the confidence limits for the $\log(-\log(S(t,\mathbf{z})))$. The confidence limits for $S(t,\mathbf{z})$ are obtained by back-transforming the confidence limits for $\log(-\log(S(t,\mathbf{z})))$.

NORMAL
IDENTITY
LINEAR
PLAIN
DIRECT
uses normal theory approximation to compute the confidence limits for $S(t,\mathbf{z})$.

By default, CLTYPE=LOG.

GROUP=*variable*
names a variable whose values identify or group the predicted curves. The *variable* must be a numeric variable in the COVARIATES= data set. Survival curves for observations that have the same value of the *variable* are overlaid in the same plot.

ROWID=*variable*
ID=*variable*
ROW=*variable*
names a variable in the COVARIATES= data set for identifying plotted survival functions and cumulative hazard functions. This option has no effect if the PLOTS= option in the PROC ICPHREG statement is not specified. Values of this variable are used to label the plotted functions for the corresponding rows in the COVARIATES= data set. You can specify ROWID=_OBS_ to use the observation numbers in the COVARIATES= data set for identification.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC ICPHREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the ICPHREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.

- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *global-options* > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. Response variables do not need to be specified in the CLASS statement.

The CLASS statement must precede the MODEL statement. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*. You can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING

DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC ICPHREG orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (., ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values of the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so this option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC ICPHREG interprets values of the ORDER= option:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. You can specify any of the *keywords* shown in the following table. The default is PARAM=GLM. Design matrix columns are created from CLASS variables according to the corresponding coding schemes.

Value of PARAM=	Coding
EFFECT	Effect coding
GLM	Less-than-full-rank reference cell coding (this *keyword* can be used only in a global option)
ORDINAL **THERMOMETER**	Cumulative parameterization for an ordinal CLASS variable
POLYNOMIAL **POLY**	Polynomial coding
REFERENCE **REF**	Reference cell coding
ORTHEFFECT	Orthogonalizes PARAM=EFFECT coding
ORTHORDINAL **ORTHOTHERM**	Orthogonalizes PARAM=ORDINAL coding
ORTHPOLY	Orthogonalizes PARAM=POLYNOMIAL coding
ORTHREF	Orthogonalizes PARAM=REFERENCE coding

All parameterizations are full rank, except for the GLM parameterization. The REF= option in the CLASS statement determines the reference level for EFFECT and REFERENCE coding and for their orthogonal parameterizations. It also indirectly determines the reference level for a singular GLM parameterization through the order of levels.

If a PARAM= option is specified as a variable option for some variables, then any variables for which PARAM= is not specified use either the EFFECT parameterization if the global PARAM= option is not specified, or the full-rank parameterization indicated in the global PARAM= option if specified. If

the global PARAM=GLM option is specified and PARAM= is also specified for some variables, GLM parameterization is used for all variables.

If PARAM=ORTHPOLY or PARAM=POLY and the classification variable is numeric, then the ORDER= option in the CLASS statement is ignored, and the internal unformatted values are used. For more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

REF=*'level' | keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*:

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n*>

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

Class Variable Default Parameterization

If the PARAM= option is not specified together with any individual CLASS variable, then by default, PARAM=GLM. Otherwise, the default is PARAM=EFFECT.

Class Variable Naming Convention

Parameter names for a CLASS predictor variable are constructed by concatenating the CLASS variable name with the CLASS levels. However, for the POLYNOMIAL and orthogonal parameterizations, parameter names are formed by concatenating the CLASS variable name and keywords that reflect the parameterization. For examples and more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

Class Variable Parameterization with Unbalanced Designs

PROC ICPHREG initially parameterizes the CLASS variables by looking at the levels of the variables across the complete data set. If you have an *unbalanced* replication of levels across variables or BY groups, then the design matrix and the parameter interpretation might be different from what you expect. For example, suppose you have a model that has one CLASS variable A with three levels (1, 2, and 3) and another CLASS variable B with two levels (1 and 2). If the third level of A occurs only with the first level of B, if you use the EFFECT parameterization, and if your model contains the effect A(B) and an intercept, then the design for A within the second level of B is not a differential effect. In particular, the design looks like the following:

		Design Matrix			
		A(B=1)		A(B=2)	
B	A	A1	A2	A1	A2
1	1	1	0	0	0
1	2	0	1	0	0
1	3	–1	–1	0	0
2	1	0	0	1	0
2	2	0	0	0	1

PROC ICPHREG detects linear dependency among the last two design variables and sets the parameter for A2(B=2) to zero, resulting in an interpretation of these parameters as if they were reference- or dummy-coded. The REFERENCE or GLM parameterization might be more appropriate for such problems.

FREQ Statement

FREQ *variable* < / *option* > **;**

The FREQ statement identifies the *variable* (in the input data set) that contains the frequency of occurrence of each observation. PROC ICPHREG treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If the frequency value is missing, the observation is not used in the estimation of the regression coefficients.

You can specify the following *option* after a slash (/):

NOTRUNCATE
NOTRUNC
specifies that frequency values are not truncated to integers.

HAZARDRATIO Statement

HAZARDRATIO < '*label*' > *variable* < / *options* > **;**

The HAZARDRATIO statement enables you to request hazard ratios for any variable in the model at customized settings. For example, if the model contains the interaction of a CLASS variable A and a continuous variable X, the following specification displays a table of hazard ratios that compares the hazards of each pair of levels of A at X = 3:

```
hazardratio A / at (X=3);
```

The HAZARDRATIO statement identifies the variable whose hazard ratios are to be evaluated. If the variable is a continuous variable, the hazard ratio compares the hazards for a particular change (by default, an increase of 1 unit) in the variable. For a CLASS variable, a hazard ratio compares the hazards of two levels of the variable. You can specify more than one HAZARDRATIO statement, and you can provide an optional label (specified as a quoted string) to identify the output.

Table 70.4 summarizes the *options* that you can specify in the HAZARDRATIO statement.

Table 70.4 HAZARDRATIO Statement Options

Option	Description
ALPHA=	Specifies the alpha level
AT	Specifies the variables that interact with the variable of interest
DIFF=	Specifies which differences to consider
E	Displays the log-hazard ratio
UNITS=	Specifies the units of change for a continuous variable of interest

You can specify the following *options* after a slash (/).

ALPHA=*number*

specifies the level of confidence intervals for the hazard ratios. The *number* must be between 0 and 1. The default is the value of the ALPHA= option in the PROC ICPHREG statement, or 0.05 if that option is not specified.

AT (*variable***=ALL | REF |** *list* **< . . .** *variable***=ALL | REF |** *list* **>)**

specifies the variables that interact with the variable of interest and the corresponding values of the interacting variables. If the interacting variable is continuous and you specify a numeric *list* after the equal sign, hazard ratios are computed for each value in the list. If the interacting variable is a CLASS variable, you can specify, after the equal sign, a *list* of quoted strings that correspond to various levels of the CLASS variable, or you can specify the keyword ALL or REF. Hazard ratios are computed at each value of the list if you specify a *list*, at each level of the interacting variable if you specify ALL, or at the reference level of the interacting variable if you specify REF.

If you do not specify the AT option, PROC ICPHREG finds all the variables that interact with the variable of interest. If an interacting variable is a CLASS variable, *variable*=ALL is the default; if the interacting variable is continuous, *variable*=*m* is the default, where *m* is the average of all the sampled values of the continuous variable.

Suppose the model contains two interactions: an interaction A*B of CLASS variables A and B, and another interaction A*X of A with a continuous variable X. If 3.5 is the average of the sampled values of X, the following two HAZARDRATIO statements are equivalent:

```
hazardratio A;
hazardratio A / at (B=ALL X=3.5);
```

DIFF=*diff-request*

specifies which differences to consider for the level comparisons of a CLASS variable. This option is ignored in the estimation of hazard ratios for a continuous variable. You can specify the following *diff-requests*:

DISTINCT

DISTINCTPAIRS

ALL

requests all comparisons of only the distinct combinations of pairs.

PAIRWISE

PERM

PERMUTATIONS

requests all possible pairwise comparisons of levels.

REF

REFERENCE

requests comparisons between the reference level and all other levels of the CLASS variable.

For example, let A be a CLASS variable that has three levels (A1, A2, and A3), and suppose A3 is specified as the reference level. The following table shows the hazard ratios that are displayed for the three alternatives of the DIFF= option:

	Hazard Ratios Displayed					
DIFF=*option*	**A1 vs A2**	**A2 vs A1**	**A1 vs A3**	**A3 vs A1**	**A2 vs A3**	**A3 vs A2**
DISTINCT	√		√		√	
PAIRWISE	√	√	√	√	√	√
REF			√		√	

By default, DIFF=DISTINCT.

E

displays the vector $\mathbf{h}$ of linear coefficients such that $\mathbf{h}'\boldsymbol{\beta}$ is the log-hazard ratio, where $\boldsymbol{\beta}$ is the vector of regression coefficients.

UNITS=*value*

specifies the units of change in the continuous explanatory variable for which the customized hazard ratio is estimated. By default, UNITS=1. This option is ignored in the computation of the hazard ratios for a CLASS variable.

ID Statement

ID *variable* **;**

The ID statement specifies additional variables for identifying observations in the input data. When you specify BASE=UNSPECIFIED, this statement can be used to identify the subjects.

MODEL Statement

MODEL (*t1*, *t2*)= *effects* < / *options* > ;

MODEL *response* < ∗ *censor* (*list*) > = *effects* < / *options* > ;

The MODEL statement identifies the variables to be used as the failure time variables and the explanatory effects, including covariates, main effects, interactions, nested effects. For more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

You can specify two forms of MODEL syntax: the first form allows two time variables, and the second form allows one time variable.

The first form of the MODEL statement enables you to analyze time-to-event data that have interval-censored outcomes. The MODEL syntax specifies two variables, *t1* and *t2*, that contain values of the endpoints of the censoring interval. Only nonnegative values are accepted. If the two values are the same (and not missing), it is assumed that there is no censoring and the actual response value is observed. If the lower value is missing, then the upper value is used as a left-censored value. If the upper value is missing, then the lower value is used as a right-censored value. If both values are present and the lower value is less than the upper value, it is assumed that the values specify a censoring interval. If the lower value is greater than the upper value or both values are missing, then the observation is not used in the analysis.

The following table summarizes the ways of specifying censoring.

Lower Value	Upper Value	Comparison	Interpretation
Not missing	Not missing	Equal	No censoring
Not missing	Not missing	Lower < upper	Censoring interval
Missing	Not missing		Upper used as left-censoring value
Not missing	Missing		Lower used as right-censoring value
Not missing	Not missing	Lower > upper	Observation not used
Missing	Missing		Observation not used

The second form of the MODEL statement enables you to analyze right-censored data or time-to-event data that contain repeated assessments and possibly time-dependent covariates (for more information, see the section "Semiparametric Model and Time-Dependent Covariates" on page 5355). The name of the failure time variable precedes the equal sign. This name can optionally be followed by an asterisk, the name of the censoring variable, and a list of censoring values (separated by blanks or commas) enclosed in parentheses. If the censoring variable takes one of these values, the corresponding failure time is considered to be censored. Following the equal sign are the explanatory effects (sometimes called independent variables or covariates) for the model.

Table 70.5 summarizes the *options* that you can specify in the MODEL statement.

Table 70.5 MODEL Statement Options

Option	Description
Model Specification Options	
ALPHA=	Specifies the confidence level
BASE=	Specifies the functional form for the baseline function
ENTRY=	Specifies the left-truncation time of the model
HAZSCALE=	Requests parameterization of the hazard function in the original scale or in log scale
NOPOLISH	Suppresses polishing of parameter estimates of the hazard function
OFFSET=	Specifies an offset variable to be added to the linear predictor
PLVARIANCE	Computes the standard error estimates on the basis of the profile likelihood function
Output Options	
CORRB	Displays the estimated correlation matrix
COVB	Displays the estimated covariance matrix

ALPHA=*value*

specifies the level for the confidence intervals for parameters. The *value* must be between 0 and 1. By default, ALPHA=0.05.

CORRB

displays the estimated correlation matrix of the parameter estimates.

COVB

displays the estimated covariance matrix of the parameter estimates.

BASE=*baseline-type*

BASEHAZ=*baseline-type*

B=*baseline-type*

specifies a functional form for the baseline function. You can specify one of the following *baseline-types*:

PCH (< NINTERVAL=*number* **>, < INTERVALS=(***numeric-list***) >)**

PIECEWISE (< NINTERVAL=*number* **>, < INTERVALS=(***numeric-list***) >)**

PIECEWISEEXPONENTIAL (< NINTERVAL=*number* **>, < INTERVALS=(***numeric-list***) >)**

PCBH (< NINTERVAL=*number* **>, < INTERVALS=(***numeric-list***) >)**

partitions the time scale into disjoint intervals and assumes the baseline hazard function is piecewise constant within intervals. The parameters are the piecewise constant values of the baseline hazard functions and are named Haz1, Haz2, . . ., and so on. If HAZARDSCALE=LOGHAZ is specified, the names are LogHaz1, LogHaz2, . . ., and so on.

You can specify one of the following two options to control how to partition the time axis into intervals of constant baseline hazards:

NINTERVAL=*number*

N=*number*

specifies the number of intervals that have a constant hazard rate in each interval. PROC ICPHREG partitions the time axis into the *number* of intervals so that each interval contains an approximately equal number of unique boundary values and imputed middle points.

INTERVALS=(*numeric-list***)**

INTERVAL=(*numeric-list***)**

specifies a list of numbers that partition the time axis into disjoint intervals that have constant hazard rate in each interval. For example, INTERVALS=(100, 150, 200, 250, 300) specifies a model that has a constant hazard in the intervals [0,100), [100,150), [150,200), [200,250), [250,300), and [300,∞).

If you specify neither NINTERVAL= nor INTERVAL=, NINTERVAL=5 by default.

SPLINES (< DF=*number* **>)**

CUBICSPLINES (< DF=*number* **>)**

models the baseline cumulative hazard function by cubic splines (Royston and Parmar 2002). The parameters are the spline coefficients and are named Coef1, Coef2, . . ., and so on.

You can specify the degrees of freedom in the DF=*number* option, where *number* must be an integer. The number of knots equals *number* plus one. The actual positions of the knots are determined from an imputed data set as follows. First, PROC ICPHREG imputes a middle point for each observation in the input data set that is not right-censored. Then, it sorts these imputed times and the input boundary values in increasing order and selects only unique values. PROC ICPHREG places the *terminal knots* at the minimum and maximum of this sequence and chooses the *interval knots* by using the same method it uses to choose the break points for the piecewise constant model. For more information, see the section "Choosing Break Points" on page 5346.

By default, DF=2.

UNSPECIFIED

DISCRETE

models the cumulative hazard function as a discrete function in which jumps are identified according to Turnbull's formulation (1976). The parameters are named Eta1, Eta2, and so on.

The default fitting method for this type of model is EMICM. An alternative is the EM algorithm. For more information about these algorithms, see the section "EM Algorithm and Extensions" on page 5353.

If you do not specify the BASEHAZ= option, the ICPHREG procedure fits a piecewise constant model as if NINTERVAL=5.

ENTRYTIME=*variable*

ENTRY=*variable*

specifies the name of the variable that represents the left-truncation time. For more information, see the section "Left-Truncation of Failure Times" on page 5351.

NOPOLISH

suppresses polishing of parameter estimates of the baseline function. Occasionally, the parameter estimates of the baseline function can reach the default optimization lower bounds. This might indicate that the model is overparameterized. By default, the ICPHREG procedure "polishes" the hazard estimates by fixing these parameters at the lower bound value and refitting the model.

The lower bound values are set to 0 if the baseline parameters are on the original scale (HAZSCALE=HAZARD). The values are set to –10.0 if they are on the log scale (HAZSCALE=LOGHAZ).

This option does not apply to the cubic spline model because its baseline parameters are unbounded.

OFFSET=*variable*

specifies a variable in the input data set to be used as an offset variable. This variable cannot be a CLASS variable, the response variable, or any of the explanatory variables.

HAZSCALE=*hazard-type*

specifies a transformation to be applied to the baseline parameters for fitting the piecewise constant model. You can choose either of the following two options:

LOGHAZ

LOG

LOGHAZARD

uses the log transformed baseline parameters.

HAZARD

HAZ

does not transform the baseline parameters. A lower bound of 0 is used for fitting the models.

This option does not apply to the cubic spline model and the semiparametric model.

PLVARIANCE

computes the standard error estimates on the basis of the profile likelihood function, as opposed to the default Louis's method (Louis 1982). For more information, see the section "Variance Estimation" on page 5356. This option applies only to the semiparametric model.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *keyword*=*name* . . . *keyword*=*name* > **;**

The OUTPUT statement creates a new SAS data set that contains all the variables in the input data set and, optionally, contains the estimated linear predictors (XBETA) and their standard error estimates, residuals, and diagnostic statistics.

The estimated linear predictor and its standard error estimate are computed for all observations in which the explanatory variables are all nonmissing, even if the response is missing. By adding to the input data set observations for which both lower and upper values are missing, you can compute these statistics for new observations for settings of the explanatory variables not present in the data without affecting the model fit.

You can specify the following options:

OUT=*SAS-data-set*

specifies the output data set. If you omit the OUT=option, PROC ICPHREG creates the output data set and gives it a default name that uses the DATA*n* convention.

keyword=name

specifies the statistics to be included in the output data set and names the new variables that contain the statistics. Specify a keyword for each statistic that you want (see the following list of keywords), an equal sign, and the name of the new variable or variables to contain the statistic. You can list only one variable after the equal sign for all the statistics.

Although you can use the OUTPUT statement without specifying a *keyword=name*, if you do so, the output data set then contains only the original variables. Formulas for the statistics are given in the section "Residuals and Diagnostic Statistics" on page 5352.

You can specify the following *keywords*:

INTERVAL represents the interval length for the input observation. The value is missing if the observation is right-censored or missing.

RESDEV represents the deviance residual for identifying poorly fitted observations.

RESLAG represents the Lagakos residual for identifying poorly fitted observations.

STDXBETA represents the standard error estimate of XBETA (see the XBETA keyword).

XBETA represents the estimate of the linear predictor $\mathbf{x}_i'\boldsymbol{\beta}$ for observation i. If there is an offset, it is included in $\mathbf{x}_i'\boldsymbol{\beta}$.

The OUTPUT statement is not available when you specify the STRATA statement or the ENTRY= option in the MODEL statement.

STRATA Statement

STRATA *variable* < / *option* > **;**

The proportional hazards assumption might not be realistic for all data. However, it might still be reasonable to perform a stratified analysis. The STRATA statement names one variable that determines the stratification. Strata are formed according to the nonmissing values of the STRATA variables unless you specify the MISSING option. In the STRATA statement, *variable* is a variable whose values are used to determine the strata levels.

You can specify the following *option*:

MISSING

allows missing values ('.' for numeric variables and blanks for character variables) as valid STRATA variable values. Otherwise, observations that have missing STRATA variable values are deleted from the analysis.

TEST Statement

TEST < *model-effects* > < / *options* > **;**

The TEST statement enables you to perform Wald tests for model effects that test Type I, Type II, or Type III hypotheses. For more information about constructing of Type I, II, and III estimable functions, see Chapter 16, "The Four Types of Estimable Functions."

Table 70.6 summarizes the *options* that you can specify in the TEST statement.

Table 70.6 TEST Statement Options

Option	Description
E	Requests Type I, Type II, and Type III coefficients
E1	Requests Type I coefficients
E2	Requests Type II coefficients
E3	Requests Type III coefficients
HTYPE=	Indicates the type of hypothesis test to perform

For information about the syntax of the TEST statement, see the section "TEST Statement" on page 556 in Chapter 20, "Shared Concepts and Topics."

Details: ICPHREG Procedure

Model and Likelihood

Suppose that the observations to be analyzed consist of interval-censored outcomes $\{[L_i, R_i]; \mathbf{Z}_i\}$, $i = 1, \ldots, n$, where n is the number of subjects. $\mathbf{Z}_i$ denotes a p-dimensional vector of covariates for the ith subject. This notation allows for exact event times, right-censored data and left-censored data as special cases. When $L_i = R_i$, the observation is an exact time; when $R_i = \infty$, the observation is right-censored; when $L_i = 0$, the observation is left-censored.

Let $S(t; \mathbf{Z}_i)$ denote the survival function for a subject whose covariate is $\mathbf{Z}_i$. Assuming that t is continuous, denote $f(t; \mathbf{Z}_i)$ as the density function for the subject. The hazard function for the subject, $\lambda(t; \mathbf{Z}_i)$, is defined as the instantaneous failure rate at time t. Mathematically, the hazard function is determined as a ratio between the density function and the survival function:

$$\lambda(t; \mathbf{Z}_i) = f(t; \mathbf{Z}_i)/S(t; \mathbf{Z}_i)$$

A quantity that is closely related to the survival function is the cumulative hazard function, defined as

$$\Lambda(t; \mathbf{Z}_i) = \int_0^t \lambda(u; \mathbf{Z}_i)\, du$$

In turn, the cumulative hazard function determines the survival function:

$$S(t; \mathbf{Z}_i) = \exp(-\Lambda(t; \mathbf{Z}_i))$$

If some of the responses are left-, right-, or interval-censored, the log likelihood can be written as

$$\begin{aligned}\log(\mathrm{L}) &= \sum \log\left[f(L_i;\mathbf{Z}_i)\right] + \sum \log\left[S(L_i;\mathbf{Z}_i)\right] \\ &+ \sum \log\left[1 - S(R_i;\mathbf{Z}_i)\right] + \sum \log\left[S(L_i;\mathbf{Z}_i) - S(R_i;\mathbf{Z}_i)\right]\end{aligned}$$

where the first sum is the total of the uncensored observations, the second sum is the total of the right-censored observations, the third sum is the total of the left-censored observations, and the last sum is the total of the interval-censored observations.

For the ith subject, the proportional hazards model (Cox 1972) assumes that

$$\lambda(t;\mathbf{Z}_i) = \lambda_0(t)\exp(\mathbf{Z}_i'\boldsymbol{\beta})$$

where $\boldsymbol{\beta}$ is a p-dimensional vector of coefficients for the covariate vector $\mathbf{Z}_i$ and $\lambda_0(t)$ is the baseline hazard function, which is the hazard rate when all the coefficients for the covariates are equal to 0.

Under the proportional hazards model, the cumulative hazard function for the ith subject is

$$\Lambda(t;\mathbf{Z}_i) = \int_0^t \lambda(u;\mathbf{Z}_i)\,du = \int_0^t \lambda_0(u)\,du\,\exp(\mathbf{Z}_i'\boldsymbol{\beta}) = \Lambda_0(t)\exp(\mathbf{Z}_i'\boldsymbol{\beta})$$

The survival function for the ith subject is

$$S(t;\mathbf{Z}_i) = \exp[-\Lambda(t;\mathbf{Z}_i)] = S_0(t)^{\exp(\mathbf{Z}_i'\boldsymbol{\beta})}$$

where $S_0(t)$ denotes the baseline survival function and $S_0(t) = \exp[-\Lambda_0(t)]$.

The density function for the subject is obtained by differentiating the survival function:

$$f(t;\mathbf{Z}_i) = -\frac{dS(t;\mathbf{Z}_i)}{dt} = \lambda(t;\mathbf{Z}_i)S(t;\mathbf{Z}_i) = \lambda_0(t)\exp(\mathbf{Z}_i'\boldsymbol{\beta})S_0(t)^{\exp(\mathbf{Z}_i'\boldsymbol{\beta})}$$

Given these quantities, the likelihood function under the proportional hazards model can be expressed as

$$\begin{aligned}\log(\mathrm{L}) &= \sum \log\left[\lambda_0(L_i)\exp(\mathbf{Z}_i'\boldsymbol{\beta})S_0(L_i)^{\exp(\mathbf{Z}_i'\boldsymbol{\beta})}\right] + \sum \log\left[S_0(L_i)^{\exp(\mathbf{Z}_i'\boldsymbol{\beta})}\right] \\ &+ \sum \log\left[1 - S_0(R_i)^{\exp(\mathbf{Z}_i'\boldsymbol{\beta})}\right] + \sum \log\left[S_0(L_i)^{\exp(\mathbf{Z}_i'\boldsymbol{\beta})} - S_0(R_i)^{\exp(\mathbf{Z}_i'\boldsymbol{\beta})}\right]\end{aligned}$$

where the first sum is the total of the uncensored observations, the second sum is the total of the right-censored observations, the third sum is the total of the left-censored observations, and the last sum is the total of the interval-censored observations.

This likelihood function is often referred as the *full likelihood* as compared to the partial likelihood (Cox 1972) because it involves parameters for the baseline hazard function in addition to the regression coefficients $\boldsymbol{\beta}$. The full likelihood is often used for analyzing interval-censored data because constructing a likelihood function that contains only the regression coefficients as conveniently as the Cox partial likelihood does for right-censored data is not straightforward (Finkelstein 1986).

Baseline Parameterization

Because any one of the baseline hazard, cumulative hazard, and survival functions determines the others, it is sufficient to parameterize one of them. For the baseline function, PROC ICPHREG supports the parameterizations that are described in the following subsections.

Piecewise Constant Model

As its name suggests, the piecewise constant hazard rate model parameterizes the baseline hazard function as a union of several disjoint intervals, within each of which the hazard rate is constant:

$$\lambda_0(t) = r_j \quad \text{if } a_{j-1} \le t < a_j,\ j = 1, \ldots, J$$

It follows that the baseline cumulative hazard function is

$$\Lambda_0(t) = \sum_{j=1}^{J} r_j \Delta_j(t)$$

where

$$\Delta_j(t) = \begin{cases} 0 & t < a_{j-1} \\ t - a_{j-1} & a_{j-1} \le t < a_j \\ a_j - a_{j-1} & t \ge a_j \end{cases}$$

To produce a meaningful hazard function, the r_j need to be bounded below by 0. Such a constraint can be removed by transforming the parameters to a natural log scale:

$$\alpha_j = \log(r_j), \quad j = 1, \ldots, J$$

PROC ICPHREG uses either the original or the transformed scale to fit piecewise constant models. You can change the scale by using the HAZSCALE= option. By default, the original scale is used.

Cubic Splines Model

For the proportional hazards model, Royston and Parmar (2002) propose modeling the log of the baseline cumulative hazard function in terms of natural cubic splines,

$$\log[\Lambda_0(t)] = \gamma_0 + \gamma_1 x + \gamma_2 v_1(x) + \cdots + \gamma_{m+1} v_J(x)$$

where $x = \log(t)$ represents the time on a log scale. The v_j are the basis functions, which are computed as

$$v_j(x) = (x - k_j)_+^3 - e_j(x - k_{\min})_+^3 - (1 - e_j)(x - k_{\max})_+^3$$

where

$$\begin{aligned} e_j &= \frac{k_{\max} - k_j}{k_{\max} - k_{\min}} \\ (x - a)_+ &= \max(0, x - a) \end{aligned}$$

Here, $k_{\min}$ and $k_{\max}$ are two terminal knots, and $k_1 < \cdots < k_J$ are m interval knots that are placed between $k_{\min}$ and $k_{\max}$. The degrees of freedom equals $m + 1$. When $m = 0$, the log of the baseline hazard becomes $\gamma_0 + \gamma_1 x$, which corresponds to a common form of the Weibull model. When $\gamma_1 = 1$, the Weibull model further reduces to the exponential model.

Semiparametric Model

Assuming that the underlying baseline hazard is a discrete function, Finkelstein (1986) proposed parameterizing the baseline survival function as a discrete function in which jumps occur only at the discrete times. You typically apply the algorithm proposed by Turnbull (1976) to reduce the number of parameters to estimate. First, you rank observed interval boundaries $\{L_i, R_i, i = 1, \ldots, n\}$ with labels of L and R attached. Then you identify the set of intervals for which L is the left boundary and R is the right boundary, and the reduced set of discrete times $s_0 = 0 < s_1 < \cdots < s_{m+1} = \infty$ is simply the ranked right boundaries of these intervals. The cumulative hazard function is assumed to be

$$\Lambda_0(t) = \sum_{j:s_j<t} \gamma_j, j = 1, \ldots, J$$

Specification of Effects

Each term in a model is called an *effect*. You specify effects in the MODEL statement by using a special notation that uses variable names and operators. There are two types of variables: *classification* (CLASS) variables and *continuous* variables. There are two primary types of operators: *crossing* and *nesting*. A third type, the *bar* operator, is used to simplify effect specification.

Variables that identify classification levels are called CLASS variables in SAS and are identified in a CLASS statement. These might also be called *categorical, qualitative, discrete,* or *nominal* variables. CLASS variables can be either character or numeric. The values of CLASS variables are called *levels*. For example, the CLASS variable Sex could have the levels "male" and "female."

In a model, an explanatory variable that is not declared in a CLASS statement is assumed to be continuous. Continuous variables must be numeric. For example, the heights and weights of subjects in an experiment are continuous variables.

The following list shows types of effects that are often useful in practice, where A, B, and C are classification variables and X1 and X2 are continuous variables:

- Regressor effects are specified by writing continuous variables by themselves: X1, X2.
- Polynomial effects are specified by joining two or more continuous variables with asterisks: X1*X2.
- Main effects are specified by writing classification variables by themselves: A, B, C.
- Crossed effects (interactions) are specified by using asterisks to join two or more classification variables: A*B, B*C, A*B*C.
- Nested effects are specified by following a main effect or crossed effect with a classification variable or list of classification variables that are enclosed in parentheses: B(A), C(B A), A*B(C). In the preceding example, B(A) is "B nested within A."
- Combinations of continuous and classification variables can be specified in the same way by using the crossing and nesting operators.

The bar operator uses a vertical bar (|) to join two effects. The bar operator is shorthand notation for including the left-hand side, the right-hand side, and the cross between them as effects in the model. For example, A |

B is equivalent to A B A*B. The effects that are joined by the bar operator can be classification variables, continuous variables, or combinations of effects that are defined by using operators. Multiple bars are permitted. For example, A | B | C means A B C A*B A*C B*C A*B*C.

You can specify the maximum number of variables in any effect that results from bar evaluation by specifying the maximum number, preceded by an @ sign. For example, A | B | C@2 results in effects that involve two or fewer variables: A B C A*B A*C B*C.

Computational Details

Design Matrix

The linear predictor part of a proportional hazards model is

$$\boldsymbol{\mu} = \mathbf{Z}'\boldsymbol{\beta}$$

where $\boldsymbol{\beta}$ is a vector of unknown regression coefficients and $\mathbf{Z}$ is a known design matrix. The ordering of these parameters is displayed in the "CLASS Level Information" table and in tables that display the parameter estimates of the fitted model.

When you use the PARAM=GLM option in the CLASS statement to specify an overparameterized model, some columns of $\mathbf{Z}$ can be linearly dependent on other columns. For example, when you specify a model that consists of a classification variable, the column that corresponds to any one of the levels of the classification variable is linearly dependent on the other columns of $\mathbf{Z}$. The columns of $\mathbf{Z}'\mathbf{Z}$ are checked in the order in which the model is specified for dependence on preceding columns. If a dependency is found, the parameter that corresponds to the dependent column and its standard error are set to 0 to indicate that it is not estimated. The test for linear dependence is controlled by the SINGULAR= option in the MODEL statement. You can use the ORDER= option in the CLASS statement to specify the order in which the levels of a classification variable are checked for dependencies. For full-rank parameterizations, the columns of the $\mathbf{Z}$ matrix are designed to be linearly independent.

Initial Values

The initial values of the regression coefficients $\boldsymbol{\beta}$ are all set to 0.

For the piecewise constant model, the initial values of the hazard parameters are set equal to the exponential rate that is estimated from an imputed data set. The data set is obtained by imputing a middle point for the interval-censored and left-censored observations while retaining the right-censored and exact observations. For the cubic spline model, the first spline coefficient, γ_0, is set to be the log of the exponential rate estimated with the previous imputed data, and the second spline coefficient, γ_1, is set to 1. The remaining spline coefficients, if there are any, are set to 0.

Maximum Likelihood Estimation

By default, the ICPHREG procedure uses a Newton-Raphson algorithm to maximize the log-likelihood function with respect to the parameters.

Denote the set of parameters that need to be estimated as $\boldsymbol{\omega} = \{\omega_j\}$, which consists of the parameters that determine baseline hazard function $\Lambda_0(t)$ and the regression coefficients $\boldsymbol{\beta}$. On the rth iteration, the

algorithm updates the parameter vector $\boldsymbol{\omega}_r$ with

$$\boldsymbol{\omega}_{r+1} = \boldsymbol{\omega}_r - \mathbf{H}^{-1}\mathbf{g}$$

where $\mathbf{H}$ is the Hessian (second derivative) matrix, and $\mathbf{g}$ is the gradient (first derivative) vector of the log-likelihood function, both evaluated at the current value of the parameter vector. That is,

$$\mathbf{g} = [g_j] = \left[\frac{\partial l}{\partial \omega_j}\right]$$

and

$$\mathbf{H} = [h_{ij}] = \left[\frac{\partial^2 l}{\partial \omega_i \partial \omega_j}\right]$$

The ICPHREG procedure also supports other optimization methods, such as quasi-Newton and Newton-Raphson with ridging. These methods are described in the section "Choosing an Optimization Algorithm" on page 545 in Chapter 20, "Shared Concepts and Topics."

Covariance and Correlation Matrix

The estimated covariance matrix of the parameter estimator is

$$\boldsymbol{\Sigma} = -\mathbf{H}^{-1}$$

where $\mathbf{H}$ is the Hessian matrix that is evaluated using the parameter estimates on the last iteration. If some parameters in the baseline function are held fixed, they are not incorporated in $\mathbf{H}$. Rows and columns that correspond to aliased parameters are not included in $\boldsymbol{\Sigma}$.

The correlation matrix is the normalized covariance matrix. That is, if σ_{ij} is an element of $\boldsymbol{\Sigma}$, then the corresponding element of the correlation matrix is $\sigma_{ij}/\sigma_i\sigma_j$, where $\sigma_i = \sqrt{\sigma_{ii}}$.

Choosing Break Points

There are no obvious ways to choose break points for parameterizing the baseline function in terms of a piecewise constant function or a cubic spline curve. For right-censored data, PROC PHREG chooses a set of points such that the resulting time intervals contain approximately equal numbers of event times. This is difficult for interval-censored data because event times are not fully observed. Friedman (1982) recommends choosing the points so that the expected number of events is comparable among the time intervals. For an interval-censored spline model, Cai and Betensky (2003) propose an ad hoc approach that uses the quantile values of the unique time points among $\{L_i, R_i, (L_i + R_i)/2, i = 1, \ldots, n\}$ for choosing the knot values.

Ibrahim, Chen, and Sinha (2001) propose the equally spaced quantile partition (ESQP) method for selecting break points in the right-censored data to fit the piecewise constant model. Suppose there are Q break points to be determined. The ICPHREG procedure modifies this method to handle interval-censored data. First, it imputes a middle point for each observation that is not right-censored. Then, it merges these values with the observed boundary values in the input data set, except for the right-censored observations. Next, it sorts these values in increasing order.

Suppose the unique values of the sorted sequence are $u_1 < u_2 < \cdots < u_M$. First, PROC ICPHREG computes the targeted quantile for each break point as $q_j = j/(Q+1)(j = 1, \cdots, Q)$. Then, it chooses the point u_{m+1}, where m equals the integer part of the product $q_j M$. If $q_j M$ is already an integer, then the chosen break point is set to be $(u_m + u_{m+1})/2$. When there are no ties in the sorted sequence for right-censored data, this method is identical to the original ESQP method.

Fit Statistics

Suppose that the model contains q estimated parameters and that n observations are used in model fitting. The fit criteria displayed by the ICPHREG procedure are calculated as follows:

- –2 log likelihood:

  $$-2\log(\mathrm{L})$$

  where L is the maximized likelihood for the model.

- Akaike's information criterion:

  $$\mathrm{AIC} = -2\log(\mathrm{L}) + 2q$$

- corrected Akaike's information criterion:

  $$\mathrm{AICC} = \mathrm{AIC} + \frac{2q(q+1)}{n-q-1}$$

- Bayesian information criterion:

  $$\mathrm{BIC} = -2\log(\mathrm{L}) + q\log(n)$$

For more information about AIC and BIC, see Akaike (1981, 1979). For a discussion of using AIC, AICC, and BIC in statistical modeling, see Simonoff (2003).

Predicted Values

Given a new vector of covariates $\mathbf{Z}_{\mathrm{new}}$, the linear predictor is computed as $\hat{\mu}_{\mathbf{Z}_{\mathrm{new}}} = \mathbf{Z}'_{\mathrm{new}}\hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}$ is the maximum likelihood estimate of $\boldsymbol{\beta}$. The variance of $\hat{\mu}_{\mathbf{Z}_{\mathrm{new}}}$ is estimated by

$$\hat{\sigma}^2_{\mathbf{Z}_{\mathrm{new}}} = \mathbf{Z}'_{\mathrm{new}}\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}}\mathbf{Z}_{\mathrm{new}}$$

where $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}}$ denotes the estimated covariance matrix for $\hat{\boldsymbol{\beta}}$.

Suppose the estimated baseline hazard is $\hat{\Lambda}_0(t)$. Given $\mathbf{Z}_{\mathrm{new}}$, the cumulative hazard function can be predicted by

$$\hat{\Lambda}(t;\mathbf{Z}_{\mathrm{new}}) = \hat{\Lambda}_0(t)e^{\mathbf{Z}'_{\mathrm{new}}\hat{\boldsymbol{\beta}}}$$

Denote the vector of parameters that is used for obtaining $\hat{\Lambda}_0(t)$ as $\boldsymbol{\tau}$. It is apparent that $\boldsymbol{\tau} \cap \boldsymbol{\beta} = \emptyset$. The vector of parameters that need to be estimated can be represented as $\boldsymbol{\omega} = (\boldsymbol{\beta}, \boldsymbol{\tau})$.

The variance of $\hat{\Lambda}(t;\mathbf{Z}_{\mathrm{new}})$ can be estimated by applying the delta method:

$$\hat{\sigma}^2(\hat{\Lambda}(t;\mathbf{Z}_{\mathrm{new}})) = P(t,\hat{\boldsymbol{\omega}})'\boldsymbol{\Sigma}\,P(t,\hat{\boldsymbol{\omega}})$$

where

$$P(t, \boldsymbol{\omega}) = \frac{\partial \Lambda(t; \mathbf{Z}_{\text{new}})}{\partial \boldsymbol{\omega}}$$

and $\boldsymbol{\Sigma}$ denotes the estimated covariance matrix for $\hat{\boldsymbol{\omega}}$.

Given $\mathbf{Z}_{\text{new}}$, the predicted survival function is estimated by

$$\hat{S}(t; \mathbf{Z}_{\text{new}}) = \exp(\hat{\Lambda}(t; \mathbf{Z}_{\text{new}}))$$

The standard error of $\hat{S}(t; \mathbf{Z}_{\text{new}})$ can be conveniently estimated by an application of the delta method:

$$\hat{\sigma}(\hat{S}(t; \mathbf{Z}_{\text{new}})) = \hat{S}(t; \mathbf{Z}_{\text{new}})\hat{\sigma}(\hat{\Lambda}(t; \mathbf{Z}_{\text{new}}))$$

By default, a natural log transformation is applied to obtain the pointwise confidence limits for $S(t; \mathbf{Z}_{\text{new}})$ and $\Lambda(t; \mathbf{Z}_{\text{new}})$. You can use the CLTYPE= option to specify a different transformation for $S(t; \mathbf{Z}_{\text{new}})$.

Hazard Ratios

Consider a dichotomous risk factor variable X that takes the value 1 if the risk factor is present and 0 if the risk factor is absent. The log-hazard function is

$$\log[\lambda(t|X)] = \log[\lambda_0(t)] + \beta_1 X$$

where $\lambda_0(t)$ is the baseline hazard function.

The hazard ratio ψ is defined as the ratio of the hazard for those who have the risk factor ($X = 1$) to the hazard for those who do not have the risk factor ($X = 0$). The log of the hazard ratio is

$$\log(\psi) \equiv \log[\psi(X = 1, X = 0)] = \log[\lambda(t|X = 1)] - \log[\lambda(t|X = 0)] = \beta_1$$

In general, the hazard ratio can be computed by exponentiating the difference of the log-hazard between any two population profiles. This is the approach taken by the HAZARDRATIO statement, so the computations are available regardless of parameterization, interactions, and nestings. However, as shown in the preceding equation for $\log(\psi)$, hazard ratios of main effects can be computed as functions of the parameter estimates. The remainder of this section is concerned with this methodology.

The parameter β_1 that is associated with X represents the change in the log-hazard from $X = 0$ to $X = 1$. So the hazard ratio is obtained by simply exponentiating the value of the parameter that is associated with the risk factor. The hazard ratio indicates how the hazard changes as you change X from 0 to 1. For example, $\psi = 2$ means that the hazard when $X = 1$ is twice the hazard when $X = 0$.

Suppose the values of the dichotomous risk factor are coded as constants a and b instead of 0 and 1. The hazard when $X = a$ becomes $\lambda(t)\exp(a\beta_1)$, and the hazard when $X = b$ becomes $\lambda(t)\exp(b\beta_1)$. The hazard ratio that corresponds to an increase in X from a to b is

$$\psi = \exp[(b - a)\beta_1] = [\exp(\beta_1)]^{b-a} \equiv [\exp(\beta_1)]^c$$

Note that for any a and b such that $c = b - a = 1$, $\psi = \exp(\beta_1)$. So the hazard ratio can be interpreted as the change in the hazard for any increase of one unit in the corresponding risk factor. However, the change in

hazard for some amount other than one unit is often of greater interest. For example, a change of one pound in body weight might be too small to be considered important, whereas a change of 10 pounds might be more meaningful. The hazard ratio for a change in X from a to b is estimated by raising the hazard ratio estimate for a unit change in X to the power of $c = b - a$ as shown previously.

For a polytomous risk factor, the computation of hazard ratios depends on how the risk factor is parameterized. For illustration, suppose that Cell is a risk factor that has four categories: Adeno, Large, Small, and Squamous.

For the effect parameterization scheme (PARAM=EFFECT) with Squamous as the reference group, the design variables for Cell are as follows:

	Design Variables		
Cell	X_1	X_2	X_3
Adeno	1	0	0
Large	0	1	0
Small	0	0	1
Squamous	–1	–1	–1

The log-hazard for Adeno is

$$\begin{aligned}\log[\lambda(t|\text{Adeno})] &= \log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) \\ &= \lambda_0(t) + \beta_1\end{aligned}$$

The log-hazard for Squamous is

$$\begin{aligned}\log[\lambda(t|\text{Squamous})] &= \log[\lambda_0(t)] + \beta_1(X_1 = -1) + \beta_2(X_2 = -1) + \beta_3(X_3 = -1)) \\ &= \log[\lambda_0(t)] - \beta_1 - \beta_2 - \beta_3\end{aligned}$$

Therefore, the log-hazard ratio of Adeno versus Squamous

$$\begin{aligned}\log[\psi(\text{Adeno, Squamous})] &= \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})] \\ &= 2\beta_1 + \beta_2 + \beta_3\end{aligned}$$

For the reference cell parameterization scheme (PARAM=REF) in which Squamous is the reference cell, the design variables for Cell are as follows:

	Design Variables		
Cell	X_1	X_2	X_3
Adeno	1	0	0
Large	0	1	0
Small	0	0	1
Squamous	0	0	0

The log-hazard ratio of Adeno versus Squamous is

$$
\begin{aligned}
&\log(\psi(\text{Adeno, Squamous})) \\
&\quad = \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})] \\
&\quad = (\log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0)) - \\
&\qquad (\log[\lambda_0(t)] + \beta_1(X_1 = 0) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0)) \\
&\quad = \beta_1
\end{aligned}
$$

For the GLM parameterization scheme (PARAM=GLM), the design variables are as follows:

	Design Variables			
Cell	X_1	X_2	X_3	X_4
Adeno	1	0	0	0
Large	0	1	0	0
Small	0	0	1	0
Squamous	0	0	0	1

The log-hazard ratio of Adeno versus Squamous is

$$
\begin{aligned}
&\log(\psi(\text{Adeno, Squamous})) \\
&\quad = \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})] \\
&\quad = \log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) + \beta_4(X_4 = 0)) - \\
&\qquad (\log(\lambda_0(t)) + \beta_1(X_1 = 0) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) + \beta_4(X_4 = 1)) \\
&\quad = \beta_1 - \beta_4
\end{aligned}
$$

Consider Cell as the only risk factor. The computation of the hazard ratio of Adeno versus Squamous for various parameterization schemes is shown in Table 70.7.

Table 70.7 Hazard Ratio of Adeno to Squamous

	Parameter Estimates				
PARAM=	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	**Hazard Ratio Estimates**
EFFECT	0.5772	–0.2115	0.2454		$\exp(2 \times 0.5772 - 0.2115 + 0.2454) = 3.281$
REF	1.8830	0.3996	0.8565		$\exp(1.8830) = 3.281$
GLM	1.8830	0.3996	0.8565	0.0000	$\exp(1.8830) = 3.281$

The fact that the log-hazard ratio ($\log(\psi)$) is a linear function of the parameters enables the HAZARDRATIO statement to compute the hazard ratio of the main effect even in the presence of interactions and nest effects.

To customize hazard ratios for specific units of change for a continuous risk factor, you can use the UNITS= option in a HAZARDRATIO statement to specify a list of relevant units for each explanatory variable in

the model. Estimates of these customized hazard ratios are shown in a separate table. Let (V_j, U_j) be a confidence interval for $\log(\psi)$. The corresponding lower and upper confidence limits for the customized hazard ratio $\exp(c\beta_j)$ are $\exp(cV_j)$ and $\exp(cU_j)$, respectively for $c > 0$, or $\exp(cU_j)$ and $\exp(cV_j)$, respectively for $c < 0$.

Let $\mathbf{e}_j$ be the *j*th unit vector—that is, the *j*th entry of the vector is 1 and all other entries are 0. The hazard ratio for the explanatory variable with regression coefficient $\beta_j = \mathbf{e}'_j\boldsymbol{\beta}$ is defined as $\exp(\beta_j)$. In general, a log-hazard ratio can be written as $\mathbf{h}'\boldsymbol{\beta}$ (a linear combination of the regression coefficients), and the hazard ratio $\exp(\mathbf{h}'\boldsymbol{\beta})$ is obtained by replacing $\mathbf{e}_j$ with $\mathbf{h}$.

Point Estimate

The hazard ratio $\exp(\mathbf{e}'_j\boldsymbol{\beta})$ is estimated by $\exp(\mathbf{e}'_j\hat{\boldsymbol{\beta}})$, where $\hat{\boldsymbol{\beta}}$ is the maximum likelihood estimate of the regression coefficients $\boldsymbol{\beta}$.

Wald's Confidence Limits

The $100(1-\alpha)\%$ confidence limits for the hazard ratio are calculated as

$$\exp\left(\mathbf{e}'_j\hat{\boldsymbol{\beta}} \pm z_{\alpha/2}\sqrt{\mathbf{e}'_j\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}}\mathbf{e}_j}\right)$$

where $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}}$ is estimated covariance matrix and $z_{\alpha/2}$ is the $100(1-\alpha/2)$ percentile point of the standard normal distribution.

Left-Truncation of Failure Times

Left-truncation occurs when individuals are not observed at the natural time origin of the phenomenon under study but come under observation at some known later time (called the left-truncation time). Any individuals whose left-truncation time exceeds the event time cannot be observed. Thus, any contribution to the likelihood must be conditional on the truncation limit having been exceeded.

You use the ENTRY= option to specify the variable that represents the left-truncation time. Suppose T1 and (T2, T3) represent the left-truncation time and the observed interval, respectively. To account for left-truncation, you specify the following statements:

```
proc icphreg;
   model (T2,T3)=X1-X10/entry=T1;
   title 'The ENTRY= option is Specified';
run;
```

For the specified truncation time to be meaningful, its value must be smaller than the values of the interval boundaries. Otherwise, the observation is removed from the analysis.

Residuals and Diagnostic Statistics

The ICPHREG procedure computes two types of residuals. Residuals are available for the piecewise constant hazard model and the cubic splines model.

For the *i*th subject whose observed interval is (L_i, R_i), the Lagakos residual is defined in Farrington (2000) as

$$r_i^L = \frac{\hat{S}_i(L_i)\log[\hat{S}_i(L_i)] - \hat{S}_i(R_i)\log[\hat{S}_i(R_i)]}{\hat{S}_i(L_i) - \hat{S}_i(R_i)}$$

You can request Lagakos residuals in an output data set by using the keyword RESLAG in the OUTPUT statement.

The deviance residual is defined as

$$r_i^D = \mathrm{sign}(r_i^L)\left\{2\log\left[\frac{\hat{S}_0(L_i)^{\eta_i} - \hat{S}_0(R_i)^{\eta_i}}{\hat{S}_0(L_i)^{\exp(\mathbf{Z}_i'\hat{\boldsymbol{\beta}})} - \hat{S}_0(R_i)^{\exp(\mathbf{Z}_i'\hat{\boldsymbol{\beta}})}}\right]^{1/2}\right\}$$

where

$$\eta_i = \frac{\log[\hat{\Lambda}_0(R_i)] - \log[\hat{\Lambda}_0(L_i)]}{\hat{\Lambda}_0(R_i) - \hat{\Lambda}_0(L_i)}$$

and where $\eta_i = 0$ if $R_i = \infty$ and $\eta_i = \infty$ if $L_i = 0$.

You can request deviance residuals in an output data set by using the keyword RESDEV in the OUTPUT statement.

The ICPHREG procedure computes the length of input intervals. For the *i*th subject whose observed interval is (L_i, R_i), the length is defined as

$$\mathrm{Len}_i = \begin{cases} R_i - L_i & \text{if } R_i \geq L_i > 0 \\ R_i & \text{if } R_i > 0 \text{ and } L_i = 0 \text{ or } L_i \text{ is missing} \\ -1 & \text{otherwise} \end{cases}$$

You can request interval lengths in an output data set by using the keyword INTERVAL in the OUTPUT statement.

EM Algorithm and Extensions

The expectation-maximization (EM) algorithm, as described in Wang et al. (2016) and Zeng, Mao, and Lin (2016), can be used to fit certain types of proportional hazards models to interval-censored data.

Suppose that the observations to be analyzed consist of interval-censored outcomes $\mathcal{D} = \{[L_i, R_i]; \mathbf{Z}_i\}$, $i = 1, \ldots, n$, where n is the number of subjects. $\mathbf{Z}_i$ denotes a p-dimensional vector of covariates for the ith subject.

Assuming that there is no exact observation ($L_i = R_i$), the full likelihood function is

$$\begin{aligned} \mathrm{L}(\boldsymbol{\theta}) &= \prod_{i=1}^{n} [S(L_i; \mathbf{Z}_i) - S(R_i; \mathbf{Z}_i)] \\ &= \prod_{i=1}^{n} [1 - S(R_i; \mathbf{Z}_i)]^{\Delta_{i1}} [S(L_i; \mathbf{Z}_i) - S(R_i; \mathbf{Z}_i)]^{\Delta_{i2}} [S(L_i; \mathbf{Z}_i)]^{\Delta_{i3}} \end{aligned}$$

where Δ_{i1} indicates whether the ith subject is left-censored ($L_i = 0$), Δ_{i2} indicates whether the ith subject is interval-censored ($0 < L_i < R_i < \infty$), and Δ_{i3} indicates whether the ith subject is right-censored ($R_i = \infty$).

Assume that the baseline hazard function is of the following form,

$$\Lambda_0(t) = \sum_{k=1}^{K} \gamma_k b_k(t)$$

where $b_k(t)$ are known functions that are nondecreasing and nonnegative, and γ_k are nonnegative baseline parameters.

Let $\{W_{ik} : i = 1, \ldots, n; k = 1, \ldots, K\}$ be a set of latent variables that follow Poisson distributions with means $\gamma_k b_k(L_i) \exp(\mathbf{Z}_i' \boldsymbol{\beta})$. Let $\{U_{ik} : i = 1, \ldots, n; k = 1, \ldots, K\}$ be a set of latent variables that follow Poisson distributions with means $\gamma_k (b_k(R_i) - b_k(L_i)) \exp(\mathbf{Z}_i' \boldsymbol{\beta})$. Define $W_i = \sum_{k=1}^{K} W_{ik}$ and $U_i = \sum_{k=1}^{K} U_{ik}$.

The full likelihood can be rewritten as

$$\mathrm{L}(\boldsymbol{\theta}) = \prod_{i=1}^{n} P(W_i > 0)^{\Delta_{i1}} P(W_i = 0, U_i > 0)^{\Delta_{i2}} P(W_i = 0, U_i = 0)^{\Delta_{i3}}$$

The complete-data likelihood is

$$\mathrm{L}_c(\boldsymbol{\theta}) = \prod_{i=1}^{n} \prod_{k=1}^{K} f_{W_{ik}}(W_{ik}) f_{U_{ik}}(U_{ik})^{\Delta_{i2} + \Delta_{i3}}$$

where $f_V(\cdot)$ denotes the Poisson probability mass function for the variable V. It is straightforward to verify that the integration of $\mathrm{L}_c(\boldsymbol{\theta})$ with respect to latent variables leads to the full likelihood $\mathrm{L}(\boldsymbol{\theta})$.

The EM algorithm proceeds as follows. Let the current parameter estimates be $\boldsymbol{\theta}^{(d)} = (\boldsymbol{\beta}^{(d)}, \boldsymbol{\gamma}^{(d)})'$. Define

$$\begin{aligned} w_{ik} &= E(W_{ij}|\mathcal{D}, \boldsymbol{\theta}^{(d)}) \\ u_{ik} &= E(U_{ij}|\mathcal{D}, \boldsymbol{\theta}^{(d)}) \\ w_i &= E(W_i|\mathcal{D}, \boldsymbol{\theta}^{(d)}) \\ u_i &= E(U_i|\mathcal{D}, \boldsymbol{\theta}^{(d)}) \end{aligned}$$

The expected complete-data log likelihood $\mathrm{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(d)}) = E[\log(\mathrm{L}_c(\boldsymbol{\theta}))|\mathcal{D}, \boldsymbol{\theta}^{(d)}]$ is computed as

$$\begin{aligned} \mathrm{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(d)}) &= \sum_{i=1}^{n}\sum_{k=1}^{K}\Big\{[w_{ik} + (\Delta_{i2} + \Delta_{i3})u_{ik}]\left[\log(\gamma_k) + \mathbf{Z}_i'\boldsymbol{\beta}\right] \\ &\quad -\gamma_k \exp(\mathbf{Z}_i'\boldsymbol{\beta})\,[(\Delta_{i2} + \Delta_{i1})b_k(R_i) + \Delta_{i3}b_k(L_i)]\Big\} + B(\boldsymbol{\theta}^{(d)}) \end{aligned}$$

where $B(\boldsymbol{\theta}^{(d)})$ is a constant.

The quantities w_{ik} and u_{ik} are computed as follows,

$$w_{ik} = \frac{\gamma_k^{(d)} b_k(R_i) w_i}{\Lambda_0^{(d)}(R_i)}$$

$$u_{ik} = \frac{\gamma_k^{(d)}[b_k(R_i) - b_k(L_i)]u_i}{\Lambda_0^{(d)}(R_i) - \Lambda_0^{(d)}(L_i)}$$

where

$$w_i = \frac{\Lambda_0^{(d)}(R_i)\exp(\mathbf{Z}_i'\boldsymbol{\beta}^{(d)})\Delta_{i1}}{1 - \exp\left[\Lambda_0^{(d)}(R_i)\exp(\mathbf{Z}_i'\boldsymbol{\beta}^{(d)})\right]}$$

$$u_i = \frac{[\Lambda_0^{(d)}(R_i) - \Lambda_0^{(d)}(L_i)]\exp(\mathbf{Z}_i'\boldsymbol{\beta}^{(d)})\Delta_{i2}}{1 - \exp\left\{\left[\Lambda_0^{(d)}(R_i) - \Lambda_0^{(d)}(L_i)\right]\exp(\mathbf{Z}_i'\boldsymbol{\beta}^{(d)})\right\}}$$

Solve $\partial\mathrm{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(d)})/\partial\gamma_k = 0$ for $k = 1, \ldots, K$. It follows that the $\gamma_k^{(d)}$ can be updated as

$$\gamma_k^{(d+1)} = \frac{\sum_{i=1}^{n}(z_{ik} + \Delta_{i2}w_{ik})}{\sum_{i=1}^{n}[(\Delta_{i1} + \Delta_{i2})b_k(R_i) + \Delta_{i3}b_k(L_i)]\exp(\mathbf{Z}_i'\boldsymbol{\beta})}$$

The partial derivative of $\mathrm{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(d)})$ with respect to $\boldsymbol{\beta}$ is

$$\frac{\partial\mathrm{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(d)})}{\partial\boldsymbol{\beta}} = \sum_{i=1}^{n}\left\{(w_i + (\Delta_{i2} + \Delta_{i3})u_i) - \left[(\Delta_{i1} + \Delta_{i2})\Lambda_0^{(d)}(R_i) + \Delta_{i3}\Lambda_0^{(d)}(L_i)\right]\exp(\mathbf{Z}_i'\boldsymbol{\beta})\right\}\mathbf{Z}_i$$

After plugging in $\{\gamma_k^{(d+1)}, k = 1, \ldots, K\}$, you can update the parameters $\boldsymbol{\beta}$ by using the one-step Newton-Raphson method (Zeng, Mao, and Lin 2016).

The EM algorithm alternates between updating $\boldsymbol{\gamma}$ and updating $\boldsymbol{\beta}$ until convergence.

You can use the EM algorithm to fit the semiparametric model and the piecewise constant hazard model in PROC ICPHREG. The option is NLOPTIONS(TECH=EM) in the PROC ICPHREG statement.

Semiparametric Model and Time-Dependent Covariates

A typical way that interval-censored data are generated is through a process of repeated assessments. Suppose that $V_1 < V_2 < \cdots < V_M$ are a random sequence of assessment times. Denote $\tilde{V} = (V_0 = 0, V_1, V_2, \ldots, V_M, V_{M+1} = \infty)$ and $\tilde{D} = (D_0 = 0, D_1, D_2, \ldots, D_M)$, where $D_m = I(V_{m-1} < T < V_m), m = 1, \ldots, M$.

For the ith subject, $i = 1, \ldots, n$, let K_i, T_i, $\tilde{V}_i = (V_{i0}, V_{i1}, \ldots, V_{i(K_i+1)})$, $\tilde{D}_i = (D_{i0}, D_{i1}, \ldots, D_{iK_i})$, and $\mathbf{Z}_i(\cdot)$ be the number of assessments, event time, assessment time vector, the indicator vector, and time-dependent covariate process, respectively. Suppose that T_i is interval-censored between two assessment times, L_i and R_i, where $L_i = \max_{V_j}\{V_j < T_i, j = 0, \ldots, K_i\}$ and $R_i = \min_{V_j}\{V_j < T_i, j = 1, \ldots, K_i + 1\}$. Let $s_1 < \cdots < s_J$ be the sorted right boundaries of the Turnbull intervals for $\{(L_i, R_i] : i = 1, \ldots, n\}$.

Suppose that the time-dependent covariates process $\mathbf{Z}_i(\cdot)$ change value only at assessment times. Let $(\mathbf{Z}_{i1}, \mathbf{Z}_{i2}, \ldots, \mathbf{Z}_{i(J+1)})$ be the observed covariate vectors at times $(0, s_1, \ldots, s_J)$.

Under the semiparametric model, the baseline cumulative hazard function is

$$\Lambda_0(t) = \sum_{j:s_j<t} \gamma_j, j = 1, \ldots, J$$

For the ith subject, the cumulative hazard function is computed as

$$\Lambda_i(t) = \sum_{j:s_j<t} \gamma_j \exp(\mathbf{Z}'_{ij}\boldsymbol{\beta})$$

The full likelihood function is

$$\mathrm{L}(\boldsymbol{\theta}) = \prod_{i=1}^{n} \{1 - \exp[-\Lambda_i(R_i)]\}^{\Delta_{i1}} \{\exp[-\Lambda_i(L_i)] - \exp[-\Lambda_i(R_i)]\}^{\Delta_{i2}} \{\exp[-\Lambda_i(L_i)]\}^{\Delta_{i3}}$$

where Δ_{i1} indicates whether the ith subject is left-censored ($L_i = 0$), Δ_{i2} indicates whether the ith subject is interval-censored ($0 < L_i < R_i < \infty$), and Δ_{i3} indicates whether the ith subject is right-censored ($R_i = \infty$).

As the following derivation shows, the EM algorithm can be adapted straightforwardly to fit the semiparametric model that contains time-dependent covariates.

Let $b_j(t) = I(s_j \le t)$, and redefine the latent Poisson variables as

$$E(W_{ij}) = \gamma_j \exp(\mathbf{Z}'_{ij}\boldsymbol{\beta}) b_j(R_i) = \gamma_j \exp(\mathbf{Z}'_{ij}\boldsymbol{\beta}) I(s_j < R_i)$$

$$E(U_{ij}) = \gamma_j \exp(\mathbf{Z}'_{ij}\boldsymbol{\beta})[b_j(R_i) - b_j(L_i)] = \gamma_j \exp(\mathbf{Z}'_{ij}\boldsymbol{\beta}) I(L_i < s_j < R_i)$$

The expected complete-data log likelihood $\mathrm{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(d)}) = E[\log(\mathrm{L}_c(\boldsymbol{\theta}))|\mathcal{D}, \boldsymbol{\theta}^{(d)}]$ becomes

$$\begin{aligned}\mathrm{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(d)}) \quad = \quad & \sum_{i=1}^{n}\sum_{k=1}^{J} \Big\{ [w_{ik} + (\Delta_{i2} + \Delta_{i3})u_{ik}] \Big[\log(\gamma_k) + \mathbf{Z}'_{ij}\boldsymbol{\beta}\Big] \\ & -\gamma_k \exp(\mathbf{Z}'_{ij}\boldsymbol{\beta}) \big[(\Delta_{i2} + \Delta_{i1}) b_j(R_i) + \Delta_{i3} b_j(L_i)\big] \Big\} + B(\boldsymbol{\theta}^{(d)})\end{aligned}$$

where $B(\boldsymbol{\theta}^{(d)})$ is a constant and w_{ik} and u_{ik} are computed as follows:

$$w_{ik} = \frac{\gamma_k^{(d)} b_j(R_i) \exp(\mathbf{Z}_{ij}'\boldsymbol{\beta}) w_i}{\Lambda_i^{(d)}(R_i)}$$

$$u_{ik} = \frac{\gamma_k^{(d)} [b_j(R_i) - b_j(L_i)] \exp(\mathbf{Z}_{ij}'\boldsymbol{\beta}) u_i}{\Lambda_i^{(d)}(R_i) - \Lambda_i^{(d)}(L_i)}$$

$$w_i = \frac{\Lambda_i^{(d)}(R_i)\Delta_{i1}}{1 - \exp\left[\Lambda_i^{(d)}(R_i)\right]}$$

$$u_i = \frac{[\Lambda_i^{(d)}(R_i) - \Lambda_i^{(d)}(L_i)]\Delta_{i2}}{1 - \exp\left\{\left[\Lambda_i^{(d)}(R_i) - \Lambda_i^{(d)}(L_i)\right]\right\}}$$

You use the ID statement to fit the semiparametric model that contains time-dependent covariates. The levels of the ID variable identify the subjects to be analyzed.

Variance Estimation

Louis's Method

Let $\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\gamma}})$ be the maximum likelihood estimates as found by the EM algorithm. Under suitable conditions, you can apply Louis's method (Louis 1982) to obtain the covariance matrix of $\hat{\boldsymbol{\theta}}$.

The observed information matrix is computed as

$$I(\hat{\boldsymbol{\theta}}) = -\frac{\partial^2 \mathrm{Q}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})}{\partial\boldsymbol{\theta}\,\partial\boldsymbol{\theta}'} - \mathrm{cov}\left\{\frac{\partial \log \mathrm{L}_c(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}}\right\}$$

and its inverse $I^{-1}(\hat{\boldsymbol{\theta}})$ is the estimated covariance of $\hat{\boldsymbol{\theta}}$.

Louis's method is the default method of calculating standard errors for the semiparametric model.

Profile Likelihood Method

You can use the profile likelihood method of Murphy and Van Der Vaart (2000) to estimate the covariance matrix of $\hat{\boldsymbol{\beta}}$. The profile log-likelihood function is defined as

$$\mathrm{PL}(\boldsymbol{\beta}) = \max_{\boldsymbol{\gamma}\in(D)} \log \mathrm{L}(\boldsymbol{\beta}, \boldsymbol{\gamma})$$

where $\mathcal{D}$ is the parameter space of $\boldsymbol{\gamma}$.

The Hessian matrix of $\mathrm{PL}(\boldsymbol{\beta})$ can be computed using numerical differentiation. Let $\mathbf{e}_k$ be the identity vector for $k = 1, \ldots, p$, and let l be a small perturbation. The (j,k)th element of the Hessian matrix can be approximated by

$$H_{ij} = \frac{\mathrm{PL}(\boldsymbol{\beta}) - \mathrm{PL}(\boldsymbol{\beta} + l\cdot\mathbf{e}_j) - \mathrm{PL}(\boldsymbol{\beta} + l\cdot\mathbf{e}_k) - \mathrm{PL}(\boldsymbol{\beta} + l\cdot\mathbf{e}_j + l\cdot\mathbf{e}_k)}{l^2}$$

The covariance matrix of $\hat{\boldsymbol{\beta}}$ is estimated by inverting the negative of the Hessian matrix.

You can use the profile likelihood method for the semiparametric model by specifying the PLVARIANCE option in the MODEL statement. But be aware that this computation is iterative and can consume a relatively large amount of CPU time.

EMICM Algorithm

Pan (1999) proposes using the iterative convex minorant (ICM) algorithm to fit semiparametric proportional hazards models to interval-censored data.

Define $\alpha_j = \sum_{k=1}^{j} \gamma_k$. Denote $\alpha_0 = 0$ and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{J-1})'$. The full likelihood function can be rewritten in terms of $\boldsymbol{\alpha}$ and the regression coefficients $\boldsymbol{\beta}$.

Maximizing the likelihood with respect to $\boldsymbol{\theta} = (\boldsymbol{\gamma}, \boldsymbol{\beta})$ is equivalent to maximizing it with respect to $(\boldsymbol{\alpha}, \boldsymbol{\beta})$. Because the α_j are naturally ordered, the optimization is subject to the following constraint:

$$C = \{\mathbf{x} = (\alpha_1, \ldots, \alpha_{J-1}) : 0 \le \alpha_1 \le \cdots \le \alpha_{J-1} \le 1\}$$

Denote the log-likelihood function as $l(\boldsymbol{\alpha}, \boldsymbol{\beta})$. Because the regression coefficients $\boldsymbol{\beta}$ are not constrained, you can update them by using the one-step Newton-Raphson method as in the EM algorithm. Pan (1999) suggests using the ICM algorithm to update the baseline parameters $\boldsymbol{\alpha}$; doing so essentially treats $\boldsymbol{\beta}$ as fixed and maximizes the function $l(\boldsymbol{\alpha}) = l(\boldsymbol{\alpha}|\boldsymbol{\beta})$. Suppose that the maximum of $l(\boldsymbol{\alpha})$ occurs at $\hat{\boldsymbol{\alpha}}$. Mathematically, it can be proved that $\hat{\boldsymbol{\alpha}}$ equals the maximizer of the following quadratic function,

$$g^*(\mathbf{x}|\mathbf{y}, \mathbf{W}) = -\frac{1}{2}(\mathbf{x} - \mathbf{y})'\mathbf{W}(\mathbf{x} - \mathbf{y})$$

where $\mathbf{y} = \hat{\boldsymbol{\alpha}} + \mathbf{W}^{-1}\nabla l(\hat{\boldsymbol{\alpha}})$, $\nabla l(\cdot)$ denotes the derivatives of $l(\cdot)$ with respect to $\boldsymbol{\alpha}$, and $\mathbf{W}$ is a positive definite matrix of size $(J-1) \times (J-1)$ (Groeneboom and Wellner 1992).

The ICM algorithm updates $\boldsymbol{\alpha}$ as follows. For the dth iteration, the algorithm updates the quantity

$$\mathbf{y}^{(d)} = \hat{\boldsymbol{\alpha}}^{(d-1)} - \mathbf{W}^{-1}(\hat{\boldsymbol{\alpha}}^{(d-1)})\nabla l(\hat{\boldsymbol{\alpha}}^{(d-1)})$$

where $\hat{\boldsymbol{\alpha}}^{(d-1)}$ is the parameter estimate from the previous iteration and $\mathbf{W}(\hat{\boldsymbol{\alpha}}^{(d-1)}) = \mathrm{diag}(w_j, j = 1, \ldots, J-1)$ is a positive definite diagonal matrix that depends on $\hat{\boldsymbol{\alpha}}^{(l-1)}$. A convenient choice for $\mathbf{W}(\boldsymbol{\alpha})$ is the negative of the second-order derivative of the log-likelihood function $l(\boldsymbol{\alpha})$:

$$w_j = w_j(\boldsymbol{\alpha}) = -\frac{\partial^2}{\partial \alpha_j^2} l(\boldsymbol{\alpha})$$

Given $\mathbf{y} = \mathbf{y}^{(d)} = (y_1^{(d)}, \ldots, y_{J-1}^{(d)})'$ and $\mathbf{W} = \mathbf{W}(\hat{\boldsymbol{\alpha}}^{(d-1)})$, the parameter estimate $\hat{\boldsymbol{\alpha}}^{(d)}$ maximizes the quadratic function $g^*(\mathbf{x}|\mathbf{y}, \mathbf{W})$.

Define the cumulative sum diagram $\{P_k, k = 0, \ldots, J-1\}$ as a set of m points in the plane, where $P_0 = (0, 0)$ and

$$P_k = \Big(\sum_{i=1}^{k} w_i, \sum_{i=1}^{k} w_i y_i^{(l)}\Big)$$

Technically, $\hat{\boldsymbol{\alpha}}^{(d)}$ equals the left derivative of the convex minorant, or in other words, the largest convex function below the diagram $\{P_k, k = 0, \ldots, J-1\}$. You can solve this optimization problem by using the pool-adjacent-violators algorithm (Groeneboom and Wellner 1992).

The EMICM algorithm combines the EM algorithm and the ICM algorithm by alternating the two different steps in its iterations. Whereas the EM step updates both the baseline parameters and the regression

coefficients, the ICM step updates only the baseline parameters. If the ICM step does not increases the likelihood value, the parameter changes are halved for the next iteration. The process repeats a maximum of five times, until an increase in the likelihood value is found.

The EMICM algorithm is the default method of fitting the semiparametric model. You can use it to fit the piecewise constant hazard model by specifying the NLOPTIONS(TECH=EMICM) option in the PROC ICPHREG statement.

Input and Output Data Sets

OUT= Output Data Set in the BASELINE Statement

The OUT= data set in the BASELINE statement contains all the variables in the COVARIATES= data set, along with statistics you request by specifying *keyword=name* options. There are *n* observations in the OUT= data set for each observation in the COVARIATES= data set, where *n* is the number of break points for the fitted model.

Missing Values

Observations that contain a missing value for both the left boundary value and the right boundary value in the response are not used in the analysis. If a FREQ variable value is missing or 0, the observation is not used. If a STRATA variable is missing, the observation is not used unless you specify the MISSING option in the STRATA statement. If any explanatory variable that you specify in the MODEL statement has a missing value, that observation is not used in the model fitting. Predicted values, residuals, and regression diagnostics are computed for all observations that have no missing explanatory variable values.

Displayed Output

PROC ICPHREG displays the following information as results of the model fitting.

Model Information

The “Model Information” table displays the two-level name of the input data set, the type of model fitted, the parameterization type that is used for the baseline function, and the name and label of the dependent variables. If you specify the FREQ statement, this table also displays the name and label of the frequency variable.

Class Level Information

If you specify a CLASS statement, PROC ICPHREG outputs the “Class Level Information” table. This table displays the design information for the classification variables when the parameterization is full rank.

Class Variable Levels

If you specify a CLASS statement, PROC ICPHREG outputs the "Class Variable Levels" table. This table displays the levels of classification variables when the parameterization is not full rank.

Fit Statistics

The "Fit Statistics" table displays the negative of twice the log likelihood, Akaike's information criterion (AIC), the corrected Akaike's information criterion (AICC), and the Bayesian information criterion (BIC). This table is displayed by default.

Analysis of Maximum Likelihood Parameter Estimates

The "Analysis of Maximum Likelihood Parameter Estimates" table displays the parameter name, the degrees of freedom for each parameter, the maximum likelihood estimate of each parameter, the estimated standard error of the parameter estimator, confidence limits for each parameter, a chi-square statistic for testing whether the parameter is 0, and the associated p-value for the statistic. This table is displayed by default.

Iteration History for Parameter Estimates

If you specify the ITHISTORY option in the PROC ICPHREG statement, the procedure outputs a table that contains the following for each iteration in the iterative procedure for model fitting: the iteration number, the negative of twice of the log likelihood, the gradient, and values of all parameters in the model.

Last Evaluation of the Gradient

If you specify the ITHISTORY option in the PROC ICPHREG statement, the procedure displays the last evaluation of the gradient vector.

Last Evaluation of the Hessian

If you specify the ITHISTORY option in the PROC ICPHREG statement, the procedure displays the last evaluation of the Hessian matrix.

Estimated Covariance Matrix

If you specify the COVB option in the MODEL statement, the procedure displays the estimated covariance matrix. This matrix is defined as the inverse of the information matrix at the final iteration and is based on the Hessian matrix that is used at the final iteration.

Estimated Correlation Matrix

If you specify the CORRB option in the MODEL statement, the procedure displays the estimated correlation matrix, which is based on the Hessian matrix that is used at the final iteration.

Hazard Ratios for *label*

If you specify the HAZARDRATIO statement, PROC ICPHREG outputs the "Hazard Ratios for *label*" table. The table displays the estimate and confidence limits for each hazard ratio. The ODS name of the "Hazard Ratios for *label*" table is HazardRatios.

Parameter Information

If you specify the ITHISTORY, COVB, or CORRB option in the MODEL statement, PROC ICPHREG outputs the "Parameter Information" table. This table displays the names of the parameters and the corresponding level information of effects that contain the CLASS variables. The ODS name of the "Parameter Information" table is ParmInfo.

Constant Hazard Time Intervals

If a piecewise constant model is used (the default model), PROC ICPHREG outputs the "Constant Hazard Time Intervals" table. This table displays information about the parameterization of the baseline function under the piecewise constant model. The ODS name of the "Constant Hazard Time Intervals" table is HazardParms.

Cubic Spline Parameters

If you fit a cubic spline model, PROC ICPHREG outputs the "Cubic Spline Parameters" table. This table displays information about the parameterization of the baseline function under the cubic spline model. The ODS name of the "Cubic Spline Parameters" table is HazardParms.

ODS Table Names

PROC ICPHREG assigns a name to each table it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 70.8. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 70.8 ODS Tables Produced by PROC ICPHREG

ODS Table Name	Description	Statement	Option
ClassLevelInfo	Design information for CLASS variables	CLASS	
ClassLevels	Classification variables levels	CLASS	PARAM=GLM
ConvergenceStatus	Convergence status	MODEL	
CorrB	Parameter estimate correlation matrix	MODEL	CORRB
CovB	Parameter estimate covariance matrix	MODEL	COVB
FitStatistics	Fit statistics	MODEL	
HazardRatios	Customized hazard ratio estimates	HAZARDRATIO	
LastGrad	Last evaluation of the gradient	PROC ICPHREG	ITHISTORY

Table 70.8 *continued*

ODS Table Name	Description	Statement	Option
LastHess	Last evaluation of the Hessian	PROC ICPHREG	ITHISTORY
IterHist	Iteration history	PROC ICPHREG	ITHISTORY
ModelInfo	Model and data information	MODEL	
NObs	Number of observations	MODEL	
OptInfo	Optimization Information	MODEL	
ParameterEstimates	Parameter estimates	MODEL	
ParmInfo	Regression effect names	CLASS	
HazardParms	Baseline parameters	MODEL	

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC ICPHREG assigns a name to each table that it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed separately in Table 70.9. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 70.9 Graphs Produced by PROC ICPHREG

ODS Graph Name	Plot Description	Statement	Option
CumhazPlot	Cumulative hazard function plot	PROC ICPHREG	PLOTS=CUMHAZ
DevianceResidByXBeta	Deviance residuals by linear predictor	PROC ICPHREG	PLOTS=RESDEV(XBETA)
DevianceResPlot	Deviance values	PROC ICPHREG	PLOTS=RESDEV
DiagnosticPlot	Panel of residuals and diagnostic plots	PROC ICPHREG	PLOTS=RESDEV RESLAG
HazardPlot	Hazard function plot	PROC ICPHREG	PLOTS=HAZARD
IntervalPlot	Interval lengths	PROC ICPHREG	PLOTS=INTERVAL
LagakosResidByXBeta	Lagakos residuals by linear predictor	PROC ICPHREG	PLOTS=RESLAG(XBETA)
LagakosResPlot	Lagakos residuals	PROC ICPHREG	PLOTS=RESLAG
SurvivalPlot	Survival function plot	PROC ICPHREG	PLOTS=SURVIVAL

Examples: ICPHREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 70.1: Fitting Cubic Spline Models

(View the complete code for this example (icphgex1.sas) in the example repository.)

This example illustrates how to use a cubic spline baseline hazard to fit a proportional hazards model.

Consider the HIV data set in the section "Getting Started: ICPHREG Procedure" on page 5313. The following statements request a cubic spline proportional hazards model and the hazard ratio between the two levels of the Stage variable.

```
proc icphreg data=hiv;
   class Stage / desc;
   model (Left, Right) = Stage / basehaz=splines;
   hazardratio Stage;
run;
```

Output 70.1.1 displays information about the fitted spline model.

Output 70.1.1 Model Information

The ICPHREG Procedure

Model Information	
Data Set	WORK.HIV
Left Boundary	Left
Right Boundary	Right
Baseline Hazard	Cubic Splines

If no suboption is specified for the spline model, PROC ICPHREG uses three knots, generating three spline coefficients. Output 70.1.2 shows the selected knots.

Output 70.1.2 Cubic Spline Coefficients

Cubic Spline Parameters	
Coefficient	**Knot**
Coef1	1
Coef2	11
Coef3	25

The table of parameter estimates for the spline model is displayed in Output 70.1.3.

Output 70.1.3 Parameter Estimates for the Spline Model

Analysis of Maximum Likelihood Parameter Estimates								
Effect	Stage	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Coef1		1	-6.0630	3.2263	-12.3865	0.2605		
Coef2		1	1.4921	2.2568	-2.9311	5.9152		
Coef3		1	-0.3086	0.6708	-1.6233	1.0060		
Stage	1	1	1.9016	0.6662	0.5959	3.2072	8.15	0.0043
Stage	0	0	0.0000					

Output 70.1.4 shows the estimated hazard ratio between the two stages and the 95% confidence limits.

Output 70.1.4 Hazard Ratio Estimate for Stage Values 1 and 0

Hazard Ratios for Stage			
Description	Point Estimate	95% Wald Confidence Limits	
Stage 1 vs 0	6.697	1.815	24.711

The cubic spline model can be considered a generalization of the Weibull proportional hazards model. It reduces to the Weibull model when there are only two knots, in which case the degrees of freedom is one (DF=1). The Weibull model assumes that the cumulative hazard function is a straight line in the log time scale whereas cubic splines offer a richer set of shapes that have more knots. The following statements fit the spline model with DF=1:

```
proc icphreg data=hiv;
   class Stage / desc;
   model (Left, Right) = Stage / basehaz=splines(df=1);
   hazardratio Stage;
run;
```

The "Fit Statistics" table is displayed in Output 70.1.5.

Output 70.1.5 Fit Statistics for the Spline Model When DF=1

The ICPHREG Procedure

Fit Statistics	
-2 Log Likelihood	30.025
AIC (Smaller is Better)	36.025
AICC (Smaller is Better)	36.914
BIC (Smaller is Better)	40.327

The table of parameter estimates for the fitted spline model is displayed in Output 70.1.6.

Output 70.1.6 Parameter Estimates for the Spline Model When DF=1

Analysis of Maximum Likelihood Parameter Estimates								
Effect	Stage	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Coef1		1	-7.3481	2.4438	-12.1378	-2.5584		
Coef2		1	2.5420	0.8974	0.7831	4.3008		
Stage	1	1	1.8265	0.6132	0.6247	3.0283	8.87	0.0029
Stage	0	0	0.0000					

You can request that PROC LIFEREG fit an accelerated failure lifetime model by using the default distribution (Weibull). This would be equivalent to fitting the proportional hazards model by using a Weibull baseline hazard (Klein and Moeschberger 1997). The following statements fit the Weibull model:

```
proc lifereg data=hiv;
   class Stage;
   model (Left, Right) = Stage;
run;
```

The table of fit statistics is displayed in Output 70.1.7.

Output 70.1.7 Fit Statistics That Are Produced by PROC LIFEREG

The LIFEREG Procedure

Fit Statistics (Unlogged Response)	
-2 Log Likelihood	30.025
Weibull AIC (smaller is better)	36.025
Weibull AICC (smaller is better)	36.914
Weibull BIC (smaller is better)	40.327

The table of parameter estimates for the Weibull model is displayed in Output 70.1.8.

Output 70.1.8 Parameter Estimates That Are Produced PROC LIFEREG

Analysis of Maximum Likelihood Parameter Estimates								
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	2.1722	0.1791	1.8211	2.5233	147.06	<.0001
Stage	0	1	0.7185	0.2711	0.1871	1.2499	7.02	0.0080
Stage	1	0	0.0000	.	.	.	.	.
Scale		1	0.3934	0.1389	0.1969	0.7858		
Weibull Shape		1	2.5420	0.8974	1.2726	5.0776		

Comparing Output 70.1.7 with Output 70.1.5, you can see that the two model fits produce identical likelihood values.

The Weibull shape estimate is equal to the second spline coefficient, but the rest of the parameter estimates are different. This is because PROC LIFEREG fits the Weibull model under the configuration of accelerated failure time models. The estimates of regression coefficients from PROC LIFEREG and PROC ICPHREG

are proportional; their ratio equals the negative of the Weibull shape parameter. For example, the estimate –0.7185 from PROC LIFEREG can also be obtained by dividing the estimate 1.8265 from PROC ICPHREG by –2.5420.

Example 70.2: Plotting Predicted Survival and Cumulative Hazard Functions

(View the complete code for this example (icphgex2.sas) in the example repository.)

This example illustrates how to plot the predicted survival and cumulative hazard functions for specified covariate patterns.

The following statements request a plot of the estimated baseline survival function:

```
ods graphics on;
proc icphreg data=hiv plot=surv;
   class Stage / desc;
   model (Left, Right) = Stage / basehaz=splines;
run;
```

Output 70.2.1 shows the predicted survival curve at the reference level.

Output 70.2.1 Estimated Survival Curve for the Reference Set

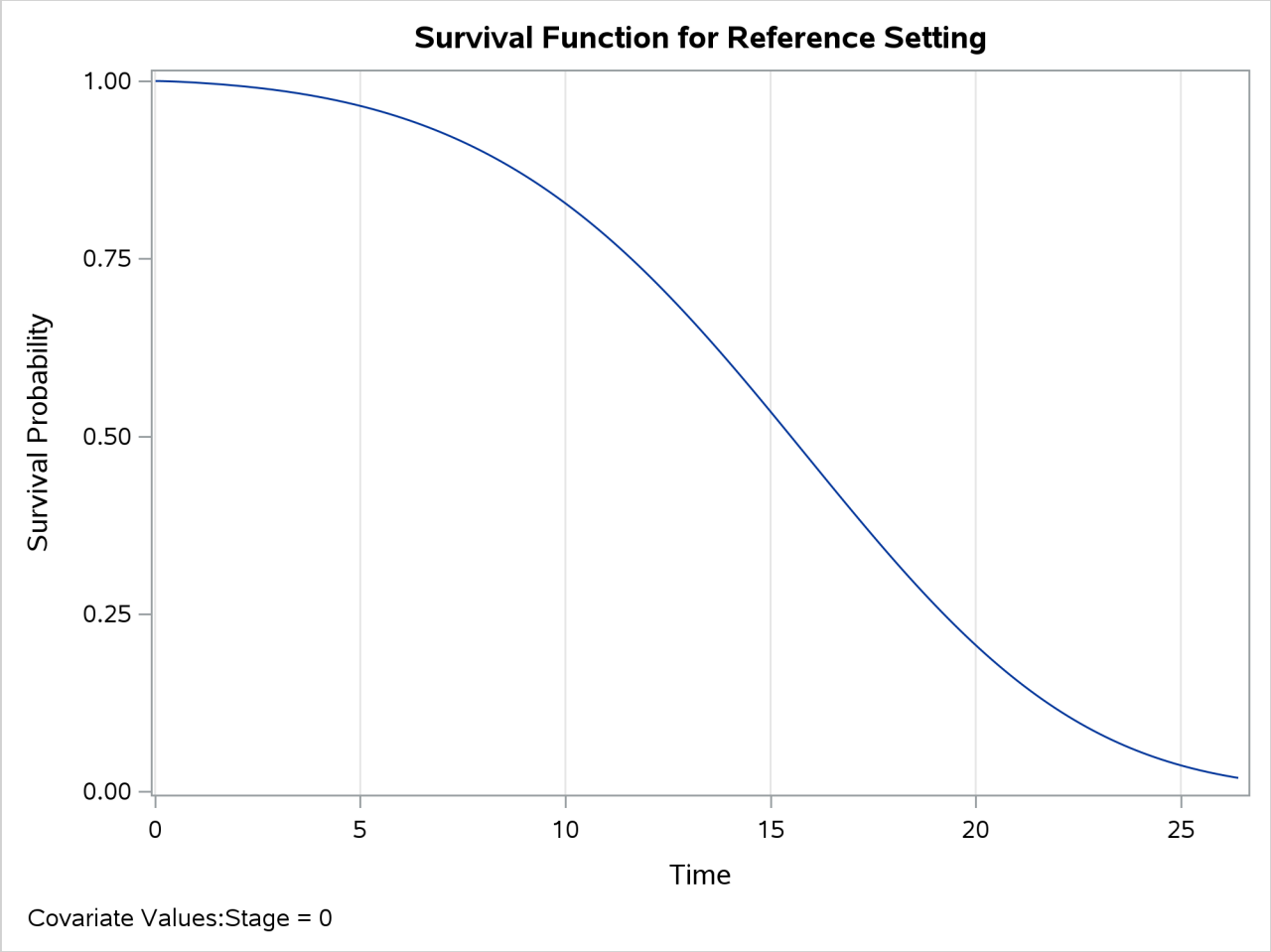

To produce curves for general covariate patterns, you can specify the COVARIATES= option in the BASELINE statement. The following statements create observations for two levels of Stage and plot the corresponding predicted curves:

```
data cov;
   Stage=0; output;
   Stage=1; output;
run;

proc icphreg data=hiv plot=surv;
   class Stage / desc;
   model (Left, Right) = Stage / basehaz=splines;
   baseline covariates=cov;
run;
```

Under the proportional hazards assumption, the two curves do not cross each other. As shown in Output 70.2.2, patients at Stage 1 have much lower survival rates than patients at Stage 0.

Output 70.2.2 Predicted Survival Curves for Specified Covariate Patterns

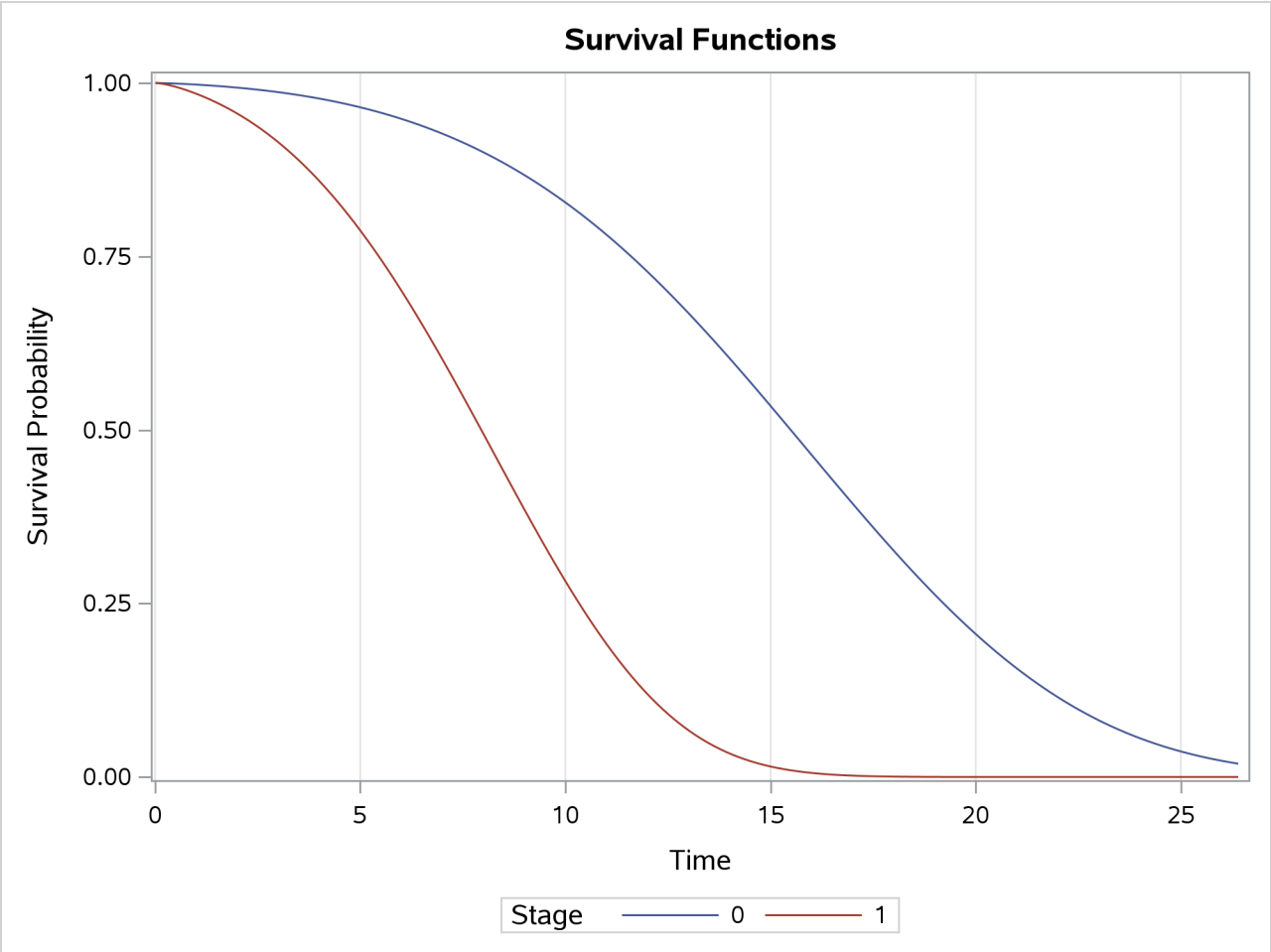

The following statements request a plot of the predicted cumulative hazard functions for the two levels of Stage:

```
proc icphreg data=hiv plot=cumhaz;
   class Stage / desc;
   model (Left, Right) = Stage / basehaz=splines;
   baseline covariates=cov;
run;
```

Output 70.2.3 shows the plot.

Output 70.2.3 Predicted Cumulative Hazards for Specified Covariate Patterns

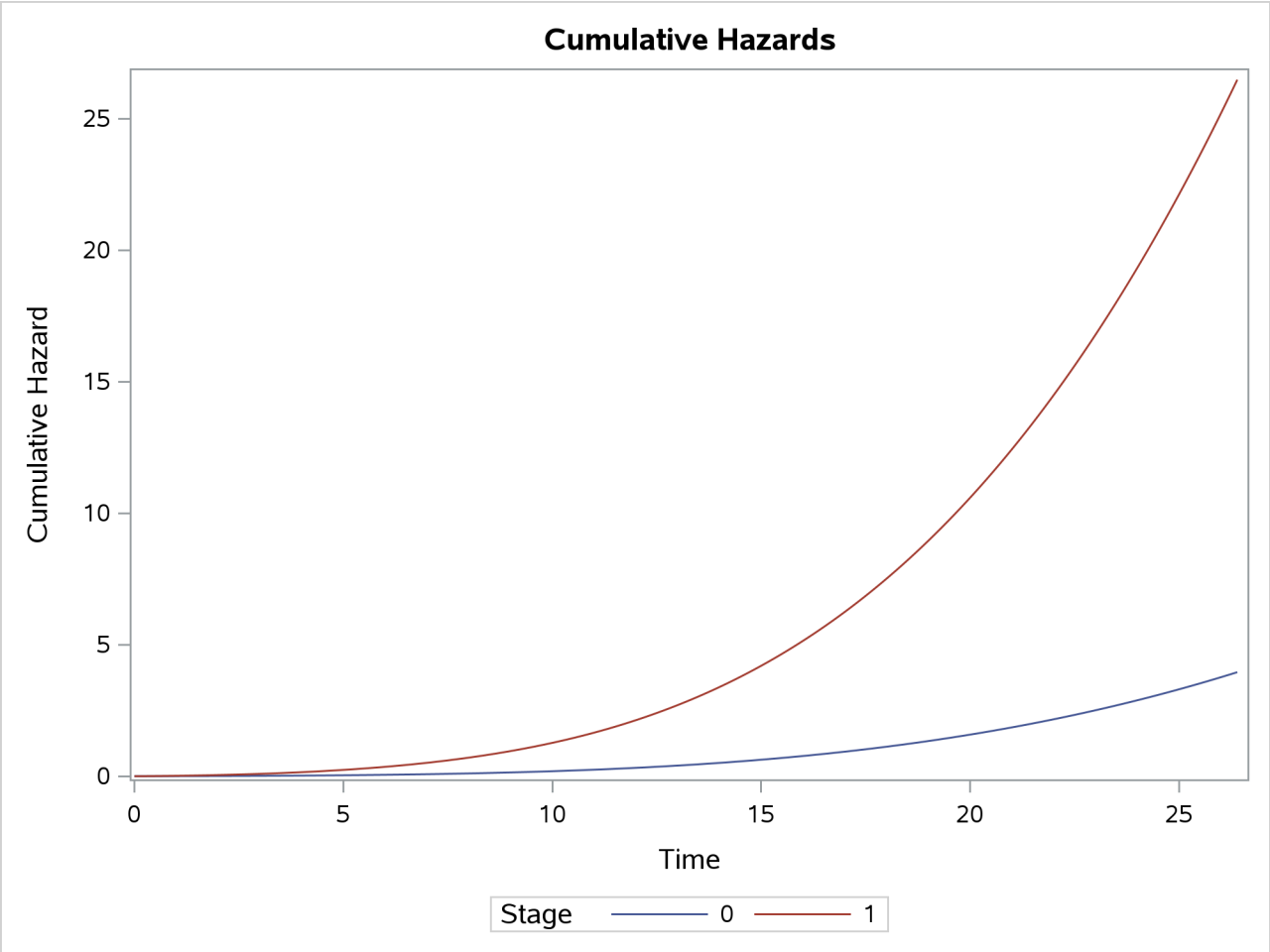

Example 70.3: Fitting Stratified Weibull Models

(View the complete code for this example (icphgex3.sas) in the example repository.)

This example illustrates how to fit stratified Weibull models by using the STRATA statement.

The following data set contains survival times for 36 patients who were diagnosed with a malignant kidney tumor. Most of the patients received the therapy of nephrectomy (removal of all or part of the kidney).

```
data hyper;
   input nephrectomy age time status @@;
   datalines;
0   1    9      1
0   1    6      1
0   1   21      1
0   2   15      1
0   2    8      1
0   2   17      1
0   3   12      1
1   1  104      0
1   1    9      1
1   1   56      1
1   1   35      1
1   1   52      1
1   1   68      1
1   1   77      0
1   1   84      1
1   1    8      1
1   1   38      1
1   1   72      1
1   1   36      1
1   1   48      1
1   1   26      1
1   1  108      1
1   1    5      1
1   2  108      0
1   2   26      1
1   2   14      1
1   2  115      1
1   2   52      1
1   2    5      0
1   2   18      1
1   2   36      1
1   2    9      1
1   3   10      1
1   3    9      1
1   3   18      1
1   3    6      1
;
```

The following statements convert the censored times into intervals:

```
data hyper;
   set hyper;
   left = time;
   if status = 0 then right = .;
   else right = time;
run;
```

The following statements fit a stratified Weibull proportional hazards model:

```
ods graphics on;
proc icphreg data=hyper plot(timerange=(0,125))=surv;
   class Age(desc);
   strata Nephrectomy;
   model (Left, Right) = Age / basehaz=splines(df=1);
run;
```

The "Cubic Splines Parameters" table, shown in Output 70.3.1, contains the parameters for the cubic splines. Because DF=1, it is equivalent to a Weibull distribution.

Output 70.3.1 Baseline Hazard Parameters

The ICPHREG Procedure

Cubic Spline Parameters		
Strata	**Coefficient**	**Knot**
0	**Coef1**	6
0	**Coef2**	21
1	**Coef3**	5
1	**Coef4**	115

The "Fit Statistics" table, shown in Output 70.3.2, contains the fit statistics.

Output 70.3.2 Model Fit Statistics

Fit Statistics	
-2 Log Likelihood	279.422
AIC (Smaller is Better)	291.422
AICC (Smaller is Better)	294.319
BIC (Smaller is Better)	300.923

The table of parameter estimates is displayed in Output 70.3.3.

Output 70.3.3 Model Parameter Estimates

Analysis of Maximum Likelihood Parameter Estimates								
Effect	age	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Coef1		1	-9.2750	2.8327	-14.8269	-3.7231		
Coef2		1	3.3252	0.9918	1.3814	5.2690		
Coef3		1	-5.2408	0.8785	-6.9627	-3.5188		
Coef4		1	1.2861	0.2013	0.8915	1.6806		
age	3	1	1.8005	0.5966	0.6312	2.9697	9.11	0.0025
age	2	1	0.1272	0.4017	-0.6602	0.9146	0.10	0.7515
age	1	0	0.0000					

Output 70.3.4 and Output 70.3.5 show the predicted survival curves for the two strata at the reference level.

Output 70.3.4 Estimated Survival Curve at the Reference Level for Stratum 1

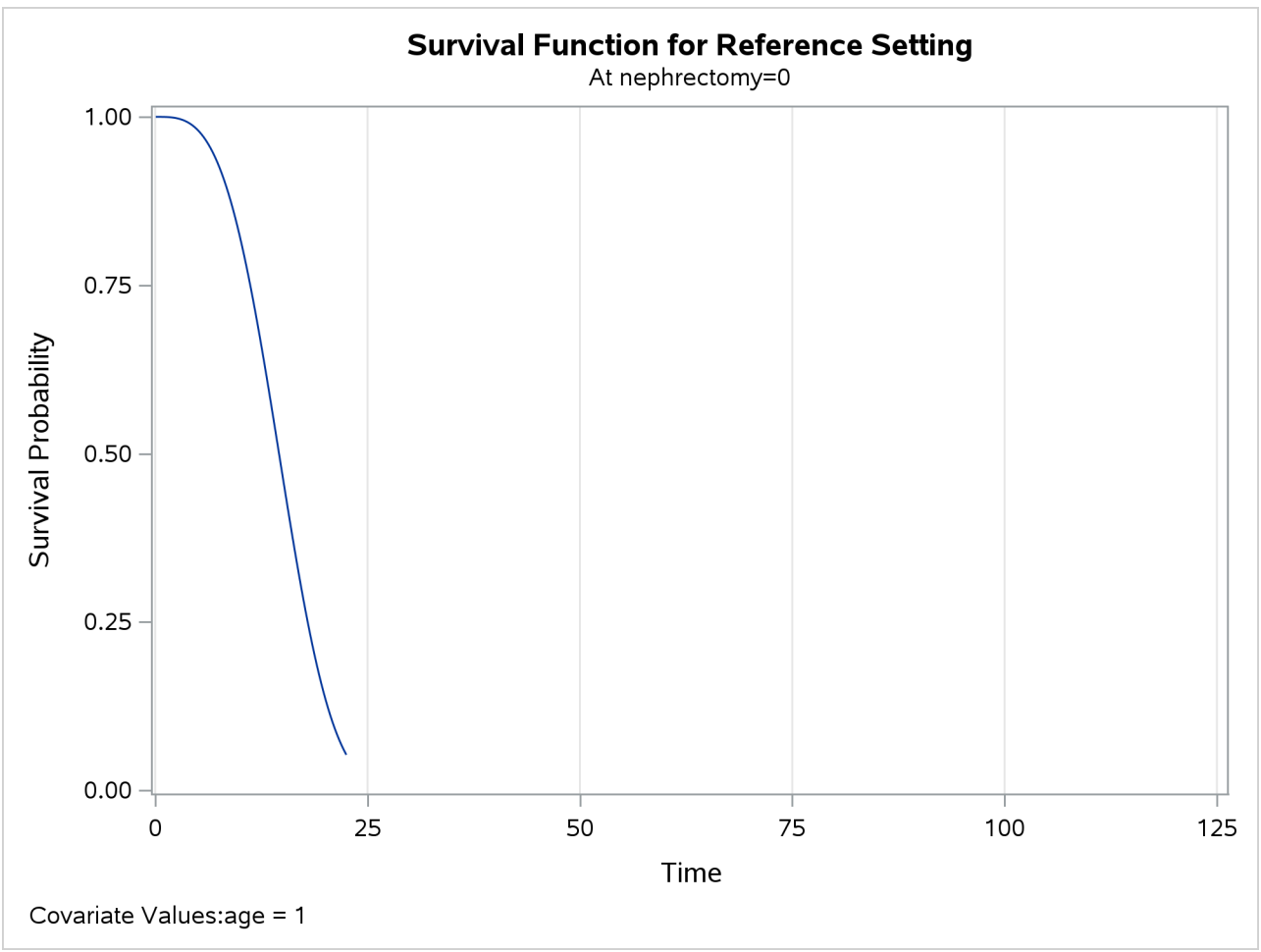

Output 70.3.5 Estimated Survival Curve at the Reference Level for Stratum 2

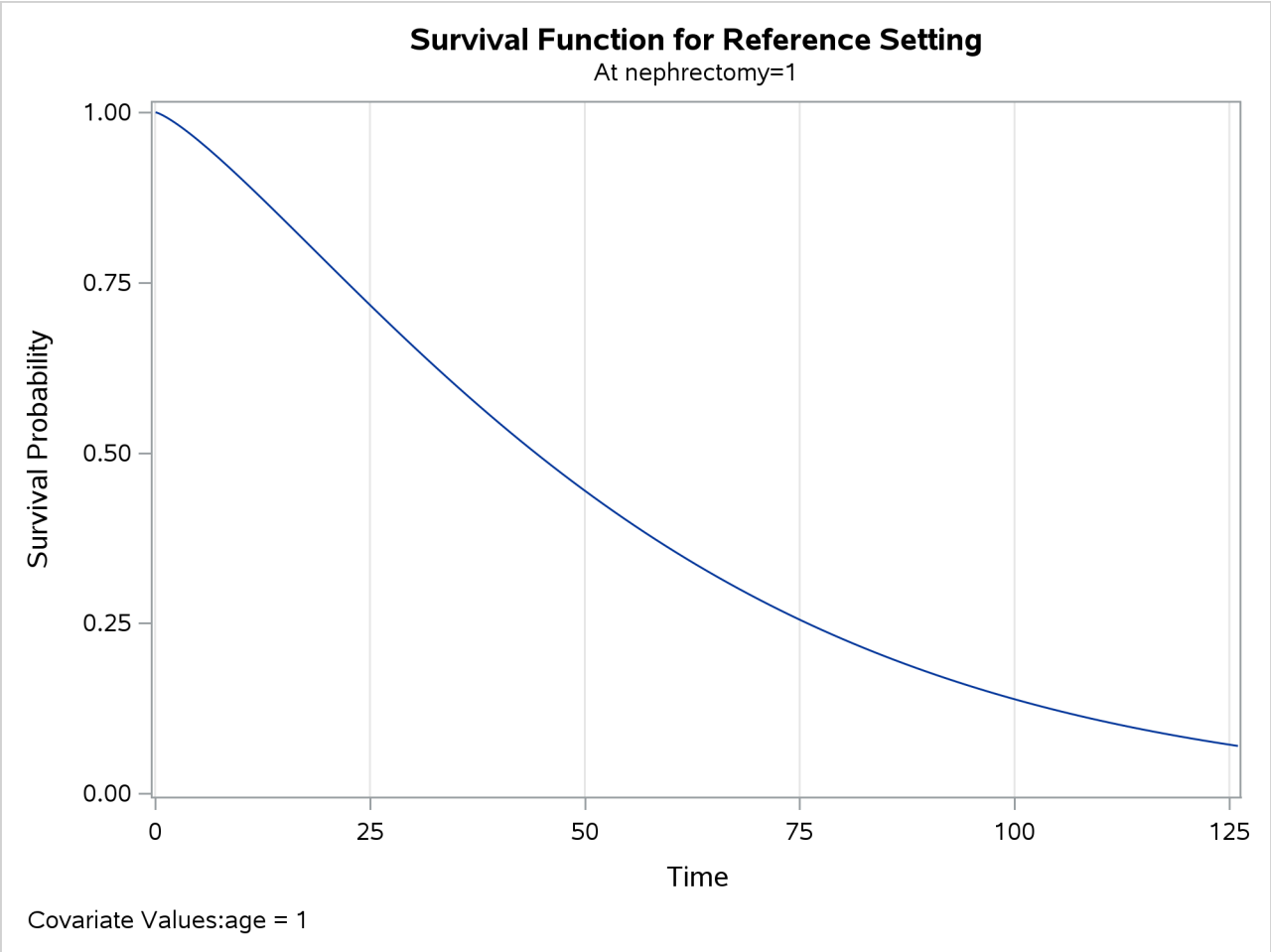

Example 70.4: Fitting Semiparametric Proportional Hazards Models

(View the complete code for this example (icphgex4.sas) in the example repository.)

This example illustrates how to use PROC ICPHREG to fit a semiparametric proportional hazards model.

Gómez et al. (2009) analyzed data from a sample of 4,386 Belgian schoolchildren who were followed up for the emergence times of permanent upper-left first premolars (teeth 24 in European dental notation). The following DATA step creates the SAS data set Tooth:

```
proc format;
   value Sex 0='Male'
             1='Female';
run;

data Tooth;
   input Id Left Right Sex Dmf @@;
   format Sex Sex.;
   datalines;
```

```
   1 2.7 3.5 1 1      2 2.4 3.4 0 1      3 4.5 5.5 1 0      4 5.9   . 1 0
   5 4.1 5.0 1 1      6 3.7 4.5 0 1      7 4.9 5.8 0 1      8 5.4 6.5 1 1
   9 4.0 4.9 1 1     10 5.9 6.7 1 0     11 5.8 6.6 1 1     12 6.0   . 1 0
  13 3.3   . 0 0     14 3.6 4.6 0 1     15 6.1 7.0 0 0     16 2.9 3.7 0 1

   ... more lines ...

4421 2.5   . 0 0   4422 3.9 4.8 1 0   4423 3.0   . 1 0   4424 2.8   . 1 0
4425 3.3 4.3 1 1   4426 4.4 5.2 1 0   4427 2.8   . 1 0   4428 3.5   . 1 0
4429 5.0 6.0 0 0   4430 4.2   . 1 0
;
```

The data set Tooth contains the variables Left and Right, which are the two assessment times, in years since the age of five; Sex, which indicates the child's gender (0 for male or 1 for female); and Dmf, a binary variable that indicates whether the primary predecessor of this tooth was sound (0) or was decayed, missing, or filled (1).

The following statements use PROC ICPHREG to fit a semiparametric proportional hazards model to these data:

```
proc icphreg data=Tooth;
   class Sex;
   model (Left, Right) = Sex Dmf / base=unspecified;
   hazardratio Sex;
run;
```

Output 70.4.1 displays information about the semiparametric proportional hazards model.

Output 70.4.1 Model Information

The ICPHREG Procedure

Model Information	
Data Set	WORK.TOOTH
Left Boundary	Left
Right Boundary	Right
Baseline Hazard	Discrete

Output 70.4.2 displays information about the number of observations in the analysis.

Output 70.4.2 Number of Observations

Number of Observations Read	4386
Number of Observations Used	4386
Right Censored Observations	1611
Interval Censored Observations	2775

As you can see, there are 2,775 interval-censored observations and 1,611 right-censored observations.

Because the baseline function is not specified a priori in the semiparametric model, PROC ICPHREG finds potential jumps in the baseline cumulative hazard function by using the algorithm of Turnbull (1976). Output 70.4.3 shows the identified locations where jumps can occur.

Output 70.4.3 Parameters in the Baseline Cumulative Hazard Function

Hazard Increments		
Interval		
[Lower	Upper)	Hazard Parameter
2.5	2.6	Eta1
2.6	2.7	Eta2
2.7	2.8	Eta3
2.8	2.9	Eta4
2.9	3	Eta5
3	3.1	Eta6
3.1	3.2	Eta7
3.2	3.3	Eta8
3.3	3.4	Eta9
3.4	3.5	Eta10
3.5	3.6	Eta11
3.6	3.7	Eta12
3.7	3.8	Eta13
3.8	3.9	Eta14
3.9	4	Eta15
4	4.1	Eta16
4.1	4.2	Eta17
4.2	4.3	Eta18
4.3	4.4	Eta19
4.4	4.5	Eta20
4.5	4.6	Eta21
4.6	4.7	Eta22
4.7	4.8	Eta23
4.8	4.9	Eta24
4.9	5	Eta25
5	5.1	Eta26
5.1	5.2	Eta27
5.2	5.3	Eta28
5.3	5.4	Eta29
5.4	5.5	Eta30
5.5	5.6	Eta31
5.6	5.7	Eta32
5.7	5.8	Eta33
5.8	5.9	Eta34
5.9	6	Eta35
6	6.1	Eta36
6.1	6.2	Eta37
6.2	6.3	Eta38
6.3	6.4	Eta39
6.4	6.5	Eta40
6.5	6.6	Eta41
6.6	6.7	Eta42
6.7	6.8	Eta43
6.8	6.9	Eta44
6.9	7	Eta45
7	7.1	Eta46

Output 70.4.3 *continued*

Hazard Increments		
Interval		
[Lower	Upper)	Hazard Parameter
7.1	7.2	Eta47
7.2	7.3	Eta48
7.3	7.4	Eta49
7.4	Infty	Eta50

The last parameter is redundant.

Output 70.4.4 displays information about the optimization.

Output 70.4.4 Optimization Information

Optimization Information	
Optimization Technique	EMICM
Parameters in Optimization	51
Fixed Parameters	7

The EMICM algorithm is used to fit the semiparametric model. There are 51 parameters to be estimated, and 7 of them are set to zero during the iterative fitting process.

The table of parameter estimates for the semiparametric model is displayed in Output 70.4.5. It appears that both the child's gender and the previous status of the primary predecessor of the tooth are strongly associated with the outcome of a tooth emergency.

Output 70.4.5 Parameter Estimates for the Semiparametric Model

Analysis of Maximum Likelihood Parameter Estimates

Effect	Sex	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Eta1		1	0.0036	0.0025	0.0000	0.0085		
Eta2		1	0.0042	0.0041	0.0000	0.0122		
Eta3		1	0.0007	0.0046	0.0000	0.0097		
Eta4		1	0.0009	0.0040	0.0000	0.0087		
Eta7		1	0.0028	0.0049	0.0000	0.0124		
Eta8		1	0.0200	0.0078	0.0046	0.0353		
Eta9		1	0.0055	0.0082	0.0000	0.0215		
Eta11		1	0.0158	0.0088	0.0000	0.0330		
Eta12		1	0.0073	0.0109	0.0000	0.0287		
Eta13		1	0.0027	0.0103	0.0000	0.0230		
Eta15		1	0.0030	0.0109	0.0000	0.0244		
Eta16		1	0.0438	0.0120	0.0204	0.0673		
Eta18		1	0.0183	0.0142	0.0000	0.0462		
Eta19		1	0.0178	0.0167	0.0000	0.0506		
Eta20		1	0.0120	0.0180	0.0000	0.0472		
Eta21		1	0.0002	0.0181	0.0000	0.0357		
Eta22		1	0.0397	0.0197	0.0011	0.0783		
Eta23		1	0.0284	0.0233	0.0000	0.0740		
Eta24		1	0.0428	0.0249	0.0000	0.0915		
Eta25		1	0.0520	0.0286	0.0000	0.1080		
Eta26		1	0.0229	0.0312	0.0000	0.0842		
Eta27		1	0.0481	0.0329	0.0000	0.1126		
Eta28		1	0.0357	0.0353	0.0000	0.1048		
Eta29		1	0.0631	0.0366	0.0000	0.1348		
Eta30		1	0.0346	0.0336	0.0000	0.1006		
Eta32		1	0.0607	0.0385	0.0000	0.1362		
Eta33		1	0.0673	0.0482	0.0000	0.1617		
Eta34		1	0.1016	0.0522	0.0000	0.2039		
Eta35		1	0.0609	0.0571	0.0000	0.1729		
Eta36		1	0.0259	0.0601	0.0000	0.1436		
Eta37		1	0.0908	0.0619	0.0000	0.2121		
Eta38		1	0.0283	0.0683	0.0000	0.1622		
Eta39		1	0.0731	0.0732	0.0000	0.2165		
Eta40		1	0.1113	0.0800	0.0000	0.2681		
Eta41		1	0.0553	0.0892	0.0000	0.2301		
Eta42		1	0.1737	0.0873	0.0025	0.3448		
Eta44		1	0.0724	0.1145	0.0000	0.2969		
Eta45		1	0.0882	0.1490	0.0000	0.3802		
Eta46		1	0.1572	0.1918	0.0000	0.5332		
Eta47		1	0.1524	0.2622	0.0000	0.6664		
Eta48		1	0.3081	0.3630	0.0000	1.0196		
Eta49		1	0.3726	0.6425	0.0000	1.6319		
Sex	Female	1	0.3216	0.0387	0.2457	0.3975	69.04	<.0001
Dmf		1	0.3352	0.0387	0.2594	0.4111	75.04	<.0001

Two types of parameters are present in Output 70.4.5: the hazard parameters (Eta1, Eta2, . . . , Eta5) and the regression coefficients for the covariates. If a hazard parameter has been set to zero during the optimization process, or if a column of the Hessian matrix that corresponds to that parameter is found to linearly depend on columns that correspond to preceding model parameters, then PROC ICPHREG does not display that parameter. The procedure does not display the chi-square statistic and associated *p*-value for the hazard parameters.

Output 70.4.6 shows the estimated hazard ratio between boys and girls and the 95% confidence limits. The hazard of tooth emergency for girls is 1.377 times that for boys.

Output 70.4.6 Hazard Ratio Estimate for Girls and Boys

Hazard Ratios for Sex			
Description	Point Estimate	95% Wald Confidence Limits	
Sex Female vs Male	1.379	1.279	1.488

The default method of computing the standard errors is Louis's method (Louis 1982). Alternatively, you can calculate the standard error estimates by using the profile likelihood method. The following statements use PROC ICPHREG to fit the same model and compute the profile-likelihood-based standard errors:

```
proc icphreg data=Tooth;
   class Sex;
   model (Left, Right) = Sex Dmf / base=unspecified plvariance;
   hazardratio Sex;
run;
```

Output 70.4.7 shows the estimated hazard ratio between boys and girls and the 95% confidence limits.

Output 70.4.7 Hazard Ratio Estimate for Girls and Boys

The ICPHREG Procedure

Hazard Ratios for Sex			
Description	Point Estimate	95% Wald Confidence Limits	
Sex Female vs Male	1.379	1.279	1.488

The hazard ratio and standard errors based on the profile likelihood are almost identical to the previous ones, which were computed by Louis's method.

Example 70.5: Fitting Semiparametric Models with Time-Dependent Covariates

(View the complete code for this example (icphgex5.sas) in the example repository.)

This example illustrates how to use PROC ICPHREG to fit a semiparametric proportional hazards model that contains time-dependent covariates.

Sparling et al. (2006) analyzed data from a long-term study about the effect of glucose exposure on retinopathy in young diabetes patients. The blood protein hemoglobin A1c (HbA1c) measures the level of glucose exposure (glycemia) over the preceding 68 weeks. More than a thousand patients participated in the study. Researchers measured their level of HbA1c at initial screening and their mean level of HbA1c during the study. Patients were assessed multiple times for retinopathy over a period of four years.

The following DATA step creates the data set Retinopathy:

```
data Retinopathy;
   input Duration Id Status CurrentHbA Time InitHbA PreHbA PreTime PreRetin;
   datalines;
   178       1     0     7.7500   988   9.63   7.9954   114    0
   178       1     0     7.7250  1618   9.63   7.9954   114    0
   178       1     0     7.9400  1956   9.63   7.9954   114    0
   175       3     0     8.8000  1111   7.93   8.1532   108    0
   175       3     0     8.6750  1859   7.93   8.1532   108    0
    72       5     1     8.2500  1027   8.53   7.8703   108    0
   106       6     0     8.2000  1110   7.26   8.5547   108    0
   106       6     0     8.5000  1838   7.26   8.5547   108    0
   147       8     0     8.3000  1202   9.27   6.8011   108    0
   147       8     0     8.3500  1834   9.27   6.8011   108    0

   ... more lines ...

    39    1438     0     8.6500  1610   7.30   8.0938    42    1
    30    1439     0     8.6750  1647   9.30   9.2750    42    1
    22    1440     0     8.0000  1702   6.60   6.9733    42    1
;
```

The variable Time represents the assessment time in days. The variable Status indicates whether retinopathy occurred before the associated assessment time. The variable Id identifies the subjects. The variable Duration represents how long the subject has had diabetes, in months. The variable PreHbA represents the mean HbA1c level in a previous study. The variable CurrentHbA represents the current mean HbA1c level at each assessment. The variable InitHbA represents the HbA1c level at initial screening. The variable PreTime represents the time in months that the subject spent in the previous study. The binary variable PreRetin indicates whether retinopathy was present prior to the current study.

The following statements use PROC ICPHREG to fit a semiparametric proportional hazards model to these data:

```
proc icphreg data=Retinopathy;
   model Time*Status(0) = Duration PreRetin InitHbA PreHbA PreTime CurrentHbA
                        / base=unspecified plvariance;
   id id;
   hazardratio CurrentHbA;
```

```
run;
```

The PLVARIANCE option computes the standard error estimates of the regression coefficients on the basis of the profile likelihood function (Zeng, Mao, and Lin 2016). The ID statement identifies the variable that represents the study subjects. The HAZARDRATIO statement displays hazard ratios for the current HbA1c level.

Output 70.5.1 displays information about the semiparametric proportional hazards model.

Output 70.5.1 Model Information

The ICPHREG Procedure

Model Information	
Data Set	WORK.RETINOPATHY
Dependent Variable	Time
Censoring Variable	Status
Censoring Value(s)	0
Baseline Hazard	Discrete

Output 70.5.2 displays the levels of the ID variable.

Output 70.5.2 Subject Identification Values

Class Level Information		
Name	**Levels**	**Values**
Id	1315	1 3 5 6 8 9 11 14 15 16 17 18 19 20 21 22 24 25 27 28 30 32 35 36 37 38 39 40 42 43 44 45 46 47 48 49 50 51 52 54 55 56 57 58 59 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 77 78 79 80 81 82 83 84 86 87 88 89 90 91 92 93 94 95 96 97 98 99 101 102 103 ...

Output 70.5.3 displays information about the number of observations in the analysis.

Output 70.5.3 Number of Observations

Number of Observations Read	2615
Number of Observations Used	2615
Right Censored Observations	1085
Interval Censored Observations	105
Left Censored Observations	125

As you can see, 1,085 patients are right-censored, 105 patients are interval-censored, and 125 patients are left-censored.

Output 70.5.4 displays information about the optimization.

Output 70.5.4 Optimization Information

Optimization Information	
Optimization Technique	EMICM
Parameters in Optimization	181
Fixed Parameters	155

The EMICM algorithm is used to fit the semiparametric model. There are 181 parameters that need to be estimated, and 155 of them are set to zero during the iterative fitting process.

The table of parameter estimates for the semiparametric model is displayed in Output 70.5.5.

Output 70.5.5 Parameter Estimates for the Semiparametric Model

Analysis of Maximum Likelihood Parameter Estimates							
Effect	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Duration	1	0.0039	0.0020	0.0000	0.0077	3.90	0.0484
PreRetin	1	-0.2739	0.1948	-0.6556	0.1079	1.98	0.1597
InitHbA	1	-0.0845	0.0451	-0.1729	0.0039	3.51	0.0611
PreHbA	1	0.6918	0.0617	0.5709	0.8126	125.86	<.0001
PreTime	1	0.0062	0.0036	-0.0009	0.0133	2.94	0.0864
CurrentHbA	1	0.1262	0.0530	0.0223	0.2301	5.67	0.0173

Output 70.5.6 shows the estimated hazard ratio for the current HbA1c level and the 95% confidence limits.

Output 70.5.6 Hazard Ratio Estimate for HbA1c

Hazard Ratios for CurrentHbA			
Description	Point Estimate	95% Wald Confidence Limits	
CurrentHbA Unit=1	1.135	1.023	1.259

For the time-dependent variable HbA1c, the 95% confidence interval for the hazard ratio is above 1. This indicates that a higher level of current HbA1c is associated with an increased risk of retinopathy.

References

Akaike, H. (1979). "A Bayesian Extension of the Minimum AIC Procedure of Autoregressive Model Fitting." *Biometrika* 66:237–242.

Akaike, H. (1981). "Likelihood of a Model and Information Criteria." *Journal of Econometrics* 16:3–14.

Cai, T., and Betensky, R. A. (2003). "Hazard Regression for Interval-Censored Data with Penalized Spline." *Biometrics* 59:570–579. http://dx.doi.org/10.1111/1541-0420.00067.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220. With discussion.

Farrington, C. P. (2000). "Residuals for Proportional Hazards Models with Interval-Censored Survival Data." *Biometrics* 56:473–482.

Finkelstein, D. M. (1986). "A Proportional Hazards Model for Interval-Censored Failure Time Data." *Biometrics* 42:845–854.

Friedman, M. (1982). "Piecewise Exponential Models for Survival Data with Covariates." *Annals of Statistics* 10:101–113.

Gómez, G., Calle, M. L., Oller, R., and Langohr, K. (2009). "Tutorial on Methods for Interval-Censored Data and Their Implementation in R." *Statistical Modelling* 9:259–297.

Groeneboom, P., and Wellner, J. A. (1992). *Information Bounds and Nonparametric Maximum Likelihood Estimation*. Basel: Birkhäuser.

Ibrahim, J. G., Chen, M.-H., and Sinha, D. (2001). *Bayesian Survival Analysis*. New York: Springer-Verlag.

Klein, J. P., and Moeschberger, M. L. (1997). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: Springer-Verlag.

Larder, B. A., Darby, G., and Richman, D. D. (1989). "HIV with Reduced Sensitivity to Zidovudine (AZT) Isolated during Prolonged Therapy." *Science* 243:1731–1734.

Louis, T. A. (1982). "Finding the Observed Information Matrix When Using the EM Algorithm." *Journal of the Royal Statistical Society, Series B* 44:226–233.

Murphy, S. A., and Van Der Vaart, A. W. (2000). "On Profile Likelihood." *Journal of the American Statistical Association* 95:449–465.

Pan, W. (1999). "Extending the Iterative Convex Minorant Algorithm to the Cox Model for Interval-Censored Data." *Journal of Computational and Graphical Statistics* 8:109–120.

Royston, P., and Parmar, M. K. B. (2002). "Flexible Parametric Proportional-Hazards and Proportional-Odds Models for Censored Survival Data, with Application to Prognostic Modelling and Estimation of Treatment Effects." *Statistics in Medicine* 21:2175–2197.

Simonoff, J. S. (2003). *Analyzing Categorical Data*. New York: Springer-Verlag.

Sparling, Y. H., Younes, N., Lachin, J. M., and Bautista, O. M. (2006). "Parametric Survival Models for Interval-Censored Data with Time-Dependent Covariates." *Biostatistics* 7:599–614.

Turnbull, B. W. (1976). "The Empirical Distribution Function with Arbitrarily Grouped, Censored, and Truncated Data." *Journal of the Royal Statistical Society, Series B* 38:290–295.

Wang, L., McMahan, C. S., Hudgens, M. G., and Qureshi, Z. P. (2016). "A Flexible, Computationally Efficient Method for Fitting the Proportional Hazards Model to Interval-Censored Data." *Biometrics* 72:222–231.

Zeng, D., Mao, L., and Lin, D. Y. (2016). "Maximum Likelihood Estimation for Semiparametric Transformation Models with Interval-Censored Data." *Biometrika* 103:253–271.

Chapter 71
The INBREED Procedure

Contents

Overview: INBREED Procedure **5382**
Getting Started: INBREED Procedure **5382**
The Format of the Input Data Set 5382
Performing the Analysis 5384
Syntax: INBREED Procedure **5386**
PROC INBREED Statement 5386
BY Statement 5388
CLASS Statement 5389
GENDER Statement 5389
MATINGS Statement 5389
VAR Statement 5390
Details: INBREED Procedure **5390**
Missing Values 5390
DATA= Data Set 5391
Computational Details 5392
OUTCOV= Data Set 5397
Displayed Output 5399
ODS Table Names 5399
Examples: INBREED Procedure **5400**
Example 71.1: Monoecious Population Analysis 5400
Example 71.2: Pedigree Analysis 5402
Example 71.3: Pedigree Analysis with BY Groups 5404
References **5405**

Overview: INBREED Procedure

The INBREED procedure calculates the covariance or inbreeding coefficients for a pedigree. PROC INBREED is unique in that it handles very large populations.

The INBREED procedure has two modes of operation. One mode carries out analysis on the assumption that all the individuals belong to the same generation. The other mode divides the population into nonoverlapping generations and analyzes each generation separately, assuming that the parents of individuals in the current generation are defined in the previous generation.

PROC INBREED also computes averages of the covariance or inbreeding coefficients within sex categories if the sex of individuals is known.

Getting Started: INBREED Procedure

This section demonstrates how you can use the INBREED procedure to calculate the inbreeding or covariance coefficients for a pedigree, how you can control the analysis mode if the population consists of nonoverlapping generations, and how you can obtain averages within sex categories.

For you to use PROC INBREED effectively, your input data set must have a definite format. The following sections first introduce this format for a fictitious population and then demonstrate how you can analyze this population by using the INBREED procedure.

The Format of the Input Data Set

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/inbrgs1.sas.)

The SAS data set used as input to the INBREED procedure must contain an observation for each individual. Each observation must include one variable identifying the individual and two variables identifying the individual's parents. Optionally, an observation can contain a known covariance coefficient and a character variable defining the gender of the individual.

For example, consider the following data:

```
data Population;
   input Individual $ Parent1 $ Parent2 $
         Covariance Sex $ Generation;
   datalines;
Mark   George Lisa    .    M  1
Kelly  Scott  Lisa    .    F  1
Mike   George Amy     .    M  1
.      Mark   Kelly  0.50  .  1
David  Mark   Kelly   .    M  2
Merle  Mike   Jane    .    F  2
Jim    Mark   Kelly  0.50  M  2
Mark   Mike   Kelly   .    M  2
```

```
;
```

It is important to order the pedigree observations so that individuals are defined before they are used as parents of other individuals. The family relationships between individuals cannot be ascertained correctly unless you observe this ordering. Also, older individuals must precede younger ones. For example, 'Mark' appears as the first parent of 'David' at observation 5; therefore, his observation needs to be defined prior to observation 5. Indeed, this is the case (see observation 1). Also, 'David' is older than 'Jim', whose observation appears after the observation for 'David', as is appropriate.

In populations with distinct, nonoverlapping generations, the older generation (parents) must precede the younger generation. For example, the individuals defined in Generation=1 appear as parents of individuals defined in Generation=2.

PROC INBREED produces warning messages when a parent cannot be found. For example, 'Jane' appears as the second parent of the individual 'Merle' even though there are no previous observations defining her own parents. If the population is treated as an overlapping population, that is, if the generation grouping is ignored, then the procedure inserts an observation for 'Jane' with missing parents just before the sixth observation, which defines 'Merle' as follows:

```
Jane   .      .       .    F  2
Merle  Mike   Jane    .    F  2
```

However, if generation grouping is taken into consideration, then 'Jane' is defined as the last observation in Generation=1, as follows:

```
Mike   George Amy     .    M  1
Jane   .      .       .    F  1
```

In this latter case, however, the observation for 'Jane' is inserted after the computations are reported for the first generation. Therefore, she does not appear in the covariance/inbreeding matrix, even though her observation is used in computations for the second generation (see Figure 71.2).

If the data for an individual are duplicated, only the first occurrence of the data is used by the procedure, and a warning message is displayed to note the duplication. For example, individual 'Mark' is defined twice, at observations 1 and 8. If generation grouping is ignored, then this is an error and observation 8 is skipped. However, if the population is processed with respect to two distinct generations, then 'Mark' refers to two different individuals, one in Generation=1 and the other in Generation=2.

If a covariance is to be assigned between two individuals, then those individuals must be defined prior to the assignment observation. For example, a covariance of 0.50 can be assigned between 'Mark' and 'Kelly' since they are previously defined. Note that assignment statements must have different formats depending on whether the population is processed with respect to generations (see the section "DATA= Data Set" on page 5391 for further information). For example, while observation 4 is valid for nonoverlapping generations, it is invalid for a processing mode that ignores generation grouping. In this latter case, observation 7 indicates a valid assignment, and observation 4 is skipped.

The latest covariance specification between any given two individuals overrides the previous one between the same individuals.

Performing the Analysis

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/inbrgs1.sas.)

To compute the covariance coefficients for the overlapping generation mode, use the following statements:

```
proc inbreed data=Population covar matrix init=0.25;
run;
```

Here, the DATA= option names the SAS data set to be analyzed, and the COVAR and MATRIX options tell the procedure to output the covariance coefficients matrix. If you omit the COVAR option, the inbreeding coefficients are output instead of the covariance coefficients.

Note that the PROC INBREED statement also contains the INIT= option. This option gives an initial covariance between any individual and unknown individuals. For example, the covariance between any individual and 'Jane' would be 0.25, since 'Jane' is unknown, except when 'Jane' appears as a parent (see Figure 71.4).

Figure 71.1 Analysis for an Overlapping Population

The INBREED Procedure

			Covariance Coefficients										
Individual	**Parent1**	**Parent2**	**George**	**Lisa**	**Mark**	**Scott**	**Kelly**	**Amy**	**Mike**	**David**	**Jane**	**Merle**	**Jim**
George			1.1250	0.2500	0.6875	0.2500	0.2500	0.2500	0.6875	0.4688	0.2500	0.4688	0.4688
Lisa			0.2500	1.1250	0.6875	0.2500	0.6875	0.2500	0.2500	0.6875	0.2500	0.2500	0.6875
Mark	**George**	**Lisa**	0.6875	0.6875	1.1250	0.2500	0.5000	0.2500	0.4688	0.8125	0.2500	0.3594	0.8125
Scott			0.2500	0.2500	0.2500	1.1250	0.6875	0.2500	0.2500	0.4688	0.2500	0.2500	0.4688
Kelly	**Scott**	**Lisa**	0.2500	0.6875	0.5000	0.6875	1.1250	0.2500	0.2500	0.8125	0.2500	0.2500	0.8125
Amy			0.2500	0.2500	0.2500	0.2500	0.2500	1.1250	0.6875	0.2500	0.2500	0.4688	0.2500
Mike	**George**	**Amy**	0.6875	0.2500	0.4688	0.2500	0.2500	0.6875	1.1250	0.3594	0.2500	0.6875	0.3594
David	**Mark**	**Kelly**	0.4688	0.6875	0.8125	0.4688	0.8125	0.2500	0.3594	1.2500	0.2500	0.3047	0.8125
Jane			0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	1.1250	0.6875	0.2500
Merle	**Mike**	**Jane**	0.4688	0.2500	0.3594	0.2500	0.2500	0.4688	0.6875	0.3047	0.6875	1.1250	0.3047
Jim	**Mark**	**Kelly**	0.4688	0.6875	0.8125	0.4688	0.8125	0.2500	0.3594	0.8125	0.2500	0.3047	1.2500

Number of Individuals	11

In the previous example, PROC INBREED treats the population as a single generation. However, you might want to process the population with respect to distinct, nonoverlapping generations. To accomplish this, you need to identify the generation variable in a CLASS statement, as shown by the following statements:

```
proc inbreed data=Population covar matrix init=0.25;
   class Generation;
run;
```

Note that, in this case, the covariance matrix is displayed separately for each generation (see Figure 71.5).

Figure 71.2 Analysis for a Nonoverlapping Population

The INBREED Procedure

Generation = 1

Covariance Coefficients					
Individual	**Parent1**	**Parent2**	**Mark**	**Kelly**	**Mike**
Mark	**George**	**Lisa**	1.1250	0.5000	0.4688
Kelly	**Scott**	**Lisa**	0.5000	1.1250	0.2500
Mike	**George**	**Amy**	0.4688	0.2500	1.1250

Number of Individuals	3

The INBREED Procedure

Generation = 2

Covariance Coefficients						
Individual	**Parent1**	**Parent2**	**David**	**Merle**	**Jim**	**Mark**
David	**Mark**	**Kelly**	1.2500	0.3047	0.8125	0.5859
Merle	**Mike**	**Jane**	0.3047	1.1250	0.3047	0.4688
Jim	**Mark**	**Kelly**	0.8125	0.3047	1.2500	0.5859
Mark	**Mike**	**Kelly**	0.5859	0.4688	0.5859	1.1250

Number of Individuals	4

You might also want to see covariance coefficient averages within sex categories. This is accomplished by indicating the variable defining the gender of individuals in a GENDER statement and by adding the AVERAGE option to the PROC INBREED statement. For example, the following statements produce the covariance coefficient averages shown in Figure 71.3:

```
proc inbreed data=Population covar average init=0.25;
   class Generation;
   gender Sex;
run;
```

Figure 71.3 Averages within Sex Categories for a Nonoverlapping Generation

The INBREED Procedure

Generation = 1

Averages of Covariance Coefficient Matrix in Generation 1		
	On Diagonal	**Below Diagonal**
Male X Male	1.1250	0.4688
Male X Female	.	0.3750
Female X Female	1.1250	0.0000
Over Sex	1.1250	0.4063

Figure 71.3 *continued*

Number of Males	2
Number of Females	1
Number of Individuals	3

The INBREED Procedure

Generation = 2

Averages of Covariance Coefficient Matrix in Generation 2		
	On Diagonal	**Below Diagonal**
Male X Male	1.2083	0.6615
Male X Female	.	0.3594
Female X Female	1.1250	0.0000
Over Sex	1.1875	0.5104

Number of Males	3
Number of Females	1
Number of Individuals	4

Syntax: INBREED Procedure

The following statements are available in the INBREED procedure:

```
PROC INBREED <options> ;
    BY variables ;
    CLASS variable ;
    GENDER variable ;
    MATINGS individual-list1 / mate-list1 <, ..., individual-listn / mate-listn> ;
    VAR variables ;
```

The PROC INBREED statement is required. Items within angle brackets (< >) are optional. The syntax of each statement is described in the following sections.

PROC INBREED Statement

PROC INBREED < *options* > ;

The PROC INBREED statement invokes the INBREED procedure. Table 71.1 summarizes the options available in the PROC INBREED statement.

Table 71.1 PROC INBREED Statement Options

Option	Description
Specify Data Sets	
DATA=	Names the SAS data set
OUTCOV=	Names an output data set to contain the inbreeding coefficients
Control Type of Coefficient	
COVAR	Specifies that all coefficients output consist of covariance coefficients
SELFDIAG	Includes an individual's self-mating kinship coefficient
Control Displayed Tables	
AVERAGE	Produces a table of averages of coefficients
IND	Displays the individuals' inbreeding coefficients
MATRIX	Displays the inbreeding coefficient matrix
Specify Default Covariance Value	
INIT=	Specifies the covariance value
Suppress Output	
INDL	Displays individuals' coefficients for only the last generation
MATRIXL	Displays coefficients for only the last generation
NOPRINT	Suppresses the display of all output

AVERAGE

A

produces a table of averages of coefficients for each pedigree of offspring. The AVERAGE option is used together with the GENDER statement to average the inbreeding/covariance coefficients within sex categories.

COVAR

C

specifies that all coefficients output consist of covariance coefficients rather than inbreeding coefficients.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC INBREED. If you omit the DATA= option, the most recently created SAS data set is used.

IND

I

displays the individuals' inbreeding coefficients (diagonal of the inbreeding coefficients matrix) for each pedigree of offspring.

If you also specify the COVAR option, the individuals' covariance coefficients (diagonal of the covariance coefficients matrix) are displayed.

INDL
displays individuals' coefficients for only the last generation of a multiparous population.

INIT=*cov*
specifies the covariance value *cov* if any of the parents are unknown; a value of 0 is assumed if you do not specify the INIT= option.

MATRIX
M
displays the inbreeding coefficient matrix for each pedigree of offspring.

If you also specify the COVAR option, the covariance matrices are displayed instead of inbreeding coefficients matrices.

MATRIXL
displays coefficients for only the last generation of a multiparous population.

NOPRINT
suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS). For more information on ODS, see Chapter 23, "Using the Output Delivery System."

OUTCOV=*SAS-data-set*
names an output data set to contain the inbreeding coefficients. When the COVAR option is also specified, covariance estimates are output to the OUTCOV= data set instead of inbreeding coefficients.

SELFDIAG
includes an individual's self-mating kinship coefficient instead of the individual's inbreeding coefficient on the diagonal of the matrix in the OUTCOV= data set when the COVAR option is not specified.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC INBREED to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the INBREED procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* **;**

To analyze the population within nonoverlapping generations, you must specify the variable that identifies generations in a CLASS statement. Values of the generation variable, called *generation numbers*, must be integers, but generations are assumed to occur in the order of their input in the input data set rather than in numerical order of the generation numbers. The name of an individual needs to be unique only within its generation.

When the MATRIXL option or the INDL option is specified, each generation requires a unique generation number in order for the specified option to work correctly. If generation numbers are not unique, all the generations with a generation number that is the same as the last generation's are output.

GENDER Statement

GENDER *variable* **;**

The GENDER statement specifies a variable that indicates the sex of the individuals. Values of the sex variable must be character beginning with 'M' or 'F', for male or female. The GENDER statement is needed only when you specify the AVERAGE option to average the inbreeding/covariance coefficients within sex categories or when you want to include a gender variable in the OUTCOV= data set.

PROC INBREED makes the following assumptions regarding the gender of individuals:

- The first parent is always assumed to be the male. See the section "VAR Statement" on page 5390.
- The second parent is always assumed to be the female. See the section "VAR Statement" on page 5390.
- If the gender of an individual is missing or invalid, this individual is assumed to be a female unless the population is overlapping and this individual appears as the first parent in a later observation.

Any contradictions to these rules are reported in the SAS log.

MATINGS Statement

MATINGS *individual-list1 / mate-list1* <**,** . . . **,** *individual-listn / mate-listn*> **;**

You can specify the MATINGS statement with PROC INBREED to specify selected matings of individuals. Each individual given in *individual-list* is mated with each individual given in *mate-list*. You can write multiple mating specifications if you separate them by commas or asterisks. The procedure reports the inbreeding coefficients or covariances for each pair of mates. For example, you can use the following statement to specify the mating of an individual named 'David' with an individual named 'Jane':

```
matings david / jane;
```

VAR Statement

VAR *individual parent1 parent2* < *covariance* > **;**

The VAR statement specifies three or four variables: the first variable contains an individual's name, the second variable contains the name of the individual's first parent, and the third variable contains the name of the individual's second parent. An optional fourth variable assigns a known value to the covariance of the individual's first and second parents in the current generation.

The first three variables in the VAR statement can be either numeric or character; however, only the first 12 characters of a character variable are recognized by the procedure. The fourth variable, if specified, must be numeric.

If you omit the VAR statement, then the procedure uses the first three unaddressed variables as the names of the individual and its parents. (Unaddressed variables are those that are not referenced in any other PROC INBREED statement.) If the input data set contains an unaddressed fourth variable, then it becomes the covariance variable.

Details: INBREED Procedure

Missing Values

A missing value for a parent implies that the parent is unknown. Unknown parents are assumed to be unrelated and not inbred unless you specify the INIT= option.

When the value of the variable identifying the individual is missing, the observation is not added to the list of individuals. However, for a multiparous population, an observation with a missing individual is valid and is used for assigning covariances.

Missing covariance values are determined from the INIT=*cov* option, if specified. Observations with missing generation variables are excluded.

If the gender of an individual is missing, it is determined from the order in which it is listed on the first observation defining its progeny for an overlapping population. If it appears as the first parent, it is set to 'M'; otherwise, it is set to 'F'. When the gender of an individual cannot be determined, it is assigned a default value of 'F'.

DATA= Data Set

Each observation in the input data set should contain necessary information such as the identification of an individual and the first and second parents of an individual. In addition, if a CLASS statement is specified, each observation should contain the generation identification; and, if a GENDER statement is specified, each observation should contain the gender of an individual. Optionally, each observation might also contain the covariance between the first and the second parents. Depending on how many statements are specified with the procedure, there should be enough variables in the input data set containing this information.

If you omit the VAR statement, then the procedure uses the first three *unaddressed variables* in the input data set as the names of the individual and his or her parents. Unaddressed variables in the input data set are those variables that are not referenced by the procedure in any other statements, such as CLASS, GENDER, or BY statements. If the input data set contains an unaddressed fourth variable, then the procedure uses it as the covariance variable.

If the individuals given by the variables associated with the first and second parents are not in the population, they are added to the population. However, if they are in the population, they must be defined prior to the observation that gives their progeny.

When there is a CLASS statement, the functions of defining new individuals and assigning covariances must be separated. This is necessary because the parents of any given individual are defined in the previous generation, while covariances are assigned between individuals in the current generation.

Therefore, there could be two types of observations for a multiparous population:

- one to define new individuals in the current generation whose parents have been defined in the previous generation, as in the following, where the missing value is for the covariance variable:

```
Mark   George Lisa    .    M  1
Kelly  Scott  Lisa    .    F  1
```

- one to assign covariances between two individuals in the current generation, as in the following, where the individual's name is missing, 'Mark' and 'Kelly' are in the current generation, and the covariance coefficient between these two individuals is 0.50:

```
.      Mark   Kelly  0.50  .  1
```

Note that the observations defining individuals must precede the observation assigning a covariance value between them. For example, if a covariance is to be assigned between 'Mark' and 'Kelly', then both of them should be defined prior to the assignment observation.

Computational Details

This section describes the rules that the INBREED procedure uses to compute the covariance and inbreeding coefficients. Each computational rule is explained by an example referring to the fictitious population introduced in the section "Getting Started: INBREED Procedure" on page 5382.

Coancestry (or Kinship Coefficient)

To calculate the inbreeding coefficient and the covariance coefficients, use the degree of relationship by descent between the two parents, which is called *coancestry* or *kinship coefficient* (Falconer and Mackay 1996, p.85), or *coefficient of parentage* (Kempthorne 1957, p.73). Denote the coancestry between individuals X and Y by f_{XY}. For information on how to calculate the coancestries among a population, see the section "Calculation of Coancestry" on page 5392.

Covariance Coefficient (or Coefficient of Relationship)

The covariance coefficient between individuals X and Y is defined by

$$\mathrm{Cov}(\mathrm{X},\mathrm{Y}) = 2f_{\mathrm{XY}}$$

where f_{XY} is the coancestry between X and Y. The covariance coefficient is sometimes called the *coefficient of relationship* or the *theoretical correlation* (Falconer and Mackay (1996, p.153); Crow and Kimura (1970, p.134)). If a covariance coefficient cannot be calculated from the individuals in the population, it is assigned to an initial value. The initial value is set to 0 if the INIT= option is not specified or to *cov* if INIT=*cov*. Therefore, the corresponding initial coancestry is set to 0 if the INIT= option is not specified or to $\frac{1}{2}$*cov* if INIT=*cov*.

Inbreeding Coefficients

The inbreeding coefficient of an individual is the probability that the pair of alleles carried by the gametes that produced it are identical by descent (Falconer and Mackay (1996, Chapter 5), Kempthorne (1957, Chapter 5)). For individual X, denote its inbreeding coefficient by F_{X}. The inbreeding coefficient of an individual is equal to the coancestry between its parents. For example, if X has parents A and B, then the inbreeding coefficient of X is

$$F_{\mathrm{X}} = f_{\mathrm{AB}}$$

Calculation of Coancestry

Given individuals X and Y, assume that X has parents A and B and that Y has parents C and D. For nonoverlapping generations, the basic rule to calculate the coancestry between X and Y is given by the following formula (Falconer and Mackay 1996, p.86):

$$f_{\mathrm{XY}} = \frac{1}{4}\left(f_{\mathrm{AC}} + f_{\mathrm{AD}} + f_{\mathrm{BC}} + f_{\mathrm{BD}}\right)$$

And the inbreeding coefficient for an offspring of X and Y, called Z, is the coancestry between X and Y:

$$F_{\mathrm{Z}} = f_{\mathrm{XY}}$$

Figure 71.4 Inbreeding Relationship for Nonoverlapping Population

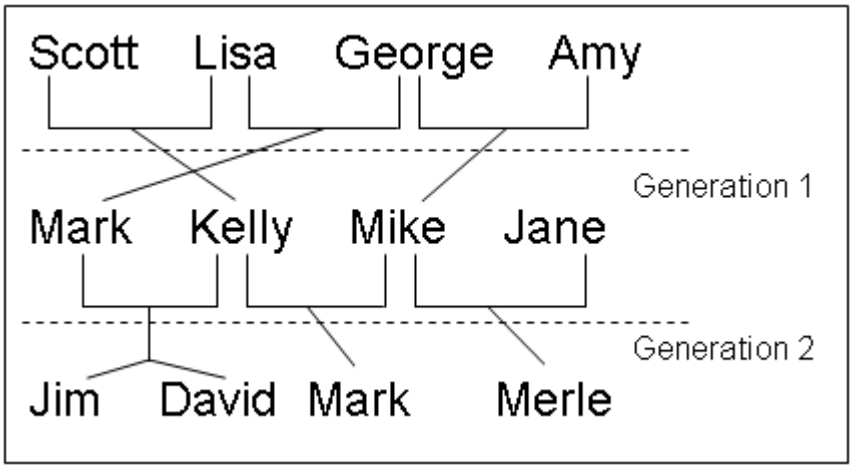

For example, in Figure 71.4, 'Jim' and 'Mark' from Generation 2 are progenies of 'Mark' and 'Kelly' and of 'Mike' and 'Kelly' from Generation 1, respectively. The coancestry between 'Jim' and 'Mark' is

$$f_{\mathrm{Jim,Mark}} = \frac{1}{\left(f_{\mathrm{Mark,Mike}} + f_{\mathrm{Mark,Kelly}} + f_{\mathrm{Kelly,Mike}} + f_{\mathrm{Kelly,Kelly}}\right)}$$

From the covariance matrix for Generation=1 in Figure 71.4 and the relationship that coancestry is half of the covariance coefficient,

$$f_{\mathrm{Jim,Mark}} = \frac{1}{4}\left(\frac{0.4688}{2} + \frac{0.5}{2} + \frac{0.25}{2} + \frac{1.125}{2}\right) = 0.29298$$

For overlapping generations, if X is older than Y, then the basic rule can be simplified to

$$F_{\mathrm{Z}} = f_{\mathrm{XY}} = \frac{1}{2}\left(f_{\mathrm{XC}} + f_{\mathrm{XD}}\right)$$

That is, the coancestry between X and Y is the average of coancestries between older X with younger Y's parents. For example, in Figure 71.5, the coancestry between 'Kelly' and 'David' is

$$f_{\mathrm{Kelly,David}} = \frac{1}{2}\left(f_{\mathrm{Kelly,Mark}} + f_{\mathrm{Kelly,Kelly}}\right)$$

Figure 71.5 Inbreeding Relationship for Overlapping Population

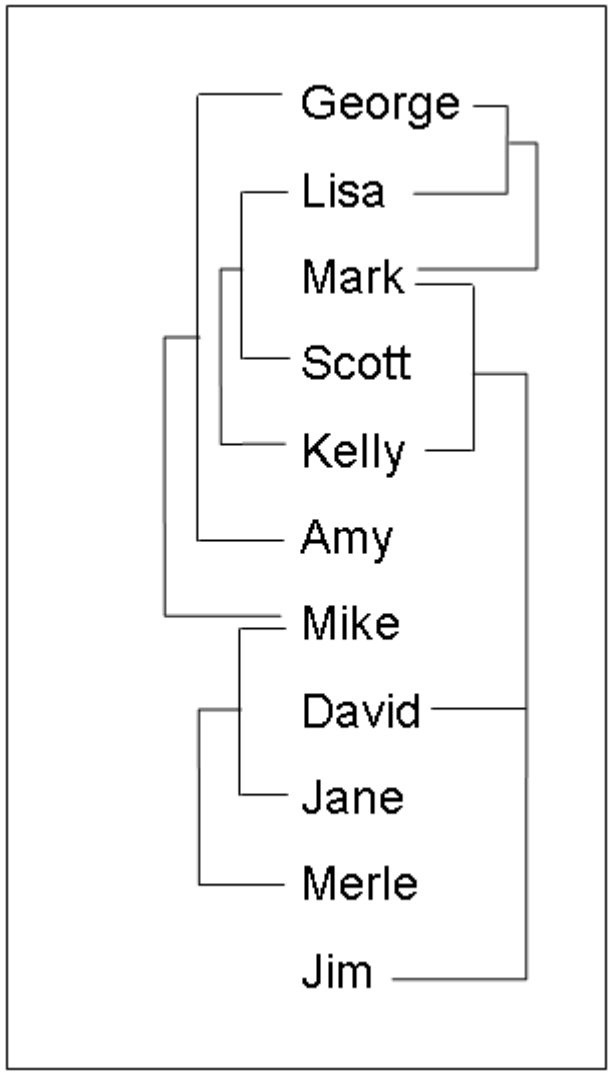

This is so because 'Kelly' is defined before 'David'; therefore, 'Kelly' is not younger than 'David', and the parents of 'David' are 'Mark' and 'Kelly'. The covariance coefficient values Cov(Kelly,Mark) and Cov(Kelly,Kelly) from the matrix in Figure 71.5 yield that the coancestry between 'Kelly' and 'David' is

$$f_{\text{Kelly,David}} = \frac{1}{2}\left(\frac{0.5}{2} + \frac{1.125}{2}\right) = 0.40625$$

The numerical values for some initial coancestries must be known in order to use these rule. Either the parents of the first generation have to be unrelated, with $f = 0$ if the INIT= option is not specified in the PROC INBREED statement, or their coancestries must have an initial value of $\frac{1}{2}cov$, where cov is set by the INIT= option. Then the subsequent coancestries among their progenies and the inbreeding coefficients of their progenies in the rest of the generations are calculated by using these initial values.

Special rules need to be considered in the calculations of coancestries for the following cases.

Self-Mating

The coancestry for an individual X with itself, f_{XX}, is the inbreeding coefficient of a progeny that is produced by self-mating. The relationship between the inbreeding coefficient and the coancestry for self-mating is

$$f_{\mathrm{XX}} = \frac{1}{2}\left(1 + F_{\mathrm{X}}\right)$$

The inbreeding coefficient F_{X} can be replaced by the coancestry between X's parents A and B, f_{AB}, if A and B are in the population:

$$f_{\mathrm{XX}} = \frac{1}{2}\left(1 + f_{\mathrm{AB}}\right)$$

If X's parents are not in the population, then F_{X} is replaced by the initial value $\frac{1}{2}cov$ if cov is set by the INIT= option, or F_{X} is replaced by 0 if the INIT= option is not specified. For example, the coancestry of 'Jim' with himself is

$$f_{\mathrm{Jim,Jim}} = \frac{1}{2}\left(1 + f_{\mathrm{Mark,Kelly}}\right)$$

where 'Mark' and 'Kelly' are the parents of 'Jim'. Since the covariance coefficient Cov(Mark,Kelly) is 0.5 in Figure 71.5 and also in the covariance matrix for GENDER=1 in Figure 71.4, the coancestry of 'Jim' with himself is

$$f_{\mathrm{Jim,Jim}} = \frac{1}{2}\left(1 + \frac{0.5}{2}\right) = 0.625$$

When INIT=0.25, then the coancestry of 'Jane' with herself is

$$f_{\mathrm{Jane,Jane}} = \frac{1}{2}\left(1 + \frac{0.25}{2}\right) = 0.5625$$

because 'Jane' is not an offspring in the population.

Offspring and Parent Mating

Assuming that X's parents are A and B, the coancestry between X and A is

$$f_{\mathrm{XA}} = \frac{1}{2}\left(f_{\mathrm{AB}} + f_{\mathrm{AA}}\right)$$

The inbreeding coefficient for an offspring of X and A, denoted by Z, is

$$F_{\mathrm{Z}} = f_{\mathrm{XA}} = \frac{1}{2}\left(f_{\mathrm{AB}} + f_{\mathrm{AA}}\right)$$

For example, 'Mark' is an offspring of 'George' and 'Lisa', so the coancestry between 'Mark' and 'Lisa' is

$$f_{\mathrm{Mark,Lisa}} = \frac{1}{2}\left(f_{\mathrm{Lisa,George}} + f_{\mathrm{Lisa,Lisa}}\right)$$

From the covariance coefficient matrix in Figure 71.5, $f_{\mathrm{Lisa,George}} = 0.25/2 = 0.125$, $f_{\mathrm{Lisa,Lisa}} = 1.125/2 = 0.5625$, so that

$$f_{\mathrm{Mark,Lisa}} = \frac{1}{2}\left(0.125 + 0.5625\right) = 0.34375$$

Thus, the inbreeding coefficient for an offspring of 'Mark' and 'Lisa' is 0.34375.

Full Sibs Mating

This is a special case for the basic rule given at the beginning of the section "Calculation of Coancestry" on page 5392. If X and Y are full sibs with same parents A and B, then the coancestry between X and Y is

$$f_{\mathrm{XY}} = \frac{1}{4}\left(2f_{\mathrm{AB}} + f_{\mathrm{AA}} + f_{\mathrm{BB}}\right)$$

and the inbreeding coefficient for an offspring of A and B, denoted by Z, is

$$F_{\mathrm{Z}} = f_{\mathrm{XY}} = \frac{1}{4}\left(2f_{\mathrm{AB}} + f_{\mathrm{AA}} + f_{\mathrm{BB}}\right)$$

For example, 'David' and 'Jim' are full sibs with parents 'Mark' and 'Kelly', so the coancestry between 'David' and 'Jim' is

$$f_{\mathrm{David,Jim}} = \frac{1}{4}\left(2f_{\mathrm{Mark,Kelly}} + f_{\mathrm{Mark,Mark}} + f_{\mathrm{Kelly,Kelly}}\right)$$

Since the coancestry is half of the covariance coefficient, from the covariance matrix in Figure 71.5,

$$f_{\mathrm{David,Jim}} = \frac{1}{4}\left(2 \times \frac{0.5}{2} + \frac{1.125}{2} + \frac{1.125}{2}\right) = 0.40625$$

Unknown or Missing Parents

When individuals or their parents are unknown in the population, their coancestries are assigned by the value $\frac{1}{2}cov$ if cov is set by the INIT= option or by the value 0 if the INIT= option is not specified. That is, if either A or B is unknown, then

$$f_{\mathrm{AB}} = \frac{1}{2}cov$$

For example, 'Jane' is not in the population, and since 'Jane' is assumed to be defined just before the observation at which 'Jane' appears as a parent (that is, between observations 4 and 5), then 'Jane' is not older than 'Scott'. The coancestry between 'Jane' and 'Scott' is then obtained by using the simplified basic rule (see the section "Calculation of Coancestry" on page 5392):

$$f_{\mathrm{Scott,Jane}} = \frac{1}{2}\left(f_{\mathrm{Scott},\cdot} + f_{\mathrm{Scott},\cdot}\right)$$

Here, dots $(\cdot)$ indicate Jane's unknown parents. Therefore, $f_{\mathrm{Scott},\cdot}$ is replaced by $\frac{1}{2}cov$, where cov is set by the INIT= option. If INIT=0.25, then

$$f_{\mathrm{Scott,Jane}} = \frac{1}{2}\left(\frac{0.25}{2} + \frac{0.25}{2}\right) = 0.125$$

For a more detailed discussion on the calculation of coancestries, inbreeding coefficients, and covariance coefficients, see Falconer and Mackay (1996); Kempthorne (1957); Crow and Kimura (1970).

OUTCOV= Data Set

The OUTCOV= data set has the following variables:

- a list of BY variables, if there is a BY statement
- the generation variable, if there is a CLASS statement
- the gender variable, if there is a GENDER statement
- _Type_, a variable indicating the type of observation. The valid values of the _Type_ variable are 'COV' for covariance estimates and 'INBREED' for inbreeding coefficients.
- _Panel_, a variable indicating the panel number used when populations delimited by BY groups contain different numbers of individuals. If there are n individuals in the first BY group and if any subsequent BY group contains a larger population, then its covariance/inbreeding matrix is divided into panels, with each panel containing n columns of data. If you put these panels side by side in increasing _Panel_ number order, then you can reconstruct the covariance or inbreeding matrix.

- _Col_, a variable used to name columns of the inbreeding or covariance matrix. The values of this variable start with 'COL', followed by a number indicating the column number. The names of the individuals corresponding to any given column *i* can be found by reading the individual's name across the row that has a _Col_ value of 'COL*i*'. When the inbreeding or covariance matrix is divided into panels, all the rows repeat for the first *n* columns, all the rows repeat for the next *n* columns, and so on.
- the variable containing the names of the individuals, that is, the first variable listed in the VAR statement
- the variable containing the names of the first parents, that is, the second variable listed in the VAR statement
- the variable containing the names of the second parents, that is, the third variable listed in the VAR statement
- a list of covariance variables Col1–Col*n*, where *n* is the maximum number of individuals in the first population

The functions of the variables _Panel_ and _Col_ can best be demonstrated by an example. Assume that there are three individuals in the first BY group and that, in the current BY group (Byvar=2), there are five individuals with the following covariance matrix.

COV	1	2	3	4	5
1	Cov(1,1)	Cov(1,2)	Cov(1,3)	Cov(1,4)	Cov(1,5)
2	Cov(2,1)	Cov(2,2)	Cov(2,3)	Cov(2,4)	Cov(2,5)
3	Cov(3,1)	Cov(3,2)	Cov(3,3)	Cov(3,4)	Cov(3,5)
4	Cov(4,1)	Cov(4,2)	Cov(4,3)	Cov(4,4)	Cov(4,5)
5	Cov(5,1)	Cov(5,2)	Cov(5,3)	Cov(5,4)	Cov(5,5)
	Panel 1			Panel 2	

Then the OUTCOV= data set appears as follows.

Byvar	_Panel_	_Col_	Individual	Parent	Parent2	Col1	Col2	Col3
2	1	COL1	1			Cov(1,1)	Cov(1,2)	Cov(1,3)
2	1	COL2	2			Cov(2,1)	Cov(2,2)	Cov(2,3)
2	1	COL3	3			Cov(3,1)	Cov(3,2)	Cov(3,3)
2	1		4			Cov(4,1)	Cov(4,2)	Cov(4,3)
2	1		5			Cov(5,1)	Cov(5,2)	Cov(5,3)
2	2		1			Cov(1,4)	Cov(1,5)	.
2	2		2			Cov(2,4)	Cov(2,5)	.
2	2		3			Cov(3,4)	Cov(3,5)	.
2	2	COL1	4			Cov(4,4)	Cov(4,5)	.
2	2	COL2	5			Cov(5,4)	Cov(5,5)	.

Notice that the first three columns go to the first panel (_Panel_=1), and the remaining two go to the second panel (_Panel_=2). Therefore, in the first panel, 'COL1', 'COL2', and 'COL3' correspond to individuals 1, 2, and 3, respectively, while in the second panel, 'COL1' and 'COL2' correspond to individuals 4 and 5, respectively.

Displayed Output

The INBREED procedure can output either covariance coefficients or inbreeding coefficients. Note that the following items can be produced for each generation if generations do not overlap.

The output produced by PROC INBREED can be any or all of the following items:

- a matrix of coefficients
- coefficients of the individuals
- coefficients for selected matings

ODS Table Names

PROC INBREED assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 71.2. For more information on ODS, see Chapter 23, "Using the Output Delivery System."

Table 71.2 ODS Tables Produced by PROC INBREED

ODS Table Name	Description	Statement	Option
AvgCovCoef	Averages of covariance coefficient matrix	GENDER	COVAR and AVERAGE
AvgInbreedingCoef	Averages of inbreeding coefficient matrix	GENDER	AVERAGE
CovarianceCoefficient	Covariance coefficient table	PROC INBREED	COVAR and MATRIX
InbreedingCoefficient	Inbreeding coefficient table	PROC INBREED	MATRIX
IndividualCovCoef	Covariance coefficients of individuals	PROC INBREED	IND and COVAR
IndividualInbreedingCoef	Inbreeding coefficients of individuals	PROC INBREED	IND
MatingCovCoef	Covariance coefficients of matings	MATINGS	COVAR
MatingInbreedingCoef	Inbreeding coefficients of matings	MATINGS	
NumberOfObservations	Number of observations	PROC INBREED	

Examples: INBREED Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 71.1: Monoecious Population Analysis

(View the complete code for this example (inbrex1.sas) in the example repository.)

The following example shows a covariance analysis within nonoverlapping generations for a monoecious population. Parents of generation 1 are unknown and therefore assumed to be unrelated. The following statements produce Output 71.1.1 through Output 71.1.3:

```
data Monoecious;
   input Generation Individual Parent1 Parent2 Covariance @@;
   datalines;
1 1 . .  .     1 2 . .  .      1 3 . .  .
2 1 1 1  .     2 2 1 2  .      2 3 2 3  .
3 1 1 2  .     3 2 1 3  .      3 3 2 1  .
3 4 1 3  .     3 . 2 3 0.50    3 . 4 3 1.135
;

title 'Inbreeding within Nonoverlapping Generations';
proc inbreed ind covar matrix data=Monoecious;
   class Generation;
run;
```

Output 71.1.1 Monoecious Population Analysis, Generation 1

Inbreeding within Nonoverlapping Generations

The INBREED Procedure

Generation = 1

Covariance Coefficients					
Individual	**Parent1**	**Parent2**	**1**	**2**	**3**
1			1.0000	.	.
2			.	1.0000	.
3			.	.	1.0000

Covariance Coefficients of Individuals			
Individual	**Parent1**	**Parent2**	**Coefficient**
1			1.0000
2			1.0000
3			1.0000

Number of Individuals	3

Output 71.1.2 Monoecious Population Analysis, Generation 2

Inbreeding within Nonoverlapping Generations

The INBREED Procedure

Generation = 2

Covariance Coefficients					
Individual	**Parent1**	**Parent2**	**1**	**2**	**3**
1	**1**	**1**	1.5000	0.5000	.
2	**1**	**2**	0.5000	1.0000	0.2500
3	**2**	**3**	.	0.2500	1.0000

Covariance Coefficients of Individuals			
Individual	**Parent1**	**Parent2**	**Coefficient**
1	**1**	**1**	1.5000
2	**1**	**2**	1.0000
3	**2**	**3**	1.0000

Number of Individuals	3

Output 71.1.3 Monoecious Population Analysis, Generation 3

Inbreeding within Nonoverlapping Generations

The INBREED Procedure

Generation = 3

Covariance Coefficients						
Individual	**Parent1**	**Parent2**	**1**	**2**	**3**	**4**
1	**1**	**2**	1.2500	0.5625	0.8750	0.5625
2	**1**	**3**	0.5625	1.0000	1.1349	0.6250
3	**2**	**1**	0.8750	1.1349	1.2500	1.1349
4	**1**	**3**	0.5625	0.6250	1.1349	1.0000

Covariance Coefficients of Individuals			
Individual	**Parent1**	**Parent2**	**Coefficient**
1	**1**	**2**	1.2500
2	**1**	**3**	1.0000
3	**2**	**1**	1.2500
4	**1**	**3**	1.0000

Number of Individuals	4

Note that, since the parents of the first generation are unknown, off-diagonal elements of the covariance matrix are all 0s and on-diagonal elements are all 1s. If there is an INIT=*cov* value, then the off-diagonal elements would be equal to *cov*, while on-diagonal elements would be equal to $1 + cov/2$.

In the third generation, individuals 2 and 4 are full siblings, so they belong to the same family. Since PROC

INBREED computes covariance coefficients between families, the second and fourth columns of inbreeding coefficients are the same, except that their intersections with the second and fourth rows are reordered. Notice that, even though there is an observation to assign a covariance of 0.50 between individuals 2 and 3 in the third generation, the covariance between 2 and 3 is set to 1.135, the same value assigned between 4 and 3. This is because families get the same covariances, and later specifications override previous ones.

Example 71.2: Pedigree Analysis

(View the complete code for this example (inbrex2.sas) in the example repository.)

In the following example, an inbreeding analysis is performed for a complicated pedigree. This analysis includes computing selective matings of some individuals and inbreeding coefficients of all individuals. Also, inbreeding coefficients are averaged within sex categories. The following statements produce Output 71.2.1:

```
data Swine;
   input Swine_Number $ Sire $ Dam $ Sex $;
   datalines;
3504 2200 2501  M
3514 2521 3112  F
3519 2521 2501  F
2501 2200 3112  M
2789 3504 3514  F
3501 2521 3514  M
3712 3504 3514  F
3121 2200 3501  F
;

title 'Least Related Matings';
proc inbreed data=Swine ind average;
   var Swine_Number Sire Dam;
   matings 2501 / 3501 3504 ,
           3712 / 3121;
   gender Sex;
run;
```

Note the following from Output 71.2.1:

- Observation 4, which defines Swine_Number=2501, should precede the first and third observations where the progeny for 2501 are given. PROC INBREED ignores observation 4 since it is given out of order. As a result, the parents of 2501 are missing or unknown.
- The first column in the “Inbreeding Averages” table corresponds to the averages taken over the on-diagonal elements of the inbreeding coefficients matrix, and the second column gives averages over the off-diagonal elements.

Output 71.2.1 Pedigree Analysis

Least Related Matings

The INBREED Procedure

Inbreeding Coefficients of Individuals			
Swine_Number	Sire	Dam	Coefficient
2200			.
2501			.
3504	2200	2501	.
2521			.
3112			.
3514	2521	3112	.
3519	2521	2501	.
2789	3504	3514	.
3501	2521	3514	0.2500
3712	3504	3514	.
3121	2200	3501	.

Inbreeding Coefficients of Matings		
Sire	Dam	Coefficient
2501	3501	.
2501	3504	0.2500
3712	3121	0.1563

Averages of Inbreeding Coefficient Matrix		
	Inbreeding	Coancestry
Male X Male	0.0625	0.1042
Male X Female	.	0.1362
Female X Female	0.0000	0.1324
Over Sex	0.0227	0.1313

Number of Males	4
Number of Females	7
Number of Individuals	11

Example 71.3: Pedigree Analysis with BY Groups

(View the complete code for this example (inbrex3.sas) in the example repository.)

This example demonstrates the structure of the OUTCOV= data set created by PROC INBREED. Note that the first BY group has three individuals, while the second has five. Therefore, the covariance matrix for the second BY group is broken up into two panels. The following statements produce Output 71.3.1.

```
data Swine;
   input Group Swine_Number $ Sire $ Dam $ Sex $;
   datalines;
1  2789 3504 3514  F
2  2501 2200 3112  .
2  3504 2501 3782  M
;

proc inbreed data=Swine covar noprint outcov=Covariance
             init=0.4;
   var Swine_Number Sire Dam;
   gender Sex;
   by Group;
run;

title 'Printout of OUTCOV= data set';
proc print data=Covariance;
   format Col1-Col3 4.2;
run;
```

Output 71.3.1 Pedigree Analysis with BY Groups

Printout of OUTCOV= data set

Obs	Group	Sex	_TYPE_	_PANEL_	_COL_	Swine_Number	Sire	Dam	COL1	COL2	COL3
1	1	M	COV	1	COL1	3504			1.20	0.40	0.80
2	1	F	COV	1	COL2	3514			0.40	1.20	0.80
3	1	F	COV	1	COL3	2789	3504	3514	0.80	0.80	1.20
4	2	M	COV	1	COL1	2200			1.20	0.40	0.80
5	2	F	COV	1	COL2	3112			0.40	1.20	0.80
6	2	M	COV	1	COL3	2501	2200	3112	0.80	0.80	1.20
7	2	F	COV	1		3782			0.40	0.40	0.40
8	2	M	COV	1		3504	2501	3782	0.60	0.60	0.80
9	2	M	COV	2		2200			0.40	0.60	.
10	2	F	COV	2		3112			0.40	0.60	.
11	2	M	COV	2		2501	2200	3112	0.40	0.80	.
12	2	F	COV	2	COL1	3782			1.20	0.80	.
13	2	M	COV	2	COL2	3504	2501	3782	0.80	1.20	.

References

Crow, J. F., and Kimura, M. (1970). *An Introduction to Population Genetics Theory*. New York: Harper & Row.

Falconer, D. S., and Mackay, T. F. C. (1996). *Introduction to Quantitative Genetics*. 4th ed. Harlow, UK: Longman.

Kempthorne, O. (1957). *An Introduction to Genetic Statistics*. New York: John Wiley & Sons.

Chapter 72
The IRT Procedure

Contents

Overview: IRT Procedure **5408**
- Basic Features 5408

Getting Started: IRT Procedure **5409**

Syntax: IRT Procedure **5413**
- PROC IRT Statement 5414
- BY Statement 5425
- COV Statement 5426
- EQUALITY Statement 5430
- FACTOR Statement 5436
- FIXVALUE Statement 5439
- FREQ Statement 5444
- GROUP Statement 5444
- MEAN Statement 5445
- MODEL Statement 5446
- VAR Statement 5448
- VARIANCE Statement 5449
- WEIGHT Statement 5451

Details: IRT Procedure **5452**
- Notation for the Item Response Theory Model 5452
- Assumptions 5454
- PROC IRT Contrasted with Other SAS Procedures 5454
- Response Models 5455
- Marginal Likelihood 5456
- Approximating the Marginal Likelihood 5456
- Maximizing the Marginal Likelihood 5457
- Prior Distributions for Parameters 5460
- Factor Score Estimation 5461
- Model and Item Fit 5462
- Item and Test information 5463
- Missing Values 5463
- Output Data Sets 5463
- ODS Table Names 5465
- ODS Graphics 5466

Examples: IRT Procedure **5467**
- Example 72.1: Unidimensional IRT Models 5467
- Example 72.2: Multidimensional Exploratory and Confirmatory IRT Models 5484

Example 72.3: Multiple-Group Analysis 5488
Example 72.4: Item Selection Using Item and Test Information 5492
Example 72.5: Subject Scoring 5499
References **5500**

Overview: IRT Procedure

The item response theory (IRT) model was first proposed in the field of psychometrics for the purpose of ability assessment. It is most widely used in education to calibrate and evaluate items in tests, questionnaires, and other instruments and to score subjects on their abilities, attitudes, or other latent traits. Today, all major psychological and educational tests are built using IRT, because the methodology can significantly improve measurement accuracy and reliability while providing potential significant reductions in assessment time and effort, especially via computerized adaptive testing. In a computerized adaptive test, items are optimally selected for each subject. Different subjects might receive entirely different items during the test. IRT plays an essential role in selecting the most appropriate items for each subject and equating scores for subjects who receive different subsets of items. Notable examples of these tests include the Scholastic Aptitude Test (SAT), Graduate Record Examination (GRE), and Graduate Management Admission Test (GMAT). In recent years, IRT models have also become increasingly popular in health behavior, quality of life, and clinical research. The Patient Reported Outcomes Measurement Information System (PROMIS) project, funded by the US National Institutes of Health, is an excellent example. By using IRT, it aims to develop item banks that clinicians and researchers can use to collect important information about therapeutic effects that is not available from traditional clinical measures.

Early IRT models (such as the Rasch model and two-parameter model) concentrate mainly on dichotomous responses. These models were later extended to incorporate other formats, such as ordinal responses, rating scales, partial credit scoring, and multiple category scoring. Early applications of IRT focused primarily on the unidimensional model, which assumes that subject responses are affected only by a single latent trait. Multidimensional IRT models have been developed, but because of their greater complexity, the majority of IRT applications still rely on unidimensional models.

For an introduction to IRT models, see De Ayala (2009) and Embretson and Reise (2000).

Basic Features

The IRT procedure enables you to estimate various item response theory models. The following list summarizes some of the basic features of the IRT procedure:

- uses the Rasch model; one-, two-, three-, and four-parameter models; graded response model with logistic or probit link; and generalized partial credit model
- enables different items to have different response models
- performs multidimensional exploratory and confirmatory analysis
- performs multiple-group analysis, with fixed values and equality constraints within and between groups

- estimates factor scores by using maximum likelihood (ML), maximum a posteriori (MAP), and expected a posteriori (EAP) methods

Getting Started: IRT Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/irtgs1.sas.)

This example shows how you can use all default settings in PROC IRT to fit an item response model. In this example, there are 50 subjects and each subject responds to 10 items. These 10 items have binary responses: 1 indicates correct and 0 indicates incorrect.

The following DATA step creates the SAS data set IrtBinary:

```
data IrtBinary;
   input item1-item10 @@;
   datalines;
1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 0 1 0 0 1 1 1 1 1 1
0 0 0 0 0 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 0 0 1 0 1 1 1 1 1 1 1 1 1 1 1

   ... more lines ...

1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1
;
```

The following statements fit an IRT model:

```
proc irt data=IrtBinary;
   var item1-item10;
run;
```

The PROC IRT statement invokes the procedure, and the DATA= option specifies the input data set IrtBinary. The VAR statement names the variables to be used in the model. As you can see from the syntax in this example, fitting a IRT model can be very simple when you use the default settings. These default settings are chosen to reflect setups that are common in practice. Some of the important default settings follow:

- The number of factors is 1.
- The two-parameter logistic model is assumed for binary variables, and the graded response model is assumed for ordinal variables.
- The link function is logistic link.
- The estimation method is based on marginal likelihood.
- The optimization method is the quasi-Newton algorithm.
- The quadrature method is adaptive Gauss-Hermite quadrature, in which the number of quadrature points per dimension is determined adaptively.

As a result, the preceding statements fit two-parameter logistic (2PL) models for all the variables that are listed in the VAR statement.

The first table that PROC IRT produces is the "Modeling Information" table, as shown in Figure 72.1. This table displays basic information about the analysis, such as the name of the input data set, the link function, the number of items and factors, the number of observations, and the estimation method. You can change the link function by using the LINK= option in the PROC IRT statement. You can change the response model for all the items by using the RESFUNC= option in the PROC IRT statement. You can specify different response functions or models for different set of variables by including a MODEL statement. If you want to do multidimensional exploratory analysis, you can simply change the number of factors by using the NFACTOR= option in the PROC IRT statement. For confirmatory analysis, you can use the FACTOR statement to specify the confirmatory factor pattern; the number of factors is implicitly defined by the number of distinctive factor names that you specify in the FACTOR statement.

Figure 72.1 Model Information

The IRT Procedure

Modeling Information	
Data Set	WORK.IRTBINARY
Link Function	Logit
Response Model	Two Parameter Model
Number of Items	10
Number of Factors	1
Number of Observations Read	100
Number of Observations Used	100
Estimation Method	Marginal Maximum Likelihood

The "Item Information" table, shown in Figure 72.2, is displayed by default and can be used to check the item-level information. In this case, each of the 10 variables has two levels, and the raw values for these two levels are 0 and 1, respectively.

Figure 72.2 Item Information

Item Information		
Item	**Levels**	**Values**
item1	2	0 1
item2	2	0 1
item3	2	0 1
item4	2	0 1
item5	2	0 1
item6	2	0 1
item7	2	0 1
item8	2	0 1
item9	2	0 1
item10	2	0 1

The eigenvalues of polychoric correlations are also computed by default and are shown in Figure 72.3. You can use the information from these eigenvalues to assess a reasonable range for the number of factors. For this example, you can observe that the first eigenvalue accounts for almost 50% of the variance, which suggests

that there is only one dominant eigenvalue and that a unidimensional model is reasonable for this example. To produce the polychoric correlation table, you specify the POLYCHORIC option in the PROC IRT statement.

Figure 72.3 Eigenvalues of Polychoric Correlation

Eigenvalues of the Polychoric Correlation Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	4.71105177	3.51149453	0.4711	0.4711
2	1.19955723	0.14183502	0.1200	0.5911
3	1.05772221	0.26577735	0.1058	0.6968
4	0.79194486	0.07204549	0.0792	0.7760
5	0.71989938	0.17782491	0.0720	0.8480
6	0.54207446	0.12713664	0.0542	0.9022
7	0.41493782	0.10631770	0.0415	0.9437
8	0.30862012	0.12256183	0.0309	0.9746
9	0.18605829	0.11792444	0.0186	0.9932
10	0.06813385		0.0068	1.0000

Next, the "Optimization Information" table, shown in Figure 72.4, lists the optimization technique, the numeric quadrature method, and the number of quadrature points per dimension. If you want to use the expectation-maximization (EM) technique, specify TECHNIQUE=EM in the PROC IRT statement. If you specify the NOAD option in the PROC IRT statement, PROC IRT uses the nonadaptive Gauss-Hermite quadrature to approximate the likelihood. You can change the number of quadrature points by specifying the QPOINTS= option in the PROC IRT statement.

Figure 72.4 Optimization Information

Optimization Information	
Optimization Technique	Quasi-Newton
Likelihood Approximation	Adaptive Gauss-Hermite Quadrature
Number of Quadrature Points	21
Number of Free Parameters	20

Figure 72.5 shows the "Iteration History" table. For each iteration, the table displays the current iteration number, number of function evaluations, objective function value, change of object function value, and maximum value of gradients. You can use this information to monitor the estimation status of the model. You can turn off the display of the "Iteration History" table by specifying the NOITPRINT option in the PROC IRT statement.

Following the "Iteration History" table is the convergence status table, shown in Figure 72.6. It shows whether the optimization algorithm converges successfully or not. You should make sure that the optimization converges successfully before you try to interpret the estimation results.

Figure 72.5 Iteration History

Iteration History					
Cycles	Iteration	Evaluations	Objective Function	Function Change	Max Abs Gradient
0	0	2	5.53444506		0.045818
0	1	4	5.43305912	-0.10138594	0.017751
0	2	6	5.41425139	-0.01880773	0.016637
0	3	8	5.40300779	-0.01124360	0.008948
0	4	10	5.40151894	-0.00148885	0.006556
0	5	12	5.40100096	-0.00051798	0.002865
0	6	15	5.40087882	-0.00012214	0.001574
0	7	18	5.40082176	-0.00005707	0.000885
0	8	21	5.40079448	-0.00002728	0.000486
0	9	24	5.40078766	-0.00000682	0.000337
0	10	27	5.40078296	-0.00000470	0.000185
0	11	30	5.40078215	-0.00000081	0.000103
0	12	32	5.40078180	-0.00000034	0.000157
0	13	34	5.40078145	-0.00000036	0.000037
0	14	37	5.40078142	-0.00000003	0.000038
0	15	40	5.40078141	-0.00000001	0.000012
1	0	2	5.40078179		0.000012
1	1	4	5.40078179	-0.00000000	8.953E-6

Figure 72.6 Convergence Status

Convergence criterion (GCONV=.000000010) satisfied.

Next is the "Model Fit Statistics" table, shown in Figure 72.7, which includes the log likelihood, Akaike's information criterion (AIC), and the Bayesian information criterion (BIC). If all the response patterns are observed, Pearson's chi-square and likelihood ratio chi-square statistics are also included in this table. Because some of the response patterns in this example are not observed, the Pearson's chi-square statistic is not included in the table.

Figure 72.7 Fit Statistics

Model Fit Statistics	
Log Likelihood	-540.0781788
AIC (Smaller is Better)	1120.1563577
BIC (Smaller is Better)	1172.2597614
LR Chi-Square	300.35009224
LR Chi-Square DF	1003

Finally, the "Item Parameter Estimates" table, shown in Figure 72.8, includes parameter estimates, standard errors, and p-values. Parameters are organized and displayed within each item. The items are listed in the order of their appearance in the modeling statements. For each item, there are two parameters: difficulty and slope. Difficulty parameters measure the difficulties of the items. As the value of the difficulty parameter increases, the item becomes more difficult. In Figure 72.8, you can observe that all the difficulty parameters are less than 0, which suggests that all the items in this example are relatively easy. The slope parameter

values for this example range from 0.94 to 2.33, suggesting that all the items are adequate measures of the latent trait.

Figure 72.8 Parameter Estimates

Item Parameter Estimates				
Item	Parameter	Estimate	Standard Error	Pr > |t|
item1	Difficulty	-0.86575	0.20048	<.0001
	Slope	2.22000	0.68402	0.0006
item2	Difficulty	-1.01624	0.21352	<.0001
	Slope	2.33903	0.76112	0.0011
item3	Difficulty	-0.91123	0.20824	<.0001
	Slope	2.18657	0.68338	0.0007
item4	Difficulty	-0.92388	0.22607	<.0001
	Slope	1.87302	0.57533	0.0006
item5	Difficulty	-1.09219	0.30470	0.0002
	Slope	1.33931	0.42768	0.0009
item6	Difficulty	-0.48807	0.24190	0.0218
	Slope	1.17677	0.37111	0.0008
item7	Difficulty	-0.61762	0.29988	0.0197
	Slope	0.94553	0.32652	0.0019
item8	Difficulty	-0.50810	0.28295	0.0363
	Slope	0.95660	0.32983	0.0019
item9	Difficulty	-0.41112	0.24309	0.0454
	Slope	1.11733	0.35666	0.0009
item10	Difficulty	-0.62316	0.27829	0.0126
	Slope	1.05215	0.34965	0.0013

Syntax: IRT Procedure

The following statements are available in the IRT procedure:

PROC IRT < *options* > ;
- **BY** *variables* ;
- **COV** *covariance parameters* ;
- **EQUALITY** *equality-constraints* ;
- **FACTOR** *factor-variables-relations* ;
- **FIXVALUE** *fixed-value-constraints* ;
- **FREQ** *variable* ;
- **GROUP** *variable* ;
- **MEAN** *mean parameters* ;
- **MODEL** *model-specification* ;
- **VAR** *variables* ;
- **VARIANCE** *variance parameters* ;
- **WEIGHT** *variable* ;

PROC IRT Statement

PROC IRT < *options* > **;**

The PROC IRT statement invokes the IRT procedure. Table 72.1 summarizes the *options* available in the PROC IRT statement. The sections that follow the table describe the PROC IRT statement *options* and then describe the other statements in alphabetical order.

Table 72.1 PROC IRT Statement Options

Option	Description
Data Set Options	
DATA=	Specifies the input data set
INMODEL=	Inputs the model specifications
OUT=	Specifies the output data set for factor scores
OUTMODEL=	Outputs the model specifications
Analysis Options	
ITEMFIT	Computes the item fit statistics and displays them in a table
ITEMSTAT	Computes the classical item statistics and displays them in a table
SCOREMETHOD=	Specifies the factor score estimation method
Model Options	
CEILPRIOR=	Specifies prior information for ceiling parameters
DESCENDING	Reverses the sort order of the levels of the response variable
GUESSPRIOR=	Specifies prior information for guessing parameters
LINK=	Specifies the link function
NFACTOR=	Specifies the number of factors
RESFUNC=	Specifies the response function
RORDER=	Specifies the sort order of the response variables
SLOPEPRIOR=	Specifies prior information for slope parameters
Technical Options	
ABSFCONV=	Specifies an absolute function difference convergence criterion
ABSGCONV=	Specifies an absolute gradient convergence criterion
ABSPCONV=	Specifies a maximum absolute parameter difference convergence criterion
FCONV=	Specifies a relative function convergence criterion
GCONV=	Specifies a relative gradient convergence criterion
MAXFUNC=	Specifies the maximum number of function calls in the optimization process
MAXITER=	Specifies the maximum number of iterations in the optimization process
MAXMITER=	Specifies the maximum number of iterations in the maximization step of the EM algorithm
NOAD	Specifies nonadaptive quadrature
QPOINTS=	Specifies the number of quadrature points per dimension

Table 72.1 *continued*

Option	Description
TECHNIQUE=	Specifies the optimization technique to obtain maximum likelihood estimates
Display Options	
NOITPRINT	Suppresses the display of the "Iteration History" table
NOPRINT	Suppresses all ODS output
PINITIAL	Displays initial parameter estimates
POLYCHORIC	Displays the polychoric correlation matrix
PLOTS=	Controls plots that are produced through ODS Graphics
Rotation Method and Properties	
RCONVERGE=	Specifies the convergence criterion for rotation cycles
RITER=	Specifies the maximum number of rotation cycles
ROTATE=	Specifies the rotation method

PROC IRT Statement Options

ABSFCONV=*r*

ABSFTOL=*r*

specifies an absolute function difference convergence criterion. Termination requires a small change of the function value in successive iterations,

$$|f(\boldsymbol{\psi}^{(k-1)}) - f(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. This criterion is not used by the expectation-maximization (EM) algorithm. By default, $r = 0$.

ABSGCONV=*r*

ABSGTOL=*r*

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small,

$$\max_j |g_j(\boldsymbol{\psi}^{(k)})| \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the jth parameter. This criterion is not used by the EM algorithm. By default, r = 1E–5.

ABSPCONV=*r*

ABSPTOL=*r*

specifies a maximum absolute parameter difference convergence criterion. This criterion is used only by the EM algorithm. Termination requires the maximum absolute parameter change in successive iterations to be small,

$$\max_j |\boldsymbol{\psi}_j^{(k-1)} - \boldsymbol{\psi}_j^{(k)}| \leq r$$

where $\boldsymbol{\psi}_j$ denotes the jth parameter that participates in the optimization. By default, r = 1E–4.

CEILPRIOR=(< *r1* >, < *r2* >)

specifies the mean (*r1*) and weight (*r2*) of the beta prior distribution for the ceiling parameters. The *r1* value corresponds to the mean of the beta distribution, and it reflects your prior estimation of the ceiling parameter. The weight, *r2*, represents your confidence about this prior estimation. PROC IRT uses these values to define the beta prior distribution for all ceiling parameters of the items that are fit by the four-parameter model (that is, when RESFUNC=FOURP). If you want to specify different prior information for different items, use the CEILPRIOR= option in the MODEL statement. For more information, see the section "Prior Distributions for Parameters" on page 5460.

The following example specifies the mean and weight to be 0.8 and 10, respectively, for the prior distribution of the ceiling parameters:

```
proc irt ceilprior=(0.8,10);
run;
```

By default, *r1* = 0.9 and *r2* = 20.

DATA=*SAS-data-set*

specifies the *SAS-data-set* to be read by PROC IRT. The default value is the most recently created data set.

DESCENDING

DESC

reverses the sorting order for the levels of the response variables. If you specify both the DESCENDING and RORDER= options, PROC IRT orders the levels according to the RORDER= option and then reverses that order.

FCONV=*r*

FTOL=*r*

specifies a relative function convergence criterion. Termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\psi}^{(k)}) - f(\boldsymbol{\psi}^{(k-1)})|}{|f(\boldsymbol{\psi}^{(k-1)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization and $f(\cdot)$ is the objective function. This criterion is not used by the EM algorithm. By default, $r = 10^{-\mathrm{FDIGITS}}$, where FDIGITS is, by default, $-\log_{10}\{\epsilon\}$ and ϵ is the machine precision.

GCONV=*r*

GTOL=*r*

specifies a relative gradient convergence criterion. For all techniques except CONGRA, termination requires the normalized predicted function reduction to be small,

$$\frac{\mathbf{g}(\boldsymbol{\psi}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\psi}^{(k)})}{|f(\boldsymbol{\psi}^{(k)})|} \leq r$$

where $\boldsymbol{\psi}$ denotes the vector of parameters that participate in the optimization, $f(\cdot)$ is the objective function, and $\mathbf{g}(\cdot)$ is the gradient. For the CONGRA technique (for which a reliable Hessian estimate $\mathbf{H}$ is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\psi}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\psi}^{(k)}) - \mathbf{g}(\boldsymbol{\psi}^{(k-1)}) \|_2 \, |f(\boldsymbol{\psi}^{(k)})|} \leq r$$

This criterion is not used by the EM algorithm. By default, r = 1E–8.

GUESSPRIOR=(< *r1* >, < *r2* >)

specifies the mean (*r1*) and weight (*r2*) of the beta prior distribution for the guessing parameters. The *r1* value corresponds to the mean of the beta distribution, and it reflects your prior estimation of the guessing parameter. The weight, *r2*, represents your confidence about this prior estimation. PROC IRT uses these values to define the beta prior distribution for all the guessing parameters of the items that are fit by a three- or four-parameter model (that is, when RESFUNC=THREEP or FOURP). If you want to specify different prior information for different items, use the GUESSPRIOR= option in the MODEL statement. For more information, see the section "Prior Distributions for Parameters" on page 5460.

The following example specifies the mean and weight to be 0.1 and 10, respectively, for the prior distribution of the guessing parameters:

```
proc irt guessprior=(0.1,10);
run;
```

By default, *r1* = 0.2 and *r2* = 20.

INMODEL< (SCORE) >=*SAS-data-set*

specifies an input data set that contains information about the analysis model. Instead of specifying and running the model in a new run, you can use the INMODEL= option to input the model specification saved as an OUTMODEL= data set in a previous PROC IRT run.

Sometimes, you might want to create an INMODEL= data set by modifying an existing OUTMODEL= data set. However, editing and modifying OUTMODEL= data sets requires a good understanding of the formats and contents of the OUTMODEL= data sets. This process could be difficult for novice users. For more information about the format of INMODEL= and OUTMODEL= data sets, see the section "Output Data Sets" on page 5463.

When you specify the INMODEL= option, the VAR, MODEL, GROUP, FACTOR, VARIANCE, COV, and EQUALITY statements are ignored. The DESCENDING, LINK, NFACTOR, RESFUNC, and RORDER options in the PROC IRT statement are also ignored. When there are duplicated specifications, the first specification is used.

Specify the SCORE suboption if you want to use the model specifications and parameter estimates from the INMODEL= data set to score a new subject without refitting the model.

You can use the INMODEL= option along with the SCORE suboption for many different purposes, including the following:

- If you specify the INMODEL= option, PROC IRT fits an IRT model to the DATA= data set based on the model specifications in the INMODEL= data set and uses the parameter estimates in the INMODEL= data set as initial values.

- If you specify the INMODEL= option and the OUT= option, PROC IRT fits an IRT model to the DATA= data set based on the model specifications in the INMODEL= data set and uses the parameter estimates in the INMODEL= data set as initial values. Then PROC IRT scores the DATA= data set by using the new parameter estimates obtained in the previous step.
- If you specify the INMODEL(SCORE)= option and the OUT= option, PROC IRT scores the DATA= data set by using the model specifications and parameter estimates in the INMODEL= data set without refitting the model.

ITEMFIT

displays the item fit statistics. These item fit statistics apply only to binary items that have one latent factor.

ITEMSTAT < (*itemstat-options*) >

displays the classical item statistics, which include the item means, item-total correlations, adjusted item-total correlations, and item means for *i* ordered groups of observations or individuals. You can specify the following *itemstat-options*:

NPARTITION=*i*

specifies the number of groups, where *i* must be an integer between 2 and 5, inclusive. By default NPARTITION=4.

The *i* ordered groups are formed by partitioning subjects based on the rank of their sum scores. By default, there are four groups, labeled G1, G2, G3, and G4, representing four ascending ranges of sum scores. The formula for calculating group values is

$$\text{floor}(\text{rank} \times i/(n+1))$$

where floor is the floor function, *rank* is the sum score's order rank, *i* is the value of the NPARTITION= option, and *n* is the number of observations that have nonmissing values of sum scores for TIES=LOW, TIES=MEAN, and TIES=HIGH. For TIES=DENSE, *n* is the number of observations that have unique nonmissing sum scores. If the number of observations is evenly divisible by the number of groups, each group has the same number of observations, provided that there are no tied sum scores at the boundaries of the groups. Sum scores with many tied values can create unbalanced groups because observations that have the same sum scores are assigned to the same group.

TIES=HIGH | LOW | MEAN | DENSE

specifies how to compute normal scores or ranks for tied data values.

HIGH	assigns the largest of the corresponding ranks.
LOW	assigns the smallest of the corresponding ranks.
MEAN	assigns the mean of the corresponding rank.
DENSE	computes scores and ranks by treating tied values as a single-order statistic. For the default method, ranks are consecutive integers that begin with the number 1 and end with the number of unique, nonmissing values of the variable that is being ranked. Tied values are assigned the same rank.

By default, TIES=MEAN.

Observations (subjects) that have missing values are excluded from the computations of the classical item statistics.

LINK=*name*

specifies the link function. You can specify the following *names*:

LOGIT requests the logistic link function.

PROBIT requests the probit link function.

By default, LINK=LOGIT.

MAXFUNC=*n*

MAXFU=*n*

specifies the maximum number of function calls in the optimization process. This option is not used by the EM algorithm. The default values are as follows, depending on which optimization technique is specified in the TECHNIQUE= option:

- NRRIDG: 125
- QUANEW: 500
- CONGRA: 1000

The optimization can terminate only after completing a full iteration. Therefore, the number of function calls that are actually performed can exceed the number that this option specifies.

MAXITER=*n*

MAXIT=*n*

specifies the maximum number of iterations in the optimization process. The default values are as follows, depending on which optimization technique is specified in the TECHNIQUE= option:

- NRRIDG: 50
- QUANEW: 200
- CONGRA: 400
- EM: 500

MAXMITER=*n*

MAXMIT=*n*

specifies the maximum number of iterations in the maximization step of the EM algorithm. By default, MAXMITER=1.

NFACTOR=*i*

NFACT=*i*

specifies the number of factors, *i*, in the model. You must specify the number of factors only for exploratory analysis, in which all the slope parameters of the items are freely estimated without being explicitly constrained by using the FACTOR statement. By default, NFACTOR=1. When you use the FACTOR statement to specify the confirmatory factor pattern, the number of factors is implicitly defined by the number of distinctive factor names that you specify in the statement.

NOAD
requests that the Gaussian quadrature be nonadaptive.

NOITPRINT
suppresses the display of the "Iteration History" table.

NOPRINT
suppresses all output displays.

OUT=*SAS-data-set*
creates an output data set that contains all the data in the DATA= data set plus estimated factor scores. For exploratory analysis, the factor scores are named _Factor1, _Factor2, and so on. For confirmatory analysis, user-specified factor names are used.

PROC IRT provides three estimation methods for factor scores. You can specify a method by using the SCOREMETHOD option. The default estimation method, maximum a posteriori (MAP), is used if the SCOREMETHOD option is not specified.

OUTMODEL=*SAS-data-set*
creates an output data set that contains the model specification, the parameter estimates, and their standard errors. You can use an OUTMODEL= data set as an input INMODEL= data set in a subsequent analysis by PROC IRT.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PINITIAL
displays the initial parameter estimates.

PLOTS <(*global-plot-options***) > < = *plot-request* < (***options***) > >**

PLOTS <(*global-plot-options***) > < = (***plot-request* < (***options***) > < . . . *plot-request* < (***options***) > >) >**
controls the plots that are produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around it. For example:

```
plots=all
plots=ICC(unpack)
plots(unpack)=(scree ICC)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc irt plots=all;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *global-plot-options*, which apply to all plots that the IRT procedure generates:

UNPACK | UNPACKPANEL
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACK to display each plot individually. You can also specify UNPACK as a suboption in the ICC, IIC, and SCREE options.

XVIEWMAX
specifies a maximum value for the X axis. You can also specify XVIEWMAX as a suboption in the ICC, IIC, and TIC options.

XVIEWMIN
specifies a minimum value for the X axis. You can also specify XVIEWMIN as a suboption in the ICC, IIC, and TIC options.

You can specify the following *plot-requests*:

ALL
displays all default plots.

ICC < (UNPACK | UNPACKPANEL), (XVIEWMAX=), (XVIEWMIN=) >
displays item characteristic curves (ICCs). By default, multiple ICC plots appear in some output panels. You can request an individual ICC plot for each item by specifying the UNPACK suboption. For binary items, the ICC plot includes only the curve for the higher category, which is often the correct response category or the endorsed category. For ordinal items that have more than two categories, the ICC plot includes curves for all the categories and also a legend to indicate the curves for different categories. For the default packed panel, the legend has values 1, 2, 3 an so on. For the unpacked individual ICC plot for each item, the legend uses the actual values of the corresponding categories.

IIC < (UNPACK | UNPACKPANEL), (XVIEWMAX=), (XVIEWMIN=) >
displays item information curves (IICs). By default, multiple IIC plots appear in some output panels. You can request an individual IIC plot for each item by specifying the UNPACK suboption.

NONE
suppresses all plots.

POLYCHORIC < *options* >
PLCORR< *options* >
displays a heat map of the polychoric correlation matrix. You can specify one or both of the following *options*:

FUZZ=*p*
displays polychoric correlations whose absolute values are less than *p* as 0 in the heat map. This option is useful when you want to focus on the patterns of sizable correlations that are larger than *p* in the heat map. By default, FUZZ=0.

OUTLINE=ON | OFF
specifies whether to display an outline of the regions in the polychoric correlation heat map. By default, OUTLINE=ON.

SCREE < (UNPACK | UNPACKPANEL) >
displays the scree and variance-explained plots in the same panel. You can display these plots individually by specifying the UNPACK suboption.

TIC < (XVIEWMAX=), (XVIEWMIN=) >
displays a test information curve (TIC) plot.

POLYCHORIC
displays the polychoric correlation matrix.

QPOINTS=*i*
specifies the number of quadrature points in each dimension of the integral. If there are d latent factors and n quadrature points, the IRT procedure evaluates n^d conditional log likelihoods for each observation to compute one value of the objective function. Increasing the number of quadrature nodes can substantially increase the computational burden. If you do not specify the number of quadrature points, it is determined adaptively by using the initial parameter estimates.

RCONVERGE=*p*
RCONV=*p*
specifies the convergence criterion for rotation cycles. Rotation stops when the scaled change of the simplicity function value is less than the RCONVERGE= value. The default convergence criterion is

$$|f_{new} - f_{old}|/K < \epsilon$$

where f_{new} and f_{old} are simplicity function values of the current cycle and the previous cycle, respectively; $K = \max(1, |f_{old}|)$ is a scaling factor; and ϵ is 1E–9 by default and is modified by the RCONVERGE= value.

RESFUNC=*name*
specifies the response functions for the variables that are included in the VAR statement. The response functions correspond to different response models. You can specify the following values:

FOURP	specifies the four-parameter model.
GPC	specifies the generalized partial credit model.
GRADED | GR	specifies the graded response model.
NOMINAL | NR	specifies the nominal response model.
ONEP	specifies the one-parameter model.
RASCH	specifies the Rasch model.
THREEP	specifies the three-parameter model.
TWOP	specifies the two-parameter model.

By default, RESFUNC=TWOP for binary items and RESFUNC=GRADED for ordinal items. The graded response model assumes that the response variables are ordinal-categorical up to 11 levels. All other models assume binary responses. Except for the generalized partial credit (GPC) model and Rasch (RASCH) model, you can use the RESFUNC= option in the MODEL statement to fit different response models to different items.

For more information about these response models, see the section "Response Models" on page 5455.

RITER=*n*

specifies the maximum number of cycles for factor rotation. The default value is the maximum between 10 times the number of variables and 100.

RORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the response variable. This order determines which threshold parameter in the model corresponds to each level in the data. If RORDER=FORMATTED for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. This option applies to all the responses in the model. When the default, RORDER=FORMATTED, is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. You can specify the following sort orders:

Value of RORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels that contain the most observations come first in the order
INTERNAL	Unformatted value

For FORMATTED and INTERNAL, the sort order is machine-dependent. For more information about sort order, see the chapter on the SORT procedure in the *SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

ROTATE=*name*

R=*name*

specifies the rotation method.

You can specify the following orthogonal rotation methods:

BIQUARTIMAX | BIQMAX specifies orthogonal biquartimax rotation.

EQUAMAX | E specifies orthogonal equamax rotation.

NONE | N specifies that no rotation be performed, leaving the original orthogonal solution.

PARSIMAX | PA specifies orthogonal parsimax rotation.

QUARTIMAX | QMAX | Q specifies orthogonal quartimax rotation.

VARIMAX | V specifies orthogonal varimax rotation.

You can specify the following oblique rotation methods:

BIQUARTIMIN | BIQMIN	specifies biquartimin rotation.
COVARIMIN | CVMIN	specifies covarimin rotation.
OBBIQUARTIMAX | OBIQMAX	specifies oblique biquartimax rotation.
OBEQUAMAX | OE	specifies oblique equamax rotation.
OBPARSIMAX | OPA	specifies oblique parsimax rotation.
OBQUARTIMAX | OQMAX	specifies oblique quartimax rotation.
OBVARIMAX | OV	specifies oblique varimax rotation.
QUARTIMIN | QMIN	specifies quartimin rotation.

By default, ROTATE=VARIMAX.

SCOREMETHOD=ML | EAP | MAP

specifies the method of factor score estimation. You can specify the following methods:

ML	requests the maximum likelihood method.
EAP	requests the expected a posteriori method.
MAP	requests the maximum a posteriori method.

By default, SCOREMETHOD=MAP.

SLOPEPRIOR=(< *r1* >, < *r2* >)

specifies the mean (*r1*) and the variance (*r2*) for the normal prior distribution of the slope parameters. If you want to minimize the influence of the prior information of the slope parameter, set the variance, *r2*, to a large value. PROC IRT uses these values to define the normal prior distribution for all the slope parameters of the items that are fit by the three- or four-parameter model (that is, when RESFUNC=THREEP or FOURP). This prior distribution is not used for items that are fit by other models, such as the two-parameter model. If you want to specify different prior information for different items, use the SLOPEPRIOR= option in the MODEL statement. For more information, see the section "Prior Distributions for Parameters" on page 5460.

The following example specifies the mean and variance to be 1 and 2, respectively, for the prior distribution of the slope parameters:

```
proc irt slopeprior=(1,2);
run;
```

By default, *r1* = 0 and *r2* = 1.

TECHNIQUE=CONGRA | EM | NONE | NRRIDG | QUANEW

TECH=CONGRA | EM | NONE | NRRIDG | QUANEW

OMETHOD=CONGRA | EM | NONE | NRRIDG | QUANEW

specifies the optimization technique to obtain maximum likelihood estimates. You can specify the following techniques:

CONGRA performs a conjugate-gradient optimization.

EM performs an EM optimization.

NONE performs no optimization.

NRRIDG performs a Newton-Raphson optimization with ridging.

QUANEW performs a dual quasi-Newton optimization.

By default, TECHNIQUE=QUANEW.

For more information about these optimization methods (except EM), see the section "Choosing an Optimization Algorithm" on page 545 in Chapter 20, "Shared Concepts and Topics." For more information about the EM algorithm, see "Expectation-Maximization (EM) Algorithm" in the section "Details: IRT Procedure" on page 5452.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC IRT to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the IRT procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Because sorting the data changes the order in which PROC IRT reads observations, the sort order for the levels of the response variables might be affected if you also specify RORDER=DATA in the PROC IRT statement.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

COV Statement

COV *assignment* < **,** *assignment* . . . > **;**

where *assignment* represents

var-list < ∗ *var-list2* > < **=** *parameter-spec* >

The COV statement defines the factor covariances in confirmatory models. In each *assignment* of the COV statement, you specify variables in the *var-list* and *var-list2* lists, followed by the covariance parameter specification in the *parameter-spec* list. The last two specifications are optional.

You can specify the following five types of the parameters for the covariances:

- an unnamed free parameter
- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

Consider a multidimensional model that has the latent factors FACTOR1, FACTOR2, FACTOR3, and FACTOR4. The following COV statement shows the five types of specifications in five *assignments*:

```
cov FACTOR2 FACTOR1 ,
    FACTOR3 FACTOR1 = (0.3),
    FACTOR3 FACTOR2 = 1.0,
    FACTOR4 FACTOR1 = phi1,
    FACTOR4 FACTOR2 = phi2(0.2);
```

In this statement, **cov(FACTOR2,FACTOR1)** is specified as an unnamed free parameter, **cov(FACTOR3,FACTOR1)** is an unnamed free parameter but with an initial value of 0.3, and **cov(FACTOR3,FACTOR2)** is a fixed value of 1.0. This value stays the same in the estimation. **cov(FACTOR4,FACTOR1)** is a free parameter named phi1, and **cov(FACTOR4,FACTOR2)** is a free parameter named phi2 that has an initial value of 0.2.

Note that the *var-list* and *var-list2* lists to the left of the equal sign in the COV statement should contain only names of latent factors that are specified in the FACTOR statement.

If you specify only the *var-list* list, then you are specifying the so-called within-list covariances. If you specify both the *var-list* and *var-list2* lists, then you are specifying the so-called between-list covariances. An asterisk is used to separate the two variable lists. You can use one of these two alternatives to specify the covariance parameters. Figure 72.9 illustrates the within-list and between-list covariance specifications.

Figure 72.9 Within-List and Between-List Covariances

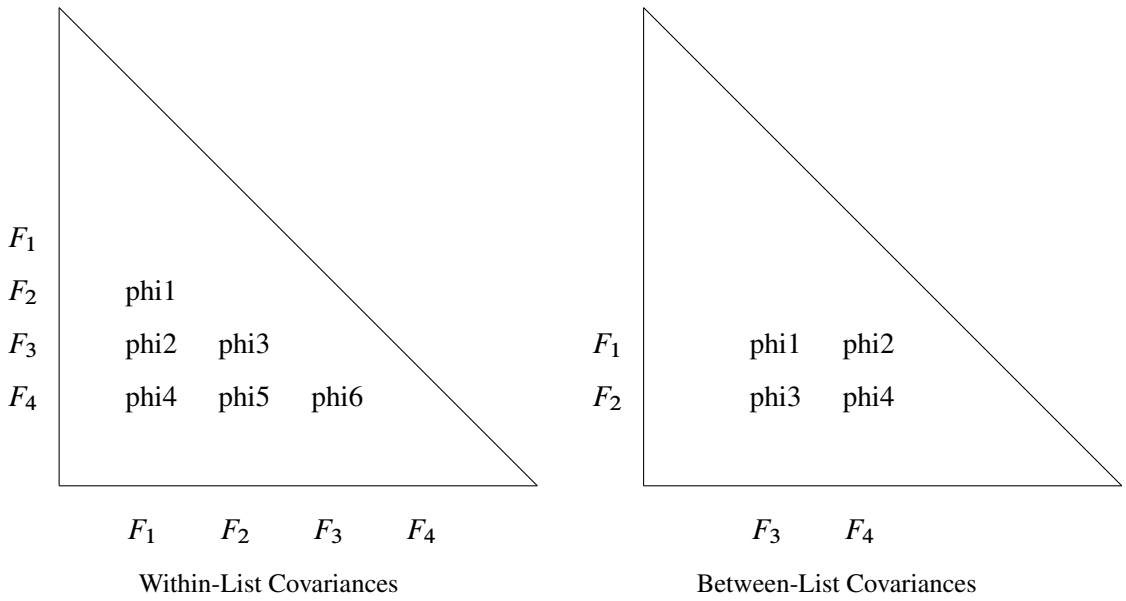

Within-List Covariances

The left panel of Figure 72.9 shows that the same set of four factors is used in both the rows and the columns. This yields six nonredundant covariances (variances are not included) to specify. In general, for a *var-list* list that has k variables in the COV statement, you can specify $k(k-1)/2$ distinct covariance parameters. The variable order of the *var-list* list is important. For example, the left panel of Figure 72.9 corresponds to the following COV statement:

```
cov F1-F4 = phi1-phi6;
```

This statement is equivalent to the following statement:

```
cov F2 F1 = phi1,
    F3 F1 = phi2, F3 F2 = phi3,
    F4 F1 = phi4, F4 F2 = phi5, F4 F3 = phi6;
```

Another way to assign distinct parameter names that have the same prefix is to use the so-called prefix name. For example, the following COV statement is exactly the same as the preceding statement:

```
cov F1-F4 = 6*phi__; /* phi with two trailing underscores */
```

In the COV statement, phi_ _ is a prefix name that has the root phi. The notation `6*` means that this prefix name is applied six times, resulting in a generation of the six parameter names phi1, phi2, . . . , phi6 for the six covariance parameters.

The root of the prefix name should have only a few characters so that the generated parameter name is not longer than 32 characters. To avoid unintentional equality constraints, the prefix names should not conflict with other parameter names.

You can also specify the within-list covariances as unnamed free parameters, as shown in the following statement:

```
cov F1-F4;
```

This statement is equivalent to the following statement:

```
cov F2 F1,
    F3 F1, F3 F2,
    F4 F1, F4 F2, F4 F3;
```

Between-List Covariances

The right panel of Figure 72.9 illustrates the application of the between-list covariance specification. The set of row variables is different from the set of column variables. You intend to specify the cross covariances of the two sets of variables. There are four of these covariances in the figure. In general, for k_1 and k_2 variable names in the two variable lists (separated by an asterisk) in a COV statement, there are $k_1 \times k_2$ distinct covariances to specify. Again, variable order is very important. For example, the right panel of Figure 72.9 corresponds to the following between-list covariance specification:

```
cov F1 F2 * F3 F4 = phi1-phi4;
```

This is equivalent to the following statement:

```
cov  F1 F3 = phi1, F1 F4 = phi2,
     F2 F3 = phi3, F2 F4 = phi4;
```

You can also use the prefix name specification for the same specification, as shown in the following statement:

```
cov  F1 F2 * F3 F4 = 4*phi__ ; /* phi with two trailing underscores */
```

Mixed Parameter Lists

You can specify different types of parameters for the list of covariances. For example, you use a list of parameters that have mixed types in the following statement:

```
cov F1-F4 = phi1(0.1)  0.2  phi3  phi4(0.4) (0.5) phi6;
```

This statement is equivalent to the following statement:

```
cov F2 F1 = phi1(0.1) ,
    F3 F1 = 0.2       , F3 F2 = phi3,
    F4 F1 = phi4(0.4) , F4 F2 = (0.5), F4 F3 = phi6;
```

Notice that an initial value that follows a parameter name is associated with the free parameter. Therefore, in the original mixed list specification, 0.1 is interpreted as the initial value of the parameter phi1, but not as the initial estimate of the covariance between F3 and F1. Similarly, 0.4 is the initial value of the parameter phi4, but not the initial estimate of the covariance between F4 and F2.

However, if you indeed want to specify that phi1 is a free parameter *without* an initial value and 0.1 is an initial estimate of the covariance between F3 and F1 (while keeping all other things the same), you can use a null initial value specification for the parameter phi1, as shown in the following statement:

```
cov F1-F4 = phi1() (0.1)  phi3  phi4(0.4) (0.5) phi6;
```

This way, 0.1 becomes the initial estimate of the covariance between F3 and F1. Because a parameter list that has mixed types might be confusing, you can break down the specifications into separate *assignments* to remove ambiguities. For example, you can use the following equivalent statement:

```
cov F2 F1 = phi1 ,
    F3 F1 = (0.1)     , F3 F2 = phi3,
    F4 F1 = phi4(0.4) , F4 F2 = (0.5), F4 F3 = phi6;
```

Shorter and Longer Parameter Lists

If you provide fewer parameters than the number of covariances in the variable lists, all the remaining parameters are treated as unnamed free parameters. For example, the following statement assigns a fixed value to **cov(F1,F3)** while treating all the other three covariances as unnamed free parameters:

```
cov  F1 F2 * F3 F4 = 1.0;
```

This statement is equivalent to the following statement:

```
cov  F1 F3 = 1.0, F1 F4, F2 F3, F2 F4;
```

If you intend to fill up all values by the last parameter specification in the list, you can use the continuation syntax **[...]**, **[..]**, or **[.]**, as in the following example:

```
cov  F1 F2 * F3 F4 = 1.0 phi [...];
```

This means that **cov(F1,F3)** is a fixed value of 1 and all the remaining three covariances are free parameters named phi. The last three covariances are thus constrained to be equal by having the same parameter name.

However, you must be careful not to provide too many parameters. For example, the following statement results in an error:

```
cov  F1 F2 * F3 F4 = 1.0 phi2(2.0) phi3 phi4 phi5 phi6;
```

The parameters after phi4 are excessive.

Default Covariance Parameters

In exploratory analysis, all factor covariances are fixed at zero for the unrotated or orthogonally rotated solutions. For confirmatory analysis, by default all factor covariances are fixed at zero. You can also use the COV statement to override these default covariance parameters in situations where you want to set parameter constraints or provide initial or fixed values.

EQUALITY Statement

EQUALITY | EQCON *equality-constraint* < **,** *equality-constraint . . .* > **;**

where a *equality-constraint* is defined as

variable-list < / *constraint-options* >

The EQUALITY statement provides a versatile way to specify various types of equality constraints on the parameters in the model. You can specify within-group or between-group equality constraints on specific sets of parameters for particular sets of variables or factors. In the *variable-list*, you specify the set of variables that are subject to the equality constraints on their respective parameters. You can either specify the names of the variables or use one of the supported keywords (see list later in this section) for *variable-list*. In the *constraint-options*, you specify the types of parameters, the specific groups (in multiple-group analysis), and the specific factors (in multidimensional models) on which the equality constraints are imposed. These equality constraints apply to the same type of parameters among variables specified in the *variable-list*.

For example, the following statements specify that all related parameters of x1 through x5 are constrained to be equal:

```
proc irt;
   model x1-x10/resfunc=graded;
   equality x1-x5;
run;
```

Because all items are fit by the graded response model, all slopes for variables x1–x5 are constrained to be the same and the intercepts for variables x1–x5 are also constrained to be the same. Because equality constraints specified in the EQUALITY statement apply to parameters among variables rather than within variables, parameters within each variable are different. For example, if each of these variables has five categories, there would be five set of constraints for the slope parameter and for each of the four intercept parameters over the five variables.

You can limit the set of parameters for the equality constraints by specifying the PARM= option (one of the *constraint-options*). For example, the following statements constrain only the slope parameters of x1–x5, instead of all related parameters in the graded response model:

```
proc irt;
   model x1-x10/resfunc=graded;
   equality x1-x5/parm=[slopes];
run;
```

If the set of parameters of the same type in the *equality-constraint* contains fixed-value parameters that have the same value, all the parameters in this set are fixed to the same value. If these fixed-value parameters have different values, PROC IRT returns an error message and stops. If the set of parameters contains model-restricted parameters that are specified in the FACTOR statement, these model-restricted parameters are ignored and the equality constraint applies to all the other parameters in the set.

In the following example, the FACTOR statement specifies a simple structure model that has two factors, where x1 to x5 load on the first factor f, x6 to x10 load on the second factor g, and the slope parameter of x6 on factor g is fixed to 1.

```
proc irt;
   factor f->x1-x5, g->x6-x10=1;
   equality x1-x10;
run;
```

The EQUALITY statement constrains all the parameters among x1 to x10 to be the same. Because the loading parameters of x1 to x5 on factor g and the loading parameters of x6 to x10 on factor f are model-restricted parameters, they are excluded from the constraints that are specified in the EQUALITY statement. Three sets of constrained parameters are specified in this example:

- The intercept parameters of x1 to x10 are the same.
- The slope (loading) parameters of x1 to x5 on factor f are the same.
- The slope (loading) parameters of x6 to x10 on factor g are the same. Because the slope parameter of x6 on factor g is fixed to 1, this set of parameters (that is, the slopes that are related to factor g) are all fixed to 1.

There are various ways to specify the target set of variables that are subject to the equality constraints. You can specify the variables directly, or you can specify one of the following keywords for *variable-list*:

ALL
specifies all items and factors in the analysis. Equality constraints on parameters related to factors apply only to multiple-group analysis.

ALLFACTOR
specifies all latent factors in the analysis.

ALLFOURP
ALLFOURPITEM
specifies all items that are fit by the four-parameter model in the analysis.

ALLGPC
ALLGPCITEM
specifies all items that are fit by the generalized partial credit model in the analysis.

ALLGR
ALLGRITEM
specifies all items that are fit by the graded response model in the analysis.

ALLITEM
specifies all items in the analysis.

ALLNR
ALLNRITEM
specifies all items that are fit by the nominal response model in the analysis.

ALLONEP
ALLONEPITEM
specifies all items that are fit by the one-parameter model in the analysis.

ALLRASCH
ALLRASCHITEM
specifies all items that are fit by the Rasch model in the analysis.

ALLTHREEP
ALLTHREEPITEM
specifies all items that are fit by the three-parameter model in the analysis.

ALLTWOP
ALLTWOPITEM
specifies all items that are fit by the two-parameter model in the analysis.

You can also specify the following keywords, with a list of *excluded-variables* for *variable-list*:

_ALL_BUT_ [*excluded-variables* **]**
specifies all items and factors except the *excluded-variables* in the analysis.

_ALLFACTOR_BUT_ [*excluded-factors* **]**
specifies all factors except the *excluded-factors* in the analysis.

_ALLFOURP_BUT_ [*excluded-items* **]**
_ALLFOURPITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the four-parameter model in the analysis.

_ALLGPC_BUT_ [*excluded-items* **]**
_ALLGPCITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the generalized partial credit model in the analysis.

_ALLGR_BUT_ [*excluded-items* **]**
_ALLGRITEM_BUT_[*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the graded response model in the analysis.

_ALLITEM_BUT_ [*excluded-items* **]**
specifies all items except the *excluded-items* in the analysis.

_ALLNR_BUT_ [*excluded-items* **]**
_ALLNRITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the nominal response model in the analysis.

_ALLONEP_BUT_ [*excluded-items* **]**

_ALLONEPITEM_BUT_ [*excluded-items* **]**

specifies all items (except the *excluded-items*) that are fit by the one-parameter model in the analysis.

_ALLRASCH_BUT_ [*excluded-items* **]**

_ALLRASCHITEM_BUT_ [*excluded-items* **]**

specifies all items (except the *excluded-items*) that are fit by the Rasch model in the analysis.

_ALLTWOP_BUT_ [*excluded-items* **]**

_ALLTWOPITEM_BUT_ [*excluded-items* **]**

specifies all items (except the *excluded-items*) that are fit by the two-parameter model in the analysis.

_ALLTHREEP_BUT_ [*excluded-items* **]**

_ALLTHREEPITEM_BUT_ [*excluded-items* **]**

specifies all items (except the *excluded-items*) that are fit by the three-parameter model in the analysis.

For example, if you have mixed model types for the item responses, the equality constraints might be set on a particular set of response variables. The following example shows that the equality constraints are applied to those variables that are fit by the three-parameter model (that is, x7–x10):

```
proc irt;
   model x1-x6/resfunc=graded,
         x7-x10/resfunc=threep;
   equality _allthreep_;
run;
```

Suppose that the preceding model does not fit well and you want to consider a less restricted model in which the equality constraints are imposed on all variables except x10 in the three-parameter model. The following statements achieve this purpose:

```
proc irt;
   model x1-x6/resfunc=graded,
         x7-x10/resfunc=threep;
   equality _allthreep_but_(x10);
run;
```

In the *constraint-options*, you can specify options for parameter types (PARM= option), the set of groups (BETWEEN_GP= and WITHIN_GP= options), and the set of factors. If you do not use these options, all related parameter types, all groups, and all factors are subject to the constraints for the specified set of variables. You can specify the following *constraint-options*:

BET < = [*group-list* **] >**

BETWEEN < = [*group-list* **] >**

BETWEEN_GP< = [*group-list* **] >**

specifies that the equality constraints be applied across or between groups in the multiple-group analysis. Setting between-group constraints is the default when you fit multiple-group models. Hence, it is not necessary to use this option when you want to set equality constraints between all groups. When only a subset of groups is subject to the intended constraints, you can specify the groups in the *group-list*. This option has no effect if you have only one group in the analysis.

PARM = [*parameter-types***]**

specifies the particular types of parameters that are subject to equality constraints. By default, all related parameters are subject to the constraints. You can specify the following *parameter-types*:

CEIL

CEILING

constrains the ceiling parameters.

COVARIANCE

COV

constrains the covariance parameters. This option only applies to latent factors. This option sets equality constraints for all the covariance parameters among the set of factors specified in the *variable-list*. You cannot use this option to specify equality constraints on a subset of covariance parameters among a set of factors. To set such equality constraints, you can use the COV statement.

GUESS

GUESSING

constrains the guessing parameters.

INTERCEPT

constrains the intercept parameters. For unidimensional models, the difficulty or threshold parameter rather than the intercept parameters are displayed in the parameter estimates table. Intercept parameter equals to difficulty or threshold parameter times the negative slope parameter.

MEAN

constrains the mean parameters. This option only applies to latent factors.

SLOPE < [*factor-list* **] >**

DISCRIMINATION < [*factor-list* **] >**

constrains the slope or discrimination parameters. The optional *factor-list* indicates the set of factors to which the constrained slope parameters pertain. The use of *factor-list* is relevant only when you conduct a confirmatory analysis by specifying the factor pattern in the FACTOR statement.

VARIANCE

VAR

constrains the variance parameters. This option only applies to latent factors.

WIT < = [*group-list* **] >**

WITHIN < = [*group-list* **] >**

WITHIN_GP< = [*group-list* **] >**

specifies that the equality constraints be applied within groups in multiple-group analyses. Setting within-group constraints is the default when you fit a single-group model. Hence, this option is not necessary when you have only one group in the analysis. In multiple-group analyses, between-group constraints are set by default. When you specify this option, within-group constraints are set instead of between-group constraints. You can also specify the specific groups in the *group-list* that are subject to the within-group constraints. The default is to apply the equality constraints to all groups.

You can combine the *constraint-options* to set various types of constraints for your model. You can also specify more than one constraint in an EQUALITY statement. You can even use multiple EQUALITY statements for better organization of the constraints.

For example, suppose that a single-group analysis is conducted using three different types of models (two-parameter, graded responses, and three-parameter model) for the response variables. Consider the following statements:

```
proc irt;
   model x1-x10/resfunc=twop, x11-x20/resfunc=graded, x21-x30/resfunc=threep;
   equality _alltwop_but_(x9-x10),
         x11-x25 / parm=[slope],
         _allthreep_ / parm=[guess];
run;
```

The first set of equality constraints applies to the intercept and slope parameters of x1–x8, leaving the parameters of x9 and x10 freely estimated. The second set of equality constraints applies to the slope parameters of variables x11–x25, even though x21–x25 have a different model type than x11–x20. The third set of equality constraints applies to the guessing parameters of all variables that are fit by the three-parameter model (that is, x21–x30).

In multiple-group analysis, constraints are set across groups by default. But within-group constraints can also be set by using the WITHIN_GP option. Suppose there are three groups in the analysis and the grouping variable GP has three distinct values, 1, 2, and 3. Consider the following example:

```
proc irt;
   group GP;
   model x1-x10/resfunc=twop,
         x11-x20/resfunc=graded,
         x21-x30/resfunc=threep;
   equality _alltwop_but_(x9-x10),
         x11-x25 / parm=[slope] between_gp=[1 2],
         _allthreep_ / parm=[guess] within_gp=[1 3];
run;
```

This example is quite similar to the preceding example, but with some modifications from using the BETWEEN_GP and WITHIN_GP options.

The first set of equality constraints is specified exactly the same way as in the preceding example. However, the effect is much different. In the current multiple-group example, the specification constrains the parameters across groups by default. This means that the intercept and slope parameters of x1–x8 are constrained over the three groups. So there would be 16 sets of equality constraints, respectively, for the 16 parameters in variables over the three groups. However, if you use the WITHIN_GP option, the parameters for x1–x8 are the same within each group. This results in three separate sets of equality constraints on 16 intercept parameters, respectively, for the three groups. Moreover, if you use the WITHIN_GP and BETWEEN_GP options together, all 48 parameters in the groups are constrained to be the same.

The second set of equality constraints applies to the slope or discrimination parameters of variables x11–x25 across groups 1 and 2 only, but not all groups. This means that there are 15 equality constraints, respectively, for the 15 slope or discrimination parameters in variables across groups 1 and 2. The discrimination parameters for these variables are not constrained within groups.

The third set of equality constraints applies to the guessing parameters of variables x21–x30 (that is, all the variables that are fit by the three-parameter model) within groups 1 and 3, respectively. The guessing

parameters for these variables are not constrained across groups.

FACTOR Statement

FACTOR *factor-variables-relation* < **,** *factor-variables-relation . . .* > **;**

where each *factor-variables-relation* is defined as

factor right-arrow var-list < **=** *parameter-spec* >

where *factor* is a name that represents an intended factor; *right-arrow* is **===>**, **--->**, **==>**, **-->**, **=>**, **->**, or **>**; *var-list* is a list of variables that have nonzero slopes associated with the factor; and *parameter-spec* represents the specifications of parameter name and values (fixed or initial).

You use the FACTOR statement to specify the pattern of relationships between variables and factors in confirmatory models. You do not need to use the FACTOR statement if you are fitting an exploratory model. To complete the specification of a confirmatory model, you might need to use the VARIANCE and COV statements to specify the variance and covariance parameters in the model, as shown in the following syntax:

FACTOR *factor-variable-relation* < **,** *factor-variables-relation . . .* > **;**
VARIANCE *variance-parameters* **;**
COV *covariance-parameters* **;**

By default, the factor variances are fixed parameters with a value of 1 in the confirmatory factor model. However, you can override these default parameters by specifying them explicitly in the VARIANCE statement. For example, in some confirmatory factor models, you might want to free some of these factor variances, or you might want to set equality constraints by using the same parameter name at different parameter locations in your model. If you free some of the factor variances, you need to fix some slope parameters in order to identify the model.

By default, factor covariances are zeros in the confirmatory model. However, you can override these default covariance parameters by specifying them explicitly in the COV statement.

Because the default parameterization of the confirmatory model already covers most commonly used parameters, the specifications in the VARIANCE and COV statements are secondary to the specifications in the FACTOR statement, which specifies the pattern of the slope matrix. All the slope parameters that are fixed to 0 in the FACTOR statement are model-restricted parameters. Any equality constraints on these model-restricted parameters (as specified in the EQUALITY statement) are ignored. The following statement illustrates the syntax of the confirmatory FACTOR statement. Suppose there are nine variables, V1–V9, in your sample and you want to fit a confirmatory IRT model that has four factors, as follows:

```
factor
   g_factor   ===>   V1-V9 ,
   factor_a   ===>   V1-V3 ,
   factor_b   ===>   V4-V6 ,
   factor_c   ===>   V7-V9 ;
```

In this factor model, you assume a general factor, g_factor, and three group factors, factor_a, factor_b, and factor_c. The general factor, g_factor, is related to all variables in the sample, whereas each group factor is related to only three variables. This example fits the following pattern of the slope matrix:

	g_factor	factor_a	factor_b	factor_c
V1	x	x		
V2	x	x		
V3	x	x		
V4	x		x	
V5	x		x	
V6	x		x	
V7	x			x
V8	x			x
V9	x			x

Here an x represents an unnamed free parameter, and all blank cells represent model-restricted parameters that are fixed to zeros.

You can specify the following five types of parameters (*parameter-spec*) at the end of each *factor-variables-relation*:

- an unnamed free parameter
- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

To illustrate these different types of parameter specifications, consider the following pattern of slopes:

	g_factor	factor_a	factor_b	factor_c
V1	g_load1	1.		
V2	g_load2	x		
V3	g_load3	x		
V4	g_load4		1.	
V5	g_load5		load_a	
V6	g_load6		load_b	
V7	g_load7			1.
V8	g_load8			load_c
V9	g_load9			load_c

Here an x represents an unnamed free parameter, a constant 1 represents a fixed value, and each name in a cell represents a name for a free parameter. You can specify this pattern by using the following FACTOR statement:

```
factor
   g_factor   ===>   V1-V9    = g_load1-g_load9 (9*0.6),
   factor_a   ===>   V1-V3    = 1. (.7  .8),
   factor_b   ===>   V4-V6    = 1. load_a (.9) load_b,
   factor_c   ===>   V7-V9    = 1. 2*load_c ;
```

In the first entry of the preceding FACTOR statement, you specify that the slopes of V1–V9 on g_factor are the free parameters g_load1–g_load9, all of which are given an initial estimate of 0.6. The syntax `9*0.6` means that `0.6` is repeated nine times. Because they are enclosed in parentheses, all these values are treated as initial estimates but not as fixed values.

You can split the second entry of the preceding FACTOR statement into the following specification:

```
factor_a   ===>   V1    = 1.  ,
factor_a   ===>   V2    = (.7),
factor_a   ===>   V3    = (.8),
```

This specification means that the first slope is a fixed value of 1 and that the other slopes are unnamed free parameters that have initial estimates of 0.7 and 0.8, respectively.

You can split the third entry of the preceding FACTOR statement into the following specification:

```
factor_b   ===>   V4    = 1.  ,
factor_b   ===>   V5    = load_a (.9),
factor_b   ===>   V6    = load_b,
```

This specification means that the first slope is a fixed value of 1, the second slope is a free parameter named load_a with an initial estimate of 0.9, and the third slope is a free parameter named load_b without an initial estimate. PROC IRT generates the initial value of this free parameter.

The fourth entry of the preceding FACTOR statement states that the first slope is a fixed 1 and the remaining two slopes are free parameters named load_c. No initial estimate is given. But because the two slopes have the same parameter name, they are constrained to be equal in the estimation.

Notice that an initial value that follows a parameter name is associated with the free parameter. For example, in the third entry of the preceding FACTOR statement, the specification `(.9)` after load_a is interpreted as the initial value of the parameter load_a, but not as the initial estimate of the next slope of V6.

However, if you indeed want to specify that load_a is a free parameter *without* an initial value and `(0.9)` is an initial estimate for the slope of V6, you can use a null initial value specification for the parameter load_a, as shown in the following specification:

```
factor_b   ===>   V4-V6    = 1. load_a() (.9),
```

This way, 0.9 becomes the initial estimate of the slope of V6. Because a parameter list that contains mixed parameter types might be confusing, you can split the specification into separate entries to remove ambiguities. For example, you can use the following equivalent specification:

```
factor_b   ===>   V4    = 1.,
factor_b   ===>   V5    = load_a,
factor_b   ===>   V6    = (.9),
```

Shorter and Longer Parameter Lists

If you provide fewer parameters than the number of slopes that are specified in the corresponding *factor-variable-relation*, all the remaining parameters are treated as unnamed free parameters. For example, the following statement assigns a fixed value of 1.0 to the first slope, while treating the remaining two slopes as unnamed free parameters:

```
factor
   factor_a   ===>   V1-V3    = 1.;
```

This statement is equivalent to the following statement:

```
factor
   factor_a   ===>   V1     = 1.,
   factor_a   ===>   V2 V3      ;
```

If you intend to fill up all values with the last parameter specification in the list, you can use the continuation syntax `[...]`, `[..]`, or `[.]`, as shown in the following example:

```
factor
   g_factor   ===>   V1-V30    = 1.  (.5) [...];
```

This means that the slope of V1 on g_factor is a fixed value of 1.0 and the remaining 29 slopes are unnamed free parameters, all of which are given an initial estimate of 0.5.

However, you must be careful not to provide too many parameters. For example, the following statement results in an error:

```
factor
   g_factor   ===>   V1-V3    = load1-load6;
```

The parameter list has six parameters for three slopes. Parameters after load3 are excessive.

FIXVALUE Statement

FIXVALUE *fixed-value-constraint* < **,** *fixed-value-constraint . . .* > **;**

where a *fixed-value-constraint* is defined as

variable-list < / *constraint-options* >

The FIXVALUE statement provides a versatile way to specify fixed value constraints on the parameters in the model. You can specify fixed value constraints on specific sets of parameters for particular sets of variables or factors within a set of groups. In the *variable-list*, you specify the set of variables that are subject to the fixed value constraints on their respective parameters. You can either specify the names of the variables or use one of the supported keywords (see list later in this section) for *variable-list*. In the *constraint-options*, you specify the types of parameters, the specific groups (in multiple-group analysis), the specific factors (in multidimensional models), and the specific intercepts on which the fixed value constraints are imposed.

For example, the following statements specify that all the slope parameters of x1 through x5 are fixed to 1:

```
proc irt;
   model x1-x10/resfunc=graded;
   fixvalue x1-x5/parm=[slope] value=[1];
run;
```

There are various ways to specify the target set of variables that are subject to the fixed value constraints. You can specify the variables directly, or you can specify one of the following keywords for *variable-list*:

ALL
specifies all items and factors in the analysis.

ALLFACTOR
specifies all latent factors in the analysis.

ALLFOURP
ALLFOURPITEM
specifies all items that are fit by the four-parameter model in the analysis.

ALLGPC
ALLGPCITEM
specifies all items that are fit by the generalized partial credit model in the analysis.

ALLGR
ALLGRITEM
specifies all items that are fit by the graded response model in the analysis.

ALLITEM
specifies all items in the analysis.

ALLNR
ALLNRITEM
specifies all items that are fit by the nominal response model in the analysis.

ALLONEP
ALLONEPITEM
specifies all items that are fit by the one-parameter model in the analysis.

ALLRASCH
ALLRASCHITEM
specifies all items that are fit by the Rasch model in the analysis.

ALLTHREEP
ALLTHREEPITEM
specifies all items that are fit by the three-parameter model in the analysis.

ALLTWOP
ALLTWOPITEM
specifies all items that are fit by the two-parameter model in the analysis.

You can also specify the following keywords, with a list of *excluded-variables* for *variable-list*:

_ALL_BUT_ [*excluded-variables* **]**
specifies all items and factors except the *excluded-variables* in the analysis.

_ALLFACTOR_BUT_ [*excluded-factors* **]**
specifies all factors except the *excluded-factors* in the analysis.

_ALLFOURP_BUT_ [*excluded-items* **]**
_ALLFOURPITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the four-parameter model in the analysis.

_ALLGPC_BUT_ [*excluded-items* **]**
_ALLGPCITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the generalized partial credit model in the analysis.

_ALLGR_BUT_ [*excluded-items* **]**
_ALLGRITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the graded response model in the analysis.

_ALLITEM_BUT_ [*excluded-items* **]**
specifies all items except the *excluded-items* in the analysis.

_ALLNR_BUT_ [*excluded-items* **]**
_ALLNRITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the nominal response model in the analysis.

_ALLONEP_BUT_ [*excluded-items* **]**
_ALLONEPITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the one-parameter model in the analysis.

_ALLRASCH_BUT_ [*excluded-items* **]**
_ALLRASCHITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the Rasch model in the analysis.

_ALLTHREEP_BUT_ [*excluded-items* **]**
_ALLTHREEPITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the three-parameter model in the analysis.

_ALLTWOP_BUT_ [*excluded-items* **]**
_ALLTWOPITEM_BUT_ [*excluded-items* **]**
specifies all items (except the *excluded-items*) that are fit by the two-parameter model in the analysis.

For example, if you have mixed model types for the item responses, the fixed value constraints might be set on a particular set of response variables. The following example shows that a fixed value of 1 is applied to the slopes of the variables that are fit by the three-parameter model (that is, x7–x10):

```
proc irt;
   model x1-x6/resfunc=graded,
         x7-x10/resfunc=threep;
   fixvalue _allthreep_/parm=[slope] value=[1];
run;
```

Suppose that the preceding model does not fit well and you want to consider a less restricted model in which the fixed slope constraints are imposed on all variables except x10 in the three-parameter model. The following statements achieve this purpose:

```
proc irt;
   model x1-x6/resfunc=graded,
         x7-x10/resfunc=threep;
   fixvalue _allthreep_but_(x10)/parm=[slope] value=[1];
run;
```

In the *constraint-options*, you can specify options for parameter types (in the PARM= option), the set of groups (in the GROUP= option), and the set of fixed values (in the VALUE= option). If you do not use these options, all related parameter types, groups, factors, and intercepts are subject to the same fixed-value constraints for the specified set of variables. You can specify the following *constraint-options*:

GROUP= [*group-list* **]**

GP= [*group-list* **]**

specifies the groups to which the fixed-value constraints apply, where *group-list* is a list of integers or character strings for the group identification. Applying fixed-value constraints to all groups is the default when you fit multiple-group models. When only a subset of groups is subject to the intended constraints, you can specify the groups in the *group-list*.

PARM= [*parameter-types***]**

specifies the particular types of parameters that are subject to fixed-value constraints. By default, all related parameters are subject to the specified constraints. You can specify the following *parameter-types*:

CEIL

CEILING

constrains the ceiling parameters.

COVARIANCE

COV

constrains the covariance parameters. This option applies only to latent factors; it sets fixed-value constraints for all the covariance parameters among the set of factors that are specified in the *variable-list*.

GUESS

GUESSING

constrains the guessing parameters.

INTERCEPT < [*integer-list* **] >**

constrains the intercept parameters. The optional *integer-list* specifies the set of intercepts to which the fixed-value constraints apply. Suppose there are five items in the model and each has five categories, which means that there are four intercept parameters for each item. In the following example, the first set of constraints in the FIXVALUE statement sets the first intercept parameter to 0 for all the items. The second set of constraints fixes the second and third intercept parameters of item x1 to be –1 and –2.

```
proc irt;
   var x1-x5;
   fixvalue _allitem_ / parm=[intercept[1]],
            x1 / parm=[intercept[2 3]] value=[-1 -2];
```

```
run;
```

PROC IRT does not support constraints on the difficulty or threshold parameter directly. The intercept parameter is equal to the difficulty or threshold parameter times the negative slope parameter.

MEAN
constrains the mean parameters. This option applies only to latent factors.

SLOPE < [*factor-list* | *integer-list* **] >**
DISCRIMINATION < [*factor-list* | *integer-list* **] >**
constrains the slope or discrimination parameters. The optional *factor-list* or *integer-list* indicates the set of factors to which the constrained slope parameters apply. *factor-list* specifies the names of the related factors for the constraint. The use of *factor-list* is relevant only when you conduct a confirmatory analysis by specifying the factor pattern in the FACTOR statement. The use of *integer-list* is relevant only with nominal response model.

VARIANCE
VAR
constrains the variance parameters. This option applies only to latent factors.

VALUE= [*value-list***]**
specifies a list of values in the *value-list* for the fixed-value constraints. By default, the *value-list* contains a single 0. If only one value is specified in *value-list*, this value applies to all the parameters in the constraint. If more values are specified in *value-list* than the number of parameters, extra values are ignored. If you specify more than one value in the *value-list* but there are fewer values in the list than the number of parameters, the extra parameters are fixed to 0.

You can combine the *constraint-options* to set various types of constraints for your model. You can also specify more than one constraint in an FIXVALUE statement. You can even use multiple FIXVALUE statements for better organization of the constraints.

For example, suppose that a single-group analysis is conducted using three different types of models (two-parameter, graded responses, and three-parameter model) for the response variables. Consider the following statements:

```
proc irt;
   model x1-x10/resfunc=twop, x11-x20/resfunc=graded, x21-x30/resfunc=threep;
   fixvalue _alltwop_but_(x9-x10)/parm=[intercept],
            x11-x25 / parm=[slope] value=[1],
            _allthreep_ / parm=[guess] value=[0.1];
run;
```

In the FIXVALUE statement, the first set of constraints fixes the intercept parameters of x1–x8 to 0, leaving the parameters of x9 and x10 to be freely estimated. The second set of constraints fixes the slope parameters of variables x11–x25 to 1, even though x21–x25 have a different model type than x11–x20 have. The third set of constraints fixes the guessing parameters to 0.1 for all variables that are fit by the three-parameter model (that is, x21–x30).

In multiple-group analysis, constraints are applied to all groups by default. Suppose there are three groups in the analysis and the grouping variable GP has three distinct values: 1, 2, and 3. Consider the following example:

```
proc irt;
   group GP;
   model x1-x10/resfunc=twop,
         x11-x20/resfunc=graded,
         x21-x30/resfunc=threep;
   fixvalue _alltwop_but_(x9-x10)/parm=[intercept],
         x11-x25 / parm=[slope] value=[1] group=[1 2],
         _allthreep_ / parm=[guess] value=[0.1] group=[1 3];
run;
```

This example is similar to the preceding example, but with some modifications from using the GROUP= options.

The first set of fixed-value constraints is specified exactly the same way as in the preceding example. In the current multiple-group example, the fixed value 0 applies to the intercept parameters of x1–x8 in all three groups.

The second set of constraints fixes the slope or discrimination parameters of variables x11–x25 to 1 for groups 1 and 2 only (not all groups).

The third set of constraints fixes the guessing parameters of variables x21–x30 (that is, all the variables that are fit by the three-parameter model) to 0.1 for groups 1 and 3.

FREQ Statement

FREQ *variable* **;**

If one variable in your data set represents the frequency of occurrence for the other values in the observation, specify the variable's name in a FREQ statement. PROC IRT then treats the data set as if each observation appears n_i times, where n_i is the value of the FREQ variable for observation i. Only the integer portion of the value is used. If the value of the FREQ variable is less than 1 or is missing, that observation is not included in the analysis. The total number of observations is considered to be the sum of the FREQ values.

GROUP Statement

GROUP *variable* **;**

The GROUP statement specifies the grouping variable that defines the groups of the observations. This statement is required if you intend to do a multiple-group analysis. The values of the grouping variable can be either integers or character strings. PROC IRT analyzes the input data set and determines the number of distinct groups in the data set by counting the number of distinct values in the grouping variable. Because there is no other explicit way to specify the number of groups or the grouping values, you must make sure that all (and only) the intended groups have been indexed properly by the grouping variable in the data set for a multiple-group analysis.

MEAN Statement

MEAN *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents:

var-list < **=** *parameter-spec* >

The MEAN statement specifies the factor mean parameters in connection with the FACTOR statement.

In each *assignment* of the MEAN statement, you list the *var-list* that you want to specify for their means. Optionally, you can provide a list of parameter specifications in a *parameter-spec* after an equal sign for each *var-list*. The syntax of the MEAN statement is exactly the same as that of the VARIANCE statement. For more information, see the VARIANCE statement on page 5449.

You can specify the following five types of the parameters for the means in the MEAN statement:

- an unnamed free parameter
- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

For example, consider a confirmatory factor model with latent factors F1, F2, F3, F4, F5. The following MEAN statement illustrates the five types of specifications in five *assignments*:

```
mean F1 ,
     F2 = (3.0),
     F3 = 1.5,
     F4 = fmean1,
     F5 = fmean2(0.6);
```

In this statement, the mean of F1 is specified as an unnamed free parameter. For this mean, PROC IRT generates a parameter name with the _Parm prefix appended with a unique integer (for example, _Parm1). The mean of F2 is an unnamed free parameter with an initial value of 3.0. PROC IRT also generates a parameter name for this mean. The mean of F3 is a fixed value of 1.5. This value stays the same during the estimation. The mean of F4 is a free parameter named fmean1. The mean of F5 is a free parameter named fmean2 with an initial value of 0.6.

If you do not use the MEAN statement, all the means of latent factors are fixed to 0 by default. For the confirmatory FACTOR models, you can override these default fixed 0 values by using the MEAN statement specifications. However, you cannot override the fixed 0 factor means for the exploratory FACTOR model.

MODEL Statement

MODEL *model-specification* < **,** *model-specification . . .* > **;**

where *model-specification* is defined as

variable-list < / *model-options* >

The MODEL statement specifies the items, their response functions or models, and prior information for guessing and ceiling parameters. You can specify different response models and prior information for different items. In the *variable-list*, you specify the set of variables that use the same model or prior information.

You can specify the following *model-options*:

CEILPRIOR=((< *r1* >,< *r2* >**)** < **,(**< *r1* >,< *r2* >**). . .** >**)**

CEILPRIOR=(< *r1* >,< *r2* >**)**

specifies the prior information for the ceiling parameters. Each pair optionally includes the mean (*r1*) and weight (*r2*) for the beta prior distribution. This option is similar to the CEILPRIOR= option in the PROC IRT statement. For the meaning and usage of the option, see the CEILPRIOR= option in the PROC IRT statement. You can specify multiple pairs of *r1* and *r2* in the CEILPRIOR= option in the MODEL statement, whereas you can specify only one pair of *r1* and *r2* in the PROC IRT statement.

The CEILPRIOR= option in the MODEL statement overwrites the default prior values and those that are specified in the PROC IRT statement. Both values in each pair are optional; if you do not specify them in this option, the values that are specified in the PROC IRT statement (or their default values) are used. If a single pair of values is specified, it applies to all the items specified in the *variable-list*. If the number of pairs specified is less than the number of items specified in the *variable-list*, the rest of the items use default settings. If more pairs are specified than the number of items in the *variable-list*, extra pairs are ignored. This option applies only to the items that are fit by the four-parameter model.

In the following example, prior information of the ceiling parameter is specified for three sets of items:

```
proc irt;
   model x1-x10/resfunc=twop ceilprior=(0.8,10),
         x11-x20/resfunc=fourp ceilprior=(0.9,),
         x21-x30/resfunc=fourp ceilprior=((0.9,10),(0.8,15));
run;
```

Because the first set of items, x1 to x10, are not fit by the four-parameter model, the specified ceiling prior information is ignored. For the second set of items, because only one pair of ceiling prior information is specified, it applies to all the items in this set. The second value for this pair is not specified, so the default weight (20) is used. Two pairs of ceiling prior information are specified for the 10 items in the last set, so the first pair applies to the first item (x21), and the second pair applies to the rest of the items (x22 to x30).

GUESSPRIOR=((< *r1* >,< *r2* >) < ,(< *r1* >,< *r2* >)... >)
GUESSPRIOR=(< *r1* >,< *r2* >)

specifies the prior information for the guessing parameters. Each pair includes the mean (*r1*) and weight (*r2*) for the beta prior distribution. This option is similar to the GUESSPRIOR= option in the PROC IRT statement. For the meaning and usage of the option, see the GUESSPRIOR= option in the PROC IRT statement. You can specify multiple pairs of *r1* and *r2* in the GUESSPRIOR= option in the MODEL statement, whereas you can specify only one pair of *r1* and *r2* in the PROC IRT statement.

The GUESSPRIOR= option in the MODEL statement overwrites the default prior values and those that are specified in the PROC IRT statement. Both values in each pair are optional; if you do not specify them in this option, the values that are specified in the PROC IRT statement (or their default values) are used. If a single pair of values is specified, it applies to all the items specified in the *variable-list*. If the number of pairs specified is less than the number of items specified in the *variable-list*, the default settings are applied to the rest of the items. If more pairs are specified than the number of items in the *variable-list*, extra pairs are ignored. This option applies only to the items that are fit by the three- and four-parameter model.

The following example shows different specifications:

```
proc irt resfunc=threep guessprior=(0.1,10) ceilprior=(0.8,10);
   var x1-x30;
   model x1-x10/guessprior=(0.2,10),
   x11-x20/resfunc=fourp ceilprior=(0.9,10) guessprior=(0.1,20);
run;
```

In the PROC IRT statement, you use the RESFUNC= option to set the response function to be THREEP for all the items, which are listed in the VAR statement. You also use the GUESSPRIOR= and the CEILPRIOR= options to set the prior information to (0.1,10) and (0.8,10) for the guessing and ceiling parameters, respectively. In the MODEL statement, you specify guessing prior information to (0.2, 10) for items x1 to x10. You also change the response function to FOURP for items x11 to x20 and also modify the prior information. However, items x21 to x30 are still fit by the THREEP model and the guessing prior information remains (0.1,10), as specified in the PROC IRT statement.

RESFUNC= [*response-model-types* **]**

specifies the response function or model. For available keywords, see the RESFUNC= option in the PROC IRT statement. For technical details about these response models, see the section "Response Models" on page 5452.

For variables that are also listed in the VAR statement, the model that is specified in the MODEL statement overwrites the default model or the model that is specified by using the RESFUNC= option in the PROC IRT statement.

Except for the generalized partial credit (GPC) model and Rasch (RASCH) model, you can specify mixed response models for different items as follows:

```
proc irt;
   model x1-x10/resfunc=twop, x11-x20/resfunc=graded, x21-x30/resfunc=threep;
run;
```

The generalized partial credit model or Rasch model requires that all items be fitted by the same model.

SLOPEPRIOR=((< *r1* >,< *r2* >) < ,(< *r1* >,< *r2* >)... >)

SLOPEPRIOR=(< *r1* >,< *r2* >)

specifies the prior information for the slope parameters. Each pair includes the mean (*r1*) and variance (*r2*) for the normal prior distribution. This option is similar to the SLOPEPRIOR= option in the PROC IRT statement. For the meaning and usage of the option, see the SLOPEPRIOR= option in the PROC IRT statement. You can specify multiple pairs of *r1* and *r2* in the SLOPEPRIOR= option in the MODEL statement, whereas you can specify only one pair of *r1* and *r2* in the PROC IRT statement.

The SLOPEPRIOR= option in the MODEL statement overwrites the default prior values and those that are specified in the PROC IRT statement. Both values in each pair are optional; if you do not specify them in this option, the values that are specified in the PROC IRT statement (or their default values) are used. If a single pair of values is specified, it applies to all the items specified in the *variable-list*. If the number of pairs specified is less than the number of items specified in the *variable-list*, the default settings are applied to the rest of the items. If more pairs are specified than the number of items in the *variable-list*, extra pairs are ignored. This option applies only to the items that are fit by the three- and four-parameter model.

The following example uses the SLOPEPRIOR= option in the MODEL statement to specify different prior information about the slope parameter for items x1, x2 and x3. Because the variance of the prior distribution for item x3 is large, the influence of the prior information for item x3 is very small.

```
proc irt;
   model x1-x3/resfunc=threep slopeprior=((0.5,1),(1,1),(1,16));
run;
```

VAR Statement

VAR *variables* **;**

The VAR statement lists the analysis variables or items in the model. If you do a multiple-group analysis, the same set of analysis variables is assumed for all groups. By default, all variables that you specify in the VAR statement are fit by the graded response model, which assumes that the analysis variables are ordinal and have 2 to 19 levels.

You can overwrite the default response model for the analysis variables in the VAR statement by using the RESFUNC= option in the PROC IRT statement. If you want to analyze the data by using a mixed type of response model, you can use the MODEL statement.

VARIANCE Statement

VARIANCE *assignment* < **,** *assignment . . .* > **;**

where *assignment* represents

var-list < **=***parameter-spec* >

The VARIANCE statement specifies the factor variance parameters in connection with the FACTOR statement. Notice that the VARIANCE statement is different from the VAR statement, which specifies variables for analysis. You can list factors only in the *var-list* of the VARIANCE statement.

In each *assignment* of the VARIANCE statement, you include the *var-list* whose variances you want to specify. Optionally, you can provide a list of parameter specifications (*parameter-spec*) after an equal sign for each *var-list*.

You can specify the following five types of the parameters for the variances of the latent factor in the VARIANCE statement:

- an unnamed free parameter
- an initial value
- a fixed value
- a free parameter with a name provided
- a free parameter with a name and initial value provided

Consider a confirmatory model that has the latent factors F1, F2, F3, F4, and F5.

The following VARIANCE statement illustrates the five types of parameter specifications in five *assignments*:

```
variance
   F1 ,
   F2 = (.5),
   F3 = 1.0,
   F4 = fvar,
   F5 = fvar(0.7);
```

In this statement, the variance of F1 is specified as an unnamed free parameter. The variance of F2 is an unnamed free parameter that has an initial value of 0.5. The variance of F3 is a fixed value of 1.0. This value stays the same during the estimation. The variance of F4 is a free parameter named fvar1. The variance of F5 is a free parameter named fvar2 that has an initial value of 0.7.

Mixed Parameter Lists

You can specify different types of parameters for the list of variances. For example, the following statement uses a list of parameters that have mixed types:

```
variance
   F1-F6 = vp1 vp2(2.0) vp3  4. (.3) vp6(.4);
```

This is equivalent to the following statement:

```
variance
   F1 = vp1
   F2 = vp2(2.0),
   F3 = vp3,
   F4 = 4. ,
   F5 = (.3),
   F6 = vp6(.4);
```

As you can see, an initial value that follows a parameter name is associated with the free parameter. For example, in the original mixed list specification, the specification `(2.0)` after vp2 is interpreted as the initial value of the parameter vp2, but not as the initial estimate of the variance of F3.

However, if you indeed want to specify that vp2 is a free parameter *without* an initial value and 2.0 is an initial estimate of the variance of F3 (while keeping all other things the same), you can use a null initial value specification for the parameter vp2, as shown in the following statement:

```
variance
   F1-F6 = vp1 vp2() (2.0)  4. (.3) vp6(.4);
```

This way, 2.0 becomes the initial estimate of the variance of F3. Because a, parameter list that contains mixed parameter types might be confusing, you can break down the specifications into separate *assignments* to remove ambiguities. For example, you can use the following equivalent statement:

```
variance
   F1 = vp1
   F2 = vp2,
   F3 = (2.),
   F4 = 4. ,
   F5 = (.3),
   F6 = vp6(.4);
```

Shorter and Longer Parameter Lists

If you provide fewer parameters than the number of variances in the *var-list*, all the remaining parameters are treated as unnamed free parameters. For example, the following statement assigns a fixed value of 1.0 to the variance of F1 while treating the other three variances as unnamed free parameters:

```
variance
   F1-F4 = 1.0;
```

This statement is equivalent to the following statement:

```
variance
   F1 = 1.0, F2-F4;
```

If you intend to fill up all values with the last parameter specification in the list, you can use the continuation syntax `[...]`, `[..]`, or `[.]`, as shown in the following example:

```
variance
   F1-F100 = 1.0 psi [...];
```

This means that the variance of F1 is fixed at 1.0 and that the variances of F1–F100 are all free parameters named psi. All variances except that for F1 are thus constrained to be equal by having the same parameter name.

However, you must be careful not to provide too many parameters. For example, the following statement results in an error:

```
variance
   F1-F6 = 1.0 psi2-psi6 extra;
```

The parameters after psi6 are excessive.

Default Variance Parameters

In the IRT model, by default, the factor variances are fixed at ones. You can use the VARIANCE statement to override these default variance parameters in situations where you want to specify parameter constraints, provide initial or fixed values, or make parameter references.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement specifies the weight variable for the observations. The WEIGHT and FREQ statements have a similar effect, except that the WEIGHT statement does not alter the number of observations. An observation is used in the analysis only if the WEIGHT variable is greater than 0 and is not missing.

Details: IRT Procedure

Notation for the Item Response Theory Model

This section introduces the mathematical notation that is used throughout the chapter to describe the item response theory (IRT) model. For a description of the fitting algorithms and the mathematical-statistical details, see the section "Maximizing the Marginal Likelihood" on page 5457.

Models for Ordinal Response Items

A d-dimensional graded response IRT model that has K ordinal responses can be expressed by the equations

$$y_{ij} = \boldsymbol{\lambda}_j \boldsymbol{\eta}_i + \epsilon_{ij}$$

$$p_{ijk} = \Pr(u_{ij} = k) = \Pr(\alpha_{(j,k-1)} < y_{ij} < \alpha_{(j,k)}), \quad k = 1, \ldots, K$$

where u_{ij} is the observed ordinal response from subject i for item j, y_{ij} is a continuous latent response that underlies u_{ij}, $\boldsymbol{\alpha}_j = (\alpha_{(j,0)} = -\infty, \alpha_{(j,1)}, \ldots, \alpha_{(j,K-1)}, \alpha_{(j,K)} = \infty)$ is a vector of threshold parameters for item j, $\boldsymbol{\lambda}_j$ is a vector of slope (discrimination) parameters for item j, $\boldsymbol{\eta}_i = (\eta_{i1}, \ldots, \eta_{id})$ is a vector of latent factors for subject i, $\boldsymbol{\eta}_i \sim N_d(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and $\boldsymbol{\epsilon}_i = (\epsilon_{i1}, \ldots, \epsilon_{iJ})$ is a vector of unique factors for subject i. All the unique factors in $\boldsymbol{\epsilon}_i$ are independent from one another, suggesting that y_{ij}, $j = 1, \ldots, J$, are independent conditional on the latent factor $\boldsymbol{\eta}_i$. This is the so-called local independence assumption. Finally, $\boldsymbol{\eta}_i$ and $\boldsymbol{\epsilon}_i$ are also independent.

Based on the preceding model specification,

$$p_{ijk} = \int_{\alpha_{(j,k-1)}}^{\alpha_{(j,k)}} p(y; \boldsymbol{\lambda}_j \boldsymbol{\eta}_i, 1)\, dy = \int_{\alpha_{(j,k-1)}-\boldsymbol{\lambda}_j \boldsymbol{\eta}_i}^{\alpha_{(j,k)}-\boldsymbol{\lambda}_j \boldsymbol{\eta}_i} p(y; 0, 1)\, dy$$

where p is determined by the link function. It is the density function of the standard normal distribution if the probit link is used, or the density function of the logistic distribution if the logistic link is used.

Let $\boldsymbol{\Lambda} = (\boldsymbol{\lambda}_1^T, \ldots, \boldsymbol{\lambda}_J^T)$ denote the slope matrix. To identify the model in exploratory analysis, the upper triangular elements of $\boldsymbol{\Lambda}$ are fixed as zero, the factor mean $\boldsymbol{\mu}$ is fixed as a zero vector, and the factor variance covariance matrix $\boldsymbol{\Sigma}$ is fixed as an identity matrix. For confirmatory analysis, it is assumed that the identification problem is solved by user-specified constraints.

The model that is specified in the preceding equation uses the latent response formulation. PROC IRT uses this parameterization for computational convenience. When there is only one latent factor, a mathematically equivalent parameterization for the model is

$$p_{ijk} = \int_{-a_j(\eta_i - b_{j,k-1})}^{-a_j(\eta_i - b_{j,k})} p(y; 0, 1)\, dy$$

where a_j is called the slope (discrimination) parameter and $b_{j,k}, k = 1, \ldots, K$, are called the threshold parameters. The threshold parameters under these two parameterizations can be translated as $b_{j,k} = \frac{\alpha_{j,k}}{\lambda_j}$, where $k = 1, \ldots, K$ and $\gamma_{j,k} = -\alpha_{j,k}$ is often called the intercept parameter.

The generalized partial credit (GPC) model is another popular IRT model for ordinal items besides the graded response model. Introduced by Muraki (1992), the GPC model is an extension of the partial credit (PC)

model proposed by Masters (1982). In the PC model, the slope (discrimination) parameter is fixed as 1 for all the items. The GPC model releases this assumption by introducing the slope parameter for each item. The GPC model can be formulated as

$$p_{ijk} = \frac{\exp(\sum_{h=1}^{k} a_j(\eta_i - b_{j,h}))}{\sum_{k=1}^{K} \exp(\sum_{h=1}^{k} a_j(\eta_i - b_{j,h}))}$$

In this formulation, a_j is called the slope (discrimination) parameter and $b_{j,h}$ is called the step parameter. For identification purposes, the first category of each item is treated as the reference level—that is, $b_{j,1}$ is fixed to zero for each item j. These fixed parameter values are omitted from the output.

The GPC model applies only to unidimensional analysis.

Models for Binary Response Items

Popular models for binary response items include one-, two-, three-, and four-parameter models and the Rasch model. When the responses are binary, the graded response model introduced in the section "Models for Ordinal Response Items" on page 5452 reduces to the two-parameter model, which can be expressed as

$$y_{ij} = a_j(\boldsymbol{\eta}_i - b_j) + \epsilon_{ij}$$

$$p_{ij} = \Pr(u_{ij} = 1) = \Pr(y_{ij} > 0)$$

where b_j is often called the item difficulty parameter.

The two-parameter model reduces to a one-parameter model when slope parameters for all the items are constrained to be equal. When the logistic link is used, the one- and two-parameter models are often abbreviated as 1PL and 2PL. When all the slope parameters are set to 1 and the factor variance is set to a free parameter, the Rasch model is obtained.

You can obtain three- and four-parameter models by introducing the guessing and ceiling parameters. Let g_j and c_j denote the item-specific guessing and ceiling parameters, respectively. Then the four-parameter model can be expressed as

$$p_{ij} = \Pr(u_{ij} = 1) = g_j + (c_j - g_j)\Pr(y_{ij} > 0)$$

This model reduces to the three-parameter model when $c_j = 1$.

Model for Nominal Response Items

For nominal response items, you can apply the nominal response model to do item analysis. The nominal response model was introduced by Bock (1972) and can be formulated as

$$p_{ijk} = \frac{\exp(a_{j,k}\eta_i + b_{j,k})}{\sum_{h=1}^{K} \exp(a_{j,h}\eta_i + b_{j,h})}$$

In this formulation, $a_{j,k}$ is the slope parameter and $b_{j,k}$ is the intercept parameter. For identification purposes, the first category of each item is treated as the reference level—that is, both $a_{j,1}$ and $b_{j,1}$ are fixed to zero for each item j. These fixed parameter values are omitted from the output.

This nominal response model applies only to unidimensional analysis.

Assumptions

The primary statistical assumptions that underlie the analyses that PROC IRT performs are as follows:

- The number of latent factors is known.
- Latent factors are normally distributed.
- Conditional on latent factors, observed responses (items) are independent. This is the so-called local independence assumption.

These assumptions are necessary if you want to make statistical inferences. In exploratory analysis, these assumptions do not apply.

PROC IRT Contrasted with Other SAS Procedures

IRT models are often referred to as latent trait models, especially in the field of sociology. The term latent trait is used to emphasize that observed discrete responses are manifestations of hypothesized traits, constructs, or attributes that cannot be directly observed. For that reason, IRT models belong to the more general modeling framework called latent variable models. Other models that belong to the latent variable model framework include factor analysis models, finite mixture models, and mixed-effects models. The relationships between these different latent variable models can be described as shown in Table 72.2.

Table 72.2 Latent Variable Models

		Latent Variable	
		Continuous	**Discrete**
Observed Variable	**Continuous**	Factor analysis	Finite mixture model
	Discrete	Item response theory	Latent class analysis

This table suggests that latent variable models can be classified into four groups, based on the measurement scale of observed and latent variables. These different latent variable models can be fit by different SAS procedures: PROC FACTOR for factor analysis models, PROC FMM for finite mixture models, and PROC IRT for item response theory models. IRT models are more closely related to factor analysis models. They can be considered a version of factor analysis models of discrete rather than continuous responses.

Response Models

PROC IRT supports several response models for binary and ordinal responses, and it allows different items to have different response models. Details about these response models and their relationships follow:

- One-parameter model: This model assumes that items are binary. The distinctive feature of the one-parameter model, compared with the two-parameter model, is that the slopes (or the discrimination parameters) of the items are the same in the model. Statistically, the one-parameter model is equivalent to the Rasch model. They give the same model fit for the same data set.
- Two-parameter model: This model assumes that items are binary. The slope (or the discrimination) and the difficulty (or the intercept) of the items are free parameters in the model. If all slopes of the two-parameter model are constrained to be same, it reduces to the one-parameter model.
- Three-parameter model: This model assumes that items are binary. The slope (or the discrimination), the difficulty (or the intercept), and the guessing parameters of the items are free parameters in the model. If all the guessing parameters are fixed to 0, the three-parameter model reduces to the two-parameter model.
- Four-parameter model: This model assumes that items are binary. The slope (or the discrimination), the difficulty (or the intercept), the guessing, and the ceiling parameters of the items are free parameters in the model. If all the guessing parameters are fixed to 0 and all the ceiling parameters are fixed to 1, the four-parameter model reduces to the two-parameter model.
- Rasch model: This model assumes that items are binary. The distinctive feature of the Rasch model, compared with the two-parameter model, is that the slope parameters (or the discrimination) of the items are all fixed to 1 (and with free factor variance) in the model. Statistically, the Rasch model is equivalent to the one-parameter model. They give the same model fit for the same data set.
- Graded response model: This model assumes that items are ordinal with at most 19 levels. The slope (or discrimination) and the threshold parameters of the items are free parameters in the model.
- Nominal response model: This model assumes that items are nominal with at most 19 categories. The slope and the intercept parameters of the items are free parameters in the model.
- Generalized partial credit model: This model assumes that items are ordinal-categorical with at most 19 levels. The slope (or discrimination) and the step parameters of the items are free parameters in the model.

You can specify the response function or model for all the variables that are listed in the VAR statement by using the RESFUNC= option in the PROC IRT statement. To specify different response functions or models for different set of variables, you can use the MODEL statement.

Marginal Likelihood

Based on the model that is specified in the section "Notation for the Item Response Theory Model" on page 5452, the marginal likelihood is

$$L(\boldsymbol{\theta}|U) = \prod_{i=1}^{N} \int \prod_{j=1}^{J} \prod_{k=1}^{K} (P_{ijk})^{v_{ijk}} \phi(\boldsymbol{\eta}; \boldsymbol{\mu}, \boldsymbol{\Sigma})\, d\boldsymbol{\eta} = \prod_{i=1}^{N} \int f(u_i|\boldsymbol{\eta}) \phi(\boldsymbol{\eta}; \boldsymbol{\mu}, \boldsymbol{\Sigma})\, d\boldsymbol{\eta}$$

where $v_{ijk} = I(u_{ij} = k)$, $\phi(\boldsymbol{\eta})$ is the multivariate normal density function for the latent factor $\boldsymbol{\eta}$, and $\boldsymbol{\theta}$ is a set of all the model parameters. The corresponding log likelihood is

$$\log L(\boldsymbol{\theta}|U) = \sum_{i=1}^{N} \log \int \prod_{j=1}^{J} \prod_{k=1}^{K} (P_{ijk})^{v_{ijk}} \phi(\boldsymbol{\eta}; \boldsymbol{\mu}, \boldsymbol{\Sigma})\, d\boldsymbol{\eta}$$

Integrations in the preceding equation cannot be solved analytically and need to be approximated by using numerical integration,

$$\log \tilde{L}(\boldsymbol{\theta}|U) = \sum_{i=1}^{N} \log \left[\sum_{g=1}^{G^d} \left[\prod_{j=1}^{J} \prod_{k=1}^{K} (P_{ijk}(\mathbf{x}_g))^{v_{ijk}} \frac{\phi(\mathbf{x}_g; \boldsymbol{\mu}, \boldsymbol{\Sigma})}{\phi(\mathbf{x}_g; 0, I)} \right] w_g \right]$$

where d is the number of factors, G is the number of quadrature points per dimension, and $\mathbf{x}_g$ and w_g are the quadrature points and weights, respectively.

Approximating the Marginal Likelihood

As discussed in the section "Marginal Likelihood" on page 5456, integrations that are involved in the marginal likelihood for IRT model cannot be solved analytically and need to be approximated by using numerical integration, mostly Gauss-Hermite quadrature.

Gauss-Hermite (G-H) Quadrature

In general, the Gauss-Hermite (G-H) quadrature can be presented as

$$\int_{-\infty}^{\infty} g(x)\, dx = \int_{-\infty}^{\infty} f(x)\phi(x)\, dx \approx \sum_{g=1}^{G} f(x_g) w_g$$

where G is the number of quadrature points and x_g and w_g are the integration points and weights, respectively, which are uniquely determined by the integration domain and the weighting kernel $\phi(x)$. Traditional G-H quadrature often uses e^{-x^2} as the weighting kernel. In the field of statistics, the density of standard normal distribution is more widely used instead, because for estimating various statistical models, the Gaussian density is often a factor of the integrand. In the case in which the Gaussian density is not a factor of the integrand, the integral is transformed into the form by dividing and multiplying the original integrand by the standard normal density.

Adaptive Gauss-Hermite Quadrature

The G order G-H quadrature is exact if $f(x)$ is a $2K-1$ degree polynomial in x. However, as many researchers (Lesaffre and Spiessens 2001; Rabe-Hesketh, Skrondal, and Pickles 2002) point out, integrands $f(u_i|\boldsymbol{\eta})\phi(\boldsymbol{\eta};\boldsymbol{\mu},\boldsymbol{\Sigma})$ often have sharp peaks and cannot be well approximated by low-degree polynomials in $\boldsymbol{\eta}$. Furthermore, the peak might be far from zero or be located between adjacent quadrature points so that substantial contributions to the integral are lost.

Note that the integrands in the marginal likelihood are a product of the prior density of $\boldsymbol{\eta}$, $\phi(\boldsymbol{\eta};\boldsymbol{\mu},\boldsymbol{\Sigma})$ and the joint probability of responses given $\boldsymbol{\eta}$, $f(\mathbf{u}_i|\boldsymbol{\eta})$. After normalization with respect to $\boldsymbol{\eta}$, the integrand, $f(u_i|\boldsymbol{\eta})\phi(\boldsymbol{\eta};\boldsymbol{\mu},\boldsymbol{\Sigma})$, is just the posterior density of $\boldsymbol{\eta}$, given the observed responses $\mathbf{u}_i$. This posterior density is approximately normal when the number of items is large. Let $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ be the mean and covariance matrix, respectively, of the posterior density. Then the ratio $\frac{f(\mathbf{u}_i|\boldsymbol{\eta})\phi(\boldsymbol{\eta};\boldsymbol{\mu},\boldsymbol{\Sigma})}{\phi(\boldsymbol{\eta};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)}$ can be well approximated by a low-degree polynomial if the number of items is relatively large. This suggests that the integral should be transformed as

$$\int f(\mathbf{u}_i|\boldsymbol{\eta})\phi(\boldsymbol{\eta})\,d\boldsymbol{\eta} = \int \frac{f(\mathbf{u}_i|\boldsymbol{\eta})\phi(\boldsymbol{\eta};\boldsymbol{\mu},\boldsymbol{\Sigma})}{\phi(\boldsymbol{\eta};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)}\phi(\boldsymbol{\eta};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)\,d\boldsymbol{\eta}$$

The integration points and weights that correspond to $\phi(\boldsymbol{\eta};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)$ are

$$z_g = \boldsymbol{\Sigma}_i^{1/2}x_g + \boldsymbol{\mu}_i$$

$$v_g = |\boldsymbol{\Sigma}_i|^{1/2}w_g$$

The preceding transformations move and scale the quadrature points to the center of the integrands such that the integrand can be better approximated using many fewer quadrature points.

Maximizing the Marginal Likelihood

You can obtain parameter estimates by maximizing the marginal likelihood by using either the expectation maximization (EM) algorithm or a Newton-type algorithm. Both algorithms are available in PROC IRT.

The most widely used estimation method for IRT models is the Gauss-Hermite quadrature–based EM algorithm, proposed by Bock and Aitkin (1981). However, this method has several important shortcomings, the most serious of which is the lack of reliable convergence criteria. Without reliable convergence criteria, estimates can be seriously biased because of spurious convergence. In comparison, gradient-based convergence criteria is readily available for Newton-type algorithms. As a result, PROC IRT uses the quasi-Newton algorithm instead of EM as the default optimization method.

Newton-Type Algorithms

Newton-type algorithms maximize the marginal likelihood directly, based on the first and second derivatives. Two of the most widely used estimation algorithms are the Newton-Raphson and Fisher scoring algorithms, which rely on the gradient and Hessian of the log likelihood. However, for latent variable models that contain categorical responses, the Hessian matrix is often expensive to compute. As a result, several quasi-Newton algorithms that require only gradients have been proposed. In the field of IRT, Bock and Lieberman (1970) propose replacing the Hessian with the following information matrix:

$$I(\boldsymbol{\theta}) = E\left[\frac{\partial \log \tilde{L}(\theta|U)}{\partial\theta}\left(\frac{\partial \log \tilde{L}(\theta|U)}{\partial\theta}\right)^T\right] = \sum_{h=1}^{2^J}\left[\frac{\partial \log \tilde{L}_i}{\partial\theta}\left(\frac{\partial \log \tilde{L}_i}{\partial\theta}\right)^T\right]$$

To calculate the preceding expectation, you need to sum over not just the observed but all 2^J possible response patterns; this becomes computationally very expensive when the number of items is large. Fortunately, other quasi-Newton algorithms that do not have these computational difficulties have been proposed. The Broyden-Fletcher-Goldfarb-Shanno (BFGS) method is one of the most popular quasi-Newton algorithms that approximate the Hessian matrix with gradient.

For the objective function, $\log \tilde{L}(\theta)$, the first derivatives with respect to θ_j are

$$\frac{\partial \log \tilde{L}(\boldsymbol{\theta}|\mathbf{U})}{\partial \theta_j} = \sum_{i=1}^{N}\left[(\tilde{L}_i)^{-1}\frac{\partial \tilde{L}_i}{\partial \theta_j}\right] = \sum_{i=1}^{N}\left[(\tilde{L}_i)^{-1}\sum_{g=1}^{G^d}\left[\frac{\partial f_i(\mathbf{x}_g)}{\partial \theta_j}w_g^*\right]\right]$$

where

$$\tilde{L}_i = \sum_{g=1}^{G^d}\left[\prod_{j=1}^{J}\prod_{k=1}^{K}(P_{ijk}(\mathbf{x}_g))^{v_{ijk}}\frac{\phi(\mathbf{x}_g;\boldsymbol{\mu},\boldsymbol{\Sigma})}{d(\mathbf{x}_g;0,I)}\right]w_g = \sum_{g=1}^{G^d} f_i(x_g)w_g$$

$$f_i(x_g) = \prod_{j=1}^{J}\prod_{k=1}^{K}(P_{ijk}(\mathbf{x}_g))^{v_{ijk}}\frac{\phi(\mathbf{x}_g;\boldsymbol{\mu},\boldsymbol{\Sigma})}{d(\mathbf{x}_g;0,I)}$$

and

$$\frac{\partial f_i(\mathbf{x}_g)}{\partial \theta_{Ij}} = \frac{\partial[(P_{ijk}(\mathbf{x}_g))]}{\partial \theta_j}\frac{f_i(\mathbf{x}_g)}{(P_{ijk}(\mathbf{x}_g))}$$

$$\frac{\partial f_i(\mathbf{x}_g)}{\partial \theta_f} = \prod_{j=1}^{J}\prod_{k=1}^{K}(P_{ijk}(\mathbf{x}_g))^{v_{ijk}}\frac{\partial \phi(\mathbf{x}_g;\boldsymbol{\mu},\boldsymbol{\Sigma})}{\partial \theta_f}$$

where, in the preceding two equations, θ_{Ij} indicate parameters that are associated with item j and θ_f represents parameters that are related to latent factors.

Expectation-Maximization (EM) Algorithm

The expectation-maximization (EM) algorithm starts from the complete data log likelihood that can be expressed as follows:

$$\begin{aligned}\log L(\boldsymbol{\theta}|\mathbf{U},\boldsymbol{\eta}) &= \sum_{i=1}^{N}\left[\sum_{j=1}^{J}\sum_{k=1}^{K} v_{ijk}\log P_{ijk} + \log\phi(\boldsymbol{\eta}_i;\boldsymbol{\mu},\boldsymbol{\Sigma})\right]\\ &= \sum_{j=1}^{J}\sum_{i=1}^{N}\left[\sum_{k=1}^{K} v_{ijk}\log P_{ijk}\right] + \sum_{i=1}^{N}\log\phi(\boldsymbol{\eta}_i;\boldsymbol{\mu},\boldsymbol{\Sigma})\end{aligned}$$

The expectation (E) step calculates the expectation of the complete data log likelihood with respect to the conditional distribution of $\boldsymbol{\eta}_i$, $f(\boldsymbol{\eta}_i|\mathbf{u}_i,\boldsymbol{\theta}^{(t)})$ as follows:

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \sum_{j=1}^{J}\sum_{i=1}^{N}\left[\sum_{k=1}^{K} v_{ijk}E\left[\log P_{ijk}|\mathbf{u}_i,\boldsymbol{\theta}^{(t)}\right]\right] + \sum_{i=1}^{N}E\left[\log\phi(\boldsymbol{\eta}_i;\boldsymbol{\mu},\boldsymbol{\Sigma})|\mathbf{u}_i,\boldsymbol{\theta}^{(t)}\right]$$

The conditional distribution $f(\boldsymbol{\eta}|u_i, \theta^{(t)})$ is

$$f(\boldsymbol{\eta}|\mathbf{u}_i, \boldsymbol{\theta}^{(t)}) = \frac{f(\mathbf{u}_i|\boldsymbol{\eta}, \boldsymbol{\theta}^{(t)})\phi(\boldsymbol{\eta}; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})}{\int f(\mathbf{u}_i|\boldsymbol{\eta}, \boldsymbol{\theta}^{(t)})\phi(\boldsymbol{\eta}; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})\, d\boldsymbol{\eta}} = \frac{f(\mathbf{u}_i|\boldsymbol{\eta}, \boldsymbol{\theta}^{(t)})\phi(\boldsymbol{\eta}; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})}{f(\mathbf{u}_i)}$$

These conditional expectations that are involved in the Q function can be expressed as follows:

$$E[\log P_{ijk}|\mathbf{u}_i, \boldsymbol{\theta}^{(t)}] = \int \log P_{ijk} f(\boldsymbol{\eta}|\mathbf{u}_i, \boldsymbol{\theta}^{(t)})\, d\boldsymbol{\eta}$$

$$E[\log \phi(\boldsymbol{\eta}; \boldsymbol{\mu}, \boldsymbol{\Sigma})|\mathbf{u}_i, \boldsymbol{\theta}^{(t)}] = \int \log \phi(\boldsymbol{\eta}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) f(\boldsymbol{\eta}|\mathbf{u}_i, \boldsymbol{\theta}^{(t)})\, d\boldsymbol{\eta}$$

Then

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \sum_{j=1}^{J} \int \left[\log P_{ijk} r_{jk}(\boldsymbol{\theta}^{(t)})\right] \phi(\boldsymbol{\eta}; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})\, d\boldsymbol{\eta} + \int \log \phi(\boldsymbol{\eta}|\boldsymbol{\theta}) N(\boldsymbol{\theta}^{(t)}) \phi(\boldsymbol{\eta}; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})\, d\boldsymbol{\eta}$$

where

$$r_{jk}(\boldsymbol{\theta}^{(t)}) = \sum_{i=1}^{N} \sum_{k=1}^{K} v_{ijk} \frac{f(\mathbf{u}_i|\eta, \boldsymbol{\theta}^{(t)})}{f(\mathbf{u}_i)}$$

and

$$N(\boldsymbol{\theta}^{(t)}) = \sum_{i=1}^{N} \frac{f(\mathbf{u}_i|\boldsymbol{\eta}, \boldsymbol{\theta}^{(t)})}{f(\mathbf{u}_i)}$$

Integrations in the preceding equations can be approximated as follows by using G-H quadrature:

$$\tilde{Q}(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)}) = \sum_{g=1}^{G} \left[\log P_{ijk}(\mathbf{x}_g) r_{jk}(\mathbf{x}_g, \boldsymbol{\theta}^{(t)}) + \log \phi(\mathbf{x}_g|\boldsymbol{\theta}) N(\mathbf{x}_g, \boldsymbol{\theta}^{(t)})\right] \frac{\phi(\mathbf{x}_g; \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)})}{d(\mathbf{x}_g; 0, I)} w_g$$

In the maximization (M) step of the EM algorithm, parameters are updated by maximizing $\tilde{Q}(\theta|\theta^{(t)})$. To summarize, the EM algorithm consists of the following two steps:

E step: Approximate $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$ by using numerical integration.

M step: Update parameter estimates by maximizing $\tilde{Q}(\boldsymbol{\theta}|\boldsymbol{\theta}^{(t)})$ with the one-step Newton-Raphson algorithm.

Prior Distributions for Parameters

Technical issues often arise when the marginal likelihood is maximized for three- and four-parameter models. For example, the optimization algorithm might not converge, or the estimates for one or both of the guessing or ceiling parameters could reach their boundaries of 0 or 1, respectively, resulting in the standard errors for the guessing and ceiling parameters not being computable. To solve these technical issues, prior distributions for the guessing, ceiling, and slope parameters are often incorporated into the marginal likelihood to produce the joint posterior distribution,

$$P(\boldsymbol{\theta}|U) = L(\boldsymbol{\theta}|U) \sum_{j=1}^{J} \left[f(g_j) f(c_j) f(\lambda_j) \right]$$

where $L(\boldsymbol{\theta}|U)$ is the likelihood function and $f(g_j)$, $f(c_j)$, and $f(\lambda_j)$ are the prior distributions of the guessing, ceiling, and slope parameters, respectively. Parameter estimates are then obtained by maximizing the joint posterior distribution. In Bayesian statistics, these parameter estimates are called maximum a posteriori probability (MAP) estimates. For more information about prior distributions and Bayesian analysis of IRT models, see Baker and Kim (2004).

The following subsections describe the prior distributions that PROC IRT incorporates for three- and four-parameter models.

Prior Distribution for the Guessing and Ceiling Parameters

The following beta prior distribution is used for the guessing and ceiling parameters:

$$f(\mathrm{x}; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} \mathrm{x}^{\alpha-1} (1 - \mathrm{x})^{\beta-1}$$

where x is either the guessing or the ceiling parameter and $B(\alpha, \beta)$ is the beta function. The mean of the beta distribution is $\frac{\alpha}{\alpha+\beta}$. Because the α and β parameters are not practically meaningful by themselves, PROC IRT does not use these two parameters to specify this prior distribution. Instead, a mean and a weight are used as the parameters for the distribution. The mean parameter, $m = \frac{\alpha}{\alpha+\beta}$, corresponds to the mean of the beta distribution, and it represents the probability of getting a correct response. The weight parameter, $w = \alpha + \beta$, represents the confidence of the prior information. The more confident you are that the parameter value is near the mean parameter value, the larger the weight parameter you specify.

By default, the mean parameter is 0.2 for the guessing prior and 0.9 for the ceiling prior. The default weight is 20 for both the guessing and the ceiling priors. You can change the default settings by using the GUESSPRIOR= and CEILPRIOR= options in either the PROC IRT statement or the MODEL statement.

Prior Distribution for the Slope Parameter

The normal prior distribution is used for the slope parameters for three- and four-parameter models:

$$f(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\sigma^2\pi}} \exp \frac{-(x-\mu)^2}{2\sigma^2}$$

Although the slope parameters exist in all IRT models, no prior distribution for the slopes would be incorporated into models other than the three- and four-parameter models. In PROC IRT, you uses the μ and σ^2 parameters directly to specify this normal prior for the slope parameter. The mean parameter, μ, represents prior information about the value of the slope parameter. The inverse of the variance parameter, $\frac{1}{\sigma^2}$, represents the precision of the prior information. A smaller variance parameter means that the prior information is more precise and has a bigger impact on the posterior distribution.

By default, the mean and variance parameters are 0 and 1. To change the default, you can use the SLOPEPRIOR= option in either the PROC IRT statement or the MODEL statement.

Factor Score Estimation

PROC IRT provides three methods of estimating factor scores: maximum likelihood (ML), maximum a posteriori (MAP), and expected a posteriori (EAP). You can specify them by using the SCOREMETHOD= option in the PROC IRT statement.

You can obtain the ML factor score by maximizing the likelihood for each observation with respect to the latent factor. You can also compute the MAP or EAP factor score by maximizing or by taking the expectation of the posterior distribution of latent factors for each observation. The likelihood and posterior distribution for each observation, $u_i = (u_{i1}, \ldots, u_{iJ})$, can be expressed, respectively, as

$$l(\boldsymbol{\eta}|u_i, \hat{\boldsymbol{\theta}}) = \prod_{j=1}^{J} \prod_{k=1}^{K} (P_{ijk})^{v_{ijk}}$$

and

$$p(\boldsymbol{\eta}|u_i, \hat{\boldsymbol{\theta}}) \propto \prod_{j=1}^{J} \prod_{k=1}^{K} (P_{ijk})^{v_{ijk}} \phi(\boldsymbol{\eta}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Factor scores are restricted to the range from –99 to 99. For unidimensional models, the ML factor score is not available for subjects whose response to all the items is either the lowest or the highest level. For example, suppose there are five binary items in the model. For subjects whose response is 1 or 0 to all five items, the ML factor score cannot be estimated. For subjects whose response to all items is the lowest level, the ML factor score is set to –99, and for subjects whose response to all items is the highest level, the ML factor score is set to 99.

Model and Item Fit

The IRT procedure includes five model fit statistics: log likelihood, Akaike's information criterion (AIC), Bayesian information criterion (BIC), likelihood ratio chi-square G^2, and Pearson's chi-square.

The following two equations compute the likelihood ratio chi-square G^2 and Pearson's chi-square,

$$G^2 = 2\left[\sum_{l=1}^{L} r_l \log \frac{r_l}{NP_l}\right]$$

$$\chi^2 = \sum_{l=1}^{L} \frac{(r_l - NP_l)^2}{NP_l}$$

where N is the number of subjects, L is number of possible response patterns, P_l is the estimated probability of observing response pattern l, and r_l is the number of subjects who have response pattern l. If the model is true, these two statistics asymptotically follow central chi-square distribution with degrees of freedom $L - m - 1$, where m is the number of free parameters in the model. When L (the number of possible response patterns) is much greater than N, the frequency table is sparse. This invalidates the use of chi-square distribution as the asymptotic distribution for these two statistics, and as a result the likelihood ratio chi-square and Pearson's chi-square statistics should not be used to evaluate overall model fit.

For item fit, PROC IRT computes the likelihood ratio G^2 and Pearson's chi-square. Pearson's chi-square statistic, proposed by Yen (1981), has the form

$$Q_{1j} = \sum_{k=1}^{10} N_k \frac{(O_{jk} - E_{jk})^2}{E_{jk}(1 - E_{jk})}$$

The likelihood ratio G^2, proposed by McKinley and Mills (1985), uses the following equation:

$$G^2 = 2\sum_{k=1}^{10} N_k \left[O_{jk} \log \frac{O_{jk}}{E_{ik}} + (1 - O_{jk}) \log \frac{1 - O_{jk}}{1 - E_{ik}}\right]$$

These two statistics approximately follow a central chi-square distribution with $10 - m_j$ degrees of freedom, where m_j is the number of free parameters for item j.

To calculate these two statistics, first order all the subjects according to their estimated factor scores, and then partition them into 10 intervals such that the number of subjects in each interval is approximately equal. O_{jk} and E_{jk} are the observed proportion and expected proportion, respectively, of subjects in interval k who have a correct response on item j. The expected proportions E_{jk} are computed as the mean predicted probability of a correct response in interval k.

Item and Test information

Let $P_{jk}(\theta)$ be the probability of endorsing category k for item j for a subject whose ability score is θ. Then the item information function can be defined as

$$I_j(\theta) = \sum_{k=1}^{K} I_k(\theta) P_{jk}(\theta)$$

where

$$I_k(\theta) = -\frac{\partial^2}{\partial\theta^2} \log P_{jk}(\theta)$$

The test information function is the sum of the information functions of the items in the test. The information function of a test that has J items is

$$I(\theta) = \sum_{j=1}^{J} I_j(\theta)$$

Missing Values

PROC IRT handles missing values differently for different tasks. During model estimation and subject scoring, observations with missing values are still used, in the sense that the nonmissing values can still contribute to the estimation or scoring. When calculating the polychoric correlation for a pair of variables, PROC IRT does pairwise deletion, in which observations that have valid data about the corresponding pair of variables are used. When calculating the item statistics table, PROC IRT uses listwise deletion, in which observations with missing values for any variables in the analysis are omitted from the computations.

Output Data Sets

OUTMODEL= Data Set

The OUTMODEL= data set contains the model specification, the computed parameter estimates, and the standard error estimates. This data set is intended to be reused as an INMODEL= data set in a subsequent analysis by PROC IRT.

The OUTMODEL= data set contains the following variables:

- BY variables, if any
- _GPNUM_ variable for group numbers, if used
- _TYPE_, a character variable that takes various values that indicate the type of model specification
- _SUBTYP_, a character variable that takes various values that indicate the type of model parameters

- _NAME_, a character variable that indicates the model type, parameter name, or variable name
- _NUM_, a numeric variable that takes various values that indicate model specifications such as the number of items, number of factors, number of groups, and number of levels for each item
- _VAR1_, a character variable that is the name or number of the first variable in the specification
- _VAR2_, a character variable that is the name or number of the second variable in the specification
- _ESTIM_, a numeric variable that is the final estimate of the parameter
- _STDERR_, a numeric variable that is the standard error estimate of the parameter

Each observation (record) of the OUTMODEL= data set contains a piece of information about the model specification. Depending on the type of specification that the value of the _TYPE_ variable indicates, the meanings of the _SUBTYP_, _NAME_, _NUM_ _VAR1_, and _VAR2_ variables differ. Table 72.3 summarizes the meanings of these variables for each value of the _TYPE_ variable.

Table 72.3 Meaning of Variables in the OUTMODEL=Data Set

TYPE	Description	_SUBTYP_	_NAME_	_NUM_	_VAR1_	_VAR2_
MODEL	Model Info		NGROUP	Number of groups		
MODEL	Model Info		NITEM	Number of items		
MODEL	Model Info		NFACTOR	Number of factors		
MODEL	Model Info		LINK		Link function	
MODEL	Model Info		RORDER		RORDER option	
MODEL	Model Info		DESC		DESC option	
VAR	Variable		Name	Number of levels	Response function	
VAR	Variable	LEVINFO	Name	Level	Level value	
FACTOR	Factor		Name			
GPVAR	Group		Group var name			
PRIOR	Prior Info	CEILING	Variable			
PRIOR	Prior Info	GUESSING	Variable			
PRIOR	Prior Info	SLOPE	Variable			
PARM	Parameter	CEILING	Name	Number	Variable	
PARM	Parameter	GUESSING	Name	Number	Variable	
PARM	Parameter	INTERCEPT	Name	Number	Variable	Category
PARM	Parameter	SLOPE	Name	Number	Variable	Factor
PARM	Parameter	COV	Name	Number	1st variable	2nd variable
PARM	Parameter	MEAN	Name	Number	Variable	

For computational convenience, the intercept parameters rather than the threshold (or difficulty) parameters are saved in the OUTMODEL= data set. For multidimensional exploratory analysis, the OUTMODEL= data set includes the unrotated factor loading matrix. In the OUTMODEL= data set, fixed parameters do not have names but only parameter numbers. For free parameters, the OUTMODEL= data set includes the parameter name and the parameter number. When this data set is used for the INMODEL= option, PROC IRT sets an equality constraint for parameters that have the same name.

ODS Table Names

PROC IRT assigns a name to each table that it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 72.4. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 72.4 ODS Tables Produced by PROC IRT

ODS Table Name	Description	Option
ConvergenceStatus	Convergence status	Default output
Eigenvalues	Polychoric correlation–based eigenvalues	Default output
FactorCov	Factor covariance estimates	Default output; available only for multidimensional models
FactorCovInit	Initial factor covariance estimates	PINITIAL option; available only for multidimensional models
FactorMean	Factor mean estimates	Default output; available only for multidimensional models
FactorMeanInit	Initial factor mean estimates	PINITIAL option; available only for multidimensional models
FactorCovRot	Rotated factor covariance estimates	ROTATE= oblique rotation methods; available only for multidimensional exploratory models
FitStatistics	Model fit statistics	Default output
GroupInfo	Group information	Default output; available only for multiple group analysis
ItemFit	Item fit statistics	ITEMFIT option; available only for binary responses that have one latent factor
ItemInfo	Item information	Default output
ItemStat	Classical item statistics	ITEMSTAT option
IterHistory	Iteration history	Default output
ModelInfo	Model information	Default output
OptInfo	Optimization information	Default output
ParameterEstimates	Item parameter estimates. Slope parameters are not included in this table for multidimensional models.	Default output

Table 72.4 *continued*

ODS Table Name	Description	Option
ParameterEstimatesInit	Initial item parameter estimates. Slope parameters are not included in this table for multidimensional models.	PINITIAL option
PolyCorr	Polychoric correlation matrix	POLYCHORIC option
Slope	Slope parameter estimates	Default output; available only for multidimensional confirmatory models
SlopeInit	Initial slope parameter estimates	PINITIAL option; available only for multidimensional models
SlopeRot	Rotated slope parameter estimates	Default output; available only for multidimensional exploratory models

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You must also specify the PLOTS= option in the PROC IRT statement.

PROC IRT assigns a name to each graph that it creates using ODS. You can use these names to refer to the graphs when using ODS. The names are listed in Table 72.5.

Table 72.5 Graphs Produced by PROC IRT

ODS Graph Name	Plot Description	PLOTS= Option
ItemCharCurve	Item characteristic curves	PLOTS=ICC. The plot panels by default; specify PLOTS=UNPACK to produce an individual plot for each item.
ItemInfoCurve	Item information curves	PLOTS=IIC. The plot panels by default; specify PLOTS=UNPACK to produce an individual plot for each item.
PolyCorrHeatMap	Heat map for polychoric correlation matrix	PLOTS=POLYCHORIC | PLCORR
ScreePlot	Scree and variance-explained plots	PLOTS=SCREE
TestInfoCurve	Test information curve	PLOTS=TIC
VariancePlot	Plot of explained variance	PLOTS=SCREE(UNPACK)

Examples: IRT Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 72.1: Unidimensional IRT Models

(View the complete code for this example (irtex1.sas) in the example repository.)

This example shows you the features that PROC IRT provides for unidimensional analysis. The data set comes from the 1978 Quality of American Life Survey. The survey was administered to a sample of all US residents aged 18 years and older in 1978. In this survey, subjects were asked to rate their satisfaction with many different aspects of their lives. This example selects eight items. These items are designed to measure people's satisfaction in the following areas on a seven-point scale: community, neighborhood, dwelling unit, life in the United States, amount of education received, own health, job, and how spare time is spent. For illustration purposes, the first five items are dichotomized and the last three items are collapsed into three levels.

The following DATA step creates the data set IrtUni.

```
data IrtUni;
   input item1-item8 @@;
   datalines;
1 0 0 0 1 1 2 1 1 1 1 1 1 3 3 3 0 1 0 0 1 1 1 1 1 0 0 1 0 1 2 3 0 0 0
0 0 1 1 1 1 0 0 1 0 1 3 3 0 0 0 0 0 1 1 3 0 0 1 0 0 1 2 2 0 1 0 0 1 1
```

```
   ... more lines ...

3 3 0 1 0 0 1 2 2 1
;
```

Because all the items are designed to measure subjects' satisfaction in different aspects of their lives, it is reasonable to start with a unidimensional IRT model. The following statements fit such a model by using several user-specified options:

```
ods graphics on;
proc irt data=IrtUni link=probit pinitial itemstat polychoric
   itemfit plots=(icc polychoric);
   var item1-item8;
   model item1-item4/resfunc=twop, item5-item8/resfunc=graded;
run;
```

The ODS GRAPHICS ON statement invokes the ODS Graphics environment and displays the plots, such as the item characteristic curve plot. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

The first option is the LINK= option, which specifies that the link function be the probit link. Next, you request initial parameter estimates by using the PINITIAL option. Item fit statistics are displayed using the ITEMFIT option. In the PROC IRT statement, you can use the PLOTS option to request different plots. In this example, you request item characteristic curves by using the PLOTS=ICC option.

In this example, you use the MODEL statement to specify different response models for different items. The specifications in the MODEL statement suggest that the first four items, item1 to item4, are fit using the two-parameter model, whereas the last four items, item5 to item8, are fit using the graded response model.

Output 72.1.1 displays two tables. From the "Modeling Information" table, you can observe that the link function has changed from the default LOGIT link to the specified PROBIT link. The "Item Information" table shows that item1 to item5 each have two levels and item6 to item8 each have three levels. The last column shows the raw values of these different levels.

Output 72.1.1 Basic Information

The IRT Procedure

Modeling Information	
Data Set	WORK.IRTUNI
Link Function	Probit
Number of Items	8
Number of Factors	1
Number of Observations Read	500
Number of Observations Used	500
Estimation Method	Marginal Maximum Likelihood

Output 72.1.1 *continued*

Item Information			
Response Model	Item	Levels	Values
TwoP	item1	2	0 1
	item2	2	0 1
	item3	2	0 1
	item4	2	0 1
Graded	item5	2	0 1
	item6	3	1 2 3
	item7	3	1 2 3
	item8	3	1 2 3

Output 72.1.2 displays the classical item statistics table, which include the item means, item-total correlations, adjusted item-total correlations, and item means for *i* ordered groups of observations or individuals. You can produce this table by specifying the ITEMSTAT option in the PROC IRT statement.

Output 72.1.2 Classical Item Statistics

The IRT Procedure

Item Statistics							
		Item-Total Correlations		Means			
Item	Mean	Unadjusted	Adjusted	G1 (N=132)	G2 (N=139)	G3 (N=119)	G4 (N=110)
item1	0.42400	0.57595	0.43291	0.11364	0.26619	0.53782	0.87273
item2	0.34400	0.53837	0.39480	0.06818	0.19424	0.45378	0.74545
item3	0.38800	0.51335	0.36132	0.09091	0.30216	0.47899	0.75455
item4	0.41000	0.44559	0.28197	0.16667	0.30935	0.47899	0.75455
item5	0.63000	0.43591	0.27436	0.32576	0.64029	0.71429	0.89091
item6	1.82800	0.50955	0.24040	1.34848	1.66187	1.98319	2.44545
item7	2.04200	0.65163	0.41822	1.30303	1.97842	2.35294	2.67273
item8	2.18600	0.66119	0.43254	1.40909	2.15827	2.50420	2.80909
Total N=500, Cronbach Alpha=0.6482							

PROC IRT produces the "Eigenvalues of the Polychoric Correlation Matrix" table in Output 72.1.3 by default. You can use these eigenvalues to assess the dimension of latent factors. For this example, the fact that only the first eigenvalue is greater than 1 suggests that a one-factor model for the items is reasonable.

Output 72.1.3 Eigenvalues of Polychoric Correlations

Eigenvalues of the Polychoric Correlation Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	3.11870486	2.12497677	0.3898	0.3898
2	0.99372809	0.10025986	0.1242	0.5141
3	0.89346823	0.03116998	0.1117	0.6257
4	0.86229826	0.10670185	0.1078	0.7335
5	0.75559640	0.17795713	0.0944	0.8280
6	0.57763928	0.10080017	0.0722	0.9002
7	0.47683911	0.15511333	0.0596	0.9598
8	0.32172578		0.0402	1.0000

To get an overall idea of the correlations among all the items in the analysis, you can request the polychoric correlation matrix and the corresponding heat map. When you have a large number of items in the analysis, the heat map is especially useful to help you find patterns among these items. To produce the polychoric correlation matrix, specify the POLYCHORIC option in the PROC IRT statement. Specify PLOTS=POLYCHORIC to get the heat map for the polychoric correlation matrix. Output 72.1.4 includes the polychoric correlation table for this example, and Output 72.1.5 includes the heat map.

Output 72.1.4 Polychoric Correlation Matrix

Polychoric Correlation Matrix								
	item1	item2	item3	item4	item5	item6	item7	item8
item1	1.0000	0.5333	0.4663	0.4181	0.2626	0.2512	0.3003	0.3723
item2	0.5333	1.0000	0.5183	0.2531	0.1543	0.2545	0.3757	0.2910
item3	0.4663	0.5183	1.0000	0.2308	0.2467	0.1455	0.2561	0.3771
item4	0.4181	0.2531	0.2308	1.0000	0.1755	0.1607	0.3181	0.1825
item5	0.2626	0.1543	0.2467	0.1755	1.0000	0.1725	0.3156	0.2846
item6	0.2512	0.2545	0.1455	0.1607	0.1725	1.0000	0.1513	0.2404
item7	0.3003	0.3757	0.2561	0.3181	0.3156	0.1513	1.0000	0.4856
item8	0.3723	0.2910	0.3771	0.1825	0.2846	0.2404	0.4856	1.0000

Output 72.1.5 Polychoric Correlation Heat Map

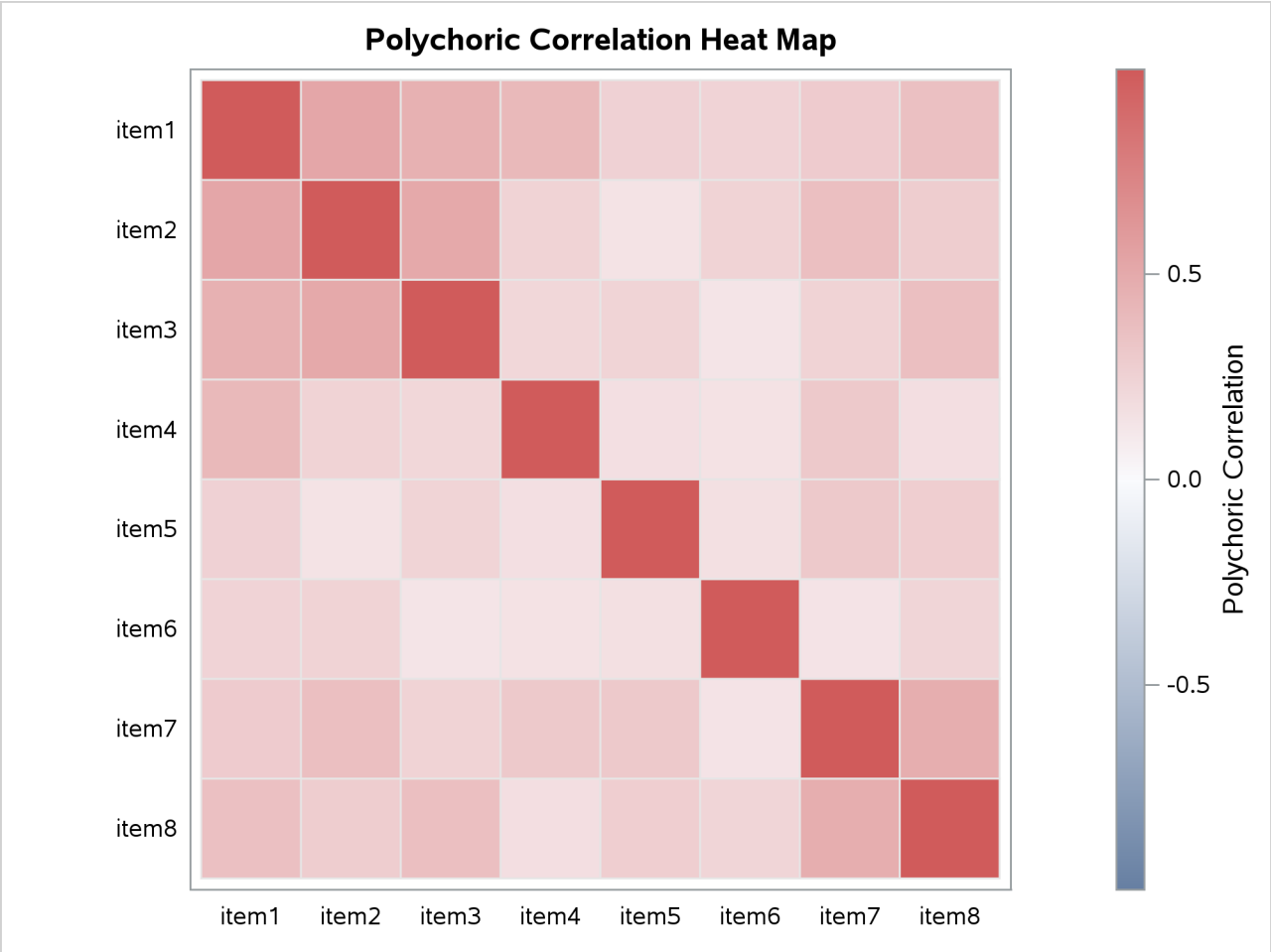

The PINITIAL option in the PROC IRT statement displays the "Initial Item Parameter Estimates" table, shown in Output 72.1.6.

Output 72.1.6 Initial Parameter Estimates

The IRT Procedure

Initial Item Parameter Estimates			
Response Model	Item	Parameter	Estimate
TwoP	item1	Difficulty	0.26723
		Slope	1.02931
	item2	Difficulty	0.60728
		Slope	0.88149
	item3	Difficulty	0.46029
		Slope	0.78642
	item4	Difficulty	0.50623
		Slope	0.50318
Graded	item5	Threshold	-0.83011
		Slope	0.43614
	item6	Threshold 1	-0.59130
		Threshold 2	1.99390
		Slope	0.36327
	item7	Threshold 1	-0.86053
		Threshold 2	0.65752
		Slope	0.69307
	item8	Threshold 1	-1.13423
		Threshold 2	0.25868
		Slope	0.71871

Output 72.1.7 includes tables that are related to the optimization. The "Optimization Information" table shows that the log likelihood is approximated by using seven adaptive Gauss-Hermite quadrature points and then maximized by using the quasi-Newton algorithm. The number of free parameters in this example is 19. The "Iteration History" table shows the number of function evaluations, the objective function (–log likelihood divided by number of subjects) values, the objective function change, and the maximum gradient for each iteration. This information is very useful in monitoring the optimization status. Output 72.1.7 shows the convergence status at the bottom. The optimization converges according to the GCONV=0.00000001 criterion.

Output 72.1.7 Optimization Information

The IRT Procedure

Optimization Information	
Optimization Technique	Quasi-Newton
Likelihood Approximation	Adaptive Gauss-Hermite Quadrature
Number of Quadrature Points	21
Number of Free Parameters	19

Output 72.1.7 *continued*

Iteration History					
Cycles	Iteration	Evaluations	Objective Function	Function Change	Max Abs Gradient
0	0	2	6.19289629		0.008696
0	1	5	6.19253485	-0.00036144	0.004212
0	2	8	6.19246549	-0.00006936	0.002424
0	3	11	6.19244193	-0.00002356	0.001333
0	4	13	6.19243242	-0.00000951	0.001446
0	5	16	6.19242952	-0.00000290	0.000422
0	6	19	6.19242860	-0.00000092	0.00015
0	7	22	6.19242847	-0.00000013	0.000072
0	8	25	6.19242842	-0.00000004	0.000033
0	9	28	6.19242841	-0.00000001	0.000023

Convergence criterion (GCONV=.000000010) satisfied.

Output 72.1.8 displays the model fit and item fit statistics. Note that the item fit statistics apply only to the binary items. That is why these fit statistics are missing for item6 to item8.

Output 72.1.8 Fit Statistics

The IRT Procedure

Model Fit Statistics	
Log Likelihood	-3096.214206
AIC (Smaller is Better)	6230.4284121
BIC (Smaller is Better)	6310.505966
LR Chi-Square	825.73118212
LR Chi-Square DF	844

Item Fit Statistics						
Response Model	Item	DF	Pearson Chi-Square	Pr > P ChiSq	LR Chi-Square	Pr > LR ChiSq
TwoP	item1	8	34.16298	<.0001	49.39389	<.0001
	item2	8	30.34200	0.0002	37.52299	<.0001
	item3	8	27.54720	0.0006	36.34739	<.0001
	item4	8	22.75949	0.0037	26.13355	0.0010
Graded	item5	8	18.32018	0.0190	19.68097	0.0116
	item6	0	.	.	.	.
	item7	0	.	.	.	.
	item8	0	.	.	.	.

The last table for this example is the "Item Parameter Estimates " table in Output 72.1.9. This table contains parameter estimates, standard errors, and p-values. These p-values suggest that all the parameters are significantly different from zero.

Output 72.1.9 Parameter Estimates

The IRT Procedure

Item Parameter Estimates					
Response Model	Item	Parameter	Estimate	Standard Error	Pr > |t|
TwoP	item1	Difficulty	0.27339	0.08301	0.0005
		Slope	0.98384	0.14145	<.0001
	item2	Difficulty	0.60265	0.10047	<.0001
		Slope	0.90014	0.13113	<.0001
	item3	Difficulty	0.46112	0.10062	<.0001
		Slope	0.79518	0.11392	<.0001
	item4	Difficulty	0.50688	0.14410	0.0002
		Slope	0.50432	0.08567	<.0001
Graded	item5	Threshold	-0.79747	0.18706	<.0001
		Slope	0.45391	0.08239	<.0001
	item6	Threshold 1	-0.59124	0.19855	0.0015
		Threshold 2	2.02711	0.39593	<.0001
		Slope	0.35772	0.06777	<.0001
	item7	Threshold 1	-0.82130	0.12753	<.0001
		Threshold 2	0.64436	0.11431	<.0001
		Slope	0.72673	0.09312	<.0001
	item8	Threshold 1	-1.08136	0.14134	<.0001
		Threshold 2	0.25167	0.09536	0.0042
		Slope	0.76377	0.09753	<.0001

Item characteristic curves (ICC) are also produced in this example. By default, these ICC plots are displayed in panels. To display an individual ICC plot for each item, use the UNPACK suboption in the PLOTS= option in the PROC IRT statement.

Output 72.1.10 ICC Plots

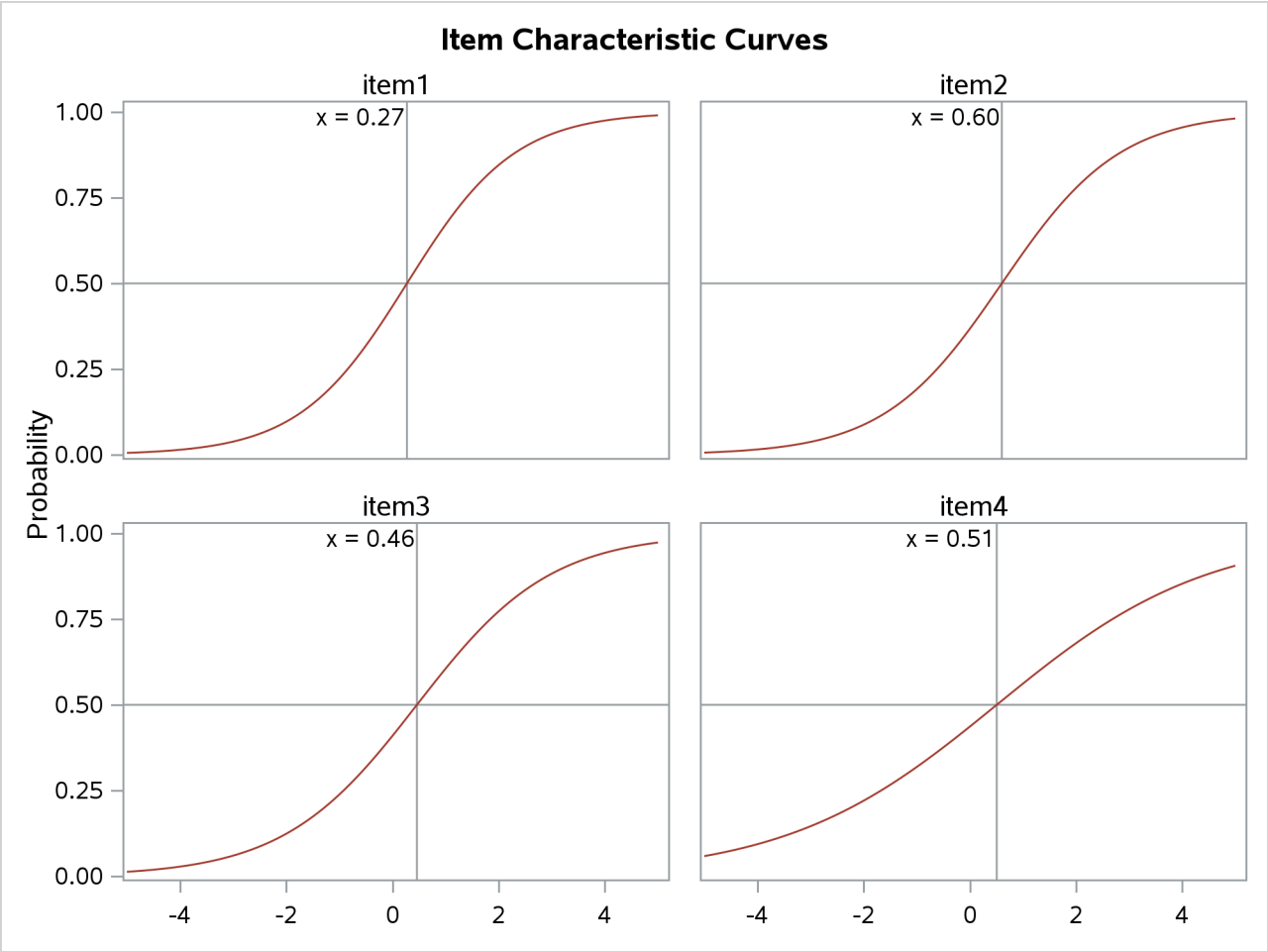

Output 72.1.10 *continued*

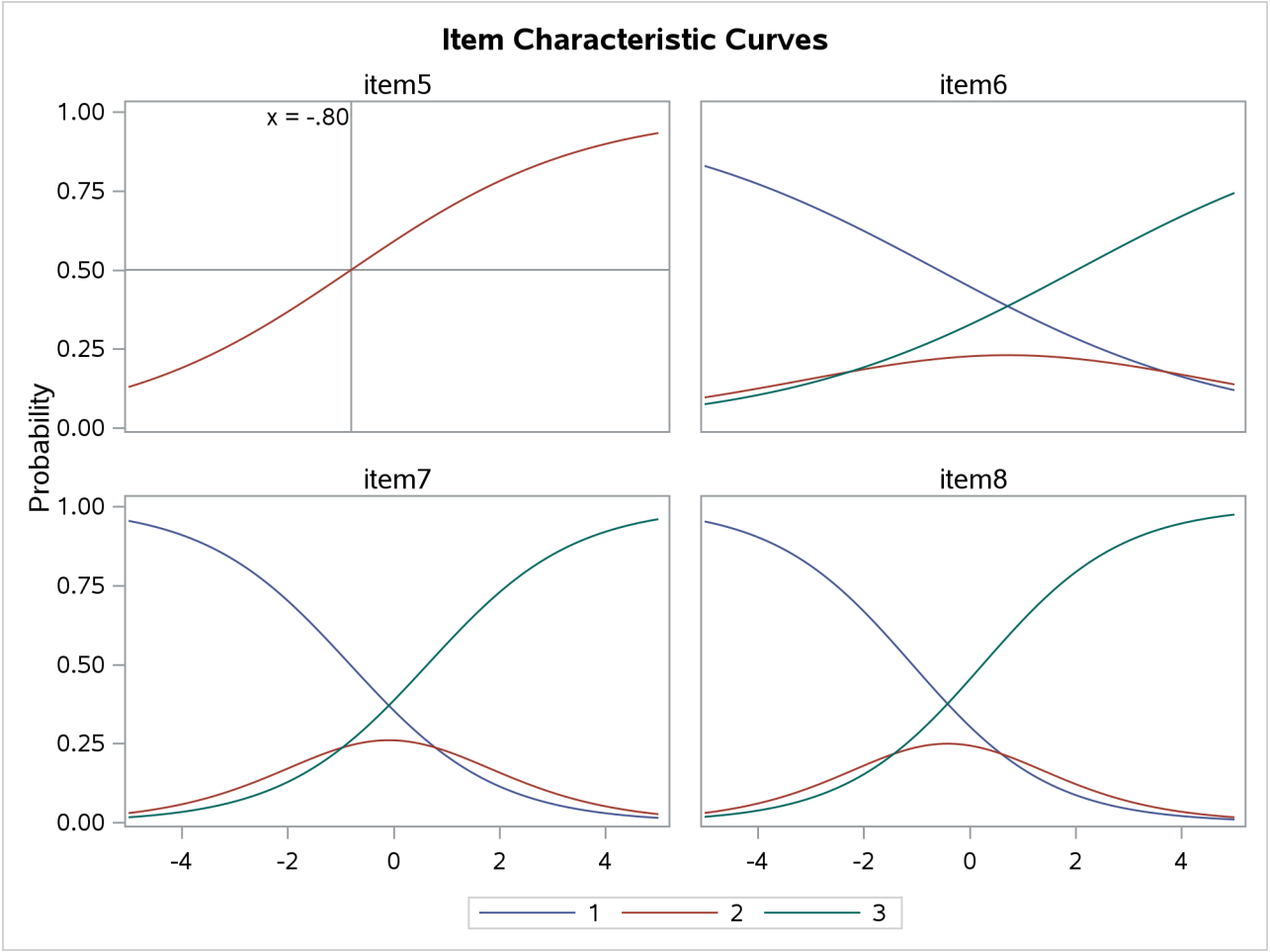

Now, suppose your research hypothesis includes some equality constraints on the model parameters—for example, the slopes for the first four items are equal. Such equality constraints can be specified easily by using the EQUALITY statement. In the following example, the slope parameters of the first four items are equal:

```
proc irt data=IrtUni;
   var item1-item8;
   model item1-item4/resfunc=twop, item5-item8/resfunc=graded;
   equality item1-item4/parm=[slope];
run;
```

To estimate the factor score for each subject and add these scores to the original data set, you can use the OUT= option in the PROC IRT statement. PROC IRT provides three factor score estimation methods: maximum likelihood (ML), maximum a posteriori (MAP), and expected a posteriori (EAP). You can choose an estimation method by using the SCOREMETHOD= option in the PROC IRT statement. The default method is maximum a posteriori. In the following, factor scores along with the original data are saved to a SAS data set called IrtUniFscore:

```
proc irt data=IrtUni out=IrtUniFscore;
   var item1-item8;
   model item1-item4/resfunc=twop,
         item5-item8/resfunc=graded;
   equality item1-item4/parm=[slope];
run;
```

Sometimes you might find it useful to sort the items based on the estimated difficulty or slope parameters. You can do this by outputting the ODS tables for the estimates into data sets and then sorting the items by using PROC SORT. A simulated data set is used to show the steps.

The following DATA step creates the data set IrtSimu:

```
data IrtSimu;
   input item1-item25 @@;
   datalines;
1 1 1 0 1 1 0 0 1 1 0 0 0 0 1 0 1 1 1 0 0 0 1 0 1 1 1 1 1 1 0 1 1 0 0
0 0 0 0 0 1 1 0 0 0 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0 0 1 1 0 0 0 0 0 0
0 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 1 0 0 0 1 0 1 1 0 1 1 0 0 1 1 0
1 0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1 0 1 1 1 1 1 1 0 1 0 1 1

   ... more lines ...

1 1 0 1 1 1 1 1 1 1 0 1 1 0 0 0 1 1 0 1 1 1 1 1 1 0 0 1 0 1 0 1 0 0 0
0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 0 1 1 1 0 1 1 1 1 1 1
;
```

First, you build the model and output the parameter estimates table into a SAS data set by using the ODS OUTPUT statement:

```
proc irt data=IrtSimu link=probit;
   var item1-item25;
   ods output ParameterEstimates=ParmEst;
run;
```

Output 72.1.11 shows the “Item Parameter Estimates” table. Notice that the difficulty and slope parameters are in the same column. The reason for this is to avoid having an extremely wide table when each item has a lot of parameters.

Output 72.1.11 Basic Information

The IRT Procedure

Item Parameter Estimates				
Item	Parameter	Estimate	Standard Error	Pr > |t|
item1	Difficulty	-1.32610	0.09788	<.0001
	Slope	1.44104	0.16074	<.0001
item2	Difficulty	-0.99737	0.07454	<.0001
	Slope	1.82022	0.18986	<.0001
item3	Difficulty	-1.25024	0.08981	<.0001
	Slope	1.58590	0.17476	<.0001
item4	Difficulty	-1.09619	0.07748	<.0001
	Slope	1.86629	0.20429	<.0001
item5	Difficulty	-1.07906	0.07806	<.0001
	Slope	1.78193	0.19059	<.0001
item6	Difficulty	-0.95076	0.09401	<.0001
	Slope	1.04083	0.10267	<.0001
item7	Difficulty	-0.65078	0.06949	<.0001
	Slope	1.45213	0.13449	<.0001
item8	Difficulty	-0.76383	0.07611	<.0001
	Slope	1.30273	0.12209	<.0001
item9	Difficulty	-0.72286	0.07058	<.0001
	Slope	1.50549	0.14221	<.0001
item10	Difficulty	-0.50732	0.06125	<.0001
	Slope	1.82143	0.17133	<.0001
item11	Difficulty	-0.01264	0.06470	0.4226
	Slope	1.26072	0.11259	<.0001
item12	Difficulty	0.04113	0.05584	0.2307
	Slope	2.01814	0.19993	<.0001
item13	Difficulty	0.16139	0.06878	0.0095
	Slope	1.12995	0.10180	<.0001
item14	Difficulty	0.01163	0.05671	0.4187
	Slope	1.88707	0.18046	<.0001
item15	Difficulty	0.07256	0.07360	0.1621
	Slope	0.96281	0.09035	<.0001
item16	Difficulty	-0.81428	0.07932	<.0001
	Slope	1.25213	0.11865	<.0001
item17	Difficulty	-0.92085	0.09315	<.0001
	Slope	1.02800	0.10107	<.0001
item18	Difficulty	-0.59398	0.06638	<.0001
	Slope	1.58234	0.14844	<.0001
item19	Difficulty	-0.97619	0.09767	<.0001
	Slope	0.97863	0.09745	<.0001
item20	Difficulty	-0.48845	0.05994	<.0001
	Slope	1.95453	0.18808	<.0001
item21	Difficulty	-0.60644	0.06851	<.0001
	Slope	1.45134	0.13403	<.0001
item22	Difficulty	-0.51245	0.06222	<.0001
	Slope	1.74231	0.16226	<.0001
item23	Difficulty	-0.90957	0.08476	<.0001

Output 72.1.11 *continued*

The IRT Procedure

Item Parameter Estimates				
Item	Parameter	Estimate	Standard Error	Pr > |t|
	Slope	1.20130	0.11604	<.0001
item24	Difficulty	-0.56501	0.06327	<.0001
	Slope	1.74199	0.16360	<.0001
item25	Difficulty	-0.58895	0.06750	<.0001
	Slope	1.48751	0.13759	<.0001

Output 72.1.12 The Difficulty Parameter SAS Data Set

Obs	Item	Difficulty
1	item1	-1.32610
2	item2	-0.99737
3	item3	-1.25024
4	item4	-1.09619
5	item5	-1.07906
6	item6	-0.95076
7	item7	-0.65078
8	item8	-0.76383
9	item9	-0.72286
10	item10	-0.50732
11	item11	-0.01264
12	item12	0.04113
13	item13	0.16139
14	item14	0.01163
15	item15	0.07256
16	item16	-0.81428
17	item17	-0.92085
18	item18	-0.59398
19	item19	-0.97619
20	item20	-0.48845
21	item21	-0.60644
22	item22	-0.51245
23	item23	-0.90957
24	item24	-0.56501
25	item25	-0.58895

Then you save the estimates of slopes and difficulties in the data set ParmEst and create two separate data sets to store the difficulty and slope parameters:

```
data Diffs(keep=Item Difficulty);
   set ParmEst;
   Difficulty = Estimate;
   if (Parameter = "Difficulty") then output;
run;
```

```
proc print data=Diffs;
run;

data Slopes(keep=Item Slope);
   set ParmEst;
   Slope = Estimate;
   if (Parameter = "Slope") then output;
run;
proc print data=Slopes;
run;
```

The two SAS data sets are shown in Output 72.1.12 and Output 72.1.13.

Output 72.1.13 The Slope Parameter SAS Data Set

Obs	Item	Slope
1	item1	1.44104
2	item2	1.82022
3	item3	1.58590
4	item4	1.86629
5	item5	1.78193
6	item6	1.04083
7	item7	1.45213
8	item8	1.30273
9	item9	1.50549
10	item10	1.82143
11	item11	1.26072
12	item12	2.01814
13	item13	1.12995
14	item14	1.88707
15	item15	0.96281
16	item16	1.25213
17	item17	1.02800
18	item18	1.58234
19	item19	0.97863
20	item20	1.95453
21	item21	1.45134
22	item22	1.74231
23	item23	1.20130
24	item24	1.74199
25	item25	1.48751

Now you can use PROC SORT to sort the items by either difficulty or slope as follows:

```
proc sort data=Diffs;
   by Difficulty;
run;
proc print data=Diffs;
run;
```

```
proc sort data=Slopes;
   by Slope;
run;
proc print data=Slopes;
run;
```

Output 72.1.14 and Output 72.1.15 show the sorted data sets.

Output 72.1.14 Items Sorted by Difficulty

Obs	Item	Difficulty
1	item1	-1.32610
2	item3	-1.25024
3	item4	-1.09619
4	item5	-1.07906
5	item2	-0.99737
6	item19	-0.97619
7	item6	-0.95076
8	item17	-0.92085
9	item23	-0.90957
10	item16	-0.81428
11	item8	-0.76383
12	item9	-0.72286
13	item7	-0.65078
14	item21	-0.60644
15	item18	-0.59398
16	item25	-0.58895
17	item24	-0.56501
18	item22	-0.51245
19	item10	-0.50732
20	item20	-0.48845
21	item11	-0.01264
22	item14	0.01163
23	item12	0.04113
24	item15	0.07256
25	item13	0.16139

Output 72.1.15 Items Sorted by Slope

Obs	Item	Slope
1	item15	0.96281
2	item19	0.97863
3	item17	1.02800
4	item6	1.04083
5	item13	1.12995
6	item23	1.20130
7	item16	1.25213
8	item11	1.26072
9	item8	1.30273
10	item1	1.44104
11	item21	1.45134
12	item7	1.45213
13	item25	1.48751
14	item9	1.50549
15	item18	1.58234
16	item3	1.58590
17	item24	1.74199
18	item22	1.74231
19	item5	1.78193
20	item2	1.82022
21	item10	1.82143
22	item4	1.86629
23	item14	1.88707
24	item20	1.95453
25	item12	2.01814

Notice that the sorting does not work correctly if any of the items have more than one threshold (ordinal response) or slope (multidimensional model).

Now, suppose you want to group the items into subgroups based on their difficulty parameters and then sort the items in each subgroup by their slope parameters. First, you need to merge the two data sets, Diffs and Slopes, into one data set. Then, you add another variable, called DiffLevel, to indicate the subgroups. The following statements show these steps:

```
proc sort data=Slopes;
   by Item;
run;
proc sort data=Diffs;
   by Item;
run;
data ItemEst;
   merge Diffs Slopes;
   by Item;
   if Difficulty < -1.0 then DiffLevel = 1;
   else if Difficulty < 0 then DiffLevel = 2;
   else if Difficulty < 1 then DiffLevel = 3;
   else DiffLevel = 4;
run;
```

```
proc print data=ItemEst;
run;
```

Output 72.1.16 shows the merged data set.

Output 72.1.16 The Merged SAS Data Set

Obs	Item	Difficulty	Slope	DiffLevel
1	item1	-1.32610	1.44104	1
2	item10	-0.50732	1.82143	2
3	item11	-0.01264	1.26072	2
4	item12	0.04113	2.01814	3
5	item13	0.16139	1.12995	3
6	item14	0.01163	1.88707	3
7	item15	0.07256	0.96281	3
8	item16	-0.81428	1.25213	2
9	item17	-0.92085	1.02800	2
10	item18	-0.59398	1.58234	2
11	item19	-0.97619	0.97863	2
12	item2	-0.99737	1.82022	2
13	item20	-0.48845	1.95453	2
14	item21	-0.60644	1.45134	2
15	item22	-0.51245	1.74231	2
16	item23	-0.90957	1.20130	2
17	item24	-0.56501	1.74199	2
18	item25	-0.58895	1.48751	2
19	item3	-1.25024	1.58590	1
20	item4	-1.09619	1.86629	1
21	item5	-1.07906	1.78193	1
22	item6	-0.95076	1.04083	2
23	item7	-0.65078	1.45213	2
24	item8	-0.76383	1.30273	2
25	item9	-0.72286	1.50549	2

Then, you can sort the items by slope within each difficulty group as follows:

```
proc sort data=ItemEst;
   by difflevel slope;
run;
proc print data=ItemEst;
run;
```

Output 72.1.17 shows the data set after sorting.

Output 72.1.17 Item Sorted by Slope within Each Difficulty Group

Obs	Item	Difficulty	Slope	DiffLevel
1	item1	-1.32610	1.44104	1
2	item3	-1.25024	1.58590	1
3	item5	-1.07906	1.78193	1
4	item4	-1.09619	1.86629	1
5	item19	-0.97619	0.97863	2
6	item17	-0.92085	1.02800	2
7	item6	-0.95076	1.04083	2
8	item23	-0.90957	1.20130	2
9	item16	-0.81428	1.25213	2
10	item11	-0.01264	1.26072	2
11	item8	-0.76383	1.30273	2
12	item21	-0.60644	1.45134	2
13	item7	-0.65078	1.45213	2
14	item25	-0.58895	1.48751	2
15	item9	-0.72286	1.50549	2
16	item18	-0.59398	1.58234	2
17	item24	-0.56501	1.74199	2
18	item22	-0.51245	1.74231	2
19	item2	-0.99737	1.82022	2
20	item10	-0.50732	1.82143	2
21	item20	-0.48845	1.95453	2
22	item15	0.07256	0.96281	3
23	item13	0.16139	1.12995	3
24	item14	0.01163	1.88707	3
25	item12	0.04113	2.01814	3

Example 72.2: Multidimensional Exploratory and Confirmatory IRT Models

(View the complete code for this example (irtex2.sas) in the example repository.)

This example illustrates how to use the IRT procedure to fit multidimensional exploratory and confirmatory IRT models. The data set that is introduced in Example 72.1 is also used here. Two more items, item9 and item10, are added to the data set. These two items are designed to measure subjects' satisfaction with their friendships and their family life, respectively.

```
data IrtMulti;
   input item1-item10 @@;
   datalines;
1 0 0 0 1 1 2 1 2 1 1 1 1 1 1 3 3 3 3 3 0 1 0 0 1 1 1 1 1 1 1 0 0 1 0
1 2 3 2 2 0 0 0 0 0 1 1 1 1 1 1 0 0 1 0 1 3 3 1 2 0 0 0 0 0 1 1 3 3 2
0 0 1 0 0 1 2 2 3 2 0 1 0 0 1 1 1 2 2 2 0 0 0 0 0 2 2 3 3 2 0 1 0 1 0
2 3 3 3 3 0 0 1 0 1 1 2 3 2 3 1 1 1 1 1 2 2 3 2 2 0 0 0 0 1 1 2 2 3 1
1 0 1 1 1 2 3 3 2 3 0 1 0 0 1 1 2 3 3 3 1 0 1 1 1 2 3 3 3 3 0 1 0 1 1
3 2 3 3 2 1 1 1 0 0 1 3 3 2 1 1 1 0 0 1 2 3 3 3 3 0 1 1 1 1 1 2 1 2 3
1 0 0 1 1 3 1 1 1 1 1 0 0 0 1 1 1 3 3 1 0 0 0 0 1 1 3 3 3 3 0 0 0 0 0
1 1 1 3 2 1 0 0 0 0 1 3 3 3 3 1 1 0 1 1 3 1 1 3 3 1 0 1 1 1 1 3 1 1 1
```

```
   ... more lines ...

3 3 1 3 2 0 0 0 1 0 1 3 2 2 1 0 0 0 0 1 1 2 2 2 3 1 0 1 0 1 2 2 3 2 1
1 0 0 1 1 1 2 2 3 1 0 1 0 1 0 1 3 1 1 1 0 1 1 0 1 3 3 3 3 2 1 0 1 0 0
1 2 1 1 1 1 0 1 1 0 1 3 3 1 3 1 1 0 1 0 2 2 2 2 3 1 1 0 1 1 3 2 3 2 2
0 0 0 1 0 2 2 3 1 2 0 0 0 1 0 2 3 3 3 2 0 1 0 0 1 2 2 1 2 1
;
```

Now, suppose that previous research results suggest that two latent factors underlie these 10 items. However, knowledge about the factor structure is very limited. The first step you can take is to fit an exploratory IRT model by using two factors. This can be accomplished easily by submitting the following statements:

```
ods graphics on;
proc irt data=IrtMulti nfactor=2 plots=scree;
   var item1-item10;
run;
```

The first table that you want to check is the "Eigenvalue" table, shown in Output 72.2.1. There are only two eigenvalues greater than 1 in this example. This result, to some extent, suggests that two factors might be enough in this example. Output 72.2.2 include the scree and variance explained plots.

Output 72.2.1 Eigenvalues of the Polychoric Correlation matrix

The IRT Procedure

Eigenvalues of the Polychoric Correlation Matrix

	Eigenvalue	Difference	Proportion	Cumulative
1	3.65750431	2.47883117	0.3658	0.3658
2	1.17867314	0.23738137	0.1179	0.4836
3	0.94129177	0.04672399	0.0941	0.5777
4	0.89456778	0.07508308	0.0895	0.6672
5	0.81948471	0.14774320	0.0819	0.7492
6	0.67174151	0.12081203	0.0672	0.8163
7	0.55092948	0.01698300	0.0551	0.8714
8	0.53394648	0.08647230	0.0534	0.9248
9	0.44747418	0.14308753	0.0447	0.9696
10	0.30438664		0.0304	1.0000

Output 72.2.2 Scree and Variance Explained Plots

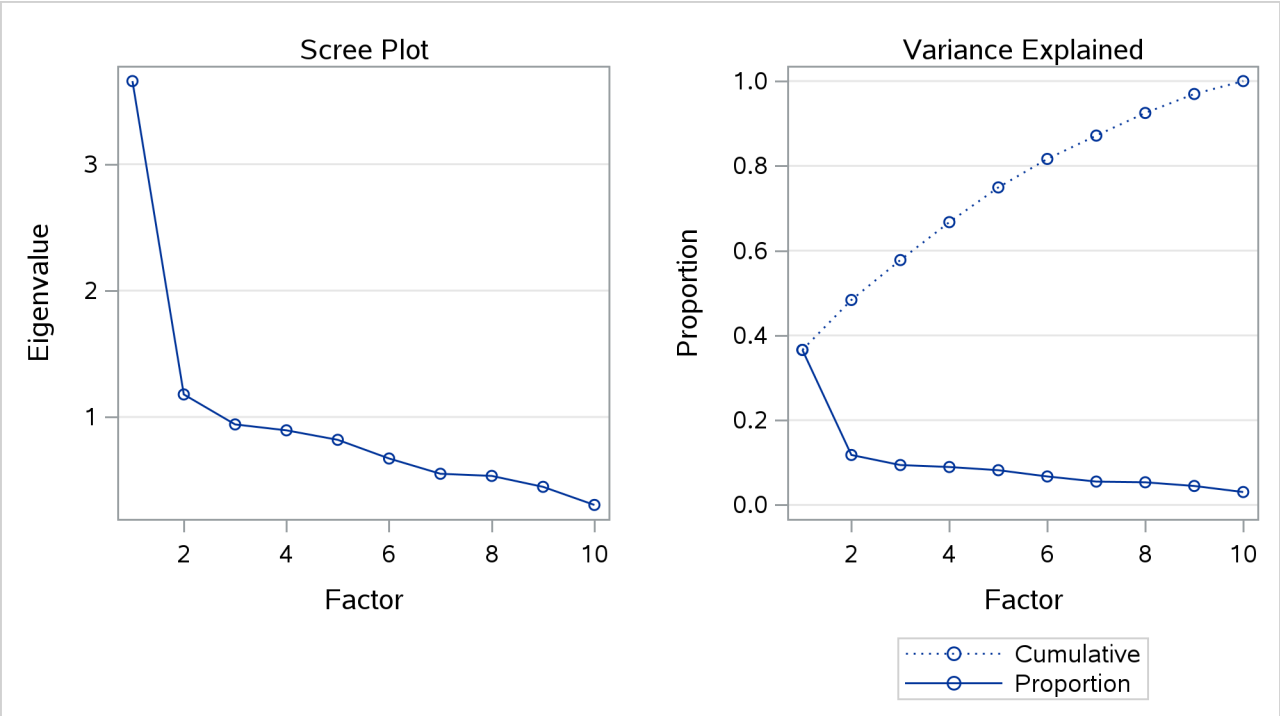

If the optimization algorithm converges successfully, the original and rotated slope matrices are produced. The default rotation method is varimax. You can use the ROTATE= option in the PROC IRT statement to specify a different rotation method.

Output 72.2.3 Slope Matrix

Rotated Slope Matrix		
	_Factor1	_Factor2
item1	1.87328	0.63493
item2	1.71933	0.56361
item3	1.29600	0.54150
item4	0.74883	0.36353
item5	0.41013	0.63816
item6	0.44024	0.39946
item7	0.68656	1.18510
item8	0.74020	1.97272
item9	0.40198	1.26214
item10	0.40047	1.26491

This example uses the default varimax rotation. Output 72.2.3 shows the rotated slope matrices. The rotated slope matrix is displayed in the standard matrix format. From the rotated slope matrix, you can see that the first factor is mainly reflected by item1 to item4 and item6, and the second factor is mainly reflected by the rest of the items. The exploratory results suggest a hypothesis about the factor structure of the items. In practice, you might want to confirm this structure by a confirmatory analysis of the new data. However, for illustration purposes, the same data set is used here to demonstrate the confirmatory model fitting by using

the following statements:

```
proc irt data=IrtMulti;
   var item1-item4 item6 item5 item7-item10;
   factor  Factor1->item1-item4 item6,
           Factor2->item5 item7-item10;
run;
```

Output 72.2.5 and Output 72.2.4 show the model fit statistics for the confirmatory model and the exploratory model, respectively. Output 72.2.6 shows the slope matrix from this confirmatory analysis. Because the number of response patterns in the data set, 500, is much lower than the total number of possible response patterns, $2^5 \times 3^5 = 7{,}776$, you cannot use Pearson's chi-square or the likelihood ratio statistic to test the overall fit of the confirmatory model.

If you have two nested models, you can still use the likelihood ratio statistic to test the difference between these two models. For illustration purposes, assume that the exploratory model and the confirmatory model are two independent models. Because the confirmatory model is nested within the exploratory model, you can use the likelihood ratio test to compare the two models. The likelihood ratio test statistic is 411.1, which is the difference between the LR chi-square values in Output 72.2.4 and Output 72.2.5. The degree of freedom is 9. The corresponding p-value is 0, which suggests that the difference between these two models is significant. There are many options that you can use to improve the model fit. For this example, you can try to free some of the fixed slope parameters or free the covariance between the factors.

Output 72.2.4 Model Fit Statistics for Exploratory Model

The IRT Procedure

Model Fit Statistics	
Log Likelihood	-3921.316112
AIC (Smaller is Better)	7910.6322244
BIC (Smaller is Better)	8053.9288997
LR Chi-Square	1898.3368937
LR Chi-Square DF	7741

Output 72.2.5 Model Fit Statistics for Confirmatory Model

The IRT Procedure

Model Fit Statistics	
Log Likelihood	-4126.854963
AIC (Smaller is Better)	8303.709927
BIC (Smaller is Better)	8409.0751295
LR Chi-Square	2309.4145963
LR Chi-Square DF	7750

Output 72.2.6 Slope Matrix

Slope Matrix Estimate/StdErr/p-value		
	Factor1	**Factor2**
item1	2.03252 0.40958 <.00001	0.00000
item2	1.77531 0.32638 <.00001	0.00000
item3	1.35921 0.22797 <.00001	0.00000
item4	0.80419 0.15626 <.00001	0.00000
item6	0.55163 0.12128 <.00001	0.00000
item5	0.00000	0.73736 0.13901 <.00001
item7	0.00000	1.33310 0.17340 <.00001
item8	0.00000	2.14117 0.31840 <.00001
item9	0.00000	1.29816 0.16674 <.00001
item10	0.00000	1.30773 0.17013 <.00001

Example 72.3: Multiple-Group Analysis

(View the complete code for this example (irtex3.sas) in the example repository.)

This example shows how to use the IRT procedure to do multiple-group analysis. The following DATA step creates the data set IrtGroup:

```
data IrtGroup;
   input item1-item8 GroupVar @@;
   datalines;
1 0 0 0 1 1 2 1 2 1 1 1 1 1 3 3 3 1 0 1 0 0 1 1 1 1 1 1 0 0 1 0 1 2 3
2 0 0 0 0 0 1 1 1 1 1 0 0 1 0 1 3 3 1 0 0 0 0 0 1 1 3 2 0 0 1 0 0 1 2
2 1 0 1 0 0 1 1 1 2 1 0 0 0 0 0 2 2 3 1 0 1 0 1 0 2 3 3 2 0 0 1 0 1 1
2 3 1 1 1 1 1 1 2 2 3 2 0 0 0 0 1 1 2 2 1 1 0 1 1 1 2 3 3 2 0 1 0 0 1
1 2 3 2 1 0 1 1 1 2 3 3 2 0 1 0 1 1 3 2 3 2 1 1 1 0 0 1 3 3 1 1 1 0 0
1 2 3 3 2 0 1 1 1 1 1 2 1 2 1 0 0 1 1 3 1 1 2 1 0 0 0 1 1 1 3 2 0 0 0
0 1 1 3 3 1 0 0 0 0 0 1 1 1 2 1 0 0 0 0 1 3 3 2 1 1 0 1 1 3 1 1 1 1 0
1 1 1 1 3 1 2 1 1 1 1 1 2 3 2 2 0 0 1 0 0 2 2 2 1 0 0 1 0 1 1 2 3 2 1
```

```
   ... more lines ...

1 0 0 2 1 3 2 1 1 1 1 1 1 3 2 1 1 1 1 0 0 3 3 1 2 1 0 0 1 1 3 3 3 2 0
0 1 0 0 1 1 1 2 0 0 0 0 1 3 1 1 2 1 0 0 1 0 1 3 3 2 0 0 1 1 0 2 2 3 2
1 0 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 2 0 0 1 1 0 1 1 3 2 0 0 0 0 1 1 1 1
2 0 0 1 0 0 1 1 1 1
;
```

To set up a multiple-group IRT model, you need to specify the grouping variable by using the GROUP statement. Very often you also want to specify cross-group equality constraints for different parameters. You can accomplish this by using the EQUALITY statement.

The model that is specified in the following statements is an extension of the model in Example 72.1. The group variable, GroupVar, is specified in the GROUP statement. It has two values, 1 and 2, to indicate group membership. Equality constraints are specified in the EQUALITY statement.

```
proc irt data=IrtGroup;
   var item1-item8;
   group GroupVar;
   model item1-item4/resfunc=twop,
         item5-item8/resfunc=graded;
   equality item1-item4/parm=[intercept] between_gp=[1 2],
            _allgr_/parm=[slope] within_gp=[1];
run;
```

Two different sets of equality constraints have been specified. The first entry specifies equality constraints on the intercept parameters for item1 to item4 between group 1 and group 2:

```
item1-item4/parm=[intercept] between_gp=[1 2]
```

Notice that 1 and 2 are the actual values for the group variable GroupVar. The second entry specifies equality constraints on the slope parameters for all the graded response items within group 1:

```
_allgr_/parm=[slope] within_gp=[1]
```

Output 72.3.1 shows the “Modeling Information” table and the “Group Information” table for this example. For multiple-group analysis, the “Modeling Information” table contains two extra pieces of information: the group variable and the number of groups. The “Group Information” table contains information about the data for each group. There are 272 observations that have been read and used for group 1; this number for group 2 is 328.

Output 72.3.1 Modeling and Group Information

The IRT Procedure

Modeling Information	
Data Set	WORK.IRTGROUP
Group Variable	GroupVar
Link Function	Logit
Number of Items	8
Number of Factors	1
Number of Groups	2
Number of Observations Read	600
Number of Observations Used	600
Estimation Method	Marginal Maximum Likelihood

Group Information		
GroupVar	**Nobs Read**	**Nobs Used**
1	272	272
2	328	328

Because there are two groups in this example, the IRT procedure produces two "Item Information" tables. For this example, these two tables contain the same information. That means that all the items have the same levels for the two groups. It is possible that the same item might have different numbers of levels, or maybe the same number of levels but different values. For example, an item has four levels, from 1 to 4, but one group might observe only levels 1 and 2, and the other group might observe only levels 3 and 4.

Output 72.3.2 Item Information

The IRT Procedure

Item Information GroupVar = 1			
Response Model	**Item**	**Levels**	**Values**
TwoP	**item1**	2	0 1
	item2	2	0 1
	item3	2	0 1
	item4	2	0 1
Graded	**item5**	2	0 1
	item6	3	1 2 3
	item7	3	1 2 3
	item8	3	1 2 3

Output 72.3.2 *continued*

Item Information GroupVar = 2			
Response Model	Item	Levels	Values
TwoP	item1	2	0 1
	item2	2	0 1
	item3	2	0 1
	item4	2	0 1
Graded	item5	2	0 1
	item6	3	1 2 3
	item7	3	1 2 3
	item8	3	1 2 3

Output 72.3.3 includes "Item Parameter Estimates" tables for both groups. You can see that the intercept parameters for item1 are the same for both groups. The same applies to item2 to item4. You can also see that the slope parameters have the same value for item5 to item8 in group 1. These results suggest that equality constraints that are specified in the EQUALITY statement have been fulfilled.

Output 72.3.3 Parameter Estimates

The IRT Procedure

Item Parameter Estimates GroupVar = 1					
Response Model	Item	Parameter	Estimate	Standard Error	Pr > |t|
TwoP	item1	Difficulty	0.24624	0.07816	0.0008
		Slope	1.89146	0.37443	<.0001
	item2	Difficulty	0.49591	0.10121	<.0001
		Slope	2.00190	0.37931	<.0001
	item3	Difficulty	0.41600	0.10696	<.0001
		Slope	1.40297	0.26917	<.0001
	item4	Difficulty	0.57875	0.16206	0.0002
		Slope	0.90521	0.20197	<.0001
Graded	item5	Threshold	-0.71306	0.16162	<.0001
		Slope	0.96022	0.10851	<.0001
	item6	Threshold 1	-0.15738	0.14694	0.1421
		Threshold 2	1.42608	0.21480	<.0001
		Slope	0.96022	0.10851	<.0001
	item7	Threshold 1	-0.86491	0.16877	<.0001
		Threshold 2	0.81551	0.17162	<.0001
		Slope	0.96022	0.10851	<.0001
	item8	Threshold 1	-1.24153	0.19175	<.0001
		Threshold 2	0.37686	0.15211	0.0066
		Slope	0.96022	0.10851	<.0001

Output 72.3.3 *continued*

Item Parameter Estimates
GroupVar = 2

Response Model	Item	Parameter	Estimate	Standard Error	Pr > |t|
TwoP	item1	Difficulty	0.32086	0.09340	0.0003
		Slope	1.45159	0.28556	<.0001
	item2	Difficulty	0.68895	0.12796	<.0001
		Slope	1.44099	0.28556	<.0001
	item3	Difficulty	0.51534	0.12286	<.0001
		Slope	1.13253	0.22767	<.0001
	item4	Difficulty	0.61810	0.16239	<.0001
		Slope	0.84757	0.18933	<.0001
Graded	item5	Threshold	-0.90708	0.27388	0.0005
		Slope	0.68620	0.17589	<.0001
	item6	Threshold 1	-0.77953	0.29941	0.0046
		Threshold 2	2.21412	0.57510	<.0001
		Slope	0.54001	0.14313	<.0001
	item7	Threshold 1	-0.84382	0.18252	<.0001
		Threshold 2	0.80748	0.16758	<.0001
		Slope	1.03841	0.18991	<.0001
	item8	Threshold 1	-1.18595	0.20728	<.0001
		Threshold 2	0.20022	0.12031	0.0480
		Slope	1.15476	0.21093	<.0001

Example 72.4: Item Selection Using Item and Test Information

(View the complete code for this example (irtex4.sas) in the example repository.)

The data set in this example comes from the 1978 Quality of American Life Survey. The survey was administered to a sample of US residents aged 18 years and older in 1978. Subjects were asked to rate their satisfaction with many different aspects of their lives. This example includes 14 items. Some of the items are as follows:

- satisfaction with community
- satisfaction with neighbors
- satisfaction with amount of education received
- satisfaction with health
- satisfaction with job
- satisfaction with income

Originally these items were designed with seven-point scales, where 1 indicates most unsatisfied and 7 indicates most satisfied. For illustration purposes, these items have been reorganized into a different number of categories, which ranges from 2 to 7. This example uses 1,000 random samples from the original data set. The following DATA step creates the data set IrtQls:

```
data IrtQls;
   input item1-item14 @@;
   datalines;
 1   1   2   1   1    2    2    2    .    2    2    2    2    2
 2   2   2   2   2    3    4    1    .    2    5    6    4    4

   ... more lines ...

 1   1   1   1   2    2    2    2    .    1    1    1    1    3
;
```

By default, the IRT procedure uses the graded response model (GRM) and the logistic link for all the ordinal items and uses the two-parameter logistic model for all the binary items. In PROC IRT, you can specify different types of response models for different items by using the MODEL statement.

Because all the items in this example are designed to measure subjects' satisfaction with their lives, it is reasonable to start with a unidimensional IRT model. The following statements fit such a model by using the default model options:

```
ods graphics on;
proc irt data=IrtQls plots=(IIC TIC);
   var item1-item14;
run;
```

This example requests item information curves (IICs) and a test information curve (TIC) by using the PLOTS=(IIC TIC) option.

Output 72.4.1 Eigenvalues of Polychoric Correlations

The IRT Procedure

Eigenvalues of the Polychoric Correlation Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	5.57173396	4.19614614	0.3980	0.3980
2	1.37558781	0.29273244	0.0983	0.4962
3	1.08285537	0.12600033	0.0773	0.5736
4	0.95685504	0.09108909	0.0683	0.6419
5	0.86576595	0.09758221	0.0618	0.7038
6	0.76818374	0.12571683	0.0549	0.7586
7	0.64246691	0.06108305	0.0459	0.8045
8	0.58138386	0.04214553	0.0415	0.8461
9	0.53923833	0.10092835	0.0385	0.8846
10	0.43830998	0.07346977	0.0313	0.9159
11	0.36484021	0.04667935	0.0261	0.9419
12	0.31816085	0.03905135	0.0227	0.9647
13	0.27910950	0.06360101	0.0199	0.9846
14	0.21550849		0.0154	1.0000

Output 72.4.1 shows the eigenvalue table for this example. You can see that the first eigenvalue is much greater than the others, suggesting that a unidimensional model is reasonable for the data.

In the context of item response theory, the amount of information that each item or the entire test provides

might not be evenly distributed across the entire continuum of latent constructs. The value of the slope parameter indicates the amount of information that the item provides. For this example, parameter estimates and item information curves are shown in Output 72.4.2 and Output 72.4.3, respectively. By examining the parameter estimates and the item information curves, you can see that items that have high slope values have tall, narrow information curves. For example, because the slope value of item9 is much larger than the slope value of item1, the information curve is taller and narrower for item9 than it is for item1.

Output 72.4.2 Parameter Estimates

The IRT Procedure

Item Parameter Estimates

Response Model	Item	Parameter	Estimate	Standard Error	Pr > |t|
Graded	item1	Threshold 1	-2.10582	0.34519	<.0001
		Threshold 2	3.26933	0.51162	<.0001
		Slope	0.45285	0.07033	<.0001
	item2	Threshold 1	-0.54239	0.07763	<.0001
		Threshold 2	0.47609	0.07308	<.0001
		Slope	1.20090	0.09670	<.0001
	item5	Threshold 1	-0.71111	0.09185	<.0001
		Threshold 2	0.62254	0.08600	<.0001
		Slope	1.03345	0.08727	<.0001
	item6	Threshold 1	-0.59487	0.11777	<.0001
		Threshold 2	1.20725	0.15234	<.0001
		Slope	0.70710	0.07601	<.0001
	item7	Threshold 1	-0.77676	0.06523	<.0001
		Threshold 2	0.25887	0.05286	<.0001
		Threshold 3	0.89740	0.06606	<.0001
		Slope	1.88550	0.12224	<.0001
	item8	Threshold 1	-0.73876	0.07456	<.0001
		Threshold 2	0.62583	0.06856	<.0001

Output 72.4.2 *continued*

The IRT Procedure

Item Parameter Estimates					
Response Model	Item	Parameter	Estimate	Standard Error	Pr > |t|
		Threshold 3	1.38498	0.09804	<.0001
		Slope	1.40786	0.09825	<.0001
	item10	Threshold 1	-0.32389	0.05950	<.0001
		Threshold 2	0.69947	0.06626	<.0001
		Slope	1.66250	0.12039	<.0001
	item11	Threshold 1	-1.02195	0.07811	<.0001
		Threshold 2	-0.01850	0.05698	0.3727
		Threshold 3	0.66511	0.06466	<.0001
		Threshold 4	1.37499	0.09022	<.0001
		Slope	1.65678	0.10894	<.0001
	item12	Threshold 1	-1.87386	0.13806	<.0001
		Threshold 2	-0.79800	0.08675	<.0001
		Threshold 3	-0.08247	0.06987	0.1190
		Threshold 4	0.62081	0.07745	<.0001
		Threshold 5	1.25476	0.10243	<.0001
		Threshold 6	1.86435	0.13716	<.0001
		Slope	1.18631	0.08592	<.0001
	item13	Threshold 1	-0.80462	0.05879	<.0001

The IRT Procedure

Item Parameter Estimates					
Response Model	Item	Parameter	Estimate	Standard Error	Pr > |t|
		Threshold 2	0.33313	0.04823	<.0001
		Threshold 3	1.10470	0.06442	<.0001
		Slope	2.48218	0.16037	<.0001
	item14	Threshold 1	-1.36395	0.09895	<.0001
		Threshold 2	0.37197	0.06328	<.0001
		Threshold 3	1.38156	0.09739	<.0001
		Slope	1.39347	0.09606	<.0001
TwoP	item3	Difficulty	0.00531	0.09877	0.4786
		Slope	0.72075	0.08488	<.0001
	item4	Difficulty	-0.35571	0.07124	<.0001
		Slope	1.22884	0.11089	<.0001
	item9	Difficulty	0.18412	0.06545	0.0025
		Slope	1.84568	0.20071	<.0001

For individual items, most of the information concentrates around the area that is defined by the difficulty parameters. The binary response item provides most of the information around the difficulty parameter. For ordinal items, most of the information falls in the region between the lowest and the highest threshold parameters. By comparing the information curves for item7 and item9, you can also see that when response items have the same slope value, the ordinal item is more informative than the binary item.

Output 72.4.3 Item Information Curves

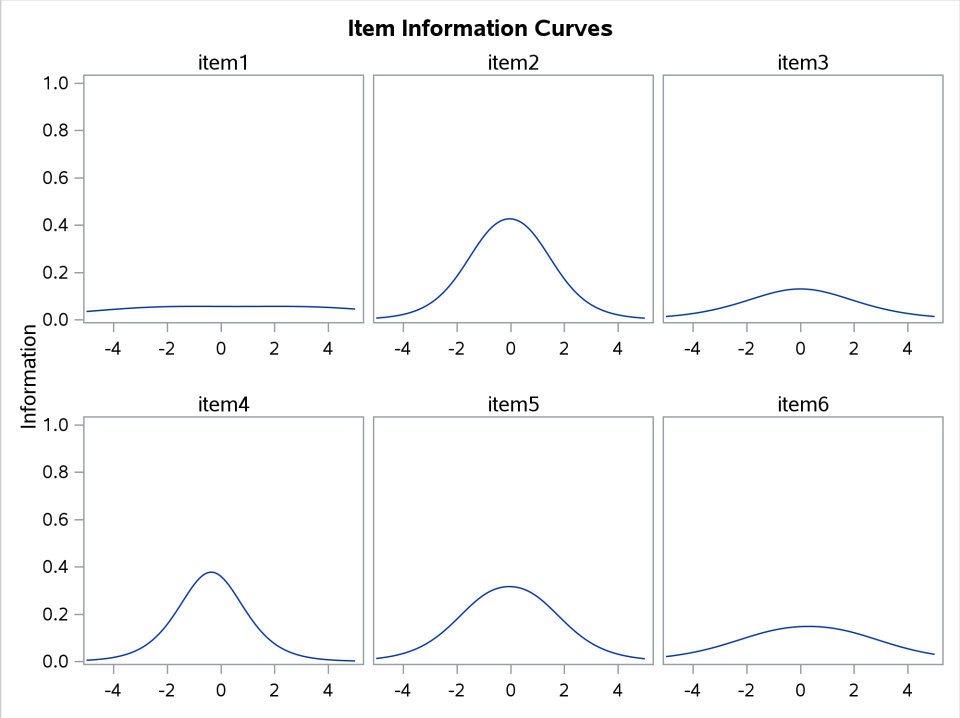

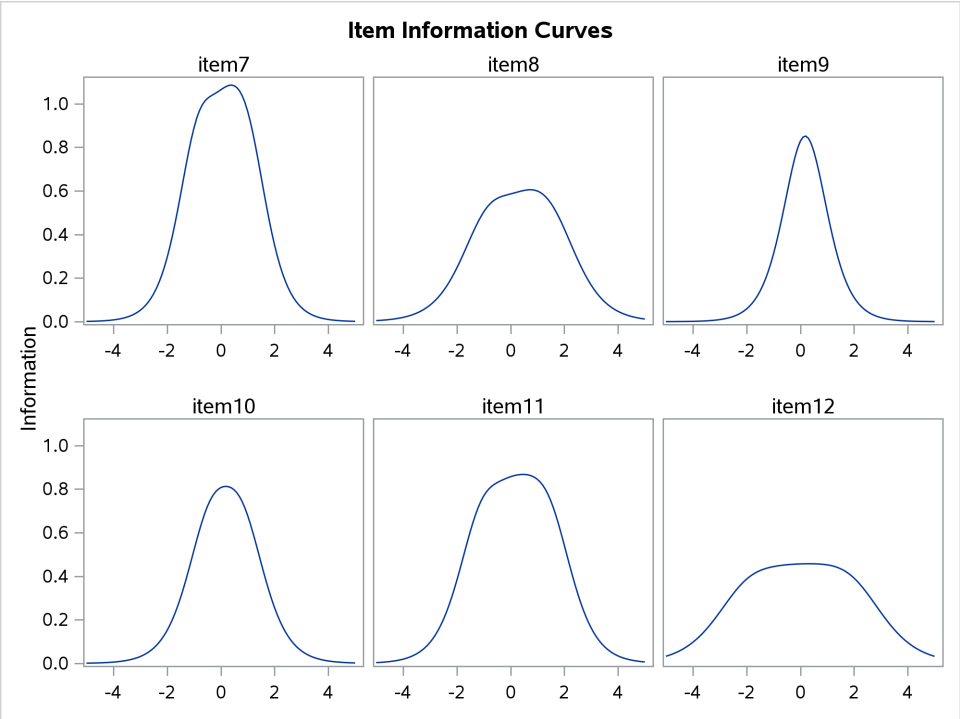

Output 72.4.3 *continued*

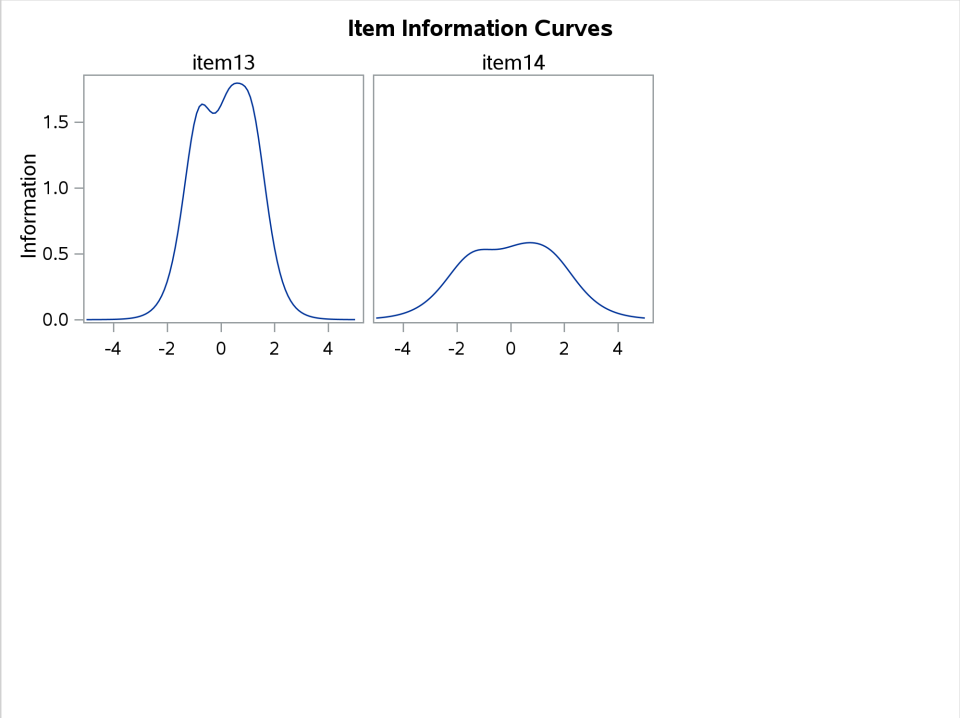

Output 72.4.4 Test Information Curves

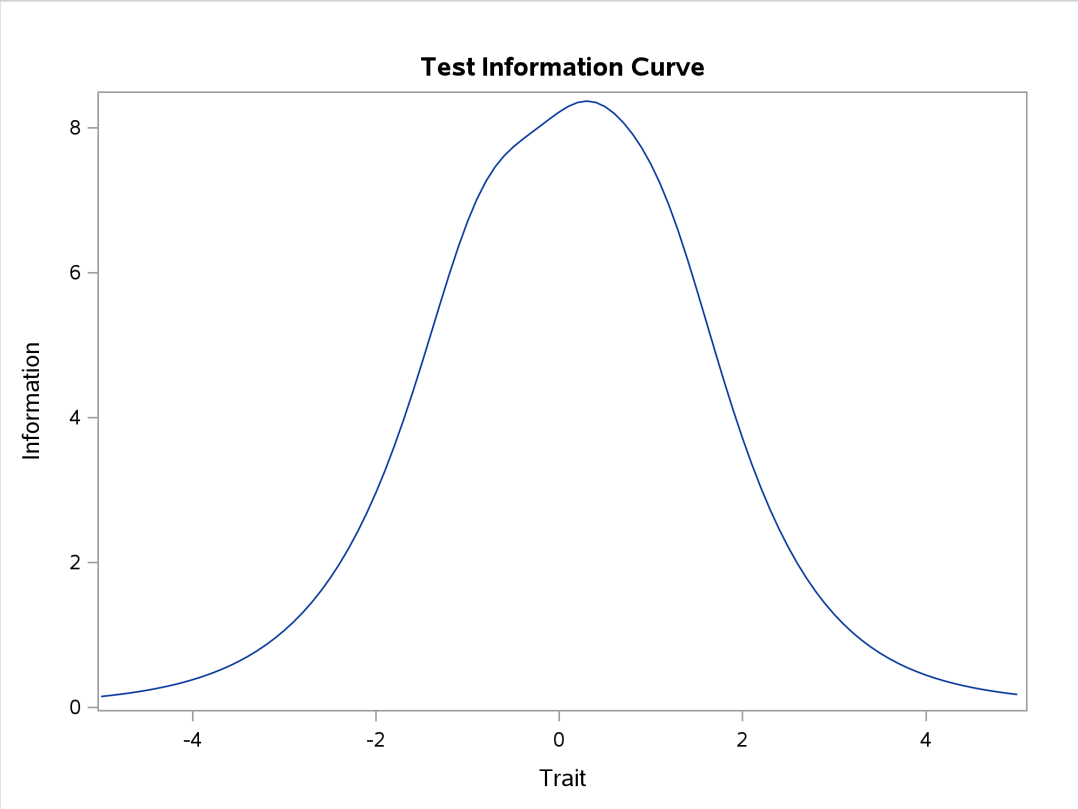

When all items in a test are considered together, the information for measuring the latent trait is called the test information. Test information is computed as a summation of the information that is provided by all the items in the test. Output 72.4.4 includes the test information curve for this example.

Item and test information are very useful for item selection. One important purpose of item selection is to maximize the test information across the continuum of latent construct of interest.

During the item selection process, ideally you want to select highly discriminating items whose threshold parameters cover the range of latent construct of interest. However, in practice you often encounter situations in which these highly discriminating items cannot provide enough information for a specific range of latent construct of interest, especially when these items are binary. In these situations, you might need to select some less discriminating items that can add information to the area that is not covered by these highly discriminating items.

For this example, the slope parameters range from 0.46 to 2.49, and the threshold parameters range from –2.1 to 3.2. Among these 14 items, three of them (item1, item3, and item6) have slope values less than 1. The slope value for item1 is less than 0.5, which is especially low. The item information curves suggest that these three items provide much less information than the other items. As a result, you might consider dropping these three items to economize future test administration. Output 72.4.5 shows the test information curves for the original test, which has 14 items, and the shorter test, which excludes item1, item3, and item6. The two information curves are almost identical, suggesting that the shorter test provides almost the same amount of information as the longer test. Because the shorter test is more economical, it is preferred for future testing.

Output 72.4.5 Test Information Curves

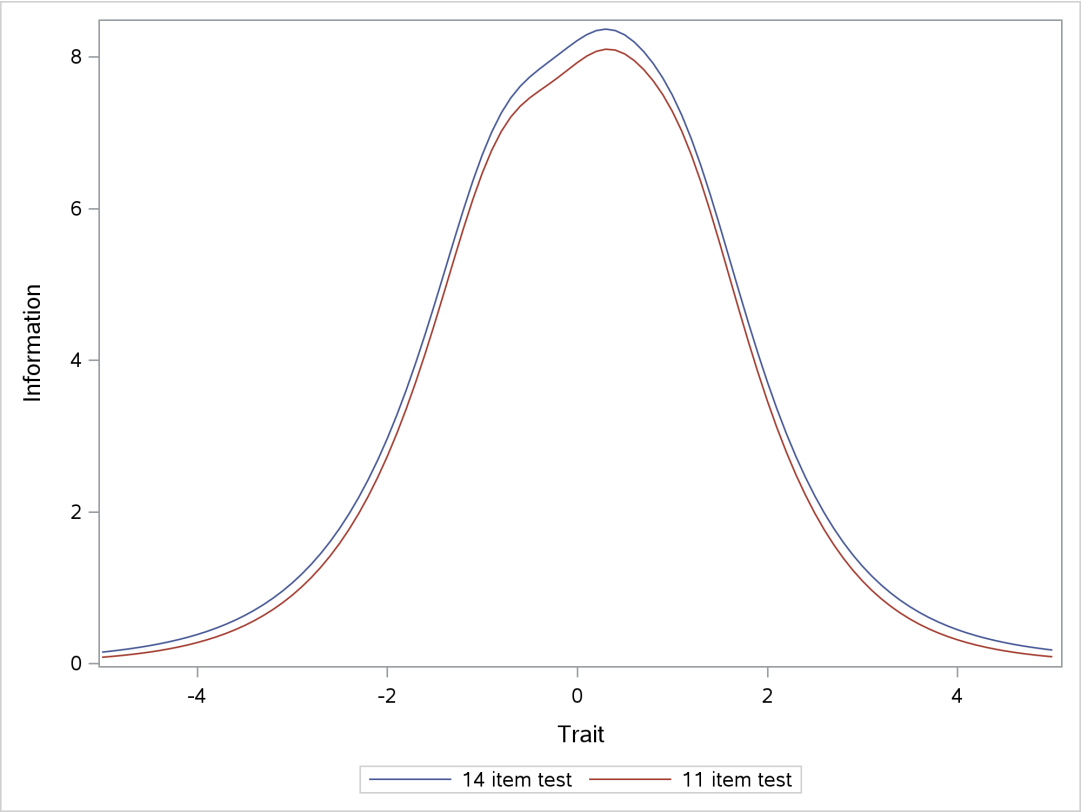

Example 72.5: Subject Scoring

(View the complete code for this example (irtex5.sas) in the example repository.)

This example also uses the American Quality of Life Survey data introduced in Example 72.1. The purpose of this example is to show you how to score subjects. You can score subjects in two different ways by using the IRT procedure. If you want to fit the model and score the subjects simultaneously, you simply use the OUT= option in the PROC IRT statement. The following statements fit an IRT model to the data set IrtUni and then score all the subjects in that data set based on the parameter estimates. Factor scores along with the original data in IrtUni are saved to a SAS data set called IrtScore. The first five observations in the OUT= data set are displayed in Output 72.5.1.

```
proc irt data=IrtUni out=IrtScore;
   var item1-item8;
run;

proc print data=IrtScore(obs=5);
run;
```

Output 72.5.1 Factor Scores for the First Five Observations

Obs	item1	item2	item3	item4	item5	item6	item7	item8	_Factor1
1	1	0	0	0	1	1	2	1	-0.38037
2	1	1	1	1	1	3	3	3	1.69157
3	0	1	0	0	1	1	1	1	-0.72592
4	1	0	0	1	0	1	2	3	0.12299
5	0	0	0	0	0	1	1	1	-1.59046

In applied research, it is not uncommon to want to save the parameter estimates (or item calibration result) and use them later to score new subjects without refitting the model. To accomplish this task, first you use the OUTMODEL= option in the PROC IRT statement to save the parameter estimates as follows:

```
proc irt data=CalData outmodel=IrtModel;
   var item1-item8;
run;
```

In the preceding statements, the data set CalData contains 500 random samples from the original data set IrtUni.

Then you use the following statements to score new subjects in the data set that is specified in the DATA= option, where the data set NewSub contains 50 random samples from the IrtUni data set. In the following statements, you use the INMODEL= option in the PROC IRT statement to input the model specification and parameter estimates from a previous analysis. To score the subjects without refitting the model, you also need to specify the SCORE suboption. Output 72.5.2 contains the first five observations in the OUT= data set.

```
proc irt data=NewSub inmodel(score)=IrtModel out=IrtScore2;
run;

proc print data=IrtScore2 (obs=5);
run;
```

Output 72.5.2 Factor Scores for the First Five Observations

Obs	Replicate	item1	item2	item3	item4	item5	item6	item7	item8	_Factor1
1	1	1	1	1	1	1	2	2	3	1.14795
2	1	1	0	1	1	1	1	3	1	0.45942
3	1	1	1	1	1	1	2	3	2	1.11387
4	1	1	1	1	1	1	1	1	2	0.56330
5	1	0	0	0	0	1	2	2	3	-0.20237

If you do not specify the SCORE suboption, PROC IRT uses the parameter values specified in the INMODEL= option as initial values, refits the model, and then uses the new parameter estimates to score the subjects.

References

Baker, F. B., and Kim, S.-H. (2004). *Item Response Theory: Parameter Estimation Techniques*. 2nd ed. New York: CRC Press.

Bock, R. D. (1972). "Estimating Item Parameters and Latent Ability When Responses Are Scored in Two or More Nominal Categories." *Psychometrika* 37:29–51.

Bock, R. D., and Aitkin, M. (1981). "Marginal Maximum Likelihood Estimation of Item Parameters: Application of an EM Algorithm." *Psychometrika* 46:443–459.

Bock, R. D., and Lieberman, M. (1970). "Fitting a Response Model for *n* Dichotomously Scored Items." *Psychometrika* 35:179–197.

De Ayala, R. J. (2009). *The Theory and Practice of Item Response Theory*. New York: Guilford Press.

Embretson, S. E., and Reise, S. P. (2000). *Item Response Theory for Psychologists*. Mahwah, NJ: Lawrence Erlbaum Associates.

Lesaffre, E., and Spiessens, B. (2001). "On the Effect of the Number of Quadrature Points in a Logistic Random Effects Model: An Example." *Journal of the Royal Statistical Society, Series C* 50:325–335.

Masters, G. N. (1982). "A Rasch Model for Partial Credit Scoring." *Psychometrika* 47:149–174.

McKinley, R. L., and Mills, C. N. (1985). "A Comparison of Several Goodness-of-Fit Statistics." *Applied Psychological Measurement* 9:49–57.

Muraki, E. (1992). "A Generalized Partial Credit Model: Application of an EM Algorithm." *Applied Psychological Measurement* 16:159–176.

Rabe-Hesketh, S., Skrondal, A., and Pickles, A. (2002). "Reliable Estimation of Generalized Linear Mixed Models Using Adaptive Quadrature." *Stata Journal* 2:1–21.

Yen, W. M. (1981). "Using Simulation Results to Choose a Latent Trait Model." *Applied Psychological Measurement* 5:245–262.

Chapter 73
The KDE Procedure

Contents

Overview: KDE Procedure **5502**
Getting Started: KDE Procedure **5502**
Syntax: KDE Procedure **5505**
- PROC KDE Statement 5505
- BIVAR Statement 5506
- BY Statement 5510
- FREQ Statement 5510
- SCORE Statement 5511
- UNIVAR Statement 5513
- WEIGHT Statement 5517

Details: KDE Procedure **5517**
- Computational Overview 5517
- Kernel Density Estimates 5517
- Binning 5519
- Kernel Distribution Estimates 5520
- Convolutions 5523
- Fast Fourier Transform 5524
- Bandwidth Selection 5525
- ODS Table Names 5527
- ODS Graphics 5527

Examples: KDE Procedure **5529**
- Example 73.1: Computing a Basic Kernel Density Estimate 5529
- Example 73.2: Changing the Bandwidth 5531
- Example 73.3: Changing the Bandwidth (Bivariate) 5533
- Example 73.4: Requesting Additional Output Tables 5535
- Example 73.5: Univariate KDE Graphics 5538
- Example 73.6: Bivariate KDE Graphics 5541

References **5545**

Overview: KDE Procedure

The KDE procedure performs univariate and bivariate kernel density estimation. Statistical *density estimation* involves approximating a hypothesized probability density function from observed data. *Kernel density estimation* is a nonparametric technique for density estimation in which a known density function (the *kernel*) is averaged across the observed data points to create a smooth approximation. PROC KDE uses a Gaussian density as the kernel, and its assumed variance determines the smoothness of the resulting estimate. See Silverman (1986) for a thorough review and discussion.

You can use PROC KDE to compute a variety of common statistics, including estimates of the percentiles of the hypothesized probability density function. You can produce a variety of plots, including univariate and bivariate histograms, plots of the kernel density estimates, and contour plots. You can also save kernel density estimates into SAS data sets.

Getting Started: KDE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/kdegs.sas.)

The following example illustrates the basic features of PROC KDE. Assume that 1000 observations are simulated from a bivariate normal density with means (0, 0), variances (10, 10), and covariance 9. The SAS DATA step to accomplish this is as follows:

```
data bivnormal;
   seed = 1283470;
   do i = 1 to 1000;
      z1 = rannor(seed);
      z2 = rannor(seed);
      z3 = rannor(seed);
      x = 3*z1+z2;
      y = 3*z1+z3;
      output;
   end;
   drop seed;
run;
```

The following statements request a bivariate kernel density estimate for the variables x and y, with contour and surface plots:

```
ods graphics on;
proc kde data=bivnormal;
   bivar x y / plots=(contour surface);
run;
ods graphics off;
```

The contour plot and the surface plot of the estimate are displayed in Figure 73.1 and Figure 73.2, respectively. Note that the correlation of 0.9 in the original data results in oval-shaped contours. These graphs are produced by specifying the PLOTS= option in the BIVAR statement with ODS Graphics enabled. For general

information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the KDE procedure, see the section "ODS Graphics" on page 5527.

Figure 73.1 Contour Plot of Estimated Density

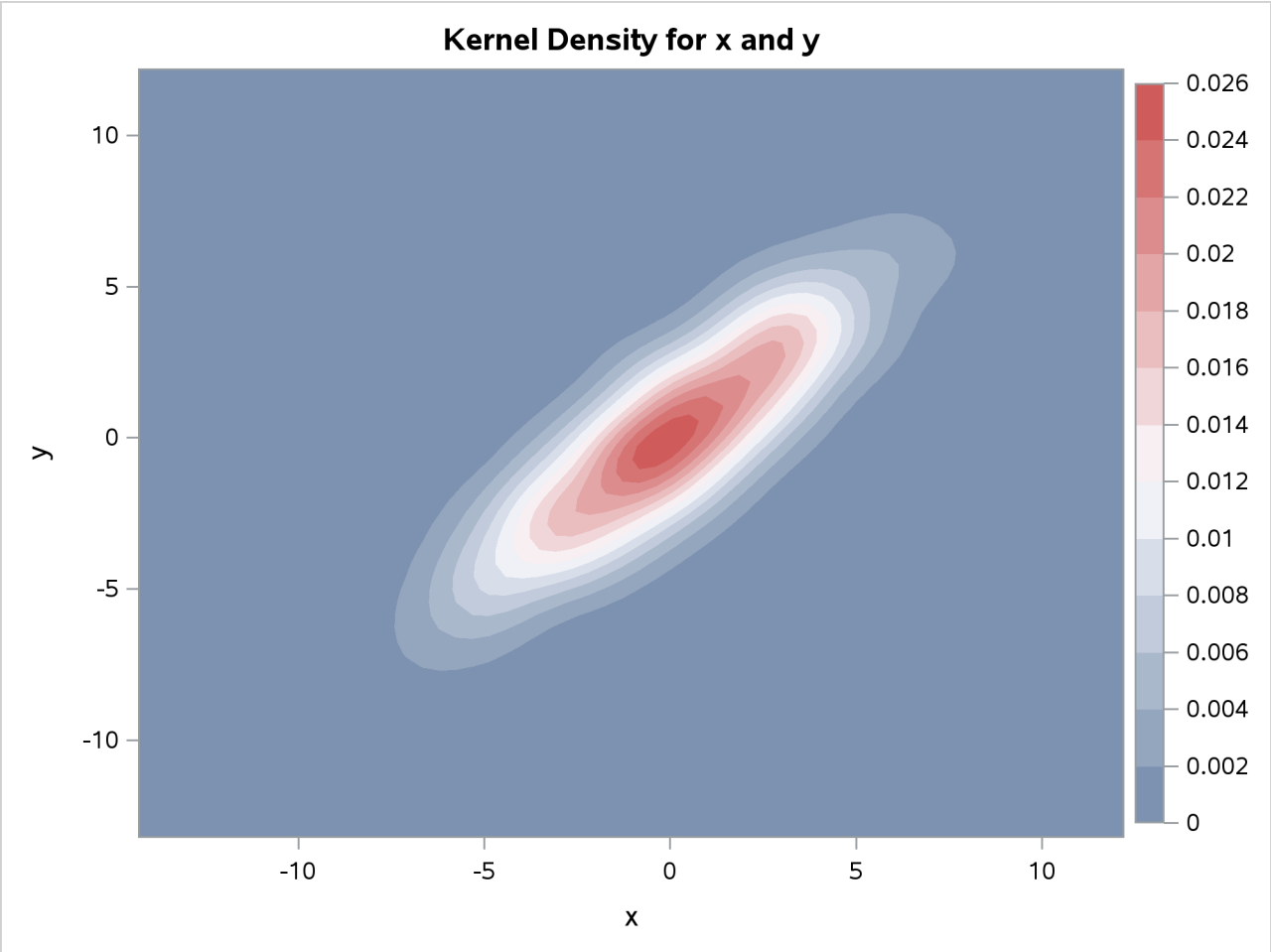

Figure 73.2 Surface Plot of Estimated Density

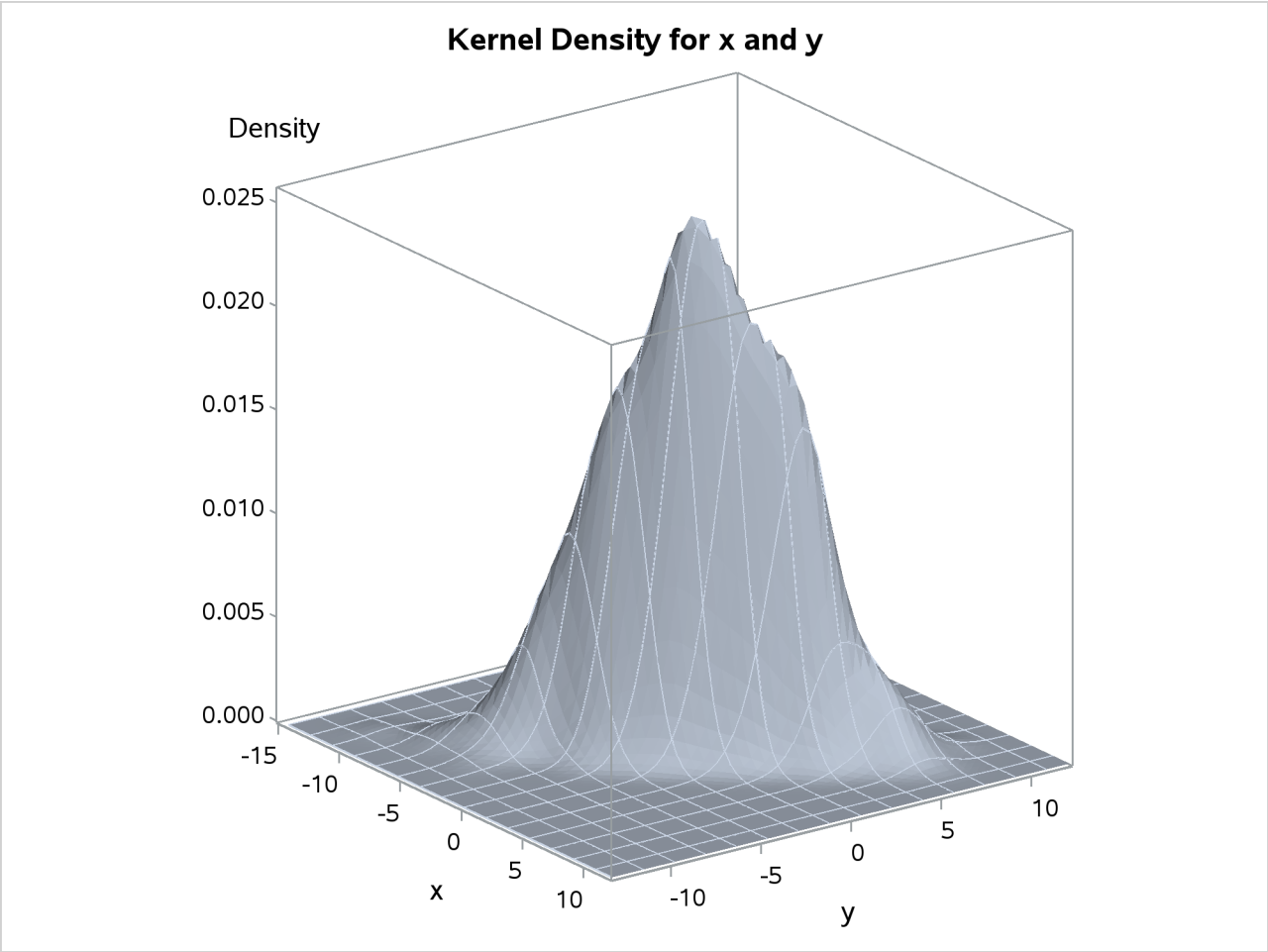

The default output tables for this analysis are shown in Figure 73.3.

Figure 73.3 Default Bivariate Tables

The KDE Procedure

Inputs	
Data Set	WORK.BIVNORMAL
Number of Observations Used	1000
Variable 1	x
Variable 2	y
Bandwidth Method	Simple Normal Reference

Controls		
	x	**y**
Grid Points	60	60
Lower Grid Limit	-14.27	-13.18
Upper Grid Limit	12.17	12.165
Bandwidth Multiplier	1	1

The "Inputs" table lists basic information about the density fit, including the input data set, the number of observations, and the variables. The bandwidth method is the technique used to select the amount of smoothing in the estimate. A simple normal reference rule is used for bivariate smoothing.

The "Controls" table lists the primary numbers controlling the kernel density fit. Here a 60×60 grid is fit to the entire range of the data, and no adjustment is made to the default bandwidth.

Syntax: KDE Procedure

The following statements are available in the KDE procedure:

PROC KDE < *options* > **;**
  BIVAR *variable-list* < / *options* > **;**
  UNIVAR *variable-list* < / *options* > **;**
  SCORE *options* **;**
  BY *variables* **;**
  FREQ *variable* **;**
  WEIGHT *variable* **;**

The PROC KDE statement invokes the procedure. The BIVAR statement computes one or more bivariate kernel density estimates on a uniformly spaced grid. The UNIVAR statement computes one or more univariate kernel density estimates on a uniformly spaced grid. The SCORE statement produces kernel density estimates at arbitrary locations for an associated BIVAR or UNIVAR statement. You can specify any number of BIVAR, UNIVAR, and SCORE statements.

The following sections describe the PROC KDE statement and then describe the other statements in alphabetical order.

PROC KDE Statement

PROC KDE < **DATA=***SAS-data-set* > **;**

The PROC KDE statement invokes the procedure and optionally specifies the input data set.

You can specify the following option:

DATA=*SAS-data-set*
  specifies the input SAS data set to be used by PROC KDE. The default is the most recently created data set.

BIVAR Statement

BIVAR *v1* <(*v-options*)> *v2* <(*v-options*)> . . . < *vN* <(*v-options*)>> < / *options* > ;

BIVAR (*v1 v2*) <(*v3 v4*) . . . (*vN–1 vN*)> < / *options* > ;

The BIVAR statement computes bivariate kernel density estimates for the specified variables. The *v-options* optionally specified in parentheses after a variable name apply only to that variable, and they override corresponding global *options* that are specified following a slash (/).

You must specify at least two variables, *v1* and *v2*. If you specify more than two variables, PROC KDE computes a bivariate kernel density estimate for each distinct pair of variables in the list. For example, if you specify the following statement, then a bivariate kernel density estimate is computed for each of the variable pairs (x, y), (x, z), and (y, z):

```
bivar x y z;
```

Alternatively, you can specify an explicit list of variable pairs, with each pair enclosed in parentheses. This requests a bivariate kernel density for each pair of variables. For example, if you specify the following statement, then bivariate kernel density estimates are computed for (x, y) and (y, z).

```
bivar (x y) (y z);
```

Table 73.1 summarizes the *options* available in the BIVAR statement.

Table 73.1 BIVAR Statement Options

Option	Description
BIVSTATS	Produces a table for each density estimate
BW=	Specifies the bandwidth
BWM=	Specifies the bandwidth multiplier
CDF	Produces the distribution function
GRIDL=	Specifies the lower grid limit
GRIDU=	Specifies the upper grid limit
LEVELS	Produces a table of levels for contours of the bivariate density
NGRID=	Specifies the number of grid points associated with each variable
NOPRINT	Suppresses output tables
OUT=	Specifies the name of the output data set
PERCENTILES	Produces a table of percentiles
PLOTS=	Requests one or more plots
TRUNCATE	Restricts the lower and upper grid limits to the minimum and maximum observed values, respectively, for each variable
UNISTATS	Produces, for each density estimate, a table that contains standard univariate statistics and the bandwidths

You can specify the following *options* in the BIVAR statement. Some *options* can be used as *v-options*, as indicated in the description of the *option*.

BIVSTATS

produces, for each density estimate, a table that contains the covariance and correlation between the two variables.

BW=*number*

specifies the bandwidth to apply to each variable in each kernel density estimate. Larger bandwidths produce a smoother estimate, whereas smaller bandwidths produce a rougher estimate. To specify different bandwidths for different variables, specify BW=*number* as a *v-option*. By default, the bandwidth is set automatically by the simple normal reference method (see the section "Bandwidth Selection" on page 5525).

BWM=*number*

specifies the bandwidth multiplier to apply to the corresponding bandwidth for each variable. Values of *number* greater than 1 increase the effective bandwidth and produce a smoother estimate. Values less than 1 decrease the effective bandwidth and produce a rougher estimate. To specify different bandwidth multipliers for different variables, specify BWM=*number* as a *v-option*. By default, BWM=1.

CDF

computes the distribution function in addition to the density function for each pair of variables. The distribution function is obtained by a seminumerical technique as described in the section "Kernel Distribution Estimates" on page 5520.

GRIDL=*number*

specifies the lower grid limit to apply to each variable in each kernel density estimate. To specify different lower grid limits for different variables, specify GRIDL=*number* as a *v-option*. The default value for a particular variable is a function of both the kernel bandwidth and the minimum observed value for that variable.

GRIDU=*number*

specifies the upper grid limit to apply to each variable in each kernel density estimate. To specify different upper grid limits for different variables, specify GRIDU=*number* as a *v-option*. The default value for a particular variable is a function of both the kernel bandwidth and the maximum observed value for that variable.

LEVELS

LEVELS=(*numlist***)**

computes a table of levels (called "Levels") for contours of the bivariate density. The number of contours is equal to the number of values in *numlist*, where each value in *numlist* specifies a percentage to be used in calculating the density volume that is enclosed by the contour. The contours are defined such that the density has a constant level along each contour, and the volume enclosed by each contour corresponds to the total density volume minus the specified percentage of the total volume. In other words, the contours correspond to slices or levels of the density surface that are taken along the density axis. The "Levels" table also provides the minimum and maximum values for each contour along the directions of the two data variables. By default, LEVELS=(1, 5, 10, 50, 90, 95, 99, 100).

NGRID=*number*
NG=*number*

specifies the number of grid points to be associated with each variable in each kernel density estimate. To specify different numbers of grid points for different variables, specify NGRID=*number* as a *v-option*. By default, NGRID=60.

NOPRINT

suppresses output tables. You can use this option when you want to produce only graphical output.

OUT=*SAS-data-set*

names the output data set in which to save kernel density estimates. This output data set contains the following variables:

- var1, whose value is the name of the first variable in a bivariate kernel density estimate
- var2, whose value is the name of the second variable in a bivariate kernel density estimate
- value1, whose value corresponds to grid coordinates for the first variable
- value2, whose value corresponds to grid coordinates for the second variable
- density, whose values are equal to kernel density estimate at the associated grid point
- count, whose values represent the number of original observations contained in the bin that corresponds to a grid point
- distribution, whose values are equal to the distribution estimate at the associated grid point (this variable is included only when the CDF global option is specified)

PERCENTILES
PERCENTILES=*numlist*

produces a table of percentiles for each BIVAR variable. You can specify a list of percentiles to be computed in *numlist*. The default percentiles are 0.5, 1, 2.5, 5, 10, 25, 50, 75, 90, 95, 97.5, 99, and 99.5.

PLOTS=(*plot-request***<(***options***)> <...***plot-request***<(***options***)>>)**

specifies which plots of the bivariate data and kernel density estimate to produce. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc kde data=octane;
   bivar Rater Customer / plots=all;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

By default, if ODS Graphics is enabled and you do not specify the PLOTS= option, then the BIVAR statement creates a contour plot. If you specify the PLOTS= option, only the requested plots are created.

You can specify the following *plot-requests*:

ALL
produces all bivariate plots.

CONTOUR
produces a contour plot of the bivariate density estimate.

CONTOURSCATTER
produces a contour plot of the bivariate density estimate overlaid with a scatter plot of the data.

HISTOGRAM < (*view-options*) >
produces a bivariate histogram of the data. You can specify one or both of the following *view-options* within parentheses:

ROTATE=*angle*
rotates the histogram *angle* degrees, where –180 < *angle* < 180. By default, ROTATE=54.

TILT=*angle*
tilts the histogram *angle* degrees, where –180 < *angle* < 180. By default, TILT=20.

HISTSURFACE < (*view-options*) >
produces a bivariate histogram of the data overlaid with a surface plot of the bivariate kernel density estimate. You can specify one or both of the following *view-options* within parentheses:

ROTATE=*angle*
rotates the histogram and kernel density surface *angle* degrees, where –180 < *angle* < 180. By default, ROTATE=54.

TILT=*angle*
tilts the histogram and kernel density surface *angle* degrees, where –180 < *angle* < 180. By default, TILT=20.

NONE
suppresses all plots, including the contour plot that is produced by default when ODS Graphics is enabled and the PLOTS= option is not specified.

SCATTER
produces a scatter plot of the data.

SURFACE < (*view-options*) >
produces a surface plot of the bivariate kernel density estimate. You can specify one or both of the following *view-options* within parentheses:

ROTATE=*angle*
rotates the kernel density surface *angle* degrees, where –180 < *angle* < 180. By default, ROTATE=54.

TILT=*angle*
tilts the kernel density surface *angle* degrees, where –180 < *angle* < 180. By default, TILT=20.

TRUNCATE

sets the lower grid limit for each variable to the minimum observed for that variable, and sets the upper grid limit for each variable to the maximum observed value for that variable.

NOTE: The GRIDL and GRIDU options take precedence over the TRUNCATE option. If one or both are specified, the corresponding lower and upper grid limits are set accordingly.

UNISTATS

produces for each density estimate a table that contains standard univariate statistics for each of the variable pairs and the bandwidths that are used to compute the kernel density estimate. The statistics indexed in the table are the mean, variance, standard deviation, range, and interquartile range.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC KDE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the KDE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

The FREQ statement specifies a variable that provides frequencies for each observation in the DATA= data set. Specifically, if n is the value of the FREQ variable for a given observation, then that observation is used n times. If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

SCORE Statement

SCORE DATA=*SAS-data-set* **OUT**=*SAS-data-set* < / *option* > **;**

The SCORE statement produces kernel density estimates at arbitrary locations for an associated UNIVAR or BIVAR statement.

You must specify the following arguments:

DATA=*SAS-data-set*
specifies the input SAS data set.

For a univariate density, the input data set contains the variable v1, where v1 is a variable that appears in a UNIVAR statement. The values of v1 indicate arbitrary points for which a density estimate is requested.

For a bivariate density, the input data set contains variables v1 and v2, where v1 and v2 are variables that appear in a BIVAR statement. Pairs (v1, v2) indicate arbitrary points for which a density estimate is requested.

When you specify a BY statement, the DATA= data set must not contain any of the BY variables. The entire data set is scored for each BY group.

OUT=*SAS-data-set*
names the output SAS data set to be produced by the SCORE statement.

For a univariate density, the output data set contains the following variables:

- var, whose value is the name of the variable in the DATA= data set
- value, whose values are taken from the DATA= data set
- density, whose values are equal to the kernel density estimate
- distribution, whose values are equal to the distribution estimate (this variable is included only when the CDF global option is specified in the corresponding UNIVAR statement)

For a bivariate density, the output data set has the following variables:

- variable1, whose value is the name of the first variable in the DATA= data set
- value1, whose values are taken from the DATA= data set
- variable2, whose value is the name of the second variable in the DATA= data set
- value2, whose values are taken from the DATA= data set
- density, whose values are equal to the kernel density estimate
- distribution, whose values are equal to the distribution estimate (this variable is included only when the CDF global option is specified in the corresponding BIVAR statement)

You can provide the following *option* in the SCORE statement.

METHOD=INTERP | EXACT

requests a particular scoring method. You can specify the following values:

EXACT corresponds to the evaluation of the appropriate binned density equation from the section "Binning" on page 5519 and the analytical distribution equation from the section "Kernel Distribution Estimates" on page 5520.

INTERP corresponds to linear interpolation of the density estimate and seminumerical distribution estimation. The seminumerical distribution technique is described in "Kernel Distribution Estimates" on page 5520. **NOTE:** Attempting to score outside the grid results in 0 for the density estimate.

By default, METHOD=INTERP.

You can include multiple SCORE statements. Each SCORE statement applies to the first UNIVAR or BIVAR statement that specifies the same variables as are included in the DATA= data set. The order of variables matters for the bivariate case. Therefore, a SCORE statement that specifies a DATA= data set consisting of variables x and y (in that order) will match only a BIVAR statement of one of the following forms:

BIVAR (*x y***)**. . . < / *options* > **;**

BIVAR *x y* . . . < / *options* > **;**

Example

Suppose the data set MyData contains the variables x and y and the data sets MyScoreInX, MyScoreInY, and MyScoreInXY contain the variables x, y, and (x, y) respectively. The following statements request both the individual (univariate) and joint (bivariate) kernel density estimates and distributions and selectively score them:

```
proc kde data=MyData;
   univar x y;
   bivar x y / CDF;
   score data=MyScoreInX out=MarginalX;
   score data=MyScoreInY out=MarginalY;
   score data=MyScoreInXY out=JointXY;
run;
```

The first SCORE statement is associated with the UNIVAR statement and produces the MarginalX output data set. This data set contains variables x, value, and density, where density is the density function at value.

The second SCORE statement is associated with the UNIVAR statement and produces the MarginalY output data set. This data set contains variables y, value and density, where density is the density function at value.

The third SCORE statement is associated with the BIVAR statement and produces the JointXY output data set. This data set contains variables x, value1, y, value2, density, and distribution, where density and distribution are the density and distribution functions, respectively, at (value1, value2).

UNIVAR Statement

UNIVAR *variable* < (*v-options*) > < . . . *variable* < (*v-options*) > > < / *options* > ;

The UNIVAR statement computes univariate kernel density estimates. You can specify various *v-options* for each variable by enclosing them in parentheses after the variable name. You can also specify global *options* following a slash (/). Global *options* apply to all the variables specified in the UNIVAR statement. However, individual variable *v-options* override the global *options*.

Table 73.2 summarizes the *options* available in the UNIVAR statement.

Table 73.2 UNIVAR Statement Options

Option	Description
BW=	Specifies a bandwidth
BWM=	Specifies a bandwidth multiplier
CDF	Produces the distribution function
GRIDL=	Specifies a lower grid limit
GRIDU=	Specifies an upper grid limit
LEVELS	Produces a table of levels for contours of the univariate density
METHOD=	Specifies which method to use to compute the bandwidth
NGRID=	Specifies a number of grid points
NOPRINT	Suppresses output tables
OUT=	Specifies the output SAS data set to contain the kernel density estimate
PERCENTILES	Produces a table of percentiles
PLOTS=	Produces plots of the univariate kernel density estimate
SJPIMAX=	Specifies the maximum grid value in determining the Sheather-Jones plug-in bandwidth
SJPIMIN=	Specifies the minimum grid value in determining the Sheather-Jones plug-in bandwidth
SJPINUM=	Specifies the number of grid values to be used in determining the Sheather-Jones plug-in bandwidth
SJPITOL=	Specifies the tolerance for termination of the bisection algorithm
TRUNCATE	Restricts the lower and upper grid limits to the minimum and maximum observed values, respectively, for each variable
UNISTATS	Produces, for each variable, a table that contains standard univariate statistics and the bandwidth

You can specify the following *options* in the UNIVAR statement. Some *options* can be used as *v-options*, as indicated in the description of the *option*.

BW=*number*

specifies the bandwidth to apply to each variable in each kernel density estimate. Larger bandwidths produce a smoother estimate, whereas smaller bandwidths produce a rougher estimate. To specify different bandwidths for different variables, specify BW=*number* as a *v-option*. By default, the bandwidth is set automatically by the Sheather-Jones plug-in method (see the section "Bandwidth Selection" on page 5525).

BWM=*number*

specifies the bandwidth multiplier to apply to the corresponding bandwidth for each variable. Values of *number* greater than 1 increase the effective bandwidth and produce a smoother estimate. Values less than 1 decrease the effective bandwidth and produce a rougher estimate. To specify different bandwidth multipliers for different variables, specify BWM=*number* as a *v-option*. By default, BWM=1.

CDF

computes the distribution function in addition to the density function for each variable. The distribution function is obtained by a seminumerical technique as described in the section "Kernel Distribution Estimates" on page 5520.

GRIDL=*number*

specifies a lower grid limit for each kernel density estimate. To specify different lower grid limits for different variables, specify GRIDL=*number* as a *v-option*. The default value for a particular variable is a function of both the kernel bandwidth and the minimum observed value for that variable.

GRIDU=*number*

specifies an upper grid limit for each kernel density estimate. To specify different upper grid limits for different variables, specify GRIDU=*number* as a *v-option*. The default value for a particular variable is a function of both the kernel bandwidth and the maximum observed value for that variable.

LEVELS

LEVELS=(*numlist***)**

computes a table of levels (called "Levels") for contours of the univariate density. The number of contours is equal to the number of values in *numlist*, where each value in *numlist* specifies a percentage to be used in calculating the density area that is enclosed by the contour. The contours are defined such that the density has a constant level along each contour, and the area enclosed by each contour corresponds to the total density area minus the specified percentage of the total area. In other words, the contours correspond to slices or levels of the density surface that are taken along the density axis. The "Levels" table also provides the minimum and maximum values for each contour along the direction of the data variable. By default, LEVELS=(1, 5, 10, 50, 90, 95, 99, 100).

METHOD=SJPI | SNR | SNRQ | SROT | OS

specifies the method for computing the bandwidth. You can specify the following values:

SJPI	Sheather-Jones plug-in method
SNR	simple normal reference method
SNRQ	simple normal reference method using interquartile range
SROT	Silverman's rule of thumb method
OS	oversmoothed method

For a description of these methods, see the section "Bandwidth Selection" on page 5525 and Jones, Marron, and Sheather (1996). By default, METHOD=SJPI.

NOTE: The BW= option takes precedence over the METHOD= option. If both are specified, the METHOD= option is ignored.

NGRID=*number*

NG=*number*

specifies a number of grid points to use for each kernel density estimate. To specify different numbers of grid points for different variables, specify NGRID=*number* as a *v-option*. By default, NGRID=401.

NOPRINT

suppresses output tables. You can use this option when you want to produce only graphical output.

OUT=*SAS-data-set*

names the output SAS data set to contain the kernel density estimate. This output data set contains the following variables:

- var, whose value is the name of the variable in the kernel density estimate
- value, whose value corresponds to grid coordinates for the variable
- density, whose values are equal to the kernel density estimate at the associated grid point
- count, whose values indicate the number of original observations contained in the bin that corresponds to a grid point
- distribution, whose values are equal to the distribution estimate at the associated grid point (*appears only when the CDF global option is specified*)

PERCENTILES

PERCENTILES=*numlist*

produces a table of percentiles for each UNIVAR variable. You can specify a list of percentiles to be computed in *numlist*. The default percentiles are 0.5, 1, 2.5, 5, 10, 25, 50, 75, 90, 95, 97.5, 99, and 99.5.

PLOTS=(*plot-request***<(***options***)> <...***plot-request* **<(***options***)>>)**

specifies which plots of the univariate kernel density estimate to produce. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc kde data=channel;
   univar length / plots=histdensity;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *plot-requests*, each of which (except for DENSITYOVERLAY) produces a separate plot for every variable listed in the UNIVAR statement:

ALL produces all plots.

DENSITY produces the univariate kernel density estimate curve.

DENSITYOVERLAY produces the overlaid univariate kernel density estimate curves. If you specify more than one variable in the UNIVAR statement, PROC KDE overlays the density curves for all the variables on a single plot.

HISTDENSITY produces the univariate histogram of data overlaid with kernel density estimate curve.

HISTOGRAM produces the univariate histogram of data.

NONE suppresses all plots.

By default, if ODS Graphics is enabled and you do not specify the PLOTS= option, then the UNIVAR statement creates a histogram overlaid with a kernel density estimate. If you specify the PLOTS= option, only the requested plots are created.

SJPIMAX=*number*
specifies the maximum grid value in determining the Sheather-Jones plug-in bandwidth. The default value is two times the oversmoothed estimate.

SJPIMIN=*number*
specifies the minimum grid value in determining the Sheather-Jones plug-in bandwidth. The default value is the maximum value divided by 18.

SJPINUM=*number*
specifies the number of grid values to use in determining the Sheather-Jones plug-in bandwidth. By default, SJPINUM=21.

SJPITOL=*number*
specifies the tolerance for terminating the bisection algorithm that is used in computing the Sheather-Jones plug-in bandwidth. By default, SJPITOL=0.001.

TRUNCATE
sets the lower grid limit for each variable to the minimum observed for that variable, and sets the upper grid limit for each variable to the maximum observed value for that variable.

NOTE: The GRIDL and GRIDU options take precedence over the TRUNCATE option. If one or both are specified, the corresponding lower and upper grid limits are set accordingly.

UNISTATS
produces, for each variable a table that contains standard univariate statistics and the bandwidth that are used to compute its kernel density estimate. The statistics listed are the mean, variance, standard deviation, range, and interquartile range.

Examples

Suppose the data set MyData contains the variables x1, x2, x3, and x4. The following statements request a univariate kernel density estimate for each of these variables:

```
proc kde data=MyData;
   univar x1 x2 x3 x4;
run;
```

You can also specify different bandwidths and other options for each variable. For example, the following statements request kernel density estimates that use Silverman's rule of thumb (SROT) method for all variables:

```
proc kde data=MyData;
   univar x1 (bwm=2)
          x2 (bwm=0.5 ngrid=100)
          x3 x4 / ngrid=200 method=srot;
run;
```

The NGRID=200 option applies to the variables x1, x3, and x4, but the *v-option* NGRID=100 applies only to x2. Bandwidth multipliers of 2 and 0.5 are specified for the variables x1 and x2, respectively.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement specifies a variable that weights the observations in computing the kernel density estimate. Observations that have higher weights have more influence in the computations. If an observation has a nonpositive or missing weight, then the entire observation is omitted from the analysis. Be cautious in using data sets that have extreme weights, because they can produce unreliable results.

Details: KDE Procedure

Computational Overview

The two main computational tasks of PROC KDE are automatic bandwidth selection and the construction of a kernel density estimate after a bandwidth has been selected. An optional task is the construction of the kernel distribution estimate. The following sections discuss density and distribution estimation, kernel bandwidth selection, and efficient computation techniques.

Kernel Density Estimates

A weighted univariate kernel density estimate involves a variable X and a weight variable W. Let $(X_i, W_i),\ i = 1, 2, \ldots, n$, denote a sample of X and W of size n. The weighted kernel density estimate of $f(x)$, the density of X, is as follows:

$$\hat{f}(x) = \frac{1}{\sum_{i=1}^{n} W_i} \sum_{i=1}^{n} W_i \varphi_h(x - X_i)$$

where h is the bandwidth and

$$\varphi_h(x) = \frac{1}{\sqrt{2\pi}h} \exp\left(-\frac{x^2}{2h^2}\right)$$

is the standard normal density rescaled by the bandwidth. If $h \to 0$ and $nh \to \infty$, then the optimal bandwidth is

$$h_{\mathrm{AMISE}} = \left[\frac{1}{2\sqrt{\pi}n \int (f'')^2}\right]^{1/5}$$

This optimal value is unknown, and so approximations methods are required. For a derivation and discussion of these results, see Silverman (1986, Chapter 3) and Jones, Marron, and Sheather (1996).

For the bivariate case, let $\mathbf{X} = (X, Y)$ be a bivariate random element taking values in R^2 with joint density function

$$f(x, y),\ (x, y) \in R^2$$

and let $\mathbf{X}_i = (X_i, Y_i),\ i = 1, 2, \ldots, n$, be a sample of size n drawn from this distribution. The kernel density estimate of $f(x, y)$ based on this sample is

$$\begin{aligned}\hat{f}(x, y) &= \frac{1}{n}\sum_{i=1}^{n}\varphi_{\mathbf{h}}(x - X_i, y - Y_i) \\ &= \frac{1}{nh_Xh_Y}\sum_{i=1}^{n}\varphi\left(\frac{x - X_i}{h_X}, \frac{y - Y_i}{h_Y}\right)\end{aligned}$$

where $(x, y) \in R^2$, $h_X > 0$ and $h_Y > 0$ are the bandwidths, and $\varphi_{\mathbf{h}}(x, y)$ is the rescaled normal density

$$\varphi_{\mathbf{h}}(x, y) = \frac{1}{h_Xh_Y}\varphi\left(\frac{x}{h_X}, \frac{y}{h_Y}\right)$$

where $\varphi(x, y)$ is the standard normal density function

$$\varphi(x, y) = \frac{1}{2\pi}\exp\left(-\frac{x^2 + y^2}{2}\right)$$

Under mild regularity assumptions about $f(x, y)$, the mean integrated squared error (MISE) of $\hat{f}(x, y)$ is

$$\begin{aligned}\mathrm{MISE}(h_X, h_Y) &= \mathrm{E}\int(\hat{f} - f)^2 \\ &= \frac{1}{4\pi nh_Xh_Y} + \frac{h_X^4}{4}\int\left(\frac{\partial^2 f}{\partial X^2}\right)^2 dx\,dy \\ &\quad + \frac{h_Y^4}{4}\int\left(\frac{\partial^2 f}{\partial Y^2}\right)^2 dx\,dy + O\left(h_X^4 + h_Y^4 + \frac{1}{nh_Xh_Y}\right)\end{aligned}$$

as $h_X \to 0$, $h_Y \to 0$ and $nh_Xh_Y \to \infty$.

Now set

$$\begin{aligned}\mathrm{AMISE}(h_X, h_Y) &= \frac{1}{4\pi nh_Xh_Y} + \frac{h_X^4}{4}\int\left(\frac{\partial^2 f}{\partial X^2}\right)^2 dx\,dy \\ &\quad + \frac{h_Y^4}{4}\int\left(\frac{\partial^2 f}{\partial Y^2}\right)^2 dx\,dy\end{aligned}$$

which is the asymptotic mean integrated squared error (AMISE). For fixed n, this has a minimum at $(h_{\mathrm{AMISE_}X}, h_{\mathrm{AMISE_}Y})$ defined as

$$h_{\mathrm{AMISE_}X} = \left[\frac{\int(\frac{\partial^2 f}{\partial X^2})^2}{4n\pi}\right]^{1/6}\left[\frac{\int(\frac{\partial^2 f}{\partial X^2})^2}{\int(\frac{\partial^2 f}{\partial Y^2})^2}\right]^{2/3}$$

and

$$h_{\text{AMISE_}Y} = \left[\frac{\int(\frac{\partial^2 f}{\partial Y^2})^2}{4n\pi}\right]^{1/6} \left[\frac{\int(\frac{\partial^2 f}{\partial Y^2})^2}{\int(\frac{\partial^2 f}{\partial X^2})^2}\right]^{2/3}$$

These are the optimal asymptotic bandwidths in the sense that they minimize MISE. However, as in the univariate case, these expressions contain the second derivatives of the unknown density f being estimated, and so approximations are required. See Wand and Jones (1993) for further details.

Binning

Binning, or assigning data to discrete categories, is an effective and fast method for large data sets (Fan and Marron 1994). When the sample size n is large, direct evaluation of the kernel estimate $\hat{f}$ at any point would involve n kernel evaluations, as shown in the preceding formulas. To evaluate the estimate at each point of a grid of size g would thus require ng kernel evaluations. When you use $g = 401$ in the univariate case or $g = 60 \times 60 = 3600$ in the bivariate case and $n \geq 1000$, the amount of computation can be prohibitively large. With binning, however, the computational order is reduced to g, resulting in a much quicker algorithm that is nearly as accurate as direct evaluation.

To bin a set of weighted univariate data $X_1, X_2, \ldots, X_n$ to a grid $x_1, x_2, \ldots, x_g$, simply assign each sample X_i, together with its weight W_i, to the nearest grid point x_j (also called the bin center). When binning is completed, each grid point x_i has an associated number c_i, which is the sum total of all the weights that correspond to sample points that have been assigned to x_i. These c_is are known as the *bin counts.*

This procedure replaces the data $(X_i, W_i),\ i = 1, 2, \ldots, n$, with the smaller set $(x_i, c_i),\ i = 1, 2, \ldots, g$, and the estimation is carried out with these new data. This is so-called *simple binning,* versus the finer *linear binning* described in Wand (1994). PROC KDE uses simple binning for the sake of faster and easier implementation. Also, it is assumed that the bin centers $x_1, x_2, \ldots, x_g$ are equally spaced and in increasing order. In addition, assume for notational convenience that $\sum_{i=1}^{n} W_i = n$ and, therefore, $\sum_{i=1}^{g} c_i = n$.

If you replace the data $(X_i, W_i), i = 1, 2, \ldots, n$, with $(x_i, c_i), i = 1, 2, \ldots, g$, the weighted estimator $\hat{f}$ then becomes

$$\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{g} c_i \varphi_h(x - x_i)$$

with the same notation as used previously. To evaluate this estimator at the g points of the same grid vector $grid = (x_1, x_2, \ldots, x_g)'$ is to calculate

$$\hat{f}(x_i) = \frac{1}{n}\sum_{j=1}^{g} c_j \varphi_h(x_i - x_j)$$

for $i = 1, 2, \ldots, g$. This can be rewritten as

$$\hat{f}(x_i) = \frac{1}{n}\sum_{j=1}^{g} c_j \varphi_h(|i - j|\delta)$$

where $\delta = x_2 - x_1$ is the increment of the grid.

The same idea of binning works similarly with bivariate data, where you estimate $\hat{f}$ over the grid matrix $grid = grid_X \times grid_Y$ as follows:

$$grid = \begin{bmatrix} (x_1, y_1) & (x_1, y_2) & \dots & (x_1, y_{g_Y}) \\ (x_2, y_1) & (x_2, y_2) & \dots & (x_2, y_{g_Y}) \\ \vdots & & & \\ (x_{g_X}, y_1) & (x_{g_X}, y_2) & \dots & (x_{g_X}, y_{g_Y}) \end{bmatrix}$$

The density estimates are then

$$\hat{f}(x_i, y_j) = \frac{1}{n} \sum_{k=1}^{g_X} \sum_{l=1}^{g_Y} c_{k,l} \varphi_{\mathbf{h}}(|i-k|\delta_X, |j-l|\delta_Y)$$

where $\delta_X = x_2 - x_1$ and $\delta_Y = y_2 - y_1$ are the increments of the grid.

Kernel Distribution Estimates

The distribution function can be obtained by integrating the kernel density estimate (Azzalini 1981). PROC KDE provides both an analytical and a seminumerical integration approach, each of which involves the closed form solution to the integral of the binned density estimator from the previous section.

For the univariate case, the distribution function is

$$\begin{aligned} \hat{F}(x) &= \int_{-\infty}^{x} \hat{f}(u)du \\ &= \frac{1}{N} \sum_{i=1}^{g} c_i \Phi_h(x - x_i) \end{aligned}$$

where h is the univariate kernel bandwidth and

$$\Phi_h(x) = \frac{1}{\sqrt{2\pi}h} \int_{-\infty}^{x} \exp\left(-\frac{u^2}{2h^2}\right) du$$

For the bivariate case, the distribution function is

$$\begin{aligned} \hat{F}(x, y) &= \int_{-\infty}^{x} \int_{-\infty}^{y} \hat{f}(u, v)dudv \\ &= \frac{1}{N} \sum_{j=1}^{g_Y} \Phi_{h_Y}(y - y_j) \sum_{i=1}^{g_X} c_{i,j} \Phi_{h_X}(x - x_i) \end{aligned}$$

where h_X and h_Y are the bivariate kernel bandwidths for variables x and y.

The analytical integration approach is simply the direct evaluation of the appropriate distribution function equation. The seminumerical integration approach is a mixture of direct evaluation of the distribution function equation and numerical integration via the extended trapezoidal rule (Press et al. 1992). This mixture depends on whether the upper integration limits fall inside or outside the binning grid. In general, there are three cases:

- Integration limits precede leading grid edges.
- Integration limits fall within grid.
- Integration limits follow trailing grid edges.

When the integration limits lead or trail the grid edges, the seminumerical approach is identical to the analytical approach. Otherwise, the seminumerical approach splits the overall integral into integrals from minus infinity to the lower grid edges, and an integral from the leading grid edges to the upper integration limits. For the univariate case, this split becomes

$$\begin{aligned}\hat{F}(x) &= \int_{-\infty}^{x_1} \hat{f}(u)du + \int_{x_1}^{x} \hat{f}(u)du \\ &= \hat{F}(x_1) + \int_{x_1}^{x} \hat{f}(u)du\end{aligned}$$

The term $\hat{F}(x_1)$ is simply the analytical distribution function up to the leading grid edge. The second term is evaluated numerically:

- If x coincides with a grid element x_k, the overall integral is

  $$\hat{F}(x) \approx \hat{F}(x_1) + \tilde{F}(x_k)$$

  where

  $$\tilde{F}(x_k) = \left[\sum_{m=1}^{k} \hat{f}(x_m) - \frac{1}{2}(\hat{f}(x_1) + \hat{f}(x_k))\right]\delta$$

- If x does not coincide with a grid element, then the numerical integral is approximated by linear interpolation that uses the nearest grid elements p and $p+1$:

  $$\tilde{F}(x) = \tilde{F}(x_p) + \frac{(\tilde{F}(x_{p+1}) - \tilde{F}(x_p))}{x_{p+1} - x_p}(x - x_p).$$

The bivariate case is similar to the univariate case, although there are multiple analytical terms to evaluate due to the nature of the integral:

$$\hat{F}(x,y) = \hat{F}(x,y_1) + \hat{F}(x_1,y) - \hat{F}(x_1,y_1) + \int_{x_1}^{x}\int_{y_1}^{y} \hat{f}(u,v)dudv$$

As with the univariate case, the fourth term is evaluated numerically:

- If (x, y) coincides with a grid element, then the overall integral is

  $$\hat{F}(x_k,y_\ell) \approx \hat{F}(x,y_1) + \hat{F}(x_1,y) - \hat{F}(x_1,y_1) + \tilde{F}(x_k,y_\ell)$$

  where $\tilde{F}(x_k, y_\ell)$ is recursively computed via

  $$\begin{aligned}
  \tilde{F}(x_k) &= \left[\sum_{n=1}^{\ell} \hat{f}(x_k,y_n) - \frac{1}{2}(\hat{f}(x_k,y_1) + \hat{f}(x_k,y_\ell))\right]\delta_Y \\
  \tilde{F}(x_k,y_\ell) &\approx \left[\sum_{m=1}^{k} \hat{F}(x_m) - \frac{1}{2}(\hat{F}(x_1) + \hat{F}(x_k))\right]\delta_X
  \end{aligned}$$

- If (x, y) does not coincide with a grid element, then the numerical integral is approximated by bilinear interpolation that uses the nearest grid elements $p, p+1, q,$ and $q+1$:

  $$\begin{aligned}
  \tilde{F}(y_q) &= \tilde{F}(x_p,y_q) + \frac{(\tilde{F}(x_{p+1},y_q) - \tilde{F}(x_p,y_q))}{x_{p+1}-x_p}(x-x_p) \\
  \tilde{F}(y_{q+1}) &= \tilde{F}(x_p,y_{q+1}) + \frac{(\tilde{F}(x_{p+1},y_{q+1}) - \tilde{F}(x_p,y_{q+1}))}{x_{p+1}-x_p}(x-x_p) \\
  \tilde{F}(x,y) &= \tilde{F}(y_q) + \frac{(\tilde{F}(y_{q+1}) - \tilde{F}(y_q))}{y_{q+1}-y_q}(y-y_q)
  \end{aligned}$$

Convolutions

The formulas for the binned estimator $\hat{f}$ in the previous subsection are in the form of a convolution product between two matrices, one of which contains the bin counts, the other of which contains the rescaled kernels evaluated at multiples of grid increments. This section defines these two matrices explicitly, and shows that $\hat{f}$ is their convolution.

Beginning with the weighted univariate case, define the following matrices:

$$\begin{aligned}\mathbf{K} &= \frac{1}{n}(\varphi_h(0), \varphi_h(\delta), \ldots, \varphi_h((g-1)\delta))' \\ \mathbf{C} &= (c_1, c_2, \ldots, c_g)'\end{aligned}$$

The first thing to note is that many terms in $\mathbf{K}$ are negligible. The term $\varphi_h(i\delta)$ is taken to be 0 when $|i\delta/h| \geq 5$, so you can define

$$l = \min(g-1, \mathrm{floor}(5h/\delta))$$

as the maximum integer multiple of the grid increment to get nonzero evaluations of the rescaled kernel. Here $\mathrm{floor}(x)$ denotes the largest integer less than or equal to x.

Next, let p be the smallest power of 2 that is greater than $g + l + 1$,

$$p = 2^{\mathrm{ceil}(\log_2(g+l+1))}$$

where $\mathrm{ceil}(x)$ denotes the smallest integer greater than or equal to x.

Modify $\mathbf{K}$ as follows:

$$\mathbf{K} = \frac{1}{n}(\varphi_h(0), \varphi_h(\delta), \ldots, \varphi_h(l\delta), \underbrace{0, \ldots, 0}_{p-2l-1}, \varphi_h(l\delta), \ldots, \varphi_h(\delta))'$$

Essentially, the negligible terms of $\mathbf{K}$ are omitted, and the rest are *symmetrized* (except for one term). The whole matrix is then padded to size $p \times 1$ with zeros in the middle. The dimension p is a highly composite number—that is, one that decomposes into many factors—leading to the most efficient fast Fourier transform operation (see Wand 1994).

The third operation is to pad the bin count matrix $\mathbf{C}$ with zeros to the same size as $\mathbf{K}$:

$$\mathbf{C} = (c_1, c_2, \ldots, c_g, \underbrace{0, \ldots, 0}_{p-g})'$$

The convolution $\mathbf{K} * \mathbf{C}$ is then a $p \times 1$ matrix, and the preceding formulas show that its first g entries are exactly the estimates $\hat{f}(x_i),\ i = 1, 2, \ldots, g$.

For bivariate smoothing, the matrix $\mathbf{K}$ is defined similarly as

$$\mathbf{K} = \begin{bmatrix} \kappa_{0,0} & \kappa_{0,1} & \dots & \kappa_{0,l_Y} & \mathbf{0} & \kappa_{0,l_Y} & \dots & \kappa_{0,1} \\ \kappa_{1,0} & \kappa_{1,1} & \dots & \kappa_{1,l_Y} & \mathbf{0} & \kappa_{1,l_Y} & \dots & \kappa_{1,1} \\ \vdots & & & & & & & \\ \kappa_{l_X,0} & \kappa_{l_X,1} & \dots & \kappa_{l_X,l_Y} & \mathbf{0} & \kappa_{l_X,l_Y} & \dots & \kappa_{l_X,1} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \kappa_{l_X,0} & \kappa_{l_X,1} & \dots & \kappa_{l_X,l_Y} & \mathbf{0} & \kappa_{l_X,l_Y} & \dots & \kappa_{l_X,1} \\ \vdots & & & & & & & \\ \kappa_{1,0} & \kappa_{1,1} & \dots & \kappa_{1,l_Y} & \mathbf{0} & \kappa_{1,l_Y} & \dots & \kappa_{1,1} \end{bmatrix}_{p_X \times p_Y}$$

where $l_X = \min(g_X - 1, \mathrm{floor}(5h_X/\delta_X))$, $p_X = 2^{\mathrm{ceil}(\log_2(g_X + l_X + 1))}$, and so forth, and $\kappa_{i,j} = \frac{1}{n}\varphi_{\mathbf{h}}(i\delta_X, j\delta_Y)$ $i = 0, 1, \dots, l_X$, $j = 0, 1, \dots, l_Y$.

The bin count matrix $\mathbf{C}$ is defined as

$$\mathbf{C} = \begin{bmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,g_Y} & 0 & \dots & 0 \\ c_{2,1} & c_{2,2} & \dots & c_{2,g_Y} & 0 & \dots & 0 \\ \vdots & & & & & & \\ c_{g_X,1} & c_{g_X,2} & \dots & c_{g_X,g_Y} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & & & & & & \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 \end{bmatrix}_{p_X \times p_Y}$$

As with the univariate case, the $g_X \times g_Y$ upper-left corner of the convolution $\mathbf{K} * \mathbf{C}$ is the matrix of the estimates $\hat{f}(grid)$.

Most of the results in this subsection are found in Wand (1994).

Fast Fourier Transform

As shown in the last subsection, kernel density estimates can be expressed as a submatrix of a certain convolution. The fast Fourier transform (FFT) is a computationally effective method for computing such convolutions. For a reference on this material, see Press et al. (1988).

The *discrete Fourier transform* of a complex vector $\mathbf{z} = (z_0, \dots, z_{N-1})$ is the vector $\mathbf{Z} = (Z_0, \dots, Z_{N-1})$, where

$$Z_j = \sum_{l=0}^{N-1} z_l e^{2\pi i l j/N}, \quad j = 0, \dots, N-1$$

and i is the square root of –1. The vector $\mathbf{z}$ can be recovered from $\mathbf{Z}$ by applying the *inverse discrete Fourier transform* formula

$$z_l = N^{-1} \sum_{j=0}^{N-1} Z_j e^{-2\pi i l j/N}, \quad l = 0, \dots, N-1$$

Discrete Fourier transforms and their inverses can be computed quickly using the FFT algorithm, especially when N is *highly composite*; that is, it can be decomposed into many factors, such as a power of 2. By the *discrete convolution theorem*, the convolution of two vectors is the inverse Fourier transform of the element-by-element product of their Fourier transforms. This, however, requires certain periodicity assumptions, which explains why the vectors **K** and **C** require zero-padding. This is to avoid *wrap-around* effects (see Press et al. 1988, pp. 410–411). The vector **K** is actually mirror-imaged so that the convolution of **C** and **K** will be the vector of binned estimates. Thus, if S denotes the inverse Fourier transform of the element-by-element product of the Fourier transforms of **K** and **C**, then the first g elements of S are the estimates.

The bivariate Fourier transform of an $N_1 \times N_2$ complex matrix having $(l_1 + 1, l_2 + 1)$ entry equal to $z_{l_1 l_2}$ is the $N_1 \times N_2$ matrix with $(j_1 + 1, j_2 + 1)$ entry given by

$$Z_{j_1 j_2} = \sum_{l_1=0}^{N_1-1} \sum_{l_2=0}^{N_2-1} z_{l_1 l_2} e^{2\pi i (l_1 j_1/N_1 + l_2 j_2/N_2)}$$

and the formula of the inverse is

$$z_{l_1 l_2} = (N_1 N_2)^{-1} \sum_{j_1=0}^{N_1-1} \sum_{j_2=0}^{N_2-1} Z_{j_1 j_2} e^{-2\pi i (l_1 j_1/N_1 + l_2 j_2/N_2)}$$

The same discrete convolution theorem applies, and zero-padding is needed for matrices **C** and **K**. In the case of **K**, the matrix is mirror-imaged twice. Thus, if S denotes the inverse Fourier transform of the element-by-element product of the Fourier transforms of **K** and **C**, then the upper-left $g_X \times g_Y$ corner of S contains the estimates.

Bandwidth Selection

Several different bandwidth selection methods are available in PROC KDE in the univariate case. Following the recommendations of Jones, Marron, and Sheather (1996), the default method follows a plug-in formula of Sheather and Jones.

This method solves the fixed-point equation

$$h = \left[\frac{R(\varphi)}{nR\left(\hat{f}''_{g(h)}\right)\left(\int x^2 \varphi(x)\, dx\right)^2} \right]^{1/5}$$

where $R(\varphi) = \int \varphi^2(x)\, dx$.

PROC KDE solves this equation by first evaluating it on a grid of values spaced equally on a log scale. The largest two values from this grid that bound a solution are then used as starting values for a bisection algorithm.

The simple normal reference rule works by assuming $\hat{f}$ is Gaussian in the preceding fixed-point equation. This results in

$$\begin{aligned} h &= \hat{\sigma}[4/(3n)]^{1/5} \\ &= 1.06\, \hat{\sigma} n^{-1/5} \end{aligned}$$

where $\hat{\sigma}$ is the sample standard deviation.

Alternatively, the bandwidth can be computed using the interquartile range, Q:

$$\begin{aligned} h &= 1.06\,\hat{\sigma} n^{-1/5} \\ &\approx 1.06\,(Q/1.34) n^{-1/5} \\ &\approx 0.785\,Q n^{-1/5} \end{aligned}$$

Silverman's rule of thumb (Silverman 1986, Section 3.4.2) is computed as

$$h = 0.9 \min[\hat{\sigma}, Q/1.34] n^{-1/5}$$

The oversmoothed bandwidth is computed as

$$h = 3\hat{\sigma}[1/(70\sqrt{\pi} n)]^{1/5}$$

When you specify a WEIGHT variable, PROC KDE uses weighted versions of Q_3, Q_1, and $\hat{\sigma}$ in the preceding expressions. The weighted quartiles are computed as weighted order statistics, and the weighted variance takes the form

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n} W_i (X_i - \bar{X})^2}{\sum_{i=1}^{n} W_i}$$

where $\bar{X} = (\sum_{i=1}^{n} W_i X_i)/(\sum_{i=1}^{n} W_i)$ is the weighted sample mean.

For the bivariate case, Wand and Jones (1993) note that automatic bandwidth selection is both difficult and computationally expensive. Their study of various ways of specifying a bandwidth matrix also shows that using two bandwidths, one in each coordinate's direction, is often adequate. PROC KDE enables you to adjust the two bandwidths by specifying a multiplier for the default bandwidths recommended by Bowman and Foster (1993):

$$\begin{aligned} h_X &= \hat{\sigma}_X n^{-1/6} \\ h_Y &= \hat{\sigma}_Y n^{-1/6} \end{aligned}$$

Here $\hat{\sigma}_X$ and $\hat{\sigma}_Y$ are the sample standard deviations of X and Y, respectively. These are the optimal bandwidths for two independent normal variables that have the same variances as X and Y. They are, therefore, conservative in the sense that they tend to oversmooth the surface.

You can specify the BWM= option to adjust the aforementioned bandwidths to provide the appropriate amount of smoothing for your application.

ODS Table Names

PROC KDE assigns a name to each table that it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 73.3. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 73.3 ODS Tables Produced by PROC KDE

ODS Table Name	Description	Statement	Option
BivariateStatistics	Bivariate statistics	BIVAR	BIVSTATS
Controls	Control variables	Default	
Inputs	Input information	Default	
Levels	Levels of density estimate	BIVAR	LEVELS
Percentiles	Percentiles of data	BIVAR / UNIVAR	PERCENTILES
UnivariateStatistics	Basic statistics	BIVAR / UNIVAR	UNISTATS

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

ODS Graph Names

PROC KDE assigns a name to each graph it creates using the Output Delivery System (ODS). You can use these names to reference the graphs when using ODS. The names are listed in Table 73.4.

Table 73.4 Graphs Produced by PROC KDE

ODS Graph Name	Plot Description	Statement	PLOTS= Option
BivariateHistogram	Bivariate histogram of data	BIVAR	HISTOGRAM
ContourPlot	Contour plot of bivariate kernel density estimate	BIVAR	CONTOUR
ContourScatterPlot	Contour plot of bivariate kernel density estimate overlaid with scatter plot	BIVAR	CONTOURSCATTER
DensityPlot	Univariate kernel density estimate curve	UNIVAR	DENSITY

Table 73.4 *continued*

ODS Graph Name	Plot Description	Statement	PLOTS= Option
DensityOverlayPlot	Overlaid univariate kernel density estimate curves	UNIVAR	DENSITYOVERLAY
HistogramDensity	Univariate histogram overlaid with kernel density estimate curve	UNIVAR	HISTDENSITY
Histogram	Univariate histogram of data	UNIVAR	HISTOGRAM
HistogramSurface	Bivariate histogram overlaid with surface plot of bivariate kernel density estimate	BIVAR	HISTSURFACE
ScatterPlot	Scatter plot of data	BIVAR	SCATTER
SurfacePlot	Surface plot of bivariate kernel density estimate	BIVAR	SURFACE

Bivariate Plots

You can specify the PLOTS= option in the BIVAR statement to request graphical displays of bivariate kernel density estimates.

By default, if ODS Graphics is enabled and you do not specify the PLOTS= option, then the BIVAR statement creates a contour plot. If you specify the PLOTS= option, only the requested plots are created.

Univariate Plots

You can specify the PLOTS= option in the UNIVAR statement to request graphical displays of univariate kernel density estimates.

By default, if ODS Graphics is enabled and you do not specify the PLOTS= option, then the UNIVAR statement creates a histogram overlaid with a kernel density estimate. If you specify the PLOTS= option, only the requested plots are created.

Binning of Bivariate Histogram

Let $(X_i, Y_i),\ i = 1, 2, \ldots, n$, be a sample of size n drawn from a bivariate distribution. For the marginal distribution of $X_i,\ i = 1, 2, \ldots, n$, the number of bins (Nbins_X) in the bivariate histogram is calculated according to the formula

$$\mathrm{Nbins}_X = \mathrm{ceil}\,(\mathrm{range}_X / \mathrm{width}_X)$$

where $\mathrm{ceil}(x)$ denotes the smallest integer greater than or equal to x,

$$\mathrm{range}_X = \max_{1 \le i \le n} (X_i) - \min_{1 \le i \le n} (X_i)$$

and the optimal bin width is obtained, following Scott (1992, p. 84), as

$$\mathrm{width}_X = 3.504\, \hat{\sigma}_X (1 - \hat{\rho}^2)^{3/8} n^{-1/4}$$

Here, $\hat{\sigma}_X$ and $\hat{\rho}$ are the sample variance and the sample correlation coefficient, respectively. When you specify a WEIGHT variable, PROC KDE uses weighted versions of $\hat{\sigma}_X$ and $\hat{\rho}$ in the preceding expressions.

Similar formulas are used to compute the number of bins for the marginal distribution of Y_i, $i = 1, 2, \ldots, n$. Further details can be found in Scott (1992).

Notice that if $|\hat{\rho}| > 0.99$, then Nbins_X is calculated as in the univariate case (see Terrell and Scott 1985). In this case $\mathrm{Nbins}_Y = \mathrm{Nbins}_X$.

Examples: KDE Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 73.1: Computing a Basic Kernel Density Estimate

(View the complete code for this example (kdex1.sas) in the example repository.)

This example illustrates the basic functionality of the UNIVAR statement. The effective channel length (in microns) is measured for 1225 field effect transistors. The channel lengths are saved as values of the variable length in a SAS data set named channel; see the file *kdex1.sas* in the SAS Sample Library. These statements create the channel data set:

```
data channel;
   input length @@;
   datalines;
0.91 1.01 0.95 1.13 1.12 0.86 0.96 1.17 1.36 1.10
0.98 1.27 1.13 0.92 1.15 1.26 1.14 0.88 1.03 1.00
0.98 0.94 1.09 0.92 1.10 0.95 1.05 1.05 1.11 1.15
1.11 0.98 0.78 1.09 0.94 1.05 0.89 1.16 0.88 1.19

   ... more lines ...

2.13 2.05 1.90 2.07 2.15 1.96 2.15 1.89 2.15 2.04
1.95 1.93 2.22 1.74 1.91
;
```

The following statements request a kernel density estimate of the variable length:

```
ods graphics on;
proc kde data=channel;
   univar length;
run;
```

Because ODS Graphics is enabled, PROC KDE produces a histogram with an overlaid kernel density estimate by default, although the PLOTS= option is not specified. The resulting graph is shown in Output 73.1.1. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the KDE procedure, see the section "ODS Graphics" on page 5527.

Output 73.1.1 Histogram with Overlaid Kernel Density Estimate

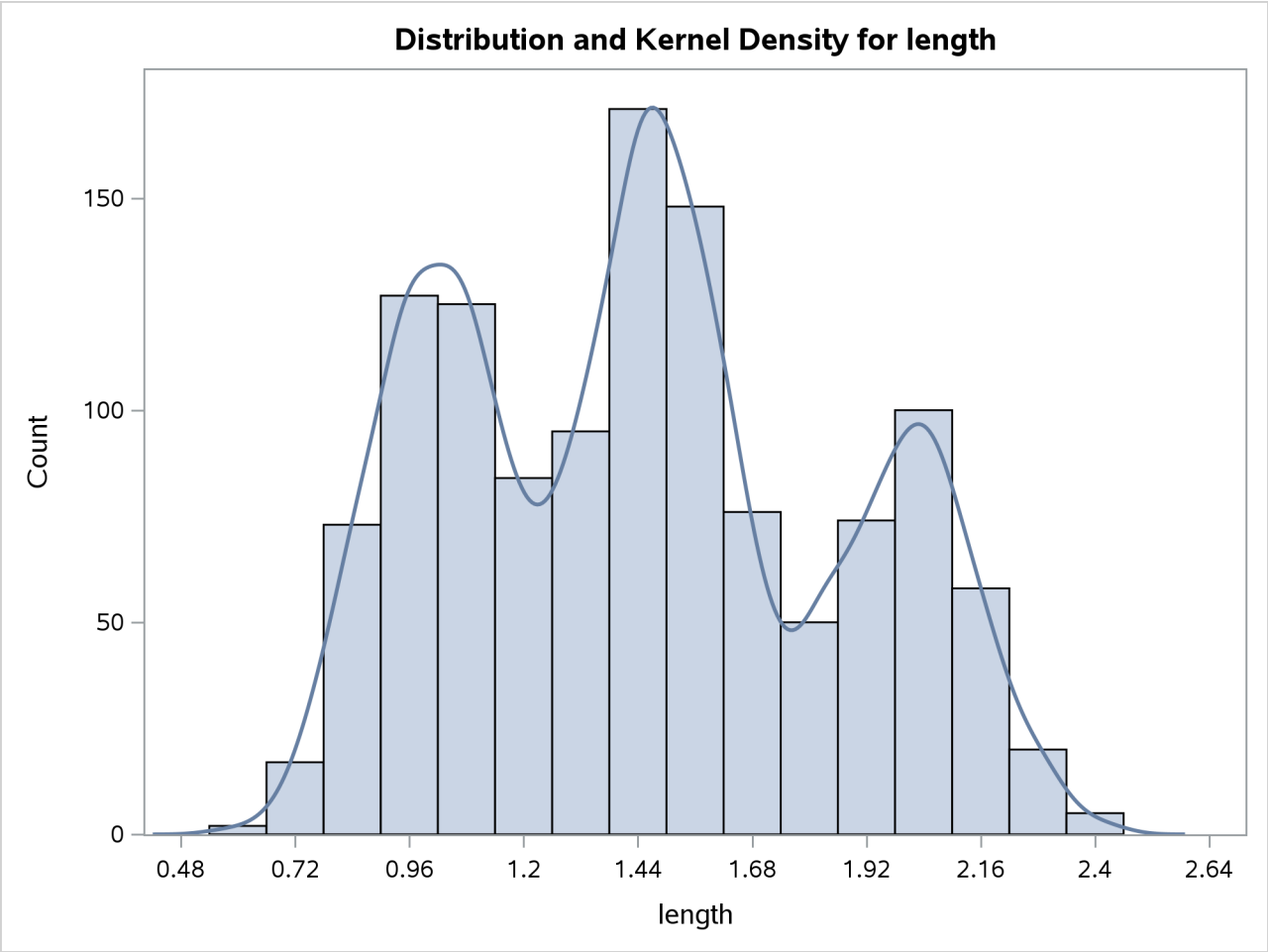

The default output tables for this analysis are the "Inputs" and "Controls" tables, shown in Output 73.1.2.

Output 73.1.2 Univariate Inputs Table

The KDE Procedure

Inputs	
Data Set	WORK.CHANNEL
Number of Observations Used	1225
Variable	length
Bandwidth Method	Sheather-Jones Plug In

Controls	
	length
Grid Points	401
Lower Grid Limit	0.421
Upper Grid Limit	2.589
Bandwidth Multiplier	1

The "Inputs" table lists basic information about the density fit, including the input data set, the number of observations, the variable used, and the bandwidth method. The default bandwidth method is the Sheather-Jones plug-in.

The "Controls" table lists the primary numbers controlling the kernel density fit. Here the default number of grid points is used and no adjustment is made to the default bandwidth.

Example 73.2: Changing the Bandwidth

(View the complete code for this example (kdex1.sas) in the example repository.)

Continuing with Example 73.1, you can specify different bandwidth multipliers that determine the smoothness of the kernel density estimate. The following statements show kernel density estimates for the variable length by specifying two different bandwidth multipliers with the BWM= option:

```
proc kde data=channel;
   univar length(bwm=2) length(bwm=0.25);
run;
ods graphics off;
```

Output 73.2.1 shows an oversmoothed estimate because the bandwidth multiplier is 2. Output 73.2.2 is created by specifying BWM=0.25, so it is an undersmoothed estimate.

Output 73.2.1 Histogram with Oversmoothed Kernel Density Estimate

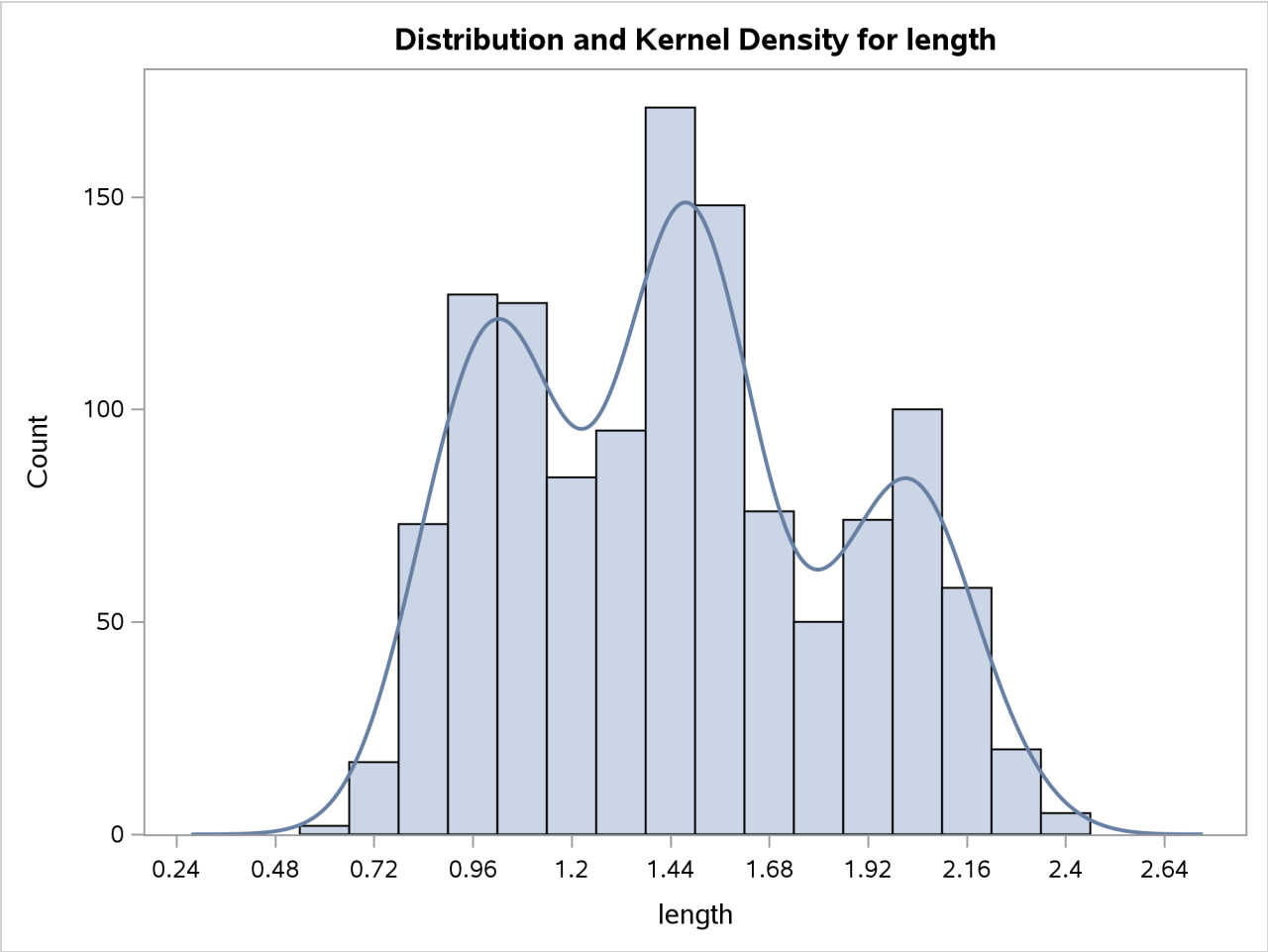

Output 73.2.2 Histogram with Undersmoothed Kernel Density Estimate

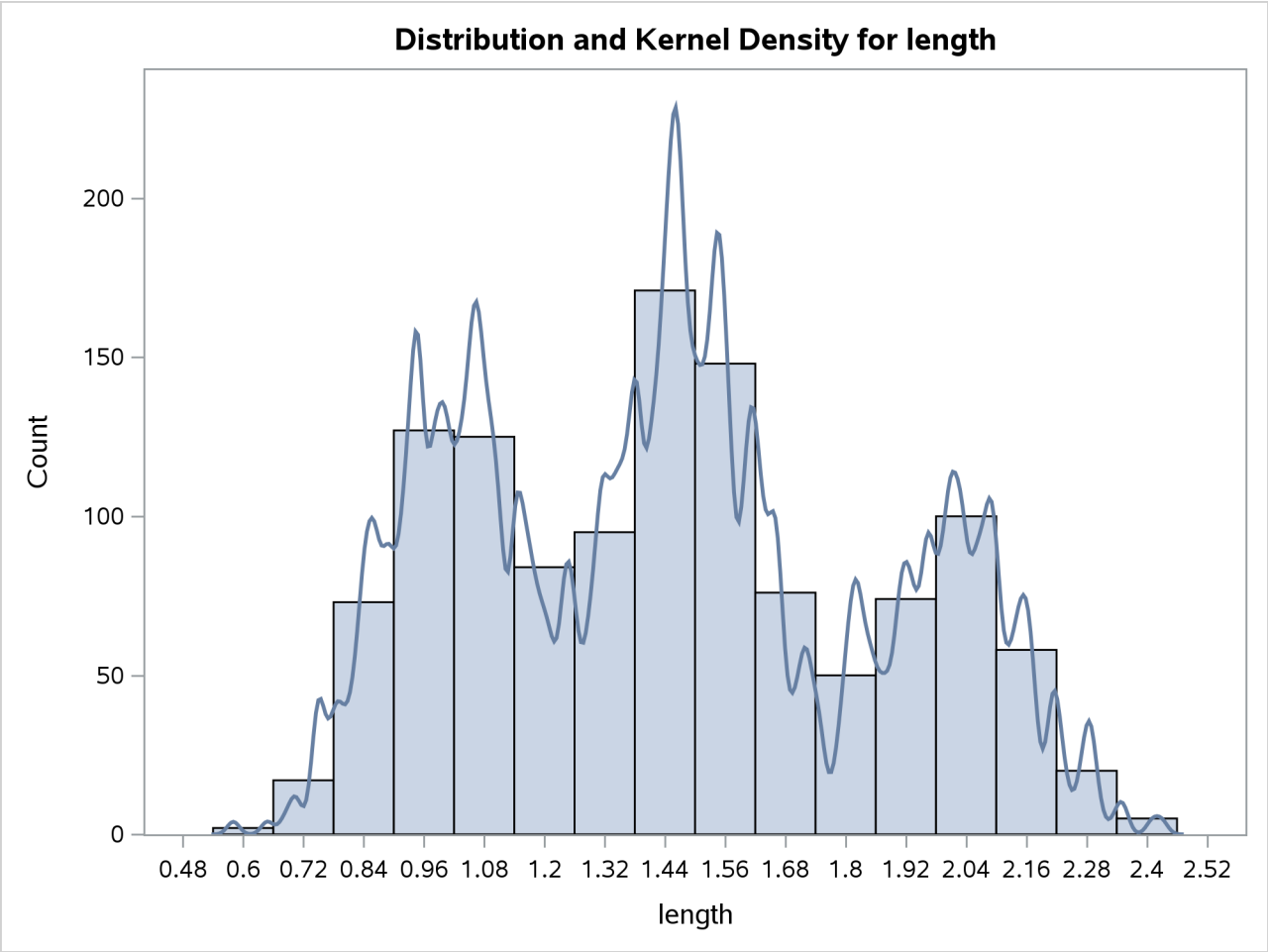

Example 73.3: Changing the Bandwidth (Bivariate)

(View the complete code for this example (kdex2.sas) in the example repository.)

Recall the analysis from the section "Getting Started: KDE Procedure" on page 5502. Suppose you would like a slightly smoother estimate. You could then rerun the analysis with a larger bandwidth:

```
ods graphics on;
proc kde data=bivnormal;
   bivar x y / bwm=2;
run;
```

The BWM= option requests bandwidth multipliers of 2 for both x and y. With ODS Graphics enabled, the BIVAR statement produces a contour plot, as shown in Output 73.3.1.

Output 73.3.1 Contour Plot of Estimated Density with Additional Smoothing

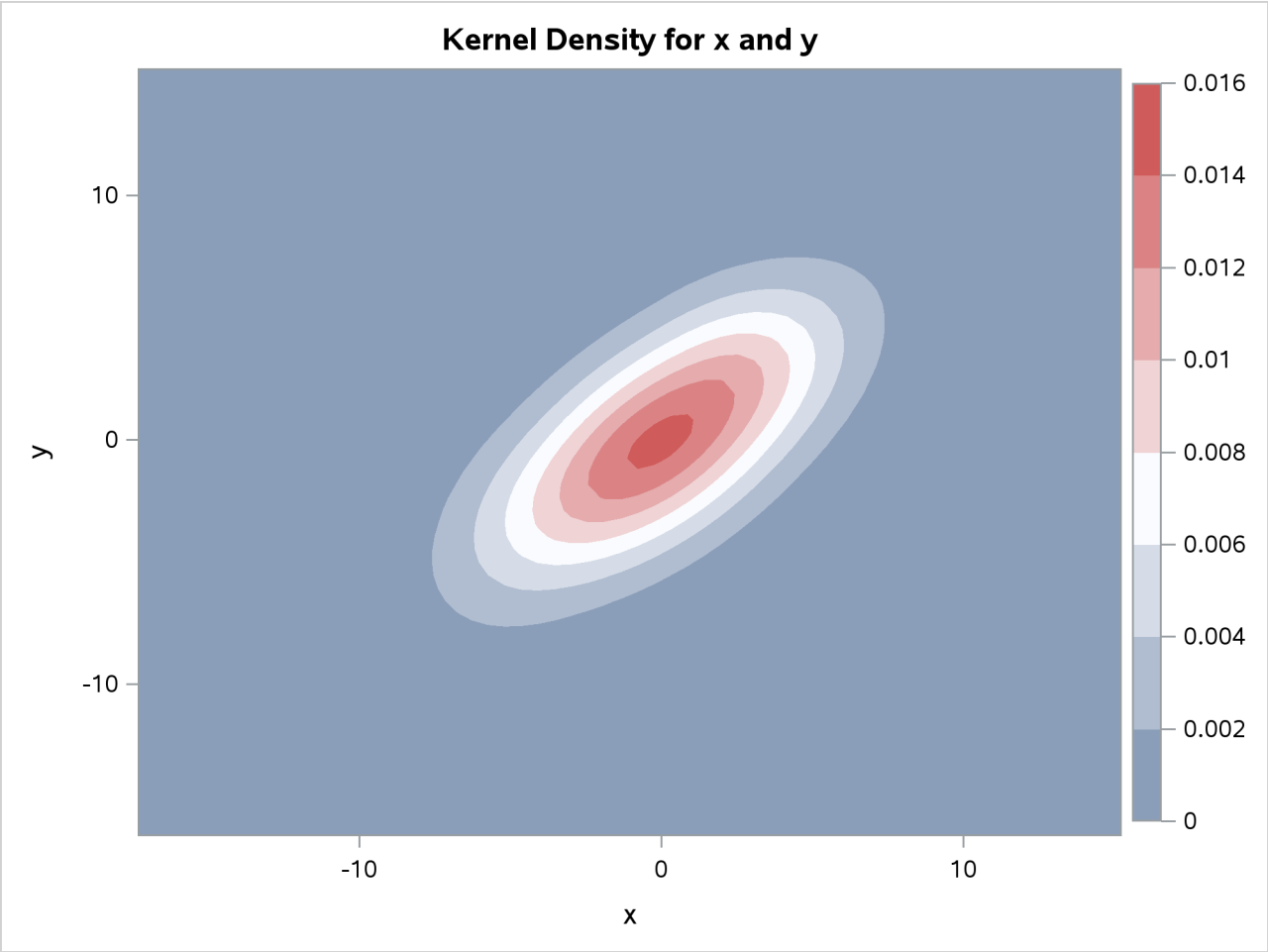

Multiple Bandwidths

You can also specify multiple bandwidths with only one run of the KDE procedure. Notice that by specifying pairs of variables inside parentheses, a kernel density estimate is computed for each pair. In the following statements the first kernel density is computed with the default bandwidth, but the second kernel density specifies a bandwidth multiplier of 0.5 for the variable x and a multiplier of 2 for the variable y:

```
proc kde data=bivnormal;
   bivar (x y) (x (bwm=0.5) y (bwm=2));
run;
ods graphics off;
```

The contour plot of the second kernel density estimate is shown in Output 73.3.2.

Output 73.3.2 Contour Plot of Estimated Density with Different Smoothing for x and y

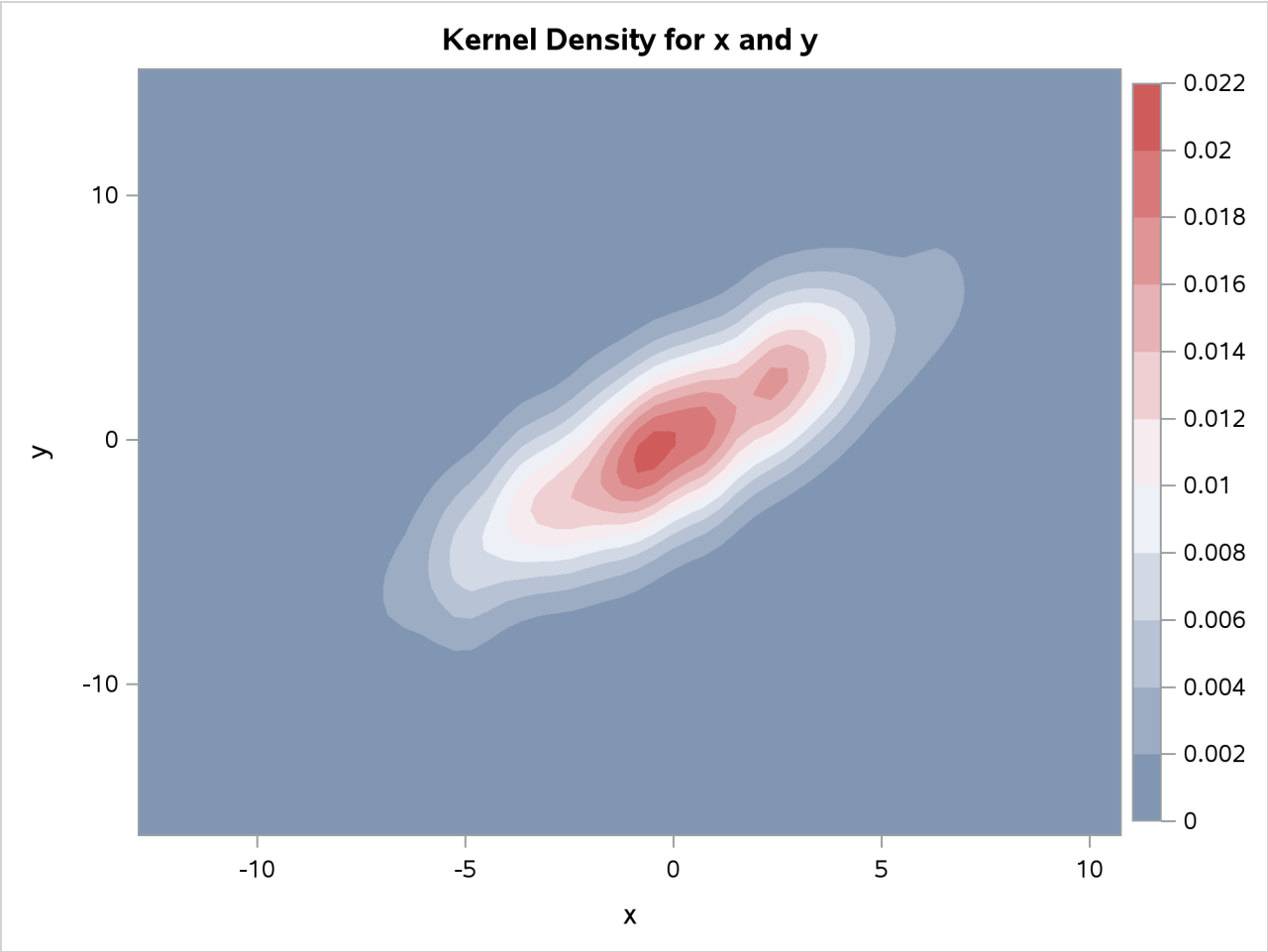

Example 73.4: Requesting Additional Output Tables

(View the complete code for this example (kdex2.sas) in the example repository.)

This example illustrates how to request output tables with summary statistics in addition to the default output tables. Using the same data as in the section "Getting Started: KDE Procedure" on page 5502, the following statements request univariate and bivariate summary statistics, percentiles, and levels of the kernel density estimate:

```
proc kde data=bivnormal;
   bivar x y / bivstats levels percentiles unistats;
run;
```

The resulting output is shown in Output 73.4.1.

Output 73.4.1 Bivariate Kernel Density Estimate Tables

The KDE Procedure

Inputs	
Data Set	WORK.BIVNORMAL
Number of Observations Used	1000
Variable 1	x
Variable 2	y
Bandwidth Method	Simple Normal Reference

Controls	x	y
Grid Points	60	60
Lower Grid Limit	-14.27	-13.18
Upper Grid Limit	12.17	12.165
Bandwidth Multiplier	1	1

Univariate Statistics	x	y
Mean	-0.075	-0.070
Variance	9.73	9.93
Standard Deviation	3.12	3.15
Range	20.39	19.09
Interquartile Range	4.46	4.51
Bandwidth	0.99	1.00

Bivariate Statistics	
Covariance	8.88
Correlation	0.90

Percentiles	x	y
0.5	-7.71	-8.44
1.0	-7.08	-7.46
2.5	-6.17	-6.31
5.0	-5.28	-5.23
10.0	-4.18	-4.11
25.0	-2.24	-2.30
50.0	-0.11	-0.058
75.0	2.22	2.21
90.0	3.81	3.94
95.0	4.88	5.22
97.5	6.03	5.94
99.0	6.90	6.77
99.5	7.71	7.07

Output 73.4.1 *continued*

		Levels			
Percent	Density	Lower for x	Upper for x	Lower for y	Upper for y
1	0.001034	-8.45	8.58	-8.89	8.30
5	0.003076	-7.10	7.24	-7.17	7.01
10	0.004886	-6.21	5.90	-6.31	6.15
50	0.01586	-3.52	3.65	-3.73	4.00
90	0.02338	-1.28	1.41	-1.58	0.99
95	0.02464	-1.28	0.97	-1.15	0.57
99	0.02547	-0.83	0.069	-0.72	0.14
100	0.02564	-0.38	-0.38	-0.29	-0.29

The "Univariate Statistics" table contains standard univariate statistics for each variable, as well as statistics associated with the density estimate. Note that the estimated variances for both x and y are fairly close to the true values of 10.

The "Bivariate Statistics" table lists the covariance and correlation between the two variables. Note that the estimated correlation is equal to its true value to two decimal places.

The "Percentiles" table lists percentiles for each variable.

The "Levels" table lists contours of the density corresponding to percentiles of the bivariate data, and the minimum and maximum values of each variable on those contours. For example, 5% of the observed data have a density value less than 0.0030. The minimum x and y values on this contour are –7.10 and –7.14, respectively (the Lower for x and Lower for y columns), and the maximum values are 7.07 and 6.77, respectively (the Upper for x and Upper for y columns).

You can also request "Percentiles" or "Levels" tables with specific percentiles:

```
proc kde data=bivnormal;
   bivar x y / levels=2.5, 50, 97.5
               percentiles=2.5, 25, 50, 75, 97.5;
run;
```

The resulting "Percentiles" and "Levels" tables are shown in Output 73.4.2.

Output 73.4.2 Customized Percentiles and Levels Tables

The KDE Procedure

	Percentiles	
	x	y
2.5	-6.17	-6.31
25.0	-2.24	-2.30
50.0	-0.11	-0.058
75.0	2.22	2.21
97.5	6.03	5.94

Output 73.4.2 *continued*

		Levels			
Percent	Density	Lower for x	Upper for x	Lower for y	Upper for y
2.5	0.001990	-7.55	8.14	-8.03	7.44
50.0	0.01586	-3.52	3.65	-3.73	4.00
97.5	0.02504	-0.83	0.52	-0.72	0.57

Example 73.5: Univariate KDE Graphics

(View the complete code for this example (kdex2.sas) in the example repository.)

This example uses data from the section "Getting Started: KDE Procedure" to illustrate the use of ODS Graphics. The following statements request the available univariate plots in PROC KDE:

```
ods graphics on;
proc kde data=bivnormal;
   univar x / plots=(density histogram histdensity);
   univar x y / plots=densityoverlay;
run;
ods graphics off;
```

Graphs are requested by specifying the PLOTS= option in the UNIVAR statement with ODS Graphics enabled. Output 73.5.1, Output 73.5.2, and Output 73.5.3 show the kernel density estimate, histogram, and histogram with kernel density estimate overlaid, respectively, produced by the first UNIVAR statement.

Output 73.5.1 Kernel Density Estimate

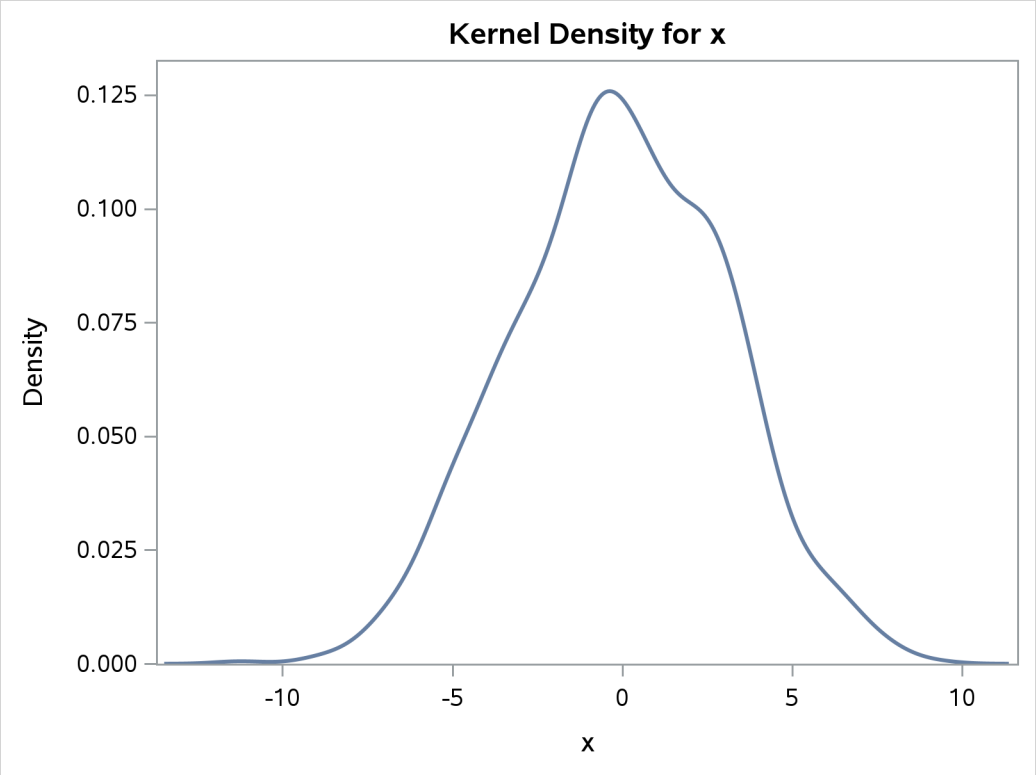

Output 73.5.2 Histogram

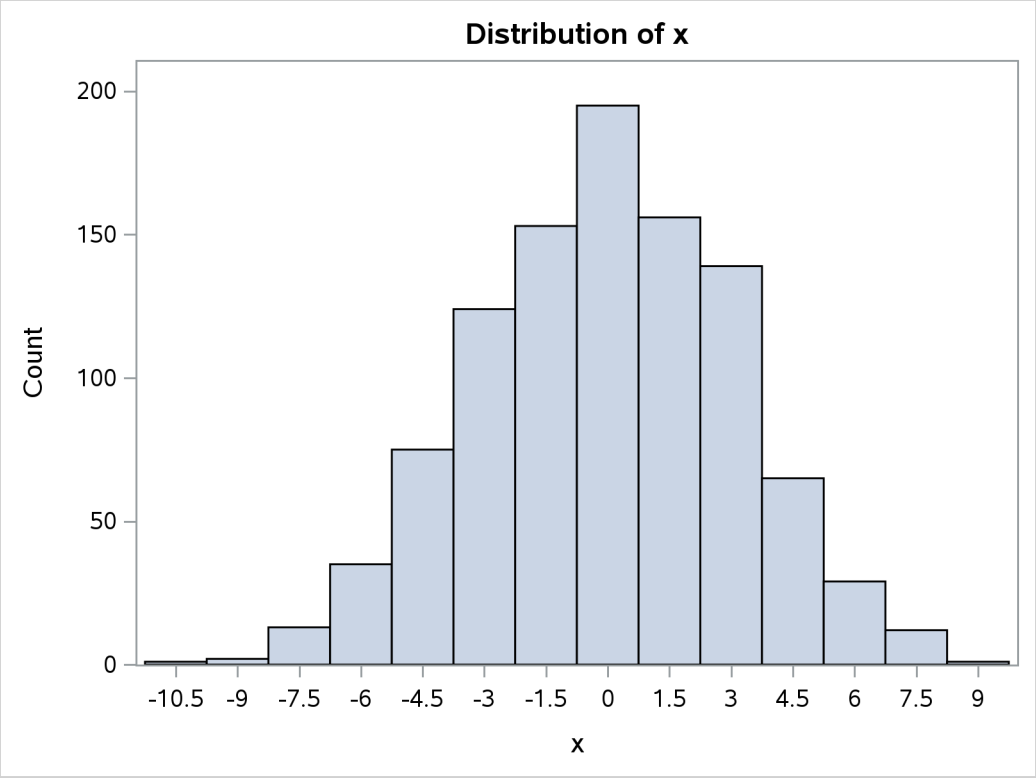

Output 73.5.3 Histogram with Overlaid Kernel Density Estimate

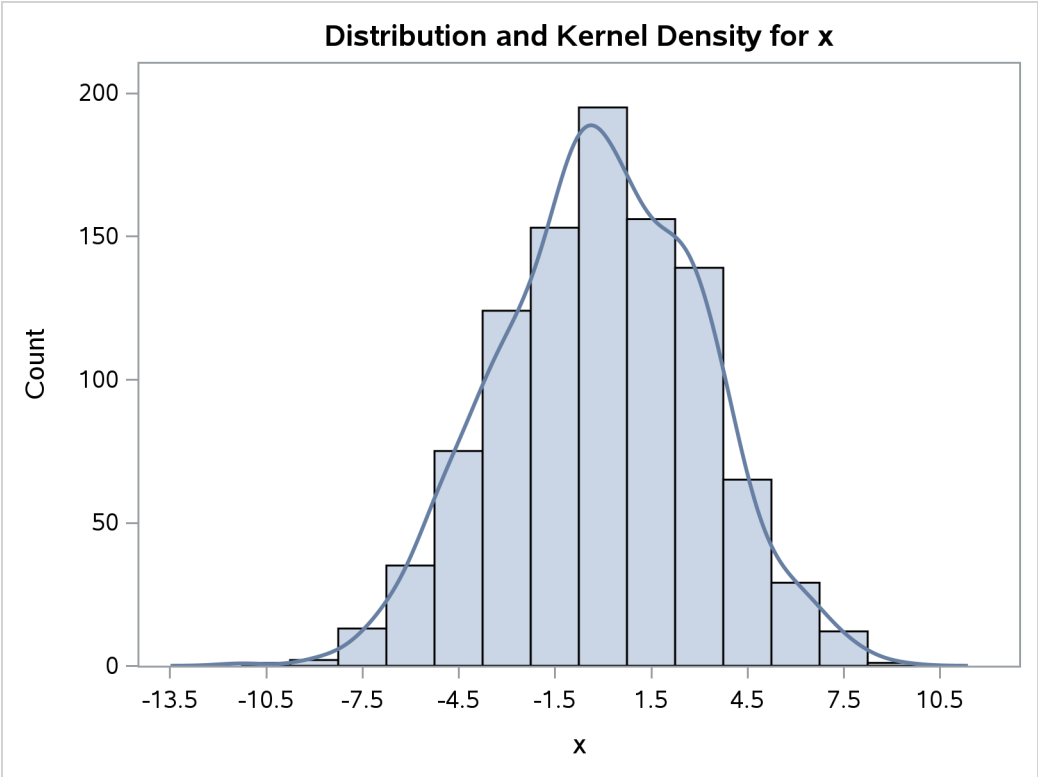

Output 73.5.4 shows the plot produced by the second UNIVAR statement, in which the kernel density estimates for x and y are overlaid.

Output 73.5.4 Overlaid Kernel Density Estimates

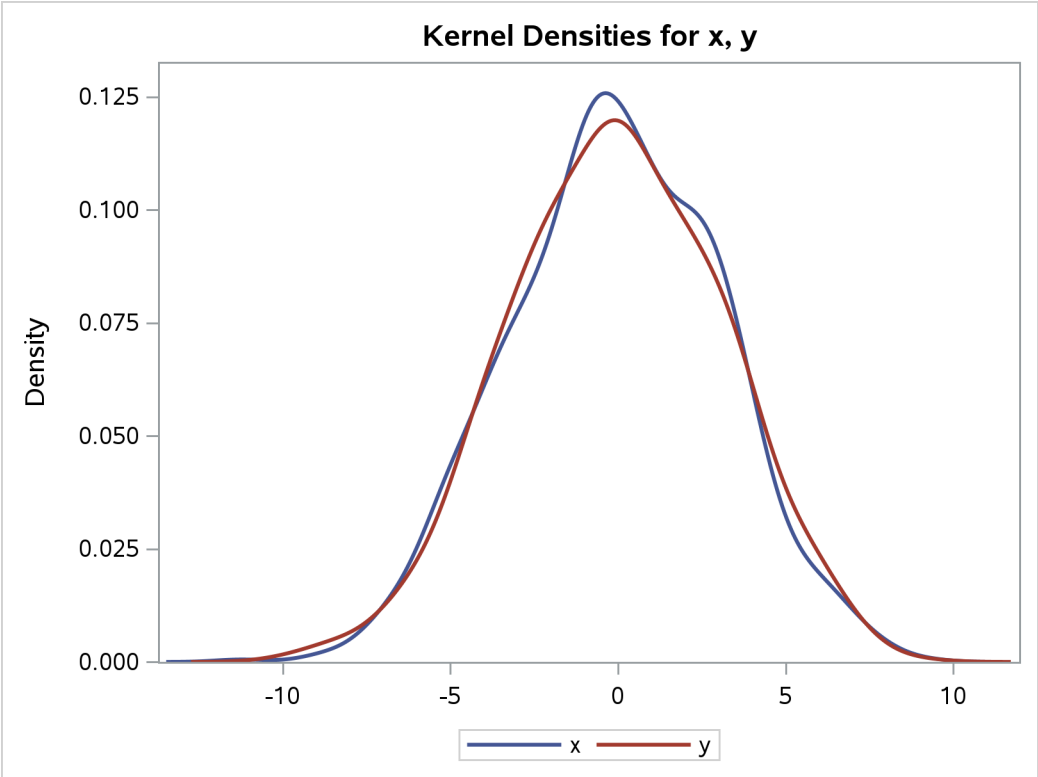

For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the KDE procedure, see the section "ODS Graphics" on page 5527.

Example 73.6: Bivariate KDE Graphics

(View the complete code for this example (kdex3.sas) in the example repository.)

This example illustrates the available bivariate graphics in PROC KDE. The octane data set comes from Rodriguez and Taniguchi (1980), where it is used for predicting customer octane satisfaction by using trained-rater observations. The variables in this data set are Rater and Customer. Either variable might have missing values. See the file *kdex3.sas* in the SAS Sample Library. The following statements create the octane data set:

```
data octane;
   input Rater Customer;
   label Rater    = 'Rater'
         Customer = 'Customer';
   datalines;
94.5 92.0
94.0 88.0
94.0 90.0
93.0 93.0

   ... more lines ...
```

```
88.0 84.0
  .H 90.0
;
```

The following statements request all the available bivariate plots in PROC KDE:

```
ods graphics on;
proc kde data=octane;
   bivar Rater Customer / plots=all;
run;
ods graphics off;
```

Output 73.6.1 shows a scatter plot of the data, Output 73.6.2 shows a bivariate histogram of the data, Output 73.6.3 shows a contour plot of bivariate density estimate, Output 73.6.4 shows a contour plot of bivariate density estimate overlaid with a scatter plot of data, Output 73.6.5 shows a surface plot of bivariate kernel density estimate, and Output 73.6.6 shows a bivariate histogram overlaid with a bivariate kernel density estimate. These graphical displays are requested by specifying the PLOTS= option in the BIVAR statement with ODS Graphics enabled. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the KDE procedure, see the section "ODS Graphics" on page 5527.

Output 73.6.1 Scatter Plot

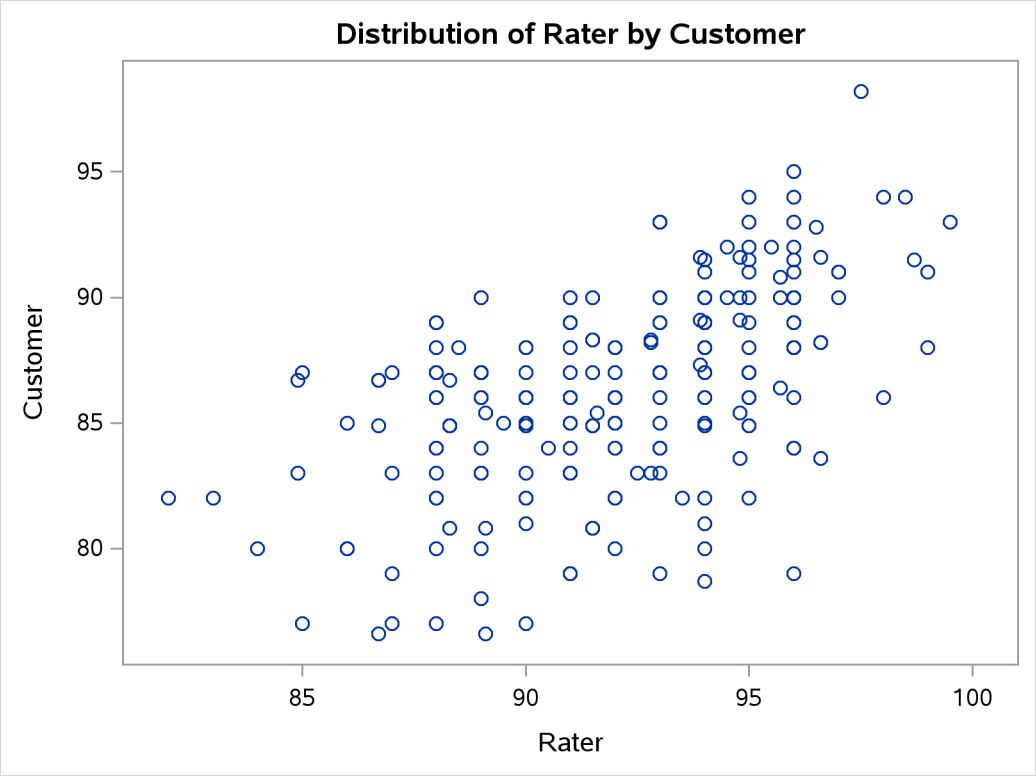

Output 73.6.2 Bivariate Histogram

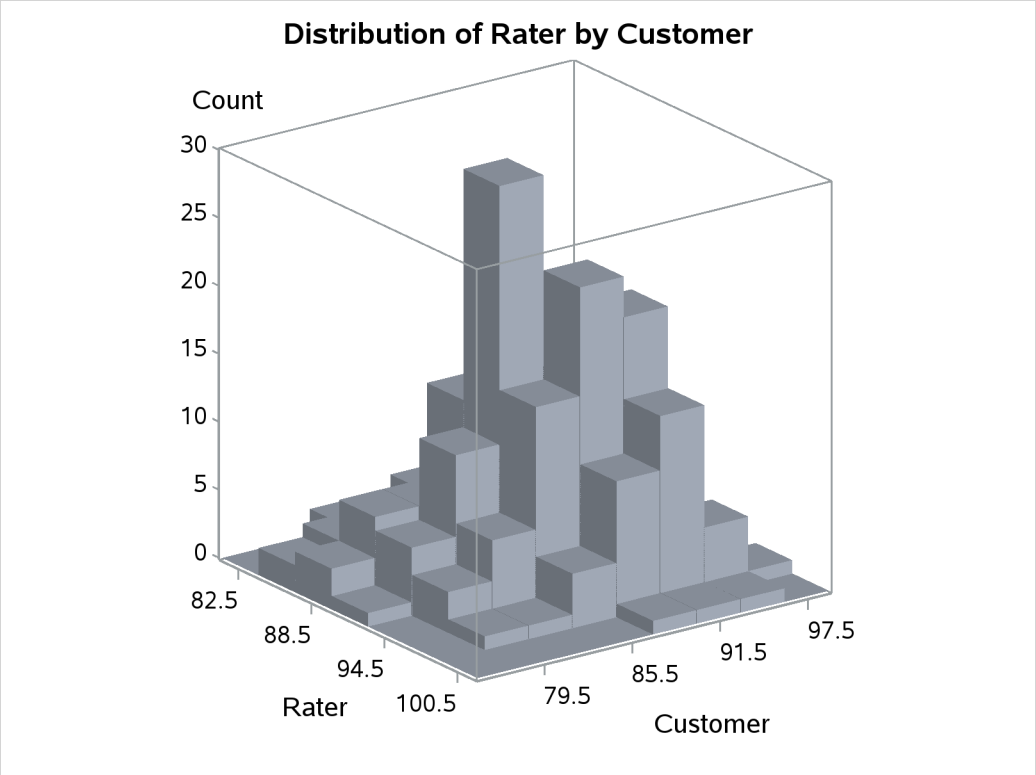

Output 73.6.3 Contour Plot

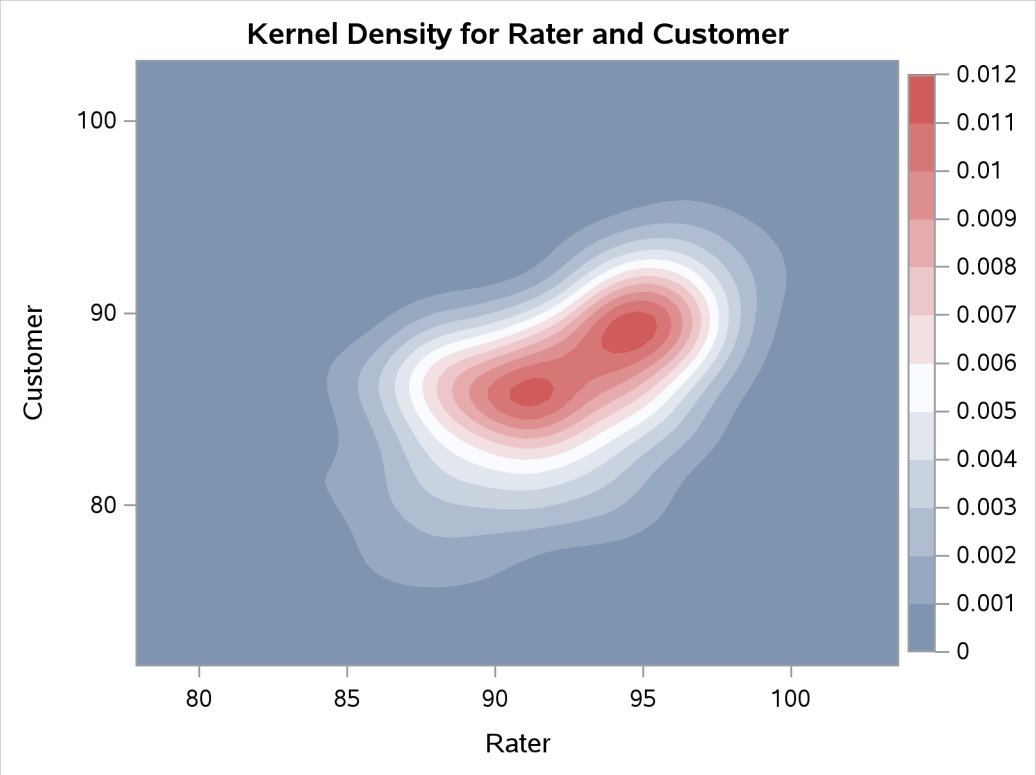

Output 73.6.4 Contour Plot with Overlaid Scatter Plot

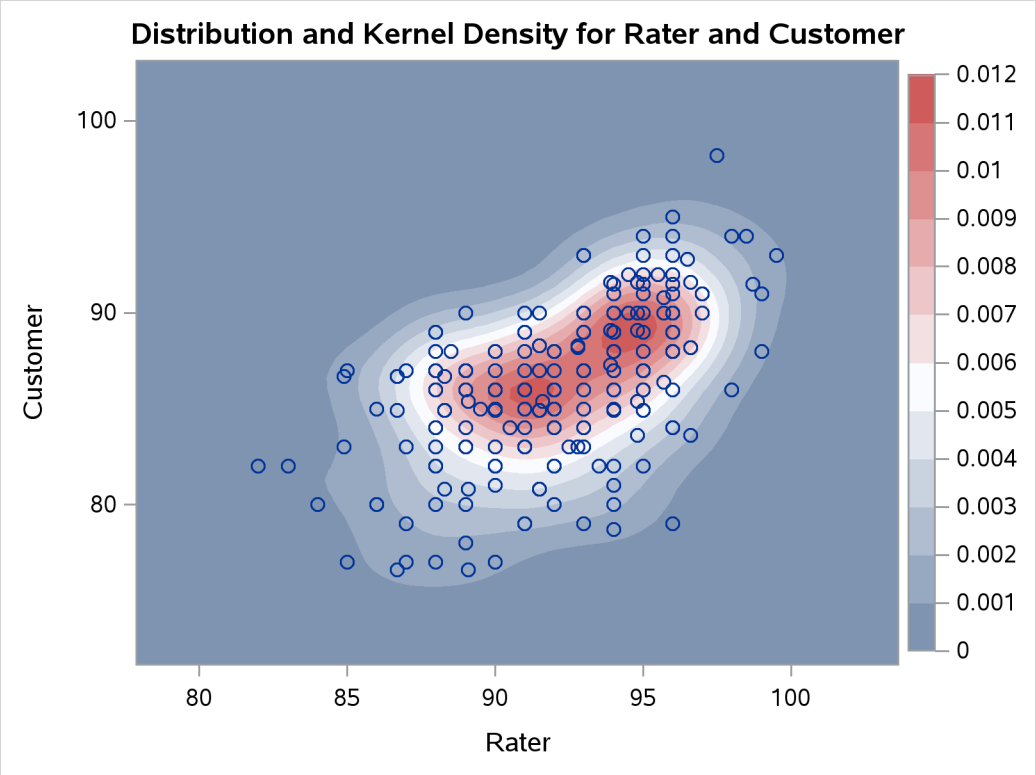

Output 73.6.5 Surface Plot

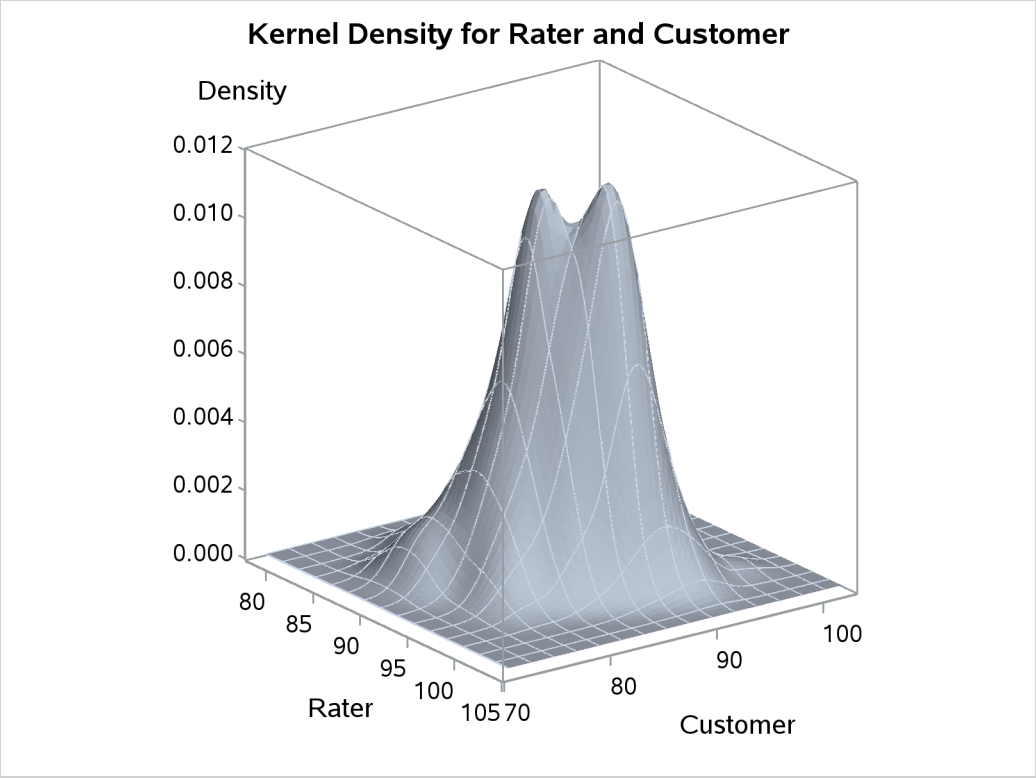

Output 73.6.6 Bivariate Histogram with Overlaid Surface Plot

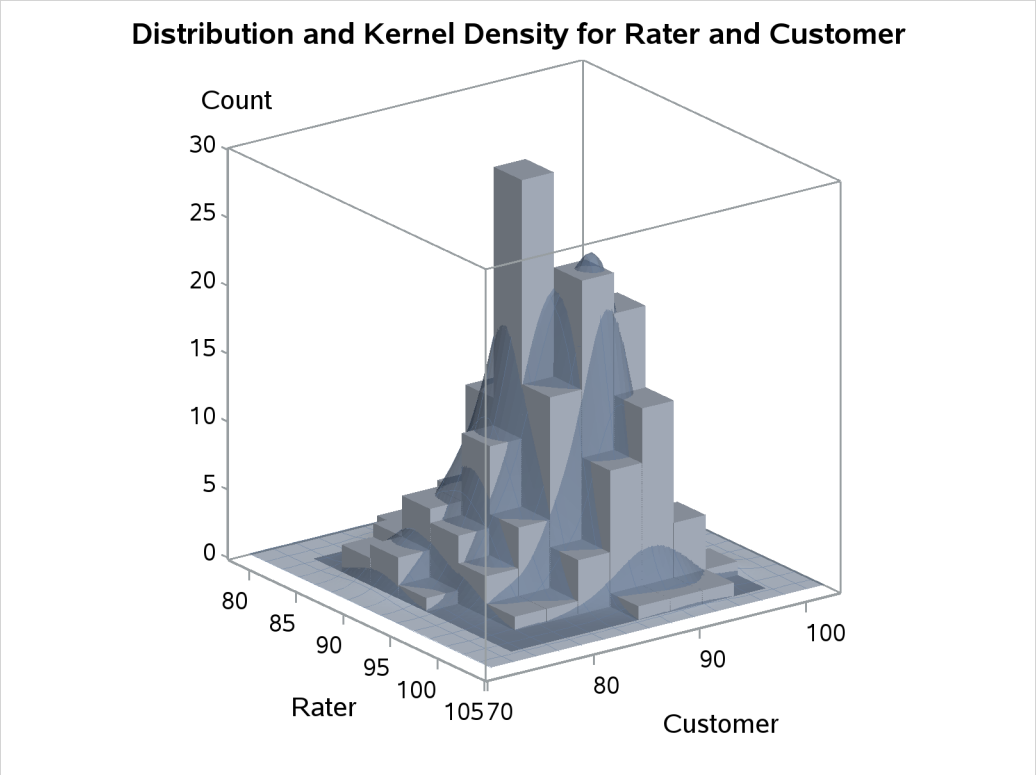

References

Azzalini, A. (1981). "A Note on the Estimation of a Distribution Function and Quantiles by a Kernel Method." *Biometrika* 68:326–328.

Bowman, A. W., and Foster, P. J. (1993). "Density Based Exploration of Bivariate Data." *Statistics and Computing* 3:171–177.

Fan, J., and Marron, J. S. (1994). "Fast Implementations of Nonparametric Curve Estimators." *Journal of Computational and Graphical Statistics* 3:35–56.

Jones, M. C., Marron, J. S., and Sheather, S. J. (1996). "A Brief Survey of Bandwidth Selection for Density Estimation." *Journal of the American Statistical Association* 91:401–407.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1988). *Numerical Recipes: The Art of Scientific Computing*. Cambridge: Cambridge University Press.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical Recipes in C: The Art of Scientific Computing*. 2nd ed. Cambridge: Cambridge University Press.

Rodriguez, R. N., and Taniguchi, B. Y. (1980). "A New Statistical Model for Predicting Customer Octane Satisfaction Using Trained-Rater Observations." *Transactions of the Society of Automotive Engineers* 4213–4235.

Scott, D. W. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization*. New York: John Wiley & Sons.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. New York: Chapman & Hall.

Terrell, G. R., and Scott, D. W. (1985). "Oversmoothed Nonparametric Density Estimates." *Journal of the American Statistical Association* 80:209–214.

Wand, M. P. (1994). "Fast Computation of Multivariate Kernel Estimators." *Journal of Computational and Graphical Statistics* 3:433–445.

Wand, M. P., and Jones, M. C. (1993). "Comparison of Smoothing Parameterizations in Bivariate Kernel Density Estimation." *Journal of the American Statistical Association* 88:520–528.

Chapter 74
The KRIGE2D Procedure

Contents

Overview: KRIGE2D Procedure . . . **5548**
- Introduction to Spatial Prediction . . . 5548

Getting Started: KRIGE2D Procedure . . . **5549**
- Spatial Prediction Using Kriging, Contour Plots . . . 5549

Syntax: KRIGE2D Procedure . . . **5554**
- PROC KRIGE2D Statement . . . 5555
- BY Statement . . . 5561
- COORDINATES Statement . . . 5562
- GRID Statement . . . 5562
- ID Statement . . . 5565
- PREDICT Statement . . . 5565
- MODEL Statement . . . 5567
- RESTORE Statement . . . 5575

Details: KRIGE2D Procedure . . . **5577**
- Theoretical Semivariogram Models . . . 5577
- The Nugget Effect . . . 5585
- Anisotropic Models . . . 5586
  - Geometric Anisotropy . . . 5587
  - Zonal Anisotropy . . . 5590
  - Anisotropic Nugget Effect . . . 5594
- Details of Ordinary Kriging . . . 5594
  - Introduction . . . 5594
  - Spatial Random Fields . . . 5595
  - Ordinary Kriging . . . 5595
- Computational Resources . . . 5598
- Output Data Sets . . . 5598
- Displayed Output . . . 5599
- ODS Table Names . . . 5600
- ODS Graphics . . . 5601

Examples: KRIGE2D Procedure . . . **5602**
- Example 74.1: Spatial Prediction of Pollutant Concentration . . . 5602
- Example 74.2: Investigating the Effect of Model Specification on Spatial Prediction . 5612
- Example 74.3: Data Quality and Prediction with Missing Values . . . 5616

References . . . **5621**

Overview: KRIGE2D Procedure

The KRIGE2D procedure performs ordinary kriging in two dimensions. PROC KRIGE2D can handle anisotropic and nested semivariogram models. Eight semivariogram models are supported: the Gaussian, exponential, spherical, power, cubic, pentaspherical, sine hole effect, and Matérn models. A single nugget effect is also supported. You can specify the correlation model by naming the form and supplying the associated parameters, or by using the contents of an item store file that was previously created by PROC VARIOGRAM.

You can specify the locations of kriging predictions in a GRID statement, or they can be read from a SAS data set. The grid specification is most suitable for a regular grid; the data set specification can handle any irregular pattern of points.

Local kriging is supported through the specification of a radius around a grid point or the specification of the number of nearest neighbors to use in the kriging system. When you perform local kriging, a separate kriging system is solved at each grid point by using a neighborhood of the data point established by the radius or number specification.

The KRIGE2D procedure writes the kriging predictions and associated standard errors for each grid to an output data set. When you perform local kriging, PROC KRIGE2D writes the neighborhood information for each grid point to an additional, optional data set. The KRIGE2D procedure does not produce any displayed output.

The KRIGE2D procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For more information about the graphics available in PROC KRIGE2D, see the section "ODS Graphics" on page 5601.

Introduction to Spatial Prediction

Many activities in science and technology involve measurements of one or more quantities at given spatial locations, with the goal of predicting the measured quantities at unsampled locations. Application areas include reservoir prediction in mining and petroleum exploration, in addition to modeling in a broad spectrum of fields (for example, environmental health, environmental pollution, natural resources and energy, hydrology, and risk analysis). Often, the unsampled locations are on a regular grid, and the predictions are used to produce surface plots or contour maps.

The preceding tasks fall within the scope of *spatial prediction*, which, in general, is any prediction method that incorporates spatial dependence. The study of these tasks involves naturally occurring uncertainties that cannot be ignored. Stochastic analysis frameworks and methods are often used to account for these uncertainties. Hence, the terms *stochastic spatial prediction* and *stochastic modeling* are also used to characterize this type of analysis.

A popular method of spatial prediction is *ordinary kriging*, which produces both predicted values and associated standard errors. Ordinary kriging requires the complete specification (the form and parameter values) of the spatial dependence that characterizes the spatial process. For this purpose, models for the spatial dependence are expressed in terms of the distance between any two locations in the spatial domain of interest. These models take the form of a covariance or semivariance function.

Spatial prediction, then, involves two steps. First, you model the covariance or semivariance of the spatial process. These measures are typically not known in advance. This step involves computing an empirical estimate, in addition to determining both the mathematical form and the values of any parameters for a theoretical form of the dependence model. Second, you use this dependence model to solve the kriging system at a specified set of spatial points, resulting in predicted values and associated standard errors.

The KRIGE2D procedure performs the second of these steps by using ordinary kriging of two-dimensional data.

This introduction concludes with a note on terminology. You might commonly encounter the terms *estimation* and *prediction* used interchangeably by experts in different fields; this could be a source of confusion. A precise statistical vernacular uses the term *estimation* to refer to inferences about the value of fixed but unknown parameters, whereas *prediction* concerns inferences about the value of random variables—see, for example, Cressie (1993, p. 106). In light of these definitions, kriging methods are clearly predictive techniques, since they are concerned with making inferences about the value of a spatial random field at observed or unobserved locations. The SAS/STAT suite of procedures for spatial analysis and prediction (VARIOGRAM, KRIGE2D, and SIM2D) follows the statistical vernacular in the use of the terms *estimation* and *prediction*.

Getting Started: KRIGE2D Procedure

Spatial Prediction Using Kriging, Contour Plots

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/kri2dgs1.sas`.)

After an appropriate semivariogram model is chosen, a number of choices are involved in producing the kriging surface. In order to illustrate these choices, you use the theoretical semivariogram model that was fitted to the coal seam thickness data empirical semivariogram in "Theoretical Semivariogram Model Fitting" on page 10971 in Chapter 131, "The VARIOGRAM Procedure." This model is Gaussian,

$$\gamma_z(h) = c_0 \left[1 - \exp\left(-\frac{h^2}{a_0^2} \right) \right]$$

with a scale of $c_0 = 7.4599$ (that is, the model sill) and a range of $a_0 = 30.1111$, based on the weighted least squares fitting results in the PROC VARIOGRAM example.

The first choice is whether to use local or global kriging. Local kriging uses only data points in the neighborhood of a grid point, and you choose this type of analysis by specifying a data search radius around the grid point. Global kriging uses all data points.

The most important consideration in this decision is the spatial covariance structure. Global kriging is appropriate when the correlation range ϵ is approximately equal to the length of the spatial domain. The correlation range ϵ is the distance r_ϵ (also known as *effective* or *practical* range) at which the covariance is 5% of its value at zero. That is,

$$C_Z(r_\epsilon) = 0.05 C_z(0)$$

For a Gaussian model, r_ϵ is $\sqrt{3}a_0 \approx 52{,}000$ feet. The data points are scattered uniformly throughout a 100×100 (10^6 ft^2) area. Hence, the linear dimension of the data is nearly double the r_ϵ range. This indicates that local kriging rather than global kriging is appropriate because data that are farther away than r_ϵ essentially add to the computational burden without significant contribution to the prediction. The following DATA step inputs the thickness data set thick, which is available from the Sashelp library. In the thick data set, thickness is represented by the Thick variable.

```
title 'Spatial Prediction With Kriging';

data thick;
   input East North Thick @@;
   label Thick='Coal Seam Thickness';
   datalines;
    0.7  59.6  34.1   2.1  82.7  42.2   4.7  75.1  39.5
    4.8  52.8  34.3   5.9  67.1  37.0   6.0  35.7  35.9
    6.4  33.7  36.4   7.0  46.7  34.6   8.2  40.1  35.4
   13.3   0.6  44.7  13.3  68.2  37.8  13.4  31.3  37.8
   17.8   6.9  43.9  20.1  66.3  37.7  22.7  87.6  42.8
   23.0  93.9  43.6  24.3  73.0  39.3  24.8  15.1  42.3
   24.8  26.3  39.7  26.4  58.0  36.9  26.9  65.0  37.8
   27.7  83.3  41.8  27.9  90.8  43.3  29.1  47.9  36.7
   29.5  89.4  43.0  30.1   6.1  43.6  30.8  12.1  42.8
   32.7  40.2  37.5  34.8   8.1  43.3  35.3  32.0  38.8
   37.0  70.3  39.2  38.2  77.9  40.7  38.9  23.3  40.5
   39.4  82.5  41.4  43.0   4.7  43.3  43.7   7.6  43.1
   46.4  84.1  41.5  46.7  10.6  42.6  49.9  22.1  40.7
   51.0  88.8  42.0  52.8  68.9  39.3  52.9  32.7  39.2
   55.5  92.9  42.2  56.0   1.6  42.7  60.6  75.2  40.1
   62.1  26.6  40.1  63.0  12.7  41.8  69.0  75.6  40.1
   70.5  83.7  40.9  70.9  11.0  41.7  71.5  29.5  39.8
   78.1  45.5  38.7  78.2   9.1  41.7  78.4  20.0  40.8
   80.5  55.9  38.7  81.1  51.0  38.6  83.8   7.9  41.6
   84.5  11.0  41.5  85.2  67.3  39.4  85.5  73.0  39.8
   86.7  70.4  39.6  87.2  55.7  38.8  88.1   0.0  41.6
   88.4  12.1  41.3  88.4  99.6  41.2  88.8  82.9  40.5
   88.9   6.2  41.5  90.6   7.0  41.5  90.7  49.6  38.9
   91.5  55.4  39.0  92.9  46.8  39.1  93.4  70.9  39.7
   55.8  50.5  38.1  96.2  84.3  40.3  98.2  58.2  39.5
;
```

Local kriging is performed by using only data points within a specified radius of each grid point. In this example, a radius of 60,000 feet is used. Other choices involved in local kriging are the minimum and maximum number of data points in each neighborhood (around a grid point). The minimum number is left at the default value of 20; the maximum number defaults to all observations in the data set within the specified radius.

The last step in contouring the data is to define the prediction grid point (node) locations. The prediction grid is typically rectangular, and you decide on the grid points population and spacing based on your available data in addition to your application needs. A convenient area that encompasses all the data points is a square of side length 100,000 feet. In the present analysis, a distance of 2,500 feet between nodes in the prediction grid is selected to obtain a smooth contour plot. Based on this choice, you obtain predictions on a square grid with 41 nodes on each side, which yields a total of 1681 grid points.

You can visualize the outcome of your analysis by using the PLOTS option in the PROC KRIGE2D statement. By default, PROC KRIGE2D produces one plot that displays the kriging prediction and its corresponding standard error at each output grid point. The locations of the Thick observations are displayed too, as outlines in the default plot. You can also ask for a plot of the thick data set observations and their values by specifying the OBSERV option in the PLOTS option.

The kriging analysis with the KRIGE2D procedure requires that you provide the prediction parameters in the PREDICT statement. You use the VAR= option to specify that you want to use the Thick variable in the kriging system, and the RADIUS= option to specify the radius of the local kriging regression. In this scenario you want to consider for your predictions all the neighboring data within a radius of 60,000 feet from each prediction location. You can specify more than one PREDICT statements; for example, you can do this when you want predictions for different variables in your DATA= data set.

The coordinates of your variable are specified in the COORDINATES statement. The MODEL statement contains the parameters that describe your data spatial correlation. Namely, the FORM= option specifies the model type, based on its mathematical form. The SCALE= and RANGE= options specify the model sill and range, respectively. You can specify more than one MODEL statement for the same PREDICT statement in order to obtain predictions based on different correlation models.

When you use the RADIUS= option to perform local kriging, as in the present example, it is suggested that the radius parameter be at least as large as your model range, so that you include data points that can contribute to your prediction.

Eventually, you specify the region of predictions with the GRID statement. The following SAS statements compute the kriged surface by using the preceding options and grid choice:

```
ods graphics on;

proc krige2d data=thick;
   coordinates xc=East yc=North;
   predict var=Thick radius=60;
   model scale=7.4599 range=30.1111 form=gauss;
   grid x=0 to 100 by 2.5 y=0 to 100 by 2.5;
run;

ods graphics off;
```

The table in Figure 74.1 shows the number of observations read and used in the kriging prediction. This table provides you with useful information in case you have missing values in the input data.

Figure 74.1 Number of Observations for the thick Data Set

Spatial Prediction With Kriging

The KRIGE2D Procedure

Dependent Variable: Thick

Number of Observations Read	75
Number of Observations Used	75

Figure 74.2 shows some general information about the kriging analysis. This includes the count of the output grid points. You have specified the RADIUS= option; therefore you also see that local kriging is requested.

Because this is a local analysis, the table also displays the parameters related to the neighborhood search around the grid points.

Figure 74.2 Kriging Analysis Information

Kriging Information	
Prediction Grid Points	1681
Type of Analysis	Local
Neighborhood Search Radius	60
Minimum Neighbors	20
Maximum Neighbors	All Within Radius

The covariance model parameters, including the effective range of the Gaussian model you specified, are shown in Figure 74.3.

Figure 74.3 Kriging Covariance Model Information

Spatial Prediction With Kriging

The KRIGE2D Procedure

Dependent Variable: Thick
Prediction: Pred1, Model: Model1

Covariance Model Information	
Type	Gaussian
Sill	7.4599
Range	30.1111
Effective Range	52.153955
Nugget Effect	0

Figure 74.4 shows a map of the kriging prediction contours based on the Thick observations in the specified spatial domain. The prediction error is displayed as a surface in the background.

Figure 74.4 Contour Plot of Kriged Coal Seam Thickness

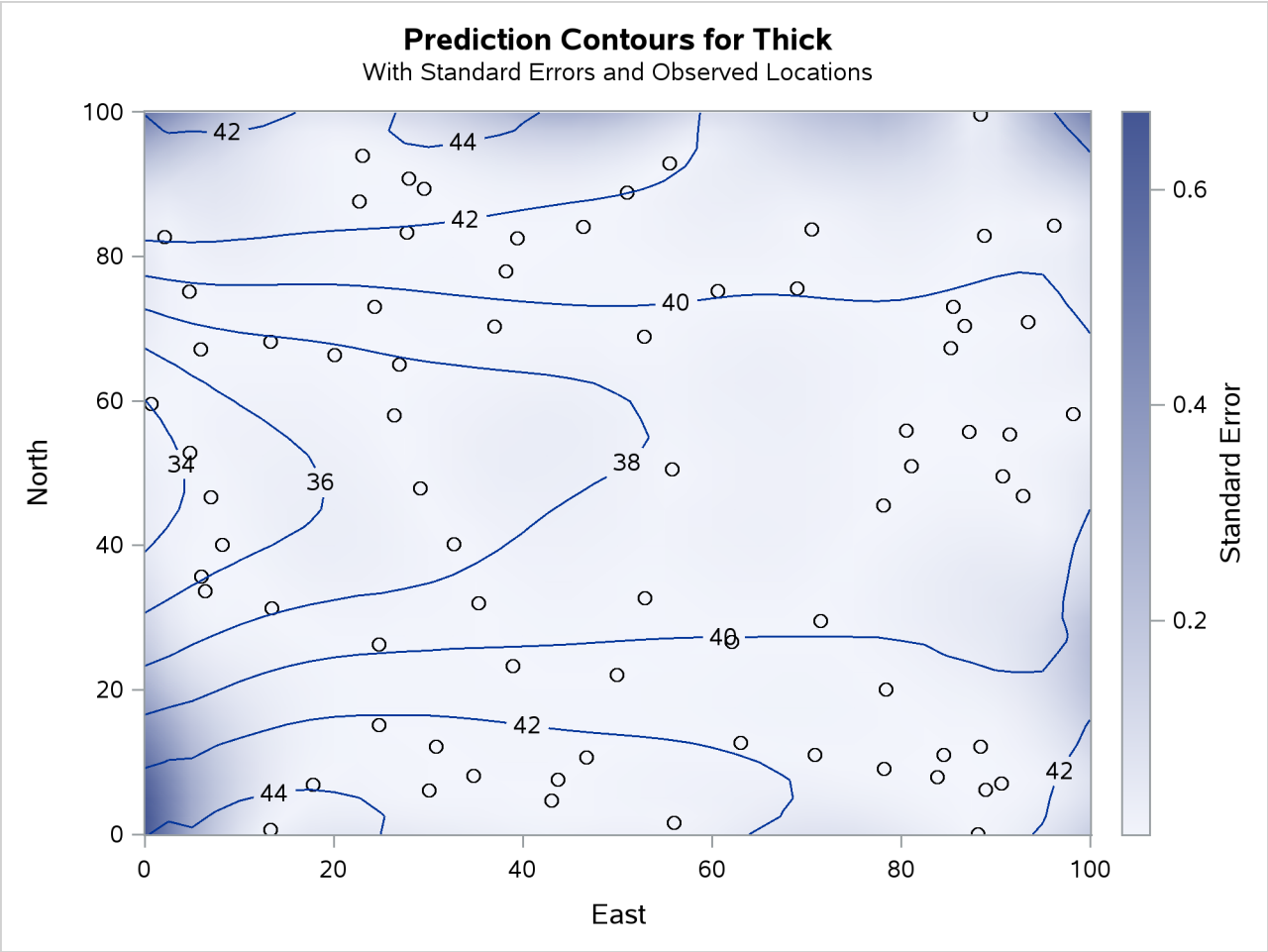

Note the locations of the observed data in Figure 74.4. The figure suggests that the Thick sampling locations are not ideally spread around the prediction area; however, there are no extended areas lacking measurements.

Based on the spatial distribution of the Thick data and the range r_ϵ of your covariance model, you can roughly see that for each prediction location there are at least several neighboring data points that contribute to the prediction value. Except perhaps for the nodes close to the boundaries of the prediction grid, you can then expect the prediction errors to be reasonably low compared to the predicted Thick values.

The kriging outcome in Figure 74.4 indicates that the standard errors are smaller in the neighborhoods where data are available. The size of these neighborhoods depends on the range of the specified covariance model that characterizes the spatial continuity of the domain, and on the prediction radius, if one is specified as in this example. The standard errors tend to increase toward the borders of the prediction area, beyond which no observations are available.

Syntax: KRIGE2D Procedure

The following statements are available in the KRIGE2D procedure:

PROC KRIGE2D *options* **;**
  BY *variables* **;**
  COORDINATES | COORD *coordinate-variables* **;**
  GRID *grid-options* **;**
  ID *variable* **;**
  PREDICT | PRED | P *predict-options* **;**
  MODEL *model-options* **;**
  RESTORE *restore-options* **;**

The PREDICT and MODEL statements are hierarchical; the PREDICT statement is followed by a MODEL statement. If more than one MODEL statement is given, only the last one is used for the analysis. The MODEL statement following a PREDICT statement uses the variable and neighborhood specifications in that PREDICT statement.

You must specify at least one PREDICT statement and one MODEL statement. You must supply a single COORDINATES statement to identify the *x* and *y* coordinate variables in the input data set. You must also specify a single GRID statement to include the grid information.

Table 74.1 outlines the *options* available in PROC KRIGE2D classified by function.

Table 74.1 Options Available in the KRIGE2D Procedure

Task	Statement	Option
Data Set Options		
Specifies input data set	PROC KRIGE2D	DATA=
Specifies grid data set	GRID	GDATA=
Specifies labels for individual grid points or in 1-D	GRID	LABEL
Specifies model data set	MODEL	MDATA=
Writes kriging predictions and standard errors	PROC KRIGE2D	OUTEST=
Writes neighborhood information for each grid point	PROC KRIGE2D	OUTNBHD=
Specifies plot display and options	PROC KRIGE2D	PLOTS
Declaring the Role of Variables		
Specifies variables to define analysis subgroups	BY	
Specifies variable with observation labels	ID	
Specifies the variables to be predicted (kriged)	PREDICT	VAR=
Specifies the *x* and *y* coordinate variables in the DATA= data set	COORDINATES	XC= YC=
Specifies the *x* and *y* coordinate variables in the GDATA= data set	GRID	XC= YC=
Controlling the Prediction		
Specifies the number of grid points in one-dimensional cases	GRID	NPTS=
Controlling Kriging Neighborhoods		
Specifies the radius of a neighborhood for all grid points	PREDICT	RADIUS=

Table 74.1 *continued*

Task	Statement	Option
Specifies the number of neighbors for all grid points	PREDICT	NUMPOINTS=
Specifies the maximum of neighbors for all grid points	PREDICT	MAXPOINTS=
Specifies the minimum of neighbors for all grid points	PREDICT	MINPOINTS=
Specifies the action when maximum not met	PREDICT	NODECREMENT
Specifies the action when minimum not met	PREDICT	NOINCREMENT
Controlling the Semivariogram Model		
Specifies an angle for an anisotropic model	MODEL	ANGLE=
Specifies a type with a functional form	MODEL	FORM=
Specifies an item store with correlation information	RESTORE	IN=
Specifies a nugget effect	MODEL	NUGGET=
Allows power exponent values outside [0,2)	MODEL	POWNOBOUND
Specifies a range parameter	MODEL	RANGE=
Specifies a minor-major axis ratio for an anisotropic model	MODEL	RATIO=
Specifies a scale parameter	MODEL	SCALE=
Specifies model and parameters from an item store	MODEL	STORESELECT

PROC KRIGE2D Statement

PROC KRIGE2D *options* **;**

The PROC KRIGE2D statement invokes the KRIGE2D procedure. Table 74.2 summarizes the *options* available in the PROC KRIGE2D statement.

Table 74.2 PROC KRIGE2D Statement Options

Option	Description
DATA=	Specifies input data set
IDGLOBAL	Uses ascending observation numbers as observation labels across BY groups
IDNUM	Uses observation number as observation labels
NOPRINT	Suppresses the normal display of results
OUTEST=	Writes kriging predictions and standard errors
OUTNBHD=	Writes neighborhood information for each grid point
PLOTS	Specifies plot display and options
PREDICTION	Produces the kriging prediction plot
SEMIVARIOGRAM	Produces the semivariogram used for the kriging prediction
SINGULARMSG=	Controls the number of warning messages displayed for a singular matrix

You can specify the following *options* in the PROC KRIGE2D statement.

DATA=*SAS-data-set*

specifies a SAS data set that contains the *x* and *y* coordinate variables and the VAR= variables in the PREDICT statement.

IDGLOBAL

specifies that ascending observation numbers be used across BY groups for the observation labels in the appropriate output data sets and the OBSERVATIONS plot, instead of resetting the observation number in the beginning of each BY group. The IDGLOBAL option is ignored if no BY variables are specified. Also, if you specify the ID statement, then the IDGLOBAL option is ignored unless you also specify the IDNUM option in the PROC KRIGE2D statement.

IDNUM

specifies that the observation number be used for the observation labels in the appropriate output data sets and the OBSERVATIONS plot. The IDNUM option takes effect when you specify the ID statement; otherwise, it is ignored.

NOPRINT

suppresses the normal display of results. The NOPRINT option is useful when you want only to create one or more output data sets with the procedure. **NOTE:** This option temporarily disables the Output Delivery System (ODS); see the section "ODS Graphics" on page 5601 for more information.

OUTEST=*SAS-data-set*

OUTE=*SAS-data-set*

specifies a SAS data set in which to store the kriging predictions, standard errors, and grid location. For details, see the section "OUTEST=*SAS-data-set*" on page 5598.

OUTNBHD=*SAS-data-set*

OUTN=*SAS-data-set*

specifies a SAS data set in which to store the neighborhood information for each grid point. Information is written to this data set only if one or more PREDICT statements have options that specify local kriging. For details, see the section "OUTNBHD=*SAS-data-set*" on page 5599.

PLOTS <(*global-plot-option***)> <=** *plot-request* **<(***options***)>>**

PLOTS <(*global-plot-option***)> <= (***plot-request* **<(***options***)> <...** *plot-request* **<(***options***)>>)>**

controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=none
plots=observ
plots=(observ(outl) prediction)
plots=(prediction(fill=pred line=se obs=grad) prediction(fill=se))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc krige2d data=thick;
   coordinates xc=East yc=North;
   predict var=thick r=60;
```

```
    model scale=7.4599 range=30.1111 form=gauss;
    grid x=0 to 100 by 10 y=0 to 100 by 10;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you omitted the PLOTS option or have specified PLOTS=ALL, then PROC KRIGE2D produces a default plot for each MODEL statement of every PREDICT statement that you specify. The default PROC KRIGE2D plot displays a contour plot of the kriging prediction and the gradient of the kriging prediction standard error at every location of the prediction grid, in addition to empty circles that indicate the observation locations. See Figure 74.4 for an example of the default KRIGE2D plot.

The following *global-plot-option* is available:

ONLY
suppresses the default plot. Only plots that are specifically requested are displayed.

The following individual *plot-requests* and plot *options* are available:

ALL
produces all appropriate plots. You can specify other *options* with ALL. For example, to request the default plot and an additional plot of the predictions, specify PLOTS=(ALL PREDICTION).

EQUATE
specifies that all appropriate plots be produced in a way that the axes coordinates have equal size units.

NONE
suppresses all plots.

OBSERVATIONS < (*observations-plot-options*) >
OBSERV < (*observations-plot-options*) >
OBS < (*observations-plot-options*) >
produces the observed data plot. Only one observations plot is created if you specify the OBSERVATIONS option more than once within a PLOTS option.

The OBSERVATIONS option has the following suboptions:

GRADIENT
specifies that observations be displayed as circles colored by the observed measurement.

LABEL < (*label-option*) >
labels the observations. The label is the ID variable if the ID statement is specified; otherwise, it is the observation number. The *label-option* can be one of the following:

EQ=*number*
specifies that labels show for any observation whose value is equal to the specified *number*.

MAX=*number*
specifies that labels show for observations with values smaller than or equal to the specified *number*.

MIN=*number*
specifies that labels show for observations with values equal to or greater than the specified *number*.

If you specify multiple instances of the OBSERVATIONS option and you specify the LABEL suboption in any of those, then the resulting observations plot displays the observations labels. If more than one *label-option* is specified in multiple LABEL suboptions, then the prevailing *label-option* in the resulting OBSERVATIONS plot emerges by adhering to the choosing order: MIN, MAX, EQ.

OUTLINE
specifies that observations be displayed as circles with a border but with a completely transparent fill.

OUTLINEGRADIENT
is the same as OBSERVATIONS(GRADIENT) except that a border is shown around each observation.

SHOWMISSING
specifies that observations with missing values be displayed in addition to the observations with nonmissing values. By default, missing values locations are not shown on the plot. If you specify multiple instances of the OBSERVATIONS option and you specify the SHOWMISSING suboption in any of those, then the resulting observations plot displays the observations with missing values.

If you omit any of the GRADIENT, OUTLINE, and OUTLINEGRADIENT suboptions, the OUTLINEGRADIENT is the default suboption. If you specify multiple instances of the OBSERVATIONS option or multiple suboptions for OBSERVATIONS, then the resulting observations plot honors the last specified GRADIENT, OUTLINE, or OUTLINEGRADIENT suboption.

PREDICTION < (*prediction-plot-options***) >**
PRED < (*prediction-plot-options***) >**
specifies that the kriging prediction plot be produced. You can specify the PREDICTION option multiple times in the same PLOTS option to request instances of plots with the following *prediction-plot-options*:

ALPHA=*number*
specifies a parameter to obtain the confidence level for constructing confidence limits based on the prediction standard error. The value of *number* must be between 0 and 1, and the confidence level is $1-$*number*. The default is ALPHA=0.05; this corresponds to the confidence level of 95%, or about 1.96 times the prediction standard error. The ALPHA= suboption is used only for prediction plots in one dimension, and it is incompatible with the FILL and LINE suboptions.

CLONLY

specifies that only the confidence limits be shown in a prediction plot without the predicted values. This suboption can be useful for identifying confidence limits when the prediction standard error is small at the prediction locations. CLONLY is used only for prediction plots in one dimension, and it is incompatible with the FILL and LINE suboptions.

CONNP

specifies that grid points that you provide as individual prediction locations be connected with a line on the area map. This suboption is ignored when you have a single grid point, a prediction grid in two dimensions, or when you also specify the NOMAP suboption. The CONNP suboption is incompatible with the FILL and LINE suboptions.

FILL=NONE | PRED | SE

produces a surface plot for either the predicted values or the standard errors. FILL=SE is the default. However, if you omit the FILL suboption, the behavior depends on the LINE suboption as follows: If you specify LINE=NONE or entirely omit the LINE suboption, then the FILL suboption is set to its default value. If LINE=PRED or LINE=SE, then the FILL suboption is set to the same value as the LINE suboption.

LINE=NONE | PRED | SE

produces a contour line plot for either the predicted values or the standard errors. LINE=PRED is the default. However, if you omit the LINE suboption the behavior depends on the FILL suboption as follows: If you specify FILL=NONE or entirely omit the FILL suboption, then the LINE suboption is set to its default value. If FILL=PRED or FILL=SE, then the LINE suboption is set to the same value as the FILL suboption.

NOMAP

specifies that the prediction plot be produced without a map of the domain where you have observations. The NOMAP suboption is used in the case of prediction in one dimension or at individual points. It is incompatible with the FILL and LINE suboptions.

OBS=*obs-options*

produces an overlaid scatter plot of the observations in addition to the specified contour plots. The following *obs-options* are available:

GRAD

specifies that observations be displayed as circles colored by the observed measurement. The same color gradient displays the prediction surface and the observations. Observations where the prediction is close to the observed values have similar colors—the greater the contrast between the color of an observation and the surface, the larger the prediction standard error is at that point.

LINEGRAD

is the same as OBS=GRAD except that a border is shown around each observation. This option is useful for identifying the location of observations where the standard errors are small, because at these points the color of the observations and the color of the surface are indistinguishable.

NONE
specifies that no observations be displayed.

OUTL
specifies that observations be displayed as circles with a border but with a completely transparent fill.

OBS=NONE is the default when you specify a grid in two dimensions, and OBS=LINEGRAD is the default used in the area map when you have a grid in one dimension. However, the default PROC KRIGE2D plot for a surface grid displays the observations locations as outlines.

SHOWD
specifies that the horizontal axis in scatter plots of linear prediction grids show the distance between grid points instead of the grid points' coordinates. When the area map is displayed, the prediction locations are also connected with a line. In all other grid configurations the SHOWD suboption is ignored, and it is incompatible with the FILL and LINE suboptions.

SHOWP
specifies that the grid points in band plots of linear prediction grids be shown as marks on the band plot. In all other grid configurations the SHOWP suboption is ignored, and it is incompatible with the FILL and LINE suboptions.

TYPE=BAND | SCAT
requests a particular type of plot when you have a linear grid, regardless of the default PREDICTION plot behavior in this case. The TYPE suboption is incompatible with the FILL and LINE suboptions.

If you specify multiple instances of the ALPHA, FILL, LINE, OBS, or TYPE suboptions in the same PREDICTION option, then the resulting predictions plot honors the last value specified for any of the suboptions. Any combination where you specify FILL=NONE and LINE=NONE is not available. When the prediction grid is in two dimensions, only the FILL, LINE, and OBS suboptions apply. If you specify incompatible suboptions in the same PREDICTION plot, then the plot instance is skipped.

The PREDICTION option produces a surface or contour line plot for grids in two dimensions and a band plot or scatter plot with error bars for grids in one dimension or individual points. In two dimensions the plot illustrates the predicted values and prediction error at each grid point. By default, when you specify a linear grid with fewer than 10 points, PROC KRIGE2D produces a PREDICTION scatter plot for each one of the prediction grid points. For 10 or more points in a linear grid, the PREDICTION plot is a band plot of the predicted means and the confidence limits at the 95% confidence level. You can override the default behavior in linear grids with the TYPE suboption. Prediction at individual locations always produces a PREDICTION scatter plot.

In cases of prediction in one dimension or at individual points, an area map is produced that shows the observations and the grid points. Band plots of linear grids display the grid points as a line on the map. When you specify individual prediction locations, the grid points are indicated with marks on the area map. The area map appears on the side of the prediction band plot or scatter plot, unless you specify the NOMAP suboption. You can also label the individual grid points or the ends of linear grid segments with the LABEL option of the GRID statement.

SEMIVARIOGRAM < (*semivar-plot-option*) >

SEMIVAR < (*semivar-plot-option*) >

specifies that the semivariogram used for the kriging prediction be produced. You can use the following *semivar-plot-option*:

MAXD=*number*

specifies a positive value for the upper limit of the semivariogram horizontal axis of distance. The SEMIVARIOGRAM plot extends by default to a distance that depends on the correlation model range. You can use the MAXD= option to adjust the default maximum distance value for the plot.

The SEMIVARIOGRAM option produces a plot for each correlation model that you specify for your prediction tasks. In an anisotropic case, the plot is not produced if you assign different anisotropy angles for different model components. The only exception is when you specify zonal components at right angles with the nonzonal model components. Also, the SEMIVARIOGRAM option is ignored for models that consist of purely zonal components.

SINGULARMSG=*number*

SMSG=*number*

controls the number of warning messages displayed for a singular matrix. When local kriging is performed, a separate kriging system is solved for each grid point. Anytime a singular matrix is encountered, a warning message is displayed up to a total of *number* times. The default is SINGULARMSG=10.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC KRIGE2D to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the KRIGE2D procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

COORDINATES Statement

COORDINATES | COORD *coordinate-variables* **;**

The following two options specify the names of the variables in the DATA= data set that contains the values of the x and y coordinates of the data.

Only one COORDINATES statement is allowed, and it is applied to all PREDICT statements. In other words, it is assumed that all the VAR= variables in all PREDICT statements have the same x and y coordinates.

This is not a limitation. Since each VAR= variable is processed separately, observations for which the current VAR= variable is missing are excluded. With the next VAR= variable, the entire data are read again, this time excluding missing values in this next variable. Hence, a single run of PROC KRIGE2D can be used for variables measured at different locations without overlap.

XCOORD= (*variable-name***)**

XC= (*variable-name***)**

specifies the name of the variable that contains the x coordinate of the data locations in the DATA= data set.

YCOORD= (*variable-name***)**

YC= (*variable-name***)**

specifies the name of the variable that contains the y coordinate of the data locations in the DATA= data set.

GRID Statement

GRID *grid-options* < / *option* > **;**

The GRID statement specifies the grid of spatial locations for kriging predictions. The grid specification is applied to all PREDICT and MODEL statements. Specify the grid in one of the following three ways:

- Specify the x and y coordinates explicitly for a grid in two dimensions.
- Specify the NPTS= option in addition to the x and y coordinates to define a grid of individual points or in one dimension.
- Specify the coordinates by using a SAS data set for a grid of individual points or in one dimension.

The GRID statement has the following *grid-options*:

NPTS=*number* **| ALL**

controls specification of a grid in one dimension or a grid of individual prediction locations.

When you specify the NPTS=*number* option and the coordinates of two points in the GRIDDATA= data set or in both the X= and Y= options, you request a linear prediction grid. Its direction is across the line defined by the specified points. The grid size is equal to the *number* of points that you specify in the NPTS= option, where *number* ≥ 2.

When you specify the NPTS=ALL option and the coordinates for any number of points in the GRIDDATA= data set or in each of the X= and Y= options, the KRIGE2D procedure performs prediction only at the specified individual locations. Use the NPTS=ALL option to examine a set of individual points anywhere on the XY plane or to specify a custom grid in one dimension.

If the number of x coordinates and the number of y coordinates in the X= and Y= options, respectively, are different, then the NPTS= option is ignored; in that case, a two-dimensional grid is used according to the specified X= and Y= options.

If you specify a prediction grid with any number of points other than two in the GRIDDATA= data set, then the option NPTS=ALL has the same effect as omitting the NPTS= option.

X=*number*
X=$x_1, \ldots, x_m$
X=x_1 **TO** x_m
X=x_1 **TO** x_m **BY** δx
specifies the x coordinate of the grid locations.

Y=*number*
Y=$y_1, \ldots, y_m$
Y=y_1 **TO** y_m
Y=y_1 **TO** y_m **BY** δy
specifies the y coordinate of the grid locations.

Use the X= and Y= options of the GRID statement to specify a grid in one or two dimensions, or a grid of individual prediction locations.

For example, the following two GRID statements are equivalent.

```
grid x=1,2,3,4,5  y=0,2,4,6,8,10;

grid x=1 to 5     y=0 to 10 by 2;
```

In the following example, the first GRID statement produces a grid in two dimensions. The second statement produces predictions only for the four individual points at the locations (1,0), (2,5), (3,7), and (4,10) on the XY plane.

```
grid x=1 to 4 y=0,5,7,10;

grid x=1 to 4 y=0,5,7,10 npts=all;
```

In the next example, the first GRID statement specifies a 2-by-2 grid in two dimensions. The second GRID statement specifies a linear grid of eight points. The grid is in the direction of the line defined by the specified points (2,8) and (3,5) on the XY plane and it extends between these two points.

```
grid x=2,3 y=8,5;

grid x=2,3 y=8,5 npts=8;
```

The last example shows a GRID statement that specifies a linear grid made of seven points across the Y axis. In this case, the syntax is sufficient to fully define a linear grid without the NPTS= option.

```
grid x=5 y=3 to 9;
```

To specify grid locations from a SAS data set, you must provide the name of the data set and the variables that contain the values of the *x* and *y* coordinates.

GRIDDATA=*SAS-data-set*

GDATA=*SAS-data-set*

specifies a SAS data set that contains the *x* and *y* grid coordinates. Use the GRIDDATA= option of the GRID statement to specify a grid in one dimension or a grid of individual prediction locations.

XCOORD= (*variable-name***)**

XC= (*variable-name***)**

specifies the name of the variable that contains the *x* coordinate of the grid locations in the GRIDDATA= data set.

YCOORD= (*variable-name***)**

YC= (*variable-name***)**

specifies the name of the variable that contains the *y* coordinate of the grid locations in the GRIDDATA= data set.

You can specify the following *option* in the GRID statement after a slash (/):

LABEL < (*suboption***) > = (***character-list***)**

specifies labels to tag grid points in prediction plots when you use grids in one dimension. You can specify one or more such labels as quoted strings in the *character-list*.

When the number of labels in the *character-list* exceeds the number of points in your grid, the labels in the list are used sequentially and any labels in excess are ignored. When the number of labels in the *character-list* is smaller than the number of points in your grid, the behavior is as follows:

- If an area map is included in the prediction plot, then blank labels are assigned to the remaining nonlabeled grid points on the map.
- For the prediction band and scatter plots, the coordinates of nonlabeled grid points are automatically assigned as their labels.

If the grid points are collinear and the horizontal axis displays distance, then two labels appear by default in the prediction plot. These are assigned to the first and the last points of the grid to help identify the ends of the linear grid segment on the plot map. This label pair is shown only when the plot includes an area map. Specifically, the two labels appear when you request prediction band plots, or prediction scatter plots for which you specify the PREDICTION(SHOWD) suboption, if applicable. The two labels do not appear if you specify explicitly the NOMAP suboption in the PLOTS=PRED option.

The two labels have default values, unless you choose to specify your own labels with the LABEL= option. If you specify more than two labels in the *character-list* under these conditions, then only the first and last labels in the list are used; any additional labels in between are ignored.

The LABEL= option has the following *suboption*:

ALL

specifies that all individual points in the grid be assigned sequentially the labels you specify in the LABEL(ALL)= option when the PREDICTION(SHOWD) suboption is applicable and specified in a prediction scatter plot. In all other cases, the ALL suboption is ignored.

The ALL suboption enables you to override the default behavior when the PREDICTION(SHOWD) suboption is specified (the default behavior is to display labels only for the first and last grid points). As a result, you can use the ALL suboption to label grid points regardless of whether you specify the NOMAP suboption in the PLOTS=PRED option.

The LABEL= option is ignored when you produce prediction plots of grids in two dimensions.

ID Statement

ID *variable* **;**

The ID statement specifies which variable to include for identification of the observations in the OUTNBHD= output data set. The ID statement variable is also used for the labels and tool tips in the OBSERVATIONS plot and the tool tips in the PREDICTION plot.

In the KRIGE2D procedure you can specify only one ID variable in the ID statement. If no ID statement is given, then PROC KRIGE2D uses the observation number in the data sets and the plots.

PREDICT Statement

PREDICT | PRED | P *predict-options* **;**

You can specify the following *options* in a PREDICT statement.

MAXPOINTS=*number*

MAXP=*number*

MAX=*number*

specifies the maximum number of data points in a neighborhood. You specify this option in conjunction with the RADIUS= option. When the number of data points in the neighborhood formed at a given grid point by the RADIUS= option is greater than the MAXPOINTS= value, the RADIUS= value is decreased just enough to honor the MAXPOINTS= value unless you specify the NODECREMENT option. The default is to include all data points within the specified RADIUS= value. Neighborhoods with very large numbers of data points might lead to unnecessarily slow execution times and potential lack of memory issues, depending on the problem setup and your computational resources. In that case, you could use the MAXPOINTS= option to set a cap for your neighborhood size. For details about numerical considerations, see the section "Computational Resources" on page 5598. Unless the RADIUS= option is also specified, when the MAXPOINTS= and NUMPOINTS= options are specified in the same PREDICT statement the MAXPOINTS= option is ignored.

MINPOINTS=*number*

MINP=*number*

MIN=*number*

specifies the minimum number of data points in a neighborhood. You specify this option in conjunction with the RADIUS= option. When the number of data points in the neighborhood formed at a given grid point by the RADIUS= option is less than the MINPOINTS= value, the RADIUS= value is increased just enough to honor the MINPOINTS= value unless you specify the NOINCREMENT option. The default is MINPOINTS=20. When enough data are available, you might improve prediction if you increase this value. When the MINPOINTS= and NUMPOINTS= options are specified in the same PREDICT statement, the MINPOINTS= option is set to the value of NUMPOINTS=.

NODECREMENT | NODECR

requests that the RADIUS= value not be decremented when the MAXPOINTS= value is exceeded at a grid point. This option is relevant only when you specify both a RADIUS= value and a MAXPOINTS= value. In this case, when the number of points in the neighborhood constructed from the RADIUS= specification is greater than the MAXPOINTS= value, the RADIUS= value is decremented enough to honor the MAXPOINTS= value, and the kriging system is solved for this grid point. If you specify the NODECREMENT option, no decrementing is done, prediction is skipped at this grid point, and a message is written to the log.

NOINCREMENT | NOINCR

requests that the RADIUS= value not be incremented when the MINPOINTS= value is not met at a grid point. This option is relevant only when you specify both a RADIUS= value and a MINPOINTS= number. In this case, when the number of points in the neighborhood constructed from the RADIUS= specification is less than the MINPOINTS= value, the RADIUS= value is incremented enough to honor the MINPOINTS= value, and the kriging system is solved for this grid point. If you specify the NOINCREMENT option, no incrementing is done, prediction is skipped at this grid point, and a message is written to the log.

NUMPOINTS=*number*

NPOINTS=*number*

NPTS=*number*

NP=*number*

specifies the exact size of a neighborhood. This option is incompatible with all other PREDICT statement options that control the neighborhood; it must appear by itself. In particular, if you specify both NUMPOINTS= and the RADIUS= option in the same PREDICT statement, then RADIUS= is honored, instead. In this event the value of the MINPOINTS= option is set to NUMPOINTS=, and the value of the MAXPOINTS= option is set to default, regardless of whether these options have been specified or not. If you specify any of the MINPOINTS= or MAXPOINTS= option without the RADIUS= option in the same PREDICT statement as NUMPOINTS=, then the NUMPOINTS= option is honored.

RADIUS=*number*

R=*number*

specifies the radius to use in a local kriging regression. When you specify this option, a separate kriging system is solved at each grid point by finding the neighborhood of this grid point that consists of all data points within the distance specified by the RADIUS= value. Thus, you can avoid unnecessary computational burden in your analysis by specifying the RADIUS= value to include data points situated

within the extent of your problem's spatial correlation. For additional control on the neighborhood, see the MAXPOINTS= and MINPOINTS= options. When you specify the RADIUS= and NUMPOINTS= options in the same PREDICT statement, then RADIUS= is honored.

VAR= *variable-name*
specifies the single numeric variable used in the kriging system.

MODEL Statement

MODEL *model-options* **;**

The MODEL statement specifies details about the correlation model that you use in the kriging system for prediction. The specified model is used in the kriging system defined by the most previous PREDICT statement. You can specify a semivariogram or covariance model in three ways:

- You specify the required parameters SCALE, RANGE, FORM, and SMOOTH (if you specify the MATERN form), and possibly the optional parameters NUGGET, ANGLE, and RATIO, explicitly in the MODEL statement.
- You specify an MDATA= data set. This data set contains variables that correspond to the required parameters SCALE, RANGE, FORM and SMOOTH (if you specify the MATERN form), and optionally variables for the NUGGET, ANGLE, and RATIO parameters.
- You can specify an input item store in the RESTORE statement. The item store contains one or more correlation models for one or more direction angles. You can specify these models in the STORESELECT option of the MODEL statement to perform a prediction task.

The three methods are mutually exclusive: you specify all parameters explicitly, they are all are read from the MDATA= data set, or you select a model and its parameters from an input item store.

Table 74.3 summarizes the *options* available in the MODEL statement.

Table 74.3 MODEL Statement Options

Option	Description
ANGLE=	Specifies the angle of the major axis
FORM=	Specifies the functional form (type)
MDATA=	Specifies the input data set containing parameter values
NUGGET=	Specifies the nugget effect for the model
POWNOBOUND	Allows values for the power model exponent parameter outside the range of $[0, 2)$
RANGE=	Specifies the range parameter
RATIO=	Specifies the ratio of the length of the minor axis
SCALE=	Specifies the scale parameter
SINGULAR=	Gives the singularity criterion for solving kriging systems
SMOOTH=	Specifies the smoothness parameter
STORESELECT	Uses the information from an input item store

You can use the following *model-options* with the MODEL statement:

ANGLE=*angle* | (*angle1, . . . , anglek*)
specifies the angle of the major axis for anisotropic models, measured in degrees clockwise from the N-S axis. The default is ANGLE=0.

In the case of a nested semivariogram model with k nestings, you have the following two ways to specify the anisotropy major axis: you can specify only one *angle* which is then applied to all nested forms, or you can specify one angle for each of the k nestings.

NOTE: The syntax makes it possible to specify different angles for different forms of the nested model, but this practice is rarely used.

FORM=*form* | (*form1, . . . , formk*)
specifies the functional form (type) of the semivariogram model. Use the syntax with the single *form* to specify a non-nested model. Use the syntax with forms *formi*, $i = 1, \ldots, k$, to specify a nested model with k structures. Each of the forms can be any of the following:

CUBIC | EXPONENTIAL | GAUSSIAN | MATERN |
PENTASPHERICAL | POWER | SINEHOLEEFFECT | SPHERICAL
CUB | EXP | GAU | MAT | PEN | POW | SHE | SPH
Usage examples follow.

For example, the syntax

```
FORM=GAU
```

specifies a model with a single Gaussian structure. Also, the syntax

```
FORM=(EXP,SHE,MAT)
```

specifies a nested model with an exponential, a sine hole effect, and a Matérn structure. Finally

```
FORM=(EXP,EXP)
```

specifies a nested model with two structures both of which are exponential.

NOTE: In the documentation, models are named either by using their full names or by using the first three letters of their structures. Also, the names of different structures in a nested model are separated by a hyphen (-). According to this convention, the previous examples illustrate how to specify a GAU, an EXP-SHE-MAT, and an EXP-EXP model, respectively, with the FORM= option.

All the supported model forms have two parameters specified by the SCALE= and RANGE= options, except for the MATERN model which has a third parameter specified by the SMOOTH= option. A FORM= value is required, unless you specify the MDATA= option or the STORESELECT option.

Computation of the MATERN covariance is numerically demanding. As a result, predictions that use Matérn covariance structures can be time-consuming.

See the section "Theoretical Semivariogram Models" on page 5577 for details about how the FORM= forms are determined.

MDATA=*SAS-data-set*

specifies the input data set that contains parameter values for the covariance or semivariogram model. The MDATA= option cannot be combined with any of the FORM= or STORESELECT options.

The MDATA= data set must contain variables named SCALE, RANGE, and FORM, and it can optionally contain variables NUGGET, ANGLE, and RATIO. If you specify the MATERN form, then you must also include a variable named SMOOTH in the MDATA= data set.

The FORM variable must be a character variable, and it can assume only the values allowed in the explicit FORM= syntax described previously. The RANGE, SCALE and SMOOTH variables must be numeric. The optional variables ANGLE, RATIO, and NUGGET must also be numeric if present.

The number of observations present in the MDATA= data set corresponds to the level of nesting of the covariance or semivariogram model. For example, to specify a non-nested model that uses a spherical covariance, an MDATA= data set might be given by the following statement:

```
data md1;
   input scale range form $;
   datalines;
   25 10 SPH
run;
```

The PROC KRIGE2D statement to use the MDATA= specification is of the form shown in the following:

```
proc krige2d data=...;
   predict var=....;
   model mdata=md1;
run;
```

This is equivalent to the following explicit specification of the covariance model parameters:

```
proc krige2d data=...;
   predict var=....;
   model scale=25 range=10 form=sph;
run;
```

The following MDATA= data set is an example of an anisotropic nested model:

```
data md1;
   input scale range form $ nugget angle ratio;
   datalines;
   20 8 SPH  5 35 0.7   .
   12 3 MAT  5 0  0.8 2.8
   4  1 GAU  5 45 0.5   .
;
```

This is equivalent to the following explicit specification of the covariance model parameters:

```
proc krige2d data=...;
   predict var=....;
   model scale=(20,12,4) range=(8,3,1) form=(SPH,MAT,GAU)
         angle=(35,0,45) ratio=(0.7,0.8,0.5) nugget=5 smooth=2.8;
run;
```

This example is somewhat artificial in that it is usually hard to detect different anisotropy directions and ratios for different nestings by using an empirical semivariogram. **NOTE:** The NUGGET variable value is the same for all nestings. This is always the case; the nugget effect is a single additive term for all models. For further details, see the section "The Nugget Effect" on page 5585.

The example also shows that if you specify a MATERN form in the nested model, then the SMOOTH variable must be specified for all nestings in the MDATA= data set. You simply specify the SMOOTH value as missing for nestings other than MATERN.

NUGGET=*number*

specifies the nugget effect for the model. The nugget effect is due to a discontinuity in the semivariogram as determined by plotting the sample semivariogram. For details, see the section "The Nugget Effect" on page 5585 and Chapter 131, "The VARIOGRAM Procedure." For models without any nugget effect, this option is left out; the default is NUGGET=0.

POWNOBOUND

specifies that values for the power model exponent parameter outside the range of $[0, 2)$ be allowed. The POWNOBOUND option applies only when you specify a power form in the MODEL statement.

Power models yield permissible covariance models only when the exponent parameter is nonnegative and less than 2. By default, PROC KRIGE2D produces an error if you specify a negative power exponent or one that is equal to or larger than 2 in the RANGE= option of the MODEL statement.

See the section "The Power Semivariogram Model" on page 5582 for more details about the power model form and its exponent parameter.

RANGE=*range* | (*range1, . . . , rangek*)

specifies the range parameter in semivariogram models. If you have anisotropy, you must specify the range of the major anisotropy axis, or the range of the minor anisotropy axis for any zonal components. In the case of a nested semivariogram model with k nestings, you must specify a range for each nested structure.

The range parameter has units of distance, and it is related to the correlation scale for the underlying spatial process.

NOTE: If you specify this parameter for a power model, then it does not correspond to a range. For power models, the parameter you specify in the RANGE option is a dimensionless power exponent whose value must range within [0,2) so that the power model is a valid semivariance function. See also the POWNOBOUND option of the MODEL statement.

See the section "Theoretical Semivariogram Models" on page 5577 for details about how the RANGE= values are determined.

RATIO=*ratio* | (*ratio1, . . . , ratiok*)

specifies the ratio of the length of the minor axis to the length of the major axis for anisotropic models. The value of the RATIO= option must be between 0 and 1. An exception is the case of zonal anisotropy, where the ratio of zonal components must be designated by a very large number for the RATIO= option. For further details, see the section "Zonal Anisotropy" on page 5590.

In the case of a nested semivariogram model with k nestings, you can specify a ratio for each nesting. The default is RATIO=1.

SCALE=*scale* | (*scale1, . . . , scalek*)

specifies the scale parameter in semivariogram models. In the case of a nested semivariogram model with k nestings, you must specify a scale for each nesting.

The scale parameter is the multiplicative factor in all supported models; it has the same units as the variance of the VAR= variable in the preceding PREDICT statement.

In power models the SCALE= parameter does not correspond to a sill because the power model has no sill. Instead, PROC KRIGE2D uses the SCALE= option to designate the slope (or scaling factor) in power model forms. The power model slope has the same variance units as the VAR= variable.

See the section "Theoretical Semivariogram Models" on page 5577 for details about how the SCALE= values are determined.

SINGULAR=*number*

gives the singularity criterion for solving kriging systems. The larger the value of the SINGULAR= option, the easier it is for a kriging system to be declared singular. The default is SINGULAR=1E–7. See the section "Ordinary Kriging" on page 5595 for more detailed information.

SMOOTH=*smooth* | (*smooth1, . . . , smoothm*)

specifies the smoothness parameter $\nu > 0$ in the Matérn type of semivariance structures. The special case $\nu = 0.5$ is equivalent to the exponential model, whereas $\nu \to \infty$ gives the Gaussian model.

When you specify m different MATERN forms in the FORM= option, you must also provide m smoothness values in the SMOOTH option. If you must specify more than one smoothness value, the values are assigned sequentially to the MATERN nestings in the order the nestings are specified. If you specify more smoothness values than necessary, then values in excess are ignored.

STORESELECT(*ssel-options***)**

SSEL(*ssel-options***)**

specifies that information from an input item store be used for the prediction. You cannot combine the STORESELECT option with any of the FORM= or MDATA= options. The STORESELECT option has the following *ssel-options*:

TYPE=*field-type*

specifies whether to perform isotropic or anisotropic prediction. You can choose the *field-type* from one of the following:

ISO
specifies an isotropic field for the prediction.

ANIGEO | GEO
specifies a field with geometric anisotropy for the prediction.

ANIZON(*zonal-form1, . . . , zonal-formn***)**
ZON(*zonal-form1, . . . , zonal-formn***)**
specifies a field with zonal anisotropy for the prediction. Each *zonal-formi*, $i = 1, \ldots, n$, can be any of the following:

CUB | EXP | GAU | MAT | PEN | POW | SHE | SPH
Each *zonal-formi*, $i = 1, \ldots, n$, is a structure in the purely zonal component of the correlation model in the direction angle of the minor anisotropy axis. For this reason, when you specify the TYPE=ANIZON suboption you must also specify the nonzonal component of the correlation model in the MODEL= suboption of the STORESELECT option.

Assume the nonzonal component has k structures; these are common across all directions and each one has the same scale in all directions. In that sense, you use the TYPE=ANIZON suboption to specify only the n zonal anisotropy structures of an input store $(k + n)$-structure nested model in the direction angle of the minor anisotropy axis.

Given this specification, $k+n$ must be up to the maximum number of nested model structures that is supported by the item store. See also the MODEL= suboption of the STORESELECT option.

In conclusion, you can use an input item store for prediction with zonal anisotropy if you know that every structure in the nonzonal model component has the same scale across all directions. When this condition does not apply for the item store models, specify the model parameters explicitly in the MODEL statement. For more details, see the examples in the section "Zonal Anisotropy" on page 5590.

Computation of the MATERN covariance is numerically demanding. As a result, predictions that use Matérn covariance structures can be time-consuming.

If you omit the TYPE= option, the default behavior is TYPE=ISO when the input item store contains information for only one angle or for the omnidirectional case. If you specify an item store with information for more than one direction, then the default behavior is TYPE=ANIGEO.

When you specify TYPE=ISO to request isotropic analysis in the presence of an item store with information for multiple directions, you must specify the ANGLEID= suboption of the STORESELECT option with one argument. This argument specifies which of the direction angles information to use for the isotropic analysis.

When you indicate the presence of anisotropy with the TYPE=ANIGEO or TYPE=ANIZON suboptions of the STORESELECT option, the following conditions apply:

- You must specify the ANGLEID= suboption of the STORESELECT option to designate the major and minor anisotropy axes. See the ANGLEID= suboption of the STORESELECT option for details.
- – For TYPE=ANIGEO, ensure that you have the same scale in all anisotropy directions.

  - For TYPE=ANIZON, ensure that the nonzonal component scale is the same in all anisotropy directions.

  If you import a nested model, these rules also apply to each one of the nested structures.

- Model ranges in the major anisotropy axis must be longer than ranges in the minor anisotropy axis.
- Any Matérn covariance structure must maintain its smoothness parameter value in all anisotropy directions.

ANGLEID=*angleid1* | (*angleid1, angleid2*)

specifies which direction angles in the input item store be used for prediction. The angles are identified by the corresponding number in the AngleID column of the "Store Models Information" table, or by the AngleID parameter in the table title when you specify the INFO(DETAILS) option in the RESTORE statement.

If you request isotropic prediction in the TYPE= suboption of the STORESELECT option and the item store has omnidirectional contents or information about only one angle, then the ANGLEID= option is ignored. The prediction input comes from the omnidirectional information. In the case of a single angle, you still perform isotropic prediction and the model parameters are provided by the model in the single direction angle in the item store. However, if the item store contains information for more than one angle, then you must specify one angle ID in *angleid1*. The model information from the corresponding angle is then used in your isotropic prediction.

When you specify an anisotropic prediction in the TYPE= option of the STORESELECT option, you need to have information about two perpendicular direction angles. One of them is the major and the other is the minor anisotropy axis. You must always specify the major anisotropy axis angle ID in *angleid1* and the minor anisotropy axis angle ID in *angleid2*. This means that the range parameters of the model forms in the angle designated by the *angleid1* need to be larger than the corresponding ranges of the forms in the angle designated by the *angleid2*. Conveniently, if the item store has only two angles, then you only need to specify the ID *angleid1* of the major anisotropy axis angle. If the item store has only one angle, then you cannot perform anisotropic prediction with input from the item store.

NOTE: You can perform geometric anisotropic analysis even if the item store does not contain information about a direction that is perpendicular to the one specified by *angleid1*. This is possible due to the geometry of the ellipse. In particular, when you specify the major axis with *angleid1* and an angle ID for a second direction with a corresponding smaller range, then PROC KRIGE2D automatically computes the minor anisotropy axis range and the necessary range ratio parameter.

Anisotropic analysis is not possible when you specify instances of the same angle in the input item store. It is possible that PROC VARIOGRAM produces an item store in which two or more directions can be the same if their corresponding correlation models were obtained for different angle tolerances or bandwidths in the VARIOGRAM procedure. Consequently, you cannot specify anisotropic prediction if the input store contains only two angles that are the same or if you specify *angleid1* and *angleid2* that correspond to equal angles.

MODEL=*form* | (*form1, . . . , formk*)

specifies the theoretical semivariogram model selection to use for the prediction. Use any combination of one, two, or three forms to describe a model in the input item store because up to three nested structures are supported. Each *formi*, $i = 1, \ldots, k$, can be any of the following:

CUB | EXP | GAU | MAT | PEN | POW | SHE | SPH

Computation of the MATERN covariance is numerically demanding. As a result, predictions that use Matérn covariance structures can be time-consuming.

All fitted models that are stored in the input item store contain information about their component parameters and also about the nugget effect if any. The KRIGE2D procedure retrieves this information when you make a model selection in the MODEL= option, and you do not need to individually specify a nugget effect or any other parameter of the model.

By default, the model that is ranked first among the models for a given angle in the item store is used for the prediction task. If more than one model is available in the item store, then you can specify the MODEL= option to use a different model for the prediction.

In an anisotropic prediction, the default selection is the model that is ranked first in the direction angle of the major anisotropy axis. If you specify the TYPE=ANIGEO option, then a model that consists of identical structures needs to be present in the selected minor anisotropy axis angle in the item store. If you specify the TYPE=ANIZON option, then a model with the exact same first k structures must be present in the selected minor anisotropy axis angle, and it must feature at least one more structure as a zonal component. The zonal component is specified separately in the TYPE=ANIZON suboption of the STORESELECT option. Consequently, remember that in zonal anisotropy the MODEL= suboption designates only the nonzonal component of the correlation model in the minor anisotropy axis direction. In all, if there are k common structures and n structures in the purely zonal component, then $k + n$ must be up to the maximum number of nested model structures that is supported by the item store.

In comparison to the other two ways of specifying a correlation model in PROC KRIGE2D, the STORESELECT option is quite different because you can avoid explicit specification of all parameter values of a model. When you specify the STORESELECT option, then the corresponding scale, range, nugget effect, and smoothness (if appropriate) parameter values are invoked as saved attributes of the model that you select from the item store.

In the case of anisotropy, you specify the angles indirectly with the ANGLEID= option of the STORESELECT option, and the ratios are computed implicitly by using the selected model ranges. Explore how to specify valid anisotropical models imported from an input item store with the two examples that follow.

In the first example, assume the input item store InStoreGeo contains exponential models in the angles $\theta_1 = 0°$, $\theta_2 = 45°$, and $\theta_3 = 90°$. You know in advance that all models have the same scale $c_1 = c_2 = c_3$ across these directions and that the respective ranges are $a_1 = 15$, $a_2 = 20$, and $a_3 = 25$ in distance units. Hence, you have a case of geometric anisotropy where the major anisotropy axis is in the direction of angle θ_3 and the minor anisotropy axis is in the direction of angle θ_1. The following statements in PROC KRIGE2D use the information in the item store InStoreGeo to perform simulation under the assumption of geometric anisotropy:

```
proc krige2d data=...;
   restore in=InStoreGeo;
   predict var=....;
   model storeselect(model=exp type=anigeo angleid=(3,1));
run;
```

For the second example, assume a case of zonal anisotropy. Consider the input item store InStoreZon, which contains models in the two angles, $\theta_1 = 30°$ and $\theta_2 = 120°$. Specifically, in θ_1 you have an exponential-spherical model: the exponential structure has scale $c_{1E} = 3$ and range $a_{1E} = 10$; the spherical structure has scale $c_{1S} = 1$ and range $a_{1S} = 6$. In direction θ_2 you have an exponential model with scale $c_{1E} = 3$ and range $a_{1E} = 12$. Hence, the zonal anisotropy major axis is in the direction of the lowest total variance, which is in angle θ_2; then, the minor axis is in the direction of angle θ_1. The following statements in PROC KRIGE2D use the information in the store InStoreZon to perform prediction under the assumption of zonal anisotropy:

```
proc krige2d data=...;
   restore in=InStoreZon;
   predict var=....;
   model storeselect(model=exp type=anizon(sph) angleid=(2,1));
run;
```

RESTORE Statement

RESTORE IN=*store-name* < / *option* > **;**

The RESTORE statement specifies an item store that provides spatial correlation model input for the PROC KRIGE2D prediction tasks. An item store is a binary file defined by the SAS System. You cannot modify the contents of an item store. The KRIGE2D procedure can use only item stores created by PROC VARIOGRAM.

Item stores enable you to use saved correlation models without having to repeat specification of these models in the MODEL statement. In principle, an item store contains the chosen model from a model fitting process in PROC VARIOGRAM. If more than one model form is fitted, then all successful fits are included in the item store. In this case, you can choose any of the available models to use for prediction with the STORESELECT(MODEL=) option in the MODEL statement. Successfully fitted models might include questionable fits, which are so flagged when you specify the INFO option to display model names.

The *store-name* is a usual one- or two-level SAS name, as for SAS data sets. If you specify a one-level name, then the item store resides in the WORK library and is deleted at the end of the SAS session. Since item stores are often used for postprocessing tasks, typical usage specifies a two-level name of the form *libname.membername*.

When you specify the RESTORE statement, the default output contains some general information about the input item store. This information includes the store name, label (if assigned), the data set that was used to create the store, BY group information, the procedure that created the store, and the creation date.

You can specify the following *option* in the RESTORE statement after a slash (/):

INFO < (*info-options*) >

specifies that additional information about the input item store be printed. This information is provided in two ODS tables. One table displays the variables in the item store, in addition to the mean and standard deviation for each of them. These statistics are based on the observations that were used to produce the store results. The second table shows the model on top of the list of all fitted models for each direction angle in the item store. The INFO option has the following *info-options*:

DETAILS

DET

specifies that more detailed information be displayed about the input item store. This option produces the full list of models for each direction angle in the item store, in addition to the model equivalence class. For more information about classes of equivalence, see the section "Classes of Equivalence" on page 11034 in Chapter 131, "The VARIOGRAM Procedure." The DETAILS option is ignored if the input item store contains information about a single fitted model.

ONLY

specifies that only information about the input item store without any prediction tasks be displayed.

When you specify an input item store with the RESTORE statement in PROC KRIGE2D, all the DATA= input data set variables must match input item store variables. If there are BY groups in the input DATA= set or in the input RESTORE variables, then PROC KRIGE2D handles the different cases as follows:

- If both PROC KRIGE2D has BY groups and the RESTORE statement has BY groups, then the analysis variables must match. This matching assumes implicitly that in each BY group of PROC KRIGE2D and the item store, the corresponding set of observations and correlation model comes from the same random field. This assumption is valid if you use the same data set, first in PROC VARIOGRAM to fit a model and save it in the item store, and then in PROC KRIGE2D to perform predictions with the resulting correlation models.
- If PROC KRIGE2D has BY groups but the item store does not, then the item store is accepted only if the procedure and the item store analysis variables match. In this case, the same item store model choice iterates across the BY groups of the input data. You are advised to proceed with caution: each BY group in the input DATA= set corresponds to a different realization of a random field. Hence, by using the same correlation model for prediction purposes, you implicitly assume that all these different realizations are instances of the same random field.
- If PROC KRIGE2D has no BY groups but the item store does, then the item store is rejected.

Details: KRIGE2D Procedure

Theoretical Semivariogram Models

Consider a stochastic spatial process represented by the stationary spatial random field (SRF) $\{Z(\boldsymbol{s}), \boldsymbol{s} \in D \subset \mathcal{R}^2\}$ (Christakos 1992). The VARIOGRAM procedure computes the empirical (also known as sample or experimental) semivariance of $Z(\boldsymbol{s})$. Prediction of the spatial process $Z(\boldsymbol{s})$ at unsampled locations by techniques such as ordinary kriging requires a theoretical semivariogram or covariance.

When you use PROC VARIOGRAM and PROC KRIGE2D to perform spatial prediction, you must determine a suitable theoretical semivariogram based on the sample semivariogram. Various methods exist to fit semivariogram models, such as least squares, maximum likelihood, and robust methods (Cressie 1993, section 2.6). You can use PROC VARIOGRAM to perform automated fitting of a semivariogram model with weighted or ordinary least squares. A different approach is manual fitting, in which a theoretical semivariogram is chosen based on visual inspection of the empirical estimate; see, for example, Hohn (1988, p. 25).

In some cases, a plot of the experimental semivariogram suggests that a single theoretical model is inadequate. Nested models, anisotropic models, and the nugget effect increase the scope of theoretical models available. All of these concepts are discussed in this section. The specification of the final theoretical model is provided by the syntax of PROC KRIGE2D.

Figure 74.5 shows the general flow of investigation. The empirical semivariogram is computed after a suitable choice is made for the LAGDISTANCE= and MAXLAGS= options in PROC VARIOGRAM, and possibly the NDIR= option or the DIRECTIONS statement for computations in more than one directions. Potential theoretical models (which can also incorporate nesting, anisotropy, and the nugget effect) are then plotted against the empirical semivariogram and evaluated. A suitable theoretical model is found by using the methodology presented in the section "Examples: VARIOGRAM Procedure" on page 11047 in Chapter 131, "The VARIOGRAM Procedure."

Eight theoretical models are supported by PROC KRIGE2D: the Gaussian, exponential, Matérn, spherical, cubic, pentaspherical, sine hole effect and power models. See also the section "Theoretical Semivariogram Models" on page 11008 in Chapter 131, "The VARIOGRAM Procedure." These eight model forms are now examined in more detail: the Gaussian, exponential, and Matérn forms are examined as one group; the spherical, cubic, and pentaspherical as a second group; and the remaining power and sine hole effect models are examined individually. For comparison purposes, the axes in the forms' illustrations are kept the same across the plots, and the corresponding parameters of the different forms have the same values.

In PROC KRIGE2D the parameters a_0 and c_0 for all forms correspond to the RANGE= and SCALE= options, respectively, in the MODEL statement. For all model forms, the dimension of c_0 is the same as the dimension of the variance of the spatial process $Z(\boldsymbol{s})$. For all forms but the power model, the dimension of a_0 is length with same units as the distance h in the semivariance $\gamma_z(h)$. See the section "The Power Semivariogram Model" on page 5582 for more details about interpretation of the power model a_0 parameter.

Figure 74.5 Flowchart for Semivariogram Selection

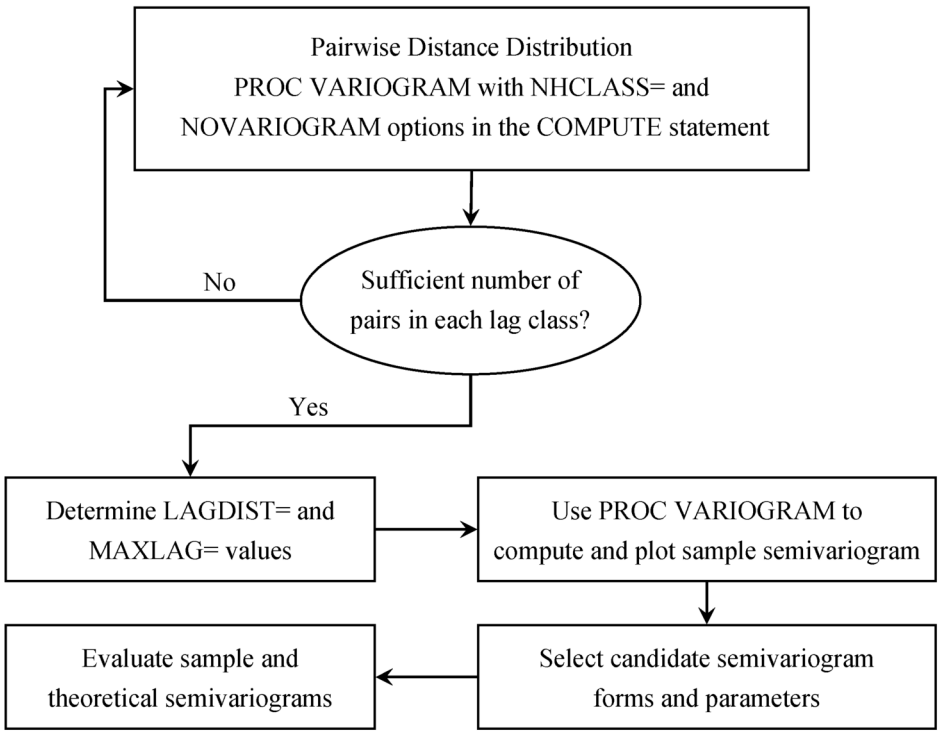

The Gaussian Semivariogram Model

The form of the Gaussian model is

$$\gamma_z(h) = c_0 \left[1 - \exp\left(-\frac{h^2}{a_0^2} \right) \right]$$

The shape is displayed in Figure 74.6, using range $a_0 = 1$ and scale $c_0 = 4$.

The vertical line at $h = r_\epsilon = \sqrt{3}a_0$ shows the *effective* (or *practical*) *range* as defined by Deutsch and Journel (1992) or the *range* ϵ defined by Christakos (1992). The effective range is the h-value where the covariance is approximately 5% of its value at zero. Alternatively, the stationarity assumption implies that the effective range is the h value where the semivariance is approximately 5% of the sill value, as shown in Figure 74.6.

In the Gaussian model the semivariance $\gamma_z(h)$ approaches the sill asymptotically at c_0.

The Exponential Semivariogram Model

The form of the exponential model is

$$\gamma_z(h) = c_0 \left[1 - \exp\left(-\frac{h}{a_0} \right) \right]$$

The shape is displayed in Figure 74.6, using range $a_0 = 1$ and scale $c_0 = 4$.

The vertical line at $h = r_\epsilon = 3a_0$ is the *effective* (or *practical*) *range* or the *range* ϵ (that is, the h-value where the covariance is approximately 5% of its value at zero).

As in the Gaussian model, the sill in this example is at 4.0 variance units (corresponding to $c_0 = 4$) and is approached asymptotically.

The major distinguishing feature of the Gaussian and exponential forms is the shape in the neighborhood of the origin $h = 0$, as Figure 74.6 illustrates. In general, small lags are important in determining an appropriate theoretical form based on an empirical semivariogram.

The Matérn Semivariogram Model

The form of the Matérn model is

$$\gamma_z(h) = c_0 \left[1 - \frac{2}{\Gamma(\nu)} \left(\frac{h\sqrt{\nu}}{a_0} \right)^{\nu} K_\nu \left(2\frac{h\sqrt{\nu}}{a_0} \right) \right]$$

where $\nu > 0$ is the smoothness factor parameter. Figure 74.6 shows an example of the Matérn form, where range $a_0 = 1$, scale $c_0 = 4$, and $\nu = 1.5$.

The Matérn semivariance $\gamma_z(h)$ is a class of semivariance models that emerge for different values of the smoothing parameter ν. The Matérn form reaches its sill value c_0 asymptotically.

The Gaussian and exponential semivariances are two frequently used members of the Matérn class of semivariances. In particular, the exponential semivariance model is derived from the Matérn class of models for $\nu = 0.5$. Also, when $\nu \to \infty$ then the Matérn semivariance gives the Gaussian model. In Figure 74.6 the selected value of $\nu = 1.5$ places the Matérn form in between the Gaussian and the exponential. The Matérn semivariance typically begins to look and behave as the Gaussian for values of $\nu > 10$.

Figure 74.6 Gaussian, Exponential, and Matérn Semivariograms with Parameters $a_0 = 1$, $c_0 = 4$, and $\nu = 1.5$

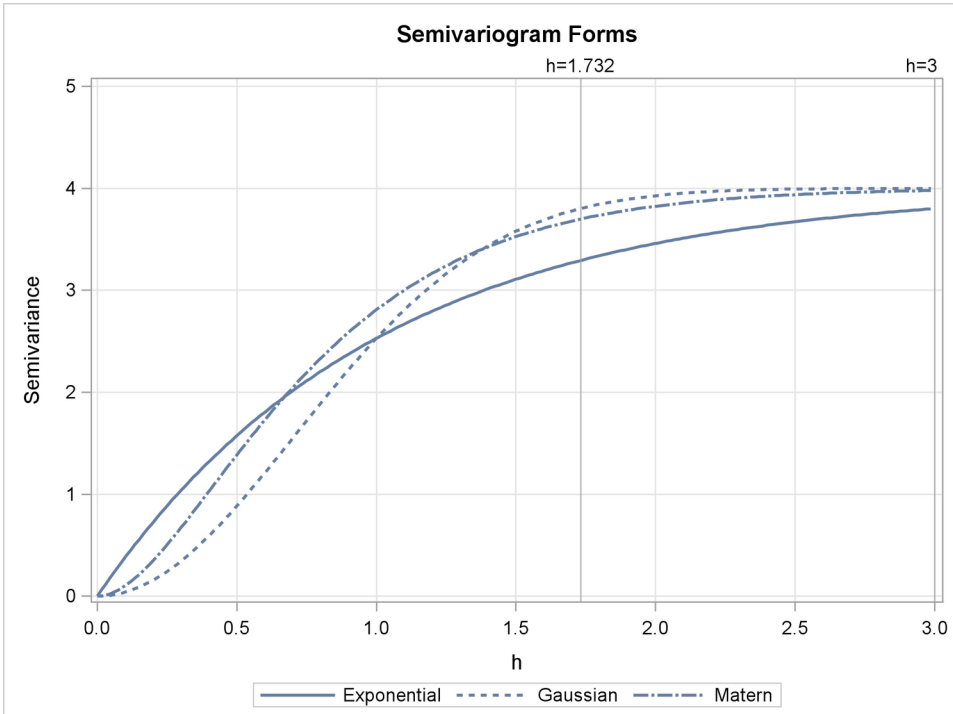

The Spherical Semivariogram Model

The form of the spherical model is

$$\gamma_z(h) = \begin{cases} c_0 \left[\frac{3}{2}\frac{h}{a_0} - \frac{1}{2}(\frac{h}{a_0})^3 \right] & \text{for } h \le a_0 \\ c_0 & \text{for } h > a_0 \end{cases}$$

The shape is displayed in Figure 74.7, using range $a_0 = 1$ and scale $c_0 = 4$.

The vertical line at $h = 1$ shows the range a_0 of the model.

In the case of the spherical model, $\gamma_z(h)$ actually reaches the sill value at c_0, unlike the Gaussian and exponential types where the sill is a horizontal asymptote.

The Cubic Semivariogram Model

The form of the cubic model is

$$\gamma_z(h) = \begin{cases} c_0 \left[7(\frac{h}{a_0})^2 - \frac{35}{4}(\frac{h}{a_0})^3 + \frac{7}{2}(\frac{h}{a_0})^5 - \frac{3}{4}(\frac{h}{a_0})^7 \right] & \text{for } h \le a_0 \\ c_0 & \text{for } h > a_0 \end{cases}$$

The cubic form shape is displayed in Figure 74.7, using range $a_0 = 1$ and scale $c_0 = 4$.

The vertical line at $h = 1$ shows the range a_0 of the model.

Similarly to the spherical model, the cubic model, $\gamma_z(h)$ reaches the sill value at c_0 and maintains this value after a distance h equal to the model range.

The Pentaspherical Semivariogram Model

The form of the pentaspherical model is

$$\gamma_z(h) = \begin{cases} c_0 \left[\frac{15}{8}\frac{h}{a_0} - \frac{5}{4}(\frac{h}{a_0})^3 + \frac{3}{8}(\frac{h}{a_0})^5 \right] & \text{for } h \le a_0 \\ c_0 & \text{for } h > a_0 \end{cases}$$

The pentaspherical form shape is displayed in Figure 74.7, using range $a_0 = 1$ and scale $c_0 = 4$.

The vertical line at $h = 1$ shows the range a_0 of the model.

The pentaspherical semivariance behaves like the spherical and cubic semivariances, in that $\gamma_z(h)$ increases with distance until it reaches the sill value c_0 at the distance h equal to the model range a_0.

Figure 74.7 accents the differences in the behavior of the featured semivariances. Specifically, the cubic and pentaspherical forms reach the sill value faster than the spherical form. Also, the spherical and pentaspherical forms exhibit a more linear behavior at distances close to the origin $h = 0$.

Figure 74.7 Spherical, Cubic, and Pentaspherical Semivariograms with Parameters $a_0 = 1$ and $c_0 = 4$

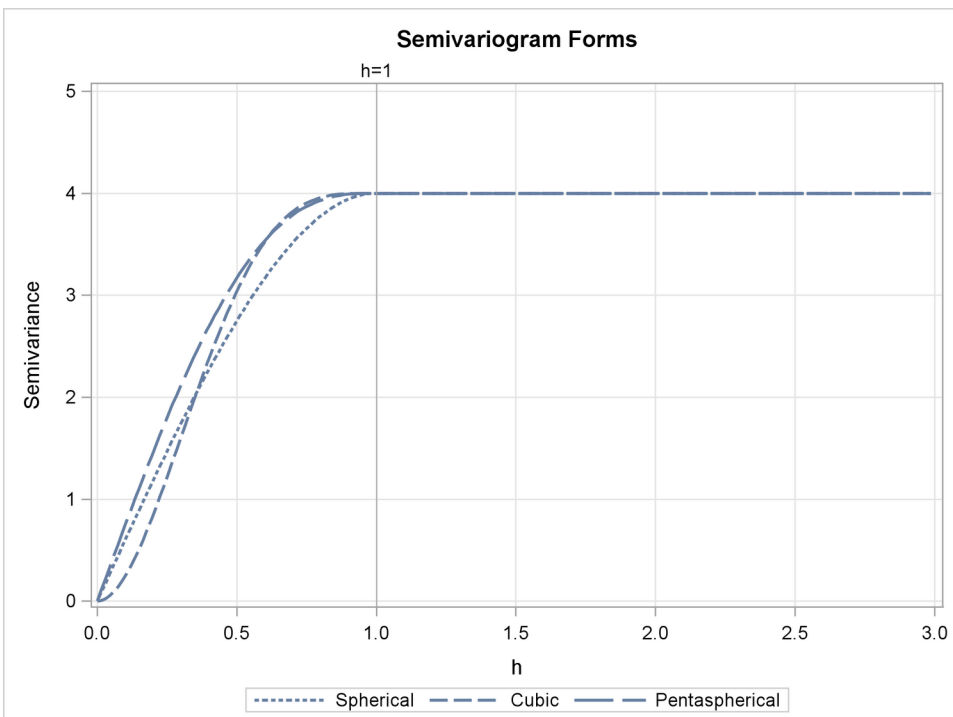

The Sine Hole Effect Semivariogram Model

The form of the sine hole effect model is

$$\gamma_z(h) = c_0 \left[1 - \frac{\sin(\pi h/a_0)}{\pi h/a_0} \right]$$

Figure 74.8 shows an example of the sine hole effect form, where range $a_0 = 1$ and scale $c_0 = 4$.

The vertical line at $h = 1$ shows the range a_0 of the model.

The sine hole effect semivariance $\gamma_z(h)$ increases with distance. It has the distinct characteristic that it reaches the sill at a distance $h = a_0$ equal to the model range and then it oscillates around the sill value with a decreasing amplitude as it moves to higher values of h.

The Power Semivariogram Model

The form of the power model is

$$\gamma_z(h) = c_0 h^{a_0}$$

For this model, the parameter a_0 is known as the power exponent. This is a dimensionless quantity which must range within $0 \leq a_0 < 2$ so that the power model is a permissible semivariance model.

The KRIGE2D procedure enables you to specify power exponent values that are outside this range when you also explicitly specify the POWNOBOUND option in the MODEL statement. However, parameter values equal to or greater than 2 can result in singular covariance matrices or negative prediction errors.

For the special case of $a_0 = 1$ the form yields a straight line. In this case the power model reduces to the linear model. The parameter c_0 designates the slope of the power form and has dimensions of the variance as in the other models.

The power model has no sill; this differentiates it from the rest of the models presented earlier. Spatial correlation that is described by a power model indicates that the stochastic process variance increases constantly with distance. The shape of the power model with $a_0 = 0.4$ and $c_0 = 4$ is displayed in Figure 74.8.

Figure 74.8 Sine Hole Effect Semivariogram with Range $a_0 = 1$ and Scale $c_0 = 4$, and Power Semivariogram with Exponent $a_0 = 0.4$ and Slope $c_0 = 4$

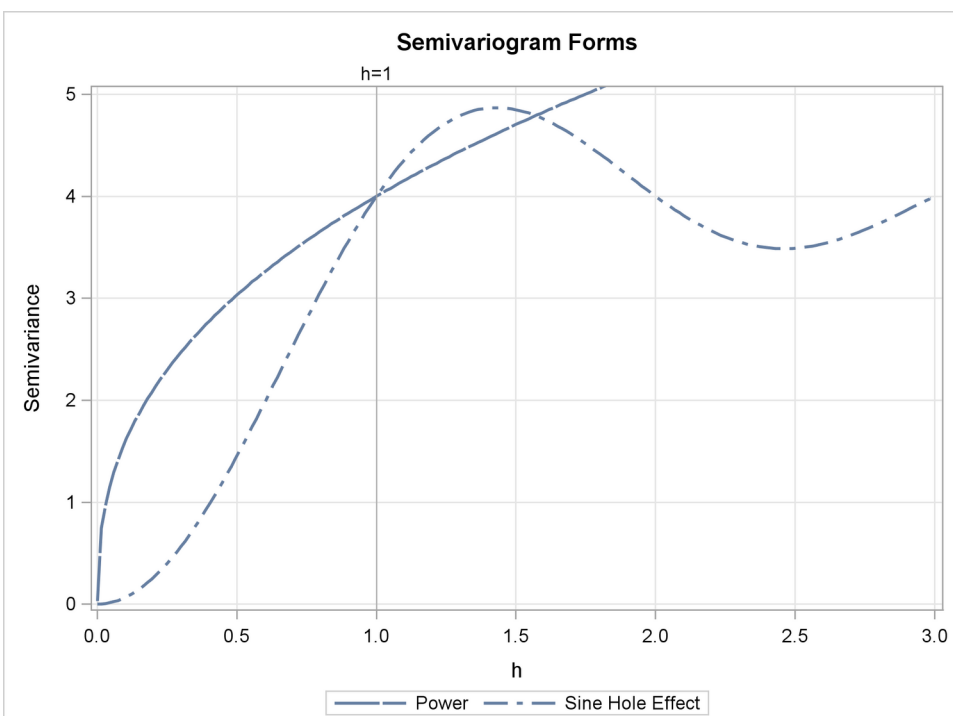

For comparison purposes, Figure 74.9 displays all eight semivariance forms that you can use with PROC KRIGE2D. The figure displays a composition of the different forms with the parameter values selected earlier throughout this section. Depending on the empirical semivariogram, these models provide you with flexibility to select an appropriate theoretical semivariance model for prediction.

Figure 74.9 Semivariogram Forms Used in PROC KRIGE2D

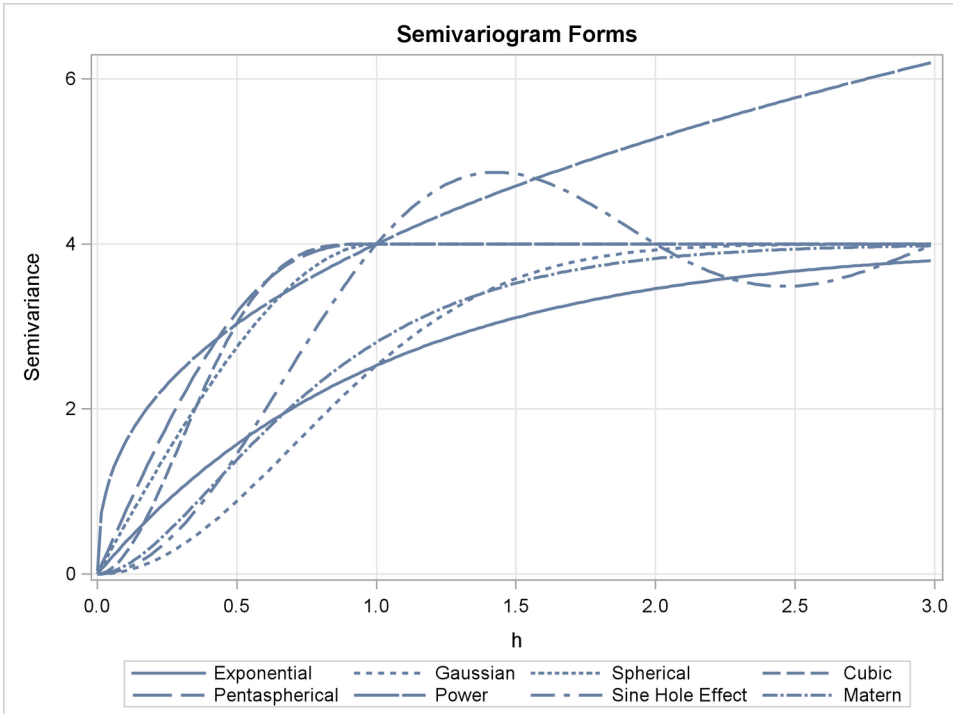

Nested Models

For a given set of spatial data, a plot of an experimental semivariogram might not seem to fit any of the individual theoretical models. In such a case, you might obtain a more accurate fit if you consider your covariance model to be the sum of two or more covariance structures. Such covariance models are called *nested* models. Nesting is common in geologic applications where correlations can exist at different length scales. At small lag distances h, the smaller scale correlations dominate, while the large scale correlations dominate at larger lag distances.

Nested models are permissible covariances if they are the sum of permissible models. Therefore, you can include in a sum any combination of the models presented in the preceding subsections and produce permissible covariance models. As an illustration, consider two semivariogram models: an exponential and a spherical,

$$\gamma_{z,1}(h) = c_{0,1} \exp(-\frac{h}{a_{0,1}})$$

and

$$\gamma_{z,2}(h) = \left\{ \begin{array}{ll} c_{0,2}\left[\frac{3}{2}\frac{h}{a_{0,2}} - \frac{1}{2}(\frac{h}{a_{0,2}})^3\right], & \text{for } h \le a_{0,2} \\ c_{0,2}, & \text{for } h > a_{0,2} \end{array} \right\}$$

with $c_{0,1} = 1, a_{0,1} = 2.5, c_{0,2} = 2$, and $a_{0,2} = 1$. If both of these correlation structures are present in a spatial process $\{Z(\boldsymbol{s}), \boldsymbol{s} \in D\}$, then the semivariance $\gamma_z(h)$ of this process can be expressed as

$$\gamma_z(h) = \gamma_{z,1}(h) + \gamma_{z,2}(h)$$

In this case, the experimental semivariogram $\gamma_z(h)$ for the process $Z(\boldsymbol{s})$ resembles the semivariogram of the sum of $\gamma_{z,1}(h)$ and $\gamma_{z,2}(h)$. This is illustrated in Figure 74.10.

The sum of $\gamma_{z,1}(h)$ and $\gamma_{z,2}(h)$ in Figure 74.10 does not resemble any *single* theoretical semivariogram; however, its shape at $h = 1$ is similar to a spherical form. The asymptotic approach to a sill at three variance units, along with the shape around $h = 0$, indicates an exponential structure. The sill value c_0 of the sum is the sum of the individual sills $c_{0,1} = 1$ and $c_{0,2} = 2$. In general, a nested model has a sill equal to the sum of the sills of its nested structures plus the nugget effect, if present.

See Hohn (1988, p. 38ff) for further examples of nested correlation structures.

Figure 74.10 Sum of Exponential and Spherical Structures at Different Scales

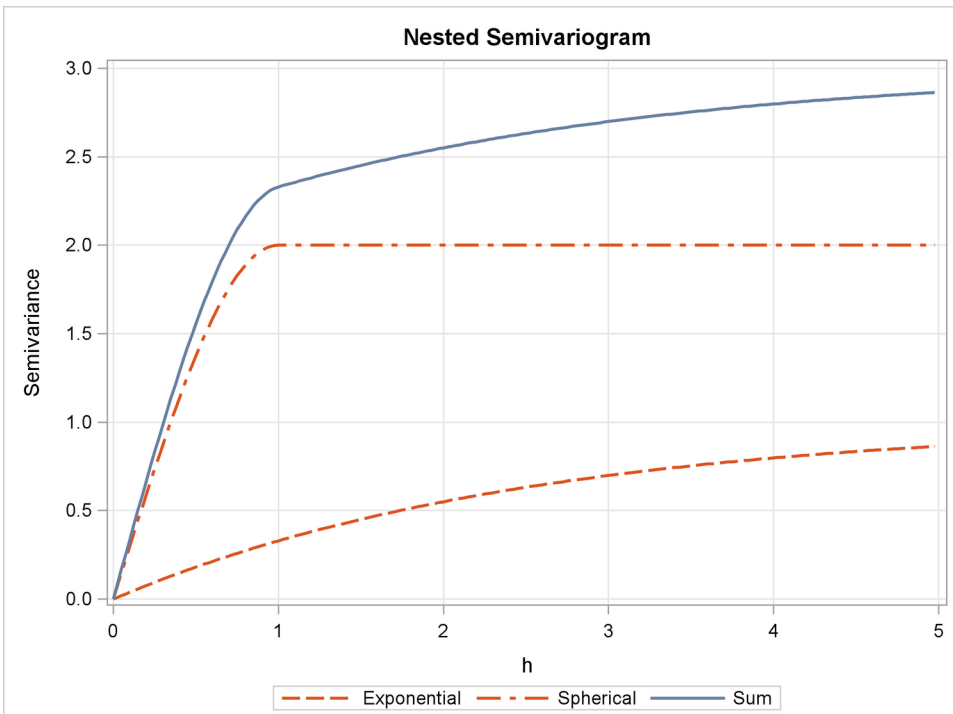

The Nugget Effect

For all the semivariogram models considered previously, the following property holds:

$$\gamma_z(0) = \lim_{h \downarrow 0} \gamma_z(h) = 0$$

However, a plot of the experimental semivariogram might indicate a discontinuity at $h = 0$; that is, $\gamma_z(h) \rightarrow c_n > 0$ as $h \rightarrow 0$, while $\gamma_z(0) = 0$. The quantity c_n is called the *nugget effect*; this term is from mining geostatistics where nuggets literally exist, and it represents variations at a much smaller scale than any of the measured pairwise distances—that is, at distances $h \ll h_{min}$, where

$$h_{min} = \min_{i,j} h_{ij} = \min_{i,j} | \boldsymbol{s}_i - \boldsymbol{s}_j |$$

Nonzero nugget effects have been associated with conceptual and theoretical difficulties; see Cressie (1993, section 2.3.1) and Christakos (1992, section 7.4.3) for details. There is no *practical* difficulty, however; you simply visually extrapolate the experimental semivariogram as $h \rightarrow 0$. The importance of availability of data at small lag distances is again illustrated.

As an example, an exponential semivariogram with a nugget effect c_n has the form

$$\gamma_z(h) = c_n + {\sigma_0}^2 \left[1 - \exp\left(-\frac{h}{a_0} \right) \right], h > 0$$

and

$$\gamma_z(0) = 0$$

where the factor ${\sigma_0}^2$ is called the *partial sill* and the sill $c_0 = c_n + {\sigma_0}^2$.

This is illustrated in Figure 74.11 for the parameters $a_0 = 1$, ${\sigma_0}^2 = 4$, and nugget effect $c_n = 1.5$.

You can specify the nugget effect in PROC KRIGE2D with the NUGGET= option in the MODEL statement. It is a separate, additive term independent of direction; that is, it is isotropic. The way to approximate an anisotropic nugget effect is described in the following section.

Figure 74.11 Exponential Semivariogram Model with a Nugget Effect $c_n = 1.5$

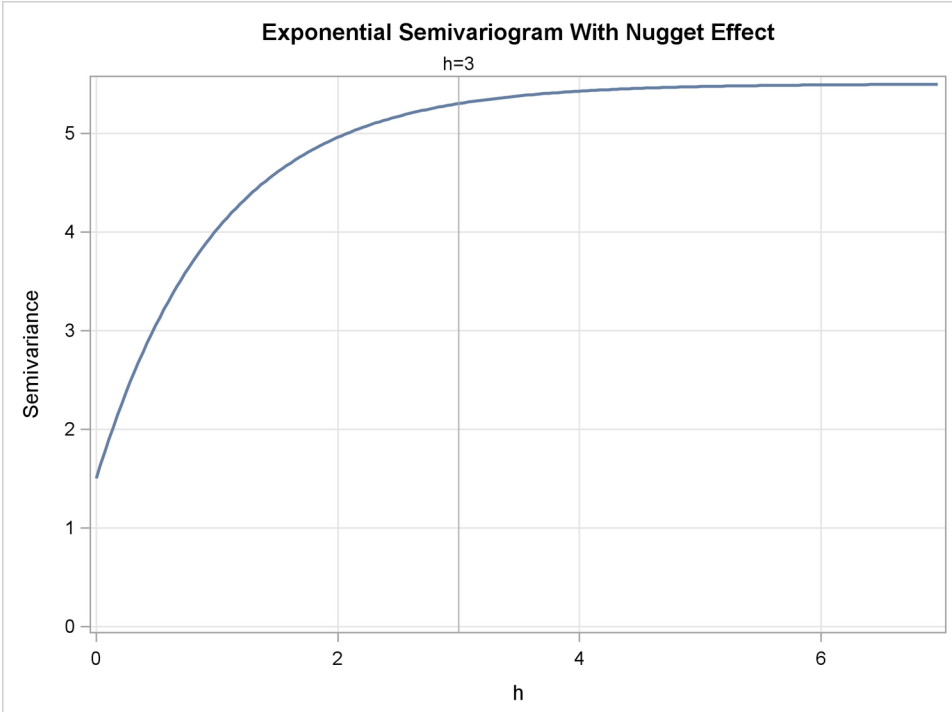

Anisotropic Models

In all of the theoretical models considered previously, the lag distance h is entered as a scalar value. This implies that the correlation between the spatial process at two point pairs P_1, P_2 is dependent *only* on the separation distance $h = \mid P_1 P_2 \mid$, not on the orientation of the vector $\boldsymbol{h}$. A spatial process described by an SRF $\{Z(\boldsymbol{s}), \boldsymbol{s} \in D \subset \mathcal{R}^2\}$ with this property is called isotropic, as is the associated covariance or semivariogram.

However, real spatial phenomena often show directional effects. Particularly in geologic applications, measurements along a particular direction might be highly correlated, while typically the perpendicular direction shows little or no correlation. Such processes are called anisotropic; see, for example, Journel and Huijbregts (1978, section III.B.4).

When the correlation structure varies across different directions, you need different models for each direction so that you can account correctly for the continuity within the SRF. The following subsections describe how techniques are applied to override the anisotropy effects for computational purposes. First, characteristics of anisotropy are examined.

The semivariogram sill is a measure of the process variability; hence the direction of the highest continuity is perpendicular to the direction where the highest sill occurs. If the sill is the same in all directions, then the direction with the highest range indicates highest continuity. The directions in which the spatial process $\{Z(\boldsymbol{s}), \boldsymbol{s} \in D\}$ is most and least correlated are called the *major* and *minor* axis of anisotropy, respectively.

In some cases, these directions are known a priori. This can occur in mining applications where the geology of a region is known in advance. In most cases however, nothing is known about possible anisotropy. Depending on the amount of data available, using several directions is usually sufficient to determine the presence of

anisotropy and to find the approximate major and minor axis directions; see the discussion in the section "Anisotropy" on page 11016 in Chapter 131, "The VARIOGRAM Procedure," documentation. You can find a detailed example of anisotropy investigation in the section "Example 131.2: An Anisotropic Case Study with Surface Trend in the Data" on page 11058 in Chapter 131, "The VARIOGRAM Procedure," documentation.

After you explore an anisotropic process and you identify the minor and major axis directions, you can compute the *anisotropy factor* parameter R which is defined as

$$R = \frac{{a_0}^{min}}{{a_0}^{max}}$$

where ${a_0}^{min}$ is the semivariogram range in the direction of the minor axis and ${a_0}^{max}$ is the semivariogram range in the direction of the major axis.

There are two types of anisotropy, depending on which semivariogram characteristics change in different directions. These types are the *geometric* and the *zonal* anisotropy, and either or both can be present. Both are examined in detail in the following subsections.

Geometric Anisotropy

Geometric anisotropy is the simplest type of anisotropy. It occurs when the same sill (or scale) parameter c_0 is present in all directions but the range a_0 changes with direction. In geometric anisotropy the covariance model uses the same forms in all directions.

Therefore, geometric anisotropy features one single sill value, and depending on the direction the semivariogram reaches the sill within a different distance. This is illustrated in Figure 74.12, where an anisotropic exponential semivariogram is plotted. Assume that the two curves displayed in this figure have the same sill $c_0 = 1.5$ and are generated using the ranges $a_{0,1} = 3$ in the direction $\theta_1 = 30°$ (effective range is $r_{\epsilon,1} = 9$) and $a_{0,2} = 1$ in the direction $\theta_2 = 120°$ (effective range is $r_{\epsilon,2} = 3$).

As you can see from the figure, the ratio of the shorter to longer range is $R = 1/3$. The anisotropy factor R is the value to use in the RATIO= parameter in the MODEL statement in PROC KRIGE2D. When you model geometric anisotropy $R \leq 1$. In fact, isotropy is a partial case of geometric anisotropy for which ${a_0}^{min} = {a_0}^{max}$ and $R = 1$.

Figure 74.12 Geometric Anisotropy with Major Axis in the Direction $\theta_1 = 30°$

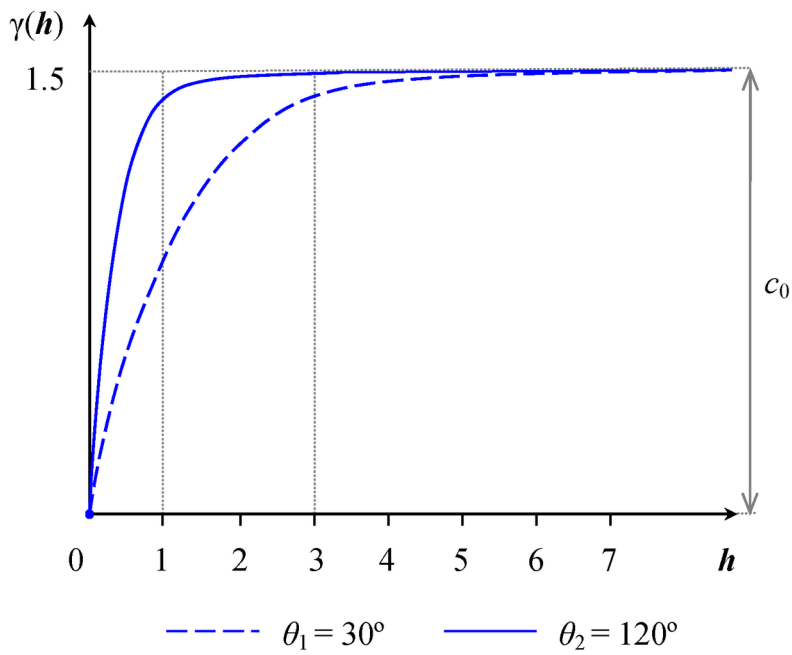

The values of the RANGE= and ANGLE= parameters in the MODEL statement in PROC KRIGE2D are set based on the major anisotropy axis characteristics. Specifically, the RANGE= parameter is the value of the major axis range ${a_0}^{max} = a_{0,1}$, and the ANGLE= parameter is the angle θ_1 of the major axis measured clockwise from north (angles measured in this way are also known as *azimuths*). You can then specify the following MODEL statement in PROC KRIGE2D to approximate the covariance structure:

```
model form=exp range=3 scale=1.5 angle=30 ratio=0.3333;
```

If you use a nested model, provide the type for each one of the nested structures with the FORM= option, and assign the individual SCALE= parameters so that they add up to the total sill (include in the sum the nugget effect, if present). In the typical case, all of your nested structures have the same anisotropy axes. This means that you specify the same ANGLE= parameter value for all structures. Each structure likely has its own values for the RANGE= and RATIO= parameters depending on the degree of its contribution to the nested model.

The terminology associated with geometric anisotropy is that of ellipses. To see how this comes about, consider the following hypothetical set of calculations. Let $\{Z(\boldsymbol{s}), \boldsymbol{s} \in D \subset \mathcal{R}^2\}$ be a geometrically anisotropic process, and assume sufficient data points are present to calculate an experimental semivariogram at a large number of angle classes $\theta \in \{0, \delta\theta, 2\delta\theta, \ldots, 180°\}$. At each of these angles θ, the experimental semivariogram is plotted and the range a_0 is recorded. A diagram in polar coordinates (a_0, θ) yields an ellipse with the major axis ${a_0}^{max}$ in the direction of the largest a_0 and the minor axis ${a_0}^{min}$ perpendicular to it. For the example in Figure 74.12, the ellipse is shown in Figure 74.13(a). Its major axis has size ${a_0}^{max}$ situated at angle θ_1 clockwise from north, and the minor axis has size ${a_0}^{min}$ oriented at angle θ_2 clockwise from north.

The KRIGE2D procedure handles geometric anisotropy by applying a reversible transformation in two steps that converts geometric anisotropy into isotropic conditions.

The first step is to align your coordinates axes with the anisotropy ellipse axes. Specifically, you choose to rotate by an angle φ the standard Cartesian orientation of the (x, y) coordinates system shown in Figure 74.13(a) so that the Y axis coincides with the ellipse minor axis. The rotation result is illustrated in Figure 74.13(b). The second step is to elongate the minor axis so its length equals that of the major axis of the ellipse. You can see the result in Figure 74.13(c). The computational details are shown in the following.

Figure 74.13 Transformation Applied to Geometric Anisotropy

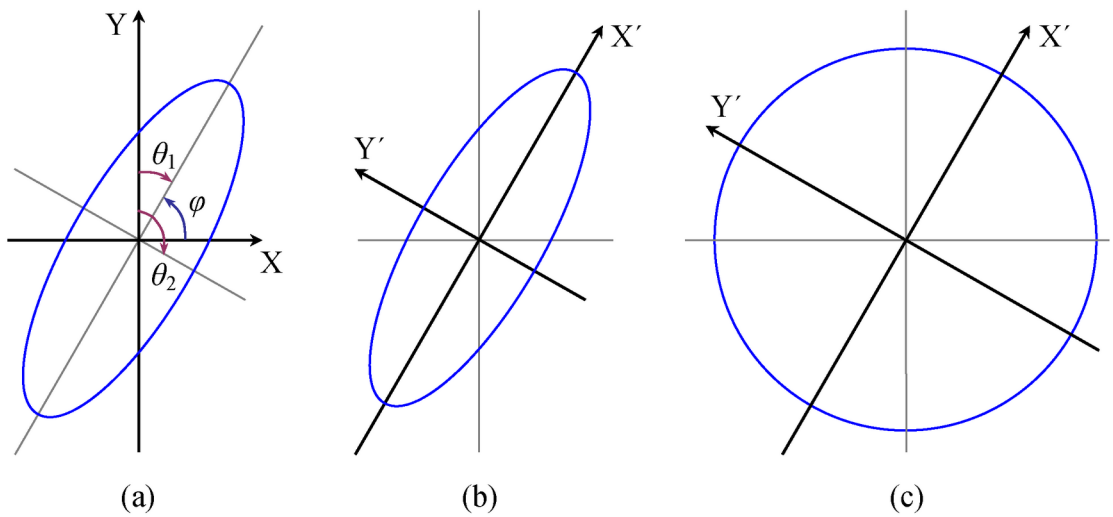

The transformation angle φ is measured in standard Cartesian orientation counterclockwise from the X axis (east). If the major axis azimuth is θ_1, then the Cartesian system of (x, y) needs to be rotated by $\varphi = 90° - \theta_1$ so that the Y axis can coincide with the ellipse minor axis; see Figure 74.13(a).

Let us call the ellipse major axis X' and the minor axis Y'. The transformation that converts any coordinates in the (x, y) system into (x', y') coordinates in terms of φ is given by the matrix:

$$\mathbf{H} = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) \\ -\sin(\varphi) & \cos(\varphi) \end{pmatrix}$$

The elongation of the minor axis in the second step is performed with the matrix:

$$\mathbf{D}_R = \begin{pmatrix} 1 & 0 \\ 0 & 1/R \end{pmatrix}$$

NOTE: These two steps are sequential and their order cannot be reversed. For any point pair P_1 and P_2 with respective coordinates $\boldsymbol{s}_1 = (x_1, y_1)$ and $\boldsymbol{s}_2 = (x_2, y_2)$ in the (x, y) axes, their distance is given by

$$| \, P_i P_j \, |_{(x,y)} = h = \sqrt{(\delta x)^2 + (\delta y)^2}$$

where the distance components $\delta x = x_2 - x_1$ and $\delta y = y_2 - y_1$. Based on the previous, the corresponding distances $\delta x'$ and $\delta y'$ in the (x', y') coordinates system are given by the vector:

$$\begin{pmatrix} \delta x' \\ \delta y' \end{pmatrix} = \mathbf{D}_R \, \mathbf{H} \begin{pmatrix} \delta x \\ \delta y \end{pmatrix} = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) \\ -\sin(\varphi)/R & \cos(\varphi)/R \end{pmatrix} \begin{pmatrix} \delta x \\ \delta y \end{pmatrix}$$

The transformed interpair distance is then:

$$| P_i P_j |_{(x',y')} = h' = \sqrt{(\delta x')^2 + (\delta y')^2}$$

As a result, the original anisotropic semivariogram in Figure 74.12 that was a function $\gamma(\boldsymbol{h}) = \gamma(h, \theta)$ of both h and θ is then transformed to an equivalent function $\hat{\gamma}(h')$ only of h':

$$\hat{\gamma}(h') = \gamma(\boldsymbol{h})$$

This single isotropic semivariogram $\hat{\gamma}(h')$ is then used for kriging purposes.

The two steps used by PROC KRIGE2D in the previous analysis can also be performed in a different manner. For instance, you might equivalently choose to rotate the (x, y) Cartesian coordinates so that the Y axis coincides with the ellipse major axis, rather than with the minor axis as was shown earlier. Also, you might prefer to compress the major axis rather than elongating the short one. In any case, you need to perform the appropriate computations for the transformation of your choice.

Zonal Anisotropy

In zonal anisotropy, the sill (or scale) parameter c_0 is different for different directions. It is not possible to transform such a structure into an isotropic semivariogram. Instead, nesting and geometric anisotropy are used together to approximate zonal anisotropy.

When the scale varies with direction, the lowest scale (that is, the lowest variance) naturally corresponds to the maximum continuity direction. The same direction has the longest range, as also discussed in the section "Geometric Anisotropy" on page 5587.

A varying scale with direction can be interpreted as having one or more model components whose individual contributions to the total variance differ with direction. For each such component, its contribution (scale) ranges between zero and a maximum value. This makes it unlikely that you can describe a natural process with a pure zonal model, because doing so would imply zero continuity in the direction of zero contribution; see also Chilès and Delfiner (1999, p. 96).

In a simple case of zonal anisotropy, a model includes one zonal component. The zonal component makes its highest contribution in a direction perpendicular to the maximum continuity direction, and it contributes zero to the maximum continuity direction. This is necessary; otherwise, there would be a direction with a total scale less than the scale in the maximum continuity direction. Following a similar reasoning, the zonal component's direction of maximum contribution cannot coincide with the one of maximum continuity. In the general case, there can be multiple zonal components, each making its highest contribution in a different direction.

The following describes how to deal with zonal anisotropy in your analysis; see also Goovaerts (1997, p. 96) and Deutsch and Journel (1992, pp. 27–32). If you start with an empirical semivariogram, you can investigate zonal anisotropy by identifying whether a maximum and a minimum scale exist in two specific directions. If they exist, typically these two directions might be perpendicular. Then proceed to identify the zonal component that causes the difference in scale by fitting the empirical semivariogram. You represent zonal component as an additional nested structure in the direction of maximum total scale.

If the minimum and maximum sills are not in perpendicular directions, then you might be seeing the combined effects of multiple zonal components in different directions. In that case you might be able to approximate the continuity behavior by assuming a single zonal component in the direction that is perpendicular to the one

of maximum continuity. Alternatively, you might decide to investigate a more elaborate configuration for the model components. In this case, you need to maintain a geometrical anisotropy part across all directions and add zonal components in an appropriate way to match your empirical semivariance in different directions.

After you have a theoretical semivariance model with zonal anisotropy, the next step is to include zonal components in your prediction or simulation analysis. In PROC KRIGE2D you can specify zonal components either explicitly or with the use of results previously saved in item stores produced by the VARIOGRAM procedure.

Specifying a zonal component explicitly in the MODEL statement has the following implications:

- The RANGE= parameter for the zonal component refers to the range value in the direction of maximum zonal contribution, unlike the case of ranges specified for nonzonal components that refer to the direction of maximum continuity.
- The anisotropy factor *R* in the RATIO= parameter for the zonal component should be specified as a large positive value to designate zero contribution in the perpendicular direction.

To explain the previous point, remember that *R* is defined as $R = {a_0}^{min}/{a_0}^{max}$. Its value specifies how much to elongate the minor anisotropy axis to make it equal to the major anisotropy axis, in order to transform geometric anisotropy into isotropy. Intuitively, an infinite *R* value makes it impossible for the minor axis to become as large as the major axis. This is equivalent to having a very large major anisotropy axis; hence, it indicates a very large range across the major axis direction. Indeed, you can consider a zero zonal contribution in the major anisotropy axis as a very large range of the zonal component along this direction. The particular range is so large that the zonal component practically never reaches its scale along this direction, and this is interpreted as zero contribution.

In the case where you specify zonal anisotropy by using the contents of an item store, you only need to specify the geometric anisotropy components in the SSEL(MODEL=) option, and the zonal components as suboptions of the SSEL(TYPE=ANIZON) option. Then, the KRIGE2D or SIM2D procedure checks whether the item store contains models that are suitable to use, based on your specifications.

The following two examples illustrate different instances of zonal anisotropy and how to specify the corresponding covariance model parameters in PROC KRIGE2D.

Example 1

The first example shows that if you can model the direction with the highest sill as a nested model, then you can treat the case as a composition of geometric anisotropy and an additional structure that acts only in the direction of the increased sill.

Consider a spatial process in which the fitting of theoretical models in your experimental semivariogram produces a correlation structure like the one shown in Figure 74.14. In the direction $\theta_1 = 40°$, the covariance model has a single exponential structure $\gamma_1(\boldsymbol{h}) = \mathrm{Exp}(a_{0,1E}, c_{0,1E})$ with range $a_{0,1E} = 2$ and sill $c_{0,1E} = 1.5$. In the direction $\theta_2 = 130°$, the covariance model $\gamma_2(\boldsymbol{h}) = \mathrm{Exp}(a_{0,2E}, c_{0,2E}) + \mathrm{Sph}(a_{0,2S}, c_{0,2S})$ has two nested structures: an exponential structure with range $a_{0,2E} = 0.5$ and sill $c_{0,2E} = 1.5$ and a spherical structure with range $a_{0,2S} = 1$ and sill $c_{0,2S} = 0.5$.

Figure 74.14 Zonal Anisotropy in Two Directions

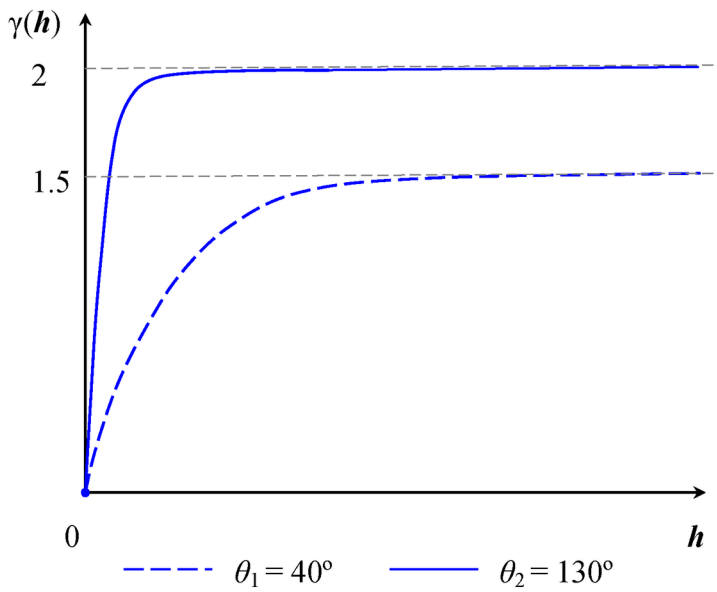

The total sill in the direction θ_2 of highest variance is the sum of the nested structures' sills $c_{0,2E} + c_{0,2S} = 2$. You can consider that your process is characterized by a geometrically anisotropic exponential structure with common sill $c_{0,E} = 1.5$ across all directions and major axis range $a_{0,1E} = 2$, and by a spherical structure which is a zonal anisotropy component that contributes only in the θ_2 direction. Based on the remarks in this section, the RATIO= parameter for the exponential structure is $R_E = 0.5/2 = 0.25$, whereas for the spherical structure you choose a large value, such as $R_S = 10^8$.

Then, you can approximate this structure in PROC KRIGE2D by specifying the two structures with the following MODEL statement:

```
model form=(exp,sph) range=(2,1) scale=(1.5,0.5)
      angle=(40,130) ratio=(0.25,1e8);
```

You can handle more elaborate cases in a similar way, where the covariance models in different directions might all be nested models. Your goal is to model the continuity by starting with a sum of isotropic or geometrically anisotropic structures whose total sill is the lowest sill in all directions. Then, in each of the directions with higher sills you add a zonal anisotropy component to the corresponding sum to compensate for the increased variability in that direction.

Example 2

The second example provides important perspective about the physics of zonal anisotropy analysis. It is an extreme case of the general guidelines for zonal anisotropy. You examine what happens when each of the directions is modeled with a single-form, non-nested model, and the sills for these models are clearly different.

Consider a spatial process with a continuity description almost identical to the one in the previous example. In the direction $\theta_1 = 40°$, the covariance model has again a single exponential structure $\gamma_1(\boldsymbol{h}) = \mathrm{Exp}(a_{0,1E}, c_{0,1E})$ with range $a_{0,1E} = 2$ and sill $c_{0,1E} = 1.5$. However, this time in the direction $\theta_2 = 130°$ you have fit the experimental semivariogram by using a single exponential structure

$\gamma_2(\boldsymbol{h}) = \mathrm{Exp}(a_{0,2E}, c_{0,2E})$ with range $a_{0,2E} = 1$ and sill $c_{0,2E} = 2$. These models are shown in Figure 74.15.

Figure 74.15 Zonal Anisotropy in Two Directions

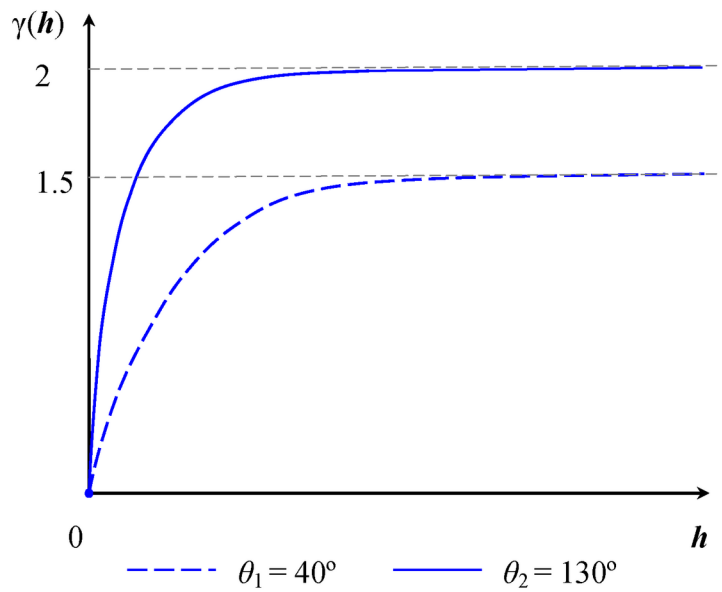

In this case you have a simplified situation with a single covariance structure in each direction, and the two structures have different scale parameter values. This is a case of zonal anisotropy in which all directions have no shared component. Hence, you have a case with two pure zonal components, where both structures can be practically approximated by specifying two models with large RATIO= values. You could then use the following MODEL statement in PROC KRIGE2D to describe the covariance in this example:

```
model form=(exp,exp) range=(2,1) scale=(1.5,2)
      angle=(40,130) ratio=(1e8,1e8);
```

The semivariogram of the specified model is accurately shown in Figure 74.15, because the angles θ_1 and θ_2 are perpendicular and each component has a contribution to all directions except for the one that is perpendicular to its angle. In the general case, θ_1 and θ_2 might not be perpendicular; hence the maximum and minimum scale values can be different from those displayed in Figure 74.15.

In general, avoid configurations with pure zonal components. Correlation models with pure zonal components might imply zero continuity along some direction, which is a very unlikely occurrence in natural processes. For that reason, in similar cases try to use the analysis illustrated in the previous example. In particular, try to model the highest sill direction as a nested structure (such that it contains a geometrical anisotropy component whose cumulative sill is equal to the lower sill) and a zonal anisotropy component that accounts for the sill difference.

Anisotropic Nugget Effect

Isotropic nugget effects can be approximated with nested models, where one of the nested structures has a very small range. Applying a geometric anisotropy specification to this nested structure results in an anisotropic nugget effect.

Details of Ordinary Kriging

Introduction

Three common characteristics are often observed with spatial data (that is, data indexed by their spatial locations):

(i) slowly varying, large-scale variations in the measured values

(ii) irregular, small-scale variations

(iii) similarity of measurements at locations close together

As an illustration, consider a hypothetical example in which an organic solvent leaks from an industrial site and spreads over a large area. Assume the solvent is absorbed and immobilized into the subsoil above any groundwater level, so you can ignore any time dependence.

To find the areal extent and the concentration values of the solvent, you need measurements. Although the problem is inherently three-dimensional, if you measure total concentration in a column of soil or take a depth-averaged concentration, it can be handled reasonably well with two-dimensional techniques.

You usually assume that measured concentrations are higher closer to the source and decrease at larger distances from the source. On top of this smooth variation, measured concentrations typically have small-scale variations, due perhaps to the inherent variability of soil properties.

You also tend to suspect that measurements made close together yield similar concentration values, while measurements made far apart can have very different values.

These physically reasonable qualitative statements have no explicit probabilistic content. A number of numerical smoothing techniques, such as inverse distance weighting and splines, make use of large-scale variations and “close distance-close value” characteristics of spatial data to interpolate the measured concentrations for contouring purposes.

While characteristics (i) and (iii) are handled by such smoothing methods, characteristic (ii), the small-scale residual variation in the concentration field, is not accounted for.

There can be situations, due to the use of the prediction map or the relative magnitude of the irregular fluctuations, where you cannot ignore these small-scale irregular fluctuations. In other words, the smoothed or predicted values of the concentration field alone are not a sufficient characterization; you also need the possible spread around these contoured values.

Spatial Random Fields

One method of incorporating characteristic (ii) into the construction of a contour map is to model the concentration field as a spatial random field (SRF). The mathematical details of SRF models are given in a number of texts, such as Cressie (1993) and Christakos (1992). The mathematics of SRFs is formidable. However, under certain simplifying assumptions, it produces classical linear predictors with very simple properties, enabling easy implementation for prediction purposes. These predictors, primarily ordinary kriging (OK), give both a prediction and a standard error of prediction at unsampled locations. This allows the construction of a map of both predicted values and level of uncertainty about the predicted values.

The key assumption in applying the SRF formalism is that the measurements come from a single realization of the SRF. However, in most geostatistical applications, the focus is on a single, unique realization. This is unlike most other situations in stochastic modeling in which there will be future experiments or observational activities (at least conceptually) under similar circumstances. This renders many traditional ideas of statistical inference ambiguous and somewhat counterintuitive.

Additional logical and methodological problems could stand in the way of applying a stochastic model to a unique but partly unknown natural process; see the introduction in Matheron (1971) and Cressie (1993, section 2.3). These difficulties have resulted in attempts to frame the prediction problem in a completely deterministic way (Isaaks and Srivastava 1988; Journel 1985). Also, some issues with kriging, and with spatial prediction methods in general, are related to the necessary assumption of ergodicity of the spatial process. This assumption is required to estimate the covariance or semivariogram from sample data. Details are provided in Cressie (1993, pp. 52–58).

Despite these difficulties, ordinary kriging remains a popular and widely used tool in modeling spatial data, especially in generating surface plots and contour maps. An abbreviated derivation of the OK predictor for point prediction and the associated standard error is discussed in the following section. Full details are given in Journel and Huijbregts (1978); Christakos (1992); Cressie (1993).

Ordinary Kriging

Denote the SRF by $Z(\boldsymbol{s}), \boldsymbol{s} \in D \subset \mathcal{R}^2$. Following the notation in Cressie (1993), the following model for $Z(\boldsymbol{s})$ is assumed:

$$Z(\boldsymbol{s}) = \mu + \varepsilon(\boldsymbol{s})$$

Here, μ is the fixed, unknown mean of the process, and $\varepsilon(\boldsymbol{s})$ is a zero mean SRF, which represents the variation around the mean.

In most practical applications, an additional assumption is required in order to estimate the covariance C_z of the $Z(\boldsymbol{s})$ process. This assumption is second-order stationarity:

$$C_z(\boldsymbol{s}_1, \boldsymbol{s}_2) = \mathrm{E}[\varepsilon(\boldsymbol{s}_1)\varepsilon(\boldsymbol{s}_2)] = C_z(\boldsymbol{s}_1 - \boldsymbol{s}_2) = C_z(\boldsymbol{h})$$

This requirement can be relaxed slightly when you are using the semivariogram instead of the covariance. In this case, second-order stationarity is required of the differences $\varepsilon(\boldsymbol{s}_1) - \varepsilon(\boldsymbol{s}_2)$ rather than $\varepsilon(\boldsymbol{s})$:

$$\gamma_z(\boldsymbol{s}_1, \boldsymbol{s}_2) = \frac{1}{2}\mathrm{E}[(\varepsilon(\boldsymbol{s}_1) - \varepsilon(\boldsymbol{s}_2))^2] = \gamma_z(\boldsymbol{s}_1 - \boldsymbol{s}_2) = \gamma_z(\boldsymbol{h})$$

By performing local kriging, the spatial processes represented by the previous equation for $Z(\boldsymbol{s})$ are more general than they appear. In local kriging, at an unsampled location $\boldsymbol{s}_0$, a separate model is fit using only data

in a neighborhood of $\boldsymbol{s}_0$. This has the effect of fitting a separate mean μ at each point, and it is similar to the *kriging with trend* (KT) method discussed in Journel and Rossi (1989).

Given the N measurements $Z(\boldsymbol{s}_1), \ldots, Z(\boldsymbol{s}_N)$ at known locations $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N$, you want to obtain a prediction of Z at an unsampled location $\boldsymbol{s}_0$. When the following three requirements are imposed on the predictor $\hat{Z}$, the OK predictor is obtained:

(i) $\hat{Z}$ is linear in $Z(\boldsymbol{s}_1), \ldots, Z(\boldsymbol{s}_N)$

(ii) $\hat{Z}$ is unbiased

(ii) $\hat{Z}$ minimizes the mean square prediction error $\mathrm{E}[(Z(\boldsymbol{s}_0) - \hat{Z}(\boldsymbol{s}_0))^2]$

Linearity requires the following form for $\hat{Z}(\boldsymbol{s}_0)$:

$$\hat{Z}(\boldsymbol{s}_0) = \sum_{i=1}^{N} \lambda_i Z(\boldsymbol{s}_i)$$

Applying the unbiasedness condition to the preceding equation yields

$$\mathrm{E}[\hat{Z}(\boldsymbol{s}_0)] = \mu \Rightarrow \sum_{i=1}^{N} \lambda_i \mathrm{E}[Z(\boldsymbol{s}_i)] = \mu \Rightarrow \sum_{i=1}^{N} \lambda_i \mu = \mu \Rightarrow \sum_{i=1}^{N} \lambda_i = 1$$

Finally, the third condition requires a constrained linear optimization that involves $\lambda_1, \ldots, \lambda_N$ and a Lagrange parameter $2m$. This constrained linear optimization can be expressed in terms of the function $L(\lambda_1, \ldots, \lambda_N, m)$ given by

$$L = \mathrm{E}\left[\left(Z(\boldsymbol{s}_0) - \sum_{i=1}^{N} \lambda_i Z(\boldsymbol{s}_i)\right)^2\right] - 2m\left(\sum_{i=1}^{N} \lambda_i - 1\right)$$

Define the $N \times 1$ column vector $\boldsymbol{\lambda}$ by

$$\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_N)'$$

and the $(N+1) \times 1$ column vector $\boldsymbol{\lambda}_0$ by

$$\boldsymbol{\lambda}_0 = (\lambda_1, \ldots, \lambda_N, m)' = \begin{pmatrix} \boldsymbol{\lambda} \\ m \end{pmatrix}$$

The optimization is performed by solving

$$\frac{\partial L}{\partial \boldsymbol{\lambda}_0} = \mathbf{0}$$

in terms of $\lambda_1, \ldots, \lambda_N$ and m.

The resulting matrix equation can be expressed in terms of either the covariance $C_z(\boldsymbol{h})$ or semivariogram $\gamma_z(\boldsymbol{h})$. In terms of the covariance, the preceding equation results in the matrix equation

$$\mathbf{C}\boldsymbol{\lambda}_0 = \mathbf{C}_0$$

where

$$\mathbf{C} = \begin{pmatrix} C_z(\mathbf{0}) & C_z(\boldsymbol{s}_1 - \boldsymbol{s}_2) & \cdots & C_z(\boldsymbol{s}_1 - \boldsymbol{s}_N) & 1 \\ C_z(\boldsymbol{s}_2 - \boldsymbol{s}_1) & C_z(\mathbf{0}) & \cdots & C_z(\boldsymbol{s}_2 - \boldsymbol{s}_N) & 1 \\ & & \ddots & & \\ C_z(\boldsymbol{s}_N - \boldsymbol{s}_1) & C_z(\boldsymbol{s}_N - \boldsymbol{s}_2) & \cdots & C_z(\mathbf{0}) & 1 \\ 1 & 1 & \cdots & 1 & 0 \end{pmatrix}$$

and

$$\mathbf{C_0} = \begin{pmatrix} C_z(\boldsymbol{s}_0 - \boldsymbol{s}_1) \\ C_z(\boldsymbol{s}_0 - \boldsymbol{s}_2) \\ \vdots \\ C_z(\boldsymbol{s}_0 - \boldsymbol{s}_N) \\ 1 \end{pmatrix}$$

The solution to the previous matrix equation is

$$\hat{\boldsymbol{\lambda}}_\mathbf{0} = \mathbf{C}^{-1}\mathbf{C_0}$$

Using this solution for $\boldsymbol{\lambda}$ and m, the ordinary kriging prediction at r_0 is

$$\hat{Z}(\boldsymbol{s}_0) = \lambda_1 Z(\boldsymbol{s}_1) + \cdots + \lambda_N Z(\boldsymbol{s}_N)$$

with associated prediction error the square root of the variance

$$\sigma_z{}^2(\boldsymbol{s}_0) = C_z(\mathbf{0}) - \boldsymbol{\lambda}'\mathbf{c_0} + m$$

where $\mathbf{c_0}$ is $\mathbf{C_0}$ with the 1 in the last row removed, making it an $N \times 1$ vector.

These formulas are used in the best linear unbiased prediction (BLUP) of random variables (Robinson 1991). Further details are provided in Cressie (1993, pp. 119–123).

Because of possible numeric problems when solving the previous matrix equation, Deutsch and Journel (1992) suggest replacing the last row and column of 1s in the preceding matrix $\mathbf{C}$ by $C_z(0)$, keeping the 0 in the $(N + 1, N + 1)$ position and similarly replacing the last element in the preceding right-hand vector $\mathbf{C_0}$ with $C_z(0)$. This results in an equivalent system but avoids numeric problems when $C_z(0)$ is large or small relative to 1.

Computational Resources

To generate a predicted value at a single grid point by using N data points, PROC KRIGE2D must solve the kriging system

$$\mathbf{C}\boldsymbol{\lambda}_0 = \mathbf{C}_0$$

where the dimensions of $\mathbf{C}$ are $(N+1) \times (N+1)$ and the right-hand-side $\mathbf{C}_0$ has one column.

Holding the matrix and vector associated with this system in core requires approximately $8N^2/2$ bytes. The CPU time used in solving the system is proportional to N^3. For large N, this time dominates the $O(N^2)$ time to compute the elements of the covariance matrix $\mathbf{C}$ from the specified covariance or semivariogram model.

For local kriging, the kriging system is set up and solved for each grid point. Part of the setup process involves determining the neighborhood of each grid point. A fast K-D tree algorithm determines neighborhoods. For G grid points, the dominant CPU time factor is setting up and solving the G kriging systems. The N in the algorithm of the section "Ordinary Kriging" on page 5595 is the number of data points in a given neighborhood, and it can differ for each grid point.

In global kriging, the entire input data set and all grid points set up and solve the single system

$$\mathbf{C}\boldsymbol{\lambda}_0 = \mathbf{C}_0$$

Again $\mathbf{C}$ has dimensions $(N+1) \times (N+1)$, but $\boldsymbol{\lambda}_0$ and $\mathbf{C}_0$ now have G columns, where G is the number of grid points. Memory requirements are approximately $8[(N^2/2) + GN]$ bytes. The CPU time used in solving the system is still dominated by the N^3 factorization of the left-hand side.

Output Data Sets

The KRIGE2D procedure produces two data sets: the OUTEST=*SAS-data-set* and the OUTNBHD=*SAS-data-set*. These data sets are described as follows.

OUTEST=*SAS-data-set*

The OUTEST= data set contains the kriging predictions and the associated standard errors. The OUTEST= data set contains the following variables:

- ESTIMATE, which is the kriging prediction for the current variable.
- GXC, which is the x coordinate of the grid point at which the kriging prediction is made.
- GYC, which is the y coordinate of the grid point at which the kriging prediction is made.
- LABEL, which is the label for the current PREDICT/MODEL combination that produces the kriging prediction. If you do not specify a label, default labels of the form Pred*j*.Model*k* are used.
- NPOINTS, which is the number of points used in the prediction. This number varies for each grid point if local kriging is performed.
- STDERR, which is the standard error of the kriging predict.
- VARNAME, which is the variable name.

OUTNBHD=*SAS-data-set*

When you specify the RADIUS= option or the NUMPOINTS= option in the PREDICT statement, local kriging is performed. Local kriging is simply ordinary kriging at a given grid location, using only those data points in a neighborhood defined by the RADIUS= value or the NUMPOINTS= value.

The OUTNBHD= data set contains one observation for each data point in each neighborhood. Hence, this data set can be large. For example, if the grid specification results in 1,000 grid points and each grid point has a neighborhood of 100 points, the resulting OUTNBHD= data set contains 100,000 points.

The OUTNBHD= data set contains the following variables:

- GXC, which is the *x* coordinate of the grid point.
- GYC, which is the *y* coordinate of the grid point.
- ID, which is the ID variable value or observation. number of the current data point
- LABEL, which is the label for the current PREDICT/MODEL combination. If you do not specify a label, default labels of the form Pred*j*.Model*k* are used.
- NPOINTS, which is the number of points used in the prediction.
- RADIUS, which is the radius used for each neighborhood.
- VALUE, which is the value of the variable at the current data point.
- VARNAME, which is the variable name of the current variable.
- XC, which is the *x* coordinate of the current data point.
- YC, which is the *y* coordinate of the current data point.

If no ID statement is specified, then the corresponding observation number is assigned to the variable ID, instead.

Displayed Output

In addition to the output data sets, the KRIGE2D procedure produces output objects as well. The KRIGE2D procedure output objects are the following:

- a default “Number of Observations” table that displays the number of observations read from the input data set and the number of observations used in the analysis.
- a map that shows the spatial distribution of the observations of the current VAR= variable in the PREDICT statement. The observations are displayed by default with circled markers whose color indicates the VAR= value at the corresponding location.
- a default table for each PREDICT statement that sums up basic information about the kriging analysis.
- a default table for each MODEL statement that shows the covariance model parameters for the corresponding PREDICT statement.

- plots of the kriging prediction and the prediction standard error at each point of the specified output grid or at specified individual locations. The KRIGE2D procedure produces by default a plot of the kriging prediction and the corresponding prediction error for each MODEL statement of every PREDICT statement that you specify. You can produce more of these plots with styles that you can specify by using the available suboptions of the PLOTS=PREDICTION option.
- a "Store Info" table with basic information about the input item store. This table is produced by default when you specify the RESTORE statement.
- a "Store Variables Information" table that describes the analysis variables of an input item store. The table is produced by default when you specify an item store with the RESTORE statement.
- a "Store Models Information" table with detailed information about the models and direction angles that are contained in an input item store. The table is produced by default when you specify an item store with the RESTORE statement.

ODS Table Names

Each table created by PROC KRIGE2D has a name associated with it, and you must use this name to reference the table when using ODS Graphics. These names are listed in Table 74.4.

Table 74.4 ODS Tables Produced by PROC KRIGE2D

ODS Table Name	Description	Statement	Option
KrigInfo	Kriging analysis general information	PROC KRIGE2D	Default output
ModelInfo	Parameters of the covariance model used in current kriging analysis	PROC KRIGE2D	Default output
NObs	Number of observations read and used	PROC KRIGE2D	Default output
StoreInfo	Input item store identity information	RESTORE	Default output
StoreModelInfo	Input item direction angles and models information	RESTORE	INFO
StoreVarInfo	Input item store variables and their statistics	RESTORE	INFO

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

For additional control of the graphics that are displayed, see the PLOTS option in the section "PROC KRIGE2D Statement" on page 5555.

ODS Graph Names

PROC KRIGE2D assigns a name to each graph it creates by using ODS Graphics. You can use these names to reference the graphs when using ODS Graphics. You must also specify the PLOTS= option indicated in Table 74.5.

Table 74.5 Graphs Produced by PROC KRIGE2D

ODS Graph Name	Plot Description	Statement	Option
ObservationsPlot	Scatter plot of observed data and colored markers indicating observed values	PROC KRIGE2D	PLOTS=OBSERV
PredictionPlot	Contour plots of the kriging prediction, surface of the prediction error, and outlines of the observation locations	PROC KRIGE2D	PLOTS=PREDICTION
Semivariogram	Plots of the semivariogram models used for all prediction tasks	PROC KRIGE2D	PLOTS=SEMIVAR

Examples: KRIGE2D Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 74.1: Spatial Prediction of Pollutant Concentration

(View the complete code for this example (kri2dex1.sas) in the example repository.)

The example in the section "Example 131.1: Aspects of Semivariogram Model Fitting" on page 11047 in Chapter 131, "The VARIOGRAM Procedure," investigates fitting of a theoretical model to describe spatial correlation in a study of 138 simulated arsenic logarithm concentration (logAs) observations. These observations form the logAsData data set, which is treated as actual data for illustration in the examples.

In this example, you use the logAsData data set and the semivariogram analysis results to predict the logAs variable values across space in a specified square region of size 500 km × 500 km. Your goal is to answer scientific questions in your analysis by means of your prediction results. This application highlights the impact of the correlation model choice on predictions. The example in section "Example 74.2: Investigating the Effect of Model Specification on Spatial Prediction" on page 5612 examines additional aspects of this impact.

The World Health Organization (WHO) standard for maximum arsenic concentration in drinking water is 10 μg/lt. Assume that you want to answer the following question: In what percentage of the study area does the Arsenic concentration exceed the WHO regulatory standard?

First, you read the logAsData data set with the following DATA step:

```
title 'Spatial Prediction of Log-Arsenic Concentration';

data logAsData;
   input East North logAs @@;
   label logAs='log(As) Concentration';
   datalines;
   193.0 296.6 -0.68153   232.6 479.1  0.96279   268.7 312.5 -1.02908
    43.6   4.9  0.65010   152.6  54.9  1.87076   449.1 395.8  0.95932
   310.9 493.6 -1.66208   287.8 164.9 -0.01779   330.0   8.0  2.06837
   225.7 241.7  0.15899   452.3  83.4 -1.21217   156.5 462.5 -0.89031
    11.5  84.4 -0.24496   144.4 335.7  0.11950   149.0 431.8 -0.57251
   234.3 123.2 -1.33642    37.8 197.8 -0.27624   183.1 173.9 -2.14558
   149.3 426.7 -1.06506   434.4  67.5 -1.04657   439.6 237.0 -0.09074
    36.4 175.2 -1.21211   370.6 244.0  3.28091   452.0  96.5 -0.77081
   247.0  86.8  0.04720   413.6 373.2  1.78235   253.5 291.7  0.56132
   129.7 111.9  1.34000   352.7  42.1  0.23621   279.3  82.7  2.12350
   382.6 290.7  0.86756   188.2 222.8 -1.23308   382.8 154.5 -0.94094
   304.4 309.2 -1.95158   337.5 387.2 -1.31294   490.7 189.8  0.40206
   159.0 100.1 -0.22272   245.5 329.2 -0.26082   372.1 379.5 -1.89078
   417.8  84.1 -1.25176   173.9 407.6 -0.24240   121.5 107.7  1.54509
   453.5 313.6  0.65895   143.5 346.7 -0.87196   157.4 125.5 -1.96165
   371.8 353.2 -0.59464   358.9 338.2 -1.07133     8.6 437.8  1.44203
   395.9 394.2 -0.24144   149.5  58.9  1.17459   453.5 420.6 -0.63951
```

```
   182.3  85.0  1.00005    21.0 290.1  0.31016    11.1 352.2 -0.88418
   131.2 238.4 -0.57184   104.9   6.3  1.12054   247.3 256.0  0.14019
   428.4 383.7  0.92448   327.8 481.1 -2.72543   199.2  92.8 -0.05717
   453.9 230.1  0.16571   205.0 250.6  0.07581   459.5 271.6  0.93700
   229.5 262.8  1.83590   370.4 228.6  2.96611   330.2 281.9  1.79723
   354.8 388.3 -3.18262   406.2 222.7  2.41594   254.4 393.1  2.03221
    96.7  85.2 -0.47156   407.2 256.8  0.66747   498.5 273.8  1.03041
   417.2 471.4 -1.42766   368.8 424.3 -0.70506   303.0  59.1  1.43070
   403.1 264.1  1.64554    21.2 360.8  0.67094   148.2  78.1  2.15323
   305.5 310.7 -1.47985   228.5 180.3 -0.68386   161.1 143.3  1.07901
    70.5 155.1  0.54652   363.1 282.6 -0.43051    86.0 472.5 -1.18855
   175.9 105.3 -2.08112    96.8 426.3  1.56592   475.1 453.1 -1.53776
   125.7 485.4  1.40054   277.9 201.6 -0.54565   406.2 125.0 -1.38657
    60.0 275.5 -0.59966   431.3 494.6 -0.36860   399.9 399.0 -0.77265
    28.8 311.1  0.91693   166.1 348.2 -0.49056   266.6  83.5  0.67277
    54.7 356.3  0.49596   433.5 460.3 -1.61309   201.7 167.6 -1.40678
   158.1 203.6 -1.32499    67.6 230.4  1.14672    81.9 250.0  0.63378
   372.0  50.7  0.72445    26.4 264.6  1.00862   300.1  91.7 -0.74089
   303.0 447.4  1.74589   108.4 386.2  1.12847    55.6 191.7  0.95175
    36.3 273.2  1.78880    94.5 298.3 -2.43320   366.1 187.3 -0.80526
   130.7 389.2 -0.31513    37.2 324.2  0.24489   295.5 211.8  0.41899
    58.6 206.2  0.18495   346.3 142.8 -0.92038   484.2 215.9  0.08012
   451.4 415.7  0.02773    58.9  86.5  0.17652   212.6 363.9  0.17215
   378.7 407.6  0.51516   265.9 305.0 -0.30718   123.2 314.8 -0.90591
    26.9 471.7  1.70285    16.5   7.1  0.51736   255.1 472.6  2.02381
   111.5 148.4 -0.09658   440.4 375.0  1.23285   406.4  19.5  1.01181
   321.2  65.8 -0.02095   466.4 357.1 -0.49272     2.0 484.6  0.50994
   200.9 205.1  0.43543    30.3 337.0  1.60882   297.0  12.7  1.79824
   158.2 450.7  0.05295   122.8 105.3  1.53936   417.8 329.7 -2.08124
;
```

For prediction of the logAs values in the specified area, assume a rectangular grid of nodes with an equal spacing of 5 km between neighboring nodes in the north and east directions. This produces a total of $101 \times 101 = 10201$ prediction locations.

In the section "Example 131.1: Aspects of Semivariogram Model Fitting" on page 11047 in Chapter 131, "The VARIOGRAM Procedure," you saved the selected fitted model that resulted from the correlation analysis into the SemivAsStore item store as shown in the following statements:

```
ods graphics on;

proc variogram data=logAsData plots=none;
   store out=SemivAsStore / label='LogAs Concentration Models';
   compute lagd=5 maxlag=40;
   coord xc=East yc=North;
   model form=auto(mlist=(exp,gau,mat) nest=1 to 2);
   var logAs;
run;
```

In the KRIGE2D procedure you specify the name of the item store you want to use for prediction input in the IN= option of the RESTORE statement. You request use of the selected model for prediction by specifying the STORESELECT option in the MODEL statement.

The INFO option of the RESTORE statement produces a table with information about the selected fitted model in the item store. To review all models in the input item store, specify the two INFO option suboptions.

In particular, specify the DET suboption to request details about all additional fitted models that are included in the item store and the ONLY suboption to suppress prediction and produce only the tables about the item store, as shown in the following statements:

```
proc krige2d data=logAsData outest=pred plots=none;
   restore in=SemivAsStore / info(det only);
   coordinates xc=East yc=North;
   predict var=logAs;
   model storeselect;
   grid x=0 to 500 by 5 y=0 to 500 by 5;
run;
```

PROC KRIGE2D produces a table with general information about the input item store identity, as shown in Output 74.1.1.

Output 74.1.1 PROC KRIGE2D and Input Item Store General Information

Spatial Prediction of Log-Arsenic Concentration

The KRIGE2D Procedure

Correlation Model Item Store Information	
Input Item Store	WORK.SEMIVASSTORE
Item Store Label	LogAs Concentration Models
Data Set Created From	WORK.LOGASDATA
By-group Information	No By-groups Present
Created By	PROC VARIOGRAM
Date Created	22APR20:03:02:30

The second table in Output 74.1.2 itemizes the variables in the item store and displays the sample mean and standard deviation of their data set of origin. Hence, the values shown in Output 74.1.2 refer to the observations in the logAsData data set.

Output 74.1.2 Variables in the Input Item Store

Item Store Variables		
Variable	**Mean**	**Std Deviation**
logAs	0.084309	1.527707

The table in Output 74.1.3 presents all the correlation models fitted to the arsenic logarithm logAs empirical semivariance that are saved in the SemivAsStore item store.

Output 74.1.3 Angle and Models Information in the Input Item Store

Item Store Models For logAs	
Class	**Model**
1	Gau-Gau
	Gau-Mat
	Mat-Mat
2	Exp-Gau
3	Exp-Mat
4	Mat
5	Gau
6	Exp
	Exp-Exp
	Mat-Exp
	Gau-Exp

According to Output 74.1.3, the Gaussian-Gaussian model is the selected model for the empirical semivariance fit based on the specific weighted least squares fit and ranking criteria. In the section "Example 131.1: Aspects of Semivariogram Model Fitting" on page 11047 in Chapter 131, "The VARIOGRAM Procedure," it is noted that all fitted models in the first five equivalence classes produce very similar semivariograms, and this is likely to lead to similar results in prediction analysis. For comparison purposes, you choose to examine the selected model, in addition to the exponential model in the SemivAsStore item store. As shown in Output 74.1.3, the exponential model is one of the least well-fit models based on the criteria used for the specific fit. You are interested in comparing the predictions from each one of these two models, and you examine their impact on your analysis.

The default item store model selection is the model on top of the list in Output 74.1.3. Hence, you specify the STORESELECT option in the MODEL statement without any suboptions, and it invokes the Gaussian-Gaussian model from the SemivAsStore item store. You assign the label SELMODEL to the corresponding MODEL statement.

You also specify a second MODEL statement with the label EXPMODEL to request prediction based on the exponential correlation form. In this case you specify the STORESELECT(MODEL=) option in the MODEL statement to request the desired form.

You omit the INFO option from the RESTORE statement. You specify the PRED and the SEMIVAR options in the PLOTS option of the PROC KRIGE2D statement to produce plots of the predicted values and the semivariance model, respectively, for each MODEL statement. You request that the prediction output be saved in the Pred output data set.

You satisfy the preceding requests by specifying the following statements:

```
proc krige2d data=logAsData outest=Pred plots(only)=(pred semivar);
   restore in=SemivAsStore;
   coordinates xc=East yc=North;
   predict var=logAs;
   SelModel: model storeselect;
   ExpModel: model storeselect(model=exp);
   grid x=0 to 500 by 5 y=0 to 500 by 5;
run;
```

When you run these statements, in addition to the input item store information table, PROC KRIGE2D also produces the number of observations table and general kriging process information, as shown in Output 74.1.4.

Output 74.1.4 Number of Observations and Kriging Information Tables

Spatial Prediction of Log-Arsenic Concentration

The KRIGE2D Procedure

Dependent Variable: logAs

Number of Observations Read	138
Number of Observations Used	138

Kriging Information	
Prediction Grid Points	10201
Type of Analysis	Global

PROC KRIGE2D first uses the Gaussian-Gaussian model. The table in Output 74.1.5 shows the saved parameter values of the fitted Gaussian-Gaussian model in the SemivAsStore item store. PROC KRIGE2D uses these parameters for the prediction based on the selected model.

Output 74.1.5 Information about the Gaussian-Gaussian Model

Spatial Prediction of Log-Arsenic Concentration

The KRIGE2D Procedure

Dependent Variable: logAs
Prediction: Pred1, Model: SelModel

Covariance Model Information for SelModel	
Nested Structure 1 Type	Gaussian
Nested Structure 1 Sill	0.3276621
Nested Structure 1 Range	62.313012
Nested Structure 1 Effective Range	107.9293
Nested Structure 2 Type	Gaussian
Nested Structure 2 Sill	1.2615484
Nested Structure 2 Range	21.459555
Nested Structure 2 Effective Range	37.16904
Nugget Effect	0.083075

The semivariogram of the Gaussian-Gaussian model with the parameters shown in Output 74.1.5 is depicted in Output 74.1.6.

Output 74.1.6 Gaussian-Gaussian Semivariogram Model Used in Kriging Predictions

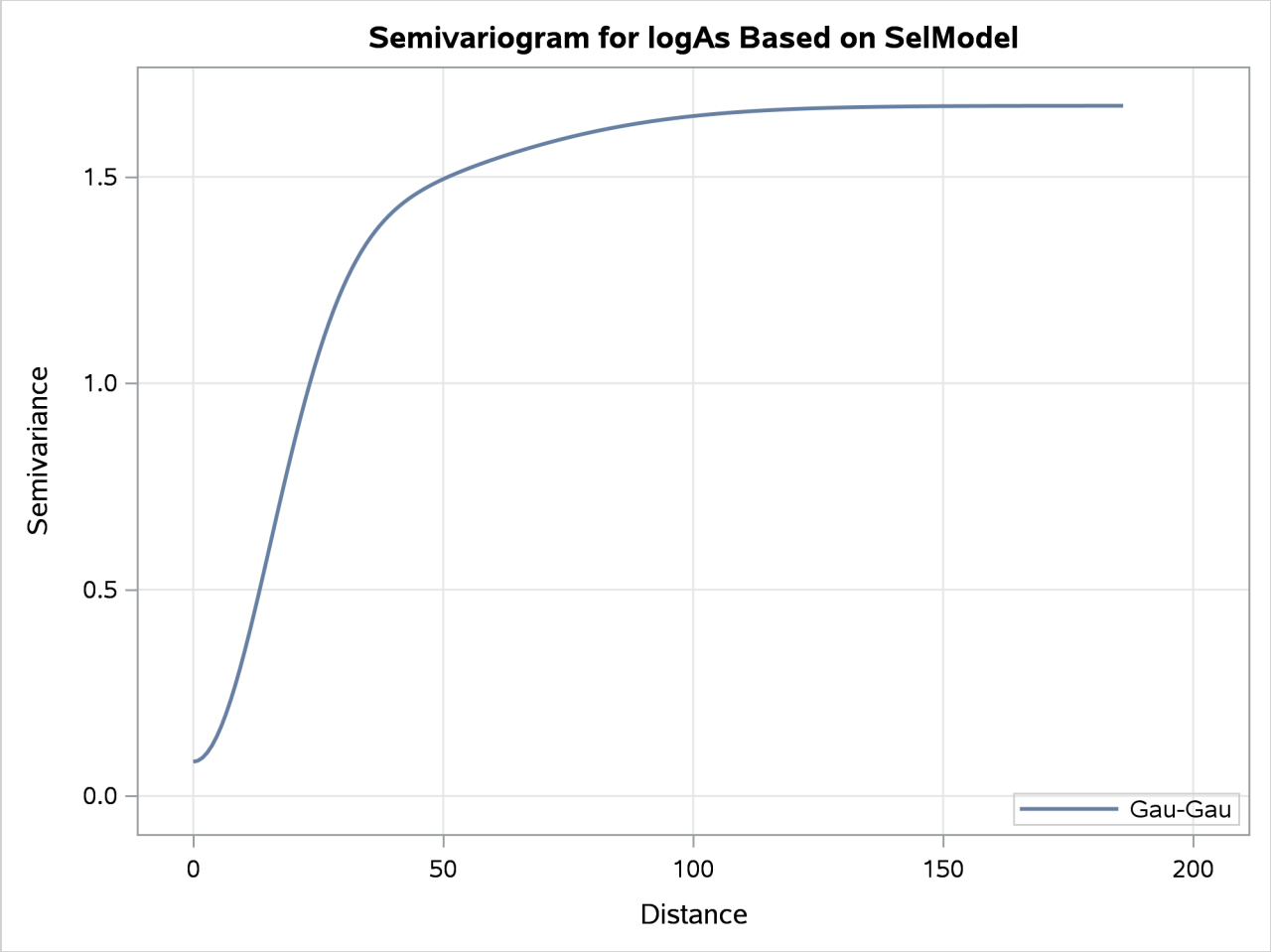

Output 74.1.7 is a map of the kriging prediction of the arsenic concentration values logAs in the specified domain. The prediction error surface shows a naturally increasing error as you move farther away from the observation locations. Interestingly, kriging predicts a small area of increased arsenic concentration values located in the central-eastern part of the domain. The WHO threshold of 10 μg/lt for the maximum allowed arsenic concentration in water translates into about 2.3 in the log scale, and the particular area exhibits values in excess of 3. Due to the suggested violation of the WHO standard, this particular area is very likely to be the focus of further environmental risk analysis.

Output 74.1.7 Predicted Arsenic Logarithm Values with Gaussian-Gaussian Covariance

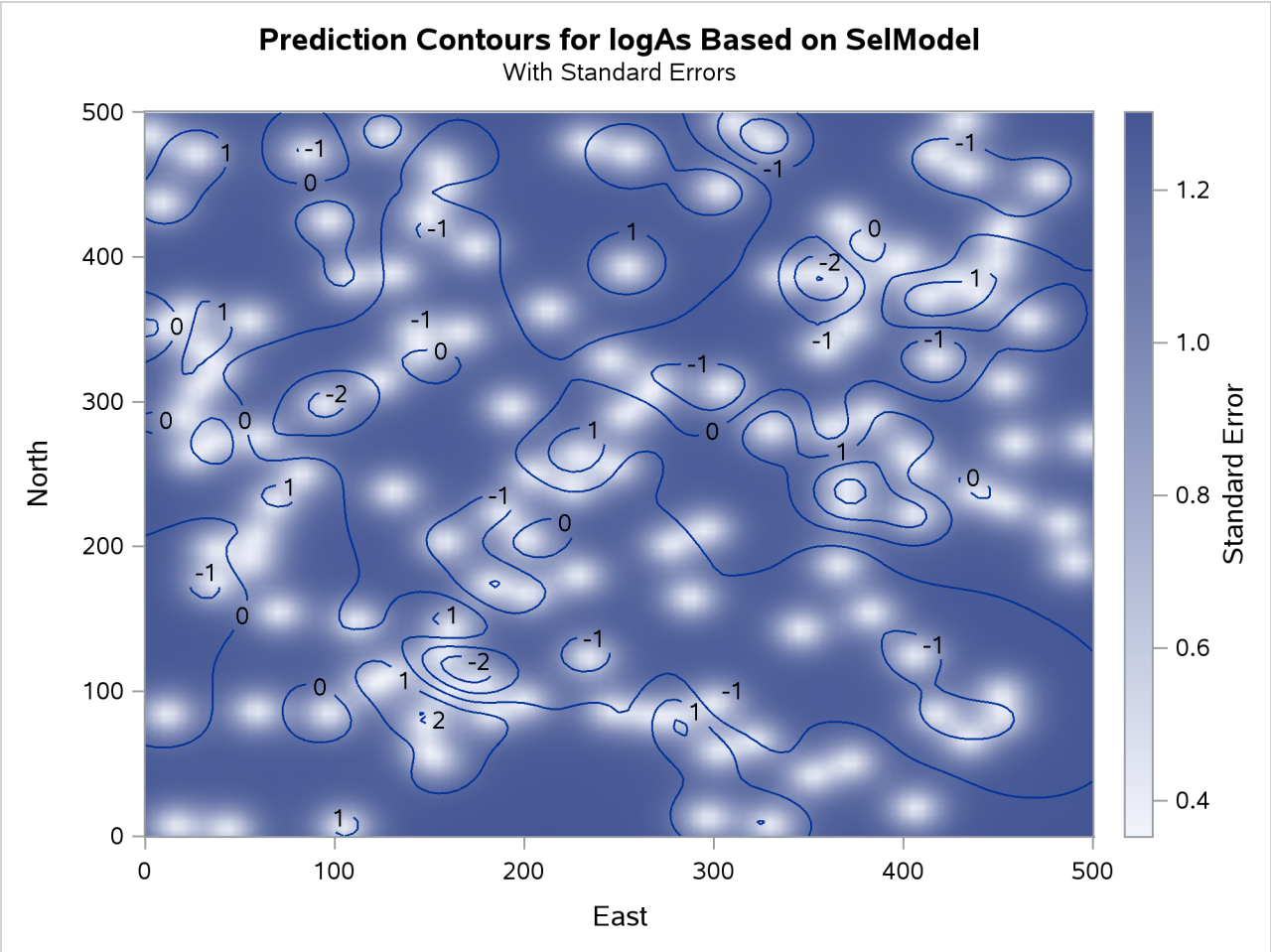

Next, PROC KRIGE2D performs prediction with the exponential model. The model parameters are also read from the SemivAsStore item store and are shown in Output 74.1.8.

Output 74.1.8 Information about the Exponential Model

Spatial Prediction of Log-Arsenic Concentration

The KRIGE2D Procedure

Dependent Variable: logAs
Prediction: Pred1, Model: ExpModel

Covariance Model Information for ExpModel	
Type	Exponential
Sill	1.6779788
Range	24.537294
Effective Range	73.611882
Nugget Effect	3.601E-19

Output 74.1.9 illustrates the semivariogram of the nested exponential model where its parameter values are those shown in Output 74.1.8.

Output 74.1.9 Exponential Semivariogram Model Used in Kriging Predictions

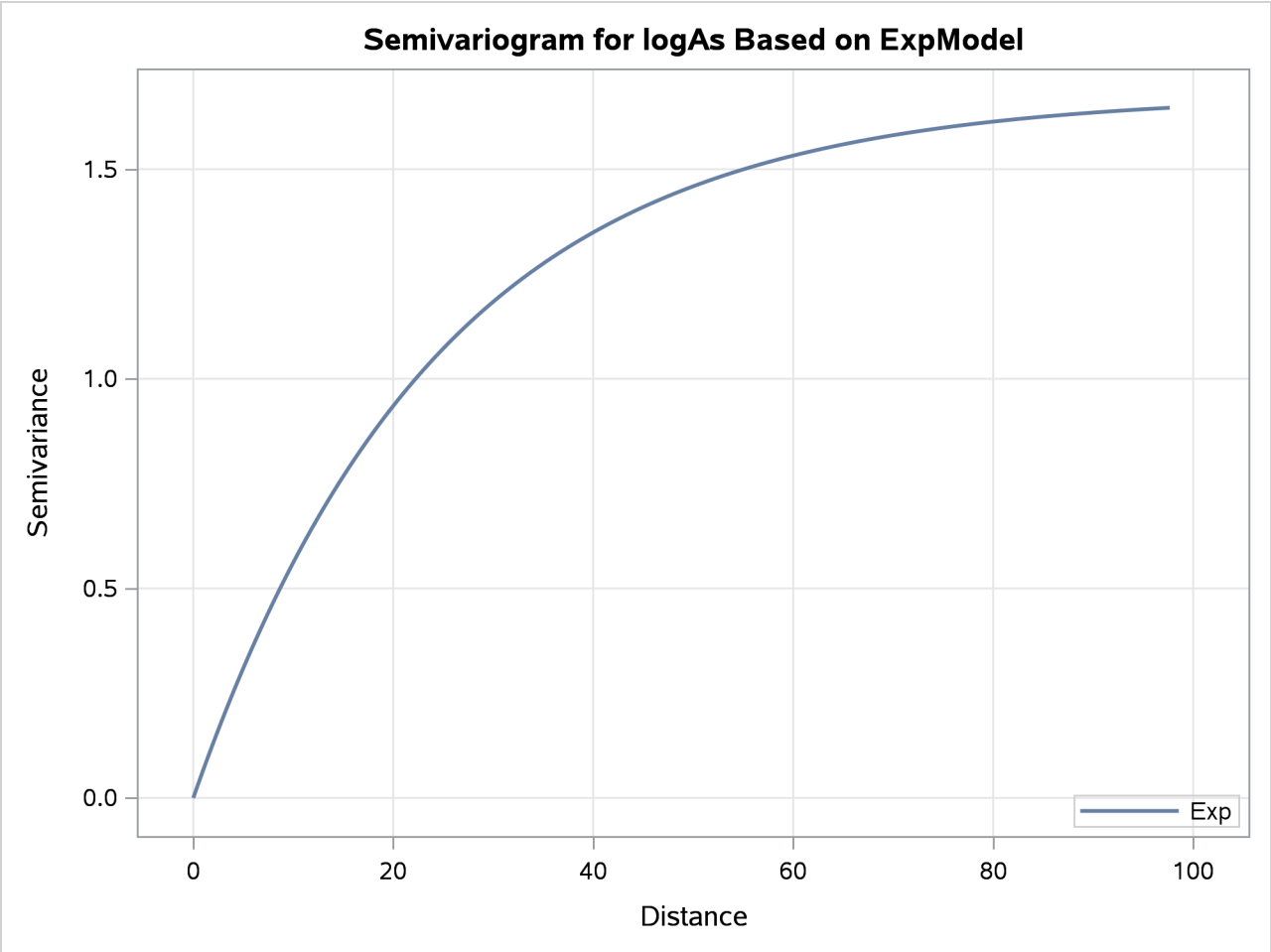

The prediction plot for the exponential model is shown in Output 74.1.10. Prediction values and spatial patterns are similar overall to those of the Gaussian-Gaussian case. Clearly, although both models predict the same basic characteristics for the arsenic logarithm concentration distribution, the exponential model suggests a more limited spatial variability in closely neighboring locations. The lack of a nugget effect in the exponential model justifies this behavior. Also, the exponential model predictions seem less inclined to deviate farther away from the near-zero mean than the Gaussian-Gaussian model predictions.The prediction error reaches about the same upper values for both models, though its low values are slightly smaller in the exponential model.

Output 74.1.10 Predicted Arsenic Logarithm Values with Exponential Covariance

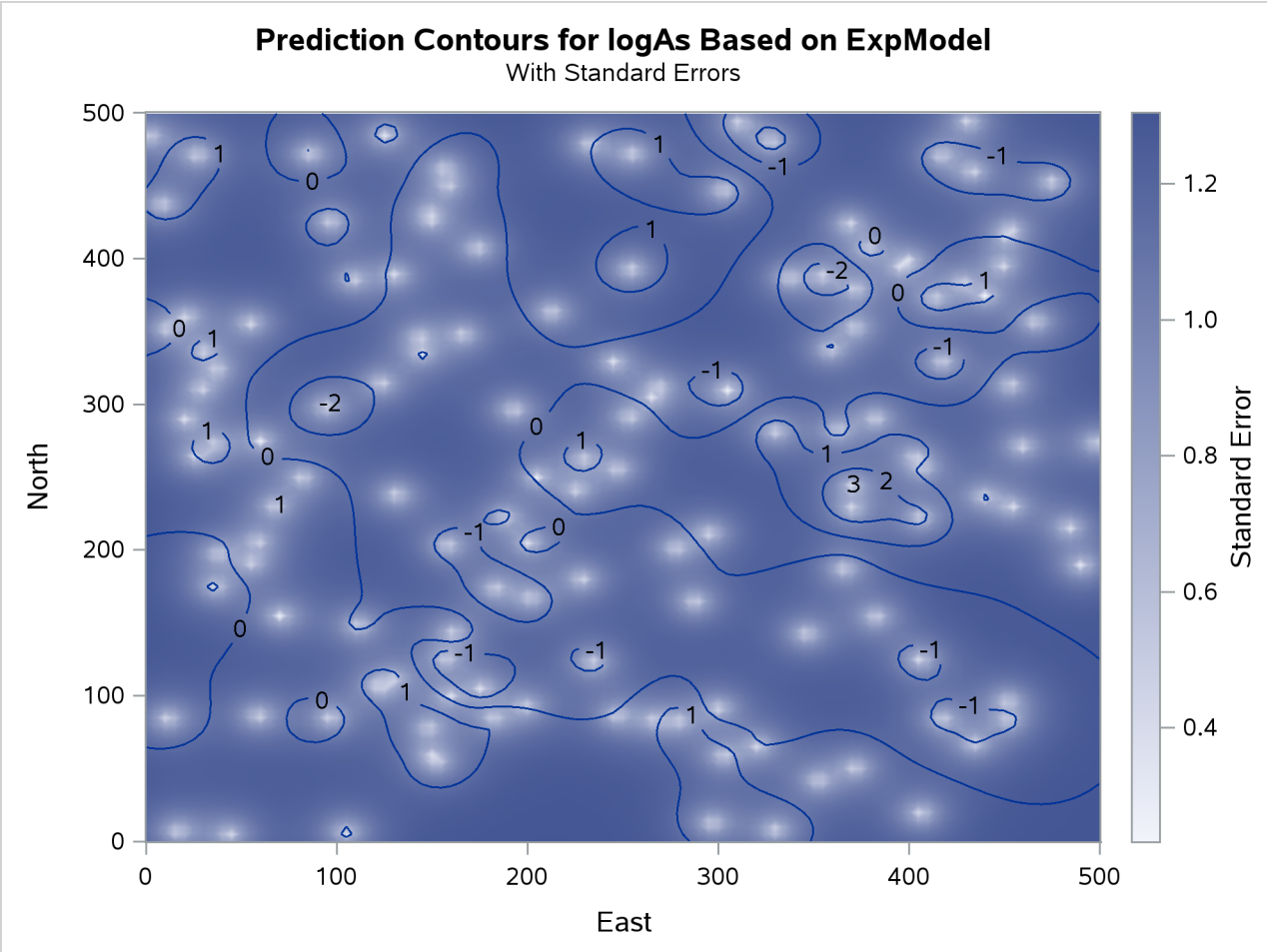

In the following two-step computation, you proceed to compute the percentage of the study area where the arsenic concentration exceeds the WHO regulatory standard according to your predictions. First, a DATA step marks the arsenic predicted values in excess of the WHO concentration threshold of 10 μg/lt and saves the outcome into an indicator variable OverLimit. The DATA step input is the prediction Pred output data set, where the logarithm arsenic prediction is stored in the estimate variable. The DATA step also transforms the arsenic logarithm values back into arsenic concentration values to compare them to the threshold value. You use the following statements:

```
data AsOverLimit;
   set Pred;
   OverLimit = (exp(estimate) > 10) * 100;
run;
```

The second step uses the MEANS procedure to express the selected nodes population, where the WHO arsenic concentration limit violation occurs, as a percentage of the entire domain area. You study the results of each correlation model separately by specifying the BY statement in the PROC MEANS. The BY variable is the Label variable in the AsOverLimit and Pred data sets. You need to sort the AsOverLimit data prior to using PROC MEANS. You run the following statements:

```
proc sort data=AsOverLimit;
   by Label;
run;
proc means data=AsOverLimit mean;
   var OverLimit;
   by label;
   label Overlimit="Percent above WHO threshold";
run;

ods graphics off;
```

The Gaussian-Gaussian model prediction produces the result in Output 74.1.11. The analysis suggests a minimal occurrence of excessive arsenic concentration in drinking water in about 0.43% of the study region.

Output 74.1.11 Violation of Arsenic Concentration Threshold Using Gaussian-Gaussian Model

Spatial Prediction of Log-Arsenic Concentration

The MEANS Procedure

Label for the PREDICT/MODEL combination=Pred1.SelModel

Analysis Variable : OverLimit Percent above WHO threshold
Mean
0.4313303

The exponential model predicts that the WHO arsenic concentration threshold is exceeded in about 0.27% of the domain, as shown in Output 74.1.12. Although this is still a minimal occurrence of the threshold violation across the region, the exponential model estimates the impact to be at about two thirds of the Gaussian-Gaussian model percentage.

Output 74.1.12 Violation of Arsenic Concentration Threshold Using Exponential Model

Spatial Prediction of Log-Arsenic Concentration

The MEANS Procedure

Label for the PREDICT/MODEL combination=Pred1.ExpModel

Analysis Variable : OverLimit Percent above WHO threshold
Mean
0.2744829

The results in Output 74.1.11 and Output 74.1.12 suggest that it might not be possible to provide a unique answer about the area percentage that is affected by increased arsenic concentration. You chose to examine two different correlation models whose performance is relatively similar, and they provide impact estimates that differ by about 37%.

You might conclude that the answer to the initial question about the percentage value lies in the neighborhood

of the results given by the two correlation models. Further analysis with more models is necessary to validate this assumption. It is important to note that apart from the continuity model choice, additional factors contribute to this investigation. Such factors could be the use of local instead of global kriging, or even going back to the empirical semivariogram computation stage and repeating the analysis for different possible spatial continuity empirical estimates. A sensible approach to tackle this analysis would be to investigate the range of the impact suggested by all candidate correlation models and to proceed by defining the best and worst case scenarios for the size of the affected area.

Eventually, when it comes to using your findings, it is important to account for the subjective nature of stochastic analysis and multiple possible answers to your questions. In that sense, some scientific questions might be more sensible than others to interpret your results correctly. For instance, you might want to investigate only whether the adversely affected domain percentage is below 1%, rather than attempting to provide a specific value for it. Then, you might consider the preceding findings sufficient, despite any fluctuations in the estimated percentage. In a different scenario, the areas with high pollutant concentration could be populated. Hence, any local health standard violation is probably unacceptable, and it can be crucial that you provide solid and more detailed assessment in that case.

The section "Example 112.3: Risk Analysis with Simulation" on page 9541 in Chapter 112, "The SIM2D Procedure," investigates a different aspect of this study and offers additional perspective about spatial analysis.

Example 74.2: Investigating the Effect of Model Specification on Spatial Prediction

(View the complete code for this example (kri2dex2.sas) in the example repository.)

It is generally believed that spatial prediction is robust against model specification, while the standard error computation is not so robust. This example investigates the effect of using these different models on the prediction and associated standard errors.

In the section "Theoretical Semivariogram Model Fitting" on page 10971 in Chapter 131, "The VARIOGRAM Procedure," a particular theoretical semivariogram is fitted to the coal seam thickness data empirical semivariogram. The chosen semivariogram is Gaussian with a scale (sill) of $c_0 = 7.2881$ and a range of $a_0 = 30.6239$.

Another possible model choice could be the spherical semivariogram. First, use a DATA step to input the thickness data:

```
title 'Effect of Model Specification on Prediction';

data thick;
   input East North Thick @@;
   label Thick='Coal Seam Thickness';
   datalines;
    0.7  59.6  34.1   2.1  82.7  42.2   4.7  75.1  39.5
    4.8  52.8  34.3   5.9  67.1  37.0   6.0  35.7  35.9
    6.4  33.7  36.4   7.0  46.7  34.6   8.2  40.1  35.4
   13.3   0.6  44.7  13.3  68.2  37.8  13.4  31.3  37.8
   17.8   6.9  43.9  20.1  66.3  37.7  22.7  87.6  42.8
   23.0  93.9  43.6  24.3  73.0  39.3  24.8  15.1  42.3
   24.8  26.3  39.7  26.4  58.0  36.9  26.9  65.0  37.8
```

```
   27.7  83.3  41.8  27.9  90.8  43.3  29.1  47.9  36.7
   29.5  89.4  43.0  30.1   6.1  43.6  30.8  12.1  42.8
   32.7  40.2  37.5  34.8   8.1  43.3  35.3  32.0  38.8
   37.0  70.3  39.2  38.2  77.9  40.7  38.9  23.3  40.5
   39.4  82.5  41.4  43.0   4.7  43.3  43.7   7.6  43.1
   46.4  84.1  41.5  46.7  10.6  42.6  49.9  22.1  40.7
   51.0  88.8  42.0  52.8  68.9  39.3  52.9  32.7  39.2
   55.5  92.9  42.2  56.0   1.6  42.7  60.6  75.2  40.1
   62.1  26.6  40.1  63.0  12.7  41.8  69.0  75.6  40.1
   70.5  83.7  40.9  70.9  11.0  41.7  71.5  29.5  39.8
   78.1  45.5  38.7  78.2   9.1  41.7  78.4  20.0  40.8
   80.5  55.9  38.7  81.1  51.0  38.6  83.8   7.9  41.6
   84.5  11.0  41.5  85.2  67.3  39.4  85.5  73.0  39.8
   86.7  70.4  39.6  87.2  55.7  38.8  88.1   0.0  41.6
   88.4  12.1  41.3  88.4  99.6  41.2  88.8  82.9  40.5
   88.9   6.2  41.5  90.6   7.0  41.5  90.7  49.6  38.9
   91.5  55.4  39.0  92.9  46.8  39.1  93.4  70.9  39.7
   55.8  50.5  38.1  96.2  84.3  40.3  98.2  58.2  39.5
;
```

Fitting of the Gaussian model is performed in the section "Theoretical Semivariogram Model Fitting" on page 10971 in Chapter 131, "The VARIOGRAM Procedure," and the fitting parameters are saved in the SemivStoreGau item store with the following statements:

```
ods graphics on;

proc variogram data=thick noprint;
   store out=SemivStoreGau / label='Thickness Gaussian Model';
   compute lagd=7 maxlag=10;
   coord xc=East yc=North;
   model form=gau;
   var Thick;
run;
```

For prediction with the saved Gaussian model, you use the following statements to run the KRIGE2D procedure with input from the SemivStoreGau item store. You invoke the item store with the RESTORE statement. The STORESELECT option in the MODEL statement that specifies that you want to use the selected model in the item store as input for your prediction.

```
proc krige2d data=thick outest=pred1 noprint;
   restore in=SemivStoreGau;
   coordinates xc=East yc=North;
   predict var=Thick r=60;
   model storeselect;
   grid x=0 to 100 by 10 y=0 to 100 by 10;
run;
```

Then, you run the KRIGE2D procedure by using a spherical model. Start by using the VARIOGRAM procedure to fit a spherical model to the thick data set empirical semivariogram. You specify the STORE statement again in PROC VARIOGRAM to save the spherical model estimated parameters in an item store with the name SemivStoreSph. You use the following statements:

```
proc variogram data=thick plots(only)=fit;
   store out=SemivStoreSph / label='Thickness Sph Model';
   compute lagd=7 maxlag=10;
   coord xc=East yc=North;
   model form=sph;
   var Thick;
run;
```

The VARIOGRAM procedure fits the spherical model successfully, and the estimated parameters for this fit are shown in Output 74.2.1.

Output 74.2.1 Spherical Model Fitting Parameter Estimates

Effect of Model Specification on Prediction

The VARIOGRAM Procedure

Dependent Variable: Thick
Angle: Omnidirectional
Current Model: Spherical

Parameter Estimates

Parameter	Estimate	Approx Std Error	DF	t Value	Approx Pr > |t|
Nugget	8.22E-18	0	8	.	.
Scale	7.1914	0.2827	8	25.44	<.0001
Range	63.2351	4.1050	8	15.40	<.0001

The fit summary is displayed in Output 74.2.2. When compared to the corresponding result in the section "Theoretical Semivariogram Model Fitting" on page 10971 in Chapter 131, "The VARIOGRAM Procedure," the goodness-of-fit criteria indicate a worse statistical fit for the spherical model compared to the Gaussian.

Output 74.2.2 Spherical Model Fit Summary

Fit Summary

Model	Weighted SSE	AIC
Sph	52.26791	23.14336

Output 74.2.3 suggests an acceptable fit of the spherical model to the thick data set. Obviously, the fit of the spherical model in the sensitive area near the semivariogram origin is less faithful to the empirical semivariance than the Gaussian model. The following analysis explores the consequence in the kriging prediction of this discrepancy.

Output 74.2.3 Fitted Spherical and Empirical Thick Semivariogram

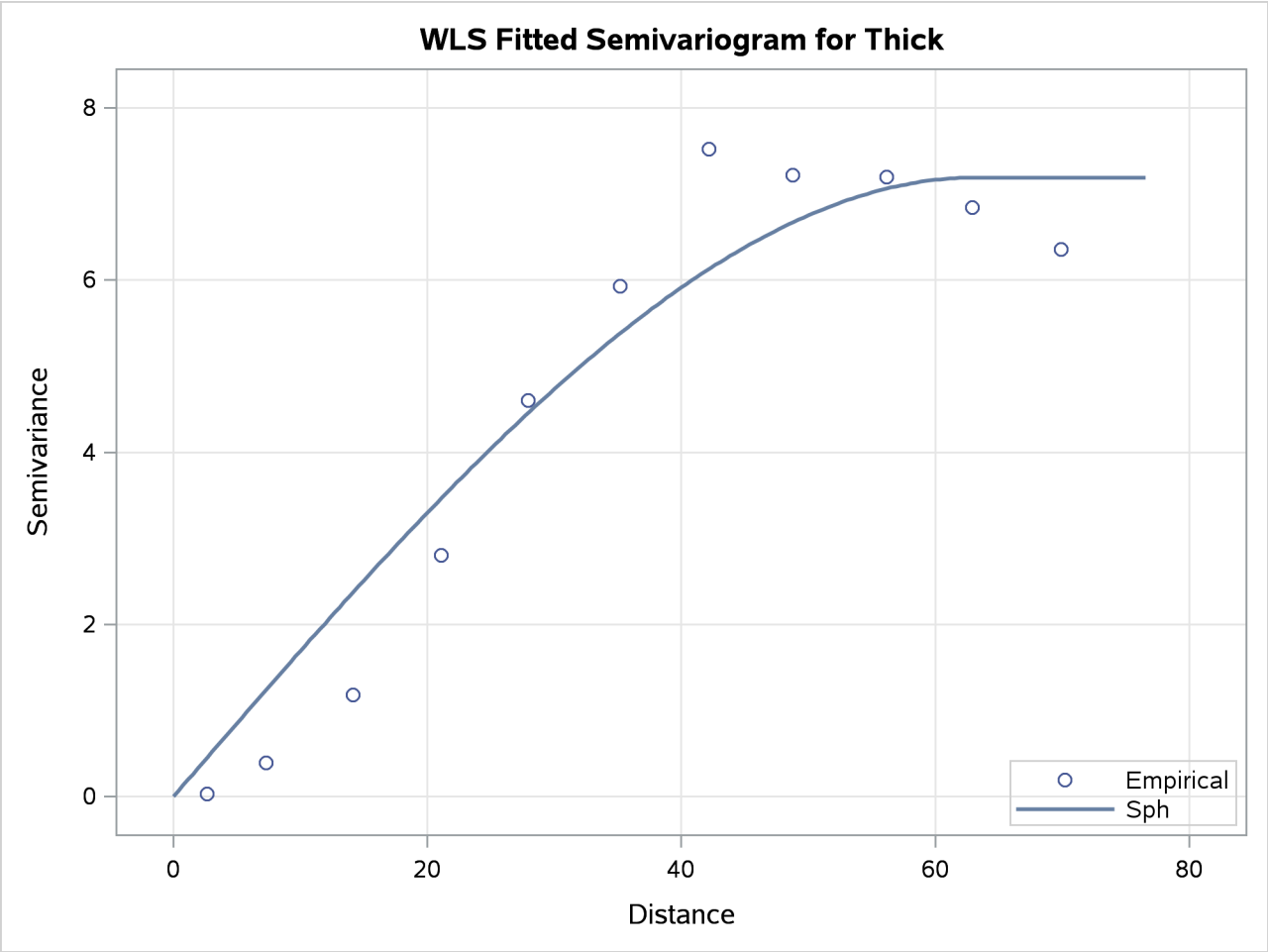

For the next step, you run the KRIGE2D procedure by using the spherical model parameters stored in the SemivStoreSph item store. You use the following statements:

```
proc krige2d data=thick outest=pred2 noprint;
   restore in=SemivStoreSph;
   coordinates xc=East yc=North;
   predict var=Thick r=60;
   model storeselect;
   grid x=0 to 100 by 10 y=0 to 100 by 10;
run;
```

Eventually, you compare the prediction results and errors of the two models. You use a DATA step to compute the relative difference of the predicted values and the prediction error for each one of the Gaussian and the spherical models. You store the prediction relative difference in the prdRelDiff variable and the prediction relative error in the stdRelDiff variable. You save the output in the compare data set with the following statements:

```
data compare;
   merge pred1(rename=(estimate=g_prd stderr=g_std))
         pred2(rename=(estimate=s_prd stderr=s_std));
   prdRelDif = ((g_prd-s_prd) / s_prd) * 100;
   stdRelDif = ((g_std-s_std) / s_std) * 100;
run;
```

The MEANS procedure uses the compare data set to produce statistics about the prediction relative difference and error for each one of the prdRelDiff and stdRelDiff variables with the following statements:

```
proc means data=compare;
   var prdRelDif stdRelDif;
run;

ods graphics off;
```

Output 74.2.4 shows that on average the predicted values are very close for the two semivariogram models. The mean relative difference in the prediction values is close to zero with a low standard deviation, whereas the relative difference values fluctuate with an absolute maximum of about 5%.

However, note that the mean relative standard error is about -96%. According to the definition of the stdRelDiff variable, the high negative value indicates that the prediction error difference between the two models is very close to the spherical model prediction error. Hence, the prediction standard error of the spherical model is substantially larger than that of the Gaussian model. In fact, the prediction relative error never gets smaller than about 66% for the two models, where the negative sign in the Minimum and Maximum columns in Output 74.2.4 means that the prediction error is always greater for the spherical model.

Output 74.2.4 Comparison of Gaussian and Spherical Models

Effect of Model Specification on Prediction

The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
prdRelDif	121	-0.0544593	1.3384023	-5.0751449	5.1926236
stdRelDif	121	-96.2515099	5.9400029	-99.8974418	-65.9275907

Example 74.3: Data Quality and Prediction with Missing Values

(View the complete code for this example (kri2dex3.sas) in the example repository.)

Kriging methods depend primarily on your data. The quantity and quality of your observations are important factors in minimizing prediction errors and increasing accuracy in your prediction analysis.

A typical aspect of data quality is measurement accuracy. In principle, the accuracy level of your data is not a parameter in kriging prediction; kriging assumes by definition that your data are perfectly accurate (hard) measurements. Whether you accept this assumption depends on your application. For example, an instrumentation error of $\pm 1\%$ in the data values might be regarded as considerable in one case, whereas the same level of uncertainty might be trivial within a different framework. Your experience and judgment are crucial when you consider whether observations in a data set might be too noisy for kriging predictions to be useful.

A second aspect of data quality involves the spatial arrangement of your observations. You need to have a sufficient number of observations in order to perform spatial prediction. Also, a key element in minimizing prediction errors is an adequate sampling density. Interpretation of the expressions "sufficient number" and "adequate sampling" is again case-specific. In any event, you want enough measurements so that you can deduce the underlying spatial correlation in the working domain; see also the discussion in the section "Choosing the Size of Classes" on page 11023 in Chapter 131, "The VARIOGRAM Procedure."

This example focuses on the effects of different sampling densities on the prediction analysis. The demonstration is a slight variation of the example in the section "Getting Started: KRIGE2D Procedure" on page 5549. Specifically, you use the same correlation structure and prediction grid. However, the thick data set, is modified as follows: three values in the central area of the grid are assumed missing, namely the observation values at locations $\boldsymbol{s}_1 = (x_1, y_1) = (55.8, 50.5)$, $\boldsymbol{s}_2 = (x_2, y_2) = (52.8, 68.9)$, and $\boldsymbol{s}_3 = (x_3, y_3) = (52.9, 32.7)$. These locations have been selected so that an extended area without observations is created in the domain. The following DATA step is the input for the modified thick data set:

```
title 'Kriging Prediction in the Presence of Missing Values';

data thick;
   input East North Thick @@;
   label Thick='Coal Seam Thickness';
   datalines;
    0.7  59.6  34.1   2.1  82.7  42.2   4.7  75.1  39.5
    4.8  52.8  34.3   5.9  67.1  37.0   6.0  35.7  35.9
    6.4  33.7  36.4   7.0  46.7  34.6   8.2  40.1  35.4
   13.3   0.6  44.7  13.3  68.2  37.8  13.4  31.3  37.8
   17.8   6.9  43.9  20.1  66.3  37.7  22.7  87.6  42.8
   23.0  93.9  43.6  24.3  73.0  39.3  24.8  15.1  42.3
   24.8  26.3  39.7  26.4  58.0  36.9  26.9  65.0  37.8
   27.7  83.3  41.8  27.9  90.8  43.3  29.1  47.9  36.7
   29.5  89.4  43.0  30.1   6.1  43.6  30.8  12.1  42.8
   32.7  40.2  37.5  34.8   8.1  43.3  35.3  32.0  38.8
   37.0  70.3  39.2  38.2  77.9  40.7  38.9  23.3  40.5
   39.4  82.5  41.4  43.0   4.7  43.3  43.7   7.6  43.1
   46.4  84.1  41.5  46.7  10.6  42.6  49.9  22.1  40.7
   51.0  88.8  42.0  52.8  68.9     .  52.9  32.7     .
   55.5  92.9  42.2  56.0   1.6  42.7  60.6  75.2  40.1
   62.1  26.6  40.1  63.0  12.7  41.8  69.0  75.6  40.1
   70.5  83.7  40.9  70.9  11.0  41.7  71.5  29.5  39.8
   78.1  45.5  38.7  78.2   9.1  41.7  78.4  20.0  40.8
   80.5  55.9  38.7  81.1  51.0  38.6  83.8   7.9  41.6
   84.5  11.0  41.5  85.2  67.3  39.4  85.5  73.0  39.8
   86.7  70.4  39.6  87.2  55.7  38.8  88.1   0.0  41.6
   88.4  12.1  41.3  88.4  99.6  41.2  88.8  82.9  40.5
   88.9   6.2  41.5  90.6   7.0  41.5  90.7  49.6  38.9
   91.5  55.4  39.0  92.9  46.8  39.1  93.4  70.9  39.7
   55.8  50.5     .  96.2  84.3  40.3  98.2  58.2  39.5
;

ods graphics on;
```

NOTE: Here you assume prior knowledge of the correlation structure model, because its parameters are based on the complete thick data set. A covariance model extracted from the incomplete set with the missing values would be a covariance model coming from a different data set; hence, it is likely to have different

parameters.

After you define the modified data set, you run PROC KRIGE2D and request the OBSERVATIONS plot with the SHOWMISSING suboption. You also request two instances of the PREDICTION plot: one that displays the prediction surface and contours, and another that plots the kriging standard error surface and contours. In both of these PREDICTION plots you specify that the observations be shown as gradient markers with outlines. The following statements compute the kriged predictions and produce the requested graphics:

```
proc krige2d data=thick outest=predictions
             plots(only)=(observ(showmissing)
                          pred(fill=pred line=pred obs=linegrad)
                          pred(fill=se line=se obs=linegrad));
   coordinates xc=East yc=North;
   predict var=Thick r=60;
   model scale=7.4599 range=30.1111 form=gauss;
   grid x=0 to 100 by 2.5 y=0 to 100 by 2.5;
run;

ods graphics off;
```

The number of observations table indicates the three missing values in Output 74.3.1.

Output 74.3.1 Number of Observations for the Modified thick Data Set

Kriging Prediction in the Presence of Missing Values

The KRIGE2D Procedure

Dependent Variable: Thick

Number of Observations Read	75
Number of Observations Used	72

Output 74.3.2 is a scatter plot of the modified observed data. The SHOWMISSING suboption produces marks in the observations plot that conveniently indicate the locations s_1, s_2, and s_3 of the missing values. Consequently, Output 74.3.2 displays an extended area with no observed Thick values in the central part of the domain.

Output 74.3.2 Scatter Plot of the Observations Spatial Distribution

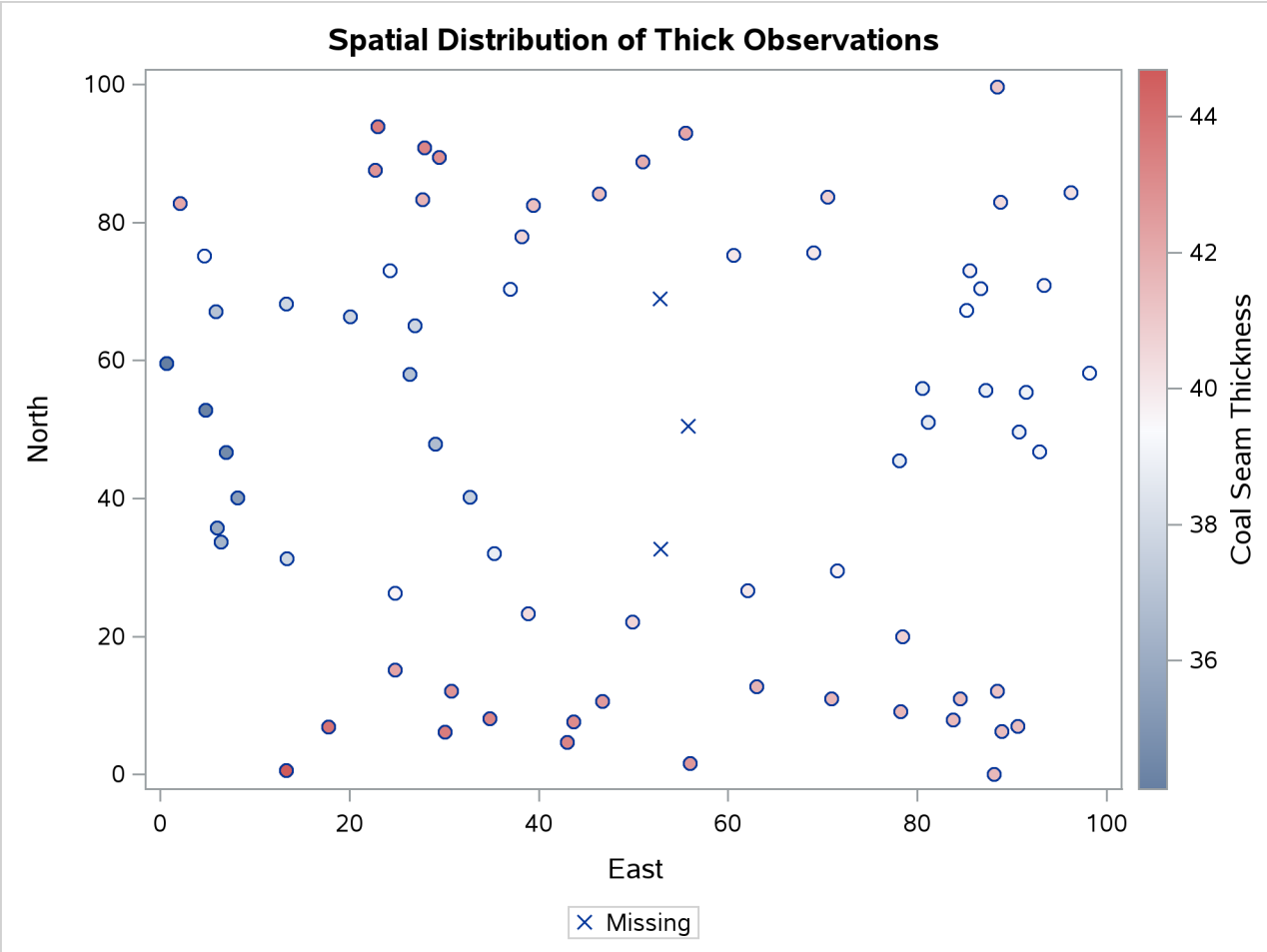

Predictions at grid points with few neighboring data points rely heavily on the underlying covariance structure. The covariance model has a range of about 30,000 feet, which suggests that within this range a grid point might have no data neighbors at all and still obtain a prediction value on the basis of the correlation structure alone. This type of behavior is demonstrated in the Output 74.3.3, which shows a circular region in the center of the plot that has no data points. Predictions at the nodes in this area are mostly influenced by the covariance structure.

You can see the impact of this effect on the predictions if you compare the prediction contours in the Output 74.3.3 to the ones in Figure 74.4. Despite the contribution of the neighboring Thick data values to the predictions within the area of no observations, the outcome is clearly altered by the absence of observations at the locations s_1, s_2, and s_3.

Output 74.3.3 Surface Plot and Contours of Kriged Coal Seam Thickness

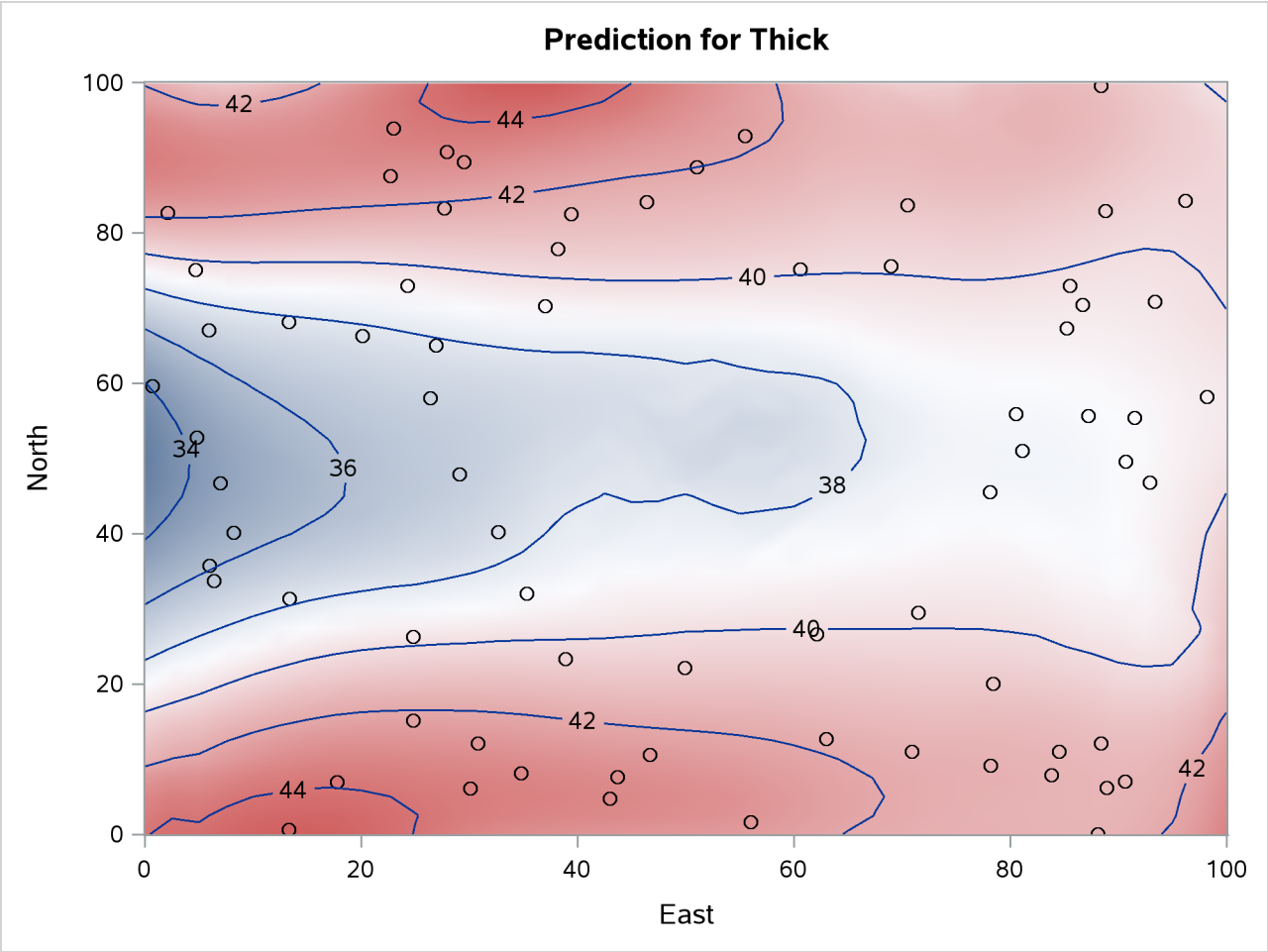

A noticeable difference is also apparent in the plot of the prediction standard errors. Output 74.3.4 displays these errors, and you can compare it to the standard error surface in Figure 74.4. The comparison shows a slight difference in the color gradient within the area of the missing data values. Output 74.3.4 uses standard error contours to enhance the effect of this difference.

The lack of information from the removed data results in an increase of the prediction uncertainty at the grid nodes that are most remotely situated from any observation in the central part of the domain. According to Output 74.3.4, the standard error at these nodes is almost comparable to the error observed near the borders of the domain, where the nodes of the prediction grid have relatively fewer data neighbors than other nodes in the domain.

Output 74.3.4 Surface Plot and Contours of Prediction Standard Errors

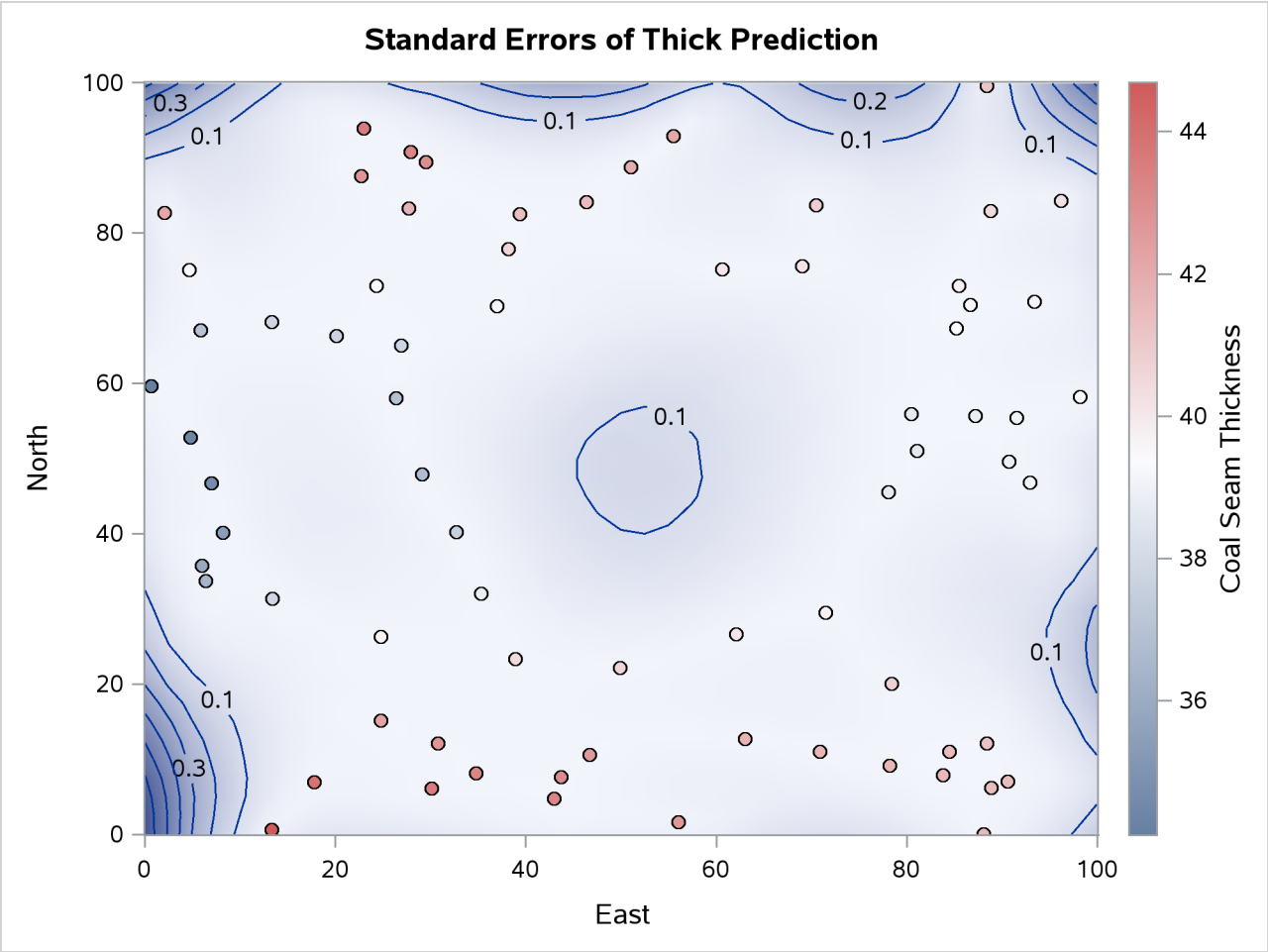

On a side note, PREDICTION plots display only observations with nonmissing values, as the plots in Output 74.3.3 and Output 74.3.4 demonstrate.

References

Chilès, J.-P., and Delfiner, P. (1999). *Geostatistics: Modeling Spatial Uncertainty*. New York: John Wiley & Sons.

Christakos, G. (1992). *Random Field Models in Earth Sciences*. New York: Academic Press.

Cressie, N. (1993). *Statistics for Spatial Data*. Rev. ed. New York: John Wiley & Sons.

Deutsch, C. V., and Journel, A. G. (1992). *GSLIB: Geostatistical Software Library and User's Guide*. New York: Oxford University Press.

Goovaerts, P. (1997). *Geostatistics for Natural Resources Evaluation*. New York: Oxford University Press.

Hohn, M. E. (1988). *Geostatistics and Petroleum Geology*. New York: Van Nostrand Reinhold.

Isaaks, E. H., and Srivastava, R. M. (1988). "Spatial Continuity Measures for Probabilistic and Deterministic Geostatistics." *Mathematical Geology* 20:313–341.

Journel, A. G. (1985). "The Deterministic Side of Geostatistics." *Mathematical Geology* 17:1–15.

Journel, A. G., and Huijbregts, C. J. (1978). *Mining Geostatistics*. New York: Academic Press.

Journel, A. G., and Rossi, M. (1989). "When Do We Need a Trend Model in Kriging?" *Mathematical Geology* 21:715–739.

Matheron, G. (1971). *The Theory of Regionalized Variables and Its Applications.* Les Cahiers du Centre de Morphologie Mathématique de Fontainebleau. Paris: École Nationale Supérieure des Mines de Paris.

Robinson, G. K. (1991). "That BLUP Is a Good Thing: The Estimation of Random Effects." *Statistical Science* 6:15–51.

Chapter 75
The LATTICE Procedure

Contents

Overview: LATTICE Procedure **5623**
Getting Started: LATTICE Procedure **5624**
Syntax: LATTICE Procedure **5626**
PROC LATTICE Statement 5626
BY Statement 5626
VAR Statement 5627
Details: LATTICE Procedure **5627**
Input Data Set 5627
Missing Values 5628
Displayed Output 5628
ODS Table Names 5629
Example: LATTICE Procedure **5629**
Example 75.1: Analysis of Variance through PROC LATTICE 5629
References **5631**

Overview: LATTICE Procedure

The LATTICE procedure computes the analysis of variance and analysis of simple covariance for data from an experiment with a lattice design. PROC LATTICE analyzes balanced square lattices, partially balanced square lattices, and some rectangular lattices.

In balanced square lattices, the number of treatments is equal to the square of the number of units per block. Incomplete blocks are grouped to form mutually orthogonal replications. The number of replicates in the basic plan is always 1 plus the number of units per block.

Partially balanced square lattices are similar to balanced lattices, although the number of replicates can vary. Partially balanced designs are constructed of the replicates in the basic plan, but not all replicates are included the same number of times, and some might not be included at all.

In rectangular lattices, there are k units per block and k (k +1) treatments. As in square lattices, blocks are grouped to form mutually orthogonal replicates in the basic plan. PROC LATTICE can analyze simple rectangular lattices (two orthogonal replications) and triple rectangular lattices (three orthogonal replications). The experiment can include several repetitions of the basic plan.

The LATTICE procedure determines from the data set which type of design has been used. It also checks to see whether the design is valid and displays an appropriate message if it is not.

Getting Started: LATTICE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/latgs1.sas.)

An example of a balanced square design is an experiment to investigate the effects of nine diets on the growth rate of pigs.

In some breeds of pigs, past experience has shown that a large part of the total variation in growth rates between animals can be attributed to the litter. Therefore, this experiment is planned so that litter differences do not contribute to the intrablock error.

First, the pigs are separated into sets of three litter-mates. Each block is assigned two sets of the three litter-mates. In a given block, one pig from each set receives a diet. Therefore, the experimental unit is a pair of pigs feeding in a particular pen on one of the nine diets. The response variable, growth rate, is the sum of the growth rates for the two pigs in a particular pen. To get the adjusted diet mean per pig, the adjusted treatment mean for the pen must be divided by 2.

The special numeric SAS variables named Group, Block, Treatment, and Rep must be used to define the design. In this example, the Treatment variable ranges from 1 to 9 and indicates the particular diet. The Block variable is 1, 2, or 3 and indicates the pen containing the two pigs. The Group variable ranges from 1 to 4 and specifies which replication within the basic plan includes the experimental unit. In this example, you would not use the Rep variable since the entire basic plan is not replicated.

You can use the following DATA step and PROC LATTICE statement to analyze this experiment. The response variable is Weight.

```
title 'Examining the Growth Rate of Pigs';

data Pigs;
   input Group Block Treatment Weight @@;
   datalines;
1 1 1 2.20  1 1 2 1.84  1 1 3 2.18  1 2 4 2.05  1 2 5 0.85
1 2 6 1.86  1 3 7 0.73  1 3 8 1.60  1 3 9 1.76
2 1 1 1.19  2 1 4 1.20  2 1 7 1.15  2 2 2 2.26  2 2 5 1.07
2 2 8 1.45  2 3 3 2.12  2 3 6 2.03  2 3 9 1.63
3 1 1 1.81  3 1 5 1.16  3 1 9 1.11  3 2 2 1.76  3 2 6 2.16
3 2 7 1.80  3 3 3 1.71  3 3 4 1.57  3 3 8 1.13
4 1 1 1.77  4 1 6 1.57  4 1 8 1.43  4 2 2 1.50  4 2 4 1.60
4 2 9 1.42  4 3 3 2.04  4 3 5 0.93  4 3 7 1.78
;

proc lattice data=Pigs;
   var Weight;
run;
```

The SAS code produces the output shown in Figure 75.1.

Figure 75.1 Output from Example LATTICE Procedure

Examining the Growth Rate of Pigs

The Lattice Procedure

Analysis of Variance for Weight

Source	DF	Sum of Squares	Mean Square
Replications	3	0.07739	0.02580
Blocks within Replications (Adj.)	8	1.4206	0.1776
Component B	8	1.4206	0.1776
Treatments (Unadj.)	8	3.2261	0.4033
Intra Block Error	16	1.2368	0.07730
Randomized Complete Block Error	24	2.6574	0.1107
Total	35	5.9609	0.1703

Additional Statistics for Weight

Variance of Means in Same Block	0.04593
LSD at .01 Level	0.6259
LSD at .05 Level	0.4543
Efficiency Relative to RCBD	120.55

Adjusted Treatment Means for Weight

Treatment	Mean
1	1.8035
2	1.7544
3	1.9643
4	1.7267
5	0.9393
6	1.8448
7	1.3870
8	1.4347
9	1.5004

Diet 3 yields the highest mean growth rate at 1.9643 pounds for the two pigs (0.9822 per pig), while diet 5 has the lowest rate at 0.9393 (0.4696 per pig). The efficiency of the experiment relative to a randomized complete block design is 120.55 percent, so using the lattice design increased precision, producing more accurate estimates of the treatment effects. The different elements of the LATTICE procedure's output are discussed in the "Displayed Output" on page 5628 section.

Syntax: LATTICE Procedure

The following statements are available in the LATTICE procedure:

PROC LATTICE < *options* > **;**
  BY *variables* **;**
  VAR *variables* **;**

Three specific numeric SAS variables, Group, Block, and Treatment, *must* be present in the data set to which PROC LATTICE is applied. For compatibility with previous releases, the variable Treatment can alternatively be named Treatmnt. A fourth numeric variable named Rep must be present when the design involves repetition of the entire basic plan. (See the "Input Data Set" on page 5627 section for more information.)

Every numeric variable other than Group, Block, Treatment, or Rep in the input SAS data set may be considered a response variable. A VAR statement tells PROC LATTICE that only the variables listed in the VAR statement are to be considered response variables. If the VAR statement is omitted, then all numeric variables, excluding Group, Block, Treatment, and Rep, are considered response variables. PROC LATTICE performs an analysis for each response variable.

PROC LATTICE Statement

PROC LATTICE < *options* > **;**

The PROC LATTICE statement invokes the LATTICE procedure.

DATA=*SAS-data-set*
  names the SAS data set to be used by PROC LATTICE. If you omit the DATA= option, the most recently created SAS data set is used.

COVARIANCE
COV
  calculates sums of products for every possible pair of response variables. A sum of products is given for each source of variation in the analysis of variance table. For each pair of response variables, the one appearing later in the data set (or in the VAR statement) is the covariable.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC LATTICE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the LATTICE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the response variables. If you do not include a VAR statement, all numeric variables in the data set are considered response variables (except Group, Block, Treatment, and Rep).

Details: LATTICE Procedure

Input Data Set

Four numeric SAS variables, Group, Block, Treatment, and Rep, are used in the input data set to define the lattice design. The Group, Block, and Treatment variables are required in the data set to which PROC LATTICE is applied. For compatibility with previous releases, the third variable can alternatively be named Treatmnt. The Rep variable must be present when the design involves repetition of the entire basic plan.

Group — specifies which orthogonal replication in the basic plan includes the experimental unit. Values of Group must be $1, 2, \ldots, n$, where n is the number of replicates in the basic plan.

Block — specifies the block in which the experimental unit is present. Values of Block must be $1, 2, \ldots, m$, where m is the number of blocks in a replication.

Treatment — specifies which treatment was applied to the experimental unit. Values of Treatment must be $1, 2, \ldots, i$, where i is the number of treatments in a replication.

Rep — specifies which repetition of the basic plan includes the experimental unit. Values of Rep must be $1, 2, \ldots, p$, where p is the number of replications of the entire basic plan. Thus, the experiment has a total of np replicates.

Missing Values

If a value of Group, Block, Treatment, or Rep is missing, the analysis is not performed and an appropriate error message is displayed.

If a value of a response variable is missing, this entire variable is dropped from the analysis. If other response variables exist that do not have missing values, they are analyzed.

Displayed Output

For each response variable, PROC LATTICE displays the following

- an "Analysis of Variance" table and related statistics, including the following as separate sources of variations:
  - Replications
  - Blocks within Replications (adjusted for treatments)
  - Treatments (unadjusted)
  - Intra-block Error
  - Randomized Complete Block Error

  The Blocks within Replications sum of squares is further broken down into "Component A" and "Component B." If there is no repetition of the basic plan, the Component B sum of squares is the same as the Blocks within Replications sum of squares. If there is repetition of the basic plan, the Component A sum of squares reflects the variation among blocks that contain the same treatments.

  The source of variation called Randomized Complete Block Error is the sum of the Blocks within Replications sum of squares and the Intra-block Error sum of squares. It is the appropriate error term if the experimental design is a randomized complete block design, with the replications filling the roles of complete blocks.

- two values for the Variance of Means. For some lattice designs, these are only approximations. The first value is applicable when the two treatments appear in the same block; the other (when it appears) applies when the two treatments never appear in the same block (a possibility in partially balanced and rectangular designs).

- an Average of Variance. Except with small designs, it is sufficient to use this average variance of means for tests between treatments (whether the two treatments appear in the same block or not); see Cochran and Cox (1957).

- the Least Significant Differences (LSDs) at the 0.01 and 0.05 levels of significance, based on the Average of Variance

- Efficiency Relative to RCBD, the efficiency of the lattice design relative to a randomized complete block design. The efficiency is the ratio of the randomized complete block mean squared error to the effective error variance; see Cochran and Cox (1957).

- the Adjusted Treatment Means. These are adjusted for blocks if the relative precision is greater than 105%, as suggested in Hinkelmann and Kempthorne (2005).

When you specify the COVARIANCE option, PROC LATTICE produces sums of products and the mean product for each source of variation in the analysis of variance table.

ODS Table Names

PROC LATTICE assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 75.1. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 75.1 ODS Tables Produced by PROC LATTICE

ODS Table Name	Description	PROC LATTICE Option
ANOVA	Analysis of variance	Default
AdjTreatmentMeans	Adjusted treatment means	Default
Statistics	Additional statistics	Default

Example: LATTICE Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 75.1: Analysis of Variance through PROC LATTICE

(View the complete code for this example (latex1.sas) in the example repository.)

In the following example, from Cochran and Cox (1957, p. 406), the data are yields (Yield) in bushels per acre of 25 varieties (Treatment) of soybeans. The data are collected in two replications (Group) of 25 varieties in five blocks (Block) containing five varieties each. This is an example of a partially balanced square lattice design.

```
data Soy(drop=plot);
   do Group = 1 to 2;
      do Block = 1 to 5;
         do Plot = 1 to 5;
            input Treatment Yield @@;
            output;
         end;
      end;
```

```
   end;
   datalines;
 1  6  2  7  3  5  4  8  5  6  6 16  7 12  8 12  9 13 10  8
11 17 12  7 13  7 14  9 15 14 16 18 17 16 18 13 19 13 20 14
21 14 22 15 23 11 24 14 25 14  1 24  6 13 11 24 16 11 21  8
 2 21  7 11 12 14 17 11 22 23  3 16  8  4 13 12 18 12 23 12
 4 17  9 10 14 30 19  9 24 23  5 15 10 15 15 22 20 16 25 19
;

proc lattice data=Soy;
run;
```

The results from these statements are shown in Output 75.1.1.

Output 75.1.1 Displayed Output from PROC LATTICE

The Lattice Procedure

Analysis of Variance for Yield

Source	DF	Sum of Squares	Mean Square
Replications	1	212.18	212.18
Blocks within Replications (Adj.)	8	501.84	62.7300
Component B	8	501.84	62.7300
Treatments (Unadj.)	24	559.28	23.3033
Intra Block Error	16	218.48	13.6550
Randomized Complete Block Error	24	720.32	30.0133
Total	49	1491.78	30.4445

Additional Statistics for Yield

Variance of Means in Same Block	15.7915
Variance of Means in Different Blocks	17.9280
Average of Variance	17.2159
LSD at .01 Level	12.1189
LSD at .05 Level	8.7959
Efficiency Relative to RCBD	174.34

Output 75.1.1 *continued*

Adjusted Treatment Means for Yield	
Treatment	**Mean**
1	19.0681
2	16.9728
3	14.6463
4	14.7687
5	12.8470
6	13.1701
7	9.0748
8	6.7483
9	8.3707
10	8.4489
11	23.5511
12	12.4558
13	12.6293
14	20.7517
15	19.3299
16	12.6224
17	10.5272
18	10.7007
19	7.3231
20	11.4013
21	11.6259
22	18.5306
23	12.2041
24	17.3265
25	15.4048

The efficiency of the experiment relative to a randomized complete block design is 174.34%. Precision is gained using the lattice design via the recovery of intra-block error information, enabling more accurate estimates of the treatment effects. Variety 8 of soybean had the lowest adjusted treatment mean (6.7483 bushels per acre), while variety 11 of soybean had the highest adjusted treatment mean (23.5511 bushels per acre).

References

Cochran, W. G., and Cox, G. M. (1957). *Experimental Designs*. 2nd ed. New York: John Wiley & Sons.

Comstock, R. E., Peterson, W. J., and Stewart, H. A. (1948). "An Application of the Balanced Lattice Design in a Feeding Trial with Swine." *Journal of Animal Science* 7:320–331.

Cornelius, P. L. (1983). "Lattice Designs." In *Encyclopedia of Statistical Sciences*, vol. 4, edited by S. Kotz, N. L. Johnson, and C. B. Read. New York: John Wiley & Sons.

Hinkelmann, K., and Kempthorne, O. (2005). *Advanced Experimental Design.* Vol. 2 of Design and Analysis of Experiments. New York: John Wiley & Sons.

Robinson, H. F., and Watson, G. S. (1949). *Analysis of Simple and Triple Rectangular Designs*. Technical Report 88, North Carolina Agricultural Experiment Station.

Chapter 76
The LIFEREG Procedure

Contents

Overview: LIFEREG Procedure **5634**
Getting Started: LIFEREG Procedure **5637**
Modeling Right-Censored Failure Time Data 5637
Bayesian Analysis of Right-Censored Data 5641
Syntax: LIFEREG Procedure **5647**
PROC LIFEREG Statement 5648
BAYES Statement 5650
BY Statement 5660
CLASS Statement 5660
ESTIMATE Statement 5661
INSET Statement 5662
LSMEANS Statement 5663
LSMESTIMATE Statement 5664
MODEL Statement 5665
OUTPUT Statement 5671
PROBPLOT Statement 5673
SLICE Statement 5677
STORE Statement 5677
TEST Statement 5677
WEIGHT Statement 5678
Details: LIFEREG Procedure **5678**
Missing Values 5678
Model Specification 5678
Computational Method 5679
Supported Distributions 5681
Predicted Values 5684
Confidence Intervals 5686
Fit Statistics 5687
Probability Plotting 5688
INEST= Data Set 5693
OUTEST= Data Set 5693
XDATA= Data Set 5694
Computational Resources 5695
Bayesian Analysis 5695
Displayed Output for Classical Analysis 5699
Displayed Output for Bayesian Analysis 5700

Plot Options Superseded by ODS Graphics 5702
ODS Table Names 5711
ODS Graphics 5713
Examples: LIFEREG Procedure **5714**
Example 76.1: Motorette Failure 5714
Example 76.2: Computing Predicted Values for a Tobit Model 5719
Example 76.3: Overcoming Convergence Problems by Specifying Initial Values . . . 5723
Example 76.4: Analysis of Arbitrarily Censored Data with Interaction Effects 5728
Example 76.5: Probability Plotting—Right Censoring 5732
Example 76.6: Probability Plotting—Arbitrary Censoring 5735
Example 76.7: Bayesian Analysis of Clinical Trial Data 5738
Example 76.8: Model Postfitting Analysis 5744
References **5748**

Overview: LIFEREG Procedure

The LIFEREG procedure fits parametric models to failure time data that can be uncensored, right censored, left censored, or interval censored. The models for the response variable consist of a linear effect composed of the covariates and a random disturbance term. The distribution of the random disturbance can be taken from a class of distributions that includes the extreme value, normal, logistic, and, by using a log transformation, the exponential, Weibull, lognormal, log-logistic, and three-parameter gamma distributions.

The model assumed for the response $\mathbf{y}$ is

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \sigma\boldsymbol{\epsilon}$$

where $\mathbf{y}$ is a vector of response values, often the log of the failure times, $\mathbf{X}$ is a matrix of covariates or independent variables (usually including an intercept term), $\boldsymbol{\beta}$ is a vector of unknown regression parameters, σ is an unknown scale parameter, and $\boldsymbol{\epsilon}$ is a vector of errors assumed to come from a known distribution (such as the standard normal distribution). If an offset variable O is specified, the form of the model is $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{O} + \sigma\boldsymbol{\epsilon}$, where $\mathbf{O}$ is a vector of values of the offset variable O. The distribution might also depend on additional shape parameters. These models are equivalent to accelerated failure time models when the log of the response is the quantity being modeled. The effect of the covariates in an accelerated failure time model is to change the scale, and not the location, of a baseline distribution of failure times.

The LIFEREG procedure estimates the parameters by maximum likelihood with a Newton-Raphson algorithm. PROC LIFEREG estimates the standard errors of the parameter estimates from the inverse of the observed information matrix.

The accelerated failure time model assumes that the effect of independent variables on an event time distribution is multiplicative on the event time. Usually, the scale function is $\exp(\mathbf{x}_c'\boldsymbol{\beta}_c)$, where $\mathbf{x}_c$ is the vector of covariate values (not including the intercept term) and $\boldsymbol{\beta}_c$ is a vector of unknown parameters. Thus, if T_0 is an event time sampled from the baseline distribution corresponding to values of zero for the covariates, then the accelerated failure time model specifies that, if the vector of covariates is $\mathbf{x}_c$, the event time is $T = \exp(\mathbf{x}_c'\boldsymbol{\beta}_c)T_0$. If $y = \log(T)$ and $y_0 = \log(T_0)$, then

$$y = \mathbf{x}_c'\boldsymbol{\beta}_c + y_0$$

This is a linear model with y_0 as the error term.

In terms of survival or exceedance probabilities, this model is

$$\Pr(T > t \mid \mathbf{x}_c) = \Pr(T_0 > \exp(-\mathbf{x}_c'\boldsymbol{\beta}_c)t)$$

The probability on the left-hand side of the equal sign is evaluated given the value $\mathbf{x}_c$ for the covariates, and the right-hand side is computed using the baseline probability distribution but at a scaled value of the argument. The right-hand side of the equation represents the value of the baseline survival function evaluated at $\exp(-\mathbf{x}_c'\boldsymbol{\beta}_c)t$.

Models usually have an intercept parameter and a scale parameter. In terms of the original untransformed event times, the effects of the intercept term and the scale term are to scale the event time and to raise the event time to a power, respectively. That is, if

$$\log(T_0) = \mu + \sigma \log(T_\epsilon)$$

then

$$T_0 = \exp(\mu) T_\epsilon^\sigma$$

Although it is possible to fit these models to the original response variable by using the NOLOG option, it is more common to model the log of the response variable. Because of this log transformation, zero values for the observed failure times are not allowed unless the NOLOG option is specified. Similarly, small values for the observed failure times lead to large negative values for the transformed response. The NOLOG option should be used only if you want to fit a distribution appropriate for the untransformed response, such as the extreme value instead of the Weibull. If you specify the normal or logistic distributions, the responses are not log transformed; that is, the NOLOG option is implicitly assumed.

Parameter estimates for the normal distribution are sensitive to large negative values, and care must be taken that the fitted model is not unduly influenced by them. Large negative values for the normal distribution can occur when fitting the lognormal distribution by log transforming the response, and some response values are near zero. Likewise, values that are extremely large after the log transformation have a strong influence in fitting the Weibull distribution (that is, the extreme value distribution for log responses). You should examine the residuals and check the effects of removing observations with large residuals or extreme values of covariates on the model parameters. The logistic distribution gives robust parameter estimates in the sense that the estimates have a bounded influence function.

The standard errors of the parameter estimates are computed from large sample normal approximations by using the observed information matrix. In small samples, these approximations might be poor. See Lawless (2003) for additional discussion and references. You can sometimes construct better confidence intervals by transforming the parameters. For example, large sample theory is often more accurate for $\log(\sigma)$ than σ. Therefore, it might be more accurate to construct confidence intervals for $\log(\sigma)$ and transform these into confidence intervals for σ. The parameter estimates and their estimated covariance matrix are available in an output SAS data set and can be used to construct additional tests or confidence intervals for the parameters. Alternatively, tests of parameters can be based on log-likelihood ratios. See Cox and Oakes (1984) for a discussion of the merits of some possible test methods including score, Wald, and likelihood ratio tests. Likelihood ratio tests are generally more reliable for small samples than tests based on the information matrix.

The log-likelihood function is computed using the log of the failure time as a response. This log likelihood differs from the log likelihood obtained using the failure time as the response by an additive term of $\sum \log(t_i)$,

where the sum is over the uncensored failure times. This term does not depend on the unknown parameters and does not affect parameter or standard error estimates. However, many published values of log likelihoods use the failure time as the basic response variable and, hence, differ by the additive term from the value computed by the LIFEREG procedure.

The classic Tobit model also fits into this class of models but with data usually censored on the left. The data considered by Tobin (1958) in his original paper came from a survey of consumers where the response variable is the ratio of expenditures on durable goods to the total disposable income. The two explanatory variables are the age of the head of household and the ratio of liquid assets to total disposable income. Because many observations in this data set have a value of zero for the response variable, the model fit by Tobin is

$$\mathbf{y} = \max(\mathbf{x}'\boldsymbol{\beta} + \epsilon, 0)$$

which is a regression model with left censoring, where $\mathbf{x}' = (1, \mathbf{x}_c')$.

Bayesian analysis of parametric survival models can be requested by using the BAYES statement in the LIFEREG procedure. In Bayesian analysis, the model parameters are treated as random variables, and inference about parameters is based on the posterior distribution of the parameters, given the data. The posterior distribution is obtained using Bayes' theorem as the likelihood function of the data weighted with a prior distribution. The prior distribution enables you to incorporate knowledge or experience of the likely range of values of the parameters of interest into the analysis. If you have no prior knowledge of the parameter values, you can use a noninformative prior distribution, and the results of the Bayesian analysis will be very similar to a classical analysis based on maximum likelihood. A closed form of the posterior distribution is often not feasible, and a Markov chain Monte Carlo method by Gibbs sampling is used to simulate samples from the posterior distribution. See Chapter 8, "Introduction to Bayesian Analysis Procedures," for an introduction to the basic concepts of Bayesian statistics. Also see the section "Bayesian Analysis: Advantages and Disadvantages" on page 154 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a discussion of the advantages and disadvantages of Bayesian analysis. See Ibrahim, Chen, and Sinha (2001) and Gilks, Richardson, and Spiegelhalter (1996) for more information about Bayesian analysis, including guidance in choosing prior distributions.

For Bayesian analysis, PROC LIFEREG generates a Gibbs chain for the posterior distribution of the model parameters. Summary statistics (mean, standard deviation, quartiles, HPD and credible intervals, correlation matrix) and convergence diagnostics (autocorrelations; Gelman-Rubin, Geweke, Raftery-Lewis, and Heidelberger and Welch tests; and the effective sample size) are computed for each parameter, as well as the correlation matrix of the posterior sample. Trace plots, posterior density plots, and autocorrelation function plots that are created using ODS Graphics are also provided for each parameter.

The LIFEREG procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Getting Started: LIFEREG Procedure

The following examples demonstrate how you can use the LIFEREG procedure to fit a parametric model to failure time data.

Suppose you have a response variable y that represents failure time; a binary variable, censor, with censor=0 indicating censored values; and two linearly independent variables, x1 and x2. The following statements perform a typical accelerated failure time model analysis. Higher-order effects such as interactions and nested effects are allowed in the independent variables list, but they are not shown in this example.

```
proc lifereg;
   model y*censor(0) = x1 x2;
run;
```

PROC LIFEREG can fit models to interval-censored data. The syntax for specifying interval-censored data is as follows:

```
proc lifereg;
   model (begin, end) = x1 x2;
run;
```

You can also model binomial data by using the *events/trials* syntax for the response, as illustrated in the following statements:

```
proc lifereg;
   model r/n=x1 x2;
run;
```

The variable n represents the number of trials, and the variable r represents the number of events.

Modeling Right-Censored Failure Time Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/lifergs1.sas.)

The following example demonstrates how you can use the LIFEREG procedure to fit a model to right-censored failure time data.

Suppose you conduct a study of two headache pain relievers. You divide patients into two groups, with each group receiving a different type of pain reliever. You record the time taken (in minutes) for each patient to report headache relief. Because some of the patients never report relief for the entire study, some of the observations are censored.

The following DATA step creates the SAS data set headache:

```
data Headache;
   input Minutes Group Censor @@;
   datalines;
11  1  0   12  1  0   19  1  0   19  1  0
19  1  0   19  1  0   21  1  0   20  1  0
21  1  0   21  1  0   20  1  0   21  1  0
```

```
20  1  0   21  1  0   25  1  0   27  1  0
30  1  0   21  1  1   24  1  1   14  2  0
16  2  0   16  2  0   21  2  0   21  2  0
23  2  0   23  2  0   23  2  0   23  2  0
25  2  1   23  2  0   24  2  0   24  2  0
26  2  1   32  2  1   30  2  1   30  2  0
32  2  1   20  2  1
;
```

The data set Headache contains the variable Minutes, which represents the reported time to headache relief; the variable Group, the group to which the patient is assigned; and the variable Censor, a binary variable indicating whether the observation is censored. Valid values of the variable Censor are 0 (no) and 1 (yes). Figure 76.1 shows the first five records of the data set Headache.

Figure 76.1 Headache Data

Obs	Minutes	Group	Censor
1	11	1	0
2	12	1	0
3	19	1	0
4	19	1	0
5	19	1	0

The following statements invoke the LIFEREG procedure:

```
proc lifereg data=Headache;
   class Group;
   model Minutes*Censor(1)=Group;
   output out=New cdf=Prob;
run;
```

The CLASS statement specifies the variable Group as the classification variable. The MODEL statement syntax indicates that the response variable Minutes is right censored when the variable Censor takes the value 1. The MODEL statement specifies the variable Group as the single explanatory variable. Because the MODEL statement does not specify the DISTRIBUTION= option, the LIFEREG procedure fits the default type 1 extreme-value distribution by using $\log$(Minutes) as the response. This is equivalent to fitting the Weibull distribution.

The OUTPUT statement creates the output data set New. In addition to containing the variables in the original data set Headache, the SAS data set New also contains the variable Prob. This new variable is created by the CDF= option to contain the estimates of the cumulative distribution function evaluated at the observed response.

The results of this analysis are displayed in the following figures.

Figure 76.2 Model Fitting Information from the LIFEREG Procedure

The LIFEREG Procedure

Model Information	
Data Set	WORK.HEADACHE
Dependent Variable	Log(Minutes)
Censoring Variable	Censor
Censoring Value(s)	1
Number of Observations	38
Noncensored Values	30
Right Censored Values	8
Left Censored Values	0
Interval Censored Values	0
Number of Parameters	3
Name of Distribution	Weibull
Log Likelihood	-9.37930239

Class Level Information		
Name	**Levels**	**Values**
Group	2	1 2

Figure 76.2 displays the class level information and model fitting information. There are 30 uncensored observations and 8 right-censored observations. The log likelihood for the Weibull distribution is –9.3793. The log-likelihood value can be used to compare the goodness of fit for nested models with different covariates, but with the same distribution.

Figure 76.3 Model Fit Statistics from the LIFEREG Procedure

Fit Statistics	
-2 Log Likelihood	18.759
AIC (smaller is better)	24.759
AICC (smaller is better)	25.464
BIC (smaller is better)	29.671

Fit Statistics (Unlogged Response)	
-2 Log Likelihood	199.747
Weibull AIC (smaller is better)	205.747
Weibull AICC (smaller is better)	206.453
Weibull BIC (smaller is better)	210.660

Figure 76.3 displays fit statistics for the model. The "Fit Statistics" table displays statistics based on the maximum extreme-value log likelihood fit by using log(Minutes) as the response. These statistics are useful in comparing the fit of a different model when the fit criteria from the model that you compare is also based on the log likelihood using log(Minutes) as the response. The "Fit Statistics (Unlogged Response)" table is based on the maximum Weibull log likelihood using Minutes as the response. The AIC, BIC, and AICC statistics in this table can be used to compare models with different covariates, in addition to models with different distributions, as long as the fit statistics for the models that you compare use Minutes as the response.

Figure 76.4 Model Parameter Estimates from the LIFEREG Procedure

Analysis of Maximum Likelihood Parameter Estimates								
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	3.3091	0.0589	3.1938	3.4245	3161.70	<.0001
Group	1	1	-0.1933	0.0786	-0.3473	-0.0393	6.05	0.0139
Group	2	0	0.0000	.	.	.	.	.
Scale		1	0.2122	0.0304	0.1603	0.2809		
Weibull Shape		1	4.7128	0.6742	3.5604	6.2381		

The table of parameter estimates is displayed in Figure 76.4. Both the intercept and the slope parameter for the variable group are significantly different from 0 at the 0.05 level. Because the variable group has only one degree of freedom, parameter estimates are given for only one level of the variable group (group=1). However, the estimate for the intercept parameter provides a baseline for group=2.

The resulting model is as follows:

$$\log(\text{minutes}) = \begin{cases} 3.30911843 - 0.1933025 & \text{for} \quad \text{group} = 1 \\ 3.30911843 & \text{for} \quad \text{group} = 2 \end{cases}$$

Note that the Weibull shape parameter for this model is the reciprocal of the extreme-value scale parameter estimate shown in Figure 76.4 ($1/0.21219 = 4.7128$).

The following statements produce a graph of the cumulative distribution values versus the variable Minutes.

```
proc sgplot data=New;
   scatter x=Minutes y=Prob / group=Group;
   discretelegend;
run;
```

Figure 76.5 displays the estimated cumulative distribution function values contained in the output data set New for each group.

Figure 76.5 Plot of the Estimated Cumulative Distribution Function

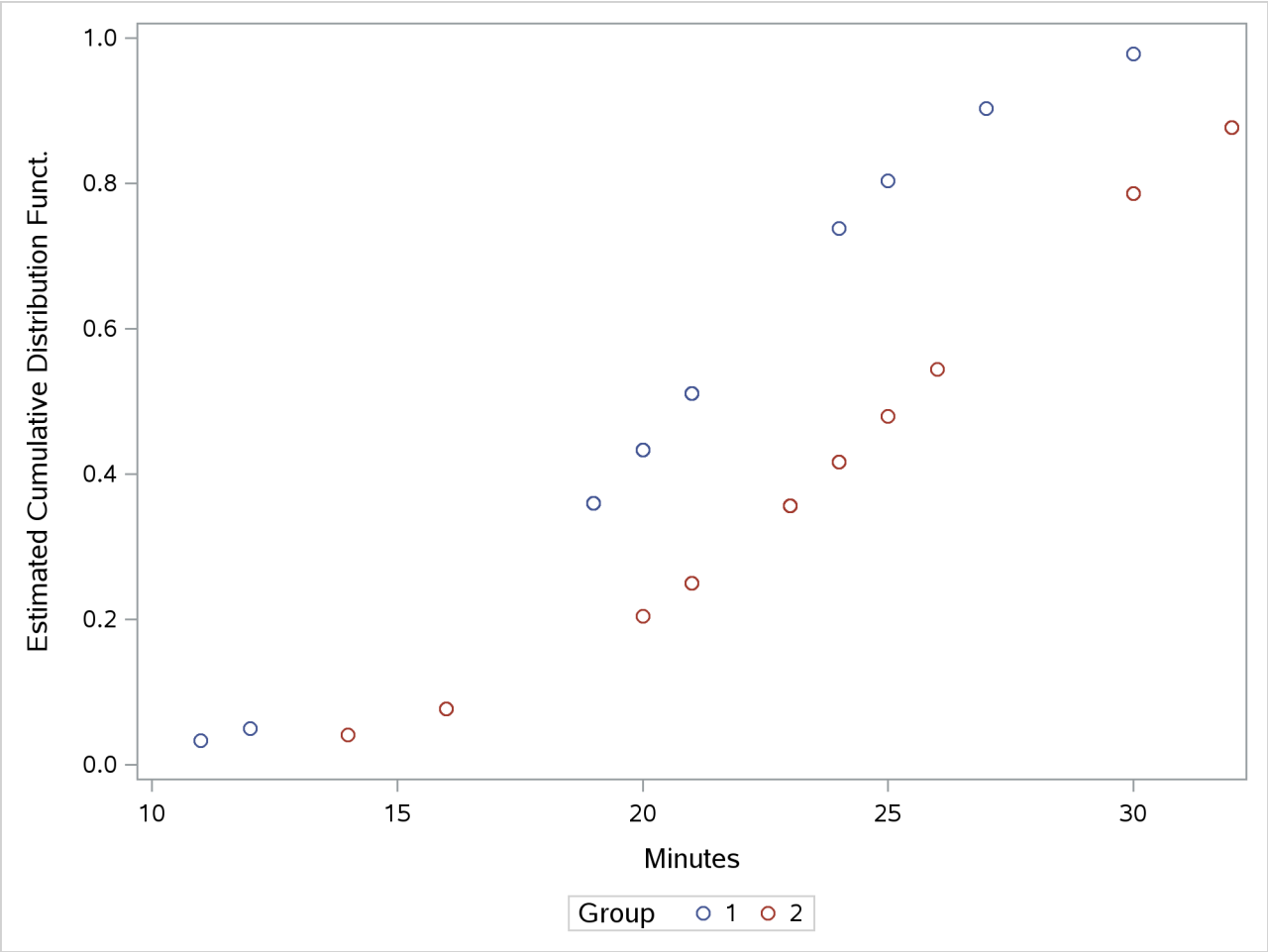

Bayesian Analysis of Right-Censored Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/lifergs2.sas.)

Nelson (1982) describes a study of the lifetimes of locomotive engine fans. This example shows how to use PROC LIFEREG to carry out a Bayesian analysis of the engine fan data. In this example, a lognormal distribution is used to model the engine lifetimes, but other survival time distributions, such as the Weibull, can also be used.

The following SAS statements create the SAS data set Fan. This data set contains a censoring indicator variable and right-censored survival times for the 70 locomotive engine fans in the study.

```
data Fan;
   input Lifetime Censor@@;
   datalines;
 450 0    460 1   1150 0   1150 0   1560 1
1600 0   1660 1   1850 1   1850 1   1850 1
1850 1   1850 1   2030 1   2030 1   2030 1
```

```
2070 0   2070 0   2080 0   2200 1   3000 1
3000 1   3000 1   3000 1   3100 0   3200 1
3450 0   3750 1   3750 1   4150 1   4150 1
4150 1   4150 1   4300 1   4300 1   4300 1
4300 1   4600 0   4850 1   4850 1   4850 1
4850 1   5000 1   5000 1   5000 1   6100 1
6100 0   6100 1   6100 1   6300 1   6450 1
6450 1   6700 1   7450 1   7800 1   7800 1
8100 1   8100 1   8200 1   8500 1   8500 1
8500 1   8750 1   8750 0   8750 1   9400 1
9900 1  10100 1  10100 1  10100 1  11500 1
;
```

Some of the fans had not failed at the time the data were collected, and the unfailed units have right-censored lifetimes. The variable Lifetime represents either a failure time or a censoring time. The variable Censor is equal to 0 if the value of Lifetime is a failure time, and it is equal to 1 if the value is a censoring time.

The following SAS statements specify a Bayesian analysis that uses a lognormal model for the engine lifetimes. There are no covariates, so the model is an intercept-only model. The OUTPOST= option saves the samples from the posterior distribution in the SAS data set Post for further processing.

```
ods graphics on;
proc lifereg data=Fan;
   model Lifetime*Censor( 1 )= / dist=lognormal;
   bayes seed=1  outpost=Post;
run;
```

The SEED= option is specified to maintain reproducibility; no other options are specified in the BAYES statement. By default, a uniform prior distribution is assumed for the intercept coefficient. The uniform prior is a flat prior on the real line with a distribution that reflects ignorance of the location of the parameter, placing equal probability on all possible values the regression coefficient can take. Using the uniform prior in the following example, you would expect the Bayesian estimates to resemble the classical results of maximizing the likelihood. If you can elicit an informative prior on the regression coefficients, you should use the COEFFPRIOR= option to specify it. A default noninformative gamma prior is used for the lognormal scale parameter σ.

You should make sure that the posterior distribution samples have achieved convergence before using them for Bayesian inference. If you do not specify additional options, PROC LIFEREG produces by default three convergence diagnostics: autocorrelations of the posterior sample, effective sample size, and the Geweke statistic. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for information about assessing the convergence of the chain of posterior samples. Trace plots, posterior density plots, and autocorrelation function plots that are created using ODS Graphics are also provided for each parameter. See the section "Visual Analysis via Trace Plots" on page 163 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for help in interpreting these plots.

The "Analysis of Maximum Likelihood Parameter Estimates" table in Figure 76.6 summarizes maximum likelihood estimates of the lognormal intercept and scale parameters.

Figure 76.6 Maximum Likelihood Estimates from the LIFEREG Procedure

The LIFEREG Procedure

Bayesian Analysis

Analysis of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits	
Intercept	1	10.1432	0.5211	9.1219	11.1646
Scale	1	1.6796	0.3893	1.0664	2.6453

Since no prior distribution for the intercept was specified, the default uniform improper distribution shown in the "Uniform Prior for Regression Coefficients" table in Figure 76.7 is used.

Noninformative prior distributions are appropriate if you have no prior knowledge of the likely range of values of the parameters, and if you want to make probability statements about the parameters or functions of the parameters. Refer, for example, to Ibrahim, Chen, and Sinha (2001) for more information about choosing prior distributions.

The default noninformative gamma prior distribution for the lognormal scale parameter is shown in the "Independent Prior Distributions for Model Parameters" table in Figure 76.7.

Figure 76.7 Noninformative Prior Distributions

The LIFEREG Procedure

Bayesian Analysis

Uniform Prior for Regression Coefficients

Parameter	Prior
Intercept	Constant

Independent Prior Distributions for Model Parameters

Parameter	Prior Distribution	Hyperparameters			
Scale	Gamma	Shape	0.001	Inverse Scale	0.001

By default, posterior mode estimates of the model parameters are used as the starting value for the simulation. These are listed in the "Initial Values of the Chain" table in Figure 76.8.

Figure 76.8 Markov Chain Initial Values

Initial Values of the Chain

Chain	Seed	Intercept	Scale
1	1	10.0501	1.59544

Summary statistics for the posterior sample are displayed in the "Fit Statistics," "Descriptive Statistics for the Posterior Sample," "Interval Statistics for the Posterior Sample," and "Posterior Correlation Matrix" tables

in Figure 76.9. Since noninformative prior distributions were used, these results are consistent with the maximum likelihood estimates shown in Figure 76.6.

Figure 76.9 Posterior Sample Summary Statistics

Fit Statistics	
DIC (smaller is better)	87.244
pD (effective number of parameters)	1.822

The LIFEREG Procedure

Bayesian Analysis

Posterior Summaries						
				Percentiles		
Parameter	N	Mean	Standard Deviation	25%	50%	75%
Intercept	10000	10.4198	0.6171	9.9671	10.3261	10.7959
Scale	10000	1.9197	0.4808	1.5676	1.8476	2.1931

Posterior Intervals					
Parameter	Alpha	Equal-Tail Interval		HPD Interval	
Intercept	0.050	9.4478	11.8994	9.3224	11.6752
Scale	0.050	1.1908	3.0570	1.1104	2.8833

Posterior Correlation Matrix		
Parameter	Intercept	Scale
Intercept	1.0000	0.8296
Scale	0.8296	1.0000

By default, PROC LIFEREG computes three convergence diagnostics: the lag1, lag5, lag10, and lag50 autocorrelations; the Geweke diagnostic; and the effective sample size. These are displayed in Figure 76.10. There is no indication that the Markov chain has not converged. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more information about convergence diagnostics and their interpretation.

Figure 76.10 Posterior Sample Summary Statistics

The LIFEREG Procedure

Bayesian Analysis

Posterior Autocorrelations				
Parameter	Lag 1	Lag 5	Lag 10	Lag 50
Intercept	0.6973	0.1764	0.0188	-0.0016
Scale	0.6955	0.1711	0.0171	-0.0003

Figure 76.10 *continued*

Geweke Diagnostics		
Parameter	**z**	**Pr > |z|**
Intercept	-0.9718	0.3311
Scale	-0.9365	0.3490

Effective Sample Sizes			
		Autocorrelation	
Parameter	**ESS**	**Time**	**Efficiency**
Intercept	1773.7	5.6380	0.1774
Scale	1805.7	5.5381	0.1806

Summary statistics of the posterior distribution samples are produced by default. However, these statistics might not be sufficient for carrying out your Bayesian inference. The samples from the posterior distribution saved in the SAS data set Post created with the OUTPOST= option can be used for further analysis.

Trace, autocorrelation, and density plots for the three model parameters shown in Figure 76.11 and Figure 76.12 are useful in diagnosing whether the Markov chain of posterior samples has converged. These plots show no evidence that the chain has not converged. See the section "Visual Analysis via Trace Plots" on page 163 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more information about interpreting these types of diagnostic plots.

Figure 76.11 Diagnostic Plots

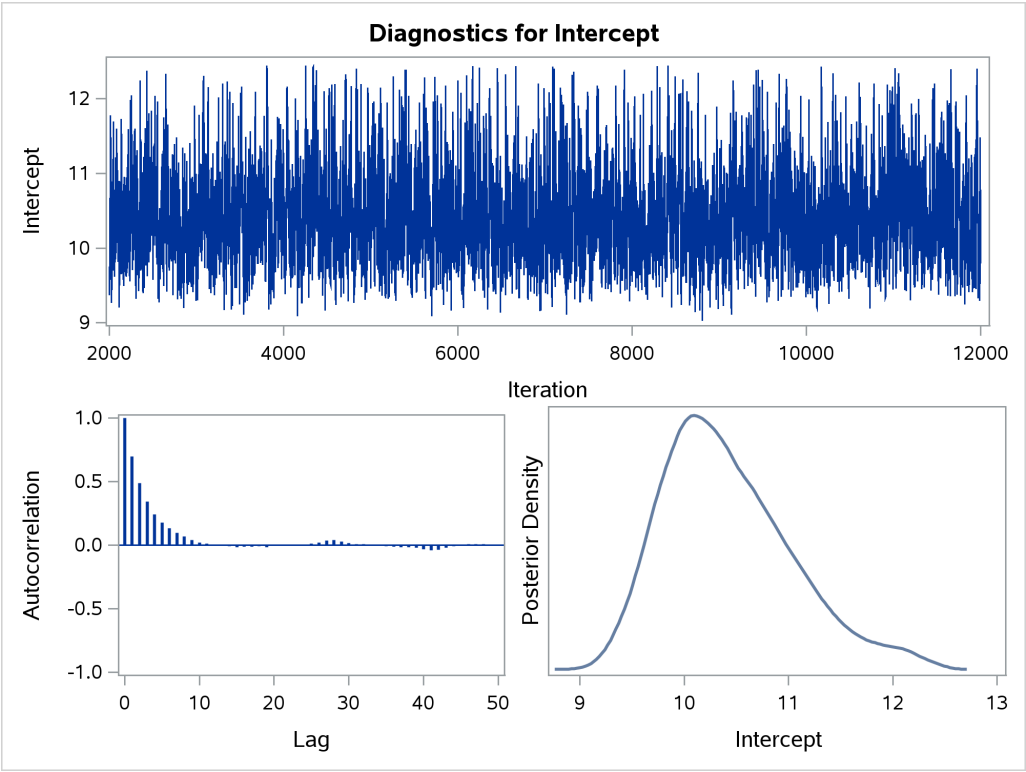

Figure 76.12 Diagnostic Plots

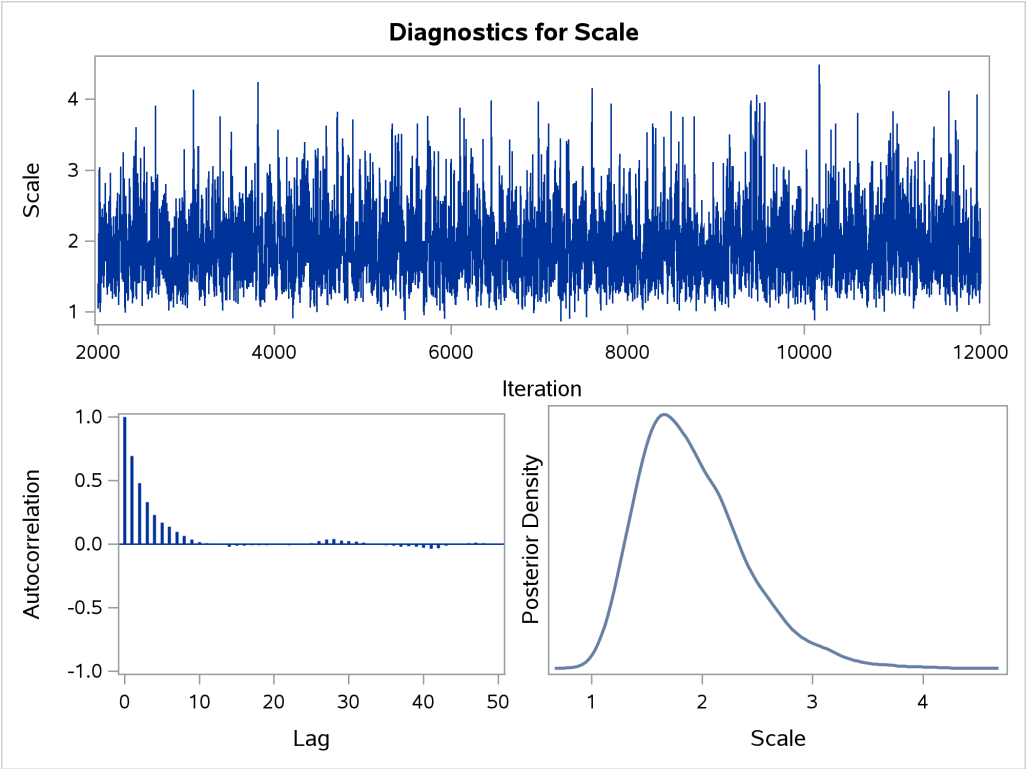

The fraction failing in the first 8000 hours of operation might be a quantity of interest. This kind of information could be useful, for example, in determining whether to improve the reliability of the engine components due to warranty considerations. The following SAS statements compute the mean and percentiles of the distribution of the fraction failing in the first 8000 hours from the posterior sample data set Post:

```
data Prob;
   set Post;
   Frac = ProbNorm(( log(8000) - Intercept ) / Scale );
   label Frac= 'Fraction Failing in 8000 Hours';
run;

proc means data = Prob(keep=Frac) n mean p10 p25 p50 p75 p90;
run;
```

The mean fraction of failures in the first 8000 hours, shown in Figure 76.13, is about 0.24, which could be used in further analysis of warranty costs. The 10th percentile is about 0.16 and the 90th percentile is about 0.32, which gives an assessment of the probable range of the fraction failing in the first 8000 hours.

Figure 76.13 Fraction Failing in 8000 Hours

The MEANS Procedure

Analysis Variable : Frac Fraction Failing in 8000 Hours						
N	Mean	10th Pctl	25th Pctl	50th Pctl	75th Pctl	90th Pctl
10000	0.2381259	0.1628591	0.1953658	0.2336532	0.2765736	0.3190793

Syntax: LIFEREG Procedure

The following statements are available in the LIFEREG procedure:

```
PROC LIFEREG <options> ;
    BAYES <options> ;
    BY variables ;
    CLASS variables ;
    ESTIMATE <'label'> estimate-specification <(divisor=n)>
                <, ... <'label'> estimate-specification <(divisor=n)>> </ options> ;
    INSET <keyword-list> </ options> ;
    LSMEANS <model-effects> </ options> ;
    LSMESTIMATE model-effect <'label'> values <(divisor=n)>
                    <, ... <'label'> values <(divisor=n)>> </ options> ;
    < label: >MODEL response< *censor(list) > = effects </ options> ;
    < label: >MODEL (lower,upper)= effects </ options> ;
    < label: >MODEL events/trials = effects </ options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=name ... keyword=name> <options> ;
    PROBPLOT </ options> ;
    SLICE model-effect </ options> ;
    STORE <OUT= >item-store-name </ LABEL='label'> ;
    TEST  <model-effects> </ options> ;
    WEIGHT variable ;
```

The MODEL statement is required; it specifies both the variables that are used in the regression part of the model and the distribution that is used for the error (random) component of the model. Each invocation of the LIFEREG procedure can use only one MODEL statement. If multiple MODEL statements are present, only the last is used. You can specify main effects and interaction terms in the MODEL statement, as in the GLM procedure. You can specify initial values in the MODEL statement or in an INEST= data set. If no initial values are specified, the starting estimates are obtained by ordinary least squares. The CLASS statement determines which explanatory variables are treated as categorical. The WEIGHT statement identifies a *variable* with values that are used to weight the observations. Observations with zero or negative weights are not used to fit the model, although predicted values can be computed for them. The OUTPUT statement creates an output data set that contains predicted values and residuals.

The ESTIMATE, LSMEANS, LSMESTIMATE, SLICE, STORE, and TEST statements are common to many procedures. Summary descriptions of functionality and syntax for these statements are also given after the PROC LIFEREG statement in alphabetical order, and full documentation about them is available in Chapter 20, "Shared Concepts and Topics."

PROC LIFEREG Statement

PROC LIFEREG < *options* > **;**

The PROC LIFEREG statement invokes the LIFEREG procedure. Table 76.1 summarizes the *options* available in the PROC LIFEREG statement.

Table 76.1 PROC LIFEREG Statement Options

Option	Description
COVOUT	Writes the estimated covariance matrix to the OUTEST= data set
DATA=	Specifies the input SAS data set
INEST=	Specifies an input SAS data set that contains initial estimates
NAMELEN=	Specifies the length of effect names
NOPRINT	Suppresses the display of the output
ORDER=	Specifies the sort order for the levels of the classification variables
OUTEST=	Specifies an output SAS data set
PLOTS=	Controls graphics created by ODS Graphics
XDATA=	Specifies a SAS input data containing values for the independent variables

You can specify the following *options* in the PROC LIFEREG statement.

COVOUT
writes the estimated covariance matrix to the OUTEST= data set if convergence is attained.

DATA=*SAS-data-set*
specifies the input SAS data set used by PROC LIFEREG. By default, the most recently created SAS data set is used.

INEST=*SAS-data-set*
specifies an input SAS data set that contains initial estimates for all the parameters in the model. See the section "INEST= Data Set" on page 5693 for a detailed description of the contents of the INEST= data set.

NAMELEN=*n*
specifies the length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 200. The default length is 20 characters.

NOPRINT
suppresses the display of the output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUTEST=*SAS-data-set*

specifies an output SAS data set containing the parameter estimates, the maximized log likelihood, and, if the COVOUT option is specified, the estimated covariance matrix. See the section "OUTEST= Data Set" on page 5693 for a detailed description of the contents of the OUTEST= data set.

PLOTS=NONE | PROBPLOT

specifies options that control graphics created by ODS Graphics.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc lifereg plots=probplot;
   model y = x;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The following *plot-requests* are available.

NONE suppresses any plots created by ODS Graphics specified in other LIFEREG statements, such as the BAYES or PROBPLOT statement.

PROBPLOT creates a default probability plot based on information in the MODEL statement. If a PROBPLOT statement is also specified, the probability plot specified in that statement is created, and this option is ignored.

XDATA=*SAS-data-set*

specifies an input SAS data set that contains values for all the independent variables in the MODEL statement and variables in the CLASS statement for probability plotting. If there are covariates specified in a MODEL statement and a probability plot is requested with a PROBPLOT statement, you specify fixed values for the effects in the MODEL statement with the XDATA= data set. See the section "XDATA= Data Set" on page 5694 for a detailed description of the contents of the XDATA= data set.

BAYES Statement

BAYES < *options* > **;**

The BAYES statement requests a Bayesian analysis of the regression model by using Gibbs sampling. The Bayesian posterior samples (also known as the chain) for the regression parameters are not tabulated. The Bayesian posterior samples (also known as the chain) for the model parameters can be output to a SAS data set.

Table 76.2 summarizes the *options* available in the BAYES statement.

Table 76.2 BAYES Statement Options

Option	Description
Monte Carlo Options	
INITIAL=	Specifies initial values of the chain
INITIALMLE	Specifies that maximum likelihood estimates be used as initial values of the chain
METROPOLIS=	Specifies the use of a Metropolis step
NBI=	Specifies the number of burn-in iterations
NMC=	Specifies the number of iterations after burn-in
SEED=	Specifies the random number generator seed
THINNING=	Controls the thinning of the Markov chain
Model and Prior Options	
COEFFPRIOR=	Specifies the prior of the regression coefficients
EXPONENTIALSCALEPRIOR=	Specifies the prior of the exponential scale parameter
GAMMASHAPEPRIOR=	Specifies the prior of the three-parameter gamma shape parameter
SCALEPRIOR=	Specifies the prior of the scale parameter
WEIBULLSCALEPRIOR=	Specifies the prior of the Weibull scale parameter
WEIBULLSHAPEPRIOR=	Specifies the prior of the Weibull shape parameter
Summary Statistics and Convergence Diagnostics	
DIAGNOSTICS=	Displays convergence diagnostics
PLOTS=	Displays diagnostic plots
STATISTICS=	Displays summary statistics of the posterior samples
Posterior Samples	
OUTPOST=	Names a SAS data set for the posterior samples

The following list describes these *options* and their suboptions.

COEFFPRIOR=UNIFORM | NORMAL < (*normal-options*) >

CPRIOR=UNIFORM | NORMAL < (*option*) >

COEFF=UNIFORM | NORMAL < (*option*) >

specifies the prior distribution for the regression coefficients. The default is COEFFPRIOR=UNIFORM. The available prior distributions are as follows:

NORMAL< (*normal-option*) >

specifies a normal distribution. The *normal-options* include the following:

CONDITIONAL

specifies that the normal prior, conditional on the current Markov chain value of the location-scale model precision parameter $\tau = \frac{1}{\sigma^2}$, is $N(\boldsymbol{\mu}, \tau^{-1}\boldsymbol{\Sigma})$, where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of the normal prior specified by other normal options.

INPUT= *SAS-data-set*

specifies a SAS data set that contains the mean and covariance information of the normal prior. The data set must have a _TYPE_ variable to represent the type of each observation and a variable for each regression coefficient. If the data set also contains a _NAME_ variable, the values of this variable are used to identify the covariances for the _TYPE_='COV' observations; otherwise, the _TYPE_='COV' observations are assumed to be in the same order as the explanatory variables in the MODEL statement. PROC LIFEREG reads the mean vector from the observation with _TYPE_='MEAN' and reads the covariance matrix from observations with _TYPE_='COV'. For an independent normal prior, the variances can be specified with _TYPE_='VAR'; alternatively, the precisions (inverse of the variances) can be specified with _TYPE_='PRECISION'.

RELVAR< =c >

specifies the normal prior $N(\mathbf{0}, c\mathbf{J})$, where $\mathbf{J}$ is a diagonal matrix with diagonal elements equal to the variances of the corresponding ML estimator. By default, $c = 10^6$.

VAR< =c >

specifies the normal prior $N(\mathbf{0}, c\mathbf{I})$, where $\mathbf{I}$ is the identity matrix.

If you do not specify an option, the normal prior $N(\mathbf{0}, 10^6\mathbf{I})$, where $\mathbf{I}$ is the identity matrix, is used. See the section "Normal Prior" on page 5697 for more details.

UNIFORM

specifies a flat prior—that is, the prior that is proportional to a constant ($p(\beta_1, \ldots, \beta_k) \propto 1$ for all $-\infty < \beta_i < \infty$).

DIAGNOSTICS=ALL | NONE | (*keyword-list*)

DIAG=ALL | NONE | (*keyword-list*)

controls the number of diagnostics produced. You can request all the following diagnostics by specifying DIAGNOSTICS=ALL. If you do not want any of these diagnostics, specify DIAGNOSTICS=NONE. If you want some but not all of the diagnostics, or if you want to change certain settings of these diagnostics, specify a subset of the following *keywords*. The default is DIAGNOSTICS=(AUTOCORR ESS GEWEKE).

AUTOCORR < (LAGS= *numeric-list*) >

computes the autocorrelations of lags given by LAGS= list for each parameter. Elements in the list are truncated to integers and repeated values are removed. If the LAGS= option is not specified, autocorrelations of lags 1, 5, 10, and 50 are computed for each variable. See the section "Autocorrelations" on page 174 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

ESS

computes Carlin's estimate of the effective sample size, the correlation time, and the efficiency of the chain for each parameter. See the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

GELMAN < (*gelman-options***) >**

computes the Gelman and Rubin convergence diagnostics. You can specify one or more of the following *gelman-options*:

NCHAIN=*number*

N=*number*

specifies the number of parallel chains used to compute the diagnostic, and must be 2 or larger. The default is NCHAIN=3. If an INITIAL= data set is used, NCHAIN defaults to the number of rows in the INITIAL= data set. If any number other than this is specified with the NCHAIN= option, the NCHAIN= value is ignored.

ALPHA=*value*

specifies the significance level for the upper bound. The default is ALPHA=0.05, resulting in a 97.5% bound.

See the section "Gelman and Rubin Diagnostics" on page 167 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

GEWEKE < (*geweke-options***) >**

computes the Geweke spectral density diagnostics, which are essentially a two-sample t test between the first f_1 portion and the last f_2 portion of the chain. The default is $f_1 = 0.1$ and $f_2 = 0.5$, but you can choose other fractions by using the following *geweke-options*:

FRAC1=*value*

specifies the fraction f_1 for the first window.

FRAC2=*value*

specifies the fraction f_2 for the second window.

See the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

HEIDELBERGER < (*heidel-options***) >**

computes the Heidelberger and Welch diagnostic for each variable, which consists of a stationarity test of the null hypothesis that the sample values form a stationary process. If the stationarity test is not rejected, a halfwidth test is then carried out. Optionally, you can specify one or more of the following *heidel-options*:

SALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the stationarity test.

HALPHA=*value*
specifies the α level ($0 < \alpha < 1$) for the halfwidth test.

EPS=*value*
specifies a positive number ϵ such that if the halfwidth is less than ϵ times the sample mean of the retained iterates, the halfwidth test is passed.

See the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

MCSE
MCERROR
computes the Monte Carlo standard error for each parameter. The Monte Caro standard error, which measures the simulation accuracy, is the standard error of the posterior mean estimate and is calculated as the posterior standard deviation divided by the square root of the effective sample size. See the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

RAFTERY< (*raftery-options***) >**
computes the Raftery and Lewis diagnostics that evaluate the accuracy of the estimated quantile ($\hat{\theta}_Q$ for a given $Q \in (0,1)$) of a chain. $\hat{\theta}_Q$ can achieve any degree of accuracy when the chain is allowed to run for a long time. A stopping criterion is when the estimated probability $\hat{P}_Q = \Pr(\theta \leq \hat{\theta}_Q)$ reaches within $\pm R$ of the value Q with probability S; that is, $\Pr(Q - R \leq \hat{P}_Q \leq Q + R) = S$. The following *raftery-options* enable you to specify Q, R, S, and a precision level ϵ for the test:

QUANTILE | Q=*value*
specifies the order (a value between 0 and 1) of the quantile of interest. The default is 0.025.

ACCURACY | R=*value*
specifies a small positive number as the margin of error for measuring the accuracy of estimation of the quantile. The default is 0.005.

PROBABILITY | S=*value*
specifies the probability of attaining the accuracy of the estimation of the quantile. The default is 0.95.

EPSILON | EPS=*value*
specifies the tolerance level (a small positive number) for the stationary test. The default is 0.001.

See the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

EXPSCALEPRIOR=GAMMA< (*options*) > | IMPROPER

ESCALEPRIOR=GAMMA< (*options*) > | IMPROPER

ESCPRIOR=GAMMA< (*options*) > | IMPROPER

specifies that Gibbs sampling be performed on the exponential distribution scale parameter and the prior distribution for the scale parameter. This prior distribution applies only when the exponential distribution and no covariates are specified.

A gamma prior $G(a, b)$ with density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$ is specified by EXPSCALEPRIOR=GAMMA, which can be followed by one of the following *gamma-options* enclosed in parentheses. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. See the section "Gamma Prior" on page 5696 for more details. The default is $G(10^{-4}, 10^{-4})$.

RELSHAPE< =*c* >

specifies independent $G(c\hat{\alpha}, c)$ distribution, where $\hat{\alpha}$ is the MLE of the exponential scale parameter. With this choice of hyperparameters, the mean of the prior distribution is $\hat{\alpha}$ and the variance is $\frac{\hat{\alpha}}{c^2}$. By default, *c*=10^{-4}.

SHAPE=*a*

ISCALE=*b*

when both specified, results in a $G(a, b)$ prior.

SHAPE=*c*

when specified alone, results in a $G(c, c)$ prior.

ISCALE=*c*

when specified alone, results in a $G(c, c)$ prior.

An improper prior with density $f(t)$ proportional to t^{-1} is specified with EXPSCALEPRIOR=IMPROPER.

GAMMASHAPEPRIOR=NORMAL< (*options*) >

GAMASHAPEPRIOR=NORMAL< (*options*) >

SHAPE1PRIOR=NORMAL< (*options*) >

specifies the prior distribution for the gamma distribution shape parameter. If you do not specify any options in a gamma model, the $N(0, 10^6)$ prior for the shape is used. You can specify MEAN= and VAR= or RELVAR= options, either alone or together, to specify the mean and variance of the normal prior for the gamma shape parameter.

MEAN=*a*

specifies a normal prior $N(a, 10^6)$. By default, *a*=0.

RELVAR< =*b* >

specifies the normal prior $N(0, bJ)$, where J is the variance of the MLE of the shape parameter. By default, *b*=10^6.

VAR=*c*

specifies the normal prior $N(0, c)$. By default, *c*=10^6.

INITIAL=*SAS-data-set*

specifies the SAS data set that contains the initial values of the Markov chains. The INITIAL= data set must contain all the variables of the model. You can specify multiple rows as the initial values of the parallel chains for the Gelman-Rubin statistics, but posterior summaries, diagnostics, and plots are computed only for the first chain. If the data set also contains the variable _SEED_, the value of the _SEED_ variable is used as the seed of the random number generator for the corresponding chain.

INITIALMLE

specifies that maximum likelihood estimates of the model parameters be used as initial values of the Markov chain. If this option is not specified, estimates of the mode of the posterior distribution obtained by optimization are used as initial values.

METROPOLIS=YES | NO

specifies the use of a Metropolis step to generate Gibbs samples for posterior distributions that are not log concave. The default value is METROPOLIS=YES.

NBI=*number*

specifies the number of burn-in iterations before the chains are saved. The default is 2000.

NMC=*number*

specifies the number of iterations after the burn-in. The default is 10000.

OUTPOST=*SAS-data-set*

OUT=*SAS-data-set*

names the SAS data set that contains the posterior samples. See the section "OUTPOST= Output Data Set" on page 5698 for more information. Alternatively, you can create the output data set by specifying an ODS OUTPUT statement as follows:

ODS OUTPUT POSTERIORSAMPLE=*SAS-data-set*

PLOTS<(*global-plot-options***)>=** *plot-request*

PLOTS<(*global-plot-options***)>=(***plot-request* < . . . *plot-request*>**)**

controls the display of diagnostic plots. Three types of plots can be requested: trace plots, autocorrelation function plots, and kernel density plots. By default, the plots are displayed in panels unless the global plot option UNPACK is specified. Also, when specifying more than one type of plots, the plots are displayed by parameters unless the global plot option GROUPBY is specified. When you specify only one plot request, you can omit the parentheses around the plot request. For example:

```
plots=none
plots(unpack)=trace
plots=(trace autocorr)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc lifereg;
   model y=x;
   bayes plots=trace;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The *global-plot-options* are as follows:

FRINGE
creates a fringe plot on the X axis of the density plot.

GROUPBY=PARAMETER | TYPE
specifies how the plots are grouped when there is more than one type of plot.

GROUPBY=TYPE
specifies that the plots be grouped by type.

GROUPBY=PARAMETER
specifies that the plots be grouped by parameter.

GROUPBY=PARAMETER is the default.

LAGS=*n*
specifies that autocorrelations be plotted up to lag *n*. If this option is not specified, autocorrelations are plotted up to lag 50.

SMOOTH
displays a fitted penalized B-spline curve for each trace plot.

UNPACKPANEL
UNPACK
specifies that all paneled plots be unpacked, meaning that each plot in a panel is displayed separately.

The *plot-requests* include the following:

ALL
specifies all types of plots. PLOTS=ALL is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

AUTOCORR
displays the autocorrelation function plots for the parameters.

DENSITY
displays the kernel density plots for the parameters.

NONE
suppresses all diagnostic plots.

TRACE
displays the trace plots for the parameters. See the section "Visual Analysis via Trace Plots" on page 163 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details.

SCALEPRIOR=GAMMA< (*options*) >

specifies that Gibbs sampling be performed on the location-scale model scale parameter and the prior distribution for the scale parameter.

A gamma prior $G(a, b)$ with density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$ is specified by SCALEPRIOR=GAMMA, which can be followed by one of the following *gamma-options* enclosed in parentheses. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. See the section "Gamma Prior" on page 5696 for details. The default is $G(10^{-4}, 10^{-4})$.

RELSHAPE< =*c* >

specifies independent $G(c\hat{\sigma}, c)$ distribution, where $\hat{\sigma}$ is the MLE of the scale parameter. With this choice of hyperparameters, the mean of the prior distribution is $\hat{\sigma}$ and the variance is $\frac{\hat{\sigma}}{c}$. By default, *c*=10^{-4}.

SHAPE=*a*

ISCALE=*b*

when both specified, results in a $G(a, b)$ prior.

SHAPE=*c*

when specified alone, results in a $G(c, c)$ prior.

ISCALE=*c*

when specified alone, results in a $G(c, c)$ prior.

SEED=*number*

specifies an integer seed in the range 1 to $2^{31} - 1$ for the random number generator in the simulation. Specifying a seed enables you to reproduce identical Markov chains for the same specification. If the SEED= option is not specified, or if you specify a nonpositive seed, a random seed is derived from the time of day.

STATISTICS < (*global-options*) > = ALL | NONE | *keyword* | (*keyword-list*)

STATS < (*global-statoptions*) > = ALL | NONE | *keyword* | (*keyword-list*)

controls the number of posterior statistics produced. Specifying STATISTICS=ALL is equivalent to specifying STATISTICS= (SUMMARY INTERVAL COV CORR). If you do not want any posterior statistics, you specify STATISTICS=NONE. The default is STATISTICS=(SUMMARY INTERVAL). See the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details. The *global-options* include the following:

ALPHA=*numeric-list*

controls the probabilities of the credible intervals. The ALPHA= values must be between 0 and 1. Each ALPHA= value produces a pair of 100(1–ALPHA)% equal-tail and HPD intervals for each parameters. The default is ALPHA=0.05, which yields the 95% credible intervals for each parameter.

PERCENT=*numeric-list*

requests the percentile points of the posterior samples. The PERCENT= values must be between 0 and 100. The default is PERCENT=25, 50, 75, which yields the 25th, 50th, and 75th percentile points, respectively, for each parameter.

The list of *keywords* includes the following:

CORR
produces the posterior correlation matrix.

COV
produces the posterior covariance matrix.

SUMMARY
produces the means, standard deviations, and percentile points for the posterior samples. The default is to produce the 25th, 50th, and 75th percentile points, but you can use the global PERCENT= option to request specific percentile points.

INTERVAL
produces equal-tail credible intervals and HPD intervals. The default is to produce the 95% equal-tail credible intervals and 95% HPD intervals, but you can use the global ALPHA= option to request intervals of any probabilities.

NONE
suppresses printing all summary statistics.

THINNING=*number*
THIN=*number*
controls the thinning of the Markov chain. Only one in every k samples is used when THINNING=k, and if NBI=n_0 and NMC=n, the number of samples kept is

$$\left[\frac{n_0+n}{k}\right]-\left[\frac{n_0}{k}\right]$$

where $[a]$ represents the integer part of the number a. The default is THINNING=1.

WEIBULLSCALEPRIOR=GAMMA<(*options***)>**
WSCALEPRIOR=GAMMA<(*options***)>**
WSCPRIOR=GAMMA<(*options***)>**
specifies that Gibbs sampling be performed on the Weibull model scale parameter and the prior distribution for the scale parameter. This option applies only when a Weibull distribution and no covariates are specified. When this option is specified, PROC LIFEREG performs Gibbs sampling on the Weibull scale parameter, which is defined as $\exp(\mu)$, where μ is the intercept term.

A gamma prior $G(a,b)$ is specified by WEIBULLSCALEPRIOR=GAMMA, which can be followed by one of the following *gamma-options* enclosed in parentheses. The gamma probability density is given by $g(t)=\frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. See the section "Gamma Prior" on page 5696 for details about the gamma prior. The default is $G(10^{-4},10^{-4})$.

RELSHAPE<=*c***>**
specifies independent $G(c\hat{\alpha},c)$ distribution, where $\hat{\alpha}$ is the MLE of the Weibull scale parameter. With this choice of hyperparameters, the mean of the prior distribution is $\hat{\alpha}$ and the variance is $\frac{\hat{\alpha}}{c}$. By default, c=10^{-4}.

SHAPE=*a*

ISCALE=*b*

when both specified, results in a $G(a, b)$ prior.

SHAPE=*c*

when specified alone, results in a $G(c, c)$ prior.

ISCALE=*c*

when specified alone, results in a $G(c, c)$ prior.

WEIBULLSHAPEPRIOR=GAMMA< (*options***) >**

WSHAPEPRIOR=GAMMA< (*options***) >**

WSHPRIOR=GAMMA< (*options***) >**

specifies that Gibbs sampling be performed on the Weibull model shape parameter and the prior distribution for the shape parameter. When this option is specified, PROC LIFEREG performs Gibbs sampling on the Weibull shape parameter, which is defined as σ^{-1}, where σ is the location-scale model scale parameter.

A gamma prior $G(a, b)$ with density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$ is specified by WEIBULL-SHAPEPRIOR=GAMMA, which can be followed by one of the following *gamma-options* enclosed in parentheses. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. See the section "Gamma Prior" on page 5696 for details about the gamma prior. The default is $G(10^{-4}, 10^{-4})$.

RELSHAPE< =*c* **>**

specifies independent $G(c\hat{\beta}, c)$ distribution, where $\hat{\beta}$ is the MLE of the Weibull shape parameter. With this choice of hyperparameters, the mean of the prior distribution is $\hat{\beta}$ and the variance is $\frac{\hat{\beta}}{c}$. By default, *c*=10^{-4}.

SHAPE< =*a* **>**

ISCALE=*b*

when both specified, results in a $G(a, b)$ prior.

SHAPE=*c*

when specified alone, results in a $G(c, c)$ prior.

ISCALE=*c*

when specified alone, results in a $G(c, c)$ prior.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC LIFEREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the LIFEREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / **TRUNCATE** > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC LIFEREG statement.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 76.3 summarizes the *options* available in the ESTIMATE statement.

Table 76.3 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the $\mathbf{L}$ matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement"

on page 475 in Chapter 20, "Shared Concepts and Topics."

INSET Statement

INSET < *keyword-list* > < / *options* > **;**

The box or table of summary information produced on plots made with the PROBPLOT statement is called an *inset*. You can use the INSET statement to customize the information that is displayed in the inset box as well as to customize the appearance of the inset box. To supply the information that is displayed in the inset box, you specify *keywords* corresponding to the information that you want shown. For example, the following statements produce a probability plot with the number of observations, the number of right-censored observations, the name of the distribution, and the estimated Weibull shape parameter in the inset:

```
proc lifereg data=epidemic;
   model life = dose / dist = Weibull;
   probplot;
   inset nobs right dist shape;
run;
```

By default, inset entries are identified with appropriate labels. However, you can provide a customized label by specifying the *keyword* for that entry followed by the equal sign (=) and the label in quotes. For example, the following INSET statement produces an inset containing the number of observations and the name of the distribution, labeled "Sample Size" and "Distribution" in the inset:

```
inset nobs='Sample Size' dist='Distribution';
```

If you specify a *keyword* that does not apply to the plot you are creating, then the *keyword* is ignored.

If you specify more than one INSET statement, only the first one is used.

Table 76.4 lists *keywords* available in the INSET statement to display summary statistics, distribution parameters, and distribution fitting information.

Table 76.4 INSET Statement Keywords

Keyword	Description
CONFIDENCE	Confidence coefficient for all confidence intervals
DIST	Name of the distribution
INTERVAL	Number of interval-censored observations
LEFT	Number of left-censored observations
NOBS	Number of observations
NMISS	Number of observations with missing values
RIGHT	Number of right-censored observations
SCALE	Value of the scale parameter
SHAPE	Value of the shape parameter
UNCENSORED	Number of uncensored observations

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 76.5 summarizes the *options* available in the LSMEANS statement.

Table 76.5 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers

Table 76.5 *continued*

Option	Description
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 76.6 summarizes the *options* available in the LSMESTIMATE statement.

Table 76.6 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference

Table 76.6 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

< *label*: > **MODEL** *response*< ∗*censor*(*list*) > **=** *effects* < / *options* > **;**

< *label*: > **MODEL** (*lower*,*upper*)**=** *effects* < / *options* > **;**

< *label*: > **MODEL** *events/trials* **=** *effects* < / *options* > **;**

Only a single MODEL statement can be used with one invocation of the LIFEREG procedure. If multiple MODEL statements are present, only the last is used. The optional *label* is used to label the model estimates in the output SAS data set and OUTEST= data set.

The first MODEL syntax is appropriate for right censoring. The variable *response* is possibly right censored. If the *response* variable can be right censored, then a second variable, denoted *censor*, must appear after the *response* variable with a list of parenthesized values, separated by commas or blanks, to indicate censoring. That is, if the *censor* variable takes on a value given in the list, the *response* is a right-censored value; otherwise, it is an observed value.

The second MODEL syntax specifies two variables, *lower* and *upper*, that contain values of the endpoints of the censoring interval. If the two values are the same (and not missing), it is assumed that there is no censoring and the actual response value is observed. If the lower value is missing, then the upper value is used as a left-censored value. If the upper value is missing, then the lower value is taken as a right-censored value. If both values are present and the lower value is less than the upper value, it is assumed that the values specify a censoring interval. If the lower value is greater than the upper value or both values are missing,

then the observation is not used in the analysis, although predicted values can still be obtained if none of the covariates are missing.

The following table summarizes the ways of specifying censoring.

lower	*upper*	**Comparison**	**Interpretation**
Not missing	Not missing	Equal	No censoring
Not missing	Not missing	Lower < upper	Censoring interval
Missing	Not missing		Upper used as left-censoring value
Not missing	Missing		Lower used as right-censoring value
Not missing	Not missing	Lower > upper	Observation not used
Missing	Missing		Observation not used

The third MODEL syntax specifies two variables that contain count data for a binary response. The value of the first variable, *events*, is the number of successes. The value of the second variable, *trials*, is the number of tries. The values of both *events* and (*trials-events*) must be nonnegative, and *trials* must be positive for the response to be valid. The values of the two variables do not need to be integers and are not modified to be integers.

The *effects* following the equal sign are the covariates in the model. Higher-order effects, such as interactions and nested terms, are allowed in the list, similar to the GLM procedure. Variable names and combinations of variable names representing higher-order terms are allowed to appear in this list. Classification, or CLASS, variables can be used as effects, and indicator variables are generated for the class levels. If you do not specify any covariates following the equal sign, an intercept-only model is fit.

Examples of three valid MODEL statements follow:

```
a: model time*flag(1,3)=temp;

b: model (start, finish)=;

c: model r/n=dose;
```

MODEL statement a indicates that the response is contained in a variable named time and that, if the variable flag takes on the values 1 or 3, the observation is right censored. The explanatory variable is temp, which could be a CLASS variable. MODEL statement b indicates that the response is known to be in the interval between the values of the variables start and finish and that there are no covariates except for a default intercept term. MODEL statement c indicates a binary response, with the variable r containing the number of responses and the variable n containing the number of trials.

Table 76.7 summarizes the *options* available in the MODEL statement.

Table 76.7 MODEL Statement Options

Option	Description
Model Specification	
ALPHA=	Sets the significance level
DISTRIBUTION=	Specifies the distribution type for failure time
NOLOG	Requests no log transformation of response
INTERCEPT=	Specifies initial estimate for intercept term
NOINT	Holds the intercept term fixed
INITIAL=	Specifies initial estimates for regression parameters
OFFSET=	Specifies an offset variable
SCALE=	Initializes the scale parameter
NOSCALE	Holds the scale parameter fixed
SHAPE1=	Initializes the first shape parameter
NOSHAPE1	Holds the first shape parameter fixed
Model Fitting	
CONVERGE=	Sets the convergence criterion
CONVG=	Sets the relative Hessian convergence criterion
MAXITER=	Sets the maximum number of iterations
SINGULAR=	Sets the tolerance for testing singularity
Output	
CORRB	Displays the estimated correlation matrix
COVB	Displays the estimated covariance matrix
ITPRINT	Displays the iteration history, final gradient, and second derivative matrix

The following *options* can appear in the MODEL statement.

ALPHA=*value*

sets the significance level for the confidence intervals for regression parameters and estimated survival probabilities. The value must be between 0 and 1. By default, ALPHA=0.05.

CONVERGE=*value*

sets the convergence criterion. Convergence is declared when the maximum change in the parameter estimates between Newton-Raphson steps is less than the value specified. The change is a relative change if the parameter is greater than 0.01 in absolute value; otherwise, it is an absolute change. By default, CONVERGE=1E–8.

CONVG=*value*

sets the relative Hessian convergence criterion; *value* must be between 0 and 1. After convergence is determined with the change in parameter criterion specified with the CONVERGE= option, the quantity $tc = \frac{\mathbf{g}'\mathbf{H}^{-1}\mathbf{g}}{|f|}$ is computed and compared to *value*, where $\mathbf{g}$ is the gradient vector, $\mathbf{H}$ is the Hessian matrix for the model parameters, and f is the log-likelihood function. If tc is greater than *value*, a warning that the relative Hessian convergence criterion has been exceeded is displayed. This criterion detects the occasional case where the change in parameter convergence criterion is satisfied, but a maximum in the log-likelihood function has not been attained. By default, CONVG=1E–4.

CORRB

produces the estimated correlation matrix of the parameter estimates.

COVB

produces the estimated covariance matrix of the parameter estimates.

DISTRIBUTION=*distribution-type*

DIST=*distribution-type*

D=*distribution-type*

specifies the distribution type assumed for the failure time. By default, PROC LIFEREG fits a type 1 extreme-value distribution to the log of the response. This is equivalent to fitting the Weibull distribution, since the scale parameter for the extreme-value distribution is related to a Weibull shape parameter and the intercept is related to the Weibull scale parameter in this case. When the NOLOG option is specified, PROC LIFEREG models the untransformed response with a type 1 extreme-value distribution as the default. See the section "Supported Distributions" on page 5681 for descriptions of the distributions. The following are valid values for *distribution-type*:

EXPONENTIAL	the exponential distribution, which is treated as a restricted Weibull distribution
GAMMA	a generalized gamma distribution (Lawless 2003, p. 240). The standard two-parameter gamma distribution is not available in PROC LIFEREG.
LLOGISTIC	a log-logistic distribution
LNORMAL	a lognormal distribution
LOGISTIC	a logistic distribution (equivalent to LLOGISTIC when the NOLOG option is specified)
NORMAL	a normal distribution (equivalent to LNORMAL when the NOLOG option is specified)
WEIBULL	a Weibull distribution. If NOLOG is specified, it fits a type 1 extreme-value distribution to the raw, untransformed data.

By default, PROC LIFEREG transforms the response with the natural logarithm before fitting the specified model when you specify the GAMMA, LLOGISTIC, LNORMAL, or WEIBULL option. You can suppress the log transformation with the NOLOG option. The following table summarizes the resulting distributions when the preceding distribution options are used in combination with the NOLOG option.

DISTRIBUTION=	NOLOG Specified?	Resulting Distribution
EXPONENTIAL	No	Exponential
EXPONENTIAL	Yes	One-parameter extreme value
GAMMA	No	Generalized log-gamma using the log of the response. (This is the same as fitting the generalized gamma using the untransformed response.)
GAMMA	Yes	Generalized log-gamma with untransformed responses
LOGISTIC	No	Logistic
LOGISTIC	Yes	Logistic (NOLOG has no effect)
LLOGISTIC	No	Log-logistic
LLOGISTIC	Yes	Logistic
LNORMAL	No	Lognormal
LNORMAL	Yes	Normal
NORMAL	No	Normal
NORMAL	Yes	Normal (NOLOG has no effect)
WEIBULL	No	Weibull
WEIBULL	Yes	Extreme value

INITIAL=*values*

sets initial values for the regression parameters. This option can be helpful in the case of convergence difficulty. Specified values are used to initialize the regression coefficients for the covariates specified in the MODEL statement. The intercept parameter is initialized with the INTERCEPT= option and is not included here. The values are assigned to the variables in the MODEL statement in the same order in which they are listed in the MODEL statement. Note that a CLASS variable requires $k - 1$ values when the CLASS variable takes on k different levels. The order of the CLASS levels is determined by the ORDER= option. If there is no intercept term, the first CLASS variable requires k initial values. If a BY statement is used, all CLASS variables must take on the same number of levels in each BY group or no meaningful initial values can be specified. The INITIAL= option can be specified as follows.

Type of List	Specification
List separated by blanks	`initial=3 4 5`
List separated by commas	`initial=3,4,5`
x to y	`initial=3 to 5`
x to y by z	`initial=3 to 5 by 1`
Combination of methods	`initial=1,3 to 5,9`

By default, PROC LIFEREG computes initial estimates with ordinary least squares. See the section "Computational Method" on page 5679 for details.

NOTE: The INITIAL= option is overwritten by the INEST= option. See the section "INEST= Data Set" on page 5693 for details.

INTERCEPT=*value*
initializes the intercept term to *value*. By default, the intercept is initialized by an ordinary least squares estimate.

ITPRINT
displays the iteration history for computing maximum likelihood estimates, the final evaluation of the gradient, and the final evaluation of the negative of the second derivative matrix—that is, the negative of the Hessian. If you perform a Bayesian analysis by specifying the BAYES statement, the iteration history for computing the mode of the posterior distribution is also displayed.

MAXITER=*n*
sets the maximum allowable number of iterations during the model estimation. By default, MAXITER=50.

NOINT
holds the intercept term fixed. Because of the usual log transformation of the response, the intercept parameter is usually a scale parameter for the untransformed response, or a location parameter for a transformed response.

NOLOG
requests that no log transformation of the response variable be performed. By default, PROC LIFEREG models the log of the response variable for the GAMMA, LLOGISTIC, LOGNORMAL, and WEIBULL distribution options. NOLOG is implicitly assumed for the NORMAL and LOGISTIC distribution options.

NOSCALE
holds the scale parameter fixed. Note that if the log transformation has been applied to the response, the effect of the scale parameter is a power transformation of the original response. If no SCALE= value is specified, the scale parameter is fixed at the value 1.

NOSHAPE1
holds the first shape parameter, SHAPE1, fixed. If no SHAPE1= value is specified, SHAPE1 is fixed at a value that depends on the DISTRIBUTION type.

OFFSET=*variable*
specifies a variable in the input data set to be used as an offset variable. This variable cannot be a CLASS variable, and it cannot be the response variable or one of the explanatory variables.

SCALE=*value*
initializes the scale parameter to *value*. If the Weibull distribution is specified, this scale parameter is the scale parameter of the type 1 extreme-value distribution, not the Weibull scale parameter. Note that, with a log transformation, the exponential model is the same as a Weibull model with the scale parameter fixed at the value 1.

SHAPE1=*value*
initializes the first shape parameter to *value*. If the specified distribution does not depend on this parameter, then this option has no effect. The only distribution that depends on this shape parameter is the generalized gamma distribution. See the section "Supported Distributions" on page 5681 for descriptions of the parameterizations of the distributions.

SINGULAR=*value*
sets the tolerance for testing singularity of the information matrix and the crossproducts matrix for the initial least squares estimates. Roughly, the test requires that a pivot be at least this value times the original diagonal value. By default, SINGULAR=1E–12.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *keyword*=*name* > . . . < *keyword*=*name* > **;**

The OUTPUT statement creates a new SAS data set containing statistics calculated after fitting the model. At least one specification of the form *keyword=name* is required.

All variables in the original data set are included in the new data set, along with the variables created as options for the OUTPUT statement. These new variables contain fitted values and estimated quantiles. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. Each OUTPUT statement applies to the preceding MODEL statement. See Example 76.1 for illustrations of the OUTPUT statement.

The following specifications can appear in the OUTPUT statement:

OUT=*SAS-data-set*
specifies the new data set. By default, the procedure uses the DATA*n* convention to name the new data set.

keyword=name
specifies the statistics to include in the output data set and gives names to the new variables. Specify a *keyword* for each desired statistic (see the following list of *keywords*), an equal sign, and the variable to contain the statistic.

The *keywords* allowed and the statistics they represent are as follows:

CENSORED=*variable*
specifies a *variable* to signal whether an observation is censored, and the type of censoring. The variable takes on values according to Table 76.8.

Table 76.8 Censoring Variable Values

Type of Response	CENSORED Variable Value
Uncensored	0
Right-censored	1
Left-censored	2
Interval-censored	3

CDF=*variable*

specifies a *variable* to contain the estimates of the cumulative distribution function evaluated at the observed response. If the data are interval censored, then the cumulative distribution function is evaluated at the response lower interval endpoint. See the section "Predicted Values" on page 5684 for more information.

CONTROL=*variable*

specifies a *variable* in the input data set to control the estimation of quantiles. See Example 76.1 for an illustration. If the specified variable has the value 1, estimates for all the values listed in the QUANTILE= list are computed for that observation in the input data set; otherwise, no estimates are computed. If no CONTROL= variable is specified, all quantiles are estimated for all observations. If the response variable in the MODEL statement is binomial, then this option has no effect.

CRESIDUAL | CRES=*variable*

specifies a *variable* to contain the Cox-Snell residuals

$$-\log(S(u_i))$$

where S is the standard survival function and

$$u_i = \frac{y_i - \mathbf{x}_i'\mathbf{b}}{\sigma}$$

If the data are interval censored, residuals are computed for y_i values corresponding to lower interval endpoints. If the response variable in the corresponding model statement is binomial, then the residuals are not computed, and this variable contains missing values.

SRESIDUAL | SRES=*variable*

specifies a *variable* to contain the standardized residuals

$$\frac{y_i - \mathbf{x}_i'\mathbf{b}}{\sigma}$$

If the data are interval censored, residuals are computed for y_i values corresponding to lower interval endpoints. If the response variable in the corresponding model statement is binomial, then the residuals are not computed, and this variable contains missing values.

PREDICTED | P=*variable*

specifies a *variable* to contain the quantile estimates. If the response variable in the corresponding model statement is binomial, then this variable contains the estimated probabilities, $1 - F(-\mathbf{x}'\mathbf{b})$.

QUANTILES | QUANTILE | Q=*value-list*

gives a list of *values* for which quantiles are calculated. The values must be between 0 and 1, noninclusive. For each value, a corresponding quantile is estimated. This option is not used if the response variable in the corresponding MODEL statement is binomial.

By default, QUANTILES=0.5. When the response is not binomial, a numeric variable, _PROB_, is added to the OUTPUT data set whenever the QUANTILES= option is specified. The variable _PROB_ gives the probability value for the quantile estimates. These are the values taken from the QUANTILES= list and are given as values between 0 and 1, not as values between 0 and 100. The list of QUANTILES values can be specified as in Table 76.9.

Table 76.9 Types of Value Lists

Type of List	Specification
List separated by blanks	`.2 .4 .6 .8`
List separated by commas	`.2,.4,.6,.8`
x to y	`.2 to .8`
x to y by z	`.2 to .8 by .1`
Combination of methods	`.1,.2 to .8 by .2`

STD_ERR | STD=*variable*

specifies a *variable* to contain the estimates of the standard errors of the estimated quantiles or $\mathbf{x}'\mathbf{b}$. If the response used in the MODEL statement is a binomial response, then these are the standard errors of $\mathbf{x}'\mathbf{b}$. Otherwise, they are the standard errors of the quantile estimates. These estimates can be used to compute confidence intervals for the quantiles. However, if the model is fit to the log of the event time, better confidence intervals can usually be computed by transforming the confidence intervals for the log response. See Example 76.1 for such a transformation.

XBETA=*variable*

specifies a *variable* to contain the computed value of $\mathbf{x}'\mathbf{b}$, where $\mathbf{x}$ is the covariate vector and $\mathbf{b}$ is the vector of parameter estimates.

PROBPLOT Statement

PROBPLOT | PPLOT <*/options*> **;**

You can use the PROBPLOT statement to create a probability plot from lifetime data. The data can be uncensored, right censored, or arbitrarily censored. You can specify any number of PROBPLOT statements after a MODEL statement. The syntax used for the response in the MODEL statement determines the type of censoring assumed in creating the probability plot. The model fit with the MODEL statement is plotted along with the data. If there are covariates in the model, they are set to constant values specified in the XDATA= data set when creating the probability plot. If no XDATA= data set is specified, continuous variables are set to their overall mean values and categorical variables specified in the CLASS statement are set to their highest levels.

Table 76.10 summarizes the *options* available in the PROBPLOT statement.

Table 76.10 PROBPLOT Statement Options

Option	Description
HCL	Computes and draws confidence limits for the predicted probabilities
HLOWER=	Specifies *value* as the lower lifetime axis tick mark
HUPPER=	Specifies *value* as the upper lifetime axis tick mark
HREF	Draws reference lines perpendicular to the horizontal axis
HREFLABELS=	Specifies labels for the lines requested by the HREF= option
ITPRINTEM	Displays the iteration history for the Turnbull algorithm

Table 76.10 *continued*

Option	Description
MAXITEM=	Specifies the maximum number of iterations for the Turnbull algorithm
NOCENPLOT	Suppresses the plotting of censored data points
NOCONF	Suppresses the default confidence bands
NODATA	Suppresses plotting of the estimated empirical probability plot
NOFIT	Suppresses the fitted probability (percentile) line and confidence bands
NOFRAME	Suppresses the frame around plotting areas
NOGRID	Suppresses grid lines
NOPOLISH	Suppresses setting small interval probabilities to zero in the Turnbull algorithm
NPINTERVALS=	Displays one of the two kinds of confidence limits
PCTLIST=	Specifies the list of percentages for which to compute percentile estimates
PLOWER=	Specifies the lower limit on the probability axis scale
PPOS=	Specifies the plotting position type
PPOUT	Displays a table of the cumulative probabilities
PRINTPROBS	Displays intervals and associated probabilities for the Turnbull algorithm
PROBLIST=	Specifies the list of initial values for the Turnbull algorithm
PUPPER=	Specifies the upper limit on the probability axis scale
ROTATE	Requests probability plots with probability scale on the horizontal axis
SQUARE	Makes the layout of the probability plots square
TOLLIKE=	Specifies the criterion for convergence in the Turnbull algorithm
TOLPROB=	Specifies the criterion for setting the interval probability to zero in the Turnbull algorithm
VREF	Draws reference lines perpendicular to the vertical axis
VREFLABELS=	Specifies labels for the lines requested by the VREF= option

You can specify the following *options* to control the content, layout, and appearance of a probability plot.

The following *options* are available if ODS Graphics is enabled.

HCL
computes and draws confidence limits for the predicted probabilities in the horizontal direction.

HLOWER=*value*
specifies the lower limit on the lifetime axis scale. The HLOWER= option specifies *value* as the lower lifetime axis tick mark. The tick mark interval and the upper axis limit are determined automatically.

HUPPER=*value*
specifies *value* as the upper lifetime axis tick mark. The tick mark interval and the lower axis limit are determined automatically.

HREF < (INTERSECT) > =*value-list*
requests reference lines perpendicular to the horizontal axis be drawn at horizontal axis values in the *value-list*. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified with the HREFLABELS= option, the intersecting vertical axis

reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS=*'label1'* **...** *'labeln'*

HREFLABEL=*'label1'* **...** *'labeln'*

HREFLAB=*'label1'* **...** *'labeln'*

specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

ITPRINTEM

displays the iteration history for the Turnbull algorithm.

MAXITEM=*n1 <,n2>*

specifies the maximum number of iterations for the Turnbull algorithm. Iteration history will be displayed in increments of *n2* if requested with the ITPRINTEM option. See the section "Arbitrarily Censored Data" on page 5690 for details.

NOCENPLOT

suppresses the plotting of censored data points.

NOCONF

suppresses the default confidence bands on the probability plot.

NODATA

suppresses plotting of the estimated empirical probability plot.

NOFIT

suppresses the fitted probability (percentile) line and confidence bands.

NOFRAME

suppresses the frame around plotting areas.

NOGRID

suppresses grid lines.

NOPOLISH

suppresses setting small interval probabilities to zero in the Turnbull algorithm.

NPINTERVALS=*interval-type*

specifies one of the two kinds of confidence limits for the estimated cumulative probabilities, pointwise (NPINTERVALS=POINT) or simultaneous (NPINTERVALS=SIMUL), requested by the PPOUT option to be displayed in the tabular output.

PCTLIST=*value-list*

specifies the list of percentages for which to compute percentile estimates; *value-list* must be a list of values separated by blanks or commas. Each value in the list must be between 0 and 100.

PLOWER=*value*
specifies the lower limit on the probability axis scale. The PLOWER= option specifies *value* as the lower probability axis tick mark. The tick mark interval and the upper axis limit are determined automatically.

PPOS=*plotting-position-type*
specifies the plotting position type. See the section "Probability Plotting" on page 5688 for details.

PPOS=	Method
EXPRANK	Expected ranks
MEDRANK	Median ranks
MEDRANK1	Median ranks (exact formula)
KM	Kaplan-Meier
MKM	Modified Kaplan-Meier (default)

PPOUT
specifies that a table of the cumulative probabilities plotted on the probability plot be displayed. Kaplan-Meier estimates of the cumulative probabilities are also displayed, along with standard errors and confidence limits. The confidence limits can be pointwise or simultaneous, as specified by the NPINTERVALS= option.

PRINTPROBS
displays intervals and associated probabilities for the Turnbull algorithm.

PROBLIST=*value-list*
specifies the list of initial values for the Turnbull algorithm.

PUPPER=*value*
specifies the upper limit on the probability axis scale. The PUPPER= option specifies *value* as the upper probability axis tick mark. The tick mark interval and the lower axis limit are determined automatically.

ROTATE
requests probability plots with probability scale on the horizontal axis.

SQUARE
makes the layout of the probability plots square.

TOLLIKE=*value*
specifies the criterion for convergence in the Turnbull algorithm.

TOLPROB=*value*
specifies the criterion for setting the interval probability to zero in the Turnbull algorithm.

VREF< (INTERSECT) >=*value-list*
requests reference lines perpendicular to the vertical axis be drawn at vertical axis values in the *value-list*. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified with the VREFLABELS= option, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS=*'label1' . . . 'labeln'*

VREFLABEL=*'label1' . . . 'labeln'*

VREFLAB=*'label1' . . . 'labeln'*

specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

NOTE: The information that is stored by the STORE statement in PROC LIFEREG is a subset of what is usually stored by other procedures that implement this statement. Because only a subset of information is stored, postprocessing features of PROC PLM that use the CODE, EFFECTPLOT, and SCORE statements are not available for item stores that PROC LIFEREG creates.

TEST Statement

TEST < *model-effects* > < / *options* > **;**

The TEST statement enables you to perform chi-square tests for model effects that test Type I, Type II, or Type III hypotheses. By default, the Type III tests are performed. For more information, see Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

If you want to use weights for each observation in the input data set, place the weights in a variable in the data set and specify the name in a WEIGHT statement. The values of the WEIGHT variable can be nonintegral and are not truncated. Observations with nonpositive or missing values for the weight variable do not contribute to the fit of the model. The WEIGHT variable multiplies the contribution to the log likelihood for each observation.

Details: LIFEREG Procedure

Missing Values

Any observation with missing values for the dependent variable is not used in the model estimation unless it is one and only one of the values in an interval specification. Also, if one of the explanatory variables or the censoring variable is missing, the observation is not used. For any observation to be used in the estimation of a model, only the variables needed in that model have to be nonmissing. Predicted values are computed for all observations with no missing explanatory variable values. If the censoring variable is missing, the CENSORED= variable in the OUT= SAS data set is also missing.

Model Specification

Main effects as well as interaction terms are allowed in the model specification, similar to the GLM procedure. For numeric variables, a main effect is a linear term equal to the value of the variable unless the variable appears in the CLASS statement. For variables listed in the CLASS statement, PROC LIFEREG creates indicator variables (variables taking the values zero or one) for every level of the variable except the last level. If there is no intercept term, the first CLASS variable has indicator variables created for all levels including the last level. The levels are ordered according to the ORDER= option. Estimates of a main effect depend upon other effects in the model and, therefore, are adjusted for the presence of other effects in the model.

Computational Method

By default, the LIFEREG procedure computes initial values for the parameters by using ordinary least squares (OLS) and ignoring censoring. This might not be the best set of starting values for a given set of data. For example, if there are extreme values in your data, the OLS fit might be excessively influenced by the extreme observations, causing an overflow or convergence problems. See Example 76.3 for one way to deal with convergence problems.

You can specify the INITIAL= option in the MODEL statement to override these starting values. You can also specify the INTERCEPT=, SCALE=, and SHAPE= options to set initial values of the intercept, scale, and shape parameters. For models with multilevel interaction effects, it is a little difficult to use the INITIAL= option to provide starting values for all parameters. In this case, you can use the INEST= data set. See the section "INEST= Data Set" on page 5693 for details. The INEST= data set overrides all previous specifications for starting values of parameters.

The rank of the design matrix $\mathbf{X}$ is estimated before the model is fit. Columns of $\mathbf{X}$ that are judged linearly dependent on other columns have the corresponding parameters set to zero. The test for linear dependence is controlled by the SINGULAR= option in the MODEL statement. Variables are included in the model in the order in which they are listed in the MODEL statement with the continuous variables included in the model before any classification variables.

The log-likelihood function is maximized by means of a ridge-stabilized Newton-Raphson algorithm. The maximized value of the log likelihood can take positive or negative values, depending on the specified model and the values of the maximum likelihood estimates of the model parameters.

If convergence of the maximum likelihood estimates is attained, a Type III chi-square test statistic is computed for each effect, testing whether there is any contribution from any of the levels of the effect. This statistic is computed as a quadratic form in the appropriate parameter estimates by using the corresponding submatrix of the asymptotic covariance matrix estimate. See Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions," for more information about Type III estimable functions. The asymptotic covariance matrix is computed as the inverse of the observed information matrix. Note that if the NOINT option is specified and CLASS variables are used, the first CLASS variable contains a contribution from an intercept term. The results are displayed in an ODS table named "Type3Analysis." Chi-square tests for individual parameters are Wald tests based on the observed information matrix and the parameter estimates. If an effect has a single degree of freedom in the parameter estimates table, the chi-square test for this parameter is equivalent to the Type III test for this effect.

Before SAS 8.2, a multiple-degree-of-freedom statistic was computed for each effect to test for contribution from any level of the effect. In general, the Type III test statistic in a main-effect-only model (no interaction terms) will be equal to the previously computed effect statistic, unless there are collinearities among the effects. If there are collinearities, the Type III statistic will adjust for them, and the value of the Type III statistic and the number of degrees of freedom might not be equal to those of the previous effect statistic.

Suppose there are n observations from the model $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \sigma\boldsymbol{\epsilon}$ (or $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{O} + \sigma\boldsymbol{\epsilon}$ if there is an offset variable), where $\mathbf{X}$ is an $n \times k$ matrix of covariate values (including the intercept), $\mathbf{y}$ is a vector of responses, $\mathbf{O}$ is a vector of offset variable values, and $\boldsymbol{\epsilon}$ is a vector of errors with survival function S, cumulative distribution function F, and probability density function f. That is, $S(t) = \Pr(\epsilon_i > t)$, $F(t) = \Pr(\epsilon_i \leq t)$, and $f(t) = dF(t)/dt$, where ϵ_i is a component of the error vector. Then, if all the responses are observed,

the log likelihood, L, can be written as

$$L = \sum \log\left(\frac{f(u_i)}{\sigma}\right)$$

where $u_i = \frac{1}{\sigma}(y_i - \mathbf{x}_i'\boldsymbol{\beta})$.

If some of the responses are left, right, or interval censored, the log likelihood can be written as

$$L = \sum \log\left(\frac{f(u_i)}{\sigma}\right) + \sum \log\left(S(u_i)\right) + \sum \log\left(F(u_i)\right) + \sum \log\left(F(u_i) - F(v_i)\right)$$

with the first sum over uncensored observations, the second sum over right-censored observations, the third sum over left-censored observations, the last sum over interval-censored observations, and

$$v_i = \frac{1}{\sigma}(z_i - \mathbf{x}_i'\boldsymbol{\beta})$$

where z_i is the lower end of a censoring interval.

If the response is specified in the binomial format, *events*/*trials*, then the log-likelihood function is

$$L = \sum r_i \log(P_i) + (n_i - r_i)\log(1 - P_i)$$

where r_i is the number of events and n_i is the number of trials for the ith observation. In this case, $P_i = 1 - F(-\mathbf{x}_i'\boldsymbol{\beta})$. For the symmetric distributions, logistic and normal, this is the same as $F(\mathbf{x}_i'\boldsymbol{\beta})$. Additional information about censored and limited dependent variable models can be found in Kalbfleisch and Prentice (1980) and Maddala (1983).

The estimated covariance matrix of the parameter estimates is computed as the negative inverse of $\mathbf{I}$, which is the information matrix of second derivatives of L with respect to the parameters evaluated at the final parameter estimates. If $\mathbf{I}$ is not positive definite, a positive-definite submatrix of $\mathbf{I}$ is inverted, and the remaining rows and columns of the inverse are set to zero. If some of the parameters, such as the scale and intercept, are restricted, the corresponding elements of the estimated covariance matrix are set to zero. The standard error estimates for the parameter estimates are taken as the square roots of the corresponding diagonal elements.

For restrictions placed on the intercept, scale, and shape parameters, one-degree-of-freedom Lagrange multiplier test statistics are computed. These statistics are computed as

$$\chi^2 = \frac{g^2}{V}$$

where g is the derivative of the log likelihood with respect to the restricted parameter at the restricted maximum and

$$V = \mathbf{I}_{11} - \mathbf{I}_{12}\mathbf{I}_{22}^{-1}\mathbf{I}_{21}$$

where the 1 subscripts refer to the restricted parameter and the 2 subscripts refer to the unrestricted parameters. The information matrix is evaluated at the restricted maximum. These statistics are asymptotically distributed as chi-squares with one degree of freedom under the null hypothesis that the restrictions are valid, provided that some regularity conditions are satisfied. See Rao (1973, p. 418) for a more complete discussion. It is possible for these statistics to be missing if the observed information matrix is not positive definite. Higher-degree-of-freedom tests for multiple restrictions are not currently computed.

A Lagrange multiplier test statistic is computed to test this constraint. Notice that this test statistic is comparable to the Wald test statistic for testing that the scale is one. The Wald statistic is the result of squaring the difference of the estimate of the scale parameter from one and dividing this by the square of its estimated standard error.

Supported Distributions

For most distributions, the baseline survival function (S) and the probability density function(f) are listed for the additive random disturbance (y_0 or $\log(T_0)$) with location parameter μ and scale parameter σ. See the section "Overview: LIFEREG Procedure" on page 5634 for more information. These distributions apply when the log of the response is modeled (this is the default analysis). The corresponding survival function (G) and its density function (g) are given for the untransformed baseline distribution (T_0).

For the normal and logistic distributions, the response is not log transformed by PROC LIFEREG, and the survival functions and probability density functions listed apply to the untransformed response.

For example, for the WEIBULL distribution, $S(w)$ and $f(w)$ are the survival function and the probability density function for the extreme-value distribution (distribution of the log of the response), while $G(t)$ and $g(t)$ are the survival function and the probability density function of a Weibull distribution (using the untransformed response).

The chosen baseline functions define the meaning of the intercept, scale, and shape parameters. Only the gamma distribution has a free shape parameter in the following parameterizations. Notice that some of the distributions do not have mean zero and that σ is not, in general, the standard deviation of the baseline distribution.

For the Weibull distribution, the accelerated failure time model is also a proportional-hazards model. However, the parameterization for the covariates differs by a multiple of the scale parameter from the parameterization commonly used for the proportional hazards model.

The distributions supported in the LIFEREG procedure follow. If there are no covariates in the model, μ = Intercept in the output; otherwise, $\mu = \mathbf{x}'\boldsymbol{\beta}$. σ = Scale in the output.

Exponential

$$
\begin{aligned}
S(w) &= \exp(-\exp(w-\mu)) \\
f(w) &= \exp(w-\mu)\exp(-\exp(w-\mu)) \\
G(t) &= \exp(-\alpha t) \\
g(t) &= \alpha\exp(-\alpha t)
\end{aligned}
$$

where $\exp(-\mu) = \alpha$.

Generalized Gamma

$S(w) = S'(u)$, $f(w) = \sigma^{-1} f'(u)$, $G(t) = G'(v)$, $g(t) = \frac{v}{t\sigma} g'(v)$, $u = \frac{w-\mu}{\sigma}$, $v = \exp(\frac{\log(t)-\mu}{\sigma})$, and

$$
\begin{aligned}
S'(u) &= \begin{cases} 1 - \frac{\Gamma(\delta^{-2}, \delta^{-2}\exp(\delta u))}{\Gamma(\delta^{-2})} & \text{if } \delta > 0 \\ \frac{\Gamma(\delta^{-2}, \delta^{-2}\exp(\delta u))}{\Gamma(\delta^{-2})} & \text{if } \delta < 0 \end{cases} \\
f'(u) &= \frac{|\delta|}{\Gamma\left(\delta^{-2}\right)} \left(\delta^{-2}\exp(\delta u)\right)^{\delta^{-2}} \exp\left(-\exp(\delta u)\delta^{-2}\right) \\
G'(v) &= \begin{cases} 1 - \frac{\Gamma(\delta^{-2}, \delta^{-2}v^{\delta})}{\Gamma(\delta^{-2})} & \text{if } \delta > 0 \\ \frac{\Gamma(\delta^{-2}, \delta^{-2}v^{\delta})}{\Gamma(\delta^{-2})} & \text{if } \delta < 0 \end{cases} \\
g'(v) &= \frac{|\delta|}{v\Gamma\left(\delta^{-2}\right)} \left(\delta^{-2}v^{\delta}\right)^{\delta^{-2}} \exp\left(-v^{\delta}\delta^{-2}\right)
\end{aligned}
$$

where $\Gamma(a)$ denotes the complete gamma function, $\Gamma(a, z)$ denotes the incomplete gamma function, and δ is a free shape parameter. The δ parameter is called Shape by PROC LIFEREG. See Lawless (2003, p. 240), and Klein and Moeschberger (1997, p. 386) for a description of the generalized gamma distribution.

Logistic

$$
\begin{aligned}
S(w) &= \left(1 + \exp\left(\frac{w-\mu}{\sigma}\right)\right)^{-1} \\
f(w) &= \frac{\exp\left(\frac{w-\mu}{\sigma}\right)}{\sigma\left(1 + \exp\left(\frac{w-\mu}{\sigma}\right)\right)^2}
\end{aligned}
$$

Log-Logistic

$$
\begin{aligned}
S(w) &= \left(1 + \exp\left(\frac{w-\mu}{\sigma}\right)\right)^{-1} \\
f(w) &= \frac{\exp\left(\frac{w-\mu}{\sigma}\right)}{\sigma\left(1 + \exp\left(\frac{w-\mu}{\sigma}\right)\right)^2} \\
G(t) &= \frac{1}{1 + \alpha t^{\gamma}} \\
g(t) &= \frac{\alpha\gamma t^{\gamma-1}}{(1 + \alpha t^{\gamma})^2}
\end{aligned}
$$

where $\gamma = 1/\sigma$ and $\alpha = \exp(-\mu/\sigma)$.

Lognormal

$$
\begin{aligned}
S(w) &= 1 - \Phi\left(\frac{w-\mu}{\sigma}\right) \\
f(w) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left(\frac{w-\mu}{\sigma}\right)^2\right) \\
G(t) &= 1 - \Phi\left(\frac{\log(t)-\mu}{\sigma}\right) \\
g(t) &= \frac{1}{\sqrt{2\pi}\sigma t} \exp\left(-\frac{1}{2}\left(\frac{\log(t)-\mu}{\sigma}\right)^2\right)
\end{aligned}
$$

where Φ is the cumulative distribution function for the normal distribution.

Normal

$$
\begin{aligned}
S(w) &= 1 - \Phi\left(\frac{w-\mu}{\sigma}\right) \\
f(w) &= \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\left(\frac{w-\mu}{\sigma}\right)^2\right)
\end{aligned}
$$

where Φ is the cumulative distribution function for the normal distribution.

Weibull

$$
\begin{aligned}
S(w) &= \exp\left(-\exp\left(\frac{w-\mu}{\sigma}\right)\right) \\
f(w) &= \frac{1}{\sigma} \exp\left(\frac{w-\mu}{\sigma}\right) \exp\left(-\exp\left(\frac{w-\mu}{\sigma}\right)\right) \\
G(t) &= \exp\left(-\alpha t^{\gamma}\right) \\
g(t) &= \gamma \alpha t^{\gamma-1} \exp\left(-\alpha t^{\gamma}\right)
\end{aligned}
$$

where $\sigma = 1/\gamma$ and $\alpha = \exp(-\mu/\sigma)$.

If your parameterization is different from the ones shown here, you can still use the procedure to fit your model. For example, a common parameterization for the Weibull distribution is

$$g(t;\lambda,\beta) = \left(\frac{\beta}{\lambda}\right)\left(\frac{t}{\lambda}\right)^{\beta-1}\exp\left(-\left(\frac{t}{\lambda}\right)^{\beta}\right)$$

$$G(t;\lambda,\beta) = \exp\left(-\left(\frac{t}{\lambda}\right)^{\beta}\right)$$

so that $\lambda = \exp(\mu)$ and $\beta = 1/\sigma$.

Again note that the expected value of the baseline log response is, in general, not zero and that the distributions are not symmetric in all cases. Thus, for a given set of covariates, $\mathbf{x}$, the expected value of the log response is not always $\mathbf{x}'\boldsymbol{\beta}$.

Some relations among the distributions are as follows:

- The gamma with Shape=1 is a Weibull distribution.
- The gamma with Shape=0 is a lognormal distribution.
- The Weibull with Scale=1 is an exponential distribution.

Predicted Values

For a given set of covariates, $\mathbf{x}$ (including the intercept term), the pth quantile of the log response, y_p, is given by

$$y_p = \mathbf{x}'\boldsymbol{\beta} + \sigma u_p$$

if no offset variable has been specified, or

$$y_p = \mathbf{x}'\boldsymbol{\beta} + \mathrm{o} + \sigma u_p$$

for a given value o of an offset variable, where u_p is the pth quantile of the baseline distribution. The estimated quantile is computed by replacing the unknown parameters with their estimates, including any shape parameters on which the baseline distribution might depend. The estimated quantile of the original response is obtained by taking the exponential of the estimated log quantile unless the NOLOG option is specified in the preceding MODEL statement.

The following table shows how u_p is computed from the baseline distribution $F(u)$:

Table 76.11 Baseline Probability Functions and u_p

Distribution	$F(u)$		u_p
Exponential	$1-\exp(-\exp(u))$		$\log(-\log(1-p))$
Generalized gamma	$\begin{cases} \frac{\Gamma(\delta^{-2},\delta^{-2}\exp(\delta u))}{\Gamma(\delta^{-2})} & \text{if } \delta > 0 \\ 1-\frac{\Gamma(\delta^{-2},\delta^{-2}\exp(\delta u))}{\Gamma(\delta^{-2})} & \text{if } \delta < 0 \end{cases}$		$F^{-1}(p)$
Logistic	$1-(1+\exp(u))^{-1}$		$\log(p/(1-p))$
Log-logistic	$1-(1+\exp(u))^{-1}$		$\log(p/(1-p))$
Lognormal	$\Phi(u)$		$\Phi^{-1}(p)$
Normal	$\Phi(u)$		$\Phi^{-1}(p)$
Weibull	$1-\exp(-\exp(u))$		$\log(-\log(1-p))$

For the generalized gamma distribution, u_p is computed numerically.

The standard errors of the quantile estimates are computed using the estimated covariance matrix of the parameter estimates and a Taylor series expansion of the quantile estimate. The standard error is computed as

$$\mathrm{STD} = \sqrt{\mathbf{z}'\mathbf{V}\mathbf{z}}$$

where $\mathbf{V}$ is the estimated covariance matrix of the parameter vector $(\boldsymbol{\beta}', \sigma, \delta)'$, and $\mathbf{z}$ is the vector

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \hat{u}_p \\ \hat{\sigma}\frac{\partial u_p}{\partial \delta} \end{bmatrix}$$

where δ is the vector of the shape parameters. Unless the NOLOG option is specified, this standard error estimate is converted into a standard error estimate for $\exp(y_p)$ as $\exp(\hat{y}_p)\mathrm{STD}$. It might be more desirable to compute confidence limits for the log response and convert them back to the original response variable than to use the standard error estimates for $\exp(y_p)$ directly. See Example 76.1 for a 90% confidence interval of the response constructed by exponentiating a confidence interval for the log response.

The variable CDF is computed as

$$\mathrm{CDF}_i = F(u_i)$$

where the residual is defined by

$$u_i = \left(\frac{y_i - \mathbf{x}_i'\mathbf{b}}{\hat{\sigma}}\right)$$

and F is the baseline cumulative distribution function. If the data are interval-censored, then the cumulative distribution function, $\mathrm{CDF}_i = F(u_i)$, is evaluated at the lower interval endpoint.

Confidence Intervals

Confidence intervals are computed for all model parameters and are reported in the "Analysis of Parameter Estimates" table. The confidence coefficient can be specified with the ALPHA=α MODEL statement option, resulting in a $(1-\alpha)\times 100\%$ two-sided confidence coefficient. The default confidence coefficient is 95%, corresponding to $\alpha = 0.05$.

Regression Parameters

A two-sided $(1-\alpha)\times 100\%$ confidence interval $[\beta_{iL},\ \ \beta_{iU}]$ for the regression parameter β_i is based on the asymptotic normality of the maximum likelihood estimator $\hat{\beta}_i$ and is computed by

$$\beta_{iL} = \hat{\beta}_i - z_{1-\alpha/2}(\mathrm{SE}_{\hat{\beta}_i})$$

$$\beta_{iU} = \hat{\beta}_i + z_{1-\alpha/2}(\mathrm{SE}_{\hat{\beta}_i})$$

where $\mathrm{SE}_{\hat{\beta}_i}$ is the estimated standard error of $\hat{\beta}_i$, and z_p is the $p\times 100$ percentile of the standard normal distribution.

Scale Parameter

A two-sided $(1-\alpha)\times 100\%$ confidence interval $[\sigma_L,\ \ \sigma_U]$ for the scale parameter σ in the location-scale model is based on the asymptotic normality of the logarithm of the maximum likelihood estimator $\log(\hat{\sigma})$, and is computed by

$$\sigma_L = \hat{\sigma}/\exp[z_{1-\alpha/2}(\mathrm{SE}_{\hat{\sigma}})/\hat{\sigma}]$$

$$\sigma_U = \hat{\sigma}\exp[z_{1-\alpha/2}(\mathrm{SE}_{\hat{\sigma}})/\hat{\sigma}]$$

See Meeker and Escobar (1998) for more information.

Weibull Scale and Shape Parameters

The Weibull distribution scale parameter η and shape parameter β are obtained by transforming the extreme-value location parameter μ and scale parameter σ:

$$\eta = \exp(\mu)$$

$$\beta = 1/\sigma$$

Consequently, two-sided $(1-\alpha)\times 100\%$ confidence intervals for the Weibull scale and shape parameters are computed as

$$[\eta_L,\ \ \eta_U] = [\exp(\mu_L),\ \ \exp(\mu_U)]$$

$$[\beta_L,\ \ \beta_U] = [1/\sigma_U,\ \ 1/\sigma_L]$$

Gamma Shape Parameter

A two-sided $(1-\alpha)\times 100\%$ confidence interval for the three-parameter gamma shape parameter δ is computed by

$$[\delta_L,\ \ \delta_U] = [\hat{\delta} - z_{1-\alpha/2}(\mathrm{SE}_{\hat{\delta}}),\ \ \hat{\delta} + z_{1-\alpha/2}(\mathrm{SE}_{\hat{\delta}})]$$

Fit Statistics

Suppose that the model contains p parameters and that n observations are used in model fitting. The fit criteria displayed by the LIFEREG procedure are calculated as follows:

- –2 log likelihood:

  $$-2\log(\mathrm{L})$$

  where L is the maximized likelihood for the model.

- Akaike's information criterion:

  $$\mathrm{AIC} = -2\log(\mathrm{L}) + 2p$$

- corrected Akaike's information criterion:

  $$\mathrm{AICC} = \mathrm{AIC} + \frac{2p(p+1)}{n-p-1}$$

- Bayesian information criterion:

  $$\mathrm{BIC} = -2\log(\mathrm{L}) + p\log(n)$$

If you specify the Weibull, exponential, lognormal, log-logistic, or gamma distribution, then maximum likelihood estimates of model parameters are computed by maximizing the log likelihood of the distribution of the logarithm of the response. This is equivalent to computing maximum likelihood parameter estimates based on the response on the original, rather than log, scale. If you specify the Weibull, exponential, lognormal, log-logistic, or gamma distribution, then fit statistics based on the maximized log likelihood $\log(\mathrm{L})$ of the log of the response are reported in the "Fit Statistics" table. Fit criteria computed in this way cannot be meaningfully compared with fit criteria that are based on the log likelihood of the unlogged response. If you specify the normal or logistic distribution, or if you specify the NOLOG option in the MODEL statement, then the fit criteria reported in the "Fit Statistics" table are based on the response on the original, rather than log, scale.

In addition to the "Fit Statistics" table described previously, if you specify the Weibull, exponential, lognormal, log-logistic, or gamma distribution, fit criteria that are based on the distribution of the response on the original scale, rather than the log of the response, are reported in the "Fit Statistics (Unlogged Response)" table.

When comparing models, you should compare fit criteria based on the log likelihood that is computed by using the response on the same scale, either always based on the log of the response or always based on the response on the original scale.

See Akaike (1981, 1979) for details of AIC and BIC. See Simonoff (2003) for a discussion of using AIC, AICC, and BIC in statistical modeling.

Probability Plotting

Probability plots are useful tools for the display and analysis of lifetime data. Probability plots use an inverse distribution scale so that a cumulative distribution function (CDF) plots as a straight line. A nonparametric estimate of the CDF of the lifetime data will plot approximately as a straight line, thus providing a visual assessment of goodness of fit.

You can use the PROBPLOT statement in PROC LIFEREG to create probability plots of data that are complete, right censored, interval censored, or a combination of censoring types (arbitrarily censored). A line representing the maximum likelihood fit from the MODEL statement and pointwise parametric confidence bands for the cumulative probabilities are also included in the plot.

A random variable Y belongs to a *location-scale* family of distributions if its CDF F is of the form

$$\Pr\{Y \le y\} = F(y) = G\left(\frac{y-\mu}{\sigma}\right)$$

where μ is the location parameter and σ is the scale parameter. Here, G is a CDF that cannot depend on any unknown parameters, and G is the CDF of Y if $\mu = 0$ and $\sigma = 1$. For example, if Y is a normal random variable with mean μ and standard deviation σ,

$$G(u) = \Phi(u) = \int_{-\infty}^{u} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{u^2}{2}\right) du$$

and

$$F(y) = \Phi\left(\frac{y-\mu}{\sigma}\right)$$

The normal, extreme-value, and logistic distributions are location-scale models. The three-parameter gamma distribution is a location-scale model if the shape parameter δ is fixed. If T has a lognormal, Weibull, or log-logistic distribution, then $\log(T)$ has a distribution that is a location-scale model. These distributions are said to be of type log-location-scale. Probability plots are constructed for lognormal, Weibull, and log-logistic distributions by using $\log(T)$ instead of T in the plots.

Let $y_{(1)} \le y_{(2)} \le \ldots \le y_{(n)}$ be ordered observations of a random sample with distribution function $F(y)$. A probability plot is a plot of the points $y_{(i)}$ against $m_i = G^{-1}(a_i)$, where $a_i = \hat{F}(y_i)$ is an estimate of the CDF $F(y_{(i)}) = G\left(\frac{y_{(i)}-\mu}{\sigma}\right)$. The nonparametric CDF estimates a_i are sometimes called *plotting positions*. The axis on which the points m_i are plotted is usually labeled with a probability scale (the scale of a_i).

If F is one of the location-scale distributions, then y is the lifetime; otherwise, the log of the lifetime is used to transform the distribution to a location-scale model.

If the data actually have the stated distribution, then $\hat{F} \approx F$,

$$m_i = G^{-1}(\hat{F}(y_i)) \approx G^{-1}\left(G\left(\frac{y_{(i)}-\mu}{\sigma}\right)\right) = \frac{y_{(i)}-\mu}{\sigma}$$

and points $(y_{(i)}, m_i)$ should fall approximately in a straight line.

There are several ways to compute the nonparametric CDF estimates used in probability plots from lifetime data. These are discussed in the next two sections.

Complete and Right-Censored Data

The censoring times must be taken into account when you compute plotting positions for right-censored data. The modified Kaplan-Meier method described in the following section is the default method for computing nonparametric CDF estimates for display on probability plots. See Abernethy (1996), Meeker and Escobar (1998), and Nelson (1982) for discussions of the methods described in the following sections.

Expected Ranks, Kaplan-Meier, and Modified Kaplan-Meier Methods

Let $y_{(1)} \leq y_{(2)} \leq \ldots \leq y_{(n)}$ be ordered observations of a random sample including failure times and censor times. Order the data in increasing order. Label all the data with reverse ranks r_i, with $r_1 = n, \ldots, r_n = 1$. For the lifetime (not censoring time) corresponding to reverse rank r_i, compute the survival function estimate

$$S_i = \left[\frac{r_i}{r_i + 1}\right] S_{i-1}$$

with $S_0 = 1$. The expected rank plotting position is computed as $a_i = 1 - S_i$. The option PPOS=EXPRANK specifies the expected rank plotting position.

For the Kaplan-Meier method,

$$S_i = \left[\frac{r_i - 1}{r_i}\right] S_{i-1}$$

The Kaplan-Meier plotting position is then computed as $a'_i = 1 - S_i$. The option PPOS=KM specifies the Kaplan-Meier plotting position.

For the modified Kaplan-Meier method, use

$$S'_i = \frac{S_i + S_{i-1}}{2}$$

where S_i is computed from the Kaplan-Meier formula with $S_0 = 1$. The plotting position is then computed as $a''_i = 1 - S'_i$. The option PPOS=MKM specifies the modified Kaplan-Meier plotting position. If the PPOS option is not specified, the modified Kaplan-Meier plotting position is used as the default method.

For complete samples, $a_i = i/(n+1)$ for the expected rank method, $a'_i = i/n$ for the Kaplan-Meier method, and $a''_i = (i - 0.5)/n$ for the modified Kaplan-Meier method. If the largest observation is a failure for the Kaplan-Meier estimator, then $F_n = 1$ and the point is not plotted.

Median Ranks

Let $y_{(1)} \leq y_{(2)} \leq \ldots \leq y_{(n)}$ be ordered observations of a random sample including failure times and censor times. A failure order number j_i is assigned to the ith failure: $j_i = j_{i-1} + \Delta$, where $j_0 = 0$. The increment Δ is initially 1 and is modified when a censoring time is encountered in the ordered sample. The new increment is computed as

$$\Delta = \frac{(n+1) - \text{ previous failure order number}}{1 + \text{ number of items beyond previous censored item}}$$

The plotting position is computed for the ith failure time as

$$a_i = \frac{j_i - 0.3}{n + 0.4}$$

For complete samples, the failure order number j_i is equal to i, the order of the failure in the sample. In this case, the preceding equation for a_i is an approximation of the median plotting position computed as the median of the ith-order statistic from the uniform distribution on (0, 1). In the censored case, j_i is not necessarily an integer, but the preceding equation still provides an approximation to the median plotting position. The PPOS=MEDRANK option specifies the median rank plotting position.

Arbitrarily Censored Data

The LIFEREG procedure can create probability plots for data that consist of combinations of exact, left-censored, right-censored, and interval-censored lifetimes—that is, arbitrarily censored data. The LIFEREG procedure uses an iterative algorithm developed by Turnbull (1976) to compute a nonparametric maximum likelihood estimate of the cumulative distribution function for the data. Since the technique is maximum likelihood, standard errors of the cumulative probability estimates are computed from the inverse of the associated Fisher information matrix. This algorithm is an example of the expectation-maximization (EM) algorithm. The default initial estimate assigns equal probabilities to each interval. You can specify different initial values with the PROBLIST= option. Convergence is determined if the change in the log likelihood between two successive iterations is less than delta, where the default value of delta is 10^{-8}. You can specify a different value for delta with the TOLLIKE= option. Iterations will be terminated if the algorithm does not converge after a fixed number of iterations. The default maximum number of iterations is 1000. Some data might require more iterations for convergence. You can specify the maximum allowed number of iterations with the MAXITEM= option in the PROBPLOT statement. The iteration history of the log likelihood is displayed if you specify the ITPRINTEM option. The iteration history of the estimated interval probabilities are also displayed if you specify both options ITPRINTEM and PRINTPROBS.

If an interval probability is smaller than a tolerance (10^{-6} by default) after convergence, the probability is set to zero, the interval probabilities are renormalized so that they add to one, and iterations are restarted. Usually the algorithm converges in just a few more iterations. You can change the default value of the tolerance with the TOLPROB= option. You can specify the NOPOLISH option to avoid setting small probabilities to zero and restarting the algorithm.

If you specify the ITPRINTEM option, a table summarizing the Turnbull estimate of the interval probabilities is displayed. The columns labeled "Reduced Gradient" and "Lagrange Multiplier" are used in checking final convergence of the maximum likelihood estimate. The Lagrange multipliers must all be greater than or equal to zero, or the solution is not maximum likelihood. See Gentleman and Geyer (1994) for more details of the convergence checking. Also see Meeker and Escobar (1998, Chapter 3) for more information.

See Example 76.6 for an illustration.

Nonparametric Confidence Intervals

You can use the PPOUT option in the PROBPLOT statement to create a table containing the nonparametric CDF estimates computed by the selected method, Kaplan-Meier CDF estimates, standard errors of the Kaplan-Meier estimator, and nonparametric confidence limits for the CDF. The confidence limits are either pointwise or simultaneous, depending on the value of the NPINTERVALS= option in the PROBPLOT statement. The method used in the LIFEREG procedure for computation of approximate pointwise and simultaneous confidence intervals for cumulative failure probabilities relies on the Kaplan-Meier estimator of the cumulative distribution function of failure time and approximate standard deviation of the Kaplan-Meier estimator. For the case of arbitrarily censored data, the Turnbull algorithm, discussed previously, provides an extension of the Kaplan-Meier estimator. Both the Kaplan-Meier and the Turnbull estimators provide an estimate of the standard error of the CDF estimator, $\mathrm{se}_{\hat{F}}$, that is used in computing confidence intervals.

Pointwise Confidence Intervals

Approximate $(1-\alpha)100\%$ pointwise confidence intervals are computed as in Meeker and Escobar (1998, Section 3.6) as

$$[F_L, \quad F_U] = \left[\frac{\hat{F}}{\hat{F} + (1-\hat{F})w}, \quad \frac{\hat{F}}{\hat{F} + (1-\hat{F})/w} \right]$$

where

$$w = \exp\left[\frac{z_{1-\alpha/2} \mathrm{se}_{\hat{F}}}{(\hat{F}(1-\hat{F}))} \right]$$

where z_p is the pth quantile of the standard normal distribution.

Simultaneous Confidence Intervals

Approximate $(1-\alpha)100\%$ simultaneous confidence bands valid over the lifetime interval (t_a, t_b) are computed as the "Equal Precision" case of Nair (1984) and Meeker and Escobar (1998, Section 3.8) as

$$[F_L, \quad F_U] = \left[\frac{\hat{F}}{\hat{F} + (1-\hat{F})w}, \quad \frac{\hat{F}}{\hat{F} + (1-\hat{F})/w} \right]$$

where

$$w = \exp\left[\frac{e_{a,b,1-\alpha/2} \mathrm{se}_{\hat{F}}}{(\hat{F}(1-\hat{F}))} \right]$$

where the factor $x = e_{a,b,1-\alpha/2}$ is the solution of

$$x \exp(-x^2/2) \log\left[\frac{(1-a)b}{(1-b)a} \right] / \sqrt{8\pi} = \alpha/2$$

The time interval (t_a, t_b) over which the bands are valid depends in a complicated way on the constants a and b defined in Nair (1984), $0 < a < b < 1$. The constants a and b are chosen by default so that the confidence bands are valid between the lowest and highest times corresponding to failures in the case of multiply censored data, or to the lowest and highest intervals for which probabilities are computed for arbitrarily censored data. You can optionally specify a and b directly with the NPINTERVALS=SIMULTANEOUS(a, b) option in the PROBPLOT statement.

Parametric Confidence Intervals

Pointwise parametric confidence bands are displayed in a probability plot, unless you specify the NOCONF option in the PROBPLOT statement. Two kinds of confidence intervals are available for display in a probability plot: confidence limits for the estimated cumulative distribution function (CDF) and confidence limits for estimated distribution percentiles.

Confidence Limits for the Estimated CDF

If the distribution is of type log-location-scale, let $y = \log(t)$ where t is the value of time at which the confidence limits are to be computed. If the distribution is of type location-scale, let y be the value at which you want to evaluate confidence limits for the estimated CDF $\hat{F}(y)$. Let

$$\hat{u} = \frac{\mathbf{y} - \mathbf{x}'\hat{\boldsymbol{\beta}}}{\hat{\sigma}}$$

where the column vector $\mathbf{x}$ of covariate values is determined by the rules summarized in the section "XDATA= Data Set" on page 5694. If an offset variable is specified, the mean of the offset variable values is included in $\mathbf{x}'\boldsymbol{\beta}$.

The CDF estimate is given by

$$\hat{F}(y) = G(\hat{u})$$

where G is the baseline distribution. The approximate standard error of $\hat{F}(y)$ is computed as in Meeker and Escobar (1998, Section 8.4.3) as

$$\mathrm{SE}_{\hat{F}} = \frac{g(\hat{u})}{\hat{\sigma}}\left[\mathrm{Var}(\mathbf{x}'\hat{\boldsymbol{\beta}}) + 2\hat{u}\mathrm{Cov}(\mathbf{x}'\hat{\boldsymbol{\beta}}, \hat{\sigma}) + \hat{u}^2\mathrm{Var}(\hat{\sigma})\right]^{\frac{1}{2}}$$

where g is the probability density function corresponding to G. Two-sided $(1-\alpha) \times 100\%$ confidence limits are given by

$$[F_L, \quad F_U] = \left[\frac{\hat{F}}{\hat{F} + (1-\hat{F}) \times w}, \quad \frac{\hat{F}}{\hat{F} + (1-\hat{F})/w}\right]$$

where

$$w = \exp\left[\frac{z_{1-\alpha/2}\mathrm{SE}_{\hat{F}}}{\hat{F}(1-\hat{F})}\right]$$

and z_p is the $p \times 100$ percentile of the standard normal distribution. The quantities $\mathrm{Var}(\mathbf{x}'\hat{\boldsymbol{\beta}})$, $\mathrm{Cov}(\mathbf{x}'\hat{\boldsymbol{\beta}}, \hat{\sigma})$, and $\mathrm{Var}(\hat{\sigma})$ are computed based on the covariance matrix of the estimated parameter vector $(\hat{\boldsymbol{\beta}}, \hat{\sigma})$.

Confidence Limits for Percentiles

If the HCL option is specified in the PROBPLOT statement, confidence limits based on estimated distribution percentiles instead of the default CDF limits are displayed in the probability plot.

For location-scale distributions, the estimated $p \times 100$ percentile of the distribution F is given by

$$y_p = \mathbf{x}'\hat{\boldsymbol{\beta}} + G^{-1}(p)\hat{\sigma}$$

where G is the baseline distribution and the column vector $\mathbf{x}$ of covariate values is determined by the rules summarized in the section "XDATA= Data Set" on page 5694. The standard error of y_p is estimated by

$\mathrm{SE}_y = z'\Sigma z$ where $\mathbf{z} = (\mathbf{x}', G^{-1}(p))'$ and Σ is the covariance matrix of the parameter estimates $(\hat{\boldsymbol{\beta}}', \hat{\sigma})'$. Two-sided $(1-\alpha) \times 100\%$ confidence limits for y_p are given by

$$[y_L, \quad y_U] = [y_p - z_{1-\alpha/2}\mathrm{SE}_y, \quad y_p + z_{1-\alpha/2}\mathrm{SE}_y]$$

For distributions of type log-location-scale, the confidence limits are computed as

$$[t_L = \exp(y_L), \quad t_U = \exp(y_U)]$$

For example, if T has the Weibull distribution, G is the standardized extreme value distribution, $[y_L, \quad y_U]$ are confidence limits for the $p \times 100$ percentile of the extreme value distribution for $\log(T)$, and $[t_L = \exp(y_L), \quad t_U = \exp(y_U)]$ are confidence limits for the $p \times 100$ percentile of the Weibull distribution for T.

INEST= Data Set

If specified, the INEST= data set specifies initial estimates for all the parameters in the model. The INEST= data set must contain the intercept variable (named Intercept) and all independent variables in the MODEL statement.

If BY processing is used, the INEST= data set should also include the BY variables, and there must be at least one observation for each BY group. If there is more than one observation in one BY group, the first observation read is used for that BY group.

If the INEST= data set also contains the _TYPE_ variable, only observations with _TYPE_ value 'PARMS' are used as starting values. Combining the INEST= data set and the MAXITER= option in the MODEL statement, partial scoring can be done, such as predicting on a validation data set by using the model built from a training data set.

You can specify starting values for the iterative algorithm in the INEST= data set. This data set overwrites the INITIAL= option in the MODEL statement, which is a little difficult to use for models including multilevel interaction effects. The INEST= data set has the same structure as the OUTEST= data set but is not required to have all the variables or observations that appear in the OUTEST= data set. One simple use of the INEST= option is passing the previous OUTEST= data set directly to the next model as an INEST= data set, assuming that the two models have the same parameterization. See Example 76.3 for an illustration.

OUTEST= Data Set

The OUTEST= data set contains parameter estimates and the log likelihood for the model. You can specify a label in the MODEL statement to distinguish between the estimates for different models fit with the LIFEREG procedure. If the COVOUT option is specified, the OUTEST= data set also contains the estimated covariance matrix of the parameter estimates. Note that, if the LIFEREG procedure does not converge, the parameter estimates are set to missing in the OUTEST data set.

The OUTEST= data set contains all variables specified in the MODEL statement and the BY statement. One observation consists of parameter values for the model with the dependent variable having the value –1. If the COVOUT option is specified, there are additional observations containing the rows of the estimated covariance matrix. For these observations, the dependent variable contains the parameter estimate for the corresponding row variable. The following variables are also added to the data set:

MODEL	a character variable containing the label of the MODEL statement, if present. Otherwise, the variable's value is blank.
NAME	a character variable containing the name of the dependent variable for the parameter estimates observations or the name of the row for the covariance matrix estimates
TYPE	a character variable containing the type of the observation, either PARMS for parameter estimates or COV for covariance estimates
DIST	a character variable containing the name of the distribution modeled
LNLIKE	a numeric variable containing the last computed value of the log likelihood
INTERCEPT	a numeric variable containing the intercept parameter estimates and covariances
SCALE	a numeric variable containing the scale parameter estimates and covariances
SHAPE1	a numeric variable containing the first shape parameter estimates and covariances if the specified distribution has additional shape parameters

Any BY variables specified are also added to the OUTEST= data set.

XDATA= Data Set

The XDATA= data set is used for plotting the predicted probability when there are covariates specified in a MODEL statement and a probability plot is specified with a PROBPLOT statement. See Example 76.4 for an illustration.

The XDATA= data set is an input SAS data set that contains values for all the independent variables in the MODEL statement and variables in the CLASS statement. The XDATA= data set has the same structure as the DATA= data set but is not required to have all the variables or observations that appear in the DATA= data set.

The XDATA= data set must contain all the independent variables in the MODEL statement and variables in the CLASS statement. Even though variables in the CLASS statement might not be used, valid values are required for these variables in the XDATA= data set. Missing values are not allowed. Missing values are not allowed in the XDATA= data set for any of the independent variables, either. Missing values are allowed for the dependent variables and other variables if they are included in the XDATA= data set.

If BY processing is used, the XDATA= data set should also include the BY variables, and there must be at least one valid observation for each BY group. If there is more than one valid observation in a BY group, the last one read is used for that BY group.

If there is no XDATA= data set in the PROC LIFEREG statement, by default, the LIFEREG procedure will use the overall mean for effects containing a continuous variable (or variables) and the highest level of a single classification variable as reference level. The rules are summarized as follows:

- If the effect contains a continuous variable (or variables), the overall mean of this effect (not the variables) is used.
- If the effect is a single classification variable, the highest level of the variable is used.

Computational Resources

Let p be the number of parameters estimated in the model. The minimum working space (in bytes) needed is

$$16p^2 + 100p$$

However, if sufficient space is available, the input data set is also kept in memory; otherwise, the input data set is reread for each evaluation of the likelihood function and its derivatives, with the resulting execution time of the procedure substantially increased.

Let n be the number of observations used in the model estimation. Each evaluation of the likelihood function and its first and second derivatives requires $O(np^2)$ multiplications and additions, n individual function evaluations for the log density or log distribution function, and n evaluations of the first and second derivatives of the function. The calculation of each updating step from the gradient and Hessian requires $O(p^3)$ multiplications and additions. The $O(v)$ notation means that, for large values of the argument, v, $O(v)$ is approximately a constant times v.

Bayesian Analysis

Gibbs Sampling

This section provides details about Bayesian analysis by Gibbs sampling in the location-scale models for survival data available in PROC LIFEREG. See the section "Gibbs Sampler" on page 157 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a general discussion of Gibbs sampling. PROC LIFEREG fits parametric location-scale survival models. That is, the probability density of the response Y can expressed in the general form

$$f(y) = g\left(\frac{y-\mu}{\sigma}\right)$$

where $Y = \log(T)$ for lifetimes T. The function g determines the specific distribution. The location parameter μ_i is modeled through regression parameters as $\mu_i = \mathbf{x}_i'\boldsymbol{\beta}$. The LIFEREG procedure can provide Bayesian estimates of the regression parameters and σ. The OUTPUT and PROBPLOT statements, if specified, are ignored. The PLOTS=PROBPLOT option in the PROC LIFEREG statement and the CORRB and COVB options in the MODEL statement are also ignored.

For the Weibull distribution, you can specify that Gibbs sampling be performed on the Weibull shape parameter $\beta = \sigma^{-1}$ instead of the scale parameter σ by specifying a prior distribution for the shape parameter with the WEIBULLSHAPEPRIOR= option. In addition, if there are no covariates in the model, you can specify Gibbs sampling on the Weibull scale parameter $\alpha = \exp(\mu)$, where μ is the intercept term, with the WEIBULLSCALEPRIOR= option.

In the case of the exponential distribution with no covariates, you can specify Gibbs sampling on the exponential scale parameter $\alpha = \exp(\mu)$, where μ is the intercept term, with the EXPSCALEPRIOR= option.

Let $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)'$ be the parameter vector. For location-scale models, the θ_i's are the regression coefficients β_i's and the scale parameter σ. In the case of the three-parameter gamma distribution, there is an

additional gamma shape parameter τ. Let $L(D|\boldsymbol{\theta})$ be the likelihood function, where D is the observed data. Let $\pi(\boldsymbol{\theta})$ be the prior distribution. The full conditional distribution of $[\theta_i|\theta_j, i \neq j]$ is proportional to the joint distribution; that is,

$$\pi(\theta_i|\theta_j, i \neq j, D) \propto L(D|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

For instance, the one-dimensional conditional distribution of θ_1 given $\theta_j = \theta_j^*, 2 \leq j \leq k$, is computed as

$$\pi(\theta_1|\theta_j = \theta_j^*, 2 \leq j \leq k, D) = L(D|(\boldsymbol{\theta} = (\theta_1, \theta_2^*, \ldots, \theta_k^*)')p(\boldsymbol{\theta} = (\theta_1, \theta_2^*, \ldots, \theta_k^*)')$$

Suppose you have a set of arbitrary starting values $\{\theta_1^{(0)}, \ldots, \theta_k^{(0)}\}$. Using the ARMS (adaptive rejection Metropolis sampling) algorithm of Gilks and Wild (1992) and Gilks, Best, and Tan (1995), you can do the following:

draw $\theta_1^{(1)}$ from $[\theta_1|\theta_2^{(0)}, \ldots, \theta_k^{(0)}]$

draw $\theta_2^{(1)}$ from $[\theta_2|\theta_1^{(1)}, \theta_3^{(0)}, \ldots, \theta_k^{(0)}]$

...

draw $\theta_k^{(1)}$ from $[\theta_k|\theta_1^{(1)}, \ldots, \theta_{k-1}^{(1)}]$

This completes one iteration of the Gibbs sampler. After one iteration, you have $\{\theta_1^{(1)}, \ldots, \theta_k^{(1)}\}$. After n iterations, you have $\{\theta_1^{(n)}, \ldots, \theta_k^{(n)}\}$. PROC LIFEREG implements the ARMS algorithm based on a program provided by Gilks (2003) to draw a sample from a full conditional distribution. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for information about assessing the convergence of the chain of posterior samples.

You can output these posterior samples into a SAS data set. The following *option* in the BAYES statement outputs the posterior samples into the SAS data set Post: OUTPOST=Post. The data set also includes the variables LogPost and LogLike, which represent the log of the posterior distribution and the log of the likelihood, respectively.

Priors for Model Parameters

The model parameters are the regression coefficients and the dispersion parameter (or the precision or scale), if the model has one. The priors for the dispersion parameter and the priors for the regression coefficients are assumed to be independent, while you can have a joint multivariate normal prior for the regression coefficients.

Scale and Shape Parameters

Gamma Prior The gamma distribution $G(a, b)$ has a PDF

$$f_{a,b}(u) = \frac{b(bu)^{a-1}\mathrm{e}^{-bu}}{\Gamma(a)}, \qquad u > 0$$

where a is the shape parameter and b is the inverse-scale parameter. The mean is $\frac{a}{b}$ and the variance is $\frac{a}{b^2}$.

Improper Prior The joint prior density is given by

$$p(u) \propto u^{-1}, \qquad u > 0$$

Regression Coefficients

Let $\boldsymbol{\beta}$ be the regression coefficients.

Normal Prior Assume $\boldsymbol{\beta}$ has a multivariate normal prior with mean vector $\boldsymbol{\beta}_0$ and covariance matrix $\boldsymbol{\Sigma}_0$. The joint prior density is given by

$$p(\boldsymbol{\beta}) \propto \mathrm{e}^{-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\beta}_0)'\boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\beta}_0)}$$

Uniform Prior The joint prior density is given by

$$p(\boldsymbol{\beta}) \propto 1$$

Posterior Distribution

Denote the observed data by D.

The posterior distribution is

$$\pi(\boldsymbol{\theta}|D) \propto L_P(D|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

where $L_P(D|\boldsymbol{\theta})$ is the likelihood function with regression coefficients and any additional parameters, such as scale or shape, $\boldsymbol{\theta}$ as parameters; and $p(\boldsymbol{\theta})$ is the joint prior distribution of the parameters.

Deviance Information Criterion

Let $\boldsymbol{\theta}_i$ be the model parameters at iteration i of the Gibbs sampler, and let $\mathrm{LL}(\boldsymbol{\theta}_i)$ be the corresponding model log likelihood. PROC LIFEREG computes the following fit statistics defined by Spiegelhalter et al. (2002):

- effective number of parameters:

$$p_D = \overline{\mathrm{LL}(\boldsymbol{\theta})} - \mathrm{LL}(\bar{\boldsymbol{\theta}})$$

- deviance information criterion (DIC):

$$\mathrm{DIC} = \overline{\mathrm{LL}(\boldsymbol{\theta})} + p_D$$

where

$$\overline{\mathrm{LL}(\boldsymbol{\theta})} = \frac{1}{n}\sum_{i=1}^{n}\mathrm{LL}(\boldsymbol{\theta}_i)$$

$$\bar{\boldsymbol{\theta}} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{\theta}_i$$

and n is the number of Gibbs samples.

Starting Values of the Markov Chains

When the BAYES statement is specified, PROC LIFEREG generates one Markov chain containing the approximate posterior samples of the model parameters. Additional chains are produced when the Gelman-Rubin diagnostics are requested. Starting values (or initial values) can be specified in the INITIAL= data set in the BAYES statement. If INITIAL= option is not specified, PROC LIFEREG picks its own initial values for the chains.

Denote $[x]$ as the integral value of x. Denote $\hat{s}(X)$ as the estimated standard error of the estimator X.

Regression Coefficients and Gamma Shape Parameter

For the first chain that the summary statistics and regression diagnostics are based on, the default initial values are estimates of the mode of the posterior distribution. If the INITIALMLE option is specified, the initial values are the maximum likelihood estimates; that is,

$$\beta_i^{(0)} = \hat{\beta}_i$$

Initial values for the rth chain ($r \geq 2$) are given by

$$\beta_i^{(0)} = \hat{\beta}_i \pm \left(2 + \left[\frac{r}{2}\right]\right)\hat{s}(\hat{\beta}_i)$$

with the plus sign for odd r and minus sign for even r.

Scale, Exponential Scale, Weibull Scale, or Weibull Shape Parameter λ

Let λ be the parameter sampled.

For the first chain that the summary statistics and diagnostics are based on, the initial values are estimates of the mode of the posterior distribution; or the maximum likelihood estimates if the INITIALMLE option is specified; that is,

$$\lambda^{(0)} = \hat{\lambda}$$

The initial values of the rth chain ($r \geq 2$) are given by

$$\lambda^{(0)} = \hat{\lambda}\mathrm{e}^{\pm\left([\frac{r}{2}]+2\right)\hat{s}(\hat{\lambda})}$$

with the plus sign for odd r and minus sign for even r.

OUTPOST= Output Data Set

The OUTPOST= data set contains the generated posterior samples. There are 2+n variables, where n is the number of model parameters. The variable Iteration represents the iteration number and the variable LogPost contains the log posterior likelihood values. The other n variables represent the draws of the Markov chain for the model parameters.

Displayed Output for Classical Analysis

For each model, PROC LIFEREG displays the following.

Model Information

The "Model Information" table displays the two-level name of the input data set, the distribution name, and the name and label of the dependent variable; the name and label of the censor indicator variable, for right-censored data; if you specify the WEIGHT statement, the name and label of the weight variable; and the maximum value of the log likelihood.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set, and the number of observations used in the analysis.

Class Level Information

The "Class Level Information" table displays the levels of classification variables if you specify a CLASS statement.

Fit Statistics

The "Fit Statistics" table displays the negative of twice the log likelihood, Akaike's information criterion (AIC), the corrected Akaike's information criterion (AICC), and the Bayesian information criterion (BIC). If the specified distribution is Weibull, lognormal, log-logistic, or gamma, the fit criteria are based on the log likelihood for the log of the response, rather than for the response on the original scale.

Fit Statistics (Unlogged Response)

If the specified distribution is Weibull, lognormal, log-logistic, or gamma, the "Fit Statistics (Unlogged Response)" table displays fit criteria that are based on the log likelihood for the response on the original, rather than log, scale. The negative of twice the log likelihood, Akaike's information criterion (AIC), the corrected Akaike's information criterion (AICC), and the Bayesian information criterion (BIC) are displayed.

Type III Analysis of Effects

The "Type III Analysis of Effects" table displays, for each effect in the model, the effect name, the degrees of freedom associated with the type III contrast for the effect, the chi-square statistic for the contrast, and the p-value for the statistic.

Analysis of Maximum Likelihood Parameter Estimates

The "Analysis of Maximum Likelihood Parameter Estimates" table displays the parameter name, the degrees of freedom for each parameter, the maximum likelihood estimate of each parameter, the estimated standard error of the parameter estimator, confidence limits for each parameter, a chi-square statistic for testing whether the parameter is zero, and the associated p-value for the statistic.

Lagrange Multiplier Statistics

If there are constrained parameters in the model, such as the scale or intercept, then the "Lagrange Multiplier Statistics" table displays a Lagrange multiplier test for the constraint.

Displayed Output for Bayesian Analysis

If a Bayesian analysis is requested with a BAYES statement, the displayed output includes the following.

Model Information

The "Model Information" table displays the two-level name of the input data set, the number of burn-in iterations, the number of iterations after the burn-in, the number of thinning iterations, the distribution name, and the name and label of the dependent variable; the name and label of the censor indicator variable, for right-censored data; if you specify the WEIGHT statement, the name and label of the weight variable; and the maximum value of the log likelihood.

Class Level Information

The "Class Level Information" table displays the levels of classification variables if you specify a CLASS statement.

Maximum Likelihood Estimates

The "Analysis of Maximum Likelihood Parameter Estimates" table displays the maximum likelihood estimate of each parameter, the estimated standard error of the parameter estimator, and confidence limits for each parameter.

Coefficient Prior

The "Coefficient Prior" table displays the prior distribution of the regression coefficients.

Independent Prior Distributions for Model Parameters

The "Independent Prior Distributions for Model Parameters" table displays the prior distributions of additional model parameters (scale, exponential scale, Weibull scale, Weibull shape, gamma shape).

Initial Values and Seeds

The "Initial Values and Seeds" table displays the initial values and random number generator seeds for the Gibbs chains.

Fit Statistics

The "Fit Statistics" table displays the deviance information criterion (DIC) and the effective number of parameters.

Posterior Summaries

The "Posterior Summaries" table contains the size of the sample, the mean, the standard deviation, and the quartiles for each model parameter.

Posterior Intervals

The "Posterior Intervals" table contains the HPD intervals and the credible intervals for each model parameter.

Correlation Matrix of the Posterior Samples

The "Correlation Matrix of the Posterior Samples" table is produced if you include the CORR suboption in the SUMMARY= option in the BAYES statement. This table displays the sample correlation of the posterior samples.

Covariance Matrix of the Posterior Samples

The "Covariance Matrix of the Posterior Samples" table is produced if you include the COV suboption in the SUMMARY= option in the BAYES statement. This table displays the sample covariance of the posterior samples.

Autocorrelations of the Posterior Samples

The "Autocorrelations of the Posterior Samples" table displays the lag1, lag5, lag10, and lag50 autocorrelations for each parameter.

Gelman and Rubin Diagnostics

The "Gelman and Rubin Diagnostics" table is produced if you include the GELMAN suboption in the DIAGNOSTIC= option in the BAYES statement. This table displays the estimate of the potential scale reduction factor and its 97.5% upper confidence limit for each parameter.

Geweke Diagnostics

The "Geweke Diagnostics" table displays the Geweke statistic and its p-value for each parameter.

Raftery and Lewis Diagnostics

The "Raftery Diagnostics" tables is produced if you include the RAFTERY suboption in the DIAGNOSTIC= option in the BAYES statement. This table displays the Raftery and Lewis diagnostics for each variable.

Heidelberger and Welch Diagnostics

The "Heidelberger and Welch Diagnostics" table is displayed if you include the HEIDELBERGER suboption in the DIAGNOSTIC= option in the BAYES statement. This table shows the results of a stationary test and a halfwidth test for each parameter.

Effective Sample Size

The "Effective Sample Size" table displays, for each parameter, the effective sample size, the correlation time, and the efficiency.

Monte Carlo Standard Errors

The "Monte Carlo Standard Errors" table displays, for each parameter, the Monte Carlo standard error, the posterior sample standard deviation, and the ratio of the two.

Plot Options Superseded by ODS Graphics

You can select one of the following two types of graphics in PROC LIFEREG: ODS and traditional. ODS Graphics is the preferred method of creating graphs, superseding traditional graphs.

When ODS Graphics is enabled, you can use the PLOTS= option in the PROC LIFEREG statement to create plots by using ODS Graphics. For more information about ODS Graphics options, see the PLOTS= option. If ODS Graphics is not enabled, then traditional graphics are produced.

You can specify the following option in the PROC LIFEREG statement:

GOUT=*graphics-catalog*
specifies a graphics catalog in which to save traditional graphical output.

The following *options* control the appearance of the box in the INSET statement when you use traditional graphics. These *options* are not available if ODS Graphics is enabled. Table 76.12 summarizes the *options* available in the INSET statement.

Table 76.12 INSET Statement Options

Option	Description
CFILL=	Specifies the color for the filling box
CFILLH=	Specifies the color for the filling box header
CFRAME=	Specifies the color for the frame
CHEADER=	Specifies the color for text in the header
CTEXT=	Specifies the color for the text
FONT=	Specifies the software font for the text
HEADER=	Specifies the text for the header or box title
HEIGHT=	Specifies the height of the text
NOFRAME	Omits the frame around the box
POS=	Determines the position of the inset
REFPOINT=	Specifies the reference point for an inset

All *options* are specified after the slash (/) in the INSET statement.

CFILL=*color*
specifies the color for the filling box.

CFILLH=*color*
specifies the color for the filling box header.

CFRAME=*color*
specifies the color for the frame.

CHEADER=*color*
specifies the color for text in the header.

CTEXT=*color*
specifies the color for the text.

FONT=*font*
specifies the software font for the text.

HEADER=*'quoted string'*
specifies the text for the header or box title.

HEIGHT=*value*
specifies the height of the text.

NOFRAME
omits the frame around the box.

POS=*value* < **DATA** | **PERCENT** >
determines the position of the inset. The *value* can be a compass point (N, NE, E, SE, S, SW, W, NW) or a pair of coordinates (x, y) enclosed in parentheses. The coordinates can be specified in screen percentage units or axis data units. The default is screen percentage units.

REFPOINT=*name*
specifies the reference point for an inset that is positioned by a pair of coordinates with the POS= option. You use the REFPOINT= option in conjunction with the POS= coordinates. The REFPOINT= option specifies which corner of the inset frame you have specified with coordinates (x, y), and it can take the value of BR (bottom right), BL (bottom left), TR (top right), or TL (top left). The default is REFPOINT=BL. If the inset position is specified as a compass point, then the REFPOINT= option is ignored.

Table 76.13 summarizes the *options* available in the PROBPLOT statement for traditional graphics.

Table 76.13 PROBPLOT Statement Options

Option	Description
Traditional Graphics	
ANNOTATE=	Specifies an Annotate data set
CAXIS=	Specifies the color for the axes and tick marks
CCENSOR=	Specifies the color for filling the censor plot area

Table 76.13 *continued*

Option	Description
CENBIN	Plots censored data as frequency counts
CENCOLOR=	Specifies the color for the censor symbol
CENSYMBOL=	Specifies symbols for censored values
CFIT=	Specifies the color for the fitted probability line and confidence curves
CFRAME=	Specifies the color for the area enclosed by the axes and frame
CGRID=	Specifies the color for grid lines
CHREF=	Specifies the color for lines requested by the HREF= option
CTEXT=	Specifies the color for tick mark values and axis labels
CVREF=	Specifies the color for lines requested by the VREF= option
DESCRIPTION=	Specifies a description that appears in the PROC GREPLAY master menu
FONT=	Specifies a software font for reference line and axis labels
HCL	Computes and draws confidence limits
HEIGHT=	Specifies the height of text used outside framed areas
HLOWER=	Specifies the lower limit on the lifetime axis scale
HOFFSET=	Specifies the offset for the horizontal axis
HREF	Draws reference lines perpendicular to the horizontal axis
HREFLABELS=	Specifies labels for the lines requested by the HREF= option
HREFLABPOS=	Specifies the vertical position of labels for HREF= lines
HUPPER=	Specifies *value* as the upper lifetime axis tick mark
INBORDER	Requests a border around probability plots
INTERTILE=	Specifies the distance between tiles
ITPRINTEM	Displays the iteration history for the Turnbull algorithm
JITTER=	Specifies the amount to jitter overlaid plot symbols, in units of symbol width
LFIT=	Specifies a line style for fitted curves and confidence limits
LGRID=	Specifies a line style for all grid lines
LHREF=	Specifies the line type for lines requested by the HREF= option
LVREF=	Specifies the line type for lines requested by the VREF= option
MAXITEM=	Specifies the maximum number of iterations for the Turnbull algorithm
NAME=	Specifies a name for the plot
NOCENPLOT	Suppresses the plotting of censored data points
NOCONF	Suppresses the default confidence bands
NODATA	Suppresses plotting of the estimated empirical probability plot
NOFIT	Suppresses the fitted probability (percentile) line and confidence bands
NOFRAME	Suppresses the frame around plotting areas
NOGRID	Suppresses grid lines
NOHLABEL	Suppresses horizontal label
NOHTICK	Suppresses horizontal tick marks
NOPOLISH	Suppresses setting small interval probabilities to zero
NOVLABEL	Suppresses vertical labels
NOVTICK	Suppresses vertical tick marks
NPINTERVALS=	Displays one of the two kinds of confidence limit
PCTLIST=	Specifies the list of percentages for which to compute percentile estimates
PLOWER=	Specifies the lower limit on the probability axis scale
PPOS=	Specifies the plotting position type

Table 76.13 *continued*

Option	Description
PPOUT	Displays a table of the cumulative probabilities
PRINTPROBS	Displays intervals and associated probabilities for the Turnbull algorithm
PROBLIST=	Specifies the list of initial values for the Turnbull algorithm
PUPPER=	Specifies the upper limit on the probability axis scale
ROTATE	Requests probability plots with probability scale on the horizontal axis
SQUARE	Makes the layout of the probability plots square
TOLLIKE=	Specifies the criterion for convergence in the Turnbull algorithm
TOLPROB=	Specifies the criterion for setting the interval probability to zero in the Turnbull algorithm
VAXISLABEL=	Specifies a label for the vertical axis
VREF	Draws reference lines perpendicular to the vertical axis
VREFLABELS=	Specifies labels for the lines requested by the VREF= option
VREFLABPOS=	Specifies the horizontal position of labels for VREF= lines
WAXIS=	Specifies line thickness for axes and frame
WFIT=	Specifies line thickness for fitted curves
WGRID=	Specifies line thickness for grids
WREFL=	Specifies line thickness for reference lines

The following *options* are available in the PROBPLOT statement if you use traditional graphics—that is, if ODS Graphics is not enabled.

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an Annotate data set, as described in *SAS/GRAPH: Reference*, that enables you to add features to the probability plot. The data set you specify with the ANNOTATE= option in the PROBPLOT statement provides the Annotate data set for all plots created by the statement.

CAXIS=*color*

CAXES=*color*

specifies the color for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CCENSOR=*color*

specifies the color for filling the censor plot area. The default is the first color in the device color list.

CENBIN

plots censored data as frequency counts (rounding for noninteger frequency) rather than as individual points.

CENCOLOR=*color*

specifies the color for the censor symbol. The default is the first color in the device color list.

CENSYMBOL=*symbol* | (*symbol list*)
specifies symbols for censored values. The *symbol* is one of the symbol names (plus, star, square, diamond, triangle, hash, paw, point, dot, and circle) or a letter (A–Z). If you do not specify the CENSYMBOL= option, the symbol used for censored values is the same as for failures.

CFIT=*color*
specifies the color for the fitted probability line and confidence curves. The default is the first color in the device color list.

CFRAME=*color*

CFR=*color*
specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*
specifies the color for grid lines. The default is the first color in the device color list.

CHREF=*color*

CH=*color*
specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*
specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*

CV=*color*
specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string*'

DES='*string*'
specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*
specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HCL
computes and draws confidence limits for the predicted probabilities based on distribution percentiles instead of the default CDF limits. See the section "Confidence Limits for Percentiles" on page 5692 for details of the computation.

HEIGHT=*value*
specifies the height of text outside framed areas. The default value is 3.846 (in percentage).

HLOWER=*value*
specifies the lower limit on the lifetime axis scale. The HLOWER= option specifies *value* as the lower lifetime axis tick mark. The tick mark interval and the upper axis limit are determined automatically.

HOFFSET=*value*
specifies the offset for the horizontal axis. The default value is 1.

HUPPER=*value*
specifies *value* as the upper lifetime axis tick mark. The tick mark interval and the lower axis limit are determined automatically.

HREF < (INTERSECT) > =*value-list*
requests that reference lines perpendicular to the horizontal axis be drawn at horizontal axis values in the *value-list*. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified with the HREFLABELS= option, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS=*'label1'* **...** *'labeln'*

HREFLABEL=*'label1'* **...** *'labeln'*

HREFLAB=*'label1'* **...** *'labeln'*
specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*
specifies the vertical position of labels for HREF= lines. The following table shows the valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Top
2	Staggered from top
3	Bottom
4	Staggered from bottom
5	Alternating from top
6	Alternating from bottom

INBORDER
requests a border around probability plots.

INTERTILE=*value*
specifies the distance between tiles.

ITPRINTEM
displays the iteration history for the Turnbull algorithm.

JITTER=*value*
specifies the amount to jitter overlaid plot symbols, in units of symbol width.

LFIT=*linetype*
specifies a line style for fitted curves and confidence limits. By default, fitted curves are drawn by connecting solid lines (*linetype = 1*), and confidence limits are drawn by connecting dashed lines (*linetype = 3*).

LGRID=*linetype*
specifies a line style for all grid lines; *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

MAXITEM=*n1 <,n2>*
specifies the maximum number of iterations for the Turnbull algorithm. Iteration history will be displayed in increments of *n2* if requested with the ITPRINTEM option. See the section "Arbitrarily Censored Data" on page 5690 for details.

NAME=*'string'*
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'LIFEREG'.

NOCENPLOT
suppresses the plotting of censored data points.

NOCONF
suppresses the default confidence bands on the probability plot.

NODATA
suppresses plotting of the estimated empirical probability plot.

NOFIT
suppresses the fitted probability (percentile) line and confidence bands.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOPOLISH
suppresses setting small interval probabilities to zero in the Turnbull algorithm.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

NPINTERVALS=*interval-type*
specifies one of the two kinds of confidence limits for the estimated cumulative probabilities, pointwise (NPINTERVALS=POINT) or simultaneous (NPINTERVALS=SIMUL), requested by the PPOUT option to be displayed in the tabular output.

PCTLIST=*value-list*
specifies the list of percentages for which to compute percentile estimates; *value-list* must be a list of values separated by blanks or commas. Each value in the list must be between 0 and 100.

PLOWER=*value*
specifies the lower limit on the probability axis scale. The PLOWER= option specifies *value* as the lower probability axis tick mark. The tick mark interval and the upper axis limit are determined automatically.

PPOS=*character-list*
specifies the plotting position type. See the section "Probability Plotting" on page 5688 for details.

PPOS=	Method
EXPRANK	Expected ranks
MEDRANK	Median ranks
MEDRANK1	Median ranks (exact formula)
KM	Kaplan-Meier
MKM	Modified Kaplan-Meier (default)

PPOUT
specifies that a table of the cumulative probabilities plotted on the probability plot be displayed. Kaplan-Meier estimates of the cumulative probabilities are also displayed, along with standard errors and confidence limits. The confidence limits can be pointwise or simultaneous, as specified by the NPINTERVALS= option.

PRINTPROBS
displays intervals and associated probabilities for the Turnbull algorithm.

PROBLIST=*value-list*
specifies the list of initial values for the Turnbull algorithm.

PUPPER=*value*
specifies the upper limit on the probability axis scale. The PUPPER= option specifies *value* as the upper probability axis tick mark. The tick mark interval and the lower axis limit are determined automatically.

ROTATE
requests probability plots with probability scale on the horizontal axis.

SQUARE
makes the layout of the probability plots square.

TOLLIKE=*value*
specifies the criterion for convergence in the Turnbull algorithm.

TOLPROB=*value*
specifies the criterion for setting the interval probability to zero in the Turnbull algorithm.

VAXISLABEL=‘*string***’**
specifies a label for the vertical axis.

VREF< (INTERSECT) >=*value-list*
requests that reference lines perpendicular to the vertical axis be drawn at vertical axis values in the *value-list*. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified with the VREFLABELS= option, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS=*'label1' . . . 'labeln'*
VREFLABEL=*'label1' . . . 'labeln'*
VREFLAB=*'label1' . . . 'labeln'*
specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*
specifies the horizontal position of labels for VREF= lines. The valid values for *n* and the corresponding label placements are shown in the following table.

n	**Label Placement**
1	Left
2	Right

WAXIS=*n*
specifies line thickness for axes and frame. The default value is 1.

WFIT=*n*
specifies line thickness for fitted curves. The default value is 1.

WGRID=*n*
specifies line thickness for grids. The default value is 1.

WREFL=*n*
specifies line thickness for reference lines. The default value is 1.

ODS Table Names

PROC LIFEREG assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed separately in Table 76.14 for a maximum likelihood analysis and in Table 76.15 for a Bayesian analysis. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 76.14 ODS Tables Produced in PROC LIFEREG for a Classical Analysis

ODS Table Name	Description	Statement	Option
ClassLevels	Classification variable levels	CLASS	Default*
ConvergenceStatus	Convergence status	MODEL	Default
CorrB	Parameter estimate correlation matrix	MODEL	CORRB
CovB	Parameter estimate covariance matrix	MODEL	COVB
IterEM	Iteration history for Turnbull algorithm	PROBPLOT	ITPRINTEM
FitStatistics	Fit statistics	MODEL	Default
FitStatisticsUL	Fit statistics for unlogged response	MODEL	DISTRIBUTION=WEIBULL, LOGNORMAL, LLOGISTIC, or GAMMA
IterHistory	Iteration history	MODEL	ITPRINT
LagrangeStatistics	Lagrange statistics	MODEL	NOINT | NOSCALE
LastGrad	Last evaluation of the gradient	MODEL	ITPRINT
LastHess	Last evaluation of the Hessian	MODEL	ITPRINT
ModelInfo	Model information	MODEL	Default
NObs	Number of observations	MODEL	Default
ParameterEstimates	Parameter estimates	MODEL	Default
ParmInfo	Parameter indices	MODEL	Default
ProbabilityEstimates	Nonparametric CDF estimates	PROBPLOT	PPOUT
TConvergenceStatus	Convergence status for Turnbull algorithm	PROBPLOT	Default
Turnbull	Probability estimates from Turnbull algorithm	PROBPLOT	ITPRINTEM
Type3Analysis	Type 3 tests	MODEL	Default*

* Depending on the data.

Table 76.15 ODS Tables Produced in PROC LIFEREG for a Bayesian Analysis

ODS Table Name	Description	Statement	Option
AutoCorr	Autocorrelations of the posterior samples	BAYES	Default
ClassLevels	Classification variable levels	CLASS	Default*
CoeffPrior	Prior distribution of the regression coefficients	BAYES	Default
ConvergenceStatus	Convergence status of maximum likelihood estimation	MODEL	Default
Corr	Correlation matrix of the posterior samples	BAYES	SUMMARY=CORR
ESS	Effective sample size	BAYES	Default
FitStatistics	Fit statistics	BAYES	Default
Gelman	Gelman and Rubin convergence diagnostics	BAYES	DIAG=GELMAN
Geweke	Geweke convergence diagnostics	BAYES	Default
Heidelberger	Heidelberger and Welch convergence diagnostics	BAYES	DIAG=HEIDELBERGER
InitialValues	Initial values of the Markov chains	BAYES	Default
MCError	Monte Carlo standard errors	BAYES	DIAG=MCSE
ModelInfo	Model information	MODEL	Default
NObs	Number of observations	MODEL	Default
ParameterEstimates	Maximum likelihood estimates of model parameters	MODEL	Default
ParmPrior	Prior distribution for scale and shape	BAYES	Default
PostIntervals	HPD and equal-tail intervals of the posterior samples	BAYES	Default
PosteriorSample	Posterior samples (for output data set only)	BAYES	
PostSummaries	Summary statistics of the posterior samples	BAYES	Default
Raftery	Raftery and Lewis convergence diagnostics	BAYES	DIAG=RAFTERY

* Depending on the data.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

Some graphs are produced by default; other graphs are produced by using statements and options.

ODS Graph Names

PROC LIFEREG assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names of the graphs that PROC LIFEREG generates are listed in Table 76.16, along with the required statements and options.

Table 76.16 Graphs Produced by PROC LIFEREG

ODS Graph Name	Description	Statement	Option
ADPanel	Autocorrelation function and density panel	BAYES	PLOTS=(AUTOCORR DENSITY)
AutocorrPanel	Autocorrelation function panel	BAYES	PLOTS= AUTOCORR
AutocorrPlot	Autocorrelation function plot	BAYES	PLOTS(UNPACK)=AUTOCORR
ProbPlot	Probability plot	PROBPLOT	Default
TAPanel	Trace and autocorrelation function panel	BAYES	PLOTS=(TRACE AUTOCORR)
TADPanel	Trace, autocorrelation, and density function panel	BAYES	Default
TDPanel	Trace and density panel	BAYES	PLOTS=(TRACE DENSITY)
TracePanel	Trace panel	BAYES	PLOTS=TRACE
TracePlot	Trace plot	BAYES	PLOTS(UNPACK)=TRACE

Examples: LIFEREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 76.1: Motorette Failure

(View the complete code for this example (liferex1.sas) in the example repository.)

This example fits a Weibull model and a lognormal model to the example given in Kalbfleisch and Prentice (1980, p. 5). An output data set called models is specified to contain the parameter estimates. By default, the natural log of the variable time is used by the procedure as the response. After this log transformation, the Weibull model is fit using the extreme-value baseline distribution, and the lognormal is fit using the normal baseline distribution.

Since the extreme-value and normal distributions do not contain any shape parameters, the variable SHAPE1 is missing in the models data set. An additional output data set, out, is created that contains the predicted quantiles and their standard errors for values of the covariate corresponding to temp=130 and temp=150. This is done with the control variable, which is set to 1 for only two observations.

Using the standard error estimates obtained from the output data set, approximate 90% confidence limits for the predicted quantities are then created in a subsequent DATA step for the log response. The logs of the predicted values are obtained because the values of the P= variable in the OUT= data set are in the same units as the original response variable, time. The standard errors of the quantiles of log(time) are approximated (using a Taylor series approximation) by the standard deviation of time divided by the mean value of time. These confidence limits are then converted back to the original scale by the exponential function.

The following statements produce Output 76.1.1:

```
title 'Motorette Failures With Operating Temperature as a Covariate';
data motors;
   input time censor temp @@;
   if _N_=1 then
      do;
         temp=130;
         time=.;
         control=1;
         z=1000/(273.2+temp);
         output;
         temp=150;
         time=.;
         control=1;
         z=1000/(273.2+temp);
         output;
      end;
   if temp>150;
   control=0;
   z=1000/(273.2+temp);
   output;
   datalines;
```

```
8064 0 150 8064 0 150 8064 0 150 8064 0 150 8064 0 150
8064 0 150 8064 0 150 8064 0 150 8064 0 150 8064 0 150
1764 1 170 2772 1 170 3444 1 170 3542 1 170 3780 1 170
4860 1 170 5196 1 170 5448 0 170 5448 0 170 5448 0 170
 408 1 190  408 1 190 1344 1 190 1344 1 190 1440 1 190
1680 0 190 1680 0 190 1680 0 190 1680 0 190 1680 0 190
 408 1 220  408 1 220  504 1 220  504 1 220  504 1 220
 528 0 220  528 0 220  528 0 220  528 0 220  528 0 220
;

proc print data=motors;
run;
```

Output 76.1.1 Motorette Failure Data

Motorette Failures With Operating Temperature as a Covariate

Obs	time	censor	temp	control	z
1	.	0	130	1	2.48016
2	.	0	150	1	2.36295
3	1764	1	170	0	2.25632
4	2772	1	170	0	2.25632
5	3444	1	170	0	2.25632
6	3542	1	170	0	2.25632
7	3780	1	170	0	2.25632
8	4860	1	170	0	2.25632
9	5196	1	170	0	2.25632
10	5448	0	170	0	2.25632
11	5448	0	170	0	2.25632
12	5448	0	170	0	2.25632
13	408	1	190	0	2.15889
14	408	1	190	0	2.15889
15	1344	1	190	0	2.15889
16	1344	1	190	0	2.15889
17	1440	1	190	0	2.15889
18	1680	0	190	0	2.15889
19	1680	0	190	0	2.15889
20	1680	0	190	0	2.15889
21	1680	0	190	0	2.15889
22	1680	0	190	0	2.15889
23	408	1	220	0	2.02758
24	408	1	220	0	2.02758
25	504	1	220	0	2.02758
26	504	1	220	0	2.02758
27	504	1	220	0	2.02758
28	528	0	220	0	2.02758
29	528	0	220	0	2.02758
30	528	0	220	0	2.02758
31	528	0	220	0	2.02758
32	528	0	220	0	2.02758

The following statements produce Output 76.1.2 and Output 76.1.3:

```
proc lifereg data=motors outest=modela covout;
   a: model time*censor(0)=z;
      output out=outa quantiles=.1 .5 .9 std=std p=predtime
         control=control;
run;

proc lifereg data=motors outest=modelb covout;
   b: model time*censor(0)=z / dist=lnormal;
      output out=outb quantiles=.1 .5 .9 std=std p=predtime
         control=control;
run;
```

Output 76.1.2 Motorette Failure: Model A

Motorette Failures With Operating Temperature as a Covariate

The LIFEREG Procedure

Model Information	
Data Set	WORK.MOTORS
Dependent Variable	Log(time)
Censoring Variable	censor
Censoring Value(s)	0
Number of Observations	30
Noncensored Values	17
Right Censored Values	13
Left Censored Values	0
Interval Censored Values	0
Number of Parameters	3
Name of Distribution	Weibull
Log Likelihood	-22.95148315

Type III Analysis of Effects

Effect	DF	Wald Chi-Square	Pr > ChiSq
z	1	99.5239	<.0001

Analysis of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-11.8912	1.9655	-15.7435	-8.0389	36.60	<.0001
z	1	9.0383	0.9060	7.2626	10.8141	99.52	<.0001
Scale	1	0.3613	0.0795	0.2347	0.5561		
Weibull Shape	1	2.7679	0.6091	1.7982	4.2605		

Output 76.1.3 Motorette Failure: Model B

Motorette Failures With Operating Temperature as a Covariate

The LIFEREG Procedure

Model Information	
Data Set	WORK.MOTORS
Dependent Variable	Log(time)
Censoring Variable	censor
Censoring Value(s)	0
Number of Observations	30
Noncensored Values	17
Right Censored Values	13
Left Censored Values	0
Interval Censored Values	0
Number of Parameters	3
Name of Distribution	Lognormal
Log Likelihood	-24.47381031

Type III Analysis of Effects

Effect	DF	Wald Chi-Square	Pr > ChiSq
z	1	42.0001	<.0001

Analysis of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-10.4706	2.7719	-15.9034	-5.0377	14.27	0.0002
z	1	8.3221	1.2841	5.8052	10.8389	42.00	<.0001
Scale	1	0.6040	0.1107	0.4217	0.8652		

The following statements produce Output 76.1.4:

```
data models;
   set modela modelb;
run;

proc print data=models;
   id _model_;
   title 'Fitted Models';
run;
```

Output 76.1.4 Motorette Failure: Fitted Models

Fitted Models

MODEL	_NAME_	_TYPE_	_DIST_	_STATUS_	_LNLIKE_	time	Intercept	z	_SCALE_
a	time	PARMS	Weibull	0 Converged	-22.9515	-1.0000	-11.8912	9.03834	0.36128
a	Intercept	COV	Weibull	0 Converged	-22.9515	-11.8912	3.8632	-1.77878	0.03448
a	z	COV	Weibull	0 Converged	-22.9515	9.0383	-1.7788	0.82082	-0.01488
a	Scale	COV	Weibull	0 Converged	-22.9515	0.3613	0.0345	-0.01488	0.00632
b	time	PARMS	Lognormal	0 Converged	-24.4738	-1.0000	-10.4706	8.32208	0.60403
b	Intercept	COV	Lognormal	0 Converged	-24.4738	-10.4706	7.6835	-3.55566	0.03267
b	z	COV	Lognormal	0 Converged	-24.4738	8.3221	-3.5557	1.64897	-0.01285
b	Scale	COV	Lognormal	0 Converged	-24.4738	0.6040	0.0327	-0.01285	0.01226

The following statements produce Output 76.1.5:

```
data out;
   set outa outb;
run;

data out1;
   set out;
   ltime=log(predtime);
   stde=std/predtime;
   upper=exp(ltime+1.64*stde);
   lower=exp(ltime-1.64*stde);
run;

title 'Quantile Estimates and Confidence Limits';
proc print data=out1;
   id temp;
run;
title;
```

Output 76.1.5 Motorette Failure: Quantile Estimates and Confidence Limits

Quantile Estimates and Confidence Limits

temp	time	censor	control	z	_PROB_	predtime	std	ltime	stde	upper	lower
130	.	0	1	2.48016	0.1	16519.27	5999.85	9.7123	0.36320	29969.51	9105.47
130	.	0	1	2.48016	0.5	32626.65	9874.33	10.3929	0.30265	53595.71	19861.63
130	.	0	1	2.48016	0.9	50343.22	15044.35	10.8266	0.29884	82183.49	30838.80
150	.	0	1	2.36295	0.1	5726.74	1569.34	8.6529	0.27404	8976.12	3653.64
150	.	0	1	2.36295	0.5	11310.68	2299.92	9.3335	0.20334	15787.62	8103.28
150	.	0	1	2.36295	0.9	17452.49	3629.28	9.7672	0.20795	24545.37	12409.24
130	.	0	1	2.48016	0.1	12033.19	5482.34	9.3954	0.45560	25402.68	5700.09
130	.	0	1	2.48016	0.5	26095.68	11359.45	10.1695	0.43530	53285.36	12779.95
130	.	0	1	2.48016	0.9	56592.19	26036.90	10.9436	0.46008	120349.65	26611.42
150	.	0	1	2.36295	0.1	4536.88	1443.07	8.4200	0.31808	7643.71	2692.83
150	.	0	1	2.36295	0.5	9838.86	2901.15	9.1941	0.29487	15957.38	6066.36
150	.	0	1	2.36295	0.9	21336.97	7172.34	9.9682	0.33615	37029.72	12294.62

Example 76.2: Computing Predicted Values for a Tobit Model

(View the complete code for this example (liferex2.sas) in the example repository.)

The LIFEREG procedure can be used to perform a Tobit analysis. The Tobit model, described by Tobin (1958), is a regression model for left-censored data assuming a normally distributed error term. The model parameters are estimated by maximum likelihood. PROC LIFEREG provides estimates of the parameters of the distribution of the *uncensored* data. See Greene (1993) and Maddala (1983) for a more complete discussion of censored normal data and related distributions. This example shows how you can use PROC LIFEREG and the DATA step to compute two of the three types of predicted values discussed there.

Consider a continuous random variable Y and a constant C. If you were to sample from the distribution of Y but discard values less than (greater than) C, the distribution of the remaining observations would be *truncated* on the left (right). If you were to sample from the distribution of Y and report values less than (greater than) C as C, the distribution of the sample would be left (right) *censored*.

The probability density function of the truncated random variable Y' is given by

$$f_{\mathrm{Y}'}(y) = \frac{f_{\mathrm{Y}}(y)}{\Pr(\mathrm{Y} > \mathrm{C})} \quad \text{for} \quad y > \mathrm{C}$$

where $f_{\mathrm{Y}}(y)$ is the probability density function of Y. PROC LIFEREG cannot compute the proper likelihood function to estimate parameters or predicted values for a truncated distribution. Suppose the model being fit is specified as follows:

$$\mathrm{Y}_i^* = \mathbf{x}_i'\boldsymbol{\beta} + \epsilon_i$$

where ϵ_i is a normal error term with zero mean and standard deviation σ.

Define the censored random variable Y_i as

$$\begin{aligned} \mathrm{Y}_i &= 0 \text{ if } \mathrm{Y}_i^* \le 0 \\ \mathrm{Y}_i &= \mathrm{Y}_i^* \text{ if } \mathrm{Y}_i^* > 0 \end{aligned}$$

This is the Tobit model for left-censored normal data. Y_i^* is sometimes called the *latent variable*. PROC LIFEREG estimates parameters of the distribution of Y_i^* by maximum likelihood.

You can use the LIFEREG procedure to compute predicted values based on the mean functions of the latent and observed variables. The mean of the latent variable Y_i^* is $\mathbf{x}_i'\boldsymbol{\beta}$, and you can compute values of the mean for different settings of $\mathbf{x}_i$ by specifying XBETA=*variable-name* in an OUTPUT statement. Estimates of $\mathbf{x}_i'\boldsymbol{\beta}$ for each observation will be written to the OUT= data set. Predicted values of the observed variable Y_i can be computed based on the mean

$$E(\mathrm{Y}_i) = \Phi\left(\frac{\mathbf{x}_i'\boldsymbol{\beta}}{\sigma}\right)(\mathbf{x}_i'\boldsymbol{\beta} + \sigma\lambda_i)$$

where

$$\lambda_i = \frac{\phi(\mathbf{x}_i'\boldsymbol{\beta}/\sigma)}{\Phi(\mathbf{x}_i'\boldsymbol{\beta}/\sigma)}$$

ϕ and Φ represent the normal probability density and cumulative distribution functions.

Although the distribution of ϵ_i in the Tobit model is often assumed normal, you can use other distributions for the Tobit model in the LIFEREG procedure by specifying a distribution with the DISTRIBUTION= option in the MODEL statement. One distribution that should be mentioned is the logistic distribution. For this distribution, the MLE has bounded influence function with respect to the response variable, but not the design variables. If you believe your data have outliers in the response direction, you might try this distribution for some robust estimation of the Tobit model.

With the logistic distribution, the predicted values of the observed variable Y_i can be computed based on the mean of Y_i^*,

$$E(\mathrm{Y}_i) = \sigma \ln(1 + \exp(\mathbf{x}_i' \boldsymbol{\beta}/\sigma))$$

The following table shows a subset of the Mroz (1987) data set. In these data, Hours is the number of hours the wife worked outside the household in a given year, Yrs_Ed is the years of education, and Yrs_Exp is the years of work experience. A Tobit model will be fit to the hours worked with years of education and experience as covariates.

Hours	Yrs_Ed	Yrs_Exp
0	8	9
0	8	12
0	9	10
0	10	15
0	11	4
0	11	6
1000	12	1
1960	12	29
0	13	3
2100	13	36
3686	14	11
1920	14	38
0	15	14
1728	16	3
1568	16	19
1316	17	7
0	17	15

If the wife was not employed (worked 0 hours), her hours worked will be left censored at zero. In order to accommodate left censoring in PROC LIFEREG, you need two variables to indicate censoring status of observations. You can think of these variables as lower and upper endpoints of interval censoring. If there is no censoring, set both variables to the observed value of Hours. To indicate left censoring, set the lower endpoint to missing and the upper endpoint to the censored value, zero in this case.

The following statements create a SAS data set with the variables Hours, Yrs_Ed, and Yrs_Exp from the preceding data. A new variable, Lower, is created such that Lower=. if Hours=0 and Lower=Hours if Hours>0.

```
data subset;
   input Hours Yrs_Ed Yrs_Exp @@;
   if Hours eq 0
      then Lower=.;
      else Lower=Hours;
   datalines;
0 8 9 0 8 12 0 9 10 0 10 15 0 11 4 0 11 6
1000 12 1 1960 12 29 0 13 3 2100 13 36
3686 14 11 1920 14 38 0 15 14 1728 16 3
1568 16 19 1316 17 7 0 17 15
;
```

The following statements fit a normal regression model to the left-censored Hours data with Yrs_Ed and Yrs_Exp as covariates. You need the estimated standard deviation of the normal distribution to compute the predicted values of the censored distribution from the preceding formulas. The data set OUTEST contains the standard deviation estimate in a variable named _SCALE_. You also need estimates of $\mathbf{x}_i'\boldsymbol{\beta}$. These are contained in the data set OUT as the variable Xbeta.

```
proc lifereg data=subset outest=OUTEST(keep=_scale_);
   model (lower, hours) = yrs_ed yrs_exp / d=normal;
   output out=OUT xbeta=Xbeta;
run;
```

Output 76.2.1 shows the results of the model fit. These tables show parameter estimates for the uncensored, or latent variable, distribution.

Output 76.2.1 Parameter Estimates from PROC LIFEREG

The LIFEREG Procedure

Model Information	
Data Set	WORK.SUBSET
Dependent Variable	Lower
Dependent Variable	Hours
Number of Observations	17
Noncensored Values	8
Right Censored Values	0
Left Censored Values	9
Interval Censored Values	0
Number of Parameters	4
Name of Distribution	Normal
Log Likelihood	-74.9369977

Analysis of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-5598.64	2850.248	-11185.0	-12.2553	3.86	0.0495
Yrs_Ed	1	373.1477	191.8872	-2.9442	749.2397	3.78	0.0518
Yrs_Exp	1	63.3371	38.3632	-11.8533	138.5276	2.73	0.0987
Scale	1	1582.870	442.6732	914.9433	2738.397		

The following statements combine the two data sets created by PROC LIFEREG to compute predicted values

for the censored distribution. The OUTEST= data set contains the estimate of the standard deviation from the uncensored distribution, and the OUT= data set contains estimates of $\mathbf{x}_i'\boldsymbol{\beta}$.

```
data predict;
   drop lambda _scale_ _prob_;
   set out;
   if _n_ eq 1 then set outest;
   lambda = pdf('NORMAL',Xbeta/_scale_)
            / cdf('NORMAL',Xbeta/_scale_);
   Predict = cdf('NORMAL', Xbeta/_scale_)
             * (Xbeta + _scale_*lambda);
   label Xbeta='MEAN OF UNCENSORED VARIABLE'
         Predict = 'MEAN OF CENSORED VARIABLE';
run;
```

Output 76.2.2 shows the original variables, the predicted means of the uncensored distribution, and the predicted means of the censored distribution.

Output 76.2.2 Predicted Means from PROC LIFEREG

Hours	Lower	Yrs_Ed	Yrs_Exp	MEAN OF UNCENSORED VARIABLE	MEAN OF CENSORED VARIABLE
0	.	8	9	-2043.42	73.46
0	.	8	12	-1853.41	94.23
0	.	9	10	-1606.94	128.10
0	.	10	15	-917.10	276.04
0	.	11	4	-1240.67	195.76
0	.	11	6	-1113.99	224.72
1000	1000	12	1	-1057.53	238.63
1960	1960	12	29	715.91	1052.94
0	.	13	3	-557.71	391.42
2100	2100	13	36	1532.42	1672.50
3686	3686	14	11	322.14	805.58
1920	1920	14	38	2032.24	2106.81
0	.	15	14	885.30	1170.39
1728	1728	16	3	561.74	951.69
1568	1568	16	19	1575.13	1708.24
1316	1316	17	7	1188.23	1395.61
0	.	17	15	1694.93	1809.97

Example 76.3: Overcoming Convergence Problems by Specifying Initial Values

(View the complete code for this example (liferex3.sas) in the example repository.)

This example illustrates the use of parameter initial value specification to help overcome convergence difficulties.

The following statements create a SAS data set.

```
data raw;
   input censor x c1 @@;
   datalines;
0 16   0.00  0 17  0.00  0 18  0.00  0 17   0.04  0 18   0.04  0 18   0.04
0 23   0.40  0 22  0.40  0 22  0.40  0 33   4.00  0 34   4.00  0 35   4.00
1 54  40.00  1 54 40.00  1 54 40.00  1 54 400.00  1 54 400.00  1 54 400.00
;
```

Output 76.3.1 shows the contents of the data set raw.

Output 76.3.1 Contents of the Data Set

Obs	censor	x	c1
1	0	16	0.00
2	0	17	0.00
3	0	18	0.00
4	0	17	0.04
5	0	18	0.04
6	0	18	0.04
7	0	23	0.40
8	0	22	0.40
9	0	22	0.40
10	0	33	4.00
11	0	34	4.00
12	0	35	4.00
13	1	54	40.00
14	1	54	40.00
15	1	54	40.00
16	1	54	400.00
17	1	54	400.00
18	1	54	400.00

The following SAS statements request that a Weibull regression model be fit to the data:

```
title 'OLS (Default) Initial Values';
proc lifereg data=raw;
   model x*censor(1) = c1 / distribution = Weibull itprint;
run;
```

Convergence was not attained in 50 iterations for this model, as the following messages to the log indicate:

```
WARNING: Convergence was not attained in 50 iterations. You might want to
```

```
         increase the maximum number of iterations (MAXITER= option) or change
         the convergence criteria (CONVERGE = value) in the MODEL statement.
WARNING: The procedure is continuing in spite of the above warning. Results
         shown are based on the last maximum likelihood iteration. Validity
         of the model fit is questionable.
```

The first line (iter=0) of the iteration history table, shown in Output 76.3.2, shows the default initial ordinary least squares (OLS) estimates of the parameters.

Output 76.3.2 Initial Least Squares

OLS (Default) Initial Values

The LIFEREG Procedure

Iteration History for Parameter Estimates					
Iter	**Ridge**	**Loglikelihood**	**Intercept**	**c1**	**Scale**
0	0	-22.891088	3.2324769714	0.0020664542	0.3995754195
1	0	-16.427074	3.5337141598	0.0028713635	0.3283544365
2	0	-13.216768	3.4480787541	0.0052801225	0.3816964358
3	0	-5.0786635	3.1966395335	0.0191439929	0.2325418958
4	0	-2.0018885	3.1848047525	0.0275425402	0.1963590539
5	0	-0.1814984	3.1478989655	0.0374731819	0.2103607621
6	0	2.90712131	3.0858183316	0.0659946149	0.1818245261
7	0.063	2.9991781	3.1014479187	0.0661096622	0.1648677081
8	0.063	3.01557837	3.0995493638	0.0662333056	0.1670552505
9	0.063	3.0301815	3.0992317977	0.0663580659	0.1669529486
10	0.063	3.0448013	3.0989901232	0.0664827053	0.1667371524
11	0.063	3.05941254	3.0987507448	0.0666071514	0.1665197313
12	0.063	3.07401474	3.0985118143	0.0667314052	0.1663026517
13	0.063	3.08860788	3.0982732928	0.066855467	0.1660859472
14	0.063	3.10319193	3.0980351787	0.0669793371	0.1658696184
15	0.063	3.11776689	3.0977974713	0.0671030156	0.1656536651
16	0.063	3.13233272	3.0975601698	0.0672265029	0.1654380873
17	0.063	3.1468894	3.0973232737	0.0673497993	0.165222885
18	0.063	3.16143692	3.0970867821	0.0674729049	0.1650080579
19	0.063	3.17597526	3.0968506943	0.06759582	0.1647936061
20	0.063	3.19050439	3.0966150098	0.0677185449	0.1645795293
21	0.063	3.2050243	3.0963797277	0.0678410799	0.1643658275
22	0.063	3.21953496	3.0961448474	0.0679634252	0.1641525006
23	0.063	3.23403635	3.0959103682	0.068085581	0.1639395483
24	0.063	3.24852845	3.0956762896	0.0682075476	0.1637269705
25	0.063	3.26301123	3.0954426107	0.0683293253	0.1635147672
26	0.063	3.27748468	3.095209331	0.0684509143	0.163302938
27	0.063	3.29194878	3.0949764498	0.0685723149	0.1630914829
28	0.063	3.3064035	3.0947439665	0.0686935273	0.1628804017
29	0.063	3.32084881	3.0945118805	0.0688145517	0.1626696942
30	0.063	3.3352847	3.0942801911	0.0689353885	0.1624593601
31	0.063	3.34971114	3.0940488977	0.0690560378	0.1622493994
32	0.063	3.36412812	3.0938179997	0.0691765	0.1620398118
33	0.063	3.3785356	3.0935874965	0.0692967752	0.1618305971
34	0.063	3.39293356	3.0933573875	0.0694168637	0.161621755
35	0.063	3.40732199	3.093127672	0.0695367658	0.1614132855
36	0.063	3.42170085	3.0928983495	0.0696564816	0.1612051882
37	0.063	3.43607013	3.0926694194	0.0697760116	0.1609974629
38	0.063	3.45042979	3.0924408811	0.0698953558	0.1607901095
39	0.063	3.46477983	3.092212734	0.0700145146	0.1605831276
40	0.063	3.4791202	3.0919849776	0.0701334882	0.160376517
41	0.063	3.4934509	3.0917576112	0.0702522768	0.1601702775
42	0.063	3.50777188	3.0915306343	0.0703708808	0.1599644088
43	0.063	3.52208314	3.0913040464	0.0704893002	0.1597589108

Output 76.3.2 *continued*

OLS (Default) Initial Values

The LIFEREG Procedure

Iteration History for Parameter Estimates					
Iter	Ridge	Loglikelihood	Intercept	c1	Scale
44	0.063	3.53638465	3.0910778468	0.0706075354	0.159553783
45	0.063	3.55067637	3.0908520349	0.0707255867	0.1593490254
46	0.063	3.5649583	3.0906266104	0.0708434542	0.1591446376
47	0.063	3.57923039	3.0904015725	0.0709611382	0.1589406193
48	0.063	3.59349263	3.0901769207	0.0710786389	0.1587369703
49	0.063	3.607745	3.0899526546	0.0711959567	0.1585336903
50	0.063	3.62198746	3.0897287734	0.0713130916	0.1583307791

The log-logistic distribution is more robust to large values of the response than the Weibull distribution, so one approach to improving the convergence performance is to fit a log-logistic distribution, and if this converges, use the resulting parameter estimates as initial values in a subsequent fit of a model with the Weibull distribution.

The following statements fit a log-logistic distribution to the data:

```
proc lifereg data=raw;
   model x*censor(1) = c1 / distribution = llogistic;
run;
```

The algorithm converges, and the maximum likelihood estimates for the log-logistic distribution are shown in Output 76.3.3

Output 76.3.3 Estimates from the Log-Logistic Distribution

OLS (Default) Initial Values

The LIFEREG Procedure

Analysis of Maximum Likelihood Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	2.8983	0.0318	2.8360	2.9606	8309.43	<.0001
c1	1	0.1592	0.0133	0.1332	0.1852	143.85	<.0001
Scale	1	0.0498	0.0122	0.0308	0.0804		

The following statements refit the Weibull model by using the maximum likelihood estimates from the log-logistic fit as initial values:

```
proc lifereg data=raw outest=outest;
   model x*censor(1) = c1 / itprint distribution = weibull
                            intercept=2.898 initial=0.16 scale=0.05;
   output out=out xbeta=xbeta;
run;
```

Examination of the resulting output in Output 76.3.4 shows that the convergence problem has been solved by specifying different initial values.

Output 76.3.4 Final Estimates from the Weibull Distribution

OLS (Default) Initial Values

The LIFEREG Procedure

Model Information	
Data Set	WORK.RAW
Dependent Variable	Log(x)
Censoring Variable	censor
Censoring Value(s)	1
Number of Observations	18
Noncensored Values	12
Right Censored Values	6
Left Censored Values	0
Interval Censored Values	0
Number of Parameters	3
Name of Distribution	Weibull
Log Likelihood	11.232023272

Algorithm converged.

Analysis of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	2.9699	0.0326	2.9059	3.0338	8278.86	<.0001
c1	1	0.1435	0.0165	0.1111	0.1758	75.43	<.0001
Scale	1	0.0844	0.0189	0.0544	0.1308		
Weibull Shape	1	11.8526	2.6514	7.6455	18.3749		

As an example of an alternative way of specifying initial values, the following invocation of PROC LIFEREG, using the INEST= data set to provide starting values for the three parameters, is equivalent to the previous invocation:

```
data in;
   input  intercept c1 _scale_;
   datalines;
2.898 0.16 0.05
;

proc lifereg data=raw inest=in outest=outest;
   model x*censor(1) = c1 / itprint distribution = weibull;
   output out=out xbeta=xbeta;
run;
```

Example 76.4: Analysis of Arbitrarily Censored Data with Interaction Effects

(View the complete code for this example (liferex4.sas) in the example repository.)

The artificial data in this example are from a study of the natural recovery time of mice after injection of a certain toxin. Twenty mice were grouped by sex (sex: 1 = Male, 2 = Female) with equal sizes. Their ages (in days) were recorded at the injection. Their recovery times (in minutes) were also recorded. Toxin density in blood was used to decide whether a mouse recovered. Mice were checked at two times for recovery. If a mouse had recovered at the first time, the observation is left censored, and no further measurement is made. The variable time1 is set to missing and time2 is set to the measurement time to indicate left censoring. If a mouse had not recovered at the first time, it was checked later at a second time. If it had recovered by the second measurement time, the observation is interval censored, and the variable time1 is set to the first measurement time and time2 is set to the second measurement time. If there was no recovery at the second measurement, the observation is right censored, and time1 is set to the second measurement time and time2 is set to missing to indicate right censoring.

The following statements create a SAS data set containing the data from the experiment:

```
title 'Natural Recovery Time';
data mice;
   input sex age time1 time2;
   datalines;
1  57  631   631
1  45  .     170
1  54  227   227
1  43  143   143
1  64  916   .
1  67  691   705
1  44  100   100
1  59  730   .
1  47  365   365
1  74  1916  1916
2  79  1326  .
2  75  837   837
2  84  1200  1235
2  54  .     365
2  74  1255  1255
2  71  1823  .
2  65  537   637
2  33  583   683
2  77  955   .
2  46  577   577
;
```

The following SAS statements create the SAS data sets xrow1 and xrow2:

```
data xrow1;
   input sex age time1 time2;
   datalines;
1  50  .  .
;

data xrow2;
```

```
   input sex age time1 time2;
   datalines;
2  60.6  .  .
;
```

The following SAS statements fit a Weibull model with age, sex, and an age-by-sex interaction term as covariates, and create a plot of predicted probabilities against recovery time for the fixed values of age and sex specified in the SAS data set xrow1:

```
ods graphics on;
proc lifereg data=mice xdata=xrow1;
   class sex;
   model (time1, time2) = age sex age*sex / dist=Weibull;

   probplot  / nodata
       plower=.5
       vref(intersect) = 75
       vreflab = '75 Percent';
   inset;
run;
```

Standard output is shown in Output 76.4.1. Tables containing general model information, Type III tests for the main effects and interaction terms, and parameter estimates are created.

Output 76.4.1 Parameter Estimates for the Interaction Model

Natural Recovery Time

The LIFEREG Procedure

Model Information	
Data Set	WORK.MICE
Dependent Variable	Log(time1)
Dependent Variable	Log(time2)
Number of Observations	20
Noncensored Values	9
Right Censored Values	5
Left Censored Values	2
Interval Censored Values	4
Number of Parameters	5
Name of Distribution	Weibull
Log Likelihood	-25.91033295

Type III Analysis of Effects

Effect	DF	Wald Chi-Square	Pr > ChiSq
age	1	33.8496	<.0001
sex	1	14.0245	0.0002
age*sex	1	10.7196	0.0011

Output 76.4.1 *continued*

Analysis of Maximum Likelihood Parameter Estimates								
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	5.4110	0.5549	4.3234	6.4986	95.08	<.0001
age		1	0.0250	0.0086	0.0081	0.0419	8.42	0.0037
sex	1	1	-3.9808	1.0630	-6.0643	-1.8974	14.02	0.0002
sex	2	0	0.0000	.	.	.	.	.
age*sex	1	1	0.0613	0.0187	0.0246	0.0980	10.72	0.0011
age*sex	2	0	0.0000	.	.	.	.	.
Scale		1	0.4087	0.0900	0.2654	0.6294		
Weibull Shape		1	2.4468	0.5391	1.5887	3.7682		

The following two plots display the predicted probability against the recovery time for two different populations. Output 76.4.2 is created with the PROBPLOT statement with the option XDATA= xrow1, which specifies the population with sex = 1, age = 50. Output 76.4.3 is created with the PROBPLOT statement with the option XDATA= xrow2, which specifies the population with sex = 2, age = 60.6. These are the default values that the LIFEREG procedure would use for the probability plot if the XDATA= option had not been specified. Reference lines are used to display specified predicted probability points and their relative locations in the plot.

Output 76.4.2 Probability Plot for Recovery Time with sex = 1, age = 50

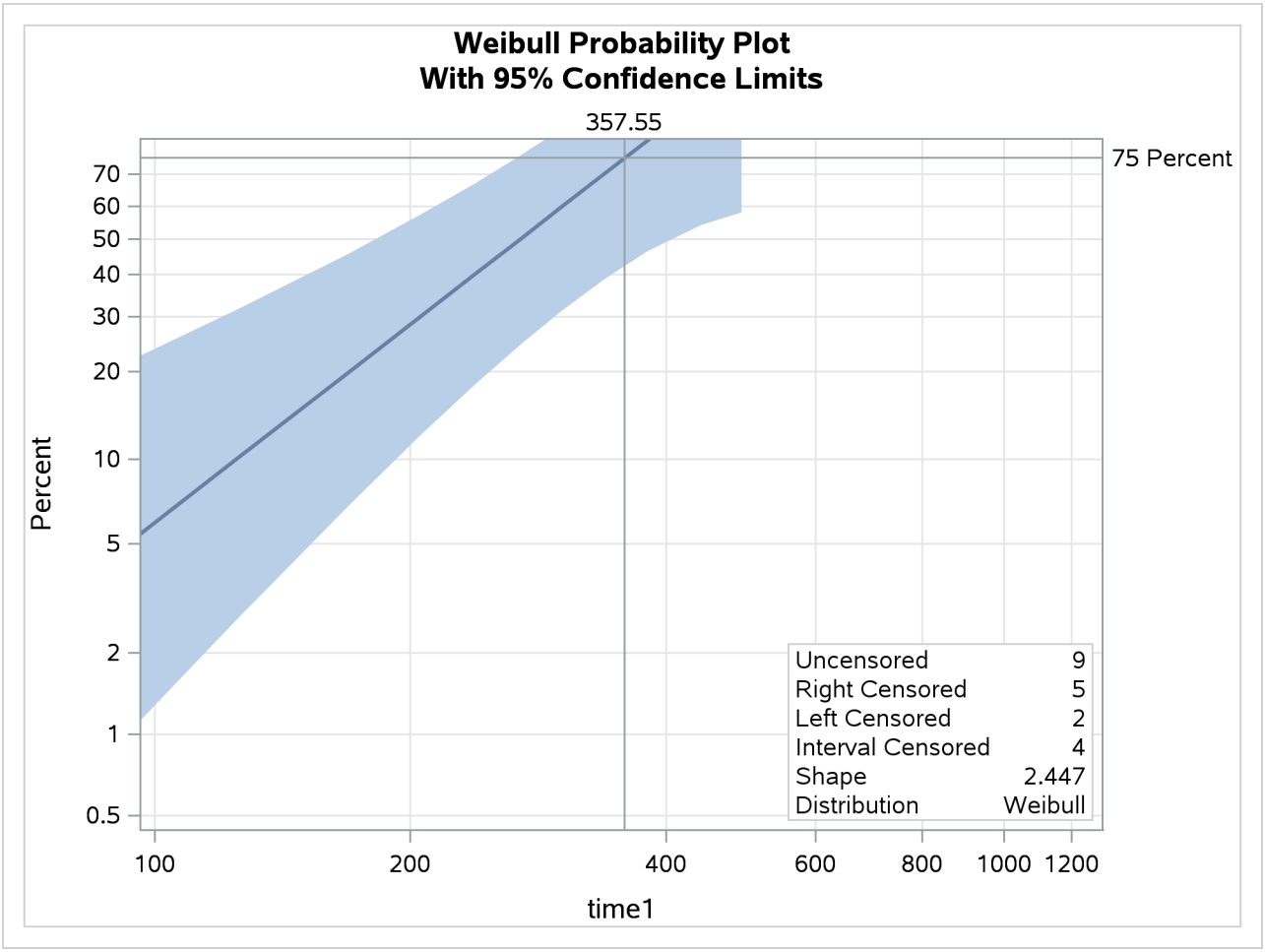

The following SAS statements fit a Weibull model with age, sex, and an age-by-sex interaction term as covariates, and create the plot of predicted probabilities against recovery time shown in Output 76.4.3, for the fixed values of age and sex specified in the SAS data set xrow2:

```
proc lifereg data=mice xdata=xrow2;
   class sex;
   model (time1, time2) = age sex age*sex / dist=Weibull;

   probplot  / nodata
       plower=.5
       vref(intersect) = 75
       vreflab = '75 Percent';
   inset;
run;
title;
```

Output 76.4.3 Probability Plot for Recovery Time with sex = 2, age = 60.6

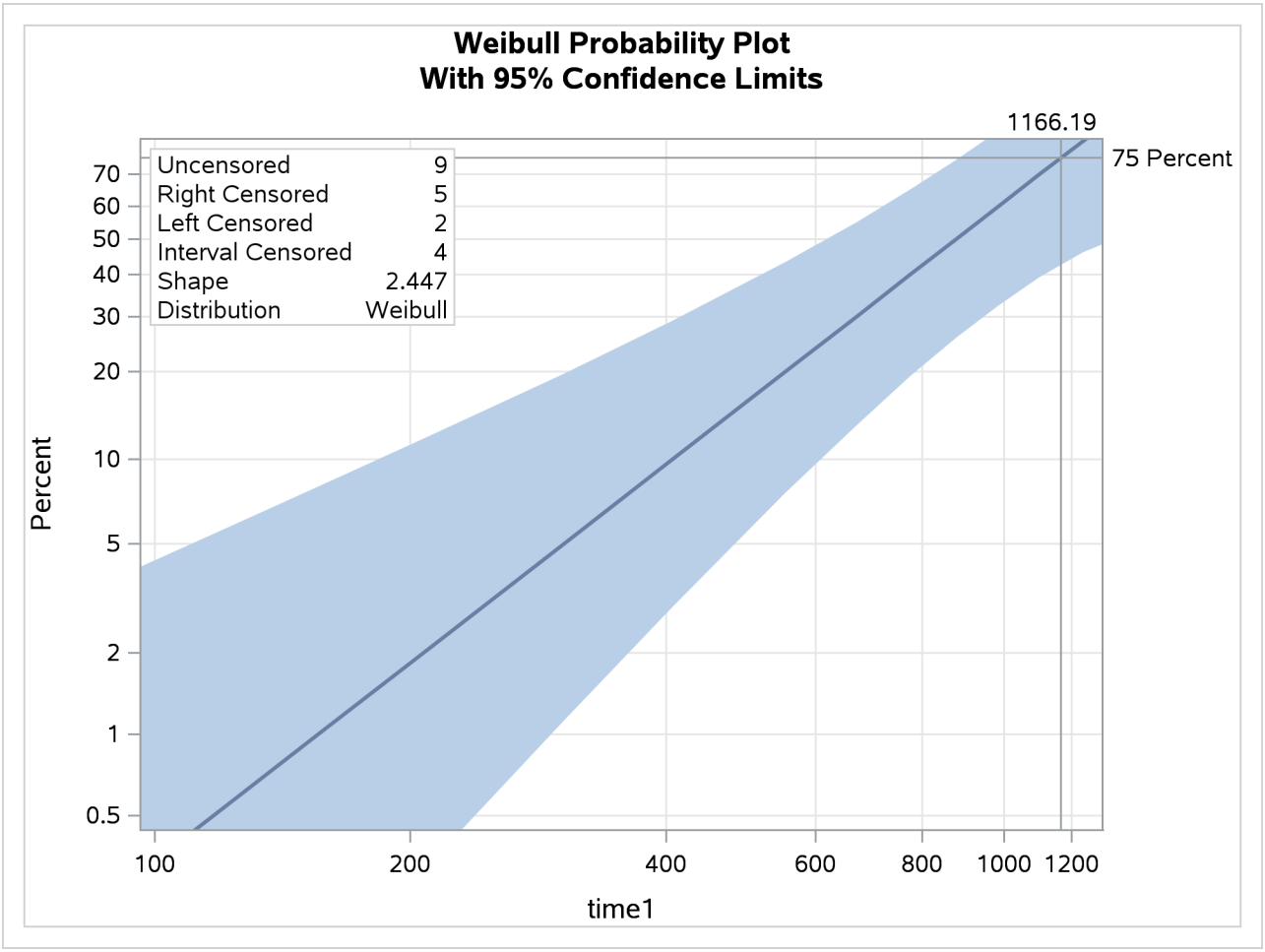

Example 76.5: Probability Plotting—Right Censoring

(View the complete code for this example (liferex5.sas) in the example repository.)

The following statements create a SAS data set containing observed and right-censored lifetimes of 70 diesel engine fans (Nelson 1982):

```
data Fan;
   input Lifetime Censor@@;
   Lifetime = Lifetime / 1000;
   datalines;
 450 0    460 1   1150 0   1150 0   1560 1
1600 0   1660 1   1850 1   1850 1   1850 1
1850 1   1850 1   2030 1   2030 1   2030 1
2070 0   2070 0   2080 0   2200 1   3000 1
3000 1   3000 1   3000 1   3100 0   3200 1
3450 0   3750 1   3750 1   4150 1   4150 1
4150 1   4150 1   4300 1   4300 1   4300 1
4300 1   4600 0   4850 1   4850 1   4850 1
```

```
4850 1   5000 1   5000 1   5000 1   6100 1
6100 0   6100 1   6100 1   6300 1   6450 1
6450 1   6700 1   7450 1   7800 1   7800 1
8100 1   8100 1   8200 1   8500 1   8500 1
8500 1   8750 1   8750 0   8750 1   9400 1
9900 1  10100 1  10100 1  10100 1  11500 1
;
```

Some of the fans had not failed at the time the data were collected, and the unfailed units have right-censored lifetimes. The variable LIFETIME represents either a failure time or a censoring time, in thousands of hours. The variable CENSOR is equal to 0 if the value of LIFETIME is a failure time, and it is equal to 1 if the value is a censoring time. The following statements use the LIFEREG procedure to produce the probability plot with an inset for the engine lifetimes:

```
ods graphics on;
proc lifereg data=Fan;
   model Lifetime*Censor( 1 ) = / d = Weibull;
   probplot
   ppout
   npintervals=simul;
   inset;
run;
```

The resulting graphical output is shown in Output 76.5.1. The estimated CDF, a line representing the maximum likelihood fit, and pointwise parametric confidence bands are plotted in the body of Output 76.5.1. The values of right-censored observations are plotted along the bottom of the graph. The "Cumulative Probability Estimates" table is also created in Output 76.5.2.

Output 76.5.1 Probability Plot for the Fan Data

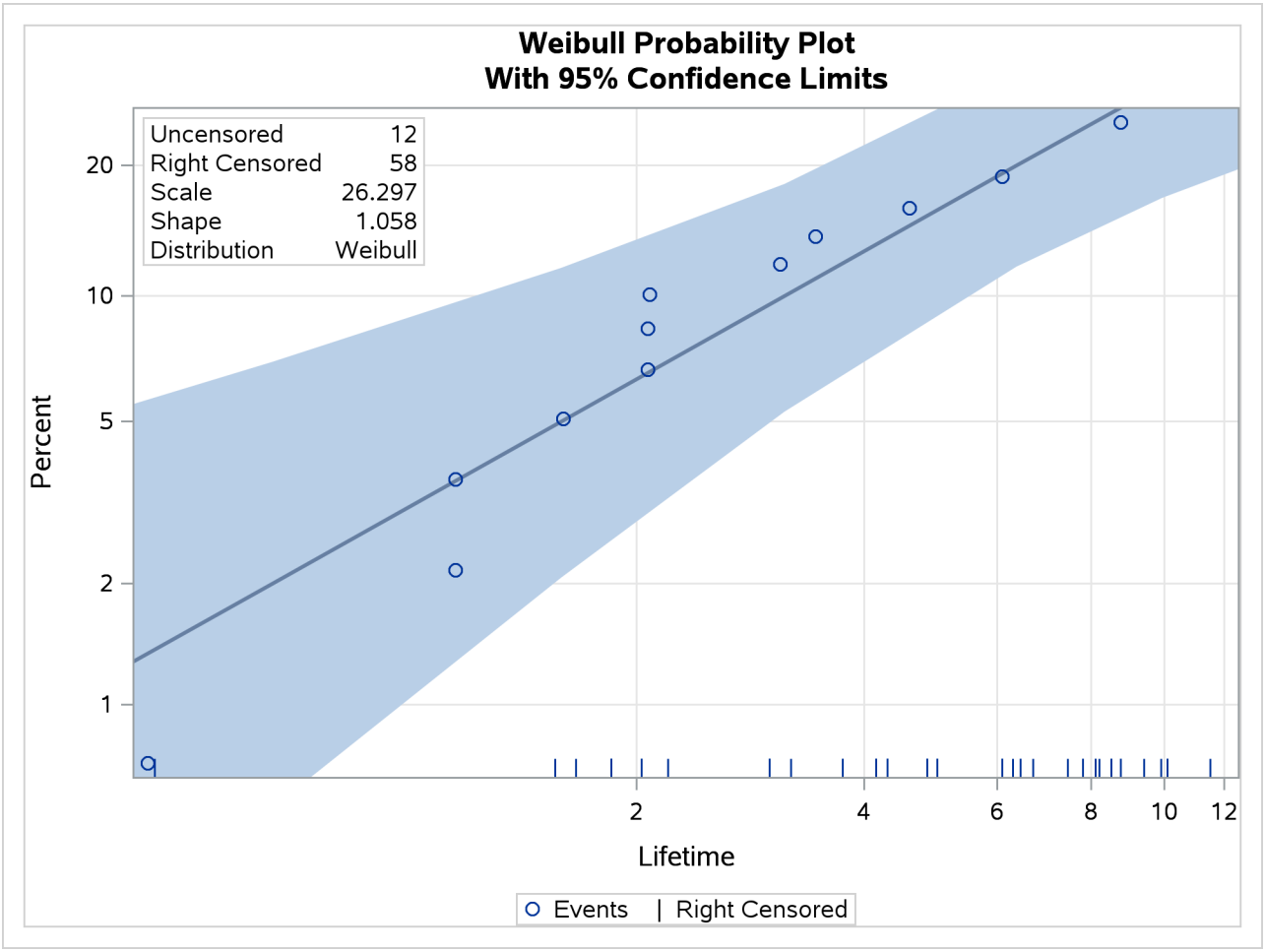

Output 76.5.2 CDF Estimates

Cumulative Probability Estimates					
		Simultaneous 95% Confidence Limits			
Lifetime	Cumulative Probability	Lower	Upper	Kaplan-Meier Estimate	Kaplan-Meier Standard Error
0.45	0.0071	0.0007	0.2114	0.0143	0.0142
1.15	0.0215	0.0033	0.2114	0.0288	0.0201
1.15	0.0360	0.0073	0.2168	0.0433	0.0244
1.6	0.0506	0.0125	0.2304	0.0580	0.0282
2.07	0.0666	0.0190	0.2539	0.0751	0.0324
2.07	0.0837	0.0264	0.2760	0.0923	0.0361
2.08	0.1008	0.0344	0.2972	0.1094	0.0392
3.1	0.1189	0.0436	0.3223	0.1283	0.0427
3.45	0.1380	0.0535	0.3471	0.1477	0.0460
4.6	0.1602	0.0653	0.3844	0.1728	0.0510
6.1	0.1887	0.0791	0.4349	0.2046	0.0581
8.75	0.2488	0.0884	0.6391	0.2930	0.0980

Example 76.6: Probability Plotting—Arbitrary Censoring

(View the complete code for this example (liferex6.sas) in the example repository.)

Table 76.17 contains microprocessor failure data (Nelson 1990). Units were inspected at predetermined time intervals. The data consist of inspection interval endpoints (in hours) and the number of units failing in each interval. A missing (.) lower endpoint indicates left censoring, and a missing upper endpoint indicates right censoring. These can be thought of as semi-infinite intervals with a lower (upper) endpoint of negative (positive) infinity for left (right) censoring.

Table 76.17 Interval-Censored Data

Lower Endpoint	Upper Endpoint	Number Failed
.	6	6
6	12	2
24	48	2
24	.	1
48	168	1
48	.	839
168	500	1
168	.	150
500	1000	2
500	.	149
1000	2000	1
1000	.	147
2000	.	122

The following SAS statements create the SAS data set Micro:

```
data Micro;
   input t1 t2 f;
   datalines;
. 6 6
6 12 2
12 24 0
24 48 2
24 .  1
48 168 1
48 .   839
168 500 1
168 .   150
500 1000 2
500 .    149
1000 2000 1
1000 . 147
2000 . 122
;
```

The following SAS statements compute the nonparametric Turnbull estimate of the cumulative distribution function and create a lognormal probability plot:

```
ods graphics on;
proc lifereg data=Micro;
   model ( t1 t2 ) = / d=lognormal intercept=25 scale=5;
   weight f;
   probplot
   pupper = 10
   itprintem
   printprobs
   maxitem = (1000,25)
   ppout;
   inset;
run;
```

The two initial values INTERCEPT=25 and SCALE=5 in the MODEL statement are used to aid convergence in the model-fitting algorithm.

The following tables are created by the PROBPLOT statement in addition to the standard tabular output from the MODEL statement. Output 76.6.1 shows the iteration history for the Turnbull estimate of the CDF for the microprocessor data. With both options ITPRINTEM and PRINTPROBS specified in the PROBPLOT statement, this table contains the log likelihoods and interval probabilities for every 25th iteration and the last iteration. It would contain only the log likelihoods if the option PRINTPROBS were not specified.

Output 76.6.1 Iteration History for the Turnbull Estimate

The LIFEREG Procedure

Iteration History for the Turnbull Estimate of the CDF									
Iteration	**Loglikelihood**	**(., 6)**	**(6, 12)**	**(24, 48)**	**(48, 168)**	**(168, 500)**	**(500, 1000)**	**(1000, 2000)**	**(2000, .)**
0	-1133.4051	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
25	-104.16622	0.00421644	0.00140548	0.00140648	0.00173338	0.00237846	0.00846094	0.04565407	0.93474475
50	-101.15151	0.00421644	0.00140548	0.00140648	0.00173293	0.00234891	0.00727679	0.01174486	0.96986811
75	-101.06641	0.00421644	0.00140548	0.00140648	0.00173293	0.00234891	0.00727127	0.00835638	0.9732621
100	-101.06534	0.00421644	0.00140548	0.00140648	0.00173293	0.00234891	0.00727125	0.00801814	0.97360037
125	-101.06533	0.00421644	0.00140548	0.00140648	0.00173293	0.00234891	0.00727125	0.00798438	0.97363413
130	-101.06533	0.00421644	0.00140548	0.00140648	0.00173293	0.00234891	0.00727125	0.007983	0.97363551

The table in Output 76.6.2 summarizes the Turnbull estimates of the interval probabilities, the reduced gradients, and Lagrange multipliers as described in the section "Arbitrarily Censored Data" on page 5690.

Output 76.6.2 Summary for the Turnbull Algorithm

Lower Lifetime	Upper Lifetime	Probability	Reduced Gradient	Lagrange Multiplier
.	6	0.0042	0	0
6	12	0.0014	0	0
24	48	0.0014	0	0
48	168	0.0017	0	0
168	500	0.0023	0	0
500	1000	0.0073	-7.219342E-9	0
1000	2000	0.0080	-0.037063236	0
2000	.	0.9736	0.0003038877	0

Output 76.6.3 shows the final estimate of the CDF, along with standard errors and nonparametric confidence limits. Two kinds of nonparametric confidence limits, pointwise or simultaneous, are available. The default is the pointwise nonparametric confidence limits. You can specify the simultaneous nonparametric confidence limits by using the NPINTERVALS=SIMUL option.

Output 76.6.3 Final CDF Estimates for Turnbull Algorithm

Cumulative Probability Estimates					
			Pointwise 95% Confidence Limits		
Lower Lifetime	**Upper Lifetime**	**Cumulative Probability**	**Lower**	**Upper**	**Standard Error**
6	6	0.0042	0.0019	0.0094	0.0017
12	24	0.0056	0.0028	0.0112	0.0020
48	48	0.0070	0.0038	0.0130	0.0022
168	168	0.0088	0.0047	0.0164	0.0028
500	500	0.0111	0.0058	0.0211	0.0037
1000	1000	0.0184	0.0094	0.0357	0.0063
2000	2000	0.0264	0.0124	0.0553	0.0101

Output 76.6.4 shows the CDF estimates, maximum likelihood fit, and pointwise parametric confidence limits plotted on a lognormal probability plot.

Output 76.6.4 Lognormal Probability Plot for the Microprocessor Data

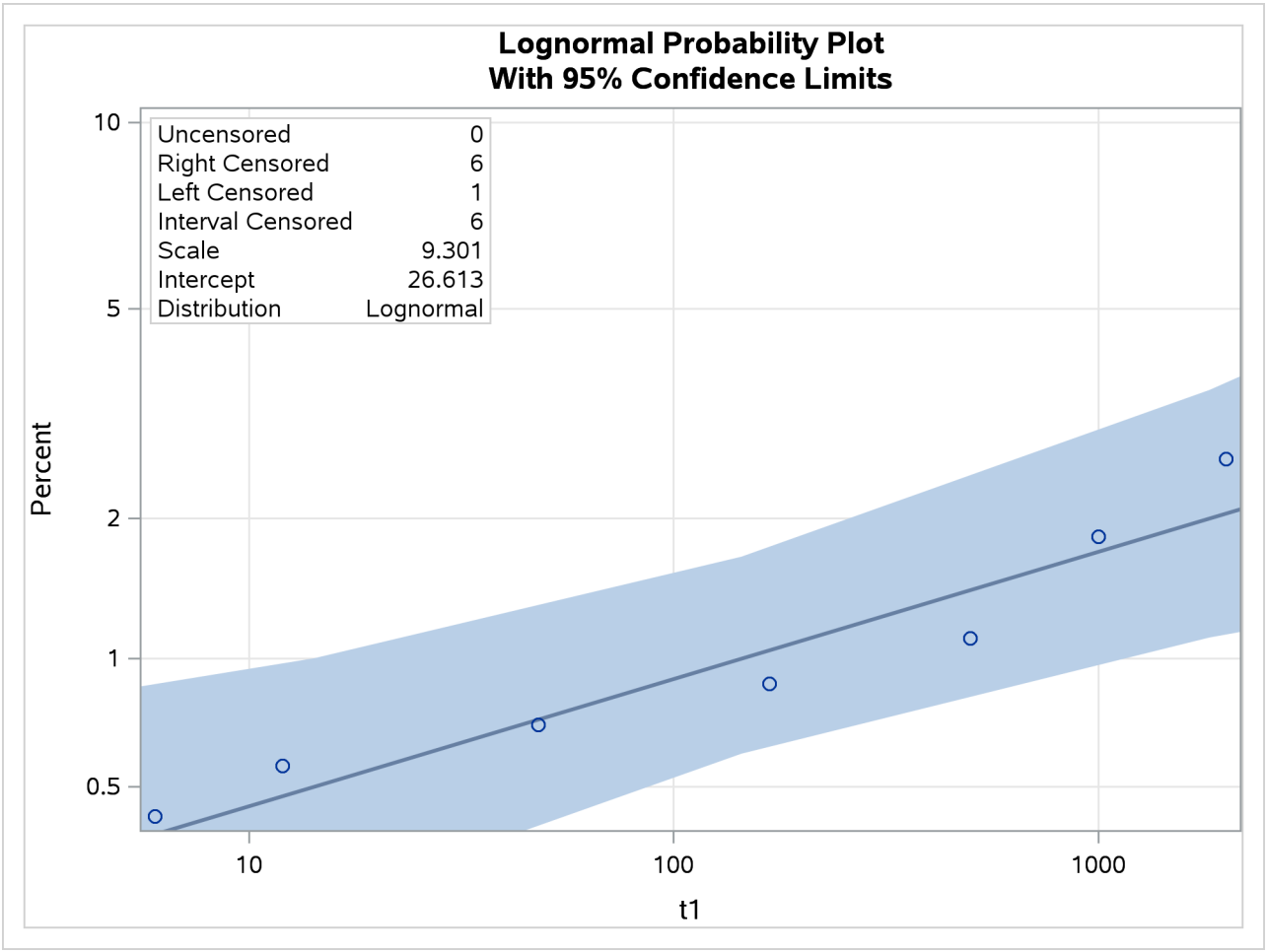

Example 76.7: Bayesian Analysis of Clinical Trial Data

(View the complete code for this example (liferex7.sas) in the example repository.)

Consider the data on melanoma patients from a clinical trial described in Ibrahim, Chen, and Sinha (2001). A partial listing of the data is shown in Output 76.7.1.

The survival time is modeled by a Weibull regression model with three covariates. An analysis of the right-censored survival data is performed with PROC LIFEREG to obtain Bayesian estimates of the regression coefficients by using the following SAS statements:

```
ods graphics on;
proc lifereg data=e1684;
   class Sex;
   model Survtime*Survcens(1)=Age Sex Perform / dist=Weibull;
   bayes WeibullShapePrior=gamma seed=9999;
```

```
run;
```

Output 76.7.1 Clinical Trial Data

Obs	survtime	survcens	age	sex	perform
1	1.57808	2	35.9945	1	0
2	1.48219	2	41.9014	1	0
3	7.33425	1	70.2164	2	0
4	0.65479	2	58.1753	2	1
5	2.23288	2	33.7096	1	0
6	9.38356	1	47.9726	1	0
7	3.27671	2	31.8219	2	0
8	0.00000	1	72.3644	2	0
9	0.80274	2	40.7151	2	0
10	9.64384	1	32.9479	1	0
11	1.66575	2	35.9205	1	0
12	0.94247	2	40.5068	2	0
13	1.68767	2	57.0384	1	0
14	5.94247	2	63.1452	1	0
15	2.34247	2	62.0630	1	0
16	0.89863	2	56.5342	1	1
17	9.03288	1	22.9945	2	0
18	9.63014	1	18.4712	1	0
19	0.52603	2	41.2521	1	0
20	1.82192	2	29.5178	1	0

Maximum likelihood estimates of the model parameters shown in Output 76.7.2 are displayed by default.

Output 76.7.2 Maximum Likelihood Parameter Estimates

The LIFEREG Procedure

Bayesian Analysis

Analysis of Maximum Likelihood Parameter Estimates

Parameter		DF	Estimate	Standard Error	95% Confidence Limits	
Intercept		1	2.4402	0.3716	1.7119	3.1685
age		1	-0.0115	0.0070	-0.0253	0.0023
sex	1	1	-0.1170	0.1978	-0.5046	0.2707
sex	2	0	0.0000	.	.	.
perform		1	0.2905	0.3222	-0.3411	0.9220
Scale		1	1.2537	0.0824	1.1021	1.4260
Weibull Shape		1	0.7977	0.0524	0.7012	0.9073

Since no prior distributions for the regression coefficients were specified, the default uniform improper distributions shown in the "Uniform Prior for Regression Coefficients" table in Output 76.7.3 are used. The specified gamma prior for the Weibull shape parameter is also shown in Output 76.7.3.

Output 76.7.3 Model Parameter Priors

The LIFEREG Procedure

Bayesian Analysis

Uniform Prior for Regression Coefficients	
Parameter	**Prior**
Intercept	Constant
age	Constant
sex1	Constant
perform	Constant

Independent Prior Distributions for Model Parameters					
Parameter	**Prior Distribution**	**Hyperparameters**			
Weibull Shape	Gamma	Shape	0.001	Inverse Scale	0.001

Fit statistics, descriptive statistics, interval statistics, and the sample parameter correlation matrix for the posterior sample are displayed in the tables in Output 76.7.4. Since noninformative prior distributions for the regression coefficients were used, the mean and standard deviations of the posterior distributions for the model parameters are close to the maximum likelihood estimates and standard errors.

Output 76.7.4 Posterior Sample Statistics

Fit Statistics	
DIC (smaller is better)	875.378
pD (effective number of parameters)	5.043

The LIFEREG Procedure

Bayesian Analysis

Posterior Summaries						
				Percentiles		
Parameter	**N**	**Mean**	**Standard Deviation**	**25%**	**50%**	**75%**
Intercept	10000	2.4834	0.3821	2.2210	2.4771	2.7396
age	10000	-0.0119	0.00724	-0.0167	-0.0118	-0.00703
sex1	10000	-0.1211	0.2043	-0.2580	-0.1205	0.0190
perform	10000	0.3383	0.3411	0.1021	0.3234	0.5615
WeibShape	10000	0.7833	0.0528	0.7478	0.7817	0.8186

Posterior Intervals					
Parameter	**Alpha**	**Equal-Tail Interval**		**HPD Interval**	
Intercept	0.050	1.7457	3.2469	1.7632	3.2591
age	0.050	-0.0264	0.00232	-0.0260	0.00253
sex1	0.050	-0.5226	0.2727	-0.5227	0.2726
perform	0.050	-0.2884	1.0353	-0.3218	0.9941
WeibShape	0.050	0.6833	0.8886	0.6815	0.8862

Output 76.7.4 *continued*

Posterior Correlation Matrix					
Parameter	Intercept	age	sex1	perform	WeibShape
Intercept	1.0000	-.8967	-.3116	-.0836	-.1324
age	-.8967	1.0000	-.0399	-.0425	0.0656
sex1	-.3116	-.0399	1.0000	0.1248	0.0365
perform	-.0836	-.0425	0.1248	1.0000	-.0413
WeibShape	-.1324	0.0656	0.0365	-.0413	1.0000

The default diagnostic statistics are displayed in Output 76.7.5. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details on Bayesian convergence diagnostics.

Output 76.7.5 Convergence Diagnostics

The LIFEREG Procedure

Bayesian Analysis

Posterior Autocorrelations				
Parameter	Lag 1	Lag 5	Lag 10	Lag 50
Intercept	0.0809	0.0252	-0.0092	-0.0013
age	0.0049	0.0085	-0.0071	0.0069
sex1	0.6378	0.0700	-0.0054	-0.0105
perform	0.6691	0.1061	-0.0027	0.0191
WeibShape	0.0926	0.0113	-0.0099	-0.0010

Geweke Diagnostics		
Parameter	z	Pr > |z|
Intercept	1.1010	0.2709
age	0.5537	0.5798
sex1	-2.2561	0.0241
perform	1.5282	0.1265
WeibShape	-2.4781	0.0132

Effective Sample Sizes			
		Autocorrelation	
Parameter	ESS	Time	Efficiency
Intercept	6832.5	1.4636	0.6832
age	10000.0	1.0000	1.0000
sex1	2457.3	4.0695	0.2457
perform	2179.9	4.5873	0.2180
WeibShape	7530.3	1.3280	0.7530

Trace, autocorrelation, and density plots for the seven model parameters are shown in Output 76.7.6 through Output 76.7.10. These plots show no indication that the Markov chains have not converged. See the sections "Assessing Markov Chain Convergence" on page 162 and "Visual Analysis via Trace Plots" on page 163 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more information about assessing the convergence of the chain of posterior samples.

Output 76.7.6 Diagnostic Plots

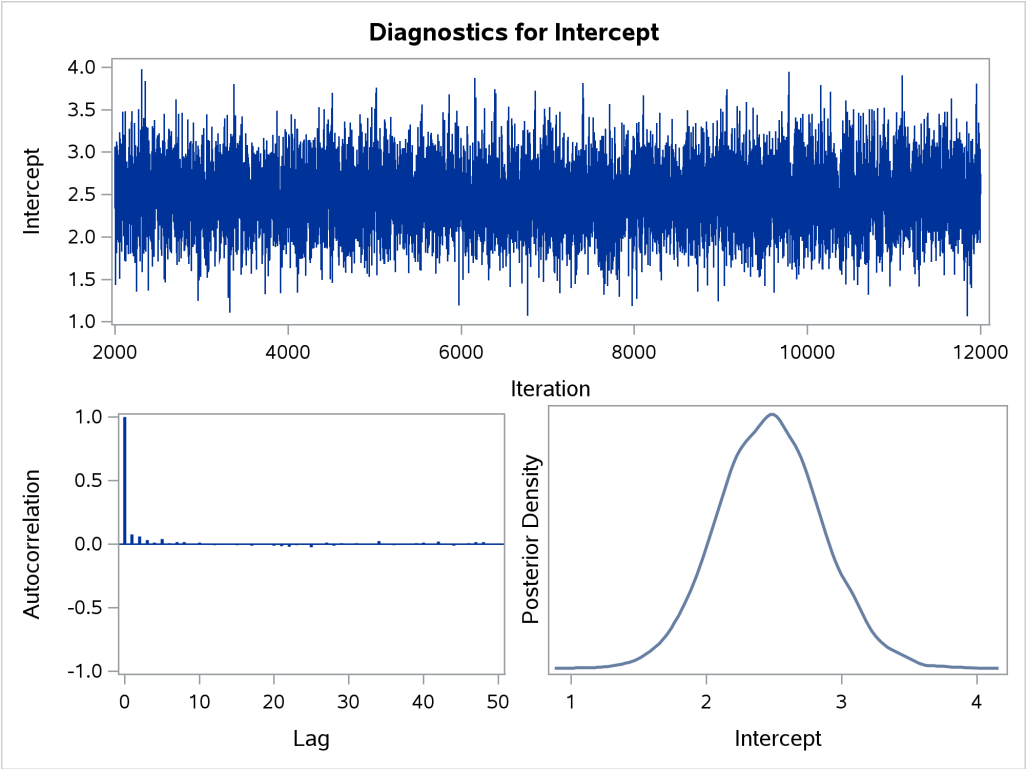

Output 76.7.7 Diagnostic Plots

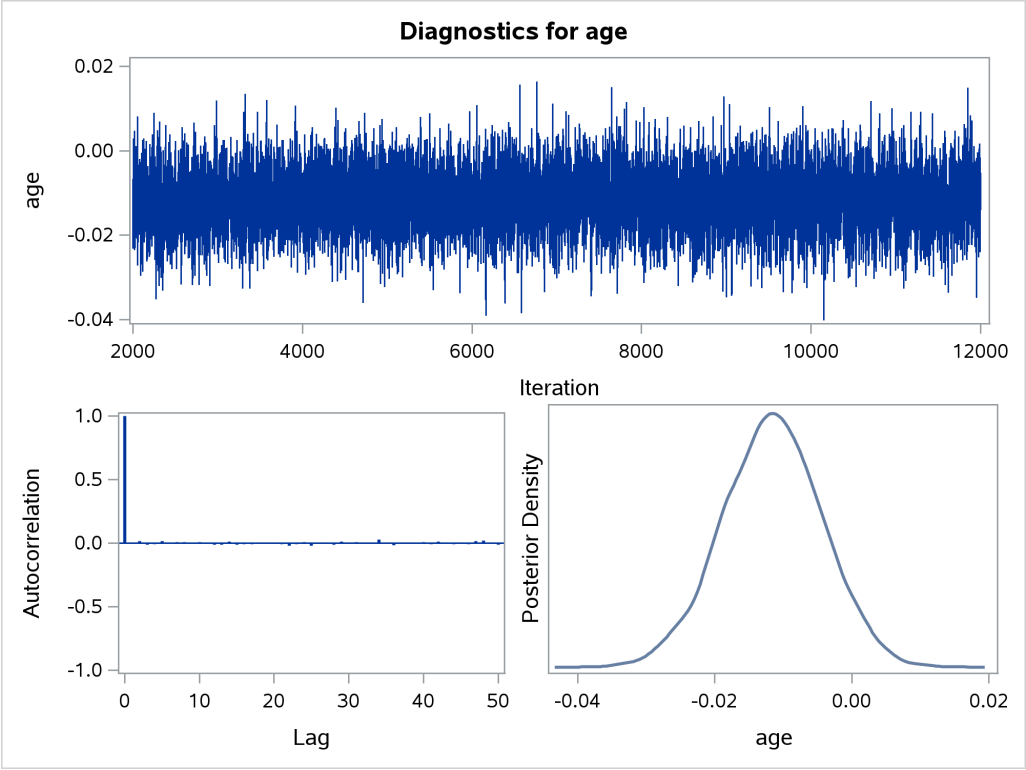

Output 76.7.8 Diagnostic Plots

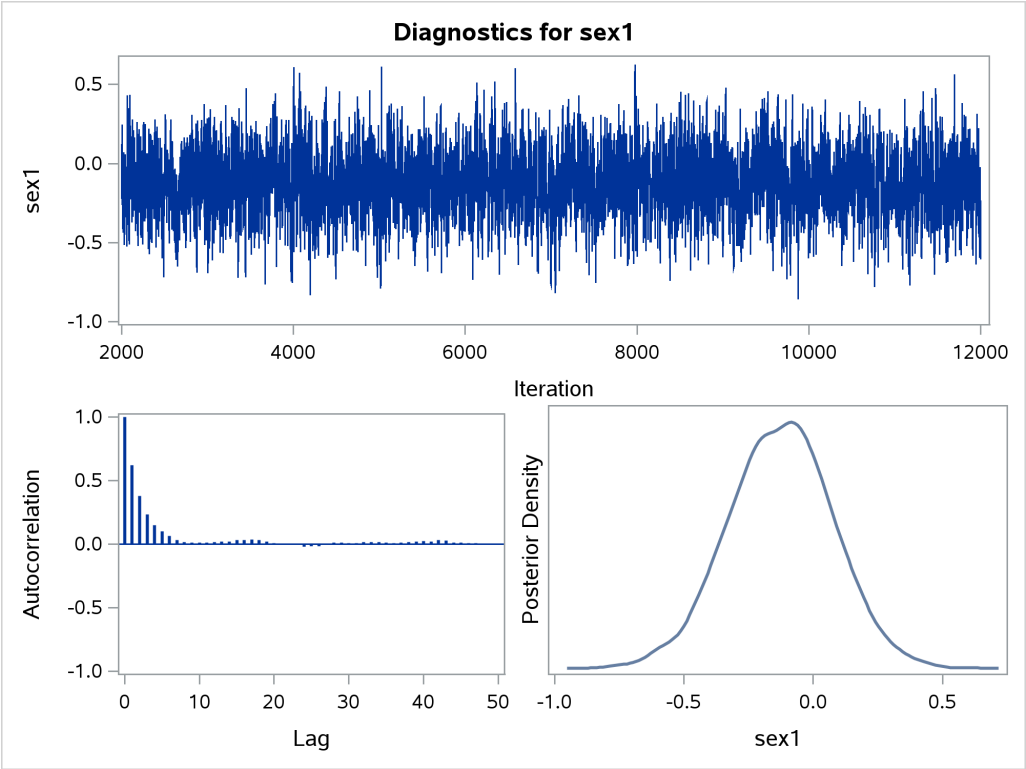

Output 76.7.9 Diagnostic Plots

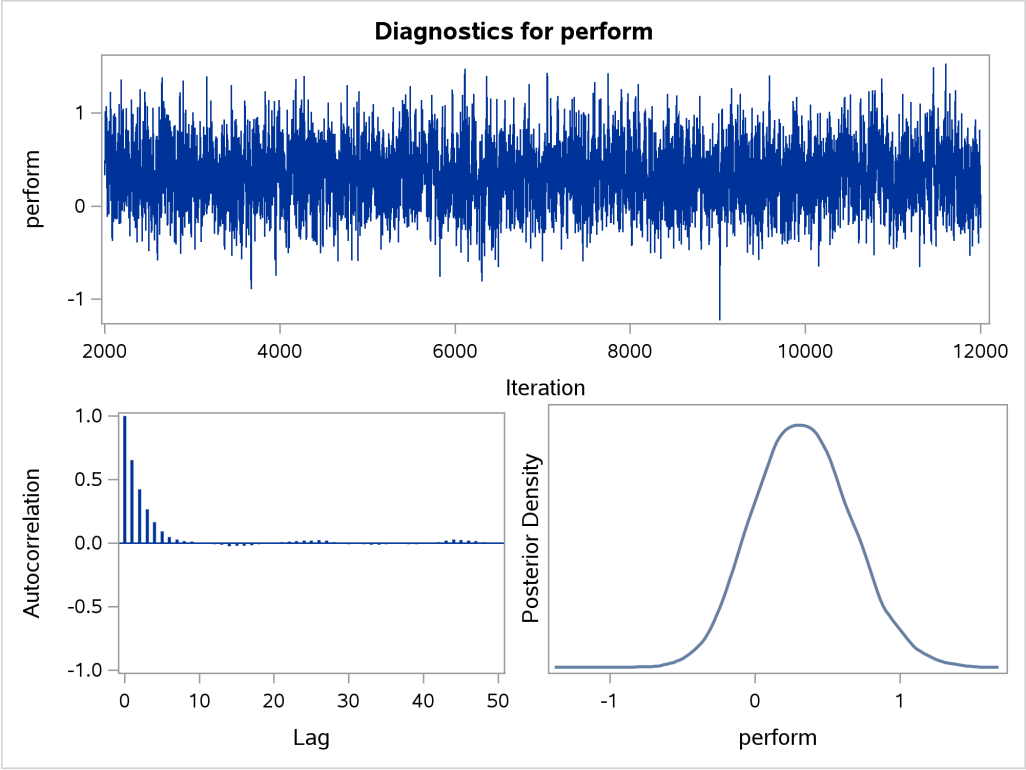

Output 76.7.10 Diagnostic Plots

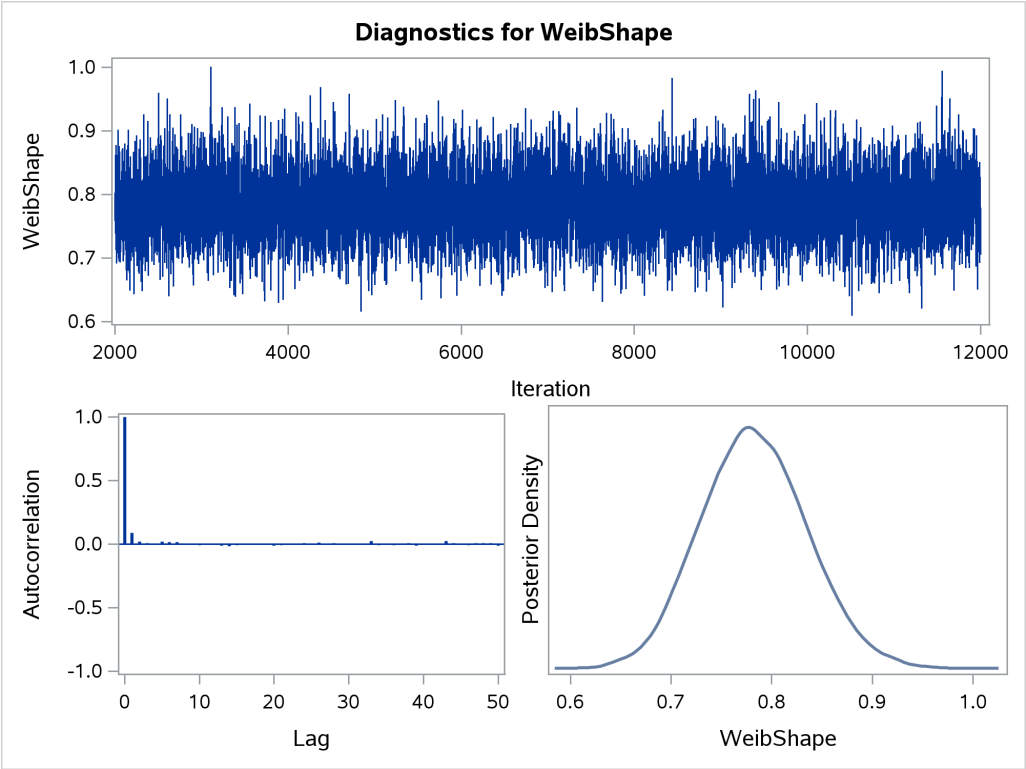

Example 76.8: Model Postfitting Analysis

(View the complete code for this example (liferex8.sas) in the example repository.)

PROC LIFEREG enables you to make model-based inferences. This example uses the larynx cancer data (Klein and Moeschberger 1997) to illustrate usage of the LSMEANS and LSMESTIMATE statements for model postfitting analysis.

The survival time is modeled by a proportional odds model with two covariates: patient age and cancer stage (1, 2, 3, 4). The following statements use PROC LIFEREG to fit this model:

```
ods graphics on;

proc sort data=Larynx;
   by DESCENDING Stage;
run;

proc lifereg data=Larynx order=data;
   class Stage;
   model Time*Death(0) = Age Stage / dist = llogistic;
   lsmeans Stage / diff adjust=Sidak;
run;
```

The LSMEANS statement compares pairwise differences in survival times among the four different cancer stages, while adjusting for age. The ADJUST=SIDAK option uses the Sidak method to control the overall Type I error rate of these comparisons. Results are displayed in Output 76.8.1.

Output 76.8.1 LS-Means Differences between Disease Stages

The LIFEREG Procedure

Differences of Stage Least Squares Means
Adjustment for Multiple Comparisons: Sidak

Stage	_Stage	Estimate	Standard Error	z Value	Pr > |z|	Adj P
4	3	-0.9604	0.4379	-2.19	0.0283	0.1581
4	2	-1.6404	0.4931	-3.33	0.0009	0.0053
4	1	-1.7661	0.4257	-4.15	<.0001	0.0002
3	2	-0.6800	0.4316	-1.58	0.1151	0.5199
3	1	-0.8057	0.3539	-2.28	0.0228	0.1292
2	1	-0.1257	0.4152	-0.30	0.7621	0.9998

All the LS-means differences and their significance are displayed by the mean-mean scatter plot in Output 76.8.2.

Output 76.8.2 Plot of Pairwise LS-Means Differences

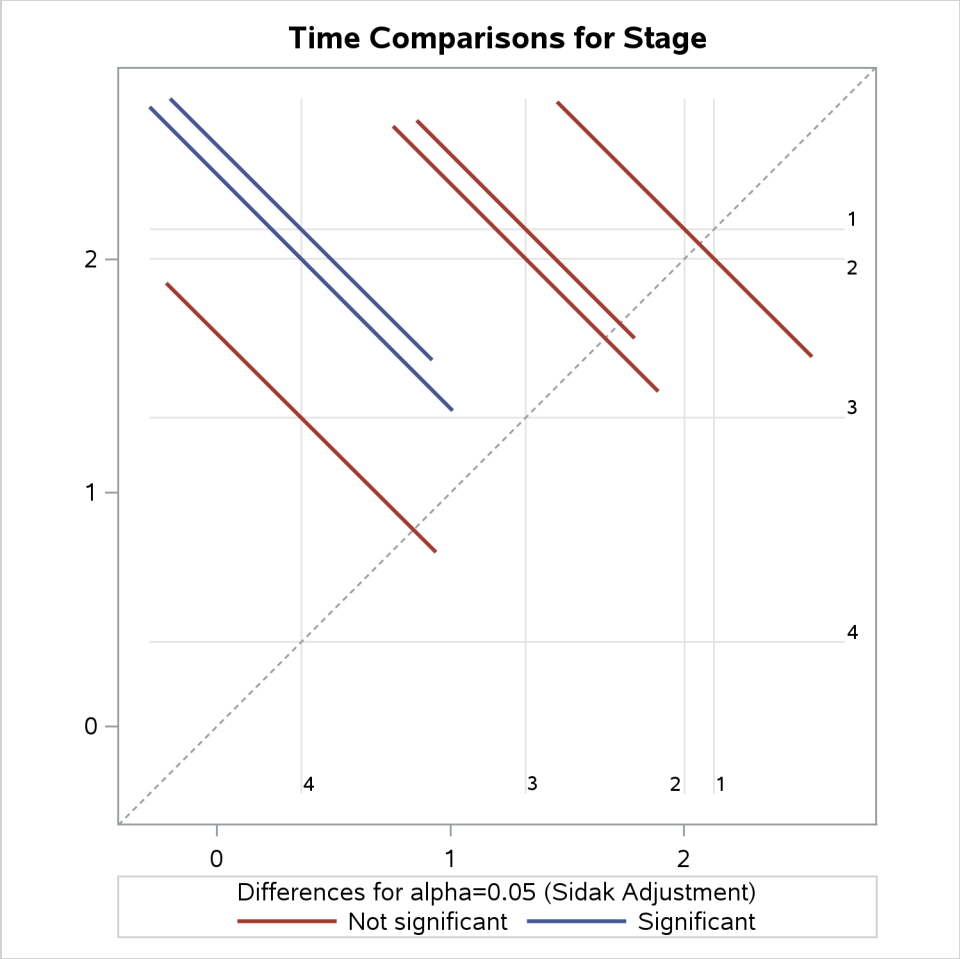

Suppose you want to jointly test whether the effects of stages 2, 3, and 4 are different from stage 1. The following LSMESTIMATE statement contrasts the LS-means of stages 2, 3, and 4 against the LS-means of stage 1:

```
proc lifereg data=Larynx order=data;
   class Stage year;
   model Time*Death(0) = Age Stage / dist = llogistic;
   lsmestimate Stage 'Stage 4 vs 1' 1 0 0 -1,
                     'Stage 3 vs 1' 0 1 0 -1,
                     'Stage 2 vs 1' 0 0 1 -1 / cl adjust=Sidak;
run;
```

The CL option produces 95% confidence limits, including both unadjusted ones and those adjusted for multiple comparisons according to the ADJUST= option. Results are displayed in Output 76.8.3.

Output 76.8.3 Custom LS-Means Tests and Relative Odds

The LIFEREG Procedure

Least Squares Means Estimates
Adjustment for Multiplicity: Sidak

Effect	Label	Estimate	Standard Error	z Value	Pr > |z|	Adj P	Alpha	Lower	Upper	Adj Lower	Adj Upper
Stage	Stage 4 vs 1	-1.7661	0.4257	-4.15	<.0001	0.0001	0.05	-2.6004	-0.9319	-2.7825	-0.7498
Stage	Stage 3 vs 1	-0.8057	0.3539	-2.28	0.0228	0.0668	0.05	-1.4993	-0.1122	-1.6507	0.03921
Stage	Stage 2 vs 1	-0.1257	0.4152	-0.30	0.7621	0.9865	0.05	-0.9395	0.6881	-1.1171	0.8657

You can also perform the preceding analysis for a Bayesian model. The following statements generate posterior samples from a Bayesian model and request an LS-means analysis to compare the stage effects:

```
proc lifereg data=Larynx order=data;
   class Stage;
   model Time*Death(0) = Age Stage / dist = llogistic;
   bayes seed=100 nmc=500 nbi=500 diagnostic=none outpost=OOO;
   lsmeans Stage / diff exp;
   lsmestimate Stage 'Stage 4 vs 1' 1 0 0 -1,
                     'Stage 3 vs 1' 0 1 0 -1,
                     'Stage 2 vs 1' 0 0 1 -1
                        / cl plots=boxplot(orient=horizontal);
run;
```

Because no prior distributions for the regression coefficients were specified, the default uniform improper distributions shown in the “Uniform Prior for Regression Coefficients” table in Output 76.8.4 are used. The specified gamma prior for the scale parameter is also shown in Output 76.8.4.

Output 76.8.4 Model Parameter Priors

The LIFEREG Procedure

Bayesian Analysis

Uniform Prior for Regression Coefficients	
Parameter	**Prior**
Intercept	Constant
Age	Constant
Stage4	Constant
Stage3	Constant
Stage2	Constant

Independent Prior Distributions for Model Parameters					
Parameter	**Prior Distribution**	**Hyperparameters**			
Scale	Gamma	Shape	0.001	Inverse Scale	0.001

Under the Bayesian framework, the LS-means differences are treated as random variables for which posterior samples are readily available according to the linear relationship of LS-means and the regression coefficients. Output 76.8.5 lists the sample mean, standard deviation, and percentiles for each LS-means difference.

Output 76.8.5 LS-Means Differences between Disease Stages

Sample Differences of Stage Least Squares Means												
					Percentiles					Percentiles for Exponentiated		
Stage	**_Stage**	**N**	**Estimate**	**Standard Deviation**	**25th**	**50th**	**75th**	**Exponentiated**	**Standard Error of Exponentiated**	**25th**	**50th**	**75th**
4	3	500	-0.9307	0.4752	-1.2743	-0.9446	-0.6086	0.4426	0.232690	0.2796	0.3888	0.5441
4	2	500	-1.6591	0.5327	-2.0161	-1.6573	-1.2861	0.2181	0.115808	0.1332	0.1907	0.2763
4	1	500	-1.8001	0.4321	-2.0951	-1.7943	-1.5491	0.1815	0.082975	0.1231	0.1663	0.2124
3	2	500	-0.7284	0.4828	-1.0488	-0.7219	-0.3975	0.5410	0.268735	0.3504	0.4858	0.6720
3	1	500	-0.8694	0.3727	-1.1199	-0.8541	-0.6149	0.4488	0.168055	0.3263	0.4257	0.5407
2	1	500	-0.1410	0.4413	-0.4126	-0.1417	0.1363	0.9585	0.462376	0.6619	0.8679	1.1461

The LSMESTIMATE statement produces summary statistics of the posterior samples for the specified LS-means contrasts. Results are presented in Output 76.8.6; they are very similar to the results based on maximum likelihood in Output 76.8.3.

Output 76.8.6 Summary Statistics of Custom LS-Means Differences

Sample Least Squares Means Estimates										
					Percentiles					
Effect	**Label**	**N**	**Estimate**	**Standard Deviation**	**25th**	**50th**	**75th**	**Alpha**	**Lower HPD**	**Upper HPD**
Stage	**Stage 4 vs 1**	500	-1.8001	0.4321	-2.0951	-1.7943	-1.5491	0.05	-2.6279	-0.8897
Stage	**Stage 3 vs 1**	500	-0.8694	0.3727	-1.1199	-0.8541	-0.6149	0.05	-1.6033	-0.2031
Stage	**Stage 2 vs 1**	500	-0.1410	0.4413	-0.4126	-0.1417	0.1363	0.05	-1.1401	0.6252

The PLOTS= option uses ODS Graphics to display the Bayesian samples. A box plot is presented in Output 76.8.7.

Output 76.8.7 Box Plot of Sampled LS-Means Differences

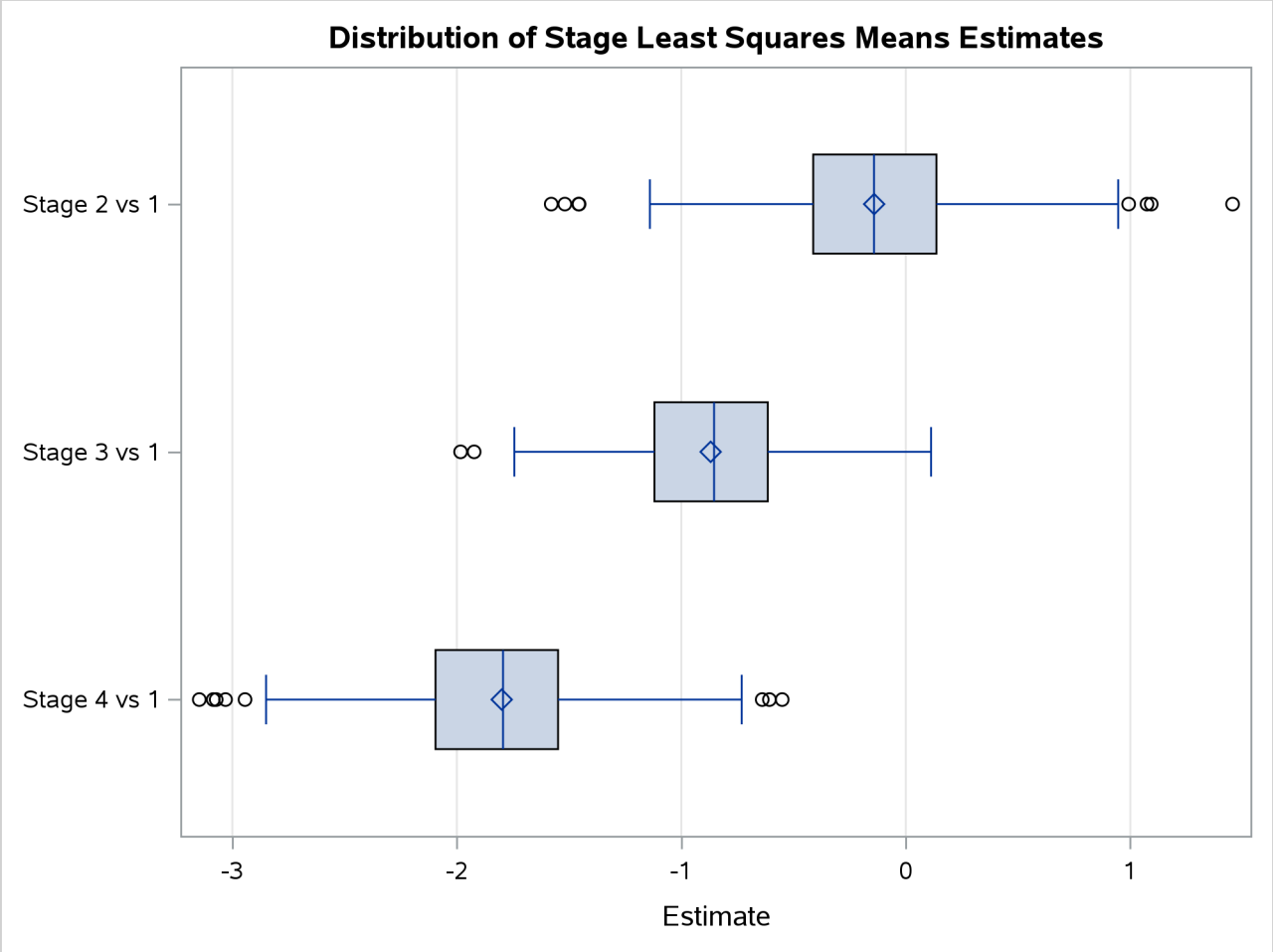

References

Abernethy, R. B. (1996). *The New Weibull Handbook*. 2nd ed. North Palm Beach, FL: Robert B. Abernethy.

Akaike, H. (1979). "A Bayesian Extension of the Minimum AIC Procedure of Autoregressive Model Fitting." *Biometrika* 66:237–242.

Akaike, H. (1981). "Likelihood of a Model and Information Criteria." *Journal of Econometrics* 16:3–14.

Cox, D. R., and Oakes, D. (1984). *Analysis of Survival Data*. London: Chapman & Hall.

Gentleman, R., and Geyer, C. J. (1994). "Maximum Likelihood for Interval Censored Data: Consistency and Computation." *Biometrika* 81:618–623.

Gilks, W. R. (2003). "Adaptive Metropolis Rejection Sampling (ARMS)." Software from MRC Biostatistics Unit, Cambridge, UK. http://www.maths.leeds.ac.uk/~wally.gilks/adaptive.rejection/web_page/Welcome.html.

Gilks, W. R., Best, N. G., and Tan, K. K. C. (1995). "Adaptive Rejection Metropolis Sampling within Gibbs Sampling." *Journal of the Royal Statistical Society, Series C* 44:455–472.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996). *Markov Chain Monte Carlo in Practice*. London: Chapman & Hall.

Gilks, W. R., and Wild, P. (1992). "Adaptive Rejection Sampling for Gibbs Sampling." *Journal of the Royal Statistical Society, Series C* 41:337–348.

Greene, W. H. (1993). *Econometric Analysis*. 2nd ed. New York: Macmillan.

Ibrahim, J. G., Chen, M.-H., and Sinha, D. (2001). *Bayesian Survival Analysis*. New York: Springer-Verlag.

Kalbfleisch, J. D., and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley & Sons.

Klein, J. P., and Moeschberger, M. L. (1997). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: Springer-Verlag.

Lawless, J. F. (2003). *Statistical Model and Methods for Lifetime Data*. 2nd ed. New York: John Wiley & Sons.

Maddala, G. S. (1983). *Limited-Dependent and Qualitative Variables in Econometrics*. New York: Cambridge University Press.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Mroz, T. A. (1987). "The Sensitivity of an Empirical Model of Married Women's Work to Economic and Statistical Assumptions." *Econometrica* 55:765–799.

Nair, V. N. (1984). "Confidence Bands for Survival Functions with Censored Data: A Comparative Study." *Technometrics* 26:265–275.

Nelson, W. (1982). *Applied Life Data Analysis*. New York: John Wiley & Sons.

Nelson, W. (1990). *Accelerated Testing: Statistical Models, Test Plans, and Data Analyses*. New York: John Wiley & Sons.

Rao, C. R. (1973). *Linear Statistical Inference and Its Applications*. 2nd ed. New York: John Wiley & Sons.

Simonoff, J. S. (2003). *Analyzing Categorical Data*. New York: Springer-Verlag.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and Van der Linde, A. (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society, Series B* 64:583–616. With discussion.

Tobin, J. (1958). "Estimation of Relationships for Limited Dependent Variables." *Econometrica* 26:24–36.

Turnbull, B. W. (1976). "The Empirical Distribution Function with Arbitrarily Grouped, Censored, and Truncated Data." *Journal of the Royal Statistical Society, Series B* 38:290–295.

Chapter 77
The LIFETEST Procedure

Contents

Overview: LIFETEST Procedure **5752**
Getting Started: LIFETEST Procedure **5753**
Syntax: LIFETEST Procedure **5761**
PROC LIFETEST Statement 5762
BY Statement 5774
FREQ Statement 5775
ID Statement 5775
STRATA Statement 5775
TEST Statement 5780
TIME Statement 5781
WEIGHT Statement 5782
Details: LIFETEST Procedure **5782**
Missing Values 5782
Computational Formulas 5783
Breslow, Fleming-Harrington, and Kaplan-Meier Methods 5783
Life-Table Method 5786
Pointwise Confidence Limits in the OUTSURV= Data Set 5788
Simultaneous Confidence Intervals for Kaplan-Meier Curves 5789
Kernel-Smoothed Hazard Estimate 5791
Comparison of Two or More Groups of Survival Data 5793
Rank Tests for the Association of Survival Time with Covariates 5798
Analysis of Competing-Risks Data 5799
Restricted Mean Analysis 5802
Output Data Sets 5804
OUTCIF= Data Set 5804
OUTSURV= Data Set 5805
OUTTEST= Data Set 5807
Displayed Output 5807
Plot Options Superseded by ODS Graphics 5817
ODS Table Names 5821
ODS Graphics 5823
Modifying the Survival Plots 5825
Examples: LIFETEST Procedure **5826**
Example 77.1: Product-Limit Estimates and Tests of Association 5826
Example 77.2: Enhanced Survival Plot and Multiple-Comparison Adjustments 5840
Example 77.3: Life-Table Estimates for Males with Angina Pectoris 5845

Example 77.4: Nonparametric Analysis of Competing-Risks Data 5852
Example 77.5: Restricted Mean Analysis 5859
References **5864**

Overview: LIFETEST Procedure

A common feature of lifetime or survival data is the presence of right-censored observations due either to withdrawal of experimental units or to termination of the experiment. For such observations, you know only that the lifetime exceeded a given value; the exact lifetime remains unknown. Such data cannot be analyzed by ignoring the censored observations because, among other considerations, the longer-lived units are generally more likely to be censored. The analysis methodology must correctly use the censored observations in addition to the uncensored observations.

Texts that discuss the survival analysis methodology include Collett (1994), Cox and Oakes (1984); Kalbfleisch and Prentice (1980); Klein and Moeschberger (1997); Lawless (1982); Lee (1992). Users interested in the theory should consult Fleming and Harrington (1991); Andersen et al. (1992).

Usually, a first step in the analysis of survival data is the estimation of the distribution of the survival times. Survival times are often called *failure* times, and *event* times are uncensored survival times. The survival distribution function (SDF), also known as the survivor function, is used to describe the lifetimes of the population of interest. The SDF evaluated at t is the probability that an experimental unit from the population will have a lifetime that exceeds t—that is,

$$S(t) = \Pr(T > t)$$

where $S(t)$ denotes the survivor function and T is the lifetime of a randomly selected experimental unit. The LIFETEST procedure can be used to compute nonparametric estimates of the survivor function either by the product-limit method (also called the Kaplan-Meier method) or by the life-table method (also called the actuarial method). The life-table estimator is a grouped-data analog of the Kaplan-Meier estimator. The procedure can also compute the Breslow estimator or the Fleming-Harrington estimator, which are asymptotic equivalent alternatives to the Kaplan-Meier estimator.

Some functions closely related to the SDF are the cumulative distribution function (CDF), the probability density function (PDF), and the hazard function. The CDF, denoted $F(t)$, is defined as $1 - S(t)$ and is the probability that a lifetime does not exceed t. The PDF, denoted $f(t)$, is defined as the derivative of $F(t)$, and the hazard function, denoted $h(t)$, is defined as $f(t)/S(t)$. If the life-table method is chosen, the estimates of the probability density function can also be computed. Plots of these estimates can be produced with ODS Graphics.

An important task in the analysis of survival data is the comparison of survival curves. It is of interest to determine whether the underlying populations of k ($k \geq 2$) samples have identical survivor functions. PROC LIFETEST provides nonparametric k-sample tests based on weighted comparisons of the estimated hazard rate of the individual population under the null and alternative hypotheses. Corresponding to various weight functions, a variety of tests can be specified, which include the log-rank test, Wilcoxon test, Tarone-Ware test, Peto-Peto test, modified Peto-Peto test, and Fleming-Harrington G_ρ family of tests. PROC LIFETEST also provides corresponding trend tests to detect ordered alternatives. Stratified tests can be specified to adjust for prognostic factors that affect the events rates in the various populations. A likelihood ratio test, based on an underlying exponential model, is also included to compare the survival curves of the samples.

There are other prognostic variables, called covariates, that are thought to be related to the failure time. These covariates can also be used to construct statistics to test for association between the covariates and the lifetime variable. PROC LIFETEST can compute two such test statistics: censored data linear rank statistics based on the exponential scores and the Wilcoxon scores. The corresponding tests are known as the log-rank test and the Wilcoxon test, respectively. These tests are computed by pooling over any defined strata, thus adjusting for the stratum variables.

One change in SAS 9.2 and later is that the calculation of confidence limits for the quartiles of survival time is based on the transformation specified by the CONFTYPE= option. Another change is that the SURVIVAL statement in SAS 9.1 is folded into the PROC LIFETEST statement; that is, options that were in the SURVIVAL statement can now be specified in the PROC LIFETEST statement. The SURVIVAL statement is no longer needed and it is not documented.

Starting in SAS/STAT 14.1, you can use PROC LIFETEST to carry out nonparametric analysis of competing-risks data. Competing risks arise in studies in which individuals are subject to a number of potential failure events and the occurrence of one event might impede the occurrence of other events. You can use PROC LIFETEST to estimate the cumulative incidence function (CIF), which is the probability subdistribution of failure of a specific cause. If you have more than one sample of competing-risks data, you can use PROC LIFETEST to perform Gray's test (Gray 1988) to compare the CIFs of the samples.

Getting Started: LIFETEST Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/liftgs.sas.)

You can use the LIFETEST procedure to compute nonparametric estimates of the survivor functions, to compare survival curves, and to compute rank tests for association of the failure time variable with covariates.

For simple analyses, only the PROC LIFETEST and TIME statements are required. Consider a sample of survival data. Suppose that the time variable is T and the censoring variable is C with value 1 indicating censored observations. The following statements compute the product-limit estimate for the sample:

```
proc lifetest;
   time t*c(1);
run;
```

You can use the STRATA statement to divide the data into various strata. A separate survivor function is then estimated for each stratum, and tests of the homogeneity of strata are performed. However, if the GROUP= option is also specified in the STRATA statement, the GROUP= variable is used to identify the samples whose survivor functions are to be compared, and the STRATA variables are used to define the strata for the stratified tests. You can specify covariates (prognostic variables) in the TEST statement, and PROC LIFETEST computes linear rank statistics to test the effects of these covariates on survival.

For example, consider the results of a small randomized trial on rats. Suppose you randomize 40 rats that have been exposed to a carcinogen into two treatment groups (Drug X and Placebo). The event of interest is death from cancer induced by the carcinogen. The response is the time from randomization to death. Four rats died of other causes; their survival times are regarded as censored observations. Interest lies in whether the survival distributions differ between the two treatments.

The following DATA step creates the data set Exposed, which contains four variables: Days (survival time in days from treatment to death), Status (censoring indicator variable: 0 if censored and 1 if not censored), Treatment (treatment indicator), and Sex (gender: F if female and M if male).

```
proc format;
   value Rx 1='Drug X' 0='Placebo';
run;
data exposed;
   input Days  Status Treatment Sex $ @@;
   format Treatment Rx.;
   datalines;
179   1   1   F   378   0   1   M
256   1   1   F   355   1   1   M
262   1   1   M   319   1   1   M
256   1   1   F   256   1   1   M
255   1   1   M   171   1   1   F
224   0   1   F   325   1   1   M
225   1   1   F   325   1   1   M
287   1   1   M   217   1   1   F
319   1   1   M   255   1   1   F
264   1   1   M   256   1   1   F
237   0   0   F   291   1   0   M
156   1   0   F   323   1   0   M
270   1   0   M   253   1   0   M
257   1   0   M   206   1   0   F
242   1   0   M   206   1   0   F
157   1   0   F   237   1   0   M
249   1   0   M   211   1   0   F
180   1   0   F   229   1   0   F
226   1   0   F   234   1   0   F
268   0   0   M   209   1   0   F
;
```

PROC LIFETEST is invoked as follows to compute the product-limit estimate of the survivor function for each treatment and to compare the survivor functions between the two treatments:

```
ods graphics on;
proc lifetest data=Exposed plots=(survival(atrisk) logsurv);
   time Days*Status(0);
   strata Treatment;
run;
ods graphics off;
```

In the TIME statement, the survival time variable, Days, is crossed with the censoring variable, Status, with the value 0 indicating censoring. That is, the values of Days are considered censored if the corresponding values of Status are 0; otherwise, they are considered as event times. In the STRATA statement, the variable Treatment is specified, which indicates that the data are to be divided into strata based on the values of Treatment. ODS Graphics must be enabled before producing graphs. Two plots are requested through the PLOTS= option—a plot of the survival curves with at risk numbers and a plot of the negative log of the survival curves.

The results of the analysis are displayed in the following figures.

Figure 77.1 displays the product-limit survival estimate for the Drug X group (Treatment=1). The figure lists,

for each observed time, the survival estimate, failure rate, standard error of the estimate, cumulative number of failures, and number of subjects remaining in the study.

Figure 77.1 Survivor Function Estimate for the Drug X-Treated Rats

The LIFETEST Procedure

Stratum 1: Treatment = Drug X

Product-Limit Survival Estimates

Days		Survival	Failure	Survival Standard Error	Number Failed	Number Left
0.000		1.0000	0	0	0	20
171.000		0.9500	0.0500	0.0487	1	19
179.000		0.9000	0.1000	0.0671	2	18
217.000		0.8500	0.1500	0.0798	3	17
224.000	*	.	.	.	3	16
225.000		0.7969	0.2031	0.0908	4	15
255.000		.	.	.	5	14
255.000		0.6906	0.3094	0.1053	6	13
256.000		.	.	.	7	12
256.000		.	.	.	8	11
256.000		.	.	.	9	10
256.000		0.4781	0.5219	0.1146	10	9
262.000		0.4250	0.5750	0.1135	11	8
264.000		0.3719	0.6281	0.1111	12	7
287.000		0.3187	0.6813	0.1071	13	6
319.000		.	.	.	14	5
319.000		0.2125	0.7875	0.0942	15	4
325.000		.	.	.	16	3
325.000		0.1062	0.8938	0.0710	17	2
355.000		0.0531	0.9469	0.0517	18	1
378.000	*	.	.	.	18	0

Note: The marked survival times are censored observations.

Figure 77.2 displays summary statistics of survival times for the Drug X group. It contains estimates of the 25th, 50th, and 75th percentiles and the corresponding 95% confidence limits. The median survival time for rats in this treatment is 256 days. The mean and standard error are also displayed; however, these values are underestimated because the largest observed time is censored and the estimation is restricted to the largest event time.

Figure 77.2 Summary Statistics of Survival Times for Drug X-Treated Rats

Quartile Estimates

			95% Confidence Interval	
Percent	Point Estimate	Transform	[Lower	Upper)
75	319.000	LOGLOG	256.000	355.000
50	256.000	LOGLOG	255.000	319.000
25	255.000	LOGLOG	171.000	256.000

Figure 77.2 *continued*

Mean	Standard Error
271.131	11.877

Note: The mean survival time and its standard error were underestimated because the largest observation was censored and the estimation was restricted to the largest event time.

Figure 77.3 and Figure 77.4 display the survival estimates and the summary statistics of the survival times for Placebo (Treatment=0). The median survival time for rats in this treatment is 235 days.

Figure 77.3 Survivor Function Estimate for Placebo-Treated Rats

The LIFETEST Procedure

Stratum 2: Treatment = Placebo

Product-Limit Survival Estimates

Days		Survival	Failure	Survival Standard Error	Number Failed	Number Left
0.000		1.0000	0	0	0	20
156.000		0.9500	0.0500	0.0487	1	19
157.000		0.9000	0.1000	0.0671	2	18
180.000		0.8500	0.1500	0.0798	3	17
206.000		.	.	.	4	16
206.000		0.7500	0.2500	0.0968	5	15
209.000		0.7000	0.3000	0.1025	6	14
211.000		0.6500	0.3500	0.1067	7	13
226.000		0.6000	0.4000	0.1095	8	12
229.000		0.5500	0.4500	0.1112	9	11
234.000		0.5000	0.5000	0.1118	10	10
237.000		0.4500	0.5500	0.1112	11	9
237.000	*	.	.	.	11	8
242.000		0.3938	0.6063	0.1106	12	7
249.000		0.3375	0.6625	0.1082	13	6
253.000		0.2813	0.7188	0.1038	14	5
257.000		0.2250	0.7750	0.0971	15	4
268.000	*	.	.	.	15	3
270.000		0.1500	0.8500	0.0891	16	2
291.000		0.0750	0.9250	0.0693	17	1
323.000		0	1.0000	.	18	0

Note: The marked survival times are censored observations.

Figure 77.4 Summary Statistics of Survival Times for Placebo-Treated Rats

Quartile Estimates				
			95% Confidence Interval	
Percent	Point Estimate	Transform	[Lower	Upper)
75	257.000	LOGLOG	237.000	.
50	235.500	LOGLOG	206.000	253.000
25	207.500	LOGLOG	156.000	229.000

Mean	Standard Error
235.156	10.211

A summary of the number of censored and event observations is shown in Figure 77.5. The figure lists, for each stratum, the number of event and censored observations, and the percentage of censored observations.

Figure 77.5 Number of Event and Censored Observations

Summary of the Number of Censored and Uncensored Values					
Stratum	Treatment	Total	Failed	Censored	Percent Censored
1	Drug X	20	18	2	10.00
2	Placebo	20	18	2	10.00
Total		40	36	4	10.00

Figure 77.6 displays the graph of the product-limit survivor function estimates versus survival time. The two treatments differ primarily at larger survival times. Note the number of subjects at risk in the plot. You can display the number of subjects at risk at specific time points by using the ATRISK= option.

Figure 77.6 Plot of Estimated Survivor Functions

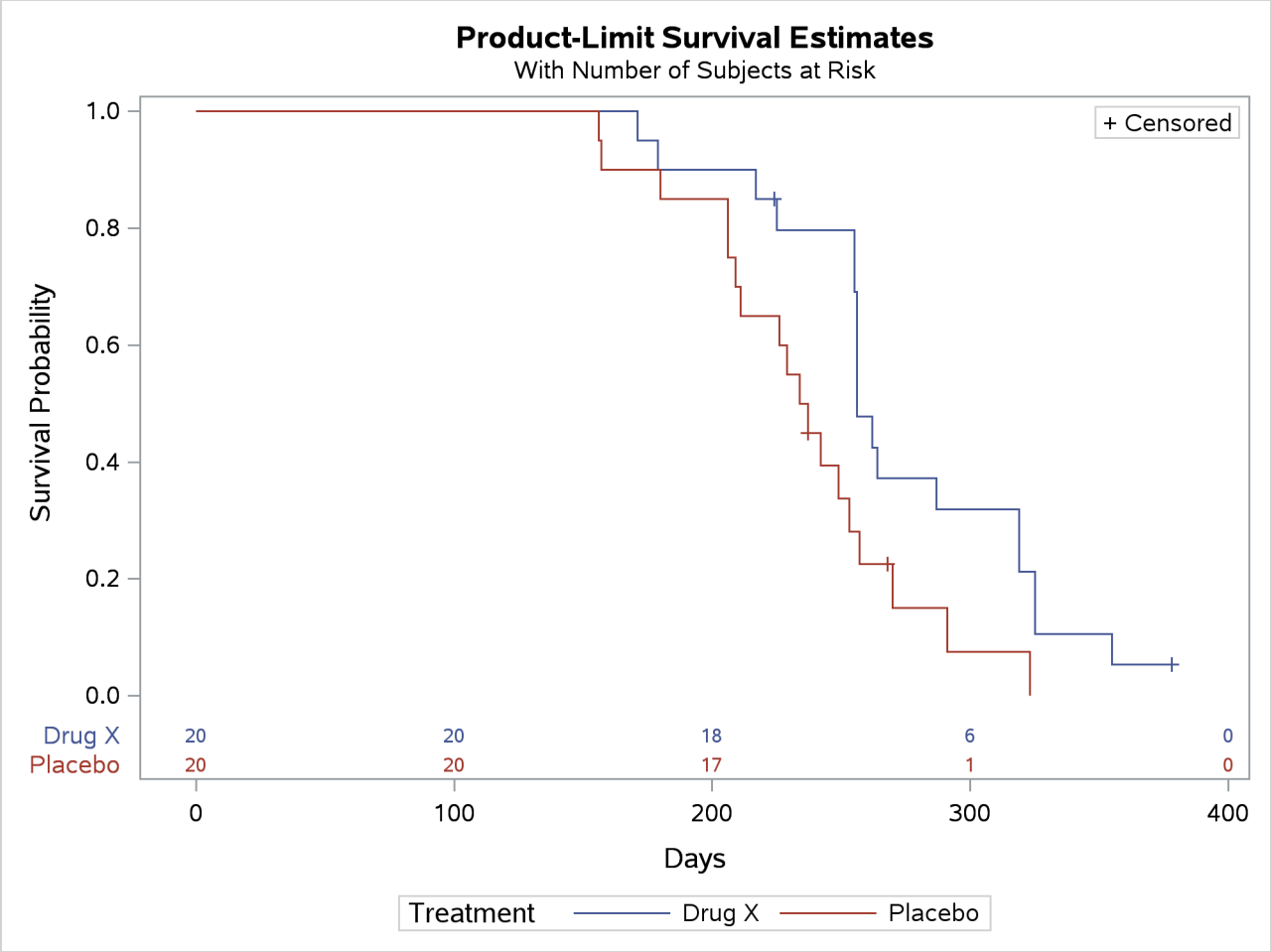

Figure 77.7 displays the graph of the log survivor function estimates versus survival time. Neither curve approximates a straight line through the origin—the exponential model is not appropriate for the survival data.

Note that these graphical displays are generated through ODS. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Figure 77.7 Plot of Estimated Negative Log Survivor Functions

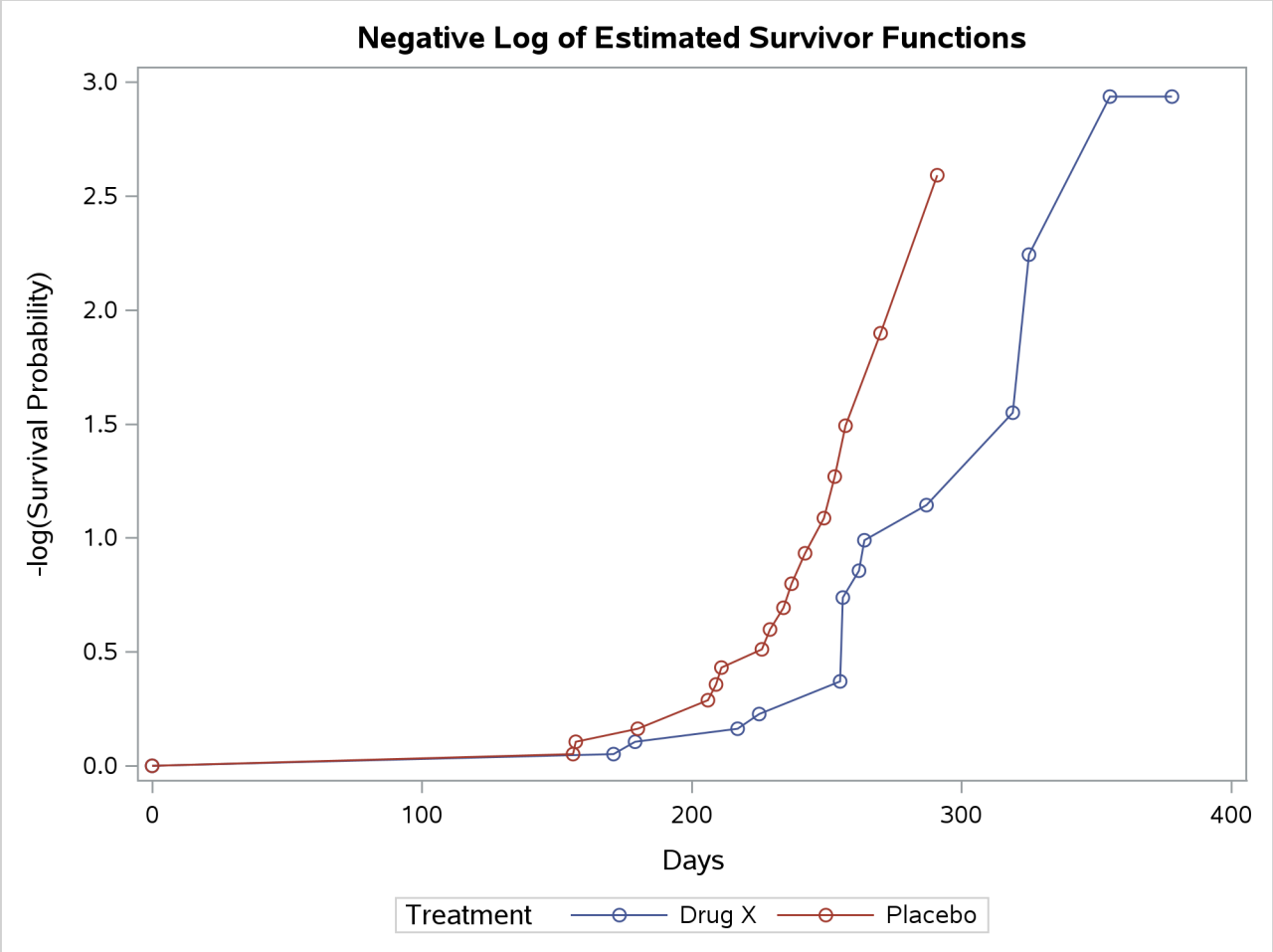

Results of the comparison of survival curves between the two treatments are shown in Figure 77.8. The rank tests for homogeneity indicate a significant difference between the treatments ($p = 0.0175$ for the log-rank test and $p = 0.0249$ for the Wilcoxon test). Rats treated with Drug X live significantly longer than those treated with Placebo. Since the survival curves for the two treatments differ primarily at longer survival times, the Wilcoxon test, which places more weight on shorter survival times, becomes less significant than the log-rank test. As noted earlier, the exponential model is not appropriate for the given survival data; consequently, the result of the likelihood ratio test should be ignored.

Figure 77.8 Results of the Two-Sample Tests

Test of Equality over Strata			
Test	**Chi-Square**	**DF**	**Pr > Chi-Square**
Log-Rank	5.6485	1	0.0175
Wilcoxon	5.0312	1	0.0249
-2Log(LR)	0.1983	1	0.6561

Next, suppose male rats and female rats are thought to have different survival rates, and you want to assess the treatment effect while adjusting for the gender differences. By specifying the variable Sex in the STRATA

statement as a stratifying variable and by specifying the variable Treatment in the GROUP= option, you can carry out a stratified test to test Treatment while adjusting for Sex. The test statistics are computed by pooling over the strata defined by the values of Sex, thus controlling for the effect of Sex. The NOTABLE option is added to the PROC LIFETEST statement as follows to avoid estimating a survival curve for each gender:

```
proc lifetest data=Exposed notable;
   time Days*Status(0);
   strata Sex / group=Treatment;
run;
```

Results of the stratified tests are shown in Figure 77.9. The treatment effect is statistically significant for both the log-rank test ($p = 0.0071$) and the Wilcoxon test ($p = 0.0150$). As compared to the results of the unstratified tests in Figure 77.8, the significance of the treatment effect has been sharpened by controlling for the effect of the gender of the subjects.

Figure 77.9 Results of the Stratified Two-Sample Tests

The LIFETEST Procedure

Stratified Test of Equality over Group			
Test	Chi-Square	DF	Pr > Chi-Square
Log-Rank	7.2466	1	0.0071
Wilcoxon	5.9179	1	0.0150

Since Treatment is a binary variable, another way to study the effect of Treatment is to carry out a censored linear rank test with Treatment as an independent variable. This test is less popular than the two-sample test; nevertheless, in situations where the independent variables are continuous and are difficult to discretize, it might be infeasible to perform a k-sample test. To compute the censored linear rank statistics to test the Treatment effect, Treatment is specified in the TEST statement as follows:

```
proc lifetest data=Exposed notable;
   time Days*Status(0);
   test Treatment;
run;
```

Results of the linear rank tests are shown Figure 77.10. The p-values are very similar to those of the two-sample tests in Figure 77.8.

Figure 77.10 Results of Linear Rank Tests of Treatment

The LIFETEST Procedure

Univariate Chi-Squares for the Wilcoxon Test				
Variable	Test Statistic	Standard Error	Chi-Square	Pr > Chi-Square
Treatment	3.9525	1.7524	5.0875	0.0241

Univariate Chi-Squares for the Log-Rank Test				
Variable	Test Statistic	Standard Error	Chi-Square	Pr > Chi-Square
Treatment	6.2708	2.6793	5.4779	0.0193

With Sex as a prognostic factor that you want to control, you can compute a stratified linear rank statistic to test the effect of Treatment by specifying Sex in the STRATA statement and Treatment in the TEST statement as in the following program. The TEST=NONE option is specified in the STRATA statement to suppress the two-sample tests for Sex.

```
proc lifetest data=Exposed notable;
   time Days*Status(0);
   strata Sex / test=none;
   test Treatment;
run;
```

Results of the stratified linear rank tests are shown in Figure 77.11. The p-values are very similar to those of the stratified tests in Figure 77.9.

Figure 77.11 Results of Stratified Linear Rank Tests of Treatment

The LIFETEST Procedure

Univariate Chi-Squares for the Wilcoxon Test				
Variable	Test Statistic	Standard Error	Chi-Square	Pr > Chi-Square
Treatment	4.2372	1.7371	5.9503	0.0147

Univariate Chi-Squares for the Log-Rank Test				
Variable	Test Statistic	Standard Error	Chi-Square	Pr > Chi-Square
Treatment	6.8021	2.5419	7.1609	0.0075

Syntax: LIFETEST Procedure

The following statements are available in the LIFETEST procedure:

PROC LIFETEST < *options* > ;
  BY *variables* ;
  FREQ *variable* < / *option* > ;
  ID *variables* ;
  STRATA *variable* < (*list*) > < . . . *variable* < (*list*) > > < / *options* > ;
  TEST *variables* ;
  TIME *variable* < ∗ *censor*(*list*) > < / *option* > ;
  WEIGHT *variable* ;

The simplest use of PROC LIFETEST is to request the nonparametric estimates of the survivor function for a sample of survival times. In such a case, only the PROC LIFETEST statement and the TIME statement are required. You can use the STRATA statement to divide the data into various strata. A separate survivor function is then estimated for each stratum, and tests of the homogeneity of strata are performed. However, if the GROUP= option is also specified in the STRATA statement, stratified tests are carried out to test the k samples that are defined by the GROUP= variable while controlling for the effect of the STRATA variables. You can specify covariates in the TEST statement. PROC LIFETEST computes linear rank statistics to test the effects of these covariates on survival.

The PROC LIFETEST statement invokes the procedure. All statements except the TIME statement are optional, and there is no required order for the statements that follow the PROC LIFETEST statement. The TIME statement specifies the variables that define the survival time and censoring indicator. The STRATA statement specifies a variable or set of variables that define the strata for the analysis. The TEST statement specifies a list of numeric covariates to be tested for their association with the response survival time. Each variable is tested individually, and a joint test statistic is also computed. The ID statement provides a list of variables whose values identify observations in the product-limit, Breslow, or Fleming-Harrington estimates. When only the TIME statement appears, no strata are defined and no tests of homogeneity are performed.

PROC LIFETEST Statement

PROC LIFETEST < *options* > **;**

The PROC LIFETEST statement invokes the LIFETEST procedure. Optionally, this statement identifies an input data set and an output data set, and specifies the computation details of the survivor function estimation. Table 77.1 summarizes the *options* available in the PROC LIFETEST statement. These *options* are described in alphabetic order.

ODS Graphics is the preferred method of creating graphs. Many new features have been added to the ODS Graphics plots. For example, you can display the number of subjects at risk in a survival plot. For information about ODS Graphics options, see the PLOTS= option.

If no plotting options are specified, PROC LIFETEST displays a table that shows the product-limit estimate of the survivor function. If ODS Graphics is enabled, PROC LIFETEST also displays a plot of the estimated survivor function. Other *options* for displaying the estimated survivor function are documented in the section "Plot Options Superseded by ODS Graphics" on page 5817.

Table 77.1 Options Available in the PROC LIFETEST Statement

Option	Description
Input and Output Data Sets	
DATA=	Specifies the input SAS data set
OUTCIF=	Names an output data set to contain cumulative incidence function (CIF) estimates
OUTSURV=	Names an output data set to contain survivor function estimates
OUTTEST=	Names an output data set to contain rank test statistics for association of survival time with covariates
Nonparametric Estimation	
ERROR=	Specifies the variance method of the CIF estimator
INTERVALS=	Specifies interval endpoints for life-table estimates
NELSON	Adds the Nelson-Aalen estimates
METHOD=	Specifies the method to compute survivor function
NINTERVAL=	Specifies the number of intervals for life-table estimates
RMST	Performs the restricted mean survival time (RMST) analysis
RMTL	Performs the restricted mean time lost (RMTL) analysis
WIDTH=	Specifies the width of intervals for life-table estimates

Table 77.1 *continued*

Option	Description
Confidence Limits for Survivorship	
ALPHA=	Sets the confidence level for interval estimation estimates
BANDMAXTIME=	Specifies the maximum time for confidence band
BANDMINTIME=	Specifies the minimum time for confidence band
CONFBAND=	Specifies the type of confidence band in the OUTSURV= data set
CONFTYPE=	Specifies the transformation applied to the survivor function to obtain confidence limits
ODS Graphics	
MAXTIME=	Specifies the maximum time value for plotting
PLOTS=	Specifies plots to display
Control Output	
ATRISK	Adds the number of subjects at risk to the survival estimate table
INTERVALS=	Displays only the estimate for the smallest time in each interval
NOLEFT	Suppresses the Number Left and Number Event columns in the survival estimate table
NOPRINT	Suppresses the display of printed output
NOTABLE	Suppresses the display of survival function estimates
REDUCEOUT	Lists only INTERVALS= or TIMELIST= observations in the OUTSURV= data set
TIMELIST=	Lists the time points at which to display the survival estimate
Miscellaneous	
ALPHAQT=	Sets the confidence level for survival time quartiles
CIFVAR	Displays the variance of the CIF estimator
MISSING	Allows missing values to be a stratum level
SINGULAR=	Sets the tolerance for testing singularity of covariance matrix of rank statistics
STDERR	Outputs the standard error for the survival estimators to the OUTSURV= data set
TIMELIM=	Specifies the time limit used to estimate the mean survival time and its standard error

ALPHA=α

specifies the level of significance α for the $100(1-\alpha)\%$ confidence intervals for the survivor, hazard, and density functions. For example, the option ALPHA=0.05 requests the 95% confidence limits for the survivor function. The default value is 0.05.

ALPHAQT=α

specifies the significance level α for the $100(1-\alpha)\%$ confidence intervals for the quartiles of the survival time. For example, the option ALPHAQT=0.05 requests a 95% confidence interval for the quartiles of the survival time. The default value is 0.05.

ATRISK

adds a column that represents the number of subjects at risk to the survival estimate table. Also added is a column that represents the number of events at each observed time. This option has no effect for the life-table method.

BANDMAXTIME=*value*

BANDMAX=*value*

specifies the maximum time for the confidence bands. The default is the largest observed event time. If the specified BANDMAX= time exceeds the largest observed event time, it is truncated to the largest observed event time.

BANDMINTIME=*value*

BANDMIN=*value*

specifies the minimum time for the confidence bands. The default is the smallest observed event time. For the equal-precision band, if the BANDMIN= value is less than the smallest observed event time, it is defaulted to the smallest observed event time.

CIFVAR

displays the variance of the cumulative incidence function (CIF) estimator for competing-risks data. By default, PROC LIFETEST displays the standard error of the CIF estimator.

CONFBAND=*keyword*

specifies the confidence bands to be output to the OUTSURV= data set. Confidence bands are available for METHOD=KM, METHOD=BRESLOW, or METHOD=FH. You can use the following *keywords*:

ALL outputs both the Hall-Wellner and the equal-precision confidence bands.

EP outputs the equal-precision confidence bands.

HW outputs the Hall-Wellner confidence bands.

CONFTYPE=*keyword*

specifies the transformation applied to $S(t)$ to obtain the pointwise confidence intervals and the confidence bands for the survivor function in addition to the confidence intervals for the quartiles of the survival times. The following *keywords* can be used; the default is CONFTYPE=LOGLOG.

ASINSQRT the arcsine-square root transformation,

$$g(x) = \sin^{-1}(\sqrt{x})$$

LOGLOG the log-log transformation,

$$g(x) = \log(-\log(x))$$

This is also referred to as the log cumulative hazard transformation since it applies the logarithmic function to the cumulative hazard function. Collett (1994) and Lachin (2000) refer to it as the complementary log-log transformation.

LINEAR the identity transformation,

$$g(x) = x$$

LOG the logarithmic transformation,

$$g(x) = \log(x)$$

LOGIT the logit transformation,

$$g(x) = \log\left(\frac{x}{1-x}\right)$$

DATA=*SAS-data-set*

names the SAS data set used by PROC LIFETEST. By default, the most recently created SAS data set is used.

ERROR=AALEN | DELTA

specifies the method of calculating the variance of the CIF estimator. When ERROR=AALEN, the variance estimator is based on the theory of counting process (Aalen 1978). When ERROR=DELTA, the delta method is used to compute the variance. By default, ERROR=AALEN. For more information, see the section "Estimation of the CIF" on page 5800.

INTERVALS=*values*

specifies a list of interval endpoints for the life-table method. These endpoints must all be nonnegative numbers. The initial interval is assumed to start at zero whether or not zero is specified in the list. Each interval contains its lower endpoint but does not contain its upper endpoint. When this option is used with METHOD=KM, METHOD=BRESLOW, or METHOD=FH, it reduces the number of survival estimates displayed by showing only the estimates for the smallest time within each specified interval. The INTERVALS= option can be specified in any of the following ways:

- a list separated by blanks `INTERVALS=1 3 5 7`
- a list separated by commas `INTERVALS=1,3,5,7`
- x to y `INTERVALS=1 to 7`
- x to y BY z `INTERVALS=1 to 7 by 1`
- a combination of the above `INTERVALS=1,3 to 5,7`

For example, the specification

```
intervals=5,10 to 30 by 10
```

produces the set of intervals

$$\{[0, 5), [5, 10), [10, 20), [20, 30), [30, \infty)\}$$

MAXTIME=*value*

specifies the maximum value of the time variable allowed on the plots so that outlying points do not determine the scale of the time axis of the plots. This option affects only the displayed plots and has no effect on any calculations.

METHOD=*type*

specifies the method to be used to compute the survival function estimates. Valid values for *type* are as follows:

BRESLOW

specifies that the Breslow estimates be computed. The Breslow estimator is the exponentiation of the negative Nelson-Aalen estimator of the cumulative hazard function.

FH

specifies that the Fleming-Harrington (FH) estimates be computed. The FH estimator is a tie-breaking modification of the Breslow estimator. If there are no tied event times, this estimator is the same as the Breslow estimator.

KM

PL

specifies that Kaplan-Meier estimates (also known as the product-limit estimates) be computed.

ACT

LIFE

LT

specifies that life-table estimates (also known as actuarial estimates) be computed.

By default, METHOD=KM.

MISSING

treats missing values as valid values for the stratum variables. By default, PROC LIFETEST does not use observations that have a missing value in any stratum variables. For more information, see the section "Missing Values" on page 5782.

NELSON

AALEN

produces the Nelson-Aalen estimates of the cumulative hazards and the corresponding standard errors. This option is ignored if METHOD=LT is specified.

NINTERVAL=*value*

specifies the number of intervals used to compute the life-table estimates of the survivor function. This parameter is overridden by the WIDTH= option or the INTERVALS= option. When you specify the NINTERVAL= option, PROC LIFETEST tries to find an interval that results in round numbers for the endpoints. Consequently, the number of intervals can be different from the number requested. Use the INTERVALS= option to control the interval endpoints. The default is NINTERVAL=10.

NOLEFT

suppresses the Number Left and Number Event columns in the survival estimate table. This option has no effect for the life-table estimate.

NOPRINT

suppresses the display of output. This option is useful when only an output data set is needed. It temporarily disables the Output Delivery System (ODS); For more information about ODS, see Chapter 23, "Using the Output Delivery System."

NOTABLE

suppresses the display of survival function estimates. Only the number of censored and event times, plots, and test results is displayed.

OUTCIF=*SAS-data-set*

creates an output SAS data set to contain the point and interval estimates for the cumulative incidence function (CIF). The data set also contains the number of subjects at risk, the number of events of interest, and the number of events of all types. For more information about the contents of the OUTCIF= data set, see the section "OUTCIF= Data Set" on page 5804.

OUTSURV=*SAS-data-set*

OUTS=*SAS-data-set*

creates an output SAS data set to contain the estimates of the survival function and corresponding confidence limits for all strata. For more information about the contents of the OUTSURV= data set, see the section "OUTSURV= Data Set" on page 5805.

OUTTEST=*SAS-data-set*

OUTT=*SAS-data-set*

creates an output SAS data set to contain the overall chi-square test statistic for association with failure time for the variables in the TEST statement, the values of the univariate rank test statistics for each variable in the TEST statement, and the estimated covariance matrix of the univariate rank test statistics. For more information about the contents of the OUTTEST= data set, see the section "OUTTEST= Data Set" on page 5807.

RMST<(*options***)>**

performs the analysis of the restricted mean survival time (RMST). You can specify the following *options*:

BC

applies bias correction to standard error estimation.

TAU=*value*

specifies the upper time limit of the RMST. The *value* must be positive. The default is the largest observed time.

RMTL<(*options***)>**

performs the analysis of the restricted mean time lost (RMTL). You can specify the following *options*:

BC

applies bias correction to standard error estimation.

TAU=*value*
specifies the upper time limit of the RMTL. The *value* must be positive. The default is the largest observed time.

PLOTS<(*global-plot-options***)>=***plot-request* **<(***options***)>**

PLOTS<(*global-plot-options***)>=(***plot-request* **<(***options***)> <...** *plot-request* **<(***options***)>>)**
controls the plots produced using ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. Here are some examples:

```
plots=none
plots=(survival(atrisk=100 to 350 by 50) logsurv)
plots(only)=hazard
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc lifetest plots=survival(atrisk);
   time T*Status(0);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, PROC LIFETEST produces a plot of the survivor function estimates, unless you use the FAILCODE option in the TIME statement to stipulate a competing-risks analysis. In such a case, PROC LIFETEST creates a plot of the cumulative incidence function (CIF) estimates.

You can specify the following *global-plot-option*:

ONLY
specifies that only the specified plots in the list be produced; otherwise, the default survivor function plot is also displayed. This option has no effect if you use the FAILCODE option in the TIME statement to stipulate a competing-risks analysis.

The *plot-requests* and *plot-request options* include the following.

ALL
produces all appropriate plots. For METHOD=KM, METHOD=BRESLOW, or METHOD=FH, specifying PLOTS=ALL is equivalent to specifying PLOTS=(SURVIVAL LOGSURV LOGLOGS); for the life-table method, specifying PLOTS=ALL is equivalent to specifying PLOTS=(SURVIVAL LOGSURV LOGLOGS DENSITY HAZARD). For a competing-risks analysis, specifying PLOTS=ALL is equivalent to specifying PLOTS=CIF.

CIF< (*cif-options*) >

plots the cumulative incidence function (CIF) estimates. If you specify a STRATA statement without the GROUP= option, PROC LIFETEST overlays the cumulative incidence curves of the strata in the same plot. If you specify a STRATA statement with the GROUP= option, PROC LIFETEST produces a panel plot, with one cell per stratum, and each cell contains the cumulative incidence curves for the groups within the given stratum.

You can specify the following *cif-options*:

CL

displays pointwise confidence limits for CIF.

TEST

displays the p-value of Gray's test (Gray 1988) for testing the homogeneity of CIFs.

HAZARD < (*hazard-options*) >

H < *hazard-options* >

plots the estimated hazard functions. Kernel-smoothed estimates are produced for METHOD=KM, METHOD=BRESLOW, or METHOD=FH. You can specify the following *hazard-options*, but only the CL option can be used for the life-table method:

BANDWIDTH=*bandwidth-option*

BW=*bandwidth-option*

specifies what bandwidth is chosen for the kernel-smoothing and how it is chosen. You can specify one of the following *bandwidth-options*.

value

sets the bandwidth to the given *value*.

numeric-list

selects the bandwidth from the given *numeric-list* that minimizes the mean integrated squared error.

RANGE(*lower,upper*)

selects the bandwidth from the interval (*lower, upper*) that minimizes the mean integrated squared error. PROC LIFETEST uses the golden section search algorithm to find the minimum. If there is more than one local minimum in the interval, there is no guarantee that the local minimum found is also the global minimum.

See the section "Optimal Bandwidth" on page 5793 for details about the mean integrated squared error. If the BANDWIDTH= option is not specified, the default is BANDWIDTH= RANGE(0.2b,20b), where $b = \frac{g_u - g_l}{8n^{.2}}$, g_l and g_u are the values of the GRIDL= and GRIDU= options, respectively, and n is the total number of noncensored observations.

CL

displays the pointwise confidence limits for the smoothed hazard.

GRIDL=*number*
specifies the lower grid limit for the kernel-smoothed estimate. The default value is the time origin.

GRIDU=*number*
specifies the upper grid limit for the kernel-smoothed estimate. The default value equals the maximum event time.

KERNEL=*kernel-option*
specifies the kernel used. You can specify the following *kernel-options*:

BIWEIGHT
BW
$K_{\mathrm{BW}}(x) = \frac{15}{16}(1 - x^2)^2, \qquad -1 \le x \le 1$

EPANECHNIKOV
E
$K_{\mathrm{E}}(x) = \frac{3}{4}(1 - x^2), \qquad -1 \le x \le 1$

UNIFORM
U
$K_{\mathrm{U}}(x) = \frac{1}{2}, \qquad -1 \le x \le 1$

By default, KERNEL=EPANECHNIKOV.

NGRID=*number*
specifies the number of grid points. By default, NGRID=101.

NMINGRID=*number*
specifies the number of grid points that are used in determining the mean integrated square error (MISE). By default, NMINGRID=51.

LOGLOGS
LLS
plots the log of negative log of estimated survivor functions versus the log of time.

LOGSURV
LS
plots the negative log of estimated survivor functions versus time.

NONE
suppresses all plots.

PDF <(CL)>
P <(CL)>
plots the estimated probability density functions (life-table method only). Pointwise confidence limits are displayed optionally by specifying the CL option.

RMST< CL >

plots the restricted mean survival time (RMST) versus τ values. If you specify a STRATA statement without the GROUP= option, PROC LIFETEST overlays the RMST curves of the strata in the same plot. If you specify a STRATA statement with the GROUP= option, the procedure produces a panel plot that has one cell per stratum, and each cell contains the RMST curves for the groups within the given stratum. The CL option displays pointwise confidence limits for the RMST.

RMTL< CL >

plots the restricted mean survival time (RMTL) versus τ values. If you specify a STRATA statement without the GROUP= option, PROC LIFETEST overlays the RMTL curves of the strata in the same plot. If you specify a STRATA statement with the GROUP= option, the procedure produces a panel plot that has one cell per stratum, and each cell contains the RMTL curves for the groups within the given stratum. The CL option displays pointwise confidence limits for the RMTL.

SURVIVAL < (*survival-options*) >

S < (*survival-options*) >

plots the estimated survivor functions. Censored times are plotted as a plus sign on the Kaplan-Meier, Breslow, or Fleming-Harrington survival curves unless the NOCENSOR option is specified. You can customize the display by using the following *survival-options*. If these options are not sufficient for your purposes, you can customize the survival plot by modifying its graph template. (For more information, see the section "Modifying the Survival Plots" on page 5825.)

ATRISK < (*options*) > < =*number-list* >

displays the numbers of subjects at risk at the given times. You can specify the following *options*:

ATRISKTICK

ATRISKLABEL

guarantees that tick values are shown on the time axis for those times when the numbers of subjects at risk are displayed. If this option is not specified, you might not be able to tell at exactly which times the number of subjects at risk are displayed. If the ATRISKTICKONLY option is also specified, it takes precedence over the ATRISKTICK option.

ATRISKTICKONLY

specifies that tick values on the time axis be shown only at the times that are given in the ATRISK= *list*. If the ATRISKTICK option is also specified, it is ignored; that is, ATRISKTICKONLY takes precedence over ATRISKTICK.

MAXLEN=*n*

specifies the number of characters *n* that are allowed for displaying the stratum labels. If *n* is greater than or equal to the maximum length of the stratum labels, the stratum labels are used in the at-risk display; otherwise, the stratum numbers are used. The default is MAXLEN=12.

OUTSIDE< (*p*) >
specifies that the at-risk table be drawn outside the plot area. PROC LIFETEST uses a graph template that has a two-row lattice layout. The upper cell displays the survival plot, and the bottom cell displays the at-risk table. You can specify an optional number *p* that represents the fractional proportion of the at-risk table height relative to the overall grid height, but that specification is not necessary. By default, *p* is the preferred row weight in the GTL layout lattice statement that ensures that the plot displays well. If you specify a value of *p* too small for the table to be properly displayed, some of the rows might get cut off.

The *number-list* identifies the times when the numbers at risk are displayed. If the *number-list* is not specified, PROC LIFETEST displays the number of subjects at risk at each default tick value on the time axis of the survival plot.

CB < =*keyword* >
displays the confidence bands (that is, simultaneous confidence intervals) for the survivor functions. You can specify one of the following *keywords*. The default is CB=HW.

ALL
displays both the equal-precision and the Hall-Wellner bands.

EP
displays the equal-precision band.

HW
displays the Hall-Wellner confidence band.

CL
displays the pointwise confidence limits for the survivor functions.

FAILURE
F
changes all the displays for survivor functions to those for the failure functions. For example, if both the FAILURE and CL options are specified, the plot displays the failure curves in addition to the pointwise confidence limits for the failure functions.

NOCENSOR
suppresses the plotting of the censored times on a Kaplan-Meier, Breslow, or Fleming-Harrington survival curve.

STRATA=*strata-option*
specifies how to display the survival/failure curves for multiple strata. This option has no effect if there is only one stratum. You can choose one of the following *strata options*:

INDIVIDUAL
UNPACK
specifies that a separate plot be displayed for each stratum.

OVERLAY
specifies that the survival/failure curves for the strata be overlaid in one plot.

PANEL
specifies that separate plots for the strata be organized into panels of two or four plots, depending on the number of strata.

The default is STRATA=OVERLAY.

TEST
displays the *p*-value of a homogeneity test specified in the STRATA statement. If more than one test is produced, the test is chosen in the following order: LOGRANK, WILCOXON, TARONE, PETO, MODPETO, FLEMING, and LR.

REDUCEOUT
specifies that the OUTSURV= data set contain only those observations that are included in the INTERVALS= or TIMELIST= option. This option has no effect if the OUTSURV= option is not specified. It also has no effect if neither the INTERVALS= option nor the TIMELIST= option is specified.

SINGULAR=*value*
specifies the tolerance for testing singularity of the covariance matrix for the rank test statistics. The test requires that a pivot for sweeping a covariance matrix be at least this number times a norm of the matrix. The default value is 1E–12.

STDERR
specifies that the standard error of the survivor function (SDF_STDERR) be output to the OUTSURV= data set. If the life-table method is used, the standard error of the density function (PDF_STDERR) and the standard error of the hazard function (HAZ_STDERR) are also output.

TIMELIM=*time-limit*
specifies the time limit used in the estimation of the mean survival time and its standard error. The mean survival time can be shown to be the area under the Kaplan-Meier survival curve. However, if the largest observed time in the data is censored, the area under the survival curve is not a closed area. In such a situation, you can choose a time limit L and estimate the mean survival curve limited to a time L (Lee 1992, pp. 72–76). This option is ignored if the largest observed time is an event time. Valid *time-limit* values are as follows:

EVENT
LET
specifies that the time limit L be the largest event time in the data. TIMELIM=EVENT is the default.

OBSERVED
LOT
specifies that the time limit L be the largest observed time in the data.

number
specifies that the time limit L be the given *number*. The *number* must be positive and at least as large as the largest event time in the data.

TIMELIST=*number-list*
specifies a list of time points at which the Kaplan-Meier estimates are displayed. The time points are listed in the column labeled Timelist. Since the Kaplan-Meier survival curve is a decreasing step function, each given time point falls in an interval that has a constant survival estimate. The event time that corresponds to the beginning of the time interval is displayed along with its survival estimate.

WIDTH=*value*
sets the width of the intervals used in the life-table calculation of the survival function. This parameter is overridden by the INTERVALS= option.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC LIFETEST to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the LIFETEST procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

The BY statement is more efficient than the STRATA statement for defining strata in large data sets. However, if you use the BY statement to define strata, PROC LIFETEST does not pool over strata for testing the association of survival time with covariates, nor does it test for homogeneity across the BY groups.

When the life-table method is used to estimate survivor functions, each BY group might have a different set of intervals. To make intervals the same across BY groups, use the INTERVALS= or WIDTH= option in the PROC LIFETEST statement.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* < / *option* > **;**

The FREQ statement identifies a *variable* that contains the frequency of occurrence of each observation. PROC LIFETEST treats each observation as if it appeared n times, where n is the value of the FREQ variable for the observation. The FREQ statement is useful for producing life tables when the data are already in the form of a summary data set. If it is not an integer, it is truncated to an integer unless the NOTRUNCATE option is specified. If it is missing or less than or equal zero, the observation is not used.

The following *option* can be specified in the FREQ statement after a slash (/):

NOTRUNCATE

NOTRUNC

specifies that the frequency values are not truncated to integers. This option does not apply to the Fleming-Harrington estimator (METHOD=FH).

ID Statement

ID *variables* **;**

The ID statement identifies *variables* whose values are used to label the observations of the Kaplan-Meier, Breslow, or Fleming-Harrington survivor function estimates. SAS format statements can be used to format the values of the ID variables.

STRATA Statement

STRATA *variable* < **(***list***)** > < . . . *variable* < **(***list***)** > > < / *options* > **;**

The STRATA statement identifies the variables that determine the strata levels. Strata are formed according to the nonmissing values of these variables. The MISSING option can be used to allow missing values as a valid stratum level. Other options enable you to specify various k-sample tests, stratified tests, or trend tests and to make multiple-comparison adjustments for paired differences.

In the preceding syntax, *variable* is a variable whose values determine the stratum levels, and *list* is a list of endpoints for a numeric variable. The values for *variable* can be formatted or unformatted. If *variable* is a character variable, or if *variable* is numeric and no list appears, then the strata are defined by the unique values of the STRATA *variable*. More than one *variable* can be specified in the STRATA statement, and each numeric variable can be followed by a list. Each interval contains its lower endpoint but not its upper endpoint. The corresponding strata are formed by the combination of levels. If a variable is numeric and is followed by a list, then the levels for that variable correspond to the intervals defined by the list. The initial interval is assumed to start at $-\infty$, and the final interval is assumed to end at ∞.

The specification of a STRATA *variable* can have any of the following forms:

- a list separated by blanks `Age(5 10 20 30)`
- a list separated by commas `Age(5,10,20,30)`
- x to y `Age(5 to 10)`
- x to y by z `Age(5 to 30 by 10)`
- a combination of the above `Age(5,10 to 50 by 10)`

For example, the specification

```
strata Age(5,20 to 50 by 10) Sex;
```

indicates the following levels for the Age variable:

$$\{(-\infty, 5), [5, 20), [20, 30), [30, 40), [40, 50), [50, \infty)\}$$

This statement also specifies that the Age strata be further subdivided by values of the variable Sex. In this example, there are six age groups by two sex groups, forming a total of 12 strata.

The specification of several STRATA *variables*, such as

```
strata A B C;
```

is equivalent to the A*B*C syntax of the TABLES statement in the FREQ procedure. The number of strata levels usually grows very rapidly with the number of STRATA variables, so you must be cautious when specifying the list of STRATA variables.

When comparing more than two survival curves, a k-sample test tells you whether the curves are significantly different from each other, but it does not identify which pairs of curves are different. A multiple-comparison adjustment of the p-values for the paired comparisons retains the same overall false positives as the k-sample test. Two types of paired comparisons can be made: comparisons between all pairs of curves and comparisons between a control curve and all other curves. You use the DIFF= option to specify the comparison type, and you use the ADJUST= option to select a method of multiple-comparison adjustments.

Table 77.2 summarizes the *options* available in the STRATA statement.

Table 77.2 Options Available in the STRATA Statement

Option	Description
Homogeneity Tests	
GROUP=	Specifies the group variable for stratified tests
NODETAIL	Suppresses printing the test statistic and covariance matrix
NOTEST	Suppresses any tests
TEST=	Specifies tests corresponding to various weight functions
TREND	Requests a trend test
Multiple Comparisons	
ADJUST=	Requests a multiple-comparison adjustment
DIFF=	Specifies the type of differences to consider
Missing Strata Value	
MISSING	Allows missing values as valid stratum values
Display Option	
NOLABEL	Uses the names of the STRATA variables in the display

You can specify the following *options* in the STRATA statement after a slash ("/").

ADJUST=*method*

ADJ=*method*

specifies the multiple-comparison method for adjusting the p-values of the paired tests. See the section "Multiple-Comparison Adjustments" on page 5796 for mathematical details; also see Westfall et al. (1999). The adjustment methods include the following:

BONFERRONI

BON

applies the Bonferroni correction to the raw p-values.

DUNNETT

performs Dunnett's two-tailed comparisons of the control group with all other groups. PROC LIFETEST uses the factor-analytic covariance approximation described in Hsu (1992) and identifies the adjustment in the results as "Dunnett-Hsu." Note that ADJUST=DUNNETT is incompatible with DIFF=ALL.

SCHEFFE

performs Scheffé's multiple-comparison adjustment.

SIDAK

applies the Šidák correction to the raw p-values.

SMM

GTE

performs the paired comparisons based on the studentized maximum modulus test.

TUKEY

performs the paired comparisons based on Tukey's studentized range test. PROC LIFETEST uses the approximation described in Kramer (1956) and identifies the adjustment as "Tukey-Kramer" in the results. Note that ADJUST=TUKEY is incompatible with DIFF=CONTROL.

SIMULATE < (*simulate-options***) >**

computes the adjusted p-values from the simulated distribution of the maximum or maximum absolute value of a multivariate normal random vector. The simulation estimates q, the true $(1-\alpha)$ quantile, where α is the value of the ALPHA= *simulate-option*.

The number of samples for the SIMULATE adjustment is set so that the tail area for the simulated q is within a certain accuracy radius γ of $1-\alpha$ with an accuracy confidence of $100(1-\epsilon)\%$. In equation form,

$$\Pr(|F(\hat{q}) - (1-\alpha)| \leq \gamma) \quad = \quad 1-\epsilon$$

where $\hat{q}$ is the simulated q and F is the true distribution function of the maximum; see Edwards and Berry (1987) for details. By default, $\gamma = 0.005$ and $\epsilon = 0.01$ so that the tail area of $\hat{q}$ is within 0.005 of 0.95 with 99% confidence.

The *simulate-options* include the following:

ACC=*value*
specifies the target accuracy radius γ of a $100(1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. The default value is ACC=0.005.

ALPHA=*value*
specifies the value α for estimating the $(1-\alpha)$ quantile. The default value is the ALPHA= value in the PROC LIFETEST statement, or 0.05 if that option is not specified.

EPS=*value*
specifies the value ϵ for a $100(1-\epsilon)\%$ confidence interval for the true probability content of the estimated $(1-\alpha)$ quantile. The default value for the accuracy confidence is 99%, corresponding to EPS=0.01.

NSAMP=*n*
specifies the sample size for the simulation. By default, *n* is set based on the values of the target accuracy radius γ and accuracy confidence $100(1-\epsilon)\%$ for an interval for the true probability content of the estimated $(1-\alpha)$ quantile. With the default values for γ, ϵ, and α (0.005, 0.01, and 0.05, respectively), NSAMP=12604 by default.

REPORT
specifies that a report on the simulation should be displayed, including a listing of the parameters, such as γ, ϵ, and α, in addition to an analysis of various methods for estimating or approximating the quantile.

SEED=*number*
specifies an integer used to start the pseudorandom number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated by default from reading the time of day from the computer's clock.

DIFF=ALL | CONTROL<(*'string'* < . . . , *'string'*>**)>**
specifies which pairs of survival curves are considered for the multiple comparisons.

DIFF=ALL
requests all paired comparisons

DIFF=CONTROL <(*'string'* < . . . *'string'*>**)>**
requests comparisons of the control curve with all other curves. To specify the control curve, you specify the quotes strings of formatted values that represent the curve in parentheses. For example, if Cell='large' identifies the control group, you specify

```
DIFF=CONTROL('large')
```

If more than one variable is used to identify the curves (for example, if Cell='large' and Sex='F' represent the control), you specify

```
DIFF=CONTROL('large' 'F')
```

The order of the quoted strings should correspond to the order of the stratum variables. If no specific curve is specified as the control, the first stratum or group value is used.

By default, DIFF=ALL unless you specify ADJUST= DUNNETT, in which case DIFF=CONTROL.

GROUP=*variable*

stipulates a stratified test. You specify the *variable* to identify the groups whose survivor functions or cumulative incidence functions you want to compare. Tests are stratified on the levels of the STRATA variables. For example, in a multicenter trial in which two forms of therapy are to be compared, you specify the variable that identifies therapies as the GROUP= *variable* and the variable that identifies centers as the STRATA variable:

```
proc lifetest;
   time T*Status(0);
   strata Center / group=Therapy;
run;
```

With this specification, PROC LIFETEST performs a stratified test to compare the therapies while controlling the effect of the centers.

The GROUP= option has a side effect on the estimation of the survivor function or the cumulative incidence function (CIF). Instead of estimating a survivor function (or CIF) for each stratum, PROC LIFETEST estimates a survivor function (or CIF) for each group within a stratum. Suppose there are 10 centers and two therapies. The preceding PROC LIFETEST specification estimates 20 survivor functions: two for each center, and one for each therapy for each center.

If the GROUP= option is not specified, PROC LIFETEST performs a homogeneity test comparing the strata.

MISSING

allows missing values to be a stratum level or a valid value of the GROUP= variable.

NODETAIL

suppresses the display of the rank statistics and the corresponding covariance matrices for various strata. If you specified the TREND option, the display of the scores for computing the trend tests is suppressed.

NOLABEL

specifies that the names instead of the labels of the STRATA variables be used in the display of the survival estimate table and in the legend of the survival plot.

NOTEST

suppresses the k-sample tests, stratified tests, and trend tests.

ORDER=FORMATTED | INTERNAL

specifies the sorting order of the values of the STRATA variables. The strata are presented in the specified order in the analysis results. You can use this option, for example, to display the curve labels in your preferred order in the survival plot legend (see Example 77.2 for an illustration). The default is ORDER=FORMATTED, which sorts the strata according to their external formatted values, except for numeric variable with no explicit format, which are sorted by the unformatted (internal) values. ORDER=INTERNAL sorts the strata by their internal values. The ORDER= option has no effect on a stratum variable with cutpoints specified.

TREND

computes the trend tests for testing the null hypothesis that the k population hazards rate are the same versus an ordered alternatives. If there is only one STRATA variable and the variable is numeric, the unformatted values of the variable are used as the scores; otherwise, the scores are $1, 2, \ldots,$ in the given order of the strata.

TEST=*test-request* | **(***test-request* < . . . *test-request*>**)**

controls the tests produced. Each test corresponds to a different weight function (see the section "Nonparametric Tests" on page 5794 for the weight functions). The *test-requests* include the following:

ALL	specifies all the nonparametric tests with $\rho 1$=1 and $\rho 2$=0 for the Fleming and Harrington test—FLEMING(1,0).
FLEMING($\rho 1$, $\rho 2$)	specifies the family of tests in Harrington and Fleming (1982), where $\rho 1$ and $\rho 2$ are nonnegative numbers. FLEMING($\rho 1$,$\rho 2$) reduces to the Fleming-Harrington G^{ρ} family (Fleming and Harrington 1981) when $\rho 2$=0, which you can specify as FLEMING(ρ) with one argument. When ρ=0, the test becomes the log-rank test. When ρ=1, the test should be very close to the Peto-Peto test.
LOGRANK	specifies the log-rank test.
NONE	suppresses all comparison tests. Specifying TEST=NONE is equivalent to specify NOTEST.
LR	specifies the likelihood ratio test based on the exponential model.
MODPETO	specifies the modified Peto-Peto test.
PETO	specifies the Peto-Peto test. The test is also referred to as the Peto-Peto-Prentice test.
WILCOXON	specifies the Wilcoxon test. The test is also referred to as the Gehan test or the Breslow test.
TARONE	specifies the Tarone-Ware test.

By default, TEST=(LOGRANK WILCOXON LR) for the k-sample tests, and TEST=(LOGRANK WILCOXON) for stratified and trend tests.

TEST Statement

TEST *variables* **;**

The TEST statement specifies a list of numeric covariates (prognostic variables) that you want tested for association with the failure time.

Two sets of rank statistics are computed. These rank statistics and their variances are pooled over all strata. Univariate (marginal) test statistics are displayed for each of the covariates.

Additionally, a sequence of test statistics for joint effects of covariates is displayed. The first element of the sequence is the largest univariate test statistic. Other variables are then added on the basis of the largest increase in the joint test statistic. The process continues until all the variables have been added or until the remaining variables are linearly dependent on the previously added variables.

For more information, see the section "Rank Tests for the Association of Survival Time with Covariates" on page 5798.

TIME Statement

TIME *variable* < ∗ *censor*(*list*) > < / *option* > ;

The TIME statement is required. It is used to indicate the failure time variable, where *variable* is the name of the failure time variable that can be optionally followed by an asterisk, the name of the censoring variable, and a parenthetical list of values that correspond to right-censoring. The censoring values should be numeric, nonmissing values. For example, the following statement identifies the variable T as containing the observed failure times (event or censored):

```
time T*Status(0,2);
```

If the variable Status has the value 0 or 2, the corresponding value of T is a right-censored value.

You can specify the following *option* after a slash (/):

FAILCODE< =*number-list* >

EVENTCODE< =*number-list* >

stipulates a competing-risks analysis, which consists of estimating cumulative incidence functions and computing Gray's test (Gray 1988) for testing the homogeneity of two or more cumulative incidence functions. You specify a number that represents the event of interest after the equal sign. For example:

```
proc lifetest;
   time T*Status(0) / failcode=1;
run;
```

For this specification, PROC LIFETEST regards a Status value of 1 as the event of interest, a value of 0 as a censored observation indicator, and all other values as competing events.

You can specify a list of values after the equal sign. PROC LIFETEST performs a separate competing-risks analysis for each value, regarding it as representing the event of interest. For example:

```
proc lifetest;
   time T*Status(0) / failcode=1 2;
run;
```

This specification produces two analyses, one for FAILCODE=1 and the other for FAILCODE=2.

If you specify the FAILCODE option without the equal sign, PROC LIFETEST produces a separate analysis for each distinct event value. Consider a data set with an event indicator variable Status that assumes four distinct values, 0, 1, 2, and 3, where Status=0 represents observations that are censored and Status=1, Status=2, and Status=3 represent three different causes of failure. Consider the following statements:

```
proc lifetest;
   time T*Status(0) / failcode;
run;
```

PROC LIFETEST produces three separate competing-risks analyses: one uses Status=1 as the failure cause of interest, one uses Status=2 as the failure cause of interest, and one uses Status=3 as the failure cause of interest. This specification is convenient for an exploratory analysis when there is no predetermined failure cause of interest.

WEIGHT Statement

WEIGHT *variable* **;**

The *variable* in the WEIGHT statement identifies the variable in the input data set that contains the weights of the subjects. Values of the WEIGHT *variable* can be nonintegral and are not truncated. Observations with negative, zero, or missing values for the WEIGHT *variable* are not used in the computation.

The implementation of weights in PROC LIFETEST is based on Xie and Liu (2005, 2011), who use inverse probability of treatment weights to reduce confounding effects. A weight is assigned to each subject as the inverse probability of being in a certain group. If a subject has a higher probability of being in a group, it is considered as overrepresented and is therefore assigned a lower weight; on the other hand, if the subject has a smaller probability of being in a group, it is considered as underrepresented and is assigned a higher weight.

You can use the WEIGHT statement only for the Kaplan-Meier curves. Other methods of estimating the survival curves using weights are not available.

Details: LIFETEST Procedure

Missing Values

Observations with a missing value for either the failure time or the censoring variable are not used in the analysis. If a stratum variable value is missing, the observation is not used; however, the MISSING option can be used to request that missing values be treated as valid stratum values. If any variable specified in the TEST statement has a missing value, that observation is not used in the calculation of the rank statistics.

Computational Formulas

Breslow, Fleming-Harrington, and Kaplan-Meier Methods

Let $t_1 < t_2 < \cdots < t_D$ represent the distinct event times. For each $i = 1, \ldots, D$, let Y_i be the number of surviving units (the size of the risk set) just prior to t_i and let d_i be the number of units that fail at t_i. If the NOTRUNCATE option is specified in the FREQ statement, Y_i and d_i can be nonintegers.

The Breslow estimate of the survivor function is

$$\hat{S}(t_i) = \exp\left(-\sum_{j=1}^{i} \frac{d_j}{Y_j}\right)$$

Note that the Breslow estimate is the exponentiation of the negative Nelson-Aalen estimate of the cumulative hazard function.

The Fleming-Harrington estimate (Fleming and Harrington 1984) of the survivor function is

$$\hat{S}(t_i) = \exp\left(-\sum_{k=1}^{i} \sum_{j=0}^{d_k-1} \frac{1}{Y_k - j}\right)$$

If the frequency values are not integers, the Fleming-Harrington estimate cannot be computed.

The Kaplan-Meier (product-limit) estimate of the survivor function at t_i is the cumulative product

$$\hat{S}(t_i) = \prod_{j=1}^{i} \left(1 - \frac{d_j}{Y_j}\right)$$

Notice that all the estimators are defined to be right continuous; that is, the events at t_i are included in the estimate of $S(t_i)$. The corresponding estimate of the standard error is computed using Greenwood's formula (Kalbfleisch and Prentice 1980) as

$$\hat{\sigma}\left(\hat{S}(t_i)\right) = \hat{S}(t_i)\sqrt{\sum_{j=1}^{i} \frac{d_j}{Y_j(Y_j - d_j)}}$$

The first quartile (or the 25th percentile) of the survival time is the time beyond which 75% of the subjects in the population under study are expected to survive. It is estimated by

$$q_{.25} = \min\{t_j | \hat{S}(t_j) < 0.75\}$$

If $\hat{S}(t)$ is exactly equal to 0.75 from t_j to t_{j+1}, the first quartile is taken to be $(t_j + t_{j+1})/2$. If it happens that $\hat{S}(t)$ is greater than or equal to 0.75 for all values of t, the first quartile cannot be estimated and is represented by a missing value in the printed output.

The general formula for estimating the 100pth percentile point is

$$q_p = \min\{t_j | \hat{S}(t_j) < 1 - p\}$$

The second quartile (the median) and the third quartile of survival times correspond to $p = 0.5$ and $p = 0.75$, respectively.

Brookmeyer and Crowley (1982) have constructed the confidence interval for the median survival time based on the confidence interval for the $S(t)$. The methodology is generalized to construct the confidence interval for the 100pth percentile based on a g-transformed confidence interval for $S(t)$ (Klein and Moeschberger 1997). You can use the CONFTYPE= option to specify the g-transformation. The $100(1-\alpha)\%$ confidence interval for the first quantile survival time is the set of all points t that satisfy

$$\left|\frac{g(\hat{S}(t)) - g(1-0.25)}{g'(\hat{S}(t))\hat{\sigma}(\hat{S}(t))}\right| \leq z_{1-\frac{\alpha}{2}}$$

where $g'(x)$ is the first derivative of $g(x)$ and $z_{1-\frac{\alpha}{2}}$ is the $(100(1-\frac{\alpha}{2}))$th percentile of the standard normal distribution.

Consider the bone marrow transplant data described in Example 77.2. The following table illustrates the construction of the confidence limits for the first quartile in the ALL group. Values of $\frac{g(\hat{S}(t))-g(1-0.25)}{g'(\hat{S}(t))\hat{\sigma}(\hat{S}(t))}$ that lie between $\pm z_{1-\frac{0.05}{2}} = \pm\ 1.965$ are highlighted.

Constructing 95% Confidence Limits for the 25th Percentile

			$\frac{g(\hat{S}(t))-g(1-0.25)}{g'(\hat{S}(t))\hat{\sigma}(\hat{S}(t))}$				
t	$\hat{S}(t)$	$\hat{\sigma}(\hat{S}(t))$	**LINEAR**	**LOGLOG**	**LOG**	**ASINSQRT**	**LOGIT**
1	0.97368	0.025967	8.6141	2.37831	9.7871	4.44648	2.47903
55	0.94737	0.036224	5.4486	2.36375	6.1098	3.60151	2.46635
74	0.92105	0.043744	3.9103	2.16833	4.3257	2.94398	2.25757
86	0.89474	0.049784	2.9073	**1.89961**	3.1713	2.38164	1.97023
104	0.86842	0.054836	2.1595	**1.59196**	2.3217	**1.87884**	**1.64297**
107	0.84211	0.059153	**1.5571**	**1.26050**	**1.6490**	**1.41733**	**1.29331**
109	0.81579	0.062886	**1.0462**	**0.91307**	**1.0908**	**0.98624**	**0.93069**
110	0.78947	0.066135	**0.5969**	**0.55415**	**0.6123**	**0.57846**	**0.56079**
122	0.73684	0.071434	**–0.1842**	**–0.18808**	**–0.1826**	**–0.18573**	**–0.18728**
129	0.71053	0.073570	**–0.5365**	**–0.56842**	**–0.5222**	**–0.54859**	**–0.56101**
172	0.68421	0.075405	**–0.8725**	**–0.95372**	**–0.8330**	**–0.90178**	**–0.93247**
192	0.65789	0.076960	**–1.1968**	**–1.34341**	**–1.1201**	**–1.24712**	**–1.30048**
194	0.63158	0.078252	**–1.5133**	**–1.73709**	**–1.3870**	**–1.58613**	**–1.66406**
230	0.60412	0.079522	**–1.8345**	–2.14672	**–1.6432**	**–1.92995**	–2.03291
276	0.57666	0.080509	–2.1531	–2.55898	**–1.8825**	–2.26871	–2.39408
332	0.54920	0.081223	–2.4722	–2.97389	–2.1070	–2.60380	–2.74691
383	0.52174	0.081672	–2.7948	–3.39146	–2.3183	–2.93646	–3.09068
418	0.49428	0.081860	–3.1239	–3.81166	–2.5177	–3.26782	–3.42460
466	0.46682	0.081788	–3.4624	–4.23445	–2.7062	–3.59898	–3.74781
487	0.43936	0.081457	–3.8136	–4.65971	–2.8844	–3.93103	–4.05931
526	0.41190	0.080862	–4.1812	–5.08726	–3.0527	–4.26507	–4.35795
609	0.38248	0.080260	–4.5791	–5.52446	–3.2091	–4.60719	–4.64271
662	0.35306	0.079296	–5.0059	–5.96222	–3.3546	–4.95358	–4.90900

Consider the LINEAR transformation where $g(x) = x$. The event times that satisfy $\left|\frac{g(\hat{S}(t))-g(1-p)}{g'(\hat{S}(t))\sqrt{\hat{V}(\hat{S}(t))}}\right| \leq 1.9599$ include 107, 109, 110, 122, 129, 172, 192, 194, and 230. The confidence of the interval [107, 230] is less than 95%. Brookmeyer and Crowley (1982) suggest extending the confidence interval to but not

including the next event time. As such the 95% confidence interval for the first quartile based on the linear transform is [107, 276). The following table lists the confidence intervals for the various transforms.

95% CI's for the 25th Percentile		
CONFTYPE	**[Lower**	**Upper)**
LINEAR	107	276
LOGLOG	86	230
LOG	107	332
ASINSQRT	104	276
LOGIT	104	230

Sometimes, the confidence limits for the quartiles cannot be estimated. For convenience of explanation, consider the linear transform $g(x) = x$. If the curve that represents the upper confidence limits for the survivor function lies above 0.75, the upper confidence limit for first quartile cannot be estimated. On the other hand, if the curve that represents the lower confidence limits for the survivor function lies above 0.75, the lower confidence limit for the quartile cannot be estimated.

The estimated mean survival time is

$$\hat{\mu} = \sum_{i=1}^{D} \hat{S}(t_{i-1})(t_i - t_{i-1})$$

where t_0 is defined to be zero. When the largest observed time is censored, this sum underestimates the mean. The standard error of $\hat{\mu}$ is estimated as

$$\hat{\sigma}(\hat{\mu}) = \sqrt{\frac{m}{m-1} \sum_{i=1}^{D-1} \frac{d_i A_i^2}{Y_i(Y_i - d_i)}}$$

where

$$\begin{aligned} A_i &= \sum_{j=i}^{D-1} \hat{S}(t_j)(t_{j+1} - t_j) \\ m &= \sum_{j=1}^{D} d_j \end{aligned}$$

If the largest observed time is not an event, you can use the TIMELIM= option to specify a time limit L and estimate the mean survival time limited to the time L and its standard error by replacing D by $D + 1$ with $t_{D+1} = L$.

Nelson-Aalen Estimate of the Cumulative Hazard Function

The Nelson-Aalen cumulative hazard estimator, defined up to the largest observed time on study, is

$$\tilde{H}(t) = \sum_{t_i \le t} \frac{d_i}{Y_i}$$

and its estimated variance is

$$\hat{\sigma}^2\left(\tilde{H}(t)\right) = \sum_{t_i \le t} \frac{d_i}{Y_i^2}$$

Adjusted Kaplan-Meier Estimate

PROC LIFETEST computes the adjusted Kaplan-Meier estimate (AKME) of the survivor function if you specify both METHOD=KM and the WEIGHT statement. Let $(T_i, \delta_i, w_i), i = 1, \ldots, n$, denote an independent sample of right-censored survival data, where T_i is the possibly right-censored time, δ_i is the censoring indicator ($\delta_i = 0$ if T_i is censored and $\delta_i = 1$ if T_i is an event time), and w_i is the weight (from the WEIGHT statement). Let $t_1 < t_2 < \cdots < t_D$ be the D distinct event times in the sample. At time $t_j, j = 1, \ldots, D$, there are $d_j = \sum_i \delta_i I(T_i = t_j)$ events out of $Y_j = \sum_i I(T_i \ge t_j)$ subjects. The weighted number of events and the weighted number at risk are $d_j^w = \sum_i w_i \delta_i I(T_i = t_j)$ and $Y_j^w = \sum_i w_i I(T_i \ge t_j)$, respectively. The AKME (Xie and Liu 2005) is

$$\hat{S}(t) = \begin{cases} 1 & \text{if } t < t_1 \\ \prod_{t_j \le t}\left[1 - \frac{d_j^w}{Y_j^w}\right] & \text{if } t \ge t_1 \end{cases}$$

The estimated variance of $\hat{S}(t)$ is

$$\hat{\sigma}^2\left(\hat{S}(t)\right) = \left(\hat{S}(t)\right)^2 \sum_{j:t_j \le t} \frac{d_j^w / Y_j^w}{M_j(1 - d_j^w / Y_j^w)}$$

where

$$M_j = \frac{\left(\sum_{i:T_i \ge t_j} w_i\right)^2}{\sum_{i:T_i \ge t_j} w_i^2}$$

Life-Table Method

The life-table estimates are computed by counting the numbers of censored and uncensored observations that fall into each of the time intervals $[t_{i-1}, t_i)$, $i = 1, 2, \ldots, k+1$, where $t_0 = 0$ and $t_{k+1} = \infty$. Let n_i be the number of units that enter the interval $[t_{i-1}, t_i)$, and let d_i be the number of events that occur in the interval. Let $b_i = t_i - t_{i-1}$, and let $n_i' = n_i - w_i/2$, where w_i is the number of units censored in the interval. The effective sample size of the interval $[t_{i-1}, t_i)$ is denoted by n_i'. Let t_{mi} denote the midpoint of $[t_{i-1}, t_i)$.

The conditional probability of an event in $[t_{i-1}, t_i)$ is estimated by

$$\hat{q}_i = \frac{d_i}{n_i'}$$

and its estimated standard error is

$$\hat{\sigma}\left(\hat{q}_i\right) = \sqrt{\frac{\hat{q}_i \hat{p}_i}{n_i'}}$$

where $\hat{p}_i = 1 - \hat{q}_i$.

The estimate of the survival function at t_i is

$$\hat{S}(t_i) = \begin{cases} 1 & i = 0 \\ \hat{S}(t_{i-1})p_{i-1} & i > 0 \end{cases}$$

and its estimated standard error is

$$\hat{\sigma}\left(\hat{S}(t_i)\right) = \hat{S}(t_i)\sqrt{\sum_{j=1}^{i-1} \frac{\hat{q}_j}{n'_j \hat{p}_j}}$$

The density function at t_{mi} is estimated by

$$\hat{f}(t_{mi}) = \frac{\hat{S}(t_i)\hat{q}_i}{b_i}$$

and its estimated standard error is

$$\hat{\sigma}\left(\hat{f}(t_{mi})\right) = \hat{f}(t_{mi})\sqrt{\sum_{j=1}^{i-1} \frac{\hat{q}_j}{n'_j \hat{p}_j} + \frac{\hat{p}_i}{n'_i \hat{q}_i}}$$

The estimated hazard function at t_{mi} is

$$\hat{h}(t_{mi}) = \frac{2\hat{q}_i}{b_i(1 + \hat{p}_i)}$$

and its estimated standard error is

$$\hat{\sigma}\left(\hat{h}(t_{mi})\right) = \hat{h}(t_{mi})\sqrt{\frac{1 - (b_i \hat{h}(t_{mi})/2)^2}{n'_i \hat{q}_i}}$$

Let $[t_{j-1}, t_j)$ be the interval in which $\hat{S}(t_{j-1}) \geq \hat{S}(t_i)/2 > \hat{S}(t_j)$. The median residual lifetime at t_i is estimated by

$$\hat{M}_i = t_{j-1} - t_i + b_j \frac{\hat{S}(t_{j-1}) - \hat{S}(t_i)/2}{\hat{S}(t_{j-1}) - \hat{S}(t_j)}$$

and the corresponding standard error is estimated by

$$\hat{\sigma}(\hat{M}_i) = \frac{\hat{S}(t_i)}{2\hat{f}(t_{mj})\sqrt{n'_i}}$$

Interval Determination

If you want to determine the intervals exactly, use the INTERVALS= option in the PROC LIFETEST statement to specify the interval endpoints. Use the WIDTH= option to specify the width of the intervals, thus indirectly determining the number of intervals. If neither the INTERVALS= option nor the WIDTH= option is specified in the life-table estimation, the number of intervals is determined by the NINTERVAL= option. The width of the time intervals is 2, 5, or 10 times an integer (possibly a negative integer) power of 10.

Let $c = \log_{10}$(maximum observed time/number of intervals), and let b be the largest integer not exceeding c. Let $d = 10^{c-b}$ and let

$$a = 2 \times I(d \leq 2) + 5 \times I(2 < d \leq 5) + 10 \times I(d > 5)$$

with I being the indicator function. The width is then given by

$$\text{width} = a \times 10^b$$

By default, NINTERVAL=10.

Pointwise Confidence Limits in the OUTSURV= Data Set

Pointwise confidence limits are computed for the survivor function, and for the density function and hazard function when the life-table method is used. Let α be specified by the ALPHA= option. Let $z_{\alpha/2}$ be the critical value for the standard normal distribution. That is, $\Phi(-z_{\alpha/2}) = \alpha/2$, where Φ is the cumulative distribution function of the standard normal random variable.

Survivor Function

When the computation of confidence limits for the survivor function $S(t)$ is based on the asymptotic normality of the survival estimator $\hat{S}(t)$, the approximate confidence interval might include impossible values outside the range [0,1] at extreme values of t. This problem can be avoided by applying the asymptotic normality to a transformation of $S(t)$ for which the range is unrestricted. In addition, certain transformed confidence intervals for $S(t)$ perform better than the usual linear confidence intervals (Borgan and Liestøl 1990). The CONFTYPE= option enables you to pick one of the following transformations: the log-log function (Kalbfleisch and Prentice 1980), the arcsine-square root function (Nair 1984), the logit function (Meeker and Escobar 1998), the log function, and the linear function.

Let g be the transformation that is being applied to the survivor function $S(t)$. By the delta method, the standard error of $g(\hat{S}(t))$ is estimated by

$$\tau(t) = \hat{\sigma}\left[g(\hat{S}(t))\right] = g'\left(\hat{S}(t)\right)\hat{\sigma}[\hat{S}(t)]$$

where g' is the first derivative of the function g. The 100(1–α)% confidence interval for $S(t)$ is given by

$$g^{-1}\left\{g[\hat{S}(t)] \pm z_{\frac{\alpha}{2}} g'[\hat{S}(t)]\hat{\sigma}[\hat{S}(t)]\right\}$$

where g^{-1} is the inverse function of g. That choices of the transformation g are as follows:

- arcsine-square root transformation: The estimated variance of $\sin^{-1}\left(\sqrt{\hat{S}(t)}\right)$ is $\hat{\tau}^2(t) = \frac{\hat{\sigma}^2[\hat{S}(t)]}{4\hat{S}(t)[1-\hat{S}(t)]}$. The 100(1–$\alpha$)% confidence interval for $S(t)$ is given by

$$\sin^2\left\{\max\left[0, \sin^{-1}(\sqrt{\hat{S}(t)}) - z_{\frac{\alpha}{2}}\hat{\tau}(t)\right]\right\} \leq S(t) \leq \sin^2\left\{\min\left[\frac{\pi}{2}, \sin^{-1}(\sqrt{\hat{S}(t)}) + z_{\frac{\alpha}{2}}\hat{\tau}(t)\right]\right\}$$

- linear transformation: This is the same as having no transformation in which g is the identity. The 100(1–α)% confidence interval for $S(t)$ is given by

$$\hat{S}(t) - z_{\frac{\alpha}{2}}\hat{\sigma}\left[\hat{S}(t)\right] \leq S(t) \leq \hat{S}(t) + z_{\frac{\alpha}{2}}\hat{\sigma}\left[\hat{S}(t)\right]$$

- log transformation: The estimated variance of $\log(\hat{S}(t))$ is $\hat{\tau}^2(t) = \frac{\hat{\sigma}^2(\hat{S}(t))}{\hat{S}^2(t)}$. The 100(1–$\alpha$)% confidence interval for $S(t)$ is given by

$$\hat{S}(t)\exp\left(-z_{\frac{\alpha}{2}}\hat{\tau}(t)\right) \le S(t) \le \hat{S}(t)\exp\left(z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)$$

- log-log transformation: The estimated variance of $\log(-\log(\hat{S}(t))$ is $\hat{\tau}^2(t) = \frac{\hat{\sigma}^2[\hat{S}(t)]}{[\hat{S}(t)\log(\hat{S}(t))]^2}$. The 100(1–$\alpha$)% confidence interval for $S(t)$ is given by

$$\left[\hat{S}(t)\right]^{\exp\left(z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)} \le S(t) \le \left[\hat{S}(t)\right]^{\exp\left(-z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)}$$

- logit transformation: The estimated variance of $\log\left(\frac{\hat{S}(t)}{1-\hat{S}(t)}\right)$ is

$$\hat{\tau}^2(t) = \frac{\hat{\sigma}^2(\hat{S}(t))}{\hat{S}^2(t)[1-\hat{S}(t)]^2}.$$

  The 100(1–α)% confidence limits for $S(t)$ are given by

$$\frac{\hat{S}(t)}{\hat{S}(t) + \left[1-\hat{S}(t)\right]\exp\left(z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)} \le S(t) \le \frac{\hat{S}(t)}{\hat{S}(t) + \left[1-\hat{S}(t)\right]\exp\left(-z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)}$$

Density and Hazard Functions

For the life-table method, a 100(1–α)% confidence interval for hazard function or density function at time t is computed as

$$\hat{g}(t) \pm z_{\alpha/2}\hat{\sigma}[\hat{g}(t)]$$

where $\hat{g}(t)$ is the estimate of either the hazard function or the density function at time t, and $\hat{\sigma}[\hat{g}(t)]$ is the corresponding standard error estimate.

Simultaneous Confidence Intervals for Kaplan-Meier Curves

The pointwise confidence interval for the survivor function $S(t)$ is valid for a single fixed time at which the inference is to be made. In some applications, it is of interest to find the upper and lower confidence bands that guarantee, with a given confidence level, that the survivor function falls within the band for all t in some interval. Hall and Wellner (1980) and Nair (1984) provide two different approaches for deriving the confidence bands. An excellent review can be found in Klein and Moeschberger (1997). You can use the CONFBAND= option in the PROC LIFETEST statement to select the confidence bands. The EP confidence band provides confidence bounds that are proportional to the pointwise confidence interval, while those of the HW band are not proportional to the pointwise confidence bounds. The maximum time, t_U, for the bands can be specified by the BANDMAX= option; the minimum time, t_L, can be specified by the BANDMIN= option. Transformations that are used to improve the pointwise confidence intervals can be applied to improve the confidence bands. It might turn out that the upper and lower bounds of the confidence bands are not decreasing in $t_L < t < t_U$, which is contrary to the nonincreasing characteristic of survivor function. Meeker and Escobar (1998) suggest making an adjustment so that the bounds do not increase: if the upper bound is increasing on the right, it is made flat from the minimum to t_U; if the lower bound is increasing from

the right, it is made flat from t_L to the maximum. PROC LIFETEST does not make any adjustment for the nondecreasing behavior of the confidence bands in the OUTSURV= data set. However, the adjustment was made in the display of the confidence bands by using ODS Graphics.

For Kaplan-Meier estimation, let $t_1 < t_2 < \cdots < t_D$ be the D distinct events times, and at time t_i, there are d_i events. Let Y_i be the number of individuals who are at risk at time t_i. The variance of $\hat{S}(t)$, given by the Greenwood formula, is $\hat{\sigma}^2[\hat{S}(t)] = \sigma_S^2(t)\hat{S}^2(t)$, where

$$\sigma_S^2(t) = \sum_{t_i \le t} \frac{d_i}{Y_i(Y_i - d_i)}$$

Let $t_L < t_U$ be the time range for the confidence band so that t_U is less than or equal to the largest event time. For the Hall-Wellner band, t_L can be zero, but for the equal-precision band, t_L is greater than or equal to the smallest event time. Let

$$a_L = \frac{n\sigma_S^2(t_L)}{1 + n\sigma_S^2(t_L)} \text{ and } a_U = \frac{n\sigma_S^2(t_U)}{1 + n\sigma_S^2(t_U)}$$

Let $\{W^0(u), 0 \le u \le 1\}$ be a Brownian bridge.

Hall-Wellner Band

The 100(1–α)% HW band of Hall and Wellner (1980) is

$$\hat{S}(t) - h_\alpha(a_L, a_U)n^{-\frac{1}{2}}[1 + n\sigma_S^2(t)]\hat{S}(t) \le S(t) \le \hat{S}(t) + h_\alpha(a_L, a_U)n^{-\frac{1}{2}}[1 + n\sigma_S^2(t)]\hat{S}(t)$$

for all $t_L \le t \le t_U$, where the critical value $h_\alpha(a_L, a_U)$ is given by

$$\alpha = \Pr\{\sup_{a_L \le u \le a_U} |W^0(u)| > h_\alpha(a_L, a_U)\}$$

The critical values are computed from the results in Chung (1986).

Note that the given confidence band has a formula similar to that of the (linear) pointwise confidence interval, where $h_\alpha(a_L, a_U)$ and $n^{-\frac{1}{2}}[1 + n\sigma_S^2(t)]\hat{S}(t)$ in the former correspond to $z_{\frac{\alpha}{2}}$ and $\hat{\sigma}(\hat{S}(t))$ in the latter, respectively. You can obtain the other transformations (arcsine-square root, log-log, log, and logit) for the confidence bands by replacing $z_{\frac{\alpha}{2}}$ and $\hat{\tau}(t)$ in the corresponding pointwise confidence interval formula by $h_\alpha(a_L, a_U)$ and the following $\hat{\tau}(t)$, respectively:

- arcsine-square root transformation:

$$\hat{\tau}(t) = \frac{1 + n\sigma_S^2(t)}{2}\sqrt{\frac{S(t)}{n[1 - S(t)]}}$$

- log transformation:

$$\hat{\tau}(t) = \frac{1 + n\sigma_S^2(t)}{\sqrt{n}}$$

- log-log transformation:

$$\hat{\tau}(t) = \frac{1 + n\sigma_S^2(t)}{\sqrt{n}|\log[\hat{S}(t)]|}$$

- logit transformation:

$$\hat{\tau}(t) = \frac{1 + n\sigma_S^2(t)}{\sqrt{n}[1 - \hat{S}(t)]}$$

Equal-Precision Band

The 100(1–α)% EP band of Nair (1984) is

$$\hat{S}(t) - e_\alpha(a_L, a_U)\hat{S}(t)\sigma_S(t) \le S(t) \le \hat{S}(t) + e_\alpha(a_L, a_U)\hat{S}(t)\sigma_S(t)$$

for all $t_L \le t \le t_U$, where $e_\alpha(a_L, a_U)$ is given by

$$\alpha = \Pr\{\sup_{a_L \le u \le a_U} \frac{|W^0(u)|}{[u(1-u)]^{\frac{1}{2}}} > e_\alpha(a_L, a_U)\}$$

PROC LIFETEST uses the approximation of Miller and Siegmund (1982, Equation 8) to approximate the tail probability in which $e_\alpha(a_L, a_U)$ is obtained by solving x in

$$\frac{4x\phi(x)}{x} + \phi(x)\left(x - \frac{1}{x}\right)\log\left[\frac{a_U(1-a_L)}{a_L(1-a_U)}\right] = \alpha$$

where $\phi(x)$ is the standard normal density function evaluated at x. Note that the confidence bounds given are proportional to the pointwise confidence intervals. As a matter of fact, this confidence band and the (linear) pointwise confidence interval have the same formula except for the critical values ($z_{\frac{\alpha}{2}}$ for the pointwise confidence interval and $e_\alpha(a_L, a_U)$ for the band). You can obtain the other transformations (arcsine-square root, log-log, log, and logit) for the confidence bands by replacing $z_{\frac{\alpha}{2}}$ by $e_\alpha(a_L, a_U)$ in the formula of the pointwise confidence intervals.

Kernel-Smoothed Hazard Estimate

Kernel-smoothed estimators of the hazard function $h(t)$ are based on the Nelson-Aalen estimator $\tilde{H}(t)$ and its variance $\hat{V}(\tilde{H}(t))$. Consider the jumps of $\tilde{H}(t)$ and $\hat{V}(\tilde{H}(t))$ at the event times $t_1 < t_2 < \cdots < t_D$ as follows:

$$\begin{aligned} \Delta\tilde{H}(t_i) &= \tilde{H}(t_i) - \tilde{H}(t_{i-1}) \\ \hat{V}(\tilde{H}(t_i)) &= \hat{V}(\tilde{H}(t_i)) - \hat{V}(\tilde{H}(t_{i-1})) \end{aligned}$$

where t_0=0.

The kernel-smoothed estimator of $h(t)$ is a weighted average of $\Delta\tilde{H}(t)$ over event times that are within a bandwidth distance b of t. The weights are controlled by the choice of kernel function, $K()$, defined on the interval [–1,1]. The choices are as follows:

- uniform kernel:

$$K_U(x) = \frac{1}{2}, \quad -1 \le x \le 1$$

- Epanechnikov kernel:

$$K_E(x) = \frac{3}{4}(1 - x^2), \quad -1 \le x \le 1$$

- biweight kernel:

$$K_{BW}(x) = \frac{15}{16}(1 - x^2)^2, \quad -1 \le x \le 1$$

The kernel-smoothed hazard rate estimator is defined for all time points on $(0, t_D)$. For time points t for which $b \le t \le t_D - b$, the kernel-smoothed estimated of $h(t)$ based on the kernel $K()$ is given by

$$\hat{h}(t) = \frac{1}{b}\sum_{i=1}^{D} K\left(\frac{t - t_i}{b}\right)\Delta\tilde{H}(t_i)$$

The variance of $\hat{h}(t)$ is estimated by

$$\hat{\sigma^2}(\hat{h}(t)) = \frac{1}{b^2}\sum_{i=1}^{D} K\left(\frac{t - t_i}{b}\right)^2 \Delta\hat{V}(\tilde{H}(t_i))$$

For $t < b$, the symmetric kernels $K()$ are replaced by the corresponding asymmetric kernels of Gasser and Müller (1979). Let $q = \frac{t}{b}$. The modified kernels are as follows:

- uniform kernel:

$$K_{U,q}(x) = \frac{4(1 + q^3)}{(1 + q)^4} + \frac{6(1 - q)}{(1 + q)^3}x, \quad -1 \le x \le q$$

- Epanechnikov kernel:

$$K_{E,q}(x) = K_E(x)\frac{64(2 - 4q + 6q^2 - 3q^3) + 240(1 - q)^2 x}{(1 + q)^4(19 - 18q + 3q^2)}, \quad -1 \le x \le q$$

- biweight kernel:

$$K_{BW,q}(x) = K_{BW}(x)\frac{64(8 - 24q + 48q^2 - 45q^3 + 15q^4) + 1120(1 - q)^3 x}{(1 + q)^5(81 - 168q + 126q^2 - 40q^3 + 5q^4)}, \quad -1 \le x \le q$$

For $t_D - b \le t \le t_D$, let $q = \frac{t_D - t}{b}$. The asymmetric kernels for $t < b$ are used with x replaced by $-x$.

Using the log transform on the smoothed hazard rate, the 100(1–α)% pointwise confidence interval for the smoothed hazard rate $h(t)$ is given by

$$\hat{h}(t) = \hat{h}(t)\exp\left[\pm\frac{z_{1-\alpha/2}\hat{\sigma}(\hat{h}(t))}{\hat{h}(t)}\right]$$

where $z_{1-\frac{\alpha}{2}}$ is the (100(1–$\frac{\alpha}{2}$))th percentile of the standard normal distribution.

Optimal Bandwidth

The following mean integrated squared error (MISE) over the range τ_L and τ_U is used as a measure of the global performance of the kernel function estimator:

$$\begin{aligned}\text{MISE}(b) &= E\int_{\tau_L}^{\tau_U}(\hat{h}(i)-h(u))^2du \\ &= E\int_{\tau_L}^{\tau_U}\hat{h}^2(u)du - 2E\int_{\tau_L}^{\tau_U}\hat{h}(u)h(u)du + E\int_{\tau_L}^{\tau_U}h^2(u)du\end{aligned}$$

The last term is independent of the choice of the kernel and bandwidth and can be ignored when you are looking for the best value of b. The first integral can be approximated by using the trapezoid rule by evaluating $\hat{h}(t)$ at a grid of points $\tau_L = u_1 < \cdots < u_M = \tau_U$. You can specify τ_L, τ_R, and M by using the options GRIDL=, GRIDU=, and NMINGRID=, respectively, of the HAZARD plot. The second integral can be estimated by the Ramlau-Hansen (1983a, b) cross-validation estimate:

$$\frac{1}{b}\sum_{i\neq j}K\left(\frac{t_i-t_j}{b}\right)\Delta\hat{H}(t_i)\Delta\hat{H}(t_j)$$

Therefore, for a fixed kernel, the optimal bandwidth is the quantity b that minimizes

$$g(b) = \sum_{i=1}^{M-1}\left[\frac{u_{i+1}-u_k}{2}\left(\hat{h}^2(u_i)+\hat{h}^2(u_{i+1})\right)\right] - \frac{2}{b}\sum_{i\neq j}K\left(\frac{t_i-t_j}{b}\right)\Delta\hat{H}(t_i)\Delta\hat{H}(t_j)$$

The minimization is carried out by the golden section search algorithm.

Comparison of Two or More Groups of Survival Data

Let K be the number of groups. Let $S_k(t)$ be the underlying survivor function of the kth group, $k = 1, \ldots, K$. The null and alternative hypotheses to be tested are

$$H_0 : S_1(t) = S_2(t) = \cdots = S_K(t) \text{ for all } t \leq \tau$$

versus

$$H_1 : \text{at least one of the } S_k(t)\text{'s is different for some } t \leq \tau$$

respectively, where τ is the largest observed time.

Likelihood Ratio Test

The likelihood ratio test statistic (Lawless 1982) for test H_0 versus H_1 assumes that the data in the various samples are exponentially distributed and tests that the scale parameters are equal. The test statistic is computed as

$$\chi^2 = 2N\log\left(\frac{T}{N}\right) - 2\sum_{k=1}^{K}N_k\log\left(\frac{T_k}{N_k}\right)$$

where N_k is the total number of events in the kth group, $N = \sum_{k=1}^{k}N_k$, T_k is the total time on test in the kth stratum, and $T = \sum_{k=1}^{K}T_k$. The approximate probability value is computed by treating χ^2 as having a chi-square distribution with $K - 1$ degrees of freedom.

Nonparametric Tests

Let $(T_i, \delta_i, X_i), i = 1, \ldots, n$, denote an independent sample of right-censored survival data, where T_i is the possibly right-censored time, δ_i is the censoring indicator (δ_i=0 if T_i is censored and δ_i=1 if T_i is an event time), and $X_i = 1, \ldots, K$ for K different groups. Let $t_1 < t_2 < \cdots < t_D$ be the distinct event times in the sample. At time $t_j, j = 1, \ldots, D$, let $W(t_j)$ be a positive weight function, and let $Y_{jk} = \sum_{i:T_i \geq t_j} I(Xi = k)$ and $d_{jk} = \sum_{i:T_i = t_j} \delta_i I(X_i = k)$ be the size of the risk set and the number of events in the kth group, respectively. Let $Y_j = \sum_{k=1}^{K} Y_{jk}$, $d_j = \sum_{k=1}^{K} d_{jk}$.

The choices of the weight function $W(t_j)$ are given in Table 77.3.

Table 77.3 Weight Functions for Various Tests

Test	$W(t_i)$
Log-rank	1.0
Wilcoxon	Y_j
Tarone-Ware	$\sqrt{Y_j}$
Peto-Peto	$\tilde{S}(t_j)$
Modified Peto-Peto	$\tilde{S}(t_j)\frac{Y_j}{Y_j+1}$
Harrington-Fleming (p,q)	$[\hat{S}(t_{j-1})]^p[1-\hat{S}(t_{j-1})]^q, p \geq 0, q \geq 0$

In Table 77.3, $\hat{S}(t)$ is the product-limit estimate at t for the pooled sample, and $\tilde{S}(t)$ is a survivor function estimate close to $\hat{S}(t)$ given by

$$\tilde{S}(t) = \prod_{t_j \leq t} \left(1 - \frac{d_j}{Y_j + 1}\right)$$

Unstratified Tests The rank statistics (Klein and Moeschberger 1997, Section 7.3) for testing H_0 versus H_1 have the form of a K-vector $\mathbf{v} = (v_1, v_2, \ldots, v_K)'$ with

$$v_k = \sum_{j=1}^{D} \left[W(t_j) \left(d_{jk} - Y_{jk} \frac{d_j}{Y_j} \right) \right]$$

and the variance of v_k and the covariance of v_k and v_h are, respectively,

$$\begin{aligned} V_{kk} &= \sum_{j=1}^{D} \left[W^2(t_j) \frac{d_j(Y_j - d_j)Y_{jk}(Y_j - Y_{jk})}{Y_j^2(Y_j - 1)} \right], \quad 1 \leq k \leq K \\ V_{kh} &= -\sum_{j=1}^{D} \left[W^2(t_j) \frac{d_j(Y_j - d_j)Y_{jk}Y_{jh}}{Y_j^2(Y_j - 1)} \right], \quad 1 \leq k \neq h \leq K \end{aligned}$$

The statistic v_k can be interpreted as a weighted sum of observed minus expected numbers of failure for the kth group under the null hypothesis of identical survival curves. Let $\mathbf{V} = (V_{kh})$. The overall test statistic for homogeneity is $\mathbf{v}'\mathbf{V}^{-}\mathbf{v}$, where $\mathbf{V}^{-}$ denotes a generalized inverse of $\mathbf{V}$. This statistic is treated as having a chi-square distribution with degrees of freedom equal to the rank of $\mathbf{V}$ for the purposes of computing an approximate probability level.

Adjusted Log-Rank Test PROC LIFETEST computes the weighted log-rank test (Xie and Liu 2005, 2011) if you specify the WEIGHT statement. Let $(T_i, \delta_i, X_i, w_i), i = 1, \ldots, n$, denote an independent sample of right-censored survival data, where T_i is the possibly right-censored time, δ_i is the censoring indicator (δ_i=0 if T_i is censored and δ_i=1 if T_i is an event time), $X_i = 1, \ldots, K$ for K different groups, and w_i is the weight from the WEIGHT statement. Let $t_1 < t_2 < \cdots < t_D$ be the distinct event times in the sample. At each $t_j, j = 1, \ldots, D$, and for each $1 \le k \le K$, let

$$d_{jk} = \sum_{i:T_i=t_j} I(X_i = k) \qquad d^w_{jk} = \sum_{i:T_i=t_j} w_i I(X_i = k)$$
$$Y_{jk} = \sum_{i:T_i\ge t_j} I(X_i = k) \qquad Y^w_{jk} = \sum_{i:T_i\ge t_j} w_i I(X_i = k)$$

Let $d_j = \sum_{k=1}^K d_{jk}$ and $Y_j = \sum_{k=1}^K Y_{jk}$ denote the number of events and the number at risk, respectively, in the combined sample at time t_j. Similarly, let $d^w_j = \sum_{k=1}^K d^w_{jk}$ and $Y^w_j = \sum_{k=1}^K Y^w_{jk}$ denote the weighted number of events and the weighted number at risk, respectively, in the combined sample at time t_j. The test statistic is

$$v_k = \sum_{j=1}^D \left(d^w_{jk} - Y^w_{jk} \frac{d^w_j}{Y^w_j} \right), \quad k = 1, \ldots, K$$

and the variance of v_k and the covariance of v_k and v_h are, respectively,

$$V_{kk} = \sum_{j=1}^D \left\{ \frac{d_j(Y_j - d_j)}{Y_j(Y_j-1)} \sum_{i=1}^{Y_j} \left[\left(\frac{Y^w_{jk}}{Y^w_j}\right)^2 w_i^2 I\{X_i \ne k\} + \left(\frac{Y^w_j - Y^w_{jk}}{Y^w_j}\right)^2 w_i^2 I\{X_i = k\} \right] \right\}, \quad 1 \le k \le K$$

$$V_{kh} = \sum_{j=1}^D \left\{ \frac{d_j(Y_j - d_j)}{Y_j(Y_j-1)} \sum_{i=1}^{Y_j} \left[\frac{Y^w_{jk} Y^w_{jh}}{(Y^w_j)^2} w_i^2 I\{X_i \ne k, h\} - \frac{(Y^w_j - Y^w_{jk})Y^w_{jh}}{(Y^w_j)^2} w_i^2 I\{X_i = k\} \right.\right.$$
$$\left.\left. - \frac{(Y^w_j - Y^w_{jh})Y^w_{jk}}{(Y^w_j)^2} w_i^2 I\{X_i = h\} \right] \right\}, \quad 1 \le k \ne h \le K$$

Let $V = (V_{kh})$. Under H_0, the weighted K-sample test has a χ^2 statistic given by

$$\chi^2 = (v_1, \ldots, v_K)\mathbf{V}^-(v_1, \ldots, v_K)'$$

with $K - 1$ degrees of freedom.

Stratified Tests Suppose the test is to be stratified on M levels of a set of STRATA variables. Based only on the data of the sth stratum ($s = 1, \ldots, M$), let $\mathbf{v}_s$ be the test statistic (Klein and Moeschberger 1997, Section 7.5) for the sth stratum, and let $\mathbf{V}_s$ be its covariance matrix. Let

$$\mathbf{v} = \sum_{s=1}^M \mathbf{v}_s$$
$$\mathbf{V} = \sum_{s=1}^M \mathbf{V}_s$$

A global test statistic is constructed as

$$\chi^2 = \mathbf{v}'\mathbf{V}^{-}\mathbf{v}$$

Under the null hypothesis, the test statistic has a χ^2 distribution with the same degrees of freedom as the individual test for each stratum.

Multiple-Comparison Adjustments Let χ^2_r denote a chi-square random variable with r degrees of freedom. Denote ϕ and Φ as the density function and the cumulative distribution function of a standard normal distribution, respectively. Let m be the number of comparisons; that is,

$$m = \begin{cases} \frac{k(k-1)}{2} & \text{DIFF} = \text{ALL} \\ k-1 & \text{DIFF} = \text{CONTROL} \end{cases}$$

For a two-sided test that compares the survival of the jth group with that of lth group, $1 \le j \ne l \le r$, the test statistic is

$$z^2_{jl} = \frac{(v_j - v_l)^2}{V_{jj} + V_{ll} - 2V_{jl}}$$

and the raw p-value is

$$p = \Pr(\chi^2_1 > z^2_{jl})$$

Adjusted p-values for various multiple-comparison adjustments are computed as follows:

- Bonferroni adjustment:

  $$p = \min\{1, m\Pr(\chi^2_1 > z^2_{jl})\}$$

- Dunnett-Hsu adjustment: With the first group being the control, let $\mathbf{C} = (c_{ij})$ be the $(r-1) \times r$ matrix of contrasts; that is,

  $$c_{ij} = \begin{cases} 1 & i = 1, \ldots, r-1, j = 2, \ldots, r \\ -1 & j = i+1, i = 2, \ldots, r \\ 0 & \text{otherwise} \end{cases}$$

  Let $\mathbf{\Sigma} \equiv (\sigma_{ij})$ and $\mathbf{R} \equiv (r_{ij})$ be covariance and correlation matrices of $\mathbf{Cv}$, respectively; that is,

  $$\mathbf{\Sigma} = \mathbf{CVC}'$$

  and

  $$r_{ij} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}}$$

  The factor-analytic covariance approximation of Hsu (1992) is to find $\lambda_1, \ldots, \lambda_{r-1}$ such that

  $$\mathbf{R} = \mathbf{D} + \boldsymbol{\lambda}\boldsymbol{\lambda}'$$

where $\mathbf{D}$ is a diagonal matrix with the jth diagonal element being $1 - \lambda_j$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_{r-1})'$. The adjusted p-value is

$$p = 1 - \int_{-\infty}^{\infty} \phi(y) \prod_{i=1}^{r-1} \left[\Phi\left(\frac{\lambda_i y + z_{jl}}{\sqrt{1 - \lambda_i^2}} \right) - \Phi\left(\frac{\lambda_i y - z_{jl}}{\sqrt{1 - \lambda_i^2}} \right) \right] dy$$

which can be obtained in a DATA step as

$$p = \mathrm{PROBMC}(\text{"DUNNETT2"}, z_{ij}, ., ., r - 1, \lambda_1, \ldots, \lambda_{r-1}).$$

- Scheffé adjustment:

$$p = \Pr(\chi^2_{r-1} > z^2_{jl})$$

- Šidák adjustment:

$$p = 1 - \{1 - \Pr(\chi^2_1 > z^2_{jl})\}^m$$

- SMM adjustment:

$$p = 1 - [2\Phi(z_{jl}) - 1]^m$$

which can also be evaluated in a DATA step as

$$p = 1 - \mathrm{PROBMC}(\text{"MAXMOD"}, z_{jl}, ., ., m).$$

- Tukey adjustment:

$$p = 1 - \int_{-\infty}^{\infty} r\phi(y)[\Phi(y) - \Phi(y - \sqrt{2}z_{jl})]^{r-1} dy$$

which can also be evaluated in a DATA step as

$$p = 1 - \mathrm{PROBMC}(\text{"RANGE"}, \sqrt{2}z_{jl}, ., ., r).$$

Trend Tests Trend tests (Klein and Moeschberger 1997, Section 7.4) have more power to detect ordered alternatives as

$$H_2 : S_1(t) \geq S_2(t) \geq \cdots \geq S_k(t), t \leq \tau, \text{ with at least one inequality}$$

or

$$H_2 : S_1(t) \leq S_2(t) \leq \cdots \leq S_k(t), t \leq \tau, \text{ with at least one inequality}$$

Let $a_1 < a_2 < \cdots < a_k$ be a sequence of scores associated with the k samples. The test statistic and its standard error are given by $\sum_{j=1}^k a_j v_j$ and $\sum_{j=1}^k \sum_{l=1}^k a_j a_l V_{jl}$, respectively. Under H_0, the z-score

$$Z = \frac{\sum_{j=1}^k a_j v_j}{\sqrt{\{\sum_{j=1}^k \sum_{l=1}^k a_j a_l V_{jl}\}}}$$

has, asymptotically, a standard normal distribution. PROC LIFETEST provides both one-tail and two-tail p-values for the test.

Rank Tests for the Association of Survival Time with Covariates

The rank tests for the association of covariates (Kalbfleisch and Prentice 1980, Chapter 6) are more general cases of the rank tests for homogeneity. In this section, the index α is used to label all observations, $\alpha = 1, 2, \ldots, n$, and the indices i, j range only over the observations that correspond to events, $i, j = 1, 2, \ldots, k$. The ordered event times are denoted as $t_{(i)}$, the corresponding vectors of covariates are denoted as $\mathbf{z}_{(i)}$, and the ordered times, both censored and event times, are denoted as t_α.

The rank test statistics have the form

$$\mathbf{v} = \sum_{\alpha=1}^{n} c_{\alpha,\delta_\alpha} \mathbf{z}_\alpha$$

where n is the total number of observations, c_{α,δ_α} are rank scores, which can be either log-rank or Wilcoxon rank scores, δ_α is 1 if the observation is an event and 0 if the observation is censored, and $\mathbf{z}_\alpha$ is the vector of covariates in the TEST statement for the αth observation. Notice that the scores, c_{α,δ_α}, depend on the censoring pattern and that the terms are summed up over all observations.

The log-rank scores are

$$c_{\alpha,\delta_\alpha} = \sum_{(j:t_{(j)} \le t_\alpha)} \left(\frac{1}{n_j} - \delta_\alpha \right)$$

and the Wilcoxon scores are

$$c_{\alpha,\delta_\alpha} = 1 - (1 + \delta_\alpha) \prod_{(j:t_{(j)} \le t_\alpha)} \frac{n_j}{n_j + 1}$$

where n_j is the number at risk just prior to $t_{(j)}$.

The estimates used for the covariance matrix of the log-rank statistics are

$$\mathbf{V} = \sum_{i=1}^{k} \frac{\mathbf{V}_i}{n_i}$$

where $\mathbf{V}_i$ is the corrected sum of squares and crossproducts matrix for the risk set at time $t_{(i)}$; that is,

$$\mathbf{V}_i = \sum_{(\alpha:t_\alpha \ge t_{(i)})} (\mathbf{z}_\alpha - \bar{\mathbf{z}}_i)'(\mathbf{z}_\alpha - \bar{\mathbf{z}}_i)$$

where

$$\bar{\mathbf{z}}_i = \sum_{(\alpha:t_\alpha \ge t_{(i)})} \frac{\mathbf{z}_\alpha}{n_i}$$

The estimate used for the covariance matrix of the Wilcoxon statistics is

$$\mathbf{V} = \sum_{i=1}^{k} \left[a_i (1 - a_i^*)(2\mathbf{z}_{(i)}\mathbf{z}'_{(i)} + \mathbf{S}_i) - (a_i^* - a_i) \left(a_i \mathbf{x}_i \mathbf{x}'_i + \sum_{j=i+1}^{k} a_j (\mathbf{x}_i \mathbf{x}'_j + \mathbf{x}_j \mathbf{x}'_i) \right) \right]$$

where

$$
\begin{aligned}
a_i &= \prod_{j=1}^{i} \frac{n_j}{n_j + 1} \\
a_i^* &= \prod_{j=1}^{i} \frac{n_j + 1}{n_j + 2} \\
\mathbf{S}_i &= \sum_{(\alpha : t_{(i+1)} > t_\alpha > t_{(i)})} \mathbf{z}_\alpha \mathbf{z}_\alpha' \\
\mathbf{x}_i &= 2\mathbf{z}_{(i)} + \sum_{(\alpha : t_{(i+1)} > t_\alpha > t_{(i)})} \mathbf{z}_\alpha
\end{aligned}
$$

In the case of tied failure times, the statistics $\mathbf{v}$ are averaged over the possible orderings of the tied failure times. The covariance matrices are also averaged over the tied failure times. Averaging the covariance matrices over the tied orderings produces functions with appropriate symmetries for the tied observations; however, the actual variances of the $\mathbf{v}$ statistics would be smaller than the preceding estimates. Unless the proportion of ties is large, it is unlikely that this will be a problem.

The univariate tests for each covariate are formed from each component of $\mathbf{v}$ and the corresponding diagonal element of $\mathbf{V}$ as v_i^2 / V_{ii}. These statistics are treated as coming from a chi-square distribution for calculation of probability values.

The statistic $\mathbf{v}'\mathbf{V}^-\mathbf{v}$ is computed by sweeping each pivot of the $\mathbf{V}$ matrix in the order of greatest increase to the statistic. The corresponding sequence of partial statistics is tabulated. Sequential increments for including a given covariate and the corresponding probabilities are also included in the same table. These probabilities are calculated as the tail probabilities of a chi-square distribution with one degree of freedom. Because of the selection process, these probabilities should not be interpreted as p-values.

If desired for data screening purposes, the output data set requested by the OUTTEST= option can be treated as a sum of squares and crossproducts matrix and processed by the REG procedure by using the option METHOD=RSQUARE. Then the sets of variables of a given size can be found that give the largest test statistics. Output 77.1 illustrates this process.

Analysis of Competing-Risks Data

Competing risks arise in studies in which individuals are exposed to two or more mutually exclusive failure events, denoted by $\delta \in \{1, \ldots, J\}$. When a failure occurs, you observe the time T and the cause of failure δ. The cumulative incidence function (CIF), also known as the subdistribution function, for failures of cause j is the probability

$$F_j(t) = \Pr(T \le t, \delta = j)$$

The nonparametric analysis of competing-risks data consists of estimating the CIF and comparing the CIFs of two or more groups.

Estimation of the CIF

For a set of competing-risks data with $J \geq 2$ causes of failure, let $t_1 < t_2 < \cdots < t_L$ be the distinct uncensored times. For each $l = 1, \ldots, L$, let Y_l be the number of subjects at risk at t_l, and let d_{jl} be the number of failures of cause j at t_l. Let $\hat{S}(t)$ be the Kaplan-Meier estimator that would have been obtained by assuming that all failure causes are of the same type. Denote $t_0 = 0$.

The nonparametric maximum likelihood estimator of the CIF of cause j is

$$\hat{F}_j(t) = \sum_{t_l \leq t} \frac{d_{ji}}{Y_l} \hat{S}(t_{l-1})$$

PROC LIFETEST provides two standard error estimators of the CIF estimator: one is based on the theory of counting processes (Aalen 1978), and the other is based on the delta method (Marubini and Valsecchi 1995). You use the ERROR= option in the PROC LIFETEST statement to choose the standard error estimator. The default is the Aalen estimator (ERROR=AALEN). Denote $d_{.l} = \sum_{j=1}^{J} d_{jl}$.

Aalen Estimator

$$\begin{aligned}
\hat{\sigma}^2_{\mathrm{A}}(\hat{F}_j(t)) = & \sum_{t_l \leq t} \left[\hat{F}_j(t) - \hat{F}_j(t_l)\right]^2 \frac{d_{.l}}{(Y_l - 1)(Y_l - d_{.l})} \\
& + \sum_{t_l \leq t} \hat{S}^2(t_{l-1}) \frac{d_{kj}(Y_l - d_{jl})}{Y_l^2(Y_l - 1)} \\
& - 2 \sum_{t_l \leq t} \left[\hat{F}_j(t) - \hat{F}_j(t_l)\right] \hat{S}(t_{l-1}) \frac{d_{jl}(Y_l - d_{jl})}{Y_l(Y_l - d_{.l})(Y_l - 1)}
\end{aligned}$$

Delta Estimator

$$\begin{aligned}
\hat{\sigma}^2_{\mathrm{D}}(\hat{F}_j(t)) = & \sum_{t_l \leq t} \left[\hat{F}_j(t) - \hat{F}_j(t_l)\right]^2 \frac{d_{.l}}{Y_l(Y_l - d_{.l})} \\
& + \sum_{t_l \leq t} \hat{S}^2(t_{l-1}) \frac{d_{jl}(Y_l - d_{jl})}{Y_l^3} \\
& - 2 \sum_{t_l \leq t} \left[\hat{F}_j(t) - \hat{F}_j(t_l)\right] \hat{S}(t_{l-1}) \frac{d_{jl}}{Y_l^2}
\end{aligned}$$

Comparison of the CIF of a Competing Risk for Two or More Groups

Let K be the number of groups. Consider failure of type 1 to be the failure type of interest. Let F_{1k} be the cumulative incidence function of type 1 in group k. The null hypothesis to be tested is

$$H_0 : F_{11} = F_{12} = \cdots = F_{1K} \equiv F_1^0$$

Gray (1988, Section 2) gives the following K-sample test procedure for testing H_0. Let $(T_{ik}, \delta_{ik}), i = 1, \ldots, n_k$ be the observed data in the kth group. Without loss of generality, assume that there are only two types of failure ($J = 2$). The number of failures of type j by t is

$$N_{jk}(t) = \sum_{i=1}^{n_k} I(T_{ik} \leq t, \delta_{ik} = j), \quad j = 1, 2$$

and the number of subjects at risk just before t in group k is

$$Y_k(t) = \sum_{i=1}^{n_k} I(T_{ik} \geq t)$$

For group k, let $\hat{S}_k(t)$ be the Kaplan-Meier estimator of the survivor function that you obtain by assuming that all failure causes are of the same type. The cumulative incidence function $F_{jk}(t)$ of type j in the kth group is estimated by

$$\hat{F}_{jk}(t) = \int_0^t \hat{S}_k(u-) Y_k^{-1}(u) dN_{jk}(u)$$

Let τ_k be the largest uncensored time in group k. Define

$$\hat{G}_{jk}(t) = 1 - \hat{F}_{jk}(t)$$

$$R_k(t) = I(\tau_k \geq t) Y_k(t) \frac{\hat{G}_{1k}(t-)}{\hat{S}_k(t-)}$$

The cumulative hazard of the subdistribution for group k, Γ_{1k}, is estimated by

$$\hat{\Gamma}_{1k}(t) = \int_0^t \frac{d\hat{F}_{1k}(u)}{\hat{G}_{1k}(u-)} = \int_0^t \frac{dN_{1k}(u)}{R_k(u-)}, \quad t \leq \tau_k$$

Under the null hypothesis H_0, you can estimate the null value of $\Gamma_{1k}(t)$, denoted by $\Gamma_1^0(t)$, by

$$\hat{\Gamma}_1^0(t) = \int_0^t \frac{dN_{1.}(u)}{R_.(u)}$$

The K-sample test is based on $\mathbf{z} = (z_1, \ldots, z_K)'$, where

$$z_k = \int_0^{\tau_k} R_k(t) \left[d\hat{\Gamma}_{1k}(t) - d\hat{\Gamma}_1^0(t) \right]$$

You can estimate the asymptotic covariance matrix $\Sigma = (\sigma_{kk'})$ as

$$\hat{\sigma}_{kk'}^2 = \sum_{r=1}^{K} \int_0^{\tau_k \wedge \tau_{k'}} \frac{a_{kr}(t) a_{k'r}(t)}{\hat{h}_r(t)} d\hat{F}_1^0(t) + \sum_{r=1}^{K} \int_0^{\tau_k \wedge \tau_{k'}} \frac{b_{2kr}(t) b_{2k'r}(t)}{\hat{h}_r(t)} d\hat{F}_{2r}(t)$$

where

$$
\begin{aligned}
\hat{h}_r(t) &= \frac{I(t \le \tau_r)Y_r(t)}{\hat{S}_r(t-)} \\
\hat{F}_1^0(t) &= \int_0^t \frac{dN_{1.}(u)}{\hat{h}_.(u)} \\
\hat{G}_1^0(t) &= 1 - \hat{F}_1^0(t) \\
a_{kr}(t) &= d_{1kr}(t) + b_{1kr}(t) \\
b_{jkr}(t) &= \left[I(j=1) - \frac{\hat{G}_1^0(t)}{\hat{S}_r(t)} \right] [c_{kr}(\tau_k) - c_{kr}(t)] \\
c_{kr}(t) &= \int_0^t d_{1kr}(u) d\hat{\Gamma}_1^0(u) \\
d_{jkr}(t) &= I(j=1)R_k(t) \frac{I(k=r) - \frac{\hat{h}_r(t)}{\hat{h}_.(t)}}{\hat{G}_1^0(t)}
\end{aligned}
$$

Because $\sum_{k=1}^{K} z_k = 0$, only $K-1$ scores are linearly independent. The K-sample test statistic is formed as a quadratic form of the first $K-1$ components of $\mathbf{z}$ and the inverse of the estimated covariance matrix. Under the null hypothesis H_0, this K-sample test statistic has approximately a chi-square distribution with $K-1$ degrees of freedom.

If you specify the GROUP= option in the STRATA statement, you can obtain a stratified version of the test by computing the contributions to z_k and $\sigma^2_{kk'}$ for each stratum, summing the contributions over the strata, and proceeding as before.

Restricted Mean Analysis

Let T be a nonnegative random variable that represents the failure time of an individual from a homogeneous population. The survivor function (also known as the survival function) of T is defined as

$$S(t) = \Pr(T > t)$$

Assume that τ is a prespecified time point of interest. Let R be the minimum of T and τ,

$$R = T \wedge \tau = \min(T, \tau)$$

The restricted mean survival time (RMST) is defined as the expected value of R:

$$\mathrm{RMST}(\tau) = E(R) = E[\min(T, \tau)]$$

It can be evaluated by the area under the survivor function over $[0, \tau]$ as

$$\mathrm{RMST}(\tau) = \int_0^\tau S(u)du$$

The restricted mean time lost (RMTL) is defined as the expected value of $\tau - R$:

$$\mathrm{RMTL}(\tau) = E(\tau - R) = \tau - E[\min(T, \tau)] = \int_0^\tau [1 - S(u)]du$$

Let $t_1 < t_2 < \cdots < t_D$ represent the distinct event times. For each $i = 1, \ldots, D$, let Y_i be the number of surviving units (the size of the risk set) just prior to t_i, and let d_i be the number of units that fail at t_i.

The Kaplan-Meier (product-limit) estimate of the survivor function at t_i is the cumulative product

$$\hat{S}(t_i) = \prod_{j=1}^{i} \left(1 - \frac{d_j}{Y_j}\right)$$

If the largest observed time is uncensored, the estimated mean survival time is

$$\hat{\mu} = \sum_{i=1}^{D} \hat{S}(t_{i-1})(t_i - t_{i-1})$$

where t_0 is defined to be zero.

$\mathrm{RMST}(\tau)$ is estimated by

$$\widehat{\mathrm{RMST}}(\tau) = \int_0^{\tau} \hat{S}(t)dt = \sum_{i=1}^{N^*} \hat{S}(t_{i-1})(t_i - t_{i-1}) + \hat{S}(t_{N^*})(\tau - t_{N^*})$$

where N^* is the number of t_i values that are less than τ.

$\mathrm{RMTL}(\tau)$ is estimated by $\widehat{\mathrm{RMTL}}(\tau) = \tau - \mathrm{RMST}(\tau)$.

The standard error of $\widehat{\mathrm{RMST}}(\tau)$ or $\widehat{\mathrm{RMTL}}(\tau)$ is estimated as

$$\hat{\sigma} = \sqrt{\frac{m}{m-1} \sum_{i=1}^{N^*} \frac{d_i A_i^2}{Y_i(Y_i - d_i)}}$$

where

$$\begin{aligned} A_i &= \int_{t_i}^{\tau} \hat{S}(t)dt = \sum_{j=i}^{N^*} \hat{S}(t_j)(t_{j+1} - t_j) + \hat{S}(t_{N^*})(\tau - t_{N^*}) \\ m &= \sum_{j=1}^{N^*} d_j \end{aligned}$$

You can use the TAU= suboption in the RMST or RMTL option in the PROC LIFETEST statement to specify a τ value in the calculation. Note that the term $\frac{m}{m-1}$ is omitted unless you specify the BC suboption in the RMST or RMTL option in the PROC LIFETEST statement. The WEIGHT statement is not supported for the restricted mean analysis.

Comparing the Restricted Means of Two or More Groups

Let K be the number of groups. Let $S_k(t)$ be the underlying survivor function of the kth group, $k = 1, \ldots, K$.

Assume that τ is a prespecified time point of interest and $S_k(\tau) > 0$. The following methods are presented in terms of $\mathrm{RMST}(\tau)$, but they also extend to the analysis of $\mathrm{RMTL}(\tau)$.

The null and alternative hypotheses to be tested are

$$H_0 : \mathrm{RMST}_1(\tau) = \mathrm{RMST}_2(\tau) = \cdots = \mathrm{RMST}_K(\tau)$$

versus

$$H_1 : \text{at least one of the } \mathrm{RMST}_k(\tau)\text{'s is different.}$$

Let $\widehat{\mathrm{RMST}}(\tau) = [\widehat{\mathrm{RMST}}_1(\tau), \widehat{\mathrm{RMST}}_2(\tau), \ldots, \widehat{\mathrm{RMST}}_K(\tau)]^T$ be the vector of estimated RMSTs for the K groups.

Let $\hat{\mathbf{\Sigma}}$ be the estimated covariance matrix for $\widehat{\mathrm{RMST}}(\tau)$. It is a diagonal matrix, and the jth diagonal element is $\hat{\sigma}_j^2$, which is the estimated variance of $\widehat{\mathrm{RMST}}_j(\tau)$.

Let D be a $(K-1) \times K$ matrix whose jth row is $e_j - e_{j+1}$, where e_j is a K-dimensional vector whose jth element is 1 and whose other elements are 0. The test statistic is computed as

$$(\widehat{\mathrm{RMST}}(\tau))^T D^T (D \hat{\mathbf{\Sigma}} D^T)^- D \widehat{\mathrm{RMST}}(\tau)$$

Under the null hypothesis H_0, this K-sample test statistic has approximately a chi-square distribution with degrees of freedom equal to the rank of $D \hat{\mathbf{\Sigma}} D^T$.

If you specify the GROUP= option in the STRATA statement, you can obtain a stratified version of the test by computing $\widehat{\mathrm{RMST}}(\tau)$ and $\hat{\mathbf{\Sigma}}$ for each stratum, summing them over the strata, and performing the chi-square test as before.

If you specify the DIFF= option in the STRATA statement, you can make inference with regard to paired differences of the RMST. If you also specify the GROUP= option, the RMST for a given group under comparison is computed as the sample mean of the stratum-specific RMSTs of the corresponding group, and the variance is obtained as the variance of the sample mean. You can use the ADJUST= option in the STRATA statement to make multiple-comparison adjustments to the p-values.

Output Data Sets

OUTCIF= Data Set

You can specify the OUTCIF= option in the PROC LIFETEST statement to create an output data set that contains the cumulative incidence estimates. The data set contains the following columns:

- any specified BY variables
- the censoring variable as given in the TIME statement to indicate the failure of interest
- a numeric variable named STRATUM that numbers the strata, if you specify the STRATA statement
- any specified STRATA variables, whose values come from either their original values or the midpoints of the stratum intervals if you use cutpoints to define strata (semi-infinite intervals are labeled by their finite endpoint)
- the GROUP= variable, if you specify the GROUP= option in the STRATA statement
- the Timelist variable, if you specify the TIMELIST= option and the REDUCEOUT option in the PROC LIFETEST statement

- the time variable as specified in the TIME statement
- AtRisk, a variable that contains the number of subjects at risk just before the specified time. This variable is omitted if you specify the REDUCEOUT option in the PROC LIFETEST statement.
- Event, a variable that contains the number of subjects that fail at the specified time from the cause of interest. This variable is omitted if you specify the REDUCEOUT option in the PROC LIFETEST statement.
- AllEventTypes, a variable that contains the number of subjects that fail at the specified time from any cause. This variable is omitted if you specify the REDUCEOUT option in the PROC LIFETEST statement.
- CIF, a variable that contains the point estimates of the cumulative incidence function
- CIF_STDERR, a variable that contains the standard errors of the CIF estimator
- ALPHA, a variable that contains the α-level of the confidence intervals
- CONFTYPE, a variable that contains the name of the transformation that is applied to the CIF to compute the confidence intervals for the CIF
- CIF_LCL, a variable that contains the lower confidence limits of the CIF
- CIF_UCL, a variable that contains the upper confidence limits of the CIF

Each estimated CIF contains an initial observation whose value is 1 for the CIF and 0 for the time. The output data set contains an observation for each distinct failure time when an event occurs or an observation is censored.

OUTSURV= Data Set

You can specify the OUTSURV= option in the PROC LIFETEST statement to create an output data set that contains the survival estimates. The data set contains the following columns:

- any specified BY variables
- a numeric variable STRATUM that numbers the strata, if you specify the STRATA statement
- any specified STRATA variables, their values coming from either their original values or the midpoints of the stratum intervals if endpoints are used to define strata (semi-infinite intervals are labeled by their finite endpoint)
- the GROUP= variables, if you specify the GROUP= option in the STRATA statement
- the time variable as specified in the TIME statement. For METHOD=KM, METHOD=BRESLOW, or METHOD=FH, it contains the observed failure or censored times. For the life-table estimates, it contains the lower endpoints of the time intervals.
- SURVIVAL, a variable that contains the survivor function estimates
- CONFTYPE, a variable that contains the name of the transformation applied to the survival time in the computation of confidence intervals

- SDF_LCL, a variable that contains the lower limits of the pointwise confidence intervals for the survivor function
- SDF_UCL, a variable that contains the upper limits of the pointwise confidence intervals for the survivor function

If the estimation uses the product-limit, Breslow, or Fleming-Harrington method, then the data set also contains the following:

- _CENSOR_, an indicator variable that has a value 1 for a censored observation and a value 0 for an event observation
- SDF_STDERR, a variable that contains the standard error of the survivor function estimator
- HW_LCL, a variable that contains the lower limits of the Hall-Wellner confidence bands (if you specify the CONFBAND=HW option or the CONFBAND=ALL option in the PROC LIFETEST statement)
- HW_UCL, a variable that contains the upper limits of the Hall-Wellner confidence bands (if you specify the CONFBAND=HW option or the CONFBAND=ALL option in the PROC LIFETEST statement)
- EP_LCL, a variable that contains the lower limits of the equal-precision confidence bands (if you specify the CONFBAND=EP option or the CONFBAND=ALL option in the PROC LIFETEST statement)
- EP_UCL, a variable that contains the upper limits of the equal-precision confidence bands (if you specify the CONFBAND=EP option or the CONFBAND=ALL option in the PROC LIFETEST statement)

If the estimation uses the life-table method, then the data set also contains the following:

- MIDPOINT, a variable that contains the value of the midpoint of the time interval
- PDF, a variable that contains the density function estimates
- PDF_LCL, a variable that contains the lower endpoints of the PDF confidence intervals
- PDF_UCL, a variable that contains the upper endpoints of the PDF confidence intervals
- HAZARD, a variable that contains the hazard estimates
- HAZ_LCL, a variable that contains the lower endpoints of the hazard confidence intervals
- HAZ_UCL, a variable that contains the upper endpoints of the hazard confidence intervals

Each survival function contains an initial observation with the value 1 for the SDF and the value 0 for the time. The output data set contains an observation for each distinct failure time if the product-limit, Breslow, or Fleming-Harrington method is used, or it contains an observation for each time interval if the life-table method is used. The product-limit, Breslow, or Fleming-Harrington survival estimates are defined to be right continuous; that is, the estimates at a given time include the factor for the failure events that occur at that time.

OUTTEST= Data Set

The OUTTEST= option in the LIFETEST statement creates an output data set that contains the rank statistics for testing the association of failure time with covariates. It contains the following:

- any specified BY variables
- _TYPE_, a character variable of length 8 that labels the type of rank test, either "LOG-RANK" or "WILCOXON"
- _NAME_, a character variable of length 8 that labels the rows of the covariance matrix and the test statistics
- the TIME variable, containing the overall test statistic in the observation that has _NAME_ equal to the name of the time variable and the univariate test statistics under their respective covariates.
- all variables listed in the TEST statement

The output is in the form of a symmetric matrix formed by the covariance matrix of the rank statistics bordered by the rank statistics and the overall chi-square statistic. If the value of _NAME_ is the name of a variable in the TEST statement, the observation contains a row of the covariance matrix and the value of the rank statistic in the time variable. If the value of _NAME_ is the name of the TIME variable, the observation contains the values of the rank statistics in the variables from the TEST list and the value of the overall chi-square test statistic in the TIME variable.

Two complete sets of statistics labeled by the _TYPE_ variable are produced, one for the log-rank test and one for the Wilcoxon test.

Displayed Output

The following sections describe the output that PROC LIFETEST produces by default. The output is organized into various tables, which are discussed in their order of appearance. The set of tables that PROC LIFETEST produces for a survival analysis, specifically with one type of failure, is different from the set of tables that it produces for an analysis of competing-risks data, which have multiple types of failure.

Tables for Survival Analysis

Product-Limit Survival Estimates

The "Product-Limit Survival Estimates" table is displayed if you request the product-limit method of estimation. The table displays the following:

- the observed (event or censored) time
- the number of units at risk (if you specify the ATRISK option in the PROC LIFETEST statement)
- the number of events (if you specify the ATRISK option in the PROC LIFETEST statement)
- the product-limit estimate of the survivor function
- the corresponding estimate of the cumulative distribution function of the failure time

- the standard error estimate of the survivor function estimator
- the Nelson-Aalen cumulative hazard function estimate (if you specify the NELSON option in the PROC LIFETEST statement)
- the standard error of the Nelson-Aalen estimator (if you specify the NELSON option in the PROC LIFETEST statement)
- the number of event times that have been observed
- the number of event or censored times that remain to be observed
- the frequency of the observed times (if you specify the FREQ statement)
- values of the ID variables (if you specify the ID statement)

The ODS name of this table is ProductLimitEstimates.

Breslow Survival Estimates

The "Breslow Survival Estimates" table is displayed if you request the Breslow method of estimation. The table displays the following:

- the observed (event or censored) time
- the number of units at risk (if you specify the ATRISK option in the PROC LIFETEST statement)
- the number of events (if you specify the ATRISK option in the PROC LIFETEST statement)
- the Breslow estimate of the survivor function
- the corresponding estimate of the cumulative distribution function of the failure time
- the standard error estimate of the survivor function estimator
- the Nelson-Aalen cumulative hazard function estimate (if you specify the NELSON option in the PROC LIFETEST statement)
- the standard error of the Nelson-Aalen estimator (if you specify the NELSON option in the PROC LIFETEST statement)
- the number of event times that have been observed
- the number of event or censored times that remain to be observed
- the frequency of the observed times (if you specify the FREQ statement)
- values of the ID variables (if you specify the ID statement)

The ODS name of this table is BreslowEstimates.

Fleming-Harrington Survival Estimates

The "Fleming-Harrington Survival Estimates" table is displayed if you request the Fleming-Harrington method of estimation. The table displays the following:

- the observed (event or censored) time
- the number of units at risk (if you specify the ATRISK option in the PROC LIFETEST statement)
- the number of events (if you specify the ATRISK option in the PROC LIFETEST statement)
- the Fleming-Harrington estimate of the survivor function
- the corresponding estimate of the cumulative distribution function of the failure time
- the standard error estimate of the survivor function estimator
- the Nelson-Aalen cumulative hazard function estimate (if you specify the NELSON option in the PROC LIFETEST statement)
- the standard error of the Nelson-Aalen estimator (if you specify the NELSON option in the PROC LIFETEST statement)
- the number of event times that have been observed
- the number of event or censored times that remain to be observed
- the frequency of the observed times (if you specify the FREQ statement)
- values of the ID variables (if you specify the ID statement)

The ODS name of this table is FlemingEstimates.

Quartile Estimates

The "Quartiles Estimates" table is displayed if you request the product-limit, Breslow, or Fleming-Harrington method of estimation. The table displays the following:

- point estimates of the quartiles of the survival times
- the lower and upper confidence limits for the quartiles

The ODS name of this table is Quartiles.

Mean Estimate

The "Mean Estimate" table is displayed if you request the product-limit, Breslow, or Fleming-Harrington method of estimation. The table displays the following:

- the estimated mean survival time
- the estimated standard error of the mean estimator

The ODS name of this table is Means.

Life-Table Survival Estimates

The "Life-Table Survival Estimates" table is displayed if you request the life-table method of estimation. The table displays the following:

- the time intervals into which the failure and censored times are distributed. Each interval is from the lower limit, up to but not including the upper limit; if the upper limit is infinity, the missing value is printed.
- the number of events that occur in the interval
- the number of censored observations that fall into the interval
- the effective sample size for the interval
- the estimate of conditional probability of events (failures) in the interval
- the standard error of the conditional probability estimator
- the estimate of the survival function at the beginning of the interval
- the estimate of the cumulative distribution function of the failure time at the beginning of the interval
- the standard error estimate of the survivor function estimator
- the estimate of the median residual lifetime, which is the amount of time elapsed before reducing the number of at-risk units to one-half. This is also known as the *median future lifetime* in Elandt-Johnson and Johnson (1980)).
- the estimated standard error of the median residual lifetime estimator
- the density function estimated at the midpoint of the interval
- the standard error estimate of the density estimator
- the hazard rate estimated at the midpoint of the interval
- the standard error estimate of the hazard estimator

The ODS name of this table is LifetableEstimates.

Summary of the Number of Censored and Uncensored Values

The "Summary of the Number of Censored and Uncensored Values" table displays following:

- the stratum identification (if you specify the STRATA statement)
- the total number of observations
- the number of event observations
- the number of censored observations
- the percentage of censored observations

The ODS name of this table is CensoredSummary.

Rank Statistics

The "Rank Statistics" table contains the test statistics of the nonparametric k-sample tests. The ODS name of this table is HomStats.

Covariance Matrix for the Log-Rank Statistics

The "Covariance Matrix for the Log-Rank Statistics" table is displayed if the log-rank k-sample test is requested. The ODS name of this table is LogrankHomCov.

Covariance Matrix for the Wilcoxon Statistics

The "Covariance Matrix for the Wilcoxon Statistics" table is displayed if the Wilcoxon k-sample test is requested. The ODS name of this table is WilHomCov.

Covariance Matrix for the Tarone Statistics

The "Covariance Matrix for the Tarone Statistics" table is displayed if the Tarone-Ware k-sample test is requested. The ODS name of this table is TaroneHomCov.

Covariance Matrix for the Peto Statistics

The "Covariance Matrix for the Peto Statistics" table is displayed if the Peto-Peto k-sample test is requested. The ODS name of this table is PetoHomCov.

Covariance Matrix for the ModPeto Statistics

The "Covariance Matrix for the ModPeto Statistics" table is displayed if the modified Peto-Peto k-sample test is requested. The ODS name of this table is ModPetoHomCov.

Covariance Matrix for the Fleming Statistics

The "Covariance Matrix for the Fleming Statistics" table is displayed if the Fleming-Harrington k-sample test is requested. The ODS name of this table is FlemingHomCov.

Legend for Strata

The "Legend for Strata" table is displayed if two or more variables are specified in the STRATA statement. The ODS name of this table is Legend.

Test of Equality over Strata

The "Test of Equality over Strata" table is displayed if an unstratified k-sample test is carried out. The table contains the chi-square statistics, degrees of freedom, and p-values of the nonparametric tests and the likelihood ratio test (which is based on the exponential distribution). The ODS name of this table is HomTests.

Stratified Test of Equality over Group

The "Stratified Test of Equality over Group" table is displayed if a stratified test is carried out. The tables contains the chi-square statistics, degrees of freedom, and p-values of the stratified tests. The ODS name of this table is HomTests.

Scores for Trend Test

The "Scores for Trend Test" table is displayed if you specify the TREND option in the STRATA statement. The table contains the set of scores used to construct the trend tests. The ODS name of this table is TrendScores.

Trend Tests

The "Trend Tests" table is displayed if you specify the TREND option in the STRATA statement. The table contains the results of the trend tests. The ODS name of this table is TrendTests.

Adjustment for Multiple Comparisons for the Log-Rank Test

The "Adjustment for Multiple Comparisons for the Log-Rank Test" table is displayed if the log-rank test and a multiple-comparison adjustment method are specified. The table contains the chi-square statistics and the raw and adjusted p-values of the paired comparisons. The ODS name of this table is SurvDiff.

Adjustment for Multiple Comparisons for the Wilcoxon Test

The "Adjustment for Multiple Comparisons for the Wilcoxon Test" table is displayed if the Wilcoxon test and a multiple-comparison method are specified. The table contains the chi-square statistics and the raw and adjusted p-values of the paired comparisons. The ODS name of this table is SurvDiff.

Adjustment for Multiple Comparisons for the Tarone Test

The "Adjustment for Multiple Comparisons for the Tarone Test" table is displayed if the Tarone-Ware test and a multiple-comparison method are specified. The table contains the chi-square statistics and the raw and adjusted p-values of the paired comparisons. The ODS name of this table is SurvDiff.

Adjustment for Multiple Comparisons for the Peto Test

The "Adjustment for Multiple Comparisons for the Peto Test" table is displayed if the Peto-Peto test and a multiple-comparison method are specified. The table contains the chi-square statistics and the raw and adjusted p-values of the paired comparisons. The ODS name of this table is SurvDiff.

Adjustment for Multiple Comparisons for the ModPeto Test

The "Adjustment for Multiple Comparisons for the ModPeto Test" table is displayed if the modified Peto-Peto test and a multiple-comparison method are specified. The table contains the chi-square statistics and the raw and adjusted p-values of the paired comparisons. The ODS name of this table is SurvDiff.

Adjustment for Multiple Comparisons for the Fleming Test

The "Adjustment for Multiple Comparisons for the Fleming Test" table is displayed if the Fleming-Harrington test and a multiple-comparison method are specified. The table contains the chi-square statistics and the raw and adjusted p-values of the paired comparisons. The ODS name of this table is SurvDiff.

Univariate Chi-Squares for the Log-Rank Test

The "Univariate Chi-Squares for the Log-Rank Test" table is displayed if you specify the TEST statement. The table displays the log-rank test results for individual variables in the TEST statement. The ODS name of this table is LogUniChiSq.

Covariance Matrix of the Log-Rank Statistics

The "Covariance Matrix of the Log-Rank Statistics" table is displayed if you specify the TEST statement. The table displays the estimated covariance matrix of the log-rank statistics for association. The ODS name of this table is LogTestCov.

Forward Stepwise Sequence of Chi-Squares for the Log-Rank Test

The "Forward Stepwise Sequence of Chi-Squares for the Log-Rank Test" table is displayed if you specify the TEST statement. The table contains the sequence of partial chi-square statistics for the log-rank test in the order of the greatest increase to the overall test statistic, the degrees of freedom of the partial chi-square statistics, the approximate probability values of the partial chi-square statistics, the chi-square increments for including the given variables, and the probability values of the chi-square increments. The ODS name of this table is LogForStepSeq.

Univariate Chi-Squares for the Wilcoxon Test

The "Univariate Chi-Squares for the Wilcoxon Test" table displays the Wilcoxon test results for individual variables in the TEST statement. The ODS name of this table is WilUniChiSq.

Covariance Matrix of the Wilcoxon Statistics

The "Covariance Matrix of the Wilcoxon Statistics" table is displayed if you specify the TEST statement. The table displays the estimated covariance matrix of the Wilcoxon statistics for association. The ODS name of this table is WilTestCov.

Forward Stepwise Sequence of Chi-Squares for the Wilcoxon Test

The "Forward Stepwise Sequence of Chi-Squares for the Wilcoxon Test" table is displayed if you specify the TEST statement. The table contains the sequence of partial chi-square statistics for the Wilcoxon test in the order of the greatest increase to the overall test statistic, the degrees of freedom of the partial chi-square statistics, the approximate probability values of the partial chi-square statistics, the chi-square increments for including the given variables, and the probability values of the chi-square increments. The ODS name of this table is WilForStepSeq.

Tables for Competing-Risks Analysis

Summary of Failure Outcomes

The "Summary of Failure Outcomes" table displays the following:

- the stratum identification, if you specify the STRATA statement
- the group identification, if you specify the GROUP= option in the STRATA statement
- the number of failure events of interest
- the number of competing events
- the number of censored observations

The ODS name of this table is FailureSummary.

Cumulative Incidence Function Estimates

The "Cumulative Incidence Function Estimates" table is displayed if you use the FAILCODE= option in the TIME statement to stipulate a competing-risk analysis. The table displays the following:

- the group identification, if you specify the GROUP= option in the STRATA statement
- the failure time of the event of interest

- the estimated cumulative incidence function
- the standard error estimate of the cumulative incidence estimator
- the lower and upper confidence limits of the cumulative incidence function

The ODS name of this table is CIF.

Gray's Test for Equality of Cumulative Incidence Functions

The "Gray's Test for Equality of Cumulative Incidence Functions" table is displayed if you specify the STRATA statement. The table displays the following:

- the failure code, if you specify more than one FAILCODE= option value
- the chi-square statistic of Gray's test (Gray 1988)
- the degrees of freedom
- the p-value

The ODS name of this table is GrayTest.

Tables for Restricted Mean Analysis

RMST Analysis Information

The "RMST Analysis Information" table is displayed if you specify the RMST option in the PROC LIFETEST statement to perform an RMST analysis. The table displays the following:

- the τ value that is used in computing the RMST

The ODS name of this table is RMSTInfo.

RMST Estimates

The "RMST Estimates" table is displayed if you specify the RMST option in the PROC LIFETEST statement to perform an RMST analysis. The table displays the following:

- the stratum identification, if you specify the STRATA statement
- the group identification, if you specify the GROUP= option in the STRATA statement
- the estimated RMST
- the standard error estimate of the RMST

The ODS name of this table is RMST.

RMST Test for Equality

The "RMST Test for Equality" table is displayed if you specify the STRATA statement and the RMST option in the PROC LIFETEST statement. The table displays the following:

- the source of the groups under comparison
- the chi-square statistic of the test
- the degrees of freedom
- the p-value

The ODS name of this table is RMSTTest.

Stratified RMST Test for Equality

The "Stratified RMST Test for Equality" table is displayed if you specify the GROUP= option in the STRATA statement and the RMST option in the PROC LIFETEST statement. The table displays the following:

- the source of the groups under comparison
- the chi-square statistic of the test
- the degrees of freedom
- the p-value

The ODS name of this table is RMSTStratifiedTest.

Restricted Mean Survival Time Comparisons

The "Restricted Mean Survival Time Comparisons" table is displayed if you specify the DIFF= option in the STRATA statement and the RMST option in the PROC LIFETEST statement. The table contains the chi-square statistics and the unadjusted and adjusted p-values of the paired comparisons. The ODS name of this table is RMSTDiff.

RMTL Analysis Information

The "RMTL Analysis Information" table if you specify the RMTL option in the PROC LIFETEST statement to perform an RMST analysis. The table displays the following:

- the τ value that is used in computing the RMTL

The ODS name of this table is RMTLInfo.

RMTL Estimates

The "RMTL Estimates" table is displayed if you specify the RMTL option in the PROC LIFETEST statement to perform an RMTL analysis. The table displays the following:

- the stratum identification, if you specify the STRATA statement
- the group identification, if you specify the GROUP= option in the STRATA statement
- the estimated RMTL
- the standard error estimate of the RMTL

The ODS name of this table is RMTL.

RMTL Test for Equality

The "RMTL Test for Equality" table is displayed if you specify the STRATA statement and the RMTL option in the PROC LIFETEST statement. The table displays the following:

- the source of the groups under comparison
- the chi-square statistic of the test
- the degrees of freedom
- the p-value

The ODS name of this table is RMTLTest.

Stratified RMTL Test for Equality

The "Stratified RMTL Test for Equality" table is displayed if you specify the GROUP= option in the STRATA statement and the RMTL option in the PROC LIFETEST statement. The table displays the following:

- the source of the groups under comparison
- the chi-square statistic of the test
- the degrees of freedom
- the p-value

The ODS name of this table is RMTLStratifiedTest.

Restricted Mean Time Lost Comparisons

The "Restricted Mean Time Lost Comparisons" table is displayed if you specify the DIFF= option in the STRATA statement and the RMTL option in the PROC LIFETEST statement. The table contains the chi-square statistics and the unadjusted and adjusted p-values of the paired comparisons. The ODS name of this table is RMTLDiff.

Plot Options Superseded by ODS Graphics

You can select one of the following three types of graphics in PROC LIFETEST: ODS, traditional, and line printer. ODS Graphics is the preferred method of creating graphs, superseding the other two.

When ODS Graphics is enabled, you can use the PLOTS= option in the PROC LIFETEST statement to create plots by using ODS Graphics. For more information about ODS Graphics options, see the PLOTS= option in the section "PROC LIFETEST Statement" on page 5762.

If ODS Graphics is not enabled and you specify the LINEPRINTER option, line printer plots are produced; otherwise traditional graphics are produced.

Table 77.4 summarizes the ways in which you can request graphics.

Table 77.4 Ways of Displaying Graphics

Graphics Result	ODS Graphics	PLOTS= Option Specified?	LINEPRINTER Option Specified?
ODS Graphics	Enabled	Yes	No
ODS Graphics survival plot	Enabled	No	No
Traditional graphics	Disabled	Yes	No
Line printer plot	Enabled	Yes	Yes
Line printer plot	Disabled	Yes	Yes
No graphics	Disabled	No	No
No graphics	Disabled	No	Yes
No graphics	Enabled	No	Yes

Table 77.5 summarizes the *options* available in the PROC LIFETEST statement for line printer and traditional graphics.

Table 77.5 Line Printer and Traditional Graphics Options Available in the PROC LIFETEST Statement

Option	Description
Line Printer Plots	
FORMCHAR(1,2,7,9)=	Defines the characters to be used for line printer plot axes
LINEPRINTER	Specifies that plots be produced by a line printer
MAXTIME=	Specifies the maximum time value for plotting
NOCENSPLOT	Suppresses the plot of censored observations
PLOTS=	Specifies the plots to display
Traditional Graphics	
ANNOTATE=	Specifies an Annotate data set that adds features to plots
CENSOREDSYMBOL=	Defines the symbol to be used for censored observations in plots
DESCRIPTION=	Specifies the string that appears in the description field of the PROC GREPLAY master menu for the plots

Table 77.5 *continued*

Option	Description
EVENTSYMBOL=	Specifies the symbol to be used for event observations in plots
GOUT=	Specifies the graphics catalog name for saving graphics output
LANNOTATE=	Specifies an input data set that contains variables for local annotation
MAXTIME=	Specifies the maximum time value for plotting
PLOTS=	Specifies the plots to display

The following *options* are used to produce line printer and traditional graphics:

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an input data set that contains appropriate variables for annotation of the traditional graphics. The ANNOTATE= option enables you to add features (for example, labels that explain extreme observations) to plots produced on graphics devices. The ANNOTATE= option cannot be used if you specify LINEPRINTER option or if ODS Graphics is enabled. The data set specified must be an ANNOTATE= type data set, as described in *SAS/GRAPH: Reference*.

The data set specified with the ANNOTATE= option in the PROC LIFETEST statement is "global" in the sense that the information in this data set is displayed in every plot produced by a single invocation of PROC LIFETEST.

CENSOREDSYMBOL=*name* | *'string'*

CS=*name* | *'string'*

specifies the symbol value for the censored observations in traditional graphics. The value, *name* or *'string'*, is the symbol value specification allowed in SAS/GRAPH software. The default is CS=CIRCLE. If you want to omit plotting the censored observations, specify CS=NONE. The CENSOREDSYMBOL= option cannot be used if you specify LINEPRINTER option or if you enable ODS Graphics.

DESCRIPTION='*string*'

DES='*string*'

specifies a descriptive string of up to 256 characters that appears in the "Description" field of the traditional graphics catalog. The description does not appear in the plots. By default, PROC LIFETEST assigns a description of the form PLOT OF *vname* versus *hname*, where *vname* and *hname* are the names of the *y* variable and the *x* variable, respectively. The DESCRIPTION= option cannot be used if you specify the LINEPRINTER option or if you enable ODS Graphics.

EVENTSYMBOL=*name* | '*string*'

ES=*name* | '*string*'

specifies the symbol value for the event observations in traditional graphics. The value, *name* or *'string'*, is the symbol value specification allowed in SAS/GRAPH software. The default is ES=NONE. The EVENTSYMBOL= option cannot be used if you specify the LINEPRINTER option or if you enable ODS Graphics.

FORMCHAR(1,2,7,9)=*'string'*

defines the characters to be used for constructing the vertical and horizontal axes of the line printer plots. The string should be four characters. The first and second characters define the vertical and horizontal bars, respectively, which are also used in drawing the steps of the Kaplan-Meier, Breslow, or Fleming-Harrington survival curve. The third character defines the tick mark for the axes, and the fourth character defines the lower left corner of the plot. The default is FORMCHAR(1,2,7,9)='|-+-'. Any character or hexadecimal string can be used to customize the plot appearance. If you use hexadecimals, you must put an x after the closing quote. For example, to send the plot output to a printer with the IBM graphics character set (1 or 2), specify the following:

```
formchar(1,2,7,9)='B3C4C5C0'x
```

See the chapter titled "The PLOT Procedure" in the *Base SAS Procedures Guide* for further information.

GOUT=*graphics-catalog*

specifies the graphics catalog for saving traditional graphics output from PROC LIFETEST. The default is Work.Gseg. The GOUT= option cannot be used if you specify the LINEPRINTER option or if you enable ODS Graphics. For more information, see the chapter titled "The GREPLAY Procedure" in *SAS/GRAPH: Reference*.

LANNOTATE=*SAS-data-set*

LANN=*SAS-data-set*

specifies an input data set that contains variables for local annotation of traditional graphics. You can use the LANNOTATE= option to specify a different annotation for each BY group, in which case the BY variables must be included in the LANNOTATE= data set. The LANNOTATE= option cannot be used if you specify the LINEPRINTER option or if you enable ODS Graphics. The data set specified must be an ANNOTATE= type data set, as described in *SAS/GRAPH: Reference*.

If there is no BY-group processing, the ANNOTATE= and LANNOTATE= options have the same effects.

LINEPRINTER

LS

specifies that plots are produced by a line printer instead of by a graphical device.

MAXTIME=*value*

specifies the maximum value of the time variable allowed on the plots so that outlying points do not determine the scale of the time axis of the plots. This option affects only the displayed plots and has no effect on any calculations.

NOCENSPLOT

NOCENS

requests that the plot of censored observations be suppressed when the LINEPRINTER and PLOTS= options are specified. This option is not needed when the life-table method is used to compute the survival estimates, because the plot of censored observations is not produced.

Line Printer PLOTS= Option

PLOTS=*plot-request*

PLOTS=(*plot-requests***)**

controls the line printer plots produced. You must also specify the LINEPRINTER option to obtain line printer plots. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. Here are some examples:

```
plots=s
plots=(s ls lls)
```

The *plot-requests* include the following:

CENSORED

C

specifies a plot of censored observations. This option is available for METHOD=KM, METHOD=BRESLOW, or METHOD=FH only.

SURVIVAL

S

specifies a plot of the estimated SDF versus time.

LOGSURV

LS

specifies a plot of the negative log of the estimated SDF versus time.

LOGLOGS

LLS

specifies a plot of the log of the negative log of the estimated SDF versus the log of time.

HAZARD

H

specifies a plot of the estimated hazard function versus time (life-table method only).

PDF

P

specifies a plot of the estimated probability density function versus time (life-table method only).

Traditional Graphics PLOTS= Option

PLOTS=*plot-request* **<(NAME=***name* | '*string*'**)>**

PLOTS=(*plot-request* **<(NAME=***name* | '*string*'**)> <, . . . ,** *plot-request* **<(NAME=***name* | '*string*'**)> >)**

controls plots produced in traditional graphics. To obtain traditional graphics, you must neither enable ODS Graphics nor specify the LINEPRINTER option. For each *plot-request*, you can use the NAME= option to specify a name to identify the plot. The name can be specified as a SAS name or as a quoted string of up to 256 characters. Only the first eight characters are used as the entry name in the GOUT= catalog. The *plot-requests* include the following:

SURVIVAL

S

plots the estimated survivor functions versus time.

LOGSURV

LS

plots the negative log of estimated survivor functions versus time.

LOGLOGS

LLS

plots the log of negative log of estimated survivor functions versus the log of time.

HAZARD

H

plots estimated hazard function versus time (life-table method only).

PDF

P

plots the estimated probability density function versus time (life-table method only).

When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. Here are some examples:

```
plots=s
plots=(s(name=Surv2), h(name=Haz2))
```

The latter requests a plot of the estimated survivor function versus time and a plot of the estimated hazard function versus time, with Surv2 and Haz2 as their names in the GOUT= catalog, respectively.

ODS Table Names

PROC LIFETEST assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 77.6. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 77.6 ODS Tables Produced by PROC LIFETEST

ODS Table Name	Description	Statement	Option
BreslowEstimates	Breslow estimates	PROC	METHOD=BRESLOW
CensoredSummary	Number of event and censored observations	PROC	METHOD=BRESLOW | FH | PL
CIF	Cumulative incidence function estimates	TIME	EVENTCODE
FailureSummary	Summary of failure outcomes for competing-risks data	TIME	EVENTCODE

Table 77.6 *continued*

ODS Table Name	Description	Statement	Option
FlemingEstimates	Fleming-Harrington estimates	PROC	METHOD=FH
FlemingHomCov	Covariance matrix for *k*-sample FLEMING statistics	STRATA	TEST=FLEMING
GrayTest	Results of *k*-sample test of Gray (1988) comparing CIFs	TIME, STRATA	EVENTCODE
HomStats	Test statistics for *k*-sample tests	STRATA	Default
HomTests	Results of *k*-sample tests	STRATA	Default
Legend	Strata legend for two or more strata variables	STRATA	
LifetableEstimates	Life-table survival estimates	PROC	METHOD=LT
LogForStepSeq	Forward stepwise sequence for the log-rank statistics for association	TEST	Default
LogrankHomCov	Covariance matrix for *k*-sample log-rank statistics	STRATA	TEST=LOGRANK
LogTestCov	Covariance matrix for log-rank statistics for association	TEST	Default
LogUniChisq	Univariate chi-squares for log-rank statistics for association	TEST	Default
Means	Mean and standard error of survival times	PROC	METHOD=PL
ModPetoHomCov	Covariance matrix for *k*-sample MODPETO statistics	STRATA	TEST=MODPETO
PetoHomCov	Covariance matrix for *k*-sample PETO statistics	STRATA	TEST=PETO
ProductLimitEstimates	Product-limit survival estimates	PROC	METHOD=PL
Quartiles	Quartiles of the survival times	PROC	METHOD=BRESLOW | FH | PL
RMST	Restricted mean survival time estimates	PROC	RMST
RMSTDiff	Adjustments for multiple comparisons	STRATA	ADJUST=
RMSTInfo	RMST analysis information	PROC	RMST
RMSTStratifiedTest	Results of stratified *k*-sample RMST test	PROC, STRATA	RMST, GROUP=
RMSTTest	Results of *k*-sample RMST test	PROC, STRATA	RMST
RMTL	Restricted mean time lost estimates	PROC	RMTL
RMTLDiff	Adjustments for multiple comparisons	STRATA	ADJUST=
RMTLInfo	RMTL analysis information	PROC	RMTL
RMTLStratifiedTest	Results of stratified *k*-sample RMTL test	PROC, STRATA	RMTL, GROUP=

Table 77.6 *continued*

ODS Table Name	Description	Statement	Option
RMTLTest	Results of *k*-sample RMTL test	PROC, STRATA	RMTL
SimDetails	Details of quantile simulations	STRATA	ADJUST=SIMULATE(REPORT)
SimResults	Quantile simulation results	STRATA	ADJUST=SIMULATE(REPORT)
SurvDiff	Adjustments for multiple comparisons	STRATA	ADJUST=
TaroneHomCov	Covariance matrix for *k*-sample TARONE statistics	STRATA	TEST=TARONE
TrendScores	Scores used to construct trend tests	STRATA	TREND
TrendTests	Results of trend tests	STRATA	TREND
WilcoxonHomCov	Covariance matrix for *k*-sample WILCOXON statistics	STRATA	TEST=WILCOXON
WilForStepSeq	Forward stepwise sequence for the log-rank statistics for association	TEST	Default
WilTestCov	Covariance matrix for log-rank statistics for association	TEST	Default
WilUniChiSq	Univariate chi-squares for Wilcoxon statistics for association	TEST	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

The survival plot is produced by default; other graphs are produced by using the PLOTS= option in the PROC LIFETEST statement. You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC LIFETEST generates are listed in Table 77.7, along with the required keywords for the PLOTS= option.

Table 77.7 Graphs Produced by PROC LIFETEST

ODS Graph Name	Plot Description	PLOTS= Option
cifPlot	Cumulative incidence function	CIF
cifPlot	Cumulative incidence function with pointwise confidence limits	CIF(CL)
cifPlot	Cumulative incidence function with Gray's test	CIF(TEST)
DensityPlot	Density function for life-table method	PDF
FailurePlot	Cumulative distribution function	SURVIVAL(FAILURE)
HazardPlot	Hazard function for life-table method or smoothed hazard for product-limit, Breslow, or Fleming-Harrington method	HAZARD
LogNegLogSurvivalPlot	Log(–log(survivor function))	LOGLOGS
NegLogSurvivalPlot	Log(survivor function)	LOGSURV
RMSTPlot	Restricted mean survival time curve	RMST
RMSTPlot	Restricted mean survival time curve with pointwise confidence limits	RMST(CL)
RMTLPlot	Restricted mean time lost curve	RMTL
RMTLPlot	Restricted mean time lost curve with pointwise confidence limits	RMTL(CL)
SurvivalPlot	Survivor function	SURVIVAL
SurvivalPlot	Survivor function with number of subjects at risk	SURVIVAL(ATRISK)
SurvivalPlot	Survivor function with pointwise confidence limits	SURVIVAL(CL)
SurvivalPlot	Survivor function with equal-precision band	SURVIVAL(CB=EP)
SurvivalPlot	Survivor function with Hall-Wellner band	SURVIVAL(CB=HW)
SurvivalPlot	Survivor function with homogeneity test	SURVIVAL(TEST)

Additional Dynamic Variables for Survival Plots Using ODS Graphics

PROC LIFETEST passes a number of summary statistics as dynamic variables to the ODS Graphics for survival plots. Table 77.8 and Table 77.9 list these additional dynamic variables for the Kaplan-Meier curves and the life-table curves, respectively. These dynamic variables are not declared in the templates for the survival curves, but you can declare them and use them to enhance the default plots. The names of the dynamic variables depend on the STRATA= suboption of the PLOTS=SURVIVAL option: STRATA=INDIVIDUAL produces a separate plot for each stratum, and STRATA=OVERALL produces one plot with overlaid curves.

Table 77.8 Additional Dynamic Variables for `Stat.Graphics.ProductLimitSurvival`

STRATA=	Dynamic	Description
OVERLAY	StrVal*j*	Label for the *j*th stratum
	NObs*j*	Number of observations in the *j*th stratum

Table 77.8 *continued*

STRATA=	Dynamic	Description
	NEvent*j*	Number of events in the *j*th stratum
	Median*j*	Median survival time of the *j*th stratum
	LowerMedian*j*	Lower median survival time of the *j*th stratum
	UpperMedian*j*	Upper median survival time of the *j*th stratum
	PctMedianConfid	Confidence of the median intervals in percent
INDIVIDUAL	NObs	Number of observations
	NEvent	Number of events
	Median	Median survival time
	LowerMedian	Lower median survival time
	UpperMedian	Upper median survival time
	PctMedianConfid	Confidence of the median interval in percent

Table 77.9 Additional Dynamic Variables for `Stat.Graphics.LifetableSurvival`

STRATA=	Dynamic	Description
OVERLAY	StrVal*j*	Label for the *j*th stratum
	NObs*j*	Number of observations in the *j*th stratum
	NEvent*j*	Number of events in the *j*th stratum
INDIVIDUAL	NObs	Number of observations
	NEvent	Number of events

For information about all of the dynamic variables that are available for use in the ODS Graphics survival plot, see the section "Dynamic Variables" on page 1000 in Chapter 26, "Customizing the Kaplan-Meier Survival Plot." For the use of the particular dynamic variables shown in this section, see the sections "Adding a Small Inset Table with Event Information" on page 979 and "Adding an External Table with Event Information" on page 981 in Chapter 26, "Customizing the Kaplan-Meier Survival Plot."

Modifying the Survival Plots

PROC LIFETEST, like other statistical procedures, provides a PLOTS= option and other options for modifying its graphical output without requiring template changes. Those options are sufficient for most purposes, and the following subsections of the section "Controlling the Survival Plot by Specifying Procedure Options" on page 937 in Chapter 26, "Customizing the Kaplan-Meier Survival Plot," provide examples:

- "Enabling ODS Graphics and the Default Kaplan-Meier Plot" on page 937
- "Individual Survival Plots" on page 939
- "Hall-Wellner Confidence Bands and Homogeneity Test" on page 940
- "Equal-Precision Bands" on page 941
- "Displaying the Patients-at-Risk Table inside the Plot" on page 943

- "Displaying the Patients-at-Risk Table outside the Plot" on page 945
- "Modifying At-Risk Table Times" on page 946
- "Reordering the Groups" on page 949
- "Suppressing the Censored Observations" on page 951
- "Failure Plots" on page 952

When those options are not sufficient, you can use a set of macros and macro variables to modify the graph templates. Using these macros and macro variables is easier than directly modifying the graph templates. The following subsections of the section "Controlling the Survival Plot by Modifying Graph Templates" on page 958 in Chapter 26, "Customizing the Kaplan-Meier Survival Plot," provide examples:

- "Changing the Plot Title" on page 960
- "Modifying the Y Axis" on page 961
- "Changing the Line Thickness" on page 963
- "Changing the Group Color" on page 964
- "Changing the Line Pattern" on page 966
- "Changing the Font" on page 967
- "Changing the Legend and Inset Position" on page 969
- "Changing How the Censored Points Are Displayed" on page 970
- "Adding a Y-Axis Reference Line" on page 971
- "Changing the Homogeneity Test Inset" on page 974
- "Changing the Second Title and Adding a Footnote" on page 976
- "Adding a Small Inset Table with Event Information" on page 979
- "Adding an External Table with Event Information" on page 981
- "Suppressing the Legend" on page 983
- "Kaplan-Meier Plot with Event Table and Other Customizations" on page 984

Examples: LIFETEST Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 77.1: Product-Limit Estimates and Tests of Association

(View the complete code for this example (liftex1.sas) in the example repository.)

The data presented in Appendix I of Kalbfleisch and Prentice (1980) are coded in the following DATA step. The response variable, SurvTime, is the survival time in days of a lung cancer patient. Negative values of SurvTime are censored values. The covariates are Cell (type of cancer cell), Therapy (type of therapy: standard or test), Prior (prior therapy: 0=no, 10=yes), Age (age in years), DiagTime (time in months from diagnosis to entry into the trial), and Kps (performance status). A censoring indicator variable Censor is created from the data, with the value 1 indicating a censored time and the value 0 indicating an event time. Since there are only two types of therapy, an indicator variable, Treatment, is constructed for therapy type, with value 0 for standard therapy and value 1 for test therapy.

```
data VALung;
   drop check m;
   retain Therapy Cell;
   infile cards column=column;
   length Check $ 1;
   label SurvTime='Failure or Censoring Time'
      Kps='Karnofsky Index'
      DiagTime='Months till Randomization'
      Age='Age in Years'
      Prior='Prior Treatment?'
      Cell='Cell Type'
      Therapy='Type of Treatment'
      Treatment='Treatment Indicator';
   M=Column;
   input Check $ @@;
   if M>Column then M=1;
   if Check='s'|Check='t' then input @M Therapy $ Cell $ ;
   else input @M SurvTime Kps DiagTime Age Prior @@;
   if SurvTime > .;
   censor=(SurvTime<0);
   SurvTime=abs(SurvTime);
   Treatment=(Therapy='test');
   datalines;
standard squamous
 72 60  7 69  0   411 70  5 64 10   228 60  3 38  0   126 60  9 63 10
118 70 11 65 10    10 20  5 49  0    82 40 10 69 10   110 80 29 68  0
314 50 18 43  0  -100 70  6 70  0    42 60  4 81  0     8 40 58 63 10
144 30  4 63  0   -25 80  9 52 10    11 70 11 48 10
standard small
 30 60  3 61  0   384 60  9 42  0     4 40  2 35  0    54 80  4 63 10
 13 60  4 56  0  -123 40  3 55  0   -97 60  5 67  0   153 60 14 63 10
 59 30  2 65  0   117 80  3 46  0    16 30  4 53 10   151 50 12 69  0
 22 60  4 68  0    56 80 12 43 10    21 40  2 55 10    18 20 15 42  0
139 80  2 64  0    20 30  5 65  0    31 75  3 65  0    52 70  2 55  0
287 60 25 66 10    18 30  4 60  0    51 60  1 67  0   122 80 28 53  0
 27 60  8 62  0    54 70  1 67  0     7 50  7 72  0    63 50 11 48  0
392 40  4 68  0    10 40 23 67 10
standard adeno
  8 20 19 61 10    92 70 10 60  0    35 40  6 62  0   117 80  2 38  0
132 80  5 50  0    12 50  4 63 10   162 80  5 64  0     3 30  3 43  0
 95 80  4 34  0
standard large
177 50 16 66 10   162 80  5 62  0   216 50 15 52  0   553 70  2 47  0
278 60 12 63  0    12 40 12 68 10   260 80  5 45  0   200 80 12 41 10
156 70  2 66  0  -182 90  2 62  0   143 90  8 60  0   105 80 11 66  0
103 80  5 38  0   250 70  8 53 10   100 60 13 37 10
test squamous
999 90 12 54 10   112 80  6 60  0   -87 80  3 48  0  -231 50  8 52 10
242 50  1 70  0   991 70  7 50 10   111 70  3 62  0     1 20 21 65 10
587 60  3 58  0   389 90  2 62  0    33 30  6 64  0    25 20 36 63  0
357 70 13 58  0   467 90  2 64  0   201 80 28 52 10     1 50  7 35  0
 30 70 11 63  0    44 60 13 70 10   283 90  2 51  0    15 50 13 40 10
test small
```

```
 25 30  2 69  0  -103 70 22 36 10    21 20  4 71  0    13 30  2 62  0
 87 60  2 60  0     2 40 36 44 10    20 30  9 54 10     7 20 11 66  0
 24 60  8 49  0    99 70  3 72  0     8 80  2 68  0    99 85  4 62  0
 61 70  2 71  0    25 70  2 70  0    95 70  1 61  0    80 50 17 71  0
 51 30 87 59 10    29 40  8 67  0
test adeno
 24 40  2 60  0    18 40  5 69 10   -83 99  3 57  0    31 80  3 39  0
 51 60  5 62  0    90 60 22 50 10    52 60  3 43  0    73 60  3 70  0
  8 50  5 66  0    36 70  8 61  0    48 10  4 81  0     7 40  4 58  0
140 70  3 63  0   186 90  3 60  0    84 80  4 62 10    19 50 10 42  0
 45 40  3 69  0    80 40  4 63  0
test large
 52 60  4 45  0   164 70 15 68 10    19 30  4 39 10    53 60 12 66  0
 15 30  5 63  0    43 60 11 49 10   340 80 10 64 10   133 75  1 65  0
111 60  5 64  0   231 70 18 67 10   378 80  4 65  0    49 30  3 37  0
;
```

In the following statements, PROC LIFETEST is invoked to compute the product-limit estimate of the survivor function for each type of cancer cell and to analyze the effects of the variables Age, Prior, DiagTime, Kps, and Treatment on the survival of the patients. These prognostic factors are specified in the TEST statement, and the variable Cell is specified in the STRATA statement. ODS Graphics must be enabled before producing graphs. Graphical displays of the product-limit estimates (S), the negative log estimates (LS), and the log of negative log estimates (LLS) are requested through the PLOTS= option in the PROC LIFETEST statement. Because of a few large survival times, a MAXTIME of 600 is used to set the scale of the time axis; that is, the time scale extends from 0 to a maximum of 600 days in the plots. The variable Therapy is specified in the ID statement to identify the type of therapy for each observation in the product-limit estimates. The OUTTEST option specifies the creation of an output data set named Test to contain the rank test matrices for the covariates.

```
ods graphics on;
proc lifetest data=VALung plots=(s,ls,lls) outtest=Test maxtime=600;
   time SurvTime*Censor(1);
   id Therapy;
   strata Cell;
   test Age Prior DiagTime Kps Treatment;
run;
ods graphics off;
```

Output 77.1.1 through Output 77.1.4 display the product-limit estimates of the survivor functions for the four cell types. Summary statistics of the survival times are also shown. The median survival times are 51 days, 156 days, 51 days, and 118 days for patients with adeno cells, large cells, small cells, and squamous cells, respectively.

Output 77.1.1 Estimation Results for Adeno Cells

The LIFETEST Procedure

Stratum 1: Cell Type = adeno

Product-Limit Survival Estimates

SurvTime		Survival	Failure	Survival Standard Error	Number Failed	Number Left	Therapy
0.000		1.0000	0	0	0	27	
3.000		0.9630	0.0370	0.0363	1	26	standard
7.000		0.9259	0.0741	0.0504	2	25	test
8.000		.	.	.	3	24	standard
8.000		0.8519	0.1481	0.0684	4	23	test
12.000		0.8148	0.1852	0.0748	5	22	standard
18.000		0.7778	0.2222	0.0800	6	21	test
19.000		0.7407	0.2593	0.0843	7	20	test
24.000		0.7037	0.2963	0.0879	8	19	test
31.000		0.6667	0.3333	0.0907	9	18	test
35.000		0.6296	0.3704	0.0929	10	17	standard
36.000		0.5926	0.4074	0.0946	11	16	test
45.000		0.5556	0.4444	0.0956	12	15	test
48.000		0.5185	0.4815	0.0962	13	14	test
51.000		0.4815	0.5185	0.0962	14	13	test
52.000		0.4444	0.5556	0.0956	15	12	test
73.000		0.4074	0.5926	0.0946	16	11	test
80.000		0.3704	0.6296	0.0929	17	10	test
83.000	*	.	.	.	17	9	test
84.000		0.3292	0.6708	0.0913	18	8	test
90.000		0.2881	0.7119	0.0887	19	7	test
92.000		0.2469	0.7531	0.0850	20	6	standard
95.000		0.2058	0.7942	0.0802	21	5	standard
117.000		0.1646	0.8354	0.0740	22	4	standard
132.000		0.1235	0.8765	0.0659	23	3	standard
140.000		0.0823	0.9177	0.0553	24	2	test
162.000		0.0412	0.9588	0.0401	25	1	standard
186.000		0	1.0000	.	26	0	test

Note: The marked survival times are censored observations.

Output 77.1.2 Estimation Results for Large Cells

The LIFETEST Procedure

Stratum 2: Cell Type = large

Product-Limit Survival Estimates

SurvTime		Survival	Failure	Survival Standard Error	Number Failed	Number Left	Therapy
0.000		1.0000	0	0	0	27	
12.000		0.9630	0.0370	0.0363	1	26	standard
15.000		0.9259	0.0741	0.0504	2	25	test
19.000		0.8889	0.1111	0.0605	3	24	test
43.000		0.8519	0.1481	0.0684	4	23	test
49.000		0.8148	0.1852	0.0748	5	22	test
52.000		0.7778	0.2222	0.0800	6	21	test
53.000		0.7407	0.2593	0.0843	7	20	test
100.000		0.7037	0.2963	0.0879	8	19	standard
103.000		0.6667	0.3333	0.0907	9	18	standard
105.000		0.6296	0.3704	0.0929	10	17	standard
111.000		0.5926	0.4074	0.0946	11	16	test
133.000		0.5556	0.4444	0.0956	12	15	test
143.000		0.5185	0.4815	0.0962	13	14	standard
156.000		0.4815	0.5185	0.0962	14	13	standard
162.000		0.4444	0.5556	0.0956	15	12	standard
164.000		0.4074	0.5926	0.0946	16	11	test
177.000		0.3704	0.6296	0.0929	17	10	standard
182.000	*	.	.	.	17	9	standard
200.000		0.3292	0.6708	0.0913	18	8	standard
216.000		0.2881	0.7119	0.0887	19	7	standard
231.000		0.2469	0.7531	0.0850	20	6	test
250.000		0.2058	0.7942	0.0802	21	5	standard
260.000		0.1646	0.8354	0.0740	22	4	standard
278.000		0.1235	0.8765	0.0659	23	3	standard
340.000		0.0823	0.9177	0.0553	24	2	test
378.000		0.0412	0.9588	0.0401	25	1	test
553.000		0	1.0000	.	26	0	standard

Note: The marked survival times are censored observations.

Output 77.1.3 Estimation Results for Small Cells

The LIFETEST Procedure

Stratum 3: Cell Type = small

Product-Limit Survival Estimates

SurvTime		Survival	Failure	Survival Standard Error	Number Failed	Number Left	Therapy
0.000		1.0000	0	0	0	48	
2.000		0.9792	0.0208	0.0206	1	47	test
4.000		0.9583	0.0417	0.0288	2	46	standard
7.000		.	.	.	3	45	standard
7.000		0.9167	0.0833	0.0399	4	44	test
8.000		0.8958	0.1042	0.0441	5	43	test
10.000		0.8750	0.1250	0.0477	6	42	standard
13.000		.	.	.	7	41	standard
13.000		0.8333	0.1667	0.0538	8	40	test
16.000		0.8125	0.1875	0.0563	9	39	standard
18.000		.	.	.	10	38	standard
18.000		0.7708	0.2292	0.0607	11	37	standard
20.000		.	.	.	12	36	standard
20.000		0.7292	0.2708	0.0641	13	35	test
21.000		.	.	.	14	34	standard
21.000		0.6875	0.3125	0.0669	15	33	test
22.000		0.6667	0.3333	0.0680	16	32	standard
24.000		0.6458	0.3542	0.0690	17	31	test
25.000		.	.	.	18	30	test
25.000		0.6042	0.3958	0.0706	19	29	test
27.000		0.5833	0.4167	0.0712	20	28	standard
29.000		0.5625	0.4375	0.0716	21	27	test
30.000		0.5417	0.4583	0.0719	22	26	standard
31.000		0.5208	0.4792	0.0721	23	25	standard
51.000		.	.	.	24	24	standard
51.000		0.4792	0.5208	0.0721	25	23	test
52.000		0.4583	0.5417	0.0719	26	22	standard
54.000		.	.	.	27	21	standard
54.000		0.4167	0.5833	0.0712	28	20	standard
56.000		0.3958	0.6042	0.0706	29	19	standard
59.000		0.3750	0.6250	0.0699	30	18	standard
61.000		0.3542	0.6458	0.0690	31	17	test
63.000		0.3333	0.6667	0.0680	32	16	standard
80.000		0.3125	0.6875	0.0669	33	15	test
87.000		0.2917	0.7083	0.0656	34	14	test
95.000		0.2708	0.7292	0.0641	35	13	test
97.000	*	.	.	.	35	12	standard
99.000		.	.	.	36	11	test
99.000		0.2257	0.7743	0.0609	37	10	test
103.000	*	.	.	.	37	9	test
117.000		0.2006	0.7994	0.0591	38	8	standard
122.000		0.1755	0.8245	0.0567	39	7	standard
123.000	*	.	.	.	39	6	standard

Output 77.1.3 *continued*

The LIFETEST Procedure

Stratum 3: Cell Type = small

Product-Limit Survival Estimates

SurvTime	Survival	Failure	Survival Standard Error	Number Failed	Number Left	Therapy
139.000	0.1463	0.8537	0.0543	40	5	standard
151.000	0.1170	0.8830	0.0507	41	4	standard
153.000	0.0878	0.9122	0.0457	42	3	standard
287.000	0.0585	0.9415	0.0387	43	2	standard
384.000	0.0293	0.9707	0.0283	44	1	standard
392.000	0	1.0000	.	45	0	standard

Note: The marked survival times are censored observations.

Output 77.1.4 Estimation Results for Squamous Cells

The LIFETEST Procedure

Stratum 4: Cell Type = squamous

Product-Limit Survival Estimates

SurvTime		Survival	Failure	Survival Standard Error	Number Failed	Number Left	Therapy
0.000		1.0000	0	0	0	35	
1.000		.	.	.	1	34	test
1.000		0.9429	0.0571	0.0392	2	33	test
8.000		0.9143	0.0857	0.0473	3	32	standard
10.000		0.8857	0.1143	0.0538	4	31	standard
11.000		0.8571	0.1429	0.0591	5	30	standard
15.000		0.8286	0.1714	0.0637	6	29	test
25.000		0.8000	0.2000	0.0676	7	28	test
25.000	*	.	.	.	7	27	standard
30.000		0.7704	0.2296	0.0713	8	26	test
33.000		0.7407	0.2593	0.0745	9	25	test
42.000		0.7111	0.2889	0.0772	10	24	standard
44.000		0.6815	0.3185	0.0794	11	23	test
72.000		0.6519	0.3481	0.0813	12	22	standard
82.000		0.6222	0.3778	0.0828	13	21	standard
87.000	*	.	.	.	13	20	test
100.000	*	.	.	.	13	19	standard
110.000		0.5895	0.4105	0.0847	14	18	standard
111.000		0.5567	0.4433	0.0861	15	17	test
112.000		0.5240	0.4760	0.0870	16	16	test
118.000		0.4912	0.5088	0.0875	17	15	standard
126.000		0.4585	0.5415	0.0876	18	14	standard
144.000		0.4257	0.5743	0.0873	19	13	standard
201.000		0.3930	0.6070	0.0865	20	12	test
228.000		0.3602	0.6398	0.0852	21	11	standard
231.000	*	.	.	.	21	10	test
242.000		0.3242	0.6758	0.0840	22	9	test
283.000		0.2882	0.7118	0.0820	23	8	test
314.000		0.2522	0.7478	0.0793	24	7	standard
357.000		0.2161	0.7839	0.0757	25	6	test
389.000		0.1801	0.8199	0.0711	26	5	test
411.000		0.1441	0.8559	0.0654	27	4	standard
467.000		0.1081	0.8919	0.0581	28	3	test
587.000		0.0720	0.9280	0.0487	29	2	test
991.000		0.0360	0.9640	0.0352	30	1	test
999.000		0	1.0000	.	31	0	test

Note: The marked survival times are censored observations.

The distribution of event and censored observations among the four cell types is summarized in Output 77.1.5.

Output 77.1.5 Summary of Censored and Uncensored Values

Summary of the Number of Censored and Uncensored Values					
Stratum	**Cell**	**Total**	**Failed**	**Censored**	**Percent Censored**
1	adeno	27	26	1	3.70
2	large	27	26	1	3.70
3	small	48	45	3	6.25
4	squamous	35	31	4	11.43
Total		137	128	9	6.57

The graph of the estimated survivor functions is shown in Output 77.1.6. The adeno cell curve and the small cell curve are much closer to each other than they are to the large cell curve or the squamous cell curve. The survival rates of the adeno cell patients and the small cell patients decrease rapidly to approximately 29% in 90 days. Shapes of the large cell curve and the squamous cell curve are quite different, although both decrease less rapidly than those of the adeno and small cells. The squamous cell curve decreases more rapidly initially than the large cell curve, but the role is reversed in the later period.

Output 77.1.6 Graph of the Estimated Survivor Functions

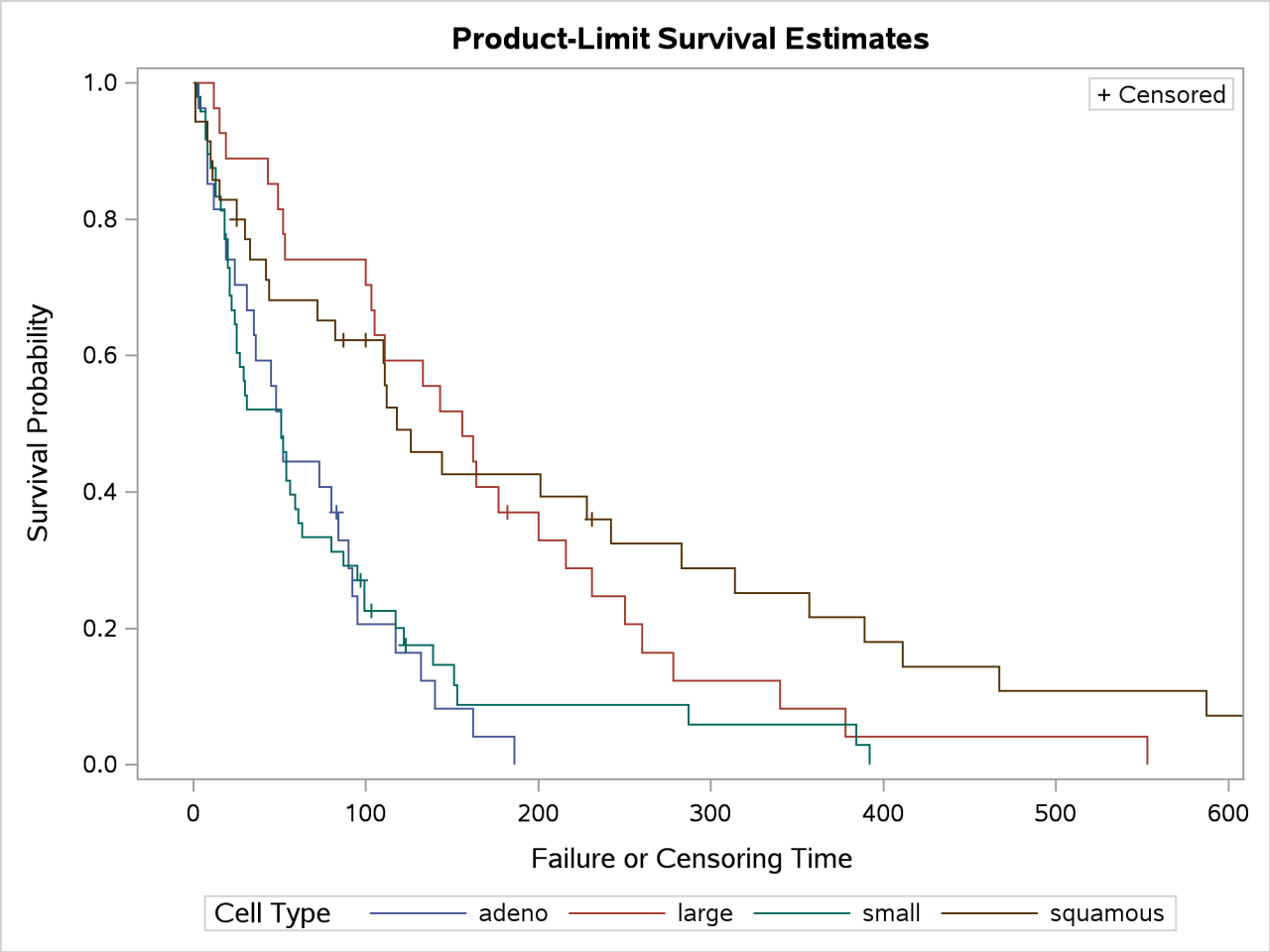

The graph of the negative log of the estimated survivor functions is displayed in Output 77.1.7. Output 77.1.8 displays the log of the negative log of the estimated survivor functions against the log of time.

Output 77.1.7 Graph of Negative Log of the Estimated Survivor Functions

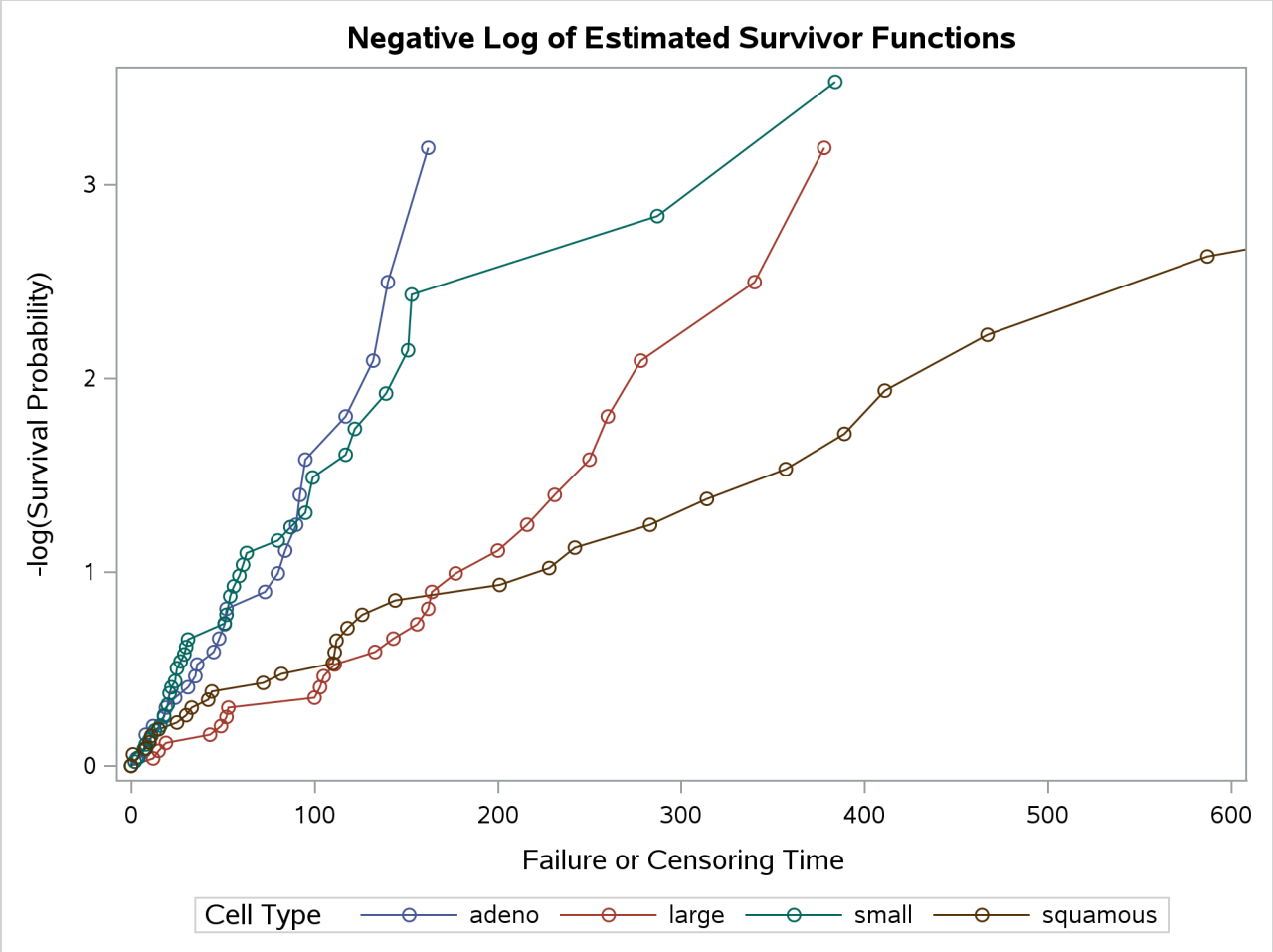

Output 77.1.8 Graph of Log of the Negative Log of the Estimated Survivor Functions

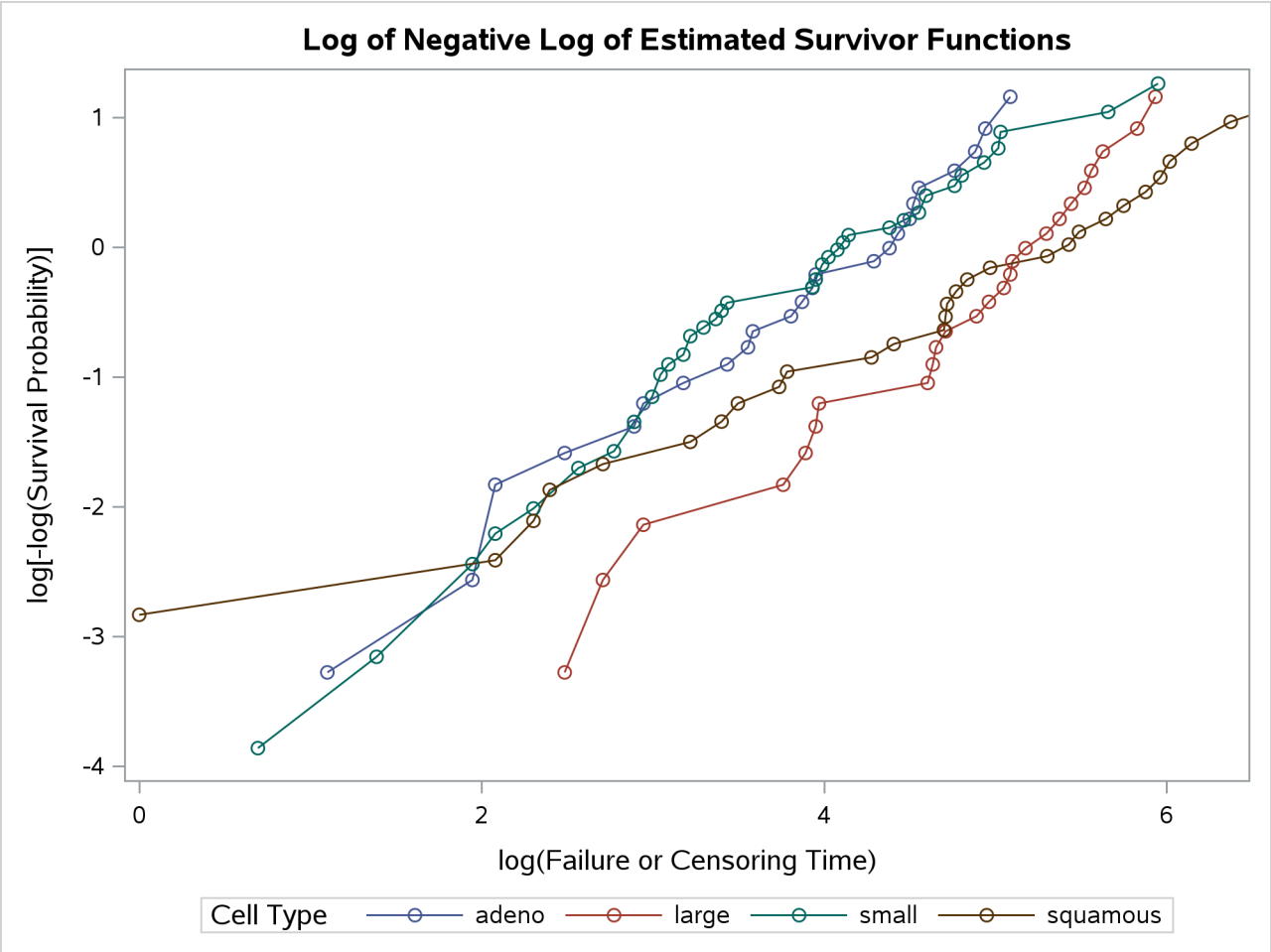

Results of the homogeneity tests across cell types are given in Output 77.1.9. The log-rank and Wilcoxon statistics and their corresponding covariance matrices are displayed. Also given is a table that consists of the approximate chi-square statistics, degrees of freedom, and p-values for the log-rank, Wilcoxon, and likelihood ratio tests. All three tests indicate strong evidence of a significant difference among the survival curves for the four types of cancer cells ($p < 0.0001$).

Output 77.1.9 Homogeneity Tests across Cell Types

Rank Statistics		
Cell	**Log-Rank**	**Wilcoxon**
adeno	10.306	697.0
large	-8.549	-1085.0
small	14.898	1278.0
squamous	-16.655	-890.0

Output 77.1.9 *continued*

Covariance Matrix for the Log-Rank Statistics				
Cell	**adeno**	**large**	**small**	**squamous**
adeno	12.9662	-4.0701	-4.4087	-4.4873
large	-4.0701	24.1990	-7.8117	-12.3172
small	-4.4087	-7.8117	21.7543	-9.5339
squamous	-4.4873	-12.3172	-9.5339	26.3384

Covariance Matrix for the Wilcoxon Statistics				
Cell	**adeno**	**large**	**small**	**squamous**
adeno	121188	-34718	-46639	-39831
large	-34718	151241	-59948	-56576
small	-46639	-59948	175590	-69002
squamous	-39831	-56576	-69002	165410

Test of Equality over Strata			
Test	**Chi-Square**	**DF**	**Pr > Chi-Square**
Log-Rank	25.4037	3	<.0001
Wilcoxon	19.4331	3	0.0002
-2Log(LR)	33.9343	3	<.0001

Results of the log-rank test of the prognostic variables are shown in Output 77.1.10. The univariate test results correspond to testing each prognostic factor marginally. The joint covariance matrix of these univariate test statistics is also displayed. In computing the overall chi-square statistic, the partial chi-square statistics following a forward stepwise entry approach are tabulated.

Consider the log-rank test in Output 77.1.10. Since the univariate test for Kps has the largest chi-square (43.4747) among all the covariates, Kps is entered first. At this stage, the partial chi-square and the chi-square increment for Kps are the same as the univariate chi-square. Among all the covariates not in the model (Age, Prior, DiagTime, Treatment), Treatment has the largest approximate chi-square increment (1.7261) and is entered next. The approximate chi-square for the model that contains Kps and Treatment is 43.4747+1.7261=45.2008 with 2 degrees of freedom. The third covariate entered is Age. The fourth is Prior, and the fifth is DiagTime. The overall chi-square statistic in the last line of the output is the partial chi-square for including all the covariates. It has a value of 46.4200 with 5 degrees of freedom, which is highly significant ($p < 0.0001$).

Output 77.1.10 Log-Rank Test of the Prognostic Factors

Univariate Chi-Squares for the Log-Rank Test					
Variable	**Test Statistic**	**Standard Error**	**Chi-Square**	**Pr > Chi-Square**	**Label**
Age	-40.7383	105.7	0.1485	0.7000	Age in Years
Prior	-19.9435	46.9836	0.1802	0.6712	Prior Treatment?
DiagTime	-115.9	97.8708	1.4013	0.2365	Months till Randomization
Kps	1123.1	170.3	43.4747	<.0001	Karnofsky Index
Treatment	-4.2076	5.0407	0.6967	0.4039	Treatment Indicator

Output 77.1.10 *continued*

Covariance Matrix for the Log-Rank Statistics					
Variable	**Age**	**Prior**	**DiagTime**	**Kps**	**Treatment**
Age	11175.4	-301.2	-892.2	-2948.4	119.3
Prior	-301.2	2207.5	2010.9	78.6	13.9
DiagTime	-892.2	2010.9	9578.7	-2295.3	21.9
Kps	-2948.4	78.6	-2295.3	29015.6	61.9
Treatment	119.3	13.9	21.9	61.9	25.4

Forward Stepwise Sequence of Chi-Squares for the Log-Rank Test						
Variable	**DF**	**Chi-Square**	**Pr > Chi-Square**	**Chi-Square Increment**	**Pr > Increment**	**Label**
Kps	1	43.4747	<.0001	43.4747	<.0001	Karnofsky Index
Treatment	2	45.2008	<.0001	1.7261	0.1889	Treatment Indicator
Age	3	46.3012	<.0001	1.1004	0.2942	Age in Years
Prior	4	46.4134	<.0001	0.1122	0.7377	Prior Treatment?
DiagTime	5	46.4200	<.0001	0.00665	0.9350	Months till Randomization

You can establish this forward stepwise entry of prognostic factors by passing the matrix corresponding to the log-rank test to the RSQUARE method in the REG procedure, as follows. PROC REG finds the sets of variables that yield the largest chi-square statistics.

```
data RSq;
   set Test;
   if _type_='LOG RANK';
   _type_='cov';
run;
proc print data=RSq;
run;
proc reg data=RSq(type=COV);
   model SurvTime=Age Prior DiagTime Kps Treatment
      / selection=rsquare;
   title 'All Possible Subsets of Covariates for the log-rank Test';
run;
```

Output 77.1.11 displays the univariate statistics and their covariance matrix for the log-rank test.

Output 77.1.11 Log-Rank Statistics and Covariance Matrix

Obs	_TYPE_	_NAME_	SurvTime	Age	Prior	DiagTime	Kps	Treatment
1	cov	SurvTime	46.42	-40.74	-19.94	-115.86	1123.14	-4.208
2	cov	Age	-40.74	11175.44	-301.23	-892.24	-2948.45	119.297
3	cov	Prior	-19.94	-301.23	2207.46	2010.85	78.64	13.875
4	cov	DiagTime	-115.86	-892.24	2010.85	9578.69	-2295.32	21.859
5	cov	Kps	1123.14	-2948.45	78.64	-2295.32	29015.62	61.945
6	cov	Treatment	-4.21	119.30	13.87	21.86	61.95	25.409

Results of the best subset regression are shown in Output 77.1.12. The variable Kps generates the largest univariate test statistic among all the covariates, the pair Kps and Age generate the largest test statistic among any other pairs of covariates, and so on. The entry order of covariates is identical to that of PROC LIFETEST.

Output 77.1.12 Best Subset Regression from the REG Procedure

All Possible Subsets of Covariates for the log-rank Test

The REG Procedure
Model: MODEL1
Dependent Variable: SurvTime

R-Square Selection Method

Number in Model	R-Square	Variables in Model
1	0.9366	Kps
1	0.0302	DiagTime
1	0.0150	Treatment
1	0.0039	Prior
1	0.0032	Age
2	0.9737	Kps Treatment
2	0.9472	Age Kps
2	0.9417	Prior Kps
2	0.9382	DiagTime Kps
2	0.0434	DiagTime Treatment
2	0.0353	Age DiagTime
2	0.0304	Prior DiagTime
2	0.0181	Prior Treatment
2	0.0159	Age Treatment
2	0.0075	Age Prior
3	0.9974	Age Kps Treatment
3	0.9774	Prior Kps Treatment
3	0.9747	DiagTime Kps Treatment
3	0.9515	Age Prior Kps
3	0.9481	Age DiagTime Kps
3	0.9418	Prior DiagTime Kps
3	0.0456	Age DiagTime Treatment
3	0.0438	Prior DiagTime Treatment
3	0.0355	Age Prior DiagTime
3	0.0192	Age Prior Treatment
4	0.9999	Age Prior Kps Treatment
4	0.9976	Age DiagTime Kps Treatment
4	0.9774	Prior DiagTime Kps Treatment
4	0.9515	Age Prior DiagTime Kps
4	0.0459	Age Prior DiagTime Treatment
5	1.0000	Age Prior DiagTime Kps Treatment

Example 77.2: Enhanced Survival Plot and Multiple-Comparison Adjustments

(View the complete code for this example (liftex2.sas) in the example repository.)

This example highlights a number of features in the survival plot that uses ODS Graphics. Also shown in this example are comparisons of survival curves based on multiple comparison adjustments. Data of 137 bone marrow transplant patients extracted from Klein and Moeschberger (1997) have been saved in the data set BMT in the Sashelp library. At the time of transplant, each patient is classified into one of three risk categories: ALL (acute lymphoblastic leukemia), AML (acute myelocytic leukemia)-Low Risk, and AML-High Risk. The endpoint of interest is the disease-free survival time, which is the time to death or relapse or to the end of the study in days. In this data set, the variable Group represents the patient's risk category, the variable T represents the disease-free survival time, and the variable Status is the censoring indicator, with the value 1 indicating an event time and the value 0 a censored time.

The following step displays the first 10 observations of the BMT data set in Output 77.2.1. The data set is available in the Sashelp library.

```
proc print data=Sashelp.BMT(obs=10);
run;
```

Output 77.2.1 A Subset of the Bone Marrow Transplant Data

Obs	Group	T	Status
1	ALL	2081	0
2	ALL	1602	0
3	ALL	1496	0
4	ALL	1462	0
5	ALL	1433	0
6	ALL	1377	0
7	ALL	1330	0
8	ALL	996	0
9	ALL	226	0
10	ALL	1199	0

In the following statements, PROC LIFETEST is invoked to compute the product-limit estimate of the survivor function for each risk category. Using ODS Graphics, you can display the number of subjects at risk in the survival plot. The PLOTS= option requests that the survival curves be plotted, and the ATRISK= suboption specifies the time points at which the at-risk numbers are displayed. In the STRATA statement, the ADJUST=SIDAK option requests the Šidák multiple-comparison adjustment, and by default, all paired comparisons are carried out.

```
ods graphics on;

proc lifetest data=sashelp.BMT plots=survival(atrisk=0 to 2500 by 500);
   time T * Status(0);
   strata Group / test=logrank adjust=sidak;
run;
```

Output 77.2.2 displays the estimated disease-free survival for the three leukemia groups with the number of subjects at risk at 0, 500, 1,000, 1,500, 2,000, and 2,500 days. Patients in the AML-Low Risk group

experience a longer disease-free survival than those in the ALL group, who in turn fare better than those in the AML-High Risk group.

Output 77.2.2 Estimated Disease-Free Survival for 137 Bone Marrow Transplant Patients

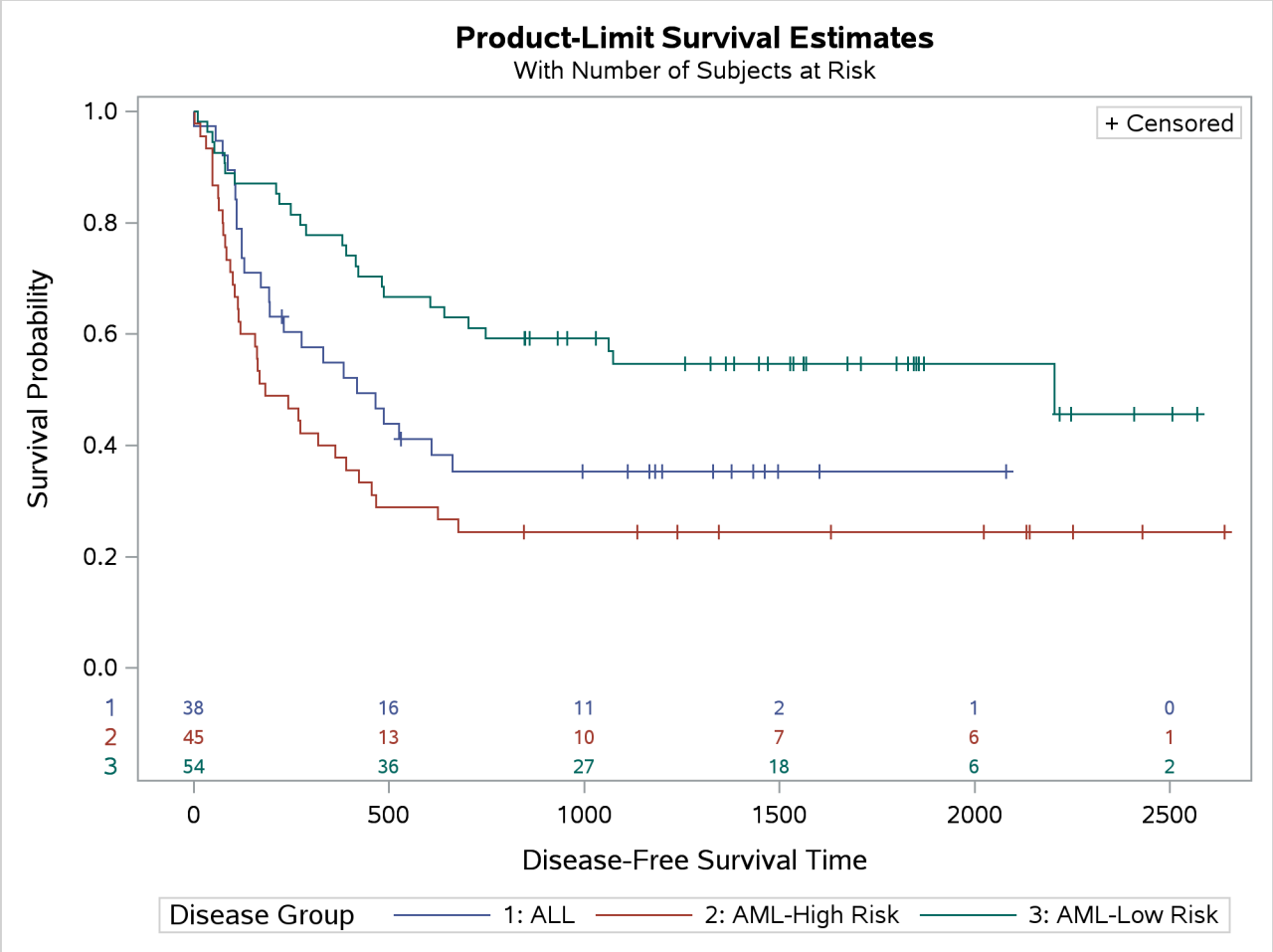

The log-rank test (Output 77.2.3) shows that the disease-free survival times for these three risk groups are significantly different ($p = 0.001$).

Output 77.2.3 Log-Rank Test of Disease Group Homogeneity

Test of Equality over Strata			
Test	**Chi-Square**	**DF**	**Pr > Chi-Square**
Log-Rank	13.8037	2	0.0010

The Šidák multiple-comparison results are shown in Output 77.2.4. There is no significant difference in disease-free survivor functions between the ALL and AML-High Risk groups ($p = 0.2779$). The difference between the ALL and AML-Low Risk groups is marginal ($p = 0.0685$), but the AML-Low Risk and AML-High Risk groups have significantly different disease-free survivor functions ($p = 0.0006$).

Output 77.2.4 All Paired Comparisons

Adjustment for Multiple Comparisons for the Logrank Test				
Strata Comparison			p-Values	
Group	Group	Chi-Square	Raw	Sidak
ALL	AML-High Risk	2.6610	0.1028	0.2779
ALL	AML-Low Risk	5.1400	0.0234	0.0685
AML-High Risk	AML-Low Risk	13.8011	0.0002	0.0006

Suppose you consider the AML-Low Risk group as the reference group. You can use the DIFF= option in the STRATA statement to designate this risk group as the control and apply a multiple-comparison adjustment to the p-values for the paired comparison between the AML-Low Risk group with each of the other groups. Consider the Šidák correction again. You specify the ADJUST= and DIFF= options as in the following statements:

```
proc lifetest data=sashelp.BMT notable plots=none;
   time T * Status(0);
   strata Group / test=logrank adjust=sidak diff=control('AML-Low Risk');
run;
```

Output 77.2.5 shows that although both the ALL and AML-High Risk groups differ from the AML-Low Risk group at the 0.05 level, the difference between the AML-High Risk and the AML-Low Risk group is highly significant ($p = 0.0004$).

Output 77.2.5 Comparisons with the Reference Group

The LIFETEST Procedure

Adjustment for Multiple Comparisons for the Logrank Test				
Strata Comparison			p-Values	
Group	Group	Chi-Square	Raw	Sidak
ALL	AML-Low Risk	5.1400	0.0234	0.0462
AML-High Risk	AML-Low Risk	13.8011	0.0002	0.0004

The survival plot that is displayed in Output 77.2.2 might be sufficient for many purposes, but you might have other preferences. Typical alternatives include displaying the number of subjects at risk outside the plot area, reordering the stratum labels in the survival plot legend, and displaying the strata in the at-risk table by using their full labels. PROC LIFETEST provides options that you can use to make these changes without requiring template changes. In the sashelp.BMT data set, the variable Group that represents the strata is a character variable with three values, namely (in alphabetical order), ALL, AML-High Risk, and AML-Low Risk. It might be desirable to present the strata in the order ALL, AML-Low Risk, and AML-High Risk. The ORDER=INTERNAL option in the STRATA statement enables you to order the strata by their internal values. In the following statements, the new data set Bmt2 is a copy of sashelp.BMT with the variable Group changed to a a numeric variable with values 1, 2, and 3 representing ALL, AML-Low Risk, and AML-High Risk, respectively. The original character values of Group are kept as the formatted values, which are used to label the strata in the printed output.

```
proc format;
   invalue $bmtifmt 'ALL' = 1 'AML-Low Risk' = 2 'AML-High Risk' = 3;
   value bmtfmt 1 = 'ALL'  2 = 'AML-Low Risk' 3 = 'AML-High Risk';
run;

data Bmt2;
   set sashelp.BMT(rename=(Group=G));
   Group = input(input(G, $bmtifmt.), 1.);
   label Group = 'Disease Group';
   format Group bmtfmt.;
   run;
```

The following statements produce a survival plot that has all the aforementioned modifications. The new data set Bmt2 is used as the input data. The OUTSIDE and MAXLEN= options are specified in the PLOTS= option. The OUTSIDE option draws the at-risk table outside the plot area. Because the longest label of the strata has 13 characters, specifying MAXLEN=13 is sufficient to display all the stratum labels in the at-risk table. The ORDER=INTERNAL option in the STRATA statement orders the strata by their numerical values 1, 2, and 3, which represent the order ALL, AML-Low Risk, and AML-High Risk, respectively.

```
proc LIFETEST data=Bmt2 plots=s(atrisk(outside maxlen=13)=0 to 2500 by 500);
   time T*Status(0);
   strata Group / order=internal;
run;
```

The modified survival plot is displayed in Output 77.2.6. The most noticeable change from Output 77.2.2 is that the number of subjects at risk is displayed below the time axis. Other changes include displaying the full labels of the strata in the at-risk table and presenting the strata in the order ALL, AML-Low Risk, and AML-High Risk.

Output 77.2.6 Modified Disease-Free Survival for Bone Marrow Transplant Patients

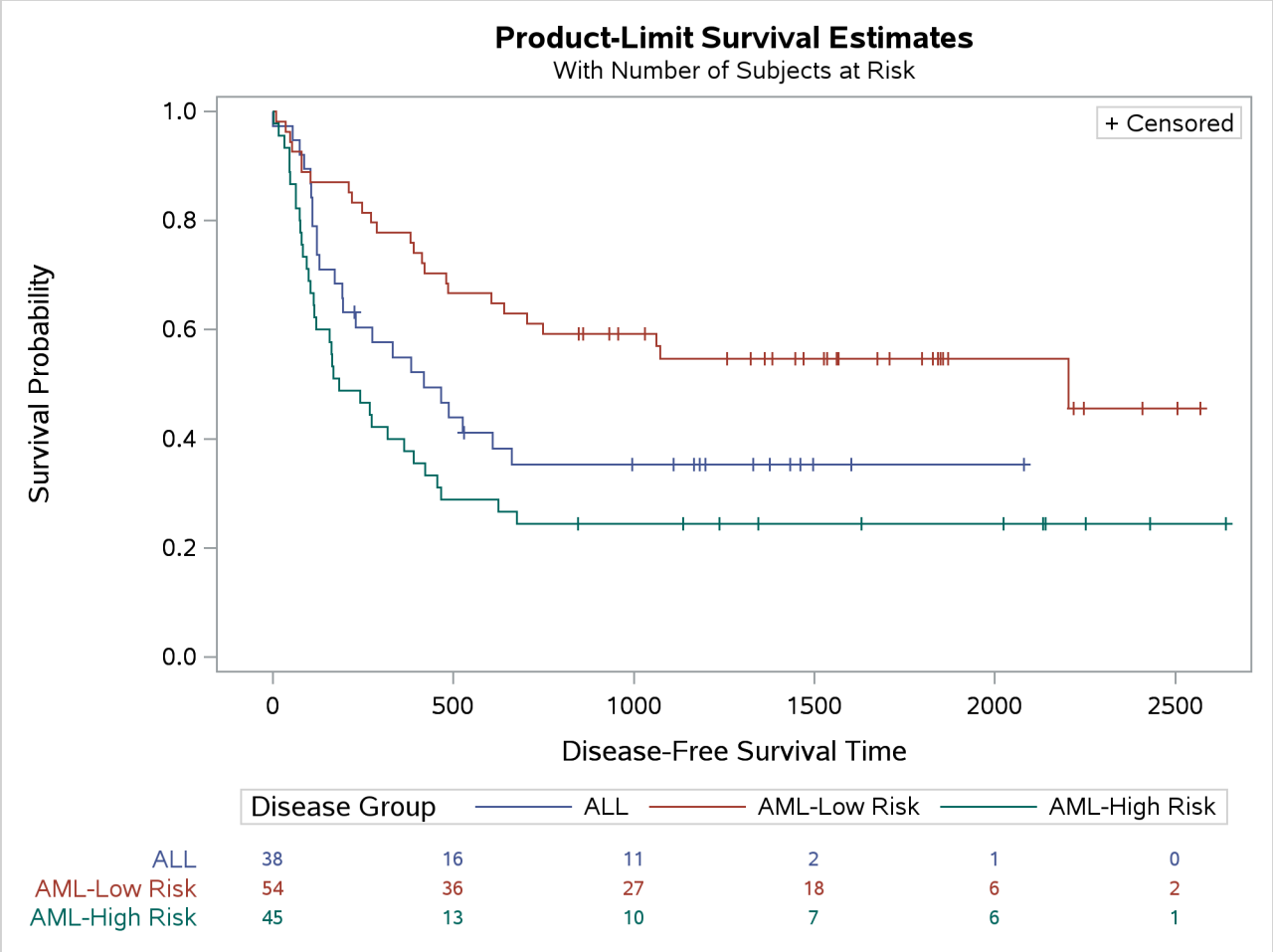

Klein and Moeschberger (1997, Section 4.4) describe in detail how to compute the Hall-Wellner (HW) and equal-precision (EP) confidence bands for the survivor function. You can output these simultaneous confidence intervals to a SAS data set by using the CONFBAND= and OUTSURV= options in the PROC LIFETEST statement. You can display survival curves with pointwise and simultaneous confidence limits through ODS Graphics. When the survival data are stratified, displaying all the survival curves and their confidence limits in the same plot can make the plot appear cluttered. In the following statements, the PLOTS= specification requests that the survivor functions be displayed along with their pointwise confidence limits (CL) and Hall-Wellner confidence bands (CB=HW). The STRATA=PANEL specification requests that the survival curves be displayed in a panel of three plots, one for each risk group.

```
proc lifetest data=Bmt2 plots=survival(cl cb=hw strata=panel);
   time T * Status(0);
   strata Group/order=internal;
run;

ods graphics off;
```

The panel plot is shown in Output 77.2.7.

Output 77.2.7 Estimated Disease-Free Survivor Functions with Confidence Limits

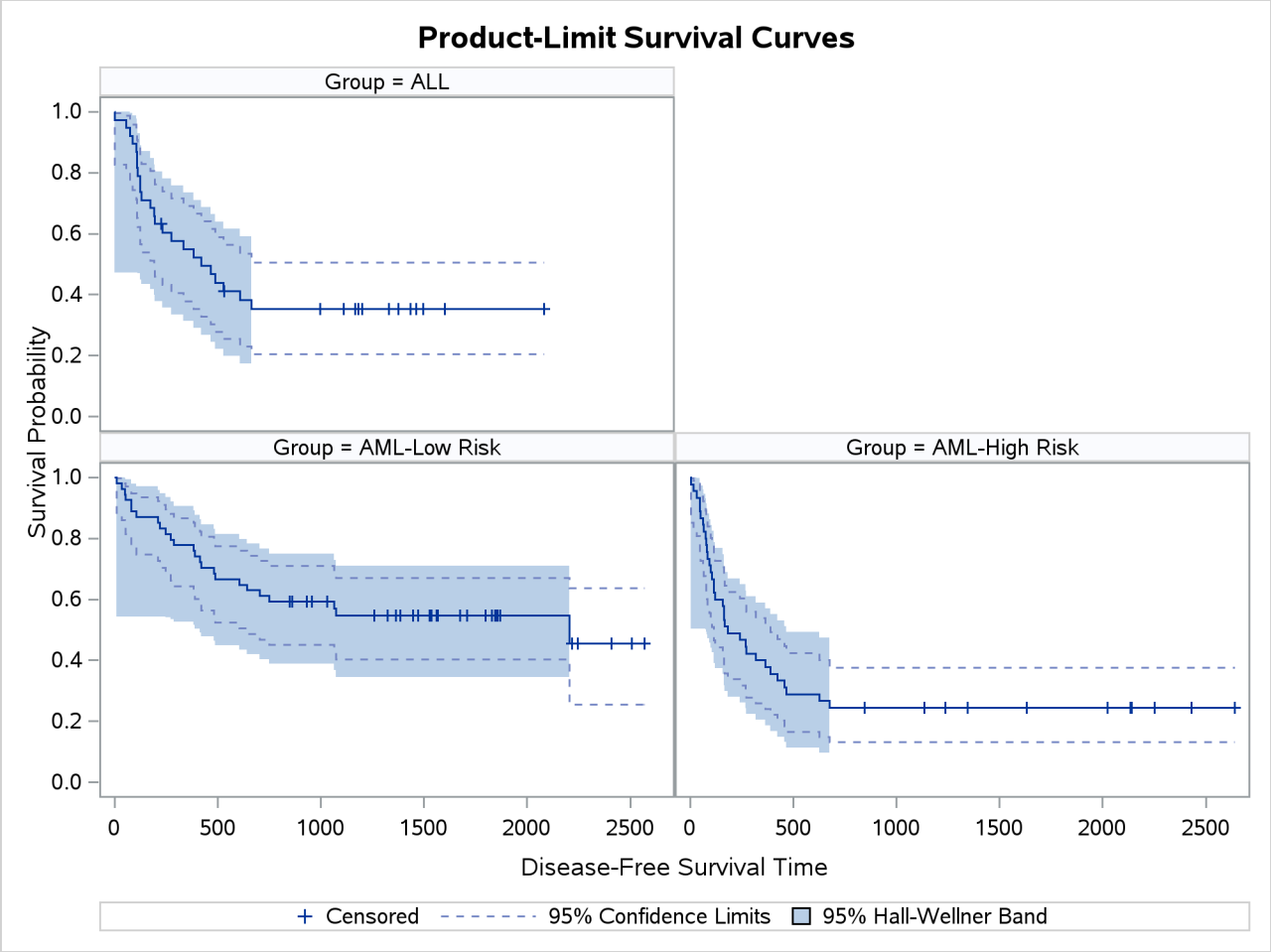

Example 77.3: Life-Table Estimates for Males with Angina Pectoris

(View the complete code for this example (liftex3.sas) in the example repository.)

The data in this example come from Lee (1992, p. 91) and represent the survival rates of males with angina pectoris. Survival time is measured as years from the time of diagnosis. In the following DATA step, the data are read as number of events and number of withdrawals in each one-year time interval for 16 intervals. Three variables are constructed from the data: Years (an artificial time variable with values that are the midpoints of the time intervals), Censored (a censoring indicator variable with the value 1 indicating censored observations and the value 0 indicating event observations), and Freq (the frequency variable). Two observations are created for each interval, one representing the event observations and the other representing the censored observations.

```
title 'Survival of Males with Angina Pectoris';
data Males;
   keep Freq Years Censored;
   retain Years -.5;
   input fail withdraw @@;
```

```
   Years + 1;
   Censored=0;
   Freq=fail;
   output;
   Censored=1;
   Freq=withdraw;
   output;
   datalines;
456   0 226  39 152  22 171  23 135 24 125 107
 83 133  74 102  51  68  42  64  43 45  34  53
 18  33   9  27   6  23   0  30
;
```

In the following statements, the ODS GRAPHICS ON specification enables ODS Graphics. PROC LIFETEST is invoked to compute the various life-table survival estimates, the median residual time, and their standard errors. The life-table method of computing estimates is requested by specifying METHOD=LT. The intervals are specified by the INTERVAL= option. Graphical displays of the life-table survivor function estimate, negative log of the estimate, log of negative log of the estimate, estimated density function, and estimated hazard function are requested by the PLOTS= option. No tests for homogeneity are carried out because the data are not stratified.

```
ods graphics on;
proc lifetest data=Males  method=lt intervals=(0 to 15 by 1)
              plots=(s,ls,lls,h,p);
   time Years*Censored(1);
   freq Freq;
run;
ods graphics off;
```

Results of the life-table estimation are shown in Output 77.3.1. The five-year survival rate is 0.5193 with a standard error of 0.0103. The estimated median residual lifetime, which is 5.33 years initially, reaches a maximum of 6.34 years at the beginning of the second year and decreases gradually to a value lower than the initial 5.33 years at the beginning of the seventh year.

Output 77.3.1 Life-Table Survivor Function Estimate

Survival of Males with Angina Pectoris

The LIFETEST Procedure

Interval		Life Table Survival Estimates									
[Lower,	Upper)	Number Failed	Number Censored	Effective Sample Size	Conditional Probability of Failure	Conditional Probability Standard Error	Survival	Failure	Survival Standard Error	Median Residual Lifetime	Median Standard Error
0	1	456	0	2418.0	0.1886	0.00796	1.0000	0	0	5.3313	0.1749
1	2	226	39	1942.5	0.1163	0.00728	0.8114	0.1886	0.00796	6.2499	0.2001
2	3	152	22	1686.0	0.0902	0.00698	0.7170	0.2830	0.00918	6.3432	0.2361
3	4	171	23	1511.5	0.1131	0.00815	0.6524	0.3476	0.00973	6.2262	0.2361
4	5	135	24	1317.0	0.1025	0.00836	0.5786	0.4214	0.0101	6.2185	0.1853
5	6	125	107	1116.5	0.1120	0.00944	0.5193	0.4807	0.0103	5.9077	0.1806
6	7	83	133	871.5	0.0952	0.00994	0.4611	0.5389	0.0104	5.5962	0.1855
7	8	74	102	671.0	0.1103	0.0121	0.4172	0.5828	0.0105	5.1671	0.2713
8	9	51	68	512.0	0.0996	0.0132	0.3712	0.6288	0.0106	4.9421	0.2763
9	10	42	64	395.0	0.1063	0.0155	0.3342	0.6658	0.0107	4.8258	0.4141
10	11	43	45	298.5	0.1441	0.0203	0.2987	0.7013	0.0109	4.6888	0.4183
11	12	34	53	206.5	0.1646	0.0258	0.2557	0.7443	0.0111	.	.
12	13	18	33	129.5	0.1390	0.0304	0.2136	0.7864	0.0114	.	.
13	14	9	27	81.5	0.1104	0.0347	0.1839	0.8161	0.0118	.	.
14	15	6	23	47.5	0.1263	0.0482	0.1636	0.8364	0.0123	.	.
15	.	0	30	15.0	0	0	0.1429	0.8571	0.0133	.	.

Interval		Evaluated at the Midpoint of the Interval			
[Lower,	Upper)	PDF	PDF Standard Error	Hazard	Hazard Standard Error
0	1	0.1886	0.00796	0.208219	0.009698
1	2	0.0944	0.00598	0.123531	0.008201
2	3	0.0646	0.00507	0.09441	0.007649
3	4	0.0738	0.00543	0.119916	0.009154
4	5	0.0593	0.00495	0.108043	0.009285
5	6	0.0581	0.00503	0.118596	0.010589
6	7	0.0439	0.00469	0.1	0.010963
7	8	0.0460	0.00518	0.116719	0.013545
8	9	0.0370	0.00502	0.10483	0.014659
9	10	0.0355	0.00531	0.112299	0.017301
10	11	0.0430	0.00627	0.155235	0.023602
11	12	0.0421	0.00685	0.17942	0.030646
12	13	0.0297	0.00668	0.149378	0.03511
13	14	0.0203	0.00651	0.116883	0.038894
14	15	0.0207	0.00804	0.134831	0.054919
15	.	.	.	.	.

The breakdown of event and censored observations in the data is shown in Output 77.3.2. Note that 32.8% of

the patients have withdrawn from the study.

Output 77.3.2 Summary of Censored and Event Observations

Summary of the Number of Censored and Uncensored Values			
Total	Failed	Censored	Percent Censored
2418	1625	793	32.80

Note: 2 observations with invalid time, censoring, or frequency values were deleted.

Output 77.3.3 displays the graph of the life-table survivor function estimate. The median survival time, read from the survivor function curve, is 5.33 years, and the 25th and 75th percentiles are 1.04 and 11.13 years, respectively.

Output 77.3.3 Life-Table Survivor Function Estimate

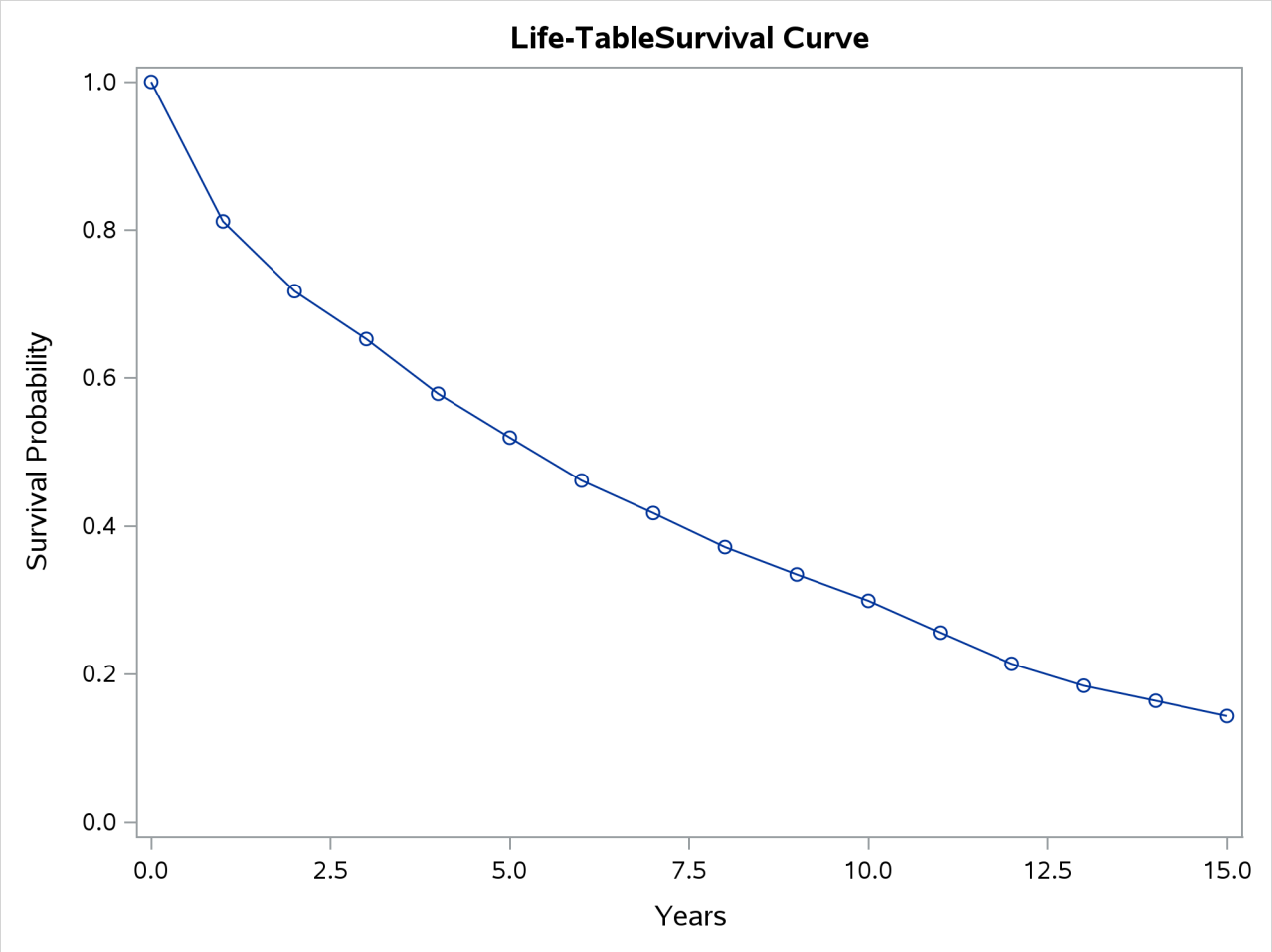

An exponential model might be appropriate for the survival of these male patients with angina pectoris since the curve of the negative log of the survivor function estimate versus the survival time (Output 77.3.4) approximates a straight line through the origin. Note that the graph of the log of the negative log of the survivor function estimate versus the log of time (Output 77.3.5) is practically a straight line.

Output 77.3.4 Negative Log of Survivor Function Estimate

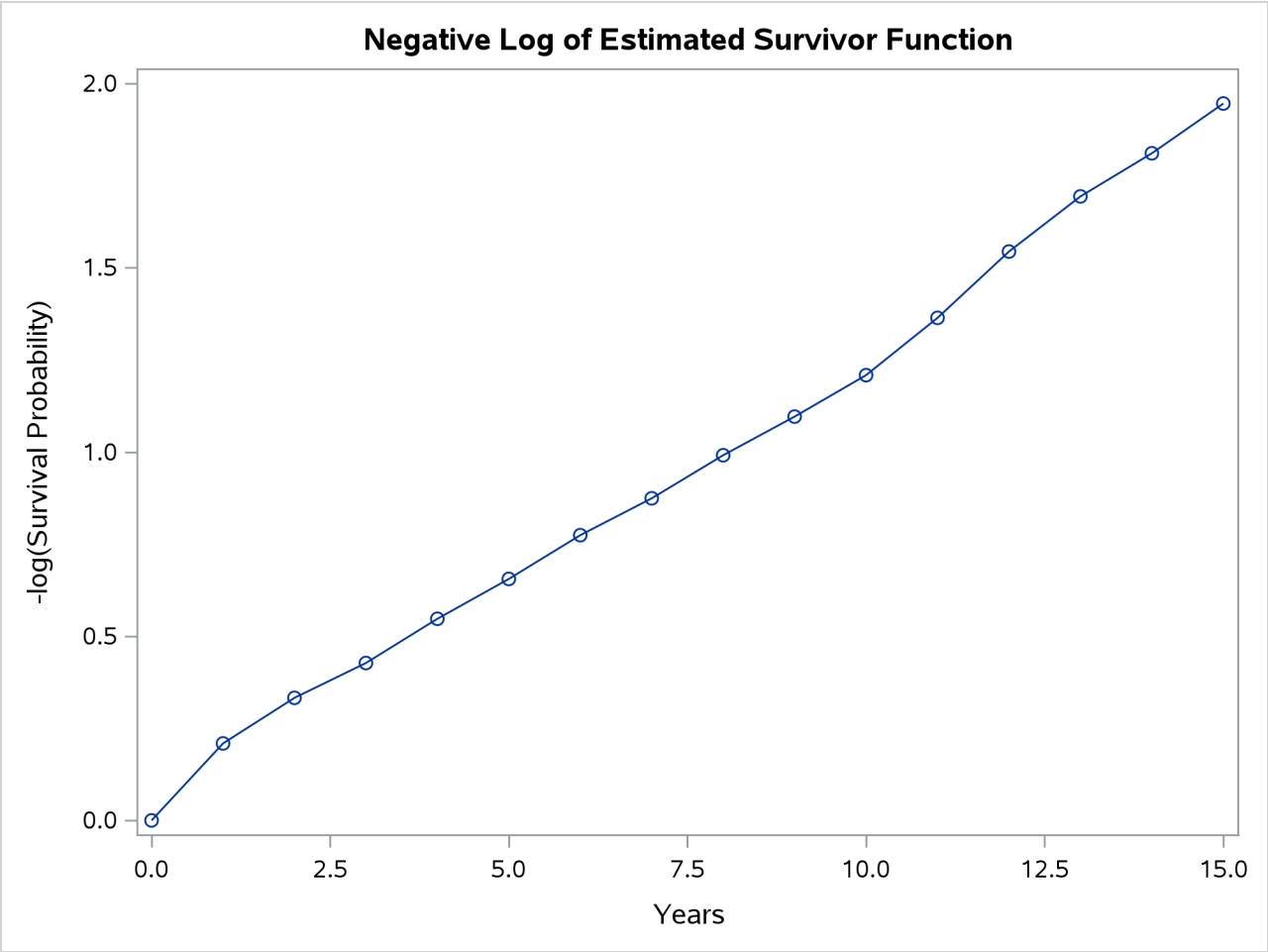

As discussed in Lee (1992), the graph of the estimated hazard function (Output 77.3.6) shows that the death rate is highest in the first year of diagnosis. From the end of the first year to the end of the tenth year, the death rate remains relatively constant, fluctuating between 0.09 and 0.12. The death rate is generally higher after the tenth year. This could indicate that a patient who has survived the first year has a better chance than a patient who has just been diagnosed. The profile of the median residual lifetimes also supports this interpretation.

Output 77.3.5 Log of Negative Log of Survivor Function Estimate

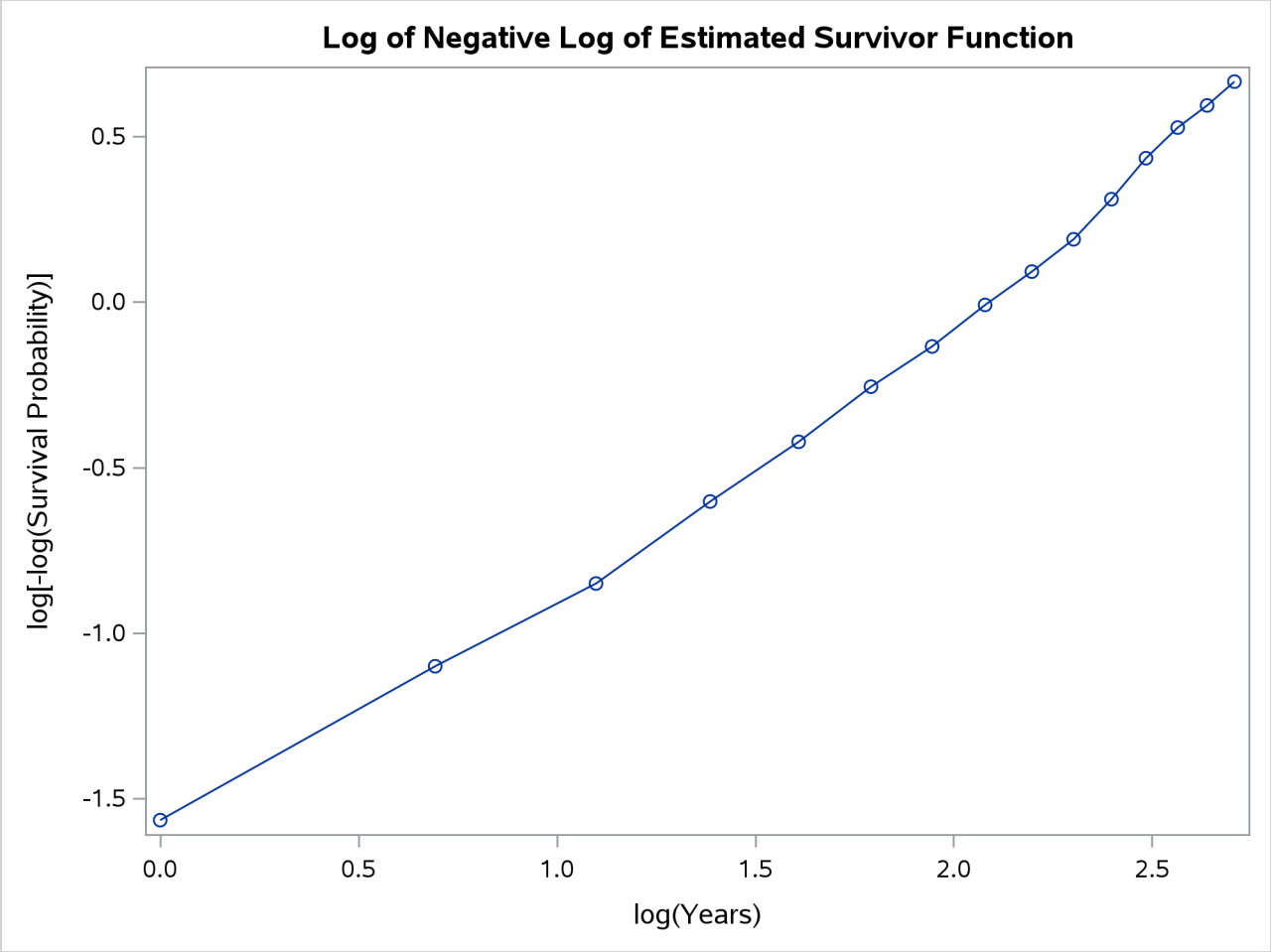

Output 77.3.6 Hazard Function Estimate

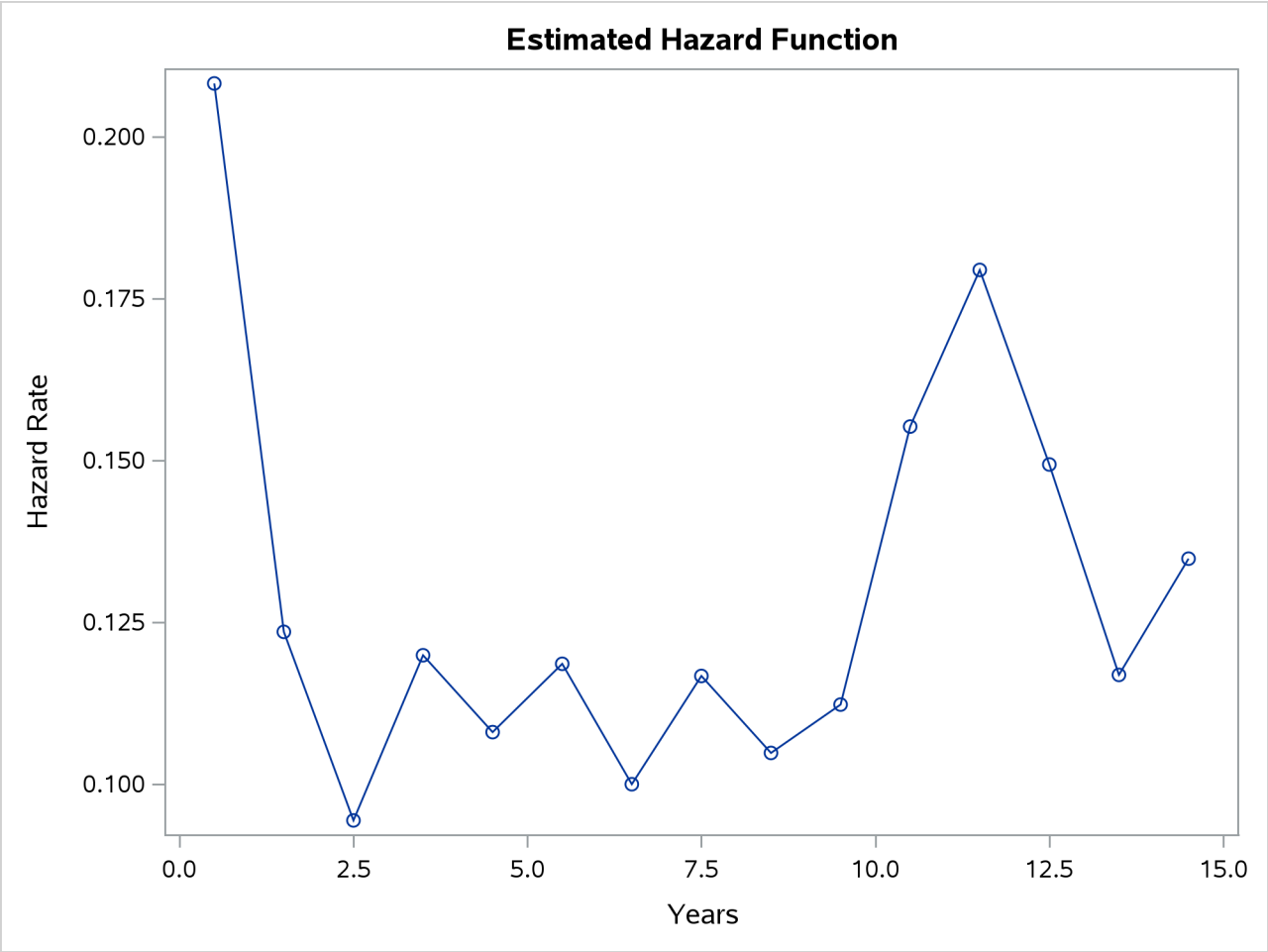

The density estimate is shown in (Output 77.3.7). Visually, it resembles the density function of an exponential distribution.

Output 77.3.7 Density Function Estimate

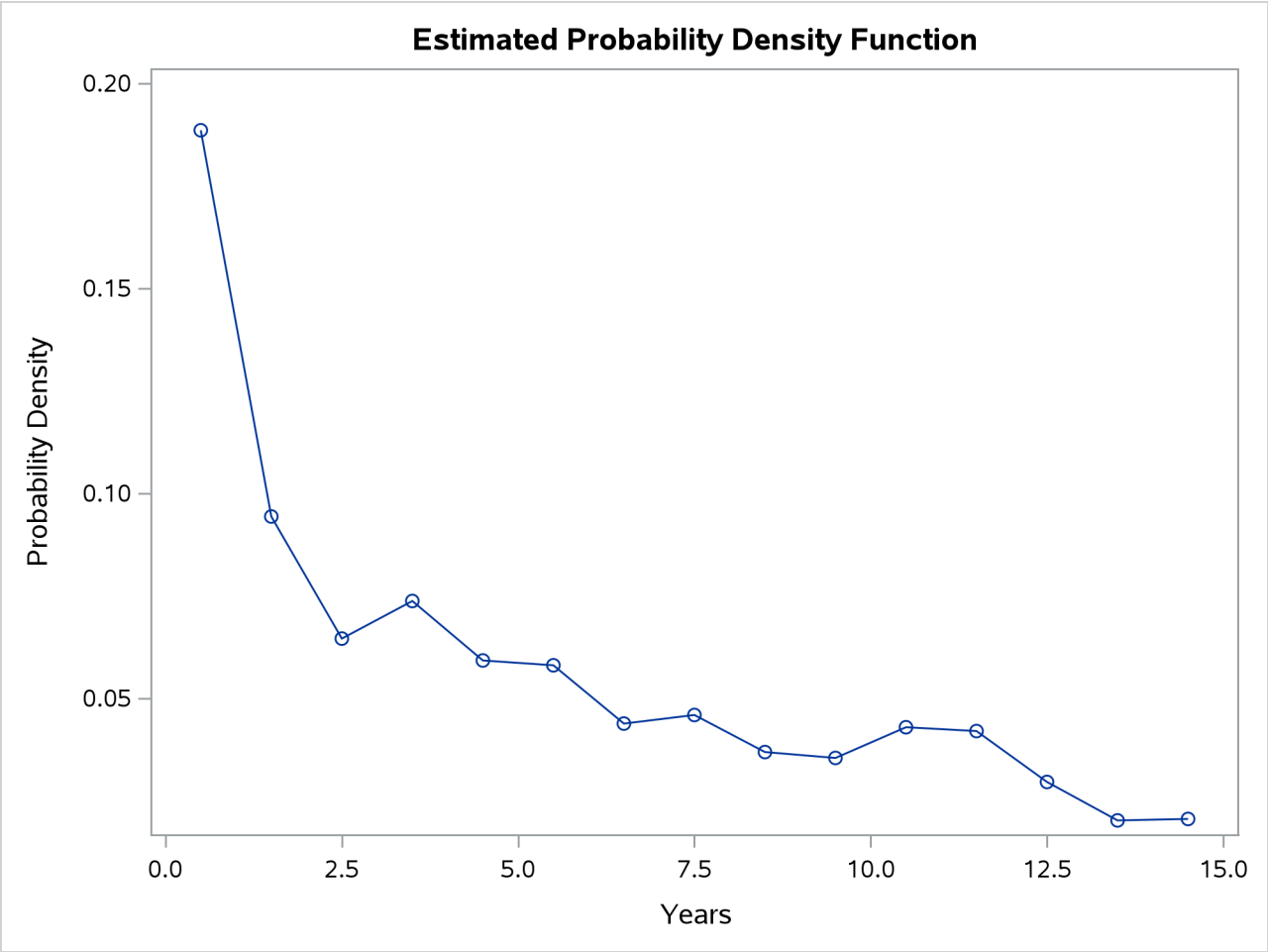

Example 77.4: Nonparametric Analysis of Competing-Risks Data

(View the complete code for this example (liftex4.sas) in the example repository.)

Bone marrow transplant (BMT) is a standard treatment for acute leukemia. Klein and Moeschberger (1997) present a set of BMT data for 137 patients, grouped into three disease categories based on their status at the time of transplantation: acute lymphoblastic leukemia (ALL), acute myelocytic leukemia (AML) low-risk, and AML high-risk. During the follow-up period, some patients might relapse or some patients might die while in remission. Relapse and death in remission are competing events, and the disease-free survival time is the time from transplant to the occurrence of the earlier of these two events.

The following DATA step creates the data set Bmt. (This Bmt data set is not identical to the Sashelp.Bmt data set in Example 77.2, but both are derived from the same study.) The variable Disease denotes the disease group of a patient, which is either ALL, AML-low risk, or AML-high risk. The variable Dftime represents the disease-free survival time, which is the time to relapse, the time to death, or censored. The failure time is expressed in years by dividing the time in days by 365.25. The variable Status has three values: 0 for censored observations, 1 for relapsed patients, and 2 for patients who die before experiencing a relapse. The

variable Gender, which indicates the gender of the BMT patients, is included to illustrate how to conduct a stratified test.

```
proc format;
   value diseaseLabel 1='ALL' 2='AML-Low Risk' 3='AML-High Risk';
   value genderLabel  0='Female' 1='Male';
run;

data Bmt;
   input Disease Dftime Status Gender@@;
   Dftime= Dftime / 365.25;
   label Dftime='Disease-Free Survival Time (Years)'
         Disease='Disease Group';
   datalines;
1       2081       0       1       1       1602    0       1
1       1496       0       1       1       1462    0       0
1       1433       0       1       1       1377    0       1
1       1330       0       1       1       996     0       1

   ... more lines ...

3       625        1       0       3       48      1       0
3       273        1       1       3       63      2       1
3       76         1       1       3       113     1       0
3       363        2       1
;
```

For competing-risks data, PROC LIFETEST estimates the cumulative incidence function (CIF). If you have multiple samples of data, it estimates the CIF for each sample and compares the CIFs between samples by using Gray's test (Gray 1988). The estimated CIF is a step function with a jump at each distinct time when the event of interest occurred. If there are a large number of such event times, the table of the estimated CIF could be quite lengthy. If you are interested in the cumulative incidence at specific time points, you can use the TIMELIST= option in the PROC LIFETEST statement to specify these time points, and PROC LIFETEST prints the CIF estimates only at these time points.

Consider relapse as the event of interest. The following statements use PROC LIFETEST to estimate the CIF for relapse. To designate relapse (Status=1) as the event of interest, you specify the option EVENTCODE=1 in the TIME statement. The TIMELIST= option in the PROC LIFETEST statement specifies the time points to display the CIF estimate, at half a year, one year, one and a half years, two years, four years, and six years. The STRATA statement identifies the disease groups as different samples of data. The PLOTS= option requests a plot of the estimated CIF, with a inset that shows the p-value of Gray's test.

```
ods graphics on;
proc lifetest data=Bmt plots=cif(test) timelist=0.5 1.0 1.5 2.0 4.0 6.0;
   time Dftime*Status(0)/eventcode=1;
   strata Disease / order=internal;
   format Disease diseaseLabel. Gender genderLabel.;
run;
```

Output 77.4.1 tabulates the number of patients in each disease group who experience the event of interest (relapse) and those who experience the competing event (death in remission).

Output 77.4.1 Distribution of Events and Censored Observations

The LIFETEST Procedure

Failed Event: Status=1

Summary of Failure Outcomes					
Stratum	Disease	Failed Events	Competing Events	Censored	Total
1	ALL	12	12	14	38
2	AML-Low Risk	9	16	29	54
3	AML-High Risk	21	13	11	45
Total		42	41	54	137

Output 77.4.2 displays the CIF estimate of relapse for the ALL patients at the selected time points. The predicted CIF at half a year after transplant is 0.1842, with a 95% confidence interval of (0.0798, 0.3224). At two years after transplant, the estimated CIF is 0.3243, with a 95% confidence interval of (0.1778, 0.4787). It is not feasible to estimate the cumulative incidence at a time beyond the largest observed time, which is 5.6975 years in the ALL group. That is why the estimates are missing at six years.

Output 77.4.2 Estimated CIF for ALL Patients

Cumulative Incidence Function Estimates Stratum 1: Disease Group = ALL					
Timelist	Dftime	Cumulative Incidence	Standard Error	95% Confidence Interval	
0.5	0.353183	0.1842	0.0639	0.0798	0.3224
1	0.629706	0.2380	0.0705	0.1164	0.3836
1.5	1.048597	0.2654	0.0733	0.1360	0.4140
2	1.812457	0.3243	0.0791	0.1788	0.4787
4	1.812457	0.3243	0.0791	0.1788	0.4787
6	.	.	.	.	.

Output 77.4.3 and Output 77.4.4 display the CIF estimates at the selected times for AML-low risk and AML-high risk patients, respectively.

Output 77.4.3 Estimated CIF for AML-Low Risk Patients

Cumulative Incidence Function Estimates Stratum 2: Disease Group = AML-Low Risk					
Timelist	Dftime	Cumulative Incidence	Standard Error	95% Confidence Interval	
0.5	0	0	0	.	.
1	0.744695	0.0741	0.0360	0.0234	0.1646
1.5	1.330595	0.1296	0.0463	0.0563	0.2344
2	1.659138	0.1481	0.0489	0.0685	0.2565
4	2.047912	0.1667	0.0514	0.0813	0.2783
6	2.047912	0.1667	0.0514	0.0813	0.2783

Output 77.4.4 Estimated CIF for AML-High Risk Patients

Cumulative Incidence Function Estimates					
Stratum 3: Disease Group = AML-High Risk					
Timelist	Dftime	Cumulative Incidence	Standard Error	95% Confidence Interval	
0.5	**0.429843**	0.2889	0.0686	0.1642	0.4259
1	**0.747433**	0.3556	0.0726	0.2181	0.4955
1.5	**1.278576**	0.4444	0.0757	0.2940	0.5844
2	**1.711157**	0.4667	0.0761	0.3137	0.6059
4	**1.711157**	0.4667	0.0761	0.3137	0.6059
6	**1.711157**	0.4667	0.0761	0.3137	0.6059

Output 77.4.5 displays the homogeneity test of Gray (1988), which indicates strong evidence of a significant difference in the CIF for relapse among the three disease groups ($p = 0.0028$).

Output 77.4.5 Homogeneity Test of CIFs for Relapse

Gray's Test for Equality of Cumulative Incidence Functions		
Chi-Square	DF	Pr > Chi-Square
11.9229	2	0.0026

The PLOTS= option produces a plot of the estimated CIFs (Output 77.4.5). Note that the range of each curve is from 0 to the largest observed time of the corresponding disease group, which is 5.6975 years for ALL patients, 7.0335 years for AML-low risk patients, and 7.2279 years for AML-high risk patients. With PLOTS=CIF(TEST) specified, that plot displays the p-value of the homogeneity test for the disease groups. The cumulative incidences of relapse are smallest for the AML-low risk patients and highest for the AML-high risk patients, with the ALL patients in between.

Output 77.4.6 CIF Estimates of Relapse in Bone Marrow Transplant Study

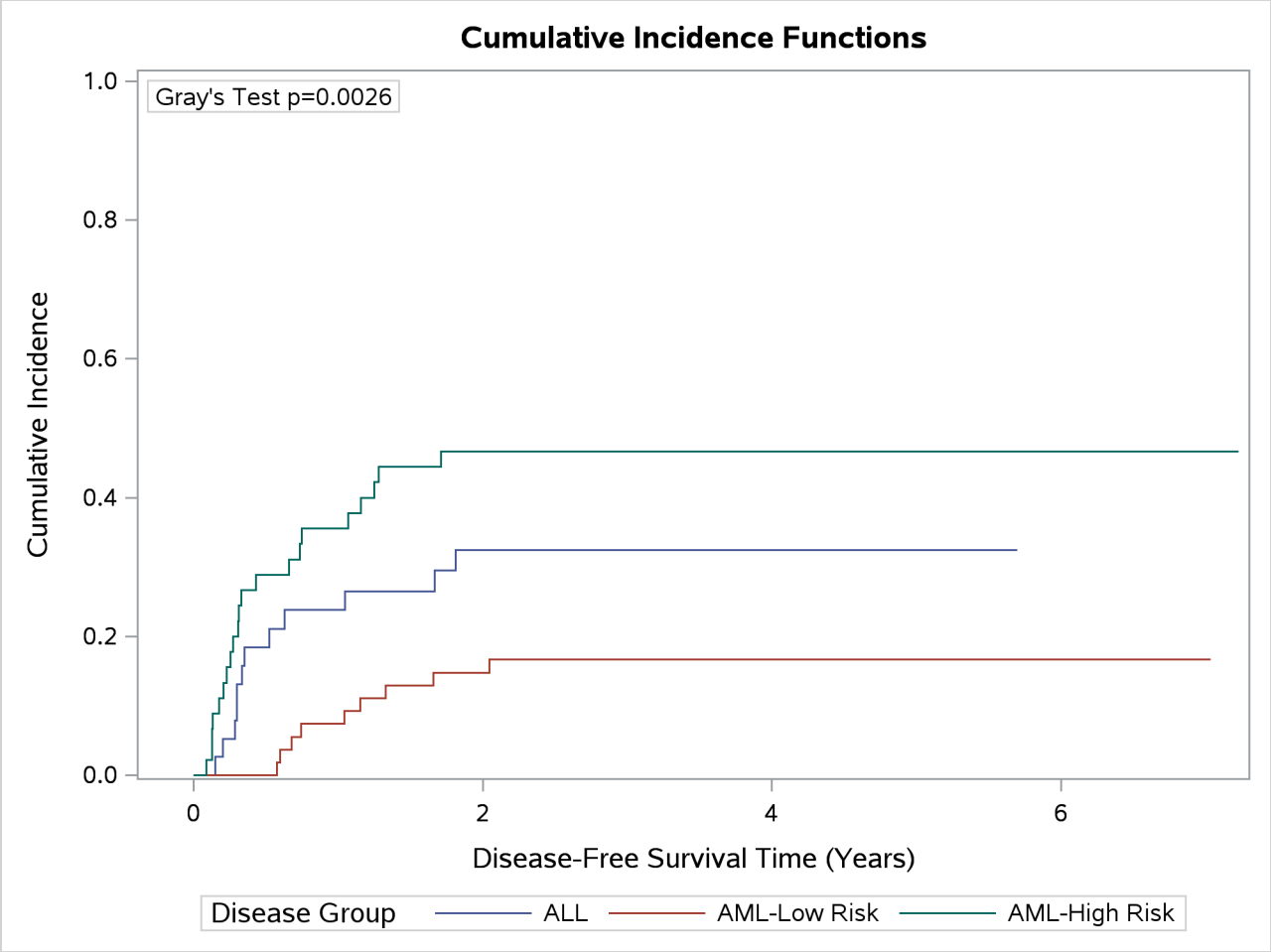

When you specify the GROUP= option in the STRATA statement, PROC LIFETEST enables you to perform a stratified test to evaluate the homogeneity of the CIFs between groups. Consider Gender as the stratifying variable for the stratified test. You specify Gender in the STRATA statement with the GROUP=DISEASE option as follows:

```
proc lifetest data=bmt plots=cif(test );
   time Dftime*Status(0)/eventcode=1;
   strata Gender/group=Disease order=internal;
   format Disease diseaseLabel. Gender genderLabel.;
run;
ods graphics off;
```

PROC LIFETEST summarizes the number of events and censored observations in each disease group by gender (Output 77.4.7). PROC LIFETEST computes a separate CIF estimate for each disease category for the female patients (Output 77.4.8) and likewise for the male patients (not shown here).

Output 77.4.7 Distribution of Events and Censored Observations

The LIFETEST Procedure

Failed Event: Status=1

Summary of Failure Outcomes

Stratum	Gender	Disease	Failed Events	Competing Events	Censored	Total
1	Female	ALL	5	3	4	12
1	Female	AML-Low Risk	3	7	14	24
1	Female	AML-High Risk	12	6	3	21
Subtotal			20	16	21	57
2	Male	ALL	7	9	10	26
2	Male	AML-Low Risk	6	9	15	30
2	Male	AML-High Risk	9	7	8	24
Subtotal			22	25	33	80
Total			42	41	54	137

Output 77.4.8 CIF Estimates for Female Patients

Cumulative Incidence Function Estimates

Stratum 1: Gender = Female

Disease	Dftime	Cumulative Incidence	Standard Error	95% Confidence Interval	
ALL	**0**	0	0	.	.
ALL	**0.150582**	0.0833	0.0833	0.00422	0.3233
ALL	**0.301164**	0.1667	0.1126	0.0235	0.4250
ALL	**0.334018**	0.2500	0.1312	0.0544	0.5168
ALL	**0.353183**	0.3333	0.1433	0.0938	0.6004
ALL	**0.629706**	0.4333	0.1591	0.1384	0.7022
AML-Low Risk	**0**	0	0	.	.
AML-Low Risk	**0.744695**	0.0417	0.0419	0.00271	0.1810
AML-Low Risk	**1.043121**	0.0833	0.0580	0.0135	0.2381
AML-Low Risk	**1.330595**	0.1250	0.0695	0.0299	0.2918
AML-High Risk	**0**	0	0	.	.
AML-High Risk	**0.131417**	0.0476	0.0477	0.00303	0.2023
AML-High Risk	**0.175222**	0.0952	0.0658	0.0153	0.2665
AML-High Risk	**0.229979**	0.1429	0.0785	0.0339	0.3267
AML-High Risk	**0.25462**	0.1905	0.0882	0.0569	0.3832
AML-High Risk	**0.309377**	0.2381	0.0958	0.0832	0.4368
AML-High Risk	**0.314853**	0.2857	0.1018	0.1122	0.4879
AML-High Risk	**0.328542**	0.3333	0.1064	0.1435	0.5370
AML-High Risk	**0.429843**	0.3810	0.1098	0.1768	0.5841
AML-High Risk	**0.733744**	0.4286	0.1125	0.2113	0.6302
AML-High Risk	**1.155373**	0.4762	0.1144	0.2467	0.6748
AML-High Risk	**1.278576**	0.5238	0.1154	0.2834	0.7178
AML-High Risk	**1.711157**	0.5714	0.1155	0.3212	0.7590

Output 77.4.9 shows the results of the stratified test with a p-value of 0.0026, which is essentially the same as the p-value of the nonstratified test. The PLOTS= option creates a panel plot with two cells: one cell for female patients and the other cell for male patients. Each cell contains three CIF curves, one for each disease group (Output 77.4.10). Regardless of the gender of the patient, an AML-high risk patient is more likely to relapse than an ALL patient, and an ALL patient is more likely to relapse than an AML-low risk patient. This ordering of probabilities is revealed in the panel plots in Output 77.4.10.

Output 77.4.9 Stratified Gray's Test

Gray's Test for Equality of Cumulative Incidence Functions		
Chi-Square	DF	Pr > Chi-Square
11.7625	2	0.0028

Output 77.4.10 Panel Plots of CIFs for Relapse

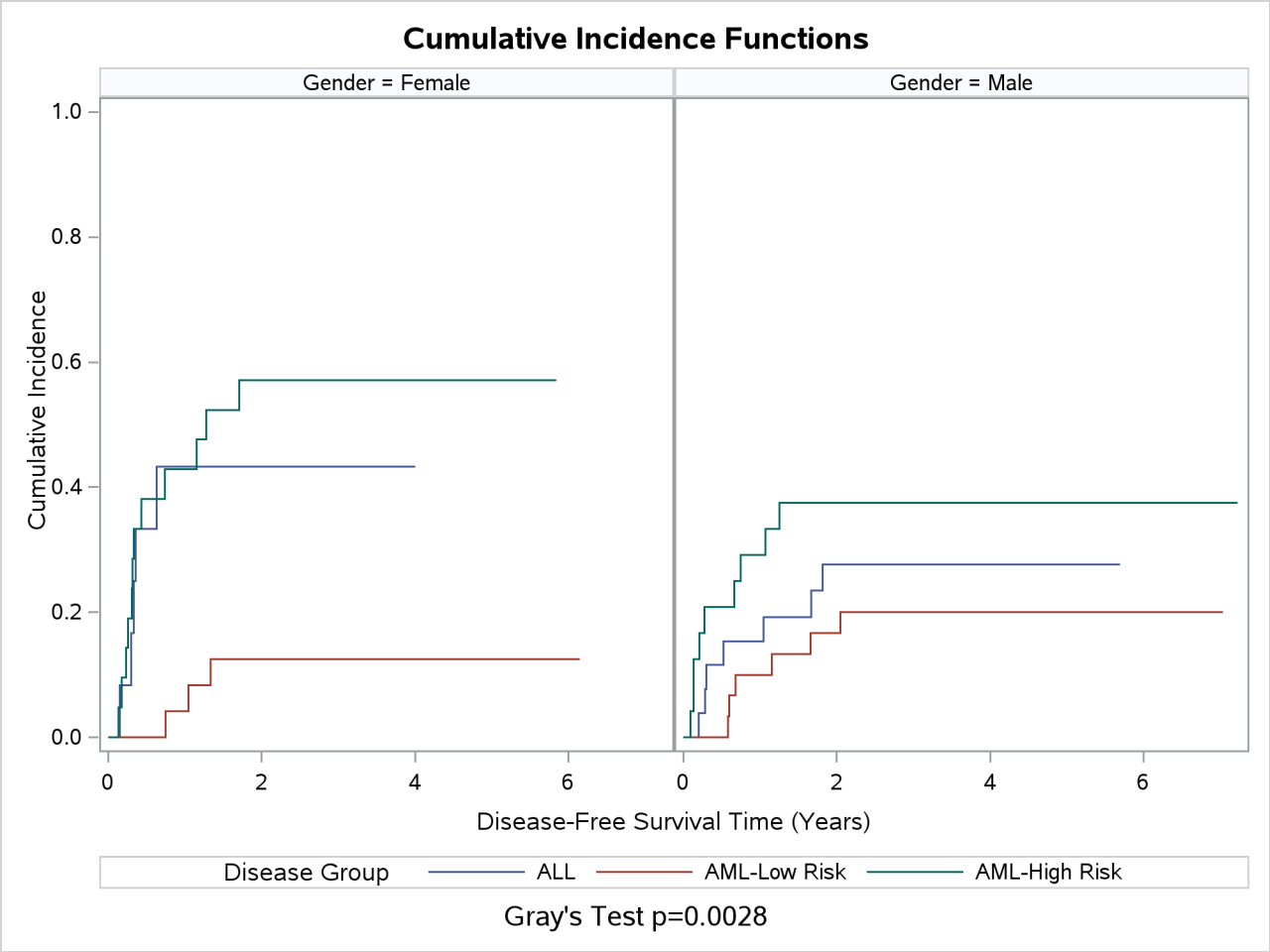

Example 77.5: Restricted Mean Analysis

(View the complete code for this example (liftex5.sas) in the example repository.)

This example illustrates how to perform nonparametric analyses with respect to the restricted mean survival time (RMST) and the restricted mean time lost (RMTL). Consider the VALung data set in Example 77.1. The failure time variable is SurvTime. The censoring indicator variable is Censor, which has a value of 1 for a censored observation and a value of 0 for an uncensored observation. Each patient has one of the four types of cancer cells (adeno, large, small, and squamous) that are identified by the variable Cell.

The following statements use PROC LIFETEST to perform analyses of the restricted mean survival time (RMST) and restricted mean time lost (RMTL) in addition to the standard analyses:

```
ods graphics on;
proc lifetest data=VALung plots=(rmst rmtl) rmst rmtl(tau=90) maxtime=600;
   time SurvTime*Censor(1);
   strata Cell;
run;
ods graphics off;
```

The RMST and RMTL options estimate the restricted mean survival time and the restricted mean time lost, respectively. The variable Cell is specified in the STRATA statement to compute the RMST for each type of cancer cell. ODS Graphics must be enabled for graphs to be produced. Graphical displays of the RMST and RMTL curves are requested through the PLOTS= option in the PROC LIFETEST statement. Because of a few large survival times, a MAXTIME= option value of 600 is used to set the upper limit of the time axis; that is, the time horizon extends from 0 to a maximum of 600 days in the plots.

Output 77.5.1 displays the τ value that the RMST analysis uses. If you omit the TAU= option, PROC LIFETEST uses the smallest value among the largest observed times across the strata as the τ value.

Output 77.5.1 RMST Analysis Information

The LIFETEST Procedure

RMST Analysis Information	
Tau	186

Output 77.5.2 displays the RMST estimates for the four cell types.

Output 77.5.2 RMST Estimates

RMST Estimates			
Stratum	Cell Type	Estimate	Standard Error
1	adeno	65.55556	9.9303
2	large	128.0370	11.9858
3	small	64.20647	8.2859
4	squamous	113.8040	12.3703

Output 77.5.3 displays information for the RMTL analysis. A τ value of 90 is shown; this is the value

specified in the TAU= option.

Output 77.5.3 RMTL Analysis Information

RMTL Analysis Information	
Tau	90

Output 77.5.4 displays the RMTL estimates for the four cell types at $\tau = 90$.

Output 77.5.4 RMTL Estimates at $\tau = 90$

RMTL Estimates			
Stratum	**Cell Type**	**Estimate**	**Standard Error**
1	adeno	36.54321	6.3206
2	large	14.33333	4.9560
3	small	41.27083	4.6763
4	squamous	22.99365	5.6430

The graph of the estimated RMST curves is shown in Output 77.5.5. These curves exhibit a behavior similar to that of the survival curves in Example 77.1: the adeno cell curve and the small cell curve are much closer to each other than they are to the large cell curve or the squamous cell curve. The shapes of the large cell curve and the squamous cell curve are quite different, although both increase more rapidly than those of the adeno and small cells. The squamous cell curve initially increases less rapidly than the large cell curve, but the role is reversed in the later period.

Output 77.5.5 RMST Curves

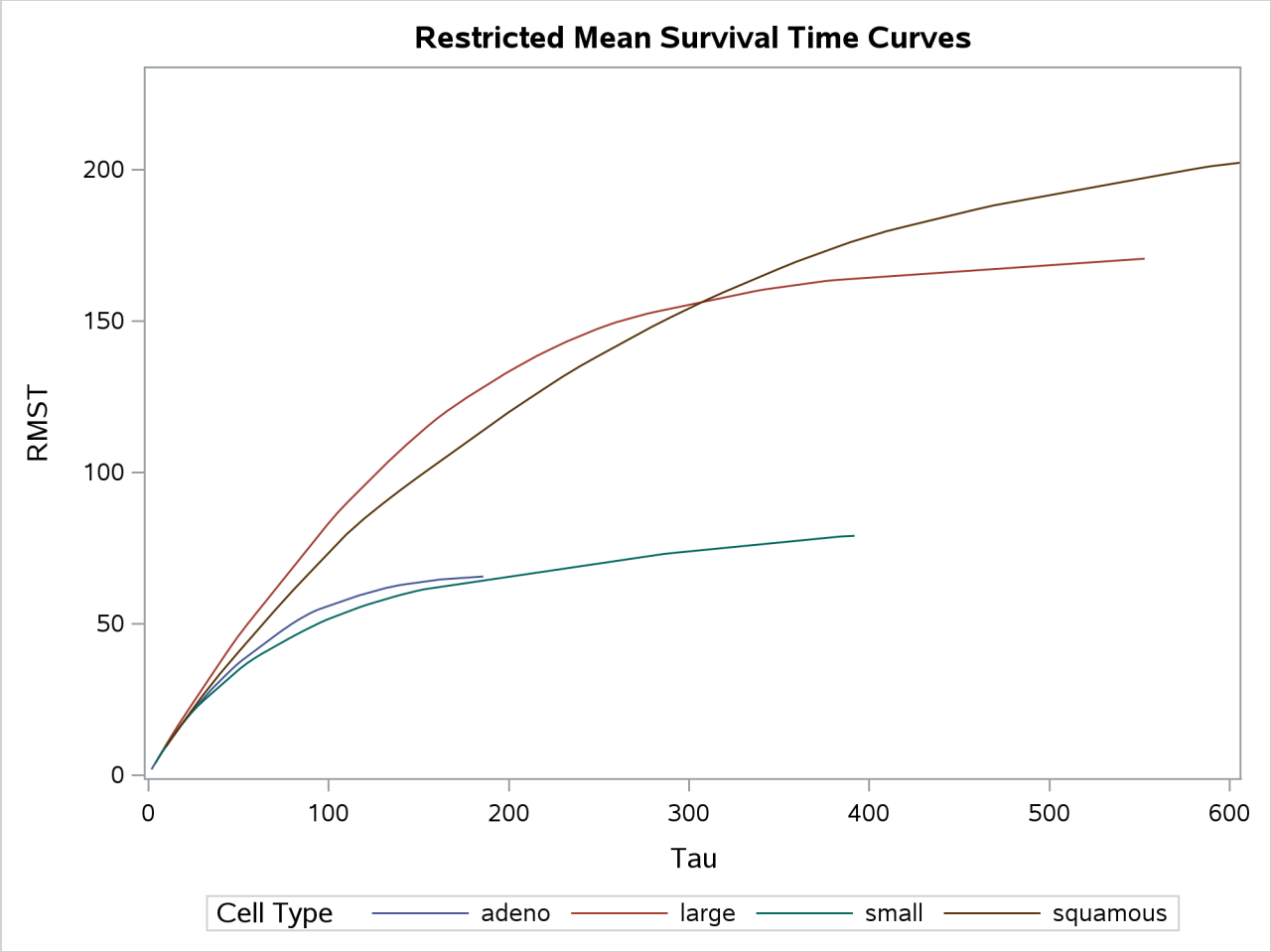

The graph of the estimated RMTL curves is displayed in Output 77.5.6. Again, the adeno cell curve and the small cell curve are much closer to each other and farther away from the large cell and squamous cell curves.

Output 77.5.6 RMTL Curves

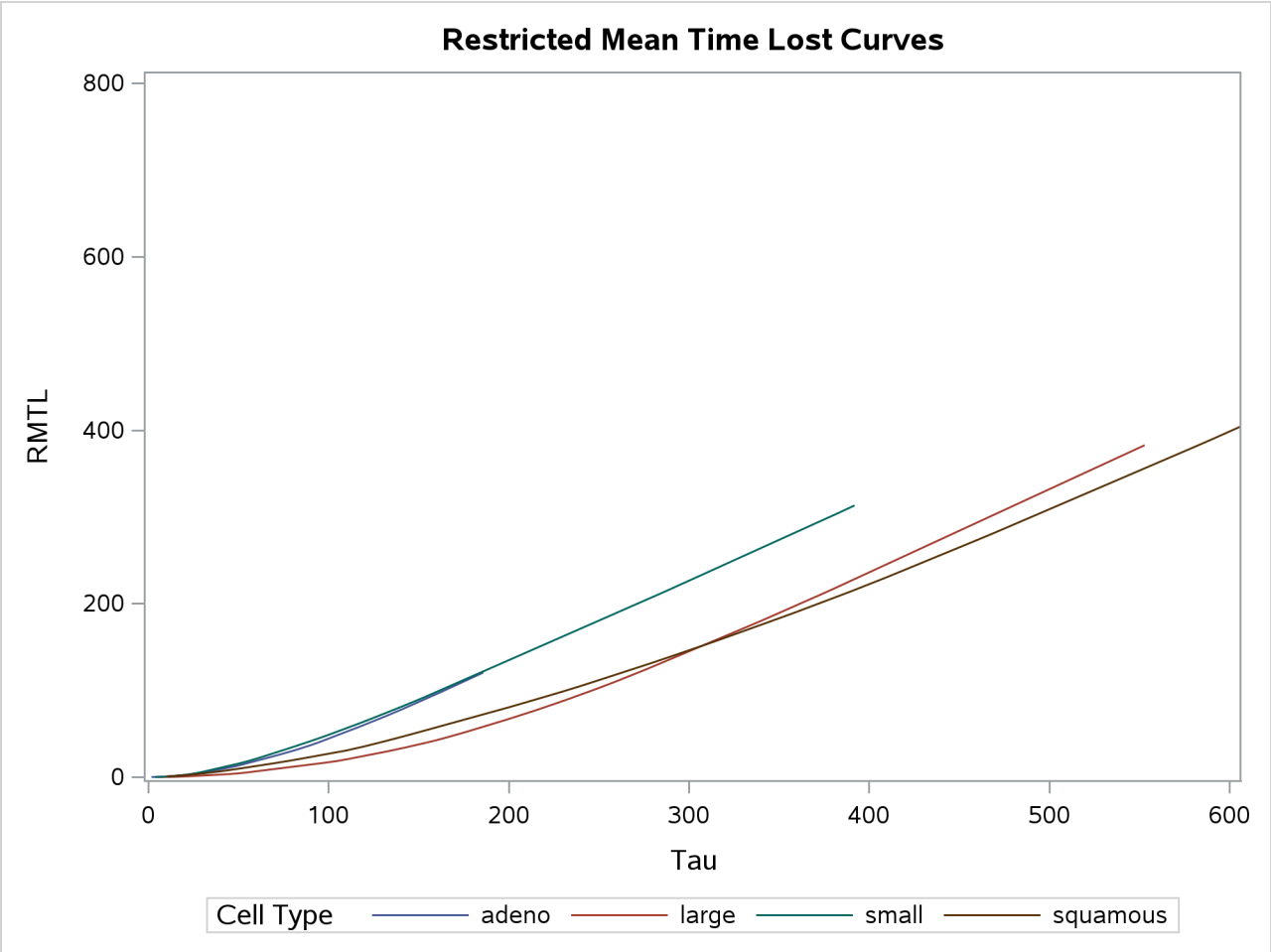

Results of the homogeneity test for the RMST across cell types are given in Output 77.5.7. The table displays the approximate chi-square statistic, degrees of freedom, and p-value. The test results indicate strong evidence that the RMSTs for the four types of cancer cells are not the same ($p < 0.0001$).

Output 77.5.7 Homogeneity Tests across Cell Types

RMST Test of Equality			
Source	**Chi-Square**	**DF**	**Pr > ChiSq**
Strata	28.4427	3	<.0001

The homogeneity test does not identify which pairs of the RMSTs are different. In the following statements, you use the DIFF= option to compute the paired differences of the RMST among the groups. To protect yourself from falsely significant results, you use the ADJUST= option to make multiple-comparison adjustments to the resulting p-values.

```
proc lifetest data=VALung rmst;
   time SurvTime*Censor(1);
   strata Cell / diff=all adj=sidak;
run;
```

The Šidák multiple-comparison results are shown in Output 77.5.8.

Output 77.5.8 All Paired Comparisons

The LIFETEST Procedure

Restricted Mean Survival Time Comparisons
Adjustment for Multiple Comparisons: Sidak

Stratum Comparison		Difference	Standard Error	Chi-Square	Pr > ChiSq Unadjusted	Pr > ChiSq Adjusted
adeno	large	-62.4815	15.5650	16.1141	<.0001	0.0004
adeno	small	1.349087	12.9332	0.0109	0.9169	1.0000
adeno	squamous	-48.2485	15.8630	9.2511	0.0024	0.0140
large	small	63.83057	14.5710	19.1901	<.0001	<.0001
large	squamous	14.23303	17.2245	0.6828	0.4086	0.9572
small	squamous	-49.5975	14.8889	11.0967	0.0009	0.0052

The results suggest that you can divide the four risk groups into two classes. The first class consists of the small and adeno cell types, and there is no significant difference in the RMST between them ($p = 1.0000$). The second class consists of the large and squamous cell types, and the paired comparison is not significant ($p = 0.9572$). However, there is significant difference in any paired comparison between the two classes.

Suppose you consider the small cell type to be the reference group. You can use the DIFF= option in the STRATA statement to designate this risk group as the control and apply a multiple-comparison adjustment to the p-values for the paired comparison between the small cell type and the other cell types. Consider the Šidák correction again. You specify the ADJUST= and DIFF= options as in the following statements:

```
proc lifetest data=VALung rmst;
   time SurvTime*Censor(1);
   strata Cell / adj=sidak diff=control('small');
run;
```

Output 77.5.9 shows that both the large and squamous cell types differ from the small cell type at the 0.05 level, whereas the difference between the adeno and small cell types is not significant ($p = 0.9994$).

Output 77.5.9 Comparisons to the Reference Group

The LIFETEST Procedure

Restricted Mean Survival Time Comparisons
Adjustment for Multiple Comparisons: Sidak

Stratum Comparison		Difference	Standard Error	Chi-Square	Pr > ChiSq Unadjusted	Pr > ChiSq Adjusted
adeno	small	1.349087	12.9332	0.0109	0.9169	0.9994
large	small	63.83057	14.5710	19.1901	<.0001	<.0001
squamous	small	49.59754	14.8889	11.0967	0.0009	0.0026

References

Aalen, O. (1978). "Nonparametric Estimation of Partial Transition Probabilities in Multiple Decrement Models." *Annals of Statistics* 6:534–545.

Andersen, P. K., Borgan, Ø., Gill, R. D., and Keiding, N. (1992). *Statistical Models Based on Counting Processes*. New York: Springer-Verlag.

Borgan, Ø., and Liestøl, K. (1990). "A Note on Confidence Interval and Bands for the Survival Curves Based on Transformations." *Scandinavian Journal of Statistics* 18:35–41.

Brookmeyer, R., and Crowley, J. (1982). "A Confidence Interval for the Median Survival Time." *Biometrics* 38:29–41.

Chung, C. F. (1986). *Formulae for Probabilities Associated with Wiener and Brownian Bridge Processes*. Technical Report 79, Laboratory for Research in Statistics and Probability, Carleton University, Ottawa.

Collett, D. (1994). *Modelling Survival Data in Medical Research*. London: Chapman & Hall.

Cox, D. R., and Oakes, D. (1984). *Analysis of Survival Data*. London: Chapman & Hall.

Edwards, D., and Berry, J. J. (1987). "The Efficiency of Simulation-Based Multiple Comparisons." *Biometrics* 43:913–928.

Elandt-Johnson, R. C., and Johnson, N. L. (1980). *Survival Models and Data Analysis*. New York: John Wiley & Sons.

Fleming, T. R., and Harrington, D. P. (1981). "A Class of Hypothesis Tests for One and Two Samples of Censored Survival Data." *Communications in Statistics—Theory and Methods* 10:763–794.

Fleming, T. R., and Harrington, D. P. (1984). "Nonparametric Estimation of the Survival Distribution in Censored Data." *Communications in Statistics—Theory and Methods* 13:2469–2486.

Fleming, T. R., and Harrington, D. P. (1991). *Counting Processes and Survival Analysis*. New York: John Wiley & Sons.

Gasser, T., and Müller, H. G. (1979). "Kernel Estimation of Regression Functions." In *Smoothing Techniques for Curve Estimation: Proceedings of a Workshop Held in Heidelberg, April 2–4, 1979*, edited by T. Gasser and M. Rosenblatt, 23–68. Vol. 757 of Lecture Notes in Mathematics. Berlin: Springer-Verlag.

Gray, R. J. (1988). "A Class of K-Sample Tests for Comparing the Cumulative Incidence of a Competing Risk." *Annals of Statistics* 16:1141–1154.

Hall, W. J., and Wellner, J. A. (1980). "Confidence Bands for a Survival Curve from Censored Data." *Biometrika* 67:133–143.

Harrington, D. P., and Fleming, T. R. (1982). "A Class of Rank Test Procedures for Censored Survival Data." *Biometrika* 69:133–143.

Hsu, J. C. (1992). "The Factor Analytic Approach to Simultaneous Inference in the General Linear Model." *Journal of Computational and Graphical Statistics* 1:151–168.

Kalbfleisch, J. D., and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley & Sons.

Klein, J. P., and Moeschberger, M. L. (1997). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: Springer-Verlag.

Kramer, C. Y. (1956). "Extension of Multiple Range Tests to Group Means with Unequal Numbers of Replications." *Biometrics* 12:307–310.

Lachin, J. M. (2000). *Biostatistical Methods: The Assessment of Relative Risks*. New York: John Wiley & Sons.

Lawless, J. F. (1982). *Statistical Methods and Methods for Lifetime Data*. New York: John Wiley & Sons.

Lee, E. T. (1992). *Statistical Methods for Survival Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Marubini, E., and Valsecchi, M. G. (1995). *Analysing Survival Data from Clinical Trials and Observational Studies*. Chichester, UK: John Wiley & Sons.

Meeker, W. Q., and Escobar, L. A. (1998). *Statistical Methods for Reliability Data*. New York: John Wiley & Sons.

Miller, R. G., Jr., and Siegmund, D. (1982). "Maximally Selected Chi-Square Statistics." *Biometrics* 1011–1016.

Nair, V. N. (1984). "Confidence Bands for Survival Functions with Censored Data: A Comparative Study." *Technometrics* 26:265–275.

Ramlau-Hansen, H. (1983a). "The Choice of a Kernel Function in the Graduation of Counting Process Intensities." *Scandinavian Actuarial Journal* 165–182.

Ramlau-Hansen, H. (1983b). "Smoothing Counting Process Intensities by Means of Kernel Functions." *Annals of Statistics* 11:453–466.

Westfall, P. H., Tobias, R. D., Rom, D., Wolfinger, R. D., and Hochberg, Y. (1999). *Multiple Comparisons and Multiple Tests Using the SAS System*. Cary, NC: SAS Institute Inc.

Xie, J., and Liu, C. (2005). "Adjusted Kaplan-Meier Estimator and Log-Rank Test with Inverse Probability of Treatment Weighting for Survival Data." *Statistics in Medicine* 24:3089–3110.

Xie, J., and Liu, C. (2011). "Adjusted Log-Rank Test for More Than Two Groups." Unpublished manuscript.

Chapter 78
The LOESS Procedure

Contents

Overview: LOESS Procedure . . . **5868**
- Local Regression and the Loess Method . . . 5868

Getting Started: LOESS Procedure . . . **5869**
- Scatter Plot Smoothing . . . 5869

Syntax: LOESS Procedure . . . **5882**
- PROC LOESS Statement . . . 5882
- BY Statement . . . 5887
- ID Statement . . . 5888
- MODEL Statement . . . 5888
- OUTPUT Statement . . . 5893
- SCORE Statement . . . 5895
- WEIGHT Statement . . . 5896

Details: LOESS Procedure . . . **5896**
- Missing Values . . . 5896
- Output Data Sets . . . 5897
- Data Scaling . . . 5898
- Direct versus Interpolated Fitting . . . 5899
- *k*-d Trees and Blending . . . 5899
- Local Weighting . . . 5900
- Iterative Reweighting . . . 5900
- Specifying the Local Polynomials . . . 5901
- Smoothing Matrix . . . 5901
- Model Degrees of Freedom . . . 5901
- Statistical Inference and Lookup Degrees of Freedom . . . 5902
- Automatic Smoothing Parameter Selection . . . 5903
- Sparse and Approximate Degrees of Freedom Computation . . . 5905
- Scoring Data Sets . . . 5906
- ODS Table Names . . . 5907
- ODS Graphics . . . 5907

Examples: LOESS Procedure . . . **5908**
- Example 78.1: Engine Exhaust Emissions . . . 5909
- Example 78.2: Sulfate Deposits in the U.S. for 1990 . . . 5915
- Example 78.3: Catalyst Experiment . . . 5920
- Example 78.4: El Niño Southern Oscillation . . . 5927

References . . . **5935**

Overview: LOESS Procedure

The LOESS procedure implements a nonparametric method for estimating regression surfaces pioneered by Cleveland, Devlin, and Grosse (1988); Cleveland and Grosse (1991); Cleveland, Grosse, and Shyu (1992). The LOESS procedure allows great flexibility because no assumptions about the parametric form of the regression surface are needed.

The SAS System provides many regression procedures such as the GLM, REG, and NLIN procedures for situations in which you can specify a reasonable parametric model for the regression surface. You can use the LOESS procedure for situations in which you do not know a suitable parametric form of the regression surface. Furthermore, the LOESS procedure is suitable when there are outliers in the data and a robust fitting method is necessary.

The main features of the LOESS procedure are as follows:

- fits nonparametric models
- supports the use of multidimensional data
- supports multiple dependent variables
- supports both direct and interpolated fitting that uses k-d trees
- performs statistical inference
- performs automatic smoothing parameter selection
- performs iterative reweighting to provide robust fitting when there are outliers in the data
- produces graphs with ODS Graphics

Local Regression and the Loess Method

Assume that for $i = 1$ to n, the ith measurement y_i of the response y and the corresponding measurement x_i of the vector $\mathbf{x}$ of p predictors are related by

$$y_i = g(x_i) + \epsilon_i$$

where g is the regression function and ϵ_i is a random error. The idea of local regression is that at a predictor $\mathbf{x}$, the regression function $g(\mathbf{x})$ can be locally approximated by the value of a function in some specified parametric class. Such a local approximation is obtained by fitting a regression surface to the data points within a chosen neighborhood of the point x_i.

In the loess method, weighted least squares is used to fit linear or quadratic functions of the predictors at the centers of neighborhoods. The radius of each neighborhood is chosen so that the neighborhood contains a specified percentage of the data points. The fraction of the data, called the *smoothing parameter,* in each local neighborhood controls the smoothness of the estimated surface. Data points in a given local neighborhood are weighted by a smooth decreasing function of their distance from the center of the neighborhood.

In a direct implementation, such fitting is done at each point at which the regression surface is to be estimated. A much faster computational procedure is to perform such local fitting at a selected sample of points in predictor space and then to blend these local polynomials to obtain a regression surface.

You can use the LOESS procedure to perform statistical inference provided that the error distribution satisfies some basic assumptions. In particular, such analysis is appropriate when the ϵ_i are iid normal random variables with mean 0. By using the iterative reweighting, the LOESS procedure can also provide statistical inference when the error distribution is symmetric but not necessarily normal. Furthermore, by doing iterative reweighting, you can use the LOESS procedure to perform robust fitting in the presence of outliers in the data.

While all output of the LOESS procedure can be optionally displayed, most often the LOESS procedure is used to produce output data sets that will be viewed and manipulated by other SAS procedures. PROC LOESS uses the Output Delivery System (ODS) to place results in output data sets. Alternatively, PROC LOESS also provides an OUTPUT statement to create SAS data sets from analysis results.

Getting Started: LOESS Procedure

Scatter Plot Smoothing

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/loessgs.sas.)

The following data from the Connecticut Tumor Registry presents age-adjusted numbers of melanoma incidences per 100,000 people for the 37 years from 1936 to 1972 (Houghton, Flannery, and Viola 1980).

```
data Melanoma;
   input  Year Incidences @@;
   format Year d4.0;
   datalines;
1936    0.9   1937   0.8  1938   0.8  1939   1.3
1940    1.4   1941   1.2  1942   1.7  1943   1.8
1944    1.6   1945   1.5  1946   1.5  1947   2.0
1948    2.5   1949   2.7  1950   2.9  1951   2.5
1952    3.1   1953   2.4  1954   2.2  1955   2.9
1956    2.5   1957   2.6  1958   3.2  1959   3.8
1960    4.2   1961   3.9  1962   3.7  1963   3.3
1964    3.7   1965   3.9  1966   4.1  1967   3.8
1968    4.7   1969   4.4  1970   4.8  1971   4.8
1972    4.8
;
```

The following PROC SGPLOT statements produce the simple scatter plot of these data displayed in Figure 78.1.

```
proc sgplot data=Melanoma;
   scatter y=Incidences x=Year;
run;
```

Figure 78.1 Scatter Plot of the Melanoma Data

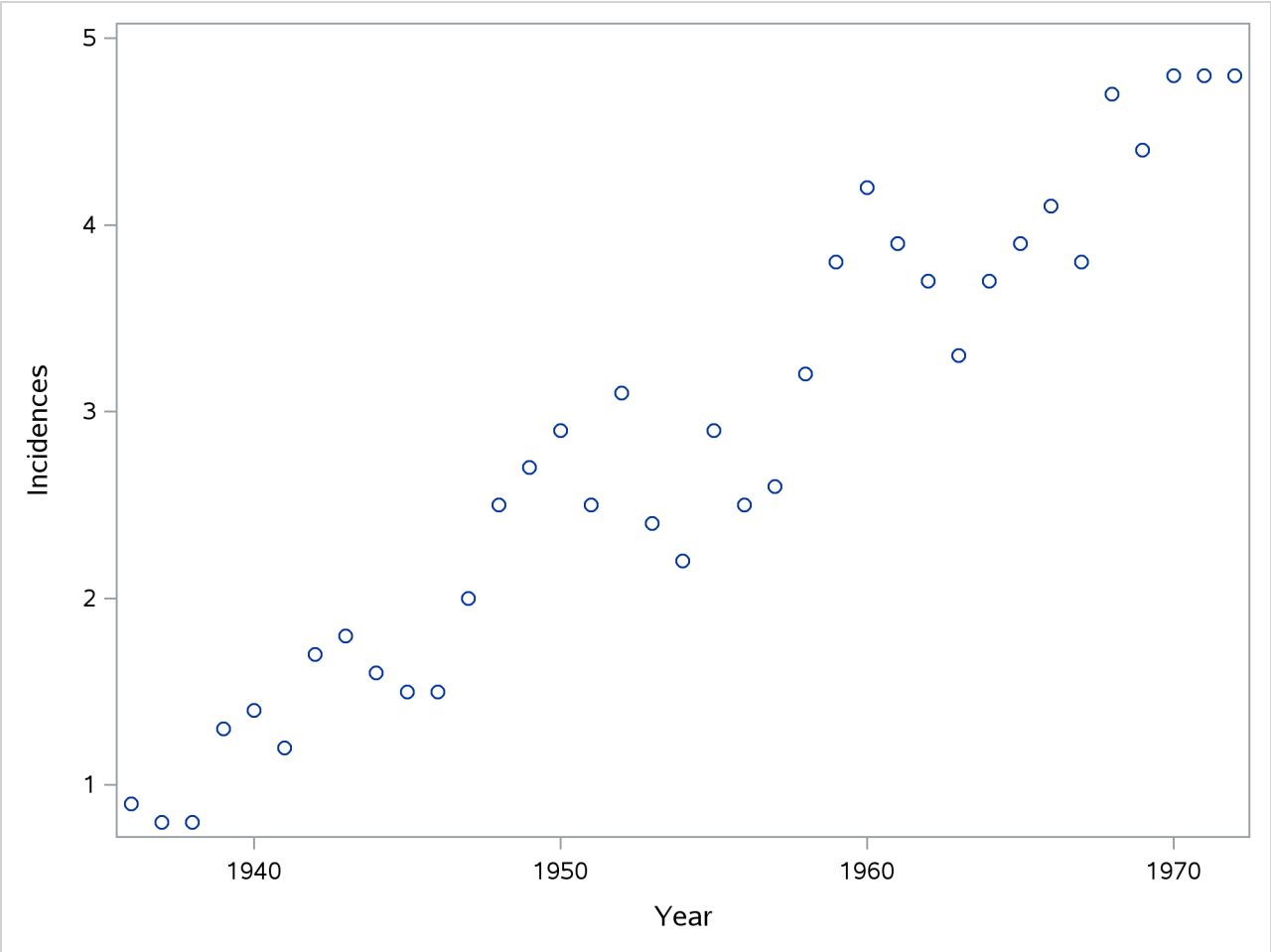

Suppose that you want to smooth the response variable Incidences as a function of the variable Year. The following PROC LOESS statements request this analysis with the default settings:

```
ods graphics on;

proc loess data=Melanoma;
   model Incidences=Year;
run;
```

You use the PROC LOESS statement to invoke the procedure and specify the data set. The MODEL statement names the dependent and independent variables.

Figure 78.2 Default Loess Fit for the Melanoma Data

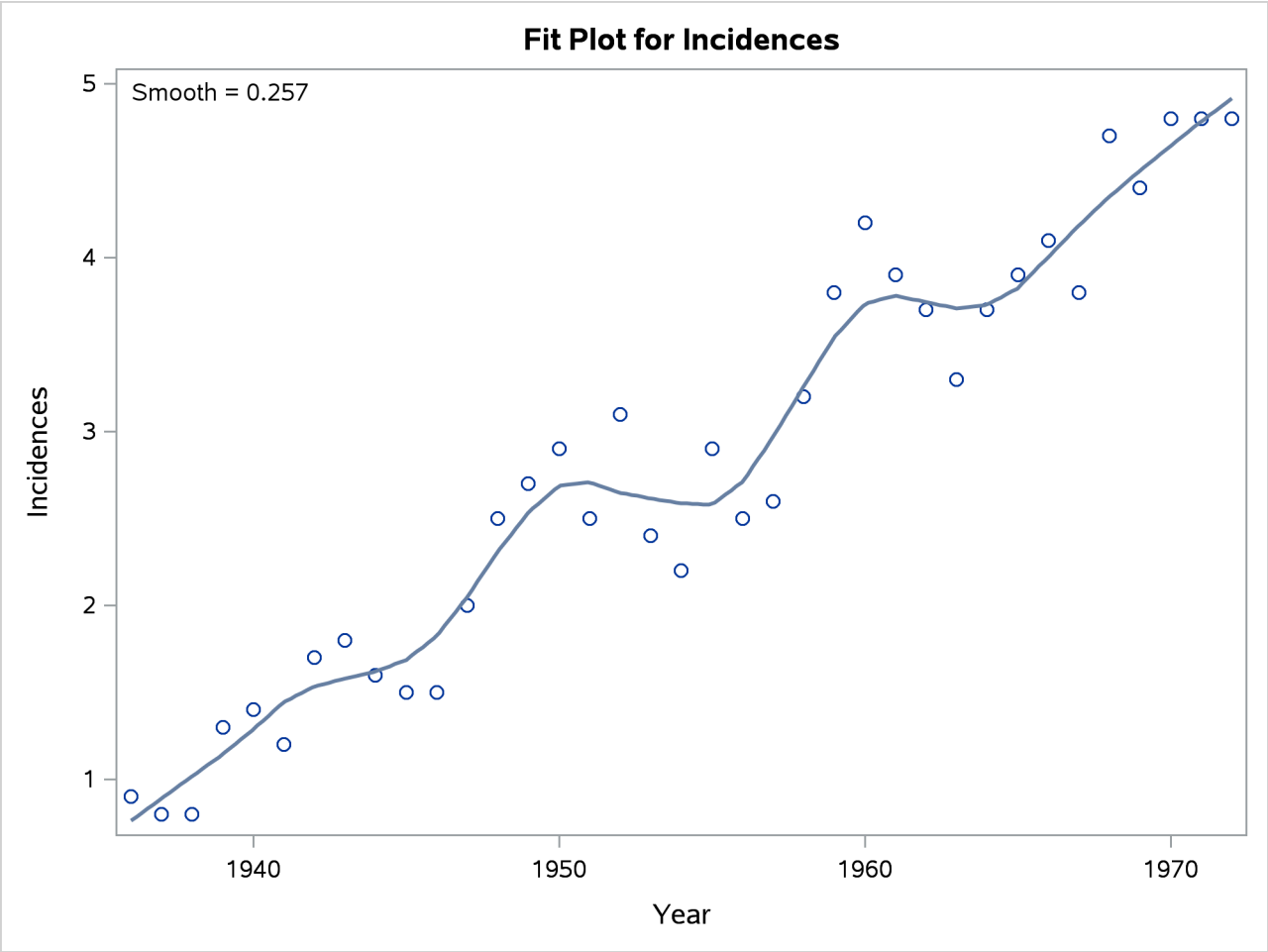

When ODS Graphics is enabled, PROC LOESS produces several default plots. Figure 78.2 shows the "Fit Plot" that overlays the loess fit on a scatter plot of the data. You can see that the loess fit captures the increasing trend in the data as well as the periodic pattern in the data, which is related to an 11-year sunspot activity cycle.

Figure 78.3 Fit Summary

The LOESS Procedure
Selected Smoothing Parameter: 0.257
Dependent Variable: Incidences

Fit Summary	
Fit Method	kd Tree
Blending	Linear
Number of Observations	37
Number of Fitting Points	37
kd Tree Bucket Size	1
Degree of Local Polynomials	1
Smoothing Parameter	0.25676
Points in Local Neighborhood	9
Residual Sum of Squares	2.03105
Trace[L]	8.62243
GCV	0.00252
AICC	-1.17277

Figure 78.3 shows the "Fit Summary" table. This table details the settings used and provides statistics about the fit that is produced. You can see that smoothing parameter value for this loess fit is 0.257. This smoothing parameter determines the fraction of the data in each local neighborhood. In this example, there are 37 data points and so the smoothing parameter value of 0.257 yields local neighborhoods containing 9 observations.

Figure 78.4 Smoothing Parameter Selection

Optimal Smoothing Criterion	
AICC	**Smoothing Parameter**
-1.17277	0.25676

The "Smoothing Criterion" table provides information about how this smoothing parameter value is selected. The default method implemented in PROC LOESS chooses the smoothing parameter that minimizes the AICC criterion (Hurvich, Simonoff, and Tsai 1998) that strikes a balance between the residual sum of squares and the complexity of the fit.

You use options in the MODEL statement to change the default settings and request optionally displayed tables. For example, the following statements request that the "Model Summary" and "Output Statistics" tables be included in the displayed output. By default, these tables are not displayed.

```
proc loess data=Melanoma;
   model Incidences=Year / details(ModelSummary OutputStatistics);
run;
```

Figure 78.5 Model Summary Table

The LOESS Procedure
Dependent Variable: Incidences

Model Summary				
Smoothing Parameter	Local Points	Residual SS	GCV	AICC
0.41892	15	3.42229	0.00339	-0.96252
0.68919	25	4.05838	0.00359	-0.93459
0.31081	11	2.51054	0.00279	-1.12034
0.20270	7	1.58513	0.00239	-1.12221
0.17568	6	1.56896	0.00241	-1.09706
0.28378	10	2.50487	0.00282	-1.10402
0.20270	7	1.58513	0.00239	-1.12221
0.25676	9	2.03105	0.00252	-1.17277
0.22973	8	2.02965	0.00256	-1.15145
0.25676	9	2.03105	0.00252	-1.17277

The “Model Summary” table shown in Figure 78.5 provides information about all the models that PROC LOESS evaluated in choosing the smoothing parameter value.

Figure 78.6 AICC Criterion by Smoothing Parameter

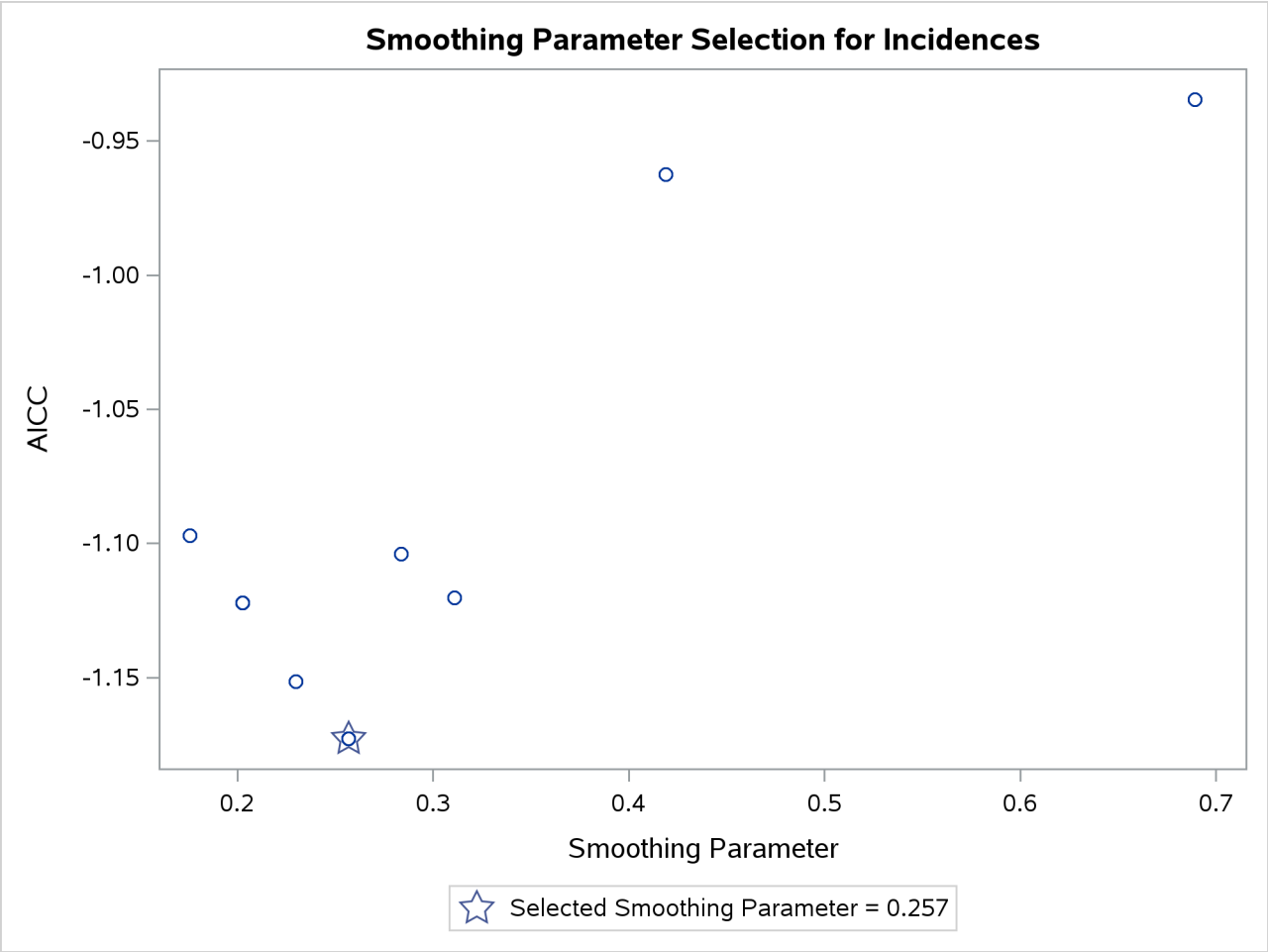

Figure 78.6 shows the "Criterion Plot" that provides a graphical display of the smoothing parameter selection process.

Figure 78.7 Output Statistics

The LOESS Procedure
Selected Smoothing Parameter: 0.257
Dependent Variable: Incidences

Output Statistics				
Obs	Year	Incidences	Predicted Incidences	Residual
1	1936	0.90000	0.76235	0.13765
2	1937	0.80000	0.88992	-0.08992
3	1938	0.80000	1.01764	-0.21764
4	1939	1.30000	1.14303	0.15697
5	1940	1.40000	1.28654	0.11346
6	1941	1.20000	1.44528	-0.24528
7	1942	1.70000	1.53482	0.16518
8	1943	1.80000	1.57895	0.22105
9	1944	1.60000	1.62058	-0.02058
10	1945	1.50000	1.68627	-0.18627
11	1946	1.50000	1.82449	-0.32449
12	1947	2.00000	2.04976	-0.04976
13	1948	2.50000	2.30981	0.19019
14	1949	2.70000	2.53653	0.16347
15	1950	2.90000	2.68921	0.21079
16	1951	2.50000	2.70779	-0.20779
17	1952	3.10000	2.64837	0.45163
18	1953	2.40000	2.61468	-0.21468
19	1954	2.20000	2.58792	-0.38792
20	1955	2.90000	2.57877	0.32123
21	1956	2.50000	2.71078	-0.21078
22	1957	2.60000	2.96981	-0.36981
23	1958	3.20000	3.26005	-0.06005
24	1959	3.80000	3.54143	0.25857
25	1960	4.20000	3.73482	0.46518
26	1961	3.90000	3.78186	0.11814
27	1962	3.70000	3.74362	-0.04362
28	1963	3.30000	3.70904	-0.40904
29	1964	3.70000	3.72917	-0.02917
30	1965	3.90000	3.82382	0.07618
31	1966	4.10000	4.00515	0.09485
32	1967	3.80000	4.18573	-0.38573
33	1968	4.70000	4.35152	0.34848
34	1969	4.40000	4.50284	-0.10284
35	1970	4.80000	4.64413	0.15587
36	1971	4.80000	4.78291	0.01709
37	1972	4.80000	4.91602	-0.11602

Figure 78.7 show the "Output Statistics" table that contains the predicted loess fit value at each observation in the input data set.

Although the default method for selecting the smoothing parameter value is often satisfactory, it is often a

good practice to examine how the loess fit varies with the smoothing parameter. In some cases, fits with different smoothing parameters might reveal important features of the data that cannot be discerned by looking at a fit with just a single "best" smoothing parameter. Example 78.4 provides such an example. You can produce the loess fits for a range of smoothing parameters by using the SMOOTH= option in the MODEL statement as follows:

```
proc loess data=Melanoma;
   model Incidences=Year/smooth=0.1 0.25 0.4 0.6 residual;
   ods output OutputStatistics=Results;
run;
```

The RESIDUAL option causes the residuals to be added to the "Output Statistics" table. Note that, even if you do not specify the DETAILS option in the MODEL statement to request the display of the "Output Statistics" table, you can use an ODS OUTPUT statement to output this and other optionally displayed tables as data sets.

PROC PRINT displays the first five observations of the Results data set:

```
proc print data=Results(obs=5);
   id obs;
run;
```

Figure 78.8 PROC PRINT Output of the Results Data Set

Obs	SmoothingParameter	Year	DepVar	Pred	Residual
1	0.1	1936	0.9	0.90000	0
2	0.1	1937	0.8	0.80000	0
3	0.1	1938	0.8	0.80000	0
4	0.1	1939	1.3	1.30000	0
5	0.1	1940	1.4	1.40000	0

Note that the fits for all the smoothing parameters are placed in single data set. A variable named SmoothingParameter that you use to distinguish each fit is included in this data set.

When you specify a list of smoothing parameters for a model and ODS Graphics is enabled, PROC LOESS produces a panel containing up to six plots that show the fit obtained for each value of the smoothing parameter that you specify. If you specify more than six smoothing values, then multiple panels are produced. For each regressor, PROC LOESS also produces panels of the residuals versus each regressor by the smoothing parameters that you specify.

Figure 78.9 Loess Fits for a Range of Smoothing Parameters

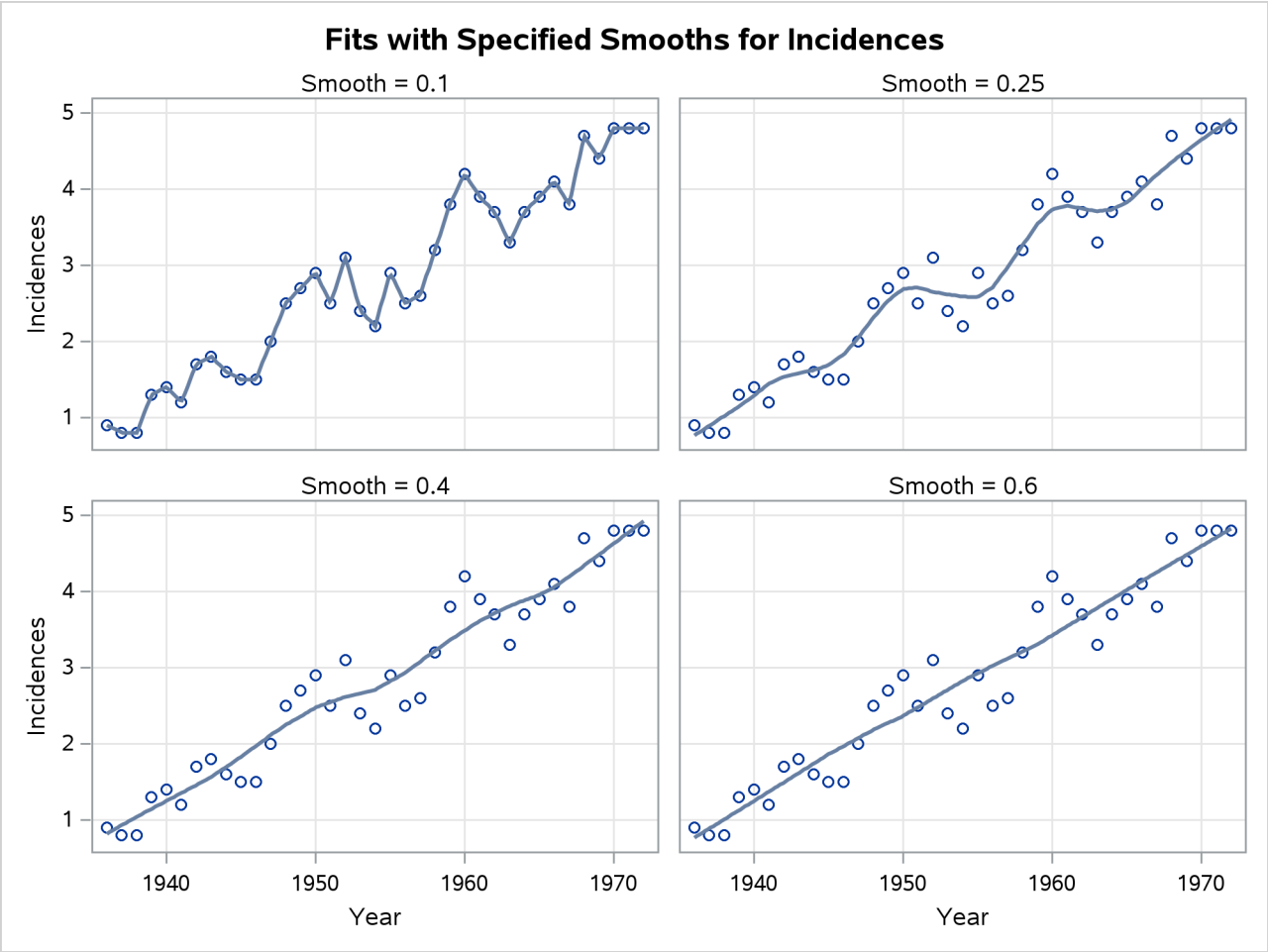

If you examine the plots in Figure 78.9, you see that a visually reasonable fit is obtained with smoothing parameter values of 0.25. With smoothing parameter value 0.1, there is gross overfitting in the sense that the original data are exactly interpolated. When the smoothing parameter value is 0.4, you obtain an overly smooth fit where the contribution of the sunspot cycle has been mostly averaged away. At smoothing parameter value 0.6 the fit shows just the increasing trend in the data.

It is also instructive to look at scatter plots of the residuals for each of the fits. These are also produced by default by PROC LOESS when ODS Graphics is enabled.

Figure 78.10 Residuals of Loess Fits for a Range of Smoothing Parameters

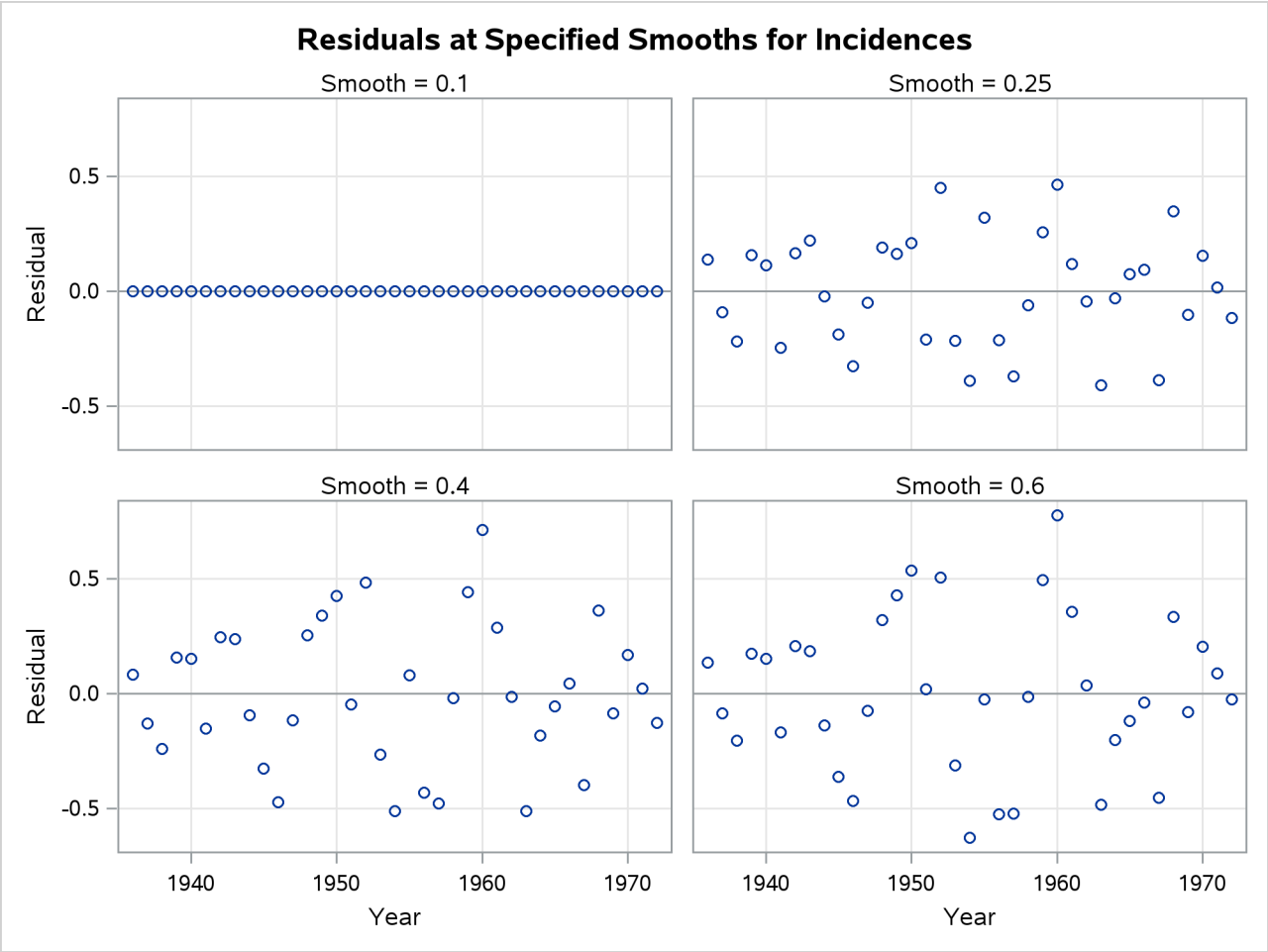

Figure 78.10 shows a scatter plot of the residuals by year for each smoothing parameter value. One way to discern patterns in these residuals is to superimpose a loess fit on each plot in the panel. You request loess fits on the residual plots in this panel by specifying the SMOOTH= suboption of the PLOTS=RESIDUALSBYSMOOTH option in the PROC LOESS statement. Note that the loess fits that are displayed on each of the residual plots are obtained independently of the loess fit that produces these residuals. The following statements show how you do this for the Melanoma data.

```
proc loess data=Melanoma plots=ResidualsBySmooth(smooth);
   model Incidences=Year/smooth=0.1 0.25 0.4 0.6;
run;
```

Figure 78.11 Residuals with Superimposed Loess Fits

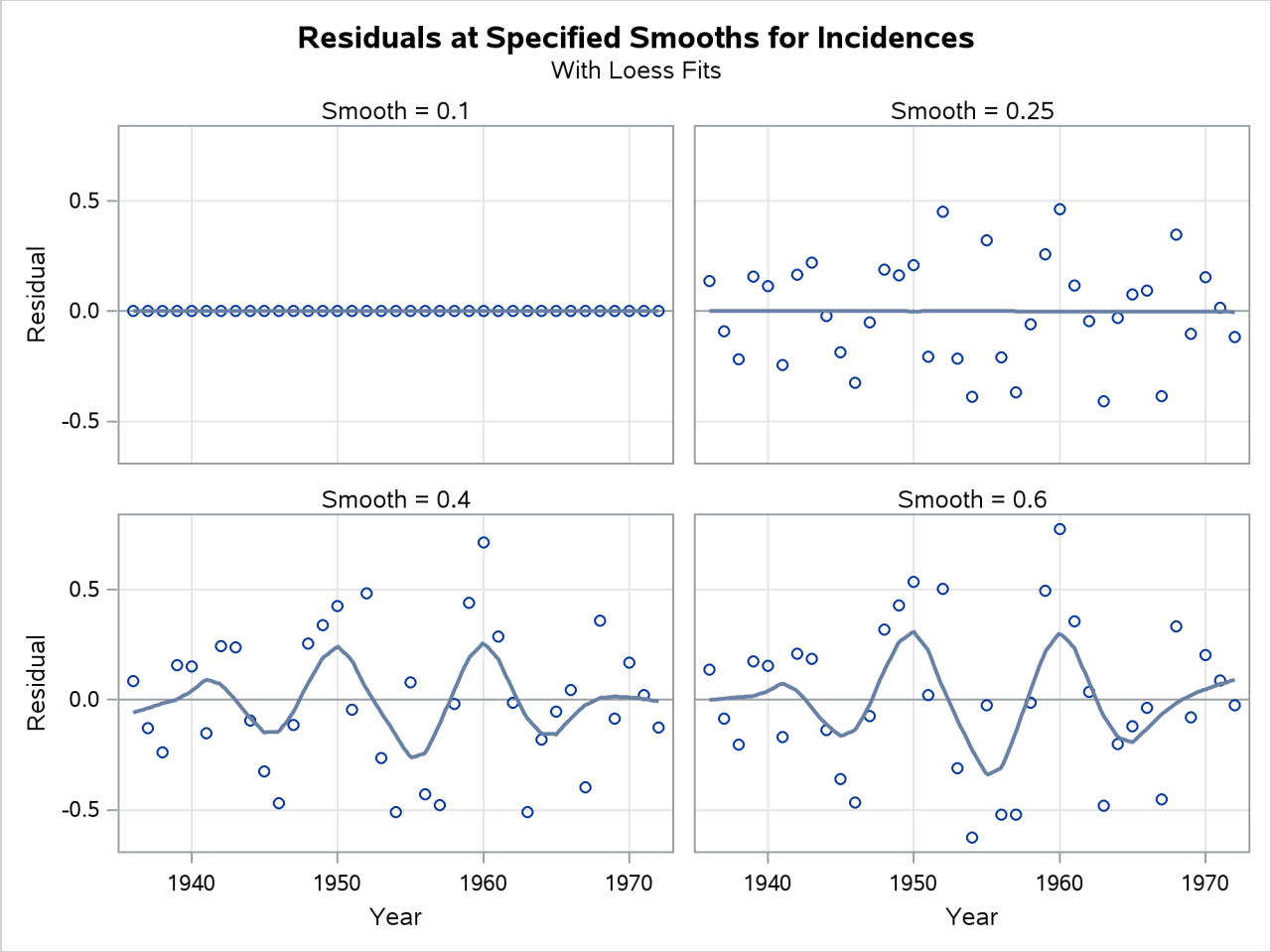

The loess fits shown on the plots in Figure 78.11 help confirm the conclusions obtained when you look at Figure 78.9. Note that residuals for smoothing parameter value 0.25 do not exhibit any pattern, confirming that at this value the loess fit of the melanoma data has successfully modeled the variation in this data. By contrast, the residuals for the fit with smoothing parameter 0.6 retain the variation caused by the sunspot cycle.

The examination of the fits and residuals obtained with a range of smoothing parameter values confirms that the value of 0.257 that PROC LOESS selects automatically is appropriate for these data. The next step in this analysis is to examine fit diagnostics and produce confidence limit for the fit. If ODS Graphics is enabled, then a panel of fit diagnostics is produced. Furthermore, you can request prediction confidence limits by adding the CLM option in the MODEL statement. By default 95% limits are produced, but you can use the ALPHA= option in the MODEL statement to change the significance level. The following statements request 90% confidence limits.

```
proc loess data=Melanoma;
   model Incidences=Year/clm alpha=0.1;
run;

ods graphics off;
```

Figure 78.12 Fit Diagnostics

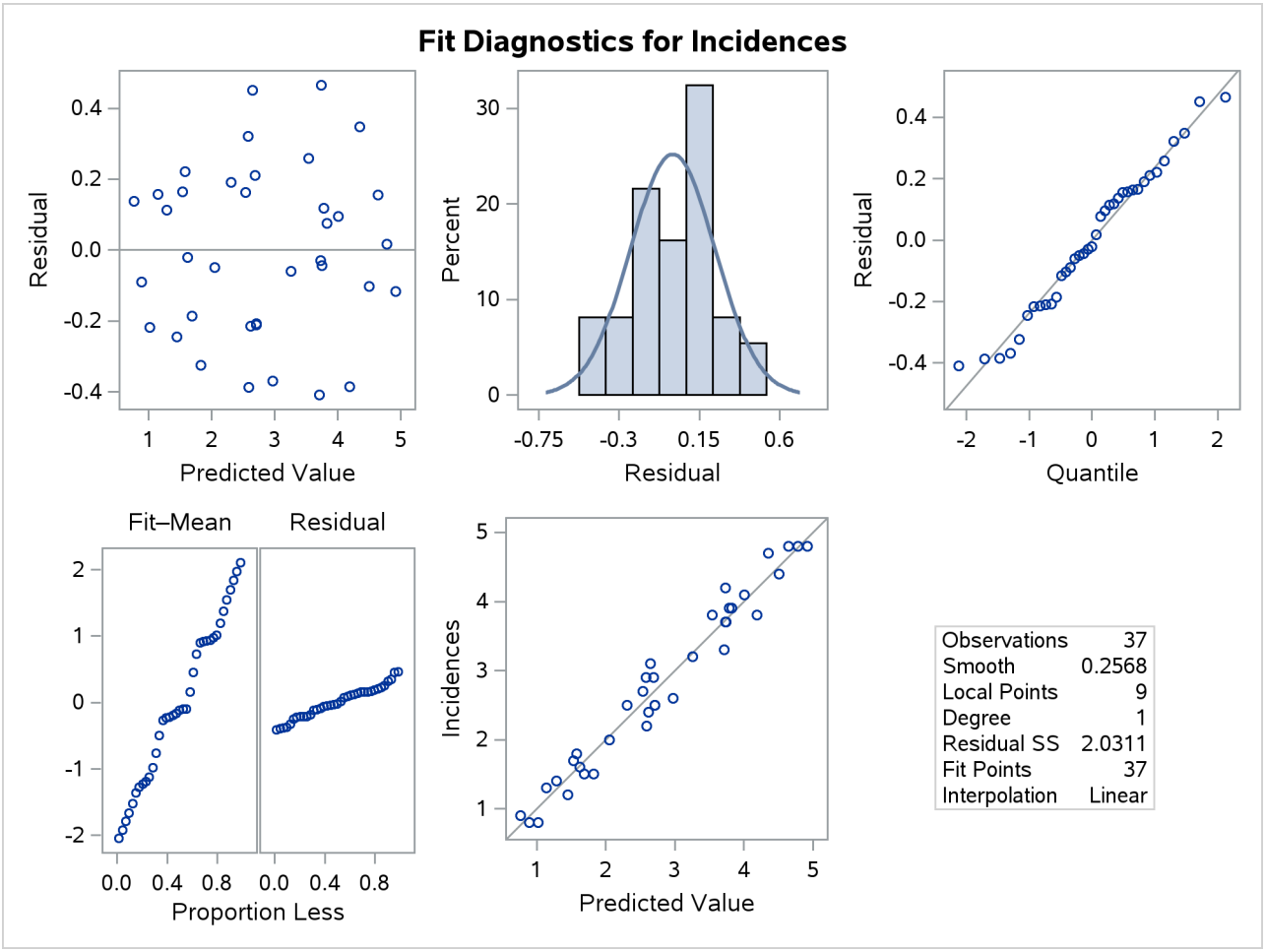

Figure 78.12 shows the fit diagnostics panel. The histogram of the residuals with overlaid normal density estimator and the normal quantile plot show that the residuals do exhibit some small departure from normality. The "Residual-Fit" spread plot shows that the spread in the centered fit is much wider that the spread in the residuals. This indicates that the fit has accounted for most of the variation in the incidences of melanoma in this data. This conclusion is supported by the absence of any clear pattern in the scatter plot of residuals by predicted values and the closeness of the points to the 45-degree reference line in the plot of observed by predicted values.

Figure 78.13 Loess Fit of Melanoma Data with 90% Confidence Limits

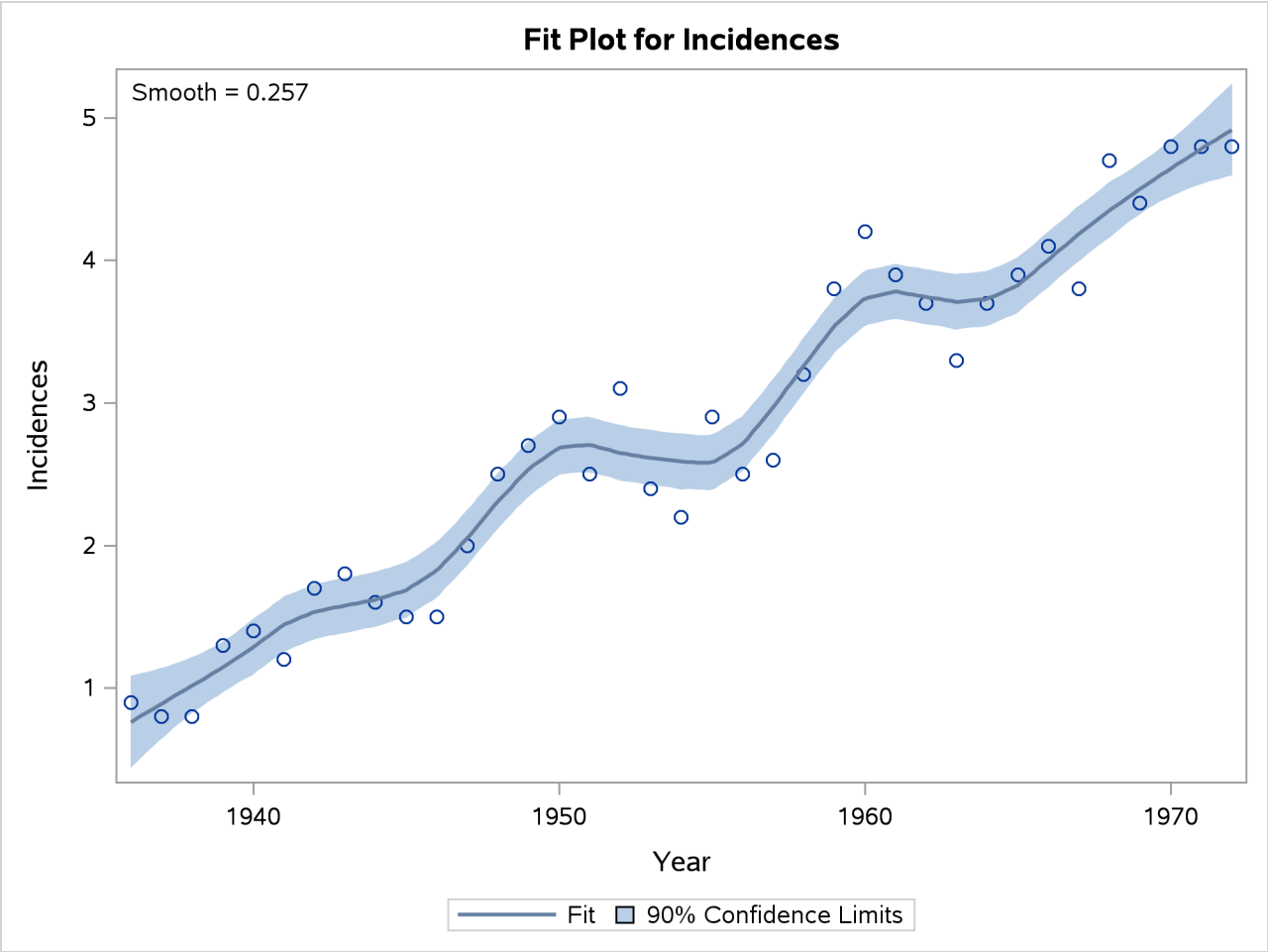

Finally, Figure 78.13 shows the selected loess fit with 90% confidence limits.

Syntax: LOESS Procedure

The following statements are available in the LOESS procedure:

PROC LOESS < **DATA=***SAS-data-set* > **;**
  MODEL *dependents* **=** *regressors* < / *options* > **;**
  OUTPUT < **OUT=***SAS-data-set* > < *keyword* < **=***name* > > < . . . *keyword* < **=***name* > > < / *options* > **;**
  ID *variables* **;**
  BY *variables* **;**
  WEIGHT *variable* **;**
  SCORE DATA=*SAS-data-set* < **ID=(***variable-list***)** > < / *options* > **;**

The PROC LOESS and MODEL statements are required. The OUTPUT, BY, WEIGHT, and ID statements are optional. The SCORE statement is optional, and more than one SCORE statement can be used.

The statements used with the LOESS procedure, in addition to the PROC LOESS statement, are as follows.

BY	specifies variables to define subgroups for the analysis.
ID	names variables to identify observations in the displayed output.
MODEL	specifies the dependent and independent variables in the loess model, details and parameters for the computational algorithm, and the required output.
OUTPUT	creates an output data set containing predicted values, residuals, and results of statistical inference.
SCORE	specifies a data set containing observations to be scored.
WEIGHT	declares a variable to weight observations.

PROC LOESS Statement

PROC LOESS < *options* > **;**

The PROC LOESS statement invokes the LOESS procedure. The PROC LOESS statement is required. You can specify the following *options* in the PROC LOESS statement:

DATA=*SAS-data-set*
  names the SAS data set to be used by PROC LOESS. If the DATA= option is not specified, PROC LOESS uses the most recently created SAS data set.

PLOTS < (*global-plot-options*) > < = *plot-request* < (*options*) > >
PLOTS < (*global-plot-options*) > < = (*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >
  controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=none
plots=residuals(smooth)
plots(unpack)=diagnostics
plots(only)=(fit residualHistogram)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc loess;
   model y = x;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you but do not specify the PLOTS= option, then PROC LOESS produces a default set of plots. The following table lists the default set of plots produced.

Table 78.1 Default Graphs Produced

Plot	Conditional On
ContourFitPanel	SMOOTH= option specified in the MODEL statement
ContourFit	Model with two regressors
CriterionPlot	Smoothing parameter selection performed
DiagnosticsPanel	Unconditional
ResidualsBySmooth	SMOOTH= option specified in the MODEL statement
ResidualPanel	Unconditional
FitPanel	SMOOTH= option specified in the MODEL statement
FitPlot	Model with one regressor
ScorePlot	One or more SCORE statements and a model with one regressor

For models with multiple dependent variables, separate plots are produced for each dependent variable. For models where multiple smoothing parameters are requested with the SMOOTH= option in the MODEL statement and smoothing parameter value selection is not requested, separate plots are produced for each smoothing parameter. If smoothing parameter value selection is requested with the SELECT= option in the MODEL statement, then the plots are produced for the selected model only. However, if you specify the STEPS suboption of the SELECT= option, then plots are produced for all smoothing parameters examined in the selection process.

The *global-plot-options* apply to all relevant plots generated by the LOESS procedure, unless they are overridden with a *specific-plot-option*. The *global-plot-options* supported by the LOESS procedure follow.

Global Plot Options

MAXPOINTS=NONE | *number*

specifies that plots with elements that require processing more than *number* points are suppressed. The default is MAXPOINTS=5000. This cutoff is ignored if you specify MAXPOINTS=NONE.

ONLY

suppresses the default plots. Only the plots specifically requested are produced.

UNPACK

suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACK to get each plot individually. You can specify PLOTS(UNPACK) to unpack the default plots. You can also specify UNPACK as a suboption with CONTOURFITPANEL, DIAGNOSTICS, FITPANEL, RESIDUALS and RESIDUALSBYSMOOTH.

Specific Plot Options

The following listing describes the specific plots and their *options*.

ALL

requests that all plots appropriate for the particular analysis be produced. You can specify other options with ALL; for example, to request all plots and unpack only the residuals, specify PLOTS=(ALL RESIDUALS(UNPACK)).

CONTOURFIT < (*contour-options*) >

produces a contour plot of the fitted surface overlaid with a scatter plot of the data for models with two regressors. Contour plots are not produced if you specify the DIRECT option in the MODEL statement. You can use the following *contour-options* to control how the observations are displayed:

OBS=GRADIENT

specifies that observations be displayed as circles colored by the observed response. The same color gradient is used to display the fitted surface and the observations. Observations where the predicted response is close to the observed response have similar colors—the greater the contrast between the color of an observation and the surface, the larger the residual is at that point. OBS=GRADIENT is the default if you do not specify any *contour-options*.

OBS=NONE

suppresses the observations.

OBS=OUTLINE

specifies that observations be displayed as circles with a border but with a completely transparent fill.

OBS=OUTLINEGRADIENT

is the same as OBS=GRADIENT except that a border is shown around each observation. This option is useful to identify the location of observations where the residuals are small, because at these points the color of the observations and the color of the surface are indistinguishable.

CONTOURFITPANEL < (< UNPACK > < *contour-options* >) >

produces panels of contour plots overlaid with a scatter plot of the data for each smoothing parameter specified in the SMOOTH= option in the MODEL statement, for models with two regressors. This plot is not produced if you specify the DIRECT option in the MODEL statement. If you do not specify the SMOOTH= option or if the model does not have two regressors, then this plot is not produced. If you specify the SELECT= option in addition to the SMOOTH= option in the MODEL statement, then you need to additionally specify the STEPS suboption of the SELECT= option to obtain this plot. Note that each panel contains at most six plots, and multiple panels are used in the case that there are more than six smoothing parameters in the SMOOTH= option in the MODEL statement. See the CONTOURFIT option for a description of the individual plots in this panel. The UNPACK option suppresses paneling, and the *contour-options* are the same as for the CONTOURFIT option.

CRITERIONPLOT | CRITERION

displays a scatter plot of the value of the SELECT= criterion versus the smoothing parameter value for all smoothing parameter values examined in the selection process. This plot is not produced if smoothing parameter selection is not done.

DIAGNOSTICSPANEL | DIAGNOSTICS < (UNPACK) >

produces a summary panel of fit diagnostics consisting of the following:

- residuals versus the predicted values
- histogram of the residuals
- normal quantile plot of the residuals
- a “Residual-Fit” (or RF) plot consisting of side-by-side quantile plots of the centered fit and the residuals.
- dependent variable values versus the predicted values

You can request the five plots in this panel as individual plots by specifying the UNPACK option. You can also request individual plots in the panel by name without having to unpack the panel. Note that the fit diagnostics panel is produced by default whenever ODS Graphics is enabled.

FITPANEL < (UNPACK) >

produces panels of plots showing the fitted LOESS curve overlaid on a scatter plot of the input data for each smoothing parameter specified in the SMOOTH= option in the MODEL statement. If you do not specify the SMOOTH= option or the model has more than one regressor, then this plot is not produced. If you specify the SELECT= option in addition to the SMOOTH= option in the MODEL statement, then you need to additionally specify the STEPS suboption of the SELECT= option to obtain this plot. Note that each panel contains at most six plots, and multiple panels are used in the case that there are more than six smoothing parameters in the SMOOTH= option in the MODEL statement. If the CLM option is specified in the MODEL statement, then a confidence band at the significance level specified in the ALPHA= option is included in each plot in the panels. If you specify the UNPACK option, then all fit panels are unpacked.

FITPLOT | FIT

produces a scatter plot of the input data with the fitted LOESS curve overlaid for models with a single regressor. If the CLM option is specified in the MODEL statement, then a confidence band at the significance level specified in the ALPHA= option is included in the plot.

NONE

suppresses all plots.

OBSERVEDBYPREDICTED

produces a scatter plot of the dependent variable values by the predicted values.

QQPLOT | QQ

produces a normal quantile plot of the residuals.

RESIDUALSBYSMOOTH < (< UNPACK > < SMOOTH >) >

produces for each regressor panels of plots showing the residuals of the LOESS fit versus the regressor for each smoothing parameter specified in the SMOOTH= option in the MODEL statement. If you do not specify the SMOOTH= option, then this plot is not produced. If you specify the SELECT= option in addition to the SMOOTH= option in the MODEL statement, then you need to additionally specify the STEPS suboption of the SELECT= option to obtain this plot. Note that each panel contains at most six plots, and multiple panels are used in the case that there are more than six smoothing parameters in the SMOOTH= option in the MODEL statement. If you specify the UNPACK option, then all RESIDUALSBYSMOOTH panels are unpacked.

The SMOOTH option requests that a nonparametric fit line be shown in each plot in the panel. The type of nonparametric fit and the options used are controlled by the template that underlies this plot. In the standard template that is provided, the nonparametric smooth is specified to be a loess fit corresponding to the default options of PROC LOESS, except that the PRESEARCH suboption is always used. It is important to note that the loess fit that is shown in each of the residual plots is computed independently of the loess fit that is used to obtain the residuals.

RESIDUALBYPREDICTED

produces a scatter plot of the residuals by the predicted values.

RESIDUALHISTOGRAM

produces a histogram of the residuals.

RESIDUALPANEL | RESIDUALS < (*residual-options*) >

produces panels of the residuals versus the regressors in the model. Note that each panel contains at most six plots, and multiple panels are used when there are more than six regressors in the model.

The following *residual-options* are available:

SMOOTH

requests that a nonparametric fit line be shown in each plot in the panel. The type of nonparametric fit and the options used are controlled by the template that underlies this plot. In the standard template that is provided, the nonparametric smooth is specified to be a loess fit corresponding to the default options of PROC LOESS, except that the PRESEARCH suboption is always used. It is important to note that the loess fit that is shown in each of the residual plots is computed independently of the loess fit that is used to obtain the residuals.

UNPACK
suppresses paneling.

RFPLOT | RF
produces a "Residual-Fit" (or RF) plot consisting of side-by-side quantile plots of the centered fit and the residuals. This plot "shows how much variation in the data is explained by the fit and how much remains in the residuals" (Cleveland 1993).

SCOREPLOT | SCORE
produces a scatter plot of the scored values at the score points for each SCORE statement. SCORE plots are not produced for models with more than one regressor. If the CLM option is specified in the MODEL statement, then confidence bars at the significance level specified in the ALPHA= option are shown at score data points.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC LOESS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the LOESS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

ID Statement

ID *variables* **;**

The ID statement is optional, and more than one ID statement can be used. The variables listed in any of the ID statements are displayed in the "Output Statistics" table beside each observation. Any variables specified as a regressor or dependent variable in the MODEL statement already appear in the "Output Statistics" table and are not treated as ID variables, even if they appear in the variable list of an ID statement.

MODEL Statement

MODEL *dependents* **=** *regressors* < / *options* > **;**

The MODEL statement names the dependent variables and the independent variables. Variables specified in the MODEL statement must be numeric variables in the data set being analyzed.

Table 78.2 summarizes the *options* available in the MODEL statement.

Table 78.2 Summary of MODEL Statement Options

Option	Description
Fit Options	
BUCKET=	Specifies the number of points in *k*-d tree buckets
DEGREE=	Specifies the degree of local polynomials (1 or 2)
DFMETHOD=	Specifies the method of computing lookup degrees of freedom
DIRECT	Specifies direct fitting at every data point
DROPSQUARE=	Specifies the variables whose squares are to be dropped from local quadratic polynomials
INTERP=	Specifies the interpolating polynomials (linear or cubic)
ITERATIONS=	Specifies the number of reweighting iterations
SCALE=	Specifies the method used to scale the regressor variables
SELECT=	Specifies that automatic smoothing parameter selection be done
SMOOTH=	Specifies the list of smoothing values
Output Statistics Table Options	
ALL	Requests CLM, RESIDUAL, SCALEDINDEP, STD, and T options
CLM	Displays confidence limits for mean predictions
RESIDUAL	Displays residuals
SCALEDINDEP	Displays scaled independent variable coordinates
STD	Displays standard errors of the mean predicted values
T	Displays *t* statistics
Other Options	
ALPHA=	Sets significance level for confidence intervals
DETAILS=	Specifies which tables are to be displayed
TRACEL	Displays the trace of the smoothing matrix

The following *options* are available in the MODEL statement after a slash (/).

ALL
requests all these options: CLM, RESIDUAL, SCALEDINDEP, STD, and T.

ALPHA=*number*
sets the significance level used for the construction of confidence intervals for the current MODEL statement. The value must be between 0 and 1; the default value of 0.05 results in 95% intervals.

BUCKET=*number*
specifies the maximum number of points in the leaf nodes of the k-d tree. The default value used is $s * n/5$, where s is a smoothing parameter value specified using the SMOOTH= option and n is the number of observations being used in the current BY group. The BUCKET= option is ignored if the DIRECT option is specified.

CLM
requests that $100(1-\alpha)\%$ confidence limits on the mean predicted value be added to the "Output Statistics" table. By default, 95% limits are computed; the ALPHA= option in the MODEL statement can be used to change the significance level. The use of this option implicitly selects the model option DFMETHOD=EXACT if the DFMETHOD= option has not been explicitly used.

DEGREE=1 | 2
sets the degree of the local polynomials to use for each local regression. The valid values are 1 for local linear fitting and 2 for local quadratic fitting, with 1 being the default.

DETAILS < (*tables* **) >**
selects which tables to display, where *tables* is one or more of the specifications KDTREE, MODELSUMMARY, OUTPUTSTATISTICS, and PREDATVERTICES:

- KDTREE displays the k-d tree structure.
- MODELSUMMARY displays the fit criteria for all smoothing parameter values that are specified in the SMOOTH= option in the MODEL statement, or that are fit with automatic smoothing parameter selection.
- OUTPUTSTATISTICS displays the predicted values and other requested statistics at the points in the input data set.
- PREDATVERTICES displays fitted values and coordinates of the k-d tree vertices where the local least squares fitting is done.

The KDTREE and PREDATVERTICES specifications are ignored if the DIRECT option is specified in the MODEL statement. Specifying the option DETAILS with no qualifying list outputs all tables.

DFMETHOD=NONE | EXACT | APPROX < (*approx-options***) >**
specifies the method used to calculate the lookup degrees of freedom used in performing statistical inference. The default is DFMETHOD=NONE, unless you specify any of the MODEL statement options ALL, CLM, STD, and T, or any SCORE statement CLM option, in which case the default is DFMETHOD=EXACT.

You can specify the following *approx-options* in parentheses after the DFMETHOD=APPROX option:

QUANTILE=*number*
specifies that the smallest 100(*number*)% of the nonzero coefficients in the smoothing matrix be set to zero in computing the approximate lookup degrees of freedom. The default value is QUANTILE=0.9.

CUTOFF=*number*
specifies that coefficients in the smoothing matrix whose magnitude is less than the specified value be set to zero in computing the approximate lookup degrees of freedom. Using the CUTOFF= option overrides the QUANTILE= option.

See the section "Sparse and Approximate Degrees of Freedom Computation" on page 5905 for a description of the method used when the DFMETHOD=APPROX option is specified.

DIRECT
specifies that local least squares fits are to be done at every point in the input data set. When the direct option is not specified, a computationally faster method is used. This faster method performs local fitting at vertices of a k-d tree decomposition of the predictor space followed by blending of the local polynomials to obtain a regression surface.

DROPSQUARE=(*variables***)**
specifies the quadratic monomials to exclude from the local quadratic fits. This option is ignored unless the DEGREE=2 option has been specified.

For example,

```
model z=x y / degree=2 dropsquare=(y)
```

uses the monomials 1, x, y, x^2, and xy in performing the local fitting.

INTERP=LINEAR | CUBIC
specifies the degree of the interpolating polynomials used for blending local polynomial fits at the k-d tree vertices. This option is ignored if the DIRECT option is specified in the model statement. INTERP=CUBIC is not supported for models with more than two regressors. The default is INTERP=LINEAR.

ITERATIONS=*number*
specifies the total number of iterations to be done. The first iteration performs an initial LOESS fit. Subsequent iterations perform iterative reweighting. Such iterations are appropriate when there are outliers in the data or when the error distribution is a symmetric long-tailed distribution. The default number of iterations is 1.

RESIDUAL | R
specifies that residuals be included in the "Output Statistics" table.

SCALE=NONE | SD < (*number***) >**
specifies the scaling method to be applied to scale the regressors. The default is NONE, in which case no scaling is applied. A specification of SD(*number*) indicates that a trimmed standard deviation is to be used as a measure of scale, where *number* is the trimming fraction. A specification of SD with no qualification defaults to 10% trimmed standard deviation.

SCALEDINDEP

specifies that scaled regressor coordinates be included in the output tables. This option is ignored if the SCALE= model option is not used or if SCALE=NONE is specified.

SELECT=*criterion* **<(<GLOBAL> <PRESEARCH> <STEPS> <RANGE(***lower***,***upper***)>)>**
SELECT=*DFCriterion* **<(***target* **<GLOBAL> <PRESEARCH> <STEPS> <RANGE(***lower***,***upper***)>)>**

specifies that automatic smoothing parameter selection be done using the named *criterion* or *DFCriterion*. Valid values for the *criterion* are as follows:

AICC specifies the AIC_C criterion (Hurvich, Simonoff, and Tsai 1998).

AICC1 specifies the AIC_{C_1} criterion (Hurvich, Simonoff, and Tsai 1998).

GCV specifies the generalized cross validation criterion (Craven and Wahba 1979).

The *DFCriterion* specifies the measure used to estimate the model degrees of freedom. The measures implemented in PROC LOESS all depend on prediction matrix $\mathbf{L}$ relating the observed and predicted values of the dependent variable. Valid values for the *DFCriterion* are as follows:

DF1 specifies $\mathrm{Trace}(\mathbf{L})$.

DF2 specifies $\mathrm{Trace}(\mathbf{L}'\mathbf{L})$.

DF3 specifies $2\mathrm{Trace}(\mathbf{L}) - \mathrm{Trace}(\mathbf{L}'\mathbf{L})$.

For both types of selection, the smoothing parameter value is selected to yield a minimum of an optimization criterion. If you specify *criterion* as one of AICC, AICC1, or GCV, the optimization criterion is the specified *criterion*. If you specify *DFCriterion* as one of DF1, DF2, or DF3, the optimization criterion is $|\mathit{DFCriterion} - \mathit{target}|$, where *target* is a specified target degree of freedom value. Note that if you specify a *DFCriterion*, then you must also specify a target value. See the section "Automatic Smoothing Parameter Selection" on page 5903 for definitions and properties of the selection criteria.

The selection is done as follows:

- If you specify the SMOOTH=*value-list* option, then PROC LOESS selects the largest value in this list that yields the global minimum of the specified optimization criterion.
- If you do not specify the SMOOTH= option, then PROC LOESS finds a local minimum of the specified optimization criterion by using a golden section search of values less than or equal to one.

You can specify the following suboptions in parentheses after the specified criterion to alter the behavior of the SELECT= option:

GLOBAL

specifies that a global minimum be found within the range of smoothing parameter values examined. This suboption has no effect if you also specify the SMOOTH= option in the MODEL statement.

PRESEARCH

requests an initial grid search to find a smoothing parameter range within which the subsequent golden section search is done. The initial point in this grid is the smoothing parameter value corresponding to the smallest number of points, n, in the local neighborhoods that yields a fit that does not interpolate all the data points. Subsequent fits with number of local points $n + 1$, $n + 2$, $n + 4$, $n + 8$, ... are evaluated until either the number of local points exceeds the number of fitting points or the SELECT=criterion starts increasing. This suboption is ignored if you additionally specify the GLOBAL suboption of the SELECT= option or if you specify the SMOOTH= option in the MODEL statement. If you additionally specify the RANGE= suboption, then the golden section search is done on the intersection of the range found by this grid search and the range that you specify in the RANGE= suboption. This option is useful for data exhibiting features at multiple scales, because in such cases the SELECT= criterion often has multiple local minima. Using the PRESEARCH option increases the likelihood that the golden section search will find the global minimum of the SELECT= criterion. See Example 78.4 for such an example.

RANGE(*lower*,*upper*)

specifies that only smoothing parameter values greater than or equal to *lower* and less than or equal to *upper* be examined.

STEPS

specifies that all models evaluated in the selection process be displayed.

For models with one dependent variable, if you specify neither the SELECT= nor the SMOOTH= options in the MODEL statement, then PROC LOESS uses SELECT=AICC.

The following table summarizes how the smoothing parameter values are chosen for various combinations of the SMOOTH= option, the SELECT= option, and the SELECT= option modifiers.

Table 78.3 Smoothing Parameter Value(s) Used for Combinations of SMOOTH= and SELECT= OPTIONS for Models with One Dependent Variable

Syntax	Search Method	Search Domain
default	Golden section using AICC	$(0, 1]$
SMOOTH=*list*	No selection	Values in *list*
SMOOTH=*list* **SELECT=***criterion*	Global	Values in *list*
SMOOTH=*list* **SELECT=***criterion* (**RANGE**(l, u))	Global	Values in *list* within $[l, u]$
SELECT=*criterion*	Golden section	$(0, 1]$
SELECT=*criterion* (**RANGE**(*l*,*u*))	Golden section	$[l, u]$
SELECT=*criterion* (**GLOBAL**)	Global	$(0, 1]$
SELECT=*criterion* (**GLOBAL RANGE**(l, u))	Global	$[l, u]$

Some examples of using the SELECT= option follow:

SELECT=GCV	specifies selection that uses the GCV *criterion*.
SELECT=DF1(6.3)	specifies selection that uses the DF1 *DFCriterion* with target value 6.3.
SELECT=AICC(STEPS)	specifies selection that uses the AICC *criterion*, showing all step details.
SELECT=DF2(7 GLOBAL)	specifies selection that uses a global search algorithm to find the smoothing parameter that yields the DF2 *DFCriterion* closest to the target value 7.

NOTE: The SELECT= option cannot be used for models with more than one dependent variable.

SMOOTH=*value-list*

specifies a list of positive smoothing parameter values. If you do not specify the SELECT= option in the MODEL statement, then a separate fit is obtained for each SMOOTH= value specified. If you do specify the SELECT= option, then models with all values specified in the SMOOTH= list are examined, and PROC LOESS selects the value that minimizes the criterion specified in the SELECT= option.

For models with two or more dependent variables, if the SMOOTH= option is not specified in the MODEL statement, then SMOOTH=0.5 is used as a default.

STD

specifies that standard errors of the mean predicted values be included in the "Output Statistics" table. The use of this option implicitly selects the model option DFMETHOD=EXACT if the DFMETHOD= option has not been explicitly used.

T

specifies that *t* statistics are to be included in the "Output Statistics" table. The use of this option implicitly selects the model option DFMETHOD=EXACT if the DFMETHOD= option has not been explicitly used.

TRACEL

specifies that the trace of the prediction matrix as well as the GCV and AICC statistics be included in the "Fit Summary" table. The use of any of the MODEL statement options ALL, CLM, DFMETHOD=EXACT, DIRECT, SELECT=, STD, and T implicitly selects the TRACEL option.

OUTPUT Statement

OUTPUT <**OUT=** *SAS-data-set*> <*keyword* <**=** *name*>> <. . . *keyword* <**=***name*>> <**/** *options*> **;**

The OUTPUT statement creates a new SAS data set that saves the predicted values and other requested statistics that are calculated after models for all smoothing parameter values that are specified in the SMOOTH= option in the MODEL statement have been fit. If you do not specify a *keyword*, then only the predicted response is included.

All the variables in the original data set are included in the new data set, along with variables created by the OUTPUT statement. These new variables contain the predicted values and a variety of other statistics that are calculated for each observation in the data set.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

You can specify the following *options* in the OUTPUT statement:

OUT=*SAS data set*
specifies the name of the new data set. By default, the procedure uses the DATA*n* convention to name the new data set.

keyword < **=***name* >
specifies the statistics to include in the output data set as new variables and optionally names the new variables. Specify a *keyword* for each desired statistic (see the following list of *keywords*), followed optionally by an equal sign and a variable to contain the statistic.

The new variables are named as follows: If you specify *keyword***=***name*, the new variable has the specified *name*. If you omit the optional **=***name* after a *keyword*, then the new variable name is formed by using a default character string that identifies the statistic. In either case, if you also specify the ROWWISE option after a slash and you specify more than one dependent variable or smoothing value in the MODEL statement, the variable name is appended with an order number. For details, see the ROWWISE option.

The *keywords* allowed and the statistics they represent are as follows:

PREDICTED | P creates a new variable that contains predicted values. The default *name* is Predicted.

RESIDUAL | R creates a new variable that contains residual values, which are calculated as ACTUAL – PREDICTED. The default *name* is Residual.

STD creates a new variable that contains standard errors of the mean predicted values. The use of this option implicitly selects the model option DFMETHOD=EXACT even if the DFMETHOD= option has not been explicitly used. The default *name* is StdErr.

T creates a new variable that contains t statistics. The use of this option implicitly selects the model option DFMETHOD=EXACT even if the DFMETHOD= option has not been explicitly used. The default *name* is tValue.

LCLM creates a new variable that contains the lower part of $100(1-\alpha)\%$ confidence limits on the mean predicted value. By default, the 95% limits are computed; the ALPHA= option in the MODEL statement can be used to change the significance level. The use of this option implicitly selects the model option DFMETHOD=EXACT even if the DFMETHOD= option has not been explicitly used. The default *name* is LowerCL.

UCLM creates a new variable that contains the upper part of $100(1-\alpha)\%$ confidence limits on the mean predicted value. By default, the 95% limits are computed; the ALPHA= option in the MODEL statement can be used to change the significance level. The use of this option implicitly selects the model option DFMETHOD=EXACT even if the DFMETHOD= option has not been explicitly used. The default *name* is UpperCL.

You can specify the following *options* in the OUTPUT statement after a slash (/).

ALL

requests all these *keywords*: PREDICTED, RESIDUAL, STD, T, LCLM, and UCLM.

ROWWISE | ROW

arranges the created OUTPUT data set in rowwise format. For each dependent variable and each smoothing value specified in the SMOOTH= option in the MODEL statement, one variable is generated for each specified *keyword* and the variable name is appended with an order number if there are multiple occurrences of the requested statistic. Those variables appear in an order that corresponds to the specified order of the dependent variables and the smoothing values in the MODEL statement. For each variable generated, a label is also created automatically; the label contains the default name of the represented statistic, the name of the dependent variable selected to be modeled, and the smoothing value used for calculating the represented statistic.

By default, the OUTPUT data set is created in columnwise format, where the input data is repeated for each dependent variable and for each smoothing value. Three extra columns, named SmoothingParameter for smoothing parameter values, DepVar for dependent variable names, and Obs for observation numbers, are also added to the OUTPUT data set to distinguish each model.

SCORE Statement

SCORE DATA=*SAS-data-set* < **ID=(***variable-list***)** > < / *options* > **;**

The fitted loess model is used to score the data in the specified SAS data set. This data set must contain all the regressor variables specified in the MODEL statement. Furthermore, when a BY statement is used, the score data set must also contain all the BY variables sorted in the order of the BY variables. A SCORE statement is optional, and more than one SCORE statement can be used. SCORE statements cannot be used if the DIRECT option is specified in the MODEL statement. The optional ID= (*variable-list*) specifies ID variables to be included in the "Score Results" table.

You find the results of the SCORE statement in the "Score Results" table. This table contains all the data in the data set named in the SCORE statement, including observations with missing values. However, only those observations with nonmissing regressor variables are scored. If no data set is named in the SCORE statement, the data set named in the PROC LOESS statement is scored. You use the PRINT option in the SCORE statement to request that the "Score Results" table be displayed. You can place the "Score Results" table in an output data set by using an ODS OUTPUT statement even if this table is not displayed.

You can specify the following *options* in the SCORE statement after a slash (/).

CLM

requests that $100(1-\alpha)\%$ confidence limits on the mean predicted value be added to the "Score Results" table. By default the 95% limits are computed; the ALPHA= option in the MODEL statement can be used to change the significance level. The use of this option implicitly selects the model option DFMETHOD=EXACT if the DFMETHOD= option has not been explicitly used.

PRINT < (VAR=*variables* **) >**

specifies that the "Score Results" table be displayed. By default only the variables named in the MODEL statement, the variables listed in the ID list in the SCORE statement, and the scored dependent variables are displayed. You can use the VAR= option to specify additional variables in the score data set that are to be included in the displayed output. Note, however, that all columns in the SCORE

data set are placed in the SCORE results table, even if you do not request that they be included in the displayed output.

RESIDUAL | R
requests that residuals be added to the "Score Results" table. If the data set you specify in DATA= option in the SCORE statement does not contain one or more of the model dependent variables, then the corresponding residual values in the "Score Results" table are set to missing.

SCALEDINDEP
specifies that scaled regressor coordinates be included in the "Score Results" table. This option is ignored if the SCALE= option is not specified in the MODEL statement.

STEPS
requests that all models evaluated during smoothing parameter value selection be scored, provided that the SELECT= option together with the STEPS modifier is specified in the MODEL statement. By default only the selected model is scored.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement specifies a variable in the input data set that contains values to be used as a priori weights for a loess fit.

The values of the weight variable must be nonnegative. If an observation's weight is zero, negative, or missing, the observation is deleted from the analysis.

Details: LOESS Procedure

Missing Values

PROC LOESS deletes any observation with missing values for any variable specified in the MODEL statement. This enables the procedure to reuse the k-d tree for all the dependent variables that appear in the MODEL statement. If you have multiple dependent variables with different missing value structures for the same set of independent variables, you might want to use separate PROC LOESS steps for each dependent variable.

Output Data Sets

PROC LOESS assigns a name to each table it creates. You can use the ODS OUTPUT statement to place one or more of these tables in output data sets. See the section "ODS Table Names" on page 5907 for a list of the table names created by PROC LOESS. For detailed information about ODS, see Chapter 23, "Using the Output Delivery System."

For example, the following statements create an output data set named MyOutStats containing the "Output Statistics" table and an output data set named MySummary containing the "Fit Summary" table.

```
proc loess data=Melanoma;
   model Incidences=Year;
   ods output OutputStatistics = MyOutStats
              FitSummary       = MySummary;
run;
```

Often, a single MODEL statement describes more than one model. For example, the following statements fit eight different models (four smoothing parameter values for each dependent variable).

```
proc loess;
   model y1 y2 = x1 x2 x3/smooth =0.1 to 0.7 by 0.2;
   ods output OutputStatistics = MyOutStats;
run;
```

The eight "Output Statistics" tables for these models are stacked in a single data set called MyOutStats. The data set contains a column named DepVarName and a column named SmoothingParameter that distinguish each model (see Figure 78.8 for an example). If you want the "Output Statistics" table for each model to be in its own data set, you can use the MATCH_ALL option in the ODS OUTPUT statement. The following statements create eight data sets named MyOutStats, MyOutStats1, . . . , MyOutStats7.

```
proc loess;
   model y1 y2 = x1 x2 x3/smooth =0.1 to 0.7 by 0.2;
   ods output OutputStatistics(match_all) = MyOutStats;
run;
```

For further options available in the ODS OUTPUT statement, see Chapter 23, "Using the Output Delivery System."

Only the "Scale Details" and "Fit Summary" tables are displayed by default. The other tables are optionally displayed by using the DETAILS option in the MODEL statement and the PRINT option in the SCORE statement. Note that it is not necessary to display a table in order for that table to be used in an ODS OUTPUT statement. For example, the following statements display the "Output Statistics" and "*k*-d Tree" tables but place the "Output Statistics" and "Prediction at Vertices" tables in output data sets.

```
proc loess data=Melanoma;
   model Incidences=Year/details(OutputStatistics kdTree);
   ods output OutputStatistics = MyOutStats
              PredAtVertices   = MyVerticesOut;
run;
```

Using the DETAILS option alone causes all tables to be displayed.

The MODEL statement options CLM, RESIDUAL, STD, SCALEDINDEP, and T control which optional columns are added to the OutputStatistics table. For example, to obtain an OutputStatistics output data set containing residuals and confidence limits in addition to the model variables and predicted value, you need to specify the RESIDUAL and CLM options in the MODEL statement as in the following example:

```
proc loess data=Melanoma;
   model Incidences=Year/residual clm;
   ods output OutputStatistics = MyOutStats;
run;
```

Finally, note that using the ALL option in the MODEL statement causes all optional columns to be included in the output. Also, ID columns can be added to the OutputStatistics table by using the ID statement.

Data Scaling

The loess algorithm to obtain a predicted value at a given point in the predictor space proceeds by doing a least squares fit that uses all data points close to the given point. Thus the algorithm depends critically on the metric used to define closeness. This has the consequence that if you have more than one predictor variable and these predictor variables have significantly different scales, then closeness depends almost entirely on the variable with the largest scaling. It also means that merely changing the units of one of your predictors can significantly change the loess model fit.

To circumvent this problem, it is necessary to standardize the scale of the independent variables in the loess model. The SCALE= option in the MODEL statement is provided for this purpose. PROC LOESS uses a symmetrically trimmed standard deviation as the scale estimate for each independent variable of the loess model. This is a robust scale estimator in that extreme values of a variable are discarded before estimating the data scaling. For example, to compute a 10% trimmed standard deviation of a sample, you discard the smallest and largest 5% of the data and compute the standard deviation of the remaining 90% of the data points. In this case, the trimming fraction is 0.1.

For example, the following statement specifies that the variables Temperature and Catalyst are scaled before performing the loess fitting. In this case, because the trimming fraction is 0.1, the scale estimate used for each of these variables is a 10% trimmed standard deviation.

```
model Yield=Temperature Catalyst / scale = SD(0.1);
```

The default trimming fraction used by PROC LOESS is 0.1 and need not be specified by the SCALE= option. Thus the following MODEL statement is equivalent to the previous MODEL statement.

```
model Yield=Temperature Catalyst / scale = SD;
```

If the SCALE= option is not specified, no scaling of the independent variables is done. This is appropriate when there is only a single independent variable or when all the independent variables are a priori scaled similarly.

When the SCALE= option is specified, the scaling details for each independent variable are added to the ScaleDetails table (see Output 78.3.2 for an example). By default, this table contains only the minimum and maximum values of each independent variable in the model. Finally, note that when the SCALE= option is used, specifying the SCALEDINDEP option in the MODEL statement adds the scaled values of

the independent variables to the OutputStatistics and PredAtVertices tables. If the SCALEDINDEP option is specified in the SCORE statement, then scaled values of the independent variables are included in the ScoreResults table. By default, only the unscaled values are placed in these tables.

Direct versus Interpolated Fitting

Local regression to obtain a predicted value at a given point in the predictor space is done by doing a least squares fit that uses all data points in a local neighborhood of the given point. This method is computationally expensive because a local neighborhood must be determined and a least squares problem must be solved for each point at which a fitted value is required. A faster method is to obtain such fits at a representative sample of points in the predictor space and to obtain fitted values at all other points by interpolation.

PROC LOESS can fit models by using either of these two methods. By default, PROC LOESS uses fitting at a sample of points and interpolation. The method fitting a local model at every data point is selected by specifying the DIRECT option in the MODEL statement.

k-d Trees and Blending

PROC LOESS uses a *k*-d tree to divide the box (also called the *initial cell* or *bucket*) enclosing all the predictor data points into rectangular cells. The vertices of these cells are the points at which local least squares fitting is done.

Starting from the initial cell, the direction of the longest cell edge is selected as the split direction. The median of this coordinate of the data in the cell is the split value. The data in the starting cell are partitioned into two child cells. The left child consists of all data from the parent cell whose coordinate in the split direction is less than the split value. This procedure is repeated for each child cell that has more than a prespecified number of points, called the *bucket size* of the *k*-d tree.

You can specify the bucket size with the BUCKET= option in the MODEL statement. If you do not specify the BUCKET= option, the default value used is the largest integer less than or equal to $ns/5$, where n is the number of observations and s is the value of the smoothing parameter. Note that if fitting is being done for a range of smoothing parameter values, the bucket size can change for each value.

The set of vertices of all the cells of the *k*-d tree are the points at which PROC LOESS performs its local fitting. The fitted value at an original data point (or at any other point within the original data cell) is obtained by blending the fitted values at the vertices of the *k*-d tree cell that contains that data point.

The univariate blending methods available in PROC LOESS are linear and cubic polynomial interpolation, with linear interpolation being the default. You can request cubic interpolation by specifying the INTERP=CUBIC option in the MODEL statement. In this case, PROC LOESS uses the unique cubic polynomial whose values and first derivatives match those of the fitted local polynomials evaluated at the two endpoints of the *k*-d tree cell edge.

In the multivariate case, such univariate interpolating polynomials are computed on each edge of the *k*-d tree cells and are combined using blending functions (Gordon 1971). In the case of two regressors, if you specify INTERP=CUBIC in the MODEL statement, PROC LOESS uses Hermite cubic polynomials as blending functions. If you do not specify INTERP=CUBIC, or if you specify a model with more than two regressors, then PROC LOESS uses linear polynomials as blending functions. In these cases, the blending method

reduces to tensor product interpolation from the 2^p vertices of each k-d tree cell, where p is the number of regressors.

While the details of the k-d tree and the fitted values at the vertices of the k-d tree are implementation details that seldom need to be examined, PROC LOESS does provide options for their display. Each k-d tree subdivision of the data used by PROC LOESS is placed in the kdTree table. The predicted values at the vertices of each k-d tree are placed in the PredAtVertices table. You can request these tables by using the DETAILS option in the MODEL statement.

Local Weighting

The size of the local neighborhoods that PROC LOESS uses in performing local fitting is determined by the smoothing parameter value s. When $s < 1$, the local neighborhood used at a point x_i contains the s fraction of the data points closest to the point x_i. When $s \geq 1$, all data points are used.

Suppose q denotes the number of points in the local neighborhoods and $d_1, d_2, \ldots, d_q$ denote the distances in increasing order of the q points closest to x_i. The point at distance d_i from x_i is given a weight w_i in the local regression that decreases as the distance from x_i increases. PROC LOESS uses a tricube weight function to define

$$w_i = \frac{32}{5}\left(1 - \left(\frac{d_i}{d_q}\right)^3\right)^3$$

If $s > 1$, then d_q is replaced by $d_q s^{1/p}$ in the previous formula, where p is the number of predictors in the model.

Finally, note that if a weight variable has been specified using a WEIGHT statement, then w_i is multiplied by the corresponding value of the specified weight variable.

Iterative Reweighting

PROC LOESS can do iterative reweighting to improve the robustness of the fit in the presence of outliers in the data. Iterative reweighting is also appropriate when statistical inference is requested and the error distribution is symmetric but not Gaussian.

The number of iterations is specified by the ITERATIONS= option in the MODEL statement. The default is ITERATIONS=1, which corresponds to no reweighting.

At iterations beyond the first iteration, the local weights w_i of the previous section are replaced by $r_i w_i$, where r_i is a weight that decreases as the residual of the fitted value at the previous iteration at the point corresponding to d_i increases. See Cleveland and Grosse (1991) and Cleveland, Grosse, and Shyu (1992) for details.

Specifying the Local Polynomials

PROC LOESS uses linear or quadratic polynomials in doing the local least squares fitting. The option DEGREE = in the MODEL statement is used to specify the degree of the local polynomials used by PROC LOESS, with DEGREE = 1 being the default. In addition, when DEGREE = 2 is specified, the MODEL statement DROPSQUARE= option can be used to exclude specific monomials during the least squares fitting.

For example, the following statements use the monomials 1, x1, x2, x1*x2, and x2*x2 for the local least squares fitting.

```
proc loess;
   model y= x1 x2/ degree=2 dropsquare=(x1);
run;
```

Smoothing Matrix

When no iterative reweighting is done, the "Smoothing Matrix" denoted by $\mathbf{L}$ defines the linear relationship between the fitted and observed dependent variable values of a loess model. You can obtain the predicted values of a loess fit from the observed values via

$$\hat{\mathbf{y}} = \mathbf{L}\mathbf{y}$$

where $\mathbf{y}$ is the vector of observed values and $\hat{\mathbf{y}}$ is the corresponding vector of predicted values of the dependent variable. Note that $\mathbf{L}$ is an n by n matrix, where n is the number of observations in the analysis. PROC LOESS does not explicitly form $\mathbf{L}$ if the DFMETHOD=EXACT option is not explicitly or implicitly selected.

Model Degrees of Freedom

The approximate model degrees of freedom in a nonparametric fit is a number that is analogous to the number of free parameters in a parametric model. There are three commonly used measures of model degrees of freedom in nonparametric models. These criteria are as follows:

$$\begin{aligned}
\mathrm{DF1} &\equiv \mathrm{Trace}(\mathbf{L}) \\
\mathrm{DF2} &\equiv \mathrm{Trace}(\mathbf{L}'\mathbf{L}) \\
\mathrm{DF3} &\equiv 2\mathrm{Trace}\mathbf{L} - \mathrm{Trace}(\mathbf{L}'\mathbf{L})
\end{aligned}$$

A discussion of their properties can be found in Hastie and Tibshirani (1990). DF2 is also referred to as the "Equivalent Number of Parameters," and this is the name that PROC LOESS uses for DF2 when it appears in the "Fit Summary" table.

Statistical Inference and Lookup Degrees of Freedom

If you denote the ith measurement of the response by y_i and the corresponding measurement of predictors by x_i, then

$$y_i = g(x_i) + \epsilon_i$$

where g is the regression function and ϵ_i are independent random errors with mean zero. If the errors are normally distributed with constant variance, then you can obtain confidence intervals for the predictions from PROC LOESS. You can also obtain confidence limits in the case where ϵ_i is heteroscedastic but $a_i\epsilon_i$ has constant variance and a_i are a priori weights that are specified using the WEIGHT statement of PROC LOESS. You can do inference in the case in which the error distribution is symmetric by using iterative reweighting. Formulas for doing statistical inference under the preceding conditions can be found in Cleveland and Grosse (1991) and Cleveland, Grosse, and Shyu (1992). Cleveland and Grosse (1991) show that standardized residuals for a loess model follow a t distribution with ρ degrees of freedom where

$$\begin{aligned}
\delta_1 &\equiv \text{Trace}(\mathbf{I}-\mathbf{L})'(\mathbf{I}-\mathbf{L}) \\
\delta_2 &\equiv \text{Trace}\left((\mathbf{I}-\mathbf{L})'(\mathbf{I}-\mathbf{L})\right)^2 \\
\rho &\equiv \text{Lookup Degrees of Freedom} \\
&\equiv \delta_1^2/\delta_2
\end{aligned}$$

The residual standard error that you find in the "Fit Summary" table is defined by

$$\text{Residual Standard Error} \equiv \sqrt{\text{Residual SS}/\delta_1}$$

The determination of ρ is computationally expensive and is not done by default. It is computed if you specify the DFMETHOD=EXACT or DFMETHOD=APPROX option in the MODEL statement. It is also computed if you specify any of the options CLM, STD, and T in the MODEL statement. Note that the values of δ_1, δ_2, and ρ are reported in the "Fit Summary" table.

If you specify the CLM option in the MODEL statement, confidence limits are added to the OutputStatistics table. By default, 95% limits are computed, but you can change this by using the ALPHA= option in the MODEL statement.

Automatic Smoothing Parameter Selection

There are several methodologies for automatic smoothing parameter selection. One class of methods chooses the smoothing parameter value to minimize a criterion that incorporates both the tightness of the fit and model complexity. Such a criterion can usually be written as a function of the error mean square, $\hat{\sigma}^2$, and a penalty function designed to decrease with increasing smoothness of the fit. This penalty function is usually defined in terms of the smoothing matrix $\mathbf{L}$ (see the section "Smoothing Matrix" on page 5901).

Examples of specific criteria are generalized cross validation (Craven and Wahba 1979) and the Akaike information criterion (Akaike 1973). These classical selectors have two undesirable properties when used with local polynomial and kernel estimators: they tend to undersmooth small data sets and tend to be nonrobust in the sense that small variations of the input data can change the choice of smoothing parameter value significantly. Hurvich, Simonoff, and Tsai (1998) obtained several corrected AIC criteria that address the small-sample bias and perform comparably with the *plug-in selectors* (Ruppert, Sheather, and Wand 1995). PROC LOESS provides automatic smoothing parameter selection that uses two of these corrected AIC criteria, named AIC_{C_1} and AIC_C in Hurvich, Simonoff, and Tsai (1998), and generalized cross validation, denoted by GCV.

The relevant formulas are

$$
\begin{aligned}
\mathrm{AIC}_{C_1} &= n\log(\hat{\sigma}^2) + n\frac{\delta_1/\delta_2(n+\nu_1)}{\delta_1^2/\delta_2 - 2} \\
\mathrm{AIC}_C &= \log(\hat{\sigma}^2) + 1 + \frac{2\,(\mathrm{Trace}(\mathbf{L})+1)}{n - \mathrm{Trace}(\mathbf{L}) - 2} \\
\mathrm{GCV} &= \frac{n\hat{\sigma}^2}{(n - \mathrm{Trace}(\mathbf{L}))^2}
\end{aligned}
$$

where n is the number of observations and

$$
\begin{aligned}
\delta_1 &\equiv \mathrm{Trace}(\mathbf{I}-\mathbf{L})'(\mathbf{I}-\mathbf{L}) \\
\delta_2 &\equiv \mathrm{Trace}\left((\mathbf{I}-\mathbf{L})'(\mathbf{I}-\mathbf{L})\right)^2 \\
\nu_1 &\equiv \text{Equivalent Number of Parameters} \\
&\equiv \mathrm{Trace}(\mathbf{L}'\mathbf{L})
\end{aligned}
$$

You invoke these methods for automatic smoothing parameter selection by specifying the SELECT=*criterion* option in the MODEL statement, where *criterion* is AICC1, AICC, or GCV. The LOESS procedure evaluates the specified criterion for a sequence of smoothing parameter values and selects the value in this sequence that minimizes the specified criterion. If multiple values yield the optimum, then the largest of these values is selected.

A second class of methods seeks to set an approximate measure of model degrees of freedom to a specified target value. These methods are useful for making meaningful comparisons between loess fits and other

nonparametric and parametric fits. Three approximate model degrees of freedom for a loess model are defined in the section "Model Degrees of Freedom" on page 5901. You invoke these methods by specifying the SELECT=*DFCriterion*(*target*) option in the MODEL statement, where *DFCriterion* is DF1, DF2, or DF3. The criterion that is minimized is given in the following table.

Table 78.4 Minimization Criteria

Syntax	Minimization Criterion
SELECT=DF1(*target*)	$\lvert \mathrm{Trace}(\mathbf{L}) - \mathit{target} \rvert$
SELECT=DF2(*target*)	$\lvert \mathrm{Trace}(\mathbf{L}'\mathbf{L}) - \mathit{target} \rvert$
SELECT=DF3(*target*)	$\lvert 2\mathrm{Trace}(\mathbf{L}) - \mathrm{Trace}(\mathbf{L}'\mathbf{L}) - \mathit{target} \rvert$

The results are summarized in the "Smoothing Criterion" table. This table is displayed whenever automatic smoothing parameter selection is performed. You can obtain details of the sequence of models examined by specifying the DETAILS(MODELSUMMARY) option in the MODEL statement to display the "Model Summary" table.

There are several ways in which you can control the sequence of models examined by PROC LOESS. If you specify the SMOOTH=*value-list* option in the MODEL statement, then only the values in this list are examined in performing the selection. For example, the following statements select the model that minimizes the AICC1 criterion among the three models with smoothing parameter values 0.1, 0.3, and 0.4:

```
proc loess;
   model y= x1/ smooth=0.1 0.3 0.4 select=AICC1;
run;
```

If you do not specify the SMOOTH= option in the MODEL statement, then by default PROC LOESS uses a golden section search method to find a local minimum of the specified criterion in the range (0, 1]. You can use the RANGE(*lower*,*upper*) modifier in the SELECT= option to change the interval in which the golden section search is performed. For example, the following statements request a golden section search to find a local minimizer of the GCV criterion for smoothing parameter values in the interval [0.1,0.5]:

```
proc loess;
   model y= x1/select=GCV( range(0.1,0.5) );
run;
```

If you want to be sure of obtaining a global minimum in the range of smoothing parameter values examined, you can specify the GLOBAL modifier in the SELECT= option. For example, the following statements request that a global minimizer of the AICC criterion be obtained for smoothing parameter values in the interval [0.2, 0.8]:

```
proc loess;
   model y= x1/select=AICC( global range(0.2,0.8) );
run;
```

Note that even though the smoothing parameter is a continuous variable, a given range of smoothing parameter values corresponds to a finite set of local models. For example, for a data set with 100 observations, the range

$[0.2, 0.4]$ corresponds to models with $20, 21, 22, \ldots, 40$ points in the local neighborhoods. If the GLOBAL modifier is specified, all possible models in the range are evaluated sequentially.

Note that by default PROC LOESS displays a "Fit Summary" and other optionally requested tables only for the selected model. You can request that these tables be displayed for all models in the selection process by adding the STEPS modifier in the SELECT= option. Also note that by default scoring requested with SCORE statements is done only for the selected model. However, if you specify the STEPS in both the MODEL and SCORE statements, then all models evaluated in the selection process are scored.

In terms of computation, AIC_C, GCV, and DF1 depend on the smoothing matrix $\mathbf{L}$ only through its trace. In the direct method, this trace can be computed efficiently. In the interpolated method that uses k-d trees, there is some additional computational cost but the overall work is not significant compared to the rest of the computation. In contrast, the quantities δ_1, δ_2, and ν_1 that appear in the AIC_{C_1} criterion, and the DF2 and DF3 criteria, depend on the entire $\mathbf{L}$ matrix and for this reason, the time needed to compute these quantities dominates the time required for the model fitting. Hence SELECT=AICC1, SELECT=DF2, and SELECT=DF3 are much more computationally expensive than SELECT=AICC, SELECT=GCV, and SELECT=DF1, especially when combined with the GLOBAL modifier. Hurvich, Simonoff, and Tsai (1998) note that AIC_C can be regarded as an approximation of AIC_{C_1} and that "the AIC_C selector generally performs well in all circumstances."

For models with one dependent variable, PROC LOESS uses SELECT=AICC as its default, if you specify neither the SMOOTH= nor the SELECT= option in the MODEL statement. With two or more dependent variables, automatic smoothing parameter selection needs to be done separately for each dependent variable. For this reason automatic smoothing parameter selection is not available for models with multiple dependent variables. In such cases you should use a separate PROC LOESS step for each dependent variable, if you want to use automatic smoothing parameter selection.

Sparse and Approximate Degrees of Freedom Computation

As noted in the section "Statistical Inference and Lookup Degrees of Freedom" on page 5902, obtaining confidence limits in loess models requires the computation of the lookup degrees of freedom. This in turn requires the computation of

$$\delta_2 \equiv \mathrm{Trace}\left((\mathbf{I}-\mathbf{L})'(\mathbf{I}-\mathbf{L})\right)^2$$

where $\mathbf{L}$ is the loess smoothing matrix (see the section "Smoothing Matrix" on page 5901).

The work in a direct implementation of this formula grows as n^3, where n is the number of observations in analysis. For large n, this work dominates the time needed to fit the loess model itself. To alleviate this computational bottleneck, Cleveland and Grosse (1991) and Cleveland, Grosse, and Shyu (1992) developed approximate methods for estimating this quantity in terms of more readily computable statistics. A different approach to obtaining a computationally cheap estimate of δ_2 has been implemented in PROC LOESS.

For large data sets with significant local structure, the loess model is often used with small values of the smoothing parameter. Recalling that the smoothing parameter defines the fraction of the data used in each local regression, this means that the loess fit at any point in regressor space depends on only a small fraction of the data. This is reflected in the smoothing matrix $\mathbf{L}$ whose (i, j) entry is nonzero only if the ith and jth observations lie in at least one common local neighborhood. Hence the smoothing matrix is a sparse matrix

(has mostly zero entries) in such cases. By exploiting this sparsity, PROC LOESS now computes δ_2 orders of magnitude faster than in previous implementations.

When each local neighborhood contains a large subset of the data—i.e., when the smoothing parameter is large—then it is no longer true that the smoothing matrix is sparse. However, since a point in a local neighborhood is given a local weight that decreases with its distance from the center of the neighborhood, many of the coefficients in the smoothing matrix turn out to be nonzero but with orders of magnitude smaller than that of the larger coefficients in the matrix. The approximate method for computing δ_2 that has been implemented in PROC LOESS exploits these disparities in magnitudes of the elements in the smoothing matrix by setting the small elements to zero. This creates a sparse approximation of the smoothing matrix to which the fast sparse methods can be applied.

In order to decide the threshold at which elements in the smoothing matrix are set to zero, PROC LOESS samples the elements in the smoothing matrix to obtain the value of the element in a specified lower quantile in this sample. The magnitude of the element at this quantile is used as a cutoff value, and all elements in the smoothing matrix whose magnitude is less than this cutoff are set to zero for the approximate computation. By default all elements in the lower ninetieth percentile are set to zero. You can use the DFMETHOD=APPROX(QUANTILE=) option in the MODEL statement to change this value. As you increase the value for the quantile to be zeroed, you speed up the degrees of freedom computation at the expense of increasing approximation errors. You can also use the DFMETHOD=APPROX(CUTOFF=) option in the MODEL statement to specify the cutoff value directly.

For small data sets, the approximate computation is not needed and would be rougher than for larger data sets. Hence PROC LOESS performs the exact computation for analyses with fewer than 500 points, even if DFMETHOD=APPROX is specified in the model statement. Also, for small values of the smoothing parameter, elements in the lower specified quantile might already all be zero. In such cases the approximate method is the same as the exact method. PROC LOESS labels as approximate any statistics that depend on the approximate computation of δ_2 only in the cases where the approximate computation was used and is different from the exact computation.

Scoring Data Sets

One or more SCORE statements can be used with PROC LOESS. A data set that includes all the variables specified in the MODEL and BY statements must be specified in each SCORE statement. Score results are placed in the ScoreResults table. This table is not displayed by default, but specifying the PRINT option in the SCORE statement produces the table. If you specify the CLM option in the SCORE statement, confidence intervals are included in the ScoreResults table.

Note that scoring is not supported when the DIRECT option is specified in the MODEL statement. Scoring at a point specified in a score data set is done by first finding the cell in the k-d tree containing this point and then interpolating the scored value from the predicted values at the vertices of this cell. This methodology precludes scoring any points that are not contained in the box that surrounds the data used in fitting the loess model.

ODS Table Names

PROC LOESS assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 78.5 ODS Tables Produced by PROC LOESS

ODS Table Name	Description	Statement	Option
FitSummary	Specified fit parameters and fit summary		Default
kdTree	Structure of *k*-d tree used	MODEL	DETAILS(kdTree)
ModelSummary	Summary of all models evaluated	MODEL	DETAILS(ModelSummary)
OutputStatistics	Coordinates and fit results at input data points	MODEL	DETAILS(OutputStatistics)
PredAtVertices	Coordinates and fitted values at *k*-d tree vertices	MODEL	DETAILS(PredAtVertices)
ScaleDetails	Extent and scaling of the independent variables		Default
ScoreResults	Coordinates and fit results at scoring points	SCORE	PRINT
SmoothingCriterion	Criterion value and selected smoothing parameter	MODEL	SELECT

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC LOESS generates are listed in Table 78.6, along with the relevant PLOTS= options.

Table 78.6 Graphs Produced by PROC LOESS

ODS Graph Name	Plot Description	PLOTS Option
ContourFitPanel	Panel of loess contour surfaces overlaid on scatter plots of data	CONTOURFITPANEL
ContourFit	Loess contour surface overlaid on scatter plot of data	CONTOURFITPANEL
CriterionPlot	Selection criterion versus smoothing parameter	CRITERION
DiagnosticsPanel	Panel of fit diagnostics	DIAGNOSTICS
FitPanel	Panel of loess curves overlaid on scatter plots of data	FITPANEL
FitPlot	Loess curve overlaid on scatter plot of data	FIT
ObservedByPredicted	Dependent variable versus loess fit	OBSERVEDBYPREDICTED
QQPlot	Normal quantile plot of residuals	QQPLOT
ResidualsBySmooth	Panel of residuals versus regressor by smoothing parameter values	RESIDUALSBYSMOOTH
ResidualByPredicted	Residuals versus loess fit	RESIDUALBYPREDICTED
ResidualHistogram	Histogram of fit residuals	RESIDUALHISTOGRAM
ResidualPanel	Panel of residuals versus regressors for fixed smoothing parameter value	RESIDUALS
ResidualPlot	Plot of residuals versus regressor	RESIDUALS
RFPlot	Side-by-side plots of quantiles of centered fit and residuals	RFPLOT
ScorePlot	Loess fit evaluated at scoring points	SCOREPLOT

Examples: LOESS Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 78.1: Engine Exhaust Emissions

(View the complete code for this example (loessex1.sas) in the example repository.)

Investigators studied the exhaust emissions of a one-cylinder engine (Brinkman 1981). The SAS data set Gas contains the results data. The dependent variable, NOx, measures the concentration, in micrograms per joule, of nitric oxide and nitrogen dioxide normalized by the amount of work of the engine. The independent variable, E, is a measure of the richness of the air and fuel mixture.

```
data Gas;
   input NOx E @@;
   format NOx f3.1;
   format E f3.1;
   datalines;
4.818  0.831   2.849  1.045
3.275  1.021   4.691  0.97
4.255  0.825   5.064  0.891
2.118  0.71    4.602  0.801
2.286  1.074   0.97   1.148
3.965  1       5.344  0.928
3.834  0.767   1.99   0.701
5.199  0.807   5.283  0.902
3.752  0.997   0.537  1.224
1.64   1.089   5.055  0.973
4.937  0.98    1.561  0.665
;
```

The following PROC SGPLOT statements produce the simple scatter plot of these data displayed in Output 78.1.1.

```
proc sgplot data=Gas;
   scatter x=E y=NOx;
run;
```

Output 78.1.1 Scatter Plot of the Gas Data

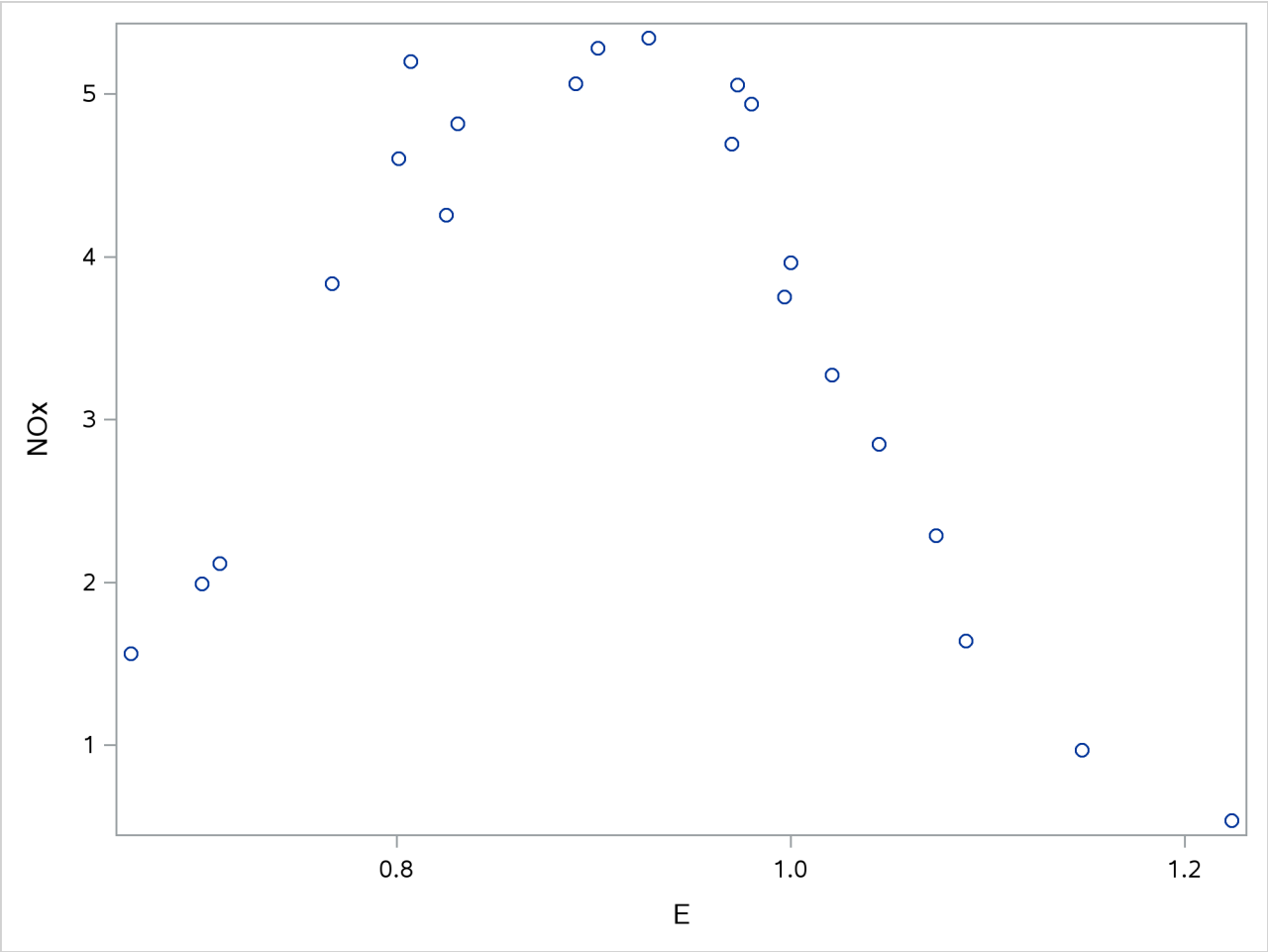

The following statements fit two loess models for these data. Because this is a small data set, it is reasonable to do direct fitting at every data point. As there is substantial curvature in the data, quadratic local polynomials are used. An ODS OUTPUT statement creates two output data sets containing the "Output Statistics" and "Fit Summary" tables.

```
ods graphics on;

proc loess data=Gas;
   ods output OutputStatistics = GasFit
              FitSummary=Summary;
   model NOx = E / degree=2 select=AICC(steps) smooth = 0.6 1.0
                   direct alpha=.01 all details;
run;

ods graphics off;
```

Output 78.1.2 Fit Summary Table

The LOESS Procedure
Selected Smoothing Parameter: 0.6
Dependent Variable: NOx

Fit Summary	
Fit Method	Direct
Number of Observations	22
Degree of Local Polynomials	2
Smoothing Parameter	0.60000
Points in Local Neighborhood	13
Residual Sum of Squares	1.71852
Trace[L]	6.42184
GCV	0.00708
AICC	-0.45637
AICC1	-9.39715
Delta1	15.12582
Delta2	14.73089
Equivalent Number of Parameters	5.96950
Lookup Degrees of Freedom	15.53133
Residual Standard Error	0.33707

The "Fit Summary" table for smoothing parameter value 0.6, shown in Output 78.1.2, records the fitting parameters specified and some overall fit statistics. See the section "Smoothing Matrix" on page 5901 for a definition of the smoothing matrix **L**, and the sections "Model Degrees of Freedom" on page 5901 and "Statistical Inference and Lookup Degrees of Freedom" on page 5902 for definitions of the statistics that appear this table.

The "Output Statistics" table for smoothing parameter value 0.6 is shown in Output 78.1.3. Note that, because the ALL option is specified in the MODEL statement, this table includes all the relevant optional columns. Furthermore, because the ALPHA=0.01 option is specified in the MODEL statement, the confidence limits in this table are 99% limits.

Output 78.1.3 Output Statistics Table

The LOESS Procedure
Selected Smoothing Parameter: 0.6
Dependent Variable: NOx

			Output Statistics					
Obs	E	NOx	Predicted NOx	Estimated Prediction Std Deviation	Residual	t Value	99% Confidence Limits	
1	0.8	4.8	4.87377	0.15528	-0.05577	-0.36	4.41841	5.32912
2	1.0	2.8	2.81984	0.15380	0.02916	0.19	2.36883	3.27085
3	1.0	3.3	3.48153	0.15187	-0.20653	-1.36	3.03617	3.92689
4	1.0	4.7	4.73249	0.13923	-0.04149	-0.30	4.32419	5.14079
5	0.8	4.3	4.82305	0.15278	-0.56805	-3.72	4.37503	5.27107
6	0.9	5.1	5.18561	0.19337	-0.12161	-0.63	4.61855	5.75266
7	0.7	2.1	2.51120	0.15528	-0.39320	-2.53	2.05585	2.96655
8	0.8	4.6	4.48267	0.15285	0.11933	0.78	4.03444	4.93089
9	1.1	2.3	2.12619	0.16683	0.15981	0.96	1.63697	2.61541
10	1.1	1.0	0.97120	0.18134	-0.00120	-0.01	0.43942	1.50298
11	1.0	4.0	4.09987	0.13477	-0.13487	-1.00	3.70467	4.49507
12	0.9	5.3	5.31258	0.17283	0.03142	0.18	4.80576	5.81940
13	0.8	3.8	3.84572	0.14929	-0.01172	-0.08	3.40794	4.28350
14	0.7	2.0	2.26578	0.16712	-0.27578	-1.65	1.77571	2.75584
15	0.8	5.2	4.58394	0.15363	0.61506	4.00	4.13342	5.03445
16	0.9	5.3	5.24741	0.19319	0.03559	0.18	4.68089	5.81393
17	1.0	3.8	4.16979	0.13478	-0.41779	-3.10	3.77457	4.56502
18	1.2	0.5	0.53059	0.32170	0.00641	0.02	-0.41278	1.47397
19	1.1	1.6	1.83157	0.17127	-0.19157	-1.12	1.32933	2.33380
20	1.0	5.1	4.66733	0.13735	0.38767	2.82	4.26456	5.07010
21	1.0	4.9	4.52385	0.13556	0.41315	3.05	4.12632	4.92139
22	0.7	1.6	1.19888	0.26774	0.36212	1.35	0.41375	1.98401

Output 78.1.4 Output Statistics Table

Optimal Smoothing Criterion	
AICC	Smoothing Parameter
-0.45637	0.60000

The combination of the options SELECT=AICC and SMOOTH=0.6 1 in the MODEL statement specifies that PROC LOESS fit models with smoothing parameters of 0.6 and 1 and select the model that yields the smaller value of the AICC statistic. The "Smoothing Criterion" shown in Output 78.1.4 shows that PROC LOESS selects the model with smoothing parameter value 0.6 as it yields the smaller value of the AICC statistic.

With ODS Graphics enabled, PROC LOESS produces a panel of fit plots whenever you specify the SMOOTH= option in the MODEL statement. These fit plots include confidence limits if you additionally specify the CLM option in the MODEL statement.

Output 78.1.5 Loess Fits with 99% Confidence Limits for the Gas Data

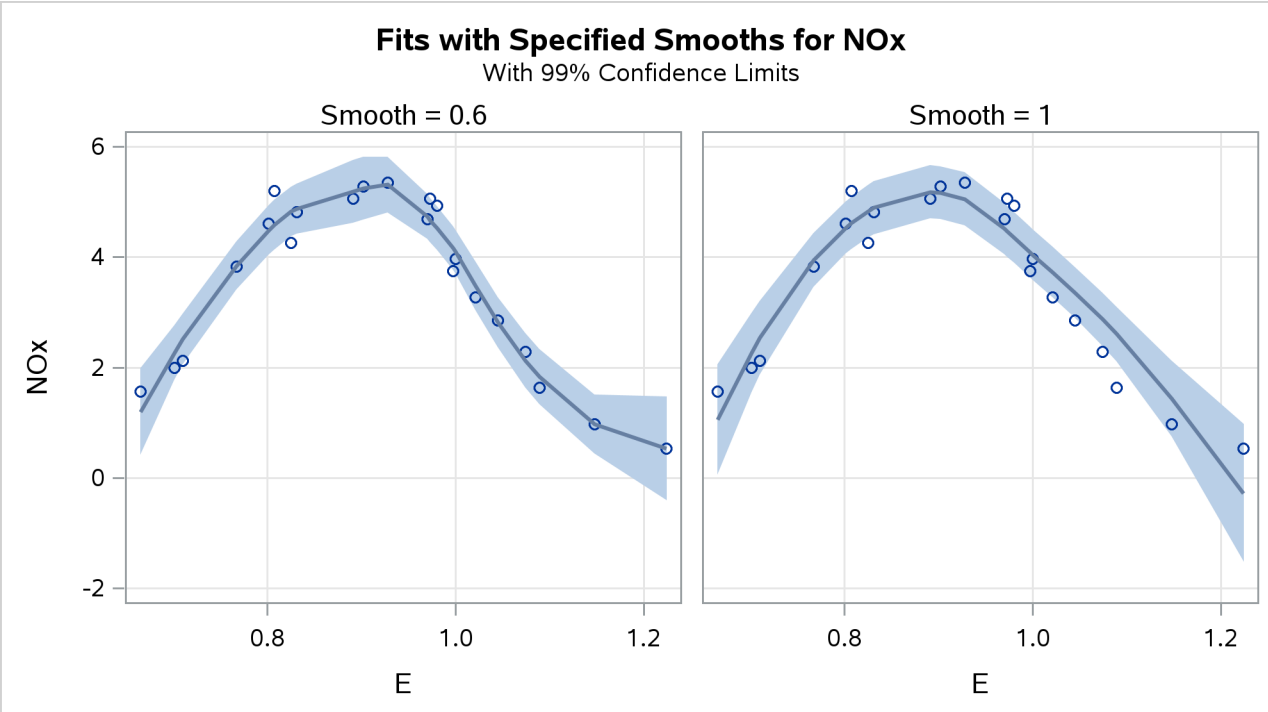

Output 78.1.5 shows the "Fit Panel" that displays the fitted models with 99% confidence limits overlaid on scatter plots of the data.

Based on the AICC criterion, the model with smoothing parameter 0.6 is preferred. You can address the question of whether the differences between these models are significant using analysis of variance. You do this by using the model with smoothing parameter value 1 as the null model.

The statistic

$$F = \frac{(\mathrm{rss}^{(n)} - \mathrm{rss})/(\delta_1^{(n)} - \delta_1)}{\mathrm{rss}/\delta_1}$$

has a distribution that is well approximated by an F distribution with

$$\nu = \frac{(\delta_1^{(n)} - \delta_1)^2}{\delta_2^{(n)} - \delta_2}$$

numerator degrees of freedom and ρ denominator degrees of freedom (Cleveland and Grosse 1991). Here quantities with superscript n refer to the null model, rss is the residual sum of squares, and δ_1, δ_2, and ρ are defined in the section "Statistical Inference and Lookup Degrees of Freedom" on page 5902.

The "Fit Summary" tables contain the information needed to carry out such an analysis. These tables have been captured in the output data set named Summary by using an ODS OUTPUT statement. The following statements extract the relevant information from this data set and carry out the analysis of variance:

```
data h0 h1;
   set Summary(keep=SmoothingParameter Label1 nValue1
               where=(Label1 in ('Residual Sum of Squares','Delta1',
                      'Delta2','Lookup Degrees of Freedom')));
   if SmoothingParameter = 1 then output h0;
   else output h1;
run;

proc transpose data=h0(drop=SmoothingParameter Label1) out=h0;
run;

data h0(drop=_NAME_);
   set h0;
   rename Col1 = RSSNull
          Col2 = delta1Null
          Col3 = delta2Null;
run;

proc transpose data=h1(drop=SmoothingParameter Label1) out=h1;
run;

data h1(drop=_NAME_);
   set h1;
   rename Col1 = RSS     Col2 = delta1
          Col3 = delta2  Col4 = rho;
run;

data ftest;
   merge h0 h1;
   nu = (delta1Null - delta1)**2 / (delta2Null - delta2);
   Numerator = (RSSNull - RSS)/(delta1Null - delta1);
   Denominator = RSS/delta1;
   FValue = Numerator / Denominator;
   PValue = 1 - ProbF(FValue, nu, rho);
   label nu     = 'Num DF'   rho    = 'Den DF'
         FValue = 'F Value'  PValue = 'Pr > F';
run;

proc print data=ftest label;
   var nu rho Numerator Denominator FValue PValue;
   format nu rho FValue 7.2 PValue 6.4;
run;
```

The results are shown in Output 78.1.6.

Output 78.1.6 Test ANOVA for Loess Models of Gas Data

Obs	Num DF	Den DF	Numerator	Denominator	F Value	Pr > F
1	2.67	15.53	1.05946	0.11362	9.32	0.0012

The small p-value confirms that the fit with smoothing parameter value 0.6 is significantly different from the loess model with smoothing parameter value 1.

Alternatively, you can use the OUTPUT statement to generate the statistics you want to include in the output data set. The following statements produce essentially the same results as the ODS OUTPUT statement does, except all the statistics for each of the two smoothing parameter values are included because the SELECT= option is not specified in the MODEL statement. In addition, with the ROW option specified, the output data set is arranged in rowwise format which enables you to compare statistics side-by-side for a sequence of smoothing values. The ALL option after the slash produces all the statistics (predicted values, residual values, standard errors of the mean predicted values, t statistics, and the lower and upper parts of $100(1-\alpha)\%$ confidence limits on the mean predicted value). All these requested statistics are given their respective default names in the output data set except the predicted value. The P=PREDVAL option causes the name for the predicted value to start with predval.

```
proc loess data=Gas;
   model NOx = E / degree=2 smooth = 0.6 1.0
                   direct alpha=.01;
   output out=GasFit p=predval /all row;
run;
```

Example 78.2: Sulfate Deposits in the U.S. for 1990

(View the complete code for this example (loessex2.sas) in the example repository.)

The following data set contains measurements in grams per square meter of sulfate (SO_4) deposits during 1990 at 179 sites throughout the 48 contiguous states.

```
data SO4;
   input Latitude Longitude SO4 @@;
   format Latitude f4.0;
   format Longitude f4.0;
   format SO4 f4.1;
   datalines;
32.45833  87.24222 1.403 34.28778  85.96889 2.103
33.07139 109.86472 0.299 36.07167 112.15500 0.304
31.95056 112.80000 0.263 33.60500  92.09722 1.950
34.17944  93.09861 2.168 36.08389  92.58694 1.578

   ... more lines ...

43.87333 104.19222 0.306 44.91722 110.42028 0.210
45.07611  72.67556 2.646
;
```

As longitudes decrease from west to east in the western hemisphere, the roles of east and west get interchanged if you use these longitudes on the horizontal axis of a plot. You can address this by using negative values to represent longitudes in the western hemisphere. The following statements change the sign of longitude in the SO4 data set and define a format to display these negative values with a suffix of “W”.

```
proc format;
   picture latitude  -90 - 0  = '000S'
                      0  - 90 = '000N';
   picture longitude -180  - 0    = '000W'
                      0    - 180  = '000E';
```

```
run;
data SO4;
   set SO4;
   format longitude longitude. latitude  latitude.;
   longitude = -longitude;
run;
```

The following statements use ODS Graphics to plot the locations of the sulfate measurements. The circles indicating the locations are colored using a gradient that denotes the value of SO4.

```
proc template;
   define statgraph gradientScatter;
      beginGraph;
         layout overlay;
            scatterPlot x=longitude y=latitude /
               markercolorgradient = SO4
               markerattrs         = (symbol=circleFilled)
               colormodel          =  ThreeColorRamp
               name                =  "Scatter";
            scatterPlot x=longitude y=latitude /
               markerattrs         = (symbol=circle);
            continuousLegend "Scatter"/title= "SO4";
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=SO4 template=gradientScatter;
run;
```

Output 78.2.1 Sulfate Measurements

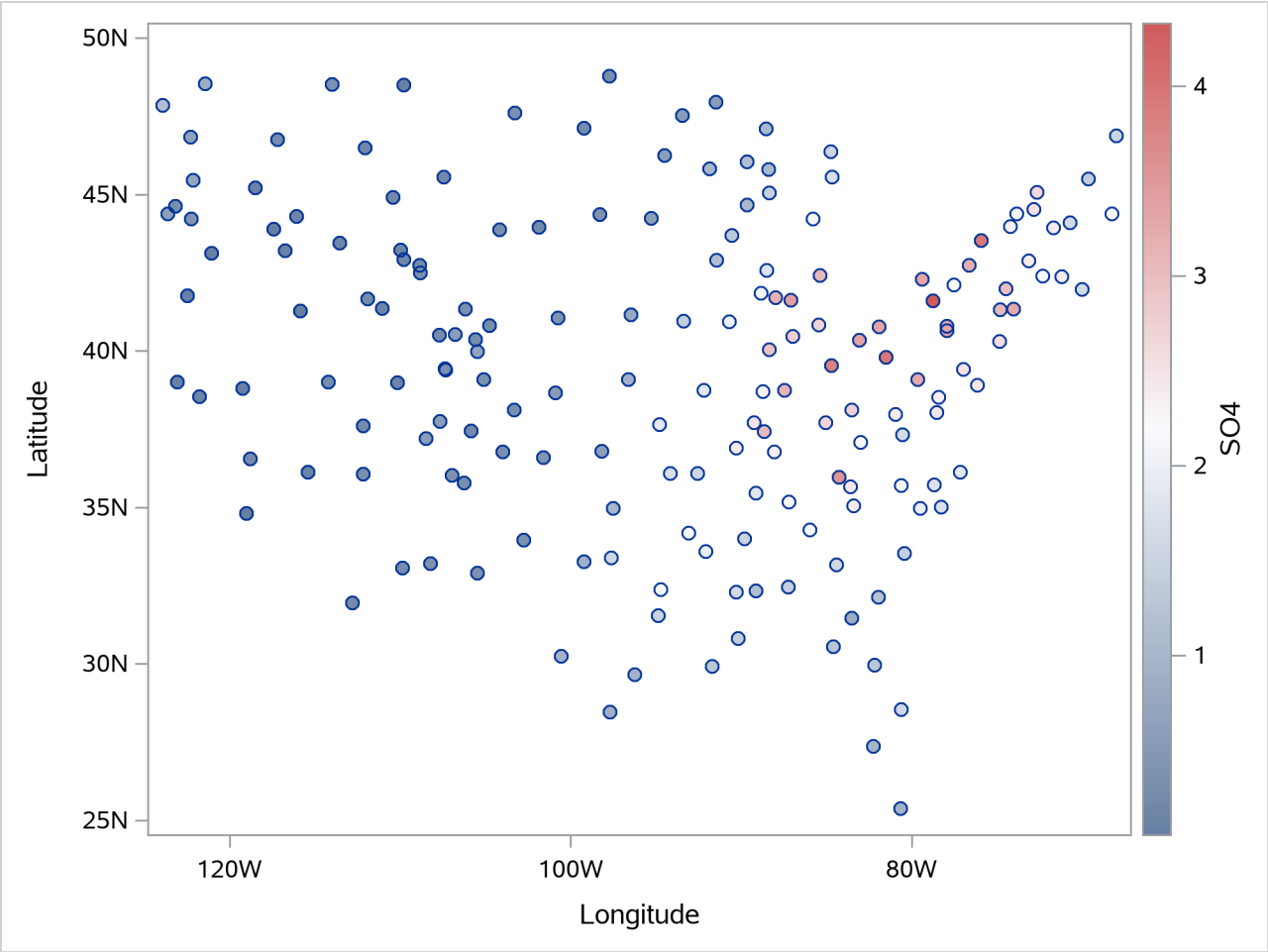

Figure 78.2.1 shows that the largest concentrations of sulfate deposits occur in the northeastern United States. The following statements fit a loess model.

```
ods graphics on;

proc loess data=SO4;
   model SO4=Longitude Latitude / degree=2 interp=cubic;
run;

ods graphics off;
```

Output 78.2.2 Fit Plot for the SO4 Data

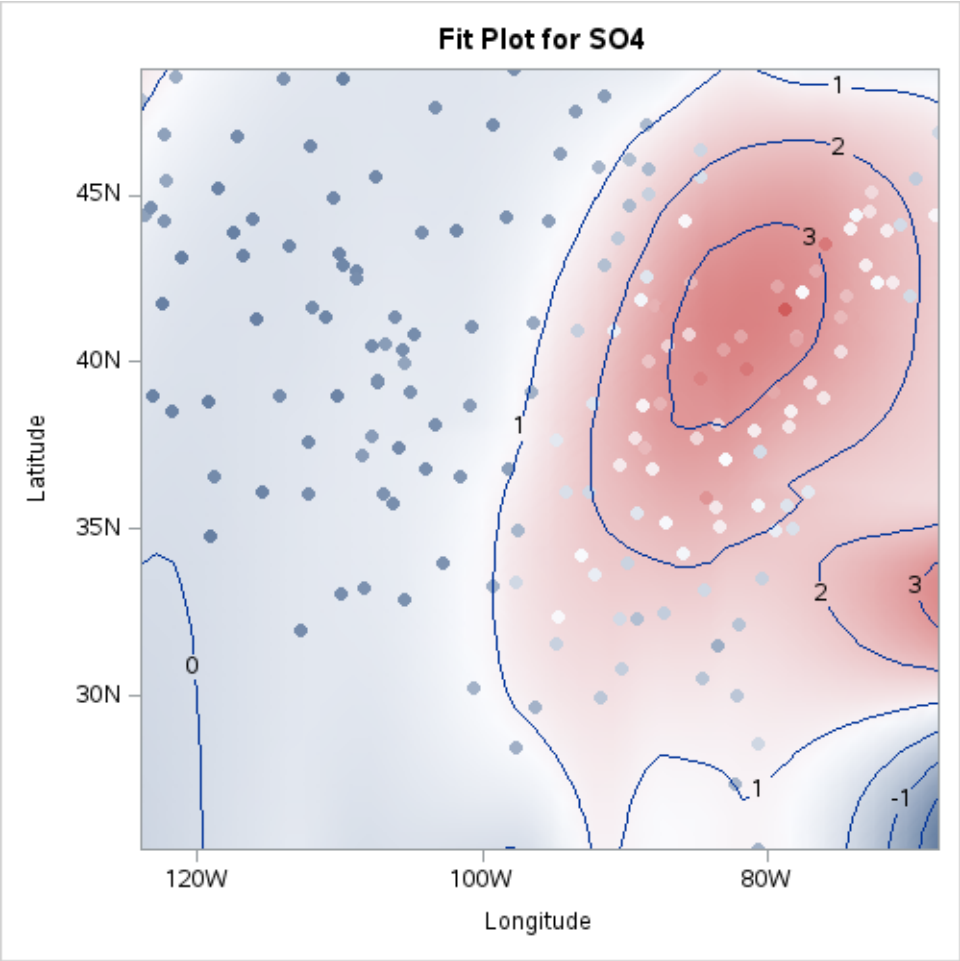

Figure 78.2.2 shows a contour plot of the fitted loess surface overlaid with a scatter plot of the data. The data are colored by the observed sulfate concentrations, using the same color gradient as the gradient-filled contour plot of the fitted surface. Note that for observations where the residual is small, the observations blend in with the contour plot. The greater the size of the residual, the greater the contrast between the observation color and the surface color.

The sulfate measurements are irregularly spaced. To facilitate producing a plot of the fitted loess surface, you can create a data set containing a regular grid of longitudes and latitudes and then use the SCORE statement to evaluate the loess surface at these points. The following statements show how you do this:

```
data PredPoints;
   format longitude longitude.
          latitude  latitude.;
   do Latitude = 26 to 46 by 1;
      do Longitude = -79 to -123 by -1;
         output;
      end;
   end;
run;
```

```
proc loess data=SO4;
   model SO4=Longitude Latitude;
   score data=PredPoints / print;
   ods Output ScoreResults=ScoreOut;
run;
```

The PRINT option in the SCORE statement requests that the "Score Results" table be displayed as part of the PROC LOESS output. The ODS OUTPUT statement outputs this table to a data set named ScoreOut. If you do not want to display the score results but you do want the score results in an output data set, then you can omit the PRINT option from the SCORE statement. To plot the surface shown in Figure 78.2.3 by using ODS Graphics, use the following statements:

```
proc template;
   define statgraph surface;
      begingraph;
         layout overlay3d / rotate=340 tilt=30 cube=false;
            surfaceplotparm x=Longitude y=Latitude z=p_SO4;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=ScoreOut template=surface;
run;
```

Output 78.2.3 Loess Fit of SO4 Surface

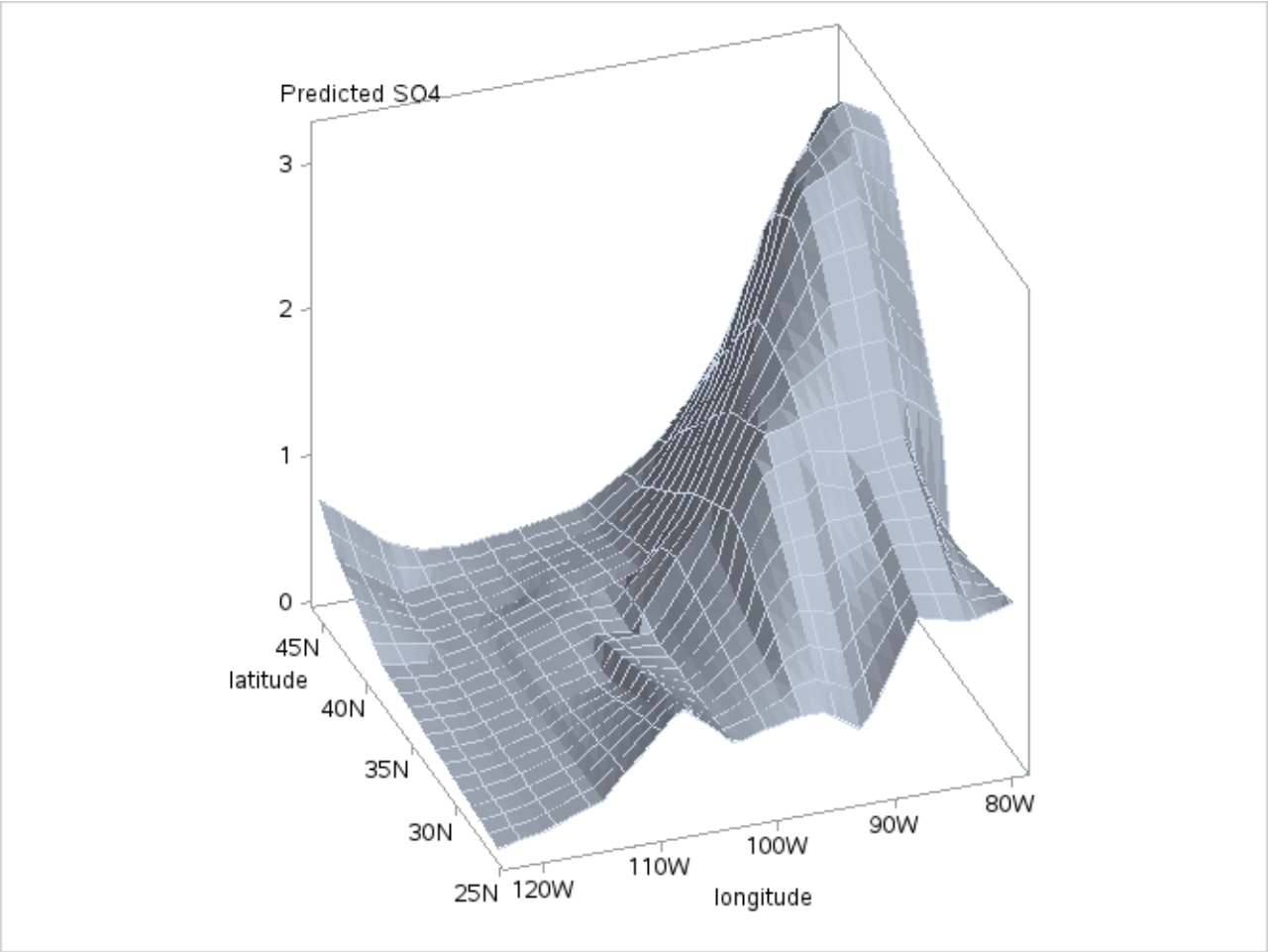

Example 78.3: Catalyst Experiment

(View the complete code for this example (loessex3.sas) in the example repository.)

The following data set records the results of an experiment to determine how the yield of a chemical reaction varies with temperature and amount of a catalyst used.

```
data Experiment;
   input Temperature Catalyst MeasuredYield;
   if ranuni(1) < 0.1
      then CorruptedYield = MeasuredYield + 10 * ranuni(1);
      else CorruptedYield = MeasuredYield;
   datalines;
 80    0.000     6.85601
 80    0.002     7.26355
 80    0.004     7.41448
 80    0.006     7.82640

   ... more lines ...
```

```
140    0.078     5.20562
140    0.080     5.49371
;
```

The aim of this example is to show how you can use PROC LOESS for robust fitting in the presence of outliers. To simulate an intermittent equipment malfunction, the variable CorruptedYield is the same as the variable MeasuredYield except for about 10% of the observations where an offset has been added. This example shows how you can use PROC LOESS obtain a fit for CorruptedYield that is close to the fit you obtain for MeasuredYield.

The following statements produce a scatter plot of Temperature by Catalyst where the observations are colored by CorruptedYield:

```
proc template;
   define statgraph gradientScatter;
      beginGraph;
         layout overlay;
            scatterPlot x=Catalyst y=Temperature /
                markercolorgradient = CorruptedYield
                markerattrs         = (symbol=circleFilled)
                colormodel          =  ThreeColorRamp
                name                =  "Yield";

            scatterPlot x=Catalyst y=Temperature /
                markerattrs         = (symbol=circle);

            continuousLegend "Yield" / title= "CorruptedYield";
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=Experiment template=gradientScatter;
run;
```

Output 78.3.1 Scatter Plot of Experiment Data Colored by CorruptedYield

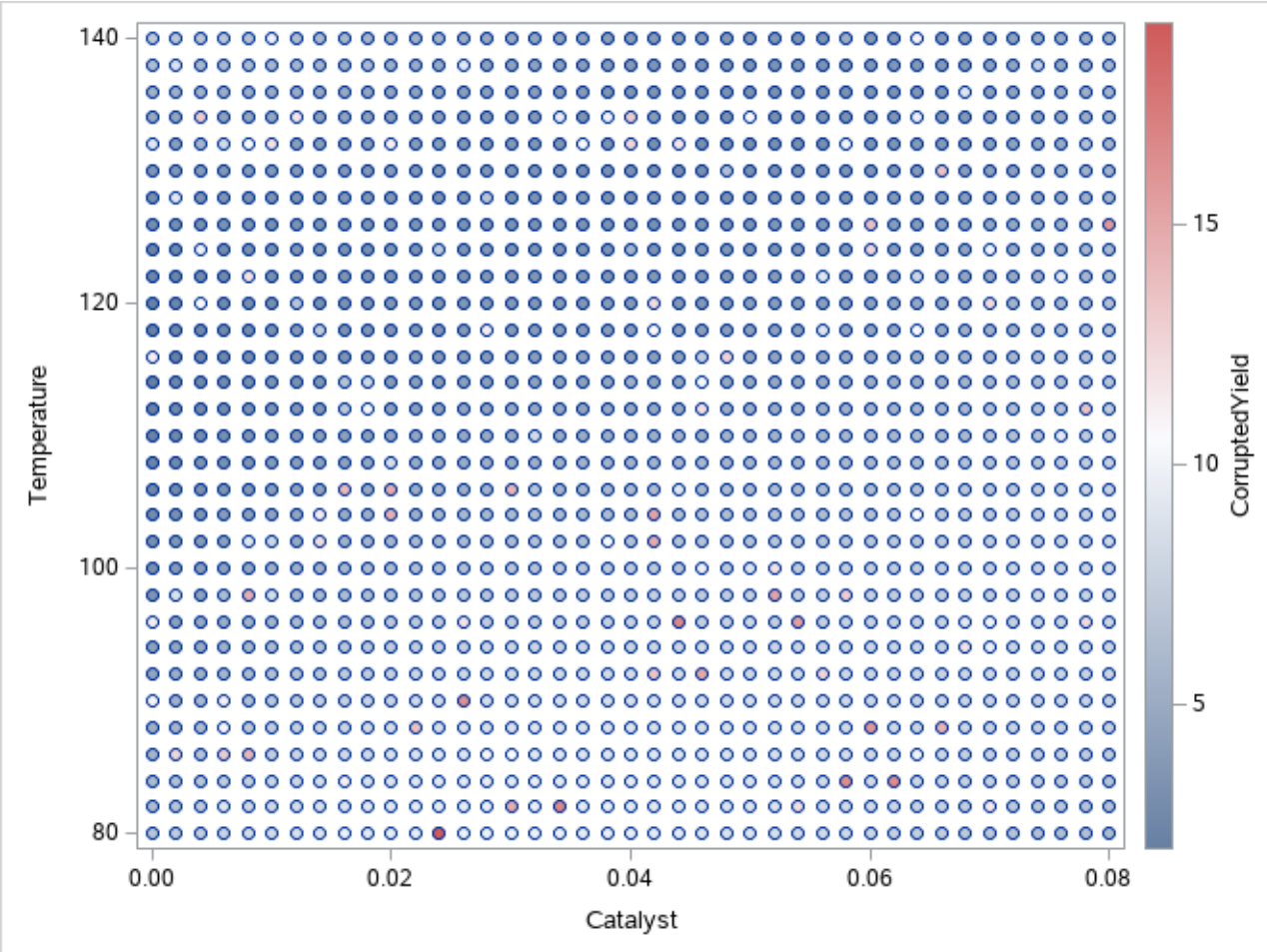

Output 78.3.1 shows a scatter plot of the data where the observations are shaded by the value of CorruptedYield. The darkly shaded points that are surrounded by lightly shaded points are points where the simulated incorrect measurements occur.

The following code fits a loess model to the measured data:

```
ods graphics on;

proc loess data=Experiment;
   model MeasuredYield = Temperature Catalyst / scale=sd(0.1);
run;
```

Output 78.3.2 Scale Details for the Experiment Data

The LOESS Procedure

Independent Variable Scaling		
Scaling applied: 10% trimmed standard deviation		
Statistic	**Temperature**	**Catalyst**
Minimum Value	80.00000	0
Maximum Value	140.00000	0.08000
Trimmed Mean	110.00000	0.04000
Trimmed Standard Deviation	14.32149	0.01894

The SCALE=SD(0.1) option in the MODEL statement specifies that the independent variables in the model are to be divided by their respective 10% trimmed standard deviations before the fitted model is computed. This is appropriate because the independent variables Temperature and Catalyst are not similarly scaled. The "Scale Details" table in Output 78.3.2 displays the details of ranges of the regressors and the scale factors applied to each regressor.

Output 78.3.3 displays the loess fit. Because the fitted surface is a good fit of the observed data, the observations on this plot are not clearly distinguishable from the fitted surface. The results are dramatically different when the outliers are included. The following statements fit a loess model to the corrupted response, using the same smoothing parameter that was selected for the measured response.

Output 78.3.3 Fit for MeasuredYield

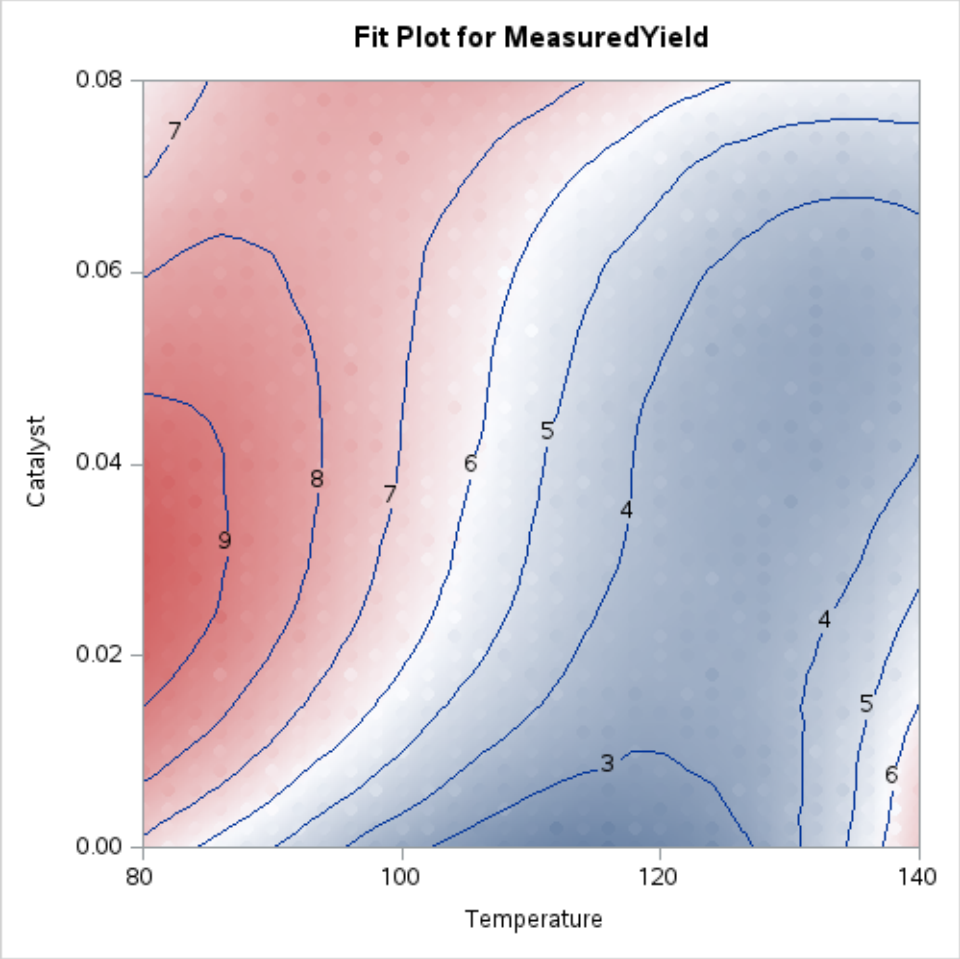


```
proc loess data=Experiment;
   model CorruptedYield = Temperature Catalyst /
                  scale=sd(0.1) smooth=0.018;
run;
```

Output 78.3.4 Fit for CorruptedYield

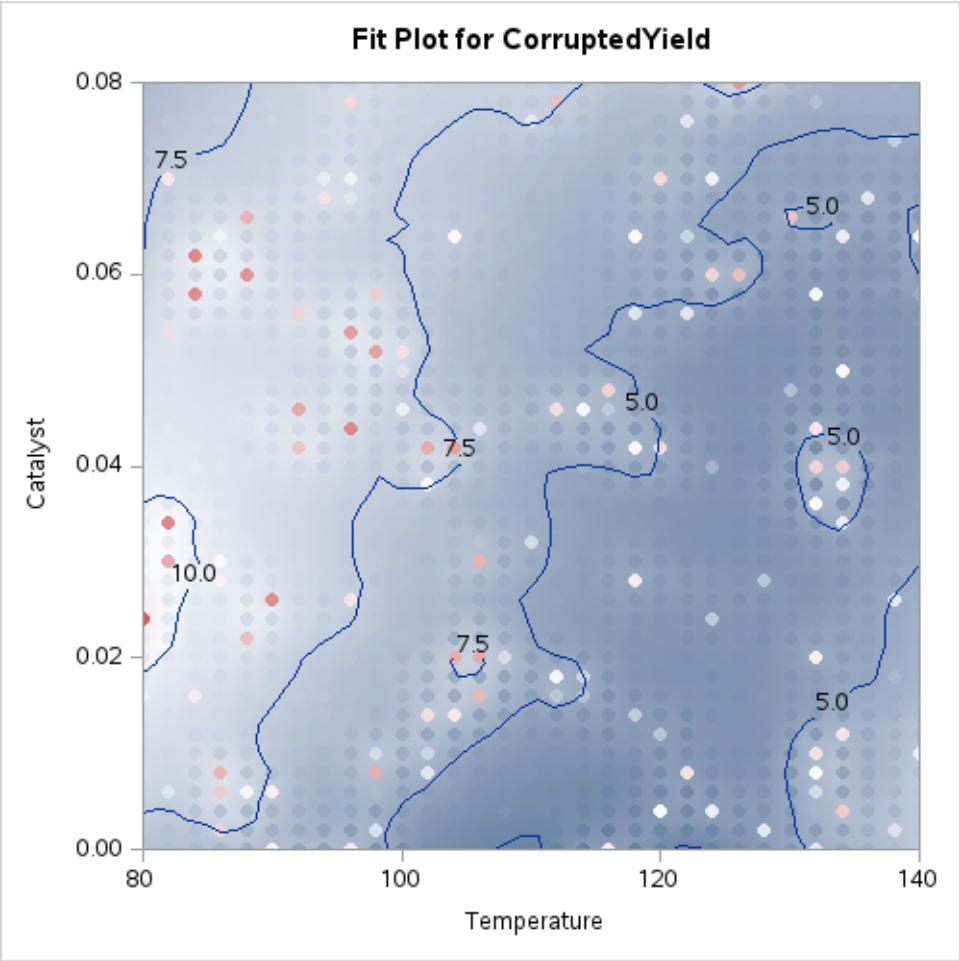

Output 78.3.4 displays the loess fit. The fit is pulled upward in the neighborhoods of these outliers. If you use a larger smoothing parameter value, then these local perturbations in the fit get smoothed out, but at the expense of smoothing away the information in the underlying measured response. In such cases a robust fitting method is indicated. The following statements show how you do this:

```
proc loess data=Experiment;
   model CorruptedYield = Temperature Catalyst /
                          scale  = sd(0.1)
                          smooth = 0.018
                          iterations=4;
run;
```

The ITERATIONS=4 option in the MODEL statement requests the initial loess fit followed by three iteratively reweighted iterations.

Output 78.3.5 Robust Fit for CorruptedYield

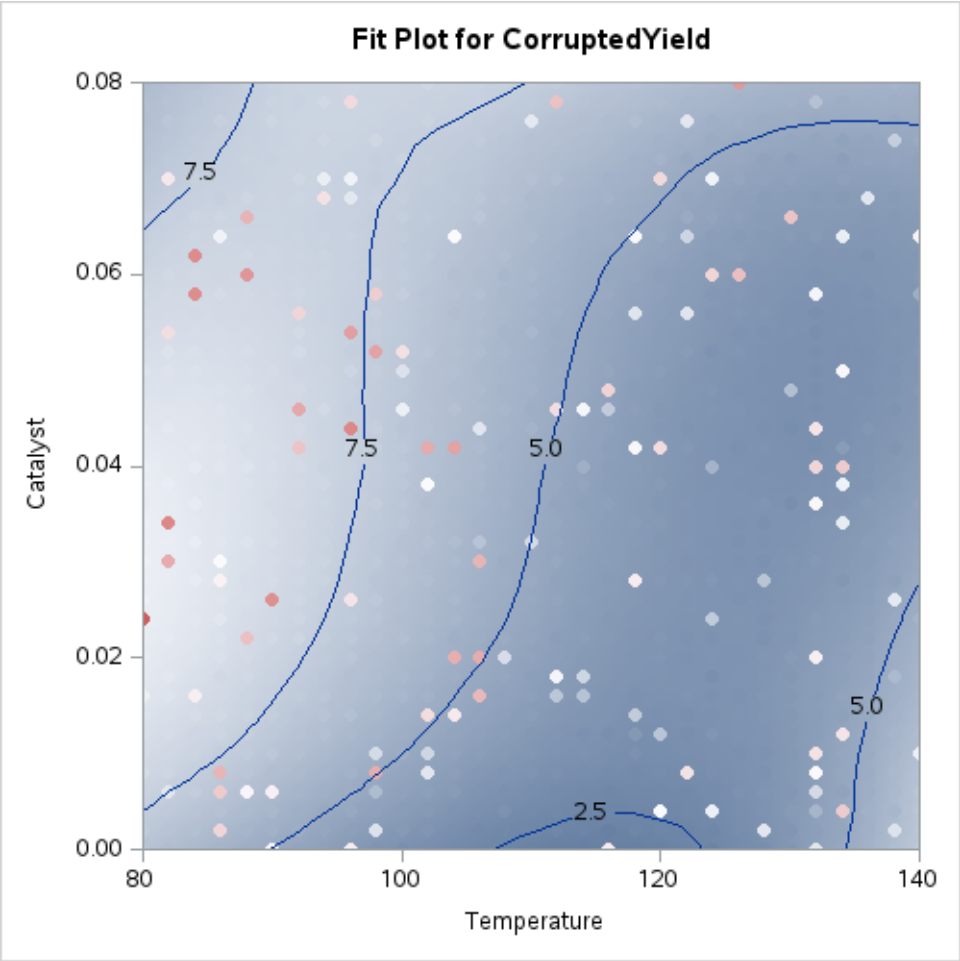

You can see the impact of the robust fitting by comparing the robust fit shown in Output 78.3.5 with the nonrobust fit in Output 78.3.4. In the robust fit you see that the local perturbations caused by the outliers have been eliminated as these the outlying observations get down-weighted during the robustness iterations. By comparing the labeled contours on the fit plot for the uncorrupted response shown in Output 78.3.3 with the labeled contours for the corrupted response shown in Output 78.3.4, you can see that the robust fit has produced a reasonable fit for the underlying measured data. The color gradient in Output 78.3.5 is chosen to accommodate the outliers that are present in the observed data, and so you cannot easily compare the color gradient in this plot with that in Output 78.3.3. The following statements repeat the robust analysis with an option added to suppress the display of the observations on the fit plot:

```
proc loess data=Experiment plots=contourFit(obs=none);
   model CorruptedYield = Temperature Catalyst /
                             scale  = sd(0.1)
                             smooth = 0.018
                             iterations=4;
run;

ods graphics off;
```

Output 78.3.6 Robust Fit for CorruptedYield with Observations Suppressed

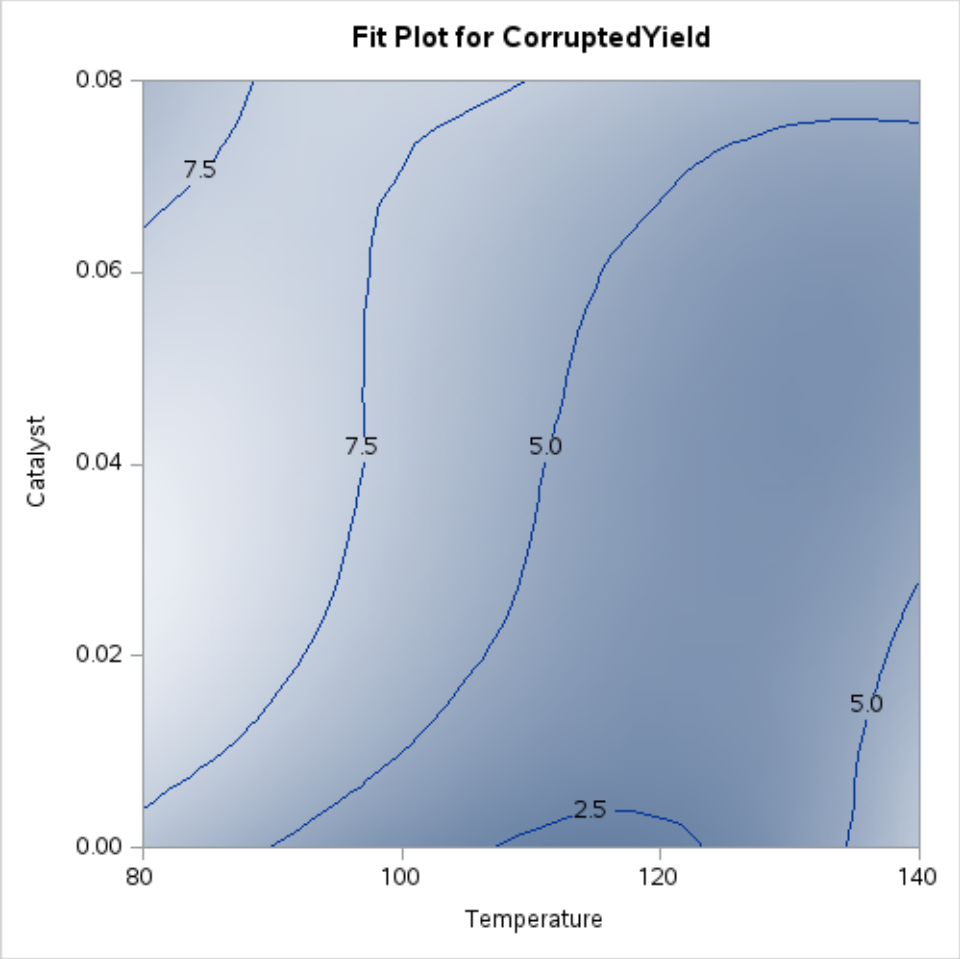

Output 78.3.6 shows the robust fit with the observations suppressed. The range of the fitted surface values in this plot is similar to the range in Output 78.3.3. By comparing this contour plot with the contour plot in Output 78.3.3, you clearly see that the robust loess fit has successfully modeled the underlying surface despite the presence of the outliers.

Example 78.4: El Niño Southern Oscillation

(View the complete code for this example (loessex4.sas) in the example repository.)

The data set sashelp.ENSO, which is available in the Sashelp library, contains measurements of monthly averaged atmospheric pressure differences between Easter Island and Darwin, Australia, for a period of 168 months (National Institute of Standards and Technology 1998).

The following PROC SGPLOT statements produce the simple scatter plot of the ENSO data, displayed in Output 78.4.1.

```
proc sgplot data=sashelp.ENSO;
   scatter y=Pressure x=Month;
run;
```

Output 78.4.1 Scatter Plot of ENSO Data

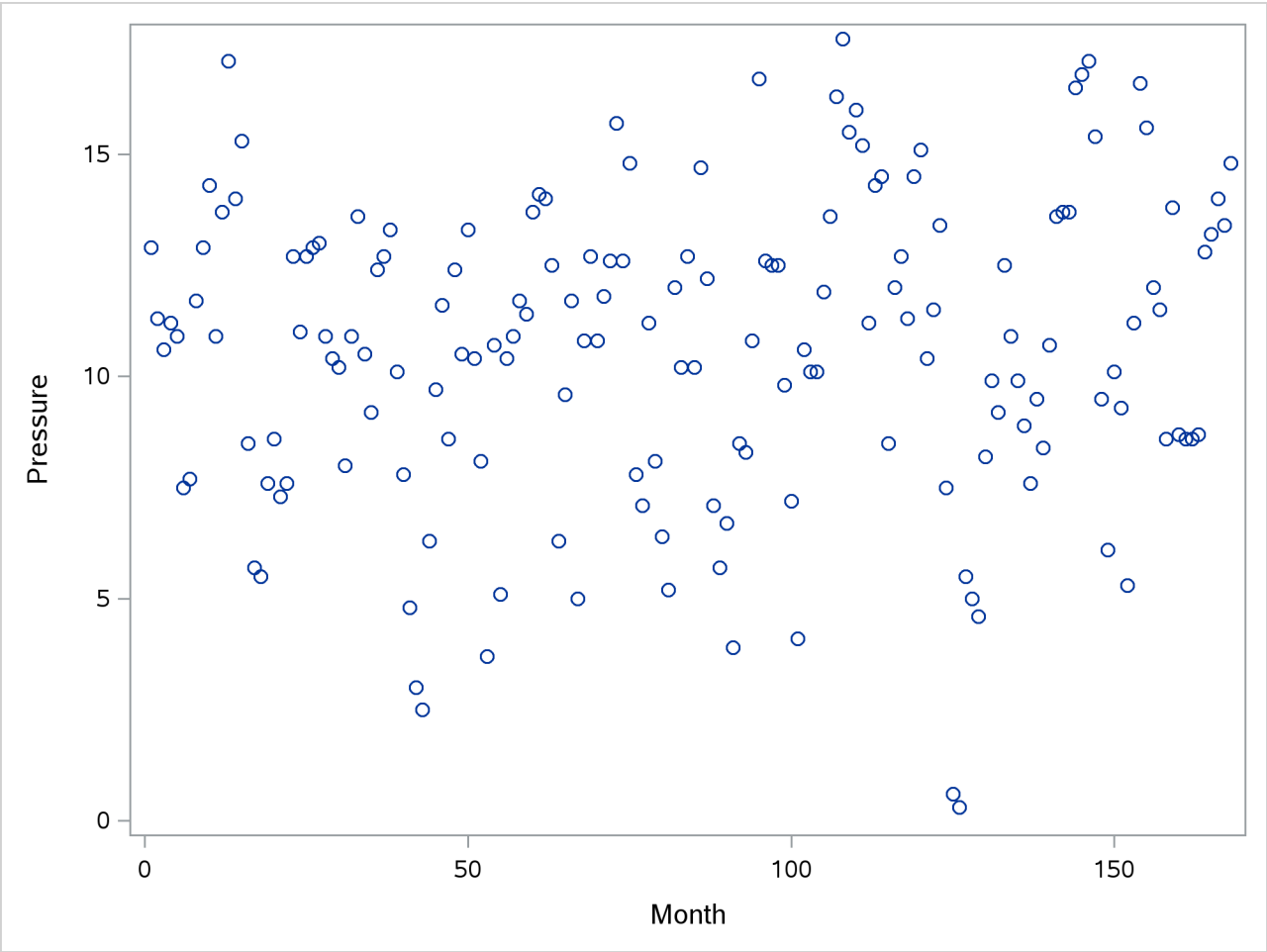

You can compute a loess fit and obtain graphical results for these data by using the following statements:

```
ods graphics on;

proc loess data=sashelp.ENSO plots=residuals(smooth);
   model Pressure=Month;
run;
```

The “Smoothing Criterion” and “Fit Summary” tables are shown in Output 78.4.2, and the fit plot is shown in Output 78.4.3.

Output 78.4.2 Output from PROC LOESS

The LOESS Procedure
Dependent Variable: Pressure

Optimal Smoothing Criterion	
AICC	Smoothing Parameter
3.41105	0.22321

The LOESS Procedure
Selected Smoothing Parameter: 0.223
Dependent Variable: Pressure

Fit Summary	
Fit Method	kd Tree
Blending	Linear
Number of Observations	168
Number of Fitting Points	33
kd Tree Bucket Size	7
Degree of Local Polynomials	1
Smoothing Parameter	0.22321
Points in Local Neighborhood	37
Residual Sum of Squares	1654.27725
Trace[L]	8.74180
GCV	0.06522
AICC	3.41105

Output 78.4.3 Oversmoothed Loess Fit for the ENSO Data

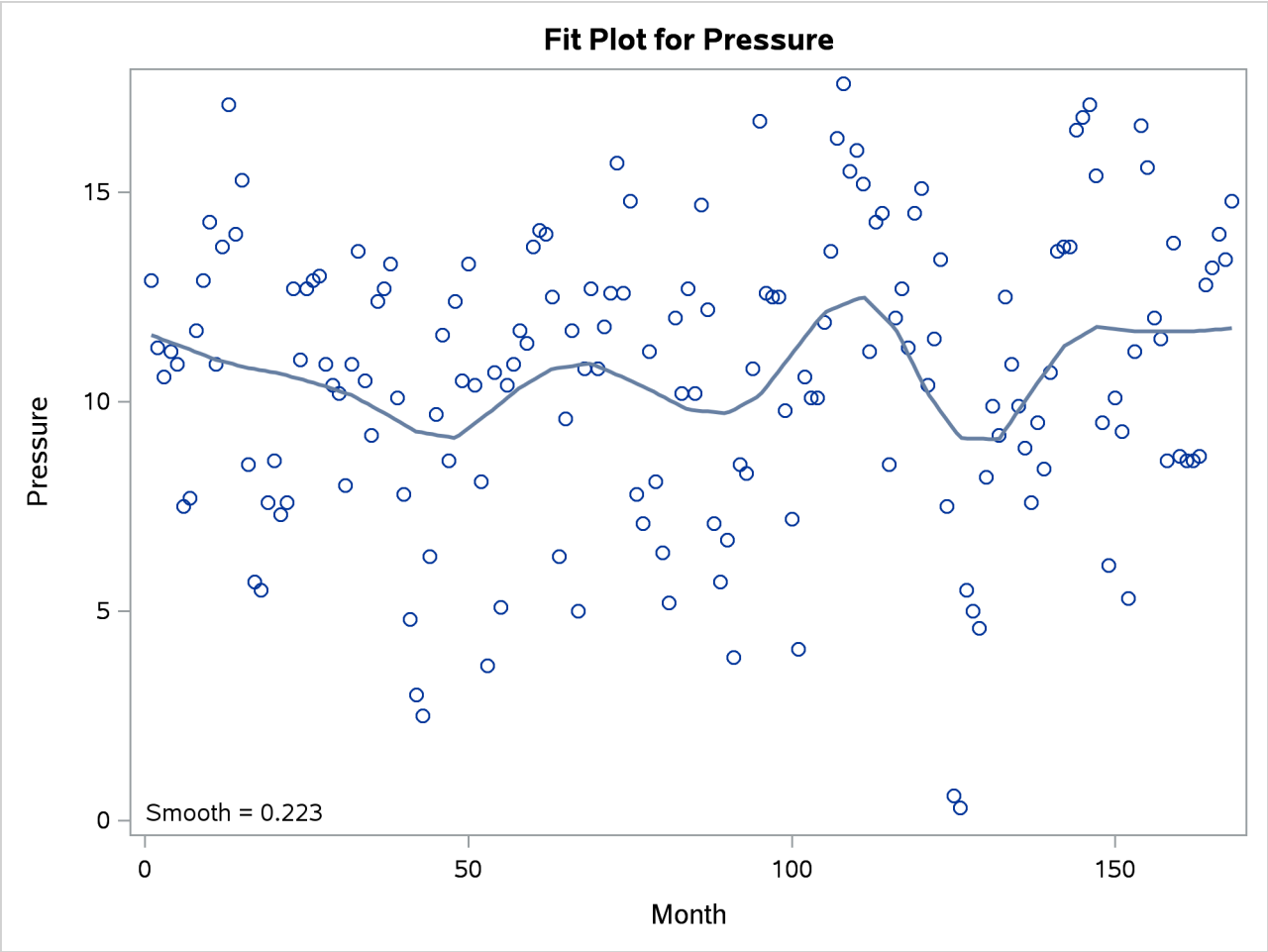

This weather-related data should exhibit an annual cycle. However, the loess fit in Output 78.4.3 indicates a longer cycle but no annual cycle. This suggests that the loess fit is oversmoothed. One way to detect oversmoothing is to look for patterns in the fit residuals. With ODS Graphics enabled, PROC LOESS produces a scatter plot of the residuals versus each regressor in the model. To aid in visually detecting patterns in these scatter plots, it is useful to superimpose a nonparametric fit on these scatter plots. You can request this by specifying the SMOOTH suboption of the PLOTS=RESIDUALS option in the PROC LOESS statement. The nonparametric fit that is produced is again a loess fit that is produced independently of the loess fit used to obtain these residuals.

With the superimposed loess fit shown in Output 78.4.4, you can clearly identify an annual cycle in the residuals, which confirms that the loess fit for the ENSO is oversmoothed. What accounts for this poor fit?

Output 78.4.4 Residuals for the Loess Fit for the ENSO Data

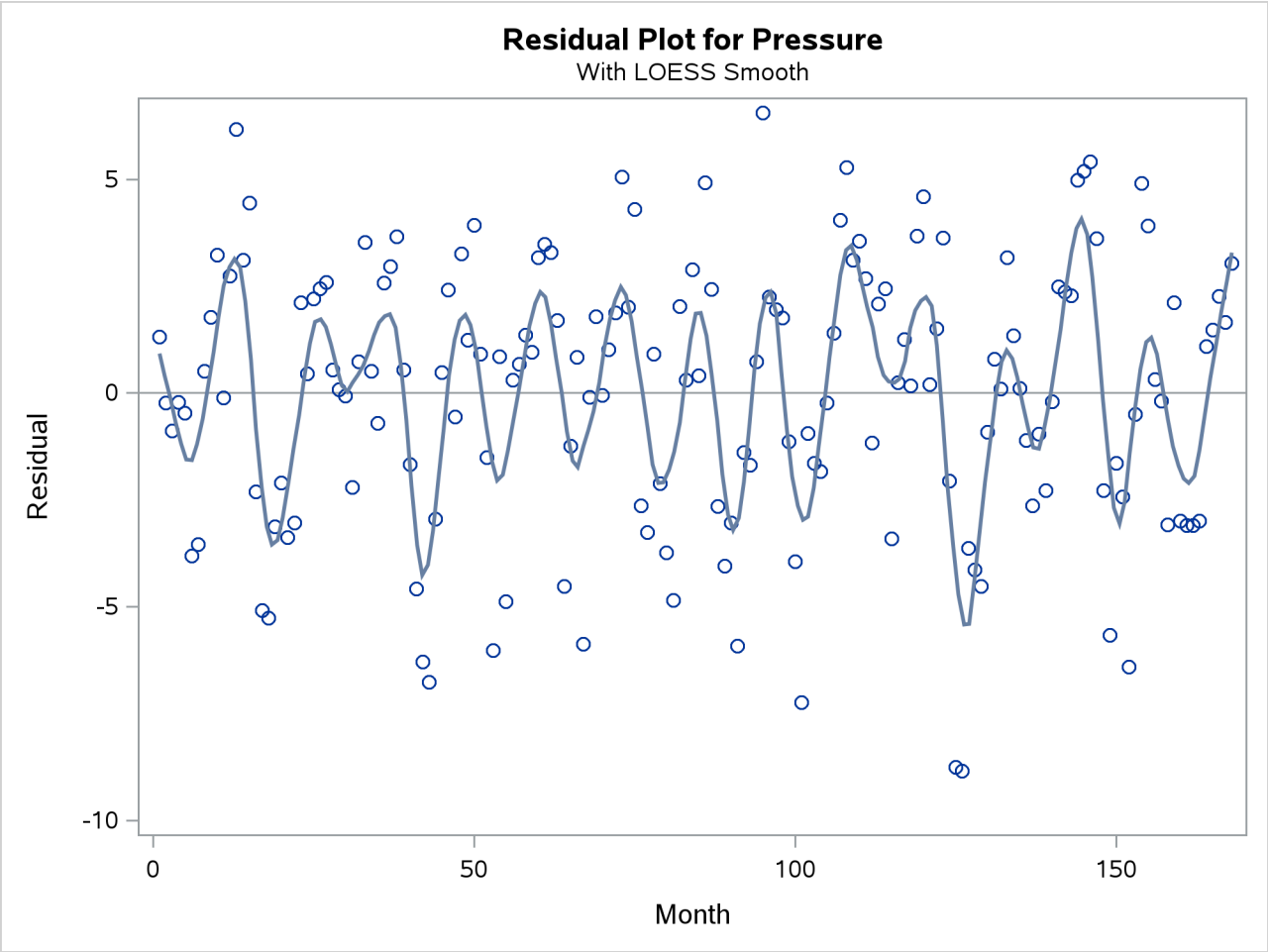

The smoothing parameter value used for the loess fit shown in Output 78.4.3 was chosen using the default method of PROC LOESS, namely a golden section minimization of the AICC criterion over the interval $(0, 1]$. One possibility is that the golden section search has found a local rather than a global minimum of the AICC criterion. You can test this by redoing the fit requesting a global minimum. You do this with the following statements:

```
proc loess data=sashelp.ENSO;
   model Pressure=Month/select=AICC(global);
run;
```

Output 78.4.5 AICC versus Smoothing Parameter Showing Local Minima

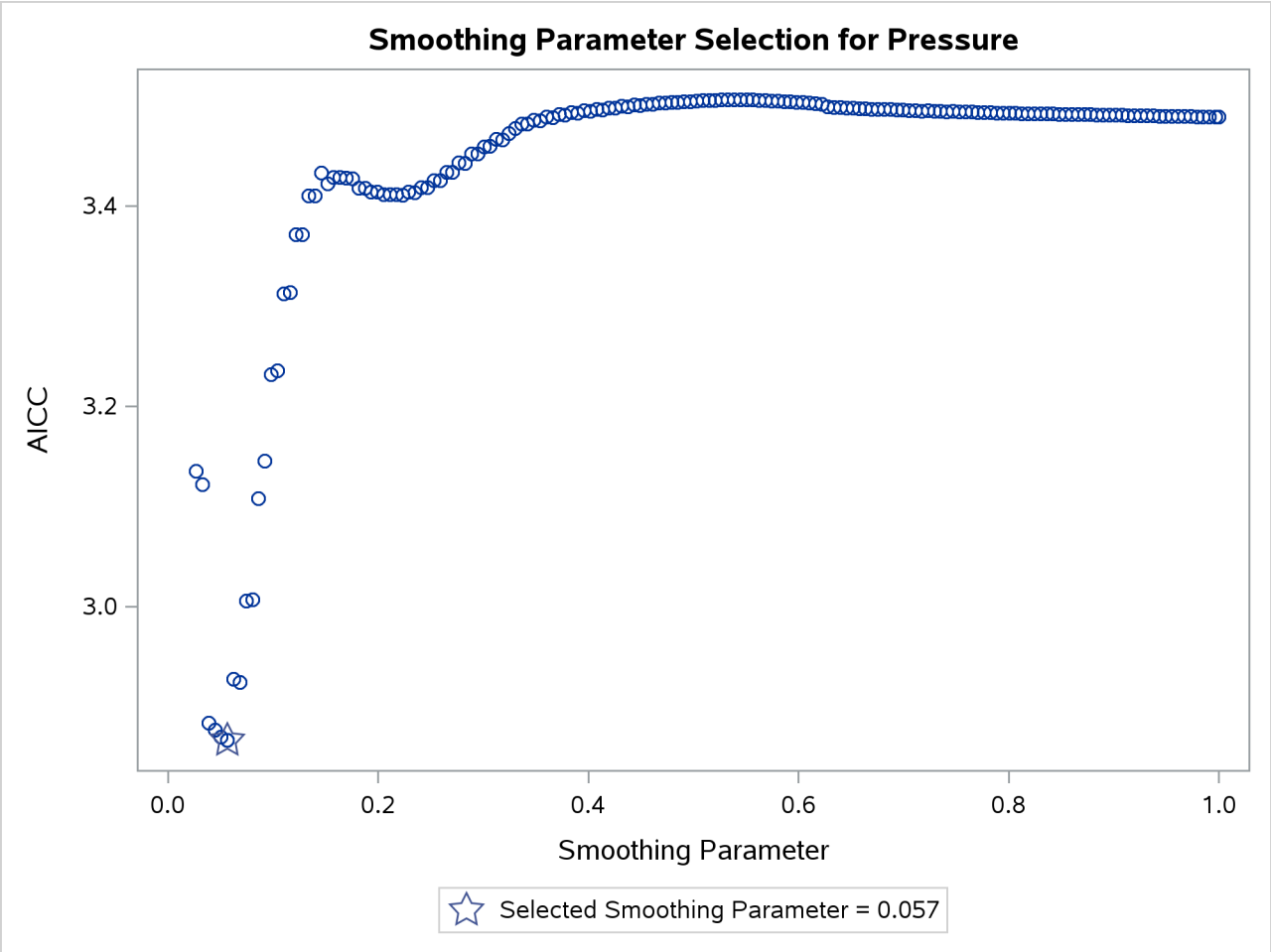

The explanation for the oversmoothed fit in Output 78.4.3 is now apparent. Output 78.4.5 shows that the golden section search algorithm found the local minimum that occurs near the value 0.22 of the smoothing parameter rather than the global minimum that occurs near 0.06. Note that if you restrict the range of smoothing parameter values examined to lie below 0.2, then the golden section search finds the global minimum, as the following statements demonstrate:

```
proc loess data=sashelp.ENSO;
   model Pressure=Month/select=AICC(range(0.03,0.2));
run;
```

Output 78.4.6 Selected Smoothing Parameter Value

The LOESS Procedure
Dependent Variable: Pressure

Optimal Smoothing Criterion	
AICC	Smoothing Parameter
2.86660	0.05655

Output 78.4.6 shows that with the restricted range of smoothing parameter values examined, PROC LOESS finds the global minimum of the AICC criterion. Often you might not know an appropriate range of smoothing parameter values to examine. In such cases, you can use the PRESEARCH suboption of the SELECT= option in the MODEL statement. When you specify this option, PROC LOESS does a preliminary search to try to locate a smoothing parameter value range that contains just the first local minimum of the criterion being used for the selection. The following statements provide an example.

```
proc loess data=sashelp.ENSO plots=residuals(smooth);
   model Pressure=Month/select=AICC(presearch);
run;

ods graphics off;
```

Output 78.4.7 Selected Smoothing Parameter Value When Presearch Is Specified

The LOESS Procedure
Dependent Variable: Pressure

Optimal Smoothing Criterion	
AICC	Smoothing Parameter
2.86660	0.05655

Output 78.4.7 shows that with the PRESEARCH suboption specified, PROC LOESS selects the smoothing parameter value that yields the global minimum of the AICC criterion. The fit obtained is shown in Output 78.4.8, and a plot of the residuals with a superimposed loess fit is shown in Output 78.4.9.

Output 78.4.8 Loess Fit Showing an Annual Cycle

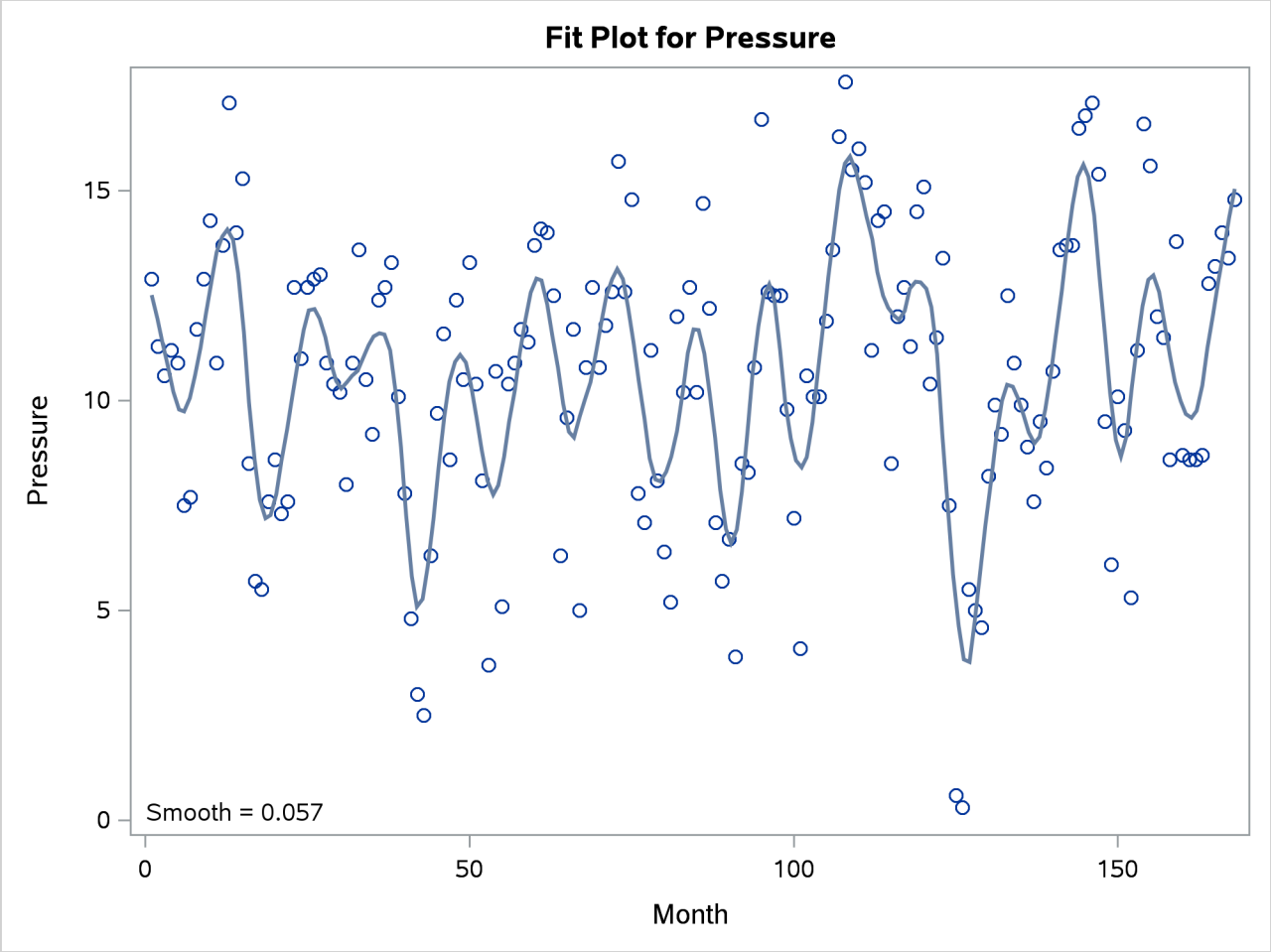

Output 78.4.9 Residuals of the Selected Model

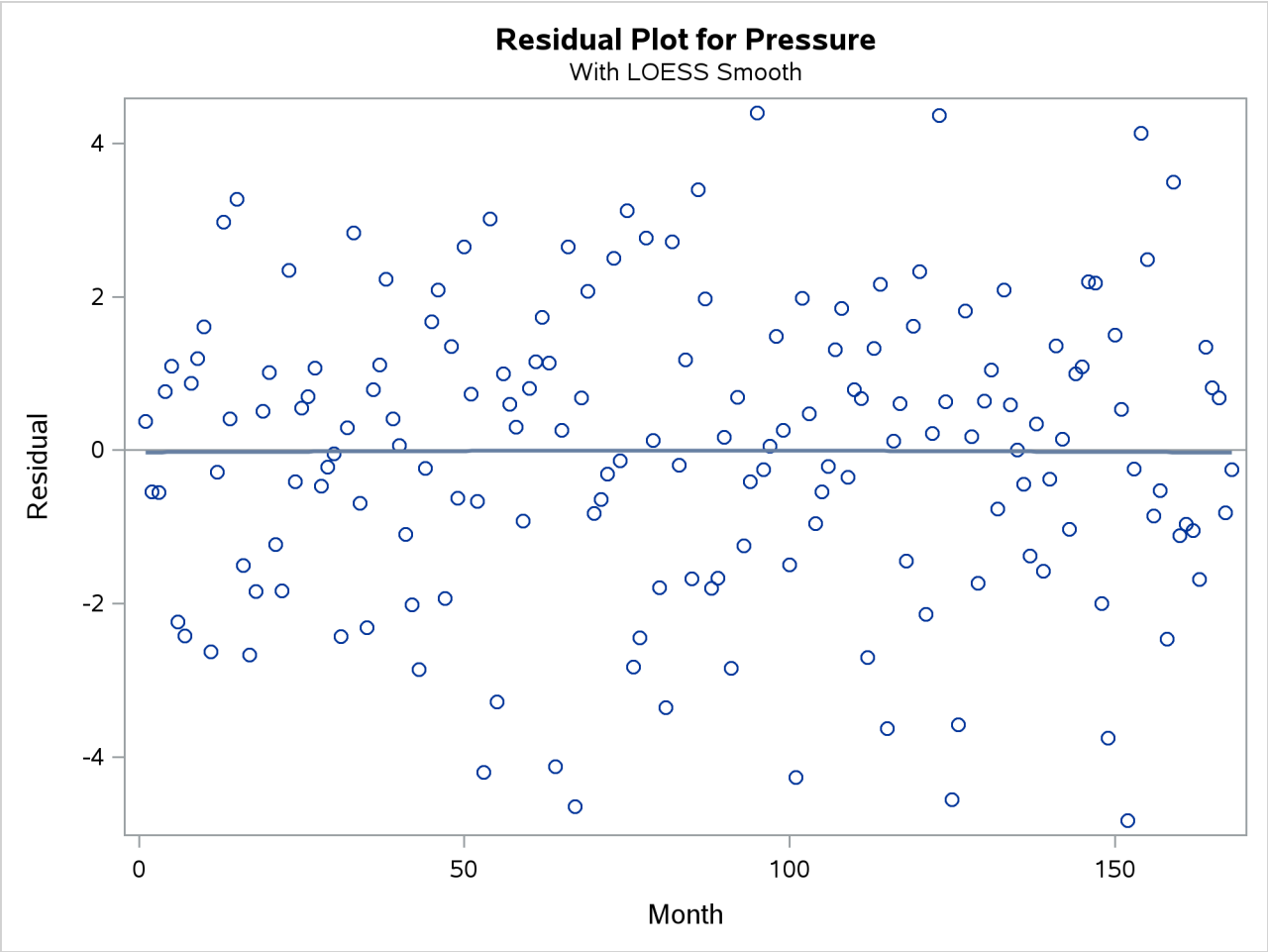

In contrast to the residual plot show in Output 78.4.4, the residuals plotted in Output 78.4.9 do not exhibit any pattern, indicating that the corresponding loess fit has captured all the systematic variation in the data.

An interesting question is whether there is some phenomenon captured in the data that would explain the presence of the local minimum near 0.22 in the AICC curve. Note that there is some evidence of a cycle of about 42 months in the oversmoothed fit in Output 78.4.3. You can see this cycle because the strong annual cycle in Output 78.4.8 has been smoothed out. The physical phenomenon that accounts for the existence of this cycle has been identified as the periodic warming of the Pacific Ocean known as "El Niño."

References

Akaike, H. (1973). "Information Theory and an Extension of the Maximum Likelihood Principle." In *Proceedings of the Second International Symposium on Information Theory*, edited by B. N. Petrov and F. Csáki, 267–281. Budapest: Akademiai Kiado.

Brinkman, N. D. (1981). "Ethanol Fuel: A Single-Cylinder Engine Study of Efficiency and Exhaust Emissions." *Society of Automotive Engineers Transactions* 90:1410–1424.

Cleveland, W. S. (1993). *Visualizing Data*. Summit, NJ: Hobart Press.

Cleveland, W. S., Devlin, S. J., and Grosse, E. (1988). "Regression by Local Fitting." *Journal of Econometrics* 37:87–114.

Cleveland, W. S., and Grosse, E. (1991). "Computational Methods for Local Regression." *Statistics and Computing* 1:47–62.

Cleveland, W. S., Grosse, E., and Shyu, M.-J. (1992). "A Package of C and Fortran Routines for Fitting Local Regression Models." Unpublished manuscript.

Craven, P., and Wahba, G. (1979). "Smoothing Noisy Data with Spline Functions." *Numerical Mathematics* 31:377–403.

Gordon, W. J. (1971). "Blending-Function Methods of Bivariate and Multivariate Interpolation and Approximation." *SIAM Journal on Numerical Analysis* 8:158–177.

Hastie, T. J., and Tibshirani, R. J. (1990). *Generalized Additive Models*. New York: Chapman & Hall.

Houghton, A. N., Flannery, J., and Viola, M. V. (1980). "Malignant Melanoma in Connecticut and Denmark." *International Journal of Cancer* 25:95–104.

Hurvich, C. M., Simonoff, J. S., and Tsai, C.-L. (1998). "Smoothing Parameter Selection in Nonparametric Regression Using an Improved Akaike Information Criterion." *Journal of the Royal Statistical Society, Series B* 60:271–293.

National Institute of Standards and Technology (1998). "Statistical Reference Data Sets." Accessed June 6, 2011. http://www.itl.nist.gov/div898/strd/general/dataarchive.html.

Ruppert, D., Sheather, S. J., and Wand, M. P. (1995). "An Effective Bandwidth Selector for Local Least Squares Regression." *Journal of the American Statistical Association* 90:1257–1270.

Chapter 79
The LOGISTIC Procedure

Contents

Overview: LOGISTIC Procedure . . . **5939**
Getting Started: LOGISTIC Procedure . . . **5942**
Syntax: LOGISTIC Procedure . . . **5949**
- PROC LOGISTIC Statement . . . 5950
- BY Statement . . . 5968
- CLASS Statement . . . 5969
- CODE Statement . . . 5972
- CONTRAST Statement . . . 5973
- EFFECT Statement . . . 5976
- EFFECTPLOT Statement . . . 5977
- ESTIMATE Statement . . . 5978
- EXACT Statement . . . 5979
- EXACTOPTIONS Statement . . . 5982
- FREQ Statement . . . 5986
- ID Statement . . . 5986
- LSMEANS Statement . . . 5986
- LSMESTIMATE Statement . . . 5988
- MARGINS Statement . . . 5989
- MODEL Statement . . . 5990
- NLOPTIONS Statement . . . 6010
- ODDSRATIO Statement . . . 6010
- OUTPUT Statement . . . 6012
- ROC Statement . . . 6017
- ROCCONTRAST Statement . . . 6018
- SCORE Statement . . . 6019
- SLICE Statement . . . 6021
- STORE Statement . . . 6022
- STRATA Statement . . . 6022
- TEST Statement . . . 6023
- UNITS Statement . . . 6024
- WEIGHT Statement . . . 6025

Details: LOGISTIC Procedure . . . **6026**
- Missing Values . . . 6026
- Response Level Ordering . . . 6026
- Link Functions and the Corresponding Distributions . . . 6028
- Determining Observations for Likelihood Contributions . . . 6029

Iterative Algorithms for Model Fitting . . . 6030
Convergence Criteria . . . 6032
Existence of Maximum Likelihood Estimates . . . 6032
Effect-Selection Methods . . . 6034
Model Fitting Information . . . 6035
Score Statistics and Tests . . . 6039
Confidence Intervals for Parameters . . . 6041
Odds Ratio Estimation . . . 6043
Linear Predictor, Predicted Probability, and Confidence Limits . . . 6046
Classification Table . . . 6047
Goodness-of-Fit Tests . . . 6050
Overdispersion . . . 6053
Receiver Operating Characteristic Curves . . . 6055
Precision-Recall Curves . . . 6058
Testing Linear Hypotheses about the Regression Coefficients . . . 6059
Joint Tests and Type 3 Tests . . . 6059
Regression Diagnostics . . . 6060
Scoring Data Sets . . . 6063
Conditional Logistic Regression . . . 6068
Exact Conditional Logistic Regression . . . 6071
Input and Output Data Sets . . . 6076
Computational Resources . . . 6082
Displayed Output . . . 6084
ODS Table Names . . . 6090
ODS Graphics . . . 6092
Examples: LOGISTIC Procedure . . . **6095**
Example 79.1: Stepwise Logistic Regression and Predicted Values . . . 6095
Example 79.2: Logistic Modeling with Categorical Predictors . . . 6107
Example 79.3: Ordinal Logistic Regression . . . 6116
Example 79.4: Nominal Response Data: Generalized Logits Model . . . 6123
Example 79.5: Stratified Sampling . . . 6130
Example 79.6: Logistic Regression Diagnostics . . . 6132
Example 79.7: ROC Curve, Customized Odds Ratios, Goodness-of-Fit Statistics, R-Square, and Confidence Limits . . . 6140
Example 79.8: Comparing Areas under Receiver Operating Characteristic Curves . . . 6145
Example 79.9: Goodness-of-Fit Tests and Subpopulations . . . 6152
Example 79.10: Overdispersion . . . 6155
Example 79.11: Conditional Logistic Regression for Matched Pairs Data . . . 6159
Example 79.12: Exact Conditional Logistic Regression . . . 6162
Example 79.13: Firth's Penalized Likelihood Compared with Other Approaches . . . 6166
Example 79.14: Complementary Log-Log Model for Infection Rates . . . 6170
Example 79.15: Complementary Log-Log Model for Interval-Censored Survival Times 6174
Example 79.16: Scoring Data Sets . . . 6179
Example 79.17: Using the LSMEANS Statement . . . 6184

Example 79.18: Partial Proportional Odds Model 6189
Example 79.19: Goodness-of-Fit Tests and Calibration 6194
Example 79.20: Using the Optimal ROC Criteria 6200
References **6203**

Overview: LOGISTIC Procedure

Binary responses (for example, success and failure), ordinal responses (for example, normal, mild, and severe), and nominal responses (for example, major TV networks viewed at a certain hour) arise in many fields of study. Logistic regression analysis is often used to investigate the relationship between these discrete responses and a set of explanatory variables. Texts that discuss logistic regression include Agresti (2013); Allison (2012); Collett (2003); Cox and Snell (1989); Hosmer and Lemeshow (2013); Stokes, Davis, and Koch (2012).

For binary response models, the response, Y, of an individual or an experimental unit can take on one of two possible values, denoted for convenience by 1 and 2 (for example, $Y = 1$ if a disease is present, otherwise $Y = 2$). Suppose $\mathbf{x}$ is a vector of explanatory variables and $\pi = \Pr(Y = 1 \mid \mathbf{x})$ is the response probability to be modeled. The linear logistic model has the form

$$\text{logit}(\pi) \equiv \log\left(\frac{\pi}{1-\pi}\right) = \alpha + \boldsymbol{\beta}'\mathbf{x}$$

where α is the intercept parameter and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_s)'$ is the vector of s slope parameters. Notice that the LOGISTIC procedure, by default, models the probability of the *lower* response levels.

The logistic model shares a common feature with a more general class of linear models: a function $g = g(\mu)$ of the mean of the response variable is assumed to be linearly related to the explanatory variables. Because the mean μ implicitly depends on the stochastic behavior of the response, and the explanatory variables are assumed to be fixed, the function g provides the link between the random (stochastic) component and the systematic (deterministic) component of the response variable Y. For this reason, Nelder and Wedderburn (1972) refer to $g(\mu)$ as a link function. One advantage of the logit function over other link functions is that differences on the logistic scale are interpretable regardless of whether the data are sampled prospectively or retrospectively (McCullagh and Nelder 1989, Chapter 4). Other link functions that are widely used in practice are the probit function and the complementary log-log function. The LOGISTIC procedure enables you to choose one of these link functions, resulting in fitting a broader class of binary response models of the form

$$g(\pi) = \alpha + \boldsymbol{\beta}'\mathbf{x}$$

For ordinal response models, the response, Y, of an individual or an experimental unit might be restricted to one of a (usually small) number of ordered values, denoted for convenience by $1, \ldots, k, k+1$. For example, the severity of coronary disease can be classified into three response categories as 1 = no disease, 2 = angina pectoris, and 3 = myocardial infarction. The LOGISTIC procedure fits a common slopes cumulative model, which is a parallel lines regression model based on the cumulative probabilities of the response categories rather than on their individual probabilities. The cumulative model defines k response functions of the form

$$g(\Pr(Y \le i \mid \mathbf{x})) = \alpha_i + \boldsymbol{\beta}'\mathbf{x}, \quad i = 1, \ldots, k$$

where $\alpha_1, \ldots, \alpha_k$ are k intercept parameters and $\boldsymbol{\beta}$ is a common vector of slope parameters. You can specify cumulative logit, cumulative probit, and cumulative complementary log-log links. These models have been considered by many researchers. Aitchison and Silvey (1957) and Ashford (1959) employ a probit scale and provide a maximum likelihood analysis; Walker and Duncan (1967) and Cox and Snell (1989) discuss the use of the log odds scale. For the log odds scale, the cumulative logit model is often referred to as the *proportional odds* model.

Besides the three preceding cumulative response models, the LOGISTIC procedure provides a fourth ordinal response model, the common slopes *adjacent-category* logit model. This model uses the adjacent-category logit link function, which is based on the individual probabilities of the response categories and has the form

$$\log\left(\frac{\Pr(Y = i \mid \mathbf{x})}{\Pr(Y = i + 1 \mid \mathbf{x})}\right) = \alpha_i + \boldsymbol{\beta}'\mathbf{x}, \quad i = 1, \ldots, k$$

You can use this model when you want to interpret results in terms of the individual response categories instead of cumulative categories.

For nominal response logistic models, where the $k + 1$ possible responses have no natural ordering, the logit model can be extended to a *multinomial* model known as a *generalized* or *baseline-category* logit model, which has the form

$$\log\left(\frac{\Pr(Y = i \mid \mathbf{x})}{\Pr(Y = k + 1 \mid \mathbf{x})}\right) = \alpha_i + \boldsymbol{\beta}_i'\mathbf{x}, \quad i = 1, \ldots, k$$

where the $\alpha_1, \ldots, \alpha_k$ are k intercept parameters and the $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_k$ are k vectors of slope parameters. These models are a special case of the *discrete choice* or *conditional logit* models introduced by McFadden (1974).

The LOGISTIC procedure enables you to relax the parallel lines assumption in ordinal response models, and apply the parallel lines assumption to nominal response models, by specifying parallel line, constrained, and unconstrained parameters as in Peterson and Harrell (1990) and Agresti (2010). The linear predictors for these models have the form

$$\alpha_i + \boldsymbol{\beta}_1'\mathbf{x}_1 + \boldsymbol{\beta}_{2i}'\mathbf{x}_2 + (\boldsymbol{\Gamma}_i\boldsymbol{\beta}_3)'\mathbf{x}_3, \quad i = 1, \ldots, k$$

where the $\boldsymbol{\beta}_1$ are the parallel line (*equal slope*) parameters, the $\boldsymbol{\beta}_{21}, \ldots, \boldsymbol{\beta}_{2k}$ are k vectors of *unequal slope* (*unconstrained*) parameters, and the $\boldsymbol{\beta}_3$ are the *constrained* slope parameters whose constraints are provided by the diagonal $\boldsymbol{\Gamma}_i$ matrix. To fit these models, you specify the EQUALSLOPES and UNEQUALSLOPES options in the MODEL statement. Models that have cumulative logits and both equal and unequal slopes parameters are called *partial proportional odds* models, and models that have only unequal slope parameters are called *general* models.

The LOGISTIC procedure fits linear logistic regression models for discrete response data by the method of maximum likelihood. It can also perform conditional logistic regression for binary response data and exact logistic regression for binary and nominal response data. The maximum likelihood estimation is carried out with either the Fisher scoring algorithm or the Newton-Raphson algorithm, and you can perform the bias-reducing penalized likelihood optimization as discussed by Firth (1993) and Heinze and Schemper (2002). You can specify starting values for the parameter estimates.

Any term specified in the model is referred to as an *effect*. The LOGISTIC procedure enables you to specify categorical variables (also known as *classification* or *CLASS variables*) and continuous variables as explanatory effects. You can also specify more complex model terms such as interactions and nested terms in the same way as in the GLM procedure. You can create complex *constructed effects* with the EFFECT

statement. An effect in the model that is not an interaction or a nested term or a constructed effect is referred to as a *main effect*.

The LOGISTIC procedure allows either a full-rank parameterization or a less-than-full-rank parameterization of the CLASS variables. The full-rank parameterization offers eight coding methods: effect, reference, ordinal, polynomial, and orthogonalizations of these. The effect coding is the same method that is used by default in the CATMOD procedure. The less-than-full-rank parameterization, often called *dummy coding*, is the same coding as that used by default in the GENMOD, GLIMMIX, GLM, HPGENSELECT, and HPLOGISTIC procedures.

The LOGISTIC procedure provides four effect selection methods: forward selection, backward elimination, stepwise selection, and best subset selection. The best subset selection is based on the score statistic. This method identifies a specified number of best models containing one, two, three effects, and so on, up to a single model containing effects for all the explanatory variables.

The LOGISTIC procedure has some additional options to control how to move effects in and out of a model with the forward selection, backward elimination, or stepwise selection model-building strategies. When there are no interaction terms, a main effect can enter or leave a model in a single step based on the p-value of the score or Wald statistic. When there are interaction terms, the selection process also depends on whether you want to preserve model hierarchy. These additional options enable you to specify whether model hierarchy is to be preserved, how model hierarchy is applied, and whether a single effect or multiple effects can be moved in a single step.

Odds ratio estimates are displayed along with parameter estimates. You can also specify the change in the continuous explanatory main effects for which odds ratio estimates are desired. Confidence intervals for the regression parameters and odds ratios can be computed based either on the profile-likelihood function or on the asymptotic normality of the parameter estimators. You can also produce odds ratios for effects that are involved in interactions or nestings, and for any type of parameterization of the CLASS variables.

Various methods to correct for overdispersion are provided, including Williams' method for grouped binary response data. The adequacy of the fitted model can be evaluated by various goodness-of-fit tests, including the Hosmer-Lemeshow test for binary response data.

Like many procedures in SAS/STAT software that enable the specification of CLASS variables, the LOGISTIC procedure provides a CONTRAST statement for specifying customized hypothesis tests concerning the model parameters. The CONTRAST statement also provides estimation of individual rows of contrasts, which is particularly useful for obtaining odds ratio estimates for various levels of the CLASS variables. The LOGISTIC procedure also provides testing capability through the ESTIMATE and TEST statements. Analyses of LS-means are enabled by the LSMEANS, LSMESTIMATE, and SLICE statements, and analyses of predictive margins are enabled by the MARGINS statement.

You can perform a conditional logistic regression on binary response data by specifying the STRATA statement. This enables you to perform matched-set and case-control analyses. The number of events and nonevents can vary across the strata. Many of the features available with the unconditional analysis are also available with a conditional analysis.

The LOGISTIC procedure enables you to perform exact logistic regression, also known as exact conditional logistic regression, by specifying one or more EXACT statements. You can test individual parameters or conduct a joint test for several parameters. The procedure computes two exact tests: the exact conditional score test and the exact conditional probability test. You can request exact estimation of specific parameters and corresponding odds ratios where appropriate. Point estimates, standard errors, and confidence intervals are provided. You can perform stratified exact logistic regression by specifying the STRATA statement.

Further features of the LOGISTIC procedure enable you to do the following:

- control the ordering of the response categories
- compute a generalized R-square measure for the fitted model
- reclassify binary response observations according to their predicted response probabilities
- test linear hypotheses about the regression parameters
- create a data set for producing a receiver operating characteristic (ROC) curve for each fitted model
- specify contrasts to compare several receiver operating characteristic curves
- create a data set containing the estimated response probabilities, residuals, and influence diagnostics
- score a data set by using a previously fitted model
- store the model for input to the PLM procedure

The LOGISTIC procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For more information about the plots implemented in PROC LOGISTIC, see the section "ODS Graphics" on page 6092.

The remaining sections of this chapter describe how to use PROC LOGISTIC and discuss the underlying statistical methodology. The section "Getting Started: LOGISTIC Procedure" on page 5942 introduces PROC LOGISTIC with an example for binary response data. The section "Syntax: LOGISTIC Procedure" on page 5949 describes the syntax of the procedure. The section "Details: LOGISTIC Procedure" on page 6026 summarizes the statistical technique employed by PROC LOGISTIC. The section "Examples: LOGISTIC Procedure" on page 6095 illustrates the use of the LOGISTIC procedure.

For more examples and discussion on the use of PROC LOGISTIC, see Stokes, Davis, and Koch (2012); Allison (1999); SAS Institute Inc. (1995).

Getting Started: LOGISTIC Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/h-l/logigs.sas.)

The LOGISTIC procedure is similar in use to the other regression procedures in the SAS System. To demonstrate the similarity, suppose the response variable y is binary or ordinal, and x1 and x2 are two explanatory variables of interest. To fit a logistic regression model, you can specify a MODEL statement similar to that used in the REG procedure. For example:

```
proc logistic;
   model y=x1 x2;
run;
```

The response variable y can be either character or numeric. PROC LOGISTIC enumerates the total number of response categories and orders the response levels according to the response variable option ORDER= in the MODEL statement.

You can also input binary response data that are grouped. In the following statements, n represents the number of trials and r represents the number of events:

```
proc logistic;
   model r/n=x1 x2;
run;
```

The following example illustrates the use of PROC LOGISTIC. The data, taken from Cox and Snell (1989, pp. 10–11), consist of the number, r, of ingots not ready for rolling, out of n tested, for a number of combinations of heating time and soaking time.

```
data ingots;
   input Heat Soak r n @@;
   datalines;
7 1.0 0 10  14 1.0 0 31  27 1.0 1 56  51 1.0 3 13
7 1.7 0 17  14 1.7 0 43  27 1.7 4 44  51 1.7 0  1
7 2.2 0  7  14 2.2 2 33  27 2.2 0 21  51 2.2 0  1
7 2.8 0 12  14 2.8 0 31  27 2.8 1 22  51 4.0 0  1
7 4.0 0  9  14 4.0 0 19  27 4.0 1 16
;
```

The following invocation of PROC LOGISTIC fits the binary logit model to the grouped data. The continuous covariates Heat and Soak are specified as predictors, and the bar notation ("|") includes their interaction, Heat*Soak. The ODDSRATIO statement produces odds ratios in the presence of interactions, and a graphical display of the requested odds ratios is produced when ODS Graphics is enabled.

```
ods graphics on;
proc logistic data=ingots;
   model r/n = Heat | Soak;
   oddsratio Heat / at(Soak=1 2 3 4);
run;
```

The results of this analysis are shown in the following figures. PROC LOGISTIC first lists background information in Figure 79.1 about the fitting of the model. Included are the name of the input data set, the response variable(s) used, the number of observations used, and the link function used.

Figure 79.1 Binary Logit Model

The LOGISTIC Procedure

Model Information	
Data Set	WORK.INGOTS
Response Variable (Events)	r
Response Variable (Trials)	n
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	19
Number of Observations Used	19
Sum of Frequencies Read	387
Sum of Frequencies Used	387

The "Response Profile" table (Figure 79.2) lists the response categories (which are Event and Nonevent when grouped data are input), their ordered values, and their total frequencies for the given data.

Figure 79.2 Response Profile for Grouped Data with Events/Trials Syntax

Response Profile		
Ordered Value	Binary Outcome	Total Frequency
1	Event	12
2	Nonevent	375

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

The "Model Fit Statistics" table (Figure 79.3) contains Akaike's information criterion (AIC), the Schwarz criterion (SC), and the negative of twice the log likelihood (–2 Log L) for the intercept-only model and the fitted model. AIC and SC can be used to compare different models, and the ones with smaller values are preferred. Results of the likelihood ratio test and the efficient score test for testing the joint significance of the explanatory variables (Soak, Heat, and their interaction) are included in the "Testing Global Null Hypothesis: BETA=0" table (Figure 79.3); the small p-values reject the hypothesis that all slope parameters are equal to zero.

Figure 79.3 Fit Statistics and Hypothesis Tests

Model Fit Statistics			
		Intercept and Covariates	
Criterion	Intercept Only	Log Likelihood	Full Log Likelihood
AIC	108.988	103.222	35.957
SC	112.947	119.056	51.791
-2 Log L	106.988	95.222	27.957

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	11.7663	3	0.0082
Score	16.5417	3	0.0009
Wald	13.4588	3	0.0037

The "Analysis of Maximum Likelihood Estimates" table in Figure 79.4 lists the parameter estimates, their standard errors, and the results of the Wald test for individual parameters. Note that the Heat*Soak parameter is not significantly different from zero (p=0.727), nor is the Soak variable (p=0.6916).

Figure 79.4 Parameter Estimates

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-5.9901	1.6666	12.9182	0.0003
Heat	1	0.0963	0.0471	4.1895	0.0407
Soak	1	0.2996	0.7551	0.1574	0.6916
Heat*Soak	1	-0.00884	0.0253	0.1219	0.7270

The "Association of Predicted Probabilities and Observed Responses" table (Figure 79.5) contains four measures of association for assessing the predictive ability of a model. They are based on the number of pairs of observations with different response values, the number of concordant pairs, and the number of discordant pairs, which are also displayed. Formulas for these statistics are given in the section "Rank Correlation of Observed Responses and Predicted Probabilities" on page 6039.

Figure 79.5 Association Table

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	73.2	**Somers' D**	0.541
Percent Discordant	19.1	**Gamma**	0.586
Percent Tied	7.6	**Tau-a**	0.033
Pairs	4500	**c**	0.771

The ODDSRATIO statement produces the "Odds Ratio Estimates and Wald Confidence Intervals" table (Figure 79.6), and a graphical display of these estimates is shown in Figure 79.7. The differences between the odds ratios are small compared to the variability shown by their confidence intervals, which confirms the previous conclusion that the Heat*Soak parameter is not significantly different from zero.

Figure 79.6 Odds Ratios of Heat at Several Values of Soak

Odds Ratio Estimates and Wald Confidence Intervals			
Odds Ratio	**Estimate**	**95% Confidence Limits**	
Heat at Soak=1	1.091	1.032	1.154
Heat at Soak=2	1.082	1.028	1.139
Heat at Soak=3	1.072	0.986	1.166
Heat at Soak=4	1.063	0.935	1.208

Figure 79.7 Plot of Odds Ratios of Heat at Several Values of Soak

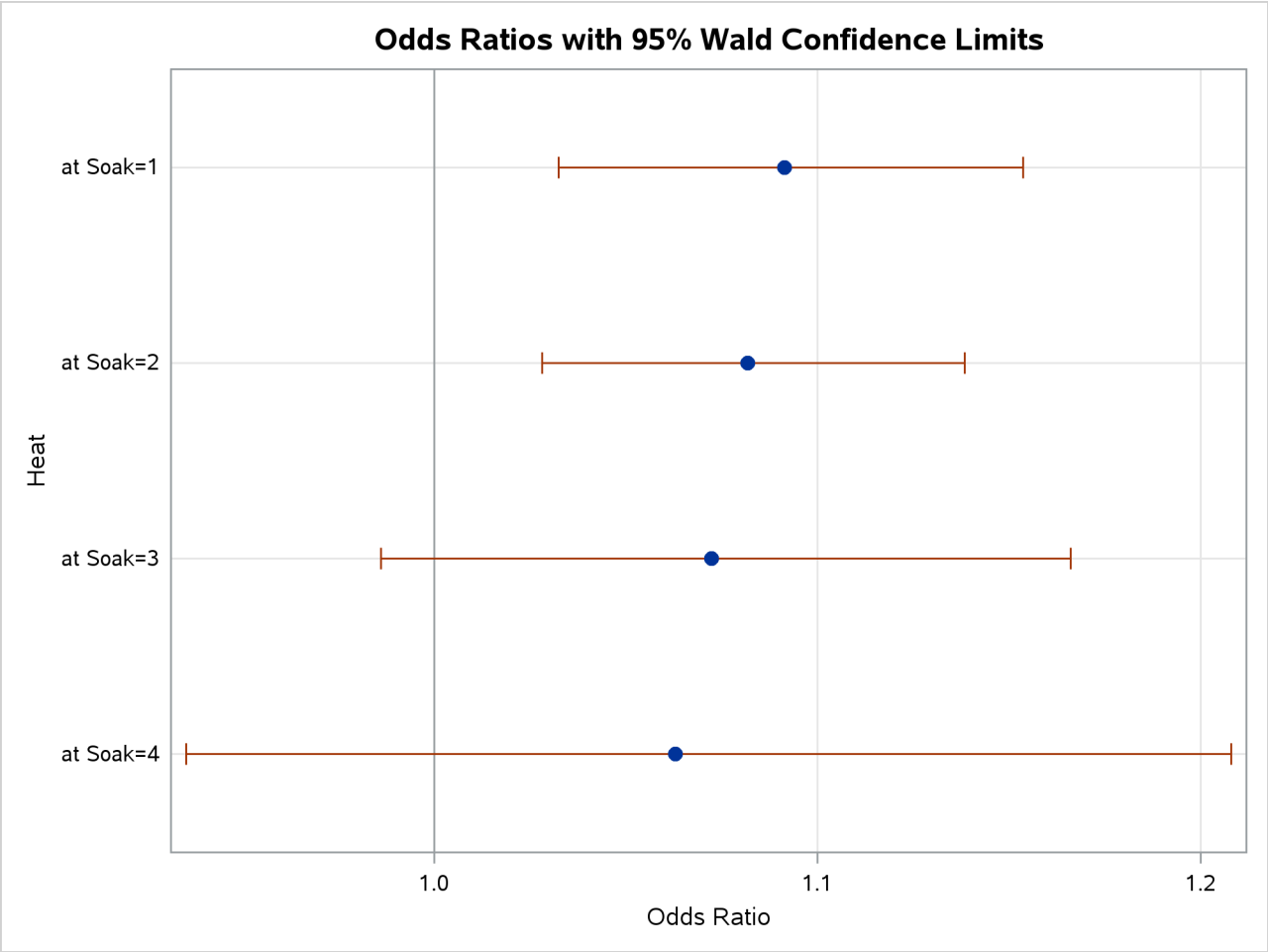

Because the Heat*Soak interaction is nonsignificant, the following statements fit a main-effects model:

```
proc logistic data=ingots;
   model r/n = Heat Soak;
run;
```

The results of this analysis are shown in the following figures. The model information and response profiles are the same as those in Figure 79.1 and Figure 79.2 for the saturated model. The "Model Fit Statistics" table in Figure 79.8 shows that the AIC and SC for the main-effects model are smaller than for the saturated model, indicating that the main-effects model might be the preferred model. As in the preceding model, the "Testing Global Null Hypothesis: BETA=0" table indicates that the parameters are significantly different from zero.

Figure 79.8 Fit Statistics and Hypothesis Tests

The LOGISTIC Procedure

Model Fit Statistics			
		Intercept and Covariates	
Criterion	Intercept Only	Log Likelihood	Full Log Likelihood
AIC	108.988	101.346	34.080
SC	112.947	113.221	45.956
-2 Log L	106.988	95.346	28.080

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	11.6428	2	0.0030
Score	15.1091	2	0.0005
Wald	13.0315	2	0.0015

The "Analysis of Maximum Likelihood Estimates" table in Figure 79.9 again shows that the Soak parameter is not significantly different from zero (p=0.8639). The odds ratio for each effect parameter, estimated by exponentiating the corresponding parameter estimate, is shown in the "Odds Ratios Estimates" table (Figure 79.9), along with 95% Wald confidence intervals. The confidence interval for the Soak parameter contains the value 1, which also indicates that this effect is not significant.

Figure 79.9 Parameter Estimates and Odds Ratios

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-5.5592	1.1197	24.6503	<.0001
Heat	1	0.0820	0.0237	11.9454	0.0005
Soak	1	0.0568	0.3312	0.0294	0.8639

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
Heat	1.085	1.036	1.137
Soak	1.058	0.553	2.026

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	73.0	Somers' D	0.537
Percent Discordant	19.3	Gamma	0.581
Percent Tied	7.6	Tau-a	0.032
Pairs	4500	c	0.769

Using these parameter estimates, you can calculate the estimated logit of π as

$$-5.5592 + 0.082 \times \text{Heat} + 0.0568 \times \text{Soak}$$

For example, if Heat=7 and Soak=1, then $\text{logit}(\widehat{\pi}) = -4.9284$. Using this logit estimate, you can calculate

$\widehat{\pi}$ as follows:

$$\widehat{\pi} = 1/(1 + e^{4.9284}) = 0.0072$$

This gives the predicted probability of the event (ingot not ready for rolling) for Heat=7 and Soak=1. Note that PROC LOGISTIC can calculate these statistics for you; use the OUTPUT statement with the PREDICTED= option, or use the SCORE statement.

To illustrate the use of an alternative form of input data, the following program creates the ingots data set with the new variables NotReady and Freq instead of n and r. The variable NotReady represents the response of individual units; it has a value of 1 for units not ready for rolling (event) and a value of 0 for units ready for rolling (nonevent). The variable Freq represents the frequency of occurrence of each combination of Heat, Soak, and NotReady. Note that, compared to the previous data set, NotReady=1 implies Freq=r, and NotReady=0 implies Freq=n–r.

```
data ingots;
   input Heat Soak NotReady Freq @@;
   datalines;
7 1.0 0 10  14 1.0 0 31  14 4.0 0 19  27 2.2 0 21  51 1.0 1  3
7 1.7 0 17  14 1.7 0 43  27 1.0 1  1  27 2.8 1  1  51 1.0 0 10
7 2.2 0  7  14 2.2 1  2  27 1.0 0 55  27 2.8 0 21  51 1.7 0  1
7 2.8 0 12  14 2.2 0 31  27 1.7 1  4  27 4.0 1  1  51 2.2 0  1
7 4.0 0  9  14 2.8 0 31  27 1.7 0 40  27 4.0 0 15  51 4.0 0  1
;
```

The following statements invoke PROC LOGISTIC to fit the main-effects model by using the alternative form of the input data set:

```
proc logistic data=ingots;
   model NotReady(event='1') = Heat Soak;
   freq Freq;
run;
```

Results of this analysis are the same as the preceding grouped main-effects analysis. The displayed output for the two runs are identical except for the background information of the model fit and the "Response Profile" table shown in Figure 79.10.

Figure 79.10 Response Profile with Single-Trial Syntax

The LOGISTIC Procedure

Response Profile		
Ordered Value	NotReady	Total Frequency
1	0	375
2	1	12

Probability modeled is NotReady=1.

By default, Ordered Values are assigned to the sorted response values in ascending order, and PROC LOGISTIC models the probability of the response level that corresponds to the Ordered Value 1. There are several methods to change these defaults; the preceding statements specify the response variable option EVENT= to model the probability of NotReady=1 as displayed in Figure 79.10. For more information, see the section "Response Level Ordering" on page 6026.

Syntax: LOGISTIC Procedure

The following statements are available in the LOGISTIC procedure:

```
PROC LOGISTIC <options> ;
    BY variables ;
    CLASS variable <(options)> <variable <(options)> ... > </ options> ;
    CODE <options> ;
    CONTRAST 'label' effect values<, effect values, ... > </ options> ;
    EFFECT name=effect-type(variables </ options>) ;
    EFFECTPLOT <plot-type <(plot-definition-options)>> </ options> ;
    ESTIMATE <'label'> estimate-specification </ options> ;
    EXACT <'label'> <INTERCEPT> <effects> </ options> ;
    EXACTOPTIONS options ;
    FREQ variable ;
    ID variables ;
    LSMEANS <model-effects> </ options> ;
    LSMESTIMATE model-effect lsmestimate-specification </ options> ;
    MARGINS model-effects </ options> ;
    <label:> MODEL variable <(variable_options)> = <effects> </ options> ;
    <label:> MODEL events/trials = <effects> </ options> ;
    NLOPTIONS options ;
    ODDSRATIO <'label'> variable </ options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=name <keyword=name ... >> </ option> ;
    ROC <'label'> <specification> </ options> ;
    ROCCONTRAST <'label'> <contrast> </ options> ;
    SCORE <options> ;
    SLICE model-effect </ options> ;
    STORE <OUT=>item-store-name </ LABEL='label'> ;
    STRATA effects </ options> ;
    <label:> TEST equation1 <,equation2, ... > </ option> ;
    UNITS <independent1=list1 <independent2=list2 ... >> </ option> ;
    WEIGHT  variable </ option> ;
```

The PROC LOGISTIC and MODEL statements are required. The CLASS and EFFECT statements (if specified) must precede the MODEL statement, and the CONTRAST, EFFECTPLOT, ESTIMATE, EXACT, LSMEANS, LSMESTIMATE, MARGINS, ROC, SLICE, and STORE statements (if specified) must follow the MODEL statement.

The PROC LOGISTIC, MODEL, and ROCCONTRAST statements can be specified at most once. If a FREQ or WEIGHT statement is specified more than once, the variable specified in the first instance is used. If a BY, OUTPUT, or UNITS statement is specified more than once, the last instance is used.

The rest of this section provides detailed syntax information for each of the preceding statements, beginning with the PROC LOGISTIC statement. The remaining statements are covered in alphabetical order. The CODE, EFFECT, EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, SLICE, and STORE statements are also available in many other procedures. Summary descriptions of functionality and syntax for these statements are provided, but you can find full documentation on them in the corresponding sections of Chapter 20, "Shared Concepts and Topics."

PROC LOGISTIC Statement

PROC LOGISTIC < *options* > **;**

The PROC LOGISTIC statement invokes the LOGISTIC procedure. Optionally, it identifies input and output data sets, suppresses the display of results, and controls the ordering of the response levels. Table 79.1 summarizes the *options* available in the PROC LOGISTIC statement.

Table 79.1 PROC LOGISTIC Statement Options

Option	Description
Input/Output Data Set Options	
COVOUT	Displays the estimated covariance matrix in the OUTEST= data set
DATA=	Names the input SAS data set
INEST=	Specifies the initial estimates SAS data set
INMODEL=	Specifies the model information SAS data set
NOCOV	Does not save covariance matrix in the OUTMODEL= data set
OUTDESIGN=	Specifies the design matrix output SAS data set
OUTDESIGNONLY	Outputs the design matrix only
OUTEST=	Specifies the parameter estimates output SAS data set
OUTMODEL=	Specifies the model output data set for scoring
Response and CLASS Variable Options	
DESCENDING	Reverses the sort order of the response variable
MAXRESPONSELEVELS=	Specifies the maximum number of response levels allowed
NAMELEN=	Specifies the maximum length of effect names
ORDER=	Specifies the sort order of the response variable
TRUNCATE	Truncates class level names
Displayed Output Options	
ALPHA=	Specifies the significance level for confidence intervals
NOPRINT	Suppresses all displayed output
PLOTS	Specifies options for plots
SIMPLE	Displays descriptive statistics
Large Data Set Option	
MULTIPASS	Does not copy the input SAS data set for internal computations
Control of Other Statement Options	
EXACTONLY	Performs exact analysis only
EXACTOPTIONS	Specifies global options for EXACT statements
ROCOPTIONS	Specifies global options for ROC statements

ALPHA=*number*

specifies the level of significance α for $100(1-\alpha)\%$ confidence intervals. The value *number* must be between 0 and 1; the default value is 0.05, which results in 95% intervals. This value is used as the default confidence level for limits computed by the following *options*:

Statement	Options
CONTRAST	ESTIMATE=
EXACT	ESTIMATE=
MODEL	CLODDS= CLPARM=
ODDSRATIO	CL=
OUTPUT	LOWER= UPPER=
PROC LOGISTIC	PLOTS=EFFECT(CLBAR CLBAND)
ROCCONTRAST	ESTIMATE=
SCORE	CLM

You can override the default in most of these cases by specifying the ALPHA= option in the separate statements.

COVOUT

adds the estimated covariance matrix to the OUTEST= data set. For the COVOUT option to have an effect, the OUTEST= option must be specified. See the section "OUTEST= Output Data Set" on page 6076 for more information.

DATA=*SAS-data-set*

names the SAS data set containing the data to be analyzed. If you omit the DATA= option, the procedure uses the most recently created SAS data set. The INMODEL= option cannot be specified with this option.

DESCENDING

DESC

reverses the sort order for the levels of the response variable. If both the DESCENDING and ORDER= options are specified, PROC LOGISTIC orders the levels according to the ORDER= option and then reverses that order. This option has the same effect as the response variable option DESCENDING in the MODEL statement. See the section "Response Level Ordering" on page 6026 for more detail.

EXACTONLY

requests only the exact analyses. The asymptotic analysis that PROC LOGISTIC usually performs is suppressed.

EXACTOPTIONS (*options***)**

specifies options that apply to every EXACT statement in the program. The available *options* are summarized here, and full descriptions are available in the EXACTOPTIONS statement.

Option	Description
ABSFCONV	Specifies the absolute function convergence criterion
ADDTOBS	Adds the observed sufficient statistic to the sampled exact distribution
BUILDSUBSETS	Builds every distribution for sampling
EPSILON=	Specifies the comparison fuzz for partial sums of sufficient statistics
FCONV	Specifies the relative function convergence criterion
MAXTIME=	Specifies the maximum time allowed in seconds
METHOD=	Specifies the DIRECT, NETWORK, NETWORKMC, or MCMC algorithm
N=	Specifies the number of Monte Carlo samples
ONDISK	Uses disk space
SEED=	Specifies the initial seed for sampling
STATUSN=	Specifies the sampling interval for printing a status line
STATUSTIME=	Specifies the time interval for printing a status line
XCONV	Specifies the relative parameter convergence criterion

INEST=*SAS-data-set*

names the SAS data set that contains initial estimates for all the parameters in the model. If BY-group processing is used, it must be accommodated in setting up the INEST= data set. See the section "INEST= Input Data Set" on page 6077 for more information.

INMODEL=*SAS-data-set*

specifies the name of the SAS data set that contains the model information needed for scoring new data. This INMODEL= data set is the OUTMODEL= data set saved in a previous PROC LOGISTIC call. The OUTMODEL= data set should not be modified before its use as an INMODEL= data set.

The DATA= option cannot be specified with this option; instead, specify the data sets to be scored in the SCORE statements. FORMAT statements are not allowed when the INMODEL= data set is specified; variables in the DATA= and PRIOR= data sets in the SCORE statement should be formatted within the data sets.

You can specify the BY statement provided that the INMODEL= data set is created under the same BY-group processing.

The CLASS, EFFECT, EFFECTPLOT, ESTIMATE, EXACT, ID, LSMEANS, LSMESTIMATE, MARGINS, MODEL, OUTPUT, ROC, ROCCONTRAST, SLICE, STORE, TEST, and UNIT statements are not available with the INMODEL= option.

MAXRESPONSELEVELS=*number*

specifies the maximum number of response levels that are allowed in your data set. If you have more response levels than the maximum number allowed, then a message is displayed in the SAS log that provides the value of *number* that is required in order to continue the analysis, and the procedure stops. By default, MAXRESPONSELEVELS=100.

MULTIPASS

forces the procedure to reread the DATA= data set as needed rather than require its storage in memory or in a temporary file on disk. By default, the data set is cleaned up and stored in memory or in a temporary file. This option can be useful for large data sets. All exact analyses are ignored in the presence of the MULTIPASS option. If a STRATA statement is specified, then the data set must first be grouped or sorted by the strata variables.

NAMELEN=*number*
specifies the maximum length of effect names in tables and output data sets to be *number* characters, where *number* is a value between 20 and 200. The default length is 20 characters.

NOCOV
specifies that the covariance matrix not be saved in the OUTMODEL= data set. The covariance matrix is needed for computing the confidence intervals for the posterior probabilities in the OUT= data set in the SCORE statement. Specifying this option will reduce the size of the OUTMODEL= data set.

NOPRINT
suppresses all displayed output. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
RORDER=DATA | FORMATTED | INTERNAL
specifies the sort order for the levels of the response variable. See the response variable option ORDER= in the MODEL statement for more information. For ordering of CLASS variable levels, see the ORDER= option in the CLASS statement.

OUTDESIGN=*SAS-data-set*
specifies the name of the data set that contains the design matrix for the model. The data set contains the same number of observations as the corresponding DATA= data set and includes the response variable (with the same format as in the DATA= data set), the FREQ variable, the WEIGHT variable, the OFFSET= variable, and the design variables for the covariates, including the Intercept variable of constant value 1 unless the NOINT option in the MODEL statement is specified.

OUTDESIGNONLY
suppresses the model fitting and creates only the OUTDESIGN= data set. This option is ignored if the OUTDESIGN= option is not specified.

OUTEST=*SAS-data-set*
creates an output SAS data set that contains the final parameter estimates and, optionally, their estimated covariances (see the preceding COVOUT option). The output data set also includes a variable named _LNLIKE_, which contains the log likelihood. See the section "OUTEST= Output Data Set" on page 6076 for more information.

OUTMODEL=*SAS-data-set*
specifies the name of the SAS data set that contains the information about the fitted model. This data set contains sufficient information to score new data without having to refit the model. It is solely used as the input to the INMODEL= option in a subsequent PROC LOGISTIC call. The OUTMODEL= option is not available with the STRATA statement. Information in this data set is stored in a very compact form, so you should not modify it manually.

NOTE: The STORE statement can also be used to save your model. See the section "STORE Statement" on page 6022 for more information.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >)

controls the plots produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses from around the *plot-request*. For example:

```
PLOTS = ALL
PLOTS = (ROC EFFECT INFLUENCE(UNPACK))
PLOTS(ONLY) = EFFECT(CLBAR SHOWOBS)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc logistic plots=all;
   model y=x;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If the PLOTS option is not specified or is specified with no *plot-requests*, then graphics are produced by default in the following situations:

- If the INFLUENCE or IPLOTS option is specified in the MODEL statement, then the INFLUENCE plots are produced unless the MAXPOINTS= cutoff is exceeded.
- If you specify the OUTROC= option in the MODEL statement, then ROC curves are produced. If you also specify a SELECTION= method, then an overlaid plot of all the ROC curves for each step of the selection process is displayed.
- If the OUTROC= option is specified in a SCORE statement, then the ROC curve for the scored data set is displayed.
- If you specify ROC statements, then an overlaid plot of the ROC curves for the model (or the selected model if a SELECTION= method is specified) and for all the ROC statement models is displayed.
- If you specify the CLODDS= option in the MODEL statement or if you specify the ODDSRATIO statement, then a plot of the odds ratios and their confidence limits is displayed.

For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

The following *global-plot-options* are available:

LABEL

displays a label on diagnostic plots to aid in identifying the outlying observations. This option enhances the plots produced by the DFBETAS, DPC, INFLUENCE, LEVERAGE, and PHAT options. If an ID statement is specified, then the plots are labeled with the ID variables. Otherwise, the observation number is displayed.

MAXPOINTS=NONE | *number*
suppresses the plots produced by the DFBETAS, DPC, INFLUENCE, LEVERAGE, and PHAT options if there are more than *number* observations. Also, observations are not displayed on the EFFECT plots when the cutoff is exceeded. The default is MAXPOINTS=5000. The cutoff is ignored if you specify MAXPOINTS=NONE.

ONLY
displays only specifically requested *plot-requests*.

UNPACKPANELS
UNPACK
suppresses paneling. By default, multiple plots can appear in some output *panels*. Specify UNPACKPANEL to display each plot separately.

The following *plot-requests* are available:

ALL
produces all appropriate plots. You can specify other options with ALL. For example, to display all plots and unpack the DFBETAS plots, you can specify `plots=(all dfbetas(unpack))`.

CALIBRATION< (*calibration-options***) >**
displays calibration plots for the fitted model. For binomial response data, a loess curve is fit to the observed events/trials ratios versus the predicted probabilities. For binary data, an indicator variable is set to 1 if the response is an event and set to 0 otherwise, and a loess curve is fit to this indicator versus the predicted probabilities. For polytomous response data, a panel of plots is produced that displays one plot for each response level as follows: an indicator variable is set to 1 if the observed response equals that level and set to 0 otherwise, and a loess curve is fit to this indicator versus the predicted probability of that level. See Output 79.19.4 for an example of this plot.

You can specify the following *calibration-options*:

ALPHA=*number*
specifies the significance level for constructing two-sided 100(1–*number*)% confidence intervals of the mean predicted values. The value of *number* must be between 0 and 1. The ALPHA= value that you specify in the PROC LOGISTIC statement is the default. If neither ALPHA= value is specified, the default value of 0.05 results in 95% intervals.

CLM
displays confidence limits of the mean predicted values. By default, 95% limits are computed. You can use the ALPHA= option to change the significance level.

RANGE=(*min***,***max***)** | **CLIP**
specifies the range of the axes. The axes might extend beyond your specified values. You can specify *min* and *max* as numbers between 0 and 1; by default, RANGE=(0,1). Specifying the RANGE=CLIP option chooses the smallest range that displays the full loess curve and confidence intervals.

SHOWOBS

displays a bar chart of the predicted probabilities in the upper and lower margins of the plot; the predicted probability axis is divided into 200 bins of equal width. For binary and binomial response data, the upper chart displays frequencies of events, and the lower chart displays frequencies of nonevents. For polytomous response data, for each response level, the upper chart displays frequencies of observations that have an observed response equal to that response level, and the lower chart displays frequencies of observations that have a different observed response from that response level. This option is ignored when a panel of plots is produced.

SMOOTH=*number*

specifies the smoothing parameter, which is the fraction of the data in each local neighborhood for the loess fit. You can specify a value between 0 and 1. By default, SMOOTH=0, which means that the value is chosen automatically.

UNPACKPANELS

UNPACK

displays the plots separately.

DFBETAS < (UNPACK) >

displays plots of DFBETAS versus the case (observation) number. This option is available only for binary and binomial response models. This displays the statistics that are generated by the DFBETAS=_ALL_ option in the OUTPUT statement. The UNPACK option displays the plots separately. See Output 79.6.5 for an example of this plot.

DPC< (*dpc-options***) >**

displays plots of DIFCHISQ and DIFDEV versus the predicted event probability, and displays the markers according to the value of the confidence interval displacement C. This option is available only for binary and binomial response models. See Output 79.6.8 for an example of this plot. You can specify the following *dpc-options*:

MAXSIZE=*Smax*

specifies the maximum size when TYPE=BUBBLE or TYPE=LABEL. For TYPE=BUBBLE, the size is the bubble radius and MAXSIZE=21 by default; for TYPE=LABEL, the size is the font size and MAXSIZE=20 by default. This *dpc-option* is ignored if TYPE=GRADIENT.

MAXVALUE=*Cmax*

displays all observations for which $C \geq Cmax$ at the value of the MAXSIZE= option when TYPE=BUBBLE or TYPE=LABEL. By default, $Cmax = \max_i(C_i)$. This *dpc-option* is ignored if TYPE=GRADIENT.

MINSIZE=*Smin*

specifies the minimum size when TYPE=BUBBLE or TYPE=LABEL. Any observation that maps to a smaller size is displayed at this size. For TYPE=BUBBLE, the size is the bubble radius and MINSIZE=3.5 by default; for TYPE=LABEL, the size is the font size and MINSIZE=2 by default. This *dpc-option* is ignored if TYPE=GRADIENT.

TYPE=BUBBLE | GRADIENT | LABEL

specifies how the C statistic is displayed. You can specify the following values:

BUBBLE displays circular markers whose areas are proportional to C and whose colors are determined by their response.

GRADIENT colors the markers according to the value of C.

LABEL displays the ID variables (if an ID statement is specified) or the observation number. The colors of the ID variable or observation numbers are determined by their response, and their font sizes are proportional to $\frac{C_i}{\max_i(C_i)}$.

By default, TYPE=GRADIENT.

UNPACKPANELS

UNPACK

displays the plots separately.

EFFECT< (*effect-options*) >

displays and enhances the effect plots for the model. For more information about effect plots and the available *effect-options*, see the section "PLOTS=EFFECT Plots" on page 5963.

NOTE: The EFFECTPLOT statement provides much of the same functionality and more options for creating effect plots. See Outputs 79.2.11, 79.3.5, 79.4.8, 79.7.4, and 79.16.4 for examples of effect plots.

INFLUENCE< (UNPACK | STDRES) >

displays index plots of RESCHI, RESDEV, leverage, confidence interval displacements C and CBar, DIFCHISQ, and DIFDEV. This option is available only for binary and binomial response models. These plots are produced by default when any *plot-request* is specified and the MAXPOINTS= cutoff is not exceeded. The UNPACK option displays the plots separately. The STDRES option also displays index plots of STDRESCHI, STDRESDEV, and RESLIK. See Outputs 79.6.3 and 79.6.4 for examples of these plots.

LEVERAGE< (UNPACK) >

displays plots of DIFCHISQ, DIFDEV, confidence interval displacement C, and the predicted probability versus the leverage. This option is available only for binary and binomial response models. The UNPACK option displays the plots separately. See Output 79.6.7 for an example of this plot.

NONE

suppresses all plots.

ODDSRATIO < (*oddsratio-options*) >

displays and enhances the odds ratio plots for the model. For more information about odds ratio plots and the available *oddsratio-options*, see the section "Odds Ratio Plots" on page 5966. See Outputs 79.7,79.2.9, 79.3.3, and 79.4.5 for examples of this plot.

PHAT< (UNPACK) >

displays plots of DIFCHISQ, DIFDEV, confidence interval displacement C, and leverage versus the predicted event probability. This option is available only for binary and binomial response models. The UNPACK option displays the plots separately. See Output 79.6.6 for an example of this plot.

PR< (STEPLEN=*value*) >

displays the precision-recall (PR) curve for binary and binomial response models. If you specify a method in the SELECTION= option, then an overlaid plot of all the PR curves for each step of the selection process is displayed. If you specify ROC statements, then an overlaid plot of the model (or the selected model if you specify the SELECTION= option) and the ROC statement models is displayed. If you specify the OUTROC= option in a SCORE statement, then the PR curve for the scored data set is also displayed.

Because PR curves are remappings of the ROC curve (Davis and Goadrich 2006), the CROSSVALIDATE, EPS, ID, WEIGHTED, METHOD=, THINX=, THINY=, and OPTIMAL suboptions in the ROCOPTIONS option are also applied to PR curves. The PRIOREVENT= and PRIOR= options in the SCORE statement apply to the curves produced by that statement.

The STEPLEN= option specifies the length of a step along the ROC curve, which is used to determine extra points at which the PR curve is computed. For more information, see the section "Precision-Recall Curves" on page 6058. See Output 79.20.3 for an example of this plot.

ROC< (ID< =*keyword* >) >

displays the ROC curve. This option is available only for binary and binomial response models. If you also specify a SELECTION= method, then an overlaid plot of all the ROC curves for each step of the selection process is displayed. If you specify ROC statements, then an overlaid plot of the model (or the selected model if a SELECTION= method is specified) and the ROC statement models is displayed. If you specify the OUTROC= option in a SCORE statement, then the ROC curve for the scored data set is displayed.

The ID= option labels certain points on the ROC curve. Typically, the labeled points are closest to the upper left corner of the plot, and points directly below or to the right of a labeled point are suppressed. This option is identical to, and has the same *keywords* as, the ID= suboption of the ROCOPTIONS option.

You can define the following macro variables to modify the labels and titles on the graphic:

_ROC_ENTRY_ID	sets the note for the ID= option on the ROC plot.
_ROC_ENTRYTITLE	sets the first title line on the ROC plot.
_ROC_ENTRYTITLE2	sets the second title line on the ROC plot.
_ROC_XAXISOPTS_LABEL	sets the X-axis label on the ROC and overlaid ROC plots.
_ROC_YAXISOPTS_LABEL	sets the Y-axis label on the ROC and overlaid ROC plots.
_ROCOVERLAY_ENTRYTITLE	sets the title on the overlaid ROC plot.

To revert to the default labels and titles, you can specify the macro variables in a %SYMDEL statement. For example:

```
%let _ROC_ENTRYTITLE=New Title;
Submit PROC LOGISTIC statement
%symdel _ROC_ENTRYTITLE;
```

See Output 79.8.3 and Example 79.8 for examples of these ROC plots.

ROCOPTIONS (*options*)

specifies options that apply to every model specified in a ROC statement. Some of these *options* also apply to the SCORE statement and to the precision-recall (PR) curves. This option is available only for binary and binomial response models. You can specify the following *options*:

ALPHA=*number*

sets the significance level for creating confidence limits of the areas and the pairwise differences. The ALPHA= value that is specified in the PROC LOGISTIC statement is the default. If neither ALPHA= value is specified, then ALPHA=0.05 by default.

AREA

displays a table of the area under the empirical ROC curve, the area under the convex hull of the empirical ROC curve when METHOD=CONVEXHULL, the area under the binormal ROC curve when METHOD=BINORMAL, the area under the lower ROC curve when METHOD=LOWER, and the area under the PR curve when PLOTS=PR.

CROSSVALIDATE

X

uses cross validated predicted probabilities instead of the model-predicted probabilities for all ROC and area under the ROC curve (AUC) computations and for all PR curve and area computations; for more information, see the section "Classification Table" on page 6047. The cross validated probabilities are also used in computations for the "Association of Predicted Probabilities and Observed Responses" table. If you use a SCORE statement, then the OUTROC= data set and the AUC statistic from the FITSTAT option use the cross validated probabilities only when you score the original data set; otherwise, the model-predicted probabilities are used.

EPS=*value*

is an alias for the ROCEPS= option in the MODEL statement; however, if you also specify that ROCEPS= value, then this option is ignored. This value is used to determine which predicted probabilities are equal. The default value is the square root of the machine epsilon, which is about 1E–8.

ID< =*keyword* **>**

displays labels on certain points on the individual ROC and PR curves and also on the SCORE statement's ROC and PR curves. This option overrides the ID= suboption of the PLOTS=ROC option. If several observations lie at the same place on the curve, the value for the last observation is displayed. If you specify the ID option with no *keyword*, any variables that are listed in the ID statement are used. If no ID statement is specified, the observation number is displayed. The following *keywords* are available:

1MSPEC	displays the false positive fraction (1 – specificity).
FALPOS	displays the fraction of falsely predicted event responses.
FALNEG	displays the fraction of falsely predicted nonevent responses.
ID	displays the ID variables.
MISCLASS	displays the misclassification rate.
NEGPRED	displays the negative predictive value (1 – FALNEG).
OBS	displays the (last) observation number.

OPTSTAT displays the value of the first statistic specified in the OPTIMAL= option.
POSPRED displays the positive predictive value (1 – FALPOS).
PROB displays the model predicted probability.
SENSIT displays the true positive fraction (sensitivity).

The SENSIT, 1MSPEC, POSPRED, and NEGPRED statistics are defined in the section "Receiver Operating Characteristic Curves" on page 6055. The misclassification rate is the number of events that are predicted as nonevents and the number of nonevents that are predicted as events as calculated by using the given cutpoint (predicted probability) divided by the number of observations. If the PEVENT= option is also specified, then POSPRED and NEGPRED are computed using the first PEVENT= value and Bayes' theorem, as discussed in the section "Positive Predictive Values, Negative Predictive Values, and Correct Classification Rates Using Bayes' Theorem" on page 6048.

METHOD=< BINORMAL | CONVEXHULL | EMPIRICAL | LOWER >
specifies the method that is used to estimate the ROC and PR curves. By default, the empirical curves are produced as described in the section "Receiver Operating Characteristic Curves" on page 6055. This is the area estimation method used for ROC comparisons.

The LOWER method modifies the empirical curve by always using a step function created by stepping up at the observed thresholds. This provides a more conservative estimate of the true ROC curve than the empirical estimate.

The CONVEXHULL method modifies the empirical curve by skipping thresholds that could be improved by replacing the observations with a weighted coin flip; it is sometimes called the attainable curve. The convex hull always has a larger AUC than the empirical curve.

The BINORMAL method assumes that the predicted values for your events and for your nonevents both have normal distributions (Metz 1978). If your events are distributed as $N(\mu_e, \sigma_e)$ and your nonevents are distributed as $N(\mu_n, \sigma_n)$, and $\mu_n < \mu_e$, then the binormal curve is given by

$$\text{sensitivity} = \Phi(A + B\Phi^{-1}(1 - \text{specificity}))$$

where

$$A = \frac{\mu_e - \mu_n}{\sigma_n} \quad \text{and} \quad B = \frac{\sigma_e}{\sigma_n}$$

The area under the binormal ROC curve is

$$\Phi\left(\frac{A}{\sqrt{B^2 + 1}}\right)$$

Although the binormal curve is smoother than the empirical curves, it can easily extend below the diagonal and might not be a good fit to your data. The METHOD= option is not available when you specify ROC statements or the ROCCI option, because the required standard errors are based on the empirical ROC curves.

NODETAILS

suppresses the display of the model fitting information for the models specified in the ROC statements.

OPTIMAL<=*statistic* | (*statistics-list*)>

marks observed thresholds on the ROC and PR curves that optimize the given *statistics*, and adds these *statistics* to the OUTROC= data set. Note that there can be more than one threshold that optimizes the statistic. You can specify more than one optimal statistic, and if the same threshold is optimal for several statistics, the marks are jittered. In the following list, let P be the prevalence $\Pr(Y = \text{event})$ of the events in the population; by default, the prevalence is the proportion of events in the training data set n_e/n, where n_e is the frequency of events and n is the total frequency. However, you can change this value by specifying the PEVENT= option in the MODEL statement or the PRIOR= or PRIOREVENT= option in the SCORE statement. For each threshold on the ROC curve, denote the true positive fraction (sensitivity, recall) for that threshold as TPF, the false positive fraction (1 – specificity) as FPF, the true negative fraction (specificity) as TNF, the precision (positive predictive value) as PPV, and the negative predictive value as NPV. The following *statistics* are available; the default is OPTIMAL=YOUDEN:

ACCURACY marks the observed threshold that has the most correct observations. This statistic is the same as EFFICIENCY, but it is computed using the observed event proportions.

COST<(*cost-options***)>** marks the threshold that has the maximum cost criterion $\text{TPF} - \frac{1-P}{P}\frac{C_{fp}-C_{tn}}{C_{fn}-C_{tp}}\text{FPF}$, where C_{tp} is the cost of a true positive, C_{tn} is the cost of a true negative, C_{fp} is the cost of a false positive, C_{fn} is the cost of a false negative, and the quotient $\frac{C_{fp}-C_{tn}}{C_{fn}-C_{tp}}$ is called the cost ratio. This criterion minimizes a cost function of the form $\bar{C} = C_0 + C_{tp}P_{tp} + C_{tn}P_{tn} + C_{fp}P_{fp} + C_{fn}P_{fn}$, where $\bar{C}$ is the average cost, C_0 is a fixed but unspecified cost, $P_{tp} = P \cdot \text{TPF}$ is the probability of a true positive for that threshold, $P_{tn} = (1-P) \cdot \text{TNF}$ is the probability of a true negative, $P_{fp} = (1-P) \cdot \text{FPF}$ is the probability of a false positive, and $P_{fn} = P \cdot \text{FNF}$ is the probability of a false negative (Metz 1978; Zweig and Campbell 1993). You can specify the following *cost-options*:

TP=*cost* is the cost C_{tp} of a true positive.
TN=*cost* is the cost C_{tn} of a true negative.
FP=*cost* is the cost C_{fp} of a false positive.
FN=*cost* is the cost C_{fn} of a false negative.
COSTRATIO=*ratio* | *ratio-list* is the cost ratio $(C_{fp} - C_{tn})/(C_{fn} - C_{tp})$. If you specify the cost ratio, then the other *cost-options* are ignored. You can specify more than one cost ratio as a numeric list of values, and they are identified on the plots and in the OUTROC= data set by appending the cost ratio to the text "_COST".

By default, TP=0, TN=0, FP=1, and FN=1, so that COSTRATIO=1. Note that the observed threshold that minimizes the average cost lies on the convex hull of the ROC curve.

DIST01 marks the threshold where the pair (sensitivity,1 – specificity) is closest to (0,1). This statistic is computed as $\sqrt{(\text{TPF}-1)^2 + \text{FPF}^2}$.

EFFICIENCY marks the threshold that has the maximum efficiency. This statistic is computed as $P \cdot \mathrm{TPF} + (1 - P)\mathrm{TNF}$.

F<(β | *beta-list*)**>** marks the threshold that has the maximum F score (Williams 2021). This statistic is computed as $(1 + \beta^2)\mathrm{PPV} \cdot \mathrm{TPF}/(\beta^2\mathrm{PPV} + \mathrm{TPF})$. Without taking prevalence into account, this is $(1 + \beta^2)n_{11}/((1 + \beta^2)n_{11} + \beta^2 n_{12} + n_{21})$. By default, $\beta = 1$. Other typical choices are $\beta = 0.5$ when precision is more important and $\beta = 2$ when recall is more important. You can specify more than one β value in the *beta-list* as a standard numeric list of values, and they are displayed in parentheses after the "F".

GMEAN marks the threshold that has the largest geometric mean. This statistic is computed as $\sqrt{\mathrm{TPF} \cdot \mathrm{TNF}}$.

MCC marks the threshold that has the maximum Matthews correlation coefficient (phi coefficient). This statistic is computed as $\sqrt{\mathrm{TPF} \cdot \mathrm{TNF} \cdot \mathrm{PPV} \cdot \mathrm{NPV}} - \sqrt{(1 - \mathrm{TPF})\mathrm{FPF}(1 - \mathrm{PPV})(1 - \mathrm{NPV})}$.

MCOST<(*cost-options***)>** marks the threshold that has the largest misclassification cost criterion. This criterion is the same as the COST option's criterion, except that costs are attributed only to misclassification so that $C_{tp} = C_{tn} = 0$. You can specify the following *cost-options*:

FP=*cost* is the cost C_{fp} of a false positive.

FN=*cost* is the cost C_{fn} of a false negative.

COSTRATIO=*ratio* | *ratio-list* is the cost ratio C_{fp}/C_{fn}. If you specify the cost ratio, then the other *cost-options* are ignored. You can specify more than one cost ratio as a numeric list of values, and they are identified on the plots and in the OUTROC= data set by appending the cost ratio to the text "_MCOST".

SSPDIFF marks the threshold where sensitivity and sensitivity are most equal. This statistic is computed as $|\mathrm{TPF} - \mathrm{TNF}|$.

YOUDEN marks the threshold that has the maximum Youden index (Youden 1950). This statistic is computed as $|\mathrm{TPF} - \mathrm{FPF}|$.

OUT=*SAS-data-set-name*

is an alias for the OUTROC= option in the MODEL statement.

THINX=*value*

suppresses labels on the ROC curve for any threshold that is too close to the preceding threshold's label. Let X = sensitivity and Y = 1 – specificity; suppose the point $(x1, y1)$ is labeled; and let the next point be $(x2, y2)$. If both $|x1 - x2| <$ THINX and $|y1 - y2| <$ THINY, then $(x2, y2)$ is not labeled. Similarly, this option suppresses labels on the PR curve for any threshold that is too close to the preceding threshold's label, where Y = precision and X = recall. Optimal threshold labels are never thinned. By default, THINX=0.05.

THINY=*value*

suppresses labels on the ROC and PR curves for any threshold that is too close to the preceding threshold's label. See the THINX= option for details. Optimal threshold labels are never thinned. By default, THINY=0.05.

WEIGHTED

uses frequency × weight in the ROC and PR computations (Izrael et al. 2002) instead of just frequency. Typically, weights are considered in the fit of the model only, and hence they are accounted for in the parameter estimates. The "Association of Predicted Probabilities and Observed Responses" table uses frequency (unless the BINWIDTH=0 option is also specified in the MODEL statement) and is suppressed when ROC comparisons are performed. This option also affects computations of the ROC and PR curves performed by the SCORE statement, as well as computations of all areas under the curves.

SIMPLE

displays simple descriptive statistics (mean, standard deviation, minimum, and maximum) for each continuous explanatory variable. For each CLASS variable involved in the modeling, the frequency counts of the classification levels are displayed. The SIMPLE option generates a breakdown of the simple descriptive statistics or frequency counts for the entire data set and also for individual response categories.

TRUNCATE

determines class levels by using no more than the first 16 characters of the formatted values of CLASS, response, and strata variables. This option invokes the same option in the CLASS statement.

PLOTS=EFFECT Plots

Only one PLOTS=EFFECT plot is produced by default; you must specify other *effect-options* to produce multiple plots. For binary response models, the following plots are produced when an EFFECT option is specified with no *effect-options*:

- If you have only continuous covariates in the model, then a plot is displayed of the predicted probability versus the first continuous covariate, fixing all other continuous covariates at their means. See Output 79.7.4 for an example with one continuous covariate.
- If you have only classification covariates in the model, then a plot is displayed of the predicted probability versus the first CLASS covariate at each level of the second CLASS covariate, if any, holding all other CLASS covariates at their reference levels. If you have exactly two CLASS covariates, and one is nested in the other, then the nested covariate is clustered within the levels of the other covariate.
- If you have both classification and continuous covariates in the model, then a plot is displayed of the predicted probability versus the first continuous covariate at up to 10 cross-classifications of the CLASS covariate levels, fixing all other continuous covariates at their means and all other CLASS covariates at their reference levels. For example, if your model has four binary covariates, there are 16 cross-classifications of the CLASS covariate levels. The plot displays the 8 cross-classifications of the levels of the first three covariates, and the fourth covariate is fixed at its reference level.

For polytomous response models, similar plots are produced by default, except that the response levels are used in place of the CLASS covariate levels. Plots for polytomous response models involving OFFSET= variables with multiple values are not available.

The following *effect-options* specify the type of graphic to produce:

AT(*variable=value-list* | **ALL<** . . . *variable=value-list* | **ALL >)**

specifies fixed values for a covariate. For continuous covariates, you can specify one or more numbers in the *value-list*. For classification covariates, you can specify one or more formatted levels of the covariate enclosed in single quotes (for example, **`A='cat' 'dog'`**), or you can specify the keyword ALL to select all levels of the classification variable. You can specify a variable at most once in the AT option. By default, continuous covariates are set to their means when they are not used on an axis, while classification covariates are set to their reference level when they are not used as an X=, SLICEBY=, or PLOTBY= effect. For example, for a model that includes a classification variable A={cat,dog} and a continuous covariate X, specifying **`AT(A='cat' X=7 9)`** will set A to 'cat' when A does not appear in the plot. When X does not define an axis it first produces plots setting $X = 7$ and then produces plots setting $X = 9$. Note in this example that specifying **`AT( A=ALL )`** is the same as specifying the PLOTBY=A option.

FITOBSONLY

computes the predicted values only at the observed data. If the FITOBSONLY option is omitted and the X-axis variable is continuous, the predicted values are computed at a grid of points extending slightly beyond the range of the data (see the EXTEND= option for more information). If the FITOBSONLY option is omitted and the X-axis effect is categorical, the predicted values are computed at all possible categories.

INDIVIDUAL

displays the individual probabilities instead of the cumulative probabilities. This option is available only with cumulative models, and it is not available with the LINK option.

LINK

displays the linear predictors instead of the probabilities on the Y axis. For example, for a binary logistic regression, the Y axis will be displayed on the logit scale. The INDIVIDUAL and POLYBAR options are not available with the LINK option.

PLOTBY=*effect*

displays an effect plot at each unique level of the PLOTBY= effect. You can specify *effect* as one CLASS variable or as an interaction of classification covariates. For polytomous response models, you can also specify the response variable as the lone PLOTBY= effect. For nonsingular parameterizations, the complete cross-classification of the CLASS variables specified in the effect define the different PLOTBY= levels. When the GLM parameterization is used, the PLOTBY= levels can depend on the model and the data.

SLICEBY=*effect*

displays predicted probabilities at each unique level of the SLICEBY= effect. You can specify *effect* as one CLASS variable or as an interaction of classification covariates. For polytomous response models, you can also specify the response variable as the lone SLICEBY= effect. For nonsingular parameterizations, the complete cross-classification of the CLASS variables specified in the effect define the different SLICEBY= levels. When the GLM parameterization is used, the SLICEBY= levels can depend on the model and the data.

X=*effect*

X=(*effect. . . effect***)**

specifies effects to be used on the X axis of the effect plots. You can specify several different X axes: continuous variables must be specified as main effects, while CLASS variables can be crossed. For nonsingular parameterizations, the complete cross-classification of the CLASS variables specified in the effect define the axes. When the GLM parameterization is used, the X= levels can depend on the model and the data.

NOTE: Any variable not specified in a SLICEBY= or PLOTBY= option is available to be displayed on the X axis. A variable can be specified in at most one of the SLICEBY=, PLOTBY=, and X= options.

The following *effect-options* enhance the graphical output:

ALPHA=*number*

specifies the size of the confidence limits. The ALPHA= value specified in the PROC LOGISTIC statement is the default. If neither ALPHA= value is specified, then ALPHA=0.05 by default.

CLBAND< =YES | NO >

displays confidence limits on the plots. This option is not available with the INDIVIDUAL option. If you have CLASS covariates on the X axis, then error bars are displayed (see the CLBAR option) unless you also specify the CONNECT option.

CLBAR

displays the error bars on the plots when you have CLASS covariates on the X axis; if the X axis is continuous, then this invokes the CLBAND option. For polytomous response models with CLASS covariates only and with the POLYBAR option specified, the stacked bar charts are replaced by side-by-side bar charts with error bars.

CLUSTER< =*percent* **>**

displays the levels of the SLICEBY= effect in a side-by-side fashion instead of stacking them. This option is available when you have CLASS covariates on the X axis. You can specify *percent* as a percentage of half the distance between X levels. The *percent* value must be between 0.1 and 1; the default *percent* depends on the number of X levels and the number of levels of the SLICEBY= effect. Default clustering can be removed by specifying the NOCLUSTER option.

CLUSTERGRID< =YES | NO >

separates the CLASS covariate levels on the X axis with grid lines when the levels of the SLICEBY= effect are clustered.

CONNECT< =YES | NO >

JOIN< =YES | NO >

connects the predicted values with a line. This option is available when you have CLASS covariates on the X axis. You can suppress default connecting lines by specifying the NOCONNECT option.

EXTEND=*value*

extends continuous X axes by a factor of *value*/2 in each direction. By default, EXTEND=0.2.

MAXATLEN=*length*
specifies the maximum number of characters to use to display the levels of all the fixed variables. If the text is too long, it is truncated and ellipses (“. . . ”) are appended. By default, *length* is equal to its maximum allowed value, 256.

NOCLUSTER
prevents clustering of the levels of the SLICEBY= effect. This option is available when you have CLASS covariates on the X axis.

NOCLUSTERGRID
removes the default grid lines that separate the levels of the CLASS covariates on the X axis when the SLICEBY= effect is clustered.

NOCONNECT
removes the line that connects the predicted values. This option is available when you have CLASS covariates on the X axis.

POLYBAR
replaces scatter plots of polytomous response models with bar charts. This option has no effect on binary-response models, and it is overridden by the CONNECT option. By default, the X axis is chosen to be a crossing of available classification variables so that there are no more than 16 levels; if no such crossing is possible then the first available classification variable is used. You can override this default by specifying the X= option.

SHOWOBS< =YES | NO >
displays observations on the plot when the MAXPOINTS= cutoff is not exceeded. For events/trials notation, the observed proportions are displayed; for single-trial binary-response models, the observed events are displayed at $\hat{p} = 1$ and the observed nonevents are displayed at $\hat{p} = 0$. For polytomous response models the predicted probabilities at the observed values of the covariate are computed and displayed.

YRANGE=(< *min* >< ,*max* >)
displays the Y axis as [*min*,*max*]. Note that the axis might extend beyond your specified values. By default, the entire Y axis, [0,1], is displayed for the predicted probabilities. This option is useful if your predicted probabilities are all contained in some subset of this range.

Odds Ratio Plots

The odds ratios and confidence limits from the default “Odds Ratio Estimates” table and from the tables produced by the CLODDS= option or the ODDSRATIO statement can be displayed in a graphic. If you have many odds ratios, you can produce multiple graphics, or *panels*, by displaying subsets of the odds ratios. Odds ratios that have duplicate labels are not displayed. See Outputs 79.2.9 and 79.3.3 for examples of odds ratio plots.

For continuous variables that are used to construct SPLINE effects, and for continuous variables of second or higher degree in the model, odds ratios depend both on the units of change and on the value of the variable at which the change is computed (the “starting” or “base” value). A special plot that shows the odds ratios along with base values that span the range of such a continuous variable is also displayed when you specify that variable in an ODDSRATIO statement. See Output 79.7.5 for an example.

The following *oddsratio-options* modify the default odds ratio plot:

CLDISPLAY=SERIF | SERIFARROW | LINE | LINEARROW | BAR< *width* > **| NONE**

controls the look of the confidence limits. The default CLDISPLAY=SERIF option displays the confidence limits as lines with serifs, and the CLDISPLAY=LINE option removes the serifs from the error bars. The CLDISPLAY=SERIFARROW and CLDISPLAY=LINEARROW options display arrowheads on any error bars that are clipped by the RANGE= option; if the entire error bar is cut from the graphic, then an arrowhead is displayed that points toward the odds ratio. The CLDISPLAY=BAR < *width* > option displays the limits along with a bar whose width is equal to the size of the marker. You can control the width of the bars and the size of the marker by specifying the *width* value as a percentage of the distance between the bars, $0 < \mathit{width} \leq 1$. The CLDISPLAY=NONE option suppresses the display of the error bars and, for the special plots of continuous variables, suppresses the display of the confidence bands.

NOTE: Your bar might disappear if you have small values of *width*.

DOTPLOT

displays dotted gridlines on the plot.

GROUP

displays the odds ratios in panels that are defined by the ODDSRATIO statements. The NPANELPOS= option is ignored when this option is specified.

LOGBASE=2 | E | 10

displays the odds ratio axis on the specified log scale.

NAT=*number*

specifies the number of starting values at which to compute the special plots of odds ratios of continuous variables. The points are evenly spread across the range of the odds ratio variable. The *number* must be an integer greater than 1. By default, NAT=50.

NPANELPOS=*number*

breaks the plot into multiple graphics that have at most |*number*| odds ratios per graphic. If *number* is positive, then the number of odds ratios per graphic is balanced; if *number* is negative, then no balancing of the number of odds ratios takes place. By default, *number* = 0 and all odds ratios are displayed in a single plot. For example, suppose you want to display 21 odds ratios. Then specifying NPANELPOS=20 displays two plots, the first with 11 odds ratios and the second with 10; but specifying NPANELPOS=–20 displays 20 odds ratios in the first plot and only 1 odds ratio in the second plot.

ORDER=ASCENDING | DESCENDING

displays the odds ratios in sorted order. By default the odds ratios are displayed in the order in which they appear in the corresponding table.

RANGE=(< *min* >< **,***max* >**) | CLIP**

specifies the range of the displayed odds ratio axis. Specifying the RANGE=CLIP option has the same effect as specifying the minimum odds ratio as *min* and the maximum odds ratio as *max*. By default, all odds ratio confidence intervals are displayed.

TYPE=HORIZONTAL | HORIZONTALSTAT | VERTICAL | VERTICALBLOCK

controls the look of the graphic. The default TYPE=HORIZONTAL option places the odds ratio values on the X axis, while the TYPE=HORIZONTALSTAT option also displays the values of the odds ratios and their confidence limits on the right side of the graphic. The TYPE=VERTICAL option places the odds ratio values on the Y axis, while the TYPE=VERTICALBLOCK option (available only with the CLODDS= option) places the odds ratio values on the Y axis and puts boxes around the labels.

For the special plots of odds ratios of continuous variables, the TYPE=HORIZONTAL and HORIZONTALSTAT options produce the same plot and display the odds ratios on the X axis and the starting values on the Y axis. The TYPE=VERTICAL and VERTICALBLOCK options produce the same plot and display the odds ratios on the Y axis and the starting values on the X axis.

UNPACKPANELS

UNPACK

displays plots separately when the default display shows multiple plots together as a panel of plots.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC LOGISTIC to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the LOGISTIC procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

If a SCORE statement is specified, then define the *training data set* to be the DATA= data set or the INMODEL= data set in the PROC LOGISTIC statement, and define the *scoring data set* to be the DATA= data set and PRIOR= data set in the SCORE statement. The training data set contains all of the BY variables, and the scoring data set must contain either all of them or none of them. If the scoring data set contains all the BY variables, matching is carried out between the training and scoring data sets. If the scoring data set does not contain any of the BY variables, the entire scoring data set is used for every BY group in the training data set and the BY variables are added to the output data sets that are specified in the SCORE statement.

CAUTION: The order of the levels in the response and classification variables is determined from all the data regardless of BY groups. However, different sets of levels might appear in different BY groups. This might affect the value of the reference level for these variables, and hence your interpretation of the model and the parameters.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *global-options* > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. Response variables do not need to be specified in the CLASS statement.

The CLASS statement must precede the MODEL statement. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*. You can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING

DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC LOGISTIC orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (., ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values of the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so this option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC LOGISTIC interprets values of the ORDER= option:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. You can specify any of the *keywords* shown in the following table. The default is PARAM=EFFECT. Design matrix columns are created from CLASS variables according to the corresponding coding schemes.

Value of PARAM=	Coding
EFFECT	Effect coding
GLM	Less-than-full-rank reference cell coding (this *keyword* can be used only in a global option)
ORDINAL **THERMOMETER**	Cumulative parameterization for an ordinal CLASS variable
POLYNOMIAL **POLY**	Polynomial coding
REFERENCE **REF**	Reference cell coding
ORTHEFFECT	Orthogonalizes PARAM=EFFECT coding
ORTHORDINAL **ORTHOTHERM**	Orthogonalizes PARAM=ORDINAL coding
ORTHPOLY	Orthogonalizes PARAM=POLYNOMIAL coding
ORTHREF	Orthogonalizes PARAM=REFERENCE coding

All parameterizations are full rank, except for the GLM parameterization. The REF= option in the CLASS statement determines the reference level for EFFECT and REFERENCE coding and for their orthogonal parameterizations. It also indirectly determines the reference level for a singular GLM parameterization through the order of levels.

If a PARAM= option is specified as a variable option for some variables, then any variables for which PARAM= is not specified use either the EFFECT parameterization if the global PARAM= option is not specified, or the full-rank parameterization indicated in the global PARAM= option if specified. If the global PARAM=GLM option is specified and PARAM= is also specified for some variables, GLM parameterization is used for all variables.

If PARAM=ORTHPOLY or PARAM=POLY and the classification variable is numeric, then the ORDER= option in the CLASS statement is ignored, and the internal unformatted values are used. For more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

REF=*'level'* | *keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*:

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n*>

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

Class Variable Naming Convention

Parameter names for a CLASS predictor variable are constructed by concatenating the CLASS variable name with the CLASS levels. However, for the POLYNOMIAL and orthogonal parameterizations, parameter names are formed by concatenating the CLASS variable name and keywords that reflect the parameterization. For examples and more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

Class Variable Parameterization with Unbalanced Designs

PROC LOGISTIC initially parameterizes the CLASS variables by looking at the levels of the variables across the complete data set. If you have an *unbalanced* replication of levels across variables or BY groups, then the design matrix and the parameter interpretation might be different from what you expect. For example, suppose you have a model that has one CLASS variable A with three levels (1, 2, and 3) and another CLASS variable B with two levels (1 and 2). If the third level of A occurs only with the first level of B, if you use the EFFECT parameterization, and if your model contains the effect A(B) and an intercept, then the design for A within the second level of B is not a differential effect. In particular, the design looks like the following:

		Design Matrix			
		A(B=1)		A(B=2)	
B	A	A1	A2	A1	A2
1	1	1	0	0	0
1	2	0	1	0	0
1	3	–1	–1	0	0
2	1	0	0	1	0
2	2	0	0	0	1

PROC LOGISTIC detects linear dependency among the last two design variables and sets the parameter for A2(B=2) to zero, resulting in an interpretation of these parameters as if they were reference- or dummy-coded. The REFERENCE or GLM parameterization might be more appropriate for such problems.

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 79.2 summarizes the *options* available in the CODE statement.

Table 79.2 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

CONTRAST Statement

CONTRAST '*label*' *row-description*<**,** . . . **,** *row-description*> <**/** *options*> **;**

where a *row-description* is defined as follows:

effect values<**,** . . . **,** *effect values*>

The CONTRAST statement provides a mechanism for obtaining customized hypothesis tests. It is similar to the CONTRAST and ESTIMATE statements in other modeling procedures.

The CONTRAST statement enables you to specify a matrix, $\mathbf{L}$, for testing the hypothesis $\mathbf{L}\boldsymbol{\beta} = \mathbf{0}$, where $\boldsymbol{\beta}$ is the vector of intercept and slope parameters. You must be familiar with the details of the model parameterization that PROC LOGISTIC uses (for more information, see the PARAM= option in the section "CLASS Statement" on page 5969). Optionally, the CONTRAST statement enables you to estimate each row, $\boldsymbol{l}_i'\boldsymbol{\beta}$, of $\mathbf{L}\boldsymbol{\beta}$ and test the hypothesis $\boldsymbol{l}_i'\boldsymbol{\beta} = 0$. Computed statistics are based on the asymptotic chi-square distribution of the Wald statistic.

There is no limit to the number of CONTRAST statements that you can specify, but they must appear after the MODEL statement.

The following parameters are specified in the CONTRAST statement:

label identifies the contrast in the displayed output. A label is required for every contrast specified, and it must be enclosed in quotation marks.

effect identifies an effect that appears in the MODEL statement. The name INTERCEPT can be used as an effect when one or more intercepts are included in the model. You do not need to include all effects that are included in the MODEL statement.

values are constants that are elements of the $\mathbf{L}$ matrix associated with the effect. To correctly specify your contrast, it is crucial to know the ordering of parameters within each effect and the variable levels associated with any parameter. The "Class Level Information" table shows the ordering of levels within variables. The E option, described later in this section, enables you to verify the proper correspondence of *values* to parameters. If too many values are specified for an effect, the extra ones are ignored. If too few values are specified, the remaining ones are set to 0.

Multiple degree-of-freedom hypotheses can be tested by specifying multiple *row-descriptions*; the rows of $\mathbf{L}$ are specified in order and are separated by commas. The degrees of freedom is the number of linearly independent constraints implied by the CONTRAST statement—that is, the rank of $\mathbf{L}$.

More details for specifying contrasts involving effects with full-rank parameterizations are given in the section "Full-Rank Parameterized Effects" on page 5974, while details for less-than-full-rank parameterized effects are given in the section "Less-Than-Full-Rank Parameterized Effects" on page 5975.

You can specify the following *options* after a slash (/):

ALPHA=*number*
specifies the level of significance α for the $100(1-\alpha)\%$ confidence interval for each contrast when the ESTIMATE option is specified. The value of *number* must be between 0 and 1. By default, *number* is equal to the value of the ALPHA= option in the PROC LOGISTIC statement, or 0.05 if that option is not specified.

E

displays the **L** matrix.

ESTIMATE=*keyword*

estimates and tests each individual contrast (that is, each row, $\boldsymbol{l}_i'\boldsymbol{\beta}$, of $\mathbf{L}\boldsymbol{\beta}$), exponentiated contrast ($e^{\boldsymbol{l}_i'\boldsymbol{\beta}}$), or predicted probability for the contrast ($g^{-1}(\boldsymbol{l}_i'\boldsymbol{\beta})$). PROC LOGISTIC displays the point estimate, its standard error, a Wald confidence interval, and a Wald chi-square test. The significance level of the confidence interval is controlled by the ALPHA= option. You can estimate the individual contrast, the exponentiated contrast, or the predicted probability for the contrast by specifying one of the following *keywords*:

PARM estimates the individual contrast.
EXP estimates the exponentiated contrast.
BOTH estimates both the individual contrast and the exponentiated contrast.
PROB estimates the predicted probability of the contrast.
ALL estimates the individual contrast, the exponentiated contrast, and the predicted probability of the contrast.

For more information about the computations of the standard errors and confidence limits, see the section "Linear Predictor, Predicted Probability, and Confidence Limits" on page 6046.

SINGULAR=*number*

tunes the estimability check. This option is ignored when a full-rank parameterization is specified. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the largest absolute value of the elements of $\mathbf{v}$. For a row vector $\boldsymbol{l}'$ of the contrast matrix $\mathbf{L}$, define $c = \text{ABS}(\boldsymbol{l})$ if ABS($\boldsymbol{l}$) is greater than 0; otherwise, $c = 1$. If $\text{ABS}(\boldsymbol{l}' - \boldsymbol{l}'\mathbf{T})$ is greater than $c * \textit{number}$, then $\boldsymbol{l}$ is declared nonestimable. The $\mathbf{T}$ matrix is the Hermite form matrix $\mathbf{I}_0^-\mathbf{I}_0$, where $\mathbf{I}_0^-$ represents a generalized inverse of the (observed or expected) information matrix $\mathbf{I}_0$ of the null model. The value for *number* must be between 0 and 1; the default value is 1E–4.

Full-Rank Parameterized Effects

If an effect involving a CLASS variable with a full-rank parameterization does not appear in the CONTRAST statement, then all of its coefficients in the **L** matrix are set to 0.

If you use effect coding by default or by specifying PARAM=EFFECT in the CLASS statement, then all parameters are directly estimable and involve no other parameters. For example, suppose an effect-coded CLASS variable A has four levels. Then there are three parameters ($\beta_1, \beta_2, \beta_3$) representing the first three levels, and the fourth parameter is represented by

$$-\beta_1 - \beta_2 - \beta_3$$

To test the first versus the fourth level of A, you would test

$$\beta_1 = -\beta_1 - \beta_2 - \beta_3$$

or, equivalently,

$$2\beta_1 + \beta_2 + \beta_3 = 0$$

which, in the form $\mathbf{L}\boldsymbol{\beta} = 0$, is

$$\begin{bmatrix} 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} = 0$$

Therefore, you would use the following CONTRAST statement:

```
contrast '1 vs. 4' A 2 1 1;
```

To contrast the third level with the average of the first two levels, you would test

$$\frac{\beta_1 + \beta_2}{2} = \beta_3$$

or, equivalently,

$$\beta_1 + \beta_2 - 2\beta_3 = 0$$

Therefore, you would use the following CONTRAST statement:

```
contrast '1&2 vs. 3' A 1 1 -2;
```

Other CONTRAST statements are constructed similarly. For example:

```
contrast '1 vs. 2    '  A  1 -1  0;
contrast '1&2 vs. 4  '  A  3  3  2;
contrast '1&2 vs. 3&4'  A  2  2  0;
contrast 'Main Effect'  A  1  0  0,
                        A  0  1  0,
                        A  0  0  1;
```

Less-Than-Full-Rank Parameterized Effects

When you use the less-than-full-rank parameterization (by specifying PARAM=GLM in the CLASS statement), each row is checked for estimability; see the section "Estimable Functions" on page 60 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," for more information. If PROC LOGISTIC finds a contrast to be nonestimable, it displays missing values in corresponding rows in the results. PROC LOGISTIC handles missing level combinations of classification variables in the same manner as PROC GLM: parameters corresponding to missing level combinations are not included in the model. This convention can affect the way in which you specify the **L** matrix in your CONTRAST statement. If the elements of **L** are not specified for an effect that contains a specified effect, then the elements of the specified effect are distributed over the levels of the higher-order effect just as the GLM procedure does for its CONTRAST and ESTIMATE statements. For example, suppose that the model contains effects A and B and their interaction A*B. If you specify a CONTRAST statement involving A alone, the **L** matrix contains nonzero terms for both A and A*B, because A*B contains A. For more information, see rule 4 in the section "Construction of Least Squares Means" on page 4244 in Chapter 53, "The GLM Procedure."

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 79.3 summarizes the *options* available in the EFFECT statement.

Table 79.3 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.

Table 79.3 *continued*

Option	Description
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

EFFECTPLOT Statement

EFFECTPLOT < *plot-type* < *(plot-definition-options)* > > < / *options* > **;**

The EFFECTPLOT statement produces a display of the fitted model and provides options for changing and enhancing the displays. Table 79.4 describes the available *plot-types* and their *plot-definition-options*.

Table 79.4 *Plot-Types* and *Plot-Definition-Options*

Plot-Type and Description	Plot-Definition-Options
BOX	
Displays a box plot of continuous response data at each level of a CLASS effect, with predicted values superimposed and connected by a line. This is an alternative to the INTERACTION *plot-type*.	PLOTBY= variable or CLASS effect X= CLASS variable or effect
CONTOUR	
Displays a contour plot of predicted values against two continuous covariates.	PLOTBY= variable or CLASS effect X= continuous variable Y= continuous variable

Table 79.4 *continued*

Plot-Type and Description	Plot-Definition-Options
FIT Displays a curve of predicted values versus a continuous variable.	PLOTBY= variable or CLASS effect X= continuous variable
INTERACTION Displays a plot of predicted values (possibly with error bars) versus the levels of a CLASS effect. The predicted values are connected with lines and can be grouped by the levels of another CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= CLASS variable or effect
MOSAIC Displays a mosaic plot of predicted values using up to three CLASS effects.	PLOTBY= variable or CLASS effect X= CLASS effects
SLICEFIT Displays a curve of predicted values versus a continuous variable grouped by the levels of a CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= continuous variable

For full details about the syntax and options of the EFFECTPLOT statement, see the section "EFFECTPLOT Statement" on page 447 in Chapter 20, "Shared Concepts and Topics."

See Outputs 79.2.11, 79.2.12, 79.3.5, 79.4.8, 79.7.4, and 79.16.4 for examples of plots produced by this statement.

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 79.5 summarizes the *options* available in the ESTIMATE statement.

Table 79.5 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference

Table 79.5 *continued*

Option	Description
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected *p*-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the **L** matrix
JOINT	Produces a joint *F* or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement" on page 475 in Chapter 20, "Shared Concepts and Topics."

EXACT Statement

EXACT <'*label*'> <**INTERCEPT**> <*effects*> <*/ options*> **;**

The EXACT statement performs exact tests of the parameters for the specified *effects* and optionally estimates the parameters and outputs the exact conditional distributions. You can specify the keyword INTERCEPT and any *effects* in the MODEL statement. Inference on the parameters of the specified effects is performed by conditioning on the sufficient statistics of all the other model parameters (possibly including the intercept).

You can specify several EXACT statements, but they must follow the MODEL statement. Each statement can optionally include an identifying *label*. If several EXACT statements are specified, any statement without a label is assigned a label of the form "Exact*n*," where *n* indicates the *n*th EXACT statement. The label is included in the headers of the displayed exact analysis tables.

If a STRATA statement is also specified, then a stratified exact logistic regression is performed. The model contains a different intercept for each stratum, and these intercepts are conditioned out of the model along

with any other nuisance parameters (parameters for effects specified in the MODEL statement that are not in the EXACT statement).

If the LINK=GLOGIT option is specified in the MODEL statement, then the METHOD=DIRECT option is invoked in the EXACTOPTIONS statement by default and a generalized logit model is fit. Since each effect specified in the MODEL statement adds k parameters to the model (where $k + 1$ is the number of response levels), exact analysis of the generalized logit model by using this method is limited to rather small problems.

The CONTRAST, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, ODDSRATIO, OUTPUT, ROC, ROCCONTRAST, SCORE, SLICE, STORE, TEST, and UNITS statements are not available with an exact analysis; results from these statements are based on the asymptotic results. Exact analyses are not performed when you specify a WEIGHT statement, a link other than LINK=LOGIT or LINK=GLOGIT, an offset variable, the NOFIT option, or a model selection method. Exact estimation is not available for ordinal response models.

For classification variables, use of the reference parameterization is recommended.

The following *options* can be specified in each EXACT statement after a slash (/):

ALPHA=*number*

specifies the level of significance α for $100(1-\alpha)\%$ confidence limits for the parameters or odds ratios. The value of *number* must be between 0 and 1. By default, *number* is equal to the value of the ALPHA= option in the PROC LOGISTIC statement, or 0.05 if that option is not specified.

CLTYPE=EXACT | MIDP

requests either the exact or mid-p confidence intervals for the parameter estimates. By default, the exact intervals are produced. The confidence coefficient can be specified with the ALPHA= option. The mid-p interval can be modified with the MIDPFACTOR= option. See the section "Exact Conditional Logistic Regression" on page 6071 for details.

ESTIMATE < =*keyword* **>**

estimates the individual parameters (conditioned on all other parameters) for the effects specified in the EXACT statement. For each parameter, a point estimate, a standard error, a confidence interval, and a p-value for a two-sided test that the parameter is zero are displayed. Note that the two-sided p-value is twice the one-sided p-value. You can optionally specify one of the following *keywords*:

PARM specifies that the parameters be estimated. This is the default.

ODDS specifies that the odds ratios be estimated. If you have classification variables, then you must also specify the PARAM=REF option in the CLASS statement.

BOTH specifies that both the parameters and odds ratios be estimated.

JOINT

performs the joint test that all of the parameters are simultaneously equal to zero, performs individual hypothesis tests for the parameter of each continuous variable, and performs joint tests for the parameters of each classification variable. The joint test is indicated in the "Conditional Exact Tests" table by the label "Joint."

JOINTONLY

performs only the joint test of the parameters. The test is indicated in the "Conditional Exact Tests" table by the label "Joint." When this option is specified, individual tests for the parameters of each continuous variable and joint tests for the parameters of the classification variables are not performed.

MIDPFACTOR=δ_1 | (δ_1, δ_2)

sets the tie factors used to produce the mid-p hypothesis statistics and the mid-p confidence intervals. δ_1 modifies both the hypothesis tests and confidence intervals, while δ_2 affects only the hypothesis tests. By default, $\delta_1 = 0.5$ and $\delta_2 = 1.0$. See the section "Exact Conditional Logistic Regression" on page 6071 for details.

ONESIDED

requests one-sided confidence intervals and p-values for the individual parameter estimates and odds ratios. The one-sided p-value is the smaller of the left- and right-tail probabilities for the observed sufficient statistic of the parameter under the null hypothesis that the parameter is zero. The two-sided p-values (default) are twice the one-sided p-values. See the section "Exact Conditional Logistic Regression" on page 6071 for more details.

OUTDIST=*SAS-data-set*

names the SAS data set that contains the exact conditional distributions. This data set contains all of the exact conditional distributions that are required to process the corresponding EXACT statement. This data set contains the possible sufficient statistics for the parameters of the effects specified in the EXACT statement, the counts, and, when hypothesis tests are performed on the parameters, the probability of occurrence and the score value for each sufficient statistic. When you request an OUTDIST= data set, the observed sufficient statistics are displayed in the "Sufficient Statistics" table. See the section "OUTDIST= Output Data Set" on page 6079 for more information.

EXACT Statement Examples

In the following example, two exact tests are computed: one for x1 and the other for x2. The test for x1 is based on the exact conditional distribution of the sufficient statistic for the x1 parameter given the observed values of the sufficient statistics for the intercept, x2, and x3 parameters; likewise, the test for x2 is conditional on the observed sufficient statistics for the intercept, x1, and x3.

```
proc logistic;
   model y= x1 x2 x3;
   exact x1 x2;
run;
```

PROC LOGISTIC determines, from all the specified EXACT statements, the distinct conditional distributions that need to be evaluated. For example, there is only one exact conditional distribution for the following two EXACT statements:

```
exact 'One' x1 / estimate=parm;
exact 'Two' x1 / estimate=parm onesided;
```

For each EXACT statement, individual tests for the parameters of the specified effects are computed unless the JOINTONLY option is specified. Consider the following EXACT statements:

```
exact 'E12' x1 x2 / estimate;
exact 'E1'  x1    / estimate;
exact 'E2'  x2    / estimate;
exact 'J12' x1 x2 / joint;
```

In the E12 statement, the parameters for x1 and x2 are estimated and tested separately. Specifying the E12 statement is equivalent to specifying both the E1 and E2 statements. In the J12 statement, the joint test for the parameters of x1 and x2 is computed in addition to the individual tests for x1 and x2.

EXACTOPTIONS Statement

EXACTOPTIONS *options* **;**

The EXACTOPTIONS statement specifies *options* that apply to every EXACT statement in the program. The following *options* are available:

ABSFCONV=*value*

specifies the absolute function convergence criterion. Convergence requires a small change in the log-likelihood function in subsequent iterations,

$$|l_i - l_{i-1}| < value$$

where l_i is the value of the log-likelihood function at iteration i.

By default, ABSFCONV=1E–12. You can also specify the FCONV= and XCONV= criteria; optimizations are terminated as soon as one criterion is satisfied.

ADDTOBS

adds the observed sufficient statistic to the sampled exact distribution if the statistic was not sampled. This option has no effect unless the METHOD=NETWORKMC or METHOD=MCMC option is specified and the ESTIMATE option is specified in the EXACT statement. If the observed statistic has not been sampled, then the parameter estimate does not exist; by specifying this option, you can produce (biased) estimates.

BUILDSUBSETS

builds every distribution for sampling. By default, some exact distributions are created by taking a subset of a previously generated exact distribution. When the METHOD=NETWORKMC or METHOD=MCMC option is invoked, this subsetting behavior has the effect of using fewer than the desired n samples; see the N= option for more details. Use the BUILDSUBSETS option to suppress this subsetting.

EPSILON=*value*

controls how the partial sums $\sum_{i=1}^{j} y_i \mathbf{x}_i$ are compared. *value* must be between 0 and 1; by default, *value*=1E–8.

FCONV=*value*

specifies the relative function convergence criterion. Convergence requires a small relative change in the log-likelihood function in subsequent iterations,

$$\frac{|l_i - l_{i-1}|}{|l_{i-1}| + 1\text{E–}6} < value$$

where l_i is the value of the log likelihood at iteration i.

By default, FCONV=1E–8. You can also specify the ABSFCONV= and XCONV= criteria; if you specify more than one criterion, then optimizations are terminated as soon as one criterion is satisfied.

MAXSWAP=*n*

MAXR=*n*

specifies the maximum number of swaps for each sample that PROC LOGISTIC makes when you specify the METHOD=MCMC option. If an intercept or a stratum is conditioned out, then *n* swaps are performed: one event is changed to a nonevent, and one nonevent is changed to an event. Although you might need large values of *n* in order to transition between any two points in the exact distribution, such values quickly increase computation time. By default, MAXSWAP=2.

MAXTIME=*seconds*

specifies the maximum clock time (in seconds) that PROC LOGISTIC can use to calculate the exact distributions. If the limit is exceeded, the procedure halts all computations and prints a note to the SAS log. The default maximum clock time is seven days.

METHOD=*keyword*

specifies which exact conditional algorithm to use for every EXACT statement specified. You can specify one of the following *keywords*:

DIRECT invokes the multivariate shift algorithm of Hirji, Mehta, and Patel (1987). This method directly builds the exact distribution, but it can require an excessive amount of memory in its intermediate stages. METHOD=DIRECT is invoked by default when you are conditioning out at most the intercept, or when the LINK=GLOGIT option is specified in the MODEL statement.

NETWORK invokes an algorithm described in Mehta, Patel, and Senchaudhuri (1992). This method builds a network for each parameter that you are conditioning out, combines the networks, then uses the multivariate shift algorithm to create the exact distribution. The NETWORK method can be faster and require less memory than the DIRECT method. The NETWORK method is invoked by default for most analyses.

NETWORKMC invokes the hybrid network and Monte Carlo algorithm of Mehta, Patel, and Senchaudhuri (1992). This method creates a network, then samples from that network; this method does not reject any of the samples at the cost of using a large amount of memory to create the network. METHOD=NETWORKMC is most useful for producing parameter estimates for problems that are too large for the DIRECT and NETWORK methods to handle and for which asymptotic methods are invalid—for example, for sparse data on a large grid.

MCMC invokes the Markov chain Monte Carlo (MCMC) algorithm of Forster, McDonald, and Smith (2003). This method uses a Metropolis-Hastings algorithm to generate samples from the exact distribution by repeatedly perturbing the response vector to obtain a new response

vector while maintaining the sufficient statistics for the nuisance parameters. You must also condition out the intercept or the strata. (Using notation from Zamar et al. 2007, the current implementation samples from $\mathbf{V}_A$, does not explicitly sample for $\boldsymbol{v} = 0$, allows candidates $\boldsymbol{v}$ and $-\boldsymbol{v}$, allows $d < 0$, and limits r by the number of observations.) The sampling is divided into parallel threads, where the number of threads is the value of the SAS system option CPUCOUNT=. The MCMC method is most useful for problems for which the NETWORKMC method has difficulty generating the network and for which asymptotic results are suspect; however, to make sure you are sampling from the equilibrium distribution, you should run your program multiple times and increase the N= and MAXSWAP= values until you believe that your results are stable. The MCMC method can take a large amount of time, depending on the number of observations in your data set, the number of samples, and the number of swaps.

N=*n*

specifies the number of Monte Carlo samples to take when you specify the METHOD=NETWORKMC or METHOD=MCMC option. By default, $n = 10{,}000$. If PROC LOGISTIC cannot obtain n samples because of a lack of memory, then a note is printed in the SAS log (the number of valid samples is also reported in the listing) and the analysis continues.

The number of samples used to produce any particular statistic might be smaller than n. For example, let $X1$ and $X2$ be continuous variables, denote their joint distribution by $f(X1,X2)$, and let $f(X1 \mid X2 = x2)$ denote the marginal distribution of $X1$ conditioned on the observed value of $X2$. If you request the JOINT test of $X1$ and $X2$, then n samples are used to generate the estimate $\hat{f}(X1,X2)$ of $f(X1,X2)$, from which the test is computed. However, the parameter estimate for $X1$ is computed from the subset of $\hat{f}(X1,X2)$ that has $X2 = x2$, and this subset need not contain n samples. Similarly, the distribution for each level of a classification variable is created by extracting the appropriate subset from the joint distribution for the CLASS variable.

In some cases, the marginal sample size can be too small to admit accurate estimation of a particular statistic; a note is printed in the SAS log when a marginal sample size is less than 100. Increasing n increases the number of samples used in a marginal distribution; however, if you want to control the sample size exactly, you can either specify the BUILDSUBSETS option or do both of the following:

- Remove the JOINT option from the EXACT statement.
- Create dummy variables in a DATA step to represent the levels of a CLASS variable, and specify them as independent variables in the MODEL statement.

NBI=*n*

BURNIN=*n*

specifies the number of burn-in samples that are discarded when you specify the METHOD=MCMC option. By default, NBI=0.

NOLOGSCALE

specifies that computations for the exact conditional models be computed by using normal scaling. Log scaling can handle numerically larger problems than normal scaling; however, computations in the log scale are slower than computations in normal scale.

NTHIN=*n*

controls the thinning rate of the sampling when you specify the METHOD=MCMC option. Every *n*th sample is kept and the rest are discarded. By default, NTHIN=1.

ONDISK

uses disk space instead of random access memory to build the exact conditional distribution. Use this option to handle larger problems at the cost of slower processing.

SEED=*seed*

specifies the initial seed for the random number generator used to take the Monte Carlo samples when you specify the METHOD=NETWORKMC or METHOD=MCMC option. The value of the SEED= option must be an integer. If you do not specify a seed, or if you specify a value less than or equal to 0, then PROC LOGISTIC uses the time of day from the computer's clock to generate an initial seed.

STATUSN=*number*

prints a status line in the SAS log after every *number* of Monte Carlo samples when you specify the METHOD=NETWORKMC or METHOD=MCMC option. When you specify METHOD=MCMC, the actual *number* that is used depends on the number of threads used in the computations. The number of samples that are taken and the current exact p-value for testing the significance of the model are displayed. You can use this status line to track the progress of the computation of the exact conditional distributions.

STATUSTIME=*seconds*

specifies the time interval (in seconds) for printing a status line in the SAS log. You can use this status line to track the progress of the computation of the exact conditional distributions. The time interval that you specify is approximate; the actual time interval varies. By default, no status reports are produced.

XCONV=*value*

specifies the relative parameter convergence criterion. Convergence requires a small relative parameter change in subsequent iterations,

$$\max_j |\delta_j^{(i)}| < value$$

where

$$\delta_j^{(i)} = \begin{cases} \beta_j^{(i)} - \beta_j^{(i-1)} & |\beta_j^{(i-1)}| < 0.01 \\ \frac{\beta_j^{(i)} - \beta_j^{(i-1)}}{\beta_j^{(i-1)}} & \text{otherwise} \end{cases}$$

and $\beta_j^{(i)}$ is the estimate of the jth parameter at iteration i.

By default, XCONV=1E–4. You can also specify the ABSFCONV= and FCONV= criteria; if more than one criterion is specified, then optimizations are terminated as soon as one criterion is satisfied.

FREQ Statement

FREQ *variable* **;**

The FREQ statement identifies a *variable* that contains the frequency of occurrence of each observation. PROC LOGISTIC treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If it is not an integer, the frequency value is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the model fitting. When the FREQ statement is not specified, each observation is assigned a frequency of 1. If you specify more than one FREQ statement, then the first statement is used.

If a SCORE statement is specified, then the FREQ variable is used for computing fit statistics and the ROC curve, but they are not required for scoring. If the DATA= data set in the SCORE statement does not contain the FREQ variable, the frequency values are assumed to be 1 and a warning message is issued in the LOG. If you fit a model and perform the scoring in the same run, the same FREQ variable is used for fitting and scoring. If you fit a model in a previous run and input it with the INMODEL= option in the current run, then the FREQ variable can be different from the one used in the previous run. However, if a FREQ variable was not specified in the previous run, you can still specify a FREQ variable in the current run.

ID Statement

ID *variable* < *variable,. . .* > **;**

The ID statement specifies variables in the DATA= data set that are used for labeling ROC and PR curves and influence diagnostic plots. If you specify more than one ID variable, then the plots are labeled by concatenating the ID variable values together. The ID variables are included in the OUTROC= data set that you specify in the MODEL or SCORE statement. For more information, see the PLOTS(LABEL) and ROCOPTIONS(ID) options in the PROC LOGISTIC statement.

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 79.6 summarizes the *options* available in the LSMEANS statement.

Table 79.6 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

NOTE: If you have classification variables in your model, then the LSMEANS statement is allowed only if you also specify the PARAM=GLM option.

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 79.7 summarizes the *options* available in the LSMESTIMATE statement.

Table 79.7 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

NOTE: If you have classification variables in your model, then the LSMESTIMATE statement is allowed only if you also specify the PARAM=GLM option.

MARGINS Statement

MARGINS *model-effects* < / *options* > **;**

The MARGINS statement computes and compares predictive margins of fixed effects. The predictive margin for a specific level (group) of a fixed effect represents the average predicted response if all the observations in the data set were in that group (Lane and Nelder 1982; Chang, Gelman, and Pagano 1982). You can compute predictive margins for any effect in the MODEL statement that involves only classification variables.

Table 79.8 summarizes the *options* available in the MARGINS statement.

Table 79.8 MARGINS Statement Options

Option	Description
Construction and Computation of Predictive Margins	
AT	Modifies the covariate value in computing predictive margins
DIFF	Computes differences of predictive margins
SLICEBY=	Partitions tests of interaction effects
SLICEDIFF	Computes differences of sliced predictive margins and determines the types of differences
USEWEIGHT	Specifies whether to use WEIGHT variables to compute predictive margins
Degrees of Freedom and *p*-Values	
ADJUST=	Specifies the method of multiple comparison adjustment of predictive margin differences
ALPHA=α	Specifies the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple comparison *p*-values further in a step-down fashion
Statistical Output	
CL	Constructs confidence limits for predictive margins and/or predictive margin differences

For more information about the syntax of the MARGINS statement, see the section "MARGINS Statement" on page 525 in Chapter 20, "Shared Concepts and Topics."

NOTE: By default, chi-square and z tests are produced, and weights are not used in the computations. If your model has classification variables, then the MARGINS statement is allowed only if you also specify the PARAM=GLM option.

MODEL Statement

< *label*: > **MODEL** *variable*< (*variable_options*) > = < *effects* > < / *options* > ;

< *label*: > **MODEL** *events/trials* = < *effects* > < / *options* > ;

The MODEL statement names the response variable and the explanatory effects, including covariates, main effects, interactions, and nested effects; see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure," for more information. If you omit the explanatory effects, PROC LOGISTIC fits an intercept-only model. You must specify exactly one MODEL statement. The optional *label* must be a valid SAS name; it is used to identify the resulting output when you specify the ROC statement or the ROCCI option.

Two forms of the MODEL statement can be specified. The first form, referred to as *single-trial* syntax, is applicable to binary, ordinal, and nominal response data. The second form, referred to as *events/trials* syntax, is restricted to the case of binary response data. The single-trial syntax is used when each observation in the DATA= data set contains information about only a single trial, such as a single subject in an experiment. When each observation contains information about multiple binary-response trials, such as the counts of the number of subjects observed and the number responding, then events/trials syntax can be used.

In the events/trials syntax, you specify two variables that contain count data for a binomial experiment. These two variables are separated by a slash. The value of the first variable, *events*, is the number of positive responses (or events). The value of the second variable, *trials*, is the number of trials. The values of both *events* and (*trials–events*) must be nonnegative and the value of *trials* must be positive for the response to be valid.

In the single-trial syntax, you specify one variable (on the left side of the equal sign) as the response variable. This variable can be character or numeric. Variable_options specific to the response variable can be specified immediately after the response variable with parentheses around them.

For both forms of the MODEL statement, explanatory *effects* follow the equal sign. Variables can be either continuous or classification variables. Classification variables can be character or numeric, and they must be declared in the CLASS statement. When an effect is a classification variable, the procedure inserts a set of coded columns into the design matrix instead of directly entering a single column containing the values of the variable.

Response Variable Options

DESCENDING
DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC LOGISTIC orders the response categories according to the ORDER= option and then reverses that order. See the section "Response Level Ordering" on page 6026 for more detail.

EVENT='*category*' | *keyword*

specifies the event category for the binary response model. PROC LOGISTIC models the probability of the event category. The EVENT= option has no effect when there are more than two response categories. You can specify the value (formatted if a format is applied) of the event category in quotation marks, or you can specify one of the following *keywords*:

FIRST designates the first ordered category as the event. This is the default.
LAST designates the last ordered category as the event.

One of the most common sets of response levels is {0,1}, with 1 representing the event for which the probability is to be modeled. Consider the example where Y takes the values 1 and 0 for event and nonevent, respectively, and Exposure is the explanatory variable. To specify the value 1 as the event category, use the following MODEL statement:

```
model Y(event='1') = Exposure;
```

ORDER= DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the response variable. The following table displays the available ORDER= options:

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | *keyword*
REF='*category*' | *keyword*
specifies the reference category for the generalized logit model and the binary response model. For the generalized logit model, each logit contrasts a nonreference category with the reference category. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotation marks, or you can specify one of the following *keywords*:

FIRST designates the first ordered category as the reference.
LAST designates the last ordered category as the reference. This is the default.

Model Options

Table 79.9 summarizes the *options* available in the MODEL statement. These *options* can be specified after a slash (/).

Table 79.9 MODEL Statement Options

Option	Description
Model Specification Options	
EQUALSLOPES	Specifies equal slope parameters
LINK=	Specifies the link function
NOFIT	Suppresses model fitting
NOINT	Suppresses the intercept
OFFSET=	Specifies the offset variable
SELECTION=	Specifies the effect selection method
UNEQUALSLOPES	Specifies unequal slope parameters
Effect Selection Options	
BEST=	Controls the number of models displayed for SCORE selection
DETAILS	Requests detailed results at each step
FAST	Uses the fast elimination method
HIERARCHY=	Specifies whether and how hierarchy is maintained and whether a single effect or multiple effects are allowed to enter or leave the model per step
INCLUDE=	Specifies the number of effects included in every model
MAXSTEP=	Specifies the maximum number of steps for STEPWISE selection
SEQUENTIAL	Adds or deletes effects in sequential order
SLENTRY=	Specifies the significance level for entering effects
SLSTAY=	Specifies the significance level for removing effects
START=	Specifies the number of variables in the first model
STOP=	Specifies the number of variables in the final model
STOPRES	Adds or deletes variables by the residual chi-square criterion
Model-Fitting Specification Options	
ABSFCONV=	Specifies the absolute function convergence criterion
FCONV=	Specifies the relative function convergence criterion
FIRTH	Specifies Firth's penalized likelihood method
GCONV=	Specifies the relative gradient convergence criterion
HALFMAX=	Specifies the maximum number of step halvings
MAXFUNCTION=	Specifies the maximum number of function calls for the conditional analysis
MAXITER=	Specifies the maximum number of iterations
NOCHECK	Suppresses checking for infinite parameters
RIDGEINIT=	Specifies the initial ridging value

Table 79.9 *continued*

Option	Description
RIDGING=	Specifies the technique used to improve the log-likelihood function when its value is worse than that of the previous step
SINGULAR=	Specifies the tolerance for testing singularity
TECHNIQUE=	Specifies the iterative algorithm for maximization
XCONV=	Specifies the relative parameter convergence criterion
Confidence Interval Options	
ALPHA=	Specifies α for the $100(1-\alpha)\%$ confidence intervals
CLODDS=	Computes confidence intervals for odds ratios
CLPARM=	Computes confidence intervals for parameters
PLCONV=	Specifies the profile-likelihood convergence criterion
Classification Options	
CTABLE	Displays the classification table
PEVENT=	Specifies prior event probabilities
PPROB=	Specifies probability cutpoints for classification
Overdispersion and Goodness-of-Fit Test Options	
AGGREGATE=	Determines subpopulations for Pearson chi-square and deviance
GOF	Computes goodness-of-fit tests and statistics
LACKFIT	Requests the Hosmer and Lemeshow goodness-of-fit test
SCALE=	Specifies the method to correct overdispersion
ROC Curve Options	
OUTROC=	Names the output ROC data set
ROCCI	Computes confidence intervals for the AUC
ROCEPS=	Specifies the probability grouping criterion
Regression Diagnostics Options	
INFLUENCE	Displays influence statistics
IPLOTS	Displays influence statistics
Display Options	
CORRB	Displays the correlation matrix
COVB	Displays the covariance matrix
EXPB	Displays exponentiated values of the estimates
ITPRINT	Displays the iteration history
NODUMMYPRINT	Suppresses the “Class Level Information” table
NOODDSRATIO	Suppresses the default “Odds Ratio” table
ORPVALUE	Displays p-values in tables produced by the CLODDS= option and ODDSRATIO statement
PARMLABEL	Displays parameter labels
PCORR	Displays the partial correlation statistic
RSQUARE	Displays the generalized R-square
STB	Displays standardized estimates

Table 79.9 *continued*

Option	Description
Computational Options	
BINWIDTH=	Specifies the bin size for estimating association statistics
NOLOGSCALE	Performs calculations by using normal scaling

The following list describes these *options*.

ABSFCONV=*value*

specifies the absolute function convergence criterion. Convergence requires a small change in the log-likelihood function in subsequent iterations,

$$|l_i - l_{i-1}| < value$$

where l_i is the value of the log-likelihood function at iteration i. See the section "Convergence Criteria" on page 6032 for more information.

AGGREGATE< =(*variable-list***) >**

specifies the subpopulations on which the Pearson chi-square test statistic and the likelihood ratio chi-square test statistic (deviance) are calculated. Observations with common values in the given list of variables are regarded as coming from the same subpopulation. Variables in the list can be any variables in the input data set. Specifying the AGGREGATE option is equivalent to specifying the AGGREGATE= option with a variable list that includes all explanatory variables in the MODEL statement. The deviance and Pearson goodness-of-fit statistics are calculated only when the SCALE= option is specified. Thus, the AGGREGATE (or AGGREGATE=) option has no effect if the SCALE= option is not specified.

See the section "Rescaling the Covariance Matrix" on page 6054 for more information.

ALPHA=*number*

sets the level of significance α for $100(1-\alpha)\%$ confidence intervals for regression parameters or odds ratios. The value of *number* must be between 0 and 1. By default, *number* is equal to the value of the ALPHA= option in the PROC LOGISTIC statement, or 0.05 if the option is not specified. This option has no effect unless confidence limits for the parameters (CLPARM= option) or odds ratios (CLODDS= option or ODDSRATIO statement) are requested.

BEST=*number*

specifies that *number* models with the highest score chi-square statistics are to be displayed for each model size. It is used exclusively with the SCORE model selection method. If the BEST= option is omitted and there are no more than 10 explanatory variables, then all possible models are listed for each model size. If the option is omitted and there are more than 10 explanatory variables, then the number of models selected for each model size is, at most, equal to the number of explanatory variables listed in the MODEL statement.

BINWIDTH=*width*

specifies the size of the bins used for estimating the association statistics. For more information, see the section "Rank Correlation of Observed Responses and Predicted Probabilities" on page 6039. Valid values are $0 \leq \mathit{width} < 1$ (for polytomous response models, $0 < \mathit{width} < 1$). If you have a binary response and fewer than 5,000,000 observations, the default is BINWIDTH=0; this means that no binning is performed and the exact values of the statistics are computed using the trapezoidal area described in the section "ROC Computations" on page 6057. This method is a bit slower and might require more memory than the binning approach.

If a nonzero *width* does not evenly divide the unit interval, it is reduced to a valid value and a message is displayed in the SAS log. The number of bins is also constrained by the amount of memory available on your machine; if a nonzero *width* is so small that you cannot allocate $8/\mathit{width}$ bytes of memory for the bins, then *width* is adjusted to a value for which memory can be allocated and a note is displayed in the SAS log.

The BINWIDTH= option is ignored and no binning is performed when a ROC statement is specified, when ROC graphics are produced, or when the SCORE statement computes an ROC area.

CLODDS=PL | WALD | BOTH

produces confidence intervals for odds ratios of main effects not involved in interactions or nestings. Computation of these confidence intervals is based on the profile likelihood (CLODDS=PL) or based on individual Wald tests (CLODDS=WALD). By specifying CLODDS=BOTH, the procedure computes two sets of confidence intervals for the odds ratios, one based on the profile likelihood and the other based on the Wald tests. The confidence coefficient can be specified with the ALPHA= option. The CLODDS=PL option is not available with the STRATA statement. Classification main effects that use parameterizations other than REF, EFFECT, or GLM are ignored. If you need to compute odds ratios for an effect involved in interactions or nestings, or using some other parameterization, then you should specify an ODDSRATIO statement for that effect.

CLPARM=PL | WALD | BOTH

requests confidence intervals for the parameters. Computation of these confidence intervals is based on the profile likelihood (CLPARM=PL) or individual Wald tests (CLPARM=WALD). If you specify CLPARM=BOTH, the procedure computes two sets of confidence intervals for the parameters, one based on the profile likelihood and the other based on individual Wald tests. The confidence coefficient can be specified with the ALPHA= option. The CLPARM=PL option is not available with the STRATA statement.

See the section "Confidence Intervals for Parameters" on page 6041 for more information.

CORRB

displays the correlation matrix of the parameter estimates.

COVB

displays the covariance matrix of the parameter estimates.

CTABLE

classifies the input binary response observations according to whether the predicted event probabilities are above or below some cutpoint value z in the range $(0, 1)$. An observation is predicted as an event if the predicted event probability exceeds or equals z. You can supply a list of cutpoints other than the default list by specifying the PPROB= option (page 6004). Also, you can compute positive and negative predictive values as posterior probabilities by using Bayes' theorem. You can use the PEVENT= option

to specify prior probabilities for computing these statistics. The CTABLE option is ignored if the data have more than two response levels. This option is not available with the STRATA statement.

For more information, see the section "Classification Table" on page 6047.

DETAILS

produces a summary of computational details for each step of the effect selection process. It produces the "Analysis of Effects Eligible for Entry" table before displaying the effect selected for entry for forward or stepwise selection. For each model fitted, it produces the "Joint Tests" or "Type 3 Analysis of Effects" table if the fitted model involves CLASS variables, the "Analysis of Maximum Likelihood Estimates" table, and measures of association between predicted probabilities and observed responses. For the statistics included in these tables, see the section "Displayed Output" on page 6084. The DETAILS option has no effect when SELECTION=NONE.

EQUALSLOPES< =*effect* | (*effect-list*) >

specifies one or more effects that have the same parameters for each response function in a polytomous response model. If you specify more than one effect, enclose the effects in parentheses. The effects must be explanatory effects that are specified in the MODEL statement. For the generalized logit model, you can also specify the INTERCEPT effect.

If you do not specify this option, the generalized logit model makes the unequal slopes assumption, $g(\Pr(Y = i)) = \alpha_i + \mathbf{x}'\boldsymbol{\beta}_i$, where the response functions have different slope parameters $\boldsymbol{\beta}_i$. If you specify this option without an *effect* or *effect-list*, all slope parameters are shared across the response functions, resulting in the model $g(\Pr(Y = i)) = \alpha_i + \mathbf{x}'\boldsymbol{\beta}$. Specifying an *effect* or *effect-list* enables you to choose which effects have the same parameters across the response functions. For any specified selection method, equal slope parameters can contain and be contained in only other equal slope parameters; for more information, see the HIERARCHY= option.

You can specify the EQUALSLOPES option along with the UNEQUALSLOPES option to create an effect that has both equal and unequal slopes. In this case, the parameters that have equal slopes model the mean effect across the response functions, whereas the parameters that have unequal slopes model deviations from the mean. For more information, see the UNEQUALSLOPES option.

For an example that uses this option, see Example 79.18. If you specify the EQUALSLOPES option, you cannot specify any of the EFFECTPLOT, ESTIMATE, EXACT, LSMEANS, LSMESTIMATE, MARGINS, ROC, ROCCONTRAST, SLICE, STORE, and STRATA statements, and you cannot specify the following *options*: CTABLE, FIRTH, OUTROC=, PEVENT=, PPROB=, RIDGING=, and TECHNIQUE=.

EXPB

EXPEST

displays the exponentiated values ($e^{\widehat{\beta}_i}$) of the parameter estimates $\widehat{\beta}_i$ in the "Analysis of Maximum Likelihood Estimates" table for the logit model. These exponentiated values are the estimated odds ratios for parameters corresponding to the continuous explanatory variables, and for CLASS effects that use reference or GLM parameterizations.

FAST

uses a computational algorithm of Lawless and Singhal (1978) to compute a first-order approximation to the remaining slope estimates for each subsequent elimination of a variable from the model. Variables are removed from the model on the basis of these approximate estimates. This option is extremely efficient because the model is not refitted for every variable removed. The FAST option is used when SELECTION=BACKWARD and in the backward elimination steps when SELECTION=STEPWISE. The FAST option is ignored when SELECTION=FORWARD or SELECTION=NONE.

FCONV=*value*

specifies the relative function convergence criterion. Convergence requires a small relative change in the log-likelihood function in subsequent iterations,

$$\frac{|l_i - l_{i-1}|}{|l_{i-1}| + 1\mathrm{E}{-6}} < value$$

where l_i is the value of the log likelihood at iteration i. See the section "Convergence Criteria" on page 6032 for more information.

FIRTH

performs Firth's penalized maximum likelihood estimation to reduce bias in the parameter estimates (Heinze and Schemper 2002; Firth 1993). This method is useful in cases of separability, as often occurs when the event is rare, and is an alternative to performing an exact logistic regression. See the section "Firth's Bias-Reducing Penalized Likelihood" on page 6032 for more information.

NOTE: The intercept-only log likelihood is modified by using the full-model Hessian, computed with the slope parameters equal to zero. When fitting a model and scoring a data set in the same PROC LOGISTIC step, the model is fit using Firth's penalty for parameter estimation purposes, but the penalty is not applied to the scored log likelihood.

GCONV=*value*

specifies the relative gradient convergence criterion. Convergence requires that the normalized prediction function reduction is small,

$$\frac{\mathbf{g}_i' \mathbf{H}_i^{-1} \mathbf{g}_i}{|l_i| + 1\mathrm{E}{-6}} < value$$

where l_i is the value of the log-likelihood function, $\mathbf{g}_i$ is the gradient vector, and $\mathbf{H}_i$ is the (expected) Hessian matrix, all at iteration i. This is the default convergence criterion, and the default value is 1E–8. See the section "Convergence Criteria" on page 6032 for more information.

GOF< (*gof-options*) **>**

performs goodness-of-fit tests for specified or selected binary response models and displays more fit statistics. This option is not available for polytomous response models or for a link function other than the logit. This option is also ignored if the optimization fails, if your data are separable, or if the specified or selected model has no covariates.

It is usually recommended that you have at least five observations for every combination of your explanatory variables in order to compute valid Pearson chi-square and deviance goodness-of-fit tests. However, when you have sparse data, particularly when you have continuous variables in your model, this condition is not met and these two tests are not appropriate. The GOF option computes additional tests that are valid in these situations, and the "Deviance and Pearson Goodness-of-Fit Statistics" table

is replaced by the "Goodness-of-Fit Statistics" table. These tests are described in detail in the section "Goodness-of-Fit Tests" on page 6050.

This option modifies the standard "Model Fit Statistics" table by displaying one column of the statistics that are discussed in the section "Model Fitting Information" on page 6035.

By default, specifying this option also sets the SCALE=NONE, AGGREGATE, LACKFIT, and PLOTS=CALIBRATION options. You can override these options, except for SCALE=NONE, by explicitly specifying them or by specifying certain *gof-options*.

The LACKFIT option is enabled in a manner that is recommended by Paul, Pennell, and Lemeshow (2013), where the number of groups is chosen to maintain the same power as grouping 1,000 observations into 10 groups. If you have fewer than 1,000 observations, then the LACKFIT(NGROUPS=10) option is used. If you have 1,000 to 25,000 observations, then the following number of groups is formed,

$$\max\left(10, \min\left(\frac{n_e}{2}, \frac{n_n}{2}, 2 + 8\left(\frac{n}{1000}\right)^2\right)\right)$$

where $n = n_e + n_n$ is the number of observations, n_e is the number of events, and n_n is the number of nonevents. If you have more than 25,000 observations, then the LACKFIT option is disabled; however, you can still specify the LACKFIT option in the MODEL statement.

You can specify the following *gof-options*:

CUTPOINT=*number*
specifies a value between 0 and 1 for which observations that have a larger predicted probability are classified as events and observations that have a smaller predicted probability are classified as nonevents. This value is used in computing the misclassification rate. By default, CUTPOINT=0.5.

NOLACKFIT
suppresses the LACKFIT computations.

NOPLOT
suppresses the calibration plots.

NOSTAT
displays the standard "Model Fit Statistics" table.

The GOF option is not available when you specify the EXACT and STRATA statements. The GOF option is ignored when you specify the SELECTION=SCORE option.

HALFMAX=*number*
specifies the maximum number of step halvings to use when you also specify the RIDGING=NONE option. By default, HALFMAX=50.

HIERARCHY=*keyword*
HIER=*keyword*
specifies whether and how the model hierarchy requirement is applied and whether a single effect or multiple effects are allowed to enter or leave the model in one step. You can specify that only CLASS effects, or both CLASS and interval effects, be subject to the hierarchy requirement. The HIERARCHY=

option is ignored unless you also specify one of the following *options*: SELECTION=FORWARD, SELECTION=BACKWARD, or SELECTION=STEPWISE.

Model hierarchy refers to the requirement that, for any term to be in the model, all effects contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must be in the model. Likewise, neither effect A nor B can leave the model while the interaction A*B is in the model.

The *keywords* you can specify in the HIERARCHY= option are as follows:

NONE indicates that the model hierarchy is not maintained. Any single effect can enter or leave the model at any given step of the selection process.

SINGLE indicates that only one effect can enter or leave the model at one time, subject to the model hierarchy requirement. For example, suppose that you specify the main effects A and B and the interaction A*B in the model. In the first step of the selection process, either A or B can enter the model. In the second step, the other main effect can enter the model. The interaction effect can enter the model only when both main effects have already been entered. Also, before A or B can be removed from the model, the A*B interaction must first be removed. All effects (CLASS and interval) are subject to the hierarchy requirement.

SINGLECLASS is the same as HIERARCHY=SINGLE except that only CLASS effects are subject to the hierarchy requirement.

MULTIPLE indicates that more than one effect can enter or leave the model at one time, subject to the model hierarchy requirement. In a forward selection step, a single main effect can enter the model, or an interaction can enter the model together with all the effects that are contained in the interaction. In a backward elimination step, an interaction itself, or the interaction together with all the effects that the interaction contains, can be removed. All effects (CLASS and continuous) are subject to the hierarchy requirement.

MULTIPLECLASS is the same as HIERARCHY=MULTIPLE except that only CLASS effects are subject to the hierarchy requirement.

The default value is HIERARCHY=SINGLE, which means that model hierarchy is to be maintained for all effects (that is, both CLASS and continuous effects) and that only a single effect can enter or leave the model at each step.

INCLUDE=*number* | **EQUALSLOPES**

specifies effects in the MODEL statement to include in every model during model selection. You can specify the following values:

number requests that the first *number* effects be included in every model.

EQUALSLOPES enables you to include all the equal slope effects in every model and perform the selection process on the unequal slope effects.

By default, INCLUDE=0. The INCLUDE= option has no effect when SELECTION=NONE. You cannot specify the INCLUDE=EQUALSLOPES option if you specify SELECTION=SCORE.

If you specify the same effect in both the EQUALSLOPES and UNEQUALSLOPES options, then that effect is treated as two separate effects. For example, suppose you specify the following MODEL statement:

```
model Y=X1 X2 / equalslopes unequalslopes selection=forward;
```

The X1 and X2 variables both generate an equal slope effect and an unequal slope effect. Specifying INCLUDE=1 includes the equal slope effect for X1 in every model; specifying INCLUDE=2 includes both the equal and unequal slope effects for X1 in every model; specifying INCLUDE=EQUALSLOPES includes the equal slope effects for X1 and for X2 in every model.

Note that the INCLUDE= and START= options perform different tasks: the INCLUDE= option includes effects in every model, whereas the START= option requires only that the effects appear in the first model.

INFLUENCE< (STDRES) >

displays diagnostic measures for identifying influential observations in the case of a binary or binomial response model. For each observation, the INFLUENCE option displays the case number (which is the sequence number of the observation), the values of the explanatory variables included in the final model, and the regression diagnostic measures developed by Pregibon (1981). The STDRES option includes standardized and likelihood residuals in the display. See Example 79.6 for an illustration.

For a discussion of these diagnostic measures, see the section "Regression Diagnostics" on page 6060. When a STRATA statement is specified, the diagnostics are computed following Storer and Crowley (1985); for more information, see the section "Regression Diagnostic Details" on page 6069.

IPLOTS

produces an index plot for the regression diagnostic statistics developed by Pregibon (1981) for binary or binomial response models. An index plot is a scatter plot with the regression diagnostic statistic represented on the Y axis and the case number on the X axis. See Example 79.6 for an illustration.

ITPRINT

displays the iteration history of the maximum-likelihood model fitting. The ITPRINT option also displays the last evaluation of the gradient vector and the final change in the –2 Log Likelihood.

LACKFIT< (DF=*df* DFREDUCE=*r* NGROUPS=*G*) >

performs the Hosmer and Lemeshow goodness-of-fit test for binary and polytomous response models (Hosmer and Lemeshow 2000; Fagerland, Hosmer, and Bofin 2008; Fagerland and Hosmer 2013, 2016). The subjects are divided into at most *G* groups of approximately the same size. You can specify *G* as any integer greater than or equal to 5; by default, *G*=10. The discrepancies between the observed and expected number of observations in these groups are summarized by the Pearson chi-square statistic, which is then compared to a chi-square distribution. The number of degrees of freedom that PROC LOGISTIC uses for this test depends on the specified model; you can override this number by specifying a nonnegative integer *df* and by modifying the reduction *r*. For more information, see the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 6050.

A small *p*-value suggests that the fitted model is not an adequate model. The LACKFIT option is not available with the STRATA statement.

LINK=*keyword*

L=*keyword*

specifies the link function linking the response probabilities to the linear predictors. You can specify one of the following *keywords*. The default is LINK=LOGIT.

ALOGIT is the adjacent-category logit function for polytomous response models. PROC LOGISTIC fits the adjacent-category logit model, in which each category is contrasted with the following category.

CLOGLOG is the complementary log-log function. PROC LOGISTIC fits the binary complementary log-log model when there are two response categories and fits the cumulative complementary log-log model when there are more than two response categories. The aliases are CCLOGLOG, CCLL, and CUMCLOGLOG.

GLOGIT is the generalized logit function for polytomous response models. PROC LOGISTIC fits the generalized logit model, in which each nonreference category is contrasted with the reference category. You can use the response variable option REF= to specify the reference category.

LOGIT is the log odds function. PROC LOGISTIC fits the binary logit model when there are two response categories and fits the cumulative logit model when there are more than two response categories. The aliases are CLOGIT and CUMLOGIT.

PROBIT is the inverse standard normal distribution function. PROC LOGISTIC fits the binary probit model when there are two response categories and fits the cumulative probit model when there are more than two response categories. The aliases are NORMIT, CPROBIT, and CUMPROBIT.

The LINK= option is not available with the STRATA statement.

For more information, see the section "Link Functions and the Corresponding Distributions" on page 6028.

MAXFUNCTION=*number*

specifies the maximum number of function calls to perform when you are maximizing the likelihood. This option is valid only when you specify an EQUALSLOPES or UNEQUALSLOPES option, or when you specify a STRATA statement. The default values are as follows:

- 125 when the number of parameters $p < 40$
- 500 when $40 \leq p < 400$
- 1,000 when $p \geq 400$

Because the optimization is terminated only after a full iteration is completed, the number of function calls that are actually performed can exceed *number*. If convergence is not attained, the displayed output and all output data sets that PROC LOGISTIC creates contain results that are based on the last maximum likelihood iteration.

MAXITER=*number*

specifies the maximum number of iterations to perform. By default, MAXITER=25. If convergence is not attained in *number* iterations, the displayed output and all output data sets created by the procedure contain results that are based on the last maximum likelihood iteration.

MAXSTEP=*number*

specifies the maximum number of times any explanatory variable is added to or removed from the model when SELECTION=STEPWISE. The default number is twice the number of explanatory variables in the MODEL statement. When the MAXSTEP= limit is reached, the stepwise selection process is terminated. All statistics that are displayed by the procedure (and included in output data sets) are based on the last model fitted. The MAXSTEP= option has no effect when SELECTION=NONE, FORWARD, or BACKWARD.

NOCHECK

disables the checking process to determine whether maximum likelihood estimates of the regression parameters exist. If you are sure that the estimates are finite, this option can reduce the execution time if the estimation takes more than eight iterations. For more information, see the section "Existence of Maximum Likelihood Estimates" on page 6032.

NODUMMYPRINT

NODESIGNPRINT

NODP

suppresses the "Class Level Information" table, which shows how the design matrix columns for the CLASS variables are coded.

NOINT

suppresses the intercept for the binary response model, the first intercept for the ordinal response model (which forces all intercepts to be nonnegative), or all intercepts for the generalized logit model. This can be particularly useful in conditional logistic analysis; see Example 79.11.

NOFIT

performs the global score test without fitting the model. The global score test evaluates the joint significance of the effects in the MODEL statement. No further analyses are performed. If the NOFIT option is specified along with other MODEL statement options, NOFIT takes effect and all other options except FIRTH, LINK=, NOINT, OFFSET=, and TECHNIQUE= are ignored. The NOFIT option is not available with the STRATA statement.

NOLOGSCALE

specifies that computations for the conditional and exact logistic regression models should be computed by using normal scaling. Log scaling can handle numerically larger problems than normal scaling; however, computations in the log scale are slower than computations in normal scale.

NOODDSRATIO

NOOR

suppresses the default "Odds Ratio" table.

OFFSET=*name*

names the offset variable. The regression coefficient for this variable will be fixed at 1. For an example that uses this option, see Example 79.14. You can also use the OFFSET= option to restrict parameters to a fixed value. For example, if you want to restrict the parameter for variable X1 to 1 and the parameter for X2 to 2, compute Restrict= $X1 + 2 * X2$ in a DATA step, specify the option **`offset=Restrict`**, and leave X1 and X2 out of the model.

ORPVALUE

displays p-values in tables produced by the CLODDS= option and the ODDSRATIO statement. A p-value for an odds ratio corresponds to the significance level such that the two-sided confidence interval for the odds ratio has "1" as one of its endpoints. Computing profile-likelihood p-values requires optimizing restricted log-likelihood equations; for more information, see the section "Confidence Intervals for Parameters" on page 6041. This option is not available with the FIRTH option.

OUTROC=*SAS-data-set*

OUTR=*SAS-data-set*

creates, for binary response models, an output SAS data set that contains the data necessary to produce the receiver operating characteristic (ROC) curve. The OUTROC= option is not available with the STRATA statement. See the section "OUTROC= Output Data Set" on page 6081 for the list of variables in this data set.

PARMLABEL

displays the labels of the parameters in the "Analysis of Maximum Likelihood Estimates" table.

PCORR

computes the partial correlation statistic $\mathrm{sign}(\beta_i)\sqrt{\frac{\chi_i^2-2}{-2\log L_0}}$ for each parameter i, where χ_i^2 is the Wald chi-square statistic for the parameter and $\log L_0$ is the log-likelihood of the intercept-only model (Hilbe 2009, p. 101). If $\chi_i^2 < 2$ then the partial correlation is set to 0. The partial correlation for the intercept terms is set to missing.

PEVENT=*value* | (*list*)

specifies one prior probability or a list of prior probabilities for the event of interest. This option is available only for binary and binomial response models. The positive and negative predictive values are then computed as posterior probabilities by Bayes' theorem. The prior probability is also used in computing the rate of correct prediction. For each prior probability in the given list, a classification table of all observations is computed. By default, the prior probability is the total sample proportion of events. The PEVENT= option is useful for stratified samples. This option affects results that are produced by the CTABLE option, the computation of the positive and negative predictive values that are displayed in ROC curves when ROCOPTIONS(ID=POSPRED) or ROCOPTIONS(ID=NEGPRED), some of the ROC optimal statistics, and the PR curves. This option is ignored by the GOF option. For more information, see the section "Positive Predictive Values, Negative Predictive Values, and Correct Classification Rates Using Bayes' Theorem" on page 6048. Also see the PPROB= option for information about how the *list* is specified.

PLCL
is the same as specifying CLPARM=PL.

PLCONV=*value*
controls the convergence criterion for confidence intervals based on the profile-likelihood function. The quantity *value* must be a positive number, with a default value of 1E–4. The PLCONV= option has no effect if profile-likelihood confidence intervals (CLPARM=PL) are not requested.

PLRL
is the same as specifying CLODDS=PL.

PPROB=*value* | (*list*)
specifies one critical probability value (or cutpoint) or a list of critical probability values for classifying observations with the CTABLE option. This option is available only for binary and binomial response models. Each *value* must be between 0 and 1. A response that has a cross validated predicted probability greater than or equal to the current PPROB= value is classified as an event response. The PPROB= option is ignored if the CTABLE option is not specified.

A classification table for each of several cutpoints can be requested by specifying a list. For example, the following statement requests a classification of the observations for each of the cutpoints 0.3, 0.5, 0.6, 0.7, and 0.8:

```
pprob= (0.3, 0.5 to 0.8 by 0.1)
```

If the PPROB= option is not specified, the default is to display the classification for a range of probabilities from the smallest estimated probability (rounded down to the nearest 0.02) to the highest estimated probability (rounded up to the nearest 0.02) with 0.02 increments.

RIDGEINIT=*number*
specifies the initial ridge value. The maximum ridge value is 2,000 times the maximum of 1 and the initial ridge value. If the initial ridge value is less than 1E–4, it is raised to 1E–4. By default, RIDGEINIT=1E–4.

RIDGING=ABSOLUTE | RELATIVE | NONE
specifies the technique used to improve the log-likelihood function when its value in the current iteration is less than that in the previous iteration. If you specify the RIDGING=ABSOLUTE option, the diagonal elements of the negative (expected) Hessian are inflated by adding the ridge value. If you specify the RIDGING=RELATIVE option, the diagonal elements are inflated by a factor of 1 plus the ridge value. If you specify the RIDGING=NONE option, the crude line search method of taking half a step is used instead of ridging. By default, RIDGING=RELATIVE.

RISKLIMITS

RL

WALDRL
is the same as specifying CLODDS=WALD.

ROCCI

displays standard errors and confidence limits for the area under the ROC curve (AUC) statistic when you have a binary response variable. This option replaces the "Association of Predicted Probabilities and Observed Responses" table with the "ROC Association Statistics" table.

ROCEPS=*number*

specifies a criterion for the ROC and PR curves that is used for grouping estimated event probabilities that are close to each other. This option is available only for binary and binomial response models. In each group, the difference between the largest and the smallest estimated event probabilities does not exceed the given value. The value of *number* must be between 0 and 1; the default value is the square root of the machine epsilon, which is about 1E–8. The smallest estimated probability in each group serves as a cutpoint for predicting an event response. The ROCEPS= option has no effect unless you specify the OUTROC= option, the BINWIDTH=0 option, an ROC statement, the PLOTS=PR option, or the PLOTS=ROC option.

RSQUARE

RSQ

requests a generalized R-square measure for binary and polytomous response models. For more information, see the section "Generalized Coefficient of Determination" on page 6036.

SCALE=*scale*

enables you to supply the value of the dispersion parameter or to specify the method for estimating the dispersion parameter. It also enables you to display the "Deviance and Pearson Goodness-of-Fit Statistics" table. To correct for overdispersion or underdispersion, the covariance matrix is multiplied by the estimate of the dispersion parameter. Valid values for *scale* are as follows:

D | DEVIANCE specifies that the dispersion parameter be estimated by the deviance divided by its degrees of freedom.

P | PEARSON specifies that the dispersion parameter be estimated by the Pearson chi-square statistic divided by its degrees of freedom.

WILLIAMS <(*constant*)> specifies that Williams' method be used to model overdispersion. This option can be used only with the events/trials syntax. An optional *constant* can be specified as the scale parameter; otherwise, a scale parameter is estimated under the full model. A set of weights is created based on this scale parameter estimate. These weights can then be used in fitting subsequent models of fewer terms than the full model. When fitting these submodels, specify the computed scale parameter as *constant*. See Example 79.10 for an illustration.

N | NONE specifies that no correction is needed for the dispersion parameter; that is, the dispersion parameter remains as 1. This specification is used for requesting the deviance and the Pearson chi-square statistic without adjusting for overdispersion.

constant sets the estimate of the dispersion parameter to be the square of the given *constant*. For example, SCALE=2 sets the dispersion parameter to 4. The value *constant* must be a positive number.

You can use the AGGREGATE (or AGGREGATE=) option to define the subpopulations for calculating the Pearson chi-square statistic and the deviance. In the absence of the AGGREGATE (or AGGREGATE=) option, each observation is regarded as coming from a different subpopulation. For

the events/trials syntax, each observation consists of *n* Bernoulli trials, where *n* is the value of the *trials* variable. For single-trial syntax, each observation consists of a single response, and for this setting it is not appropriate to carry out the Pearson or deviance goodness-of-fit analysis. Thus, PROC LOGISTIC ignores specifications SCALE=P, SCALE=D, and SCALE=N when single-trial syntax is specified without the AGGREGATE (or AGGREGATE=) option.

The "Deviance and Pearson Goodness-of-Fit Statistics" table includes the Pearson chi-square statistic, the deviance, the degrees of freedom, the ratio of each statistic divided by its degrees of freedom, and the corresponding *p*-value. The SCALE= option is not available with the STRATA statement. For more information, see the section "Overdispersion" on page 6053.

SELECTION=BACKWARD | B
| FORWARD | F
| NONE | N
| STEPWISE | S
| SCORE

specifies the method used to select the variables in the model. BACKWARD requests backward elimination, FORWARD requests forward selection, NONE fits the complete model specified in the MODEL statement, and STEPWISE requests stepwise selection. SCORE requests best subset selection. By default, SELECTION=NONE. SCORE selection is not available when you also specify LINK=GLOGIT.

For more information, see the section "Effect-Selection Methods" on page 6034.

SEQUENTIAL
SEQ

forces effects to be added to the model in the order specified in the MODEL statement or eliminated from the model in the reverse order of that specified in the MODEL statement. The model-building process continues until the next effect to be added has an insignificant adjusted chi-square statistic or until the next effect to be deleted has a significant Wald chi-square statistic. The SEQUENTIAL option has no effect when SELECTION=NONE.

SINGULAR=*value*

specifies the tolerance for testing the singularity of the Hessian matrix (Newton-Raphson algorithm) or the expected value of the Hessian matrix (Fisher scoring algorithm). The Hessian matrix is the matrix of second partial derivatives of the log-likelihood function. The test requires that a pivot for sweeping this matrix be at least this number times a norm of the matrix. Values of the SINGULAR= option must be numeric. By default, *value* is the machine epsilon times 1E7, which is approximately 1E–9. However, if you specify an ordinal, orthogonalized ordinal, or orthogonalized reference parameterization in the CLASS statement, then *value* is the machine epsilon times 1E4, which is approximately 1E–12.

SLENTRY=*value*
SLE=*value*

specifies the significance level of the score chi-square for entering an effect into the model in the FORWARD or STEPWISE method. Values of the SLENTRY= option should be between 0 and 1, inclusive. By default, SLENTRY=0.05. The SLENTRY= option has no effect when SELECTION=NONE, SELECTION=BACKWARD, or SELECTION=SCORE.

SLSTAY=*value*
SLS=*value*

specifies the significance level of the Wald chi-square for an effect to stay in the model in a backward elimination step. Values of the SLSTAY= option should be between 0 and 1, inclusive. By default, SLSTAY=0.05. The SLSTAY= option has no effect when SELECTION=NONE, SELECTION=FORWARD, or SELECTION=SCORE.

START=*number* | **EQUALSLOPES**

specifies which effects in the MODEL statement are included in the initial model. You can specify the following values:

number requests that the first *number* effects be included in the initial model. The value of *number* ranges from 0 to s, where s is the total number of effects that are specified in the MODEL statement. The default value of *number* is s when SELECTION=BACKWARD and 0 when SELECTION=FORWARD or SELECTION=STEPWISE. When SELECTION=SCORE, START=*number* specifies that the smallest models contain *number* effects, where *number* ranges from 1 to s; the default value is 1.

EQUALSLOPES enables you to begin the model selection process with all the equal slope effects in the model.

The START= option has no effect when SELECTION=NONE. You cannot specify the START=EQUALSLOPES option if you specify SELECTION=SCORE.

If you specify the same effect in both the EQUALSLOPES and UNEQUALSLOPES options, then that effect is treated as two separate effects. For example, suppose you specify the following MODEL statement:

```
model Y=X1 X2 / equalslopes unequalslopes selection=forward;
```

The X1 and X2 variables both generate an equal slope effect and an unequal slope effect. Specifying START=1 includes the equal slope effect for X1 in the initial model; specifying START=2 includes both the equal and unequal slope effects for X1 in the initial model; specifying START=EQUALSLOPES includes the equal slope effects for X1 and for X2 in the initial model.

Note that the INCLUDE= and START= options perform different tasks: the INCLUDE= option includes effects in every model, whereas the START= option requires only that the effects appear in the first model.

STB

displays the standardized estimates for the parameters in the "Analysis of Maximum Likelihood Estimates" table. The standardized estimate of β_i is given by $\widehat{\beta}_i/(s/s_i)$, where s_i is the total sample standard deviation for the ith explanatory variable and

$$s = \begin{cases} \pi/\sqrt{3} & \text{Logistic} \\ 1 & \text{Normal} \\ \pi/\sqrt{6} & \text{Extreme-value} \end{cases}$$

The sample standard deviations for parameters associated with CLASS and EFFECT variables are computed using their codings. For the intercept parameters, the standardized estimates are set to missing.

STOP=*number*

specifies the maximum (SELECTION=FORWARD) or minimum (SELECTION=BACKWARD) number of effects to be included in the final model. The effect selection process is stopped when *number* effects are found. The value of *number* ranges from 0 to *s*, where *s* is the total number of effects in the MODEL statement. The default value of *number* is *s* for the FORWARD method and 0 for the BACKWARD method. For the SCORE method, STOP=*number* specifies that the largest models contain *number* effects, where *number* ranges from 1 to *s*; the default value of *number* is *s*. The STOP= option has no effect when SELECTION=NONE or STEPWISE.

STOPRES

SR

specifies that the removal or entry of effects be based on the value of the residual chi-square. If SELECTION=FORWARD, then the STOPRES option adds the effects into the model one at a time until the residual chi-square becomes insignificant (until the *p*-value of the residual chi-square exceeds the SLENTRY= *value*). If SELECTION=BACKWARD, then the STOPRES option removes effects from the model one at a time until the residual chi-square becomes significant (until the *p*-value of the residual chi-square becomes less than the SLSTAY= *value*). The STOPRES option has no effect when SELECTION=NONE or SELECTION=STEPWISE.

TECHNIQUE=FISHER | NEWTON

TECH=FISHER | NEWTON

specifies the optimization technique for estimating the regression parameters. NEWTON (or NR) is the Newton-Raphson algorithm and FISHER (or FS) is the Fisher scoring algorithm. Both techniques yield the same estimates, but the estimated covariance matrices are slightly different except for the case when the LOGIT link is specified for binary response data. By default, TECHNIQUE=FISHER. If you specify the LINK=GLOGIT or LINK=ALOGIT option, then Newton-Raphson is the default and only available method. The TECHNIQUE= option is not applied to conditional and exact conditional analyses. This option is not available when the EQUALSLOPES or UNEQUALSLOPES option is specified. For more information, see the section "Iterative Algorithms for Model Fitting" on page 6030.

UNEQUALSLOPES< =*effect* **>**

UNEQUALSLOPES< =(*effect-list***) >**

specifies one or more effects in a model for which you want a different set of parameters for each response function in a polytomous response model. If you specify more than one effect, enclose the effects in parentheses. The effects must be explanatory effects that are specified in the MODEL statement. Each member of the *effect-list* can have one of the following forms:

effect
effect=numberlist
_C_=numberlist

where the *numberlist* enables you to specify *constrained* parameters (Peterson and Harrell 1990). To assign a default *numberlist* for all the explanatory effects in this option, specify the *_C_=numberlist* form.

For example, suppose your ordinal response variable Y has three levels, {0, 1, 2}, so that you have two cumulative response functions. Let the CLASS variable A have three levels, {a, b, c}, with reference coding. The following table shows how the *numberlist* is distributed across the models:

If You Specify	Then You Fit This Model
`unequalslopes=(A=1 2 3 4)`	$g(\Pr(Y=0)) = \alpha_0 + 1(A=\text{'a'})\beta_1 + 3(A=\text{'b'})\beta_2$
	$g(\Pr(Y \le 1)) = \alpha_1 + 2(A=\text{'a'})\beta_1 + 4(A=\text{'b'})\beta_2$
`unequalslopes=(_C_=1 2)`	$g(\Pr(Y=0)) = \alpha_0 + 1(A=\text{'a'})\beta_1 + 1(A=\text{'b'})\beta_2$
	$g(\Pr(Y \le 1)) = \alpha_1 + 2(A=\text{'a'})\beta_1 + 2(A=\text{'b'})\beta_2$

If you do not specify this option, the cumulative response models (and the adjacent-category logit model) make the parallel lines assumption, $g(\Pr(Y \le i)) = \alpha_i + \mathbf{x}'\boldsymbol{\beta}$, where each response function has the same slope parameters $\boldsymbol{\beta}$. If you specify this option without an *effect* or *effect-list*, all slope parameters vary across the response functions, resulting in the model $g(\Pr(Y \le i)) = \alpha_i + \mathbf{x}'\boldsymbol{\beta}_i$. Specifying an *effect* or *effect-list* enables you to choose which effects have different parameters across the response functions, and whether the parameters are constrained or unconstrained. For any specified selection method, unconstrained parameters can contain and be contained in only other unconstrained parameters, and constrained parameters can contain and be contained in only other constrained parameters; for more information, see the HIERARCHY= option. If you select the first $\mathbf{x}_1$ parameters to have equal slopes and the remaining $\mathbf{x}_2$ parameters to have unequal slopes, the model can be written as $g(\Pr(Y \le i)) = \alpha_i + \mathbf{x}_1'\boldsymbol{\beta}_1 + \mathbf{x}_2'\boldsymbol{\beta}_{2i}$. Such a model that uses the CLOGIT link is called a *partial proportional odds model* (Peterson and Harrell 1990).

You can specify this option along with the EQUALSLOPES option to create an effect that has both equal and unequal slopes. In this case, the parameters that have equal slopes model the mean effect across the response functions, whereas the parameters that have unequal slopes model deviations from the mean. To distinguish between these two types of parameters, the unconstrained unequal slope parameters are prefixed with "U_" and the constrained parameters are prefixed with "C_". You can use the "Joint Tests" or "Type 3 Analysis of Effects" table to test whether the unequal slope parameters are zero.

For an example that uses this option, see Example 79.18. If you specify the UNEQUALSLOPES option, you cannot specify any of the EFFECTPLOT, ESTIMATE, EXACT, LSMEANS, LSMESTIMATE, MARGINS, ROC, ROCCONTRAST, SLICE, STORE, and STRATA statements, and you cannot specify the following *options*: CTABLE, FIRTH, OUTROC=, PEVENT=, PPROB=, RIDGING=, and TECHNIQUE=.

WALDCL

CL

is the same as specifying CLPARM=WALD.

XCONV=*value*

specifies the relative parameter convergence criterion. Convergence requires a small relative parameter change in subsequent iterations,

$$\max_j |\delta_j^{(i)}| < value$$

where

$$\delta_j^{(i)} = \begin{cases} \beta_j^{(i)} - \beta_j^{(i-1)} & |\beta_j^{(i-1)}| < 0.01 \\ \frac{\beta_j^{(i)} - \beta_j^{(i-1)}}{\beta_j^{(i-1)}} & \text{otherwise} \end{cases}$$

and $\beta_j^{(i)}$ is the estimate of the jth parameter at iteration i. See the section "Convergence Criteria" on page 6032 for more information.

NLOPTIONS Statement

NLOPTIONS < *options* > **;**

The NLOPTIONS statement controls the optimization process for conditional analyses (which result from specifying a STRATA statement) and for partial parallel slope models (which result from specifying the EQUALSLOPES or UNEQUALSLOPES option in the MODEL statement). An option that is specified in the NLOPTIONS statement takes precedence over the same option specified in the MODEL statement.

The default optimization techniques are chosen according to the number of parameters, p, as follows:

- Newton-Raphson with ridging when $p < 40$
- quasi-Newton when $40 \leq p < 400$
- conjugate gradient when $p \geq 400$

The available *options* are described in the section "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics."

ODDSRATIO Statement

ODDSRATIO < '*label*' > *variable* < / *options* > **;**

The ODDSRATIO statement produces odds ratios for variable even when the variable is involved in interactions with other covariates, and for classification variables that use any parameterization. You can also specify variables on which constructed effects are based, in addition to the names of COLLECTION or MULTIMEMBER effects. You can specify several ODDSRATIO statements.

If variable is continuous, then the odds ratios honor any values specified in the UNITS statement. If variable is a classification variable, then odds ratios comparing each pairwise difference between the levels of variable are produced. If variable interacts with a continuous variable, then the odds ratios are produced at the mean of the interacting covariate by default. If variable interacts with a classification variable, then the odds ratios are produced at each level of the interacting covariate by default. The computed odds ratios are independent of the parameterization of any classification variable.

The odds ratios are uniquely labeled by concatenating the following terms to variable:

1. If this is a polytomous response model, then prefix the response variable and the level describing the logit followed by a colon; for example, "Y 0:".
2. If variable is continuous and the UNITS statement provides a value that is not equal to 1, then append "Units=value"; otherwise, if variable is a classification variable, then append the levels being contrasted; for example, "cat vs dog".
3. Append all interacting covariates preceded by "At"; for example, "At X=1.2 A=cat".

If you are also creating odds ratio plots, then this label is displayed on the plots (see the PLOTS option for more information). If you specify a 'label' in the ODDSRATIO statement, then the odds ratios produced by this statement are also labeled: 'label', 'label 2', 'label 3',. . . , and these are the labels used in the plots. If there

are any duplicated labels across all ODDSRATIO statements, then the corresponding odds ratios are not displayed on the plots.

The following *options* are available:

AT(*covariate=value-list* **| REF | ALL<...** *covariate=value-list* **| REF | ALL>)**
specifies fixed levels for covariates. These levels are used in the odds ratio computation when the covariate interacts with *variable*. The specified levels for *variable* are used when *variable* constructs a SPLINE or POLYNOMIAL effect.

For continuous covariates, you can specify one or more numbers in the *value-list*. For classification covariates, you can specify one or more formatted levels of the covariate enclosed in single quotes (for example, **`A='cat' 'dog'`**), you can specify the keyword REF to select the reference-level, or you can specify the keyword ALL to select all levels of the classification variable. By default, continuous covariates are set to their means, while CLASS covariates are set to ALL. For a model that includes a classification variable A={cat,dog} and a continuous covariate X, specifying **`AT(A='cat' X=7 9)`** will set A to 'cat', and X to 7 and then 9.

CL=WALD | PL | BOTH
specifies whether to create Wald or profile-likelihood confidence limits, or both. By default, Wald confidence limits are produced.

DIFF=REF | ALL
specifies whether the odds ratios for a classification variable are computed against the reference level, or all pairs of variable are compared. By default, DIFF=ALL. The DIFF= option is ignored when *variable* is continuous.

PLCONV=*value*
controls the convergence criterion for confidence intervals based on the profile-likelihood function. The quantity *value* must be a positive number, with a default value of 1E–4. The PLCONV= option has no effect if profile-likelihood confidence intervals (CL=PL) are not requested.

PLMAXITER=*number*
specifies the maximum number of iterations to perform. By default, PLMAXITER=25. If convergence is not attained in *number* iterations, the odds ratio or the confidence limits are set to missing. The PLMAXITER= option has no effect if profile-likelihood confidence intervals (CL=PL) are not requested.

PLSINGULAR=*value*
specifies the tolerance for testing the singularity of the Hessian matrix (Newton-Raphson algorithm) or the expected value of the Hessian matrix (Fisher scoring algorithm). The test requires that a pivot for sweeping this matrix be at least this number times a norm of the matrix. Values of the PLSINGULAR= option must be numeric. By default, *value* is the machine epsilon times 1E7, which is approximately 1E–9. The PLSINGULAR= option has no effect if profile-likelihood confidence intervals (CL=PL) are not requested.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*> < *keyword-options*> / < *option*> **;**

The OUTPUT statement creates a new SAS data set that contains all the variables in the input data set and, optionally, the estimated linear predictors and their standard error estimates, the estimates of the cumulative or individual response probabilities, and the confidence limits for the cumulative probabilities. Regression diagnostic statistics and estimates of cross validated response probabilities are also available for binary response models. If you specify more than one OUTPUT statement, only the last one is used. Formulas for the statistics are given in the sections "Linear Predictor, Predicted Probability, and Confidence Limits" on page 6046 and "Regression Diagnostics" on page 6060, and, for conditional logistic regression, in the section "Conditional Logistic Regression" on page 6068.

If you use the single-trial syntax, the data set also contains a variable named _LEVEL_, which indicates the level of the response that the given row of output is referring to. For instance, the value of the cumulative probability variable is the probability that the response variable is as large as the corresponding value of _LEVEL_. For more information, see the section "OUT= Output Data Set in the OUTPUT Statement" on page 6077.

The estimated linear predictor, its standard error estimate, all predicted probabilities, and the confidence limits for the cumulative probabilities are computed for all observations in which the explanatory variables have no missing values, even if the response is missing. By adding observations with missing response values to the input data set, you can compute these statistics for new observations or for settings of the explanatory variables not present in the data without affecting the model fit. Alternatively, the SCORE statement can be used to compute predicted probabilities and confidence intervals for new observations.

Table 79.10 summarizes the options available in the OUTPUT statement. The statistic and diagnostic *keyword-options* specify the statistics to be included in the output data set and name the new variables that contain the statistics. If a STRATA statement is specified, only the PREDICTED=, RESCHI=, STDRESCHI=, DFBETAS=, and H= options are available; for more information, see the section "Regression Diagnostic Details" on page 6069.

Table 79.10 OUTPUT Statement Options

Option	Description
ALPHA=	Specifies α for the $100(1-\alpha)\%$ confidence intervals
OUT=	Names the output data set
Statistic *keyword-options*	
LOWER=	Names the lower confidence limit
PREDICTED=	Names the predicted probabilities
PREDPROBS=	Requests the individual, cumulative, or cross validated predicted probabilities
STDXBETA=	Names the standard error estimate of the linear predictor
UPPER=	Names the upper confidence limit
XBETA=	Names the linear predictor
Diagnostic *keyword-options* for Binary Response	
C=	Names the confidence interval displacement
CBAR=	Names the confidence interval displacement

Table 79.10 *continued*

Option	Description
DFBETAS=	Names the standardized deletion parameter differences
DIFCHISQ=	Names the deletion chi-square goodness-of-fit change
DIFDEV=	Names the deletion deviance change
H=	Names the leverage
RESCHI=	Names the Pearson chi-square residual
RESDEV=	Names the deviance residual
RESLIK=	Names the likelihood residual
STDRESCHI=	Names the standardized Pearson chi-square residual
STDRESDEV=	Names the standardized deviance residual

Details on the specifications in the OUTPUT statement follow.

OUT=*SAS-data-set*
names the output data set. If you omit the OUT= option, the output data set is created and given a default name by using the DATA*n* convention.

keyword-options
specifies the statistics to include in the output data set and provides names for the new variables that contain the statistics. The *keyword-options* have the following form:

*keyword***=***name* | *name-list* | *option*

The *keyword* designates the desired statistic (see the following list of *keywords*), *name* provides the name of the new variable that contains the statistic, *name-list* is a list of such names, and *option* generates the statistics using default names.

The *keywords* allowed and the statistics they represent are as follows:

C=*name*
specifies the confidence interval displacement diagnostic that measures the influence of individual observations on the regression estimates.

CBAR=*name*
specifies the confidence interval displacement diagnostic that measures the overall change in the global regression estimates due to deleting an individual observation.

DFBETAS=_ALL_ | *name-list*
specifies the standardized differences in the regression estimates for assessing the effects of individual observations on the estimated regression parameters in the fitted model. You can specify a list of up to $s + 1$ variable names, where s is the number of explanatory variables in the MODEL statement, or you can specify just the keyword _ALL_. In the former specification, the first variable contains the standardized differences in the intercept estimate, the second variable contains the standardized differences in the parameter estimate for the first explanatory variable in the MODEL statement, and so on. In the latter specification, the DFBETAS statistics are named DFBETA_*xxx*, where *xxx* is the name of the regression parameter. For example, if the model contains two variables X1 and X2, the specification DFBETAS=_ALL_ produces three DFBETAS statistics: DFBETA_Intercept, DFBETA_X1, and DFBETA_X2. If an explanatory variable is

not included in the final model, the corresponding output variable named in DFBETAS=*var-list* contains missing values.

DIFCHISQ=*name*
specifies the change in the chi-square goodness-of-fit statistic attributable to deleting the individual observation.

DIFDEV=*name*
specifies the change in the deviance attributable to deleting the individual observation.

H=*name*
specifies the diagonal element of the hat matrix for detecting extreme points in the design space.

LOWER=*name*

L=*name*
names the variable containing the lower confidence limits for π, where π is the probability of the event response if events/trials syntax or single-trial syntax with binary response is specified; for a cumulative model, π is the cumulative probability (that is, the probability that the response is less than or equal to the value of _LEVEL_); for the generalized logit model, it is the individual probability (that is, the probability that the response category is represented by the value of _LEVEL_). See the ALPHA= option to set the confidence level.

PREDICTED=*name*

PRED=*name*

PROB=*name*

P=*name*
names the variable containing the predicted probabilities. For the events/trials syntax or single-trial syntax with binary response, it is the predicted event probability. For a cumulative model, it is the predicted cumulative probability (that is, the probability that the response variable is less than or equal to the value of _LEVEL_); and for the generalized logit model, it is the predicted individual probability (that is, the probability of the response category represented by the value of _LEVEL_).

PREDPROBS=(*options***)**
requests individual, cumulative, or cross validated predicted probabilities. Descriptions of the *options* are as follows.

INDIVIDUAL | I requests the predicted probability of each response level. For a response variable Y with three levels, 1, 2, and 3, the individual probabilities are Pr(Y=1), Pr(Y=2), and Pr(Y=3).

CUMULATIVE | C requests the cumulative predicted probability of each response level. For a response variable Y with three levels, 1, 2, and 3, the cumulative probabilities are Pr(Y$\leq$1), Pr(Y$\leq$2), and Pr(Y$\leq$3). The cumulative probability for the last response level always has the constant value of 1. For generalized logit models, the cumulative predicted probabilities are not computed and are set to missing.

CROSSVALIDATE | XVALIDATE | X requests the cross validated individual predicted probability of each response level. These probabilities are derived from the leave-one-out principle—that is, dropping the data of one subject and reestimating the parameter

estimates. PROC LOGISTIC uses a less expensive one-step approximation to compute the parameter estimates. This option is valid only for binary response models; for nominal and ordinal models, the cross validated probabilities are not computed and are set to missing.

For more information, see the section "Details of the PREDPROBS= Option" on page 6016 at the end of this section.

RESCHI=*name*
specifies the Pearson (chi-square) residual for identifying observations that are poorly accounted for by the model.

RESDEV=*name*
specifies the deviance residual for identifying poorly fitted observations.

RESLIK=*name*
specifies the likelihood residual for identifying poorly fitted observations.

STDRESCHI=*name*
specifies the standardized Pearson (chi-square) residual for identifying observations that are poorly accounted for by the model.

STDRESDEV=*name*
specifies the standardized deviance residual for identifying poorly fitted observations.

STDXBETA=*name*
names the variable containing the standard error estimates of XBETA. For more information, see the section "Linear Predictor, Predicted Probability, and Confidence Limits" on page 6046.

UPPER=*name*

U=*name*
names the variable containing the upper confidence limits for π, where π is the probability of the event response if events/trials syntax or single-trial syntax with binary response is specified; for a cumulative model, π is cumulative probability (that is, the probability that the response is less than or equal to the value of _LEVEL_); for the generalized logit model, it is the individual probability (that is, the probability that the response category is represented by the value of _LEVEL_). See the ALPHA= option to set the confidence level.

XBETA=*name*
names the variable containing the estimates of the linear predictor $\alpha_i + \boldsymbol{\beta}'\mathbf{x}$, where i is the corresponding ordered value of _LEVEL_.

The following *option* is available in the OUTPUT statement and is specified after a slash (/):

ALPHA=*number*
sets the level of significance α for $100(1-\alpha)\%$ confidence limits for the appropriate response probabilities. The value of *number* must be between 0 and 1. By default, *number* is equal to the value of the ALPHA= option in the PROC LOGISTIC statement, or 0.05 if that option is not specified. You can specify this option after a slash(/).

Details of the PREDPROBS= Option

You can request any of the three types of predicted probabilities. For example, you can request both the individual predicted probabilities and the cross validated probabilities by specifying PREDPROBS=(I X).

When you specify the PREDPROBS= option, two automatic variables, _FROM_ and _INTO_, are included for the single-trial syntax and only one variable, _INTO_, is included for the events/trials syntax. The variable _FROM_ contains the formatted value of the observed response. The variable _INTO_ contains the formatted value of the response level with the largest individual predicted probability.

If you specify PREDPROBS=INDIVIDUAL, the OUT= data set contains *k* additional variables representing the individual probabilities, one for each response level, where *k* is the maximum number of response levels across all BY groups. The names of these variables have the form IP_*xxx*, where *xxx* represents the particular level. The representation depends on the following situations:

- If you specify events/trials syntax, *xxx* is either 'Event' or 'Nonevent'. Thus, the variable containing the event probabilities is named IP_Event and the variable containing the nonevent probabilities is named IP_Nonevent.
- If you specify the single-trial syntax with more than one BY group, *xxx* is 1 for the first ordered level of the response, 2 for the second ordered level of the response, and so forth, as given in the "Response Profile" table. The variable containing the predicted probabilities Pr(Y=1) is named IP_1, where Y is the response variable. Similarly, IP_2 is the name of the variable containing the predicted probabilities Pr(Y=2), and so on.
- If you specify the single-trial syntax with no BY-group processing, *xxx* is the left-justified formatted value of the response level (the value might be truncated so that IP_*xxx* does not exceed 32 characters). For example, if Y is the response variable with response levels 'None', 'Mild', and 'Severe', the variables representing individual probabilities Pr(Y='None'), P(Y='Mild'), and P(Y='Severe') are named IP_None, IP_Mild, and IP_Severe, respectively.

If you specify PREDPROBS=CUMULATIVE, the OUT= data set contains *k* additional variables representing the cumulative probabilities, one for each response level, where *k* is the maximum number of response levels across all BY groups. The names of these variables have the form CP_*xxx*, where *xxx* represents the particular response level. The naming convention is similar to that given by PREDPROBS=INDIVIDUAL. The PREDPROBS=CUMULATIVE values are the same as those output by the PREDICT= option, but are arranged in variables on each output observation rather than in multiple output observations.

If you specify PREDPROBS=CROSSVALIDATE, the OUT= data set contains *k* additional variables representing the cross validated predicted probabilities of the *k* response levels, where *k* is the maximum number of response levels across all BY groups. The names of these variables have the form XP_*xxx*, where *xxx* represents the particular level. The representation is the same as that given by PREDPROBS=INDIVIDUAL except that for the events/trials syntax there are four variables for the cross validated predicted probabilities instead of two:

XP_EVENT_R1E is the cross validated predicted probability of an event when a single event is removed from the current observation.

XP_NONEVENT_R1E is the cross validated predicted probability of a nonevent when a single event is removed from the current observation.

XP_EVENT_R1N is the cross validated predicted probability of an event when a single nonevent is removed from the current observation.

XP_NONEVENT_R1N is the cross validated predicted probability of a nonevent when a single nonevent is removed from the current observation.

The cross validated predicted probabilities are precisely those used in the CTABLE option. For more information about the computation, see the section "Predicted Probability of an Event for Classification" on page 6049.

ROC Statement

ROC <'*label*'> <*specification*> <*/ options*> **;**

The ROC statements specify models to be used in the ROC comparisons. This statement is available only for binary and binomial response models. You can specify more than one ROC statement. ROC statements are identified by their *label*—if you do not specify a *label*, the *i*th ROC statement is labeled "ROC*i*". Additionally, the specified or selected model is labeled with the MODEL statement label or "Model" if the MODEL label is not present. The *specification* can be either a list of effects that have previously been specified in the MODEL statement, or PRED=*variable*, where the *variable* does not have to be specified in the MODEL statement. The PRED= option enables you to input a criterion produced outside PROC LOGISTIC; for example, you can fit a random-intercept model by using PROC GLIMMIX or use survey weights in PROC SURVEYLOGISTIC, then use the predicted values from those models to produce an ROC curve for the comparisons. If you do not make a *specification*, then an intercept-only model is fit to the data, resulting in a noninformative ROC curve that can be used for comparing the area under another ROC curve to 0.5.

You can specify a ROCCONTRAST statement and a ROCOPTIONS option in the PROC LOGISTIC statement to control how the models are compared, while the PLOTS=ROC option controls the ODS Graphics displays. See Example 79.8 for an example that uses the ROC statement.

If you specify any *options*, then a "ROC Model Information" table summarizing the new ROC model is displayed. The *options* are ignored for the PRED= specification. The following *options* are available:

NOINT
suppresses the intercept for the model. Note that specifying the NOINT option in the MODEL statement does not affect the ROC model.

NOOFFSET
does not include an offset variable if the OFFSET= option is specified in the MODEL statement. A constant offset has no effect on the ROC curve, although the cutpoints might be different, but a nonconstant offset can affect the parameter estimates and hence the ROC curve.

LINK=*keyword*
specifies the link function to be used in the model. The available keywords are LOGIT, NORMIT, and CLOGLOG. The logit link is the default. Note that the LINK= option in the MODEL statement is ignored.

ROCCONTRAST Statement

ROCCONTRAST < '*label*' > < *contrast* > < / *options* > **;**

The ROCCONTRAST statement compares the different ROC models. This statement is available only for binary and binomial response models. You can specify only one ROCCONTRAST statement. The ROCOPTIONS options in the PROC LOGISTIC statement control how the models are compared. You can specify one of the following *contrast* specifications:

REFERENCE< (MODEL | *'roc-label'*) >
produces a contrast matrix of differences between each ROC curve and a reference curve. The MODEL keyword specifies that the reference curve is that produced from the MODEL statement; the *roc-label* specifies the label of the ROC curve that is to be used as the reference curve. If neither the MODEL keyword nor the *roc-label* label is specified, then the reference ROC curve is either the curve produced from the MODEL statement, the selected model if a selection method is specified, or the model from the first ROC statement if the NOFIT option is specified.

ADJACENTPAIRS
produces a contrast matrix of each ROC curve minus the succeeding curve.

matrix
specifies the contrast in the form **`row1,row2,...`**, where each *row* contains the coefficients used to compare the ROC curves. Each *row* must contain the same number of entries as there are ROC curves being compared. The elements of each *row* refer to the ROC statements in the order in which they are specified. However, the first element of each *row* refers either to the fitted model, the selected model if a SELECTION= method is specified, or the first specified ROC statement if the NOFIT option is specified.

If no *contrast* is specified, then the REFERENCE contrast with the default reference curve is used. See the section "Comparing ROC Curves" on page 6056 for more information about comparing ROC curves, and see Example 79.8 for an example.

The following *options* are available:

E
displays the contrast.

ESTIMATE < = ROWS | ALLPAIRS >
produces estimates of each row of the contrast when ESTIMATE or ESTIMATE=ROWS is specified. If the ESTIMATE=ALLPAIRS option is specified, then estimates of every pairwise difference of ROC curves are produced.

The row contrasts are labeled "ModelLabel1 – ModelLabel2", where the model labels are as described in the ROC statement; in particular, for the REFERENCE contrast, ModelLabel2 is the reference model label. If you specify your own contrast matrix, then the *i*th contrast row estimate is labeled "Row*i*".

COV
displays covariance matrices used in the computations.

SCORE Statement

SCORE < *options* > **;**

The SCORE statement creates a data set that contains all the data in the DATA= data set together with posterior probabilities and, optionally, prediction confidence intervals. Fit statistics are displayed on request. If you have binary response data, the SCORE statement can be used to create a data set containing data for the ROC curve. You can specify several SCORE statements. FREQ, WEIGHT, and BY statements can be used with the SCORE statements. Weights do not affect the computation of predicted probabilities, their confidence limits, or the predicted response level. Weights affect some fit statistics as described in "Fit Statistics for Scored Data Sets" on page 6063. The SCORE statement is not available with the STRATA statement.

If a SCORE statement is specified in the same run as fitting the model, FORMAT statements should be specified after the SCORE statement in order for the formats to apply to all the DATA= and PRIOR= data sets in the SCORE statement.

See the section "Scoring Data Sets" on page 6063 for more information, and see Example 79.16 for an illustration of how to use this statement.

Table 79.11 summarizes the *options* available in the SCORE statement.

Table 79.11 SCORE Statement Options

Option	Description
ALPHA=	Specifies the significance level
CLM	Outputs the Wald-test-based confidence limits
CUMULATIVE	Outputs the cumulative predicted probabilities
CUTPOINT=	Specifies a cutpoint for classifying events
DATA=	Names the SAS data that you want to score
FITSTAT	Displays fit statistics
OUT=	Names the SAS data set that contains the predicted information
OUTROC=	Names the SAS data set that contains the ROC curve
PRIOR=	Names the SAS data set that contains the priors of the response categories
PRIOREVENT=	Specifies the prior event probability
ROCEPS=	Specifies the criterion for grouping estimated event probabilities

You can specify the following *options*:

ALPHA=*number*
specifies the significance level α for $100(1-\alpha)\%$ confidence intervals. By default, the value of *number* is equal to the ALPHA= option in the PROC LOGISTIC statement, or 0.05 if that option is not specified. This option has no effect unless the CLM option in the SCORE statement is requested.

CLM

outputs the Wald-test-based confidence limits for the predicted probabilities. This option is not available when the INMODEL= data set is created with the NOCOV option.

CUMULATIVE

outputs the cumulative predicted probabilities $\Pr(Y \leq i), i = 1, \ldots, k+1$, to the OUT= data set. This option is valid only when you have more than two response levels; otherwise, the option is ignored and a note is printed in the SAS log. These probabilities are named CP_*level_i*, where *level_i* is the *i*th response level.

If the CLM option is also specified in the SCORE statement, then the Wald-based confidence limits for the cumulative predicted probabilities are also output. The confidence limits are named CLCL_*level_i* and CUCL_*level_i*. In particular, for the lowest response level, the cumulative values (CP, CLCL, CUCL) should be identical to the individual values (P, LCL, UCL), and for the highest response level CP=CLCL=CUCL=1.

CUTPOINT=*value*

specifies a value between 0 and 1 for which observations that have a larger predicted probability are classified as events and observations that have a smaller predicted probability are classified as nonevents. This value is used in computing the error rate for binary response models when you also specify the FITSTAT option. The CUTPOINT= option is ignored for polytomous response models. By default, CUTPOINT=0.5.

DATA=*SAS-data-set*

names the SAS data set that you want to score. If you omit the DATA= option in the SCORE statement, then scoring is performed on the DATA= input data set in the PROC LOGISTIC statement, if specified; otherwise, the DATA=_LAST_ data set is used.

It is not necessary for the DATA= data set in the SCORE statement to contain the response variable unless you are specifying the FITSTAT or OUTROC= option.

Only those variables involved in the fitted model effects are required in the DATA= data set in the SCORE statement. For example, the following statements use forward selection to select effects:

```
proc logistic data=Neuralgia outmodel=sasuser.Model;
   class Treatment Sex;
   model Pain(event='Yes')= Treatment|Sex Age
         / selection=forward sle=.01;
run;
```

Suppose that Treatment and Age are the effects selected for the final model. You can score a data set that does not contain the variable Sex because the effect Sex is not in the model that the scoring is based on. For example, the following statements score the Neuralgia data set after dropping the Sex variable:

```
proc logistic inmodel=sasuser.Model;
   score data=Neuralgia(drop=Sex);
run;
```

FITSTAT
displays fit statistics for the data set you are scoring. The data set must contain the response variable. For more information, see the section "Fit Statistics for Scored Data Sets" on page 6063.

OUT=*SAS-data-set*
names the SAS data set that contains the predicted information. If you omit the OUT= option, the output data set is created and given a default name by using the DATA*n* convention.

OUTROC=*SAS-data-set*
names the SAS data set that contains the ROC curve for the DATA= data set. The ROC curve is computed only for binary response data. See the section "OUTROC= Output Data Set" on page 6081 for the list of variables in this data set.

PRIOR=*SAS-data-set*
names the SAS data set that contains the priors of the response categories. The priors can be values proportional to the prior probabilities; thus, they do not necessarily sum to one. This data set should include a variable named _PRIOR_ that contains the prior probabilities. For events/trials MODEL statement syntax, this data set should also include an _OUTCOME_ variable that contains the values EVENT and NONEVENT; for single-trial syntax, this data set should include the response variable that contains the unformatted response categories. See Example 79.16 for an example.

PRIOREVENT=*value* | **SCOREDATA**
specifies the prior event probability for a binary response model. The PRIOREVENT=SCOREDATA option uses the proportion of events in the data set that you specify in the DATA= option of the SCORE statement. If you specify both the PRIOR= and PRIOREVENT= options, the PRIOR= option takes precedence.

ROCEPS=*value*
specifies the criterion for grouping estimated event probabilities that are close to each other for the ROC curve. In each group, the difference between the largest and the smallest estimated event probability does not exceed the given value. The *value* must be between 0 and 1; the default value is the ROCOPTIONS(EPS=) value. If you omit that option, then the default is the MODEL ROCEPS= value; and if you omit that option, then the default is the square root of the machine epsilon, which is about 1E–8. The smallest estimated probability in each group serves as a cutpoint for predicting an event response. The ROCEPS= option has no effect if you omit the OUTROC= option in the SCORE statement.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

NOTE: If you have classification variables in your model, then the SLICE statement is allowed only if you also specify the PARAM=GLM option.

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

The STORE statement is not available for models created with the LINK=ALOGIT option, the EQUALSLOPES or UNEQUALSLOPES options, a STRATA statement, or an EXACT statement.

STRATA Statement

STRATA *variable* < (*option*) > . . . < *variable* < (*option*) > > < / *options* > **;**

The STRATA statement names the *variables* that define *strata* or *matched sets* to use in *stratified logistic regression* of binary response data.

Observations that have the same *variable* values are in the same matched set. For a stratified logistic model, you can analyze 1:1, 1:n, m:n, and general m_i:n_i matched sets where the number of cases and controls varies across strata. At least one variable must be specified to invoke the stratified analysis, and the usual unconditional asymptotic analysis is not performed. The stratified logistic model has the form

$$\text{logit}(\pi_{hi}) = \alpha_h + \mathbf{x}'_{hi}\boldsymbol{\beta}$$

where π_{hi} is the event probability for the ith observation in stratum h with covariates $\mathbf{x}_{hi}$ and where the stratum-specific intercepts α_h are the nuisance parameters that are to be conditioned out.

STRATA variables can also be specified in the MODEL statement as classification or continuous covariates; however, the effects are nondegenerate only when crossed with a nonstratification variable. Specifying several STRATA statements is the same as specifying one STRATA statement that contains all the strata variables. The STRATA variables can be either character or numeric, and the formatted values of the STRATA variables determine the levels. Thus, you can also use formats to group values into levels; see the discussion of the FORMAT procedure in the *Base SAS Procedures Guide*.

The "Strata Summary" table is displayed by default. For an exact logistic regression, it displays the number of strata that have a specific number of events and non-events. For example, if you are analyzing a 1:5 matched study, this table enables you to verify that every stratum in the analysis has exactly one event and five non-events. Strata that contain only events or only non-events are reported in this table, but such strata are uninformative and are not used in the analysis.

If an EXACT statement is also specified, then a stratified *exact* logistic regression is performed.

The EFFECTPLOT, SCORE, and WEIGHT statements are not available with a STRATA statement. The following MODEL options are also not supported with a STRATA statement: CLPARM=PL, CLODDS=PL, CTABLE, FIRTH, LACKFIT, LINK=, NOFIT, OUTMODEL=, OUTROC=, ROC, and SCALE=.

The following *option* can be specified for a stratification variable by enclosing the *option* in parentheses after the variable name, or it can be specified globally for all STRATA variables after a slash (/).

MISSING
treats missing values ('.', ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid STRATA variable values.

The following strata *options* are also available after the slash:

CHECKDEPENDENCY=*keyword*
CHECK=*keyword*
specifies which variables are to be tested for dependency before the analysis is performed. The available *keywords* are as follows:

NONE performs no dependence checking. Typically, a message about a singular information matrix is displayed if you have dependent variables. Dependent variables can be identified after the analysis by noting any missing parameter estimates.

COVARIATES checks dependence between covariates and an added intercept. Dependent covariates are removed from the analysis. However, covariates that are linear functions of the strata variable might not be removed, which results in a singular information matrix message being displayed in the SAS log. This is the default.

ALL checks dependence between all the strata and covariates. This *option* can adversely affect performance if you have a large number of strata.

NOSUMMARY
suppresses the display of the "Strata Summary" table.

INFO
displays the "Strata Information" table, which includes the stratum number, levels of the STRATA variables that define the stratum, the number of events, the number of non-events, and the total frequency for each stratum. Since the number of strata can be very large, this table is displayed only by request.

TEST Statement

< *label*: > **TEST** *equation1* < , *equation2*, . . . > < / *option* > ;

The TEST statement tests linear hypotheses about the regression coefficients. The Wald test is used to perform a joint test of the null hypotheses H_0: $\mathbf{L}\boldsymbol{\beta} = \mathbf{c}$ specified in a single TEST statement, where $\boldsymbol{\beta}$ is the vector of intercept and slope parameters. When $\mathbf{c} = \mathbf{0}$ you should specify a CONTRAST statement instead.

Each *equation* specifies a linear hypothesis (a row of the $\mathbf{L}$ matrix and the corresponding element of the $\mathbf{c}$ vector). Multiple *equations* are separated by commas. The *label*, which must be a valid SAS name, is used to identify the resulting output and should always be included. You can submit multiple TEST statements.

The form of an *equation* is as follows:

term< ± term . . . > < = ± *term* < ± *term*. . . > >

where *term* is a parameter of the model, or a constant, or a constant times a parameter. Intercept and CLASS variable parameter names should be specified as described in the section "Parameter Names in the OUTEST= Data Set" on page 6076. For multinomial response models, this form enables you to construct tests of

parameters from specific logits. When no equal sign appears, the expression is set to 0. The following statements illustrate possible uses of the TEST statement:

```
proc logistic;
   model y= a1 a2 a3 a4;
   test1: test intercept + .5 * a2 = 0;
   test2: test intercept + .5 * a2;
   test3: test a1=a2=a3;
   test4: test a1=a2, a2=a3;
run;
```

Note that the first and second TEST statements are equivalent, as are the third and fourth TEST statements.

You can specify the following *option* in the TEST statement after a slash(/):

PRINT

displays intermediate calculations in the testing of the null hypothesis $H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{c}$. These calculations include $\mathbf{L}\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})\mathbf{L}'$ bordered by $(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{c})$ and $[\mathbf{L}\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})\mathbf{L}']^{-1}$ bordered by $[\mathbf{L}\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})\mathbf{L}']^{-1}(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{c})$, where $\widehat{\boldsymbol{\beta}}$ is the maximum likelihood estimator of $\boldsymbol{\beta}$ and $\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})$ is the estimated covariance matrix of $\widehat{\boldsymbol{\beta}}$.

For more information, see the section "Testing Linear Hypotheses about the Regression Coefficients" on page 6059.

UNITS Statement

UNITS < *independent1*=*list1* | (*numlist1*) < ∗**SD** > >
< *independent2*=*list2* | (*numlist2*) < ∗**SD** > > . . . < / *option* > **;**

The UNITS statement enables you to specify units of change for the continuous explanatory variables so that customized odds ratios can be estimated. If you specify more than one UNITS statement, only the last one is used. An estimate of the corresponding odds ratio is produced for each unit of change specified for an explanatory variable. The UNITS statement is ignored for CLASS variables. Odds ratios are computed only for main effects that are not involved in interactions or nestings, unless an ODDSRATIO statement is also specified. If the CLODDS= option is specified in the MODEL statement, the corresponding confidence limits for the odds ratios are also displayed, as are odds ratios and confidence limits for any CLASS main effects that are not involved in interactions or nestings. The CLASS effects must use the GLM, reference, or effect coding.

The UNITS statement also enables you to customize the odds ratios for effects specified in ODDSRATIO statements, in which case interactions and nestings are allowed, and CLASS variables can be specified with any parameterization.

The term *independent* is the name of an explanatory variable; if you specify the same *independent* variable multiple times, only the first occurrence is used. The *list* represents a list of units of change, separated by spaces, that are of interest for that variable. Each unit of change in a list has one of the following forms:

- *number*
- SD or –SD
- *number* * SD

where *number* is any nonzero number, and SD is the sample standard deviation of the corresponding independent variable. For example, $X = -2$ requests an odds ratio that represents the change in the odds when the variable X is decreased by two units. $X = 2*\text{SD}$ requests an estimate of the change in the odds when X is increased by two sample standard deviations.

The *numlist* is a numeric list enclosed in parentheses. For example, the first three of the following statements specify that the X variable has units 1, 2, 3, and 4, and the last two statements specify the same units as multiples of the sample standard deviation:

```
units X=1 2 3 4;
units X=(1 2 3 4);
units X=(1 to 4 by 1);
units X=SD 2*SD 3*SD 4*SD;
units X=(1 to 4)*SD;
```

You can specify the following *option* in the UNITS statement after a slash (/):

DEFAULT=*list*

gives a list of units of change for all explanatory variables that are not specified in the UNITS statement. Each unit of change can be in any of the forms described previously. If you omit this option, PROC LOGISTIC does not produce customized odds ratio estimates for any continuous explanatory variable that is not listed in the UNITS statement. For more information, see the section "Odds Ratio Estimation" on page 6043.

WEIGHT Statement

WEIGHT *variable* < / *option* > **;**

When a WEIGHT statement appears, each observation in the input data set is weighted by the value of the WEIGHT variable. Unlike a FREQ variable, the values of the WEIGHT variable can be nonintegral and are not truncated. Observations with negative, zero, or missing values for the WEIGHT variable are not used in the model fitting. When you omit the WEIGHT statement, each observation is assigned a weight of 1. You cannot specify both the WEIGHT statement and the STRATA statement. If you specify more than one WEIGHT statement, then the first WEIGHT variable is used.

If a SCORE statement is specified, then the WEIGHT variable is used for computing fit statistics (see the section "Fit Statistics for Scored Data Sets" on page 6063) and the ROC curve (see the WEIGHTED option of the ROCOPTIONS option), but it is not required for scoring. Weights do not affect the computation of predicted probabilities, their confidence limits, or the predicted response level. If the DATA= data set in the SCORE statement does not contain the WEIGHT variable, the weights are assumed to be 1 and a warning message is issued in the SAS log. If you fit a model and perform the scoring in the same run, the same WEIGHT variable is used for fitting and scoring. If you fit a model in a previous run and input it with the INMODEL= option in the current run, then the WEIGHT variable can be different from the one used in the previous run; however, if a WEIGHT variable was not specified in the previous run, you can still specify a WEIGHT variable in the current run.

CAUTION: PROC LOGISTIC does not compute the proper variance estimators if you are analyzing survey data and specifying the sampling weights through the WEIGHT statement. The SURVEYLOGISTIC procedure is designed to perform the necessary, and correct, computations.

The following *option* can be added to the WEIGHT statement after a slash (/):

NORMALIZE
NORM

causes the weights specified by the WEIGHT variable to be normalized so that they add up to the actual sample size. Weights w_i are normalized by multiplying them by $\frac{n}{\sum_{i=1}^{n} w_i}$, where n is the sample size. With this option, the estimated covariance matrix of the parameter estimators is invariant to the scale of the WEIGHT variable.

If an INMODEL= option is specified, then this option is ignored. If a SCORE statement is specified, then this option is ignored unless you score the same data set that you fit.

Details: LOGISTIC Procedure

Missing Values

Any observation with missing values for the response, offset, strata, or explanatory variables is excluded from the analysis; however, missing values are valid for variables specified with the MISSING option in the CLASS or STRATA statement. Observations with a nonpositive or missing weight or with a frequency less than 1 are also excluded. The estimated linear predictor and its standard error estimate, the fitted probabilities and confidence limits, and the regression diagnostic statistics are not computed for any observation with missing offset or explanatory variable values. However, if only the response value is missing, the linear predictor, its standard error, the fitted individual and cumulative probabilities, and confidence limits for the cumulative probabilities can be computed and output to a data set by using the OUTPUT statement.

Response Level Ordering

Response level ordering is important because, by default, PROC LOGISTIC models the probability of response levels with *lower Ordered Value*. Ordered Values are assigned to response levels in ascending sorted order (that is, the lowest response level is assigned Ordered Value 1, the next lowest is assigned Ordered Value 2, and so on) and are displayed in the "Response Profiles" table. If your response variable Y takes values in $\{1, \ldots, k+1\}$, then, by default, the functions modeled with the binary or cumulative model are

$$\text{logit}(\Pr(Y \le i|\mathbf{x})), \quad i = 1, \ldots, k$$

and for the generalized logit model the functions modeled are

$$\log\left(\frac{\Pr(Y = i|\mathbf{x})}{\Pr(Y = k+1|\mathbf{x})}\right), \quad i = 1, \ldots, k$$

where the highest Ordered Value $Y = k+1$ is the reference level. You can change which probabilities are modeled by specifying the EVENT=, REF=, DESCENDING, or ORDER= response variable options in the MODEL statement.

For binary response data with event and nonevent categories, if your event category has a higher Ordered Value, then by default the nonevent is modeled. Because the default response function modeled is

$$\text{logit}(\pi) = \log\left(\frac{\pi}{1-\pi}\right)$$

where π is the probability of the response level assigned Ordered Value 1, and because

$$\text{logit}(\pi) = -\text{logit}(1 - \pi)$$

the effect of modeling the nonevent is to change the signs of α and $\boldsymbol{\beta}$ in the model for the event, $\text{logit}(\pi) = \alpha + \boldsymbol{\beta}'\mathbf{x}$.

For example, suppose the binary response variable Y takes the values 1 and 0 for event and nonevent, respectively, and Exposure is the explanatory variable. By default, PROC LOGISTIC assigns Ordered Value 1 to response level Y=0, and Ordered Value 2 to response level Y=1. As a result, PROC LOGISTIC models the probability of the nonevent (Ordered Value=1) category, and your parameter estimates have the opposite sign from those in the model for the event. To model the event without using a DATA step to change the values of the variable Y, you can control the ordering of the response levels or select the event or reference level, as shown in the following list:

- Explicitly state which response level is to be modeled by using the response variable option EVENT= in the MODEL statement:

```
model Y(event='1') = Exposure;
```

- Specify the nonevent category for the response variable in the response variable option REF= in the MODEL statement. This option is most useful for generalized logit models where the EVENT= option cannot be used.

```
model Y(ref='0') = Exposure;
```

- Specify the response variable option DESCENDING in the MODEL statement to assign the lowest Ordered Value to Y=1:

```
model Y(descending)=Exposure;
```

- Assign a format to Y such that the first formatted value (when the formatted values are put in sorted order) corresponds to the event. In the following example, Y=1 is assigned the formatted value ‘event’ and Y=0 is assigned the formatted value ‘nonevent’. Because ORDER=FORMATTED by default, Ordered Value 1 is assigned to response level Y=1, so the procedure models the event.

```
proc format;
   value Disease 1='event' 0='nonevent';
run;
proc logistic;
   format Y Disease.;
   model Y=Exposure;
run;
```

Link Functions and the Corresponding Distributions

Five link functions are available in the LOGISTIC procedure. The logit function is the default. To specify a different link function, use the LINK= option in the MODEL statement. The link functions and the corresponding distributions are as follows:

- The logit function

  $$g(p) = \log(p/(1-p))$$

  is the inverse of the cumulative logistic distribution function, which is

  $$F(x) = 1/(1+\exp(-x)) = \exp(x)/(1+\exp(x))$$

- The probit (or normit) function

  $$g(p) = \Phi^{-1}(p)$$

  is the inverse of the cumulative standard normal distribution function, which is

  $$F(x) = \Phi(x) = (2\pi)^{-1/2}\int_{-\infty}^{x}\exp(-z^2/2)dz$$

  Traditionally, the probit function contains the additive constant 5, but throughout PROC LOGISTIC, the terms probit and normit are used interchangeably.

- The complementary log-log function

  $$g(p) = \log(-\log(1-p))$$

  is the inverse of the cumulative extreme-value function (also called the Gompertz distribution), which is

  $$F(x) = 1-\exp(-\exp(x))$$

- The generalized logit function extends the binary logit link to a vector of levels $(p_1, \ldots, p_{k+1})$ by contrasting each level with a fixed level:

  $$g(p_i) = \log(p_i/p_{k+1}) \quad i = 1, \ldots, k$$

- The adjacent-category logit function extends the binary logit link to a vector of levels $(p_1, \ldots, p_{k+1})$ by contrasting each level with the following level:

  $$g(p_i) = \log(p_i/p_{i+1}) \quad i = 1, \ldots, k$$

The variances of the normal, logistic, and extreme-value distributions are not the same. Their respective means and variances are shown in the following table:

Distribution	Mean	Variance
Normal	0	1
Logistic	0	$\pi^2/3$
Extreme-value	$-\gamma$	$\pi^2/6$

Here γ is the Euler constant. In comparing parameter estimates from different link functions, you need to take into account the different scalings of the corresponding distributions and, for the complementary log-log function, a possible shift in location. For example, if the fitted probabilities are in the neighborhood of 0.1 to 0.9, then the parameter estimates from the logit link function should be about $\pi/\sqrt{3}$ larger than the estimates from the probit link function.

Determining Observations for Likelihood Contributions

If you use events/trials MODEL statement syntax, split each observation into two observations. One has response value 1 with a frequency equal to the frequency of the original observation (which is 1 if the FREQ statement is not used) times the value of the *events* variable. The other observation has response value 2 and a frequency equal to the frequency of the original observation times the value of (*trials–events*). These two observations will have the same explanatory variable values and the same FREQ and WEIGHT values as the original observation.

For either single-trial or events/trials syntax, let *j* index all observations. In other words, for single-trial syntax, *j* indexes the actual observations. And, for events/trials syntax, *j* indexes the observations after splitting (as described in the preceding paragraph). If your data set has 30 observations and you use single-trial syntax, *j* has values from 1 to 30; if you use events/trials syntax, *j* has values from 1 to 60.

Suppose the response variable in a cumulative response model can take on the ordered values $1, \ldots, k, k+1$, where k is an integer ≥ 1. The likelihood for the *j*th observation with ordered response value y_j and explanatory variables vector $\mathbf{x}_j$ is given by

$$L_j = \begin{cases} F(\alpha_1 + \boldsymbol{\beta}'\mathbf{x}_j) & y_j = 1 \\ F(\alpha_i + \boldsymbol{\beta}'\mathbf{x}_j) - F(\alpha_{i-1} + \boldsymbol{\beta}'\mathbf{x}_j) & 1 < y_j = i \leq k \\ 1 - F(\alpha_k + \boldsymbol{\beta}'\mathbf{x}_j) & y_j = k+1 \end{cases}$$

where $F(\cdot)$ is the logistic, normal, or extreme-value distribution function, $\alpha_1, \ldots, \alpha_k$ are ordered intercept parameters, and $\boldsymbol{\beta}$ is the common slope parameter vector.

For the adjacent-category logit model, the likelihood for the *j*th observation with ordered response value y_j and explanatory variables vector $\mathbf{x}_j$ is given by

$$L_j = \Pr(Y = y_j|\mathbf{x}_j) = \begin{cases} \dfrac{e^{\sum_{i=y_j}^{k} \alpha_i + \mathbf{x}_j'\boldsymbol{\beta}}}{1 + \sum_{m=1}^{k} e^{\sum_{i=m}^{k} \alpha_i + \mathbf{x}_j'\boldsymbol{\beta}}} & 1 \leq y_j \leq k \\ \dfrac{1}{1 + \sum_{m=1}^{k} e^{\sum_{i=m}^{k} \alpha_i + \mathbf{x}_j'\boldsymbol{\beta}}} & y_j = k+1 \end{cases}$$

where the $\alpha_1, \ldots, \alpha_k$ are not necessarily ordered.

For the generalized logit model, letting the $k+1$st level be the reference level, the intercepts $\alpha_1, \ldots, \alpha_k$ are unordered and the slope vector $\boldsymbol{\beta}_i$ varies with each logit. The likelihood for the jth observation with response value y_j and explanatory variables vector $\mathbf{x}_j$ is given by

$$L_j = \Pr(Y = y_j|\mathbf{x}_j) = \begin{cases} \dfrac{e^{\alpha_i + \mathbf{x}'_j \boldsymbol{\beta}_i}}{1 + \sum_{m=1}^{k} e^{\alpha_m + \mathbf{x}'_j \boldsymbol{\beta}_m}} & 1 \le y_j = i \le k \\ \dfrac{1}{1 + \sum_{m=1}^{k} e^{\alpha_m + \mathbf{x}'_j \boldsymbol{\beta}_m}} & y_j = k+1 \end{cases}$$

Iterative Algorithms for Model Fitting

This section describes the two iterative maximum likelihood algorithms that are available in PROC LOGISTIC for fitting an unconditional logistic regression. For information about available optimization techniques for conditional logistic regression and models that specify the EQUALSLOPES or UNEQUALSLOPES options, see the section "NLOPTIONS Statement" on page 6010. Exact logistic regression uses a special algorithm, which is described in the section "Exact Conditional Logistic Regression" on page 6071.

The default maximum likelihood algorithm is the Fisher scoring method, which is equivalent to fitting by iteratively reweighted least squares. The alternative algorithm is the Newton-Raphson method. For generalized logit models, adjacent-category logit models, and models that specify the EQUALSLOPES or UNEQUALSLOPES options, only the Newton-Raphson technique is available. Both algorithms produce the same parameter estimates. However, the estimated covariance matrix of the parameter estimators can differ slightly because Fisher scoring is based on the expected information matrix whereas the Newton-Raphson method is based on the observed information matrix. For a binary logit model, the observed and expected information matrices are identical, resulting in identical estimated covariance matrices for both algorithms. You can specify the TECHNIQUE= option to select a fitting algorithm, and you can specify the FIRTH option to perform a bias-reducing penalized maximum likelihood fit.

Iteratively Reweighted Least Squares Algorithm (Fisher Scoring)

Consider the multinomial variable $\mathbf{Z}_j = (Z_{1j}, \ldots, Z_{k+1,j})'$ such that

$$Z_{ij} = \begin{cases} 1 & \text{if } Y_j = i \\ 0 & \text{otherwise} \end{cases}$$

With π_{ij} denoting the probability that the jth observation has response value i, the expected value of $\mathbf{Z}_j$ is $\boldsymbol{\pi}_j = (\pi_{1j}, \ldots, \pi_{k+1,j})'$ where $\pi_{k+1,j} = 1 - \sum_{i=1}^{k} \pi_{ij}$. The covariance matrix of $\mathbf{Z}_j$ is $\mathbf{V}_j$, which is the covariance matrix of a multinomial random variable for one trial with parameter vector $\boldsymbol{\pi}_j$. Let $\boldsymbol{\beta}$ be the vector of regression parameters; in other words, $\boldsymbol{\beta} = (\alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_s)'$. Let $\mathbf{D}_j$ be the matrix of partial derivatives of $\boldsymbol{\pi}_j$ with respect to $\boldsymbol{\beta}$. The estimating equation for the regression parameters is

$$\sum_j \mathbf{D}'_j \mathbf{W}_j (\mathbf{Z}_j - \boldsymbol{\pi}_j) = \mathbf{0}$$

where $\mathbf{W}_j = w_j f_j \mathbf{V}_j^-$, w_j and f_j are the weight and frequency of the jth observation, and $\mathbf{V}_j^-$ is a generalized inverse of $\mathbf{V}_j$. PROC LOGISTIC chooses $\mathbf{V}_j^-$ as the inverse of the diagonal matrix with $\boldsymbol{\pi}_j$ as the diagonal.

With a starting value of $\boldsymbol{\beta}^{(0)}$, the maximum likelihood estimate of $\boldsymbol{\beta}$ is obtained iteratively as

$$\boldsymbol{\beta}^{(m+1)} = \boldsymbol{\beta}^{(m)} + (\sum_j \mathbf{D}_j' \mathbf{W}_j \mathbf{D}_j)^{-1} \sum_j \mathbf{D}_j' \mathbf{W}_j (\mathbf{Z}_j - \boldsymbol{\pi}_j)$$

where $\mathbf{D}_j$, $\mathbf{W}_j$, and $\boldsymbol{\pi}_j$ are evaluated at $\boldsymbol{\beta}^{(m)}$. The expression after the plus sign is the step size. If the likelihood evaluated at $\boldsymbol{\beta}^{(m+1)}$ is less than that evaluated at $\boldsymbol{\beta}^{(m)}$, then $\boldsymbol{\beta}^{(m+1)}$ is recomputed by step-halving or ridging as determined by the value of the RIDGING= option. The iterative scheme continues until convergence is obtained—that is, until $\boldsymbol{\beta}^{(m+1)}$ is sufficiently close to $\boldsymbol{\beta}^{(m)}$. Then the maximum likelihood estimate of $\boldsymbol{\beta}$ is $\widehat{\boldsymbol{\beta}} = \boldsymbol{\beta}^{(m+1)}$.

The covariance matrix of $\widehat{\boldsymbol{\beta}}$ is estimated by

$$\widehat{\mathrm{Cov}}(\widehat{\boldsymbol{\beta}}) = (\sum_j \widehat{\mathbf{D}}_j' \widehat{\mathbf{W}}_j \widehat{\mathbf{D}}_j)^{-1} = \widehat{\mathbf{I}}^{-1}$$

where $\widehat{\mathbf{D}}_j$ and $\widehat{\mathbf{W}}_j$ are, respectively, $\mathbf{D}_j$ and $\mathbf{W}_j$ evaluated at $\widehat{\boldsymbol{\beta}}$. $\widehat{\mathbf{I}}$ is the information matrix, or the negative expected Hessian matrix, evaluated at $\widehat{\boldsymbol{\beta}}$.

By default, starting values are zero for the slope parameters, and for the intercept parameters, starting values are the observed cumulative logits (that is, logits of the observed cumulative proportions of response). Alternatively, the starting values can be specified with the INEST= option.

Newton-Raphson Algorithm

For cumulative models, let the parameter vector be $\boldsymbol{\beta} = (\alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_s)'$, and for the generalized logit model let $\boldsymbol{\beta} = (\alpha_1, \ldots, \alpha_k, \boldsymbol{\beta}_1', \ldots, \boldsymbol{\beta}_k')'$. The gradient vector and the Hessian matrix are given, respectively, by

$$\begin{aligned} \mathbf{g} &= \sum_j w_j f_j \frac{\partial l_j}{\partial \boldsymbol{\beta}} \\ \mathbf{H} &= \sum_j w_j f_j \frac{\partial^2 l_j}{\partial \boldsymbol{\beta}^2} \end{aligned}$$

where $l_j = \log L_j$ is the log likelihood for the jth observation. With a starting value of $\boldsymbol{\beta}^{(0)}$, the maximum likelihood estimate $\widehat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ is obtained iteratively until convergence is obtained:

$$\boldsymbol{\beta}^{(m+1)} = \boldsymbol{\beta}^{(m)} - \mathbf{H}^{-1}\mathbf{g}$$

where $\mathbf{H}$ and $\mathbf{g}$ are evaluated at $\boldsymbol{\beta}^{(m)}$. If the likelihood evaluated at $\boldsymbol{\beta}^{(m+1)}$ is less than that evaluated at $\boldsymbol{\beta}^{(m)}$, then $\boldsymbol{\beta}^{(m+1)}$ is recomputed by step-halving or ridging.

The covariance matrix of $\widehat{\boldsymbol{\beta}}$ is estimated by

$$\widehat{\mathrm{Cov}}(\widehat{\boldsymbol{\beta}}) = \widehat{\mathbf{I}}^{-1}$$

where the observed information matrix $\widehat{\mathbf{I}} = -\widehat{\mathbf{H}}$ is computed by evaluating $\mathbf{H}$ at $\widehat{\boldsymbol{\beta}}$.

Firth's Bias-Reducing Penalized Likelihood

Firth's bias-reducing penalized likelihood method is available for binary response models. The log likelihood $l(\boldsymbol{\beta})$ is penalized by the determinant of the information matrix $\mathbf{I}$,

$$l(\boldsymbol{\beta}) + \frac{1}{2}\log|\mathbf{I}(\boldsymbol{\beta})|$$

and the components of the gradient $\mathbf{g}(\boldsymbol{\beta})$ are computed as

$$\mathbf{g}(\beta_j) + \frac{1}{2}\mathrm{trace}\left(\mathbf{I}(\boldsymbol{\beta})^{-1}\frac{\partial\mathbf{I}(\boldsymbol{\beta})}{\partial\beta_j}\right)$$

The Hessian matrix is not modified by this penalty, and the optimization is performed in the usual manner.

Note that for the probit and complementary log-log links, the penalty for the Fisher scoring method is based on the expected information matrix, but the penalty for the Newton-Raphson algorithm is based on the observed information matrix. Because these are noncanonical links, the resulting parameter estimates will be different.

Convergence Criteria

Four convergence criteria are available: ABSFCONV=, FCONV=, GCONV=, and XCONV=. If you specify more than one convergence criterion, the optimization is terminated as soon as one of the criteria is satisfied. If none of the criteria is specified, the default is GCONV=1E–8.

If you specify a STRATA statement or the EQUALSLOPES or UNEQUALSLOPES options in the MODEL statement, all unspecified (or nondefault) criteria are also compared to zero. For example, specifying only the criterion XCONV=1E–8 but attaining FCONV=0 terminates the optimization even if the XCONV= criterion is not satisfied, because the log likelihood has reached its maximum. More convergence criteria are also available; for more information, see the section "NLOPTIONS Statement" on page 6010.

Existence of Maximum Likelihood Estimates

The likelihood equation for a logistic regression model does not always have a finite solution. Sometimes there is a nonunique maximum on the boundary of the parameter space, at infinity. The existence, finiteness, and uniqueness of maximum likelihood estimates for the logistic regression model depend on the patterns of data points in the observation space (Albert and Anderson 1984; Santner and Duffy 1986). Existence checks are not performed for conditional logistic regression.

Consider a binary response model. Let Y_j be the response of the jth subject, and let $\mathbf{x}_j$ be the vector of explanatory variables (including the constant 1 associated with the intercept). There are three mutually exclusive and exhaustive types of data configurations: complete separation, quasi-complete separation, and overlap.

Complete Separation There is a complete separation of data points if there exists a vector $\mathbf{b}$ that correctly allocates all observations to their response groups; that is,

$$\begin{cases} \mathbf{b}'\mathbf{x}_j > 0 & Y_j = 1 \\ \mathbf{b}'\mathbf{x}_j < 0 & Y_j = 2 \end{cases}$$

This configuration gives nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, the log likelihood diminishes to zero, and the dispersion matrix becomes unbounded.

Quasi-complete Separation The data are not completely separable, but there is a vector **b** such that

$$\begin{cases} \mathbf{b}'\mathbf{x}_j \geq 0 & Y_j = 1 \\ \mathbf{b}'\mathbf{x}_j \leq 0 & Y_j = 2 \end{cases}$$

and equality holds for at least one subject in each response group. This configuration also yields nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, the dispersion matrix becomes unbounded and the log likelihood diminishes to a nonzero constant.

Overlap If neither complete nor quasi-complete separation exists in the sample points, there is an overlap of sample points. In this configuration, the maximum likelihood estimates exist and are unique.

Complete separation and quasi-complete separation are problems typically encountered with small data sets. Although complete separation can occur with any type of data, quasi-complete separation is not likely with truly continuous explanatory variables.

The LOGISTIC procedure uses a simple empirical approach to recognize the data configurations that lead to infinite parameter estimates. The basis of this approach is that any convergence method of maximizing the log likelihood must yield a solution giving complete separation, if such a solution exists. In maximizing the log likelihood, there is no checking for complete or quasi-complete separation if convergence is attained in eight or fewer iterations. Subsequent to the eighth iteration, the probability of the observed response is computed for each observation. If the predicted response equals the observed response for every observation, there is a complete separation of data points and the iteration process is stopped. If the complete separation of data has not been determined and an observation is identified to have an extremely large probability ($\geq$0.95) of predicting the observed response, there are two possible situations. First, there is overlap in the data set, and the observation is an atypical observation of its own group. The iterative process, if allowed to continue, will stop when a maximum is reached. Second, there is quasi-complete separation in the data set, and the asymptotic dispersion matrix is unbounded. If any of the diagonal elements of the dispersion matrix for the standardized observations vectors (all explanatory variables standardized to zero mean and unit variance) exceeds 5000, quasi-complete separation is declared and the iterative process is stopped. If either complete separation or quasi-complete separation is detected, a warning message is displayed in the procedure output.

Checking for quasi-complete separation is less foolproof than checking for complete separation. The NOCHECK option in the MODEL statement turns off the process of checking for infinite parameter estimates. In cases of complete or quasi-complete separation, turning off the checking process typically results in the procedure failing to converge.

To address the separation issue, you can change your model, specify the FIRTH option to use Firth's penalized likelihood method, or for small data sets specify an EXACT statement to perform an exact logistic regression.

Effect-Selection Methods

Five effect-selection methods are available by specifying the SELECTION= option in the MODEL statement. The simplest method (and the default) is SELECTION=NONE, for which PROC LOGISTIC fits the complete model as specified in the MODEL statement. The other four methods are FORWARD for forward selection, BACKWARD for backward elimination, STEPWISE for stepwise selection, and SCORE for best subsets selection. Intercept parameters are forced to stay in the model unless the NOINT option is specified.

When SELECTION=FORWARD, PROC LOGISTIC first estimates parameters for effects forced into the model. These effects are the intercepts and the first *n* explanatory effects in the MODEL statement, where *n* is the number specified by the START= or INCLUDE= option in the MODEL statement (*n* is zero by default). Next, the procedure computes the score chi-square statistic for each effect not in the model and examines the largest of these statistics. If it is significant at the SLENTRY= level, the corresponding effect is added to the model. Once an effect is entered in the model, it is never removed from the model. The process is repeated until none of the remaining effects meet the specified level for entry or until the STOP= value is reached. For more information, see the section "Score Statistics and Tests" on page 6039.

When SELECTION=BACKWARD, parameters for the complete model as specified in the MODEL statement are estimated unless the START= option is specified. In that case, only the parameters for the intercepts and the first *n* explanatory effects in the MODEL statement are estimated, where *n* is the number specified by the START= option. Results of the Wald test for individual parameters are examined. The least significant effect that does not meet the SLSTAY= level for staying in the model is removed. Once an effect is removed from the model, it remains excluded. The process is repeated until no other effect in the model meets the specified level for removal or until the STOP= value is reached. Backward selection is often less successful than forward or stepwise selection because the full model fit in the first step is the model most likely to result in a complete or quasi-complete separation of response values as described in the section "Existence of Maximum Likelihood Estimates" on page 6032. For more information, see the section "Testing Linear Hypotheses about the Regression Coefficients" on page 6059.

The SELECTION=STEPWISE option is similar to the SELECTION=FORWARD option except that effects already in the model do not necessarily remain. Effects are entered into and removed from the model in such a way that each forward selection step can be followed by one or more backward elimination steps. The stepwise selection process terminates if no further effect can be added to the model or if the current model is identical to a previously visited model.

For SELECTION=SCORE, PROC LOGISTIC uses the branch-and-bound algorithm of Furnival and Wilson (1974) adapted to find a specified number of models with the highest score chi-square statistic for all possible model sizes, from 1, 2, 3 effect models, and so on, up to the single model containing all of the explanatory effects. The number of models displayed for each model size is controlled by the BEST= option. You can use the START= option to impose a minimum model size, and you can use the STOP= option to impose a maximum model size. For instance, with BEST=3, START=2, and STOP=5, the SCORE selection method displays the three best models (that is, the three models with the highest score chi-square statistics) containing 2, 3, 4, and 5 effects. The SELECTION=SCORE option is not available for models that contain CLASS variables or when you also specify LINK=GLOGIT. For more information, see the section "Score Statistics and Tests" on page 6039.

The options FAST, SEQUENTIAL, and STOPRES can alter the default criteria for entering or removing effects from the model when they are used with the FORWARD, BACKWARD, or STEPWISE selection method.

Model Fitting Information

The statistics that this section discusses are useful for comparing competing models that are not necessarily nested. They are different measures of how well your model is fitted to your data.

Let N be the number of observations in your data. For the jth observation, let r_j be the number of events and n_j be the number of trials when events/trials syntax is specified or $n_j = 1$ for single-trial syntax. Let w_j and f_j be the weight and frequency values, respectively, and denote $F = \sum_j f_j$ and $W = \sum_j f_j w_j$. The total sample size is $n = \sum_j f_j n_j$.

Let p denote the number of parameters in the model, including the intercept parameters. Let s be the number of explanatory effects—that is, the number of slope parameters. Let k be the total number of response functions; this is the same as the number of intercepts in the model unless you specify the NOINT option in the MODEL statement. For this section, assume that the NOINT option is not specified. For binary and cumulative response models, $p = k + s$. For the generalized logit model, $p = k(s + 1)$.

Information Criteria

For the jth observation, let $\widehat{\pi}_j^{\text{obs}}$ be the estimated probability of the observed response. The criteria that the LOGISTIC procedure displays are calculated as follows:

- –2 log likelihood:

$$-2\log L = -2\sum_j \frac{w_j}{\sigma^2} f_j \log(\widehat{\pi}_j^{\text{obs}})$$

  where σ^2 is the dispersion parameter, which equals 1 unless the SCALE= option is specified. For binary response models that use events/trials MODEL statement syntax, this is

$$-2\log L = -2\sum_j \frac{w_j}{\sigma^2} f_j [\log \binom{n_j}{r_j} + r_j \log(\widehat{\pi}_j) + (n_j - r_j)\log(1 - \widehat{\pi}_j)]$$

  where $\widehat{\pi}_j$ denotes the estimated event probability. This statistic is reported both with and without the constant term.

- Akaike's information criterion:

$$\text{AIC} = -2\log L + 2p$$

- Schwarz (Bayesian information) criterion:

$$\text{SC} = -2\log L + p\log\left(\sum_j f_j n_j\right)$$

- Akaike's corrected information criterion:

$$\text{AICC} = \begin{cases} -2\log L + 2pF/(F - p - 1) & \text{when } F > p + 2 \\ -2\log L + 2p(p + 2) & \text{otherwise} \end{cases}$$

The AIC and SC statistics give two different ways of adjusting the –2 log-likelihood statistic for the number of terms in the model and the number of observations used. You can use these statistics to compare different models for the same data (for example, when you use the SELECTION=STEPWISE option in the MODEL statement). The models that you are comparing do not have to be nested; lower values of the statistics indicate a more desirable model.

The AICC is a small-sample, bias-corrected version of Akaike's information criterion, as promoted in Hurvich and Tsai (1989) and Burnham and Anderson (1998), for example. The AICC is displayed in the "Model Fit Statistics" table for the selected model when you specify the GOF option in the MODEL statement, and in the "Fit Statistics for SCORE Data" table when you specify the FITSTAT option in the SCORE statement.

The difference in the –2 log-likelihood statistics between the intercepts-only model and the specified model has a $p - k$ degree-of-freedom chi-square distribution under the null hypothesis that all the explanatory effects in the model are zero. The likelihood ratio test in the "Testing Global Null Hypothesis: BETA=0" table displays this difference and the associated p-value for this statistic. The score and Wald tests in that table test the same hypothesis and are asymptotically equivalent; for more information, see the sections "Residual Chi-Square" on page 6040 and "Testing Linear Hypotheses about the Regression Coefficients" on page 6059.

Generalized Coefficient of Determination

Like the AIC and SC statistics that are described in the section "Information Criteria" on page 6035, R-square statistics are most useful for comparing competing models that are not necessarily nested—larger values indicate better models. The statistics that are discussed in this section are based on the likelihood of the fitted model. Specifying the NORMALIZE option in the WEIGHT statement makes these coefficients invariant to the scale of the weights.

Let L_0 denote the likelihood of the intercept-only model, and let L denote the likelihood of the specified model.

Maddala (1983) and Cox and Snell (1989, pp. 208–209) propose the following generalization of the coefficient of determination to a more general linear model:

$$R^2 = 1 - \left(\frac{L_0}{L}\right)^{2/W}$$

The quantity R^2 achieves a maximum of less than one for discrete models, where the maximum is given by

$$R^2_{\max} = 1 - (L_0)^{2/W}$$

Cragg and Uhler (1970), Maddala (1983), and Nagelkerke (1991) propose the following adjusted coefficient, which can achieve a maximum value of 1:

$$R^2_{\mathrm{N}} = \frac{R^2}{R^2_{\max}}$$

The RSQUARE option in the MODEL statement displays R^2, labeled as "R-Square," and R^2_{N}, labeled as "Max-rescaled R-Square," in the "RSquare" table. The GOF option in the MODEL statement displays these two statistics in the "Model Fit Statistics" table. The FITSTAT option in the SCORE statement displays them in the "Fit Statistics for SCORE Data" table.

You produce the remaining statistics in this section by specifying the GOF option when you have a binary logistic regression model. These statistics are displayed in the "Model Fit Statistics" table.

McFadden (1974) suggests a measure analogous to the R-square in the linear regression model, labeled as "McFadden's R-Square," also called the likelihood ratio index or the deviance R-square:

$$R^2_{\mathrm{M}} = 1 - \frac{\log L}{\log L_0}$$

Estrella (1998) devised an R-square measure, labeled "Estrella's R-Square," which satisfies a requirement that its derivative equal 1 when its value is 0:

$$R^2_{\mathrm{E}} = 1 - \left(1 - \frac{2(\log L - \log L_0)}{-2\log L_0}\right)^{-2\log(L_0)/W}$$

Estrella also adjusts R^2_{E}, in a measure labeled "Estrella's Adjusted R-Square," by imposing a penalty for the number of parameters in the model in the same fashion as the AIC:

$$R^2_{\mathrm{EA}} = 1 - \left(\frac{\log L - p}{\log L_0}\right)^{-2\log(L_0)/W}$$

Aldrich and Nelson (1984) based another measure on the model chi-square statistic that compares the full model to the intercept-only model:

$$R^2_{\mathrm{AN}} = \frac{2(\log L - \log L_0)}{2(\log L - \log L_0) + W}$$

Veall and Zimmermann (1996) adjust R^2_{AN} to obtain an upper limit of 1:

$$R^2_{\mathrm{VZ}} = \frac{2(\log L - \log L_0)(-2\log L_0 + W)}{-2\log L_0(2(\log L - \log L_0) + W)}$$

Discussions of these and other pseudo-R^2 measures can also be found in Allison (2014), Menard (2000), Smith and McKenna (2013), and Windmeijer (1995).

Probability- and Residual-Based Model Fitting Criteria

Measures that are discussed in this section use residuals and predicted probabilities to evaluate the strength of the fit.

For binary response models, let $\widehat{\pi}_j$ denote the estimated event probability for observation j. For polytomous response models, let $\widehat{\pi}_{ij}$ denote the predicted probability of response level i, $1 \le i \le k+1$. Let e_j be the residual, which for binomial models is $y_j/n_j - \widehat{\pi}_j$, for binary response models is $y_j - \widehat{\pi}_j$, and for polytomous response models is $\sum_j y_{ij} - \widehat{\pi}_{ij}$.

Average Square Error and Brier Score

The average square error is computed as

$$\mathrm{ASE} = \frac{1}{F}\sum f_j e_j^2$$

If you have specified a WEIGHT statement, then the weighted average square error is

$$\mathrm{WASE} = \frac{1}{W}\sum f_j w_j e_j^2$$

If you specify the FITSTAT option in the SCORE statement, then these statistics are displayed in the "Fit Statistics for SCORE Data" table, and they are labeled as "Brier Score" or "Brier Reliability," as discussed in the section "Fit Statistics for Scored Data Sets" on page 6063. If you specify the GOF option and are fitting a binary logistic regression model, then these statistics are displayed in the "Model Fit Statistics" table.

Misclassification Rate

The misclassification rate, or error rate, is the proportion of observations that are incorrectly classified. By default, observations for which $\widehat{\pi}_j >= 0.5$ are classified as events; otherwise they are classified as nonevents.

If you specify the FITSTAT option in the SCORE statement, then the error rate is displayed in the "Fit Statistics for SCORE Data" table. You can change the cutpoint from 0.5 by specifying the PEVENT= option in the MODEL statement; the first value that you specify in that option is used.

If you specify the GOF option in the MODEL statement and are fitting a binary logistic regression model, then the misclassification rate is displayed in the "Model Fit Statistics" table. You can change the cutpoint from 0.5 by specifying the GOF(CUTPOINT=) option.

Lave and Efron's OLS R-Square

Lave (1970) and Efron and Hinkley (1978) propose a statistic that is analogous to the general linear model R^2,

$$R^2_{\mathrm{E}} = 1 - \frac{\sum f_j w_j e_j^2}{W_1(1 - W_1/W)}$$

where W_1 is the total frequency*weight of the events. If you specify the GOF option in the MODEL statement and are fitting a binary logistic regression model, then this statistic is displayed in the "Model Fit Statistics" table.

Tjur's R-Square

For a binary response model, write the mean of the model-predicted probabilities of event (Y=1) observations as

$$\overline{\widehat{\pi}}_1 = \frac{1}{F_1} \sum_{j=1}^{N} f_j \widehat{\pi}_j I(y_j = 1)$$

and of the nonevent (Y=2) observations as

$$\overline{\widehat{\pi}}_2 = \frac{1}{F_2} \sum_{j=1}^{N} f_j \widehat{\pi}_j I(y_j = 2)$$

where F_1 is the total frequency of events, F_2 is the total frequency of nonevents, and $I(\cdot)$ is the indicator function. Tjur (2009) defines the statistic

$$R^2_{\mathrm{T}} = \overline{\widehat{\pi}}_1 - \overline{\widehat{\pi}}_2$$

and relates it to other R-square measures. If you specify the GOF option in the MODEL statement and are fitting a binary logistic regression model, then this statistic is displayed in the "Model Fit Statistics" table and labeled "Tjur's R-Square." Tjur calls it the *coefficient of discrimination* because it is a measure of the model's ability to distinguish between the event and nonevent distributions; it is called the "Mean Difference" and the "Difference of Means" in other SAS procedures. This statistic is the same as the d' or Δm statistic (with unit standard error) that is discussed in the signal detection literature (McNicol 2005).

Rank Correlation of Observed Responses and Predicted Probabilities

The predicted mean score of an observation is the sum of the Ordered Values (shown in the "Response Profile" table) minus one, weighted by the corresponding predicted probabilities for that observation; that is, the predicted means score $= \sum_{i=1}^{k+1}(i-1)\widehat{\pi}_i$, where $k+1$ is the number of response levels and $\widehat{\pi}_i$ is the predicted probability of the ith (ordered) response.

A pair of observations with different observed responses is said to be *concordant* if the observation with the lower ordered response value has a lower predicted mean score than the observation with the higher ordered response value. If the observation with the lower ordered response value has a higher predicted mean score than the observation with the higher ordered response value, then the pair is *discordant*. If the pair is neither concordant nor discordant, it is a *tie*. If you have more than two response levels, enumeration of the total numbers of concordant and discordant pairs is carried out by categorizing the predicted mean score into intervals of length $k/500$ and accumulating the corresponding frequencies of observations. You can change the length of these intervals by specifying the BINWIDTH= option in the MODEL statement.

Let N be the sum of observation frequencies in the data. Suppose there are a total of t pairs with different responses: n_c of them are concordant, n_d of them are discordant, and $t - n_c - n_d$ of them are tied. PROC LOGISTIC computes the following four indices of rank correlation for assessing the predictive ability of a model:

$$
\begin{array}{lcl}
c & = & (n_c + 0.5(t - n_c - n_d))/t \\
\text{Somers' } D\text{(Gini coefficient)} & = & (n_c - n_d)/t \\
\text{Goodman-Kruskal gamma} & = & (n_c - n_d)/(n_c + n_d) \\
\text{Kendall's tau-}a & = & (n_c - n_d)/(0.5N(N-1))
\end{array}
$$

If there are no ties, then Somers' D (Gini's coefficient) $= 2c - 1$. Note that the concordance index, c, also gives an estimate of the area under the receiver operating characteristic (ROC) curve when the response is binary (Hanley and McNeil 1982). See the section "ROC Computations" on page 6057 for more information about this area.

For binary responses, the predicted mean score is equal to the predicted probability for Ordered Value 2.

These statistics are not available when the STRATA statement is specified.

Score Statistics and Tests

To understand the general form of the score statistics, let $\mathbf{g}(\boldsymbol{\beta})$ be the vector of first partial derivatives of the log likelihood with respect to the parameter vector $\boldsymbol{\beta}$, and let $\mathbf{H}(\boldsymbol{\beta})$ be the matrix of second partial derivatives of the log likelihood with respect to $\boldsymbol{\beta}$. That is, $\mathbf{g}(\boldsymbol{\beta})$ is the gradient vector, and $\mathbf{H}(\boldsymbol{\beta})$ is the Hessian matrix. Let $\mathbf{I}(\boldsymbol{\beta})$ be either $-\mathbf{H}(\boldsymbol{\beta})$ or the expected value of $-\mathbf{H}(\boldsymbol{\beta})$. Consider a null hypothesis H_0. Let $\widehat{\boldsymbol{\beta}}_{H_0}$ be the MLE of $\boldsymbol{\beta}$ under H_0. The chi-square score statistic for testing H_0 is defined by

$$\mathbf{g}'(\widehat{\boldsymbol{\beta}}_{H_0})\mathbf{I}^{-1}(\widehat{\boldsymbol{\beta}}_{H_0})\mathbf{g}(\widehat{\boldsymbol{\beta}}_{H_0})$$

and it has an asymptotic χ^2 distribution with r degrees of freedom under H_0, where r is the number of restrictions imposed on $\boldsymbol{\beta}$ by H_0.

Score statistics are used when performing forward, stepwise and score selection; for more information see the section "Effect-Selection Methods" on page 6034.

Residual Chi-Square

When you use SELECTION=FORWARD, BACKWARD, or STEPWISE, the procedure calculates a residual chi-square score statistic and reports the statistic, its degrees of freedom, and the p-value. This section describes how the statistic is calculated.

Suppose there are s explanatory effects of interest. The full cumulative response model has a parameter vector

$$\boldsymbol{\beta} = (\alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_s)'$$

where $\alpha_1, \ldots, \alpha_k$ are intercept parameters, and $\beta_1, \ldots, \beta_s$ are the common slope parameters for the s explanatory effects. The full generalized logit model has a parameter vector

$$\begin{aligned} \boldsymbol{\beta} &= (\alpha_1, \ldots, \alpha_k, \boldsymbol{\beta}_1', \ldots, \boldsymbol{\beta}_k')' \quad \text{with} \\ \boldsymbol{\beta}_i' &= (\beta_{i1}, \ldots, \beta_{is}), \quad i = 1, \ldots, k \end{aligned}$$

where β_{ij} is the slope parameter for the jth effect in the ith logit.

Consider the null hypothesis $H_0\colon \beta_{t+1} = \cdots = \beta_s = 0$, where $t < s$ for the cumulative response model, and $H_0\colon \beta_{i,t+1} = \cdots = \beta_{is} = 0, t < s, i = 1, \ldots, k$, for the generalized logit model. For the reduced model with t explanatory effects, let $\widehat{\alpha}_1, \ldots, \widehat{\alpha}_k$ be the MLEs of the unknown intercept parameters, let $\widehat{\beta}_1, \ldots, \widehat{\beta}_t$ be the MLEs of the unknown slope parameters, and let $\widehat{\boldsymbol{\beta}}_{i(t)}' = (\widehat{\beta}_{i1}, \ldots, \widehat{\beta}_{it}), i = 1, \ldots, k$, be those for the generalized logit model. The residual chi-square is the chi-square score statistic testing the null hypothesis H_0; that is, the residual chi-square is

$$\mathrm{g}'(\widehat{\boldsymbol{\beta}}_{H_0})\mathbf{I}^{-1}(\widehat{\boldsymbol{\beta}}_{H_0})\mathrm{g}(\widehat{\boldsymbol{\beta}}_{H_0})$$

where for the cumulative response model $\widehat{\boldsymbol{\beta}}_{H_0} = (\widehat{\alpha}_1, \ldots, \widehat{\alpha}_k, \widehat{\beta}_1, \ldots, \widehat{\beta}_t, 0, \ldots, 0)'$, and for the generalized logit model $\widehat{\boldsymbol{\beta}}_{H_0} = (\widehat{\alpha}_1, \ldots, \widehat{\alpha}_k, \widehat{\boldsymbol{\beta}}_{1(t)}', \mathbf{0}_{(s-t)}', \ldots \widehat{\boldsymbol{\beta}}_{k(t)}', \mathbf{0}_{(s-t)}')'$, where $\mathbf{0}_{(s-t)}$ denotes a vector of $s - t$ zeros.

The residual chi-square has an asymptotic chi-square distribution with $s - t$ degrees of freedom ($k(s - t)$ for the generalized logit model). A special case is the global score chi-square, where the reduced model consists of the k intercepts and no explanatory effects. The global score statistic is displayed in the "Testing Global Null Hypothesis: BETA=0" table. The table is not produced when the NOFIT option is used, but the global score statistic is displayed.

Testing Individual Effects Not in the Model

These tests are performed when you specify SELECTION=FORWARD or STEPWISE, and are displayed when the DETAILS option is specified. In the displayed output, the tests are labeled "Score Chi-Square" in the "Analysis of Effects Eligible for Entry" table and in the "Summary of Stepwise (Forward) Selection" table. This section describes how the tests are calculated.

Suppose that k intercepts and t explanatory variables (say $v_1, \ldots, v_t$) have been fit to a model and that v_{t+1} is another explanatory variable of interest. Consider a full model with the k intercepts and $t + 1$ explanatory variables $(v_1, \ldots, v_t, v_{t+1})$ and a reduced model with v_{t+1} excluded. The significance of v_{t+1} adjusted for $v_1, \ldots, v_t$ can be determined by comparing the corresponding residual chi-square with a chi-square distribution with one degree of freedom (k degrees of freedom for the generalized logit model).

Testing the Parallel Lines Assumption

For an ordinal response, PROC LOGISTIC performs a test of the parallel lines assumption. In the displayed output, this test is labeled "Score Test for the Equal Slopes Assumption" when the LINK= option is NORMIT or CLOGLOG. When LINK=LOGIT, the test is labeled as "Score Test for the Proportional Odds Assumption" in the output. For small sample sizes, this test might be too liberal (Stokes, Davis, and Koch 2000, p. 249). This section describes the methods used to calculate the test.

For this test the number of response levels, $k+1$, is assumed to be strictly greater than 2. Let Y be the response variable taking values $1, \ldots, k, k+1$. Suppose there are s explanatory variables. Consider the general cumulative model without making the parallel lines assumption

$$g(\Pr(Y \le i \mid \mathbf{x})) = (1, \mathbf{x}')\boldsymbol{\beta}_i, \quad 1 \le i \le k$$

where $g(\cdot)$ is the link function, and $\boldsymbol{\beta}_i = (\alpha_i, \beta_{i1}, \ldots, \beta_{is})'$ is a vector of unknown parameters consisting of an intercept α_i and s slope parameters $\beta_{i1}, \ldots, \beta_{is}$. The parameter vector for this general cumulative model is

$$\boldsymbol{\beta} = (\boldsymbol{\beta}_1', \ldots, \boldsymbol{\beta}_k')'$$

Under the null hypothesis of parallelism $H_0\colon \beta_{1m} = \beta_{2m} = \cdots = \beta_{km}, 1 \le m \le s$, there is a single common slope parameter for each of the s explanatory variables. Let $\beta_1, \ldots, \beta_s$ be the common slope parameters. Let $\widehat{\alpha}_1, \ldots, \widehat{\alpha}_k$ and $\widehat{\beta}_1, \ldots, \widehat{\beta}_s$ be the MLEs of the intercept parameters and the common slope parameters. Then, under H_0, the MLE of $\boldsymbol{\beta}$ is

$$\widehat{\boldsymbol{\beta}}_{H_0} = (\widehat{\boldsymbol{\beta}}_1', \ldots, \widehat{\boldsymbol{\beta}}_k')' \quad \text{with} \quad \widehat{\boldsymbol{\beta}}_i = (\widehat{\alpha}_i, \widehat{\beta}_1, \ldots, \widehat{\beta}_s)' \quad 1 \le i \le k$$

and the chi-square score statistic $\mathbf{g}'(\widehat{\boldsymbol{\beta}}_{H_0})\mathbf{I}^{-1}(\widehat{\boldsymbol{\beta}}_{H_0})\mathbf{g}(\widehat{\boldsymbol{\beta}}_{H_0})$ has an asymptotic chi-square distribution with $s(k-1)$ degrees of freedom. This tests the parallel lines assumption by testing the equality of separate slope parameters simultaneously for all explanatory variables.

Confidence Intervals for Parameters

There are two methods of computing confidence intervals for the regression parameters. One is based on the profile-likelihood function, and the other is based on the asymptotic normality of the parameter estimators. The latter is not as time-consuming as the former, because it does not involve an iterative scheme; however, it is not thought to be as accurate as the former, especially with small sample size. You use the CLPARM= option to request confidence intervals for the parameters.

Likelihood Ratio-Based Confidence Intervals

The likelihood ratio-based confidence interval is also known as the profile-likelihood confidence interval. The construction of this interval is derived from the asymptotic χ^2 distribution of the generalized likelihood ratio test (Venzon and Moolgavkar 1988). Suppose that the parameter vector is $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_s)'$ and you want to compute a confidence interval for β_j. The profile-likelihood function for $\beta_j = \gamma$ is defined as

$$l_j^*(\gamma) = \max_{\boldsymbol{\beta} \in \mathcal{B}_j(\gamma)} l(\boldsymbol{\beta})$$

where $\mathcal{B}_j(\gamma)$ is the set of all $\boldsymbol{\beta}$ with the jth element fixed at γ, and $l(\boldsymbol{\beta})$ is the log-likelihood function for $\boldsymbol{\beta}$. (The penalized log-likelihood function is used when you specify the FIRTH option in the MODEL statement.) If $l_{\max} = l(\widehat{\boldsymbol{\beta}})$ is the log likelihood evaluated at the maximum likelihood estimate $\widehat{\boldsymbol{\beta}}$, then $2(l_{\max} - l_j^*(\beta_j))$ has a limiting chi-square distribution with one degree of freedom if β_j is the true parameter value. Let $l_0 = l_{\max} - 0.5\chi_1^2(1-\alpha)$, where $\chi_1^2(1-\alpha)$ is the $100(1-\alpha)$th percentile of the chi-square distribution with one degree of freedom. A $100(1-\alpha)\%$ confidence interval for β_j is

$$\{\gamma : l_j^*(\gamma) \geq l_0\}$$

The endpoints of the confidence interval are found by solving numerically for values of β_j that satisfy equality in the preceding relation. To obtain an iterative algorithm for computing the confidence limits, the log-likelihood function in a neighborhood of $\boldsymbol{\beta}$ is approximated by the quadratic function

$$\tilde{l}(\boldsymbol{\beta} + \boldsymbol{\delta}) = l(\boldsymbol{\beta}) + \boldsymbol{\delta}'\mathbf{g} + \frac{1}{2}\boldsymbol{\delta}'\mathbf{V}\boldsymbol{\delta}$$

where $\mathbf{g} = \mathbf{g}(\boldsymbol{\beta})$ is the gradient vector and $\mathbf{V} = \mathbf{V}(\boldsymbol{\beta})$ is the Hessian matrix. The increment $\boldsymbol{\delta}$ for the next iteration is obtained by solving the likelihood equations

$$\frac{d}{d\boldsymbol{\delta}}\{\tilde{l}(\boldsymbol{\beta} + \boldsymbol{\delta}) + \lambda(\mathbf{e}_j'\boldsymbol{\delta} - \gamma)\} = \mathbf{0}$$

where λ is the Lagrange multiplier, $\mathbf{e}_j$ is the jth unit vector, and γ is an unknown constant. The solution is

$$\boldsymbol{\delta} = -\mathbf{V}^{-1}(\mathbf{g} + \lambda\mathbf{e}_j)$$

By substituting this $\boldsymbol{\delta}$ into the equation $\tilde{l}(\boldsymbol{\beta} + \boldsymbol{\delta}) = l_0$, you can estimate λ as

$$\lambda = \pm\left(\frac{2(l_0 - l(\boldsymbol{\beta}) + \frac{1}{2}\mathbf{g}'\mathbf{V}^{-1}\mathbf{g})}{\mathbf{e}_j'\mathbf{V}^{-1}\mathbf{e}_j}\right)^{\frac{1}{2}}$$

The upper confidence limit for β_j is computed by starting at the maximum likelihood estimate of $\boldsymbol{\beta}$ and iterating with positive values of λ until convergence is attained. The process is repeated for the lower confidence limit by using negative values of λ.

Convergence is controlled by the value ϵ specified with the PLCONV= option in the MODEL statement (the default value of ϵ is 1E–4). Convergence is declared on the current iteration if the following two conditions are satisfied:

$$|l(\boldsymbol{\beta}) - l_0| \leq \epsilon$$

and

$$(\mathbf{g} + \lambda\mathbf{e}_j)'\mathbf{V}^{-1}(\mathbf{g} + \lambda\mathbf{e}_j) \leq \epsilon$$

Wald Confidence Intervals

Wald confidence intervals are sometimes called the normal confidence intervals. They are based on the asymptotic normality of the parameter estimators. The $100(1-\alpha)\%$ Wald confidence interval for β_j is given by

$$\widehat{\beta}_j \pm z_{1-\alpha/2}\widehat{\sigma}_j$$

where z_p is the $100p$th percentile of the standard normal distribution, $\widehat{\beta}_j$ is the maximum likelihood estimate of β_j, and $\widehat{\sigma}_j$ is the standard error estimate of $\widehat{\beta}_j$.

Odds Ratio Estimation

Consider a dichotomous response variable with outcomes *event* and *nonevent*. Consider a dichotomous risk factor variable X that takes the value 1 if the risk factor is present and 0 if the risk factor is absent. According to the logistic model, the log odds function, $\text{logit}(X)$, is given by

$$\text{logit}(X) \equiv \log\left(\frac{\Pr(event \mid X)}{\Pr(nonevent \mid X)}\right) = \alpha + X\beta$$

The odds ratio ψ is defined as the ratio of the odds for those with the risk factor ($X = 1$) to the odds for those without the risk factor ($X = 0$). The log of the odds ratio is given by

$$\log(\psi) \equiv \log(\psi(X = 1, X = 0)) = \text{logit}(X = 1) - \text{logit}(X = 0) = (\alpha + 1 \times \beta) - (\alpha + 0 \times \beta) = \beta$$

In general, the odds ratio can be computed by exponentiating the difference of the logits between any two population profiles. This is the approach taken by the ODDSRATIO statement, so the computations are available regardless of parameterization, interactions, and nestings. However, as shown in the preceding equation for $\log(\psi)$, odds ratios of main effects can be computed as functions of the parameter estimates, and the remainder of this section is concerned with this methodology.

The parameter, β, associated with X represents the change in the log odds from $X = 0$ to $X = 1$. So the odds ratio is obtained by simply exponentiating the value of the parameter associated with the risk factor. The odds ratio indicates how the odds of the event change as you change X from 0 to 1. For instance, $\psi = 2$ means that the odds of an event when $X = 1$ are twice the odds of an event when $X = 0$. You can also express this as follows: the percent change in the odds of an event from $X = 0$ to $X = 1$ is $(\psi - 1)100\% = 100\%$.

Suppose the values of the dichotomous risk factor are coded as constants a and b instead of 0 and 1. The odds when $X = a$ become $\exp(\alpha + a\beta)$, and the odds when $X = b$ become $\exp(\alpha + b\beta)$. The odds ratio corresponding to an increase in X from a to b is

$$\psi = \exp[(b-a)\beta] = [\exp(\beta)]^{b-a} \equiv [\exp(\beta)]^c$$

Note that for any a and b such that $c = b - a = 1$, $\psi = \exp(\beta)$. So the odds ratio can be interpreted as the change in the odds for any increase of one unit in the corresponding risk factor. However, the change in odds for some amount other than one unit is often of greater interest. For example, a change of one pound in body weight might be too small to be considered important, while a change of 10 pounds might be more meaningful. The odds ratio for a change in X from a to b is estimated by raising the odds ratio estimate for a unit change in X to the power of $c = b - a$ as shown previously.

For a polytomous risk factor, the computation of odds ratios depends on how the risk factor is parameterized. For illustration, suppose that Race is a risk factor with four categories: White, Black, Hispanic, and Other.

For the effect parameterization scheme (PARAM=EFFECT) with White as the reference group (REF='White'), the design variables for Race are as follows:

	Design Variables		
Race	X_1	X_2	X_3
Black	1	0	0
Hispanic	0	1	0
Other	0	0	1
White	–1	–1	–1

The log odds for Black is

$$\begin{aligned} \text{logit(Black)} &= \alpha + (X_1 = 1)\beta_1 + (X_2 = 0)\beta_2 + (X_3 = 0)\beta_3 \\ &= \alpha + \beta_1 \end{aligned}$$

The log odds for White is

$$\begin{aligned} \text{logit(White)} &= \alpha + (X_1 = -1)\beta_1 + (X_2 = -1)\beta_2 + (X_3 = -1)\beta_3 \\ &= \alpha - \beta_1 - \beta_2 - \beta_3 \end{aligned}$$

Therefore, the log odds ratio of Black versus White becomes

$$\begin{aligned} \log(\psi(\text{Black}, \text{White})) &= \text{logit(Black)} - \text{logit(White)} \\ &= 2\beta_1 + \beta_2 + \beta_3 \end{aligned}$$

For the reference cell parameterization scheme (PARAM=REF) with White as the reference cell, the design variables for race are as follows:

	Design Variables		
Race	X_1	X_2	X_3
Black	1	0	0
Hispanic	0	1	0
Other	0	0	1
White	0	0	0

The log odds ratio of Black versus White is given by

$$\begin{aligned} \log(\psi(\text{Black}, \text{White})) &= \text{logit(Black)} - \text{logit(White)} \\ &= (\alpha + (X_1 = 1)\beta_1 + (X_2 = 0)\beta_2 + (X_3 = 0)\beta_3) - \\ &\quad (\alpha + (X_1 = 0)\beta_1 + (X_2 = 0)\beta_2 + (X_3 = 0)\beta_3) \\ &= \beta_1 \end{aligned}$$

For the GLM parameterization scheme (PARAM=GLM), the design variables are as follows:

	Design Variables			
Race	X_1	X_2	X_3	X_4
Black	1	0	0	0
Hispanic	0	1	0	0
Other	0	0	1	0
White	0	0	0	1

The log odds ratio of Black versus White is

$$\begin{aligned}
\log(\psi(\text{Black}, \text{White})) &= \text{logit(Black)} - \text{logit(White)} \\
&= (\alpha + (X_1 = 1)\beta_1 + (X_2 = 0)\beta_2 + (X_3 = 0)\beta_3 + (X_4 = 0)\beta_4) - \\
&\quad (\alpha + (X_1 = 0)\beta_1 + (X_2 = 0)\beta_2 + (X_3 = 0)\beta_3 + (X_4 = 1)\beta_4) \\
&= \beta_1 - \beta_4
\end{aligned}$$

Consider the hypothetical example of heart disease among race in Hosmer and Lemeshow (2000, p. 56). The entries in the following contingency table represent counts:

	Race			
Disease Status	**White**	**Black**	**Hispanic**	**Other**
Present	5	20	15	10
Absent	20	10	10	10

The computation of odds ratio of Black versus White for various parameterization schemes is tabulated in Table 79.12.

Table 79.12 Odds Ratio of Heart Disease Comparing Black to White

	Parameter Estimates				
PARAM=	$\widehat{\beta}_1$	$\widehat{\beta}_2$	$\widehat{\beta}_3$	$\widehat{\beta}_4$	**Odds Ratio Estimates**
EFFECT	0.7651	0.4774	0.0719		$\exp(2 \times 0.7651 + 0.4774 + 0.0719) = 8$
REF	2.0794	1.7917	1.3863		$\exp(2.0794) = 8$
GLM	2.0794	1.7917	1.3863	0.0000	$\exp(2.0794) = 8$

Because the log odds ratio ($\log(\psi)$) is a linear function of the parameters, the Wald confidence interval for $\log(\psi)$ can be derived from the parameter estimates and the estimated covariance matrix. Confidence intervals for the odds ratios are obtained by exponentiating the corresponding confidence limits for the log odd ratios. In the displayed output of PROC LOGISTIC, the "Odds Ratio Estimates" table contains the odds ratio estimates and the corresponding 95% Wald confidence intervals. For continuous explanatory variables, these odds ratios correspond to a unit increase in the risk factors.

To customize odds ratios for specific units of change for a continuous risk factor, you can use the UNITS statement to specify a list of relevant units for each explanatory variable in the model. Estimates of these customized odds ratios are given in a separate table. Let (L_j, U_j) be a confidence interval for $\log(\psi)$. The corresponding lower and upper confidence limits for the customized odds ratio $\exp(c\beta_j)$ are $\exp(cL_j)$ and $\exp(cU_j)$, respectively (for $c > 0$), or $\exp(cU_j)$ and $\exp(cL_j)$, respectively (for $c < 0$). You use the CLODDS= option or ODDSRATIO statement to request the confidence intervals for the odds ratios.

For a generalized logit model, odds ratios are computed similarly, except k odds ratios are computed for each effect, corresponding to the k logits in the model.

Linear Predictor, Predicted Probability, and Confidence Limits

This section describes how predicted probabilities and confidence limits are calculated by using the maximum likelihood estimates (MLEs) obtained from PROC LOGISTIC. For a specific example, see the section "Getting Started: LOGISTIC Procedure" on page 5942. Predicted probabilities and confidence limits can be output to a data set with the OUTPUT statement.

Binary and Cumulative Response Models

For a vector of explanatory variables $\mathbf{x}$, the linear predictor

$$\eta_i = g(\Pr(Y \le i \mid \mathbf{x})) = \alpha_i + \mathbf{x}'\boldsymbol{\beta} \quad 1 \le i \le k$$

is estimated by

$$\hat{\eta}_i = \widehat{\alpha}_i + \mathbf{x}'\widehat{\boldsymbol{\beta}}$$

where $\widehat{\alpha}_i$ and $\widehat{\boldsymbol{\beta}}$ are the MLEs of α_i and $\boldsymbol{\beta}$. The estimated standard error of η_i is $\hat{\sigma}(\hat{\eta}_i)$, which can be computed as the square root of the quadratic form $(1, \mathbf{x}')\widehat{\mathbf{V}}_{\mathbf{b}}(1, \mathbf{x}')'$, where $\widehat{\mathbf{V}}_{\mathbf{b}}$ is the estimated covariance matrix of the parameter estimates. The asymptotic $100(1-\alpha)\%$ confidence interval for η_i is given by

$$\hat{\eta}_i \pm z_{\alpha/2}\hat{\sigma}(\hat{\eta}_i)$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile point of a standard normal distribution.

The predicted probability and the $100(1-\alpha)\%$ confidence limits for $\pi_i = \Pr(Y \le i \mid \mathbf{x})$ are obtained by back-transforming the corresponding measures for the linear predictor, as shown in the following table:

Link	Predicted Probability	100(1–α)% Confidence Limits
LOGIT	$1/(1+\exp(-\hat{\eta}_i))$	$1/(1+\exp(-\hat{\eta}_i \pm z_{\alpha/2}\hat{\sigma}(\hat{\eta}_i)))$
PROBIT	$\Phi(\hat{\eta}_i)$	$\Phi(\hat{\eta}_i \pm z_{\alpha/2}\hat{\sigma}(\hat{\eta}_i))$
CLOGLOG	$1-\exp(-\exp(\hat{\eta}_i))$	$1-\exp(-\exp(\hat{\eta}_i \pm z_{\alpha/2}\hat{\sigma}(\hat{\eta}_i)))$

The CONTRAST statement also enables you to estimate the exponentiated contrast, $e^{\hat{\eta}_i}$. The corresponding standard error is $e^{\hat{\eta}_i}\hat{\sigma}(\hat{\eta}_i)$, and the confidence limits are computed by exponentiating those for the linear predictor: $\exp\{\hat{\eta}_i \pm z_{\alpha/2}\hat{\sigma}(\hat{\eta}_i)\}$.

Generalized Logit Model

For a vector of explanatory variables $\mathbf{x}$, define the linear predictors $\eta_i = \alpha_i + \mathbf{x}'\boldsymbol{\beta}_i$, and let π_i denote the probability of obtaining the response value i:

$$\pi_i = \begin{cases} \pi_{k+1}e^{\eta_i} & 1 \le i \le k \\ \dfrac{1}{1+\sum_{j=1}^{k} e^{\eta_j}} & i = k+1 \end{cases}$$

By the *delta method*,

$$\sigma^2(\pi_i) = \left(\frac{\partial \pi_i}{\partial \boldsymbol{\beta}}\right)' \mathbf{V}(\boldsymbol{\beta}) \frac{\partial \pi_i}{\partial \boldsymbol{\beta}}$$

A $100(1-\alpha)\%$ confidence level for π_i is given by

$$\widehat{\pi}_i \pm z_{\alpha/2} \hat{\sigma}(\widehat{\pi}_i)$$

where $\widehat{\pi}_i$ is the estimated expected probability of response i, and $\hat{\sigma}(\widehat{\pi}_i)$ is obtained by evaluating $\sigma(\pi_i)$ at $\boldsymbol{\beta} = \widehat{\boldsymbol{\beta}}$.

Note that the contrast $\hat{\eta}_i$ and exponentiated contrast $e^{\hat{\eta}_i}$, their standard errors, and their confidence intervals are computed in the same fashion as for the cumulative response models, replacing $\boldsymbol{\beta}$ with $\boldsymbol{\beta}_i$.

Classification Table

For binary response data, the response is either an *event* or a *nonevent*. In PROC LOGISTIC, the response with Ordered Value 1 is regarded as the event, and the response with Ordered Value 2 is the nonevent. PROC LOGISTIC models the probability of the event. From the fitted model, a predicted event probability can be computed for each observation. A method to compute a reduced-bias estimate of the predicted probability is given in the section "Predicted Probability of an Event for Classification" on page 6049. If the (reduced-bias) predicted event probability exceeds or equals some cutpoint value $z \in [0, 1]$, the observation is predicted to be an event observation; otherwise, it is predicted to be a nonevent observation.

Suppose that n_1 of n individuals experience an event, such as a disease, and the remaining $n_2 = n - n_1$ individuals do not experience that event (that is, they have a nonevent response). The 2×2 frequency (classification, confusion, decision, error) table in Table 79.13 is obtained by cross-classifying the observed and predicted responses, where n_{ij} is the total number of observations that are observed to have $Y = i$ and are classified into j. In this table, let $Y = 1$ denote an observed event and $Y = 2$ denote a nonevent, and let the decision rule, D, classify an observation as an event when $\hat{\pi} \geq z$; $D = 1$ indicates that the observation is classified as an event, and $D = 2$ indicates that the observation is classified as a nonevent.

Table 79.13 Classification Matrix

	$D = 1$ ($\hat{\pi} \geq z$)	$D = 2$ ($\hat{\pi} < z$)	**Total**
$Y = 1$ (event)	n_{11}	n_{12}	n_1
$Y = 2$ (nonevent)	n_{21}	n_{22}	n_2

The CTABLE option produces this table, and the PPROB= option selects one or more cutpoints z. Each cutpoint generates a classification table. If the PEVENT= option is also specified, a classification table is produced for each combination of PEVENT= and PPROB= values.

The cells of the classification matrix in Table 79.13 have the following interpretations:

n_{11} the number of *true positives*, which is the number of event observations that are correctly classified as events

n_{21} the number of *false positives*, which is the number of nonevent observations that are incorrectly classified as events

n_{12} the number of *false negatives*, which is the number of event observations that are incorrectly classified as nonevents

n_{22} the number of *true negatives*, which is the number of nonevent observations that are correctly classified as nonevents

n_1 the total number of actual events

n_2 the total number of actual nonevents

The statistics in Table 79.14 are computed from the classification table in Table 79.13.

Table 79.14 Statistics from the Classification Matrix with Cutpoint z

Statistic	Equation	Value
Sensitivity	$\Pr(D=1\|Y=1)$	n_{11}/n_1
1–specificity	$\Pr(D=1\|Y=2)$	n_{21}/n_2
Correct classification rate	$\Pr(Y=1\&D=1)+\Pr(Y=2\&D=2)$	$(n_{11}+n_{22})/n$
Misclassification rate	$\Pr(Y=1\&D=2)+\Pr(Y=2\&D=1)$	$(n_{12}+n_{21})/n$
Positive predictive value	$\Pr(Y=1\|D=1)$	$n_{11}/(n_{11}+n_{21})$
Negative predictive value	$\Pr(Y=2\|D=2)$	$n_{22}/(n_{12}+n_{22})$

The accuracy of the classification is measured by its ability to predict events and nonevents correctly. *Sensitivity* (true positive fraction, TPF, or recall) is the proportion of event responses that are predicted to be events. *Specificity* (true negative fraction, 1–FPF) is the proportion of nonevent responses that are predicted to be nonevents.

You can also measure accuracy by how well the classification predicts the response. The *positive predictive value* (precision, PPV) is the proportion of observations classified as events that are correctly classified. The *negative predictive value* (NPV) is the proportion of observations classified as nonevents that are correctly classified. The *correct classification rate* (accuracy, PC) is the proportion of observations that are correctly classified, and the *misclassification rate* (error rate) is the proportion of observations that are incorrectly classified. Given prior probabilities (*prevalence*, $\Pr(Y=1)$) that are specified by the PEVENT= option, you can compute these conditional probabilities as posterior probabilities by using Bayes' theorem, as shown in the following section.

NOTE: Current literature defines the false positive fraction as FPF= n_{21}/n_2 and the false negative fraction as FNF= n_{12}/n_1, and 1–PPV is called the false discovery rate and 1–NPV is called the false omission rate.

Positive Predictive Values, Negative Predictive Values, and Correct Classification Rates Using Bayes' Theorem

If the prevalence of the disease in the population $\Pr(Y=1)$ is provided by the value of the PEVENT= option, then PROC LOGISTIC uses Bayes' theorem to modify the PPV, NPV, and PC as follows (Fleiss, Levin, and Paik 2003):

$$
\begin{aligned}
\text{PPV} &= \Pr(Y=1|D=1) = \frac{\Pr(D=1|Y=1)\Pr(Y=1)}{\Pr(D=1|Y=2)+\Pr(Y=1)[\Pr(D=1|Y=1)-\Pr(D=1|Y=2)]} \\
\text{NPV} &= \Pr(Y=2|D=2) = \frac{\Pr(D=2|Y=2)(1-\Pr(Y=1))}{\Pr(D=2|Y=2)+\Pr(Y=1)[\Pr(D=2|Y=1)-\Pr(D=2|Y=2)]} \\
\text{PC} &= \Pr(Y=1\&D=1)+\Pr(Y=2\&D=2) \\
&= \Pr(D=1|Y=1)\Pr(Y=1)+\Pr(D=2|Y=2)[1-\Pr(Y=1)]
\end{aligned}
$$

If you do not specify the PEVENT= option, then PROC LOGISTIC uses the sample proportion of diseased individuals; that is, $\Pr(Y=1)=n_1/n$. In such a case, the preceding values reduce to those in Table 79.14. Note that for a stratified sampling or case-control situation in which n_1 and n_2 are chosen a priori, n_1/n is not a desirable estimate of $\Pr(Y=1)$, so you should specify the PEVENT= option.

Predicted Probability of an Event for Classification

When you classify a set of binary data, if the same observations used to fit the model are also used to estimate the classification error, the resulting error-count estimate is biased. One way of reducing the bias is to remove the binary observation to be classified from the data, reestimate the parameters of the model, and then classify the observation based on the new parameter estimates. However, it would be costly to fit the model by leaving out each observation one at a time. The LOGISTIC procedure provides a less expensive one-step approximation to the preceding parameter estimates. Let $\widehat{\boldsymbol{\beta}}$ be the MLE of the parameter vector $(\alpha, \beta_1, \ldots, \beta_s)'$ based on all observations. Let $\widehat{\boldsymbol{\beta}}_{(j)}$ denote the MLE computed without the jth observation. The one-step estimate of $\widehat{\boldsymbol{\beta}}_{(j)}$ is given by

$$
\widehat{\boldsymbol{\beta}}^1_{(j)} = \widehat{\boldsymbol{\beta}} - \frac{w_j(y_j-\widehat{\pi}_j)}{1-h_j}\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})\begin{pmatrix} 1 \\ \mathbf{x}_j \end{pmatrix}
$$

where

y_j	is 1 for an observed event response and 0 otherwise
w_j	is the weight of the observation
$\widehat{\pi}_j$	is the predicted event probability based on $\widehat{\boldsymbol{\beta}}$
h_j	is the hat diagonal element (defined on page 6060) with $n_j = 1$ and $r_j = y_j$
$\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})$	is the estimated covariance matrix of $\widehat{\boldsymbol{\beta}}$

Goodness-of-Fit Tests

The following subsection describes the standard Pearson and deviance goodness-of-fit tests. To be valid, these two tests require sufficient replication within subpopulations. When there are continuous predictors in the model, the data are often too sparse to use these statistics. The remaining subsections describe tests that are all designed to be valid in such situations.

Let N be the number of observations in your data, and let m denote the number of subpopulation profiles. You can use the AGGREGATE (or AGGREGATE=) option in the MODEL statement to define the subpopulation profiles. If you omit this option, each observation is regarded as coming from a separate subpopulation, and m=N.

Let $k+1$ be the number of response levels and p be the number of parameters that are estimated. For the jth profile (or observation) and the ith response level, let w_{ij} be the total weight (sum of the product of the frequencies and the weights of the observations within that profile), let $w_j = \sum_{i=1}^{k+1} w_{ij}$, and let $\widehat{\pi}_{ij}$ be the fitted probability. Let r_{ij} be the number of responses with level i, and let $n_j = \sum_i r_{ij}$ be the number of trials. Let $\widehat{\pi}_{ij}$ denote the predicted probability of response level i, $1 \le i \le k+1$.

Pearson and Deviance Goodness-of-Fit Test

The Pearson chi-square statistic χ^2_{P} and the deviance χ^2_{D} are given by

$$\chi^2_{\mathrm{P}} = \sum_{j=1}^{m}\sum_{i=1}^{k+1} \frac{(r_{ij} - n_j\widehat{\pi}_{ij})^2}{n_j\widehat{\pi}_{ij}}$$

$$\chi^2_{\mathrm{D}} = 2\sum_{j=1}^{m}\sum_{i=1}^{k+1} r_{ij}\log\left(\frac{r_{ij}}{n_j\widehat{\pi}_{ij}}\right)$$

Each of these chi-square statistics has $mk - p$ degrees of freedom

A large difference between the Pearson statistic and the deviance provides some evidence that the data are too sparse to use either statistic.

Without the AGGREGATE (or AGGREGATE=) option, the Pearson chi-square statistic and the deviance are calculated only for events/trials syntax.

The Hosmer-Lemeshow Goodness-of-Fit Test

Sufficient replication within subpopulations is required to make the Pearson and deviance goodness-of-fit tests valid. When there are one or more continuous predictors in the model, the data are often too sparse to use these statistics. Hosmer and Lemeshow (2000) proposed a statistic that they show, through simulation, is distributed as chi-square when there is no replication in any of the subpopulations. Fagerland, Hosmer, and Bofin (2008) and Fagerland and Hosmer (2013, 2016) extend this test to polytomous response models.

The observations are sorted in increasing order of a scored value. For binary response variables, the scored value of an observation is its estimated event probability. The event is the response level specified in the response variable option EVENT=, the response level that is not specified in the REF= option, or, if neither of these options was specified, the response level identified in the "Response Profiles" table as "Ordered Value 1." For nominal response variables (LINK=GLOGIT), the scored value of an observation is 1 minus

the estimated probability of the reference level (specified using the REF= option). For ordinal response variables, the scored value of an observation is $\sum_{i=1}^{K} i\widehat{\pi}_i$, where K is the number of response levels and $\widehat{\pi}_i$ is the predicted probability of the ith ordered response.

The observations (and frequencies) are then combined into G groups. By default $G = 10$, but you can specify $G \geq 5$ with the NGROUPS= suboption of the LACKFIT option in the MODEL statement. For single-trial syntax, observations with identical scored values are combined and are placed in the same group. Let F be the total frequency. The target frequency for each group is $T = \lfloor F/G + 0.5 \rfloor$, which is the integer part of $F/G + 0.5$. Load the first group (g_j, $j = 1$) with the observation that has the smallest scored value and with frequency f_1, and let the next-smallest observation have a frequency of f. PROC LOGISTIC performs the following steps for each observation to create the groups:

1. If $j = G$, then add this observation to group g_j.
2. Otherwise, if $f_j < T$ and $f_j + \lfloor f/2 \rfloor \leq T$, then add this observation to group g_j.
3. Otherwise, start loading the next group (g_{j+1}) with $f_{j+1} = f$, and set $j = j + 1$.

If the final group g_j has frequency $f_j < \frac{F}{2G}$, then add these observations to the preceding group. The total number of groups actually created, g, can be less than G. There must be at least three groups in order for the Hosmer-Lemeshow statistic to be computed.

For binary response variables, the Hosmer-Lemeshow goodness-of-fit statistic is obtained by calculating the Pearson chi-square statistic from the $2 \times g$ table of observed and expected frequencies, where g is the number of groups. The statistic is written

$$\chi^2_{\mathrm{HL}} = \sum_{j=1}^{g} \frac{(O_j - F_j\bar{\pi}_j)^2}{F_j\bar{\pi}_j(1-\bar{\pi}_j)}$$

where F_j is the total frequency of subjects in the jth group, O_j is the total frequency of event outcomes in the jth group, and $\bar{\pi}_j$ is the average estimated predicted probability of an event outcome for the jth group. (Note that the predicted probabilities are computed as shown in the section "Linear Predictor, Predicted Probability, and Confidence Limits" on page 6046 and are not the cross validated estimates discussed in the section "Classification Table" on page 6047.) The Hosmer-Lemeshow statistic is then compared to a chi-square distribution with $(g - r)$ degrees of freedom, where the value of r can be specified in the DFREDUCE= suboption of the LACKFIT option in the MODEL statement. The default is $r = 2$.

For polytomous response variables, the Pearson chi-square statistic is computed from a $2K \times g$ table of observed and expected frequencies,

$$\chi^2_{\mathrm{HL}} = \sum_{j=1}^{g} \sum_{k=1}^{K} \frac{(O_{jk} - E_{jk})^2}{E_{jk}}$$

where O_{jk} is the sum of the observed frequencies and E_{jk} is the sum of the model predicted probabilities of the observations in group j with response k. The Hosmer-Lemeshow statistic is then compared to a chi-square distribution. The number of degrees of freedom for this test of cumulative and adjacent-category logit models with the equal-slopes assumption is given by Fagerland and Hosmer (2013) and Fagerland and Hosmer (2016) as $(g–r)(K–1)+(K–2)$; PROC LOGISTIC uses this number for all models that make the equal-slopes assumption. The number of degrees of freedom for this test of the generalized logit model is given by Fagerland, Hosmer, and Bofin (2008) as $(g–r)(K–1)$, where K is the number of response levels; PROC LOGISTIC uses this number for all models that do not make the equal-slopes assumption. The degrees of freedom can also be specified using the DF= suboption of the LACKFIT option in the MODEL statement.

Large values of χ^2_{HL} (and small p-values) indicate a lack of fit of the model.

Goodness-of-Fit Tests with Sparse Data

The tests in this section are valid even when the data are sparse and there is very little or no replication in the data. These tests are currently available only for binary logistic regression models, and they are reported in the "Goodness-of-Fit Tests" table when you specify the GOF option in the MODEL statement. Let $\widehat{\pi}_j$ denote the predicted event probability, and let $\widehat{\mathbf{V}}$ be the covariance matrix for the fitted model.

Information Matrix Test

The general misspecification test of White (1982) is applied by Orme (1988) to binary response data. The design vector $\mathbf{x}_j$ for observation j is expanded by the upper triangular matrix of $\mathbf{x}_j\mathbf{x}_j'$; that is, by vech($\mathbf{x}_j$). The $\mathbf{x}_j$ are scaled by $\sqrt{\hat{\pi}_j(1-\hat{\pi}_j)}$, the vech($\mathbf{x}_j$) values are scaled by $\sqrt{\hat{\pi}_j(1-\hat{\pi}_j)}(1-2\hat{\pi}_j)$, and new response values are created by using the binary form of the residual $e_j = y_j - \hat{\pi}_j$:

$$Y^* = \frac{e_j}{\sqrt{\hat{\pi}_j(1-\hat{\pi}_j)}}$$

The model sum of squares from a linear regression of Y^* against the expanded set of covariates has a chi-square distribution with degrees of freedom equal to the number of scaled vech($\mathbf{x}_j$) covariates that are nondegenerate. This test is labeled "Information Matrix" in the "Goodness-of-Fit Tests" table.

Kuss (2002) modifies this test by expanding the design matrix using only the diagonal values of $\mathbf{x}_j\mathbf{x}_j'$. This test is labeled "Information Matrix Diagonal" in the "Goodness-of-Fit Tests" table.

Osius-Rojek Test

Osius and Rojek (1992) use fixed-cells asymptotics to derive the mean and variance of the Pearson chi-square statistic. The mean is m, the number of subpopulation profiles, and the variance is as follows:

$$\sigma^2 = 2m + \sum_{j=1}^{m} \frac{1}{w_j n_j}\left(\frac{1}{\hat{\pi}_j(1-\hat{\pi}_j)} - 6\right) - \mathbf{c}'\widehat{\mathbf{V}}\mathbf{c} \quad \text{where} \quad \mathbf{c} = \sum_{j=1}^{m}(1-2\hat{\pi}_j)\mathbf{x}_j$$

Standardize the Pearson statistic, $\dfrac{\chi^2_{\mathrm{P}} - m}{\sigma}$, then square it to obtain a χ^2_1 test.

Unweighted Residual Sum-of-Squares Test

Copas (1989) bases a test on the asymptotic normal distribution of the numerator of the Pearson chi-square statistic

$$S = \sum_{j=1}^{N}\sum_{i=1}^{2} w_{ij}(r_{ij} - n_j\widehat{\pi}_{ij})^2$$

which has mean $\mathrm{E}(S) = \mathrm{trace}(\widehat{\mathbf{V}})$. Hosmer et al. (1997) simplify the distribution of $S - \mathrm{trace}(\widehat{\mathbf{V}})$ for binary response models, so that its variance, σ^2, is the residual sum of squares of a regression of a vector with entries $(1-2\widehat{\pi}_j)$ on $\mathbf{X}$ with weights $\widehat{\mathbf{V}}$. Standardize the statistic, $\dfrac{S - \mathrm{trace}(\widehat{\mathbf{V}})}{\sigma}$, then square it to obtain a χ^2_1 test.

Spiegelhalter Test

Spiegelhalter (1986) derives a test based on the Brier score, written in binary form as

$$B = \frac{1}{W}\sum_{j=1}^{N} w_j(y_j - \hat{\pi}_j)^2$$

where $W = \sum_{j=1}^{N} w_j$. B is asymptotically normal with

$$\mathrm{E}(B) = \frac{1}{W}\sum_{j=1}^{N} w_j\hat{\pi}_j(1-\hat{\pi}_j)$$

and

$$\mathrm{Var}(B) = \frac{1}{W^2}\sum_{j=1}^{N} w_j(1-2\hat{\pi}_j)^2\hat{\pi}_j(1-\hat{\pi}_j)$$

Standardize the Brier score, $\dfrac{B-\mathrm{E}(B)}{\sqrt{\mathrm{Var}(B)}}$, then square it to obtain a χ^2_1 test.

Stukel Test

Stukel (1988) adds two covariates to the model and tests that they are insignificant. For a binary response, where $\eta_j = \mathbf{x}'_j\boldsymbol{\beta}$ and $I(\cdot)$ is the indicator function, add the following covariates:

$$\begin{aligned} z_a &= \eta_j^2 I(\eta_j \geq 0) \\ z_b &= \eta_j^2 I(\eta_j < 0) \end{aligned}$$

Then use a score test (see the section "Score Statistics and Tests" on page 6039) to test whether this larger model is significantly different from the fitted model.

Overdispersion

This section uses the definitions that are provided in the section "Goodness-of-Fit Tests" on page 6050.

For a correctly specified model, the Pearson chi-square statistic and the deviance, divided by their degrees of freedom, should be approximately equal to one. When their values are much larger than one, the assumption of binomial variability might not be valid and the data are said to exhibit overdispersion. Underdispersion, which results in the ratios being less than one, occurs less often in practice.

When fitting a model, there are several problems that can cause the goodness-of-fit statistics to exceed their degrees of freedom. Among these are such problems as outliers in the data, using the wrong link function, omitting important terms from the model, and needing to transform some predictors. These problems should be eliminated before proceeding to use the following methods to correct for overdispersion.

Rescaling the Covariance Matrix

One way of correcting overdispersion is to multiply the covariance matrix by a dispersion parameter. This method assumes that the sample sizes in each subpopulation are approximately equal. You can supply the value of the dispersion parameter directly, or you can estimate the dispersion parameter based on either the Pearson chi-square statistic or the deviance for the fitted model.

The dispersion parameter is estimated by

$$\widehat{\sigma^2} = \begin{cases} \chi^2_P/(mk-p) & \text{SCALE=PEARSON} \\ \chi^2_D/(mk-p) & \text{SCALE=DEVIANCE} \\ (constant)^2 & \text{SCALE=}constant \end{cases}$$

In order for the Pearson statistic and the deviance to be distributed as chi-square, there must be sufficient replication within the subpopulations. When this is not true, the data are sparse, and the p-values for these statistics are not valid and should be ignored. Similarly, these statistics, divided by their degrees of freedom, cannot serve as indicators of overdispersion. A large difference between the Pearson statistic and the deviance provides some evidence that the data are too sparse to use either statistic.

Note that the parameter estimates are not changed by this method. However, their standard errors are adjusted for overdispersion, affecting their significance tests.

Williams' Method

Suppose that the data consist of n binomial observations. For the jth observation, let r_j/n_j be the observed proportion and let $\mathbf{x}_j$ be the associated vector of explanatory variables. Suppose that the response probability for the jth observation is a random variable P_j with mean and variance

$$E(P_j) = \pi_j \quad \text{and} \quad V(P_j) = \phi\pi_j(1-\pi_j)$$

where p_j is the probability of the event, and ϕ is a nonnegative but otherwise unknown scale parameter. Then the mean and variance of r_j are

$$E(r_j) = n_j\pi_j \quad \text{and} \quad V(r_j) = n_i j\pi_j(1-\pi_j)[1+(n_j-1)\phi]$$

Williams (1982) estimates the unknown parameter ϕ by equating the value of Pearson's chi-square statistic for the full model to its approximate expected value. Suppose w^*_j is the weight associated with the jth observation. The Pearson chi-square statistic is given by

$$\chi^2 = \sum_{j=1}^{n} \frac{w^*_j(r_j - n_j\widehat{\pi}_j)^2}{n_j\widehat{\pi}_j(1-\widehat{\pi}_j)}$$

Let $g'(\cdot)$ be the first derivative of the link function $g(\cdot)$. The approximate expected value of χ^2 is

$$E_{\chi^2} = \sum_{j=1}^{n} w^*_j(1 - w^*_j v_j d_j)[1+\phi(n_j-1)]$$

where $v_j = n_j/(\pi_j(1-\pi_j)[g'(\pi_j)]^2)$ and d_j is the variance of the linear predictor $\widehat{\alpha}_j + \mathbf{x}'_j\widehat{\boldsymbol{\beta}}$. The scale parameter ϕ is estimated by the following iterative procedure.

At the start, let $w_j^* = 1$ and let π_j be approximated by r_j/n_j, $j = 1, 2, \ldots, n$. If you apply these weights and approximated probabilities to χ^2 and E_{χ^2} and then equate them, an initial estimate of ϕ is

$$\hat{\phi}_0 = \frac{\chi^2 - (n-p)}{\sum_j (n_j - 1)(1 - v_j d_j)}$$

where p is the total number of parameters. The initial estimates of the weights become $\hat{w}_{j0}^* = [1 + (n_j - 1)\hat{\phi}_0]^{-1}$. After a weighted fit of the model, the $\widehat{\alpha}_j$ and $\widehat{\boldsymbol{\beta}}$ are recalculated, and so is χ^2. Then a revised estimate of ϕ is given by

$$\hat{\phi}_1 = \frac{\chi^2 - \sum_j w_j^*(1 - w_j^* v_j d_j)}{w_j^*(n_j - 1)(1 - w_j^* v_j d_j)}$$

The iterative procedure is repeated until χ^2 is very close to its degrees of freedom.

Once ϕ has been estimated by $\hat{\phi}$ under the full model, weights of $(1 + (n_j - 1)\hat{\phi})^{-1}$ can be used to fit models that have fewer terms than the full model. See Example 79.10 for an illustration.

NOTE: If the WEIGHT statement is specified with the NORMALIZE option, then the initial w_j^* values are set to the normalized weights, and the weights resulting from Williams' method will not add up to the actual sample size. However, the estimated covariance matrix of the parameter estimates remains invariant to the scale of the WEIGHT variable.

Receiver Operating Characteristic Curves

Receiver operating characteristic (ROC) curves are used to evaluate and compare the performance of diagnostic tests; they can also be used to evaluate model fit. An ROC curve is just a plot of the proportion of true positives (events predicted to be events) versus the proportion of false positives (nonevents predicted to be events).

In a sample of n individuals, suppose that n_1 individuals are observed to have a certain condition or event, indicated by $Y = 1$; denote this group as $\mathcal{C}_1$. Let the remaining $n_2 = n - n_1$ individuals who do not have the condition be indicated by $Y = 2$, and denote this group as $\mathcal{C}_2$. Risk factors are identified for the sample, and a logistic regression model is fitted to the data. For the jth individual, an estimated probability $\widehat{\pi}_j$ of the event of interest is calculated. By default, the $\widehat{\pi}_j$ are computed as shown in the section "Linear Predictor, Predicted Probability, and Confidence Limits" on page 6046; you can use the cross validated probabilities (discussed in the section "Classification Table" on page 6047) by specifying the CROSSVALIDATE suboption of the ROCOPTIONS option.

Suppose that the n individuals undergo a test for predicting the event and that the test is based on the estimated probability of the event. Higher values of this estimated probability are assumed to be associated with the event. Define this test as a decision rule, D, which classifies an observation as an event when $\hat{\pi} \geq z$; $D = 1$ indicates that the observation is classified as an event, and $D = 2$ indicates that the observation is classified as a nonevent. You can construct an ROC curve by varying the cutpoint z that determines which estimated event probabilities are considered to predict the event. For a given cutpoint, the resulting 2×2 classification table is shown in Table 79.15, where n_{ij} is the total number of observations that are observed to have $Y = i$ and are classified into $D = j$.

Table 79.15 Classification Matrix for Cutpoint z

	$\hat{\pi} \geq z$ $(D=1)$	$\hat{\pi} < z$ $(D=2)$	**Total**
$\mathcal{C}_1(Y=1)$ (event)	n_{11}	n_{12}	n_1
$\mathcal{C}_2(Y=2)$ (nonevent)	n_{21}	n_{22}	n_2

For each cutpoint z, you can output the measures in Table 79.16 to a data set by specifying the OUTROC= option in the MODEL statement or the OUTROC= option in the SCORE statement.

Table 79.16 Statistics from the Classification Matrix with Cutpoint z

Statistic	**Equation**	**Description**
PROB	z	Cutpoint
POS	n_{11}	Number of correctly predicted event responses
NEG	n_{22}	Number of correctly predicted nonevent responses
FALPOS	n_{12}	Number of falsely predicted event responses
FALNEG	n_{21}	Number of falsely predicted nonevent responses
SENSIT	$\frac{n_{11}}{n_1}$	Sensitivity of the test
1MSPEC	$\frac{n_{21}}{n_2}$	One minus the specificity of the test

The ROC curve is a plot of sensitivity (_SENSIT_) against 1–specificity (_1MSPEC_). The plot can be produced by using the PLOTS option or by using the GPLOT or SGPLOT procedure with the OUTROC= data set. See Example 79.7 for an illustration. The area under the ROC curve, as determined by the trapezoidal rule, is estimated by the concordance index, c, in the "Association of Predicted Probabilities and Observed Responses" table.

Comparing ROC Curves

ROC curves can be created from each model fit in a selection routine, from the specified model in the MODEL statement, from specified models in ROC statements, or from input variables which act as $\widehat{\pi}$ in the preceding discussion. Association statistics are computed for these models, and the models are compared when the ROCCONTRAST statement is specified. The ROC comparisons are performed by using a contrast matrix to take differences of the areas under the empirical ROC curves (DeLong, DeLong, and Clarke-Pearson 1988). For example, if you have three curves and the second curve is the reference, the contrast used for the overall test is

$$\mathbf{L}_1 = \begin{pmatrix} \boldsymbol{l}_1' \\ \boldsymbol{l}_2' \end{pmatrix} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & -1 & 1 \end{bmatrix}$$

and you can optionally estimate and test each row of this contrast, in order to test the difference between the reference curve and each of the other curves. If you do not want to use a reference curve, the global test optionally uses the following contrast:

$$\mathbf{L}_2 = \begin{pmatrix} \boldsymbol{l}_1' \\ \boldsymbol{l}_2' \end{pmatrix} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix}$$

You can also specify your own contrast matrix. Instead of estimating the rows of these contrasts, you can request that the difference between every pair of ROC curves be estimated and tested. Demler, Pencina, and

D'Agostino (2012) caution that testing the difference in the AUC between two nested models is not a valid approach if the added predictor is not significantly associated with the response; in any case, if you use this approach, you are more likely to fail to reject the null.

By default for the reference contrast, the specified or selected model is used as the reference unless the NOFIT option is specified in the MODEL statement, in which case the first ROC model is the reference.

In order to label the contrasts, a name is attached to every model. The name for the specified or selected model is the MODEL statement label, or "Model" if the MODEL label is not present. The ROC statement models are named with their labels, or as "ROC*i*" for the *i*th ROC statement if a label is not specified. The contrast $\mathbf{L}_1$ is labeled as "Reference = ModelName", where ModelName is the reference model name, while $\mathbf{L}_2$ is labeled "Adjacent Pairwise Differences". The estimated rows of the contrast matrix are labeled "ModelName1 – ModelName2". In particular, for the rows of $\mathbf{L}_1$, ModelName2 is the reference model name. If you specify your own contrast matrix, then the contrast is labeled "Specified" and the *i*th contrast row estimates are labeled "Row*i*".

If ODS Graphics is enabled, then all ROC curves are displayed individually and are also overlaid in a final display. If a selection method is specified, then the curves produced in each step of the model selection process are overlaid onto a single plot and are labeled "Step*i*", and the selected model is displayed on a separate plot and on a plot with curves from specified ROC statements. See Example 79.8 for an example.

ROC Computations

The trapezoidal area under an empirical ROC curve is equal to the Mann-Whitney two-sample rank measure of association statistic (a generalized U-statistic) applied to two samples, $\{X_i\}, i = 1, \ldots, n_1$, in $\mathcal{C}_1$ and $\{Y_i\}, i = 1, \ldots, n_2$, in $\mathcal{C}_2$. PROC LOGISTIC uses the predicted probabilities in place of $\mathbf{X}$ and $\mathbf{Y}$; however, in general any criterion could be used. Denote the frequency of observation i in $\mathcal{C}_k$ as f_{ki}, and denote the total frequency in $\mathcal{C}_k$ as F_k. The WEIGHTED option replaces f_{ki} with $f_{ki} w_{ki}$, where w_{ki} is the weight of observation i in group $\mathcal{C}_k$. The trapezoidal area under the curve is computed as

$$\begin{aligned} \hat{c} &= \frac{1}{F_1 F_2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \psi(X_i, Y_j) f_{1i} f_{2j} \\ \psi(X, Y) &= \begin{cases} 1 & Y < X \\ \frac{1}{2} & Y = X \\ 0 & Y > X \end{cases} \end{aligned}$$

so that $E(\hat{c}) = \Pr(Y < X) + \frac{1}{2}\Pr(Y = X)$. Note that the concordance index, c, in the "Association of Predicted Probabilities and Observed Responses" table does not use weights unless both the WEIGHTED and BINWIDTH=0 options are in effect.

To compare K empirical ROC curves, first compute the trapezoidal areas. Asymptotic normality of the estimated area follows from U-statistic theory, and a covariance matrix $\mathbf{S}$ can be computed; For more information, see DeLong, DeLong, and Clarke-Pearson (1988). A Wald confidence interval for the rth area, $1 \le r \le K$, can be constructed as

$$\hat{c}_r \pm z_{1-\frac{\alpha}{2}} s_{r,r}$$

where $s_{r,r}$ is the rth diagonal of $\mathbf{S}$.

For a contrast of ROC curve areas, $\mathbf{Lc}$, the statistic

$$(\hat{\mathbf{c}} - \mathbf{c})'\mathbf{L}'\left[\mathbf{LSL}'\right]^{-1}\mathbf{L}(\hat{\mathbf{c}} - \mathbf{c})$$

has a chi-square distribution with df=rank($\mathbf{LSL}'$). For a row of the contrast, $\boldsymbol{l}'\mathbf{c}$,

$$\frac{\boldsymbol{l}'\hat{\mathbf{c}} - \boldsymbol{l}'\mathbf{c}}{[\boldsymbol{l}'\mathbf{S}\boldsymbol{l}]^{1/2}}$$

has a standard normal distribution. The corresponding confidence interval is

$$\boldsymbol{l}'\hat{\mathbf{c}} \pm z_{1-\frac{\alpha}{2}}\left[\boldsymbol{l}'\mathbf{S}\boldsymbol{l}\right]^{1/2}$$

Precision-Recall Curves

Precision-recall (PR) curves are often used to evaluate predictive performance of models for skewed data where, for example, the number of nonevents is far greater than the number of events and for which ROC curves might be misleading (Saito and Rehmsmeier 2015).

PR curves are produced by mapping from the ROC curve, following Davis and Goadrich (2006). To perform this mapping, you use the sensitivity (TPF), the specificity (1 – FPF), and the prevalence of events (P) in order to compute the precision (PPV), given by

$$\mathrm{PPV} = \frac{\mathrm{P} \cdot \mathrm{TPF}}{\mathrm{P} \cdot (1 - \mathrm{FPF}) + (1 - \mathrm{P})(1 - \mathrm{TPF})}$$

See the section "Receiver Operating Characteristic Curves" on page 6055 for definitions. You cannot linearly interpolate between values on the PR curve because of this nonlinear PPV computation; instead, you linearly interpolate between points on the ROC curve and map these new points back to the PR curve. The STEPLEN= value specifies the length of a step along the ROC curve, which is used to determine the extra points at which the PR curve is computed. That is, if two neighboring points on the ROC curve are farther apart than *value*, then intermediate points at multiples of *value* from the first point are also mapped. You can specify a *value* between 0 and 1; the default is 0.01.

The PR curve often needs to be extrapolated back to TPF = 0; this is performed by using either the first nontrivial PPV computed from the observed ROC points or the prevalence, whichever is larger.

The trapezoidal area under the PR curve is also computed and displayed. This area is often used to compare models, even though it is not statistically relevant in the same way that the AUC for the ROC curve is, as discussed in the section "ROC Computations" on page 6057.

Testing Linear Hypotheses about the Regression Coefficients

Linear hypotheses for $\boldsymbol{\beta}$ are expressed in matrix form as

$$H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{c}$$

where $\mathbf{L}$ is a matrix of coefficients for the linear hypotheses, and $\mathbf{c}$ is a vector of constants. The vector of regression coefficients $\boldsymbol{\beta}$ includes slope parameters as well as intercept parameters. The Wald chi-square statistic for testing H_0 is computed as

$$\chi^2_{\mathrm{W}} = (\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{c})'[\mathbf{L}\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})\mathbf{L}']^{-1}(\mathbf{L}\widehat{\boldsymbol{\beta}} - \mathbf{c})$$

where $\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})$ is the estimated covariance matrix. Under H_0, χ^2_{W} has an asymptotic chi-square distribution with r degrees of freedom, where r is the rank of $\mathbf{L}$.

Joint Tests and Type 3 Tests

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, performing the Type 3 test of a main effect corresponds to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions." Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are zero. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), performing the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses reference parameterization (as specified by the PARAM=REF option in the CLASS statement), performing the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects. For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).

If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimable function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.

Regression Diagnostics

For binary response data, regression diagnostics developed by Pregibon (1981) can be requested by specifying the INFLUENCE option. For diagnostics available with conditional logistic regression, see the section "Regression Diagnostic Details" on page 6069. These diagnostics can also be obtained from the OUTPUT statement.

This section uses the following notation:

r_j, n_j	r_j is the number of event responses out of n_j trials for the jth observation. If events/trials syntax is used, r_j is the value of *events* and n_j is the value of *trials*. For single-trial syntax, $n_j = 1$, and $r_j = 1$ if the ordered response is 1, and $r_j = 0$ if the ordered response is 2.
w_j	is the weight of the jth observation.
π_j	is the probability of an event response for the jth observation given by $\pi_j = F(\alpha + \boldsymbol{\beta}'\mathbf{x}_j)$, where $F(\cdot)$ is the inverse link function defined on page 6028.
$\widehat{\boldsymbol{\beta}}$	is the maximum likelihood estimate (MLE) of $(\alpha, \beta_1, \ldots, \beta_s)'$.
$\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})$	is the estimated covariance matrix of $\widehat{\boldsymbol{\beta}}$.
$\hat{p}_j, \hat{q}_j$	$\hat{p}_j$ is the estimate of π_j evaluated at $\widehat{\boldsymbol{\beta}}$, and $\hat{q}_j = 1 - \hat{p}_j$.

Pregibon (1981) suggests using the index plots of several diagnostic statistics to identify influential observations and to quantify the effects on various aspects of the maximum likelihood fit. In an index plot, the diagnostic statistic is plotted against the observation number. In general, the distributions of these diagnostic statistics are not known, so cutoff values cannot be given for determining when the values are large. However, the IPLOTS and INFLUENCE options in the MODEL statement and the PLOTS option in the PROC LOGISTIC statement provide displays of the diagnostic values, allowing visual inspection and comparison of the values across observations. In these plots, if the model is correctly specified and fits all observations well, then no extreme points should appear.

The next five sections give formulas for these diagnostic statistics.

Hat Matrix Diagonal (Leverage)

The diagonal elements of the hat matrix are useful in detecting extreme points in the design space where they tend to have larger values. The jth diagonal element is

$$h_j = \begin{cases} \widetilde{w}_j (1, \mathbf{x}_j')\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})(1, \mathbf{x}_j')' & \text{Fisher scoring} \\ \widehat{w}_j (1, \mathbf{x}_j')\widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}})(1, \mathbf{x}_j')' & \text{Newton-Raphson} \end{cases}$$

where

$$\begin{aligned} \widetilde{w}_j &= \frac{w_j n_j}{\hat{p}_j \hat{q}_j [g'(\hat{p}_j)]^2} \\ \widehat{w}_j &= \widetilde{w}_j + \frac{w_j (r_j - n_j \hat{p}_j)[\hat{p}_j \hat{q}_j g''(\hat{p}_j) + (\hat{q}_j - \hat{p}_j) g'(\hat{p}_j)]}{(\hat{p}_j \hat{q}_j)^2 [g'(\hat{p}_j)]^3} \end{aligned}$$

and $g'(\cdot)$ and $g''(\cdot)$ are the first and second derivatives of the link function $g(\cdot)$, respectively.

For a binary response logit model, the hat matrix diagonal elements are

$$h_j = w_j n_j \hat{p}_j \hat{q}_j (1, \mathbf{x}_j') \widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}}) \begin{pmatrix} 1 \\ \mathbf{x}_j \end{pmatrix}$$

If the estimated probability is extreme (less than 0.1 or greater than 0.9, approximately), then the hat diagonal might be greatly reduced in value. Consequently, when an observation has a very large or very small estimated probability, its hat diagonal value is not a good indicator of the observation's distance from the design space (Hosmer and Lemeshow 2000, p. 171).

Residuals

Residuals are useful in identifying observations that are not explained well by the model. Pearson residuals are components of the Pearson chi-square statistic and deviance residuals are components of the deviance. The Pearson residual for the *j*th observation is

$$\chi_j = \frac{\sqrt{w_j}(r_j - n_j \hat{p}_j)}{\sqrt{n_j \hat{p}_j \hat{q}_j}}$$

The Pearson chi-square statistic is the sum of squares of the Pearson residuals.

The deviance residual for the *j*th observation is

$$d_j = \begin{cases} -\sqrt{-2w_j n_j \log(\hat{q}_j)} & \text{if } r_j = 0 \\ \pm\sqrt{2w_j[r_j \log(\frac{r_j}{n_j \hat{p}_j}) + (n_j - r_j)\log(\frac{n_j - r_j}{n_j \hat{q}_j})]} & \text{if } 0 < r_j < n_j \\ \sqrt{-2w_j n_j \log(\hat{p}_j)} & \text{if } r_j = n_j \end{cases}$$

where the plus (minus) in $\pm$ is used if r_j/n_j is greater (less) than $\hat{p}_j$. The deviance is the sum of squares of the deviance residuals.

The STDRES option in the INFLUENCE and PLOTS=INFLUENCE options computes three more residuals (Collett 2003). The Pearson and deviance residuals are standardized to have approximately unit variance:

$$\begin{aligned} e_{p_j} &= \frac{\chi_j}{\sqrt{1 - h_j}} \\ e_{d_j} &= \frac{d_j}{\sqrt{1 - h_j}} \end{aligned}$$

The likelihood residuals, which estimate components of a likelihood ratio test of deleting an individual observation, are a weighted combination of the standardized Pearson and deviance residuals

$$e_{l_j} = \text{sign}(r_j - n_j \hat{p}_j)\sqrt{h_j e_{p_j}^2 + (1 - h_j) e_{d_j}^2}$$

DFBETAS

For each parameter estimate, the procedure calculates a DFBETAS diagnostic for each observation. The DFBETAS diagnostic for an observation is the standardized difference in the parameter estimate due to deleting the observation, and it can be used to assess the effect of an individual observation on each estimated parameter of the fitted model. Instead of reestimating the parameter every time an observation is deleted, PROC LOGISTIC uses the one-step estimate. See the section "Predicted Probability of an Event for Classification" on page 6049. For the *j*th observation, the DFBETAS are given by

$$\text{DFBETAS}i_j = \mathbf{\Delta}_i \widehat{\boldsymbol{\beta}}_j^1 / \hat{\sigma}_i$$

where $i = 0, 1, \ldots, s$, $\hat{\sigma}_i$ is the standard error of the *i*th component of $\widehat{\boldsymbol{\beta}}$, and $\mathbf{\Delta}_i \widehat{\boldsymbol{\beta}}_j^1$ is the *i*th component of the one-step difference

$$\mathbf{\Delta}\widehat{\boldsymbol{\beta}}_j^1 = \frac{w_j(r_j - n_j \hat{p}_j)}{1 - h_j} \widehat{\mathbf{V}}(\widehat{\boldsymbol{\beta}}) \begin{pmatrix} 1 \\ \mathbf{x}_j \end{pmatrix}$$

$\mathbf{\Delta}\widehat{\boldsymbol{\beta}}_j^1$ is the approximate change $(\widehat{\boldsymbol{\beta}} - \widehat{\boldsymbol{\beta}}_j^1)$ in the vector of parameter estimates due to the omission of the *j*th observation. The DFBETAS are useful in detecting observations that are causing instability in the selected coefficients.

C and CBAR

C and CBAR are confidence interval displacement diagnostics that provide scalar measures of the influence of individual observations on $\widehat{\boldsymbol{\beta}}$. These diagnostics are based on the same idea as the Cook distance in linear regression theory (Cook and Weisberg 1982), but use the one-step estimate. C and CBAR for the *j*th observation are computed as

$$C_j = \chi_j^2 h_j / (1 - h_j)^2$$

and

$$\overline{C}_j = \chi_j^2 h_j / (1 - h_j)$$

respectively.

Typically, to use these statistics, you plot them against an index and look for outliers.

DIFDEV and DIFCHISQ

DIFDEV and DIFCHISQ are diagnostics for detecting ill-fitted observations; in other words, observations that contribute heavily to the disagreement between the data and the predicted values of the fitted model. DIFDEV is the change in the deviance due to deleting an individual observation while DIFCHISQ is the change in the Pearson chi-square statistic for the same deletion. By using the one-step estimate, DIFDEV and DIFCHISQ for the *j*th observation are computed as

$$\text{DIFDEV} = d_j^2 + \overline{C}_j$$

and

$$\text{DIFCHISQ} = \overline{C}_j / h_j$$

Scoring Data Sets

Scoring a data set, which is especially important for predictive modeling, means applying a previously fitted model to a new data set in order to compute the conditional, or *posterior*, probabilities of each response category given the values of the explanatory variables in each observation.

The SCORE statement enables you to score new data sets and output the scored values and, optionally, the corresponding confidence limits into a SAS data set. If the response variable is included in the new data set, then you can request fit statistics for the data, which is especially useful for test or validation data. If the response is binary, you can also create a SAS data set containing the *receiver operating characteristic* (ROC) curve. You can specify multiple SCORE statements in the same invocation of PROC LOGISTIC.

By default, the posterior probabilities are based on implicit prior probabilities that are proportional to the frequencies of the response categories in the *training data* (the data used to fit the model). Explicit prior probabilities should be specified with the PRIOR= or PRIOREVENT= option when the sample proportions of the response categories in the training data differ substantially from the operational data to be scored. For example, to detect a rare category, it is common practice to use a training set in which the rare categories are overrepresented; without prior probabilities that reflect the true incidence rate, the predicted posterior probabilities for the rare category will be too high. By specifying the correct priors, the posterior probabilities are adjusted appropriately.

The model fit to the DATA= data set in the PROC LOGISTIC statement is the default model used for the scoring. Alternatively, you can save a model fit in one run of PROC LOGISTIC and use it to score new data in a subsequent run. The OUTMODEL= option in the PROC LOGISTIC statement saves the model information in a SAS data set. Specifying this data set in the INMODEL= option of a new PROC LOGISTIC run will score the DATA= data set in the SCORE statement without refitting the model.

The STORE statement can also be used to save your model. The PLM procedure can use this model to score new data sets; see Chapter 94, "The PLM Procedure," for more information. You cannot specify priors in PROC PLM.

Fit Statistics for Scored Data Sets

Specifying the FITSTAT option displays the following fit statistics when the data set being scored includes the response variable:

Statistic	Description
Total frequency	$F = \sum_i f_i n_i$
Total weight	$W = \sum_i f_i w_i n_i$
Log likelihood	$\log L = \sum_i f_i w_i \log(\widehat{\pi}_i)$
Full log likelihood	$\log L_f = \text{constant} + \log L$
Misclassification (error) rate	$\frac{\sum_i 1\{\text{F_}Y_i \neq \text{I_}Y_i\} f_i n_i}{F}$
AIC	$-2\log L_f + 2p$
AICC	$-2\log L_f + \frac{2pn}{n-p-1}$
BIC	$-2\log L_f + p\log(n)$
SC	$-2\log L_f + p\log(F)$
R-square	$R^2 = 1 - \left(\frac{L_0}{L}\right)^{2/F}$
Maximum-rescaled R-square	$\frac{R^2}{1 - L_0^{2/F}}$
AUC	Area under the ROC curve
Brier score (polytomous response)	$\frac{1}{W}\sum_i f_i w_i \sum_j (y_{ij} - \widehat{\pi}_{ij})^2$
Brier score (binary response)	$\frac{1}{W}\sum_i f_i w_i (r_i(1-\widehat{\pi}_i)^2 + (n_i - r_i)\widehat{\pi}_i^2)$
Brier reliability (events/trials syntax)	$\frac{1}{W}\sum_i f_i w_i (r_i/n_i - \widehat{\pi}_i)^2$

In the preceding table, f_i is the frequency of the ith observation in the data set being scored, w_i is the weight of the observation, and $n = \sum_i f_i$. The number of trials when events/trials syntax is specified is n_i, and with single-trial syntax $n_i = 1$. The values F_Y_i and I_Y_i are described in the section "OUT= Output Data Set in a SCORE Statement" on page 6078. The indicator function $1\{A\}$ is 1 if A is true and 0 otherwise. The likelihood of the model is L, and L_0 denotes the likelihood of the intercept-only model. For polytomous response models, y_i is the observed polytomous response level, $\widehat{\pi}_{ij}$ is the predicted probability of the jth response level for observation i, and $y_{ij} = 1\{y_i = j\}$. For binary response models, $\widehat{\pi}_i$ is the predicted probability of the observation, r_i is the number of events when you specify events/trials syntax, and $r_i = y_i$ when you specify single-trial syntax.

The log likelihood, Akaike's information criterion (AIC), and Schwarz criterion (SC) are described in the section "Model Fitting Information" on page 6035. The full log likelihood is displayed for models specified with events/trials syntax, and the constant term is described in the section "Model Fitting Information" on page 6035. The AICC is a small-sample bias-corrected version of the AIC (Hurvich and Tsai 1993; Burnham and Anderson 1998). The Bayesian information criterion (BIC) is the same as the SC except when events/trials syntax is specified. The area under the ROC curve for binary response models is defined in the section "ROC Computations" on page 6057. The R-square and maximum-rescaled R-square statistics, defined in "Generalized Coefficient of Determination" on page 6036, are not computed when you specify both an OFFSET= variable and the INMODEL= data set. The Brier score (Brier 1950) is the weighted squared difference between the predicted probabilities and their observed response levels. For events/trials syntax, the Brier reliability is the weighted squared difference between the predicted probabilities and the observed proportions (Murphy 1973).

Posterior Probabilities and Confidence Limits

Let F be the inverse link function. That is,

$$F(t) = \begin{cases} \frac{1}{1+\exp(-t)} & \text{logistic} \\ \Phi(t) & \text{normal} \\ 1-\exp(-\exp(t)) & \text{complementary log-log} \end{cases}$$

The first derivative of F is given by

$$F'(t) = \begin{cases} \frac{\exp(-t)}{(1+\exp(-t))^2} & \text{logistic} \\ \phi(t) & \text{normal} \\ \exp(t)\exp(-\exp(t)) & \text{complementary log-log} \end{cases}$$

Suppose there are $k+1$ response categories. Let Y be the response variable with levels $1,\ldots,k+1$. Let $\mathbf{x} = (x_0, x_1, \ldots, x_s)'$ be a $(s+1)$-vector of covariates, with $x_0 \equiv 1$. Let $\boldsymbol{\beta}$ be the vector of intercept and slope regression parameters.

Posterior probabilities are given by

$$p(Y=i|\mathbf{x}) = \frac{p_o(Y=i|\mathbf{x})\frac{p_n(Y=i)}{p_o(Y=i)}}{\sum_j p_o(Y=j|\mathbf{x})\frac{p_n(Y=j)}{p_o(Y=j)}} \quad i=1,\ldots,k+1$$

where the old posterior probabilities $(p_o(Y=i|\mathbf{x}), i=1,\ldots,k+1)$ are the conditional probabilities of the response categories given $\mathbf{x}$, the old priors $(p_o(Y=i), i=1,\ldots,k+1)$ are the sample proportions of response categories of the training data, and the new priors $(p_n(Y=i), i=1,\ldots,k+1)$ are specified in the PRIOR= or PRIOREVENT= option. To simplify notation, absorb the old priors into the new priors; that is

$$p(Y=i) = \frac{p_n(Y=i)}{p_o(Y=i)} \quad i=1,\ldots,k+1$$

Note if the PRIOR= and PRIOREVENT= options are not specified, then $p(Y=i)=1$.

The posterior probabilities are functions of $\boldsymbol{\beta}$ and their estimates are obtained by substituting $\boldsymbol{\beta}$ by its MLE $\widehat{\boldsymbol{\beta}}$. The variances of the estimated posterior probabilities are given by the *delta method* as follows:

$$\mathrm{Var}(\widehat{p}(Y=i|\mathbf{x})) = \left[\frac{\partial p(Y=i|\mathbf{x})}{\partial \boldsymbol{\beta}}\right]' \mathrm{Var}(\widehat{\boldsymbol{\beta}}) \left[\frac{\partial p(Y=i|\mathbf{x})}{\partial \boldsymbol{\beta}}\right]$$

where

$$\frac{\partial p(Y=i|\mathbf{x})}{\partial \boldsymbol{\beta}} = \frac{\frac{\partial p_o(Y=i|\mathbf{x})}{\partial \boldsymbol{\beta}} p(Y=i)}{\sum_j p_o(Y=j|\mathbf{x})p(Y=j)} - \frac{p_o(Y=i|\mathbf{x})p(Y=i)\sum_j \frac{\partial p_o(Y=j|\mathbf{x})}{\partial \boldsymbol{\beta}} p(Y=j)}{[\sum_j p_o(Y=j|\mathbf{x})p(Y=j)]^2}$$

and the old posterior probabilities $p_o(Y=i|\mathbf{x})$ are described in the following sections.

A $100(1-\alpha)\%$ confidence interval for $p(Y=i|\mathbf{x})$ is

$$\widehat{p}(Y=i|\mathbf{x}) \pm z_{1-\alpha/2}\sqrt{\widehat{\mathrm{Var}}(\widehat{p}(Y=i|\mathbf{x}))}$$

where z_τ is the upper 100τth percentile of the standard normal distribution.

Binary and Cumulative Response Models

Let $\alpha_1, \ldots, \alpha_k$ be the intercept parameters and let $\boldsymbol{\beta}_s$ be the vector of slope parameters. Denote $\boldsymbol{\beta} = (\alpha_1, \ldots, \alpha_k, \boldsymbol{\beta}_s')'$. Let

$$\eta_i = \eta_i(\boldsymbol{\beta}) = \alpha_i + \mathbf{x}'\boldsymbol{\beta}_s, i = 1, \ldots, k$$

Estimates of $\eta_1, \ldots, \eta_k$ are obtained by substituting the maximum likelihood estimate $\widehat{\boldsymbol{\beta}}$ for $\boldsymbol{\beta}$.

The predicted probabilities of the responses are

$$\widehat{p_o}(Y = i|\mathbf{x}) = \widehat{\Pr}(Y = i) = \begin{cases} F(\hat{\eta}_1) & i = 1 \\ F(\hat{\eta}_i) - F(\hat{\eta}_{i-1}) & i = 2, \ldots, k \\ 1 - F(\hat{\eta}_k) & i = k + 1 \end{cases}$$

For $i = 1, \ldots, k$, let $\boldsymbol{\delta}_i(\mathbf{x})$ be a $(k + 1)$ column vector with ith entry equal to 1, $k + 1$ entry equal to $\mathbf{x}$, and all other entries 0. The derivative of $p_o(Y = i|\mathbf{x})$ with respect to $\boldsymbol{\beta}$ are

$$\frac{\partial p_o(Y = i|\mathbf{x})}{\partial \boldsymbol{\beta}} = \begin{cases} F'(\alpha_1 + \mathbf{x}'\boldsymbol{\beta}_s)\boldsymbol{\delta}_1(\mathbf{x}) & i = 1 \\ F'(\alpha_i + \mathbf{x}'\boldsymbol{\beta}_s)\boldsymbol{\delta}_i(\mathbf{x}) - F'(\alpha_{i-1} + \mathbf{x}'\boldsymbol{\beta}_s)\boldsymbol{\delta}_{i-1}(\mathbf{x}) & i = 2, \ldots, k \\ -F'(\alpha_k + \mathbf{x}'\boldsymbol{\beta}_s)\boldsymbol{\delta}_k(\mathbf{x}) & i = k + 1 \end{cases}$$

The cumulative posterior probabilities are

$$p(Y \le i|\mathbf{x}) = \frac{\sum_{j=1}^{i} p_o(Y = j|\mathbf{x})p(Y = j)}{\sum_{j=1}^{k+1} p_o(Y = j|\mathbf{x})p(Y = j)} = \sum_{j=1}^{i} p(Y = j|\mathbf{x}) \quad i = 1, \ldots, k + 1$$

Their derivatives are

$$\frac{\partial p(Y \le i|\mathbf{x})}{\partial \boldsymbol{\beta}} = \sum_{j=1}^{i} \frac{\partial p(Y = j|\mathbf{x})}{\partial \boldsymbol{\beta}} \quad i = 1, \ldots, k + 1$$

In the delta-method equation for the variance, replace $p(Y = \cdot|\mathbf{x})$ with $p(Y \le \cdot|\mathbf{x})$.

Finally, for the cumulative response model, use

$$\begin{aligned} \widehat{p_o}(Y \le i|\mathbf{x}) &= F(\hat{\eta}_i) \quad i = 1, \ldots, k \\ \widehat{p_o}(Y \le k + 1|\mathbf{x}) &= 1 \\ \frac{\partial p_o(Y \le i|\mathbf{x})}{\partial \boldsymbol{\beta}} &= F'(\alpha_i + \mathbf{x}'\boldsymbol{\beta}_s)\delta_i(\mathbf{x}) \quad i = 1, \ldots, k \\ \frac{\partial p_o(Y \le k + 1|\mathbf{x})}{\partial \boldsymbol{\beta}} &= 0 \end{aligned}$$

Generalized Logit Model

Consider the last response level (Y=k+1) as the reference. Let $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_k$ be the (intercept and slope) parameter vectors for the first k logits, respectively. Denote $\boldsymbol{\beta} = (\boldsymbol{\beta}_1', \ldots, \boldsymbol{\beta}_k')'$. Let $\eta = (\eta_1, \ldots, \eta_k)'$ with

$$\eta_i = \eta_i(\boldsymbol{\beta}) = \mathbf{x}'\boldsymbol{\beta}_i \quad i = 1, \ldots, k$$

Estimates of $\eta_1, \ldots, \eta_k$ are obtained by substituting the maximum likelihood estimate $\widehat{\boldsymbol{\beta}}$ for $\boldsymbol{\beta}$.

The predicted probabilities are

$$\begin{aligned}
\widehat{p_o}(Y = k+1|\mathbf{x}) \equiv \Pr(Y = k+1|\mathbf{x}) &= \frac{1}{1 + \sum_{l=1}^{k} \exp(\hat{\eta}_l)} \\
\widehat{p_o}(Y = i|\mathbf{x}) \equiv \Pr(Y = i|\mathbf{x}) &= \widehat{p_o}(Y = k+1|\mathbf{x}) \exp(\eta_i), i = 1, \ldots, k
\end{aligned}$$

The derivative of $p_o(Y = i|\mathbf{x})$ with respect to $\boldsymbol{\beta}$ are

$$\begin{aligned}
\frac{\partial p_o(Y = i|\mathbf{x})}{\partial \boldsymbol{\beta}} &= \frac{\partial \eta}{\partial \boldsymbol{\beta}} \frac{\partial p_o(Y = i|\mathbf{x})}{\partial \eta} \\
&= (I_k \otimes \mathbf{x}) \left(\frac{\partial p_o(Y = i|\mathbf{x})}{\partial \eta_1}, \cdots, \frac{\partial p_o(Y = i|\mathbf{x})}{\partial \eta_k} \right)'
\end{aligned}$$

where

$$\frac{\partial p_o(Y = i|\mathbf{x})}{\partial \eta_j} = \begin{cases} p_o(Y = i|\mathbf{x})(1 - p_o(Y = i|\mathbf{x})) & j = i \\ -p_o(Y = i|\mathbf{x}) p_o(Y = j|\mathbf{x}) & \text{otherwise} \end{cases}$$

Special Case of Binary Response Model with No Priors

Let $\boldsymbol{\beta}$ be the vector of regression parameters. Let

$$\eta = \eta(\boldsymbol{\beta}) = \mathbf{x}'\boldsymbol{\beta}$$

The variance of $\hat{\eta}$ is given by

$$\mathrm{Var}(\hat{\eta}) = \mathbf{x}'\mathrm{Var}(\widehat{\boldsymbol{\beta}})\mathbf{x}$$

A $100(1-\alpha)$ percent confidence interval for η is

$$\hat{\eta} \pm z_{1-\alpha/2} \sqrt{\widehat{\mathrm{Var}}(\hat{\eta})}$$

Estimates of $p_o(Y = 1|\mathbf{x})$ and confidence intervals for the $p_o(Y = 1|\mathbf{x})$ are obtained by back-transforming $\hat{\eta}$ and the confidence intervals for η, respectively. That is,

$$\widehat{p_o}(Y = 1|\mathbf{x}) = F(\hat{\eta})$$

and the confidence intervals are

$$F\left(\hat{\eta} \pm z_{1-\alpha/2} \sqrt{\widehat{\mathrm{Var}}(\hat{\eta})}\right)$$

Conditional Logistic Regression

The method of maximum likelihood described in the preceding sections relies on large-sample asymptotic normality for the validity of estimates and especially of their standard errors. When you do not have a large sample size compared to the number of parameters, this approach might be inappropriate and might result in biased inferences. This situation typically arises when your data are stratified and you fit intercepts to each stratum so that the number of parameters is of the same order as the sample size. For example, in a 1:1 matched pairs study with n pairs and p covariates, you would estimate $n-1$ intercept parameters and p slope parameters. Taking the stratification into account by "conditioning out" (and not estimating) the stratum-specific intercepts gives consistent and asymptotically normal MLEs for the slope coefficients. See Breslow and Day (1980) and Stokes, Davis, and Koch (2012) for more information. If your nuisance parameters are not just stratum-specific intercepts, you can perform an exact conditional logistic regression.

Computational Details

For each stratum h, $h = 1, \ldots, H$, number the observations as $i = 1, \ldots, n_h$ so that hi indexes the ith observation in stratum h. Denote the p covariates for the hith observation as $\mathbf{x}_{hi}$ and its binary response as y_{hi}, and let $\mathbf{y} = (y_{11}, \ldots, y_{1n_1}, \ldots, y_{H1}, \ldots, y_{Hn_H})'$, $\mathbf{X}_h = (\mathbf{x}_{h1} \ldots \mathbf{x}_{hn_h})'$, and $\mathbf{X} = (\mathbf{X}_1' \ldots \mathbf{X}_H')'$. Let the dummy variables $z_h, h = 1, \ldots, H$, be indicator functions for the strata ($z_h = 1$ if the observation is in stratum h), and denote $\mathbf{z}_{hi} = (z_1, \ldots, z_H)$ for the hith observation, $\mathbf{Z}_h = (\mathbf{z}_{h1} \ldots \mathbf{z}_{hn_h})'$, and $\mathbf{Z} = (\mathbf{Z}_1' \ldots \mathbf{Z}_H')'$. Denote $\mathbf{X}^* = (\mathbf{Z}|\mathbf{X})$ and $\mathbf{x}^*_{hi} = (\mathbf{z}'_{hi}|\mathbf{x}'_{hi})'$. Arrange the observations in each stratum h so that $y_{hi} = 1$ for $i = 1, \ldots, m_h$, and $y_{hi} = 0$ for $i = m_{h+1}, \ldots, n_h$. Suppose all observations have unit frequency.

Consider the binary logistic regression model on page 5939 written as

$$\text{logit}(\boldsymbol{\pi}) = \mathbf{X}^* \boldsymbol{\theta}$$

where the parameter vector $\boldsymbol{\theta} = (\boldsymbol{\alpha}', \boldsymbol{\beta}')'$ consists of $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_H)'$, α_h is the intercept for stratum $h, h = 1, \ldots, H$, and $\boldsymbol{\beta}$ is the parameter vector for the p covariates.

From the section "Determining Observations for Likelihood Contributions" on page 6029, you can write the likelihood contribution of observation $hi, i = 1, \ldots, n_h, h = 1, \ldots, H$, as

$$L_{hi}(\boldsymbol{\theta}) = \frac{e^{y_{hi}\mathbf{x}^{*\prime}_{hi}\boldsymbol{\theta}}}{1 + e^{\mathbf{x}^{*\prime}_{hi}\boldsymbol{\theta}}}$$

where $y_{hi} = 1$ when the response takes Ordered Value 1, and $y_{hi} = 0$ otherwise.

The full likelihood is

$$L(\boldsymbol{\theta}) = \prod_{h=1}^{H} \prod_{i=1}^{n_h} L_{hi}(\boldsymbol{\theta}) = \frac{e^{\mathbf{y}'\mathbf{X}^*\boldsymbol{\theta}}}{\prod_{h=1}^{H} \prod_{i=1}^{n_h} \left(1 + e^{\mathbf{x}^{*\prime}_{hi}\boldsymbol{\theta}}\right)}$$

Unconditional likelihood inference is based on maximizing this likelihood function.

When your nuisance parameters are the stratum-specific intercepts $(\alpha_1, \ldots, \alpha_H)'$, and the slopes $\boldsymbol{\beta}$ are your parameters of interest, "conditioning out" the nuisance parameters produces the conditional likelihood (Lachin 2000)

$$L(\boldsymbol{\beta}) = \prod_{h=1}^{H} L_h(\boldsymbol{\beta}) = \prod_{h=1}^{H} \frac{\prod_{i=1}^{m_h} \exp(\mathbf{x}'_{hi}\boldsymbol{\beta})}{\sum \prod_{j=j_1}^{j_{m_h}} \exp(\mathbf{x}'_{hj}\boldsymbol{\beta})}$$

where the summation is over all $\binom{n_h}{m_h}$ subsets $\{j_1, \ldots, j_{m_h}\}$ of m_h observations chosen from the n_h observations in stratum h. Note that the nuisance parameters have been factored out of this equation.

For conditional asymptotic inference, maximum likelihood estimates $\widehat{\boldsymbol{\beta}}$ of the regression parameters are obtained by maximizing the conditional likelihood, and asymptotic results are applied to the conditional likelihood function and the maximum likelihood estimators. A relatively fast method of computing this conditional likelihood and its derivatives is given by Gail, Lubin, and Rubinstein (1981) and Howard (1972). The optimization techniques can be controlled by specifying the NLOPTIONS statement.

Sometimes the log likelihood converges but the estimates diverge. This condition is flagged by having inordinately large standard errors for some of your parameter estimates, and can be monitored by specifying the ITPRINT option. Unfortunately, broad existence criteria such as those discussed in the section "Existence of Maximum Likelihood Estimates" on page 6032 do not exist for this model. It might be possible to circumvent such a problem by standardizing your independent variables before fitting the model.

Regression Diagnostic Details

Diagnostics are used to indicate observations that might have undue influence on the model fit or that might be outliers. Further investigation should be performed before removing such an observation from the data set.

The derivations in this section use an augmentation method described by Storer and Crowley (1985), which provides an estimate of the "one-step" DFBETAS estimates advocated by Pregibon (1984). The method also provides estimates of conditional stratum-specific predicted values, residuals, and leverage for each observation. The augmentation method can take a lot of time and memory.

Following Storer and Crowley (1985), the log-likelihood contribution can be written as

$$\begin{aligned} l_h &= \log(L_h) = \mathbf{y}_h'\boldsymbol{\gamma}_h - a(\boldsymbol{\gamma}_h) \quad \text{where} \\ a(\boldsymbol{\gamma}_h) &= \log\left[\sum \prod_{j=j_1}^{j_{m_h}} \exp(\boldsymbol{\gamma}_{hj})\right] \end{aligned}$$

and the h subscript on matrices indicates the submatrix for the stratum, $\boldsymbol{\gamma}_h = (\gamma_{h1}, \ldots, \gamma_{hn_h})'$, and $\gamma_{hi} = \mathbf{x}_{hi}'\boldsymbol{\beta}$. Then the gradient and information matrix are

$$\begin{aligned} \mathbf{g}(\boldsymbol{\beta}) &= \left\{\frac{\partial l_h}{\partial \boldsymbol{\beta}}\right\}_{h=1}^{H} = \mathbf{X}'(\mathbf{y} - \boldsymbol{\pi}) \\ \boldsymbol{\Lambda}(\boldsymbol{\beta}) &= \left\{\frac{\partial^2 l_h}{\partial \boldsymbol{\beta}^2}\right\}_{h=1}^{H} = \mathbf{X}'\mathrm{diag}(\mathbf{U}_1, \ldots, \mathbf{U}_H)\mathbf{X} \end{aligned}$$

where

$$
\begin{aligned}
\pi_{hi} &= \frac{\partial a(\boldsymbol{\gamma}_h)}{\partial \gamma_{hi}} = \frac{\sum_{j(i)} \prod_{j=j_1}^{j_{m_h}} \exp(\gamma_{hj})}{\sum \prod_{j=j_1}^{j_{m_h}} \exp(\gamma_{hj})} \\
\boldsymbol{\pi}_h &= (\pi_{h1}, \ldots, \pi_{hn_h}) \\
\mathbf{U}_h &= \frac{\partial^2 a(\boldsymbol{\gamma}_h)}{\partial \boldsymbol{\gamma}_h^2} = \left\{ \frac{\partial^2 \mathbf{a}(\boldsymbol{\gamma}_h)}{\partial \gamma_{hi} \partial \gamma_{hj}} \right\} = \{a_{ij}\} \\
a_{ij} &= \frac{\sum_{k(i,j)} \prod_{k=k_1}^{k_{m_h}} \exp(\gamma_{hk})}{\sum \prod_{k=k_1}^{k_{m_h}} \exp(\gamma_{hk})} - \frac{\partial a(\boldsymbol{\gamma}_h)}{\partial \gamma_{hi}} \frac{\partial a(\boldsymbol{\gamma}_h)}{\partial \gamma_{hj}} = \pi_{hij} - \pi_{hi}\pi_{hj}
\end{aligned}
$$

and where π_{hi} is the conditional stratum-specific probability that subject i in stratum h is a case, the summation on $j(i)$ is over all subsets from $\{1, \ldots, n_h\}$ of size m_h that contain the index i, and the summation on $k(i, j)$ is over all subsets from $\{1, \ldots, n_h\}$ of size m_h that contain the indices i and j.

To produce the true one-step estimate $\boldsymbol{\beta}_{hi}^1$, start at the MLE $\widehat{\boldsymbol{\beta}}$, delete the hith observation, and use this reduced data set to compute the next Newton-Raphson step. Note that if there is only one event or one nonevent in a stratum, deletion of that single observation is equivalent to deletion of the entire stratum. The augmentation method does not take this into account.

The augmented model is

$$
\text{logit}(\Pr(y_{hi} = 1 | \mathbf{x}_{hi})) = \mathbf{x}_{hi}'\boldsymbol{\beta} + \mathbf{z}_{hi}'\gamma
$$

where $\mathbf{z}_{hi} = (0, \ldots, 0, 1, 0, \ldots, 0)'$ has a 1 in the hith coordinate, and use $\boldsymbol{\beta}^0 = (\widehat{\boldsymbol{\beta}}', 0)'$ as the initial estimate for $(\boldsymbol{\beta}', \gamma)'$. The gradient and information matrix before the step are

$$
\begin{aligned}
\mathbf{g}(\boldsymbol{\beta}^0) &= \begin{bmatrix} \mathbf{X}' \\ \mathbf{z}_{hi}' \end{bmatrix} (\mathbf{y} - \boldsymbol{\pi}) = \begin{bmatrix} \mathbf{0} \\ y_{hi} - \pi_{hi} \end{bmatrix} \\
\boldsymbol{\Lambda}(\boldsymbol{\beta}^0) &= \begin{bmatrix} \mathbf{X}' \\ \mathbf{z}_{hi}' \end{bmatrix} \mathbf{U} [\mathbf{X} \quad \mathbf{z}_{hi}] = \begin{bmatrix} \boldsymbol{\Lambda}(\boldsymbol{\beta}) & \mathbf{X}'\mathbf{U}\mathbf{z}_{hi} \\ \mathbf{z}_{hi}'\mathbf{U}\mathbf{X} & \mathbf{z}_{hi}'\mathbf{U}\mathbf{z}_{hi} \end{bmatrix}
\end{aligned}
$$

Inserting the $\boldsymbol{\beta}^0$ and $(\mathbf{X}', \mathbf{z}_{hi}')'$ into the Gail, Lubin, and Rubinstein (1981) algorithm provides the appropriate estimates of $\mathbf{g}(\boldsymbol{\beta}^0)$ and $\boldsymbol{\Lambda}(\boldsymbol{\beta}^0)$. Indicate these estimates with $\widehat{\boldsymbol{\pi}} = \boldsymbol{\pi}(\widehat{\boldsymbol{\beta}})$, $\widehat{\mathbf{U}} = \mathbf{U}(\widehat{\boldsymbol{\beta}})$, $\widehat{\mathbf{g}}$, and $\widehat{\boldsymbol{\Lambda}}$.

DFBETA is computed from the information matrix as

$$
\begin{aligned}
\Delta_{hi}\boldsymbol{\beta} &= \boldsymbol{\beta}^0 - \boldsymbol{\beta}_{hi}^1 \\
&= -\widehat{\boldsymbol{\Lambda}}^{-1}(\boldsymbol{\beta}^0)\widehat{\mathbf{g}}(\boldsymbol{\beta}^0) \\
&= -\widehat{\boldsymbol{\Lambda}}^{-1}(\widehat{\boldsymbol{\beta}})(\mathbf{X}'\widehat{\mathbf{U}}\mathbf{z}_{hi})\mathbf{M}^{-1}\mathbf{z}_{hi}'(\mathbf{y} - \widehat{\boldsymbol{\pi}})
\end{aligned}
$$

where

$$
\mathbf{M} = (\mathbf{z}_{hi}'\widehat{\mathbf{U}}\mathbf{z}_{hi}) - (\mathbf{z}_{hi}'\widehat{\mathbf{U}}\mathbf{X})\widehat{\boldsymbol{\Lambda}}^{-1}(\widehat{\boldsymbol{\beta}})(\mathbf{X}'\widehat{\mathbf{U}}\mathbf{z}_{hi})
$$

For each observation in the data set, a DFBETA statistic is computed for each parameter β_j, $1 \le j \le p$, and standardized by the standard error of β_j from the full data set to produce the estimate of DFBETAS.

The estimated leverage is defined as

$$h_{hi} = \frac{\text{trace}\{(\mathbf{z}'_{hi}\widehat{\mathbf{U}}\mathbf{X})\widehat{\boldsymbol{\Lambda}}^{-1}(\widehat{\boldsymbol{\beta}})(\mathbf{X}'\widehat{\mathbf{U}}\mathbf{z}_{hi})\}}{\text{trace}\{\mathbf{z}'_{hi}\widehat{\mathbf{U}}\mathbf{z}_{hi}\}}$$

This definition of leverage produces different values from those defined by Pregibon (1984); Moolgavkar, Lustbader, and Venzon (1985); Hosmer and Lemeshow (2000); however, it has the advantage that no extra computations beyond those for the DFBETAS are required.

The estimated residuals $e_{hi} = y_{hi} - \widehat{\pi}_{hi}$ are obtained from $\widehat{\mathbf{g}}(\boldsymbol{\beta}^0)$, and the weights, or predicted probabilities, are then $\widehat{\pi}_{hi} = y_{hi} - e_{hi}$. The residuals are standardized and reported as (estimated) Pearson residuals:

$$\frac{r_{hi} - n_{hi}\widehat{\pi}_{hi}}{\sqrt{n_{hi}\widehat{\pi}_{hi}(1-\widehat{\pi}_{hi})}}$$

where r_{hi} is the number of events in the observation and n_{hi} is the number of trials.

The STDRES option in the INFLUENCE option computes the standardized Pearson residual:

$$e_{s,hi} = \frac{e_{hi}}{\sqrt{1-h_{hi}}}$$

For events/trials MODEL statement syntax, treat each observation as two observations (the first for the nonevents and the second for the events) with frequencies $f_{h,2i-1} = n_{hi} - r_{hi}$ and $f_{h,2i} = r_{hi}$, and augment the model with a matrix $\mathbf{Z}_{hi} = [\mathbf{z}_{h,2i-1}\mathbf{z}_{h,2i}]$ instead of a single $\mathbf{z}_{hi}$ vector. Writing $\gamma_{hi} = \mathbf{x}'_{hi}\boldsymbol{\beta} f_{hi}$ in the preceding section results in the following gradient and information matrix:

$$\begin{aligned}
\mathbf{g}(\boldsymbol{\beta}^0) &= \begin{bmatrix} \mathbf{0} \\ f_{h,2i-1}(y_{h,2i-1} - \pi_{h,2i-1}) \\ f_{h,2i}(y_{h,2i} - \pi_{h,2i}) \end{bmatrix} \\
\boldsymbol{\Lambda}(\boldsymbol{\beta}^0) &= \begin{bmatrix} \boldsymbol{\Lambda}(\boldsymbol{\beta}) & \mathbf{X}'\text{diag}(\mathbf{f})\mathbf{U}\text{diag}(\mathbf{f})\mathbf{Z}_{hi} \\ \mathbf{Z}'_{hi}\text{diag}(\mathbf{f})\mathbf{U}\text{diag}(\mathbf{f})\mathbf{X} & \mathbf{Z}'_{hi}\text{diag}(\mathbf{f})\mathbf{U}\text{diag}(\mathbf{f})\mathbf{Z}_{hi} \end{bmatrix}
\end{aligned}$$

The predicted probabilities are then $\widehat{\pi}_{hi} = y_{h,2i} - e_{h,2i}/r_{h,2i}$, while the leverage and the DFBETAS are produced from $\boldsymbol{\Lambda}(\boldsymbol{\beta}^0)$ in a fashion similar to that for the preceding single-trial equations.

Exact Conditional Logistic Regression

The theory of exact logistic regression, also known as exact conditional logistic regression, was originally laid out by Cox (1970), and the computational methods employed in PROC LOGISTIC are described in Hirji, Mehta, and Patel (1987); Hirji (1992); Mehta, Patel, and Senchaudhuri (1992). Other useful references for the derivations include Cox and Snell (1989); Agresti (1990); Mehta and Patel (1995).

Exact conditional inference is based on generating the conditional distribution for the sufficient statistics of the parameters of interest. This distribution is called the *permutation* or *exact conditional* distribution. Using the notation in the section "Computational Details" on page 6068, follow Mehta and Patel (1995) and first note that the sufficient statistics $\mathbf{T} = (T_1, \ldots, T_p)$ for the parameter vector of intercepts and slopes, $\boldsymbol{\beta}$, are

$$T_j = \sum_{i=1}^{n} y_i x_{ij}, \quad j = 1, \ldots, p$$

Denote a vector of observable sufficient statistics as $\mathbf{t} = (t_1, \ldots, t_p)'$.

The probability density function (PDF) for $\mathbf{T}$ can be created by summing over all binary sequences $\mathbf{y}$ that generate an observable $\mathbf{t}$ and letting $C(\mathbf{t}) = ||\{\mathbf{y} : \mathbf{y}'\mathbf{X} = \mathbf{t}'\}||$ denote the number of sequences $\mathbf{y}$ that generate $\mathbf{t}$

$$\Pr(\mathbf{T} = \mathbf{t}) = \frac{C(\mathbf{t}) \exp(\mathbf{t}'\boldsymbol{\beta})}{\prod_{i=1}^{n}[1 + \exp(\mathbf{x}_i'\boldsymbol{\beta})]}$$

In order to condition out the nuisance parameters, partition the parameter vector $\boldsymbol{\beta} = (\boldsymbol{\beta}_{\mathrm{N}}', \boldsymbol{\beta}_{\mathrm{I}}')'$, where $\boldsymbol{\beta}_{\mathrm{N}}$ is a $p_{\mathrm{N}} \times 1$ vector of the nuisance parameters, and $\boldsymbol{\beta}_{\mathrm{I}}$ is the parameter vector for the remaining $p_{\mathrm{I}} = p - p_{\mathrm{N}}$ parameters of interest. Likewise, partition $\mathbf{X}$ into $\mathbf{X}_{\mathrm{N}}$ and $\mathbf{X}_{\mathrm{I}}$, $\mathbf{T}$ into $\mathbf{T}_{\mathrm{N}}$ and $\mathbf{T}_{\mathrm{I}}$, and $\mathbf{t}$ into $\mathbf{t}_{\mathrm{N}}$ and $\mathbf{t}_{\mathrm{I}}$. The nuisance parameters can be removed from the analysis by conditioning on their sufficient statistics to create the conditional likelihood of $\mathbf{T}_{\mathrm{I}}$ given $\mathbf{T}_{\mathrm{N}} = \mathbf{t}_{\mathrm{N}}$,

$$\begin{aligned}\Pr(\mathbf{T}_{\mathrm{I}} = \mathbf{t}_{\mathrm{I}} | \mathbf{T}_{\mathrm{N}} = \mathbf{t}_{\mathrm{N}}) &= \frac{\Pr(\mathbf{T} = \mathbf{t})}{\Pr(\mathbf{T}_{\mathrm{N}} = \mathbf{t}_{\mathrm{N}})} \\ = f_{\boldsymbol{\beta}_{\mathrm{I}}}(\mathbf{t}_{\mathrm{I}} | \mathbf{t}_{\mathrm{N}}) &= \frac{C(\mathbf{t}_{\mathrm{N}}, \mathbf{t}_{\mathrm{I}}) \exp(\mathbf{t}_{\mathrm{I}}'\boldsymbol{\beta}_{\mathrm{I}})}{\sum_u C(\mathbf{t}_{\mathrm{N}}, \mathbf{u}) \exp(\mathbf{u}'\boldsymbol{\beta}_{\mathrm{I}})}\end{aligned}$$

where $C(\mathbf{t}_{\mathrm{N}}, \mathbf{u})$ is the number of vectors $\mathbf{y}$ such that $\mathbf{y}'\mathbf{X}_{\mathrm{N}} = \mathbf{t}_{\mathrm{N}}$ and $\mathbf{y}'\mathbf{X}_{\mathrm{I}} = \mathbf{u}$. Note that the nuisance parameters have factored out of this equation, and that $C(\mathbf{t}_{\mathrm{N}}, \mathbf{t}_{\mathrm{I}})$ is a constant.

The goal of the exact conditional analysis is to determine how likely the observed response $\mathbf{y}_0$ is with respect to all 2^n possible responses $\mathbf{y} = (y_1, \ldots, y_n)'$. One way to proceed is to generate every $\mathbf{y}$ vector for which $\mathbf{y}'\mathbf{X}_{\mathrm{N}} = \mathbf{t}_{\mathrm{N}}$, and count the number of vectors $\mathbf{y}$ for which $\mathbf{y}'\mathbf{X}_{\mathrm{I}}$ is equal to each unique $\mathbf{t}_{\mathrm{I}}$. Generating the conditional distribution from complete enumeration of the joint distribution is conceptually simple; however, this method becomes computationally infeasible very quickly. For example, if you had only 30 observations, you would have to scan through 2^{30} different $\mathbf{y}$ vectors.

Several algorithms are available in PROC LOGISTIC to generate the exact distribution. All of the algorithms are based on the following observation. Given any $\mathbf{y} = (y_1, \ldots, y_n)'$ and a design $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)'$, let $\mathbf{y}_{(i)} = (y_1, \ldots, y_i)'$ and $\mathbf{X}_{(i)} = (\mathbf{x}_1, \ldots, \mathbf{x}_i)'$ be the first i rows of each matrix. Write the sufficient statistic based on these i rows as $\mathbf{t}_{(i)}' = \mathbf{y}_{(i)}'\mathbf{X}_{(i)}$. A recursion relation results: $\mathbf{t}_{(i+1)} = \mathbf{t}_{(i)} + y_{i+1}\mathbf{x}_{i+1}$.

The following methods are available:

- The *multivariate shift algorithm* developed by Hirji, Mehta, and Patel (1987), which steps through the recursion relation by adding one observation at a time and building an intermediate distribution at each step. If it determines that $\mathbf{t}_{(i)}$ for the nuisance parameters could eventually equal $\mathbf{t}$, then $\mathbf{t}_{(i)}$ is added to the intermediate distribution.

- An extension of the multivariate shift algorithm to generalized logit models by Hirji (1992). Because the generalized logit model fits a new set of parameters to each logit, the number of parameters in the model can easily get too large for this algorithm to handle. Note for these models that the hypothesis tests for each effect are computed across the logit functions, while individual parameters are estimated for each logit function.

- A network algorithm described in Mehta, Patel, and Senchaudhuri (1992), which builds a network for each parameter that you are conditioning out in order to identify feasible y_i for the $\mathbf{y}$ vector. These

networks are combined and the set of feasible y_i is further reduced, and then the multivariate shift algorithm uses this knowledge to build the exact distribution without adding as many intermediate $\mathbf{t}_{(i+1)}$ as the multivariate shift algorithm does.

- A hybrid Monte Carlo and network algorithm described by Mehta, Patel, and Senchaudhuri (2000), which extends their 1992 algorithm by sampling from the combined network to build the exact distribution.
- A Markov chain Monte Carlo algorithm described by Forster, McDonald, and Smith (2003), which generates the exact distribution by repeatedly perturbing the response vector to obtain a new response vector while maintaining the sufficient statistics for the nuisance parameters, and the resulting $\mathbf{t}$ are added to the exact distribution.

The bulk of the computation time and memory for these algorithms is consumed by the creation of the networks and the exact joint distribution. After the joint distribution for a set of effects is created, the computational effort required to produce hypothesis tests and parameter estimates for any subset of the effects is (relatively) trivial. See the section "Computational Resources for Exact Logistic Regression" on page 6082 for more computational notes about exact analyses.

NOTE: An alternative to using these exact conditional methods is to perform Firth's bias-reducing penalized likelihood method (see the FIRTH option in the MODEL statement); this method has the advantage of being much faster and less memory intensive than exact algorithms, but it might not converge to a solution.

Hypothesis Tests

Consider testing the null hypothesis $H_0\colon \boldsymbol{\beta}_\mathrm{I} = \mathbf{0}$ against the alternative $H_A\colon \boldsymbol{\beta}_\mathrm{I} \neq \mathbf{0}$, conditional on $\mathbf{T}_\mathrm{N} = \mathbf{t}_\mathrm{N}$. Under the null hypothesis, the test statistic for the *exact probability test* is just $f_{\boldsymbol{\beta}_\mathrm{I}=\mathbf{0}}(\mathbf{t}_\mathrm{I}|\mathbf{t}_\mathrm{N})$, while the corresponding p-value is the probability of getting a less likely (more extreme) statistic,

$$p(\mathbf{t}_\mathrm{I}|\mathbf{t}_\mathrm{N}) = \sum_{u \in \Omega_p} f_0(\mathbf{u}|\mathbf{t}_\mathrm{N})$$

where $\Omega_p = \{\mathbf{u}\text{: there exist } \mathbf{y} \text{ with } \mathbf{y}'\mathbf{X}_\mathrm{I} = \mathbf{u},\ \mathbf{y}'\mathbf{X}_\mathrm{N} = \mathbf{t}_\mathrm{N}, \text{ and } f_0(\mathbf{u}|\mathbf{t}_\mathrm{N}) \leq f_0(\mathbf{t}_\mathrm{I}|\mathbf{t}_\mathrm{N})\}$.

The exact probability test is not necessarily a sum of tail areas and can be inflated if the distribution is skewed. The more robust *exact conditional scores test* is a sum of tail areas and is usually preferred to the exact probability test. To compute the exact conditional scores test, the conditional mean $\boldsymbol{\mu}_\mathrm{I}$ and variance matrix $\boldsymbol{\Sigma}_\mathrm{I}$ of the $\mathbf{T}_\mathrm{I}$ (conditional on $\mathbf{T}_\mathrm{N} = \mathbf{t}_\mathrm{N}$) are calculated, and the score statistic for the observed value,

$$s = (\mathbf{t}_\mathrm{I} - \boldsymbol{\mu}_\mathrm{I})'\boldsymbol{\Sigma}_\mathrm{I}^{-1}(\mathbf{t}_\mathrm{I} - \boldsymbol{\mu}_\mathrm{I})$$

is compared to the score for each member of the distribution

$$S(\mathbf{T}_\mathrm{I}) = (\mathbf{T}_\mathrm{I} - \boldsymbol{\mu}_\mathrm{I})'\boldsymbol{\Sigma}_\mathrm{I}^{-1}(\mathbf{T}_\mathrm{I} - \boldsymbol{\mu}_\mathrm{I})$$

The resulting p-value is

$$p(\mathbf{t}_\mathrm{I}|\mathbf{t}_\mathrm{N}) = \Pr(S \geq s) = \sum_{u \in \Omega_s} f_0(\mathbf{u}|\mathbf{t}_\mathrm{N})$$

where $\Omega_s = \{\mathbf{u}\text{: there exist } \mathbf{y} \text{ with } \mathbf{y}'\mathbf{X}_\mathrm{I} = \mathbf{u},\ \mathbf{y}'\mathbf{X}_\mathrm{N} = \mathbf{t}_\mathrm{N}, \text{ and } S(\mathbf{u}) \geq s\}$.

The mid-p statistic, defined as

$$p(\mathbf{t}_{\mathrm{I}}|\mathbf{t}_{\mathrm{N}}) - \frac{1}{2} f_0(\mathbf{t}_{\mathrm{I}}|\mathbf{t}_{\mathrm{N}})$$

was proposed by Lancaster (1961) to compensate for the discreteness of a distribution. See Agresti (1992) for more information. However, to allow for more flexibility in handling ties, you can write the mid-p statistic as (based on a suggestion by LaMotte (2002) and generalizing Vollset, Hirji, and Afifi (1991))

$$\sum_{u \in \Omega_<} f_0(\mathbf{u}|\mathbf{t}_{\mathrm{N}}) + \delta_1 f_0(\mathbf{t}_{\mathrm{I}}|\mathbf{t}_{\mathrm{N}}) + \delta_2 \sum_{u \in \Omega_=} f_0(\mathbf{u}|\mathbf{t}_{\mathrm{N}})$$

where, for $i \in \{p, s\}$, $\Omega_<$ is Ω_i using strict inequalities, and $\Omega_=$ is Ω_i using equalities with the added restriction that $\mathbf{u} \neq \mathbf{t}_{\mathrm{I}}$. Letting $(\delta_1, \delta_2) = (0.5, 1.0)$ yields Lancaster's mid-p.

CAUTION: When the exact distribution has ties and you specify METHOD=NETWORKMC or METHOD=MCMC, the sampling algorithm estimates $p(\mathbf{t}|\mathbf{t}_{\mathrm{N}})$ with error, and hence it cannot determine precisely which values contribute to the reported p-values. For example, if the exact distribution has densities $\{0.2, 0.2, 0.2, 0.4\}$ and if the observed statistic has probability 0.2, then the exact probability p-value is exactly 0.6. Under Monte Carlo sampling, if the densities after N samples are $\{0.18, 0.21, 0.23, 0.38\}$ and the observed probability is 0.21, then the resulting p-value is 0.39. Therefore, the exact probability test p-value for this example fluctuates between 0.2, 0.4, and 0.6, and the reported p-values are actually lower bounds for the true p-values. If you need more precise values, you can specify the OUTDIST= option, determine appropriate cutoff values for the observed probability and score, and then construct p-value estimates from the OUTDIST= data set and display them in the SAS log by using the following statements:

```
data _null_;
   set outdist end=end;
   retain pvalueProb 0 pvalueScore 0;
   if prob < ProbCutOff then pvalueProb+prob;
   if score > ScoreCutOff then pvalueScore+prob;
   if end then put pvalueProb= pvalueScore=;
run;
```

Because the METHOD=MCMC samples are correlated, the covariance that is computed for the exact conditional scores test is biased. Specifying the NTHIN= option might reduce this bias.

Inference for a Single Parameter

Exact parameter estimates are derived for a single parameter β_i by regarding all the other parameters $\boldsymbol{\beta}_{\mathrm{N}} = (\beta_1, \ldots, \beta_{i-1}, \beta_{i+1}, \ldots, \beta_{p_{\mathrm{N}}+p_{\mathrm{I}}})'$ as nuisance parameters. The appropriate sufficient statistics are $\mathbf{T}_{\mathrm{I}} = T_i$ and $\mathbf{T}_{\mathrm{N}} = (T_1, \ldots, T_{i-1}, T_{i+1}, \ldots, T_{p_{\mathrm{N}}+p_{\mathrm{I}}})'$, with their observed values denoted by the lowercase t. Hence, the conditional PDF used to create the parameter estimate for β_i is

$$f_{\beta_i}(t_i|\mathbf{t}_{\mathrm{N}}) = \frac{C(\mathbf{t}_{\mathrm{N}}, t_i)\exp(t_i\beta_i)}{\sum_{u \in \Omega} C(\mathbf{t}_{\mathrm{N}}, u)\exp(u\beta_i)}$$

for $\Omega = \{u\colon \text{ there exist } \mathbf{y} \text{ with } T_i = u \text{ and } \mathbf{T}_{\mathrm{N}} = \mathbf{t}_{\mathrm{N}}\}$.

The maximum exact conditional likelihood estimate is the quantity $\widehat{\beta}_i$, which maximizes the conditional PDF. A Newton-Raphson algorithm is used to perform this search. However, if the observed t_i attains either its maximum or minimum value in the exact distribution (that is, either $t_i = \min\{u : u \in \Omega\}$ or $t_i = \max\{u : u \in \Omega\}$), then the conditional PDF is monotonically increasing in β_i and cannot be maximized.

In this case, a median unbiased estimate (Hirji, Tsiatis, and Mehta 1989) $\widehat{\beta}_i$ is produced that satisfies $f_{\widehat{\beta}_i}(t_i|\mathbf{t}_\mathrm{N}) = 0.5$, and a Newton-Raphson algorithm is used to perform the search.

The standard error of the exact conditional likelihood estimate is just the negative of the inverse of the second derivative of the exact conditional log likelihood (Agresti 2002).

Likelihood ratio tests based on the conditional PDF are used to test the null $H_0\colon \beta_i = 0$ against the alternative $H_A\colon \beta_i > 0$. The critical region for this UMP test consists of the upper tail of values for T_i in the exact distribution. Thus, the p-value $p_+(t_i;0)$ for a one-tailed test is

$$p_+(t_i;0) = \sum_{u \geq t_i} f_0(u|\mathbf{t}_\mathrm{N})$$

Similarly, the p-value $p_-(t_i;0)$ for the one-tailed test of H_0 against $H_A\colon \beta_i < 0$ is

$$p_-(t_i;0) = \sum_{u \leq t_i} f_0(u|\mathbf{t}_\mathrm{N})$$

The p-value $p(t_i;0)$ for a two-tailed test of H_0 against $H_A\colon \beta_i \neq 0$ is

$$p(t_i;0) = 2\min[p_-(t_i;0),\, p_+(t_i;0)]$$

By default, the p-value that is reported for a single parameter in the "Exact Parameter Estimates" table is for the two-tailed test. For median unbiased estimates, the p-value for a one-tailed test is always reported.

An upper $100(1-2\epsilon)\%$ exact confidence limit for $\widehat{\beta}_i$ corresponding to the observed t_i is the solution $\beta_U(t_i)$ of $\epsilon = p_-(t_i, \beta_U(t_i))$, while the lower exact confidence limit is the solution $\beta_L(t_i)$ of $\epsilon = p_+(t_i, \beta_L(t_i))$. Again, a Newton-Raphson procedure is used to search for the solutions. Note that one of the confidence limits for a median unbiased estimate is set to infinity and the other is computed at 2ϵ, which results in the display of a one-sided $100(1-2\epsilon)\%$ confidence interval.

Specifying the ONESIDED option displays only one p-value and one confidence interval, because small values of $p_+(t_i;0)$ and $p_-(t_i;0)$ support different alternative hypotheses and only one of these p-values can be less than 0.50.

The mid-p confidence limits are the solutions to $\min\{p_-(t_i, \beta(t_i)),\, p_+(t_i, \beta(t_i))\} - (1-\delta_1) f_{\beta(t_i)}(t_i|\mathbf{t}_\mathrm{N}) = \epsilon$ for $\epsilon = \alpha/2, 1-\alpha/2$ (Vollset, Hirji, and Afifi 1991). $\delta_1 = 1$ produces the usual exact (or *max-p*) confidence interval, $\delta_1 = 0.5$ yields the mid-p interval, and $\delta_1 = 0$ gives the *min-p* interval. The mean of the endpoints of the max-p and min-p intervals provides the *mean-p* interval as defined by Hirji, Mehta, and Patel (1988).

Estimates and confidence intervals for the odds ratios are produced by exponentiating the estimates and interval endpoints for the parameters.

Input and Output Data Sets

OUTEST= Output Data Set

The OUTEST= data set contains one observation for each BY group containing the maximum likelihood estimates of the regression coefficients. If you also use the COVOUT option in the PROC LOGISTIC statement, there are additional observations containing the rows of the estimated covariance matrix. If you specify SELECTION=FORWARD, BACKWARD, or STEPWISE, only the estimates of the parameters and covariance matrix for the final model are output to the OUTEST= data set.

Variables in the OUTEST= Data Set

The OUTEST= data set contains the following variables:

- any BY variables specified
- _LINK_, a character variable of length 8 with four possible values: CLOGLOG for the complementary log-log function, LOGIT for the logit function, NORMIT for the probit (alias normit) function, and GLOGIT for the generalized logit function
- _TYPE_, a character variable of length 8 with two possible values: PARMS for parameter estimates or COV for covariance estimates. If an EXACT statement is also specified, then two other values are possible: EPARMMLE for the exact maximum likelihood estimates and EPARMMUE for the exact median unbiased estimates.
- _NAME_, a character variable containing the name of the response variable when _TYPE_=PARMS, EPARMMLE, and EPARMMUE, or the name of a model parameter when _TYPE_=COV
- _STATUS_, a character variable that indicates whether the estimates have converged
- one variable for each intercept parameter
- one variable for each slope parameter and one variable for the offset variable if the OFFSET= option if specified. If an effect is not included in the final model in a model building process, the corresponding parameter estimates and covariances are set to missing values.
- _LNLIKE_, the log likelihood

Parameter Names in the OUTEST= Data Set

If there are only two response categories in the entire data set, the intercept parameter is named Intercept. If there are more than two response categories in the entire data set, the intercept parameters are named Intercept_xxx, where xxx is the value (formatted if a format is applied) of the corresponding response category.

For continuous explanatory variables, the names of the parameters are the same as the corresponding variables. For CLASS variables, the parameter names are obtained by concatenating the corresponding CLASS variable name with the CLASS category; for more information, see the section "Class Variable Naming Convention" on page 5971. For interaction and nested effects, the parameter names are created by concatenating the names of each effect.

For multinomial response functions, names of unconstrained unequal slope parameters that correspond to each nonreference response category contain _xxx as the suffix, where xxx is the value (formatted if a format is applied) of the corresponding nonreference response category. For example, suppose the variable Net3 represents the television network (ABC, CBS, and NBC) that is viewed at a certain time. The following statements fit a generalized logit model that uses Age and Gender (a CLASS variable that has values Female and Male) as explanatory variables:

```
proc logistic;
   class Gender;
   model Net3 = Age Gender / link=glogit;
run;
```

There are two logit functions, one that contrasts ABC with NBC and one that contrasts CBS with NBC. For each logit, there are three parameters: an intercept parameter, a slope parameter for Age, and a slope parameter for Gender (because there are only two gender levels and the EFFECT parameterization is used by default). The names of the parameters and their descriptions are as follows:

Intercept_ABC	intercept parameter for the logit that contrasts ABC with NBC
Intercept_CBS	intercept parameter for the logit that contrasts CBS with NBC
Age_ABC	Age slope parameter for the logit that contrasts ABC with NBC
Age_CBS	Age slope parameter for the logit that contrasts CBS with NBC
GenderFemale_ABC	Gender=Female slope parameter for the logit that contrasts ABC with NBC
GenderFemale_CBS	Gender=Female slope parameter for the logit that contrasts CBS with NBC

In a polytomous response model, if you specify an effect in both the EQUALSLOPES and UNEQUALSLOPES options, then its unequal slope parameter names are prefixed with "U_" if they are unconstrained or "C_" if they are constrained.

INEST= Input Data Set

You can specify starting values for the iterative algorithm in the INEST= data set. The INEST= data set has the same structure as the OUTEST= data set but is not required to have all the variables or observations that appear in the OUTEST= data set. A previous OUTEST= data set can be used as, or modified for use as, an INEST= data set.

The INEST= data set must contain the intercept variables (named Intercept for binary response models and Intercept, Intercept_2, Intercept_3, and so forth, for ordinal and nominal response models) and all explanatory variables in the MODEL statement. If BY processing is used, the INEST= data set should also include the BY variables, and there must be one observation for each BY group. If the INEST= data set also contains the _TYPE_ variable, only observations with _TYPE_ value 'PARMS' are used as starting values.

OUT= Output Data Set in the OUTPUT Statement

The OUT= data set in the OUTPUT statement contains all the variables in the input data set along with statistics you request by specifying *keyword=name* options or the PREDPROBS= option in the OUTPUT statement. In addition, if you use the single-trial syntax and you request any of the XBETA=, STDXBETA=, PREDICTED=, LCL=, and UCL= options, the OUT= data set contains the automatic variable _LEVEL_. The value of _LEVEL_ identifies the response category upon which the computed values of XBETA=, STDXBETA=, PREDICTED=, LCL=, and UCL= are based.

When there are more than two response levels, only variables named by the XBETA=, STDXBETA=, PREDICTED=, LOWER=, and UPPER= options and the variables given by PREDPROBS=(INDIVIDUAL CUMULATIVE) have their values computed; the other variables have missing values. If you fit a generalized logit model, the cumulative predicted probabilities are not computed.

When there are only two response categories, each input observation produces one observation in the OUT= data set.

If there are more than two response categories and you specify only the PREDPROBS= option, then each input observation produces one observation in the OUT= data set. However, if you fit an ordinal (cumulative) model and specify options other than the PREDPROBS= options, each input observation generates as many output observations as one fewer than the number of response levels, and the predicted probabilities and their confidence limits correspond to the cumulative predicted probabilities. If you fit a generalized logit model and specify options other than the PREDPROBS= options, each input observation generates as many output observations as the number of response categories; the predicted probabilities and their confidence limits correspond to the probabilities of individual response categories.

For observations in which only the response variable is missing, values of the XBETA=, STDXBETA=, PREDICTED=, UPPER=, LOWER=, and the PREDPROBS= options are computed even though these observations do not affect the model fit. This enables, for instance, predicted probabilities to be computed for new observations.

OUT= Output Data Set in a SCORE Statement

The OUT= data set in a SCORE statement contains all the variables in the data set being scored. The data set being scored can be either the input DATA= data set in the PROC LOGISTIC statement or the DATA= data set in the SCORE statement. The DATA= data set in the SCORE statement does not need to contain the response variable.

If the data set being scored contains the response variable, then denote the *normalized* levels (left-justified, formatted values of 16 characters or less) of your response variable Y by $Y_1, \ldots, Y_{k+1}$. For each response level, the OUT= data set also contains the following:

- F_Y, the normalized levels of the response variable Y in the data set being scored. If the events/trials syntax is used, the F_Y variable is not created.
- I_Y, the normalized levels that the observations are classified into. Note that an observation is classified into the level with the largest probability. If the events/trials syntax is used, the _INTO_ variable is created instead, and it contains the values EVENT and NONEVENT.
- P_Y_i, the posterior probabilities of the normalized response level Y_i
- If the CLM option is specified in the SCORE statement, the OUT= data set also includes the following:
  - LCL_Y_i, the lower $100(1-\alpha)\%$ confidence limits for P_Y_i
  - UCL_Y_i, the upper $100(1-\alpha)\%$ confidence limits for P_Y_i

OUTDESIGN= Output Data Set

The OUTDESIGN= data set contains the same number of observations as the corresponding DATA= data set and includes the response variable (with the same format as in the DATA= data set), the FREQ variable, the WEIGHT variable, the OFFSET= variable, and the design variables for the covariates, including an Intercept variable of constant value 1 unless the NOINT option in the MODEL statement is specified. This data set does not contain the full design matrix because the covariate columns are not distinguished by response function.

OUTDIST= Output Data Set

The OUTDIST= data set contains every exact conditional distribution necessary to process the corresponding EXACT statement. For example, the following statements create one distribution for the x1 parameter and another for the x2 parameters, and produce the data set dist shown in Table 79.17:

```
data test;
   input y x1 x2 count;
   datalines;
0 0 0 1
1 0 0 1
0 1 1 2
1 1 1 1
1 0 2 3
1 1 2 1
1 2 0 3
1 2 1 2
1 2 2 1
;

proc logistic data=test exactonly;
   class x2 / param=ref;
   model y=x1 x2;
   exact x1 x2/ outdist=dist;
run;
proc print data=dist;
run;
```

Table 79.17 OUTDIST= Data Set

Obs	x1	x20	x21	Count	Score	Prob
1	.	0	0	3	5.81151	0.03333
2	.	0	1	15	1.66031	0.16667
3	.	0	2	9	3.12728	0.10000
4	.	1	0	15	1.46523	0.16667
5	.	1	1	18	0.21675	0.20000
6	.	1	2	6	4.58644	0.06667
7	.	2	0	19	1.61869	0.21111
8	.	2	1	2	3.27293	0.02222
9	.	3	0	3	6.27189	0.03333
10	2	.	.	6	3.03030	0.12000
11	3	.	.	12	0.75758	0.24000
12	4	.	.	11	0.00000	0.22000
13	5	.	.	18	0.75758	0.36000
14	6	.	.	3	3.03030	0.06000

The first nine observations in the dist data set contain an exact distribution for the parameters of the x2 effect (hence the values for the x1 parameter are missing), and the remaining five observations are for the x1 parameter. If a joint distribution was created, there would be observations with values for both the x1 and x2 parameters. For CLASS variables, the corresponding parameters in the dist data set are identified by concatenating the variable name with the appropriate classification level.

The data set contains the possible sufficient statistics of the parameters for the effects specified in the EXACT statement, and the Count variable contains the number of different responses that yield these statistics. In particular, there are six possible response vectors $\mathbf{y}$ for which the dot product $\mathbf{y}'\mathbf{x1}$ was equal to 2, and for which $\mathbf{y}'\mathbf{x20}$, $\mathbf{y}'\mathbf{x21}$, and $\mathbf{y}'\mathbf{1}$ were equal to their actual observed values (displayed in the "Sufficient Statistics" table).

When hypothesis tests are performed on the parameters, the Prob variable contains the probability of obtaining that statistic (which is just the count divided by the total count), and the Score variable contains the score for that statistic.

The OUTDIST= data set can contain a different exact conditional distribution for each specified EXACT statement. For example, consider the following EXACT statements:

```
exact 'O1'   x1    /           outdist=o1;
exact 'OJ12' x1 x2 / jointonly outdist=oj12;
exact 'OA12' x1 x2 / joint     outdist=oa12;
exact 'OE12' x1 x2 / estimate  outdist=oe12;
```

The O1 statement outputs a single exact conditional distribution. The OJ12 statement outputs only the joint distribution for x1 and x2. The OA12 statement outputs three conditional distributions: one for x1, one for x2, and one jointly for x1 and x2. The OE12 statement outputs two conditional distributions: one for x1 and the other for x2. Data set oe12 contains both the x1 and x2 variables; the distribution for x1 has missing values in the x2 column while the distribution for x2 has missing values in the x1 column.

OUTROC= Output Data Set

The OUTROC= data set contains data necessary for producing the ROC curve, and can be created by specifying the OUTROC= option in the MODEL statement or the OUTROC= option in the SCORE statement: It has the following variables:

- any BY variables specified
- _STEP_, the model step number. This variable is not included if model selection is not requested.
- _PROB_, the estimated probability of an event. These estimated probabilities serve as cutpoints for predicting the response. Any observation with an estimated event probability that exceeds or equals _PROB_ is predicted to be an event; otherwise, it is predicted to be a nonevent. Predicted probabilities that are close to each other are grouped together, with the maximum allowable difference between the largest and smallest values less than a constant that is specified by the ROCEPS= option. The smallest estimated probability is used to represent the group.
- _POS_, the number of correctly predicted event responses
- _NEG_, the number of correctly predicted nonevent responses
- _FALPOS_, the number of falsely predicted event responses
- _FALNEG_, the number of falsely predicted nonevent responses
- _SENSIT_, the sensitivity, which is the proportion of event observations that were predicted to have an event response
- _1MSPEC_, one minus specificity, which is the proportion of nonevent observations that were predicted to have an event response
- any specified ID variables

Note that none of these statistics are affected by the bias-correction method discussed in the section "Classification Table" on page 6047. An ROC curve is obtained by plotting _SENSIT_ against _1MSPEC_.

For each optimal statistic that you specify in the OPTIMAL= option, two variables are added to the data set. The first variable, which contains the statistic, is named by appending underbars to the option names. The second variable is named by adding "OPT" to the first variable's name; it is 1 when the statistic is at its optimal value and 0 otherwise. For example, if you specify the OPTIMAL=DIST01 option, then the first variable is named _DIST01_ and the second variable is named _OPTDIST01_. If you specify more than one F statistic, then the β are appended; similarly, if you specify more than one COST or MCOST statistic, the cost ratios are appended. For example, if one of your β values is 0.5, then the corresponding variables for the F statistic are _F0_5_ and _OPTF0_5_.

For more information, see the section "Receiver Operating Characteristic Curves" on page 6055.

Computational Resources

The memory needed to fit an unconditional model is approximately $8n(p+2)+24(p+2)^2$ bytes, where p is the number of parameters estimated and n is the number of observations in the data set. For cumulative response models with more than two response levels, a test of the parallel lines assumption requires an additional memory of approximately $4k^2(m+1)^2+24(m+2)^2$ bytes, where k is the number of response levels and m is the number of slope parameters. However, if this additional memory is not available, the procedure skips the test and finishes the other computations. You might need more memory if you use the SELECTION= option for model building.

The data that consist of relevant variables (including the design variables for model effects) and observations for fitting the model are stored in a temporary utility file. If sufficient memory is available, such data will also be kept in memory; otherwise, the data are reread from the utility file for each evaluation of the likelihood function and its derivatives, with the resulting execution time of the procedure substantially increased. Specifying the MULTIPASS option in the MODEL statement avoids creating this utility file and also does not store the data in memory; instead, the DATA= data set is reread when needed. This saves approximately $8n(p+2)$ bytes of memory but increases the execution time.

If a conditional logistic regression is performed, then approximately $4(m^2+m+4)\max_h(m_h)+(8s_H+36)H+12s_H$ additional bytes of memory are needed, where m_h is the number of events in stratum h, H is the total number of strata, and s_H is the number of variables used to define the strata. If the CHECKDEPENDENCY=ALL option is specified in the STRATA statement, then an extra $4(m+H)(m+H+1)$ bytes are required, and the resulting execution time of the procedure might be substantially increased.

Computational Resources for Exact Logistic Regression

Many problems require a prohibitive amount of time and memory for exact computations, depending on the speed and memory available on your computer. For such problems, consider whether exact methods are really necessary. Stokes, Davis, and Koch (2012) suggest looking at exact p-values when the sample size is small and the approximate p-values from the unconditional analysis are less than 0.10, and they provide *rules of thumb* for determining when various models are valid.

A formula does not exist that can predict the amount of time and memory necessary to generate the exact conditional distributions for a particular problem. The time and memory required depends on several factors, including the total sample size, the number of parameters of interest, the number of nuisance parameters, and the order in which the parameters are processed. To provide a feel for how these factors affect performance, data sets containing Nobs $\in \{10,\ldots,500\}$ observations with up to 10 independent uniform binary covariates (X1,...,*XN*) and a binary response variable (Y) are generated, and the following statements create exact conditional distributions for X1 conditional on the other covariates by using the default METHOD=NETWORK. Figure 79.11 displays results that were obtained on a 64-bit, 3.4GHz PC with 16GB RAM running Microsoft Windows 7.

```
data one;
   do obs=1 to HalfNobs;
      do Y=0 to 1;
         X1=round(ranuni(0));
         ...
         XN=round(ranuni(0));
         output;
```

```
      end;
   end;
options fullstimer;
proc logistic exactonly;
   exactoptions method=network maxtime=1200;
   class X: / param=ref;
   model Y=X:;
   exact X1 / outdist=dist;
run;
```

Figure 79.11 Mean Time and Memory Required

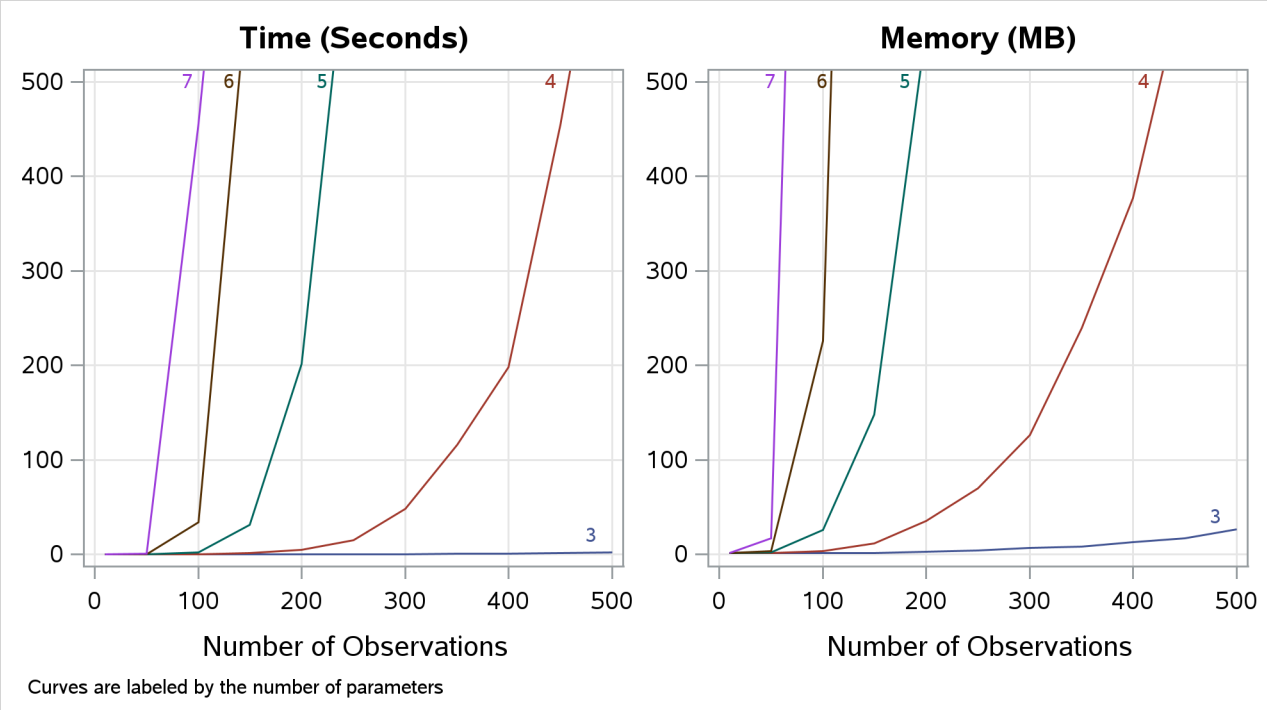

At any time while PROC LOGISTIC is deriving the distributions, you can terminate the computations by pressing the system interrupt key sequence (see the SAS Companion for your system) and choosing to stop computations. If you run out of memory, see the SAS Companion for your system to see how to allocate more.

You can use the EXACTOPTIONS option MAXTIME= to limit the total amount of time PROC LOGISTIC uses to derive all of the exact distributions. If PROC LOGISTIC does not finish within that time, the procedure terminates.

Frequency calculation is performed in the log scale by default. This reduces the need to check for excessively large frequencies but can be slower than not scaling. You can turn off the log scaling by specifying the NOLOGSCALE option in the EXACTOPTIONS statement. If a frequency in the exact distribution is larger than the largest integer that can be held in double precision, a warning is printed to the SAS log. Because inaccuracies due to adding small numbers to these large frequencies might have little or no effect on the statistics, the exact computations continue.

You can monitor the progress of the procedure by submitting your program with the EXACTOPTIONS option

STATUSTIME=. If the procedure is too slow, you can try another method by specifying the EXACTOPTIONS option METHOD=, you can try reordering the variables in the MODEL statement (note that CLASS variables are always processed before continuous covariates), or you can try reparameterizing your classification variables as in the following statement:

```
class class-variables / param=ref ref=first order=freq;
```

If you condition out CLASS variables that use reference or GLM coding but you are not using the STRATA statement, then you can speed up the analysis by specifying one of the nuisance CLASS variables in the STRATA statement. This performance gain occurs because STRATA variables partition your data set into smaller pieces. However, moving two (or more) nuisance CLASS variables into the STRATA statement results in a different model, because the sufficient statistics for the second CLASS variable are actually computed across the levels of the first CLASS variable.

Displayed Output

If you use the NOPRINT option in the PROC LOGISTIC statement, the procedure does not display any output. Otherwise, the tables displayed by the LOGISTIC procedure are discussed in the following section in the order in which they appear in the output. Some of the tables appear only in conjunction with certain options or statements; for more information, see the section "ODS Table Names" on page 6090.

NOTE: The EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, and SLICE statements also create tables, which are not listed in this section. For information about these tables, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Table Summary

Model Information and the Number of Observations

See the section "Missing Values" on page 6026 for information about missing-value handling, and the sections "FREQ Statement" on page 5986 and "WEIGHT Statement" on page 6025 for information about valid frequencies and weights.

Response Profile

Displays the Ordered Value assigned to each response level. For more information, see the section "Response Level Ordering" on page 6026.

Class Level Information

Displays the design values for each CLASS explanatory variable. For more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

Simple Statistics Tables

The following tables are displayed if you specify the SIMPLE option in the PROC LOGISTIC statement:

- **Descriptive Statistics for Continuous Explanatory Variables**
- **Frequency Distribution of Class Variables**

- **Weight Distribution of Class Variables**
  Displays if you also specify a WEIGHT statement.

Strata Tables for (Exact) Conditional Logistic Regression

The following tables are displayed if you specify a STRATA statement:

- **Strata Summary**
  Shows the pattern of the number of events and the number of nonevents in a stratum. See the section "STRATA Statement" on page 6022 for more information.
- **Strata Information**
  Displays if you specify the INFO option in a STRATA statement.

Maximum Likelihood Iteration History

Displays if you specify the ITPRINT option in the MODEL statement. For more information, see the sections "Iterative Algorithms for Model Fitting" on page 6030, "Convergence Criteria" on page 6032, and "Existence of Maximum Likelihood Estimates" on page 6032.

Deviance and Pearson Goodness-of-Fit Statistics

Displays if you specify the SCALE= option in the MODEL statement. Small p-values reject the null hypotheses that the fitted model is adequate. For more information, see the sections "Pearson and Deviance Goodness-of-Fit Test" on page 6050 and "Overdispersion" on page 6053.

Goodness-of-Fit Statistics

Displays if you specify the GOF option in the MODEL statement and you fit a binary logistic regression model. Small p-values reject the null hypotheses that the fitted model is adequate. For more information, see the section "Goodness-of-Fit Tests" on page 6050.

Score Test for the Equal Slopes (Proportional Odds) Assumption

Tests the parallel lines assumption if you fit an ordinal response model with the LINK=CLOGLOG or LINK=PROBIT options. If you specify LINK=LOGIT, this is called the "Proportional Odds" assumption. The table is not displayed if you specify the EQUALSLOPES or UNEQUALSLOPES option in the MODEL statement. Small p-values reject the null hypothesis that the slope parameters for each explanatory variable are constant across all the response functions. For more information, see the section "Testing the Parallel Lines Assumption" on page 6041.

Model Fit Statistics

Computes various fit criteria based on a model with intercepts only and a model with intercepts and explanatory variables. If you specify the NOINT option in the MODEL statement, these statistics are calculated without considering the intercept parameters. If you specify the GOF option in the MODEL statement, then more criteria are displayed in a single column. For more information, see the section "Model Fitting Information" on page 6035.

If you also specify the RSQUARE option in the MODEL statement (without specifying the GOF option), two generalized R-square measures are displayed; for more information, see the section "Generalized Coefficient of Determination" on page 6036.

Testing Global Null Hypothesis: BETA=0

Tests the joint effect of the explanatory variables included in the model. Small p-values reject the null hypothesis that all slope parameters are equal to zero, $H_0\colon \boldsymbol{\beta} = \mathbf{0}$. For more information, see the sections "Model Fitting Information" on page 6035, "Residual Chi-Square" on page 6040, and "Testing Linear Hypotheses about the Regression Coefficients" on page 6059.

Score Test for Global Null Hypothesis

Displays instead of the "Testing Global Null Hypothesis: BETA=0" table if the NOFIT option is specified in the MODEL statement. The global score test evaluates the joint significance of the effects in the MODEL statement. Small p-values reject the null hypothesis that all slope parameters are equal to zero, $H_0\colon \boldsymbol{\beta} = \mathbf{0}$. For more information, see the section "Residual Chi-Square" on page 6040.

Areas

Displays if you specify the ROCOPTIONS(AREA) option in the PROC LOGISTIC statement. For more information, see that option.

Model Selection Tables

The tables in this section are produced when the SELECTION= option is specified in the MODEL statement. See the section "Effect-Selection Methods" on page 6034 for more information.

- **Residual Chi-Square Test**
  Displays if you specify SELECTION=FORWARD, BACKWARD, or STEPWISE in the MODEL statement. Small p-values reject the null hypothesis that the reduced model is adequate. For more information, see the section "Residual Chi-Square" on page 6040.

- **Analysis of Effects Eligible for Entry**
  Displays if you specify the DETAILS option and the SELECTION=FORWARD or STEPWISE option in the MODEL statement. Small p-values reject $H_0\colon \beta_i \neq 0$. The score chi-square is used to determine entry; for more information, see the section "Testing Individual Effects Not in the Model" on page 6040.

- **Analysis of Effects Eligible for Removal**
  Displays if you specify the SELECTION=BACKWARD or STEPWISE option in the MODEL statement. Small p-values reject $H_0\colon \beta_i = 0$. The Wald chi-square is used to determine removal; for more information, see the section "Testing Linear Hypotheses about the Regression Coefficients" on page 6059.

- **Analysis of Effects Removed by Fast Backward Elimination**
  Displays if you specify the FAST option and the SELECTION=BACKWARD or STEPWISE option in the MODEL statement. This table gives the approximate chi-square statistic for the variable removed, the corresponding p-value with respect to a chi-square distribution with one degree of freedom, the residual chi-square statistic for testing the joint significance of the variable and the preceding ones, the degrees of freedom, and the p-value of the residual chi-square with respect to a chi-square distribution with the corresponding degrees of freedom.

- **Summary of Forward, Backward, and Stepwise Selection**
  Displays if you specify SELECTION=FORWARD, BACKWARD, or STEPWISE in the MODEL statement. The score chi-square is used to determine entry; for more information, see the section "Testing Individual Effects Not in the Model" on page 6040. The Wald chi-square is used to determine

removal; for more information, see the section "Testing Linear Hypotheses about the Regression Coefficients" on page 6059.

- **Regression Models Selected by Score Criterion**
  Displays the score chi-square for all models if you specify the SELECTION=SCORE option in the MODEL statement. Small p-values reject the null hypothesis that the fitted model is adequate. For more information, see the sections "Effect-Selection Methods" on page 6034 and "Score Statistics and Tests" on page 6039.

Joint Tests or Type 3 Analysis of Effects

Displays if the model contains a CLASS variable and performs Wald chi-square tests of the joint effect of the parameters for each CLASS variable in the model. Small p-values reject $H_0\colon \boldsymbol{\beta}_i = 0$. The title of this table for main-effects models or models that use GLM parameterization is "Type 3 Analysis of Effects"; for all other models the title is "Joint Tests." For more information, see the sections "Testing Linear Hypotheses about the Regression Coefficients" on page 6059 and "Joint Tests and Type 3 Tests" on page 6059.

Analysis of Maximum Likelihood Estimates

CLASS effects are identified by their (nonreference) level. For generalized logit models, a response variable column displays the nonreference level of the logit. The table includes the following:

- the estimated standard error of the parameter estimate, computed as the square root of the corresponding diagonal element of the estimated covariance matrix
- the Wald chi-square statistic, computed by squaring the ratio of the parameter estimate divided by its standard error estimate. For more information, see the section "Testing Linear Hypotheses about the Regression Coefficients" on page 6059.
- the p-value tests the null hypothesis $H_0\colon \beta_i = 0$; small values reject the null.
- the standardized estimate for the slope parameter, if you specify the STB option in the MODEL statement. For more information, see the STB option on page 6007.
- exponentiated values of the estimates of the slope parameters, if you specify the EXPB option in the MODEL statement. For more information, see the EXPB option on page 5996.
- the label of the variable, if you specify the PARMLABEL option in the MODEL statement and if space permits. Due to constraints on the line size, the variable label might be suppressed in order to display the table in one panel. Use the SAS system option LINESIZE= to specify a larger line size to accommodate variable labels. A shorter line size can break the table into two panels allowing labels to be displayed.

Odds Ratio Estimates

Displays the odds ratio estimates and the corresponding 95% Wald confidence intervals for variables that are not involved in nestings or interactions. For continuous explanatory variables, these odds ratios correspond to a unit increase in the risk factors. For more information, see the section "Odds Ratio Estimation" on page 6043.

Association of Predicted Probabilities and Observed Responses

For more information, see the section "Rank Correlation of Observed Responses and Predicted Probabilities" on page 6039.

Parameter Estimates and Profile-Likelihood or Wald Confidence Intervals

Displays if you specify the CLPARM= option in the MODEL statement. For more information, see the section "Confidence Intervals for Parameters" on page 6041.

Odds Ratio Estimates and Profile-Likelihood or Wald Confidence Intervals

Displays if you specify the ODDSRATIO statement for any effects with any class parameterizations. Also displays if you specify the CLODDS= option in the MODEL statement, except odds ratios are computed only for main effects not involved in interactions or nestings, and if the main effect is a CLASS variable, the parameterization must be EFFECT, REFERENCE, or GLM. For more information, see the section "Odds Ratio Estimation" on page 6043.

Estimated Covariance or Correlation Matrix

Displays if you specify the COVB or CORRB option in the MODEL statement. For more information, see the section "Iterative Algorithms for Model Fitting" on page 6030.

Contrast Test Results

Displays the Wald test for each specified CONTRAST statement. Small p-values reject $H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$. The "Coefficients of Contrast" table displays the contrast matrix if you specify the E option, and the "Contrast Estimation and Testing Results by Row" table displays estimates and Wald tests for each row of the contrast matrix if you specify the ESTIMATE= option. For more information, see the sections "CONTRAST Statement" on page 5973, "Testing Linear Hypotheses about the Regression Coefficients" on page 6059, and "Linear Predictor, Predicted Probability, and Confidence Limits" on page 6046.

Linear Hypotheses Testing Results

Displays the Wald test for each specified TEST statement. For more information, see the sections "Testing Linear Hypotheses about the Regression Coefficients" on page 6059 and "TEST Statement" on page 6023.

Hosmer and Lemeshow Goodness-of-Fit Test

Displays if you specify the LACKFIT or GOF option in the MODEL statement. Small p-values reject the null hypothesis that the fitted model is adequate. If you have 50 or fewer groups, the "Partition for the Hosmer and Lemeshow Test" table displays the grouping that is used in the test. However, if you have more than 50 groups, this table is not displayed, but you can instead use the ODS OUTPUT statement to save this table to an output data set. For more information, see the section "The Hosmer-Lemeshow Goodness-of-Fit Test" on page 6050.

Classification Table

Displays if you use the CTABLE option in the MODEL statement. If you specify a list of cutpoints with the PPROB= option, then the cutpoints are displayed in the Prob Level column. If you specify the prior event probabilities with the PEVENT= option, then the probabilities are displayed in the Prob Event column. The Correct column displays the number of correctly classified events and nonevents, the Incorrect Event column displays the number of nonevents incorrectly classified as events, and the Incorrect Nonevent column gives the number of events incorrectly classified as nonevents. For more information, see the section "Classification Table" on page 6047.

Regression Diagnostics

Displays if you specify the INFLUENCE option in the MODEL statement. See the section "Regression Diagnostics" on page 6060 for more information about diagnostics from an unconditional analysis, and the section "Regression Diagnostic Details" on page 6069 for information about diagnostics from a conditional analysis.

Fit Statistics for SCORE Data

Displays if you specify the FITSTAT option in the SCORE statement. For more information, see the section "Scoring Data Sets" on page 6063.

ROC Association Statistic and Contrast Tables

Displayed if a ROC statement, a ROCCONTRAST statement, or the ROCCI option is specified. For more information about the Mann-Whitney statistics and the test and estimation computations, see the section "ROC Computations" on page 6057. For more information about the other statistics, see the section "Rank Correlation of Observed Responses and Predicted Probabilities" on page 6039.

Exact Conditional Logistic Regression Tables

The tables in this section are produced when the EXACT statement is specified. If the METHOD=NETWORKMC option is specified, the test and estimate tables are renamed "Monte Carlo" tables and a Monte Carlo standard error column ($\sqrt{p(1-p)/n}$) is displayed.

- **Sufficient Statistics**
  Displays if you request an OUTDIST= data set in an EXACT statement. The table lists the parameters and their observed sufficient statistics.

- **(Monte Carlo) Conditional Exact Tests**
  For more information, see the section "Hypothesis Tests" on page 6073.

- **(Monte Carlo) Exact Parameter Estimates**
  Displays if you specify the ESTIMATE option in the EXACT statement. This table gives individual parameter estimates for each variable (conditional on the values of all the other parameters in the model), confidence limits, and a two-sided p-value (twice the one-sided p-value) for testing that the parameter is zero. For more information, see the section "Inference for a Single Parameter" on page 6074.

- **(Monte Carlo) Exact Odds Ratios**
  Displays if you specify the ESTIMATE=ODDS or ESTIMATE=BOTH option in the EXACT statement. For more information, see the section "Inference for a Single Parameter" on page 6074.

ODS Table Names

PROC LOGISTIC assigns a name to each table it creates. You can refer to the tables by these names when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 79.18. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

The EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, and SLICE statements also create tables, which are not listed in Table 79.18. For information about these tables, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Table 79.18 ODS Tables Produced by PROC LOGISTIC

ODS Table Name	Description	Statements	Options
Areas	Area under the curves	PROC LOGISTIC	ROCOPTIONS(AREA) (with ROC and PR curves)
Association	Association of predicted probabilities and observed responses	MODEL (without STRATA)	Default
BestSubsets	Best subset selection	MODEL	SELECTION=SCORE
ClassFreq	Frequency breakdown of CLASS variables	PROC LOGISTIC	SIMPLE (with CLASS variables)
Classification	Classification table	MODEL	CTABLE
ClassLevelInfo	CLASS variable levels and design variables	MODEL	Default (with CLASS variables)
ClassWgt	Weight breakdown of CLASS variables	PROC LOGISTIC, WEIGHT	SIMPLE (with CLASS variables)
CLOddsPL	Odds ratio estimates and profile-likelihood confidence intervals	MODEL	CLODDS=PL
CLOddsWald	Odds ratio estimates and Wald confidence intervals	MODEL	CLODDS=WALD
CLParmPL	Parameter estimates and profile-likelihood confidence intervals	MODEL	CLPARM=PL
CLParmWald	Parameter estimates and Wald confidence intervals	MODEL	CLPARM=WALD
ContrastCoeff	L matrix from CONTRAST	CONTRAST	E
ContrastEstimate	Estimates from CONTRAST	CONTRAST	ESTIMATE=
ContrastTest	Wald test for CONTRAST	CONTRAST	Default
ConvergenceStatus	Convergence status	MODEL	Default
CorrB	Estimated correlation matrix of parameter estimators	MODEL	CORRB
CovB	Estimated covariance matrix of parameter estimators	MODEL	COVB

Table 79.18 *continued*

ODS Table Name	Description	Statements	Options
CumulativeModelTest	Test of the cumulative model assumption	MODEL	Default with cumulative response
EffectInModel	Test effects in model	MODEL	SELECTION=S | B
EffectNotInModel	Test for effects not in model	MODEL	SELECTION=S | F
ExactOddsRatio	Exact odds ratios	EXACT	ESTIMATE=ODDS, ESTIMATE=BOTH
ExactParmEst	Parameter estimates	EXACT	ESTIMATE, ESTIMATE=PARM, ESTIMATE=BOTH
ExactTests	Conditional exact tests	EXACT	Default
FastElimination	Fast backward elimination	MODEL	SELECTION=B, FAST
FitStatistics	Model fit statistics	MODEL	Default
GlobalScore	Global score test	MODEL	NOFIT
GlobalTests	Test for global null hypothesis	MODEL	Default
GoodnessOfFit	Pearson and deviance goodness-of-fit tests	MODEL	SCALE, GOF
IndexPlots	Batch capture of the index plots	MODEL	IPLOTS
Influence	Regression diagnostics	MODEL	INFLUENCE
IterHistory	Iteration history	MODEL	ITPRINT
LackFitChiSq	Hosmer-Lemeshow chi-square test results	MODEL	LACKFIT, GOF
LackFitPartition	Partition for the Hosmer-Lemeshow test	MODEL	LACKFIT, GOF
LastGradient	Last evaluation of gradient	MODEL	ITPRINT
Linear	Linear combination	PROC LOGISTIC	Default
LogLikeChange	Final change in the log likelihood	MODEL	ITPRINT
ModelANOVA	Joint or Type 3 tests of effects	MODEL	Default (with CLASS variables)
ModelBuildingSummary	Summary of model building	MODEL	SELECTION=B | F | S
ModelInfo	Model information	PROC LOGISTIC	Default
NObs	Number of observations	PROC LOGISTIC	Default
OddsEst	Adjusted odds ratios	UNITS	Default
OddsRatios	Odds ratio estimates	MODEL	Default
OddsRatiosPL	Odds ratio estimates and PL confidence intervals	ODDSRATIOS	CL=PL
OddsRatiosWald	Odds ratio estimates and Wald confidence intervals	ODDSRATIOS	CL=WALD
ParameterEstimates	Maximum likelihood estimates of model parameters	MODEL	Default
ResidualChiSq	Residual chi-square	MODEL	SELECTION=F | B

Table 79.18 *continued*

ODS Table Name	Description	Statements	Options
ResponseProfile	Response profile	PROC LOGISTIC	Default
ROCAssociation	Association table for ROC models	ROC MODEL	Default ROCCI
ROCContrastCoeff	L matrix from ROCCONTRAST	ROCCONTRAST	E
ROCContrastCov	Covariance of ROCCONTRAST rows	ROCCONTRAST	COV
ROCContrastEstimate	Estimates from ROCCONTRAST	ROCCONTRAST	ESTIMATE=
ROCContrastTest	Wald test from ROCCONTRAST	ROCCONTRAST	Default
ROCCov	Covariance between ROC curves	ROCCONTRAST	COV
RSquare	R-square	MODEL	RSQUARE
ScoreFitStat	Fit statistics for scored data	SCORE	FITSTAT
SimpleStatistics	Summary statistics for explanatory variables	PROC LOGISTIC	SIMPLE
StrataInfo	Event and nonevent frequencies for each stratum	STRATA	INFO
StrataSummary	Number of strata with specific response frequencies	STRATA	Default
SuffStats	Sufficient statistics	EXACT	OUTDIST=
TestPrint1	**L**[Cov(**b**)]**L**' and **Lb**-**c**	TEST	PRINT
TestPrint2	Ginv(**L**[Cov(**b**)]**L**') and Ginv(**L**[Cov(**b**)]**L**')(**Lb**-**c**)	TEST	PRINT
TestStmts	Linear hypotheses testing results	TEST	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You must also specify the *options* in the PROC LOGISTIC statement that are indicated in Table 79.19.

When ODS Graphics is enabled, then the EFFECT, EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements can produce plots that are associated with their analyses. For information about these plots, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

PROC LOGISTIC assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 79.19.

Table 79.19 Graphs Produced by PROC LOGISTIC

ODS Graph Name	Plot Description	Statement or Option
CalibrationPlot	Panel of calibration plots for polytomous response	PLOTS=CALIBRATION or MODEL / GOF
	Calibration plot for a single polytomous response level	PLOTS=CALIBRATION(UNPACK)
	Calibration plot for binary response	PLOTS=CALIBRATION or MODEL / GOF
DfBetasPlot	Panel of dfbetas by case number	PLOTS=DFBETAS or MODEL / INFLUENCE or IPLOTS
	Effect dfbetas by case number	PLOTS=DFBETAS(UNPACK)
DPCPlot	Difchisq and/or difdev by predicted probability by CI displacement C	PLOTS=DPC
EffectPlot	Predicted probability	PLOTS=EFFECT
InfluencePlots	Panel of influence statistics by case number	PLOTS=INFLUENCE or MODEL / INFLUENCE or IPLOTS
CBarPlot	CI displacement Cbar by case number	PLOTS=INFLUENCE(UNPACK)
CPlot	CI displacement C by case number	PLOTS=INFLUENCE(UNPACK)
DevianceResidualPlot	Deviance residual by case number	PLOTS=INFLUENCE(UNPACK)
DifChisqPlot	Difchisq by case number	PLOTS=INFLUENCE(UNPACK)
DifDeviancePlot	Difdev by case number	PLOTS=INFLUENCE(UNPACK)
LeveragePlot	Hat diagonal by case number	PLOTS=INFLUENCE(UNPACK)
LikelihoodResidualPlot	Likelihood residual by case number	PLOTS=INFLUENCE(UNPACK STDRES)
PearsonResidualPlot	Pearson chi-square residual by case number	PLOTS=INFLUENCE(UNPACK)
StdDevianceResidualPlot	Standardized deviance residual by case number	PLOTS=INFLUENCE(UNPACK STDRES)
StdPearsonResidualPlot	Standardized Pearson chi-square residual by case number	PLOTS=INFLUENCE(UNPACK STDRES)

Table 79.19 *continued*

ODS Graph Name	Plot Description	Statement or Option
LeveragePlots	Panel of influence statistics by leverage	PLOTS=LEVERAGE
LeverageCPlot	CI displacement C by leverage	PLOTS=LEVERAGE(UNPACK)
LeverageDifChisqPlot	Difchisq by leverage	PLOTS=LEVERAGE(UNPACK)
LeverageDifDevPlot	Difdev by leverage	PLOTS=LEVERAGE(UNPACK)
LeveragePhatPlot	Predicted probability by leverage	PLOTS=LEVERAGE(UNPACK)
ORPlot	Odds ratios	Default or PLOTS=ODDSRATIO and MODEL / CLODDS= or ODDSRATIO
PhatPlots	Panel of influence by predicted probability	PLOTS=PHAT
PhatCPlot	CI displacement C by predicted probability	PLOTS=PHAT(UNPACK)
PhatDifChisqPlot	Difchisq by predicted probability	PLOTS=PHAT(UNPACK)
PhatDifDevPlot	Difdev by predicted probability	PLOTS=PHAT(UNPACK)
PhatLeveragePlot	Leverage by predicted probability	PLOTS=PHAT(UNPACK)
PRCurve	Precision-recall curve	PLOTS=PR
PROverlay	PR curves for comparisons	PLOTS=PR and MODEL / SELECTION= or ROC
ROCCurve	Receiver operating characteristics curve	PLOTS=ROC or MODEL / OUTROC= or SCORE OUTROC= or ROC
ROCOverlay	ROC curves for comparisons	PLOTS=ROC and MODEL / SELECTION= or ROC

Examples: LOGISTIC Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 79.1: Stepwise Logistic Regression and Predicted Values

(View the complete code for this example (logiex1.sas) in the example repository.)

Consider a study on cancer remission (Lee 1974). The data consist of patient characteristics and whether or not cancer remission occurred. The following DATA step creates the data set Remission containing seven variables. The variable remiss is the cancer remission indicator variable with a value of 1 for remission and a value of 0 for nonremission. The other six variables are the risk factors thought to be related to cancer remission.

```
data Remission;
   input remiss cell smear infil li blast temp;
   label remiss='Complete Remission';
   datalines;
1   .8   .83  .66  1.9  1.1     .996
1   .9   .36  .32  1.4   .74    .992
0   .8   .88  .7    .8   .176   .982
0  1     .87  .87   .7  1.053   .986
1   .9   .75  .68  1.3   .519   .98
0  1     .65  .65   .6   .519   .982
1   .95  .97  .92  1    1.23    .992
0   .95  .87  .83  1.9  1.354  1.02
0  1     .45  .45   .8   .322   .999
0   .95  .36  .34   .5  0      1.038
0   .85  .39  .33   .7   .279   .988
0   .7   .76  .53  1.2   .146   .982
0   .8   .46  .37   .4   .38   1.006
0   .2   .39  .08   .8   .114   .99
0  1     .9   .9   1.1  1.037   .99
1  1     .84  .84  1.9  2.064  1.02
0   .65  .42  .27   .5   .114  1.014
0  1     .75  .75  1    1.322  1.004
0   .5   .44  .22   .6   .114   .99
1  1     .63  .63  1.1  1.072   .986
0  1     .33  .33   .4   .176  1.01
0   .9   .93  .84   .6  1.591  1.02
1  1     .58  .58  1     .531  1.002
0   .95  .32  .3   1.6   .886   .988
1  1     .6   .6   1.7   .964   .99
1  1     .69  .69   .9   .398   .986
0  1     .73  .73   .7   .398   .986
;
```

The following invocation of PROC LOGISTIC illustrates the use of stepwise selection to identify the prognostic factors for cancer remission. A significance level of 0.3 is required to allow a variable into

the model (SLENTRY=0.3), and a significance level of 0.35 is required for a variable to stay in the model (SLSTAY=0.35). A detailed account of the variable selection process is requested by specifying the DETAILS option. The Hosmer and Lemeshow goodness-of-fit test for the final selected model is requested by specifying the LACKFIT option. The OUTEST= and COVOUT options in the PROC LOGISTIC statement create a data set that contains parameter estimates and their covariances for the final selected model. The response variable option EVENT= chooses remiss=1 (remission) as the event so that the probability of remission is modeled. The OUTPUT statement creates a data set that contains the cumulative predicted probabilities and the corresponding confidence limits, and the individual and cross validated predicted probabilities for each observation. The ODS OUTPUT statement writes the "Association" table from each selection step to a SAS data set.

```
title 'Stepwise Regression on Cancer Remission Data';
proc logistic data=Remission outest=betas covout;
   model remiss(event='1')=cell smear infil li blast temp
                / selection=stepwise
                  slentry=0.3
                  slstay=0.35
                  details
                  lackfit;
   output out=pred p=phat lower=lcl upper=ucl
          predprob=(individual crossvalidate);
   ods output Association=Association;
run;

proc print data=betas;
   title2 'Parameter Estimates and Covariance Matrix';
run;

proc print data=pred;
   title2 'Predicted Probabilities and 95% Confidence Limits';
run;
```

In stepwise selection, an attempt is made to remove any insignificant variables from the model before adding a significant variable to the model. Each addition or deletion of a variable to or from a model is listed as a separate step in the displayed output, and at each step a new model is fitted. Details of the model selection steps are shown in Outputs 79.1.1 through 79.1.5.

Prior to the first step, the intercept-only model is fit and individual score statistics for the potential variables are evaluated (Output 79.1.1).

Output 79.1.1 Startup Model

Stepwise Regression on Cancer Remission Data

The LOGISTIC Procedure

Step 0. Intercept entered:

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

-2 Log L	= 34.372

Output 79.1.1 *continued*

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-0.6931	0.4082	2.8827	0.0895

Residual Chi-Square Test		
Chi-Square	DF	Pr > ChiSq
9.4609	6	0.1493

Analysis of Effects Eligible for Entry			
Effect	DF	Score Chi-Square	Pr > ChiSq
cell	1	1.8893	0.1693
smear	1	1.0745	0.2999
infil	1	1.8817	0.1701
li	1	7.9311	0.0049
blast	1	3.5258	0.0604
temp	1	0.6591	0.4169

In Step 1 (Output 79.1.2), the variable li is selected into the model because it is the most significant variable among those to be chosen ($p = 0.0049 < 0.3$). The intermediate model that contains an intercept and li is then fitted. li remains significant ($p = 0.0146 < 0.35$) and is not removed.

Output 79.1.2 Step 1 of the Stepwise Analysis

Step 1. Effect li entered:

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	36.372	30.073
SC	37.668	32.665
-2 Log L	34.372	26.073

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	8.2988	1	0.0040
Score	7.9311	1	0.0049
Wald	5.9594	1	0.0146

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-3.7771	1.3786	7.5064	0.0061
li	1	2.8973	1.1868	5.9594	0.0146

Output 79.1.2 *continued*

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
li	18.124	1.770	185.563

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	84.0	Somers' D	0.710
Percent Discordant	13.0	Gamma	0.732
Percent Tied	3.1	Tau-a	0.328
Pairs	162	c	0.855

Residual Chi-Square Test		
Chi-Square	DF	Pr > ChiSq
3.1174	5	0.6819

Analysis of Effects Eligible for Removal			
Effect	DF	Wald Chi-Square	Pr > ChiSq
li	1	5.9594	0.0146

Note: No effects for the model in Step 1 are removed.

Analysis of Effects Eligible for Entry			
Effect	DF	Score Chi-Square	Pr > ChiSq
cell	1	1.1183	0.2903
smear	1	0.1369	0.7114
infil	1	0.5715	0.4497
blast	1	0.0932	0.7601
temp	1	1.2591	0.2618

In Step 2 (Output 79.1.3), the variable temp is added to the model. The model then contains an intercept and the variables li and temp. Both li and temp remain significant at 0.35 level; therefore, neither li nor temp is removed from the model.

Output 79.1.3 Step 2 of the Stepwise Analysis

Step 2. Effect temp entered:

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	36.372	30.648
SC	37.668	34.535
-2 Log L	34.372	24.648

Output 79.1.3 *continued*

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	9.7239	2	0.0077
Score	8.3648	2	0.0153
Wald	5.9052	2	0.0522

Analysis of Maximum Likelihood Estimates					
Parameter	**DF**	**Estimate**	**Standard Error**	**Wald Chi-Square**	**Pr > ChiSq**
Intercept	1	47.8448	46.4381	1.0615	0.3029
li	1	3.3017	1.3593	5.9002	0.0151
temp	1	-52.4214	47.4897	1.2185	0.2697

Odds Ratio Estimates			
Effect	**Point Estimate**	**95% Wald Confidence Limits**	
li	27.158	1.892	389.856
temp	<0.001	<0.001	>999.999

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	87.0	**Somers' D**	0.747
Percent Discordant	12.3	**Gamma**	0.752
Percent Tied	0.6	**Tau-a**	0.345
Pairs	162	**c**	0.873

Residual Chi-Square Test		
Chi-Square	**DF**	**Pr > ChiSq**
2.1429	4	0.7095

Analysis of Effects Eligible for Removal			
Effect	**DF**	**Wald Chi-Square**	**Pr > ChiSq**
li	1	5.9002	0.0151
temp	1	1.2185	0.2697

Note: No effects for the model in Step 2 are removed.

Analysis of Effects Eligible for Entry			
Effect	**DF**	**Score Chi-Square**	**Pr > ChiSq**
cell	1	1.4700	0.2254
smear	1	0.1730	0.6775
infil	1	0.8274	0.3630
blast	1	1.1013	0.2940

In Step 3 (Output 79.1.4), the variable cell is added to the model. The model then contains an intercept and the variables li, temp, and cell. None of these variables are removed from the model because all are significant at the 0.35 level.

Output 79.1.4 Step 3 of the Stepwise Analysis

Step 3. Effect cell entered:

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	36.372	29.953
SC	37.668	35.137
-2 Log L	34.372	21.953

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	12.4184	3	0.0061
Score	9.2502	3	0.0261
Wald	4.8281	3	0.1848

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	67.6339	56.8875	1.4135	0.2345
cell	1	9.6521	7.7511	1.5507	0.2130
li	1	3.8671	1.7783	4.7290	0.0297
temp	1	-82.0737	61.7124	1.7687	0.1835

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
cell	>999.999	0.004	>999.999
li	47.804	1.465	>999.999
temp	<0.001	<0.001	>999.999

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	88.9	Somers' D	0.778
Percent Discordant	11.1	Gamma	0.778
Percent Tied	0.0	Tau-a	0.359
Pairs	162	c	0.889

Residual Chi-Square Test		
Chi-Square	DF	Pr > ChiSq
0.1831	3	0.9803

Output 79.1.4 *continued*

Analysis of Effects Eligible for Removal			
Effect	DF	Wald Chi-Square	Pr > ChiSq
cell	1	1.5507	0.2130
li	1	4.7290	0.0297
temp	1	1.7687	0.1835

Note: No effects for the model in Step 3 are removed.

Analysis of Effects Eligible for Entry			
Effect	DF	Score Chi-Square	Pr > ChiSq
smear	1	0.0956	0.7572
infil	1	0.0844	0.7714
blast	1	0.0208	0.8852

Finally, none of the remaining variables outside the model meet the entry criterion, and the stepwise selection is terminated. A summary of the stepwise selection is displayed in Output 79.1.5.

Output 79.1.5 Summary of the Stepwise Selection

Summary of Stepwise Selection							
	Effect						
Step	Entered	Removed	DF	Number In	Score Chi-Square	Wald Chi-Square	Pr > ChiSq
1	li		1	1	7.9311		0.0049
2	temp		1	2	1.2591		0.2618
3	cell		1	3	1.4700		0.2254

Results of the Hosmer and Lemeshow test are shown in Output 79.1.6. There is no evidence of a lack of fit in the selected model ($p = 0.5054$).

Output 79.1.6 Display of the LACKFIT Option

Partition for the Hosmer and Lemeshow Test					
		remiss = 1		remiss = 0	
Group	Total	Observed	Expected	Observed	Expected
1	3	0	0.00	3	3.00
2	3	0	0.01	3	2.99
3	3	0	0.19	3	2.81
4	3	0	0.56	3	2.44
5	4	1	1.09	3	2.91
6	3	2	1.35	1	1.65
7	3	2	1.84	1	1.16
8	3	3	2.15	0	0.85
9	2	1	1.80	1	0.20

Output 79.1.6 *continued*

Hosmer and Lemeshow Goodness-of-Fit Test		
Chi-Square	DF	Pr > ChiSq
6.2983	7	0.5054

The data set betas created by the OUTEST= and COVOUT options is displayed in Output 79.1.7. The data set contains parameter estimates and the covariance matrix for the final selected model. Note that all explanatory variables listed in the MODEL statement are included in this data set; however, variables that are not included in the final model have all missing values.

Output 79.1.7 Data Set of Estimates and Covariances

Stepwise Regression on Cancer Remission Data
Parameter Estimates and Covariance Matrix

Obs	_LINK_	_TYPE_	_STATUS_	_NAME_	Intercept	cell	smear	infil	li	blast	temp	_LNLIKE_	_ESTTYPE_
1	LOGIT	PARMS	0 Converged	remiss	67.63	9.652	.	.	3.8671	.	-82.07	-10.9767	MLE
2	LOGIT	COV	0 Converged	Intercept	3236.19	157.097	.	.	64.5726	.	-3483.23	-10.9767	MLE
3	LOGIT	COV	0 Converged	cell	157.10	60.079	.	.	6.9454	.	-223.67	-10.9767	MLE
4	LOGIT	COV	0 Converged	smear	.	.	.	.	.	.	.	-10.9767	MLE
5	LOGIT	COV	0 Converged	infil	.	.	.	.	.	.	.	-10.9767	MLE
6	LOGIT	COV	0 Converged	li	64.57	6.945	.	.	3.1623	.	-75.35	-10.9767	MLE
7	LOGIT	COV	0 Converged	blast	.	.	.	.	.	.	.	-10.9767	MLE
8	LOGIT	COV	0 Converged	temp	-3483.23	-223.669	.	.	-75.3513	.	3808.42	-10.9767	MLE

The data set pred created by the OUTPUT statement is displayed in Output 79.1.8. It contains all the variables in the input data set, the variable phat for the (cumulative) predicted probability, the variables lcl and ucl for the lower and upper confidence limits for the probability, and four other variables (IP_1, IP_0, XP_1, and XP_0) for the PREDPROBS= option. The data set also contains the variable _LEVEL_, indicating the response value to which phat, lcl, and ucl refer. For instance, for the first row of the OUTPUT data set, the values of _LEVEL_ and phat, lcl, and ucl are 1, 0.72265, 0.16892, and 0.97093, respectively; this means that the estimated probability that remiss=1 is 0.723 for the given explanatory variable values, and the corresponding 95% confidence interval is (0.16892, 0.97093). The variables IP_1 and IP_0 contain the predicted probabilities that remiss=1 and remiss=0, respectively. Note that values of phat and IP_1 are identical because they both contain the probabilities that remiss=1. The variables XP_1 and XP_0 contain the cross validated predicted probabilities that remiss=1 and remiss=0, respectively.

Output 79.1.8 Predicted Probabilities and Confidence Intervals

Stepwise Regression on Cancer Remission Data
Predicted Probabilities and 95% Confidence Limits

Obs	remiss	cell	smear	infil	li	blast	temp	_FROM_	_INTO_	IP_0	IP_1	XP_0	XP_1	_LEVEL_	phat	lcl	ucl
1	1	0.80	0.83	0.66	1.9	1.100	0.996	1	1	0.27735	0.72265	0.43873	0.56127	1	0.72265	0.16892	0.97093
2	1	0.90	0.36	0.32	1.4	0.740	0.992	1	1	0.42126	0.57874	0.47461	0.52539	1	0.57874	0.26788	0.83762
3	0	0.80	0.88	0.70	0.8	0.176	0.982	0	0	0.89540	0.10460	0.87060	0.12940	1	0.10460	0.00781	0.63419
4	0	1.00	0.87	0.87	0.7	1.053	0.986	0	0	0.71742	0.28258	0.67259	0.32741	1	0.28258	0.07498	0.65683
5	1	0.90	0.75	0.68	1.3	0.519	0.980	1	1	0.28582	0.71418	0.36901	0.63099	1	0.71418	0.25218	0.94876
6	0	1.00	0.65	0.65	0.6	0.519	0.982	0	0	0.72911	0.27089	0.67269	0.32731	1	0.27089	0.05852	0.68951
7	1	0.95	0.97	0.92	1.0	1.230	0.992	1	0	0.67844	0.32156	0.72923	0.27077	1	0.32156	0.13255	0.59516
8	0	0.95	0.87	0.83	1.9	1.354	1.020	0	1	0.39277	0.60723	0.09906	0.90094	1	0.60723	0.10572	0.95287
9	0	1.00	0.45	0.45	0.8	0.322	0.999	0	0	0.83368	0.16632	0.80864	0.19136	1	0.16632	0.03018	0.56123
10	0	0.95	0.36	0.34	0.5	0.000	1.038	0	0	0.99843	0.00157	0.99840	0.00160	1	0.00157	0.00000	0.68962
11	0	0.85	0.39	0.33	0.7	0.279	0.988	0	0	0.92715	0.07285	0.91723	0.08277	1	0.07285	0.00614	0.49982
12	0	0.70	0.76	0.53	1.2	0.146	0.982	0	0	0.82714	0.17286	0.63838	0.36162	1	0.17286	0.00637	0.87206
13	0	0.80	0.46	0.37	0.4	0.380	1.006	0	0	0.99654	0.00346	0.99644	0.00356	1	0.00346	0.00001	0.46530
14	0	0.20	0.39	0.08	0.8	0.114	0.990	0	0	0.99982	0.00018	0.99981	0.00019	1	0.00018	0.00000	0.96482
15	0	1.00	0.90	0.90	1.1	1.037	0.990	0	1	0.42878	0.57122	0.35354	0.64646	1	0.57122	0.25303	0.83973
16	1	1.00	0.84	0.84	1.9	2.064	1.020	1	1	0.28530	0.71470	0.47213	0.52787	1	0.71470	0.15362	0.97189
17	0	0.65	0.42	0.27	0.5	0.114	1.014	0	0	0.99938	0.00062	0.99937	0.00063	1	0.00062	0.00000	0.62665
18	0	1.00	0.75	0.75	1.0	1.322	1.004	0	0	0.77711	0.22289	0.73612	0.26388	1	0.22289	0.04483	0.63670
19	0	0.50	0.44	0.22	0.6	0.114	0.990	0	0	0.99846	0.00154	0.99842	0.00158	1	0.00154	0.00000	0.79644
20	1	1.00	0.63	0.63	1.1	1.072	0.986	1	1	0.35089	0.64911	0.42053	0.57947	1	0.64911	0.26305	0.90555
21	0	1.00	0.33	0.33	0.4	0.176	1.010	0	0	0.98307	0.01693	0.98170	0.01830	1	0.01693	0.00029	0.50475
22	0	0.90	0.93	0.84	0.6	1.591	1.020	0	0	0.99378	0.00622	0.99348	0.00652	1	0.00622	0.00003	0.56062
23	1	1.00	0.58	0.58	1.0	0.531	1.002	1	0	0.74739	0.25261	0.84423	0.15577	1	0.25261	0.06137	0.63597
24	0	0.95	0.32	0.30	1.6	0.886	0.988	0	1	0.12989	0.87011	0.03637	0.96363	1	0.87011	0.40910	0.98481
25	1	1.00	0.60	0.60	1.7	0.964	0.990	1	1	0.06868	0.93132	0.08017	0.91983	1	0.93132	0.44114	0.99573
26	1	1.00	0.69	0.69	0.9	0.398	0.986	1	0	0.53949	0.46051	0.62312	0.37688	1	0.46051	0.16612	0.78529
27	0	1.00	0.73	0.73	0.7	0.398	0.986	0	0	0.71742	0.28258	0.67259	0.32741	1	0.28258	0.07498	0.65683

If you want to order the selected models based on a statistic such as the AIC, R-square, or area under the ROC curve (AUC), you can use the ODS OUTPUT statement to save the appropriate table to a data set and then display the statistic along with the step number. For example, the following program orders the steps according to the "c" statistic from the Association data set:

```
data Association(rename=(Label2=Statistic nValue2=Value));
   set Association;
   if (Label2='c');
   keep Step Label2 nValue2;
proc sort data=Association;
   by Value;
title;
proc print data=Association;
run;
```

The results, displayed in Output 79.1.9, show that the model that has the largest AUC (0.889) is the final model selected by the stepwise method. You can also perform this analysis by using the %SELECT macro

(SAS Institute Inc. 2015).

Output 79.1.9 Selection Steps Ordered by AUC

Obs	Step	Statistic	Value
1	1	c	0.854938
2	2	c	0.873457
3	3	c	0.888889

Next, a different variable selection method is used to select prognostic factors for cancer remission, and an efficient algorithm is employed to eliminate insignificant variables from a model. The following statements invoke PROC LOGISTIC to perform the backward elimination analysis:

```
title 'Backward Elimination on Cancer Remission Data';
proc logistic data=Remission;
   model remiss(event='1')=temp cell li smear blast
         / selection=backward fast slstay=0.2
           ctable pprob=(0 to 1 by 0.05);
run;
```

The backward elimination analysis (SELECTION=BACKWARD) starts with a model that contains all explanatory variables given in the MODEL statement. By specifying the FAST option, PROC LOGISTIC eliminates insignificant variables without refitting the model repeatedly. This analysis uses a significance level of 0.2 to retain variables in the model (SLSTAY=0.2), which is different from the previous stepwise analysis where SLSTAY=.35. The CTABLE option is specified to produce classifications of input observations on the basis of the final selected model, and the PPROB= option lists the cutpoints that are used to create the classification table.

Results of the fast elimination analysis are shown in Output 79.1.10 and Output 79.1.11. Initially, a full model that contains all six risk factors is fit to the data (Output 79.1.10). In the next step (Output 79.1.11), PROC LOGISTIC removes blast, smear, cell, and temp from the model all at once. This leaves li and the intercept as the only variables in the final model. Note that in this analysis, only parameter estimates for the final model are displayed because the DETAILS option has not been specified.

Output 79.1.10 Initial Step in Backward Elimination

Backward Elimination on Cancer Remission Data

The LOGISTIC Procedure

Model Information		
Data Set	WORK.REMISSION	
Response Variable	remiss	Complete Remission
Number of Response Levels	2	
Model	binary logit	
Optimization Technique	Fisher's scoring	

Number of Observations Read	27
Number of Observations Used	27

Output 79.1.10 *continued*

Response Profile		
Ordered Value	remiss	Total Frequency
1	0	18
2	1	9

Probability modeled is remiss=1.

Backward Elimination Procedure

Step 0. The following effects were entered:

Intercept temp cell li smear blast

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	36.372	33.857
SC	37.668	41.632
-2 Log L	34.372	21.857

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	12.5146	5	0.0284
Score	9.3295	5	0.0966
Wald	4.7284	5	0.4499

Output 79.1.11 Fast Elimination Step

Step 1. Fast Backward Elimination:

Analysis of Effects Removed by Fast Backward Elimination						
Effect Removed	Chi-Square	DF	Pr > ChiSq	Residual Chi-Square	DF	Pr > Residual ChiSq
blast	0.0008	1	0.9768	0.0008	1	0.9768
smear	0.0951	1	0.7578	0.0959	2	0.9532
cell	1.5134	1	0.2186	1.6094	3	0.6573
temp	0.6535	1	0.4189	2.2628	4	0.6875

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Output 79.1.11 *continued*

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	36.372	30.073
SC	37.668	32.665
-2 Log L	34.372	26.073

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	8.2988	1	0.0040
Score	7.9311	1	0.0049
Wald	5.9594	1	0.0146

Residual Chi-Square Test		
Chi-Square	DF	Pr > ChiSq
2.8530	4	0.5827

Summary of Backward Elimination					
Step	Effect Removed	DF	Number In	Wald Chi-Square	Pr > ChiSq
1	blast	1	4	0.0008	0.9768
1	smear	1	3	0.0951	0.7578
1	cell	1	2	1.5134	0.2186
1	temp	1	1	0.6535	0.4189

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-3.7771	1.3786	7.5064	0.0061
li	1	2.8973	1.1868	5.9594	0.0146

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
li	18.124	1.770	185.563

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	84.0	Somers' D	0.710
Percent Discordant	13.0	Gamma	0.732
Percent Tied	3.1	Tau-a	0.328
Pairs	162	c	0.855

Note that you can also use the FAST option when SELECTION=STEPWISE. However, the FAST option operates only on backward elimination steps. In this example, the stepwise process only adds variables, so the FAST option would not be useful.

Results of the CTABLE option are shown in Output 79.1.12.

Output 79.1.12 Classifying Input Observations

Classification Table									
	Correct		Incorrect		Percentages				
Prob Level	Event	Non-Event	Event	Non-Event	Correct	Sensi-tivity	Speci-ficity	Pos Pred	Neg Pred
0.000	9	0	18	0	33.3	100.0	0.0	33.3	.
0.050	9	0	18	0	33.3	100.0	0.0	33.3	.
0.100	9	4	14	0	48.1	100.0	22.2	39.1	100.0
0.150	9	7	11	0	59.3	100.0	38.9	45.0	100.0
0.200	8	13	5	1	77.8	88.9	72.2	61.5	92.9
0.250	8	13	5	1	77.8	88.9	72.2	61.5	92.9
0.300	6	13	5	3	70.4	66.7	72.2	54.5	81.3
0.350	5	14	4	4	70.4	55.6	77.8	55.6	77.8
0.400	5	15	3	4	74.1	55.6	83.3	62.5	78.9
0.450	5	15	3	4	74.1	55.6	83.3	62.5	78.9
0.500	4	16	2	5	74.1	44.4	88.9	66.7	76.2
0.550	3	16	2	6	70.4	33.3	88.9	60.0	72.7
0.600	3	16	2	6	70.4	33.3	88.9	60.0	72.7
0.650	3	16	2	6	70.4	33.3	88.9	60.0	72.7
0.700	3	16	2	6	70.4	33.3	88.9	60.0	72.7
0.750	2	16	2	7	66.7	22.2	88.9	50.0	69.6
0.800	2	17	1	7	70.4	22.2	94.4	66.7	70.8
0.850	0	17	1	9	63.0	0.0	94.4	0.0	65.4
0.900	0	17	1	9	63.0	0.0	94.4	0.0	65.4
0.950	0	18	0	9	66.7	0.0	100.0	.	66.7
1.000	0	18	0	9	66.7	0.0	100.0	.	66.7

Each row of the "Classification Table" corresponds to a cutpoint applied to the predicted probabilities, which is given in the Prob Level column. The 2×2 frequency tables of observed and predicted responses are given by the next four columns. For example, with a cutpoint of 0.5, 4 events and 16 nonevents were classified correctly. On the other hand, 2 nonevents were incorrectly classified as events and 5 events were incorrectly classified as nonevents. For this cutpoint, the correct classification rate is 20/27 (=74.1%), which is given in the sixth column. Accuracy of the classification is summarized by the sensitivity, specificity, and positive and negative predictive values, which are displayed in the last four columns. You can control the number of cutpoints used, and their values, by using the PPROB= option.

Example 79.2: Logistic Modeling with Categorical Predictors

(View the complete code for this example (logiex2.sas) in the example repository.)

Consider a study of the analgesic effects of treatments on elderly patients with neuralgia. Two test treatments and a placebo are compared. The response variable is whether the patient reported pain or not. Researchers recorded the age and gender of 60 patients and the duration of complaint before the treatment began. The following DATA step creates the data set Neuralgia:

```
data Neuralgia;
   input Treatment $ Sex $ Age Duration Pain $ @@;
   datalines;
P  F  68   1  No   B  M  74  16  No  P  F  67  30  No
P  M  66  26  Yes  B  F  67  28  No  B  F  77  16  No
A  F  71  12  No   B  F  72  50  No  B  F  76   9  Yes
A  M  71  17  Yes  A  F  63  27  No  A  F  69  18  Yes
B  F  66  12  No   A  M  62  42  No  P  F  64   1  Yes
A  F  64  17  No   P  M  74   4  No  A  F  72  25  No
P  M  70   1  Yes  B  M  66  19  No  B  M  59  29  No
A  F  64  30  No   A  M  70  28  No  A  M  69   1  No
B  F  78   1  No   P  M  83   1  Yes B  F  69  42  No
B  M  75  30  Yes  P  M  77  29  Yes P  F  79  20  Yes
A  M  70  12  No   A  F  69  12  No  B  F  65  14  No
B  M  70   1  No   B  M  67  23  No  A  M  76  25  Yes
P  M  78  12  Yes  B  M  77   1  Yes B  F  69  24  No
P  M  66   4  Yes  P  F  65  29  No  P  M  60  26  Yes
A  M  78  15  Yes  B  M  75  21  Yes A  F  67  11  No
P  F  72  27  No   P  F  70  13  Yes A  M  75   6  Yes
B  F  65   7  No   P  F  68  27  Yes P  M  68  11  Yes
P  M  67  17  Yes  B  M  70  22  No  A  M  65  15  No
P  F  67   1  Yes  A  M  67  10  No  P  F  72  11  Yes
A  F  74   1  No   B  M  80  21  Yes A  F  69   3  No
;
```

The data set Neuralgia contains five variables: Treatment, Sex, Age, Duration, and Pain. The last variable, Pain, is the response variable. A specification of Pain=Yes indicates there was pain, and Pain=No indicates no pain. The variable Treatment is a categorical variable with three levels: A and B represent the two test treatments, and P represents the placebo treatment. The gender of the patients is given by the categorical variable Sex. The variable Age is the age of the patients, in years, when treatment began. The duration of complaint, in months, before the treatment began is given by the variable Duration.

The following statements use the LOGISTIC procedure to fit a two-way logit with interaction model for the effect of Treatment and Sex, with Age and Duration as covariates. The categorical variables Treatment and Sex are declared in the CLASS statement.

```
proc logistic data=Neuralgia;
   class Treatment Sex;
   model Pain= Treatment Sex Treatment*Sex Age Duration / expb;
run;
```

In this analysis, PROC LOGISTIC models the probability of no pain (Pain=No). By default, effect coding is used to represent the CLASS variables. Two design variables are created for Treatment and one for Sex, as shown in Output 79.2.1.

Output 79.2.1 Effect Coding of CLASS Variables

The LOGISTIC Procedure

Class Level Information			
Class	**Value**	**Design Variables**	
Treatment	**A**	1	0
	B	0	1
	P	-1	-1
Sex	**F**	1	
	M	-1	

PROC LOGISTIC displays a table of the Type 3 analysis of effects based on the Wald test (Output 79.2.2). Note that the Treatment*Sex interaction and the duration of complaint are not statistically significant (p = 0.9318 and p = 0.8752, respectively). This indicates that there is no evidence that the treatments affect pain differently in men and women, and no evidence that the pain outcome is related to the duration of pain.

Output 79.2.2 Wald Tests of Individual Effects

Joint Tests			
Effect	**DF**	**Wald Chi-Square**	**Pr > ChiSq**
Treatment	2	11.9886	0.0025
Sex	1	5.3104	0.0212
Treatment*Sex	2	0.1412	0.9318
Age	1	7.2744	0.0070
Duration	1	0.0247	0.8752

Note: Under full-rank parameterizations, Type 3 effect tests are replaced by joint tests. The joint test for an effect is a test that all the parameters associated with that effect are zero. Such joint tests might not be equivalent to Type 3 effect tests under GLM parameterization.

Parameter estimates are displayed in Output 79.2.3. The Exp(Est) column contains the exponentiated parameter estimates requested with the EXPB option. These values can, but do not necessarily, represent odds ratios for the corresponding variables. For continuous explanatory variables, the Exp(Est) value corresponds to the odds ratio for a unit increase of the corresponding variable. For CLASS variables that use effect coding, the Exp(Est) values have no direct interpretation as a comparison of levels. However, when the reference coding is used, the Exp(Est) values represent the odds ratio between the corresponding level and the reference level. Following the parameter estimates table, PROC LOGISTIC displays the odds ratio estimates for those variables that are not involved in any interaction terms. If the variable is a CLASS variable, the odds ratio estimate comparing each level with the reference level is computed regardless of the coding scheme. In this analysis, because the model contains the Treatment*Sex interaction term, the odds ratios for Treatment and Sex were not computed. The odds ratio estimates for Age and Duration are precisely the values given in the Exp(Est) column in the parameter estimates table.

Output 79.2.3 Parameter Estimates with Effect Coding

Analysis of Maximum Likelihood Estimates							
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq	Exp(Est)
Intercept		1	19.2236	7.1315	7.2661	0.0070	2.232E8
Treatment	A	1	0.8483	0.5502	2.3773	0.1231	2.336
Treatment	B	1	1.4949	0.6622	5.0956	0.0240	4.459
Sex	F	1	0.9173	0.3981	5.3104	0.0212	2.503
Treatment*Sex	A F	1	-0.2010	0.5568	0.1304	0.7180	0.818
Treatment*Sex	B F	1	0.0487	0.5563	0.0077	0.9302	1.050
Age		1	-0.2688	0.0996	7.2744	0.0070	0.764
Duration		1	0.00523	0.0333	0.0247	0.8752	1.005

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
Age	0.764	0.629	0.929
Duration	1.005	0.942	1.073

The following PROC LOGISTIC statements illustrate the use of forward selection on the data set Neuralgia to identify the effects that differentiate the two Pain responses. The option SELECTION=FORWARD is specified to carry out the forward selection. The term Treatment|Sex@2 illustrates another way to specify main effects and two-way interactions. (Note that, in this case, the "@2" is unnecessary because no interactions besides the two-way interaction are possible).

```
proc logistic data=Neuralgia;
   class Treatment Sex;
   model Pain=Treatment|Sex@2 Age Duration
         /selection=forward expb;
run;
```

Results of the forward selection process are summarized in Output 79.2.4. The variable Treatment is selected first, followed by Age and then Sex. The results are consistent with the previous analysis (Output 79.2.2) in which the Treatment*Sex interaction and Duration are not statistically significant.

Output 79.2.4 Effects Selected into the Model

The LOGISTIC Procedure

Summary of Forward Selection					
Step	Effect Entered	DF	Number In	Score Chi-Square	Pr > ChiSq
1	Treatment	2	1	13.7143	0.0011
2	Age	1	2	10.6038	0.0011
3	Sex	1	3	5.9959	0.0143

Output 79.2.5 shows the Type 3 analysis of effects, the parameter estimates, and the odds ratio estimates for the selected model. All three variables, Treatment, Age, and Sex, are statistically significant at the 0.05 level (p=0.0018, p=0.0213, and p=0.0057, respectively). Because the selected model does not contain the Treatment*Sex interaction, odds ratios for Treatment and Sex are computed. The estimated odds ratio is

24.022 for treatment A versus placebo, 41.528 for Treatment B versus placebo, and 6.194 for female patients versus male patients. Note that these odds ratio estimates are not the same as the corresponding values in the Exp(Est) column in the parameter estimates table because effect coding was used. From Output 79.2.5, it is evident that both Treatment A and Treatment B are better than the placebo in reducing pain; females tend to have better improvement than males; and younger patients are faring better than older patients.

Output 79.2.5 Type 3 Effects and Parameter Estimates with Effect Coding

Type 3 Analysis of Effects

Effect	DF	Wald Chi-Square	Pr > ChiSq
Treatment	2	12.6928	0.0018
Sex	1	5.3013	0.0213
Age	1	7.6314	0.0057

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq	Exp(Est)
Intercept		1	19.0804	6.7882	7.9007	0.0049	1.9343E8
Treatment	A	1	0.8772	0.5274	2.7662	0.0963	2.404
Treatment	B	1	1.4246	0.6036	5.5711	0.0183	4.156
Sex	F	1	0.9118	0.3960	5.3013	0.0213	2.489
Age		1	-0.2650	0.0959	7.6314	0.0057	0.767

Odds Ratio Estimates

Effect		Point Estimate	95% Wald Confidence Limits	
Treatment A vs P		24.022	3.295	175.121
Treatment B vs P		41.528	4.500	383.262
Sex	F vs M	6.194	1.312	29.248
Age		0.767	0.636	0.926

Finally, the following statements refit the previously selected model, except that reference coding is used for the CLASS variables instead of effect coding:

```
ods graphics on;
proc logistic data=Neuralgia plots(only)=(oddsratio(range=clip));
   class Treatment Sex /param=ref;
   model Pain= Treatment Sex Age / noor;
   oddsratio Treatment;
   oddsratio Sex;
   oddsratio Age;
   contrast 'Pairwise A vs P' Treatment 1  0 / estimate=exp;
   contrast 'Pairwise B vs P' Treatment 0  1 / estimate=exp;
   contrast 'Pairwise A vs B' Treatment 1 -1 / estimate=exp;
   contrast 'Female vs Male' Sex 1 / estimate=exp;
   effectplot / at(Sex=all) noobs;
   effectplot slicefit(sliceby=Sex plotby=Treatment) / noobs;
run;
```

The ODDSRATIO statements compute the odds ratios for the covariates, and the NOOR option suppresses the default odds ratio table. Four CONTRAST statements are specified; they provide another method of

producing the odds ratios. The three contrasts labeled 'Pairwise' specify a contrast vector, L, for each of the pairwise comparisons between the three levels of Treatment. The contrast labeled 'Female vs Male' compares female to male patients. The option ESTIMATE=EXP is specified in all CONTRAST statements to exponentiate the estimates of $\mathbf{L}'\boldsymbol{\beta}$. With the given specification of contrast coefficients, the first of the 'Pairwise' CONTRAST statements corresponds to the odds ratio of A versus P, the second corresponds to B versus P, and the third corresponds to A versus B. You can also specify the 'Pairwise' contrasts in a single contrast statement with three rows. The 'Female vs Male' CONTRAST statement corresponds to the odds ratio that compares female to male patients.

The PLOTS(ONLY)= option displays only the requested odds ratio plot when ODS Graphics is enabled. The EFFECTPLOT statements do not honor the ONLY option, and display the fitted model. The first EFFECTPLOT statement by default produces a plot of the predicted values against the continuous Age variable, grouped by the Treatment levels. The AT option produces one plot for males and another for females; the NOOBS option suppresses the display of the observations. In the second EFFECTPLOT statement, a SLICEFIT plot is specified to display the Age variable on the X axis, the fits are grouped by the Sex levels, and the PLOTBY= option produces a panel of plots that displays each level of the Treatment variable.

The reference coding is shown in Output 79.2.6. The Type 3 analysis of effects and the parameter estimates for the reference coding are displayed in Output 79.2.7. Although the parameter estimates are different because of the different parameterizations, the "Type 3 Analysis of Effects" table remains the same as in Output 79.2.5. With effect coding, the treatment A parameter estimate (0.8772) estimates the effect of treatment A compared to the average effect of treatments A, B, and placebo. The treatment A estimate (3.1790) under the reference coding estimates the difference in effect of treatment A and the placebo treatment.

Output 79.2.6 Reference Coding of CLASS Variables

The LOGISTIC Procedure

Class Level Information

Class	Value	Design Variables	
Treatment	A	1	0
	B	0	1
	P	0	0
Sex	F	1	
	M	0	

Output 79.2.7 Type 3 Effects and Parameter Estimates with Reference Coding

Type 3 Analysis of Effects

Effect	DF	Wald Chi-Square	Pr > ChiSq
Treatment	2	12.6928	0.0018
Sex	1	5.3013	0.0213
Age	1	7.6314	0.0057

Output 79.2.7 *continued*

Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	15.8669	6.4056	6.1357	0.0132
Treatment	A	1	3.1790	1.0135	9.8375	0.0017
Treatment	B	1	3.7264	1.1339	10.8006	0.0010
Sex	F	1	1.8235	0.7920	5.3013	0.0213
Age		1	-0.2650	0.0959	7.6314	0.0057

The ODDSRATIO statement results are shown in Output 79.2.8, and the resulting plot is displayed in Output 79.2.9. Note in Output 79.2.9 that the odds ratio confidence limits are truncated due to specifying the RANGE=CLIP option; this enables you to see which intervals contain "1" more clearly. The odds ratios are identical to those shown in the "Odds Ratio Estimates" table in Output 79.2.5 with the addition of the odds ratio for "Treatment A vs B". Both treatments A and B are highly effective over placebo in reducing pain, as can be seen from the odds ratios comparing treatment A against P and treatment B against P (the second and third rows in the table). However, the 95% confidence interval for the odds ratio comparing treatment A to B is (0.0932, 3.5889), indicating that the pain reduction effects of these two test treatments are not very different. Again, the 'Sex F vs M' odds ratio shows that female patients fared better in obtaining relief from pain than male patients. The odds ratio for Age shows that a patient one year older is 0.77 times as likely to show no pain; that is, younger patients have more improvement than older patients.

Output 79.2.8 Results from the ODDSRATIO Statements

Odds Ratio Estimates and Wald Confidence Intervals			
Odds Ratio	Estimate	95% Confidence Limits	
Treatment A vs B	0.578	0.093	3.589
Treatment A vs P	24.022	3.295	175.121
Treatment B vs P	41.528	4.500	383.262
Sex F vs M	6.194	1.312	29.248
Age	0.767	0.636	0.926

Output 79.2.9 Plot of the ODDSRATIO Statement Results

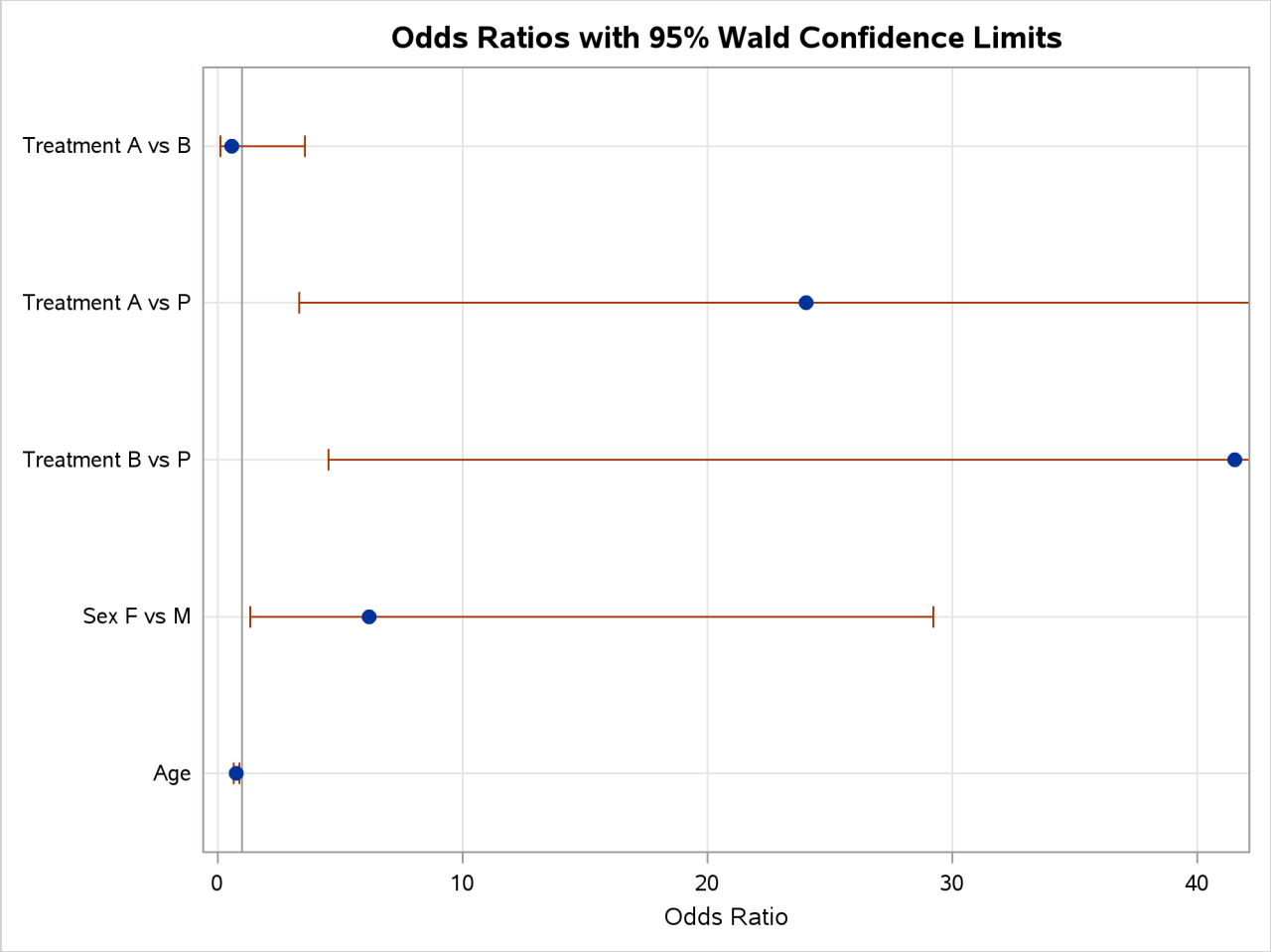

Output 79.2.10 contains two tables: the "Contrast Test Results" table and the "Contrast Estimation and Testing Results by Row" table. The former contains the overall Wald test for each CONTRAST statement. The latter table contains estimates and tests of individual contrast rows. The estimates for the first two rows of the 'Pairwise' CONTRAST statements are the same as those given in the two preceding odds ratio tables (Output 79.2.5 and Output 79.2.8). The third row estimates the odds ratio comparing A to B, agreeing with Output 79.2.8, and the last row computes the odds ratio comparing pain relief for females to that for males.

Output 79.2.10 Results of CONTRAST Statements

Contrast Test Results

Contrast	DF	Wald Chi-Square	Pr > ChiSq
Pairwise A vs P	1	9.8375	0.0017
Pairwise B vs P	1	10.8006	0.0010
Pairwise A vs B	1	0.3455	0.5567
Female vs Male	1	5.3013	0.0213

Output 79.2.10 *continued*

Contrast Estimation and Testing Results by Row									
Contrast	Type	Row	Estimate	Standard Error	Alpha	Confidence Limits		Wald Chi-Square	Pr > ChiSq
Pairwise A vs P	EXP	1	24.0218	24.3473	0.05	3.2951	175.1	9.8375	0.0017
Pairwise B vs P	EXP	1	41.5284	47.0877	0.05	4.4998	383.3	10.8006	0.0010
Pairwise A vs B	EXP	1	0.5784	0.5387	0.05	0.0932	3.5889	0.3455	0.5567
Female vs Male	EXP	1	6.1937	4.9053	0.05	1.3116	29.2476	5.3013	0.0213

ANCOVA-style plots of the model-predicted probabilities against the Age variable for each combination of Treatment and Sex are displayed in Output 79.2.11 and Output 79.2.12. These plots confirm that females always have a higher probability of pain reduction in each treatment group, the placebo treatment has a lower probability of success than the other treatments, and younger patients respond to treatment better than older patients.

Output 79.2.11 Model-Predicted Probabilities by Sex

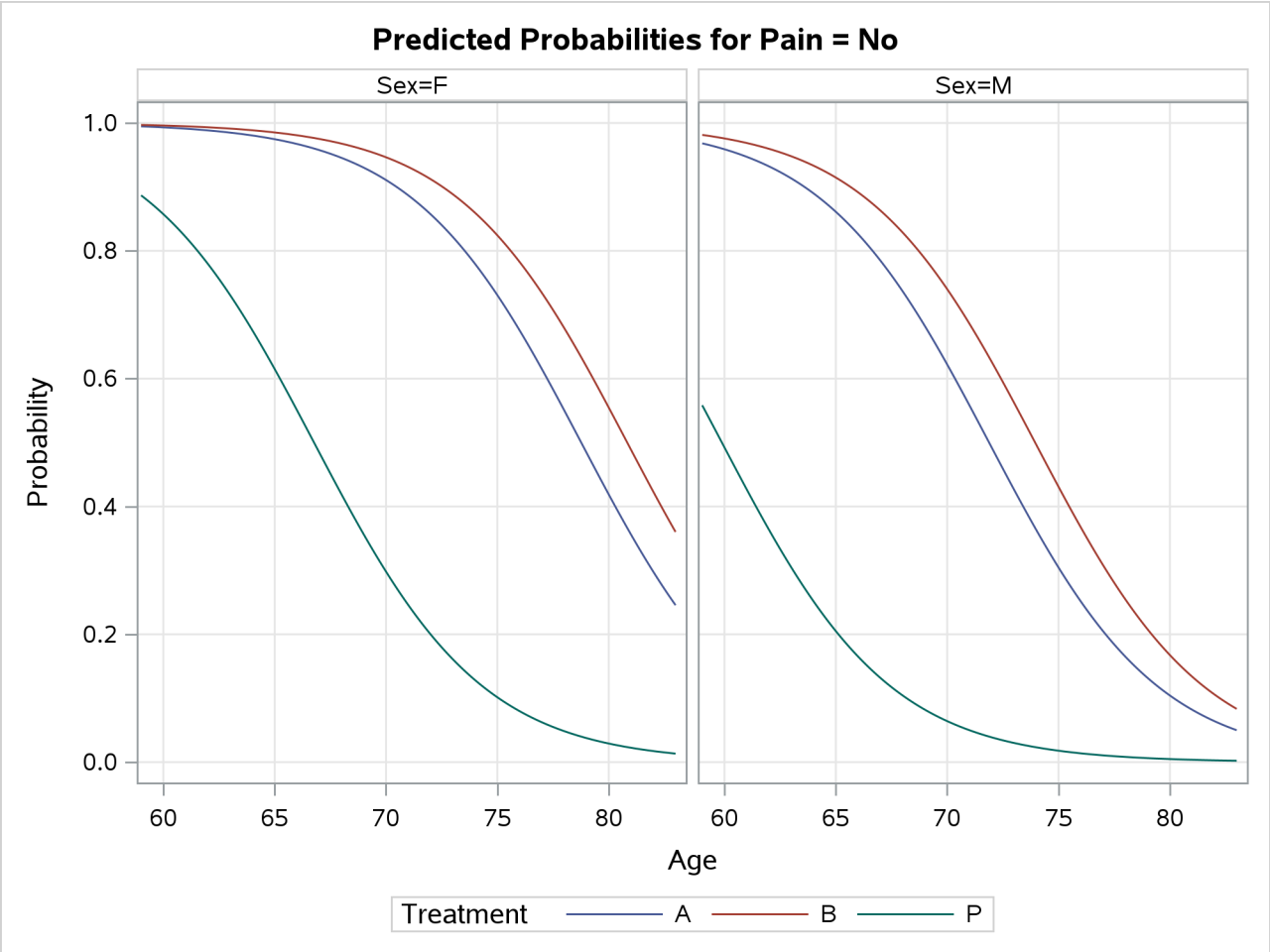

Output 79.2.12 Model-Predicted Probabilities by Treatment

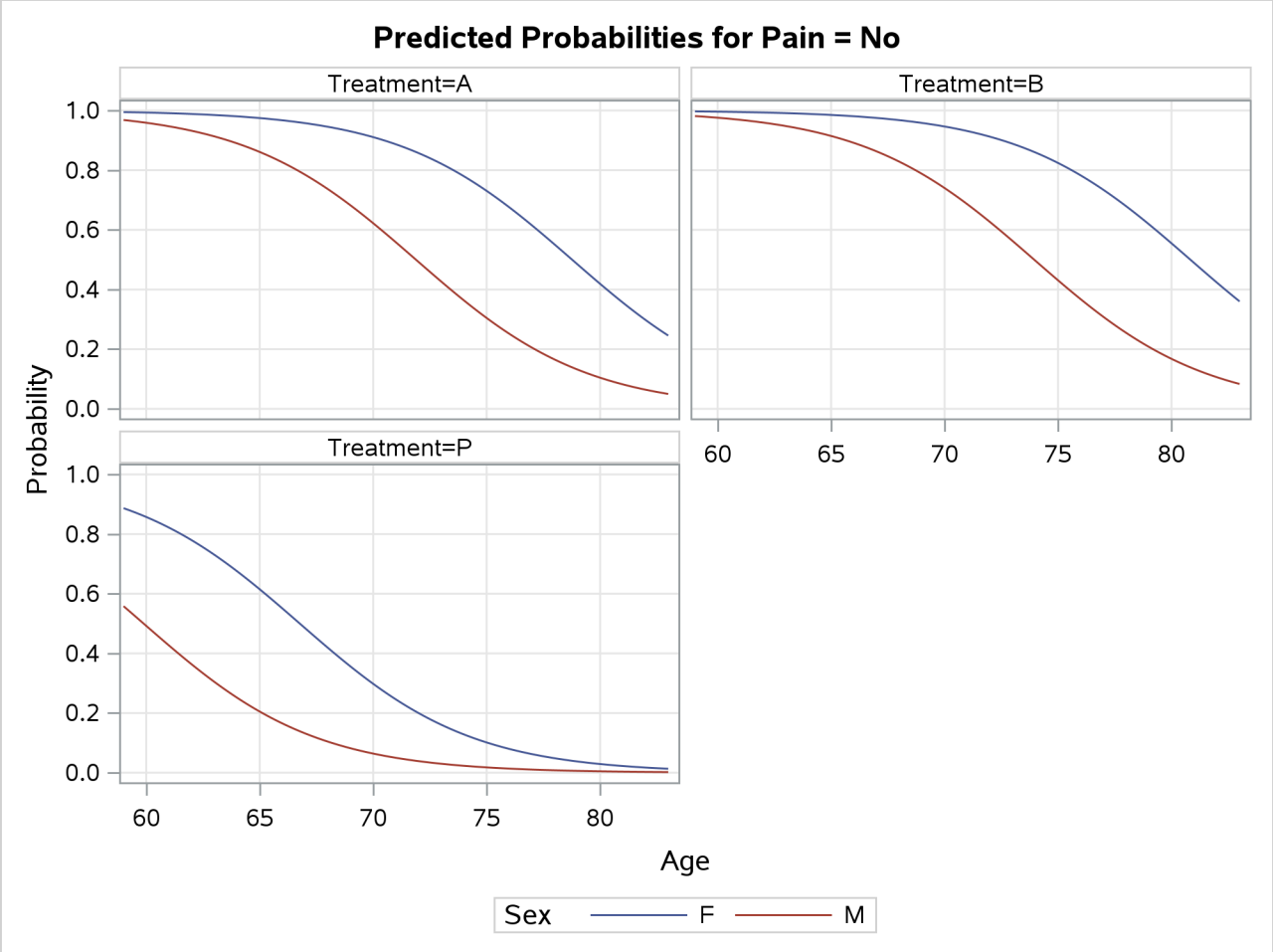

Example 79.3: Ordinal Logistic Regression

(View the complete code for this example (logiex3.sas) in the example repository.)

Consider a study of the effects on taste of various cheese additives. Researchers tested four cheese additives and obtained 52 response ratings for each additive. Each response was measured on a scale of nine categories ranging from strong dislike (1) to excellent taste (9). The data, given in McCullagh and Nelder (1989, p. 175) in the form of a two-way frequency table of additive by rating, are saved in the data set Cheese by using the following program. The variable y contains the response rating. The variable Additive specifies the cheese additive (1, 2, 3, or 4). The variable freq gives the frequency with which each additive received each rating.

```
data Cheese;
   do Additive = 1 to 4;
      do y = 1 to 9;
         input freq @@;
         output;
      end;
   end;
```

```
   label y='Taste Rating';
   datalines;
0  0  1  7  8  8 19  8  1
6  9 12 11  7  6  1  0  0
1  1  6  8 23  7  5  1  0
0  0  0  1  3  7 14 16 11
;
```

The response variable y is ordinally scaled. A cumulative logit model is used to investigate the effects of the cheese additives on taste. The following statements invoke PROC LOGISTIC to fit this model with y as the response variable and three indicator variables as explanatory variables, with the fourth additive as the reference level. With this parameterization, each Additive parameter compares an additive to the fourth additive. The COVB option displays the estimated covariance matrix, and the NOODDSRATIO option suppresses the default odds ratio table. The ODDSRATIO statement computes odds ratios for all combinations of the Additive levels. The PLOTS option produces a graphical display of the odds ratios, and the EFFECTPLOT statement displays the predicted probabilities.

```
ods graphics on;
proc logistic data=Cheese plots(only)=oddsratio(range=clip);
   freq freq;
   class Additive (param=ref ref='4');
   model y=Additive / covb nooddsratio;
   oddsratio Additive;
   effectplot / polybar;
   title 'Multiple Response Cheese Tasting Experiment';
run;
```

The "Response Profile" table in Output 79.3.1 shows that the strong dislike (y=1) end of the rating scale is associated with lower Ordered Values in the "Response Profile" table; hence the probability of disliking the additives is modeled.

The score chi-square for testing the proportional odds assumption is 17.287, which is not significant with respect to a chi-square distribution with 21 degrees of freedom ($p = 0.694$). This indicates that the proportional odds assumption is reasonable. The positive value (1.6128) for the parameter estimate for Additive1 indicates a tendency toward the lower-numbered categories of the first cheese additive relative to the fourth. In other words, the fourth additive tastes better than the first additive. The second and third additives are both less favorable than the fourth additive. The relative magnitudes of these slope estimates imply the preference ordering: fourth, first, third, second.

Output 79.3.1 Proportional Odds Model Regression Analysis

Multiple Response Cheese Tasting Experiment

The LOGISTIC Procedure

Model Information		
Data Set	WORK.CHEESE	
Response Variable	y	Taste Rating
Number of Response Levels	9	
Frequency Variable	freq	
Model	cumulative logit	
Optimization Technique	Fisher's scoring	

Output 79.3.1 *continued*

Number of Observations Read	36
Number of Observations Used	28
Sum of Frequencies Read	208
Sum of Frequencies Used	208

Response Profile

Ordered Value	y	Total Frequency
1	1	7
2	2	10
3	3	19
4	4	27
5	5	41
6	6	28
7	7	39
8	8	25
9	9	12

Probabilities modeled are cumulated over the lower Ordered Values.

Note: 8 observations having nonpositive frequencies or weights were excluded since they do not contribute to the analysis.

Class Level Information

Class	Value	Design Variables		
Additive	**1**	1	0	0
	2	0	1	0
	3	0	0	1
	4	0	0	0

Model Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Score Test for the Proportional Odds Assumption

Chi-Square	DF	Pr > ChiSq
17.2866	21	0.6936

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	875.802	733.348
SC	902.502	770.061
-2 Log L	859.802	711.348

Output 79.3.1 *continued*

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	148.4539	3	<.0001
Score	111.2670	3	<.0001
Wald	115.1504	3	<.0001

Type 3 Analysis of Effects			
Effect	**DF**	**Wald Chi-Square**	**Pr > ChiSq**
Additive	3	115.1504	<.0001

Analysis of Maximum Likelihood Estimates						
Parameter		**DF**	**Estimate**	**Standard Error**	**Wald Chi-Square**	**Pr > ChiSq**
Intercept	**1**	1	-7.0801	0.5624	158.4851	<.0001
Intercept	**2**	1	-6.0249	0.4755	160.5500	<.0001
Intercept	**3**	1	-4.9254	0.4272	132.9484	<.0001
Intercept	**4**	1	-3.8568	0.3902	97.7087	<.0001
Intercept	**5**	1	-2.5205	0.3431	53.9704	<.0001
Intercept	**6**	1	-1.5685	0.3086	25.8374	<.0001
Intercept	**7**	1	-0.0669	0.2658	0.0633	0.8013
Intercept	**8**	1	1.4930	0.3310	20.3439	<.0001
Additive	**1**	1	1.6128	0.3778	18.2265	<.0001
Additive	**2**	1	4.9645	0.4741	109.6427	<.0001
Additive	**3**	1	3.3227	0.4251	61.0931	<.0001

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	67.6	**Somers' D**	0.578
Percent Discordant	9.8	**Gamma**	0.746
Percent Tied	22.6	**Tau-a**	0.500
Pairs	18635	**c**	0.789

The odds ratio results in Output 79.3.2 show the preferences more clearly. For example, the "Additive 1 vs 4" odds ratio says that the first additive has 5.017 times the odds of receiving a lower score than the fourth additive; that is, the first additive is 5.017 times more likely than the fourth additive to receive a lower score. Output 79.3.3 displays the odds ratios graphically; the range of the confidence limits is truncated by the RANGE=CLIP option, so you can see that "1" is not contained in any of the intervals.

Output 79.3.2 Odds Ratios of All Pairs of Additive Levels

Odds Ratio Estimates and Wald Confidence Intervals			
Odds Ratio	**Estimate**	**95% Confidence Limits**	
Additive 1 vs 2	0.035	0.015	0.080
Additive 1 vs 3	0.181	0.087	0.376
Additive 1 vs 4	5.017	2.393	10.520
Additive 2 vs 3	5.165	2.482	10.746
Additive 2 vs 4	143.241	56.558	362.777
Additive 3 vs 4	27.734	12.055	63.805

Output 79.3.3 Plot of Odds Ratios for Additive

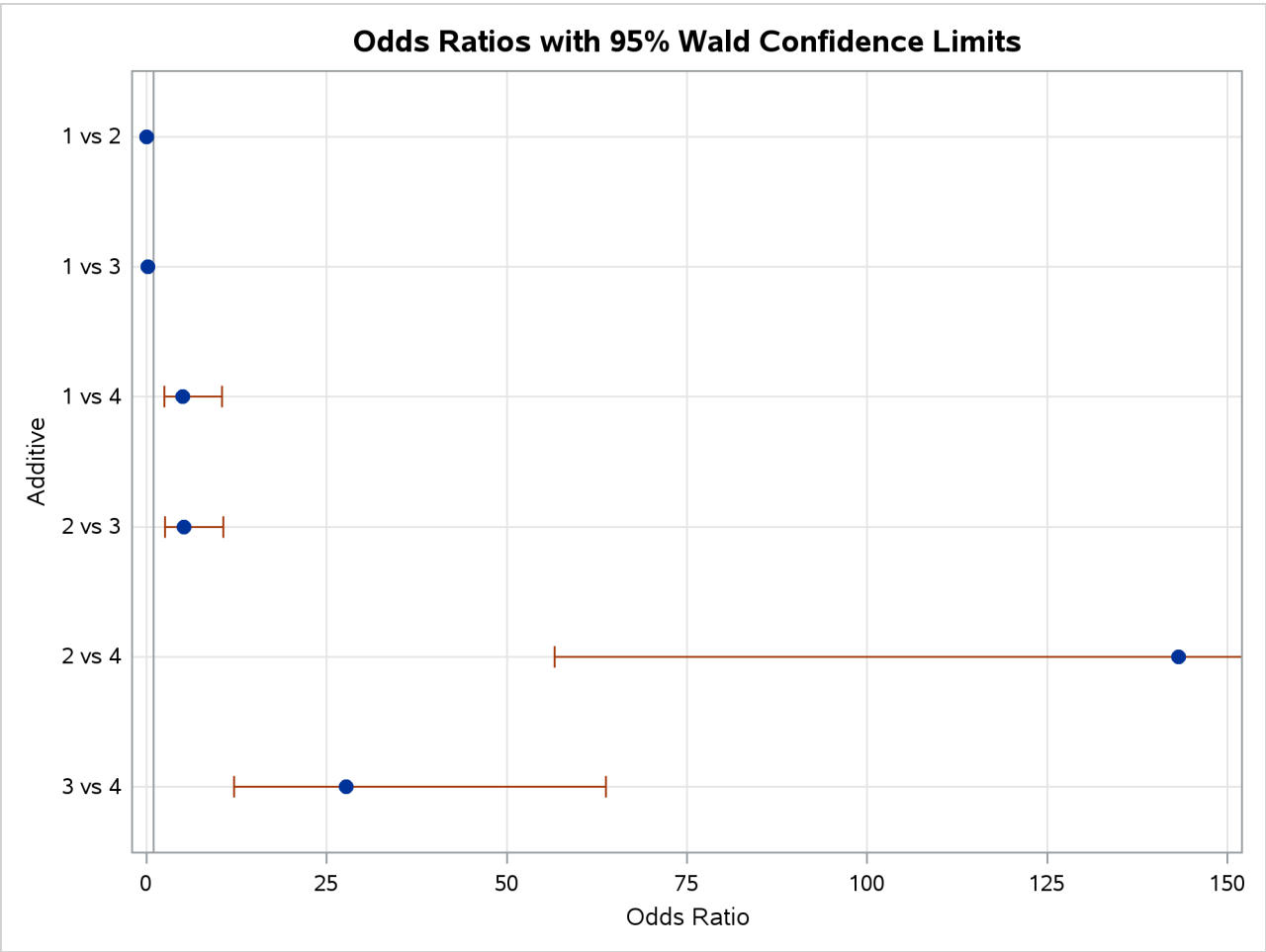

The estimated covariance matrix of the parameters is displayed in Output 79.3.4.

Output 79.3.4 Estimated Covariance Matrix

Estimated Covariance Matrix											
Parameter	**Intercept_1**	**Intercept_2**	**Intercept_3**	**Intercept_4**	**Intercept_5**	**Intercept_6**	**Intercept_7**	**Intercept_8**	**Additive1**	**Additive2**	**Additive3**
Intercept_1	0.316291	0.219581	0.176278	0.147694	0.114024	0.091085	0.057814	0.041304	-0.09419	-0.18686	-0.13565
Intercept_2	0.219581	0.226095	0.177806	0.147933	0.11403	0.091081	0.057813	0.041304	-0.09421	-0.18161	-0.13569
Intercept_3	0.176278	0.177806	0.182473	0.148844	0.114092	0.091074	0.057807	0.0413	-0.09427	-0.1687	-0.1352
Intercept_4	0.147694	0.147933	0.148844	0.152235	0.114512	0.091109	0.05778	0.041277	-0.09428	-0.14717	-0.13118
Intercept_5	0.114024	0.11403	0.114092	0.114512	0.117713	0.091821	0.057721	0.041162	-0.09246	-0.11415	-0.11207
Intercept_6	0.091085	0.091081	0.091074	0.091109	0.091821	0.09522	0.058312	0.041324	-0.08521	-0.09113	-0.09122
Intercept_7	0.057814	0.057813	0.057807	0.05778	0.057721	0.058312	0.07064	0.04878	-0.06041	-0.05781	-0.05802
Intercept_8	0.041304	0.041304	0.0413	0.041277	0.041162	0.041324	0.04878	0.109562	-0.04436	-0.0413	-0.04143
Additive1	-0.09419	-0.09421	-0.09427	-0.09428	-0.09246	-0.08521	-0.06041	-0.04436	0.142715	0.094072	0.092128
Additive2	-0.18686	-0.18161	-0.1687	-0.14717	-0.11415	-0.09113	-0.05781	-0.0413	0.094072	0.22479	0.132877
Additive3	-0.13565	-0.13569	-0.1352	-0.13118	-0.11207	-0.09122	-0.05802	-0.04143	0.092128	0.132877	0.180709

Output 79.3.5 displays the probability of each taste rating y within each additive. You can see that Additive=1 mostly receives ratings of 5 to 7, Additive=2 mostly receives ratings of 2 to 5, Additive=3 mostly receives ratings of 4 to 6, and Additive=4 mostly receives ratings of 7 to 9, which also confirms the previously discussed preference orderings.

Output 79.3.5 Model-Predicted Probabilities

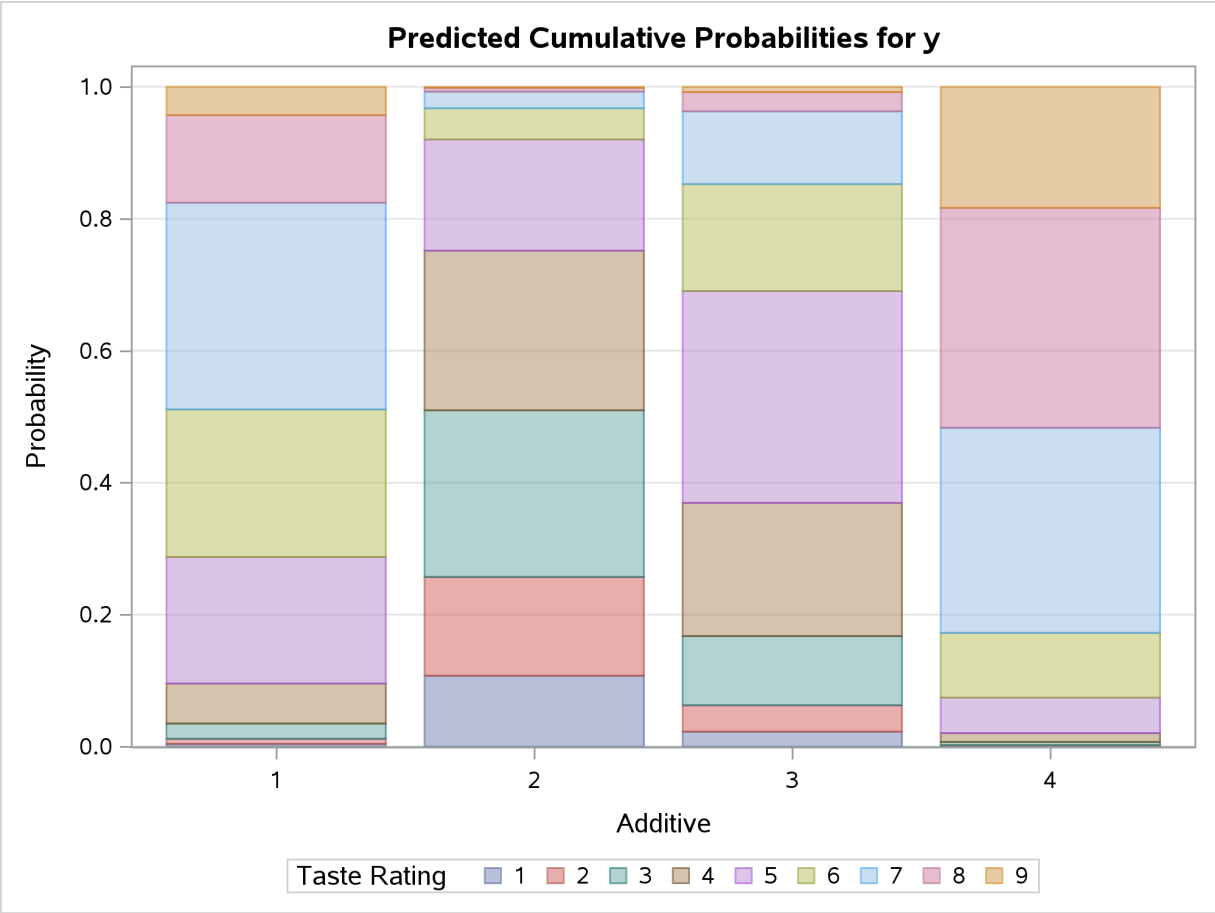

Alternatively, you can use an adjacent-category logit model. The following statements invoke PROC LOGISTIC to fit this model:

```
proc logistic data=Cheese
   plots(only)=effect(x=y sliceby=additive connect yrange=(0,0.4));
   freq freq;
   class Additive (param=ref ref='4');
   model y=Additive / nooddsratio link=alogit;
   oddsratio Additive;
   title 'Multiple Response Cheese Tasting Experiment';
run;
```

You can see that the parameter estimates for the intercepts in Output 79.3.6 are no longer strictly increasing, as they are for the cumulative model in Output 79.3.1.

Output 79.3.6 Adjacent-Category Logistic Regression Analysis

Multiple Response Cheese Tasting Experiment

The LOGISTIC Procedure

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	**1**	1	-2.2262	0.5542	16.1350	<.0001
Intercept	**2**	1	-2.4238	0.4620	27.5216	<.0001
Intercept	**3**	1	-1.9852	0.3800	27.2909	<.0001
Intercept	**4**	1	-1.8164	0.3294	30.4096	<.0001
Intercept	**5**	1	-0.6908	0.3097	4.9737	0.0257
Intercept	**6**	1	-1.0433	0.2841	13.4902	0.0002
Intercept	**7**	1	0.0322	0.2667	0.0146	0.9040
Intercept	**8**	1	0.5160	0.3527	2.1402	0.1435
Additive	**1**	1	0.7172	0.1783	16.1776	<.0001
Additive	**2**	1	1.9838	0.2463	64.8748	<.0001
Additive	**3**	1	1.3826	0.2122	42.4478	<.0001

The odds ratio results in Output 79.3.7 show the same preferences that you obtain from the proportional odds model. Output 79.3.8 displays the probability of each additive according to the taste rating.

Output 79.3.7 Odds Ratios of All Pairs of Additive Levels

Odds Ratio Estimates and Wald Confidence Intervals

Odds Ratio	Estimate	95% Confidence Limits	
Additive 1 vs 2	0.282	0.192	0.414
Additive 1 vs 3	0.514	0.377	0.700
Additive 1 vs 4	2.049	1.444	2.906
Additive 2 vs 3	1.824	1.355	2.456
Additive 2 vs 4	7.271	4.487	11.782
Additive 3 vs 4	3.985	2.629	6.040

Output 79.3.8 Model-Predicted Probabilities

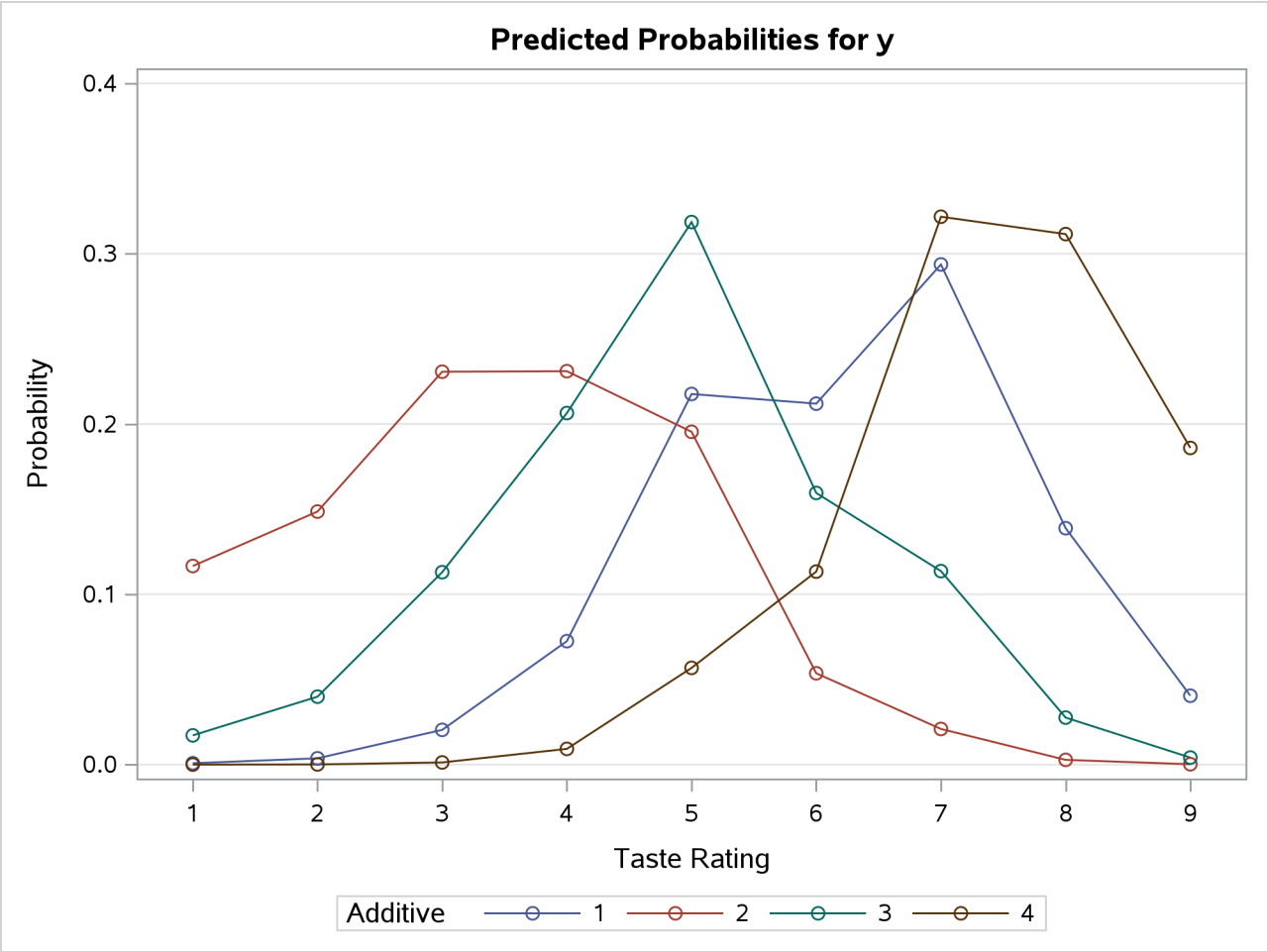

Example 79.4: Nominal Response Data: Generalized Logits Model

(View the complete code for this example (logiex4.sas) in the example repository.)

Over the course of one school year, third graders from three different schools are exposed to three different styles of mathematics instruction: a self-paced computer-learning style, a team approach, and a traditional class approach. The students are asked which style they prefer and their responses, classified by the type of program they are in (a regular school day versus a regular day supplemented with an afternoon school program), are displayed in Table 79.20. The data set is from Stokes, Davis, and Koch (2012), and is also analyzed in the section "Generalized Logits Model" on page 2236 in Chapter 36, "The CATMOD Procedure."

Table 79.20 School Program Data

		Learning Style Preference		
School	**Program**	**Self**	**Team**	**Class**
1	Regular	10	17	26
1	Afternoon	5	12	50
2	Regular	21	17	26
2	Afternoon	16	12	36
3	Regular	15	15	16
3	Afternoon	12	12	20

The levels of the response variable (self, team, and class) have no essential ordering, so a logistic regression is performed on the generalized logits. The model to be fit is

$$\log\left(\frac{\pi_{hij}}{\pi_{hir}}\right) = \alpha_j + \mathbf{x}'_{hi}\boldsymbol{\beta}_j$$

where π_{hij} is the probability that a student in school h and program i prefers teaching style j, $j \neq r$, and style r is the baseline style (in this case, class). There are separate sets of intercept parameters α_j and regression parameters $\boldsymbol{\beta}_j$ for each logit, and the vector $\mathbf{x}_{hi}$ is the set of explanatory variables for the hith population. Thus, two logits are modeled for each school and program combination: the logit comparing self to class and the logit comparing team to class.

The following statements create the data set school and request the analysis. The LINK=GLOGIT option forms the generalized logits. The response variable option ORDER=DATA means that the response variable levels are ordered as they exist in the data set: self, team, and class; thus, the logits are formed by comparing self to class and by comparing team to class. The ODDSRATIO statement produces odds ratios in the presence of interactions, and a graphical display of the requested odds ratios is produced when ODS Graphics is enabled.

```
data school;
   length Program $ 9;
   input School Program $ Style $ Count @@;
   datalines;
1 regular   self 10  1 regular   team 17  1 regular   class 26
1 afternoon self  5  1 afternoon team 12  1 afternoon class 50
2 regular   self 21  2 regular   team 17  2 regular   class 26
2 afternoon self 16  2 afternoon team 12  2 afternoon class 36
3 regular   self 15  3 regular   team 15  3 regular   class 16
3 afternoon self 12  3 afternoon team 12  3 afternoon class 20
;

ods graphics on;
proc logistic data=school;
   freq Count;
   class School Program(ref=first);
   model Style(order=data)=School Program School*Program / link=glogit;
   oddsratio program;
run;
```

Summary information about the model, the response variable, and the classification variables are displayed in Output 79.4.1.

Output 79.4.1 Analysis of Saturated Model

The LOGISTIC Procedure

Model Information	
Data Set	WORK.SCHOOL
Response Variable	Style
Number of Response Levels	3
Frequency Variable	Count
Model	generalized logit
Optimization Technique	Newton-Raphson

Number of Observations Read	18
Number of Observations Used	18
Sum of Frequencies Read	338
Sum of Frequencies Used	338

Response Profile

Ordered Value	Style	Total Frequency
1	self	79
2	team	85
3	class	174

Logits modeled use Style='class' as the reference category.

Class Level Information

Class	Value	Design Variables	
School	**1**	1	0
	2	0	1
	3	-1	-1
Program	**afternoon**	-1	
	regular	1	

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

The "Testing Global Null Hypothesis: BETA=0" table in Output 79.4.2 shows that the parameters are significantly different from zero.

Output 79.4.2 Analysis of Saturated Model

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	699.404	689.156
SC	707.050	735.033
-2 Log L	695.404	665.156

Output 79.4.2 *continued*

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	30.2480	10	0.0008
Score	28.3738	10	0.0016
Wald	25.6828	10	0.0042

However, the "Type 3 Analysis of Effects" table in Output 79.4.3 shows that the interaction effect is clearly nonsignificant.

Output 79.4.3 Analysis of Saturated Model

Joint Tests			
Effect	**DF**	**Wald Chi-Square**	**Pr > ChiSq**
School	4	14.5522	0.0057
Program	2	10.4815	0.0053
School*Program	4	1.7439	0.7827

Note: Under full-rank parameterizations, Type 3 effect tests are replaced by joint tests. The joint test for an effect is a test that all the parameters associated with that effect are zero. Such joint tests might not be equivalent to Type 3 effect tests under GLM parameterization.

Analysis of Maximum Likelihood Estimates								
Parameter			**Style**	**DF**	**Estimate**	**Standard Error**	**Wald Chi-Square**	**Pr > ChiSq**
Intercept			**self**	1	-0.8097	0.1488	29.5989	<.0001
Intercept			**team**	1	-0.6585	0.1366	23.2449	<.0001
School	**1**		**self**	1	-0.8194	0.2281	12.9066	0.0003
School	**1**		**team**	1	-0.2675	0.1881	2.0233	0.1549
School	**2**		**self**	1	0.2974	0.1919	2.4007	0.1213
School	**2**		**team**	1	-0.1033	0.1898	0.2961	0.5863
Program	**regular**		**self**	1	0.3985	0.1488	7.1684	0.0074
Program	**regular**		**team**	1	0.3537	0.1366	6.7071	0.0096
School*Program	**1**	**regular**	**self**	1	0.2751	0.2281	1.4547	0.2278
School*Program	**1**	**regular**	**team**	1	0.1474	0.1881	0.6143	0.4332
School*Program	**2**	**regular**	**self**	1	-0.0998	0.1919	0.2702	0.6032
School*Program	**2**	**regular**	**team**	1	-0.0168	0.1898	0.0079	0.9293

The table produced by the ODDSRATIO statement is displayed in Output 79.4.4. The differences between the program preferences are small across all the styles (logits) compared to their variability as displayed by the confidence limits in Output 79.4.5, confirming that the interaction effect is nonsignificant.

Output 79.4.4 Odds Ratios for Style

Odds Ratio Estimates and Wald Confidence Intervals			
Odds Ratio	**Estimate**	**95% Confidence Limits**	
Style self: Program regular vs afternoon at School=1	3.846	1.190	12.435
Style team: Program regular vs afternoon at School=1	2.724	1.132	6.554
Style self: Program regular vs afternoon at School=2	1.817	0.798	4.139
Style team: Program regular vs afternoon at School=2	1.962	0.802	4.799
Style self: Program regular vs afternoon at School=3	1.563	0.572	4.265
Style team: Program regular vs afternoon at School=3	1.563	0.572	4.265

Output 79.4.5 Plot of Odds Ratios for Style

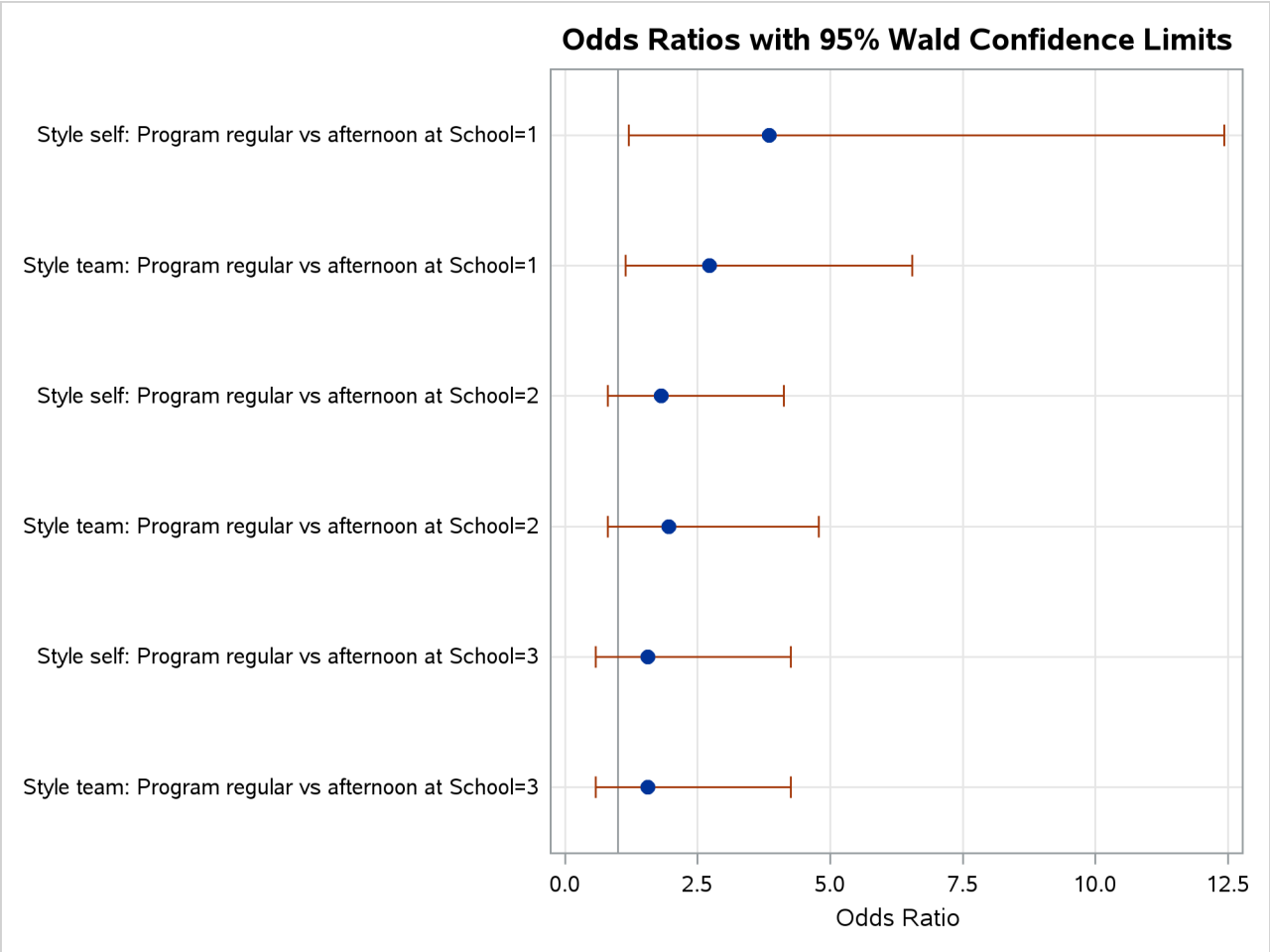

Because the interaction effect is clearly nonsignificant, a main-effects model is fit with the following statements. The EFFECTPLOT statement creates a plot of the predicted values versus the levels of the School variable at each level of the Program variables. The CLM option adds confidence bars, and the NOOBS option suppresses the display of the observations.

```
proc logistic data=school;
   freq Count;
   class School Program(ref=first);
   model Style(order=data)=School Program / link=glogit;
   effectplot interaction(plotby=Program) / clm noobs;
run;
```

All of the global fit tests in Output 79.4.6 suggest the model is significant, and the Type 3 tests show that the school and program effects are also significant.

Output 79.4.6 Analysis of Main-Effects Model

The LOGISTIC Procedure

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	699.404	682.934
SC	707.050	713.518
-2 Log L	695.404	666.934

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	28.4704	6	<.0001
Score	27.1190	6	0.0001
Wald	25.5881	6	0.0003

Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
School	4	14.8424	0.0050
Program	2	10.9160	0.0043

The parameter estimates, tests for individual parameters, and odds ratios are displayed in Output 79.4.7. The Program variable has nearly the same effect on both logits, while School=1 has the largest effect of the schools.

Output 79.4.7 Estimates

Analysis of Maximum Likelihood Estimates							
Parameter		**Style**	**DF**	**Estimate**	**Standard Error**	**Wald Chi-Square**	**Pr > ChiSq**
Intercept		**self**	1	-0.7978	0.1465	29.6502	<.0001
Intercept		**team**	1	-0.6589	0.1367	23.2300	<.0001
School	**1**	**self**	1	-0.7992	0.2198	13.2241	0.0003
School	**1**	**team**	1	-0.2786	0.1867	2.2269	0.1356
School	**2**	**self**	1	0.2836	0.1899	2.2316	0.1352
School	**2**	**team**	1	-0.0985	0.1892	0.2708	0.6028
Program	**regular**	**self**	1	0.3737	0.1410	7.0272	0.0080
Program	**regular**	**team**	1	0.3713	0.1353	7.5332	0.0061

Odds Ratio Estimates				
Effect	**Style**	**Point Estimate**	**95% Wald Confidence Limits**	
School 1 vs 3	**self**	0.269	0.127	0.570
School 1 vs 3	**team**	0.519	0.267	1.010
School 2 vs 3	**self**	0.793	0.413	1.522
School 2 vs 3	**team**	0.622	0.317	1.219
Program regular vs afternoon	**self**	2.112	1.215	3.670
Program regular vs afternoon	**team**	2.101	1.237	3.571

The interaction plots in Output 79.4.8 show that School=1 and Program=afternoon have a preference for the traditional classroom style. Of course, because these are not simultaneous confidence intervals, the nonoverlapping 95% confidence limits do not take the place of an actual test.

Output 79.4.8 Model-Predicted Probabilities

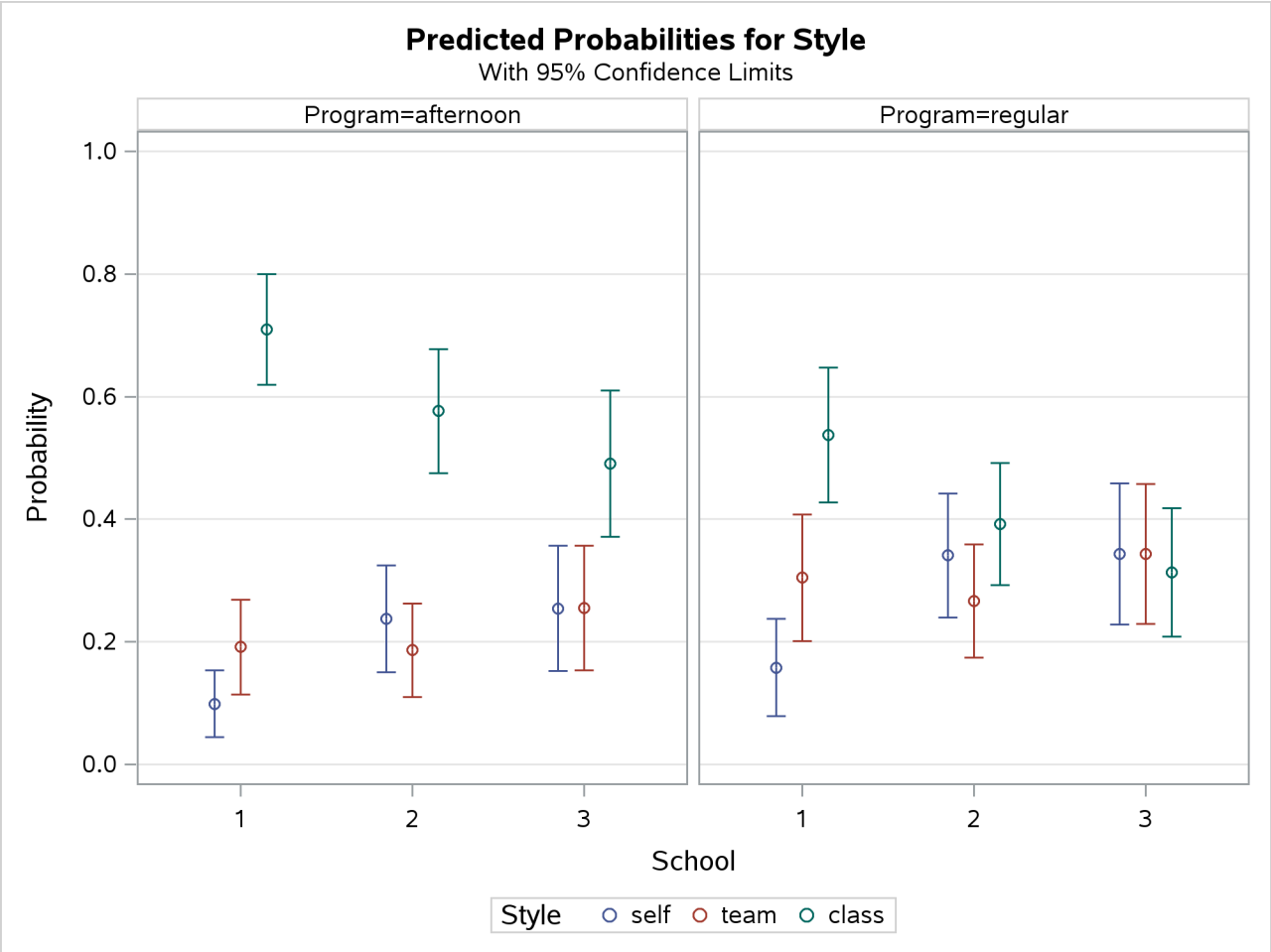

Example 79.5: Stratified Sampling

(View the complete code for this example (logiex5.sas) in the example repository.)

Consider the hypothetical example in Fleiss (1981, pp. 6–7), in which a test is applied to a sample of 1,000 people known to have a disease and to another sample of 1,000 people known not to have the same disease. In the diseased sample, 950 test positive; in the nondiseased sample, only 10 test positive. If the true disease rate in the population is 1 in 100, specifying PEVENT=0.01 results in the correct positive and negative predictive values for the stratified sampling scheme. Omitting the PEVENT= option is equivalent to using the overall sample disease rate (1000/2000 = 0.5) as the value of the PEVENT= option, which would ignore the stratified sampling.

The statements to create the data set and perform the analysis are as follows:

```
data Screen;
   do Disease='Present','Absent';
      do Test=1,0;
         input Count @@;
```

```
            output;
         end;
      end;
      datalines;
950  50
 10 990
;

proc logistic data=Screen;
   freq Count;
   model Disease(event='Present')=Test
         / pevent=.5 .01 ctable pprob=.5;
run;
```

The response variable option EVENT= indicates that Disease='Present' is the event. The CTABLE option is specified to produce a classification table. Specifying PPROB=0.5 indicates a cutoff probability of 0.5. A list of two probabilities, 0.5 and 0.01, is specified for the PEVENT= option; 0.5 corresponds to the overall sample disease rate, and 0.01 corresponds to a true disease rate of 1 in 100.

The classification table is shown in Output 79.5.1.

Output 79.5.1 Positive and Negative Predictive Values

The LOGISTIC Procedure

Classification Table										
		Correct		Incorrect		Percentages				
Prob Event	Prob Level	Event	Non-Event	Event	Non-Event	Correct	Sensi-tivity	Speci-ficity	Pos Pred	Neg Pred
0.500	**0.500**	950	990	10	50	97.0	95.0	99.0	99.0	95.2
0.010	**0.500**	950	990	10	50	99.0	95.0	99.0	49.0	99.9

In the classification table, the column "Prob Level" represents the cutoff values (the settings of the PPROB= option) for predicting whether an observation is an event. The "Correct" columns list the numbers of subjects that are correctly predicted as events and nonevents, respectively, and the "Incorrect" columns list the number of nonevents incorrectly predicted as events and the number of events incorrectly predicted as nonevents, respectively. For PEVENT=0.5, the positive predictive value is 99% and the negative predictive value is 95.2%. These results ignore the fact that the samples were stratified and incorrectly assume that the overall sample proportion of disease (which is 0.5) estimates the true disease rate. For a true disease rate of 0.01, the positive and negative predictive values are 49% and 99.9%, respectively, as shown in the second line of the classification table.

Example 79.6: Logistic Regression Diagnostics

(View the complete code for this example (logiex6.sas) in the example repository.)

In a controlled experiment to study the effect of the rate and volume of air intake on a transient reflex vasoconstriction in the skin of the digits, 39 tests under various combinations of rate and volume of air intake were obtained (Finney 1947). The endpoint of each test is whether or not vasoconstriction occurred. Pregibon (1981) uses this set of data to illustrate the diagnostic measures he proposes for detecting influential observations and to quantify their effects on various aspects of the maximum likelihood fit.

The vasoconstriction data are saved in the data set vaso:

```
data vaso;
   length Response $12;
   input Volume Rate Response @@;
   LogVolume=log(Volume);
   LogRate=log(Rate);
   datalines;
3.70  0.825  constrict       3.50  1.09   constrict
1.25  2.50   constrict       0.75  1.50   constrict
0.80  3.20   constrict       0.70  3.50   constrict
0.60  0.75   no_constrict    1.10  1.70   no_constrict
0.90  0.75   no_constrict    0.90  0.45   no_constrict
0.80  0.57   no_constrict    0.55  2.75   no_constrict
0.60  3.00   no_constrict    1.40  2.33   constrict
0.75  3.75   constrict       2.30  1.64   constrict
3.20  1.60   constrict       0.85  1.415  constrict
1.70  1.06   no_constrict    1.80  1.80   constrict
0.40  2.00   no_constrict    0.95  1.36   no_constrict
1.35  1.35   no_constrict    1.50  1.36   no_constrict
1.60  1.78   constrict       0.60  1.50   no_constrict
1.80  1.50   constrict       0.95  1.90   no_constrict
1.90  0.95   constrict       1.60  0.40   no_constrict
2.70  0.75   constrict       2.35  0.03   no_constrict
1.10  1.83   no_constrict    1.10  2.20   constrict
1.20  2.00   constrict       0.80  3.33   constrict
0.95  1.90   no_constrict    0.75  1.90   no_constrict
1.30  1.625  constrict
;
```

In the data set vaso, the variable Response represents the outcome of a test. The variable LogVolume represents the log of the volume of air intake, and the variable LogRate represents the log of the rate of air intake.

The following statements invoke PROC LOGISTIC to fit a logistic regression model to the vasoconstriction data, where Response is the response variable, and LogRate and LogVolume are the explanatory variables. Regression diagnostics are displayed when ODS Graphics is enabled, and the INFLUENCE option is specified to display a table of the regression diagnostics.

```
ods graphics on;
title 'Occurrence of Vasoconstriction';
proc logistic data=vaso;
   model Response=LogRate LogVolume/influence;
```

```
run;
```

Results of the model fit are shown in Output 79.6.1. Both LogRate and LogVolume are statistically significant to the occurrence of vasoconstriction ($p = 0.0131$ and $p = 0.0055$, respectively). Their positive parameter estimates indicate that a higher inspiration rate or a larger volume of air intake is likely to increase the probability of vasoconstriction.

Output 79.6.1 Logistic Regression Analysis for Vasoconstriction Data

Occurrence of Vasoconstriction

The LOGISTIC Procedure

Model Information	
Data Set	WORK.VASO
Response Variable	Response
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	39
Number of Observations Used	39

Response Profile

Ordered Value	Response	Total Frequency
1	constrict	20
2	no_constrict	19

Probability modeled is Response='constrict'.

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	56.040	35.227
SC	57.703	40.218
-2 Log L	54.040	29.227

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	24.8125	2	<.0001
Score	16.6324	2	0.0002
Wald	7.8876	2	0.0194

Output 79.6.1 *continued*

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-2.8754	1.3208	4.7395	0.0295
LogRate	1	4.5617	1.8380	6.1597	0.0131
LogVolume	1	5.1793	1.8648	7.7136	0.0055

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
LogRate	95.744	2.610	>999.999
LogVolume	177.562	4.592	>999.999

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	93.7	Somers' D	0.874
Percent Discordant	6.3	Gamma	0.874
Percent Tied	0.0	Tau-a	0.448
Pairs	380	c	0.937

The INFLUENCE option displays the values of the explanatory variables (LogRate and LogVolume) for each observation, a column for each diagnostic produced, and the *case number* that represents the sequence number of the observation (Output 79.6.2).

Output 79.6.2 Regression Diagnostics from the INFLUENCE Option

	Covariates		Regression Diagnostics									
Case Number	LogRate	LogVolume	Pearson Residual	Deviance Residual	Hat Matrix Diagonal	Intercept DfBeta	LogRate DfBeta	LogVolume DfBeta	Confidence Interval Displacement C	Confidence Interval Displacement CBar	Delta Deviance	Delta Chi-Square
1	-0.1924	1.3083	0.2205	0.3082	0.0927	-0.0165	0.0193	0.0556	0.00548	0.00497	0.1000	0.0536
2	0.0862	1.2528	0.1349	0.1899	0.0429	-0.0134	0.0151	0.0261	0.000853	0.000816	0.0369	0.0190
3	0.9163	0.2231	0.2923	0.4049	0.0612	-0.0492	0.0660	0.0589	0.00593	0.00557	0.1695	0.0910
4	0.4055	-0.2877	3.5181	2.2775	0.0867	1.0734	-0.9302	-1.0180	1.2873	1.1756	6.3626	13.5523
5	1.1632	-0.2231	0.5287	0.7021	0.1158	-0.0832	0.1411	0.0583	0.0414	0.0366	0.5296	0.3161
6	1.2528	-0.3567	0.6090	0.7943	0.1524	-0.0922	0.1710	0.0381	0.0787	0.0667	0.6976	0.4376
7	-0.2877	-0.5108	-0.0328	-0.0464	0.00761	-0.00280	0.00274	0.00265	8.321E-6	8.258E-6	0.00216	0.00109
8	0.5306	0.0953	-1.0196	-1.1939	0.0559	-0.1444	0.0613	0.0570	0.0652	0.0616	1.4870	1.1011
9	-0.2877	-0.1054	-0.0938	-0.1323	0.0342	-0.0178	0.0173	0.0153	0.000322	0.000311	0.0178	0.00911
10	-0.7985	-0.1054	-0.0293	-0.0414	0.00721	-0.00245	0.00246	0.00211	6.256E-6	6.211E-6	0.00172	0.000862
11	-0.5621	-0.2231	-0.0370	-0.0523	0.00969	-0.00361	0.00358	0.00319	0.000014	0.000013	0.00274	0.00138
12	1.0116	-0.5978	-0.5073	-0.6768	0.1481	-0.1173	0.0647	0.1651	0.0525	0.0447	0.5028	0.3021
13	1.0986	-0.5108	-0.7751	-0.9700	0.1628	-0.0931	-0.00946	0.1775	0.1395	0.1168	1.0577	0.7175
14	0.8459	0.3365	0.2559	0.3562	0.0551	-0.0414	0.0538	0.0527	0.00404	0.00382	0.1307	0.0693
15	1.3218	-0.2877	0.4352	0.5890	0.1336	-0.0940	0.1408	0.0643	0.0337	0.0292	0.3761	0.2186
16	0.4947	0.8329	0.1576	0.2215	0.0402	-0.0198	0.0234	0.0307	0.00108	0.00104	0.0501	0.0259
17	0.4700	1.1632	0.0709	0.1001	0.0172	-0.00630	0.00701	0.00914	0.000089	0.000088	0.0101	0.00511
18	0.3471	-0.1625	2.9062	2.1192	0.0954	0.9595	-0.8279	-0.8477	0.9845	0.8906	5.3817	9.3363
19	0.0583	0.5306	-1.0718	-1.2368	0.1315	-0.2591	0.2024	-0.00488	0.2003	0.1740	1.7037	1.3227
20	0.5878	0.5878	0.2405	0.3353	0.0525	-0.0331	0.0421	0.0518	0.00338	0.00320	0.1156	0.0610
21	0.6931	-0.9163	-0.1076	-0.1517	0.0373	-0.0180	0.0158	0.0208	0.000465	0.000448	0.0235	0.0120
22	0.3075	-0.0513	-0.4193	-0.5691	0.1015	-0.1449	0.1237	0.1179	0.0221	0.0199	0.3437	0.1956
23	0.3001	0.3001	-1.0242	-1.1978	0.0761	-0.1961	0.1275	0.0357	0.0935	0.0864	1.5212	1.1355
24	0.3075	0.4055	-1.3684	-1.4527	0.0717	-0.1281	0.0410	-0.1004	0.1558	0.1447	2.2550	2.0171
25	0.5766	0.4700	0.3347	0.4608	0.0587	-0.0403	0.0570	0.0708	0.00741	0.00698	0.2193	0.1190
26	0.4055	-0.5108	-0.1595	-0.2241	0.0548	-0.0366	0.0329	0.0373	0.00156	0.00147	0.0517	0.0269
27	0.4055	0.5878	0.3645	0.4995	0.0661	-0.0327	0.0496	0.0788	0.0101	0.00941	0.2589	0.1423
28	0.6419	-0.0513	-0.8989	-1.0883	0.0647	-0.1423	0.0617	0.1025	0.0597	0.0559	1.2404	0.8639
29	-0.0513	0.6419	0.8981	1.0876	0.1682	0.2367	-0.1950	0.0286	0.1961	0.1631	1.3460	0.9697
30	-0.9163	0.4700	-0.0992	-0.1400	0.0507	-0.0224	0.0227	0.0159	0.000554	0.000526	0.0201	0.0104
31	-0.2877	0.9933	0.6198	0.8064	0.2459	0.1165	-0.0996	0.1322	0.1661	0.1253	0.7755	0.5095
32	-3.5066	0.8544	-0.00073	-0.00103	0.000022	-3.22E-6	3.405E-6	2.48E-6	1.18E-11	1.18E-11	1.065E-6	5.324E-7
33	0.6043	0.0953	-1.2062	-1.3402	0.0510	-0.0882	-0.0137	-0.00216	0.0824	0.0782	1.8744	1.5331
34	0.7885	0.0953	0.5447	0.7209	0.0601	-0.0425	0.0877	0.0671	0.0202	0.0190	0.5387	0.3157
35	0.6931	0.1823	0.5404	0.7159	0.0552	-0.0340	0.0755	0.0711	0.0180	0.0170	0.5295	0.3091
36	1.2030	-0.2231	0.4828	0.6473	0.1177	-0.0867	0.1381	0.0631	0.0352	0.0311	0.4501	0.2641
37	0.6419	-0.0513	-0.8989	-1.0883	0.0647	-0.1423	0.0617	0.1025	0.0597	0.0559	1.2404	0.8639
38	0.6419	-0.2877	-0.4874	-0.6529	0.1000	-0.1395	0.1032	0.1397	0.0293	0.0264	0.4526	0.2639
39	0.4855	0.2624	0.7053	0.8987	0.0531	0.0326	0.0190	0.0489	0.0295	0.0279	0.8355	0.5254

Because ODS Graphics is enabled, influence plots are displayed in Outputs 79.6.3 through 79.6.5. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information

about the graphics available in the LOGISTIC procedure, see the section "ODS Graphics" on page 6092. The vertical axis of an index plot represents the value of the diagnostic, and the horizontal axis represents the sequence (case number) of the observation. The index plots are useful for identification of extreme values.

The index plots of the Pearson residuals and the deviance residuals (Output 79.6.3) indicate that case 4 and case 18 are poorly accounted for by the model. The index plot of the diagonal elements of the hat matrix (Output 79.6.3) suggests that case 31 is an extreme point in the design space. The index plots of DFBETAS (Output 79.6.5) indicate that case 4 and case 18 are causing instability in all three parameter estimates. The other four index plots in Outputs 79.6.3 and 79.6.4 also point to these two cases as having a large impact on the coefficients and goodness of fit.

Output 79.6.3 Residuals, Hat Matrix, and CI Displacement C

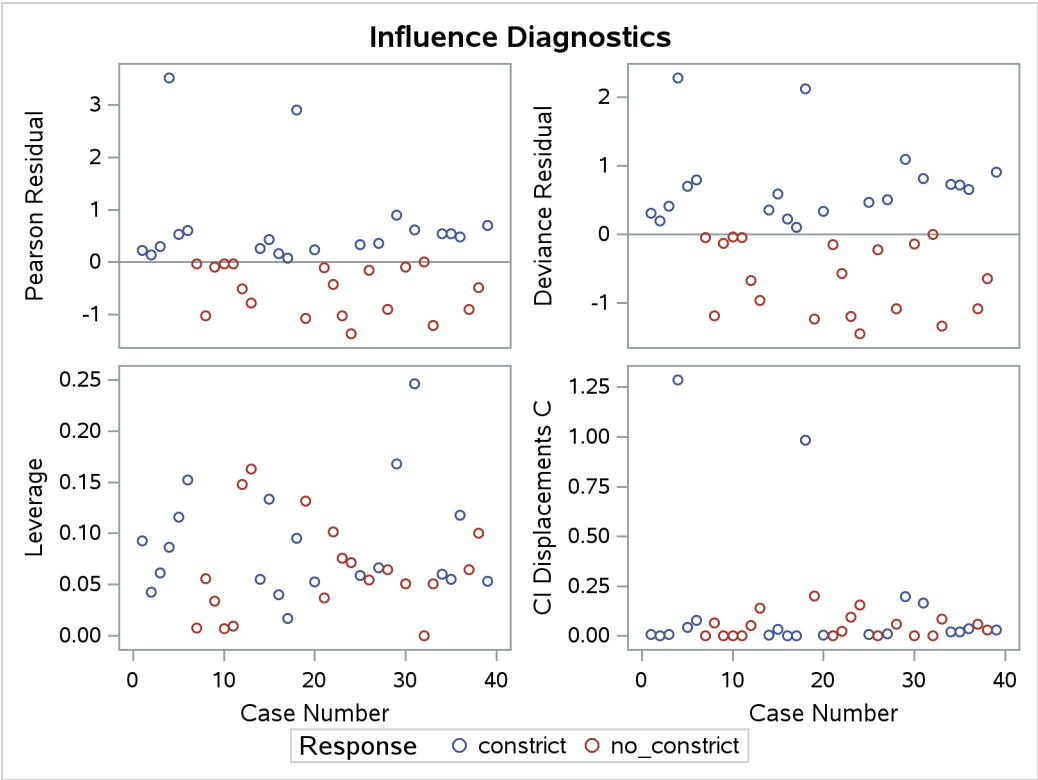

Output 79.6.4 CI Displacement CBar, Change in Deviance and Pearson Chi-Square

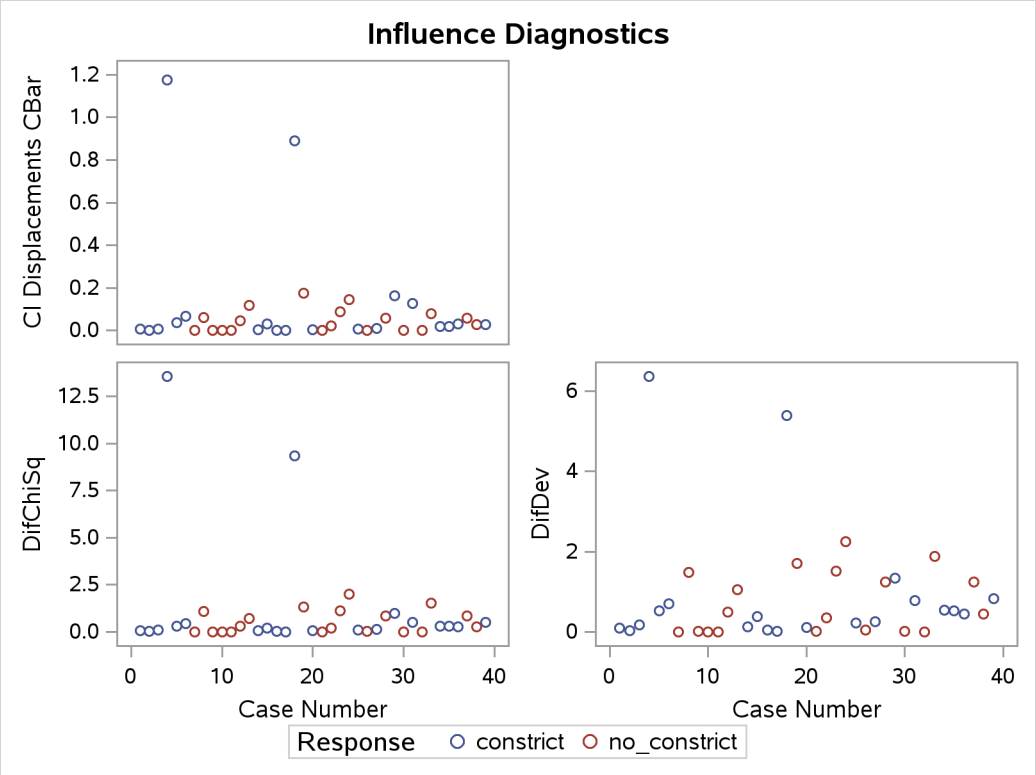

Output 79.6.5 DFBETAS Plots

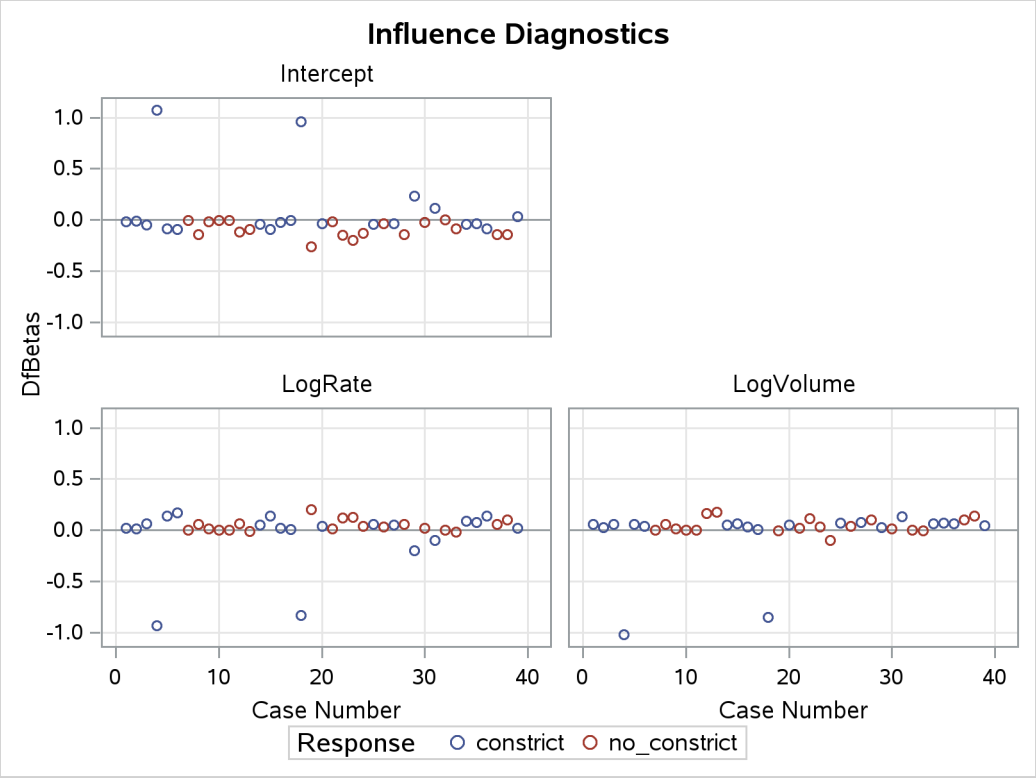

Other versions of diagnostic plots can be requested by specifying the appropriate options in the PLOTS= option. For example, the following statements produce three other sets of influence diagnostic plots: the PHAT option plots several diagnostics against the predicted probabilities (Output 79.6.6), the LEVERAGE option plots several diagnostics against the leverage (Output 79.6.7), and the DPC option plots the deletion diagnostics against the predicted probabilities and colors the observations according to the confidence interval displacement diagnostic (Output 79.6.8). The LABEL option displays the observation numbers on the plots. In all plots, you are looking for the outlying observations, and again cases 4 and 18 are noted.

```
proc logistic data=vaso plots(only label)=(phat leverage dpc);
   model Response=LogRate LogVolume;
run;
```

Output 79.6.6 Diagnostics versus Predicted Probability

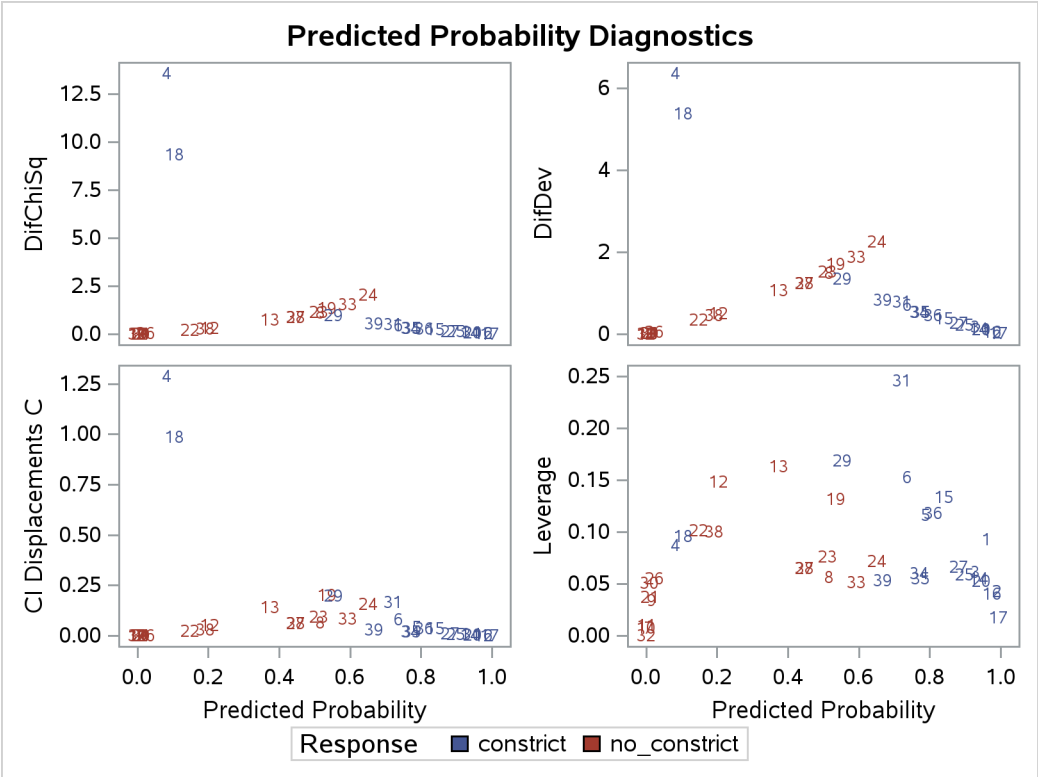

Output 79.6.7 Diagnostics versus Leverage

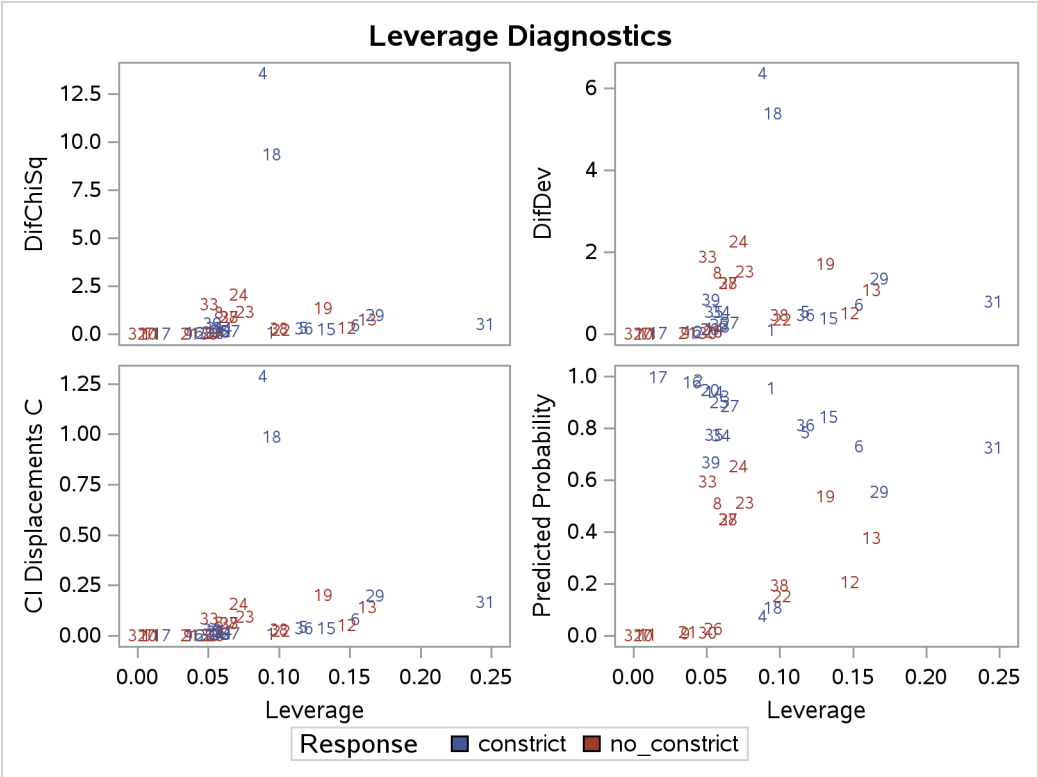

Output 79.6.8 Three Diagnostics

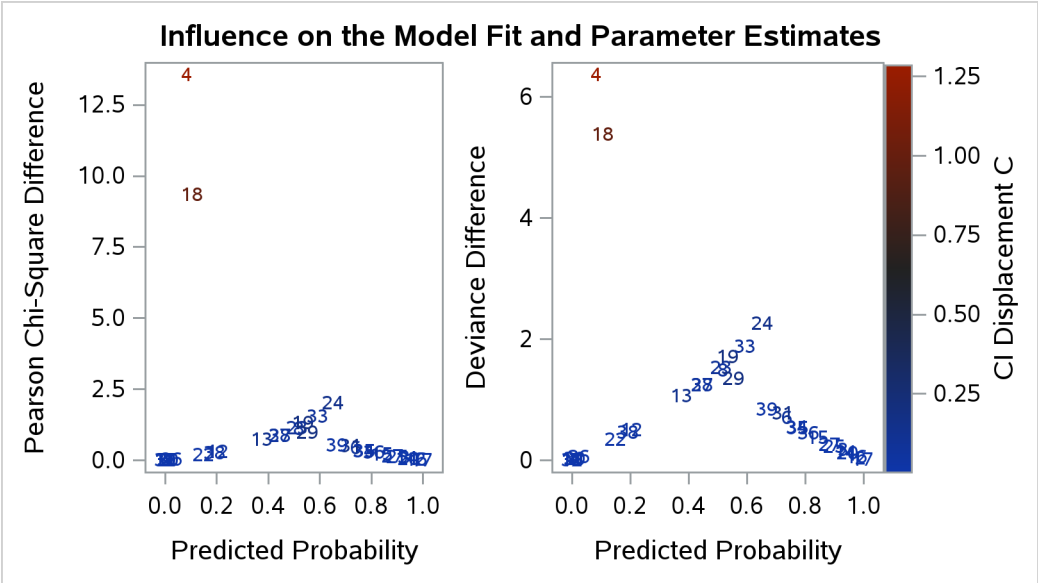

Example 79.7: ROC Curve, Customized Odds Ratios, Goodness-of-Fit Statistics, R-Square, and Confidence Limits

(View the complete code for this example (logiex7.sas) in the example repository.)

This example plots an ROC curve, estimates a customized odds ratio, produces the traditional goodness-of-fit analysis, displays the generalized R-square measures for the fitted model, calculates the normal confidence intervals for the regression parameters, and produces a display of the probability function and prediction curves for the fitted model. The data consist of three variables: n (number of subjects in the sample), disease (number of diseased subjects in the sample), and age (age for the sample). A linear logistic regression model is used to study the effect of age on the probability of contracting the disease. The statements to produce the data set and perform the analysis are as follows:

```
data Data1;
   input disease n age id $;
   datalines;
 0 14 25 a
 0 20 35 b
 0 19 45 c
 7 18 55 d
 6 12 65 e
17 17 75 f
;

ods graphics on;
%let _ROC_XAXISOPTS_LABEL=False Positive Fraction;
%let _ROC_YAXISOPTS_LABEL=True Positive Fraction;
proc logistic data=Data1 plots(only)=roc(id=id);
   model disease/n=age / scale=none
                         clparm=wald
                         clodds=pl
                         rsquare;
   units age=10;
   id id;
   effectplot;
run;
%symdel _ROC_XAXISOPTS_LABEL _ROC_YAXISOPTS_LABEL;
```

The option SCALE=NONE is specified to produce the deviance and Pearson goodness-of-fit analysis without adjusting for overdispersion. The RSQUARE option is specified to produce generalized R-square measures of the fitted model. The CLPARM=WALD option is specified to produce the Wald confidence intervals for the regression parameters. The UNITS statement is specified to produce customized odds ratio estimates for a change of 10 years in the age variable, and the CLODDS=PL option is specified to produce profile-likelihood confidence limits for the odds ratio. The PLOTS= option with ODS Graphics enabled produces a graphical display of the ROC curve with certain points labeled with their ID values, the two macro variables modify the ROC axis labels, and the EFFECTPLOT statement displays the model fit.

The results in Output 79.7.1 show that the deviance and Pearson statistics indicate no lack of fit in the model.

Output 79.7.1 Deviance and Pearson Goodness-of-Fit Analysis

The LOGISTIC Procedure

Deviance and Pearson Goodness-of-Fit Statistics				
Criterion	**Value**	**DF**	**Value/DF**	**Pr > ChiSq**
Deviance	7.7756	4	1.9439	0.1002
Pearson	6.6020	4	1.6505	0.1585

Number of events/trials observations: 6

Output 79.7.2 shows that the R-square for the model is 0.74. The odds of an event increases by a factor of 7.9 for each 10-year increase in age.

Output 79.7.2 R-Square, Confidence Intervals, and Customized Odds Ratio

Model Fit Statistics			
		Intercept and Covariates	
Criterion	**Intercept Only**	**Log Likelihood**	**Full Log Likelihood**
AIC	124.173	52.468	18.075
SC	126.778	57.678	23.285
-2 Log L	122.173	48.468	14.075

R-Square	0.5215	**Max-rescaled R-Square**	0.8925

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	73.7048	1	<.0001
Score	55.3274	1	<.0001
Wald	23.3475	1	<.0001

Analysis of Maximum Likelihood Estimates					
Parameter	**DF**	**Estimate**	**Standard Error**	**Wald Chi-Square**	**Pr > ChiSq**
Intercept	1	-12.5016	2.5555	23.9317	<.0001
age	1	0.2066	0.0428	23.3475	<.0001

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	92.6	**Somers' D**	0.906
Percent Discordant	2.0	**Gamma**	0.958
Percent Tied	5.4	**Tau-a**	0.384
Pairs	2100	**c**	0.953

Parameter Estimates and Wald Confidence Intervals			
Parameter	**Estimate**	**95% Confidence Limits**	
Intercept	-12.5016	-17.5104	-7.4929
age	0.2066	0.1228	0.2904

Output 79.7.2 *continued*

Odds Ratio Estimates and Profile-Likelihood Confidence Intervals				
Effect	**Unit**	**Estimate**	**95% Confidence Limits**	
age	10.0000	7.892	3.881	21.406

Because ODS Graphics is enabled, a graphical display of the ROC curve is produced as shown in Output 79.7.3.

Output 79.7.3 Receiver Operating Characteristic Curve

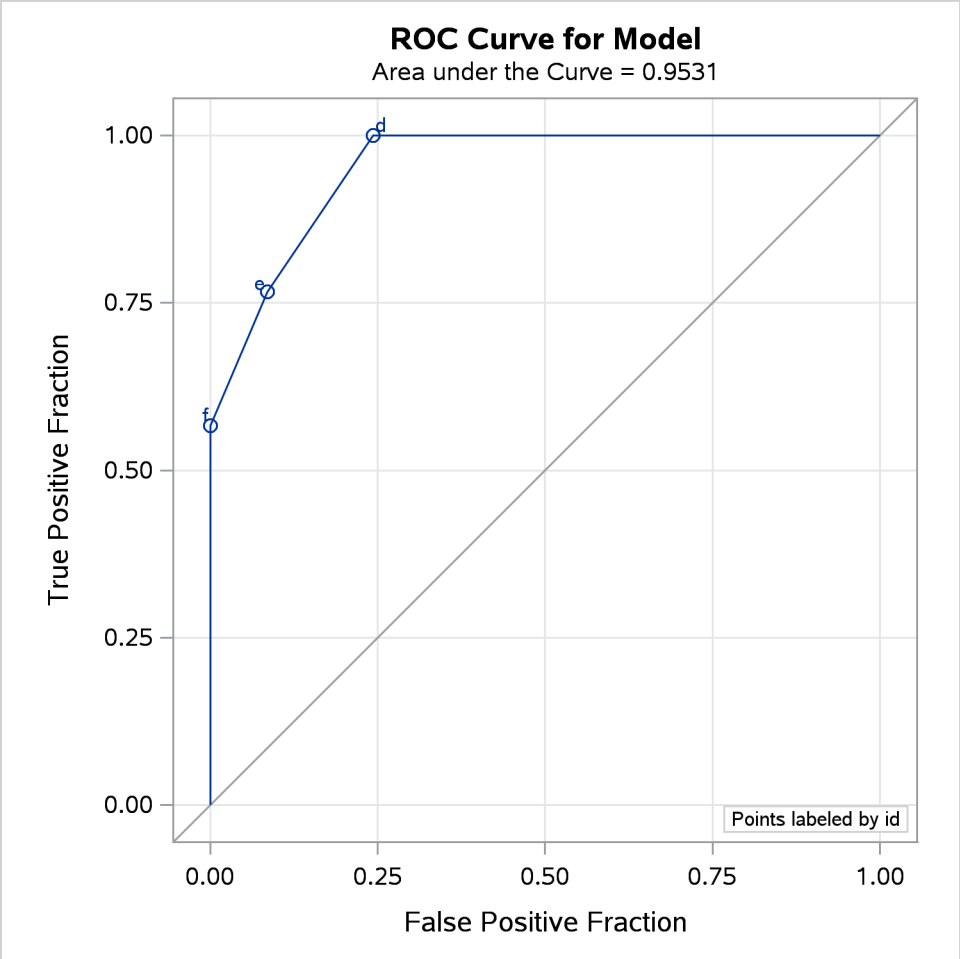

Note that the area under the ROC curve is estimated by the statistic c in the "Association of Predicted Probabilities and Observed Responses" table. In this example, the area under the ROC curve is 0.953. By default, the Y axis is labeled "Sensitivity" and the X axis is labeled "1–Specificity".

Because there is only one continuous covariate and because ODS Graphics is enabled, the EFFECTPLOT statement produces a graphical display of the predicted probability curve with bounding 95% confidence limits as shown in Output 79.7.4.

Output 79.7.4 Predicted Probability and 95% Prediction Limits

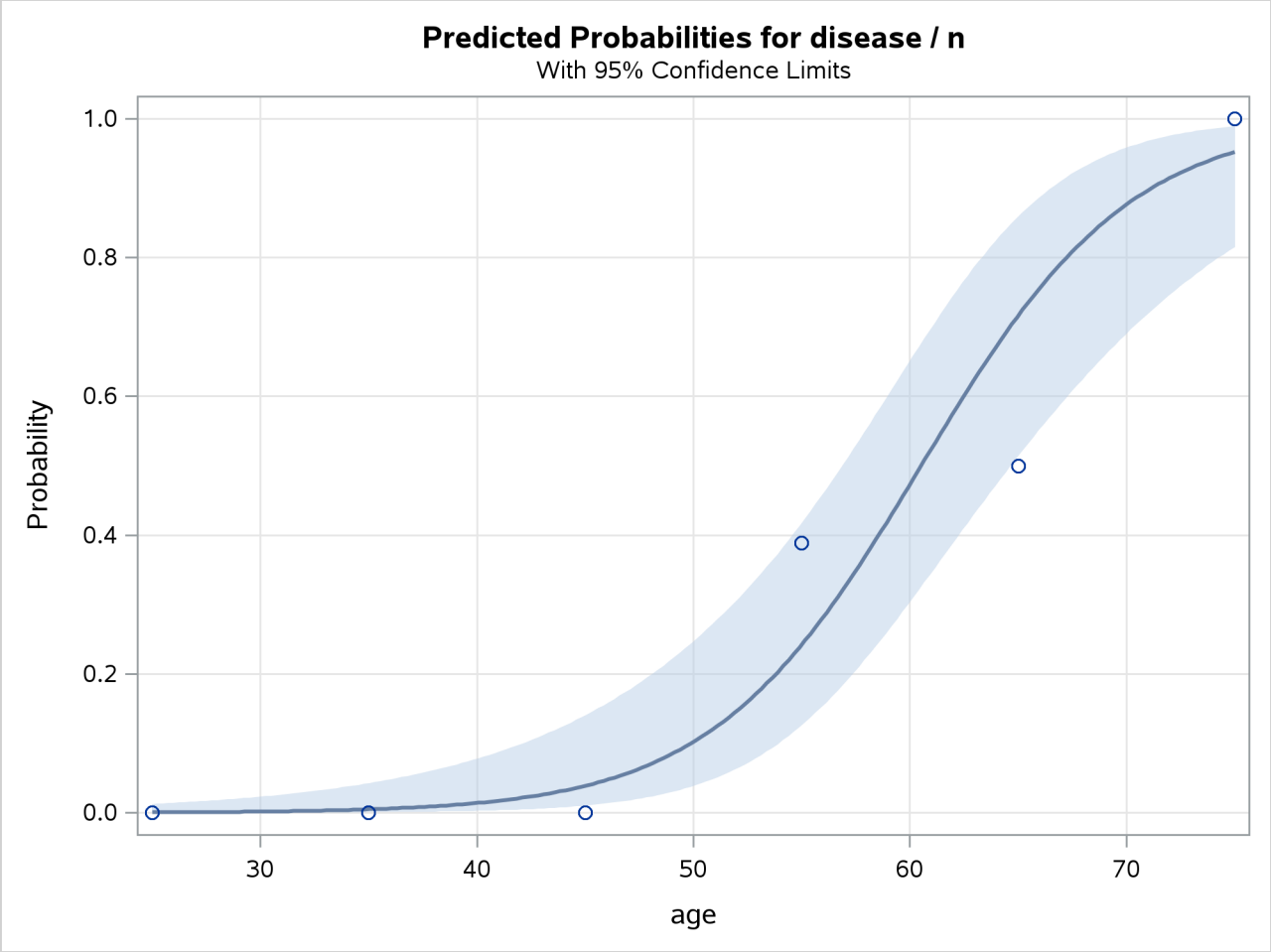

When you have more complicated effects, you can use the ODDSRATIO statement to produce customized odds ratios. In the following program, the Age variable is used to create the second-degree polynomial effect polyAge. The units of change are 1 year by default. Two odds ratio plots are produced: Output 79.7.5 displays the odds ratios for a 1-year change in age for individuals at ages 25 to 75; and Output 79.7.6 displays the odds ratio for a 1-year change in age for individuals at the mean age, 49.5.

```
proc logistic data=Data1 plots(only)=oddsratio;
   effect polyAge=polynomial(Age / degree=2);
   model disease/n=polyAge;
   oddsratio Age;
run;
```

Output 79.7.5 Customized Odds Ratios Computed at All Ages

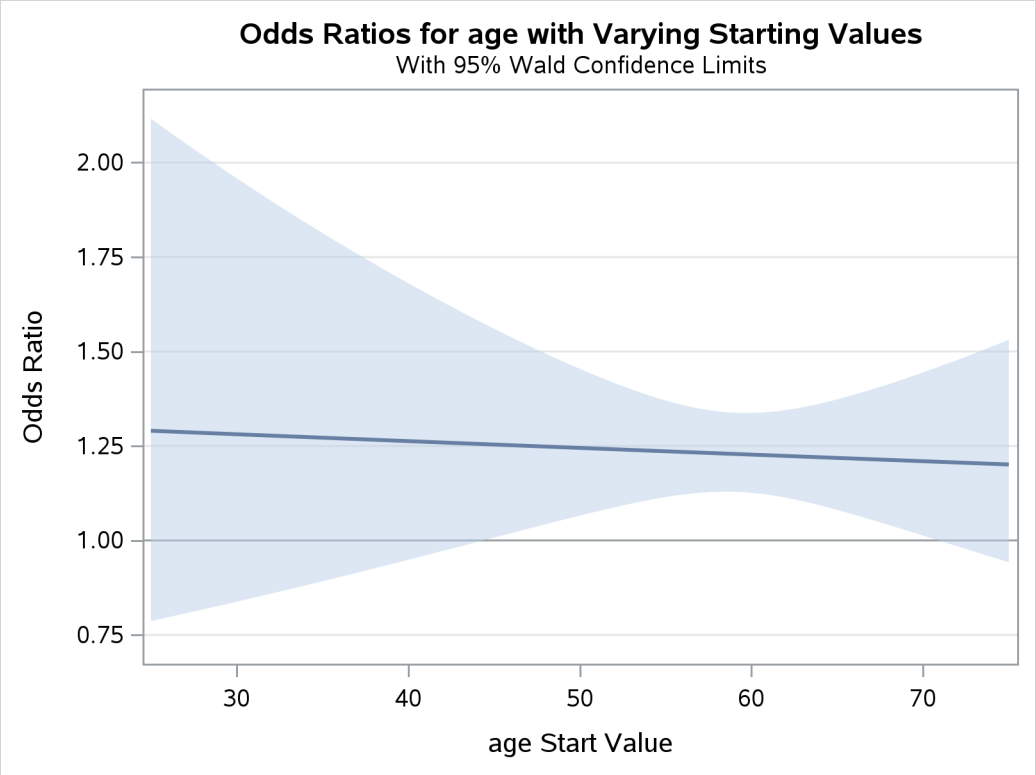

Output 79.7.6 Customized Odds Ratio Computed at the Mean Age

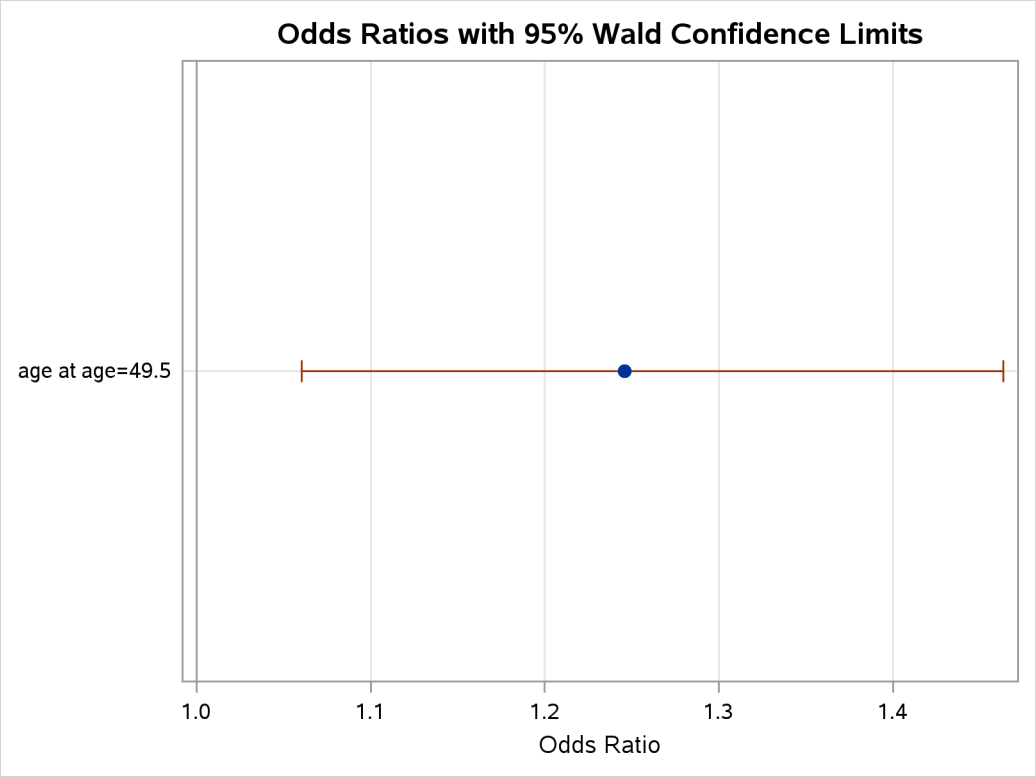

Example 79.8: Comparing Areas under Receiver Operating Characteristic Curves

(View the complete code for this example (logiex8.sas) in the example repository.)

DeLong, DeLong, and Clarke-Pearson (1988) report on 49 patients with ovarian cancer who also suffer from an intestinal obstruction. Three (correlated) screening tests are measured to determine whether a patient will benefit from surgery. The three tests are the K-G score and two measures of nutritional status: total protein and albumin. The data are as follows:

```
data roc;
   input alb tp totscore popind @@;
   totscore = 10 - totscore;
   datalines;
3.0 5.8 10 0   3.2 6.3  5 1   3.9 6.8  3 1   2.8 4.8  6 0
3.2 5.8  3 1   0.9 4.0  5 0   2.5 5.7  8 0   1.6 5.6  5 1
3.8 5.7  5 1   3.7 6.7  6 1   3.2 5.4  4 1   3.8 6.6  6 1
4.1 6.6  5 1   3.6 5.7  5 1   4.3 7.0  4 1   3.6 6.7  4 0
2.3 4.4  6 1   4.2 7.6  4 0   4.0 6.6  6 0   3.5 5.8  6 1
3.8 6.8  7 1   3.0 4.7  8 0   4.5 7.4  5 1   3.7 7.4  5 1
3.1 6.6  6 1   4.1 8.2  6 1   4.3 7.0  5 1   4.3 6.5  4 1
3.2 5.1  5 1   2.6 4.7  6 1   3.3 6.8  6 0   1.7 4.0  7 0
3.7 6.1  5 1   3.3 6.3  7 1   4.2 7.7  6 1   3.5 6.2  5 1
2.9 5.7  9 0   2.1 4.8  7 1   2.8 6.2  8 0   4.0 7.0  7 1
3.3 5.7  6 1   3.7 6.9  5 1   3.6 6.6  5 1
;
```

In the following statements, the NOFIT option is specified in the MODEL statement to prevent PROC LOGISTIC from fitting the model with three covariates. Each ROC statement lists one of the covariates, and PROC LOGISTIC then fits the model with that single covariate. Note that the original data set contains six more records with missing values for one of the tests, but PROC LOGISTIC ignores all records with missing values; hence there is a common sample size for each of the three models. The ROCCONTRAST statement implements the nonparametric approach of DeLong, DeLong, and Clarke-Pearson (1988) to compare the three ROC curves, the REFERENCE option specifies that the K-G Score curve is used as the reference curve in the contrast, the E option displays the contrast coefficients, and the ESTIMATE option computes and tests each comparison. With ODS Graphics enabled, the **plots=roc(id=prob)** specification in the PROC LOGISTIC statement displays several plots, and the plots of individual ROC curves have certain points labeled with their predicted probabilities.

```
ods graphics on;
proc logistic data=roc plots=roc(id=prob);
   model popind(event='0') = alb tp totscore / nofit;
   roc 'Albumin' alb;
   roc 'K-G Score' totscore;
   roc 'Total Protein' tp;
   roccontrast reference('K-G Score') / estimate e;
run;
```

The initial model information is displayed in Output 79.8.1.

Output 79.8.1 Initial LOGISTIC Output

The LOGISTIC Procedure

Model Information	
Data Set	WORK.ROC
Response Variable	popind
Number of Response Levels	2
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	43
Number of Observations Used	43

Response Profile		
Ordered Value	popind	Total Frequency
1	0	12
2	1	31

Probability modeled is popind=0.

Score Test for Global Null Hypothesis		
Chi-Square	DF	Pr > ChiSq
10.7939	3	0.0129

For each ROC model, the model fitting details in Outputs 79.8.2, 79.8.4, and 79.8.6 can be suppressed with the ROCOPTIONS(NODETAILS) option; however, the convergence status is always displayed.

The ROC curves for the three models are displayed in Outputs 79.8.3, 79.8.5, and 79.8.7. Note that the labels on the ROC curve are produced by specifying the ID=PROB option, and are the predicted probabilities for the cutpoints.

Output 79.8.2 Fit Tables for Popind=Alb

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	52.918	49.384
SC	54.679	52.907
-2 Log L	50.918	45.384

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	5.5339	1	0.0187
Score	5.6893	1	0.0171
Wald	4.6869	1	0.0304

Output 79.8.2 *continued*

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	2.4646	1.5913	2.3988	0.1214
alb	1	-1.0520	0.4859	4.6869	0.0304

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
alb	0.349	0.135	0.905

Output 79.8.3 ROC Curve for Popind=Alb

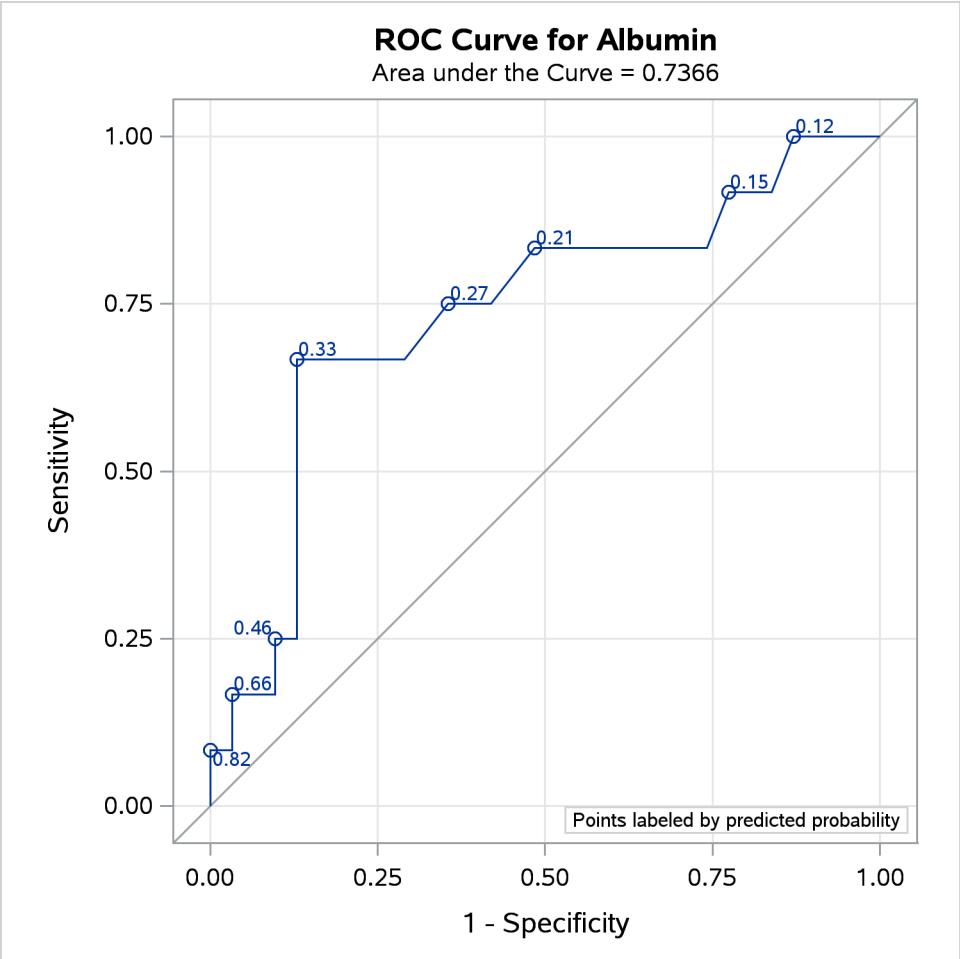

Output 79.8.4 Fit Tables for Popind=Totscore

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Output 79.8.4 *continued*

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	52.918	46.262
SC	54.679	49.784
-2 Log L	50.918	42.262

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	8.6567	1	0.0033
Score	8.3613	1	0.0038
Wald	6.3845	1	0.0115

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	2.1542	1.2477	2.9808	0.0843
totscore	1	-0.7696	0.3046	6.3845	0.0115

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
totscore	0.463	0.255	0.841

Output 79.8.5 ROC Curve for Popind=Totscore

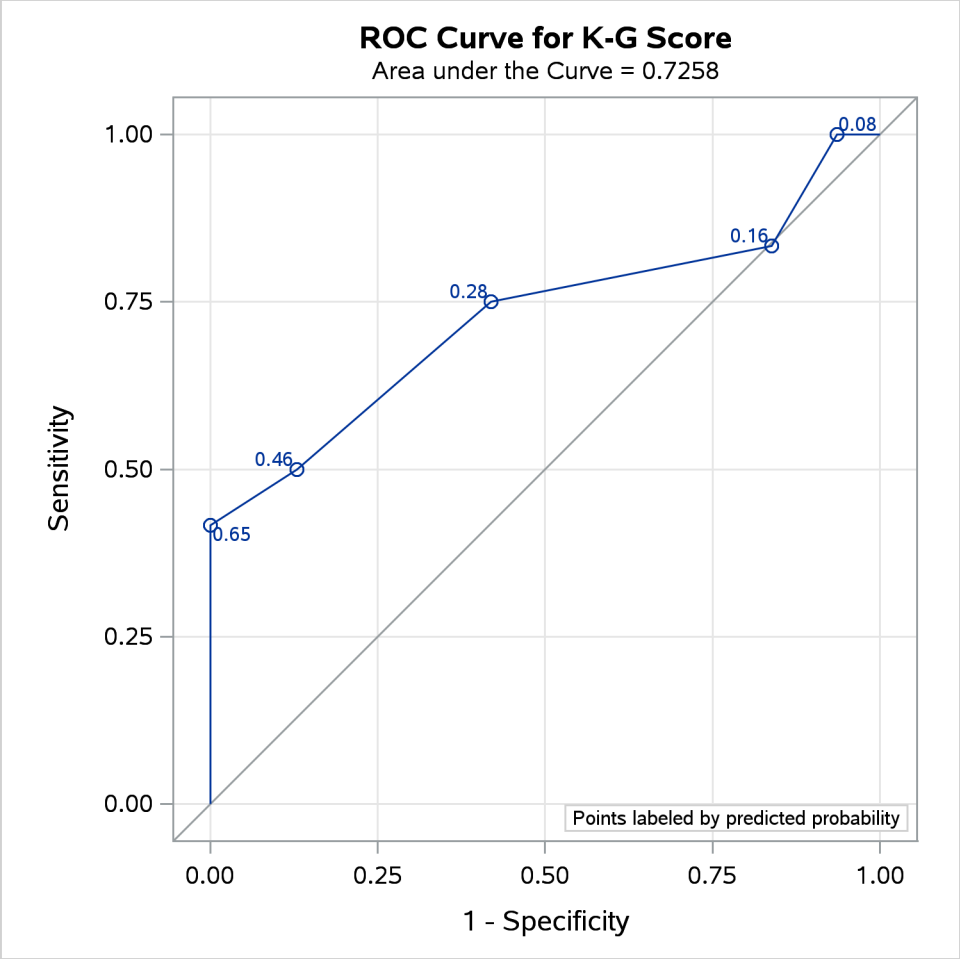

Output 79.8.6 Fit Tables for Popind=Tp

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	52.918	51.794
SC	54.679	55.316
-2 Log L	50.918	47.794

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	3.1244	1	0.0771
Score	3.1123	1	0.0777
Wald	2.9059	1	0.0883

Output 79.8.6 *continued*

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	2.8295	2.2065	1.6445	0.1997
tp	1	-0.6279	0.3683	2.9059	0.0883

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
tp	0.534	0.259	1.099

Output 79.8.7 ROC Curve for Popind=Tp

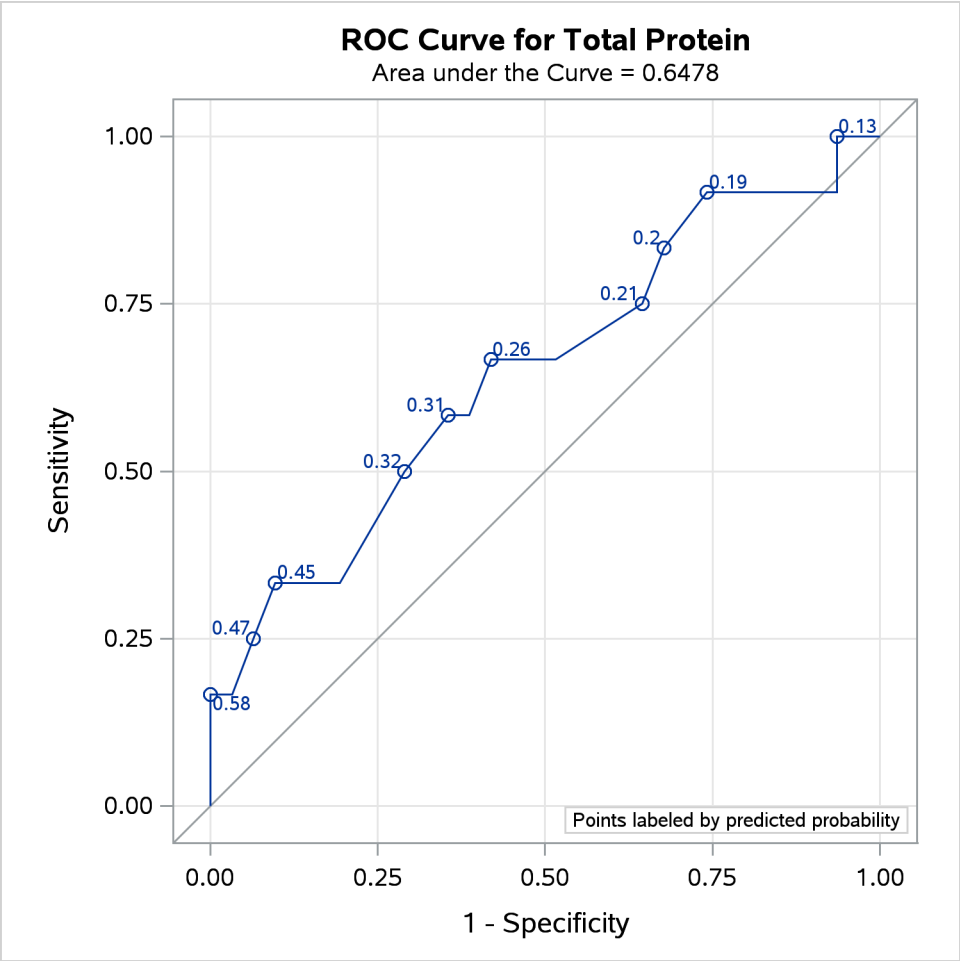

All ROC curves being compared are also overlaid on the same plot, as shown in Output 79.8.8.

Output 79.8.8 Overlay of All Models Being Compared

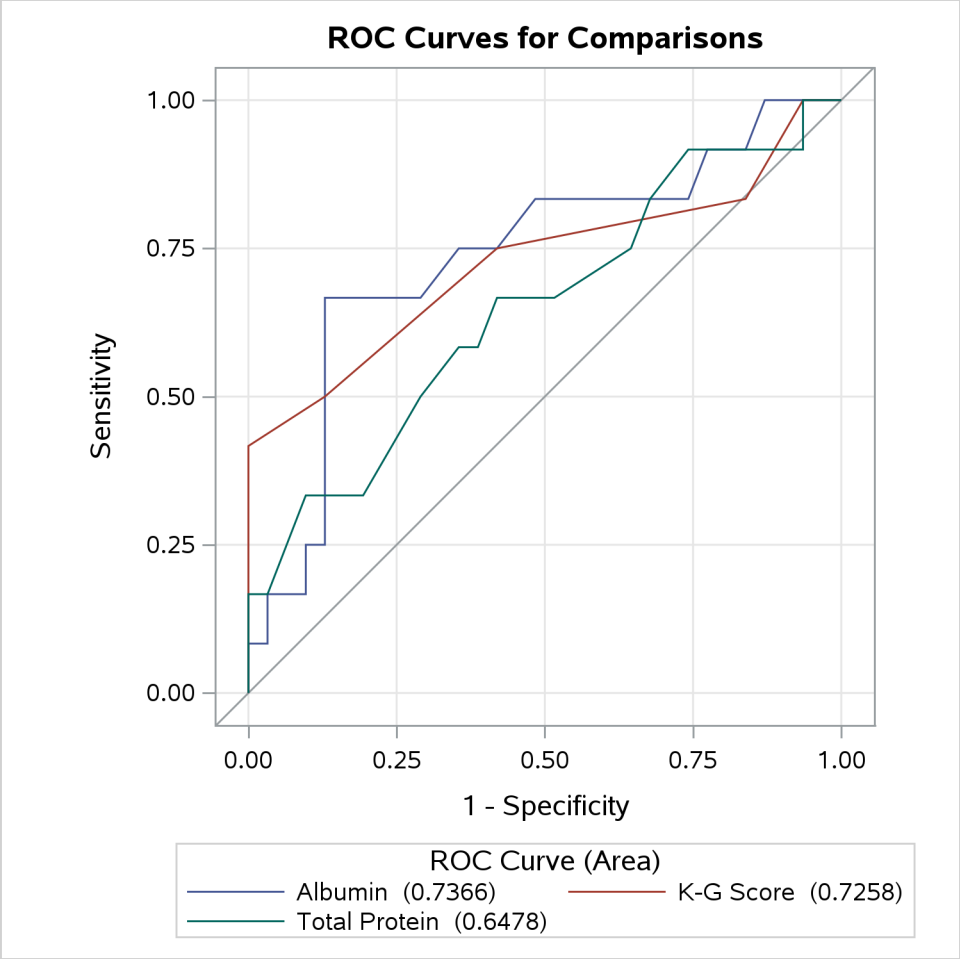

Output 79.8.9 displays the association statistics, and displays the area under the ROC curve along with its standard error and a confidence interval for each model in the comparison. The confidence interval for Total Protein contains 0.50; hence it is not significantly different from random guessing, which is represented by the diagonal line in the preceding ROC plots.

Output 79.8.9 ROC Association Table

ROC Association Statistics							
		Mann-Whitney					
ROC Model	Area	Standard Error	95% Wald Confidence Limits		Somers' D	Gamma	Tau-a
Albumin	0.7366	0.0927	0.5549	0.9182	0.4731	0.4809	0.1949
K-G Score	0.7258	0.1028	0.5243	0.9273	0.4516	0.5217	0.1860
Total Protein	0.6478	0.1000	0.4518	0.8439	0.2957	0.3107	0.1218

Output 79.8.10 shows that the contrast used 'K-G Score' as the reference level. This table is produced by specifying the E option in the ROCCONTRAST statement.

Output 79.8.10 ROC Contrast Coefficients

ROC Contrast Coefficients		
ROC Model	Row1	Row2
Albumin	1	0
K-G Score	-1	-1
Total Protein	0	1

Output 79.8.11 shows that the 2-degrees-of-freedom test that the 'K-G Score' is different from at least one other test is not significant at the 0.05 level.

Output 79.8.11 ROC Test Results (2 Degrees of Freedom)

ROC Contrast Test Results			
Contrast	DF	Chi-Square	Pr > ChiSq
Reference = K-G Score	2	2.5340	0.2817

Output 79.8.12 is produced by specifying the ESTIMATE option in the ROCCONTRAST statement. Each row shows that the curves are not significantly different.

Output 79.8.12 ROC Contrast Row Estimates (1-Degree-of-Freedom Tests)

ROC Contrast Estimation and Testing Results by Row						
		Standard	95% Wald			
Contrast	Estimate	Error	Confidence Limits		Chi-Square	Pr > ChiSq
Albumin - K-G Score	0.0108	0.0953	-0.1761	0.1976	0.0127	0.9102
Total Protein - K-G Score	-0.0780	0.1046	-0.2830	0.1271	0.5554	0.4561

Example 79.9: Goodness-of-Fit Tests and Subpopulations

(View the complete code for this example (logiex9.sas) in the example repository.)

A study is done to investigate the effects of two binary factors, A and B, on a binary response, Y. Subjects are randomly selected from subpopulations defined by the four possible combinations of levels of A and B. The number of subjects responding with each level of Y is recorded, and the following DATA step creates the data set One:

```
data One;
   do A=0,1;
      do B=0,1;
         do Y=1,2;
            input F @@;
            output;
         end;
      end;
   end;
   datalines;
23 63 31 70 67 100 70 104
;
```

The following statements fit a full model to examine the main effects of A and B as well as the interaction effect of A and B:

```
proc logistic data=One;
   freq F;
   model Y=A B A*B;
run;
```

Results of the model fit are shown in Output 79.9.1. Notice that neither the A*B interaction nor the B main effect is significant.

Output 79.9.1 Full Model Fit

The LOGISTIC Procedure

Model Information	
Data Set	WORK.ONE
Response Variable	Y
Number of Response Levels	2
Frequency Variable	F
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	8
Number of Observations Used	8
Sum of Frequencies Read	528
Sum of Frequencies Used	528

Response Profile

Ordered Value	Y	Total Frequency
1	1	191
2	2	337

Probability modeled is Y=1.

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	693.061	691.914
SC	697.330	708.990
-2 Log L	691.061	683.914

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	7.1478	3	0.0673
Score	6.9921	3	0.0721
Wald	6.9118	3	0.0748

Output 79.9.1 *continued*

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-1.0074	0.2436	17.1015	<.0001
A	1	0.6069	0.2903	4.3714	0.0365
B	1	0.1929	0.3254	0.3515	0.5533
A*B	1	-0.1883	0.3933	0.2293	0.6321

Pearson and deviance goodness-of-fit tests cannot be obtained for this model because a full model containing four parameters is fit, leaving no residual degrees of freedom. For a binary response model, the goodness-of-fit tests have $m - q$ degrees of freedom, where m is the number of subpopulations and q is the number of model parameters. In the preceding model, $m = q = 4$, resulting in zero degrees of freedom for the tests.

The following statements fit a reduced model containing only the A effect, so two degrees of freedom become available for testing goodness of fit. Specifying the SCALE=NONE option requests the Pearson and deviance statistics. With single-trial syntax, the AGGREGATE= option is needed to define the subpopulations in the study. Specifying AGGREGATE=(A B) creates subpopulations of the four combinations of levels of A and B. Although the B effect is being dropped from the model, it is still needed to define the original subpopulations in the study. If AGGREGATE=(A) were specified, only two subpopulations would be created from the levels of A, resulting in $m = q = 2$ and zero degrees of freedom for the tests.

```
proc logistic data=One;
   freq F;
   model Y=A / scale=none aggregate=(A B);
run;
```

The goodness-of-fit tests in Output 79.9.2 show that dropping the B main effect and the A*B interaction simultaneously does not result in significant lack of fit of the model. The tests' large p-values indicate insufficient evidence for rejecting the null hypothesis that the model fits.

Output 79.9.2 Reduced Model Fit

The LOGISTIC Procedure

Deviance and Pearson Goodness-of-Fit Statistics				
Criterion	Value	DF	Value/DF	Pr > ChiSq
Deviance	0.3541	2	0.1770	0.8377
Pearson	0.3531	2	0.1765	0.8382

Number of unique profiles: 4

Example 79.10: Overdispersion

(View the complete code for this example (logiex10.sas) in the example repository.)

In a seed germination test, seeds of two cultivars were planted in pots of two soil conditions. The following statements create the data set seeds, which contains the observed proportion of seeds that germinated for various combinations of cultivar and soil condition. The variable n represents the number of seeds planted in a pot, and the variable r represents the number germinated. The indicator variables cult and soil represent the cultivar and soil condition, respectively.

```
data seeds;
   input pot n r cult soil;
   datalines;
 1 16     8      0       0
 2 51    26      0       0
 3 45    23      0       0
 4 39    10      0       0
 5 36     9      0       0
 6 81    23      1       0
 7 30    10      1       0
 8 39    17      1       0
 9 28     8      1       0
10 62    23      1       0
11 51    32      0       1
12 72    55      0       1
13 41    22      0       1
14 12     3      0       1
15 13    10      0       1
16 79    46      1       1
17 30    15      1       1
18 51    32      1       1
19 74    53      1       1
20 56    12      1       1
;
```

PROC LOGISTIC is used as follows to fit a logit model to the data, with cult, soil, and cult × soil interaction as explanatory variables. The option SCALE=NONE is specified to display goodness-of-fit statistics.

```
proc logistic data=seeds;
   model r/n=cult soil cult*soil/scale=none;
   title 'Full Model With SCALE=NONE';
run;
```

Results of fitting the full factorial model are shown in Output 79.10.1. Both Pearson χ^2 and deviance are highly significant ($p < 0.0001$), suggesting that the model does not fit well.

Output 79.10.1 Results of the Model Fit for the Two-Way Layout

Full Model With SCALE=NONE

The LOGISTIC Procedure

Deviance and Pearson Goodness-of-Fit Statistics				
Criterion	Value	DF	Value/DF	Pr > ChiSq
Deviance	68.3465	16	4.2717	<.0001
Pearson	66.7617	16	4.1726	<.0001

Number of events/trials observations: 20

Model Fit Statistics			
		Intercept and Covariates	
Criterion	Intercept Only	Log Likelihood	Full Log Likelihood
AIC	1256.852	1213.003	156.533
SC	1261.661	1232.240	175.769
-2 Log L	1254.852	1205.003	148.533

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	49.8488	3	<.0001
Score	49.1682	3	<.0001
Wald	47.7623	3	<.0001

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-0.3788	0.1489	6.4730	0.0110
cult	1	-0.2956	0.2020	2.1412	0.1434
soil	1	0.9781	0.2128	21.1234	<.0001
cult*soil	1	-0.1239	0.2790	0.1973	0.6569

If the link function and the model specification are correct and if there are no outliers, then the lack of fit might be due to overdispersion. Without adjusting for the overdispersion, the standard errors are likely to be underestimated, causing the Wald tests to be too sensitive. In PROC LOGISTIC, there are three SCALE= options to accommodate overdispersion. With unequal sample sizes for the observations, SCALE=WILLIAMS is preferred. The Williams model estimates a scale parameter ϕ by equating the value of Pearson χ^2 for the full model to its approximate expected value. The full model considered in the following statements is the model with cultivar, soil condition, and their interaction. Using a full model reduces the risk of contaminating ϕ with lack of fit due to incorrect model specification.

```
proc logistic data=seeds;
   model r/n=cult soil cult*soil / scale=williams;
   title 'Full Model With SCALE=WILLIAMS';
run;
```

Results of using Williams' method are shown in Output 79.10.2. The estimate of ϕ is 0.075941 and is given in the formula for the Weight Variable at the beginning of the displayed output.

Output 79.10.2 Williams' Model for Overdispersion

Full Model With SCALE=WILLIAMS

The LOGISTIC Procedure

Model Information	
Data Set	WORK.SEEDS
Response Variable (Events)	r
Response Variable (Trials)	n
Weight Variable	1 / (1 + 0.075941 * (n - 1))
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	20
Number of Observations Used	20
Sum of Frequencies Read	906
Sum of Frequencies Used	906
Sum of Weights Read	198.3216
Sum of Weights Used	198.3216

Response Profile

Ordered Value	Binary Outcome	Total Frequency	Total Weight
1	Event	437	92.95346
2	Nonevent	469	105.36819

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Deviance and Pearson Goodness-of-Fit Statistics

Criterion	Value	DF	Value/DF	Pr > ChiSq
Deviance	16.4402	16	1.0275	0.4227
Pearson	16.0000	16	1.0000	0.4530

Number of events/trials observations: 20

Note: Since the Williams method was used to accommodate overdispersion, the Pearson chi-squared statistic and the deviance can no longer be used to assess the goodness of fit of the model.

Model Fit Statistics

		Intercept and Covariates	
Criterion	Intercept Only	Log Likelihood	Full Log Likelihood
AIC	276.155	273.586	44.579
SC	280.964	292.822	63.815
-2 Log L	274.155	265.586	36.579

Output 79.10.2 *continued*

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	8.5687	3	0.0356
Score	8.4856	3	0.0370
Wald	8.3069	3	0.0401

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-0.3926	0.2932	1.7932	0.1805
cult	1	-0.2618	0.4160	0.3963	0.5290
soil	1	0.8309	0.4223	3.8704	0.0491
cult*soil	1	-0.0532	0.5835	0.0083	0.9274

Because neither cult nor cult × soil is statistically significant ($p = 0.5290$ and $p = 0.9274$, respectively), a reduced model that contains only the soil condition factor is fitted, with the observations weighted by $1/(1 + 0.075941(N - 1))$. This can be done conveniently in PROC LOGISTIC by including the scale estimate in the SCALE=WILLIAMS option as follows:

```
proc logistic data=seeds;
   model r/n=soil / scale=williams(0.075941);
   title 'Reduced Model With SCALE=WILLIAMS(0.075941)';
run;
```

Results of the reduced model fit are shown in Output 79.10.3. Soil condition remains a significant factor ($p = 0.0064$) for the seed germination.

Output 79.10.3 Reduced Model with Overdispersion Controlled

Reduced Model With SCALE=WILLIAMS(0.075941)

The LOGISTIC Procedure

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-0.5249	0.2076	6.3949	0.0114
soil	1	0.7910	0.2902	7.4284	0.0064

Example 79.11: Conditional Logistic Regression for Matched Pairs Data

(View the complete code for this example (logiex11.sas) in the example repository.)

In matched pairs, or *case-control*, studies, conditional logistic regression is used to investigate the relationship between an outcome of being an event (case) or a nonevent (control) and a set of prognostic factors.

The following data are a subset of the data from the Los Angeles Study of the Endometrial Cancer Data in Breslow and Day (1980). There are 63 matched pairs, each consisting of a case of endometrial cancer (Outcome=1) and a control (Outcome=0). The case and corresponding control have the same ID. Two prognostic factors are included: Gall (an indicator variable for gall bladder disease) and Hyper (an indicator variable for hypertension). The goal of the case-control analysis is to determine the relative risk for gall bladder disease, controlling for the effect of hypertension.

```
data Data1;
   do ID=1 to 63;
      do Outcome = 1 to 0 by -1;
         input Gall Hyper @@;
         output;
      end;
   end;
   datalines;
0 0  0 0    0 0  0 0    0 1  0 1    0 0  1 0    1 0  0 1
0 1  0 0    1 0  0 0    1 1  0 1    0 0  0 0    0 0  0 0
1 0  0 0    0 0  0 1    1 0  0 1    1 0  1 0    1 0  0 1
0 1  0 0    0 0  1 1    0 0  1 1    0 0  0 1    0 1  0 0
0 0  1 1    0 1  0 1    0 1  0 0    0 0  0 0    0 0  0 0
0 0  0 1    1 0  0 1    0 0  0 1    1 0  0 0    0 1  0 0
0 1  0 0    0 1  0 0    0 1  0 0    0 0  0 0    1 1  1 1
0 0  0 1    0 1  0 0    0 1  0 1    0 1  0 1    0 1  0 0
0 0  0 0    0 1  1 0    0 0  0 1    0 0  0 0    1 0  0 0
0 0  0 0    1 1  0 0    0 1  0 0    0 0  0 0    0 1  0 1
0 0  0 0    0 1  0 1    0 1  0 0    0 1  0 0    1 0  0 0
0 0  0 0    1 1  1 0    0 0  0 0    0 0  0 0    1 1  0 0
1 0  1 0    0 1  0 0    1 0  0 0
;
```

There are several ways to approach this problem with PROC LOGISTIC:

- Specify the STRATA statement to perform a conditional logistic regression.
- Specify EXACT and STRATA statements to perform an exact logistic regression on the original data set, if you believe the data set is too small or too sparse for the usual asymptotics to hold.
- Transform each matched pair into a single observation, and then specify a PROC LOGISTIC statement on this transformed data without a STRATA statement; this also performs a conditional logistic regression and produces essentially the same results.
- Specify an EXACT statement on the transformed data.

SAS statements and selected results for the first two approaches are given in the remainder of this example.

Conditional Analysis Using the STRATA Statement

In the following statements, PROC LOGISTIC is invoked with the ID variable declared in the STRATA statement to obtain the conditional logistic model estimates for a model containing Gall as the only predictor variable:

```
proc logistic data=Data1;
   strata ID;
   model outcome(event='1')=Gall;
run;
```

Results from the conditional logistic analysis are shown in Output 79.11.1. Note that there is no intercept term in the "Analysis of Maximum Likelihood Estimates" tables.

The odds ratio estimate for Gall is 2.60, which is marginally significant ($p = 0.0694$) and which is an estimate of the relative risk for gall bladder disease. A 95% confidence interval for this relative risk is (0.927, 7.293).

Output 79.11.1 Conditional Logistic Regression (Gall as Risk Factor)

The LOGISTIC Procedure

Conditional Analysis

Model Information	
Data Set	WORK.DATA1
Response Variable	Outcome
Number of Response Levels	2
Number of Strata	63
Model	binary logit
Optimization Technique	Newton-Raphson ridge

Number of Observations Read	126
Number of Observations Used	126
Number of Observations Informative	126

Response Profile

Ordered Value	Outcome	Total Frequency
1	0	63
2	1	63

Probability modeled is Outcome=1.

Strata Summary

Response Pattern	Outcome 0	Outcome 1	Number of Strata	Frequency
1	1	1	63	126

Newton-Raphson Ridge Optimization

Without Parameter Scaling

Convergence criterion (GCONV=1E-8) satisfied.

Output 79.11.1 *continued*

Model Fit Statistics		
Criterion	Without Covariates	With Covariates
AIC	87.337	85.654
SC	87.337	88.490
-2 Log L	87.337	83.654

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	3.6830	1	0.0550
Score	3.5556	1	0.0593
Wald	3.2970	1	0.0694

Analysis of Conditional Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Gall	1	0.9555	0.5262	3.2970	0.0694

Odds Ratio Estimates			
Effect	Point Estimate	95% Wald Confidence Limits	
Gall	2.600	0.927	7.293

Exact Analysis Using the STRATA Statement

When you believe there are not enough data or that the data are too sparse, you can perform a stratified exact logistic regression. The following statements perform stratified exact logistic regressions on the original data set by specifying both the STRATA and EXACT statements:

```
proc logistic data=Data1 exactonly;
   strata ID;
   model outcome(event='1')=Gall;
   exact Gall / estimate=both;
run;
```

Output 79.11.2 Exact Logistic Regression (Gall as Risk Factor)

The LOGISTIC Procedure

Exact Conditional Analysis

Exact Conditional Tests				
			p-Value	
Effect	Test	Statistic	Exact	Mid
Gall	Score	3.5556	0.0963	0.0799
	Probability	0.0327	0.0963	0.0799

Output 79.11.2 *continued*

Exact Parameter Estimates					
Parameter	Estimate	Standard Error	95% Confidence Limits		p-Value
Gall	0.9555	0.5262	-0.1394	2.2316	0.0963

Exact Odds Ratios				
Parameter	Estimate	95% Confidence Limits		p-Value
Gall	2.600	0.870	9.315	0.0963

Note that the score statistic in the "Conditional Exact Tests" table in Output 79.11.2 is identical to the score statistic in Output 79.11.1 from the conditional analysis. The exact odds ratio confidence interval is much wider than its conditional analysis counterpart, but the parameter estimates are similar. The exact analysis confirms the marginal significance of Gall as a predictor variable.

Example 79.12: Exact Conditional Logistic Regression

(View the complete code for this example (logiex12.sas) in the example repository.)

The following data, from Hand et al. (1994), contain the results of a study of 49 anxious or depressed children. The Diagnosis variable indicates whether the child was anxious or depressed when the study began, the Friendships variable indicates whether the child has good friendships, the Total variable represents the total number of children in the study who exhibit the specified values for Diagnosis and Friendships, and the Recovered variable represents the number of children whose mothers believe that their child has recovered at the end of the study.

```
data one;
   length Diagnosis $ 9;
   input Diagnosis $ Friendships $ Recovered Total @@;
   datalines;
Anxious   Poor 0 0    Anxious   Good 13 21
Depressed Poor 0 8    Depressed Good 15 20
;
```

Notice that no children in the study are both anxious and have poor friendships and that no children who are depressed and have poor friendships have recovered. The following statements fit an unconditional logistic regression model to these data.

```
proc logistic data=one;
   class Diagnosis Friendships / param=ref;
   model Recovered/Total = Diagnosis Friendships;
run;
```

Because the data set has quasi-complete separation, the unconditional logistic regression results are not reliable and Output 79.12.1 is displayed.

Output 79.12.1 Unconditional Logistic Regression Results

The LOGISTIC Procedure

Model Convergence Status
Quasi-complete separation of data points detected.

The sparseness of the data and the separability of the data set make this a good candidate for an exact logistic regression. In the following code, the EXACTONLY option suppresses the unconditional logistic regression results, the EXACT statement requests an exact analysis of the two covariates, the OUTDIST= option outputs the exact distribution into a SAS data set, the JOINT option computes a joint test for the significance of the two covariates in the model, and the ESTIMATE option produces parameter estimates for the two covariates.

```
proc logistic data=one exactonly;
   class Diagnosis Friendships / param=ref;
   model Recovered/Total = Diagnosis Friendships;
   exact Diagnosis Friendships
       / outdist=dist joint estimate;
run;

proc print data=dist(obs=10);
run;

proc print data=dist(firstobs=162 obs=175);
run;

proc print data=dist(firstobs=176 obs=184);
run;
```

Tests for the joint significance of the Diagnosis and Friendships covariates are labeled "Joint" in Output 79.12.2. Both of these tests reject the null hypothesis that all the parameters are identically zero.

Output 79.12.2 Exact Tests

The LOGISTIC Procedure

Exact Conditional Analysis

Exact Conditional Tests				
			p-Value	
Effect	Test	Statistic	Exact	Mid
Joint	Score	13.1905	0.0008	0.0008
	Probability	0.000081	0.0007	0.0007
Diagnosis	Score	0.7915	0.5055	0.4159
	Probability	0.1791	0.5055	0.4159
Friendships	Score	12.4615	0.0004	0.0002
	Probability	0.000414	0.0004	0.0002

In Output 79.12.2, the joint probability statistic (0.000081) is the probability of the observed sufficient statistics (Friendships=28 and Diagnosis=13) in the joint exact distribution (part of which is displayed in Output 79.12.3). Note that the joint exact distribution has sufficient statistics displayed for both covariates; the marginal distributions have sufficient statistics displayed for only one covariate. The associated exact p-value (0.0007) in Output 79.12.2 is the sum of the probabilities of all sufficient statistics in the joint exact

distribution that have probabilities less than or equal to 0.000081. The mid-p-value (0.0007) adjusts the exact p-value for the discreteness of the exact distribution by subtracting half the probability of the observed sufficient statistic (Hirji 2006). The score statistic (13.1905) is a weighted distance of the observed sufficient statistics from the mean sufficient statistics, and the score p-value (0.0008) is the sum of all probabilities in the Dist data set for sufficient statistics that are no closer to the mean.

Output 79.12.3 First 10 Observations in the Joint Exact Distribution

Obs	DiagnosisAnxious	FriendshipsGood	Count	Score	Prob
1	0	20	1	48.0000	2.5608E-14
2	1	20	420	40.3905	1.0755E-11
3	1	21	168	40.6905	4.3022E-12
4	2	20	39900	33.5619	1.02177E-9
5	2	21	33600	33.4619	8.6044E-10
6	2	22	5880	34.7619	1.5058E-10
7	3	20	1516200	27.5143	3.88272E-8
8	3	21	2021600	27.0143	5.17696E-8
9	3	22	744800	27.9143	1.9073E-8
10	3	23	74480	30.2143	1.9073E-9

For the univariate tests of the Diagnosis variable, PROC LOGISTIC extracts the part of the joint exact distribution for which Friendships=28; this is displayed in Output 79.12.4. In addition, PROC LOGISTIC computes the probability and score tests in Output 79.12.2 in the same fashion as the joint tests. Both of the tests do not reject the null hypothesis that the parameter is zero.

Output 79.12.4 Marginal Exact Distribution for Diagnosis

Obs	DiagnosisAnxious	FriendshipsGood	Count	Score	Prob
162	8	.	203490	17.6871	0.00001
163	9	.	5878600	12.5487	0.00033
164	10	.	67016040	8.2899	0.00380
165	11	.	402096240	4.9108	0.02282
166	12	.	1424090850	2.4113	0.08082
167	13	.	3154908960	0.7915	0.17905
168	14	.	4507012800	0.0513	0.25579
169	15	.	4206545280	0.1907	0.23874
170	16	.	2563363530	1.2098	0.14548
171	17	.	1005240600	3.1086	0.05705
172	18	.	245725480	5.8870	0.01395
173	19	.	35271600	9.5450	0.00200
174	20	.	2645370	14.0827	0.00015
175	21	.	77520	19.5000	0.00000

For the univariate tests of the Friendships variable, PROC LOGISTIC extracts the part of the exact distribution for which Diagnosis=13; this is displayed in Output 79.12.5. Both tests in Output 79.12.2 reject the null hypothesis that the parameter is zero.

Output 79.12.5 Marginal Exact Distribution for Friendships

Obs	Diagnosis	Anxious	Friendships	Good	Count	Score	Prob
176		.		20	15774544800	9.3600	0.00207
177		.		21	205069082400	4.9985	0.02692
178		.		22	956989051200	1.9938	0.12560
179		.		23	2.1053759E12	0.3462	0.27633
180		.		24	2.3924726E12	0.0554	0.31401
181		.		25	1.4354836E12	1.1215	0.18841
182		.		26	441687254400	3.5446	0.05797
183		.		27	63098179200	7.3246	0.00828
184		.		28	3154908960	12.4615	0.00041

The parameter estimates are displayed in Output 79.12.6. Similar to the univariate tests, the parameter estimates are derived from the marginal exact distributions.

The Diagnosis parameter estimate (–0.5981) is computed by an iterative search for the parameter that maximizes the univariate conditional probability density function, as described in the section "Inference for a Single Parameter" on page 6074. The Diagnosis parameter is not significantly different from zero.

Because the observed sufficient statistic for the Friendships parameter is on the edge of its distribution, the Friendships parameter estimate is the value for which the univariate conditional probability density function is equal to 0.5. In a similar fashion, the confidence limits for both the Diagnosis and Friendships parameters are created by finding values to make the two tail probabilities equal to 0.025.

Output 79.12.6 Exact Parameter Estimates

Exact Parameter Estimates

Parameter		Estimate		Standard Error	95% Confidence Limits		p-Value
Diagnosis	**Anxious**	-0.5981		0.6760	-2.1970	0.9103	0.5737
Friendships	**Good**	3.2612	*	.	1.4948	Infinity	0.0004

Note: * indicates a median unbiased estimate and a one-sided p-value.

To conclude, you find that the Friendships variable is a significant effect for explaining how mothers perceive the recovery of their children.

You can also use the exact parameter estimates to compute predicted probabilities for any data. This facility is not built into PROC LOGISTIC for exact logistic regression, because exact methods can be very expensive and the computations can fail. But this example is well behaved, so you can use the following statements to score the data:

```
proc logistic data=one exactonly outest=est;
   class Diagnosis Friendships / param=ref;
   model Recovered/Total = Diagnosis Friendships;
   exact Intercept Diagnosis Friendships / estimate;
run;
proc means data=est noprint;
   output out=out;
run;
data out; set out; if _STAT_='MEAN'; drop _TYPE_; run;
```

```
data est(type=est); set out; _TYPE_='PARMS'; run;
```

You specify the INTERCEPT keyword in the EXACT statement to compute an exact estimate for the intercept in addition to the other parameters. The parameter estimates are stored in the OUTEST= data set. Because there are both maximum-likelihood and median-unbiased estimates, the PROC MEANS statement accumulates the estimates into one observation, and then a TYPE=EST data set is formed.

The following program uses the scoring facility for unconditional logistic regression to score the original data set by using the exact parameter estimates:

```
proc logistic data=one inest=est;
   class Diagnosis Friendships / param=ref;
   model Recovered/Total = Diagnosis Friendships / maxiter=0;
   score out=score;
run;

proc print data=score;
   var Diagnosis Friendships P_Event;
run;
```

Output 79.12.7 shows that good friendships correspond to high recovery probabilities.

Output 79.12.7 Data Scored by Using Exact Parameter Estimates

Obs	Diagnosis	Friendships	P_Event
1	Anxious	Poor	0.04741
2	Anxious	Good	0.56484
3	Depressed	Poor	0.08300
4	Depressed	Good	0.70243

Example 79.13: Firth's Penalized Likelihood Compared with Other Approaches

(View the complete code for this example (logiex13.sas) in the example repository.)

Firth's penalized likelihood approach is a method of addressing issues of separability, small sample sizes, and bias of the parameter estimates. This example performs some comparisons between results from using the FIRTH option to results from the usual unconditional, conditional, and exact logistic regression analyses. When the sample size is large enough, the unconditional estimates and the Firth penalized-likelihood estimates should be nearly the same. These examples show that Firth's penalized likelihood approach compares favorably with unconditional, conditional, and exact logistic regression; however, this is not an exhaustive analysis of Firth's method. For more detailed analyses with separable data sets, see Heinze (2006, 1999) and Heinze and Schemper (2002).

Comparison on 2x2 Tables with One Zero Cell

A 2×2 table with one cell having zero frequency, where the rows of the table are the levels of a covariate while the columns are the levels of the response variable, is an example of a quasi-completely separated data set. The parameter estimate for the covariate under unconditional logistic regression will move off to infinity, although PROC LOGISTIC will stop the iterations at an earlier point in the process. An exact logistic regression is sometimes performed to determine the importance of the covariate in describing the variation in the data, but the median-unbiased parameter estimate, while finite, might not be near the true value, and one confidence limit (for this example, the upper) is always infinite.

The following DATA step produces 1000 different 2×2 tables, all following an underlying probability structure, with one cell having a near zero probability of being observed:

```
%let beta0=-15;
%let beta1=16;
data one;
   keep sample X y pry;
   do sample=1 to 1000;
      do i=1 to 100;
         X=rantbl(987987,.4,.6)-1;
         xb= &beta0 + X*&beta1;
         exb=exp(xb);
         pry= exb/(1+exb);
         cut= ranuni(393993);
         if (pry < cut) then y=1; else y=0;
         output;
      end;
   end;
run;
```

The following statements perform the bias-corrected and exact logistic regression on each of the 1000 different data sets, output the odds ratio tables by using the ODS OUTPUT statement, and compute various statistics across the data sets by using the MEANS procedure:

```
ods exclude all;
proc logistic data=one;
   by sample;
   class X(param=ref);
   model y(event='1')=X / firth clodds=pl;
   ods output cloddspl=firth;
run;
proc logistic data=one exactonly;
   by sample;
   class X(param=ref);
   model y(event='1')=X;
   exact X / estimate=odds;
   ods output exactoddsratio=exact;
run;
ods select all;
proc means data=firth;
   var LowerCL OddsRatioEst UpperCL;
run;
proc means data=exact;
   var LowerCL Estimate UpperCL;
```

```
run;
```

The results of the PROC MEANS statements are summarized in Table 79.21. You can see that the odds ratios are all quite large; the confidence limits on every table suggest that the covariate X is a significant factor in explaining the variability in the data.

Table 79.21 Odds Ratio Results

Method	Mean Estimate	Standard Error	Minimum Lower CL	Maximum Upper CL
Firth	231.59	83.57	10.40	111317
Exact	152.02	52.30	11.16	∞

Comparison on Case-Control Data

Case-control models contain an intercept term for every case-control pair in the data set. This means that there are a large number of parameters compared to the number of observations. Breslow and Day (1980) note that the estimates from unconditional logistic regression are biased with the corresponding odds ratios off by a power of 2 from the true value; conditional logistic regression was developed to remedy this.

The following DATA step produces 1000 case-control data sets, with pair indicating the strata:

```
%let beta0=1;
%let beta1=2;
data one;
   do sample=1 to 1000;
      do pair=1 to 20;
         ran=ranuni(939393);
         a=3*ranuni(9384984)-1;
         pdf0= pdf('NORMAL',a,.4,1);
         pdf1= pdf('NORMAL',a,1,1);
         pry0= pdf0/(pdf0+pdf1);
         pry1= 1-pry0;
         xb= log(pry0/pry1);
         x= (xb-&beta0*pair/100) / &beta1;
         y=0;
         output;
         x= (-xb-&beta0*pair/100) / &beta1;
         y=1;
         output;
      end;
   end;
run;
```

Unconditional, conditional, exact, and Firth-adjusted analyses are performed on the data sets, and the mean, minimum, and maximum odds ratios and the mean upper and lower limits for the odds ratios are displayed in Table 79.22. **CAUTION:** Due to the exact analyses, this program takes a long time and a lot of resources to run. You might want to reduce the number of samples generated.

```
ods exclude all;
proc logistic data=one;
   by sample;
   class pair / param=ref;
   model y=x pair / clodds=pl;
   ods output cloddspl=oru;
run;
data oru;
   set oru;
   if Effect='x';
   rename lowercl=lclu uppercl=uclu oddsratioest=orestu;
run;
proc logistic data=one;
   by sample;
   strata pair;
   model y=x / clodds=wald;
   ods output cloddswald=orc;
run;
data orc;
   set orc;
   if Effect='x';
   rename lowercl=lclc uppercl=uclc oddsratioest=orestc;
run;
proc logistic data=one exactonly;
   by sample;
   strata pair;
   model y=x;
   exact x / estimate=both;
   ods output ExactOddsRatio=ore;
run;
proc logistic data=one;
   by sample;
   class pair / param=ref;
   model y=x pair / firth clodds=pl;
   ods output cloddspl=orf;
run;
data orf;
   set orf;
   if Effect='x';
   rename lowercl=lclf uppercl=uclf oddsratioest=orestf;
run;
data all;
   merge oru orc ore orf;
run;
ods select all;
proc means data=all;
run;
```

You can see from Table 79.22 that the conditional, exact, and Firth-adjusted results are all comparable, while the unconditional results are several orders of magnitude different.

Table 79.22 Odds Ratio Estimates

Method	N	Minimum	Mean	Maximum
Unconditional	1000	0.00045	112.09	38038
Conditional	1000	0.021	4.20	195
Exact	1000	0.021	4.20	195
Firth	1000	0.018	4.89	71

Further examination of the data set all shows that the differences between the square root of the unconditional odds ratio estimates and the conditional estimates have mean –0.00019 and standard deviation 0.0008, verifying that the unconditional odds ratio is about the square of the conditional odds ratio. The conditional and exact conditional odds ratios are also nearly equal, with their differences having mean 3E–7 and standard deviation 6E–6. The differences between the Firth and the conditional odds ratios can be large (mean 0.69, standard deviation 5.40), but their relative differences, $\frac{Firth - Conditional}{Conditional}$, have mean 0.20 with standard deviation 0.19, so the largest differences occur with the larger estimates.

Example 79.14: Complementary Log-Log Model for Infection Rates

(View the complete code for this example (logiex14.sas) in the example repository.)

Antibodies produced in response to an infectious disease like malaria remain in the body after the individual has recovered from the disease. A serological test detects the presence or absence of such antibodies. An individual with such antibodies is called seropositive. In geographic areas where the disease is endemic, the inhabitants are at fairly constant risk of infection. The probability of an individual never having been infected in Y years is $\exp(-\mu Y)$, where μ is the mean number of infections per year (see the appendix of Draper, Voller, and Carpenter 1972). Rather than estimating the unknown μ, epidemiologists want to estimate the probability of a person living in the area being infected in one year. This infection rate γ is given by

$$\gamma = 1 - e^{-\mu}$$

The following statements create the data set sero, which contains the results of a serological survey of malarial infection. Individuals of nine age groups (Group) were tested. The variable A represents the midpoint of the age range for each age group. The variable N represents the number of individuals tested in each age group, and the variable R represents the number of individuals that are seropositive.

```
data sero;
   input Group A N R;
   X=log(A);
   label X='Log of Midpoint of Age Range';
   datalines;
1  1.5  123  8
2  4.0  132  6
3  7.5  182 18
4 12.5  140 14
```

```
5 17.5  138 20
6 25.0  161 39
7 35.0  133 19
8 47.0   92 25
9 60.0   74 44
;
```

For the ith group with the age midpoint A_i, the probability of being seropositive is $p_i = 1 - \exp(-\mu A_i)$. It follows that

$$\log(-\log(1 - p_i)) = \log(\mu) + \log(A_i)$$

By fitting a binomial model with a complementary log-log link function and by using X=log(A) as an offset term, you can estimate $\alpha = \log(\mu)$ as an intercept parameter. The following statements invoke PROC LOGISTIC to compute the maximum likelihood estimate of α. The LINK=CLOGLOG option is specified to request the complementary log-log link function. Also specified is the CLPARM=PL option, which requests the profile-likelihood confidence limits for α.

```
proc logistic data=sero;
   model R/N= / offset=X
                link=cloglog
                clparm=pl
                scale=none;
   title 'Constant Risk of Infection';
run;
```

Results of fitting this constant risk model are shown in Output 79.14.1.

Output 79.14.1 Modeling Constant Risk of Infection

Constant Risk of Infection

The LOGISTIC Procedure

Model Information		
Data Set	WORK.SERO	
Response Variable (Events)	R	
Response Variable (Trials)	N	
Offset Variable	X	Log of Midpoint of Age Range
Model	binary cloglog	
Optimization Technique	Fisher's scoring	

Number of Observations Read	9
Number of Observations Used	9
Sum of Frequencies Read	1175
Sum of Frequencies Used	1175

Response Profile		
Ordered Value	**Binary Outcome**	**Total Frequency**
1	Event	193
2	Nonevent	982

Output 79.14.1 *continued*

Intercept-Only Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

-2 Log L = 967.1158

Deviance and Pearson Goodness-of-Fit Statistics				
Criterion	Value	DF	Value/DF	Pr > ChiSq
Deviance	41.5032	8	5.1879	<.0001
Pearson	50.6883	8	6.3360	<.0001

Number of events/trials observations: 9

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-4.6605	0.0725	4133.5626	<.0001
X	0	1.0000	0	.	.

Parameter Estimates and Profile-Likelihood Confidence Intervals			
Parameter	Estimate	95% Confidence Limits	
Intercept	-4.6605	-4.8057	-4.5219

Output 79.14.1 shows that the maximum likelihood estimate of $\alpha = \log(\mu)$ and its estimated standard error are $\widehat{\alpha} = -4.6605$ and $\widehat{\sigma}_{\widehat{\alpha}} = 0.0725$, respectively. The infection rate is estimated as

$$\widehat{\gamma} = 1 - e^{-\widehat{\mu}} = 1 - e^{-e^{\widehat{\beta}_0}} = 1 - e^{-e^{-4.6605}} = 0.00942$$

The 95% confidence interval for γ, obtained by back-transforming the 95% confidence interval for α, is (0.0082, 0.0108); that is, there is a 95% chance that, in repeated sampling, the interval of 8 to 11 infections per thousand individuals contains the true infection rate.

The goodness-of-fit statistics for the constant risk model are statistically significant ($p < 0.0001$), indicating that the assumption of constant risk of infection is not correct. You can fit a more extensive model by allowing a separate risk of infection for each age group. Suppose μ_i is the mean number of infections per year for the ith age group. The probability of seropositive for the ith group with the age midpoint A_i is $p_i = 1 - \exp(-\mu_i A_i)$, so that

$$\log(-\log(1 - p_i)) = \log(\mu_i) + \log(A_i)$$

In the following statements, a complementary log-log model is fit containing Group as an explanatory classification variable with the GLM coding (so that a dummy variable is created for each age group), no intercept term, and X=log(A) as an offset term. The ODS OUTPUT statement saves the estimates and their 95% profile-likelihood confidence limits to the ClparmPL data set. Note that $\log(\mu_i)$ is the regression parameter associated with Group= i.

```
proc logistic data=sero;
   ods output ClparmPL=ClparmPL;
   class Group / param=glm;
   model R/N=Group / noint
```

```
                        offset=X
                        link=cloglog
                        clparm=pl;
    title 'Infectious Rates and 95% Confidence Intervals';
  run;
```

Results of fitting the model with a separate risk of infection are shown in Output 79.14.2.

Output 79.14.2 Modeling Separate Risk of Infection

Infectious Rates and 95% Confidence Intervals

The LOGISTIC Procedure

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Group	1	1	-3.1048	0.3536	77.0877	<.0001
Group	2	1	-4.4542	0.4083	119.0164	<.0001
Group	3	1	-4.2769	0.2358	328.9593	<.0001
Group	4	1	-4.7761	0.2674	319.0600	<.0001
Group	5	1	-4.7165	0.2238	443.9920	<.0001
Group	6	1	-4.5012	0.1606	785.1350	<.0001
Group	7	1	-5.4252	0.2296	558.1114	<.0001
Group	8	1	-4.9987	0.2008	619.4666	<.0001
Group	9	0	0	.	.	.
X		0	1.0000	0	.	.

Parameter Estimates and Profile-Likelihood Confidence Intervals

Parameter		Estimate	95% Confidence Limits	
Group	1	-3.1048	-3.8880	-2.4833
Group	2	-4.4542	-5.3769	-3.7478
Group	3	-4.2769	-4.7775	-3.8477
Group	4	-4.7761	-5.3501	-4.2940
Group	5	-4.7165	-5.1896	-4.3075
Group	6	-4.5012	-4.8333	-4.2019
Group	7	-5.4252	-5.9116	-5.0063
Group	8	-4.9987	-5.4195	-4.6289

For the first age group (Group=1), the point estimate of $\log(\mu_1)$ is –3.1048, which transforms into an infection rate of $1 - \exp(-\exp(-3.1048)) = 0.0438$. A 95% confidence interval for this infection rate is obtained by transforming the 95% confidence interval for $\log(\mu_1)$. For the first age group, the lower and upper confidence limits are $1 - \exp(-\exp(-3.8880) = 0.0203$ and $1 - \exp(-\exp(-2.4833)) = 0.0801$, respectively; that is, there is a 95% chance that, in repeated sampling, the interval of 20 to 80 infections per thousand individuals contains the true infection rate. The following statements perform this transformation on the estimates and confidence limits saved in the ClparmPL data set; the resulting estimated infection rates in one year's time for each age group are displayed in Table 79.23. Note that the infection rate for the first age group is high compared to that of the other age groups.

```
data ClparmPL;
   set ClparmPL;
   Estimate=round( 1000*( 1-exp(-exp(Estimate)) ) );
   LowerCL =round( 1000*( 1-exp(-exp(LowerCL )) ) );
   UpperCL =round( 1000*( 1-exp(-exp(UpperCL )) ) );
run;
```

Table 79.23 Infection Rate in One Year

Age Group	Number Infected per 1,000 People: Point Estimate	95% Confidence Limits: Lower	Upper
1	44	20	80
2	12	5	23
3	14	8	21
4	8	5	14
5	9	6	13
6	11	8	15
7	4	3	7
8	7	4	10
9	15	11	20

Example 79.15: Complementary Log-Log Model for Interval-Censored Survival Times

(View the complete code for this example (logiex15.sas) in the example repository.)

Often survival times are not observed more precisely than the interval (for instance, a day) within which the event occurred. Survival data of this form are known as grouped or interval-censored data. A discrete analog of the continuous proportional hazards model (Prentice and Gloeckler 1978; Allison 1982) is used to investigate the relationship between these survival times and a set of explanatory variables. For more methods of fitting these data, see Chapter 70, "The ICPHREG Procedure."

Suppose T_i is the discrete survival time variable of the ith subject with covariates $\mathbf{x}_i$. The discrete-time hazard rate λ_{it} is defined as

$$\lambda_{it} = \Pr(T_i = t \mid T_i \geq t, \mathbf{x}_i), \quad t = 1, 2, \ldots$$

Using elementary properties of conditional probabilities, it can be shown that

$$\Pr(T_i = t) = \lambda_{it} \prod_{j=1}^{t-1} (1 - \lambda_{ij}) \quad \text{and} \quad \Pr(T_i > t) = \prod_{j=1}^{t} (1 - \lambda_{ij})$$

Suppose t_i is the observed survival time of the ith subject. Suppose $\delta_i = 1$ if $T_i = t_i$ is an event time and 0

otherwise. The likelihood for the grouped survival data is given by

$$
\begin{aligned}
L &= \prod_i [\Pr(T_i = t_i)]^{\delta_i} [\Pr(T_i > t_i)]^{1-\delta_i} \\
&= \prod_i \left(\frac{\lambda_{it_i}}{1-\lambda_{it_i}} \right)^{\delta_i} \prod_{j=1}^{t_i} (1-\lambda_{ij}) \\
&= \prod_i \prod_{j=1}^{t_i} \left(\frac{\lambda_{ij}}{1-\lambda_{ij}} \right)^{y_{ij}} (1-\lambda_{ij})
\end{aligned}
$$

where $y_{ij} = 1$ if the ith subject experienced an event at time $T_i = j$ and 0 otherwise.

Note that the likelihood L for the grouped survival data is the same as the likelihood of a binary response model with event probabilities λ_{ij}. If the data are generated by a continuous-time proportional hazards model, Prentice and Gloeckler (1978) have shown that

$$\lambda_{ij} = 1 - \exp(-\exp(\alpha_j + \boldsymbol{\beta}'\mathbf{x}_i))$$

which can be rewritten as

$$\log(-\log(1-\lambda_{ij})) = \alpha_j + \boldsymbol{\beta}'\mathbf{x}_i$$

where the coefficient vector $\boldsymbol{\beta}$ is identical to that of the continuous-time proportional hazards model, and α_j is a constant related to the conditional survival probability in the interval defined by $T_i = j$ at $\mathbf{x}_i = \mathbf{0}$. The grouped data survival model is therefore equivalent to the binary response model with complementary log-log link function. To fit the grouped survival model by using PROC LOGISTIC, you must treat each discrete time unit for each subject as a separate observation. For each of these observations, the response is dichotomous, corresponding to whether or not the subject died in the time unit.

Consider a study of the effect of insecticide on flour beetles. Four different concentrations of an insecticide were sprayed on separate groups of flour beetles. The following DATA step saves the number of male and female flour beetles dying in successive intervals in the data set Beetles:

```
data Beetles(keep=time sex conc freq);
   input time m20 f20 m32 f32 m50 f50 m80 f80;
   conc=.20; freq= m20; sex=1; output;
             freq= f20; sex=2; output;
   conc=.32; freq= m32; sex=1; output;
             freq= f32; sex=2; output;
   conc=.50; freq= m50; sex=1; output;
             freq= f50; sex=2; output;
   conc=.80; freq= m80; sex=1; output;
             freq= f80; sex=2; output;
   datalines;
 1   3   0  7  1  5  0  4  2
 2  11   2 10  5  8  4 10  7
 3  10   4 11 11 11  6  8 15
 4   7   8 16 10 15  6 14  9
 5   4   9  3  5  4  3  8  3
 6   3   3  2  1  2  1  2  4
 7   2   0  1  0  1  1  1  1
```

```
 8   1   0  0  1  1  4  0  1
 9   0   0  1  1  0  0  0  0
10   0   0  0  0  0  0  1  1
11   0   0  0  0  1  1  0  0
12   1   0  0  0  0  1  0  0
13   1   0  0  0  0  1  0  0
14 101 126 19 47  7 17  2  4
;
```

The data set Beetles contains four variables: time, sex, conc, and freq. The variable time represents the interval death time; for example, time=2 is the interval between day 1 and day 2. Insects surviving the duration (13 days) of the experiment are given a time value of 14. The variable sex represents the sex of the insects (1=male, 2=female), conc represents the concentration of the insecticide (mg/cm^2), and freq represents the frequency of the observations.

To use PROC LOGISTIC with the grouped survival data, you must expand the data so that each beetle has a separate record for each day of survival. A beetle that died in the third day (time=3) would contribute three observations to the analysis, one for each day it was alive at the beginning of the day. A beetle that survives the 13-day duration of the experiment (time=14) would contribute 13 observations.

The following DATA step creates a new data set named Days containing the beetle-day observations from the data set Beetles. In addition to the variables sex, conc, and freq, the data set contains an outcome variable y and a classification variable day. The variable y has a value of 1 if the observation corresponds to the day that the beetle died, and it has a value of 0 otherwise. An observation for the first day will have a value of 1 for day; an observation for the second day will have a value of 2 for day, and so on. For instance, Output 79.15.1 shows an observation in the Beetles data set with time=3, and Output 79.15.2 shows the corresponding beetle-day observations in the data set Days.

```
data Days;
   set Beetles;
   do day=1 to time;
      if (day < 14) then do;
         y= (day=time);
         output;
      end;
   end;
run;
```

Output 79.15.1 An Observation with Time=3 in Beetles Data Set

Obs	time	conc	freq	sex
17	3	0.2	10	1

Output 79.15.2 Corresponding Beetle-Day Observations in Days

Obs	time	conc	freq	sex	day	y
25	3	0.2	10	1	1	0
26	3	0.2	10	1	2	0
27	3	0.2	10	1	3	1

The following statements invoke PROC LOGISTIC to fit a complementary log-log model for binary data with

the response variable Y and the explanatory variables day, sex, and Variableconc. Specifying the EVENT= option ensures that the event (y=1) probability is modeled. The GLM coding in the CLASS statement creates an indicator column in the design matrix for each level of day. The coefficients of the indicator effects for day can be used to estimate the baseline survival function. The NOINT option is specified to prevent any redundancy in estimating the coefficients of day. The Newton-Raphson algorithm is used for the maximum likelihood estimation of the parameters.

```
proc logistic data=Days outest=est1;
   class day / param=glm;
   model y(event='1')= day sex conc
         / noint link=cloglog technique=newton;
   freq freq;
run;
```

Results of the model fit are given in Output 79.15.3. Both sex and conc are statistically significant for the survival of beetles sprayed by the insecticide. Female beetles are more resilient to the chemical than male beetles, and increased concentration of the insecticide increases its effectiveness.

Output 79.15.3 Parameter Estimates for the Grouped Proportional Hazards Model

Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
day	1	1	-3.9314	0.2934	179.5602	<.0001
day	2	1	-2.8751	0.2412	142.0596	<.0001
day	3	1	-2.3985	0.2299	108.8833	<.0001
day	4	1	-1.9953	0.2239	79.3960	<.0001
day	5	1	-2.4920	0.2515	98.1470	<.0001
day	6	1	-3.1060	0.3037	104.5799	<.0001
day	7	1	-3.9704	0.4230	88.1107	<.0001
day	8	1	-3.7917	0.4007	89.5233	<.0001
day	9	1	-5.1540	0.7316	49.6329	<.0001
day	10	1	-5.1350	0.7315	49.2805	<.0001
day	11	1	-5.1131	0.7313	48.8834	<.0001
day	12	1	-5.1029	0.7313	48.6920	<.0001
day	13	1	-5.0951	0.7313	48.5467	<.0001
sex		1	-0.5651	0.1141	24.5477	<.0001
conc		1	3.0918	0.2288	182.5665	<.0001

The coefficients of parameters for the day variable are the maximum likelihood estimates of $\alpha_1, \ldots, \alpha_{13}$, respectively. The baseline survivor function $S_0(t)$ is estimated by

$$\hat{S}_0(t) = \widehat{\Pr}(T > t) = \prod_{j \le t} \exp(-\exp(\widehat{\alpha}_j))$$

and the survivor function for a given covariate pattern (sex=x_1 and conc=x_2) is estimated by

$$\hat{S}(t) = [\hat{S}_0(t)]^{\exp(-0.5651x_1 + 3.0918x_2)}$$

The following statements compute the survival curves for male and female flour beetles exposed to the insecticide in concentrations of 0.20 mg/cm^2 and 0.80 mg/cm^2:

```
data one (keep=day survival element s_m20 s_f20 s_m80 s_f80);
   array dd[13] day1-day13;
   array sc[4] m20 f20 m80 f80;
   array s_sc[4] s_m20 s_f20 s_m80 s_f80 (1 1 1 1);
   set est1;
   m20= exp(sex + .20 * conc);
   f20= exp(2 * sex + .20 * conc);
   m80= exp(sex + .80 * conc);
   f80= exp(2 * sex + .80 * conc);
   survival=1;
   day=0;
   output;
   do day=1 to 13;
      element= exp(-exp(dd[day]));
      survival= survival * element;
      do i=1 to 4;
         s_sc[i] = survival ** sc[i];
      end;
      output;
   end;
run;
```

Instead of plotting the curves as step functions, the following statements use the PBSPLINE statement in the SGPLOT procedure to smooth the curves with a penalized B-spline. For more information about the implementation of the penalized B-spline method, see Chapter 126, "The TRANSREG Procedure." The SAS autocall macro %MODSTYLE is specified to change the marker symbols for the plot. For more information about the %MODSTYLE macro, see the section "ODS Style Template Modification Macro" on page 761 in Chapter 24, "Statistical Graphics Using ODS." The smoothed survival curves are displayed in Output 79.15.4.

```
%modstyle(name=LogiStyle,parent=htmlblue,markers=circlefilled);
ods listing style=LogiStyle;
proc sgplot data=one;
   title 'Flour Beetles Sprayed with Insecticide';
   xaxis grid integer;
   yaxis grid label='Survival Function';
   pbspline y=s_m20 x=day /
      legendlabel = "Male at 0.20 conc." name="pred1";
   pbspline y=s_m80 x=day /
      legendlabel = "Male at 0.80 conc." name="pred2";
   pbspline y=s_f20 x=day /
      legendlabel = "Female at 0.20 conc." name="pred3";
   pbspline y=s_f80 x=day /
      legendlabel = "Female at 0.80 conc." name="pred4";
   discretelegend "pred1" "pred2" "pred3" "pred4" / across=2;
run;
```

Output 79.15.4 Predicted Survival at Insecticide Concentrations of 0.20 and 0.80 mg/cm^2

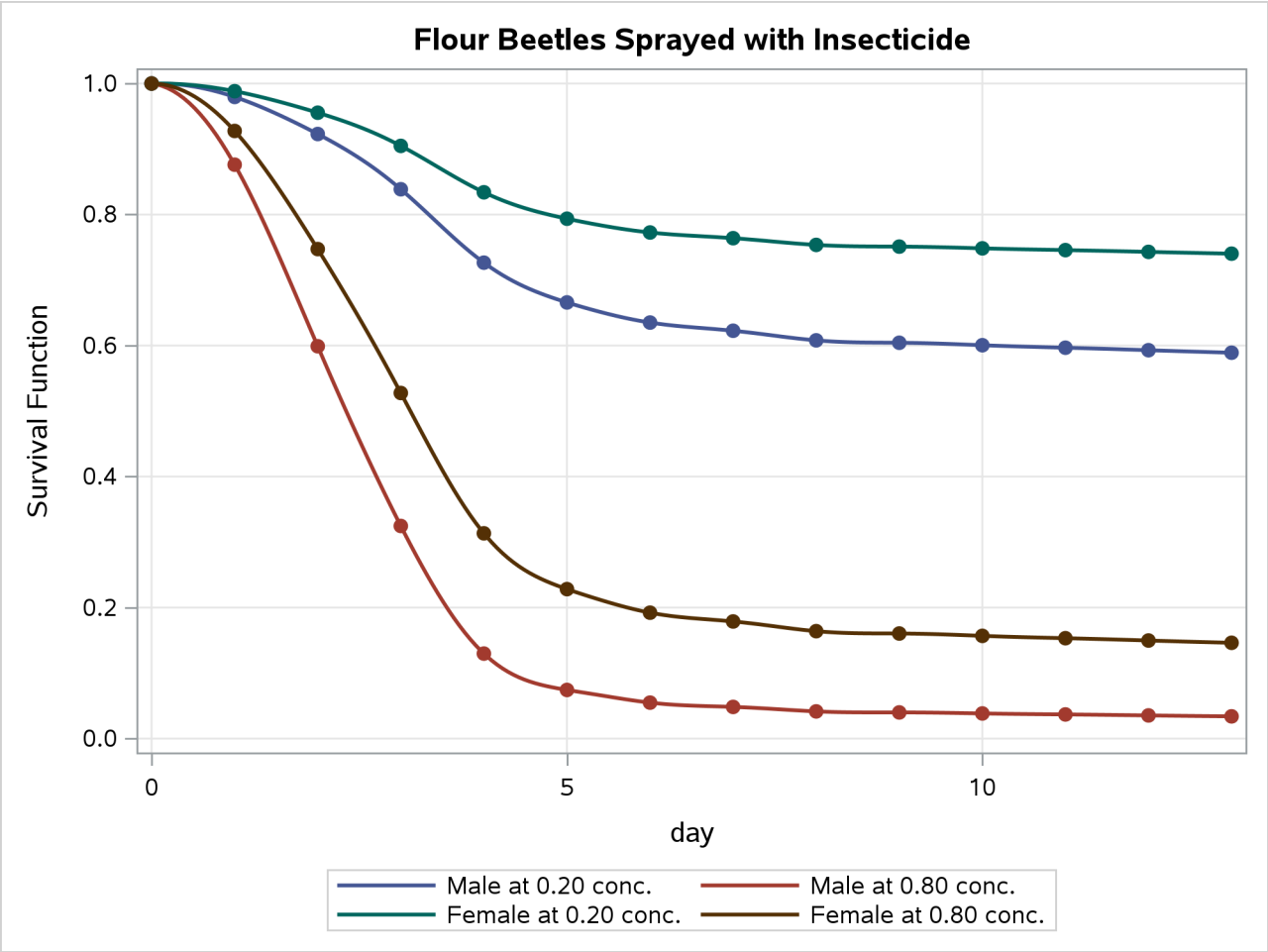

The probability of survival is displayed on the vertical axis. Notice that most of the insecticide effect occurs by day 6 for both the high and low concentrations.

Example 79.16: Scoring Data Sets

(View the complete code for this example (logiex16.sas) in the example repository.)

This example first illustrates the syntax used for scoring data sets, then uses a previously scored data set to score a new data set. A generalized logit model is fit to the remote-sensing data set used in the section "Example 42.4: Linear Discriminant Analysis of Remote-Sensing Data on Crops" on page 2802 in Chapter 42, "The DISCRIM Procedure," to illustrate discrimination and classification methods. In the following DATA step, the response variable is Crop and the prognostic factors are x1 through x4:

```
data Crops;
   length Crop $ 10;
   infile datalines truncover;
   input Crop $ @@;
   do i=1 to 3;
```

```
      input x1-x4 @@;
      if (x1 ^= .) then output;
   end;
   input;
   datalines;
Corn       16 27 31 33  15 23 30 30  16 27 27 26
Corn       18 20 25 23  15 15 31 32  15 32 32 15
Corn       12 15 16 73
Soybeans   20 23 23 25  24 24 25 32  21 25 23 24
Soybeans   27 45 24 12  12 13 15 42  22 32 31 43
Cotton     31 32 33 34  29 24 26 28  34 32 28 45
Cotton     26 25 23 24  53 48 75 26  34 35 25 78
Sugarbeets 22 23 25 42  25 25 24 26  34 25 16 52
Sugarbeets 54 23 21 54  25 43 32 15  26 54  2 54
Clover     12 45 32 54  24 58 25 34  87 54 61 21
Clover     51 31 31 16  96 48 54 62  31 31 11 11
Clover     56 13 13 71  32 13 27 32  36 26 54 32
Clover     53 08 06 54  32 32 62 16
;
```

In the following statements, you specify a SCORE statement to use the fitted model to score the Crops data. The data together with the predicted values are saved in the data set Score1. The output from the EFFECTPLOT statement is discussed at the end of this section.

```
ods graphics on;
proc logistic data=Crops;
   model Crop=x1-x4 / link=glogit;
   score out=Score1;
   effectplot slicefit(x=x3);
run;
```

In the following statements, the model is fit again, and the data and the predicted values are saved into the data set Score2. The OUTMODEL= option saves the fitted model information in the permanent SAS data set sasuser.CropModel, and the STORE statement saves the fitted model information into the SAS data set CropModel2. Both the OUTMODEL= option and the STORE statement are specified to illustrate their use; you would usually specify only one of these model-storing methods.

```
proc logistic data=Crops outmodel=sasuser.CropModel;
   model Crop=x1-x4 / link=glogit;
   score data=Crops out=Score2;
   store CropModel2;
run;
```

To score data without refitting the model, specify the INMODEL= option to identify a previously saved SAS data set of model information. In the following statements, the model is read from the sasuser.CropModel data set, and the data and the predicted values are saved in the data set Score3. Note that the data set being scored does not have to include the response variable.

```
proc logistic inmodel=sasuser.CropModel;
   score data=Crops out=Score3;
run;
```

Another method available to score the data without refitting the model is to invoke the PLM procedure. In the following statements, the stored model is named in the SOURCE= option. The PREDICTED= option

computes the linear predictors, and the ILINK option transforms the linear predictors to the probability scale. The SCORE statement scores the Crops data set, and the predicted probabilities are saved in the data set ScorePLM. See Chapter 94, "The PLM Procedure," for more information.

```
proc plm source=CropModel2;
   score data=Crops out=ScorePLM predicted=p / ilink;
run;
```

For each observation in the Crops data set, the ScorePLM data set contains 5 observations—one for each level of the response variable. The following statements transform this data set into a form that is similar to the other scored data sets in this example:

```
proc transpose data=ScorePLM out=Score4 prefix=P_ let;
   id _LEVEL_;
   var p;
   by x1-x4  notsorted;
run;
data Score4(drop=_NAME_ _LABEL_);
   merge Score4 Crops(keep=Crop x1-x4);
   F_Crop=Crop;
run;
proc summary data=ScorePLM nway;
   by x1-x4 notsorted;
   var p;
   output out=into maxid(p(_LEVEL_))=I_Crop;
run;
data Score4;
   merge Score4 into(keep=I_Crop);
run;
```

To set prior probabilities on the responses, specify the PRIOR= option to identify a SAS data set containing the response levels and their priors. In the following statements, the Prior data set contains the values of the response variable (because this example uses single-trial MODEL statement syntax) and a _PRIOR_ variable containing values proportional to the default priors. The data and the predicted values are saved in the data set Score5.

```
data Prior;
   length Crop $10.;
   input Crop _PRIOR_;
   datalines;
Clover     11
Corn        7
Cotton      6
Soybeans    6
Sugarbeets  6
;

proc logistic inmodel=sasuser.CropModel;
   score data=Crops prior=prior out=Score5 fitstat;
run;
```

The "Fit Statistics for SCORE Data" table displayed in Output 79.16.1 shows that 47.22% of the observations are misclassified.

Output 79.16.1 Fit Statistics for Data Set Prior

Fit Statistics for SCORE Data											
Data Set	Total Frequency	Log Likelihood	Error Rate	AIC	AICC	BIC	SC	R-Square	Max-Rescaled R-Square	AUC	Brier Score
WORK.CROPS	36	-32.2247	0.4722	104.4493	160.4493	136.1197	136.1197	0.744081	0.777285	.	0.492712

The data sets Score1, Score2, Score3, Score4, and Score5 are identical. The following statements display the scoring results in Output 79.16.2:

```
proc freq data=Score1;
   table F_Crop*I_Crop / nocol nocum nopercent;
run;
```

Output 79.16.2 Classification of Data Used for Scoring

Frequency
Row Pct

Table of F_Crop by I_Crop

F_Crop(From: Crop)	I_Crop(Into: Crop) Clover	Corn	Cotton	Soybeans	Sugarbeets	Total
Clover	6 54.55	0 0.00	2 18.18	2 18.18	1 9.09	11
Corn	0 0.00	7 100.00	0 0.00	0 0.00	0 0.00	7
Cotton	4 66.67	0 0.00	1 16.67	1 16.67	0 0.00	6
Soybeans	1 16.67	1 16.67	1 16.67	3 50.00	0 0.00	6
Sugarbeets	2 33.33	0 0.00	0 0.00	2 33.33	2 33.33	6
Total	13	8	4	8	3	36

The following statements use the previously fitted and saved model in the sasuser.CropModel data set to score the observations in a new data set, Test. The results of scoring the test data are saved in the ScoredTest data set and displayed in Output 79.16.3.

```
data Test;
   input Crop $ 1-10 x1-x4;
   datalines;
Corn       16 27 31 33
Soybeans   21 25 23 24
Cotton     29 24 26 28
Sugarbeets 54 23 21 54
Clover     32 32 62 16
;

proc logistic noprint inmodel=sasuser.CropModel;
   score data=Test out=ScoredTest;
run;

proc print data=ScoredTest label noobs;
   var F_Crop I_Crop P_Clover P_Corn P_Cotton P_Soybeans P_Sugarbeets;
run;
```

Output 79.16.3 Classification of Test Data

From: Crop	Into: Crop	Predicted Probability: Crop=Clover	Predicted Probability: Crop=Corn	Predicted Probability: Crop=Cotton	Predicted Probability: Crop=Soybeans	Predicted Probability: Crop=Sugarbeets
Corn	Corn	0.00342	0.90067	0.00500	0.08675	0.00416
Soybeans	Soybeans	0.04801	0.03157	0.02865	0.82933	0.06243
Cotton	Clover	0.43180	0.00015	0.21267	0.07623	0.27914
Sugarbeets	Clover	0.66681	0.00000	0.17364	0.00000	0.15955
Clover	Cotton	0.41301	0.13386	0.43649	0.00033	0.01631

The EFFECTPLOT statement that is specified in the first PROC LOGISTIC invocation produces a plot of the model-predicted probabilities versus X3 while holding the other three covariates at their means (Output 79.16.4). This plot shows how the value of X3 affects the probabilities of the various crops when the other prognostic factors are fixed at their means. If you are interested in the effect of X3 when the other covariates are fixed at a certain level—say, 10—specify the following EFFECTPLOT statement.

```
effectplot slicefit(x=x3) / at(x1=10 x2=10 x4=10)
```

Output 79.16.4 Model-Predicted Probabilities

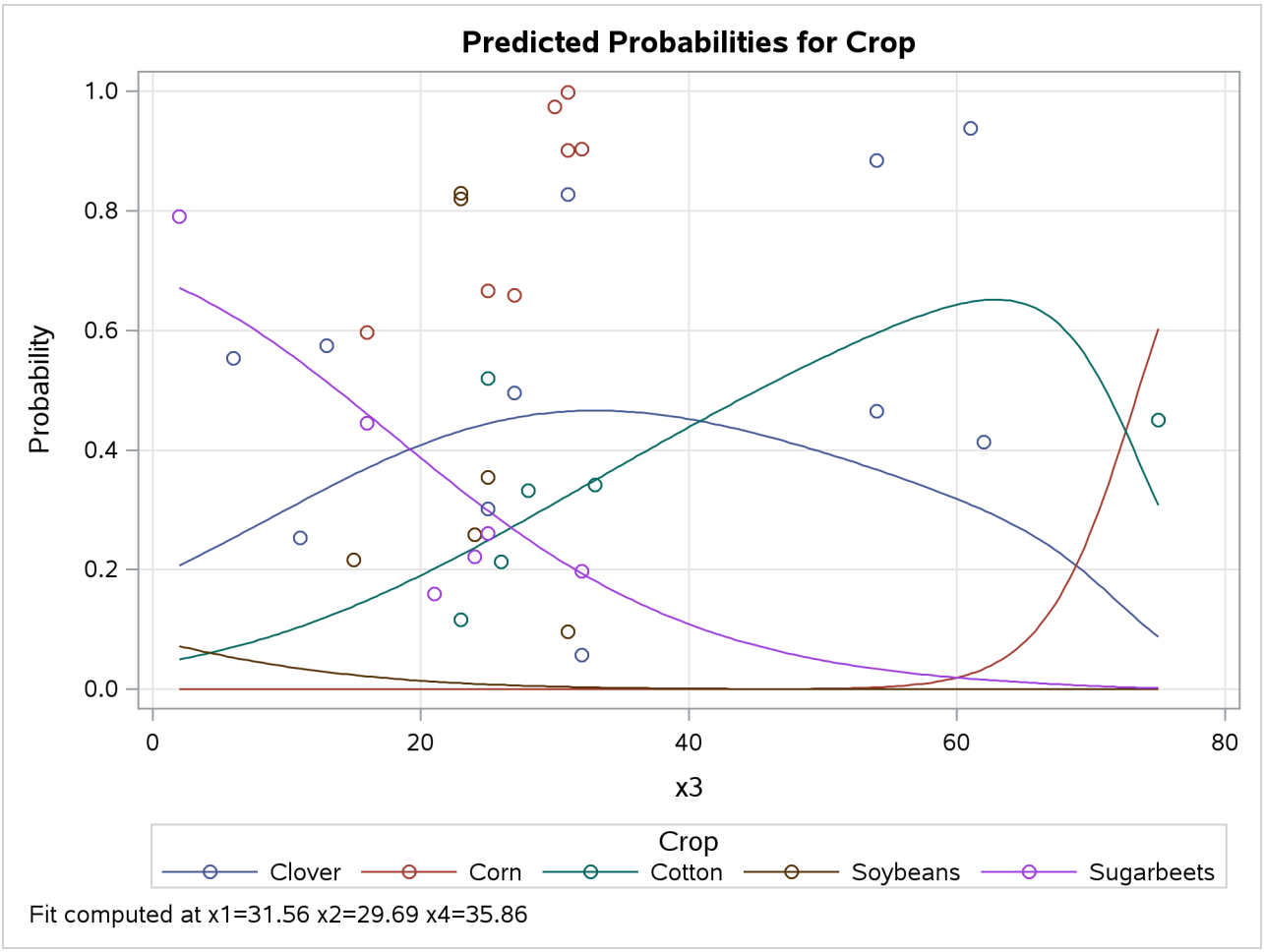

Example 79.17: Using the LSMEANS Statement

(View the complete code for this example (logiex17.sas) in the example repository.)

Recall the main-effects model fit to the Neuralgia data set in Example 79.2. The Treatment*Sex interaction, which was previously shown to be nonsignificant, is added back into the model for this discussion.

In the following statements, the ODDSRATIO statement is specified to produce odds ratios of pairwise differences of the Treatment parameters in the presence of the Sex interaction. The LSMEANS statement is specified with several options: the E option displays the coefficients that are used to compute the LS-means for each Treatment level, the DIFF option takes all pairwise differences of the LS-means for the levels of the Treatment variable, the ODDSRATIO option computes odds ratios of these differences, the CL option produces confidence intervals for the differences and odds ratios, and the ADJUST=BON option performs a very conservative adjustment of the p-values and confidence intervals.

```
proc logistic data=Neuralgia;
   class Treatment Sex / param=glm;
   model Pain= Treatment|Sex Age;
   oddsratio Treatment;
   lsmeans Treatment / e diff oddsratio cl adjust=bon;
run;
```

The results from the ODDSRATIO statement are displayed in Output 79.17.1. All pairwise differences of levels of the Treatment effect are compared. However, because of the interaction between the Treatment and Sex variables, each difference is computed at each of the two levels of the Sex variable. These results show that the difference between Treatment levels A and B is insignificant for both genders.

To compute these odds ratios, you must first construct a linear combination of the parameters, $\boldsymbol{l}'\boldsymbol{\beta}$, for each level that is compared with all other levels fixed at some value. For example, to compare Treatment=A with B for Sex=F, you fix the Age variable at its mean, 70.05, and construct the following $\boldsymbol{l}$ vectors:

		Treatment			Sex		Treatment*Sex						
	Intercept	A	B	P	F	M	AF	AM	BF	BM	PF	PM	Age
$\boldsymbol{l}'_A$	1	1	0	0	1	0	1	0	0	0	0	0	70.05
$\boldsymbol{l}'_B$	1	0	1	0	1	0	0	0	1	0	0	0	70.05
$\boldsymbol{l}'_A - \boldsymbol{l}'_B$	0	1	–1	0	0	0	1	0	–1	0	0	0	0

Then the odds ratio for Treatment A versus B at Sex=F is computed as $\exp((\boldsymbol{l}'_A - \boldsymbol{l}'_B)\boldsymbol{\beta})$. Different $\boldsymbol{l}$ vectors must be similarly constructed when Sex=M because the resulting odds ratio will be different due to the interaction.

Output 79.17.1 Odds Ratios from the ODDSRATIO Statement

Odds Ratio Estimates and Wald Confidence Intervals			
Odds Ratio	Estimate	95% Confidence Limits	
Treatment A vs B at Sex=F	0.398	0.016	9.722
Treatment A vs P at Sex=F	16.892	1.269	224.838
Treatment B vs P at Sex=F	42.492	2.276	793.254
Treatment A vs B at Sex=M	0.663	0.078	5.623
Treatment A vs P at Sex=M	34.766	1.807	668.724
Treatment B vs P at Sex=M	52.458	2.258	>999.999

The results from the LSMEANS statement are displayed in Output 79.17.2 through Output 79.17.4.

The LS-means are computed by constructing each of the $\boldsymbol{l}$ coefficient vectors shown in Output 79.17.2, and then computing $\boldsymbol{l}'\boldsymbol{\beta}$. The LS-means are not estimates of the event probabilities; they are estimates of the linear predictors on the logit scale and therefore are estimated log odds. In order to obtain event probabilities, you need to apply the inverse-link transformation by specifying the ILINK option in the LSMEANS statement. Notice in Output 79.17.2 that the Sex rows of the coefficient vectors do not select either Sex=F or Sex=M. Instead, the LS-means are computed at an average of these two levels, so only one result needs to be reported. For more information about the construction of LS-means, see the section "Construction of Least Squares Means" on page 4244 in Chapter 53, "The GLM Procedure."

Output 79.17.2 Treatment LS-Means Coefficients

Coefficients for Treatment Least Squares Means					
Parameter	**Treatment**	**Sex**	**Row1**	**Row2**	**Row3**
Intercept: Pain=No			1	1	1
Treatment A	A		1		
Treatment B	B			1	
Treatment P	P				1
Sex F		F	0.5	0.5	0.5
Sex M		M	0.5	0.5	0.5
Treatment A * Sex F	A	F	0.5		
Treatment A * Sex M	A	M	0.5		
Treatment B * Sex F	B	F		0.5	
Treatment B * Sex M	B	M		0.5	
Treatment P * Sex F	P	F			0.5
Treatment P * Sex M	P	M			0.5
Age			70.05	70.05	70.05

The Treatment LS-means shown in Output 79.17.3 are all significantly nonzero at the 0.05 level. These LS-means are *predicted population margins* of the logits; that is, they estimate the marginal means over a balanced population, and they are effectively the within-Treatment means appropriately adjusted for the other effects in the model. The LS-means are not event probabilities; in order to obtain event probabilities, you need to apply the inverse-link transformation by specifying the ILINK option in the LSMEANS statement. For more information about LS-means, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

Output 79.17.3 Treatment LS-Means

Treatment Least Squares Means							
Treatment	**Estimate**	**Standard Error**	**z Value**	**Pr > |z|**	**Alpha**	**Lower**	**Upper**
A	1.3195	0.6664	1.98	0.0477	0.05	0.01331	2.6257
B	1.9864	0.7874	2.52	0.0116	0.05	0.4431	3.5297
P	-1.8682	0.7620	-2.45	0.0142	0.05	-3.3618	-0.3747

Pairwise differences between the Treatment LS-means, requested with the DIFF option, are displayed in Output 79.17.4. The LS-mean for the level that is displayed in the _Treatment column is subtracted from the LS-mean for the level in the Treatment column, so the first row displays the LS-mean for Treatment level A minus the LS-mean for Treatment level B. The difference (–0.6669) is the estimated difference in log odds,

or equivalently the log odds ratio of the two treatments. The Pr > |z| column indicates that the A and B levels are not significantly different; however, both of these levels are significantly different from level P. If the inverse-link transformation is specified by the ILINK option, then these differences do not transform back to differences in probabilities.

There are two odds ratios for Treatment level A versus B in Output 79.17.1; these are constructed at each level of the interacting covariate Sex. In contrast, there is only one LS-means odds ratio for Treatment level A versus B in Output 79.17.4. This odds ratio is computed at an average of the interacting effects by creating the $\boldsymbol{l}$ vectors shown in Output 79.17.2 (the Row1 column corresponds to l_A, and the Row2 column corresponds to l_B) and computing $\exp(\boldsymbol{l}'_A\boldsymbol{\beta} - \boldsymbol{l}'_B\boldsymbol{\beta})$.

Because multiple tests are performed, you can protect yourself from falsely significant results by adjusting your p-values for multiplicity. The ADJUST=BON option performs the very conservative Bonferroni adjustment, and adds the columns labeled with 'Adj' to Output 79.17.4. By comparing the Pr > |z| column to the Adj P column, you can see that the p-values are adjusted upwards; in this case, there is no change in your conclusions. The confidence intervals are also adjusted for multiplicity—all adjusted intervals are wider than the unadjusted intervals, but again your conclusions in this example are unchanged. You can specify other adjustment methods by using the ADJUST= option.

Output 79.17.4 Differences and Odds Ratios for the Treatment LS-Means

Differences of Treatment Least Squares Means
Adjustment for Multiple Comparisons: Bonferroni

Treatment	_Treatment	Estimate	Standard Error	z Value	Pr > |z|	Adj P	Alpha	Lower	Upper	Adj Lower	Adj Upper	Odds Ratio
A	B	-0.6669	1.0026	-0.67	0.5059	1.0000	0.05	-2.6321	1.2982	-3.0672	1.7334	0.513
A	P	3.1877	1.0376	3.07	0.0021	0.0064	0.05	1.1541	5.2214	0.7037	5.6717	24.234
B	P	3.8547	1.2126	3.18	0.0015	0.0044	0.05	1.4780	6.2313	0.9517	6.7576	47.213

Differences of Treatment Least Squares Means
Adjustment for Multiple Comparisons: Bonferroni

Treatment	_Treatment	Lower Confidence Limit for Odds Ratio	Upper Confidence Limit for Odds Ratio	Adj Lower Odds Ratio	Adj Upper Odds Ratio
A	B	0.072	3.663	0.047	5.660
A	P	3.171	185.195	2.021	290.542
B	P	4.384	508.441	2.590	860.612

If you want to jointly test whether the active treatments are different from the placebo, you can specify a custom hypothesis test among LS-means by using the LSMESTIMATE statement. In the following statements, the LS-means for the two treatments are contrasted against the LS-mean of the placebo, and the JOINT option performs a joint test that the two treatments are not different from placebo.

```
proc logistic data=Neuralgia;
   class Treatment Sex / param=glm;
   model Pain= Treatment|Sex Age;
   lsmestimate treatment 1 0 -1, 0 1 -1 / joint;
run;
```

Output 79.17.5 displays the results from the LSMESTIMATE statement. The "Least Squares Means Estimates" table displays the differences of the two active treatments against the placebo, and the results are identical to the second and third rows of Output 79.17.4. The "Chi-Square Test for Least Squares Means Estimates"

table displays the joint test. In all of these tests, you reject the null hypothesis that the treatment has the same effect as the placebo.

Output 79.17.5 Custom LS-Mean Tests

Least Squares Means Estimates

Effect	Label	Estimate	Standard Error	z Value	Pr > |z|
Treatment	Row 1	3.1877	1.0376	3.07	0.0021
Treatment	Row 2	3.8547	1.2126	3.18	0.0015

Chi-Square Test for Least Squares Means Estimates

Effect	Num DF	Chi-Square	Pr > ChiSq
Treatment	2	12.13	0.0023

If you want to work with LS-means but you prefer to compute the Treatment odds ratios within the Sex levels in the same fashion as the ODDSRATIO statement does, you can specify the SLICE statement. In the following statements, you specify the same options in the SLICE statement as you do in the LSMEANS statement, except that you also specify the SLICEBY= option to perform an LS-means analysis partitioned into sets that are defined by the Sex variable:

```
proc logistic data=Neuralgia;
   class Treatment Sex / param=glm;
   model Pain= Treatment|Sex Age;
   slice Treatment*Sex / sliceby=Sex diff oddsratio cl adjust=bon;
run;
```

The results for Sex=F are displayed in Output 79.17.6 and Output 79.17.7. The joint test in Output 79.17.6 tests the equality of the LS-means of the levels of Treatment for Sex=F, and rejects equality at level 0.05. In Output 79.17.7, the odds ratios and confidence intervals match those reported for Sex=F in Output 79.17.1, and multiplicity adjustments are performed.

Output 79.17.6 Joint Test of Treatment Equality for Females

Chi-Square Test for Treatment*Sex Least Squares Means Slice

Slice	Num DF	Chi-Square	Pr > ChiSq
Sex F	2	8.22	0.0164

Output 79.17.7 Differences of the Treatment LS-Means for Females

Simple Differences of Treatment*Sex Least Squares Means
Adjustment for Multiple Comparisons: Bonferroni

Slice	Treatment	_Treatment	Estimate	Standard Error	z Value	Pr > |z|	Adj P	Alpha	Lower	Upper	Adj Lower	Adj Upper	Odds Ratio
Sex F	A	B	-0.9224	1.6311	-0.57	0.5717	1.0000	0.05	-4.1193	2.2744	-4.8272	2.9824	0.398
Sex F	A	P	2.8269	1.3207	2.14	0.0323	0.0970	0.05	0.2384	5.4154	-0.3348	5.9886	16.892
Sex F	B	P	3.7493	1.4933	2.51	0.0120	0.0361	0.05	0.8225	6.6761	0.1744	7.3243	42.492

Simple Differences of Treatment*Sex Least Squares Means
Adjustment for Multiple Comparisons: Bonferroni

Slice	Treatment	_Treatment	Lower Confidence Limit for Odds Ratio	Upper Confidence Limit for Odds Ratio	Adj Lower Odds Ratio	Adj Upper Odds Ratio
Sex F	A	B	0.016	9.722	0.008	19.734
Sex F	A	P	1.269	224.838	0.715	398.848
Sex F	B	P	2.276	793.254	1.190	>999.999

Similarly, the results for Sex=M are shown in Output 79.17.8 and Output 79.17.9.

Output 79.17.8 Joint Test of Treatment Equality for Males

Chi-Square Test for Treatment*Sex
Least Squares Means Slice

Slice	Num DF	Chi-Square	Pr > ChiSq
Sex M	2	6.64	0.0361

Output 79.17.9 Differences of the Treatment LS-Means for Males

Simple Differences of Treatment*Sex Least Squares Means
Adjustment for Multiple Comparisons: Bonferroni

Slice	Treatment	_Treatment	Estimate	Standard Error	z Value	Pr > |z|	Adj P	Alpha	Lower	Upper	Adj Lower	Adj Upper	Odds Ratio
Sex M	A	B	-0.4114	1.0910	-0.38	0.7061	1.0000	0.05	-2.5496	1.7268	-3.0231	2.2003	0.663
Sex M	A	P	3.5486	1.5086	2.35	0.0187	0.0560	0.05	0.5919	6.5054	-0.06286	7.1601	34.766
Sex M	B	P	3.9600	1.6049	2.47	0.0136	0.0408	0.05	0.8145	7.1055	0.1180	7.8021	52.458

Simple Differences of Treatment*Sex Least Squares Means
Adjustment for Multiple Comparisons: Bonferroni

Slice	Treatment	_Treatment	Lower Confidence Limit for Odds Ratio	Upper Confidence Limit for Odds Ratio	Adj Lower Odds Ratio	Adj Upper Odds Ratio
Sex M	A	B	0.078	5.623	0.049	9.028
Sex M	A	P	1.807	668.724	0.939	>999.999
Sex M	B	P	2.258	>999.999	1.125	>999.999

Example 79.18: Partial Proportional Odds Model

(View the complete code for this example (logiex18.sas) in the example repository.)

Cameron and Trivedi (1998, p. 68) studied the number of doctor visits from the Australian Health Survey 1977–78. The data set contains a dependent variable, dvisits, which contains the number of doctor visits in the past two weeks (0, 1, or 2, where 2 represents two or more visits) and the following explanatory variables: sex, which indicates whether the patient is female; age, which contains the patient's age in years divided by 100; income, which contains the patient's annual income (in units of $10,000); levyplus, which indicates whether the patient has private health insurance; freepoor, which indicates that the patient has free government health insurance due to low income; freerepa, which indicates that the patient has free government health insurance for other reasons; illness, which contains the number of illnesses in the past two weeks; actdays, which contains the number of days the illness caused reduced activity; hscore, which is a questionnaire score; chcond1, which indicates a chronic condition that does not limit activity; and chcond2, which indicates a chronic condition that limits activity.

```
data docvisit;
   input sex age agesq income levyplus freepoor freerepa
         illness actdays hscore chcond1 chcond2 dvisits;
   if ( dvisits > 2) then dvisits = 2;
   datalines;
1 0.19 0.0361 0.55  1  0  0  1  4  1  0  0  1
1 0.19 0.0361 0.45  1  0  0  1  2  1  0  0  1
0 0.19 0.0361 0.90  0  0  0  3  0  0  0  0  1

   ... more lines ...

1 0.37 0.1369 0.25  0  0  1  1  0  1  0  0  0
1 0.52 0.2704 0.65  0  0  0  0  0  0  0  0  0
0 0.72 0.5184 0.25  0  0  1  0  0  0  0  0  0
;
```

Because the response variable dvisits has three levels, the proportional odds model constructs two response functions. There is an intercept parameter for each of the two response functions, $\alpha_1 < \alpha_2$, and common slope parameters $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{12})$ across the functions. The model can be written as

$$\text{logit}(\Pr(Y \le i \mid \mathbf{x})) = \alpha_i + \boldsymbol{\beta}'\mathbf{x}, \quad i = 1, 2$$

The following statements fit a proportional odds model to this data:

```
proc logistic data=docvisit;
   model dvisits = sex age agesq income levyplus
                   freepoor freerepa illness actdays hscore
                   chcond1 chcond2;
run;
```

Selected results are displayed in Output 79.18.1.

Output 79.18.1 Test of Proportional Odds Assumption

Score Test for the Proportional Odds Assumption		
Chi-Square	DF	Pr > ChiSq
27.4256	12	0.0067

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	734.2971	12	<.0001
Score	811.8964	12	<.0001
Wald	690.7156	12	<.0001

The test of the proportional odds assumption in Output 79.18.1 rejects the null hypothesis that all the slopes are equal across the two response functions. This test is very anticonservative; that is, it tends to reject the null hypothesis even when the proportional odds assumption is reasonable.

The proportional odds assumption for ordinal response models can be relaxed by specifying the UNEQUALSLOPES option in the MODEL statement. A *general* (*nonproportional odds*) model has different slope parameters $\boldsymbol{\beta}_i = (\beta_{1,i}, \ldots, \beta_{12,i})$ for every logit i:

$$\mathrm{logit}(\Pr(Y \le i \mid \mathbf{x})) = \alpha_i + \boldsymbol{\beta}_i'\mathbf{x}, \quad i = 1, 2$$

The following statements fit the general model:

```
proc logistic data=docvisit;
   model dvisits = sex age agesq income levyplus
                   freepoor freerepa illness actdays hscore
                   chcond1 chcond2 / unequalslopes;
   sex:      test sex_0     =sex_1;
   age:      test age_0     =age_1;
   agesq:    test agesq_0   =agesq_1;
   income:   test income_0  =income_1;
   levyplus: test levyplus_0=levyplus_1;
   freepoor: test freepoor_0=freepoor_1;
   freerepa: test freerepa_0=freerepa_1;
   illness:  test illness_0 =illness_1;
   actdays:  test actdays_0 =actdays_1;
   hscore:   test hscore_0  =hscore_1;
   chcond1:  test chcond1_0 =chcond1_1;
   chcond2:  test chcond2_0 =chcond2_1;
run;
```

The TEST statements test the proportional odds assumption for each of the covariates in the model. The parameter names are constructed by appending the response level that identifies the response function, as described in the section "Parameter Names in the OUTEST= Data Set" on page 6076. Selected results from fitting the general model to the data are displayed in Output 79.18.2.

Output 79.18.2 Results for the General Model

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	761.4797	24	<.0001
Score	957.6793	24	<.0001
Wald	688.2306	24	<.0001

Analysis of Maximum Likelihood Estimates						
Parameter	**dvisits**	**DF**	**Estimate**	**Standard Error**	**Wald Chi-Square**	**Pr > ChiSq**
Intercept	**0**	1	2.3238	0.2754	71.2018	<.0001
Intercept	**1**	1	4.2862	0.4890	76.8368	<.0001
sex	**0**	1	-0.2637	0.0818	10.3909	0.0013
sex	**1**	1	-0.1232	0.1451	0.7210	0.3958
age	**0**	1	1.7489	1.5115	1.3389	0.2472
age	**1**	1	-2.0974	2.6003	0.6506	0.4199
agesq	**0**	1	-2.4718	1.6636	2.2076	0.1373
agesq	**1**	1	2.6883	2.8398	0.8961	0.3438
income	**0**	1	-0.00857	0.1266	0.0046	0.9460
income	**1**	1	0.6464	0.2375	7.4075	0.0065
levyplus	**0**	1	-0.2658	0.0997	7.0999	0.0077
levyplus	**1**	1	-0.2869	0.1820	2.4848	0.1150
freepoor	**0**	1	0.6773	0.2601	6.7811	0.0092
freepoor	**1**	1	0.9020	0.4911	3.3730	0.0663
freerepa	**0**	1	-0.4044	0.1382	8.5637	0.0034
freerepa	**1**	1	-0.0958	0.2361	0.1648	0.6848
illness	**0**	1	-0.2645	0.0287	84.6792	<.0001
illness	**1**	1	-0.3083	0.0499	38.1652	<.0001
actdays	**0**	1	-0.1521	0.0116	172.2764	<.0001
actdays	**1**	1	-0.1863	0.0134	193.7700	<.0001
hscore	**0**	1	-0.0620	0.0172	12.9996	0.0003
hscore	**1**	1	-0.0568	0.0252	5.0940	0.0240
chcond1	**0**	1	-0.1140	0.0909	1.5721	0.2099
chcond1	**1**	1	-0.2478	0.1743	2.0201	0.1552
chcond2	**0**	1	-0.2660	0.1255	4.4918	0.0341
chcond2	**1**	1	-0.3146	0.2116	2.2106	0.1371

Output 79.18.2 *continued*

Linear Hypotheses Testing Results			
Label	Wald Chi-Square	DF	Pr > ChiSq
sex	1.0981	1	0.2947
age	2.5658	1	0.1092
agesq	3.8309	1	0.0503
income	8.8006	1	0.0030
levyplus	0.0162	1	0.8989
freepoor	0.2569	1	0.6122
freerepa	2.0099	1	0.1563
illness	0.8630	1	0.3529
actdays	6.9407	1	0.0084
hscore	0.0476	1	0.8273
chcond1	0.6906	1	0.4060
chcond2	0.0615	1	0.8042

The preceding general model fits $12 \times 2 = 24$ slope parameters, and according to the "Linear Hypotheses Testing Results" table in Output 79.18.2, several variables are unnecessarily allowing nonproportional odds. You can obtain a more parsimonious model by specifying a subset of the parameters to have nonproportional odds. The following statements allow the parameters for the variables in the "Linear Hypotheses Testing Results" table that have p-values less than 0.1 (actdays, agesq, and income) to vary across the response functions:

```
proc logistic data=docvisit;
   model dvisits= sex age agesq income levyplus freepoor
         freerepa illness actdays hscore chcond1 chcond2
   / unequalslopes=(actdays agesq income);
run;
```

Selected results from fitting this partial proportional odds model are displayed in Output 79.18.3.

Output 79.18.3 Results for Partial Proportional Odds Model

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	752.5512	15	<.0001
Score	947.3269	15	<.0001
Wald	683.4719	15	<.0001

Output 79.18.3 *continued*

Analysis of Maximum Likelihood Estimates						
Parameter	dvisits	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	0	1	2.3882	0.2716	77.2988	<.0001
Intercept	1	1	3.7597	0.3138	143.5386	<.0001
sex		1	-0.2485	0.0807	9.4789	0.0021
age		1	1.3000	1.4864	0.7649	0.3818
agesq	0	1	-2.0110	1.6345	1.5139	0.2186
agesq	1	1	-0.8789	1.6512	0.2833	0.5945
income	0	1	0.0209	0.1261	0.0275	0.8683
income	1	1	0.4283	0.2221	3.7190	0.0538
levyplus		1	-0.2703	0.0989	7.4735	0.0063
freepoor		1	0.6936	0.2589	7.1785	0.0074
freerepa		1	-0.3648	0.1358	7.2155	0.0072
illness		1	-0.2707	0.0281	92.7123	<.0001
actdays	0	1	-0.1522	0.0115	173.5696	<.0001
actdays	1	1	-0.1868	0.0129	209.7134	<.0001
hscore		1	-0.0609	0.0166	13.5137	0.0002
chcond1		1	-0.1200	0.0901	1.7756	0.1827
chcond2		1	-0.2628	0.1227	4.5849	0.0323

You can also specify the following code to let stepwise selection determine which parameters have unequal slopes:

```
proc logistic data=docvisit;
   model dvisits= sex age agesq income levyplus freepoor
         freerepa illness actdays hscore chcond1 chcond2
   / equalslopes unequalslopes selection=stepwise details;
run;
```

This selection process chooses sex, freepoor, illness, and hscore as the proportional odds effects and chooses U_actdays and U_agesq as the unconstrained general effects. For more information about model selection for partial proportional odds models, see Derr (2013).

The partial proportional odds model can be written in the same form as the general model by letting $\mathbf{x} = (x_1, \ldots, x_q, x_{q+1}, \ldots, x_{12})$ and $\boldsymbol{\beta}_i = (\beta_1, \ldots, \beta_q, \beta_{q+1,i}, \ldots, \beta_{12,i})$. So the first q parameters have proportional odds and the remaining parameters do not. The last $12 - q$ parameters can be rewritten to have a common slope plus an increment from the common slope: $\beta_{q+j} + \gamma_{q+j,i}$, $j = 1, \ldots, 12 - q$, where the new parameters $\boldsymbol{\gamma}_i$ contain the increments from the common slopes. The model in this form makes it obvious that the proportional odds model is a submodel of the partial proportional odds model, and both of these are submodels of the general model. This means that you can use likelihood ratio tests to compare models.

You can use the following statements to compute the likelihood ratio tests from the Likelihood Ratio row of the "Testing Global Null hypothesis: BETA=0" tables in the preceding outputs:

```
data a;
   label p='Pr>ChiSq';
   format p 8.6;
   input Test $10. ChiSq1 DF1 ChiSq2 DF2;
   ChiSq= ChiSq1-ChiSq2;
```

```
   DF= DF1-DF2;
   p=1-probchi(ChiSq,DF);
   keep Test Chisq DF p;
   datalines;
Gen vs PO     761.4797 24    734.2971 12
PPO vs PO     752.5512 15    734.2971 12
Gen vs PPO    761.4797 24    752.5512 15
;

proc print data=a label noobs;
   var Test ChiSq DF p;
run;
```

Output 79.18.4 Likelihood Ratio Tests

Test	ChiSq	DF	Pr>ChiSq
Gen vs PO	27.1826	12	0.007273
PPO vs PO	18.2541	3	0.000390
Gen vs PPO	8.9285	9	0.443900

Therefore, you reject the proportional odds model in favor of both the general model and the partial proportional odds model, and the partial proportional odds model fits as well as the general model. The likelihood ratio test of the general model versus the proportional odds model is very similar to the score test of the proportional odds assumption in Output 79.18.1 because of the large sample size (Stokes, Davis, and Koch 2000, p. 249).

NOTE: The proportional odds model has increasing intercepts, which ensures the increasing nature of the cumulative response functions. However, none of the parameters in the partial proportional odds or general models are constrained. Because of this, sometimes during the optimization process a predicted individual probability can be negative; the optimization continues because it might recover from this situation. Sometimes your final model will predict negative individual probabilities for some of the observations; in this case a message is displayed, and you should check your data for outliers and possibly redefine your model. Other times the model fits your data well, but if you try to score new data you can get negative individual probabilities. This means the model is not appropriate for the data you are trying to score, a message is displayed, and the estimates are set to missing. These optimization issues do not arise when you fit an adjacent-category or generalized logit model that has equal and unequal slope effects.

Example 79.19: Goodness-of-Fit Tests and Calibration

(View the complete code for this example (logiex19.sas) in the example repository.)

The following data were originally published by Mroz (1987) and downloaded from Wooldridge (2002). This data set is based on a sample of 753 married white women. The dependent variable is a discrete variable of labor force participation (InLaborForce). Explanatory variables are the number of children age 5 or younger (KidsLt6); the number of children ages 6 to 18 (KidsGe6); the woman's age (Age); the woman's years of schooling (Education); the wife's labor experience (Experience); the square of experience (SqExperience); and the family income, excluding the wife's wages (IncomeExcl).

```
data Mroz;
   input InLaborForce IncomeExcl Education Experience
         SqExperience Age KidsLt6 KidsGe6 lWage;
   datalines;
1 10.91006    12 14  196   32  1   0   1.210154
1 19.49998    12 5   25    30  0   2   0.3285121

   ... more lines ...

0 24.984      12 15  225   60  0   0   .
0 28.363      9  12  144   39  0   3   .
;
```

The following program sets aside about 30% of the data set to be used for testing the model. To do this, a new response variable, InLaborForce2, is set to missing for the test data; otherwise it is set to the original value of InLaborForce.

```
data Mroz;
   set Mroz;
   if (ranuni(3939)<.3) then InLaborForce2=.;
   else InLaborForce2=InLaborForce;
run;
```

In the following statements, the GOF option modifies the "Model Fitting Statistics" table by displaying a single column with more statistics in Output 79.19.1, displays more tests in the "Goodness-of-Fit" table, displays the results of the Hosmer-Lemeshow test, and displays a calibration plot because ODS Graphics is enabled. The SHOWOBS option modifies the calibration plot by displaying a bar chart of the events in the upper margin of the plot and a bar chart of the nonevents in the bottom margin. The CLM option displays confidence intervals for the mean loess fit as a band around the loess curve. The OUTPUT statement creates a data set that contains the linear predictors and the original InLaborForce variable.

```
ods graphics on;
proc logistic data=Mroz plots=calibration(clm showobs);
   model InLaborForce2 = IncomeExcl Education Experience
                         SqExperience Age KidsLt6 KidsGe6  / gof;
   id InLaborForce;
   output out=Scored xbeta=XBeta;
run;
```

Because of the continuous predictors, the Pearson and deviance tests in Output 79.19.1 cannot be trusted. However, all the other tests remain valid and do not reject the null hypothesis that the fit is adequate.

Output 79.19.1 Goodness-of-Fit Tests

Goodness-of-Fit Tests			
Criterion	Value	DF	Pr > ChiSq
Deviance	606.5035	544	0.0324
Pearson	538.4877	544	0.5586
Information Matrix	38.2455	35	0.3243
Information Matrix Diagonal	11.4027	8	0.1799
Osius and Rojek	0.8412	1	0.3590
Residual Sum of Squares	1.3502	1	0.2453
Spiegelhalter	0.0201	1	0.8871
Stukel	2.2581	2	0.3233

Number of unique profiles: 552

The "Model Fit Statistics" table displays an assortment of statistics in Output 79.19.2 that are most useful when you compare this model to other models. The misclassification rate and the average square error are of moderate size.

Output 79.19.2 Model Fit Statistics

Model Fit Statistics	
Criterion	Value
AIC	622.5035
AICC	622.7687
SC	657.0119
-2 Log L	606.5035
-2 Log L (Intercept-Only)	755.2843
Misclassification Rate	0.2754
Average Square Error	0.1862
R-Square	0.2363
Max-rescaled R-Square	0.3169
Aldrich-Nelson's R-Square	0.2123
Efron's R-Square	0.2417
Estrella's R-Square	0.2593
Estrella's Adjusted R-Square	0.2324
McFadden's R-Square	0.1970
Tjur's R-Square	0.2440
Veall-Zimmerman's R-Square	0.3675

The Hosmer-Lemeshow test is computed using 10 groups for an 8-degrees-of-freedom chi-square test, and you do not reject the null hypothesis that the fit is adequate. The results are shown in Output 79.19.3.

Output 79.19.3 Hosmer-Lemeshow Test

Partition for the Hosmer and Lemeshow Test					
		InLaborForce2 = 0		InLaborForce2 = 1	
Group	Total	Observed	Expected	Observed	Expected
1	55	3	5.22	52	49.78
2	55	10	9.12	45	45.88
3	55	10	12.24	45	42.76
4	55	20	15.85	35	39.15
5	55	24	19.39	31	35.61
6	55	17	23.44	38	31.56
7	55	36	28.84	19	26.16
8	55	30	34.31	25	20.69
9	55	40	41.02	15	13.98
10	57	49	49.58	8	7.42

Hosmer and Lemeshow Goodness-of-Fit Test		
Chi-Square	DF	Pr > ChiSq
13.2967	8	0.1020

The calibration plot in Output 79.19.4 shows you how well the predicted probabilities align with the observed responses. The smoothing parameter that the procedure selects seems rather small; you can change the smoothing parameter for the loess fit by specifying the PLOTS=CALIBRATION(SMOOTH=) option. The confidence band is not a simultaneous band, so you cannot use the fact that the band does or does not contain the diagonal line as a calibration test. The displayed Spiegelhalter z statistic is a standard normal; its squared value is reported in Output 79.19.1.

Output 79.19.4 Calibration Plot for Training Model

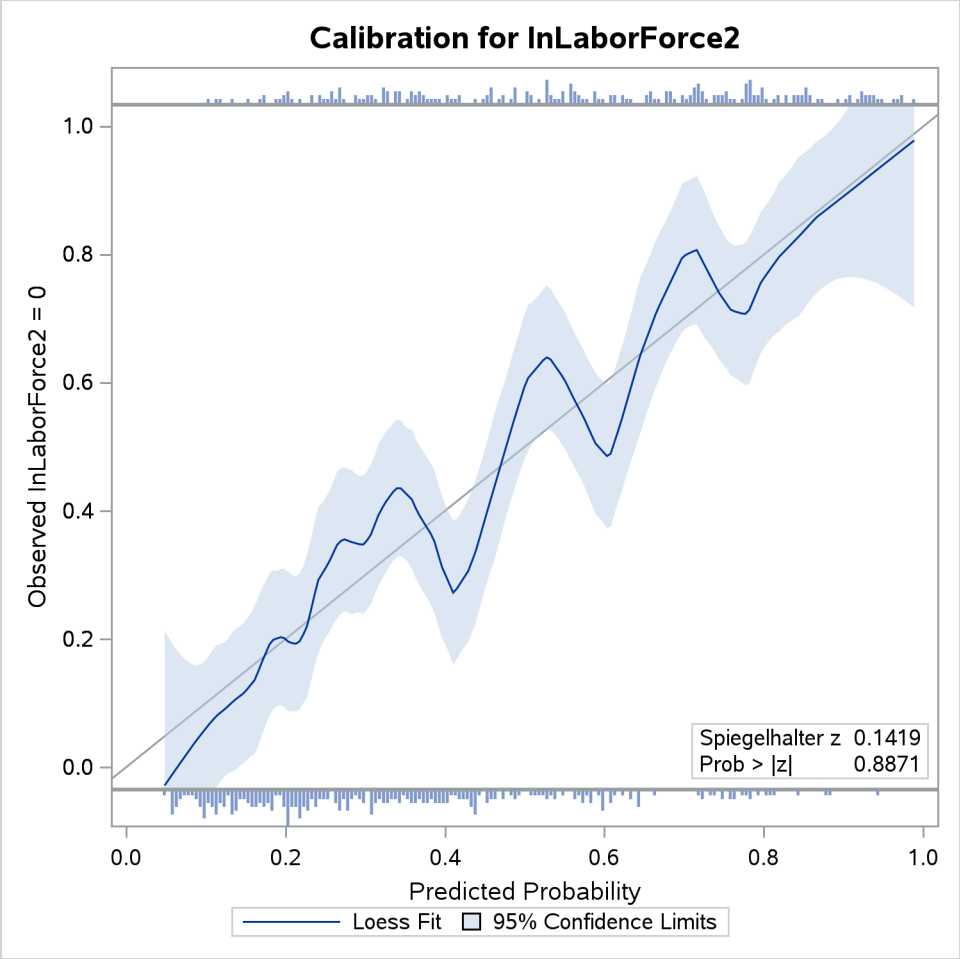

Now that the model has been fit to the training data, refit the testing data by using the original response variable versus the linear predictor (Harrell 2015). Ideally, the intercept parameter should be near 0, and the slope parameter should be near 1. In the following statements, the WHERE data set option extracts the test data from the Scored data set, and the TEST statement performs a Wald test of this joint hypothesis. The calibration plot of this fit should also be close to the diagonal.

```
proc logistic data=Scored(where=(InLaborForce2=.))
              plots=calibration(clm showobs);
   model InLaborForce = XBeta;
   test intercept = 0, XBeta = 1;
run;
```

The TEST result in Output 79.19.5 shows that you do not reject the null that the intercept parameter is 0 and the slope parameter is 1.

Output 79.19.5 Calibration Test for Test Data

Linear Hypotheses Testing Results			
Label	Wald Chi-Square	DF	Pr > ChiSq
Test 1	2.7568	2	0.2520

The calibration plot displayed in Output 79.19.6 follows the diagonal quite well.

Output 79.19.6 Calibration Plot for Test Data

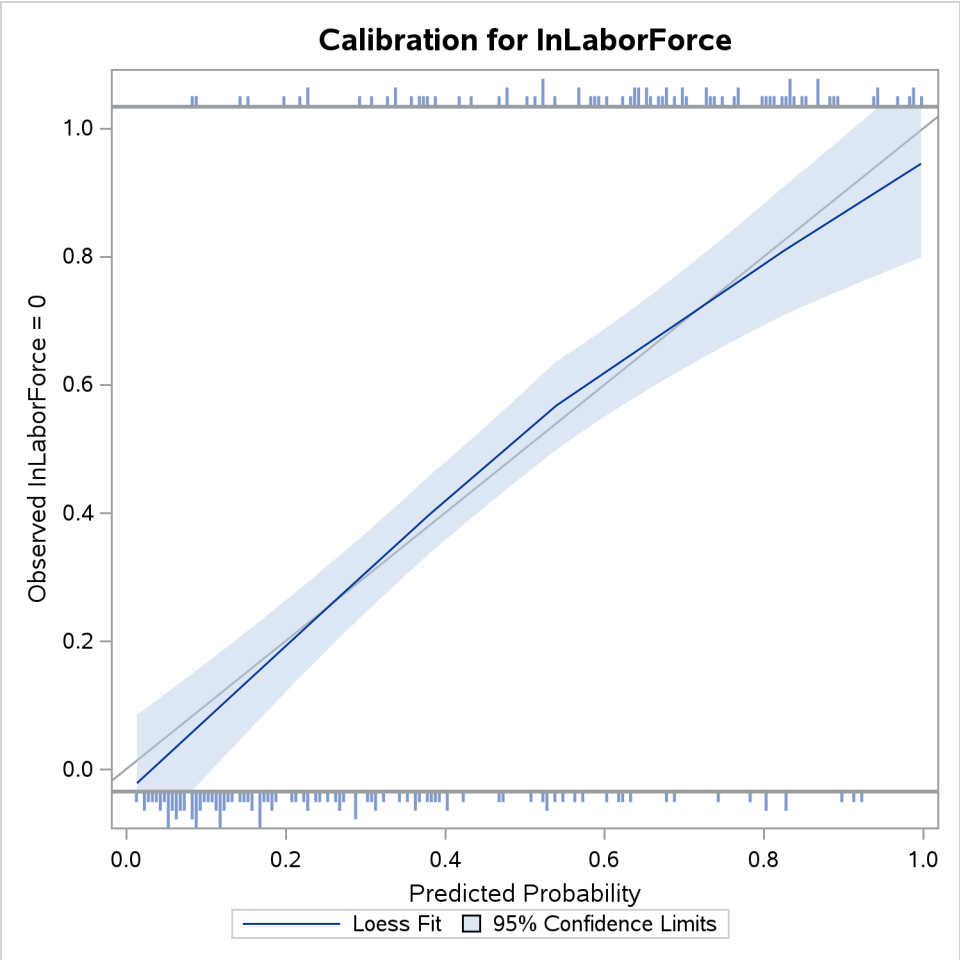

Example 79.20: Using the Optimal ROC Criteria

(View the complete code for this example (logiex20.sas) in the example repository.)

Suppose you want to classify observations on the basis of your model fit to the Mroz training data set as described in Example 79.19. A standard approach is to compute the model-predicted probabilities for each observation, then assign observations whose probabilities are greater than 0.5 to be events and assign the rest to be nonevents. This example compares this method to approaches that are based on the computation of optimal statistics on the ROC curve.

```
data Mroz;
   input InLaborForce IncomeExcl Education Experience
         SqExperience Age KidsLt6 KidsGe6 lWage;
   datalines;
1 10.91006    12 14  196   32  1   0   1.210154
1 19.49998    12 5   25    30  0   2   0.3285121

   ... more lines ...

0 24.984      12 15  225   60  0   0   .
0 28.363      9  12  144   39  0   3   .
;

data Mroz;
   set Mroz;
   if (ranuni(3939)<.3) then InLaborForce2=.;
   else InLaborForce2=InLaborForce;
run;
```

The following program again uses the training data to fit a main-effects model to the InLaborForce2 variable. The PLOTS option requests both an ROC plot and a precision-recall plot; the precision-recall curve is often used in cases where you have a large imbalance in the number of events and nonevents in your data set. In the ROCOPTIONS option, you specify the BINORMAL method because you believe the event and nonevent probabilities are approximately normally distributed. The CROSSVALIDATE option computes cross validated predicted probabilities instead of the usual model-predicted probabilities. The ID=PROB option displays the predicted probabilities in the plot. The OPTIMAL option computes two different optimal-threshold statistics: the Youden statistic finds the probability that separates the true and false positives by the largest value, and the cost statistic optimizes a criterion in which the cost of a correct decision is zero but the cost of a false negative is three times that of a false positive. The AREA option displays a table that contains the area under the binormal ROC, empirical ROC, and precision-recall curves. The OUTROC= option creates an output data set that contains the ROC values, and the DATA _NULL_ step displays the optimal thresholds in the log.

```
ods graphics on;
proc logistic data=Mroz plots(only)=(roc pr)
   rocoptions(method=binormal area crossvalidate id=prob
              optimal=(youden cost(tp=0 tn=0 fp=1 fn=3)));
   model InLaborForce2(event='0') = IncomeExcl Education Experience
                          SqExperience Age KidsLt6 KidsGe6 / outroc=outr;
   id InLaborForce;
   output out=out predprobs=x;
run;
```

```
data _null_;
   set outr;
   if _optcost_=1 | _optyouden_=1;
   put _optcost_ _optyouden_ _prob_;
run;
```

The areas under the various curves are shown in Output 79.20.1.

Output 79.20.1 Areas under the Curves

Areas under the Curves	
Empirical ROC	0.7703
Binormal ROC	0.7723
Precision-Recall	0.6941

The binormal ROC curve shown in Output 79.20.2 is much smoother than the empirical curve. The predicted-probability threshold that corresponds to the maximum Youden index is 0.38970, whereas the threshold that corresponds to the cost criterion is 0.22976.

Output 79.20.2 ROC Curve for Mroz Model

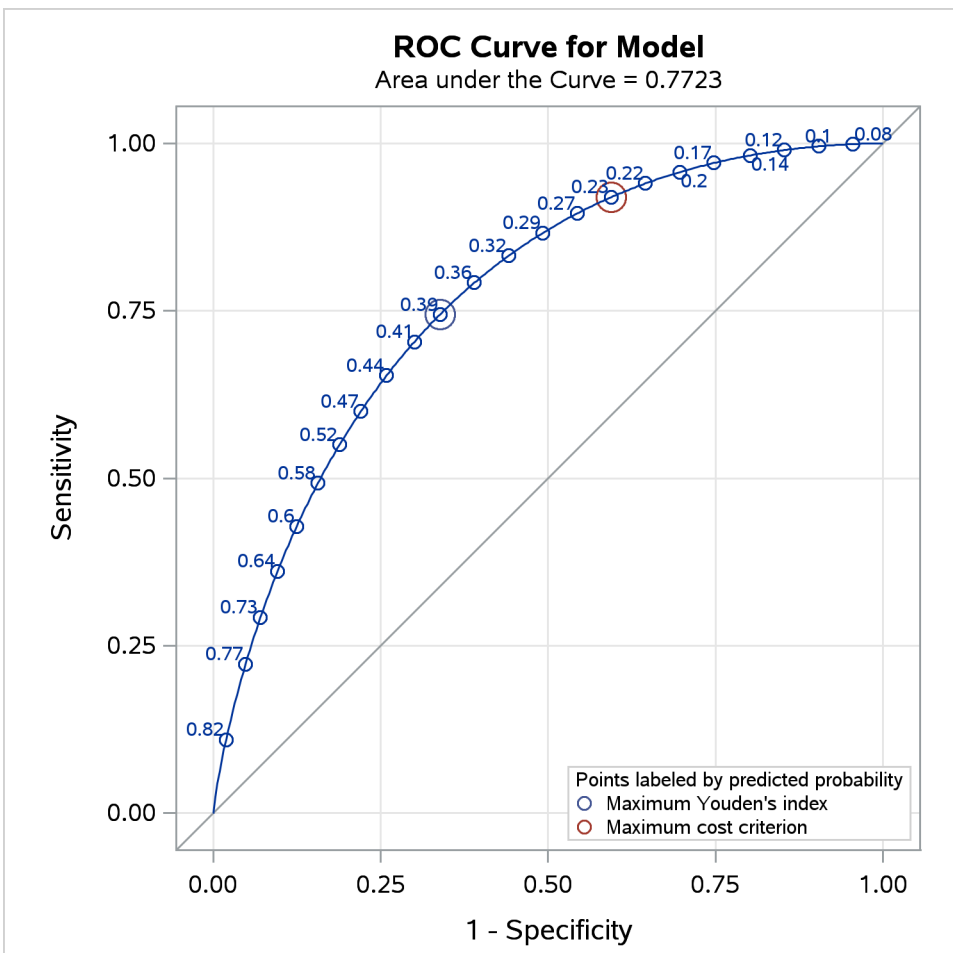

The following program counts the number of false positives and false negatives in the testing data when you use these thresholds. The results are summarized in Table 79.24, where you can see that although more total mistakes are made when the cost criterion is used, the number of false negatives is much smaller because of their higher cost.

```
data out;
   set out;
   if InLaborForce2=.;
   standard =(xp_0>0.50);
   youden=(xp_0>0.38970);
   cost  =(xp_0>0.22976);
run;

data out;
   set out end=eof;
   retain FNstandard FPstandard FNyouden FPyouden FNcost FPcost 0;
   if (InLaborForce=1) then do;
      if (youden=1)   then FPyouden=FPyouden+1;
      if (cost=1)     then FPcost=FPcost+1;
      if (standard=1) then FPstandard=FPstandard+1;
   end;
   if (InLaborForce=0) then do;
      if (youden=0)   then FNyouden=FNyouden+1;
      if (cost=0)     then FNcost=FNcost+1;
      if (standard=0) then FNstandard=FNstandard+1;
   end;
   if (eof) then output;
proc print;
run;
```

Table 79.24 Number of False Negatives and False Positives

Threshold	False Negatives	False Positives	Total Incorrect
Standard	23	23	46
Youden	13	35	48
Cost	4	64	68

The precision-recall curve is displayed in Output 79.20.3.

Output 79.20.3 Precision-Recall Curve for Mroz Model

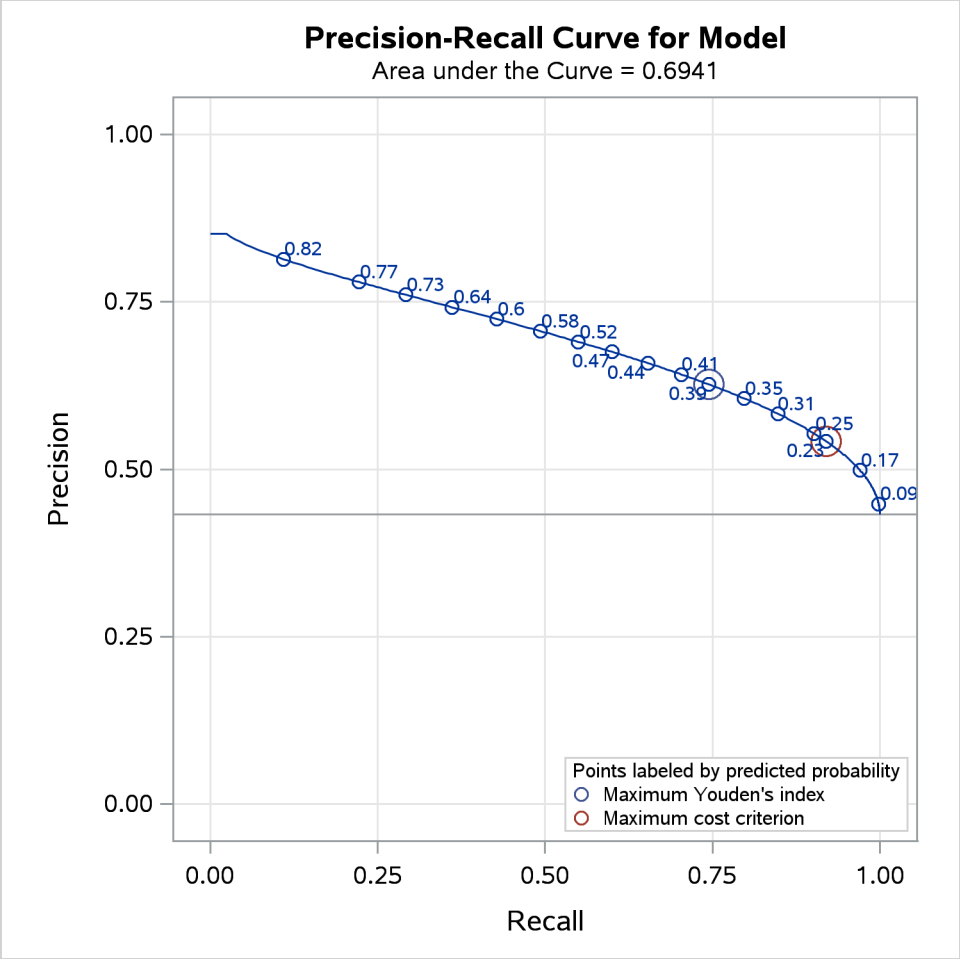

References

Agresti, A. (1990). *Categorical Data Analysis*. New York: John Wiley & Sons.

Agresti, A. (1992). "A Survey of Exact Inference for Contingency Tables." *Statistical Science* 7:131–177.

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Agresti, A. (2010). *Analysis of Ordinal Categorical Data*. 2nd ed. New York: John Wiley & Sons.

Agresti, A. (2013). *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Aitchison, J., and Silvey, S. (1957). "The Generalization of Probit Analysis to the Case of Multiple Responses." *Biometrika* 44:131–140.

Albert, A., and Anderson, J. A. (1984). "On the Existence of Maximum Likelihood Estimates in Logistic Regression Models." *Biometrika* 71:1–10.

Aldrich, J. H., and Nelson, F. D. (1984). *Linear Probability, Logit, and Probit Models*, vol. 45 of *Sage University Paper Series on Quantitative Applications in the Social Sciences*. Newbury Park, CA: Sage Publications.

Allison, P. D. (1982). "Discrete-Time Methods for the Analysis of Event Histories." In *Sociological Methods and Research*, edited by S. Leinhardt, 61–98. San Francisco: Jossey-Bass.

Allison, P. D. (1999). *Logistic Regression Using the SAS System: Theory and Application*. Cary, NC: SAS Institute Inc.

Allison, P. D. (2012). *Logistic Regression Using SAS: Theory and Application*. 2nd ed. Cary, NC: SAS Institute Inc.

Allison, P. D. (2014). "Measures of Fit for Logistic Regression." In *Proceedings of the SAS Global Forum 2014 Conference*, 1–12. Cary, NC: SAS Institute Inc.

Ashford, J. R. (1959). "An Approach to the Analysis of Data for Semi-quantal Responses in Biology Response." *Biometrics* 15:573–581.

Bartolucci, A. A., and Fraser, M. D. (1977). "Comparative Step-Up and Composite Test for Selecting Prognostic Indicator Associated with Survival." *Biometrical Journal* 19:437–448.

Breslow, N. E., and Day, N. E. (1980). *The Analysis of Case-Control Studies*. Statistical Methods in Cancer Research, IARC Scientific Publications, vol. 1, no. 32. Lyon: International Agency for Research on Cancer.

Brier, G. W. (1950). "Verification of Forecasts Expressed in Terms of Probability." *Monthly Weather Review* 78:1–3.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Cameron, A. C., and Trivedi, P. K. (1998). *Regression Analysis of Count Data*. Cambridge: Cambridge University Press.

Chang, I.-M., Gelman, R., and Pagano, M. (1982). "Corrected Group Prognostic Curves and Summary Statistics." *Journal of Chronic Diseases* 35:669–674.

Collett, D. (2003). *Modelling Binary Data*. 2nd ed. London: Chapman & Hall.

Cook, R. D., and Weisberg, S. (1982). *Residuals and Influence in Regression*. New York: Chapman & Hall.

Copas, J. B. (1989). "Unweighted Sum of Squares Test for Proportions." *Journal of the Royal Statistical Society, Series C* 38:71–80.

Cox, D. R. (1970). *The Analysis of Binary Data*. New York: Chapman & Hall.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220. With discussion.

Cox, D. R., and Snell, E. J. (1989). *The Analysis of Binary Data*. 2nd ed. London: Chapman & Hall.

Cragg, J. G., and Uhler, R. S. (1970). "The Demand for Automobiles." *Canadian Journal of Economics* 3:386–406.

Davis, J., and Goadrich, M. (2006). "The Relationship between Precision-Recall and ROC Curves." In *Proceedings of the 23rd International Conference on Machine Learning*, 233–240. New York: ACM.

DeLong, E. R., DeLong, D. M., and Clarke-Pearson, D. L. (1988). "Comparing the Areas under Two or More Correlated Receiver Operating Characteristic Curves: A Nonparametric Approach." *Biometrics* 44:837–845.

Demler, O. V., Pencina, M. J., and D'Agostino, R. B. (2012). "Misuse of DeLong Test to Compare AUCs for Nested Models." *Statistics in Medicine* 31:2577–2587.

Derr, B. (2013). "Ordinal Response Modeling with the LOGISTIC Procedure." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings13/446-2013.pdf.

Draper, C. C., Voller, A., and Carpenter, R. G. (1972). "The Epidemiologic Interpretation of Serologic Data in Malaria." *American Journal of Tropical Medicine and Hygiene* 21:696–703.

Efron, B., and Hinkley, D. V. (1978). "Assessing the Accuracy of the Maximum Likelihood Estimator: Observed versus Expected Fisher Information." *Biometrika* 65:457–483.

Estrella, A. (1998). "A New Measure of Fit for Equations with Dichotomous Dependent Variables." *Journal of Business and Economic Statistics* 16:198–205.

Fagerland, M. W., and Hosmer, D. W. (2013). "A Goodness-of-Fit Test for the Proportional Odds Regression Model." *Statistics in Medicine* 32:2235–2249.

Fagerland, M. W., and Hosmer, D. W. (2016). "Tests for Goodness of Fit in Ordinal Logistic Regression Models." *Journal of Statistical Computation and Simulation* 86:3398–3418.

Fagerland, M. W., Hosmer, D. W., and Bofin, A. M. (2008). "Multinomial Goodness-of-Fit Tests for Logistic Regression Models." *Statistics in Medicine* 27:4238–4253.

Finney, D. J. (1947). "The Estimation from Individual Records of the Relationship between Dose and Quantal Response." *Biometrika* 34:320–334.

Firth, D. (1993). "Bias Reduction of Maximum Likelihood Estimates." *Biometrika* 80:27–38.

Fleiss, J. L. (1981). *Statistical Methods for Rates and Proportions*. 2nd ed. New York: John Wiley & Sons.

Fleiss, J. L., Levin, B., and Paik, M. C. (2003). *Statistical Methods for Rates and Proportions*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Forster, J. J., McDonald, J. W., and Smith, P. W. F. (2003). "Markov Chain Monte Carlo Exact Inference for Binomial and Multinomial Logistic Regression Models." *Statistics and Computing* 13:169–177.

Freeman, D. H., Jr. (1987). *Applied Categorical Data Analysis*. New York: Marcel Dekker.

Furnival, G. M., and Wilson, R. W. (1974). "Regression by Leaps and Bounds." *Technometrics* 16:499–511.

Gail, M. H., Lubin, J. H., and Rubinstein, L. V. (1981). "Likelihood Calculations for Matched Case-Control Studies and Survival Studies with Tied Death Times." *Biometrika* 68:703–707.

Hand, D. J., Daly, F., Lunn, A. D., McConway, K. J., and Ostrowski, E. (1994). *A Handbook of Small Data Sets*. London: Chapman & Hall.

Hanley, J. A., and McNeil, B. J. (1982). "The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve." *Radiology* 143:29–36.

Harrell, F. E. (1986). "The LOGIST Procedure." In *SUGI Supplemental Library Guide, Version 5 Edition*. Cary, NC: SAS Institute Inc.

Harrell, F. E., Jr. (2015). *Regression Modeling Strategies: With Applications to Linear Models, Logistic Regression, and Survival Analysis*. 2nd ed. New York: Springer.

Heinze, G. (1999). *The Application of Firth's Procedure to Cox and Logistic Regression*. Technical Report 10, updated January 2001, Department of Medical Computer Sciences, Section of Clinical Biometrics, University of Vienna.

Heinze, G. (2006). "A Comparative Investigation of Methods for Logistic Regression with Separated or Nearly Separated Data." *Statistics in Medicine* 25:4216–4226.

Heinze, G., and Schemper, M. (2002). "A Solution to the Problem of Separation in Logistic Regression." *Statistics in Medicine* 21:2409–2419.

Hilbe, J. M. (2009). *Logistic Regression Models*. London: Chapman & Hall/CRC.

Hirji, K. F. (1992). "Computing Exact Distributions for Polytomous Response Data." *Journal of the American Statistical Association* 87:487–492.

Hirji, K. F. (2006). *Exact Analysis of Discrete Data*. Boca Raton, FL: Chapman & Hall/CRC.

Hirji, K. F., Mehta, C. R., and Patel, N. R. (1987). "Computing Distributions for Exact Logistic Regression." *Journal of the American Statistical Association* 82:1110–1117.

Hirji, K. F., Mehta, C. R., and Patel, N. R. (1988). "Exact Inference for Matched Case-Control Studies." *Biometrics* 44:803–814.

Hirji, K. F., Tsiatis, A. A., and Mehta, C. R. (1989). "Median Unbiased Estimation for Binary Data." *American Statistician* 43:7–11.

Hosmer, D. W., Hosmer, T., Le Cessie, S., and Lemeshow, S. (1997). "A Comparison of Goodness-of-Fit Tests for the Logistic Regression Model." *Statistics in Medicine* 16:965–980.

Hosmer, D. W., Jr., and Lemeshow, S. (2000). *Applied Logistic Regression*. 2nd ed. New York: John Wiley & Sons.

Hosmer, D. W., Jr., and Lemeshow, S. (2013). *Applied Logistic Regression*. 3rd ed. New York: John Wiley & Sons.

Howard, S. (1972). "Remark on the Paper by Cox ('Regression Methods and Life Tables')." *Journal of the Royal Statistical Society, Series B* 34:187–220.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Hurvich, C. M., and Tsai, C.-L. (1993). "A Corrected Akaike Information Criterion for Vector Autoregressive Model Selection." *Journal of Time Series Analysis* 14:271–279.

Izrael, D., Battaglia, A. A., Hoaglin, D. C., and Battaglia, M. P. (2002). "Use of the ROC Curve and the Bootstrap in Comparing Weighted Logistic Regression Models." In *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi27/p248-27.pdf.

Kuss, O. (2002). "Global Goodness-of-Fit Tests in Logistic Regression with Sparse Data." *Statistics in Medicine* 21:3789–3801.

Lachin, J. M. (2000). *Biostatistical Methods: The Assessment of Relative Risks*. New York: John Wiley & Sons.

LaMotte, L. R. (2002). Personal communication, June.

Lancaster, H. O. (1961). "Significance Tests in Discrete Distributions." *Journal of the American Statistical Association* 56:223–234.

Lane, P. W., and Nelder, J. A. (1982). "Analysis of Covariance and Standardization as Instances of Prediction." *Biometrics* 38:613–621.

Lave, C. A. (1970). "The Demand for Urban Mass Transportation." *Review of Economics and Statistics* 52:320–323.

Lawless, J. F., and Singhal, K. (1978). "Efficient Screening of Nonnormal Regression Models." *Biometrics* 34:318–327.

Lee, E. T. (1974). "A Computer Program for Linear Logistic Regression Analysis." *Computer Programs in Biomedicine* 4:80–92.

Littell, R. C., Freund, R. J., and Spector, P. C. (1991). *SAS System for Linear Models*. 3rd ed. Cary, NC: SAS Institute Inc.

Maddala, G. S. (1983). *Limited-Dependent and Qualitative Variables in Econometrics*. New York: Cambridge University Press.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

McFadden, D. (1974). "Conditional Logit Analysis of Qualitative Choice Behavior." In *Frontiers in Econometrics*, edited by P. Zarembka, 105–142. New York: Academic Press.

McNicol, D. (2005). *A Primer of Signal Detection Theory*. Mahwah, NJ: Lawrence Erlbaum Associates.

Mehta, C. R., and Patel, N. R. (1995). "Exact Logistic Regression: Theory and Examples." *Statistics in Medicine* 14:2143–2160.

Mehta, C. R., Patel, N. R., and Senchaudhuri, P. (1992). "Exact Stratified Linear Rank Tests for Ordered Categorical and Binary Data." *Journal of Computational and Graphical Statistics* 1:21–40.

Mehta, C. R., Patel, N. R., and Senchaudhuri, P. (2000). "Efficient Monte Carlo Methods for Conditional Logistic Regression." *Journal of the American Statistical Association* 95:99–108.

Menard, S. (2000). "Coefficients of Determination for Multiple Logistic Regression Analysis." *American Statistician* 54:17–24.

Metz, C. E. (1978). "Basic Principles of ROC Analysis." *Seminars in Nuclear Medicine* 8:283–298.

Moolgavkar, S. H., Lustbader, E. D., and Venzon, D. J. (1985). "Assessing the Adequacy of the Logistic Regression Model for Matched Case-Control Studies." *Statistics in Medicine* 4:425–435.

Mroz, T. A. (1987). "The Sensitivity of an Empirical Model of Married Women's Work to Economic and Statistical Assumptions." *Econometrica* 55:765–799.

Muller, K. E., and Fetterman, B. A. (2002). *Regression and ANOVA: An Integrated Approach Using SAS Software*. Cary, NC: SAS Institute Inc.

Murphy, A. H. (1973). "A New Vector Partition of the Probability Score." *Journal of Applied Meteorology* 12:595–600.

Naessens, J. M., Offord, K. P., Scott, W. F., and Daood, S. L. (1986). "The MCSTRAT Procedure." In *SUGI Supplemental Library User's Guide, Version 5 Edition*, 307–328. Cary, NC: SAS Institute Inc.

Nagelkerke, N. J. D. (1991). "A Note on a General Definition of the Coefficient of Determination." *Biometrika* 78:691–692.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Orme, C. (1988). "The Calculation of the Information Matrix Test for Binary Data Models." *The Manchester School* 56:370–376.

Osius, G., and Rojek, D. (1992). "Normal Goodness-of-Fit Tests for Multinomial Models with Large Degrees of Freedom." *Journal of the American Statistical Association* 87:1145–1152.

Paul, P., Pennell, M. L., and Lemeshow, S. (2013). "Standardizing the Power of the Hosmer-Lemeshow Goodness of Fit Test in Large Data Sets." *Statistics in Medicine* 32:67–80.

Peterson, B. L., and Harrell, F. E. (1990). "Partial Proportional Odds Models for Ordinal Response Variables." *Journal of the Royal Statistical Society, Series B* 39:205–217.

Pregibon, D. (1981). "Logistic Regression Diagnostics." *Annals of Statistics* 9:705–724.

Pregibon, D. (1984). "Data Analytic Methods for Matched Case-Control Studies." *Biometrics* 40:639–651.

Prentice, R. L., and Gloeckler, L. A. (1978). "Regression Analysis of Grouped Survival Data with Applications to Breast Cancer Data." *Biometrics* 34:57–67.

Press, S. J., and Wilson, S. (1978). "Choosing between Logistic Regression and Discriminant Analysis." *Journal of the American Statistical Association* 73:699–705.

Saito, T., and Rehmsmeier, M. (2015). "The Precision-Recall Plot Is More Informative Than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets." *PloS ONE* 10:e0118432.

Santner, T. J., and Duffy, D. E. (1986). "A Note on A. Albert and J. A. Anderson's Conditions for the Existence of Maximum Likelihood Estimates in Logistic Regression Models." *Biometrika* 73:755–758.

SAS Institute Inc. (1995). *Logistic Regression Examples Using the SAS System*. Cary, NC: SAS Institute Inc.

SAS Institute Inc. (2015). "Sample 54866: Logistic Model Selection Using Area under Curve (AUC) or R-Square Selection Criteria." SAS Institute Inc., Cary, NC. http://support.sas.com/kb/54/866.html.

Smith, T. J., and McKenna, C. M. (2013). "A Comparison of Logistic Regression Pseudo R2 Indices." *Multiple Linear Regression Viewpoints* 39:17–26.

Spiegelhalter, D. J. (1986). "Probabilistic Prediction in Patient Management and Clinical Trials." *Statistics in Medicine* 5:421–433.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2000). *Categorical Data Analysis Using the SAS System*. 2nd ed. Cary, NC: SAS Institute Inc.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2012). *Categorical Data Analysis Using SAS*. 3rd ed. Cary, NC: SAS Institute Inc.

Storer, B. E., and Crowley, J. (1985). "A Diagnostic for Cox Regression and General Conditional Likelihoods." *Journal of the American Statistical Association* 80:139–147.

Stukel, T. A. (1988). "Generalized Logistic Models." *Journal of the American Statistical Association* 83:426–431.

Tjur, T. (2009). "Coefficients of Determination in Logistic Regression Models—a New Proposal: The Coefficient of Discrimination." *American Statistician* 63:366–372.

Veall, M. R., and Zimmermann, K. F. (1996). "Pseudo-R2 Measures for Some Common Limited Dependent Variable Models." *Journal of Economic Surveys* 10:241–259.

Venzon, D. J., and Moolgavkar, S. H. (1988). "A Method for Computing Profile-Likelihood-Based Confidence Intervals." *Journal of the Royal Statistical Society, Series C* 37:87–94.

Vollset, S. E., Hirji, K. F., and Afifi, A. A. (1991). "Evaluation of Exact and Asymptotic Interval Estimators in Logistic Analysis of Matched Case-Control Studies." *Biometrics* 47:1311–1325.

Walker, S. H., and Duncan, D. B. (1967). "Estimation of the Probability of an Event as a Function of Several Independent Variables." *Biometrika* 54:167–179.

White, H. (1982). "Maximum Likelihood Estimation of Misspecified Models." *Econometrica* 50:1–25.

Williams, C. K. I. (2021). "The Effect of Class Imbalance on Precision-Recall Curves." *Neural Computation* 33:853–857.

Williams, D. A. (1982). "Extra-binomial Variation in Logistic Linear Models." *Journal of the Royal Statistical Society, Series C* 31:144–148.

Windmeijer, F. A. G. (1995). "Goodness-of-Fit Measures in Binary Choice Models." *Econometric Reviews* 14:101–116.

Wooldridge, J. M. (2002). *Econometric Analysis of Cross Section and Panel Data*. Cambridge, MA: MIT Press.

Youden, W. J. (1950). "Index for Rating Diagnostic Tests." *Cancer* 3:32–35.

Zamar, D., McNeney, B., , and Graham, J. (2007). "elrm: Software Implementing Exact-Like Inference for Logistic Regression Models." *Journal of Statistical Software* 21:1–18.

Zweig, M. H., and Campbell, G. (1993). "Receiver-Operating Characteristic (ROC) Plots: A Fundamental Evaluation Tool in Clinical Medicine." *Clinical Chemistry* 39:561–577.

Chapter 80
The MCMC Procedure

Contents

Overview: MCMC Procedure . . . **6213**
PROC MCMC Compared with Other SAS Procedures . . . 6214
Getting Started: MCMC Procedure . . . **6214**
Simple Linear Regression . . . 6215
The Behrens-Fisher Problem . . . 6221
Random-Effects Model . . . 6224
Syntax: MCMC Procedure . . . **6229**
PROC MCMC Statement . . . 6229
ARRAY Statement . . . 6248
BEGINCNST/ENDCNST Statement . . . 6248
BEGINNODATA/ENDNODATA Statements . . . 6250
BY Statement . . . 6250
CMPTMODEL Statement . . . 6251
MODEL Statement . . . 6254
PARMS Statement . . . 6263
PREDDIST Statement . . . 6264
PRIOR/HYPERPRIOR Statement . . . 6265
Programming Statements . . . 6266
RANDOM Statement . . . 6268
UDS Statement . . . 6276
Details: MCMC Procedure . . . **6277**
How PROC MCMC Works . . . 6277
Blocking of Parameters . . . 6280
Sampling Methods . . . 6282
Tuning the Proposal Distribution . . . 6283
Direct Sampling . . . 6286
Conjugate Sampling . . . 6286
Initial Values of the Markov Chains . . . 6287
Assignments of Parameters . . . 6288
Standard Distributions . . . 6289
Usage of Multivariate Distributions . . . 6303
Specifying a New Distribution . . . 6305
Using Density Functions in the Programming Statements . . . 6306
Truncation and Censoring . . . 6309
Some Useful SAS Functions . . . 6311
Matrix Functions in PROC MCMC . . . 6313

Create Design Matrix . . . 6317
Modeling Joint Likelihood . . . 6319
Access Lag and Lead Variables . . . 6320
Compartment Models . . . 6323
CALL ODE and CALL QUAD Subroutines . . . 6333
Regenerating Diagnostics Plots . . . 6339
Caterpillar Plot . . . 6341
Autocall Macros for Postprocessing . . . 6343
Gamma and Inverse Gamma Distributions . . . 6346
Posterior Predictive Distribution . . . 6348
Handling of Missing Data . . . 6352
Functions of Random-Effects Parameters . . . 6355
Spatial Prior . . . 6361
Floating Point Errors and Overflows . . . 6363
Handling Error Messages . . . 6366
Computational Resources . . . 6368
Displayed Output . . . 6369
ODS Table Names . . . 6373
ODS Graphics . . . 6375
Examples: MCMC Procedure . . . **6376**
Example 80.1: Simulating Samples From a Known Density . . . 6376
Example 80.2: Box-Cox Transformation . . . 6385
Example 80.3: Logistic Regression Model with a Diffuse Prior . . . 6394
Example 80.4: Logistic Regression Model with Jeffreys' Prior . . . 6400
Example 80.5: Poisson Regression . . . 6403
Example 80.6: Nonlinear Poisson Regression Models . . . 6406
Example 80.7: Logistic Regression Random-Effects Model . . . 6415
Example 80.8: Nonlinear Poisson Regression Multilevel Random-Effects Model . . . 6417
Example 80.9: Multivariate Normal Random-Effects Model . . . 6423
Example 80.10: Missing at Random Analysis . . . 6427
Example 80.11: Nonignorably Missing Data (MNAR) Analysis . . . 6430
Example 80.12: Change Point Models . . . 6434
Example 80.13: Exponential and Weibull Survival Analysis . . . 6438
Example 80.14: Time Independent Cox Model . . . 6451
Example 80.15: Time Dependent Cox Model . . . 6458
Example 80.16: Piecewise Exponential Frailty Model . . . 6463
Example 80.17: Normal Regression with Interval Censoring . . . 6470
Example 80.18: Constrained Analysis . . . 6472
Example 80.19: Implement a New Sampling Algorithm . . . 6478
Example 80.20: Using a Transformation to Improve Mixing . . . 6487
Example 80.21: Gelman-Rubin Diagnostics . . . 6496
Example 80.22: One-Compartment Model with Pharmacokinetic Data . . . 6503
References . . . **6507**

Overview: MCMC Procedure

The MCMC procedure is a general purpose Markov chain Monte Carlo (MCMC) simulation procedure that is designed to fit Bayesian models. Bayesian statistics is different from traditional statistical methods such as frequentist or classical methods. For a short introduction to Bayesian analysis and related basic concepts, see Chapter 8, "Introduction to Bayesian Analysis Procedures." Also see the section "A Bayesian Reading List" on page 178 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for a guide to Bayesian textbooks of varying degrees of difficulty.

In essence, Bayesian statistics treats parameters as unknown random variables, and it makes inferences based on the posterior distributions of the parameters. There are several advantages associated with this approach to statistical inference. Some of the advantages include its ability to use prior information and to directly answer specific scientific questions that can be easily understood. For further discussions of the relative advantages and disadvantages of Bayesian analysis, see the section "Bayesian Analysis: Advantages and Disadvantages" on page 154 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

It follows from Bayes' theorem that a posterior distribution is the product of the likelihood function and the prior distribution of the parameter. In all but the simplest cases, it is very difficult to obtain the posterior distribution directly and analytically. Often, Bayesian methods rely on simulations to generate sample from the desired posterior distribution and use the simulated draws to approximate the distribution and to make all of the inferences.

PROC MCMC is a flexible, simulation-based procedure that is suitable for fitting a wide range of Bayesian models. To use PROC MCMC, you need to specify a likelihood function for the data and a prior distribution for the parameters. If you are fitting hierarchical models, you can specify a hyperprior distribution or distributions for the random-effects parameters. PROC MCMC then obtains samples from the corresponding posterior distributions, produces summary and diagnostic statistics, and saves the posterior samples in an output data set that can be used for further analysis. Although PROC MCMC supports a suite of standard distributions, you can analyze data that have any likelihood, prior, or hyperprior, as long as these functions are programmable using the SAS DATA step functions. There are no constraints on how the parameters can enter the model, in either linear or any nonlinear functional form.

The MODEL statement in PROC MCMC can automatically model missing data, response variables, or covariates. The procedure treats the missing values as unknown parameters and incorporates the sampling of the missing values as part of the simulation.

PROC MCMC selects a sampling method for each parameter or a block of parameters. For example, when conjugacy is available, samples are drawn directly from the full conditional distribution by using standard random number generators. In other cases, PROC MCMC uses an adaptive blocked random walk Metropolis algorithm that uses a normal proposal distribution. You can also choose alternative sampling algorithms, such as the slice sampler.

PROC MCMC Compared with Other SAS Procedures

PROC MCMC is unlike most other SAS/STAT procedures in that the nature of the statistical inference is Bayesian. You specify prior distributions for the parameters with PRIOR statements, the likelihood function for the data with MODEL statements, and specify, if needed, the random effects with RANDOM statements. PROC MCMC derives inferences from simulation rather than through analytic or numerical methods. You should expect slightly different answers from each run for the same problem, unless the same random number seed is used. The model specification is similar to PROC NLIN, and PROC MCMC shares some of the syntax of PROC NLMIXED.

You can also perform a Bayesian analysis by using the BCHOICE, GENMOD, PHREG, LIFEREG, and FMM procedures for discrete choice models, generalized linear models, accelerated life failure models, Cox regression models, piecewise constant baseline hazard models (also known as piecewise exponential models), and finite mixture models. See Chapter 30, "The BCHOICE Procedure," Chapter 51, "The GENMOD Procedure," Chapter 92, "The PHREG Procedure," Chapter 76, "The LIFEREG Procedure," and Chapter 46, "The FMM Procedure."

You can fit Bayesian generalized linear mixed-effects models (GLMMs) by using the BGLIMM procedure. Although you can use both PROC BGLIMM and PROC MCMC to fit Bayesian GLMMs, PROC BGLIMM is designed specifically for this class of models and uses optimal sampling algorithms that PROC MCMC does not use, for the same model specifications. PROC BGLIMM enables you to fit Bayesian GLMMs that do not require customized specifications, such as nonstandard prior distributions. On the other hand, PROC MCMC supports a larger class of hierarchical models, such as nonlinear random-effects models, that PROC BGLIMM does not support.

Getting Started: MCMC Procedure

There are three examples in this "Getting Started" section: a simple linear regression, the Behrens-Fisher estimation problem, and a random-effects model. The regression model is chosen for its simplicity; the Behrens-Fisher problem illustrates some advantages of the Bayesian approach; and the random-effects model is one of the most prevalently used models.

Keep in mind that PARMS statements declare the parameters in the model, PRIOR statements declare the prior distributions, MODEL statements declare the likelihood for the data, and RANDOM statements declare the random effects. In most cases, you do not need to supply initial values. PROC MCMC advises you if it is unable to generate starting values for the Markov chain.

Simple Linear Regression

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcgs1.sas.)

This section illustrates some basic features of PROC MCMC by using a linear regression model. The model is as follows:

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

for the observations $i = 1, 2, \ldots, n$.

The data set Sashelp.class, which is available in the Sashelp library, identifies the children and their observed heights (the variable Height) and weights (the variable Weight).

The equation of interest is as follows:

$$\text{Weight}_i = \beta_0 + \beta_1 \text{Height}_i + \epsilon_i$$

The observation errors, ϵ_i, are assumed to be independent and identically distributed with a normal distribution with mean zero and variance σ^2.

$$\text{Weight}_i \sim \text{normal}(\beta_0 + \beta_1 \text{Height}_i, \sigma^2)$$

The likelihood function for each of the Weight, which is specified in the MODEL statement, is as follows:

$$p(\text{Weight}|\beta_0, \beta_1, \sigma^2, \text{Height}_i) = \phi(\beta_0 + \beta_1 \text{Height}_i, \sigma^2)$$

where $p(\cdot|\cdot)$ denotes a conditional probability density and ϕ is the normal density. There are three parameters in the likelihood: β_0, β_1, and σ^2. You use the PARMS statement to indicate that these are the parameters in the model.

Suppose you want to use the following three prior distributions on each of the parameters:

$$\begin{aligned}
\pi(\beta_0) &= \phi(0, \text{var} = 1e6) \\
\pi(\beta_1) &= \phi(0, \text{var} = 1e6) \\
\pi(\sigma^2) &= f_{i\Gamma}(\text{shape} = 3/10, \text{scale} = 10/3)
\end{aligned}$$

where $\pi(\cdot)$ indicates a prior distribution and $f_{i\Gamma}$ is the density function for the inverse gamma distribution. The normal priors on β_0 and β_1 have large variances, expressing your lack of knowledge about the regression coefficients. The priors correspond to an equal-tail 95% credible intervals of approximately (-2000, 2000) for β_0 and β_1. Priors of this type are often called *vague* or *diffuse* priors. See the section "Prior Distributions" on page 150 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more information. Typically diffuse prior distributions have little influence on the posterior distribution and are appropriate when stronger prior information about the parameters is not available.

A frequently used prior for the variance parameter σ^2 is the inverse gamma distribution. See Table 80.22 in the section "Standard Distributions" on page 6289 for the density definition. The inverse gamma distribution is a conjugate prior (see the section "Conjugate Sampling" on page 6286) for the variance parameter in a normal distribution. Also see the section "Gamma and Inverse Gamma Distributions" on page 6346 for typical usages of the gamma and inverse gamma prior distributions. With a shape parameter of 3/10 and a

scale parameter of 10/3, this prior corresponds to an equal-tail 95% credible interval of (1.7, 1E6), with the mode at 2.5641 for σ^2. Alternatively, you can use any other prior distribution with positive support on this variance component. For example, you can use the gamma prior.

According to Bayes' theorem, the likelihood function and prior distributions determine the posterior (joint) distribution of β_0, β_1, and σ^2 as follows:

$$\pi(\beta_0, \beta_1, \sigma^2|\text{Weight}, \text{Height}) \quad \propto \quad \pi(\beta_0)\pi(\beta_1)\pi(\sigma^2)\,p(\text{Weight}|\beta_0, \beta_1, \sigma^2, \text{Height})$$

You do not need to know the form of the posterior distribution when you use PROC MCMC. PROC MCMC automatically obtains samples from the desired posterior distribution, which is determined by the prior and likelihood you supply.

The following statements fit this linear regression model with diffuse prior information:

```
ods graphics on;
proc mcmc data=sashelp.class outpost=classout nmc=10000 thin=2 seed=246810;
   parms beta0 0 beta1 0;
   parms sigma2 1;
   prior beta0 beta1 ~ normal(mean = 0, var = 1e6);
   prior sigma2 ~ igamma(shape = 3/10, scale = 10/3);
   mu = beta0 + beta1*height;
   model weight ~ n(mu, var = sigma2);
run;
ods graphics off;
```

When ODS Graphics is enabled, diagnostic plots, such as the trace and autocorrelation function plots of the posterior samples, are displayed. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

The PROC MCMC statement invokes the procedure and specifies the input data set Class. The output data set Classout contains the posterior samples for all of the model parameters. The NMC= option specifies the number of posterior simulation iterations. The THIN= option controls the thinning of the Markov chain and specifies that one of every 2 samples is kept. Thinning is often used to reduce the correlations among posterior sample draws. In this example, 5,000 simulated values are saved in the Classout data set. The SEED= option specifies a seed for the random number generator, which guarantees the reproducibility of the random stream. For more information about Markov chain sample size, burn-in, and thinning, see the section "Burn-In, Thinning, and Markov Chain Samples" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

The PARMS statements identify the three parameters in the model: beta0, beta1, and sigma2. Each statement also forms a block of parameters, where the parameters are updated simultaneously in each iteration. In this example, beta0 and beta1 are sampled jointly, conditional on sigma2; and sigma2 is sampled conditional on fixed values of beta0 and beta1. In simple regression models such as this, you expect the parameters beta0 and beta1 to have high posterior correlations, and placing them both in the same block improves the mixing of the chain—that is, the efficiency that the posterior parameter space is explored by the Markov chain. For more information, see the section "Blocking of Parameters" on page 6280. The PARMS statements also assign initial values to the parameters (see the section "Initial Values of the Markov Chains" on page 6287). The regression parameters are given 0 as their initial values, and the scale parameter sigma2 starts at value 1. If you do not provide initial values, PROC MCMC chooses starting values for every parameter.

The PRIOR statements specify prior distributions for the parameters. The parameters beta0 and beta1 both share the same prior—a normal prior with mean 0 and variance 1e6. The parameter sigma2 has an inverse gamma distribution with a shape parameter of 3/10 and a scale parameter of 10/3. For a list of standard distributions that PROC MCMC supports, see the section "Standard Distributions" on page 6289.

The MU assignment statement calculates the expected value of Weight as a linear function of Height. The MODEL statement uses the shorthand notation, **n**, for the normal distribution to indicate that the response variable, Weight, is normally distributed with parameters mu and sigma2. The functional argument MEAN= in the normal distribution is optional, but you have to indicate whether sigma2 is a variance (VAR=), a standard deviation (SD=), or a precision (PRECISION=) parameter. See Table 80.2 in the section "MODEL Statement" on page 6254 for distribution specifications.

The distribution parameters can contain expressions. For example, you can write the MODEL statement as follows:

```
model weight ~ n(beta0 + beta1*height, var = sigma2);
```

Before you do any posterior inference, it is essential that you examine the convergence of the Markov chain (see the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures"). You cannot make valid inferences if the Markov chain has not converged. A very effective convergence diagnostic tool is the trace plot. Although PROC MCMC produces graphs at the end of the procedure output (see Figure 80.5), you should visually examine the convergence graph first.

The first table that PROC MCMC produces is the "Number of Observations" table, as shown in Figure 80.1. This table lists the number of observations read from the DATA= data set and the number of observations used in the analysis.

Figure 80.1 Observation Information

The MCMC Procedure

Number of Observations Read	19
Number of Observations Used	19

The "Parameters" table, shown in Figure 80.2, lists the names of the parameters, the blocking information, the sampling method used, the starting values, and the prior distributions. For more information about blocking information, see the section "Blocking of Parameters" on page 6280; for more information about starting values, see the section "Initial Values of the Markov Chains" on page 6287. The first block, which consists of the parameters beta0 and beta1, uses a random walk Metropolis algorithm. The second block, which consists of the parameter sigma2, is updated via its full conditional distribution in conjugacy. You should check this table to ensure that you have specified the parameters correctly, especially for complicated models.

Figure 80.2 Parameter Information

	Parameters			
Block	**Parameter**	**Sampling Method**	**Initial Value**	**Prior Distribution**
1	**beta0**	N-Metropolis	0	normal(mean = 0, var = 1e6)
	beta1		0	normal(mean = 0, var = 1e6)
2	**sigma2**	Conjugate	1.0000	igamma(shape = 3/10, scale = 10/3)

For each posterior distribution, PROC MCMC also reports summary and interval statistics (posterior means,

standard deviations, and 95% highest posterior density credible intervals), as shown in Figure 80.3. For more information about posterior statistics, see the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Figure 80.3 MCMC Summary and Interval Statistics

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta0	5000	-142.8	33.4337	-210.8	-81.6694
beta1	5000	3.8924	0.5333	2.9056	4.9545
sigma2	5000	137.3	51.1034	59.2363	236.3

By default, PROC MCMC computes the effective sample sizes (ESSs) as a convergence diagnostic test to help you determine whether the chain has converged. The ESSs are shown in Figure 80.4. For details and interpretations of ESS and additional convergence diagnostics, see the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Figure 80.4 MCMC Convergence Diagnostics

The MCMC Procedure

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
beta0	1102.2	4.5366	0.2204
beta1	1119.0	4.4684	0.2238
sigma2	2910.0	1.7182	0.5820

PROC MCMC produces a number of graphs, shown in Figure 80.5, which also aid convergence diagnostic checks. With the trace plots, there are two important aspects to examine. First, you want to check whether the mean of the Markov chain has stabilized and appears constant over the graph. Second, you want to check whether the chain has good mixing and is "dense," in the sense that it quickly traverses the support of the distribution to explore both the tails and the mode areas efficiently. The plots show that the chains appear to have reached their stationary distributions.

Next, you should examine the autocorrelation plots, which indicate the degree of autocorrelation for each of the posterior samples. High correlations usually imply slow mixing. Finally, the kernel density plots estimate the posterior marginal distributions for each parameter.

Figure 80.5 Diagnostic Plots for β_0, β_1 and σ^2

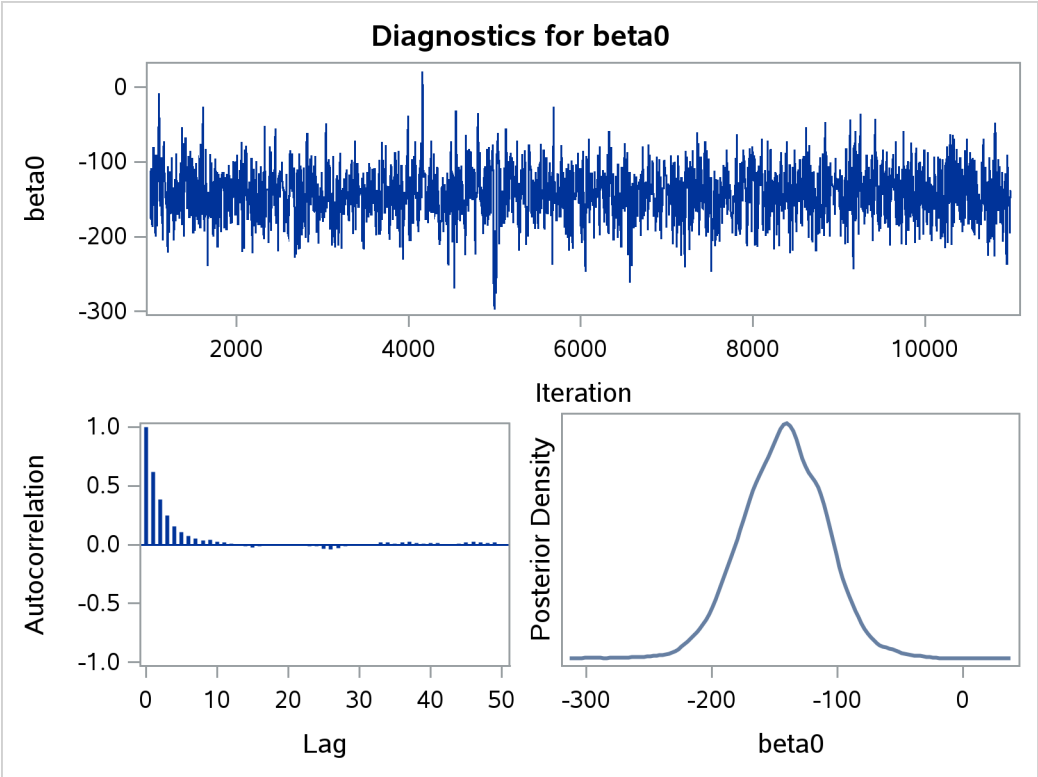

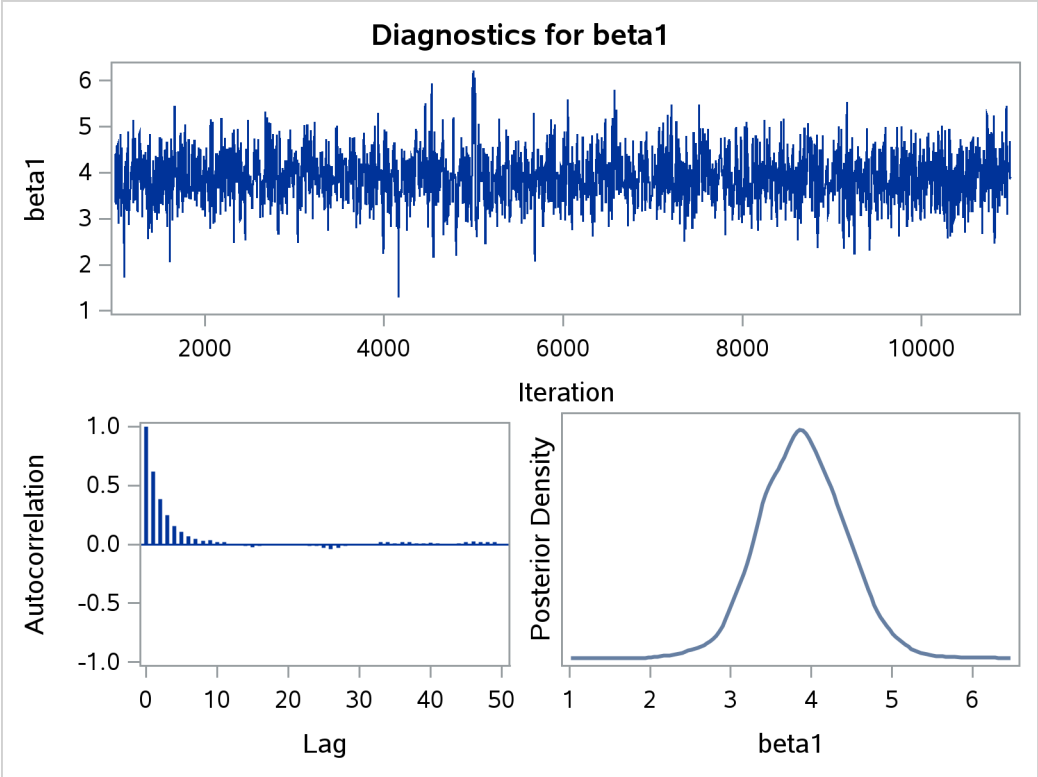

Figure 80.5 *continued*

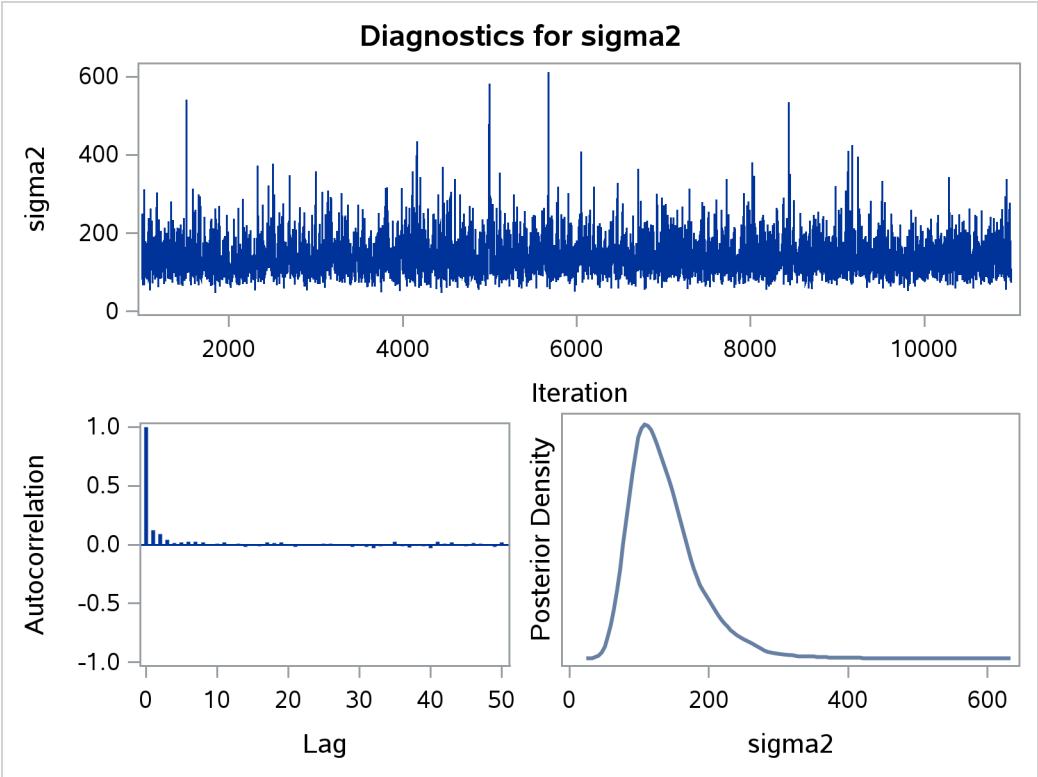

In regression models such as this, you expect the posterior estimates to be very similar to the maximum likelihood estimators with noninformative priors on the parameters, The REG procedure produces the following fitted model (code not shown):

$$\text{Weight} = -143.0 + 3.9 \times \text{Height}$$

These are very similar to the means show in Figure 80.3. With PROC MCMC, you can carry out informative analysis that uses specifications to indicate prior knowledge on the parameters. Informative analysis is likely to produce different posterior estimates, which are the result of information from both the likelihood and the prior distributions. Incorporating additional information in the analysis is one major difference between the classical and Bayesian approaches to statistical inference.

The Behrens-Fisher Problem

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcgs2.sas.)

One of the famous examples in the history of statistics is the Behrens-Fisher problem (Fisher 1935). Consider the situation where there are two independent samples from two different normal distributions:

$$y_{11}, y_{12}, \ldots, y_{1n_1} \sim \text{normal}(\mu_1, \sigma_1^2)$$
$$y_{21}, y_{22}, \ldots, y_{2n_2} \sim \text{normal}(\mu_2, \sigma_2^2)$$

Note that $n_1 \neq n_2$. When you do not want to assume that the variances are equal, testing the hypothesis $H_0 : \mu_1 = \mu 2$ is a difficult problem in the classical statistics framework, because the distribution under H_0 is not known. Within the Bayesian framework, this problem is straightforward because you can estimate the posterior distribution of $\mu_1 - \mu_2$ while taking into account the uncertainties in all of parameters by treating them as random variables.

Suppose you have the following set of data:

```
title 'The Behrens-Fisher Problem';

data behrens;
   input y ind @@;
   datalines;
121 1  94 1 119 1 122 1 142 1 168 1 116 1
172 1 155 1 107 1 180 1 119 1 157 1 101 1
145 1 148 1 120 1 147 1 125 1 126 2 125 2
130 2 130 2 122 2 118 2 118 2 111 2 123 2
126 2 127 2 111 2 112 2 121 2
;
```

The response variable is y, and the ind variable is the group indicator, which takes two values: 1 and 2. There are 19 observations that belong to group 1 and 14 that belong to group 2.

The likelihood functions for the two samples are as follows:

$$\begin{aligned} p(y_{1i}|\mu_1, \sigma_1^2) &= \phi(y_{1i}; \mu_1, \sigma_1^2) \quad \text{for} \quad i = 1, \ldots, 19 \\ p(y_{2j}|\mu_2, \sigma_2^2) &= \phi(y_{2j}; \mu_2, \sigma_2^2) \quad \text{for} \quad j = 1, \ldots, 14 \end{aligned}$$

Berger (1985) showed that a uniform prior on the support of the location parameter is a noninformative prior. The distribution is invariant under location transformations—that is, $\theta = \mu + c$. You can use this prior for the mean parameters in the model:

$$\begin{aligned} \pi(\mu_1) &\propto 1 \\ \pi(\mu_2) &\propto 1 \end{aligned}$$

In addition, Berger (1985) showed that a prior of the form $1/\sigma^2$ is noninformative for the scale parameter, and it is invariant under scale transformations (that is $\tau = c\sigma^2$). You can use this prior for the variance

parameters in the model:

$$\begin{aligned}\pi(\sigma_1^2) &\propto 1/\sigma_1^2\\ \pi(\sigma_2^2) &\propto 1/\sigma_2^2\end{aligned}$$

The log densities of the prior distributions on σ_1^2 and σ_2^2 are:

$$\begin{aligned}\log(\pi(\sigma_1^2)) &= -\log(\sigma_1^2)\\ \log(\pi(\sigma_2^2)) &= -\log(\sigma_2^2)\end{aligned}$$

The following statements generate posterior samples of $\mu_1, \mu_2, \sigma_1^2, \sigma_2^2$, and the difference in the means: $\mu_1 - \mu_2$:

```
proc mcmc data=behrens outpost=postout seed=123
          nmc=40000 monitor=(_parms_ mudif)
          statistics(alpha=0.01);
   ods select PostSumInt;
   parm mu1 0 mu2 0;
   parm sig21 1;
   parm sig22 1;
   prior mu: ~ general(0);
   prior sig21 ~ general(-log(sig21), lower=0);
   prior sig22 ~ general(-log(sig22), lower=0);
   mudif = mu1 - mu2;
   if ind = 1 then do;
      mu = mu1;
      s2 = sig21;
   end;
   else do;
      mu = mu2;
      s2 = sig22;
   end;
   model y ~ normal(mu, var=s2);
run;
```

The PROC MCMC statement specifies an input data set (Behrens), an output data set containing the posterior samples (Postout), a random number seed, and the simulation size. The MONITOR= option specifies a list of symbols, which can be either parameters or functions of the parameters in the model, for which inference is to be done. The symbol _parms_ is a shorthand for all model parameters—in this case, mu1, mu2, sig21, and sig22. The symbol mudif is defined in the program as the difference between μ_1 and μ_2.

The global suboption ALPHA=0.01 in the STATISTICS= option specifies 99% highest posterior density (HPD) credible intervals for all parameters.

The ODS SELECT statement displays the summary statistics and interval statistics tables while excluding all other output. For a complete list of ODS tables that PROC MCMC can produce, see the sections "Displayed Output" on page 6369 and "ODS Table Names" on page 6373.

The PARMS statements assign the parameters mu1 and mu2 to the same block, and sig21 and sig22 each to their own separate blocks. There are a total of three blocks. The PARMS statements also assign an initial value to each parameter.

The PRIOR statements specify prior distributions for the parameters. Because the priors are all nonstandard (uniform on the real axis for μ_1 and μ_2 and $1/\sigma^2$ for σ_1^2 and σ_2^2), you must use the GENERAL function here. The argument in the GENERAL function is an expression for the log of the distribution, up to an additive constant. This distribution can have any functional form, as long as it is programmable using SAS functions and expressions. The function specifies a distribution on the log scale, not on the original scale. The log of the prior on mu1 and mu2 is 0, and the log of the priors on sig21 and sig22 are –log(sig21) and –log(sig22) respectively. See the section "Specifying a New Distribution" on page 6305 for more information about how to specify an arbitrary distribution. The LOWER= option indicates that both variance terms must be strictly positive.

The MUDIF assignment statement calculates the difference between mu1 and mu2. The IF-ELSE statements enable different y's to have different mean and variance, depending on their group indicator ind. The MODEL statement specifies the normal likelihood function for each observation in the model.

Figure 80.6 displays the posterior summary and interval statistics.

Figure 80.6 Posterior Summary and Interval Statistics

The Behrens-Fisher Problem

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	99% HPD Interval	
mu1	40000	134.8	6.0092	119.1	152.3
mu2	40000	121.4	1.9119	116.1	126.6
sig21	40000	685.0	255.3	260.0	1580.5
sig22	40000	51.1811	23.8675	14.2322	136.0
mudif	40000	13.3730	6.3095	-3.3609	30.7938

The mean difference has a posterior mean value of 13.37, and the lower endpoints of the 99% credible intervals are negative. This suggests that the mean difference is positive with a high probability. However, if you want to estimate the probability that $\mu_1 - \mu_2 > 0$, you can do so as follows.

The following statements produce Figure 80.7:

```
proc format;
   value diffmt low-0 = 'mu1 - mu2 <= 0' 0<-high = 'mu1 - mu2 > 0';
run;

proc freq data = postout;
   tables mudif /nocum;
   format mudif diffmt.;
run;
```

The sample estimate of the posterior probability that $\mu_1 - \mu_2 > 0$ is 0.98. This example illustrates an advantage of Bayesian analysis. You are not limited to making inferences based on model parameters only. You can accurately quantify uncertainties with respect to any function of the parameters, and this allows for flexibility and easy interpretations in answering many scientific questions.

Figure 80.7 Estimated Probability of $\mu_1 - \mu_2 > 0$.

The Behrens-Fisher Problem

The FREQ Procedure

mudif	Frequency	Percent
mu1 - mu2 <= 0	753	1.88
mu1 - mu2 > 0	39247	98.12

Random-Effects Model

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcgs3.sas.)

This example illustrates how you can fit a normal likelihood random-effects model in PROC MCMC. PROC MCMC offers you the ability to model beyond the normal likelihood (see "Example 80.7: Logistic Regression Random-Effects Model" on page 6415, "Example 80.8: Nonlinear Poisson Regression Multilevel Random-Effects Model" on page 6417, and "Example 80.16: Piecewise Exponential Frailty Model" on page 6463).

Consider a scenario in which data are collected in groups and you want to model group-specific effects. You can use a random-effects model (sometimes also known as a variance-components model):

$$y_{ij} = \beta_0 + \beta_1 x_{ij} + \gamma_i + e_{ij}, \quad e_{ij} \sim \text{normal}(0, \sigma^2)$$

where $i = 1, 2, \ldots, I$ is the group index and $j = 1, 2, \ldots, n_i$ indexes the observations in the ith group. In the regression model, the fixed effects β_0 and β_1 are the intercept and the coefficient for variable x_{ij}, respectively. The random effect γ_i is the mean for the ith group, and e_{ij} are the error term.

Consider the following SAS data set:

```
title 'Random-Effects Model';

data heights;
   input Family G$ Height @@;
   datalines;
1 F 67   1 F 66   1 F 64   1 M 71   1 M 72   2 F 63
2 F 63   2 F 67   2 M 69   2 M 68   2 M 70   3 F 63
3 M 64   4 F 67   4 F 66   4 M 67   4 M 67   4 M 69
;
```

The response variable Height measures the heights (in inches) of 18 individuals. The covariate x is the gender (variable G), and the individuals are grouped according to Family (group index). Since the variable G is a character variable and PROC MCMC does not support a CLASS statement, you need to create the corresponding design matrix. In this example, the design matrix for a factor variable of level 2 (M and F) can be constructed using the following statement:

```
data input;
   set heights;
   if g eq 'F' then gf = 1;
   else gf = 0;
   drop g;
run;
```

The data set variable gf is a numeric variable and can be used in the regression model in PROC MCMC.

In data sets with factor variables that have more levels, you can consider using PROC TRANSREG to construct the design matrix. See the section "Create Design Matrix" on page 6317 for more information.

To model the data, you can assume that Height is normally distributed:

$$y_{ij} \sim \text{normal}(\mu_{ij}, \sigma^2), \quad \mu_{ij} = \beta_0 + \beta_1 \mathsf{gf}_{ij} + \gamma_i$$

The priors on the parameters β_0, β_1, γ_i are also assumed to be normal:

$$\begin{aligned} \beta_0, \beta_1 &\sim \text{normal}(0, \text{var} = 1e5) \\ \gamma_i &\sim \text{normal}(0, \text{var} = \sigma_\gamma^2) \end{aligned}$$

Priors on the variance terms, σ^2 and σ_γ^2, are inverse gamma:

$$\sigma^2, \sigma_\gamma^2 \sim \text{igamma}(\text{shape} = 0.01, \text{scale} = 0.01)$$

The inverse gamma distribution is a conjugate prior for the variance in the normal likelihood and the variance in the prior distribution of the random effect.

The following statements fit a linear random-effects model to the data and produce the output shown in Figure 80.9 and Figure 80.10:

```
ods graphics on;
proc mcmc data=input outpost=postout nmc=50000 seed=7893 plots=trace;
   ods select Parameters REparameters PostSumInt tracepanel;
   parms b0 0 b1 0 s2 1 s2g 1;

   prior b: ~ normal(0, var = 10000);
   prior s: ~ igamma(0.01, scale = 0.01);
   random gamma ~ normal(0, var = s2g) subject=family monitor=(gamma);
   mu = b0 + b1 * gf + gamma;
   model height ~ normal(mu, var = s2);
run;
ods graphics off;
```

Some of the statements are very similar to those shown in the previous two examples. The ODS GRAPHICS ON statement enables ODS Graphics. The PROC MCMC statement specifies the input and output data sets, the simulation size, and a random number seed. The ODS SELECT statement displays the model parameter and random-effects parameter information tables, summary statistics table, the interval statistics table, and the trace plots.

The PARMS statement lumps all four model parameters in a single block. They are b0 (overall intercept), b1 (main effect for gf), s2 (variance of the likelihood function), and s2g (variance of the random effect). If a random walk Metropolis sampler is the only applicable sampler for all parameters, then these four parameters

are updated in a single block. However, because PROC MCMC updates the parameters s2 and s2g via conjugacy, these parameters are separated into individual blocks. (See the Block column in "Parameters" table in Figure 80.8.)

The PRIOR statements specify priors for all the parameters. The notation b: is a shorthand for all symbols that start with the letter 'b'. In this example, b: includes b0 and b1. Similarly, s: stands for both s2 and s2g. This shorthand notation can save you some typing, and it keeps your statements tidy.

The RANDOM statement specifies a single random effect to be gamma, and specifies that it has a normal prior centered at 0 with variance s2g. The SUBJECT= argument in the RANDOM statement defines a group index (family) in the model, where all observations from the same family should have the same group indicator value. The MONITOR= option outputs analysis for all the random-effects parameters.

Finally, the MU assignment statement calculates the expected value of the height of the model. The calculation includes the random-effects term gamma. The MODEL statement specifies the likelihood function for height.

The "Parameters" and "Random-Effects Parameters" tables, shown in Figure 80.8, contain information about the model parameters and the four random-effects parameters.

Figure 80.8 Model and Random-Effects Parameter Information

Random-Effects Model

The MCMC Procedure

Parameters

Block	Parameter	Sampling Method	Initial Value	Prior Distribution
1	s2	Conjugate	1.0000	igamma(0.01, scale = 0.01)
2	s2g	Conjugate	1.0000	igamma(0.01, scale = 0.01)
3	b0	N-Metropolis	0	normal(0, var = 10000)
	b1		0	normal(0, var = 10000)

Random Effect Parameters

Parameter	Sampling Method	Subject	Number of Subjects	Subject Values	Prior Distribution
gamma	N-Metropolis	Family	4	1 2 3 4	normal(0, var = s2g)

The posterior summary and interval statistics for the model parameters and the random-effects parameters are shown in Figure 80.9.

Figure 80.9 Posterior Summary and Interval Statistics

Random-Effects Model

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
b0	50000	68.3094	1.4598	65.5446	70.8492
b1	50000	-3.5358	0.9524	-5.5319	-1.7448
s2	50000	4.1449	1.9730	1.3930	7.9364
s2g	50000	5.9587	28.4772	0.00155	22.0497
gamma_1	50000	1.0855	1.5801	-1.0753	4.6908
gamma_2	50000	0.1550	1.3862	-2.5341	3.0563
gamma_3	50000	-1.2087	1.6501	-5.1118	1.0319
gamma_4	50000	0.2603	1.4094	-2.2312	3.2988

Trace plots for all the parameters are shown in Figure 80.10. The mixing looks very reasonable, suggesting convergence.

Figure 80.10 Plots for b_1 and Log of the Posterior Density

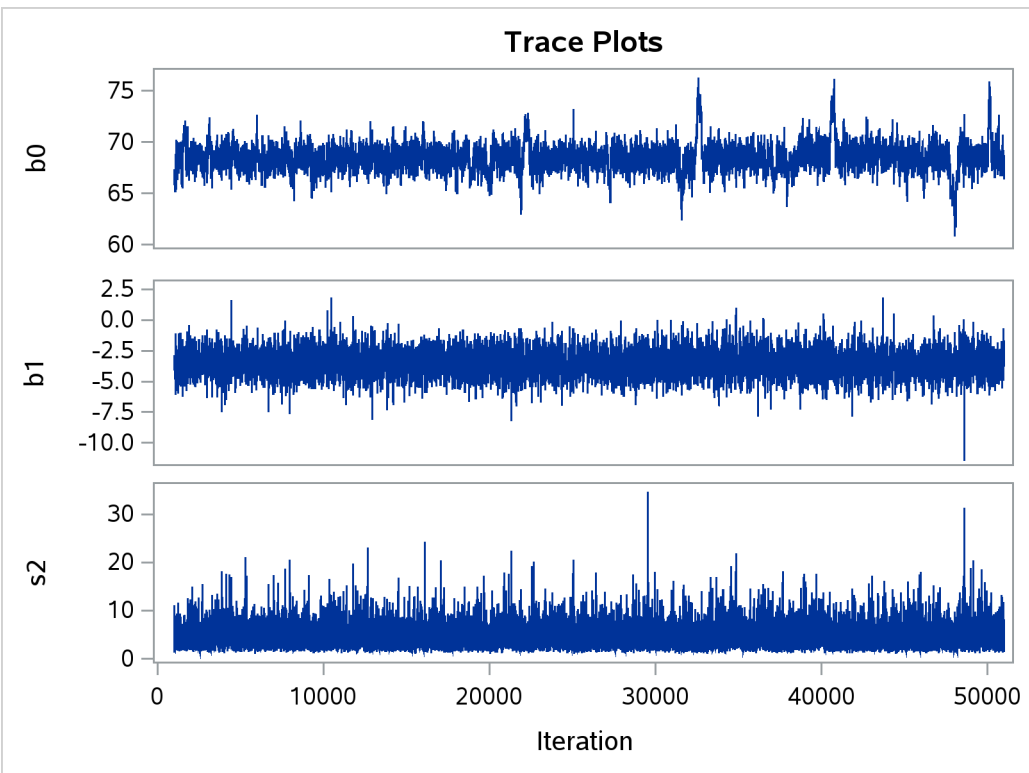

Figure 80.10 *continued*

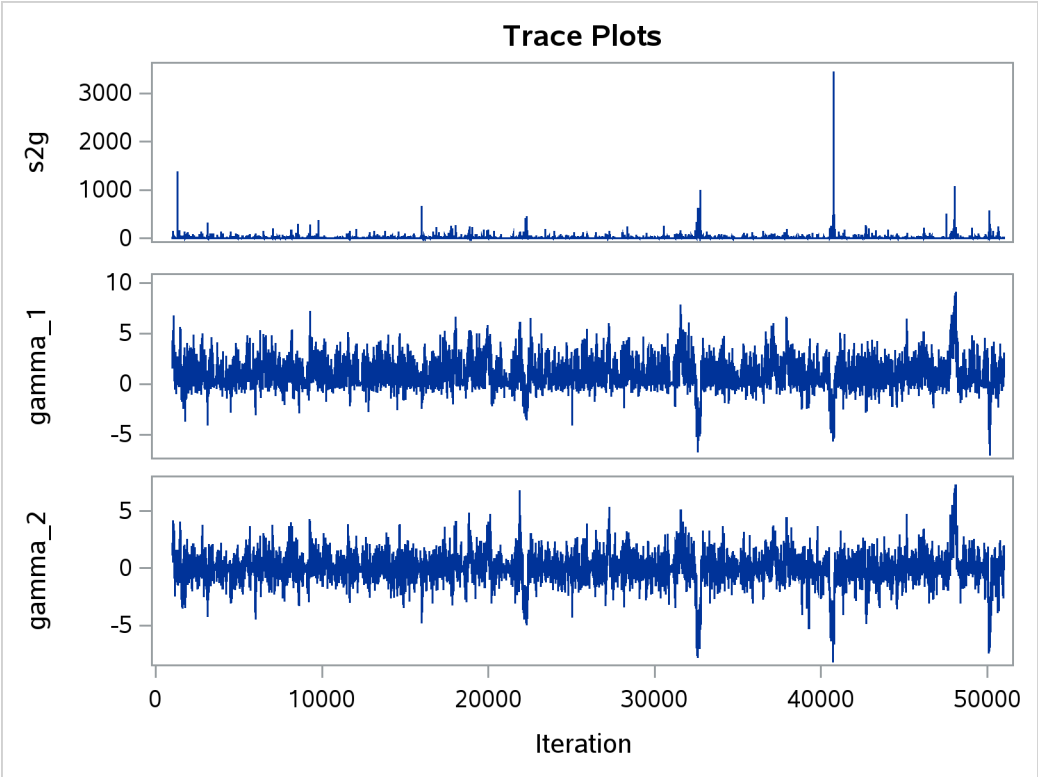

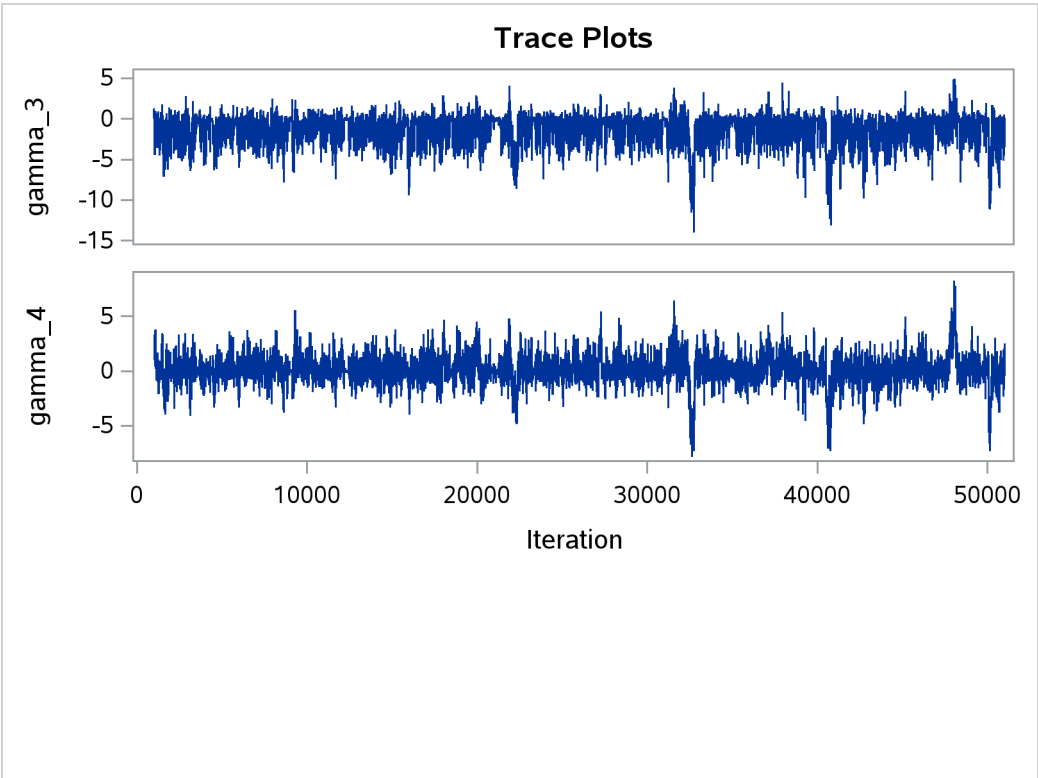

From the summary statistics table, you see that the HPD interval for β_0 is positive, and the interval for β_1

is negative, respectively. This indicates the positive effect of the intercept and the negative effect of gf on predicting height in this model.

Syntax: MCMC Procedure

The following statements are available in the MCMC procedure. Items within < > are optional.

PROC MCMC < *options* > **;**
  ARRAY *arrayname* **[** *dimensions* **]** < **$** > < *variables-and-constants* > **;**
  BEGINCNST/ENDCNST ;
  BEGINNODATA/ENDNODATA ;
  BY *variables* **;**
  CMPTMODEL *required-options conditionally-required-options* < *options* > **;**
  MODEL *variable* ∼ *distribution* < *options* > **;**
  PARMS *parameter* < **=** > *number* < / *options* > **;**
  PREDDIST < '*label*' > **OUTPRED=***SAS-data-set* < *options* > **;**
  PRIOR/HYPERPRIOR *parameter* ∼ *distribution* **;**
  Programming statements ;
  RANDOM *random-effects-specification* < / *options* > **;**
  UDS *subroutine-name* **(***subroutine-argument-list***) ;**

The PARMS statements declare parameters in the model and assign optional starting values for the Markov chain. The PRIOR/HYPERPRIOR statements specify the prior distributions of the parameters. The MODEL statements specify the log-likelihood functions for the response variables. These statements form the basis of most Bayesian models.

In addition, you can use the ARRAY statement to define constant or parameter arrays, the BEGINCNST/ENDCNST and BEGINNODATA/ENDNODATA statements to omit unnecessary evaluation and reduce simulation time, the PREDDIST statement to generate samples from the posterior predictive distribution, the program statements to specify more complicated models that you want to fit, the RANDOM statement to specify random effects and their prior distributions, and the UDS statement to define your own Gibbs samplers to sample parameters in the model.

The following sections provide a description of each of these statements.

PROC MCMC Statement

PROC MCMC *options* **;**

The PROC MCMC statement invokes the MCMC procedure. Table 80.1 summarizes the *options* available in the PROC MCMC statement.

Table 80.1 PROC MCMC Statement Options

Option	Description
Basic Options	
DATA=	Names the input data set
OUTPOST=	Names the output data set for posterior samples of parameters
Debugging Output	
LIST	Displays the model program and variables
LISTCODE	Displays the compiled model program
TRACE	Displays detailed model execution messages
Frequently Used MCMC Options	
ALG=	Specifies the default sampling algorithm
MAXTUNE=	Specifies the maximum number of tuning loops
MINTUNE=	Specifies the minimum number of tuning loops
NBI=	Specifies the number of burn-in iterations
NMC=	Specifies the number of MCMC iterations, excluding the burn-in iterations
NTHREADS=	Specifies the number of threads to use
NTU=	Specifies the number of tuning iterations
PROPCOV=	Controls options for constructing the initial proposal covariance matrix
SEED=	Specifies the random seed for simulation
THIN=	Specifies the thinning rate
Less Frequently Used MCMC Options	
ACCEPTTOL=	Specifies a tolerance for acceptance probabilities
BINARYJOINT	Jointly samples a block of binary parameters
DISCRETE=	Controls the sampling of discrete parameters
INIT=	Controls the generation of initial values
MCHISTORY=	Displays the Markov chain sampling history
MAXINDEXPRINT=	Specifies the maximum number of observation indices to print in models with missing data
MAXSUBVALUEPRINT=	Specifies the maximum number of subject values to print in the “Random Effects Parameters” table
REOBSINFO	Displays more detailed information about each random effect
SCALE=	Specifies the initial scale applied to the proposal distribution
TARGACCEPT=	Specifies the target acceptance rate for the random-walk sampler
TARGACCEPTI=	Specifies the target acceptance rate for the independence sampler
TUNEWT=	Specifies the weight used in covariance updating
Summary, Diagnostics, and Plotting Options	
AUTOCORLAG=	Specifies the number of autocorrelation lags used to compute effective sample sizes and Monte Carlo errors
DIAGNOSTICS=	Controls the convergence diagnostics
DIC	Computes the deviance information criterion (DIC)
MONITOR=	Outputs the analysis for a list of symbols of interest
PLOTS=	Controls plotting
STATISTICS=	Controls posterior statistics

Table 80.1 *continued*

Option	Description
Other Options	
INF=	Specifies the machine numerical limit for infinity
JOINTMODEL	Specifies the joint log-likelihood function
MISSING=	Indicates how missing values are handled
NOLOGDIST	Omits the calculation of the logarithm of the joint distribution of the parameters
SIMREPORT=	Controls the frequency of the report for the expected run time
SINGDEN=	Specifies the singularity tolerance

These *options* are described in alphabetical order.

ACCEPTTOL=*n*

specifies a tolerance for acceptance probabilities. By default, ACCEPTTOL=0.075.

ALG=*value*

PROPDIST=*value*

specifies the default sampling algorithm for continuous parameters when more optimal algorithms, such as conjugate samplers, are not available. For more information, see the sections "Hamiltonian Monte Carlo Sampler" on page 161 and "Metropolis and Metropolis-Hastings Algorithms" on page 156 in Chapter 8, "Introduction to Bayesian Analysis Procedures." By default, ALG=NORMAL, a normal kernel based random-walk Metropolis.

You can specify the following *values*:

HMC< (*hmc-options***) >**

specifies the Hamiltonian Monte Carlo algorithm with a fixed step size and predetermined number of steps. You can specify the following *hmc-options*:

NSTEPS=*value*

N=*value*

specifies the number of steps in the HMC algorithm. By default, N=15.

SAVEGRAD

saves the gradient calculation in the OUTPOST= data set.

STEPSIZE=*value*

specifies the step size in the HMC algorithm. By default, STEPSIZE=0.1.

NORMAL

N

specifies a normal distribution as the proposal distribution in the random-walk Metropolis algorithm. This is the default.

NUTS< (*nuts-options*) >
specifies the No-U-Turn Sampler of the Hamiltonian algorithm. You can specify the following *nuts-options*:

DELTA=*value*
specifies the target acceptance rate during the tuning process. By default, DELTA=0.6. Increasing the *value* can often improve mixing, but it can also significantly slow down the sampling.

FCALLS
outputs the number of function evaluations at each iteration.

MAXHEIGHT=*value*
specifies the maximum height of the NUTS tree. The taller the tree, the more gradient evaluations per iteration the procedure calculates. The number of evaluations is 2^{height}. By default, MAXHEIGHT=10. Usually, the height of a tree should be no more than 7 or 8 during the sampling stage, but it can go higher during the tuning stage. A larger number indicates that the algorithm is having difficulty converging. PROC MCMC stops when the height of a NUTS tree surpasses MAXHEIGHT by the number of times specified in the MAXTIME= option. You can increase the height of the tree and the MAXTIME value.

MAXTIME=*value*
specifies the maximum number of iterations that it takes the algorithm to surpass the MAXHEIGHT of the NUTS tree before the procedure stops. By default, MAXTIME=1.

NTU=*value*
specifies the number of tuning iterations used by NUTS. By default, NTU=1000.

SAVEGRAD
saves the gradient calculation in the OUTPOST= data set.

T< (*df*) >
specifies a *t* distribution with the degrees of freedom *df* in the random-walk Metropolis algorithm. By default, *df* = 3. If *df* > 100, the normal distribution is used, because the two distributions are almost identical.

AUTOCORLAG=*n*
ACLAG=*n*
specifies the maximum number of autocorrelation lags used in computing the effective sample size; see the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The value is used in the calculation of the Monte Carlo standard error; see the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures." By default, AUTOCORLAG=MIN(500, MCsample/4), where MCsample is the Markov chain sample size kept after thinning—that is, MCsample $= \left[\frac{\text{NMC}}{\text{NTHIN}}\right]$. If the value of the AUTOCORLAG= option is set too low, you might observe significant lags, and the effective sample size cannot be calculated accurately. A warning message appears, and you can increase either AUTOCORLAG= or NMC=, accordingly.

BINARYJOINT

jointly samples binary parameters in a block.

DISCRETE=*keyword*

specifies the proposal distribution used in sampling discrete parameters. By default, DISCRETE=BINNING.

You can specify the following *keywords*:

BINNING

uses continuous proposal distributions for all discrete parameter blocks. The proposed sample is then discretized (binned) before further calculations. This sampling method approximates the correlation structure among the discrete parameters in the block and could improve mixing in some cases.

GEO

uses independent symmetric geometric proposal distributions for all discrete parameter blocks. This proposal does not take parameter correlations into account. However, it can work better than the BINNING option in cases where the range of the parameters is relatively small and a normal approximation can perform poorly.

DIAGNOSTICS=NONE | (*keyword-list***)**

DIAG=NONE | (*keyword-list***)**

specifies options for MCMC convergence diagnostics. By default, PROC MCMC computes the Geweke test, sample autocorrelations, effective sample sizes, and Monte Carlo errors. The Raftery-Lewis and Heidelberger-Welch tests are also available. See the section "Assessing Markov Chain Convergence" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details on convergence diagnostics. You can request all of the diagnostic tests by specifying DIAGNOSTICS=ALL. You can suppress all the tests by specifying DIAGNOSTICS=NONE.

You can use postprocessing autocall macros to calculate convergence diagnostics of the posterior samples after PROC MCMC has exited. See the section "Autocall Macros for Postprocessing" on page 6343.

The following *options* are available.

ALL

computes all diagnostic tests and statistics. You can combine the option ALL with any other specific tests to modify test options. For example DIAGNOSTICS=(ALL AUTOCORR(LAGS=(1 5 35))) computes all tests with default settings and autocorrelations at lags 1, 5, and 35.

AUTOCORR < (*autocorr-options***) >**

computes default autocorrelations at lags 1, 5, 10, and 50 for each variable. You can choose other lags by using the following *autocorr-options*:

LAGS | AC=*numeric-list*

specifies autocorrelation lags. The *numeric-list* must take positive integer values.

ESS

computes the effective sample sizes (Kass et al. (1998)) of the posterior samples of each parameter. It also computes the correlation time and the efficiency of the chain for each parameter. Small values of ESS might indicate a lack of convergence. See the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

GEWEKE < (*Geweke-options*) >

computes the Geweke spectral density diagnostics; this is a two-sample t-test between the first f_1 portion and the last f_2 portion of the chain. See the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The default is FRAC1=0.1 and FRAC2=0.5, but you can choose other fractions by using the following *Geweke-options*:

FRAC1 | F1=*value*

specifies the beginning FRAC1 proportion of the Markov chain. By default, FRAC1=0.1.

FRAC2 | F2=*value*

specifies the end FRAC2 proportion of the Markov chain. By default, FRAC2=0.5.

HEIDELBERGER | HEIDEL < (*Heidel-options*) >

computes the Heidelberger and Welch diagnostic (which consists of a stationarity test and a halfwidth test) for each variable. The stationary diagnostic test tests the null hypothesis that the posterior samples are generated from a stationary process. If the stationarity test is passed, a halfwidth test is then carried out. See the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

These diagnostics are not performed by default. You can specify the DIAGNOSTICS=HEIDELBERGER option to request these diagnostics, and you can also specify suboptions, such as DIAGNOSTICS=HEIDELBERGER(EPS=0.05), as follows:

SALPHA=*value*

specifies the α level ($0 < \alpha < 1$) for the stationarity test. By default, SALPHA=0.05.

HALPHA=*value*

specifies the α level ($0 < \alpha < 1$) for the halfwidth test. By default, HALPHA=0.05.

EPS=*value*

specifies a small positive number ϵ such that if the halfwidth is less than ϵ times the sample mean of the retaining iterates, the halfwidth test is passed. By default, EPS=0.1.

MCSE

MCERROR

computes the Monte Carlo standard error for the posterior samples of each parameter.

NONE

suppresses all of the diagnostic tests and statistics. This is not recommended.

RAFTERY | RL < (*Raftery-options*) >

computes the Raftery and Lewis diagnostics, which evaluate the accuracy of the estimated quantile ($\hat{\theta}_Q$ for a given $Q \in (0, 1)$) of a chain. $\hat{\theta}_Q$ can achieve any degree of accuracy when the chain is allowed to run for a long time. The algorithm stops when the estimated probability $\hat{P}_Q = \Pr(\theta \leq \hat{\theta}_Q)$ reaches within $\pm R$ of the value Q with probability S; that is, $\Pr(Q - R \leq \hat{P}_Q \leq Q + R) = S$. See the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The *Raftery-options* enable you to specify Q, R, S, and a precision level ϵ for a stationary test.

These diagnostics are not performed by default. You can specify the DIAGNOSTICS=RAFERTY option to request these diagnostics, and you can also specify suboptions, such as DIAGNOSTICS=RAFERTY(QUANTILE=0.05), as follows:

QUANTILE | Q=*value*

specifies the order (a value between 0 and 1) of the quantile of interest. By default, QUANTILE=0.025.

ACCURACY | R=*value*

specifies a small positive number as the margin of error for measuring the accuracy of estimation of the quantile. By default, ACCURACY=0.005.

PROB | S=*value*

specifies the probability of attaining the accuracy of the estimation of the quantile. By default, PROB=0.95.

EPS=*value*

specifies the tolerance level (a small positive number) for the stationary test. By default, EPS=0.001.

DIC

computes the Deviance Information Criterion (DIC). DIC is calculated using the posterior mean estimates of the parameters. See the section "Deviance Information Criterion (DIC)" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

DATA=*SAS-data-set*

specifies the input data set. Observations in this data set are used to compute the log-likelihood function that you specify with PROC MCMC statements.

INF=*value*

specifies the numerical definition of infinity in PROC MCMC. The default is INF=1E15. For example, PROC MCMC considers 1E16 to be outside of the support of the normal distribution and assigns a missing value to the log density evaluation. You can select a larger value with the INF= option. The minimum value allowed is 1E10.

INIT=(*keyword-list*)

specifies options for generating the initial values for the parameters. These options apply only to prior distributions that are recognized by PROC MCMC. See the section "Standard Distributions" on page 6289 for a list of these distributions. If either of the functions GENERAL or DGENERAL is used, you must supply explicit initial values for the parameters. By default, INIT=MODE. The following *keywords* are used:

MODE
uses the mode of the prior density as the initial value of the parameter, if you did not provide one. If the mode does not exist or if it is on the boundary of the support of the density, the mean value is used. If the mean is outside of the support or on the boundary, which can happen if the prior distribution is truncated, a random number drawn from the prior is used as the initial value.

PINIT
tabulates parameter values after the tuning phase. This option also tabulates the tuned proposal parameters used by the Metropolis algorithm. These proposal parameters include covariance matrices for continuous parameters and probability vectors for discrete parameters for each block. By default, PROC MCMC does not display the initial values or the tuned proposal parameters after the tuning phase.

RANDOM
generates a random number from the prior density and uses it as the initial value of the parameter, if you did not provide one.

REINIT
resets the parameters, after the tuning phase, with the initial values that you provided explicitly or that were assigned by PROC MCMC. By default, PROC MCMC does not reset the parameters because the tuning phase usually moves the Markov chains to a more favorable place in the posterior distribution.

LIST
displays the model program and variable lists. The LIST option is a debugging feature and is not normally needed.

LISTCODE
displays the compiled program code. The LISTCODE option is a debugging feature and is not normally needed.

JOINTMODEL
JOINTLLIKE
specifies how the likelihood function is calculated. By default, PROC MCMC assumes that the observations in the data set are independent so that the joint log-likelihood function is the sum of the individual log-likelihood functions for the observations, where the individual log-likelihood function is specified in the MODEL statement. When your data are not independent, you can specify the JOINTMODEL option to modify the way that PROC MCMC computes the joint log-likelihood function. In this situation, PROC MCMC no longer steps through the input data set to sum the individual log likelihood.

To use this option correctly, you need to do the following two things:

- create ARRAY symbols to store all data set variables that are used in the program. This can be accomplished with the BEGINCNST and ENDCNST statements.
- program the joint log-likelihood function by using these ARRAY symbols only. The MODEL statement specifies the joint log-likelihood function for the entire data set. Typically, you use the function GENERAL in the MODEL statement.

See the sections "BEGINCNST/ENDCNST Statement" on page 6248 and "Modeling Joint Likelihood" on page 6319 for details.

MAXTUNE=*n*

specifies an upper limit for the number of proposal tuning loops. By default, MAXTUNE=24. See the section "Covariance Tuning" on page 6284 for more details.

MAXINDEXPRINT=*number* | **ALL**

MAXIPRINT=*number* | **ALL**

specifies the maximum number of observation indices to print in the ODS tables "Missing Response Information" table and "Missing Covariates Information" table. This option applies only to programs that model missing data. The default value is 20. MAXINDEXPRINT=ALL prints all observation indices for every missing variable that is modeled in PROC MCMC.

MAXSUBVALUEPRINT=*number* | **ALL**

MAXSVPRINT=*number* | **ALL**

specifies the maximum number of subject values to display in the "Subject Values" column of the ODS table "Random Effects Parameters." This option applies only to programs that have RANDOM statements. The default value is 20. MAXSUBVALUEPRINT=ALL prints all subject values for every random effect in the program.

MCHISTORY=*keyword*

MCHIST=*keyword*

controls the display of the Markov chain sampling history.

BRIEF

produces a summary output for the tuning, burn-in, and sampling history tables. The tables show the following when applicable:

- "RWM Scale" shows the scale, or the range of the scales, used in each random walk Metropolis block that is normal or is based on a *t* distribution.
- "Probability" shows the proposal probability parameter, or the range of the parameters, used in each random walk Metropolis block that is based on a geometric distribution.
- "RWM Acceptance Rate" shows the acceptance rate, or the range of the acceptance rates, for each random walk Metropolis block.
- "IM Acceptance Rate" shows the acceptance rate, or the range of the acceptance rates, for each independent Metropolis block.

DETAILED

produces detailed output of the tuning, burn-in, and sampling history tables, including scale values, acceptance probabilities, blocking information, and so on. Use this option with caution, especially in random-effects models that have a large number of random-effects groups. This option can produce copious output.

NONE

produces none of the tuning history, burn-in history, and sampling history tables.

The default is MCHISTORY=NONE.

MINTUNE=*n*

specifies a lower limit for the number of proposal tuning loops. By default, MINTUNE=2. See the section "Covariance Tuning" on page 6284 for more details.

MISSING=*keyword*

MISS=*keyword*

specifies how to handle missing values. For more information, see the section "Handling of Missing Data" on page 6352. By default, MISSING=CCMODELY. PROC MCMC models missing response variables and discard observations with missing covariates.

AC | ALLCASE

enables you to directly model the missing values in an all-cases analysis and does not attempt to model the missing response values. You can use any techniques that you want to deal with the missing values (for example, you can use fully Bayesian or multiple imputation).

ACMODELY

models the variable in missing responses without discarding observations that have missing covariates. This option is useful in modeling censored data.

CC | COMPLETECASE

assumes a complete case analysis, so all observations that have missing variable values are discarded prior to the simulation.

CCMODELY

models the missing response variables and discards observations that have missing covariates.

By default, MISSING=CCMODELY.

MONITOR= (*symbol-list***)**

outputs analysis for selected symbols of interest in the program. The symbols can be any of the following: model parameters (symbols in the PARMS statement), secondary parameters (assigned using the operator "="), the log of the posterior density (LOGPOST), the log of the prior density (LOGPRIOR), the log of the hyperprior density (LOGHYPER) if the HYPER statement is used, or the log of the likelihood function (LOGLIKE). You can use the keyword _PARMS_ as a shorthand for all of the model parameters. PROC MCMC performs only posterior analyses (such as plotting, diagnostics, and summaries) on the symbols selected with the MONITOR= option. You can also choose to monitor an entire array by specifying the name of the array. By default MONITOR=_PARMS_.

Posterior samples of any secondary parameters listed in the MONITOR= option are saved in the OUTPOST= data set. Posterior samples of model parameters are always saved to the OUTPOST= data set, regardless of whether they appear in the MONITOR= option.

NBI=*n*

specifies the number of burn-in iterations to perform before beginning to save parameter estimate chains. By default, NBI=1000. See the section "Burn-In, Thinning, and Markov Chain Samples" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details.

NMC=*n*

specifies the number of iterations in the main simulation loop. This is the MCMC sample size if THIN=1. By default, NMC=1000.

NOLOGDIST

omits the calculation of the logarithm of the joint distribution of the model parameters at each iteration. The option applies only if all parameters in the model are updated directly from their target distribution, either from the full conditional posterior via conjugacy or from the marginal distribution. Such algorithms do not require the calculation of the joint posterior distribution; hence PROC MCMC runs faster by avoiding these unnecessary calculations. As a result, the OUTPOST= data set does not contain the LOGPRIOR, LOGLIKE, and LOGPOST variables.

NTHREADS=*n*

specifies the number of threads for simulation. PROC MCMC performs two types of threading. In sampling model parameters, PROC MCMC allocates data to different threads and calculates the objective function by accumulating values from each thread; in sampling of random-effects parameters and missing data variables, each thread generates a subset of these parameters simultaneously at each iteration. Most sampling algorithms are threaded. NTHREADS=–1 sets the number of available threads to the number of hyperthreaded cores available on the system. By default, NTHREADS=1.

NTU=*n*

specifies the number of iterations to use in each proposal tuning phase. By default, NTU=500.

OUTPOST=*SAS-data-set*

specifies an output data set that contains the posterior samples of all model parameters, the iteration numbers (variable name ITERATION), the log of the posterior density (LOGPOST), the log of the prior density (LOGPRIOR), the log of the hyperprior density (LOGHYPER), if the HYPER statement is used, and the log likelihood (LOGLIKE). Any secondary parameters (assigned using the operator "=") listed in the MONITOR= option are saved to this data set. By default, no OUTPOST= data set is created.

PLOTS<(*global-plot-options***)>=(***plot-request* **<...** *plot-request***>)**

PLOT<(*global-plot-options***)>=(***plot-request* **<...** *plot-request***>)**

controls the display of diagnostic plots. Three types of plots can be requested: trace plots, autocorrelation function plots, and kernel density plots. By default, the plots are displayed in panels unless the global plot option UNPACK is specified. Also when more than one type of plot is specified, the plots are grouped by parameter unless the global plot option GROUPBY=TYPE is specified. When you specify only one plot request, you can omit the parentheses around the plot-request, as shown in the following example:

```
plots=none
plots(unpack)=trace
plots=(trace density)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc mcmc data=exi seed=7 outpost=p1 plots=all;
   parm mu;
```

```
    prior mu ~ normal(0, sd=10);
    model y ~ normal(mu, sd=1);
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, then PROC MCMC produces, for each parameter, a panel that contains the trace plot, the autocorrelation function plot, and the density plot. This is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

The *global-plot-options* include the following:

FRINGE
adds a fringe plot to the horizontal axis of the density plot.

GROUPBY | GROUP=PARAMETER | TYPE
specifies how the plots are grouped when there is more than one type of plot. GROUPBY=PARAMETER is the default. The choices are as follows:

TYPE
specifies that the plots are grouped by type.

PARAMETER
specifies that the plots are grouped by parameter.

LAGS=*n*
specifies the number of autocorrelation lags used in plotting the ACF graph. By default, LAGS=50.

SMOOTH
smooths the trace plot with a fitted penalized B-spline curve (Eilers and Marx 1996).

UNPACKPANEL
UNPACK
specifies that all paneled plots are to be unpacked, so that each plot in a panel is displayed separately.

The *plot-requests* are as follows:

ALL
requests all types of plots. PLOTS=ALL is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

AUTOCORR | ACF
displays the autocorrelation function plots for the parameters.

DENSITY | D | KERNEL | K
displays the kernel density plots for the parameters.

NONE
suppresses the display of all plots.

TRACE | T
displays the trace plots for the parameters.

Consider a model with four parameters, X1–X4. Displays for various specifications are depicted as follows.

- PLOTS=(TRACE AUTOCORR) displays the trace and autocorrelation plots for each parameter side by side with two parameters per panel:

Display 1	Trace(X1)	Autocorr(X1)
	Trace(X2)	Autocorr(X2)
Display 2	Trace(X3)	Autocorr(X3)
	Trace(X4)	Autocorr(X4)

- PLOTS(GROUPBY=TYPE)=(TRACE AUTOCORR) displays all the paneled trace plots, followed by panels of autocorrelation plots:

Display 1	Trace(X1)	
	Trace(X2)	
Display 2	Trace(X3)	
	Trace(X4)	
Display 3	Autocorr(X1)	Autocorr(X2)
	Autocorr(X3)	Autocorr(X4)

- PLOTS(UNPACK)=(TRACE AUTOCORR) displays a separate trace plot and a separate correlation plot, parameter by parameter:

Display 1	Trace(X1)
Display 2	Autocorr(X1)
Display 3	Trace(X2)
Display 4	Autocorr(X2)
Display 5	Trace(X3)
Display 6	Autocorr(X3)
Display 7	Trace(X4)
Display 8	Autocorr(X4)

- PLOTS(UNPACK GROUPBY=TYPE)=(TRACE AUTOCORR) displays all the separate trace plots followed by the separate autocorrelation plots:

Display 1	Trace(X1)
Display 2	Trace(X2)
Display 3	Trace(X3)
Display 4	Trace(X4)
Display 5	Autocorr(X1)
Display 6	Autocorr(X2)
Display 7	Autocorr(X3)
Display 8	Autocorr(X4)

PROPCOV=*value*

specifies the method used in constructing the initial covariance matrix for the Metropolis-Hastings algorithm. The QUANEW and NMSIMP methods find numerically approximated covariance matrices at the optimum of the posterior density function with respect to all continuous parameters. The optimization does not apply to discrete parameters. The tuning phase starts at the optimized values; in some problems, this can greatly increase convergence performance. If the approximated covariance matrix is not positive definite, then an identity matrix is used instead. Valid values are as follows:

IND

uses the identity covariance matrix. This is the default. See the section "Tuning the Proposal Distribution" on page 6283.

CONGRA< (*optimize-options***) >**

performs a conjugate-gradient optimization.

DBLDOG< (*optimize-options***) >**

performs a double-dogleg optimization.

QUANEW< (*optimize-options***) >**

performs a quasi-Newton optimization.

NMSIMP | SIMPLEX< (*optimize-options***) >**

performs a Nelder-Mead simplex optimization.

The *optimize-options* are as follows:

ITPRINT

prints optimization iteration steps and results.

REOBSINFO < (*display-options*) >

displays the ODS table "Random Effect Observation Information." The table lists the name of each random effect, the unique values in the corresponding subject variable, the number of observations in each subject, and the observation indices for each subject value.

To understand how this option works, consider the following statements:

```
data input;
   array names{*} $ n1-n10 ("John" "Mary" "Chris" "Rob" "Greg"
                    "Jen" "Henry" "Alice" "James" "Toby");
   call streaminit(17);
   do i = 1 to 20;
      j = ceil(rand("uniform") * 10 );
      index = names[j];
      output;
   end;
   drop n: j;
run;

proc print data=input;
run;
```

The input data set (Figure 80.11) contains the index variable, which indicates subjects in a hypothetical random-effects model.

Figure 80.11 Subject Variable in an Input Data Set

Obs	i	index
1	1	Mary
2	2	James
3	3	Mary
4	4	Greg
5	5	Chris
6	6	James
7	7	James
8	8	Chris
9	9	James
10	10	James
11	11	Chris
12	12	Rob
13	13	Rob
14	14	Greg
15	15	Greg
16	16	Alice
17	17	Jen
18	18	Alice
19	19	John
20	20	Chris

The following statements illustrate the use of the REOBSINFO option:

```
ods select reobsinfo;
proc mcmc data=input reobsinfo stats=none diag=none;
   random u ~ normal(0, sd=1) subject=index;
   model general(0);
run;
```

Figure 80.12 displays the "Random Effect Observation Information" table. The table contains the name of the random-effect parameter (u), the values of the subject variable index, the total number of observations, and the row index of these observations in each of the subject values.

Figure 80.12 Random Effect Observation Information

The MCMC Procedure

Random Effect Observation Information			
Parameter	Subject Values	Number of Observations in Subject	Observation Indices
u	Mary	2	1 3
	James	5	2 6 7 9 10
	Greg	3	4 14 15
	Chris	4	5 8 11 20
	Rob	2	12 13
	Alice	2	16 18
	Jen	1	17
	John	1	19

The *display-options* are as follows:

MAXVALUEPRINT=*number* | **ALL**

MAXVPRINT=*number* | **ALL**

prints the number of subject values for each random effect (that is, the number of rows that are displayed in the "Random Effect Observation Information" table for each random effect). The default value is 20. MAXVALUEPRINT=ALL displays all subject values.

MAXOBSPRINT=*number* | **ALL**

MAXOPRINT=*number* | **ALL**

prints the number of observation indices for each subject value of every random effect (that is, the maximum number of indices that are displayed in the "Observation Indices" column in the "Random Effect Observation Information" table). The default value is 20. MAXOBSPRINT=ALL displays indices for every subject value.

SCALE=*value*

controls the initial multiplicative scale to the covariance matrix of the proposal distribution. By default, SCALE=2.38. See the section "Scale Tuning" on page 6283 for more details.

SEED=*n*

specifies the random number seed. By default, SEED=0, and PROC MCMC gets a random number seed from the clock.

SIMREPORT=*n*

controls the number of times that PROC MCMC reports the expected run time of the simulation. This can be useful for monitoring the progress of CPU-intensive programs. For example, with SIMREPORT=2, PROC MCMC reports the simulation progress twice. By default, SIMREPORT=0, and there is no reporting. The expected run times are displayed in the log file.

SINGDEN=*value*

defines the singularity criterion in PROC MCMC. By default, SINGDEN=1E-11. The *value* indicates the exclusion of an endpoint in an interval. The mathematical notation "(0" is equivalent to "[*value*" in PROC MCMC—that is, $x < 0$ is treated as $x \leq$ *value* in PROC MCMC. The maximum SINGDEN allowed is 1E-6.

STATISTICS< (*global-options***) > = NONE | ALL |***stats-request***

STATS< (*global-options***) > = NONE | ALL |***stats-request***

specifies options for posterior statistics. By default, PROC MCMC computes the posterior mean, standard deviation, quantiles, and two 95% credible intervals: equal-tail and highest posterior density (HPD). Other available statistics include the posterior correlation and covariance. See the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. You can request all of the posterior statistics by specifying STATS=ALL. You can suppress all the calculations by specifying STATS=NONE.

You can use postprocessing autocall macros to calculate posterior summary statistics of the posterior samples after PROC MCMC has exited. See the section "Autocall Macros for Postprocessing" on page 6343.

You can specify the following *global-options* to display interval and percentile estimates:

ALPHA=*numeric-list*

specifies the α level for the equal-tail and HPD intervals. The value α must be between 0 and 0.5. By default, ALPHA=0.05.

PERCENTAGE | PERCENT=*numeric-list*

calculates the posterior percentages. The *numeric-list* contains values between 0 and 100. By default, PERCENTAGE=(25 50 75).

You can specify the following *stats-requests*:

ALL

computes all posterior statistics. You can combine the option ALL with any other options. For example STATS(ALPHA=(0.02 0.05 0.1))=ALL computes all statistics with the default settings and intervals at α levels of 0.02, 0.05, and 0.1.

BRIEF

computes the posterior means, standard deviations, and the $100(1-\alpha)\%$ equal-tail intervals for each variable.

CORR

computes the posterior correlation matrix.

COV

computes the posterior covariance matrix.

INTERVAL

INT

computes the $100(1-\alpha)\%$ equal-tail and HPD credible intervals for each variable. For more information, see the sections "Equal-Tail Credible Interval" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures," and "Highest Posterior Density (HPD) Interval" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures." By default, $\alpha = 0.05$, but you can use the ALPHA= *global-option* to request other intervals of any probabilities.

NONE
suppresses all of the statistics.

SUMMARY
SUM
computes the posterior means, standard deviations, and percentile points for each variable. By default, the 25th, 50th, and 75th percentile points are produced, but you can use the global PERCENT= option to request specific percentile points.

TARGACCEPT=*value*
specifies the target acceptance rate for the random walk Metropolis algorithm. For more information, see the section "Metropolis and Metropolis-Hastings Algorithms" on page 156 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The numeric *value* must be between 0.01 and 0.99. By default, TARGACCEPT=0.45 for models that have one parameter; TARGACCEPT=0.35 for models that have two, three, or four parameters; and TARGACCEPT=0.234 for models that have more than four parameters (Roberts, Gelman, and Gilks 1997; Roberts and Rosenthal 2001).

TARGACCEPTI=*value*
specifies the target acceptance rate for the independence sampler algorithm. The independence sampler is used for blocks of binary parameters. For more information, see the section "Independence Sampler" on page 160 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The numeric *value* must be between 0 and 1. By default, TARGACCEPTI=0.6.

THIN=*n*
NTHIN=*n*
controls the thinning rate of the simulation. PROC MCMC keeps every *n*th simulation sample and discards the rest. All the posterior statistics and diagnostics are calculated using the thinned samples. By default, THIN=1. For more information, see the section "Burn-In, Thinning, and Markov Chain Samples" on page 162 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

TRACE
displays the result of each operation in each statement in the model program as it is executed. This debugging option is very rarely needed, and it produces voluminous output. If you use this option, also specify small numbers in the NMC=, NBI=, MAXTUNE=, and NTU= options.

TUNEWT=*value*
specifies the multiplicative weight used in updating the covariance matrix of the proposal distribution. The numeric *value* must be between 0 and 1. By default, TUNEWT=0.75. For more information, see the section "Covariance Tuning" on page 6284.

ARRAY Statement

ARRAY *arrayname* **[** *dimensions* **]** < **$** > < *variables-and-constants* > **;**

The ARRAY statement associates a name (of no more than eight characters) with a list of variables and constants. The ARRAY statement is similar to, but not the same as, the ARRAY statement in the DATA step, and it is the same as the ARRAY statements in the NLIN, NLP, NLMIXED, and MODEL procedures. The array name is used with subscripts in the program to refer to the array elements, as illustrated in the following statements:

```
array r[8] r1-r8;

do i = 1 to 8;
   r[i] = 0;
end;
```

The ARRAY statement does not support all the features of the ARRAY statement in the DATA step. Implicit indexing of variables cannot be used; all array references must have explicit subscript expressions. Only exact array dimensions are allowed; lower-bound specifications are not supported. A maximum of six dimensions is allowed.

Both variables and constants can be array elements. Constant array elements cannot have values assigned to them while variables can. Both the dimension specification and the list of elements are optional, but at least one must be specified. When the list of elements is not specified or fewer elements than the size of the array are listed, array variables are created by appending element numbers to the array name to complete the element list. You can index array elements by enclosing a subscript in braces ({ }) or brackets ([ ]), but not in parentheses (( )). The parentheses are reserved for function calls only.

For example, the following statement names an array day:

```
array day[365];
```

By default, the variables names are day1 to day365. However, since **day** is a SAS function, any subscript that uses parentheses gives you the wrong results. The expression **day(4)** returns the value 5 and does not reference the array element day4.

BEGINCNST/ENDCNST Statement

BEGINCNST ;

ENDCNST ;

The BEGINCNST and ENDCNST statements define a block within which PROC MCMC processes the programming statements only during the setup stage of the simulation. You can use the BEGINCNST and ENDCNST statements to define constants or import data set variables into arrays. Storing data in arrays enables you to work with data that are not identically distributed (see the section "Modeling Joint Likelihood" on page 6319) or to implement your own Markov chain sampler (see the section "UDS Statement" on page 6276). You can also use the BEGINCNST and ENDCNST statements to assign initial values to the parameters (see the section "Assignments of Parameters" on page 6288).

Assign Constants

Whenever you have programming statements that calculate constants that do not need to be evaluated multiple times throughout the simulation, you should put them within the BEGINCNST and ENDCNST statements. Using these statements can reduce redundant processing. For example, you can assign a constant to a symbol or fill in an array with numbers:

```
array cnst[17];
begincnst;
   offset = 17;
   do i = 1 to 17;
      cnst[i] = i * i;
   end;
endcnst;
```

During the setup process, PROC MCMC evaluates the programming statements within the BEGINCNST/ENDCNST once for each observation in the data set and ignores the statements in the rest of the simulation.

READ_ARRAY Function

Sometimes you might need to store variables, either from the current input data set or from a different data set, in arrays and use these arrays to specify your model. The READ_ARRAY function is convenient for that purpose.

The following two forms of the READ_ARRAY function are available:

rc = READ_ARRAY (*data-set***,** *array***) ;**

rc = READ_ARRAY (*data-set***,** *array* < **,** *"col-name1"*> < **,** *"col-name2"*> < **,** . . . >**) ;**

where

- *rc* returns 0 if the function is able to successfully read the data set.
- *data-set* specifies the name of the data set from which the array data is read. The value specified for *data-set* must be a character literal or a variable that contains the member name (libname.memname) of the data set to be read from.
- *array* specifies the PROC MCMC array variable into which the data is read. The value specified for *array* must be a local temporary array variable because the function might need to grow or shrink its size to accommodate the size of the data set.
- *col-name* specifies optional names for the specific columns of the data set that are read. If specified, *col-name* must be a literal string enclosed in quotation marks. In addition, *col-name* cannot be a PROC MCMC variable. If column names are not specified, PROC MCMC reads all of the columns in the data set.

When SAS translates between an array and a data set, the array is indexed as [row,column].

The READ_ARRAY function attempts to dynamically resize the array to match the dimensions of the input data set. Therefore, the array must be dynamic; that is, the array must be declared with the /NOSYMBOLS option.

For examples that use the READ_ARRAY function, see "Modeling Joint Likelihood" on page 6319, "Example 80.14: Time Independent Cox Model" on page 6451, and "Example 80.19: Implement a New Sampling Algorithm" on page 6478.

BEGINNODATA/ENDNODATA Statements

BEGINNODATA ;

ENDNODATA ;

BEGINPRIOR ;

ENDPRIOR ;

The BEGINNODATA and ENDNODATA statements define a block within which PROC MCMC processes the programming statements without stepping through the entire data set. The programming statements are executed only twice: at the first and the last observation of the data set. The BEGINNODATA and ENDNODATA statements are best used to reduce unnecessary observation-level computations. Any computations that are identical to every observation, such as transformation of parameters, should be enclosed in these statements.

At the first observation, PROC MCMC executes all programming statements, including those that are enclosed by these two statements. This enables a quick update of all the symbols enclosed by the BEGINNODATA and ENDNODATA statements. The goal is to ensure that subsequent statements (for example, the MODEL statement) use symbol values that have been calculated correctly. At the last observation, PROC MCMC executes the enclosed programming statements again and adds the log of the prior density to the log of the posterior density.

The BEGINPRIOR and ENDPRIOR statements are aliases for the BEGINNODATA and ENDNODATA statements, respectively. You can enclose PRIOR statements in the BEGINNODATA and ENDNODATA statements.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC MCMC to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the MCMC procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.

- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CMPTMODEL Statement

CMPTMODEL *required-options conditionally-required-options* < *options* > **;**

The CMPTMODEL statement computes predicted concentrations from a specified one-, two-, or three-compartment model. The CMPTMODEL statement includes three types of options:

- The following *required-options* are required: PCONC=, TIME=, NCOMPS=, ADMTYPE=, and PARMTYPE=.
- The following *conditionally-required-options* are conditionally required depending on the specification of the *required-options*: CL*n*=, VOL*n*=, K12=, K21=, K13=, K31=, KA=, K*n*0=, RATE=, and DURN=.
- The following *options* are optional: DOSE*n*=, SCALE*n*=, PCONC0=, PCONC2=, and PCONC3=.

You must specify the following *required-options*:

ADMTYPE=IVB | INF | ORAL
specifies the administration type. You can specify the following values:

IVB specifies the bolus type of administration.

INF specifies the infusion type of administration.

ORAL specifies the oral type of administration.

NCOMPS=1 | 2 | 3
specifies the number of compartments in a model.

PARMTYPE=1 | 2
PTYPE=1 | 2
specifies the parameterization type. You can specify the following values:

1 parameterizes the compartment model in terms of elimination and transfer rate constants.

2 parameterizes the compartment model in terms of clearance and volume constants.

PCONC=*variable*
PCONC1=*variable*
specifies the outcome variable that is the predicted concentration in the first or central compartment at each time point.

TIME=*variable*
specifies the time value at which the predicted concentrations are computed, where *variable* can be a data set variable or a defined variable.

You can combine the NCOMPS=, ADMTYPE=, and PARMTYPE= options to fit a compartment model (one-, two-, or three- compartments) with different administration methods (bolus, infusion, or oral) using different parameterizations. The PCONC= variable is the outcome that is the evaluated predicted concentration at the TIME= value. You can use this variable in model specification in your program.

You might need to specify one or more of the following *conditionally-required-options* based on what you specify in the *required-options*:

CL*n***=***variable*
specifies the clearance value for the *n*th compartment, where *n* takes the values of 1, 2, or 3. (CL is an alias for CL1.) This option is valid only in models that require clearance (PARMTYPE=2). The *variable* can be a data set variable or a defined variable.

- You must specify a CL1= option when you specify a one-compartment model (NCOMPS=1)
- You must specify CL1= and CL2= options when you specify a two-compartment model (NCOMPS=2)
- You must specify CL1=, CL2=, and CL3= options when you specify a three-compartment model (NCOMPS=3)

DURN=*variable*
specifies the duration of the infusion. This option is valid only in infusion models (ADMTYPE=INF). The *variable* can be a data set variable or a defined variable. In an infusion model, you must specify either this option or the RATE= option.

KA=*variable*
specifies the absorption rate constant. This option is valid only for the oral type of compartment models (ADMTYPE=ORAL).

K12=*variable*
specifies the transfer rate constant from the first (central) compartment to the second compartment. This option is valid only in models that require transfer rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable. You must specify this option in a two- or three-compartment model.

K21=*variable*
specifies the transfer rate constant from the second compartment to the first (central) compartment. This option is valid only in models that require transfer rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable. You must specify this option in a two- or three- compartment model.

K13=*variable*
specifies the transfer rate constant from the first (central) compartment to the third compartment. This option is valid only in models that require transfer rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable. You must specify this option in a three-compartment model.

K31=*variable*
specifies the transfer rate constant from the third compartment to the first (central) compartment. This option is valid only in models that require transfer rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable. You must specify this option in a three-compartment model.

K*n***0=***variable*
specifies the elimination rate constant for the *n*th compartment, where *n* takes the values of 1, 2, or 3. (Ke is an alias for K10.) This option is valid only in models that require elimination rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable.

- You must specify a K10= option when you specify a one-compartment model (NCOMPS=1)
- You must specify a K10= option and you can optionally specify a K20= option when you specify a two-compartment model (NCOMPS=2)
- You must specify a K10= option and you can optionally specify K20= and K30= options when you specify a three-compartment model (NCOMPS=3)

RATE=*variable*
specifies the rate of the infusion. This option is valid only in infusion models (ADMTYPE=INF). The *variable* can be a data set variable or a defined variable. In an infusion model, you must specify either this option or the DURN= option.

VOL*n***=***variable*
specifies the apparent volume of distribution of a drug in the *n*th compartment, where *n* takes the values of 1, 2, or 3. (VOL is an alias for VOL1.) This option is valid only in models that require volume (PARMTYPE=2). The *variable* can be a data set variable or a defined variable.

- You must specify a VOL1= option when you specify a one-compartment model (NCOMPS=1)
- You must specify VOL1= and VOL2= options when you specify a two-compartment model (NCOMPS=2)
- You must specify VOL1=, VOL2=, and VOL3= options when you specify a three-compartment model (NCOMPS=3)

You can also specify the following optional *options*:

DOSE*n***=***variable*
specifies the amount of the dose for the *n*th compartment, where *n* takes the values of 0, 1, 2, or 3. (DOSE is an alias for DOSE1.) The DOSE0= option specifies the amount of dose in the zero (depot) compartment, and is valid only in oral models (ADMTYPE=ORAL). The *variable* can be a data set variable or a defined variable.

- DOSE1= is optional and valid for one-, two-, and three-compartment models
- DOSE2= is optional and valid only for two-, and three-compartment models
- DOSE3= is optional and valid only for a three-compartment model

When ADMTYPE=ORAL is specified, DOSE0=1, DOSE1=0, DOSE2=0, and DOSE3=0 by default. Otherwise, DOSE1=1, DOSE2=0, and DOSE3=0 by default.

PCONC0=*variable*
specifies the outcome *variable* that is the predicted concentration of the zero (depot) compartment. This option is valid only when the ADMTYPE= ORAL option is specified.

PCONC2=*variable*
specifies the outcome *variable* that is the predicted concentration of the second compartment. This option is valid only in two- and three-compartment models.

PCONC3=*variable*
specifies the outcome *variable* that is the predicted concentration of the second compartment. This option is valid only in three-compartment models.

SCALE*n***=***variable*
specifies the scale value for the *n*th compartment. The *variable* can be a data set variable or a defined variable. SCALE0= option is valid only in oral models (ADMTYPE=ORAL).

- SCALE1= is optional and valid for one-, two-, and three-compartment models
- SCALE2= is optional and valid only for two-, and three-compartment models
- SCALE3= is optional and valid only for a three-compartment model

By default, all optional SCALE*n* values are set to 1.

For more information, see the section "Compartment Models" on page 6323.

MODEL Statement

MODEL *dependent-variable-list* ~ *distribution* < *options* > **;**

The MODEL statement specifies the conditional distribution of the data given the parameters (the likelihood function). You specify a single dependent variable or a list of dependent variables, a tilde ~, and then a distribution with its arguments. The dependent variables can be variables from the input data set or functions of the symbols in the program. You must specify the dependent variables unless you use the GENERAL function or the DGENERAL function (see the section "Specifying a New Distribution" on page 6305 for more details).

The MODEL statement assumes that the observations are independent of each other, conditional on the model parameters. If you want to model dependent data—that is, $f(y_i|\theta, y_j)$ for $j \neq i$—you can use the JOINTMODEL option in the PROC MCMC statement. See the section "Modeling Joint Likelihood" on page 6319 for more details. By default, the log-likelihood value is the sum of the individual log-likelihood value for each observation.

You can specify multiple MODEL statements. You can define likelihood functions that are independent of each other. For example, in the following statements, the dependent variables y1 and y2 are independent of each other:

```
model y1 ~ normal(alpha, var=s21);
model y2 ~ normal(beta,  var=s22);
```

Alternatively, you can use marginal and conditional distributions to define a joint log-likelihood function for multiple dependent variables. For example, the following statements jointly define a distribution over (y1, y2). They specify a marginal distribution for the dependent variable y1 and a conditional distribution for the dependent variable y2:

```
model y1 ~ normal(alpha, var=s21);
model y2 ~ normal(beta * y1,  var=s22);
```

Every program must have at least one MODEL statement. If you want to run a Monte Carlo simulation that does not require a response variable, use the GENERAL function in the MODEL statement:

```
model general(0);
```

PROC MCMC interprets the statement as a flat likelihood function with a constant log-likelihood value of 0.

PROC MCMC is a programming language that is similar to the DATA step, and the order of statement evaluation is important. For example, the MODEL statement must come after any SAS programming statements that define or modify arguments used in the construction of the log likelihood. In PROC MCMC, a symbol can be defined multiple times and used at different places. Using an expression out of order produces erroneous results that can also be hard to detect.

Do not embed the MODEL statement within programming statements. For example, suppose you have three response variables, y1, y2, and y3, and want to model each with a normal distribution. The following statements lead to erroneous output:

```
array Y[3] y1 y2 y3;
do i = 1 to 3;
   model y[i] ~ normal(mu, sd=s);
end;
```

Instead, you should do one of the following.

- Use separate MODEL statements:

  ```
  model y1 ~ normal(mu, sd=s);
  model y2 ~ normal(mu, sd=s);
  model y3 ~ normal(mu, sd=s);
  ```

- Use the GENERAL function to construct a joint distribution of the three dependent variables and use a single MODEL statement to specify the log-likelihood function:

  ```
  llike = logpdf("normal", y1, mu, s) +
          logpdf("normal", y2, mu, s) +
          logpdf("normal", y3, mu, s);
  model y1 y2 y3 ~ general(llike);
  ```

  See the section "Specifying a New Distribution" on page 6305 for more information about how to use the GENERAL function to specify an arbitrary distribution.

Missing data are allowed in the response variables; the MODEL statement augments missing data automatically. In each iteration, PROC MCMC samples missing values from their posterior distributions and

incorporates them as part of the simulation. PROC MCMC creates one variable for each missing response value. There are two ways to create the missing value variable names; see the NAMESUFFIX= option for the naming convention of the variables.

Distributions in MODEL Statement

Standard distributions that the MODEL statement supports are listed in the Table 80.2 (univariate) and Table 80.3 (multivariate). See the section "Standard Distributions" on page 6289 for density specifications. You can also specify all distributions except the multinomial distribution in the PRIOR and HYPERPRIOR statements. The RANDOM statement supports only a subset of the distributions (see Table 80.4).

PROC MCMC allows some distributions to be parameterized in multiple ways. For example, you can specify a normal distribution with a variance, standard deviation, or precision parameter. For distributions that have different parameterizations, you must specify an option to clearly name the ambiguous parameter. For example, in the normal distribution, you must indicate whether the second argument represents variance, standard deviation, or precision.

All univariate distributions, except for binary and uniform distributions, can have the optional LOWER= and UPPER= arguments (which specify a truncated density) and CLOWER= and CUPPER= arguments (which specify censoring). For more information, see the section "Truncation and Censoring" on page 6309. Truncation and censoring are not supported for multivariate distributions.

Table 80.2 Univariate Distributions

Distribution Name	Definition
beta(<**a=**>α, <**b=**>β)	Beta distribution with shape parameters α and β
binary(<**prob|p=**>p)	Binary (Bernoulli) distribution with probability of success p. You can use the alias **bern** for this distribution.
binomial (<**n=**>n, <**prob|p=**>p)	Binomial distribution with count n and probability of success p
cauchy (<**location|loc|l=**>θ, <**scale|s=**>λ)	Cauchy distribution with location θ and scale λ
chisq(<**df=**>ν)	χ^2 distribution with ν degrees of freedom
dgeneral(*ll*)	General log-likelihood function that you construct using SAS programming statements for single or multiple discrete parameters. Also see the function **general**. The name **dlogden** is an alias for this function.
expchisq(<**df=**>ν)	Log transformation of a χ^2 distribution with ν degrees of freedom: $\theta \sim$ **chisq**$(\nu) \Leftrightarrow \log(\theta) \sim$ **expchisq**(ν). You can use the alias **echisq** for this distribution.

Table 80.2 *continued*

Distribution Name	Definition
expexpon(**scale**|**s=** λ) **expexpon**(**iscale**|**is=** λ)	Log transformation of an exponential distribution with scale or inverse-scale parameter λ: $\theta \sim$ **expon**$(\lambda) \Leftrightarrow \log(\theta) \sim$ **expexpon**(λ). You can use the alias **eexpon** for this distribution.
expGamma(< **shape**|**sp=** > *a*, **scale**|**s=** λ) **expGamma**(< **shape**|**sp=** > *a*, **iscale**|**is=** λ)	Log transformation of a gamma distribution with shape a and scale or inverse-scale λ: $\theta \sim$ **gamma**$(a, \lambda) \Leftrightarrow \log(\theta) \sim$ **expgamma**(a, λ). You can use the alias **egamma** for this distribution.
expichisq(< **df=** > ν)	Log transformation of an inverse χ^2 distribution with ν degrees of freedom: $\theta \sim$ **ichisq**$(\nu) \Leftrightarrow \log(\theta) \sim$ **expichisq**(ν). You can use the alias **eichisq** for this distribution.
expiGamma(< **shape**|**sp=** > *a*, **scale**|**s=** λ) **expiGamma**(< **shape**|**sp=** > *a*, **iscale**|**is=** λ)	Log transformation of an inverse gamma distribution with shape a and scale or inverse-scale λ: $\theta \sim$ **igamma**$(a, \lambda) \Leftrightarrow \log(\theta) \sim$ **expigamma**(a, λ). You can use the alias **eigamma** for this distribution.
expsichisq(< **df=** > ν, < **scale**|**s=** > s)	Log transformation of a scaled inverse χ^2 distribution with ν degrees of freedom and scale parameter s: $\theta \sim$ **sichisq**$(\nu) \Leftrightarrow \log(\theta) \sim$ **expsichisq**(ν). You can use the alias **esichisq** for this distribution.
expon(**scale**|**s=** λ) **expon**(**iscale**|**is=** λ)	Exponential distribution with scale or inverse-scale parameter λ
gamma(< **shape**|**sp=** > *a*, **scale**|**s=** λ) **gamma**(< **shape**|**sp=** > *a*, **iscale**|**is=** λ)	Gamma distribution with shape a and scale or inverse-scale λ
geo(< **prob**|**p=** > p)	Geometric distribution with probability p

Table 80.2 *continued*

Distribution Name	Definition
general(*ll*)	General log-likelihood function that you construct using SAS programming statements for a single or multiple continuous parameters. The argument *ll* is an expression for the log of the distribution. If there are multiple variables specified before the tilde in a MODEL, PRIOR, or HYPERPRIOR statement, *ll* is interpreted as the log of the joint distribution for these variables. Note that in the MODEL statement, the response variable specified before the tilde is just a place holder and is of no consequence; the variable must have appeared in the construction of *ll* in the programming statements. **general**(*constant*) is equivalent to a uniform distribution on the real line. You can use the alias **logden** for this distribution.
ichisq(<**df=**>ν)	Inverse χ^2 distribution with ν degrees of freedom
igamma(<**shape|sp=**> a, **scale|s=** λ) **igamma**(<**shape|sp=**> a, **iscale|is=** λ)	Inverse gamma distribution with shape a and scale or inverse-scale λ
laplace(<**location|loc|l=**> θ, **scale|s=** λ) **laplace**(<**location|loc|l=**> θ, **iscale|is=** λ)	Laplace distribution with location θ and scale or inverse-scale λ. This is also known as the *double exponential* distribution. You can use the alias **dexpon** for this distribution.
logistic(<**location|loc|l=**> a, <**scale|s=**> b)	Logistic distribution with location a and scale b
lognormal(<**mean|m=**> μ, **sd=** λ) **lognormal**(<**mean|m=**> μ, **var|v=** λ) **lognormal**(<**mean|m=**> μ, **prec=** λ)	Log-normal distribution with mean μ and a value of λ for the standard deviation, variance, or precision. You can use the aliases **lognormal** or **lnorm** for this distribution.
negbin(<**n=**> n, <**prob|p=**> p)	Negative binomial distribution with count n and probability of success p. You can use the alias **nb** for this distribution.
normal(<**mean|m=**> μ, **sd=** λ) **normal**(<**mean|m=**> μ, **var|v=** λ) **normal**(<**mean|m=**> μ, **prec=** λ)	Normal (Gaussian) distribution with mean μ and a value of λ for the standard deviation, variance, or precision. You can use the aliases **gaussian**, **norm**, or **n** for this distribution.
pareto(<**shape|sp=**> a, <**scale|s=**> b)	Pareto distribution with shape a and scale b
poisson(<**mean|m=**> λ)	Poisson distribution with mean λ

Table 80.2 *continued*

Distribution Name	Definition
sichisq(<**df=**> ν, <**scale**|**s=**> s)	Scaled inverse χ^2 distribution with ν degrees of freedom and scale parameter s
t(<**mean**|**m=**> μ, **sd=** λ, <**df=**> ν) **t**(<**mean**|**m=**> μ, **var**|**v=** λ, <**df=**> ν) **t**(<**mean**|**m=**> μ, **prec=** λ, <**df=**> ν)	T distribution with mean μ, standard deviation or variance or precision λ, and ν degrees of freedom
table(<**p=**> p)	Table (categorical) distribution with probability vector p. You can also use the alias **cat** for this distribution.
uniform(<**left**|**l=**> a, <**right**|**r=**> b)	Uniform distribution with range a and b. You can use the alias **unif** for this distribution.
wald(<**mean**|**m=**> μ, <**iscale**|**is=**> λ)	Wald distribution with mean parameter μ and inverse scale parameter λ. This is also known as the *Inverse Gaussian* distribution. You can use the alias **igaussian** for this distribution.
weibull(μ, c, σ)	Weibull distribution with location (threshold) parameter μ, shape parameter c, and scale parameter σ

Table 80.3 Multivariate Distributions

Distribution Name	Definition
dirichlet(<**alpha=**>α)	Dirichlet distribution with parameter vector α, where α must be a one-dimensional array of length greater than 1
iwish(<**df=**>ν, <**scale=**>S)	Inverse Wishart distribution with ν degrees of freedom and symmetric positive definite scale array S
multinom(<**p=**>p)	Multinomial distribution with probability vector p
mvn(<**mu=**>μ, <**cov=**>Σ)	Multivariate normal distribution with mean vector μ and covariance matrix Σ
MVNAR(<**mu=**>μ, **sd=** λ, <**rho=**>ρ) **MVNAR**(<**mu=**>μ, **var=** λ, <**rho=**>ρ) **MVNAR**(<**mu=**>μ, **prec=** λ, <**rho=**>ρ)	Multivariate normal distribution with mean vector μ and a covariance matrix Σ. The covariance matrix Σ is a multiple of the scale and a matrix with a first-order autoregressive structure. When RHO=0, this distribution becomes a multivariate normal distribution with shared variance.

Options for the MODEL Statement

The *options* in the MODEL statement apply when there are missing values in the response variable, or in the case of the ICOND= option, when there are lag or lead variables for the response variable. You can specify the following *options*.

ICOND=*variable-list* | *numeric-list*

specifies the initial conditions (or initial states) of the lag or lead variables for the response variable when the observation indices are out of the range. (For more information about rules of constructing lag and lead variables in PROC MCMC, see the section "Access Lag and Lead Variables" on page 6320.) For example, you can use the ICOND= option to specify the lag 1 value of the response for the first observation. This option works similarly to the ICOND= option in the RANDOM statement, except that the index is done according to observations, not a subject variable. The initial conditions can be model parameters, functions of model parameters, or constants. By default, *numeric-list* is set to 0.

The ICOND= option in a MODEL statement sets the initial conditions for all lag or lead variables (for the associated response variable) that appear in the program, not just those that appear in the MODEL statement. Suppose you have a maximum L number of lag variables and a maximum M number of lead variables of the response y in the program, and there are n observations. The program has the following variables that need to be resolved during the simulation:

$$\mathsf{y}_{L+1}, \ldots, \mathsf{y}_0, \mathsf{y}_1, \ldots, \mathsf{y}_n, \mathsf{y}_{n+1}, \ldots, \mathsf{y}_{n+M}$$

Of these variables, n are observations of y from the input data set and the remaining L+M are initial conditions that are specified in the ICOND= option. In essence, the ICOND= numeric list stretches the input data set by filling in the first L and last M values. As PROC MCMC steps through the input data set, it resolves the current, lagged, and lead variables according to this stretched vector of observations.

The *variable-list* (or the *number-list*) should be of length L+M, which can be greater than the number of lag or lead response variables that appear in a program. Here is an example.

Suppose you want to fit an autoregressive model of order 2. And instead of two lagged values, the model requires only the second lag,

$$Y_i = A + \phi \cdot Y_{i-2} + \epsilon_i$$

where the noise is assumed to be normal. To specify this autoregressive model, you would use the statements

```
mu = A + phi * y.l2;
model y ~ normal(mu, var=s2) icond=(-2 -1);
```

where the Y_{i-2}, or the lag-2 of Y, variable is constructed by concatenating the variable name, the letter L (for "lag"), and a lag number.

This model requires two initial conditions for the lag-2 variable of Y, at the first and second observations. Therefore, the ICOND= option expects a numeric list of two values. In this example, at the first observation, the variable y.l2 is given a value of –2; at the second observation, y.l2 is given a value of –1. If you provide a partial list that contains less than the expected number of conditions, PROC MCMC fills the remaining list with the value of 0.

INITIAL=*SAS-data-set* | *constant* | *numeric-list*

specifies the initial values of the missing values. By default, PROC MCMC uses a sample average of the nonmissing values of a response variable as the starting values for all missing values in the simulation for that variable. You can use the INITIAL= option to start the Markov chain at a different place.

If you use a *SAS-data-set* to store initial values, the data set must consist of variable names that agree with the missing variable names that are used by PROC MCMC. The easiest way to find the names of the internally created variables is to run a default analysis with a very small number of simulations and check the variable names in the OUTPOST= data set. You can provide a subset of the initial values in the *SAS-data-set*, and PROC MCMC uses a default mechanism to fill in the rest of the missing initial values.

For example, the following statement creates a data set with initial values for the first three missing values of a response variable:

```
data RandomInit;
   input y_1 y_2 y_3;
   datalines;
2.3 3 -3
;
```

The following MODEL statement uses the values in the RandomInit data set as the initial values of the corresponding missing values in the model:

```
model y ~ normal(0,var=s2u) init=randominit;
```

Specifying a *constant* assigns that constant as the initial value to all missing values in that response variable. For example, the following statement assigns the value 5 to be used as an initial value for all missing y_i in the model:

```
model y ~ normal(0,var=s2u) init=5;
```

If you have a multidimensional response variable, you can provide a list of numbers that have the same length as the dimension of your response array. Each number is then given to all corresponding missing variables in order. For example, the following statement assigns the value 2 to be used as an initial value for all missing $w1_i$ and the value 3 to be used for all missing $w2_i$ in the model:

```
array w[2] w1 w2;
model w ~ mvn(mu, cov) init=(2 3);
```

MONITOR= (*symbol-list* | *number-list* | **RANDOM(***number***))**

outputs analysis for selected missing data variables. You can choose to monitor the missing values by listing the response variable names, the missing data variable names, or indices, or you can have them randomly selected by PROC MCMC.

For example, suppose that the data set contains 10 observations and the response variable y has missing values in observations 2, 3, 7, 9, and 10. To monitor all missing data variables (five in total), you specify the response variable name in the MONITOR= option:

```
model y ~ normal(0,var=s2u) monitor=(y);
```

Suppose you want to monitor the missing data variables that correspond to the missing values in observations 2, 3, and 10. You have two options: provide either a list of variable names or a list of indices.

The following statement selects monitored variables by their variable names:

```
model y ~ normal(0,var=s2u) monitor=(y_2 y_3 y_10);
```

The variable names must match the internally created variable names for each missing value. See NAMESUFFIX= option for the naming convention of the variables. By default, the names are created by concatenating the response variable with the observation index; hence you use the name_obs format to construct the names. The numbers 2, 3, and 10 are the corresponding observation indices to the missing values in the input data set.

The following statement selects monitored variables by indices:

```
model y ~ normal(0,var=s2u) monitor=(1 2 5);
```

The indices are not a list of the observation numbers, but rather the order by which the missing values appear in the data set: PROC MCMC reports back the first, the second, and the fifth missing value variables that it creates. The actual variable names that appear in the output are still y_2, y_3, and y_10, honoring the control of the NAMESUFFIX= option.

Lastly, PROC MCMC can randomly choose a subset of the variables to monitor. The following statement randomly selects 3 variables to monitor:

```
model y ~ normal(0,var=s2u) monitor=(random(3));
```

The list of the random indices is controlled by the SEED= option in the PROC MCMC statement. Therefore, the selected variables will be the same when the SEED= option is the same.

NAMESUFFIX=OBSERVATION | POSITION | ORDER

specifies how the names of the missing data variables are created. By default, the names are created by concatenating the response variable symbol, an underscore ("_"), and the observation number of the missing value.

NAMESUFFIX=OBSERVATION constructs the parameter names by appending the observation number to the response variable symbol. This is the default. NAMESUFFIX=POSITION or NAMESUFFIX=ORDER construct the parameter names by appending the numbers 1, 2, 3, and so on, where the number indicates the order in which the missing values appear in the data set.

For example, suppose you have a response variable y with 10 observations in total, of which five are missing (observations 2, 3, 7, 9, and 10). By default, PROC MCMC creates five variable names y_2, y_3, y_7, y_9, and y_10. Using NAMESUFFIX=POSITION changes the names to y_1, y_2, y_3, y_4, and y_5.

NOOUTPOST

suppresses the output of the posterior samples of missing data variables to the posterior output data set (which is specified in the OUTPOST= option in the PROC MCMC statement). In models with a large number of missing values (for example, tens of thousands), PROC MCMC can run faster if it does not save the posterior samples.

When you specify both the NOOUTPOST option and the MONITOR= option, PROC MCMC outputs the list of variables that are monitored.

The maximum number of variables that can be saved to an OUTPOST= data set is 32,767. If the total number of parameters in your model, including the number of missing data variables, exceeds the limit, the NOOUTPOST option is evoked automatically and PROC MCMC does not save the missing value draws to the posterior output data set. You can use the MONITOR= option to select a subset of the parameters to store in the OUTPOST= data set.

PARMS Statement

PARMS *name* |(*name-list*)< **=** > < **{** > *number* | *number-list* < **}** >
< *name* |(*name-list*)< **=** > < **{** > *number* | *number-list* < **}** > . . . >
< / *options* > **;**

The PARMS statement lists the names of the parameters in the model and specifies optional initial values for these parameters. These parameters are referred to as the *model parameters*. You can specify multiple PARMS statements. Each PARMS statement defines a block of parameters, and the blocked Metropolis algorithm updates the parameters in each block simultaneously. See the section "Blocking of Parameters" on page 6280 for more details. PROC MCMC generates missing initial values from the prior distributions whenever needed, as long as they are the standard distributions and not the GENERAL or DGENERAL function.

If your model contains a multidimensional parameter (for example, a parameter with a multivariate normal prior distribution), you must declare the parameter as an array (using the ARRAY statement). You can use braces { } after the parameter name in the PARM statement to assign initial values. For example:

```
array mu[3];
parms mu {1 2 3};
```

You cannot use the ARRAY statement to assign initial values. If you use the ARRAY statement to store values in array elements, the declared array becomes a constant array and cannot be used as parameters in the PARMS statement. For example, the following statement assigns three numbers to mu:

```
array mu[3] (1 2 3);
```

The array mu can no longer be a model parameter.

Every parameter in the PARMS statement must have a corresponding prior distribution in the PRIOR statement. The program exits if this one-to-one requirement is not satisfied.

You can specify the following *options* to control different samplers explicitly for that block of parameters.

NORMAL | N

uses the normal proposal distribution in the random walk Metropolis. This is the default.

T <(*df*)>

uses the t distribution with *df* degrees of freedom as an alternative proposal distribution. A t distribution with a small number of degrees of freedom has thicker tails and can sometimes improve the mixing of the Markov chain. When $df > 100$, the normal distribution is used instead.

SLICE

applies the slice sampler to each parameter in the PARMS statement individually. See the section"Slice Sampler" on page 159 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details. PROC MCMC does not implement a multidimensional version of the slice sampler. Because the slice sampler usually requires multiple evaluations of the objective function (the posterior distribution) in each iteration, the associated computational cost could be potentially high with this sampling algorithm.

UDS

implements a user-defined sampler for any of the parameters in the block. See the section "UDS Statement" on page 6276 for details and "Example 80.19: Implement a New Sampling Algorithm" on page 6478 for a realistic example. When you specify the UDS option, PROC MCMC hands off the sampling of these parameters to you at each iteration and relies on your sampler to return a random draw from the conditional posterior distribution. This option is useful if you have a model-specific sampler that you want to implement or a new algorithm that can improve the convergence and mixing of the Markov chain. This functionality is for advanced users, and you should proceed with caution.

PREDDIST Statement

PREDDIST <'*label*'> **OUTPRED=***SAS-data-set* <**NSIM=***n*> <**COVARIATES=***SAS-data-set*>
<**STATISTICS=***options*> **;**

The PREDDIST statement creates a new SAS data set that contains random samples from the posterior predictive distribution of the response variable. The posterior predictive distribution is the distribution of unobserved observations (prediction) conditional on the observed data. Let $\mathbf{y}$ be the observed data, $\mathbf{X}$ be the covariates, θ be the parameter, and $\mathbf{y}_{\text{pred}}$ be the unobserved data. The posterior predictive distribution is defined to be the following:

$$\begin{aligned} p(\mathbf{y}_{\text{pred}}|\mathbf{y},\mathbf{X}) &= \int p(\mathbf{y}_{\text{pred}},\theta|\mathbf{y},\mathbf{X})d\theta \\ &= \int p(\mathbf{y}_{\text{pred}}|\theta,\mathbf{y},\mathbf{X})p(\theta|\mathbf{y},\mathbf{X})d\theta \end{aligned}$$

Given the assumption that the observed and unobserved data are conditional independent given θ, the posterior predictive distribution can be further simplified as the following:

$$p(\mathbf{y}_{\text{pred}}|\mathbf{y},\mathbf{X}) = \int p(\mathbf{y}_{\text{pred}}|\theta)p(\theta|\mathbf{y},\mathbf{X})d\theta$$

The posterior predictive distribution is an integral of the likelihood function $p(\mathbf{y}_{\text{pred}}|\theta)$ with respect to the posterior distribution $p(\theta|\mathbf{y})$. The PREDDIST statement generates samples from a posterior predictive distribution based on draws from the posterior distribution of θ.

The PREDDIST statement works only on response variables that have standard distributions, and it does not support either the GENERAL or DGENERAL functions. Multiple PREDDIST statements can be specified, and an optional label (specified as a quoted string) helps identify the output.

The following list explains specifications in the PREDDIST statement:

COVARIATES=*SAS-data-set*
names the SAS data set that contains the sets of explanatory variable values for which the predictions are established. This data set must contain data with the same variable names as are used in the likelihood function. If you omit the COVARIATES= option, the DATA= data set specified in the PROC MCMC statement is used instead.

NSIM=*n*
specifies the number of simulated predicted values. By default, NSIM= uses the NMC= option value specified in the PROC MCMC statement.

OUTPRED=*SAS-data-set*
creates an output data set to contain the samples from the posterior predictive distribution. The output variable names are listed as resp_1–resp_*m*, where resp is the name of the response variable and *m* is the number of observations in the COVARIATES= data set in the PREDDIST statement. If the COVARIATES= data set is not specified, *m* is the number of observations in the DATA= data set specified in the PROC statement.

SAVEPARM
outputs to the OUTPRED= data set sampled parameter values that are used in each predictive draw.

STATISTICS< (*global-options***) > = NONE | ALL |***stats-request***
STATS< (*global-options***) > = NONE | ALL |***stats-request***
specifies options for calculating posterior statistics. This option works identically to the STATISTICS= option in the PROC statement. By default, this option takes the specification of the STATISTICS= option in the PROC MCMC statement.

For an example that uses the PREDDIST statement, see the section "Posterior Predictive Distribution" on page 6348.

PRIOR/HYPERPRIOR Statement

PRIOR *parameter-list* ∼ *distribution* **;**

HYPERPRIOR *parameter-list* ∼ *distribution* **;**

HYPER *parameter-list* ∼ *distribution* **;**

The PRIOR statement specifies the prior distribution of the model parameters. You must specify a single parameter or a list of parameters, a tilde ∼, and then a distribution with its parameters.

You can specify multiple PRIOR statements to define models with multiple prior components. Your model can have as many hierarchical levels as you want. But in many cases, such as random-effects models, it is better to use the RANDOM statements to build up the model hierarchy. The log of the prior is the sum of the log prior values from each of the PRIOR statements. Similar to the MODEL statement, you can use the PRIOR statement to specify marginal or conditional prior distributions. See the section "MODEL

Statement" on page 6254 for the names of the standard distributions and the section "Standard Distributions" on page 6289 for density specification.

The PRIOR statements are processed twice at every Markov chain simulation—that is, twice per pass through the data set. The statements are called at the first and the last observation of the data set, just as the BEGINNODATA and ENDNODATA statements are processed. If you run a Monte Carlo simulation that is data-independent, you can specify the NOLOGDIST option in the PROC MCMC statement to omit the calculation of the prior distribution. Omitting this calculation enables PROC MCMC to run faster.

The HYPERPRIOR statement is treated internally the same as the PRIOR statement. It provides a notational convenience in case you want to fit a multilevel hierarchical model. It specifies the hyperprior distribution of the prior distribution parameters. The log of the hyperprior is the sum of the log hyperprior values from each of the HYPERPRIOR statements.

Parameters in the PRIOR statements can appear as hyperparameters in the RANDOM statement. The reverse is not allowed: random-effects parameters cannot be hyperparameters in a PRIOR statement.

You can have a program that contains a RANDOM statement but no PRIOR statements. (In SAS 9.3 and earlier, each program had to contain a PRIOR statement.) A program that contains a RANDOM statement but no PRIOR statements could be a random-effects model with no fixed-effects parameters or hyperparameters to the random effects. A MODEL statement is still required in every program.

Programming Statements

This section lists the programming statements available in PROC MCMC to compute the priors and log-likelihood functions. This section also documents the differences between programming statements in PROC MCMC and programming statements in the DATA step. The syntax of programming statements used in PROC MCMC is identical to that used in the NLMIXED procedure (see Chapter 89, "The NLMIXED Procedure") and the MODEL procedure (see Chapter 24, "MODEL Procedure" (*SAS/ETS User's Guide*)). Most of the programming statements that can be used in the DATA step can also be used in PROC MCMC. See *SAS DATA Step Statements: Reference* for a description of SAS programming statements.

There are also a number of unique functions in PROC MCMC that calculate the log density of various distributions in the procedure. You can find them at the section "Using Density Functions in the Programming Statements" on page 6306.

For the list of matrix-based functions that is supported in PROC MCMC, see the section "Matrix Functions in PROC MCMC" on page 6313.

The following are valid statements:

ABORT;
ARRAY *arrayname* < **[** *dimensions* **]** > < **$** > < *variables-and-constants* >**;**
CALL *name* < **(***expression* < **,** *expression . . .* >**)** >**;**
DELETE;
DO < *variable* **=** *expression* < **TO** *expression* > < **BY** *expression* > >
< **,** *expression* < **TO** *expression* > < **BY** *expression* > > . . .
< **WHILE** *expression* > < **UNTIL** *expression* >**;**
END;
GOTO *statement-label***;**
IF *expression***;**
IF *expression* **THEN** *program-statement***;**
ELSE *program-statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement-label***;**
PUT < *variable* > < **=** > . . . **;**
RETURN;
SELECT < **(***expression***)** >**;**
STOP;
SUBSTR(*variable***,** *index***,** *length***)=** *expression***;**
WHEN (*expression***)***program-statement***;**
OTHERWISE *program-statement***;**

For the most part, the SAS programming statements work the same as they do in the DATA step, as documented in *SAS Programmers Guide: Essentials*. However, there are several differences:

- The ABORT statement does not allow any arguments.
- The DO statement does not allow a character index variable. Thus

  ```
  do i = 1,2,3;
  ```

  is supported; however, the following statement is not supported:

  ```
  do i = 'A','B','C';
  ```

- The PUT statement, used mostly for program debugging in PROC MCMC (see the section "Handling Error Messages" on page 6366), supports only some of the features of the DATA step PUT statement, and it has some features that are not available with the DATA step PUT statement:
  - The PROC MCMC PUT statement does not support line pointers, factored lists, iteration factors, overprinting, _INFILE_, _OBS_, the colon (:) format modifier, or "$".
  - The PROC MCMC PUT statement does support expressions, but the expression must be enclosed in parentheses. For example, the following statement displays the square root of x:

    ```
    put (sqrt(x));
    ```

- The WHEN and OTHERWISE statements enable you to specify more than one target statement. That is, DO/END groups are not necessary for multiple statement WHENs. For example, the following syntax is valid:

```
select;
   when (exp1) stmt1;
               stmt2;
   when (exp2) stmt3;
               stmt4;
end;
```

You should avoid defining variables that begin with an underscore (_). They might conflict with internal variables created by PROC MCMC. The MODEL statement must come after any SAS programming statements that define or modify terms used in the construction of the log likelihood.

RANDOM Statement

RANDOM *random-effect* ∼ *distribution* **SUBJECT=***variable* < *options* > **;**

The RANDOM statement defines a single random effect and its prior distribution or an array of random effects and their prior distribution. The *random-effect* must be represented by either a symbol or an array. The RANDOM statement must consist of the *random-effect*, a tilde (∼), the distribution for the random effect, and then a SUBJECT= variable.

SUBJECT=*variable* | **_OBS_**
identifies the subjects in the random-effects model. The *variable* must be part of the input data set, and it can be either a numeric variable or character literal. The *variable* does not need to be sorted, and the input data set does not need to be clustered according to it. SUBJECT=_OBS_ enables you fit an observation-level random-effects model (each observation has its own random effect) without specifying a subject variable in the input data set.

The random-effects parameters associated with each subject in the same RANDOM statement are assumed to be conditionally independent of each other, given other parameters and data set variables in the model. The other parameters include model parameters (declared in the PARMS statements), random-effects parameters (from other RANDOM statements), and missing data variables.

Table 80.4 shows the *distributions* that you can specify in the RANDOM statement.

Table 80.4 Valid Distributions in the RANDOM Statement

Distribution Name	Definition
beta(< **a=** >α, < **b=** >β)	Beta distribution with shape parameters α and β
binary(< **prob|p=** > p)	Binary (Bernoulli) distribution with probability of success p. You can use the alias **bern** for this distribution.
gamma(< **shape|sp=** > a, **scale|s=** λ) **gamma**(< **shape|sp=** > a, **iscale|is=** λ)	Gamma distribution with shape a and scale or inverse-scale λ

Table 80.4 *continued*

Distribution Name	Definition
dgeneral(*ll*)	General log-prior function that you construct using SAS programming statements for univariate or multivariate discrete random effects. See the section "Specifying a New Distribution" on page 6305 for more details.
general(*ll*)	General log-prior function that you construct using SAS programming statements for univariate or multivariate continuous random effects. See the section "Specifying a New Distribution" on page 6305 for more details.
igamma(< **shape|sp=** > *a*, **scale|s=** λ) **igamma**(< **shape|sp=** > *a*, **iscale|is=** λ)	Inverse gamma distribution with shape a and scale or inverse-scale λ
laplace(< **location|loc|l=** > θ, **scale|s=** λ) **laplace**(< **location|loc|l=** > θ, **iscale|is=** λ)	Laplace distribution with location θ and scale or inverse-scale λ. This is also known as the *double exponential* distribution. You can use the alias **dexpon** for this distribution.
normal(< **mean|m=** > μ, **sd=** λ) **normal**(< **mean|m=** > μ, **var|v=** λ) **normal**(< **mean|m=** > μ, **prec=** λ)	Normal (Gaussian) distribution with mean μ and a value of λ for the standard deviation, variance, or precision. You can use the aliases **gaussian**, **norm**, or **n** for this distribution.
poisson(< **mean|m=** > λ)	Poisson distribution with mean λ
table(< **p=** > p)	Table (categorical) distribution with probability vector p. You can also use the alias **cat** for this distribution
uniform(< **left|l=** > a, < **right|r=** > b)	Uniform distribution with range a and b. You can use the alias **unif** for this distribution.
MVN(< **mu=** >$\boldsymbol{\mu}$, < **cov=** >$\boldsymbol{\Sigma}$)	Multivariate normal distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
MVNAR(< **mu=** >μ, **sd=** λ, < **rho=** >ρ) **MVNAR**(< **mu=** >μ, **var=** λ, < **rho=** >ρ) **MVNAR**(< **mu=** >μ, **prec=** λ, < **rho=** >ρ)	Multivariate normal distribution with mean vector μ and a covariance matrix Σ. The covariance matrix Σ is a multiple of the scale and a matrix with a first-order autoregressive structure

Table 80.4 *continued*

Distribution Name	Definition
normalcar(**neighbors=**, **num=**, <**sd=**>λ) **normalcar**(**neighbors=**, **num=**, <**var=**>λ) **normalcar**(**neighbors=**, **num=**, <**prec=**>λ)	Intrinsic Gaussian conditional autoregressive (CAR) distribution. The NUM= option specifies the name of the data set variable that contains the number of neighbors for each subject; the NEIGHBORS= option specifies the prefix of data set variables that contain the neighboring indices of each subject; λ is the standard deviation, variance, or precision of the distribution.

The following RANDOM statement specifies a scale effect, where s2u can be a constant or a model parameter and index is a data set variable that indicates group membership of the random effect u:

```
random u ~ normal(0,var=s2u) subject=index;
```

The following statements specify multidimensional effects, where mu and cov can be either parameters in the model or constant arrays:

```
array w[2];
array mu[2];
array cov[2,2];
random w ~ mvn(mu, cov) subject=index;
```

You can specify multiple RANDOM statements. Hyperparameters in the prior distribution of a random effect can be other random effects in the model. For example, the following statements are allowed because the random effect g appears in the distribution for the random effect u:

```
random g ~ normal(0,var=s2g) subject=month;
random u ~ normal(g,var=s2u) subject=day;
```

These two RANDOM statements specify a nested hierarchical model in which the random effect g is the hyperparameter of the random effect u. You can build the hierarchical structure as deep as you want. You can also use multiple RANDOM statements to build non-nested random-effects models.

The number of random-effects parameters in each RANDOM statement is determined by the number of unique values in the SUBJECT= variable, which can be either unsorted numeric or unsorted character literal. Unlike the model parameters that are explicitly declared in the PARMS statement (with therefore a fixed total number), the number of random-effects parameters in a program depends on the values of the SUBJECT= data set variable. That number can change from one BY group to another.

The order of the RANDOM statements, or their relative placement with respect to other statements in the program (such as the PRIOR statement or the MODEL statement), is not important. The programming order becomes relevant if any hyperparameters are defined variables in the program. For example, in the following statements, the hyperparameter s is defined as a function of some variable or parameter in the model:

```
s = sqrt(s2g);
random g ~ normal(0,sd=s) subject=month;
```

That definition of s must appear before the RANDOM statement that requires it. If you switched the order of the statements as follows, PROC MCMC would not be able to calculate the prior density for some subjects correctly and would produce erroneous results.

```
random g ~ normal(0,sd=s) subject=month;
s = sqrt(s2g);
```

The names of the random-effects parameters are created internally. See the NAMESUFFIX= option for the naming convention of the random-effects parameters. The random-effects parameters are updated conditionally in the simulation. All posterior draws are saved to the OUTPOST= output data set by default, and you can use the MONITOR= option to monitor any of the parameters. For more information about available sampling algorithms, see the ALGORITHM= option. For more information about how to set a random-effects parameter to a constant (also known as corner-point constraint), see the CONSTRAINT option.

You can specify the following *options* in the RANDOM statement:

ALGORITHM=*option*

ALG=*option*

specifies the algorithm to use to sample the posterior distribution. The following *options* are available:

RWM

uses the random-walk Metropolis algorithm with normal proposal.

SLICE

uses the slice sampling algorithm.

GEO

uses the discrete random-walk Metropolis with symmetric geometric proposal.

When possible, PROC MCMC samples directly from the full conditional distribution. Otherwise, the default sampling algorithm is the RWM.

CENTER | NOCENTER

specifies whether to re-center the random-effects parameters after each draw. This option applies only when you use the NORMALCAR prior. The default is CENTER.

CONSTRAINT(VALUE=*value***) = FIRST | LAST | NONE | '***formatted-value***'**

ZERO=FIRST | LAST | NONE | '*formatted-value***'**

sets one of the random-effects parameters to a fixed value. The default is ZERO=NONE, which does not fix any of the parameters to be a constant. This option enables you to eliminate one of the parameters.

For example, this option could be useful if you want to fit a regression model with categorical covariates and, instead of creating a design matrix, you treat the parameters as "random effects" and fit an equivalent random-effects model.

Suppose you have a regression that includes a categorical variable X with J levels. You can construct a full-rank design matrix with J–1 dummy variables ($X_2 \cdots X_J$ with X_1 being the base group) and fit a

regression such as the following:

$$\mu_i = \beta_0 + \beta_2 \cdot X_2 \cdots \beta_J \cdot X_J$$

The following statements in a PROC MCMC step fit such a hypothetical regression model:

```
parms beta0 betax2 ... betaxJ;
prior beta: ~ n(0, sd=100);
mu = beta0 + betax2 * x2 + ... betaxJ * xJ;
...
```

Equivalently, you can also treat this model as a random-effects model such as the following, where β_j are random effects for each category in X:

$$\mu_i = \beta_0 + \beta_j \text{ for } j = 1, \ldots, J$$

However, this random-effects model is over-parameterized. The ZERO= option rids the model with one random-effects parameter of choice and fixes it to be zero. The following example statements fit such a hypothetical random-effects model:

```
parms beta0;
prior beta0 ~ n(0, sd=100);
random beta ~ n(0, sd=100) subject=x zero=first;
mu = beta0 + beta;
...
```

The specification ZERO=FIRST sets the first random-effects parameter to 0, implying $\beta_1 = 0$. This random-effects parameter corresponds to the first category in the SUBJECT= variable. The category is what the first observation of the SUBJECT= variable takes.

The specification ZERO=LAST sets the last random-effects parameter to be 0, implying $\beta_J = 0$. This random-effects parameter corresponds to the last category in the SUBJECT= variable. The category is not necessarily the same category that the last observation of the SUBJECT= variable takes because the SUBJECT= variable does not need to be sorted.

The specification ZERO='*formatted-value*' sets the random-effects parameter for the category (in the SUBJECT= variable) with a formatted value that matches '*formatted-value*' to 0. For example, ZERO='3' sets $\beta_3 = 0$.

The CONSTRAINT(VALUE=*value*) option works similarly to the ZERO= option. You can assign an arbitrary *value* to any one of the random-effects parameter. For example, the specification CONSTRAINT(VALUE=0)=FIRST is equivalent to ZERO=FIRST.

ICOND=*variable-list* | *numeric-list*

ISTATES=*variable-list* | *numeric-list*

specifies the initial conditions (or initial states) of the lag or lead variable of the random effect when the subject indices are out of the range of the subjects. (For more information about rules of constructing lag and lead variables in PROC MCMC, see the section "Access Lag and Lead Variables" on page 6320.) This works similarly to the ICOND= option in the MODEL statement, except that the index is done

according to a subject variable, not observations. The initial conditions can be model parameters, functions of model parameters, or constants. By default, *numeric-list* is set to 0.

The ICOND= option in a RANDOM statement sets the initial conditions for all lag or lead variables (of the associated random effect) that appear in the program, not just those that appear in the RANDOM statement. Suppose you have a maximum L number of lag variables and a maximum M number of lead variables of the random effect mu in the program, and there are n clusters. The program has the following vector of variables that need to be resolved during simulation:

$$\mu_{L+1}, \ldots, \mu_0, \mu_1, \ldots, \mu_n, \mu_{n+1}, \ldots, \mu_{n+M}$$

Of these variables, n ($\mu_1, \ldots, \mu_n$) are random-effects parameters and the remaining $L+M$ are initial conditions that are specified in the ICOND= option. The *variable-list* (or the *number-list*) should be a vector of length $L+M$, which can be greater than the number of lag/lead random-effect variables that appear in a program. If you provide a partial list that contains fewer than $L+M$ states, PROC MCMC fills the remaining vector with the value of 0.

INITIAL=*SAS-data-set* | *constant* | *numeric-list*

specifies the initial values of the random-effects parameters. By default, PROC MCMC uses the same option as specified in the INIT= option to generate initial values for the random-effects parameter: either it uses the mode of the prior density or it randomly draws a sample from that distribution. You can start the Markov chain at different places by providing a *SAS-data-set*, a constant, or a *numeric-list* for multivariate random-effects parameters.

If you use a *SAS-data-set*, the data set must consist of variable names that agree with the random-effects parameters in the model (see the NAMESUFFIX= option for the naming convention of the random-effects parameters). The easiest way to find the names of the internally created parameter names is to run a default analysis with a very small number of simulations and check the variable names in the OUTPOST= data set. You can provide a subset of the initial values in the *SAS-data-set* and PROC MCMC will use the default mechanism to fill in the rest of the random-effects parameters.

For example, the following statement creates a data set with initial values for the random-effects parameters u_1, u_2, and u_3:

```
data RandomInit;
   input u_1 u_2 u_3;
   datalines;
2.3 3 -3
;
```

The following RANDOM statement takes the values in the RandomInit data set to be the initial values of the corresponding random-effects parameters in the model:

```
random u ~ normal(0,var=s2u) subject=index init=randominit;
```

Specifying a *constant* assigns that constant as the initial value to all random-effects parameters in the statement. For example, the following statement assigns the value 5 to be used as an initial value for all u_i in the model:

```
random u ~ normal(0,var=s2u) subject=index init=5;
```

If you have multiple effects, you can provide a list of numbers, where the length of the list the same as the dimension of your random-effects array. Each number is then given to all corresponding random-effects parameters in order. For example, the following statement assigns the value 2 to be used as an initial value for all $w1_i$ and the value 3 to be used for all $w2_i$ in the model:

```
array w[2] w1 w2;
random w ~ mvn(mu, cov) subject=index init=(2 3);
```

If you use the GENERAL or DGENERAL functions in the RANDOM statement, you must provide initial values for these parameters.

MONITOR= (*symbol-list* | *number-list* | RANDOM(*number*))

outputs analysis for selected random-effects parameters. You can choose to monitor the random-effects parameters by listing the effect names or effect indices, or you can have them randomly selected by PROC MCMC.

To monitor all random-effects parameters, you specify the effect name in the MONITOR= option:

```
random u ~ normal(0,var=s2u) subject=index monitor=(u);
```

You have three options for monitoring a subset of the random-effects parameters. You can provide a list of the parameter names, you can provide a number list of the parameter indices, or you can have PROC MCMC randomly choose a subset of parameters for you.

For example, if you want to monitor analysis for parameters u_1 through u_10, u_23, and u_57, you can provide the names as follows:

```
random u ~ normal(0,var=s2u) subject=index monitor=(u_1-u_10 u_23 u_57);
```

The naming convention in the *symbol-list* must agree with the NAMESUFFIX= option, which controls how the parameter names of the *random-effect* are created. By default, NAMESUFFIX=SUBJECT, and the *symbol-list* must use suffixes that correspond to the formatted values in the SUBJECT= data set variable. With the NAMESUFFIX=POSITION option, the *symbol-list* must use suffixes that agree with the input order of the SUBJECT= variable. If the SUBJECT= variable has a character value, you cannot use the hyphen (-) in the *symbol-list* to indicate a range of variables.

To monitor the same list of random-effects parameters, you can provide their indices:

```
random u ~ normal(0,var=s2u) subject=index monitor=(1 to 10 by 1 23 57);
```

PROC MCMC can also randomly choose a subset of the parameters to monitor:

```
random u ~ normal(0,var=s2u) subject=index monitor=(random(12));
```

The sequence of the random indices is controlled by the SEED= option in the PROC MCMC statement.

By default, PROC MCMC does not monitor any random-effects parameters. When you specify this option, it takes the specification of the STATISTICS= and PLOTS= options in the PROC MCMC statement. By default, PROC MCMC outputs all the posterior samples of all random-effects parameters to the OUTPOST= output data set. You can use the NOOUTPOST option to suppress the saving of the random-effects parameters.

NAMESUFFIX=*option*

specifies how the names of the random-effects parameters are internally created from the SUBJECT= *variable* that is specified in the RANDOM statement. PROC MCMC creates the names by concatenating the *random-effect* symbol with an underscore and a series of numbers or characters. The following *options* control the type of methods that are used in such construction:

SUBJECT

constructs the parameter names by appending the formatted values of the SUBJECT= *variable* in the input data set.

POSITION

constructs the parameter names by appending the numbers 1, 2, 3, and so on, where the number indicates the order in which the SUBJECT= *variable* appears in the data set.

For example, suppose you have an input data set with four observations and the SUBJECT= variable zipcode has four values (with three of them unique): 27513, 01440, 27513, and 15217. The following SAS statement creates three random-effects parameters named u_27513, u_01440, and u_15217:

```
random u ~ normal(0,var=10) subject=zipcode namesuffix=subject;
```

On the other hand, using NAMESUFFIX=POSITION creates three parameters named u_1, u_2, and u_3:

```
random u ~ normal(0,var=10) subject=zipcode namesuffix=position;
```

By default, NAMESUFFIX=SUBJECT.

NOOUTPOST

suppresses the output of the posterior samples of random-effects parameters to the OUTPOST= data set. In models with a large number of random-effects parameters (for example, tens of thousands), PROC MCMC can run faster if it does not save the posterior samples of the random-effects parameters.

When you specify both the NOOUTPOST option and the MONITOR= option, PROC MCMC outputs the list of variables that are monitored.

The maximum number of variables that can be saved to an OUTPOST= data set is 32,767. If you run a large-scale random-effects model with the number of parameters exceeding the limit, the NOOUTPOST option is evoked automatically and PROC MCMC does not save the random-effects parameter draws to the posterior output data set. You can use the MONITOR= option to select a subset of the parameters to store in the OUTPOST= data set.

UDS Statement

UDS *subroutine-name* **(***subroutine-argument-list***) ;**

UDS stands for user defined sampler. The UDS statement enables you to use a separate algorithm, other than the default random walk Metropolis, to update parameters in the model. The purpose of the UDS statement is to give you a greater amount of flexibility and better control over the updating schemes of the Markov chain. Multiple UDS statements are allowed.

For the UDS statement to work properly, you have to do the following:

- write a subroutine by using PROC FCMP (see the FCMP Procedure in the *Base SAS Procedures Guide*) and save it to a SAS catalog (see the example in this section). The subroutine must update some parameters in the model. These are the UDS parameters. The subroutine is called the UDS subroutine.
- declare any UDS parameters in the PARMS statement with a sampling option, as in </ UDS> (see the section "PARMS Statement" on page 6263).
- specify the prior distributions for all UDS parameters, using the PRIOR statements.

NOTE: All UDS parameters must appear in three places: the UDS statement, the PARMS statement, and the PRIOR statement. Otherwise, PROC MCMC exits.

To obtain a valid Markov chain, a UDS subroutine must update a parameter from its full posterior conditional distribution and not the posterior marginal distribution. The posterior conditional is something that you need to provide. This conditional is implicitly based on a prior distribution. PROC MCMC has no means to verify that the implied prior in the UDS subroutine is the same as the prior that you specified in the PRIOR statement. You need to make sure that the two distributions agree; otherwise, you will get misleading results.

The priors in the PRIOR statements do not directly affect the sampling of the UDS parameters. They could affect the sampling of the other parameters in the model, which, in turn, changes the behavior of the Markov chain. You can see this by noting cases where the hyperparameters of the UDS parameters are model parameters; the priors should be part of the posterior conditional distributions of these hyperparameters, and they cannot be omitted.

Some additional information is listed to help you better understand the UDS statement:

- Most features of the SAS programming language can be used in subroutines processed by PROC FCMP (see the FCMP Procedure in the *Base SAS Procedures Guide*).
- The UDS statement does not support FCMP functions—a FCMP function returns a value, while a subroutine does not. A subroutine updates some of its subroutine arguments. These arguments are called OUTARGS arguments.
- The UDS parameters cannot be in the same block as other parameters. The optional argument </ UDS> in the PARMS statement prevents parameters that use the default Metropolis from being mixed with those that are updated by the UDS subroutines.
- You can put all the UDS parameters in the same PARMS statement or have a separate UDS statement for each of them.

- The same subroutine can be used in multiple UDS statements. This feature comes in handy if you have a generic sampler that can be applied to different parameters.
- PROC MCMC updates the UDS parameters by calling the UDS subroutines directly. At every iteration, PROC MCMC first samples parameters that use the Metropolis algorithm, then the UDS parameters. Sampling of the UDS parameters proceeds in the order in which the UDS statements are listed.
- A UDS subroutine accepts any symbols in the program as well as any input data set variables as its arguments.
- Only the OUTARGS arguments in a UDS subroutine are updated in PROC MCMC. You can modify other arguments in the subroutine, but the changes are not global in PROC MCMC.
- If a UDS subroutine has an argument that is a SAS data set variable, PROC MCMC steps through the data set while updating the UDS parameters. The subroutine is called once per observation in the data set for every iteration.
- If a UDS subroutine does not have any arguments that are data set variables, PROC MCMC does not access the data set while executing the subroutine. The subroutine is called once per iteration.
- To reduce the overhead in calling the UDS subroutine and accessing the data set repeatedly, you might consider reading all the input data set variables into arrays and using the arrays as the subroutine arguments. See the section "BEGINCNST/ENDCNST Statement" on page 6248 about how to use the BEGINCNST and ENDCNST statements to store data set variables.

For an example that uses the UDS statement, see "Example 80.19: Implement a New Sampling Algorithm" on page 6478.

Details: MCMC Procedure

How PROC MCMC Works

PROC MCMC is a simulation-based procedure that applies a variety of sampling algorithms to the program at hand. The default sampling methods include conjugate sampling (from full conditional), direct sampling from the marginal distribution, inverse cumulative distribution function, random walk Metropolis with normal proposal, and discretized random walk Metropolis with normal proposal. You can request alternate sampling algorithms, such as random walk Metropolis with t distribution proposal, discretized random walk Metropolis with symmetric geometric proposal, and the slice sampling algorithm.

PROC MCMC applies the more efficient sampling algorithms first, whenever possible. When a parameter does not appear in the conditional distributions of other random variables in the program, PROC MCMC generates samples directly from its prior distribution (which is also its marginal distribution). This usually occurs in data-independent Monte Carlo simulation programs (see "Example 80.1: Simulating Samples From a Known Density" on page 6376 for an example) or missing data problems, where the missing response variables are generated directly from the conditional sampling distribution (or the conditional likelihood). When conjugacy is detected, PROC MCMC uses random number generators to draw values from the full conditional distribution. (For information about detecting conjugacy, see the section "Conjugate Sampling"

on page 6286.) In other situations, PROC MCMC resorts to the random walk Metropolis with normal proposal to generate posterior samples for continuous parameters and a discretized version for discrete parameters. See the section "Metropolis and Metropolis-Hastings Algorithms" on page 156 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for details about the Metropolis algorithm. For the actual implementation details of the Metropolis algorithm in PROC MCMC, such as tuning of the covariance matrices, see the section "Tuning the Proposal Distribution" on page 6283.

A key component of the Metropolis algorithm is the calculation of the objective function. In most cases, the objective function that PROC MCMC uses in a Metropolis step is the logarithm of the joint posterior distribution, which is calculated with the inclusion of all data and parameters. The rest of this section describes how PROC MCMC calculates the objective function for parameters that use the Metropolis algorithm.

Model Parameters

To calculate the log of the posterior density, PROC MCMC assumes that all observations in the data set are independent,

$$\log(p(\theta|\mathbf{y})) = \log(\pi(\theta)) + \sum_{i=1}^{n} \log(f(y_i|\theta))$$

where θ is a parameter or a vector of parameters that are defined in the PARMS statements (referred to as the *model parameters*). The term $\log(\pi(\theta))$ is the sum of the log of the prior densities specified in the PRIOR and HYPERPRIOR statements. The term $\log(f(y_i|\theta))$ is the log likelihood specified in the MODEL statement. The MODEL statement specifies the log likelihood for a single observation in the data set.

If you want to model dependent data—that is, $\log(f(\mathbf{y}|\theta)) \neq \sum_i \log(f(y_i|\theta))$—you can use the JOINT-MODEL option in the PROC MCMC statement. See the section "Modeling Joint Likelihood" on page 6319 for more details.

The statements in PROC MCMC are similar to DATA step statements; PROC MCMC evaluates every statement in order for each observation. At the beginning of the data set, the log likelihood is set to be 0. As PROC MCMC steps through the data set, it cumulatively adds the log likelihood for each observation. Statements between the BEGINNODATA and ENDNODATA statements are evaluated only at the first and the last observations. At the last observation, the log of the prior and hyperprior distributions is added to the sum of the log likelihood to obtain the log of the posterior distribution.

Calculation of the $\log(p(\theta|\mathbf{y}))$ objective function involves a complete pass through the data set, making it potentially computationally expensive. If $\theta = \{\theta_1, \theta_2\}$ is multidimensional, you can choose to update a portion of the parameters at each iteration step by declaring them in separate PARMS statements (see the section "Blocking of Parameters" on page 6280 for more information). PROC MCMC updates each block of parameters while holding others constant. The objective functions that are used in each update are the same as the log of the joint posterior density:

$$\log(p(\theta_1|\mathbf{y}, \theta_2)) = \log(p(\theta_2|\mathbf{y}, \theta_1)) = \log(p(\theta|\mathbf{y}))$$

In other words, PROC MCMC does not derive the conditional distribution explicitly for each block of parameters, and it uses the full joint distribution in the Metropolis step for every block update.

Random-Effects Models

For programs that require RANDOM statements, PROC MCMC includes the sum of the density evaluation of the random-effects parameters in the calculation of the objective function for θ,

$$\log(p(\theta|\boldsymbol{\gamma},\mathbf{y})) = \log(\pi(\theta)) + \sum_{j=1}^{J}\log(\pi(\gamma_j|\theta)) + \sum_{i=1}^{n}\log(f(y_i|\theta,\boldsymbol{\gamma}))$$

where $\boldsymbol{\gamma} = \{\gamma_1,\ldots,\gamma_J\}$ are random-effects parameters and $\pi(\gamma_j|\theta)$ is the prior distribution of the random-effects parameters. The likelihood function can be conditional on $\boldsymbol{\gamma}$, but the prior distributions of θ, which must be independent of $\boldsymbol{\gamma}$, cannot.

The objective function used in the Metropolis step for the random-effects parameter γ_j contains only the portion of the data that belong to the *j*th cluster:

$$\log(p(\gamma_j|\theta,\mathbf{y})) = \log(\pi(\gamma_j|\theta)) + \sum_{i\in\{j\text{th cluster}\}}\log(f(y_i|\theta,\gamma_j))$$

The calculation does not include $\log(\theta)$, the prior density piece, because that is a known constant. Evaluation of this objective function involves only a portion of the data set, making it more computationally efficient. In fact, updating every random-effects parameters in a single RANDOM statement involves only one pass through the data set.

You can have multiple RANDOM statements in a program, which adds more pieces to the posterior calculation, such as

$$\log(p(\theta|\boldsymbol{\gamma},\boldsymbol{\alpha},\mathbf{y})) = \log(\pi(\theta)) + \sum_{j=1}^{J}\log(\pi(\gamma_j|\theta)) + \sum_{k=1}^{K}\log(\pi(\alpha_k|\theta)) + \sum_{i=1}^{n}\log(f(y_i|\theta,\boldsymbol{\gamma},\boldsymbol{\alpha}))$$

where $\boldsymbol{\alpha} = \{\alpha_1,\ldots,\alpha_K\}$ is another random effect. The random effects $\boldsymbol{\gamma}$ and $\boldsymbol{\alpha}$ can form their own hierarchy (as in a nested model), or they can enter the program in a non-nested fashion. The objective functions for γ_j and α_k are calculated using only observations that belong to their respective clusters.

Models with Missing Values

Missing values in the response variables of the MODEL statement are treated as random variables, and they add another layer in the conditional updates in the simulation. Suppose that

$$\mathbf{y} = \{\mathbf{y}_{\mathbf{obs}},\mathbf{y}_{\mathbf{mis}}\}$$

The response variable $\mathbf{y}$ consists of n_1 observed values $\mathbf{y}_{\mathbf{obs}}$ and n_2 missing values $\mathbf{y}_{\mathbf{mis}}$. The log of the posterior distribution is thus formed by

$$\log(p(\theta|\boldsymbol{\gamma},\mathbf{y}_{\mathbf{mis}},\mathbf{y}_{\mathbf{obs}})) = \log(\pi(\theta)) + \sum_{j=1}^{J}\log(\pi(\gamma_j|\theta)) + \sum_{i=1}^{n_2}\log(f(y_{\mathbf{mis},i}|\theta,\boldsymbol{\gamma}))\sum_{i=1}^{n_1}\log(f(y_{\mathbf{obs},i}|\theta,\boldsymbol{\gamma}))$$

where the expression is evaluated at the drawn $\boldsymbol{\gamma}$ and $\mathbf{y}_{\mathbf{obs}}$ values.

The conditional distribution of the random-effects parameter γ_j is

$$\log(p(\gamma_j|\theta,\mathbf{y})) = \log(\pi(\gamma_j|\theta)) + \sum_{i\in\{j\text{th cluster}\}}\log(f(y_i|\theta,\gamma_j))$$

where the y_i are either the observed or the imputed values of the response variable.

The missing values are usually sampled directly from the sampling distribution and do not require the Metropolis sampler. When a response variable takes on a GENERAL function, the objective function is simply the likelihood function: $\log\big(f(y_{\mathbf{mis},i}|\theta,\gamma_j)\big)$.

Blocking of Parameters

In a multivariate parameter model, if all k parameters are proposed with one joint distribution $q(\cdot|\cdot)$, acceptance or rejection would occur for all of them. This can be rather inefficient, especially when parameters have vastly different scales. A way to avoid this difficulty is to allocate the k parameters into d blocks and update them separately. The PARMS statement puts model parameters in separate blocks, and each block of parameters is updated sequentially in the procedure.

Suppose you want to sample from a multivariate distribution with probability density function $p(\theta|\mathbf{y})$ where $\theta = \{\theta_1, \theta_2, \ldots, \theta_k\}$. Now suppose that these k parameters are separated into d blocks—for example, $p(\theta|\mathbf{x}) = f_d(z)$ where $z = \{z_1, z_2, \ldots, z_d\}$, where each z_j contains a nonempty subset of the $\{\theta_i\}$, and where each θ_i is contained in one and only one z_j. In the MCMC context, the z's are blocks of parameters. In the blocked algorithm, a proposal consists of several parts. Instead of proposing a simultaneous move for all the θ's, a proposal is made for the θ_i's in z_1 only, then for the θ_i's in z_2, and so on for d subproposals. Any accepted proposal can involve any number of the blocks moving. The parameters do not necessarily all move at once as in the all-at-once Metropolis algorithm.

Formally, the blocked Metropolis algorithm is as follows. Let w_j be the collection of θ_i that are in block z_j, and let $q_j(\cdot|w_j)$ be a symmetric multivariate distribution that is centered at the current values of w_j.

1. Let $t = 0$. Choose points for all w_j^t. A point can be an arbitrary point as long as $p(w_j^t|\mathbf{y}) > 0$.
2. For $j = 1, \ldots, d$:
   a) Generate a new sample, $w_{j,\text{new}}$, using the proposal distribution $q_j(\cdot|w_j^t)$.
   b) Calculate the following quantity:
   $$r = \min\left\{\frac{p(w_{j,\text{new}}|w_1^t, \ldots, w_{j-1}^t, w_{j+1}^{t-1}, \ldots, w_d^{t-1}, \mathbf{y})}{p(w_j^t|w_1^t, \ldots, w_{j-1}^t, w_{j+1}^{t-1}, \ldots, w_d^{t-1}, \mathbf{y})}, 1\right\}.$$
   c) Sample u from the uniform distribution $U(0, 1)$.
   d) Set $w_j^{t+1} = w_{j,\text{new}}$ if $r < u$; $w_j^{t+1} = w_j^t$ otherwise.
3. Set $t = t + 1$. If $t < T$, the number of desired samples, go back to Step 2; otherwise, stop.

With PROC MCMC, you can sample all parameters simultaneously by putting them all in a single PARMS statement, you can sample parameters individually by putting each parameter in its own PARMS statement, or you can sample certain subsets of parameters together by grouping each subset in its own PARMS statements. For example, if the model you are interested in has five parameters, alpha, beta, gamma, phi, sigma, the all-at-once strategy is as follows:

```
parms alpha beta gamma phi sigma;
```

The one-at-a-time strategy is as follows:

```
parms alpha;
parms beta;
parms gamma;
parms phi;
parms sigma;
```

A two-block strategy could be as follows:

```
parms alpha beta gamma;
parms phi sigma;
```

The exceptions to the previously described blocking strategies are parameters that are sampled directly (either from their full conditional or marginal distributions) and parameters that are array-based (with multivariate prior distributions). In these cases, the parameters are taken out of an existing block and are updated individually. You can use the sampling options in the PARMS statement to override the default behavior.

One of the greatest challenges in MCMC sampling is achieving good mixing of the chains—the chains should quickly traverse the support of the stationary distribution. A number of factors determine the behavior of a Metropolis sampler; blocking is one of them, so you want to be extremely careful when you choose a good design. Generally speaking, forming blocks of parameters has its advantages, but it is not true that the larger the block the faster the convergence.

When simultaneously sampling a large number of parameters, the algorithm might find it difficult to achieve good mixing. As the number of parameters gets large, it is much more likely to have (proposal) samples that fall well into the tails of the target distribution, producing too small a test ratio. As a result, few proposed values are accepted and convergence is slow. On the other hand, when the algorithm samples each parameter individually, the computational cost increases linearly. Each block of Metropolis parameters requires one additional pass through the data set, so a five-block updating strategy could take five times longer than a single-block updating strategy. In addition, there is a chance that the chain might mix far too slowly because the conditional distributions (of θ_i given all other θ's) might be very "narrow," as a result of posterior correlation among the parameters. When that happens, it takes a long time for the chain to fully explore that dimension alone. There are no theoretical results that can help determine an optimal "blocking" for an arbitrary parametric model. A rule followed in practice is to form small groups of correlated parameters that belong to the same context in the formulation of the model. The best mixing is usually obtained with a blocking strategy somewhere between the all-at-once and one-at-a-time strategies.

Sampling Methods

When suitable, PROC MCMC chooses the optimal sampling method for each parameter. That involves direct sampling either from the conditional posterior via conjugacy (see the section "Conjugate Sampling" on page 6286) or via the marginal posterior (see the section "Direct Sampling" on page 6286). Alternatively, PROC MCMC samples according to Table 80.5. Each block of parameters is classified by the nature of the prior distributions. "Continuous" means all priors of the parameters in the same block have a continuous distribution. "Discrete" means all priors are discrete. "Mixed" means that some parameters are continuous and others are discrete. Parameters that have binary priors are treated differently, as indicated in the table.

Table 80.5 Sampling Methods in PROC MCMC

Blocks	Default Method	Alternative Method
Continuous	Multivariate normal (MVN)	Multivariate t (MVT); slice sampler
Discrete (other than binary)	Binned MVN	Binned MVT or symmetric geometric
Mixed	MVN	MVT
Binary (single dimensional)	Inverse CDF	
Binary (multidimensional)	Independence sampler	

For a block of continuous parameters, PROC MCMC uses a multivariate normal distribution as the default proposal distribution. In the tuning phase, PROC MCMC finds an optimal scale c and a tuning covariance matrix Σ.

For a discrete block of parameters, PROC MCMC uses a discretized multivariate normal distribution as the default proposal distribution. The scale c and covariance matrix $\mathbf{\Sigma}$ are tuned. Alternatively, you can use an independent symmetric geometric proposal distribution. The density has form $\frac{p(1-p)^{|\theta|}}{2(1-p)}$ and has variance $\frac{(2-p)(1-p)}{p^2}$. In the tuning phase, the procedure finds an optimal proposal probability p for every parameter in the block.

You can change the proposal distribution, from the normal to a t distribution. You can either use the PROC option PROPDIST=T(df) or PARMS statement option < / T(df) > to make the change. The t distributions have thicker tails, and they can propose to the tail areas more efficiently than the normal distribution. It can help with the mixing of the Markov chain if some of the parameters have a skewed tails. See "Example 80.6: Nonlinear Poisson Regression Models" on page 6406. The independence sampler (see the section "Independence Sampler" on page 160 in Chapter 8, "Introduction to Bayesian Analysis Procedures") is used for a block of binary parameters. The inverse CDF method is used for a block that consists of a single binary parameter.

For parameters with continuous prior distributions, you can use the slice sampler as an alternative sampling algorithm. To do so, specify the SLICE option in the PARMS. When you specify the SLICE option, all parameters are updated individually. PROC MCMC does not support a multivariate version of the slice sampler. For more in information about the slice sampler, see the section"Slice Sampler" on page 159 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

The sampling algorithms for the random-effects parameters are chosen in a similar fashion. The preferred algorithms are the direct method either from the full conditional or the marginal. When these are not attainable, Metropolis with normal proposal becomes the default for continuous random-effects parameters,

and discrete Metropolis with normal proposal becomes the default for discrete random-effects parameters. You can use the ALGORITHM= option in the RANDOM statement to choose the slice sampler or discrete Metropolis with symmetric geometric as the alternatives.

The sampling preference of the missing data variables is the same as the random-effects parameters. The reserve sampling algorithm is the Metropolis. There is no alternative sampling method available for the missing data variables.

Tuning the Proposal Distribution

One key factor in achieving high efficiency of a Metropolis-based Markov chain is finding a good proposal distribution for each block of parameters. This process is referred to as tuning. The tuning phase consists of a number of loops. The minimum number of loops is controlled by the option MINTUNE=, with a default value of 2. The option MAXTUNE= controls the maximum number of tuning loops, with a default value of 24. Each loop lasts for NTU= iterations, where by default NTU= 500. At the end of every loop, PROC MCMC examines the acceptance probability for each block. The acceptance probability is the percentage of NTU= proposals that have been accepted. If the probability falls within the acceptance tolerance range (see the section "Scale Tuning" on page 6283), the current configuration of $c/\mathbf{\Sigma}$ or p is kept. Otherwise, these parameters are modified before the next tuning loop.

Continuous Distribution: Normal or *t* Distribution

A good proposal distribution should resemble the actual posterior distribution of the parameters. Large sample theory states that the posterior distribution of the parameters approaches a multivariate normal distribution (see Gelman et al. 2004, Appendix B, and Schervish 1995, Section 7.4). That is why a normal proposal distribution often works well in practice. The default proposal distribution in PROC MCMC is the normal distribution: $q_j(\theta_{\text{new}}|\theta^t) = \text{MVN}(\theta_{\text{new}}|\theta^t, c^2\mathbf{\Sigma})$. As an alternative, you can choose a multivariate t distribution as the proposal distribution. It is a good distribution to use if you think that the posterior distribution has thick tails and a t distribution can improve the mixing of the Markov chain. See "Example 80.6: Nonlinear Poisson Regression Models" on page 6406.

Scale Tuning

The acceptance rate is closely related to the sampling efficiency of a Metropolis chain. For a random walk Metropolis, high acceptance rate means that most new samples occur right around the current data point. Their frequent acceptance means that the Markov chain is moving rather slowly and not exploring the parameter space fully. On the other hand, a low acceptance rate means that the proposed samples are often rejected; hence the chain is not moving much. An efficient Metropolis sampler has an acceptance rate that is neither too high nor too low. The scale c in the proposal distribution $q(\cdot|\cdot)$ effectively controls this acceptance probability. Roberts, Gelman, and Gilks (1997) showed that if both the target and proposal densities are normal, the optimal acceptance probability for the Markov chain should be around 0.45 in a single dimensional problem, and asymptotically approaches 0.234 in higher dimensions. The corresponding optimal scale is 2.38, which is the initial scale set for each block.

Due to the nature of stochastic simulations, it is impossible to fine-tune a set of variables such that the Metropolis chain has the exact desired acceptance rate. In addition, Roberts and Rosenthal (2001) empirically demonstrated that an acceptance rate between 0.15 and 0.5 is at least 80% efficient, so there is really no need to fine-tune the algorithms to reach acceptance probability that is within small tolerance of the optimal values. PROC MCMC works with a probability range, determined by the PROC options TARGACCEPT

± ACCEPTTOL. The default value of TARGACCEPT is a function of the number of parameters in the model, as outlined in Roberts, Gelman, and Gilks (1997). The default value of ACCEPTTOL= is 0.075. If the observed acceptance rate in a given tuning loop is less than the lower bound of the range, the scale is reduced; if the observed acceptance rate is greater than the upper bound of the range, the scale is increased. During the tuning phase, a scale parameter in the normal distribution is adjusted as a function of the observed acceptance rate and the target acceptance rate. The following updating scheme is used in PROC MCMC [1]:

$$c_{\text{new}} = \frac{c_{\text{cur}} \cdot \Phi^{-1}(p_{\text{opt}}/2)}{\Phi^{-1}(p_{\text{cur}}/2)}$$

where c_{cur} is the current scale, p_{cur} is the current acceptance rate, p_{opt} is the optimal acceptance probability.

Covariance Tuning

To tune a covariance matrix, PROC MCMC takes a weighted average of the old proposal covariance matrix and the recent observed covariance matrix, based on NTU samples in the current loop. The TUNEWT=*w* option determines how much weight is put on the recently observed covariance matrix. The formula used to update the covariance matrix is as follows:

$$\text{COV}_{\text{new}} = w\,\text{COV}_{\text{cur}} + (1 - w)\text{COV}_{\text{old}}$$

There are two ways to initialize the covariance matrix:

- The default is an identity matrix multiplied by the initial scale of 2.38 (controlled by the PROC option SCALE=) and divided by the square root of the number of estimated parameters in the model. It can take a number of tuning phases before the proposal distribution is tuned to its optimal stage, since the Markov chain needs to spend time learning about the posterior covariance structure. If the posterior variances of your parameters vary by more than a few orders of magnitude, if the variances of your parameters are much different from 1, or if the posterior correlations are high, then the proposal tuning algorithm might have difficulty with forming an acceptable proposal distribution.
- Alternatively, you can use a numerical optimization routine, such as the quasi-Newton method, to find a starting covariance matrix. The optimization is performed on the joint posterior distribution, and the covariance matrix is a quadratic approximation at the posterior mode. In some cases this is a better and more efficient way of initializing the covariance matrix. However, there are cases, such as when the number of parameters is large, where the optimization could fail to find a matrix that is positive definite. In that case, the tuning covariance matrix is reset to the identity matrix.

A side product of the optimization routine is that it also finds the *maximum a posteriori* (MAP) estimates with respect to the posterior distribution. The MAP estimates are used as the initial values of the Markov chain.

If any of the parameters are discrete, then the optimization is performed conditional on these discrete parameters at their respective fixed initial values. On the other hand, if all parameters are continuous, you can in some cases skip the tuning phase (by setting MAXTUNE=0) or the burn-in phase (by setting NBI=0).

[1] Roberts, Gelman, and Gilks (1997) and Roberts and Rosenthal (2001) demonstrate that the relationship between acceptance probability and scale in a random walk Metropolis is $p = 2\Phi\left(-\sqrt{I}c/2\right)$, where c is the scale, p is the acceptance rate, Φ is the CDF of a standard normal, and $I \equiv E_f[(f'(x)/f(x))^2]$, $f(x)$ is the density function of samples. This relationship determines the updating scheme, with I being replaced by the identity matrix to simplify calculation.

Discrete Distribution: Symmetric Geometric

By default, PROC MCMC uses the normal density as the proposal distribution in all Metropolis random walks. For parameters that have discrete prior distributions, PROC MCMC discretizes proposed samples. You can choose an alternative symmetric geometric proposal distribution by specifying the option DISCRETE=GEO.

The density of the symmetric geometric proposal distribution is as follows:

$$\frac{p_g(1-p_g)^{|\theta|}}{2(1-p_g)}$$

where the symmetry centers at θ. The distribution has a variance of

$$\sigma^2 = \frac{(2-p_g)(1-p_g)}{p_g^2}$$

Tuning for the proposal p_g uses the following formula:

$$\frac{\sigma_{\text{new}}}{\sigma_{\text{cur}}} = \frac{\Phi^{-1}(p_{\text{opt}}/2)}{\Phi^{-1}(p_{\text{cur}}/2)}$$

where σ_{new} is the standard deviation of the new proposal geometric distribution, σ_{cur} is the standard deviation of the current proposal distribution, p_{opt} is the target acceptance probability, and p_{cur} is the current acceptance probability for the discrete parameter block.

The updated p_g is the solution to the following equation that is between 0 and 1 :

$$\sqrt{\frac{(2-p_g)(1-p_g)}{p_g^2}} = \frac{\sigma_{\text{cur}} \cdot \Phi^{-1}(p_{\text{opt}}/2)}{\Phi^{-1}(p_{\text{cur}}/2)}$$

Binary Distribution: Independence Sampler

Blocks consisting of a single parameter with a binary prior do not require any tuning; the inverse-CDF method applies. Blocks that consist of multiple parameters with binary prior are sampled by using an independence sampler with binary proposal distributions. See the section "Independence Sampler" on page 160 in Chapter 8, "Introduction to Bayesian Analysis Procedures." During the tuning phase, the success probability p of the proposal distribution is taken to be the probability of acceptance in the current loop. Ideally, an independence sampler works best if the acceptance rate is 100%, but that is rarely achieved. The algorithm stops when the probability of success exceeds the TARGACCEPTI=*value*, which has a default value of 0.6.

Direct Sampling

The word "direct" is reserved for sampling that is done directly from the prior distribution of a model or a random-effects parameter or from the sampling distribution of a missing data variable. If the parameter is updated via sampling from its full conditional posterior distribution, the sampling method is referred to as conjugate sampling. (See the section "Conjugate Sampling" on page 6286.)

Whenever a parameter does not appear in the hierarchy of another parameter in the model, PROC MCMC samples directly from its distribution. For a model parameter or a random-effects parameter, this distribution is its prior distribution. For a missing data variable, this distribution is the sampling distribution of the response variable. Therefore, direct sampling takes place most frequently in data-independent Monte Carlo simulations or the sampling of missing response variables.

Conjugate Sampling

Conjugate prior is a family of prior distributions in which the prior and the posterior distributions are of the same family of distributions. For example, if you model an independently and identically distributed random variable y_i using a normal likelihood with known variance σ^2,

$$y_i \sim \text{normal}(\mu, \sigma^2)$$

a normal prior on μ

$$\mu \sim \text{normal}(\mu_0, \sigma_0^2)$$

is a conjugate prior because the posterior distribution of μ is also a normal distribution given $\text{y} = \{y_i\}$, σ^2, μ_0, and σ_0^2 :

$$\mu|\text{y} \sim \text{normal}\left(\left(\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}\right)^{-1} \cdot \left(\frac{\mu_0}{\sigma_0^2} + \frac{n \cdot \bar{\text{y}}}{\sigma^2}\right), \left(\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}\right)^{-1}\right)$$

Conjugate sampling is efficient because it enables the Markov chain to obtain samples from the target distribution directly. When appropriate, PROC MCMC uses conjugate sampling methods to draw conditional posterior samples. Table 80.6 lists scenarios that lead to conjugate sampling in PROC MCMC.

Table 80.6 Conjugate Sampling in PROC MCMC

Family	Parameter	Prior
Normal with known μ	Variance σ^2	Inverse gamma family
Normal with known μ	Precision τ	Gamma family
Normal with known scale parameter (σ^2, σ, or τ)	Mean μ	Normal
Multivariate normal with known Σ	Mean $\boldsymbol{\mu}$	Multivariate normal
Multivariate normal with known $\boldsymbol{\mu}$	Covariance Σ	Inverse Wishart
Multinomial	$\mathbf{p}$	Dirichlet
Binomial/binary	p	Beta
Poisson	λ	Gamma family

In most cases, Family in Output 80.6 refers to the likelihood function. However, it does not necessarily have to be the case. The Family is a distribution that is conditional on the parameter of interest, and it can appear in any level of the hierarchical model, including on the random-effects level.

PROC MCMC can detect conjugacy only if the model parameter (not a function or a transformation of the model parameter) is used in the prior and Family distributions. For example, the following statements lead to a conjugate sampler being used on the parameter mu:

```
parm mu;
prior mu ~ n(0, sd=1000);
model y ~ n(mu, var=s2);
```

However, if you modify the program slightly in the following way, although the conjugacy still holds in theory, PROC MCMC cannot detect conjugacy on mu because the parameter enters the normal likelihood function through the symbol w:

```
parm mu;
prior mu ~ n(0, sd=1000);
w = mu;
model y ~ n(w, var=s2);
```

In this case, PROC MCMC resorts to the default sampling algorithm, which is a random walk Metropolis based on a normal kernel.

Similarly, the following statements also prevent PROC MCMC from detecting conjugacy on the parameter mu:

```
parm mu;
prior mu ~ n(0, sd=1000);
model y ~ n(mu + 2, var=s2);
```

In a normal family, an often-used and often-confused conjugate prior on the variance is the inverse gamma distribution, and a conjugate prior on the precision is the gamma distribution. See "Gamma and Inverse Gamma Distributions" on page 6346 for typical usages of these prior distributions.

When conjugacy is detected in a model, PROC MCMC performs a numerical optimization on the joint posterior distribution at the start of the MCMC simulation. If the only sampling methods required in the program are conjugate samplers or direct samplers, PROC MCMC omits this optimization step. To turn off this optimization routine, use the PROPCOV=IND option in the PROC MCMC statement.

Initial Values of the Markov Chains

There are three types of parameters in a PROC MCMC program: the *model parameters* in the PARMS statement, the *random-effects parameters* in the RANDOM statement, and the *missing data variables* in the MODEL statement. The last category is used to model missing values in the input data set.

When the model parameters and random-effects parameters have missing initial values, PROC MCMC generates initial values based on the prior distributions. PROC MCMC either uses the mode value (the default) or draws a random number (if the INIT=RANDOM option is specified). For distributions that do not have modes, such as the uniform distribution, PROC MCMC uses the mean instead. In general, PROC MCMC avoids using starting values that are close to the boundary of support of the prior distribution. For example, the exponential prior has a mode at 0, and PROC MCMC starts an initial value at the mean. This

avoids some potential numerical problems. If you use the GENERAL or DGENERAL function in the PRIOR statements, you must provide initial values for those parameters.

With missing data variables, PROC MCMC uses the sample average of the nonmissing values (of the response variable) as the initial value. If all values of a particular variable are missing, PROC MCMC resorts to using the mode value or a random number from the sampling distribution (the likelihood), depending on the specification of the INIT= option.

To assign a different set of initial values to the model parameters, you use either the PARMS statements or programming statements within the BEGINCNST and ENDCNST statements. See the section "Assignments of Parameters" on page 6288 for more information about how to assign parameter values within the BEGINCNST and ENDCNST statements.

To assign initial values to the random-effects parameters, you can use the INIT= option in the RANDOM statement. Either you can give a constant value to all random-effects parameters that are associated with that statement (for example, use **init=3**), or you can assign values individually by providing a data set that stores different values for different parameters.

A mirroring INIT= option in the MODEL statement enables you to assign different initial values to the missing data variables.

If you use the PROPCOV= optimization option in the PROC MCMC statement, PROC MCMC starts the tuning at the optimized values. PROC MCMC overwrites the initial values that you might have provided at the beginning of the Markov chain unless you use the option INIT=REINIT.

Assignments of Parameters

In general, you cannot alter the values of any model parameters in PROC MCMC. For example, the following assignment statement produces an error:

```
parms alpha;
alpha = 27;
```

This restriction prevents incorrect calculation of the posterior density—assignments of parameters in the program would override the parameter values generated by PROC MCMC and lead to an incorrect value of the density function.

However, you can modify parameter values and assign initial values to parameters within the block defined by the BEGINCNST and ENDCNST statements. The following syntax is allowed:

```
parms alpha;
begincnst;
   alpha = 27;
endcnst;
```

The initial value of alpha is 27. Assignments within the BEGINCNST/ENDCNST block override initial values specified in the PARMS statement. For example, with the following statements, the Markov chain starts at alpha = 27, not 23.

```
parms alpha 23;
begincnst;
   alpha = 27;
endcnst;
```

This feature enables you to systematically assign initial values. Suppose that z is an array parameter of the same length as the number of observations in the input data set. You want to start the Markov chain with each z_i having a different value depending on the data set variable y. The following statements set $z_i = |y|$ for the first half of the observations and $z_i = 2.3$ for the rest:

```
/* a rather artificial input data set. */
data inputdata;
   do ind = 1 to 10;
      y = rand('normal');
      output;
   end;
run;

proc mcmc data=inputdata;
   array z[10];
   begincnst;
      if ind <= 5 then z[ind] = abs(y);
      else z[ind] = 2.3;
   endcnst;
   parms z:;
   prior z: ~ normal(0, sd=1);
   model general(0);
run;
```

Elements of z are modified as PROC MCMC executes the programming statements between the BEGINCNST and ENDCNST statements. This feature could be useful when you use the GENERAL function and you find that the PARMS statements are too cumbersome for assigning starting values.

Standard Distributions

The section "Univariate Distributions" on page 6290 (Table 80.7 through Table 80.36) lists all univariate distributions that PROC MCMC recognizes. The section "Multivariate Distributions" on page 6301 (Table 80.37 through Table 80.41) lists all multivariate distributions that PROC MCMC recognizes. With the exception of the multinomial distribution, all these distributions can be used in the MODEL, PRIOR, and HYPERPRIOR statements. The multinomial distribution is supported only in the MODEL statement. The RANDOM statement supports a limited number of distributions; see Table 80.4 for the complete list.

See the section "Using Density Functions in the Programming Statements" on page 6306 for information about how to use distributions in the programming statements. To specify an arbitrary distribution, you can use the GENERAL and DGENERAL functions. See the section "Specifying a New Distribution" on page 6305 for more details. See the section "Truncation and Censoring" on page 6309 for tips about how to work with truncated distributions and censoring data.

Univariate Distributions

Table 80.7 Beta Distribution

PROC specification	**beta**(a, b)
Density	$\frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\theta^{a-1}(1-\theta)^{b-1}$
Parameter restriction	$a > 0, b > 0$
Range	$\begin{cases} [0,1] & \text{when } a = 1, b = 1 \\ [0,1) & \text{when } a = 1, b \neq 1 \\ (0,1] & \text{when } a \neq 1, b = 1 \\ (0,1) & \text{otherwise} \end{cases}$
Mean	$\frac{a}{a+b}$
Variance	$\frac{ab}{(a+b)^2(a+b+1)}$
Mode	$\begin{cases} \frac{a-1}{a+b-2} & a > 1, b > 1 \\ 0 \text{ and } 1 & a < 1, b < 1 \\ 0 & \begin{cases} a < 1, b \geq 1 \\ a = 1, b > 1 \end{cases} \\ 1 & \begin{cases} a \geq 1, b < 1 \\ a > 1, b = 1 \end{cases} \\ \text{does not exist uniquely} & a = b = 1 \end{cases}$
Random number	If $\min(a, b) > 1$, see (Cheng 1978); if $\max(a, b) < 1$, see (Atkinson and Whittaker 1976) and (Atkinson 1979); if $\min(a, b) < 1$ and $\max(a, b) > 1$, see (Cheng 1978); if $a = 1$ or $b = 1$, use the inversion method; if $a = b = 1$, use a uniform random number generator.

Table 80.8 Binary Distribution

PROC specification	**binary**(p)
Density	$p^{\theta}(1-p)^{1-\theta}$
Parameter restriction	$0 \leq p \leq 1$
Range	$\begin{cases} \{0\} & \text{when } p = 0 \\ \{1\} & \text{when } p = 1 \\ \{0,1\} & \text{otherwise} \end{cases}$
Mean	(p)
Variance	$p(1-p)$
Mode	$\begin{cases} \{1\} & \text{when } p = 1 \\ \{0\} & \text{otherwise} \end{cases}$
Random number	Generate $u \sim \text{uniform}(0, 1)$. If $u \leq p$, $\theta = 1$; else, $\theta = 0$.

Table 80.9 Binomial Distribution

PROC specification	**binomial**(n, p)
Density	$\binom{n}{\theta} p^{\theta}(1-p)^{n-\theta}$
Parameter restriction	$n = 0, 1, 2, \ldots, 0 \leq p \leq 1$
Range	$\theta \in \{0, \ldots, n\}$
Mean	np
Variance	$np(1-p)$
Mode	$\lfloor (n+1)p \rfloor$

Table 80.10 Cauchy Distribution

PROC specification	**cauchy**(a, b)
Density	$\frac{1}{\pi}\left(\frac{b}{b^2+(\theta-a)^2}\right)$
Parameter restriction	$b > 0$
Range	$\theta \in (-\infty, \infty)$
Mean	Does not exist.
Variance	Does not exist.
Mode	a
Random number	Generate $u_1, u_2 \sim$ uniform$(0, 1)$; let $v = 2u_2 - 1$. Repeat the procedure until $u_1^2 + v^2 < 1$. $y = v/u_1$ is a draw from the standard Cauchy, and $\theta = a + by$ (Ripley 1987).

Table 80.11 χ^2 Distribution

PROC specification	**chisq**(ν)
Density	$\frac{1}{\Gamma(\nu/2)2^{\nu/2}}\theta^{(\nu/2)-1}e^{-\theta/2}$
Parameter restriction	$\nu > 0$
Range	$\theta \in [0, \infty)$ if $\nu = 2$; $(0, \infty)$ otherwise.
Mean	ν
Variance	2ν
Mode	$\nu - 2$ if $\nu \geq 2$; does not exist otherwise.
Random number	χ^2 is a special case of the gamma distribution: $\theta \sim$ gamma$(\nu/2, \text{scale=2})$ is a draw from the χ^2 distribution.

Table 80.12 Exponential χ^2 Distribution

PROC specification	**expchisq**(ν)
Density	$\frac{1}{\Gamma(\nu/2)2^{\nu/2}} \exp(\theta)^{\nu/2} \exp(-\exp(\theta)/2)$
Parameter restriction	$\nu > 0$
Range	$\theta \in (-\infty, \infty)$
Mode	$\log(\nu)$
Random number	Generate $x_1 \sim \chi^2(\nu)$, and $\theta = \log(x_1)$ is a draw from the exponential χ^2 distribution.
Relationship to the χ^2 distribution	$\theta \sim \chi^2(\nu) \Leftrightarrow \log(\theta) \sim \exp \chi^2(\nu)$

Table 80.13 Exponential Exponential Distribution

PROC specification	**expexpon**(scale = b)	**expexpon**(iscale = β)
Density	$\frac{1}{b} \exp(\theta) \exp(-\exp(\theta)/b)$	$\beta \exp(\theta) \exp(-\exp(\theta) \cdot \beta)$
Parameter restriction	$b > 0$	$\beta > 0$
Range	$\theta \in (-\infty, \infty)$	Same
Mode	$\log(b)$	$\log(1/\beta)$
Random number	Generate $x_1 \sim$ expon(scale=b), and $\theta = \log(x_1)$ is a draw from the exponential exponential distribution. Note that an exponential exponential distribution is not the same as the double exponential distribution.	
Relationship to the exponential distribution	$\theta \sim \text{expon}(b) \Leftrightarrow \log(\theta) \sim \text{expExpon}(b)$	

Table 80.14 Exponential Gamma Distribution

PROC specification	**expgamma**(a, scale = b)	**expgamma**(a, iscale = β)
Density	$\frac{1}{b^a \Gamma(a)} e^{a\theta} \exp(-e^{\theta}/b)$	$\frac{\beta^a}{\Gamma(a)} e^{a\theta} \exp(-e^{\theta} \cdot \beta)$
Parameter restriction	$a > 0, b > 0$	$a > 0, \beta > 0$
Range	$\theta \in (-\infty, \infty)$	Same
Mode	$\log(ab)$	$\log(a/\beta)$
Random number	Generate $x_1 \sim \text{gamma}(a, \text{scale} = b)$, and $\theta = \log(x_1)$ is a draw from the exponential gamma distribution.	
Relationship to the Γ distribution	$\theta \sim \text{gamma}(a, b) \Leftrightarrow \log(\theta) \sim \text{expGamma}(a, b)$	

Table 80.15 Exponential Inverse χ^2 Distribution

PROC specification	**expichisq**(ν)
Density	$\frac{1}{\Gamma(\frac{\nu}{2})2^{\nu/2}} \exp(-\nu\theta/2) \exp(-1/(2\exp(\theta)))$
Parameter restriction	$\nu > 0$
Range	$\theta \in (-\infty, \infty)$
Mode	$-\log(\nu)$
Random number	Generate $x_1 \sim i\chi^2(\nu)$, and $\theta = \log(x_1)$ is a draw from the exponential inverse χ^2 distribution.
Relationship to the $i\chi^2$ distribution	$\theta \sim i\chi^2(\nu) \Leftrightarrow \log(\theta) \sim \exp i\chi^2(\nu)$

Table 80.16 Exponential Inverse Gamma Distribution

PROC specification	**expigamma**(a, scale = b)	**expigamma**(a, iscale = β)
Density	$\frac{b^a}{\Gamma(a)} \exp(-\alpha\theta) \exp(-b/\exp(\theta))$	$\frac{1}{\beta^\alpha \Gamma(a)} \exp(-\alpha\theta) \exp(-\frac{1}{\beta \exp(\theta)})$
Parameter restriction	$a > 0, b > 0$	$a > 0, \beta > 0$
Range	$\theta \in (-\infty, \infty)$	Same
Mode	$-\log(a/b)$	$-\log(a\beta)$
Random number	Generate $x_1 \sim \text{igamma}(a, \text{scale} = b)$, and $\theta = \log(x_1)$ is a draw from the exponential inverse gamma distribution.	
Relationship to the $i\Gamma$ distribution	$\theta \sim \text{igamma}(a, b) \Leftrightarrow \log(\theta) \sim \text{eigamma}(a, b)$	

Table 80.17 Exponential Scaled Inverse χ^2 Distribution

PROC specification	**expsichisq**(ν, s)
Density	$\frac{(\frac{\nu}{2})^{\nu/2}}{\Gamma(\frac{\nu}{2})} s^\nu \exp(-\nu\theta/2) \exp(-\nu s^2/(2\exp(\theta)))$
Parameter restriction	$\nu > 0, s > 0$
Range	$\theta \in (-\infty, \infty)$
Mode	$\log(s^2)$
Random number	Generate $x_1 \sim si\chi^2(\nu, s)$, and $\theta = \log(x_1)$ is a draw from the exponential scaled inverse χ^2 distribution.
Relationship to the $si\chi^2$ distribution	$\theta \sim si\chi^2(\nu, s) \Leftrightarrow \log(\theta) \sim \exp si\chi^2(\nu, s)$

Table 80.18 Exponential Distribution

PROC specification	**expon**(scale = b)	**expon**(iscale = β)
Density	$\frac{1}{b}e^{-\theta/b}$	$\beta e^{-\beta\theta}$
Parameter restriction	$b > 0$	$\beta > 0$
Range	$\theta \in [0, \infty)$	Same
Mean	b	$1/\beta$
Variance	b^2	$1/\beta^2$
Mode	0	0
Random number	The exponential distribution is a special case of the gamma distribution: $\theta \sim \text{gamma}(1, \text{scale} = b)$ is a draw from the exponential distribution.	

Table 80.19 Gamma Distribution

PROC specification	**gamma**(a, scale = b)	**gamma**(a, iscale = β)
Density	$\frac{1}{b^a\Gamma(a)}\theta^{a-1}e^{-\theta/b}$	$\frac{\beta^a}{\Gamma(a)}\theta^{a-1}e^{-\beta\theta}$
Parameter restriction	$a > 0, b > 0$	$a > 0, \beta > 0$
Range	$\theta \in [0, \infty)$ if $a = 1$; $(0, \infty)$ otherwise.	Same
Mean	ab	a/β
Variance	ab^2	a/β^2
Mode	$(a-1)b$ if $a \geq 1$	$(a-1)/\beta$ if $a \geq 1$
Random number	See (McGrath and Irving 1973).	

Table 80.20 Geometric Distribution

PROC specification	**geo**(p)
Density *	$p(1-p)^{\theta}$
Parameter restriction	$0 < p \leq 1$
Range	$\theta \in \begin{cases} \{0,1,2,\ldots\} & 0 < p < 1 \\ \{0\} & p = 1 \end{cases}$
Mean	$\frac{1-p}{p}$
Variance	$\frac{1-p}{p^2}$
Mode	0
Random number	Based on samples obtained from a Bernoulli distribution with probability p until the first success.

*The random variable θ is the total number of failures in an experiment *before* the first success. This density function is not to be confused with another popular formulation, $p(1-p)^{\theta-1}$, which counts the total number of trials *until* the first success.

Table 80.21 Inverse χ^2 Distribution

PROC specification	**ichisq**(ν)
Density	$\frac{1}{\Gamma(\nu/2)2^{\nu/2}}\theta^{-(\nu/2+1)}e^{-1/(2\theta)}$
Parameter restriction	$\nu > 0$
Range	$\theta \in (0,\infty)$
Mean	$\frac{1}{\nu-2}$ if $\nu > 2$
Variance	$\frac{2}{(\nu-2)^2(\nu-4)}$ if $\nu > 4$
Mode	$\frac{1}{\nu+2}$
Random number	Inverse χ^2 is a special case of the inverse gamma distribution: $\theta \sim \text{igamma}(\nu/2, \text{iscale} = 2)$ is a draw from the inverse χ^2 distribution.

Table 80.22 Inverse Gamma Distribution

PROC specification	**igamma**(a, scale = b)	**igamma**(a, iscale = β)
Density	$\frac{b^a}{\Gamma(a)}\theta^{-(a+1)}e^{-b/\theta}$	$\frac{1}{\beta^a\Gamma(a)}\theta^{-(a+1)}e^{-1/\beta\theta}$
Parameter restriction	$a > 0, b > 0$	$a > 0, \beta > 0$
Range	$\theta \in (0, \infty)$	Same
Mean	$\frac{b}{a-1}$ if $a > 1$	$\frac{1}{\beta(a-1)}$ if $a > 1$
Variance	$\frac{b^2}{(a-1)^2(a-2)}$	$\frac{1}{\beta^2(a-1)^2(a-2)}$
Mode	$\frac{b}{a+1}$	$\frac{1}{\beta(a+1)}$
Random number	Generate $x_1 \sim$ gamma(a, scale $= b$), and $\theta = 1/x_1$ is a draw from the igamma(a, iscale $= b$) distribution.	
Relationship to the gamma distribution	$\theta \sim$ gamma(a, iscale $= b$) $\Leftrightarrow$ $1/\theta \sim$ igamma(a, scale $= b$)	

Table 80.23 Laplace (Double Exponential) Distribution

PROC specification	**laplace**(a, scale = b)	**laplace**(a, iscale = β)
Density	$\frac{1}{2b}e^{-\lvert\theta-a\rvert/b}$	$\frac{\beta}{2}e^{-\beta\lvert\theta-a\rvert}$
Parameter restriction	$b > 0$	$\beta > 0$
Range	$\theta \in (-\infty, \infty)$	Same
Mean	a	a
Variance	$2b^2$	$2/\beta^2$
Mode	a	a
Random number	Inverse CDF. $F(\theta) = \begin{cases} \frac{1}{2}\exp\left(-\frac{a-\theta}{b}\right) & \theta < a \\ 1 - \frac{1}{2}\exp\left(-\frac{\theta-a}{b}\right) & \theta \geq a \end{cases}$. Generate $u_1, u_2 \sim$ uniform(0, 1). If $u_1 < 0.5, \theta = a + b\log(u_2)$; else $\theta = a - b\log(u_2)$. θ is a draw from the Laplace distribution.	

Table 80.24 Logistic Distribution

PROC specification	**logistic**(a, b)
Density	$\frac{\exp\left(-\frac{\theta-a}{b}\right)}{b\left(1+\exp\left(-\frac{\theta-a}{b}\right)\right)^2}$
Parameter restriction	$b > 0$
Range	$\theta \in (-\infty, \infty)$
Mean	a
Variance	$\frac{\pi^2 b^2}{3}$
Mode	a
Random number	Inverse CDF method with $F(\theta) = \left(1 + \exp(-\frac{\theta-a}{b})\right)^{-1}$. Generate $u \sim$ uniform$(0, 1)$, and $\theta = a - b\log(1/u - 1)$ is a draw from the logistic distribution.

Table 80.25 Lognormal Distribution

PROC specification	**lognormal**$(\mu$, sd = $s)$	**lognormal**$(\mu$, var = $v)$	**lognormal**$(\mu$, prec = $\tau)$
Density	$\frac{1}{\theta s\sqrt{2\pi}} \exp\left(-\frac{(\log\theta-\mu)^2}{2s^2}\right)$	$\frac{1}{\theta\sqrt{2\pi v}} \exp\left(-\frac{(\log\theta-\mu)^2}{2v}\right)$	$\frac{1}{\theta}\sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau(\log\theta-\mu)^2}{2}\right)$
Parameter restriction	$s > 0$	$v > 0$	$\tau > 0$
Range	$\theta \in (0, \infty)$	Same	Same
Mean	$\exp(\mu + s^2/2)$	$\exp(\mu + v/2)$	$\exp(\mu + 1/(2\tau))$
Variance	$\exp(2(\mu + s^2)) - \exp(2\mu + s^2)$	$\exp(2(\mu + v)) - \exp(2\mu + v)$	$\exp(2(\mu + 1/\tau)) - \exp(2\mu + 1/\tau)$
Mode	$\exp(\mu - s^2)$	$\exp(\mu - v)$	$\exp(\mu - 1/\tau)$
Random number	Generate $x_1 \sim$ normal$(0, 1)$, and $\theta = \exp(\mu + sx_1)$ is a draw from the lognormal distribution.		

Table 80.26 Negative Binomial Distribution

PROC specification	**negbin**(n, p)
Density	$\binom{\theta + n - 1}{n - 1} p^n (1 - p)^\theta$
Parameter restriction	$n = 1, 2, \ldots,$ and$0 < p \leq 1$

Range	$\theta \in \begin{cases} \{0,1,2,\ldots\} & 0 < p < 1 \\ \{0\} & p = 1 \end{cases}$
Mean	$\text{round}\left(\frac{n(1-p)}{p}\right)$
Variance	$\frac{n(1-p)}{p^2}$
Mode	$\begin{cases} 0 & n = 1 \\ \text{round}\left(\frac{(n-1)(1-p)}{p}\right) & n > 1 \end{cases}$
Random number	Generate $x_1 \sim \text{gamma}(n,1)$, and $\theta \sim \text{Poisson}(x_1 \cdot (1-p)/p)$ (Fishman 1996).

Table 80.27 Normal Distribution

PROC specification	**normal**(μ, sd = s)	**normal**(μ, var = v)	**normal**(μ, prec = τ)
Density	$\frac{1}{s\sqrt{2\pi}} \exp\left(-\frac{(\theta-\mu)^2}{2s^2}\right)$	$\frac{1}{\sqrt{2\pi v}} \exp\left(-\frac{(\theta-\mu)^2}{2v}\right)$	$\sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau(\theta-\mu)^2}{2}\right)$
Parameter restriction	$s > 0$	$v > 0$	$\tau > 0$
Range	$\theta \in (-\infty, \infty)$	Same	Same
Mean	μ	Same	Same
Variance	s^2	v	$1/\tau$
Mode	μ	Same	Same

Table 80.28 NormalCAR Distribution

PROC specification	**normalcar**(**neighbors=**, **num=**, sd= s)	**normalcar**(**neighbors=**, **num=**, var= v)	**normalcar**(**neighbors=**, **num=**, prec= τ)
Density	$\theta_i \mid \theta_{-i} \sim N(\sum_{j \in N(i)} \theta_j / m_i, s^2)$	$\theta_i \mid \theta_{-i} \sim N(\sum_{j \in N(i)} \theta_j / m_i, v)$	$\theta_i \mid \theta_{-i} \sim N(\sum_{j \in N(i)} \theta_j / m_i, 1/\tau)$
Notation	i is the area or site index, m_i is the number of neighbors to i, and $N(i)$ is the set of neighbors to i.		
Parameter restriction	$s > 0$	$v > 0$	$\tau > 0$
Range	$\theta \in (-\infty, \infty)$	Same	Same
Mean	average of neighbors	Same	Same
Variance	s^2	v	$1/\tau$
Mode	average of neighbors	Same	Same

Table 80.29 Pareto Distribution

PROC specification	**pareto**(a, b)
Density	$\frac{a}{b}\left(\frac{b}{\theta}\right)^{a+1}$
Parameter restriction	$a > 0, b > 0$
Range	$\theta \in [b, \infty)$
Mean	$\frac{ab}{a-1}$ if $a > 1$
Variance	$\frac{b^2 a}{(a-1)^2(a-2)}$ if $a > 2$
Mode	b
Random number	Inverse CDF method with $F(\theta) = 1 - (b/\theta)^a$. Generate $u \sim$ uniform(0, 1), and $\theta = \frac{b}{u^{1/a}}$ is a draw from the Pareto distribution.
Useful transformation	$x = 1/\theta$ is Beta$(a, 1)\mathbf{I}\{x < 1/b\}$.

Table 80.30 Poisson Distribution

PROC specification	**poisson**(λ)
Density	$\frac{\lambda^\theta}{\theta!}\exp(-\lambda)$
Parameter restriction	$\lambda \geq 0$
Range	$\theta \in \begin{cases} \{0, 1, \ldots\} & \text{if } \lambda > 0 \\ \{0\} & \text{if } \lambda = 0 \end{cases}$
Mean	λ
Variance	λ, if $\lambda > 0$
Mode	round(λ)

Table 80.31 Scaled Inverse χ^2 Distribution

PROC specification	**sichisq**(ν, s^2)
Density	$\frac{(s^2\nu/2)^{\nu/2}}{\Gamma(\nu/2)}\theta^{-(\nu/2+1)}e^{-\nu s^2/(2\theta)}$
Parameter restriction	$\nu > 0, s > 0$
Range	$\theta \in (0, \infty)$
Mean	$\frac{\nu}{\nu-2}s^2$ if $\nu > 2$
Variance	$\frac{2\nu^2}{(\nu-2)^2(\nu-4)}s^4$ if $\nu > 4$
Mode	$\frac{\nu}{\nu+2}s^2$
Random number	Scaled inverse χ^2 is a special case of the inverse gamma distribution: $\theta \sim$ igamma$(\nu/2, \text{scale} = (\nu s^2)/2)$ is a draw from the scaled inverse χ^2 distribution.

Table 80.32 t Distribution

PROC specification	**t**(μ, sd = s, ν)	**t**(μ, var = v, ν)	**t**(μ, prec = τ, ν)
Density	$\frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})s\sqrt{\nu\pi}}(1+\frac{(\theta-\mu)^2}{\nu s^2})^{-\frac{\nu+1}{2}}$	$\frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})\sqrt{\nu\pi v}}(1+\frac{(\theta-\mu)^2}{\nu v})^{-\frac{\nu+1}{2}}$	$\frac{\Gamma(\frac{\nu+1}{2})\sqrt{\tau}}{\Gamma(\frac{\nu}{2})\sqrt{\nu\pi}}(1+\frac{\tau(\theta-\mu)^2}{\nu})^{-\frac{\nu+1}{2}}$
Parm restriction	$s > 0, \nu > 0$	$v > 0, \nu > 0$	$\tau > 0, \nu > 0$
Range	$\theta \in (-\infty, \infty)$	Same	Same
Mean	μ if $\nu > 1$	Same	Same
Variance	$\frac{\nu}{\nu-2}s^2$ if $\nu > 2$	$\frac{\nu}{\nu-2}v$ if $\nu > 2$	$\frac{\nu}{\nu-2}\frac{1}{\tau}$ if $\nu > 2$
Mode	μ	Same	Same
Random number	$x_1 \sim$ normal$(0,1)$, $x_2 \sim \chi^2(d)$, and $\theta = m + \sigma x_1\sqrt{d/x_2}$ is a draw from the t distribution.		

Table 80.33 Table (Categorical) Distribution

PROC specification	**table**(**p**), where $\mathbf{p} = \{p_i\}$, for $i = 1, 2, \ldots, k$
Density	$f(\theta = i) = p_i$
Parameter restriction	$\sum_i^k p_i = 1$ with all $p_i > 0$
Range	$\theta \in \{1, 2, \ldots, k\}$
Mode	i such that $p_i = \max(p_1, \ldots, p_k)$
Random number	Inverse CDF method with $F(\theta = i) = \sum_{j=1}^{i} p_j$.

Table 80.34 Uniform Distribution

PROC specification	**uniform**(a, b)
Density	$\begin{cases} \frac{1}{a-b} & \text{if } a > b \\ \frac{1}{b-a} & \text{if } b > a \\ 1 & \text{if } a = b \end{cases}$
Parameter restriction	none
Range	$\theta \in [a, b]$
Mean	$\frac{a+b}{2}$
Variance	$\frac{\lvert b-a\rvert^2}{12}$
Mode	Does not exist
Random number	Mersenne Twister (Matsumoto and Kurita 1992, 1994; Matsumoto and Nishimura 1998)

Table 80.35 Wald Distribution

PROC specification	**wald**(μ, λ)
Density	$\sqrt{\frac{\lambda}{2\pi\theta^3}} \exp\left(\frac{-\lambda(\theta-\mu)^2}{2\mu^2\theta}\right)$
Parameter restriction	$\mu > 0, \lambda > 0$
Range	$\theta \in (0, \infty)$
Mean	μ
Variance	μ^3/λ
Mode	$\mu\left[\left(1 + \frac{9\mu^2}{4\lambda^2}\right)^{1/2} - \frac{3\mu}{2\lambda}\right]$
Random number	Generate $\nu_0 \sim \chi^2_{(1)}$. Let $x_1 = \mu + \frac{\mu^2\nu_0}{2\lambda} - \frac{\mu}{2\lambda}\sqrt{4\mu\lambda\nu_0 + \mu^2\nu_0^2}$ and $x_2 = \mu^2/x_1$. Perform a Bernoulli trial, $w \sim$ Bernoulli$(\frac{\mu}{\mu+x_1})$. If $w = 1$, choose $\theta = x_1$; otherwise, choose $\theta = x_2$ (Michael, Schucany, and Haas 1976).

Table 80.36 Weibull Distribution

PROC specification	**weibull**(μ, c, σ)
Density	$\exp\left(-\left(\frac{\theta-\mu}{\sigma}\right)^c\right) \frac{c}{\sigma}\left(\frac{\theta-\mu}{\sigma}\right)^{c-1}$
Parameter restriction	$c > 0, \sigma > 0$
Range	$\theta \in [\mu, \infty)$ if $c = 1$; (μ, ∞) otherwise
Mean	$\mu + \sigma\Gamma(1 + 1/c)$
Variance	$\sigma^2[\Gamma(1 + 2/c) - \Gamma^2(1 + 1/c)]$
Mode	$\mu + \sigma(1 - 1/c)^{1/c}$ if $c > 1$
Random number	Inverse CDF method with $F(\theta) = 1 - \exp\left(-\left(-\frac{\theta-\mu}{\sigma}\right)^c\right)$. Generate $u \sim$ uniform$(0, 1)$, and $\theta = \mu + \sigma \cdot (-\ln u)^{1/c}$ is a draw from the Weibull distribution.

Multivariate Distributions

Table 80.37 Dirichlet Distribution

PROC specification	$\boldsymbol{\theta} \sim$ **dirich**$(\boldsymbol{\alpha})$, where $\boldsymbol{\theta} = \{\theta_i\}, \boldsymbol{\alpha} = \{\alpha_i\}$, for $i = 1, 2, \ldots, k$
Density	$\frac{\Gamma(\alpha_0)}{\prod_{i=1}^k \Gamma(\alpha_i)} \prod_{i=1}^k \theta_i^{\alpha_i - 1}$, where $\alpha_0 = \sum_{i=1}^k \alpha_i$
Parameter restriction	$\alpha_i > 0$
Range	$\theta_i > 0, \sum_{i=1}^k \theta_i = 1$
Mean	α_j / α_0
Mode	$(\alpha_j - 1)/(\alpha_0 - k)$

Table 80.38 Inverse Wishart Distribution

PROC specification	$\boldsymbol{\theta} \sim \mathbf{iwishart}(\nu, \mathbf{S})$, both $\boldsymbol{\theta}$ and $\mathbf{S}$ are $k \times k$ matrices
Density	$\left(2^{\frac{\nu k}{2}} \pi^{\frac{k(k-1)}{4}} \prod_{i=1}^{k} \Gamma\left(\frac{\nu+1-i}{2}\right)\right)^{-1} \lvert\mathbf{S}\rvert^{\frac{\nu}{2}} \lvert\boldsymbol{\theta}\rvert^{-\frac{\nu+k+1}{2}} \exp\left(-\frac{1}{2}\mathrm{tr}(\mathbf{S}\boldsymbol{\theta}^{-1})\right)$
Parameter restriction	$\mathbf{S}$ must be symmetric and positive definite; $\nu > k-1$
Range	$\boldsymbol{\theta}$ is symmetric and positive definite
Mean	$\mathbf{S}/(\nu-k-1)$
Mode	$\mathbf{S}/(\nu+k+1)$

Table 80.39 Multivariate Normal Distribution

PROC specification	$\boldsymbol{\theta} \sim \mathbf{mvn}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{\theta} = \{\theta_k\}$, $\boldsymbol{\mu} = \{\mu_k\}$, for $i = 1, 2, \ldots, k$, and $\boldsymbol{\Sigma}$ is a $k \times k$ variance matrix
Density	$\exp\left(-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{\theta}-\boldsymbol{\mu})\right) \Big/ \sqrt{(2\pi)^k \lvert\boldsymbol{\Sigma}\rvert}$
Parameter restriction	$\boldsymbol{\Sigma}$ must be symmetric and positive definite
Range	$-\infty < \theta_i < \infty$
Mean	$\boldsymbol{\mu}$
Mode	$\boldsymbol{\mu}$

Table 80.40 Autoregressive Multivariate Normal Distribution

PROC specification	$\boldsymbol{\theta} \sim$**MVNAR**($\boldsymbol{\mu}$, sd=σ,ρ) $\boldsymbol{\theta} \sim$**MVNAR**($\boldsymbol{\mu}$, var=σ^2,ρ) $\boldsymbol{\theta} \sim$**MVNAR**($\boldsymbol{\mu}$, prec=$1/\sigma^2$, ρ)
Density	$\exp\left(-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})'(\sigma^2\boldsymbol{\Sigma})^{-1}(\boldsymbol{\theta}-\boldsymbol{\mu})\right) \Big/ \sqrt{(2\pi)^k \left\lvert(\sigma^2\boldsymbol{\Sigma})\right\rvert}$ where $$\boldsymbol{\Sigma} = \begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 & \cdots & \rho^k \\ \rho & 1 & \rho & \rho^2 & \cdots & \rho^{k-1} \\ \rho^2 & \rho & 1 & \rho & \cdots & \rho^{k-2} \\ \rho^3 & \rho^2 & \rho & 1 & \cdots & \rho^{k-3} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho^k & \rho^{k-1} & \rho^{k-2} & \rho^{k-3} & \cdots & 1 \end{bmatrix}$$
Parameter restriction	$\sigma > 0$ and $-1 < \rho < 1$
Range	$-\infty < \theta_i < \infty$
Mean	$\boldsymbol{\mu}$
Mode	$\boldsymbol{\mu}$
Special Case	When $\rho = 0$, the distribution simplifies to $\mathbf{mvn}(\boldsymbol{\mu}, \sigma^2 \cdot \mathbf{I}_k)$, where $\mathbf{I}_k$ denotes the $k \times k$ identity matrix

Table 80.41 Multinomial Distribution

PROC specification	$\boldsymbol{\theta} \sim$ **multinom**(**p**), where $\boldsymbol{\theta} = \{\theta_i\}$ and $\mathbf{p} = \{p_i\}$, for $i = 1, 2, \ldots, k$
Density	$\frac{n!}{\theta_1 \cdots \theta_k} p_1^{\theta_1} \cdots p_k^{\theta_k}$, where $\sum_i^k \theta_i = n$
Parameter restriction	$\sum_i^k p_i = 1$ with all $p_i > 0$
Range	$\theta_i \in \{0, \ldots, n\}$, nonnegative integers
Mean	$n \cdot \mathbf{p}$

Usage of Multivariate Distributions

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcmvn.sas.)

The following simple example illustrates the usage of the multivariate distributions in PROC MCMC. Suppose you are interested in estimating the mean and covariance of multivariate data using this multivariate normal model:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim \text{MVN}\left(\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{pmatrix}\right)$$

where

$$\begin{aligned} \boldsymbol{\mu} &= \begin{pmatrix} 1 \\ 2 \end{pmatrix} \\ \boldsymbol{\Sigma} &= \begin{pmatrix} 2.4 & 3 \\ 3 & 8.1 \end{pmatrix} \end{aligned}$$

You can use the following independent prior on $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$:

$$\begin{aligned} \boldsymbol{\mu} &\sim \text{MVN}\left(\boldsymbol{\mu_0} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \boldsymbol{\Sigma_0} = \begin{pmatrix} 100 & 0 \\ 0 & 100 \end{pmatrix}\right) \\ \boldsymbol{\Sigma} &\sim \text{iWishart}\left(\nu = 2, \text{S} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right) \end{aligned}$$

The following IML procedure statements simulate 100 random multivariate normal samples:

```
title 'An Example that Uses Multivariate Distributions';
proc iml;
   N = 100;
   Mean = {1 2};
   Cov =  {2.4 3, 3 8.1};
   call randseed(1);
   x = RANDNORMAL( N, Mean, Cov );

   SampleMean = x[:,];
```

```
   n = nrow(x);
   y = x - repeat( SampleMean, n );
   SampleCov = y`*y / (n-1);
   print SampleMean Mean, SampleCov Cov;

   cname = {"x1", "x2"};
   create inputdata from x [colname = cname];
   append from x;
   close inputdata;
quit;
```

Figure 80.13 prints the sample mean and covariance of the simulated data, in addition to the true mean and covariance matrix.

Figure 80.13 Simulated Multivariate Normal Data

An Example that Uses Multivariate Distributions

SampleMean		Mean	
0.9987751	2.115693	1	2

SampleCov		Cov	
2.8252975	3.7190704	2.4	3
3.7190704	9.2916805	3	8.1

The following PROC MCMC statements estimate the posterior mean and covariance of the multivariate normal data:

```
proc mcmc data=inputdata seed=17 nmc=3000 diag=none;
   ods select PostSumInt;
   array data[2] x1 x2;
   array mu[2];
   array Sigma[2,2];
   array mu0[2] (0 0);
   array Sigma0[2,2] (100 0 0 100);
   array S[2,2] (1 0 0 1);
   parm mu Sigma;
   prior mu ~ mvn(mu0, Sigma0);
   prior Sigma ~ iwish(2, S);
   model data ~ mvn(mu, Sigma);
run;
```

To use the multivariate distribution, you must specify parameters (or random variables in the MODEL statement) in an array form. The first ARRAY statement creates an one-dimensional array data, which contains two numeric variables, x1 and x2, from the input data set. The data variable is your response variable. The subsequent statements defines two array-parameters (mu and Sigma) and three constant array-hyperparameters (mu0, Sigma0, and S). The PARMS statement declares mu and Sigma to be model parameters. The two PRIOR statements specify the multivariate normal and inverse Wishart distributions as the prior for mu and Sigma, respectively. The MODEL statement specifies the multivariate normal likelihood with data as the random variable, mu as the mean, and Sigma as the covariance matrix.

Figure 80.14 lists the estimated posterior statistics for the parameters.

Figure 80.14 Estimated Mean and Covariance

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
mu1	3000	0.9941	0.1763	0.6338	1.3106
mu2	3000	2.1135	0.3112	1.4939	2.7165
Sigma1	3000	2.8726	0.4084	2.1001	3.6723
Sigma2	3000	3.7573	0.6418	2.5791	5.0223
Sigma3	3000	3.7573	0.6418	2.5791	5.0223
Sigma4	3000	9.3987	1.3224	7.0155	12.0969

Specifying a New Distribution

To work with a new density that is not listed in the section "Standard Distributions" on page 6289, you can use the GENERAL and DGENERAL functions. The letter "D" stands for discrete. The new distributions have to be specified on the logarithm scale.

Suppose you want to use the inverse beta distribution:

$$p(\alpha|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)+\Gamma(b)} \cdot \alpha^{(a-1)} \cdot (1+\alpha)^{-(a+b)}$$

The following statements in PROC MCMC define the density on its log scale:

```
a = 3; b = 5;
const = lgamma(a + b) - lgamma(a) - lgamma(b);
lp = const + (a - 1) * log(alpha) - (a + b) * log(1 + alpha);
prior alpha ~ general(lp);
```

The symbol lp is the expression for the log of an inverse beta (a = 3, b = 5). The function **`general(lp)`** assigns that distribution to alpha. The constant term, const, can be omitted because the Markov simulation requires only the log of the density kernel.

You can use the GENERAL function to specify a distribution for a single variable or for multiple variables. It is important to emphasize that the argument *lp* is an expression for the log of the joint distribution for these variables. On the contrary, any standard distribution is applied separately to each random variable in the statement.

When you use the GENERAL function in the MODEL statement, you do not need to specify the dependent variable on the left of the tilde ∼. The log-likelihood function takes the dependent variable into account; hence, there is no need to explicitly state the dependent variable in the MODEL statement. However, in the PRIOR and RANDOM statements, you need to explicitly state the parameter names and a tilde with the GENERAL function.

You can specify any distribution function by using the GENERAL and DGENERAL functions as long as the distribution function is programmable with SAS statements. When the function is used in the PRIOR statements, you must supply initial values in either the PARMS statement or within the BEGINCNST and ENDCNST statements. See the sections "PARMS Statement" on page 6263 and "BEGINCNST/ENDCNST

Statement" on page 6248. When the function is used in the RANDOM statement, you must use the INITIAL= option in the RANDOM statement to supply initial values

NOTE: PROC MCMC does not verify that the GENERAL function you specify is a valid distribution—that is, an integrable density. You must use the function with caution.

Using Density Functions in the Programming Statements

Density Functions in PROC MCMC

PROC MCMC has a number of internally defined log-density functions for univariate and multivariate distributions. These functions have the basic form of LPDF*dist*(*x*, *parm-list*), where *dist* is the name of the distribution (see Table 80.42 for univariate distributions and Table 80.43 for multivariate distributions). The argument *x* is the random variable, and *parm-list* is the list of parameters.

In addition, the univariate functions allow for optional boundary arguments, such as LPDF*dist*(*x*, *parm-list*, < *lower* >, < *upper* >), where *lower* and *upper* are optional but positional boundary arguments. With the exception of the Bernoulli and uniform distribution, you can specify limits on all univariate distributions.

To set a lower bound on the normal density:

```
lpdfnorm(x, 0, 1, -2);
```

To set just an upper bound, specify a missing value for the lower bound argument:

```
lpdfnorm(x, 0, 1, ., 2);
```

Leaving both limits out gives you the unbounded density. You can also specify both bounds:

```
lpdfnorm(x, 0, 1);
lpdfnorm(x, 0, 1, -3, 4);
```

See Table 80.42 for the function names of univariate distributions and Table 80.43 for multivariate distributions.

Table 80.42 Logarithm of Univariate Density Functions in PROC MCMC

Distribution Name	Function Call
Beta	**lpdfbeta(***x, a, b,*< *lower*>, < *upper*>**);**
Binary	**lpdfbern(***x, p***);**
Binomial	**lpdfbin(***x, n,p,* < *lower*>, < *upper*>**);**
Cauchy	**lpdfcau(***x, loc, scale,* < *lower*>, < *upper*>**);**
χ^2	**lpdfchisq(***x, df,*< *lower*>, < *upper*>**);**
Exponential χ^2	**lpdfechisq(***x, df,* < *lower*>, < *upper*>**);**

Table 80.42 *continued*

Distribution Name	Function Call
Exponential gamma	**lpdfegamma(***x, sp,scale, < lower>, < upper>***);**
Exponential exponential	**lpdfeexpon(***x, scale,< lower>, < upper>***);**
Exponential inverse χ^2	**lpdfeichisq(***x, df, < lower>, < upper>***);**
Exponential inverse gamma	**lpdfeigamma(***x, sp, scale, < lower>, < upper>***);**
Exponential scaled inverse χ^2	**lpdfesichisq(***x, df, scale, < lower>, < upper>***);**
Exponential	**lpdfexpon(***x, scale, < lower>, < upper>***);**
Gamma	**lpdfgamma(***x, sp, scale, < lower>, < upper>***);**
Geometric	**lpdfgeo(***x, p, < lower>, < upper>***);**
Inverse χ^2	**lpdfichisq(***x, df, < lower>, < upper>***);**
Inverse gamma	**lpdfigamma(***x, sp, scale, < lower>, < upper>***);**
Laplace	**lpdfdexp(***x, loc, scale, < lower>, < upper>***);**
Logistic	**lpdflogis(***x, loc, scale, < lower>, < upper>***);**
Lognormal	**lpdflnorm(***x, loc, sd, < lower>, < upper>***);**
Negative binomial	**lpdfnegbin(***x, n, p, < lower>, < upper>***);**
Normal	**lpdfnorm(***x, mu, sd, < lower>, < upper>***);**
Pareto	**lpdfpareto(***x, sp, scale, < lower>, < upper>***);**
Poisson	**lpdfpoi(***x, mean, < lower>, < upper>***);**
Scaled inverse χ^2	**lpdfsichisq(***x, df, scale, < lower>, < upper>***);**
t	**lpdft(***x, mu, sd, df, < lower>,< upper>***);**
Uniform	**lpdfunif(***x, a, b***);**
Wald	**lpdfwald(***x, mean, scale, < lower>, < upper>***);**
Weibull	**lpdfwei(***x, loc, sp, scale, < lower>, < upper>***);**

In the multivariate log-density functions, arrays must be used in place for the random variable and parameters in the model.

Table 80.43 Logarithm of Multivariate Density Functions in PROC MCMC

Distribution Name	Function Call
Dirichlet	**lpdfdirich(***x_array, alpha_array***);**
Inverse Wishart	**lpdfiwish(***x_array, df, S_array***);**
Multivariate normal	**lpdfmvn(***x_array, mu_array, cov_array***);**
Multinomial	**lpdfmnom(***x_array, p_array***);**

Standard Distributions, the LOGPDF Functions, and the LPDF*dist* Functions

Standard distributions listed in the section "Standard Distributions" on page 6289 are *names* only, and they can be used only in the MODEL, PRIOR, and HYPERPRIOR statements to specify either a prior distribution or a conditional distribution of the data given parameters. They do not return any values, and you cannot use them in the programming statements.

The LOGPDF functions are DATA step functions that compute the logarithm of various probability density (mass) functions. For example,

logpdf(*"beta", x, 2, 15***);**

returns the log of a beta density with parameters a = 2 and b = 15, evaluated at x. All the LOGPDF functions are supported in PROC MCMC.

The LPDF*dist* functions are unique to PROC MCMC. They compute the logarithm of various probability density (mass) functions. The functions are the same as the LOGPDF functions when it comes to calculating the log density. For example,

lpdfbeta(*x, 2,15***);**

returns the same value as

logpdf(*"beta", x, 2, 15***);**

The LPDF*dist* functions cover a greater class of probability density functions, and the univariate distribution functions take the optional but positional boundary arguments. There are no corresponding LCDF*dist* or LSDF*dist* functions in PROC MCMC. To work with the cumulative probability function or the survival functions, you need to use the LOGCDF and the LOGSDF DATA step functions.

Truncation and Censoring

Truncated Distributions

To specify a truncated distribution, you can use the LOWER= and/or UPPER= options. Almost all of the univariate standard distributions, including the GENERAL and DGENERALfunctions, take these optional truncation arguments. The binary, the uniform, and the tabled distributions do not support these truncation options. Multivariate distributions, such as the multivariate normal, do not support these options neither.

For example, you can specify the following:

```
prior alpha ~ normal(mean = 0, sd = 1, lower = 3, upper = 45);
```

or

```
parms beta;
a = 3; b = 7;
ll = (a + 1) * log(b / beta);
prior beta ~ general(ll, upper = b + 17);
```

The preceding statements state that if beta is less than b+17, the log of the prior density is ll, as calculated by the equation; otherwise, the log of the prior density is missing—the log of zero.

When the same distribution is applied to multiple parameters in a PRIOR statement, the LOWER= and UPPER= truncations apply to all parameters in that statement. For example, the following statements define a Poisson density for theta and gamma:

```
parms theta gamma;
lambda = 7;
l1 = theta * log(lambda) - lgamma(1 + theta);
l2 = gamma * log(lambda) - lgamma(1 + gamma);
ll = l1 + l2;
prior theta gamma ~ dgeneral(ll, lower = 1);
```

The LOWER=1 condition is applied to both theta and gamma, meaning that for the assignment to ll to be meaningful, both theta and gamma have to be greater than 1. If either of the parameters is less than 1, the log of the joint prior density becomes a missing value.

PROC MCMC calculates the normalizing constant in all truncated distributions (with exception to the GENERAL and the DGENERAL functions), and you can use parameters in the LOWER= or UPPER= option.

Note that if you use either the GENERAL or DGENERAL function, you must compute the normalizing constant in cases where it is required. A truncated distribution has the probability distribution

$$p(\theta|a < \theta < b) = \frac{p(\theta)}{F(b) - F(a)}$$

where $p(\cdot)$ is the density function and $F(\cdot)$ is the cumulative distribution function. In SAS functions, $p(\cdot)$ is the probability density function and $F(\cdot)$ is the cumulative distribution function. The following example shows how to construct a truncated gamma prior on theta, with SHAPE=3, SCALE=2, LOWER=A, and UPPER=B:

```
lp = logpdf('gamma', theta, 3, 2)
        - log(cdf('gamma', a, 3, 2) - cdf('gamma', b, 3, 2));
prior theta ~ general(lp);
```

This density specification is different from the following more naive definition, without taking into account the normalizing constant:

```
lp = logpdf('gamma', theta, 3, 2);
prior theta ~ general(lp, lower=a, upper=b);
```

If a or b is a parameter, you get very different results from the two formulations.

Censoring

You can use either of two approaches to model censored data. One is to specify the marginal distribution, and the other is to treat the censored data as missing values.

Suppose you partition the data into four categories: uncensored (with observation x), left-censored (with observation xl), right-censored (with observation xr), and interval-censored (with observations xl and xr). The likelihood is the normal distribution with mean mu and standard deviation s. The following statements construct the corresponding log likelihood for the observed data:

```
if uncensored then
   ll = logpdf('normal', x, mu, s);
else if leftcensored then
   ll = logcdf('normal', xl, mu, s);
else if rightcensored then
   ll = logsdf('normal', xr, mu, s);
else /* this is the case of interval censored. */
   ll = log(cdf('normal', xr, mu, s) - cdf('normal', xl, mu, s));
model general(ll);
```

Alternatively, you can treat censored data as missing values and impute the values in the Markov chain. In the following statement, the CLOWER= and CUPPER= options are the censoring indicators:

```
model x ~ normal(mu, sd=1, clower=xl, cupper=xr);
```

Missing x values become parameters, and PROC MCMC samples according to the censoring information. Specify the MISSING=ACMODELY option in the PROC MCMC statement if the xl or xr variables contain missing values. This option enables PROC MCMC to draw missing response variables without discarding observations that have missing covariates. By default, PROC MCMC models missing values but throws away observations that have missing values in nonresponse variables.

See "Example 80.17: Normal Regression with Interval Censoring" on page 6470.

Some Useful SAS Functions

Table 80.44 Some Useful SAS Functions

SAS Function	Definition
abs(*x***);**	$\|x\|$
airy(*x***);**	Returns the value of the AIRY function.
beta(*x1, x2***);**	$\int_0^1 z^{x1-1}(1-z)^{x2-1}dz$
call logistic(*x***);**	$\frac{\exp(x)}{1+\exp(x)}$
call softmax(*x1, . . . , xn***);**	Each element is replaced by $\exp(x_j)/\sum \exp(x_j)$
call stdize(*x1, . . . , xn***);**	Standardize values
cdf();	Cumulative distribution function
cdf(*'normal', x, 0, 1***);**	Standard normal cumulative distribution function
comb(*x1, x2***);**	$\frac{x1!}{x2!(x1-x2)!}$
constant(*'.'***);**	Calculate commonly used constants
cos(*x***);**	cosine(x)
css(*x1, . . . , xn***);**	$\sum_i (x_i - \bar{x})^2$
cv(*x1, . . . , xn***);**	std(x) / mean(x) * 100
dairy(*x***);**	Derivative of the AIRY function
dimN(*m***);**	Returns the numbers of elements in the Nth dim of array m
x1 eq x2	Returns 1 if x1 = x2; 0 otherwise
x1x2**	$x1^{x2}$
geomean(*x1, . . . , xn***);**	$\exp\left(\frac{\log(x_1)+\cdots+\log(x_n)}{n}\right)$
difN(*x***);**	Returns differences between the argument and its Nth lag
digamma(*x1***);**	$\frac{\Gamma'(x1)}{\Gamma(x1)}$
erf(*x***);**	$\frac{2}{\sqrt{\pi}}\int_0^x \exp(-z^2)dz$
erfc(*x***);**	1 – erf(x)
fact(*x***);**	$x!$
floor(*x***);**	Greatest integer $\leq x$
gamma(*x***);**	$\int_0^\infty z^{x-1}\exp(-1)dz$
harmean(*x1, . . . , xn***);**	$\frac{n}{1/x_1+\cdots 1/x_n}$
ibessel(*nu, x, kode***);**	Modified Bessel function of order *nu* evaluated at x
jbessel(*nu, x***);**	Bessel function of order *nu* evaluated at x
lagN(*x***);**	Returns values from a queue
largest(*k, x1, . . . , xn***);**	Returns the kth largest element
lgamma(*x***);**	$\ln(\Gamma(x))$
lgamma(*x+1***);**	$\ln(x!)$
log(*x, logN(x)***);**	$\ln(x)$
logbeta(*x1, x2***);**	lgamma(x_1) + lgamma(x_2) - lgamma($x_1 + x_2$)
logcdf();	Log of a left cumulative distribution function
logpdf();	Log of a probability density (mass) function
logsdf();	Log of a survival function
max(*x1, x2***);**	Returns x_1 if $x_1 > x_2$; x_2 otherwise
mean(*of x1–xn***);**	$\sum_i x_i/n$

Table 80.44 *continued*

SAS Function	Definition
median(*of x1–xn***);**	Returns the median of nonmissing values
min(*x1, x2***);**	Returns x_1 if $x_1 < x_2$; x_2 otherwise
missing(*x***);**	Returns 1 if x is missing; 0 otherwise
mod(*x1, x2***);**	Returns the remainder from x_1/x_2
n(*x1, . . . , xn***);**	Returns number of nonmissing values
nmiss(*of y1–yn***);**	Number of missing values
quantile();	Computes the quantile from a specific distribution
pdf();	Probability density (mass) functions
perm(*n, r***);**	$\frac{n!}{(n-r)!}$
put();	Returns a value that uses a specified format
round(*x***);**	Rounds x
rms(*of x1–xn***);**	$\sqrt{\frac{x_1^2+\cdots x_n^2}{n}}$
sdf();	Survival function
sign(*x***);**	Returns –1 if $x < 0$; 0 if $x = 0$; 1 if $x > 0$
sin(*x***);**	sine(x)
smallest(*s, x1, . . . , en* **);**	Returns the sth smallest component of $x_1, \ldots, x_n$
sortn(*of x1-xn***);**	Sorts the values of the variables
sqrt(*x***);**	$\sqrt{x}$
std(*x1, . . . , xn)* **);**	Standard deviation of $x_1, \ldots, x_n$ (n-1 in denominator)
sum(*of x:***);**	$\sum_i x_i$
trigamma(*x***);**	Derivative of the DIGAMMA(x) function
uss(*of x1–xn***);**	Uncorrected sum of squares

Here are examples of some commonly used transformations:

- logit

```
mu = beta0 + beta1 * z1;
call logistic(mu);
```

- log

```
w = beta0 + beta1 * z1;
mu = exp(w);
```

- probit

```
w = beta0 + beta1 * z1;
mu = cdf(`normal', w, 0, 1);
```

- cloglog

```
w = beta0 + beta1 * z1;
mu = 1 - exp(-exp(w));
```

Matrix Functions in PROC MCMC

The MCMC procedure provides you with a number of CALL routines for performing simple matrix operations on declared arrays. With the exception of FILLMATRIX, IDENTITY, and ZEROMATRIX, the CALL routines listed in Table 80.45 do not support matrices or arrays that contain missing values.

Table 80.45 Matrix Functions in PROC MCMC

CALL Routine	Description
ADDMATRIX	Performs an element-wise addition of two matrices or of a matrix and a scalar.
CHOL	Calculates the Cholesky decomposition for a particular symmetric matrix.
DET	Calculates the determinant of a specified matrix, which must be square.
ELEMMULT	Performs an element-wise multiplication of two matrices.
FILLMATRIX	Replaces all of the element values of the input matrix with the specified value. You can use this routine with multidimensional numeric arrays.
IDENTITY	Converts the input matrix to an identity matrix. Diagonal element values of the matrix are set to 1, and the rest of the values are set to 0.
INV	Calculates a matrix that is the inverse of the input matrix. The input matrix must be a square, nonsingular matrix.
MULT	Calculates the matrix product of two input matrices.
SUBTRACTMATRIX	Performs an element-wide subtraction of two matrices or of a matrix and a scalar.
TRANSPOSE	Returns the transpose of a matrix.
ZEROMATRIX	Replaces all of the element values of the numeric input matrix with 0.

ADDMATRIX CALL Routine

The ADDMATRIX CALL routine performs an element-wise addition of two matrices or of a matrix and a scalar.

The syntax of the ADDMATRIX CALL routine is

CALL ADDMATRIX (*X*, *Y*, *Z*) **;**

where

X specifies a scalar or an input matrix with dimensions $m \times n$ (that is, $X[m, n]$)

Y specifies a scalar or an input matrix with dimensions $m \times n$ (that is, $Y[m, n]$)

Z specifies an output matrix with dimensions $m \times n$ (that is, $Z[m, n]$)

such that

$$Z = X + Y$$

CHOL CALL Routine

The CHOL CALL routine calculates the Cholesky decomposition for a particular symmetric matrix.

The syntax of the CHOL CALL routine is

CALL CHOL (*X***,** *Y* **<,** *validate***>) ;**

where

X specifies a symmetric positive-definite input matrix with dimensions $m \times m$ (that is, *X*[m, m])

Y is a variable that contains the Cholesky decomposition and specifies an output matrix with dimensions $m \times m$ (that is, $Y[m, m]$)

validate specifies an optional argument that can increase the processing speed by avoiding error checking:

If *validate* = 0 or is not specified, then the matrix *X* is checked for symmetry.

If *validate* = 1, then the matrix *X* is assumed to be symmetric.

such that

$$X = YY^*$$

where Y is a lower triangular matrix with strictly positive diagonal entries and Y^* denotes the conjugate transpose of Y.

Both input and output matrices must be square and have the same dimensions. If X is symmetric positive-definite, Y is a lower triangle matrix. If X is not symmetric positive-definite, Y is filled with missing values.

DET CALL Routine

The determinant, the product of the eigenvalues, is a single numeric value. If the determinant of a matrix is zero, then that matrix is singular (that is, it does not have an inverse). The routine performs an LU decomposition and collects the product of the diagonals.

The syntax of the DET CALL routine is

CALL DET (*X***,** *a***) ;**

where

X specifies an input matrix with dimensions $m \times m$ (that is, $X[m, m]$)

a specifies the returned determinate value

such that

$$a = |X|$$

ELEMMULT CALL Routine

The ELEMMULT CALL routine performs an element-wise multiplication of two matrices.

The syntax of the ELEMMULT CALL routine is

CALL ELEMMULT (*X*, *Y*, *Z*) ;

where

X specifies an input matrix with dimensions $m \times n$ (that is, $X[m, n]$)

Y specifies an input matrix with dimensions $m \times n$ (that is, $Y[m, n]$)

Z specifies an output matrix with dimensions $m \times n$ (that is, $Z[m, n]$)

FILLMATRIX CALL Routine

The FILLMATRIX CALL routine replaces all of the element values of the input matrix with the specified value. You can use the FILLMATRIX CALL routine with multidimensional numeric arrays.

The syntax of the FILLMATRIX CALL routine is

CALL FILLMATRIX (*X*, *Y*) ;

where

X specifies an input numeric matrix

Y specifies the numeric value that is used to fill the matrix

IDENTITY CALL Routine

The IDENTITY CALL routine converts the input matrix to an identity matrix. Diagonal element values of the matrix are set to 1, and the rest of the values are set to 0.

The syntax of the IDENTITY CALL routine is

CALL IDENTITY (*X*) ;

where

X specifies an input matrix with dimensions $m \times m$ (that is, $X[m, m]$)

INV CALL Routine

The INV CALL routine calculates a matrix that is the inverse of the input matrix. The input matrix must be a square, nonsingular matrix.

The syntax of the INV CALL routine is

CALL INV (*X*, *Y*) ;

where

X specifies an input matrix with dimensions $m \times m$ (that is, $X[m, m]$)

Y specifies an output matrix with dimensions $m \times m$ (that is, $Y[m, m]$)

MULT CALL Routine

The MULT CALL routine calculates the matrix product of two input matrices.

The syntax of the MULT CALL routine is

CALL MULT (*X*, *Y*, *Z*) ;

where

X specifies an input matrix with dimensions $m \times n$ (that is, $X[m, n]$)

Y specifies an input matrix with dimensions $n \times p$ (that is, $Y[n, p]$)

Z specifies an output matrix with dimensions $m \times p$ (that is, $Z[m, p]$)

The number of columns for the first input matrix must be the same as the number of rows for the second matrix. The calculated matrix is the last argument.

SUBTRACTMATRIX CALL Routine

The SUBTRACTMATRIX CALL routine performs an element-wide subtraction of two matrices or of a matrix and a scalar.

The syntax of the SUBTRACTMATRIX CALL routine is

CALL SUBTRACTMATRIX (*X*, *Y*, *Z*) ;

where

X specifies a scalar or an input matrix with dimensions $m \times n$ (that is, $X[m, n]$)

Y specifies a scalar or an input matrix with dimensions $m \times n$ (that is, $Y[m, n]$)

Z specifies an output matrix with dimensions $m \times n$ (that is, $Z[m, n]$)

such that

$$Z = X - Y$$

TRANSPOSE CALL Routine

The TRANSPOSE CALL routine returns the transpose of a matrix.

The syntax of the TRANSPOSE CALL routine is

CALL TRANSPOSE (*X*, *Y*) ;

where

X specifies an input matrix with dimensions $m \times n$ (that is, $X[m, n]$)

Y specifies an output matrix with dimensions $n \times m$ (that is, $Y[n, m]$)

ZEROMATRIX CALL Routine

The ZEROMATRIX CALL routine replaces all of the element values of the numeric input matrix with 0. You can use the ZEROMATRIX CALL routine with multidimensional numeric arrays.

The syntax of the ZEROMATRIX CALL routine is

CALL ZEROMATRIX (*X*) ;

where

X specifies a numeric input matrix.

Create Design Matrix

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcdes.sas.)

PROC MCMC does not support a CLASS statement; therefore you need to construct the right design matrix (with dummy or indicator variables) prior to calling PROC MCMC. The best tool to use is the TRANSREG procedure (see Chapter 126, "The TRANSREG Procedure"). This procedure offers both indicator and effects coding methods. You can specify any categorical variables in the CLASS expansion, and use the ZERO= option to select a reference category. You can also specify any other data set variables (predictors, the responses, and so on) to the output data set in the ID statement.

For example, the following statements create a data set that contains two categorical variables (City and G), and two continuous variables (x and resp):

```
title 'Create Design Matrix';
data categorical;
   input City$ G$ x resp @@;
   datalines;
Chicago F 69.0 112.5  Chicago F 56.5  84.0
Chicago M 65.3  98.0  Chicago M 59.8  84.5
NewYork M 62.8 102.5  NewYork M 63.5 102.5
NewYork F 57.3  83.0  NewYork M 57.5  85.0
;
```

Suppose you are interested in creating a design matrix that uses dummy variable coding for the categorical variables City, G and their interaction City * G. You can use the following PROC TRANSREG statements:

```
proc transreg data=categorical design;
   model class(city g city*g / zero=last);
   id x resp;
   output out=input_mcmc(drop=_: Int:);
run;
```

The DESIGN option specifies that the primary goal is to code the design matrix. The MODEL statement indicates the variable of interest. The CLASS option in the MODEL statement expands the variables of interest to a list of "dummy" variables. The ZERO=LAST option sets the reference level. The ID statement includes x and resp in the OUT= data set. And the OUTPUT statement creates a new data set Input_MCMC that stores the design matrix and original variables from the original data set.

A quick call of the PRINT procedure shows the output from the PROC TRANSREG call:

```
proc print data=input_mcmc;
run;
```

Figure 80.15 prints the design matrix that is generated by PROC TRANSREG. The Input_mcmc data set contains all the variables from the original Categorical data set, in addition to corresponding dummy variables (CityChicago, GF, and CityChicagoGF) for the categorical variables.

Figure 80.15 Design Matrix Generated by PROC TRANSREG

Create Design Matrix

Obs	CityChicago	GF	CityChicagoGF	City	G	x	resp
1	1	1	1	Chicago	F	69.0	112.5
2	1	1	1	Chicago	F	56.5	84.0
3	1	0	0	Chicago	M	65.3	98.0
4	1	0	0	Chicago	M	59.8	84.5
5	0	0	0	NewYork	M	62.8	102.5
6	0	0	0	NewYork	M	63.5	102.5
7	0	1	0	NewYork	F	57.3	83.0
8	0	0	0	NewYork	M	57.5	85.0

You can now proceed to call PROC MCMC using this input data set Input_mcmc and the corresponding dummy variables.

PROC TRANSREG automatically creates a macro variable, &_TRGIND, which contains a list of variable names that it creates. The `%put &_trgind;` statement prints the following:

```
CityChicago GF CityChicagoGF
```

The macro variable &_TRGIND can come handy if you want to build a regression model; you can refer to &_TRGIND in the following way:

```
proc mcmc data=input_mcmc;
   array data[5] 1 &_trgind x;
   array beta[5] beta0-beta4;
   ...;
   call mult(beta, data, mu);
   ...;
```

The first ARRAY statement defines a one-dimensional array of length 5, and it takes on five values: a constant 1 and variables CityChicago, GF, CityChicagoGF, and x. The second ARRAY statement defines an array of beta, which are the model parameters. Later in the program, you can use the CALL MULT function to calculate the regression mean and store the value in the symbol mu.

Modeling Joint Likelihood

PROC MCMC assumes that the input observations are independent and that the joint log likelihood is the sum of individual log-likelihood functions. You specify the log likelihood of one observation in the MODEL statement. PROC MCMC evaluates that function for each observation in the data set and cumulatively sums them up. If observations are not independent of each other, this summation produces the incorrect log likelihood.

There are two ways to model dependent data. You can either use the DATA step LAG function or use the PROC option JOINTMODEL. The LAG function returns values of a variable from a queue. As PROC MCMC steps through the data set, the LAG function queues each data set variable, and you have access to the current value as well as to all previous values of any variable. If the log likelihood for observation x_i depends only on observations 1 to i in the data set, you can use this SAS function to construct the log-likelihood function for each observation. Note that the LAG function enables you to access observations from different rows, but the log-likelihood function in the MODEL statement must be generic enough that it applies to all observations. See "Example 80.14: Time Independent Cox Model" on page 6451 and "Example 80.15: Time Dependent Cox Model" on page 6458 for how to use this LAG function.

A second option is to create arrays, store all relevant variables in the arrays, and construct the joint log likelihood for the entire data set instead of for each observation. Following is a simple example that illustrates the usage of this option. For a more realistic example that models dependent data, see "Example 80.14: Time Independent Cox Model" on page 6451 and "Example 80.15: Time Dependent Cox Model" on page 6458.

```
/* allocate the sample size. */
data exi;
   call streaminit(17);
   do ind = 1 to 100;
      y = rand("normal", 2.3, 1);
      output;
   end;
run;
```

The log-likelihood function for each observation is as follows:

$$\log(f(y_i|\mu,\sigma)) = \log(\phi(y_i;\mu,\mathrm{var}=\sigma^2))$$

The joint log-likelihood function is as follows:

$$\log(f(\mathbf{y}|\mu,\sigma)) = \sum_i \log(\phi(y_i;\mu,\mathrm{var}=\sigma^2))$$

The following statements fit a simple model with an unknown mean (mu) in PROC MCMC, with the variance in the likelihood assumed known. The MODEL statement indicates a normal likelihood for each observation y.

```
proc mcmc data=exi seed=7 outpost=p1;
   parm mu;
   prior mu ~ normal(0, sd=10);
   model y ~ normal(mu, sd=1);
run;
```

The following statements show how you can specify the log-likelihood function for the entire data set:

```
data a;
run;

proc mcmc data=a seed=7 outpost=p2 jointmodel;
   array data[1] / nosymbols;
   begincnst;
      rc = read_array("exi", data, "y");
      n = dim(data, 1);
   endcnst;

   parm mu;
   prior mu ~ normal(0, sd=10);
   ll = 0;
   do i = 1 to n;
      ll = ll + lpdfnorm(data[i], mu, 1);
   end;
   model general(ll);
run;
```

The JOINTMODEL option indicates that the function used in the MODEL statement calculates the log likelihood for the entire data set, rather than just for one observation. Given this option, PROC MCMC no longer steps through the input data during the simulation. Consequently, you can no longer use any data set variables to construct the log-likelihood function. Instead, you store the data set in arrays and use arrays instead of data set variables to calculate the log likelihood.

The ARRAY statement allocates a temporary array (data). The READ_ARRAY function selects the y variable from the exi data set and stores it in the data array. See the section "READ_ARRAY Function" on page 6249. In the programming statements, you use a DO loop to construct the joint log likelihood. The expression ll in the GENERAL function now takes the value of the joint log likelihood for all data.

You can run the following statements to see that two PROC MCMC runs produce identical results.

```
proc compare data=p1 compare=p2;
   var mu;
run;
```

Access Lag and Lead Variables

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmclag.sas.)

There are two types of random variables in PROC MCMC that are indexed: the response (MODEL statement) is indexed by observations, and the random effect (RANDOM statement) is indexed by the SUBJECT= option variable. As the procedure steps through the input data set, the response or the random-effects symbols are filled with values of the current observation or the random-effects parameters in the current subject. Often you might want to access lag or lead variables across an index. For example, the likelihood function for y_i can depend on y_{i-1} in an autoregressive model, or the prior distribution for μ_j can depend on μ_k, where $k \neq j$, in a dynamic linear model. In these situations, you can use the following rules to construct symbols to access values from other observations or subjects:

rv.*Li*: the *i*th lag of the variable rv. This looks back to the past.

rv.*Ni*: the *i*th lead of the variable rv. This looks forward to the future.

The construction is allowed for random variables that are associated with an index, either a response variable or a random-effects variable. You concatenate the variable name, a dot, either the letter *L* (for "lag") or the letter *N* (for "next"), and a lag or a lead number. PROC MCMC resolves these variables according to the indices that are associated with the random variable, with respect to the current observation.

For example, the following RANDOM statement specifies a first-order Markov dependence in the random effect mu that is indexed by the subject variable time:

```
random mu ~ normal(mu.l1,sd=1) subject=time;
```

This corresponds to the prior

$$\mu_t \sim \text{normal}(\mu_{t-1}, \text{sd} = 1)$$

At each observation, PROC MCMC fills in the symbol mu with the random-effects parameter μ_t that belongs to the current cluster *t*. To fill in the symbol mu.l1, the procedure looks back and finds a lag-1 random-effects parameter, μ_{t-1}, from the last cluster *t-1*. As the procedure moves forward in the input data set, these two symbols are constantly updated, as appropriate.

When the index is out of range, such as *t*–1 when *t* is 1, PROC MCMC fills in the missing state from the ICOND= option in either the MODEL or RANDOM statement. The following example illustrates how PROC MCMC fills in the values of these lag and lead variables as it steps through the data set.

Assume that the random effect mu has five levels, indexed by sub $= \{\text{a}, \text{b}, \text{c}, \text{d}, \text{e}\}$. The model contains two lag variables and one lead variable (mu.l1, mu.l2, and mu.n2):

```
mn = (mu.l1 + mu.l2 + mu.n2) / 3
random mu ~ normal(mn, sd=1) subject=time icond=(alpha beta gamma kappa);
```

In this setup, instead of a list of five random-effects parameters that the variable mu can be assigned values to, there is now a list of nine variables for the variables mu, mu.l1, mu.l2, and mu.n2. The list lines up in the following order:

$$\text{alpha}, \text{beta}, \mu_\text{a}, \mu_\text{b}, \mu_\text{c}, \mu_\text{d}, \mu_\text{e}, \text{gamma}, \text{kappa}$$

PROC MCMC finds relevant symbol values according to this list, as the procedure steps through different subject cluster. The process is illustrated in Table 80.46.

Table 80.46 Processing Lag and Lead Variables

sub	mu	mu.l2	mu.l1	mu.n2
a	μ_a	alpha	beta	μ_c
b	μ_b	beta	μ_a	μ_d
c	μ_c	μ_a	μ_b	μ_e
d	μ_d	μ_b	μ_c	gamma
e	μ_e	μ_c	μ_d	kappa

For observations in cluster *a*, PROC MCMC sets the random-effects variable mu to μ_a, looks back two lags and fills in mu.l2 with alpha, looks back one lag and fills in mu.l1 with beta, and looks forward two leads and

fills in mu.n2 with μ_c. As the procedure moves to observations in cluster b, mu becomes μ_b, mu.l2 becomes beta, mu.l1 becomes μ_a, and mu.n2 becomes μ_d. For observations in the last cluster, cluster e, mu becomes μ_e, mu.l2 becomes μ_c, mu.l1 becomes μ_d, and mu.n2 is filled with kappa.

The following example fits a simple first-order dynamic linear model, in which the data set contains a time index and the response variable y:

```
data dlm;
   input time y;
   datalines;
1   1.353412529
2   4.840739953
3   1.604892523
4   6.8947921
5   3.509644288
6   4.020173553
7   3.842884451
8   4.49057276
9   2.204570502
10   4.007351323
11   2.005515044
12   2.781756057
;
```

You can fit the following model to the data:

$$
\begin{aligned}
Y_t &\sim \text{normal}(\mu_t, \text{var}{=}\sigma_y^2) \\
\mu_t &\sim \text{normal}(\mu_{t-1}, \text{var}{=}\sigma_\mu^2) \\
\mu_0 &= \alpha \\
\alpha &\sim \text{normal}(0, \text{var}{=}10) \\
\sigma_y^2, \sigma_\mu^2 &\sim \text{igamma}(\text{shape}{=}3, \text{scale}{=}2)
\end{aligned}
$$

The following PROC MCMC statements estimate parameters from this dynamic linear model:

```
proc mcmc data=dlm outpost=dlmO nmc=20000 seed=23;
   ods select PostSumInt;
   parms alpha 0;
   parms var_y 1 var_mu 1;
   prior alpha ~ n(0, sd=10);
   prior var_y var_mu ~ igamma(shape=3, scale=2);
   random mu ~ n(mu.l1,var=var_mu) s=time icond=(alpha) monitor=(mu);
   model y~n(mu, var=var_y);
run;
```

The key component is the mu.l1 specification in the RANDOM statement, where the prior for mu depends on its lag-1 value. The ICOND= option specifies the initial condition of mu and assigns it to be alpha, which is a parameter in the model.

Figure 80.16 lists the estimated posterior statistics for the parameters.

Figure 80.16 Posterior Summary Statistics of the Dynamic Linear Model

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
alpha	20000	2.6498	1.2924	-0.0555	5.0366
var_y	20000	1.7400	0.8337	0.5651	3.3498
var_mu	20000	0.8299	0.5720	0.2109	1.9582
mu_1	20000	2.6899	0.9253	0.8380	4.5390
mu_2	20000	3.4175	0.7622	1.9336	4.9541
mu_3	20000	3.3820	0.7431	1.8543	4.7950
mu_4	20000	4.3364	0.8297	2.7406	6.0198
mu_5	20000	3.9308	0.7194	2.4967	5.3248
mu_6	20000	3.8642	0.7348	2.4117	5.3832
mu_7	20000	3.7716	0.7399	2.3019	5.1525
mu_8	20000	3.6754	0.7316	2.1683	5.0342
mu_9	20000	3.1796	0.7209	1.7485	4.6257
mu_10	20000	3.2237	0.7355	1.7802	4.6253
mu_11	20000	2.8074	0.7839	1.2985	4.3701
mu_12	20000	2.8006	0.8758	1.1286	4.5473

Compartment Models

Pharmacokinetics (PK) is a branch of medicine that models the movement of a drug through the body (Gabrielsson and Weiner 2006). PK is sometimes referred to as the study of what the body does to a drug. Compartment models are basic building blocks of PK models. In a study, a body is divided in to several *compartments*, groups of organs or tissues that are kinetically homogeneous. The main interest of PK is to model how a drug moves through these compartments—for example, to estimate the amount of a drug and further concentrations of the drug that are present in a compartment at any given time. The concentrations of the drug, over time, are typically modeled via a set of differential equations that depend on a variety of variables, such as the amount of drug given, elimination rates, transfer rates, and so on. The sets of differential equations can also depend on how the drug is administered to the body, distributed through the body, and eliminated from the body. These models are known as compartment models, which are divided based on the number of compartments. Not all analytical solutions to multiple-compartment differential equations models are known, but Abuhelwa, Foster, and Upton (2015) and Fisher and Shafer (2007) provide closed-form solutions for those that correspond to the one-, two-, and three- compartments. These are the models that are handled by the CMPTMODEL statement.

Routes of Drug Administration

There are three types of drug administration methods: intravenous bolus, intravenous infusion, and extravascular dose administrations. A bolus medication typically has no or a very short time lag for the drug to enter a compartment. Intravenous infusions are given over a period of time, and the drug enters into the body at a constant rate. In infusion, the two quantities, *rate* and *duration* of the infusion, are of interest and are used in calculating the amount of drug present in the body. The rate and the duration of the infusion are related, so

knowing one determines the other. For compartment models that have infusion type of administration, it is sufficient to provide either the rate or the duration information.

One-, Two-, and Three-Compartment Models for Intravenous Administration

This section lists a number of assumptions that are made on the set of compartment models that the CMPTMODEL statement supports. The one-, two-, and three-compartment models all have a central compartment and can have one or more peripheral compartments that are linked only to the central compartment but not to each other. After a drug is administered to an administration site, it is distributed to the central compartment and then to other peripheral compartments. The rates at which the drug moves from the central compartment to and from the other peripheral compartments are characterized by transfer rate constants. The rates at which the drug is eliminated from the central or the peripheral compartments are characterized by elimination rate constants.

Schematic representation of the one-, two-, and three- compartment models are given in Figure 80.17, Figure 80.18, and Figure 80.19, respectively. In each figure, Dose is the dosage of the drug that is given intravenously, and k_{10} represents the rate at which the drug is eliminated from compartment 1. In Figure 80.18 and Figure 80.19, k_{12} represents the transfer rate constant from compartment 1 to compartment 2 and k_{21} represents the transfer rate constant from compartment 2 to compartment 1. Similarly, in Figure 80.19, k_{13} represents the transfer rate constant from compartment 1 to compartment 3 and k_{31} represents the transfer rate constant from compartment 3 to compartment 1. In these scenarios, compartment 1 is the central compartment and compartments 2 and 3 are the peripheral compartments. In compartment models, a drug can be administered only to and eliminated only from the central compartment.

Figure 80.17 One-Compartment Model

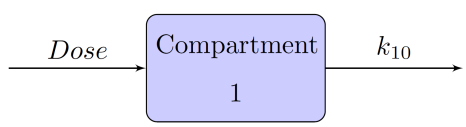

Figure 80.18 Two-Compartment Model

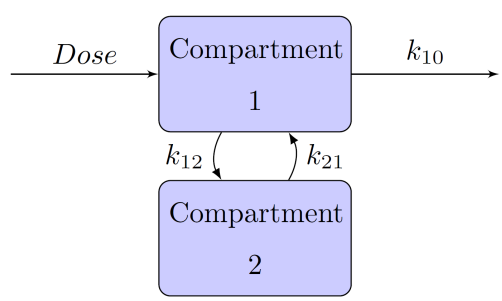

Figure 80.19 Three-Compartment Model

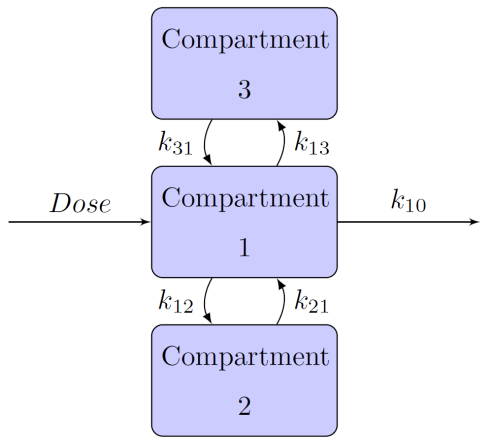

One-, Two-, and Three-Compartment Models for Extravascular Administration

Extravascular administrations, such as oral administration, are different from bolus and infusion in the sense that there is an absorption phase before the drug enters the central compartment. Schematic representation of the one-, two-, and three- compartment models when the drug is administered using an oral method are given in Figure 80.20, Figure 80.21, and Figure 80.22, respectively. In these models, compartment 0 is also called the *depot* compartment.

Figure 80.20 One-Compartment Model with Absorption Phase

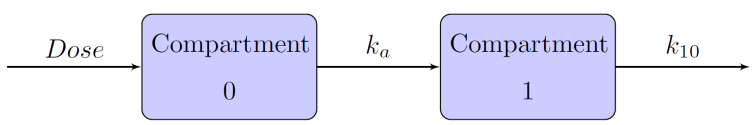

Figure 80.21 Two-Compartment Model with Absorption Phase

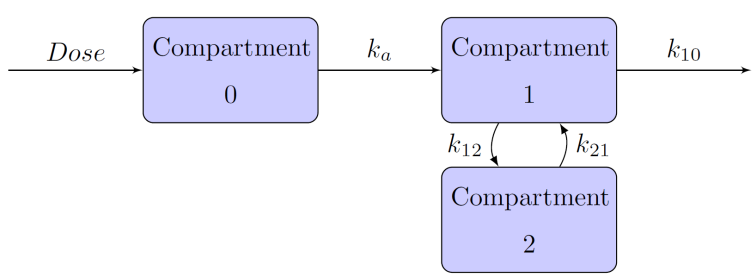

Figure 80.22 Three-Compartment Model with Absorption Phase

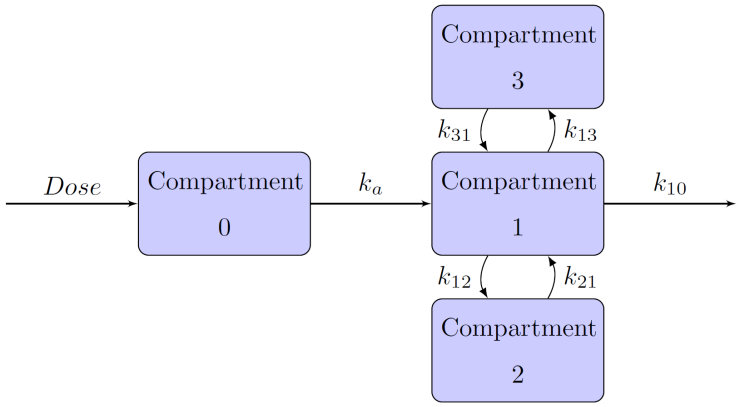

Compartment Models Specification

The CMPTMODEL statement in PROC MCMC enables you to fit one-, two-, and three-compartment models that use bolus, infusion, or oral types of drug administration. The CMPTMODEL statement computes the predicted concentrations of the drug for all compartments at each time point.

The CMPTMODEL statement supports a large number of options, which are grouped into three types: required, conditionally-required, and optional options. You use the required options to specify the type of compartment model that you want to fit (the NCOMPS=, ADMTYPE=, and PARMTYPE= options) and to provide the input time variable (in the TIME= option) and the outcome variable (in the PCONC= option) that is the predicted concentration in the first (central) compartment at each time point.

The statement supports the following nine types of compartment models:

1. one-compartment model with bolus dose administration
2. one-compartment model with infusion type of dose administration
3. one-compartment model with oral dose administration
4. two-compartment model with bolus dose administration
5. two-compartment model with infusion type of dose administration
6. two-compartment model with oral dose administration
7. three-compartment model with bolus dose administration
8. three-compartment model with infusion type of dose administration
9. three-compartment model with oral dose administration.

Each of these nine models can be parameterized in two ways: in terms of elimination and transfer rate constants of each compartment, or in terms of clearance and volume parameters of each compartment. The relationship between rate constants and the clearance and volumes of each compartment are:

$$K_{10} = CL1/VOL1$$
$$K_{12} = CL2/VOL1$$
$$K_{13} = CL3/VOL1$$
$$K_{21} = CL2/VOL2$$
$$K_{31} = CL3/VOL3$$

where CL*n* and VOL*n* are the clearance and volume of the *n*th compartment for *n*=1,2,3. You can use the PARMTYPE= option in the CMPTMODEL statement to specify these parameterizations. In total, you can specify 18 types of compartment models by using combinations of the NCOMPS=, ADMTYPE=, and PARMTYPE= options in the CMPTMODEL statement.

The conditionally required options define the specifics needed in some of the compartment models. For example, if you specify a two-compartment model with a bolus type of drug administration in terms of rate constants by using NCOMPS=2, ADMTYPE=IVB, and PARMTYPE=1, then you must specify the K12=, K21=, and K10= options. Here is an example:

```
PROC MCMC data=TwoComp;

...other statements...

erate10= exp(beta1);
trate12= exp(beta2);
trate21= exp(beta3);

CMPTMODEL ncomps=2 admtype=ivb parmtype=1 pconc=pred time=time
          k10=erate10 k12=trate12 k21=trate21
          dose1=ddose scale1=vol/1000;
model conc ~ normal(pred,var=s2);

...other statements...

run;
```

The K12= and K21= options specify the transfer rate constants, and the K10= option specifies the elimination rate constant from the central compartment. In addition, you can specify dosage information (DOSE*n*=) or scaling information (SCALE*n*=) in the syntax. For more information, see the section "CMPTMODEL Statement" on page 6251.

Table 80.47, Table 80.48, and Table 80.49 list all conditionally required options and valid optional options for each of the 18 compartment models.

Table 80.47 One-Compartment Models

Model	Required	Conditionally Required	Optional
One compartment with bolus dose	NCOMPS=1 ADMTYPE=IVB PARMTYPE=1	K10=	DOSE1= SCALE1=
One compartment with bolus dose	NCOMPS=1 ADMTYPE=IVB PARMTYPE=2	CL1= VOL1=	DOSE1= SCALE1=
One compartment with infusion dose	NCOMPS=1 ADMTYPE=INF PARMTYPE=1	K10= RATE= (or) DURN=	DOSE1= SCALE1=
One compartment with infusion dose	NCOMPS=1 ADMTYPE=INF PARMTYPE=2	CL1= VOL1= RATE= (or) DURN=	DOSE1= SCALE1=
One compartment with oral dose	NCOMPS=1 ADMTYPE=ORAL PARMTYPE=1	K10= Ka=	DOSE1= SCALE1=
One compartment with oral dose	NCOMPS=1 ADMTYPE=ORAL PARMTYPE=2	CL1= VOL1= Ka=	DOSE1= SCALE1=

Table 80.48 Two-Compartment Models

Model	Required	Conditionally Required	Optional
Two compartments with bolus dose	NCOMPS=2 ADMTYPE=IVB PARMTYPE=1	K10= K12= K21=	DOSE1= DOSE2= SCALE1= SCALE2= K20= PCONC2=
Two compartments with bolus dose	NCOMPS=2 ADMTYPE=IVB PARMTYPE=2	CL1= VOL1= CL2= VOL2=	DOSE1= DOSE2= SCALE1= SCALE2= K20= PCONC2=
Two compartments with infusion dose	NCOMPS=2 ADMTYPE=INF PARMTYPE=1	K10= K12= K21= RATE= (or) DURN=	DOSE1= SCALE1=
Two compartments with infusion dose	NCOMPS=2 ADMTYPE=INF PARMTYPE=2	CL1= VOL1= CL2= VOL2= RATE= (or) DURN=	DOSE1= SCALE1=
Two compartments with oral dose	NCOMPS=2 ADMTYPE=ORAL PARMTYPE=1	K10= K12= K21= Ka=	DOSE1= DOSE2= SCALE1= SCALE2= K20= PCONC2=
Two compartments with oral dose	NCOMPS=2 ADMTYPE=ORAL PARMTYPE=2	CL1= VOL1= CL2= VOL2= Ka=	DOSE1= DOSE2= SCALE1= SCALE2= K20= PCONC2=

Table 80.49 Three-Compartment Models

Model	Required	Conditionally Required	Optional
Three compartments with bolus dose	NCOMPS=3 ADMTYPE=IVB PARMTYPE=1	K10= K12= K21= K13= K31=	DOSE1= DOSE2= DOSE3= SCALE1= SCALE2= SCALE3= K20= K30= PCONC2= PCONC3=
Three compartments with bolus dose	NCOMPS=3 ADMTYPE=IVB PARMTYPE=2	CL1= VOL1= CL2= VOL2= CL3= VOL3=	DOSE1= DOSE2= DOSE3= SCALE1= SCALE2= SCALE3= K20= K30= PCONC2= PCONC3=
Three compartments with infusion dose	NCOMPS=3 ADMTYPE=INF PARMTYPE=1	K10= K12= K21= K13= K31= RATE= (or) DURN=	DOSE1= SCALE1=
Three compartments with infusion dose	NCOMPS=3 ADMTYPE=INF PARMTYPE=2	CL1= VOL1= CL2= VOL2= CL3= VOL3= RATE= (or) DURN=	DOSE1= SCALE1=
Three compartments with oral dose	NCOMPS=3 ADMTYPE=ORAL PARMTYPE=1	K10= K12= K21= K13= K31= Ka=	DOSE1= DOSE2= DOSE3= SCALE1= SCALE2= SCALE3= K20= K30= PCONC2= PCONC3=
Three compartments with oral dose	NCOMPS=3 ADMTYPE=ORAL PARMTYPE=2	CL1= VOL1= CL2= VOL2= CL3= VOL3= Ka=	DOSE1= DOSE2= DOSE3= SCALE1= SCALE2= SCALE3= K20= K30= PCONC2= PCONC3=

One-Compartment Model with an Oral Dose Administration

Consider the example "Example 80.22: One-Compartment Model with Pharmacokinetic Data" on page 6503, which studies the dispersion of the theophylline drug through a living individual. In this example, Pinheiro and Bates (1995) considered an one-compartment model with an oral dose. The paper mentions the solution to the one-compartment model. Instead of using the explicit solution, you can use the CMPTMODEL statement to fit the same model as follows:

```
proc mcmc data=theoph nmc=10000 seed=27 outpost=theophO diag=none;
   array b[2];
   array cov[2,2];
   array S[2,2] (1 0 0 1);
```

```
    parms beta1=-3.22 beta2=0.47 beta3=-2.45;
    parms cov {0.03 0 0 0.4};
    parms s2y;
    prior beta: ~ normal(0, sd=100);
    prior cov ~ iwish(2, S);
    prior s2y ~ igamma(shape=3, scale=2);
    random b ~ mvn(muB, cov) subject=subject;

    cl = exp(beta1 + b1);
    ka = exp(beta2 + b2);
    ke = exp(beta3);
    vl = cl/ke;

    CMPTMODEL ncomps=1 admtype=oral time=time pconc=predConc
              parmtype=1 ka=ka k10=ke dose0=dose scale1=vl;

    model conc ~ normal(predConc,var=s2y);
run;
```

The NCOMPS=1 and ADMTYPE=ORAL options specify a one-compartment model with oral administration. Time is a data set variable that indicates time, and predConc is an outcome variable that contains predicted concentration. The PARMTYPE=1 option requests the compartment model using absorption and elimination rate constants, with KA=ka and K10=ke indicating the absorption and elimination rate constants, respectively. The dosage value for each patient in the depot compartment is specified in DOSE0=option, where dose is a data set variable. Lastly, SCALE1=vl scales the predicted concentrations by vl.

Multiple Doses

In PK field experiments or observational studies, a patient can often receive a drug multiple times, either continually or periodically over a period of time. The patient can also receive multiple types of drug in a study. For example, a patient might receive a bolus injection in the morning and an infusion drug at a constant rate in the evening. Calculation of predicted concentration in each compartment in the presence of these multiple doses or multiple types of dosing is different from single-dosing compartment models. For differential equations of a number of multiple-dosage scenarios and solutions to predicted concentration for the central compartment, see Gabrielsson and Weiner (2006).

The CMPTMODEL statement handles multiple doses in one-, two-, or three-compartment models and computes predicted concentrations in the central compartment. The syntax specification does not change from single-dose models, and the structure and content of the input data set are understood by the statement to fit various types of multiple-dosage models.

Within the biopharmaceutical industry, data in multiple-dosage studies are often structured by following a convention that has been popularized by NONMEM software (Beal et al. 2011). This convention is used to name variables and specify variable values; see Owen and Fiedler-Kelly (2014). If your data are available in a SAS data set that follows this convention, the data set must be converted to a SAS data set that is suitable to be analyzed using PROC MCMC. You can use the autocall macro %PKCONVRT for this purpose.

The %PKCONVRT macro takes two arguments: an input data set (specified by the first argument) and an output data set (specified by the second argument). The organization of the input data set should follow the convention. The output data set can serve as the DATA= data set for PROC MCMC to fit single-dose or multiple-dose models. Here is a simple example:

```
data pk_ex;
input ID  TIME  AMT DV  EVID;
datalines;
1     0.00    60000        .       1
1     0.20    50000        .       1
1     0.25        0    1126.1      0
1     0.50        0     869.9      0
1     0.75        0     883.6      0
1     1.00        0    1244.0      0
1     1.50        0     995.2      0
2     0.00    70000        .       1
2     0.25        0    1126.1      0
2     0.50        0     869.9      0
2     0.75        0     883.6      0
2     1.00        0    1244.0      0

   .... more lines ...
;
run;

%pkconvrt(data=pk_ex, out=out_ex);
```

If you want to fit a two-compartment model for the preceding multiple dosage data, use the following MCMC program with the out_ex data set that is produced from the %PKCONVRT macro:

```
proc mcmc data=out_ex nmc=10000 seed=17071 outpost=post_out;
   array b[4] b1 b2 b3 b4;
   array b0[4] (0 0 0 0);
   array Sigma[4,4];
   array S[4,4] (1 0 0 0, 0 1 0 0, 0 0 1 0, 0 0 0 1);

   parms beta1=2 beta2=2 beta3=2 beta4=2 s2=0.6 Sigma;
   prior beta: ~n(0, sd=10000);
   prior s2 ~ igamma(shape=3, scale=2);
   prior Sigma ~ iwish(4, S);

   random b ~ mvn(b0, Sigma) subject=id;
   cl1 = exp(beta1+b1);
   vl1 = exp(beta2+b2);
   cl2 = exp(beta3+b3);
   vl2 = exp(beta4+b4);

   CMPTMODEL ncomps=2 admtype=ivb parmtype=2 pconc=pred time=time
      vol1=vl1 cl1=cl1 vol2=vl2 cl2=cl2 scale1=vl1;
   model conc ~ normal(pred,var=s2);
run;
```

The CMPTMODEL statement computes the predictions only for the central compartment for multiple dosage data. Therefore, options PCONC2=, PCONC3=, and PCONC0= are not valid in a multiple dose model. In addition, the K20=, K30=, DOSE2=, DOSE3=, SCALE2=, SCALE3=, SCALE0= options are ignored.

The %PKCONVRT autocall macro computes the elapsed time (from the time that a dose is administered to the time that a concentration is measured) in scenarios for both single and multiple continuous doses. The macro writes the computed time in the data set that is specified in the OUT= option. This data set also

contains all the dosage information for all the time points at which the concentrations are measured. When you specify this data set in the PROC MCMC statement and you specify a CMPTMODEL statement, the elapsed times and the dosage information from the data set are used directly in computing the predicted concentrations in the central compartment. As a result, the CMPTMODEL statement ignores any specified DOSE1= and TIME= options. Similarly, for absorption models (ADMTYPE = ORAL), the DOSE0= and TIME= options in the CMPTMODEL statement are ignored and overwritten with the elapsed times and the dosage information from the OUT= data set. For examples that use the %PKCONVRT autocall macro and the CMPTMODEL statement to fit various compartment models, see Kurada and Chen (2018).

In addition, the CMPTMODEL statement supports steady-state dosing scenarios (Owen and Fiedler-Kelly 2014) in one-, two-, and three-compartment models. The %PKCONVRT macro keeps track of the elapsed time and dosing information that are required in the CMPTMODEL statement to fit steady-state dose models. The %PKCONVRT autocall macro supports input data sets that have combinations of steady-state and regular dosing, single and multiple dosing. For example, your data set can contain dosing history information on patients who receive multiple regular bolus doses and others who receive a single steady-state dose. When you use the output data set from the %PKCONVRT macro as input to PROC MCMC, the CMPTMODEL statement properly interprets all these distinct scenarios and computes the predicted concentrations accordingly.

To use the %PKCONVRT autocall macro in conjunction with PROC MCMC for a Bayesian analysis of pharmacokinetic data, specify the DOSEDROP=TRUE option. This option deletes dosing records and keeps measurement records only. By default, the DOSEDROP= option is set to FALSE and the %PKCONVRT macro keeps all records, including dosing records, from the input data set. In these records, the response variable DV is filled with missing values. By default, PROC MCMC treats all missing response values as parameters, draws samples, and incorporates them in the posterior inference. Imputing missing concentrated values at dosing time will change the posterior distribution and lead to a different conclusion than an analysis that uses only observations from measurement records. The DOSEDROP=TRUE option ensures the deletion of the dosing records.

The %PKCONVRT macro creates new variables (such as the _CMPT_ variable) in the OUT= data set. According to SAS conventions, it is best to avoid having variables whose name begins with an underscore _ in the DATA = data set. Also programming variables in the MCMC program that start with _CMPT_ are reserved for the internal computations; hence, avoid programming variables names that begin with _CMPT_ .

CALL ODE and CALL QUAD Subroutines

The CALL ODE subroutine numerically solves a set of first-order ordinary differential equations (ODEs), including piecewise differential equations. The CALL QUAD subroutine calculates multidimensional integrand. You can use them as programming statements in PROC MCMC. These subroutines require that you define an objective function, for either a set of simultaneous differential equations or an integrand function, by using PROC FCMP (see the FCMP procedure in the *Base SAS Procedures Guide*) and call these subroutines in PROC MCMC.

CALL ODE

The CALL ODE subroutine performs numerical integration of first-order vector differential equations of the form $\frac{dy}{dt} = f(t, y(t))$ over the subinterval $t \in [t_i, t_f]$ with the initial values $y(t_i) = y_0$. The subroutine can also be used to solve piecewise differential equations.

You specify the CALL ODE subroutine in PROC MCMC by using the following syntax:

```
CALL ODE("DeqFun", Soln, Init, ti, tf, G1, G2, ... , Gk <, ode_opt>);
```

DeqFun: the name of the PROC FCMP subroutine of a set of simultaneous set of ordinary differential equations, $\frac{dy}{dt} = f(t, y(t))$

Soln: an argument that contains solutions. Soln can be a numeric variable or an array. If it is an array, then the size of the array determines the dimension of the problem. Otherwise, the dimension of the problem is one.

Init: initial values of the variable y. Init can be either a numeric variable or an array. The dimension must match that of Soln.

ti: initial time value of the subinterval t

tf: final time value of the subinterval t

Gi: input arguments in the DeqFun subroutine

ode_opt: subroutine options. ode_opt is a positional array of size 4, with the following definition:

- ode_opt[1]: convergence accuracy. The default value is 1E–8.
- ode_opt[2]: minimum allowable step size in the integration process. The default value is 1E–12.
- ode_opt[3]: maximum allowable step size in the integration process. The default value is the largest double-precision floating-point number.
- ode_opt[4]: initial step size to start the integration process. The default value is 1E–5.

To specify the DeqFun subroutine by using PROC FCMP, use the following statements:

```
proc fcmp outlib=sasuser.funcs.ODE;
   subroutine DeqFun(t,y[*],dy[*], A1,A2,...,Ak);
   outargs dy;
   dy[1] = -A1*y[1];
   dy[2] = A2*y[1]-Ak-1*y[2];
   ...
   dy[n] = t*y[n] + Ak;
   endsub;
run;
```

The OUTLIB= option specifies an output data set to which the compiled subroutine is written. The first three arguments of the DeqFun subroutine must be (1) the time variable t, (2) the with-respect-to variable y, which can be an array, and (3) the differential equation function variable dy, which can also be an array. The remaining optional input arguments are variables that are required in the construction of the differential equations. You must declare variable dy as the updated variable in the OUTARGS statement.

To include the DeqFun in your PROC MCMC program, you must specify the SAS data set that contains the compiler subroutine:

```
options cmplib=sasuser.funcs;
```

See "Example 80.22: One-Compartment Model with Pharmacokinetic Data" on page 6503 for an example of fitting a pharmacokinetic model by using the CALL ODE subroutine.

The CALL ODE subroutine uses the polyalgorithm of Byrne and Hindmarsh (1975) to provide a numerical solution for the set of ordinary differential equations. Note that you can model any n-order differential equation as a system of first-order differential equations.

Piecewise Differential Equations

Suppose you have a system of piecewise ODEs of the following form:

$$\frac{d\mathbf{y}}{dt} = \begin{cases} f_1(t, \mathbf{y}(t)) & t_1 \le t < t_2 \\ f_2(t, \mathbf{y}(t)) & t_2 \le t < t_3 \\ \vdots & \vdots \\ f_{m-1}(t, \mathbf{y}(t)) & t_{m-1} \le t < t_m \\ f_m(t, \mathbf{y}(t)) & t_m \le t \end{cases}$$

The system of equations is defined over m intervals ($t \in [t_i, t_f]$) with initial values $y(t_j) = y_(t_{j0})$ for $j = 1, 2, \ldots, m$. The variable $\boldsymbol{y}(t)$ can be a vector. The CALL ODE subroutine solves the system of piecewise ODEs in each semiclosed interval $[t_i, t_j)$ for $i < j = 1, 2, \ldots, m$.

The CALL ODE subroutine differentiates between a regular ODE problem and a piecewise ODE problem based on the dimension of the second input argument to the subroutine. That is the Soln argument, which stores the solutions to the ODEs. If Soln is either a numeric variable or a one-dimensional array, the CALL ODE subroutine treats the problem as a regular ODE problem; if Soln is a two-dimensional array, the CALL ODE subroutine treats it as a piecewise ODE problem and solves accordingly.

You specify the CALL ODE subroutine in PROC MCMC to solve piecewise ODEs by using the following syntax:

```
CALL ODE("DeqFun", Soln, ., ti, tf, G1, G2, ... , Gk,
         ode_grid, "InitFun", H1, H2, ..., Hl <, ode_opt>);
```

DeqFun: the name of the PROC FCMP subroutine of a system of ODEs

Soln: an argument that contains solutions. Soln must be an m (number of intervals) $\times$ n (number of equations) array. Upon completion, the CALL ODE subroutine fills in the last row of the Soln array (Soln$[m, 1], \ldots,$ Soln$[m, n]$) with the final solutions.

. (period): the third positional argument takes a missing value, or a period (.). In a regular ODE problem, the third argument is reserved for initial values, which are fixed per solution. In a piecewise problem, initial values can depend on solutions to previous set of ODEs, and they are set through a separate PROC FCMP subroutine (see description of InitFun).

ti: initial time value of the subinterval t

tf: final time value of the subinterval t

Gi: input arguments in the DeqFun subroutine

ode_grid: numerical array that stores the interval boundaries. You must use the keyword ode_grid for this array, and the array should be placed after all the Gi variables. Elements in ode_grid must be sorted in ascending order. The dimension of this array should be *m*, the number of rows in Soln.

InitFun: the name of the PROC FCMP subroutine that specifies the initial values of the ODEs at different intervals. The subroutine requires an output array argument that should be filled with the required initial values for each interval. The InitFun subroutine should come after the ode_grid variable.

Hi: input arguments in the InitFun subroutine

ode_opt: subroutine options

The following example statements construct a two-interval piecewise ODE in the DeqFun subroutine, set initial values in the InitFun subroutine, and call the general ODE solver in PROC MCMC:

```
proc fcmp outlib=sasuser.funcs.ODE;
   subroutine DeqFun(t,y[*],dy[*],t1,beta,alpha);
   outargs dy;
   if (t <= t1) then do;
       dy[1] = -y[1];
       dy[2] = beta * y[1] - y[2];
       dy[3] = - y[3];
   end; else do;
       dy[1] = 0;
       dy[2] = alpha * y[1] + y[2];
       dy[3] = alpha * y[3] * y[3];
   end;
   endsub;
run;

proc fcmp outlib=sasuser.funcs.ODE;
   subroutine InitFun(init[*,*],d,Soln[*,*]);
   outargs init;
   init[1,1] = d;
   init[1,2] = d;
   init[1,3] = d;
   init[2,1] = Soln[1,1];
   init[2,2] = Soln[1,2];
   init[2,3] = Soln[1,3];
   endsub;
run;

options cmplib=sasuser.funcs;

proc mcmc ...;
   array Soln[2, 3];
   array ode_grid[2];

   ode_grid[1]=0;
   ode_grid[2]=ti;
   call ode("DeqFun", Soln, ., 0, tf, ti, beta, alpha,
            ode_grid, "InitFun", dose, Soln);
   ...;
run;
```

The DeqFun subroutine specifies three differential equations in two intervals, $t \leq t_2$ and $t > t_2$. The InitFun subroutine specifies the initial values for the first set of equations as d (an input variable) and specifies the initial values for the second set of equations as solutions to the first set of equations. In the PROC MCMC statements, Soln is an array of size 2×3. The ode_grid array has two elements: the first element is the lower bound (0 in this case), and the second element is ti (for example, a data set variable). The DeqFun and InitFun subroutines are passed to the ODE solver. Three variables (ti, beta, and alpha) are input arguments to the DeqFun subroutine. Two variables (dose and Soln) are input arguments to the InitFun subroutine. The solution array, Soln, is passed to the InitFun subroutine and enables the ODE solver to use solutions to the first set of ODEs as the initial values for the second set. The CALL ODE subroutine fills the second row of the Soln array (Soln[2,1], Soln[2,2], and Soln[2,3]) with the final solution to the entire system of the ODEs upon completion.

CALL QUAD

The CALL QUAD subroutine is a numerical integrator that performs the integration in a twofold fashion based on the dimension of the problem. In a unidimensional problem, the subroutine uses the adaptive Romberg type of integration techniques; in a multidimensional problem, it uses the Laplace approximation. You specify the CALL QUAD subroutine syntax in PROC MCMC by using the following syntax:

```
CALL QUAD("IntFun",Res, LL, UL, G1, G2, ... , Gk <, quad_init> <, quad_opt>);
```

IntFun: the name of the PROC FCMP subroutine of an integrand function

Res: contains the approximated integral value

LL: lower limit of the integral. LL must be an array if the problem is multidimensional.

UL: upper limit of the integral. UL must be an array if the problem is multidimensional.

Gi: input arguments in the IntFun subroutine

quad_init: starting values that are used in optimizing the IntFun function in Laplace approximation. This optional array is applicable only in a multidimensional problem, and you must use the keyword quad_init for this array. By default, the optimization starts at the average values of LL and UL. When you use the variable quad_init, it must precede the integration option array quad_opt.

quad_opt: integration options. You must use the keyword quad_opt for this array. For a unidimensional integration problem, quad_opt must be a positional array of size 4:

- quad_opt[1]: relative accuracy. The default value is 1E–7.
- quad_opt[2]: the approximate location of a maximum of the integrand. The default value is the centered location between the LL and UL limits.
- quad_opt[3]: approximate estimate of any scale in the integrand along the independent variables. The default value is 1.
- quad_opt[4]: the number of refinements allowed in order to achieve the required accuracy

In a multidimensional integration problem, quad_opt is a positional array of size 2:

- quad_opt[1]: relative accuracy. The default value is 1E–7.

- quad_opt[2]: optimization technique:
  - 1: Quasi-Newton
  - 2: Newton-Raphson
  - 3: Newton-Raphson with ridging
  - 4: Nelder-Mead simplex method

To specify the subroutine IntFun in PROC FCMP, you can use the following example:

```
proc fcmp outlib=sasuser.funcs.QUAD;
   subroutine IntFun(y[*],fy,pi);
   outargs fy;
   fy = (exp(-(y[1]*y[1]+y[2]*y[2])/2))/(2*pi);
   endsub;
run;
```

The first argument to IntFun must be the with-respect-to vector in the integral. The variable y can be a scalar or an array, depending on your problem. The second argument must be the variable that contains the integrand value, which should be declared as the updated variable in the OUTARGS statement. The remaining input arguments are optional variables, if needed in constructing the integrand function.

To include the IntFun subroutine in your PROC MCMC program, you must specify the SAS data set that contains the compiler subroutine:

```
options cmplib=sasuser.funcs;

proc mcmc ...;
   array LL[2] (-100 -100);
   array UL[2] ( 100  100);

   call quad("IntFun", Res, LL, UL, pi);
   ...;
run;
```

In the PROC MCMC statements, LL and UL are arrays that contain the lower and upper limits of the integral. The variable pi is an input argument to the IntFun subroutine. The solution is stored in Res.

In case of one-dimensional integration, the CALL QUAD subroutine uses adaptive Romberg-type integration techniques. (See Rice 1973; Sikorsky 1982; Sikorsky and Stenger 1984; Stenger 1973b, a.) Many adaptive numerical integration methods (Ralston and Rabinowitz 1978) start at one end of the interval and proceed toward the other end, working on subintervals while locally maintaining a certain prescribed precision. This is not the case with the CALL QUAD subroutine. The CALL QUAD subroutine is an adaptive global-type integrator that produces a quick, rough estimate of the integration result and then refines the estimate until it achieves the prescribed accuracy. This gives the subroutine an advantage over Gauss-Hermite and Gauss-Laguerre quadratures (Ralston and Rabinowitz 1978; Squire 1987), particularly for infinite and semi-infinite intervals, because those methods perform only a single evaluation.

The Laplace approximation is used in the multidimensional problem. To approximate a k-dimensional integration of an integrand, $h(t)$, between the limits a and b, you can use the approximation formula

$$\int_a^b h(t)dt \cong h(c)\sqrt{\frac{(2\pi)^k}{|H(t)|_{t=c}}}$$

where $-\log[h(t)]$ assumes a minimum over $[a, b]$ at an interior critical point c, and $H(t)$ is the Hessian of $-\log[h(t)]$,

$$H(t) = \frac{\partial^2 \{-\log(h(t))\}}{\partial t \, \partial t'}$$

The minimum of the negative log integrand is found by using numerical optimization. By default, the CALL QUAD subroutine uses the quasi-Newton method. You can select other methods by using the quad_opt option.

Regenerating Diagnostics Plots

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcren.sas.)

By default, PROC MCMC generates three plots: the trace plot, the autocorrelation plot, and the kernel density plot. Unless ODS Graphics is enabled before calling the procedure, it is hard to generate the same graph afterwards. Directly using the **Stat.MCMC.Graphics.TraceAutocorrDensity** template is not feasible. The easiest way to regenerate the same graph is with the %TADPlot autocall macro. The %TADPlot macro requires you to specify an input data set (which usually is the output data set from a previous PROC MCMC call) and a list of variables that you want to plot.

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

A simple regression example, with three parameters, is used here for illustrational purposes. For an explanation of the regression model and the data involved, see the section "Simple Linear Regression" on page 6215. The following statements fit a regression model:

```
ods select none;
proc mcmc data=sashelp.class nmc=50000 thin=5 outpost=classout seed=246810;
   parms beta0 0 beta1 0;
   parms sigma2 1;
   prior beta0 beta1 ~ normal(0, var = 1e6);
   prior sigma2 ~ igamma(3/10, scale = 10/3);
   mu = beta0 + beta1*height;
   model weight ~ normal(mu, var = sigma2);
run;
ods select all;
```

The output data set Classout contains posterior draws for beta0, beta1, and sigma2. It also stores the log of the prior density (LogPrior), log of the likelihood (LogLike), and the log of the posterior density (LogPost). If you want to examine the beta0 and LogPost variable, you can use the following statements to generate the graphs:

```
ods graphics on;
%tadplot(data=classout, var=beta0 logpost);
ods graphics off;
```

Figure 80.23 displays the regenerated diagnostics plots for variables beta0 and Logpost from the data set Classout.

Figure 80.23 Regenerated Diagnostics Plots for beta0 and Logpost

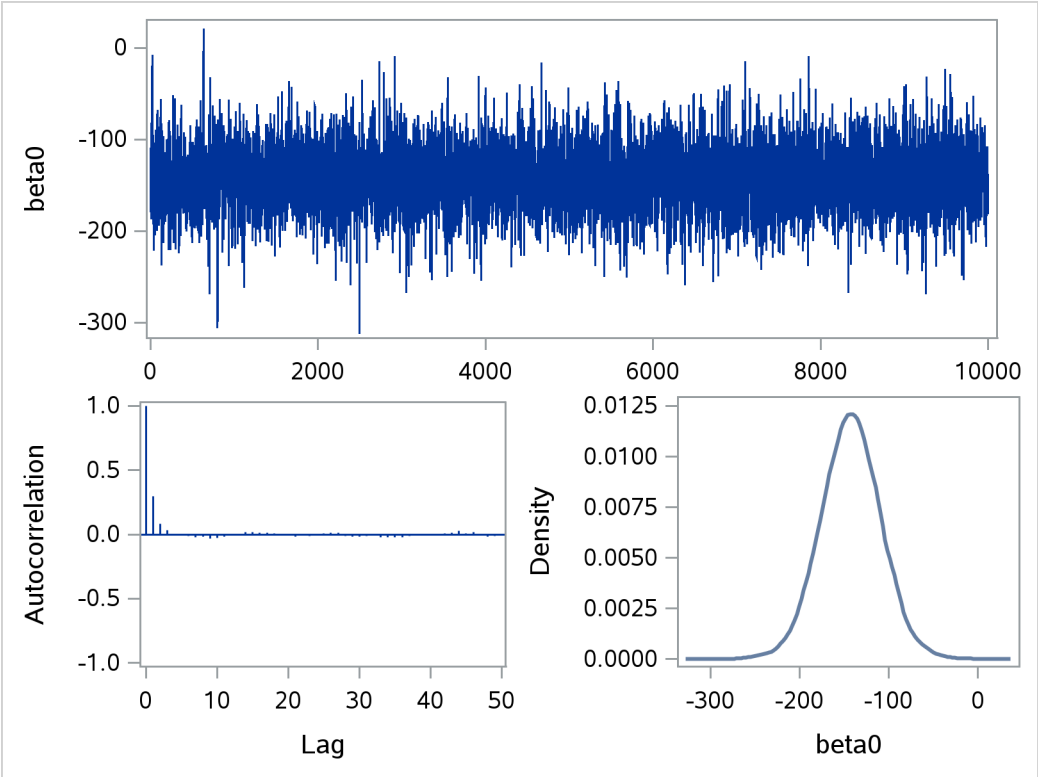

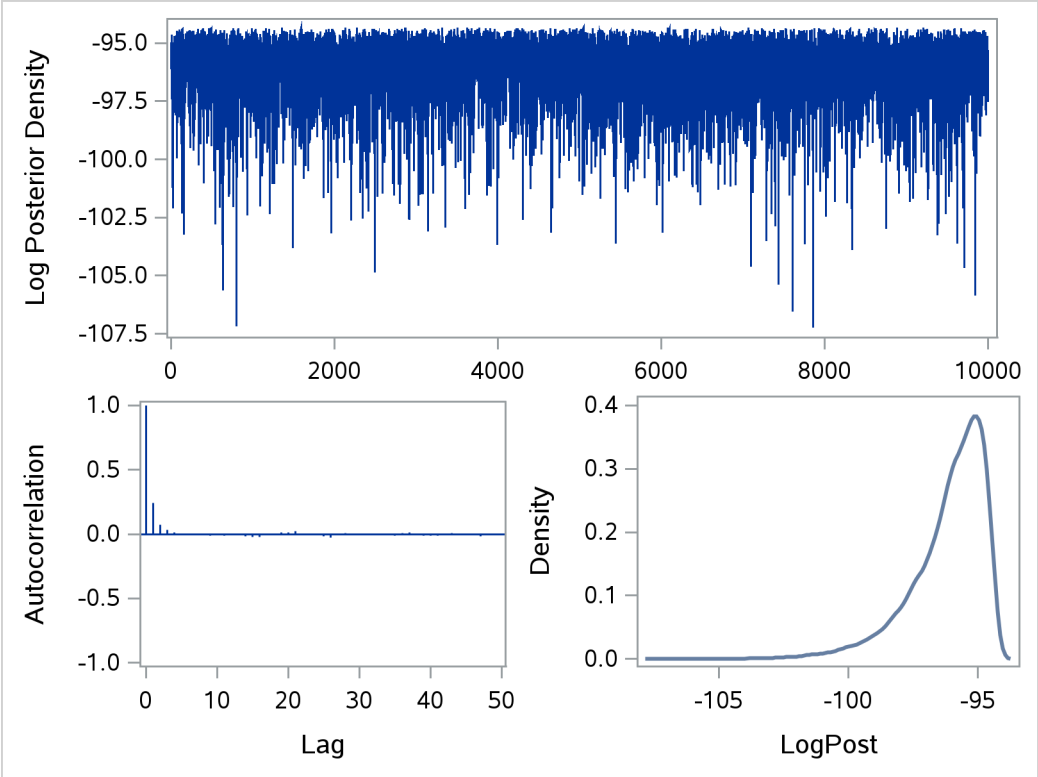

Caterpillar Plot

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmccat.sas.)

The caterpillar plot is a side-by-side bar plot of 95% intervals for multiple parameters. Typically, it is used to visualize and compare random-effects parameters, which can come in large numbers in certain models. You can use the %CATER autocall macro to create a caterpillar plot. The %CATER macro requires you specify an input data set and a list of variables that you want to plot.

A random-effects model that has 21 random-effects parameters is used here for illustrational purpose. For an explanation of the random-effects model and the data involved, see "Example 80.7: Logistic Regression Random-Effects Model" on page 6415. The following statements generate a SAS data set and fit the model:

```
title 'Create a Caterpillar Plot';

data seeds;
   input r n seed extract @@;
   ind = _N_;
   datalines;
10  39  0  0    23  62  0  0    23  81  0  0    26  51  0  0
17  39  0  0     5   6  0  1    53  74  0  1    55  72  0  1
32  51  0  1    46  79  0  1    10  13  0  1     8  16  1  0
10  30  1  0     8  28  1  0    23  45  1  0     0   4  1  0
 3  12  1  1    22  41  1  1    15  30  1  1    32  51  1  1
 3   7  1  1
;

ods select none;
proc mcmc data=seeds outpost=postout seed=332786 nmc=20000;
   parms beta0 0 beta1 0 beta2 0 beta3 0 s2 1;
   prior s2 ~ igamma(0.01, s=0.01);
   prior beta: ~ general(0);
   w = beta0 + beta1*seed + beta2*extract + beta3*seed*extract;
   random delta ~ normal(w, var=s2) subject=ind;
   pi = logistic(delta);
   model r ~ binomial(n = n, p = pi);
run;
ods select all;
```

The output data set Postout contains posterior draws for all 21 random-effects parameters, delta_1 ⋯ delta_21. You can use the following statements to generate a caterpillar plot for the 21 parameters:

```
%CATER(data=postout, var=delta:);
```

Figure 80.24 is a caterpillar plot of the random-effects parameters delta_1–delta_21.

Figure 80.24 Caterpillar Plot of the Random-Effects Parameters

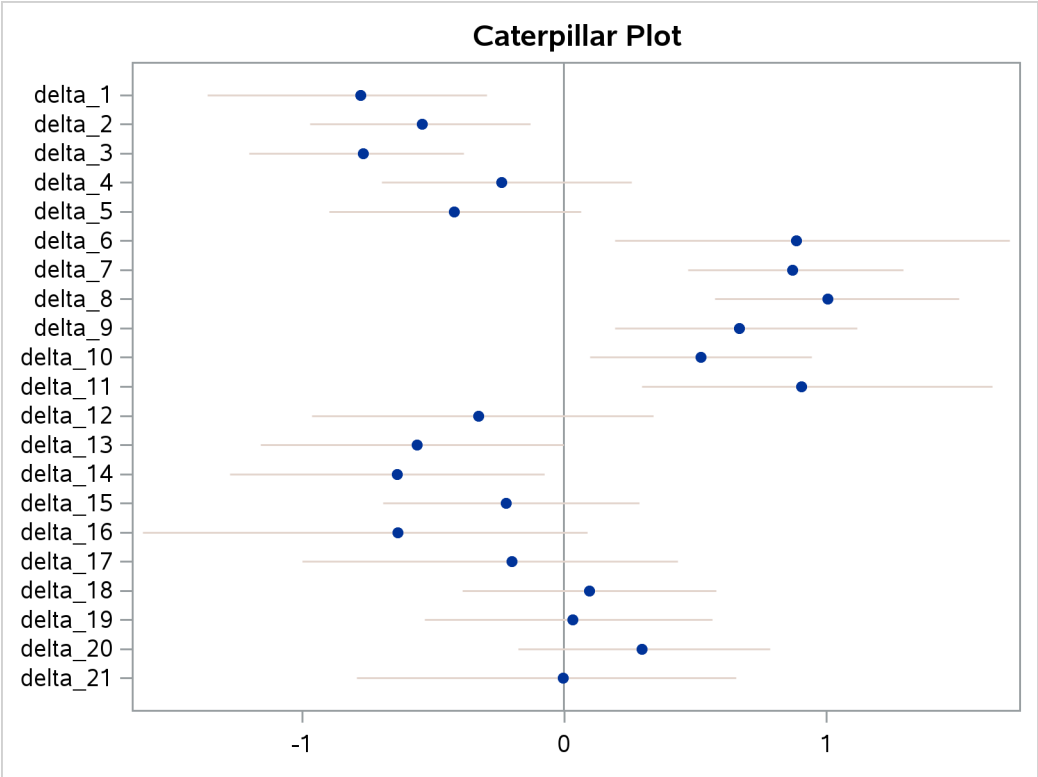

If you want to change the display of the caterpillar plot, such as using a different line pattern, color, or size of the markers, you need to first modify the **Stat.MCMC.Graphics.Caterpillar** template and then call the %CATER macro again.

You can use the following statements to view the source of the **Stat.MCMC.Graphics.Caterpillar** template:

```
proc template;
   path sashelp.tmplmst;
   source Stat.MCMC.Graphics.Caterpillar;
run;
```

Figure 80.25 lists the source statements of the template that is used to generate the template for the caterpillar plot.

Figure 80.25 Source Statements for Stat.MCMC.Graphics.Caterpillar Template

```
define statgraph Stat.MCMC.Graphics.Caterpillar;
   dynamic _OverallMean _VarName _VarMean _XLower _XUpper _byline_ _bytitle_
      _byfootnote_;
   begingraph;
      entrytitle "Caterpillar Plot";
      layout overlay / yaxisopts=(offsetmin=0.05 offsetmax=0.05 display=(line
         ticks tickvalues)) xaxisopts=(display=(line ticks tickvalues));
         referenceline x=_OVERALLMEAN / lineattrs=(color=
            GraphReference:ContrastColor);
         HighLowPlot y=_VARNAME high=_XUPPER low=_XLOWER / lineattrs=
            GRAPHCONFIDENCE;
         scatterplot y=_VARNAME x=_VARMEAN / markerattrs=(size=5 symbol=
            circlefilled);
      endlayout;
      if (_BYTITLE_)
         entrytitle _BYLINE_ / textattrs=GRAPHVALUETEXT;
      else
         if (_BYFOOTNOTE_)
            entryfootnote halign=left _BYLINE_;
         endif;
      endif;
   endgraph;
end;
```

You can use the TEMPLATE procedure (see Chapter 24, "Statistical Graphics Using ODS") to modify the graph template. Subsequent calls to the %CATER macro will use the modified template to make the graph.

Autocall Macros for Postprocessing

Although PROC MCMC provides a number of convergence diagnostic tests and posterior summary statistics, PROC MCMC performs the calculations only if you specify the options in advance. If you wish to analyze the posterior draws of unmonitored parameters or functions of the parameters that are calculated in later DATA step calls, you can use the autocall macros in Table 80.50.

Table 80.50 Postprocessing Autocall Macros

Macro	Description
%ESS	Effective sample sizes
%GEWEKE*	Geweke diagnostic
%HEIDEL*	Heidelberger-Welch diagnostic
%MCSE	Monte Carlo standard errors
%RAFTERY	Raftery diagnostic
%POSTACF	Autocorrelation
%POSTCOR	Correlation matrix
%POSTCOV	Covariance matrix
%POSTINT	Equal-tail and HPD intervals
%POSTSUM	Summary statistics
%SUMINT	Mean, standard deviation, and HPD interval

*The %GEWEKE and %HEIDEL macros use a different optimization routine than that used in PROC MCMC. As a result, there might be numerical differences in some cases, especially when the sample size is small.

Table 80.51 lists options that are shared by all postprocessing autocall macros. See Table 80.52 for macro-specific options.

Table 80.51 Shared Options

Option	Description
DATA=*SAS-data-set*	Input data set that contains posterior samples
VAR=*variable-list*	Specifies the variables on which you want to carry out the calculation.
PRINT=YES | NO	Displays the results. The default is YES.
OUT=*SAS-data-set*	Specifies a name for the output SAS data set to contain the results.

Suppose that the data set that contains posterior samples is called post and that the variables of interest are defined in the macro variable &PARMS. The following statements call the %ESS macro and calculates the effective sample sizes for each variable:

```
%let parms = alpha beta u_1-u_17;
%ESS(data=post, var=&parms);
```

By default, the ESS estimates are displayed. You can choose not to display the result and save the output to a data set with the following statement:

```
%ESS(data=post, var=&parms, print=NO, out=eout);
```

Some of the macros can take additional options, which are listed in Table 80.52.

Table 80.52 Macro-Specific Options

Macro	Option	Description
%ESS	AUTOCORLAG=*numeric*	Specifies the maximum number of autocorrelation lags used in computing the ESS estimates. By default, AUTOCORLAG=MIN(500, NOBS/4), where NOBS is the sample size of the input data set.
	HIST=YES|NO	Displays a histogram of all ESS estimates. The default is NO.
%HEIDEL	SALPHA=*numeric*	Specifies the α level for the stationarity test. By default, SALPHA=0.05.
	HALPHA=*numeric*	Specifies the α level for the halfwidth test. By default, HALPHA=0.05.
	EPS=*numeric*	Specifies a small positive number ϵ such that if the halfwidth is less than ϵ times the sample mean of the remaining iterations, the halfwidth test is passed. By default, EPS=0.1.
%GEWEKE	FRAC1=*numeric*	Specifies the earlier portion of the Markov chain used in the test. By default, FRAC1=0.1.
	FRAC2=*numeric*	Specifies the latter portion of the Markov chain used in the test. By default, FRAC2=0.5.
%MCSE	AUTOCORLAG=*numeric*	Specifies the maximum number of autocorrelation lags used in computing the Monte Carlo standard error estimates. By default, AUTOCORLAG=MIN(500, NOBS/4), where NOBS is the sample size of the input data set.
%RAFTERY	Q=*numeric*	Specifies the order of the quantile of interest. By default, Q=0.025.
	R=*numeric*	Specifies the margin of error for measuring the accuracy of estimation of the quantile. By default, R=0.005.
	S=*numeric*	Specifies the probability of attaining the accuracy of the estimation of the quantile. By default, S=0.95.
	EPS=*numeric*	Specifies the tolerance level for the stationary test. By default, EPS=0.001.
%POSTACF	LAGS=%STR(*numeric-list*)	Specifies autocorrelation lags calculated. The default values are 1, 5, 10, and 50.
%POSTINT	ALPHA=*value*	Specifies the α level ($0 < \alpha < 1$) for the interval estimates. By default, ALPHA=0.05.

For example, the following statement calculates and displays autocorrelation at lags 1, 6, 11, 50, and 100. Note that the lags in the *numeric-list* need to be separated by commas “,”.

```
%PostACF(data=post, var=&parms, lags=%str(1 to 15 by 5, 50, 100));
```

Gamma and Inverse Gamma Distributions

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcgsm.sas.)

The gamma and inverse gamma distributions are widely used in Bayesian analysis. With their respective scale and inverse scale parameterizations, they are a frequent source of confusion in the field. This section aims to clarify their parameterizations and common usages.

The gamma distribution is often used as the conjugate prior for the precision parameter ($\tau = 1/\sigma^2$) in a normal distribution. See Table 80.19 in the section "Standard Distributions" on page 6289 for the density definitions. You can specify the distribution in two ways:

- **gamma(shape=, scale=)** which has mean **shape·scale** and variance **shape · scale**^{2}
- **gamma(shape=, iscale=)** which has mean $\frac{\textbf{shape}}{\textbf{iscale}}$ and variance $\frac{\textbf{shape}}{\textbf{iscale}^2}$

The parameterization of the gamma distribution that is preferred by most Bayesian analysts is to have the same number in both hyperparameter positions, which results in a prior distribution that has mean 1. To do this, you should use the **iscale=** parameterization. In addition, if you choose a small value (for example, 0.01), the prior distribution takes on a large variance (100 in this example). To specify this prior in PROC MCMC, use **gamma(shape=0.01, iscale=0.01)**[2], *not* **gamma(shape=0.01, scale=0.01)**.

If you specify the **scale=** parameterization, as in **gamma(shape=0.01, scale=0.01)**, you would get a prior distribution that has mean 0.0001 and variance 0.000001. This would lead to a completely different posterior inference: the prior would push the precision parameter estimate close to 0, or the variance estimate to a large value.

The inverse gamma distribution is often used as the conjugate prior of the variance parameter (σ^2) in a normal distribution. See Table 80.22 in the section "Standard Distributions" on page 6289 for the density definitions. Similar to the gamma distribution, you can specify the inverse gamma distribution in two ways:

- **igamma(shape=, scale=)**
- **igamma(shape=, iscale=)**

The inverse gamma distribution does not have a mean when the shape parameter is less than or equal to 1 and does not have a variance when the shape parameter is less than or equal to 2.

A gamma prior distribution on the precision is the equivalent to an inverse gamma prior distribution on the variance. The equivalency is the following:

$$\tau \sim \textbf{gamma(shape=0.01, iscale=0.01)} \Leftrightarrow \sigma^2 \sim \textbf{igamma(shape=0.01, scale=0.01)}$$

NOTE: This mnemonic might help you remember the parameterization scheme of the distributions. If you prefer to have identical hyperparameter values in the distribution, you should specify one and only one "i.". When the "i" appears in the **igamma** distribution name for the variance parameter, choose the **scale=**

[2]Specifying the same number at both positions and choosing a small value has been popularized by the WinBUGS software program. The WinBUGS's distribution specification of **dgamma(0.01, 0.01)** is equivalent to specifying **gamma(shape=0.01, iscale=0.01)** in PROC MCMC.

parameterization; when the "i" appears in the **iscale=** parameterization, choose the **gamma** distribution for the precision parameter.

If you are not sure about the choices of other hyperparameter values and what type of prior distributions they induce, you can write a simple PROC MCMC program and see the distributions as in the following example:

```
data a;
run;

ods graphics on;
ods select DensityPanel;
proc mcmc data=a stats=none diag=none nmc=10000 outpost=gout
   plots=density seed=1;
   parms gamma_3_is2 gamma_001_sc4 igamma_12_sc001 igamma_2_is01;
   prior gamma_3_is2     ~  gamma(shape=3, iscale=2);
   prior gamma_001_sc4   ~  gamma(shape=0.01, scale=4);
   prior igamma_12_sc001 ~ igamma(shape=12, scale=0.01);
   prior igamma_2_is01   ~ igamma(shape=2, iscale=0.1);
   model general(0);
run;
ods graphics off;
```

The preceding statements specify four different gamma and inverse gamma distributions with various scale and inverse scale parameter values. The output of kernel density plots of these four prior distributions is shown in Figure 80.26. Note how the X axis scales vary across different distributions.

Figure 80.26 Density Plots of Different Gamma and Inverse Gamma Distributions

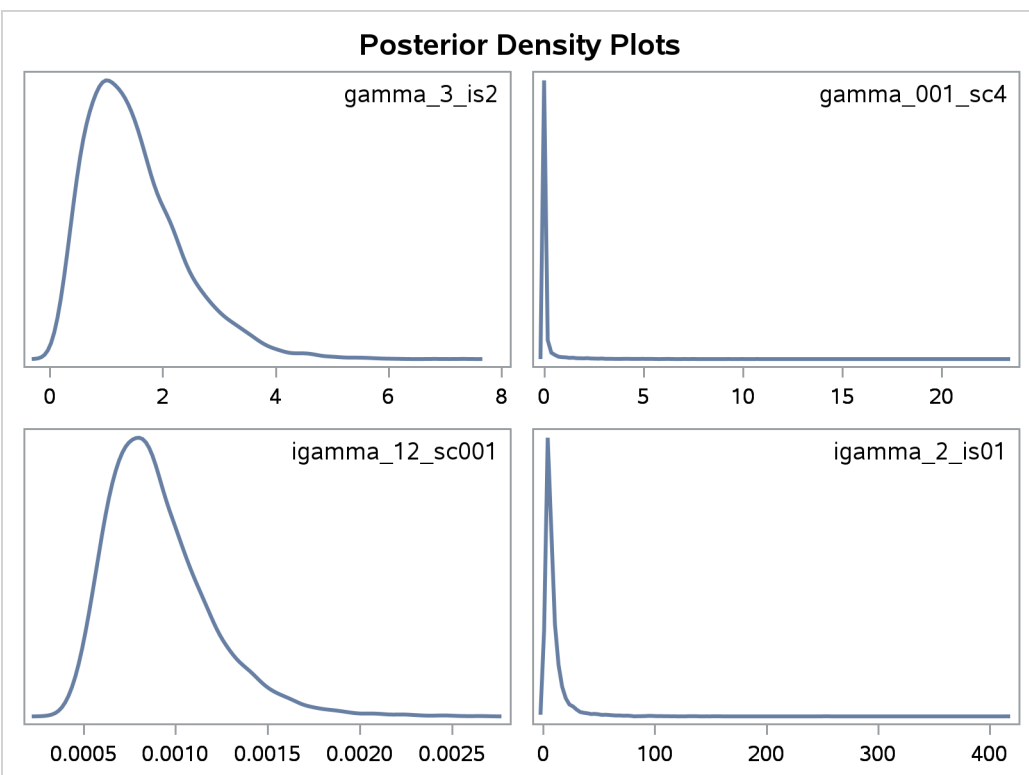

Posterior Predictive Distribution

The posterior predictive distribution

$$p(\mathbf{y}_{\text{pred}}|\mathbf{y}) = \int p(\mathbf{y}_{\text{pred}}|\theta)p(\theta|\mathbf{y})d\theta$$

can often be used to check whether the model is consistent with data. For more information about using predictive distribution as a model checking tool, see Gelman et al. 2004, Chapter 6 and the bibliography in that chapter. The idea is to generate replicate data from $p(\mathbf{y}_{\text{pred}}|\mathbf{y})$—call them $\mathbf{y}^i_{\text{pred}}$, for $i = 1, \ldots, M$, where M is the total number of replicates—and compare them to the observed data to see whether there are any large and systematic differences. Large discrepancies suggest a possible model misfit. One way to compare the replicate data to the observed data is to first summarize the data to some test quantities, such as the mean, standard deviation, order statistics, and so on. Then compute the tail-area probabilities of the test statistics (based on the observed data) with respect to the estimated posterior predictive distribution that uses the M replicate $\mathbf{y}_{\text{pred}}$ samples.

Let $T(\cdot)$ denote the function of the test quantity, $T(\mathbf{y})$ the test quantity that uses the observed data, and $T(\mathbf{y}^i_{\text{pred}})$ the test quantity that uses the ith replicate data from the posterior predictive distribution. You calculate the tail-area probability by using the following formula:

$$\Pr(T(\mathbf{y}_{\text{pred}}) > T(\mathbf{y})|\theta)$$

The following example shows how you can use PROC MCMC to estimate this probability.

An Example for the Posterior Predictive Distribution

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcppd.sas.)

This example uses a normal mixed model to analyze the effects of coaching programs for the scholastic aptitude test (SAT) in eight high schools. For the original analysis of the data, see Rubin (1981). The presentation here follows the analysis and posterior predictive check presented in Gelman et al. (2004). The data are as follows:

```
title 'An Example for the Posterior Predictive Distribution';

data SAT;
   input effect se @@;
   ind=_n_;
   datalines;
28.39 14.9  7.94 10.2 -2.75 16.3
 6.82 11.0 -0.64  9.4  0.63 11.4
18.01 10.4 12.16 17.6
;
```

The variable effect is the reported test score difference between coached and uncoached students in eight schools. The variable se is the corresponding estimated standard error for each school. In a normal mixed effect model, the variable effect is assumed to be normally distributed:

$$\text{effect}_i \sim \text{normal}(\mu_i, \text{se}^2) \quad \text{for } i = 1, \ldots, 8$$

The parameter μ_i has a normal prior with hyperparameters (m, v):

$$\mu_i \sim \text{normal}(m, \text{var} = v)$$

The hyperprior distribution on m is a uniform prior on the real axis, and the hyperprior distribution on v is a uniform prior from 0 to infinity.

The following statements fit a normal mixed model and use the PREDDIST statement to generate draws from the posterior predictive distribution.

```
ods exclude all;
proc mcmc data=SAT outpost=out nmc=40000 seed=12;
   parms m 0;
   parms v 1 /slice;
   prior m ~ general(0);
   prior v ~ general(1,lower=0);
   random mu ~ normal(m,var=v) subject=ind monitor=(mu);
   model effect ~ normal(mu,sd=se);
   preddist outpred=pout nsim=5000;
run;
ods exclude none;
```

The ODS EXCLUDE ALL statement disables all ODS output because you are primarily interested in the samples of the predictive distribution. The HYPER, PRIOR, and MODEL statements specify the Bayesian model of interest. The PREDDIST statement generates samples from the posterior predictive distribution and stores the samples in the Pout data set. The predictive variables are named effect_1, . . ., effect_8. When no COVARIATES option is specified, the covariates in the original input data set SAT are used in the prediction. The NSIM= option specifies the number of predictive simulation iterations.

The following statements use the Pout data set to calculate the four test quantities of interest: the average (mean), the sample standard deviation (sd), the maximum effect (max), and the minimum effect (min). The output is stored in the Pred data set.

```
data pred;
   set pout;
   mean = mean(of effect:);
   sd = std(of effect:);
   max = max(of effect:);
   min = min(of effect:);
run;
```

The following statements compute the corresponding test statistics, the mean, standard deviation, and the minimum and maximum statistics on the real data and store them in macro variables. You then calculate the tail-area probabilities by counting the number of samples in the data set Pred that are greater than the observed test statistics based on the real data.

```
proc means data=SAT noprint;
   var effect;
   output out=stat mean=mean max=max min=min stddev=sd;
run;

data _null_;
   set stat;
   call symputx('mean',mean);
```

```
   call symputx('sd',sd);
   call symputx('min',min);
   call symputx('max',max);
run;

data _null_;
   set pred end=eof nobs=nobs;
   ctmean + (mean>&mean);
   ctmin + (min>&min);
   ctmax + (max>&max);
   ctsd + (sd>&sd);
   if eof then do;
      pmean = ctmean/nobs; call symputx('pmean',pmean);
      pmin = ctmin/nobs; call symputx('pmin',pmin);
      pmax = ctmax/nobs; call symputx('pmax',pmax);
      psd = ctsd/nobs; call symputx('psd',psd);
   end;
run;
```

You can plot histograms of each test quantity to visualize the posterior predictive distributions. In addition, you can see where the estimated *p*-values fall on these densities. Figure 80.27 shows the histograms. To put all four histograms on the same panel, you need to use PROC TEMPLATE to define a new graph template. (See Chapter 24, "Statistical Graphics Using ODS.") The following statements define the template **twobytwo**:

```
proc template;
   define statgraph twobytwo;
      begingraph;
         layout lattice / rows=2 columns=2;
            layout overlay / yaxisopts=(display=none)
                             xaxisopts=(label="mean");
               layout gridded / columns=2 border=false
                                autoalign=(topleft topright);
                  entry halign=right "p-value =";
                  entry halign=left eval(strip(put(&pmean, 12.2)));
               endlayout;
               histogram mean / binaxis=false;
               lineparm x=&mean y=0 slope=. /
                        lineattrs=(color=red thickness=5);
            endlayout;
            layout overlay / yaxisopts=(display=none)
                             xaxisopts=(label="sd");
               layout gridded / columns=2 border=false
                                autoalign=(topleft topright);
                  entry halign=right "p-value =";
                  entry halign=left eval(strip(put(&psd, 12.2)));
               endlayout;
               histogram sd / binaxis=false;
               lineparm x=&sd y=0 slope=. /
                        lineattrs=(color=red thickness=5);
            endlayout;
            layout overlay / yaxisopts=(display=none)
                             xaxisopts=(label="max");
               layout gridded / columns=2 border=false
                                autoalign=(topleft topright);
```

```
                 entry halign=right "p-value =";
                 entry halign=left eval(strip(put(&pmax, 12.2)));
              endlayout;
              histogram max / binaxis=false;
              lineparm x=&max y=0 slope=. /
                       lineattrs=(color=red thickness=5);
           endlayout;
           layout overlay / yaxisopts=(display=none)
                            xaxisopts=(label="min");
              layout gridded / columns=2 border=false
                               autoalign=(topleft topright);
                 entry halign=right "p-value =";
                 entry halign=left eval(strip(put(&pmin, 12.2)));
              endlayout;
              histogram min / binaxis=false;
              lineparm x=&min y=0 slope=. /
                       lineattrs=(color=red thickness=5);
           endlayout;
        endlayout;
     endgraph;
  end;
run;
```

You call PROC SGRENDER to create the graph, which is shown in Figure 80.27. (See the SGRENDER procedure in the *SAS ODS Graphics: Procedures Guide*.) There are no extreme p-values observed; this supports the notion that the predicted results are similar to the actual observations and that the model fits the data.

```
proc sgrender data=pred template=twobytwo;
run;
```

Figure 80.27 Posterior Predictive Distribution Check for the SAT example

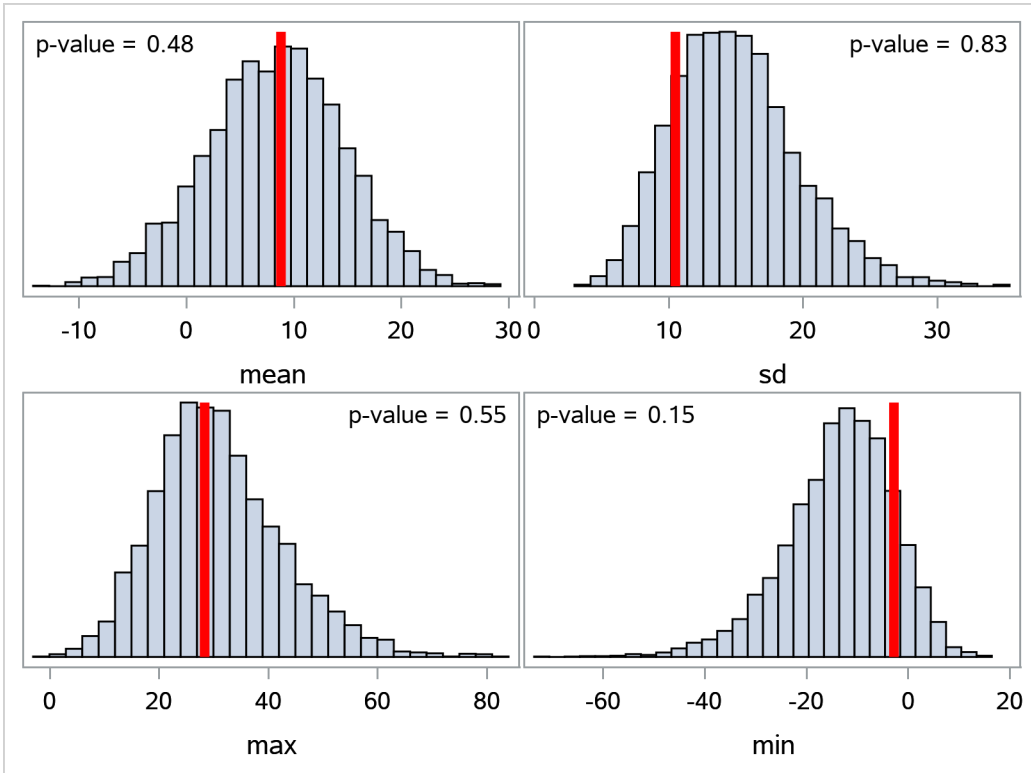

Note that the posterior predictive distribution is not the same as the prior predictive distribution. The prior predictive distribution is $p(\mathbf{y})$, which is also known as the marginal distribution of the data. The prior predictive distribution is an integral of the likelihood function with respect to the prior distribution

$$p(\mathbf{y}_{\text{pred}}) = \int p(\mathbf{y}_{\text{pred}}|\theta)p(\theta)d\theta$$

and the distribution is not conditional on observed data.

Handling of Missing Data

PROC MCMC automatically augments missing values[3] via the use of the MODEL statement. PROC MCMC treats missing values as unknown parameters, assigns distributions to the variables, and incorporates the sampling of the missing data as part of the Markov chain.

You can use the MISSING= option in the PROC MCMC statement to specify how you want PROC MCMC to handle the missing values.

- If you specify MISSING=CC (CC stands for complete cases), PROC MCMC discards all observations that have missing or partial missing values before carrying out the simulation.

[3] A missing value is usually, although not necessarily, represented by a single period (.) in the input data set.

- If you specify MISSING=AC (AC stands for all cases), PROC MCMC neither discards any missing values nor augments them.
- If you specify MISSING=CCMODELY, PROC MCMC treats missing response values as parameters and incorporates them as part of the simulation. But the procedure discards all observations that have missing covariates. A covariate is a data set variable that appears in the program but does not appear to the left of the tilde in the MODEL statement.
- If you specify MISSING=ACMODELY, PROC MCMC samples missing responses without discarding observations that contain missing values in covariates.

Generally speaking, there are three types of missing data models, as discussed by Rubin (1976). Also see Little and Rubin (2002) for a comprehensive treatment of missing data analysis. The rest of this section provides an overview of these three types of missing data models and explains how to use PROC MCMC to fit them.

Missing Completely at Random (MCAR)

Data are said to be MCAR if the probability of a missing value (or the failure of observing a value) does not depend on any other observations in the data set, regardless of whether they are observed or missing. That is, the observed and unobserved values are independent of each other: if y_i is missing, it is MCAR if the probability of observing y_i is independent of other y_j (and other covariates x_i) in the data set. Under this assumption, both the observed and unobserved data are random samples of all the data; hence, fitting a model based only on the observed data does not introduce any biases. This type of analysis is called a complete-case analysis. To carry out a complete-case analysis, you must specify MISSING=CC in the PROC MCMC statement.

Missing at Random (MAR)

Data are said to be MAR if the probability of a missing value can depend on some observed quantities but does not depend on any unobserved data. For example, suppose that x_i are completely observed for all observations and some y_i are missing. MAR states that the probability of observing y_i is independent of other missing y_i (values that could have been observed) and that it depends only on x_i (and, potentially, observed y_i).

The MAR assumption states that the missing y_i are no longer random samples and that they need to be modeled (via the likelihood specification of the missing values). At the same time, the independence assumption of the missing values on the unobserved quantities states that the missing mechanism (usually an binary indicator variable such that $r_i = 1$ if y_i is missing and $r_i = 0$ otherwise) can be ignored and does not need to be taken into account. Hence, MAR is sometimes referred to as *ignorably missing*. It is not the missing values that can be ignored, it is the missing mechanism that can be ignored.

By default, PROC MCMC treats the missing data as MAR (this assumes that you do not input a binary indicator variable r_i and model it specifically): each missing value becomes an extra parameter and PROC MCMC updates it in every iteration. PROC MCMC assumes that both the missing values and observed values arise from the same distribution (which is specified in the MODEL statement),

$$\mathbf{y} = \{\mathbf{y_{obs}}, \mathbf{y_{mis}}\} \sim f(\mathbf{y}|\theta)$$

where $\mathbf{y}$ consists of observed ($\mathbf{y_{obs}}$) and missing ($\mathbf{y_{mis}}$) values, and $f(\mathbf{y}|\theta)$ is the likelihood function with parameters θ.

You can use the MODEL statement to model missing covariates. Using multiple MODEL statements enables you to specify, for example, a marginal distribution for missing values in covariate x and a conditional distribution for the response variable y given x as follows:

```
model x ~ normal(alpha, var=s2_x);
model y ~ normal(beta * x, var=s2_y);
```

In each iteration, PROC MCMC draws samples for every missing value in variable x, then every missing value in variable y, conditional on the drawn values of the x variable.

Missing Not at Random (MNAR)

Data are said to be MNAR if the probability of a missing value depends on unobserved data (or data that could have been observed): the probability that y_i is missing depends on the missing values of other y_i. This is a very general scenario that assumes that the missing mechanism is no longer ignorable (it is sometimes referred to as *nonignorably missing*) and that a model for the missing mechanism is required in order to make correct inferences about the model parameters.

Let $\mathbf{R} = (r_1, \ldots, r_n)$ be the missing value indicator for $\mathbf{Y} = (y_1, \ldots, y_n)$, where $r_i = 1$ if y_i is missing and $r_i = 0$ otherwise. This $\mathbf{R}$ is usually part of an input data set where you preprocess the response variable and create this missing value indicator variable. Modeling MNAR data implies that you must specify a joint likelihood function over $\mathbf{R}$ and $\mathbf{Y}$: $f(\mathbf{R}, \mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})$, where $\mathbf{X}$ represents the covariates and $\boldsymbol{\theta}$ represents the model parameters. This joint distribution can be factored in two ways: a pattern-mixture model and a selection model.

The *selection model* factors the joint distribution $\mathbf{R}$ and $\mathbf{Y}$ into a marginal distribution for $\mathbf{Y}$ and a conditional distribution for $\mathbf{R}$,

$$f(\mathbf{R}, \mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) \propto f(\mathbf{Y}|\mathbf{X}, \boldsymbol{\alpha}) \cdot f(\mathbf{R}|\mathbf{Y}, \mathbf{X}, \boldsymbol{\beta})$$

where $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \boldsymbol{\beta})$, $f(\mathbf{R}|\mathbf{Y}, \mathbf{X}, \boldsymbol{\alpha})$ is usually a binary model with a logit or probit link that involves regression parameters $\boldsymbol{\alpha}$, and $f(\mathbf{Y}|\mathbf{X}, \boldsymbol{\beta})$ is the sampling distribution that generates y_i with model parameters $\boldsymbol{\beta}$.

The *pattern-mixture model* factors the opposite way, a marginal distribution for $\mathbf{R}$ and a conditional distribution for $\mathbf{Y}$,

$$f(\mathbf{R}, \mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) \propto f(\mathbf{R}|\mathbf{X}, \boldsymbol{\gamma}) \cdot f(\mathbf{Y}|\mathbf{R}, \mathbf{X}, \boldsymbol{\delta})$$

where $\boldsymbol{\theta} = (\boldsymbol{\gamma}, \boldsymbol{\delta})$.

You can use PROC MCMC to fit either model by specifying multiple MODEL statements: one for the marginal distribution and one for the conditional distribution. Suppose that the variable r is the missing data indicator, which is modeled using a logit model, and that the response variable y is a Poisson regression that includes the missing variable indicator as one of its covariates. The following statements are a PROC MCMC program that fits a pattern-mixture model:

```
pi = logistic(alpha * x1);
model r ~ binary(pi);
mu = beta0 + beta1 * x2 + beta3 * r;
model y ~ poisson(exp(mu));
```

The first MODEL statement uses a binary model with logit link to model the missing mechanism, and the second MODEL statement models the response variable with a Poisson regression that includes the missing

value indicator as one of its covariates. Each of the two sets of regression has its covariates and regression coefficients. If this hypothetical data set contained missing values in covariates x1 and x2, you could add two more MODEL statements to handle each variable as follows:

```
model x1 ~ normal(mu1, var=s2_x1);
pi = logistic(alpha * x1);
model r ~ binary(pi);
model x2 ~ normal(mu2, var=s2_x2);
mu = beta0 + beta1 * x2 + beta3 * r;
model y ~ poisson(exp(mu));
```

Functions of Random-Effects Parameters

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcmre.sas.)

When you specify a RANDOM statement in a program, PROC MCMC internally creates a random-effects parameter for every unique value in the SUBJECT= variable. You can calculate any transformations of these random-effects parameters by applying SAS functions to the effect, and you can use the transformed variable in the subsequent statements. For example, the following statements perform a logit transformation of an effect:

```
random u ~ normal(mu, var=s2) subject=students;
p = logistic(u);
...
```

The value of the variable p changes with u as the procedure steps through the input data set: for different unique values of the students variable, u takes on a different parameter value, and p changes accordingly.

To save all the transformed values in p to the OUTPOST= data set, you cannot just specify the MONITOR=(p) option in the PROC MCMC statement. With such a specification, PROC MCMC can save only one value of p (usually the value associated with the last observation in the data set); it cannot save all values. To output all transformed values, you must create an array to store every transformation and use the MONITOR= option to save the entire array to the OUTPOST= data set. The difficult part of the programming involves the creation of the correct array index to use in different types of the SUBJECT= variables. The rest of this section describes how to monitor functions of random-effects parameters in different situations.

Indexing Subject Variable

This subsection describes how to monitor transformation of an effect u when the students variable is an indexing subject variable. An indexing subject variable is an integer variable that takes value from one to the total number of unique subjects in a variable. In other words, the variable can be used as an index in a SAS array. The indexing subject variable does not need to be sorted for the example code in this section to work. An example of an indexing variable takes the values of (`1 2 3 4 5 1 2 3 4 5`), where the total number of observation is $n=10$ and the number of unique values is $m=5$.

The following statements create an indexing variable students in the data set a:

```
data a;
   input students @@;
   datalines;
1 1 2 2 3 3 4 4 5 5 6 6 7 7 8 8 9 9 10 10
;
```

The following statements run a random-effects model without any response variables. There are only random-effects parameters in the model; the program calculates the logit transformation of each effect, saves the results to the OUTPOST= data set, and produces Figure 80.28:

```
proc mcmc data=a monitor=(p) diag=none stats=none outpost=a1
   plots=none seed=1;
   array p[10];
   random u ~ n(0, sd=1) subject=students;
   p[students] = logistic(u);
   model general(0);
run;

proc print data=a1(obs=3);
run;
```

The ARRAY statement creates an array p of size 10, which is the number of unique values in students. The p array stores all the transformed values. The RANDOM statement declares u to be an effect with the subject variable students. The P[STUDENTS] assignment statement calculates the logit transformations of u and saves them in appropriate array elements—this is why the students variable must be an indexing variable. Because the students variable used in the p[] array is also the subject variable in the RANDOM statement, PROC MCMC can match each random-effects parameter with the corresponding element in array p. The MONITOR= option monitors all elements in p and saves the output to the a1 data set. The a1 data set contains variables p1–p10. Figure 80.28 shows the first three observations of the OUTPOST= data set.

Figure 80.28 Monitor Functions of Random Effect u

Obs	Iteration	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	u_1	u_2
1	1	0.5050	0.7334	0.5375	0.5871	0.6862	0.4944	0.5740	0.5991	0.6812	0.8678	0.0198	1.0120
2	2	0.5563	0.4254	0.7260	0.5280	0.4593	0.5797	0.5813	0.7360	0.9079	0.3351	0.2261	-0.3006
3	3	0.6132	0.8031	0.4735	0.3135	0.7047	0.5273	0.6761	0.2379	0.2938	0.3986	0.4607	1.4058

Obs	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_10	LogReff	LogLike	LogPost
1	0.1504	0.3521	0.7826	-0.0225	0.2983	0.4019	0.7595	1.8819	-12.2658	0	-12.2658
2	0.9742	0.1120	-0.1630	0.3213	0.3281	1.0253	2.2887	-0.6854	-13.2393	0	-13.2393
3	-0.1060	-0.7837	0.8698	0.1092	0.7361	-1.1641	-0.8772	-0.4112	-12.3984	0	-12.3984

The variable p1 is the logit transformation of the variable u_1, p2 is the logit transformation of the variable u_2, and so on.

The same idea works for a students variable that is unsorted. The following statements create an unsorted indexing variable students with repeated measures in each subject, fit the same model, and produce Figure 80.29:

```
data a;
   input students @@;
   datalines;
   1 1 1 3 5 3 4 5 3 1 5 5 4 4 2 2 2 2 4 3
;
proc mcmc data=a monitor=(p) diag=none stats=none outpost=a1
   plots=none seed=1;
   array p[5];
   random u ~ n(0, sd=1) subject=students;
   p[students] = logistic(u);
   model general(0);
run;

proc print data=a1(obs=3);
run;
```

Figure 80.29 Monitor Functions of Random Effect u When the students Variable is Unsorted

Obs	Iteration	p1	p2	p3	p4	p5	u_1	u_3	u_5	u_4	u_2	LogReff	LogLike	LogPost
1	1	0.5050	0.6862	0.7334	0.5871	0.5375	0.0198	1.0120	0.1504	0.3521	0.7826	-5.4865	0	-5.4865
2	2	0.4944	0.8678	0.5740	0.6812	0.5991	-0.0225	0.2983	0.4019	0.7595	1.8819	-6.7793	0	-6.7793
3	3	0.5563	0.4593	0.4254	0.5280	0.7260	0.2261	-0.3006	0.9742	0.1120	-0.1630	-5.1595	0	-5.1595

There are five random-effects parameters in this example, and the array p also has five elements. The values p1–p5 are the transformations of u_1–u_5, respectively. The u variables are not sorted from u_1 to u_5 because PROC MCMC creates the names according to the order by which the subject variable appears in the input data set. Nevertheless, because students is an indexing variable, the first element p[1] stores the transformation that corresponds to **`students=1`** (which is u_1), the second element p[2] stores the transformation that corresponds to **`students=2`**, and so on.

Non-Indexing Subject Variable

A non-indexing subject variable can take values of character literals (for example, names) or numerals (for example, ZIP code or a person's weight). This section illustrates how to monitor functions of random-effects parameters in these situations.

Suppose you have unsorted character literals as the subject variable:

```
data a;
   input students$ @@;
   datalines;
   smith john john mary kay smith lizzy ben ben dylan
   ben toby abby mary kay kay lizzy ben dylan mary
;
```

A statement such as following does not work anymore because a character variable cannot be used as an array index:

```
p[students] = logistic(u);
```

In this situation, you usually need to do two things: (1) find out the number of unique values in the subject variable, and (2) create a numeric index variable that replaces the students array index. You can use the following statements to do the first task:

```
proc sql noprint;
   select count(distinct(students)) into :nuniq from a;
quit;
%put &nuniq;
```

The PROC SQL call counts the distinct values in the students variable and saves the count to the macro variable &nuniq. The macro variable is used later to specify the element size of the p array. In this example, the a data set contains 20 observations and 9 unique elements (the value of &nuinq).

The following statements create an Index variable in the a data set that is in the order by which the students names appear in the data set:

```
proc freq data=a order=data noprint;
   tables students / out=_f(keep=students);
run;

proc print data=_f;
run;

data a(drop=n);
   set a;
   do i = 1 to nobs until(students=n);
      set _f(keep=students rename=(students=n)) point=i nobs=nobs;
      Index = i;
   end;
run;

proc print data=a;
run;
```

The PROC FREQ call identifies the unique students names and saves them to the _f data set, which is displayed in Figure 80.30.

Figure 80.30 Unique Names in the Variable Students

Obs	students
1	smith
2	john
3	mary
4	kay
5	lizzy
6	ben
7	dylan
8	toby
9	abby

The DATA step steps through the a data set and creates an Index variable to match the order in which the

students names appear in the data set. The new a data set[4] is displayed in Figure 80.31.

Figure 80.31 New Index Variable in the a Data Set

Obs	students	Index
1	smith	1
2	john	2
3	john	2
4	mary	3
5	kay	4
6	smith	1
7	lizzy	5
8	ben	6
9	ben	6
10	dylan	7
11	ben	6
12	toby	8
13	abby	9
14	mary	3
15	kay	4
16	kay	4
17	lizzy	5
18	ben	6
19	dylan	7
20	mary	3

Student smith is the first subject, and his Index value is one. The same student appears again in the sixth observation, which is given the same Index value. Now, this Index variable can be used to index the p array, in a similar fashion as demonstrated in previous programs:

```
data _f;
   set _f;
   subj = compress('p_'||students);
run;
proc sql noprint;
   select subj into :pnames separated by ' ' from _f;
quit;
%put &pnames;

proc mcmc data=a monitor=(p) diag=none stats=none outpost=a1
   plots=none seed=1;
   array p[&nuniq] &pnames;
   random u ~ n(0, sd=1) subject=students;
   p[index] = logistic(u);
   model general(0);
run;

proc print data=a1(obs=3);
run;
```

[4]The programming code that creates and adds the Index variable to the data set a keeps all variables from the original data set and does not discard them.

The first part of the DATA step and the PROC SQL call create array names for the p array that match the subject names in the students variable. It is not necessary to include these steps for the PROC MCMC program to work, but it makes the output more readable. The first DATA step steps through the _f data set and creates a subj variable that concatenates the prefix characters p_ with the names of the students. The PROC SQL calls put all the subj values to a macro called &pnames, which looks like the following:

```
p_smith p_john p_mary p_kay p_lizzy p_ben p_dylan p_toby p_abby
```

In the PROC MCMC program, the ARRAY statement defines an p array with size &nuniq (9), and use the macro variable &pnames to name the array elements. The P[INDEX] assignment statement uses the Index variable to find the correct array element to store the transformation. Figure 80.32 displays the first few observations of the OUTPOST=a1 data set.

Figure 80.32 First Few Observations of the Outpost Data Set

Obs	Iteration	p_smith	p_john	p_mary	p_kay	p_lizzy	p_ben	p_dylan	p_toby	p_abby	u_smith	u_john
1	1	0.5050	0.7334	0.5375	0.5871	0.6862	0.4944	0.5740	0.5991	0.6812	0.0198	1.0120
2	2	0.8678	0.5563	0.4254	0.7260	0.5280	0.4593	0.5797	0.5813	0.7360	1.8819	0.2261
3	3	0.9079	0.3351	0.6132	0.8031	0.4735	0.3135	0.7047	0.5273	0.6761	2.2887	-0.6854

Obs	u_mary	u_kay	u_lizzy	u_ben	u_dylan	u_toby	u_abby	LogReff	LogLike	LogPost
1	0.1504	0.3521	0.7826	-0.0225	0.2983	0.4019	0.7595	-9.5761	0	-9.5761
2	-0.3006	0.9742	0.1120	-0.1630	0.3213	0.3281	1.0253	-11.2370	0	-11.2370
3	0.4607	1.4058	-0.1060	-0.7837	0.8698	0.1092	0.7361	-13.1867	0	-13.1867

There are nine random-effects parameters (u_smith, u_john, and so on). There are nine elements of the p array (p_smith, p_john, and so on); each is the logit transformation of corresponding u elements.

You can use the same statements for subject variables that are numeric non-indexing variables. The following statements create a students variable that take large numbers that cannot be used as indices to an array. The rest of the program monitors functions of the effect u. The output is not displayed here.

```
data a;
   call streaminit(1);
   do i = 1 to 20;
      students = rand("poisson", 20);
      output;
   end;
   drop i;
run;

proc sql noprint;
   select count(distinct(students)) into :nuniq from a;
quit;
%put &nuniq;

proc freq data=a order=data noprint;
   tables students / out=_f(keep=students);
run;

data a(drop=n);
```

```
   set a;
   do i = 1 to nobs until(students=n);
      set _f(keep=students rename=(students=n)) point=i nobs=nobs;
      Index = i;
   end;
run;

data _f;
   set _f;
   subj = compress('p_'||students);
proc sql noprint;
   select subj into :pnames separated by ' ' from _f;
   quit;
%put &pnames;

proc mcmc data=a monitor=(p) diag=none stats=none outpost=a1
   plots=none seed=1;
   array p[&nuniq] &pnames;
   random u ~ n(0, sd=1) subject=students;
   p[index] = logistic(u);
   model general(0);
run;

proc print data=a1(obs=3);
run;
```

Spatial Prior

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mcmcspa.sas.)

You can use the conditional autoregressive Gaussian (NORMALCAR) prior in the RANDOM statement to model random effects that are spatially correlated. Suppose that the areas are indexed by the variable ID in a SAS data set. To model spatial dependence among areas, you can use the following statement:

```
random a ~ normalcar(neighbors=, num=, sd|var|prec=) subject=ID;
```

The NUM= option specifies the data set variable that indicates the number of neighbors for each ID, and the NEIGHBORS= option specifies the prefix (without quotation marks) of data variables that contain the adjacent IDs (neighbors) for each subject. The names of all neighboring variables must have this same prefix and must be followed by numbers 1, 2, . . ., N, where N is the maximum number of neighbors that a subject has in the data set. Missing values are allowed in the neighboring ID variables if a subject has fewer than N neighbors.

The default specification of the NORMALCAR prior corresponds to the full conditional distribution of each subject is the following CAR model,

$$\theta_i|\theta_{-i} \sim \text{normal}\left(\sum_{j\in N(i)} \theta_j/m_i, s^2\right)$$

where $N(i)$ is the set of neighbors of area i, m_i is the total number of neighbors of area i, and s is the standard deviation of the normal distribution.

You can also specify a CAR prior in which the variance is weighed by the number of neighbors:

$$\theta_i|\theta_{-i} \sim \text{normal}\left(\sum_{j\in N(i)} \theta_j/m_i, s^2/m_i\right)$$

To specify this prior, you use the following statement, where NumNei is the variable of number of neighbors and s2 is the variance parameter:

```
random a ~ normalcar(neighbor=, num=NumNei, var=s2/NumNei) subject=ID;
```

In most cases, the CAR prior is an improper prior because the corresponding covariance matrix of $\theta = \{\theta_i, \ldots, \theta_p\}$ is less than full rank. If the posterior distribution is proper, then using an improper prior is not an issue. A common practice is to re-center all the θ draws after each iteration (Banerjee, Carlin, and Gelfand 2004, p. 164). In effect, this re-centering sets the prior mean of the random effects to be zero. The centering step is performed by using the CENTER option in the RANDOM statement.

Here is an example that illustrates the use of the NORMALCAR prior distribution.

The following statements create a SAS data set that contains lung cancer data from a London Health Authority annual report (Thomas et al. 2004):

```
title 'Spatial Analysis';

data London;
   input ID N1-N9 NumNei O E Depriv;
   datalines;
  1   4   8   9  12  17   .   .   .   .  5    4   7.2090    1.233
  2   7  10  13  14   .   .   .   .   .  4    8   7.8144    8.162

   ... more lines ...

 44  22  24  25  26  31  32  41   .   .  7    7   6.3737    3.961
run;
```

The 44 wards are indicated by the ID variable. N1–N9 are variables of neighboring indices for each ward. For example, ward 1 is adjacent to five wards: 4, 8, 9, 12, and 17. The NumNei variable is the total number of neighbor wards that each ward has. The maximum number of neighbors wards is nine in the data set, so every observation must have nine neighboring ID variables. Missing data (".") are used in neighboring ID variables that have fewer than nine neighbors. The O and E variables are simulated observed and expected counts of lung cancer incidence. The Depriv variable is the socioeconomic variable for each ward.

The following random-effects model is considered,

$$\begin{aligned} \mathsf{O}_i &\sim \text{Poisson}(\mu_i) \\ \log(\mu_i) &= \log(\mathsf{E}_i) + \alpha + \beta\mathsf{Deprive}_i + b_i + h_i \end{aligned}$$

where the random effects b_i are given a spatial prior and h_i is a normal prior for each area. You can fit this model using the following PROC MCMC statements:

```
ods select none;
proc mcmc data=london seed=615926 nbi=10000 nmc=50000 thin=10
   plots=none outpost=londonpost;
   parms tau_b 0.5 tau_h 0.2;
   parms alpha 0 beta 0;
   prior tau: ~ gamma(0.5, is=0.0005);
   prior alpha ~ general(0);
   prior beta ~ n(0, prec=1e-5);
   random h ~ n(0, prec=tau_h) s=id;
   random b ~ normalcar(neighbors=n,num=numnei,prec=tau_b*numnei) s=id;
   mu=e*exp(alpha + beta*depriv + b + h);
   model o ~ poisson(mu);
run;
```

The two RANDOM statements specify the h_i and the b_i random effects; the NORMALCAR prior is assigned to b_i. In the NORMALCAR prior, NEIGHBORS=N specifies the prefix "n" for all the neighboring ID variables, NUM=NUMNEI specifies the number of neighbors that each ID has, and PREC=TAU_B*NUMNEI weights the precision parameter by the number of neighbors in the model.

The example assumes that the adjacency matrices have been created and stored in a SAS data set. More often, you might have the map polygon data in an .shp file. In that case, you can use the %NEIGHBOR autocall macro to import simple polygons and to compute the adjacent units and number of neighbors for each spatial unit:

```
%neighbor("map.shp", IDVAR=ID, OUTNBRS=neighbor, OUTADJ=adjacent);
```

The first argument is the path to the shape file, which must be specified in quotation marks. The IDVAR= option specifies the ID variable that identifies the sites or the units. The OUTNBRS= and OUTADJ= options save the neighborhood and adjacency information, respectively, to two SAS data sets. The default values for the IDVAR=, OUTNBRS=, and OUTADJ= options are ID, Neighbor, and Adjacent, respectively. You can combine these SAS data sets with covariates and response information before you use PROC MCMC to fit the spatial model.

Floating Point Errors and Overflows

When performing a Markov chain Monte Carlo simulation, you must calculate a proposed jump and an objective function (usually a posterior density). These calculations might lead to arithmetic exceptions and overflows. A typical cause of these problems is parameters with widely varying scales. If the posterior variances of your parameters vary by more than a few orders of magnitude, the numerical stability of the optimization problem can be severely reduced and can result in computational difficulties. A simple remedy is to rescale all the parameters so that their posterior variances are all approximately equal. Changing the SCALE= option might help if the scale of your parameters is much different than one. Another source of numerical instability is highly correlated parameters. Often a model can be reparameterized to reduce the posterior correlations between parameters.

If parameter rescaling does not help, consider the following actions:

- provide different initial values or try a different seed value
- use boundary constraints to avoid the region where overflows might happen
- change the algorithm (specified in programming statements) that computes the objective function

Problems Evaluating Code for Objective Function

The initial values must define a point for which the programming statements can be evaluated. However, during simulation, the algorithm might iterate to a point where the objective function cannot be evaluated. If you program your own likelihood, priors, and hyperpriors by using SAS statements and the GENERAL function in the MODEL, PRIOR, AND HYPERPRIOR statements, you can specify that an expression cannot be evaluated by setting the value you pass back through the GENERAL function to missing. This tells the PROC MCMC that the proposed set of parameters is invalid, and the proposal will not be accepted. If you use the shorthand notation that the MODEL, PRIOR, AND HYPERPRIOR statements provide, this error checking is done for you automatically.

Long Run Times

PROC MCMC can take a long time to run for problems with complex models, many parameters, or large input data sets. Although the techniques used by PROC MCMC are some of the best available, they are not guaranteed to converge or proceed quickly for all problems. Ill-posed or misspecified models can cause the algorithms to use more extensive calculations designed to achieve convergence, and this can result in longer run times. You should make sure that your model is specified correctly, that your parameters are scaled to the same order of magnitude, and that your data reasonably match the model that you are specifying.

To speed general computations, you should check over your programming statements to minimize the number of unnecessary operations. For example, you can use the proportional kernel in the priors or the likelihood and not add constants in the densities. You can also use the BEGINCNST and ENDCNST to reduce unnecessary computations on constants, and the BEGINNODATA and ENDNODATA statements to reduce observation-level calculations.

Reducing the number of blocks (the number of the PARMS statements) can speed up the sampling process. A single-block program is approximately three times faster than a three-block program for the same number of iterations. On the other hand, you do not want to put too many parameters in a single block, because blocks with large size tend not to produce well-mixed Markov chains.

If some parameters satisfy the conditional independence assumption, such as in the random-effects models or latent variable models, consider using the RANDOM statement to model these parameters. This statement takes advantage of the conditional independence assumption and can sample a larger number of parameters at a more efficient pace.

Slow or No Convergence

If the simulator is slow or fails to converge, you can try changing the model as follows:

- Change the number of Monte Carlo iterations (NMC=), or the number of burn-in iterations (NBI=), or both. Perhaps the chain just needs to run a little longer. Note that after the simulation, you can always use the DATA step or the FIRSTOBS data set option to throw away initial observations where the algorithm has not yet burned in, so it is not always necessary to set NBI= to a large value.

- Increase the number of tuning. The proposal tuning can often work better in large models (models that have more parameters) with larger values of NTU=. The idea of tuning is to find a proposal distribution that is a good approximation to the posterior distribution. Sometimes 500 iterations per tuning phase (the default) is not sufficient to find a good approximating covariance.

- Change the initial values to more feasible starting values. Sometimes the proposal tuning starts badly if the initial values are too far away from the main mass of the posterior density, and it might not be able to recover.

- Use the PROPCOV= option to start the Markov chain at better starting values. With the PROPCOV=QUANEW option, PROC MCMC optimizes the object function and uses the posterior mode as the starting value of the Markov chain. In addition, a quadrature approximation to the posterior mode is used as the proposal covariance matrix. This option works well in many cases and can improve the mixing of the chain and shorten the tuning and burn-in time.

- Parameterize your model to include conjugacy, such as using the gamma prior on the precision parameter in a normal distribution or using an inverse gamma on the variance parameter. For a list of conjugate sampling methods that PROC MCMC supports, see the section "Conjugate Sampling" on page 6286.

- Change the blocking by using the PARMS statements. Sometimes poor mixing and slow convergence can be attributed to highly correlated parameters being in different parameter blocks.

- Modify the target acceptance rate. A target acceptance rate of about 25% works well for many multi-parameter problems, but if the mixing is slow, a lower target acceptance rate might be better.

- Change the initial scaling or the TUNEWT= option to possibly help the proposal tuning.

- Consider using a different proposal distribution. If from a trace plot you see that a chain traverses to the tail area and sometimes takes quite a few simulations before it comes back, you can consider using a t proposal distribution. You can do this by either using the PROC option PROPDIST=T or using a PARMS statement option T.

- Transform parameters and sample on a different scale. For example, if a parameter has a gamma distribution, sample on the logarithm scale instead. A parameter a that has a gamma distribution is equivalent to $\log(a)$ that has an egamma distribution, with the same distribution specification. For example, the following two formulations are equivalent:

```
parm a;
prior a ~ gamma(shape = 0.001, scale = 0.001);
```

and

```
parm la;
prior la ~ egamma(shape = 0.001, scale = 0.001);
a = exp(la);
```

See "Example 80.6: Nonlinear Poisson Regression Models" on page 6406 and "Example 80.20: Using a Transformation to Improve Mixing" on page 6487. You can also use the logit transformation on parameters that have uniform(0, 1) priors. This prior is often used on probability parameters. The logit

transformation is as follows: $q = \log(\frac{p}{1-p})$. The distribution on q is the Jacobian of the transformation: $\exp(-q)(1 + \exp(-q))^{-2}$.

Again, the following two formulations are equivalent:

```
parm p;
prior p ~ uniform(0, 1);
```

and

```
parm q;
lp = -q - 2 * log(1 + exp(-q));
prior q ~ general(lp);
p = 1/(1+exp(-q));
```

Precision of Solution

In some applications, PROC MCMC might produce parameter values that are not precise enough. Usually, this means that there were not enough iterations in the simulation. At best, the precision of MCMC estimates increases with the square of the simulation sample size. Autocorrelation in the parameter values deflate the precision of the estimates. For more information about autocorrelations in Markov chains, see the section "Autocorrelations" on page 174 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Handling Error Messages

PROC MCMC does not have a debugger. This section covers a few ways to debug and resolve error messages.

Using the PUT Statement

Adding the PUT statement often helps to find errors in a program. The following statements produce an error:

```
data a;
run;

proc mcmc data=a seed=1;
   parms sigma lt w;

   beginnodata;
   prior sigma ~ unif(0.001,100);
   s2 = sigma*sigma;
   prior lt ~ gamma(shape=1, iscale=0.001);
   t = exp(lt);
   c = t/s2;
   d = 1/(s2);
   prior w ~ gamma(shape=c, iscale=d);
   endnodata;
   model general(0);
run;
```

```
ERROR: PROC MCMC is unable to generate an initial value for the
       parameter w.  The first parameter in the prior distribution is
       missing.
```

To find out why the shape parameter c is missing, you can add the put statement and examine all the calculations that lead up to the assignment of c:

```
proc mcmc data=a seed=1;
   parms sigma lt w;

   beginnodata;
   prior sigma ~ unif(0.001,100);
   s2 = sigma*sigma;
   prior lt ~ gamma(shape=1, iscale=0.001);
   t = exp(lt);
   c = t/s2;
   d = 1/(s2);
   put c= t= s2= lt=; /* display the values of these symbols. */
   prior w ~ gamma(shape=c, iscale=d);
   endnodata;

   model general(0);
run;
```

In the log file, you see the following:

```
c=. t=. s2=. lt=.
c=. t=. s2=2500.0500003 lt=1000
c=. t=. s2=2500.0500003 lt=1000
ERROR: PROC MCMC is unable to generate an initial value for the parameter w.
       The first parameter in the prior distribution is missing.
```

You can ignore the first few lines. They are the results of initial set up by PROC MCMC. The last line is important. The variable c is missing because t is the exponential of a very large number, 1000, in lt. The value 1000 is assigned to lt by PROC MCMC because none was given. The gamma prior with shape of 1 and inverse scale of 0.001 has mode 0 (see "Standard Distributions" on page 6289 for more details). PROC MCMC avoids starting the Markov chain at the boundary of the support of the distribution, and it uses the mean value here instead. The mean of the gamma prior is 1000, hence the problem. You can change how the initial value is generated by using the PROC statement INIT=RANDOM. Remember to take out the put statement once you identify the problem. Otherwise, you will see a voluminous output in the log file.

Using the HYPER Statement

You can use the HYPER statement to narrow down possible errors in the prior distribution specification. With multiple PRIOR statements in a program, you might see the following error message if one of the prior distributions is not specified correctly:

```
ERROR: The initial prior parameter specifications must yield log
       of positive prior density values.
```

This message is displayed when PROC MCMC detects an error in the prior distribution calculation but cannot pinpoint the specific parameter at fault. It is frequently, although not necessarily, associated with parameters that have GENERAL or DGENERAL distributions. If you have a complicated model with many PRIOR statements, finding the parameter at fault can be time consuming. One way is to change a subset of the PRIOR statements to HYPER statements. The two statements are treated the same in PROC MCMC and the simulation is not affected, but you get a different message if the hyperprior distributions are calculated incorrectly:

```
ERROR: The initial hyperprior parameter specifications must yield
       log of positive hyperprior density values.
```

This message can help you identify more easily which distributions are producing the error, and you can then use the PUT statement to further investigate.

Computational Resources

It is impossible to estimate how long it will take for a general Markov chain to converge to its stationary distribution. It takes a skilled and thoughtful analysis of the chain to decide whether it has converged to the target distribution and whether the chain is mixing rapidly enough. In some cases, you might be able to estimate how long a particular simulation might take. The running time of a program that does not have RANDOM statements is approximately linear to the following factors: the number of samples in the input data set, the number of simulations, the number of blocks in the program, and the speed of your computer. For an analysis that uses a data set of size *nsamples*, a simulation length of *nsim*, and a block design of *nblocks*, PROC MCMC evaluates the log-likelihood function the following number of times, excluding the tuning phase:

$$nsamples \times nsim \times nblocks$$

The faster your computer evaluates a single log-likelihood function, the faster this program runs. Suppose you have *nsamples* equal to 200, *nsim* equal to 55,000, and *nblocks* equal to 3. PROC MCMC evaluates the log-likelihood function approximately 3.3×10^7 times. If your computer can evaluate the log likelihood for one observation 10^6 times per second, this program takes approximately a half a minute to run. If you want to increase the number of simulations five-fold, the run time increases approximately five-fold.

Each RANDOM statement adds one pass through the input data at each iteration. If the Metropolis algorithm is used to sample the random-effects parameter, the conditional density (objective function) is calculated twice per pass through the data, which requires a computational resource that is approximately equivalent to adding two blocks of parameters.

Of course, larger problems take longer than shorter ones, and if your model is amenable to frequentist treatment, then one of the other SAS procedures might be more suitable. With “regular” likelihoods and a lot of data, the results of standard frequentist analysis are often asymptotically equivalent to a Bayesian approach. If PROC MCMC requires too much CPU time, then perhaps another SAS/STAT tool would be suitable.

Displayed Output

This section describes the output that PROC MCMC displays. For a quick reference of all ODS table names, see the section "ODS Table Names" on page 6373. ODS tables are arranged under four groups, which are listed in the following sections: "Model and Data Related ODS Tables" on page 6369, "Sampling Related ODS Tables" on page 6369, "Posterior Statistics Related ODS Tables" on page 6370, "Convergence Diagnostics Related ODS Tables" on page 6371, and "Optimization Related ODS Tables" on page 6373.

Model and Data Related ODS Tables

Missing Data Information Table

The "Missing Data Information" table (ODS table name MISSDATAINFO) displays the name of the response variable that contains missing values, the number of missing observations, the corresponding observation indices in the input data set, and the sampling method used in the simulation for the missing values.

Number of Observation Table

The "NObs" table (ODS table name NOBS) shows the number of observations that is in the data set and the number of observations that is used in the analysis. By default, observations with missing values are not used (see the section "Handling of Missing Data" on page 6352 for more details). This table is displayed by default.

Parameters

The "Parameters" table (ODS table name Parameters) shows the name of each parameter, the block number of each parameter, the sampling method used for the block, the initial values, and the prior or hyperprior distributions. This table is displayed by default.

REObsInfo

The "Random Effect Observation Information" table (ODS table name REObsInfo) lists the name of the random effect, each subject value, the number of observations in each subject, and their corresponding observation indices in the input data set. You can request this table by specifying the REOBSINFO option.

REParameters

The "REParameters" table (ODS table name REParameters) lists the name of the random effect, sampling algorithm, the subject variable, the number of subjects, unique values of the subject variable, and the prior distribution. This table is displayed by default if a RANDOM statement is used in the program.

Sampling Related ODS Tables

Burn-In History

The "Burn-In History" table (ODS table name BurnInHistory) shows the scales and acceptance rates for each parameter block in the burn-in phase. The table is not displayed by default and can be requested by specifying the option MCHISTORY=BRIEF | DETAILED.

Parameters Initial Value Table

The "Parameters Initial" table (ODS table name ParametersInit) shows the value of each parameter after the tuning phase. This table is not displayed by default and can be requested by specifying the option INIT=PINIT.

Posterior Samples

The "Posterior Samples" table (ODS table name PosteriorSample) stores posterior draws of all parameters. It is not printed by PROC MCMC. You can create an ODS output data set of the chain by specifying the following:

```
ODS OUTPUT PosteriorSample = SAS-data-set;
```

Sampling History

The "Sampling History" table (ODS table name SamplingHistory) shows the scales and acceptance rates for each parameter block in the main sampling phase. The table is not displayed by default and can be requested by specifying the option MCHISTORY=BRIEF | DETAILED.

Tuning Covariance

The "Tuning Covariance" table (ODS table name TuneCov) shows the proposal covariance matrices for each parameter block after the tuning phase. The table is not displayed by default and can be requested by specifying the option INIT=PINIT. For more details about proposal tuning, see the section "Tuning the Proposal Distribution" on page 6283.

Tuning History

The "Tuning History" table (ODS table name TuningHistory) shows the number of tuning phases used in establishing the proposal distribution. The table also displays the scales and acceptance rates for each parameter block at each of the tuning phases. For more information about the self-adapting proposal tuning algorithm used by PROC MCMC, see the section "Tuning the Proposal Distribution" on page 6283. The table is not displayed by default and can be requested by specifying the option MCHISTORY=BRIEF | DETAILED.

Tuning Probability Vector

The "Tuning Probability" table (ODS table name TuneP) shows the proposal probability vector for each discrete parameter block (when the option DISCRETE=GEO is specified and the geometric proposal distribution is used for discrete parameters) after the tuning phase. The table is not displayed by default and can be requested by specifying the option INIT=PINIT. For more information about proposal tuning, see the section "Tuning the Proposal Distribution" on page 6283.

Posterior Statistics Related ODS Tables

PROC MCMC calculates some essential posterior statistics and outputs them to a number of ODS tables that you can request and save individually. For details of the calculations, see the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Summary and Interval Statistics

The "Posterior Summaries and Intervals" table (ODS table name PostSumInt) contains a summary of basic point and interval statistics for each parameter. The table lists the number of posterior samples, the posterior mean and standard deviation estimates, and the 95% HPD interval estimates. This table is displayed by default.

Summary Statistics

The "Posterior Summaries" table (ODS table name PostSummaries) contains basic statistics for each parameter. The table lists the number of posterior samples, the posterior mean and standard deviation estimates, and the percentile estimates. The table is not displayed by default and can be requested by specifying the option STATISTICS=SUMMARY.

Correlation Matrix

The "Posterior Correlation Matrix" table (ODS table name Corr) contains the posterior correlation of model parameters. The table is not displayed by default and can be requested by specifying the option STATISTICS=CORR.

Covariance Matrix

The "Posterior Covariance Matrix" table (ODS table name Cov) contains the posterior covariance of model parameters. The table is not displayed by default and can be requested by specifying the option STATISTICS=COV.

Deviance Information Criterion

The "Deviance Information Criterion" table (ODS table name DIC) contains the DIC of the model. The table is not displayed by default and can be requested by specifying the option DIC. For details of the calculations, see the section "Deviance Information Criterion (DIC)" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Interval Statistics

The "Posterior Intervals" table (ODS table name PostIntervals) contains the equal-tail and highest posterior density (HPD) interval estimates for each parameter. The default α value is 0.05, and you can change it to other levels by using the STATISTICS= option. The table is not displayed by default and can be requested by specifying the option STATISTICS=INTERVAL.

Convergence Diagnostics Related ODS Tables

PROC MCMC has convergence diagnostic tests that check for Markov chain convergence. PROC MCMC produces a number of ODS tables that you can request and save individually. For details in calculation, see the section "Statistical Diagnostic Tests" on page 166 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Autocorrelation

The "Autocorrelations" table (ODS table name AUTOCORR) contains the first-order autocorrelations of the posterior samples for each parameter. The "Parameter" column states the name of the parameter. By default, PROC MCMC displays lag 1, 5, 10, and 50 estimates of the autocorrelations. You can request different autocorrelations by using the DIAGNOSTICS=AUTOCORR(LAGS=) option. The table is not displayed by default and can be requested by specifying the option DIAGNOSTICS=AUTOCORR.

Effective Sample Size

The "Effective Sample Sizes" table (ODS table name ESS) calculates the effective sample size of each parameter. See the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The table is displayed by default.

Monte Carlo Standard Errors

The "Monte Carlo Standard Errors" table (ODS table name MCSE) calculates the standard errors of the posterior mean estimate. See the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The table is not displayed by default and can be requested by specifying the option DIAGNOSTICS=MCSE.

Geweke Diagnostics

The "Geweke Diagnostics" table (ODS table name Geweke) lists the result of the Geweke diagnostic test. See the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The table is not displayed by default and can be requested by specifying the option DIAGNOSTICS=GEWEKE.

Heidelberger-Welch Diagnostics

The "Heidelberger-Welch Diagnostics" table (ODS table name Heidelberger) lists the result of the Heidelberger-Welch diagnostic test. The test is consisted of two parts: a stationary test and a half-width test. See the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The table is not displayed by default and can be requested by specifying DIAGNOSTICS=HEIDEL.

Raftery-Lewis Diagnostics

The "Raftery-Lewis Diagnostics" table (ODS table name Raftery) lists the result of the Raftery-Lewis diagnostic test. See the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for more details. The table is not displayed by default and can be requested by specifying DIAGNOSTICS=RAFTERY.

Summary and Interval Statistics for Prediction

The "Posterior Summaries and Intervals for Prediction" table (ODS table name PredSumInt) contains a summary of basic point and interval statistics for each prediction. The table lists the number of posterior samples, the posterior mean and standard deviation estimates, and the 95% HPD interval estimates. This table is displayed by default if any PREDDIST statement is used in the program.

Summary Statistics for Prediction

The "Posterior Summaries for Prediction" table (ODS table name PredSummaries) contains basic statistics for each prediction. The table lists the number of posterior samples, the posterior mean and standard deviation estimates, and the percentile estimates. This table is not displayed by default and can be requested by specifying the option STATISTIC=SUMMARY.

Interval Statistics for Prediction

The "Posterior Intervals for Prediction" table (ODS table name PredIntervals) contains the equal-tail and highest posterior density (HPD) interval estimates for each prediction. The default α value is 0.05, and you can change it to other levels by using the STATISTICS option in a PREDDIST statement, or the STATISTICS= option in the PROC MCMC statement if the option is not specified in a statement. This table is not displayed by default and can be requested by specifying the option STATISTIC=INTERVAL.

Optimization Related ODS Tables

PROC MCMC can perform optimization on the joint posterior distribution. This is requested by the PROPCOV= option. The most commonly used optimization method is the quasi-Newton method: PROPCOV=QUANEW(ITPRINT). The ITPRINT option displays the ODS tables, listed as follows:

Input Options

The "Input Options" table (ODS table name InputOptions) lists optimization options used in the procedure.

Optimization Start

The "Optimization Start" table (ODS table name ProblemDescription) shows the initial state of the optimization.

Iteration History

The "Iteration History" table (ODS table name IterHist) shows iteration history of the optimization.

Optimization Results

The "Optimization Results" table (ODS table name IterStop) shows the results of the optimization, includes information about the number of function calls, and the optimized objective function, which is the joint log posterior density.

Convergence Status

The "Convergence Status" table (ODS table name ConvergenceStatus) shows whether the convergence criterion is satisfied.

Parameters Value After Optimization Table

The "Parameter Values After Optimization" table (ODS table name OptiEstimates) lists the parameter values that maximize the joint log posterior. These are the maximum a posteriori point estimates, and they are used to start the Markov chain.

Covariance Matrix After Optimization Table

The "Proposal Covariance" table (ODS table name OptiCov) lists covariance matrices for each block parameter by using quadrature approximation at the posterior mode. These covariance matrices are used in the proposal distribution.

ODS Table Names

PROC MCMC assigns a name to each table it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 80.53. For more information about ODS, see Chapter 24, "Statistical Graphics Using ODS."

Table 80.53 ODS Tables Produced in PROC MCMC

ODS Table Name	Description	Statement or Option
AutoCorr	Autocorrelation statistics for each parameter	DIAGNOSTICS=AUTOCORR
BurnInHistory	History of burn-in phase sampling	MCHISTORY=BRIEF | DETAILED
ConvergenceStatus	Optimization convergence status	PROPCOV=method(ITPRINT)
Corr	Correlation matrix of the posterior samples	STATS=CORR
Cov	Covariance matrix of the posterior samples	STATS=COV
DIC	Deviance information criterion	DIC
ESS	Effective sample size for each parameter	Default
MCSE	Monte Carlo standard error for each parameter	DIAGNOSTICS=MCSE
Geweke	Geweke diagnostics for each parameter	DIAGNOSTICS=GEWEKE
Heidelberger	Heidelberger-Welch diagnostics for each parameter	DIAGNOSTICS=HEIDEL
InputOptions	Optimization input table	PROPCOV=method(ITPRINT)
IterHist	Optimization iteration history	PROPCOV=method(ITPRINT)
IterStop	Optimization results table	PROPCOV=method(ITPRINT)
MissDataInfo	Response variable, number of missing observations, missing observation indices, and sampling algorithm	Default with sampling of missing values
NObs	Number of observations	Default
OptiEstimates	Parameter values after either optimization	PROPCOV=method(ITPRINT)
OptiCov	Covariance used in proposal distribution after optimization	PROPCOV=method(ITPRINT)
Parameters	Summary of the PARMS, BLOCKING, PRIOR, sampling method, and initial value specification	Default
ParametersInit	Parameter values after the tuning phase	INIT=PINIT
PosteriorSample	Posterior samples for each parameter	(For ODS output data set only)
PostIntervals	Equal-tail and HPD intervals for each parameter	STATISTICS=INTERVAL
PostSumInt	Basic posterior statistics for each parameter, including sample size, mean, standard deviation, and HPD intervals	Default
PostSummaries	Basic posterior statistics for each parameter, including sample size, mean, standard deviation, and percentiles	STATISTICS=SUMMARY

Table 80.53 *continued*

ODS Table Name	Description	Statement or Option
PredIntervals	Equal-tail and HPD intervals for each prediction	STATISTICS=INTERVAL in PREDDIST statement
PredSumInt	Basic posterior statistics for each prediction, including sample size, mean, standard deviation, and HPD intervals	Default with any PREDDIST statement
PredSummaries	Basic posterior statistics for each prediction	STATISTICS=SUMMARY in PREDDIST statement
ProblemDescription	Optimization table	PROPCOV=method(ITPRINT)
REObsInfo	Random effect, subject values, number of observations in each unique subject value, and corresponding observation indices	REOBSINFO
REParameters	Random effect, sampling method, subject variable, number of subjects, unique values of the subject variable, and prior distribution of the random effect	Default with any RANDOM statement
Raftery	Raftery-Lewis diagnostics for each parameter	DIAGNOSTICS=RAFTERY
SamplingHistory	History of main phase sampling	MCHISTORY=BRIEF | DETAILED
TuneCov	Proposal covariance matrix (for continuous parameters) after the tuning phase	INIT=PINIT
TuneP	Proposal probability vector (for discrete parameters) after the tuning phase	INIT=PINIT and DISCRETE=GEO
TuningHistory	History of proposal distribution tuning	MCHISTORY=BRIEF | DETAILED

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC MCMC generates are listed in Table 80.54.

Table 80.54 Graphs Produced by PROC MCMC

ODS Graph Name	Plot Description	Statement and Option
ADPanel	Autocorrelation function and density panel	PLOTS=(AUTOCORR DENSITY)
AutocorrPanel	Autocorrelation function panel	PLOTS=AUTOCORR
AutocorrPlot	Autocorrelation function plot	PLOTS(UNPACK)=AUTOCORR
DensityPanel	Density panel	PLOTS=DENSITY
DensityPlot	Density plot	PLOTS(UNPACK)=DENSITY
TAPanel	Trace and autocorrelation function panel	PLOTS=(TRACE AUTOCORR)
TADPanel	Trace, density, and autocorrelation function panel	PLOTS=(TRACE AUTOCORR DENSITY)
TDPanel	Trace and density panel	PLOTS=(TRACE DENSITY)
TracePanel	Trace panel	PLOTS=TRACE
TracePlot	Trace plot	PLOTS(UNPACK)=TRACE

Examples: MCMC Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 80.1: Simulating Samples From a Known Density

(View the complete code for this example (mcmcex1.sas) in the example repository.)

This example illustrates how you can obtain random samples from a known function. The target distributions are the normal distribution (a standard distribution) and a mixture of the normal distributions (a nonstandard distribution). For more information, see the sections "Standard Distributions" on page 6289 and "Specifying a New Distribution" on page 6305). This example also shows how you can use PROC MCMC to estimate an integral (area under a curve). Monte Carlo simulation is data-independent; hence, you do not need an input data set from which to draw random samples from the desired distribution.

Sampling from a Normal Density

When you run a simulation without an input data set, the posterior distribution is the same as the prior distribution. Hence, if you want to generate samples from a distribution, you declare the distribution in the PRIOR statement and set the likelihood function to a constant. Although there is no contribution from any data set variable to the likelihood calculation, you still must specify a data set and the MODEL statement needs a distribution. You can input an empty data set and use the GENERAL function to provide a flat likelihood. The following statements generate 10,000 samples from a standard normal distribution:

```
title 'Simulating Samples from a Normal Density';
data x;
run;

ods graphics on;
proc mcmc data=x outpost=simout seed=23 nmc=10000 diagnostics=none;
   ods exclude nobs;
   parm alpha 0;
   prior alpha ~ normal(0, sd=1);
   model general(0);
run;
```

The ODS GRAPHICS ON statement enables ODS Graphics. The PROC MCMC statement specifies the input and output data sets, a random number seed, and the size of the simulation sample. The STATISTICS= option displays only the summary and interval statistics. The ODS EXCLUDE statement excludes the display of the NObs table. PROC MCMC draws independent samples from the normal distribution directly (see Output 80.1.1). Therefore, the simulation does not require any tuning, and PROC MCMC omits the default burn-in phrase.

Output 80.1.1 Parameters Information

Simulating Samples from a Normal Density

The MCMC Procedure

Parameters				
Block	Parameter	Sampling Method	Initial Value	Prior Distribution
1	alpha	Direct	0	normal(0, sd=1)

The summary statistics (Output 80.1.2) are what you would expect from a standard normal distribution.

Output 80.1.2 MCMC Summary and Interval Statistics from a Normal Target Distribution

Simulating Samples from a Normal Density

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
alpha	10000	0.00195	0.9949	-1.9664	1.9302

The trace plot (Output 80.1.3) shows perfect mixing with no autocorrelation in the lag plot. This is expected

because these are independent draws.

Output 80.1.3 Diagnostics Plots for α

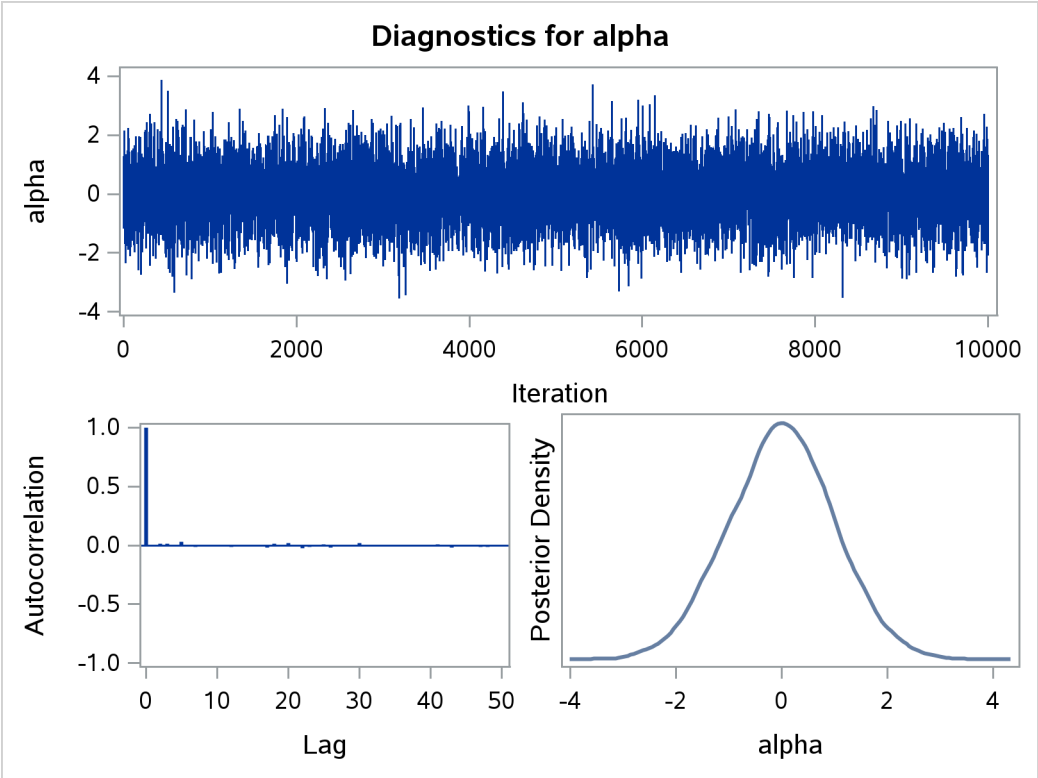

You can overlay the estimated kernel density with the true density to visually compare the densities, as displayed in Output 80.1.4. To create the kernel comparison plot, you first call PROC KDE (see Chapter 73, "The KDE Procedure") to obtain a kernel density estimate of the posterior density on alpha. Then you evaluate a grid of alpha values on a normal density. The following statements evaluate kernel density and compute the corresponding normal density:

```
proc kde data=simout;
   ods exclude inputs controls;
   univar alpha /out=sample;
run;

data den;
   set sample;
   alpha = value;
   true  = pdf('normal', alpha, 0, 1);
   keep alpha density true;
run;
```

Next you plot the two curves on top of each other by using PROC SGPLOT (see Chapter 24, "Statistical Graphics Using ODS") as follows:

```
proc sgplot data=den;
   yaxis label="Density";
   series y=density x=alpha / legendlabel = "MCMC Kernel";
   series y=true x=alpha / legendlabel = "True Density";
   discretelegend;
run;
```

Output 80.1.4 shows the result. You can see that the kernel estimate and the true density are very similar to each other.

Output 80.1.4 Estimated Density versus the True Density

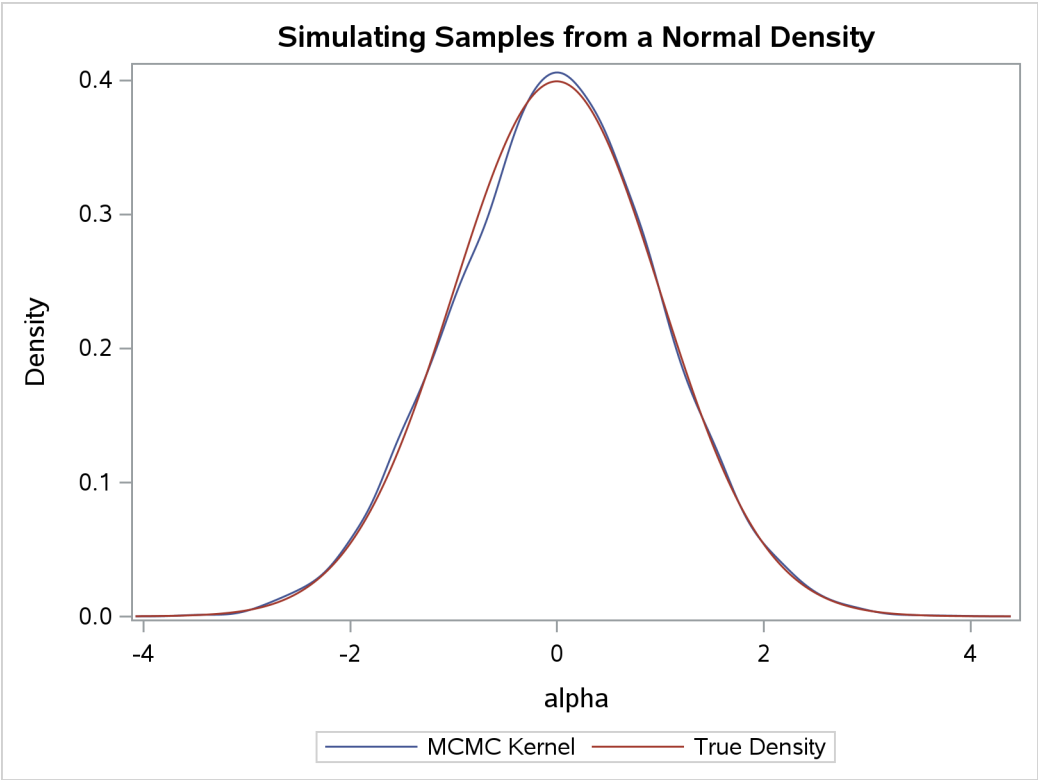

Density Visualization Macro

In programs that do not involve any data set variables, PROC MCMC samples directly from the (joint) prior distributions of the parameters. The modification makes the sampling from a known distribution more efficient and more precise. For example, you can write simple programs, such as the following macro, to understand different aspects of a prior distribution of interest, such as its moments, intervals, shape, spread, and so on:

```
%macro density(dist=, seed=0);
   %let savenote = %sysfunc(getoption(notes));
   options nonotes;
   title "&dist distribution.";
   data _a;
   run;
```

```
   ods select densitypanel postsumint;
   proc mcmc data=_a nmc=10000 diag=none nologdist
      plots=density seed=&seed;
      parms alpha;
      prior alpha ~ &dist;
      model general(0);
   run;

   proc datasets nolist;
      delete _a;
   run;
   options &savenote;
%mend;

%density(dist=beta(4, 12), seed=1);
```

The macro %density creates an empty data set, invokes PROC MCMC, draws 10,000 samples from a beta(4, 12) distribution, displays summary and interval statistics, and generates a kernel density plot. Summary and interval statistics from the beta distribution are displayed in Output 80.1.5.

Output 80.1.5 Beta Distribution Statistics

beta(4, 12) distribution.

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
alpha	10000	0.2494	0.1039	0.0657	0.4590

The distribution is displayed in Output 80.1.6.

Output 80.1.6 Density Plot

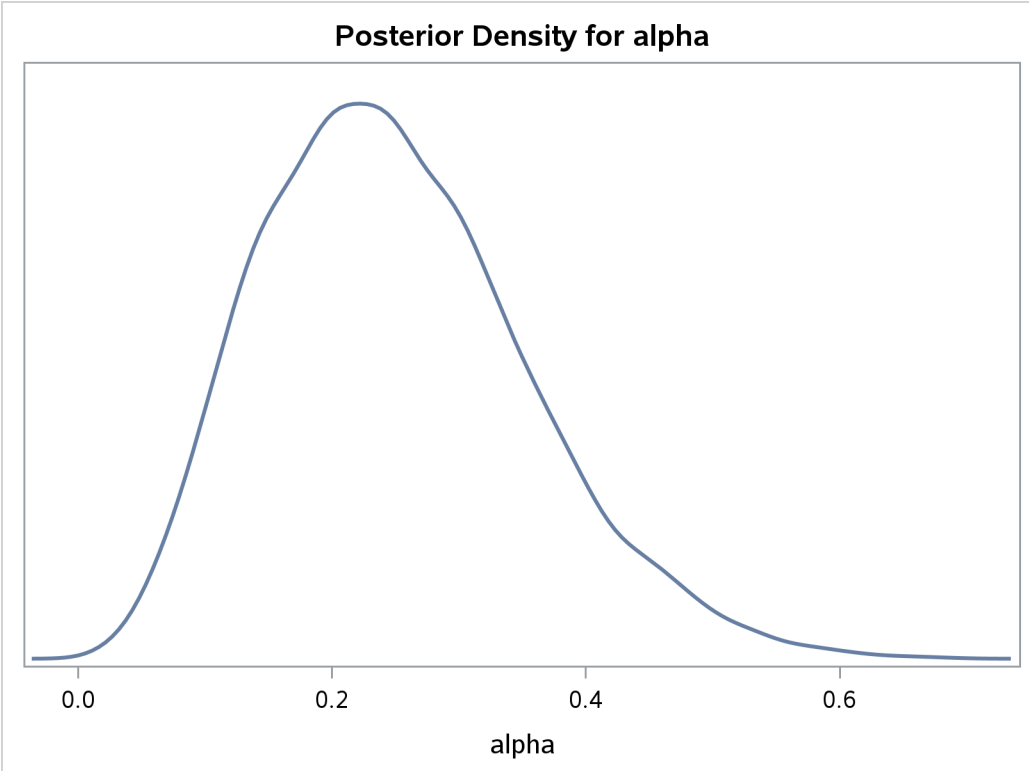

Calculation of Integrals

One advantage of MCMC methods is to estimate any integral under the curve of a target distribution. This can be done fairly easily using the MCMC procedure. Suppose you are interested in estimating the following cumulative probability:

$$\int_{\alpha=0}^{1.3} \phi(\alpha|0,1)d\alpha$$

To estimate this integral, PROC MCMC draws samples from the distribution and counts the portion of the simulated values that fall within the desired range of [0, 1.3]. This becomes a Monte Carlo estimate of the integral. The following statements simulate samples from a standard normal distribution and estimate the integral:

```
proc mcmc data=x outpost=simout seed=23 nmc=10000 nologdist
   monitor=(int) diagnostics=none;
   ods select postsumint;
   parm alpha 0;
   prior alpha ~ normal(0, sd=1);
   int = (0 <= alpha <= 1.3);
   model general(0);
run;
```

The ODS SELECT statement displays the posterior summary statistics table. The MONITOR= option outputs analysis on the variable int (the integral estimate). The STATISTICS= option computes the summary statistics.

The NOLOGDIST option omits the calculation of the log of the prior distribution at each iteration, shortening the simulation time[5]. The INT assignment statement sets int to be 1 if the simulated alpha value falls between 0 and 1.3, and 0 otherwise. PROC MCMC supports the usage of the IF-ELSE logical control if you need to account for more complex conditions. Output 80.1.7 displays the estimated integral value:

Output 80.1.7 Monte Carlo Integral from a Normal Distribution

beta(4, 12) distribution.

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
int	10000	0.4079	0.4915	0	1.0000

In this simulation, 4079 samples fall between 0 and 1.3, making the expected probability 0.4079. In this example, you can verify the actual cumulative probability by calling the CDF function in the DATA step:

```
data _null_;
   int = cdf("normal", 1.3, 0, 1) - cdf("normal", 0, 0, 1);
   put int=;
run;
```

The value is 0.4032.

Sampling from a Mixture of Normal Densities

Suppose you are interested in generating samples from a three-component mixture of normal distributions, with the density specified as follows:

$$p(\alpha) = 0.3 \cdot \phi(-3, \sigma = 2) + 0.4 \cdot \phi(2, \sigma = 1) + 0.3 \cdot \phi(10, \sigma = 4)$$

You can either specify the distribution directly or use a latent variable approach to generate samples from the normal mixture.

To specify the normal mixture density directly in PROC MCMC, you need to construct the function because the normal mixture distribution is not one of the standard distributions that PROC MCMC supports. The following statements generate random samples from the normal mixture density:

```
title 'Simulating Samples from a Mixture of Normal Densities';
data x;
run;

proc mcmc data=x outpost=simout seed=1234 nmc=30000;
   ods select TADpanel;
   parm alpha 0.3;
   lp = logpdf('normalmix', alpha, 3, 0.3, 0.4, 0.3, -3, 2, 10, 2, 1, 4);
   prior alpha  ~ general(lp);
```

[5]In this example, the NOLOGDIST option saves only a fraction of the time. But in more complex simulation schemes that involve a larger number of distributions and parameters, the time reduction could be significant.

```
    model general(0);
  run;
```

The ODS SELECT statement displays the diagnostic plots. All other tables, such as the NObs tables, are excluded. The PROC MCMC statement uses the input data set X, saves output to the Simout data set, sets a random number seed, and draws 30,000 samples.

The LP assignment statement evaluates the log density of alpha at the mixture density, using the SAS function LOGPDF. The number 3 after alpha in the LOGPDF function indicates that the density is a three-component normal mixture. The following three numbers, 0.3, 0.4, and 0.3, are the weights in the mixture; –3, 2, and 10 are the means; 2, 1, and 4 are the standard deviations. The PRIOR statement assigns this log density function to alpha as its prior. Note that the GENERAL function interprets the density on the log scale, and not the original scale–you must use the LOGPDF function, not the PDF function. Output 80.1.8 displays the results. The kernel density clearly shows three modes.

Output 80.1.8 Plots of Posterior Samples from a Mixture Normal Distribution

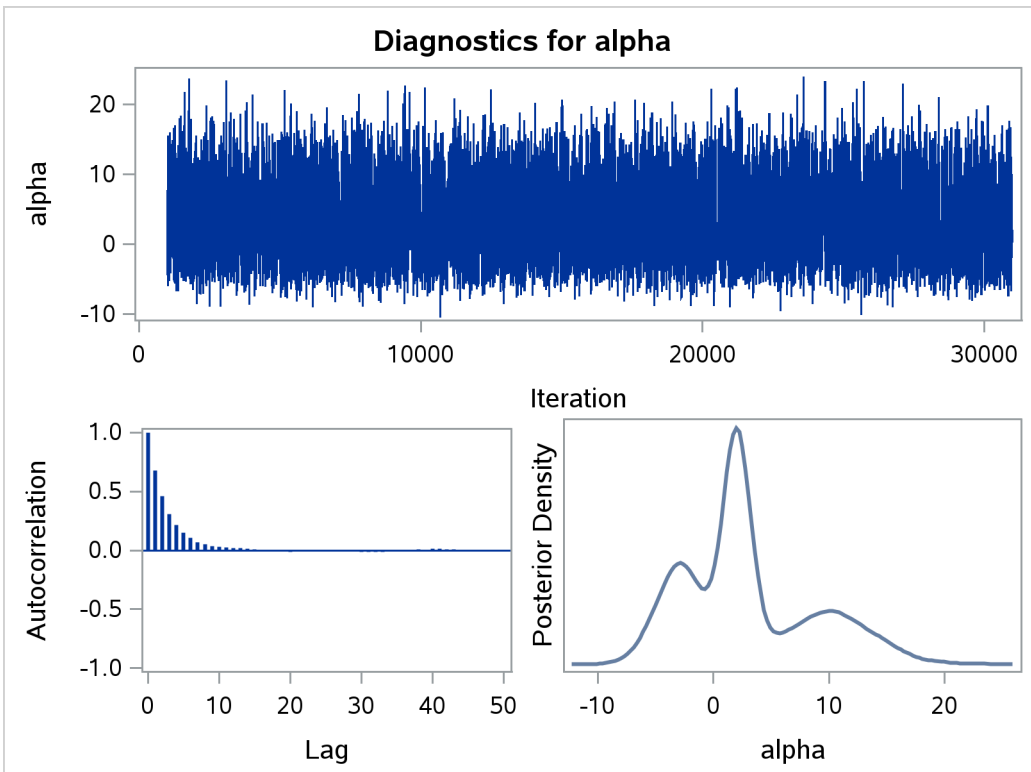

Alternatively, the normal mixture distribution can also decomposed into a marginal distribution for the component (call it Z) and a conditional model of the response variable Y given Z, as

$$
\begin{aligned}
z &\sim \text{categorical}(p_1, p_2, \ldots, p_K) \\
y|z &\sim \text{normal}(\mu_z, \sigma_z^2)
\end{aligned}
$$

where K is the total number of mixture components, z is the component indicator, and μ_z and σ_z^2 are the model parameters for the zth component.

PROC MCMC supports a categorical distribution that can be used to model the discrete random variable for components. You can use PRIOR statements to specify a normal mixture distribution and generate samples accordingly:

```
proc mcmc data=x outpost=simout_m seed=1234 nmc=30000;
   array p[3] (0.3 0.4 0.3);
   array mu[3] (-3 2 10);
   array sd[3] (2 1 4);
   parm z alpha;
   prior z ~ table(p);
   prior alpha ~ normal(mu[z], sd=sd[z]);
   model general(0);
run;
```

The ARRAY statements define one array p for the mixture weights, one array mu for the means of the normal distributions, and one array sd for the corresponding standard deviations. The PRIOR statements specify a categorical prior on the parameter z and a conditional normal prior on alpha. The mean and standard deviation of the alpha parameter depend on the component indicator z. No output is created.

You can use the following set of statements, which are similar to the previous example, to overlay the estimated kernel density with the true density. The comparison is shown in Output 80.1.9.

```
proc kde data=simout_m;
   ods exclude inputs controls;
   univar alpha /out=sample;
run;

data den;
   set sample;
   alpha = value;
   true  = pdf('normalmix', alpha, 3, 0.3, 0.4, 0.3, -3, 2, 10, 2, 1, 4);
   keep alpha density true;
run;

proc sgplot data=den;
   yaxis label="Density";
   series y=density x=alpha /
      legendlabel = "MCMC Kernel - Latent Variable Approach";
   series y=true x=alpha / legendlabel = "True Density";
   discretelegend;
run;
```

Output 80.1.9 Estimated Density (Latent Variable Approach) versus the True Density

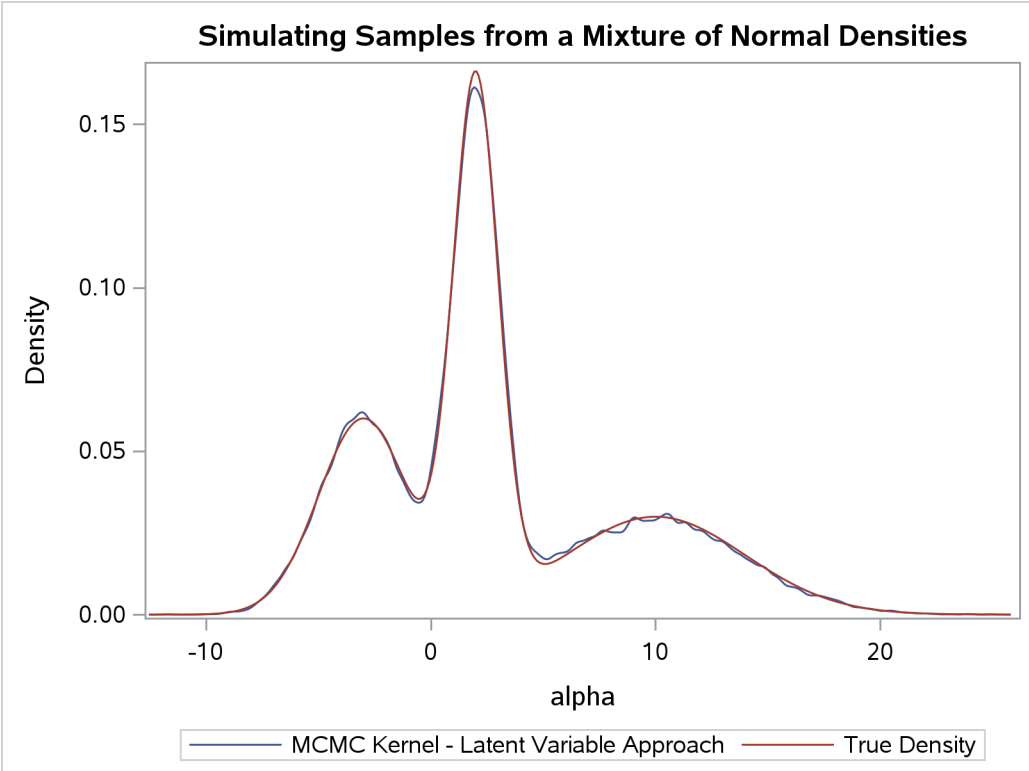

Example 80.2: Box-Cox Transformation

(View the complete code for this example (mcmcex2.sas) in the example repository.)

Box-Cox transformations (Box and Cox 1964) are often used to find a power transformation of a dependent variable to ensure the normality assumption in a linear regression model. This example illustrates how you can use PROC MCMC to estimate a Box-Cox transformation for a linear regression model. Two different priors on the transformation parameter λ are considered: a continuous prior and a discrete prior. You can estimate the probability of λ being 0 with a discrete prior but not with a continuous prior. The IF-ELSE statements are demonstrated in the example.

Using a Continuous Prior on λ

The following statements create a SAS data set with measurements of y (the response variable) and x (a single dependent variable):

```
title 'Box-Cox Transformation, with a Continuous Prior on Lambda';
data boxcox;
   input y x @@;
   datalines;
10.0  3.0  72.6  8.3  59.7  8.1  20.1  4.8  90.1  9.8   1.1  0.9
78.2  8.5  87.4  9.0   9.5  3.4   0.1  1.4   0.1  1.1  42.5  5.1
57.0  7.5   9.9  1.9   0.5  1.0 121.1  9.9  37.5  5.9  49.5  6.7
```

```
   ... more lines ...

 2.6  1.8  58.6  7.9  81.2  8.1  37.2  6.9
;
```

The Box-Cox transformation of y takes on the form of:

$$y(\lambda) = \begin{cases} \frac{y^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0; \\ \log(y) & \text{if } \lambda = 0. \end{cases}$$

The transformed response $y(\lambda)$ is assumed to be normally distributed:

$$y_i(\lambda) \sim \text{normal}(\beta_0 + \beta_1 x_i, \sigma^2)$$

The likelihood with respect to the original response y_i is as follows:

$$p(y_i|\lambda, \beta, \sigma^2, x_i) \propto \phi(y_i|\beta_0 + \beta_1 x_i, \sigma^2) \cdot J(\lambda, y_i)$$

where $J(\lambda, y_i)$ is the Jacobian:

$$J(\lambda, y) = \begin{cases} y_i^{\lambda-1} & \text{if } \lambda \neq 0; \\ 1/y_i & \text{if } \lambda = 0. \end{cases}$$

And on the log-scale, the Jacobian becomes:

$$\log(J(\lambda, y)) = \begin{cases} (\lambda - 1) \cdot \log(y_i) & \text{if } \lambda \neq 0; \\ -\log(y_i) & \text{if } \lambda = 0. \end{cases}$$

There are four model parameters: $\lambda, \boldsymbol{\beta} = \{\beta_0, \beta_1\}$, and σ^2. You can considering using a flat prior on $\boldsymbol{\beta}$ and a gamma prior on σ^2.

To consider only power transformations ($\lambda \neq 0$), you can use a continuous prior (for example, a uniform prior from –2 to 2) on λ. One issue with using a continuous prior is that you cannot estimate the probability of $\lambda = 0$. To do so, you need to consider a discrete prior that places positive probability mass on the point 0. See "Modeling $\lambda = 0$" on page 6390.

The following statements fit a Box-Cox transformation model:

```
ods graphics on;
proc mcmc data=boxcox nmc=50000 propcov=quanew seed=12567
          monitor=(lda);
   ods select PostSumInt TADpanel;
   parms beta0 0  beta1 0  lda 1 s2 1;

   beginnodata;
   prior beta: ~ general(0);
   prior s2 ~ gamma(shape=3, scale=2);
   prior lda ~ unif(-2,2);
   sd = sqrt(s2);
   endnodata;

   ys = (y**lda-1)/lda;
   mu = beta0+beta1*x;
   ll = (lda-1)*log(y)+lpdfnorm(ys, mu, sd);
   model general(ll);
run;
```

The PROPCOV= option initializes the Markov chain at the posterior mode and uses the estimated inverse Hessian matrix as the initial proposal covariance matrix. The MONITOR= option selects λ as the variable to report. The ODS SELECT statement displays the summary statistics table, the interval statistics table, and the diagnostic plots.

The PARMS statement puts all four parameters, β_0, β_1, λ, and σ^2, in a single block and assigns initial values to each of them. Three PRIOR statements specify previously stated prior distributions for these parameters. The assignment to sd transforms a variance to a standard deviation. It is better to place the transformation inside the BEGINNODATA and ENDNODATA statements to save computational time.

The assignment to the symbol ys evaluates the Box-Cox transformation of y, where mu is the regression mean and ll is the log likelihood of the transformed variable ys. Note that the log of the Jacobian term is included in the calculation of ll.

Summary statistics and interval statistics for lda are listed in Output 80.2.1.

Output 80.2.1 Box-Cox Transformation

Box-Cox Transformation, with a Continuous Prior on Lambda

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
lda	50000	0.4698	0.0288	0.4119	0.5249

The posterior mean of λ is 0.47, with a 95% equal-tail interval of $[0.42, 0.53]$ and a similar HPD interval. The preferred power transformation would be 0.5 (rounding λ up to the square root transformation).

Output 80.2.2 shows diagnostics plots for lda. The chain appears to converge, and you can proceed to make inferences. The density plot shows that the posterior density is relatively symmetric around its mean estimate.

Output 80.2.2 Diagnostic Plots for λ

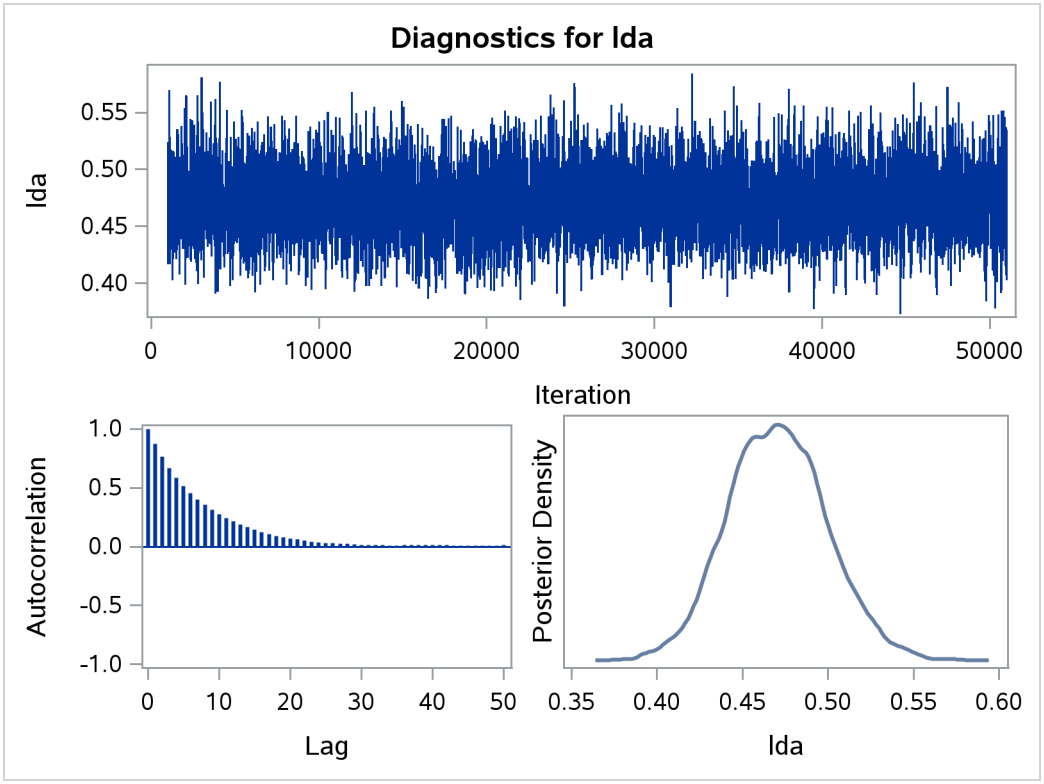

To verify the results, you can use PROC TRANSREG (see Chapter 126, "The TRANSREG Procedure") to find the estimate of λ.

```
proc transreg data=boxcox details pbo;
   ods output boxcox = bc;
   model boxcox(y / convenient lambda=-2 to 2 by 0.01) = identity(x);
   output out=trans;
run;
```

Output from PROC TRANSREG is shown in Output 80.2.5 and Output 80.2.4. PROC TRANSREG produces a similar point estimate of $\lambda = 0.46$, and the 95% confidence interval is shown in Output 80.2.5.

Output 80.2.3 Box-Cox Transformation Using PROC TRANSREG

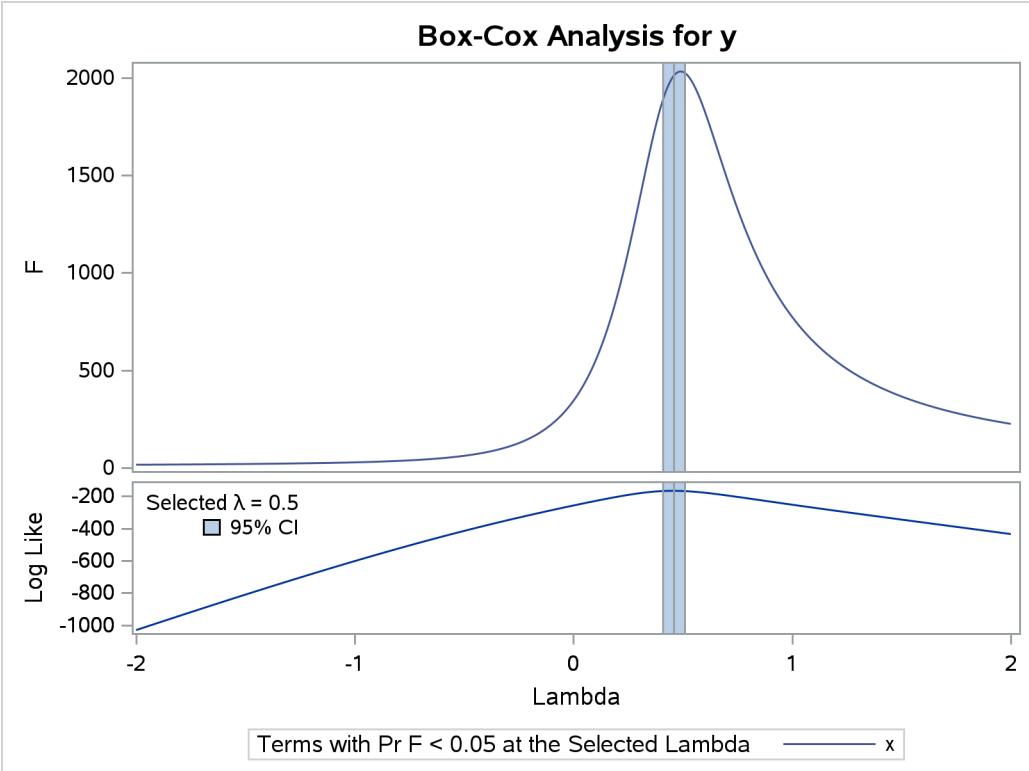

Output 80.2.4 Estimates Reported by PROC TRANSREG

Box-Cox Transformation, with a Continuous Prior on Lambda

The TRANSREG Procedure

Model Statement Specification Details				
Type	**DF**	**Variable**	**Description**	**Value**
Dep	1	BoxCox(y)	Lambda Used	0.5
			Lambda	0.46
			Log Likelihood	-167.0
			Conv. Lambda	0.5
			Conv. Lambda LL	-168.3
			CI Limit	-169.0
			Alpha	0.05
			Options	Convenient Lambda Used
Ind	1	Identity(x)	DF	1

The ODS data set Bc contains the 95% confidence interval estimates produced by PROC TRANSREG. This ODS table is rather large, and you want to see only the relevant portion. The following statements generate the part of the table that is important and display Output 80.2.5:

```
proc print noobs label data=bc(drop=rmse);
   title2 'Confidence Interval';
   where ci ne ' ' or abs(lambda - round(lambda, 0.5)) < 1e-6;
   label convenient = '00'x ci = '00'x;
run;
```

The estimated 90% confidence interval is $[0.41, 0.51]$, which is very close to the reported Bayesian credible intervals. The resemblance of the intervals is probably due to the noninformative prior that you used in this analysis.

Output 80.2.5 Estimated Confidence Interval on λ

Box-Cox Transformation, with a Continuous Prior on Lambda
Confidence Interval

Dependent	Lambda		R-Square	Log Likelihood	
BoxCox(y)	-2.00		0.14	-1030.56	
BoxCox(y)	-1.50		0.17	-810.50	
BoxCox(y)	-1.00		0.22	-602.53	
BoxCox(y)	-0.50		0.39	-415.56	
BoxCox(y)	0.00		0.78	-257.92	
BoxCox(y)	0.41		0.95	-168.40	*
BoxCox(y)	0.42		0.95	-167.86	*
BoxCox(y)	0.43		0.95	-167.46	*
BoxCox(y)	0.44		0.95	-167.19	*
BoxCox(y)	0.45		0.95	-167.05	*
BoxCox(y)	0.46		0.95	-167.04	<
BoxCox(y)	0.47		0.95	-167.16	*
BoxCox(y)	0.48		0.95	-167.41	*
BoxCox(y)	0.49		0.95	-167.79	*
BoxCox(y)	0.50	+	0.95	-168.28	*
BoxCox(y)	0.51		0.95	-168.89	*
BoxCox(y)	1.00		0.89	-253.09	
BoxCox(y)	1.50		0.79	-345.35	
BoxCox(y)	2.00		0.70	-435.01	

Modeling $\lambda = 0$

With a continuous prior on λ, you can get only a continuous posterior distribution, and this makes the probability of $\Pr(\lambda = 0|\text{data})$ equal to 0 by definition. To consider $\lambda = 0$ as a viable solution to the Box-Cox transformation, you need to use a discrete prior that places some probability mass on the point 0 and allows for a meaningful posterior estimate of $\Pr(\lambda = 0|\text{data})$.

This example uses a simulation study where the data are generated from an exponential likelihood. The simulation implies that the correct transformation should be the logarithm and λ should be 0. Consider the following exponential model:

$$y = \exp(x + \epsilon),$$

where $\epsilon \sim \text{normal}(0, 1)$. The transformed data can be fitted with a linear model:

$$\log(y) = x + \epsilon$$

The following statements generate a SAS data set with a gridded x and corresponding y:

```
title 'Box-Cox Transformation, Modeling Lambda = 0';
data boxcox;
   do x = 1 to 8 by 0.025;
      ly = x + normal(7);
      y = exp(ly);
      output;
   end;
run;
```

The log-likelihood function, after taking the Jacobian into consideration, is as follows:

$$\log p(y_i|\lambda, x_i) = \begin{cases} (\lambda-1)\log(y_i) - \frac{1}{2}\left(\log\sigma^2 + \frac{\left((y_i^\lambda-1)/\lambda - x_i\right)^2}{\sigma^2}\right) + C_1 & \text{if } \lambda \neq 0; \\ -\log(y_i) - \frac{1}{2}\left(\log\sigma^2 + \frac{(\log(y_i)-x_i)^2}{\sigma^2}\right) + C_2 & \text{if } \lambda = 0. \end{cases}$$

where C_1 and C_2 are two constants.

You can use the function DGENERAL to place a discrete prior on λ. The function is similar to the function GENERAL, except that it indicates a discrete distribution. For example, you can specify a discrete uniform prior from –2 to 2 using

```
prior lda ~ dgeneral(1, lower=-2, upper=2);
```

This places equal probability mass on five points, –2, –1, 0, 1, and 2. This prior might not work well here because the grid is too coarse. To consider smaller values of λ, you can sample a parameter that takes a wider range of integer values and transform it back to the λ space. For example, set alpha as your model parameter and give it a discrete uniform prior from –200 to 200. Then define λ as alpha/100 so λ can take values between –2 and 2 but on a finer grid.

The following statements fit a Box-Cox transformation by using a discrete prior on λ:

```
proc mcmc data=boxcox outpost=simout nmc=50000 seed=12567
          monitor=(lda);
   ods select PostSumInt;
   parms s2 1 alpha 10;
   beginnodata;
   prior s2 ~ gamma(shape=3, scale=2);
   if alpha=0 then lp = log(2);
      else lp = log(1);
   prior alpha ~ dgeneral(lp, lower=-200, upper=200);
   lda = alpha * 0.01;
   sd = sqrt(s2);
   endnodata;
   if alpha=0 then
      ll = -ly+lpdfnorm(ly, x, sd);
   else do;
      ys = (y**lda - 1)/lda;
      ll = (lda-1)*ly+lpdfnorm(ys, x, sd);
```

```
   end;
   model general(ll);
run;
```

There are two parameters, s2 and alpha, in the model. They are placed in a single PARMS statement so that they are sampled in the same block.

The parameter s2 takes a gamma distribution, and alpha takes a discrete prior. The IF-ELSE statements state that alpha takes twice as much prior density when it is 0 than otherwise. Note that on the original scale, $\Pr(\text{alpha} = 0) = 2 \cdot \Pr(\text{alpha} \neq 0)$. Translating that to the log scale, the densities become $\log(2)$ and $\log(1)$, respectively. The LDA assignment statement transforms alpha to the parameter of interest: lda takes values between –2 and 2. You can model lda on a even smaller scale by dividing alpha by a larger constant. However, an increment of 0.01 in the Box-Cox transformation is usually sufficient. The SD assignment statement calculates the square root of the variance term.

The log-likelihood function uses another set of IF-ELSE statements, separating the case of $\lambda = 0$ from the others. The formulas are stated previously. The output of the program is shown in Output 80.2.6.

Output 80.2.6 Box-Cox Transformation

Box-Cox Transformation, Modeling Lambda = 0

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
lda	50000	-0.00001	0.00199	0	0

From the summary statistics table, you see that the point estimate for λ is 0 and both of the 95% equal-tail and HPD credible intervals are 0. This strongly suggests that $\lambda = 0$ is the best estimate for this problem. In addition, you can also count the frequency of λ among posterior samples to get a more precise estimate on the posterior probability of λ being 0.

The following statements use PROC FREQ to produce Output 80.2.7 and Output 80.2.8:

```
proc freq data=simout;
   ods select onewayfreqs freqplot;
   tables lda /nocum plot=freqplot(scale=percent);
run;
ods graphics off;
```

Output 80.2.7 shows the frequency count table. An estimate of $\Pr(\lambda = 0|\text{data})$ is 96%. The conclusion is that the log transformation should be the appropriate transformation used here, which agrees with the simulation setup. Output 80.2.8 shows the histogram of λ.

Output 80.2.7 Frequency Counts of λ

Box-Cox Transformation, Modeling Lambda = 0

The FREQ Procedure

lda	Frequency	Percent
-0.0100	1011	2.02
0	48029	96.06
0.0100	960	1.92

Output 80.2.8 Histogram of λ

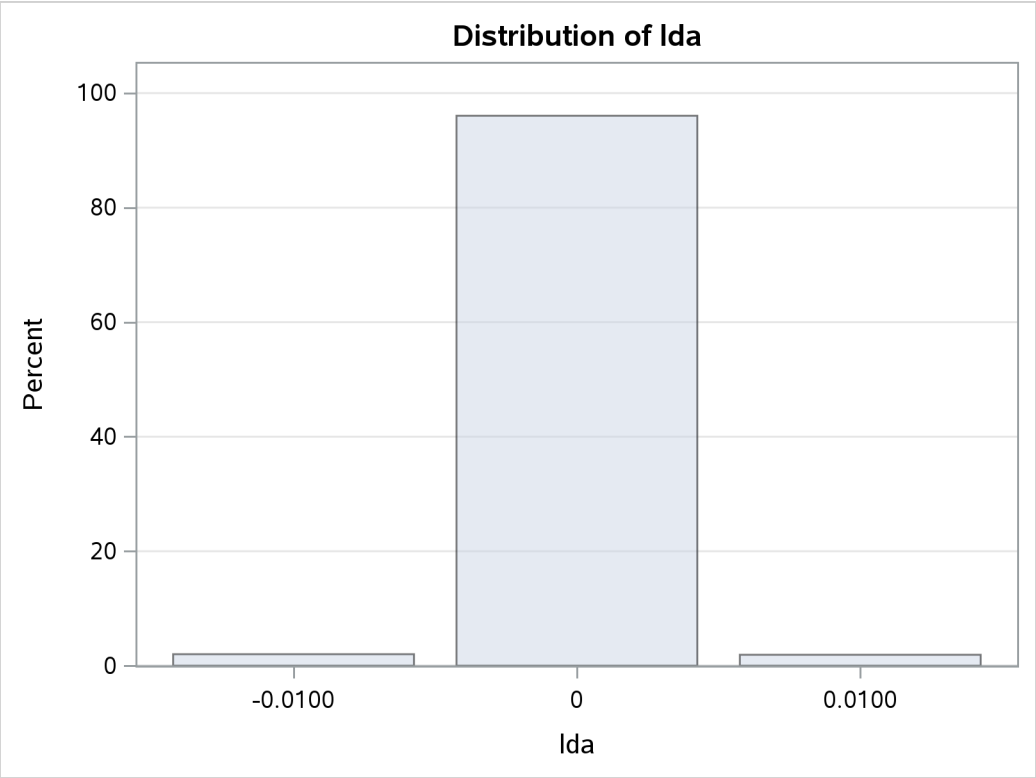

Example 80.3: Logistic Regression Model with a Diffuse Prior

(View the complete code for this example (mcmcex3.sas) in the example repository.)

This example illustrates how to fit a logistic regression model with a diffuse prior in PROC MCMC. You can also use the BAYES statement in PROC GENMOD. See Chapter 51, "The GENMOD Procedure."

The following statements create a SAS data set with measurements of the number of deaths, y, among n beetles that have been exposed to an environmental contaminant x:

```
title 'Logistic Regression Model with a Diffuse Prior';
data beetles;
   input n y x @@;
   datalines;
6  0  25.7   8  2  35.9   5  2  32.9   7  7  50.4   6  0  28.3
7  2  32.3   5  1  33.2   8  3  40.9   6  0  36.5   6  1  36.5
6  6  49.6   6  3  39.8   6  4  43.6   6  1  34.1   7  1  37.4
8  2  35.2   6  6  51.3   5  3  42.5   7  0  31.3   3  2  40.6
;
```

You can model the data points y_i with a binomial distribution,

$$y_i | p_i \sim \text{binomial}(n_i, p_i)$$

where p_i is the success probability and links to the regression covariate x_i through a logit transformation:

$$logit(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \alpha + \beta x_i$$

The priors on α and β are both diffuse normal:

$$\begin{aligned} \alpha &\sim \text{normal}(0, \text{var} = 10000) \\ \beta &\sim \text{normal}(0, \text{var} = 10000) \end{aligned}$$

These statements fit a logistic regression with PROC MCMC:

```
ods graphics on;
proc mcmc data=beetles ntu=1000 nmc=20000 propcov=quanew
          diag=(mcse ess) outpost=beetleout seed=246810;
   ods select PostSumInt mcse ess TADpanel;
   parms (alpha  beta) 0;
   prior alpha beta ~ normal(0, var = 10000);
   p = logistic(alpha + beta*x);
   model y ~ binomial(n,p);
run;
```

The key statement in the program is the assignment to p that calculates the probability of death. The SAS function LOGISTIC does the proper transformation. The MODEL statement specifies that the response variable, y, is binomially distributed with parameters n (from the input data set) and p. The summary statistics table, interval statistics table, the Monte Carlo standard error table, and the effective sample sizes table are shown in Output 80.3.1.

Output 80.3.1 MCMC Results

Logistic Regression Model with a Diffuse Prior

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
alpha	20000	-11.7688	2.0941	-15.9411	-7.7492
beta	20000	0.2919	0.0541	0.1901	0.4029

Logistic Regression Model with a Diffuse Prior

The MCMC Procedure

Monte Carlo Standard Errors			
Parameter	MCSE	Standard Deviation	MCSE/SD
alpha	0.0418	2.0941	0.0200
beta	0.00109	0.0541	0.0201

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
alpha	2507.0	7.9778	0.1253
beta	2478.5	8.0694	0.1239

The summary statistics table shows that the sample mean of the output chain for the parameter alpha is –11.77. This is an estimate of the mean of the marginal posterior distribution for the intercept parameter alpha. The estimated posterior standard deviation for alpha is 2.09. The two 95% credible intervals for alpha are both negative, which indicates with very high probability that the intercept term is negative. On the other hand, you observe a positive effect on the regression coefficient beta. Exposure to the environment contaminant increases the probability of death.

The Monte Carlo standard errors of each parameter are significantly small relative to the posterior standard deviations. A small MCSE/SD ratio indicates that the Markov chain has stabilized and the mean estimates do not vary much over time. Note that the precision in the parameter estimates increases with the square of the MCMC sample size, so if you want to double the precision, you must quadruple the MCMC sample size.

MCMC chains do not produce independent samples. Each sample point depends on the point before it. In this case, the correlation time estimate, read from the effective sample sizes table, is roughly 8. This means that it takes four observations from the MCMC output to make inferences about alpha with the same precision that you would get from using an independent sample. The effective sample size of around 2500 reflects this loss of efficiency. The coefficient beta has similar efficiency. You can often observe that some parameters have significantly better mixing (better efficiency) than others, even in a single Markov chain run.

Output 80.3.2 Plots for Parameters in the Logistic Regression Example

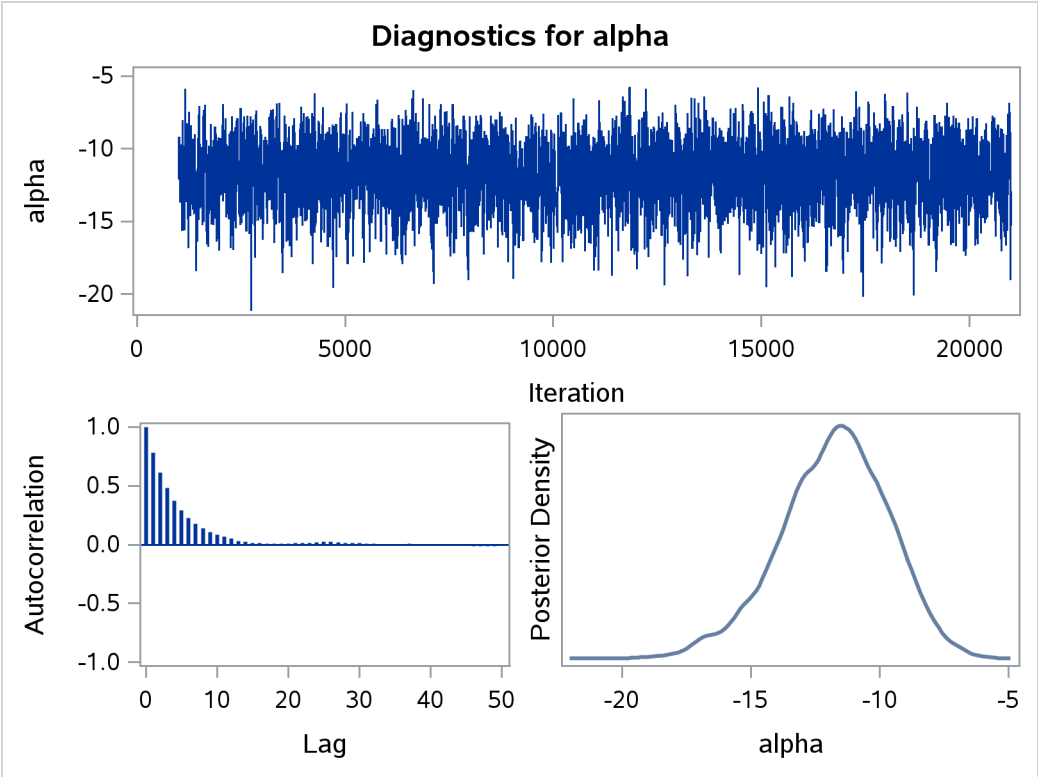

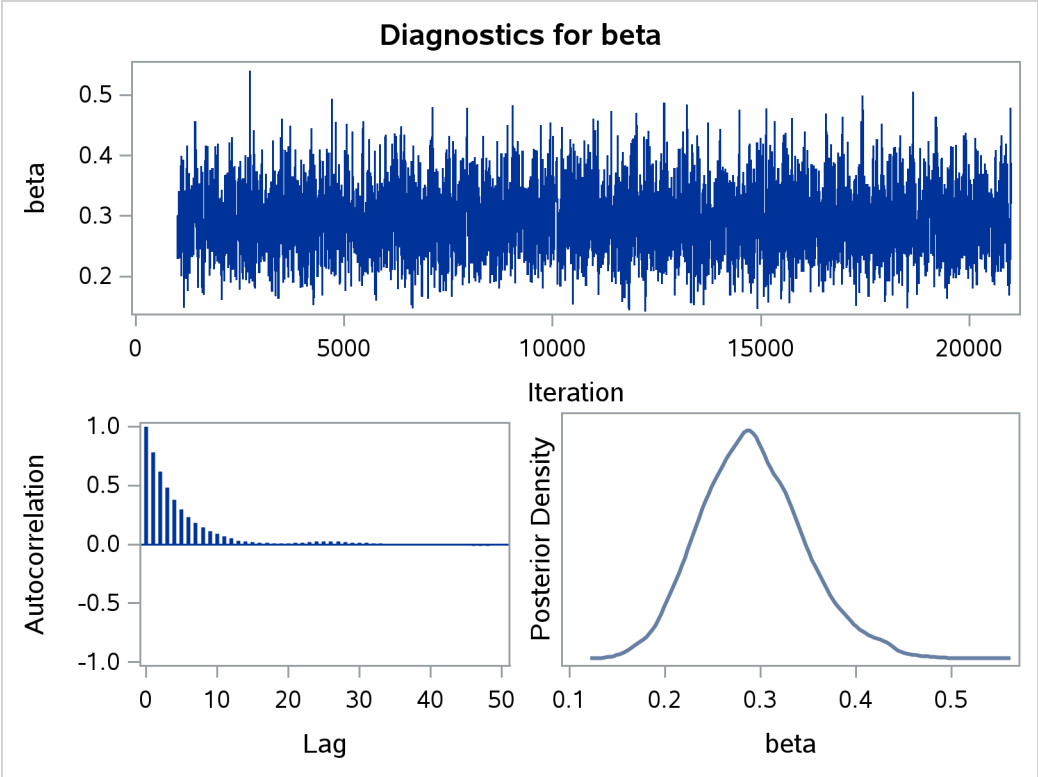

Trace plots and autocorrelation plots of the posterior samples are shown in Output 80.3.2. Convergence looks

good in both parameters; there is good mixing in the trace plot and quick drop-off in the ACF plot.

One advantage of Bayesian methods is the ability to directly answer scientific questions. In this example, you might want to find out the posterior probability that the environmental contaminant increases the probability of death—that is, $\Pr(\beta > 0|\mathbf{y})$. This can be estimated using the following steps:

```
proc format;
   value betafmt low-0 = 'beta <= 0' 0<-high = 'beta > 0';
run;

proc freq data=beetleout;
   tables beta /nocum;
   format beta betafmt.;
run;
```

Output 80.3.3 Frequency Counts

Logistic Regression Model with a Diffuse Prior

The FREQ Procedure

beta	Frequency	Percent
beta > 0	20000	100.00

All of the simulated values for β are greater than zero, so the sample estimate of the posterior probability that $\beta > 0$ is 100%. The evidence overwhelmingly supports the hypothesis that increased levels of the environmental contaminant increase the probability of death.

If you are interested in making inference based on any quantities that are transformations of the random variables, you can either do it directly in PROC MCMC or by using the DATA step after you run the simulation. Transformations sometimes can make parameter inference quite formidable using direct analytical methods, but with simulated chains, it is easy to compute chains for any set of parameters. Suppose you are interested in the lethal dose and want to estimate the level of the covariate x that corresponds to a probability of death, p. Abbreviate this quantity as ldp. In other words, you want to solve the logit transformation with a fixed value p. The lethal dose is as follows:

$$\mathsf{ldp} = \frac{\log\left(\frac{p}{1-p}\right) - \alpha}{\beta}$$

You can obtain an estimate of any ldp by using the posterior mean estimates for α and β. For example, lp95, which corresponds to $p = 0.95$, is calculated as follows:

$$\mathsf{lp95} = \frac{\log\left(\frac{0.95}{1-0.95}\right) + 11.77}{0.29} = 50.79$$

where –11.77 and 0.29 are the posterior mean estimates of α and β, respectively, and 50.79 is the estimated lethal dose that leads to a 95% death rate.

While it is easy to obtain the point estimates, it is harder to estimate other posterior quantities, such as the standard deviation directly. However, with PROC MCMC, you can trivially get estimates of any posterior quantities of lp95. Consider the following program in PROC MCMC:

```
proc mcmc data=beetles ntu=1000 nmc=20000 propcov=quanew
          outpost=beetleout seed=246810 plot=density
          monitor=(pi30 ld05 ld50 ld95);
   ods select PostSumInt densitypanel;
   parms (alpha  beta) 0;
   begincnst;
      c1 = log(0.05 / 0.95);
      c2 = -c1;
   endcnst;

   beginnodata;
   prior alpha beta ~ normal(0, var = 10000);
   pi30 = logistic(alpha + beta*30);
   ld05 = (c1 - alpha) / beta;
   ld50 = - alpha / beta;
   ld95 = (c2 - alpha) / beta;
   endnodata;
   pi = logistic(alpha + beta*x);
   model y ~ binomial(n,pi);
run;
ods graphics off;
```

The program estimates four additional posterior quantities. The three lpd quantities, ld05, ld50, and ld95, are the three levels of the covariate that kills 5%, 50%, and 95% of the population, respectively. The predicted probability when the covariate x takes the value of 30 is pi30. The MONITOR= option selects the quantities of interest. The PLOTS= option selects kernel density plots as the only ODS graphical output, excluding the trace plot and autocorrelation plot.

Programming statements between the BEGINCNST and ENDCNST statements define two constants. These statements are executed once at the beginning of the simulation. The programming statements between the BEGINNODATA and ENDNODATA statements evaluate the quantities of interest. The symbols, pi30, ld05, ld50, and ld95, are functions of the parameters alpha and beta only. Hence, they should not be processed at the observation level and should be included in the BEGINNODATA and ENDNODATA statements. Output 80.3.4 lists the posterior summary and Output 80.3.5 shows the density plots of these posterior quantities.

Output 80.3.4 PROC MCMC Results

Logistic Regression Model with a Diffuse Prior

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
pi30	20000	0.0524	0.0252	0.0126	0.1022
ld05	20000	29.9310	1.8731	26.2171	33.2648
ld50	20000	40.3780	0.9371	38.5334	42.1808
ld95	20000	50.8251	2.5327	46.2350	55.7957

The posterior mean estimate of lp95 is 50.82, which is close to the estimate of 50.79 by using the posterior mean estimates of the parameters. With PROC MCMC, in addition to the mean estimate, you can get the

standard deviation, quantiles, and interval estimates at any level of significance.

From the density plots, you can see, for example, that the sample distribution for π_{30} is skewed to the right, and almost all of your posterior belief concerning π_{30} is concentrated in the region between zero and 0.15.

Output 80.3.5 Density Plots of Quantities of Interest in the Logistic Regression Example

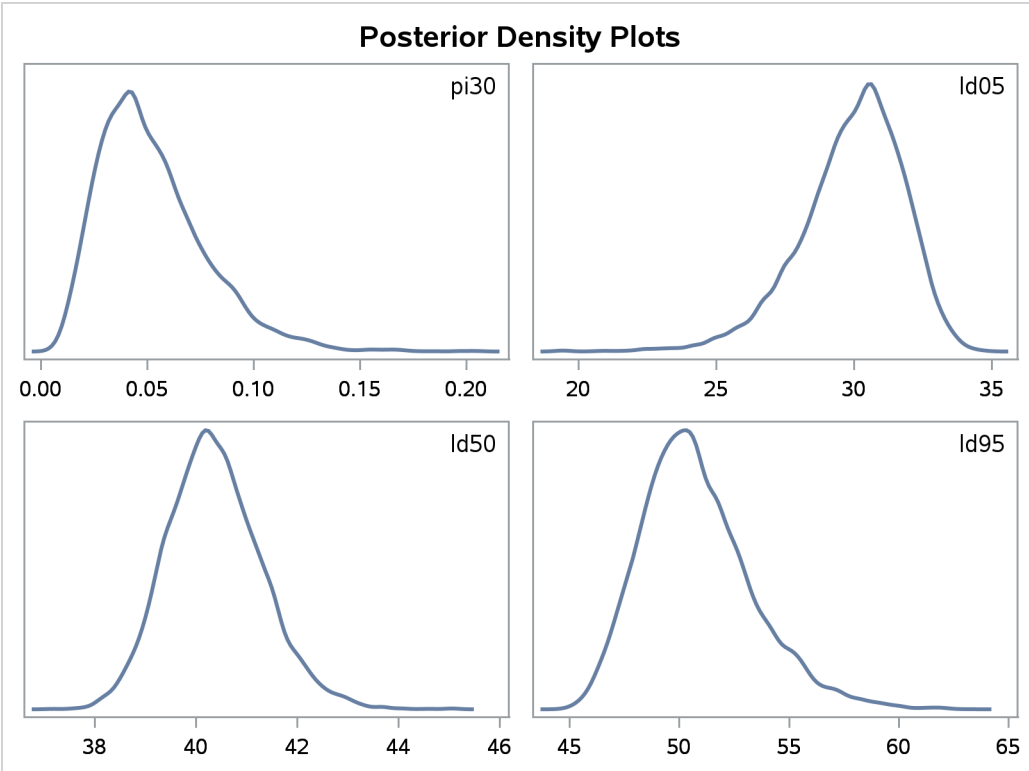

It is easy to use the DATA step to calculate these quantities of interest. The following DATA step uses the simulated values of α and β to create simulated values from the posterior distributions of ld05, ld50, ld95, and π_{30}:

```
data transout;
   set beetleout;
   pi30 = logistic(alpha + beta*30);
   ld05 = (log(0.05 / 0.95) - alpha) / beta;
   ld50 = (log(0.50 / 0.50) - alpha) / beta;
   ld95 = (log(0.95 / 0.05) - alpha) / beta;
run;
```

Subsequently, you can use SAS/INSIGHT, or the UNIVARIATE, CAPABILITY, or KDE procedures to analyze the posterior sample. If you want to regenerate the default ODS graphs from PROC MCMC, see "Regenerating Diagnostics Plots" on page 6339.

Example 80.4: Logistic Regression Model with Jeffreys' Prior

(View the complete code for this example (mcmcex4.sas) in the example repository.)

A controlled experiment was run to study the effect of the rate and volume of air inspired on a transient reflex vasoconstriction in the skin of the fingers. Thirty-nine tests under various combinations of rate and volume of air inspired were obtained (Finney 1947). The result of each test is whether or not vasoconstriction occurred. Pregibon (1981) uses this set of data to illustrate the diagnostic measures he proposes for detecting influential observations and to quantify their effects on various aspects of the maximum likelihood fit. The following statements create the data set Vaso:

```
title 'Logistic Regression Model with Jeffreys Prior';
data vaso;
   input vol rate resp @@;
   lvol = log(vol);
   lrate = log(rate);
   ind = _n_;
   cnst = 1;
   datalines;
3.7  0.825  1  3.5  1.09  1  1.25  2.5   1  0.75  1.5  1
0.8  3.2    1  0.7  3.5   1  0.6   0.75  0  1.1   1.7  0
0.9  0.75   0  0.9  0.45  0  0.8   0.57  0  0.55  2.75 0
0.6  3.0    0  1.4  2.33  1  0.75  3.75  1  2.3  1.64  1
3.2  1.6    1  0.85 1.415 1  1.7   1.06  0  1.8  1.8   1
0.4  2.0    0  0.95 1.36  0  1.35  1.35  0  1.5  1.36  0
1.6  1.78   1  0.6  1.5   0  1.8   1.5   1  0.95 1.9   0
1.9  0.95   1  1.6  0.4   0  2.7   0.75  1  2.35 0.03  0
1.1  1.83   0  1.1  2.2   1  1.2   2.0   1  0.8  3.33  1
0.95 1.9    0  0.75 1.9   0  1.3   1.625 1
;
```

The variable resp represents the outcome of a test. The variable lvol represents the log of the volume of air intake, and the variable lrate represents the log of the rate of air intake. You can model the data by using logistic regression. You can model the response with a binary likelihood:

$$\text{resp}_i \sim \text{binary}(p_i)$$

with

$$p_i = \frac{1}{1 + \exp(-(\beta_0 + \beta_1 \text{lvol}_i + \beta_2 \text{lrate}_i))}$$

Let $\mathbf{X}$ be the design matrix in the regression. Jeffreys' prior for this model is

$$p(\boldsymbol{\beta}) \propto |\mathbf{X}'\mathbf{M}\mathbf{X}|^{1/2}$$

where $\mathbf{M}$ is a 39 by 39 matrix with off-diagonal elements being 0 and diagonal elements being $p_i(1 - p_i)$. For details on Jeffreys' prior, see "Jeffreys' Prior" on page 151 in Chapter 8, "Introduction to Bayesian Analysis Procedures." You can use a number of matrix functions, such as the determinant function, in PROC MCMC to construct Jeffreys' prior. The following statements illustrate how to fit a logistic regression with Jeffreys' prior:

```
%let n = 39;
proc mcmc data=vaso nmc=10000 outpost=mcmcout seed=17;
   ods select PostSumInt;

   array beta[3] beta0 beta1 beta2;
   array m[&n, &n];
   array x[1] / nosymbols;
   array xt[3, &n];
   array xtm[3, &n];
   array xmx[3, 3];
   array p[&n];

   parms beta0 1 beta1 1 beta2 1;

   begincnst;
      if (ind eq 1) then do;
         rc = read_array("vaso", x, "cnst", "lvol", "lrate");
         call transpose(x, xt);
         call zeromatrix(m);
      end;
   endcnst;

   beginnodata;
   call mult(x, beta, p);              /* p = x * beta */
   do i = 1 to &n;
      p[i] = 1 / (1 + exp(-p[i]));     /* p[i] = 1/(1+exp(-x*beta)) */
      m[i,i] = p[i] * (1-p[i]);
   end;
   call mult (xt, m, xtm);             /* xtm = xt * m        */
   call mult (xtm, x, xmx);            /* xmx = xtm * x       */
   call det (xmx, lp);                 /* lp = det(xmx)       */
   lp = 0.5 * log(lp);                 /* lp = -0.5 * log(lp) */
   prior beta: ~ general(lp);
   endnodata;

   model resp ~ bern(p[ind]);
run;
```

The first ARRAY statement defines an array beta with three elements: beta0, beta1, and beta2. The subsequent statements define arrays that are used in the construction of Jeffreys' prior. These include m (the M matrix), x (the design matrix), xt (the transpose of x), and some additional work spaces.

The explanatory variables lvol and lrate are saved in the array x in the BEGINCNST and ENDCNST statements. See "BEGINCNST/ENDCNST Statement" on page 6248 for details. After all the variables are read into x, you transpose the x matrix and store it to xt. The ZEROMATRIX function call assigns all elements in matrix m the value zero. To avoid redundant calculation, it is best to perform these calculations as the last observation of the data set is processed—that is, when ind is 39.

You calculate Jeffreys' prior in the BEGINNODATA and ENDNODATA statements. The probability vector p is the product of the design matrix x and parameter vector beta. The diagonal elements in the matrix m are $p_i(1 - p_i)$. The expression lp is the logarithm of Jeffreys' prior. The PRIOR statement assigns lp as the prior for the $\boldsymbol{\beta}$ regression coefficients. The MODEL statement assigns a binary likelihood to resp, with probability p[ind]. The p array is calculated earlier using the matrix function MULT. You use the ind variable to pick out

the right probability value for each resp.

Posterior summary statistics are displayed in Output 80.4.1.

Output 80.4.1 PROC MCMC Results, Jeffreys' prior

Logistic Regression Model with Jeffreys Prior

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta0	10000	-2.9587	1.3258	-5.5936	-0.6027
beta1	10000	5.2905	1.8193	1.8590	8.7222
beta2	10000	4.6889	1.8189	1.3611	8.2490

You can also use PROC GENMOD to fit the same model by using the following statements:

```
proc genmod data=vaso descending;
   ods select PostSummaries PostIntervals;
   model resp = lvol lrate / d=bin link=logit;
   bayes seed=17 coeffprior=jeffreys nmc=20000 thin=2;
run;
```

The MODEL statement indicates that resp is the response variable and lvol and lrate are the covariates. The options in the MODEL statement specify a binary likelihood and a logit link function. The BAYES statement requests Bayesian capability. The SEED=, NMC=, and THIN= arguments work in the same way as in PROC MCMC. The COEFFPRIOR=JEFFREYS option requests Jeffreys' prior in this analysis.

The PROC GENMOD statements produce Output 80.4.2, with estimates very similar to those reported in Output 80.4.1. Note that you should not expect to see identical output from PROC GENMOD and PROC MCMC, even with the simulation setup and identical random number seed. The two procedures use different sampling algorithms. PROC GENMOD uses the adaptive rejection metropolis algorithm (ARMS) (Gilks and Wild 1992; Gilks 2003) while PROC MCMC uses a random walk Metropolis algorithm. The asymptotic answers, which means that you let both procedures run an very long time, would be the same as they both generate samples from the same posterior distribution.

Output 80.4.2 PROC GENMOD Results

Logistic Regression Model with Jeffreys Prior

The GENMOD Procedure

Bayesian Analysis

Posterior Summaries						
				Percentiles		
Parameter	N	Mean	Standard Deviation	25%	50%	75%
Intercept	10000	-2.8773	1.3213	-3.6821	-2.7326	-1.9097
lvol	10000	5.2059	1.8707	3.8535	4.9574	6.3337
lrate	10000	4.5525	1.8140	3.2281	4.3722	5.6643

Output 80.4.2 *continued*

Posterior Intervals					
Parameter	Alpha	Equal-Tail Interval		HPD Interval	
Intercept	0.050	-5.7447	-0.6877	-5.4593	-0.5488
lvol	0.050	2.2066	9.4415	2.0729	9.2343
lrate	0.050	1.5906	8.5272	1.3351	8.1152

Example 80.5: Poisson Regression

(View the complete code for this example (mcmcex5.sas) in the example repository.)

You can use the Poisson distribution to model the distribution of cell counts in a multiway contingency table. Aitkin et al. (1989) have used this method to model insurance claims data. Suppose the following hypothetical insurance claims data are classified by two factors: age group (with two levels) and car type (with three levels). The following statements create the data set:

```
title 'Poisson Regression';
data insure;
   input n c car $ age;
   ln = log(n);
   datalines;
   500   42  small  0
   1200  37  medium 0
   100    1  large  0
   400  101  small  1
   500   73  medium 1
   300   14  large  1
;

proc transreg data=insure design;
   model class(car / zero=last);
   id n c age ln;
   output out=input_insure(drop=_: Int:);
   run;
```

The variable n represents the number of insurance policy holders and the variable c represents the number of insurance claims. The variable car is the type of car involved (classified into three groups), and it is coded into two levels. The variable age is the age group of a policy holder (classified into two groups).

Assume that the number of claims c has a Poisson probability distribution and that its mean, μ_i, is related to the factors car and age for observation i by

$$\begin{aligned}
\log(\mu_i) &= \log(\mathrm{n}_i) + \mathbf{x}'\boldsymbol{\beta} \\
&= \log(\mathrm{n}_i) + \beta_0 + \\
&\quad \mathrm{car}_i(1)\beta_1 + \mathrm{car}_i(2)\beta_2 + \mathrm{car}_i(3)\beta_3 + \\
&\quad \mathrm{age}_i(1)\beta_4 + \mathrm{age}_i(2)\beta_5
\end{aligned}$$

The indicator variables $\text{car}_i(j)$ is associated with the jth level of the variable car for observation i in the following way:

$$\text{car}_i(j) = \begin{cases} 1 & \text{if car} = j \\ 0 & \text{if car} \neq j \end{cases}$$

A similar coding applies to age. The $\boldsymbol{\beta}$'s are parameters. The logarithm of the variable n is used as an offset—that is, a regression variable with a constant coefficient of 1 for each observation. Having the offset constant in the model is equivalent to fitting an expanded data set with 3000 observations, each with response variable y observed on an individual level. The log link relates the mean and the factors car and age.

The following statements run PROC MCMC:

```
proc mcmc data=input_insure outpost=insureout nmc=5000 propcov=quanew
          maxtune=0 seed=7;
   ods select PostSumInt;
   array data[4] 1 &_trgind age;
   array beta[4] alpha beta_car1 beta_car2 beta_age;
   parms alpha beta:;
   prior alpha beta: ~ normal(0, prec = 1e-6);
   call mult(data, beta, mu);
   model c ~ poisson(exp(mu+ln));
run;
```

The analysis uses a relatively flat prior on all the regression coefficients, with mean at 0 and precision at 10^{-6}. The option MAXTUNE=0 skips the tuning phase because the optimization routine (PROPCOV=QUANEW) provides good initial values and proposal covariance matrix.

There are four parameters in the model: alpha is the intercept; beta_car1 and beta_car2 are coefficients for the CLASS variable car, which has three levels; and beta_age is the coefficient for age. The symbol mu connects the regression model and the Poisson mean by using the log link. The MODEL statement specifies a Poisson likelihood for the response variable c.

Posterior summary and interval statistics are shown in Output 80.5.1.

Output 80.5.1 MCMC Results

Poisson Regression

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
alpha	5000	-2.6403	0.1344	-2.9133	-2.3831
beta_car1	5000	-1.8335	0.2917	-2.4692	-1.3336
beta_car2	5000	-0.6931	0.1255	-0.9485	-0.4589
beta_age	5000	1.3151	0.1386	1.0387	1.5812

To fit the same model by using PROC GENMOD, you can do the following. Note that the default normal prior on the coefficients $\boldsymbol{\beta}$ is $N(0, \text{prec} = 1e-6)$, the same as used in the PROC MCMC. The following statements run PROC GENMOD and create Output 80.5.2:

```
proc genmod data=insure;
   ods select PostSummaries PostIntervals;
   class car age(descending);
   model c = car age / dist=poisson link=log offset=ln;
   bayes seed=17 nmc=5000 coeffprior=normal;
run;
```

To compare, posterior summary and interval statistics from PROC GENMOD are reported in Output 80.5.2, and they are very similar to PROC MCMC results in Output 80.5.1.

Output 80.5.2 PROC GENMOD Results

Poisson Regression

The GENMOD Procedure

Bayesian Analysis

Posterior Summaries

Parameter	N	Mean	Standard Deviation	Percentiles 25%	50%	75%
Intercept	5000	-2.6424	0.1336	-2.7334	-2.6391	-2.5547
carlarge	5000	-1.8040	0.2764	-1.9859	-1.7929	-1.6101
carmedium	5000	-0.6908	0.1311	-0.7797	-0.6898	-0.6044
age1	5000	1.3207	0.1384	1.2264	1.3209	1.4140

Posterior Intervals

Parameter	Alpha	Equal-Tail Interval		HPD Interval	
Intercept	0.050	-2.9154	-2.3893	-2.8997	-2.3850
carlarge	0.050	-2.3668	-1.2891	-2.2992	-1.2378
carmedium	0.050	-0.9437	-0.4231	-0.9434	-0.4230
age1	0.050	1.0455	1.5871	1.0266	1.5629

Note that the descending option in the CLASS statement reverses the sort order of the CLASS variable age so that the results agree with PROC MCMC. If this option is not used, the estimate for age has a reversed sign as compared to Output 80.5.2.

Example 80.6: Nonlinear Poisson Regression Models

(View the complete code for this example (mcmcex6.sas) in the example repository.)

This example illustrates how to fit a nonlinear Poisson regression with PROC MCMC. In addition, it shows how you can improve the mixing of the Markov chain by selecting a different proposal distribution or by sampling on the transformed scale of a parameter. This example shows how to analyze count data for calls to a technical support help line in the weeks immediately following a product release. This information could be used to decide upon the allocation of technical support resources for new products. You can model the number of daily calls as a Poisson random variable, with the average number of calls modeled as a nonlinear function of the number of weeks that have elapsed since the product's release. The data are input into a SAS data set as follows:

```
title 'Nonlinear Poisson Regression';
data calls;
   input weeks calls @@;
   datalines;
1   0   1   2   2   2   2   1   3   1   3   3
4   5   4   8   5   5   5   9   6  17   6   9
7  24   7  16   8  23   8  27
;
```

During the first several weeks after a new product is released, the number of questions that technical support receives concerning the product increases in a sigmoidal fashion. The expression for the mean value in the classic Poisson regression involves the log link. There is some theoretical justification for this link, but with MCMC methodologies, you are not constrained to exploring only models that are computationally convenient. The number of calls to technical support tapers off after the initial release, so in this example you can use a logistic-type function to model the mean number of calls received weekly for the time period immediately following the initial release. The mean function $\lambda(t)$ is modeled as follows:

$$\lambda_i = \frac{\gamma}{1 + \exp\left[-(\alpha + \beta t_i)\right]}$$

The likelihood for every observation calls_i is

$$\text{calls}_i \sim \text{Poisson}\ (\lambda_i)$$

Past experience with technical support data for similar products suggests the following prior distributions:

$$\begin{aligned} \gamma &\sim \text{gamma}(\text{shape} = 3.5, \text{scale} = 12) \\ \alpha &\sim \text{normal}(-5, \text{sd} = 0.5) \\ \beta &\sim \text{normal}(0.75, \text{sd} = 0.5) \end{aligned}$$

The following PROC MCMC statements fit this model:

```
ods graphics on;
proc mcmc data=calls outpost=callout seed=53197 ntu=1000 nmc=20000
       propcov=quanew stats=none diag=ess;
   ods select TADpanel ess;
   parms alpha -4 beta 1 gamma 2;
```

```
    prior gamma ~ gamma(3.5, scale=12);
    prior alpha ~ normal(-5, sd=0.25);
    prior beta  ~ normal(0.75, sd=0.5);
    lambda = gamma*logistic(alpha+beta*weeks);
    model calls ~ poisson(lambda);
run;
```

The one PARMS statement defines a block of all parameters and sets their initial values individually. The PRIOR statements specify the informative prior distributions for the three parameters. The assignment statement defines λ, the mean number of calls. Instead of using the SAS function LOGISTIC, you can use the following statement to calculate λ and get the same result:

```
lambda = gamma / (1 + exp(-(alpha+beta*weeks)));
```

Mixing is not particularly good with this run of PROC MCMC. The ODS SELECT statement displays the diagnostic graphs and effective sample sizes (ESS) calculation while excluding all other output. The graphical output is shown in Output 80.6.1, and the ESS of each parameters are all relatively low (Output 80.6.2).

Output 80.6.1 Plots for Parameters

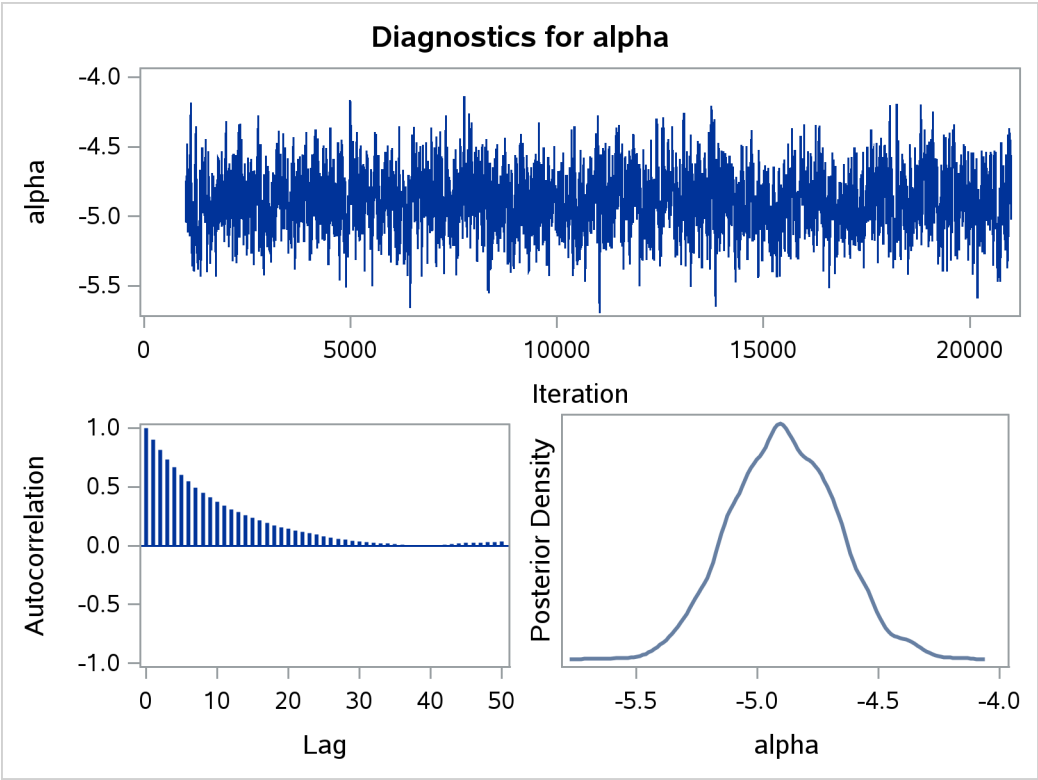

Output 80.6.1 *continued*

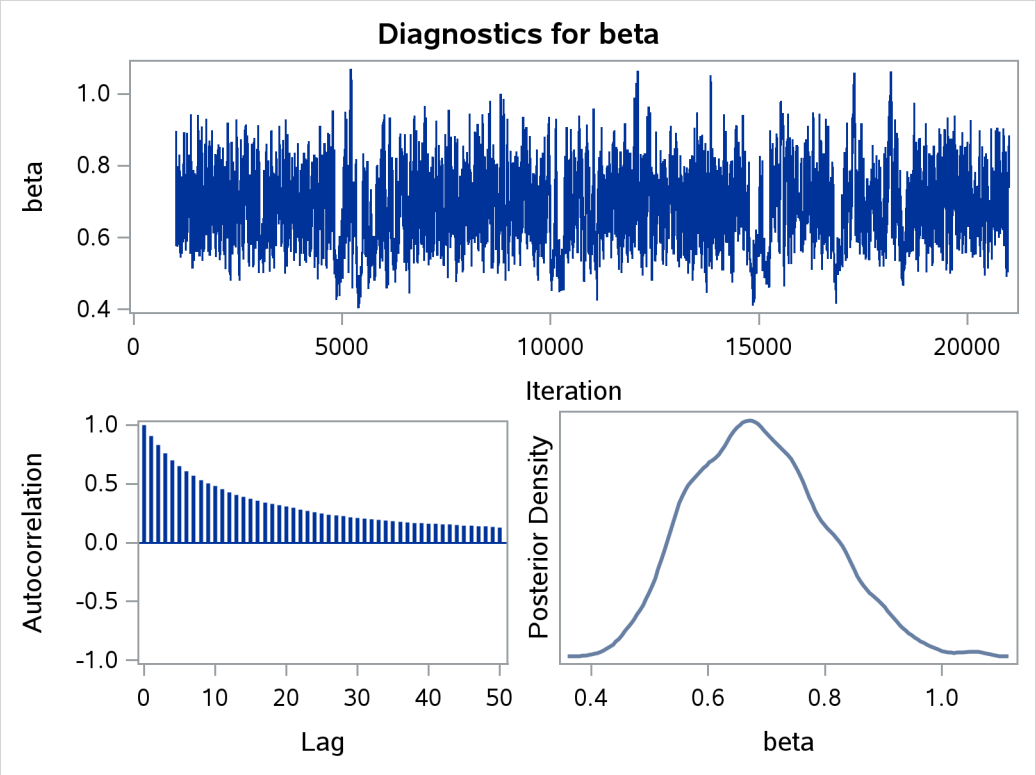

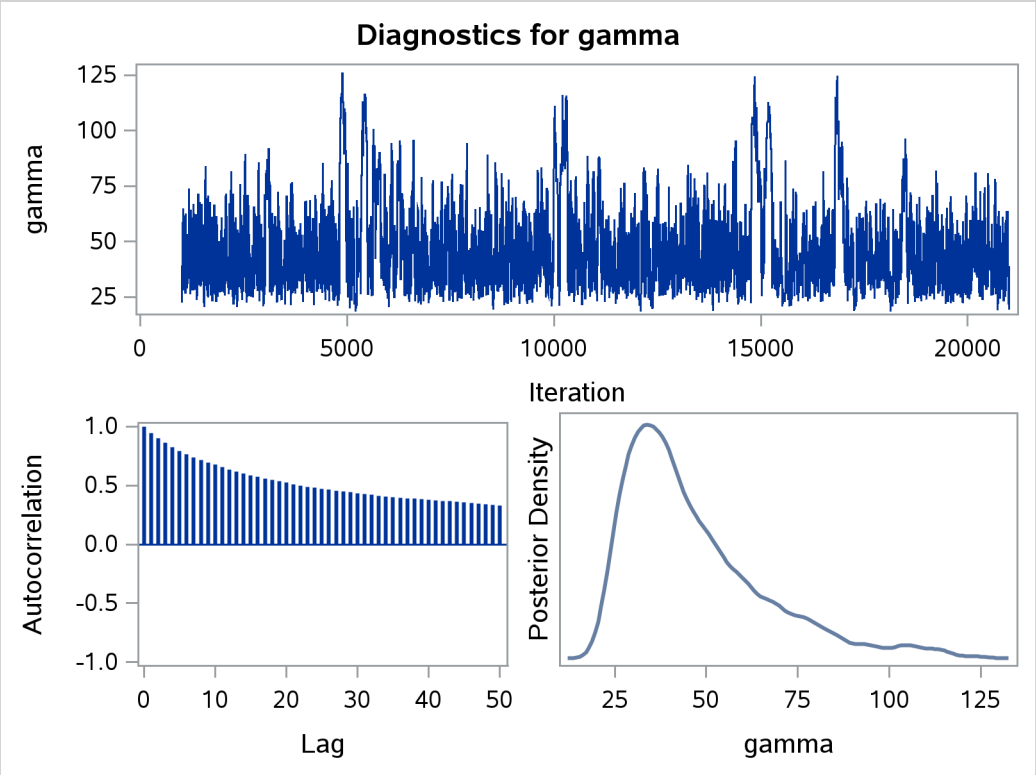

Output 80.6.2 Effective Sample Sizes

Nonlinear Poisson Regression

The MCMC Procedure

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
alpha	897.4	22.2870	0.0449
beta	231.6	86.3553	0.0116
gamma	162.9	122.8	0.0081

Often a simple scatter plot of the posterior samples can reveal a potential cause of the bad mixing. You can use PROC SGSCATTER to generate pairwise scatter plots of the three model parameters. The following statements generate Output 80.6.3:

```
proc sgscatter data=callout;
   matrix alpha beta gamma;
run;
```

Output 80.6.3 Pairwise Scatter Plots of the Parameters

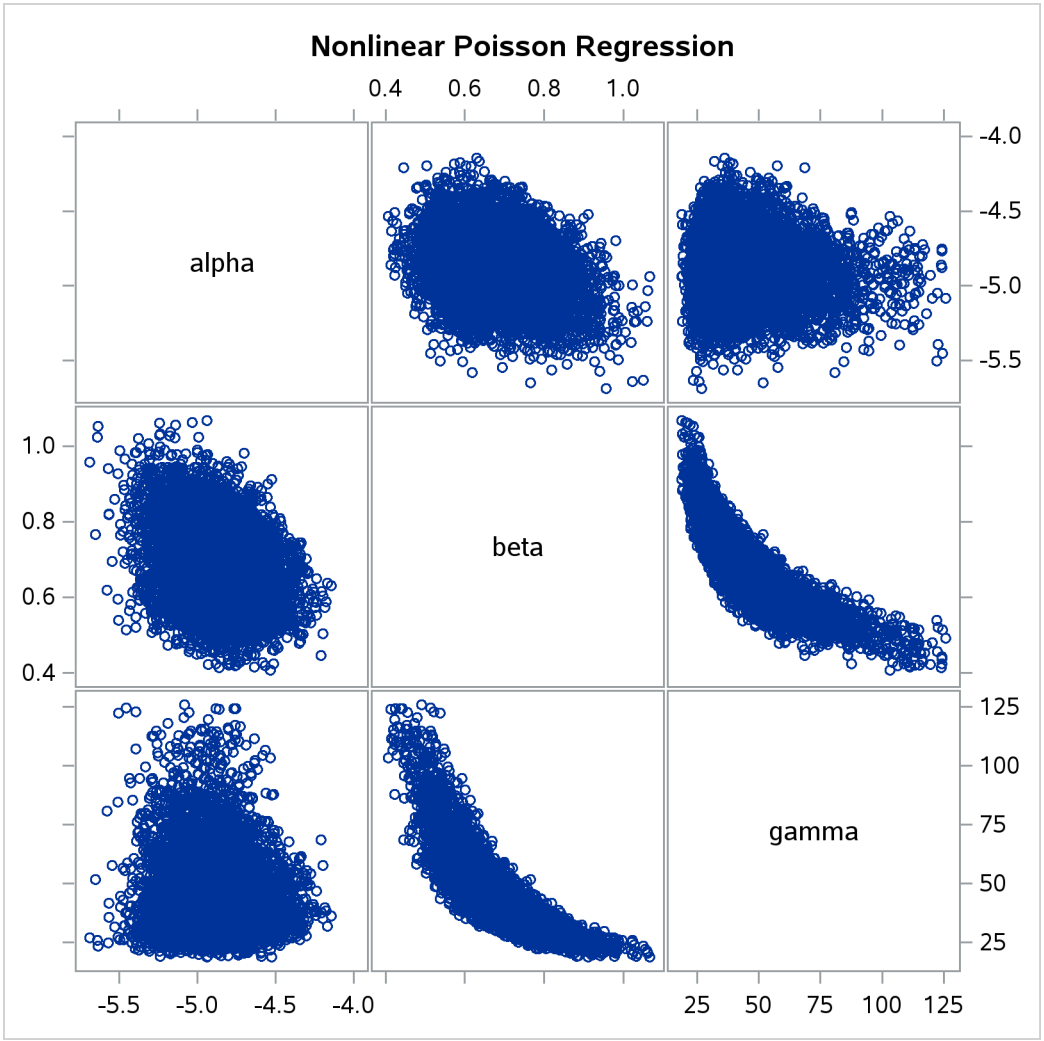

The nonlinearity in parameters beta and gamma stands out immediately. This explains why a random walk Metropolis with normal proposal has a difficult time exploring the joint distribution efficiently—the algorithm works best when the target distribution is unimodal and symmetric (normal-like). When there is nonlinearity in the parameters, it is impossible to find a single proposal scale parameter that optimally adapts to different regions of the joint parameter space. As a result, the Markov chain can be inefficient in traversing some parts of the distribution. This is evident in examining the trace plot of the gamma parameter. You see that the Markov chain sometimes gets stuck in the far-right tail and does not travel back to the high-density area quickly. This effect can be seen around the simulations 8,000 and 18,000 in Output 80.6.1.

Reparameterization can often improve the mixing of the Markov chain. Note that the parameter gamma has a positive support and that the posterior distribution is right-skewed. This suggests that the chain might mix more rapidly if you sample on the logarithm of the parameter gamma.

Let $\delta = \log(\gamma)$, and reparameterize the mean function as follows:

$$\lambda_i = \frac{\exp(\delta)}{1 + \exp\left[-(\alpha + \beta t_i)\right]}$$

To obtain the same inference, you use an induced prior on delta based on the gamma prior on the gamma parameter. This involves a transformation of variables, and you can obtain the following equivalency, where $|\exp(\delta)|$ is the Jacobian:

$$\begin{aligned} & \pi(\gamma) = \text{gamma}(\gamma; a, \text{scale} = b) = \frac{1}{b^a \Gamma(a)} \gamma^{a-1} \exp\left(-\gamma/b\right) \\ \Leftrightarrow \quad & \pi(\delta) = \text{gamma}(\gamma = \exp(\delta); a, \text{scale} = b) \cdot |\exp(\delta)| \end{aligned}$$

The distribution on δ simplifies to the following:

$$\pi(\delta) = \frac{1}{b^a \Gamma(a)} \exp\left(a\delta\right) \exp\left(-\exp\left(\delta\right)/b\right)$$

PROC MCMC supports such a distribution on the logarithm transformation of a gamma random variable. It is called the ExpGamma distribution.

In the original model, you specify a prior on gamma:

```
prior gamma ~ gamma(3.5, scale=12);
```

You can obtain the same inference by specifying an ExpGamma prior on delta and take an exponential transformation to get back to gamma:

```
prior delta ~ egamma(3.5, scale=12);
gamma = exp(delta);
```

The following statements produce Output 80.6.6 and Output 80.6.4:

```
proc mcmc data=calls outpost=tcallout seed=53197 ntu=1000 nmc=20000
       propcov=quanew diag=ess plots=(trace) monitor=(alpha beta gamma);
   ods select PostSumInt ESS TRACEpanel;
   parms alpha -4 beta 1 delta 2;
   prior alpha ~ normal(-5, sd=0.25);
   prior beta  ~ normal(0.75, sd=0.5);
   prior delta ~ egamma(3.5, scale=12);
   gamma = exp(delta);
   lambda = gamma*logistic(alpha+beta*weeks);
   model calls ~ poisson(lambda);
run;
```

The PARMS statement declares delta, instead of gamma, as a model parameter. The prior distribution of delta is **egamma**, as opposed to the **gamma** distribution. The GAMMA assignment statement transforms delta to gamma. The LAMBDA assignment statement calculates the mean for the Poisson by using the gamma parameter. The MODEL statement specifies a Poisson likelihood for the calls response.

The trace plots in Output 80.6.4 show better mixing of the parameters, and the effective sample sizes in Output 80.6.5 show substantial improvements over the original formulation of the model. The improvements are especially obvious in beta and gamma, where the increase is fivefold to tenfold.

Output 80.6.4 Plots for Parameters, Sampling on the Log Scale of Gamma

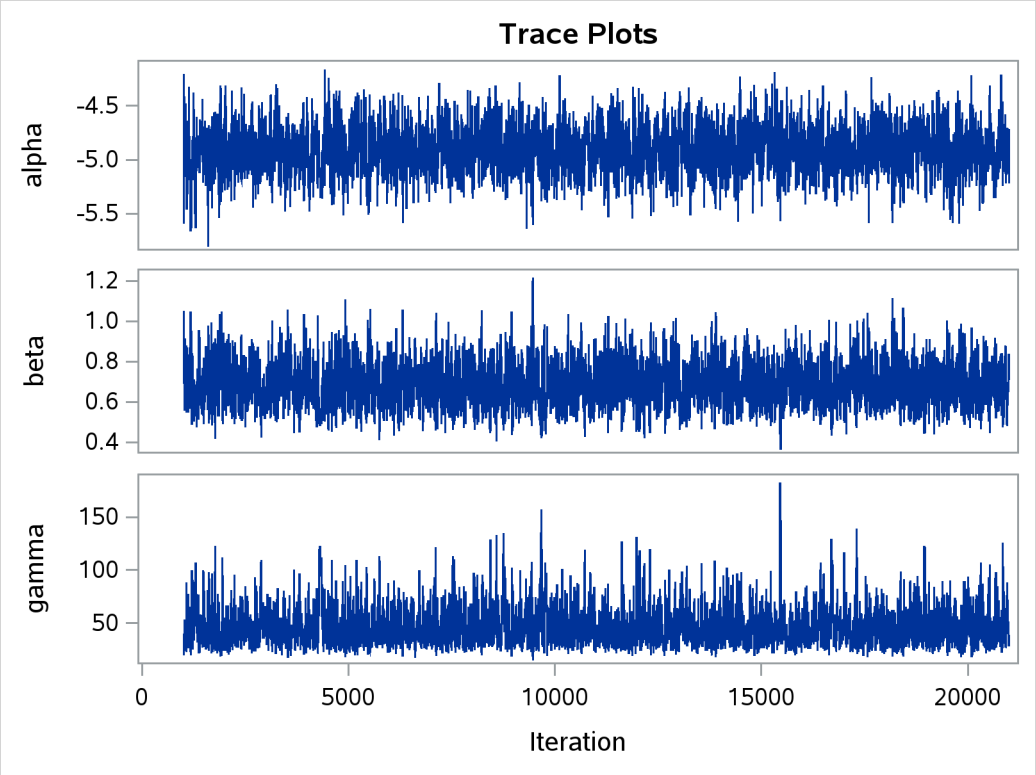

Output 80.6.5 Effective Sample Sizes, Sampling on the Log Scale of Gamma

Nonlinear Poisson Regression

The MCMC Procedure

Effective Sample Sizes

Parameter	ESS	Autocorrelation Time	Efficiency
alpha	1457.3	13.7242	0.0729
beta	1071.9	18.6589	0.0536
gamma	951.8	21.0123	0.0476

Output 80.6.6 shows the posterior summary and interval statistics of the nonlinear Poisson regression.

Output 80.6.6 MCMC Results, Sampling on the Log Scale of Gamma

Nonlinear Poisson Regression

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
alpha	20000	-4.9054	0.2268	-5.3551	-4.4658
beta	20000	0.6904	0.1160	0.4732	0.9059
gamma	20000	46.7576	19.7467	20.5841	87.6046

Note that the delta parameter has a more symmetric density than the skewed gamma parameter. A pairwise scatter plot (Output 80.6.7) shows a more linear relationship between beta and delta. The Metropolis algorithm always works better if the target distribution is approximately normal.

```
proc sgscatter data=tcallout;
   matrix alpha beta delta;
run;
```

Output 80.6.7 Pairwise Scatter Plots of the Transformed Parameters

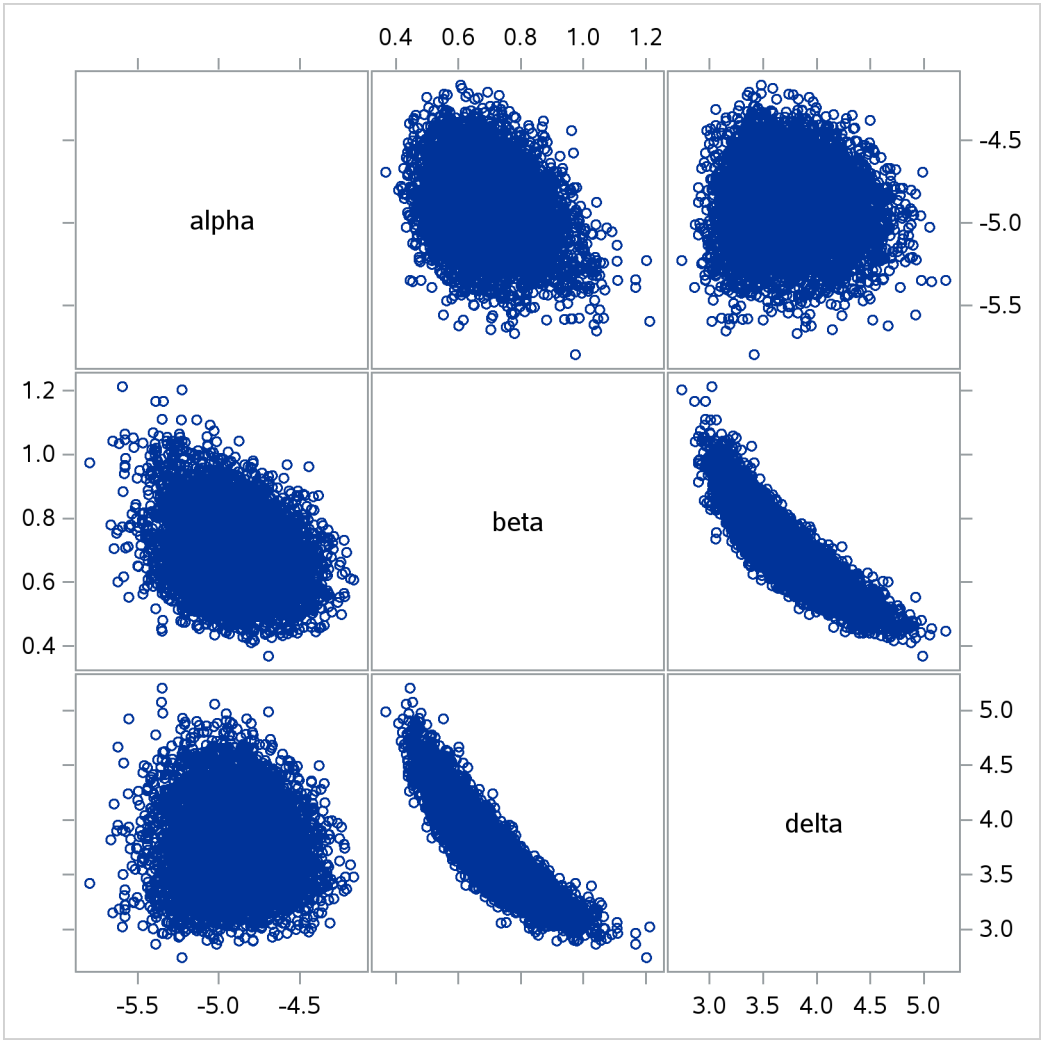

If you are still unsatisfied with the slight nonlinearity in the parameters beta and delta, you can try another transformation on beta. Normally you would not want to do a logarithm transformation on a parameter that has support on the real axis, because you would risk taking the logarithm of negative values. However, because all the beta samples are positive and the marginal posterior distribution is away from 0, you can try a such a transformation.

Let $\kappa = \log(\beta)$. The prior distribution on κ is the following:

$$\pi(\kappa) = \text{normal}(\beta = \exp(\kappa); \mu, \sigma^2) \cdot |\exp(\kappa)|$$

You can specify the prior distribution in PROC MCMC by using a GENERAL function:

```
parms kappa;
lprior = logpdf("normal", exp(kappa), 0.75, 0.5) + kappa;
prior kappa ~ general(lp);
beta = exp(kappa);
```

The PARMS statement declares the transformed parameter kappa, which will be sampled. The LPRIOR assignment statement defines the logarithm of the prior distribution on kappa. The LOGPDF function is used here to simplify the specification of the distribution. The PRIOR statement specifies the nonstandard distribution as the prior on kappa. Finally, the BETA assignment statement transforms kappa back to the beta parameter.

Applying logarithm transformations on both beta and gamma yields the best mixing. (The results are not shown here, but you can find the code in the file *mcmcex6.sas* in the SAS Sample Library.) The transformed parameters kappa and delta have much clearer linear correlation. However, the improvement over the case where gamma alone is transformed is only marginally significant (50% increase in ESS).

This example illustrates that PROC MCMC can fit Bayesian nonlinear models just as easily as Bayesian linear models. More importantly, transformations can sometimes improve the efficiency of the Markov chain, and that is something to always keep in mind. Also see "Example 80.20: Using a Transformation to Improve Mixing" on page 6487 for another example of how transformations can improve mixing of the Markov chains.

Example 80.7: Logistic Regression Random-Effects Model

(View the complete code for this example (mcmcex7.sas) in the example repository.)

This example illustrates how you can use PROC MCMC to fit random-effects models. In the example "Random-Effects Model" on page 6224 in "Getting Started: MCMC Procedure" on page 6214, you already saw PROC MCMC fit a linear random-effects model. This example shows how to fit a logistic random-effects model in PROC MCMC. Although you can use PROC MCMC to analyze random-effects models, you might want to first consider some other SAS procedures. For example, you can use PROC MIXED (see Chapter 84, "The MIXED Procedure") to analyze linear mixed effects models, PROC NLMIXED (see Chapter 89, "The NLMIXED Procedure") for nonlinear mixed effects models, and PROC GLIMMIX (see Chapter 52, "The GLIMMIX Procedure") for generalized linear mixed effects models. In addition, a sampling-based Bayesian analysis is available in the MIXED procedure through the PRIOR statement (see "PRIOR Statement" on page 6819 in Chapter 84, "The MIXED Procedure").

The data are taken from Crowder (1978). The Seeds data set is a 2×2 factorial layout, with two types of seeds, *O. aegyptiaca 75* and *O. aegyptiaca 73*, and two root extracts, *bean* and *cucumber*. You observe r, which is the number of germinated seeds, and n, which is the total number of seeds. The independent variables are seed and extract.

The following statements create the data set:

```
title 'Logistic Regression Random-Effects Model';
data seeds;
   input r n seed extract @@;
   ind = _N_;
   datalines;
10  39  0  0    23  62  0  0    23  81  0  0    26  51  0  0
17  39  0  0     5   6  0  1    53  74  0  1    55  72  0  1
32  51  0  1    46  79  0  1    10  13  0  1     8  16  1  0
10  30  1  0     8  28  1  0    23  45  1  0     0   4  1  0
 3  12  1  1    22  41  1  1    15  30  1  1    32  51  1  1
 3   7  1  1
;
```

You can model each observation r_i as having its own probability of success p_i, and the likelihood is as follows:

$$\mathrm{r}_i \sim \mathrm{binomial}(\mathrm{n}_i, p_i)$$

You can use the logit link function to link the covariates of each observation, seed and extract, to the probability of success,

$$\begin{aligned} \mu_i &= \beta_0 + \beta_1 \cdot \mathrm{seed}_i + \beta_2 \cdot \mathrm{extract}_i + \beta_3 \cdot \mathrm{seed}_i \cdot \mathrm{extract}_i \\ p_i &= \mathrm{logistic}(\mu_i + \delta_i) \end{aligned}$$

where δ_i is assumed to be an iid random effect with a normal prior:

$$\delta_i \sim \mathrm{normal}(0, \mathrm{var} = \sigma^2)$$

The four $\boldsymbol{\beta}$ regression coefficients and the standard deviation σ^2 in the random effects are model parameters; they are given noninformative priors as follows:

$$\begin{aligned} \pi(\beta_0, \beta_1, \beta_2, \beta_3) &\propto 1 \\ \sigma^2 &\sim \mathrm{igamma}(\mathrm{shape} = 0.01, \mathrm{scale} = 0.01) \end{aligned}$$

Another way of expressing the same model is as

$$p_i = \mathrm{logistic}(\delta_i)$$

where

$$\delta_i \sim \mathrm{normal}(\beta_0 + \beta_1 \cdot \mathrm{seed}_i + \beta_2 \cdot \mathrm{extract}_i + \beta_3 \cdot \mathrm{seed}_i \cdot \mathrm{extract}_i, \sigma^2)$$

The two models are equivalent. In the first model, the random effects δ_i centers at 0 in the normal distribution, and in the second model, δ_i centers at the regression mean. This hierarchical centering can sometimes improve mixing.

The following statements fit the second model and generate Output 80.7.1:

```
proc mcmc data=seeds outpost=postout seed=332786 nmc=20000;
   ods select PostSumInt;
   parms beta0 0 beta1 0 beta2 0 beta3 0 s2 1;
   prior s2 ~ igamma(0.01, s=0.01);
   prior beta: ~ general(0);
   w = beta0 + beta1*seed + beta2*extract + beta3*seed*extract;
   random delta ~ normal(w, var=s2) subject=ind;
   pi = logistic(delta);
   model r ~ binomial(n = n, p = pi);
run;
```

The PROC MCMC statement specifies the input and output data sets, sets a seed for the random number generator, and requests a large simulation size. The ODS SELECT statement displays the summary statistics and interval statistics tables. The PARMS statement declares the model parameters, and the PRIOR statements specify the prior distributions for $\boldsymbol{\beta}$ and σ^2.

The symbol w calculates the regression mean, and the RANDOM statement specifies the random effect, with a normal prior distribution, centered at w with variance σ^2. Note that the variable w is a function of the input data set variables. You can use data set variable in constructing the hyperparameters of the random-effects parameters, as long as the hyperparameters remain constant within each subject group. The SUBJECT= option indicates the group index for the random-effects parameters.

The symbol pi is the logit transformation. The MODEL specifies the response variable r as a binomial distribution with parameters n and pi.

Output 80.7.1 lists the posterior mean and interval estimates of the regression parameters.

Output 80.7.1 Logistic Regression Random-Effects Model

Logistic Regression Random-Effects Model

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta0	20000	-0.5570	0.1929	-0.9422	-0.1816
beta1	20000	0.0776	0.3276	-0.5690	0.7499
beta2	20000	1.3667	0.2923	0.8463	1.9724
beta3	20000	-0.8469	0.4718	-1.7741	0.0742
s2	20000	0.1171	0.0993	0.00163	0.3045

Example 80.8: Nonlinear Poisson Regression Multilevel Random-Effects Model

(View the complete code for this example (mcmcex8.sas) in the example repository.)

This example uses the pump failure data of Gaver and O'Muircheartaigh (1987) to illustrate how to fit a multilevel random-effects model with PROC MCMC. The number of failures and the time of operation are recorded for 10 pumps. Each of the pumps is classified into one of two groups that correspond to either continuous or intermittent operation. The following statements generate the data set:

```
title 'Nonlinear Poisson Regression Random-Effects Model';
data pump;
   input y t group @@;
   pump = _n_;
   logtstd = log(t) - 2.4564900;
   datalines;
 5  94.320 1   1  15.720 2    5  62.880 1
14 125.760 1   3   5.240 2   19  31.440 1
 1   1.048 2   1   1.048 2    4   2.096 2
22  10.480 2
;
```

Each row denotes data for a single pump, and the variable logtstd contains the centered operation times. Letting y_{ij} denote the number of failures for the jth pump in the ith group, Draper (1996) considers the

following hierarchical model for these data, where the data set variable logtstd is $\log t_{ij} - \overline{\log t}$:

$$\begin{aligned} y_{ij}|\lambda_{ij} &\sim \text{Poisson}(\lambda_{ij}) \\ \log \lambda_{ij} &= \alpha_i + \beta_i(\log t_{ij} - \overline{\log t}) + e_{ij} \end{aligned}$$

This model specifies different intercepts and slopes for each group ($i = 1, 2$), and the random effect e_{ij} is a mechanism for accounting for overdispersion. You can use noninformative priors on the parameters α_i, β_i, and σ^2, and a normal prior on e_{ij},

$$\begin{aligned} u_i &= \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix} \sim \text{mvn}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1e6 & 0 \\ 0 & 1e6 \end{pmatrix}\right) \quad \text{for } i = 1, 2 \\ \sigma^2 &\sim \text{igamma}\,(\text{shape} = 0.01, \text{scale} = 0.01) \\ e_{ij}|\sigma^2 &\sim \text{normal}(0, \sigma^2) \end{aligned}$$

where u_i is a multidimensional random effect. The following statements fit such a random-effects model:

```
ods graphics on;
proc mcmc data=pump outpost=postout seed=248601 nmc=10000
   plots=trace stats=none diag=none;
   ods select tracepanel;
   array u[2] alpha beta;
   array mu[2] (0 0);
   parms s2;
   prior s2 ~ igamma(0.01, scale=0.01);
   random u ~ MVNAR(mu, sd=1e6, rho=0) subject=group monitor=(u);
   random e ~ normal(0, var=s2) subject=pump monitor=(random(1));
   w = alpha + beta * logtstd;
   lambda = exp(w+e);
   model y ~ poisson(lambda);
run;
```

The PROC MCMC statement specifies the input data set (Pump), the output data set (Postout), a seed for the random number generator, and a simulation sample size of 10,000. The program requests that only trace plots be produced, disabling all posterior calculations and convergence diagnostics tests. The ODS SELECT statement displays the trace plots, which are the primary focus.

The first ARRAY statement declares an array u of size 2 and names the elements alpha and beta. The array u stores the random-effects parameters alpha and beta. The next ARRAY statement defines the mean of the multivariate normal prior on u.

The PARMS statement declares the only model parameter here, the variance s2 in the prior distribution for the random effect e_{ij}. The PRIOR statement specifies an inverse gamma prior on the variance. The first RANDOM statement specifies a multivariate normal prior on u. The MVNAR distribution is a multivariate normal distribution with a first-order autoregressive covariance. When the argument *rho* is set to 0, this distribution simplifies to a multivariate normal distribution with a shared variance. The RANDOM statement also indicates the group variable as its subject index and monitors all elements u. The second RANDOM statement specifies a normal prior on the effect e, where the subject index variable is pump. The MONITOR= option requests that PROC MCMC randomly choose one of the 10 e random-effects parameters to monitor.

Next, programming statements construct the mean of the Poisson likelihood, and the MODEL statement specifies the likelihood function for each observation.

Output 80.8.1 shows trace plots for $\sigma^2, \alpha_1, \alpha_2, \beta_1, \beta_2,$ and e_8. You can see that the chains are mixing poorly.

Output 80.8.1 Trace Plots of σ^2, α, $\boldsymbol{\beta}$, and e_8 without Hierarchical Centering

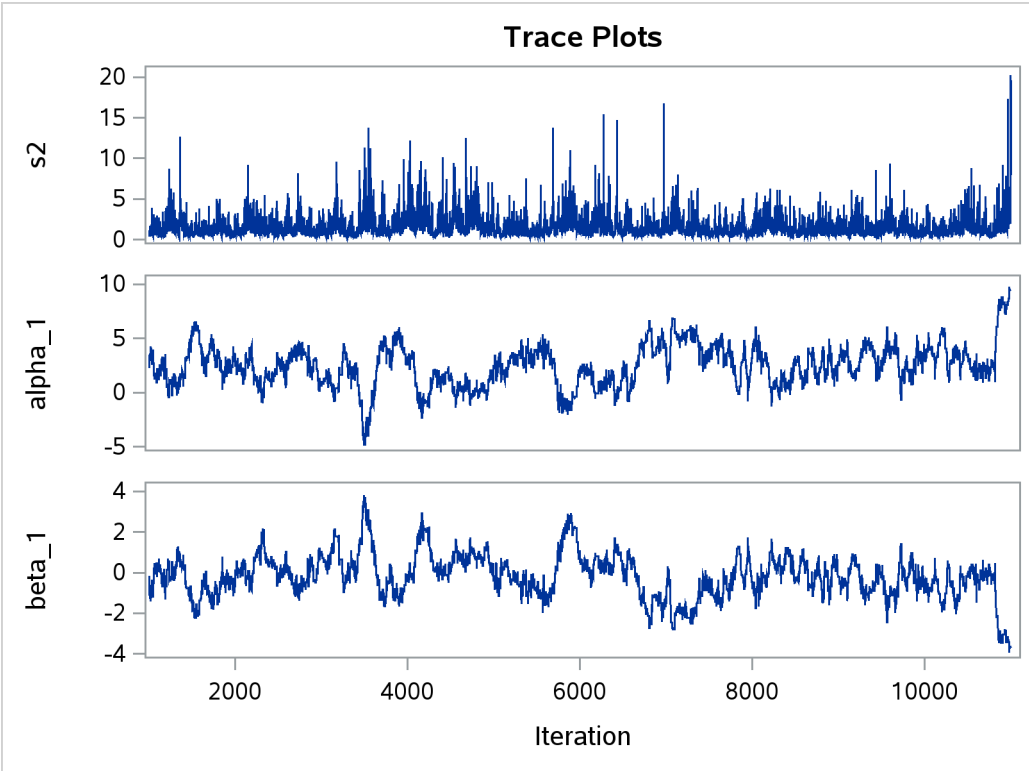

Output 80.8.1 *continued*

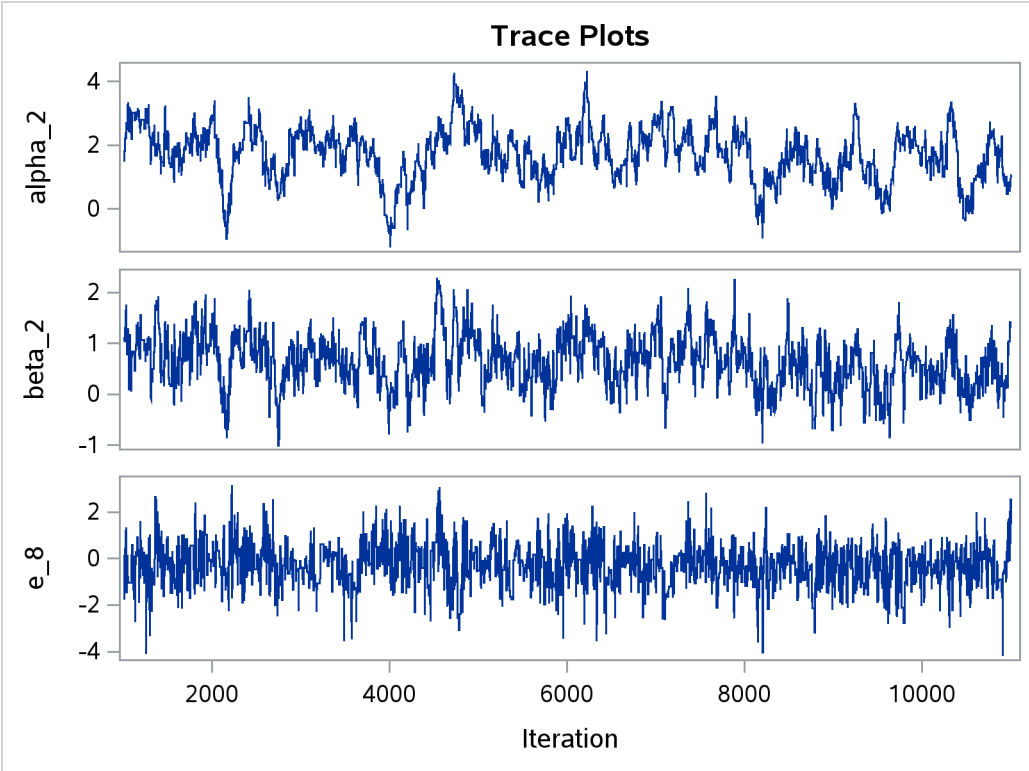

To improve mixing, you can repeat the same analysis by using a hierarchical centering technique, where instead of using a normal prior centered at 0 on e_{ij}, you center the random effects on the group means:

$$
\begin{aligned}
y_{ij}|\lambda_{ij} &\sim \text{Poisson}(\lambda_{ij}) \\
\log \lambda_{ij} &\sim \text{normal}\left(\alpha_i + \beta_i(\log t_{ij} - \overline{\log t}), \text{var} = \sigma^2\right)
\end{aligned}
$$

The following statements illustrate how to fit a multilevel hierarchical centering random-effects model:

```
proc mcmc data=pump outpost=postout_c seed=248601 nmc=10000
   plots=trace diag=none;
   ods select tracepanel postsumint;
   array u[2] alpha beta;
   array mu[2] (0 0);
   parms s2 1;
   prior s2 ~ igamma(0.01, scale=0.01);
   random u ~ MVNAR(mu, sd=1e6, rho=0) subject=group monitor=(u);
   w = alpha + beta * logtstd;
   random llambda ~ normal(w, var = s2) subject=pump monitor=(random(1));
   lambda = exp(llambda);
   model y ~ poisson(lambda);
run;
```

The difference between these statements and the previous statements on page 6418 is that these statements have the variable w as the prior mean of the random effect llambda. The symbol lambda is the exponential of

the corresponding $\log \lambda_{ij}$ (llambda), and the MODEL statement assigns the response variable y a Poisson likelihood with a mean parameter lambda, the same way it did in the previous statements.

The trace plots of the monitored parameters are shown in Output 80.8.2. The mixing is significantly improved over the previous model. The posterior summary and interval statistics tables are shown in Output 80.8.3.

Output 80.8.2 Trace Plots of σ^2, α, and $\boldsymbol{\beta}$ with Hierarchical Centering

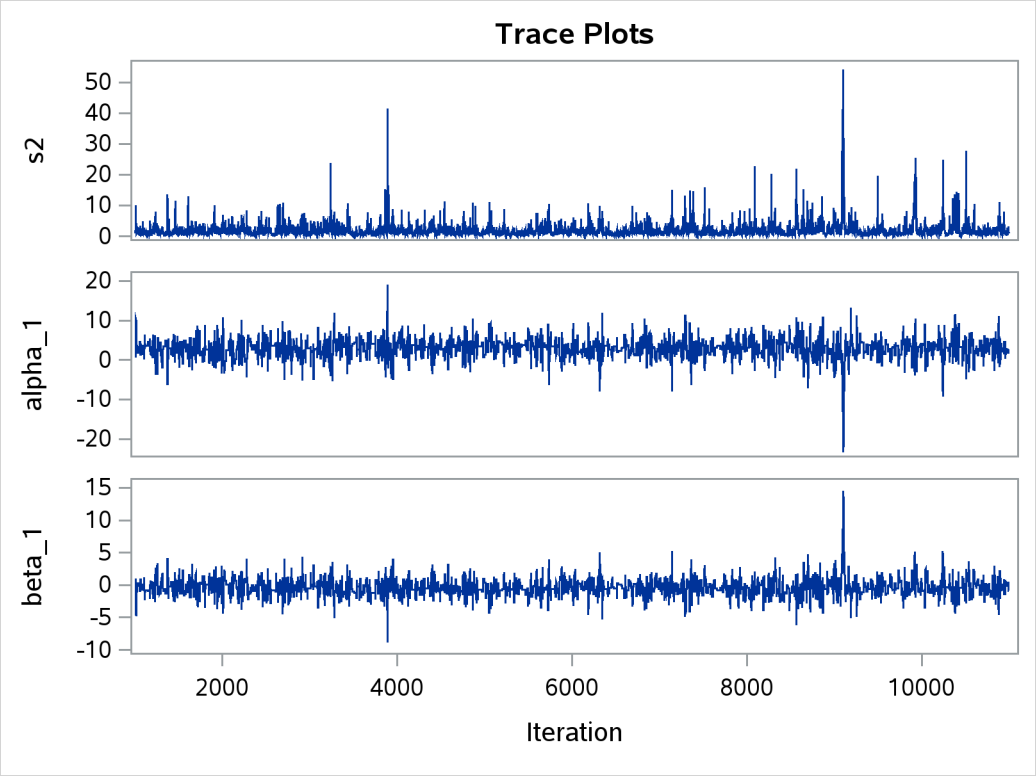

Output 80.8.2 *continued*

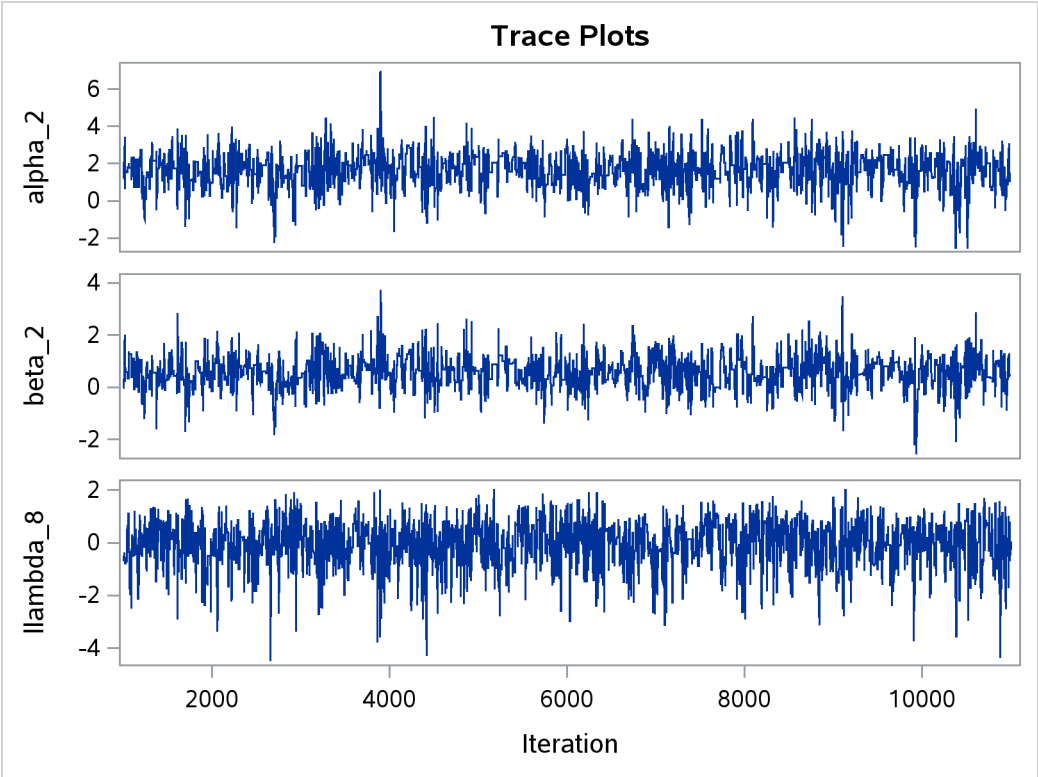

Output 80.8.3 Posterior Summary Statistics

Nonlinear Poisson Regression Random-Effects Model

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
s2	10000	1.7606	2.2022	0.1039	4.7631
alpha_1	10000	2.9286	2.4247	-1.9416	7.4115
beta_1	10000	-0.4018	1.3110	-2.9323	2.0623
alpha_2	10000	1.6105	0.8801	-0.0436	3.2985
beta_2	10000	0.5652	0.5804	-0.5469	1.7381
llambda_8	10000	-0.0560	0.8395	-1.6933	1.4612

You can generate a caterpillar plot (Output 80.8.4) of the random-effects parameters by calling the %CATER macro:

```
%CATER(data=postout_c, var=llambda_:);
ods graphics off;
```

Varying llambda indicates nonconstant dispersion in the Poisson model.

Output 80.8.4 Caterpillar Plots of the Random-Effects Parameters

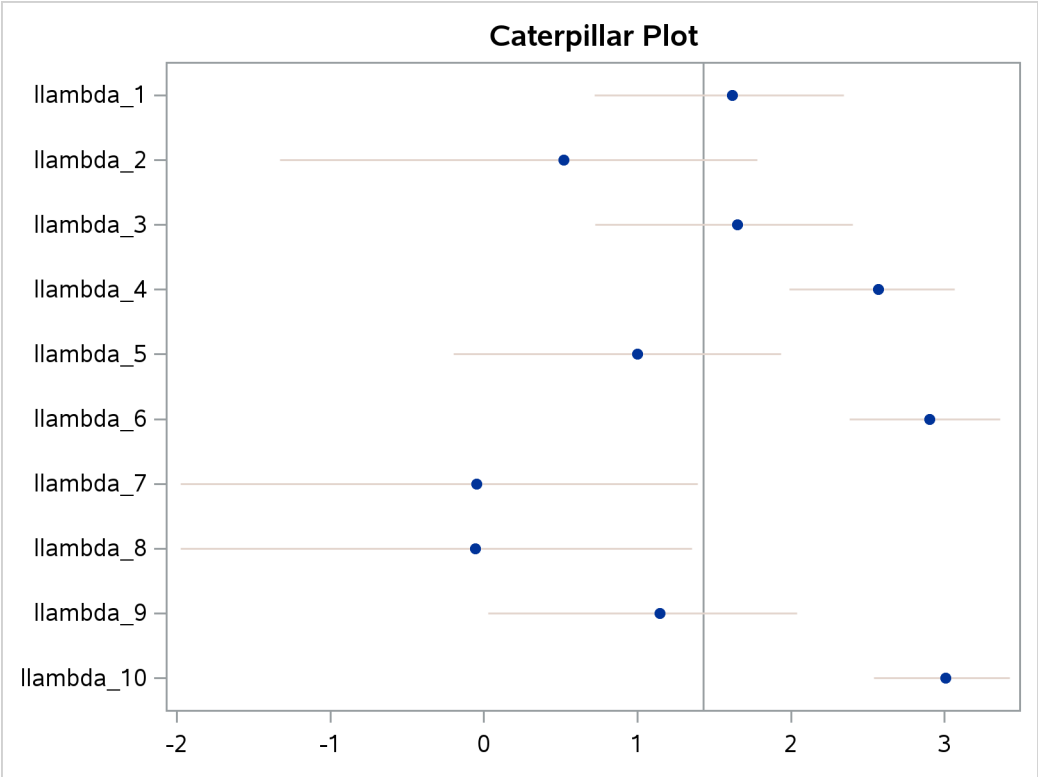

Example 80.9: Multivariate Normal Random-Effects Model

(View the complete code for this example (mcmcex9.sas) in the example repository.)

Gelfand et al. (1990) use a multivariate normal hierarchical model to estimate growth regression coefficients for the growth of 30 young rats in a control group over a period of 5 weeks. The following statements create a SAS data set with measurements of Weight, Age (in days), and Subject.

```
title 'Multivariate Normal Random-Effects Model';
data rats;
   array days[5] (8 15 22 29 36);
   input weight @@;
   subject = ceil(_n_/5);
   index = mod(_n_-1, 5) + 1;
   age = days[index];
   drop index days:;
   datalines;
151 199 246 283 320 145 199 249 293 354
147 214 263 312 328 155 200 237 272 297
135 188 230 280 323 159 210 252 298 331
141 189 231 275 305 159 201 248 297 338
177 236 285 350 376 134 182 220 260 296
160 208 261 313 352 143 188 220 273 314
154 200 244 289 325 171 221 270 326 358
```

```
163 216 242 281 312 160 207 248 288 324
142 187 234 280 316 156 203 243 283 317
157 212 259 307 336 152 203 246 286 321
154 205 253 298 334 139 190 225 267 302
146 191 229 272 302 157 211 250 285 323
132 185 237 286 331 160 207 257 303 345
169 216 261 295 333 157 205 248 289 316
137 180 219 258 291 153 200 244 286 324
;
```

The model assumes normal measurement errors,

$$\text{Weight}_{ij} \sim \text{normal}\left(\alpha_i + \beta_i \text{Age}_{ij}, \sigma^2\right), \quad i = 1, \ldots, 30; j = 1, \ldots, 5$$

where i indexes rat (Subject variable), j indexes the time period, Weight_{ij} and Age_{ij} denote the weight and age of the ith rat in week j, and σ^2 is the variance in the normal likelihood. The individual intercept and slope coefficients are modeled as the following:

$$\boldsymbol{\theta}_i = \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix} \sim \text{MVN}\left(\boldsymbol{\theta}_c = \begin{pmatrix} \alpha_c \\ \beta_c \end{pmatrix}, \boldsymbol{\Sigma}_c\right), \quad i = 1, \ldots, 30$$

You can use the following independent prior distributions on $\boldsymbol{\theta}_c$, $\boldsymbol{\Sigma}_c$, and σ^2:

$$\begin{aligned}
\boldsymbol{\theta}_c &\sim \text{MVN}\left(\boldsymbol{\mu}_{\mathbf{0}} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \boldsymbol{\Sigma}_{\mathbf{0}} = \begin{pmatrix} 1000 & 0 \\ 0 & 1000 \end{pmatrix}\right) \\
\boldsymbol{\Sigma}_c &\sim \text{iwishart}\left(\rho = 2, \mathbf{S} = \rho \cdot \begin{pmatrix} 0.01 & 0 \\ 0 & 10 \end{pmatrix}\right) \\
\sigma^2 &\sim \text{igamma}\,(\text{shape} = 0.01, \text{scale} = 0.01)
\end{aligned}$$

The following statements fit this multivariate normal random-effects model:

```
proc mcmc data=rats nmc=10000 outpost=postout
   seed=17 init=random;
   ods select Parameters REParameters PostSumInt;
   array theta[2] alpha beta;
   array theta_c[2];
   array Sig_c[2,2];
   array mu0[2] (0 0);
   array Sig0[2,2] (1000 0 0 1000);
   array S[2,2] (0.02 0 0 20);

   parms theta_c Sig_c {121 0 0 0.26} var_y;
   prior theta_c ~ mvn(mu0, Sig0);
   prior Sig_c ~ iwish(2, S);
   prior var_y ~ igamma(0.01, scale=0.01);

   random theta ~ mvn(theta_c, Sig_c) subject=subject
      monitor=(alpha_9 beta_9 alpha_25 beta_25);
   mu = alpha + beta * age;
   model weight ~ normal(mu, var=var_y);
run;
```

The ODS SELECT statement displays information about model parameters, random-effects parameters, and the posterior summary statistics. The ARRAY statements allocate memory space for the multivariate parameters and hyperparameters in the model. The parameters are $\boldsymbol{\theta}$ (theta where the variable name of each element is alpha or beta), $\boldsymbol{\theta}_c$ (theta_c), and $\boldsymbol{\Sigma}_c$ (Sig_c). The hyperparameters are $\boldsymbol{\mu_0}$ (mu0), $\boldsymbol{\Sigma_0}$ (Sig0), and $\mathbf{S}$ (S). The multivariate hyperparameters are assigned with constant values using parentheses ( ).

The PARMS statement declares model parameters and assigns initial values to Sig_c using braces { }. The PRIOR statements specify the prior distributions. The RANDOM statement defines an array random effect theta and specifies a multivariate normal prior distribution. The SUBJECT= option indicates cluster membership for each of the random-effects parameter. The MONITOR= option monitors the individual intercept and slope coefficients of subjects 9 and 25.

You can use the following syntax in the RANDOM statement to monitor all parameters in an array random effect:

```
monitor=(theta)
```

This would produce posterior summary statistics on $\alpha_1 \cdots \alpha_{30}$ and $\beta_1 \cdots \beta_{30}$.

The following syntax monitors all α_i parameters:

```
monitor=(alpha)
```

If you did not name elements of theta to be alpha and beta, the SAS System creates variable names automatically in a consecutive fashion, as in theta1 and theta2.

Output 80.9.1 Parameter and Random-Effects Parameter Information Table

Multivariate Normal Random-Effects Model

The MCMC Procedure

Parameters

Block	Parameter	Array Index	Sampling Method	Initial Value	Prior Distribution
1	theta_c1		Conjugate	-4.5834	MVNormal(mu0, Sig0)
	theta_c2			5.7930	
2	Sig_c1	[1,1]	Conjugate	121.0	iWishart(2, S)
	Sig_c2	[1,2]		0	
	Sig_c3	[2,1]		0	
	Sig_c4	[2,2]		0.2600	
3	var_y		Conjugate	2806714	igamma(0.01, scale=0.01)

Random Effect Parameters

Parameter	Sampling Method	Subject	Number of Subjects	Subject Values	Prior Distribution
theta	N-Metropolis	subject	30	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...	MVNormal(theta_c, Sig_c)

Output 80.9.1 displays the parameter and random-effects parameter information tables. The Array Index column in "Parameters" table shows the index reference of the elements in the array parameter Sig_c. The total number of subjects in the study is 30.

Output 80.9.2 Multivariate Normal Random-Effects Model

Multivariate Normal Random-Effects Model

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
theta_c1	10000	106.1	2.2486	101.7	110.6
theta_c2	10000	6.1975	0.1988	5.8058	6.5815
Sig_c1	10000	110.8	45.9169	37.9670	203.8
Sig_c2	10000	-1.4267	2.3320	-6.2878	2.7756
Sig_c3	10000	-1.4267	2.3320	-6.2878	2.7756
Sig_c4	10000	1.0591	0.2979	0.5549	1.6538
var_y	10000	37.6855	5.9591	27.0943	49.4449
alpha_9	10000	119.4	5.6756	108.1	130.5
beta_9	10000	7.4670	0.2382	7.0146	7.9278
alpha_25	10000	86.5673	6.3694	74.4247	99.9007
beta_25	10000	6.7804	0.2612	6.2529	7.2906

Output 80.9.2 displays posterior summary statistics for model parameters and the random-effects parameters for subjects 9 and 25. You can see that there is a substantial difference in the intercepts and growth rates between the two rats.

A seemingly confusing message might occur if a symbol name matches an internally generated variable name for elements of an array. For example, if, instead of using the symbol var_y in the SAS program for the model variance σ^2, you used s2, the SAS System produces the following error message:

```
ERROR: The initial value 0 for the parameter S2 is outside of the prior
       distribution support set.
```

This is confusing because the program does not assign an initial value for the parameter s2 in the PARMS statement, and you might expect that PROC MCMC would not generate an invalid initial value. The confusion is caused by the ARRAY statement that defines the array variable S:

```
array S[2,2] (0.02 0 0 20);
```

Elements of S are automatically given names s1–s4. PROC MCMC interprets s2 as an element in S that was given a value of 0, hence producing this error message.

Example 80.10: Missing at Random Analysis

(View the complete code for this example (mcmcex10.sas) in the example repository.)

This example illustrates how PROC MCMC treats missing at random (MAR) data. For a short overview of missing data problems, see the section "Handling of Missing Data" on page 6352.

Researchers studied the effects of air pollution on respiratory disease in children. The response variable (y) represented whether a child exhibited wheezing symptoms; it was recorded as 1 for symptoms exhibited and 0 for no symptoms exhibited. City of residency (x1) and maternal smoking status (x2) were the explanatory variables. The variable x1 was coded as 1 if the child lived in the more polluted city, Steel City, and 0 if the child lived in Green Hills. The variable x2 was the number of cigarettes the mother reported that she smoked per day. Both the covariates contain missing values: 17 for x1 and 30 for x2, respectively. The total number of observations in the data set is 390. The following statements generate the data set air:

```
title 'Missing at Random Analysis';
data air;
   input y x1 x2 @@;
   datalines;
 0  0  0   0  0  0   0  1  0   0  0  0   0  0 11   0  1  7
 0  0  8   0  1 10   0  1  9   0  0  0   1  1  6   0  1 10
 0  1 12   0  0  .   0  0  0   0  1  0   0  1  7   1  1 15
 0  0  8   0  0  0   1  1  0   1  0  6   0  0  0   1  1 11
 0  0  0   1  0  0   1  0  5   0  0  8   0  0  0   0  1  9

   ... more lines ...

 0  0 11   0  0  0   0  0  6   0  0 12   0  0 10   0  1 10
 0  1 11   0  0  9   1  0 11   0  1  7   0  0  7   0  0  0
 0  . 11   1  1  6   0  0  8   0  0  0   0  1 12   0  0  0
 0  1  0   1  1  8   0  0  0   0  1 11   0  1  0   0  1  8
 0  .  0   1  0  0   1  1 10   0  .  4   1  1 16   0  . 13
;
```

Suppose you want to fit a logistic regression model for whether the subject develops wheezing symptoms with density for the $i = 1, \ldots, 390$ subjects as follows:

$$\begin{aligned} \mathsf{y}_i &\sim \text{binary}(p_i) \\ \text{logit}(p_i) &= \beta_0 + \beta_1 \cdot \mathsf{x1}_i + \beta_2 \cdot \mathsf{x2}_i \\ \pi(\beta_0), \pi(\beta_1), \pi(\beta_2) &= \text{normal}(0, \sigma^2 = 10) \end{aligned}$$

Suppose you specify a joint distribution of x1 and x2 in terms of the product of a conditional and marginal distribution; that is,

$$p(\mathsf{x1}, \mathsf{x2}|\boldsymbol{\alpha}) = p(\mathsf{x1}|\mathsf{x2}, \alpha_{10}, \alpha_{11}) p(\mathsf{x2}|\alpha_{20})$$

where $p(\mathsf{x1}_i|\mathsf{x2}_i, \alpha_{10}, \alpha_{11})$ could be a logistic model and $p(\mathsf{x2}_i|\alpha_{20})$ could be a Poisson distribution that models the following counts:

$$\begin{aligned}
p(\mathsf{x1}_i|\mathsf{x2}_i, \alpha_{10}, \alpha_{11}) &= \text{binary}(p_{c,i}) \\
\text{logit}(p_{c,i}) &= \alpha_{10} + \alpha_{11} \cdot \mathsf{x2}_i \\
\pi(\alpha_{10}), \pi(\alpha_{11}) &= \text{normal}(0, \sigma^2 = 10) \\
p(\mathsf{x2}_i|\alpha_{20}) &= \text{Poisson}(e^{\alpha_{20}}) \\
\pi(\alpha_{20}) &= \text{normal}(0, \sigma^2 = 2)
\end{aligned}$$

The researchers are interested in interpreting how the odds of developing a wheeze changes for a child living in the more polluted city. The odds ratio can be written as the follows:

$$\text{OR}_{\mathsf{x1}} = \exp(\beta_1)$$

Similarly, the odds ratio for the maternal smoking effect can be written as follows:

$$\text{OR}_{\mathsf{x2}} = \exp(\beta_2)$$

The following statements fit a Bayesian logistic regression with missing covariates:

```
proc mcmc data=air seed=1181 nmc=10000 monitor=(_parms_ orx1 orx2)
   diag=none plots=none;
   parms beta0 -1 beta1 0.1 beta2 .01;
   parms alpha10 0 alpha11 0 alpha20 0;

   prior beta: alpha1: ~ normal(0,var=10);
   prior alpha20 ~ normal(0,var=2);

   beginnodata;
   pm = exp(alpha20);
   orx1 = exp(beta1);
   orx2 = exp(beta2);
   endnodata;
   model x2 ~ poisson(pm) monitor=(1 3 10);
   p1 = logistic(alpha10 + alpha11 * x2);
   model x1 ~ binary(p1) monitor=(random(3));
   p = logistic(beta0 + beta1*x1 + beta2*x2);
   model y ~ binary(p);
run;
```

The PARMS statements specify the parameters in the model and assign initial values to each of them. The PRIOR statements specify priors for all the model parameters. The notations beta: and alpha: in the PRIOR statements are shorthand for all variables that start with "beta," and "alpha," respectively. The shorthand notation is not necessary, but it keeps your code succinct.

The BEGINNODATA and ENDNODATA statements enclose three programming statements that calculate the Poisson mean (pm) and the two odds ratios (ORX1 and ORX2). These enclosed statements are independent of any data set variables, and they are run only once per iteration to reduce unnecessary observation-level computations.

The first MODEL statement assigns a Poisson likelihood with mean pm to x2. The statement models missing values in x2 automatically, creating one variable for each of the missing values, and augments them accordingly. By default, PROC MCMC does not output analyses of the posterior samples of the missing values. You can use the MONITOR= option to choose the missing values that you want to monitor. In the example, the first, third, and tenth missing values are monitored.

The P1 assignment statement calculates $p_{c.i}$. The second MODEL statement assigns a binary likelihood with probability p1 and requests a random choice of three missing data variables of x1 to monitor.

The P assignment statement calculates p_i in the logistic model. The third MODEL statement specifies the complete data likelihood function for Y.

Output 80.10.1 displays the number of observations read from the DATA= data set, the number of observations used in the analysis, and the "Missing Data Information" table. No observations were omitted from the data set in the analysis.

The "Missing Data Information" table lists the variables that contain missing values, which are x1 and x2, the number of missing observations in each variable, the observation indices of these missing values, and the sampling algorithms used. By default, the first 20 observation indices of each variable are printed in the table.

Output 80.10.1 Observation Information and Missing Data Information

Missing at Random Analysis

The MCMC Procedure

Number of Observations Read	390
Number of Observations Used	390

Missing Data Information Table

Variable	Number of Missing Obs	Observation Indices	Sampling Method
x2	30	14 41 50 55 59 66 71 83 88 90 118 158 174 175 178 183 196 203 210 212 ...	N-Metropolis
x1	17	50 92 93 167 194 231 273 296 303 304 308 330 349 373 385 388 390	Inverse CDF

There are 30 missing values in the variable x2, and 17 in x1. Internally, PROC MCMC creates 30 and 17 variables for the missing values in x2 and x1, respectively. The default naming convention for these missing values is to concatenate the response variable and the observation number. For example, the first missing value in x2 is the fourteenth observation, and the corresponding variable is x2_14.

Output 80.10.2 displays the summary and interval statistics for each parameter, the odds ratios, and the monitored missing values.

Output 80.10.2 Posterior Summary and Interval Statistics

Missing at Random Analysis

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta0	10000	-1.3732	0.2078	-1.7909	-0.9676
beta1	10000	0.4797	0.2387	0.0268	0.9491
beta2	10000	0.0156	0.0227	-0.0265	0.0642
alpha10	10000	-0.2166	0.1422	-0.4662	0.0874
alpha11	10000	0.0126	0.0201	-0.0267	0.0521
alpha20	10000	1.5635	0.0235	1.5199	1.6105
orx1	10000	1.6627	0.4094	0.9205	2.4493
orx2	10000	1.0160	0.0231	0.9739	1.0663
x2_14	10000	4.9022	2.2083	1.0000	9.0000
x2_50	10000	4.8924	2.1626	1.0000	9.0000
x2_90	10000	4.8263	2.0816	1.0000	8.0000
x1_296	10000	0.4160	0.4929	0	1.0000
x1_304	10000	0.4460	0.4971	0	1.0000
x1_373	10000	0.4443	0.4969	0	1.0000

The odds ratio for x1 is the multiplicative change in the odds of a child wheezing in Steel City compared to the odds of the child wheezing in Green Hills. The estimated odds ratio (ORX1) value is 1.6736 with a corresponding 95% equal-tail credible interval of (1.0248, 2.5939). City of residency is a significant factor in a child's wheezing status. The estimated odds ratio for x2 is the multiplicative change in the odds of developing a wheeze for each additional reported cigarette smoked per day. The odds ratio of ORX2 indicates that the odds of a child developing a wheeze is 1.0150 times higher for each reported cigarette a mother smokes. The corresponding 95% equal-tail credible interval is (0.9695, 1.0619). Since this interval contains the value 1, maternal smoking is not considered to be an influential effect.

Example 80.11: Nonignorably Missing Data (MNAR) Analysis

(View the complete code for this example (mcmcex11.sas) in the example repository.)

This example illustrates how to fit a nonignorably missing data model (MNAR) with PROC MCMC. For a short overview of missing data problems, see the section "Handling of Missing Data" on page 6352.

This data set comes from an environmental study that involve workers in a cotton factory. A similar data set was analyzed from Ibrahim, Chen, and Lipsitz (2001). There are 912 workers in the data set, and the response variable of interest is whether they develop dyspnea (difficult or labored respiration). The data are collected over three time points, and there are six covariates. The following statements create the data set:

```
title 'Nonignorably Missing Data Analysis';
data dyspnea;
   input smoke1 smoke2 smoke3 y1 y2 y3 yrswrk1 yrswrk2 yrswrk3
      age expd sex hgt;
   datalines;
```

```
 0  0  0  0  0  0  28.1  33.1  39.1  48 1 1 165.0
 0  0  0  0  .  0   5.1  10.1  16.1  45 1 0 147.0
 0  0  0  0  .  0  26.0  31.0  37.0  46 1 0 156.0

   ... more lines ...

 1  1  1  0  .  .   6.0  11.0  17.0  25 0 1 180.0
 0  0  0  0  .  .  20.0  25.0  31.0  42 0 0 159.0
;
```

The following variables are included in the data set:

- y1, y2, and y3: dichotomous outcome at the three time periods, which takes the value 1 if the worker has dyspnea, 0 if not (there are missing values in y2 and y3)
- smoke1, smoke2, smoke3: smoking status (0=no, and 1=yes)
- yrswrk1, yrswrk2, yrswrk3: years worked at the cotton factory
- age: age of the worker
- expd: cotton dust exposure (0=no, 1=yes)
- sex: gender (0=female, 1=male)
- hgt: height of the worker

Prior to the analysis, three missing data indicator variables (r1, r2, and r3, one for each of the response variables) are created, and they are set to 1 if the response variable is missing, and 0 otherwise. The covariates age, hgt, yrswrk1, yrswkr2, and yrswrk3 are standardized:

```
data dyspnea;
   array y[3] y1-y3;
   array r[3];
   set dyspnea;
   do i = 1 to 3;
      if y[i] = . then r[i] = 1;
      else r[i] = 0;
   end;
   output;
run;

proc standard data=dyspnea out=dyspnea mean=0 std=1;
   var age hgt yrswrk:;
run;
```

There are no missing values in response variable y1, 128 missing values in y2, and 131 in y3. Ibrahim, Chen, and Lipsitz (2001) used a logistic regression for each of the response variables, where δ_i is a scalar random effect on the observational level:

$$
\begin{aligned}
\mathrm{y}_{ki} &\sim \mathrm{binary}(p_{ki}) \quad k = 1,2,3; \quad i = 1,\ldots,912 \\
p_{ki} &= \mathrm{logistic}(\mu_{ki} + \delta_i) \\
\mu_{ki} &= \beta_1 + \beta_2 \cdot \mathrm{expd}_i + \beta_3 \cdot \mathrm{sex}_i + \beta_4 \cdot \mathrm{hgt}_i + \beta_5 \cdot \mathrm{age}_i + \beta_6 \cdot \mathrm{yrswrk}_{ki} + \beta_7 \cdot \mathrm{smoke}_{ki} \\
\delta_i &\sim \mathrm{n}(0, \sigma^2)
\end{aligned}
$$

Ibrahim, Chen, and Lipsitz (2001) noted that taking δ_i to be higher dimensional (3) would make the model either not identifiable or nearly not identifiable because of the multiple missing values for some subjects.

The first response variable y1 does not contain any missing values, making it meaningless to model the corresponding r1 because every value is 1. Hence, only r2 and r3 are considered in the missing mechanism part of the model. Ibrahim, Chen, and Lipsitz (2001) suggest the following logistic regression for r2 and r3, where the regression mean for each r depends not only on the current response variable y but also the response from previous time period:

$$
\begin{aligned}
\mathsf{r}_{ki} &\sim \text{binary}(q_{ki}) \quad k = 2, 3; \quad i = 1, \ldots, 912 \\
q_{ki} &= \text{logistic}(\nu_{ki}) \\
\mu_c &= \phi_1 + \phi_2 \cdot \text{expd}_i + \phi_3 \cdot \text{sex}_i + \phi_4 \cdot \text{hgt}_i + \phi_5 \cdot \text{age}_i + \phi_6 \cdot \text{yrswrk}_{ki} + \phi_7 \cdot \text{smoke}_{ki} \\
\nu_{2i} &= \mu_c + \phi_8 \cdot \mathsf{y}_{1i} + \phi_9 \cdot \mathsf{y}_{2i} \\
\nu_{3i} &= \mu_c + \phi_9 \cdot \mathsf{y}_{2i} + \phi_{10} \cdot \mathsf{y}_{3i}
\end{aligned}
$$

The missing mechanism model introduces an additional 10 parameters to the model. Normal priors with large standard deviations are used here.

The following statements fit a nonignorably missing model to the dyspnea data set:

```
ods select MissDataInfo REParameters Postsumint;
proc mcmc data=dyspnea seed=17 outpost=dysp2 nmc=20000
     propcov=simplex diag=none monitor=(beta1-beta7);
   array p[3];
   array yrswrk[3];
   array smoke[3];

   parms beta1-beta7 s2;
   parms phi1-phi10;
   prior beta: phi: ~ n(0, var=1e6);
   prior s2 ~ igamma(2, scale=2);
   random d ~ n(0, var=s2) subject=_obs_;
   mu = beta1 + beta2*expd + beta3*sex + beta4*hgt + beta5*age + d;
   do j = 1 to 3;
      p[j] = logistic(mu + beta6*yrswrk[j] + beta7*smoke[j]);
   end;
   model y1 ~ binary(p1);
   model y2 ~ binary(p2);
   model y3 ~ binary(p3);

   nu = phi1 + phi2*expd + phi3*sex + phi4*hgt + phi5*age;
   q2 = logistic(nu + phi6*yrswrk[2] + phi7*smoke[2] +
                 phi8*y1 + phi9*y2);
   model r2 ~ binary(q2);
   q3 = logistic(nu + phi6*yrswrk[3] + phi7*smoke[3] +
                 phi9*y2 + phi10*y3);
   model r3 ~ binary(q3);
run;
```

The first ARRAY statement declares an array p of size 3. This arrays stores three binary probabilities of the response variables. The next two ARRAY statements create storage arrays for some of yrswrk and smoke variables for later programming convenience. The first PARMS statement declares eight parameters, $\beta_1 - \beta_7$

and σ^2. The second PARMS statement declares the 10 ϕ parameters for the missing mechanism model. The PRIOR statements assign prior distributions to these parameters.

The RANDOM statement defines an observational-level random effect d that has a normal prior with variance s2. The SUBJECT=_OBS_ option enables the specification of individual random effects without an explicit input data set variable.

The MU assignment statement and the following DO loop statements calculate the binary probabilities for the three response variables. Note that different yrswrk and smoke variables are used in the DO loop for different years. The three MODEL statements assign three binary distributions to the response variables.

The NU assignment statement starts the calculation for the regression mean in the logistic model for r2 and r3. The variables q2 and q3 are the binary probabilities for the missing mechanisms. Note that their calculations are conditional on the response variables y (selection model). The last two MODEL statements for r2 and r3 complete the specification of the models.

Missing data information and random-effects parameters information are displayed in Output 80.11.1. You can read the total number of missing observations from each variable and its indices from the table. The missing values are sampled using the inverse CDF method. There are 912 random-effects parameters in the model.

Output 80.11.1 Missing Data and Random-Effects Information

Nonignorably Missing Data Analysis

The MCMC Procedure

Missing Data Information Table

Variable	Number of Missing Obs	Observation Indices	Sampling Method
y2	128	2 3 9 11 13 19 20 21 30 31 32 35 39 40 43 56 58 71 75 95 ...	Inverse CDF
y3	131	9 14 16 20 21 29 31 32 43 45 56 72 86 115 117 121 124 142 149 160 ...	Inverse CDF

Random Effect Parameters

Parameter	Sampling Method	Subject	Number of Subjects	Subject Values	Prior Distribution
d	N-Metropolis	_OBS_	912	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 ...	normal(0, var=s2)

The posterior summary and interval statistics of all the β parameters are shown in Output 80.11.2. There are a number of significant regression coefficients in modeling the probability of a worker developing dyspnea, including those for expd (β_2), sex (β_3), age (β_5), and smoke (β_7).

Output 80.11.2 Posterior Summary Statistics for β

Nonignorably Missing Data Analysis

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta1	20000	-2.3365	0.1752	-2.6819	-1.9912
beta2	20000	0.5367	0.1523	0.2570	0.8335
beta3	20000	-0.5920	0.2560	-1.0681	-0.0768
beta4	20000	-0.0660	0.1044	-0.2728	0.1437
beta5	20000	0.6210	0.1730	0.3005	0.9720
beta6	20000	-0.1700	0.1651	-0.4627	0.1833
beta7	20000	0.5799	0.2186	0.1521	1.0029

Example 80.12: Change Point Models

(View the complete code for this example (mcmcex12.sas) in the example repository.)

Consider the data set from Bacon and Watts (1971), where y_i is the logarithm of the height of the stagnant surface layer and the covariate x_i is the logarithm of the flow rate of water. The following statements create the data set:

```
title 'Change Point Model';
data stagnant;
   input y x @@;
   ind = _n_;
   datalines;
 1.12  -1.39   1.12  -1.39   0.99  -1.08   1.03  -1.08
 0.92  -0.94   0.90  -0.80   0.81  -0.63   0.83  -0.63
 0.65  -0.25   0.67  -0.25   0.60  -0.12   0.59  -0.12
 0.51   0.01   0.44   0.11   0.43   0.11   0.43   0.11
 0.33   0.25   0.30   0.25   0.25   0.34   0.24   0.34
 0.13   0.44  -0.01   0.59  -0.13   0.70  -0.14   0.70
-0.30   0.85  -0.33   0.85  -0.46   0.99  -0.43   0.99
-0.65   1.19
;
```

A scatter plot (Output 80.12.1) shows the presence of a nonconstant slope in the data. This suggests a change point regression model (Carlin, Gelfand, and Smith 1992). The following statements generate the scatter plot in Output 80.12.1:

```
proc sgplot data=stagnant;
   scatter x=x y=y;
run;
```

Output 80.12.1 Scatter Plot of the Stagnant Data Set

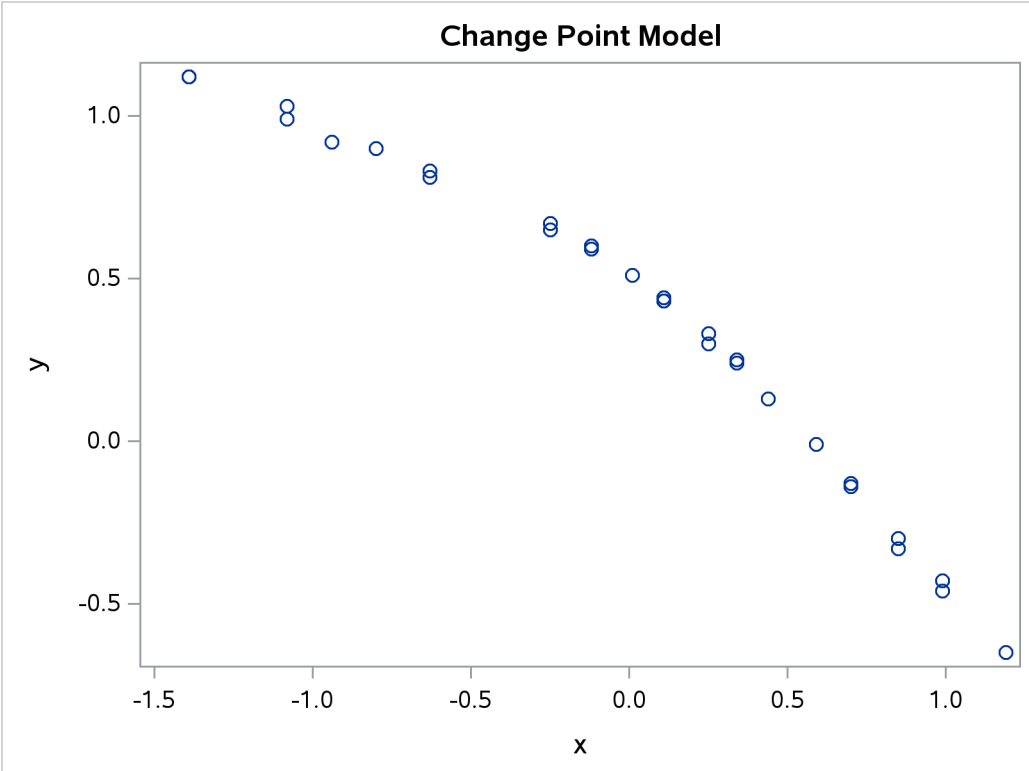

Let the change point be cp. Following formulation by Spiegelhalter et al. (1996b), the regression model is as follows:

$$y_i \sim \begin{cases} \text{normal}(\alpha + \beta_1(x_i - \text{cp}), \sigma^2) & \text{if } x_i < \text{cp} \\ \text{normal}(\alpha + \beta_2(x_i - \text{cp}), \sigma^2) & \text{if } x_i >= \text{cp} \end{cases}$$

You might consider the following diffuse prior distributions:

$$\begin{aligned} \text{cp} &\sim \text{uniform}(-1.3, 1.1) \\ \alpha, \beta_1, \beta_2 &\sim \text{normal}(0, \text{var} = 1e6) \\ \sigma^2 &\sim \text{uniform}(0, 5) \end{aligned}$$

The following statements generate Output 80.12.2:

```
proc mcmc data=stagnant outpost=postout seed=24860 ntu=1000
          nmc=20000;
   ods select PostSumInt;
   ods output PostSumInt=ds;

   array beta[2];
   parms alpha cp beta1 beta2;
   parms s2;

   prior cp ~ unif(-1.3, 1.1);
   prior s2  ~ uniform(0, 5);
```

```
   prior alpha beta:  ~ normal(0, v = 1e6);

   j = 1 + (x >= cp);
   mu = alpha + beta[j] * (x - cp);
   model y ~ normal(mu, var=s2);
run;
```

The PROC MCMC statement specifies the input data set (Stagnant), the output data set (Postout), a random number seed, a tuning sample of 1000, and an MCMC sample of 20000. The ODS SELECT statement displays only the summary statistics table. The ODS OUTPUT statement saves the summary statistics table to the data set Ds.

The ARRAY statement allocates an array of size 2 for the beta parameters. You can use beta1 and beta2 as parameter names without allocating an array, but having the array makes it easier to construct the likelihood function. The two PARMS statements put the five model parameters in two blocks. The three PRIOR statements specify the prior distributions for these parameters.

The symbol j indicates the segment component of the regression. When x is less than the change point, **(x >= cp)** returns 0 and j is assigned the value 1; if x is greater than or equal to the change point, **(x >= cp)** returns 1 and j is 2. The symbol mu is the mean for the jth segment, and beta[j] changes between the two regression coefficients depending on the segment component. The MODEL statement assigns the normal model to the response variable y.

Posterior summary statistics are shown in Output 80.12.2.

Output 80.12.2 MCMC Estimates of the Change Point Regression Model

Change Point Model

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
alpha	20000	0.5349	0.0249	0.4843	0.5813
cp	20000	0.0283	0.0314	-0.0353	0.0846
beta1	20000	-0.4200	0.0146	-0.4482	-0.3911
beta2	20000	-1.0136	0.0167	-1.0476	-0.9817
s2	20000	0.000451	0.000145	0.000220	0.000735

You can use PROC SGPLOT to visualize the model fit. Output 80.12.3 shows the fitted regression lines over the original data. In addition, on the bottom of the plot is the kernel density of the posterior marginal distribution of cp, the change point. The kernel density plot shows the relative variability of the posterior distribution on the data plot. You can use the following statements to create the plot:

```
data _null_;
   set ds;
   call symputx(parameter, mean);
run;

data b;
   missing A;
   input x1 @@;
```

```
   if x1 eq .A then x1 = &cp;
   if _n_ <= 2 then y1 = &alpha + &beta1 * (x1 - &cp);
   else y1 = &alpha + &beta2 * (x1 - &cp);
   datalines;
   -1.5 A 1.2
;

proc kde data=postout;
   univar cp / out=m1 (drop=count);
run;

data m1;
   set m1;
   density = (density / 25) - 0.653;
run;

data all;
   set stagnant b m1;
run;

proc sgplot data=all noautolegend;
   scatter x=x y=y;
   series x=x1 y=y1 / lineattrs = graphdata2;
   series x=value y=density / lineattrs = graphdata1;
run;
```

The macro variables &alpha, &beta1, &beta2, and &cp store the posterior mean estimates from the data set Ds. The data set b contains three predicted values, at the minimum and maximum values of x and the estimated change point &cp. These input values give you fitted values from the regression model. Data set M1 contains the kernel density estimates of the parameter cp. The density is scaled down so the curve would fit in the plot. Finally, you use PROC SGPLOT to overlay the scatter plot, regression line and kernel density plots in the same graph.

Output 80.12.3 Estimated Fit to the Stagnant Data Set

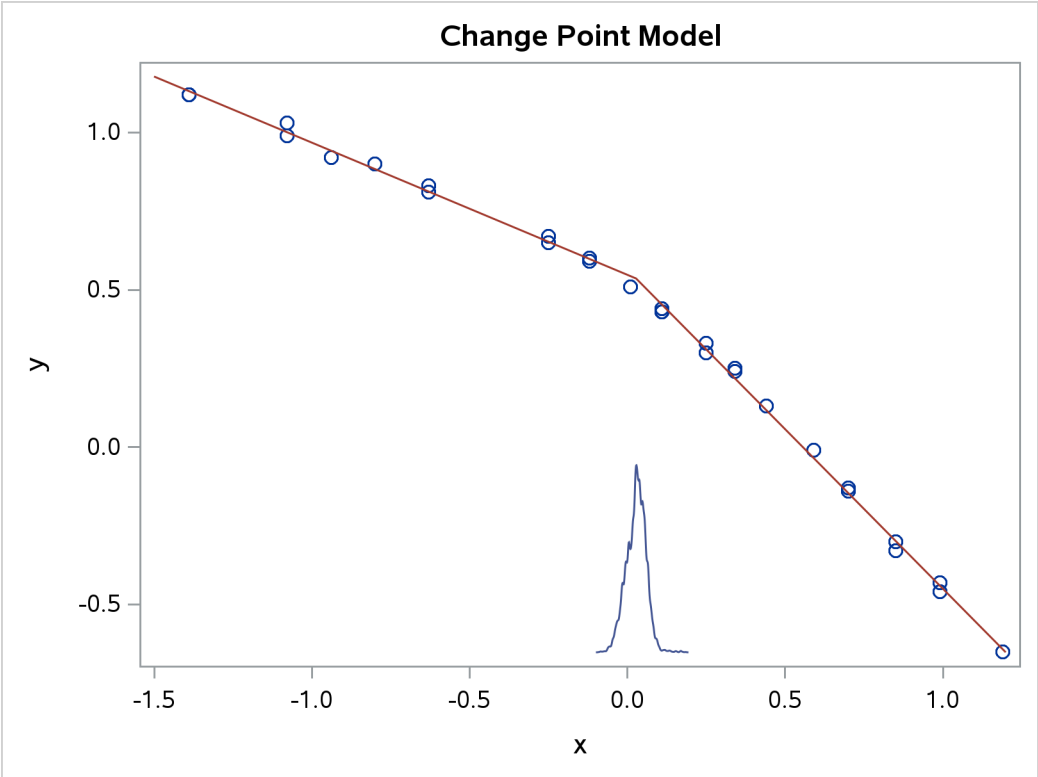

Example 80.13: Exponential and Weibull Survival Analysis

(View the complete code for this example (mcmcex13.sas) in the example repository.)

This example covers two commonly used survival analysis models: the exponential model and the Weibull model. The deviance information criterion (DIC) is used to do model selections, and you can also find programs that visualize posterior quantities. Exponential and Weibull models are widely used for survival analysis. This example shows you how to use PROC MCMC to analyze the treatment effect for the E1684 melanoma clinical trial data. These data were collected to assess the effectiveness of using interferon alpha-2b in chemotherapeutic treatment of melanoma. The following statements create the data set:

```
data e1684;
   input t t_cen treatment @@;
   if t = . then do;
      t = t_cen;
      v = 0;
   end;
   else
      v = 1;
   ifn = treatment - 1;
   et = exp(t);
   lt = log(t);
   drop t_cen;
   datalines;
```

```
  1.57808  0.00000  2   1.48219  0.00000  2    .       7.33425  1
  2.23288  0.00000  1    .       9.38356  2   3.27671  0.00000  1
   .       9.64384  1   1.66575  0.00000  2   0.94247  0.00000  1
  1.68767  0.00000  2   2.34247  0.00000  2   0.89863  0.00000  1

    ... more lines ...

  3.39178  0.00000  1   .        4.36164  2   .        4.81918  2
;
```

The data set E1684 contains the following variables: t is the failure time that equals the censoring time whether the observation was censored, v indicates whether the observation is an actual failure time or a censoring time, treatment indicates two levels of treatments, and ifn indicates the use of interferon as a treatment. The variables et and lt are the exponential and logarithm transformation of the time t. The published data contains other potential covariates that are not listed here. This example concentrates on the effectiveness of the interferon treatment.

Exponential Survival Model

The density function for exponentially distributed survival times is as follows:

$$p(t_i|\lambda_i) = \lambda_i \exp(-\lambda_i t_i)$$

Note that this formulation of the exponential distribution is different from what is used in the SAS probability function PDF. The definition used in PDF for the exponential distributions is as follows:

$$p(t_i|\nu_i) = \frac{1}{\nu_i} \exp(-\frac{t_i}{\nu_i})$$

The relationship between λ and ν is as follows:

$$\lambda_i = \frac{1}{\nu_i}$$

The corresponding survival function, using the λ_i formulation, is as follows:

$$S(t_i|\lambda_i) = \exp(-\lambda_i t_i)$$

If you have a sample $\{t_i\}$ of n independent exponential survival times, each with mean λ_i, then the likelihood function in terms of λ is as follows:

$$\begin{aligned} L(\lambda|t) &= \Pi_{i=1}^{n} p(t_i|\lambda_i)^{\nu_i} S(t_i|\lambda_i)^{1-\nu_i} \\ &= \Pi_{i=1}^{n} (\lambda_i \exp(-\lambda_i t_i))^{\nu_i} (\exp(-\lambda_i t_i))^{1-\nu_i} \\ &= \Pi_{i=1}^{n} \lambda_i^{\nu_i} \exp(-\lambda_i t_i) \end{aligned}$$

If you link the covariates to λ with $\lambda_i = \exp \mathbf{x}_i' \boldsymbol{\beta}$, where $\mathbf{x}_i$ is the vector of covariates corresponding to the ith observation and $\boldsymbol{\beta}$ is a vector of regression coefficients, then the log-likelihood function is as follows:

$$l(\boldsymbol{\beta}|t, \mathbf{x}) = \sum_{i=1}^{n} \nu_i \mathbf{x}_i' \boldsymbol{\beta} - t_i \exp(\mathbf{x}_i' \boldsymbol{\beta})$$

In the absence of prior information about the parameters in this model, you can choose diffuse normal priors for the β:

$$\beta \sim \text{normal}(0, \text{sd} = 10000)$$

There are two ways to program the log-likelihood function in PROC MCMC. You can use the SAS functions LOGPDF and LOGSDF. Alternatively, you can use the simplified log-likelihood function, which is more computationally efficient. You get identical results by using either approaches.

The following PROC MCMC statements fit an exponential model with simplified log-likelihood function:

```
title 'Exponential Survival Model';
ods graphics on;
proc mcmc data=e1684 outpost=expsurvout nmc=10000 seed=4861
   diag=(mcse ess);
   ods select PostSumInt TADpanel
              ess mcse;
   parms (beta0 beta1) 0;
   prior beta: ~ normal(0, sd = 10000);
   /*****************************************************/
   /* (1) the logpdf and logsdf functions are not used  */
   /*****************************************************/
   /*     nu = 1/exp(beta0 + beta1*ifn);
          llike = v*logpdf("exponential", t, nu) +
                  (1-v)*logsdf("exponential", t, nu);
   */
   /****************************************************/
   /* (2) the simplified likelihood formula is used    */
   /****************************************************/
   l_h = beta0 + beta1*ifn;
   llike = v*(l_h) -  t*exp(l_h);
   model general(llike);
run;
```

The two assignment statements that are commented out calculate the log-likelihood function by using the SAS functions LOGPDF and LOGSDF for the exponential distribution. The next two assignment statements calculate the log likelihood by using the simplified formula. The first approach is slower because of the redundant calculation involved in calling both LOGPDF and LOGSDF.

An examination of the trace plots for β_0 and β_1 (see Output 80.13.1) reveals that the sampling has gone well with no particular concerns about the convergence or mixing of the chains.

Output 80.13.1 Posterior Plots for β_0 and β_1 in the Exponential Survival Analysis

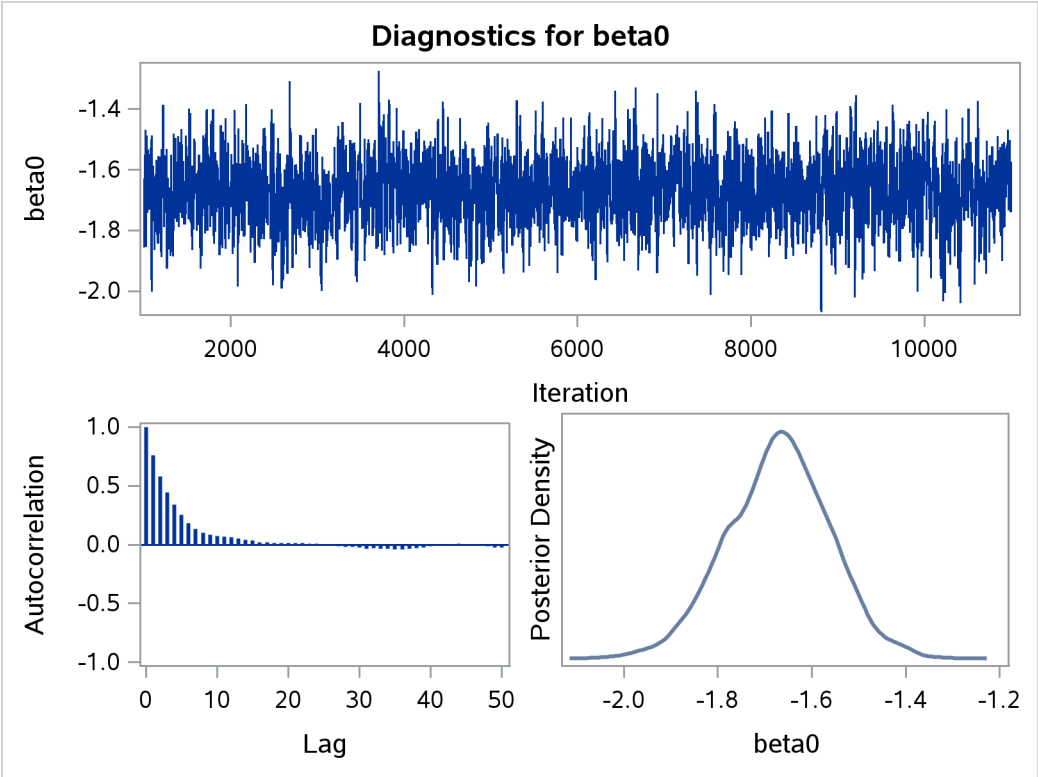

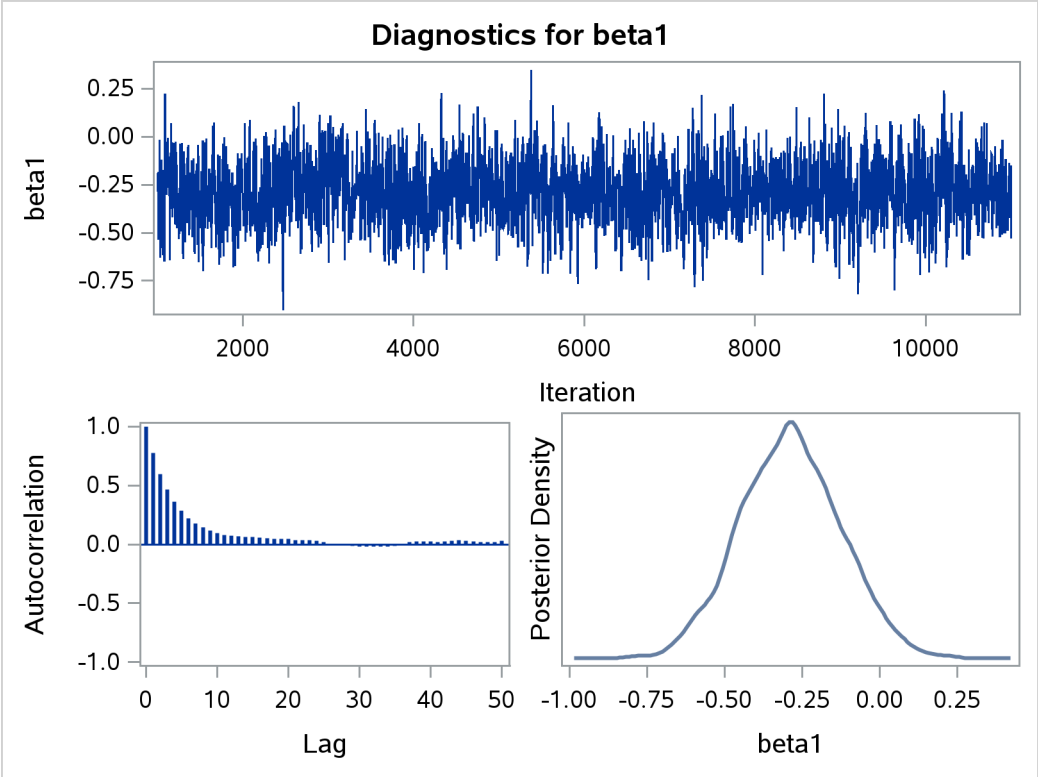

The MCMC results are shown in Output 80.13.2.

Output 80.13.2 Posterior Summary and Interval Statistics

Exponential Survival Model

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta0	10000	-1.6715	0.1091	-1.8930	-1.4673
beta1	10000	-0.2879	0.1615	-0.6104	0.0169

The Monte Carlo standard errors and effective sample sizes are shown in Output 80.13.3. The posterior means for β_0 and β_1 are estimated with high precision, with small standard errors with respect to the standard deviation. This indicates that the mean estimates have stabilized and do not vary greatly in the course of the simulation. The effective sample sizes are roughly the same for both parameters.

Output 80.13.3 MCSE and ESS

Exponential Survival Model

The MCMC Procedure

Monte Carlo Standard Errors			
Parameter	MCSE	Standard Deviation	MCSE/SD
beta0	0.00302	0.1091	0.0277
beta1	0.00485	0.1615	0.0301

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
beta0	1304.1	7.6682	0.1304
beta1	1107.2	9.0319	0.1107

The next part of this example shows fitting a Weibull regression to the data and then comparing the two models with DIC to see which one provides a better fit to the data.

Weibull Survival Model

The density function for Weibull distributed survival times is as follows:

$$p(t_i|\alpha, \lambda_i) = \alpha t_i^{\alpha-1} \exp(\lambda_i - \exp(\lambda_i) t_i^{\alpha})$$

Note that this formulation of the Weibull distribution is different from what is used in the SAS probability function PDF. The definition used in PDF is as follows:

$$p(t_i|\alpha, \gamma_i) = \exp\left(-\left(\frac{t_i}{\gamma_i}\right)^{\alpha}\right) \frac{\alpha}{\gamma_i} \left(\frac{t_i}{\gamma_i}\right)^{\alpha-1}$$

The relationship between λ and γ in these two parameterizations is as follows:

$$\lambda_i = -\alpha \log \gamma_i$$

The corresponding survival function, using the λ_i formulation, is as follows:

$$S(t_i|\alpha, \lambda_i) = \exp(-\exp(\lambda_i)t_i^{\alpha})$$

If you have a sample $\{t_i\}$ of n independent Weibull survival times, with parameters α, and λ_i, then the likelihood function in terms of α and λ is as follows:

$$\begin{aligned} L(\alpha, \lambda|t) &= \Pi_{i=1}^{n} p(t_i|\alpha, \lambda_i)^{\nu_i} S(t_i|\alpha, \lambda_i)^{1-\nu_i} \\ &= \Pi_{i=1}^{n} (\alpha t_i^{\alpha-1} \exp(\lambda_i - \exp(\lambda_i)t_i^{\alpha}))^{\nu_i} (\exp(-\exp(\lambda_i)t_i^{\alpha}))^{1-\nu_i} \\ &= \Pi_{i=1}^{n} (\alpha t_i^{\alpha-1} \exp(\lambda_i))^{\nu_i} (\exp(-\exp(\lambda_i)t_i^{\alpha})) \end{aligned}$$

If you link the covariates to λ with $\lambda_i = \mathbf{x}_i'\boldsymbol{\beta}$, where $\mathbf{x}_i$ is the vector of covariates corresponding to the ith observation and $\boldsymbol{\beta}$ is a vector of regression coefficients, the log-likelihood function becomes this:

$$l(\alpha, \boldsymbol{\beta}|t, \mathbf{x}) = \sum_{i=1}^{n} \nu_i (\log(\alpha) + (\alpha - 1)\log(t_i) + \mathbf{x}_i'\boldsymbol{\beta}) - \exp(\mathbf{x}_i'\boldsymbol{\beta})t_i^{\alpha})$$

As with the exponential model, in the absence of prior information about the parameters in this model, you can use diffuse normal priors on β. You might want to choose a diffuse gamma distribution for α. Note that when $\alpha = 1$, the Weibull survival likelihood reduces to the exponential survival likelihood. Equivalently, by looking at the posterior distribution of α, you can conclude whether fitting an exponential survival model would be more appropriate than the Weibull model.

PROC MCMC also enables you to make inference on any functions of the parameters. Quantities of interest in survival analysis include the value of the survival function at specific times for specific treatments and the relationship between the survival curves for different treatments. With PROC MCMC, you can compute a sample from the posterior distribution of the interested survival functions at any number of points. The data in this example range from about 0 to 10 years, and the treatment of interest is the use of interferon.

Like in the previous exponential model example, there are two ways to fit this model: using the SAS functions LOGPDF and LOGSDF, or using the simplified log likelihood functions. The example uses the latter method. The following statements run PROC MCMC and produce Output 80.13.4:

```
title 'Weibull Survival Model';
proc mcmc data=e1684 outpost=weisurvout nmc=10000 seed=1234
          monitor=(_parms_ surv_ifn surv_noifn) stats=(summary intervals);
   ods select PostSummaries;
   ods output PostSummaries=ds PostIntervals=is;
   array surv_ifn[10];
   array surv_noifn[10];
   parms alpha 1 (beta0 beta1) 0;
   prior beta: ~ normal(0, var=10000);
   prior alpha ~ gamma(0.001,is=0.001);

   beginnodata;
      do t1 = 1 to 10;
         surv_ifn[t1] = exp(-exp(beta0+beta1)*t1**alpha);
         surv_noifn[t1] = exp(-exp(beta0)*t1**alpha);
      end;
   endnodata;
```

```
    lambda = beta0 + beta1*ifn;
    /*****************************************************/
    /* (1) the logpdf and logsdf functions are not used  */
    /*****************************************************/
    /*     gamma =  exp(-lambda /alpha);
           llike = v*logpdf('weibull', t, alpha, gamma) +
                   (1-v)*logsdf('weibull', t, alpha, gamma);
    */
    /*****************************************************/
    /* (2) the simplified likelihood formula is used    */
    /*****************************************************/
    llike  = v*(log(alpha) + (alpha-1)*log(t) + lambda) -
             exp(lambda)*(t**alpha);
    model general(llike);
  run;
```

The MONITOR= option indicates the parameters and quantities of interest that PROC MCMC tracks. The symbol _PARMS_ specifies all model parameters. The array surv_ifn stores the expected survival probabilities for patients who received interferon over a period of 10 years. Similarly, surv_noifn stores the expected survival probabilities for patients who did not received interferon.

The BEGINNODATA and ENDNODATA statements enclose the calculations for the survival probabilities. The assignment statements proceeding the MODEL statement calculate the log likelihood for the Weibull survival model. The MODEL statement specifies the log likelihood that you programmed.

An examination of the trace plots for α, β_0, and β_1 (not displayed here) reveals that the sampling has gone well, with no particular concerns about the convergence or mixing of the chains.

Output 80.13.4 displays the posterior summary statistics.

Output 80.13.4 Posterior Summary Statistics

Weibull Survival Model

The MCMC Procedure

Posterior Summaries						
				Percentiles		
Parameter	N	Mean	Standard Deviation	25	50	75
alpha	10000	0.7891	0.0539	0.7514	0.7880	0.8260
beta0	10000	-1.3581	0.1369	-1.4519	-1.3597	-1.2624
beta1	10000	-0.2512	0.1541	-0.3541	-0.2606	-0.1521
surv_ifn1	10000	0.8175	0.0227	0.8027	0.8187	0.8331
surv_ifn2	10000	0.7066	0.0291	0.6874	0.7072	0.7265
surv_ifn3	10000	0.6203	0.0331	0.5983	0.6205	0.6436
surv_ifn4	10000	0.5495	0.0360	0.5253	0.5497	0.5747
surv_ifn5	10000	0.4899	0.0381	0.4635	0.4895	0.5170
surv_ifn6	10000	0.4390	0.0396	0.4118	0.4382	0.4666
surv_ifn7	10000	0.3949	0.0406	0.3669	0.3934	0.4223
surv_ifn8	10000	0.3564	0.0413	0.3281	0.3551	0.3840
surv_ifn9	10000	0.3225	0.0416	0.2940	0.3212	0.3505
surv_ifn10	10000	0.2926	0.0416	0.2638	0.2911	0.3208
surv_noifn1	10000	0.7719	0.0274	0.7535	0.7736	0.7913
surv_noifn2	10000	0.6401	0.0339	0.6171	0.6415	0.6635
surv_noifn3	10000	0.5415	0.0374	0.5161	0.5428	0.5662
surv_noifn4	10000	0.4635	0.0395	0.4365	0.4636	0.4890
surv_noifn5	10000	0.4001	0.0406	0.3725	0.3995	0.4261
surv_noifn6	10000	0.3475	0.0411	0.3195	0.3459	0.3745
surv_noifn7	10000	0.3034	0.0411	0.2758	0.3012	0.3299
surv_noifn8	10000	0.2661	0.0406	0.2384	0.2630	0.2921
surv_noifn9	10000	0.2342	0.0399	0.2069	0.2311	0.2592
surv_noifn10	10000	0.2069	0.0389	0.1803	0.2035	0.2312

An examination of the α parameter reveals that the exponential model might not be inappropriate here. The estimated posterior mean of α is 0.7856 with a posterior standard deviation of 0.0533. As noted previously, if $\alpha = 1$, then the Weibull survival distribution is the exponential survival distribution. With these data, you can see that the evidence is in favor of $\alpha < 1$. The value 1 is almost 4 posterior standard deviations away from the posterior mean. The following statements compute the posterior probability of the hypothesis that $\alpha < 1$.:

```
proc format;
   value alphafmt low-<1 = 'alpha < 1' 1-high = 'alpha >= 1';
run;

proc freq data=weisurvout;
   tables alpha /nocum;
   format alpha alphafmt.;
run;
```

The PROC FREQ results are shown in Output 80.13.5.

Output 80.13.5 Frequency Analysis of α

Weibull Survival Model

The FREQ Procedure

alpha	Frequency	Percent
alpha < 1	9998	99.98
alpha >= 1	2	0.02

The output from PROC FREQ shows that 100% of the 10000 simulated values for α are less than 1. This is a very strong indication that the exponential model is too restrictive to model these data well.

You can examine the estimated survival probabilities over time individually, either through the posterior summary statistics or by looking at the kernel density plots. Alternatively, you might find it more informative to examine these quantities in relation with each other. For example, you can use a side-by-side box plot to display these posterior distributions by using PROC SGPLOT. For more information, see the section "Statistical Graphics Using ODS" on page 665 in Chapter 24, "Statistical Graphics Using ODS." First you need to take the posterior output data set Weisurvout and stack variables that you want to plot. For example, to plot all the survival times for patients who received interferon, you want to stack surv_inf1–surv_inf10. The macro %Stackdata takes an input data set dataset, stacks the wanted variables vars, and outputs them into the output data set.

The following statements define the macro stackdata:

```
/* define macro stackdata */
%macro StackData(dataset,output,vars);
   data &output;
      length var $ 32;
      if 0 then set &dataset nobs=nnn;
      array lll[*] &vars;
      do jjj=1 to dim(lll);
         do iii=1 to nnn;
            set &dataset point=iii;
            value = lll[jjj];
            call vname(lll[jjj],var);
            output;
         end;
      end;
      stop;
      keep var value;
   run;
%mend;

/* stack the surv_ifn variables and saved them to survifn. */
%StackData(weisurvout, survifn, surv_ifn1-surv_ifn10);
```

Once you stack the data, use PROC SGPLOT to create the side-by-side box plots. The following statements generate Output 80.13.6:

```
proc sgplot data=survifn;
   yaxis label='Survival Probability' values=(0 to 1 by 0.2);
   xaxis label='Time' discreteorder=data;
   vbox value / category=var;
run;
```

Output 80.13.6 Side-by-Side Box Plots of Estimated Survival Probabilities

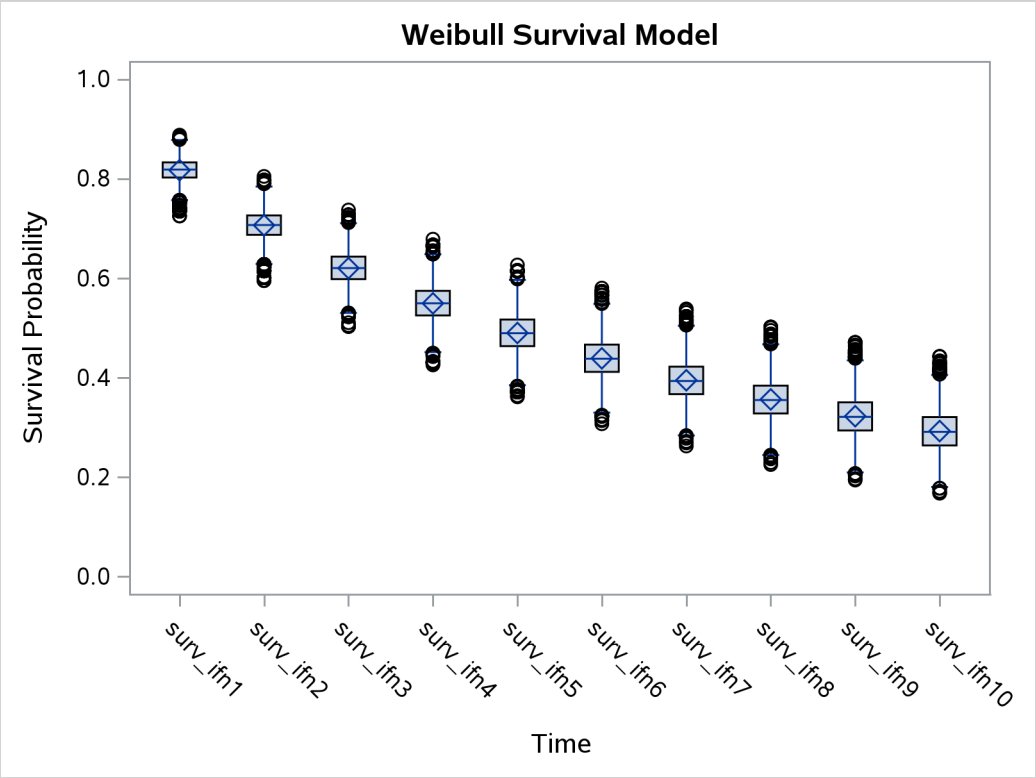

There is a clear decreasing trend over time of the survival probabilities for patients who receive the treatment. You might ask how does this group compare to those who did not receive the treatment? In this case, you want to overlay the two predicted curves for the two groups of patients and add the corresponding credible interval. See Output 80.13.7. To generate the graph, you first take the posterior mean estimates from the ODS output table ds and the lower and upper HPD interval estimates is, store them in the data set Surv, and draw the figure by using PROC SGPLOT.

The following statements generate data set Surv:

```
data surv;
   set ds;
   if _n_ >= 4 then do;
      set is point=_n_;
      group = 'with interferon   ';
      time = _n_ - 3;
      if time > 10 then do;
         time = time - 10;
         group = 'without interferon';
      end;
```

```
      output;
   end;
   keep time group mean hpdlower hpdupper;
run;
```

The following SGPLOT statements generate Output 80.13.7:

```
proc sgplot data=surv;
   yaxis label="Survival Probability" values=(0 to 1 by 0.2);
   series x=time y=mean / group = group name='i';
   band x=time lower=hpdlower upper=hpdupper / group = group transparency=0.7;
   keylegend 'i';
run;
ods graphics off;
```

In Output 80.13.7, the solid line is the survival curve for patients who received interferon; the shaded region centers at the solid line is the 95% HPD intervals; the medium-dashed line is the survival curve for patients who did not receive interferon; and the shaded region around the dashed line is the corresponding 95% HPD intervals.

Output 80.13.7 Predicted Survival Probability Curves with 95% HPD Intervals

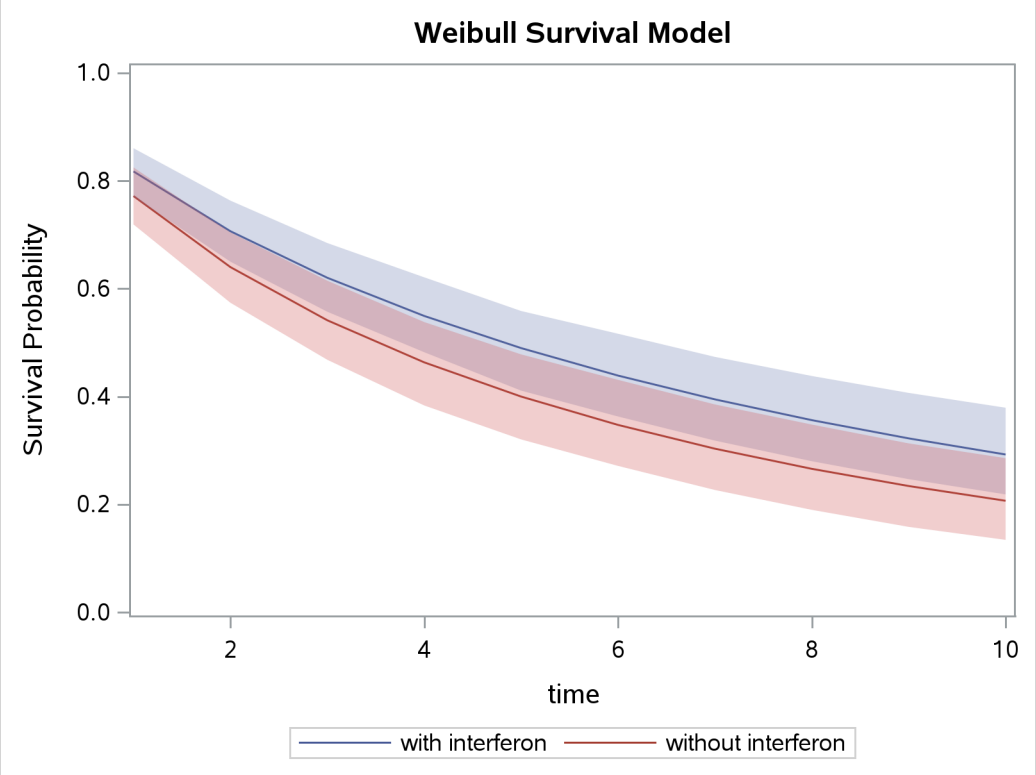

The plot suggests that there is an effect of using interferon because patients who received interferon have sustained better survival probabilities than those who did not. However, the effect might not be very significant, as the 95% credible intervals of the two groups do overlap. For more on these interferon studies, see Ibrahim, Chen, and Lipsitz (2001).

Weibull or Exponential?

Although the evidence from the Weibull model fit shows that the posterior distribution of α has a significant amount of density mass less than 1, suggesting that the Weibull model is a better fit to the data than the exponential model, you might still be interested in comparing the two models more formally. You can use the Bayesian model selection criterion (see the section "Deviance Information Criterion (DIC)" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures") to determine which model fits the data better.

The PROC MCMC DIC option requests the calculation of DIC, and the procedure displays the ODS output table DIC. The table includes the posterior mean of the deviation, $\overline{D(\boldsymbol{\theta})}$, deviation at the estimate, $D(\overline{\boldsymbol{\theta}})$, effective number of parameters, p_D, and DIC. It is important to remember that the standardizing term, $p(\mathbf{y})$, which is a function of the data alone, is not taken into account in calculating the DIC. This term is irrelevant only if you compare two models that have the *same* likelihood function. If you do not have identical likelihood functions, using DIC for model selection purposes without taking this standardizing term into account can produce incorrect results. In addition, you want to be careful in interpreting the DIC whenever you use the GENERAL function to construct the log-likelihood, as the case in this example. Using the GENERAL function, you can obtain identical posterior samples with two log-likelihood functions that differ only by a constant. This difference translates to a difference in the DIC calculation, which could be very misleading.

If $\alpha = 1$, the Weibull likelihood is identical to the exponential likelihood. It is safe in this case to directly compare DICs from these two models. However, if you do not want to work out the mathematical detail or you are uncertain of the equivalence, a better way of comparing the DICs is to run the Weibull model twice: once with α being a parameter and once with $\alpha = 1$. This ensures that the likelihood functions are the same, and the DIC comparison is meaningful.

The following statements fit a Weibull model:

```
title 'Model Comparison between Weibull and Exponential';
proc mcmc data=e1684 outpost=weisurvout nmc=10000 seed=4861 dic;
   ods select dic;
   parms alpha 1 (beta0 beta1) 0;
   prior beta: ~ normal(0, var=10000);
   prior alpha ~ gamma(0.001,is=0.001);

   lambda = beta0 + beta1*ifn;
   llike  = v*(log(alpha) + (alpha-1)*log(t) + lambda) -
            exp(lambda)*(t**alpha);
   model general(llike);
run;
```

The DIC option requests the calculation of DIC, and the table is displayed in Output 80.13.8.

Output 80.13.8 DIC Table from the Weibull Model

Model Comparison between Weibull and Exponential

The MCMC Procedure

Deviance Information Criterion	
Dbar (posterior mean of deviance)	858.623
Dmean (deviance evaluated at posterior mean)	855.633
pD (effective number of parameters)	2.990
DIC (smaller is better)	861.614

The GENERAL or DGENERAL function is used in this program. To make meaningful comparisons, you must ensure that all GENERAL or DGENERAL functions include appropriate normalizing constants. Otherwise, DIC comparisons can be misleading.

The note in Output 80.13.8 reminds you of the importance of ensuring identical likelihood functions when you use the GENERAL function. The DIC value is 861.6.

Based on the same set of code, the following statements fit an exponential model by setting $\alpha = 1$:

```
proc mcmc data=e1684 outpost=expsurvout nmc=10000 seed=4861 dic;
   ods select dic;
   parms beta0 beta1 0;
   prior beta: ~ normal(0, var=10000);
   begincnst;
      alpha = 1;
   endcnst;

   lambda = beta0 + beta1*ifn;
   llike  = v*(log(alpha) + (alpha-1)*log(t) + lambda) -
            exp(lambda)*(t**alpha);
   model general(llike);
run;
```

Output 80.13.9 displays the DIC table.

Output 80.13.9 DIC Table from the Exponential Model

Model Comparison between Weibull and Exponential

The MCMC Procedure

Deviance Information Criterion	
Dbar (posterior mean of deviance)	870.133
Dmean (deviance evaluated at posterior mean)	868.190
pD (effective number of parameters)	1.943
DIC (smaller is better)	872.075

The GENERAL or DGENERAL function is used in this program. To make meaningful comparisons, you must ensure that all GENERAL or DGENERAL functions include appropriate normalizing constants. Otherwise, DIC comparisons can be misleading.

The DIC value of 872.075 is greater than 861. A smaller DIC indicates a better fit to the data; hence, you can conclude that the Weibull model is more appropriate for this data set. You can see the equivalencing of the exponential model you fitted in "Exponential Survival Model" on page 6439 by running the following comparison.

The following statements are taken from the section "Exponential Survival Model" on page 6439, and they fit the same exponential model:

```
proc mcmc data=e1684 outpost=expsurvout1 nmc=10000 seed=4861 dic;
   ods select none;
   parms (beta0 beta1) 0;
   prior beta: ~ normal(0, sd = 10000);
   l_h = beta0 + beta1*ifn;
   llike = v*(l_h) -  t*exp(l_h);
   model general(llike);
run;

proc compare data=expsurvout compare=expsurvout1;
   var beta0 beta1;
run;
```

The posterior samples of beta0 and beta1 in the data set Expsurvout1 are identical to those in the data set Expsurvout. The comparison results are not shown here.

Example 80.14: Time Independent Cox Model

(View the complete code for this example (mcmcex14.sas) in the example repository.)

This example has two purposes. One is to illustrate how to use PROC MCMC to fit a Cox proportional hazard model. Specifically, the time independent model. See "Example 80.15: Time Dependent Cox Model" on page 6458 for an example on fitting time dependent Cox model. Note that it is much easier to fit a Bayesian Cox model by specifying the BAYES statement in PROC PHREG (see Chapter 92, "The PHREG Procedure"). If you are interested only in fitting a Cox regression survival model, you should use PROC PHREG.

The second objective of this example is to demonstrate how to model data that are not independent. That is the case where the likelihood for observation i depends on other observations in the data set. In other words, if you work with a likelihood function that cannot be broken down simply as $L(\mathbf{y}) = \prod_i^n L(y_i)$, you can use this example for illustrative purposes. By default, PROC MCMC assumes that the programming statements and model specification is intended for a single row of observations in the data set. The Cox model is chosen because the complexity in the data structure requires more elaborate coding.

The Cox proportional hazard model is widely used in the analysis of survival time, failure time, or other duration data to explain the effect of exogenous explanatory variables. The data set used in this example is taken from Krall, Uthoff, and Harley (1975), who analyzed data from a study on myeloma in which researchers treated 65 patients with alkylating agents. Of those patients, 48 died during the study and 17 survived. The following statements generate the data set that is used in this example:

```
data Myeloma;
   input Time Vstatus LogBUN HGB Platelet Age LogWBC Frac
         LogPBM Protein SCalc;
   label Time='survival time'
```

```
         VStatus='0=alive 1=dead';
   datalines;
 1.25  1  2.2175   9.4  1  67  3.6628  1  1.9542  12  10
 1.25  1  1.9395  12.0  1  38  3.9868  1  1.9542  20  18
 2.00  1  1.5185   9.8  1  81  3.8751  1  2.0000   2  15
 2.00  1  1.7482  11.3  0  75  3.8062  1  1.2553   0  12

   ... more lines ...

77.00  0  1.0792  14.0  1  60  3.6812  0  0.9542   0  12
;

proc sort data = Myeloma;
   by descending time;
run;

data _null_;
   set Myeloma nobs=_n;
   call symputx('N', _n);
   stop;
run;
```

The variable Time represents the survival time in months from diagnosis. The variable VStatus consists of two values, 0 and 1, indicating whether the patient was alive or dead, respectively, at the end of the study. If the value of VStatus is 0, the corresponding value of Time is censored. The variables thought to be related to survival are LogBUN (log(BUN) at diagnosis), HGB (hemoglobin at diagnosis), Platelet (platelets at diagnosis: 0=abnormal, 1=normal), Age (age at diagnosis in years), LogWBC (log(WBC) at diagnosis), Frac (fractures at diagnosis: 0=none, 1=present), LogPBM (log percentage of plasma cells in bone marrow), Protein (proteinuria at diagnosis), and SCalc (serum calcium at diagnosis). Interest lies in identifying important prognostic factors from these explanatory variables. In addition, there are 65 (&n) observations in the data set Myeloma. The likelihood used in these examples is the Breslow likelihood:

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \left[\prod_{j=1}^{d_i} \frac{\exp(\boldsymbol{\beta}'\mathbf{Z}_j(t_i))}{\sum_{l \in \mathcal{R}_i} \exp(\boldsymbol{\beta}'\mathbf{Z}_l(t_i))} \right]^{v_i}$$

where

- $\boldsymbol{\beta}$ is the vector parameters
- n is the total number of observations in the data set
- t_i is the ith time, which can be either event time or censored time
- $\mathbf{Z}_l(t)$ is the vector explanatory variables for the lth individual at time t
- d_i is the multiplicity of failures at t_i. If there are no ties in time, d_i is 1 for all i.
- $\mathcal{R}_i$ is the risk set for the ith time t_i, which includes all observations that have survival time greater than or equal to t_i
- v_i indicates whether the patient is censored. The value 0 corresponds to censoring. Note that the censored time t_i enters the likelihood function only through the formation of the risk set $\mathcal{R}_i$.

Priors on the coefficients are independent normal priors with very large variance (1e6). Throughout this example, the symbol bZ represents the regression term $\boldsymbol{\beta}'\mathbf{Z}_j(t_i)$ in the likelihood, and the symbol S represents the term $\sum_{l \in \mathcal{R}_i} \exp(\boldsymbol{\beta}'\mathbf{Z}_l(t_i))$.

The regression model considered in this example uses the following formula:

$$\begin{aligned}\boldsymbol{\beta}'\mathbf{Z}_j &= \beta_1\text{logbun} + \beta_2\text{hgb} + \beta_3\text{platelet} + \beta_4\text{age} + \\ &\quad \beta_5\text{logwbc} + \beta_6\text{frac} + \beta_7\text{logpbm} + \beta_8\text{protein} + \beta_9\text{scalc}\end{aligned}$$

The hard part of coding this in PROC MCMC is the construction of the risk set $\mathcal{R}_i$. $\mathcal{R}_i$ contains all observations that have survival time greater than or equal to t_i. First suppose that there are no ties in time. Sorting the data set by the variable time into descending order gives you $\mathcal{R}_i$ that is in the right order. Observation i's risk set consists of all data points j such that $j <= i$ in the data set. You can cumulatively increment S in the SAS statements.

With potential ties in time, at observation i, you need to know whether any subsequent observations, $i + 1$ and so on, have the same survival time as t_i. Suppose that the ith, the $i + 1$, and the $i + 2$ observations all have the same survival time; all three of them need to be included in the risk set calculation. This means that to calculate the likelihood for some observations, you need to access both the previous and subsequent observations in the data set. There are two ways to do this. One is to use the LAG function; the other is to use the option JOINTMODEL.

The LAG function returns values from a queue (see *SAS Functions and CALL Routines: Reference*). So for the ith observation, you can use LAG1 to access variables from the previous row in the data set. You want to compare the lag1 value of time with the current time value. Depending on whether the two time values are equal, you can add correction terms in the calculation for the risk set S.

The following statements sort the data set by time into descending order, with the largest survival time on top:

```
title 'Cox Model with Time Independent Covariates';
proc freq data=myeloma;
   ods select none;
   tables time / out=freqs;
run;

proc sort data = freqs;
   by descending time;
run;

data myelomaM;
   set myeloma;
   ind = _N_;
run;
ods select all;
```

The following statements run PROC MCMC and produce Output 80.14.1:

```
proc mcmc data=myelomaM outpost=outi nmc=50000 ntu=3000 seed=1;
   ods select PostSumInt;
   array beta[9];
   parms beta: 0;
   prior beta: ~ normal(0, var=1e6);
```

```
   bZ = beta1 * LogBUN + beta2 * HGB + beta3 * Platelet
      + beta4 * Age + beta5 * LogWBC + beta6 * Frac +
      beta7 * LogPBM + beta8 * Protein + beta9 * SCalc;

   if ind = 1 then do;        /* first observation        */
      S = exp(bZ);
      l = vstatus * bZ;
      v = vstatus;
   end;
   else if (1 < ind < &N) then do;
      if (lag1(time) ne time) then do;
         l = vstatus * bZ;
         l = l - v * log(S);  /* correct the loglike value   */
         v = vstatus;         /* reset v count value         */
         S = S + exp(bZ);
      end;
      else do;                /* still a tie                 */
         l = vstatus * bZ;
         S = S + exp(bZ);
         v = v + vstatus;     /* add # of nonsensored values */
      end;
   end;
   else do;                   /* last observation            */
      if (lag1(time) ne time) then do;
         l = - v * log(S);    /* correct the loglike value   */
         S = S + exp(bZ);
         l = l + vstatus * (bZ - log(S));
      end;
      else do;
         S = S + exp(bZ);
         l = vstatus * bZ - (v + vstatus) * log(S);
      end;
   end;
   model general(l);
run;
```

The symbol bZ is the regression term, which is independent of the time variable. The symbol ind indexes observation numbers in the data set. The symbol S keeps track of the risk set term for every observation. The symbol l calculates the log likelihood for each observation. Note that the value of l for observation ind is not necessarily the correct log likelihood value for that observation, especially in cases where the observation ind is in the tied times. Correction terms are added to subsequent values of l when the time variable becomes different in order to make up the difference. The total sum of l calculated over the entire data set is correct. The symbol v keeps track of the sum of vstatus, as censored data do not enter the likelihood and need to be taken out.

You use the function LAG1 to detect if two adjacent time values are different. If they are, you know that the current observation is in a different risk set than the last one. You then need to add a correction term to the log likelihood value of l. The IF-ELSE statements break the observations into three parts: the first observation, the last observation and everything in the middle.

Output 80.14.1 Summary Statistics on Cox Model with Time Independent Explanatory Variables and Ties in the Survival Time, Using PROC MCMC

Cox Model with Time Independent Covariates

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta1	50000	1.7600	0.6441	0.5117	3.0465
beta2	50000	-0.1308	0.0720	-0.2746	0.00524
beta3	50000	-0.2017	0.5148	-1.2394	0.7984
beta4	50000	-0.0126	0.0194	-0.0512	0.0245
beta5	50000	0.3373	0.7256	-1.1124	1.7291
beta6	50000	0.3992	0.4337	-0.4385	1.2575
beta7	50000	0.3749	0.4861	-0.5423	1.3689
beta8	50000	0.0106	0.0271	-0.0451	0.0616
beta9	50000	0.1272	0.1064	-0.0763	0.3406

An alternative to using the LAG function is to use the PROC option JOINTMODEL. With this option, the log-likelihood function you specify applies not to a single observation but to the entire data set. See "Modeling Joint Likelihood" on page 6319 for details on how to properly use this option. The basic idea is that you store all necessary data set variables in arrays and use only the arrays to construct the log likelihood of the entire data set. This approach works here because for every observation *i*, you can use index to access different values of arrays to construct the risk set S. To use the JOINTMODEL option, you need to do some additional data manipulation. You want to create a stop variable for each observation, which indicates the observation number that should be included in S for that observation. For example, if observations 4, 5, 6 all have the same survival time, the stop value for all of them is 6.

The following statements generate a new data set MyelomaM that contains the stop variable:

```
data myelomaM;
   merge myelomaM freqs(drop=percent);
   by descending time;
   retain stop;
   if first.time then do;
      stop = _n_ + count - 1;
   end;
run;
```

The following SAS program fits the same Cox model by using the JOINTMODEL option:

```
data a;
run;

proc mcmc data=a outpost=outa nmc=50000 ntu=3000 seed=1 jointmodel;
   ods select none;
   array beta[9];
   array data[1] / nosymbols;
   array timeA[1] / nosymbols;
   array vstatusA[1] / nosymbols;
   array stopA[1] / nosymbols;
```

```
      array bZ[&n];
      array S[&n];

      begincnst;
      rc = read_array("myelomam", data, "logbun", "hgb", "platelet",
                      "age", "logwbc", "frac", "logpbm", "protein", "scalc");
      rc = read_array("myelomam", timeA, "time");
      rc = read_array("myelomam", vstatusA, "vstatus");
      rc = read_array("myelomam", stopA, "stop");
      endcnst;

      parms (beta:) 0;
      prior beta: ~ normal(0, var=1e6);

      jl = 0;
      /* calculate each bZ and cumulatively adding S as if there are no ties.*/
      call mult(data, beta, bZ);
      S[1] = exp(bZ[1]);
      do i = 2 to &n;
         S[i] = S[i-1] + exp(bZ[i]);
      end;

      do i = 1 to &n;
         /* correct the S[i] term, when needed. */
         if(stopA[i] > i) then do;
            do j = (i+1) to stopA[i];
               S[i] = S[i] + exp(bZ[j]);
            end;
         end;
         jl = jl + vstatusA[i] * (bZ[i] - log(S[i]));
      end;
      model general(jl);
   run;
   ods select all;
```

No output tables were produced because this PROC MCMC run produces identical posterior samples as does the previous example.

Because the JOINTMODEL option is specified here, you do not need to specify myelomaM as the input data set. An empty data set a is used to speed up the procedure run.

Multiple ARRAY statements allocate array symbols that are used to store the parameters (beta), the response and the covariates (data, timeA, vstatusA, and stopA), and the work space (bZ and S). The data, timeA, vstatusA, and stopA arrays are declared with the /NOSYMBOLS option. This option enables PROC MCMC to dynamically resize these arrays to match the dimensions of the input data set. See the section "READ_ARRAY Function" on page 6249. The bZ and S arrays store the regression term and the risk set term for every observation.

The BEGINCNST and ENDCNST statements enclose programming statements that read the data set variables into these arrays. The rest of the programming statements construct the log likelihood for the entire data set.

The CALL MULT function calculates the regression term in the model and stores the result in the array bZ. In the first DO loop, you sum the risk set term S as if there are no ties in time. This underevaluates some of the S elements. For observations that have a tied time, you make the necessary correction to the corresponding S values. The correction takes place in the second DO loop. Any observation that has a tied time also has

a stopA[i] that is different from i. You add the right terms to S and sum up the joint log likelihood jl. The MODEL statement specifies that the log likelihood takes on the value of jl.

To see that you get identical results from these two approaches, use PROC COMPARE to compare the posterior samples from two runs:

```
proc compare data=outi compare=outa;
   ods select comparesummary;
   var beta1-beta9;
run;
```

The output is not shown here.

Generally, the JOINTMODEL option can be slightly faster than using the default setup. The savings come from avoiding the overhead cost of accessing the data set repeatedly at every iteration. However, the speed gain is not guaranteed because it largely depends on the efficiency of your programs.

PROC PHREG fits the same model, and you get very similar results to PROC MCMC. The following statements fit the model using PROC PHREG and produce Output 80.14.2:

```
proc phreg data=Myeloma;
   ods select PostSumInt;
   model Time*VStatus(0)=LogBUN HGB Platelet Age LogWBC
                         Frac LogPBM Protein Scalc;
   bayes seed=1 nmc=10000 outpost=phout;
run;
```

Output 80.14.2 Summary Statistics for Cox Model with Time Independent Explanatory Variables and Ties in the Survival Time, Using PROC PHREG

Cox Model with Time Independent Covariates

The PHREG Procedure

Bayesian Analysis

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
LogBUN	10000	1.7610	0.6593	0.4107	2.9958
HGB	10000	-0.1279	0.0727	-0.2801	0.00599
Platelet	10000	-0.2179	0.5169	-1.1871	0.8341
Age	10000	-0.0130	0.0199	-0.0519	0.0251
LogWBC	10000	0.3150	0.7451	-1.1783	1.7483
Frac	10000	0.3766	0.4152	-0.4273	1.2021
LogPBM	10000	0.3792	0.4909	-0.5939	1.3241
Protein	10000	0.0102	0.0267	-0.0405	0.0637
SCalc	10000	0.1248	0.1062	-0.0846	0.3322

Example 80.15: Time Dependent Cox Model

(View the complete code for this example (mcmcex15.sas) in the example repository.)

This example uses the same Myeloma data set as in "Example 80.14: Time Independent Cox Model" on page 6451, and illustrates the fitting of a time dependent Cox model. The following statements generate the data set once again:

```
data Myeloma;
   input Time Vstatus LogBUN HGB Platelet Age LogWBC Frac
         LogPBM Protein SCalc;
   label Time='survival time'
         VStatus='0=alive 1=dead';
   datalines;
 1.25  1  2.1775   9.4  1  67  3.6628  1  1.9542  12  10
 1.25  1  1.9395  12.0  1  38  3.9868  1  1.9542  20  18
 2.00  1  1.5185   9.8  1  81  3.8751  1  2.0000   2  15
 2.00  1  1.7482  11.3  0  75  3.8062  1  1.2553   0  12

   ... more lines ...

77.00  0  1.0792  14.0  1  60  3.6812  0  0.9542   0  12
;
```

To model $\mathbf{Z}_i(t_i)$ as a function of the survival time, you can relate time t_i to covariates by using this formula:

$$\boldsymbol{\beta}'\mathbf{Z}_j(t_i) = (\beta_1 + \beta_2 t_i)\text{logbun} + (\beta_3 + \beta_4 t_i)\text{hgb} + (\beta_5 + \beta_6 t_i)\text{platelet}$$

For illustrational purposes, only three explanatory variables, LOGBUN, HBG, and PLATELET, are used in this example.

Since $\mathbf{Z}_j(t_i)$ depends on t_i, every term in the summation of $\sum_{l \in \mathcal{R}_i} \exp(\boldsymbol{\beta}'\mathbf{Z}_l(t_i))$ is a product of the current time t_i and all observations that are in the risk set. You can use the JOINTMODEL option, as in the last example, or you can modify the input data set such that every row contains not only the current observation but also all observations that are in the corresponding risk set. When you construct the log likelihood for each observation, you have all the relevant data at your disposal.

The following statements illustrate how you can create a new data set with different risk sets at different rows:

```
title 'Cox Model with Time Dependent Covariates';
proc sort data = Myeloma;
   by descending time;
run;

data _null_;
   set Myeloma nobs=_n;
   call symputx('N', _n);
   stop;
run;

ods select none;
proc freq data=myeloma;
   tables time / out=freqs;
```

```
run;
ods select all;

proc sort data = freqs;
   by descending time;
run;

data myelomaM;
   set myeloma;
   ind = _N_;
run;

data myelomaM;
   merge myelomaM freqs(drop=percent); by descending time;
   retain stop;
   if first.time then do;
      stop = _n_ + count - 1;
   end;
run;

%macro array(list);
   %global mcmcarray;
   %let mcmcarray = ;
   %do i = 1 %to 32000;
      %let v = %scan(&list, &i, %str( ));
      %if %nrbquote(&v) ne %then %do;
         array _&v[&n];
         %let mcmcarray = &mcmcarray array _&v[&n] _&v.1 - _&v.&n%str(;);
         do i = 1 to stop;
            set myelomaM(keep=&v) point=i;
            _&v[i] = &v;
         end;
      %end;
      %else %let i = 32001;
   %end;
%mend;

data z;
   set myelomaM;
   %array(logbun hgb platelet);
   drop vstatus logbun hgb platelet count stop;
run;

data myelomaM;
   merge myelomaM z; by descending time;
run;
```

The data set MyelomaM contains 65 observations and 209 variables. For each observation, you see added variables stop, _logbun1 through _logbun65, _hgb1 through _hgb65, and _platelet1 through _platelet65. The variable stop indicates the number of observations that are in the risk set of the current observation. The rest are transposed values of model covariates of the entire data set. The data set contains a number of missing values. This is due to the fact that only the relevant observations are kept, such as _logbun*1* to _logbun*stop*. The rest of the cells are filled in with missing values. For example, the first observation has a unique survival time of 92 and stop is 1, making it a risk set of itself. You see nonmissing values only in

_logbun1, _hgb1, and _platelet1.

The following statements fit the Cox model by using PROC MCMC:

```
proc mcmc data=myelomaM outpost=outi nmc=50000 ntu=3000 seed=17
          missing=ac;
   ods select PostSumInt;
   array beta[6];
   &mcmcarray
   parms (beta:) 0;
   prior beta: ~ normal(0, prec=1e-6);

   b = (beta1 + beta2 * time) * logbun +
       (beta3 + beta4 * time) * hgb +
       (beta5 + beta6 * time) * platelet;
   S = 0;
   do i = 1 to stop;
      S = S + exp( (beta1 + beta2 * time) * _logbun[i] +
                   (beta3 + beta4 * time) * _hgb[i] +
                   (beta5 + beta6 * time) * _platelet[i]);
   end;
   loglike = vstatus * (b - log(S));

   model general(loglike);
run;
```

Note that the option MISSING= is set to AC. This is due to missing cells in the input data set. You must use this option so that PROC MCMC retains observations that contain missing values.

The macro variable &mcmcarray is defined in the earlier part in this example. You can use a **%put** statement to print its value:

```
%put &mcmcarray;
```

This statement prints the following:

```
array _logbun[65] _logbun1 - _logbun65; array _hgb[65] _hgb1 - _hgb65; array
_platelet[65] _platelet1 - _platelet65;
```

The macro uses the ARRAY statement to allocate three arrays, each of which links their corresponding data set variables. This makes it easier to reference these data set variables in the program. The PARMS statement puts all the parameters in the same block. The PRIOR statement gives them normal priors with large variance. The symbol b is the regression term, and S is cumulatively added from 1 to stop for each observation in the DO loop. The symbol loglike completes the construction of log likelihood for each observation and the MODEL statement completes the model specification.

Posterior summary and interval statistics are shown in Output 80.15.1.

Output 80.15.1 Summary Statistics on Cox Model with Time Dependent Explanatory Variables and Ties in the Survival Time, Using PROC MCMC

Cox Model with Time Dependent Covariates

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta1	50000	3.2397	0.8226	1.6664	4.8752
beta2	50000	-0.1411	0.0471	-0.2294	-0.0458
beta3	50000	-0.0369	0.1017	-0.2272	0.1685
beta4	50000	-0.00409	0.00360	-0.0112	0.00264
beta5	50000	0.3548	0.7359	-1.0394	1.8100
beta6	50000	-0.0417	0.0359	-0.1122	0.0269

You can also use the option JOINTMODEL to get the same inference and avoid transposing the data for every observation:

```
proc mcmc data=myelomaM outpost=outa nmc=50000 ntu=3000 seed=17 jointmodel;
   ods select none;
   array beta[6];       array timeA[&n];       array vstatusA[&n];
   array logbunA[&n];   array hgbA[&n];        array plateletA[&n];
   array stopA[&n];     array bZ[&n];          array S[&n];

   begincnst;
      timeA[ind]=time;          vstatusA[ind]=vstatus;
      logbunA[ind]=logbun;      hgbA[ind]=hgb;
      plateletA[ind]=platelet;  stopA[ind]=stop;
   endcnst;

   parms (beta:) 0;
   prior beta: ~ normal(0, prec=1e-6);

   jl = 0;
   do i = 1 to &n;
      v1 = beta1 + beta2 * timeA[i];
      v2 = beta3 + beta4 * timeA[i];
      v3 = beta5 + beta6 * timeA[i];
      bZ[i] = v1 * logbunA[i] + v2 * hgbA[i] + v3 * plateletA[i];

      /* sum over risk set without considering ties in time. */
      S[i] = exp(bZ[i]);
      if (i > 1) then do;
         do j = 1 to (i-1);
            b1 = v1 * logbunA[j] + v2 * hgbA[j] + v3 * plateletA[j];
            S[i] = S[i] + exp(b1);
         end;
      end;
   end;

   /* make correction to the risk set due to ties in time. */
```

```
   do i = 1 to &n;
      if(stopA[i] > i) then do;
         v1 = beta1 + beta2 * timeA[i];
         v2 = beta3 + beta4 * timeA[i];
         v3 = beta5 + beta6 * timeA[i];
         do j = (i+1) to stopA[i];
            b1 = v1 * logbunA[j] + v2 * hgbA[j] + v3 * plateletA[j];
            S[i] = S[i] + exp(b1);
         end;
      end;
      jl = jl + vstatusA[i] * (bZ[i] - log(S[i]));
   end;
   model general(jl);
run;
```

The multiple ARRAY statements allocate array symbols that are used to store the parameters (beta), the response (timeA), the covariates (vstatusA, logbunA, hgbA, plateletA, and stopA), and work space (bZ and S). The bZ and S arrays store the regression term and the risk set term for every observation. Programming statements in the BEGINCNST and ENDCNST statements input the response and covariates from the data set to the arrays.

Using the same technique shown in the example "Example 80.14: Time Independent Cox Model" on page 6451, the next DO loop calculates the regression term and corresponding S for every observation, pretending that there are no ties in time. This means that the risk set for observation *i* involves only observation 1 to *i*. The correction terms are added to the corresponding S[i] in the second DO loop, conditional on whether the stop variable is greater than the observation count itself. The symbol jl cumulatively adds the log likelihood for the entire data set, and the MODEL statement specifies the joint log-likelihood function.

The following statements run PROC COMPARE and show that the output data set outa contains identical posterior samples as outi:

```
proc compare data=outi compare=outa;
   ods select comparesummary;
   var beta1-beta6;
run;
```

The results are not shown here.

The following statements use PROC PHREG to fit the same time dependent Cox model:

```
proc phreg data=Myeloma;
   ods select PostSumInt;
   model Time*VStatus(0)=LogBUN z2 hgb z3 platelet z4;
   z2 = Time*logbun;
   z3 = Time*hgb;
   z4 = Time*platelet;
   bayes seed=1 nmc=10000 outpost=phout;
run;
```

Coding is simpler than PROC MCMC. See Output 80.15.2 for posterior summary and interval statistics:

Output 80.15.2 Summary Statistics on Cox Model with Time Dependent Explanatory Variables and Ties in the Survival Time, Using PROC PHREG

Cox Model with Time Dependent Covariates

The PHREG Procedure

Bayesian Analysis

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
LogBUN	10000	3.2171	0.8154	1.6453	4.8061
z2	10000	-0.1387	0.0470	-0.2308	-0.0478
HGB	10000	-0.0390	0.1001	-0.2407	0.1539
z3	10000	-0.00403	0.00358	-0.0112	0.00279
Platelet	10000	0.3656	0.7689	-1.1521	1.8144
z4	10000	-0.0412	0.0373	-0.1148	0.0312

Example 80.16: Piecewise Exponential Frailty Model

(View the complete code for this example (mcmcex16.sas) in the example repository.)

This example illustrates how to fit a piecewise exponential frailty model using PROC MCMC. Part of the notation and presentation in this example follows Clayton (1991) and the Luek example in Spiegelhalter et al. (1996a).

Generally speaking, the proportional hazards model assumes the hazard function,

$$\lambda_i(t|\mathbf{z}_i) = \lambda_0(t)\exp\left\{\boldsymbol{\beta}'\mathbf{z}_i\right\}$$

where $i = 1, \ldots, n$ indexes subject, $\lambda_0(t)$ is the baseline hazard function, and $\mathbf{z}_i$ are the covariates for subject i. If you define $N_i(t)$ to be the number of observed failures of the ith subject up to time t, then the hazard function for the ith subject can be seen as a special case of a *multiplicative intensity model* (Clayton 1991). The intensity process for $N_i(t)$ becomes

$$I_i(t) = Y_i(t)\lambda_0(t)\exp(\boldsymbol{\beta}'\mathbf{z}_i)$$

where $Y_i(t)$ indicates observation of the subject at time t (taking the value of 1 if the subject is observed and 0 otherwise). Under *noninformative censoring*, the corresponding likelihood is proportional to

$$\prod_{i=1}^{n}\left[\prod_{t\geq 0} I_i(t)\right]^{dN_i(t)} \exp\left[-\int_{t\geq 0} I_i(t)dt\right]$$

where $dN_i(t)$ is the increment of $N_i(t)$ over the small time interval $[t, t+dt)$: it takes a value of 1 if the subject i fails in the time interval, 0 otherwise. This is a Poisson kernel with the random variable being the increments of dN_i and the means $I_i(t)dt$

$$dN_i(t) \sim \text{Poisson}(I_i(t)dt)$$

where

$$I_i(t)dt = Y_i(t)\exp(\boldsymbol{\beta}'\mathbf{z})d\Lambda_0(t)$$

and

$$\Lambda_0(t) = \int_0^t \lambda_0(u)du.$$

The integral is the increment in the integrated baseline hazard function that occurs during the time interval $[t, t + dt)$.

This formulation provides an alternative way to fit a piecewise exponential model. You partition the time axis to a few intervals, where each interval has its own hazard rate, $\Lambda_0(t)$. You count the $Y_i(t)$ and $dN_i(t)$ in each interval, and fit a Poisson model to each count.

The following DATA step creates the data set Blind (Lin 1994) that represents 197 diabetic patients who have a high risk of experiencing blindness in both eyes as defined by DRS criteria:

```
title 'Piecewise Exponential Model';
data Blind;
   input ID Time Status DiabeticType Treatment @@;
   datalines;
   5 46.23 0 1 1    5 46.23 0 1 0   14 42.50 0 0 1   14 31.30 1 0 0
  16 42.27 0 0 1   16 42.27 0 0 0   25 20.60 0 0 1   25 20.60 0 0 0
  29 38.77 0 0 1   29  0.30 1 0 0   46 65.23 0 0 1   46 54.27 1 0 0
  49 63.50 0 0 1   49 10.80 1 0 0   56 23.17 0 0 1   56 23.17 0 0 0

   ... more lines ...

1705  8.00 0 0 1 1705  8.00 0 0 0 1717 51.60 0 1 1 1717 42.33 1 1 0
1727 49.97 0 1 1 1727  2.90 1 1 0 1746 45.90 0 0 1 1746  1.43 1 0 0
1749 41.93 0 1 1 1749 41.93 0 1 0
;
```

One eye of each patient is treated with laser photocoagulation. The hypothesis of interest is whether the laser treatment delays the occurrence of blindness. The following variables are included in Blind:

- ID, patient's identification
- Time, failure time
- Status, event indicator (0=censored and 1=uncensored)
- Treatment, treatment received (1=laser photocoagulation and 0=otherwise)
- DiabeticType, type of diabetes (0=juvenile onset with age of onset at 20 or under, and 1= adult onset with age of onset over 20)

For illustrational purposes, a piecewise exponential model that ignores the patient-level frailties is first fit to the entire data set. The formulation of the Poisson counting process makes it straightforward to add the frailty terms, as it is demonstrated later.

The following statements create a partition (of length 8) along the time axis, with $s_0 < s_1 < s_2 < \cdots < s_J$, with $s_0 = 0.1 < y_i$ and $s_J = 80 > y_i$ for all i. The time intervals are stored in the Partition data set:

```
data partition;
   input int_1-int_9;
   datalines;
   0.1  6.545  13.95  26.47  38.8  45.88  54.35  62 80
;
```

To obtain reasonable estimates, placing an equal number of observations in each interval is recommended. You can find the partition points by calculating the percentile statistics of the time variable (for example, by using the UNIVARIATE procedure).

The following regression model and prior distributions are used in the analysis:

$$\begin{aligned} \boldsymbol{\beta}'\mathbf{z}_i &= \beta_1\text{treatment} + \beta_2\text{diabetictype} + \beta_3\text{treatment * diabetictype} \\ \beta_1, \beta_2, \beta_3 &\sim \text{normal}(0, \text{var } = 1e6) \\ \lambda_j &\sim \text{gamma}(\text{shape} = 0.01, \text{iscale} = 0.01) \quad \text{for} \quad j = 1, \ldots, 8 \end{aligned}$$

The following statements calculate $Y_i(t)$ for each observation i, at every time point t in the Partition data set. The statements also find the observed failure time interval, $dN_i(t)$, for each observation:

```
%let n = 8;
data _a;
   set blind;
   if _n_ eq 1 then set partition;
   array int[*] int_:;
   array Y[&n];
   array dN[&n];
   do k = 1 to (dim(int)-1);
      Y[k] = (time - int[k] + 0.001 >= 0);
      dN[k] = Y[k] * ( int[k+1] - time - 0.001 >= 0) * status;
   end;
   output;
   drop int_: k;
run;
```

The DATA step reads in the Blind data set. At the first observation, it also reads in the Partition data set. The first ARRAY statement creates the int array and name the elements int_:. Because the names match the variable names in the Partition data set, all values of the int_: variables (there is only one observation) in the Partition data set are therefore stored in the int array. The next two ARRAY statements create arrays Y and dN, each with length 8. They store values of $Y_i(t)$ and $dN_i(t)$, resulting from each failure time in the Blind data set.

The following statements print the first 10 observations of the constructed data set _a and display them in Output 80.16.1:

```
proc print data=_a(obs=10);
run;
```

Output 80.16.1 First 10 Observations of the Data Set _a

Piecewise Exponential Model

Obs	ID	Time	Status	DiabeticType	Treatment	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	dN1	dN2	dN3	dN4	dN5	dN6	dN7	dN8
1	5	46.23	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	5	46.23	0	1	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
3	14	42.50	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
4	14	31.30	1	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0
5	16	42.27	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
6	16	42.27	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
7	25	20.60	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	25	20.60	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
9	29	38.77	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
10	29	0.30	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

The first subject in _a experienced blindness in the left eye at time 46.23, and the time falls in the sixth interval as defined in the Partition data set. Therefore, Y1 through Y6 all take a value of 1, and Y7 and Y8 are 0. The variable dN# takes on a value of 1 if the subject is observed to go blind in that interval. Since the first observation is censored (status == 1), the actual failure time is unknown. Hence all dN# are 0. The first observed failure time occurs in observation number 4 (the right eye of the second subject), where the time variable takes a value of 31.30, Y1 through Y4 are 1, and dN4 is 1.

Note that each observation in the _a data set has 8 Y and 8 dN, meaning that you would need eight MODEL statements in a PROC MCMC call, each for a Poisson likelihood. Alternatively, you can expand _a, put one Y and one dN in every observation, and fit the data using a single MODEL statement in PROC MCMC. The following statements expand the data set _a and save the results in the data set _b:

```
data _b;
   set _a;
   array y[*] y:;
   array dn[*] dn:;
   do i = 1 to (dim(y));
      y_val       = y[i];
      dn_val      = dn[i];
      int_index   = i;
      output;
   end;
   keep y_: dn_: diabetictype treatment int_index id;
run;

data _b;
   set _b;
   rename y_val=Y dn_val=dN;
run;
```

You can use the following PROC PRINT statements to see the first few observations in _b:

```
proc print data=_b(obs=10);
run;
```

Output 80.16.2 First 20 Observations of the Data Set _b

Obs	ID	DiabeticType	Treatment	Y	dN	int_index
1	5	1	1	1	0	1
2	5	1	1	1	0	2
3	5	1	1	1	0	3
4	5	1	1	1	0	4
5	5	1	1	1	0	5
6	5	1	1	1	0	6
7	5	1	1	0	0	7
8	5	1	1	0	0	8
9	5	1	0	1	0	1
10	5	1	0	1	0	2

The data set _b now contains 3,152 observations (see Output 80.16.2 for the first few observations). The Time and Status variables are no longer needed; hence they are discarded from the data set. The int_index variable is an index variable that indicates interval membership of each observation.

Because the variable Y does not contribute to the likelihood calculation when it takes a value of 0 (it amounts to a Poisson likelihood that has a mean and response variable that are both 0), you can remove these observations. This speeds up the calculation in PROC MCMC:

```
data inputdata;
   set _b;
   if Y > 0;
run;
```

The data set Inputdata has 1,775 observations, as opposed to 3,152 observations in _b. The following statements fit a piecewise exponential model in PROC MCMC:

```
proc mcmc data=inputdata nmc=10000 outpost=postout seed=12351
     maxtune=5;
   ods select PostSumInt ESS;
   parms beta1-beta3 0;
   prior beta: ~ normal(0, var = 1e6);
   random lambda ~ gamma(0.01, iscale = 0.01) subject=int_index;
   bZ = beta1*treatment + beta2*diabetictype + beta3*treatment*diabetictype;
   idt = exp(bz) * lambda;
   model dN ~ poisson(idt);
run;
```

The PARMS statement declares three regression parameters, beta1–beta3. The PRIOR statement specifies a noninformative normal prior on the regression coefficients. The RANDOM statement specifies the random effect, lambda, its prior distribution, and interval membership which is indexed by the data set variable int_index.

The symbol bZ calculates the regression mean, and the symbol idt is the mean of the Poisson likelihood. It corresponds to the equation

$$I_i(t)dt = Y_i(t)\exp(\boldsymbol{\beta}'\mathbf{z})d\Lambda_0(t)$$

Note that the $Y_i(t)$ term is omitted in the assignment statement because Y takes only the value of 1 in the input data set.

Output 80.16.3 displays posterior estimates of the three regression parameters.

Output 80.16.3 Posterior Summary Statistics

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta1	10000	-0.4174	0.2129	-0.8121	0.0203
beta2	10000	0.3138	0.1956	-0.0885	0.6958
beta3	10000	-0.7899	0.3308	-1.4300	-0.1046

To understand the results, you can create a 2×2 table (Table 80.55) and plug in the posterior mean estimates to the regression model. A –0.41 estimate for subjects who received laser treatment and had juvenile diabetes suggests that the laser treatment is effective in delaying blindness. And the effect is much more pronounced (–0.80) for adult subjects who have diabetes and received treatment.

Table 80.55 Estimates of Regression Effects in the Survival Model

$\hat{\boldsymbol{\beta}}'\mathbf{Z}$		Diabetic Type	
		0	1
Treatment	0	0	0.32
	1	–0.41	–0.80

You can also use the macro %CATER (“Caterpillar Plot” on page 6341) to draw a caterpillar plot to visualize the eight hazards in the model:

```
%cater(data=postout, var=lambda_:);
```

Output 80.16.4 Caterpillar Plot of the Hazards in the Piecewise Exponential Model

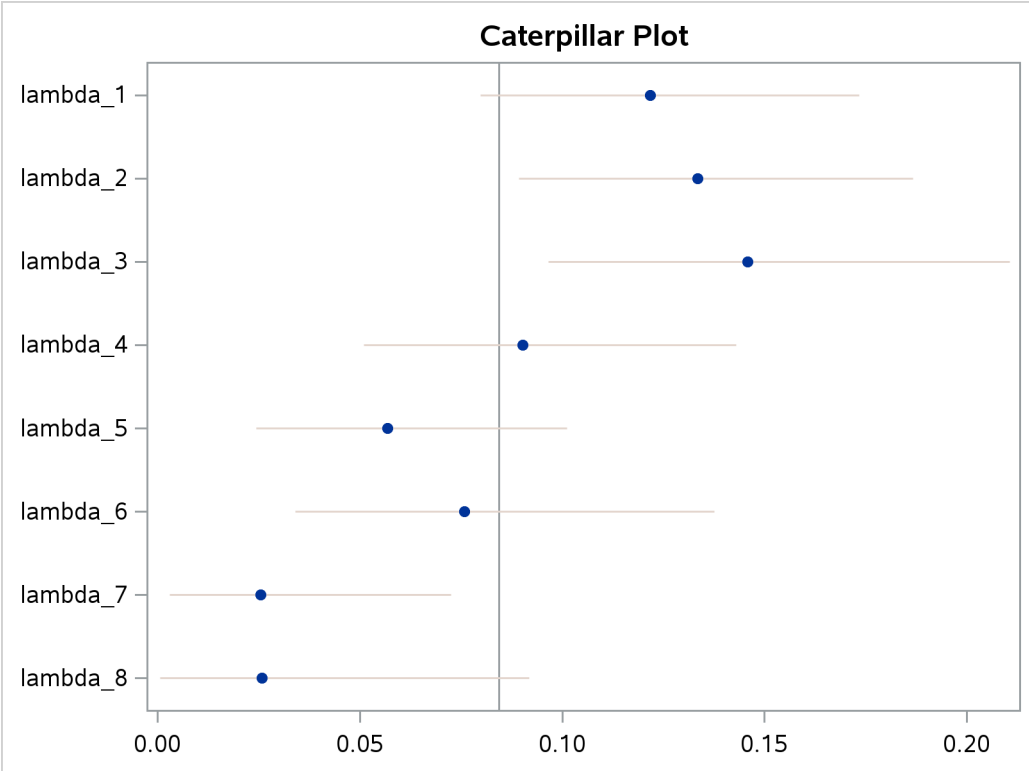

The fitted hazards show a nonconstant underlying hazard function (read along the y-axis as lambda_# are hazards along the time-axis) in the model.

Now suppose you want to include patient-level information and fit a frailty model to the blind data set, where the random effect enters the model through the regression term, where the subject is indexed by the variable ID in the data.

$$
\begin{aligned}
\boldsymbol{\beta}'\mathbf{z}_i &= \beta_1\text{treatment} + \beta_2\text{diabetictype} + \beta_3\text{treatment} * \text{diabetictype} + u_{id} \\
u_{id} &\sim \text{normal}(0, \text{var} = \sigma^2) \\
\sigma^2 &\sim \text{igamma}(\text{shape} = 0.01, \text{scale} = 0.01)
\end{aligned}
$$

where id indexes patient.

The actual coding in PROC MCMC of a piecewise exponential frailty model is rather straightforward:

```
ods select none;
proc mcmc data=inputdata nmc=10000 outpost=postout seed=12351
     stats=summary diag=none;
   parms beta1-beta3 0 s2;
   prior beta: ~ normal(0, var = 1e6);
   prior s2 ~ igamma(0.01, scale=0.01);
   random lambda ~ gamma(0.01, iscale = 0.01) subject=int_index;
   random u ~ normal(0, var=s2) subject=id;
   bZ = beta1*treatment + beta2*diabetictype + beta3*treatment*diabetictype + u;
   idt = exp(bZ) * lambda;
```

```
   model dN ~ poisson(idt);
run;
```

A second RANDOM statement defines a subject-level random effect u, and the random-effects parameters enter the model in the term for the regression mean, bZ. An additional model parameter, s2, the variance of the random-effects parameters, is needed for the model. The results are not shown here.

Example 80.17: Normal Regression with Interval Censoring

(View the complete code for this example (mcmcex17.sas) in the example repository.)

You can use PROC MCMC to fit failure time data that can be right-, left-, or interval-censored. To illustrate, a normal regression model is used in this example.

You can use either of two approaches to fit interval-censored data. One is to specify the marginal model, and the other is to treat the censored data as missing values.

Assume that you have a simple regression model with no covariates,

$$\mathbf{y} = \mu + \sigma\boldsymbol{\epsilon}$$

where $\mathbf{y}$ is a vector of response values (the failure times), μ is the grand mean, σ is an unknown scale parameter, and $\boldsymbol{\epsilon}$ are errors from the standard normal distribution. Instead of observing y_i directly, you observe only a truncated value t_i. If the true y_i occurs after the censored time t_i, the data are called *right-censored*. If y_i occurs before the censored time, the data are called *left-censored*. A failure time y_i can be censored at both ends, and these data are called *interval-censored*. The likelihood for y_i is

$$p(y_i|\mu) = \begin{cases} \phi(y_i|\mu,\sigma) & \text{if } y_i \text{ is uncensored} \\ S(t_{l,i}|\mu) & \text{if } y_i \text{ is right-censored by } t_{l,i} \\ 1 - S(t_{r,i}|\mu) & \text{if } y_i \text{ is left-censored by } t_{r,i} \\ S(t_{l,i}|\mu) - S(t_{r,i}|\mu) & \text{if } y_i \text{ is interval-censored by } t_{l,i} \text{ and } t_{r,i} \end{cases}$$

where $S(\cdot)$ is the survival function and $S(t) = \Pr(T > t)$. When a datum is uncensored, you use a normal likelihood. When a datum is censored, you use the cumulative distribution to account for the likelihood.

Gentleman and Geyer (1994) uses the following data on cosmetic deterioration for early breast cancer patients who are treated with radiotherapy:

```
title 'Normal Regression with Interval Censoring';
data cosmetic;
   t = .;
   label tl = 'Time to Event (Months)';
   input tl tr @@;
   datalines;
45  .   6 10   .  7  46  .  46  .   7 16  17  .   7 14
37 44   .  8   4 11  15  .  11 15  22  .  46  .  46  .
25 37  46  .  26 40  46  .  27 34  36 44  46  .  36 48
37  .  40  .  17 25  46  .  11 18  38  .   5 12  37  .
 .  5  18  .  24  .  36  .   5 11  19 35  17 25  24  .
32  .  33  .  19 26  37  .  34  .  36  .
;
```

The data consist of time interval endpoints (in months). Nonmissing equal endpoints (tl = tr) indicate noncensoring; a nonmissing lower endpoint (tl $\neq$.) and a missing upper endpoint (tr = .) indicate right-censoring; a missing lower endpoint (tl = .) and a nonmissing upper endpoint (tr $\neq$.) indicate left-censoring; and nonmissing unequal endpoints (tl $\neq$ tr) indicate interval censoring. In this data set, all observations are censored (all t are missing).

With this data set, you can consider using proper but diffuse priors on both μ and σ. For example,

$$\begin{aligned} \mu &\sim \text{normal}(0, \text{sd} = 1000) \\ \sigma &\sim \text{gamma}(0.001, \text{iscale} = 0.001) \end{aligned}$$

The following SAS statements fit an interval-censoring model by using its marginal distribution and generate Output 80.17.1:

```
proc mcmc data=cosmetic outpost=postout seed=1 nmc=20000 missing=AC;
   ods select PostSumInt;
   parms mu 60 sigma 50;

   prior mu ~ normal(0, sd=1000);
   prior sigma ~ gamma(shape=0.001,iscale=0.001);

   if (tl^=. and tr^=. and tl=tr) then
      llike = logpdf('normal',tr,mu,sigma);
   else if (tl^=. and tr=.) then
      llike = logsdf('normal',tl,mu,sigma);
   else if (tl=. and tr^=.) then
      llike = logcdf('normal',tr,mu,sigma);
   else
      llike = log(sdf('normal',tl,mu,sigma) -
         sdf('normal',tr,mu,sigma));

   model general(llike);
run;
```

Because there are missing cells in the input data, you want to use the MISSING=AC option so that PROC MCMC does not delete any observations that contain missing values. The IF-ELSE statements distinguish different censoring cases for y_i according to the likelihood. The SAS functions LOGCDF, LOGSDF, LOGPDF, and SDF are useful here. The MODEL statement assigns llike as the log likelihood to the response. The Markov chain appears to have converged in this example (evidence not shown here), and the posterior estimates are shown in Output 80.17.1.

Output 80.17.1 Interval Censoring

Normal Regression with Interval Censoring

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
mu	20000	41.7807	5.7882	31.3604	53.6115
sigma	20000	29.1122	6.0503	19.4041	41.6742

The marginal model approach is more efficient because the censored observations are integrated out. However, you might not always have the cumulative distributions readily available in all scenarios. One general and alternative approach is to treat all censored variables as latent variables (or missing data). You fit the same model by imputing the would-be values in the censored data and estimating the model parameters. The censoring information, which specifies the restricted range of the unobserved variables, is specified in the CLOWER= and CUPPER= options.

The following SAS statements fit censored data by using the missing data approach:

```
proc mcmc data=cosmetic outpost=postout seed=117207154
   nmc=20000 missing=ACMODELY;
   ods select none;
   parms mu 60 sigma 50;

   prior mu ~ normal(0, sd=1000);
   prior sigma ~ gamma(shape=0.001,iscale=0.001);

   model t ~ normal(mu, sd=sigma, clower=tl, cupper=tr);
run;
```

By default, PROC MCMC discards observations that have missing values in covariate variables (a covariate is a data set variable that appears in the program but not to the left of the tilde in a MODEL statement). To keep observations that have missing tl or tr values, specify the MISSING=ACMODELY option, which keeps all observations and models the missing response variable. This approach produces estimates that are equivalent to those from the marginal model approach. The results are not shown here.

Example 80.18: Constrained Analysis

(View the complete code for this example (mcmcex18.sas) in the example repository.)

Conjoint analysis uses regression techniques to model consumer preferences and to estimate consumer utility functions. A problem with conventional conjoint analysis is that sometimes your estimated utilities do not make sense. Your results might suggest, for example, that the consumers would prefer to spend more on a product than to spend less. With PROC MCMC, you can specify constraints on the part-worth utilities (parameter estimates). Suppose that the consumer product being analyzed is an off-road motorcycle. The relevant attributes are how large each motorcycle is (less than 300cc, 301–550cc, and more than 551cc), how much it costs (less than $5000, $5001–$6000, $6001–$7000, and more than $7000), whether or not it has an electric starter, whether or not the engine is counter-balanced, and whether the bike is from Japan or Europe. The preference variable is a ranking of the bikes. You could perform an ordinary conjoint analysis with PROC TRANSREG (see Chapter 126, "The TRANSREG Procedure") as follows:

```
options validvarname=any;
proc format;
   value sizef  1 = '< 300cc' 2 = '300-550cc' 3 = '> 551cc';
   value pricef 1 = '< $5000' 2 = '$5000 - $6000'
                3 = '$6001 - $7000' 4 = '> $7000';
   value startf 1 = 'Electric Start' 2 = 'Kick Start';
   value balf   1 = 'Counter Balanced' 2 = 'Unbalanced';
   value orif   1 = 'Japanese' 2 = 'European';
run;
```

```
data bikes;
   input Size Price Start Balance Origin Rank @@;
   format size sizef. price pricef. start startf.
          balance balf. origin orif.;
   datalines;
2 1 2 1 2  3  1 4 2 2 2  7  1 2 1 1 2  6
3 3 1 1 2  1  1 3 2 1 1  5  3 4 2 2 2 12
2 3 2 2 1  9  1 1 1 2 1  8  2 2 1 2 2 10
2 4 1 1 1  4  3 1 1 2 1 11  3 2 2 1 1  2
;

title 'Ordinary Conjoint Analysis by PROC TRANSREG';
proc transreg data=bikes utilities cprefix=0 lprefix=0;
   ods select Utilities;
   model identity(rank / reflect) =
         class(size price start balance origin / zero=sum);
   output out=coded(drop=intercept) replace;
run;
```

The DATA step reads the experimental design and dependent variable Rank and assigns formats to label the factor levels. PROC TRANSREG is run specifying UTILITIES, which requests a conjoint analysis. The rank variable is reflected around its mean ($1 \rightarrow 12$, $2 \rightarrow 11$, $\ldots$, $12 \rightarrow 1$) so that in the analysis, larger part-worth utilities correspond to higher preference. The OUT=CODED data set contains the reflected ranks and a binary coding of the factors that can be used in other analyses. See Kuhfeld (2010) for more information about conjoint analysis and coding with PROC TRANSREG.

The Utilities table from the conjoint analysis is shown in Output 80.18.1. Notice the part-worth utilities for price. The part-worth utility for < \$5000 is 0.25. As price increases to the \$5000–\$6000 range, utility decreases to –0.5. Then as price increases to the \$6001–\$7000 range, part-worth utility *increases* to 0.5. Finally, for the most expensive bikes, utility decreases again to –0.25. In cases like this, you might want to impose constraints on the solution so that the part-worth utility for price never increases as prices go up.

Output 80.18.1 Ordinary Conjoint Analysis by PROC TRANSREG

Ordinary Conjoint Analysis by PROC TRANSREG

The TRANSREG Procedure

Utilities Table Based on the Usual Degrees of Freedom

Label	Utility	Standard Error	Importance (% Utility Range)	Variable
Intercept	6.5000	0.95743		Intercept
< 300cc	-0.0000	1.35401	0.000	Class.< 300cc
300-550cc	-0.0000	1.35401		Class.300-550cc
> 551cc	0.0000	1.35401		Class.> 551cc
< $5000	0.2500	1.75891	13.333	Class.< $5000
$5000 - $6000	-0.5000	1.75891		Class.$5000 - $6000
$6001 - $7000	0.5000	1.75891		Class.$6001 - $7000
> $7000	-0.2500	1.75891		Class.> $7000
Electric Start	-0.1250	1.01550	3.333	Class.Electric Start
Kick Start	0.1250	1.01550		Class.Kick Start
Counter Balanced	3.0000	1.01550	80.000	Class.Counter Balanced
Unbalanced	-3.0000	1.01550		Class.Unbalanced
Japanese	-0.1250	1.01550	3.333	Class.Japanese
European	0.1250	1.01550		Class.European

You could run PROC TRANSREG again, specifying monotonicity constraints on the part-worth utilities for price:

```
title 'Constrained Conjoint Analysis by PROC TRANSREG';
proc transreg data=bikes utilities cprefix=0 lprefix=0;
   ods select ConservUtilities;
   model identity(rank / reflect) =
         monotone(price / tstandard=center)
         class(size start balance origin / zero=sum);
run;
```

The output from this PROC TRANSREG step is shown in Output 80.18.2.

Output 80.18.2 Constrained Conjoint Analysis by PROC TRANSREG

Constrained Conjoint Analysis by PROC TRANSREG

The TRANSREG Procedure

Utilities Table Based on Conservative Degrees of Freedom

Label	Utility	Standard Error	Importance (% Utility Range)	Variable
Intercept	6.5000	0.97658		Intercept
Price	-0.1581	.	7.143	Monotone(Price)
< $5000	0.2500	.		
$5000 - $6000	0.0000	.		
$6001 - $7000	0.0000	.		
> $7000	-0.2500	.		
< 300cc	-0.0000	1.38109	0.000	Class.< 300cc
300-550cc	-0.0000	1.38109		Class.300-550cc
> 551cc	0.0000	1.38109		Class.> 551cc
Electric Start	-0.2083	1.00663	5.952	Class.Electric Start
Kick Start	0.2083	1.00663		Class.Kick Start
Counter Balanced	3.0000	0.97658	85.714	Class.Counter Balanced
Unbalanced	-3.0000	0.97658		Class.Unbalanced
Japanese	-0.0417	1.00663	1.190	Class.Japanese
European	0.0417	1.00663		Class.European

This monotonicity constraint is one of the few constraints on the part-worth utilities that you can specify in PROC TRANSREG. In contrast, PROC MCMC enables you to specify any constraint that can be written in the DATA step language. You can perform the restricted conjoint analysis with PROC MCMC by using the coded factors that were output from PROC TRANSREG. The data set is Coded.

The likelihood is a simple regression model:

$$\text{rank}_i \sim \text{normal}(\mathbf{x}_i'\boldsymbol{\beta}, \sigma)$$

where rank is the response, the covariates are '< 300cc'n, '300-500cc'n, '< $5000'n, '$5000 - $6000'n, '$6001 - $7000'n, 'Electric Start'n, 'Counter Balanced'n, and Japanese. Note that OPTIONS VALIDVARNAME=ANY enables PROC TRANSREG to create names for the coded variables with blanks and special characters. That is why the name-literal notation ('*variable-name*'n) is used for the input data set variables.

Suppose that there are two constraints you want to put on some of the parameters: one is that the parameters for '< $5000'n, '$5000 - $6000'n, and '$6001 - $7000'n decrease in order, and the other is that the parameter for 'Counter Balanced'n is strictly positive. You can consider a truncated multivariate normal prior as follows:

$$(\beta_{\text{'< \$5000'n}}, \beta_{\text{'\$5000 - \$6000'n}}, \beta_{\text{'\$6001 - \$7000'n}}, \beta_{\text{'Counter Balanced'n}}) \sim \text{MVN}(0, \sigma\mathbf{I})$$

with the following set of constraints:

$$\begin{aligned} \beta_{\text{'< \$5000'n}} &> \beta_{\text{'\$5000 - \$6000'n}} > \beta_{\text{'\$6001 - \$7000'n}} > 0 \\ \beta_{\text{'Counter Balanced'n}} &> 0 \end{aligned}$$

The condition that $\beta_{\text{'\$6001 - \$7000'n}} > 0$ reflects an implied constraint that, by definition, 0 is the utility for the highest price range, > $7000, which is the reference level for the binary coded price variable. The following statements fit the desired model:

```
title 'Bayesian Constrained Conjoint Analysis by PROC MCMC';
proc mcmc data=coded outpost=bikesout ntu=3000 nmc=50000
   propcov=quanew seed=448 diag=none;
   ods select PostSumInt;
   array pw[4] pw5000 pw5000_6000 pw6001_7000 pwCounterBalanced;
   array sigma[4,4];
   array mu[4];

   begincnst;
      call identity(sigma);
      call mult(sigma, 100, sigma);
      call zeromatrix(mu);
   endcnst;

   parms intercept pw300cc pw300_550cc pwElectricStart pwJapanese tau 1;
   parms pw5000 0.3 pw5000_6000 0.2 pw6001_7000 0.1 pwCounterBalanced 1;

   beginnodata;
   prior intercept pw300: pwE: pwJ: ~ normal(0, var=100);
   if (pw5000      >= pw5000_6000 & pw5000_6000 >= pw6001_7000 &
       pw6001_7000 >= 0           & pwCounterBalanced > 0) then
       lp = lpdfmvn(pw, mu, sigma);
   else
       lp = .;
   prior pw5000 pw5000_6000 pw6001_7000 pwC: ~ general(lp);
   prior tau  ~ gamma(0.01, iscale=0.01);
   endnodata;

   mean = intercept +
          pw300cc           * '< 300cc'n          +
          pw300_550cc       * '300-550cc'n        +
          pw5000            * '< $5000'n          +
          pw5000_6000       * '$5000 - $6000'n    +
          pw6001_7000       * '$6001 - $7000'n    +
          pwElectricStart   * 'Electric Start'n   +
          pwCounterBalanced * 'Counter Balanced'n +
          pwJapanese        * Japanese;
   model rank ~ normal(mean, prec=tau);
run;
```

The two ARRAY statements allocate a 4×4 dimensional array for the prior covariance and an array of size 4 for the prior means. In the BEGINCNST and ENDCNST statements, the CALL IDENTITY function sets sigma to be an identity matrix; the CALL MULT function sets sigma's diagonal elements to be 100 (the diagonal variance terms); and the CALL ZEROMATRIX function sets mu to be a vector of zeros (the prior means). For matrix functions in PROC MCMC, see the section "Matrix Functions in PROC MCMC" on page 6313.

There are two PARMS statements, with each of them naming a block of parameters. The first PARMS statement blocks the following: the intercept, the two size parameters, the one start-type parameter, the one origin parameter, and the precision. The second PARMS statement blocks the three price parameters and the

one balance parameter, parameters that have the constraint multivariate normal prior. The second PARMS statement also specifies initial values for the parameter estimates. The initial values reflect the constraints on these parameters. The initial part-worth utilities all decrease from 0.3 to 0.2 to 0.1 to 0.0 (for the implicit reference level) as the prices increase. Also, the initial part-worth utility for the counter-balanced engine is set to a positive value, 1.

In the PRIOR statements, regression coefficients without constraints are given an independent normal prior with mean at 0 and variance of 100. The next IF-ELSE construction imposes the constraints. When these constraints are met, pw5000, pw5000_6000, pw6001_7000, pwCounterBalanced are jointly distributed as a multivariate normal prior with mean mu and covariance sigma. Otherwise, the prior is not defined and lp is assigned a missing value. The parameter tau is given a gamma prior, which is a conjugate prior for that parameter.

The model specification is linear. The mean is comprised of an intercept and the sum of terms like pw300cc * '< 300cc'n, which is a parameter times an input data set variable. The MODEL statement specifies that the linear model for rank is normally distributed with mean mean and precision tau.

The MCMC results are shown in Output 80.18.3.

Output 80.18.3 MCMC Results

Bayesian Constrained Conjoint Analysis by PROC MCMC

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
intercept	50000	2.1679	2.5351	-2.8636	7.3002
pw300cc	50000	0.1095	2.4763	-4.7698	5.1516
pw300_550cc	50000	0.0961	2.4206	-4.9501	4.8323
pwElectricStart	50000	-1.0173	2.1034	-5.3289	3.0740
pwJapanese	50000	-0.4418	2.1367	-4.8215	3.9265
tau	50000	0.1134	0.0763	0.00805	0.2639
pw5000	50000	4.1792	2.1701	0.6291	8.4404
pw5000_6000	50000	2.5997	1.6701	0.1724	5.8191
pw6001_7000	50000	1.4686	1.2512	0.000053	3.9253
pwCounterBalanced	50000	5.7915	1.9897	1.4204	9.4765

The estimates of the part-worth utility for the price categories are ordered as expected. This agrees with the intuition that there is a higher preference for a less expensive motor bike when all other things are equal, and that is what you see when you look at the estimated posterior means for the price part-worths. The estimated standard deviations of the price part-worths in this model are of approximately the same order of magnitude as the posterior means. This indicates that the part-worth utilities for this subject are not significantly far from each other, and that this subject's ranking of the options was not significantly influenced by the difference in price.

One advantage of Bayesian analysis is that you can incorporate prior information in the data analysis. Constraints on the parameter space are one possible source of information that you might have before you examine the data. This example shows that it can be accomplished in PROC MCMC.

Example 80.19: Implement a New Sampling Algorithm

(View the complete code for this example (mcmcex19.sas) in the example repository.)

This example illustrates using the UDS statement to implement a new Markov chain sampler. The algorithm demonstrated here is proposed by Holmes and Held (2006), hereafter referred to as HH. They presented a Gibbs sampling algorithm for generating draws from the posterior distribution of the parameters in a probit regression model. The notation follows closely to HH.

The data used here is the remission data set from a PROC LOGISTIC example:

```
title 'Implement a New Sampling Algorithm';
data inputdata;
   input remiss cell smear infil li blast temp;
   ind = _n_;
   cnst = 1;
   label remiss='Complete Remission';
   datalines;
   1  0.8   0.83  0.66  1.9  1.1    0.996

   ... more lines ...

   0  1     0.73  0.73  0.7  0.398  0.986
;
```

The variable remiss is the cancer remission indicator variable with a value of 1 for remission and a value of 0 for nonremission. There are six explanatory variables: cell, smear, infil, li, blast, and temp. These variables are the risk factors thought to be related to cancer remission. The binary regression model is as follows:

$$\text{remiss}_i \sim \text{binary}(p_i)$$

where the covariates are linked to p_i through a probit transformation:

$$\text{probit}(p_i) = \mathbf{x}'\boldsymbol{\beta}$$

$\boldsymbol{\beta}$ are the regression coefficients and $\mathbf{x}'$ the explanatory variables. Suppose you want to use independent normal priors on the regression coefficients:

$$\beta_i \sim \text{normal}(0, \text{var} = 25)$$

Fitting a probit model with PROC MCMC is straightforward. You can use the following statements:

```
proc mcmc data=inputdata nmc=100000 propcov=quanew seed=17
          outpost=mcmcout;
   ods select PostSumInt ess;
   parms beta0-beta6;
   prior beta: ~ normal(0,var=25);
   mu = beta0 + beta1*cell + beta2*smear +
         beta3*infil +  beta4*li + beta5*blast +  beta6*temp;
   p = cdf('normal', mu, 0, 1);
   model remiss ~ bern(p);
run;
```

The expression mu is the regression mean, and the CDF function links mu to the probability of remission p in the binary likelihood.

The summary statistics and effective sample sizes tables are shown in Output 80.19.1. There are high autocorrelations among the posterior samples, and efficiency is relatively low. The correlation time is reduced only after a large amount of thinning.

Output 80.19.1 Random Walk Metropolis

Implement a New Sampling Algorithm

The MCMC Procedure

Posterior Summaries and Intervals

Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta0	100000	-2.0107	3.8405	-9.2214	5.7105
beta1	100000	2.5452	2.8012	-2.8579	8.0920
beta2	100000	-0.8095	3.2102	-7.0811	5.4883
beta3	100000	1.5889	3.5031	-5.3397	8.4183
beta4	100000	2.0270	0.8836	0.3722	3.8051
beta5	100000	-0.2896	0.9572	-2.1911	1.5439
beta6	100000	-3.2557	3.8146	-10.4698	4.5242

Implement a New Sampling Algorithm

The MCMC Procedure

Effective Sample Sizes

Parameter	ESS	Autocorrelation Time	Efficiency
beta0	4525.6	22.0963	0.0453
beta1	4687.0	21.3358	0.0469
beta2	3476.1	28.7677	0.0348
beta3	3941.4	25.3719	0.0394
beta4	3602.6	27.7580	0.0360
beta5	3437.2	29.0931	0.0344
beta6	4290.3	23.3086	0.0429

As an alternative to the random walk Metropolis, you can use the Gibbs algorithm to sample from the posterior distribution. The Gibbs algorithm is described in the section "Gibbs Sampler" on page 157 in Chapter 8, "Introduction to Bayesian Analysis Procedures." While the Gibbs algorithm generally applies to a wide range of statistical models, the actual implementation can be problem-specific. In this example, performing a Gibbs sampler involves introducing a class of auxiliary variables (also known as latent variables). You first reformulate the model by adding a z_i for each observation in the data set:

$$
\begin{aligned}
y_i &= \begin{cases} 1 & \text{if } z_i > 0 \\ 0 & \text{otherwise} \end{cases} \\
z_i &= \mathbf{x}_i' \boldsymbol{\beta} + \epsilon_i \\
\epsilon &\sim \text{normal}(0, 1) \\
\boldsymbol{\beta} &\sim \pi(\boldsymbol{\beta})
\end{aligned}
$$

If $\boldsymbol{\beta}$ has a normal prior, such as $\pi(\boldsymbol{\beta}) = N(\mathbf{b}, \mathbf{v})$, you can work out a closed form solution to the full conditional distribution of $\boldsymbol{\beta}$ given the data and the latent variables z_i. The full conditional distribution is also a multivariate normal, due to the conjugacy of the problem. See the section "Conjugate Priors" on page 151 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The formula is shown here:

$$\begin{aligned} \boldsymbol{\beta}|\mathbf{z}, \mathbf{x} &\sim \text{normal}(\mathbf{B}, \mathbf{V}) \\ \mathbf{B} &= \mathbf{V}((v)^{-1}\mathbf{b} + \mathbf{x}'\mathbf{z}) \\ \mathbf{V} &= (\mathbf{v}^{-1} + \mathbf{x}'\mathbf{x})^{-1} \end{aligned}$$

The advantage of creating the latent variables is that the full conditional distribution of $\mathbf{z}$ is also easy to work with. The distribution is a truncated normal distribution:

$$z_i|\boldsymbol{\beta}, \mathbf{x}_i, y_i \sim \begin{cases} \text{normal}(\mathbf{x}_i\boldsymbol{\beta}, 1)I(z_i > 0) & \text{if } y_i = 1 \\ \text{normal}(\mathbf{x}_i\boldsymbol{\beta}, 1)I(z_i \le 0) & \text{otherwise} \end{cases}$$

You can sample $\boldsymbol{\beta}$ and z iteratively, by drawing $\boldsymbol{\beta}$ given z and vice verse. HH point out that a high degree of correlation could exist between $\boldsymbol{\beta}$ and z, and it makes this iterative way of sampling inefficient. As an improvement, HH proposed an algorithm that samples $\boldsymbol{\beta}$ and $\mathbf{z}$ jointly. At each iteration, you sample z_i from the posterior marginal distribution (this is the distribution that is conditional only on the data and not on any parameters) and then sample $\boldsymbol{\beta}$ from the same posterior full conditional distribution as described previously:

1. Sample z_i from its posterior marginal distribution:

$$\begin{aligned} z_i|\mathbf{z}_{-i}, y_i &\sim \begin{cases} \text{normal}(m_i, v_i)I(z_i > 0) & \text{if } y_i = 1 \\ \text{normal}(m_i, v_i)I(z_i \le 0) & \text{otherwise} \end{cases} \\ m_i &= \mathbf{x}_i\mathbf{B} - w_i(z_i - \mathbf{x}_i\mathbf{B}) \\ v_i &= 1 + w_i \\ w_i &= h_i/(1 - h_i) \\ h_i &= (\mathbf{H})_i i, \mathbf{H} = \mathbf{x}\mathbf{V}\mathbf{x}' \end{aligned}$$

2. Sample $\boldsymbol{\beta}$ from the same posterior full conditional distribution described previously.

For a detailed description of each of the conditional terms, refer to the original paper.

PROC MCMC cannot sample from the probit model by using this sampling scheme but you can implement the algorithm by using the UDS statement. To sample z_i from its marginal, you need a function that draws random variables from a truncated normal distribution. The functions, RLTNORM and RRTNORM, generate left- and right-truncated normal variates, respectively. The algorithm is taken from Robert (1995).

The functions are written in PROC FCMP (see the FCMP Procedure in the *Base SAS Procedures Guide*):

```
proc fcmp outlib=sasuser.funcs.uds;
   /*****************************************/
   /* Generate left-truncated normal variate */
   /*****************************************/
   function rltnorm(mu,sig,lwr);
   if lwr<mu then do;
      ans = lwr-1;
```

```
      do while(ans<lwr);
         ans = rand('normal',mu,sig);
      end;
   end;
   else do;
      mul = (lwr-mu)/sig;
      alpha = (mul + sqrt(mul**2 + 4))/2;
      accept=0;
      do while(accept=0);
         z = mul + rand('exponential')/alpha;
         lrho = -(z-alpha)**2/2;
         u = rand('uniform');
         lu = log(u);
         if lu <= lrho then accept=1;
      end;
      ans = sig*z + mu;
   end;
   return(ans);
   endsub;

   /*****************************************/
   /* Generate right-truncated normal variate */
   /*****************************************/
   function rrtnorm(mu,sig,uppr);
   ans = 2*mu - rltnorm(mu,sig, 2*mu-uppr);
   return(ans);
   endsub;
run;
```

The function call to RLTNORM(mu,sig,lwr) generates a random number from the left-truncated normal distribution:

$$\theta \sim \text{normal}(\text{mu}, \text{sd} = \text{sig}) I(\theta > \text{lwr})$$

Similarly, the function call to RRTNORM(mu,sig,uppr) generates a random number from the right-truncated normal distribution:

$$\theta \sim \text{normal}(\text{mu}, \text{sd} = \text{sig}) I(\theta < \text{uppr})$$

These functions are used to generate the latent variables z_i.

Using the algorithm A1 from the HH paper as an example, Output 80.56 lists a line-by-line implementation with the PROC MCMC coding style. The table is broken into three portions: set up the constants, initialize the parameters, and sample one draw from the posterior distribution. The left column of the table is identical to the A1 algorithm stated in the appendix of HH. The right column of the table lists SAS statements.

Table 80.56 Holmes and Held (2006), algorithm A1.
Side-by-Side Comparison to SAS

Define Constants	In the BEGINCNST/ENDCNST Statements

$V \leftarrow (X'X + v^{-1})^{-1}$	`call transpose(x,xt); /* xt = transpose(x) */`
	`call mult(xt,x,xtx);`
	`call inv(v,v); /* v = inverse(v) */`
	`call addmatrix(xtx,v,xtx); /* xtx = xtx+v */`
	`call inv(xtx,v); /* v = inverse(xtx) */`

$L \leftarrow \mathrm{Chol}(V)$	`call chol(v,L);`
$S \leftarrow VX'$	`call mult(v,xt,S);`

FOR $j = 1$ to n
$H[j] \leftarrow X[j,]S[,j]$
$W[j] \leftarrow H[j]/(1 - H[j])$
$Q[j] \leftarrow W[j] + 1$
END
	`call mult(x,S,HatMat);`
	`do j=1 to &n;`
	`  H = HatMat[j,j];`
	`  W[j] = H/(1-H);`
	`  sQ[j] = sqrt(W[j] + 1); /* use s.d.  in SAS */`
	`  end;`

Initial Values	In the BEGINCNST/ENDCNST Statements

$Z \sim \mathrm{normal}(0, I_n)\mathrm{Ind}(Y, Z)$	`do j=1 to &n;`
	`  if(y[j]=1) then`
	`  Z[j] = rltnorm(0,1,0);`
	`  else`
	`  Z[j] = rrtnorm(0,1,0);`
	`  end;`

$B \leftarrow SZ$	`call mult(S,Z,B);`

Table 80.56 *continued*

Draw One Sample	Subroutine HH

FOR $j = 1$ to n
$z_{old} \leftarrow Z[j]$
$m \leftarrow X[j,]B$
$m \leftarrow m - W[j](Z[j] - m)$
$Z[j] \sim \text{normal}(m, Q[j])\text{Ind}(Y[j], Z[j])$
$B \leftarrow B + (Z[j] - z_{old})S[, j]$
END
$T \sim \text{normal}(0, I_p)$
$\beta[, i] \leftarrow B + LT$
	`do j=1 to &n;`
	`  zold = Z[j];`
	`  m = 0;`
	`  do k= 1 to &p;`
	`  m = m + X[j,k] * B[k];`
	`  end;`
	`  m = m - W[j]*(Z[j]-m);`
	`  if (y[j]=1) then`
	`  Z[j] = rltnorm(m,sQ[j],0);`
	`  else`
	`  Z[j] = rrtnorm(m,sQ[j],0);`
	`  diff = Z[j] - zold;`
	`  do k= 1 to &p;`
	`  B[k] = B[k] + diff * S[k,j];`
	`  end;`
	`  end;`
	`do j = 1 to &p;`
	`  T[j] = rand('normal');`
	`  end;`
	`call mult(L,T,T);`
	`call addmatrix(B,T,beta);`

The following statements define the subroutine HH (algorithm A1) in PROC FCMP and store it in library **sasuser.funcs.uds**:

```
/* define the HH algorithm in PROC FCMP. */
%let n = 27;
%let p = 7;
options cmplib=sasuser.funcs;
proc fcmp outlib=sasuser.funcs.uds;
   subroutine HH(beta[*],Z[*],B[*],x[*,*],y[*],W[*],sQ[*],S[*,*],L[*,*]);
   outargs beta, Z, B;
   array T[&p] / nosym;
   do j=1 to &n;
      zold = Z[j];
      m = 0;
      do k = 1 to &p;
         m = m + X[j,k] * B[k];
      end;
      m = m - W[j]*(Z[j]-m);
      if (y[j]=1) then
         Z[j] = rltnorm(m,sQ[j],0);
      else
         Z[j] = rrtnorm(m,sQ[j],0);
      diff = Z[j] - zold;
      do k = 1 to &p;
```

```
         B[k] = B[k] + diff * S[k,j];
      end;
   end;
   do j=1 to &p;
      T[j] = rand('normal');
   end;
   call mult(L,T,T);
   call addmatrix(B,T,beta);
   endsub;
run;
```

Note that one-dimensional array arguments take the form of *name*[*] and two-dimensional array arguments take the form of *name*[*,*]. Three variables, beta, Z, and B, are OUTARGS variables, making them the only arguments that can be modified in the subroutine. For the UDS statement to work, all OUTARGS variables have to be model parameters. Technically, only beta and Z are model parameters, and B is not. The reason that B is declared as an OUTARGS is because the array must be updated throughout the simulation, and this is the only way to modify its values. The input array x contains all of the explanatory variables, and the array y stores the response. The rest of the input arrays, W, sQ, S, and L, store constants as detailed in the algorithm. The following statements illustrate how to fit a Bayesian probit model by using the HH algorithm:

```
options cmplib=sasuser.funcs;

proc mcmc data=inputdata nmc=5000 monitor=(beta) outpost=hhout;
   ods select PostSumInt ess;
   array xtx[&p,&p];      /* work space                        */
   array xt[&p,&n];       /* work space                        */
   array v[&p,&p];        /* work space                        */
   array HatMat[&n,&n];   /* work space                        */
   array S[&p,&n];        /* V * Xt                            */
   array W[&n];
   array y[1]/ nosymbols; /* y stores the response variable    */
   array x[1]/ nosymbols; /* x stores the explanatory variables */
   array sQ[&n];          /* sqrt of the diagonal elements of Q */
   array B[&p];           /* conditional mean of beta          */
   array L[&p,&p];        /* Cholesky decomp of conditional cov */
   array Z[&n];           /* latent variables Z                */
   array beta[&p] beta0-beta6;   /* regression coefficients    */

   begincnst;
      call streaminit(83101);
      if ind=1 then do;
         rc = read_array("inputdata", x, "cnst", "cell", "smear", "infil",
                         "li", "blast", "temp");
         rc = read_array("inputdata", y, "remiss");
         call identity(v);
         call mult(v, 25, v);
         call transpose(x,xt);
         call mult(xt,x,xtx);
         call inv(v,v);
         call addmatrix(xtx,v,xtx);
         call inv(xtx,v);
         call chol(v,L);
         call mult(v,xt,S);
```

```
            call mult(x,S,HatMat);
            do j=1 to &n;
               H = HatMat[j,j];
               W[j] = H/(1-H);
               sQ[j] = sqrt(W[j] + 1);
            end;

            do j=1 to &n;
               if(y[j]=1) then
                   Z[j] = rltnorm(0,1,0);
               else
                   Z[j] = rrtnorm(0,1,0);
            end;
            call mult(S,Z,B);
         end;
      endcnst;

      uds HH(beta,Z,B,x,y,W,sQ,S,L);
      parms z: beta: 0 B1-B7 / uds;
      prior z: beta: B1-B7 ~ general(0);

      model general(0);
   run;
```

The OPTIONS statement names the catalog of FCMP subroutines to use. The cmplib library stores the subroutine HH. You do not need to set a random number seed in the PROC MCMC statement because all random numbers are generated from the HH subroutine. The initialization of the rand function is controlled by the streaminit function, which is called in the program with a seed value of 83101.

A number of arrays are allocated. Some of them, such as xtx, xt, v, and HatMat, allocate work space for constant arrays. Other arrays are used in the subroutine sampling. Explanations of the arrays are shown in comments in the statements.

In the BEGINCNST and ENDCNST statement block, you read data set variables in the arrays x and y, calculate all the constant terms, and assign initial values to Z and B. For the READ_ARRAY function, see the section "READ_ARRAY Function" on page 6249. For listings of all array functions and their definitions, see the section "Matrix Functions in PROC MCMC" on page 6313.

The UDS statement declares that the subroutine HH is used to sample the parameters beta, Z, and B. You also specify the UDS option in the PARMS statement. Because all parameters are updated through the UDS interface, it is not necessary to declare the actual form of the prior for any of the parameters. Each parameter is declared to have a prior of general(0). Similarly, it is not necessary to declare the actual form of the likelihood. The MODEL statement also takes a flat likelihood of the form **general(0)**.

Summary statistics and effective sample sizes are shown in Output 80.19.2. The posterior estimates are very close to what was shown in Output 80.19.1. The HH algorithm produces samples that are much less correlated.

Output 80.19.2 Holms and Held

Implement a New Sampling Algorithm

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta0	5000	-2.0567	3.8260	-9.4031	5.2733
beta1	5000	2.7254	2.8079	-2.3940	8.5828
beta2	5000	-0.8318	3.2017	-6.6219	5.8170
beta3	5000	1.6319	3.5108	-5.7117	7.9353
beta4	5000	2.0567	0.8800	0.3155	3.7289
beta5	5000	-0.3473	0.9490	-2.1478	1.5889
beta6	5000	-3.3787	3.7991	-10.6821	4.1930

Implement a New Sampling Algorithm

The MCMC Procedure

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
beta0	3651.3	1.3694	0.7303
beta1	1563.8	3.1973	0.3128
beta2	5005.9	0.9988	1.0012
beta3	4853.2	1.0302	0.9706
beta4	2611.2	1.9148	0.5222
beta5	3049.2	1.6398	0.6098
beta6	3503.2	1.4273	0.7006

It is interesting to compare the two approaches of fitting a generalized linear model. The random walk Metropolis on a seven-dimensional parameter space produces autocorrelations that are substantially higher than the HH algorithm. A much longer chain is needed to produce roughly equivalent effective sample sizes. On the other hand, the Metropolis algorithm is faster to run. The running time of these two examples is roughly the same, with the random walk Metropolis with 100000 samples, a 20-fold increase over that in the HH algorithm example. The speed difference can be attributed to a number of factors, ranging from the implementation of the software and the overhead cost of calling PROC FCMP subroutine and functions. In addition, the HH algorithm requires more parameters by creating an equal number of latent variables as the sample size. Sampling more parameters takes time. A larger number of parameters also increases the challenge in convergence diagnostics, because it is imperative to have convergence in all parameters before you make valid posterior inferences. Finally, you might feel that coding in PROC MCMC is easier. However, this really is not a fair comparison to make here. Writing a Metropolis algorithm from scratch would have probably taken just as much, if not more, effort than the HH algorithm.

Example 80.20: Using a Transformation to Improve Mixing

(View the complete code for this example (mcmcex20.sas) in the example repository.)

Proper transformations of parameters can often improve the mixing in PROC MCMC. You already saw this in "Example 80.6: Nonlinear Poisson Regression Models" on page 6406, which sampled using the log scale of parameters that priors that are strictly positive, such as the gamma priors. This example shows another useful transformation: the logit transformation on parameters that take a uniform prior on [0, 1].

The data set is taken from Sharples (1990). It is used in Chaloner and Brant (1988) and Chaloner (1994) to identify outliers in the data set in a two-level hierarchical model. Congdon (2003) also uses this data set to demonstrates the same technique. This example uses the data set to illustrate how mixing can be improved using transformation and does not address the question of outlier detection as in those papers. The following statements create the data set:

```
data inputdata;
   input nobs grp y @@;
   ind = _n_;
   datalines;
1 1 24.80 2 1 26.90 3 1 26.65
4 1 30.93 5 1 33.77 6 1 63.31
1 2 23.96 2 2 28.92 3 2 28.19
4 2 26.16 5 2 21.34 6 2 29.46
1 3 18.30 2 3 23.67 3 3 14.47
4 3 24.45 5 3 24.89 6 3 28.95
1 4 51.42 2 4 27.97 3 4 24.76
4 4 26.67 5 4 17.58 6 4 24.29
1 5 34.12 2 5 46.87 3 5 58.59
4 5 38.11 5 5 47.59 6 5 44.67
;
```

There are five groups (grp, $j = 1, \ldots, 5$) with six observations (nobs, $i = 1, \ldots, 6$) in each. The two-level hierarchical model is specified as follows:

$$
\begin{aligned}
y_{ij} &\sim \text{normal}(\theta_j, \text{prec} = \tau_w) \\
\theta_j &\sim \text{normal}(\mu, \text{prec} = \tau_b) \\
\mu &\sim \text{normal}(0, \text{prec} = 1e-6) \\
\tau &\sim \text{gamma}(0.001, \text{iscale} = 0.001) \\
p &\sim \text{uniform}(0, 1)
\end{aligned}
$$

with the precision parameters related to each other in the following way:

$$
\begin{aligned}
\tau_b &= \tau/p \\
\tau_w &= \tau_b - \tau
\end{aligned}
$$

The total number of parameters in this model is eight: $\theta_1, \ldots, \theta_5, \mu, \tau$, and p.

The following statements fit the model:

```
ods graphics on;
proc mcmc data=inputdata nmc=50000 thin=10 ntu=2000
          outpost=m1 seed=17797 plot=trace;
   parms p;
   parms tau;
   parms mu;

   prior p ~ uniform(0,1);
   prior tau ~ gamma(shape=0.001,iscale=0.001);
   prior mu ~ normal(0,prec=0.00000001);
   beginnodata;
   taub = tau/p;
   tauw = taub-tau;
   endnodata;

   random theta ~ normal(mu, prec=taub) subject=grp monitor=(theta);
   model y ~ normal(theta,prec=tauw);
run;
```

The ODS SELECT statement displays the effective sample size table and the trace plots. The ODS GRAPHICS ON statement enables ODS Graphics. The PROC MCMC statement specifies the usual options for the procedure run and produces trace plots (PLOTS=TRACE). The three PARMS statements put three model parameters, p, tau, and mu, in three different blocks. The PRIOR statements specify the prior distributions, and the programming statements enclosed with the BEGINNODATA and ENDNODATA statements calculate the transformation to taub and tauw. The RANDOM statement specifies the random effect, its prior distribution, and the subject variable. The resulting trace plots are shown in Output 80.20.1, and the effective sample size table is shown in Output 80.20.2.

Output 80.20.1 Trace Plots

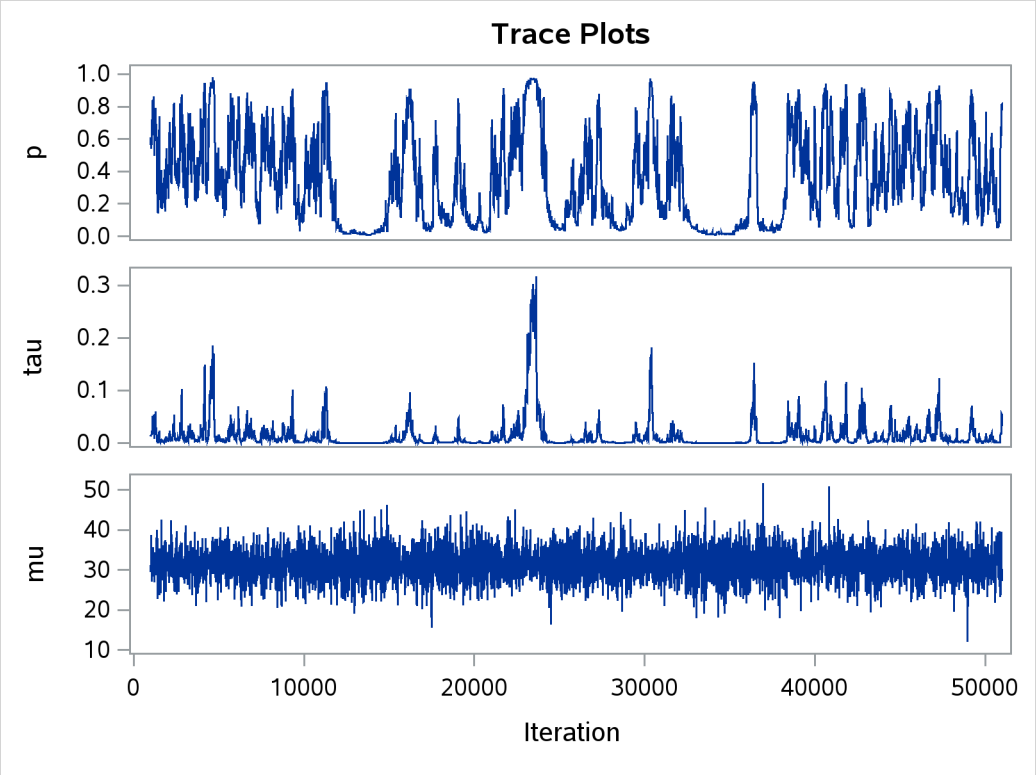

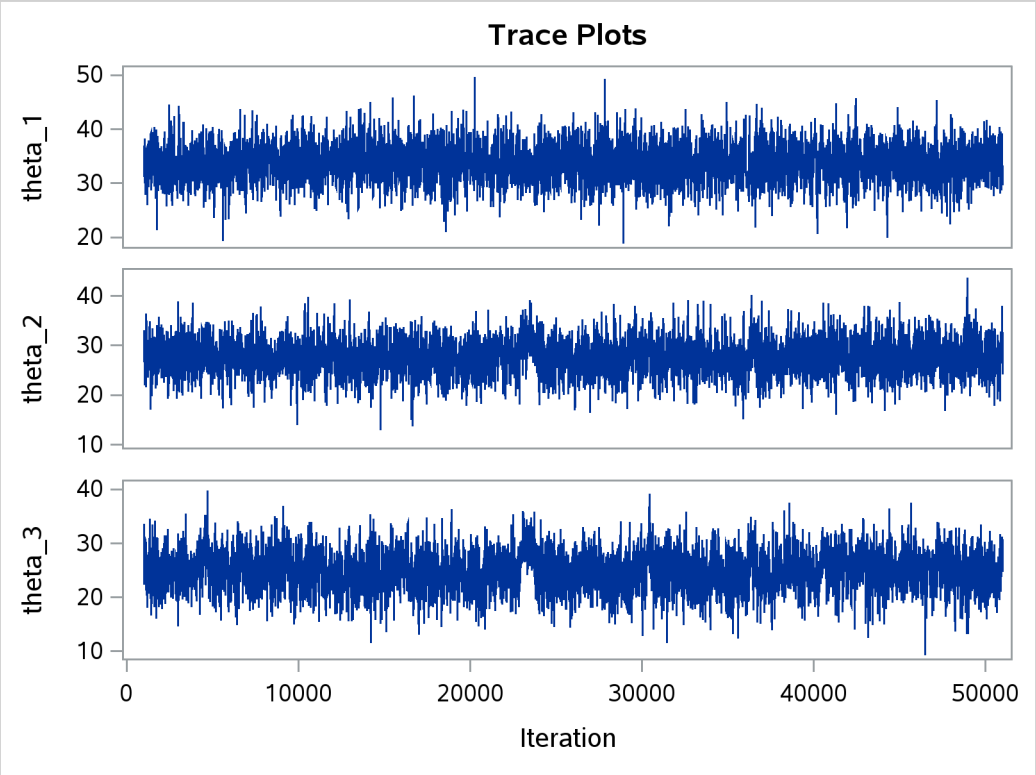

Output 80.20.1 *continued*

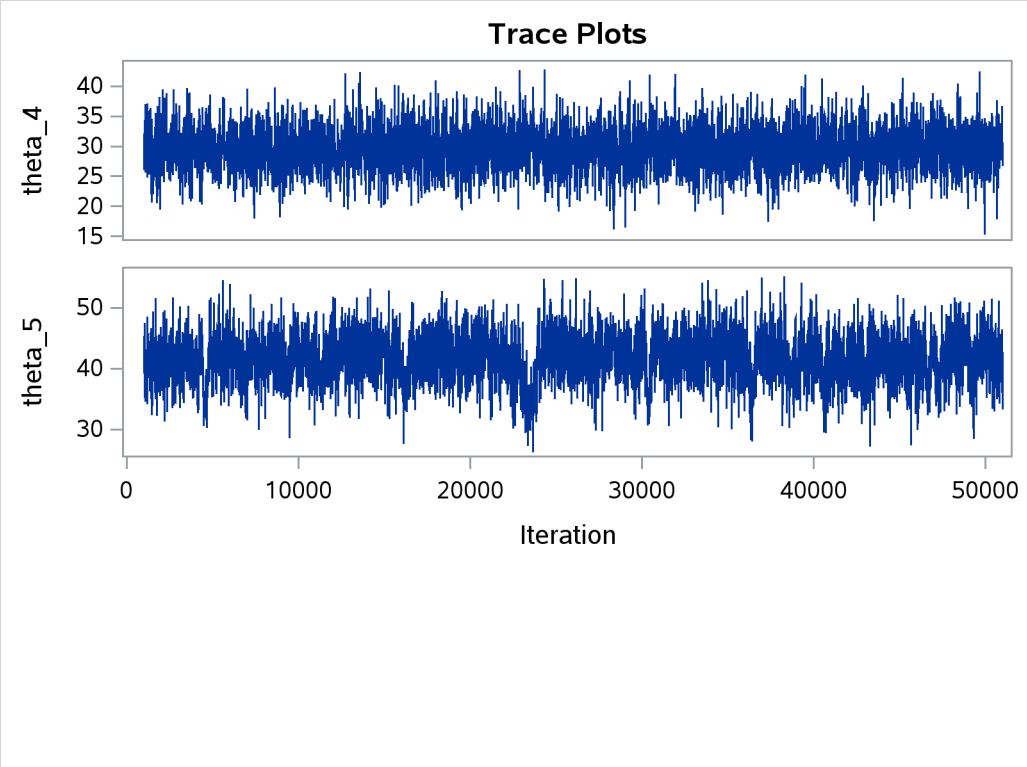

Output 80.20.2 Bad Effective Sample Sizes

The MCMC Procedure

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
p	76.3	65.5445	0.0153
tau	100.1	49.9391	0.0200
mu	4901.2	1.0202	0.9802
theta_1	4330.1	1.1547	0.8660
theta_2	4575.3	1.0928	0.9151
theta_3	1273.1	3.9274	0.2546
theta_4	5000.0	1.0000	1.0000
theta_5	621.8	8.0412	0.1244

The trace plots show that most parameters have relatively good mixing. Two exceptions appear to be p and τ. The trace plot of p shows a slow periodic movement. The τ parameter does not have good mixing either. When the values are close to zero, the chain stays there for periods of time. An inspection of the effective sample sizes table reveals the same conclusion: p and τ have much smaller ESSs than the rest of the parameters.

A scatter plot of the posterior samples of p and τ reveals why mixing is bad in these two dimensions. The following statements generate the scatter plot in Output 80.20.3:

```
title 'Scatter Plot of Parameters on Original Scales';

proc sgplot data=m1;
   yaxis label = 'p';
   xaxis label = 'tau' values=(0 to 0.4 by 0.1);
   scatter x = tau y = p;
run;
```

Output 80.20.3 Scatter Plot of τ versus p

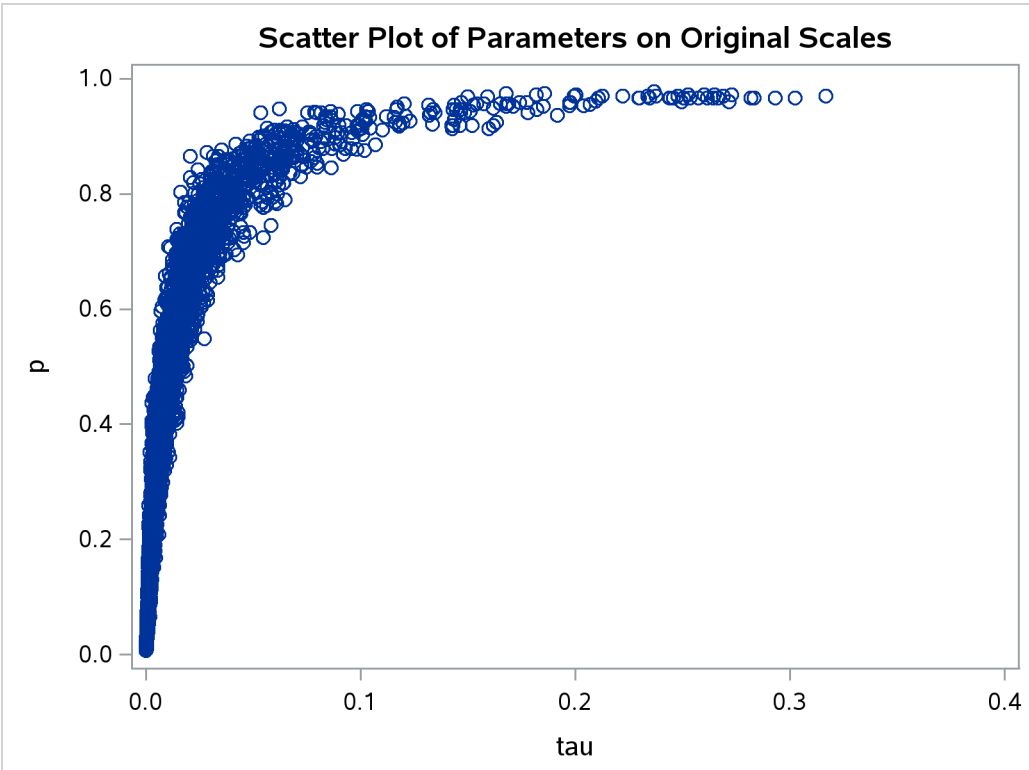

The two parameters clearly have a nonlinear relationship. It is not surprising that the Metropolis algorithm does not work well here. The algorithm is designed for cases where the parameters are linearly related with each other.

To improve on mixing, you can sample on the log of τ, instead of sampling on τ. The formulation is:

$$\begin{aligned} \tau &\sim \text{gamma}(\text{shape} = 0.001, \text{iscale} = 0.001) \\ \log(\tau) &\sim \text{egamma}(\text{shape} = 0.001, \text{iscale} = 0.001) \end{aligned}$$

See the section "Standard Distributions" on page 6289 for the definitions of the gamma and egamma distributions. In addition, you can sample on the logit of p. Note that

$$p \sim \text{uniform}(0, 1)$$

is equivalent to

$$\text{lgp} = \text{logit}(p) \sim \text{logistic}(0, 1)$$

The following statements fit the same model by using transformed parameters:

```
proc mcmc data=inputdata nmc=50000 thin=10 outpost=m2 seed=17
          monitor=(p tau mu) plot=trace;
   ods select ess tracepanel;
   parms ltau lgp mu ;

   prior ltau ~ egamma(shape=0.001,iscale=0.001);
   prior lgp ~ logistic(0,1);
   prior mu ~ normal(0,prec=0.00000001);

   beginnodata;
   tau = exp(ltau);
   p = logistic(lgp);
   taub = tau/p;
   tauw = taub-tau;
   endnodata;

   random theta ~ normal(mu, prec=taub) subject=grp monitor=(theta);
   model y ~ normal(theta,prec=tauw);
run;
```

The variable lgp is the logit transformation of p, and ltau is the log transformation of τ. The prior for ltau is egamma, and the prior for lgp is logistic. The TAU and P assignment statements transform the parameters back to their original scales. The rest of the programs remain unchanged. Trace plots (Output 80.20.4) and effective sample size (Output 80.20.5) both show significant improvements in the mixing for both p and τ.

Output 80.20.4 Trace Plots after Transformation

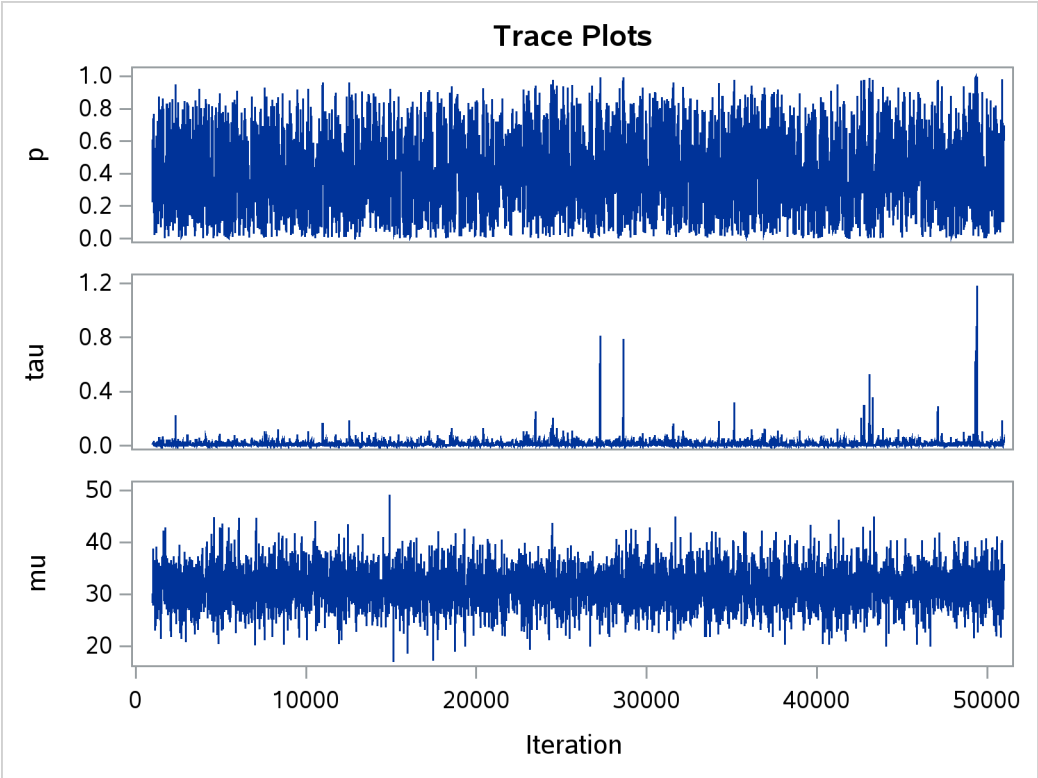

Output 80.20.4 *continued*

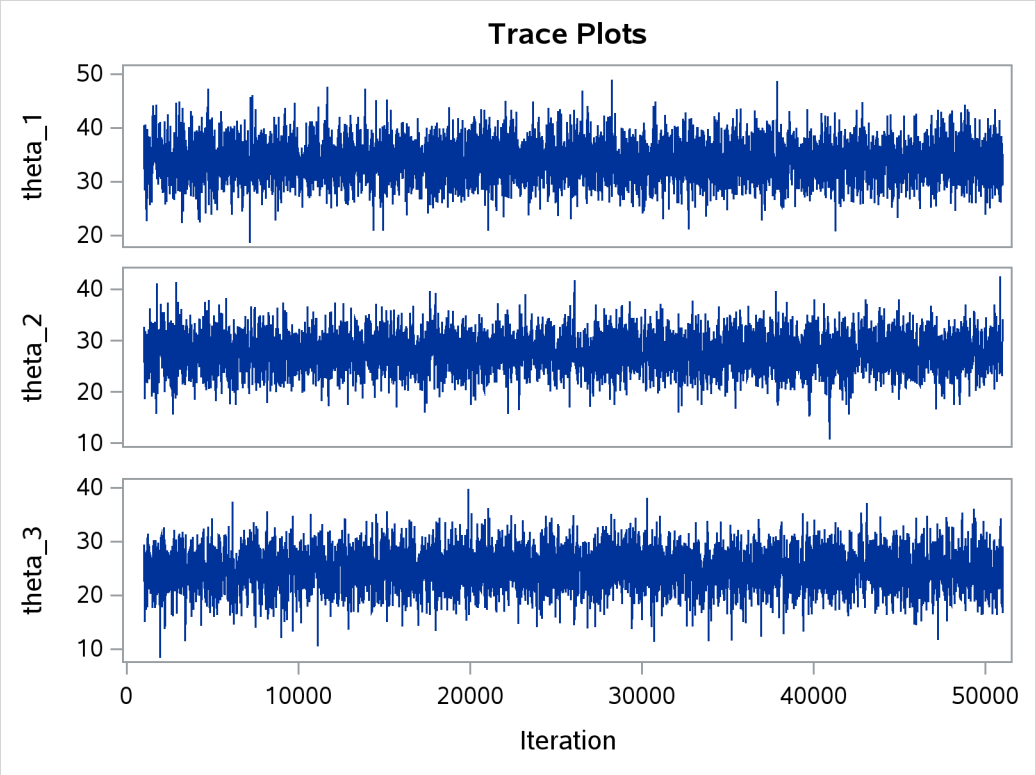

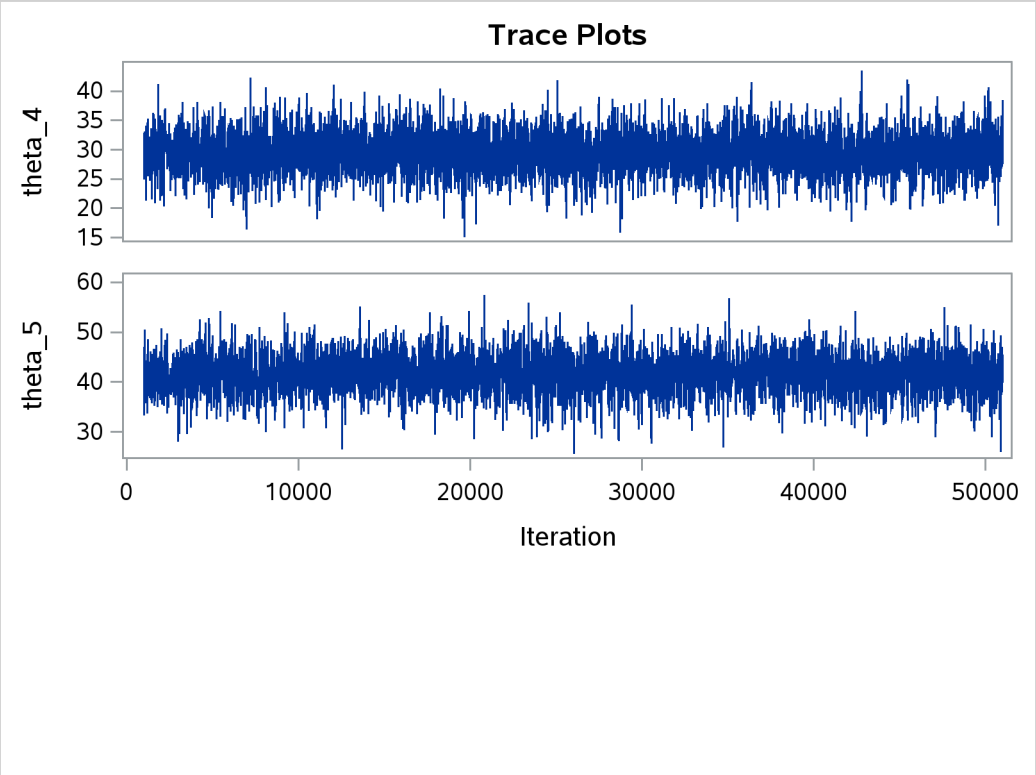

Output 80.20.5 Effective Sample Sizes after Transformation

The MCMC Procedure

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
p	3078.7	1.6241	0.6157
tau	2798.9	1.7864	0.5598
mu	5000.0	1.0000	1.0000
theta_1	5127.1	0.9752	1.0254
theta_2	4798.1	1.0421	0.9596
theta_3	4741.2	1.0546	0.9482
theta_4	5196.7	0.9621	1.0393
theta_5	4282.7	1.1675	0.8565

The following statements generate Output 80.20.6 and Output 80.20.7:

```
title 'Scatter Plot of Parameters on Transformed Scales';

proc sgplot data=m2;
   yaxis label = 'logit(p)';
   xaxis label = 'log(tau)';
   scatter x = ltau y = lgp;
run;

title 'Scatter Plot of Parameters on Original Scales';

proc sgplot data=m2;
   yaxis label = 'p';
   xaxis label = 'tau' values=(0 to 5.0 by 1);
   scatter x = tau y = p;
run;
ods graphics off;
```

Output 80.20.6 Scatter Plot of $\log(\tau)$ versus logit(p), After Transformation

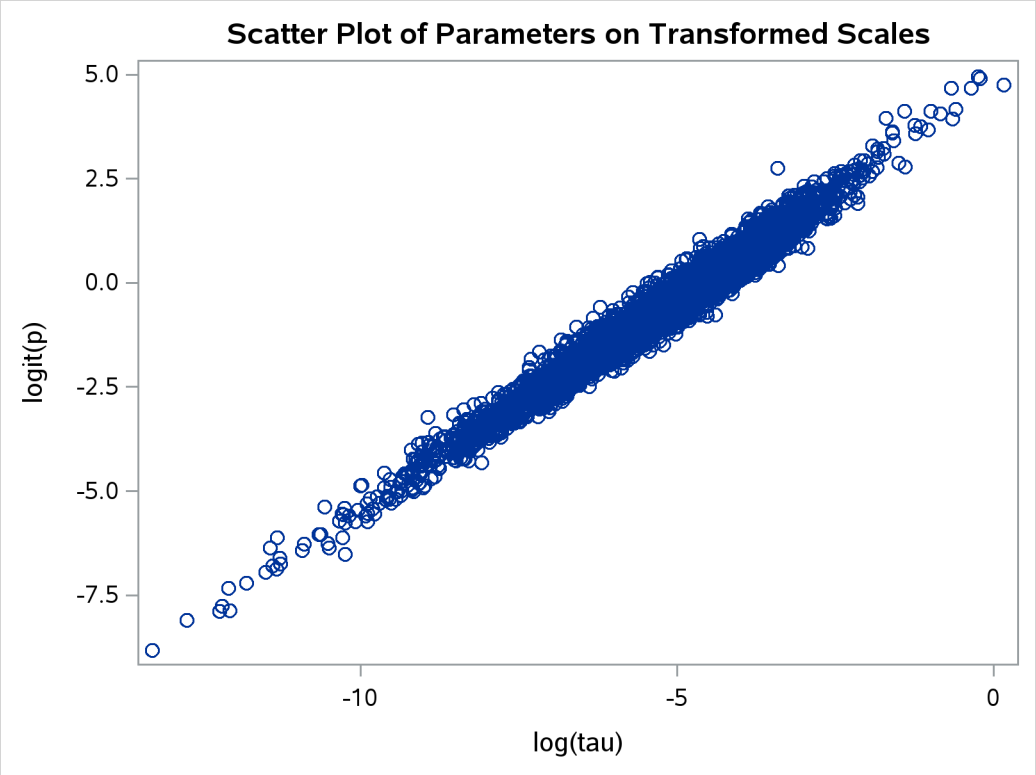

Output 80.20.7 Scatter Plot of τ versus p, After Transformation

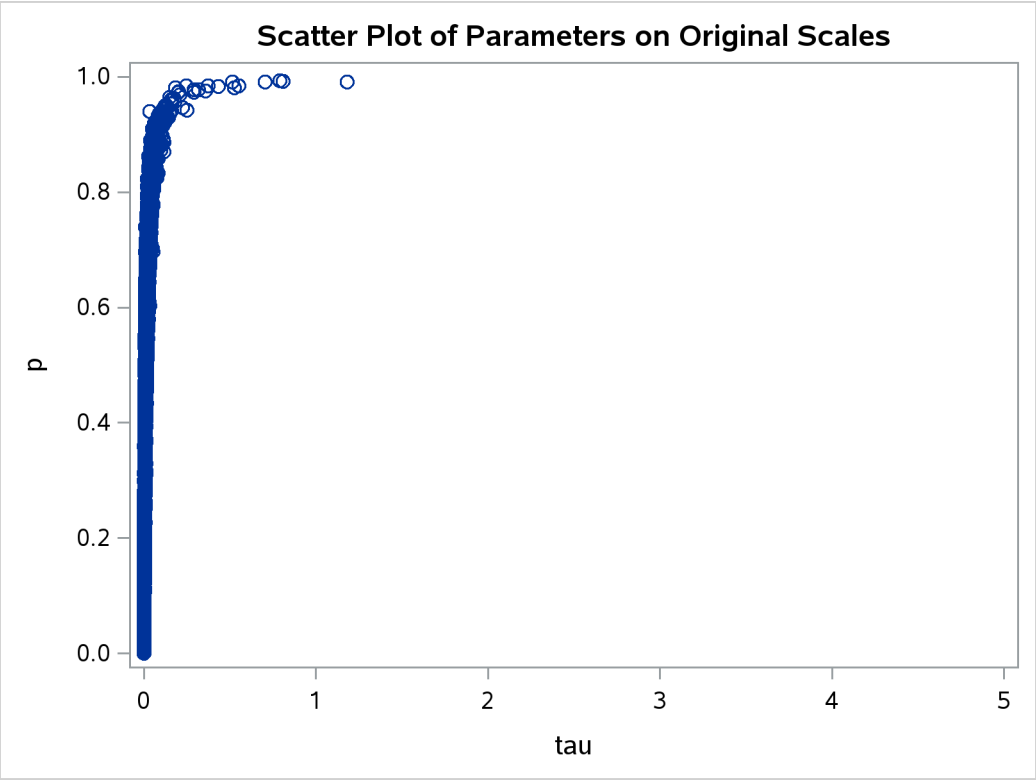

The scatter plot of $\log(\tau)$ versus $\text{logit}(p)$ shows a linear relationship between the two transformed parameters, and this explains the improvement in mixing. In addition, the transformations also help the Markov chain better explore in the original parameter space. Output 80.20.7 shows a scatter plot of τ versus p. The plot is similar to Output 80.20.3. However, note that tau has a far longer tail in Output 80.20.7, extending all the way to 5 as opposed to 0.15 in Output 80.20.3. This means that the second Markov chain can explore this dimension of the parameter more efficiently, and as a result, you are able to draw more precise inference with an equal number of simulations.

Example 80.21: Gelman-Rubin Diagnostics

(View the complete code for this example (mcmcex21.sas) in the example repository.)

PROC MCMC does not have the Gelman-Rubin test (see the section "Gelman and Rubin Diagnostics" on page 167 in Chapter 8, "Introduction to Bayesian Analysis Procedures") as a part of its diagnostics. The Gelman-Rubin diagnostics rely on parallel chains to test whether they all converge to the same posterior distribution. This example demonstrates how you can carry out this convergence test. The regression model from the section "Simple Linear Regression" on page 6215 is used. The model has three parameters: β_0 and β_1 are the regression coefficients, and σ^2 is the variance of the error distribution.

To run a Gelman-Rubin diagnostic test, you want to start Markov chains at different places in the parameter space. Suppose you want to start β_0 at 10, –15, and 0; β_1 at –5, 10, and 0; and σ^2 at 1, 20, and 50. You can put these starting values in the following Init SAS data set:

```
title 'Simple Linear Regression, Gelman-Rubin Diagnostics';

data init;
   input Chain beta0 beta1 sigma2;
   datalines;
   1   10  -5   1
   2  -15  10  20
   3    0   0  50
;
```

The following statements run PROC MCMC three times, each with starting values specified in the data set Init:

```
/* define constants */
%let nchain = 3;
%let nparm = 3;
%let nsim = 50000;
%let var = beta0 beta1 sigma2;

%macro gmcmc;
   %do i=1 %to &nchain;
      data _null_;
         set init;
         if Chain=&i;
         %do j = 1 %to &nparm;
            call symputx("init&j", %scan(&var, &j));
         %end;
         stop;
```

```
      run;

      proc mcmc data=sashelp.class outpost=out&i init=reinit nbi=0 nmc=&nsim
                stats=none seed=7;
         parms beta0 &init1 beta1 &init2;
         parms sigma2 &init3 / n;
         prior beta0 beta1 ~ normal(0, var = 1e6);
         prior sigma2 ~ igamma(3/10, scale = 10/3);
         mu = beta0 + beta1*height;
         model weight ~ normal(mu, var = sigma2);
      run;
   %end;
%mend;

ods exclude all;
%gmcmc;
ods exclude none;
```

The macro variables nchain, nparm, nsim, and var define the number of chains, the number of parameters, the number of Markov chain simulations, and the parameter names, respectively. The macro GMCMC gets initial values from the data set Init, assigns them to the macro variables init1, init2 and init3, starts the Markov chain at these initial values, and stores the posterior draws to three output data sets: Out1, Out2, and Out3.

In the PROC MCMC statement, the INIT=REINIT option restarts the Markov chain after tuning at the assigned initial values. No burn-in is requested.

You can use the autocall macro GELMAN to calculate the Gelman-Rubin statistics by using the three chains. The GELMAN macro has the following arguments:

```
%macro gelman(dset, nparm, var, nsim, nc=3, alpha=0.05);
```

The argument dset is the name of the data set that stores the posterior samples from all the runs, nparm is the number of parameters, var is the name of the parameters, nsim is the number of simulations, nc is the number of chains with a default value of 3, and alpha is the α significant level in the test with a default value of 0.05. This macro creates two data sets: _Gelman_Ests stores the diagnostic estimates and _Gelman_Parms stores the names of the parameters.

The following statements calculate the Gelman-Rubin diagnostics:

```
data all;
   set out1(in=in1) out2(in=in2) out3(in=in3);
   if in1 then Chain=1;
   if in2 then Chain=2;
   if in3 then Chain=3;
run;

%gelman(all, &nparm, &var, &nsim);

data GelmanRubin(label='Gelman-Rubin Diagnostics');
   merge _Gelman_Parms _Gelman_Ests;
run;

proc print data=GelmanRubin;
run;
```

The Gelman-Rubin statistics are shown in Output 80.21.1.

Output 80.21.1 Gelman-Rubin Diagnostics of the Regression Example

Simple Linear Regression, Gelman-Rubin Diagnostics

Obs	Parameter	Between-chain	Within-chain	Estimate	UpperBound
1	beta0	5384.76	1168.64	1.0002	1.0001
2	beta1	1.20	0.30	1.0002	1.0002
3	sigma2	8034.41	2890.00	1.0010	1.0011

The Gelman-Rubin statistics do not reveal any concerns about the convergence or the mixing of the multiple chains. To get a better visual picture of the multiple chains, you can draw overlapping trace plots of these parameters from the three Markov chains runs.

The following statements create Output 80.21.2:

```
/* plot the trace plots of three Markov chains. */
%macro trace;
   %do i = 1 %to &nparm;
      proc sgplot data=all cycleattrs;
         series x=Iteration y=%scan(&var, &i) / group=Chain;
      run;
   %end;
%mend;
%modstyle(name=linestyle, linestyles=Solid)
ods listing style=linestyle;
%trace;
```

Output 80.21.2 Trace Plots of Three Chains for Each of the Parameters

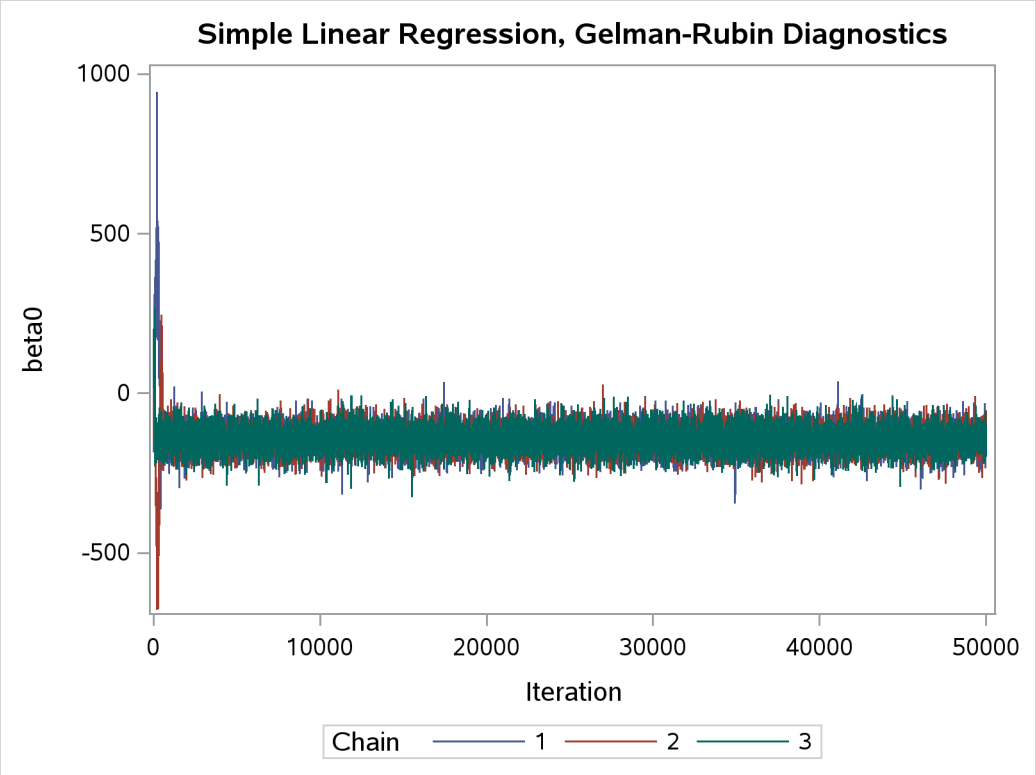

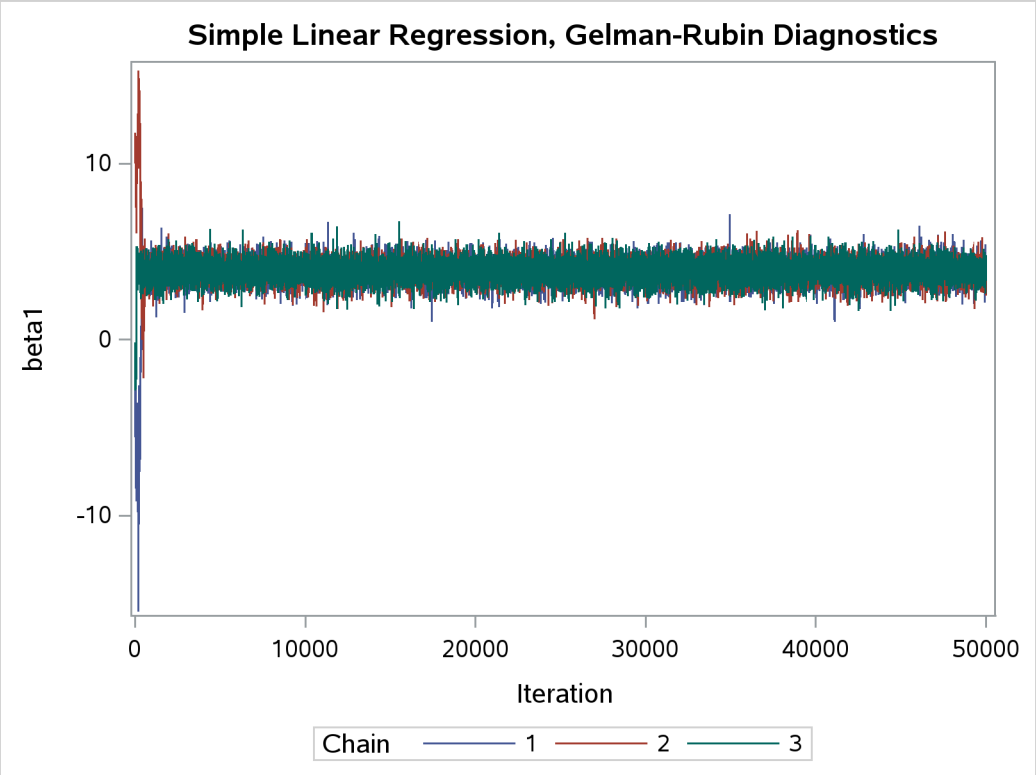

Output 80.21.2 *continued*

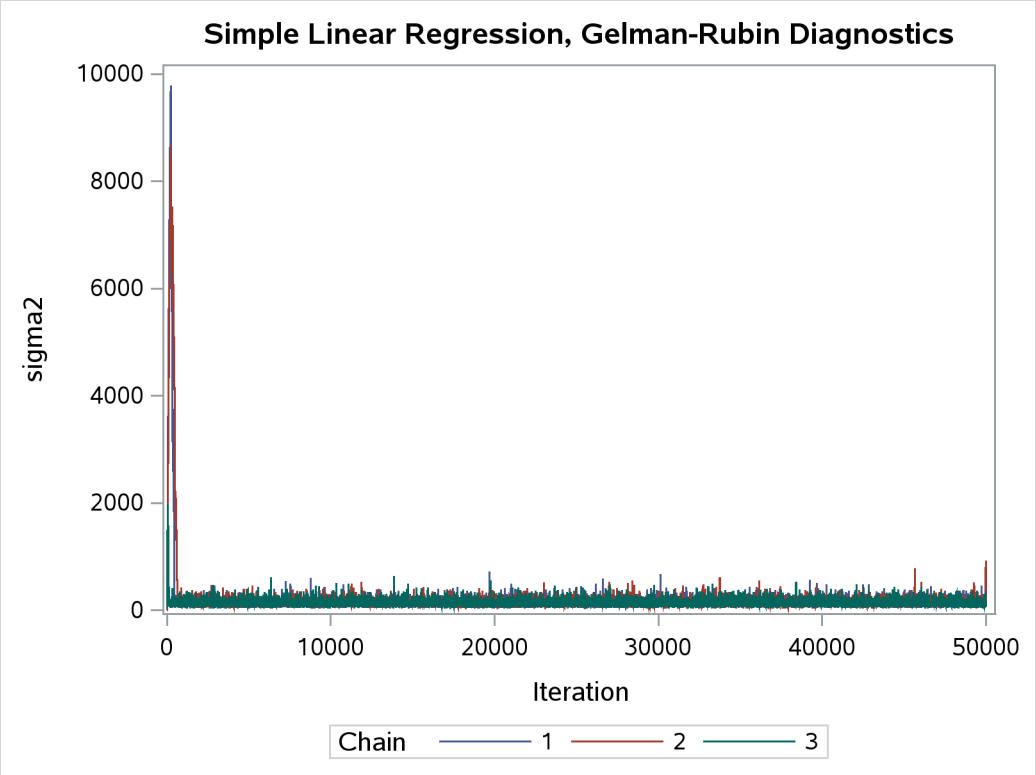

The trace plots show that three chains all eventually converge to the same regions even though they started at very different locations. In addition to the trace plots, you can also plot the potential scale reduction factor (PSRF). See the section "Gelman and Rubin Diagnostics" on page 167 in Chapter 8, "Introduction to Bayesian Analysis Procedures," for definition and details.

The following statements calculate PSRF for each parameter. They use the GELMAN macro repeatedly and can take a while to run:

```
/* define sliding window size */
%let nwin = 200;
data PSRF;
run;

%macro PSRF(nsim);
   %do k = 1 %to %sysevalf(&nsim/&nwin, floor);
      %gelman(all, &nparm, &var, nsim=%sysevalf(&k*&nwin));
      data GelmanRubin;
         merge _Gelman_Parms _Gelman_Ests;
      run;

      data PSRF;
         set PSRF GelmanRubin;
      run;
   %end;
%mend PSRF;
```

```
options nonotes;
%PSRF(&nsim);
options notes;

data PSRF;
   set PSRF;
   if _n_ = 1 then delete;
run;

proc sort data=PSRF;
   by Parameter;
run;

%macro sepPSRF(nparm=, var=, nsim=);
   %do k = 1 %to &nparm;
      data save&k; set PSRF;
         if _n_ > %sysevalf(&k*&nsim/&nwin, floor) then delete;
         if _n_ < %sysevalf((&k-1)*&nsim/&nwin + 1, floor) then delete;
         Iteration + &nwin;
      run;

      proc sgplot data=save&k(firstobs=10) cycleattrs;
         series x=Iteration y=Estimate;
         series x=Iteration y=upperbound;
         yaxis label="%scan(&var, &k)";
      run;
   %end;
%mend sepPSRF;

%sepPSRF(nparm=&nparm, var=&var, nsim=&nsim);
```

Output 80.21.3 PSRF Plot for Each Parameter

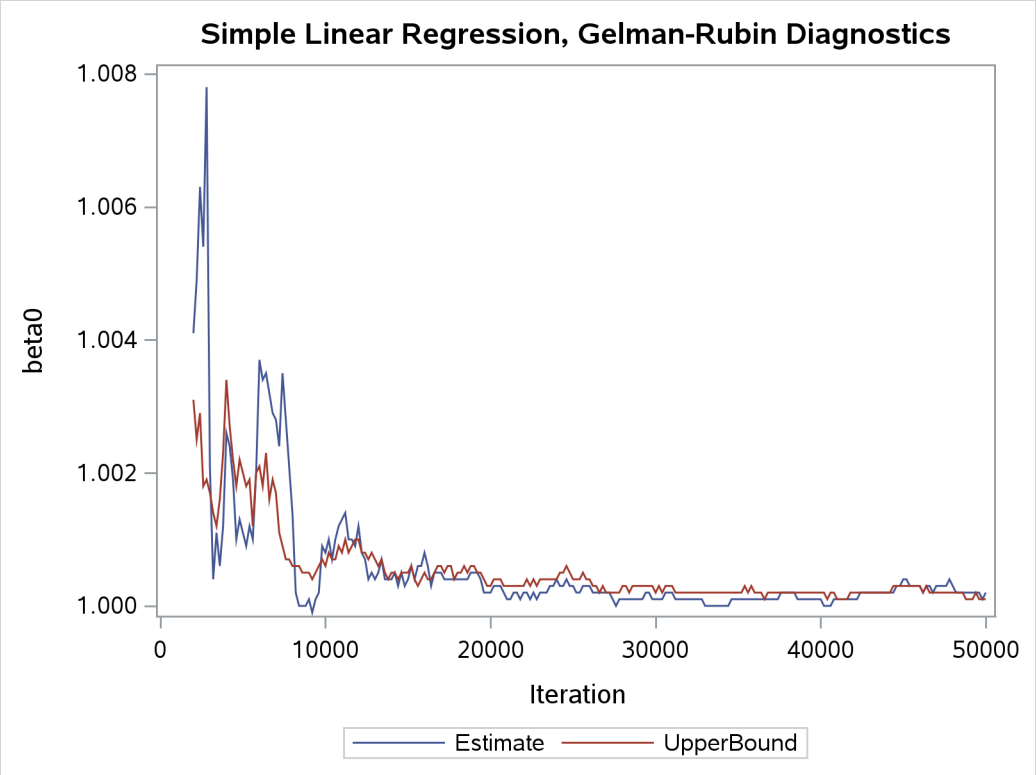

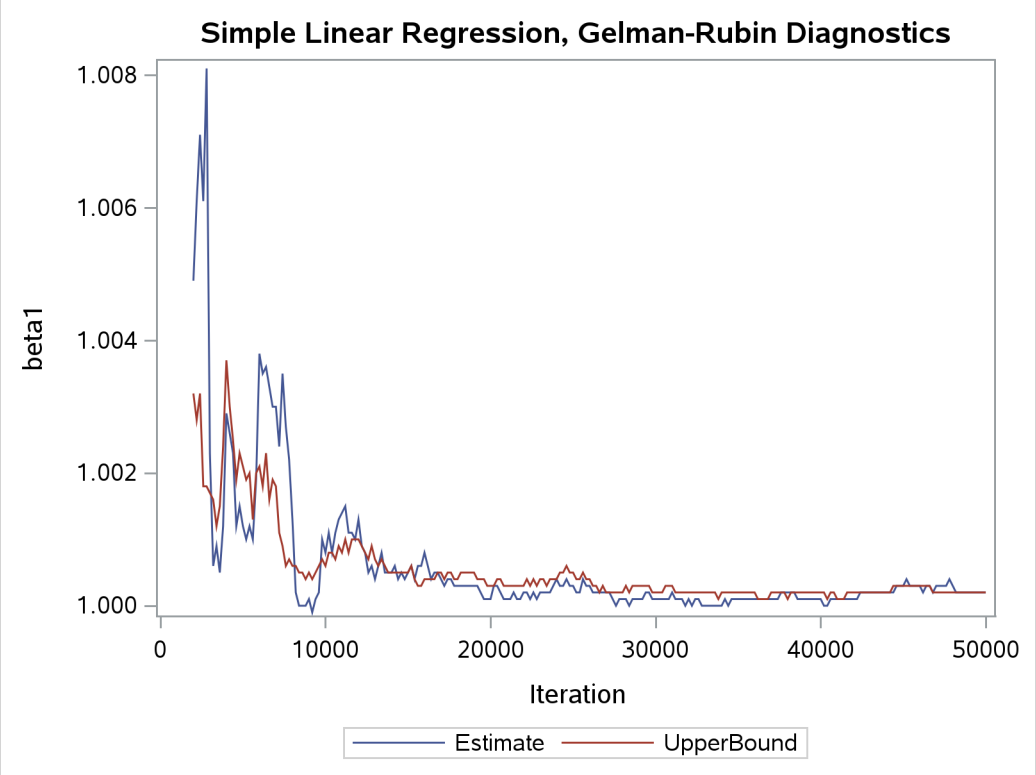

Output 80.21.3 *continued*

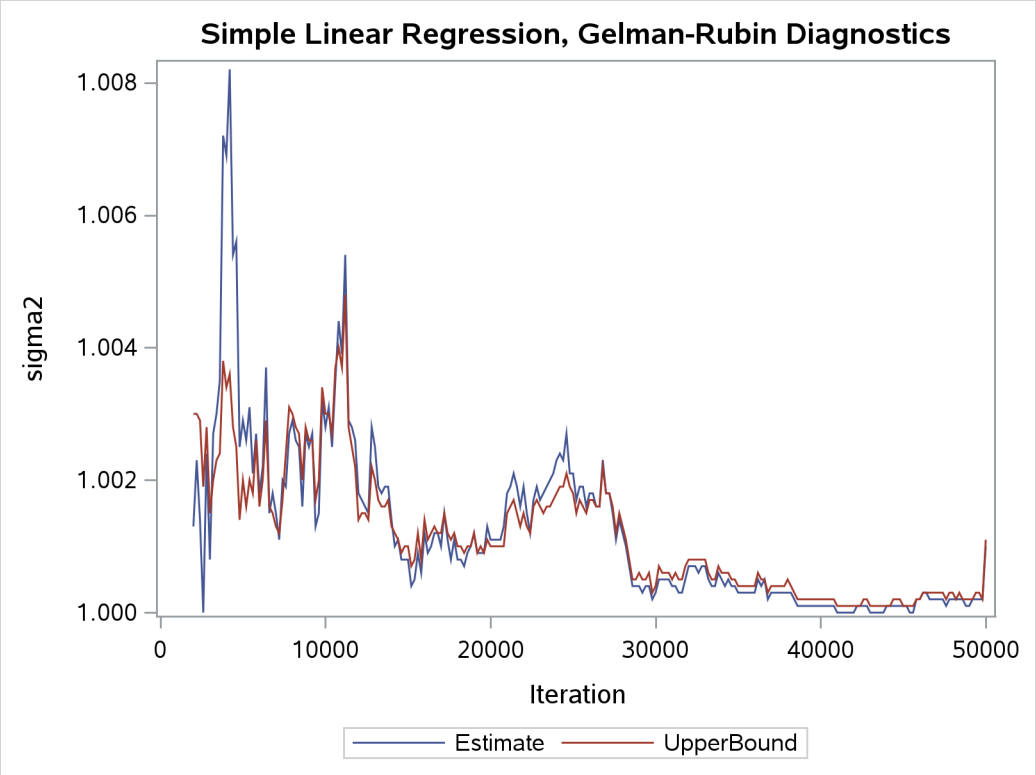

PSRF is the square root of the ratio of the between-chain variance and the within-chain variance. A large PSRF indicates that the between-chain variance is substantially greater than the within-chain variance, so that longer simulation is needed. You want the PSRF to converge to 1 eventually, as it appears to be the case in this simulation study.

Example 80.22: One-Compartment Model with Pharmacokinetic Data

(View the complete code for this example (mcmcex22.sas) in the example repository.)

A popular application of nonlinear mixed models is in the field of pharmacokinetics, which studies how a drug disperses through a living individual. This example considers the theophylline data from Pinheiro and Bates (1995). Serum concentrations of the drug theophylline, which is used to treat respiratory diseases, are measured in 12 subjects over a 25-hour period after oral administration. The data are as follows.

```
data theoph;
   input subject time conc dose;
   datalines;
 1  0.00  0.74 4.02
 1  0.25  2.84 4.02
 1  0.57  6.57 4.02
 1  1.12 10.50 4.02
 1  2.02  9.66 4.02
 1  3.82  8.58 4.02
 1  5.10  8.36 4.02
```

```
 1  7.03  7.47 4.02

   ... more lines ...

12 24.15  1.17 5.30
;
```

A commonly used one-compartment model is based on the differential equations

$$
\begin{aligned}
\frac{dA_0(t)}{dt} &= -K_a A_0(t) \\
\frac{dA(t)}{dt} &= K_a A_0(t) - K_e A(t)
\end{aligned}
$$

where K_a is the absorption rate and K_e is the elimination rate.

The initial values are

$$
\begin{aligned}
A_a(t=0) &= x \\
A(t=0) &= 0
\end{aligned}
$$

where x is the dosage.

The expected concentration of the substance in the body is computed by dividing the solution to the ODE, $A(t)$, by Cl, the clearance,

$$
\begin{aligned}
\mu_i(t) &= A_i(t)/Cl \\
y_i(t) &\sim \text{normal}(\mu_i(t), \sigma^2)
\end{aligned}
$$

where i is the subject index.

Pinheiro and Bates (1995) consider the following first-order compartment model for these data, where $\log(Cl)$ and $\log(K_a)$ are modeled using random effects to account for the patient-to-patient variability:

$$
\begin{aligned}
Cl_i &= \exp(\beta_1 + b_{i1}) \\
K_{a_i} &= \exp(\beta_2 + b_{i2}) \\
K_{e_i} &= \exp(\beta_3)
\end{aligned}
$$

Here the βs denote fixed-effects parameters and the b_is denote random-effects parameters with an unknown covariance matrix.

Although there is an analytical solution to this set of differential equations, this example illustrates the use of a general ODE solver that does not require a closed-form solution. To use the ODE solver, you want to first define the set of differential equations in PROC FCMP and store the objective function, called OneComp, in a user library:

```
proc fcmp outlib=sasuser.funcs.PK;
   subroutine OneComp(t,y[*],dy[*],ka,ke);
   outargs dy;
   dy[1] = -ka*y[1];
   dy[2] = ka*y[1]-ke*y[2];
   endsub;
run;
```

The first argument of the OneComp subroutine is the time variable in the differential equation. The second argument is the with-respect-to variable, which can be an array in case a multidimensional problem is required. The third argument is an array that stores the differential equations. This dy argument must also be the OUTARGS variable in the subroutine. The subsequent variables are additional variables, depending on the problem at hand.

In the OneComp subroutine, you can define K_a and K_e as functions of β_1, β_2, and the random-effects parameter b_2. The dy[1] and dy[2] are the differential equation, with dy[1] storing dA_a/dt and dy[2] storing dA_c/dt.

The following PROC MCMC statements use the CALL ODE subroutine to solve the set of differential equations defined in OneComp and then use that solution in the construction of the likelihood function:

```
options cmplib=sasuser.funcs;

proc mcmc data=theoph nmc=10000 seed=27 outpost=theophO diag=none
   nthreads=8;
   ods select PostSumInt;
   array b[2];
   array muB[2] (0 0);
   array cov[2,2];
   array S[2,2] (1 0 0 1);

   array init[2] dose 0;
   array sol[2];

   parms beta1 -3.22 beta2 0.47 beta3 -2.45 ;
   parms cov {0.03 0 0 0.4};
   parms s2y;

   prior beta: ~ normal(0, sd=100);
   prior cov ~ iwish(2, S);
   prior s2y  ~ igamma(shape=3, scale=2);

   random b ~ mvn(muB, cov) subject=subject;
   cl   = exp(beta1 + b1);
   ka   = exp(beta2 + b2);
   ke   = exp(beta3);
   v    = cl/ke;
   call ode('OneComp',sol,init,0,time,ka,ke);
   mu   = (sol[2]/v);
   model conc ~ normal(mu,var=s2y);
run;
```

The INIT array stores the initial values of the two differential equations, with $A_a(t = 0) =$ dose and $A(t = 0) = 0$. The array is used as an input argument to the CALL ODE subroutine.

The RANDOM statement specifies two-dimensional random effects, b, with a multivariate normal prior. The first random effect, b1, enters the model through the clearance variable, cl. The second random effect, b2, is part of the differential equations. The CALL ODE subroutine solves the OneComp set of differential equations and returns the solution to the SOL array. The second array element, SOL[2], is the solution to $A_i(t)$ for every subject i at every time t.

Posterior summary statistics are reported in Output 80.22.1.

Output 80.22.1 Posterior Summary Statistics

The MCMC Procedure

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
beta1	10000	-3.1989	0.0780	-3.3553	-3.0609
beta2	10000	0.4244	0.1915	0.00824	0.7664
beta3	10000	-2.4589	0.0495	-2.5647	-2.3702
cov1	10000	0.1306	0.0634	0.0449	0.2477
cov2	10000	-0.00222	0.0952	-0.1874	0.1888
cov3	10000	-0.00222	0.0952	-0.1874	0.1888
cov4	10000	0.6230	0.3295	0.1778	1.2376
s2y	10000	0.5231	0.0720	0.3888	0.6639

In this problem, the closed-form solution of the ODE is known:

$$C_{it} = \frac{Dk_{e_i}k_{a_i}}{Cl_i(k_{a_i} - k_{e_i})}[\exp(-k_{e_i}t) - \exp(-k_{a_i}t)] + e_{it}$$

You can manually enter the equation in PROC MCMC and use the following program to fit the same model:

```
proc mcmc data=theoph nmc=10000 seed=22 outpost=theophC;
   array b[2];

   array mu[2] (0 0);
   array cov[2,2];
   array S[2,2] (1 0 0 1);

   parms beta1 -3.22 beta2 0.47 beta3 -2.45 ;
   parms cov {0.03 0 0 0.4};
   parms s2y;

   prior beta: ~ normal(0, sd=100);
   prior cov ~ iwish(2, S);
   prior s2y  ~ igamma(shape=3, scale=2);

   random b ~ mvn(mu, cov) subject=subject;
   cl   = exp(beta1 + b1);
   ka   = exp(beta2 + b2);
   ke   = exp(beta3);
```

```
    pred = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
    model conc ~ normal(pred,var=s2y);
run;
```

Because this program makes it unnecessary to numerically solve the ODE at every observation and every iteration, it runs much faster than the program that uses the CALL ODE subroutine. But few pharmacokinetic models have known solutions that enable you to do that.

References

Abuhelwa, A. Y., Foster, D. J., and Upton, R. N. (2015). "ADVAN-Style Analytical Solutions for Common Pharmacokinetic Models." *Journal of Pharmacological and Toxicological Methods* 73:42–48.

Aitkin, M., Anderson, D. A., Francis, B., and Hinde, J. (1989). *Statistical Modelling in GLIM*. Oxford: Oxford Science Publications.

Atkinson, A. C. (1979). "The Computer Generation of Poisson Random Variables." *Journal of the Royal Statistical Society, Series C* 28:29–35.

Atkinson, A. C., and Whittaker, J. (1976). "A Switching Algorithm for the Generation of Beta Random Variables with at Least One Parameter Less Than One." *Journal of the Royal Statistical Society, Series A* 139:462–467.

Bacon, D. W., and Watts, D. G. (1971). "Estimating the Transition between Two Intersecting Straight Lines." *Biometrika* 58:525–534.

Banerjee, S., Carlin, B. P., and Gelfand, A. E. (2004). *Hierarchical Modeling and Analysis for Spatial Data*. Boca Raton, FL: Chapman & Hall/CRC.

Beal, S. L., Sheiner, L. B., Boeckmann, A. J., and Bauer, R. J., eds. (2011). *NONMEM User's Guides (1989–2011)*. Hanover, MD: Icon Development Solutions.

Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis*. 2nd ed. New York: Springer-Verlag.

Box, G. E. P., and Cox, D. R. (1964). "An Analysis of Transformations." *Journal of the Royal Statistical Society, Series B* 26:211–234.

Byrne, G. D., and Hindmarsh, A. C. (1975). "A Polyalgorithm for the Numerical Solution of Ordinary Differential Equations." *ACM Transactions on Mathematical Software* 1:71–96. http://portal.acm.org/citation.cfm?doid=355626.355636.

Carlin, B. P., Gelfand, A. E., and Smith, A. F. M. (1992). "Hierarchical Bayesian Analysis of Changepoint Problems." *Journal of the Royal Statistical Society, Series C* 41:389–405.

Chaloner, K. (1994). "Residual Analysis and Outliers in Bayesian Hierarchical Models." In *Aspects of Uncertainty: A Tribute to D. V. Lindley*, 149–157. New York: John Wiley & Sons.

Chaloner, K., and Brant, R. (1988). "A Bayesian Approach to Outlier Detection and Residual Analysis." *Biometrika* 75:651–659.

Cheng, R. C. H. (1978). "Generating Beta Variates with Non-integral Shape Parameters." *Communications ACM* 28:290–295.

Clayton, D. G. (1991). "A Monte Carlo Method for Bayesian Inference in Frailty Models." *Biometrics* 47:467–485.

Congdon, P. (2003). *Applied Bayesian Modeling*. New York: John Wiley & Sons.

Crowder, M. J. (1978). "Beta-Binomial Anova for Proportions." *Journal of the Royal Statistical Society, Series C* 27:34–37.

Draper, D. (1996). "Discussion of the Paper by Lee and Nelder." *Journal of the Royal Statistical Society, Series B* 58:662–663.

Eilers, P. H. C., and Marx, B. D. (1996). "Flexible Smoothing with *B*-Splines and Penalties." *Statistical Science* 11:89–121. With discussion.

Finney, D. J. (1947). "The Estimation from Individual Records of the Relationship between Dose and Quantal Response." *Biometrika* 34:320–334.

Fisher, D., and Shafer, S. (2007). "Pharmacokinetic and Pharmacodynamic Analysis with NONMEM: Basic Concepts." Course materials for Fisher/Shafer NONMEM Workshop, March 7–11, Ghent, Belgium.

Fisher, R. A. (1935). "The Fiducial Argument in Statistical Inference." *Annals of Eugenics* 6:391–398.

Fishman, G. S. (1996). *Monte Carlo: Concepts, Algorithms, and Applications*. New York: John Wiley & Sons.

Gabrielsson, J., and Weiner, D. (2006). *Pharmacokinetic and Pharmacodynamic Data Analysis: Concepts and Applications*. 4th ed. Stockholm: Swedish Pharmaceutical Press.

Gaver, D. P., and O'Muircheartaigh, I. G. (1987). "Robust Empirical Bayes Analysis of Event Rates." *Technometrics* 29:1–15.

Gelfand, A. E., Hills, S. E., Racine-Poon, A., and Smith, A. F. M. (1990). "Illustration of Bayesian Inference in Normal Data Models Using Gibbs Sampling." *Journal of the American Statistical Association* 85:972–985.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. 2nd ed. London: Chapman & Hall.

Gentleman, R., and Geyer, C. J. (1994). "Maximum Likelihood for Interval Censored Data: Consistency and Computation." *Biometrika* 81:618–623.

Gilks, W. R. (2003). "Adaptive Metropolis Rejection Sampling (ARMS)." Software from MRC Biostatistics Unit, Cambridge, UK. `http://www.maths.leeds.ac.uk/~wally.gilks/adaptive.rejection/web_page/Welcome.html`.

Gilks, W. R., and Wild, P. (1992). "Adaptive Rejection Sampling for Gibbs Sampling." *Journal of the Royal Statistical Society, Series C* 41:337–348.

Holmes, C. C., and Held, L. (2006). "Bayesian Auxiliary Variable Models for Binary and Multinomial Regression." *Bayesian Analysis* 1:145–168.

Ibrahim, J. G., Chen, M.-H., and Lipsitz, S. R. (2001). "Missing Responses in Generalised Linear Mixed Models When the Missing Data Mechanism Is Nonignorable." *Biometrika* 88:551–564.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998). "Markov Chain Monte Carlo in Practice: A Roundtable Discussion." *American Statistician* 52:93–100.

Krall, J. M., Uthoff, V. A., and Harley, J. B. (1975). "A Step-Up Procedure for Selecting Variables Associated with Survival." *Biometrics* 31:49–57.

Kuhfeld, W. F. (2010). *Conjoint Analysis*. Technical report, SAS Institute Inc., Cary, NC. http://support.sas.com/resources/papers/tnote/tnote_marketresearch.html.

Kurada, R. R., and Chen, F. (2018). "Fitting Compartment Models Using PROC NLMIXED." In *Proceedings of the SAS Global Forum 2018 Conference*. Cary, NC: SAS Institute Inc. https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2018/1883-2018.pdf.

Lin, D. Y. (1994). "Cox Regression Analysis of Multivariate Failure Time Data: The Marginal Approach." *Statistics in Medicine* 13:2233–2247.

Little, R. J. A., and Rubin, D. B. (2002). *Statistical Analysis with Missing Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Matsumoto, M., and Kurita, Y. (1992). "Twisted GFSR Generators." *ACM Transactions on Modeling and Computer Simulation* 2:179–194.

Matsumoto, M., and Kurita, Y. (1994). "Twisted GFSR Generators II." *ACM Transactions on Modeling and Computer Simulation* 4:254–266.

Matsumoto, M., and Nishimura, T. (1998). "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-random Number Generator." *ACM Transactions on Modeling and Computer Simulation* 8:3–30.

McGrath, E. J., and Irving, D. C. (1973). *Techniques for Efficient Monte Carlo Simulation, Vol. 2: Random Number Generation for Selected Probability Distributions*. Technical report, Science Applications Inc., La Jolla, CA.

Michael, J. R., Schucany, W. R., and Haas, R. W. (1976). "Generating Random Variates Using Transformations with Multiple Roots." *American Statistician* 30:88–90.

Owen, J. S., and Fiedler-Kelly, J. (2014). *Introduction to Population Pharmacokinetic/Pharmacodynamic Analysis with Nonlinear Mixed Effects Models*. Hoboken, NJ: John Wiley & Sons.

Pinheiro, J. C., and Bates, D. M. (1995). "Approximations to the Log-Likelihood Function in the Nonlinear Mixed-Effects Model." *Journal of Computational and Graphical Statistics* 4:12–35.

Pregibon, D. (1981). "Logistic Regression Diagnostics." *Annals of Statistics* 9:705–724.

Ralston, A., and Rabinowitz, P. (1978). *A First Course in Numerical Analysis*. New York: McGraw-Hill.

Rice, S. O. (1973). "Efficient Evaluation of Integrals of Analytic Functions by the Trapezoidal Rule." *Bell System Technical Journal* 52:707–722.

Ripley, B. D. (1987). *Stochastic Simulation*. New York: John Wiley & Sons.

Robert, C. P. (1995). "Simulation of Truncated Normal Variables." *Statistics and Computing* 5:121–125.

Roberts, G. O., Gelman, A., and Gilks, W. R. (1997). "Weak Convergence and Optimal Scaling of Random Walk Metropolis Algorithms." *Annals of Applied Probability* 7:110–120.

Roberts, G. O., and Rosenthal, J. S. (2001). "Optimal Scaling for Various Metropolis-Hastings Algorithms." *Statistical Science* 16:351–367.

Rubin, D. B. (1976). "Inference and Missing Data." *Biometrika* 63:581–592.

Rubin, D. B. (1981). "Estimation in Parallel Randomized Experiments." *Journal of Educational Statistics* 6:377–411.

Schervish, M. J. (1995). *Theory of Statistics*. New York: Springer-Verlag.

Sharples, L. (1990). "Identification and Accommodation of Outliers in General Hierarchical Models." *Biometrika* 77:445–453.

Sikorsky, K. (1982). "Optimal Quadrature Algorithms in H_P Spaces." *Numerische Mathematik* 39:405–410.

Sikorsky, K., and Stenger, F. (1984). "Optimal Quadratures in H_P Spaces." *ACM Transactions on Mathematical Software* 3:140–151.

Spiegelhalter, D. J., Thomas, A., Best, N. G., and Gilks, W. R. (1996a). "BUGS Examples, Volume 1." Version 0.5 (version ii).

Spiegelhalter, D. J., Thomas, A., Best, N. G., and Gilks, W. R. (1996b). "BUGS Examples, Volume 2." Version 0.5 (version ii).

Squire, W. (1987). "Comparison of Gauss-Hermite and Midpoint Quadrature with Application to the Voigt Function." In *Numerical Integration: Recent Developments*, edited by P. Keast and G. Fairweather, 111–112. Dordrecht, Netherlands: D. Reidel Publishing.

Stenger, F. (1973a). "Integration Formulas Based on the Trapezoidal Formula." *Journal of the Institute of Mathematics and Its Applications* 12:103–114.

Stenger, F. (1973b). "Remarks on Integration Formulas Based on the Trapezoidal Formula." *Journal of the Institute of Mathematics and Its Applications* 19:145–147.

Thomas, A., Best, N. G., Lunn, D., Arnold, R., and Spiegelhalter, D. (2004). "GeoBUGS User Manual." http://www.mrc-bsu.cam.ac.uk/wp-content/uploads/geobugs12manual.pdf.

Chapter 81
The MDS Procedure

Contents

Overview: MDS Procedure **6512**
Getting Started: MDS Procedure **6513**
Syntax: MDS Procedure **6516**
PROC MDS Statement 6517
BY Statement 6527
ID Statement 6528
INVAR Statement 6528
MATRIX Statement 6528
VAR Statement 6529
WEIGHT Statement 6529
Details: MDS Procedure **6529**
Formulas 6529
OUT= Data Set 6531
OUTFIT= Data Set 6532
OUTRES= Data Set 6533
INITIAL= Data Set 6534
Missing Values 6534
Normalization of the Estimates 6534
Comparison with Earlier Procedures 6535
Displayed Output 6535
ODS Table Names 6537
ODS Graphics 6537
Example: MDS Procedure **6538**
Example 81.1: Jacobowitz Body Parts Data from Children and Adults 6538
References **6548**

Overview: MDS Procedure

Multidimensional scaling (MDS) refers to a class of methods. These methods estimate coordinates for a set of objects in a space of specified dimensionality. The input data are measurements of distances between pairs of objects. A variety of models can be used that include different ways of computing distances and various functions relating the distances to the actual data. The MDS procedure fits two- and three-way, metric and nonmetric multidimensional scaling models.

The data for the MDS procedure consist of one or more square symmetric or asymmetric matrices of similarities or dissimilarities between *objects* or *stimuli* (Kruskal and Wish 1978, pp. 7–11). Such data are also called *proximity* data. In psychometric applications, each matrix typically corresponds to a *subject*, and models that fit different parameters for each subject are called *individual difference* models.

Missing values are permitted. In particular, if the data are all missing except within some off-diagonal rectangle, the analysis is called *unfolding*. There are, however, many difficulties intrinsic to unfolding models (Heiser 1981). PROC MDS does not perform external unfolding; for analyses requiring external unfolding, use the TRANSREG procedure instead.

The MDS procedure estimates the following parameters by nonlinear least squares:

configuration	the coordinates of each object in a Euclidean (Kruskal and Wish 1978, pp. 17–19) or weighted Euclidean space (Kruskal and Wish 1978, pp. 61–63) of one or more dimensions
dimension coefficients	for each data matrix, the coefficients that multiply each coordinate of the *common* or *group* weighted Euclidean space to yield the *individual* unweighted Euclidean space. These coefficients are the square roots of the *subject weights* (Kruskal and Wish 1978, pp. 61–63). A plot of the dimension coefficients is directly interpretable in that it shows how a unit square in the group space is transformed to a rectangle in each individual space. A plot of subject weights has no such simple interpretation. The weighted Euclidean model is related to the INDSCAL model (Carroll and Chang 1970).
transformation parameters	intercept, slope, or exponent in a linear, affine, or power transformation relating the distances to the data (Kruskal and Wish 1978, pp. 19–22). For a nonmetric analysis, monotone transformations involving no explicit parameters are used (Kruskal and Wish 1978, pp. 22–25). For a discussion of metric versus nonmetric transformations, see Kruskal and Wish (1978, pp. 76–78).

Depending on the LEVEL= option, PROC MDS fits either a regression model of the form

$$fit(datum) = fit(trans(distance)) + error$$

or a measurement model of the form

$$fit(trans(datum)) = fit(distance) + error$$

where

fit is a predetermined power or logarithmic transformation specified by the FIT= option.

trans is an estimated ("optimal") linear, affine, power, or monotone transformation specified by the LEVEL= option.

datum is a measure of the similarity or dissimilarity of two objects or stimuli.

distance is a distance computed from the estimated coordinates of the two objects and estimated dimension coefficients in a space of one or more dimensions. If there are no dimension coefficients (COEF=IDENTITY), this is an unweighted Euclidean distance. If dimension coefficients are used (COEF=DIAGONAL), this is a weighted Euclidean distance where the weights are the squares of the dimension coefficients; alternatively, you can multiply each dimension by its coefficient and compute an unweighted Euclidean distance.

error is an error term assumed to have an approximately normal distribution and to be independently and identically distributed for all data. Under these assumptions, least-squares estimation is statistically appropriate.

For an introduction to multidimensional scaling, see Kruskal and Wish (1978) and Arabie, Carroll, and DeSarbo (1987). A more advanced treatment is given by Young (1987). Many practical issues of data collection and analysis are discussed in Schiffman, Reynolds, and Young (1981). The fundamentals of psychological measurement, including both unidimensional and multidimensional scaling, are expounded by Torgerson (1958). Nonlinear least-squares estimation of PROC MDS models is discussed in Null and Sarle (1982).

Getting Started: MDS Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mdsgs.sas.)

The simplest application of PROC MDS is to reconstruct a map from a table of distances between points on the map (Kruskal and Wish 1978, pp. 7–9). A data set containing a table of flying mileages between 10 U.S. cities is available in the Sashelp library.

Since the flying mileages are very good approximations to Euclidean distance, no transformation is needed to convert distances from the model to data. The analysis can therefore be done at the absolute level of measurement, as displayed in the following PROC MDS step (LEVEL=ABSOLUTE). The following statements produce Figure 81.1 and Figure 81.2:

```
title 'Analysis of Flying Mileages between Ten U.S. Cities';

ods graphics on;

proc mds data=sashelp.mileages level=absolute;
```

```
    id city;
run;
```

PROC MDS first displays the iteration history. In this example, only one iteration is required. The badness-of-fit criterion 0.001689 indicates that the data fit the model extremely well. You can also see that the fit is excellent in the fit plot in Figure 81.2.

Figure 81.1 Iteration History from PROC MDS

Analysis of Flying Mileages between Ten U.S. Cities

Multidimensional Scaling: Data=SASHELP.MILEAGES.DATA Shape=TRIANGLE Condition=MATRIX Level=ABSOLUTE Coef=IDENTITY Dimension=2 Formula=1 Fit=1

Gconverge=0.01 Maxiter=100 Over=1 Ridge=0.0001

Iteration	Type	Badness-of-Fit Criterion	Change in Criterion	Convergence Measure
0	Initial	0.003273	.	0.8562
1	Lev-Mar	0.001689	0.001584	0.005128

Convergence criterion is satisfied.

While PROC MDS can recover the relative positions of the cities, it cannot determine absolute location or orientation. In this case, north is toward the bottom of the plot. (See the first plot in Figure 81.2.)

Figure 81.2 Plot of Estimated Configuration and Fit

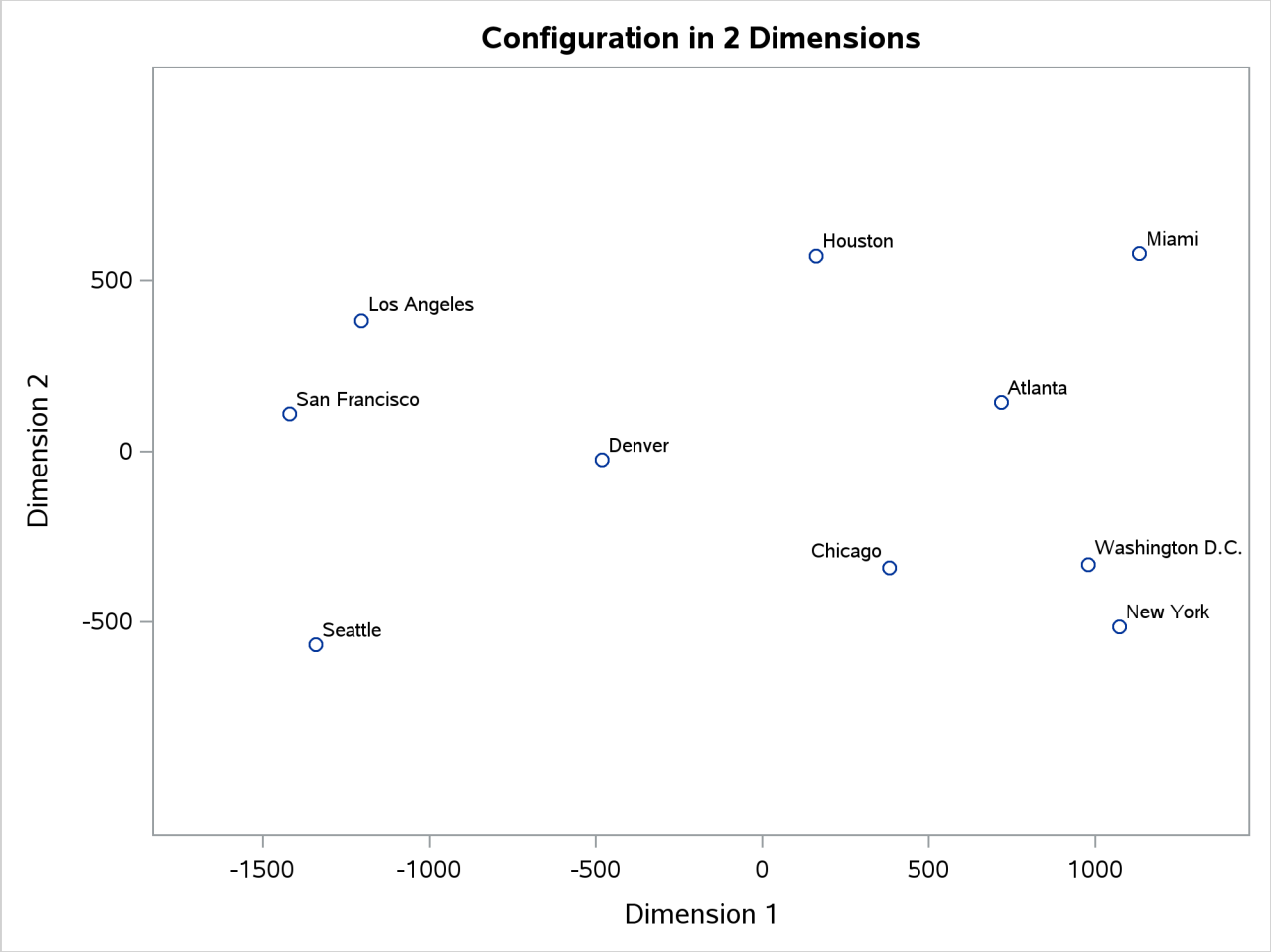

Figure 81.2 *continued*

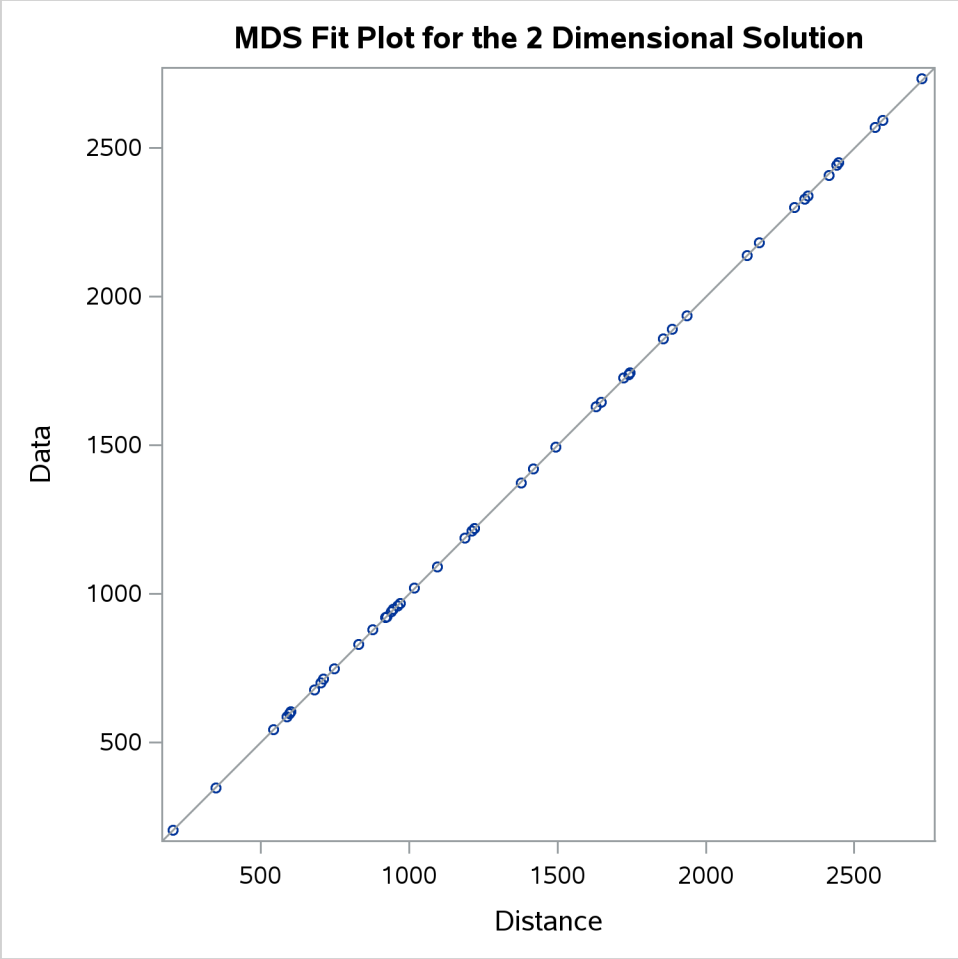

Syntax: MDS Procedure

The following statements are available in the MDS procedure:

PROC MDS < *options* > **;**
  VAR *variables* **;**
  INVAR *variables* **;**
  ID | **OBJECT** *variable* **;**
  MATRIX | **SUBJECT** *variable* **;**
  WEIGHT *variables* **;**
  BY *variables* **;**

The PROC MDS statement is required. All other statements are optional.

PROC MDS Statement

PROC MDS < *options* > **;**

The PROC MDS statement invokes the MDS procedure. PROC MDS produces an iteration history by default. Graphical displays are produced when ODS Graphics is enabled. Additional displayed output is controlled by the interaction of the PCONFIG, PCOEF, PTRANS, PFIT, and PFITROW options with the PININ, PINIT, PITER, and PFINAL options. The PCONFIG, PCOEF, PTRANS, PFIT, and PFITROW options specify *which* estimates and fit statistics are to be displayed. The PININ, PINIT, PITER, and PFINAL options specify *when* the estimates and fit statistics are to be displayed. If you specify at least one of the PCONFIG, PCOEF, PTRANS, PFIT, and PFITROW options but none of the PININ, PINIT, PITER, and PFINAL options, the final results (PFINAL) are displayed. If you specify at least one of the PININ, PINIT, PITER, and PFINAL options but none of the PCONFIG, PCOEF, PTRANS, PFIT, and PFITROW options, all estimates (PCONFIG, PCOEF, PTRANS) and the fit statistics for each matrix and for the entire sample (PFIT) are displayed. If you do not specify any of these nine options, no estimates or fit statistics are displayed (except the badness-of-fit criterion in the iteration history).

The types of estimates written to the OUT= data set are determined by the OCONFIG, OCOEF, OTRANS, and OCRIT options. If you do not specify any of these four options, the estimates of all the parameters of the PROC MDS model and the value of the badness-of-fit criterion appear in the OUT= data set. If you specify one or more of these *options*, only the information requested by the specified options appears in the OUT= data set. Also, the OITER option causes these statistics to be written to the OUT= data set after initialization and on each iteration, as well as after the iterations have terminated.

Table 81.2 summarizes the *options* available in the PROC MDS statement.

Table 81.2 Summary of PROC MDS Statement Options

Option	Description
Data Set Options	
DATA=	Specifies the input SAS data set
INITIAL=	Specifies the input SAS data set containing initial values
OUT=	Specifies the output data set
OUTFIT=	Specifies the output fit data set
OUTRES=	Specifies the output residual data set
Input Control	
CUTOFF=	Replaces data values with missing values
SHAPE=	Specifies the shape of the input data matrices
SIMILAR=	Specifies that the data are similarity measurements
Model	
COEF=	Specifies the type of matrix for the coefficients
CONDITION=	Specifies the conditionality of the data
DIMENSION=	Specifies the number of dimensions
LEVEL=	Specifies the measurement level
NEGATIVE	Permits slopes or powers to be negative
UNTIE	Permits tied data to be untied

Table 81.2 *continued*

Option	Description
Initialization	
INAV=	Affects the computation of initial coordinates
NOULB	Specifies the missing data initialization
RANDOM=	Specifies initial random coordinates
Estimation	
ALTERNATE=	Specifies the alternating-least-squares algorithm
CONVERGE=	Specifies the convergence criterion
EPSILON=	Specifies the amount added to squared distances
FIT=	Specifies a predetermined transformation
FORMULA=	Specifies the badness-of-fit formula
GCONVERGE=	Specifies the gradient convergence criterion
MAXITER=	Specifies the maximum number of iterations
MCONVERGE=	Specifies the monotone convergence criterion
MINCRIT=	Specifies the minimum badness-of-fit criterion
NONORM	Suppresses normalization of the initial and final estimates
OVER=	Specifies the maximum overrelaxation factor
RIDGE=	Specifies the initial ridge value
SINGULAR=	Specifies the singularity criterion
Control Output Data Set Contents	
OCOEF	Writes the dimension coefficients to the OUT= data set
OCONFIG	Writes the coordinates of the objects to the OUT= data set
OCRIT	Writes the badness-of-fit criterion to the OUT= data set
OITER	Writes current values after initialization and on every iteration
OTRANS	Writes the transformation parameter estimates to the OUT= data set
Control Displayed Output	
DECIMALS=	Specifies how many decimal places to use
NOPHIST	Suppresses the iteration history
PCOEF	Displays the estimated dimension coefficients
PCONFIG	Displays the estimated coordinates
PDATA	Displays each data matrix
PFINAL	Displays final estimates
PFIT	Displays the badness-of-fit criterion
PFITROW	Displays the badness-of-fit criterion for each row
PINAVDATA	Displays INAV= data set information
PINEIGVAL	Displays the initial eigenvalues
PINEIGVEC	Displays the initial eigenvectors
PININ	Displays values read from the INITIAL= data set
PINIT	Displays initial values
PITER	Displays estimates on each iteration
PLOTS=	Controls the graphical displays
PTRANS	Displays the estimated transformation parameters

ALTERNATE=NONE | NO | N
ALTERNATE=MATRIX | MAT | M | SUBJECT | SUB | S
ALTERNATE=ROW | R < *=n* >
ALT=N | M| S | R < *=n* >

determines what form of alternating-least-squares algorithm is used. The default depends on the amount of memory available. The following ALTERNATE= options are listed in order of decreasing memory requirements:

NONE causes all parameters to be adjusted simultaneously on each iteration. This option is usually best for a small number of subjects and objects.

MATRIX adjusts all the parameters for the first subject, then all the parameters for the second subject, and so on, and finally adjusts all parameters that do not correspond to a subject, such as coordinates and unconditional transformations. This option usually works best for a large number of subjects with a small number of objects.

ROW treats subject parameters the same way as the ALTERNATE=MATRIX option but also includes separate stages for unconditional parameters and for subsets of the objects. The ALT=ROW option usually works best for a large number of objects. Specifying ALT=ROW=*n* divides the objects into subsets of *n* objects each, except possibly for one subset when *n* does not divide the number of objects evenly. If you omit =*n*, the number of objects in the subsets is determined from the amount of memory available. The smaller the value of *n*, the less memory is required.

When you specify the LEVEL=ORDINAL option, the monotone transformation is always computed in a separate stage and is listed as a separate iteration in the iteration history. In this case, estimation is done by iteratively reweighted least squares. The weights are recomputed according to the FORMULA= option on each monotone iteration; hence, it is possible for the badness-of-fit criterion to increase after a monotone iteration.

COEF=IDENTITY | IDEN | I
COEF=DIAGONAL | DIAG | D

specifies the type of matrix for the dimension coefficients.

IDENTITY is the default, which yields Euclidean distances.

DIAGONAL produces weighted Euclidean distances, in which each subject can have different weights for the dimensions. The dimension coefficients that PROC MDS outputs are related to the square roots of what are called subject weights in PROC ALSCAL; the normalization in PROC MDS also differs from that in PROC ALSCAL. The weighted Euclidean model is related to the INDSCAL model (Carroll and Chang 1970).

CONDITION=UN | U
CONDITION=MATRIX | MAT | M | SUBJECT | SUB | S
CONDITION=ROW | R
COND=U | M | S | R

specifies the conditionality of the data (Young 1987, pp. 60–63). The data are divided into disjoint subsets called *partitions*. Within each partition, a separate transformation is applied, as specified by the LEVEL= option. The three types of conditionality are as follows:

UN (unconditional) puts all the data into a single partition.

MATRIX (matrix conditional) makes each data matrix a partition.

ROW (row conditional) makes each row of each data matrix a partition.

The default is CONDITION=MATRIX. The CONDITION= option also determines the default value for the SHAPE= option. If you specify the CONDITION=ROW option and omit the SHAPE= option, each data matrix is stored as a square and possibly asymmetric matrix. If you specify the CONDITION=UN or CONDITION=MATRIX option and omit the SHAPE= option, only one triangle is stored. See the SHAPE= option for details.

CONVERGE | CONV=*p*

sets both the gradient convergence criterion and the monotone convergence criterion to p, where $0 \leq p \leq 1$. The default is CONVERGE=0.01; smaller values might greatly increase the number of iterations required. Values less than 0.0001 might be impossible to satisfy because of the limits of machine precision. (See the GCONVERGE= and MCONVERGE= options.)

CUTOFF=*n*

replaces data values less than n with missing values. The default is CUTOFF=0.

DATA=*SAS-data-set*

specifies the SAS data set containing one or more square matrices to be analyzed. In typical psychometric data, each matrix contains judgments from one subject, so there is a one-to-one correspondence between data matrices and subjects.

The data matrices contain similarity or dissimilarity measurements to be modeled and, optionally, weights for these data. The data are generally assumed to be dissimilarities unless you use the SIMILAR option. However, if there are nonmissing diagonal values and these values are predominantly larger than the off-diagonal values, the data are assumed to be similarities and are treated as if the SIMILAR option is specified. The diagonal elements are not otherwise used in fitting the model.

Each matrix must have exactly the same number of observations as the number of variables specified by the VAR statement or determined by defaults. This number is the number of objects or stimuli.

The first observation and variable are assumed to contain data for the first object, the second observation and variable are assumed to contain data for the second object, and so on.

When there are two or more matrices, the observations in each matrix must correspond to the same objects in the same order as in the first matrix.

The matrices can be symmetric or asymmetric, as specified by the SHAPE= option.

DECIMALS | DEC=*n*

specifies how many decimal places to use when displaying the parameter estimates and fit statistics. The default is DECIMALS=2, which is generally reasonable except in conjunction with the LEVEL=ABSOLUTE option and very large or very small data.

DIMENSION | DIMENS | DIM=*n* **<TO** *m* **< BY=***i* **>>**

specifies the number of dimensions to use in the MDS model, where $1 \leq n, m <$ *number of objects*. The parameter i can be either positive or negative but not zero. If you specify different values for n and m, a separate model is fitted for each requested dimension. If you specify only DIMENSION=n,

then only *n* dimensions are fitted. The default is DIMENSION=2 if there are three or more objects; otherwise, DIMENSION=1 is the only valid specification. The analyses for each number of dimensions are done independently. For information about choosing the dimensionality, see Kruskal and Wish (1978 , pp. 48–60).

EPSILON | EPS=*n*

specifies a number *n*, $0 < n < 1$, that determines the amount added to squared distances computed from the model to avoid numerical problems such as division by 0. This amount is computed as ϵ equal to *n* times the mean squared distance in the initial configuration. The distance in the MDS model is thus computed as

$$distance = \sqrt{sqdist + \epsilon}$$

where *sqdist* is the squared Euclidean distance or the weighted squared Euclidean distance.

The default is EPSILON=1E–12, which is small enough to have no practical effect on the estimates unless the FIT= value is nonpositive and there are dissimilarities that are very close to 0. Hence, when the FIT= value is nonpositive, dissimilarities less than *n* times 100 times the maximum dissimilarity are not permitted.

FIT=DISTANCE | DIS | D

FIT=SQUARED | SQU | S

FIT=LOG | L

FIT=*n*

specifies a predetermined (not estimated) transformation to apply to both sides of the MDS model before the error term is added.

The default is FIT=DISTANCE or, equivalently, FIT=1, which fits data to distances.

The option FIT=SQUARED or FIT=2 fits squared data to squared distances. This gives greater importance to large data and distances and lesser importance to small data and distances in fitting the model.

The FIT=LOG or FIT=0 option fits log data to log distances. This gives lesser importance to large data and distances and greater importance to small data and distances in fitting the model.

In general, the FIT=*n* option fits *n*th-power data to *n*th-power distances. Values of *n* that are large in absolute value can cause floating-point overflows.

If the FIT= value is 0 or negative, the data must be strictly positive (see the EPSILON= option). Negative data might produce strange results with any value other than FIT=1.

FORMULA=*0* **| OLS | O**

FORMULA=*1* **| USS | U**

FORMULA=*2* **| CSS | C**

FOR=O | U | C

determines how the badness-of-fit criterion is standardized in correspondence with stress formulas 1 and 2 (Kruskal and Wish 1978, pp. 24–26). The default is FORMULA=1 unless you specify FIT=LOG, in which case the default is FORMULA=2. Data partitions are defined by the CONDITION= option.

0 fits a regression model by ordinary least squares (Null and Sarle 1982) without standardizing the partitions; this option cannot be used with the LEVEL=ORDINAL option. The badness-of-fit criterion is the square root of the error sum of squares.

1 standardizes each partition by the uncorrected sum of squares of the (possibly transformed) data; this option should not be used with the FIT=LOG option. With the FIT=DISTANCE and LEVEL=ORDINAL options, this is equivalent to Kruskal's stress formula 1 or an obvious generalization thereof. With the FIT=SQUARED and LEVEL=ORDINAL options, this is equivalent to Young's s-stress formula 1 or an obvious generalization thereof. The badness-of-fit criterion is analogous to $\sqrt{1-\mathrm{R}^2}$, where R is a multiple correlation about the origin.

2 standardizes each partition by the corrected sum of squares of the (possibly transformed) data; this option is the recommended method for unfolding. With the FIT=DISTANCE and LEVEL=ORDINAL options, this is equivalent to Kruskal's stress formula 2 or an obvious generalization thereof. With the FIT=SQUARED and LEVEL=ORDINAL options, this is equivalent to Young's s-stress formula 2 or an obvious generalization thereof. The badness-of-fit criterion is analogous to $\sqrt{1-\mathrm{R}^2}$, where R is a multiple correlation computed with a denominator corrected for the mean.

GCONVERGE | GCONV=*p*

sets the gradient convergence criterion to *p*, where $0 \le p \le 1$. The default is GCONVERGE=0.01; smaller values might greatly increase the number of iterations required. Values less than 0.0001 might be impossible to satisfy because of the limits of machine precision.

The gradient convergence measure is the multiple correlation of the Jacobian matrix with the residual vector, uncorrected for the mean. (See the CONVERGE= and MCONVERGE= options.)

INAV=DATA | D

INAV=SSCP | S

affects the computation of initial coordinates. The default is INAV=DATA.

DATA computes a weighted average of the data matrices. Its value is estimated only if an element is missing from every data matrix. The weighted average of the data matrices with missing values filled in is then converted to a scalar products matrix (or what would be a scalar products matrix if the fit were perfect), from which the initial coordinates are computed.

SSCP estimates missing values in each data matrix and converts each data matrix to a scalar products matrix. The initial coordinates are computed from the unweighted average of the scalar products matrices.

INITIAL | IN=*SAS-data-set*

specifies a SAS data set containing initial values for some or all of the parameters of the MDS model. If the INITIAL= option is omitted, the initial values are computed from the data.

LEVEL=ABSOLUTE | ABS | A

LEVEL=RATIO | RAT | R

LEVEL=INTERVAL | INT | I

LEVEL=LOGINTERVAL | LOG | L

LEVEL=ORDINAL | ORD | O

specifies the measurement level of the data and hence the type of estimated (optimal) transformations

applied to the data or distances (Young 1987, pp. 57–60; Krantz et al. 1971, pp. 9–12) within each partition as specified by the CONDITION= option. LEVEL=ORDINAL specifies a nonmetric analysis, while all other LEVEL= options specify metric analyses. The default is LEVEL=ORDINAL.

ABSOLUTE permits no optimal transformations. Hence, the distinction between regression and measurement models is irrelevant.

RATIO fits a regression model in which the distances are multiplied by a slope parameter in each partition (a linear transformation). In this case, the regression model is equivalent to the measurement model with the slope parameter reciprocated.

INTERVAL fits a regression model in which the distances are multiplied by a slope parameter and added to an intercept parameter in each partition (an affine transformation). In this case, the regression and measurement models differ if there is more than one partition.

LOGINTERVAL fits a regression model in which the distances are raised to a power and multiplied by a slope parameter in each partition (a power transformation).

ORDINAL fits a measurement model in which a least-squares monotone increasing transformation is applied to the data in each partition. At the ordinal measurement level, the regression and measurement models differ.

MAXITER | ITER=*n*

specifies the maximum number of iterations, where $n \geq 0$. The default is MAXITER=100.

MCONVERGE | MCONV=*p*

sets the monotone convergence criterion to p, where $0 \leq p \leq 1$, for use with the LEVEL=ORDINAL option. The default is MCONVERGE=0.01; if you want greater precision, MCONVERGE=0.001 is usually reasonable, but smaller values might greatly increase the number of iterations required.

The monotone convergence criterion is the Euclidean norm of the change in the optimally scaled data divided by the Euclidean norm of the optimally scaled data, averaged across partitions defined by the CONDITION= option. (See the CONVERGE= and GCONVERGE= options.)

MINCRIT | CRITMIN=*n*

causes iteration to terminate when the badness-of-fit criterion is less than or equal to n, where $n \geq 0$. The default is MINCRIT=1E–6.

NEGATIVE

permits slopes or powers to be negative with the LEVEL=RATIO, INTERVAL, or LOGINTERVAL option.

NONORM

suppresses normalization of the initial and final estimates.

NOPHIST | NOPRINT | NOP

suppresses the output of the iteration history.

NOULB
causes missing data to be estimated during initialization by the average nonmissing value, where the average is computed according to the FIT= option. Otherwise, missing data are estimated by interpolating between the Rabinowitz (1976) upper and lower bounds.

OCOEF
writes the dimension coefficients to the OUT= data set. See the OUT= option for interactions with other options.

OCONFIG
writes the coordinates of the objects to the OUT= data set. See the OUT= option for interactions with other options.

OCRIT
writes the badness-of-fit criterion to the OUT= data set. See the OUT= option for interactions with other options.

OITER | OUTITER
writes current values to the output data sets after initialization and on every iteration. Otherwise, only the final values are written to any output data sets. (See the OUT=, OUTFIT=, and OUTRES= options.)

OTRANS
writes the transformation parameter estimates to the OUT= data set if any such estimates are computed. There are no transformation parameters with the LEVEL=ORDINAL option. See the OUT= option for interactions with other options.

OUT=*SAS-data-set*
creates a SAS data set containing, by default, the estimates of all the parameters of the PROC MDS model and the value of the badness-of-fit criterion. However, if you specify one or more of the OCONFIG, OCOEF, OTRANS, and OCRIT options, only the information requested by the specified options appears in the OUT= data set. (See also the OITER option.)

OUTFIT=*SAS-data-set*
creates a SAS data set containing goodness-of-fit and badness-of-fit measures for each partition as well as for the entire data set. (See also the OITER option.)

OUTRES=*SAS-data-set*
creates a SAS data set containing one observation for each nonmissing data value from the DATA= data set. Each observation contains the original data value, the estimated distance computed from the MDS model, transformed data and distances, and the residual. (See also the OITER option.)

OVER=*n*
specifies the maximum overrelaxation factor, where $n \geq 1$. Values between 1 and 2 are generally reasonable. The default is OVER=2 with the LEVEL=ORDINAL, ALTERNATE=MATRIX, or ALTERNATE=ROW option; otherwise, the default is OVER=1. Use this option only if you have convergence problems.

PCOEF
produces the estimated dimension coefficients.

PCONFIG
produces the estimated coordinates of the objects in the configuration.

PDATA
displays each data matrix.

PFINAL
displays final estimates.

PFIT
displays the badness-of-fit criterion and various types of correlations between the data and fitted values for each data matrix, as well as for the entire sample.

PFITROW
displays the badness-of-fit criterion and various types of correlations between the data and fitted values for each row, as well as for each data matrix and for the entire sample. This option works only with the CONDITION=ROW option.

PINAVDATA
displays the sum of the weights and the weighted average of the data matrices computed during initialization with the INAV=DATA option.

PINEIGVAL
displays the eigenvalues computed during initialization.

PINEIGVEC
displays the eigenvectors computed during initialization.

PININ
displays values read from the INITIAL= data set. Since these values might be incomplete, the PFIT and PFITROW options do not apply.

PINIT
displays initial values.

PITER
displays estimates on each iteration.

PLOTS<(*global-plot-option*)> <= *plot-request*<(*options*)>>
PLOTS<(*global-plot-options*)> <= (*plot-request*<(*options*)> <... *plot-request*<(*options*)>>)>

specifies options that control the details of the plots. When you specify only one plot request, you can omit the parentheses around the plot request.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc mds plots(flip);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The global plot option is as follows:

FLIP
flips or interchanges the X-axis and Y-axis dimensions for configuration and coefficient plots.

The plot requests include the following:

COEFFICIENTS(ONE)
combines the INDSCAL coefficients panel of plots into a single plot. By default, the display consists of a panel with two plots. The vectors are displayed in the left plot, and the labels are displayed in the right plot. The right plot provides a magnification of the region of the vector endpoints. In contrast, the single display, requested by COEFFICIENTS(ONE), is more compact, but there is less room for vector labels. It is often easier to identify the vectors in the default display.

NONE
suppresses all plots.

By default, a fit plot is produced. When more then one dimension is requested, plots of the configuration are also plotted. For individual differences models with more than one dimension, the subject weights or coefficients are plotted. When more than one value is specified for the DIMENSION= option, the badness-of-fit plot is produced.

PTRANS
displays the estimated transformation parameters if any are computed. There are no transformation parameters with the LEVEL=ORDINAL option.

RANDOM<=*seed*>
causes initial coordinate values to be pseudo-random numbers. In one dimension, the pseudo-random numbers are uniformly distributed on an interval. In two or more dimensions, the pseudo-random numbers are uniformly distributed on the circumference of a circle or the surface of a (hyper)sphere.

RIDGE=*n*
specifies the initial ridge value, where $n \geq 0$. The default is RIDGE=1E–4.

If you get a floating-point overflow in the first few iterations, specify a larger value such as RIDGE=0.01, RIDGE=1, or RIDGE=100.

If you know that the initial estimates are very good, using RIDGE=0 might speed convergence.

SHAPE=TRIANGULAR | TRIANGLE | TRI | T
SHAPE=SQUARE | SQU | S

determines whether the entire data matrix for each subject or only one triangle of the matrix is stored and analyzed. If you specify the CONDITION=ROW option, the default is SHAPE=SQUARE. Otherwise, the default is SHAPE=TRIANGLE.

SQUARE causes the entire matrix to be stored and analyzed. The matrix can be asymmetric.

TRIANGLE causes only one triangle to be stored. However, PROC MDS reads both upper and lower triangles to look for nonmissing values and to symmetrize the data if needed. If corresponding elements in the upper and lower triangles both contain nonmissing values, only the average of the two values is stored and analyzed (Kruskal and Wish 1978 , p. 74). Also, if an OUTRES= data set is requested, only the average of the two corresponding elements is output.

SIMILAR | SIM< =*max* >

causes the data to be treated as similarity measurements rather than dissimilarities. If =*max* is not specified, each data value is converted to a dissimilarity by subtracting it from the maximum value in the data set or BY group. If =*max* is specified, each data value is subtracted from the maximum of *max* and the data. The diagonal data are included in computing these maxima.

By default, the data are assumed to be dissimilarities unless there are nonmissing diagonal values and these values are predominantly larger than the off-diagonal values. In this case, the data are assumed to be similarities and are treated as if the SIMILAR option is specified.

SINGULAR=*p*

specifies the singularity criterion p, $0 \leq p \leq 1$. The default is SINGULAR=1E–8.

UNTIE

permits tied data to be assigned different optimally scaled values with the LEVEL=ORDINAL option. Otherwise, tied data are assigned equal optimally scaled values. The UNTIE option has no effect with values of the LEVEL= option other than LEVEL=ORDINAL.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC MDS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the MDS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.

- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

If the INITIAL= data set contains the BY variables, the BY groups must appear in the same order as in the DATA= data set. If the BY variables are not in the INITIAL= data set, the entire data set is used to provide initial values for each BY group in the DATA= data set.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

ID Statement

ID | OBJECT | OBJ *variable* **;**

The ID statement specifies a variable in the DATA= data set that contains descriptive labels for the objects. The labels are used in the output and are copied to the OUT= data set. If there is more than one data matrix, only the ID values from the observations containing the first data matrix are used.

The ID variable is not used to establish any correspondence between observations and variables.

If the ID statement is omitted, the variable labels or names are used as object labels.

INVAR Statement

INVAR *variables* **;**

The INVAR statement specifies the numeric variables in the INITIAL= data set that contain initial parameter estimates. The first variable corresponds to the first dimension, the second variable to the second dimension, and so on.

If the INVAR statement is omitted, the variables Dim1, . . . , Dim$m$ are used, where m is the maximum number of dimensions.

MATRIX Statement

MATRIX | MAT | SUBJECT | SUB *variable* **;**

The MATRIX statement specifies a variable in the DATA= data set that contains descriptive labels for the data matrices or subjects. The labels are used in the output and are copied to the OUT= and OUTRES= data sets. Only the first observation from each data matrix is used to obtain the label for that matrix.

If the MATRIX statement is omitted, the matrices are labeled 1, 2, 3, and so on.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the numeric variables in the DATA= data set that contain similarity or dissimilarity measurements on a set of objects or stimuli. Each variable corresponds to one object.

If the VAR statement is omitted, all numeric variables that are not specified in another statement are used.

To analyze a subset of the objects in a data set, you can specify the variable names corresponding to the columns in the subset, but you must also use a DATA step or a WHERE clause to specify the rows in the subset. PROC MDS expects to read one or more square matrices, and you must ensure that the rows in the data set correctly correspond to the columns in number and order.

WEIGHT Statement

WEIGHT *variables* **;**

The WEIGHT statement specifies numeric variables in the DATA= data set that contain weights for each similarity or dissimilarity measurement. These weights are used to compute weighted least-squares estimates. The number of WEIGHT variables must be the same as the number of VAR variables, and the variables in the WEIGHT statement must be in the same order as the corresponding variables in the VAR statement.

If the WEIGHT statement is omitted, all data within a partition are assigned equal weights.

Data with zero or negative weights are ignored in fitting the model but are included in the OUTRES= data set and in monotone transformations.

Details: MDS Procedure

Formulas

The following notation is used:

A_p	intercept for partition p
B_p	slope for partition p
C_p	power for partition p
D_{rcs}	distance computed from the model between objects r and c for subject s
F_{rcs}	data weight for objects r and c for subject s obtained from the cth WEIGHT variable, or 1 if there is no WEIGHT statement
f	value of the FIT= option
N	number of objects
O_{rcs}	observed dissimilarity between objects r and c for subject s

P_{rcs}	partition index for objects r and c for subject s
Q_{rcs}	dissimilarity after applying any applicable estimated transformation for objects r and c for subject s
R_{rcs}	residual for objects r and c for subject s
S_p	standardization factor for partition p
$T_p(\cdot)$	estimated transformation for partition p
V_{sd}	coefficient for subject s on dimension d
X_{nd}	coordinate for object n on dimension d

Summations are taken over nonmissing values.

Distances are computed from the model as

$$D_{rcs} = \begin{cases} \sqrt{\sum_d (X_{rd} - X_{cd})^2} & \text{for COEF=IDENTITY: Euclidean distance} \\ \sqrt{\sum_d V_{sd}^2 (X_{rd} - X_{cd})^2} & \text{for COEF=DIAGONAL: weighted Euclidean distance} \end{cases}$$

Partition indexes are

$$\begin{aligned} P_{rcs} &= 1 && \text{for CONDITION=UN} \\ &= s && \text{for CONDITION=MATRIX} \\ &= (s-1)N + r && \text{for CONDITION=ROW} \end{aligned}$$

The estimated transformation for each partition is

$$\begin{aligned} T_p(d) &= d && \text{for LEVEL=ABSOLUTE} \\ &= B_p d && \text{for LEVEL=RATIO} \\ &= A_p + B_p d && \text{for LEVEL=INTERVAL} \\ &= B_p d^{C_p} && \text{for LEVEL=LOGINTERVAL} \end{aligned}$$

For LEVEL=ORDINAL, $T_p(\cdot)$ is computed as a least-squares monotone transformation.

For LEVEL=ABSOLUTE, RATIO, or INTERVAL, the residuals are computed as

$$\begin{aligned} Q_{rcs} &= O_{rcs} \\ R_{rcs} &= Q_{rcs}^f - [T_{P_{rcs}}(D_{rcs})]^f \end{aligned}$$

For LEVEL=ORDINAL, the residuals are computed as

$$\begin{aligned} Q_{rcs} &= T_{P_{rcs}}(O_{rcs}) \\ R_{rcs} &= Q_{rcs}^f - D_{rcs}^f \end{aligned}$$

If f is 0, then natural logarithms are used in place of the fth powers.

For each partition, let

$$U_p = \frac{\sum_{r,c,s} F_{rcs}}{\sum_{r,c,s|P_{rcs}=p} F_{rcs}}$$

and

$$\overline{Q}_p = \frac{\sum_{r,c,s|P_{rcs}=p} Q_{rcs} F_{rcs}}{\sum_{r,c,s|P_{rcs}=p} F_{rcs}}$$

Then the standardization factor for each partition is

$$\begin{aligned} S_p &= 1 && \text{for FORMULA=0} \\ &= U_p \sum_{r,c,s|P_{rcs}=p} Q_{rcs}^2 F_{rcs} && \text{for FORMULA=1} \\ &= U_p \sum_{r,c,s|P_{rcs}=p} (Q_{rcs} - \overline{Q}_p)^2 F_{rcs} && \text{for FORMULA=2} \end{aligned}$$

The badness-of-fit criterion that the MDS procedure tries to minimize is

$$\sqrt{\sum_{r,c,s} \frac{R_{rcs}^2 F_{rcs}}{S_{P_{rcs}}}}$$

OUT= Data Set

The OUT= data set contains the following variables:

- BY variables, if any
- _ITER_ (if the OUTITER option is specified), a numeric variable containing the iteration number
- _DIMENS_, a numeric variable containing the number of dimensions
- _MATRIX_ or the variable in the MATRIX statement, identifying the data matrix or subject to which the observation pertains. This variable contains a missing value for observations that pertain to the data set as a whole and not to a particular matrix, such as the coordinates (_TYPE_='CONFIG').
- _TYPE_, a character variable of length 10 identifying the type of information in the observation

  The values of _TYPE_ are as follows:

CONFIG	the estimated coordinates of the configuration of objects
DIAGCOEF	the estimated dimension coefficients for COEF=DIAGONAL
INTERCEPT	the estimated intercept parameters

SLOPE	the estimated slope parameters
POWER	the estimated power parameters
CRITERION	the badness-of-fit criterion

- _LABEL_ or the variable in the ID statement, containing the variable label or value of the ID variable of the object to which the observation pertains. This variable contains a missing value for observations that do not pertain to a particular object or dimension.
- _NAME_, a character variable containing the variable name of the object or dimension to which the observation pertains. This variable contains a missing value for observations that do not pertain to a particular object or dimension.
- Dim1, . . . , Dim*m*, where *m* is the maximum number of dimensions

OUTFIT= Data Set

The OUTFIT= data set contains various measures of goodness and badness of fit. There is one observation for the entire sample plus one observation for each matrix. For the CONDITION=ROW option, there is also one observation for each row.

The OUTFIT= data set contains the following variables:

- BY variables, if any
- _ITER_ (if the OUTITER option is specified), a numeric variable containing the iteration number
- _DIMENS_, a numeric variable containing the number of dimensions
- _MATRIX_ or the variable in the MATRIX statement, identifying the data matrix or subject to which the observation pertains
- _LABEL_ or the variable in the ID statement, containing the variable label or value of the ID variable of the object to which the observation pertains when CONDITION=ROW
- _NAME_, a character variable containing the variable name of the object or dimension to which the observation pertains when CONDITION=ROW
- N, the number of nonmissing data
- WEIGHT, the weight of the partition
- CRITER, the badness-of-fit criterion
- DISCORR, the correlation between the transformed data and the distances for LEVEL=ORDINAL or the correlation between the data and the transformed distances otherwise
- UDISCORR, the correlation uncorrected for the mean between the transformed data and the distances for LEVEL=ORDINAL or the correlation between the data and the transformed distances otherwise
- FITCORR, the correlation between the fit-transformed data and the fit-transformed distances

- UFITCORR, the correlation uncorrected for the mean between the fit-transformed data and the fit-transformed distances

OUTRES= Data Set

The OUTRES= data set has one observation for each nonmissing data value. It contains the following variables:

- BY variables, if any
- _ITER_ (if the OUTITER option is specified), a numeric variable containing the iteration number
- _DIMENS_, a numeric variable containing the number of dimensions
- _MATRIX_ or the variable in the MATRIX statement, identifying the data matrix or subject to which the observation pertains
- _ROW_, containing the variable label or value of the ID variable of the row to which the observation pertains
- _COL_, containing the variable label or value of the ID variable of the column to which the observation pertains
- DATA, the original data value
- TRANDATA, the optimally transformed data value when LEVEL=ORDINAL
- DISTANCE, the distance computed from the PROC MDS model
- TRANSDIST, the optimally transformed distance when the LEVEL= option is not ORDINAL or ABSOLUTE
- FITDATA, the data value further transformed according to the FIT= option
- FITDIST, the distance further transformed according to the FIT= option
- WEIGHT, the combined weight of the data value based on the WEIGHT variable(s), if any, and the standardization specified by the FORMULA= option
- RESIDUAL, FITDATA minus FITDIST

If you assign a nonmissing data value a weight of zero, PROC MDS will ignore it when the model is fit, but the value will still appear in the OUTRES= data set (see the section "WEIGHT Statement" on page 6529).

INITIAL= Data Set

The INITIAL= data set has the same structure as the OUT= data set but is not required to have all of the variables or observations that appear in the OUT= data set. You can use an OUT= data set previously created by PROC MDS (without the OUTITER option) as an INITIAL= data set in a subsequent invocation of the procedure.

The only variables that are required are Dim1, . . . , Dim*m* (where *m* is the maximum number of dimensions) or equivalent variables specified in the INVAR statement. If these are the only variables, then all the observations are assumed to contain coordinates of the configuration; you cannot read dimension coefficients or transformation parameters.

To read initial values for the dimension coefficients or transformation parameters, the INITIAL= data set must contain the _TYPE_ variable and either the variable specified in the ID statement or, if no ID statement is used, the variable _NAME_. In addition, if there is more than one data matrix, either the variable specified in the MATRIX statement or, if no MATRIX statement is used, the variable _MATRIX_ or _MATNUM_ is required.

If the INITIAL= data set contains the variable _DIMENS_, initial values are obtained from observations with the corresponding number of dimensions. If there is no _DIMENS_ variable, the same observations are used for each number of dimensions analyzed.

If you want PROC MDS to read initial values from some but not all of the observations in the INITIAL= data set, use the WHERE= data set option to select the desired observations.

Missing Values

Missing data in the similarity or dissimilarity matrices are ignored in fitting the model and are omitted from the OUTRES= data set. Any matrix that is completely missing is omitted from the analysis.

Missing weights are treated as 0.

Missing values are also permitted in the INITIAL= data set, but a large number of missing values might yield a degenerate initial configuration.

Normalization of the Estimates

In multidimensional scaling models, the parameter estimates are not uniquely determined; the estimates can be transformed in various ways without changing their badness of fit. The initial and final estimates from PROC MDS are, therefore, normalized (unless you specify the NONORM option) to make it easier to compare results from different analyses.

The configuration always has a mean of 0 for each dimension.

With the COEF=IDENTITY option, the configuration is rotated to a principal-axis orientation. Unless you specify the LEVEL=ABSOLUTE option, the entire configuration is scaled so that the root-mean-square element is 1, and the transformations are adjusted to compensate.

With the COEF=DIAGONAL option, each dimension is scaled to a root-mean-square value of 1, and the dimension coefficients are adjusted to compensate. Unless you specify the LEVEL=ABSOLUTE option, the

dimension coefficients are normalized as follows. If you specify the CONDITION=UN option, all of the dimension coefficients are scaled to a root-mean-square value of 1. For other values of the CONDITION= option, the dimension coefficients are scaled separately for each subject to a root-mean-square value of 1. In either case, the transformations are adjusted to compensate.

Each dimension is reflected to give a positive rank correlation with the order of the objects in the data set.

For the LEVEL=ORDINAL option, if the intercept, slope, or power parameters are fitted, the transformed data are normalized to eliminate these parameters if possible.

Comparison with Earlier Procedures

PROC MDS shares many of the features of the ALSCAL procedure (Young, Lewyckyj, and Takane 1986; Young 1982), as well as some features of the MLSCALE procedure (Ramsay 1986). Both PROC ALSCAL and PROC MLSCALE are no longer a part of SAS; however, they are described in the *SUGI Supplemental Library User's Guide, Version 5 Edition*. The MDS procedure generally produces results similar to those from the ALSCAL procedure (Young, Lewyckyj, and Takane 1986; Young 1982) if you use the following *options* in PROC MDS:

- FIT=SQUARED
- FORMULA=1 except for unfolding data, which require FORMULA=2
- PFINAL to get output similar to that from PROC ALSCAL

Running the MDS procedure with certain options generally produces results similar to those from using the MLSCALE procedure (Ramsay 1986) with other options. This is illustrated with the following statements:

```
proc mds fit=log level=loginterval ... ;

proc mlscale stvarnce=constant suvarnce=constant ... ;
```

Alternatively, using the FIT=DISTANCE option in the PROC MDS statement produces results similar to those from specifying the NORMAL option in the PROC MLSCALE statement.

Displayed Output

Unless you specify the NOPHIST option, PROC MDS displays the iteration history containing the following:

- Iteration number
- Type of iteration:

Initial	initial configuration
Monotone	monotone transformation
Gau-New	Gauss-Newton step
Lev-Mar	Levenberg-Marquardt step

- Badness-of-Fit Criterion
- Change in Criterion
- Convergence Measures:

  Monotone — the Euclidean norm of the change in the optimally scaled data divided by the Euclidean norm of the optimally scaled data, averaged across partitions

  Gradient — the multiple correlation of the Jacobian matrix with the residual vector, uncorrected for the mean

Depending on what options are specified, PROC MDS can also display the following tables:

- Data Matrix and possibly Weight Matrix for each subject
- Eigenvalues from the computation of the initial coordinates
- Sum of Data Weights and Pooled Data Matrix computed during initialization with INAV=DATA
- Configuration, the estimated coordinates of the objects
- Dimension Coefficients
- A table of transformation parameters, including one or more of the following:

  Intercept
  Slope
  Power

- A table of fit statistics for each matrix and possibly each row, including the following:

  Number of Nonmissing Data
  Weight of the matrix or row, permitting both observation weights and standardization factors
  Badness-of-Fit Criterion
  Distance Correlation computed between the distances and data with optimal transformation
  Uncorrected Distance Correlation not corrected for the mean
  Fit Correlation computed after applying the FIT= transformation to both distances and data
  Uncorrected Fit Correlation not corrected for the mean

ODS Table Names

PROC MDS assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 81.3. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 81.3 ODS Tables Produced by PROC MDS

ODS Table Name	Description	Option
ConvergenceStatus	Convergence status	Default
DimensionCoef	Dimension coefficients	PCOEF with COEF= not IDENTITY
FitMeasures	Measures of fit	PFIT
IterHistory	Iteration history	Default
PConfig	Configuration of coordinates	PCONFIG
PData	Data matrices	PDATA
PInAvData	INAV= data set information	PINAVDATA
PInEigval	Initial eigenvalues	PINEIGVAL
PInEigvec	Initial eigenvectors	PINEIGVEC
PInWeight	Initialization weights	PINWEIGHT
Transformations	Transformation parameters	PTRANS with LEVEL=RATIO, INTERVAL, LOGINTERVAL

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

All graphs are produced by default when they are appropriate. You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC MDS generates are listed in Table 81.4, along with the required options.

Table 81.4 Graphs Produced by PROC MDS

ODS Graph Name	Plot Description	Option
BadnessPlot	Badness of fit	DIMENSION=*range*
CoefficientsPlot	Individual coefficients	DIMENSION=n, $n > 1$ COEF=DIAGONAL
ConfigPlot	Configuration	DIMENSION=n, $n > 1$
FitPlot	Fit	Default

Example: MDS Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 81.1: Jacobowitz Body Parts Data from Children and Adults

(View the complete code for this example (mdsex.sas) in the example repository.)

Jacobowitz (1975) collected conditional rank-order data regarding perceived similarity of parts of the body from children of ages 6, 8, and 10 years and from college sophomores. The data set includes data from 15 children (6-year-olds) and 15 sophomores. The method of data collection and some results of an analysis are also described by Young (1987, pp. 4–10). The following statements create the input data set:

```
data body;
   title  'Jacobowitz Body Parts Data from 6-Year-Olds and Adults';
   title2 'First 15 Subjects (obs 1-225) Are Children';
   title3 'Second 15 Subjects (obs 226-450) Are Adults';
   input  (Cheek Face Mouth Head Ear Body Arm Elbow Hand
           Palm Finger Leg Knee Foot Toe) (2.);
   if _n_ <= 225 then Subject='C'; else subject='A';
   datalines;
 0  2  1  3  4 10  5  9  6  7  8 11 12 13 14
 2  0 12  1 13  3  8 10 11  9  7  4  5  6 14
 3  2  0  1  4  9  5 11  6  7  8 10 13 12 14
 2  1  3  0  4  9  5  6 11  7  8 10 12 13 14

   ... more lines ...

10 12 11 13  9 14  8  7  4  6  2  3  5  1  0
;
```

The data are analyzed as row conditional (CONDITION=ROW) at the ordinal level of measurement (LEVEL=ORDINAL) by using the weighted Euclidean model (COEF=DIAGONAL) in three dimensions (DIMENSION=3). The final estimates are displayed (PFINAL). The estimates (OUT=OUT) and fitted values (OUTRES=RES) are saved in output data sets. The following statements produce Output 81.1.1:

```
ods graphics on;

proc mds data=body condition=row level=ordinal coef=diagonal
         dimension=3 pfinal out=out outres=res;
   subject subject;
   title5 'Nonmetric Weighted MDS';
run;
```

Output 81.1.1 Analysis of Body Parts Data

Jacobowitz Body Parts Data from 6-Year-Olds and Adults
First 15 Subjects (obs 1-225) Are Children
Second 15 Subjects (obs 226-450) Are Adults

Nonmetric Weighted MDS

Multidimensional Scaling: Data=WORK.BODY.DATA
Shape=SQUARE Condition=ROW Level=ORDINAL
Coef=DIAGONAL Dimension=3 Formula=1 Fit=1

Mconverge=0.01 Gconverge=0.01 Maxiter=100 Over=2 Ridge=0.0001 Alternate=MATRIX

				Convergence Measures	
Iteration	Type	Badness-of-Fit Criterion	Change in Criterion	Monotone	Gradient
0	Initial	0.5938	.	.	.
1	Monotone	0.2344	0.3594	0.4693	0.4028
2	Gau-New	0.2080	0.0264	.	.
3	Monotone	0.1963	0.0118	0.0556	0.2630
4	Gau-New	0.1927	0.003592	.	.
5	Monotone	0.1797	0.0130	0.0463	0.1544
6	Gau-New	0.1779	0.001809	.	.
7	Monotone	0.1744	0.003430	0.0225	0.1210
8	Gau-New	0.1736	0.000807	.	.
9	Monotone	0.1717	0.001929	0.0161	0.1128
10	Gau-New	0.1712	0.000474	.	.
11	Monotone	0.1698	0.001413	0.0135	0.1119
12	Gau-New	0.1696	0.000188	.	.
13	Monotone	0.1684	0.001261	0.0121	0.1121
14	Gau-New	0.1683	0.000117	.	.
15	Monotone	0.1672	0.001096	0.0111	0.1064
16	Gau-New	0.1670	0.000131	.	.
17	Monotone	0.1661	0.000902	0.0103	0.0965
18	Gau-New	0.1660	0.000160	.	.
19	Monotone	0.1652	0.000736	0.009740	0.0980
20	Gau-New	0.1651	0.000169	.	0.1062
21	Gau-New	0.1645	0.000542	.	0.0161
22	Gau-New	0.1645	4.2645E-6	.	0.009969

Convergence criteria are satisfied.

Output 81.1.1 *continued*

Configuration			
	Dim1	**Dim2**	**Dim3**
Cheek	0.25	1.47	2.06
Face	0.36	0.32	0.33
Mouth	0.78	0.08	1.08
Head	2.10	1.07	-0.01
Ear	0.26	0.72	-0.34
Body	1.51	-0.61	-0.68
Arm	-0.74	2.07	-0.59
Elbow	-0.51	-0.21	0.01
Hand	0.46	-1.50	-0.60
Palm	-1.44	-0.81	1.48
Finger	-0.24	-0.24	-0.81
Leg	-1.68	0.26	-0.05
Knee	-1.19	-1.19	-1.36
Foot	0.16	-0.03	-1.56
Toe	-0.10	-1.39	1.02

Output 81.1.1 *continued*

Dimension Coefficients			
Subject	**1**	**2**	**3**
C	1.00	1.12	0.86
C	0.96	1.02	1.01
C	0.98	1.05	0.98
C	1.02	1.08	0.89
C	0.95	1.04	1.01
C	0.99	1.12	0.89
C	1.07	1.00	0.93
C	1.04	1.02	0.94
C	0.99	1.15	0.83
C	0.89	1.11	0.99
C	1.04	1.03	0.92
C	1.06	1.01	0.93
C	0.92	1.24	0.78
C	0.97	0.98	1.05
C	1.03	1.00	0.97
A	0.93	1.17	0.88
A	0.89	1.12	0.97
A	0.88	1.17	0.94
A	0.81	1.14	1.02
A	0.90	1.11	0.98
A	0.90	1.17	0.91
A	0.92	1.17	0.88
A	0.97	1.19	0.80
A	0.95	1.16	0.87
A	1.08	1.07	0.83
A	0.95	1.20	0.81
A	1.00	0.97	1.02
A	0.89	1.18	0.91
A	0.97	1.15	0.86
A	0.93	1.21	0.82

Output 81.1.1 *continued*

Subject	Number of Nonmissing Data	Weight	Badness-of-Fit Criterion	Distance Correlation	Uncorrected Distance Correlation
C	160	0.03	0.15	0.86	0.99
C	163	0.03	0.19	0.78	0.98
C	166	0.03	0.20	0.79	0.98
C	158	0.03	0.16	0.84	0.99
C	173	0.03	0.18	0.83	0.98
C	164	0.03	0.14	0.90	0.99
C	158	0.03	0.20	0.77	0.98
C	170	0.03	0.18	0.83	0.98
C	156	0.03	0.15	0.88	0.99
C	165	0.03	0.18	0.79	0.98
C	153	0.03	0.19	0.79	0.98
C	162	0.03	0.17	0.83	0.98
C	161	0.03	0.14	0.90	0.99
C	164	0.03	0.17	0.83	0.99
C	161	0.03	0.18	0.81	0.98
A	163	0.03	0.15	0.87	0.99
A	174	0.04	0.17	0.85	0.99
A	172	0.03	0.15	0.89	0.99
A	175	0.04	0.17	0.85	0.98
A	171	0.03	0.15	0.87	0.99
A	163	0.03	0.16	0.86	0.99
A	173	0.03	0.14	0.90	0.99
A	160	0.03	0.14	0.89	0.99
A	164	0.03	0.14	0.90	0.99
A	158	0.03	0.16	0.86	0.99
A	165	0.03	0.16	0.87	0.99
A	168	0.03	0.18	0.82	0.98
A	175	0.04	0.15	0.89	0.99
A	172	0.03	0.16	0.88	0.99
A	175	0.04	0.15	0.90	0.99
- All -	4962	1.00	0.16	0.85	0.99

Output 81.1.1 *continued*

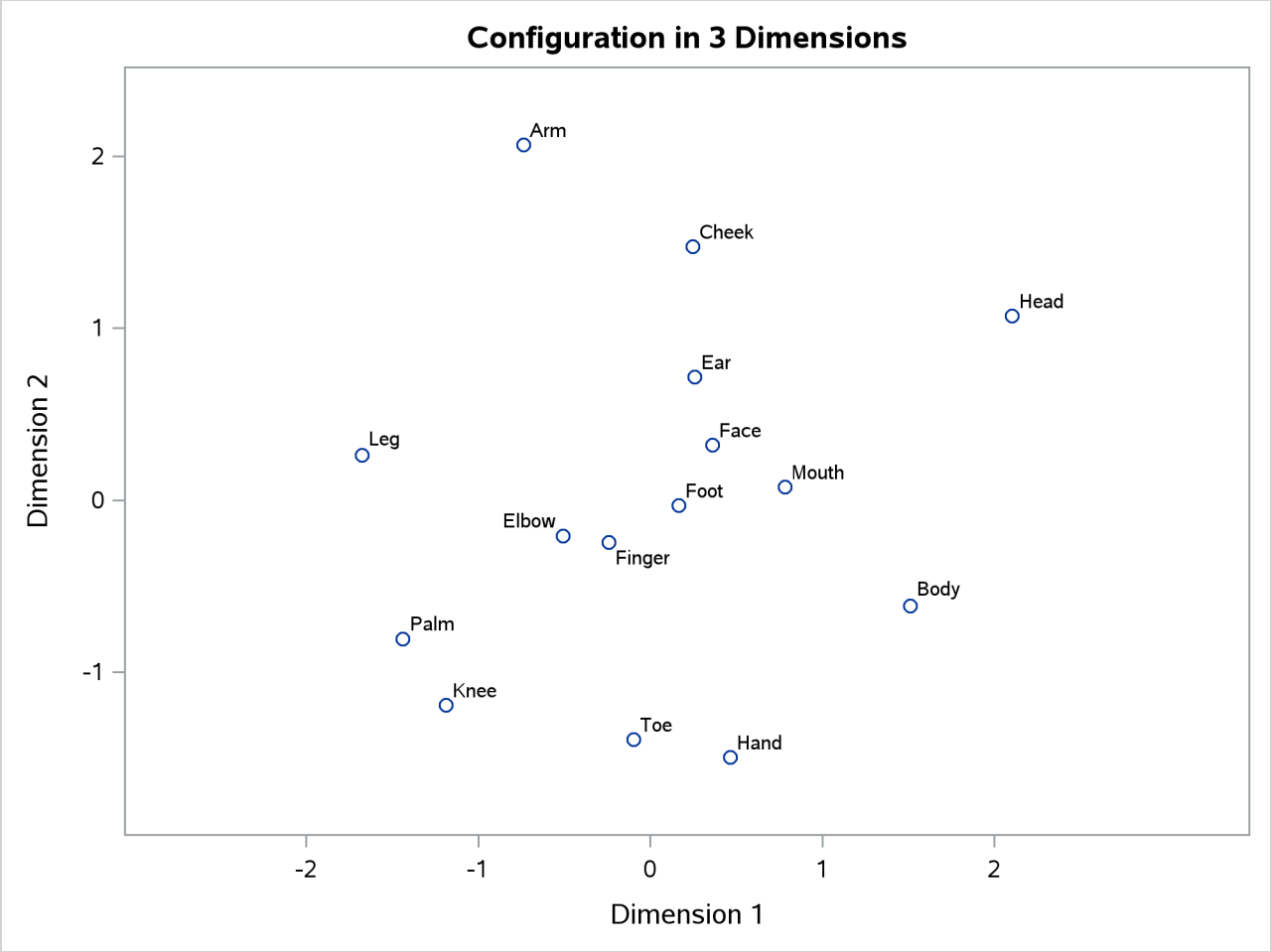

Output 81.1.1 *continued*

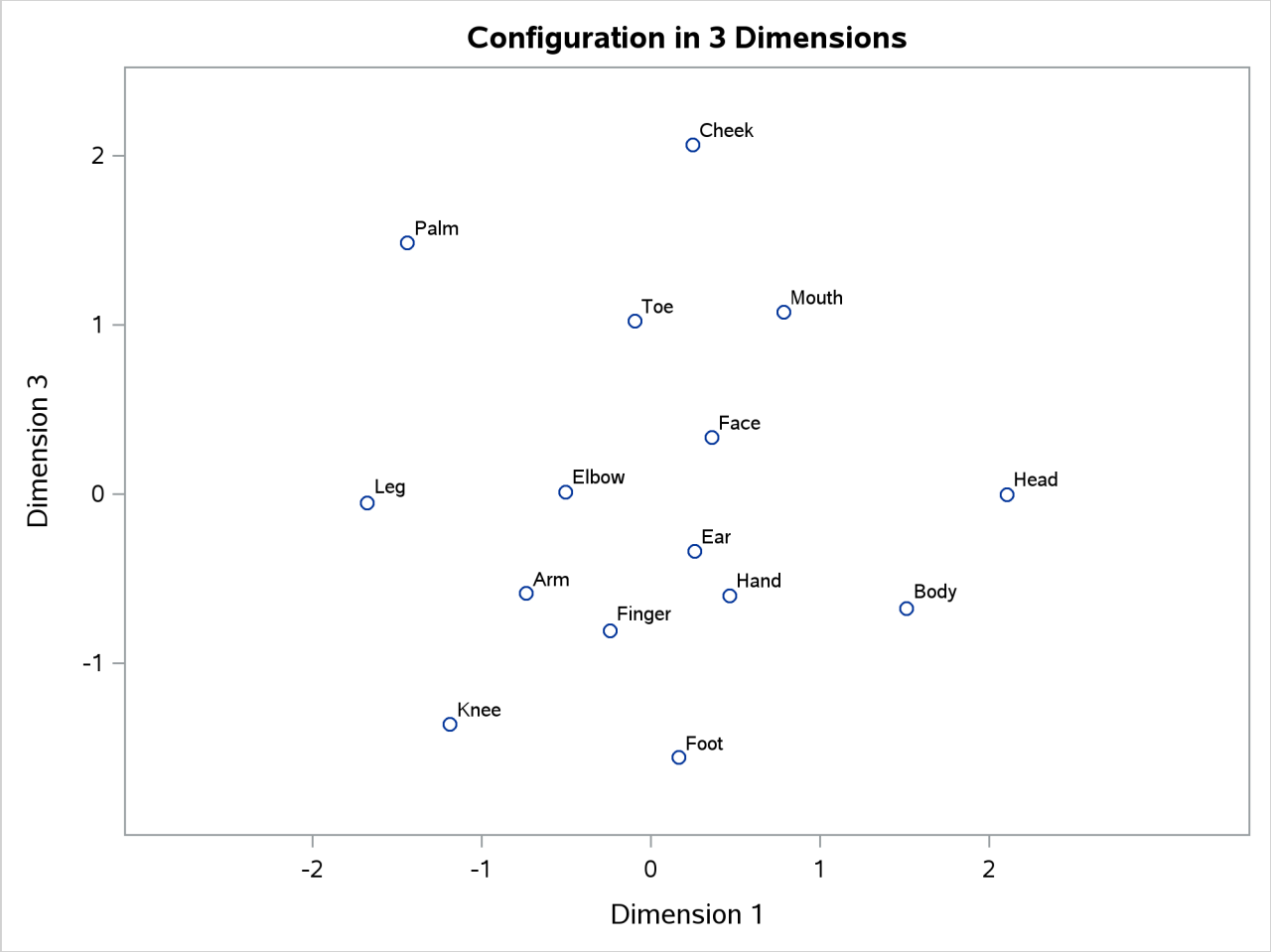

Output 81.1.1 *continued*

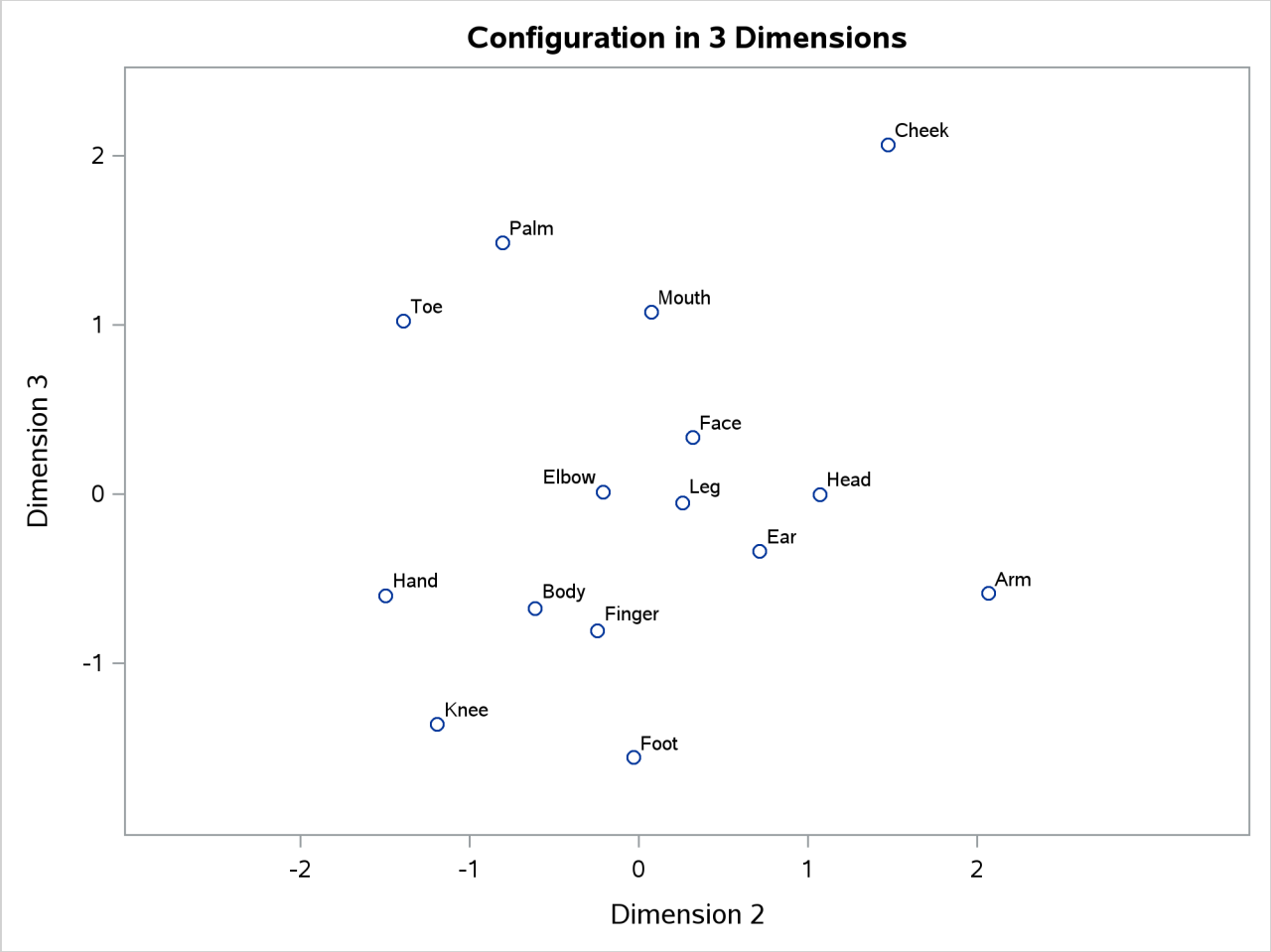

Output 81.1.1 *continued*

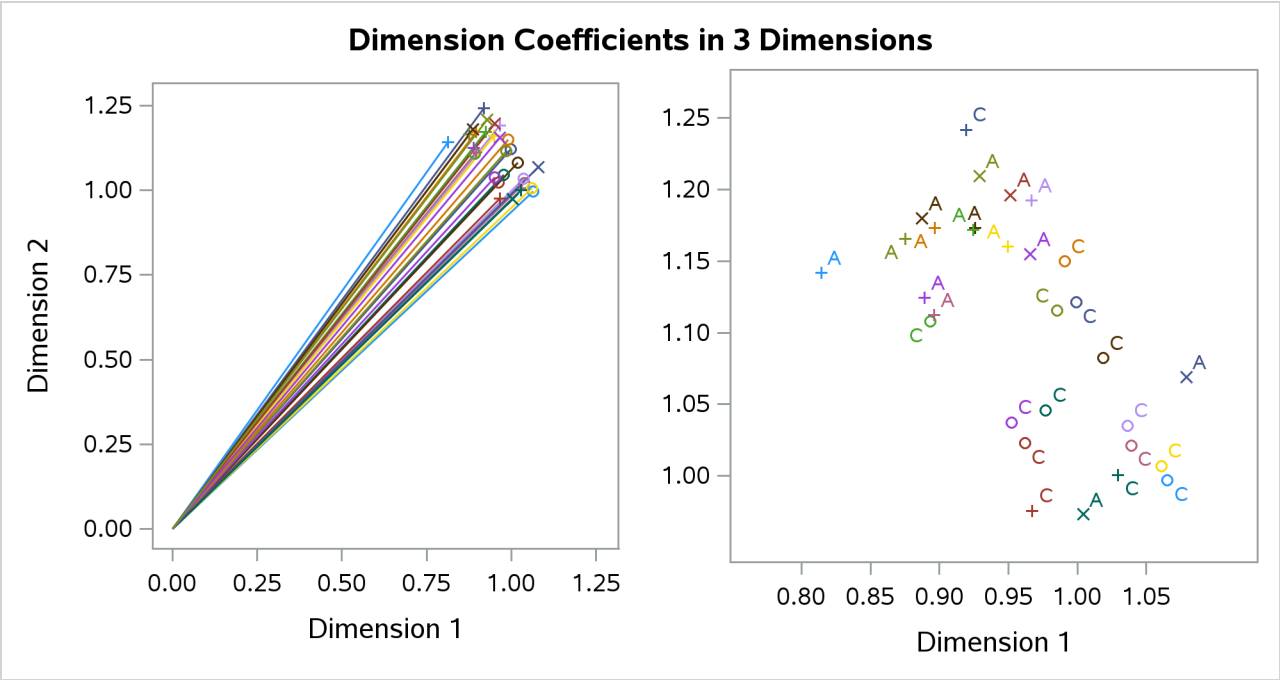

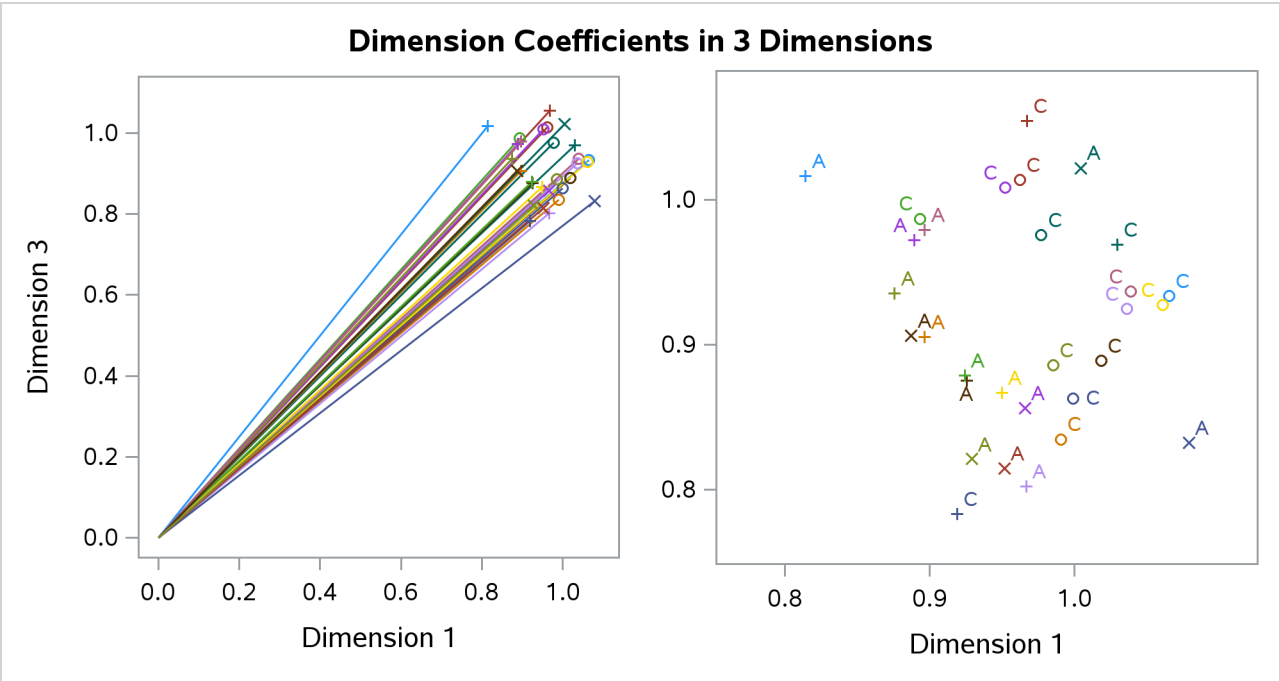

Output 81.1.1 *continued*

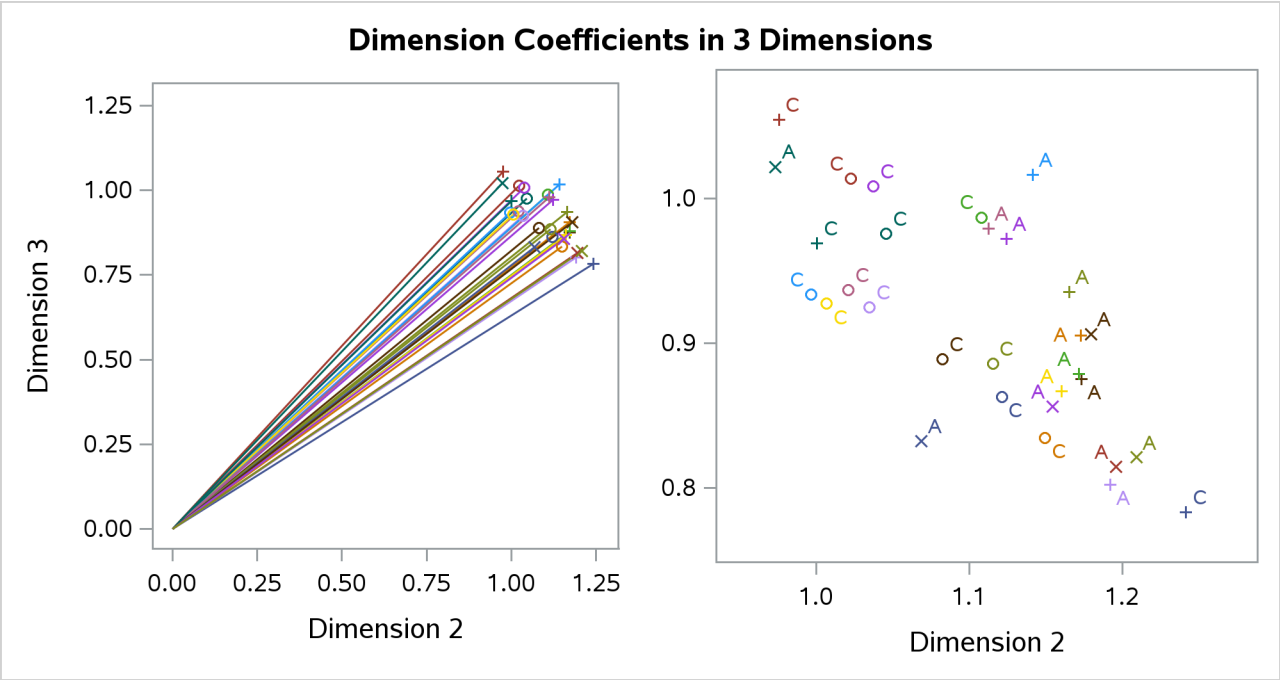

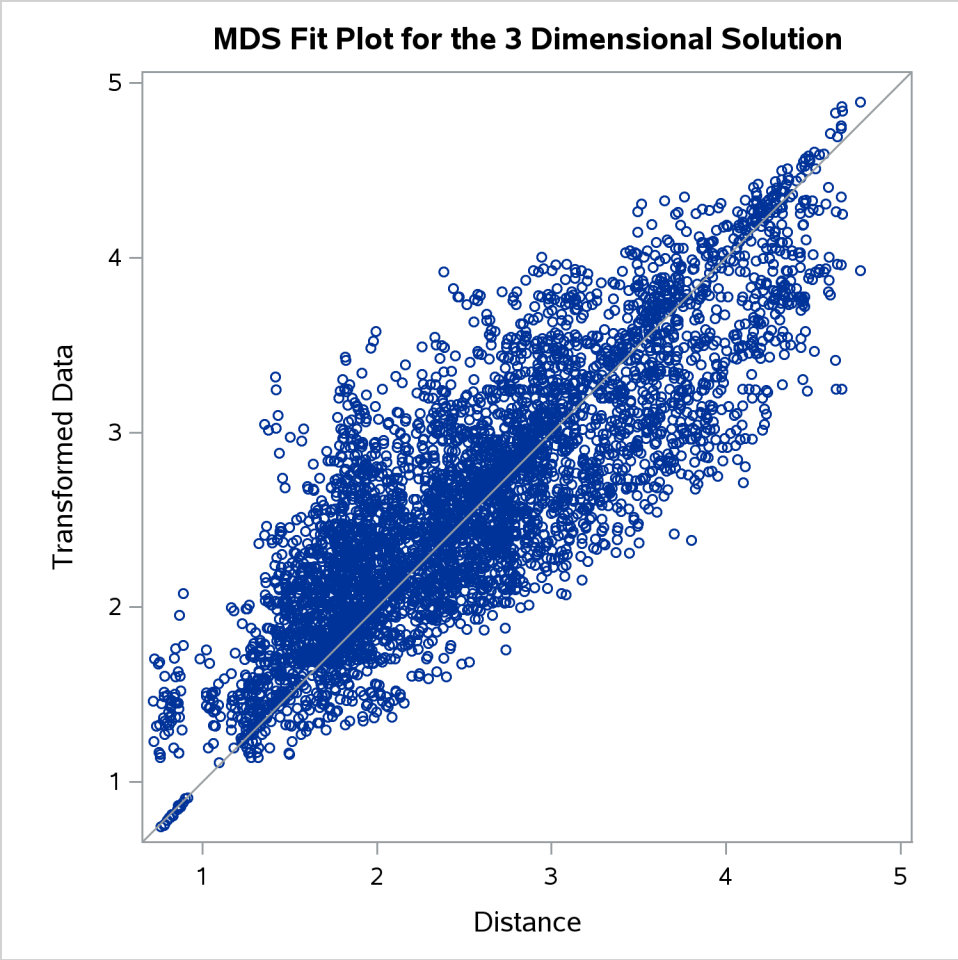

Often the output of greatest interest in an MDS analysis is the graphical output. The first plots show two-dimensional view of the three-dimensional configuration. Next, the coefficients are plotted. The last plot is the fit plot.

In the fit plot, the transformed data are plotted on the vertical axis, and the distances from the model are plotted on the horizontal axis. If the model fits perfectly, all points lie on a diagonal line from lower left to upper right. The vertical departure of each point from this diagonal line represents the residual of the corresponding observation.

The configuration has a tripodal shape with Body at the apex. The three legs of the tripod can be distinguished in the plot of dimension 2 by dimension 1, which shows three distinct clusters with Body in the center. Dimension 1 separates head parts from arm and leg parts. Dimension 2 separates arm parts from leg parts. The plot of dimension 3 by dimension 1 shows the tripod from the side. Dimension 3 distinguishes the more inclusive body parts (at the top) from the less inclusive body parts (at the bottom).

The plots of dimension coefficients show that children differ from adults primarily in the emphasis given to dimension 2. Children give about the same weight (approximately 1) to each dimension. Adults are much more variable than children, but all have coefficients less than 1.0 for dimension 2, with an average of about 0.7. Referring back to the configuration plot, you can see that adults consider arm parts to be more similar to leg parts than children do. Many adults also give a high weight to dimension 1, indicating that they consider head parts to be more dissimilar from arm and leg parts than children do. Dimension 3 shows considerable variability for both children and adults.

References

Arabie, P., Carroll, J. D., and DeSarbo, W. S. (1987). *Three-Way Scaling and Clustering.* Vol. 07-065 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Carroll, J. D., and Chang, J. J. (1970). "Analysis of Individual Differences in Multidimensional Scaling via an N-Way Generalization of the 'Eckart-Young' Decomposition." *Psychometrika* 35:283–319.

Heiser, W. J. (1981). *Unfolding Analysis of Proximity Data*. Leiden, Netherlands: Department of Data Theory, University of Leiden.

Jacobowitz, D. (1975). "The Acquisition of Semantic Structures." Ph.D. diss., University of North Carolina at Chapel Hill.

Krantz, D. H., Luce, R. D., Suppes, P., and Tversky, A. (1971). *Foundations of Measurement*. New York: Academic Press.

Kruskal, J. B., and Wish, M. (1978). *Multidimensional Scaling.* Vol. 07-011 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Null, C. H., and Sarle, W. S. (1982). "Multidimensional Scaling by Least Squares." In *Proceedings of the Seventh Annual SAS Users Group International Conference*, 643–650. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI82/Sugi-82-120%20Null%20Sarle.pdf.

Rabinowitz, G. (1976). "A Procedure for Ordering Object Pairs Consistent with the Multidimensional Unfolding Model." *Psychometrika* 41:349–373.

Ramsay, J. O. (1986). "The MLSCALE Procedure." In *SUGI Supplemental Library User's Guide, Version 5 Edition*. Cary, NC: SAS Institute Inc.

Schiffman, S. S., Reynolds, M. L., and Young, F. W. (1981). *Introduction to Multidimensional Scaling*. New York: Academic Press.

Torgerson, W. S. (1958). *Theory and Methods of Scaling*. New York: John Wiley & Sons.

Young, F. W. (1982). "Enhancements in ALSCAL-82." In *Proceedings of the Seventh Annual SAS Users Group International Conference*, 633–642. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI82/Sugi-82-119%20Young.pdf.

Young, F. W. (1987). *Multidimensional Scaling: History, Theory, and Applications.* Edited by R. M. Hamer. Hillsdale, NJ: Lawrence Erlbaum Associates.

Young, F. W., Lewyckyj, R., and Takane, Y. (1986). "The ALSCAL Procedure." In *SUGI Supplemental Library User's Guide, Version 5 Edition*. Cary, NC: SAS Institute Inc.

Chapter 82
The MI Procedure

Contents

Overview: MI Procedure . . . **6553**
Getting Started: MI Procedure . . . **6555**
Syntax: MI Procedure . . . **6558**
- PROC MI Statement . . . 6559
- BY Statement . . . 6563
- CLASS Statement . . . 6564
- EM Statement . . . 6564
- FCS Statement . . . 6565
- FREQ Statement . . . 6570
- MCMC Statement . . . 6570
- MNAR Statement . . . 6575
- MONOTONE Statement . . . 6577
- TRANSFORM Statement . . . 6581
- VAR Statement . . . 6583

Details: MI Procedure . . . **6583**
- Descriptive Statistics . . . 6583
- EM Algorithm for Data with Missing Values . . . 6584
- Statistical Assumptions for Multiple Imputation . . . 6585
- Missing Data Patterns . . . 6586
- Imputation Methods . . . 6587
- Monotone Methods for Data Sets with Monotone Missing Patterns . . . 6589
- Monotone and FCS Regression Methods . . . 6590
- Monotone and FCS Predictive Mean Matching Methods . . . 6591
- Monotone and FCS Discriminant Function Methods . . . 6592
- Monotone and FCS Logistic Regression Methods . . . 6594
- Monotone Propensity Score Method . . . 6597
- FCS Methods for Data Sets with Arbitrary Missing Patterns . . . 6598
- Checking Convergence in FCS Methods . . . 6600
- MCMC Method for Arbitrary Missing Multivariate Normal Data . . . 6601
- Producing Monotone Missingness with the MCMC Method . . . 6605
- MCMC Method Specifications . . . 6608
- Checking Convergence in MCMC . . . 6609
- Input Data Sets . . . 6611
- Output Data Sets . . . 6613
- Combining Inferences from Multiply Imputed Data Sets . . . 6615
- Multiple Imputation Efficiency . . . 6616

Number of Imputations 6617
Imputer's Model Versus Analyst's Model 6618
Missing Data Statistics 6619
Parameter Simulation versus Multiple Imputation 6620
Sensitivity Analysis for the MAR Assumption 6621
Multiple Imputation with Pattern-Mixture Models 6622
Specifying Sets of Observations for Imputation in Pattern-Mixture Models 6623
Adjusting Imputed Values in Pattern-Mixture Models 6624
Summary of Issues in Multiple Imputation 6628
Plot Options Superseded by ODS Graphics 6630
ODS Table Names 6635
ODS Graphics 6636
Examples: MI Procedure **6637**
Example 82.1: EM Algorithm for MLE 6639
Example 82.2: Monotone Propensity Score Method 6642
Example 82.3: Monotone Regression Method 6645
Example 82.4: Monotone Logistic Regression Method for CLASS Variables 6647
Example 82.5: Monotone Discriminant Function Method for CLASS Variables 6650
Example 82.6: FCS Methods for Continuous Variables 6652
Example 82.7: FCS Method for CLASS Variables 6655
Example 82.8: FCS Method with Trace Plot 6658
Example 82.9: MCMC Method 6663
Example 82.10: Producing Monotone Missingness with MCMC 6665
Example 82.11: Checking Convergence in MCMC 6667
Example 82.12: Saving and Using Parameters for MCMC 6669
Example 82.13: Transforming to Normality 6670
Example 82.14: Multistage Imputation 6673
Example 82.15: Creating Control-Based Pattern Imputation in Sensitivity Analysis . . 6676
Example 82.16: Adjusting Imputed Continuous Values in Sensitivity Analysis 6679
Example 82.17: Adjusting Imputed Classification Levels in Sensitivity Analysis . . . 6682
Example 82.18: Adjusting Imputed Values with Parameters in a Data Set 6684
Example 82.19: Selecting Variables for the Imputation Model 6689
Example 82.20: Selecting a Variable Order for Monotone Imputation 6695
References **6698**

Overview: MI Procedure

Missing values are an issue in a substantial number of statistical analyses. Most SAS statistical procedures exclude observations with any missing variable values from the analysis. These observations are called incomplete cases. While using only complete cases is simple, you lose information that is in the incomplete cases. Excluding observations with missing values also ignores the possible systematic difference between the complete cases and incomplete cases, and the resulting inference might not be applicable to the population of all cases, especially with a smaller number of complete cases.

Some SAS procedures use all the available cases in an analysis—that is, cases with useful information. For example, the CORR procedure estimates a variable mean by using all cases with nonmissing values for this variable, ignoring the possible missing values in other variables. PROC CORR also estimates a correlation by using all cases with nonmissing values for this pair of variables. This estimation might make better use of the available data, but the resulting correlation matrix might not be positive definite.

Another strategy is single imputation, in which you substitute a value for each missing value. Standard statistical procedures for complete data analysis can then be used with the filled-in data set. For example, each missing value can be imputed from the variable mean of the complete cases. This approach treats missing values as if they were known in the complete-data analyses. Single imputation does not reflect the uncertainty about the predictions of the unknown missing values, and the resulting estimated variances of the parameter estimates are biased toward zero (Rubin 1987, p. 13).

Instead of filling in a single value for each missing value, multiple imputation replaces each missing value with a set of plausible values that represent the uncertainty about the right value to impute (Rubin 1976, 1987). The multiply imputed data sets are then analyzed by using standard procedures for complete data and combining the results from these analyses. No matter which complete-data analysis is used, the process of combining results from different data sets is essentially the same.

Multiple imputation does not attempt to estimate each missing value through simulated values, but rather to represent a random sample of the missing values. This process results in valid statistical inferences that properly reflect the uncertainty due to missing values; for example, valid confidence intervals for parameters.

Multiple imputation inference involves three distinct phases:

1. The missing data are filled in m times to generate m complete data sets.
2. The m complete data sets are analyzed by using standard procedures.
3. The results from the m complete data sets are combined for the inference.

The MI procedure is a multiple imputation procedure that creates multiply imputed data sets for incomplete p-dimensional multivariate data. It uses methods that incorporate appropriate variability across the m imputations. The imputation method of choice depends on the patterns of missingness in the data and the type of the imputed variable.

A data set with variables $Y_1, Y_2, \ldots, Y_p$ (in that order) is said to have a *monotone missing pattern* when the event that a variable Y_j is missing for a particular individual implies that all subsequent variables $Y_k, k > j$, are missing for that individual.

For data sets with monotone missing patterns, the variables with missing values can be imputed sequentially with covariates constructed from their corresponding sets of preceding variables. To impute missing values

for a continuous variable, you can use a regression method (Rubin 1987, pp. 166–167), a predictive mean matching method (Heitjan and Little 1991; Schenker and Taylor 1996), or a propensity score method (Rubin 1987, pp. 124, 158; Lavori, Dawson, and Shera 1995). To impute missing values for a classification variable, you can use a logistic regression method when the classification variable has a binary, nominal, or ordinal response, or you can use a discriminant function method when the classification variable has a binary or nominal response.

For data sets with arbitrary missing patterns, you can use either of the following methods to impute missing values: a Markov chain Monte Carlo (MCMC) method (Schafer 1997) that assumes multivariate normality, or a fully conditional specification (FCS) method (Brand 1999; Van Buuren 2007) that assumes the existence of a joint distribution for all variables.

You can use the MCMC method to impute either all the missing values or just enough missing values to make the imputed data sets have monotone missing patterns. With a monotone missing data pattern, you have greater flexibility in your choice of imputation models, such as the monotone regression method that do not use Markov chains. You can also specify a different set of covariates for each imputed variable.

An FCS method does not start with an explicitly specified multivariate distribution for all variables, but rather uses a separate conditional distribution for each imputed variable. For each imputation, the process contains two phases: the preliminary filled-in phase followed by the imputation phase. At the filled-in phase, the missing values for all variables are filled in sequentially over the variables taken one at a time. These filled-in values provide starting values for these missing values at the imputation phase. At the imputation phase, the missing values for each variable are imputed sequentially for a number of burn-in iterations before the imputation.

As in methods for data sets with monotone missing patterns, you can use a regression method or a predictive mean matching method to impute missing values for a continuous variable, a logistic regression method to impute missing values for a classification variable with a binary or ordinal response, and a discriminant function method to impute missing values for a classification variable with a binary or nominal response.

After the m complete data sets are analyzed using standard SAS procedures, you can use the MIANALYZE procedure to generate valid statistical inferences about these parameters by combining results from the m analyses.

The number of imputations, m, must be specified in advance. The relative efficiency of an estimator based on a small number of imputations is high for cases with modest missing information (Rubin 1987, p. 114), and often a value of m as low as three or five is adequate (Rubin 1996, p. 480). For more information about relative efficiency, see the section "Multiple Imputation Efficiency" on page 6616.

Although a small number of imputations can suffice for high relative efficiency, they might not be adequate for other aspects of inference, such as confidence intervals and p-values. Recent studies examine these aspects and recommend much larger values of m than the traditionally advised values of three to five (Allison 2012; Van Buuren 2019, pp. 58–60). For more information, see the section "Number of Imputations" on page 6617.

Multiple imputation inference assumes that the model (variables) you used to analyze the multiply imputed data (the analyst's model) is the same as the model used to impute missing values in multiple imputation (the imputer's model). But in practice, the two models might not be the same. The consequences for different scenarios (Schafer 1997, pp. 139–143) are discussed in the section "Imputer's Model Versus Analyst's Model" on page 6618.

Multiple imputation usually assumes that the data are missing at random (MAR). That is, for a variable Y, the probability that an observation is missing depends only on the observed values of other variables, not

on the unobserved values of Y. The MAR assumption cannot be verified, because the missing values are not observed. For a study that assumes MAR, the sensitivity of inferences to departures from the MAR assumption should be examined.

The pattern-mixture model approach to sensitivity analysis models the distribution of a response as the mixture of a distribution of the observed responses and a distribution of the missing responses. Missing values can then be imputed under a plausible scenario for which the missing data are missing not at random (MNAR). If this scenario leads to a conclusion that is different from inference under MAR, then the conclusion under MAR is not robust to MNAR.

Getting Started: MI Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/migs.sas.)

The Fitness data described in the REG procedure are measurements of 31 individuals in a physical fitness course. See Chapter 105, "The REG Procedure," for more information.

The Fitness1 data set is constructed from the Fitness data set and contains three variables: Oxygen, RunTime, and RunPulse. Some values have been set to missing, and the resulting data set has an arbitrary pattern of missingness in these three variables.

```
*---------------------Data on Physical Fitness-------------------------*
| These measurements were made on men involved in a physical fitness   |
| course at N.C. State University. Certain values have been set to     |
| missing and the resulting data set has an arbitrary missing pattern. |
| Only selected variables of                                           |
| Oxygen (intake rate, ml per kg body weight per minute),              |
| Runtime (time to run 1.5 miles in minutes),                          |
| RunPulse (heart rate while running) are used.                        |
*----------------------------------------------------------------------*;
data Fitness1;
   input Oxygen RunTime RunPulse @@;
   datalines;
44.609  11.37  178     45.313  10.07  185
54.297   8.65  156     59.571    .      .
49.874   9.22    .     44.811  11.63  176
  .     11.95  176          .  10.85    .
39.442  13.08  174     60.055   8.63  170
50.541    .      .     37.388  14.03  186
44.754  11.12  176     47.273    .      .
51.855  10.33  166     49.156   8.95  180
40.836  10.95  168     46.672  10.00    .
46.774  10.25    .     50.388  10.08  168
39.407  12.63  174     46.080  11.17  156
45.441   9.63  164       .      8.92    .
45.118  11.08    .     39.203  12.88  168
45.790  10.47  186     50.545   9.93  148
48.673   9.40  186     47.920  11.50  170
47.467  10.50  170
;
```

Suppose that the data are multivariate normally distributed and the missing data are missing at random (MAR). That is, the probability that an observation is missing can depend on the observed variable values of the individual, but not on the missing variable values of the individual. See the section "Statistical Assumptions for Multiple Imputation" on page 6585 for a detailed description of the MAR assumption.

The following statements invoke the MI procedure and impute missing values for the Fitness1 data set:

```
proc mi data=Fitness1 seed=501213 mu0=50 10 180 out=outmi;
   mcmc;
   var Oxygen RunTime RunPulse;
run;
```

The "Model Information" table in Figure 82.1 describes the method used in the multiple imputation process. By default, the MCMC statement uses the Markov chain Monte Carlo (MCMC) method with a single chain to create 25 imputations. The posterior mode, the highest observed-data posterior density, with a noninformative prior, is computed from the expectation-maximization (EM) algorithm and is used as the starting value for the chain.

Figure 82.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FITNESS1
Method	MCMC
Multiple Imputation Chain	Single Chain
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	25
Number of Burn-in Iterations	200
Number of Iterations	100
Seed for random number generator	501213

The MI procedure takes 200 burn-in iterations before the first imputation and 100 iterations between imputations. In a Markov chain, the information in the current iteration influences the state of the next iteration. The burn-in iterations are iterations in the beginning of each chain that are used both to eliminate the series of dependence on the starting value of the chain and to achieve the stationary distribution. The between-imputation iterations in a single chain are used to eliminate the series of dependence between the two imputations.

The "Missing Data Patterns" table in Figure 82.2 lists distinct missing data patterns with their corresponding frequencies and percentages. An "X" means that the variable is observed in the corresponding group, and a "." means that the variable is missing. The table also displays group-specific variable means. The MI procedure sorts the data into groups based on whether the analysis variables are observed or missing. For a detailed description of missing data patterns, see the section "Missing Data Patterns" on page 6586.

Figure 82.2 Missing Data Patterns

Missing Data Patterns								
						Group Means		
Group	Oxygen	RunTime	RunPulse	Freq	Percent	Oxygen	RunTime	RunPulse
1	X	X	X	21	67.74	46.353810	10.809524	171.666667
2	X	X	.	4	12.90	47.109500	10.137500	.
3	X	.	.	3	9.68	52.461667	.	.
4	.	X	X	1	3.23	.	11.950000	176.000000
5	.	X	.	2	6.45	.	9.885000	.

After the completion of *m* imputations, the "Variance Information" table in Figure 82.3 displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences. It also displays the degrees of freedom for the total variance. The relative increase in variance due to missing values, the fraction of missing information, and the relative efficiency (in units of variance) for each variable are also displayed. A detailed description of these statistics is provided in the section "Combining Inferences from Multiply Imputed Data Sets" on page 6615.

Figure 82.3 Variance Information

Variance Information (25 Imputations)							
	Variance						
Variable	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Oxygen	0.037126	0.936472	0.975084	27.018	0.041231	0.039724	0.998414
RunTime	0.001317	0.065716	0.067086	27.593	0.020843	0.020451	0.999183
RunPulse	1.386290	3.394043	4.835784	18.43	0.424786	0.303282	0.988014

The "Parameter Estimates" table in Figure 82.4 displays the estimated mean and standard error of the mean for each variable. The inferences are based on the *t* distribution. The table also displays a 95% confidence interval for the mean and a *t* statistic with the associated *p*-value for the hypothesis that the population mean is equal to the value specified with the MU0= option. A detailed description of these statistics is provided in the section "Combining Inferences from Multiply Imputed Data Sets" on page 6615.

Figure 82.4 Parameter Estimates

Parameter Estimates (25 Imputations)										
Variable	Mean	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Oxygen	47.100050	0.987463	45.0740	49.1261	27.018	46.774347	47.434726	50.000000	-2.94	0.0067
RunTime	10.564553	0.259010	10.0336	11.0955	27.593	10.472584	10.636629	10.000000	2.18	0.0380
RunPulse	171.490381	2.199042	166.8781	176.1027	18.43	169.175377	173.421951	180.000000	-3.87	0.0011

In addition to the output tables, the procedure also creates a data set with imputed values. The imputed data sets are stored in the Outmi data set, with the index variable _Imputation_ indicating the imputation numbers. The data set can now be analyzed using standard statistical procedures with _Imputation_ as a BY variable.

The following statements list the first 10 observations of data set Outmi:

```
proc print data=outmi (obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

The table in Figure 82.5 shows that the precision of the imputed values differs from the precision of the observed values. You can use the ROUND= option to make the imputed values consistent with the observed values.

Figure 82.5 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Oxygen	RunTime	RunPulse
1	1	44.6090	11.3700	178.000
2	1	45.3130	10.0700	185.000
3	1	54.2970	8.6500	156.000
4	1	59.5710	8.0747	155.925
5	1	49.8740	9.2200	176.837
6	1	44.8110	11.6300	176.000
7	1	42.8857	11.9500	176.000
8	1	46.9992	10.8500	173.099
9	1	39.4420	13.0800	174.000
10	1	60.0550	8.6300	170.000

Syntax: MI Procedure

The following statements are available in the MI procedure:

PROC MI < *options* > ;
  BY *variables* ;
  CLASS *variables* ;
  EM < *options* > ;
  FCS < *options* > ;
  FREQ *variable* ;
  MCMC < *options* > ;
  MNAR *options* ;
  MONOTONE < *options* > ;
  TRANSFORM *transform* (*variables*< / *options* >) < . . . *transform* (*variables*< / *options* >) > ;
  VAR *variables* ;

The BY statement specifies groups in which separate multiple imputation analyses are performed.

The CLASS statement lists the classification variables in the VAR statement. If the MNAR statement is specified, the CLASS statement also includes the identification variables in the MNAR statement. Classification variables can be either character or numeric.

The EM statement uses the EM algorithm to compute the maximum likelihood estimate (MLE) of the data with missing values, assuming a multivariate normal distribution for the data.

The FREQ statement specifies the variable that represents the frequency of occurrence for other values in the observation.

For a data set with a monotone missing pattern, you can use the MONOTONE statement to specify applicable monotone imputation methods; otherwise, you can use either the MCMC statement assuming multivariate normality or the FCS method assuming a joint distribution for variables exists. Note that you can specify no more than one of these statements. When none of these three statements is specified, the MCMC method with its default options is used.

The FCS statement uses a multivariate imputation by chained equations method to impute values for a data set with an arbitrary missing pattern, assuming a joint distribution exists for the data.

The MCMC statement uses a Markov chain Monte Carlo method to impute values for a data set with an arbitrary missing pattern, assuming a multivariate normal distribution for the data.

The MNAR statement imputes missing values, assuming that the missing data are missing not at random (MNAR). The MNAR statement is applicable only if you also specify either an FCS or MONOTONE statement.

The MONOTONE statement specifies monotone methods to impute continuous and classification variables for a data set with a monotone missing pattern.

The TRANSFORM statement specifies the variables to be transformed before the imputation process; the imputed values of these transformed variables are reverse-transformed to the original forms before the imputation.

The VAR statement lists the numeric variables to be analyzed. If you omit the VAR statement, all numeric variables not listed in other statements are used.

The PROC MI statement is the only required statement for the MI procedure. The rest of this section provides detailed syntax information for each of these statements, beginning with the PROC MI statement. The remaining statements are presented in alphabetical order.

PROC MI Statement

PROC MI < *options* > **;**

The PROC MI statement invokes the MI procedure. Table 82.1 summarizes the options available in the PROC MI statement.

Table 82.1 Summary of Options in PROC MI Statement

Option	Description
Data Sets	
DATA=	Specifies the input data set
OUT=	Specifies the output data set with imputed values
Imputation Details	
MAXIMUM=	Specifies maximum values for imputed variable values
MINIMUM=	Specifies minimum values for imputed variable values
MINMAXITER=	Specifies the maximum number of iterations to impute values in the specified range

Table 82.1 *continued*

Option	Description
NIMPUTE=	Specifies the number of imputations
ROUND=	Specifies units to round imputed variable values
SEED=	Specifies the seed to start the random number generator
SINGULAR=	Specifies the singularity criterion
Statistical Analysis	
ALPHA=	Specifies the level for the confidence interval, $(1-\alpha)$
FLUX	Computes missing data statistics and displays a plot of the statistics
MU0=	Specifies means under the null hypothesis
SIMPLE	Computes univariate statistics and correlations
Control Output	
DISPLAYPATTERN=	Displays missing data patterns table
NOPRINT	Suppresses all displayed output

You can specify the following *options*:

ALPHA=α

specifies that confidence limits be constructed for the mean estimates with confidence level $100(1-\alpha)\%$, where $0 < \alpha < 1$. By default, ALPHA=0.05.

DATA=*SAS-data-set*

names the SAS data set to be analyzed by PROC MI. By default, the procedure uses the most recently created SAS data set.

DISPLAYPATTERN=ALL | NOMEANS | NONE

creates a missing data patterns table. You can specify the following values:

ALL displays both the missing data patterns and the group means in the table.

NOMEANS displays only the missing data patterns in the table.

NONE does not display the missing data patterns table.

By default, DISPLAYPATTERN=ALL.

FLUX

computes a set of inbound and outbound missing data statistics and displays a scatter plot of the influx/outflux statistics. For a detailed description of these statistics, see the section "Missing Data Statistics" on page 6619.

MAXIMUM=*numbers*

specifies maximum values for imputed variables. When an intended imputed value is greater than the maximum, PROC MI redraws another value for imputation. If only one number is specified, that number is used for all variables. If more than one number is specified, you must use a VAR statement, and the specified numbers must correspond to variables in the VAR statement. The default number is a missing value, which indicates no restriction on the maximum for the corresponding variable

The MAXIMUM= option is related to the MINIMUM= and ROUND= options, which are used to make the imputed values more consistent with the observed variable values. These options apply only if you use the MCMC method, the monotone regression method, or the FCS regression method. For more information about these methods, see the section "Imputation Methods" on page 6587.

When you specify a maximum for the first variable only, you must also specify a missing value after the maximum. Otherwise, the maximum is used for all variables. For example, "MAXIMUM= 100 ." sets a maximum of 100 only for the first analysis variable and no maximum for the remaining variables. "MAXIMUM= . 100" sets a maximum of 100 only for the second analysis variable and no maximum for the other variables.

MINIMUM=*numbers*

specifies the minimum values for imputed variables. When an intended imputed value is less than the minimum, PROC MI redraws another value for imputation. If only one number is specified, that number is used for all variables. If more than one number is specified, you must use a VAR statement, and the specified numbers must correspond to variables in the VAR statement. The default number is a missing value, which indicates no restriction on the minimum for the corresponding variable

MINMAXITER=*number*

specifies the maximum number of iterations for imputed values to be in the specified range when you also specify the MINIMUM= or MAXIMUM= option. By default, MINMAXITER=100.

MU0=*numbers*

THETA0=*numbers*

specifies the parameter values $\boldsymbol{\mu}_0$ under the null hypothesis $\boldsymbol{\mu} = \boldsymbol{\mu}_0$ for the population means corresponding to the analysis variables. Each hypothesis is tested with a t test. If only one number is specified, that number is used for all variables. If more than one number is specified, you must use a VAR statement, and the specified numbers must correspond to variables in the VAR statement. By default, MU0=0.

If a variable is transformed as specified in a TRANSFORM statement, then the same transformation for that variable is also applied to its corresponding specified MU0= value in the t test. If the parameter values $\boldsymbol{\mu}_0$ for a transformed variable are not specified, then a value of zero is used for the resulting $\boldsymbol{\mu}_0$ after transformation.

NIMPUTE=*n* | **PCTMISSING** < (*range-options*) >

specifies the number of imputations. NIMPUTE=*n* specifies the number explicitly, and NIMPUTE=PCTMISSING uses the percentage of incomplete cases as the number of imputations. By default, NIMPUTE=25.

When you specify NIMPUTE=PCTMISSING, the number of imputations is the resulting percentage rounded up to an integer. You can use the following *range-options* to set the range for the number of imputations:

MIN=*min*

specifies the minimum number of imputations, $2 \leq min \leq 100$. If the resulting number of imputations is less than *min*, then *min* is used. By default, MIN=5.

MAX=*max*
specifies the maximum number of imputations, $2 \leq max \leq 100$. If the resulting number of imputations is greater than *max*, then *max* is used. By default, MAX=50.

The classic advice of using only a small number of imputations is based on considerations of relative efficiency. Recent studies, based on other aspects such as confidence intervals and p-values, recommend a much larger number of imputations. Thus, the default number of imputations was increased from 5 to 25 starting in SAS/STAT 14.1. For more information, see the section "Number of Imputations" on page 6617.

You can specify NIMPUTE=0 to skip the imputation. In this case, only tables of model information, missing data patterns, descriptive statistics (SIMPLE option), and the MLE from the EM algorithm (EM statement) are displayed.

NOPRINT
suppresses the display of all output. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

OUT=*SAS-data-set*
creates an output SAS data set that contains imputation results. The data set includes an index variable, _Imputation_, to identify the imputation number. For each imputation, the data set contains all variables in the input data set with missing values being replaced by the imputed values. See the section "Output Data Sets" on page 6613 for a description of this data set.

ROUND=*numbers*
specifies the units to round variables in the imputation. If only one *number* is specified, it is used for all continuous variables. If more than one *number* is specified, you must use a VAR statement, and the specified *numbers* must correspond to variables in the VAR statement. When the classification variables are listed in the VAR statement, their corresponding roundoff units are not used. The default number is a missing value, which indicates no rounding for imputed variables.

When specifying a roundoff unit for the first variable only, you must also specify a missing value after the roundoff unit. Otherwise, the roundoff unit is used for all variables. For example, the option "ROUND= 10 ." sets a roundoff unit of 10 for the first analysis variable only and no rounding for the remaining variables. The option "ROUND= . 10" sets a roundoff unit of 10 for the second analysis variable only and no rounding for other variables.

The ROUND= option sets the precision of imputed values. For example, with a roundoff unit of 0.001, each value is rounded to the nearest multiple of 0.001. That is, each value has three significant digits after the decimal point. See Example 82.3 for an illustration of this option.

SEED=*number*
specifies a positive integer to start the pseudo-random number generator. The default is a value generated from reading the time of day from the computer's clock. However, in order to duplicate the results under identical situations, you must use the same value of the seed explicitly in subsequent runs of the MI procedure.

The seed information is displayed in the "Model Information" table so that the results can be reproduced by specifying this seed with the SEED= option. You need to specify the same seed number in the future to reproduce the results.

SIMPLE

displays simple descriptive univariate statistics and pairwise correlations from available cases. For a detailed description of these statistics, see the section "Descriptive Statistics" on page 6583.

SINGULAR=*p*

specifies the criterion for determining the singularity of a covariance matrix based on standardized variables, where $0 < p < 1$. By default, SINGULAR=1E–8.

Suppose that $\mathbf{S}$ is a covariance matrix and v is the number of variables in $\mathbf{S}$. Based on the spectral decomposition $\mathbf{S} = \mathbf{\Gamma\Lambda\Gamma}'$, where $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues λ_j, $j = 1, \ldots, v$, where $\lambda_i \geq \lambda_j$ when $i < j$, and $\mathbf{\Gamma}$ is a matrix with the corresponding orthonormal eigenvectors of $\mathbf{S}$ as columns, $\mathbf{S}$ is considered singular when an eigenvalue λ_j is less than $p\bar{\lambda}$, where the average $\bar{\lambda} = \sum_{k=1}^{v} \lambda_k / v$.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC MI to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the MI procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* **;**

The CLASS statement specifies the classification variables in the VAR statement. Classification variables can be either character or numeric. The CLASS statement must be used in conjunction with either an FCS or MONOTONE statement.

Classification levels are determined from the formatted values of the classification variables. See "The FORMAT Procedure" in the *Base SAS Procedures Guide* for details.

EM Statement

EM < *options* > **;**

The expectation-maximization (EM) algorithm is a technique for maximum likelihood estimation in parametric models for incomplete data. The EM statement uses the EM algorithm to compute the MLE for $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the means and covariance matrix, of a multivariate normal distribution from the input data set with missing values. Either the means and covariances from complete cases or the means and standard deviations from available cases can be used as the initial estimates for the EM algorithm. You can also specify the correlations for the estimates from available cases.

You can also use the EM statement with the NIMPUTE=0 option in the PROC MI statement to compute the EM estimates without multiple imputation, as shown in Example 82.1.

The following seven options are available with the EM statement (in alphabetical order):

CONVERGE=*p*
XCONV=*p*
sets the convergence criterion. The value must be between 0 and 1. The iterations are considered to have converged when the change in the parameter estimates between iteration steps is less than *p* for each parameter—that is, for each of the means and covariances. For each parameter, the change is a relative change if the parameter is greater than 0.01 in absolute value; otherwise, it is an absolute change. By default, CONVERGE=1E–4.

INITIAL=CC | AC | AC(R=*r***)**
sets the initial estimates for the EM algorithm. INITIAL=CC uses the means and covariances from complete cases; INITIAL=AC uses the means and standard deviations from available cases, and the correlations are set to zero; and INITIAL=AC(R= *r*) uses the means and standard deviations from available cases with correlation *r*, where $-1/(p-1) < r < 1$ and *p* is the number of variables to be analyzed. By default, INITIAL=AC.

ITPRINT
prints the iteration history in the EM algorithm.

MAXITER=*number*
specifies the maximum number of iterations for the EM algorithm. By default, MAXITER=200.

OUT=*SAS-data-set*
creates an output SAS data set that contains results from the EM algorithm. The data set contains all variables in the input data set, with missing values being replaced by the expected values from the EM algorithm. See the section "Output Data Sets" on page 6613 for a description of this data set.

OUTEM=*SAS-data-set*
creates an output SAS data set of TYPE=COV that contains the MLE of the parameter vector $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. These estimates are computed with the EM algorithm. See the section "Output Data Sets" on page 6613 for a description of this output data set.

OUTITER < (*options* **) > =***SAS-data-set*
creates an output SAS data set of TYPE=COV that contains parameters for each iteration. The data set includes a variable named _Iteration_ to identify the iteration number. The parameters in the output data set depend on the options specified. You can specify the MEAN and COV options to output the mean and covariance parameters. When no options are specified, the output data set contains the mean parameters for each iteration. See the section "Output Data Sets" on page 6613 for a description of this data set.

FCS Statement

FCS < *options* > **;**

The FCS statement specifies a multivariate imputation by fully conditional specification methods. If you specify an FCS statement, you must also specify a VAR statement.

Table 82.2 summarizes the options available for the FCS statement.

Table 82.2 Summary of Options in FCS Statement

Option	Description
Imputation Details	
NBITER=	Specifies the number of burn-in iterations
Data Set	
OUTITER=	Outputs parameter estimates used in iterations
ODS Graphics Output	
PLOTS=TRACE	Displays trace plots
Imputation Methods	
DISCRIM	Specifies the discriminant function method
LOGISTIC	Specifies the logistic regression method
REG	Specifies the regression method
REGPMM	Specifies the predictive mean matching method

The following options are available for the FCS statement in addition to the imputation methods specified (in alphabetical order):

NBITER=*number*
specifies the number of burn-in iterations before each imputation. By default, NBITER=20.

OUTITER < (*options*) > =*SAS-data-set*
creates an output SAS data set of TYPE=COV that contains parameters used in the imputation step for each iteration. The data set includes variables named _Imputation_ and _Iteration_ to identify the imputation number and iteration number.

The parameters in the output data set depend on the options specified. You can specify the options MEAN and STD to output parameters of means and standard deviations, respectively. When no options are specified, the output data set contains the mean parameters used in the imputation step for each iteration. See the section "Output Data Sets" on page 6613 for a description of this data set.

PLOTS < (LOG) > < = TRACE < (*trace-options*) > >
requests statistical graphics of trace plots from iterations via the Output Delivery System (ODS).

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc mi data=Fitness1 seed=501213 mu0=50 10 180;
   mcmc plots=(trace(mean(Oxygen)) acf(mean(Oxygen)));
   var Oxygen RunTime RunPulse;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The global plot option LOG requests that the logarithmic transformations of parameters be used. By default, PLOTS=TRACE(MEAN).

The available *trace-options* are as follows:

MEAN < (*variables*) >
displays plots of means for continuous variables in the list. When the MEAN option is specified without variables, all continuous variables are used.

STD < (*variables*) >
displays plots of standard deviations for continuous variables in the list. When the STD option is specified without variables, all continuous variables are used.

The discriminant function, logistic regression, regression, and predictive mean matching methods are available in the FCS statement. You specify each method with the syntax

method < (< *imputed* < = *effects* > > < /*options* >) >

That is, for each method, you can specify the imputed variables and, optionally, a set of effects to impute these variables. Each effect is a variable or a combination of variables in the VAR statement. The syntax for the specification of effects is the same as for the GLM procedure. See Chapter 53, "The GLM Procedure," for more information.

One general form of an effect involving several variables is

*X1 * X2 * A * B * C (D E)*

where A, B, C, D, and E are classification variables and X1 and X2 are continuous variables.

When an FCS statement is used without specifying any methods, the regression method is used for all imputed continuous variables and the discriminant function method is used for all imputed classification variables. In this case, for each imputed continuous variable, all other variables in the VAR statement are used as the covariates, and for each imputed classification variable, all other continuous variables in the VAR statement are used as the covariates.

When a method for continuous variables is specified without imputed variables, the method is used for all continuous variables in the VAR statement that are not specified in other methods. Similarly, when a method for classification variables is specified without imputed variables, the method is used for all classification variables in the VAR statement that are not specified in other methods.

For each imputed variable that does not use the discriminant function method, if no covariates are specified, then all other variables in the VAR statement are used as the covariates. That is, each continuous variable is used as a regressor effect, and each classification variable is used as a main effect. For an imputed variable that uses the discriminant function method, if no covariates are specified, then all other variables in the VAR statement are used as the covariates with the CLASSEFFECTS=INCLUDE option, and all other continuous variables in the VAR statement are used as the covariates with the CLASSEFFECTS=EXCLUDE option (which is the default).

With an FCS statement, the variables are imputed sequentially in the order specified in the VAR statement. For a continuous variable, you can use a regression method or a regression predicted mean matching method to impute missing values. For a nominal classification variable, you can use either a discriminant function method or a logistic regression method (generalized logit model) to impute missing values without using the ordering of the class levels. For an ordinal classification variable, you can use a logistic regression method (cumulative logit model) to impute missing values by using the ordering of the class levels. For a binary classification variable, either a discriminant function method or a logistic regression method can be used. By default, a regression method is used for a continuous variable, and a discriminant function method is used for a classification variable.

Note that except for the regression method, all other methods impute values from the observed values. See the section "FCS Methods for Data Sets with Arbitrary Missing Patterns" on page 6598 for a detailed description of the FCS methods.

You can specify the following imputation methods in an FCS statement (in alphabetical order):

DISCRIM **<(** *imputed* **<** **=** *effects* **>** **<** / *options* **>** **)** **>**

specifies the discriminant function method of classification variables. The available options are as follows:

CLASSEFFECTS=EXCLUDE | INCLUDE

specifies whether the CLASS variables are used as covariate effects. CLASSEFFECTS=EXCLUDE excludes the CLASS variables from covariate effects, and CLASSEFFECTS=INCLUDE includes the CLASS variables as covariate effects. By default, CLASSEFFECTS=EXCLUDE.

DETAILS
displays the group means and pooled covariance matrix used in each imputation.

PCOV=FIXED | POSTERIOR
specifies the pooled covariance used in the discriminant method. PCOV=FIXED uses the observed-data pooled covariance matrix for each imputation, and PCOV=POSTERIOR draws a pooled covariance matrix from its posterior distribution. By default, PCOV=POSTERIOR.

PRIOR=EQUAL | JEFFREYS <=*c*> | PROPORTIONAL | RIDGE <=*d*>
specifies the prior probabilities of group membership. PRIOR=EQUAL sets the prior probabilities equal for all groups; PRIOR=JEFFREYS <=*c*> specifies a noninformative prior, $0 < c < 1$; PRIOR=PROPORTIONAL sets the prior probabilities proportion to the group sample sizes; and PRIOR=RIDGE <=*d*> specifies a ridge prior, $d > 0$. If the noninformative prior c is not specified, c=0.5 is used. If the ridge prior d is not specified, d=0.25 is used. By default, PRIOR=JEFFREYS.

See the section "Monotone and FCS Discriminant Function Methods" on page 6592 for a detailed description of the method.

LOGISTIC <(*imputed* **<** = *effects* **>** **<**/ *options* **>** **) >**
specifies the logistic regression method for classification variables. The available options are as follows:

DESCENDING
reverses the sort order for the levels of the response variables.

DETAILS
displays the regression coefficients in the logistic regression model used in each imputation.

LIKELIHOOD=NOAUGMENT
LIKELIHOOD=AUGMENT <(WEIGHT=*w* | NPARM <(MULT=*m*)>) >
specifies whether to add new observations to the likelihood function in the computation of maximum likelihood estimates. LIKELIHOOD=AUGMENT adds observations in each response group to the likelihood function, and LIKELIHOOD=NOAUGMENT makes no adjustment to the likelihood function. By default, LIKELIHOOD=NOAUGMENT.

The LIKELIHOOD=AUGMENT option is useful when the maximum likelihood parameter estimates do not exist. When LIKELIHOOD=AUGMENT, each added observation contributes the same weight, and the WEIGHT= option specifies the total added weight:

WEIGHT=*w*
explicitly specifies the total added weight w.

WEIGHT=NPARM <(MULT=*m*)>
uses the number of parameters in the logistic regression model as the total added weight. For example, for a simple binary logistic regression model that consists only of p continuous effects, the added weight is p+1. The MULT=m option specifies the multiplier for the total added weight, $0 < m \leq 1$, and the resulting total added weight is m times the number of parameters in the model. By default, MULT=1.

BY default, WEIGHT=NPARM. You can specify either the MULT=m suboption in WEIGHT=NPARM or the WEIGHT=w option to use a different total added weight in the computation of maximum likelihood estimates. For example, if the ratio between the number

of parameters and the number of available observations (before augmentation) is large, you can use either MULT=*m* or WEIGHT=*w* to reduce the weight for the added observations (that is, reduce the effect from the added observations in the computation of maximum likelihood estimates). For more information about the augmented data approach, see the section "Logistic Regression with Augmented Data" on page 6597.

LINK=GLOGIT | LOGIT
specifies the link function that links the response probabilities to the linear predictors. LINK=LOGIT uses the log odds function to fit the binary logit model when there are two response categories and to fit the cumulative logit model when there are more than two response categories. LINK=GLOGIT uses the generalized logit function to fit the generalized logit model, in which each nonreference category is contrasted with the last category. By default, LINK=LOGIT.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the response variable. ORDER=DATA sorts by the order of appearance in the input data set; ORDER=FORMATTED sorts by their external formatted values; ORDER=FREQ sorts by the descending frequency counts; and ORDER=INTERNAL sorts by the unformatted values. By default, ORDER=FORMATTED.

See the section "Monotone and FCS Logistic Regression Methods" on page 6594 for a detailed description of the method.

REG | REGRESSION < (*imputed* < = *effects* > < / DETAILS >) >
specifies the regression method of continuous variables. The DETAILS option displays the regression coefficients in the regression model used in each imputation.

With a regression method, you can use the MAXIMUM=, MINIMUM=, and ROUND= options to make the imputed values more consistent with the observed variable values.

See the section "Monotone and FCS Regression Methods" on page 6590 for a detailed description of the method.

REGPMM < (*imputed* < = *effects* > < / *options* >) >
REGPREDMEANMATCH < (*imputed* < = *effects* > < / *options* >) >
specifies the predictive mean matching method for continuous variables. This method is similar to the regression method except that it imputes a value randomly from a set of observed values whose predicted values are closest to the predicted value for the missing value from the simulated regression model (Heitjan and Little 1991; Schenker and Taylor 1996).

The available options are DETAILS and K=. The DETAILS option displays the regression coefficients in the regression model used in each imputation. The K= option specifies the number of closest observations to be used in the selection. By default, K=5.

See the section "Monotone and FCS Predictive Mean Matching Methods" on page 6591 for a detailed description of the method.

With an FCS statement, the missing values of variables in the VAR statement are imputed. After the initial filled in, these variables with missing values are imputed sequentially in the order specified in the VAR statement in each iteration. For example, the following MI procedure statements use the regression method to impute variable y1 from effect y2, the regression method to impute variable y3 from effects y1 and y2, the logistic regression method to impute variable c1 from effects y1, y2, and y1 ∗ y2, and the default regression method for continuous variables to impute variable y2 from effects y1, y3, and c1:

```
proc mi;
   class c1;
   fcs reg(y1= y2);
   fcs reg(y3= y1 y2);
   fcs logistic(c1= y1 y2 y1*y2);
   var y1 y2 y3 c1;
run;
```

FREQ Statement

FREQ *variable* **;**

To run a procedure on an input data set that contains observations that occur multiple times, you can use a variable in the data set to represent how frequently observations occur and specify a FREQ statement with the name of that variable as its argument (*variable*) when you run the procedure.

When you specify a FREQ statement in other SAS procedures, they treat the data set as if each observation appeared *n* times, where *n* is the value of *variable* in the observation. However, PROC MI treats the data set differently because as PROC MI imputes each missing value in each observation, it generates only one imputed value for that missing value. That is, when you specify a FREQ *variable*, each imputed observation (with its imputed value in place of the missing value) is treated as if it appeared *n* times. In contrast, if an observation actually occurs *n* times in the data set, the missing value at each occurrence is imputed separately, and the resulting *n* observations are not identical.

PROC MI uses only the integer portion of each value of *variable*; if any value is less than 1, PROC MI does not use the corresponding observation in the analysis. When PROC MI calculates significance probabilities, it considers the total number of observations to be equal to the sum of the values of *variable*.

MCMC Statement

MCMC < *options* > **;**

The MCMC statement specifies the details of the MCMC method for imputation.

Table 82.3 summarizes the options available for the MCMC statement.

Table 82.3 Summary of Options in MCMC Statement

Option	Description
Data Sets	
INEST=	Inputs parameter estimates for imputations
OUTEST=	Outputs parameter estimates used in imputations
OUTITER=	Outputs parameter estimates used in iterations
Imputation Details	
CHAIN=	Specifies single or multiple chain
IMPUTE=	Specifies monotone or full imputation
INITIAL=	Specifies initial parameter estimates for MCMC

Table 82.3 *continued*

Option	Description
MONOTONESEARCH	Requests an optimized variable order for monotone imputation
NBITER=	Specifies the number of burn-in iterations for each chain
NITER=	Specifies the number of iterations between imputations in a chain
PRIOR=	Specifies the prior parameter information
START=	Specifies starting parameters
ODS Graphics Output	
PLOTS=ACF	Displays autocorrelation plots
PLOTS=TRACE	Displays trace plots
Printed Output	
DISPLAYINIT	Displays initial parameter values for MCMC
WLF	Displays the worst linear function

The following options are available for the MCMC statement (in alphabetical order).

CHAIN=SINGLE | MULTIPLE

specifies whether a single chain is used for all imputations or a separate chain is used for each imputation. By default, CHAIN=SINGLE.

DISPLAYINIT

displays initial parameter values in the MCMC method for each imputation.

IMPUTE=FULL | MONOTONE

specifies whether a full-data imputation is used for all missing values or a monotone-data imputation is used for a subset of missing values to make the imputed data sets have a monotone missing pattern. By default, IMPUTE=FULL. When IMPUTE=MONOTONE, the order in the VAR statement is used to complete the monotone pattern.

INEST=*SAS-data-set*

names a SAS data set of TYPE=EST that contains parameter estimates for imputations. These estimates are used to impute values for observations in the DATA= data set. A detailed description of the data set is provided in the section "Input Data Sets" on page 6611.

INITIAL=EM < (*options***) >**

INITIAL=INPUT=*SAS-data-set*

specifies the initial mean and covariance estimates for the MCMC method. By default, INITIAL=EM.

You can specify INITIAL=INPUT=*SAS-data-set* to read the initial estimates of the mean and covariance matrix for each imputation from a SAS data set. See the section "Input Data Sets" on page 6611 for a description of this data set.

When INITIAL=EM, PROC MI derives parameter estimates for a posterior mode, the highest observed-data posterior density, from the EM algorithm. The MLE from the EM algorithm is used to start the EM algorithm for the posterior mode, and the resulting EM estimates are used to begin the MCMC method. The prior information specified in the PRIOR= option is also used in the process to compute the posterior mode.

The following four options are available when INITIAL=EM:

BOOTSTRAP < =*number* >
requests bootstrap resampling, which uses a simple random sample with replacement from the input data set for the initial estimate. You can explicitly specify the number of observations in the random sample. Alternatively, you can implicitly specify the number of observations in the random sample by specifying the proportion $p, 0 < p <= 1$, to request $[np]$ observations in the random sample, where n is the number of observations in the data set and $[np]$ is the integer part of np. This produces an overdispersed initial estimate that provides different starting values for the MCMC method. If you specify the BOOTSTRAP option without the *number*, $p = 0.75$ is used by default.

CONVERGE=*p*
XCONV=*p*
sets the convergence criterion. The value must be between 0 and 1. The iterations are considered to have converged when the change in the parameter estimates between iteration steps is less than *p* for each parameter—that is, for each of the means and covariances. For each parameter, the change is a relative change if the parameter is greater than 0.01 in absolute value; otherwise, it is an absolute change. By default, CONVERGE=1E–4.

ITPRINT
prints the iteration history in the EM algorithm for the posterior mode.

MAXITER=*number*
specifies the maximum number of iterations for the EM algorithm. By default, MAXITER=200.

MONOTONESEARCH(< *monotonesearch-options* >)
requests an optimized variable order for monotone imputation. When you specify this option, PROC MI searches for a permutation of the analysis variables to minimize the computational cost of imputing to a monotone pattern. For more information, see the section "MONOTONESEARCH Option" on page 6607. You can specify the following *monotonesearch-options*:

MAXITER=*number*
specifies the maximum number of iterations for the simulated annealing algorithm. By default, MAXITER=2500.

METHOD=EXACT | SA
specifies whether to find an exact solution by computing all possible order permutations or to use simulated annealing to optimize the variable order. By default, if the data set contains 10 or fewer variables, METHOD=EXACT; otherwise METHOD=SA.

STARTTEMP=*number*
specifies the starting temperature for the simulated annealing algorithm. By default, STARTTEMP=10.

NBITER=*number*
specifies the number of burn-in iterations before the first imputation in each chain. By default, NBITER=200.

NITER=*number*
specifies the number of iterations between imputations in a single chain. By default, NITER=100.

OUTEST=*SAS-data-set*
creates an output SAS data set of TYPE=EST. The data set contains parameter estimates used in each imputation. The data set also includes a variable named _Imputation_ to identify the imputation number. See the section "Output Data Sets" on page 6613 for a description of this data set.

OUTITER < (*options*) > =*SAS-data-set*
creates an output SAS data set of TYPE=COV that contains parameters used in the imputation step for each iteration. The data set includes variables named _Imputation_ and _Iteration_ to identify the imputation number and iteration number.

The parameters in the output data set depend on the options specified. You can specify the options MEAN, STD, COV, LR, LR_POST, and WLF to output parameters of means, standard deviations, covariances, –2 log LR statistic, –2 log LR statistic of the posterior mode, and the worst linear function, respectively. When no options are specified, the output data set contains the mean parameters used in the imputation step for each iteration. See the section "Output Data Sets" on page 6613 for a description of this data set.

PLOTS < (LOG) > < = *plot-request* >

PLOTS < (LOG) > < = (*plot-request* < . . . *plot-request* >) >
requests statistical graphics via the Output Delivery System (ODS). To request these graphs, ODS Graphics must be enabled and you must specify options in the MCMC statement. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

The global plot option LOG requests that the logarithmic transformations of parameters be used. The plot request options include the following:

ACF < (*acf-options*) >
displays plots of the autocorrelation function of parameters from iterations. The default is ACF(MEAN).

ALL
produces all appropriate plots.

NONE
suppresses all plots.

TRACE < (*trace-options*) >
displays trace plots of parameters from iterations. The default is TRACE(MEAN).

The available *acf-options* are as follows:

NLAG=*n*
specifies the maximum lag of the series. By default, NLAG=20. The autocorrelations at each lag are displayed in the graph.

COV <(< *variables* > < *variable1*variable2* > . . .)>
displays plots of variances for variables in the list and covariances for pairs of variables in the list. When the COV option is specified without variables, variances for all variables and covariances for all pairs of variables are used.

MEAN < (*variables*) >
displays plots of means for variables in the list. When the MEAN option is specified without variables, all variables are used.

WLF
displays the plot for the worst linear function.

The available *trace-options* are as follows:

COV <(< *variables* > < *variable1*variable2* > . . .)>

displays plots of variances for variables in the list and covariances for pairs of variables in the list. When the COV option is specified without variables, variances for all variables and covariances for all pairs of variables are used.

MEAN < (*variables*) >
displays plots of means for variables in the list. When the MEAN option is specified without variables, all variables are used.

WLF
displays the plot of the worst linear function.

PRIOR=*name*

PRIOR=JEFFREYS | RIDGE=*number* **| INPUT=***SAS-data-set*
specifies the prior information for the means and covariances. The PRIOR=JEFFREYS option specifies a noninformative prior, the RIDGE=*number* option specifies a ridge prior, and the INPUT=*SAS-data-set* option specifies a data set that contains prior information.

For a detailed description of the prior information, see the section "Bayesian Estimation of the Mean Vector and Covariance Matrix" on page 6602 and the section "Posterior Step" on page 6603. If you do not specify the PRIOR= option, the default is PRIOR=JEFFREYS.

The PRIOR=INPUT= option specifies a TYPE=COV data set from which the prior information of the mean vector and the covariance matrix is read. See the section "Input Data Sets" on page 6611 for a description of this data set.

START=VALUE | DIST
specifies that the initial parameter estimates are used either as the starting value (START=VALUE) or as the starting distribution (START=DIST) in the first imputation step of each chain. If you specify the IMPUTE=MONOTONE option, then START=VALUE is used in the procedure. By default, START=VALUE.

WLF

displays the worst linear function of parameters. This scalar function of parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ is "worst" in the sense that its values from iterations converge most slowly among parameters. For a detailed description of this statistic, see the section "Worst Linear Function of Parameters" on page 6610.

MNAR Statement

MNAR *options* **;**

The MNAR statement imputes missing values by using the pattern-mixture model approach, assuming the missing data are missing not at random (MNAR), which is described in the section "Multiple Imputation with Pattern-Mixture Models" on page 6622. By comparing inferential results for these values to results for imputed values that are obtained under the missing at random (MAR) assumption, you can assess the sensitivity of the conclusions to the MAR assumption. The inference under MAR is questionable if it leads to results that are different from the results for a plausible MNAR scenario.

There are two main options in the MNAR statement, MODEL and ADJUST. You use the MODEL option to specify a subset of observations from which imputation models are to be derived for specified variables. You use the ADJUST option to specify an imputed variable and adjustment parameters (such as shift and scale) for adjusting the imputed variable values for a specified subset of observations.

The MNAR statement is applicable only if it is used along with a MONOTONE statement or an FCS statement. For a detailed explanation of the imputation process for the MNAR statement and how this process is implemented differently using the MONOTONE and FCS statements, see the section "Multiple Imputation with Pattern-Mixture Models" on page 6622.

MODEL(*imputed-variables* / *model-options* **)**

specifies a set of *imputed-variables* in the VAR statement and the subset of observations from which the imputation models for these variables are to be derived. You can specify multiple MODEL options in the MNAR statement, but only one MODEL option for each imputed variable.

When an imputed variable that is listed in the VAR statement is not specified as an *imputed-variable* in the MODEL option, all available observations are used to construct the imputation for that variable.

The following *model-options* provide various ways to specify the subset of observations:

MODELOBS=CCMV < (K= *k***) >**

MODELOBS=NCMV < (K= *k***) >**

MODELOBS=(*obs-variable*=*character-list***)**

identifies the subset of observations that are used to derive the imputation models.

When you use the MNAR statement along with an FCS statement, only the MODELOBS=(*obs-variable*=*character-list*) *model-option* is applicable. When you use the MNAR statement along with a MONOTONE statement, all three *model-options* are applicable.

MODELOBS=CCMV specifies the complete-case missing values method (Little 1993; Molenberghs and Kenward 2007, p. 35). This method derives the imputation model from the group of observations for which all the variables are observed.

MODELOBS=CCMV(K=*k*) uses the *k* groups of observations together with as many observed variables as possible to derive the imputation models. For a data set that has a monotone missing

pattern and p variables, there are at most p groups of observations for which the same number of variables is observed. The default is K=1, which uses observations from the group for which all the variables in the VAR statement are observed (this corresponds to MODELOBS=CCMV).

MODELOBS=NCMV specifies the neighboring-case missing values method (Molenberghs and Kenward 2007, pp. 35–36). For an imputed variable Y_j, this method uses the observations for which Y_j is observed and Y_{j+1} is missing.

For an imputed variable Y_j, MODELOBS=NCMV(K=k) uses the k closest groups of observations for which Y_j is observed and for which Y_{j+k} is missing. The default is K=1, which corresponds to MODELOBS=NCMV.

MODELOBS=(*obs-variable*=*character-list*) identifies the subset of observations from which the imputation models are to be derived in terms of specified levels of the *obs-variable*. You must also specify the *obs-variable* in the CLASS statement. If you include the *obs-variable* in the VAR statement, it must be completely observed.

For a detailed description of the options for specifying the observations for deriving the imputation model. see the section "Specifying Sets of Observations for Imputation in Pattern-Mixture Models" on page 6623.

ADJUST(*imputed-variable* / *adjust-options* **)**

ADJUST(*imputed-variable* **(EVENT='***level***')** / *adjust-options* **)**

specifies an *imputed-variable* in the VAR statement and the subset of observations from which the imputed values for the variable are to be adjusted. If the *imputed-variable* is a classification variable, you must specify the EVENT= option to identify the response category to which the adjustments are applied. The *adjust-options* specify the subset of observations and the adjustment parameters.

You can specify multiple ADJUST options. Each ADJUST option adjusts the imputed values of an *imputed-variable* for the subset of observations that are specified in the option. The ADJUST option applies only to continuous *imputed-variables* whose values are imputed using the regression and predictive mean matching methods, and to classification *imputed-variables* whose values are imputed using the logistic regression method.

You can use the following *adjust-option* to specify the subset of observations to be adjusted:

ADJUSTOBS= (*obs-variable*=*character-list* **)**

identifies the subset of observations for which the imputed values of *imputed-variable* are to be adjusted in terms of specified levels of the *obs-variable*. You must also specify the *obs-variable* in the CLASS statement. If the *obs-variable* appears in the VAR statement, it must be completely observed.

If you do not specify the ADJUSTOBS= option, all the imputed values of *imputed-variable* are adjusted.

You can use the following *adjust-options* to explicitly specify adjustment parameters:

SCALE=*c*
specifies a scale parameter for adjusting imputed values of a continuous *imputed-variable*. The value of *c* must be positive. By default, $c = 1$ (no scale adjustment is made). The SCALE= option does not apply to adjusting imputed values of classification variables.

SHIFT | DELTA=δ
specifies the shift parameter for imputed values of *imputed-variable*. By default, $\delta = 0$ (no shift adjustment is made).

SIGMA=σ
specifies the sigma parameter for imputed values of *imputed-variable*, where $\sigma \geq 0$. For a specified $\sigma > 0$, a simulated shift parameter is generated from the normal distribution, with mean δ and standard deviation σ in each imputation. By default, $\sigma = 0$, which means that the same shift adjustment δ is made for imputed values of *imputed-variable*.

You can use the following *adjust-option* to adjust imputed values by using parameters that are stored in a data set:

PARMS(*parms-options* **)=***SAS-data-set*
names the SAS data set that contains the adjustment parameters at each imputation for imputed values of *imputed-variable*. You can specify the following *parms-options*:

SHIFT | DELTA=*variable*
identifies the *variable* for the shift parameter.

SCALE=*variable*
identifies the *variable* for the scale parameter of a continuous *imputed-variable*.

When the PARMS= data set does not contain a variable named _IMPUTATION_, the same adjustment parameters are used for each imputation. When the PARMS= data set contains a variable named _IMPUTATION_, whose values are 1, 2, ..., n, where n is the number of imputations, the adjustment parameters are used for the corresponding imputations.

For a classification *imputed-variable* whose values are imputed by using an ordinal logistic regression method, you cannot specify the SHIFT= and SIGMA= parameters for more than one EVENT= level if the imputed variable has more than two response levels. For a detailed description of imputed value adjustments, see the section "Adjusting Imputed Values in Pattern-Mixture Models" on page 6624.

MONOTONE Statement

MONOTONE < *method* < **(**< *imputed* < **=** *effects* > > < / *options* >**)** > >
< . . . *method* < **(**< *imputed* < **=** *effects* > > < / *options* >**)** > > **;**

The MONOTONE statement specifies imputation methods for data sets with monotone missingness. You must also specify a VAR statement, and the data set must have a monotone missing pattern with variables ordered in the VAR list.

Table 82.4 summarizes the options available for the MONOTONE statement.

Table 82.4 Summary of Imputation Methods in MONOTONE Statement

Option	Description
DISCRIM	Specifies the discriminant function method
LOGISTIC	Specifies the logistic regression method
PROPENSITY	Specifies the propensity scores method
REG	Specifies the regression method
REGPMM	Specifies the predictive mean matching method

For each method, you can specify the imputed variables and, optionally, a set of the effects to impute these variables. Each effect is a variable or a combination of variables preceding the imputed variable in the VAR statement. The syntax for specification of effects is the same as for the GLM procedure. See Chapter 53, "The GLM Procedure," for more information.

One general form of an effect involving several variables is

*X1 * X2 * A * B * C (D E)*

where A, B, C, D, and E are classification variables and X1 and X2 are continuous variables.

When a MONOTONE statement is used without specifying any methods, the regression method is used for all imputed continuous variables and the discriminant function method is used for all imputed classification variables. In this case, for each imputed continuous variable, all preceding variables in the VAR statement are used as the covariates, and for each imputed classification variable, all preceding continuous variables in the VAR statement are used as the covariates.

When a method for continuous variables is specified without imputed variables, the method is used for all continuous variables in the VAR statement that are not specified in other methods. Similarly, when a method for classification variables is specified without imputed variables, the method is used for all classification variables in the VAR statement that are not specified in other methods.

For each imputed variable that does not use the discriminant function method, if no covariates are specified, then all preceding variables in the VAR statement are used as the covariates. That is, each preceding continuous variable is used as a regressor effect, and each preceding classification variable is used as a main effect. For an imputed variable that uses the discriminant function method, if no covariates are specified, then all preceding variables in the VAR statement are used as the covariates with the CLASSEFFECTS=INCLUDE option, and all preceding continuous variables in the VAR statement are used as the covariates with the CLASSEFFECTS=EXCLUDE option (which is the default).

With a MONOTONE statement, the variables are imputed sequentially in the order given by the VAR statement. For a continuous variable, you can use a regression method, a regression predicted mean matching method, or a propensity score method to impute missing values. For a nominal classification variable, you can use either a discriminant function method or a logistic regression method (generalized logit model) to impute missing values without using the ordering of the class levels. For an ordinal classification variable, you can use a logistic regression method (cumulative logit model) to impute missing values by using the ordering of the class levels. For a binary classification variable, either a discriminant function method or a logistic regression method can be used.

Note that except for the regression method, all other methods impute values from the observed observation values. You can specify the following methods in a MONOTONE statement.

DISCRIM <(*imputed* < = *effects*> <| *options*>) >
specifies the discriminant function method of classification variables. The available options are as follows:

CLASSEFFECTS=EXCLUDE | INCLUDE
specifies whether the CLASS variables are used as covariate effects. CLASSEFFECTS=EXCLUDE excludes the CLASS variables from covariate effects, and CLASSEFFECTS=INCLUDE includes the CLASS variables as covariate effects. By default, CLASSEFFECTS=EXCLUDE.

DETAILS
displays the group means and pooled covariance matrix used in each imputation.

PCOV=FIXED | POSTERIOR
specifies the pooled covariance used in the discriminant method. PCOV=FIXED uses the observed-data pooled covariance matrix for each imputation, and PCOV=POSTERIOR draws a pooled covariance matrix from its posterior distribution. By default, PCOV=POSTERIOR.

PRIOR=EQUAL | JEFFREYS <=*c*> **| PROPORTIONAL | RIDGE** <=*d*>
specifies the prior probabilities of group membership. PRIOR=EQUAL sets the prior probabilities equal for all groups; PRIOR=JEFFREYS <=*c*> specifies a noninformative prior, $0 < c < 1$; PRIOR=PROPORTIONAL sets the prior probabilities proportion to the group sample sizes; and PRIOR=RIDGE <=*d*> specifies a ridge prior, $d > 0$. If the noninformative prior c is not specified, c=0.5 is used. If the ridge prior d is not specified, d=0.25 is used. By default, PRIOR=JEFFREYS.

See the section "Monotone and FCS Discriminant Function Methods" on page 6592 for a detailed description of the method.

LOGISTIC <(*imputed* < = *effects*> <| *options*>) >
specifies the logistic regression method for classification variables. The available options are as follows:

DESCENDING
reverses the sort order for the levels of the response variables.

DETAILS
displays the regression coefficients in the logistic regression model used in each imputation.

LIKELIHOOD=NOAUGMENT
LIKELIHOOD=AUGMENT <(**WEIGHT=***w* **| NPARM** <(**MULT=***m*)>)>
specifies whether to add new observations to the likelihood function in the computation of maximum likelihood estimates. LIKELIHOOD=AUGMENT adds observations in each response group to the likelihood function, and LIKELIHOOD=NOAUGMENT makes no adjustment to the likelihood function. By default, LIKELIHOOD=NOAUGMENT.

The LIKELIHOOD=AUGMENT option is useful when the maximum likelihood parameter estimates do not exist. When LIKELIHOOD=AUGMENT, each added observation contributes the same weight, and the WEIGHT= option specifies the total added weight:

WEIGHT=*w*
explicitly specifies the total added weight *w*.

WEIGHT=NPARM < (MULT=*m*) >
uses the number of parameters in the logistic regression model as the total added weight. For example, for a simple binary logistic regression model that consists only of p continuous effects, the added weight is p+1. The MULT=*m* option specifies the multiplier for the total added weight, $0 < m \leq 1$, and the resulting total added weight is *m* times the number of parameters in the model. By default, MULT=1.

By default, WEIGHT=NPARM. You can specify either the MULT=*m* suboption in WEIGHT=NPARM or the WEIGHT=*w* option to use a different total added weight in the computation of maximum likelihood estimates. For example, if the ratio between the number of parameters and the number of available observations (before augmentation) is large, you can use either MULT=*m* or WEIGHT=*w* to reduce the weight for the added observations (that is, reduce the effect from the added observations in the computation of maximum likelihood estimates). For more information about the augmented data approach, see the section "Logistic Regression with Augmented Data" on page 6597.

LINK=GLOGIT | LOGIT
specifies the link function linking the response probabilities to the linear predictors. LINK=LOGIT uses the log odds function to fit the binary logit model when there are two response categories and to fit the cumulative logit model when there are more than two response categories. LINK=GLOGIT uses the generalized logit function to fit the generalized logit model where each nonreference category is contrasted with the last category. By default, LINK=LOGIT.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the response variable. ORDER=DATA sorts by the order of appearance in the input data set; ORDER=FORMATTED sorts by their external formatted values; ORDER=FREQ sorts by the descending frequency counts; and ORDER=INTERNAL sorts by the unformatted values. By default, ORDER=FORMATTED.

See the section "Monotone and FCS Logistic Regression Methods" on page 6594 for a detailed description of the method.

PROPENSITY < (*imputed* < = *effects* > < / *options* >) >
specifies the propensity scores method of variables. Each variable is either a classification variable or a continuous variable. The available *options* are DETAILS and NGROUPS=. The DETAILS option displays the regression coefficients in the logistic regression model for propensity scores. The NGROUPS= option specifies the number of groups created based on propensity scores. By default, NGROUPS=5.

See the section "Monotone Propensity Score Method" on page 6597 for a detailed description of the method.

REG | REGRESSION < (*imputed* < = *effects* > < / DETAILS >) >
specifies the regression method of continuous variables. The DETAILS option displays the regression coefficients in the regression model used in each imputation.

With a regression method, you can use the MAXIMUM=, MINIMUM=, and ROUND= options to make the imputed values more consistent with the observed variable values.

See the section "Monotone and FCS Regression Methods" on page 6590 for a detailed description of the method.

REGPMM < (*imputed* < = *effects* > < / *options* >) >

REGPREDMEANMATCH < (*imputed* < = *effects* > < / *options* >) >

specifies the predictive mean matching method for continuous variables. This method is similar to the regression method except that it imputes a value randomly from a set of observed values whose predicted values are closest to the predicted value for the missing value from the simulated regression model (Heitjan and Little 1991; Schenker and Taylor 1996).

The available options are DETAILS and K=. The DETAILS option displays the regression coefficients in the regression model used in each imputation. The K= option specifies the number of closest observations to be used in the selection. By default, K=5.

See the section "Monotone and FCS Predictive Mean Matching Methods" on page 6591 for a detailed description of the method.

With a MONOTONE statement, the variables with missing values are imputed sequentially in the order specified in the VAR statement. For example, the following MI procedure statements use the default regression method for continuous variables to impute variable y2 from the effect y1, the logistic regression method to impute variable c1 from effects y1, y2, and y1 ∗ y2, and the regression method to impute variable y3 from effects y1, y2, and c1:

```
proc mi;
   class c1;
   var y1 y2 c1 y3;
   monotone logistic(c1= y1 y2 y1*y2);
   monotone reg(y3= y1 y2 c1);
run;
```

The variable y1 is not imputed since it is the leading variable in the VAR statement.

TRANSFORM Statement

TRANSFORM *transform* (*variables*< / *options* >)< . . . *transform* (*variables*< / *options* >) > ;

The TRANSFORM statement lists the transformations and their associated variables to be transformed. The options are transformation options that provide additional information for the transformation.

The MI procedure assumes that the data are from a multivariate normal distribution when either the regression method or the MCMC method is used. When some variables in a data set are clearly non-normal, it is useful to transform these variables to conform to the multivariate normality assumption. With a TRANSFORM statement, variables are transformed before the imputation process, and these transformed variable values are displayed in all of the results. When you specify an OUT= option, the variable values are back-transformed to create the imputed data set.

The following transformations can be used in the TRANSFORM statement:

BOXCOX

specifies the Box-Cox transformation of variables. The variable Y is transformed to $\frac{(Y+c)^{\lambda}-1}{\lambda}$, where c is a constant such that each value of $Y + c$ must be positive. If the specified constant $\lambda = 0$, the logarithmic transformation is used.

EXP

specifies the exponential transformation of variables. The variable Y is transformed to $e^{(Y+c)}$, where c is a constant.

LOG

specifies the logarithmic transformation of variables. The variable Y is transformed to $\log(Y + c)$, where c is a constant such that each value of $Y + c$ must be positive.

LOGIT

specifies the logit transformation of variables. The variable Y is transformed to $\log(\frac{Y/c}{1-Y/c})$, where the constant c>0 and the values of Y/c must be between 0 and 1.

POWER

specifies the power transformation of variables. The variable Y is transformed to $(Y + c)^{\lambda}$, where c is a constant such that each value of $Y + c$ must be positive and the constant $\lambda \neq 0$.

The following options provide the constant c and λ values in the transformations.

C=*number*

specifies the c value in the transformation. The default is $c = 1$ for logit transformation and $c = 0$ for other transformations.

LAMBDA=*number*

specifies the λ value in the power and Box-Cox transformations. You must specify the λ value for these two transformations.

For example, the following statement requests that variables $\log(y1)$, a logarithmic transformation for the variable y1, and $\sqrt{y2 + 1}$, a power transformation for the variable y2, be used in the imputation:

```
transform log(y1) power(y2/c=1 lambda=.5);
```

If the MU0= option is used to specify a parameter value $\boldsymbol{\mu}_0$ for a transformed variable, the same transformation for the variable is also applied to its corresponding MU0= value in the t test. Otherwise, $\boldsymbol{\mu}_0 = 0$ is used for the transformed variable. See Example 82.10 for a usage of the TRANSFORM statement.

VAR Statement

VAR *variables* **;**

The VAR statement lists the variables to be analyzed. The variables can be either character or numeric. If you omit the VAR statement, all continuous variables not mentioned in other statements are used. The VAR statement is required if you specify either an FCS statement, a MONOTONE statement, an IMPUTE=MONOTONE option in the MCMC statement, or more than one number in the MU0=, MAXIMUM=, MINIMUM=, or ROUND= option.

The classification variables in the VAR statement, which can be either character or numeric, are further specified in the CLASS statement.

Details: MI Procedure

Descriptive Statistics

Suppose $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)'$ is the $(n \times p)$ matrix of complete data, which might not be fully observed, n_0 is the number of observations fully observed, and n_j is the number of observations with observed values for variable Y_j.

With complete cases, the sample mean vector is

$$\overline{\mathbf{y}} = \frac{1}{n_0} \sum \mathbf{y}_i$$

and the CSSCP matrix is

$$\sum (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})'$$

where each summation is over the fully observed observations.

The sample covariance matrix is

$$\mathbf{S} = \frac{1}{n_0 - 1} \sum (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})'$$

and is an unbiased estimate of the covariance matrix.

The correlation matrix $\mathbf{R}$, which contains the Pearson product-moment correlations of the variables, is derived by scaling the corresponding covariance matrix:

$$\mathbf{R} = \mathbf{D}^{-1} \mathbf{S} \, \mathbf{D}^{-1}$$

where $\mathbf{D}$ is a diagonal matrix whose diagonal elements are the square roots of the diagonal elements of $\mathbf{S}$.

With available cases, the corrected sum of squares for variable Y_j is

$$\sum (y_{ji} - \overline{y}_j)^2$$

where $\overline{y}_j = \frac{1}{n_j} \sum y_{ji}$ is the sample mean and each summation is over observations with observed values for variable Y_j.

The variance is

$$s_{jj}^2 = \frac{1}{n_j - 1} \sum (y_{ji} - \overline{y}_j)^2$$

The correlations for available cases contain pairwise correlations for each pair of variables. Each correlation is computed from all observations that have nonmissing values for the corresponding pair of variables.

EM Algorithm for Data with Missing Values

The EM algorithm (Dempster, Laird, and Rubin 1977) is a technique that finds maximum likelihood estimates in parametric models for incomplete data. For a detailed description and applications of the EM algorithm, see the books by Little and Rubin (2002); Schafer (1997); McLachlan and Krishnan (1997).

The EM algorithm is an iterative procedure that finds the MLE of the parameter vector by repeating the following steps:

1. The expectation E-step
Given a set of parameter estimates, such as a mean vector and covariance matrix for a multivariate normal distribution, the E-step calculates the conditional expectation of the complete-data log likelihood given the observed data and the parameter estimates.

2. The maximization M-step
Given a complete-data log likelihood, the M-step finds the parameter estimates to maximize the complete-data log likelihood from the E-step.

The two steps are iterated until the iterations converge.

In the EM process, the observed-data log likelihood is nondecreasing at each iteration. For multivariate normal data, suppose there are G groups with distinct missing patterns. Then the observed-data log likelihood being maximized can be expressed as

$$\log L(\theta | Y_{obs}) = \sum_{g=1}^{G} \log L_g(\boldsymbol{\theta} | Y_{obs})$$

where $\log L_g(\boldsymbol{\theta} | Y_{obs})$ is the observed-data log likelihood from the gth group, and

$$\log L_g(\boldsymbol{\theta} | Y_{obs}) = -\frac{n_g}{2} \log |\Sigma_g| - \frac{1}{2} \sum_{ig} (\mathbf{y}_{ig} - \boldsymbol{\mu}_g)' \boldsymbol{\Sigma}_g{}^{-1} (\mathbf{y}_{ig} - \boldsymbol{\mu}_g)$$

where n_g is the number of observations in the gth group, the summation is over observations in the gth group, $\mathbf{y}_{ig}$ is a vector of observed values corresponding to observed variables, $\boldsymbol{\mu}_g$ is the corresponding mean vector, and $\boldsymbol{\Sigma}_g$ is the associated covariance matrix.

A sample covariance matrix is computed at each step of the EM algorithm. If the covariance matrix is singular, the linearly dependent variables for the observed data are excluded from the likelihood function. That is, for each observation with linear dependency among its observed variables, the dependent variables

are excluded from the likelihood function. Note that this can result in an unexpected change in the likelihood between iterations prior to the final convergence.

See Schafer (1997, pp. 163–181) for a detailed description of the EM algorithm for multivariate normal data.

By default, PROC MI uses the means and standard deviations from available cases as the initial estimates for the EM algorithm. The correlations are set to zero. These estimates provide a good starting value with positive definite covariance matrix. For a discussion of suggested starting values for the algorithm, see Schafer (1997, p. 169).

You can specify the convergence criterion with the CONVERGE= option in the EM statement. The iterations are considered to have converged when the maximum change in the parameter estimates between iteration steps is less than the value specified. You can also specify the maximum number of iterations used in the EM algorithm with the MAXITER= option.

The MI procedure displays tables of the initial parameter estimates used to begin the EM process and the MLE parameter estimates derived from EM. You can also display the EM iteration history with the ITPRINT option. PROC MI lists the iteration number, the likelihood –2 log L, and the parameter values $\boldsymbol{\mu}$ at each iteration. You can also save the MLE derived from the EM algorithm in a SAS data set by specifying the OUTEM= option.

Statistical Assumptions for Multiple Imputation

The MI procedure assumes that the data are from a multivariate distribution and contain missing values that can occur for any of the variables. It also assumes that the data are from a multivariate normal distribution when either the regression method or the MCMC method is used.

Suppose $\mathbf{Y}$ is the $n \times p$ matrix of complete data, which is not fully observed, and denote the observed part of $\mathbf{Y}$ by $\mathbf{Y}_{obs}$ and the missing part by $\mathbf{Y}_{mis}$. The MI and MIANALYZE procedures assume that the missing data are missing at random (MAR); that is, the probability that an observation is missing can depend on $\mathbf{Y}_{obs}$, but not on $\mathbf{Y}_{mis}$ (Rubin 1976, 1987, p. 53).

To be more precise, suppose that $\mathbf{R}$ is the $n \times p$ matrix of response indicators whose elements are zero or one depending on whether the corresponding elements of Y are missing or observed. Then the MAR assumption is that the distribution of $\mathbf{R}$ can depend on Y_{obs} but not on Y_{mis}:

$$\mathrm{pr}(\mathbf{R}|Y_{obs}, Y_{mis}) = \mathrm{pr}(\mathbf{R}|Y_{obs})$$

For example, consider a trivariate data set with variables Y_1 and Y_2 fully observed, and a variable Y_3 that has missing values. MAR assumes that the probability that Y_3 is missing for an individual can be related to the individual's values of variables Y_1 and Y_2, but not to its value of Y_3. On the other hand, if a complete case and an incomplete case for Y_3 with exactly the same values for variables Y_1 and Y_2 have systematically different values, then there exists a response bias for Y_3, and MAR is violated.

The MAR assumption is not the same as missing completely at random (MCAR), which is a special case of MAR. Under the MCAR assumption, the missing data values are a simple random sample of all data values; the missingness does not depend on the values of any variables in the data set.

Although the MAR assumption cannot be verified with the data and it can be questionable in some situations, the assumption becomes more plausible as more variables are included in the imputation model (Schafer 1997, pp. 27–28; Van Buuren, Boshuizen, and Knook 1999, p. 687).

Furthermore, the MI and MIANALYZE procedures assume that the parameters $\boldsymbol{\theta}$ of the data model and the parameters $\boldsymbol{\phi}$ of the model for the missing-data indicators are distinct. That is, knowing the values of $\boldsymbol{\theta}$ does not provide any additional information about $\boldsymbol{\phi}$, and vice versa. If both the MAR and distinctness assumptions are satisfied, the missing-data mechanism is said to be ignorable Rubin 1987, pp. 50–54; Schafer 1997, pp. 10–11).

Missing Data Patterns

The MI procedure sorts the data into groups based on whether the analysis variables are observed or missing. Note that the input data set does not need to be sorted in any order.

For example, with variables Y_1, Y_2, and Y_3 (in that order) in a data set, up to eight groups of observations can be formed from the data set. Figure 82.6 displays the eight groups of observations and an unique missing pattern for each group.

Figure 82.6 Missing Data Patterns

Missing Data Patterns

Group	Y1	Y2	Y3
1	X	X	X
2	X	X	.
3	X	.	X
4	X	.	.
5	.	X	X
6	.	X	.
7	.	.	X
8	.	.	.

An “X” in Figure 82.6 means that the variable is observed in the corresponding group and a “.” means that the variable is missing. The MI procedure denotes variables that have missing values by “.” or “O”. The value “.” means that the variable is missing and will be imputed, and the value “O” means that the variable is missing and will not be imputed.

The variable order is used to derive the order of the groups from the data set, and thus determines the order of missing values in the data to be imputed. If you specify a different order of variables in the VAR statement, then the results are different even if the other specifications remain the same.

A data set with variables $Y_1, Y_2, \ldots, Y_p$ (in that order) is said to have a *monotone missing pattern* when the event that a variable Y_j is missing for a particular individual implies that all subsequent variables $Y_k, k > j$, are missing for that individual. Alternatively, when a variable Y_j is observed for a particular individual, it is assumed that all previous variables $Y_k, k < j$, are also observed for that individual.

For example, Figure 82.7 displays a data set of three variables with a monotone missing pattern.

Figure 82.7 Monotone Missing Patterns

Monotone Missing Data Patterns

Group	Y1	Y2	Y3
1	X	X	X
2	X	X	.
3	X	.	.

Figure 82.8 displays a data set of three variables with a non-monotone missing pattern.

Figure 82.8 Non-monotone Missing Patterns

Non-monotone Missing Data Patterns

Group	Y1	Y2	Y3
1	X	X	X
2	X	.	X
3	.	X	.
4	.	.	X

A data set with an *arbitrary missing pattern* is a data set with either a monotone missing pattern or a non-monotone missing pattern.

Imputation Methods

This section describes the methods for multiple imputation that are available in the MI procedure. The method of choice depends on the pattern of missingness in the data and the type of the imputed variable, as summarized in Table 82.5.

Table 82.5 Imputation Methods in PROC MI

Pattern of Missingness	Type of Imputed Variable	Type of Covariates	Available Methods
Monotone	Continuous	Arbitrary	• Monotone regression • Monotone predicted mean matching • Monotone propensity score
Monotone	Classification (ordinal)	Arbitrary	• Monotone logistic regression
Monotone	Classification (nominal)	Arbitrary	• Monotone discriminant function • Monotone logistic regression
Arbitrary	Continuous	Continuous	• MCMC full-data imputation • MCMC monotone-data imputation
Arbitrary	Continuous	Arbitrary	• FCS regression • FCS predicted mean matching
Arbitrary	Classification (ordinal)	Arbitrary	• FCS logistic regression

Table 82.5 *continued*

Pattern of Missingness	Type of Imputed Variable	Type of Covariates	Available Methods
Arbitrary	Classification (nominal)	Arbitrary	• FCS discriminant function • FCS logistic regression

To impute missing values for a continuous variable in data sets with monotone missing patterns, you should use either a parametric method that assumes multivariate normality or a nonparametric method that uses propensity scores Rubin 1987, pp. 124, 158; Lavori, Dawson, and Shera 1995). Parametric methods available include the regression method (Rubin 1987, pp. 166–167) and the predictive mean matching method (Heitjan and Little 1991; Schenker and Taylor 1996).

To impute missing values for a classification variable in data sets with monotone missing patterns, you should use the logistic regression method or the discriminant function method. Use the logistic regression method when the classification variable has a binary, nominal, or ordinal response, and use the discriminant function method when the classification variable has a binary or nominal response.

For data sets with arbitrary missing patterns, you can use either of the following methods to impute missing values: a Markov chain Monte Carlo (MCMC) method (Schafer 1997) that assumes multivariate normality, or a fully conditional specification (FCS) method (Van Buuren 2007; Brand 1999) that assumes the existence of a joint distribution for all variables.

For continuous variables in data sets with arbitrary missing patterns, you can use the MCMC method to impute either all the missing values or just enough missing values to make the imputed data sets have monotone missing patterns. With a monotone missing data pattern, you have greater flexibility in your choice of imputation models. In addition to the MCMC method, you can implement other methods, such as the regression method, that do not use Markov chains. You can also specify a different set of covariates for each imputed variable.

Although the regression and MCMC methods assume multivariate normality, inferences based on multiple imputation can be robust to departures from multivariate normality if the amount of missing information is not large, because the imputation model is effectively applied not to the entire data set but only to its missing part (Schafer 1997, pp.147–148).

To impute missing values for both continuous and classification variables in data sets with arbitrary missing patterns, you can use FCS methods to impute missing values for all variables assuming a joint distribution for these variables exists (Brand 1999; Van Buuren 2007). Similar to the methods of imputing missing values for variables in data sets with monotone missing patterns, you can use the regression and predictive mean matching methods to impute missing values for a continuous variable, the logistic regression method to impute missing values for a classification variable when the variable has a binary, nominal, or ordinal response, and the discriminant function method to impute missing values for a classification variable when the variable has a binary or nominal response.

You can also use a TRANSFORM statement to transform variables to conform to the multivariate normality assumption. Variables are transformed before the imputation process and then are reverse-transformed to create the imputed data set.

Li (1988) presents a theoretical argument for convergence of the MCMC method in the continuous case and uses it to create imputations for incomplete multivariate continuous data. In practice, however, it is not

easy to check the convergence of a Markov chain, especially for a large number of parameters. PROC MI generates statistics and plots that you can use to check for convergence of the MCMC method. The details are described in the section "Checking Convergence in MCMC" on page 6609.

Note that all continuous variables are standardized before the imputation process and then are transformed back to the original scale after the imputation process.

Monotone Methods for Data Sets with Monotone Missing Patterns

For data sets with monotone missing data patterns, you can use monotone methods to impute missing values for the variables. A monotone method creates multiple imputations by imputing missing values sequentially over the variables taken one at a time.

For example, with variables Y_1, Y_2, ..., Y_p (in that order) in the VAR statement, a monotone method sequentially simulates a draw for missing values for variables $Y_2, \ldots, Y_p$. That is, the missing values are imputed by using the sequence

$$
\begin{aligned}
\boldsymbol{\theta}_2^{(*)} &\sim P(\boldsymbol{\theta}_2 \mid Y_{1(obs)}, Y_{2(obs)}) \\
Y_2^{(*)} &\sim P(Y_2 \mid \boldsymbol{\theta}_2^{(*)}) \\
&\ldots \\
&\ldots \\
\boldsymbol{\theta}_p^{(*)} &\sim P(\boldsymbol{\theta}_p \mid Y_{1(obs)}, \ldots, Y_{p(obs)}) \\
Y_p^{(*)} &\sim P(Y_p \mid \boldsymbol{\theta}_p^{(*)})
\end{aligned}
$$

where $Y_{j(obs)}$ is the set of observed Y_j values, $\boldsymbol{\theta}_j^{(*)}$ is the set of simulated parameters for the conditional distribution of Y_j given covariates constructed from variables Y_1, Y_2, ..., Y_{j-1}, and $Y_j^{(*)}$ is the set of imputed Y_j values.

The missing values for the leading variable Y_1 are not imputed, and missing values for Y_2, ..., Y_p are not imputed for those observations with missing Y_1 values. For each subsequent variable Y_j with missing values, the corresponding imputation method is used to fit a model with covariates constructed from its preceding variables $Y_1, Y_2, \ldots, Y_{j-1}$. The observed observations for Y_j, which include only observations with observed values for $Y_1, Y_2, \ldots, Y_{j-1}$, are used in the model fitting. With this resulting model, a new model is drawn and then used to impute missing values for Y_j.

You can specify a separate monotone method for each imputed variable. If a method is not specified for the variable, then the default method is used. That is, a regression method is used for a continuous variable and a discriminant function method is used for a classification variable. For each imputed variable, you can also specify a set of covariates that are constructed from its preceding variables. If a set of covariates is not specified for the variable, all preceding variables in the VAR list are used as covariates.

You can use a regression method, a predictive mean matching method, or a propensity score method to impute missing values for a continuous variable; a logistic regression method for a classification variable with a binary or ordinal response; and a discriminant function method for a classification variable with a binary or nominal response. See the sections "Monotone and FCS Regression Methods" on page 6590, "Monotone

and FCS Predictive Mean Matching Methods" on page 6591, "Monotone Propensity Score Method" on page 6597, "Monotone and FCS Discriminant Function Methods" on page 6592, and "Monotone and FCS Logistic Regression Methods" on page 6594 for these methods.

Monotone and FCS Regression Methods

The regression method is the default imputation method in the MONOTONE and FCS statements for continuous variables.

In the regression method, a regression model is fitted for a continuous variable with the covariates constructed from a set of effects. Based on the fitted regression model, a new regression model is simulated from the posterior predictive distribution of the parameters and is used to impute the missing values for each variable (Rubin 1987, pp. 166–167). That is, for a continuous variable Y_j with missing values, a model

$$Y_j = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k$$

is fitted using observations with observed values for the variable Y_j and its covariates $X_1, X_2, \ldots, X_k$.

The fitted model includes the regression parameter estimates $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k)$ and the associated covariance matrix $\hat{\sigma}_j^2 \mathbf{V}_j$, where $\mathbf{V}_j$ is the usual $\mathbf{X}'\mathbf{X}$ inverse matrix derived from the intercept and covariates $X_1, X_2, \ldots, X_k$.

The following steps are used to generate imputed values for each imputation:

1. New parameters $\boldsymbol{\beta}_* = (\beta_{*0}, \beta_{*1}, \ldots, \beta_{*(k)})$ and σ_{*j}^2 are drawn from the posterior predictive distribution of the parameters. That is, they are simulated from $(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k)$, σ_j^2, and $\mathbf{V}_j$. The variance is drawn as

   $$\sigma_{*j}^2 = \hat{\sigma}_j^2 (n_j - k - 1)/g$$

   where g is a $\chi^2_{n_j - k - 1}$ random variate and n_j is the number of nonmissing observations for Y_j. The regression coefficients are drawn as

   $$\boldsymbol{\beta}_* = \hat{\boldsymbol{\beta}} + \sigma_{*j} \mathbf{V}'_{hj} \mathbf{Z}$$

   where $\mathbf{V}_{hj}$ is the upper triangular matrix in the Cholesky decomposition, $\mathbf{V}_j = \mathbf{V}'_{hj} \mathbf{V}_{hj}$, and $\mathbf{Z}$ is a vector of $k + 1$ independent random normal variates.

2. The missing values are then replaced by

   $$\beta_{*0} + \beta_{*1} x_1 + \beta_{*2} x_2 + \ldots + \beta_{*(k)} x_k + z_i \sigma_{*j}$$

   where $x_1, x_2, \ldots, x_k$ are the values of the covariates and z_i is a simulated normal deviate.

Monotone and FCS Predictive Mean Matching Methods

The predictive mean matching method is also an imputation method available for continuous variables. It is similar to the regression method except that for each missing value, it imputes a value randomly from a set of observed values whose predicted values are closest to the predicted value for the missing value from the simulated regression model (Heitjan and Little 1991; Schenker and Taylor 1996).

Following the description of the model in the section "Monotone and FCS Regression Methods" on page 6590, the following steps are used to generate imputed values:

1. New parameters $\boldsymbol{\beta}_* = (\beta_{*0}, \beta_{*1}, \ldots, \beta_{*(k)})$ and σ^2_{*j} are drawn from the posterior predictive distribution of the parameters. That is, they are simulated from $(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k)$, σ^2_j, and $\mathbf{V}_j$. The variance is drawn as

   $$\sigma^2_{*j} = \hat{\sigma}^2_j (n_j - k - 1)/g$$

   where g is a $\chi^2_{n_j - k - 1}$ random variate and n_j is the number of nonmissing observations for Y_j. The regression coefficients are drawn as

   $$\boldsymbol{\beta}_* = \hat{\boldsymbol{\beta}} + \sigma_{*j} \mathbf{V}'_{hj} \mathbf{Z}$$

   where $\mathbf{V}_{hj}$ is the upper triangular matrix in the Cholesky decomposition, $\mathbf{V}_j = \mathbf{V}'_{hj} \mathbf{V}_{hj}$, and $\mathbf{Z}$ is a vector of $k + 1$ independent random normal variates.

2. For each missing value, a predicted value

   $$y_{i*} = \beta_{*0} + \beta_{*1} x_1 + \beta_{*2} x_2 + \ldots + \beta_{*(k)} x_k$$

   is computed with the covariate values $x_1, x_2, \ldots, x_k$.

3. A set of k_0 observations whose corresponding predicted values are closest to y_{i*} is generated. You can specify k_0 with the K= option.

4. The missing value is then replaced by a value drawn randomly from these k_0 observed values.

The predictive mean matching method requires the number of closest observations to be specified. A smaller k_0 tends to increase the correlation among the multiple imputations for the missing observation and results in a higher variability of point estimators in repeated sampling. On the other hand, a larger k_0 tends to lessen the effect from the imputation model and results in biased estimators (Schenker and Taylor 1996, p. 430).

The predictive mean matching method ensures that imputed values are plausible; it might be more appropriate than the regression method if the normality assumption is violated (Horton and Lipsitz 2001, p. 246).

Monotone and FCS Discriminant Function Methods

The discriminant function method is the default imputation method in the MONOTONE and FCS statements for classification variables.

For a nominal classification variable Y_j with responses $1, \ldots, g$ and a set of effects from its preceding variables, if the covariates $X_1, X_2, \ldots, X_k$ associated with these effects within each group are approximately multivariate normal and the within-group covariance matrices are approximately equal, the discriminant function method (Brand 1999, pp. 95–96) can be used to impute missing values for the variable Y_j.

Denote the group-specific means for covariates $X_1, X_2, \ldots, X_k$ by

$$\overline{\mathbf{X}}_t = (\overline{X}_{t1}, \overline{X}_{t2}, \ldots, \overline{X}_{tk}),\ t = 1, 2, \ldots, g$$

then the pooled covariance matrix is computed as

$$\mathbf{S} = \frac{1}{n-g} \sum_{t=1}^{g} (n_t - 1)\mathbf{S}_t$$

where $\mathbf{S}_t$ is the within-group covariance matrix, n_t is the group-specific sample size, and $n = \sum_{t=1}^{g} n_t$ is the total sample size.

In each imputation, new parameters of the group-specific means ($\mathbf{m}_{*t}$), pooled covariance matrix ($\mathbf{S}_*$), and prior probabilities of group membership (q_{*t}) can be drawn from their corresponding posterior distributions (Schafer 1997, p. 356).

Pooled Covariance Matrix and Group-Specific Means

For each imputation, the MI procedure uses either the fixed observed pooled covariance matrix (PCOV=FIXED) or a drawn pooled covariance matrix (PCOV=POSTERIOR) from its posterior distribution with a noninformative prior. That is,

$$\mathbf{\Sigma}|\mathbf{X} \quad \sim \quad W^{-1}\left(n-g,\ (n-g)\mathbf{S}\right)$$

where W^{-1} is an inverted Wishart distribution.

The group-specific means are then drawn from their posterior distributions with a noninformative prior

$$\boldsymbol{\mu}_t|(\mathbf{\Sigma}, \overline{\mathbf{X}}_t) \quad \sim \quad N\left(\overline{\mathbf{X}}_t,\ \frac{1}{n_t}\mathbf{\Sigma}\right)$$

See the section "Bayesian Estimation of the Mean Vector and Covariance Matrix" on page 6602 for a complete description of the inverted Wishart distribution and posterior distributions that use a noninformative prior.

Prior Probabilities of Group Membership

The prior probabilities are computed through the drawing of new group sample sizes. When the total sample size n is considered fixed, the group sample sizes $(n_1, n_2, \ldots, n_g)$ have a multinomial distribution. New multinomial parameters (group sample sizes) can be drawn from their posterior distribution by using a Dirichlet prior with parameters $(\alpha_1, \alpha_2, \ldots, \alpha_g)$.

After the new sample sizes are drawn from the posterior distribution of $(n_1, n_2, \ldots, n_g)$, the prior probabilities q_{*t} are computed proportionally to the drawn sample sizes.

See Schafer (1997, pp. 247–255) for a complete description of the Dirichlet prior.

Imputation Steps

The discriminant function method uses the following steps in each imputation to impute values for a nominal classification variable Y_j with g responses:

1. Draw a pooled covariance matrix $\mathbf{S}_*$ from its posterior distribution if the PCOV=POSTERIOR option is used.

2. For each group, draw group means $\mathbf{m}_{*t}$ from the observed group mean $\overline{\mathbf{X}}_t$ and either the observed pooled covariance matrix (PCOV=FIXED) or the drawn pooled covariance matrix $\mathbf{S}_*$ (PCOV=POSTERIOR).

3. For each group, compute or draw q_{*t}, prior probabilities of group membership, based on the PRIOR= option:

   - PRIOR=EQUAL, $q_{*t} = 1/g$, prior probabilities of group membership are all equal.
   - PRIOR=PROPORTIONAL, $q_{*t} = n_t/n$, prior probabilities are proportional to their group sample sizes.
   - PRIOR=JEFFREYS=*c*, a noninformative Dirichlet prior with $\alpha_t = c$ is used.
   - PRIOR=RIDGE=*d*, a ridge prior is used with $\alpha_t = d * n_t/n$ for $d \geq 1$ and $\alpha_t = d * n_t$ for $d < 1$.

4. With the group means $\mathbf{m}_{*t}$, the pooled covariance matrix $\mathbf{S}_*$, and the prior probabilities of group membership q_{*t}, the discriminant function method derives linear discriminant function and computes the posterior probabilities of an observation belonging to each group

   $$p_t(\mathbf{x}) = \frac{\exp(-0.5D_t^2(\mathbf{x}))}{\sum_{u=1}^{g} \exp(-0.5D_u^2(\mathbf{x}))}$$

   where $D_t^2(\mathbf{x}) = (\mathbf{x} - \mathbf{m}_{*t})'\mathbf{S}_*^{-1}(\mathbf{x} - \mathbf{m}_{*t}) - 2\log(q_{*t})$ is the generalized squared distance from $\mathbf{x}$ to group t.

5. Draw a random uniform variate u, between 0 and 1, for each observation with missing group value. With the posterior probabilities, $p_1(\mathbf{x}) + p_2(\mathbf{x}) + \ldots, + p_g(\mathbf{x}) = 1$, the discriminant function method imputes $Y_j = 1$ if the value of u is less than $p_1(\mathbf{x})$, $Y_j = 2$ if the value is greater than or equal to $p_1(\mathbf{x})$ but less than $p_1(\mathbf{x}) + p_2(\mathbf{x})$, and so on.

Monotone and FCS Logistic Regression Methods

The logistic regression method is another imputation method available for classification variables. In the logistic regression method, a logistic regression model is fitted for a classification variable with a set of covariates constructed from the effects, where the classification variable is an ordinal response or a nominal response variable.

In the MI procedure, ordered values are assigned to response levels in ascending sorted order. If the response variable Y takes values in $\{1, \ldots, K\}$, then for ordinal response models, the cumulative model has the form

$$\text{logit}(\Pr(Y \le j|\mathbf{x})) = \log\left(\frac{\Pr(Y \le j|\mathbf{x})}{1 - \Pr(Y \le j|\mathbf{x})}\right) = \alpha_j + \boldsymbol{\beta}'\mathbf{x}, \quad j = 1, \ldots, K-1$$

where $\alpha_1, \ldots, \alpha_{K-1}$ are K-1 intercept parameters, and $\boldsymbol{\beta}$ is the vector of slope parameters.

For nominal response logistic models, where the K possible responses have no natural ordering, the generalized logit model has the form

$$\log\left(\frac{\Pr(Y = j \mid \mathbf{x})}{\Pr(Y = K \mid \mathbf{x})}\right) = \alpha_j + \boldsymbol{\beta}_j'\mathbf{x}, \quad j = 1, \ldots, K-1$$

where the $\alpha_1, \ldots, \alpha_{K-1}$ are K-1 intercept parameters, and the $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{K-1}$ are K-1 vectors of slope parameters.

Binary Response Logistic Regression

For a binary classification variable, based on the fitted regression model, a new logistic regression model is simulated from the posterior predictive distribution of the parameters and is used to impute the missing values for each variable (Rubin 1987, pp. 167–170).

For a binary variable Y with responses 1 and 2, a logistic regression model is fitted using observations with observed values for the imputed variable Y:

$$\text{logit}(p_1) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

where $X_1, X_2, \ldots, X_p$ are covariates for Y, $p_1 = \Pr(Y = 1|X_1, X_2, \ldots, X_p)$, and $\text{logit}(p_1) = \log(p_1/(1-p_1))$

The fitted model includes the regression parameter estimates $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_p)$ and the associated covariance matrix $\mathbf{V}$.

The following steps are used to generate imputed values for a binary variable Y with responses 1 and 2:

1. New parameters $\boldsymbol{\beta}_* = (\beta_{*0}, \beta_{*1}, \ldots, \beta_{*(p)})$ are drawn from the posterior predictive distribution of the parameters.

   $$\boldsymbol{\beta}_* = \hat{\boldsymbol{\beta}} + \mathbf{V}_h'\mathbf{Z}$$

   where $\mathbf{V}_h$ is the upper triangular matrix in the Cholesky decomposition, $\mathbf{V} = \mathbf{V}_h'\mathbf{V}_h$, and $\mathbf{Z}$ is a vector of $p + 1$ independent random normal variates.

2. For an observation with missing Y_j and covariates $x_1, x_2, \ldots, x_p$, compute the predicted probability that Y= 1:

$$p_1 = \frac{\exp(\mu_1)}{1 + \exp(\mu_1)}$$

where $\mu_1 = \beta_{*0} + \beta_{*1}\, x_1 + \beta_{*2}\, x_2 + \ldots + \beta_{*(p)}\, x_p$.

3. Draw a random uniform variate, u, between 0 and 1. If the value of u is less than p_1, impute Y= 1; otherwise impute Y= 2.

The binary logistic regression imputation method can be extended to include the ordinal classification variables with more than two levels of responses, and the nominal classification variables. The LINK=LOGIT and LINK=GLOGIT options can be used to specify the cumulative logit model and the generalized logit model, respectively. The options ORDER= and DESCENDING can be used to specify the sort order for the levels of the imputed variables.

Ordinal Response Logistic Regression

For an ordinal classification variable, based on the fitted regression model, a new logistic regression model is simulated from the posterior predictive distribution of the parameters and is used to impute the missing values for each variable.

For a variable Y with ordinal responses 1, 2, . . . , K, a logistic regression model is fitted using observations with observed values for the imputed variable Y:

$$\text{logit}(p_j) = \alpha_j + \beta_1\, X_1 + \beta_2\, X_2 + \ldots + \beta_p\, X_p$$

where $X_1, X_2, \ldots, X_p$ are covariates for Y and $p_j = \Pr(Y \le j | X_1, X_2, \ldots, X_k)$.

The fitted model includes the regression parameter estimates $\hat{\alpha} = (\hat{\alpha}_0, \ldots, \hat{\alpha}_{K-1})$ and $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k)$, and their associated covariance matrix $\mathbf{V}$.

The following steps are used to generate imputed values for an ordinal classification variable Y with responses 1, 2, . . . , K:

1. New parameters γ_* are drawn from the posterior predictive distribution of the parameters.

$$\gamma_* = \hat{\gamma} + \mathbf{V}_h' \mathbf{Z}$$

where $\hat{\gamma} = (\hat{\alpha}, \hat{\boldsymbol{\beta}})$, $\mathbf{V}_h$ is the upper triangular matrix in the Cholesky decomposition, $\mathbf{V} = \mathbf{V}_h' \mathbf{V}_h$, and $\mathbf{Z}$ is a vector of $(p+1)(K-1)$ independent random normal variates.

2. For an observation with missing Y and covariates $x_1, x_2, \ldots, x_k$, compute the predicted cumulative probability for $\mathsf{Y} \le j$:

$$p_j = \text{pr}(\mathsf{Y} \le j) = \frac{e^{\alpha_j + \mathbf{x}'\boldsymbol{\beta}}}{e^{\alpha_j + \mathbf{x}'\boldsymbol{\beta}} + 1}$$

3. Draw a random uniform variate, u, between 0 and 1, then impute

$$Y = \begin{cases} 1 & \text{if } u < p_1 \\ k & \text{if } p_{k-1} \le u < p_k \\ K & \text{if } p_{K-1} \le u \end{cases}$$

Nominal Response Logistic Regression

For a nominal classification variable, based on the fitted regression model, a new logistic regression model is simulated from the posterior predictive distribution of the parameters and is used to impute the missing values for each variable.

For a variable Y with nominal responses 1, 2, . . . , K, a logistic regression model is fitted using observations with observed values for the imputed variable Y:

$$\log\left(\frac{p_j}{p_K}\right) = \alpha_j + \beta_{j1}\, X_1 + \beta_{j2}\, X_2 + \ldots + \beta_{jp}\, X_p$$

where $X_1, X_2, \ldots, X_p$ are covariates for Y and $p_j = \Pr(Y = j | X_1, X_2, \ldots, X_p)$.

The fitted model includes the regression parameter estimates $\hat{\alpha} = (\hat{\alpha}_0, \ldots, \hat{\alpha}_{K-1})$ and $\hat{\boldsymbol{\beta}} = (\hat{\boldsymbol{\beta}}_0, \ldots, \hat{\beta}_{K-1})$, and their associated covariance matrix $\mathbf{V}$, where $\hat{\boldsymbol{\beta}}_j = (\hat{\beta}_{j0}, \hat{\beta}_{j1}, \ldots, \hat{\beta}_{jp})$,

The following steps are used to generate imputed values for a nominal classification variable Y with responses 1, 2, . . . , K:

1. New parameters γ_* are drawn from the posterior predictive distribution of the parameters.

   $$\gamma_* = \hat{\gamma} + \mathbf{V}_h' \mathbf{Z}$$

   where $\hat{\gamma} = (\hat{\alpha}, \hat{\boldsymbol{\beta}})$, $\mathbf{V}_h$ is the upper triangular matrix in the Cholesky decomposition, $\mathbf{V} = \mathbf{V}_h' \mathbf{V}_h$, and $\mathbf{Z}$ is a vector of $p + K - 1$ independent random normal variates.

2. For an observation with missing Y and covariates $x_1, x_2, \ldots, x_k$, compute the predicted probability for Y= j, j=1, 2, . . . , K-1:

   $$\mathrm{pr}(\mathsf{Y} = j) = \frac{e^{\alpha_j + \mathbf{x}'\boldsymbol{\beta}_j}}{\sum_{k=1}^{K-1} e^{\alpha_k + \mathbf{x}'\boldsymbol{\beta}_k} + 1}$$

   and

   $$\mathrm{pr}(\mathsf{Y} = K) = \frac{1}{\sum_{k=1}^{K-1} e^{\alpha_k + \mathbf{x}'\boldsymbol{\beta}_k} + 1}$$

3. Compute the cumulative probability for $\mathsf{Y} \le j$:

   $$P_j = \sum_{k=1}^{j} \mathrm{pr}(\mathsf{Y} = k)$$

4. Draw a random uniform variate, u, between 0 and 1, then impute

   $$Y = \begin{cases} 1 & \text{if } u < p_1 \\ k & \text{if } p_{k-1} \le u < p_k \\ K & \text{if } p_{K-1} \le u \end{cases}$$

Logistic Regression with Augmented Data

In a logistic regression model, you might not be able to find the maximum likelihood estimates of the parameters if there is no overlap of the sample points from response groups—that is, if the data points have either a complete separation pattern or a quasi-complete separation pattern.

Complete separation of data points occurs when a linear combination of predictors correctly allocates all observations to their response groups. Quasi-complete separation occurs when a linear combination of predictors correctly allocates all observations to their response groups except for a subset of observations where the values of linear combinations of predictors are identical. For more information about complete separation patterns and quasi-complete separation patterns, see the section "Existence of Maximum Likelihood Estimates" on page 6032 in Chapter 79, "The LOGISTIC Procedure."

To address the separation issue in multiple imputation, White, Daniel, and Royston (2010) add observations to each response group and then use the augmented data to fit a weighted logistic regression. In each response group, $2p$ observations are added, where p is the number of predictors. More specifically, corresponding to each predictor, two observations are added: the first with the predictor mean minus the predictor standard deviation, and the second with the predictor mean plus the predictor standard deviation. In both observations, the values of other predictors are fixed at their corresponding means. Each additional observation contributes the same weight, and the total added weight is $p+1$. Each available observation in the data set (before augmentation) has a weight of 1. With this approach, there is an overlap of sample points, and maximum likelihood estimates can be obtained.

In the MONOTONE and FCS statements, the LIKELIHOOD=AUGMENT suboption in the LOGISTIC option requests maximum likelihood estimates based on augmented data. When LIKELIHOOD=AUGMENT, you can use the WEIGHT=*w* option to specify the total added weight *w* explicitly, or you can use the WEIGHT=NPARM option to specify the number of parameters as the total added weight. More specifically, for logistic regression models that consist only of p continuous effects, the added weight is $p+1$ for a simple binary logistic model, $p+k-1$ for an ordinal response model, and $(p+1)\ (k-1)$ for a nominal response model, where k is the number of response levels.

If the ratio between the number of parameters and the number of available observations (before augmentation) is large, the effect from the added observations in the computation of maximum likelihood estimates can be significant. You can use the MULT=*m* suboption in the WEIGHT=NPARM option to reduce the total added weight, where the multiplier $0 < m \leq 1$. The resulting total added weight is then *m* times the number of parameters. Alternatively, you can use the WEIGHT=*w* option to specify a smaller total added weight *w* explicitly.

Monotone Propensity Score Method

The propensity score method is another imputation method available for continuous variables when the data set has a monotone missing pattern.

A propensity score is generally defined as the conditional probability of assignment to a particular treatment given a vector of observed covariates (Rosenbaum and Rubin 1983). In the propensity score method, for a variable with missing values, a propensity score is generated for each observation to estimate the probability that the observation is missing. The observations are then grouped based on these propensity scores, and an approximate Bayesian bootstrap imputation (Rubin 1987, p. 124) is applied to each group (Lavori, Dawson, and Shera 1995).

The propensity score method uses the following steps to impute values for variable Y_j with missing values:

1. Creates an indicator variable R_j with the value 0 for observations with missing Y_j and 1 otherwise.
2. Fits a logistic regression model

   $$\text{logit}(p_j) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k$$

   where $X_1, X_2, \ldots, X_k$ are covariates for Y_j, $p_j = \Pr(R_j = 0 | X_1, X_2, \ldots, X_k)$, and $\text{logit}(p) = \log(p/(1-p))$.
3. Creates a propensity score for each observation to estimate the probability that it is missing.
4. Divides the observations into a fixed number of groups (typically assumed to be five) based on these propensity scores.
5. Applies an approximate Bayesian bootstrap imputation to each group. In group k, suppose that Y_{obs} denotes the n_1 observations with nonmissing Y_j values and Y_{mis} denotes the n_0 observations with missing Y_j. The approximate Bayesian bootstrap imputation first draws n_1 observations randomly with replacement from Y_{obs} to create a new data set Y^*_{obs}. This is a nonparametric analog of drawing parameters from the posterior predictive distribution of the parameters. The process then draws the n_0 values for Y_{mis} randomly with replacement from Y^*_{obs}.

Steps 1 through 5 are repeated sequentially for each variable with missing values.

The propensity score method was originally designed for a randomized experiment with repeated measures on the response variables. The goal was to impute the missing values on the response variables. The method uses only the covariate information that is associated with whether the imputed variable values are missing; it does not use correlations among variables. It is effective for inferences about the distributions of individual imputed variables, such as a univariate analysis, but it is not appropriate for analyses that involve relationship among variables, such as a regression analysis (Schafer 1999, p. 11). It can also produce badly biased estimates of regression coefficients when data on predictor variables are missing (Allison 2000).

FCS Methods for Data Sets with Arbitrary Missing Patterns

For a data set with an arbitrary missing data pattern, you can use FCS methods to impute missing values for all variables, assuming the existence of a joint distribution for these variables (Brand 1999; Van Buuren 2007). FCS method involves two phases in each imputation: the preliminary filled-in phase followed by the imputation phase.

At the filled-in phase, the missing values for all variables are filled in sequentially over the variables taken one at a time. The missing values for each variable are filled in using the specified method, or the default method for the variable if a method is not specified, with preceding variables serving as the covariates. These filled-in values provide starting values for these missing values at the imputation phase.

At the imputation phase, the missing values for each variable are imputed using the specified method and covariates at each iteration. The default method for the variable is used if a method is not specified, and the remaining variables are used as covariates if the set of covariates is not specified. After a number of iterations, as specified with the NBITER= option, the imputed values in each variable are used for the imputation. At each iteration, the missing values are imputed sequentially over the variables taken one at a time.

The MI procedure orders the variables as they are ordered in the VAR statement. For example, if the order of the p variables in the VAR statement is $Y_1, Y_2, \ldots, Y_p$, then $Y_1, Y_2, \ldots, Y_p$ (in that order) are used in the filled-in and imputation phases.

The filled-in phase replaces missing values with filled-in values for each variable. That is, with p variables $Y_1, Y_2, \ldots, Y_p$ (in that order), the missing values are filled in by using the sequence,

$$
\begin{aligned}
\boldsymbol{\theta}_1^{(0)} &\sim P(\boldsymbol{\theta}_1 \mid Y_{1(obs)}) \\
Y_{1(*)}^{(0)} &\sim P(Y_1 \mid \boldsymbol{\theta}_1^{(0)}) \\
Y_1^{(0)} &= (Y_{1(obs)}, Y_{1(*)}^{(0)}) \\
&\ldots \\
&\ldots \\
\boldsymbol{\theta}_p^{(0)} &\sim P(\boldsymbol{\theta}_p \mid Y_1^{(0)}, \ldots, Y_{p-1}^{(0)}, Y_{p(obs)}) \\
Y_{p(*)}^{(0)} &\sim P(Y_p \mid \boldsymbol{\theta}_p^{(0)}) \\
Y_p^{(0)} &= (Y_{p(obs)}, Y_{p(*)}^{(0)})
\end{aligned}
$$

where $Y_{j(obs)}$ is the set of observed Y_j values, $Y_{j(*)}^{(0)}$ is the set of filled-in Y_j values, $Y_j^{(0)}$ is the set of both observed and filled-in Y_j values, and $\boldsymbol{\theta}_j^{(0)}$ is the set of simulated parameters for the conditional distribution of Y_j given variables $Y_1, Y_2, \ldots, Y_{j-1}$.

For each variable Y_j with missing values, the corresponding imputation method is used to fit the model with covariates $Y_1, Y_2, \ldots, Y_{j-1}$. The observed observations for Y_j, which might include observations with filled-in values for $Y_1, Y_2, \ldots, Y_{j-1}$, are used in the model fitting. With this resulting model, a new model is drawn and then used to impute missing values for Y_j.

The imputation phase replaces these filled-in values $Y_{j(*)}^{(0)}$ with imputed values for each variable sequentially at each iteration. That is, with p variables $Y_1, Y_2, \ldots, Y_p$ (in that order), the missing values are imputed with the sequence at iteration $t + 1$,

$$
\begin{aligned}
\boldsymbol{\theta}_1^{(t+1)} &\sim P(\boldsymbol{\theta}_1 \mid Y_{1(obs)}, Y_2^{(t)}, \ldots, Y_p^{(t)}) \\
Y_{1(*)}^{(t+1)} &\sim P(Y_1 \mid \boldsymbol{\theta}_1^{(t+1)}) \\
Y_1^{(t+1)} &= (Y_{1(obs)}, Y_{1(*)}^{(t+1)}) \\
&\ldots \\
&\ldots \\
\boldsymbol{\theta}_p^{(t+1)} &\sim P(\boldsymbol{\theta}_p \mid Y_1^{(t+1)}, \ldots, Y_{p-1}^{(t+1)}, Y_{p(obs)}) \\
Y_{p(*)}^{(t+1)} &\sim P(Y_p \mid \boldsymbol{\theta}_p^{(t+1)}) \\
Y_p^{(t+1)} &= (Y_{p(obs)}, Y_{p(*)}^{(t+1)})
\end{aligned}
$$

where $Y_{j(obs)}$ is the set of observed Y_j values, $Y_{j(*)}^{(t+1)}$ is the set of imputed Y_j values at iteration $t+1$, $Y_{j(*)}^{(t)}$ is the set of filled-in Y_j values ($t = 0$) or the set of imputed Y_j values at iteration t ($t > 0$), $Y_j^{(t+1)}$ is the set of both observed and imputed Y_j values at iteration $t+1$, and $\boldsymbol{\theta}_j^{(t+1)}$ is the set of simulated parameters for the conditional distribution of Y_j given covariates constructed from $Y_1, \ldots, Y_{j-1}, Y_{j+1}, \ldots, Y_p$.

At each iteration, a specified model is fitted for each variable with missing values by using observed observations for that variable, which might include observations with imputed values for other variables. With this resulting model, a new model is drawn and then used to impute missing values for the imputed variable.

The steps are iterated long enough for the results to reliably simulate an approximately independent draw of the missing values for an imputed data set.

The imputation methods used in the filled-in and imputation phases are similar to the corresponding monotone methods for monotone missing data. You can use a regression method or a predictive mean matching method to impute missing values for a continuous variable, a logistic regression method for a classification variable with a binary or ordinal response, and a discriminant function method for a classification variable with a binary or nominal response. See the sections "Monotone and FCS Regression Methods" on page 6590, "Monotone and FCS Predictive Mean Matching Methods" on page 6591, "Monotone and FCS Discriminant Function Methods" on page 6592, and "Monotone and FCS Logistic Regression Methods" on page 6594 for these methods.

The FCS method requires fewer iterations than the MCMC method (Van Buuren 2007). Often, as few as five or 10 iterations are enough to produce satisfactory results (Van Buuren 2007; Brand 1999).

Checking Convergence in FCS Methods

The parameters used in the imputation step at each iteration can be saved in an output data set with the OUTITER= option. These include the means and standard deviations. You can then monitor the convergence by displaying trace plots for those parameter values with the PLOTS=TRACE option.

A trace plot for a parameter ξ is a scatter plot of successive parameter estimates ξ_i against the iteration number i. The plot provides a simple way to examine the convergence behavior of the estimation algorithm for ξ. Long-term trends in the plot indicate that successive iterations are highly correlated and that the series of iterations has not converged.

You can display trace plots for the variable means and standard deviations. You can also request logarithmic transformations for positive parameters in the plots with the LOG option. With the LOG option, if a parameter value is less than or equal to zero, then the value is not displayed in the corresponding plot.

See Example 82.8 for a usage of the trace plot.

MCMC Method for Arbitrary Missing Multivariate Normal Data

The Markov chain Monte Carlo (MCMC) method originated in physics as a tool for exploring equilibrium distributions of interacting molecules. In statistical applications, it is used to generate pseudorandom draws from multidimensional and otherwise intractable probability distributions via Markov chains. A Markov chain is a sequence of random variables in which the distribution of each element depends only on the value of the previous element.

In MCMC simulation, you constructs a Markov chain long enough for the distribution of the elements to stabilize to a stationary distribution, which is the distribution of interest. Repeatedly simulating steps of the chain simulates draws from the distribution of interest. See Schafer (1997) for a detailed discussion of this method.

In Bayesian inference, information about unknown parameters is expressed in the form of a posterior probability distribution. This posterior distribution is computed using Bayes' theorem,

$$p(\boldsymbol{\theta}|y) = \frac{p(y|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(y|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

MCMC has been applied as a method for exploring posterior distributions in Bayesian inference. That is, through MCMC, you can simulate the entire joint posterior distribution of the unknown quantities and obtain simulation-based estimates of posterior parameters that are of interest.

In many incomplete-data problems, the observed-data posterior $p(\boldsymbol{\theta}|Y_{obs})$ is intractable and cannot easily be simulated. However, when Y_{obs} is augmented by an estimated or simulated value of the missing data Y_{mis}, the complete-data posterior $p(\boldsymbol{\theta}|Y_{obs}, Y_{mis})$ is much easier to simulate. Assuming that the data are from a multivariate normal distribution, data augmentation can be applied to Bayesian inference with missing data by repeating the following steps:

1. The imputation I-step
Given an estimated mean vector and covariance matrix, the I-step simulates the missing values for each observation independently. That is, if you denote the variables with missing values for observation i by $Y_{i(mis)}$ and the variables with observed values by $Y_{i(obs)}$, then the I-step draws values for $Y_{i(mis)}$ from a conditional distribution for $Y_{i(mis)}$ given $Y_{i(obs)}$.

2. The posterior P-step
Given a complete sample, the P-step simulates the posterior population mean vector and covariance matrix. These new estimates are then used in the next I-step. Without prior information about the parameters, a noninformative prior is used. You can also use other informative priors. For example, a prior information about the covariance matrix can help to stabilize the inference about the mean vector for a near singular covariance matrix.

That is, with a current parameter estimate $\boldsymbol{\theta}^{(t)}$ at the tth iteration, the I-step draws $Y_{mis}^{(t+1)}$ from $p(Y_{mis}|Y_{obs}, \boldsymbol{\theta}^{(t)})$ and the P-step draws $\boldsymbol{\theta}^{(t+1)}$ from $p(\boldsymbol{\theta}|Y_{obs}, Y_{mis}^{(t+1)})$. The two steps are iterated long enough for the results to reliably simulate an approximately independent draw of the missing values for a multiply imputed data set (Schafer 1997).

This creates a Markov chain $(Y_{mis}^{(1)}, \boldsymbol{\theta}^{(1)})$, $(Y_{mis}^{(2)}, \boldsymbol{\theta}^{(2)})$, $\ldots$, which converges in distribution to $p(Y_{mis}, \boldsymbol{\theta}|Y_{obs})$. Assuming the iterates converge to a stationary distribution, the goal is to simulate an approximately independent draw of the missing values from this distribution.

To validate the imputation results, you should repeat the process with different random number generators and starting values based on different initial parameter estimates.

The next three sections provide details for the imputation step, Bayesian estimation of the mean vector and covariance matrix, and the posterior step.

Imputation Step

In each iteration, starting with a given mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, the imputation step draws values for the missing data from the conditional distribution Y_{mis} given Y_{obs}.

Suppose $\boldsymbol{\mu} = \left[\boldsymbol{\mu}_1', \boldsymbol{\mu}_2'\right]'$ is the partitioned mean vector of two sets of variables, Y_{obs} and Y_{mis}, where $\boldsymbol{\mu}_1$ is the mean vector for variables Y_{obs} and $\boldsymbol{\mu}_2$ is the mean vector for variables Y_{mis}.

Also suppose

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{12}' & \boldsymbol{\Sigma}_{22} \end{bmatrix}$$

is the partitioned covariance matrix for these variables, where $\boldsymbol{\Sigma}_{11}$ is the covariance matrix for variables Y_{obs}, $\boldsymbol{\Sigma}_{22}$ is the covariance matrix for variables Y_{mis}, and $\boldsymbol{\Sigma}_{12}$ is the covariance matrix between variables Y_{obs} and variables Y_{mis}.

By using the sweep operator (Goodnight 1979) on the pivots of the $\boldsymbol{\Sigma}_{11}$ submatrix, the matrix becomes

$$\begin{bmatrix} \boldsymbol{\Sigma}_{11}^{-1} & \boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12} \\ -\boldsymbol{\Sigma}_{12}'\boldsymbol{\Sigma}_{11}^{-1} & \boldsymbol{\Sigma}_{22.1} \end{bmatrix}$$

where $\boldsymbol{\Sigma}_{22.1} = \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{12}'\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}$ can be used to compute the conditional covariance matrix of $\mathbf{Y}_{mis}$ after controlling for $\mathbf{Y}_{obs}$.

For an observation with the preceding missing pattern, the conditional distribution of Y_{mis} given $Y_{obs} = \mathbf{y}_1$ is a multivariate normal distribution with the mean vector

$$\boldsymbol{\mu}_{2.1} = \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{12}'\boldsymbol{\Sigma}_{11}^{-1}(\mathbf{y}_1 - \boldsymbol{\mu}_1)$$

and the conditional covariance matrix

$$\boldsymbol{\Sigma}_{22.1} = \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{12}'\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}$$

Bayesian Estimation of the Mean Vector and Covariance Matrix

Suppose that $\mathbf{Y} = (\mathbf{y}_1', \mathbf{y}_2', \ldots, \mathbf{y}_n')'$ is an $(n \times p)$ matrix made up of n $(p \times 1)$ independent vectors $\mathbf{y}_i$, each of which has a multivariate normal distribution with mean zero and covariance matrix $\boldsymbol{\Lambda}$. Then the SSCP matrix

$$\mathbf{A} = \mathbf{Y}'\mathbf{Y} = \sum_i \mathbf{y}_i\mathbf{y}_i'$$

has a Wishart distribution $W(n, \boldsymbol{\Lambda})$.

When each observation $\mathbf{y}_i$ is distributed with a multivariate normal distribution with an unknown mean $\boldsymbol{\mu}$, then the CSSCP matrix

$$\mathbf{A} = \sum_i (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})'$$

has a Wishart distribution $W(n-1, \boldsymbol{\Lambda})$.

If $\mathbf{A}$ has a Wishart distribution $W(n, \boldsymbol{\Lambda})$, then $\mathbf{B} = \mathbf{A}^{-1}$ has an inverted Wishart distribution $W^{-1}(n, \boldsymbol{\Psi})$, where n is the degrees of freedom and $\boldsymbol{\Psi} = \boldsymbol{\Lambda}^{-1}$ is the precision matrix (Anderson 1984).

Note that, instead of using the parameter $\boldsymbol{\Psi} = \boldsymbol{\Lambda}^{-1}$ for the inverted Wishart distribution, Schafer (1997) uses the parameter $\boldsymbol{\Lambda}$.

Suppose that each observation in the data matrix $\mathbf{Y}$ has a multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. Then with a prior inverted Wishart distribution for $\boldsymbol{\Sigma}$ and a prior normal distribution for $\boldsymbol{\mu}$

$$\begin{aligned} \boldsymbol{\Sigma} &\sim W^{-1}(m, \boldsymbol{\Psi}) \\ \boldsymbol{\mu}|\boldsymbol{\Sigma} &\sim N\left(\boldsymbol{\mu}_0, \frac{1}{\tau}\boldsymbol{\Sigma}\right) \end{aligned}$$

where $\tau > 0$ is a fixed number.

The posterior distribution (Anderson 1984, p. 270; Schafer 1997, p. 152) is

$$\begin{aligned} \boldsymbol{\Sigma}|\mathbf{Y} &\sim W^{-1}\left(n+m,\ (n-1)\mathbf{S} + \boldsymbol{\Psi} + \frac{n\tau}{n+\tau}(\overline{\mathbf{y}} - \boldsymbol{\mu}_0)(\overline{\mathbf{y}} - \boldsymbol{\mu}_0)'\right) \\ \boldsymbol{\mu}|(\boldsymbol{\Sigma}, \mathbf{Y}) &\sim N\left(\frac{1}{n+\tau}(n\overline{\mathbf{y}} + \tau\boldsymbol{\mu}_0),\ \frac{1}{n+\tau}\boldsymbol{\Sigma}\right) \end{aligned}$$

where $(n-1)\mathbf{S}$ is the CSSCP matrix.

Posterior Step

In each iteration, the posterior step simulates the posterior population mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ from prior information for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, and the complete sample estimates.

You can specify the prior parameter information by using one of the following methods:

- PRIOR=JEFFREYS, which uses a noninformative prior
- PRIOR=INPUT=, which provides a prior information for $\boldsymbol{\Sigma}$ in the data set. Optionally, it also provides a prior information for $\boldsymbol{\mu}$ in the data set.
- PRIOR=RIDGE=, which uses a ridge prior

The next four subsections provide details of the posterior step for different prior distributions.

1. A Noninformative Prior

Without prior information about the mean and covariance estimates, you can use a noninformative prior by specifying the PRIOR=JEFFREYS option. The posterior distributions (Schafer 1997, p. 154) are

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} &\sim W^{-1}\left(n-1,\ (n-1)\mathbf{S}\right) \\ \boldsymbol{\mu}^{(t+1)}|(\boldsymbol{\Sigma}^{(t+1)},\mathbf{Y}) &\sim N\left(\bar{\mathbf{y}},\ \frac{1}{n}\boldsymbol{\Sigma}^{(t+1)}\right) \end{aligned}$$

2. An Informative Prior for μ and Σ

When prior information is available for the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, you can provide it with a SAS data set that you specify with the PRIOR=INPUT= option:

$$\begin{aligned} \boldsymbol{\Sigma} &\sim W^{-1}\left(d^*,\ d^*\mathbf{S}^*\right) \\ \boldsymbol{\mu}|\boldsymbol{\Sigma} &\sim N\left(\boldsymbol{\mu}_0,\ \frac{1}{n_0}\boldsymbol{\Sigma}\right) \end{aligned}$$

To obtain the prior distribution for $\boldsymbol{\Sigma}$, PROC MI reads the matrix $\mathbf{S}^*$ from observations in the data set with _TYPE_='COV', and it reads $n^* = d^* + 1$ from observations with _TYPE_='N'.

To obtain the prior distribution for $\boldsymbol{\mu}$, PROC MI reads the mean vector $\boldsymbol{\mu}_0$ from observations with _TYPE_='MEAN', and it reads n_0 from observations with _TYPE_='N_MEAN'. When there are no observations with _TYPE_='N_MEAN', PROC MI reads n_0 from observations with _TYPE_='N'.

The resulting posterior distribution, as described in the section "Bayesian Estimation of the Mean Vector and Covariance Matrix" on page 6602, is given by

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} &\sim W^{-1}\left(n+d^*,\ (n-1)\mathbf{S}+d^*\mathbf{S}^*+\mathbf{S}_m\right) \\ \boldsymbol{\mu}^{(t+1)}\mid\left(\boldsymbol{\Sigma}^{(t+1)},\mathbf{Y}\right) &\sim N\left(\frac{1}{n+n_0}(n\bar{\mathbf{y}}+n_0\boldsymbol{\mu}_0),\ \frac{1}{n+n_0}\boldsymbol{\Sigma}^{(t+1)}\right) \end{aligned}$$

where

$$\mathbf{S}_m = \frac{nn_0}{n+n_0}(\bar{\mathbf{y}}-\boldsymbol{\mu}_0)(\bar{\mathbf{y}}-\boldsymbol{\mu}_0)'$$

3. An Informative Prior for Σ

When the sample covariance matrix $\mathbf{S}$ is singular or near singular, prior information about $\boldsymbol{\Sigma}$ can also be used without prior information about $\boldsymbol{\mu}$ to stabilize the inference about $\boldsymbol{\mu}$. You can provide it with a SAS data set that you specify with the PRIOR=INPUT= option.

To obtain the prior distribution for $\boldsymbol{\Sigma}$, PROC MI reads the matrix $\mathbf{S}^*$ from observations in the data set with _TYPE_='COV', and it reads n^* from observations with _TYPE_='N'.

The resulting posterior distribution for $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ (Schafer 1997, p. 156) is

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} &\sim W^{-1}\left(n+d^*,\ (n-1)\mathbf{S}+d^*\mathbf{S}^*\right) \\ \boldsymbol{\mu}^{(t+1)}\mid\left(\boldsymbol{\Sigma}^{(t+1)},\mathbf{Y}\right) &\sim N\left(\bar{\mathbf{y}},\ \frac{1}{n}\boldsymbol{\Sigma}^{(t+1)}\right) \end{aligned}$$

Note that if the PRIOR=INPUT= data set also contains observations with _TYPE_='MEAN', then a complete informative prior for both $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ will be used.

4. A Ridge Prior

A special case of the preceding adjustment is a ridge prior with $\mathbf{S}^* = \text{Diag}(\mathbf{S})$ (Schafer 1997, p. 156). That is, $\mathbf{S}^*$ is a diagonal matrix with diagonal elements equal to the corresponding elements in $\mathbf{S}$.

You can request a ridge prior by using the PRIOR=RIDGE= option. You can explicitly specify the number $d^* \geq 1$ in the PRIOR=RIDGE=d^* option. Or you can implicitly specify the number by specifying the proportion p in the PRIOR=RIDGE=p option to request $d^* = (n-1)p$.

The posterior is then given by

$$\begin{aligned} \boldsymbol{\Sigma}^{(t+1)}|\mathbf{Y} &\sim W^{-1}\left(n + d^*,\ (n-1)\mathbf{S} + d^*\text{Diag}(\mathbf{S})\right) \\ \boldsymbol{\mu}^{(t+1)} \,\Big|\, \left(\boldsymbol{\Sigma}^{(t+1)}, \mathbf{Y}\right) &\sim N\left(\bar{\mathbf{y}},\ \frac{1}{n}\boldsymbol{\Sigma}^{(t+1)}\right) \end{aligned}$$

Producing Monotone Missingness with the MCMC Method

The monotone data MCMC method was first proposed by Li (1988) and Liu (1993) described the algorithm. The method is useful especially when a data set is close to having a monotone missing pattern. In this case, the method needs to impute only a few missing values to the data set to have a monotone missing pattern in the imputed data set. Compared to a full data imputation that imputes all missing values, the monotone data MCMC method imputes fewer missing values in each iteration and achieves approximate stationarity in fewer iterations (Schafer 1997, p. 227).

You can request the monotone MCMC method by specifying the option IMPUTE=MONOTONE in the MCMC statement. The "Missing Data Patterns" table now denotes the variables with missing values by "." or "O". The value "." means that the variable is missing and will be imputed, and the value "O" means that the variable is missing and will not be imputed. The "Variance Information" and "Parameter Estimates" tables are not created.

You must specify the variables in the VAR statement. The variable order in the list determines the monotone missing pattern in the imputed data set. With a different order in the VAR list, the results will be different because the monotone missing pattern to be constructed will be different.

Different variable orders require a different number of imputations in order to generate a monotone pattern. When you specify the MONOTONESEARCH option in the MCMC statement, PROC MI optimizes the order of the analysis variables so that the smallest number of imputations is required. It also produces a table that shows the number of imputations that would be required for generating a monotone order, given different variable orders.

Assuming that the data are from a multivariate normal distribution, then like the MCMC method, the monotone MCMC method repeats the following steps:

1. The imputation I-step

Given an estimated mean vector and covariance matrix, the I-step simulates the missing values for each observation independently. Only a subset of missing values are simulated to achieve a monotone pattern of missingness.

2. The posterior P-step
Given a new sample with a monotone pattern of missingness, the P-step simulates the posterior population mean vector and covariance matrix with a noninformative Jeffreys prior. These new estimates are then used in the next I-step.

Imputation Step

The I-step is almost identical to the I-step described in the section "MCMC Method for Arbitrary Missing Multivariate Normal Data" on page 6601 except that only a subset of missing values need to be simulated. To state this precisely, denote the variables with observed values for observation i by $Y_{i(obs)}$ and the variables with missing values by $Y_{i(mis)} = (Y_{i(m1)}, Y_{i(m2)})$, where $Y_{i(m1)}$ is a subset of the missing variables that will cause a monotone missingness when their values are imputed. Then the I-step draws values for $Y_{i(m1)}$ from a conditional distribution for $Y_{i(m1)}$ given $Y_{i(obs)}$.

Posterior Step

The P-step is different from the P-step described in the section "MCMC Method for Arbitrary Missing Multivariate Normal Data" on page 6601. Instead of simulating the $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ parameters from the full imputed data set, this P-step simulates the $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ parameters through simulated regression coefficients from regression models based on the imputed data set with a monotone pattern of missingness. The step is similar to the process described in the section "Monotone and FCS Regression Methods" on page 6590.

That is, for the variable Y_j, a model

$$Y_j = \beta_0 + \beta_1 Y_1 + \beta_2 Y_2 + \cdots + \beta_{j-1} Y_{j-1}$$

is fitted using n_j nonmissing observations for variable Y_j in the imputed data sets.

The fitted model consists of the regression parameter estimates $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_{j-1})$ and the associated covariance matrix $\hat{\sigma}_j^2 \mathbf{V}_j$, where $\mathbf{V}_j$ is the usual $\mathbf{X}'\mathbf{X}$ inverse matrix from the intercept and variables $Y_1, Y_2, \ldots, Y_{j-1}$.

For each imputation, new parameters $\boldsymbol{\beta}_* = (\beta_{*0}, \beta_{*1}, \ldots, \beta_{*(j-1)})$ and σ_{*j}^2 are drawn from the posterior predictive distribution of the parameters. That is, they are simulated from $(\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_{j-1})$, σ_j^2, and $\mathbf{V}_j$. The variance is drawn as

$$\sigma_{*j}^2 = \hat{\sigma}_j^2 (n_j - j)/g$$

where g is a $\chi^2_{n_j - p + j - 1}$ random variate and n_j is the number of nonmissing observations for Y_j. The regression coefficients are drawn as

$$\boldsymbol{\beta}_* = \hat{\boldsymbol{\beta}} + \sigma_{*j} \mathbf{V}'_{hj} \mathbf{Z}$$

where $\mathbf{V}_{hj}$ is the upper triangular matrix in the Cholesky decomposition, $\mathbf{V}_j = \mathbf{V}'_{hj} \mathbf{V}_{hj}$, and $\mathbf{Z}$ is a vector of j independent random normal variates.

These simulated values of $\boldsymbol{\beta}_*$ and σ_{*j}^2 are then used to re-create the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. For a detailed description of how to produce monotone missingness with the MCMC method for a multivariate normal data, see Schafer (1997, pp. 226–235).

MONOTONESEARCH Option

The number of imputations that are required in order to achieve a monotone missing data pattern depends on the order of the variables. The data set Fitness1 that is created in the section "Getting Started: MI Procedure" on page 6555 can be used to illustrate this. The data set consists of three variables: Oxygen, RunTime, and RunPulse. This data set has the missing data patterns shown in Figure 82.9.

Figure 82.9 Missing Data Patterns

The MI Procedure

Missing Data Patterns					
Group	**Oxygen**	**RunTime**	**RunPulse**	**Freq**	**Percent**
1	X	X	X	21	67.74
2	X	X	O	4	12.90
3	X	O	O	3	9.68
4	.	X	X	1	3.23
5	.	X	O	2	6.45

An "X" in Figure 82.9 means that the variable is observed in the corresponding group, and a "." means that the variable is missing. PROC MI denotes variables that have missing values by "." or "O". The value "." means that the variable is missing and will be imputed. The value "O" means that the variable is missing and will not be imputed to create a monotone missing data pattern. This pattern shows that only one missing value from group 4 and two missing values from group 5 need to be imputed. If you change the variable order, the number of required imputations might change as well. For example, if the variables are ordered as Oxygen, RunPulse, RunTime, then the number of required imputations is 9, as shown in Figure 82.10.

Figure 82.10 Missing Data Patterns

The MI Procedure

Missing Data Patterns					
Group	**Oxygen**	**RunPulse**	**RunTime**	**Freq**	**Percent**
1	X	X	X	21	67.74
2	X	.	X	4	12.90
3	X	O	O	3	9.68
4	.	X	X	1	3.23
5	.	.	X	2	6.45

For a data set that contains N variables, there are $N!$ possible orderings, each of which might require a different number of imputations in order to create a monotone pattern. As the number of variables grows, it becomes difficult to determine whether a data set is close to monotone and whether a different variable order would require a smaller number of imputations.

When you specify the MONOTONESEARCH option in the MCMC statement, a hybrid optimization scheme suggests variable orders. These orders reduce the number of imputations that are required in order to achieve a monotone missing data pattern when you specify the IMPUTE=MONOTONE option. PROC MI reports the cost of imputing the data set to a monotone pattern both as the raw number of missing values that need to be filled and as the percentage of all missing values that this number represents. It returns values for the following three variable orders:

- the order that you specify in the VAR statement
- the order by increasing missingness, as recommended by Van Buuren (2019)
- the order that is suggested by optimization

For data sets that contain 10 or fewer variables, PROC MI uses a nonrecursive Heap's algorithm to efficiently generate all possible variable orders and their cost. This guarantees that the optimal variable order is found. For larger data sets, it uses simulated annealing to search for a variable order that minimizes the cost of imputation. You can specify the METHOD= option to request that PROC MI use a specific method regardless of the number of variables.

You can specify the STARTTEMP= option to change the starting temperature that the simulated annealing algorithm uses. The value of the temperature during iterations of the simulated annealing algorithm influences the acceptance step of a proposed variable permutation. A higher temperature value makes acceptance of a proposed order that requires more imputations than the previous one more likely. This helps the algorithm escape local minima. PROC MI always returns the lowest cost that it encounters during the algorithm execution. By default, STARTTEMP=10.

You can use the MAXITER= option to set the maximum number of iterations of the simulated annealing algorithm. By default, MAXITER=2500.

PROC MI implements the fast annealing method as described by Szu and Hartley (1987). The temperature t is lowered according to the schedule $t = \frac{\text{STARTTEMP}}{k}$, where k is the current iteration number. If you increase the number of iterations, you might also want to increase the starting temperature to allow the algorithm to explore more of the solution space, because at very low temperatures only steps that reduce the number of imputations are accepted.

MCMC Method Specifications

With the MCMC method, you can impute either all missing values (IMPUTE=FULL) or just enough missing values to make the imputed data set have a monotone missing pattern (IMPUTE=MONOTONE). In the process, either a single chain for all imputations (CHAIN=SINGLE) or a separate chain for each imputation (CHAIN=MULTIPLE) is used. The single chain might be somewhat more precise for estimating a single quantity such as a posterior mean (Schafer 1997, p. 138). See Schafer (1997, pp. 137–138) for a discussion of single versus multiple chains.

You can specify the number of initial burn-in iterations before the first imputation with the NBITER= option. This number is also used for subsequent chains for multiple chains. For a single chain, you can also specify the number of iterations between imputations with the NITER= option.

You can explicitly specify initial parameter values for the MCMC method with the INITIAL=INPUT= data set option. Alternatively, you can use the EM algorithm to derive a set of initial parameter values for MCMC with the option INITIAL=EM. These estimates are used as either the starting value (START=VALUE) or the starting distribution (START=DIST) for the MCMC method. For multiple chains, these estimates are used again as either the starting value (START=VALUE) or the starting distribution (START=DIST) for the subsequent chains.

You can specify the prior parameter information in the PRIOR= option. You can use a noninformative prior (PRIOR=JEFFREYS), a ridge prior (PRIOR=RIDGE), or an informative prior specified in a data set (PRIOR=INPUT).

The parameter estimates used to generate imputed values in each imputation can be saved in a data set with the OUTEST= option. Later, this data set can be read with the INEST= option to provide the reference distribution for imputing missing values for a new data set.

By default, the MCMC method uses a single chain to produce five imputations. It completes 200 burn-in iterations before the first imputation and 100 iterations between imputations. The posterior mode computed from the EM algorithm with a noninformative prior is used as the starting values for the MCMC method.

INITIAL=EM Specifications

The EM algorithm is used to find the maximum likelihood estimates for incomplete data in the EM statement. You can also use the EM algorithm to find a posterior mode, the parameter estimates that maximize the observed-data posterior density. The resulting posterior mode provides a good starting value for the MCMC method.

With the INITIAL=EM option, PROC MI uses the MLE of the parameter vector as the initial estimates in the EM algorithm for the posterior mode. You can use the ITPRINT option within the INITIAL=EM option to display the iteration history for the EM algorithm.

You can use the CONVERGE= option to specify the convergence criterion in deriving the EM posterior mode. The iterations are considered to have converged when the maximum change in the parameter estimates between iteration steps is less than the value specified. By default, CONVERGE=1E–4.

You can also use the MAXITER= option to specify the maximum number of iterations of the EM algorithm. By default, MAXITER=200.

With the BOOTSTRAP option, you can use overdispersed starting values for the MCMC method. In this case, PROC MI applies the EM algorithm to a bootstrap sample, a simple random sample with replacement from the input data set, to derive the initial estimates for each chain (Schafer 1997, p. 128).

Checking Convergence in MCMC

The theoretical convergence of the MCMC method has been explored under various conditions, as described in Schafer (1997, p. 70). However, in practice, verification of convergence is not a simple matter.

The parameters used in the imputation step for each iteration can be saved in an output data set with the OUTITER= option. These include the means, standard deviations, covariances, worst linear function, and observed-data LR statistics. You can then monitor the convergence in a single chain by displaying trace plots and autocorrelations for those parameter values (Schafer 1997, p. 120). The trace and autocorrelation function plots for parameters such as variable means, covariances, and the worst linear function can be displayed by specifying the PLOTS=TRACE and PLOTS=ACF options, respectively.

You can apply the EM algorithm to a bootstrap sample to obtain overdispersed starting values for multiple chains (Gelman and Rubin 1992). This provides a conservative estimate of the number of iterations needed before each imputation.

The next four subsections describe useful statistics and plots that can be used to check the convergence of the MCMC method.

LR Statistics

You can save the observed-data likelihood ratio (LR) statistic in each iteration with the LR option in the OUTITER= data set. The statistic is based on the observed-data likelihood with parameter values used in the iteration and the observed-data maximum likelihood derived from the EM algorithm.

In each iteration, the LR statistic is given by

$$-2\log\left(\frac{f(\hat{\boldsymbol{\theta}}_i)}{f(\hat{\boldsymbol{\theta}})}\right)$$

where $f(\hat{\boldsymbol{\theta}})$ is the observed-data maximum likelihood derived from the EM algorithm and $f(\hat{\boldsymbol{\theta}}_i)$ is the observed-data likelihood for $\hat{\boldsymbol{\theta}}_i$ used in the iteration.

Similarly, you can also save the observed-data LR posterior mode statistic for each iteration with the LR_POST option. This statistic is based on the observed-data posterior density with parameter values used in each iteration and the observed-data posterior mode derived from the EM algorithm for posterior mode.

For large samples, these LR statistics tends to be approximately χ^2 distributed with degrees of freedom equal to the dimension of $\boldsymbol{\theta}$ (Schafer 1997, p. 131). For example, with a large number of iterations, if the values of the LR statistic do not behave like a random sample from the described χ^2 distribution, then there is evidence that the MCMC method has not converged.

Worst Linear Function of Parameters

The worst linear function (WLF) of parameters (Schafer 1997, pp. 129–131) is a scalar function of parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ that is "worst" in the sense that its function values converge most slowly among parameters in the MCMC method. The convergence of this function is evidence that other parameters are likely to converge as well.

For linear functions of parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$, a worst linear function of $\boldsymbol{\theta}$ has the highest asymptotic rate of missing information. The function can be derived from the iterative values of $\boldsymbol{\theta}$ near the posterior mode in the EM algorithm. That is, an estimated worst linear function of $\boldsymbol{\theta}$ is

$$w(\boldsymbol{\theta}) = \mathbf{v}'(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})$$

where $\hat{\boldsymbol{\theta}}$ is the posterior mode and the coefficients $\mathbf{v} = \hat{\boldsymbol{\theta}}_{(-1)} - \hat{\boldsymbol{\theta}}$ are the difference between the estimated value of $\boldsymbol{\theta}$ one step prior to convergence and the converged value $\hat{\boldsymbol{\theta}}$.

You can display the coefficients of the worst linear function, $\mathbf{v}$, by specifying the WLF option in the MCMC statement. You can save the function value from each iteration in an OUTITER= data set by specifying the WLF option within the OUTITER option. You can also display the worst linear function values from iterations in an autocorrelation plot or a trace plot by specifying PLOTS=ACF(WLF) or PLOTS=TRACE(WLF), respectively.

Note that when the observed-data posterior is nearly normal, the WLF is one of the slowest functions to approach stationarity. When the posterior is not close to normal, other functions might take much longer than the WLF to converge, as described in Schafer (1997, p.130).

Trace Plot

A trace plot for a parameter ξ is a scatter plot of successive parameter estimates ξ_i against the iteration number i. The plot provides a simple way to examine the convergence behavior of the estimation algorithm for ξ. Long-term trends in the plot indicate that successive iterations are highly correlated and that the series of iterations has not converged.

You can display trace plots for worst linear function, variable means, variable variances, and covariances of variables. You can also request logarithmic transformations for positive parameters in the plots with the LOG option. When a parameter value is less than or equal to zero, the value is not displayed in the corresponding plot. See Example 82.11 for a usage of the trace plot.

Autocorrelation Function Plot

To examine relationships of successive parameter estimates ξ, the autocorrelation function (ACF) can be used. For a stationary series, $\xi_i, i \geq 1$, in trace data, the autocorrelation function at lag k is

$$\rho_k = \frac{\mathrm{Cov}(\xi_i, \xi_{i+k})}{\mathrm{Var}(\xi_i)}$$

The sample kth order autocorrelation is computed as

$$r_k = \frac{\sum_{i=1}^{n-k}(\xi_i - \bar{\xi})(\xi_{i+k} - \bar{\xi})}{\sum_{i=1}^{n}(\xi_i - \bar{\xi})^2}$$

You can display autocorrelation function plots for the worst linear function, variable means, variable variances, and covariances of variables. You can also request logarithmic transformations for parameters in the plots with the LOG option. When a parameter has values less than or equal to zero, the corresponding plot is not created.

You specify the maximum number of lags of the series with the NLAG= option. The autocorrelations at each lag less than or equal to the specified lag are displayed in the graph. In addition, the plot also displays approximate 95% confidence limits for the autocorrelations. At lag k, the confidence limits indicate a set of approximate 95% critical values for testing the hypothesis $\rho_j = 0, j \geq k$. See Example 82.11 for an illustration of the autocorrelation function plot.

Input Data Sets

You can specify the input data set with missing values by using the DATA= option in the PROC MI statement. When an MCMC method is used, you can specify the data set that contains the reference distribution information for imputation with the INEST= option, the data set that contains initial parameter estimates for the MCMC method with the INITIAL=INPUT= option, and the data set that contains information for the prior distribution with the PRIOR=INPUT= option in the MCMC statement.

When the ADJUST option is specified in the MNAR statement, you can use the PARMS= option to specify the data set that contains adjustment parameters for the sensitivity analysis.

DATA=*SAS-data-set*

The input DATA= data set is an ordinary SAS data set that contains multivariate data with missing values.

INEST=*SAS-data-set*

The input INEST= data set is a TYPE=EST data set and contains a variable _Imputation_ to identify the imputation number. For each imputation, PROC MI reads the point estimate from the observations with _TYPE_='PARM' or _TYPE_='PARMS' and the associated covariances from the observations with _TYPE_='COV' or _TYPE_='COVB'. These estimates are used as the reference distribution to impute values for observations in the DATA= data set. When the input INEST= data set also contains observations with _TYPE_='SEED', PROC MI reads the seed information for the random number generator from these observations. Otherwise, the SEED= option provides the seed information.

INITIAL=INPUT=*SAS-data-set*

The input INITIAL=INPUT= data set is a TYPE=COV or CORR data set and provides initial parameter estimates for the MCMC method. The covariances derived from the TYPE=COV/CORR data set are divided by the number of observations to get the correct covariance matrix for the point estimate (sample mean).

If TYPE=COV, PROC MI reads the number of observations from the observations with _TYPE_='N', the point estimate from the observations with _TYPE_='MEAN', and the covariances from the observations with _TYPE_='COV'.

If TYPE=CORR, PROC MI reads the number of observations from the observations with _TYPE_='N', the point estimate from the observations with _TYPE_='MEAN', the correlations from the observations with _TYPE_='CORR', and the standard deviations from the observations with _TYPE_='STD'.

PARMS= *SAS-data-set*

The input PARMS= data set is an ordinary SAS data set that contains adjustment parameters for imputed values of the specified imputed variables.

The PARMS= data set contains variables _Imputation_ for the imputation number, the SHIFT= or DELTA= variable for the shift parameter, and the SCALE= variable for the scale parameter. Either the shift or scale variable must be included in the data set.

PRIOR=INPUT=*SAS-data-set*

The input PRIOR=INPUT= data set is a TYPE=COV data set that provides information for the prior distribution. You can use the data set to specify a prior distribution for $\mathbf{\Sigma}$ of the form

$$\mathbf{\Sigma} \sim W^{-1}\left(d^*,\, d^*\mathbf{S}^*\right)$$

where $d^* = n^* - 1$ is the degrees of freedom. PROC MI reads the matrix $\mathbf{S}^*$ from observations with _TYPE_='COV' and reads n^* from observations with _TYPE_='N'.

You can also use this data set to specify a prior distribution for $\boldsymbol{\mu}$ of the form

$$\boldsymbol{\mu} \sim N\left(\boldsymbol{\mu}_0,\, \frac{1}{n_0}\mathbf{\Sigma}\right)$$

PROC MI reads the mean vector $\boldsymbol{\mu}_0$ from observations with _TYPE_='MEAN' and reads n_0 from observations with _TYPE_='N_MEAN'. When there are no observations with _TYPE_='N_MEAN', PROC MI reads n_0 from observations with _TYPE_='N'.

Output Data Sets

You can specify the output data set of imputed values with the OUT= option in the PROC MI statement. When an EM statement is used, you can specify the data set that contains the original data set with missing values being replaced by the expected values from the EM algorithm by using the OUT= option in the EM statement. You can also specify the data set that contains MLE computed with the EM algorithm by using the OUTEM= option.

When an MCMC method is used, you can specify the data set that contains parameter estimates used in each imputation with the OUTEST= option in the MCMC statement, and you can specify the data set that contains parameters used in the imputation step for each iteration with the OUTITER option in the MCMC statement.

OUT=*SAS-data-set* in the PROC MI statement

The OUT= data set contains all the variables in the original data set and a new variable named _Imputation_ that identifies the imputation. For each imputation, the data set contains all variables in the input DATA= data set with missing values being replaced by imputed values. Note that when the NIMPUTE=1 option is specified, the variable _Imputation_ is not created.

OUT=*SAS-data-set* in an EM statement

The OUT= data set contains the original data set with missing values being replaced by expected values from the EM algorithm.

OUTEM=*SAS-data-set*

The OUTEM= data set is a TYPE=COV data set and contains the MLE computed with the EM algorithm. The observations with _TYPE_='MEAN' contain the estimated mean and the observations with _TYPE_='COV' contain the estimated covariances.

OUTEST=*SAS-data-set*

The OUTEST= data set is a TYPE=EST data set and contains parameter estimates used in each imputation in the MCMC method. It also includes an index variable named _Imputation_, which identifies the imputation.

The observations with _TYPE_='SEED' contain the seed information for the random number generator. The observations with _TYPE_='PARM' or _TYPE_='PARMS' contain the point estimate, and the observations with _TYPE_='COV' or _TYPE_='COVB' contain the associated covariances. These estimates are used as the parameters of the reference distribution to impute values for observations in the DATA= data set.

Note that these estimates are the values used in the I-step before each imputation. These are not the parameter values simulated from the P-step in the same iteration. See Example 82.12 for a usage of this option.

OUTITER <(*options*)> =*SAS-data-set* in an EM statement

The OUTITER= data set in an EM statement is a TYPE=COV data set and contains parameters for each iteration. It also includes a variable _Iteration_ that provides the iteration number.

The parameters in the output data set depend on the options specified. You can specify the MEAN and COV options for OUTITER. With the MEAN option, the output data set contains the mean parameters in observations with the variable _TYPE_='MEAN'. Similarly, with the COV option, the output data set contains the covariance parameters in observations with the variable _TYPE_='COV'. When no options are specified, the output data set contains the mean parameters for each iteration.

OUTITER <(*options*)> =*SAS-data-set* in an FCS statement

The OUTITER= data set in an FCS statement is a TYPE=COV data set and contains parameters for each iteration. It also includes variables named _Imputation_ and _Iteration_, which provide the imputation number and iteration number.

The parameters in the output data set depend on the options specified. You can specify the MEAN and STD options for OUTITER. With the MEAN option, the output data set contains the mean parameters used in the imputation in observations with the variable _TYPE_='MEAN'. Similarly, with the STD option, the output data set contains the standard deviation parameters used in the imputation in observations with the variable _TYPE_='STD'. When no options are specified, the output data set contains the mean parameters for each iteration.

OUTITER <(*options*)> =*SAS-data-set* in an MCMC statement

The OUTITER= data set in an MCMC statement is a TYPE=COV data set and contains parameters used in the imputation step for each iteration. It also includes variables named _Imputation_ and _Iteration_, which provide the imputation number and iteration number.

The parameters in the output data set depend on the options specified. Table 82.6 summarizes the options available for OUTITER and the corresponding values for the output variable _TYPE_.

Table 82.6 Summary of Options for OUTITER in an MCMC Statement

Option	Output Parameters	_TYPE_
MEAN	Mean parameters	MEAN
STD	Standard deviations	STD
COV	Covariances	COV
LR	–2 log LR statistic	LOG_LR
LR_POST	–2 log LR statistic of the posterior mode	LOG_POST
WLF	Worst linear function	WLF

When no options are specified, the output data set contains the mean parameters used in the imputation step for each iteration. For a detailed description of the worst linear function and LR statistics, see the section "Checking Convergence in MCMC" on page 6609.

Combining Inferences from Multiply Imputed Data Sets

With m imputations, m different sets of the point and variance estimates for a parameter Q can be computed. Suppose $\hat{Q}_i$ and $\hat{W}_i$ are the point and variance estimates from the ith imputed data set, $i = 1, 2, \ldots, m$. Then the combined point estimate for Q from multiple imputation is the average of the m complete-data estimates:

$$\overline{Q} = \frac{1}{m}\sum_{i=1}^{m}\hat{Q}_i$$

Suppose $\overline{W}$ is the within-imputation variance, which is the average of the m complete-data estimates,

$$\overline{W} = \frac{1}{m}\sum_{i=1}^{m}\hat{W}_i$$

and B is the between-imputation variance

$$B = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{Q}_i - \overline{Q})^2$$

Then the variance estimate associated with $\overline{Q}$ is the total variance (Rubin 1987)

$$T = \overline{W} + (1 + \frac{1}{m})B$$

The statistic $(Q - \overline{Q})T^{-(1/2)}$ is approximately distributed as t with v_m degrees of freedom (Rubin 1987), where

$$v_m = (m-1)\left[1 + \frac{\overline{W}}{(1+m^{-1})B}\right]^2$$

The degrees of freedom v_m depend on m and the ratio

$$r = \frac{(1+m^{-1})B}{\overline{W}}$$

The ratio r is called the relative increase in variance due to nonresponse (Rubin 1987). When there is no missing information about Q, the values of r and B are both zero. With a large value of m or a small value of r, the degrees of freedom v_m will be large and the distribution of $(Q - \overline{Q})T^{-(1/2)}$ will be approximately normal.

Another useful statistic is the fraction of missing information about Q:

$$\hat{\lambda} = \frac{r + 2/(v_m + 3)}{r + 1}$$

Both statistics r and λ are helpful diagnostics for assessing how the missing data contribute to the uncertainty about Q.

When the complete-data degrees of freedom v_0 are small, and there is only a modest proportion of missing data, the computed degrees of freedom, v_m, can be much larger than v_0, which is inappropriate. For example, with $m = 5$ and $r = 10\%$, the computed degrees of freedom $v_m = 484$, which is inappropriate for data sets with complete-data degrees of freedom less than 484.

Barnard and Rubin (1999) recommend the use of adjusted degrees of freedom

$$v_m^* = \left[\frac{1}{v_m} + \frac{1}{\hat{v}_{obs}}\right]^{-1}$$

where $\hat{v}_{obs} = (1-\gamma)\, v_0(v_0+1)/(v_0+3)$ and $\gamma = (1+m^{-1})B/T$.

Note that the MI procedure uses the adjusted degrees of freedom, v_m^*, for inference.

Multiple Imputation Efficiency

The relative efficiency (RE) of using the finite m imputation estimator, rather than using an infinite number for the fully efficient imputation, in units of variance, is approximately a function of m and λ (Rubin 1987, p. 114):

$$\mathrm{RE} = \left(1 + \frac{\lambda}{m}\right)^{-1}$$

where $m$is the number of imputations and λ is the fraction of missing information.

Table 82.7 shows relative efficiencies with different values of m and λ.

Table 82.7 Relative Efficiencies

	λ				
m	**10%**	**20%**	**30%**	**50%**	**70%**
3	0.9677	0.9375	0.9091	0.8571	0.8108
5	0.9804	0.9615	0.9434	0.9091	0.8772
10	0.9901	0.9804	0.9709	0.9524	0.9346
20	0.9950	0.9901	0.9852	0.9756	0.9662

The table shows that if the fraction of missing information is modest, only a small number of imputations are needed. For example, if $\lambda = 0.3$, only three imputations are needed to have a 91% efficiency and five imputations are needed to have a 94% efficiency.

Number of Imputations

In multiple imputation, the number of imputations, m, must be specified in advance. The classic recommendation of a small m is based on a relative efficiency argument. For example, with 30% missing information, only three imputations are needed for 91% efficiency and five imputations are needed for 94% efficiency. Thus, often as few as three to five imputations are adequate in multiple imputation (Rubin 1996, p. 480). For more information about the relative efficiency of an estimator based on m imputations, see the section "Multiple Imputation Efficiency" on page 6616.

Recent studies by Graham, Olchowski, and Gilreath (2007); Bodner (2008); and Von Hippel (2009, p. 278) note that a small number of imputations that suffice for high relative efficiency might not be adequate for other inferential goals, such as confidence intervals and p-values. These studies recommend much larger numbers of imputations.

Graham, Olchowski, and Gilreath (2007) use simulations to examine the effect of m on the power of a hypothesis test, and they recommend the use of many more imputations than the classic recommendation of three to five imputations. For example, with a 1% power falloff tolerance in multiple imputation, as compared to an infinite number of imputations, multiple imputation requires 20 imputations for 30% missing information and 40 imputations for 50% missing information (Graham, Olchowski, and Gilreath 2007, p. 212).

Bodner (2008) points out that with small m there is substantial imprecision in important statistics such as p-values and widths of confidence intervals. That is, different conclusions might be drawn for the same test in separate multiple imputation runs. Bodner (2008) uses simulation to compute the minimum number of imputations that are needed for an estimated 95% confidence interval width to achieve a specified precision. For example, for a 95% confidence interval width to be within 10% of its true value 95% of the time, multiple imputation requires 24 imputations if $\lambda = 0.3$ and 59 imputations if $\lambda = 0.5$ (Bodner 2008, p. 668), where λ is the fraction of missing information. Because λ is unknown, a conservative estimate of λ is the proportion of cases with missing values (Bodner 2008, p. 670).

Von Hippel (2009, p. 278) shows that with a small number of imputations, only the point estimates are reliable. That is, the point estimates will not change much if the missing values are imputed again. For other statistics (such as standard error and p-value) to be reliable, the rule of thumb is to use the percentages of cases with missing values as the number of imputations. White, Royston, and Wood (2011, pp. 387–388) also suggest the use of this rule of thumb, at least when $\lambda \leq 0.5$, and the resulting number of imputations often provides an adequate level to reproduce the results if the missing values are imputed again.

These recent studies suggest the use of a much larger number of imputations than the classic recommendation of three to five imputations. Thus, the default number of imputations in PROC MI was changed from NIMPUTE=5 to NIMPUTE=25 starting in SAS/STAT 14.1. Alternatively, you can specify NIMPUTE=PCTMISSING to use the percentage of cases with missing values as the number of imputations.

In practice, you can verify the needed number of imputations informally by replicating sets of m imputations and checking whether the estimates are stable between sets (Horton and Lipsitz 2001, p. 246).

Imputer's Model Versus Analyst's Model

Multiple imputation inference assumes that the model you used to analyze the multiply imputed data (the analyst's model) is the same as the model used to impute missing values in multiple imputation (the imputer's model). But in practice, the two models might not be the same (Schafer 1997, p. 139).

Schafer (1997, pp. 139–143) provides comprehensive coverage of this topic, and the following example is based on his work.

Consider a trivariate data set with variables Y_1 and Y_2 fully observed, and a variable Y_3 with missing values. An imputer creates multiple imputations with the model $Y_3 = Y_1 \ Y_2$. However, the analyst can later use the simpler model $Y_3 = Y_1$. In this case, the analyst assumes more than the imputer. That is, the analyst assumes there is no relationship between variables Y_3 and Y_2.

The effect of the discrepancy between the models depends on whether the analyst's additional assumption is true. If the assumption is true, the imputer's model still applies. The inferences derived from multiple imputations will still be valid, although they might be somewhat conservative because they reflect the additional uncertainty of estimating the relationship between Y_3 and Y_2.

On the other hand, suppose that the analyst models $Y_3 = Y_1$, and there is a relationship between variables Y_3 and Y_2. Then the model $Y_3 = Y_1$ will be biased and is inappropriate. Appropriate results can be generated only from appropriate analyst models.

Another type of discrepancy occurs when the imputer assumes more than the analyst. For example, suppose that an imputer creates multiple imputations with the model $Y_3 = Y_1$, but the analyst later fits a model $Y_3 = Y_1 \ Y_2$. When the assumption is true, the imputer's model is a correct model and the inferences still hold.

On the other hand, suppose that there is a relationship between Y_3 and Y_2. Imputations that are created under the incorrect assumption that there is no relationship between Y_3 and Y_2 make the analyst's estimate of the relationship biased toward zero. Multiple imputations that are created under an incorrect model can lead to incorrect conclusions.

Thus, generally you should include as many variables as you can when doing multiple imputation. The precision that you lose from including unimportant predictors is usually a relatively small price to pay for the general validity of analyses of the resultant multiply imputed data set (Rubin 1996). But at the same time, you need to keep the model building and fitting feasible (Barnard and Meng 1999, pp. 19–20).

To produce high-quality imputations for a particular variable, the imputation model should also include variables that are potentially related to the imputed variable and variables that are potentially related to the missingness of the imputed variable (Schafer 1997, p. 143).

Similar suggestions were also given by Van Buuren, Boshuizen, and Knook (1999, p. 687). They recommend that the imputation model include three sets of covariates: variables in the analyst's model, variables associated with the missingness of the imputed variable, and variables correlated with the imputed variable. They also recommend the removal of the covariates not in the analyst's model if they have too many missing values for observations that contain missing imputed variables. As a general rule, Van Buuren (2019, p. 168) suggests using no more than 15–25 variables. See the section "Missing Data Statistics" on page 6619 for several statistics that you can use to select a suitable set of variables.

Note that it is good practice to include a description of the imputer's model with the multiply imputed data set (Rubin 1996, p. 479). That way, the analysts have information about the variables that are involved in the imputation and which relationships among the variables have been implicitly set to zero.

Missing Data Statistics

Van Buuren (2019) suggests a number of missing data statistics to represent the potential connectedness of observed to missing variables in a data set. He classifies these statistics as either inbound or outbound. Inbound statistics represent information flow from other, observed variables to the missing variable of interest. Outbound statistics represent the information flow from a variable under consideration to missing variables in the data set.

Generally speaking, when you select variables as imputation targets, higher values of inbound statistics are preferable. Higher values are also better for outbound statistics as potential predictors of missing variables. Variables that have low values of inbound and outbound statistics that are not of direct analytical interest can be candidates for removal from the imputation model.

Assume that you have a data set Y that contains p variables and n observations. Let r_{ij} be an indicator variable whose value is 0 for missing observations y_{ij} and 1 otherwise.

The inbound statistic $I(j,k)$ that is given by

$$I(j,k) = \frac{\sum_i^n (1 - r_{ij}) r_{ik}}{\sum_i^n (1 - r_{ij})}$$

reflects the proportion of usable cases—that is, the number of variable pairs (Y_j, Y_k) such that Y_j is missing while Y_k is observed, divided by the total number of observations for which Y_j is missing. This proportion can be interpreted as reflecting the potential information flow from variable Y_k that can be used to impute values for Y_j. Higher values are preferred.

The average inbound statistic I_j for the *j*th variable results from averaging over the columns of the data set:

$$I_{j_{\text{avg}}} = \frac{1}{p-1} \sum_{k \neq j}^{p} I(j,k)$$

Summing the numerator of the formula for $I(j,k)$ over the available variables and dividing by the total number of observations gives the influx coefficient:

$$I_j = \frac{\sum_k^p \sum_i^n (1 - r_{ij}) r_{ik}}{\sum_k^p \sum_i^k r_{ik}}$$

which corresponds to the total number of variable pairs for which Y_j is missing while Y_k is observed, divided by the count of observed values.

The outbound statistic $O(j,k)$ is defined as

$$O(j,k) = \frac{\sum_i^n r_{ij} (1 - r_{ik})}{\sum_i^n r_{ij}}$$

which represents the number of variable pairs (Y_j, Y_k) for which Y_j is observed while Y_k is missing, divided by the total number of observed Y_j. This proportion can be interpreted as reflecting the potential information flow from variable Y_j that can be used to impute variable Y_k.

Averaging over the available columns, you get the average outbound statistic:

$$O_{j_{\text{avg}}} = \frac{1}{p-1} \sum_{k \neq j}^{p} O(j,k)$$

Summing the numerator of the formula for $O(j,k)$ over the available variables and dividing by the total number of missing values gives the outflux coefficient:

$$O_j = \frac{\sum_k^p \sum_i^n r_{ij}(1 - r_{ik})}{\sum_k^p \sum_i^n (1 - r_{ij})}$$

which corresponds to the total number of variable pairs for which Y_j is observed while Y_k is missing, divided by the count of missing values.

An additional outbound statistic that is considered by White and Carlin (2010) is the fraction of incomplete cases among those for which Y_j is observed (FICO):

$$f_j = \frac{\sum_i^n r_{ij} q_i}{\sum_i^n r_{ij}}$$

where q_i is an indicator whose value is 1 if the ith row of observations is incomplete and 0 otherwise. Higher values of FICO are associated with increased efficiency of multiple imputation over complete-case analysis.

When you specify the FLUX option in the PROC MI statement, the procedure produces a table that displays the influx and outflux statistics, the average inbound and outbound statistics, and the percentage of observed cases. In addition to the table, a scatter plot of the influx and outflux statistics is displayed. When you construct an imputation model, variables closer to the diagonal are preferable. Variables close to the lower left corner—that is, those that have both low influx and low outflux—are candidates for removal from the imputation model unless they are of direct analytical interest.

Parameter Simulation versus Multiple Imputation

As an alternative to multiple imputation, parameter simulation can also be used to analyze the data for many incomplete-data problems. Although the MI procedure does not offer parameter simulation, the trade-offs between the two methods (Schafer 1997, pp. 89–90, 135–136) are examined in this section.

The parameter simulation method simulates random values of parameters from the observed-data posterior distribution and makes simple inferences about these parameters (Schafer 1997, p. 89). When a set of well-defined population parameters $\boldsymbol{\theta}$ are of interest, parameter simulation can be used to directly examine and summarize simulated values of $\boldsymbol{\theta}$. This usually requires a large number of iterations, and involves calculating appropriate summaries of the resulting dependent sample of the iterates of the $\boldsymbol{\theta}$. If only a small set of parameters are involved, parameter simulation is suitable (Schafer 1997).

Multiple imputation requires only a small number of imputations. Generating and storing a few imputations can be more efficient than generating and storing a large number of iterations for parameter simulation.

When fractions of missing information are low, methods that average over simulated values of the missing data, as in multiple imputation, can be much more efficient than methods that average over simulated values of $\boldsymbol{\theta}$ as in parameter simulation (Schafer 1997).

Sensitivity Analysis for the MAR Assumption

Multiple imputation usually assumes that the data are missing at random (MAR). Suppose the data set contains variables $Y = (Y_{obs}, Y_{mis})$, where Y_{obs} are fully observed variables and Y_{mis} is a variable that contains missing observations. Also suppose $\mathbf{R}$ is a response indicator whose element is 0 or 1, depending on whether Y_{mis} is missing or observed. Then the MAR assumption is that the probability that a Y_{mis} observation is missing can depend on Y_{obs} but not on Y_{mis}. That is,

$$\mathrm{pr}(\,\mathbf{R} \mid Y_{obs}, Y_{mis}\,) = \mathrm{pr}(\,\mathbf{R} \mid Y_{obs}\,)$$

The MAR assumption cannot be verified, because the missing values are not observed. In clinical trials, for a study that assumes MAR, the sensitivity of inferences to departures from the MAR assumption should be examined, as recommended by the National Research Council (2010, p. 111):

> Recommendation 15: Sensitivity analysis should be part of the primary reporting of findings from clinical trials. Examining sensitivity to the assumptions about the missing data mechanism should be a mandatory component of reporting.

If it is plausible that the missing data are not MAR, you can perform sensitivity analysis under the missing not at random (MNAR) assumption. You can generate inferences for various scenarios under MNAR and then examine the results. If the results under MNAR differ from the results under MAR, then the conclusion under MAR is in question.

Based on the factorization of the joint distribution $\mathrm{pr}(\,Y, \mathbf{R}\,)$, there are two common strategies for sensitivity analysis under MNAR: the pattern-mixture model approach and the selection model approach. The pattern-mixture model approach is implemented in the MI procedure because it is natural and straightforward.

Pattern-Mixture Model Approach

In the pattern-mixture model approach (Little 1993; Molenberghs and Kenward 2007, pp. 30, 34–37; National Research Council 2010, pp. 88–89), the joint distribution is factorized as

$$\mathrm{pr}(\,Y, \mathbf{R}\,) = \mathrm{pr}(\,Y \mid \mathbf{R}) \;\mathrm{pr}(\,\mathbf{R}\,)$$

This allows for different distributions for missing values and for observed values. For example,

$$\mathrm{pr}(\,Y, \mathbf{R}\,) = \mathrm{pr}(\,Y \mid \mathbf{R}) \;\mathrm{pr}(\mathbf{R}\,) = \mathrm{pr}(\,Y \mid \mathbf{R} = \mathbf{1}) \;\mathrm{pr}(\mathbf{R} = \mathbf{1}\,) + \mathrm{pr}(\,Y \mid \mathbf{R} = \mathbf{0}) \;\mathrm{pr}(\mathbf{R} = \mathbf{0}\,)$$

which is a mixture of distributions for two different patterns. Here, the "pattern" refers to a group of observations that have the same distribution; the term is not used in the same sense as "missing data pattern."

In the pattern-mixture model approach, the joint distribution is factored as

$$\mathrm{pr}(Y_{obs}, Y_{mis}, \mathbf{R}) = \mathrm{pr}(Y_{mis} \mid Y_{obs}, \mathbf{R}) \;\mathrm{pr}(Y_{obs}, \mathbf{R})$$

and under the MNAR assumption,

$$\mathrm{pr}(\,Y_{mis} \mid Y_{obs}, \mathbf{R} = 0\,) \neq \mathrm{pr}(\,Y_{mis} \mid Y_{obs}, \mathbf{R} = 1\,)$$

It is straightforward to create imputations by using pattern-mixture models. The next three sections provide details for this approach.

Selection Model Approach

In the selection model approach (Rubin 1987, p. 207; Little and Rubin 2002, pp. 313–314; Molenberghs and Kenward 2007, p. 30), the joint distribution is factorized as

$$\mathrm{pr}(Y, \mathbf{R}) = \mathrm{pr}(\mathbf{R} \mid Y)\ \mathrm{pr}(Y)$$

where $Y = (Y_{obs}, Y_{mis})$, $\mathrm{pr}(Y)$ is the marginal distribution of Y, and $\mathrm{pr}(\mathbf{R} \mid Y)$ is the conditional distribution of the missing mechanism R given Y. The term "selection" comes from the specification of R that selects individuals to be observed in the conditional distribution $\mathrm{pr}(\mathbf{R} \mid Y)$. Both distributions, $\mathrm{pr}(Y)$ and $\mathrm{pr}(\mathbf{R} \mid Y)$, must be specified for the analysis. The MI procedure does not provide this approach.

Multiple Imputation with Pattern-Mixture Models

For $Y = (Y_{obs}, Y_{mis})$, the joint distribution of Y and $\mathbf{R}$ can be expressed as

$$\mathrm{pr}(Y_{obs}, Y_{mis}, \mathbf{R}) = \mathrm{pr}(Y_{mis} \mid Y_{obs}, \mathbf{R})\ \mathrm{pr}(Y_{obs}, \mathbf{R})$$

Under the MAR assumption,

$$\mathrm{pr}(\mathbf{R} \mid Y_{obs}, Y_{mis}) = \mathrm{pr}(\mathbf{R} \mid Y_{obs})$$

and it can be shown that

$$\mathrm{pr}(Y_{mis} \mid Y_{obs}, \mathbf{R}) = \mathrm{pr}(Y_{mis} \mid Y_{obs})$$

That is,

$$\mathrm{pr}(Y_{mis} \mid Y_{obs}, \mathbf{R} = 0) = \mathrm{pr}(Y_{mis} \mid Y_{obs}, \mathbf{R} = 1)$$

Thus the posterior distribution $\mathrm{pr}(Y_{mis} \mid Y_{obs}, \mathbf{R} = 1)$ can be used to create imputations for missing data.

Under the MNAR assumption, each pattern that has missing Y_{mis} values might have a different distribution than the corresponding pattern that has observed Y_{mis} values. For example, in a clinical trial, suppose the data set contains an indicator variable Trt, with a value of 1 for patients in the treatment group and a value of 0 for patients in the placebo control group, a variable Y_0 for the baseline efficacy score, and a variable Y for the efficacy score at a follow-up visit. Assume that Trt and Y_0 are fully observed and Y is not fully observed. The indicator variable $\mathbf{R}$ is 0 or 1, depending on whether Y is missing or observed.

Then, under the MAR assumption,

$$\mathrm{pr}(Y \mid \mathsf{Trt} = 0, Y_0, \mathbf{R} = 0) = \mathrm{pr}(Y \mid \mathsf{Trt} = 0, Y_0, \mathbf{R} = 1)$$

and

$$\mathrm{pr}(Y \mid \mathsf{Trt} = 1, Y_0, \mathbf{R} = 0) = \mathrm{pr}(Y \mid \mathsf{Trt} = 1, Y_0, \mathbf{R} = 1)$$

Under the MNAR assumption,

$$\mathrm{pr}(Y \mid \mathsf{Trt} = 0, Y_0, \mathbf{R} = 0) \neq \mathrm{pr}(Y \mid \mathsf{Trt} = 0, Y_0, \mathbf{R} = 1)$$

or

$$\mathrm{pr}(\,Y \mid \mathsf{Trt} = 1, Y_0, \mathbf{R} = 0\,) \neq \mathrm{pr}(\,Y \mid \mathsf{Trt} = 1, Y_0, \mathbf{R} = 1\,)$$

Thus, under MNAR, missing Y values in the treatment group can be imputed from a posterior distribution generated from observations in the control group, and the imputed values can be adjusted to reflect the systematic difference between the distributions for missing and observed Y values.

Multiple imputation inference, under either the MAR or MNAR assumption, involves three distinct phases:

1. The missing data are filled in *m* times to generate *m* complete data sets.
2. The *m* complete data sets are analyzed by using other SAS procedures.
3. The results from the *m* complete data sets are combined for the inference.

For sensitivity analysis, you must specify the MNAR statement together with a MONOTONE statement or an FCS statement. When you specify a MONOTONE statement, the variables that have missing values are imputed sequentially in each imputation. When you specify an FCS statement, each imputation is carried out in two phases: the preliminary filled-in phase, followed by the imputation phase. The variables that have missing values are imputed sequentially for a number of burn-in iterations before the imputation.

Under the MNAR assumption, the following steps are used to impute missing values for each imputed variable in each imputation (when you specify a MONOTONE statement) or in each iteration (when you specify an FCS statement):

1. For each imputed variable, a conditional model, such as a regression model for continuous variables, is fitted using either all applicable observations or a specified subset of observations.
2. A new model is simulated from the posterior predictive distribution of the fitted model.
3. Missing values of the variable are imputed based on the new model, and the imputed values for a specified subset of observations can be adjusted using specified shift and scale parameters.

The next two sections provide details for specifying subsets of observations for imputation models and for adjusting imputed values.

Specifying Sets of Observations for Imputation in Pattern-Mixture Models

By default, all available observations are used to derive the imputation model. By using the MODEL option in the MNAR statement, you can specify the set of observations that are used to derive the model. You specify a classification variable (*obs-variable*) by using the option MODELOBS= (*obs-variable*= '*level1*' < '*level2*' **...** >). The MI procedure uses the group of observations for which *obs-variable* equals one of the specified classification levels.

When you use the MNAR statement together with a MONOTONE statement, you can also use the MODELOBS=CCMV and MODELOBS=NCMV options to specify the set of observations for deriving the imputation model. For a monotone missing pattern data set that contains the variables $Y_1, Y_2, \ldots, Y_p$ (in that order), there are at most p groups of observations such that the same number of variables is observed for

observations in each group. The complete-case missing values (CCMV) method (Little 1993; Molenberghs and Kenward 2007, p. 35) uses the group of observations for which all variables are observed (complete cases) to derive the imputation model. The neighboring-case missing values (NCMV) method (Molenberghs and Kenward 2007, pp. 35–36) uses only the neighboring group of observations (that is, for Y_j, the group of observations with Y_j observed and Y_{j+1} missing).

In PROC MI, the option MODELOBS=CCMV(K=*k*) uses the *k* groups of observations together with as many observed variables as possible to derive the imputation model. For instance, specifying K=1 (which is the default) uses observations from the group that has all variables observed (complete cases). Specifying K=2 uses observations from the two groups that have the most variables observed (the group of observations that has all variables observed and the group of observations that has Y_{p-1} observed but Y_p missing).

For an imputed variable Y_j, the option MODELOBS=NCMV(K=*k*) uses the *k* closest groups of observations that have observed Y_j but have as few observed variables as possible to derive the imputation model. For instance, specifying K=1 (which is the default) uses the group of observations that has Y_j observed but Y_{j+1} missing (neighboring cases). Specifying K=2 uses observations from the two closest groups that have Y_j observed (the group of observations that has Y_j observed but Y_{j+1} missing, and the group of observations that has Y_{j+1} observed and Y_{j+2} missing).

When you use the MNAR statement together with an FCS statement, the MODEL option applies only after the preliminary filled-in phase in each of the imputations.

For an illustration of the MODEL option, see Example 82.15.

Adjusting Imputed Values in Pattern-Mixture Models

It is straightforward to specify pattern-mixture models under the MNAR assumption. When you impute continuous variables by using the regression and predictive mean matching methods, you can adjust the imputed values directly (Carpenter and Kenward 2013, pp. 237–239; Van Buuren 2019, pp. 96–97). When you impute classification variables by using the logistic regression method, you can adjust the imputed classification levels by modifying the log odds ratios for the classification levels (Carpenter and Kenward 2013, pp. 240–241; Van Buuren 2019, pp. 96–97). By modifying the log odds ratios, you modify the predicted probabilities for the classification levels.

For each imputed variable, you can use the ADJUST option to do the following:

- specify a subset of observations for which imputed values are adjusted. Otherwise, all imputed values are adjusted.
- adjust imputed continuous variable values by using the SHIFT=, SCALE=, and SIGMA= options. These options add a constant, multiply by a constant factor, and add a simulated value to the imputed values, respectively.
- adjust imputed classification variable levels by adjusting predicted probabilities for the classification levels by using the SHIFT= and SIGMA= options. These options add a constant and add a simulated constant value, respectively, to the log odds ratios for the classification levels.

In addition, you can provide the shift and scale parameters for each imputation by using a PARMS= data set.

When you use the MNAR statement together with a MONOTONE statement, the variables are imputed sequentially. For each imputed variable, the values can be adjusted using the ADJUST option, and these adjusted values are used to impute values for subsequent variables.

When you use the MNAR statement together with an FCS statement, there are two phases in each imputation: the preliminary filled-in phase, followed by the imputation phase. For each imputed variable, the values can be adjusted using the ADJUST option in the imputation phase in each of the imputations. These adjusted values are used to impute values for other variables in the imputation phase.

For illustrations of adjusting imputed continuous values, adjusting log odds ratio for imputed classification levels, and adjusting imputed continuous values by using parameters that are stored in an input data set, see Example 82.16, Example 82.17, and Example 82.18, respectively.

Specifying the Imputed Values to Be Adjusted

By default, all available imputed values are adjusted. You can specify a subset of imputed values to be adjusted by using the ADJUSTOBS= suboption in the ADJUST option.

You can specify a classification variable to identify the subset of imputed values to be adjusted by using the ADJUSTOBS= (*obs-variable*= '*level1*' **<** '*level2*' **...** **>**) option. This subset consists of the imputed values in the set of observations for which *obs-variable* equals one of the specified levels.

Adjusting Imputed Continuous Variables

For an imputed continuous variable, the SCALE=*c* option specifies the scale parameter, $c > 0$, for imputed values; the SHIFT=δ option specifies the shift parameter, δ, for imputed values; and the SIGMA=σ option specifies the sigma parameter, $\sigma > 0$, for imputed values.

When the sigma parameter is not specified, the adjusted value for each imputed value y is given by

$$y^* = c\,y + \delta$$

where c is the scale parameter and δ is the shift parameter.

When you specify a sigma parameter σ, a simulated shift parameter is generated from the normal distribution that has mean δ and standard deviation σ in each imputation

$$\delta^* \sim N\left(\delta,\ \sigma^2\right)$$

The adjusted value is then given by

$$y^* = c\,y + \delta^*$$

Adjusting Imputed Classification Variables

For an imputed classification variable, you can specify adjustment parameters for the response level. The SHIFT=δ option specifies the shift parameter δ, the SIGMA=σ option specifies the sigma parameter $\sigma > 0$, and the EVENT='*level*' option identifies the response level.

When the sigma parameter is not specified, the shift parameter δ is used in all imputations. When you specify a sigma parameter σ, a simulated shift parameter is generated from the normal distribution that has mean δ and standard deviation σ for each imputation

$$\delta^* \sim N\left(\delta,\ \sigma^2\right)$$

The next three sections provide details for adjusting imputed binary, ordinal, and nominal response variables.

Adjusting Imputed Binary Response Variables

For an imputed binary classification variable Y, the shift parameter δ is applied to the logit function values for the corresponding response level.

For instance, if Y has binary responses 1 and 2, a simulated logit model

$$\text{logit}(\,\text{pr}(Y = 1 \mid \mathbf{x})\,) = \alpha + \mathbf{x}'\boldsymbol{\beta}$$

is used to impute the missing response values. For a detailed description of this simulated logit model, see the section "Binary Response Logistic Regression" on page 6594.

For an observation that has missing Y and covariates $\mathbf{x_0}$, the predicted probabilities that Y=1 and Y=2 are then given by

$$\text{pr}(Y = 1) = \frac{e^{\alpha + \mathbf{x_0}'\boldsymbol{\beta}}}{e^{\alpha + \mathbf{x_0}'\boldsymbol{\beta}} + 1} = \frac{e^{d_1}}{e^{d_1} + e^{d_2}}$$

$$\text{pr}(Y = 2) = \frac{1}{e^{\alpha + \mathbf{x_0}'\boldsymbol{\beta}} + 1} = \frac{e^{d_2}}{e^{d_1} + e^{d_2}}$$

where $d_1 = \alpha + \mathbf{x_0}'\boldsymbol{\beta}$ and $d_2 = 0$.

When you provide the shift parameters δ_1 for the response Y=1 and δ_2 for the response Y=2, the predicted probabilities are

$$\text{pr}(Y = 1) = \frac{e^{d_1^*}}{e^{d_1^*} + e^{d_2^*}}$$

$$\text{pr}(Y = 2) = \frac{e^{d_2^*}}{e^{d_1^*} + e^{d_2^*}}$$

where $d_1^* = d_1 + \delta_1$ and $d_2^* = d_2 + \delta_2 = \delta_2$.

For example, the following statement specifies the shift parameters $\delta_1 = 0.8$ and $\delta_2 = 1.6$:

```
mnar adjust( y(event='1') / shift=0.8)
     adjust( y(event='2') / shift=1.6);
```

The statement

```
mnar adjust( y(event='1') / shift=0.8 sigma=0.2);
```

simulates a shift parameter δ_1 from

$$\delta \sim N\left(0.8,\ 0.2^2\right)$$

in each imputation. Because an adjustment is not specified for Y=2, the corresponding shift parameter is $\delta_2 = 0$.

Adjusting Imputed Ordinal Response Variables

For an imputed ordinal classification variable Y, the shift parameter δ is applied to the cumulative logit function values for the corresponding response level.

For instance, if Y has ordinal responses 1, 2, . . . , K, a simulated cumulative logit model that has covariates $\mathbf{x}$,

$$\text{logit}(\,\text{pr}(Y \le k \mid \mathbf{x})\,) = \alpha_k + \mathbf{x}'\boldsymbol{\beta}$$

is used to impute the missing response values, where k = 1, 2, . . . , K–1. For a detailed description of this model, see the section "Ordinal Response Logistic Regression" on page 6595.

For an observation that has missing Y and covariates $\mathbf{x_0}$, the predicted cumulative probability for $Y \le j$, j = 1, 2, . . . , K–1, is then given by

$$\text{pr}(Y \le j) = \frac{e^{\alpha_j + \mathbf{x_0}'\boldsymbol{\beta}}}{e^{\alpha_j + \mathbf{x_0}'\boldsymbol{\beta}} + 1} = \frac{e^{d_j}}{e^{d_j} + e^{d_K}}$$

where $d_j = \alpha_j + \mathbf{x_0}'\boldsymbol{\beta}$ and $d_K = 0$.

The predicted probabilities for $Y = k$ are

$$\text{pr}(Y = k) = \begin{cases} \frac{e^{d_1}}{e^{d_1}+e^{d_K}} & \text{if } k = 1 \\ \frac{e^{d_k}}{e^{d_k}+e^{d_K}} - \frac{e^{d_{(k-1)}}}{e^{d_{(k-1)}}+e^{d_K}} & \text{if } 1 < k < K \\ \frac{e^{d_K}}{e^{d_{(K-1)}}+e^{d_K}} & \text{if } k = K \end{cases}$$

For an ordinal logistic regression method that has two response levels, the section "Adjusting Imputed Binary Response Variables" on page 6626 explains how the predicted probabilities are adjusted using shift parameters.

For an ordinal logistic regression method that has more than two response levels, only one classification level can be adjusted. When you provide the shift parameter δ for the response level $Y = k$, the predicted probability for $Y = k$ is then given by

$$\text{pr}(Y = k) = \begin{cases} \frac{e^{d_1^*}}{e^{d_1^*}+e^{d_K}} & \text{if } k = 1 \\ \frac{e^{d_k^*}}{e^{d_k^*}+e^{d_K}} - \frac{e^{d_{(k-1)}}}{e^{d_{(k-1)}}+e^{d_K}} & \text{if } 1 < k < K \\ \frac{e^{d_K^*}}{e^{d_{(K-1)}}+e^{d_K^*}} & \text{if } k = K \end{cases}$$

where $d_k^* = d_k + \delta$.

The predicted probabilities for the remaining $Y \ne k$ are then adjusted proportionally. When the shift parameter δ is less than 0, the value d_k^* can be less than d_{k-1} for $1 < k < K$. In this case, $\text{pr}(Y = k)$ is set to 0.

Adjusting Imputed Nominal Response Variables

For an imputed nominal classification variable Y, the shift parameter δ is applied to the generalized logit model function values for the corresponding response level.

For instance, if
VariableY has nominal responses 1, 2, . . . , K, a simulated generalized logit model

$$\log\left(\frac{\mathrm{pr}(Y = k \mid \mathbf{x})}{\mathrm{pr}(Y = K \mid \mathbf{x})}\right) = \alpha_k + \mathbf{x}'\boldsymbol{\beta}_k$$

is used to impute the missing response values, where k=1, 2, . . . , K–1. For a detailed description of this model, see the section "Nominal Response Logistic Regression" on page 6596.

For an observation with missing Y and covariates $\mathbf{x_0}$, the predicted probability for Y = j, $j < K$, is then given by

$$\mathrm{pr}(Y = j) = \frac{e^{\alpha_j + \mathbf{x_0}'\boldsymbol{\beta}_j}}{\sum_{k=1}^{K-1} e^{\alpha_k + \mathbf{x_0}'\boldsymbol{\beta}_k} + 1} = \frac{e^{d_j}}{\sum_{k=1}^{K} e^{d_k}}$$

and

$$\mathrm{pr}(Y = K) = \frac{1}{\sum_{k=1}^{K-1} e^{\alpha_k + \mathbf{x_0}'\boldsymbol{\beta}_k} + 1} = \frac{e^{d_K}}{\sum_{k=1}^{K} e^{d_k}}$$

where $d_k = \alpha_k + \mathbf{x}'\boldsymbol{\beta}_k$ for $k < K$ and $d_K = 0$.

When you use the shift parameters δ_k for $Y = k, k = 1, 2, \ldots, K$, the predicted probabilities are

$$\mathrm{pr}(Y = j) = \frac{e^{d_j^*}}{\sum_{k=1}^{K} e^{d_k^*}}$$

where $d_k^* = d_k + \delta_k$.

Summary of Issues in Multiple Imputation

This section summarizes issues that are encountered in applications of the MI procedure.

The MAR Assumption

Multiple imputation usually assumes that the data are missing at random (MAR). But the assumption cannot be verified, because the missing values are not observed. Although the MAR assumption becomes more plausible as more variables are included in the imputation model (Schafer 1997, pp. 27–28; Van Buuren, Boshuizen, and Knook 1999, p. 687), it is important to examine the sensitivity of inferences to departures from the MAR assumption.

Number of Imputations

Based on estimator efficiency, only a small number of imputations are needed for data that have modest missing information (Rubin 1987, p. 114). However, based on other inferential aspects, such as confidence intervals and p-values, a larger number of imputations are needed for reliable results (Allison 2012; Van Buuren 2019, pp. 58–60). You can informally verify the number of imputations by replicating sets of m imputations and checking whether the estimates are stable (Horton and Lipsitz 2001, p. 246).

Imputation Model

In general, you should include as many variables as you can in the imputation model (Rubin 1996). However, it is also important to limit the variables to a reasonable number, as discussed by Barnard and Meng (1999, pp. 19–20). For the imputation of a particular variable, the model should include variables in the complete-data model, variables that are correlated with the imputed variable, and variables that are associated with the missingness of the imputed variable (Schafer 1997, p. 143; Van Buuren, Boshuizen, and Knook 1999, p.687). Diagnostic statistics for missing data such as inbound and outbound statistics can aid in the variable selection process (Van Buuren 2019, pp. 108–111).

Multivariate Normality Assumption

Although the regression and MCMC methods assume multivariate normality, inferences based on multiple imputation can be robust to departures from the multivariate normality if the amount of missing information is not large (Schafer 1997, pp. 147–148).

You can use variable transformations to make the normality assumption more tenable. Variables are transformed before the imputation process and then back-transformed to create imputed values.

Monotone Regression Method

With the multivariate normality assumption, either the regression method or the predictive mean matching method can be used to impute continuous variables in data sets with monotone missing patterns.

The predictive mean matching method ensures that imputed values are plausible and might be more appropriate than the regression method if the normality assumption is violated (Horton and Lipsitz 2001, p. 246).

Monotone Propensity Score Method

The propensity score method can also be used to impute continuous variables in data sets with monotone missing patterns.

The propensity score method does not use correlations among variables and is not appropriate for analyses involving relationship among variables, such as a regression analysis (Schafer 1999, p. 11). It can also produce badly biased estimates of regression coefficients when data on predictor variables are missing (Allison 2000).

MCMC Monotone-Data Imputation

The MCMC method is used to impute continuous variables in data sets with arbitrary missing patterns, assuming a multivariate normal distribution for the data. It can also be used to impute just enough missing values to make the imputed data sets have a monotone missing pattern. Then, a more flexible monotone imputation method can be used for the remaining missing values.

Checking Convergence in MCMC

In an MCMC method, parameters are drawn after the MCMC is run long enough to converge to its stationary distribution. In practice, however, it is not simple to verify the convergence of the process, especially for a large number of parameters.

You can check for convergence by examining the observed-data likelihood ratio statistic and worst linear function of the parameters in each iteration. You can also check for convergence by examining a plot of autocorrelation function, as well as a trace plot of parameters (Schafer 1997, p. 120).

EM Estimates

The EM algorithm can be used to compute the MLE of the mean vector and covariance matrix of the data with missing values, assuming a multivariate normal distribution for the data. However, the covariance matrix associated with the estimate of the mean vector cannot be derived from the EM algorithm.

In the MI procedure, you can use the EM algorithm to compute the posterior mode, which provides a good starting value for the MCMC method (Schafer 1997, p. 169).

Sensitivity Analysis

Multiple imputation inference often assumes that the data are missing at random (MAR). But the MAR assumption cannot be verified, because the missing values are not observed. For a study that assumes MAR, the sensitivity of inferences to departures from the MAR assumption should be examined.

In the MI procedure, you can use the MNAR statement to imputes missing values for scenarios under the MNAR assumption. You can then generate inferences and examine the results. If the results under MNAR differ from the results under MAR, then the conclusion under MAR is in question.

Plot Options Superseded by ODS Graphics

You can select one of two types of graphics in PROC MI: ODS and traditional. When ODS Graphics is enabled, you can use the PLOTS= option in the MCMC statement to create plots by using ODS Graphics. When ODS Graphics is not enabled, you can use the TIMEPLOT and ACFPLOT options in the MCMC statement to create traditional graphics. ODS Graphics is the preferred method of creating graphs, superseding traditional graphics. For more information about ODS Graphics options see the PLOTS= option in the section "MCMC Statement" on page 6570. This section describes the options that are available in the MCMC statement for traditional graphics.

Table 82.8 summarizes the options available for traditional graphics in the MCMC statement.

Table 82.8 Traditional Graphics Options in the MCMC Statement

Option	Description
TIMEPLOT	Displays trace plots
ACFPLOT	Displays autocorrelation plots
GOUT=	Specifies the graphics catalog name for saving traditional graphics output

The following options are available in the MCMC statement for traditional graphics (in alphabetical order).

ACFPLOT < (*options*< / *display-options* >) >
displays the traditional autocorrelation function plots of parameters from iterations. The ACFPLOT option is applicable only if ODS Graphics is not enabled.

The available options are as follows.

COV < (< *variables* > < *variable1*variable2* > < . . . *variable1*variable2* >) >
displays plots of variances for variables in the list and covariances for pairs of variables in the list. When the option COV is specified without variables, variances for all variables and covariances for all pairs of variables are used.

MEAN < (*variables*) >
displays plots of means for variables in the list. When the option MEAN is specified without variables, all variables are used.

WLF
displays the plot for the worst linear function.

When the ACFPLOT is specified without the preceding options, the procedure displays plots of means for all variables that are used.

The display options provide additional information for the autocorrelation function plots. By default, the MI procedure uses the star (*) as the plot symbol to display the points with a height of one (percentage screen unit) in the plot, a solid line to display the reference line of zero autocorrelation, vertical line segments to connect autocorrelations to the reference line, and a pair of dashed lines to display approximately 95% confidence limits for the autocorrelations.

You can use the SYMBOL=, CSYMBOL=, and HSYMBOL= options to change the shape, color, and height of the plot symbol, respectively, and the CNEEDLES= and WNEEDLES= options to change the color and width of the needles, respectively. You can also use the LREF=, CREF=, and WREF= options to change the line type, color, and width of the reference line, respectively. Similarly, you can use the LCONF=, CCONF=, and WCONF= options to change the line type, color, and width of the confidence limits, respectively.

By default, the plot title “Autocorrelation Plot” is displayed in a autocorrelation function plot. You can request another title by using the TITLE= option within the ACFPLOT option. When another title is also specified in a TITLE statement, this title is displayed as the main title and the plot title is displayed as a subtitle in the plot.

You can use options in the GOPTIONS statement to change the color and height of the title. See the chapter “The SAS/GRAPH Statements” in *SAS/GRAPH: Reference* for a description of title options.

The available display options are as follows:

CCONF=*color*
specifies the color of the displayed confidence limits. The default is CCONF=BLACK.

CFRAME=*color*
specifies the color for filling the area enclosed by the axes and the frame. By default, this area is not filled.

CNEEDLES=*color*
specifies the color of the vertical line segments (needles) that connect autocorrelations to the reference line. The default is CNEEDLES=BLACK.

CREF=*color*
specifies the color of the displayed reference line. The default is CREF=BLACK.

CSYMBOL=*color*
specifies the color of the displayed data points. The default is CSYMBOL=BLACK.

HSYMBOL=*number*
specifies the height of data points in percentage screen units. The default is HSYMBOL=1.

LCONF=*linetype*
specifies the line type for the displayed confidence limits. The default is LCONF=1, a solid line.

LOG
requests that the logarithmic transformations of parameters be used to compute the autocorrelations; it is generally used for the variances of variables. When a parameter has values less than or equal to zero, the corresponding plot is not created.

LREF=*linetype*
specifies the line type for the displayed reference line. The default is LREF=3, a dashed line.

NAME=*'string'*
specifies a descriptive name, up to eight characters, that appears in the name field of the PROC GREPLAY master menu. The default is NAME='MI'.

NLAG=*number*
specifies the maximum lag of the series. The default is NLAG=20. The autocorrelations at each lag are displayed in the graph.

SYMBOL=*value*
specifies the symbol for data points in percentage screen units. The default is SYMBOL=STAR.

TITLE=*'string'*
specifies the title to be displayed in the autocorrelation function plots. The default is TITLE='Autocorrelation Plot'.

WCONF=*number*
specifies the width of the displayed confidence limits in percentage screen units. If you specify the WCONF=0 option, the confidence limits are not displayed. The default is WCONF=1.

WNEEDLES=*number*
specifies the width of the displayed needles that connect autocorrelations to the reference line, in percentage screen units. If you specify the WNEEDLES=0 option, the needles are not displayed. The default is WNEEDLES=1.

WREF=*number*
specifies the width of the displayed reference line in percentage screen units. If you specify the WREF=0 option, the reference line is not displayed. The default is WREF=1.

For example, the following statement requests autocorrelation function plots for the means and variances of the variable y1, respectively:

```
acfplot( mean( y1) cov(y1) /log);
```

Logarithmic transformations of both the means and variances are used in the plots. For a detailed description of the autocorrelation function plot, see the section "Autocorrelation Function Plot" on page 6611; see also Schafer (1997, pp. 120–126) and the *SAS/ETS User's Guide*.

GOUT=*graphics-catalog*
specifies the graphics catalog for saving graphics output from PROC MI. The default is WORK.GSEG. For more information, see "The GREPLAY Procedure" in *SAS/GRAPH: Reference*.

TIMEPLOT < *(options* < */ display-options* > *)* >
displays the traditional trace (time series) plots of parameters from iterations. The TIMEPLOT option is applicable only if ODS Graphics is not enabled.

The available options are as follows:

COV <(< *variables* > < *variable1*variable2* > . . . **) >**

displays plots of variances for variables in the list and covariances for pairs of variables in the list. When the option COV is specified without variables, variances for all variables and covariances for all pairs of variables are used.

MEAN < **(***variables***)** >
displays plots of means for variables in the list. When the option MEAN is specified without variables, all variables are used.

WLF
displays the plot of the worst linear function.

When the TIMEPLOT is specified without the preceding options, the procedure displays plots of means for all variables that are used.

The display options provide additional information for the trace plots. By default, the MI procedure uses solid line segments to connect data points in a trace plot. You can use the CCONNECT=, LCONNECT=, and WCONNECT= options to change the color, line type, and width of the line segments, respectively. When WCONNECT=0 is specified, the data points are not connected, and the procedure uses the plus sign (+) as the plot symbol to display the points with a height of one (percentage screen unit) in a trace plot. You can use the SYMBOL=, CSYMBOL=, and HSYMBOL= options to change the shape, color, and height of the plot symbol, respectively.

By default, the plot title "Trace Plot" is displayed in a trace plot. You can request another title by using the TITLE= option in the TIMEPLOT option. When another title is also specified in a TITLE statement, this title is displayed as the main title and the plot title is displayed as a subtitle in the plot.

You can use options in the GOPTIONS statement to change the color and height of the title. See the chapter "The SAS/GRAPH Statements" in *SAS/GRAPH: Reference* for an illustration of title options.

The available display options are as follows:

CCONNECT=*color*
specifies the color of the line segments that connect data points in the trace plots. The default is CCONNECT=BLACK.

CFRAME=*color*
specifies the color for filling the area enclosed by the axes and the frame. By default, this area is not filled.

CSYMBOL=*color*
specifies the color of the data points to be displayed in the trace plots. The default is CSYMBOL=BLACK.

HSYMBOL=*number*
specifies the height of data points in percentage screen units. The default is HSYMBOL=1.

LCONNECT=*linetype*
specifies the line type for the line segments that connect data points in the trace plots. The default is LCONNECT=1, a solid line.

LOG
requests that the logarithmic transformations of parameters be used; it is generally used for the variances of variables. When a parameter value is less than or equal to zero, the value is not displayed in the corresponding plot.

NAME=*'string'*
specifies a descriptive name, up to eight characters, that appears in the name field of the PROC GREPLAY master menu. The default is NAME='MI'.

SYMBOL=*value*
specifies the symbol for data points in percentage screen units. The default is SYMBOL=PLUS.

TITLE=*'string'*
specifies the title to be displayed in the trace plots. The default is TITLE='Trace Plot'.

WCONNECT=*number*
specifies the width of the line segments that connect data points in the trace plots, in percentage screen units. If you specify the WCONNECT=0 option, the data points are not connected. The default is WCONNECT=1.

For a detailed description of the trace plot, see the section "Trace Plot" on page 6611 and Schafer (1997, pp. 120–126).

ODS Table Names

PROC MI assigns a name to each table it creates. You must use these names to reference tables when using the Output Delivery System (ODS). These names are listed in Table 82.9. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 82.9 ODS Tables Produced by PROC MI

ODS Table Name	Description	Statement	Option
Corr	Pairwise correlations		SIMPLE
EMEstimates	EM (MLE) estimates	EM	
EMInitEstimates	EM initial estimates	EM	
EMIterHistory	EM (MLE) iteration history	EM	ITPRINT
EMPostEstimates	EM (posterior mode) estimates	MCMC	INITIAL=EM
EMPostIterHistory	EM (posterior mode) iteration history	MCMC	INITIAL=EM (ITPRINT)
EMWLF	Worst linear function	MCMC	WLF
FCSDiscrim	Discriminant model group means	FCS	DISCRIM (/DETAILS)
FCSLogistic	Logistic model	FCS	LOGISTIC (/DETAILS)
FCSModel	FCS models	FCS	
FCSReg	Regression model	FCS	REG (/DETAILS)
FCSRegPMM	Predicted mean matching model	FCS	REGPMM (/DETAILS)
Flux	Missing Data Statistics	PROC MI	FLUX
MCMCInitEstimates	MCMC initial estimates	MCMC	DISPLAYINIT
MissPattern	Missing data patterns		
MNARModel	Observations that are used for imputation model under MNAR	MNAR	MODEL
MNARAdjust	Adjustment parameters and imputed values to be adjusted under MNAR	MNAR	ADJUST
ModelInfo	Model information		
MonoDiscrim	Discriminant model group means	MONOTONE	DISCRIM (/DETAILS)
MonoLogistic	Logistic model	MONOTONE	LOGISTIC (/DETAILS)
MonoModel	Monotone models	MONOTONE	
MonoPropensity	Propensity score model logistic function	MONOTONE	PROPENSITY (/DETAILS)
MonoReg	Regression model	MONOTONE	REG (/DETAILS)
MonoRegPMM	Predicted mean matching model	MONOTONE	REGPMM (/DETAILS)
ParameterEstimates	Parameter estimates		

Table 82.9 *continued*

ODS Table Name	Description	Statement	Option
Transform	Variable transformations	TRANSFORM	
Univariate	Univariate statistics		SIMPLE
VarianceInfo	Between, within, and total variances		

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC MI assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. To request these graphs, ODS Graphics must be enabled and you must specify the options indicated in Table 82.10.

Table 82.10 Graphs Produced by PROC MI

ODS Graph Name	Description	Statement	Option
ACFPlot	ACF plot	MCMC	PLOTS=ACF
FluxPlot	Flux plot	PROC MI	FLUX
TracePlot	Trace plot	FCS	PLOTS=TRACE
		MCMC	PLOTS=TRACE

Examples: MI Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The Fish data that are described in the STEPDISC procedure are measurements of 159 fish of seven species caught in Finland's Lake Laengelmaevesi. For each fish, the length, height, and width are measured. Three different length measurements are recorded: from the nose of the fish to the beginning of its tail (Length1), from the nose to the notch of its tail (Length2), and from the nose to the end of its tail (Length3). See Chapter 117, "The STEPDISC Procedure," for more information.

The Fish1 data set is constructed from the Fish data set and contains only one species of the fish and the three length measurements. Some values have been set to missing, and the resulting data set has a monotone missing pattern in the variables Length1, Length2, and Length3. The Fish1 data set is used in Example 82.2 with the propensity score method and in Example 82.3 with the regression method.

The Fish2 data set is also constructed from the Fish data set and contains two species of fish. Some values have been set to missing, and the resulting data set has a monotone missing pattern in the variables Length, Width, and Species. The Fish2 data set is used in Example 82.4 with the logistic regression method and in Example 82.5 with the discriminant function method. Note that some values of the variable Species have also been altered in the data set.

The Fish3 data set is similar to the data set Fish2 except some additional values have been set to missing and the resulting data set has an arbitrary missing pattern. The Fish3 data set is used in Example 82.7 and in Example 82.8.

The Fitness1 data set created in the section "Getting Started: MI Procedure" on page 6555 is used in other examples.

The following statements create the Fish1 data set:

```
*----------------------------Fish1 Data----------------------------*
| The data set contains one species of the fish (Bream) and         |
| three measurements: Length1, Length2, Length3.                    |
| Some values have been set to missing, and the resulting data set  |
| has a monotone missing pattern in the variables                   |
| Length1, Length2, and Length3.                                    |
*-------------------------------------------------------------------*;
data Fish1;
   title 'Fish Measurement Data';
   input Length1 Length2 Length3 @@;
   datalines;
23.2 25.4 30.0    24.0 26.3 31.2    23.9 26.5 31.1
26.3 29.0 33.5    26.5 29.0   .     26.8 29.7 34.7
26.8   .    .     27.6 30.0 35.0    27.6 30.0 35.1
28.5 30.7 36.2    28.4 31.0 36.2    28.7   .    .
29.1 31.5   .     29.5 32.0 37.3    29.4 32.0 37.2
29.4 32.0 37.2    30.4 33.0 38.3    30.4 33.0 38.5
30.9 33.5 38.6    31.0 33.5 38.7    31.3 34.0 39.5
31.4 34.0 39.2    31.5 34.5   .     31.8 35.0 40.6
31.9 35.0 40.5    31.8 35.0 40.9    32.0 35.0 40.6
32.7 36.0 41.5    32.8 36.0 41.6    33.5 37.0 42.6
```

```
35.0 38.5 44.1    35.0 38.5 44.0    36.2 39.5 45.3
37.4 41.0 45.9    38.0 41.0 46.5
;
```

The Fish2 data set contains two of the seven species in the Fish data set. For each of the two species (Bream and Pike), the length from the nose of the fish to the end of its tail and the width of each fish are measured.

The following statements create the Fish2 data set:

```
*----------------------------Fish2 Data----------------------------*
| The data set contains two species of the fish (Parkki and Perch)   |
| and two measurements: Length and Width.                            |
| Some values have been set to missing, and the resulting data set   |
| has a monotone missing pattern in the variables                    |
| Length, Width, and Species.                                        |
*--------------------------------------------------------------------*;
data Fish2;
   title 'Fish Measurement Data';
   input Species $ Length Width @@;
   datalines;
Parkki  16.5  2.3265    Parkki  17.4  2.3142    .      19.8   .
Parkki  21.3  2.9181    Parkki  22.4  3.2928    .      23.2  3.2944
Parkki  23.2  3.4104    Parkki  24.1  3.1571    .      25.8  3.6636
Parkki  28.0  4.1440    Parkki  29.0  4.2340    Perch   8.8  1.4080
.       14.7  1.9992    Perch   16.0  2.4320    Perch  17.2  2.6316
Perch   18.5  2.9415    Perch   19.2  3.3216    .      19.4   .
Perch   20.2  3.0502    Perch   20.8  3.0368    Perch  21.0  2.7720
Perch   22.5  3.5550    Perch   22.5  3.3075    .      22.5   .
Perch   22.8  3.5340    .       23.5   .        Perch  23.5  3.5250
Perch   23.5  3.5250    Perch   23.5  3.5250    Perch  23.5  3.9950
.       24.0   .        Perch   24.0  3.6240    Perch  24.2  3.6300
Perch   24.5  3.6260    Perch   25.0  3.7250    .      25.5  3.7230
Perch   25.5  3.8250    Perch   26.2  4.1658    Perch  26.5  3.6835
.       27.0  4.2390    Perch   28.0  4.1440    Perch  28.7  5.1373
.       28.9  4.3350    .       28.9   .        .      28.9  4.5662
Perch   29.4  4.2042    Perch   30.1  4.6354    Perch  31.6  4.7716
Perch   34.0  6.0180    .       36.5  6.3875    .      37.3  7.7957
.       39.0   .        .       38.3   .        Perch  39.4  6.2646
Perch   39.3  6.3666    Perch   41.4  7.4934    Perch  41.4  6.0030
Perch   41.3  7.3514    .       42.3   .        Perch  42.5  7.2250
Perch   42.4  7.4624    Perch   42.5  6.6300    Perch  44.6  6.8684
Perch   45.2  7.2772    Perch   45.5  7.4165    Perch  46.0  8.1420
Perch   46.6  7.5958
;
```

The following statements create the Fish3 data set:

```
*----------------------------Fish3 Data----------------------------*
| The data set contains three species of the fish                    |
| (Parkki, Perch, and Roach) and two measurements: Length and Width. |
| Some values have been set to missing, and the resulting data set   |
| has an arbitrary missing pattern in the variables                  |
| Length, Width, and Species.                                        |
*--------------------------------------------------------------------*;
data Fish3;
```

```
   title 'Fish Measurement Data';
   input Species $ Length Width @@;
   datalines;
Roach   16.2  2.2680    Roach   20.3  2.8217    Roach   21.2   .
Roach     .   3.1746    Roach   22.2  3.5742    Roach   22.8  3.3516
Roach   23.1  3.3957    .       23.7   .        Roach   24.7  3.7544
Roach   24.3  3.5478    Roach   25.3   .        Roach   25.0  3.3250
Roach   25.0  3.8000    Roach   27.2  3.8352    Roach   26.7  3.6312
Roach   26.8  4.1272    Roach   27.9  3.9060    Roach   29.2  4.4968
Roach   30.6  4.7736    Roach   35.0  5.3550    Parkki  16.5  2.3265
Parkki  17.4   .        Parkki  19.8  2.6730    Parkki  21.3  2.9181
Parkki  22.4  3.2928    Parkki  23.2  3.2944    Parkki  23.2  3.4104
Parkki  24.1  3.1571    .         .   3.6636    Parkki  28.0  4.1440
Parkki  29.0  4.2340    Perch    8.8  1.4080    .       14.7  1.9992
Perch   16.0  2.4320    Perch   17.2  2.6316    Perch   18.5  2.9415
Perch   19.2  3.3216    .       19.4  3.1234    Perch   20.2   .
Perch   20.8  3.0368    Perch   21.0  2.7720    Perch   22.5  3.5550
Perch   22.5  3.3075    Perch   22.5  3.6675    Perch     .   3.5340
Perch   23.5  3.4075    Perch   23.5  3.5250    Perch   23.5  3.5250
.       23.5  3.5250    Perch   23.5  3.9950    Perch   24.0  3.6240
Perch   24.0  3.6240    Perch   24.2  3.6300    Perch   24.5  3.6260
Perch   25.0  3.7250    Perch     .   3.7230    Perch   25.5  3.8250
Perch     .   4.1658    Perch   26.5  3.6835    .       27.0  4.2390
Perch     .   4.1440    Perch   28.7  5.1373    .       28.9  4.3350
Perch   28.9  4.3350    Perch   28.9  4.5662    Perch   29.4  4.2042
Perch   30.1  4.6354    Perch   31.6  4.7716    Perch   34.0  6.0180
Perch   36.5  6.3875    Perch   37.3  7.7957    Perch   39.0   .
Perch   38.3  6.7408    Perch     .   6.2646    .       39.3   .
Perch   41.4  7.4934    Perch   41.4  6.0030    Perch   41.3  7.3514
Perch   42.3  7.1064    Perch   42.5  7.2250    Perch   42.4  7.4624
Perch   42.5  6.6300    Perch   44.6  6.8684    Perch   45.2  7.2772
Perch   45.5  7.4165    Perch   46.0  8.1420    .       46.6  7.5958
;
```

Example 82.1: EM Algorithm for MLE

(View the complete code for this example (miex1.sas) in the example repository.)

This example uses the EM algorithm to compute the maximum likelihood estimates for parameters of multivariate normally distributed data with missing values. The following statements invoke the MI procedure and request the EM algorithm to compute the MLE for $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ of a multivariate normal distribution from the input data set Fitness1:

```
proc mi data=Fitness1 seed=1518971 simple nimpute=0;
   em itprint outem=outem;
   var Oxygen RunTime RunPulse;
run;
```

Note that when you specify the NIMPUTE=0 option, the missing values are not imputed.

The “Model Information” table in Output 82.1.1 describes the method and options used in the procedure if a positive number is specified in the NIMPUTE= option.

Output 82.1.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FITNESS1
Method	MCMC
Multiple Imputation Chain	Single Chain
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	0
Number of Burn-in Iterations	200
Number of Iterations	100
Seed for random number generator	1518971

The "Missing Data Patterns" table in Output 82.1.2 lists distinct missing data patterns with corresponding frequencies and percentages. Here, a value of "X" means that the variable is observed in the corresponding group and a value of "." means that the variable is missing. The table also displays group-specific variable means.

Output 82.1.2 Missing Data Patterns

Missing Data Patterns								
						Group Means		
Group	**Oxygen**	**RunTime**	**RunPulse**	**Freq**	**Percent**	**Oxygen**	**RunTime**	**RunPulse**
1	X	X	X	21	67.74	46.353810	10.809524	171.666667
2	X	X	.	4	12.90	47.109500	10.137500	.
3	X	.	.	3	9.68	52.461667	.	.
4	.	X	X	1	3.23	.	11.950000	176.000000
5	.	X	.	2	6.45	.	9.885000	.

With the SIMPLE option, the procedure displays simple descriptive univariate statistics for available cases in the "Univariate Statistics" table in Output 82.1.3 and correlations from pairwise available cases in the "Pairwise Correlations" table in Output 82.1.4.

Output 82.1.3 Univariate Statistics

Univariate Statistics							
						Missing Values	
Variable	**N**	**Mean**	**Std Dev**	**Minimum**	**Maximum**	**Count**	**Percent**
Oxygen	28	47.11618	5.41305	37.38800	60.05500	3	9.68
RunTime	28	10.68821	1.37988	8.63000	14.03000	3	9.68
RunPulse	22	171.86364	10.14324	148.00000	186.00000	9	29.03

Output 82.1.4 Pairwise Correlations

Pairwise Correlations			
	Oxygen	**RunTime**	**RunPulse**
Oxygen	1.000000000	-0.849118562	-0.343961742
RunTime	-0.849118562	1.000000000	0.247258191
RunPulse	-0.343961742	0.247258191	1.000000000

When you use the EM statement, the MI procedure displays the initial parameter estimates for the EM algorithm in the "Initial Parameter Estimates for EM" table in Output 82.1.5.

Output 82.1.5 Initial Parameter Estimates for EM

Initial Parameter Estimates for EM				
TYPE	**_NAME_**	**Oxygen**	**RunTime**	**RunPulse**
MEAN		47.116179	10.688214	171.863636
COV	Oxygen	29.301078	0	0
COV	RunTime	0	1.904067	0
COV	RunPulse	0	0	102.885281

When you use the ITPRINT option in the EM statement, the "EM (MLE) Iteration History" table in Output 82.1.6 displays the iteration history for the EM algorithm.

Output 82.1.6 EM (MLE) Iteration History

EM (MLE) Iteration History				
Iteration	**-2 Log L**	**Oxygen**	**RunTime**	**RunPulse**
0	289.544782	47.116179	10.688214	171.863636
1	263.549489	47.116179	10.688214	171.863636
2	255.851312	47.139089	10.603506	171.538203
3	254.616428	47.122353	10.571685	171.426790
4	254.494971	47.111080	10.560585	171.398296
5	254.483973	47.106523	10.556768	171.389208
6	254.482920	47.104899	10.555485	171.385257
7	254.482813	47.104348	10.555062	171.383345
8	254.482801	47.104165	10.554923	171.382424
9	254.482800	47.104105	10.554878	171.381992
10	254.482800	47.104086	10.554864	171.381796
11	254.482800	47.104079	10.554859	171.381708
12	254.482800	47.104077	10.554858	171.381669

The "EM (MLE) Parameter Estimates" table in Output 82.1.7 displays the maximum likelihood estimates for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ of a multivariate normal distribution from the data set Fitness1.

Output 82.1.7 EM (MLE) Parameter Estimates

EM (MLE) Parameter Estimates				
TYPE	_NAME_	Oxygen	RunTime	RunPulse
MEAN		47.104077	10.554858	171.381669
COV	Oxygen	27.797931	-6.457975	-18.031298
COV	RunTime	-6.457975	2.015514	3.516287
COV	RunPulse	-18.031298	3.516287	97.766857

You can also output the EM (MLE) parameter estimates to an output data set with the OUTEM= option. The following statements list the observations in the output data set Outem:

```
proc print data=outem;
   title 'EM Estimates';
run;
```

The output data set Outem in Output 82.1.8 is a TYPE=COV data set. The observation with _TYPE_='MEAN' contains the MLE for the parameter $\boldsymbol{\mu}$, and the observations with _TYPE_='COV' contain the MLE for the parameter $\boldsymbol{\Sigma}$ of a multivariate normal distribution from the data set Fitness1.

Output 82.1.8 EM Estimates

EM Estimates

Obs	_TYPE_	_NAME_	Oxygen	RunTime	RunPulse
1	MEAN		47.1041	10.5549	171.382
2	COV	Oxygen	27.7979	-6.4580	-18.031
3	COV	RunTime	-6.4580	2.0155	3.516
4	COV	RunPulse	-18.0313	3.5163	97.767

Example 82.2: Monotone Propensity Score Method

(View the complete code for this example (miex2.sas) in the example repository.)

This example uses the propensity score method to impute missing values for variables in a data set with a monotone missing pattern. The following statements invoke the MI procedure and request the propensity score method. The resulting data set is named Outex2.

```
proc mi data=Fish1 seed=899603 out=outex2;
   monotone propensity;
   var Length1 Length2 Length3;
run;
```

Note that the VAR statement is required and the data set must have a monotone missing pattern with variables as ordered in the VAR statement.

The "Model Information" table in Output 82.2.1 describes the method and options used in the multiple imputation process. By default, 25 imputations are created for the missing data.

Output 82.2.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FISH1
Method	Monotone
Number of Imputations	25
Seed for random number generator	899603

When monotone methods are used in the imputation, MONOTONE is displayed as the method. The "Monotone Model Specification" table in Output 82.2.2 displays the detailed model specification. By default, the observations are sorted into five groups based on their propensity scores.

Output 82.2.2 Monotone Model Specification

Monotone Model Specification	
Method	**Imputed Variables**
Propensity(Groups= 5)	Length2 Length3

Without covariates specified for imputed variables Length2 and Length3, the variable Length1 is used as the covariate for Length2, and the variables Length1 and Length2 are used as covariates for Length3.

The "Missing Data Patterns" table in Output 82.2.3 lists distinct missing data patterns with corresponding frequencies and percentages. Here, values of "X" and "." indicate that the variable is observed or missing, respectively, in the corresponding group. The table confirms a monotone missing pattern for these three variables.

Output 82.2.3 Missing Data Patterns

Missing Data Patterns								
						Group Means		
Group	**Length1**	**Length2**	**Length3**	**Freq**	**Percent**	**Length1**	**Length2**	**Length3**
1	X	X	X	30	85.71	30.603333	33.436667	38.720000
2	X	X	.	3	8.57	29.033333	31.666667	.
3	X	.	.	2	5.71	27.750000	.	.

For the imputation process, first, missing values of Length2 in group 3 are imputed using observed values of Length1. Then the missing values of Length3 in group 2 are imputed using observed values of Length1 and Length2. And finally, the missing values of Length3 in group 3 are imputed using observed values of Length1 and imputed values of Length2.

After the completion of m imputations, the "Variance Information" table in Output 82.2.4 displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences. It also displays the degrees of freedom for the total variance. The relative increase in variance due to missingness, the fraction of missing information, and the relative efficiency for each variable are also displayed. A detailed description of these statistics is provided in the section "Combining Inferences from Multiply Imputed Data Sets" on page 6615.

Output 82.2.4 Variance Information

Variance Information (25 Imputations)							
	Variance						
Variable	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Length2	0.003228	0.457211	0.460568	31.925	0.007343	0.007294	0.999708
Length3	0.037617	0.542147	0.581269	29.829	0.072161	0.067656	0.997301

The "Parameter Estimates" table in Output 82.2.5 displays the estimated mean and standard error of the mean for each variable. The inferences are based on the *t* distributions. For each variable, the table also displays a 95% mean confidence interval and a *t* statistic with the associated *p*-value for the hypothesis that the population mean is equal to the value specified in the MU0= option, which is 0 by default.

Output 82.2.5 Parameter Estimates

Parameter Estimates (25 Imputations)										
Variable	Mean	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Length2	33.040343	0.678652	31.65785	34.42284	31.925	32.957143	33.122857	0	48.69	<.0001
Length3	38.325143	0.762410	36.76772	39.88257	29.829	37.808571	38.614286	0	50.27	<.0001

The following statements list the first 10 observations of the data set Outex2, as shown in Output 82.2.6. The missing values are imputed from observed values with similar propensity scores.

```
proc print data=outex2(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.2.6 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Length1	Length2	Length3
1	1	23.2	25.4	30.0
2	1	24.0	26.3	31.2
3	1	23.9	26.5	31.1
4	1	26.3	29.0	33.5
5	1	26.5	29.0	38.6
6	1	26.8	29.7	34.7
7	1	26.8	29.0	35.0
8	1	27.6	30.0	35.0
9	1	27.6	30.0	35.1
10	1	28.5	30.7	36.2

Example 82.3: Monotone Regression Method

(View the complete code for this example (miex3.sas) in the example repository.)

This example uses the regression method to impute missing values for all variables in a data set with a monotone missing pattern. The following statements invoke the MI procedure and request the regression method for the variable Length2 and the predictive mean matching method for variable Length3. The resulting data set is named Outex3.

```
proc mi data=Fish1 round=.1  mu0= 0 35 45
        seed=13951639 nimpute=8 out=outex3;
   monotone reg(Length2/ details)
            regpmm(Length3= Length1 Length2 Length1*Length2/ details);
   var Length1 Length2 Length3;
run;
```

The ROUND= option is used to round the imputed values to the same precision as observed values. The values specified with the ROUND= option are matched with the variables Length1, Length2, and Length3 in the order listed in the VAR statement. The MU0= option requests *t* tests for the hypotheses that the population means corresponding to the variables in the VAR statement are Length2=35 and Length3=45.

The "Missing Data Patterns" table lists distinct missing data patterns with corresponding frequencies and percentages. It is identical to the table in Output 82.2.3 in Example 82.2.

The "Monotone Model Specification" table in Output 82.3.1 displays the model specification.

Output 82.3.1 Monotone Model Specification

The MI Procedure

Monotone Model Specification

Method	Imputed Variables
Regression	Length2
Regression-PMM(K= 5)	Length3

When you use the DETAILS option, the parameters estimated from the observed data and the parameters used in each imputation are displayed in Output 82.3.2 and Output 82.3.3.

Output 82.3.2 Regression Model

Regression Models for Monotone Method

Imputed Variable	Effect	Obs-Data	Imputation 1	2	3	4	5	6	7	8
Length2	Intercept	-0.04249	-0.049184	-0.055470	-0.051346	-0.064193	-0.030719	-0.030694	-0.050964	-0.017976
Length2	Length1	0.98587	1.001934	0.995275	0.992294	0.983122	0.995883	0.989193	0.968480	0.977476

Output 82.3.3 Regression Predicted Mean Matching Model

Regression Models for Monotone Predicted Mean Matching Method										
			Imputation							
Imputed Variable	Effect	Obs Data	1	2	3	4	5	6	7	8
Length3	Intercept	-0.01304	0.004134	-0.011417	-0.034177	-0.010532	0.004685	-0.013917	0.012658	-0.020144
Length3	Length1	-0.01332	0.025320	-0.037494	0.308765	0.156606	-0.147118	0.097745	-0.254054	0.086950
Length3	Length2	0.98918	0.955510	1.025741	0.673374	0.828384	1.146440	0.860564	1.226015	0.879854
Length3	Length1*Length2	-0.02521	-0.034964	-0.022017	-0.017919	-0.029335	-0.034671	-0.023384	-0.023829	-0.026785

After the completion of eight imputations (NIMPUTE=8), the "Variance Information" table in Output 82.3.4 displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences. The relative increase in variance due to missingness, the fraction of missing information, and the relative efficiency for each variable are also displayed. These statistics are described in the section "Combining Inferences from Multiply Imputed Data Sets" on page 6615.

Output 82.3.4 Variance Information

Variance Information (8 Imputations)							
	Variance						
Variable	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Length2	0.000089650	0.439208	0.439309	32.155	0.000230	0.000230	0.999971
Length3	0.000433	0.487356	0.487842	32.13	0.000998	0.000998	0.999875

The "Parameter Estimates" table in Output 82.3.5 displays a 95% mean confidence interval and a t statistic with its associated p-value for each of the hypotheses requested with the MU0= option.

Output 82.3.5 Parameter Estimates

Parameter Estimates (8 Imputations)										
Variable	Mean	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Length2	33.106071	0.662804	31.75624	34.45590	32.155	33.088571	33.117143	35.000000	-2.86	0.0074
Length3	38.416786	0.698457	36.99430	39.83927	32.13	38.382857	38.445714	45.000000	-9.43	<.0001

The following statements list the first 10 observations of the data set Outex3 in Output 82.3.6. Note that the imputed values of Length2 are rounded to the same precision as the observed values.

```
proc print data=outex3(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.3.6 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Length1	Length2	Length3
1	1	23.2	25.4	30.0
2	1	24.0	26.3	31.2
3	1	23.9	26.5	31.1
4	1	26.3	29.0	33.5
5	1	26.5	29.0	34.7
6	1	26.8	29.7	34.7
7	1	26.8	28.8	34.7
8	1	27.6	30.0	35.0
9	1	27.6	30.0	35.1
10	1	28.5	30.7	36.2

Example 82.4: Monotone Logistic Regression Method for CLASS Variables

(View the complete code for this example (miex4.sas) in the example repository.)

This example uses logistic regression method to impute values for a binary variable in a data set with a monotone missing pattern.

In the following statements, the logistic regression method is used for the binary CLASS variable Species:

```
proc mi data=Fish2 seed=1305417 nimpute=15 out=outex4;
   class Species;
   monotone reg( Width/ details)
            logistic( Species= Length Width Length*Width/ details);
   var Length Width Species;
run;
```

The "Model Information" table in Output 82.4.1 describes the method and options used in the multiple imputation process.

Output 82.4.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FISH2
Method	Monotone
Number of Imputations	15
Seed for random number generator	1305417

The "Monotone Model Specification" table in Output 82.4.2 describes methods and imputed variables in the imputation model. The procedure uses the logistic regression method to impute the variable Species in the model. Missing values in other variables are not imputed.

Output 82.4.2 Monotone Model Specification

Monotone Model Specification	
Method	**Imputed Variables**
Regression	Width
Logistic Regression	Species

The "Missing Data Patterns" table in Output 82.4.3 lists distinct missing data patterns with corresponding frequencies and percentages. The table confirms a monotone missing pattern for these variables.

Output 82.4.3 Missing Data Patterns

Missing Data Patterns							
						Group Means	
Group	**Length**	**Width**	**Species**	**Freq**	**Percent**	**Length**	**Width**
1	X	X	X	49	73.13	28.595918	4.482518
2	X	X	.	9	13.43	27.533333	4.444844
3	X	.	.	9	13.43	28.633333	.

When you use the DETAILS option, parameters estimated from the observed data and the parameters used in each imputation are displayed in the "Logistic Models for Monotone Method" table in Output 82.4.4.

Output 82.4.4 Regression Model

Regression Models for Monotone Method											
			Imputation								
Imputed Variable	**Effect**	**Obs-Data**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Width	Intercept	0.00284	-0.029987	0.049363	-0.015273	-0.064915	0.059375	0.018049	-0.028171	-0.016050	-0.012890
Width	Length	0.96212	0.981287	0.906104	0.962814	0.978103	0.952034	0.920482	0.908541	0.962650	0.949542

Regression Models for Monotone Method							
		Imputation					
Imputed Variable	**Effect**	**10**	**11**	**12**	**13**	**14**	**15**
Width	Intercept	-0.056099	-0.013302	0.031642	0.051256	-0.032029	0.030396
Width	Length	0.962843	0.921873	0.947360	1.003013	0.950239	0.955749

Output 82.4.5 Logistic Regression Model

Logistic Models for Monotone Method										
			Imputation							
Imputed Variable	**Effect**	**Obs-Data**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Species	Intercept	-3.93577	-5.016163	-3.422209	-4.706398	-2.049090	-4.568278	-4.336259	-4.250352	-2.843154
Species	Length	10.41940	16.262215	6.082966	9.832246	4.992717	11.886805	5.789312	7.662947	5.757570
Species	Width	-14.56630	-21.856472	-8.653119	-15.534802	-7.401465	-15.621272	-12.855797	-14.816308	-8.792538
Species	Length*Width	-0.48936	-0.208880	0.795883	-0.011135	-0.461227	0.080406	-2.586760	-2.604478	-0.317211

Logistic Models for Monotone Method								
		Imputation						
Imputed Variable	**Effect**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Species	Intercept	-1.055153	-3.501466	-2.140199	-3.629155	-4.020008	-2.615227	-4.964532
Species	Length	0.572346	10.900700	10.743223	10.504352	12.346335	5.432583	11.926760
Species	Width	-1.775130	-15.547003	-12.353169	-14.555215	-16.481415	-11.694606	-18.401905
Species	Length*Width	0.027353	-0.809353	-0.060720	-0.544245	-0.507705	-2.484002	-2.094407

The following statements list the first 10 observations of the data set Outex4 in Output 82.4.6:

```
proc print data=outex4(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.4.6 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Species	Length	Width
1	1	Parkki	16.5	2.32650
2	1	Parkki	17.4	2.31420
3	1	Parkki	19.8	2.20482
4	1	Parkki	21.3	2.91810
5	1	Parkki	22.4	3.29280
6	1	Perch	23.2	3.29440
7	1	Parkki	23.2	3.41040
8	1	Parkki	24.1	3.15710
9	1	Perch	25.8	3.66360
10	1	Parkki	28.0	4.14400

Note that a missing value of the variable Species is not imputed if the corresponding covariates are missing and not imputed, as shown by observation 4 in the table.

Example 82.5: Monotone Discriminant Function Method for CLASS Variables

(View the complete code for this example (miex5.sas) in the example repository.)

This example uses discriminant monotone methods to impute values of a CLASS variable from the observed observation values in a data set with a monotone missing pattern.

The following statements impute the continuous variables Height and Width with the regression method and the classification variable Species with the discriminant function method:

```
proc mi data=Fish2 seed=7545417 out=outex5;
   class Species;
   monotone discrim( Species= Length Width/ details);
   var Length Width Species;
run;
```

The "Model Information" table in Output 82.5.1 describes the method and options used in the multiple imputation process.

Output 82.5.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FISH2
Method	Monotone
Number of Imputations	25
Seed for random number generator	7545417

The "Monotone Model Specification" table in Output 82.5.2 describes methods and imputed variables in the imputation model. The procedure uses the regression method to impute the variables Height and Width, and uses the logistic regression method to impute the variable Species in the model.

Output 82.5.2 Monotone Model Specification

Monotone Model Specification	
Method	**Imputed Variables**
Regression	Width
Discriminant Function	Species

The "Missing Data Patterns" table in Output 82.5.3 lists distinct missing data patterns with corresponding frequencies and percentages. The table confirms a monotone missing pattern for these variables.

Output 82.5.3 Missing Data Patterns

Missing Data Patterns							
						Group Means	
Group	**Length**	**Width**	**Species**	**Freq**	**Percent**	**Length**	**Width**
1	X	X	X	49	73.13	28.595918	4.482518
2	X	X	.	9	13.43	27.533333	4.444844
3	X	.	.	9	13.43	28.633333	.

When you use the DETAILS option, the parameters estimated from the observed data and the parameters used in each imputation are displayed in Output 82.5.4.

Output 82.5.4 Discriminant Model

Group Means for Monotone Discriminant Method											
			Imputation								
Species	**Variable**	**Obs-Data**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Parkki	**Length**	-0.62249	-0.917467	-0.909076	-0.146825	-0.682080	-1.187056	-0.850499	-0.697413	-0.440116	-1.018781
Parkki	**Width**	-0.71787	-0.921200	-1.036075	-0.343058	-0.844507	-1.353333	-0.881096	-0.830821	-0.478435	-1.131564
Perch	**Length**	0.13937	0.042471	0.219096	0.079881	0.080076	0.190100	0.298206	-0.061338	0.145498	0.134497
Perch	**Width**	0.14408	0.047041	0.197736	0.082832	0.118336	0.193865	0.295603	-0.051199	0.147394	0.159755

Group Means for Monotone Discriminant Method											
			Imputation								
Species	**Variable**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**
Parkki	**Length**	-0.651330	-0.706491	-0.247232	-0.852547	-0.846080	-0.294569	-0.456750	-0.286640	-0.300657	-0.574438
Parkki	**Width**	-0.785122	-0.666500	-0.390243	-0.960644	-0.803615	-0.342767	-0.603262	-0.481397	-0.516386	-0.685527
Perch	**Length**	0.073197	-0.007668	0.402478	0.271589	0.166251	0.239710	0.135714	0.107864	0.138797	0.571539
Perch	**Width**	0.090466	0.069233	0.419146	0.239319	0.182764	0.216010	0.167861	0.085148	0.127604	0.572531

Group Means for Monotone Discriminant Method							
		Imputation					
Species	**Variable**	**20**	**21**	**22**	**23**	**24**	**25**
Parkki	**Length**	-0.165544	-0.123289	-0.557388	-0.667641	-0.332219	-0.242661
Parkki	**Width**	-0.239261	-0.159966	-0.650667	-0.815406	-0.449938	-0.246102
Perch	**Length**	0.045337	-0.085437	0.306678	-0.003056	0.115950	0.304774
Perch	**Width**	0.051245	-0.013271	0.294392	0.026570	0.164894	0.317910

The following statements list the first 10 observations of the data set Outex5 in Output 82.5.5. Note that all missing values of the variables Width and Species are imputed.

```
proc print data=outex5(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.5.5 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Species	Length	Width
1	1	Parkki	16.5	2.32650
2	1	Parkki	17.4	2.31420
3	1	Perch	19.8	3.03975
4	1	Parkki	21.3	2.91810
5	1	Parkki	22.4	3.29280
6	1	Perch	23.2	3.29440
7	1	Parkki	23.2	3.41040
8	1	Parkki	24.1	3.15710
9	1	Perch	25.8	3.66360
10	1	Parkki	28.0	4.14400

Example 82.6: FCS Methods for Continuous Variables

(View the complete code for this example (miex6.sas) in the example repository.)

This example uses FCS regression methods to impute values for all continuous variables in a data set that has an arbitrary missing pattern.

The following statements invoke the MI procedure and impute missing values for the Fitness1 data set:

```
proc mi data=Fitness1 seed=1213 nimpute=pctmissing(min=5 max=20)
        mu0=50 10 180 out=outex6;
   fcs nbiter=20 reg(Oxygen/details);
   var Oxygen RunTime RunPulse;
run;
```

The NIMPUTE=PCTMISSING option uses the percentage of the incomplete cases as the number of imputations. The MIN=5 (which is the default) and MAX=20 options restrict the number of imputations to be in the range of 5 to 20. That is, 5 imputations are generated if the percentage of the incomplete cases is less than 5, and 20 imputations are generated if this percentage is greater than 20.

The FCS statement requests multivariate imputations by FCS methods, and the NBITER=20 option (which is the default) specifies the number of burn-in iterations before each imputation.

The "Model Information" table in Output 82.6.1 describes the method and options used in the multiple imputation process.

Output 82.6.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FITNESS1
Method	FCS
Number of Imputations	20
Number of Burn-in Iterations	20
Seed for random number generator	1213

The "FCS Model Specification" table in Output 82.6.2 describes methods and imputed variables in the imputation model. With the REG(OXYGEN) option in the FCS statement, the procedure uses the regression method to impute variable Oxygen. By default, the regression method is also used to impute variables RunTime and RunPulse.

Output 82.6.2 FCS Model Specification

FCS Model Specification	
Method	**Imputed Variables**
Regression	Oxygen RunTime RunPulse

The "Missing Data Patterns" table in Output 82.6.3 lists distinct missing data patterns with corresponding frequencies and percentages.

Output 82.6.3 Missing Data Patterns

Missing Data Patterns								
						Group Means		
Group	Oxygen	RunTime	RunPulse	Freq	Percent	Oxygen	RunTime	RunPulse
1	X	X	X	21	67.74	46.353810	10.809524	171.666667
2	X	X	.	4	12.90	47.109500	10.137500	.
3	X	.	.	3	9.68	52.461667	.	.
4	.	X	X	1	3.23	.	11.950000	176.000000
5	.	X	.	2	6.45	.	9.885000	.

For the NIMPUTE=PCTMISSING option, the percentage of the incomplete cases, 10/31 = 32.3%, is used as the number of imputations. But the number 33 (after rounding up) is greater than 20 (as specified in the MAX= option), so only 20 imputations are generated.

When you specify the DETAILS option in REG(OXYGEN/DETAILS), the parameters that are used in each imputation for Oxygen are displayed in Output 82.6.4.

Output 82.6.4 FCS Regression Model for Oxygen

Regression Models for FCS Method										
		Imputation								
Imputed Variable	Effect	1	2	3	4	5	6	7	8	9
Oxygen	Intercept	-0.132359	0.093555	0.078587	0.063256	-0.073869	-0.070292	-0.242377	-0.176468	-0.105706
Oxygen	RunTime	-0.908663	-0.753423	-1.125549	-0.634844	-0.569809	-0.797221	-0.498457	-0.922488	-0.790878
Oxygen	RunPulse	-0.134745	0.052640	-0.135864	-0.158692	-0.319878	-0.277367	-0.510742	-0.035716	-0.169551

Regression Models for FCS Method										
		Imputation								
Imputed Variable	Effect	10	11	12	13	14	15	16	17	18
Oxygen	Intercept	-0.100698	-0.046309	-0.267186	0.074579	-0.121640	-0.041454	-0.085027	-0.025642	-0.016753
Oxygen	RunTime	-0.748476	-0.833819	-0.745716	-0.612349	-0.747333	-0.744806	-0.864134	-0.720153	-0.615441
Oxygen	RunPulse	-0.086702	-0.158535	-0.006667	-0.175998	0.030089	-0.135503	-0.120457	-0.213634	-0.065866

Regression Models for FCS Method			
		Imputation	
Imputed Variable	Effect	19	20
Oxygen	Intercept	-0.148376	-0.165002
Oxygen	RunTime	-0.658032	-0.774503
Oxygen	RunPulse	-0.227149	-0.041462

The following statements list the first 10 observations of the data set Outex6 in Output 82.6.5. Note that all missing values of all variables are imputed.

```
proc print data=outex6(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.6.5 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Oxygen	RunTime	RunPulse
1	1	44.6090	11.3700	178.000
2	1	45.3130	10.0700	185.000
3	1	54.2970	8.6500	156.000
4	1	59.5710	10.1985	185.842
5	1	49.8740	9.2200	173.379
6	1	44.8110	11.6300	176.000
7	1	44.6299	11.9500	176.000
8	1	47.4258	10.8500	183.926
9	1	39.4420	13.0800	174.000
10	1	60.0550	8.6300	170.000

After the completion of the specified four imputations, the "Variance Information" table in Output 82.6.6 displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences. The relative increase in variance due to missingness, the fraction of missing information, and the relative efficiency for each variable are also displayed. These statistics are described in the section "Combining Inferences from Multiply Imputed Data Sets" on page 6615.

Output 82.6.6 Variance Information

Variance Information (20 Imputations)

	Variance						
Variable	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Oxygen	0.026152	0.927386	0.954846	27.339	0.029610	0.028843	0.998560
RunTime	0.002608	0.067218	0.069957	27.02	0.040736	0.039296	0.998039
RunPulse	2.430764	3.542847	6.095149	14.23	0.720410	0.429183	0.978992

The "Parameter Estimates" table in Output 82.6.7 displays a 95% mean confidence interval and a t statistic with its associated p-value for each of the hypotheses requested with the MU0= option.

Output 82.6.7 Parameter Estimates

Parameter Estimates (20 Imputations)

Variable	Mean	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Oxygen	47.080908	0.977162	45.0771	49.0847	27.339	46.758087	47.512585	50.000000	-2.99	0.0059
RunTime	10.564752	0.264493	10.0221	11.1074	27.02	10.511826	10.706529	10.000000	2.14	0.0420
RunPulse	171.412215	2.468836	166.1251	176.6993	14.23	168.633931	174.275234	180.000000	-3.48	0.0036

Example 82.7: FCS Method for CLASS Variables

(View the complete code for this example (miex7.sas) in the example repository.)

This example uses FCS methods to impute missing values in both continuous and CLASS variables in a data set that has an arbitrary missing pattern. The following statements invoke the MI procedure and impute missing values for the Fish3 data set:

```
proc mi data=Fish3 seed=1305417 nimpute=15 out=outex7;
   class Species;
   fcs nbiter=10 discrim(Species/details) reg(Width/details);
   var Species Length Width;
run;
```

The DISCRIM option uses the discriminant function method to impute the classification variable Species, and the REG option uses the regression method to impute the continuous variable Width. By default, the regression method is also used to impute the other continuous variable, Length.

The "Model Information" table in Output 82.7.1 describes the method and options that are used in the multiple imputation process.

Output 82.7.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FISH3
Method	FCS
Number of Imputations	15
Number of Burn-in Iterations	10
Seed for random number generator	1305417

The "FCS Model Specification" table in Output 82.7.2 describes the methods and imputed variables in the imputation model. The procedure uses the discriminant function method to impute the variable Species and uses the regression method to impute other variables.

Output 82.7.2 FCS Model Specification

FCS Model Specification	
Method	**Imputed Variables**
Regression	Length Width
Discriminant Function	Species

The "Missing Data Patterns" table in Output 82.7.3 lists distinct missing data patterns along with their corresponding frequencies and percentages.

Output 82.7.3 Missing Data Patterns

Missing Data Patterns							
						Group Means	
Group	Species	Length	Width	Freq	Percent	Length	Width
1	X	X	X	67	77.01	27.910448	4.361860
2	X	X	.	5	5.75	24.620000	.
3	X	.	X	6	6.90	.	4.167667
4	.	X	X	6	6.90	26.683333	4.136233
5	.	X	.	2	2.30	31.500000	.
6	.	.	X	1	1.15	.	3.663600

When you specify the DETAILS option for the variables Species and Width, the parameters that are used in each imputation for these two variables are displayed in the "Group Means for FCS Discriminant Method" table in Output 82.7.4 and in the "Regression Models for FCS Method" table in Output 82.7.5.

Output 82.7.4 FCS Discrim Model for Species

Group Means for FCS Discriminant Method											
		Imputation									
Species	Variable	1	2	3	4	5	6	7	8	9	10
Parkki	Length	-0.268298	-0.611484	-0.430752	-0.508489	-1.096890	-0.691899	-1.097177	-0.605319	-0.349771	-0.717117
Parkki	Width	-0.374514	-0.920031	-0.695627	-0.444730	-1.183297	-0.684968	-1.230322	-0.630649	-0.599680	-0.735174
Perch	Length	0.073272	0.281238	0.135766	0.105996	0.280959	-0.003145	0.356800	0.347392	0.055848	0.448266
Perch	Width	0.104187	0.345404	0.211220	0.109806	0.365960	0.116315	0.397011	0.404836	0.125563	0.524498
Roach	Length	-0.293847	-0.296757	-0.485885	0.094638	-0.028394	-0.258734	-0.391279	-0.426078	-0.616262	-0.305244
Roach	Width	-0.507327	-0.352964	-0.626142	-0.033285	-0.243456	-0.459994	-0.429801	-0.433738	-0.668304	-0.419368

Group Means for FCS Discriminant Method						
		Imputation				
Species	Variable	11	12	13	14	15
Parkki	Length	-0.636064	-0.501049	-0.384597	-0.640036	-1.019628
Parkki	Width	-0.834453	-0.520368	-0.569160	-0.741567	-1.064032
Perch	Length	0.371622	0.231022	0.150028	0.237300	0.341506
Perch	Width	0.482840	0.286552	0.260146	0.287303	0.341318
Roach	Length	-0.451065	-0.205316	-0.125811	-0.320887	-0.420115
Roach	Width	-0.466063	-0.317969	-0.238243	-0.467192	-0.520025

Output 82.7.5 FCS Regression Model for Width

Regression Models for FCS Method											
			Imputation								
Imputed Variable	Effect	Species	1	2	3	4	5	6	7	8	9
Width	Intercept		-0.080952	-0.008262	-0.040466	-0.083230	-0.047121	-0.028571	-0.055278	-0.051276	-0.095069
Width	Species	Parkki	-0.100521	-0.096675	-0.022778	-0.160418	-0.092341	0.006502	-0.022182	-0.111298	-0.101806
Width	Species	Perch	0.150457	0.119791	0.108795	0.132785	0.152929	0.056631	0.114786	0.159393	0.118097
Width	Length		0.928032	0.939600	1.039315	0.975903	0.961029	0.976143	0.977985	0.973528	0.973916

Regression Models for FCS Method							
		Imputation					
Imputed Variable	Effect	10	11	12	13	14	15
Width	Intercept	-0.075439	-0.029298	-0.033294	-0.059441	-0.117950	-0.090120
Width	Species	-0.025496	0.015175	-0.035755	-0.082546	-0.214051	-0.076865
Width	Species	0.114437	0.091675	0.068180	0.120584	0.238069	0.125656
Width	Length	1.013839	0.943677	0.956911	0.995960	0.924327	0.919800

The following statements list the first 10 observations of the data set Outex7, which are shown in Output 82.7.6:

```
proc print data=outex7(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.7.6 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Species	Length	Width
1	1	Roach	16.2000	2.26800
2	1	Roach	20.3000	2.82170
3	1	Roach	21.2000	2.40895
4	1	Roach	18.6497	3.17460
5	1	Roach	22.2000	3.57420
6	1	Roach	22.8000	3.35160
7	1	Roach	23.1000	3.39570
8	1	Perch	23.7000	3.88340
9	1	Roach	24.7000	3.75440
10	1	Roach	24.3000	3.54780

After the completion of five imputations by default, the "Variance Information" table in Output 82.7.7 displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences for continuous variables. The relative increase in variance due to missingness, the fraction of missing information, and the relative efficiency for each variable are also displayed. These statistics are described in the section "Combining Inferences from Multiply Imputed Data Sets" on page 6615.

Output 82.7.7 Variance Information

Variance Information (15 Imputations)							
	Variance						
Variable	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Length	0.004567	0.812129	0.817001	83.548	0.005999	0.005968	0.999602
Width	0.000162	0.029149	0.029321	83.555	0.005920	0.005890	0.999607

The "Parameter Estimates" table in Output 82.7.8 displays a 95% mean confidence interval and a t statistic with its associated p-value for each of the hypotheses requested with the default MU0=0 option.

Output 82.7.8 Parameter Estimates

Parameter Estimates (15 Imputations)										
Variable	Mean	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Length	27.584713	0.903881	25.78710	29.38232	83.548	27.447764	27.716689	0	30.52	<.0001
Width	4.298698	0.171235	3.95815	4.63924	83.555	4.275600	4.320615	0	25.10	<.0001

Example 82.8: FCS Method with Trace Plot

(View the complete code for this example (miex8.sas) in the example repository.)

This example uses FCS methods to impute missing values in both continuous and classification variables in a data set with an arbitrary missing pattern. The following statements use a logistic regression method to impute values of the classification variable Species:

```
ods graphics on;
proc mi data=Fish3 seed=1305417 nimpute=5 out=outex8;
   class Species;
   fcs plots=trace
       logistic(Species= Length Width Length*Width /details link=glogit);
   var Species Length Width;
run;
```

The "Model Information" table in Output 82.8.1 describes the method and options used in the multiple imputation process. By default, a regression method is used to impute missing values in each continuous variable.

Output 82.8.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FISH3
Method	FCS
Number of Imputations	5
Number of Burn-in Iterations	20
Seed for random number generator	1305417

The "FCS Model Specification" table in Output 82.8.2 describes methods and imputed variables in the imputation model. The procedure uses the logistic regression method to impute the variable Species, and the regression method to impute variables Height and Width.

Output 82.8.2 FCS Model Specification

FCS Model Specification	
Method	**Imputed Variables**
Regression	Length Width
Logistic Regression	Species

The "Missing Data Patterns" table in Output 82.8.3 lists distinct missing data patterns with corresponding frequencies and percentages.

Output 82.8.3 Missing Data Patterns

Missing Data Patterns							
						Group Means	
Group	**Species**	**Length**	**Width**	**Freq**	**Percent**	**Length**	**Width**
1	X	X	X	67	77.01	27.910448	4.361860
2	X	X	.	5	5.75	24.620000	.
3	X	.	X	6	6.90	.	4.167667
4	.	X	X	6	6.90	26.683333	4.136233
5	.	X	.	2	2.30	31.500000	.
6	.	.	X	1	1.15	.	3.663600

When you use the DETAILS keyword in the LOGISTIC option, parameters estimated from the observed data and the parameters used in each imputation are displayed in the "Logistic Models for FCS Method" table in Output 82.8.4.

Output 82.8.4 FCS Logistic Regression Model for Species

Logistic Models for FCS Method							
			Imputation				
Imputed Variable	**Effect**	**Species**	**1**	**2**	**3**	**4**	**5**
Species	Intercept	Parkki	-2.172588	-2.324226	-2.418362	-1.832884	-0.929242
Species	Intercept	Perch	1.878263	0.445966	1.585375	0.919562	1.547549
Species	Length	Parkki	6.107448	6.377145	2.447654	-1.004869	2.363073
Species	Length	Perch	-5.493897	-4.711566	-7.778194	-5.400749	-0.053788
Species	Width	Parkki	-8.624156	-6.965179	-5.718729	-0.997851	-2.978868
Species	Width	Perch	8.111323	5.608314	9.426901	5.502755	1.241239
Species	Length*Width	Parkki	-0.006404	2.138551	0.883903	0.072525	-0.152662
Species	Length*Width	Perch	1.151183	1.278025	1.117492	-0.195462	0.672738

With ODS Graphics enabled, the PLOTS=TRACE option displays trace plots of means for all continuous variables by default, as shown in Output 82.8.5 and Output 82.8.6. The dashed vertical lines indicate the imputed iterations—that is, the variable values used in the imputations. The plot shows no apparent trends for the two variables.

Output 82.8.5 Trace Plot for Length

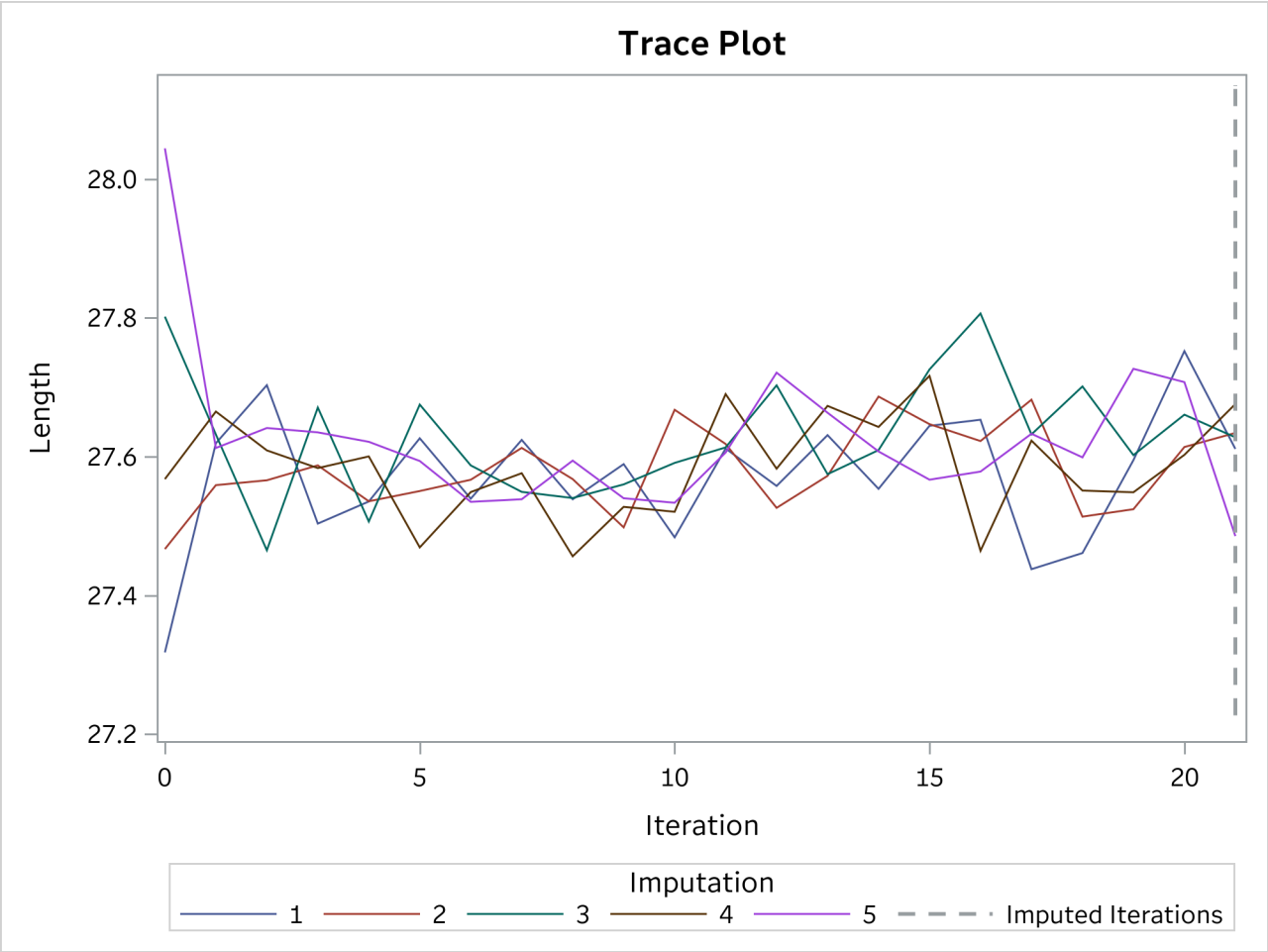

Output 82.8.6 Trace Plot for Width

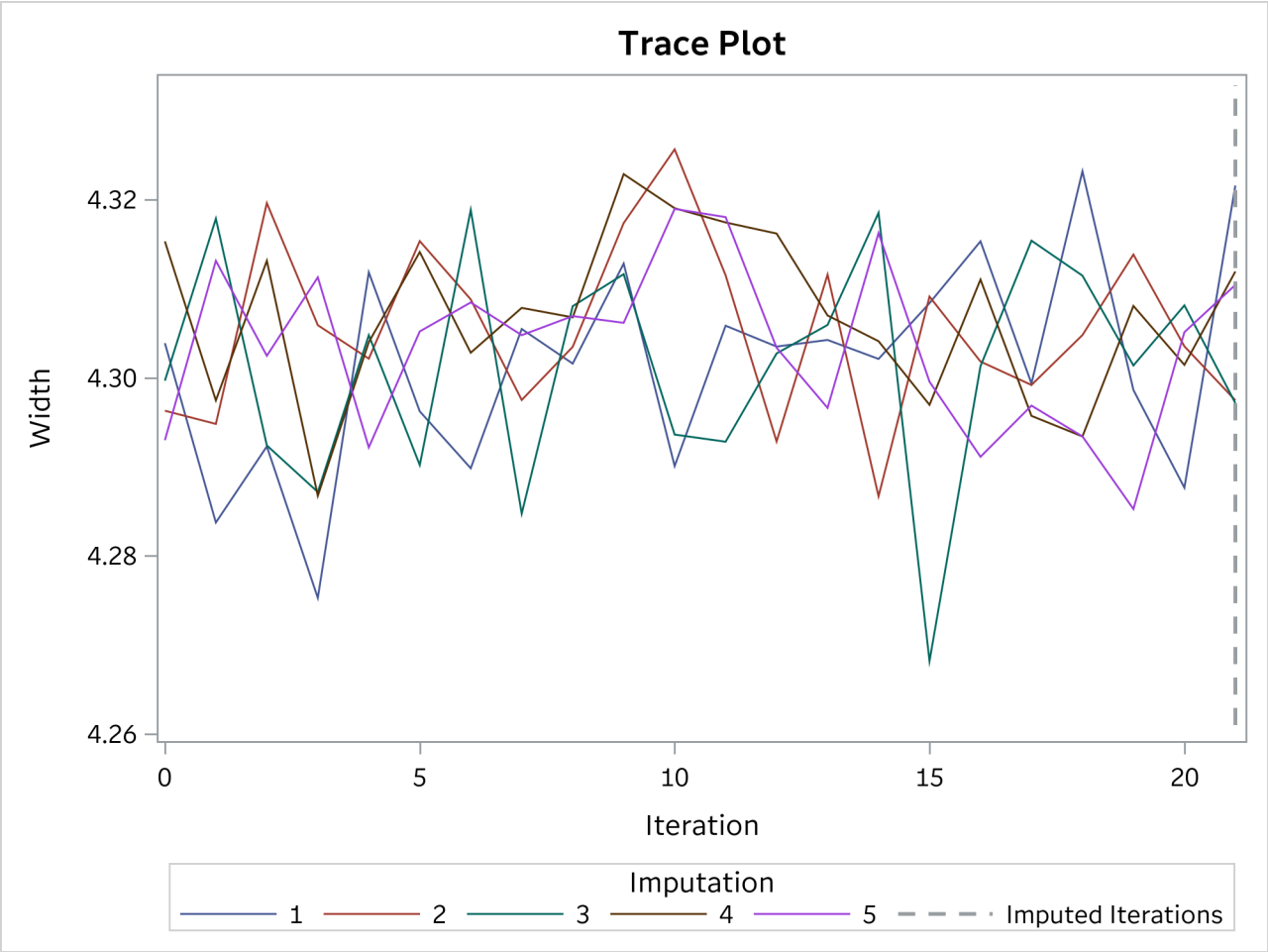

The following statements list the first 10 observations of the data set Outex8 in Output 82.8.7:

```
proc print data=outex8(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.8.7 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Species	Length	Width
1	1	Roach	16.2000	2.26800
2	1	Roach	20.3000	2.82170
3	1	Roach	21.2000	3.40493
4	1	Roach	22.4203	3.17460
5	1	Roach	22.2000	3.57420
6	1	Roach	22.8000	3.35160
7	1	Roach	23.1000	3.39570
8	1	Roach	23.7000	3.73166
9	1	Roach	24.7000	3.75440
10	1	Roach	24.3000	3.54780

After the completion of five imputations by default, the "Variance Information" table in Output 82.8.8 displays the between-imputation variance, within-imputation variance, and total variance for combining complete-data inferences for continuous variables. The relative increase in variance due to missingness, the fraction of missing information, and the relative efficiency for each variable are also displayed. These statistics are described in the section "Combining Inferences from Multiply Imputed Data Sets" on page 6615.

Output 82.8.8 Variance Information

Variance Information (5 Imputations)

	Variance						
Variable	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Length	0.005177	0.815388	0.821601	83.332	0.007620	0.007590	0.998484
Width	0.000108	0.028944	0.029074	83.656	0.004496	0.004486	0.999104

The "Parameter Estimates" table in Output 82.8.9 displays a 95% mean confidence interval and a t statistic with its associated p-value for each of the hypotheses requested with the default MU0=0 option.

Output 82.8.9 Parameter Estimates

Parameter Estimates (5 Imputations)

Variable	Mean	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Length	27.606967	0.906422	25.80424	29.40970	83.332	27.485512	27.675952	0	30.46	<.0001
Width	4.307702	0.170510	3.96860	4.64680	83.656	4.297146	4.321571	0	25.26	<.0001

Example 82.9: MCMC Method

(View the complete code for this example (miex9.sas) in the example repository.)

This example uses the MCMC method to impute missing values for a data set with an arbitrary missing pattern. The following statements invoke the MI procedure and specify the MCMC method with six imputations:

```
proc mi data=Fitness1 seed=21355417 nimpute=40 mu0=50 10 180;
   mcmc chain=multiple displayinit initial=em(itprint);
   var Oxygen RunTime RunPulse;
run;
```

The "Model Information" table in Output 82.9.1 describes the method used in the multiple imputation process. When you use the CHAIN=MULTIPLE option, the procedure uses multiple chains and completes the default 200 burn-in iterations before each imputation. The 200 burn-in iterations are used to make the iterations converge to the stationary distribution before the imputation.

Output 82.9.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FITNESS1
Method	MCMC
Multiple Imputation Chain	Multiple Chains
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	40
Number of Burn-in Iterations	200
Seed for random number generator	21355417

By default, the procedure uses a noninformative Jeffreys prior to derive the posterior mode from the EM algorithm as the starting values for the MCMC method.

The "Missing Data Patterns" table in Output 82.9.2 lists distinct missing data patterns with corresponding statistics.

Output 82.9.2 Missing Data Patterns

Missing Data Patterns								
						Group Means		
Group	**Oxygen**	**RunTime**	**RunPulse**	**Freq**	**Percent**	**Oxygen**	**RunTime**	**RunPulse**
1	X	X	X	21	67.74	46.353810	10.809524	171.666667
2	X	X	.	4	12.90	47.109500	10.137500	.
3	X	.	.	3	9.68	52.461667	.	.
4	.	X	X	1	3.23	.	11.950000	176.000000
5	.	X	.	2	6.45	.	9.885000	.

When you use the ITPRINT option within the INITIAL=EM option, the procedure displays the "EM (Posterior Mode) Iteration History" table in Output 82.9.3.

Output 82.9.3 EM (Posterior Mode) Iteration History

EM (Posterior Mode) Iteration History					
Iteration	-2 Log L	-2 Log Posterior	Oxygen	RunTime	RunPulse
0	254.482800	282.909549	47.104077	10.554858	171.381669
1	255.081168	282.051584	47.104077	10.554857	171.381652
2	255.271408	282.017488	47.104077	10.554857	171.381644
3	255.318622	282.015372	47.104002	10.554523	171.381842
4	255.330259	282.015232	47.103861	10.554388	171.382053
5	255.333161	282.015222	47.103797	10.554341	171.382150
6	255.333896	282.015222	47.103774	10.554325	171.382185
7	255.334085	282.015222	47.103766	10.554320	171.382196

When you use the DISPLAYINIT option in the MCMC statement, the "Initial Parameter Estimates for MCMC" table in Output 82.9.4 displays the starting mean and covariance estimates used in the MCMC method. The same starting estimates are used in the MCMC method for multiple chains because the EM algorithm is applied to the same data set in each chain. You can explicitly specify different initial estimates for different imputations, or you can use the bootstrap method to generate different parameter estimates from the EM algorithm for the MCMC method.

Output 82.9.4 Initial Parameter Estimates

Initial Parameter Estimates for MCMC				
TYPE	_NAME_	Oxygen	RunTime	RunPulse
MEAN		47.103766	10.554320	171.382196
COV	Oxygen	24.549967	-5.726112	-15.926036
COV	RunTime	-5.726112	1.781407	3.124798
COV	RunPulse	-15.926036	3.124798	83.164045

Output 82.9.5 and Output 82.9.6 display variance information and parameter estimates, respectively, from the multiple imputation.

Output 82.9.5 Variance Information

Variance Information (40 Imputations)							
	Variance						
Variable	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Oxygen	0.020887	0.917452	0.938861	27.529	0.023336	0.022830	0.999430
RunTime	0.001897	0.066931	0.068875	27.371	0.029045	0.028264	0.999294
RunPulse	2.240072	3.674188	5.970262	16.273	0.624920	0.389201	0.990364

Output 82.9.6 Parameter Estimates

Parameter Estimates (40 Imputations)										
Variable	Mean	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
Oxygen	47.106918	0.968949	45.1206	49.0933	27.529	46.817748	47.429633	50.000000	-2.99	0.0059
RunTime	10.558597	0.262440	10.0205	11.0967	27.371	10.467541	10.659412	10.000000	2.13	0.0424
RunPulse	171.150201	2.443412	165.9775	176.3229	16.273	167.120956	174.569040	180.000000	-3.62	0.0022

Example 82.10: Producing Monotone Missingness with MCMC

(View the complete code for this example (miex10.sas) in the example repository.)

This example uses the MCMC method to impute just enough missing values for a data set with an arbitrary missing pattern so that each imputed data set has a monotone missing pattern based on the order of variables in the VAR statement.

The following statements invoke the MI procedure and specify the IMPUTE=MONOTONE option to create the imputed data set with a monotone missing pattern. You must specify a VAR statement to provide the order of variables in order for the imputed data to achieve a monotone missing pattern.

```
proc mi data=Fitness1 seed=17655417 out=outex10;
   mcmc impute=monotone;
   var Oxygen RunTime RunPulse;
run;
```

The "Model Information" table in Output 82.10.1 describes the method used in the multiple imputation process.

Output 82.10.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FITNESS1
Method	Monotone-data MCMC
Multiple Imputation Chain	Single Chain
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	25
Number of Burn-in Iterations	200
Number of Iterations	100
Seed for random number generator	17655417

The "Missing Data Patterns" table in Output 82.10.2 lists distinct missing data patterns with corresponding statistics. Here, an "X" means that the variable is observed in the corresponding group, a "." means that the variable is missing and will be imputed to achieve the monotone missingness for the imputed data set, and an "O" means that the variable is missing and will not be imputed. The table also displays group-specific variable means.

Output 82.10.2 Missing Data Patterns

Missing Data Patterns								
						Group Means		
Group	Oxygen	RunTime	RunPulse	Freq	Percent	Oxygen	RunTime	RunPulse
1	X	X	X	21	67.74	46.353810	10.809524	171.666667
2	X	X	O	4	12.90	47.109500	10.137500	.
3	X	O	O	3	9.68	52.461667	.	.
4	.	X	X	1	3.23	.	11.950000	176.000000
5	.	X	O	2	6.45	.	9.885000	.

As shown in the table in Output 82.10.2, the MI procedure needs to impute only three missing values from group 4 and group 5 to achieve a monotone missing pattern for the imputed data set.

When you use the MCMC method to produce an imputed data set with a monotone missing pattern, tables of variance information and parameter estimates are not created.

The following statements are used just to show the monotone missingness of the output data set Outex10:

```
proc mi data=outex10 seed=15541 nimpute=0;
   var Oxygen RunTime RunPulse;
run;
```

The "Missing Data Patterns" table in Output 82.10.3 displays a monotone missing data pattern.

Output 82.10.3 Monotone Missing Data Patterns

The MI Procedure

Missing Data Patterns								
						Group Means		
Group	Oxygen	RunTime	RunPulse	Freq	Percent	Oxygen	RunTime	RunPulse
1	X	X	X	550	70.97	46.179104	10.861364	171.863636
2	X	X	.	150	19.35	47.774669	10.053333	.
3	X	.	.	75	9.68	52.461667	.	.

The following statements impute one value for each missing value in the monotone missingness data set Outex10:

```
proc mi data=outex10 nimpute=1 seed=51343672 out=outex10a;
   monotone method=reg;
   var Oxygen RunTime RunPulse;
   by _Imputation_;
run;
```

You can then analyze these data sets by using other SAS procedures and combine these results by using the MIANALYZE procedure. Note that the VAR statement is required with a MONOTONE statement to provide the variable order for the monotone missing pattern.

Example 82.11: Checking Convergence in MCMC

(View the complete code for this example (miex11.sas) in the example repository.)

This example uses the MCMC method with a single chain. It also displays trace and autocorrelation plots to check convergence for the single chain.

The following statements use the MCMC method to create an iteration plot for the successive estimates of the mean of Oxygen. These statements also create an autocorrelation function plot for the variable Oxygen.

```
ods graphics on;
proc mi data=Fitness1 seed=501213 nimpute=5 mu0=50 10 180;
   mcmc plots=(trace(mean(Oxygen)) acf(mean(Oxygen)));
   var Oxygen RunTime RunPulse;
run;
```

With ODS Graphics enabled, the TRACE(MEAN(OXYGEN)) option in the PLOTS= option displays the trace plot of means for the variable Oxygen, as shown in Output 82.11.1. The dashed vertical lines indicate the imputed iterations—that is, the Oxygen values used in the imputations. The plot shows no apparent trends for the variable Oxygen.

Output 82.11.1 Trace Plot for Oxygen

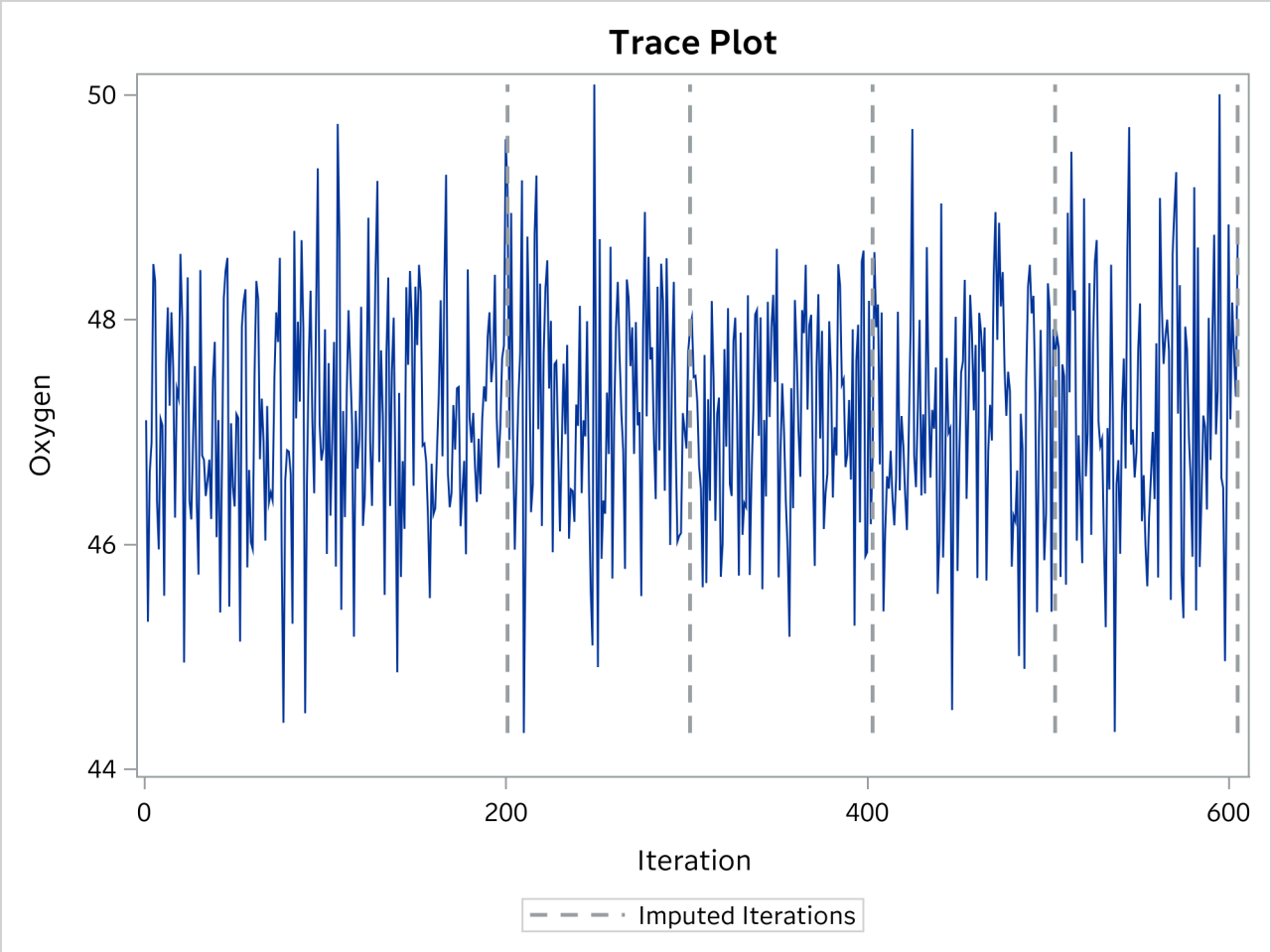

The ACF(MEAN(OXYGEN)) option in the PLOTS= option displays the autocorrelation plot of means for the variable Oxygen, as shown in Output 82.11.2. The autocorrelation function plot shows no significant positive or negative autocorrelation.

Output 82.11.2 Autocorrelation Function Plot for Oxygen

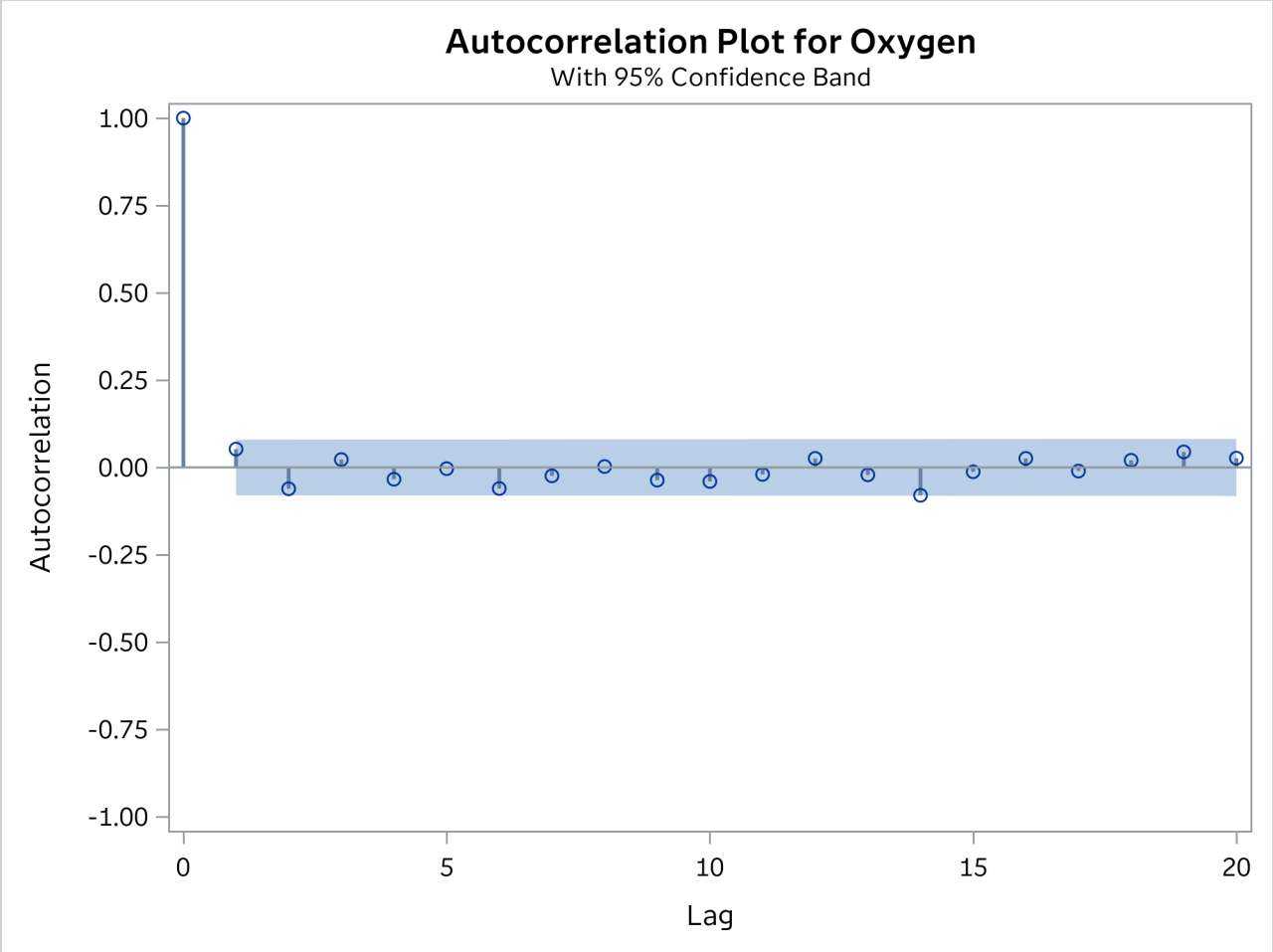

You can also create plots for the worst linear function, the means of other variables, the variances of variables, and the covariances between variables. Alternatively, you can use the OUTITER option to save statistics such as the means, standard deviations, covariances, –2 log LR statistic, –2 log LR statistic of the posterior mode, and worst linear function from each iteration in an output data set. Then you can do a more in-depth trace (time series) analysis of the iterations with other procedures, such as PROC AUTOREG and PROC ARIMA in the *SAS/ETS User's Guide*.

For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the MI procedure, see the section "ODS Graphics" on page 6636.

Example 82.12: Saving and Using Parameters for MCMC

(View the complete code for this example (miex12.sas) in the example repository.)

This example uses the MCMC method with multiple chains as specified in Example 82.9. It saves the parameter values used for each imputation in an output data set of type EST called Miest. This output data set can then be used to impute missing values in other similar input data sets. The following statements invoke the MI procedure and specify the MCMC method with multiple chains to create 40 imputations:

```
proc mi data=Fitness1 seed=21355417 nimpute=40 mu0=50 10 180;
   mcmc chain=multiple initial=em outest=miest;
   var Oxygen RunTime RunPulse;
run;
```

The following statements list the parameters used for the imputations in Output 82.12.1. Note that the data set includes observations with _TYPE_='SEED' which contains the seed to start the next random number generator.

```
proc print data=miest(obs=15);
   title 'Parameters for the Imputations';
run;
```

Output 82.12.1 OUTEST Data Set

Parameters for the Imputations

Obs	_Imputation_	_TYPE_	_NAME_	Oxygen	RunTime	RunPulse
1	1	SEED		825240167.00	825240167.00	825240167.00
2	1	PARM		46.77	10.47	169.41
3	1	COV	Oxygen	30.59	-8.32	-50.99
4	1	COV	RunTime	-8.32	2.90	17.03
5	1	COV	RunPulse	-50.99	17.03	200.09
6	2	SEED		1895925872.00	1895925872.00	1895925872.00
7	2	PARM		47.41	10.37	173.34
8	2	COV	Oxygen	22.35	-4.44	-21.18
9	2	COV	RunTime	-4.44	1.76	1.25
10	2	COV	RunPulse	-21.18	1.25	125.67
11	3	SEED		137653011.00	137653011.00	137653011.00
12	3	PARM		48.21	10.36	170.52
13	3	COV	Oxygen	23.59	-5.25	-19.76
14	3	COV	RunTime	-5.25	1.66	5.00
15	3	COV	RunPulse	-19.76	5.00	110.99

The following statements invoke the MI procedure and use the INEST= option in the MCMC statement:

```
proc mi data=Fitness1 mu0=50 10 180;
   mcmc inest=miest;
   var Oxygen RunTime RunPulse;
run;
```

The "Model Information" table in Output 82.12.2 describes the method used in the multiple imputation process. The remaining tables for the example are identical to the tables in Output 82.9.2, Output 82.9.4,

Output 82.9.5, and Output 82.9.6 in Example 82.9.

Output 82.12.2 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FITNESS1
Method	MCMC
INEST Data Set	WORK.MIEST
Number of Imputations	40

Example 82.13: Transforming to Normality

(View the complete code for this example (miex13.sas) in the example repository.)

This example applies the MCMC method to the Fitness1 data set in which the variable Oxygen is transformed. Assume that Oxygen is skewed and can be transformed to normality with a logarithmic transformation. The following statements invoke the MI procedure and specify the transformation. The TRANSFORM statement specifies the log transformation for Oxygen. Note that the values displayed for Oxygen in all of the results correspond to transformed values.

```
proc mi data=Fitness1 seed=32937921
        nimpute=pctmissing mu0=50 10 180
        out=outex13;
   transform log(Oxygen);
   mcmc chain=multiple displayinit;
   var Oxygen RunTime RunPulse;
run;
```

The NIMPUTE=PCTMISSING option uses the percentage of the incomplete cases as the number of imputations.

The "Model Information" table in Output 82.13.1 describes the method and options used in the multiple imputation process.

Output 82.13.1 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FITNESS1
Method	MCMC
Multiple Imputation Chain	Multiple Chains
Initial Estimates for MCMC	EM Posterior Mode
Start	Starting Value
Prior	Jeffreys
Number of Imputations	33
Number of Burn-in Iterations	200
Seed for random number generator	32937921

The "Missing Data Patterns" table in Output 82.13.2 lists distinct missing data patterns with corresponding statistics for the Fitness1 data. Note that the values of Oxygen shown in the tables are transformed values.

Output 82.13.2 Missing Data Patterns

Missing Data Patterns								
						Group Means		
Group	**Oxygen**	**RunTime**	**RunPulse**	**Freq**	**Percent**	**Oxygen**	**RunTime**	**RunPulse**
1	X	X	X	21	67.74	3.829760	10.809524	171.666667
2	X	X	.	4	12.90	3.851813	10.137500	.
3	X	.	.	3	9.68	3.955298	.	.
4	.	X	X	1	3.23	.	11.950000	176.000000
5	.	X	.	2	6.45	.	9.885000	.
Transformed Variables: Oxygen								

For the NIMPUTE=PCTMISSING option, the percentage of the incomplete cases, 10/31 = 32.3%, is used as the number of imputations. Thus, 33 imputations (after rounding up) are generated.

The "Variable Transformations" table in Output 82.13.3 lists the variables that have been transformed.

Output 82.13.3 Variable Transformations

Variable Transformations	
Variable	**_Transform_**
Oxygen	LOG

The "Initial Parameter Estimates for MCMC" table in Output 82.13.4 displays the starting mean and covariance estimates used in the MCMC method.

Output 82.13.4 Initial Parameter Estimates

Initial Parameter Estimates for MCMC				
TYPE	**_NAME_**	**Oxygen**	**RunTime**	**RunPulse**
MEAN		3.846122	10.557605	171.382949
COV	Oxygen	0.010827	-0.120891	-0.328772
COV	RunTime	-0.120891	1.744580	3.011180
COV	RunPulse	-0.328772	3.011180	82.747609
Transformed Variables: Oxygen				

Output 82.13.5 displays variance information from the multiple imputation.

Output 82.13.5 Variance Information

Variance Information (33 Imputations)							
	Variance						
Variable	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
* Oxygen	0.000008582	0.000407	0.000416	27.572	0.021732	0.021297	0.999355
RunTime	0.002396	0.068237	0.070706	27.17	0.036179	0.034989	0.998941
RunPulse	0.862937	3.230086	4.119173	21.41	0.275252	0.218114	0.993434
* Transformed Variables							

Output 82.13.6 displays parameter estimates from the multiple imputation. Note that the parameter value of μ_0 has also been transformed using the logarithmic transformation.

Output 82.13.6 Parameter Estimates

Parameter Estimates (33 Imputations)										
Variable	Mean	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Mu0	t for H0: Mean=Mu0	Pr > |t|
* Oxygen	3.846347	0.020388	3.8046	3.8881	27.572	3.838599	3.851483	3.912023	-3.22	0.0033
RunTime	10.543129	0.265905	9.9977	11.0886	27.17	10.452018	10.633302	10.000000	2.04	0.0509
RunPulse	171.648705	2.029575	167.4329	175.8645	21.41	169.858210	173.316307	180.000000	-4.11	0.0005
* Transformed Variables										

The following statements list the first 10 observations of the data set Outex13 in Output 82.13.7. Note that the values for Oxygen are in the original scale.

```
proc print data=outex13(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.13.7 Imputed Data Set in Original Scale

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Oxygen	RunTime	RunPulse
1	1	44.6090	11.3700	178.000
2	1	45.3130	10.0700	185.000
3	1	54.2970	8.6500	156.000
4	1	59.5710	7.1440	167.012
5	1	49.8740	9.2200	170.092
6	1	44.8110	11.6300	176.000
7	1	38.5834	11.9500	176.000
8	1	43.7376	10.8500	158.851
9	1	39.4420	13.0800	174.000
10	1	60.0550	8.6300	170.000

Note that the results in Output 82.13.7 can also be produced from the following statements without using a TRANSFORM statement. A transformed value of log(50)=3.91202 is used in the MU0= option.

```
data temp;
   set Fitness1;
   LogOxygen= log(Oxygen);
run;
proc mi data=temp seed=14337921 mu0=3.91202 10 180 out=outtemp;
   mcmc chain=multiple displayinit;
   var LogOxygen RunTime RunPulse;
run;
data outex13;
   set outtemp;
   Oxygen= exp(LogOxygen);
run;
```

Example 82.14: Multistage Imputation

(View the complete code for this example (miex14.sas) in the example repository.)

This example uses two separate imputation procedures to complete the imputation process. In the first case, the MI procedure statements use the MCMC method to impute just enough missing values for a data set with an arbitrary missing pattern so that each imputed data set has a monotone missing pattern. In the second case, the MI procedure statements use a MONOTONE statement to impute missing values for data sets with monotone missing patterns.

The following statements are identical to those in Example 82.10. The statements invoke the MI procedure and specify the IMPUTE=MONOTONE option to create the imputed data set with a monotone missing pattern.

```
proc mi data=Fitness1 seed=17655417 nimpute=10 out=outex14;
   mcmc impute=monotone;
   var Oxygen RunTime RunPulse;
run;
```

The "Missing Data Patterns" table in Output 82.14.1 lists distinct missing data patterns with corresponding statistics. Here, an "X" means that the variable is observed in the corresponding group, a "." means that the variable is missing and will be imputed to achieve the monotone missingness for the imputed data set, and an "O" means that the variable is missing and will not be imputed. The table also displays group-specific variable means.

Output 82.14.1 Missing Data Patterns

The MI Procedure

Missing Data Patterns								
						Group Means		
Group	Oxygen	RunTime	RunPulse	Freq	Percent	Oxygen	RunTime	RunPulse
1	X	X	X	21	67.74	46.353810	10.809524	171.666667
2	X	X	O	4	12.90	47.109500	10.137500	.
3	X	O	O	3	9.68	52.461667	.	.
4	.	X	X	1	3.23	.	11.950000	176.000000
5	.	X	O	2	6.45	.	9.885000	.

As shown in the table, the MI procedure needs to impute only three missing values from group 4 and group 5 to achieve a monotone missing pattern for the imputed data set. When the MCMC method is used to produce an imputed data set with a monotone missing pattern, tables of variance information and parameter estimates are not created.

The following statements impute one value for each missing value in the monotone missingness data set Outex14:

```
proc mi data=outex14
        nimpute=1 seed=51343672
        out=outex14a;
   monotone reg;
   var Oxygen RunTime RunPulse;
   by _Imputation_;
run;
```

You can then analyze these data sets by using other SAS procedures and combine these results by using the MIANALYZE procedure. Note that the VAR statement is required with a MONOTONE statement to provide the variable order for the monotone missing pattern.

The "Model Information" table in Output 82.14.2 shows that a monotone method is used to generate imputed values in the first BY group.

Output 82.14.2 Model Information

The MI Procedure

Imputation Number=1

Model Information	
Data Set	WORK.OUTEX14
Method	Monotone
Number of Imputations	1
Seed for random number generator	51343672

The "Monotone Model Specification" table in Output 82.14.3 describes methods and imputed variables in the imputation model. The MI procedure uses the regression method to impute the variables RunTime and RunPulse in the model.

Output 82.14.3 Monotone Model Specification

Imputation Number=1

Monotone Model Specification	
Method	**Imputed Variables**
Regression	RunTime RunPulse

The "Missing Data Patterns" table in Output 82.14.4 lists distinct missing data patterns with corresponding statistics. It shows a monotone missing pattern for the imputed data set.

Output 82.14.4 Missing Data Patterns

Imputation Number=1

Missing Data Patterns								
						Group Means		
Group	Oxygen	RunTime	RunPulse	Freq	Percent	Oxygen	RunTime	RunPulse
1	X	X	X	22	70.97	46.057479	10.861364	171.863636
2	X	X	.	6	19.35	46.745227	10.053333	.
3	X	.	.	3	9.68	52.461667	.	.

The following statements list the first 10 observations of the data set Outex14a in Output 82.14.5:

```
proc print data=outex14a(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.14.5 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Oxygen	RunTime	RunPulse
1	1	44.6090	11.3700	178.000
2	1	45.3130	10.0700	185.000
3	1	54.2970	8.6500	156.000
4	1	59.5710	7.1569	169.914
5	1	49.8740	9.2200	159.315
6	1	44.8110	11.6300	176.000
7	1	39.8345	11.9500	176.000
8	1	45.3196	10.8500	151.252
9	1	39.4420	13.0800	174.000
10	1	60.0550	8.6300	170.000

This example presents an alternative to the full-data MCMC imputation, in which imputation of only a few missing values is needed to achieve a monotone missing pattern for the imputed data set. The example uses a monotone MCMC method that imputes fewer missing values in each iteration and achieves approximate stationarity in fewer iterations (Schafer 1997, p. 227). The example also demonstrates how to combine the monotone MCMC method with a method for monotone missing data, which does not rely on iterations of steps.

Example 82.15: Creating Control-Based Pattern Imputation in Sensitivity Analysis

(View the complete code for this example (miex15.sas) in the example repository.)

This example illustrates the pattern-mixture model approach to multiple imputation under the MNAR assumption by creating control-based pattern imputation.

Suppose that a pharmaceutical company is conducting a clinical trial to test the efficacy of a new drug. The trial consists of two groups of equally allocated patients: a treatment group that receives the new drug and a placebo control group. The variable Trt is an indicator variable, with a value of 1 for patients in the treatment group and a value of 0 for patients in the control group. The variable Y0 is the baseline efficacy score, and the variable Y1 is the efficacy score at a follow-up visit.

If the data set does not contain any missing values, then a regression model such as

$$Y1 = Trt\ Y0$$

can be used to test the treatment effect.

Suppose that the variables Trt and Y0 are fully observed and the variable Y1 contains missing values in both the treatment and control groups. Multiple imputation for missing values often assumes that the values are missing at random. But if missing Y1 values for individuals in the treatment group imply that these individuals no longer receive the treatment, then it is reasonable to assume that the conditional distribution of Y1 given Y0 for individuals who have missing Y1 values in the treatment group is similar to the corresponding distribution of individuals in the control group.

Ratitch and O'Kelly (2011) describe an implementation of the pattern-mixture model approach that uses a control-based pattern imputation. That is, an imputation model for the missing observations in the treatment group is constructed not from the observed data in the treatment group but rather from the observed data in the control group. This model is also the imputation model that is used to impute missing observations in the control group.

Table 82.11 shows the variables in the data set. For the control-based pattern imputation, all missing Y1 values are imputed based on the model that is constructed using observed Y1 data from the control group (Trt=0) only.

Table 82.11 Variables

Variables		
Trt	**Y0**	**Y1**
0	X	X
1	X	X
0	X	.
1	X	.

Suppose the data set Mono1 contains the data from the trial that have missing values in Y1. Output 82.15.1 lists the first 10 observations.

Output 82.15.1 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Trt	y0	y1
1	0	10.5212	11.3604
2	0	8.5871	8.5178
3	0	9.3274	.
4	0	9.7519	.
5	0	9.3495	9.4369
6	1	11.5192	13.2344
7	1	10.7841	.
8	1	9.7717	10.9407
9	1	10.1455	10.8279
10	1	8.2463	9.6844

The following statements implement the control-based pattern imputation:

```
proc mi data=Mono1 seed=14823 nimpute=15 out=outex15;
   class Trt;
   monotone reg (/details);
   mnar model( y1 / modelobs= (Trt='0'));
   var y0 y1;
run;
```

The MNAR statement imputes missing values for scenarios under the MNAR assumption. The MODEL option specifies that only observations where TRT=0 are used to derive the imputation model for the variable Y1. Thus, Y0 and Y1 (but not Trt) are specified in the VAR list.

The "Model Information" table in Output 82.15.2 describes the method that is used in the multiple imputation process.

Output 82.15.2 Model Information

The MI Procedure

Model Information	
Data Set	WORK.MONO1
Method	Monotone
Number of Imputations	15
Seed for random number generator	14823

The "Monotone Model Specification" table in Output 82.15.3 describes methods and imputed variables in the imputation model. The MI procedure uses the regression method to impute the variable Y1.

Output 82.15.3 Monotone Model Specification

Monotone Model Specification	
Method	Imputed Variables
Regression	y1

The "Missing Data Patterns" table in Output 82.15.4 lists distinct missing data patterns and their corresponding frequencies and percentages. The table confirms a monotone missing pattern for these variables.

Output 82.15.4 Missing Data Patterns

Missing Data Patterns						
					Group Means	
Group	y0	y1	Freq	Percent	y0	y1
1	X	X	75	75.00	9.996993	10.709706
2	X	.	25	25.00	10.181488	.

By default, for each imputed variable, all available observations are used in the imputation model. When you specify the MODEL option in the MNAR statement, the "Observations Used for Imputation Models Under MNAR Assumption" table in Output 82.15.5 lists the subset of observations that are used for the imputation model for Y1.

Output 82.15.5 Observations Used for Imputation Models under MNAR Assumption

Observations Used for Imputation Models Under MNAR Assumption	
Imputed Variable	Observations
y1	Trt = 0

When you specify the DETAILS option, the parameters that are estimated from the observed data and the parameters that are used in each imputation are displayed in Output 82.15.6.

Output 82.15.6 Regression Model

Regression Models for Monotone Method											
						Imputation					
Imputed Variable	Effect	Obs-Data	1	2	3	4	5	6	7	8	9
y1	Intercept	-0.30169	-0.174265	-0.280404	-0.275183	0.090601	-0.457480	-0.241909	-0.501351	-0.058460	-0.436650
y1	y0	0.69364	0.641733	0.629970	0.507776	0.752283	0.831001	0.970075	0.724584	0.623638	0.563499

Regression Models for Monotone Method							
				Imputation			
Imputed Variable	Effect	10	11	12	13	14	15
y1	Intercept	-0.509949	-0.542411	-0.082799	-0.243293	-0.502742	-0.213113
y1	y0	0.621280	0.677104	0.562119	0.512430	0.693212	0.699355

The following statements list the first 10 observations of the output data set Outex15 in Output 82.15.7:

```
proc print data=outex15(obs=10);
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.15.7 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Trt	y0	y1
1	1	0	10.5212	11.3604
2	1	0	8.5871	8.5178
3	1	0	9.3274	9.5786
4	1	0	9.7519	9.6060
5	1	0	9.3495	9.4369
6	1	1	11.5192	13.2344
7	1	1	10.7841	10.7873
8	1	1	9.7717	10.9407
9	1	1	10.1455	10.8279
10	1	1	8.2463	9.6844

Example 82.16: Adjusting Imputed Continuous Values in Sensitivity Analysis

(View the complete code for this example (miex16.sas) in the example repository.)

This example illustrates the pattern-mixture model approach to multiple imputation under the MNAR assumption by using specified shift parameters to adjust imputed continuous values.

Suppose that a pharmaceutical company is conducting a clinical trial to test the efficacy of a new drug. The trial consists of two groups of equally allocated patients: a treatment group that receives the new drug and a placebo control group. The variable Trt is an indicator variable, with a value of 1 for patients in the treatment group and a value of 0 for patients in the control group. The variable Y0 is the baseline efficacy score, and the variables Y1 and Y2 are the efficacy scores at two successive follow-up visits.

Suppose the data set Fcs1 contains the data from the trial that have possible missing values in Y1 and Y2. Output 82.16.1 lists the first 10 observations in the data set Fcs1.

Output 82.16.1 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Trt	y0	y1	y2
1	0	11.4826	11.0428	13.1181
2	0	9.6775	11.0418	8.9792
3	0	9.9504	.	11.2598
4	0	11.0282	11.4097	.
5	0	10.7107	10.5782	.
6	1	9.0601	8.4791	10.6421
7	1	9.0467	9.4985	10.4719
8	1	10.6290	9.4941	.
9	1	10.1277	10.9886	11.1983
10	1	9.6910	8.4576	10.9535

Also suppose that for the treatment group, the distribution of missing Y1 responses has an expected value that is 0.4 lower than that of the corresponding distribution of the observed Y1 responses. Similarly, the

distribution of missing Y2 responses has an expected value that is 0.5 lower than that of the corresponding distribution of the observed Y1 responses.

The following statements adjust the imputed Y1 and Y2 values by –0.4 and –0.5, respectively, for observations in the treatment group:

```
proc mi data=Fcs1 seed=52387 out=outex16;
   class Trt;
   fcs nbiter=25 reg( /details);
   mnar adjust( y1 /shift=-0.4 adjustobs=(Trt='1'))
        adjust( y2 /shift=-0.5 adjustobs=(Trt='1'));
   var Trt y0 y1 y2;
run;
```

The MNAR statement imputes missing values for scenarios under the MNAR assumption. The ADJUST option specifies parameters for adjusting the imputed values for specified subsets of observations. The first ADJUST option specifies the shift parameter $\delta = -0.4$ for the imputed Y1 values for observations for which TRT=1. The second ADJUST option specifies the shift parameter $\delta = -0.5$ for the imputed Y2 values for observations for which TRT=1.

Because Trt is listed in the VAR statement, it is used as a covariate for other imputed variables in the imputation process. In addition, because Trt is specified in the ADJUSTOBS= suboption, it is also used to select the subset of observations from which the imputed values for the variable are to be adjusted.

The "Model Information" table in Output 82.16.2 describes the method that is used in the multiple imputation process.

Output 82.16.2 Model Information

The MI Procedure

Model Information	
Data Set	WORK.FCS1
Method	FCS
Number of Imputations	25
Number of Burn-in Iterations	25
Seed for random number generator	52387

The "FCS Model Specification" table in Output 82.16.3 describes methods and imputed variables in the imputation model. The MI procedure uses the regression method to impute all the variables.

Output 82.16.3 FCS Model Specification

FCS Model Specification	
Method	**Imputed Variables**
Regression	y0 y1 y2
Discriminant Function	Trt

The "Missing Data Patterns" table in Output 82.16.4 lists distinct missing data patterns and their corresponding frequencies and percentages.

Output 82.16.4 Missing Data Patterns

Missing Data Patterns									
							Group Means		
Group	Trt	y0	y1	y2	Freq	Percent	y0	y1	y2
1	X	X	X	X	39	39.00	10.108397	10.380942	10.606255
2	X	X	X	.	29	29.00	10.207179	10.626839	.
3	X	X	.	X	32	32.00	9.604041	.	10.396557

The "MNAR Adjustments to Imputed Values" table in Output 82.16.5 lists the adjustment parameters for the five imputations.

Output 82.16.5 MNAR Adjustments to Imputed Values

MNAR Adjustments to Imputed Values		
Imputed Variable	Observations	Shift
y1	Trt = 1	-0.4000
y2	Trt = 1	-0.5000

The following statements list the first 10 observations of the data set Outex16 in Output 82.16.6:

```
proc print data=outex16(obs=10);
   var _Imputation_ Trt y0 y1 y2;
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.16.6 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Trt	y0	y1	y2
1	1	0	11.4826	11.0428	13.1181
2	1	0	9.6775	11.0418	8.9792
3	1	0	9.9504	11.1409	11.2598
4	1	0	11.0282	11.4097	10.8214
5	1	0	10.7107	10.5782	9.4899
6	1	1	9.0601	8.4791	10.6421
7	1	1	9.0467	9.4985	10.4719
8	1	1	10.6290	9.4941	10.7865
9	1	1	10.1277	10.9886	11.1983
10	1	1	9.6910	8.4576	10.9535

Example 82.17: Adjusting Imputed Classification Levels in Sensitivity Analysis

(View the complete code for this example (miex17.sas) in the example repository.)

This example illustrates the pattern-mixture model approach to multiple imputation under the MNAR assumption by adjusting imputed classification levels.

Carpenter and Kenward (2013, pp. 240–241) describe an implementation of sensitivity analysis that adjusts an imputed missing covariate, where the covariate is a nominal classification variable.

Suppose a high school class is conducting a study to analyze the effects of an extra web-based study class and grade level on the improvement of test scores. The regression model that is used for the study is

$$\text{Score} = \text{Grade} \;\; \text{Study} \;\; \text{Score0}$$

where Grade is the grade level (with the values 6 to 8), Study is an indicator variable (with the values 1 for "completes the study class" and 0 for "does not complete the study class"), Score0 is the current test score, and Score is the test score for the subsequent test.

Also suppose that Study, Score0, and Score are fully observed and the classification variable Grade contains missing grade levels. Output 82.17.1 lists the first 10 observations in the data set Mono2.

Output 82.17.1 Student Test Data

First 10 Obs in the Student Test Data

Obs	Grade	Score0	Score	Study
1	6	64.4898	68.8210	1
2	6	72.0700	76.5328	1
3	6	65.7766	75.5567	1
4	.	70.2853	76.0180	1
5	6	74.3388	80.0617	1
6	6	70.2207	76.1606	1
7	6	68.6904	77.9770	1
8	.	72.6758	79.6895	1
9	6	64.8939	69.3889	1
10	6	66.6038	72.7793	1

The following statements use the MONOTONE and MNAR statements to impute missing values for Grade under the MNAR assumption:

```
proc mi data=Mono2 seed=34857 nimpute=20 out=outex17;
   class Study Grade;
   monotone logistic (Grade / link=glogit);
   mnar adjust( Grade (event='6') /shift=2);
   var Study Score0 Score Grade;
run;
```

The LINK=GLOGIT suboption specifies that the generalized logit function be used in fitting the logistic model for Grade. The ADJUST option specifies a shift parameter $\delta = 2$ that is applied to the generalized

logit model function values for the response level GRADE=6. This assumes that students who have a missing grade level are more likely to be students in grade 6.

The "Model Information" table in Output 82.17.2 describes the method that is used in the multiple imputation process.

Output 82.17.2 Model Information

The MI Procedure

Model Information	
Data Set	WORK.MONO2
Method	Monotone
Number of Imputations	20
Seed for random number generator	34857

The "Monotone Model Specification" table in Output 82.17.3 describes methods and imputed variables in the imputation model. The MI procedure uses the logistic regression method (generalized logit model) to impute the variable Grade.

Output 82.17.3 Monotone Model Specification

Monotone Model Specification	
Method	**Imputed Variables**
Regression	Score0 Score
Logistic Regression	Grade

The "Missing Data Patterns" table in Output 82.17.4 lists distinct missing data patterns and their corresponding frequencies and percentages.

Output 82.17.4 Missing Data Patterns

Missing Data Patterns								
							Group Means	
Group	**Study**	**Score0**	**Score**	**Grade**	**Freq**	**Percent**	**Score0**	**Score**
1	X	X	X	X	128	85.33	70.418230	74.469573
2	X	X	X	.	22	14.67	69.338503	73.666293

The "MNAR Adjustments to Imputed Values" table in Output 82.17.5 lists the adjustment parameter for the 10 imputations.

Output 82.17.5 MNAR Adjustments to Imputed Values

MNAR Adjustments to Imputed Values		
Imputed Variable	**Event**	**Shift**
Grade	6	2.0000

The following statements list the first 10 observations of the data set Outex17 in Output 82.17.6:

```
proc print data=outex17(obs=10);
   var _Imputation_ Grade Study Score0 Score;
   title 'First 10 Observations of the Imputed Student Test Data Set';
run;
```

Output 82.17.6 Imputed Data Set

First 10 Observations of the Imputed Student Test Data Set

Obs	_Imputation_	Grade	Study	Score0	Score
1	1	6	1	64.4898	68.8210
2	1	6	1	72.0700	76.5328
3	1	6	1	65.7766	75.5567
4	1	6	1	70.2853	76.0180
5	1	6	1	74.3388	80.0617
6	1	6	1	70.2207	76.1606
7	1	6	1	68.6904	77.9770
8	1	6	1	72.6758	79.6895
9	1	6	1	64.8939	69.3889
10	1	6	1	66.6038	72.7793

Example 82.18: Adjusting Imputed Values with Parameters in a Data Set

(View the complete code for this example (miex18.sas) in the example repository.)

This example illustrates the pattern-mixture model approach in multiple imputation under the MNAR assumption by adjusting imputed values, using parameters that are stored in a data set.

Suppose that a pharmaceutical company is conducting a clinical trial to test the efficacy of a new drug. The trial consists of two groups of equally allocated patients: a treatment group that receives the new drug and a placebo control group. The variable Trt is an indicator variable, with a value of 1 for patients in the treatment group and a value of 0 for patients in the control group. The variable Y0 is the baseline efficacy score, and the variable Y1 is the efficacy score at a follow-up visit.

If the data set does not contain any missing values, then a regression model such as

$$Y1 = Trt \;\; Y0$$

can be used to test the efficacy of the treatment effect.

Now suppose that the variables Trt and Y0 are fully observed and the variable Y1 contains missing values in both the treatment and control groups. Table 82.12 shows the variables in the data set.

Table 82.12 Variables

Variables		
Trt	**Y0**	**Y1**
0	X	X
1	X	X
0	X	.
1	X	.

Suppose the data set Mono3 contains the data from the trial that have missing values in Y1. Output 82.18.1 lists the first 10 observations.

Output 82.18.1 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Trt	y0	y1
1	0	10.5212	11.3604
2	0	8.5871	8.5178
3	0	9.3274	.
4	0	9.7519	.
5	0	9.3495	9.4369
6	1	11.5192	13.1344
7	1	10.7841	.
8	1	9.7717	10.8407
9	1	10.1455	10.7279
10	1	8.2463	9.5844

Multiple imputation often assumes that missing values are MAR. Here, however, it is plausible that the distributions of missing Y1 responses in the treatment and control groups have lower expected values than the corresponding distributions of the observed Y1 responses. Carpenter and Kenward (2013, pp. 129–130) describe an implementation of the pattern-mixture model approach that uses different shift parameters for the treatment and control groups, where the two parameters are correlated.

Assume that the expected shifts of the missing follow-up responses in the control and treatment groups, δ_c and δ_t, have a multivariate normal distribution

$$\begin{pmatrix} \delta_c \\ \delta_t \end{pmatrix} \sim N\left(\begin{pmatrix} -0.5 \\ -1 \end{pmatrix}, \begin{pmatrix} 0.01 & 0.001 \\ 0.001 & 0.01 \end{pmatrix} \right)$$

The following statements generate shift parameters for the control and treatment groups for six imputations:

```
proc iml;

   nimpute= 10;
   call randseed( 15323);
   mean= { -0.5 -1};
   cov= { 0.01 0.001 , 0.001 0.01};
```

```
   /*---- Simulate nimpute bivariate normal variates ----*/
    d= randnormal( nimpute, mean, cov);

    impu= j(nimpute, 1, 0);
    do j=1 to nimpute;  impu[j,]= j;  end;
    delta= impu || d;

   /*--- Output shift parameters for groups ----*/
    create parm1 from delta[colname={_Imputation_ Shift_C Shift_T}];
    append from delta;
quit;
```

Output 82.18.2 lists the generated shift parameters in Parm1.

Output 82.18.2 Shift Parameters for Imputations

Shift Parameters for Imputations

Obs	_IMPUTATION_	SHIFT_C	SHIFT_T
1	1	-0.56986	-0.90494
2	2	-0.38681	-0.84523
3	3	-0.58342	-0.92793
4	4	-0.48210	-0.99031
5	5	-0.57188	-1.02095
6	6	-0.57604	-1.00853
7	7	-0.44167	-0.93250
8	8	-0.53309	-1.06614
9	9	-0.53281	-1.16694
10	10	-0.53502	-1.11011

The following statements impute missing values for Y1 under the MNAR assumption. The shift parameters for the 10 imputations that are stored in the Parm1 data set are used to adjust the imputed values.

```
proc mi data=Mono3 seed=1423741 nimpute=10 out=outex18;
   class Trt;
   monotone reg;
   mnar adjust( y1 / adjustobs=(Trt='0') parms(shift=shift_c)=parm1)
        adjust( y1 / adjustobs=(Trt='1') parms(shift=shift_t)=parm1);
   var Trt y0 y1;
run;
```

The ADJUST option specifies parameters for adjusting the imputed values of Y1 for specified subsets of observations. The first ADJUST option specifies that the shift parameters that are stored in the variable SHIFT_C are to be applied to the imputed Y1 values of observations where TRT=0 for the corresponding imputations. The second ADJUST option specifies that the shift parameters that are stored in the variable SHIFT_T are to be applied to the imputed Y1 values of observations where TRT=1 for the corresponding imputations.

The "Model Information" table in Output 82.18.3 describes the method that is used in the multiple imputation process.

Output 82.18.3 Model Information

The MI Procedure

Model Information	
Data Set	WORK.MONO3
Method	Monotone
Number of Imputations	10
Seed for random number generator	1423741

The "Monotone Model Specification" table in Output 82.18.4 describes methods and imputed variables in the imputation model. The MI procedure uses the regression method to impute the variable Y1.

Output 82.18.4 Monotone Model Specification

Monotone Model Specification	
Method	**Imputed Variables**
Regression	y0 y1

The "Missing Data Patterns" table in Output 82.18.5 lists distinct missing data patterns and their corresponding frequencies and percentages. The table confirms a monotone missing pattern for these variables.

Output 82.18.5 Missing Data Patterns

Missing Data Patterns							
						Group Means	
Group	**Trt**	**y0**	**y1**	**Freq**	**Percent**	**y0**	**y1**
1	X	X	X	75	75.00	9.996993	10.655039
2	X	X	.	25	25.00	10.181488	.

The "MNAR Adjustments to Imputed Values" table in Output 82.18.6 lists the adjustment parameters for the 10 imputations.

Output 82.18.6 MNAR Adjustments to Imputed Values

MNAR Adjustments to Imputed Values			
Imputed Variable	Imputation	Observations	Shift
y1	1	Trt = 0	-0.5699
	1	Trt = 1	-0.9049
	2	Trt = 0	-0.3868
	2	Trt = 1	-0.8452
	3	Trt = 0	-0.5834
	3	Trt = 1	-0.9279
	4	Trt = 0	-0.4821
	4	Trt = 1	-0.9903
	5	Trt = 0	-0.5719
	5	Trt = 1	-1.0209
	6	Trt = 0	-0.5760
	6	Trt = 1	-1.0085
	7	Trt = 0	-0.4417
	7	Trt = 1	-0.9325
	8	Trt = 0	-0.5331
	8	Trt = 1	-1.0661
	9	Trt = 0	-0.5328
	9	Trt = 1	-1.1669
	10	Trt = 0	-0.5350
	10	Trt = 1	-1.1101

The following statements list the first 10 observations of the data set Outex18 in Output 82.18.7:

```
proc print data=outex18(obs=10);
   var _Imputation_ Trt Y0 Y1;
   title 'First 10 Observations of the Imputed Data Set';
run;
```

Output 82.18.7 Imputed Data Set

First 10 Observations of the Imputed Data Set

Obs	_Imputation_	Trt	y0	y1
1	1	0	10.5212	11.3604
2	1	0	8.5871	8.5178
3	1	0	9.3274	8.2456
4	1	0	9.7519	10.5152
5	1	0	9.3495	9.4369
6	1	1	11.5192	13.1344
7	1	1	10.7841	9.4660
8	1	1	9.7717	10.8407
9	1	1	10.1455	10.7279
10	1	1	8.2463	9.5844

Example 82.19: Selecting Variables for the Imputation Model

(View the complete code for this example (miex19.sas) in the example repository.)

This example shows how to use the FLUX and SIMPLE options in the PROC MI statement in variable selection for an imputation model. The data contain several variables that come from the Framingham Heart Study. The analysis and interpretation here are purely illustrative; you should not draw definitive conclusions from this example.

The following statements print the first 10 observations of the heart data set, which are shown in Output 82.19.1:

```
proc print data=sashelp.heart(obs=10);
run;
```

Output 82.19.1 First 10 Observations of heart Data Set

Obs	Status	DeathCause	AgeCHDdiag	Sex	AgeAtStart	Height	Weight	Diastolic	Systolic	MRW
1	Dead	Other	.	Female	29	62.50	140	78	124	121
2	Dead	Cancer	.	Female	41	59.75	194	92	144	183
3	Alive		.	Female	57	62.25	132	90	170	114
4	Alive		.	Female	39	65.75	158	80	128	123
5	Alive		.	Male	42	66.00	156	76	110	116
6	Alive		.	Female	58	61.75	131	92	176	117
7	Alive		.	Female	36	64.75	136	80	112	110
8	Dead	Other	.	Male	53	65.50	130	80	114	99
9	Alive		.	Male	35	71.00	194	68	132	124
10	Dead	Cerebral Vascular Disease	.	Male	52	62.50	129	78	124	106

Obs	Smoking	AgeAtDeath	Cholesterol	Chol_Status	BP_Status	Weight_Status	Smoking_Status
1	0	55	.		Normal	Overweight	Non-smoker
2	0	57	181	Desirable	High	Overweight	Non-smoker
3	10	.	250	High	High	Overweight	Moderate (6-15)
4	0	.	242	High	Normal	Overweight	Non-smoker
5	20	.	281	High	Optimal	Overweight	Heavy (16-25)
6	0	.	196	Desirable	High	Overweight	Non-smoker
7	15	.	196	Desirable	Normal	Overweight	Moderate (6-15)
8	0	77	276	High	Normal	Normal	Non-smoker
9	0	.	211	Borderline	Normal	Overweight	Non-smoker
10	5	82	284	High	Normal	Normal	Light (1-5)

This data set contains both categorical variables and continuous variables. The variables Status, Sex, AgeAtStart, Diastolic, Systolic, and BP_Status are fully observed. All other variables have some amount of missingness. Suppose that the goal is to create imputations for this data set. The following code uses the SIMPLE and FLUX options to display additional summary statistics that you can use to create an imputation model:

```
ods select MissPattern Flux FluxPlot Corr;

ods graphics on;
```

```
proc mi data=sashelp.heart simple flux nimpute=0
      displaypattern=nomeans;
   class _character_;
   var _all_;
   fcs;
run;
```

The ODS SELECT statement is used to select the most relevant output to aid in the variable selection process. The NIMPUTE=0 option suppresses the creation of imputed values. Because the data set contains character variables, the classification variable is specified by using the special variable _character_, which designates that all character variables in the data set are classification variables. The VAR option uses the special variable _all_ to indicate that all variables in the data set are to be used in the analysis. The CLASS statement is specified, so the FCS statement must also be included.

Output 82.19.2 shows that the heart data set has an arbitrary pattern of missingness.

Output 82.19.2 Missing Data Pattern in heart Data

The MI Procedure

Missing Data Patterns												
Group	Status	DeathCause	AgeCHDdiag	Sex	AgeAtStart	Height	Weight	Diastolic	Systolic	MRW	Smoking	AgeAtDeath
1	X	X	X	X	X	X	X	X	X	X	X	X
2	X	X	X	X	X	X	X	X	X	X	X	X
3	X	X	X	X	X	X	X	X	X	X	.	X
4	X	X	X	X	X	X	.	X	X	.	X	X
5	X	X	X	X	X	.	X	X	X	X	X	X
6	X	X	.	X	X	X	X	X	X	X	X	X
7	X	X	.	X	X	X	X	X	X	X	X	X
8	X	X	.	X	X	X	X	X	X	X	.	X
9	X	X	.	X	X	X	X	X	X	X	.	X
10	X	X	.	X	X	.	X	X	X	X	X	X
11	X	X	.	X	X	.	.	X	X	.	X	X
12	X	.	X	X	X	X	X	X	X	X	X	.
13	X	.	X	X	X	X	X	X	X	X	X	.
14	X	.	X	X	X	X	X	X	X	X	.	.
15	X	.	X	X	X	X	X	X	X	X	.	.
16	X	.	.	X	X	X	X	X	X	X	X	.
17	X	.	.	X	X	X	X	X	X	X	X	.
18	X	.	.	X	X	X	X	X	X	X	.	.
19	X	.	.	X	X	X	X	X	X	X	.	.
20	X	.	.	X	X	X	.	X	X	.	X	.
21	X	.	.	X	X	.	.	X	X	.	X	.

Missing Data Patterns							
Group	Cholesterol	Chol_Status	BP_Status	Weight_Status	Smoking_Status	Freq	Percent
1	X	X	X	X	X	864	16.59
2	.	.	X	X	X	21	0.40
3	.	.	X	X	.	6	0.12
4	X	X	X	.	X	2	0.04
5	X	X	X	X	X	1	0.02
6	X	X	X	X	X	1050	20.16
7	.	.	X	X	X	29	0.56
8	X	X	X	X	.	1	0.02
9	.	.	X	X	.	13	0.25
10	X	X	X	X	X	3	0.06
11	X	X	X	.	X	1	0.02
12	X	X	X	X	X	540	10.37
13	.	.	X	X	X	12	0.23
14	X	X	X	X	.	1	0.02
15	.	.	X	X	.	2	0.04
16	X	X	X	X	X	2585	49.63
17	.	.	X	X	X	62	1.19
18	X	X	X	X	.	6	0.12
19	.	.	X	X	.	7	0.13
20	X	X	X	.	X	2	0.04
21	X	X	X	.	X	1	0.02

Output 82.19.3 shows a table of select missing data statistics. The variables Status, Sex, AgeAtStart, Diastolic, Systolic, and BP_Status are fully observed. The influx and outflux values are similar for most variables; only three are unusual. You can see this most easily in the scatter plot shown in Output 82.19.4.

Output 82.19.3 Influx-Outflux Pattern Statistics

Influx-Outflux Pattern Statistics						
Variable	Influx	Outflux	Average Inbound	Average Outbound	FICO	Percent Observed
Status	0	1.0000	0	0.1271	0.8341	100
DeathCause	0.5824	0.1216	0.8817	0.04046	0.5660	38.22
AgeCHDdiag	0.6996	0.1149	0.9066	0.05249	0.4037	27.82
Sex	0	1.0000	0	0.1271	0.8341	100
AgeAtStart	0	1.0000	0	0.1271	0.8341	100
Height	0.00106	0.9982	0.8646	0.1271	0.8339	99.88
Weight	0.00092	0.9972	0.7500	0.1269	0.8339	99.88
Diastolic	0	1.0000	0	0.1271	0.8341	100
Systolic	0	1.0000	0	0.1271	0.8341	100
MRW	0.00092	0.9972	0.7500	0.1269	0.8339	99.88
Smoking	0.00545	0.9824	0.7378	0.1258	0.8330	99.31
AgeAtDeath	0.5824	0.1216	0.8817	0.04046	0.5660	38.22
Cholesterol	0.02498	0.9399	0.8006	0.1231	0.8291	97.08
Chol_Status	0.02498	0.9399	0.8006	0.1231	0.8291	97.08
BP_Status	0	1.0000	0	0.1271	0.8341	100
Weight_Status	0.00092	0.9972	0.7500	0.1269	0.8339	99.88
Smoking_Status	0.00545	0.9824	0.7378	0.1258	0.8330	99.31

Output 82.19.4 Scatter Plot of Influx and Outflux Values for heart Data

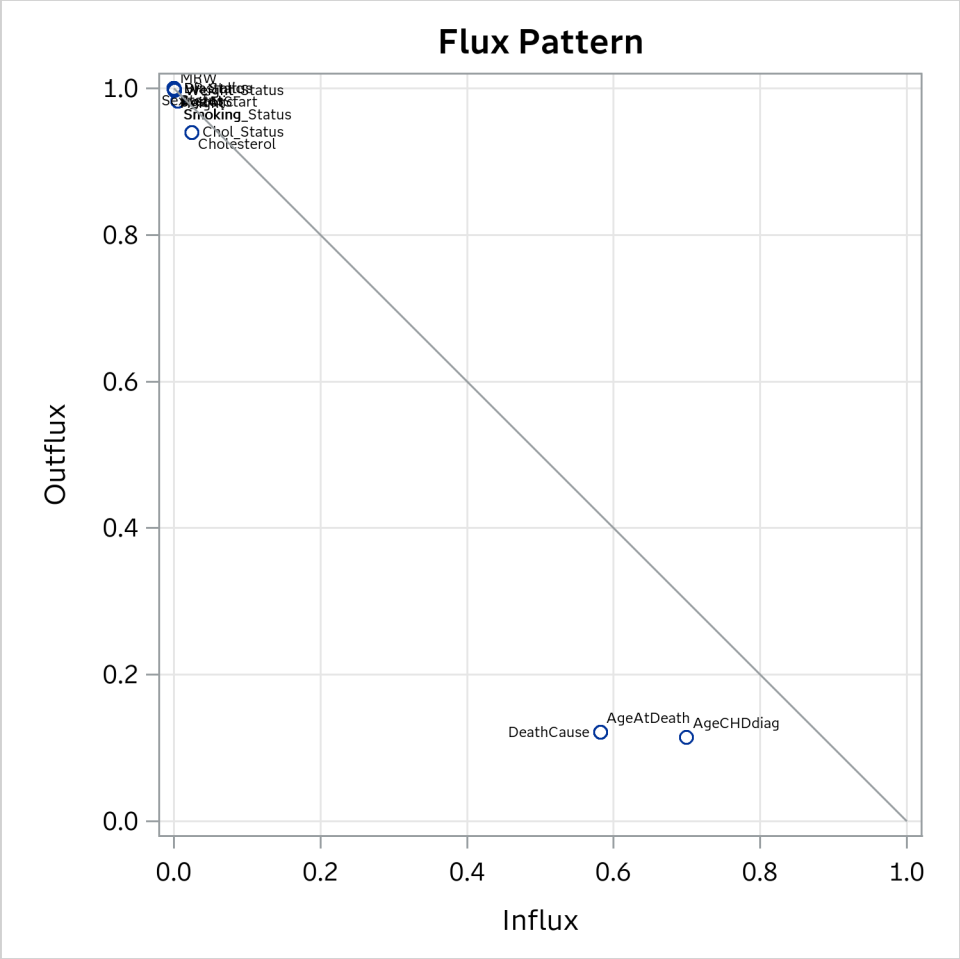

Nearly all variables are clustered in the upper left corner, signifying high outflux and low influx. They are also close to the diagonal reference, indicating that their influx and outflux values are balanced. Such variables are useful to include in an imputation model and provide good estimates of missing values.

Three variables appear in a separate cluster in the lower right corner: DeathCause, AgeAtDeath, and AgeCHDdiag. All three variables combine a low outflux (< 0.2) with a relatively high influx. This combination indicates that imputations for these variables are highly model-dependent. Such variables need greater care in modeling; if they are not of interest, they are candidates for removal from the imputation model.

In reviewing these specific variables, note that this might represent a case of missing data that do need not to be imputed. You can expect the variables AgeAtDeath and DeathCause to be missing for all observations in the data set whose Status value is "alive." In this case, the Status variable is fully observed. Only observations whose Status value is "Dead" need to have AgeAtDeath and DeathCause imputed. If such cases exist, you would be able to impute these variables separately from the "intentionally" missing ones by using the MNAR statement, as shown in "Example 82.15: Creating Control-Based Pattern Imputation in Sensitivity Analysis" on page 6676.

Whether the variable AgeCHDdiag is intentionally missing depends on whether the particular individual has a CHD diagnosis. Because the data set does not include a variable that indicates a CHD diagnosis, you cannot

answer whether missing values in AgeCHDdiag are missing intentionally or whether they require imputation. To avoid creating a biased data set, you should impute these variables by using the MNAR statement, as shown in "Example 82.15: Creating Control-Based Pattern Imputation in Sensitivity Analysis" on page 6676.

When you specify the SIMPLE option, the procedure prints a list of pairwise correlations, shown in Output 82.19.5. You can use these correlations as both inclusion and exclusion criteria when you design an imputation model. When you impute a missing variable, choosing covariates that are highly correlated to the target variable is beneficial. In this data set, you can see that Diastolic is highly correlated with Systolic. In cases in which only one of them is missing, the other would make a good variable to include in the imputation model. In addition, this table can be used to avoid potential collinearity in regression models that are generated using PROC MI.

Output 82.19.5 Pairwise Correlations

Pairwise Correlations

	AgeCHDdiag	AgeAtStart	Height	Weight	Diastolic	Systolic
AgeCHDdiag	1.000000000	0.550908991	-0.210415504	-0.137414605	-0.035104887	0.069241448
AgeAtStart	0.550908991	1.000000000	-0.131733393	0.093519327	0.275401211	0.379378605
Height	-0.210415504	-0.131733393	1.000000000	0.517385012	-0.014245067	-0.071127255
Weight	-0.137414605	0.093519327	0.517385012	1.000000000	0.327568337	0.263584256
Diastolic	-0.035104887	0.275401211	-0.014245067	0.327568337	1.000000000	0.796060887
Systolic	0.069241448	0.379378605	-0.071127255	0.263584256	0.796060887	1.000000000
MRW	0.006421658	0.204514655	-0.136289482	0.767171433	0.385109505	0.362574158
Smoking	-0.283566835	-0.167430706	0.287031561	0.088703399	-0.064121402	-0.092378164
AgeAtDeath	0.748112037	0.688604555	-0.136575313	0.004604067	0.010038815	0.102173521
Cholesterol	0.003628956	0.273414661	-0.079586533	0.072428448	0.183363653	0.199350303

Pairwise Correlations

	MRW	Smoking	AgeAtDeath	Cholesterol
AgeCHDdiag	0.006421658	-0.283566835	0.748112037	0.003628956
AgeAtStart	0.204514655	-0.167430706	0.688604555	0.273414661
Height	-0.136289482	0.287031561	-0.136575313	-0.079586533
Weight	0.767171433	0.088703399	0.004604067	0.072428448
Diastolic	0.385109505	-0.064121402	0.010038815	0.183363653
Systolic	0.362574158	-0.092378164	0.102173521	0.199350303
MRW	1.000000000	-0.125236962	0.102925364	0.137354823
Smoking	-0.125236962	1.000000000	-0.285249042	-0.011779654
AgeAtDeath	0.102925364	-0.285249042	1.000000000	0.078444026
Cholesterol	0.137354823	-0.011779654	0.078444026	1.000000000

Example 82.20: Selecting a Variable Order for Monotone Imputation

(View the complete code for this example (miex20.sas) in the example repository.)

This example shows how to use the MONOTONESEARCH option in the MCMC statement in the MI procedure. The following statements create the data set test0, which contains several variables that have an arbitrary missingness pattern:

```
%let seed1=34423; %let seed2=47564; %let seed3=99;
data test0;
   array vrb{4} vrb1-vrb4;
   do c=1 to 3;
   do g=1 to 4;
   do i=1 to 10;

   cls= c; if (ranuni(&seed1.) > .8) then cls=.;
   gp= g;  if (ranuni(&seed1.) > .8) then gp=.;

   do j=1 to 4;
      vrb{j}= rannor(&seed2.);
      if (ranuni(&seed1.) > .8) then vrb{j}= .;
   end;

   frq= int(10*ranuni(&seed3.));
   if (frq < 1) then frq= 1;

   if (vrb{1}^=.) or (vrb{2}^=.) or (vrb{3}^=.) or (vrb{4}^=.)
      then output;

   end; end; end; drop g c i j;
run;
```

The first 10 observations of this data set are displayed in Output 82.20.1.

Output 82.20.1 Data Set with Arbitrary Missingness Patterns

Obs	vrb1	vrb2	vrb3	vrb4
1	-0.35851	1.27186	-2.08304	-1.41027
2	0.56234	-1.70966	.	1.23885
3	.	0.93827	1.19951	0.28085
4	1.18831	-2.14854	-1.03242	-0.97547
5	-0.04129	-1.15515	0.27841	1.76727
6	.	0.69177	0.22743	-1.54723
7	0.99017	0.96084	0.20089	1.30463
8	-0.71539	0.19170	-0.52539	-0.37715
9	-0.57624	-0.90563	-0.05805	1.28465
10	2.11989	-1.49948	0.70663	.

The goal is to impute this data set to a monotone missingness pattern before proceeding with the imputation. The number of missing values that need to be imputed depends on the order of the variables—that is, the

column order—and the best order is not known. The following statements request suggestions for column order from PROC MI:

```
proc mi data=test0 nimpute=0 displaypattern=nomeans;
   mcmc impute=monotone monotonesearch;
   freq frq;
   var vrb1-vrb4;
   ods output MonoVarOrderCost=VarOrderTable;
run;
```

The NIMPUTE=0 option specifies that no imputations be performed. The DISPLAYPATTERN=NOMEANS option displays the "Missing Data Patterns" table without means.

The IMPUTE=MONOTONE option in the MCMC statement creates a "Missing Data Patterns" table that visually indicates which variables are not imputed to create a monotone missing data pattern, given the variable order that is specified in the VAR statement. The MONOTONESEARCH option requests that PROC MI find an optimized variable order to minimize the number of missing values that need to be imputed to create a monotone missing data pattern.

The FREQ statement specifies the variable FRQ as the frequency variable. Each observation in the data set represents FRQ observations with the same values for the analysis variables.

When you specify the IMPUTE=MONOTONE option, PROC MI produces a "Missing Data Patterns" table as shown in Output 82.20.2. This table shows which variables need to be imputed so that the data set has a monotone missing pattern. It uses "." to signify that a variable needs to be imputed and "O" to signify that a variable does not need to be imputed.

Output 82.20.2 Missing Data Patterns in test0

The MI Procedure

Missing Data Patterns						
Group	**vrb1**	**vrb2**	**vrb3**	**vrb4**	**Freq**	**Percent**
1	X	X	X	X	238	44.57
2	X	X	X	O	37	6.93
3	X	X	.	X	73	13.67
4	X	X	O	O	1	0.19
5	X	.	X	X	44	8.24
6	X	.	X	O	11	2.06
7	X	.	.	X	7	1.31
8	.	X	X	X	68	12.73
9	.	X	X	O	25	4.68
10	.	X	.	X	3	0.56
11	.	X	O	O	1	0.19
12	.	.	X	X	11	2.06
13	.	.	X	O	8	1.50
14	.	.	.	X	7	1.31

In this example, the first pattern group is fully observed and requires no values to be imputed. The second pattern group has a monotone order. Therefore, the missing values in vrb4 do not need to be imputed. This cell is therefore filled with an "O". For the third pattern group, the 73 missing values in vrb3 break the monotone pattern and need to be imputed, as the "." indicates. The other patterns are interpreted similarly.

The MONOTONESEARCH option produces the table shown in Output 82.20.3. This table shows the cost that is associated with the current variable order. It additionally suggests two alternative orders that reduce the number of missing values that need to be imputed to create a monotone missing data pattern.

Output 82.20.3 Variable Order Cost

Monotone Variable Order Cost		
Order Method	Imputations Needed	Percent to Fill
Current	301	77.98
Percent Missing	245	63.47
Optimized	221	57.25

The table shows that 301 missing values need to be imputed in the data set, given the order of variables in the VAR statement, in order to produce a monotone missing data pattern. This represents approximately 78% of all missing values in the data, as shown in the "Current" order method row. By sorting the variables according to increasing missingness, this number can be reduced to 245 values, as shown in the "Percent Missing" row. By using PROC MI's optimization method, you can further reduce this number to 221 values, as shown in the "Optimized" method row.

This data set has only four variables, so PROC MI defaults to optimizing by using Heap's algorithm to check all possible permutations. This means that no other possible order of the variables could require a smaller number of imputations. For more information about the methods that PROC MI uses to find an optimal variable order, see the section "MONOTONESEARCH Option" on page 6607.

When you specify the MONOTONESEARCH option, PROC MI populates the _MIVOPCT macro with the variable list that represents the variable order that is associated with the percentage missing solution and the _MIVOOPT macro with the variable order that represents the optimized solution. You can display their values in the log by using the &PUT statement. For example, to display the _MIVOOPT macro, you can use the following statement: `%PUT &=_MIVOOPT;`.

The "Monotone Variable Order Cost" table shown in Output 82.20.3 contains an additional, nonprinting column that stores the variable order information for each ordering method. In this example, you used the ODS OUTPUT statement to capture and save the MonoVarOrderCost data table to a SAS data set. The following statements print the MonoVarOrderCost table, including the captured nonprinting column:

```
proc print data=VarOrderTable;
run;
```

Output 82.20.4 shows the captured MonoVarOrderCost data set, including the nonprinting column VarList, which contains the variable order that is associated with each solution.

Output 82.20.4 Suggested Variable Orders

Obs	OrderMethod	ImpNeed	PctFill	VarList
1	Current	301	77.98	vrb1 vrb2 vrb3 vrb4
2	Percent Missing	245	63.47	vrb4 vrb2 vrb3 vrb1
3	Optimized	221	57.25	vrb3 vrb2 vrb4 vrb1

You can use the prepopulated macros in the following statements. You call PROC MI with the IMPUTE=MONOTONE option, and one of the optimized macros is supplied for the variable order.

```
proc mi data=test0 seed=17655417 nimpute=10 out=outex20;
   mcmc impute=monotone;
   var &_MIVOOPT.;
run;
```

The resulting data set, outex20, is now monotone. This can be seen in subsequent calls to PROC MI:

```
proc mi data=outex20 nimpute=0 displaypattern=nomeans;
   var &_MIVOOPT.;
run;
```

These statements use the NIMPUTE=0 and DISPLAYPATTERN=NOMEANS options to display the "Missing Data Patterns" table for the imputed data set outex20 without creating imputations and without means, respectively. The VAR statement uses the optimized variable order as saved in the _MIVOOPT macro. The "Missing Data Patterns" table for this variable order is shown in Output 82.20.5. As you can see, the missing data pattern is now monotone.

Output 82.20.5 Monotone Missing Data Patterns in outex20 Generated Using Optimized Variable Order

The MI Procedure

Missing Data Patterns						
Group	vrb3	vrb2	vrb4	vrb1	Freq	Percent
1	X	X	X	X	950	79.83
2	X	X	X	.	190	15.97
3	X	X	.	.	40	3.36
4	X	.	.	.	10	0.84

The next step is to fully impute the missing data set either by using a MONOTONE statement or by using the FCS statement to take advantage of the fast burn-in of a monotone data set.

References

Allison, P. (2012). "Why You Probably Need More Imputations Than You Think." Accessed February 20, 2015. http://www.statisticalhorizons.com/more-imputations.

Allison, P. D. (2000). "Multiple Imputation for Missing Data: A Cautionary Tale." *Sociological Methods and Research* 28:301–309.

Allison, P. D. (2001). *Missing Data*. Thousand Oaks, CA: Sage Publications.

Anderson, T. W. (1984). *An Introduction to Multivariate Statistical Analysis*. 2nd ed. New York: John Wiley & Sons.

Barnard, J., and Meng, X. L. (1999). "Applications of Multiple Imputation in Medical Studies: From AIDS to NHANES." *Statistical Methods in Medical Research* 8:17–36.

Barnard, J., and Rubin, D. B. (1999). "Small-Sample Degrees of Freedom with Multiple Imputation." *Biometrika* 86:948–955.

Bodner, T. E. (2008). "What Improves with Increased Missing Data Imputations?" *Structural Equation Modeling* 15:651–675.

Brand, J. P. L. (1999). "Development, Implementation, and Evaluation of Multiple Imputation Strategies for the Statistical Analysis of Incomplete Data Sets." Ph.D. thesis, Erasmus University.

Carpenter, J. R., and Kenward, M. G. (2013). *Multiple Imputation and Its Application*. New York: John Wiley & Sons.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society, Series B* 39:1–38.

Gadbury, G. L., Coffey, C. S., and Allison, D. B. (2003). "Modern Statistical Methods for Handling Missing Repeated Measurements in Obesity Trial Data: Beyond LOCF." *Obesity Reviews* 4:175–184.

Gelman, A., and Rubin, D. B. (1992). "Inference from Iterative Simulation Using Multiple Sequences." *Statistical Science* 7:457–472.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Graham, J. W., Olchowski, A. E., and Gilreath, T. D. (2007). "How Many Imputations Are Really Needed? Some Practical Clarifications of Multiple Imputation Theory." *Prevention Science* 8:206–213.

Heitjan, F., and Little, R. J. A. (1991). "Multiple Imputation for the Fatal Accident Reporting System." *Journal of the Royal Statistical Society, Series C* 40:13–29.

Horton, N. J., and Lipsitz, S. R. (2001). "Multiple Imputation in Practice: Comparison of Software Packages for Regression Models with Missing Variables." *American Statistician* 55:244–254.

Lavori, P. W., Dawson, R., and Shera, D. (1995). "A Multiple Imputation Strategy for Clinical Trials with Truncation of Patient Data." *Statistics in Medicine* 14:1913–1925.

Li, K. H. (1988). "Imputation Using Markov Chains." *Journal of Statistical Computation and Simulation* 30:57–79.

Li, K. H., Raghunathan, T. E., and Rubin, D. B. (1991). "Large-Sample Significance Levels from Multiply Imputed Data Using Moment-Based Statistics and an F Reference Distribution." *Journal of the American Statistical Association* 86:1065–1073.

Little, R. J. A. (1993). "Pattern-Mixture Models for Multivariate Incomplete Data." *Journal of the American Statistical Association* 88:125–134.

Little, R. J. A. (1995). "Modeling the Drop-Out Mechanism in Repeated-Measures Studies." *Journal of the American Statistical Association* 90:1112–1121.

Little, R. J. A., and Rubin, D. B. (2002). *Statistical Analysis with Missing Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Liu, C. (1993). "Bartlett's Decomposition of the Posterior Distribution of the Covariance for Normal Monotone Ignorable Missing Data." *Journal of Multivariate Analysis* 46:198–206.

McLachlan, G. J., and Krishnan, T. (1997). *The EM Algorithm and Extensions*. New York: John Wiley & Sons.

Molenberghs, G., and Kenward, M. G. (2007). *Missing Data in Clinical Studies*. New York: John Wiley & Sons.

National Research Council (2010). *The Prevention and Treatment of Missing Data in Clinical Trials.* Panel on Handling Missing Data in Clinical Trials, Committee on National Statistics, Division of Behavioral and Social Sciences and Education. Washington, DC: National Academies Press.

O'Neill, R. T., and Temple, R. (2012). "The Prevention and Treatment of Missing Data in Clinical Trials: An FDA Perspective on the Importance of Dealing with It." *Clinical Pharmacology and Therapeutics* 91:550–554.

Ratitch, B., and O'Kelly, M. (2011). "Implementation of Pattern-Mixture Models Using Standard SAS/STAT Procedures." In *Proceedings of PharmaSUG 2011 (Pharmaceutical Industry SAS Users Group).* Paper SP04. Cary, NC: SAS Institute Inc.

Rosenbaum, P. R., and Rubin, D. B. (1983). "The Central Role of the Propensity Score in Observational Studies for Causal Effects." *Biometrika* 70:41–55.

Rubin, D. B. (1976). "Inference and Missing Data." *Biometrika* 63:581–592.

Rubin, D. B. (1987). *Multiple Imputation for Nonresponse in Surveys*. New York: John Wiley & Sons.

Rubin, D. B. (1996). "Multiple Imputation after 18+ Years." *Journal of the American Statistical Association* 91:473–489.

Schafer, J. L. (1997). *Analysis of Incomplete Multivariate Data*. New York: Chapman & Hall.

Schafer, J. L. (1999). "Multiple Imputation: A Primer." *Statistical Methods in Medical Research* 8:3–15.

Schenker, N., and Taylor, J. M. G. (1996). "Partially Parametric Techniques for Multiple Imputation." *Computational Statistics and Data Analysis* 22:425–446.

Szu, H., and Hartley, R. (1987). "Fast Simulated Annealing." *Physics Letters A* 122:157–162.

Tanner, M. A., and Wong, W. H. (1987). "The Calculation of Posterior Distributions by Data Augmentation." *Journal of the American Statistical Association* 82:528–540.

Van Buuren, S. (2007). "Multiple Imputation of Discrete and Continuous Data by Fully Conditional Specification." *Statistical Methods in Medical Research* 16:219–242.

Van Buuren, S. (2019). *Flexible Imputation of Missing Data*. 2nd ed. Boca Raton, FL: Chapman & Hall/CRC.

Van Buuren, S., Boshuizen, H. C., and Knook, D. L. (1999). "Multiple Imputation of Missing Blood Pressure Covariates in Survival Analysis." *Statistics in Medicine* 18:681–694.

Von Hippel, P. T. (2009). "How to Impute Interactions, Squares, and Other Transformed Variables." *Sociological Methodology* 39:265–291.

White, I. R., and Carlin, J. B. (2010). "Bias and Efficiency of Multiple Imputation Compared with Complete-Case Analysis for Missing Covariate Values." *Statistics in Medicine* 29:2920–2931. https://doi.org/10.1002/sim.3944.

White, I. R., Daniel, R., and Royston, P. (2010). "Avoiding Bias Due to Perfect Prediction in Multiple Imputation of Incomplete Categorical Variables." *Computational Statistics and Data Analysis* 54:2267–2275.

White, I. R., Royston, P., and Wood, A. M. (2011). "Multiple Imputation Using Chained Equations: Issues and Guidance for Practice." *Statistics in Medicine* 30:377–399.

Chapter 83
The MIANALYZE Procedure

Contents

Overview: MIANALYZE Procedure **6704**
Getting Started: MIANALYZE Procedure **6705**
Syntax: MIANALYZE Procedure **6707**
PROC MIANALYZE Statement 6708
BY Statement 6711
CLASS Statement 6711
MODELEFFECTS Statement 6712
STDERR Statement 6712
TEST Statement 6713
Details: MIANALYZE Procedure **6714**
Input Data Sets 6714
Combining Inferences from Imputed Data Sets 6719
Multiple Imputation Efficiency 6720
Multivariate Inferences 6721
Testing Linear Hypotheses about the Parameters 6722
Examples of the Complete-Data Inferences 6723
ODS Table Names 6724
Examples: MIANALYZE Procedure **6725**
Example 83.1: Reading Means and Standard Errors from a DATA= Data Set 6728
Example 83.2: Reading Means and Covariance Matrices from a DATA= COV Data Set 6730
Example 83.3: Reading Regression Results from a DATA= EST Data Set 6732
Example 83.4: Reading Mixed Model Results from PARMS= and COVB= Data Sets . 6733
Example 83.5: Reading Generalized Linear Model Results 6736
Example 83.6: Reading GLM Results from PARMS= and XPXI= Data Sets 6738
Example 83.7: Reading Logistic Model Results from a PARMS= Data Set 6740
Example 83.8: Reading Mixed Model Results with Classification Covariates 6741
Example 83.9: Reading Nominal Logistic Model Results 6743
Example 83.10: Using a TEST Statement 6747
Example 83.11: Combining Correlation Coefficients 6748
Example 83.12: Sensitivity Analysis with Control-Based Pattern Imputation 6751
Example 83.13: Sensitivity Analysis with the Tipping-Point Approach 6754
References **6758**

Overview: MIANALYZE Procedure

The MIANALYZE procedure combines the results of the analyses of imputations and generates valid statistical inferences. Multiple imputation provides a useful strategy for analyzing data sets with missing values. Instead of filling in a single value for each missing value, Rubin's (1976, 1987) multiple imputation strategy replaces each missing value with a set of plausible values that represent the uncertainty about the right value to impute.

Multiple imputation inference involves three distinct phases:

1. The missing data are filled in *m* times to generate *m* complete data sets.
2. The *m* complete data sets are analyzed using standard statistical analyses.
3. The results from the *m* complete data sets are combined to produce inferential results.

A companion procedure, PROC MI, creates multiply imputed data sets for incomplete multivariate data. It uses methods that incorporate appropriate variability across the *m* imputations.

The analyses of imputations are obtained by using standard SAS procedures (such as PROC REG) for complete data. No matter which complete-data analysis is used, the process of combining results from different imputed data sets is essentially the same and results in valid statistical inferences that properly reflect the uncertainty due to missing values. These results of analyses are combined in the MIANALYZE procedure to derive valid inferences.

The MIANALYZE procedure reads parameter estimates and associated standard errors or covariance matrix that are computed by the standard statistical procedure for each imputed data set. The MIANALYZE procedure then derives valid univariate inference for these parameters. With an additional assumption about the population between and within imputation covariance matrices, multivariate inference based on Wald tests can also be derived.

The MODELEFFECTS statement lists the effects to be analyzed, and the CLASS statement lists the classification variables in the MODELEFFECTS statement. The variables in the MODELEFFECTS statement that are not specified in a CLASS statement are assumed to be continuous.

When each effect in the MODELEFFECTS statement is a continuous variable by itself, a STDERR statement specifies the standard errors when both parameter estimates and associated standard errors are stored as variables in the same data set.

For some parameters of interest, you can use TEST statements to test linear hypotheses about the parameters. For others, it is not straightforward to compute estimates and associated covariance matrices with standard statistical SAS procedures. Examples include correlation coefficients between two variables and ratios of variable means. These special cases are described in the section "Examples of the Complete-Data Inferences" on page 6723.

Getting Started: MIANALYZE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mianags.sas.)

The Fitness data described in the REG procedure are measurements of 31 individuals in a physical fitness course. See Chapter 105, "The REG Procedure," for more information. The Fitness1 data set is constructed from the Fitness data set and contains three variables: Oxygen, RunTime, and RunPulse. Some values have been set to missing, and the resulting data set has an arbitrary pattern of missingness in these three variables.

```
*----------------- Data on Physical Fitness -----------------*
| These measurements were made on men involved in a physical |
| fitness course at N.C. State University.                   |
| Only selected variables of                                 |
| Oxygen (oxygen intake, ml per kg body weight per minute),  |
| Runtime (time to run 1.5 miles in minutes), and            |
| RunPulse (heart rate while running) are used.              |
| Certain values were changed to missing for the analysis.   |
*------------------------------------------------------------*;
data Fitness1;
   input Oxygen RunTime RunPulse @@;
   datalines;
44.609  11.37  178     45.313  10.07  185
54.297   8.65  156     59.571    .      .
49.874   9.22    .     44.811  11.63  176
  .     11.95  176          .  10.85    .
39.442  13.08  174     60.055   8.63  170
50.541    .      .     37.388  14.03  186
44.754  11.12  176     47.273    .      .
51.855  10.33  166     49.156   8.95  180
40.836  10.95  168     46.672  10.00    .
46.774  10.25    .     50.388  10.08  168
39.407  12.63  174     46.080  11.17  156
45.441   9.63  164       .      8.92    .
45.118  11.08    .     39.203  12.88  168
45.790  10.47  186     50.545   9.93  148
48.673   9.40  186     47.920  11.50  170
47.467  10.50  170
;
```

Suppose that the data are multivariate normally distributed and that the missing data are missing at random (see the section "Statistical Assumptions for Multiple Imputation" on page 6585 in Chapter 82, "The MI Procedure," for more information about these assumptions). The following statements use the MI procedure to impute missing values for the Fitness1 data set:

```
proc mi data=Fitness1 seed=3237851 noprint out=outmi;
   var Oxygen RunTime RunPulse;
run;
```

The MI procedure creates imputed data sets, which are stored in the Outmi data set. A variable named _Imputation_ indicates the imputation numbers. Based on m imputations, m different sets of the point and variance estimates for a parameter can be computed. In PROC MI, $m = 25$ is the default.

The following statements generate regression coefficients for each of the 25 imputed data sets:

```
proc reg data=outmi outest=outreg covout noprint;
   model Oxygen= RunTime RunPulse;
   by _Imputation_;
run;
```

The following statements display (in Figure 83.1) output parameter estimates and covariance matrices from PROC REG for the first two imputed data sets:

```
proc print data=outreg(obs=8);
   var _Imputation_ _Type_ _Name_
      Intercept RunTime RunPulse;
   title 'Parameter Estimates from Imputed Data Sets';
run;
```

Figure 83.1 Parameter Estimates

Parameter Estimates from Imputed Data Sets

Obs	_Imputation_	_TYPE_	_NAME_	Intercept	RunTime	RunPulse
1	1	PARMS		86.544	-2.82231	-0.05873
2	1	COV	Intercept	100.145	-0.53519	-0.55077
3	1	COV	RunTime	-0.535	0.10774	-0.00345
4	1	COV	RunPulse	-0.551	-0.00345	0.00343
5	2	PARMS		83.021	-3.00023	-0.02491
6	2	COV	Intercept	79.032	-0.66765	-0.41918
7	2	COV	RunTime	-0.668	0.11456	-0.00313
8	2	COV	RunPulse	-0.419	-0.00313	0.00264

The following statements combine the 25 sets of regression coefficients:

```
proc mianalyze data=outreg;
   modeleffects Intercept RunTime RunPulse;
run;
```

The “Model Information” table in Figure 83.2 lists the input data set(s) and the number of imputations.

Figure 83.2 Model Information Table

The MIANALYZE Procedure

Model Information	
Data Set	WORK.OUTREG
Number of Imputations	25

The “Variance Information” table in Figure 83.3 displays the between-imputation, within-imputation, and total variances for combining complete-data inferences. It also displays the degrees of freedom for the total variance, the relative increase in variance due to missing values, the fraction of missing information, and the relative efficiency for each parameter estimate.

Figure 83.3 Variance Information Table

Variance Information (25 Imputations)							
	Variance						
Parameter	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Intercept	22.485821	75.413875	98.799129	428.38	0.310092	0.240234	0.990482
RunTime	0.021126	0.124930	0.146902	1072.9	0.175870	0.151147	0.993990
RunPulse	0.000656	0.002622	0.003304	562.35	0.260376	0.209393	0.991694

The "Parameter Estimates" table in Figure 83.4 displays a combined estimate and standard error for each regression coefficient (parameter). Inferences are based on t distributions. The table displays a 95% confidence interval and a t test with the associated p-value for the hypothesis that the parameter is equal to the value specified with the THETA0= option (in this case, zero by default). The minimum and maximum parameter estimates from the imputed data sets are also displayed.

Figure 83.4 Parameter Estimates

Parameter Estimates (25 Imputations)										
Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	92.700420	9.939775	73.16362	112.2372	428.38	83.020730	100.839807	0	9.33	<.0001
RunTime	-3.030325	0.383278	-3.78238	-2.2783	1072.9	-3.280042	-2.754668	0	-7.91	<.0001
RunPulse	-0.079621	0.057482	-0.19253	0.0333	562.35	-0.135862	-0.024910	0	-1.39	0.1666

Syntax: MIANALYZE Procedure

The following statements are available in the MIANALYZE procedure:

```
PROC MIANALYZE <options> ;
    BY variables ;
    CLASS variables ;
    MODELEFFECTS effects ;
    <label:> TEST equation1 <, ..., <equationk>> </options> ;
    STDERR variables ;
```

The BY statement specifies groups in which separate analyses are performed.

The CLASS statement lists the classification variables in the MODELEFFECTS statement. Classification variables can be either character or numeric.

The required MODELEFFECTS statement lists the effects to be analyzed. The variables in the statement that are not specified in a CLASS statement are assumed to be continuous.

The STDERR statement lists the standard errors associated with the effects in the MODELEFFECTS statement when both parameter estimates and standard errors are saved as variables in the same DATA= data set. The STDERR statement can be used only when each effect in the MODELEFFECTS statement is a continuous variable by itself.

The TEST statement tests linear hypotheses about the parameters. An F statistic is used to jointly test the null hypothesis ($H_0 : \mathbf{L}_{\epsilon} = \mathbf{c}$) specified in a single TEST statement. Several TEST statements can be used.

The PROC MIANALYZE and MODELEFFECTS statements are required for the MIANALYZE procedure. The rest of this section provides detailed syntax information for each of these statements, beginning with the PROC MIANALYZE statement. The remaining statements are in alphabetical order.

PROC MIANALYZE Statement

PROC MIANALYZE < *options* > **;**

The PROC MIANALYZE statement invokes the MIANALYZE procedure. Table 83.1 summarizes the options available in the PROC MIANALYZE statement.

Table 83.1 Summary of PROC MIANALYZE Options

Option	Description
Input Data Sets	
DATA=	Specifies the COV, CORR, or EST type data set
DATA=	Specifies the data set for parameter estimates and standard errors
PARMS=	Specifies the data set for parameter estimates
PARMINFO=	Specifies the data set for parameter information
COVB=	Specifies the data set for covariance matrices
XPXI=	Specifies the data set for $(\mathbf{X}'\mathbf{X})^{-1}$ matrices
Statistical Analysis	
THETA0=	Specifies parameters under the null hypothesis
ALPHA=	Specifies the level for the confidence interval
EDF=	Specifies the complete-data degrees of freedom
Printed Output	
WCOV	Displays the within-imputation covariance matrix
BCOV	Displays the between-imputation covariance matrix
TCOV	Displays the total covariance matrix
MULT	Displays multivariate inferences

The following options can be used in the PROC MIANALYZE statement. They are listed in alphabetical order.

ALPHA=α

specifies that confidence limits are to be constructed for the parameter estimates with confidence level $100(1-\alpha)\%$, where $0 < \alpha < 1$. The default is ALPHA=0.05.

BCOV

displays the between-imputation covariance matrix.

COVB <(EFFECTVAR=STACKING | ROWCOL)> =*SAS-data-set*

names an input SAS data set that contains covariance matrices of the parameter estimates from imputed data sets. If you provide a COVB= data set, you must also provide a PARMS= data set.

The EFFECTVAR= option identifies the variables for parameters displayed in the covariance matrix and is used only when the PARMINFO= option is not specified. The default is EFFECTVAR= STACKING.

See the section "Input Data Sets" on page 6714 for a detailed description of the COVB= option.

DATA=*SAS-data-set*

names an input SAS data set.

If the input DATA= data set is not a specially structured SAS data set, the data set contains both the parameter estimates and associated standard errors. The parameter estimates are specified in the MODELEFFECTS statement and the standard errors are specified in the STDERR statement.

If the data set is a specially structured input SAS data set, it must have a TYPE of EST, COV, or CORR that contains estimates from imputed data sets:

- If TYPE=EST, the data set contains the parameter estimates and associated covariance matrices.
- If TYPE=COV, the data set contains the sample means, sample sizes, and covariance matrices. Each covariance matrix for variables is divided by the sample size n to create the covariance matrix for parameter estimates.
- If TYPE=CORR, the data set contains the sample means, sample sizes, standard errors, and correlation matrices. The covariance matrices are computed from the correlation matrices and associated standard errors. Each covariance matrix for variables is divided by the sample size n to create the covariance matrix for parameter estimates.

If you do not specify an input data set with the DATA= or PARMS= option, then the most recently created SAS data set is used as an input DATA= data set. See the section "Input Data Sets" on page 6714 for a detailed description of the input data sets.

EDF=*number*

specifies the complete-data degrees of freedom for the parameter estimates. This is used to compute an adjusted degrees of freedom for each parameter estimate. By default, EDF=∞ and the degrees of freedom for each parameter estimate are not adjusted.

MULT

MULTIVARIATE

requests multivariate inference for the parameters. It is based on Wald tests and is a generalization of the univariate inference. See the section "Multivariate Inferences" on page 6721 for a detailed description of the multivariate inference.

PARMINFO=*SAS-data-set*

names an input SAS data set that contains parameter information associated with variables PRM1, PRM2,..., and so on. These variables are used as variables for parameters in a COVB= data set. See the section "Input Data Sets" on page 6714 for a detailed description of the PARMINFO= option.

PARMS < (*options***) > =***SAS-data-set*

names an input SAS data set that contains parameter estimates computed from imputed data sets. When a COVB= data set is not specified, the input PARMS= data set also contains standard errors associated with these parameter estimates. If multivariate inference is requested, you must also provide a COVB= or XPXI= data set.

The available *options* are as follows:

CLASSVAR=FULL | LEVEL | CLASSVAL

identifies the associated classification variables when reading the classification levels from observations. The CLASSVAR= option is applicable only when the model effects contain classification variables. The default is CLASSVAR= FULL.

LINK=NONE | LOGIT | GLOGIT

identifies the type of parameter estimates. The LINK=NONE option (which is the default) indicates the parameter estimates that are derived from a procedure other than the LOGISTIC procedure.

The LINK=LOGIT option indicates the parameter estimates that are derived from the LOGISTIC procedure for ordinal responses. It is applicable only when the variable Intercept is in the MODELEFFECTS statement and the logistic model has more than two response levels. Otherwise, LINK=NONE should be used.

The LINK=GLOGIT option indicates the parameter estimates that are derived from the LOGISTIC procedure for nominal responses.

For a detailed description of the PARMS= option, see the section "PARMS < (*parms-options*) >= Data Set" on page 6716

TCOV

displays the total covariance matrix derived by assuming that the population between-imputation and within-imputation covariance matrices are proportional to each other.

THETA0=*numbers*

MU0=*numbers*

specifies the parameter values $\boldsymbol{\theta}_0$ under the null hypothesis $\boldsymbol{\theta} = \boldsymbol{\theta}_0$ in the t tests for location for the effects. If only one number $\boldsymbol{\theta}_0$ is specified, that number is used for all effects. If more than one number is specified, the specified numbers correspond to effects in the MODELEFFECTS statement in the order in which they appear in the statement. When an effect contains classification variables, the corresponding value is not used and the test is not performed.

WCOV

displays the within-imputation covariance matrices.

XPXI=*SAS-data-set*

names an input SAS data set that contains the $(\mathbf{X}'\mathbf{X})^{-1}$ matrices associated with the parameter estimates computed from imputed data sets. If you provide an XPXI= data set, you must also provide a PARMS= data set. In this case, PROC MIANALYZE reads the standard errors of the estimates from the PARMS= data. The standard errors and $(\mathbf{X}'\mathbf{X})^{-1}$ matrices are used to derive the covariance matrices.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC MIANALYZE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the MIANALYZE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* **;**

The CLASS statement specifies the classification variables in the MODELEFFECTS statement. Classification variables can be either character or numeric. Classification levels are determined from the formatted values of the classification variables. See "The FORMAT Procedure" in the *Base SAS Procedures Guide* for details.

MODELEFFECTS Statement

MODELEFFECTS *effects* **;**

The MODELEFFECTS statement lists the effects in the data set to be analyzed. Each effect is a variable or a combination of variables, and is specified with a special notation that uses variable names and operators.

Each variable is either a classification (or CLASS) variable or a continuous variable. If a variable is not declared in the CLASS statement, it is assumed to be continuous. Crossing and nesting operators can be used in an effect to create crossed and nested effects.

One general form of an effect involving several variables is

X1 * *X2* * *A* * *B* * *C* (*D E*)

where A, B, C, D, and E are classification variables and X1 and X2 are continuous variables.

When the input DATA= data set is not a specially structured SAS data set, you must also specify standard errors of the parameter estimates in an STDERR statement.

STDERR Statement

STDERR *variables* **;**

The STDERR statement lists standard errors associated with effects in the MODELEFFECTS statement, when the input DATA= data set contains both parameter estimates and standard errors as variables in the data set.

With the STDERR statement, only continuous effects are allowed in the MODELEFFECTS statement. The specified standard errors correspond to parameter estimates in the order in which they appear in the MODELEFFECTS statement.

For example, you can use the following MODELEFFECTS and STDERR statements to identify both the parameter estimates and associated standard errors in a SAS data set:

```
proc mianalyze;
   modeleffects y1 y2;
   stderr sy1 sy2;
run;
```

TEST Statement

< *label*: > **TEST** *equation1* < **,** . . . **,** < *equationk* > > < / *options* > **;**

The TEST statement tests linear hypotheses about the parameters $\boldsymbol{\beta}$. An F test is used to jointly test the null hypotheses ($H_0 : \mathbf{L}\boldsymbol{\beta} = \mathbf{c}$) specified in a single TEST statement in which the MULT option is specified.

Each *equation* specifies a linear hypothesis (a row of the $\mathbf{L}$ matrix and the corresponding element of the $\mathbf{c}$ vector); multiple *equations* are separated by commas. The label, which must be a valid SAS name, is used to identify the resulting output. You can submit multiple TEST statements. When a label is not included in a TEST statement, a label of "Test *j*" is used for the *j*th TEST statement.

The form of an *equation* is as follows:

term < ± *term* . . . > < = ± *term* < ± *term* . . . > >

where *term* is a parameter of the model, or a constant, or a constant times a parameter. When no equal sign appears, the expression is set to 0. Only parameters for regressor effects (continuous variables by themselves) are allowed.

For each TEST statement, PROC MIANALYZE displays a "Test Specification" table of the $\mathbf{L}$ matrix and the $\mathbf{c}$ vector. The procedure also displays a "Variance Information" table of the between-imputation, within-imputation, and total variances for combining complete-data inferences, and a "Parameter Estimates" table of a combined estimate and standard error for each linear component. The linear components are labeled TestPrm1, TestPrm2, . . . in the tables.

The following statements illustrate possible uses of the TEST statement:

```
proc mianalyze;
   modeleffects intercept a1 a2 a3;
   test1: test intercept + a2 = 0;
   test2: test intercept + a2;
   test3: test a1=a2=a3;
   test4: test a1=a2, a2=a3;
run;
```

The first and second TEST statements are equivalent and correspond to the specification in Figure 83.5.

Figure 83.5 Test Specification for test1 and test2

The MIANALYZE Procedure
Test: test1

Test Specification

	L Matrix				
Parameter	**intercept**	**a1**	**a2**	**a3**	**C**
TestPrm1	1.000000	0	1.000000	0	0

The third and fourth TEST statements are also equivalent and correspond to the specification in Figure 83.6.

Figure 83.6 Test Specification for test3 and test4

The MIANALYZE Procedure
Test: test3

Test Specification					
	L Matrix				
Parameter	intercept	a1	a2	a3	C
TestPrm1	0	1.000000	-1.000000	0	0
TestPrm2	0	0	1.000000	-1.000000	0

The ALPHA= and EDF options specified in the PROC MIANALYZE statement are also applied to the TEST statement. You can specify the following options in the TEST statement after a slash(/):

BCOV
displays the between-imputation covariance matrix.

MULT
displays the multivariate inference for parameters.

TCOV
displays the total covariance matrix.

WCOV
displays the within-imputation covariance matrix.

For more information, see the section "Testing Linear Hypotheses about the Parameters" on page 6722.

Details: MIANALYZE Procedure

Input Data Sets

You specify input data sets based on the type of inference you requested. For univariate inference, you can use one of the following options:

- a DATA= data set, which provides both parameter estimates and the associated standard errors
- a DATA=EST, COV, or CORR data set, which provides both parameter estimates and the associated standard errors either explicitly (type CORR) or through the covariance matrix (type EST, COV)
- PARMS= data set, which provides both parameter estimates and the associated standard errors

For multivariate inference, which includes the testing of linear hypotheses about parameters, you can use one of the following option combinations:

- a DATA=EST, COV, or CORR data set, which provides parameter estimates and the associated covariance matrix either explicitly (type EST, COV) or through the correlation matrix and standard errors (type CORR) in a single data set

- PARMS= and COVB= data sets, which provide parameter estimates in a PARMS= data set and the associated covariance matrix in a COVB= data set
- PARMS=, COVB=, and PARMINFO= data sets, which provide parameter estimates in a PARMS= data set, the associated covariance matrix in a COVB= data set with variables named PRM1, PRM2, . . . , and the effects associated with these variables in a PARMINFO= data set
- PARMS= and XPXI= data sets, which provide parameter estimates and the associated standard errors in a PARMS= data set and the associated $(\mathbf{X}'\mathbf{X})^{-1}$ matrix in an XPXI= data set

The appropriate combination depends on the type of inference and the SAS procedure you used to create the data sets. For instance, if you used PROC REG to create an OUTEST= data set that contains the parameter estimates and covariance matrix, you would use the DATA= option to read the OUTEST= data set.

When the input DATA= data set is a specially structured SAS data set, the data set must contain the variable _Imputation_ to identify the imputation by number. Otherwise, each observation corresponds to an imputation and contains both parameter estimates and associated standard errors.

If you do not specify an input data set with the DATA= or PARMS= option, then the most recently created SAS data set is used as an input DATA= data set. Note that with a DATA= data set, each effect represents a continuous variable; only regressor effects (continuous variables by themselves) are allowed in the MODELEFFECTS statement.

DATA= SAS Data Set

The DATA= data set provides both parameter estimates and the associated standard errors computed from imputed data sets. Such data sets are typically created with an OUTPUT statement in procedures such as PROC MEANS and PROC UNIVARIATE.

The MIANALYZE procedure reads parameter estimates from observations with variables in the MODELEFFECTS statement, and standard errors for parameter estimates from observations with variables in the STDERR statement. The order of the variables for standard errors must match the order of the variables for parameter estimates.

DATA=EST, COV, or CORR SAS Data Set

The specially structured DATA= data set provides both parameter estimates and the associated covariance matrix computed from imputed data sets. Such data sets are created by procedures such as PROC CORR (type COV, CORR) and PROC REG (type EST).

With a DATA=EST data set, the MIANALYZE procedure reads parameter estimates from observations with _TYPE_='PARM', _TYPE_='PARMS', _TYPE_='OLS', or _TYPE_='FINAL', and covariance matrices for parameter estimates from observations with _TYPE_='COV' or _TYPE_='COVB'.

With a DATA=COV data set, the procedure reads sample means from observations with _TYPE_='MEAN', sample size n from observations with _TYPE_='N', and covariance matrices for variables from observations with _TYPE_='COV'.

With a DATA=CORR data set, the procedure reads sample means from observations with _TYPE_='MEAN', sample size n from observations with _TYPE_='N', correlation matrices for variables from observations with _TYPE_='CORR', and standard errors for variables from observations with _TYPE_='STD'. The standard errors and correlation matrix are used to generate a covariance matrix for the variables.

Note that with a DATA=COV or DATA=CORR data set, each covariance matrix for the variables is divided by n to create the covariance matrix for the sample means.

PARMS <(*parms-options*)>= Data Set

The PARMS= data set contains both parameter estimates and the associated standard errors computed from imputed data sets. Such data sets are typically created with an ODS OUTPUT statement in procedures such as PROC GENMOD, PROC GLM, PROC LOGISTIC, and PROC MIXED.

The MIANALYZE procedure reads effect names from observations with the variable Parameter, Effect, Variable, or Parm. It then reads parameter estimates from observations with the variable Estimate and standard errors for parameter estimates from observations with the variable StdErr.

The available *parms-options* include the CLASSVAR= option to identify classification variables and the LINK= option to input logistic regression results. When the parameter estimates are derived from the LOGISTIC procedure, the LINK= option can be used to identify the variable required when the parameter estimates are read from observations. The available options are as follows:

- LINK=NONE (which is the default), in which each model effect is completely identified from the effect name. This option should be used for all procedures except PROC LOGISTIC.
- LINK=LOGIT, in which the variable ClassVal0 is used to identify response levels for Intercept from PROC LOGISTIC for ordinal responses. This option is applicable only when the variable Intercept is in the MODELEFFECTS statement and the logistic model has more than two response levels. Otherwise, LINK=NONE should be used.
- LINK=GLOGIT, in which the variable Response is used to identify response levels for the parameters from PROC LOGISTIC for nominal responses.

When the effects contain classification variables, the CLASSVAR= option can be used to identify the variables when reading the classification levels from observations. The available options are:

- CLASSVAR=FULL (which is the default), the data set contains the classification variables explicitly. PROC MIANALYZE reads the classification levels from observations with their corresponding classification variables. PROC MIXED generates this type of table.
- CLASSVAR=LEVEL, PROC MIANALYZE reads the classification levels for the effect from observations with variables Level1, Level2, and so on, where the variable Level1 contains the classification level for the first classification variable in the effect, and the variable Level2 contains the classification level for the second classification variable in the effect. For each effect, the variables in the crossed list are displayed before the variables in the nested list. The variable order in the CLASS statement is used for variables inside each list. PROC GENMOD generates this type of table.

  For example, with the following statements, the variable Level1 has the classification level of the variable c2 for the effect c2:

```
proc mianalyze parms(classvar=Level)=dataparm;
   class c1 c2 c3;
   modeleffects c2 c3(c2 c1);
run;
```

For the effect c3(c2 c1), the variable Level1 has the classification level of the variable c3, Level2 has the level of c1, and Level3 has the level of c2.

- CLASSVAR=CLASSVAL, PROC MIANALYZE reads the classification levels for the effect from observations with variables ClassVal0, ClassVal1, and so on, where the variable ClassVal0 contains the classification level for the first classification variable in the effect, and the variable ClassVal1 contains the classification level for the second classification variable in the effect. For each effect, the variables in the crossed list are displayed before the variables in the nested list. The variable order in the CLASS statement is used for variables inside each list. PROC LOGISTIC generates this type of tables.

PARMS <(*parms-options*)>= and COVB <(EFFECTVAR=*etype*)>= Data Sets

The PARMS= data set contains parameter estimates, and the COVB= data set contains associated covariance matrices computed from imputed data sets. Such data sets are typically created with an ODS OUTPUT statement in procedures such as PROC LOGISTIC, PROC MIXED, and PROC REG.

When you specify a PARMS= data set, the MIANALYZE procedure reads effect names from observations with the variable Parameter, Effect, Variable, or Parm. It then reads parameter estimates from observations with the variable Estimate.

The available *parms-options* include the CLASSVAR= option to identify classification variables and the LINK= option to input logistic regression results. For a detailed description of the PARMS= option, see the section "PARMS <(*parms-options*)>= Data Set" on page 6716.

The EFFECTVAR=*etype* option identifies the variables for parameters displayed in the covariance matrix. The available types are STACKING and ROWCOL:

- EFFECTVAR=STACKING (which is the default), each parameter is displayed by stacking variables in the effect. Begin with the variables in the crossed list, followed by the continuous list, then followed by the nested list. Each classification variable is displayed with its classification level attached. PROC LOGISTIC generates this type of table. When each effect is a continuous variable by itself, each stacked parameter name reduces to the effect name. PROC REG generates this type of table.

  The MIANALYZE procedure reads parameter names from observations with the variable Parameter, Effect, Variable, Parm, or RowName. It then reads covariance matrices from observations with the stacked variables in a COVB= data set.

- EFFECTVAR=ROWCOL, parameters are displayed by the variables Col1, Col2, . . . The parameter associated with the variable Col1 is identified by the observation with value 1 for the variable Row. The parameter associated with the variable Col2 is identified by the observation with value 2 for the variable Row. PROC MIXED generates this type of table.

  The MIANALYZE procedure reads the parameter indices from observations with the variable Row and the effect names from observations with the variable Parameter, Effect, Variable, Parm, or RowName. It then reads covariance matrices from observations with the variables Col1, Col2, and so on in a COVB= data set.

  When the effects contain classification variables, the data set contains the classification variables explicitly and the MIANALYZE procedure also reads the classification levels from their corresponding classification variables.

PARMS <(CLASSVAR= *ctype*)> =, PARMINFO=, and COVB= Data Sets

The input PARMS= data set contains parameter estimates, the PARMINFO= data set identifies parameters with the variables Prm1, Prm2, and so on, and the COVB= data set contains associated covariance matrices with the variables Prm1, Prm2, and so on. Such data sets are typically created with an ODS OUTPUT statement using procedure such as PROC GENMOD.

When you specify a PARMS= data set, the MIANALYZE procedure reads effect names from observations with the variable Parameter, Effect, Variable, or Parm. It then reads parameter estimates from observations with the variable Estimate.

When the effects contain classification variables, the option CLASSVAR= *ctype* can be used to identify the associated classification variables when reading the classification levels from observations. The available types are FULL, LEVEL, and CLASSVAL, and they are described in the section "PARMS <(*parms-options*) >= Data Set" on page 6716. The default is CLASSVAR= FULL.

When you specify a COVB= data set, the MIANALYZE procedure reads parameter names from observations with the variable Parameter, Effect, Variable, Parm, or RowName. It then reads covariance matrices from observations with the variables Prm1, Prm2, and so on.

The parameters associated with the variables Prm1, Prm2, and so on are identified in the PARMINFO= data set. PROC MIANALYZE reads the parameter names from observations with the variable Parameter and the corresponding effect from observations with the variable Effect. When the effects contain classification variables, the data set contains the classification variables explicitly and the MIANALYZE procedure also reads the classification levels from observations with their corresponding classification variables.

PARMS= and XPXI= Data Sets

The input PARMS= data set contains parameter estimates, and the input XPXI= data set contains associated $(\mathbf{X}'\mathbf{X})^{-1}$ matrices computed from imputed data sets. Such data sets are typically created with an ODS OUTPUT statement in a procedure such as PROC GLM.

When you specify a PARMS= data set, the MIANALYZE procedure reads parameter names from observations with the variable Parameter, Effect, Variable, or Parm. It then reads parameter estimates from observations with the variable Estimate and standard errors for parameter estimates from observations with the variable StdErr.

When you specify a XPXI= data set, the MIANALYZE procedure reads parameter names from observations with the variable Parameter and $(\mathbf{X}'\mathbf{X})^{-1}$ matrices from observations with the parameter variables in the data set.

Note that this combination can be used only when each effect is a continuous variable by itself.

Combining Inferences from Imputed Data Sets

With m imputations, m different sets of the point and variance estimates for a parameter Q can be computed. Suppose that $\hat{Q}_i$ and $\hat{W}_i$ are the point and variance estimates, respectively, from the ith imputed data set, $i = 1, 2, \ldots, m$. Then the combined point estimate for Q from multiple imputation is the average of the m complete-data estimates:

$$\overline{Q} = \frac{1}{m}\sum_{i=1}^{m}\hat{Q}_i$$

Suppose that $\overline{W}$ is the within-imputation variance, which is the average of the m complete-data estimates:

$$\overline{W} = \frac{1}{m}\sum_{i=1}^{m}\hat{W}_i$$

And suppose that B is the between-imputation variance:

$$B = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{Q}_i - \overline{Q})^2$$

Then the variance estimate associated with $\overline{Q}$ is the total variance (Rubin 1987)

$$T = \overline{W} + (1 + \frac{1}{m})B$$

The statistic $(Q - \overline{Q})T^{-(1/2)}$ is approximately distributed as t with v_m degrees of freedom (Rubin 1987), where

$$v_m = (m-1)\left[1 + \frac{\overline{W}}{(1+m^{-1})B}\right]^2$$

The degrees of freedom v_m depend on m and the ratio

$$r = \frac{(1+m^{-1})B}{\overline{W}}$$

The ratio r is called the relative increase in variance due to nonresponse (Rubin 1987). When there is no missing information about Q, the values of r and B are both zero. With a large value of m or a small value of r, the degrees of freedom v_m will be large and the distribution of $(Q - \overline{Q})T^{-(1/2)}$ will be approximately normal.

Another useful statistic is the fraction of missing information about Q:

$$\hat{\lambda} = \frac{r + 2/(v_m + 3)}{r + 1}$$

Both statistics r and λ are helpful diagnostics for assessing how the missing data contribute to the uncertainty about Q.

When the complete-data degrees of freedom v_0 are small, and there is only a modest proportion of missing data, the computed degrees of freedom, v_m, can be much larger than v_0, which is inappropriate. For example, with $m = 5$ and $r = 10\%$, the computed degrees of freedom $v_m = 484$, which is inappropriate for data sets with complete-data degrees of freedom less than 484.

Barnard and Rubin (1999) recommend the use of adjusted degrees of freedom

$$v_m^* = \left[\frac{1}{v_m} + \frac{1}{\hat{v}_{obs}}\right]^{-1}$$

where $\hat{v}_{obs} = (1-\gamma)\, v_0(v_0+1)/(v_0+3)$ and $\gamma = (1+m^{-1})B/T$.

If you specify the complete-data degrees of freedom v_0 with the EDF= option, the MIANALYZE procedure uses the adjusted degrees of freedom, v_m^*, for inference. Otherwise, the degrees of freedom v_m are used.

Multiple Imputation Efficiency

The relative efficiency (RE) of using the finite m imputation estimator, rather than using an infinite number for the fully efficient imputation, in units of variance, is approximately a function of m and λ (Rubin 1987, p. 114):

$$\mathrm{RE} = (1 + \frac{\lambda}{m})^{-1}$$

Table 83.2 shows relative efficiencies with different values of m and λ.

Table 83.2 Relative Efficiencies

	λ				
m	**10%**	**20%**	**30%**	**50%**	**70%**
3	0.9677	0.9375	0.9091	0.8571	0.8108
5	0.9804	0.9615	0.9434	0.9091	0.8772
10	0.9901	0.9804	0.9709	0.9524	0.9346
20	0.9950	0.9901	0.9852	0.9756	0.9662

The table shows that for situations with little missing information, only a small number of imputations are necessary. In practice, the number of imputations needed can be informally verified by replicating sets of m imputations and checking whether the estimates are stable between sets (Horton and Lipsitz 2001, p. 246).

Multivariate Inferences

Multivariate inference based on Wald tests can be done with m imputed data sets. The approach is a generalization of the approach taken in the univariate case (Rubin 1987, p. 137; Schafer 1997, p. 113). Suppose that $\hat{\mathbf{Q}}_i$ and $\hat{\mathbf{W}}_i$ are the point and covariance matrix estimates for a p-dimensional parameter $\mathbf{Q}$ (such as a multivariate mean) from the $i\,\mathrm{th}$ imputed data set, $i = 1, 2, \ldots, m$. Then the combined point estimate for $\mathbf{Q}$ from the multiple imputation is the average of the m complete-data estimates:

$$\overline{\mathbf{Q}} = \frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{Q}}_i$$

Suppose that $\overline{\mathbf{W}}$ is the within-imputation covariance matrix, which is the average of the m complete-data estimates:

$$\overline{\mathbf{W}} = \frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{W}}_i$$

And suppose that $\mathbf{B}$ is the between-imputation covariance matrix:

$$\mathbf{B} = \frac{1}{m-1}\sum_{i=1}^{m}(\hat{\mathbf{Q}}_i - \overline{\mathbf{Q}})(\hat{\mathbf{Q}}_i - \overline{\mathbf{Q}})'$$

Then the covariance matrix associated with $\overline{\mathbf{Q}}$ is the total covariance matrix

$$\mathbf{T}_0 = \overline{\mathbf{W}} + (1 + \frac{1}{m})\mathbf{B}$$

The natural multivariate extension of the t statistic used in the univariate case is the F statistic

$$F_0 = (\mathbf{Q} - \overline{\mathbf{Q}})'\mathbf{T}_0^{-1}(\mathbf{Q} - \overline{\mathbf{Q}})/p$$

with degrees of freedom p and

$$v = (m-1)(1 + 1/r)^2$$

where

$$r = (1 + \frac{1}{m})\,\mathrm{trace}(\mathbf{B}\overline{\mathbf{W}}^{-1})/p$$

is an average relative increase in variance due to nonresponse (Rubin 1987, p. 137; Schafer 1997, p. 114).

However, the reference distribution of the statistic F_0 is not easily derived. Especially for small m, the between-imputation covariance matrix $\mathbf{B}$ is unstable and does not have full rank for $m \leq p$ (Schafer 1997, p. 113).

One solution is to make an additional assumption that the population between-imputation and within-imputation covariance matrices are proportional to each other (Schafer 1997, p. 113). This assumption

implies that the fractions of missing information for all components of $\mathbf{Q}$ are equal. Under this assumption, a more stable estimate of the total covariance matrix is

$$\mathbf{T} = (1 + r)\overline{\mathbf{W}}$$

With the total covariance matrix $\mathbf{T}$, the F statistic (Rubin 1987, p. 137)

$$F = (\mathbf{Q} - \overline{\mathbf{Q}})'\mathbf{T}^{-1}(\mathbf{Q} - \overline{\mathbf{Q}})/p$$

has an F distribution with degrees of freedom p and v_1, where

$$v_1 = \frac{1}{2}(p + 1)(m - 1)(1 + \frac{1}{r})^2$$

For $t = p(m - 1) \leq 4$, PROC MIANALYZE uses the degrees of freedom v_1 in the analysis. For $t = p(m - 1) > 4$, PROC MIANALYZE uses v_2, a better approximation of the degrees of freedom given by Li, Raghunathan, and Rubin (1991):

$$v_2 = 4 + (t - 4)\left[1 + \frac{1}{r}(1 - \frac{2}{t})\right]^2$$

Testing Linear Hypotheses about the Parameters

Linear hypotheses for parameters $\boldsymbol{\beta}$ are expressed in matrix form as

$$H_0 : \mathbf{L}\boldsymbol{\beta} = \mathbf{c}$$

where $\mathbf{L}$ is a matrix of coefficients for the linear hypotheses and $\mathbf{c}$ is a vector of constants.

Suppose that $\hat{\mathbf{Q}}_i$ and $\hat{\mathbf{U}}_i$ are the point and covariance matrix estimates, respectively, for a p-dimensional parameter $\mathbf{Q}$ from the $i\,\mathrm{th}$ imputed data set, i=1, 2, . . . , m. Then for a given matrix $\mathbf{L}$, the point and covariance matrix estimates for the linear functions $\mathbf{LQ}$ in the $i\,\mathrm{th}$ imputed data set are, respectively,

$$\mathbf{L}\hat{\mathbf{Q}}_i$$

$$\mathbf{L}\hat{\mathbf{U}}_i\mathbf{L}'$$

The inferences described in the section "Combining Inferences from Imputed Data Sets" on page 6719 and the section "Multivariate Inferences" on page 6721 are applied to these linear estimates for testing the null hypothesis $H_0 : \mathbf{L}\boldsymbol{\beta} = \mathbf{c}$.

For each TEST statement, the "Test Specification" table displays the $\mathbf{L}$ matrix and the $\mathbf{c}$ vector, the "Variance Information" table displays the between-imputation, within-imputation, and total variances for combining complete-data inferences, and the "Parameter Estimates" table displays a combined estimate and standard error for each linear component.

With the WCOV and BCOV options in the TEST statement, the procedure displays the within-imputation and between-imputation covariance matrices, respectively.

With the TCOV option, the procedure displays the total covariance matrix derived under the assumption that the population between-imputation and within-imputation covariance matrices are proportional to each other.

With the MULT option in the TEST statement, the "Multivariate Inference" table displays an F test for the null hypothesis $\mathbf{L}\boldsymbol{\beta} = \mathbf{c}$ of the linear components.

Examples of the Complete-Data Inferences

For a given parameter of interest, it is not always possible to compute the estimate and associated covariance matrix directly from a SAS procedure. This section describes examples of parameters with their estimates and associated covariance matrices, which provide the input to the MIANALYZE procedure. Some are straightforward, and others require special techniques.

Means

For a population mean vector $\boldsymbol{\mu}$, the usual estimate is the sample mean vector

$$\overline{\mathbf{y}} = \frac{1}{n} \sum \mathbf{y}_i$$

A variance estimate for $\overline{\mathbf{y}}$ is $\frac{1}{n}\mathbf{S}$, where $\mathbf{S}$ is the sample covariance matrix

$$\mathbf{S} = \frac{1}{n-1} \sum (\mathbf{y}_i - \overline{\mathbf{y}})(\mathbf{y}_i - \overline{\mathbf{y}})'$$

These statistics can be computed from a procedure such as PROC CORR. This approach is illustrated in Example 83.2.

Regression Coefficients

Many SAS procedures are available for regression analysis. Among them, PROC REG provides the most general analysis capabilities, and others like PROC LOGISTIC and PROC MIXED provide more specialized analyses.

Some regression procedures, such as REG and LOGISTIC, create an EST type data set that contains both the parameter estimates for the regression coefficients and their associated covariance matrix. You can read an EST type data set in the MIANALYZE procedure with the DATA= option. This approach is illustrated in Example 83.3.

Other procedures, such as GLM, MIXED, and GENMOD, do not generate EST type data sets for regression coefficients. For PROC MIXED and PROC GENMOD, you can use ODS OUTPUT statement to save parameter estimates in a data set and the associated covariance matrix in a separate data set. These data sets are then read in the MIANALYZE procedure with the PARMS= and COVB= options, respectively. This approach is illustrated in Example 83.4 for PROC MIXED and in Example 83.5 for PROC GENMOD.

PROC GLM does not display tables for covariance matrices. However, you can use the ODS OUTPUT statement to save parameter estimates and associated standard errors in a data set and the associated $(\mathbf{X}'\mathbf{X})^{-1}$ matrix in a separate data set. These data sets are then read in the MIANALYZE procedure with the PARMS= and XPXI= options, respectively. This approach is illustrated in Example 83.6.

For univariate inference, only parameter estimates and associated standard errors are needed. You can use the ODS OUTPUT statement to save parameter estimates and associated standard errors in a data set. This data set is then read in the MIANALYZE procedure with the PARMS= option. This approach is illustrated in Example 83.4.

Correlation Coefficients

For the population correlation coefficient ρ, a point estimate is the sample correlation coefficient r. However, for nonzero ρ, the distribution of r is skewed.

The distribution of r can be normalized through Fisher's z transformation

$$z(r) = \frac{1}{2}\log\left(\frac{1+r}{1-r}\right)$$

$z(r)$ is approximately normally distributed with mean $z(\rho)$ and variance $1/(n-3)$.

With a point estimate $\hat{z}$ and an approximate 95% confidence interval (z_1, z_2) for $z(\rho)$, a point estimate $\hat{r}$ and a 95% confidence interval (r_1, r_2) for ρ can be obtained by applying the inverse transformation

$$r = \tanh(z) = \frac{e^{2z}-1}{e^{2z}+1}$$

to $z = \hat{z}, z_1$, and z_2.

This approach is illustrated in Example 83.11.

Ratios of Variable Means

For the ratio μ_1/μ_2 of means for variables Y_1 and Y_2, the point estimate is $\overline{y}_1/\overline{y}_2$, the ratio of the sample means. The Taylor expansion and delta method can be applied to the function y_1/y_2 to obtain the variance estimate (Schafer 1997, p. 196)

$$\frac{1}{n}\left[\left(\frac{\overline{y}_1}{\overline{y}_2^2}\right)^2 s_{22} - 2\left(\frac{\overline{y}_1}{y_2^2}\right)\left(\frac{1}{\overline{y}_2}\right)s_{12} + \left(\frac{1}{\overline{y}_2}\right)^2 s_{11}\right]$$

where s_{11} and s_{22} are the sample variances of Y_1 and Y_2, respectively, and s_{12} is the sample covariance between Y_1 and Y_2.

A ratio of sample means will be approximately unbiased and normally distributed if the coefficient of variation of the denominator (the standard error for the mean divided by the estimated mean) is 10% or less (Cochran 1977, p. 166; Schafer 1997, p. 196).

ODS Table Names

PROC MIANALYZE assigns a name to each table it creates. You must use these names to reference tables when using the Output Delivery System (ODS). These names are listed in Table 83.3. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 83.3 ODS Tables Produced by PROC MIANALYZE

ODS Table Name	Description	Statement	Option
BCov	Between-imputation covariance matrix		BCOV
ModelInfo	Model information		

Table 83.3 *continued*

ODS Table Name	Description	Statement	Option
MultStat	Multivariate inference		MULT
ParameterEstimates	Parameter estimates		
TCov	Total covariance matrix		TCOV
TestBCov	Between-imputation covariance matrix for $\mathbf{L}\boldsymbol{\beta}$	TEST	BCOV
TestMultStat	Multivariate inference for $\mathbf{L}\boldsymbol{\beta}$	TEST	MULT
TestParameterEstimates	Parameter estimates for $\mathbf{L}\boldsymbol{\beta}$	TEST	
TestSpec	Test specification, $\mathbf{L}$ and $\mathbf{c}$	TEST	
TestTCov	Total covariance matrix for $\mathbf{L}\boldsymbol{\beta}$	TEST	TCOV
TestVarianceInfo	Variance information for $\mathbf{L}\boldsymbol{\beta}$	TEST	
TestWCov	Within-imputation covariance matrix for $\mathbf{L}\boldsymbol{\beta}$	TEST	WCOV
VarianceInfo	Variance information		
WCov	Within-imputation covariance matrix		WCOV

Examples: MIANALYZE Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The following statements generate five imputed data sets to be used in this section. The data set Fitness1 was created in the section "Getting Started: MIANALYZE Procedure" on page 6705. For more information about the MI procedure, See Chapter 82, "The MI Procedure."

```
proc mi data=Fitness1 seed=3237851 noprint out=outmi;
   var Oxygen RunTime RunPulse;
run;
```

The Fish data described in the STEPDISC procedure are measurements of 159 fish of seven species caught in Finland's Lake Laengelmaevesi. For each fish, the length, height, and width are measured. See Chapter 117, "The STEPDISC Procedure," for more information.

The Fish2 data set is constructed from the Fish data set and contains two species of fish. Some values have been set to missing, and the resulting data set has a monotone missing pattern in the variables Length, Width, and Species.

The following statements create the Fish2 data set. It contains two species of fish in the Fish data set.

```
*----------------------------Fish2 Data----------------------------*
| The data set contains two species of the fish (Parkki and Perch)   |
| and two measurements: Length and Width.                            |
| Some values have been set to missing, and the resulting data set   |
| has a monotone missing pattern in the variables                    |
| Length, Width, and Species.                                        |
*--------------------------------------------------------------------*;
data Fish2;
   title 'Fish Measurement Data';
```

```
   input Species $ Length Width @@;
   datalines;
Parkki  16.5  2.3265    Parkki  17.4  2.3142    .      19.8   .
Parkki  21.3  2.9181    Parkki  22.4  3.2928    .      23.2  3.2944
Parkki  23.2  3.4104    Parkki  24.1  3.1571    .      25.8  3.6636
Parkki  28.0  4.1440    Parkki  29.0  4.2340    Perch   8.8  1.4080
.       14.7  1.9992    Perch   16.0  2.4320    Perch  17.2  2.6316
Perch   18.5  2.9415    Perch   19.2  3.3216    .      19.4   .
Perch   20.2  3.0502    Perch   20.8  3.0368    Perch  21.0  2.7720
Perch   22.5  3.5550    Perch   22.5  3.3075    .      22.5   .
Perch   22.8  3.5340    .       23.5   .        Perch  23.5  3.5250
Perch   23.5  3.5250    Perch   23.5  3.5250    Perch  23.5  3.9950
.       24.0   .        Perch   24.0  3.6240    Perch  24.2  3.6300
Perch   24.5  3.6260    Perch   25.0  3.7250    .      25.5  3.7230
Perch   25.5  3.8250    Perch   26.2  4.1658    Perch  26.5  3.6835
.       27.0  4.2390    Perch   28.0  4.1440    Perch  28.7  5.1373
.       28.9  4.3350    .       28.9   .        .      28.9  4.5662
Perch   29.4  4.2042    Perch   30.1  4.6354    Perch  31.6  4.7716
Perch   34.0  6.0180    .       36.5  6.3875    .      37.3  7.7957
.       39.0   .        .       38.3   .        Perch  39.4  6.2646
Perch   39.3  6.3666    Perch   41.4  7.4934    Perch  41.4  6.0030
Perch   41.3  7.3514    .       42.3   .        Perch  42.5  7.2250
Perch   42.4  7.4624    Perch   42.5  6.6300    Perch  44.6  6.8684
Perch   45.2  7.2772    Perch   45.5  7.4165    Perch  46.0  8.1420
Perch   46.6  7.5958
;
```

The following statements generate 25 imputed data sets to be used in this section. The default regression method is used to impute missing values in continuous variable Width, and the discriminant function method is used to impute the variable Species.

```
proc mi data=Fish2 seed=1305417 out=outfish2;
   class Species;
   monotone logistic( Species= Length Width);
   var Length Width Species;
run;
```

The Fish3 data set is constructed from the Fish data set and contains three species of fish. Some values have been set to missing, and the resulting data set has an arbitrary missing pattern in the variables Length, Width, and Species.

The following statements create the Fish3 data set. It contains two species of fish in the Fish data set.

```
*-----------------------------Fish3 Data-----------------------------*
| The data set contains three species of the fish                    |
| (Parkki, Perch, and Roach) and two measurements: Length and Width. |
| Some values have been set to missing, and the resulting data set   |
| has an arbitrary missing pattern in the variables                  |
| Length, Width, and Species.                                        |
*--------------------------------------------------------------------*;
data Fish3;
   title 'Fish Measurement Data';
   input Species $ Length Width @@;
   datalines;
Roach   16.2  2.2680    Roach   20.3  2.8217    Roach   21.2   .
```

```
Roach     .   3.1746    Roach   22.2  3.5742    Roach   22.8  3.3516
Roach   23.1  3.3957    .       23.7   .        Roach   24.7  3.7544
Roach   24.3  3.5478    Roach   25.3   .        Roach   25.0  3.3250
Roach   25.0  3.8000    Roach   27.2  3.8352    Roach   26.7  3.6312
Roach   26.8  4.1272    Roach   27.9  3.9060    Roach   29.2  4.4968
Roach   30.6  4.7736    Roach   35.0  5.3550    Parkki  16.5  2.3265
Parkki  17.4   .        Parkki  19.8  2.6730    Parkki  21.3  2.9181
Parkki  22.4  3.2928    Parkki  23.2  3.2944    Parkki  23.2  3.4104
Parkki  24.1  3.1571    .         .   3.6636    Parkki  28.0  4.1440
Parkki  29.0  4.2340    Perch    8.8  1.4080    .       14.7  1.9992
Perch   16.0  2.4320    Perch   17.2  2.6316    Perch   18.5  2.9415
Perch   19.2  3.3216    .       19.4  3.1234    Perch   20.2   .
Perch   20.8  3.0368    Perch   21.0  2.7720    Perch   22.5  3.5550
Perch   22.5  3.3075    Perch   22.5  3.6675    Perch     .   3.5340
Perch   23.5  3.4075    Perch   23.5  3.5250    Perch   23.5  3.5250
.       23.5  3.5250    Perch   23.5  3.9950    Perch   24.0  3.6240
Perch   24.0  3.6240    Perch   24.2  3.6300    Perch   24.5  3.6260
Perch   25.0  3.7250    Perch     .   3.7230    Perch   25.5  3.8250
Perch     .   4.1658    Perch   26.5  3.6835    .       27.0  4.2390
Perch     .   4.1440    Perch   28.7  5.1373    .       28.9  4.3350
Perch   28.9  4.3350    Perch   28.9  4.5662    Perch   29.4  4.2042
Perch   30.1  4.6354    Perch   31.6  4.7716    Perch   34.0  6.0180
Perch   36.5  6.3875    Perch   37.3  7.7957    Perch   39.0   .
Perch   38.3  6.7408    Perch     .   6.2646    .       39.3   .
Perch   41.4  7.4934    Perch   41.4  6.0030    Perch   41.3  7.3514
Perch   42.3  7.1064    Perch   42.5  7.2250    Perch   42.4  7.4624
Perch   42.5  6.6300    Perch   44.6  6.8684    Perch   45.2  7.2772
Perch   45.5  7.4165    Perch   46.0  8.1420    .       46.6  7.5958
;
```

The following statements generate 25 imputed data sets to be used in this section. The default regression method is used to impute missing values in continuous variable Width, and the nominal logistic regression method is used to impute the variable Species.

```
proc mi data=Fish3 seed=30535 out=outfish3;
   class Species;
   fcs logistic ( Species= Length Width / link=glogit);
   var Length Width Species;
run;
```

Example 83.1 through Example 83.7 use different input option combinations to combine parameter estimates computed from different procedures. Example 83.8 combines parameter estimates with classification variables, and Example 83.9 combines nominal logistic regression parameter estimates Example 83.10 shows the use of a TEST statement, and Example 83.11 combines statistics that are not directly derived from procedures.

The MI procedure provides sensitivity analysis for the MAR assumption. Example 83.12 illustrate sensitivity analysis by using the pattern-mixture model approach, and Example 83.13 performs sensitivity analysis by searching and examining the tipping point that reverses the study conclusion.

Example 83.1: Reading Means and Standard Errors from a DATA= Data Set

(View the complete code for this example (mianaex1.sas) in the example repository.)

This example creates an ordinary SAS data set that contains sample means and standard errors computed from imputed data sets. These estimates are then combined to generate valid univariate inferences about the population means.

The following statements use the UNIVARIATE procedure to generate sample means and standard errors for the variables in each imputed data set:

```
proc univariate data=outmi noprint;
   var Oxygen RunTime RunPulse;
   output out=outuni mean=Oxygen RunTime RunPulse
                     stderr=SOxygen SRunTime SRunPulse;
   by _Imputation_;
run;
```

The following statements display the output data set from PROC UNIVARIATE shown in Output 83.1.1:

```
proc print data=outuni (obs=10);
   title 'UNIVARIATE Means and Standard Errors (First 10 Imputations)';
run;
```

Output 83.1.1 UNIVARIATE Output Data Set

UNIVARIATE Means and Standard Errors (First 10 Imputations)

Obs	_Imputation_	Oxygen	RunTime	RunPulse	SOxygen	SRunTime	SRunPulse
1	1	47.0120	10.4441	171.216	0.95984	0.28520	1.59910
2	2	47.2407	10.5040	171.244	0.93540	0.26661	1.75638
3	3	47.4995	10.5922	171.909	1.00766	0.26302	1.85795
4	4	47.1485	10.5279	171.146	0.95439	0.26405	1.75011
5	5	47.0042	10.4913	172.072	0.96528	0.27275	1.84807
6	6	47.2949	10.6135	169.969	0.95669	0.24533	2.06949
7	7	46.9617	10.5080	171.701	0.93181	0.27111	2.28368
8	8	46.9729	10.5222	170.622	0.94561	0.25687	1.69879
9	9	46.9864	10.5545	171.356	0.93664	0.26233	1.71076
10	10	46.9587	10.5518	171.197	0.94270	0.25214	1.89054

The following statements combine the means and standard errors from imputed data sets, The EDF= option requests that the adjusted degrees of freedom be used in the analysis. For sample means based on 31 observations, the complete-data error degrees of freedom is 30.

```
proc mianalyze data=outuni edf=30;
   modeleffects Oxygen RunTime RunPulse;
   stderr SOxygen SRunTime SRunPulse;
run;
```

The “Model Information” table in Output 83.1.2 lists the input data set(s) and the number of imputations.

Output 83.1.2 Model Information

The MIANALYZE Procedure

Model Information	
Data Set	WORK.OUTUNI
Number of Imputations	25

The "Variance Information" table in Output 83.1.3 displays the between-imputation variance, within-imputation variance, and total variance for each univariate inference. It also displays the degrees of freedom for the total variance. The relative increase in variance due to missing values, the fraction of missing information, and the relative efficiency for each imputed variable are also displayed. A detailed description of these statistics is provided in the section "Combining Inferences from Imputed Data Sets" on page 6719 and the section "Multiple Imputation Efficiency" on page 6720.

Output 83.1.3 Variance Information

Variance Information (25 Imputations)							
	Variance						
Parameter	**Between**	**Within**	**Total**	**DF**	**Relative Increase in Variance**	**Fraction Missing Information**	**Relative Efficiency**
Oxygen	0.026098	0.925531	0.952673	27.354	0.029325	0.028556	0.998859
RunTime	0.002938	0.068197	0.071253	26.918	0.044810	0.043035	0.998282
RunPulse	0.598494	3.345356	3.967790	23.196	0.186059	0.158595	0.993696

The "Parameter Estimates" table in Output 83.1.4 displays the estimated mean and corresponding standard error for each variable. The table also displays a 95% confidence interval for the mean and a t statistic with the associated p-value for testing the hypothesis that the mean is equal to the value specified. You can use the THETA0= option to specify the value for the null hypothesis, which is zero by default. The table also displays the minimum and maximum parameter estimates from the imputed data sets.

Output 83.1.4 Parameter Estimates

Parameter Estimates (25 Imputations)							
Parameter	**Estimate**	**Std Error**	**95% Confidence Limits**		**DF**	**Minimum**	**Maximum**
Oxygen	47.084579	0.976050	45.0831	49.0861	27.354	46.850020	47.499541
RunTime	10.549499	0.266933	10.0017	11.0973	26.918	10.428848	10.680797
RunPulse	171.388418	1.991931	167.2697	175.5071	23.196	169.881385	173.125658

Parameter Estimates (25 Imputations)			
Parameter	**Theta0**	**t for H0: Parameter=Theta0**	**Pr > |t|**
Oxygen	0	48.24	<.0001
RunTime	0	39.52	<.0001
RunPulse	0	86.04	<.0001

Note that the results in this example could also have been obtained with the MI procedure.

Example 83.2: Reading Means and Covariance Matrices from a DATA= COV Data Set

(View the complete code for this example (mianaex2.sas) in the example repository.)

This example creates a COV-type data set that contains sample means and covariance matrices computed from imputed data sets. These estimates are then combined to generate valid statistical inferences about the population means.

The following statements use the CORR procedure to generate sample means and a covariance matrix for the variables in each imputed data set:

```
proc corr data=outmi cov nocorr noprint out=outcov(type=cov);
   var Oxygen RunTime RunPulse;
   by _Imputation_;
run;
```

The following statements display (in Output 83.2.1) output sample means and covariance matrices from PROC CORR for the first two imputed data sets:

```
proc print data=outcov(obs=12);
   title 'CORR Means and Covariance Matrices'
         ' (First Two Imputations)';
run;
```

Output 83.2.1 COV Data Set

CORR Means and Covariance Matrices (First Two Imputations)

Obs	_Imputation_	_TYPE_	_NAME_	Oxygen	RunTime	RunPulse
1	1	COV	Oxygen	28.5603	-7.2652	-11.812
2	1	COV	RunTime	-7.2652	2.5214	2.536
3	1	COV	RunPulse	-11.8121	2.5357	79.271
4	1	MEAN		47.0120	10.4441	171.216
5	1	STD		5.3442	1.5879	8.903
6	1	N		31.0000	31.0000	31.000
7	2	COV	Oxygen	27.1240	-6.6761	-10.217
8	2	COV	RunTime	-6.6761	2.2035	2.611
9	2	COV	RunPulse	-10.2170	2.6114	95.631
10	2	MEAN		47.2407	10.5040	171.244
11	2	STD		5.2081	1.4844	9.779
12	2	N		31.0000	31.0000	31.000

Note that the covariance matrices in the data set Outcov are estimated covariance matrices of variables, $V(\mathbf{y})$. The estimated covariance matrix of the sample means is $V(\overline{\mathbf{y}}) = V(\mathbf{y})/n$, where n is the sample size, and is not the same as an estimated covariance matrix for variables.

The following statements combine the results for the imputed data sets, and derive both univariate and multivariate inferences about the means. The EDF= option is specified to request that the adjusted degrees of freedom be used in the analysis. For sample means based on 31 observations, the complete-data error degrees of freedom is 30.

```
proc mianalyze data=outcov edf=30;
   modeleffects Oxygen RunTime RunPulse;
run;
```

The "Variance Information" and "Parameter Estimates" tables display the same results as in Output 83.1.2 and Output 83.1.3, respectively, in Example 83.1.

With the WCOV, BCOV, and TCOV options, as in the following statements, the procedure displays the between-imputation covariance matrix, within-imputation covariance matrix, and total covariance matrix assuming that the between-imputation covariance matrix is proportional to the within-imputation covariance matrix in Output 83.2.2.

```
proc mianalyze data=outcov edf=30 wcov bcov tcov mult;
   modeleffects Oxygen RunTime RunPulse;
run;
```

Output 83.2.2 Covariance Matrices

The MIANALYZE Procedure

Within-Imputation Covariance Matrix			
	Oxygen	**RunTime**	**RunPulse**
Oxygen	0.925531500	-0.215584249	-0.621865880
RunTime	-0.215584249	0.068197432	0.116221427
RunPulse	-0.621865880	0.116221427	3.345355692

Between-Imputation Covariance Matrix			
	Oxygen	**RunTime**	**RunPulse**
Oxygen	0.0260976444	0.0018582126	0.0215847560
RunTime	0.0018582126	0.0029383770	0.0064057721
RunPulse	0.0215847560	0.0064057721	0.5984941329

Total Covariance Matrix			
	Oxygen	**RunTime**	**RunPulse**
Oxygen	1.106311852	-0.257693455	-0.743332446
RunTime	-0.257693455	0.081518163	0.138922492
RunPulse	-0.743332446	0.138922492	3.998790592

With the MULT option, the procedure assumes that the between-imputation covariance matrix is proportional to the within-imputation covariance matrix and displays a multivariate inference for all the parameters taken jointly.

Output 83.2.3 Multivariate Inference

Multivariate Inference Assuming Proportionality of Between/Within Covariance Matrices				
Avg Relative Increase in Variance	**Num DF**	**Den DF**	**F for H0: Parameter=Theta0**	**Pr > F**
0.195326	3	2433.6	13296.2	<.0001

The "Multivariate Inference" table in Output 83.2.3 shows a significant p-value for the null hypothesis that

the population means are all equal to zero.

Example 83.3: Reading Regression Results from a DATA= EST Data Set

(View the complete code for this example (mianaex3.sas) in the example repository.)

This example creates an EST-type data set that contains regression coefficients and their corresponding covariance matrices computed from imputed data sets. These estimates are then combined to generate valid statistical inferences about the regression model.

The following statements use the REG procedure to generate regression coefficients for each imputed data set:

```
proc reg data=outmi outest=outreg covout noprint;
   model Oxygen= RunTime RunPulse;
   by _Imputation_;
run;
```

The following statements display (in Output 83.3.1) output regression coefficients and their covariance matrices from PROC REG for the first two imputed data sets:

```
proc print data=outreg(obs=8);
   var _Imputation_ _Type_ _Name_
      Intercept RunTime RunPulse;
   title 'REG Model Coefficients and Covariance Matrices'
         ' (First Two Imputations)';
run;
```

Output 83.3.1 EST-Type Data Set

REG Model Coefficients and Covariance Matrices (First Two Imputations)

Obs	_Imputation_	_TYPE_	_NAME_	Intercept	RunTime	RunPulse
1	1	PARMS		86.544	-2.82231	-0.05873
2	1	COV	Intercept	100.145	-0.53519	-0.55077
3	1	COV	RunTime	-0.535	0.10774	-0.00345
4	1	COV	RunPulse	-0.551	-0.00345	0.00343
5	2	PARMS		83.021	-3.00023	-0.02491
6	2	COV	Intercept	79.032	-0.66765	-0.41918
7	2	COV	RunTime	-0.668	0.11456	-0.00313
8	2	COV	RunPulse	-0.419	-0.00313	0.00264

The following statements combine the results for the imputed data sets. The EDF= option is specified to request that the adjusted degrees of freedom be used in the analysis. For a regression model with three independent variables (including the Intercept) and 31 observations, the complete-data error degrees of freedom is 28.

```
proc mianalyze data=outreg edf=28;
   modeleffects Intercept RunTime RunPulse;
run;
```

Output 83.3.2 Variance Information

The MIANALYZE Procedure

Variance Information (25 Imputations)

Parameter	Variance Between	Variance Within	Variance Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Intercept	22.485821	75.413875	98.799129	19.102	0.310092	0.240234	0.990482
RunTime	0.021126	0.124930	0.146902	21.823	0.175870	0.151147	0.993990
RunPulse	0.000656	0.002622	0.003304	20.042	0.260376	0.209393	0.991694

The "Variance Information" table in Output 83.3.2 displays the between-imputation, within-imputation, and total variances for combining complete-data inferences.

The "Parameter Estimates" table in Output 83.3.3 displays the estimated mean and standard error of the regression coefficients. The inferences are based on the t distribution. The table also displays a 95% mean confidence interval and a t test with the associated p-value for the hypothesis that the regression coefficient is equal to zero. Since the p-value for RunPulse is 0.1812, this variable can be removed from the regression model.

Output 83.3.3 Parameter Estimates

Parameter Estimates (25 Imputations)

Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	92.700420	9.939775	71.90376	113.4971	19.102	83.020730	100.839807	0	9.33	<.0001
RunTime	-3.030325	0.383278	-3.82557	-2.2351	21.823	-3.280042	-2.754668	0	-7.91	<.0001
RunPulse	-0.079621	0.057482	-0.19951	0.0403	20.042	-0.135862	-0.024910	0	-1.39	0.1812

Example 83.4: Reading Mixed Model Results from PARMS= and COVB= Data Sets

(View the complete code for this example (mianaex4.sas) in the example repository.)

This example creates data sets that contains parameter estimates and covariance matrices computed by a mixed model analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the parameters.

The following PROC MIXED statements generate the fixed-effect parameter estimates and covariance matrix for each imputed data set:

```
ods select none;
proc mixed data=outmi;
   model Oxygen= RunTime RunPulse RunTime*RunPulse/solution covb;
   by _Imputation_;
   ods output SolutionF=mixparms CovB=mixcovb;
run;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The ODS OUTPUT statement creates two SAS data sets, named mixparms and mixcovb, from the two ODS tables. The following statements display (in Output 83.4.1) output parameter estimates for the first two imputed data sets:

```
proc print data=mixparms (obs=8);
   var _Imputation_ Effect Estimate StdErr;
   title 'MIXED Model Coefficients (First Two Imputations)';
run;
```

Output 83.4.1 PROC MIXED Model Coefficients

MIXED Model Coefficients (First Two Imputations)

Obs	_Imputation_	Effect	Estimate	StdErr
1	1	Intercept	148.09	81.5231
2	1	RunTime	-8.8115	7.8794
3	1	RunPulse	-0.4123	0.4684
4	1	RunTime*RunPulse	0.03437	0.04517
5	2	Intercept	64.3607	64.6034
6	2	RunTime	-1.1270	6.4307
7	2	RunPulse	0.08160	0.3688
8	2	RunTime*RunPulse	-0.01069	0.03664

The following statements display (in Output 83.4.2) the output covariance matrices associated with the parameter estimates from PROC MIXED for the first two imputed data sets:

```
proc print data=mixcovb (obs=8);
   var _Imputation_ Row Effect Col1 Col2 Col3 Col4;
   title 'Covariance Matrices (First Two Imputations)';
run;
```

Output 83.4.2 PROC MIXED Covariance Matrices

Covariance Matrices (First Two Imputations)

Obs	_Imputation_	Row	Effect	Col1	Col2	Col3	Col4
1	1	1	Intercept	6646.01	-637.40	-38.1515	3.6542
2	1	2	RunTime	-637.40	62.0842	3.6548	-0.3556
3	1	3	RunPulse	-38.1515	3.6548	0.2194	-0.02099
4	1	4	RunTime*RunPulse	3.6542	-0.3556	-0.02099	0.002040
5	2	1	Intercept	4173.59	-411.46	-23.7889	2.3441
6	2	2	RunTime	-411.46	41.3545	2.3414	-0.2353
7	2	3	RunPulse	-23.7889	2.3414	0.1360	-0.01338
8	2	4	RunTime*RunPulse	2.3441	-0.2353	-0.01338	0.001343

Note that the variables Col1, Col2, Col3, and Col4 are used to identify the effects Intercept, RunTime, RunPulse, and RunTime*RunPulse, respectively, through the variable Row.

For univariate inference, only parameter estimates and their associated standard errors are needed. The following statements use the MIANALYZE procedure with the input PARMS= data set to produce univariate results:

```
proc mianalyze parms=mixparms edf=28;
   modeleffects Intercept RunTime RunPulse RunTime*RunPulse;
run;
```

The "Variance Information" table in Output 83.4.3 displays the between-imputation, within-imputation, and total variances for combining complete-data inferences.

Output 83.4.3 Variance Information

The MIANALYZE Procedure

Variance Information (25 Imputations)							
	Variance						
Parameter	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Intercept	1457.114815	4592.351134	6107.750541	18.748	0.329983	0.251939	0.990023
RunTime	12.937905	43.960426	57.415848	19.175	0.306080	0.237831	0.990576
RunPulse	0.045956	0.152072	0.199867	19.026	0.314288	0.242732	0.990384
RunTime*RunPulse	0.000409	0.001447	0.001872	19.398	0.293964	0.230483	0.990865

The "Parameter Estimates" table in Output 83.4.4 displays the estimated mean and standard error of the regression coefficients.

Output 83.4.4 Parameter Estimates

Parameter Estimates (25 Imputations)							
Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum
Intercept	155.866055	78.152099	-7.8574	319.5895	18.748	64.360719	207.413571
RunTime	-9.265145	7.577325	-25.1149	6.5846	19.175	-14.937943	-1.127010
RunPulse	-0.443201	0.447065	-1.3788	0.4924	19.026	-0.738608	0.081597
RunTime*RunPulse	0.035841	0.043272	-0.0546	0.1263	19.398	-0.010690	0.068289

Parameter Estimates (25 Imputations)			
Parameter	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	0	1.99	0.0609
RunTime	0	-1.22	0.2362
RunPulse	0	-0.99	0.3340
RunTime*RunPulse	0	0.83	0.4176

Since each covariance matrix contains variables Row, Col1, Col2, Col3, and Col4 for parameters, the EFFECTVAR=ROWCOL option is needed when you specify the COVB= option. The following statements illustrate the use of the MIANALYZE procedure with input PARMS= and COVB(EFFECTVAR=ROWCOL)= data sets:

```
proc mianalyze parms=mixparms edf=28
               covb(effectvar=rowcol)=mixcovb;
   modeleffects Intercept RunTime RunPulse RunTime*RunPulse;
run;
```

Example 83.5: Reading Generalized Linear Model Results

(View the complete code for this example (mianaex5.sas) in the example repository.)

This example creates data sets that contains parameter estimates and corresponding covariance matrices computed by a generalized linear model analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the model parameters.

The following statements use PROC GENMOD to generate the parameter estimates and covariance matrix for each imputed data set:

```
ods select none;
proc genmod data=outmi;
   model Oxygen= RunTime RunPulse/covb;
   by _Imputation_;
   ods output ParameterEstimates=gmparms
              ParmInfo=gmpinfo
              CovB=gmcovb;
run;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The following statements print (in Output 83.5.1) the output parameter estimates and covariance matrix from PROC GENMOD for the first two imputed data sets:

```
proc print data=gmparms (obs=8);
   var _Imputation_ Parameter Estimate StdErr;
   title 'GENMOD Model Coefficients (First Two Imputations)';
run;
```

Output 83.5.1 PROC GENMOD Model Coefficients

GENMOD Model Coefficients (First Two Imputations)

Obs	_Imputation_	Parameter	Estimate	StdErr
1	1	Intercept	86.5440	9.5107
2	1	RunTime	-2.8223	0.3120
3	1	RunPulse	-0.0587	0.0556
4	1	Scale	2.6692	0.3390
5	2	Intercept	83.0207	8.4489
6	2	RunTime	-3.0002	0.3217
7	2	RunPulse	-0.0249	0.0488
8	2	Scale	2.5727	0.3267

The following statements display the parameter information table in Output 83.5.2. The table identifies parameter names used in the covariance matrices. The parameters Prm1, Prm2, and Prm3 are used for the effects Intercept, RunTime, and RunPulse, respectively, in each covariance matrix.

```
proc print data=gmpinfo (obs=6);
   title 'GENMOD Parameter Information (First Two Imputations)';
run;
```

Output 83.5.2 PROC GENMOD Model Information

GENMOD Parameter Information (First Two Imputations)

Obs	_Imputation_	Parameter	Effect
1	1	Prm1	Intercept
2	1	Prm2	RunTime
3	1	Prm3	RunPulse
4	2	Prm1	Intercept
5	2	Prm2	RunTime
6	2	Prm3	RunPulse

The following statements display (in Output 83.5.3) the output covariance matrices from PROC GENMOD for the first two imputed data sets. Note that the GENMOD procedure computes maximum likelihood estimates for each covariance matrix.

```
proc print data=gmcovb (obs=8);
   var _Imputation_ RowName Prm1 Prm2 Prm3;
   title 'GENMOD Covariance Matrices (First Two Imputations)';
run;
```

Output 83.5.3 PROC GENMOD Covariance Matrices

GENMOD Covariance Matrices (First Two Imputations)

Obs	_Imputation_	RowName	Prm1	Prm2	Prm3
1	1	Prm1	90.453923	-0.483394	-0.497473
2	1	Prm2	-0.483394	0.0973159	-0.003113
3	1	Prm3	-0.497473	-0.003113	0.0030954
4	1	Scale	3.954E-15	-9.59E-17	-1.59E-17
5	2	Prm1	71.383332	-0.603037	-0.378616
6	2	Prm2	-0.603037	0.1034766	-0.002826
7	2	Prm3	-0.378616	-0.002826	0.0023843
8	2	Scale	1.811E-14	1.091E-16	-1.06E-16

The following statements use the MIANALYZE procedure with input PARMS=, PARMINFO=, and COVB= data sets:

```
proc mianalyze parms=gmparms covb=gmcovb parminfo=gmpinfo;
   modeleffects Intercept RunTime RunPulse;
run;
```

Since the GENMOD procedure computes maximum likelihood estimates for the covariance matrix, the EDF= option is not used. The “Parameter Estimates” table in Output 83.5.4 displays the estimated mean and standard error of the regression coefficients.

Output 83.5.4 Parameter Estimates

The MIANALYZE Procedure

Parameter Estimates (25 Imputations)

Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	92.700420	9.565616	73.89020	111.5106	367.43	83.020730	100.839807	0	9.69	<.0001
RunTime	-3.030325	0.367167	-3.75093	-2.3097	903.54	-3.280042	-2.754668	0	-8.25	<.0001
RunPulse	-0.079621	0.055231	-0.18815	0.0289	479.31	-0.135862	-0.024910	0	-1.44	0.1501

The resulting parameter estimates are identical to the estimates in Output 83.3.3 in Example 83.3. However, the standard errors are slightly different because in this example, maximum likelihood estimates for the standard errors are combined without the EDF= option, whereas in Example 83.3, unbiased estimates for the standard errors are combined with the EDF= option.

Example 83.6: Reading GLM Results from PARMS= and XPXI= Data Sets

(View the complete code for this example (mianaex6.sas) in the example repository.)

This example creates data sets that contains parameter estimates and corresponding $(\mathbf{X}'\mathbf{X})^{-1}$ matrices computed by a general linear model analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the model parameters.

The following statements use PROC GLM to generate the parameter estimates and $(\mathbf{X}'\mathbf{X})^{-1}$ matrix for each imputed data set:

```
ods select none;
proc glm data=outmi;
   model Oxygen= RunTime RunPulse/inverse;
   by _Imputation_;
   ods output ParameterEstimates=glmparms
              InvXPX=glmxpxi;
quit;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The following statements display (in Output 83.6.1) the output parameter estimates and standard errors from PROC GLM for the first two imputed data sets:

```
proc print data=glmparms (obs=6);
   var _Imputation_ Parameter Estimate StdErr;
   title 'GLM Model Coefficients (First Two Imputations)';
run;
```

Output 83.6.1 PROC GLM Model Coefficients

GLM Model Coefficients (First Two Imputations)

Obs	_Imputation_	Parameter	Estimate	StdErr
1	1	Intercept	86.544034	10.00726811
2	1	RunTime	-2.822311	0.32824165
3	1	RunPulse	-0.058729	0.05854109
4	2	Intercept	83.020730	8.88996885
5	2	RunTime	-3.000229	0.33847204
6	2	RunPulse	-0.024910	0.05137859

The following statements display (in Output 83.6.2) $(\mathbf{X}'\mathbf{X})^{-1}$ matrices from PROC GLM for the first two imputed data sets:

```
proc print data=glmxpxi (obs=8);
   var _Imputation_ Parameter Intercept RunTime RunPulse;
   title 'GLM X''X Inverse Matrices (First Two Imputations)';
run;
```

Output 83.6.2 PROC GLM $(\mathbf{X}'\mathbf{X})^{-1}$ Matrices

GLM X'X Inverse Matrices (First Two Imputations)

Obs	_Imputation_	Parameter	Intercept	RunTime	RunPulse
1	1	Intercept	12.696250656	-0.067849956	-0.069826009
2	1	RunTime	-0.067849956	0.0136594055	-0.000436938
3	1	RunPulse	-0.069826009	-0.000436938	0.0004344762
4	1	Oxygen	86.544033929	-2.822310769	-0.058729234
5	2	Intercept	10.784620785	-0.091107072	-0.057201387
6	2	RunTime	-0.091107072	0.0156332765	-0.000426902
7	2	RunPulse	-0.057201387	-0.000426902	0.0003602208
8	2	Oxygen	83.020730343	-3.000228818	-0.024910305

The standard errors for the estimates in the output Glmparms data set are needed to create the covariance matrix from the $(\mathbf{X}'\mathbf{X})^{-1}$ matrix. The following statements use the MIANALYZE procedure with input PARMS= and XPXI= data sets to produce the same results as displayed in Output 83.3.2 and Output 83.3.3 in Example 83.3:

```
proc mianalyze parms=glmparms xpxi=glmxpxi edf=28;
   modeleffects Intercept RunTime RunPulse;
run;
```

Example 83.7: Reading Logistic Model Results from a PARMS= Data Set

(View the complete code for this example (mianaex7.sas) in the example repository.)

This example creates data sets that contains parameter estimates computed by a logistic regression analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the model parameters.

The following statements use PROC LOGISTIC to generate the parameter estimates for each imputed data set:

```
ods select none;
proc logistic data=outfish2;
   class Species;
   model Species= Length Width / covb;
   by _Imputation_;
   ods output ParameterEstimates=lgsparms;
run;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The following statements display (in Output 83.7.1) the output logistic regression coefficients from PROC LOGISTIC for the first two imputed data sets:

```
proc print data=lgsparms (obs=6);
   title 'LOGISTIC Model Coefficients (First Two Imputations)';
run;
```

Output 83.7.1 PROC LOGISTIC Model Coefficients

LOGISTIC Model Coefficients (First Two Imputations)

Obs	_Imputation_	Variable	DF	Estimate	StdErr	WaldChiSq	ProbChiSq	_ESTTYPE_
1	1	Intercept	1	0.1637	1.8405	0.0079	0.9291	MLE
2	1	Length	1	1.4543	0.5167	7.9231	0.0049	MLE
3	1	Width	1	-10.2950	3.4860	8.7216	0.0031	MLE
4	2	Intercept	1	0.6473	1.9003	0.1160	0.7334	MLE
5	2	Length	1	1.2831	0.4778	7.2123	0.0072	MLE
6	2	Width	1	-9.2991	3.2187	8.3469	0.0039	MLE

The following statements use the MIANALYZE procedure with input PARMS= data set:

```
proc mianalyze parms=lgsparms;
   modeleffects Intercept Length Width;
run;
```

The “Variance Information” table in Output 83.7.2 displays the between-imputation, within-imputation, and total variances for combining complete-data inferences.

Output 83.7.2 Variance Information

The MIANALYZE Procedure

Variance Information (25 Imputations)							
	Variance						
Parameter	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Intercept	0.602599	3.034752	3.661455	819.21	0.206509	0.173178	0.993121
Length	0.077877	0.188227	0.269219	265.18	0.430288	0.306054	0.987906
Width	3.757896	8.382542	12.290754	237.36	0.466232	0.323655	0.987219

The "Parameter Estimates" table in Output 83.7.3 displays the combined parameter estimates with associated standard errors.

Output 83.7.3 Parameter Estimates

Parameter Estimates (25 Imputations)										
Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	-0.130560	1.913493	-3.8865	3.62537	819.21	-2.321742	0.827356	0	-0.07	0.9456
Length	1.169782	0.518863	0.1482	2.19140	265.18	0.385118	1.554454	0	2.25	0.0250
Width	-8.284998	3.505817	-15.1915	-1.37851	237.36	-11.149787	-2.878900	0	-2.36	0.0189

Example 83.8: Reading Mixed Model Results with Classification Covariates

(View the complete code for this example (mianaex8.sas) in the example repository.)

This example creates data sets that contains parameter estimates and corresponding covariance matrices with classification variables computed by a mixed regression model analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the model parameters.

The following statements use PROC MIXED to generate the parameter estimates and covariance matrix for each imputed data set:

```
ods select none;
proc mixed data=outfish2;
   class Species;
   model Length= Species Width/ solution;
   by _Imputation_;
   ods output SolutionF=mxparms;
run;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The following statements display (in Output 83.8.1) the output mixed model coefficients from PROC MIXED for the first two imputed data sets:

```
proc print data=mxparms (obs=8);
   var _Imputation_ Effect Species Estimate StdErr;
   title 'MIXED Model Coefficients (First Two Imputations)';
run;
```

Output 83.8.1 PROC MIXED Model Coefficients

MIXED Model Coefficients (First Two Imputations)

Obs	_Imputation_	Effect	Species	Estimate	StdErr
1	1	Intercept		4.5106	0.8244
2	1	Species	Parkki	1.5774	0.7020
3	1	Species	Perch	0	.
4	1	Width		5.2585	0.1599
5	2	Intercept		4.5250	0.8771
6	2	Species	Parkki	1.4885	0.7693
7	2	Species	Perch	0	.
8	2	Width		5.2389	0.1701

The following statements use the MIANALYZE procedure with an input PARMS= data set:

```
proc mianalyze parms(classvar=full)=mxparms;
   class Species;
   modeleffects Intercept Species Width;
run;
```

The "Variance Information" table in Output 83.8.2 displays the between-imputation, within-imputation, and total variances for combining complete-data inferences.

Output 83.8.2 Variance Information

The MIANALYZE Procedure

Variance Information (25 Imputations)

Parameter	Species	Variance Between	Variance Within	Variance Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Intercept		0.065665	0.668667	0.736959	2794.9	0.102131	0.093316	0.996281
Species	Parkki	0.077291	0.525640	0.606023	1364.1	0.152924	0.133909	0.994672
Species	Perch	0	.	.	.	.	.	.
Width		0.002276	0.025876	0.028243	3416	0.091488	0.084356	0.996637

The "Parameter Estimates" table in Output 83.8.3 displays the combined parameter estimates with associated standard errors.

Output 83.8.3 Parameter Estimates

Parameter Estimates (25 Imputations)								
Parameter	**Species**	**Estimate**	**Std Error**	**95% Confidence Limits**		**DF**	**Minimum**	**Maximum**
Intercept		4.519259	0.858463	2.83597	6.202545	2794.9	4.013624	5.162463
Species	Parkki	1.277902	0.778475	-0.24924	2.805039	1364.1	0.647898	1.710559
Species	Perch	0	.	.	.	.	0	0
Width		5.284285	0.168056	4.95478	5.613786	3416	5.206516	5.369231

Parameter Estimates (25 Imputations)				
Parameter	**Species**	**Theta0**	**t for H0: Parameter=Theta0**	**Pr > |t|**
Intercept		0	5.26	<.0001
Species	Parkki	0	1.64	0.1009
Species	Perch	0	.	.
Width		0	31.44	<.0001

Example 83.9: Reading Nominal Logistic Model Results

(View the complete code for this example (mianaex9.sas) in the example repository.)

This example creates data sets to contain parameter estimates that are computed by a nominal logistic regression analysis for a set of imputed data sets. These estimates are then combined to generate valid statistical inferences about the model parameters.

The following statements use PROC LOGISTIC to generate the parameter estimates and covariance matrix for each imputed data set:

```
ods select none;
proc logistic data=outfish3;
   class Species;
   model Species= Length Width / link=glogit covb;
   by _Imputation_;
   ods output ParameterEstimates=lgsparms
              CovB=lgscovb;
run;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The following statements display (in Output 83.9.1) the output logistic regression coefficients from PROC LOGISTIC for the first two imputed data sets:

```
proc print data=lgsparms (obs=12);
   title 'LOGISTIC Model Coefficients (First Two Imputations)';
run;
```

Output 83.9.1 PROC LOGISTIC Model Coefficients

LOGISTIC Model Coefficients (First Two Imputations)

Obs	_Imputation_	Variable	Response	DF	Estimate	StdErr	WaldChiSq	ProbChiSq	_ESTTYPE_
1	1	Intercept	Parkki	1	1.7737	1.7712	1.0029	0.3166	MLE
2	1	Intercept	Perch	1	1.1036	1.3426	0.6757	0.4111	MLE
3	1	Length	Parkki	1	-0.0353	0.2700	0.0171	0.8960	MLE
4	1	Length	Perch	1	-0.8560	0.2635	10.5529	0.0012	MLE
5	1	Width	Parkki	1	-0.3784	1.6650	0.0517	0.8202	MLE
6	1	Width	Perch	1	5.6213	1.6333	11.8455	0.0006	MLE
7	2	Intercept	Parkki	1	2.3507	1.7930	1.7188	0.1898	MLE
8	2	Intercept	Perch	1	0.6321	1.3370	0.2235	0.6364	MLE
9	2	Length	Parkki	1	-0.3479	0.2460	2.0004	0.1573	MLE
10	2	Length	Perch	1	-0.6108	0.2130	8.2274	0.0041	MLE
11	2	Width	Parkki	1	1.5786	1.5300	1.0645	0.3022	MLE
12	2	Width	Perch	1	4.1610	1.3110	10.0734	0.0015	MLE

The following statements display the covariance matrices that are associated with parameter estimates derived from the first two imputations in Output 83.9.2:

```
proc print data=lgscovb (obs=12);
   title 'LOGISTIC Model Covariance Matrices (First Two Imputations)';
run;
```

Output 83.9.2 PROC LOGISTIC Covariance Matrices

LOGISTIC Model Covariance Matrices (First Two Imputations)

Obs	_Imputation_	Parameter	Intercept_Parkki	Intercept_Perch	Length_Parkki
1	1	Intercept_Parkki	3.137016	1.150943	-0.25136
2	1	Intercept_Perch	1.150943	1.80259	-0.12448
3	1	Length_Parkki	-0.25136	-0.12448	0.072903
4	1	Length_Perch	-0.11416	-0.16709	0.028705
5	1	Width_Parkki	0.857307	0.557913	-0.43386
6	1	Width_Perch	0.484917	0.676397	-0.16464
7	2	Intercept_Parkki	3.214747	1.25981	-0.19425
8	2	Intercept_Perch	1.25981	1.787564	-0.11454
9	2	Length_Parkki	-0.19425	-0.11454	0.060501
10	2	Length_Perch	-0.10076	-0.13446	0.029263
11	2	Width_Parkki	0.436385	0.460885	-0.35903
12	2	Width_Perch	0.365388	0.463036	-0.17062

Obs	Length_Perch	Width_Parkki	Width_Perch
1	-0.11416	0.857307	0.484917
2	-0.16709	0.557913	0.676397
3	0.028705	-0.43386	-0.16464
4	0.069437	-0.16666	-0.42309
5	-0.16666	2.77239	1.00217
6	-0.42309	1.00217	2.66758
7	-0.10076	0.436385	0.365388
8	-0.13446	0.460885	0.463036
9	0.029263	-0.35903	-0.17062
10	0.04535	-0.17499	-0.27173
11	-0.17499	2.34089	1.081586
12	-0.27173	1.081586	1.718756

The following statements use the MIANALYZE procedure with the input PARMS= and COVB= data sets:

```
proc mianalyze parms(link=glogit)=lgsparms
               covb(effectvar=stacking)=lgscovb
               mult;
   modeleffects Intercept Length Width;
run;
```

The “Variance Information” table in Output 83.9.3 displays the between-imputation, within-imputation, and total variances for combining complete-data inferences.

Output 83.9.3 Variance Information

The MIANALYZE Procedure

Variance Information (25 Imputations)								
		Variance						
Parameter	Response	Between	Within	Total	DF	Relative Increase in Variance	Fraction Missing Information	Relative Efficiency
Intercept	Parkki	0.539659	3.540706	4.101951	1282	0.158512	0.138167	0.994504
Intercept	Perch	0.182814	1.502203	1.692330	1901.5	0.126565	0.113278	0.995489
Length	Parkki	0.069216	0.089623	0.161607	120.96	0.803187	0.454374	0.982149
Length	Perch	0.027025	0.047967	0.076073	175.82	0.585945	0.376513	0.985163
Width	Parkki	3.131748	3.792307	7.049325	112.43	0.858849	0.471354	0.981495
Width	Perch	1.079472	1.816671	2.939322	164.52	0.617972	0.389321	0.984666

The "Parameter Estimates" table in Output 83.9.4 displays the combined parameter estimates and their associated standard errors.

Output 83.9.4 Parameter Estimates

Parameter Estimates (25 Imputations)								
Parameter	Response	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum
Intercept	Parkki	1.247968	2.025327	-2.72535	5.22129	1282	-0.450300	2.607579
Intercept	Perch	0.466573	1.300896	-2.08476	3.01791	1901.5	-0.536319	1.129313
Length	Parkki	0.285276	0.402004	-0.51060	1.08115	120.96	-0.347887	0.748875
Length	Perch	-0.566649	0.275813	-1.11098	-0.02232	175.82	-0.856010	-0.254865
Width	Parkki	-2.531763	2.655056	-7.79220	2.72867	112.43	-5.690349	1.578570
Width	Perch	3.835827	1.714445	0.45068	7.22098	164.52	1.818114	5.621285

Parameter Estimates (25 Imputations)				
Parameter	Response	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept	Parkki	0	0.62	0.5379
Intercept	Perch	0	0.36	0.7199
Length	Parkki	0	0.71	0.4793
Length	Perch	0	-2.05	0.0414
Width	Parkki	0	-0.95	0.3424
Width	Perch	0	2.24	0.0266

The "Multivariate Inference" table in Output 83.9.5 displays multivariate inference for the parameters assuming proportionality of the between-imputation and within-imputation covariance matrices.

Output 83.9.5 Multivariate Inference

Multivariate Inference Assuming Proportionality of Between/Within Covariance Matrices				
Avg Relative Increase in Variance	Num DF	Den DF	F for H0: Parameter=Theta0	Pr > F
0.321721	6	2317.5	3.14	0.0046

Example 83.10: Using a TEST Statement

(View the complete code for this example (mianax10.sas) in the example repository.)

This example creates a DATA=EST data set to contain regression coefficients and their corresponding covariance matrices that are computed from imputed data sets. These estimates are then combined to generate valid statistical inferences about the regression model. A TEST statement is used to test linear hypotheses about the parameters.

The following statements use the REG procedure to generate regression coefficients for each imputed data set:

```
proc reg data=outmi outest=outreg covout noprint;
   model Oxygen= RunTime RunPulse;
   by _Imputation_;
run;
```

The following statements combine the results for the imputed data sets. A TEST statement is used to test linear hypotheses of INTERCEPT=0 and RUNTIME=RUNPULSE.

```
proc mianalyze data=outreg edf=28;
   modeleffects Intercept RunTime RunPulse;
   test Intercept, RunTime=RunPulse / mult;
run;
```

The "Test Specification" table in Output 83.10.1 displays the **L** matrix and the **c** vector in a TEST statement. Because no label is specified for the TEST statement, "Test 1" is used as the label.

Output 83.10.1 Test Specification

The MIANALYZE Procedure
Test: Test 1

Test Specification

	L Matrix			
Parameter	**Intercept**	**RunTime**	**RunPulse**	**C**
TestPrm1	1.000000	0	0	0
TestPrm2	0	1.000000	-1.000000	0

The "Variance Information" table in Output 83.10.2 displays the between-imputation variance, within-imputation variance, and total variance for each univariate inference. A detailed description of these statistics is provided in the section "Combining Inferences from Imputed Data Sets" on page 6719 and the section "Multiple Imputation Efficiency" on page 6720.

Output 83.10.2 Variance Information

Variance Information

	Variance						
Parameter	**Between**	**Within**	**Total**	**DF**	**Relative Increase in Variance**	**Fraction Missing Information**	**Relative Efficiency**
TestPrm1	22.485821	75.413875	98.799129	19.102	0.310092	0.240234	0.990482
TestPrm2	0.022075	0.136285	0.159243	21.99	0.168452	0.145646	0.994208

The "Parameter Estimates" table in Output 83.10.3 displays the estimated mean and standard error of the linear components. The inferences are based on the t distribution. The table also displays a 95% mean confidence interval and a t test along with the associated p-value for the hypothesis that each linear component of $\mathbf{L}\boldsymbol{\beta}$ is equal to 0.

Output 83.10.3 Parameter Estimates

Parameter Estimates										
Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	C	t for H0: Parameter=C	Pr > |t|
TestPrm1	92.700420	9.939775	71.90376	113.4971	19.102	83.020730	100.839807	0	9.33	<.0001
TestPrm2	-2.950704	0.399052	-3.77831	-2.1231	21.99	-3.210987	-2.699158	0	-7.39	<.0001

When you specify the MULT option, PROC MIANALYZE assumes that the between-imputation covariance matrix is proportional to the within-imputation covariance matrix and displays a multivariate inference for all the linear components that are taken jointly in Output 83.10.4.

Output 83.10.4 Multivariate Inference

Multivariate Inference Assuming Proportionality of Between/Within Covariance Matrices				
Avg Relative Increase in Variance	Num DF	Den DF	F for H0: Parameter=Theta0	Pr > F
0.238124	2	1114.8	68.41	<.0001

Example 83.11: Combining Correlation Coefficients

(View the complete code for this example (mianax11.sas) in the example repository.)

This example combines sample correlation coefficients that are computed from a set of imputed data sets by using Fisher's z transformation.

Fisher's z transformation of the sample correlation r is

$$z = \frac{1}{2}\log\left(\frac{1+r}{1-r}\right)$$

The statistic z is approximately normally distributed, with mean

$$\log\left(\frac{1+\rho}{1-\rho}\right)$$

and variance $1/(n-3)$, where ρ is the population correlation coefficient and n is the number of observations.

The following statements use the CORR procedure to compute the correlation r and its associated Fisher's z statistic between the variables Oxygen and RunTime for each imputed data set.

```
ods select none;
proc corr data=outmi fisher(biasadj=no);
   var Oxygen RunTime;
   by _Imputation_;
   ods output FisherPearsonCorr=outz;
run;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The ODS OUTPUT statement is used to save Fisher's z statistic in an output data set. The following statements display the number of observations and Fisher's z statistic for each imputed data set in Output 83.11.1:

```
proc print data=outz (obs=10);
   title 'Fisher''s Correlation Statistics (First 10 Imputations)';
   var _Imputation_ NObs ZVal;
run;
```

Output 83.11.1 Output z Statistics

Fisher's Correlation Statistics (First 10 Imputations)

Obs	_Imputation_	NObs	ZVal
1	1	31	-1.27869
2	2	31	-1.30715
3	3	31	-1.27922
4	4	31	-1.39243
5	5	31	-1.40146
6	6	31	-1.22323
7	7	31	-1.27163
8	8	31	-1.17783
9	9	31	-1.36075
10	10	31	-1.23652

The following statements generate the standard error associated with the z statistic, $1/\sqrt{n-3}$:

```
data outz;
   set outz;
   StdZ= 1. / sqrt(NObs-3);
run;
```

The following statements use the MIANALYZE procedure to generate a combined parameter estimate $\hat{z}$ and its variance, as shown in Output 83.11.2. The ODS OUTPUT statement is used to save the parameter estimates in an output data set.

```
proc mianalyze data=outz;
   ods output ParameterEstimates=parms;
   modeleffects ZVal;
   stderr StdZ;
run;
```

Output 83.11.2 Combining Fisher's z Statistics

The MIANALYZE Procedure

Parameter Estimates (25 Imputations)

Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
ZVal	-1.292006	0.202699	-1.68963	-0.89438	1403.6	-1.401459	-1.098356	0	-6.37	<.0001

In addition to the estimate for z, PROC MIANALYZE also generates 95% confidence limits for z, $\hat{z}_{.025}$ and $\hat{z}_{.975}$. The following statements print the estimate and 95% confidence limits for z in Output 83.11.3:

```
proc print data=parms;
   title 'Parameter Estimates with 95% Confidence Limits';
   var Estimate LCLMean UCLMean;
run;
```

Output 83.11.3 Parameter Estimates with 95% Confidence Limits

Parameter Estimates with 95% Confidence Limits

Obs	Estimate	LCLMean	UCLMean
1	-1.292006	-1.68963	-0.89438

An estimate of the correlation coefficient with its corresponding 95% confidence limits is then generated from the following inverse transformation as described in the section "Correlation Coefficients" on page 6724:

$$r = \tanh(z) = \frac{e^{2z} - 1}{e^{2z} + 1}$$

for $z = \hat{z}$, $\hat{z}_{.025}$, and $\hat{z}_{.975}$.

The following statements generate and display an estimate of the correlation coefficient and its 95% confidence limits, as shown in Output 83.11.4:

```
data corr_ci;
   set parms;
   r=       tanh( Estimate);
   r_lower= tanh( LCLMean);
   r_upper= tanh( UCLMean);
run;
proc print data=corr_ci;
   title 'Estimated Correlation Coefficient'
         ' with 95% Confidence Limits';
   var r r_lower r_upper;
run;
```

Output 83.11.4 Estimated Correlation Coefficient

Estimated Correlation Coefficient with 95% Confidence Limits

Obs	r	r_lower	r_upper
1	-0.85965	-0.93410	-0.71355

Example 83.12: Sensitivity Analysis with Control-Based Pattern Imputation

(View the complete code for this example (mianax12.sas) in the example repository.)

This example illustrates sensitivity analysis in multiple imputation under the MNAR assumption by creating control-based pattern imputation.

Suppose that a pharmaceutical company is conducting a clinical trial to test the efficacy of a new drug. The trial is conducted with two groups that have an equal number of patients: a treatment group that receives the new drug and a placebo control group. The variable Trt is an indicator variable, with a value of 1 for patients in the treatment group and a value of 0 for patients in the control group. The variable Y0 is the baseline efficacy score, and the variable Y1 is the efficacy score at a follow-up visit.

If the data set does not contain any missing values, then you can use a regression model such as the following to test the treatment effect:

$$Y1 = Trt \;\; Y0$$

Suppose that the variables Trt and Y0 are fully observed and the variable Y1 contains missing values in both the treatment and control groups, as shown in Table 83.4.

Table 83.4 Variables

Variables		
Trt	**Y0**	**Y1**
0	X	X
1	X	X
0	X	.
1	X	.

Suppose the data set Mono1 contains the data from the trial that have missing values in Y1. Output 83.12.1 lists the first 10 observations.

Output 83.12.1 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Trt	y0	y1
1	0	10.5212	11.3604
2	0	8.5871	8.5178
3	0	9.3274	.
4	0	9.7519	.
5	0	9.3495	9.4369
6	1	11.5192	13.2344
7	1	10.7841	.
8	1	9.7717	10.9407
9	1	10.1455	10.8279
10	1	8.2463	9.6844

Multiple imputation often assumes that missing values are missing at random (MAR), and the following statements use the MI procedure to impute missing values under this assumption:

```
proc mi data=Mono1 seed=14823 nimpute=20 out=outex12a;
   class Trt;
   monotone reg;
   var Trt y0 y1;
run;
```

The following statements generate regression coefficients for each of the 20 imputed data sets:

```
ods select none;
proc reg data=outex12a;
   model y1= Trt y0;
   by  _Imputation_;
   ods output parameterestimates=regparms;
run;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The following statements combine the 20 sets of regression coefficients:

```
proc mianalyze parms=regparms;
   modeleffects Trt;
run;
```

The "Parameter Estimates" table in Output 83.12.2 displays a combined estimate and standard error for the regression coefficient for Trt. The table shows a t test statistic of 3.39, with the associated p-value 0.0008 for the test that the regression coefficient is equal to 0.

Output 83.12.2 Parameter Estimates

The MIANALYZE Procedure

Parameter Estimates (20 Imputations)

Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Trt	0.890609	0.262600	0.372745	1.408473	197.29	0.624115	1.228827	0	3.39	0.0008

The conclusion in Output 83.12.2 is based on the MAR assumption. But if missing Y1 values for individuals in the treatment group imply that these individuals no longer receive the treatment, then it is reasonable to assume that the conditional distribution of Y1, given Y0 for individuals who have missing Y1 values in the treatment group, is similar to the corresponding distribution of individuals in the control group.

Ratitch and O'Kelly (2011) describe an implementation of the pattern-mixture model approach that uses a control-based pattern imputation. That is, an imputation model for the missing observations in the treatment group is constructed not from the observed data in the treatment group but rather from the observed data in the control group. This model is also the imputation model that is used to impute missing observations in the control group.

The following statements implement the control-based pattern imputation:

```
proc mi data=Mono1 seed=14823 nimpute=20 out=outex12b;
   class Trt;
   monotone reg;
   mnar model( y1 /modelobs=(Trt='0'));
   var y0 y1;
run;
```

The MNAR statement imputes missing values for scenarios under the MNAR assumption. The MODEL option specifies that only observations where TRT=0 are used to derive the imputation model for the variable Y1. Thus, Y0 and Y1 (but not Trt) are specified in the VAR list.

The following statements generate regression coefficients for each of the 20 imputed data sets:

```
ods select none;
proc reg data=outex12b;
   model y1= Trt y0;
   by _Imputation_;
   ods output parameterestimates=regparms;
run;
ods select all;
```

The following statements combine the 20 sets of regression coefficients:

```
proc mianalyze parms=regparms;
   modeleffects Trt;
run;
```

Output 83.12.3 Parameter Estimates

The MIANALYZE Procedure

Parameter Estimates (20 Imputations)

Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Trt	0.708802	0.279447	0.157975	1.259628	213.68	0.329363	0.919297	0	2.54	0.0119

The "Parameter Estimates" table in Output 83.12.3 shows a t test statistic of 2.54, with the p-value 0.0119 for the test that the parameter is equal to 0. Thus, for a two-sided Type I error level of 0.05, the significance of the treatment effect is not reversed by control-based pattern imputation.

Example 83.13: Sensitivity Analysis with the Tipping-Point Approach

(View the complete code for this example (mianax13.sas) in the example repository.)

This example illustrates sensitivity analysis in multiple imputation under the MNAR assumption in which you search for a tipping point that reverses the study conclusion.

Suppose that a pharmaceutical company is conducting a clinical trial to test the efficacy of a new drug. The trial is conducted with two groups that have an equal number of patients: a treatment group that receives the new drug and a placebo control group. The variable Trt is an indicator variable, with a value of 1 for patients in the treatment group and a value of 0 for patients in the control group. The variable Y0 is the baseline efficacy score, and the variable Y1 is the efficacy score at a follow-up visit.

If the data set does not contain any missing values, then you can use a regression model such as the following to test the efficacy of the treatment effect:

$$Y1 = Trt \;\; Y0$$

Now suppose in a trial that the variables Trt and Y0 are fully observed and the variable Y1 contains missing values in both the treatment and control groups. The data set Mono2 contains the data from the trial and Figure 83.13.1 lists the first 10 observations.

Output 83.13.1 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Trt	y0	y1
1	0	11.4826	.
2	0	10.0090	10.8667
3	0	11.3643	10.6660
4	0	11.3098	10.8297
5	0	11.3094	.
6	1	10.3815	10.5587
7	1	11.2001	13.7616
8	1	9.7002	10.3460
9	1	10.0801	.
10	1	11.2667	11.0634

Multiple imputation often assumes that missing values are missing at random (MAR), and the following statements use the MI procedure to impute missing values under this assumption:

```
proc mi data=Mono2 seed=14823 out=outmi;
   class Trt;
   monotone reg;
   var Trt y0 y1;
run;
```

The following statements generate regression coefficients for each of the 25 imputed data sets:

```
ods select none;
proc reg data=outmi;
   model y1= Trt y0;
   by  _Imputation_;
   ods output parameterestimates=regparms;
run;
ods select all;
```

Because of the ODS SELECT statements, no output is displayed. The following statements combine the 20 sets of regression coefficients:

```
proc mianalyze parms=regparms;
   modeleffects Trt;
run;
```

The "Parameter Estimates" table in Output 83.13.2 displays a combined estimate and standard error for the regression coefficient for Trt. The table displays a 95% confidence interval (0.3353, 1.3208), which does not contain 0. The table also shows a t statistic of 3.31, with the associated p-value 0.0011 for the test that the regression coefficient is equal to 0.

Output 83.13.2 Parameter Estimates

The MIANALYZE Procedure

Parameter Estimates (25 Imputations)

Parameter	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Trt	0.828052	0.249933	0.335262	1.320842	203.53	0.556144	1.124339	0	3.31	0.0011

The conclusion in Output 83.13.2 is based on the MAR assumption. But if it is plausible that, for the treatment group, the distribution of missing Y1 responses has a lower expected value than that of the corresponding distribution of the observed Y1 responses, the conclusion under the MAR assumption should be examined.

The following macro performs multiple imputation analysis for a specified sequence of shift parameters, which adjust the imputed values for observations in the treatment group (TRT=1):

```
/*-----------------------------------------------------------*/
/*--- Performs multiple imputation analysis             ---*/
/*--- for specified shift parameters:                   ---*/
/*--- data= input data set                              ---*/
/*--- smin= min shift parameter                         ---*/
/*--- smax= max shift parameter                         ---*/
/*--- sinc= increment of the shift parameter            ---*/
/*--- outparms= output reg parameters                   ---*/
/*-----------------------------------------------------------*/
%macro miparms( data=, smin=, smax=, sinc=, outparms=);
ods select none;

data &outparms;
   set _null_;
run;

/*------------ # of shift values -------------*/
```

```
%let ncase= %sysevalf( (&smax-&smin)/&sinc, ceil );

/*---- Multiple imputation analysis for each shift ----*/
%do jc=0 %to &ncase;
   %let sj= %sysevalf( &smin + &jc * &sinc);

  /*---- Generates 25 imputed data sets ----*/
   proc mi data=&data seed=14823 out=outmi;
      class Trt;
      monotone reg;
      mnar adjust( y1 / shift=&sj  adjustobs=(Trt='1') );
      var Trt y0 y1;
   run;

  /*------ Perform reg test -------*/
   proc reg data=outmi;
      model y1= Trt y0;
      by  _Imputation_;
      ods output parameterestimates=regparm;
   run;

  /*------ Combine reg results -------*/
   proc mianalyze parms=regparm;
      modeleffects Trt;
      ods output parameterestimates=miparm;
   run;

   data miparm;
      set miparm;
      Shift= &sj;
      keep Shift Probt;
   run;

  /*----- Output multiple imputation results ----*/
   data &outparms;
      set &outparms miparm;
   run;

%end;

ods select all;
%mend miparms;
```

Assume that the tipping point for the shift parameter that reverses the study conclusion is between –3 and 0. The following statement performs multiple imputation analysis for each of the shift parameters –3.0, –2.9, . . . , 0:

```
%miparms( data=Mono2, smin=-3, smax=0, sinc=0.1, outparms=parm1);
```

The following statements display the plot of *p*-values for shift parameters between –3 and 0:

```
title 'P-value Plot';
proc sgplot data=parm1;
   series x=Shift y=Probt / lineattrs=graphfit(color=blue);
   refline 0.05 / axis=y lineattrs=graphfit(thickness=1px pattern=shortdash);
```

```
   xaxis label="Shift Parameter";
   yaxis label="P-value";
run;
```

Output 83.13.3 Finding Tipping Point for Shift Parameter between –3 and 0

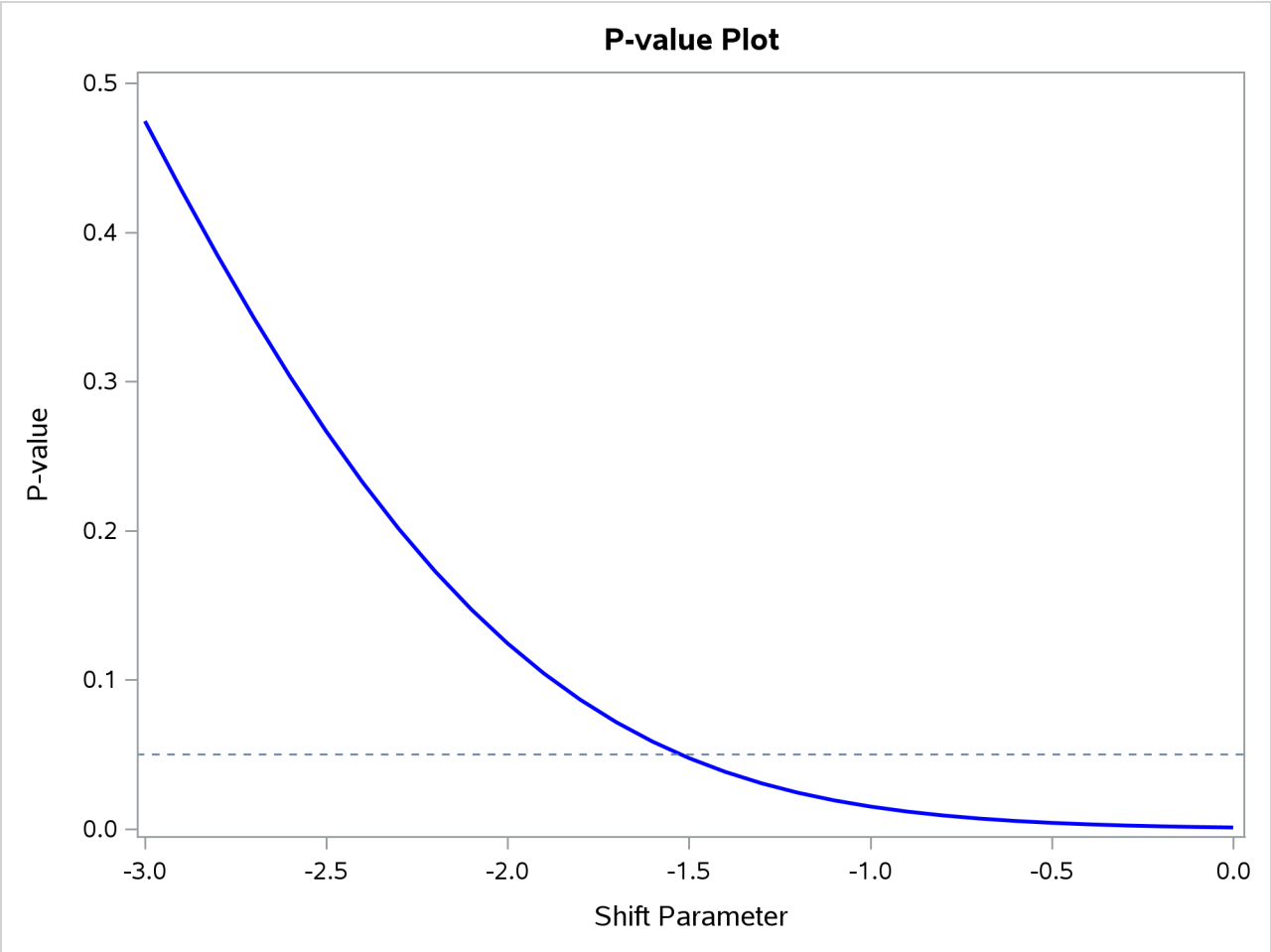

For a two-sided Type I error level of 0.05, the tipping point for the shift parameter is slightly less than –1.5. The following statement performs multiple imputation analysis for shift parameters –1.60, –1.59, . . . , –1.50:

```
%miparms( data=Mono2, smin=-1.6, smax=-1.5, sinc=0.01, outparms=parm2);
```

The following statements display the *p*-values that are associated with the shift parameters:

```
proc print label data=parm2;
   var Shift Probt;
   title 'P-values for Shift Parameters';
   label Probt='Pr > |t|';
   format Probt 8.4;
run;
```

Output 83.13.4 Finding Tipping Point for Shift Parameter between –1.6 and –1.5

P-values for Shift Parameters

Obs	Shift	Pr > |t|
1	-1.60	0.0586
2	-1.59	0.0574
3	-1.58	0.0562
4	-1.57	0.0551
5	-1.56	0.0539
6	-1.55	0.0528
7	-1.54	0.0517
8	-1.53	0.0507
9	-1.52	0.0496
10	-1.51	0.0486
11	-1.50	0.0476

The study conclusion under MAR is reversed when the shift parameter is –1.53. Thus, if the shift parameter –1.53 is plausible, the conclusion under MAR is questionable.

References

Allison, P. D. (2000). "Multiple Imputation for Missing Data: A Cautionary Tale." *Sociological Methods and Research* 28:301–309.

Allison, P. D. (2001). *Missing Data*. Thousand Oaks, CA: Sage Publications.

Barnard, J., and Rubin, D. B. (1999). "Small-Sample Degrees of Freedom with Multiple Imputation." *Biometrika* 86:948–955.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Gadbury, G. L., Coffey, C. S., and Allison, D. B. (2003). "Modern Statistical Methods for Handling Missing Repeated Measurements in Obesity Trial Data: Beyond LOCF." *Obesity Reviews* 4:175–184.

Horton, N. J., and Lipsitz, S. R. (2001). "Multiple Imputation in Practice: Comparison of Software Packages for Regression Models with Missing Variables." *American Statistician* 55:244–254.

Li, K. H., Raghunathan, T. E., and Rubin, D. B. (1991). "Large-Sample Significance Levels from Multiply Imputed Data Using Moment-Based Statistics and an F Reference Distribution." *Journal of the American Statistical Association* 86:1065–1073.

Little, R. J. A., and Rubin, D. B. (2002). *Statistical Analysis with Missing Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Ratitch, B., and O'Kelly, M. (2011). "Implementation of Pattern-Mixture Models Using Standard SAS/STAT Procedures." In *Proceedings of PharmaSUG 2011 (Pharmaceutical Industry SAS Users Group)*. Paper SP04. Cary, NC: SAS Institute Inc.

Rubin, D. B. (1976). "Inference and Missing Data." *Biometrika* 63:581–592.

Rubin, D. B. (1987). *Multiple Imputation for Nonresponse in Surveys*. New York: John Wiley & Sons.

Rubin, D. B. (1996). "Multiple Imputation after 18+ Years." *Journal of the American Statistical Association* 91:473–489.

Schafer, J. L. (1997). *Analysis of Incomplete Multivariate Data*. New York: Chapman & Hall.

Chapter 84
The MIXED Procedure

Contents

Overview: MIXED Procedure . . . **6762**
- Basic Features . . . 6763
- Notation for the Mixed Model . . . 6764
- PROC MIXED Contrasted with Other SAS Procedures . . . 6765

Getting Started: MIXED Procedure . . . **6766**
- Clustered Data Example . . . 6766

Syntax: MIXED Procedure . . . **6772**
- PROC MIXED Statement . . . 6774
- BY Statement . . . 6786
- CLASS Statement . . . 6787
- CODE Statement . . . 6788
- CONTRAST Statement . . . 6788
- ESTIMATE Statement . . . 6791
- ID Statement . . . 6793
- LSMEANS Statement . . . 6794
- LSMESTIMATE Statement . . . 6800
- MODEL Statement . . . 6801
- PARMS Statement . . . 6816
- PRIOR Statement . . . 6819
- RANDOM Statement . . . 6823
- REPEATED Statement . . . 6828
- SLICE Statement . . . 6842
- STORE Statement . . . 6842
- WEIGHT Statement . . . 6842

Details: MIXED Procedure . . . **6843**
- Mixed Models Theory . . . 6843
- Parameterization of Mixed Models . . . 6855
- Residuals and Influence Diagnostics . . . 6860
- Default Output . . . 6868
- ODS Table Names . . . 6872
- ODS Graphics . . . 6877
- Computational Issues . . . 6882

Examples: MIXED Procedure . . . **6886**
- Example 84.1: Split-Plot Design . . . 6886
- Example 84.2: Repeated Measures . . . 6891
- Example 84.3: Plotting the Likelihood . . . 6902

Example 84.4: Known G and R 6909
Example 84.5: Random Coefficients 6915
Example 84.6: Line-Source Sprinkler Irrigation 6922
Example 84.7: Influence in Heterogeneous Variance Model 6927
Example 84.8: Influence Analysis for Repeated Measures Data 6936
Example 84.9: Examining Individual Test Components 6945
Example 84.10: Isotonic Contrasts for Ordered Mean Values 6950
References **6951**

Overview: MIXED Procedure

The MIXED procedure fits a variety of mixed linear models to data and enables you to use these fitted models to make statistical inferences about the data. A *mixed linear model* is a generalization of the standard linear model used in the GLM procedure, the generalization being that the data are permitted to exhibit correlation and nonconstant variability. The mixed linear model, therefore, provides you with the flexibility of modeling not only the means of your data (as in the standard linear model) but their variances and covariances as well.

The primary assumptions underlying the analyses performed by PROC MIXED are as follows:

- The data are normally distributed (Gaussian).
- The means (expected values) of the data are linear in terms of a certain set of parameters.
- The variances and covariances of the data are in terms of a different set of parameters, and they exhibit a structure matching one of those available in PROC MIXED.

Since Gaussian data can be modeled entirely in terms of their means and variances/covariances, the two sets of parameters in a mixed linear model actually specify the complete probability distribution of the data. The parameters of the mean model are referred to as *fixed-effects parameters*, and the parameters of the variance-covariance model are referred to as *covariance parameters*.

The fixed-effects parameters are associated with known explanatory variables, as in the standard linear model. These variables can be either qualitative (as in the traditional analysis of variance) or quantitative (as in standard linear regression). However, the covariance parameters are what distinguishes the mixed linear model from the standard linear model.

The need for covariance parameters arises quite frequently in applications, the following being the two most typical scenarios:

- The experimental units on which the data are measured can be grouped into clusters, and the data from a common cluster are correlated.
- Repeated measurements are taken on the same experimental unit, and these repeated measurements are correlated or exhibit variability that changes.

The first scenario can be generalized to include one set of clusters nested within another. For example, if students are the experimental unit, they can be clustered into classes, which in turn can be clustered into schools. Each level of this hierarchy can introduce an additional source of variability and correlation. The second scenario occurs in longitudinal studies, where repeated measurements are taken over time. Alternatively, the repeated measures could be spatial or multivariate in nature.

PROC MIXED provides a variety of covariance structures to handle the previous two scenarios. The most common of these structures arises from the use of *random-effects parameters*, which are additional unknown random variables assumed to affect the variability of the data. The variances of the random-effects parameters, commonly known as *variance components*, become the covariance parameters for this particular structure. Traditional mixed linear models contain both fixed- and random-effects parameters, and, in fact, it is the combination of these two types of effects that led to the name *mixed model*. PROC MIXED fits not only these traditional variance component models but numerous other covariance structures as well.

PROC MIXED fits the structure you select to the data by using the method of *restricted maximum likelihood (REML)*, also known as *residual maximum likelihood*. It is here that the Gaussian assumption for the data is exploited. Other estimation methods are also available, including *maximum likelihood* and *MIVQUE0*. The details behind these estimation methods are discussed in subsequent sections.

After a model has been fit to your data, you can use it to draw statistical inferences via both the fixed-effects and covariance parameters. PROC MIXED computes several different statistics suitable for generating hypothesis tests and confidence intervals. The validity of these statistics depends upon the mean and variance-covariance model you select, so it is important to choose the model carefully. Some of the output from PROC MIXED helps you assess your model and compare it with others.

Basic Features

PROC MIXED provides easy accessibility to numerous mixed linear models that are useful in many common statistical analyses. In the style of the GLM procedure, PROC MIXED fits the specified mixed linear model and produces appropriate statistics.

Here are some basic features of PROC MIXED:

- covariance structures, including variance components, compound symmetry, unstructured, AR(1), Toeplitz, spatial, general linear, and factor analytic
- GLM-type grammar, by using MODEL, RANDOM, and REPEATED statements for model specification and CONTRAST, ESTIMATE, and LSMEANS statements for inferences
- appropriate standard errors for all specified estimable linear combinations of fixed and random effects, and corresponding t and F tests
- subject and group effects that enable blocking and heterogeneity, respectively
- REML and ML estimation methods implemented with a Newton-Raphson algorithm
- capacity to handle unbalanced data
- ability to create a SAS data set corresponding to any table

PROC MIXED uses the Output Delivery System (ODS), a SAS subsystem that provides capabilities for displaying and controlling the output from SAS procedures. ODS enables you to convert any of the output from PROC MIXED into a SAS data set. See the section "ODS Table Names" on page 6872.

The MIXED procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the MIXED procedure, see the PLOTS= option in the PROC MIXED statement and the section "ODS Graphics" on page 6877.

Notation for the Mixed Model

This section introduces the mathematical notation used throughout this chapter to describe the mixed linear model. You should be familiar with basic matrix algebra (see Searle 1982). A more detailed description of the mixed model is contained in the section "Mixed Models Theory" on page 6843.

A statistical model is a mathematical description of how data are generated. The standard linear model, as used by the GLM procedure, is one of the most common statistical models:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

In this expression, $\mathbf{y}$ represents a vector of observed data, $\boldsymbol{\beta}$ is an unknown vector of fixed-effects parameters with known design matrix $\mathbf{X}$, and $\boldsymbol{\epsilon}$ is an unknown random error vector modeling the statistical noise around $\mathbf{X}\boldsymbol{\beta}$. The focus of the standard linear model is to model the mean of $\mathbf{y}$ by using the fixed-effects parameters $\boldsymbol{\beta}$. The residual errors $\boldsymbol{\epsilon}$ are assumed to be independent and identically distributed Gaussian random variables with mean 0 and variance σ^2.

The mixed model generalizes the standard linear model as follows:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

Here, $\boldsymbol{\gamma}$ is an unknown vector of random-effects parameters with known design matrix $\mathbf{Z}$, and $\boldsymbol{\epsilon}$ is an unknown random error vector whose elements are no longer required to be independent and homogeneous.

To further develop this notion of variance modeling, assume that $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$ are Gaussian random variables that are uncorrelated and have expectations $\mathbf{0}$ and variances $\mathbf{G}$ and $\mathbf{R}$, respectively. The variance of $\mathbf{y}$ is thus

$$\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$$

Note that, when $\mathbf{R} = \sigma^2\mathbf{I}$ and $\mathbf{Z} = \mathbf{0}$, the mixed model reduces to the standard linear model.

You can model the variance of the data, $\mathbf{y}$, by specifying the structure (or form) of $\mathbf{Z}$, $\mathbf{G}$, and $\mathbf{R}$. The model matrix $\mathbf{Z}$ is set up in the same fashion as $\mathbf{X}$, the model matrix for the fixed-effects parameters. For $\mathbf{G}$ and $\mathbf{R}$, you must select some *covariance structure*. Possible covariance structures include the following:

- variance components
- compound symmetry (common covariance plus diagonal)
- unstructured (general covariance)
- autoregressive

- spatial
- general linear
- factor analytic

By appropriately defining the model matrices **X** and **Z**, as well as the covariance structure matrices **G** and **R**, you can perform numerous mixed model analyses.

PROC MIXED Contrasted with Other SAS Procedures

PROC MIXED is a generalization of the GLM procedure in the sense that PROC GLM fits standard linear models, and PROC MIXED fits the wider class of mixed linear models. Both procedures have similar CLASS, MODEL, CONTRAST, ESTIMATE, and LSMEANS statements, but their RANDOM and REPEATED statements differ (see the following paragraphs). Both procedures use the non-full-rank model parameterization, although the sorting of classification levels can differ between the two. PROC MIXED computes only Type I–Type III tests of fixed effects, while PROC GLM computes Types I–IV.

The RANDOM statement in PROC MIXED incorporates random effects constituting the $\boldsymbol{\gamma}$ vector in the mixed model. However, in PROC GLM, effects specified in the RANDOM statement are still treated as fixed as far as the model fit is concerned, and they serve only to produce corresponding expected mean squares. These expected mean squares lead to the traditional ANOVA estimates of variance components. PROC MIXED computes REML and ML estimates of variance parameters, which are generally preferred to the ANOVA estimates (Searle 1988; Harville 1988; Searle, Casella, and McCulloch 1992). Optionally, PROC MIXED also computes MIVQUE0 estimates, which are similar to ANOVA estimates.

The REPEATED statement in PROC MIXED is used to specify covariance structures for repeated measurements on subjects, while the REPEATED statement in PROC GLM is used to specify various transformations with which to conduct the traditional univariate or multivariate tests. In repeated measures situations, the mixed model approach used in PROC MIXED is more flexible and more widely applicable than either the univariate or multivariate approach. In particular, the mixed model approach provides a larger class of covariance structures and a better mechanism for handling missing values (Wolfinger and Chang 1995).

PROC MIXED subsumes the VARCOMP procedure. PROC MIXED provides a wide variety of covariance structures, while PROC VARCOMP estimates only simple random effects. PROC MIXED carries out several analyses that are absent in PROC VARCOMP, including the estimation and testing of linear combinations of fixed and random effects.

The ARIMA and AUTOREG procedures provide more time series structures than PROC MIXED, although they do not fit variance component models. The CALIS procedure fits general covariance matrices, but the fixed effects structure of the model is formed differently than in PROC MIXED. The LATTICE and NESTED procedures fit special types of mixed linear models that can also be handled in PROC MIXED, although PROC MIXED might run slower because of its more general algorithm. The TSCSREG procedure analyzes time series cross-sectional data, and it fits some structures not available in PROC MIXED.

The GLIMMIX procedure fits generalized linear mixed models (GLMMs). Linear mixed models—where the data are normally distributed, given the random effects—are in the class of GLMMs. The MIXED procedure can estimate covariance parameters with ANOVA methods that are not available in the GLIMMIX procedure (see METHOD=TYPE1, METHOD=TYPE2, and METHOD=TYPE3 in the PROC MIXED statement). Also, PROC MIXED can perform a sampling-based Bayesian analysis through the PRIOR statement, and

the procedure supports certain Kronecker-type covariance structures. These features are not available in the GLIMMIX procedure. The GLIMMIX procedure, on the other hand, accommodates nonnormal data and offers a broader array of post-processing features than the MIXED procedure.

Getting Started: MIXED Procedure

Clustered Data Example

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/mixgs1.sas.)

Consider the following SAS data set as an introductory example:

```
data heights;
   input Family Gender$ Height @@;
   datalines;
1 F 67   1 F 66   1 F 64   1 M 71   1 M 72   2 F 63
2 F 63   2 F 67   2 M 69   2 M 68   2 M 70   3 F 63
3 M 64   4 F 67   4 F 66   4 M 67   4 M 67   4 M 69
;
```

The response variable Height measures the heights (in inches) of 18 individuals. The individuals are classified according to Family and Gender. You can perform a traditional two-way analysis of variance of these data with the following PROC MIXED statements:

```
proc mixed data=heights;
   class Family Gender;
   model Height = Gender Family Family*Gender;
run;
```

The PROC MIXED statement invokes the procedure. The CLASS statement instructs PROC MIXED to consider both Family and Gender as classification variables. Dummy (indicator) variables are, as a result, created corresponding to all of the distinct levels of Family and Gender. For these data, Family has four levels and Gender has two levels.

The MODEL statement first specifies the response (dependent) variable Height. The explanatory (independent) variables are then listed after the equal (=) sign. Here, the two explanatory variables are Gender and Family, and these are the main effects of the design. The third explanatory term, Family*Gender, models an interaction between the two main effects.

PROC MIXED uses the dummy variables associated with Gender, Family, and Family*Gender to construct the $\mathbf{X}$ matrix for the linear model. A column of 1s is also included as the first column of $\mathbf{X}$ to model a global intercept. There are no $\mathbf{Z}$ or $\mathbf{G}$ matrices for this model, and $\mathbf{R}$ is assumed to equal $\sigma^2\mathbf{I}$, where $\mathbf{I}$ is an 18×18 identity matrix.

The RUN statement completes the specification. The coding is precisely the same as with the GLM procedure. However, much of the output from PROC MIXED is different from that produced by PROC GLM.

The output from PROC MIXED is shown in Figure 84.1–Figure 84.7.

The "Model Information" table in Figure 84.1 describes the model, some of the variables that it involves, and the method used in fitting it. This table also lists the method (profile, factor, parameter, or none) for handling the residual variance.

Figure 84.1 Model Information

The Mixed Procedure

Model Information	
Data Set	WORK.HEIGHTS
Dependent Variable	Height
Covariance Structure	Diagonal
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Residual

The "Class Level Information" table in Figure 84.2 lists the levels of all variables specified in the CLASS statement. You can check this table to make sure that the data are correct.

Figure 84.2 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
Family	4	1 2 3 4
Gender	2	F M

The "Dimensions" table in Figure 84.3 lists the sizes of relevant matrices. This table can be useful in determining CPU time and memory requirements.

Figure 84.3 Dimensions

Dimensions	
Covariance Parameters	1
Columns in X	15
Columns in Z	0
Subjects	1
Max Obs per Subject	18

The "Number of Observations" table in Figure 84.4 displays information about the sample size being processed.

Figure 84.4 Number of Observations

Number of Observations	
Number of Observations Read	18
Number of Observations Used	18
Number of Observations Not Used	0

The "Covariance Parameter Estimates" table in Figure 84.5 displays the estimate of σ^2 for the model.

Figure 84.5 Covariance Parameter Estimates

Covariance Parameter Estimates	
Cov Parm	**Estimate**
Residual	2.1000

The "Fit Statistics" table in Figure 84.6 lists several pieces of information about the fitted mixed model, including values derived from the computed value of the restricted/residual likelihood.

Figure 84.6 Fit Statistics

Fit Statistics	
-2 Res Log Likelihood	41.6
AIC (Smaller is Better)	43.6
AICC (Smaller is Better)	44.1
BIC (Smaller is Better)	43.9

The "Type 3 Tests of Fixed Effects" table in Figure 84.7 displays significance tests for the three effects listed in the MODEL statement. The Type 3 F statistics and p-values are the same as those produced by the GLM procedure. However, because PROC MIXED uses a likelihood-based estimation scheme, it does not directly compute or display sums of squares for this analysis.

Figure 84.7 Tests of Fixed Effects

Type 3 Tests of Fixed Effects				
Effect	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Gender	1	10	17.63	0.0018
Family	3	10	5.90	0.0139
Family*Gender	3	10	2.89	0.0889

The Type 3 test for Family*Gender effect is not significant at the 5% level, but the tests for both main effects are significant.

The important assumptions behind this analysis are that the data are normally distributed and that they are independent with constant variance. For these data, the normality assumption is probably realistic since the data are observed heights. However, since the data occur in clusters (families), it is very likely that observations from the same family are statistically correlated—that is, not independent.

The methods implemented in PROC MIXED are still based on the assumption of normally distributed data, but you can drop the assumption of independence by modeling statistical correlation in a variety of ways. You can also model variances that are heterogeneous—that is, nonconstant.

For the height data, one of the simplest ways of modeling correlation is through the use of *random effects*. Here the family effect is assumed to be normally distributed with zero mean and some unknown variance. This is in contrast to the previous model in which the family effects are just constants, or *fixed effects*.

Declaring Family as a random effect sets up a common correlation among all observations having the same level of Family.

Declaring Family*Gender as a random effect models an additional correlation between all observations that have the same level of both Family and Gender. One interpretation of this effect is that a female in a certain family exhibits more correlation with the other females in that family than with the other males, and likewise for a male. With the height data, this model seems reasonable.

The statements to fit this correlation model in PROC MIXED are as follows:

```
proc mixed;
   class Family Gender;
   model Height = Gender;
   random Family Family*Gender;
run;
```

Note that Family and Family*Gender are now listed in the RANDOM statement. The dummy variables associated with them are used to construct the $\mathbf{Z}$ matrix in the mixed model. The $\mathbf{X}$ matrix now consists of a column of 1s and the dummy variables for Gender.

The $\mathbf{G}$ matrix for this model is diagonal, and it contains the variance components for both Family and Family*Gender. The $\mathbf{R}$ matrix is still assumed to equal $\sigma^2\mathbf{I}$, where $\mathbf{I}$ is an identity matrix.

The output from this analysis is as follows.

Figure 84.8 Model Information

The Mixed Procedure

Model Information	
Data Set	WORK.HEIGHTS
Dependent Variable	Height
Covariance Structure	Variance Components
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

The "Model Information" table in Figure 84.8 shows that the containment method is used to compute the degrees of freedom for this analysis. This is the default method when a RANDOM statement is used; for more information, see the description of the DDFM= option.

Figure 84.9 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
Family	4	1 2 3 4
Gender	2	F M

The "Class Level Information" table in Figure 84.9 is the same as before. The "Dimensions" table in Figure 84.10 displays the new sizes of the $\mathbf{X}$ and $\mathbf{Z}$ matrices.

Figure 84.10 Dimensions and Number of Observations

Dimensions	
Covariance Parameters	3
Columns in X	3
Columns in Z	12
Subjects	1
Max Obs per Subject	18

Number of Observations	
Number of Observations Read	18
Number of Observations Used	18
Number of Observations Not Used	0

The "Iteration History" table in Figure 84.11 displays the results of the numerical optimization of the restricted/residual likelihood. Six iterations are required to achieve the default convergence criterion of 1E–8.

Figure 84.11 REML Estimation Iteration History

Iteration History			
Iteration	Evaluations	-2 Res Log Like	Criterion
0	1	74.11074833	
1	2	71.51614003	0.01441208
2	1	71.13845990	0.00412226
3	1	71.03613556	0.00058188
4	1	71.02281757	0.00001689
5	1	71.02245904	0.00000002
6	1	71.02245869	0.00000000

Convergence criteria met.

The "Covariance Parameter Estimates" table in Figure 84.12 displays the results of the REML fit. The Estimate column contains the estimates of the variance components for Family and Family*Gender, as well as the estimate of σ^2.

Figure 84.12 Covariance Parameter Estimates (REML)

Covariance Parameter Estimates	
Cov Parm	Estimate
Family	2.4010
Family*Gender	1.7657
Residual	2.1668

The "Fit Statistics" table in Figure 84.13 contains basic information about the REML fit.

Figure 84.13 Fit Statistics

Fit Statistics	
-2 Res Log Likelihood	71.0
AIC (Smaller is Better)	77.0
AICC (Smaller is Better)	79.0
BIC (Smaller is Better)	75.2

The "Type 3 Tests of Fixed Effects" table in Figure 84.14 contains a significance test for the lone fixed effect, Gender. Note that the associated p-value is not nearly as significant as in the previous analysis. This illustrates the importance of correctly modeling correlation in your data.

Figure 84.14 Type 3 Tests of Fixed Effects

Type 3 Tests of Fixed Effects				
Effect	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Gender	1	3	7.95	0.0667

An additional benefit of the random effects analysis is that it enables you to make inferences about gender that apply to an entire population of families, whereas the inferences about gender from the analysis where Family and Family*Gender are fixed effects apply only to the particular families in the data set.

PROC MIXED thus offers you the ability to model correlation directly and to make inferences about fixed effects that apply to entire populations of random effects.

Syntax: MIXED Procedure

The following statements are available in the MIXED procedure:

```
PROC MIXED <options> ;
    BY variables ;
    CLASS variable <(REF= option)> ... <variable <(REF= option)>> </global-options> ;
    CODE <options> ;
    ID variables ;
    MODEL dependent = <fixed-effects> </options> ;
    RANDOM random-effects </options> ;
    REPEATED <repeated-effect> </options> ;
    PARMS (value-list)... </options> ;
    PRIOR <distribution> </options> ;
    CONTRAST 'label' <fixed-effect values ... >
                <| random-effect values ... >, ... </options> ;
    ESTIMATE 'label' <fixed-effect values ... >
               <| random-effect values ... > </options> ;
    LSMEANS fixed-effects </options> ;
    LSMESTIMATE model-effect lsmestimate-specification </options> ;
    SLICE model-effect </options> ;
    STORE <OUT=>item-store-name </ LABEL='label'> ;
    WEIGHT variable ;
```

Items within angle brackets (< >) are optional. The CONTRAST, ESTIMATE, LSMEANS, and RANDOM statements can appear multiple times; all other statements can appear only once.

The PROC MIXED and MODEL statements are required, and the MODEL statement must appear after the CLASS statement if a CLASS statement is included. The CONTRAST, ESTIMATE, LSMEANS, RANDOM, and REPEATED statements must follow the MODEL statement. The CONTRAST and ESTIMATE statements must also follow any RANDOM statements. The LSMESTIMATE, SLICE, and STORE statements are shared with many procedures. Summary descriptions of functionality and syntax for these statements are also given after the PROC MIXED statement in alphabetical order, but you can find full documentation on them in Chapter 20, "Shared Concepts and Topics."

Table 84.1 summarizes the basic functions and important *options* of each PROC MIXED statement. The syntax of each statement in Table 84.1 is described in the following sections in alphabetical order after the description of the PROC MIXED statement.

Table 84.1 Summary of PROC MIXED Statements

Statement	Description	Options
PROC MIXED	Invokes the procedure	DATA= specifies input data set, METHOD= specifies estimation method
BY	Performs multiple PROC MIXED analyses in one invocation	None

Table 84.1 *continued*

Statement	Description	Options
CLASS	Declares qualitative variables that create indicator variables in design matrices	None
CODE	Requests that the procedure write SAS DATA step code to a file or catalog entry	FILE= names the file where the generated code is saved, CATALOG= names the catalog entry where the generated code is saved, IMPUTE imputes predicted values for observations with missing or invalid covariates, RESIDUAL computes residuals
ID	Lists additional variables to be included in predicted values tables	None
MODEL	Specifies dependent variable and fixed effects, setting up **X**	S requests solution for fixed-effects parameters, DDFM= specifies denominator degrees of freedom method, OUTP= outputs predicted values to a data set, INFLUENCE computes influence diagnostics
RANDOM	Specifies random effects, setting up **Z** and **G**	SUBJECT= creates block-diagonality, TYPE= specifies covariance structure, S requests solution for random-effects parameters, G displays estimated **G**
REPEATED	Sets up **R**	SUBJECT= creates block-diagonality, TYPE= specifies covariance structure, R displays estimated blocks of **R**, GROUP= enables between-subject heterogeneity, LOCAL adds a diagonal matrix to **R**
PARMS	Specifies a grid of initial values for the covariance parameters	HOLD= and NOITER hold the covariance parameters or their ratios constant, PARMSDATA= reads the initial values from a SAS data set
PRIOR	Performs a sampling-based Bayesian analysis for variance component models	NSAMPLE= specifies the sample size, SEED= specifies the starting seed
CONTRAST	Constructs custom hypothesis tests	E displays the **L** matrix coefficients
ESTIMATE	Constructs custom scalar estimates	CL produces confidence limits

Table 84.1 *continued*

Statement	Description	Options
LSMEANS	Computes least squares means for classification fixed effects	DIFF computes differences of the least squares means, ADJUST= performs multiple comparisons adjustments, AT changes covariates, OM changes weighting, CL produces confidence limits, SLICE= tests simple effects
LSMESTIMATE	Provides custom hypothesis tests among the least squares means	ADJUST= determines the method for multiple comparison adjustment of LS-mean differences, JOINT requests a joint F or chi-square test for the rows of the estimate
SLICE	Performs a partitioned analysis of LS–means for an interaction	ADJUST= determines the method for multiple comparison adjustment of LS-mean differences, DIFF requests differences of LS-means
STORE	Saves the context and results of the analysis	LABEL= adds a custom label
WEIGHT	Specifies a variable by which to weight $\mathbf{R}$	None

PROC MIXED Statement

PROC MIXED < *options* > **;**

The PROC MIXED statement invokes the MIXED procedure. Table 84.2 summarizes the *options* available in the PROC MIXED statement. These and other *options* in the PROC MIXED statement are then described fully in alphabetical order.

Table 84.2 PROC MIXED Statement Options

Option	Description
Basic Options	
DATA=	Specifies input data set
METHOD=	Specifies the estimation method
NOPROFILE	Includes scale parameter in optimization
ORDER=	Determines the sort order of CLASS variables
Displayed Output	
ASYCORR	Displays asymptotic correlation matrix of covariance parameter estimates
ASYCOV	Displays asymptotic covariance matrix of covariance parameter estimates
CL	Requests confidence limits for covariance parameter estimates
COVTEST	Displays asymptotic standard errors and Wald tests for covariance parameters

Table 84.2 *continued*

Option	Description
IC	Displays a table of information criteria
ITDETAILS	Displays estimates and gradients added to "Iteration History"
LOGNOTE	Writes periodic status notes to the log
MMEQ	Displays mixed model equations
MMEQSOL	Displays the solution to the mixed model equations
NOCLPRINT	Suppresses "Class Level Information" completely or in parts
NOITPRINT	Suppresses "Iteration History" table
PLOTS=	Produces ODS statistical graphics
RANKS=	Displays a table of ranks of design matrices **X** and (**XZ**)
RATIO	Produces ratio of covariance parameter estimates with residual variance
Optimization Options	
MAXFUNC=	Specifies the maximum number of likelihood evaluations
MAXITER=	Specifies the maximum number of iterations
Computational Options	
CONVF	Requests and tunes the relative function convergence criterion
CONVG	Requests and tunes the relative gradient convergence criterion
CONVH	Requests and tunes the relative Hessian convergence criterion
DFBW	Selects between-within degree of freedom method
EMPIRICAL	Computes empirical ("sandwich") estimators
NOBOUND	Unbounds covariance parameter estimates
RIDGE=	Specifies starting value for minimum ridge value
SCORING=	Applies Fisher scoring where applicable

You can specify the following *options.*

ABSOLUTE

makes the convergence criterion absolute. By default, it is relative (divided by the current objective function value). See the CONVF, CONVG, and CONVH options in this section for a description of various convergence criteria.

ALPHA=*number*

requests that confidence limits be constructed for the covariance parameter estimates with confidence level $1 - \mathit{number}$. The value of *number* must be between 0 and 1; the default is 0.05.

ANOVAF

The ANOVAF option computes F tests in models with REPEATED statement and without RANDOM statement by a method similar to that of Brunner, Domhof, and Langer (2002). The method consists of computing special F statistics and adjusting their degrees of freedom. The technique is a generalization of the Greenhouse-Geisser adjustment in MANOVA models (Greenhouse and Geisser 1959). For more details, see the section "*F* Tests With the ANOVAF Option" on page 6853.

ASYCORR

produces the asymptotic correlation matrix of the covariance parameter estimates. It is computed from the corresponding asymptotic covariance matrix (see the description of the ASYCOV option, which follows). The name of the "Asymptotic Correlation" table is AsyCorr.

ASYCOV

requests that the asymptotic covariance matrix of the covariance parameters be displayed. By default, this matrix is the observed inverse Fisher information matrix, which equals $2\mathbf{H}^{-1}$, where $\mathbf{H}$ is the Hessian (second derivative) matrix of the objective function. For more information about this matrix, see the section "Covariance Parameter Estimates" on page 6870. When you use the SCORING= option and PROC MIXED converges without stopping the scoring algorithm, PROC MIXED uses the expected Hessian matrix to compute the covariance matrix instead of the observed Hessian. The ODS name of the "Asymptotic Covariance" table is AsyCov.

CL<=WALD>

requests confidence limits for the covariance parameter estimates. A Satterthwaite approximation is used to construct limits for all parameters that have a lower boundary constraint of zero. These limits take the form

$$\frac{\nu\widehat{\sigma}^2}{\chi^2_{\nu,1-\alpha/2}} \le \sigma^2 \le \frac{\nu\widehat{\sigma}^2}{\chi^2_{\nu,\alpha/2}}$$

where $\nu = 2Z^2$, Z is the Wald statistic $\widehat{\sigma}^2/\mathrm{se}(\widehat{\sigma}^2)$, and the denominators are quantiles of the χ^2-distribution with ν degrees of freedom. See Milliken and Johnson (1992) and Burdick and Graybill (1992) for similar techniques.

For all other parameters, Wald Z-scores and normal quantiles are used to construct the limits. Wald limits are also provided for variance components if you specify the NOBOUND option. The optional =WALD specification requests Wald limits for all parameters.

The confidence limits are displayed as extra columns in the "Covariance Parameter Estimates" table. The confidence level is $1 - \alpha = 0.95$ by default; this can be changed with the ALPHA= option.

CONVF<=*number*>

requests the relative function convergence criterion with tolerance *number*. The relative function convergence criterion is

$$\frac{|f_k - f_{k-1}|}{|f_k|} \le \textit{number}$$

where f_k is the value of the objective function at iteration k. To prevent the division by $|f_k|$, use the ABSOLUTE option. The default convergence criterion is CONVH, and the default tolerance is 1E–8.

CONVG <=*number*>

requests the relative gradient convergence criterion with tolerance *number*. The relative gradient convergence criterion is

$$\frac{\max_j |g_{jk}|}{|f_k|} \le \textit{number}$$

where f_k is the value of the objective function, and g_{jk} is the jth element of the gradient (first derivative) of the objective function, both at iteration k. To prevent division by $|f_k|$, use the ABSOLUTE option. The default convergence criterion is CONVH, and the default tolerance is 1E–8.

CONVH<=*number*>

requests the relative Hessian convergence criterion with tolerance *number*. The relative Hessian convergence criterion is

$$\frac{\mathbf{g}_k{}'\mathbf{H}_k^{-1}\mathbf{g}_k}{|f_k|} \leq \textit{number}$$

where f_k is the value of the objective function, $\mathbf{g}_k$ is the gradient (first derivative) of the objective function, and $\mathbf{H}_k$ is the Hessian (second derivative) of the objective function, all at iteration k.

If $\mathbf{H}_k$ is singular, then PROC MIXED uses the following relative criterion:

$$\frac{\mathbf{g}_k'\mathbf{g}_k}{|f_k|} \leq \textit{number}$$

To prevent the division by $|f_k|$, use the ABSOLUTE option. The default convergence criterion is CONVH, and the default tolerance is 1E–8.

COVTEST

produces asymptotic standard errors and Wald Z-tests for the covariance parameter estimates.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC MIXED. The default is the most recently created data set.

DFBW

has the same effect as the DDFM=BW option in the MODEL statement.

EMPIRICAL

computes the estimated variance-covariance matrix of the fixed-effects parameters by using the asymptotically consistent estimator described in Huber (1967); White (1980); Liang and Zeger (1986); Diggle, Liang, and Zeger (1994). This estimator is commonly referred to as the "sandwich" estimator, and it is computed as follows:

$$(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}\left(\sum_{i=1}^{S}\mathbf{X}_i'\widehat{\mathbf{V}_i}^{-1}\widehat{\boldsymbol{\epsilon}}_i\widehat{\boldsymbol{\epsilon}}_i{}'\widehat{\mathbf{V}_i}^{-1}\mathbf{X}_i\right)(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}$$

Here, $\widehat{\boldsymbol{\epsilon}}_i = y_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}}$, S is the number of subjects, and matrices with an i subscript are those for the ith subject. You must include the SUBJECT= option in either a RANDOM or REPEATED statement for this option to take effect.

When you specify the EMPIRICAL option, PROC MIXED adjusts all standard errors and test statistics involving the fixed-effects parameters. This changes output in the following tables (listed in Table 84.26): Contrast, CorrB, CovB, Diffs, Estimates, InvCovB, LSMeans, Slices, SolutionF, Tests1–Tests3. The OUTP= and OUTPM= data sets are also affected. Finally, the Satterthwaite and Kenward-Roger degrees of freedom methods are not available if you specify the EMPIRICAL option.

IC

displays a table of various information criteria. The criteria are all in smaller-is-better form, and are described in Table 84.3.

Table 84.3 Information Criteria

Criterion	Formula	Reference
AIC	$-2\ell + 2d$	Akaike (1974)
AICC	$-2\ell + 2dn^*/(n^* - d - 1)$	Hurvich and Tsai (1989)
		Burnham and Anderson (1998)
HQIC	$-2\ell + 2d \log\log n$ for $n > 1$	Hannan and Quinn (1979)
BIC	$-2\ell + d \log n$ for $n > 0$	Schwarz (1978)
CAIC	$-2\ell + d(\log n + 1)$ for $n > 0$	Bozdogan (1987)

Here ℓ denotes the maximum value of the (possibly restricted) log likelihood, d the dimension of the model, and n the number of observations. In SAS 6 of SAS/STAT software, n equals the number of valid observations for maximum likelihood estimation and $n - p$ for restricted maximum likelihood estimation, where p equals the rank of $\mathbf{X}$. In later versions, n equals the number of effective subjects as displayed in the "Dimensions" table, unless this value equals 1, in which case n equals the number of levels of the first random effect you specify in a RANDOM statement. If the number of effective subjects equals 1 and you have no RANDOM statements, then n reverts to the SAS 6 values. For AICC (a finite-sample corrected version of AIC), n^* equals the SAS 6 values of n, unless this number is less than $d + 2$, in which case it equals $d + 2$. When $n \leq 1$, the value of the HQIC criterion is -2ℓ. When n=0, the values of the BIC and CAIC criteria are -2ℓ and $-2\ell + d$, respectively.

For restricted likelihood estimation, d equals q, the effective number of estimated covariance parameters. In SAS 6, when a parameter estimate lies on a boundary constraint, then it is still included in the calculation of d, but in later versions it is not. The most common example of this behavior is when a variance component is estimated to equal zero. For maximum likelihood estimation, d equals $q + p$ where p is by default the sum of the Type 3 degrees of freedom associated with each fixed effect or the rank of X if you specify NOTEST option. The value of d is displayed in the "Information Criteria" table as the value of Parms variable; see Table 84.27.

The ODS name of the "Information Criteria" table is InfoCrit.

INFO

is a default option. The creation of the "Model Information," "Dimensions," and "Number of Observations" tables can be suppressed by using the NOINFO option.

Note that in SAS 6 this option displays the "Model Information" and "Dimensions" tables.

ITDETAILS

displays the parameter values at each iteration and enables the writing of notes to the SAS log pertaining to "infinite likelihood" and "singularities" during Newton-Raphson iterations.

LOGNOTE

writes periodic notes to the log describing the current status of computations. It is designed for use with analyses requiring extensive CPU resources.

MAXFUNC=*number*

specifies the maximum number of likelihood evaluations in the optimization process. The default is 150.

MAXITER=*number*

specifies the maximum number of iterations. The default is 50.

METHOD=REML | ML | MIVQUE0 | TYPE1 | TYPE2 | TYPE3

specifies the estimation method for the covariance parameters. The REML specification performs residual (restricted) maximum likelihood, and it is the default method. The ML specification performs maximum likelihood, and the MIVQUE0 specification performs minimum variance quadratic unbiased estimation of the covariance parameters.

The METHOD=TYPE*n* specifications apply only to variance component models with no SUBJECT= effects and no REPEATED statement. An analysis of variance table is included in the output, and the expected mean squares are used to estimate the variance components (see Chapter 53, "The GLM Procedure," for further explanation). The resulting method-of-moment variance component estimates are used in subsequent calculations, including standard errors computed from ESTIMATE and LSMEANS statements. The ODS table names are Type1, Type2, and Type3, respectively.

MMEQ

requests that the coefficient matrix and the right-hand side of the mixed model equations be displayed. If $\widehat{\mathbf{G}}$ is nonsingular, the coefficient matrix and the right-hand side have the following form:

$$\begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}+\widehat{\mathbf{G}}^{-1} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \end{bmatrix}$$

If $\widehat{\mathbf{G}}$ is singular, the coefficient matrix and right-hand side have the following modified form:

$$\begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}\widehat{\mathbf{G}} \\ \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}\widehat{\mathbf{G}}+\mathbf{G} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\tau}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \\ \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \end{bmatrix}$$

See the section "Estimating Fixed and Random Effects in the Mixed Model" on page 6850 for further information about these equations.

MMEQSOL

requests that a solution to the mixed model equations be produced, in addition to the inverted coefficients matrix. If $\widehat{\mathbf{G}}$ is nonsingular, the formula is the same as the preceding description of the MMEQ option. If $\widehat{\mathbf{G}}$ is singular, $\widehat{\boldsymbol{\beta}}$ and $\mathbf{G}\widehat{\boldsymbol{\tau}}$ are displayed in addition to the inverse of the modified coefficient matrix.

See the section "Estimating Fixed and Random Effects in the Mixed Model" on page 6850 for further information about these equations and solution transformation.

NAMELEN< =*number*>
specifies the length to which long effect names are shortened. The default and minimum value is 20.

NOBOUND
has the same effect as the NOBOUND option in the PARMS statement.

NOCLPRINT< =*number*>
suppresses the display of the "Class Level Information" table if you do not specify *number*. If you do specify *number*, only levels with totals that are less than *number* are listed in the table.

NOINFO
suppresses the display of the "Model Information," "Dimensions," and "Number of Observations" tables.

NOITPRINT
suppresses the display of the "Iteration History" table.

NOPROFILE
includes the residual variance as part of the Newton-Raphson iterations. This option applies only to models that have a residual variance parameter. By default, this parameter is profiled out of the likelihood calculations, except when you have specified the HOLD= option in the PARMS statement.

ORD
displays ordinates of the relevant distribution in addition to *p*-values. The ordinate can be viewed as an approximate odds ratio of hypothesis probabilities.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < = (*plot-request*< (*options*) >< . . . *plot-request*< (*options*) > >) >

requests that the MIXED procedure produce statistical graphics via the Output Delivery System, provided that ODS Graphics is enabled.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc mixed data=heights plots=all;
   class Family Gender;
   model Height = Gender / residual;
   random Family Family*Gender;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

For examples of the basic statistical graphics produced by the MIXED procedure and aspects of their computation and interpretation, see the section "ODS Graphics" on page 6877.

The *global-plot-options* apply to all relevant plots generated by the MIXED procedure. The *global-plot-options* supported by the MIXED procedure follow.

Global Plot Options

OBSNO

uses the data set observation number to identify observations in tooltips, provided that the observation number can be determined. Otherwise, the number displayed in tooltips is the index of the observation as it is used in the analysis within the BY group.

ONLY

suppresses the default plots. Only the plots specifically requested are produced.

UNPACKPANEL

UNPACK

displays each graph separately. (By default, some graphs can appear together in a single panel.)

MAXPOINTS=NONE | *number*

specifies that plots with elements that require processing more than *number* points be suppressed. The default is MAXPOINTS=5000. No plots are suppressed if you specify MAXPOINTS=NONE.

Specific Plot Options

The following listing describes the specific plots and their *options*.

ALL

requests that all plots appropriate for the particular analysis be produced.

BOXPLOT < (*boxplot-options*) >

requests box plots for the effects in your model that consist of classification effects only. Note that these effects can involve more than one classification variable (interaction and nested effects), but they cannot contain any continuous variables. By default, the BOXPLOT request produces box plots based on (conditional) raw residuals for the qualifying effects in the MODEL, RANDOM, and REPEATED statements. See the discussion of the *boxplot-options* in a later section for information about how to tune your box plot request.

DISTANCE< (USEINDEX) >

requests a plot of the likelihood or restricted likelihood distance. When influence diagnostics are requested with set selection according to an effect, the USEINDEX option enables you to replace the formatted tick values on the horizontal axis with integer indices of the effect levels in order to reduce the space taken up by the horizontal plot axis.

INFLUENCEESTPLOT< (*options*) >

requests panels of the deletion estimates in an influence analysis, provided that the INFLUENCE option is specified in the MODEL statement. No plots are produced for fixed-effects parameters associated with singular columns in the **X** matrix or for covariance parameters associated with singularities in the ASYCOV matrix. By default, separate panels are produced for the fixed-effects and covariance parameters delete estimates. The FIXED and RANDOM options enable you to select these specific panels. The UNPACK option produces separate plots for each of the parameter estimates. The USEINDEX option replaces formatted tick values for the horizontal axis with integer indices.

INFLUENCESTATPANEL< (*options*) >

requests panels of influence statistics. For iterative influence analysis (see the INFLUENCE option in the MODEL statement), the panel shows the Cook's D and CovRatio statistics for fixed-effects and covariance parameters, enabling you to gauge impact on estimates and precision for both types of estimates. In noniterative analysis, only statistics for the fixed effects are plotted. The UNPACK option produces separate plots from the elements in the panel. The USEINDEX option replaces formatted tick values for the horizontal axis with integer indices.

RESIDUALPANEL < (*residual-plot-options*) >

requests a panel of raw residuals. By default, the conditional residuals are produced. See the discussion of *residual-plot-options* in a later section for information about how to tune this panel.

STUDENTPANEL < (*residual-plot-options*) >

requests a panel of studentized residuals. By default, the conditional residuals are produced. See the discussion of *residual-plot-options* in a later section for information about how to tune this panel.

PEARSONPANEL < (*residual-plot-options*) >

requests a panel of Pearson residuals. By default, the conditional residuals are produced. See the discussion of *residual-plot-options* in a later section for information about how to tune this panel.

PRESS< (USEINDEX) >
requests a plot of PRESS residuals or PRESS statistics. These are based on "leave-one-out" or "leave-set-out" prediction of the marginal mean. When influence diagnostics are requested with set selection according to an effect, the USEINDEX option enables you to replace the formatted tick values on the horizontal axis with integer indices of the effect levels in order to reduce the space taken up by the horizontal plot axis.

VCIRYPANEL < (*residual-plot-options*) >
requests a panel of residual graphics based on the scaled residuals. See the VCIRY option in the MODEL statement for details about these scaled residuals. Only the UNPACK and BOX options of the *residual-plot-options* are available for this type of residual panel.

NONE
suppresses all plots.

Residual Plot Options

The *residual-plot-options* determine both the composition of the panels and the type of residuals being plotted.

BOX
BOXPLOT
replaces the inset of summary statistics in the lower-right corner of the panel with a box plot of the residual (the "PROC GLIMMIX look").

CONDITIONAL
BLUP
constructs plots from conditional residuals.

MARGINAL
NOBLUP
constructs plots from marginal residuals.

UNPACK
produces separate plots from the elements of the panel. The inset statistics are not part of the unpack operation.

Box Plot Options

The *boxplot-options* determine whether box plots are produced for residuals or for residuals and observed values, and for which model effects the box plots are constructed. The available *boxplot-options* are as follows.

CONDITIONAL

BLUP

constructs box plots from conditional residuals—that is, residuals using the estimated BLUPs of random effects.

FIXED

produces box plots for all fixed effects (MODEL statement) consisting entirely of classification variables

GROUP

produces box plots for all GROUP= effects (RANDOM and REPEATED statement) consisting entirely of classification variables

MARGINAL

NOBLUP

constructs box plots from marginal residuals.

NPANEL=*number*

provides the ability to break a box plot into multiple graphics. If *number* is negative, no balancing of the number of boxes takes place and *number* is the maximum number of boxes per graphic. If *number* is positive, the number of boxes per graphic is balanced. For example, suppose variable A has 125 levels, and consider the following statements:

```
ods graphics on;
proc mixed plots=boxplot(npanel=20);
   class A;
   model y = A;
run;
```

The box balancing results in six plots with 18 boxes each and one plot with 17 boxes. If *number* is zero, and this is the default, all levels of the effect are displayed in a single plot.

OBSERVED

adds box plots of the observed data for the selected effects.

RANDOM

produces box plots for all random effects (RANDOM statement) consisting entirely of classification variables. This does not include effects specified in the GROUP= or SUBJECT= options of the RANDOM statement.

REPEATED

produces box plots for the repeated effects (REPEATED statement). This does not include effects specified in the GROUP= or SUBJECT= options of the REPEATED statement.

STUDENT

constructs box plots from studentized residuals rather than from raw residuals.

SUBJECT
produces box plots for all SUBJECT= effects (RANDOM and REPEATED statement) consisting entirely of classification variables.

USEINDEX
uses as the horizontal axis label the index of the effect level rather than the formatted value(s). For classification variables with many levels or model effects that involve multiple classification variables, the formatted values identifying the effect levels can take up too much space as axis tick values, leading to extensive thinning. The USEINDEX option replaces tick values constructed from formatted values with the internal level number.

Multiple Plot Requests

You can list a plot request one or more times with different options. For example, the following statements request a panel of marginal raw residuals, individual plots generated from a panel of the conditional raw residuals, and a panel of marginal studentized residuals:

```
ods graphics on;
proc mixed plots(only)=(
       ResidualPanel(marginal)
       ResidualPanel(unpack conditional)
       StudentPanel(marginal box));
```

The inset of residual statistics is replaced in this last panel by a box plot of the studentized residuals. Similarly, if you specify the INFLUENCE option in the MODEL statement, then the following statements request statistical graphics of fixed-effects deletion estimates (in a panel), covariance parameter deletion estimates (unpacked in individual plots), and box plots for the SUBJECT= and fixed classification effects based on residuals and observed values:

```
ods graphics on / imagefmt=staticmap;
proc mixed plots(only)=(
       InfluenceEstPlot(fixed)
       InfluenceEstPlot(random unpack)
       BoxPlot(observed fixed subject));
```

The STATICMAP image format enables tooltips that show, for example, values of influence diagnostics associated with a particular delete estimate.

This concludes the syntax section for the PLOTS= option in the PROC MIXED statement.

RANKS
displays the ranks of design matrices **X** and (**XZ**).

RATIO
produces the ratio of the covariance parameter estimates to the estimate of the residual variance when the latter exists in the model.

RIDGE=*number*
specifies the starting value for the minimum ridge value used in the Newton-Raphson algorithm. The default is 0.3125.

SCORING< *=number*>
requests that Fisher scoring be used in association with the estimation method up to iteration *number*, which is 0 by default. When you use the SCORING= option and PROC MIXED converges without stopping the scoring algorithm, PROC MIXED uses the expected Hessian matrix to compute approximate standard errors for the covariance parameters instead of the observed Hessian. The output from the ASYCOV and ASYCORR options is similarly adjusted.

SIGITER
is an alias for the NOPROFILE option.

UPDATE
is an alias for the LOGNOTE option.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC MIXED to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the MIXED procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Because sorting the data changes the order in which PROC MIXED reads observations, the sort order for the levels of the CLASS variable might be affected if you have specified ORDER=DATA in the PROC MIXED statement. This, in turn, affects specifications in the CONTRAST or ESTIMATE statement.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(REF=** *option***)** > . . . < *variable* < **(REF=** *option***)** > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC MIXED statement.

You can specify the following REF= option to indicate how the levels of an individual classification variable are to be ordered by enclosing it in parentheses after the variable name:

REF=*'level'* **| FIRST | LAST**

specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC MIXED's singular parameterization. You can specify the *level* of the variable to use as the reference level; specify a value that corresponds to the formatted value of the variable if a format is assigned. Alternatively, you can specify REF=FIRST to designate that the first ordered level serve as the reference, or REF=LAST to designate that the last ordered level serve as the reference. To specify that REF=FIRST or REF=LAST be used for all classification variables, use the REF= *global-option* after the slash (/) in the CLASS statement.

You can specify the following *global-options* in the CLASS statement after a slash (/):

REF=FIRST | LAST

specifies a level of all classification variables to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC MIXED's singular parameterization. Specify REF=FIRST to designate that the first ordered level for each classification variable serve as the reference. Specify REF=LAST to designate that the last ordered level serve as the reference. This option applies to all the variables specified in the CLASS statement. To specify different reference levels for different classification variables, use REF= options for individual variables.

TRUNCATE

specifies that class levels be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 84.4 summarizes the *options* available in the CODE statement.

Table 84.4 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

CONTRAST Statement

CONTRAST *'label'* < *fixed-effect values . . .* >
< | *random-effect values . . .* >, . . . < / *options* > **;**

The CONTRAST statement provides a mechanism for obtaining custom hypothesis tests. It is patterned after the CONTRAST statement in PROC GLM, although it has been extended to include random effects. This enables you to select an appropriate inference space (McLean, Sanders, and Stroup 1991).

You can test the hypothesis $\mathbf{L}'\boldsymbol{\phi} = \mathbf{0}$, where $\mathbf{L}' = (\mathbf{K}'\,\mathbf{M}')$ and $\boldsymbol{\phi}' = (\boldsymbol{\beta}'\,\boldsymbol{\gamma}')$, in several inference spaces. The inference space corresponds to the choice of $\mathbf{M}$. When $\mathbf{M} = \mathbf{0}$, your inferences apply to the entire population from which the random effects are sampled; this is known as the *broad* inference space. When all elements of $\mathbf{M}$ are nonzero, your inferences apply only to the observed levels of the random effects. This is known as the *narrow* inference space, and you can also choose it by specifying all of the random effects as fixed. The GLM procedure uses the narrow inference space. Finally, by setting to zero the portions of $\mathbf{M}$ corresponding to selected main effects and interactions, you can choose *intermediate* inference spaces. The broad inference space is usually the most appropriate, and it is used when you do not specify any random effects in the CONTRAST statement.

The CONTRAST statement has the following arguments:

label identifies the contrast in the table. A label is required for every contrast specified. Labels can be up to 200 characters and must be enclosed in quotes.

fixed-effect identifies an effect that appears in the MODEL statement. The keyword INTERCEPT can be used as an effect when an intercept is fitted in the model. You do not need to include all effects that are in the MODEL statement.

random-effect identifies an effect that appears in the RANDOM statement. The first random effect must follow a vertical bar (|); however, random effects do not have to be specified.

values are constants that are elements of the $\mathbf{L}$ matrix associated with the fixed and random effects.

The rows of $\mathbf{L}'$ are specified in order and are separated by commas. The rows of the $\mathbf{K}'$ component of $\mathbf{L}'$ are specified on the left side of the vertical bars (|). These rows test the fixed effects and are, therefore, checked for estimability. The rows of the $\mathbf{M}'$ component of $\mathbf{L}'$ are specified on the right side of the vertical bars. They test the random effects, and no estimability checking is necessary.

If PROC MIXED finds the fixed-effects portion of the specified contrast to be nonestimable (see the SINGULAR= option), then it displays a message in the log.

The following CONTRAST statement reproduces the F test for the effect A in the split-plot example (see Example 84.1):

```
contrast 'A broad'
          A  1 -1 0    A*B  .5 .5 -.5 -.5 0 0 ,
          A  1 0 -1    A*B  .5 .5 0 0 -.5 -.5 / df=6;
```

Note that no random effects are specified in the preceding contrast; thus, the inference space is broad. The resulting F test has two numerator degrees of freedom because $\mathbf{L}'$ has two rows. The denominator degrees of freedom is, by default, the residual degrees of freedom (9), but the DF= option changes the denominator degrees of freedom to 6.

The following CONTRAST statement reproduces the F test for A when Block and A*Block are considered fixed effects (the narrow inference space):

```
contrast 'A narrow'
          A        1 -1 0
          A*B     .5 .5 -.5 -.5 0 0 |
          A*Block .25  .25  .25  .25
                 -.25 -.25 -.25 -.25
                   0    0    0    0 ,
          A        1 0 -1
          A*B     .5 .5 0 0 -.5 -.5 |
          A*Block .25  .25  .25  .25
                   0    0    0    0
                 -.25 -.25 -.25 -.25 ;
```

The preceding contrast does not contain coefficients for B and Block, because they cancel out in estimated differences between levels of A. Coefficients for B and Block are necessary to estimate the mean of one of the levels of A in the narrow inference space (see Example 84.1).

If the elements of $\mathbf{L}$ are not specified for an effect that contains a specified effect, then the elements of the specified effect are automatically "filled in" over the levels of the higher-order effect. This feature is designed to preserve estimability for cases where there are complex higher-order effects. The coefficients for the

higher-order effect are determined by equitably distributing the coefficients of the lower-level effect, as in the construction of least squares means. In addition, if the intercept is specified, it is distributed over all classification effects that are not contained by any other specified effect. If an effect is not specified and does not contain any specified effects, then all of its coefficients in $\mathbf{L}$ are set to 0. You can override this behavior by specifying coefficients for the higher-order effect.

If too many values are specified for an effect, the extra ones are ignored; if too few are specified, the remaining ones are set to 0. If no random effects are specified, the vertical bar can be omitted; otherwise, it must be present. If a SUBJECT= effect is used in the RANDOM statement, then the coefficients specified for the effects in the RANDOM statement are equitably distributed across the levels of the SUBJECT effect. You can use the E option to see exactly which $\mathbf{L}$ matrix is used.

The SUBJECT and GROUP options in the CONTRAST statement are useful for the case when a SUBJECT= or GROUP= variable appears in the RANDOM statement, and you want to contrast different subjects or groups. By default, CONTRAST statement coefficients on random effects are distributed equally across subjects and groups.

PROC MIXED handles missing level combinations of classification variables similarly to the way PROC GLM does. Both procedures delete fixed-effects parameters corresponding to missing levels in order to preserve estimability. However, PROC MIXED does not delete missing level combinations for random-effects parameters because linear combinations of the random-effects parameters are always estimable. These conventions can affect the way you specify your CONTRAST coefficients.

The CONTRAST statement computes the statistic

$$F = \frac{\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}' \mathbf{L}(\mathbf{L}'\widehat{\mathbf{C}}\mathbf{L})^{-1}\mathbf{L}' \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}}{r}$$

where $r = \mathrm{rank}(\mathbf{L}'\widehat{\mathbf{C}}\mathbf{L})$, and approximates its distribution with an F distribution. In this expression, $\widehat{\mathbf{C}}$ is an estimate of the generalized inverse of the coefficient matrix in the mixed model equations. For more information about this F statistic, see the section "Inference and Test Statistics" on page 6852.

The numerator degrees of freedom in the F approximation are $r = \mathrm{rank}(\mathbf{L}'\widehat{\mathbf{C}}\mathbf{L})$, and the denominator degrees of freedom are taken from the "Tests of Fixed Effects" table and corresponds to the final effect you list in the CONTRAST statement. You can change the denominator degrees of freedom by using the DF= option.

You can specify the following *options* in the CONTRAST statement after a slash (/).

CHISQ

requests that chi-square tests be performed in addition to any F tests. A chi-square statistic equals its corresponding F statistic times the associate numerator degrees of freedom, and the same degrees of freedom are used to compute the p-value for the chi-square test. This p-value is always less than that for the F -test, as it effectively corresponds to an F test with infinite denominator degrees of freedom.

DF=*number*

specifies the denominator degrees of freedom for the F test. For the degrees-of-freedom methods DDFM=BETWITHIN, DDFM=CONTAIN, and DDFM=RESIDUAL, the default is the denominator degrees of freedom taken from the "Tests of Fixed Effects" table and corresponds to the final effect you list in the CONTRAST statement. For DDFM=SATTERTHWAITE, DDFM=KENWARDROGER, and DDFM=KENWARDROGER2, the denominator degrees of freedom are computed separately for each contrast.

E

requests that the **L** matrix coefficients for the contrast be displayed. The ODS name of the "L Matrix Coefficients" table is Coef.

GROUP *coeffs*

GRP *coeffs*

sets up random-effect contrasts between different groups when a GROUP= variable appears in the RANDOM statement. By default, CONTRAST statement coefficients on random effects are distributed equally across groups.

SINGULAR=*number*

tunes the estimability checking. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the absolute value of the element of $\mathbf{v}$ with the largest absolute value. If ABS($\mathbf{K}' - \mathbf{K}'\mathbf{T}$) is greater than c**number* for any row of $\mathbf{K}'$ in the contrast, then $\mathbf{K}$ is declared nonestimable. Here $\mathbf{T}$ is the Hermite form matrix $(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}$, and c is ABS($\mathbf{K}'$) except when it equals 0, and then c is 1. The value for *number* must be between 0 and 1; the default is 1E–4.

SUBJECT *coeffs*

SUB *coeffs*

sets up random-effect contrasts between different subjects when a SUBJECT= variable appears in the RANDOM statement. By default, CONTRAST statement coefficients on random effects are distributed equally across subjects.

ESTIMATE Statement

ESTIMATE *'label'* < *fixed-effect values . . .* >
< | *random-effect values . . .* > < / *options* > **;**

The ESTIMATE statement is exactly like a CONTRAST statement, except only one-row $\mathbf{L}$ matrices are permitted. The actual estimate, $\mathbf{L}'\widehat{\boldsymbol{\phi}}$, is displayed along with its approximate standard error. An approximate t test that $\mathbf{L}'\widehat{\boldsymbol{\phi}} = 0$ is also produced.

PROC MIXED selects the degrees of freedom to match those displayed in the "Tests of Fixed Effects" table for the final effect you list in the ESTIMATE statement. You can modify the degrees of freedom by using the DF= option.

If PROC MIXED finds the fixed-effects portion of the specified estimate to be nonestimable, then it displays "Non-est" for the estimate entries.

The following examples of ESTIMATE statements compute the mean of the first level of A in the split-plot example (see Example 84.1) for various inference spaces:

```
estimate 'A1 mean narrow'   intercept 1
                            A 1 B .5 .5 A*B .5 .5 |
                            block   .25 .25 .25 .25
                            A*Block .25 .25 .25 .25
                                     0   0   0   0
                                     0   0   0   0;
estimate 'A1 mean intermed' intercept 1
                            A 1 B .5 .5 A*B .5 .5 |
```

```
                                Block .25 .25 .25 .25;
  estimate 'A1 mean broad'      intercept 1
                                A 1 B .5 .5 A*B .5 .5;
```

The construction of the **L** vector for an ESTIMATE statement follows the same rules as listed under the CONTRAST statement.

Table 84.5 summarizes the *options* available in the ESTIMATE statement.

Table 84.5 ESTIMATE Statement Options

Option	Description
ALPHA=	Specifies the confidence level
CL	Constructs *t*-type confidence limits
DF=	Specifies the degrees of freedom
DIVISOR=	Specifies a value by which to divide all coefficients
E	Displays the **L** matrix coefficients
GROUP	Sets up random-effect contrasts between different groups
LOWER	Performs lower-tailed tests
SINGULAR=	Tunes the estimability checking
SUBJECT	Sets up random-effect contrasts between different subjects
UPPER	Performs upper-tailed tests

You can specify the following *options* in the ESTIMATE statement after a slash (/).

ALPHA=*number*

requests that a *t*-type confidence interval be constructed with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

CL

requests that *t*-type confidence limits be constructed. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

DF=*number*

specifies the degrees of freedom for the *t* test and confidence limits. The default is the denominator degrees of freedom taken from the "Tests of Fixed Effects" table and corresponds to the final effect you list in the ESTIMATE statement.

DIVISOR=*number*

specifies a value by which to divide all coefficients so that fractional coefficients can be entered as integer numerators.

E

requests that the **L** matrix coefficients be displayed. The ODS name of this "L Matrix Coefficients" table is "Coef."

GROUP *coeffs*

GRP *coeffs*

sets up random-effect contrasts between different groups when a GROUP= variable appears in the RANDOM statement. By default, ESTIMATE statement coefficients on random effects are distributed equally across groups.

LOWER

LOWERTAILED

requests that the *p*-value for the *t* test be based only on values less than the *t* statistic. A two-tailed test is the default. A lower-tailed confidence limit is also produced if you specify the CL option.

SINGULAR=*number*

tunes the estimability checking as documented for the SINGULAR= option in the CONTRAST statement.

SUBJECT *coeffs*

SUB *coeffs*

sets up random-effect contrasts between different subjects when a SUBJECT= variable appears in the RANDOM statement. By default, ESTIMATE statement coefficients on random effects are distributed equally across subjects. For example, the ESTIMATE statement in the following code from Example 84.5 constructs the difference between the random slopes of the first two batches.

```
proc mixed data=rc;
   class batch;
   model y = month / s;
   random int month / type=un sub=batch s;
   estimate 'slope b1 - slope b2' | month 1 / subject 1 -1;
run;
```

UPPER

UPPERTAILED

requests that the *p*-value for the *t* test be based only on values greater than the *t* statistic. A two-tailed test is the default. An upper-tailed confidence limit is also produced if you specify the CL option.

ID Statement

ID *variables* **;**

The ID statement specifies which variables from the input data set are to be included in the OUTP= and OUTPM= data sets from the MODEL statement. If you do not specify an ID statement, then all variables are included in these data sets. Otherwise, only the variables you list in the ID statement are included. Specifying an ID statement with no variables prevents any variables from being included in these data sets.

LSMEANS Statement

LSMEANS *fixed-effects* < / *options* > **;**

The LSMEANS statement computes least squares means (LS-means) of fixed effects. As in the GLM procedure, LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs. The **L** matrix constructed to compute them is the same as the **L** matrix formed in PROC GLM; however, the standard errors are adjusted for the covariance parameters in the model.

Each LS-mean is computed as $\mathbf{L}\widehat{\boldsymbol{\beta}}$, where **L** is the coefficient matrix associated with the least squares mean and $\widehat{\boldsymbol{\beta}}$ is the estimate of the fixed-effects parameter vector (see the section "Estimating Fixed and Random Effects in the Mixed Model" on page 6850). The approximate standard errors for the LS-mean is computed as the square root of $\mathbf{L}(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}\mathbf{L}'$.

LS-means can be computed for any effect in the MODEL statement that involves CLASS variables. You can specify multiple effects in one LSMEANS statement or in multiple LSMEANS statements, and all LSMEANS statements must appear after the MODEL statement. As in the ESTIMATE statement, the **L** matrix is tested for estimability, and if this test fails, PROC MIXED displays "Non-est" for the LS-means entries.

Assuming the LS-mean is estimable, PROC MIXED constructs an approximate *t* test to test the null hypothesis that the associated population quantity equals zero. By default, the denominator degrees of freedom for this test are the same as those displayed for the effect in the "Tests of Fixed Effects" table (see the section "Default Output" on page 6868).

Table 84.6 summarizes the *options* available in the LSMEANS statement. All LSMEANS *options* are subsequently discussed in alphabetical order.

Table 84.6 Summary of LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Requests differences of LS-means
OM	Specifies weighting scheme for LS-mean computation
SINGULAR=	Tunes estimability checking
SLICE=	Partitions *F* tests (simple effects)
Degrees of Freedom and *p*-Values	
ADJDFE=	Determines whether to compute rowwise denominator degrees of freedom by using DDFM=SATTERTHWAITE, DDFM=KENWARDROGER, or DDFM=KENWARDROGER2
ADJUST=	Determines the method for multiple comparison adjustment of LS-mean differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
DF=	Assigns specific value to degrees of freedom for tests and confidence limits

Table 84.6 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for means and or mean differences
CORR	Displays correlation matrix of LS-means
COV	Displays covariance matrix of LS-means
E	Prints the **L** matrix

You can specify the following *options* in the LSMEANS statement after a slash (/).

ADJDFE=SOURCE | ROW

specifies how denominator degrees of freedom are determined when *p*-values and confidence limits are adjusted for multiple comparisons with the ADJUST= option. When you do not specify the ADJDFE= option, or when you specify ADJDFE=SOURCE, the denominator degrees of freedom for multiplicity-adjusted results are the denominator degrees of freedom for the LS-mean effect in the "Type 3 Tests of Fixed Effects" table. When you specify ADJDFE=ROW, the denominator degrees of freedom for multiplicity-adjusted results correspond to the degrees of freedom displayed in the DF column of the "Differences of Least Squares Means" table.

The ADJDFE=ROW setting is particularly useful if you want multiplicity adjustments to take into account that denominator degrees of freedom are not constant across LS-mean differences. This can be the case, for example, when the DDFM=SATTERTHWAITE, DDFM=KENWARDROGER, or DDFM=KENWARDROGER2 degrees-of-freedom method is in effect.

In one-way models with heterogeneous variance, combining certain ADJUST= options with the ADJDFE=ROW option corresponds to particular methods of performing multiplicity adjustments in the presence of heteroscedasticity. For example, the following statements fit a heteroscedastic one-way model and perform Dunnett's T3 method (Dunnett 1980), which is based on the studentized maximum modulus (ADJUST=SMM):

```
proc mixed;
   class A;
   model y = A / ddfm=satterth;
   repeated / group=A;
   lsmeans A / adjust=smm adjdfe=row;
run;
```

If you combine the ADJDFE=ROW option with ADJUST=SIDAK, the multiplicity adjustment corresponds to the T2 method of Tamhane (1979), whereas ADJUST=TUKEY corresponds to the method of Games-Howell (Games and Howell 1976). Note that ADJUST=TUKEY gives the exact results for the case of fractional degrees of freedom in the one-way model, but it does not take into account that the degrees of freedom are subject to variability. A more conservative method, such as ADJUST=SMM, might protect the overall error rate better.

Unless the ADJUST= option of the LSMEANS statement is specified, the ADJDFE= option has no effect.

ADJUST=BON

ADJUST=DUNNETT

ADJUST=SCHEFFE

ADJUST=SIDAK

ADJUST=SIMULATE< (*sim-options*) >

ADJUST=SMM | GT2

ADJUST=TUKEY

requests a multiple comparison adjustment for the p-values and confidence limits for the differences of LS-means. By default, PROC MIXED adjusts all pairwise differences unless you specify ADJUST=DUNNETT, in which case PROC MIXED analyzes all differences with a control level. The ADJUST= option implies the DIFF option.

The BON (Bonferroni) and SIDAK adjustments involve correction factors described in Chapter 53, "The GLM Procedure," and Chapter 86, "The MULTTEST Procedure"; also see Westfall and Young (1993) and Westfall et al. (1999). When you specify ADJUST=TUKEY and your data are unbalanced, PROC MIXED uses the approximation described in Kramer (1956). Similarly, when you specify ADJUST=DUNNETT and the LS-means are correlated, PROC MIXED uses the factor-analytic covariance approximation described in Hsu (1992). The preceding references also describe the SCHEFFE and SMM adjustments.

The SIMULATE adjustment computes adjusted p-values and confidence limits from the simulated distribution of the maximum or maximum absolute value of a multivariate t random vector. All covariance parameters except the residual variance are fixed at their estimated values throughout the simulation, potentially resulting in some underdispersion. The simulation estimates q, the true $(1-\alpha)$ quantile, where $1-\alpha$ is the confidence coefficient. The default α is 0.05, and you can change this value with the ALPHA= option in the LSMEANS statement.

The number of samples is set so that the tail area for the simulated q is within γ of $1-\alpha$ with $100(1-\epsilon)\%$ confidence. In equation form,

$$P(|F(\widehat{q}) - (1-\alpha)| \leq \gamma) = 1 - \epsilon$$

where $\hat{q}$ is the simulated q and F is the true distribution function of the maximum; see Edwards and Berry (1987) for details. By default, $\gamma = 0.005$ and $\epsilon = 0.01$, placing the tail area of $\hat{q}$ within 0.005 of 0.95 with 99% confidence. The ACC= and EPS= *sim-options* reset γ and ϵ, respectively; the NSAMP= *sim-option* sets the sample size directly; and the SEED= *sim-option* specifies an integer used to start the pseudo-random number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. For additional descriptions of these and other simulation options, see the section "LSMEANS Statement" on page 4172 in Chapter 53, "The GLM Procedure."

ALPHA=*number*

requests that a t-type confidence interval be constructed for each of the LS-means with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

AT *variable* **=** *value*

AT (*variable-list***)= (***value-list***)**

AT MEANS

enables you to modify the values of the covariates used in computing LS-means. By default, all covariate effects are set equal to their mean values for computation of standard LS-means. The AT option enables you to assign arbitrary values to the covariates. Additional columns in the output table indicate the values of the covariates.

If there is an effect containing two or more covariates, the AT option sets the effect equal to the product of the individual means rather than the mean of the product (as with standard LS-means calculations). The AT MEANS option sets covariates equal to their mean values (as with standard LS-means) and incorporates this adjustment to crossproducts of covariates.

As an example, consider the following invocation of PROC MIXED:

```
proc mixed;
   class A;
   model Y = A X1 X2 X1*X2;
   lsmeans A;
   lsmeans A / at means;
   lsmeans A / at X1=1.2;
   lsmeans A / at (X1 X2)=(1.2 0.3);
run;
```

For the first two LSMEANS statements, the LS-means coefficient for X1 is $\overline{x_1}$ (the mean of X1) and for X2 is $\overline{x_2}$ (the mean of X2). However, for the first LSMEANS statement, the coefficient for X1*X2 is $\overline{x_1 x_2}$, but for the second LSMEANS statement, the coefficient is $\overline{x_1} \cdot \overline{x_2}$. The third LSMEANS statement sets the coefficient for X1 equal to 1.2 and leaves it at $\overline{x_2}$ for X2, and the final LSMEANS statement sets these values to 1.2 and 0.3, respectively.

If a WEIGHT variable is present, it is used in processing AT variables. Also, observations with missing dependent variables are included in computing the covariate means, unless these observations form a missing cell and the FULLX option in the MODEL statement is not in effect. You can use the E option in conjunction with the AT option to check that the modified LS-means coefficients are the ones you want.

The AT option is disabled if you specify the BYLEVEL option.

BYLEVEL

requests PROC MIXED to process the OM data set by each level of the LS-mean effect (LSMEANS effect) in question. For more details, see the OM option later in this section.

CL

requests that *t*-type confidence limits be constructed for each of the LS-means. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

CORR

displays the estimated correlation matrix of the least squares means as part of the "Least Squares Means" table.

COV

displays the estimated covariance matrix of the least squares means as part of the "Least Squares Means" table.

DF=*number*

specifies the degrees of freedom for the *t* test and confidence limits. The default is the denominator degrees of freedom taken from the "Tests of Fixed Effects" table corresponding to the LS-means effect, unless you specify the DDFM=SATTERTHWAITE, DDFM=KENWARDROGER, or DDFM=KENWARDROGER2 option in the MODEL statement. For these DDFM= methods, degrees of freedom are determined separately for each test; for more information, see the DDFM= option.

DIFF<=*difftype***>**

PDIFF<=*difftype***>**

requests that differences of the LS-means be displayed. The optional *difftype* specifies which differences to produce, with possible values being ALL, CONTROL, CONTROLL, and CONTROLU. The *difftype* ALL requests all pairwise differences, and it is the default. The *difftype* CONTROL requests the differences with a control, which, by default, is the first level of each of the specified LSMEANS effects.

To specify which levels of the effects are the controls, list the quoted formatted values in parentheses after the keyword CONTROL. For example, if the effects A, B, and C are classification variables, each having two levels, 1 and 2, the following LSMEANS statement specifies the (1,2) level of A*B and the (2,1) level of B*C as controls:

```
lsmeans A*B B*C / diff=control('1' '2', '2' '1');
```

For multiple effects, the results depend upon the order of the list, and so you should check the output to make sure that the controls are correct.

Two-tailed tests and confidence limits are associated with the CONTROL *difftype*. For one-tailed results, use either the CONTROLL or CONTROLU *difftype*. The CONTROLL *difftype* tests whether the noncontrol levels are significantly smaller than the control; the upper confidence limits for the control minus the noncontrol levels are considered to be infinity and are displayed as missing. Conversely, the CONTROLU *difftype* tests whether the noncontrol levels are significantly larger than the control; the upper confidence limits for the noncontrol levels minus the control are considered to be infinity and are displayed as missing.

If you want to perform multiple comparison adjustments on the differences of LS-means, you must specify the ADJUST= option.

The differences of the LS-means are displayed in a table titled "Differences of Least Squares Means." The ODS table name is Diffs.

E

requests that the **L** matrix coefficients for all LSMEANS effects be displayed. The ODS name of this "**L** Matrix Coefficients" table is Coef.

OM< =*OM-data-set* >

OBSMARGINS< =*OM-data-set* >

specifies a potentially different weighting scheme for the computation of LS-means coefficients. The standard LS-means have equal coefficients across classification effects; however, the OM option changes these coefficients to be proportional to those found in *OM-data-set*. This adjustment is reasonable when you want your inferences to apply to a population that is not necessarily balanced but has the margins observed in *OM-data-set*.

By default, *OM-data-set* is the same as the analysis data set. You can optionally specify another data set that describes the population for which you want to make inferences. This data set must contain all model variables except for the dependent variable (which is ignored if it is present). In addition, the levels of all CLASS variables must be the same as those occurring in the analysis data set. Specifying an *OM-data-set* enables you to construct arbitrarily weighted LS-means.

In computing the observed margins, PROC MIXED uses all observations for which there are no missing or invalid independent variables, including those for which there are missing dependent variables. Also, if *OM-data-set* has a WEIGHT variable, PROC MIXED uses weighted margins to construct the LS-means coefficients. If *OM-data-set* is balanced, the LS-means are unchanged by the OM option.

The BYLEVEL option modifies the observed-margins LS-means. Instead of computing the margins across all of the *OM-data-set*, PROC MIXED computes separate margins for each level of the LSMEANS effect in question. In this case the resulting LS-means are actually equal to raw means for fixed-effects models and certain balanced random-effects models, but their estimated standard errors account for the covariance structure that you have specified. If the AT option is specified, the BYLEVEL option disables it.

You can use the E option in conjunction with either the OM or BYLEVEL option to check that the modified LS-means coefficients are the ones you want. It is possible that the modified LS-means are not estimable when the standard ones are, or vice versa. Nonestimable LS-means are noted as "Non-est" in the output.

PDIFF

is the same as the DIFF option.

SINGULAR=*number*

tunes the estimability checking as documented for the SINGULAR= option in the CONTRAST statement.

SLICE= *fixed-effect* | (*fixed-effects*)

specifies effects by which to partition interaction LSMEANS effects. This can produce what are known as tests of simple effects (Winer 1971). For example, suppose that A*B is significant, and you want to test the effect of A for each level of B. The appropriate LSMEANS statement is as follows:

```
lsmeans A*B / slice=B;
```

This code tests for the simple main effects of A for B, which are calculated by extracting the appropriate rows from the coefficient matrix for the A*B LS-means and by using them to form an F test. For more information about this F test, see the section "Inference and Test Statistics" on page 6852.

The SLICE option produces a table titled "Tests of Effect Slices." The ODS table name is Slices.

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 84.7 summarizes the *options* available in the LSMESTIMATE statement.

Table 84.7 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference

Table 84.7 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

MODEL *dependent* **=** < *fixed-effects* > < / *options* > **;**

The MODEL statement names a single dependent variable and the fixed effects, which determine the **X** matrix of the mixed model (see the section "Parameterization of Mixed Models" on page 6855 for details). The specification of effects is the same as in the GLM procedure; however, unlike PROC GLM, you do not specify random effects in the MODEL statement. The MODEL statement is required.

An intercept is included in the fixed-effects model by default. If no fixed effects are specified, only this intercept term is fit. The intercept can be removed by using the NOINT option.

Table 84.8 summarizes the *options* available in the MODEL statement. These are subsequently discussed in detail in alphabetical order.

Table 84.8 Summary of MODEL Statement Options

Option	Description
Model Building	
NOINT	Excludes fixed-effect intercept from model
Statistical Computations	
ALPHA=α	Determines the confidence level $(1-\alpha)$ for fixed effects

Table 84.8 *continued*

Option	Description
ALPHAP=α	Determines the confidence level $(1-\alpha)$ for predicted values
CHISQ	Requests chi-square tests
DDF=	Specifies denominator degrees of freedom (list)
DDFM=	Specifies the method for computing denominator degrees of freedom
HTYPE=	Selects the type of hypothesis test
INFLUENCE	Requests influence and case-deletion diagnostics
NOTEST	Suppresses hypothesis tests for the fixed effects
OUTP=	Specifies output data set for predicted values and related quantities
OUTPM=	Specifies output data set for predicted means and related quantities
RESIDUAL	Adds Pearson-type and studentized residuals to output data sets
VCIRY	Adds scaled marginal residual to output data sets
Statistical Output	
CL	Displays confidence limits for fixed-effects parameter estimates
CORRB	Displays correlation matrix of fixed-effects parameter estimates
COVB	Displays covariance matrix of fixed-effects parameter estimates
COVBI	Displays inverse covariance matrix of fixed-effects parameter estimates
E, E1, E2, E3	Displays **L** matrix coefficients
INTERCEPT	Adds a row for the intercept to test tables
SOLUTION	Displays fixed-effects parameter estimates (and scale parameter in GLM models)
Singularity Tolerances	
SINGCHOL=	Tunes sensitivity in computing Cholesky roots
SINGRES=	Tunes singularity criterion for residual variance
SINGULAR=	Tunes the sensitivity in sweeping
ZETA=	Tunes the sensitivity in forming Type 3 functions

You can specify the following *options* in the MODEL statement after a slash (/).

ALPHA=*number*

requests that a t-type confidence interval be constructed for each of the fixed-effects parameters with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

ALPHAP=*number*

requests that a t-type confidence interval be constructed for the predicted values with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

CHISQ

requests that chi-square tests be performed for all specified effects in addition to the F tests. Type 3 tests are the default; you can produce the Type 1 and Type 2 tests by using the HTYPE= option.

CL

requests that t-type confidence limits be constructed for each of the fixed-effects parameter estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

CONTAIN

has the same effect as the DDFM=CONTAIN option.

CORRB

produces the approximate correlation matrix of the fixed-effects parameter estimates. The ODS name of this table is CorrB.

COVB

produces the approximate variance-covariance matrix of the fixed-effects parameter estimates $\widehat{\boldsymbol{\beta}}$. By default, this matrix equals $(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}$ and results from sweeping $(\mathbf{X}\ \mathbf{y})'\widehat{\mathbf{V}}^{-1}(\mathbf{X}\ \mathbf{y})$ on all but its last pivot and removing the y border. The EMPIRICAL option in the PROC MIXED statement changes this matrix into "empirical sandwich" form. The ODS name of this table is CovB. If the degrees-of-freedom method of Kenward and Roger (1997) is in effect (DDFM=KENWARDROGER or DDFM=KENWARDROGER2), the COVB matrix changes because the method entails an adjustment of the variance-covariance matrix of the fixed effects by the method proposed by Prasad and Rao (1990); Harville and Jeske (1992). See also Kackar and Harville (1984).

COVBI

produces the inverse of the approximate variance-covariance matrix of the fixed-effects parameter estimates. The ODS name of this table is InvCovB.

DDF=*value-list*

enables you to specify your own denominator degrees of freedom for the fixed effects. The *value-list* specification is a list of numbers or missing values (.) separated by commas. The degrees of freedom should be listed in the order in which the effects appear in the "Tests of Fixed Effects" table. If you want to retain the default degrees of freedom for a particular effect, use a missing value for its location in the list. For example, the following statement assigns 3 denominator degrees of freedom to A and 4.7 to A*B, while those for B remain the same:

```
model Y = A B A*B / ddf=3,.,4.7;
```

If you specify DDFM=SATTERTHWAITE, DDFM=KENWARDROGER, or DDFM=KENWARDROGER2, the DDF= option has no effect.

DDFM=

DDFM=CONTAIN

DDFM=BETWITHIN

DDFM=RESIDUAL

DDFM=SATTERTHWAITE

DDFM=KENWARDROGER< (FIRSTORDER) >

DDFM=KENWARDROGER< (LINEAR) >

DDFM=KENWARDROGER2

specifies the method for computing the denominator degrees of freedom for the tests of fixed effects resulting from the MODEL, CONTRAST, ESTIMATE, and LSMEANS statements.

Table 84.9 lists syntax aliases for the degrees-of-freedom methods.

Table 84.9 Aliases for DDFM= Option

DDFM= Option	Alias
BETWITHIN	BW
CONTAIN	CON
KENWARDROGER	KENROG, KR
KENWARDROGER2	KENROG2, KR2
RESIDUAL	RES
SATTERTHWAITE	SATTERTH, SAT

The DDFM=CONTAIN option invokes the *containment method* to compute denominator degrees of freedom, and it is the default when you specify a RANDOM statement. The containment method is carried out as follows: Denote the fixed effect in question A, and search the RANDOM effect list for the effects that *syntactically* contain A. For example, the random effect B(A) contains A, but the random effect C does not, even if it has the same levels as B(A).

Among the random effects that contain A, compute their rank contribution to the $(\mathbf{X}\ \mathbf{Z})$ matrix. The DDF assigned to A is the smallest of these rank contributions. If no effects are found, the DDF for A is set equal to the residual degrees of freedom, $N - \mathrm{rank}(\mathbf{X}\ \mathbf{Z})$. This choice of DDF matches the tests performed for balanced split-plot designs and should be adequate for moderately unbalanced designs.

CAUTION: If you have a $\mathbf{Z}$ matrix with a large number of columns, the overall memory requirements and the computing time after convergence can be substantial for the containment method. If it is too large, you might want to use the DDFM=BETWITHIN option.

The DDFM=BETWITHIN option is the default for REPEATED statement specifications (with no RANDOM statements). It is computed by dividing the residual degrees of freedom into between-subject and within-subject portions. PROC MIXED then checks whether a fixed effect changes within any subject. If so, it assigns within-subject degrees of freedom to the effect; otherwise, it assigns the between-subject degrees of freedom to the effect (see Schluchter and Elashoff 1990). If there are multiple within-subject effects containing classification variables, the within-subject degrees of freedom are partitioned into components corresponding to the subject-by-effect interactions.

One exception to the preceding method is the case where you have specified no RANDOM statements and a REPEATED statement with the TYPE=UN option. In this case, all effects are assigned the

between-subject degrees of freedom to provide for better small-sample approximations to the relevant sampling distributions. DDFM=KENWARDROGER or DDFM=KENWARDROGER2 might be a better option to try for this case.

The DDFM=RESIDUAL option performs all tests by using the residual degrees of freedom, $n - \mathrm{rank}(\mathbf{X})$, where n is the number of observations.

The DDFM=SATTERTHWAITE option performs a general Satterthwaite approximation for the denominator degrees of freedom, computed as follows. Suppose $\boldsymbol{\theta}$ is the vector of unknown parameters in $\mathbf{V}$, and suppose $\mathbf{C} = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}$, where $^{-}$ denotes a generalized inverse. Let $\widehat{\mathbf{C}}$ and $\widehat{\boldsymbol{\theta}}$ be the corresponding estimates.

Consider the one-dimensional case, and consider ℓ to be a vector defining an estimable linear combination of $\boldsymbol{\beta}$. The Satterthwaite degrees of freedom for the t statistic

$$t = \frac{\ell\widehat{\boldsymbol{\beta}}}{\sqrt{\ell\hat{\mathbf{C}}\ell'}}$$

is computed as

$$\nu = \frac{2(\ell\hat{\mathbf{C}}\ell')^2}{\mathbf{g}'\mathbf{A}\mathbf{g}}$$

where $\mathbf{g}$ is the gradient of $\ell\mathbf{C}\ell'$ with respect to $\boldsymbol{\theta}$, evaluated at $\widehat{\boldsymbol{\theta}}$, and $\mathbf{A}$ is the asymptotic variance-covariance matrix of $\widehat{\boldsymbol{\theta}}$ obtained from the second derivative matrix of the likelihood equations.

For the multidimensional case, let $\mathbf{L}$ be an estimable contrast matrix and denote the rank of $\mathbf{L}\widehat{\mathbf{C}}\mathbf{L}'$ as $q > 1$. The Satterthwaite denominator degrees of freedom for the F statistic

$$F = \frac{\widehat{\boldsymbol{\beta}}'\mathbf{L}'(\mathbf{L}\widehat{\mathbf{C}}\mathbf{L}')^{-1}\mathbf{L}\widehat{\boldsymbol{\beta}}}{q}$$

are computed by first performing the spectral decomposition $\mathbf{L}\widehat{\mathbf{C}}\mathbf{L}' = \mathbf{P}'\mathbf{D}\mathbf{P}$, where $\mathbf{P}$ is an orthogonal matrix of eigenvectors and $\mathbf{D}$ is a diagonal matrix of eigenvalues, both of dimension $q \times q$. Define ℓ_m to be the mth row of $\mathbf{PL}$, and let

$$\nu_m = \frac{2(D_m)^2}{\mathbf{g}_m'\mathbf{A}\mathbf{g}_m}$$

where D_m is the mth diagonal element of $\mathbf{D}$ and $\mathbf{g}_m$ is the gradient of $\ell_m\mathbf{C}\ell_m'$ with respect to $\boldsymbol{\theta}$, evaluated at $\widehat{\boldsymbol{\theta}}$. Then let

$$E = \sum_{m=1}^{q} \frac{\nu_m}{\nu_m - 2} I(\nu_m > 2)$$

where the indicator function eliminates terms for which $\nu_m \leq 2$. The degrees of freedom for F are then computed as

$$\nu = \frac{2E}{E - q}$$

provided $E > q$; otherwise ν is set to zero.

This method is a generalization of the techniques described in Giesbrecht and Burns (1985); McLean and Sanders (1988); Fai and Cornelius (1996). The method can also include estimated random effects. In this case, append $\widehat{\boldsymbol{\gamma}}$ to $\widehat{\boldsymbol{\beta}}$ and change $\widehat{\mathbf{C}}$ to be the inverse of the coefficient matrix in the mixed model equations. The calculations require extra memory to hold c matrices that are the size of the mixed model equations, where c is the number of covariance parameters. In the notation of Table 84.29, this is approximately $8q(p+g)(p+g)/2$ bytes. Extra computing time is also required to process these matrices. The Satterthwaite method implemented here is intended to produce an accurate F approximation; however, the results can differ from those produced by PROC GLM. Also, the small sample properties of this approximation have not been extensively investigated for the various models available with PROC MIXED.

The DDFM=KENWARDROGER option performs the degrees of freedom calculations detailed by Kenward and Roger (1997). This approximation involves inflating the estimated variance-covariance matrix of the fixed and random effects by the method proposed by Prasad and Rao (1990) and Harville and Jeske (1992), see also Kackar and Harville (1984). Satterthwaite-type degrees of freedom are then computed based on this adjustment. By default, the observed information matrix of the covariance parameter estimates is used in the calculations. For covariance structures that have nonzero second derivatives with respect to the covariance parameters, the Kenward-Roger covariance matrix adjustment includes a second-order term. This term can result in standard error shrinkage. Also, the resulting adjusted covariance matrix can then be indefinite and is not invariant under reparameterization. The FIRSTORDER or LINEAR suboption of the DDFM=KENWARDROGER option eliminates the second derivatives from the calculation of the covariance matrix adjustment. The LINEAR suboption is an alias for FIRSTORDER. For the case of scalar estimable functions, the resulting estimator is referred to as the Prasad-Rao estimator $\widetilde{m}^{@}$ in Harville and Jeske (1992). The following are examples of covariance structures that generally lead to nonzero second derivatives: TYPE=ANTE(1), TYPE=AR(1), TYPE=ARH(1), TYPE=ARMA(1,1), TYPE=CSH, TYPE=FA, TYPE=FA0(q), TYPE=TOEPH, TYPE=UNR, and all TYPE=SP() structures.

The DDFM=KENWARDROGER2 option specifies an improved approximation of the DDFM=KENWARDROGER method that uses a less biased precision estimator, as proposed by Kenward and Roger (2009). For an intrinsically linear covariance parameterization, this option produces the same precision estimator as that obtained using DDFM=KR(FIRSTORDER).

When the asymptotic variance matrix of the covariance parameters is found to be singular, a generalized inverse is used. Covariance parameters with zero variance then do not contribute to the degrees-of-freedom adjustment for DDFM=SATTERTHWAITE, DDFM=KENWARDROGER, or DDFM=KENWARDROGER2, and a message is written to the log.

This method changes output in the following tables (listed in Table 84.26): Contrast, CorrB, CovB, Diffs, Estimates, InvCovB, LSMeans, Slices, SolutionF, SolutionR, Tests1–Tests3. The OUTP= and OUTPM= data sets are also affected.

E

requests that Type 1, Type 2, and Type 3 **L** matrix coefficients be displayed for all specified effects. The ODS name of the table is Coef.

E1

requests that Type 1 **L** matrix coefficients be displayed for all specified effects. The ODS name of the table is Coef.

E2

requests that Type 2 **L** matrix coefficients be displayed for all specified effects. The ODS name of the table is Coef.

E3

requests that Type 3 **L** matrix coefficients be displayed for all specified effects. The ODS name of the table is Coef.

FULLX

requests that columns of the **X** matrix that consist entirely of zeros not be eliminated from **X**; otherwise, they are eliminated by default. For a column corresponding to a missing cell to be added to **X**, its particular levels must be present in at least one observation in the analysis data set along with a missing dependent variable. The use of the FULLX option can affect coefficient specifications in the CONTRAST and ESTIMATE statements, as well as covariate coefficients from LSMEANS statements specified with the AT MEANS option.

HTYPE=*value-list*

indicates the type of hypothesis test to perform on the fixed effects. Valid entries for *values* in the list are 1, 2, and 3; the default value is 3. You can specify several types by separating the values with a comma or a space. The ODS table names are Tests1 for the Type 1 tests, Tests2 for the Type 2 tests, and Tests3 for the Type 3 tests.

INFLUENCE< (*influence-options***) >**

specifies that influence and case deletion diagnostics are to be computed.

The INFLUENCE option computes influence diagnostics by noniterative or iterative methods. The noniterative diagnostics rely on recomputation formulas under the assumption that covariance parameters or their ratios remain fixed. With the possible exception of a profiled residual variance, no covariance parameters are updated. This is the default behavior because of its computational efficiency. However, the impact of an observation on the overall analysis can be underestimated if its effect on covariance parameters is not assessed. Toward this end, iterative methods can be applied to gauge the overall impact of observations and to obtain influence diagnostics for the covariance parameter estimates.

If you specify the INFLUENCE option without further *suboptions*, PROC MIXED computes single-case deletion diagnostics and influence statistics for each observation in the data set by updating estimates for the fixed-effects parameter estimates, and also the residual variance, if it is profiled. The EFFECT=, SELECT=, ITER=, SIZE=, and KEEP= suboptions provide additional flexibility in the computation and reporting of influence statistics. Table 84.10 briefly describes important *suboptions* and their effect on the influence analysis.

Table 84.10 Summary of INFLUENCE Option Default and Suboptions

Description	Suboption
Compute influence diagnostics for individual observations	Default
Measure influence of sets of observations chosen according to a classification variable or effect	EFFECT=
Remove pairs of observations and report the results sorted by degree of influence	SIZE=2
Remove triples, quadruples of observations, etc.	SIZE=
Allow selection of individual observations, observations sharing specific levels of effects, and construction of tuples from specified subsets of observations	SELECT=
Update fixed effects and covariance parameters by refitting the mixed model, adding up to n iterations	ITER=$n > 0$
Compute influence diagnostics for the covariance parameters	ITER=$n > 0$
Update only fixed effects and the residual variance, if it is profiled	ITER=0
Add the reduced-data estimates to the data set created with ODS OUTPUT	ESTIMATES

The modifiers and their default values are discussed in the following paragraphs. The set of computed influence diagnostics varies with the *suboptions*. The most extensive set of influence diagnostics is obtained when ITER=n with $n > 0$.

You can produce statistical graphics of influence diagnostics when ODS Graphics is enabled. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the MIXED procedure, see the section "ODS Graphics" on page 6877.

You can specify the following *influence-options* in parentheses:

EFFECT=*effect*

specifies an effect according to which observations are grouped. Observations sharing the same level of the *effect* are removed from the analysis as a group. The *effect* must contain only classification variables, but they need not be contained in the model.

Removing observations can change the rank of the $(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}$ matrix. This is particularly likely to happen when multiple observations are eliminated from the analysis. If the rank of the estimated variance-covariance matrix of $\widehat{\boldsymbol{\beta}}$ changes or its singularity pattern is altered, no influence diagnostics are computed.

ESTIMATES

EST

specifies that the updated parameter estimates should be written to the ODS output data set. The values are not displayed in the "Influence" table, but if you use ODS OUTPUT to create a data set from the listing, the estimates are added to the data set. If ITER=0, only the fixed-effects estimates are saved. In iterative influence analyses, fixed-effects and covariance parameters are

stored. The p fixed-effects parameter estimates are named Parm1–Parm$p$, and the q covariance parameter estimates are named CovP1–CovPq. The order corresponds to that in the "Solution for Fixed Effects" and "Covariance Parameter Estimates" tables. If parameter updates fail—for example, because of a loss of rank or a nonpositive definite Hessian—missing values are reported.

ITER=*n*

controls the maximum number of additional iterations PROC MIXED performs to update the fixed-effects and covariance parameter estimates following data point removal. If you specify $n > 0$, then statistics such as DFFITS, MDFFITS, and the likelihood distances measure the impact of observation(s) on all aspects of the analysis. Typically, the influence will grow compared to values at ITER=0. In models without RANDOM or REPEATED effects, the ITER= option has no effect.

This documentation refers to analyses when $n > 0$ simply as iterative influence analysis, even if final covariance parameter estimates can be updated in a single step (for example, when METHOD=MIVQUE0 or METHOD=TYPE3). This nomenclature reflects the fact that only if $n > 0$ are all model parameters updated, which can require additional iterations. If $n > 0$ and METHOD=REML (default) or METHOD=ML, the procedure updates fixed effects *and* variance-covariance parameters after removing the selected observations with additional Newton-Raphson iterations, starting from the converged estimates for the entire data. The process stops for each observation or set of observations if the convergence criterion is satisfied or the number of further iterations exceeds n. If $n > 0$ and METHOD=TYPE1, TYPE2, or TYPE3, ANOVA estimates of the covariance parameters are recomputed in a single step.

Compared to noniterative updates, the computations are more involved. In particular for large data sets or a large number of random effects (or both), iterative updates require considerably more resources. A one-step (ITER=1) or two-step update might be a good compromise. The output includes the number of iterations performed, which is less than n if the iteration converges. If the process does not converge in n iterations, you should be careful in interpreting the results, especially if n is fairly large.

Bounds and other restrictions on the covariance parameters carry over from the full-data model. Covariance parameters that are not iterated in the model fit to the full data (the NOITER or HOLD= option in the PARMS statement) are likewise not updated in the refit. In certain models, such as random-effects models, the ratios between the covariance parameters and the residual variance are maintained rather than the actual value of the covariance parameter estimate (see the section "Influence Diagnostics" on page 6862).

KEEP=*n*

determines how many observations are retained for display and in the output data set or how many tuples if you specify SIZE=. The output is sorted by an influence statistic as discussed for the SIZE= suboption.

SELECT=*value-list*

specifies which observations or effect levels are chosen for influence calculations. If the SELECT= suboption is not specified, diagnostics are computed as follows:

- for all observations, if EFFECT= or SIZE= are not given
- for all levels of the specified effect, if EFFECT= is specified
- for all tuples of size k formed from the observations in *value-list*, if SIZE=k is specified

When you specify an effect with the EFFECT= option, the values in *value-list* represent indices of the levels in the order in which PROC MIXED builds classification effects. Which observations in the data set correspond to this index depends on the order of the variables in the CLASS statement, not the order in which the variables appear in the interaction effect. See the section "Parameterization of Mixed Models" on page 6855 to understand precisely how the procedure indexes nested and crossed effects and how levels of classification variables are ordered. The actual values of the classification variables involved in the effect are shown in the output so you can determine which observations were removed.

If the EFFECT= suboption is not specified, the SELECT= value list refers to the sequence in which observations are read from the input data set or from the current BY group if there is a BY statement. This indexing is not necessarily the same as the observation numbers in the input data set, for example, if a WHERE clause is specified or during BY processing.

SIZE=*n*

instructs PROC MIXED to remove groups of observations formed as tuples of size n. For example, SIZE=2 specifies all $n \times (n-1)/2$ unique pairs of observations. The number of tuples for SIZE=k is $n!/(k!(n-k)!)$ and grows quickly with n and k. Using the SIZE= option can result in considerable computing time. The MIXED procedure displays by default only the 50 tuples with the greatest influence. Use the KEEP= option to override this default and to retain a different number of tuples in the listing or ODS output data set. Regardless of the KEEP= specification, all tuples are evaluated and the results are ordered according to an influence statistic. This statistic is the (restricted) likelihood distance as a measure of overall influence if ITER= $n > 0$ or when a residual variance is profiled. When likelihood distances are unavailable, the results are ordered by the PRESS statistic.

To reduce computational burden, the SIZE= option can be combined with the SELECT=*value-list* modifier. For example, the following statements evaluate all $15 = 6 \times 5/2$ pairs formed from observations 13, 14, 18, 30, 31, and 33 and display the five pairs with the greatest influence:

```
proc mixed;
   class a m f;
   model penetration = a m /
                  influence(size=2 keep=5
                            select=13,14,18,30,31,33);
   random f(m);
run;
```

If any observation in a tuple contains missing values or has otherwise not contributed to the analysis, the tuple is not evaluated. This guarantees that the displayed results refer to the same number of observations, so that meaningful statistics are available by which to order the results. If computations fail for a particular tuple—for example, because the $(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}$ matrix changes rank or the $\mathbf{G}$ matrix is not positive definite—no results are produced. Results are retained when the maximum number of iterative updates is exceeded in iterative influence analyses.

The SIZE= suboption cannot be combined with the EFFECT= suboption. As in the case of the EFFECT= suboption, the statistics being computed are those appropriate for removal of multiple data points, even if SIZE=1.

The ODS name of the "Influence Diagnostics" table is Influence. The variables in this table depend on whether you specify the EFFECT=, SIZE=, or KEEP= suboption and whether covariance parameters

are iteratively updated. When ITER=0 (the default), certain influence diagnostics are meaningful only if the residual variance is profiled. Table 84.11 and Table 84.12 summarize the statistics obtained depending on the model and modifiers. The last column in these tables gives the variable name in the ODS OUTPUT INFLUENCE= data set. Restricted likelihood distances are reported instead of the likelihood distance unless METHOD=ML. See the section "Influence Diagnostics" on page 6862 for details about the individual statistics.

Table 84.11 Statistics Computed with INFLUENCE Option, Noniterative Analysis (ITER=0)

Suboption	σ^2 **Profiled**	**Statistic**	**Variable Name**
Default	Yes	Observed value	Observed
		Predicted value	Predicted
		Marginal residual	Residual
		Leverage	Leverage
		PRESS residual	PRESSRes
		Internally studentized marginal residual	Student
		Externally studentized marginal residual	RStudent
		RMSE without deleted observations	RMSE
		Cook's *D*	CookD
		DFFITS	DFFITS
		CovRatio	COVRATIO
		(Restricted) likelihood distance	RLD, LD
Default	No	Observed value	Observed
		Predicted value	Predicted
		Marginal residual	Residual
		Leverage	Leverage
		PRESS residual	PRESSRes
		Internally studentized marginal residual	Student
		Cook's *D*	CookD
EFFECT=, **SIZE=**, or **KEEP=**	Yes	Observations in level (tuple)	Nobs
		PRESS statistic	PRESS
		Cook's *D*	CookD
		MDFFITS	MDFFITS
		CovRatio	COVRATIO
		COVTRACE	COVTRACE
		RMSE without deleted level (tuple)	RMSE
		(Restricted) likelihood distance	RLD, LD
EFFECT=, **SIZE=**, or **KEEP=**	No	Observations in level (tuple)	Nobs
		PRESS statistic	PRESS
		Cook's *D*	CookD

Table 84.12 Statistics Computed with INFLUENCE Option, Iterative Analysis (ITER=$n > 0$)

Suboption	Statistic	Variable Name
Default	Number of iterations	Iter
	Observed value	Observed
	Predicted value	Predicted
	Marginal residual	Residual
	Leverage	Leverage
	PRESS residual	PRESSres
	Internally studentized marginal residual	Student
	Externally studentized marginal residual	RStudent
	RMSE without deleted obs (if possible)	RMSE
	Cook's *D*	CookD
	DFFITS	DFFITS
	CovRatio	COVRATIO
	Cook's *D* CovParms	CookDCP
	CovRatio CovParms	COVRATIOCP
	MDFFITS CovParms	MDFFITSCP
	(Restricted) likelihood distance	RLD, LD
EFFECT=, **SIZE=**, or **KEEP=**	Observations in level (tuple)	Nobs
	Number of iterations	Iter
	PRESS statistic	PRESS
	RMSE without deleted level (tuple)	RMSE
	Cook's *D*	CookD
	MDFFITS	MDFFITS
	CovRatio	COVRATIO
	COVTRACE	COVTRACE
	Cook's *D* CovParms	CookDCP
	CovRatio CovParms	COVRATIOCP
	MDFFITS CovParms	MDFFITSCP
	(Restricted) likelihood distance	RLD, LD

INTERCEPT

adds a row to the tables for Type 1, 2, and 3 tests corresponding to the overall intercept.

LCOMPONENTS

requests an estimate for each row of the $\mathbf{L}$ matrix used to form tests of fixed effects. Components corresponding to Type 3 tests are the default; you can produce the Type 1 and Type 2 component estimates with the HTYPE= option.

Tests of fixed effects involve testing of linear hypotheses of the form $\mathbf{L}\boldsymbol{\beta} = \mathbf{0}$. The matrix $\mathbf{L}$ is constructed from Type 1, 2, or 3 estimable functions. By default the MIXED procedure constructs Type 3 tests. In many situations, the individual rows of the matrix $\mathbf{L}$ represent contrasts of interest. For example, in a one-way classification model, the Type 3 estimable functions define differences

of factor-level means. In a balanced two-way layout, the rows of $\mathbf{L}$ correspond to differences of cell means.

For example, suppose factors A and B have a and b levels, respectively. The following statements produce $(a - 1)$ one degree of freedom tests for the rows of $\mathbf{L}$ associated with the Type 1 and Type 3 estimable functions for factor A, $(b - 1)$ tests for the rows of $\mathbf{L}$ associated with factor B, and a single test for the Type 1 and Type 3 coefficients associated with regressor X:

```
class A B;
model y = A B x / htype=1,3 lcomponents;
```

The denominator degrees of freedom associated with a row of $\mathbf{L}$ are the same as those in the corresponding "Tests of Fixed Effects" table, except for DDFM=KENWARDROGER, DDFM=KENWARDROGER2, and DDFM=SATTERTHWAITE. For these degrees-of-freedom methods, the denominator degrees of freedom are computed separately for each row of $\mathbf{L}$.

The ODS name of the table containing all requested component tests is LComponents. See Example 84.9 for applications of the LCOMPONENTS option.

NOCONTAIN

has the same effect as the DDFM=RESIDUAL option.

NOINT

requests that no intercept be included in the model. An intercept is included by default.

NOTEST

specifies that no hypothesis tests be performed for the fixed effects.

OUTP=*SAS-data-set*

OUTPRED=*SAS-data-set*

specifies an output data set containing predicted values and related quantities. This option replaces the P option from SAS 6.

Predicted values are formed by using the rows from $(\mathbf{X}\ \mathbf{Z})$ as $\mathbf{L}$ matrices. Thus, predicted values from the original data are $\mathbf{X}\widehat{\boldsymbol{\beta}} + \mathbf{Z}\widehat{\boldsymbol{\gamma}}$. Their approximate standard errors of prediction are formed from the quadratic form of $\mathbf{L}$ with $\widehat{\mathbf{C}}$ defined in the section "Statistical Properties" on page 6851. The L95 and U95 variables provide a t-type confidence interval for the predicted values, and they correspond to the L95M and U95M variables from the GLM and REG procedures for fixed-effects models. The residuals are the observed minus the predicted values. Predicted values for data points other than those observed can be obtained by using missing dependent variables in your input data set.

Specifications that have a REPEATED statement with the SUBJECT= option and missing dependent variables compute predicted values by using empirical best linear unbiased prediction (EBLUP). Using hats (ˆ) to denote estimates, the EBLUP formula is

$$\hat{\mathbf{m}} = \mathbf{X}_{\mathbf{m}}\hat{\boldsymbol{\beta}} + \hat{\mathbf{C}}_{\mathbf{m}}\hat{\mathbf{V}}^{-1}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})$$

where $\mathbf{m}$ represents a hypothetical realization of a missing data vector with associated design matrix $\mathbf{X}_{\mathbf{m}}$. The matrix $\mathbf{C}_{\mathbf{m}}$ is the model-based covariance matrix between $\mathbf{m}$ and the observed data $\mathbf{y}$, and other notation is as presented in the section "Mixed Models Theory" on page 6843.

The estimated prediction variance is as follows:

$$\widehat{\mathrm{Var}}(\hat{\mathbf{m}} - \mathbf{m}) = \hat{\mathbf{V}}_{\mathbf{m}} - \hat{\mathbf{C}}_{\mathbf{m}}\hat{\mathbf{V}}^{-1}\hat{\mathbf{C}}'_{\mathbf{m}} + \\ [\mathbf{X}_{\mathbf{m}} - \hat{\mathbf{C}}_{\mathbf{m}}\hat{\mathbf{V}}^{-1}\mathbf{X}](\mathbf{X}'\hat{\mathbf{V}}^{-1}\mathbf{X})^{-}[\mathbf{X}_{\mathbf{m}} - \hat{\mathbf{C}}_{\mathbf{m}}\hat{\mathbf{V}}^{-1}\mathbf{X}]'$$

where $\mathbf{V}_{\mathbf{m}}$ is the model-based variance matrix of $\mathbf{m}$. For further details, see Henderson (1984) and Harville (1990). This feature can be useful for forecasting time series or for computing spatial predictions.

By default, all variables from the input data set are included in the OUTP= data set. You can select a subset of these variables by using the ID statement.

OUTPM=*SAS-data-set*

OUTPREDM=*SAS-data-set*

specifies an output data set containing predicted means and related quantities. This option replaces the PM option from SAS 6.

The output data set is of the same form as that resulting from the OUTP= option, except that the predicted values do not incorporate the EBLUP values $\mathbf{Z}\widehat{\boldsymbol{\gamma}}$. They also do not use the EBLUPs for specifications that have a REPEATED statement with the SUBJECT= option and missing dependent variables. The predicted values are formed as $\mathbf{X}\widehat{\boldsymbol{\beta}}$ in the OUTPM= data set, and standard errors are quadratic forms in the approximate variance-covariance matrix of $\widehat{\boldsymbol{\beta}}$ as displayed by the COVB option.

By default, all variables from the input data set are included in the OUTPM= data set. You can select a subset of these variables by using the ID statement.

RESIDUAL

RESIDUALS

requests that Pearson-type and (internally) studentized residuals be added to the OUTP= and OUTPM= data sets. Studentized residuals are raw residuals standardized by their estimated standard error. When residuals are internally studentized, the data point in question has contributed to the estimation of the covariance parameter estimates on which the standard error of the residual is based. Externally studentized marginal residuals can be computed with the INFLUENCE option. Pearson-type residuals scale the residual by the standard deviation of the response.

The option has no effect unless the OUTP= or OUTPM= option is specified or unless ODS Graphics is enabled. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the MIXED procedure, see the section "ODS Graphics" on page 6877. For computational details about studentized and Pearson residuals in MIXED, see the section "Residual Diagnostics" on page 6860.

SINGCHOL=*number*

tunes the sensitivity in computing Cholesky roots. If a diagonal pivot element is less than *D*number* as PROC MIXED performs the Cholesky decomposition on a matrix, the associated column is declared to be linearly dependent upon previous columns and is set to **0**. The value *D* is the original diagonal element of the matrix. The default for *number* is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGRES=*number*

sets the tolerance for which the residual variance is considered to be zero. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGULAR=*number*

tunes the sensitivity in sweeping. If a diagonal pivot element is less than *D*number* as PROC MIXED sweeps a matrix, the associated column is declared to be linearly dependent upon previous columns, and the associated parameter is set to 0. The value *D* is the original diagonal element of the matrix. The default is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SOLUTION

S

requests that a solution for the fixed-effects parameters be produced. Using notation from the section "Mixed Models Theory" on page 6843, the fixed-effects parameter estimates are $\widehat{\boldsymbol{\beta}}$ and their approximate standard errors are the square roots of the diagonal elements of $(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}$. You can output this approximate variance matrix with the COVB option or modify it with the EMPIRICAL option in the PROC MIXED statement or the DDFM=KENWARDROGER or DDFM=KENWARDROGER2 option in the MODEL statement.

Along with the estimates and their approximate standard errors, a *t* statistic is computed as the estimate divided by its standard error. The degrees of freedom for this *t* statistic matches the one appearing in the "Tests of Fixed Effects" table under the effect containing the parameter. The "Pr > |t|" column contains the two-tailed *p*-value corresponding to the *t* statistic and associated degrees of freedom. You can use the CL option to request confidence intervals for all of the parameters; they are constructed around the estimate by using a radius of the standard error times a percentage point from the *t* distribution.

VCIRY

requests that responses and marginal residuals be scaled by the inverse Cholesky root of the marginal variance-covariance matrix. The variables ScaledDep and ScaledResid are added to the OUTPM= data set. These quantities can be important in bootstrapping of data or residuals. Examination of the scaled residuals is also helpful in diagnosing departures from normality. Notice that the results of this scaling operation can depend on the order in which the MIXED procedure processes the data.

The VCIRY option has no effect unless you also use the OUTPM= option or unless ODS Graphics is enabled. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the MIXED procedure, see the section "ODS Graphics" on page 6877.

XPVIX

is an alias for the COVBI option.

XPVIXI

is an alias for the COVB option.

ZETA=*number*

tunes the sensitivity in forming Type 3 functions. Any element in the estimable function basis with an absolute value less than *number* is set to 0. The default is 1E–8.

PARMS Statement

PARMS (*value-list***)**. . . < / *options* > **;**

The PARMS statement specifies initial values for the covariance parameters, or it requests a grid search over several values of these parameters. You must specify the values in the order in which they appear in the "Covariance Parameter Estimates" table.

The *value-list* specification can take any of several forms:

m	a single value
$m_1, m_2, \ldots, m_n$	several values
m to n	a sequence where m equals the starting value, n equals the ending value, and the increment equals 1
m to n by i	a sequence where m equals the starting value, n equals the ending value, and the increment equals i
m_1, m_2 to m_3	mixed values and sequences

You can use the PARMS statement to input known parameters. Referring to the split-plot example (Example 84.1), suppose the three variance components are known to be 60, 20, and 6. The SAS statements to fix the variance components at these values are as follows:

```
proc mixed data=sp noprofile;
   class Block A B;
   model Y = A B A*B;
   random Block A*Block;
   parms (60) (20) (6) / noiter;
run;
```

The NOPROFILE option requests PROC MIXED to refrain from profiling the residual variance parameter during its calculations, thereby enabling its value to be held at 6 as specified in the PARMS statement. The NOITER option prevents any Newton-Raphson iterations so that the subsequent results are based on the given variance components. You can also specify known parameters of **G** by using the GDATA= option in the RANDOM statement.

If you specify more than one set of initial values, PROC MIXED performs a grid search of the likelihood surface and uses the best point on the grid for subsequent analysis. Specifying a large number of grid points can result in long computing times. The grid search feature is also useful for exploring the likelihood surface. (See Example 84.3.)

The results from the PARMS statement are the values of the parameters on the specified grid (denoted by CovP1–CovP*n*), the residual variance (possibly estimated) for models with a residual variance parameter, and various functions of the likelihood.

The ODS name of the "Parameter Search" table is ParmSearch.

Table 84.13 summarizes the *options* available in the PARMS statement.

Table 84.13 PARMS Statement Options

Option	Description
HOLD=	Holds parameter values equal to the specified values
LOGDETH	Evaluates the log determinant of the Hessian matrix
LOWERB=	Specifies lower boundary constraints
NOBOUND	Removes boundary constraints on covariance parameters
NOITER	Performs inferences using the best value from the grid search
NOPRINT	Suppresses the "Parameter Search" table
NOPROFILE	Specifies a different computational method for the residual variance
OLS	Requests starting values corresponding to the usual general linear model
PARMSDATA=	Reads in covariance parameter values from a SAS data set
RATIOS	Indicates that ratios with the residual variance are specified
UPPERB=	Specifies upper boundary constraints

You can specify the following *options* in the PARMS statement after a slash (/).

HOLD=*value-list*
EQCONS=*value-list*

specifies which parameter values PROC MIXED should hold to equal the specified values. For example, the following statement constrains the first and third covariance parameters to equal 5 and 2, respectively:

```
parms (5) (3) (2) (3) / hold=1,3;
```

LOGDETH

evaluates the log determinant of the Hessian matrix for each point specified in the PARMS statement. A Log Det H column is added to the "Parameter Search" table.

LOWERB=*value-list*

enables you to specify lower boundary constraints on the covariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that PROC MIXED uses for the covariance parameters, and each number corresponds to the lower boundary constraint. A missing value instructs PROC MIXED to use its default constraint, and if you do not specify numbers for all of the covariance parameters, PROC MIXED assumes the remaining ones are missing.

An example for which this option is useful is when you want to constrain the **G** matrix to be positive definite in order to avoid the more computationally intensive algorithms required when **G** becomes singular. The corresponding statements for a random coefficients model are as follows:

```
proc mixed;
   class person;
   model y = time;
   random int time / type=fa0(2) sub=person;
   parms / lowerb=1e-4,.,1e-4;
run;
```

Here the TYPE=FA0(2) structure is used in order to specify a Cholesky root parameterization for the 2×2 unstructured blocks in $\mathbf{G}$. This parameterization ensures that the $\mathbf{G}$ matrix is nonnegative definite, and the PARMS statement then ensures that it is positive definite by constraining the two diagonal terms to be greater than or equal to 1E–4.

NOBOUND

requests the removal of boundary constraints on covariance parameters. For example, variance components have a default lower boundary constraint of 0, and the NOBOUND option allows their estimates to be negative.

NOITER

requests that no Newton-Raphson iterations be performed and that PROC MIXED use the best value from the grid search to perform inferences. By default, iterations begin at the best value from the PARMS grid search.

NOPRINT

suppresses the display of the "Parameter Search" table.

NOPROFILE

specifies a different computational method for the residual variance during the grid search. By default, PROC MIXED estimates this parameter by using the profile likelihood when appropriate. This estimate is displayed in the Variance column of the "Parameter Search" table. The NOPROFILE option suppresses the profiling and uses the actual value of the specified variance in the likelihood calculations.

OLS

requests starting values corresponding to the usual general linear model. Specifically, all variances and covariances are set to zero except for the residual variance, which is set equal to its ordinary least squares (OLS) estimate. This option is useful when the default MIVQUE0 procedure produces poor starting values for the optimization process.

PARMSDATA=*SAS-data-set*

PDATA=*SAS-data-set*

reads in covariance parameter values from a SAS data set. The data set should contain the Est or Covp1–Covp*n* variables.

RATIOS

indicates that ratios with the residual variance are specified instead of the covariance parameters themselves. The default is to use the individual covariance parameters.

UPPERB=*value-list*

enables you to specify upper boundary constraints on the covariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that PROC MIXED uses for the covariance parameters, and each number corresponds to the upper boundary constraint. A missing value instructs PROC MIXED to use its default constraint, and if you do not specify numbers for all of the covariance parameters, PROC MIXED assumes that the remaining ones are missing.

PRIOR Statement

PRIOR < *distribution* > < / *options* > **;**

The PRIOR statement enables you to carry out a sampling-based Bayesian analysis in PROC MIXED. It currently operates only with variance component models. Other TYPE= structures are not supported. The analysis produces a SAS data set containing a pseudo-random sample from the joint posterior density of the variance components and other parameters in the mixed model.

The posterior analysis is performed after all other PROC MIXED computations. It begins with the "Posterior Sampling Information" table, which provides basic information about the posterior sampling analysis, including the prior densities, sampling algorithm, sample size, and random number seed. The ODS name of this table is Posterior.

By default, PROC MIXED uses an independence chain algorithm in order to generate the posterior sample (Tierney 1994). This algorithm works by generating a pseudo-random proposal from a convenient base distribution, chosen to be as close as possible to the posterior. The proposal is then retained in the sample with probability proportional to the ratio of weights constructed by taking the ratio of the true posterior to the base density. If a proposal is not accepted, then a duplicate of the previous observation is added to the chain.

In selecting the base distribution, PROC MIXED makes use of the fact that the fixed-effects parameters can be analytically integrated out of the joint posterior, leaving the marginal posterior density of the variance components. In order to better approximate the marginal posterior density of the variance components, PROC MIXED transforms them by using the MIVQUE(0) equations. You can display the selected transformation with the PTRANS option or specify your own with the TDATA= option. The density of the transformed parameters is then approximated by a product of inverted gamma densities (see Gelfand et al. 1990).

To determine the parameters for the inverted gamma densities, PROC MIXED evaluates the logarithm of the posterior density over a grid of points in each of the transformed parameters, and you can display the results of this search with the PSEARCH option. PROC MIXED then performs a linear regression of these values on the logarithm of the inverted gamma density. The resulting base densities are displayed in the "Base Densities" table; the ODS name of this table is Base. You can input different base densities with the BDATA= option.

At the end of the sampling, the "Acceptance Rates" table displays the acceptance rate computed as the number of accepted samples divided by the total number of samples generated. The ODS name of the "Acceptance Rates" table is AccRates.

The OUT= option specifies the output data set containing the posterior sample. PROC MIXED automatically includes all variance component parameters in this data set (labeled COVP1–COVP*n*), the Type 3 F statistics constructed as in Ghosh (1992) discussing Schervish (1992) (labeled T3F*n*), the log values of the posterior (labeled LOGF), the log of the base sampling density (labeled LOGG), and the log of their ratio (labeled LOGRATIO). If you specify the SOLUTION option in the MODEL statement, the data set also contains a random sample from the posterior density of the fixed-effects parameters (labeled BETA*n*); and if you specify the SOLUTION option in the RANDOM statement, the table contains a random sample from the posterior density of the random-effects parameters (labeled GAM*n*). PROC MIXED also generates additional variables corresponding to any CONTRAST, ESTIMATE, or LSMEANS statement that you specify.

Subsequently, you can use SAS/INSIGHT or the UNIVARIATE, CAPABILITY, or KDE procedure to analyze the posterior sample.

The prior density of the variance components is, by default, a noninformative version of Jeffreys' prior (Box and Tiao 1973). You can also specify informative priors with the DATA= option or a flat (equal to 1) prior for the variance components. The prior density of the fixed-effects parameters is assumed to be flat (equal to 1), and the resulting posterior is conditionally multivariate normal (conditioning on the variance component parameters) with mean $(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{y}$ and variance $(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}$.

Table 84.14 summarizes the *options* available in the PRIOR statement.

Table 84.14 PRIOR Statement Options

Option	Description
DATA=	Inputs the prior densities of the variance components
JEFFREYS	Specifies a noninformative reference version of Jeffreys' prior
FLAT	Specifies a prior density equal to 1 everywhere
ALG=	Specifies the algorithm used for generating the posterior sample
BDATA=	Inputs the base densities used by the sampling algorithm
GRID=	Specifies a grid of values over which to evaluate the posterior density
GRIDT=	Specifies a transformed grid of values over which to evaluate the posterior density
IFACTOR=	An alias for the SFACTOR= option
LOGNOTE=	Writes a note to the log after generating the sample
LOGRBOUND=	Specifies the bounding constant for rejection sampling
NSAMPLE=	Specifies the number of posterior samples to generate
NSEARCH=	Specifies the number of posterior evaluations
OUT=	Creates an output data set containing the sample from the posterior density
OUTG=	Creates an output data set from the grid evaluations
OUTGT=	Creates an output data set from the transformed grid evaluations
PSEARCH	Displays the search used to determine the parameters for the inverted gamma densities
PTRANS	Displays the transformation of the variance components
SEED=	Specifies an integer used to start the pseudo-random number generator
SFACTOR=	Adjusts the search range of the transformed parameters
TDATA=	Inputs the transformation used by the sampling algorithm
TRANS=	Specifies the algorithm that determines the transformation of the covariance parameters
UPDATE=	An alias for the LOGNOTE= option

The *distribution* argument in the PRIOR statement determines the prior density for the variance component parameters of your mixed model. Valid values are as follows.

DATA=

enables you to input the prior densities of the variance components used by the sampling algorithm. This data set must contain the Type and Parm1–Parm*n* variables, where *n* is the largest number of parameters among each of the base densities. The format of the DATA= data set matches that created by PROC MIXED in the "Base Densities" table, so you can output the densities from one run and use them as input for a subsequent run.

JEFFREYS

specifies a noninformative reference version of Jeffreys' prior constructed by using the square root of the determinant of the expected information matrix as in (1.3.92) of Box and Tiao (1973). This is the default prior.

FLAT

specifies a prior density equal to 1 everywhere, making the likelihood function the posterior.

You can specify the following *options* in the PRIOR statement after a slash (/).

ALG=IC | INDCHAIN

ALG=IS | IMPSAMP

ALG=RS | REJSAMP

ALG=RWC | RWCHAIN

specifies the algorithm to use for generating the posterior sample. The ALG=IC option requests an independence chain algorithm. The option ALG=IS requests importance sampling, ALG=RS requests rejection sampling, and ALG=RWC requests a random walk chain. The IC algorithm works only for models that have random effects. If you specify ALG=IC and the model has no random effects, then it silently switches to ALG=IS. By default, when the model has the random-effects, ALG=IC; otherwise, ALG=IS. When ALG=IS, the "Acceptance Rates" table is not generated. For more information about these techniques, see Ripley (1987); Smith and Gelfand (1992); Tierney (1994).

BDATA=

enables you to input the base densities used by the sampling algorithm. This data set must contain the Type and Parm1–Parm*n* variables, where *n* is the largest number of parameters among each of the base densities. The format of the BDATA= data set matches that created by PROC MIXED in the "Base Densities" table, so you can output the densities from one run and use them as input for a subsequent run.

GRID=(*value-list*)

specifies a grid of values over which to evaluate the posterior density. The *value-list* syntax is the same as in the PARMS statement, and you must specify an output data set name with the OUTG= option.

GRIDT=(*value-list*)

specifies a transformed grid of values over which to evaluate the posterior density. The *value-list* syntax is the same as in the PARMS statement, and you must specify an output data set name with the OUTGT= option.

IFACTOR=*number*

is an alias for the SFACTOR= option.

LOGNOTE=*number*

instructs PROC MIXED to write a note to the SAS log after it generates the sample corresponding to each multiple of *number*. This is useful for monitoring the progress of CPU-intensive runs.

LOGRBOUND=*number*

specifies the bounding constant for rejection sampling. The value of *number* equals the maximum of $\log\{f/g\}$ over the variance component parameter space, where f is the posterior density and g is the product inverted gamma densities used to perform rejection sampling.

When performing the rejection sampling, you might encounter the following message:

```
WARNING: The log ratio bound of LL was violated at sample XX.
```

When this occurs, PROC MIXED reruns an optimization algorithm to determine a new log upper bound and then restarts the rejection sampling. The resulting OUT= data set contains all observations that have been generated; therefore, assuming that you have requested N samples, you should retain only the final N observations in this data set for analysis purposes.

NSAMPLE=*number*

specifies the number of posterior samples to generate. The default is 1000, but more accurate results are obtained with larger samples such as 10000.

NSEARCH=*number*

specifies the number of posterior evaluations PROC MIXED makes for each transformed parameter in determining the parameters for the inverted gamma densities. The default is 20.

OUT=*SAS-data-set*

creates an output data set containing the sample from the posterior density.

OUTG=*SAS-data-set*

creates an output data set from the grid evaluations specified in the GRID= option.

OUTGT=*SAS-data-set*

creates an output data set from the transformed grid evaluations specified in the GRIDT= option.

PSEARCH

displays the search used to determine the parameters for the inverted gamma densities. The ODS name of the table is Search.

PTRANS

displays the transformation of the variance components. The ODS name of the table is Trans.

SEED=*number*

specifies an integer used to start the pseudo-random number generator for the simulation. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is by default generated from reading the time of day from the computer clock. You should use a positive seed (less than $2^{31} - 1$) whenever you want to duplicate the sample in another run of PROC MIXED.

SFACTOR=*number*

enables you to adjust the range over which PROC MIXED searches the transformed parameters in order to determine the parameters for the inverted gamma densities. PROC MIXED determines the range by first transforming the estimates from the standard PROC MIXED analysis (REML, ML, or MIVQUE0, depending upon which estimation method you select). It then multiplies and divides the transformed estimates by $2*$*number* to obtain upper and lower bounds, respectively. Transformed values that produce negative variance components in the original scale are not included in the search. The default value is 1; *number* must be greater than 0.5.

TDATA=*SAS-data-set*
enables you to input the transformation of the covariance parameters used by the sampling algorithm. This data set should contain the CovP1–CovP*n* variables. The format of the TDATA= data set matches that created by PROC MIXED in the Trans table, so you can output the transformation from one run and use it as input for a subsequent run.

TRANS=EXPECTED | MIVQUE0 | OBSERVED
specifies the particular algorithm used to determine the transformation of the covariance parameters. The default is MIVQUE0, indicating a transformation based on the MIVQUE(0) equations. The other two options indicate the type of Hessian matrix used in constructing the transformation via a Cholesky root.

UPDATE=*number*
is an alias for the LOGNOTE= option.

RANDOM Statement

RANDOM *random-effects* < / *options* > **;**

The RANDOM statement defines the random effects constituting the $\boldsymbol{\gamma}$ vector in the mixed model. It can be used to specify traditional variance component models (as in the VARCOMP procedure) and to specify random coefficients. The random effects can be classification or continuous, and multiple RANDOM statements are possible.

Using notation from the section "Mixed Models Theory" on page 6843, the purpose of the RANDOM statement is to define the $\mathbf{Z}$ matrix of the mixed model, the random effects in the $\boldsymbol{\gamma}$ vector, and the structure of $\mathbf{G}$. The $\mathbf{Z}$ matrix is constructed exactly as the $\mathbf{X}$ matrix for the fixed effects, and the $\mathbf{G}$ matrix is constructed to correspond with the effects constituting $\mathbf{Z}$. The structure of $\mathbf{G}$ is defined by using the TYPE= option.

You can specify INTERCEPT (or INT) as a random effect to indicate the intercept. PROC MIXED does not include the intercept in the RANDOM statement by default as it does in the MODEL statement.

Table 84.15 summarizes the *options* available in the RANDOM statement. All *options* are subsequently discussed in alphabetical order.

Table 84.15 Summary of RANDOM Statement Options

Option	Description
Construction of Covariance Structure	
GDATA=	Requests that the $\mathbf{G}$ matrix be read from a SAS data set
GROUP=	Varies covariance parameters by groups
LDATA=	Specifies data set with coefficient matrices for TYPE=LIN
NOFULLZ	Eliminates columns in $\mathbf{Z}$ corresponding to missing values
RATIOS	Indicates that ratios are specified in the GDATA= data set
SUBJECT=	Identifies the subjects in the model
TYPE=	Specifies the covariance structure
Statistical Output	
ALPHA=α	Determines the confidence level $(1-\alpha)$

Table 84.15 *continued*

Option	Description
CL	Requests confidence limits for predictors of random effects
G	Displays the estimated **G** matrix
GC	Displays the Cholesky root (lower) of estimated **G** matrix
GCI	Displays the inverse Cholesky root (lower) of estimated **G** matrix
GCORR	Displays the correlation matrix corresponding to estimated **G** matrix
GI	Displays the inverse of the estimated **G** matrix
SOLUTION	Displays solutions $\widehat{\boldsymbol{\gamma}}$ of the G-side random effects
V	Displays blocks of the estimated **V** matrix
VC	Displays the lower-triangular Cholesky root of blocks of the estimated **V** matrix
VCI	Displays the inverse Cholesky root of blocks of the estimated **V** matrix
VCORR	Displays the correlation matrix corresponding to blocks of the estimated **V** matrix
VI	Displays the inverse of the blocks of the estimated **V** matrix

You can specify the following *options* in the RANDOM statement after a slash (/).

ALPHA=*number*
requests that a *t*-type confidence interval be constructed for each of the random-effect estimates with confidence level 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

CL
requests that *t*-type confidence limits be constructed for each of the random-effect estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option.

G
requests that the estimated **G** matrix be displayed. PROC MIXED displays blanks for values that are 0. If you specify the SUBJECT= option, then the block of the **G** matrix corresponding to the first subject is displayed. The ODS name of the table is G.

GC
displays the lower-triangular Cholesky root of the estimated **G** matrix according to the rules listed under the G option. The ODS name of the table is CholG.

GCI
displays the inverse Cholesky root of the estimated **G** matrix according to the rules listed under the G option. The ODS name of the table is InvCholG.

GCORR
displays the correlation matrix corresponding to the estimated **G** matrix according to the rules listed under the G option. The ODS name of the table is GCorr.

GDATA=*SAS-data-set*

requests that the **G** matrix be read in from a SAS data set. This **G** matrix is assumed to be known; therefore, only **R**-side parameters from effects in the REPEATED statement are included in the Newton-Raphson iterations. If no REPEATED statement is specified, then only a residual variance is estimated.

The information in the GDATA= data set can appear in one of two ways. The first is a sparse representation for which you include Row, Col, and Value variables to indicate the row, column, and value of **G**, respectively. All unspecified locations are assumed to be 0. The second representation is for dense matrices. In it you include Row and Col1–Col*n* variables to indicate, respectively, the row and columns of **G**, which is a symmetric matrix of order n. For both representations, you must specify effects in the RANDOM statement that generate a **Z** matrix that contains n columns. (See Example 84.4.)

If you have more than one RANDOM statement, only one GDATA= option is required in any one of them, and the data set you specify must contain the entire **G** matrix defined by all of the RANDOM statements.

If the GDATA= data set contains variance ratios instead of the variances themselves, then use the RATIOS option.

Known parameters of **G** can also be input by using the PARMS statement with the HOLD= option.

GI

displays the inverse of the estimated **G** matrix according to the rules listed under the G option. The ODS name of the table is InvG.

GROUP=*effect*

GRP=*effect*

defines an effect specifying heterogeneity in the covariance structure of **G**. All observations having the same level of the group effect have the same covariance parameters. Each new level of the group effect produces a new set of covariance parameters with the same structure as the original group. You should exercise caution in defining the group effect, because strange covariance patterns can result from its misuse. Also, the group effect can greatly increase the number of estimated covariance parameters, which can adversely affect the optimization process.

Continuous variables are permitted as arguments to the GROUP= option. PROC MIXED does not sort by the values of the continuous variable; rather, it considers the data to be from a new subject or group whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of subjects or groups and also prevents the production of a large "Class Level Information" table.

LDATA=*SAS-data-set*

reads the coefficient matrices associated with the TYPE=LIN(*number*) option. The data set must contain the variables Parm, Row, Col1–Col*n* or Parm, Row, Col, Value. The Parm variable denotes which of the *number* coefficient matrices is currently being constructed, and the Row, Col1–Col*n*, or Row, Col, Value variables specify the matrix values, as they do with the GDATA= option. Unspecified values of these matrices are set equal to 0.

NOFULLZ

eliminates the columns in $\mathbf{Z}$ corresponding to missing levels of random effects involving CLASS variables. By default, these columns are included in $\mathbf{Z}$.

RATIOS

indicates that ratios with the residual variance are specified in the GDATA= data set instead of the covariance parameters themselves. The default GDATA= data set contains the individual covariance parameters.

SOLUTION

S

requests that the solution for the random-effects parameters be produced. Using notation from the section "Mixed Models Theory" on page 6843, these estimates are the empirical best linear unbiased predictors (EBLUPs) $\widehat{\boldsymbol{\gamma}} = \widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{V}}^{-1}(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})$. They can be useful for comparing the random effects from different experimental units and can also be treated as residuals in performing diagnostics for your mixed model.

The numbers displayed in the SE Pred column of the "Solution for Random Effects" table are not the standard errors of the $\widehat{\boldsymbol{\gamma}}$ displayed in the Estimate column; rather, they are the standard errors of predictions $\widehat{\boldsymbol{\gamma}}_i - \boldsymbol{\gamma}_i$, where $\widehat{\boldsymbol{\gamma}}_i$ is the ith EBLUP and $\boldsymbol{\gamma}_i$ is the ith random-effect parameter.

SUBJECT=*effect*

SUB=*effect*

identifies the subjects in your mixed model. Complete independence is assumed across subjects; thus, for the RANDOM statement, the SUBJECT= option produces a block-diagonal structure in $\mathbf{G}$ with identical blocks. The $\mathbf{Z}$ matrix is modified to accommodate this block diagonality. In fact, specifying a subject effect is equivalent to nesting all other effects in the RANDOM statement within the subject effect.

Continuous variables are permitted as arguments to the SUBJECT= option. PROC MIXED does not sort by the values of the continuous variable; rather, it considers the data to be from a new subject or group whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of subjects or groups and also prevents the production of a large "Class Level Information" table.

When you specify the SUBJECT= option and a classification random effect, computations are usually much quicker if the levels of the random effect are duplicated within each level of the SUBJECT= effect.

TYPE=*covariance-structure*

specifies the covariance structure of $\mathbf{G}$. Valid values for *covariance-structure* and their descriptions are listed in Table 84.17 and Table 84.18. Although a variety of structures are available, most applications call for either TYPE=VC or TYPE=UN. The TYPE=VC (variance components) option is the default structure, and it models a different variance component for each random effect.

The TYPE=UN (unstructured) option is useful for correlated random coefficient models. For example, the following statement specifies a random intercept-slope model that has different variances for the intercept and slope and a covariance between them:

```
random intercept age / type=un subject=person;
```

You can also use TYPE=FA0(2) here to request a **G** estimate that is constrained to be nonnegative definite.

If you are constructing your own columns of **Z** with continuous variables, you can use the TYPE=TOEP(1) structure to group them together to have a common variance component. If you want to have different covariance structures in different parts of **G**, you must use multiple RANDOM statements with different TYPE= options.

V<=*value-list*>
requests that blocks of the estimated **V** matrix be displayed. The first block determined by the SUBJECT= effect is the default displayed block. PROC MIXED displays entries that are 0 as blanks in the table.

You can optionally use the *value-list* specification, which indicates the subjects for which blocks of **V** are to be displayed. For example, the following statement displays block matrices for the first, third, and seventh persons:

```
random int time / type=un subject=person v=1,3,7;
```

The ODS table name is V.

VC<=*value-list*>
displays the Cholesky root of the blocks of the estimated **V** matrix. The *value-list* specification is the same as in the V option. The ODS table name is CholV.

VCI<=*value-list*>
displays the inverse of the Cholesky root of the blocks of the estimated **V** matrix. The *value-list* specification is the same as in the V option. The ODS table name is InvCholV.

VCORR<=*value-list*>
displays the correlation matrix corresponding to the blocks of the estimated **V** matrix. The *value-list* specification is the same as in the V option. The ODS table name is VCorr.

VI<=*value-list*>
displays the inverse of the blocks of the estimated **V** matrix. The *value-list* specification is the same as in the V option. The ODS table name is InvV.

REPEATED Statement

REPEATED < *repeated-effect* > < / *options* > **;**

The REPEATED statement is used to specify the **R** matrix in the mixed model. Its syntax is different from that of the REPEATED statement in PROC GLM. If no REPEATED statement is specified, **R** is assumed to be equal to $\sigma^2\mathbf{I}$.

For many repeated measures models, no repeated effect is required in the REPEATED statement. Simply use the SUBJECT= option to define the blocks of **R** and the TYPE= option to define their covariance structure. In this case, the repeated measures data must be similarly ordered for each subject, and you must indicate all missing response variables with periods in the input data set unless they all fall at the end of a subject's repeated response profile. These requirements are necessary in order to inform PROC MIXED of the proper location of the observed repeated responses.

Specifying a repeated effect is useful when you do not want to indicate missing values with periods in the input data set. The repeated effect must contain only classification variables. Make sure that the levels of the repeated effect are different for each observation within a subject; otherwise, PROC MIXED constructs identical rows in **R** corresponding to the observations with the same level. This results in a singular **R** and an infinite likelihood.

Whether you specify a REPEATED effect or not, the rows of **R** for each subject are constructed in the order in which they appear in the input data set.

Table 84.16 summarizes the *options* available in the REPEATED statement. All *options* are subsequently discussed in alphabetical order.

Table 84.16 Summary of REPEATED Statement Options

Option	Description
Construction of Covariance Structure	
GROUP=	Defines an effect specifying heterogeneity in the R-side covariance structure
LDATA=	Specifies data set with coefficient matrices for TYPE=LIN
LOCAL	Requests that a diagonal matrix be added to **R**
LOCALW	Specifies that only the local effects are weighted
NONLOCALW	Specifies that only the nonlocal effects are weighted
SUBJECT=	Identifies the subjects in the R-side model
TYPE=	Specifies the R-side covariance structure
Statistical Output	
HLM	Produces a table of Hotelling-Lawley-McKeon statistics (McKeon 1974)
HLPS	Produces a table of Hotelling-Lawley-Pillai-Samson statistics (Pillai and Samson 1959)
R	Displays blocks of the estimated **R** matrix
RC	Display the Cholesky root (lower) of blocks of the estimated **R** matrix
RCI	Displays the inverse Cholesky root (lower) of blocks of the estimated **R** matrix

Table 84.16 *continued*

Option	Description
RCORR	Displays the correlation matrix corresponding to blocks of the estimated **R** matrix
RI	Displays the inverse of blocks of the estimated **R** matrix

You can specify the following *options* in the REPEATED statement after a slash (/).

GROUP=*effect*

GRP=*effect*

defines an effect that specifies heterogeneity in the covariance structure of **R**. All observations that have the same level of the GROUP effect have the same covariance parameters. Each new level of the GROUP effect produces a new set of covariance parameters with the same structure as the original group. You should exercise caution in properly defining the GROUP effect, because strange covariance patterns can result with its misuse. Also, the GROUP effect can greatly increase the number of estimated covariance parameters, which can adversely affect the optimization process.

Continuous variables are permitted as arguments to the GROUP= option. PROC MIXED does not sort by the values of the continuous variable; rather, it considers the data to be from a new subject or group whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of subjects or groups and also prevents the production of a large "Class Level Information" table.

HLM

produces a table of Hotelling-Lawley-McKeon statistics (McKeon 1974) for all fixed effects whose levels change across data having the same level of the SUBJECT= effect (the *within-subject* fixed effects). This option applies only when you specify a REPEATED statement with the TYPE=UN option and no RANDOM statements. For balanced data, this model is equivalent to the multivariate model for repeated measures in PROC GLM.

The Hotelling-Lawley-McKeon statistic has a slightly better F approximation than the Hotelling-Lawley-Pillai-Samson statistic (see the description of the HLPS option, which follows). Both of the Hotelling-Lawley statistics can perform much better in small samples than the default F statistic (Wright 1994).

Separate tables are produced for Type 1, 2, and 3 tests, according to the ones you select. The ODS table names are HLM1, HLM2, and HLM3, respectively.

HLPS

produces a table of Hotelling-Lawley-Pillai-Samson statistics (Pillai and Samson 1959) for all fixed effects whose levels change across data having the same level of the SUBJECT= effect (the *within-subject* fixed effects). This option applies only when you specify a REPEATED statement with the TYPE=UN option and no RANDOM statements. For balanced data, this model is equivalent to the multivariate model for repeated measures in PROC GLM, and this statistic is the same as the Hotelling-Lawley Trace statistic produced by PROC GLM.

Separate tables are produced for Type 1, 2, and 3 tests, according to the ones you select. The ODS table names are HLPS1, HLPS2, and HLPS3, respectively.

LDATA=*SAS-data-set*

reads the coefficient matrices associated with the TYPE=LIN(*number*) option. The data set must contain the variables Parm, Row, Col1–Col*n* or Parm, Row, Col, Value. The Parm variable denotes which of the *number* coefficient matrices is currently being constructed, and the Row, Col1–Col*n*, or Row, Col, Value variables specify the matrix values, as they do with the RANDOM statement option GDATA=. Unspecified values of these matrices are set equal to 0.

LOCAL

LOCAL=EXP(< *effects* >**)**

LOCAL=POM(*POM-data-set***)**

requests that a diagonal matrix be added to $\mathbf{R}$. With just the LOCAL option, this diagonal matrix equals $\sigma^2\mathbf{I}$, and σ^2 becomes an additional variance parameter that PROC MIXED profiles out of the likelihood provided that you do not specify the NOPROFILE option in the PROC MIXED statement. The LOCAL option is useful if you want to add an observational error to a time series structure (Jones and Boadi-Boateng 1991) or a nugget effect to a spatial structure Cressie (1993).

The LOCAL=EXP(*<effects>*) option produces exponential local effects, also known as dispersion effects, in a log-linear variance model. These local effects have the form

$$\sigma^2 \mathrm{diag}[\exp(\mathbf{U}\boldsymbol{\delta})]$$

where $\mathbf{U}$ is the full-rank design matrix corresponding to the effects that you specify and $\boldsymbol{\delta}$ are the parameters that PROC MIXED estimates. An intercept is not included in $\mathbf{U}$ because it is accounted for by σ^2. PROC MIXED constructs the full-rank $\mathbf{U}$ in terms of 1s and –1s for classification effects. Be sure to scale continuous effects in $\mathbf{U}$ sensibly.

The LOCAL=POM(*POM-data-set*) option specifies the power-of-the-mean structure. This structure possesses a variance of the form $\sigma^2|\mathbf{x}_i'\boldsymbol{\beta}^*|^\theta$ for the ith observation, where $\mathbf{x}_i$ is the ith row of $\mathbf{X}$ (the design matrix of the fixed effects) and $\boldsymbol{\beta}^*$ is an estimate of the fixed-effects parameters that you specify in *POM-data-set*.

The SAS data set specified by *POM-data-set* contains the numeric variable Estimate (in previous releases, the variable name was required to be EST), and it has at least as many observations as there are fixed-effects parameters. The first p observations of the Estimate variable in *POM-data-set* are taken to be the elements of $\boldsymbol{\beta}^*$, where p is the number of columns of $\mathbf{X}$. You must order these observations according to the non-full-rank parameterization of the MIXED procedure. One easy way to set up *POM-data-set* for a $\boldsymbol{\beta}^*$ corresponding to ordinary least squares is illustrated by the following statements:

```
ods output SolutionF=sf;
proc mixed;
   class a;
   model y = a x / s;
run;

proc mixed;
   class a;
   model y = a x;
   repeated / local=pom(sf);
run;
```

Note that the generalized least squares estimate of the fixed-effects parameters from the second PROC MIXED step usually is not the same as your specified $\boldsymbol{\beta}^*$. However, you can iterate the POM fitting until the two estimates agree. Continuing from the previous example, the statements for performing one step of this iteration are as follows:

```
ods output SolutionF=sf1;
proc mixed;
   class a;
   model y = a x / s;
   repeated / local=pom(sf);
run;

proc compare brief data=sf compare=sf1;
   var estimate;
run;

data sf;
   set sf1;
run;
```

Unfortunately, this iterative process does not always converge. For further details, see the description of pseudo-likelihood in Chapter 3 of Carroll and Ruppert (1988).

LOCALW

specifies that only the local effects and no others be weighted. By default, all effects are weighted. The LOCALW option is used in connection with the WEIGHT statement and the LOCAL option in the REPEATED statement.

NONLOCALW

specifies that only the nonlocal effects and no others be weighted. By default, all effects are weighted. The NONLOCALW option is used in connection with the WEIGHT statement and the LOCAL option in the REPEATED statement.

R< **=***value-list*>

requests that blocks of the estimated **R** matrix be displayed. The first block determined by the SUBJECT= effect is the default displayed block. PROC MIXED displays blanks for value-lists that are 0.

The *value-list* indicates the subjects for which blocks of **R** are to be displayed. For example, the following statement displays block matrices for the first, third, and fifth persons:

```
repeated / type=cs subject=person r=1,3,5;
```

See the PARMS statement for the possible forms of *value-list*. The ODS table name is R.

RC<=*value-list*>
produces the Cholesky root of blocks of the estimated **R** matrix. The *value-list* specification is the same as with the R option. The ODS table name is CholR.

RCI<=*value-list*>
produces the inverse Cholesky root of blocks of the estimated **R** matrix. The *value-list* specification is the same as with the R option. The ODS table name is InvCholR.

RCORR<=*value-list*>
produces the correlation matrix corresponding to blocks of the estimated **R** matrix. The *value-list* specification is the same as with the R option. The ODS table name is RCorr.

RI<=*value-list*>
produces the inverse of blocks of the estimated **R** matrix. The *value-list* specification is the same as with the R option. The ODS table name is InvR.

SSCP
requests that an unstructured **R** matrix be estimated from the sum-of-squares-and-crossproducts matrix of the residuals. It applies only when you specify TYPE=UN and have no RANDOM statements. Also, you must have a sufficient number of subjects for the estimate to be positive definite.

This option is useful when the size of the blocks of **R** is large (for example, greater than 10) and you want to use or inspect an unstructured estimate that is much quicker to compute than the default REML estimate. The two estimates will agree for certain balanced data sets when you have a classification fixed effect defined across all time points within a subject.

SUBJECT=*effect*
SUB=*effect*
identifies the subjects in your mixed model. Complete independence is assumed across subjects; therefore, the SUBJECT= option produces a block-diagonal structure in **R** with identical blocks. When the SUBJECT= effect consists entirely of classification variables, the blocks of **R** correspond to observations sharing the same level of that effect. These blocks are sorted according to this effect as well.

Continuous variables are permitted as arguments to the SUBJECT= option. PROC MIXED does not sort by the values of the continuous variable; rather, it considers the data to be from a new subject or group whenever the value of the continuous variable changes from the previous observation. Using a continuous variable decreases execution time for models with a large number of subjects or groups and also prevents the production of a large “Class Level Information” table.

If you want to model nonzero covariance among all of the observations in your SAS data set, specify SUBJECT=INTERCEPT to treat the data as if they are all from one subject. However, be aware that in this case PROC MIXED manipulates an **R** matrix with dimensions equal to the number of observations. If no SUBJECT= effect is specified, then every observation is assumed to be from a different subject and **R** is assumed to be diagonal. For this reason, you usually want to use the SUBJECT= option in the REPEATED statement.

TYPE=*covariance-structure*

specifies the covariance structure of the **R** matrix. The SUBJECT= option defines the blocks of **R**, and the TYPE= option specifies the structure of these blocks. Valid values for *covariance-structure* and their descriptions are provided in Table 84.17 and Table 84.18. The default structure is VC.

Table 84.17 Covariance Structures

Structure	Description	Parms	(i, j) element
ANTE(1)	Antedependence	$2t-1$	$\sigma_i\sigma_j\prod_{k=i}^{j-1}\rho_k$
AR(1)	Autoregressive(1)	2	$\sigma^2\rho^{\lvert i-j\rvert}$
ARH(1)	Heterogeneous AR(1)	$t+1$	$\sigma_i\sigma_j\rho^{\lvert i-j\rvert}$
ARMA(1,1)	ARMA(1,1)	3	$\sigma^2[\gamma\rho^{\lvert i-j\rvert-1}1(i\neq j)+1(i=j)]$
CS	Compound symmetry	2	$\sigma_1+\sigma^2 1(i=j)$
CSH	Heterogeneous CS	$t+1$	$\sigma_i\sigma_j[\rho 1(i\neq j)+1(i=j)]$
FA(q)	Factor analytic	$\frac{q}{2}(2t-q+1)+t$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}+\sigma_i^2 1(i=j)$
FA0(q)	No diagonal FA	$\frac{q}{2}(2t-q+1)$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}$
FA1(q)	Equal diagonal FA	$\frac{q}{2}(2t-q+1)+1$	$\Sigma_{k=1}^{\min(i,j,q)}\lambda_{ik}\lambda_{jk}+\sigma^2 1(i=j)$
HF	Huynh-Feldt	$t+1$	$(\sigma_i^2+\sigma_j^2)/2+\lambda 1(i\neq j)$
LIN(q)	General linear	q	$\Sigma_{k=1}^{q}\theta_k\mathbf{A}_{ij}$
TOEP	Toeplitz	t	$\sigma_{\lvert i-j\rvert+1}$
TOEP(q)	Banded Toeplitz	q	$\sigma_{\lvert i-j\rvert+1}1(\lvert i-j\rvert<q)$
TOEPH	Heterogeneous TOEP	$2t-1$	$\sigma_i\sigma_j\rho_{\lvert i-j\rvert}$
TOEPH(q)	Banded hetero TOEP	$t+q-1$	$\sigma_i\sigma_j\rho_{\lvert i-j\rvert}1(\lvert i-j\rvert<q)$
UN	Unstructured	$t(t+1)/2$	σ_{ij}
UN(q)	Banded	$\frac{q}{2}(2t-q+1)$	$\sigma_{ij}1(\lvert i-j\rvert<q)$
UNR	Unstructured corrs	$t(t+1)/2$	$\sigma_i\sigma_j\rho_{\max(i,j)\min(i,j)}$
UNR(q)	Banded correlations	$\frac{q}{2}(2t-q+1)$	$\sigma_i\sigma_j\rho_{\max(i,j)\min(i,j)}$
UN@AR(1)	Direct product AR(1)	$t_1(t_1+1)/2+1$	$\sigma_{i_1j_1}\rho^{\lvert i_2-j_2\rvert}$
UN@CS	Direct product CS	$t_1(t_1+1)/2+1$	$\begin{cases}\sigma_{i_1j_1} & i_2=j_2\\ \sigma^2\sigma_{i_1j_1} & i_2\neq j_2\\ 0\le\sigma^2\le 1\end{cases}$
UN@UN	Direct product UN	$t_1(t_1+1)/2+$ $t_2(t_2+1)/2-1$	$\sigma_{1,i_1j_1}\sigma_{2,i_2j_2}$
VC	Variance components	q	$\sigma_k^2 1(i=j)$ and i corresponds to kth effect

In Table 84.17, "Parms" is the number of covariance parameters in the structure, t is the overall dimension of the covariance matrix, and $1(A)$ equals 1 when A is true and 0 otherwise. For example, $1(i=j)$ equals 1 when $i=j$ and 0 otherwise, and $1(|i-j|<q)$ equals 1 when $|i-j|<q$ and 0 otherwise. For the TYPE=TOEPH structures, $\rho_0=1$, and for the TYPE=UNR structures,

$\rho_{ii} = 1$ for all i. For the direct product structures, the subscripts "1" and "2" see the first and second structure in the direct product, respectively, and $i_1 = \text{int}((i + t_2 - 1)/t_2)$, $j_1 = \text{int}((j + t_2 - 1)/t_2)$, $i_2 = \text{mod}(i - 1, t_2) + 1$, and $j_2 = \text{mod}(j - 1, t_2) + 1$.

Table 84.18 Spatial Covariance Structures

Structure	Description	Parms	(i, j) element
SP(EXP)(*c-list*)	Exponential	2	$\sigma^2 \exp\{-d_{ij}/\theta\}$
SP(EXPA)(*c-list*)	Anisotropic exponential	$2c + 1$	$\sigma^2 \prod_{k=1}^{c} \exp\{-\theta_k d(i, j, k)^{p_k}\}$
SP(EXPGA)(c_1 c_2)	2D exponential, geometrically anisotropic	4	$\sigma^2 \exp\{-d_{ij}(\theta, \lambda)/\rho\}$
SP(GAU)(*c-list*)	Gaussian	2	$\sigma^2 \exp\{-d_{ij}^2/\rho^2\}$
SP(GAUGA)(c_1 c_2)	2D Gaussian, geometrically anisotropic	4	$\sigma^2 \exp\{-d_{ij}(\theta, \lambda)^2/\rho^2\}$
SP(LIN)(*c-list*)	Linear	2	$\sigma^2(1 - \rho d_{ij})\ 1(\rho d_{ij} \le 1)$
SP(LINL)(*c-list*)	Linear log	2	$\sigma^2(1 - \rho \log(d_{ij}))$ $\times 1(\rho \log(d_{ij}) \le 1, d_{ij} > 0)$
SP(LEAR)(*c-list*)	Linear exponent autoregressive	3	$\sigma^2 \rho^{d_{\min} + \delta[(d_{ij} - d_{\min})/(d_{\max} - d_{\min})]}$
SP(MATERN)(*c-list*)	Matérn	3	$\sigma^2 \frac{1}{\Gamma(\nu)} \left(\frac{d_{ij}}{2\rho}\right)^{\nu} 2K_{\nu}(d_{ij}/\rho)$
SP(MATHSW)(*c-list*)	Matérn (Handcock-Stein-Wallis)	3	$\sigma^2 \frac{1}{\Gamma(\nu)} \left(\frac{d_{ij}\sqrt{\nu}}{\rho}\right)^{\nu} 2K_{\nu}\left(\frac{2d_{ij}\sqrt{\nu}}{\rho}\right)$
SP(POW)(*c-list*)	Power	2	$\sigma^2 \rho^{d_{ij}}$
SP(POWA)(*c-list*)	Anisotropic power	$c + 1$	$\sigma^2 \rho_1^{d(i,j,1)} \rho_2^{d(i,j,2)} \ldots \rho_c^{d(i,j,c)}$
SP(SPH)(*c-list*)	Spherical	2	$\sigma^2[1 - (\frac{3d_{ij}}{2\rho}) + (\frac{d_{ij}^3}{2\rho^3})]\ 1(d_{ij} \le \rho)$
SP(SPHGA)(c_1 c_2)	2D spherical, geometrically anisotropic	4	$\sigma^2[1 - (\frac{3d_{ij}(\theta,\lambda)}{2\rho}) + (\frac{d_{ij}(\theta,\lambda)^3}{2\rho^3})]$ $\times 1(d_{ij}(\theta, \lambda) \le \rho)$

In Table 84.18, *c-list* contains the names of the numeric variables used as coordinates of the location of the observation in space, and d_{ij} is the Euclidean distance between the *i*th and *j*th vectors of these coordinates, which correspond to the *i*th and *j*th observations in the input data set. For SP(POWA) and SP(EXPA), *c* is the number of coordinates, and $d(i, j, k)$ is the absolute distance between the *k*th coordinate, $k = 1, \ldots, c$, of the *i*th and *j*th observations in the input data set. For the geometrically anisotropic structures SP(EXPGA), SP(GAUGA), and SP(SPHGA), exactly two spatial coordinate variables must be specified as c_1 and c_2. Geometric anisotropy is corrected by applying a rotation θ and scaling λ to the coordinate system, and $d_{ij}(\theta, \lambda)$ represents the Euclidean distance between two points in the transformed space. SP(MATERN) and SP(MATHSW) represent covariance structures in a class defined by Matérn (see Matérn 1986; Handcock and Stein 1993; Handcock and Wallis 1994). The function K_{ν} is the modified Bessel function of the second kind of (real) order $\nu > 0$; the parameter ν governs the smoothness of the process (see below for more details).

Table 84.19 lists some examples of the structures in Table 84.17 and Table 84.18.

Table 84.19 Covariance Structure Examples

Description	Structure	Example
Variance components	VC (default)	$\begin{bmatrix} \sigma_B^2 & 0 & 0 & 0 \\ 0 & \sigma_B^2 & 0 & 0 \\ 0 & 0 & \sigma_{AB}^2 & 0 \\ 0 & 0 & 0 & \sigma_{AB}^2 \end{bmatrix}$
Compound symmetry	CS	$\begin{bmatrix} \sigma^2+\sigma_1 & \sigma_1 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma^2+\sigma_1 & \sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma^2+\sigma_1 & \sigma_1 \\ \sigma_1 & \sigma_1 & \sigma_1 & \sigma^2+\sigma_1 \end{bmatrix}$
Unstructured	UN	$\begin{bmatrix} \sigma_1^2 & \sigma_{21} & \sigma_{31} & \sigma_{41} \\ \sigma_{21} & \sigma_2^2 & \sigma_{32} & \sigma_{42} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 & \sigma_{43} \\ \sigma_{41} & \sigma_{42} & \sigma_{43} & \sigma_4^2 \end{bmatrix}$
Banded main diagonal	UN(1)	$\begin{bmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & 0 \\ 0 & 0 & 0 & \sigma_4^2 \end{bmatrix}$
First-order autoregressive	AR(1)	$\sigma^2 \begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 \\ \rho & 1 & \rho & \rho^2 \\ \rho^2 & \rho & 1 & \rho \\ \rho^3 & \rho^2 & \rho & 1 \end{bmatrix}$
Toeplitz	TOEP	$\begin{bmatrix} \sigma^2 & \sigma_1 & \sigma_2 & \sigma_3 \\ \sigma_1 & \sigma^2 & \sigma_1 & \sigma_2 \\ \sigma_2 & \sigma_1 & \sigma^2 & \sigma_1 \\ \sigma_3 & \sigma_2 & \sigma_1 & \sigma^2 \end{bmatrix}$
Toeplitz with two bands	TOEP(2)	$\begin{bmatrix} \sigma^2 & \sigma_1 & 0 & 0 \\ \sigma_1 & \sigma^2 & \sigma_1 & 0 \\ 0 & \sigma_1 & \sigma^2 & \sigma_1 \\ 0 & 0 & \sigma_1 & \sigma^2 \end{bmatrix}$
Spatial power	SP(POW)(*c*)	$\sigma^2 \begin{bmatrix} 1 & \rho^{d_{12}} & \rho^{d_{13}} & \rho^{d_{14}} \\ \rho^{d_{21}} & 1 & \rho^{d_{23}} & \rho^{d_{24}} \\ \rho^{d_{31}} & \rho^{d_{32}} & 1 & \rho^{d_{34}} \\ \rho^{d_{41}} & \rho^{d_{42}} & \rho^{d_{43}} & 1 \end{bmatrix}$
Heterogeneous AR(1)	ARH(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho^2 & \sigma_1\sigma_4\rho^3 \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho^2 \\ \sigma_3\sigma_1\rho^2 & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho^3 & \sigma_4\sigma_2\rho^2 & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$
First-order autoregressive moving average	ARMA(1,1)	$\sigma^2 \begin{bmatrix} 1 & \gamma & \gamma\rho & \gamma\rho^2 \\ \gamma & 1 & \gamma & \gamma\rho \\ \gamma\rho & \gamma & 1 & \gamma \\ \gamma\rho^2 & \gamma\rho & \gamma & 1 \end{bmatrix}$

Table 84.19 *continued*

Description	Structure	Example
Heterogeneous CS	CSH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho & \sigma_1\sigma_3\rho & \sigma_1\sigma_4\rho \\ \sigma_2\sigma_1\rho & \sigma_2^2 & \sigma_2\sigma_3\rho & \sigma_2\sigma_4\rho \\ \sigma_3\sigma_1\rho & \sigma_3\sigma_2\rho & \sigma_3^2 & \sigma_3\sigma_4\rho \\ \sigma_4\sigma_1\rho & \sigma_4\sigma_2\rho & \sigma_4\sigma_3\rho & \sigma_4^2 \end{bmatrix}$
First-order factor analytic	FA(1)	$\begin{bmatrix} \lambda_1^2+d_1 & \lambda_1\lambda_2 & \lambda_1\lambda_3 & \lambda_1\lambda_4 \\ \lambda_2\lambda_1 & \lambda_2^2+d_2 & \lambda_2\lambda_3 & \lambda_2\lambda_4 \\ \lambda_3\lambda_1 & \lambda_3\lambda_2 & \lambda_3^2+d_3 & \lambda_3\lambda_4 \\ \lambda_4\lambda_1 & \lambda_4\lambda_2 & \lambda_4\lambda_3 & \lambda_4^2+d_4 \end{bmatrix}$
Huynh-Feldt	HF	$\begin{bmatrix} \sigma_1^2 & \frac{\sigma_1^2+\sigma_2^2}{2}-\lambda & \frac{\sigma_1^2+\sigma_3^2}{2}-\lambda \\ \frac{\sigma_2^2+\sigma_1^2}{2}-\lambda & \sigma_2^2 & \frac{\sigma_2^2+\sigma_3^2}{2}-\lambda \\ \frac{\sigma_3^2+\sigma_1^2}{2}-\lambda & \frac{\sigma_3^2+\sigma_2^2}{2}-\lambda & \sigma_3^2 \end{bmatrix}$
First-order antedependence	ANTE(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_1 & \sigma_1\sigma_3\rho_1\rho_2 \\ \sigma_2\sigma_1\rho_1 & \sigma_2^2 & \sigma_2\sigma_3\rho_2 \\ \sigma_3\sigma_1\rho_2\rho_1 & \sigma_3\sigma_2\rho_2 & \sigma_3^2 \end{bmatrix}$
Heterogeneous Toeplitz	TOEPH	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_1 & \sigma_1\sigma_3\rho_2 & \sigma_1\sigma_4\rho_3 \\ \sigma_2\sigma_1\rho_1 & \sigma_2^2 & \sigma_2\sigma_3\rho_1 & \sigma_2\sigma_4\rho_2 \\ \sigma_3\sigma_1\rho_2 & \sigma_3\sigma_2\rho_1 & \sigma_3^2 & \sigma_3\sigma_4\rho_1 \\ \sigma_4\sigma_1\rho_3 & \sigma_4\sigma_2\rho_2 & \sigma_4\sigma_3\rho_1 & \sigma_4^2 \end{bmatrix}$
Unstructured correlations	UNR	$\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_{21} & \sigma_1\sigma_3\rho_{31} & \sigma_1\sigma_4\rho_{41} \\ \sigma_2\sigma_1\rho_{21} & \sigma_2^2 & \sigma_2\sigma_3\rho_{32} & \sigma_2\sigma_4\rho_{42} \\ \sigma_3\sigma_1\rho_{31} & \sigma_3\sigma_2\rho_{32} & \sigma_3^2 & \sigma_3\sigma_4\rho_{43} \\ \sigma_4\sigma_1\rho_{41} & \sigma_4\sigma_2\rho_{42} & \sigma_4\sigma_3\rho_{43} & \sigma_4^2 \end{bmatrix}$
Direct product AR(1)	UN@AR(1)	$\begin{bmatrix} \sigma_1^2 & \sigma_{21} \\ \sigma_{21} & \sigma_2^2 \end{bmatrix} \otimes \begin{bmatrix} 1 & \rho & \rho^2 \\ \rho & 1 & \rho \\ \rho^2 & \rho & 1 \end{bmatrix} =$
		$\begin{bmatrix} \sigma_1^2 & \sigma_1^2\rho & \sigma_1^2\rho^2 & \sigma_{21} & \sigma_{21}\rho & \sigma_{21}\rho^2 \\ \sigma_1^2\rho & \sigma_1^2 & \sigma_1^2\rho & \sigma_{21}\rho & \sigma_{21} & \sigma_{21}\rho \\ \sigma_1^2\rho^2 & \sigma_1^2\rho & \sigma_1^2 & \sigma_{21}\rho^2 & \sigma_{21}\rho & \sigma_{21} \\ \sigma_{21} & \sigma_{21}\rho & \sigma_{21}\rho^2 & \sigma_2^2 & \sigma_2^2\rho & \sigma_2^2\rho^2 \\ \sigma_{21}\rho & \sigma_{21} & \sigma_{21}\rho & \sigma_2^2\rho & \sigma_2^2 & \sigma_2^2\rho \\ \sigma_{21}\rho^2 & \sigma_{21}\rho & \sigma_{21} & \sigma_2^2\rho^2 & \sigma_2^2\rho & \sigma_2^2 \end{bmatrix}$

The following provides some further information about these covariance structures:

ANTE(1) specifies the first-order antedependence structure (see Kenward 1987; Patel 1991; Macchiavelli and Arnold 1994). In Table 84.17, σ_i^2 is the ith variance parameter, and ρ_k is the kth autocorrelation parameter satisfying $|\rho_k| < 1$.

AR(1) specifies a first-order autoregressive structure. PROC MIXED imposes the constraint $|\rho| < 1$ for stationarity.

ARH(1) specifies a heterogeneous first-order autoregressive structure. As with TYPE=AR(1), PROC MIXED imposes the constraint $|\rho| < 1$ for stationarity.

ARMA(1,1) specifies the first-order autoregressive moving-average structure. In Table 84.17, ρ is the autoregressive parameter, γ models a moving-average component, and σ^2 is the residual variance. In the notation of Fuller (1976, p. 68), $\rho = \theta_1$ and

$$\gamma = \frac{(1 + b_1\theta_1)(\theta_1 + b_1)}{1 + b_1^2 + 2b_1\theta_1}$$

The example in Table 84.19 and $|b_1| < 1$ imply that

$$b_1 = \frac{\beta - \sqrt{\beta^2 - 4\alpha^2}}{2\alpha}$$

where $\alpha = \gamma - \rho$ and $\beta = 1 + \rho^2 - 2\gamma\rho$. PROC MIXED imposes the constraints $|\rho| < 1$ and $|\gamma| < 1$ for stationarity, although for some values of ρ and γ in this region the resulting covariance matrix is not positive definite. When the estimated value of ρ becomes negative, the computed covariance is multiplied by $\cos(\pi d_{ij})$ to account for the negativity.

CS specifies the compound-symmetry structure, which has constant variance and constant covariance.

CSH specifies the heterogeneous compound-symmetry structure. This structure has a different variance parameter for each diagonal element, and it uses the square roots of these parameters in the off-diagonal entries. In Table 84.17, σ_i^2 is the ith variance parameter, and ρ is the correlation parameter satisfying $|\rho| < 1$.

FA(q) specifies the factor-analytic structure with q factors (Jennrich and Schluchter 1986). This structure is of the form $\mathbf{\Lambda\Lambda}' + \mathbf{D}$, where $\mathbf{\Lambda}$ is a $t \times q$ rectangular matrix and $\mathbf{D}$ is a $t \times t$ diagonal matrix with t different parameters. When $q > 1$, the elements of $\mathbf{\Lambda}$ in its upper-right corner (that is, the elements in the ith row and jth column for $j > i$) are set to zero to fix the rotation of the structure.

FA0(q) is similar to the FA(q) structure except that no diagonal matrix $\mathbf{D}$ is included. When $q < t$—that is, when the number of factors is less than the dimension of the matrix—this structure is nonnegative definite but not of full rank. In this situation, you can use it for approximating an unstructured $\mathbf{G}$ matrix in the RANDOM statement or for combining with the LOCAL option in the REPEATED statement. When $q = t$, you can use this structure to constrain $\mathbf{G}$ to be nonnegative definite in the RANDOM statement.

FA1(q) is similar to the TYPE=FA(q) structure except that all of the elements in $\mathbf{D}$ are constrained to be equal. This offers a useful and more parsimonious alternative to the full factor-analytic structure.

HF specifies the Huynh-Feldt covariance structure (Huynh and Feldt 1970). This structure is similar to the TYPE=CSH structure in that it has the same number of parameters and heterogeneity along the main diagonal. However, it constructs the off-diagonal elements by taking arithmetic rather than geometric means.

You can perform a likelihood ratio test of the Huynh-Feldt conditions by running PROC MIXED twice, once with TYPE=HF and once with TYPE=UN, and then subtracting their respective values of –2 times the maximized likelihood.

If PROC MIXED does not converge under your Huynh-Feldt model, you can specify your own starting values with the PARMS statement. The default MIVQUE(0) starting values can sometimes be poor for this structure. A good choice for starting values is often the parameter estimates corresponding to an initial fit that uses TYPE=CS.

LIN(*q*) specifies the general linear covariance structure with q parameters. This structure consists of a linear combination of known matrices that are input with the LDATA= option. This structure is very general, and you need to make sure that the variance matrix is positive definite. By default, PROC MIXED sets the initial values of the parameters to 1. You can use the PARMS statement to specify other initial values.

LINEAR(*q*) is an alias for TYPE=LIN(*q*).

SIMPLE is an alias for TYPE=VC.

SP(EXPA)(*c-list*) specifies the spatial anisotropic exponential structure, where *c-list* is a list of variables indicating the coordinates. This structure has (i, j) element equal to

$$\sigma^2 \prod_{k=1}^{c} \exp\{-\theta_k d(i, j, k)^{p_k}\}$$

where c is the number of coordinates and $d(i, j, k)$ is the absolute distance between the kth coordinate $(k = 1, \ldots, c)$ of the ith and jth observations in the input data set. There are $2c + 1$ parameters to be estimated: θ_k, p_k $(k = 1, \ldots, c)$, and σ^2.

You might want to constrain some of the EXPA parameters to known values. For example, suppose you have three coordinate variables C1, C2, and C3 and you want to constrain the powers p_k to equal 2, as in Sacks et al. (1989). Suppose further that you want to model covariance across the entire input data set and you suspect the θ_k and σ^2 estimates are close to 3, 4, 5, and 1, respectively. Then specify the following statements:

```
repeated / type=sp(expa)(c1 c2 c3)
   subject=intercept;
parms (3) (4) (5) (2) (2) (2) (1) /
   hold=4,5,6;
```

SP(EXPGA)(c_1 c_2) specify modification of the isotropic SP(EXP) covariance structure.

SP(GAUGA)(c_1 c_2) specify modification of the isotropic SP(GAU) covariance structure.

SP(SPHGA)(c_1 c_2) specify modification of the isotropic SP(SPH) covariance structure.

These are structures that allow for geometric anisotropy in two dimensions. The coordinates are specified by the variables c1 and c2.

If the spatial process is geometrically anisotropic in $\mathbf{c} = [c_{i1}, c_{i2}]$, then it is isotropic in the coordinate system

$$\mathbf{Ac} = \begin{bmatrix} 1 & 0 \\ 0 & \lambda \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \mathbf{c} = \mathbf{c}^*$$

for a properly chosen angle θ and scaling factor λ. Elliptical isocorrelation contours are thereby transformed to spherical contours, adding two parameters to the respective isotropic covariance structures. Euclidean distances (see Table 84.18) are expressed in terms of $\mathbf{c}^*$.

The angle θ of the clockwise rotation is reported in radians, $0 \leq \theta \leq 2\pi$. The scaling parameter λ represents the ratio of the range parameters in the direction of the major and minor axis of the correlation contours. In other words, following a rotation of the coordinate system by angle θ, isotropy is achieved by compressing or magnifying distances in one coordinate by the factor λ.

Fixing $\lambda = 1.0$ reduces the models to isotropic ones for any angle of rotation. If the scaling parameter is held constant at 1.0, you should also hold constant the angle of rotation, as in the following statements:

```
repeated / type=sp(expga)(gxc gyc)
           subject=intercept;
parms (6) (1.0) (0.0) (1) / hold=2,3;
```

If λ is fixed at any other value than 1.0, the angle of rotation can be estimated. Specifying a starting grid of angles and scaling factors can considerably improve the convergence properties of the optimization algorithm for these models. Only a single random effect with geometrically anisotropic structure is permitted.

SP(MATERN)(*c-list*) | **SP(MATHSW)**(*c-list*) specifies covariance structures in the Matérn class of covariance functions (Matérn 1986). Two observations for the same subject (block of $\mathbf{R}$) that are Euclidean distance d_{ij} apart have covariance

$$\sigma^2 \frac{1}{\Gamma(\nu)} \left(\frac{d_{ij}}{2\rho}\right)^{\nu} 2K_{\nu}(d_{ij}/\rho) \qquad \nu > 0,\ \rho > 0$$

where K_{ν} is the modified Bessel function of the second kind of (real) order $\nu > 0$. The smoothness (continuity) of a stochastic process with covariance function in this class increases with ν. The Matérn class thus enables data-driven estimation of the smoothness properties. The covariance is identical to the exponential model for $\nu = 0.5$ (TYPE=SP(EXP)(*c-list*)), while for $\nu = 1$ the model advocated by Whittle (1954) results. As $\nu \to \infty$ the model approaches the gaussian covariance structure (TYPE=SP(GAU)(*c-list*)).

The MATHSW structure represents the Matérn class in the parameterization of Handcock and Stein (1993) and Handcock and Wallis (1994),

$$\sigma^2 \frac{1}{\Gamma(\nu)} \left(\frac{d_{ij}\sqrt{\nu}}{\rho}\right)^{\nu} 2K_{\nu}\left(\frac{2d_{ij}\sqrt{\nu}}{\rho}\right)$$

Since computation of the function K_{ν} and its derivatives is numerically very intensive, fitting models with Matérn covariance structures can be more time-consuming than with other spatial covariance structures. Good starting values are essential.

SP(POW)(*c-list*) | **SP(POWA)**(*c-list*) specifies the spatial power structures. When the estimated value of ρ becomes negative, the computed covariance is multiplied by $\cos(\pi d_{ij})$ to account for the negativity.

SP(LEAR)(*c-list*) specifies a linear exponent autoregressive (LEAR) correlation structure as proposed by Simpson et al. (2010). For two observations with distance metric d_{ij}, the covariance is

$$\mathrm{Cov}\left[\xi_i, \xi_j\right] = \sigma^2 \begin{cases} \rho^{d_{\min}+\delta[(d_{ij}-d_{\min})/(d_{\max}-d_{\min})]} & i \neq j \text{ and } d_{\min} \neq d_{\max} \\ \rho^{d_{\min}} & i \neq j \text{ and } d_{\min} = d_{\max} \\ 1 & i = j \end{cases}$$

where $d_{\min}$ and $d_{\max}$ are the smallest and largest distance between any two observations, $\delta \geq 0$ is the decay speed, and $0 \leq \rho < 1$. See TYPE=SP(EXP) for the computation of the distance d_{ij} from the variables specified in *c-list*. When the estimated value of ρ becomes negative, the computed covariance is multiplied by $\cos(\pi d_{ij})$ to account for the negativity.

For power analysis of repeated measures designs that have a LEAR correlation structure, see the section "POWER Statement" on page 4370 in Chapter 55, "The GLMPOWER Procedure."

Note that TYPE=SP(LEAR) is not supported for GROUP= option in this SAS release.

TOEP<(*q*)> specifies a banded Toeplitz structure. This can be viewed as a moving-average structure with order equal to $q - 1$. The TYPE=TOEP option is a full Toeplitz matrix, which can be viewed as an autoregressive structure with order equal to the dimension of the matrix. The specification TYPE=TOEP(1) is the same as $\sigma^2 I$, where I is an identity matrix, and it can be useful for specifying the same variance component for several effects.

TOEPH<(*q*)> specifies a heterogeneous banded Toeplitz structure. In Table 84.17, σ_i^2 is the ith variance parameter and ρ_j is the jth correlation parameter satisfying $|\rho_j| < 1$. If you specify the order parameter q, then PROC MIXED estimates only the first q bands of the matrix, setting all higher bands equal to 0. The option TOEPH(1) is equivalent to both the TYPE=UN(1) and TYPE=UNR(1) options.

UN<(*q*)> specifies a completely general (unstructured) covariance matrix parameterized directly in terms of variances and covariances. The variances are constrained to be nonnegative, and the covariances are unconstrained. This structure is not constrained to be nonnegative definite in order to avoid nonlinear constraints; however, you can use the TYPE=FA0 structure if you want this constraint to be imposed by a Cholesky factorization. If you specify the order parameter q, then PROC MIXED estimates only the first q bands of the matrix, setting all higher bands equal to 0.

UNR<(*q*)> specifies a completely general (unstructured) covariance matrix parameterized in terms of variances and correlations. This structure fits the same model as the TYPE=UN(q) option but with a different parameterization. The ith variance parameter is σ_i^2. The parameter ρ_{jk} is the correlation between the jth and kth measurements; it satisfies $|\rho_{jk}| < 1$. If you specify the order parameter r, then PROC MIXED estimates only the first q bands of the matrix, setting all higher bands equal to zero.

UN@AR(1) | UN@CS | UN@UN specify direct (Kronecker) product structures designed for multivariate repeated measures (see Galecki 1994). These structures are constructed by taking the Kronecker product of an unstructured matrix (modeling covariance

across the multivariate observations) with an additional covariance matrix (modeling covariance across time or another factor). The upper-left value in the second matrix is constrained to equal 1 to identify the model. See the *SAS/IML User's Guide* for more details about direct products.

To use these structures in the REPEATED statement, you must specify two distinct REPEATED effects, both of which must be included in the CLASS statement. The first effect indicates the multivariate observations, and the second identifies the levels of time or some additional factor. Note that the input data set must still be constructed in "univariate" format; that is, all dependent observations are still listed observation-wise in one single variable. Although this construction provides for general modeling possibilities, it forces you to construct variables indicating both dimensions of the Kronecker product.

For example, suppose your observed data consist of heights and weights of several children measured over several successive years. Your input data set should then contain variables similar to the following:

- Y, all of the heights and weights, with a separate observation for each
- Var, indicating whether the measurement is a height or a weight
- Year, indicating the year of measurement
- Child, indicating the child on which the measurement was taken

Your PROC MIXED statements for a Kronecker AR(1) structure across years would then be as follows:

```
proc mixed;
   class Var Year Child;
   model Y = Var Year Var*Year;
   repeated Var Year / type=un@ar(1)
                       subject=Child;
run;
```

You should nearly always want to model different means for the multivariate observations; hence the inclusion of Var in the MODEL statement. The preceding mean model consists of cell means for all combinations of VAR and YEAR.

VC specifies standard variance components and is the default structure for both the RANDOM and REPEATED statements. In the RANDOM statement, a distinct variance component is assigned to each effect. In the REPEATED statement, this structure is usually used only with the GROUP= option to specify a heterogeneous variance model.

Jennrich and Schluchter (1986) provide general information about the use of covariance structures, and Wolfinger (1996) presents details about many of the heterogeneous structures. Modeling with spatial covariance structures is discussed in many sources (Marx and Thompson 1987; Zimmerman and Harville 1991; Cressie 1993; Brownie, Bowman, and Burton 1993; Stroup, Baenziger, and Mulitze 1994; Brownie and Gumpertz 1997; Gotway and Stroup 1997; Chilès and Delfiner 1999; Schabenberger and Gotway 2005; Littell et al. 2006).

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

NOTE: Use the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics," only for definitions of the options that you can use with the SLICE statement. PROC MIXED uses a slightly different syntax for the LSMEANS, which is described in the section "LSMEANS Statement" on page 6794.

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

If you do not specify a REPEATED statement, the WEIGHT statement operates exactly like the one in PROC GLM. In this case PROC MIXED replaces $\mathbf{X}'\mathbf{X}$ and $\mathbf{Z}'\mathbf{Z}$ with $\mathbf{X}'\mathbf{W}\mathbf{X}$ and $\mathbf{Z}'\mathbf{W}\mathbf{Z}$, where $\mathbf{W}$ is the diagonal weight matrix. If you specify a REPEATED statement, then the WEIGHT statement replaces $\mathbf{R}$ with $\mathbf{LRL}$, where $\mathbf{L}$ is a diagonal matrix with elements $\mathbf{W}^{-1/2}$. Observations with nonpositive or missing weights are not included in the PROC MIXED analysis.

If a computation in PROC MIXED involves $\mathbf{R}$, then the WEIGHT statement replaces $\mathbf{R}$ with $\mathbf{W}^{-1/2}\mathbf{R}\mathbf{W}^{-1/2}$. For example, the covariance matrix $\mathbf{V}$ for the observations usually have the form $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$, which with the WEIGHT statement becomes $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{W}^{-1/2}\mathbf{R}\mathbf{W}^{-1/2}$.

Details: MIXED Procedure

Mixed Models Theory

This section provides an overview of a likelihood-based approach to general linear mixed models. This approach simplifies and unifies many common statistical analyses, including those involving repeated measures, random effects, and random coefficients. The basic assumption is that the data are linearly related to unobserved multivariate normal random variables. For extensions to nonlinear and nonnormal situations see the documentation of the GLIMMIX and NLMIXED procedures. Additional theory and examples are provided in Littell et al. (2006); Verbeke and Molenberghs (1997, 2000); Brown and Prescott (1999).

Matrix Notation

Suppose that you observe n data points $y_1, \ldots, y_n$ and that you want to explain them by using n values for each of p explanatory variables $x_{11}, \ldots, x_{1p}, x_{21}, \ldots, x_{2p}, \ldots, x_{n1}, \ldots, x_{np}$. The x_{ij} values can be either regression-type continuous variables or dummy variables indicating class membership. The standard linear model for this setup is

$$y_i = \sum_{j=1}^{p} x_{ij}\beta_j + \epsilon_i \quad i = 1, \ldots, n$$

where $\beta_1, \ldots, \beta_p$ are unknown *fixed-effects* parameters to be estimated and $\epsilon_1, \ldots, \epsilon_n$ are unknown independent and identically distributed normal (Gaussian) random variables with mean 0 and variance σ^2.

The preceding equations can be written simultaneously by using vectors and a matrix, as follows:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1p} \\ x_{21} & x_{22} & \ldots & x_{2p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{np} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

For convenience, simplicity, and extendability, this entire system is written as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\mathbf{y}$ denotes the vector of observed y_i's, $\mathbf{X}$ is the known matrix of x_{ij}'s, $\boldsymbol{\beta}$ is the unknown fixed-effects parameter vector, and $\boldsymbol{\epsilon}$ is the unobserved vector of independent and identically distributed Gaussian random errors.

In addition to denoting data, random variables, and explanatory variables in the preceding fashion, the subsequent development makes use of basic matrix operators such as transpose ($'$), inverse ($^{-1}$), generalized inverse ($^{-}$), determinant ($|\cdot|$), and matrix multiplication. See Searle (1982) for details about these and other matrix techniques.

Formulation of the Mixed Model

The previous general linear model is certainly a useful one (Searle 1971), and it is the one fitted by the GLM procedure. However, many times the distributional assumption about ϵ is too restrictive. The mixed model extends the general linear model by allowing a more flexible specification of the covariance matrix of ϵ. In other words, it allows for both correlation and heterogeneous variances, although you still assume normality.

The mixed model is written as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

where everything is the same as in the general linear model except for the addition of the known design matrix, $\mathbf{Z}$, and the vector of unknown *random-effects parameters,* $\boldsymbol{\gamma}$. The matrix $\mathbf{Z}$ can contain either continuous or dummy variables, just like $\mathbf{X}$. The name *mixed model* comes from the fact that the model contains both fixed-effects parameters, $\boldsymbol{\beta}$, and random-effects parameters, $\boldsymbol{\gamma}$. See Henderson (1990) and Searle, Casella, and McCulloch (1992) for historical developments of the mixed model.

A key assumption in the foregoing analysis is that $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$ are normally distributed with

$$\mathrm{E}\begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\epsilon} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$
$$\mathrm{Var}\begin{bmatrix} \boldsymbol{\gamma} \\ \boldsymbol{\epsilon} \end{bmatrix} = \begin{bmatrix} \mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{bmatrix}$$

The variance of $\mathbf{y}$ is, therefore, $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$. You can model $\mathbf{V}$ by setting up the random-effects design matrix $\mathbf{Z}$ and by specifying covariance structures for $\mathbf{G}$ and $\mathbf{R}$.

Note that this is a general specification of the mixed model, in contrast to many texts and articles that discuss only simple random effects. Simple random effects are a special case of the general specification with $\mathbf{Z}$ containing dummy variables, $\mathbf{G}$ containing variance components in a diagonal structure, and $\mathbf{R} = \sigma^2\mathbf{I}_n$, where $\mathbf{I}_n$ denotes the $n \times n$ identity matrix. The general linear model is a further special case with $\mathbf{Z} = \mathbf{0}$ and $\mathbf{R} = \sigma^2\mathbf{I}_n$.

The following two examples illustrate the most common formulations of the general linear mixed model.

Example: Growth Curve with Compound Symmetry

Suppose that you have three growth curve measurements for s individuals and that you want to fit an overall linear trend in time. Your $\mathbf{X}$ matrix is as follows:

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ \vdots & \vdots \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}$$

The first column (coded entirely with 1s) fits an intercept, and the second column (coded with times of $1, 2, 3$) fits a slope. Here, $n = 3s$ and $p = 2$.

Suppose further that you want to introduce a common correlation among the observations from a single individual, with correlation being the same for all individuals. One way of setting this up in the general mixed

model is to eliminate the **Z** and **G** matrices and let the **R** matrix be block diagonal with blocks corresponding to the individuals and with each block having the *compound-symmetry* structure. This structure has two unknown parameters, one modeling a common covariance and the other modeling a residual variance. The form for **R** would then be as follows:

$$\mathbf{R} = \begin{bmatrix} \sigma_1^2+\sigma^2 & \sigma_1^2 & \sigma_1^2 & & & & \\ \sigma_1^2 & \sigma_1^2+\sigma^2 & \sigma_1^2 & & & & \\ \sigma_1^2 & \sigma_1^2 & \sigma_1^2+\sigma^2 & & & & \\ & & & \ddots & & & \\ & & & & \sigma_1^2+\sigma^2 & \sigma_1^2 & \sigma_1^2 \\ & & & & \sigma_1^2 & \sigma_1^2+\sigma^2 & \sigma_1^2 \\ & & & & \sigma_1^2 & \sigma_1^2 & \sigma_1^2+\sigma^2 \end{bmatrix}$$

where blanks denote zeros. There are $3s$ rows and columns altogether, and the common correlation is $\sigma_1^2/(\sigma_1^2+\sigma^2)$.

The PROC MIXED statements to fit this model are as follows:

```
proc mixed;
   class indiv;
   model y = time;
   repeated / type=cs subject=indiv;
run;
```

Here, indiv is a classification variable indexing individuals. The MODEL statement fits a straight line for time ; the intercept is fit by default just as in PROC GLM. The REPEATED statement models the **R** matrix: TYPE=CS specifies the compound symmetry structure, and SUBJECT=INDIV specifies the blocks of **R**.

An alternative way of specifying the common intra-individual correlation is to let

$$\mathbf{Z} = \begin{bmatrix} 1 & & & \\ 1 & & & \\ 1 & & & \\ & 1 & & \\ & 1 & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \\ & & & 1 \\ & & & 1 \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_1^2 & & \\ & & \ddots & \\ & & & \sigma_1^2 \end{bmatrix}$$

and $\mathbf{R} = \sigma^2\mathbf{I}_n$. The **Z** matrix has $3s$ rows and s columns, and **G** is $s \times s$.

You can set up this model in PROC MIXED in two different but equivalent ways:

```
proc mixed;
   class indiv;
   model y = time;
   random indiv;
run;

proc mixed;
   class indiv;
   model y = time;
   random intercept / subject=indiv;
run;
```

Both of these specifications fit the same model as the previous one that used the REPEATED statement; however, the RANDOM specifications constrain the correlation to be positive, whereas the REPEATED specification leaves the correlation unconstrained.

Example: Split-Plot Design

The split-plot design involves two experimental treatment factors, A and B, and two different sizes of experimental units to which they are applied (see Winer 1971; Snedecor and Cochran 1980; Milliken and Johnson 1992; Steel, Torrie, and Dickey 1997). The levels of A are randomly assigned to the larger-sized experimental unit, called *whole plots*, whereas the levels of B are assigned to the smaller-sized experimental unit, the *subplots*. The subplots are assumed to be nested within the whole plots, so that a whole plot consists of a cluster of subplots and a level of A is applied to the entire cluster.

Such an arrangement is often necessary by nature of the experiment, the classical example being the application of fertilizer to large plots of land and different crop varieties planted in subdivisions of the large plots. For this example, fertilizer is the whole-plot factor A and variety is the subplot factor B.

The first example is a split-plot design for which the whole plots are arranged in a randomized block design. The appropriate PROC MIXED statements are as follows:

```
proc mixed;
   class a b block;
   model y = a|b;
   random block a*block;
run;
```

Here

$$\mathbf{R} = \sigma^2 \mathbf{I}_{24}$$

and **X**, **Z**, and **G** have the following form:

$$
\mathbf{X} = \begin{bmatrix}
1 & 1 & & & 1 & & 1 & & & & & \\
1 & 1 & & & & 1 & & 1 & & & & \\
1 & & 1 & & 1 & & & & 1 & & & \\
1 & & 1 & & & 1 & & & & 1 & & \\
1 & & & 1 & 1 & & & & & & 1 & \\
1 & & & 1 & & 1 & & & & & & 1 \\
\vdots & & \vdots & & \vdots & & & & & \vdots & & \\
1 & 1 & & & 1 & & 1 & & & & & \\
1 & 1 & & & & 1 & & 1 & & & & \\
1 & & 1 & & 1 & & & & 1 & & & \\
1 & & 1 & & & 1 & & & & 1 & & \\
1 & & & 1 & 1 & & & & & & 1 & \\
1 & & & 1 & & 1 & & & & & & 1
\end{bmatrix}
$$

$$
\mathbf{Z} = \begin{bmatrix}
1 & & & & 1 & & & & & & & & & & & \\
1 & & & & 1 & & & & & & & & & & & \\
1 & & & & & 1 & & & & & & & & & & \\
1 & & & & & 1 & & & & & & & & & & \\
1 & & & & & & 1 & & & & & & & & & \\
1 & & & & & & 1 & & & & & & & & & \\
 & 1 & & & & & & 1 & & & & & & & & \\
 & 1 & & & & & & 1 & & & & & & & & \\
 & 1 & & & & & & & 1 & & & & & & & \\
 & 1 & & & & & & & 1 & & & & & & & \\
 & 1 & & & & & & & & 1 & & & & & & \\
 & 1 & & & & & & & & 1 & & & & & & \\
 & & 1 & & & & & & & & 1 & & & & & \\
 & & 1 & & & & & & & & 1 & & & & & \\
 & & 1 & & & & & & & & & 1 & & & & \\
 & & 1 & & & & & & & & & 1 & & & & \\
 & & 1 & & & & & & & & & & 1 & & & \\
 & & 1 & & & & & & & & & & 1 & & & \\
 & & & 1 & & & & & & & & & & 1 & & \\
 & & & 1 & & & & & & & & & & 1 & & \\
 & & & 1 & & & & & & & & & & & 1 & \\
 & & & 1 & & & & & & & & & & & 1 & \\
 & & & 1 & & & & & & & & & & & & 1 \\
 & & & 1 & & & & & & & & & & & & 1
\end{bmatrix}
$$

$$
\mathbf{G} = \begin{bmatrix}
\sigma_B^2 & & & & & & & \\
 & \sigma_B^2 & & & & & & \\
 & & \sigma_B^2 & & & & & \\
 & & & \sigma_B^2 & & & & \\
 & & & & \sigma_{AB}^2 & & & \\
 & & & & & \sigma_{AB}^2 & & \\
 & & & & & & \ddots & \\
 & & & & & & & \sigma_{AB}^2
\end{bmatrix}
$$

where σ_B^2 is the variance component for Block and σ_{AB}^2 is the variance component for A*Block. Changing the RANDOM statement as follows fits the same model, but with **Z** and **G** sorted differently:

```
random int a / subject=block;
```

$$
\mathbf{Z} = \begin{bmatrix}
1 & 1 & & & & & & & & & & & & & & & & & & & & & & \\
1 & 1 & & & & & & & & & & & & & & & & & & & & & & \\
1 & & 1 & & & & & & & & & & & & & & & & & & & & & \\
1 & & 1 & & & & & & & & & & & & & & & & & & & & & \\
1 & & & 1 & & & & & & & & & & & & & & & & & & & & \\
1 & & & 1 & & & & & & & & & & & & & & & & & & & & \\
& & & & 1 & 1 & & & & & & & & & & & & & & & & & & \\
& & & & 1 & 1 & & & & & & & & & & & & & & & & & & \\
& & & & 1 & & 1 & & & & & & & & & & & & & & & & & \\
& & & & 1 & & 1 & & & & & & & & & & & & & & & & & \\
& & & & 1 & & & 1 & & & & & & & & & & & & & & & & \\
& & & & 1 & & & 1 & & & & & & & & & & & & & & & & \\
& & & & & & & & 1 & 1 & & & & & & & & & & & & & & \\
& & & & & & & & 1 & 1 & & & & & & & & & & & & & & \\
& & & & & & & & 1 & & 1 & & & & & & & & & & & & & \\
& & & & & & & & 1 & & 1 & & & & & & & & & & & & & \\
& & & & & & & & 1 & & & 1 & & & & & & & & & & & & \\
& & & & & & & & 1 & & & 1 & & & & & & & & & & & & \\
& & & & & & & & & & & & 1 & 1 & & & & & & & & & & \\
& & & & & & & & & & & & 1 & 1 & & & & & & & & & & \\
& & & & & & & & & & & & 1 & & 1 & & & & & & & & & \\
& & & & & & & & & & & & 1 & & 1 & & & & & & & & & \\
& & & & & & & & & & & & 1 & & & 1 & & & & & & & & \\
& & & & & & & & & & & & 1 & & & 1 & & & & & & & & \\
& & & & & & & & & & & & & & & & 1 & & & & & & & \\
& & & & & & & & & & & & & & & & 1 & & & & & & & \\
& & & & & & & & & & & & & & & & & 1 & & & & & & \\
& & & & & & & & & & & & & & & & & 1 & & & & & & \\
& & & & & & & & & & & & & & & & & & 1 & & & & & \\
& & & & & & & & & & & & & & & & & & 1 & & & & & \\
& & & & & & & & & & & & & & & & & & & 1 & & & & \\
& & & & & & & & & & & & & & & & & & & 1 & & & & \\
& & & & & & & & & & & & & & & & & & & & 1 & & & \\
& & & & & & & & & & & & & & & & & & & & 1 & & & \\
& & & & & & & & & & & & & & & & & & & & & 1 & & \\
& & & & & & & & & & & & & & & & & & & & & 1 & & \\
& & & & & & & & & & & & & & & & & & & & & & 1 & \\
& & & & & & & & & & & & & & & & & & & & & & 1 &
\end{bmatrix}
$$

$$
\mathbf{G} = \begin{bmatrix}
\sigma_B^2 & & & & & & & & & & & & & & & & & & & & & & & \\
& \sigma_{AB}^2 & & & & & & & & & & & & & & & & & & & & & & \\
& & \sigma_{AB}^2 & & & & & & & & & & & & & & & & & & & & & \\
& & & \sigma_{AB}^2 & & & & & & & & & & & & & & & & & & & & \\
& & & & \ddots & & & & & & & & & & & & & & & & & & & \\
& & & & & & \sigma_B^2 & & & & & & & & & & & & & & & & & \\
& & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & & & & \\
& & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & & & \\
& & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & & \\
& & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & & \\
& & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & & \\
& & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & & \\
& & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & & \\
& & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & & \\
& & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & & \\
& & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & & \\
& & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & & \\
& & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & & \\
& & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & & \\
& & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & & \\
& & & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & & \\
& & & & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2 & \\
& & & & & & & & & & & & & & & & & & & & & & & \sigma_{AB}^2
\end{bmatrix}
$$

Estimating Covariance Parameters in the Mixed Model

Estimation is more difficult in the mixed model than in the general linear model. Not only do you have $\boldsymbol{\beta}$ as in the general linear model, but you have unknown parameters in $\boldsymbol{\gamma}$, $\mathbf{G}$, and $\mathbf{R}$ as well. Least squares is no longer the best method. *Generalized least squares* (GLS) is more appropriate, minimizing

$$(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'\mathbf{V}^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

However, it requires knowledge of $\mathbf{V}$ and, therefore, knowledge of $\mathbf{G}$ and $\mathbf{R}$. Lacking such information, one approach is to use *estimated* GLS, in which you insert some reasonable estimate for $\mathbf{V}$ into the minimization problem. The goal thus becomes finding a reasonable estimate of $\mathbf{G}$ and $\mathbf{R}$.

In many situations, the best approach is to use *likelihood-based* methods, exploiting the assumption that $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$ are normally distributed (Hartley and Rao 1967; Patterson and Thompson 1971; Harville 1977; Laird and Ware 1982; Jennrich and Schluchter 1986). PROC MIXED implements two likelihood-based methods: *maximum likelihood* (ML) and *restricted/residual maximum likelihood* (REML). A favorable theoretical property of ML and REML is that they accommodate data that are missing at random (Rubin 1976; Little 1995).

PROC MIXED constructs an objective function associated with ML or REML and maximizes it over all unknown parameters. Using calculus, it is possible to reduce this maximization problem to one over only the parameters in $\mathbf{G}$ and $\mathbf{R}$. The corresponding log-likelihood functions are as follows:

$$\begin{aligned}
\text{ML}: \quad & l(\mathbf{G},\mathbf{R}) = -\frac{1}{2}\log|\mathbf{V}| - \frac{1}{2}\mathbf{r}'\mathbf{V}^{-1}\mathbf{r} - \frac{n}{2}\log(2\pi) \\
\text{REML}: \quad & l_R(\mathbf{G},\mathbf{R}) = -\frac{1}{2}\log|\mathbf{V}| - \frac{1}{2}\log|\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}| - \frac{1}{2}\mathbf{r}'\mathbf{V}^{-1}\mathbf{r} - \frac{n-p}{2}\log(2\pi)\}
\end{aligned}$$

where $\mathbf{r} = \mathbf{y} - \mathbf{X}(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{y}$ and p is the rank of $\mathbf{X}$. PROC MIXED actually minimizes –2 times these functions by using a ridge-stabilized Newton-Raphson algorithm. Lindstrom and Bates (1988) provide reasons for preferring Newton-Raphson to the Expectation-Maximum (EM) algorithm (Dempster, Laird, and Rubin 1977; Laird, Lange, and Stram 1987), as well as analytical details for implementing a QR-decomposition approach to the problem. Wolfinger, Tobias, and Sall (1994) present the sweep-based algorithms that are implemented in PROC MIXED.

One advantage of using the Newton-Raphson algorithm is that the second derivative matrix of the objective function evaluated at the optima is available upon completion. Denoting this matrix $\mathbf{H}$, the asymptotic theory of maximum likelihood (see Serfling 1980) shows that $2\mathbf{H}^{-1}$ is an asymptotic variance-covariance matrix of the estimated parameters of $\mathbf{G}$ and $\mathbf{R}$. Thus, tests and confidence intervals based on asymptotic normality can be obtained. However, these can be unreliable in small samples, especially for parameters such as variance components that have sampling distributions that tend to be skewed to the right.

If a residual variance σ^2 is a part of your mixed model, it can usually be *profiled* out of the likelihood. This means solving analytically for the optimal σ^2 and plugging this expression back into the likelihood formula (see Wolfinger, Tobias, and Sall 1994). This reduces the number of optimization parameters by one and can improve convergence properties. PROC MIXED profiles the residual variance out of the log likelihood whenever it appears reasonable to do so. This includes the case when $\mathbf{R}$ equals $\sigma^2\mathbf{I}$ and when it has blocks with a compound symmetry, time series, or spatial structure. PROC MIXED does not profile the log likelihood when $\mathbf{R}$ has unstructured blocks, when you use the HOLD= or NOITER option in the PARMS statement, or when you use the NOPROFILE option in the PROC MIXED statement.

Instead of ML or REML, you can use the noniterative MIVQUE0 method to estimate $\mathbf{G}$ and $\mathbf{R}$ (Rao 1972; LaMotte 1973; Wolfinger, Tobias, and Sall 1994). In fact, by default PROC MIXED uses MIVQUE0

estimates as starting values for the ML and REML procedures. For variance component models, another estimation method involves equating Type 1, 2, or 3 expected mean squares to their observed values and solving the resulting system. However, Swallow and Monahan (1984) present simulation evidence favoring REML and ML over MIVQUE0 and other method-of-moment estimators.

Estimating Fixed and Random Effects in the Mixed Model

ML, REML, MIVQUE0, or Type1–Type3 provide estimates of $\mathbf{G}$ and $\mathbf{R}$, which are denoted $\widehat{\mathbf{G}}$ and $\widehat{\mathbf{R}}$, respectively. To obtain estimates of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, the standard method is to solve the *mixed model equations* (Henderson 1984):

$$\begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}+\widehat{\mathbf{G}}^{-1} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \end{bmatrix}$$

The solutions can also be written as

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{y}$$
$$\widehat{\boldsymbol{\gamma}} = \widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{V}}^{-1}(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}})$$

and have connections with empirical Bayes estimators (Laird and Ware 1982; Carlin and Louis 1996).

Note that the mixed model equations are extended normal equations and that the preceding expression assumes that $\widehat{\mathbf{G}}$ is nonsingular. For the extreme case where the eigenvalues of $\widehat{\mathbf{G}}$ are very large, $\widehat{\mathbf{G}}^{-1}$ contributes very little to the equations and $\widehat{\boldsymbol{\gamma}}$ is close to what it would be if $\boldsymbol{\gamma}$ actually contained fixed-effects parameters. On the other hand, when the eigenvalues of $\widehat{\mathbf{G}}$ are very small, $\widehat{\mathbf{G}}^{-1}$ dominates the equations and $\widehat{\boldsymbol{\gamma}}$ is close to 0. For intermediate cases, $\widehat{\mathbf{G}}^{-1}$ can be viewed as shrinking the fixed-effects estimates of $\boldsymbol{\gamma}$ toward 0 (Robinson 1991).

If $\widehat{\mathbf{G}}$ is singular, then the mixed model equations are modified (Henderson 1984) as follows:

$$\begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}\widehat{\mathbf{G}} \\ \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z}\widehat{\mathbf{G}}+\mathbf{G} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\tau}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \\ \widehat{\mathbf{G}}'\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{y} \end{bmatrix}$$

Denote the generalized inverses of the nonsingular $\widehat{\mathbf{G}}$ and singular $\widehat{\mathbf{G}}$ forms of the mixed model equations by $\mathbf{C}$ and $\mathbf{M}$, respectively. In the nonsingular case, the solution $\widehat{\boldsymbol{\gamma}}$ estimates the random effects directly, but in the singular case the estimates of random effects are achieved through a back-transformation $\widehat{\boldsymbol{\gamma}} = \widehat{\mathbf{G}}\widehat{\boldsymbol{\tau}}$ where $\widehat{\boldsymbol{\tau}}$ is the solution to the modified mixed model equations. Similarly, while in the nonsingular case $\mathbf{C}$ itself is the estimated covariance matrix for $(\widehat{\boldsymbol{\beta}}, \widehat{\boldsymbol{\gamma}})$, in the singular case the covariance estimate for $(\widehat{\boldsymbol{\beta}}, \widehat{\mathbf{G}}\widehat{\boldsymbol{\tau}})$ is given by $\mathbf{PMP}$ where

$$\mathbf{P} = \begin{bmatrix} \mathbf{I} & \\ & \widehat{\mathbf{G}} \end{bmatrix}$$

An example of when the singular form of the equations is necessary is when a variance component estimate falls on the boundary constraint of 0.

Model Selection

The previous section on estimation assumes the specification of a mixed model in terms of $\mathbf{X}$, $\mathbf{Z}$, $\mathbf{G}$, and $\mathbf{R}$. Even though $\mathbf{X}$ and $\mathbf{Z}$ have known elements, their specific form and construction are flexible, and several possibilities can present themselves for a particular data set. Likewise, several different covariance structures for $\mathbf{G}$ and $\mathbf{R}$ might be reasonable.

Space does not permit a thorough discussion of model selection, but a few brief comments and references are in order. First, subject matter considerations and objectives are of great importance when selecting a model; see Diggle (1988) and Lindsey (1993).

Second, when the data themselves are looked to for guidance, many of the graphical methods and diagnostics appropriate for the general linear model extend to the mixed model setting as well (Christensen, Pearson, and Johnson 1992).

Finally, a likelihood-based approach to the mixed model provides several statistical measures for model adequacy as well. The most common of these are the likelihood ratio test and Akaike's and Schwarz's criteria (Bozdogan 1987; Wolfinger 1993; Keselman et al. 1998, 1999).

Statistical Properties

If $\mathbf{G}$ and $\mathbf{R}$ are known, $\widehat{\boldsymbol{\beta}}$ is the *best linear unbiased estimator* (BLUE) of $\boldsymbol{\beta}$, and $\widehat{\boldsymbol{\gamma}}$ is the *best linear unbiased predictor* (BLUP) of $\boldsymbol{\gamma}$ (Searle 1971; Harville 1988, 1990; Robinson 1991; McLean, Sanders, and Stroup 1991). Here, "best" means minimum mean squared error. The covariance matrix of $(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}, \widehat{\boldsymbol{\gamma}} - \boldsymbol{\gamma})$ is

$$\mathbf{C} = \begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{R}^{-1}\mathbf{Z} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z} + \mathbf{G}^{-1} \end{bmatrix}^{-}$$

where $^{-}$ denotes a generalized inverse (see Searle 1971).

However, $\mathbf{G}$ and $\mathbf{R}$ are usually unknown and are estimated by using one of the aforementioned methods. These estimates, $\widehat{\mathbf{G}}$ and $\widehat{\mathbf{R}}$, are therefore simply substituted into the preceding expression to obtain

$$\widehat{\mathbf{C}} = \begin{bmatrix} \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} \\ \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{X} & \mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} + \widehat{\mathbf{G}}^{-1} \end{bmatrix}^{-}$$

as the approximate variance-covariance matrix of $(\widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}, \widehat{\boldsymbol{\gamma}} - \boldsymbol{\gamma})$. In this case, the BLUE and BLUP acronyms no longer apply, but the word *empirical* is often added to indicate such an approximation. The appropriate acronyms thus become EBLUE and EBLUP.

McLean and Sanders (1988) show that $\widehat{\mathbf{C}}$ can also be written as

$$\widehat{\mathbf{C}} = \begin{bmatrix} \widehat{\mathbf{C}}_{11} & \widehat{\mathbf{C}}_{21}' \\ \widehat{\mathbf{C}}_{21} & \widehat{\mathbf{C}}_{22} \end{bmatrix}$$

where

$$\begin{aligned} \widehat{\mathbf{C}}_{11} &= (\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-} \\ \widehat{\mathbf{C}}_{21} &= -\widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{V}}^{-1}\mathbf{X}\widehat{\mathbf{C}}_{11} \\ \widehat{\mathbf{C}}_{22} &= (\mathbf{Z}'\widehat{\mathbf{R}}^{-1}\mathbf{Z} + \widehat{\mathbf{G}}^{-1})^{-1} - \widehat{\mathbf{C}}_{21}\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{Z}\widehat{\mathbf{G}} \end{aligned}$$

Note that $\widehat{\mathbf{C}}_{11}$ is the familiar estimated generalized least squares formula for the variance-covariance matrix of $\widehat{\boldsymbol{\beta}}$.

As a cautionary note, $\widehat{\mathbf{C}}$ tends to underestimate the true sampling variability of $(\widehat{\boldsymbol{\beta}}\ \widehat{\boldsymbol{\gamma}})$ because no account is made for the uncertainty in estimating **G** and **R**. Although inflation factors have been proposed (Kackar and Harville 1984; Kass and Steffey 1989; Prasad and Rao 1990), they tend to be small for data sets that are fairly well balanced. PROC MIXED does not compute any inflation factors by default, but rather accounts for the downward bias by using the approximate t and F statistics described subsequently. The DDFM=KENWARDROGER or DDFM=KENWARDROGER2 option in the MODEL statement prompts PROC MIXED to compute a specific inflation factor along with Satterthwaite-based degrees of freedom.

Inference and Test Statistics

For inferences concerning the covariance parameters in your model, you can use likelihood-based statistics. One common likelihood-based statistic is the *Wald Z*, which is computed as the parameter estimate divided by its asymptotic standard error. The asymptotic standard errors are computed from the inverse of the second derivative matrix of the likelihood with respect to each of the covariance parameters. The Wald Z is valid for large samples, but it can be unreliable for small data sets and for parameters such as variance components, which are known to have a skewed or bounded sampling distribution.

A better alternative is the likelihood ratio χ^2 statistic. This statistic compares two covariance models, one a special case of the other. To compute it, you must run PROC MIXED twice, once for each of the two models, and then subtract the corresponding values of –2 times the log likelihoods. You can use either ML or REML to construct this statistic, which tests whether the full model is necessary beyond the reduced model.

As long as the reduced model does not occur on the boundary of the covariance parameter space, the χ^2 statistic computed in this fashion has a large-sample χ^2 distribution that is χ^2 with degrees of freedom equal to the difference in the number of covariance parameters between the two models. If the reduced model does occur on the boundary of the covariance parameter space, the asymptotic distribution becomes a mixture of χ^2 distributions (Self and Liang 1987). A common example of this is when you are testing that a variance component equals its lower boundary constraint of 0.

A final possibility for obtaining inferences concerning the covariance parameters is to simulate or resample data from your model and construct empirical sampling distributions of the parameters. The SAS macro language and the ODS system are useful tools in this regard.

F and *t* Tests for Fixed- and Random-Effects Parameters

For inferences concerning the fixed- and random-effects parameters in the mixed model, consider estimable linear combinations of the following form:

$$\mathbf{L}\begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{bmatrix}$$

The estimability requirement (Searle 1971) applies only to the $\boldsymbol{\beta}$ portion of **L**, because any linear combination of $\boldsymbol{\gamma}$ is estimable. Such a formulation in terms of a general **L** matrix encompasses a wide variety of common inferential procedures such as those employed with Type 1–Type 3 tests and LS-means. The CONTRAST and ESTIMATE statements in PROC MIXED enable you to specify your own **L** matrices. Typically, inference on fixed effects is the focus, and, in this case, the $\boldsymbol{\gamma}$ portion of **L** is assumed to contain all 0s.

Statistical inferences are obtained by testing the hypothesis

$$H : \mathbf{L}\begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\gamma} \end{bmatrix} = 0$$

or by constructing point and interval estimates.

When $\mathbf{L}$ consists of a single row, a general t statistic can be constructed as follows (see McLean and Sanders 1988; Stroup 1989a):

$$t = \frac{\mathbf{L}\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}}{\sqrt{\mathbf{L}\widehat{\mathbf{C}}\mathbf{L}'}}$$

Under the assumed normality of $\boldsymbol{\gamma}$ and $\boldsymbol{\epsilon}$, t has an exact t distribution only for data exhibiting certain types of balance and for some special unbalanced cases. In general, t is only approximately t-distributed, and its degrees of freedom must be estimated. See the DDFM= option for a description of the various degrees-of-freedom methods available in PROC MIXED.

With $\widehat{\nu}$ being the approximate degrees of freedom, the associated confidence interval is

$$\mathbf{L}\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix} \pm t_{\widehat{\nu},\alpha/2}\sqrt{\mathbf{L}\widehat{\mathbf{C}}\mathbf{L}'}$$

where $t_{\widehat{\nu},\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the $t_{\widehat{\nu}}$ distribution.

When the rank of $\mathbf{L}$ is greater than 1, PROC MIXED constructs the following general F statistic:

$$F = \frac{\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}' \mathbf{L}'(\mathbf{L}\widehat{\mathbf{C}}\mathbf{L}')^{-1}\mathbf{L}\begin{bmatrix} \widehat{\boldsymbol{\beta}} \\ \widehat{\boldsymbol{\gamma}} \end{bmatrix}}{r}$$

where $r = \mathrm{rank}(\mathbf{L}\widehat{\mathbf{C}}\mathbf{L}')$. Analogous to t, F in general has an approximate F distribution with r numerator degrees of freedom and $\widehat{\nu}$ denominator degrees of freedom.

The t and F statistics enable you to make inferences about your fixed effects, which account for the variance-covariance model you select. An alternative is the χ^2 statistic associated with the likelihood ratio test. This statistic compares two fixed-effects models, one a special case of the other. It is computed just as when comparing different covariance models, although you should use ML and not REML here because the penalty term associated with restricted likelihoods depends upon the fixed-effects specification.

F Tests With the ANOVAF Option

The ANOVAF option computes F tests by the following method in models with REPEATED statement and without RANDOM statement. Let $\mathbf{L}$ denote the matrix of estimable functions for the hypothesis $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$, where $\boldsymbol{\beta}$ are the fixed-effects parameters. Let $\mathbf{M} = \mathbf{L}'(\mathbf{L}\mathbf{L}')^{-}\mathbf{L}$, and suppose that $\widehat{\mathbf{C}}$ denotes the estimated variance-covariance matrix of $\widehat{\boldsymbol{\beta}}$ (see the section "Statistical Properties" for the construction of $\widehat{\mathbf{C}}$).

The ANOVAF F statistics are computed as

$$F_A = \widehat{\boldsymbol{\beta}}'\mathbf{L}'\left(\mathbf{L}\mathbf{L}'\right)^{-1}\mathbf{L}\widehat{\boldsymbol{\beta}}\Big/t_1 = \widehat{\boldsymbol{\beta}}'\mathbf{M}\widehat{\boldsymbol{\beta}}\Big/t_1$$

Notice that this is a modification of the usual F statistic where $(\mathbf{L}\widehat{\mathbf{C}}\mathbf{L}')^{-1}$ is replaced with $(\mathbf{L}\mathbf{L}')^{-1}$ and $\mathrm{rank}(\mathbf{L})$ is replaced with $t_1 = \mathrm{trace}(\mathbf{M}\widehat{\mathbf{C}})$; see, for example, Brunner, Domhof, and Langer (2002, Sec. 5.4). The p-values for this statistic are computed from either an F_{ν_1,ν_2} or an $F_{\nu_1,\infty}$ distribution. The respective

degrees of freedom are determined by the MIXED procedure as follows:

$$\nu_1 = \frac{t_1^2}{\text{trace}(\mathbf{M}\widehat{\mathbf{C}}\mathbf{M}\widehat{\mathbf{C}})}$$
$$\nu_2^* = \frac{2t_1^2}{\mathbf{g}'\mathbf{A}\mathbf{g}}$$
$$\nu_2 = \begin{cases} \max\{\min\{\nu_2^*, df_e\}, 1\} & \mathbf{g}'\mathbf{A}\mathbf{g} > 1\text{E}3 \times \text{MACEPS} \\ 1 & \text{otherwise} \end{cases}$$

The term $\mathbf{g}'\mathbf{A}\mathbf{g}$ in the term ν_2^* for the denominator degrees of freedom is based on approximating $\text{Var}[\text{trace}(\mathbf{M}\widehat{\mathbf{C}})]$ based on a first-order Taylor series about the true covariance parameters. This generalizes results in the appendix of Brunner, Dette, and Munk (1997) to a broader class of models. The vector $\mathbf{g} = [g_1, \ldots, g_q]$ contains the partial derivatives

$$\text{trace}\left(\mathbf{L}'\left(\mathbf{L}\mathbf{L}'\right)^{-1}\mathbf{L}\frac{\partial\widehat{\mathbf{C}}}{\partial\theta_i}\right)$$

and $\mathbf{A}$ is the asymptotic variance-covariance matrix of the covariance parameter estimates (ASYCOV option).

PROC MIXED reports ν_1 and ν_2 as "NumDF" and "DenDF" under the "ANOVA F" heading in the output. The corresponding p-values are denoted as "Pr > F(DDF)" for F_{ν_1,ν_2} and "Pr > F(infty)" for $F_{\nu_1,\infty}$, respectively.

P-values computed with the ANOVAF option can be identical to the nonparametric tests in Akritas, Arnold, and Brunner (1997) and in Brunner, Domhof, and Langer (2002), provided that the response data consist of properly created (and sorted) ranks and that the covariance parameters are estimated by MIVQUE0 in models with REPEATED statement and properly chosen SUBJECT= and/or GROUP= effects.

If you model an unstructured covariance matrix in a longitudinal model with one or more repeated factors, the ANOVAF results are identical to a multivariate MANOVA where degrees of freedom are corrected with the Greenhouse-Geisser adjustment (Greenhouse and Geisser 1959). For example, suppose that factor A has 2 levels and factor B has 4 levels. The following two sets of statements produce the same p-values:

```
proc mixed data=Mydata anovaf method=mivque0;
   class id A B;
   model score = A | B / chisq;
   repeated / type=un subject=id;
   ods select Tests3;
run;

proc transpose data=MyData out=tdata;
   by id;
   var score;
run;
proc glm data=tdata;
   model col: = / nouni;
   repeated A 2, B 4;
   ods output ModelANOVA=maov epsilons=eps;
run;
proc transpose data=eps(where=(substr(statistic,1,3)='Gre')) out=teps;
   var cvalue1;
```

```
run;

data aov; set maov;
   if (_n_ = 1) then merge teps;
   if (Source='A') then do;
      pFddf = ProbF;
      pFinf = 1 - probchi(df*Fvalue,df);
      output;
   end; else if (Source='B') then do;
      pFddf = ProbFGG;
      pFinf = 1 - probchi(df*col1*Fvalue,df*col1);
      output;
   end; else if (Source='A*B') then do;
      pfddF = ProbFGG;
      pFinf = 1 - probchi(df*col2*Fvalue,df*col2);
      output;
   end;
run;
proc print data=aov label noobs;
   label Source = 'Effect'
         df     = 'NumDF'
         Fvalue = 'Value'
         pFddf  = 'Pr > F(DDF)'
         pFinf  = 'Pr > F(infty)';
   var Source df Fvalue pFddf pFinf;
   format pF: pvalue6.;
run;
```

The PROC GLM code produces *p*-values that correspond to the ANOVAF *p*-values shown as Pr > F(DDF) in the MIXED output. The subsequent DATA step computes the *p*-values that correspond to Pr > F(infty) in the PROC MIXED output.

Parameterization of Mixed Models

Recall that a mixed model is of the form

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon}$$

where $\mathbf{y}$ represents univariate data, $\boldsymbol{\beta}$ is an unknown vector of fixed effects with known model matrix $\mathbf{X}$, $\boldsymbol{\gamma}$ is an unknown vector of random effects with known model matrix $\mathbf{Z}$, and $\boldsymbol{\epsilon}$ is an unknown random error vector.

PROC MIXED constructs a mixed model according to the specifications in the MODEL, RANDOM, and REPEATED statements. Each effect in the MODEL statement generates one or more columns in the model matrix $\mathbf{X}$, and each effect in the RANDOM statement generates one or more columns in the model matrix $\mathbf{Z}$. Effects in the REPEATED statement do not generate model matrices; they serve only to index observations within subjects. This section shows precisely how PROC MIXED builds $\mathbf{X}$ and $\mathbf{Z}$.

Intercept

By default, all models automatically include a column of 1s in **X** to estimate a fixed-effect intercept parameter μ. You can use the NOINT option in the MODEL statement to suppress this intercept. The NOINT option is useful when you are specifying a classification effect in the MODEL statement and you want the parameter estimate to be in terms of the mean response for each level of that effect, rather than in terms of a deviation from an overall mean.

By contrast, the intercept is not included by default in **Z**. To obtain a column of 1s in **Z**, you must specify in the RANDOM statement either the INTERCEPT effect or some effect that has only one level.

Regression Effects

Numeric variables, or polynomial terms involving them, can be included in the model as regression effects (covariates). The actual values of such terms are included as columns of the model matrices **X** and **Z**. You can use the bar operator with a regression effect to generate polynomial effects. For instance, X|X|X expands to X X*X X*X*X, a cubic model.

Main Effects

If a classification variable has *m* levels, PROC MIXED generates *m* columns in the model matrix for its main effect. Each column is an indicator variable for a given level. The order of the columns is the sort order of the values of their levels and can be controlled with the ORDER= option in the PROC MIXED statement. Table 84.20 is an example.

Table 84.20 Example of Main Effects

Data		**I**	A		B		
A	B	μ	A1	A2	B1	B2	B3
1	1	1	1	0	1	0	0
1	2	1	1	0	0	1	0
1	3	1	1	0	0	0	1
2	1	1	0	1	1	0	0
2	2	1	0	1	0	1	0
2	3	1	0	1	0	0	1

Typically, there are more columns for these effects than there are degrees of freedom for them. In other words, PROC MIXED uses an overparameterized model.

Interaction Effects

Often a model includes interaction (crossed) effects. With an interaction, PROC MIXED first reorders the terms to correspond to the order of the variables in the CLASS statement. Thus, B*A becomes A*B if A precedes B in the CLASS statement. Then, PROC MIXED generates columns for all combinations of levels that occur in the data. The order of the columns is such that the rightmost variables in the cross index faster than the leftmost variables (Table 84.21). Empty columns (that would contain all 0s) are not generated for **X**, but they are for **Z**.

Table 84.21 Example of Interaction Effects

Data		I	A		B			A*B					
A	B	μ	A1	A2	B1	B2	B3	A1B1	A1B2	A1B3	A2B1	A2B2	A2B3
1	1	1	1	0	1	0	0	1	0	0	0	0	0
1	2	1	1	0	0	1	0	0	1	0	0	0	0
1	3	1	1	0	0	0	1	0	0	1	0	0	0
2	1	1	0	1	1	0	0	0	0	0	1	0	0
2	2	1	0	1	0	1	0	0	0	0	0	1	0
2	3	1	0	1	0	0	1	0	0	0	0	0	1

In the preceding matrix, main-effects columns are not linearly independent of crossed-effects columns; in fact, the column space for the crossed effects contains the space of the main effect.

When your model contains many interaction effects, you might be able to code them more parsimoniously by using the bar operator (|). The bar operator generates all possible interaction effects. For example, A|B|C expands to A B A*B C A*C B*C A*B*C. To eliminate higher-order interaction effects, use the at sign (@) in conjunction with the bar operator. For instance, A|B|C|D @2 expands to A B A*B C A*C B*C D A*D B*D C*D.

Nested Effects

Nested effects are generated in the same manner as crossed effects. Hence, the design columns generated by the following two statements are the same (but the ordering of the columns is different):

```
model Y=A B(A);

model Y=A A*B;
```

The nesting operator in PROC MIXED is more a notational convenience than an operation distinct from crossing. Nested effects are typically characterized by the property that the nested variables never appear as main effects. The order of the variables within nesting parentheses is made to correspond to the order of these variables in the CLASS statement. The order of the columns is such that variables outside the parentheses index faster than those inside the parentheses, and the rightmost nested variables index faster than the leftmost variables (Table 84.22).

Table 84.22 Example of Nested Effects

Data		I	A		B(A)					
A	B	μ	A1	A2	B1A1	B2A1	B3A1	B1A2	B2A2	B3A2
1	1	1	1	0	1	0	0	0	0	0
1	2	1	1	0	0	1	0	0	0	0
1	3	1	1	0	0	0	1	0	0	0
2	1	1	0	1	0	0	0	1	0	0
2	2	1	0	1	0	0	0	0	1	0
2	3	1	0	1	0	0	0	0	0	1

Note that nested effects are often distinguished from interaction effects by the implied randomization structure of the design. That is, they usually indicate random effects within a fixed-effects framework. The fact that random effects can be modeled directly in the RANDOM statement might make the specification of nested effects in the MODEL statement unnecessary.

Continuous-Nesting-Class Effects

When a continuous variable nests with a classification variable, the design columns are constructed by multiplying the continuous values into the design columns for the class effect (Table 84.23).

Table 84.23 Example of Continuous-Nesting-Class Effects

Data		**I**	A		X(A)	
X	A	μ	A1	A2	X(A1)	X(A2)
21	1	1	1	0	21	0
24	1	1	1	0	24	0
22	1	1	1	0	22	0
28	2	1	0	1	0	28
19	2	1	0	1	0	19
23	2	1	0	1	0	23

This model estimates a separate slope for X within each level of A.

Continuous-by-Class Effects

Continuous-by-class effects generate the same design columns as continuous-nesting-class effects. The two models are made different by the presence of the continuous variable as a regressor by itself, as well as a contributor to a compound effect. Table 84.24 shows an example.

Table 84.24 Example of Continuous-by-Class Effects

Data		**I**	**X**	A		X*A	
X	A	μ	X	A1	A2	X*A1	X*A2
21	1	1	21	1	0	21	0
24	1	1	24	1	0	24	0
22	1	1	22	1	0	22	0
28	2	1	28	0	1	0	28
19	2	1	19	0	1	0	19
23	2	1	23	0	1	0	23

You can use continuous-by-class effects to test for homogeneity of slopes.

General Effects

An example that combines all the effects is X1*X2*A*B*C (D E). The continuous list comes first, followed by the crossed list, followed by the nested list in parentheses. You should be aware of the sequencing of parameters when you use the CONTRAST or ESTIMATE statement to compute some function of the parameter estimates.

Effects might be renamed by PROC MIXED to correspond to ordering rules. For example, B*A(E D) might be renamed A*B(D E) to satisfy the following:

- Classification variables that occur outside parentheses (crossed effects) are sorted in the order in which they appear in the CLASS statement.
- Variables within parentheses (nested effects) are sorted in the order in which they appear in the CLASS statement.

The sequencing of the parameters generated by an effect can be described by which variables have their levels indexed faster:

- Variables in the crossed list index faster than variables in the nested list.
- Within a crossed or nested list, variables to the right index faster than variables to the left.

For example, suppose a model includes four effects—A, B, C, and D—each having two levels, 1 and 2. Suppose the CLASS statement is as follows:

```
class A B C D;
```

Then the order of the parameters for the effect B*A(C D), which is renamed A*B (C D), is

$$\begin{array}{llll}
A_1B_1C_1D_1 \rightarrow & A_1B_2C_1D_1 \rightarrow & A_2B_1C_1D_1 \rightarrow & A_2B_2C_1D_1 \rightarrow \\
A_1B_1C_1D_2 \rightarrow & A_1B_2C_1D_2 \rightarrow & A_2B_1C_1D_2 \rightarrow & A_2B_2C_1D_2 \rightarrow \\
A_1B_1C_2D_1 \rightarrow & A_1B_2C_2D_1 \rightarrow & A_2B_1C_2D_1 \rightarrow & A_2B_2C_2D_1 \rightarrow \\
A_1B_1C_2D_2 \rightarrow & A_1B_2C_2D_2 \rightarrow & A_2B_1C_2D_2 \rightarrow & A_2B_2C_2D_2
\end{array}$$

Note that first the crossed effects B and A are sorted in the order in which they appear in the CLASS statement so that A precedes B in the parameter list. Then, for each combination of the nested effects in turn, combinations of A and B appear. The B effect moves fastest because it is rightmost in the cross list. Then A moves next fastest, and D moves next fastest. The C effect is the slowest since it is leftmost in the nested list.

When numeric levels are used, levels are sorted by their character format, which might not correspond to their numeric sort sequence (for example, noninteger levels). Therefore, it is advisable to include a desired format for numeric levels or to use the ORDER=INTERNAL option in the PROC MIXED statement to ensure that levels are sorted by their internal values.

Implications of the Non-Full-Rank Parameterization

For models with fixed effects involving classification variables, there are more design columns in $\mathbf{X}$ constructed than there are degrees of freedom for the effect. Thus, there are linear dependencies among the columns of $\mathbf{X}$. In this event, all of the parameters are not estimable; there is an infinite number of solutions to the mixed model equations. PROC MIXED uses a generalized inverse (a g_2-inverse, Pringle and Rayner 1971) to obtain values for the estimates (Searle 1971). The solution values are not displayed unless you specify the SOLUTION option in the MODEL statement. The solution has the characteristic that estimates are 0 whenever the design column for that parameter is a linear combination of previous columns. With this parameterization, hypothesis tests are constructed to test linear functions of the parameters that are estimable.

Some procedures (such as the CATMOD procedure) reparameterize models to full rank by using restrictions on the parameters. PROC GLM and PROC MIXED do not reparameterize, making the hypotheses that are commonly tested more understandable. See Goodnight (1978) for additional reasons for not reparameterizing.

Missing Level Combinations

PROC MIXED handles missing level combinations of classification variables similarly to the way PROC GLM does. Both procedures delete fixed-effects parameters corresponding to missing levels in order to preserve estimability. However, PROC MIXED does not delete missing level combinations for random-effects parameters because linear combinations of the random-effects parameters are always estimable. These conventions can affect the way you specify your CONTRAST and ESTIMATE coefficients.

Residuals and Influence Diagnostics

Residual Diagnostics

Consider a residual vector of the form $\widetilde{\mathbf{e}} = \mathbf{PY}$, where $\mathbf{P}$ is a projection matrix, possibly an oblique projector. A typical element $\widetilde{e}_i$ with variance v_i and estimated variance $\widehat{v}_i$ is said to be *standardized* as

$$\frac{\widetilde{e}_i}{\sqrt{\mathrm{Var}[\widetilde{e}_i]}} = \frac{\widetilde{e}_i}{\sqrt{v_i}}$$

and *studentized* as

$$\frac{\widetilde{e}_i}{\sqrt{\widehat{v}_i}}$$

External studentization uses an estimate of $\mathrm{Var}[\widetilde{e}_i]$ that does not involve the ith observation. Externally studentized residuals are often preferred over internally studentized residuals because they have well-known distributional properties in standard linear models for independent data.

Residuals that are scaled by the estimated variance of the response, i.e., $\widetilde{e}_i / \sqrt{\widehat{\mathrm{Var}}[Y_i]}$, are referred to as Pearson-type residuals.

Marginal and Conditional Residuals

The marginal and conditional means in the linear mixed model are $\mathrm{E}[\mathbf{Y}] = \mathbf{X}\boldsymbol{\beta}$ and $\mathrm{E}[\mathbf{Y}|\boldsymbol{\gamma}] = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma}$, respectively. Accordingly, the vector $\mathbf{r}_m$ of marginal residuals is defined as

$$\mathbf{r}_m = \mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}}$$

and the vector $\mathbf{r}_c$ of conditional residuals is

$$\mathbf{r}_c = \mathbf{Y} - \mathbf{X}\widehat{\boldsymbol{\beta}} - \mathbf{Z}\widehat{\boldsymbol{\gamma}} = \mathbf{r}_m - \mathbf{Z}\widehat{\boldsymbol{\gamma}}$$

Following Gregoire, Schabenberger, and Barrett (1995), let $\mathbf{Q} = \mathbf{X}(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}\mathbf{X}'$ and $\mathbf{K} = \mathbf{I} - \mathbf{Z}\widehat{\mathbf{G}}\mathbf{Z}'\widehat{\mathbf{V}}^{-1}$. Then

$$\widehat{\mathrm{Var}}[\mathbf{r}_m] = \widehat{\mathbf{V}} - \mathbf{Q}$$
$$\widehat{\mathrm{Var}}[\mathbf{r}_c] = \mathbf{K}(\widehat{\mathbf{V}} - \mathbf{Q})\mathbf{K}'$$

For an individual observation the raw, studentized, and Pearson-type residuals computed by the MIXED procedure are given in Table 84.25.

Table 84.25 Residual Types Computed by the MIXED Procedure

Type of Residual	Marginal	Conditional
Raw	$r_{mi} = Y_i - \mathbf{x}_i'\widehat{\boldsymbol{\beta}}$	$r_{ci} = r_{mi} - \mathbf{z}_i'\widehat{\boldsymbol{\gamma}}$
Studentized	$r_{mi}^{student} = \frac{r_{mi}}{\sqrt{\widehat{\mathrm{Var}}[r_{mi}]}}$	$r_{ci}^{student} = \frac{r_{ci}}{\sqrt{\widehat{\mathrm{Var}}[r_{ci}]}}$
Pearson	$r_{mi}^{pearson} = \frac{r_{mi}}{\sqrt{\widehat{\mathrm{Var}}[Y_i]}}$	$r_{ci}^{pearson} = \frac{r_{ci}}{\sqrt{\widehat{\mathrm{Var}}[Y_i \mid \boldsymbol{\gamma}]}}$

When the OUTPM= option is specified in addition to the RESIDUAL option in the MODEL statement, $r_{mi}^{student}$ and $r_{mi}^{pearson}$ are added to the data set as variables Resid, StudentResid, and PearsonResid, respectively. When the OUTP= option is specified, $r_{ci}^{student}$ and $r_{ci}^{pearson}$ are added to the data set. Raw residuals are part of the OUTPM= and OUTP= data sets without the RESIDUAL option.

Scaled Residuals

For correlated data, a set of scaled quantities can be defined through the Cholesky decomposition of the variance-covariance matrix. Since fitted residuals in linear models are rank-deficient, it is customary to draw on the variance-covariance matrix of the data. If $\mathrm{Var}[\mathbf{Y}] = \mathbf{V}$ and $\mathbf{C}'\mathbf{C} = \mathbf{V}$, then $\mathbf{C}'^{-1}\mathbf{Y}$ has uniform dispersion and its elements are uncorrelated.

Scaled residuals in a mixed model are meaningful for quantities based on the marginal distribution of the data. Let $\widehat{\mathbf{C}}$ denote the Cholesky root of $\widehat{\mathbf{V}}$, so that $\widehat{\mathbf{C}}'\widehat{\mathbf{C}} = \widehat{\mathbf{V}}$, and define

$$\mathbf{Y}_c = \widehat{\mathbf{C}}'^{-1}\mathbf{Y}$$
$$\mathbf{r}_{m(c)} = \widehat{\mathbf{C}}'^{-1}\mathbf{r}_m$$

By analogy with other scalings, the inverse Cholesky decomposition can also be applied to the residual vector, $\widehat{\mathbf{C}}'^{-1}\mathbf{r}_m$, although $\mathbf{V}$ is not the variance-covariance matrix of $\mathbf{r}_m$.

To diagnose whether the covariance structure of the model has been specified correctly can be difficult based on $\mathbf{Y}_c$, since the inverse Cholesky transformation affects the expected value of $\mathbf{Y}_c$. You can draw on $\mathbf{r}_{m(c)}$ as a vector of (approximately) uncorrelated data with constant mean.

When the OUTPM= option in the MODEL statement is specified in addition to the VCIRY option, $\mathbf{Y}_c$ is added as variable ScaledDep and $\mathbf{r}_{m(c)}$ is added as ScaledResid to the data set.

Influence Diagnostics

Basic Idea and Statistics

The general idea of quantifying the influence of one or more observations relies on computing parameter estimates based on all data points, removing the cases in question from the data, refitting the model, and computing statistics based on the change between full-data and reduced-data estimation. Influence statistics can be coarsely grouped by the aspect of estimation that is their primary target:

- overall measures compare changes in objective functions: (restricted) likelihood distance (Cook and Weisberg 1982, Ch. 5.2)
- influence on parameter estimates: Cook's D (Cook 1977, 1979), MDFFITS (Belsley, Kuh, and Welsch 1980, p. 32)
- influence on precision of estimates: CovRatio and CovTrace
- influence on fitted and predicted values: PRESS residual, PRESS statistic (Allen 1974), DFFITS (Belsley, Kuh, and Welsch 1980, p. 15)
- outlier properties: internally and externally studentized residuals, leverage

For linear models for uncorrelated data, it is not necessary to refit the model after removing a data point in order to measure the impact of an observation on the model. The change in fixed effect estimates, residuals, residual sums of squares, and the variance-covariance matrix of the fixed effects can be computed based on the fit to the full data alone. By contrast, in mixed models several important complications arise. Data points can affect not only the fixed effects but also the covariance parameter estimates on which the fixed-effects estimates depend. Furthermore, closed-form expressions for computing the change in important model quantities might not be available.

This section provides background material for the various influence diagnostics available with the MIXED procedure. See the section "Mixed Models Theory" on page 6843 for relevant expressions and definitions. The parameter vector $\boldsymbol{\theta}$ denotes all unknown parameters in the $\mathbf{R}$ and $\mathbf{G}$ matrix.

The observations whose influence is being ascertained are represented by the set U and referred to simply as "the observations in U." The estimate of a parameter vector, such as $\boldsymbol{\beta}$, obtained from all observations except those in the set U is denoted $\widehat{\boldsymbol{\beta}}_{(U)}$. In case of a matrix $\mathbf{A}$, the notation $\mathbf{A}_{(U)}$ represents the matrix with the rows in U removed; these rows are collected in $\mathbf{A}_U$. If $\mathbf{A}$ is symmetric, then notation $\mathbf{A}_{(U)}$ implies removal of rows and columns. The vector Y_U comprises the responses of the data points being removed, and $\mathbf{V}_{(U)}$ is the variance-covariance matrix of the remaining observations. When $k = 1$, lowercase notation emphasizes that single points are removed, such as $\mathbf{A}_{(u)}$.

Managing the Covariance Parameters

An important component of influence diagnostics in the mixed model is the estimated variance-covariance matrix $\mathbf{V} = \mathbf{ZGZ}' + \mathbf{R}$. To make the dependence on the vector of covariance parameters explicit, write it as $\mathbf{V}(\boldsymbol{\theta})$. If one parameter, σ^2, is profiled or factored out of $\mathbf{V}$, the remaining parameters are denoted as $\boldsymbol{\theta}^*$. Notice that in a model where $\mathbf{G}$ is diagonal and $\mathbf{R} = \sigma^2\mathbf{I}$, the parameter vector $\boldsymbol{\theta}^*$ contains the ratios of each variance component and σ^2 (see Wolfinger, Tobias, and Sall 1994). When ITER=0, two scenarios are distinguished:

1. If the residual variance is not profiled, either because the model does not contain a residual variance or because it is part of the Newton-Raphson iterations, then $\widehat{\boldsymbol{\theta}}_{(U)} \equiv \widehat{\boldsymbol{\theta}}$.

2. If the residual variance is profiled, then $\widehat{\boldsymbol{\theta}}^*_{(U)} \equiv \widehat{\boldsymbol{\theta}}^*$ and $\widehat{\sigma}^2_{(U)} \neq \widehat{\sigma}^2$. Influence statistics such as Cook's D and internally studentized residuals are based on $\mathbf{V}(\widehat{\boldsymbol{\theta}})$, whereas externally studentized residuals and the DFFITS statistic are based on $\mathbf{V}(\widehat{\boldsymbol{\theta}}_U) = \sigma^2_{(U)}\mathbf{V}(\widehat{\boldsymbol{\theta}}^*)$. In a random components model with uncorrelated errors, for example, the computation of $\mathbf{V}(\widehat{\boldsymbol{\theta}}_U)$ involves scaling of $\widehat{\mathbf{G}}$ and $\widehat{\mathbf{R}}$ by the full-data estimate $\widehat{\sigma}^2$ and multiplying the result with the reduced-data estimate $\widehat{\sigma}^2_{(U)}$.

Certain statistics, such as MDFFITS, CovRatio, and CovTrace, require an estimate of the variance of the fixed effects that is based on the reduced number of observations. For example, $\mathbf{V}(\widehat{\boldsymbol{\theta}}_U)$ is evaluated at the reduced-data parameter estimates but computed for the entire data set. The matrix $\mathbf{V}_{(U)}(\widehat{\boldsymbol{\theta}}_{(U)})$, on the other hand, has rows and columns corresponding to the points in U removed. The resulting matrix is evaluated at the delete-case estimates.

When influence analysis is iterative, the entire vector $\boldsymbol{\theta}$ is updated, whether the residual variance is profiled or not. The matrices to be distinguished here are $\mathbf{V}(\widehat{\boldsymbol{\theta}})$, $\mathbf{V}(\widehat{\boldsymbol{\theta}}_{(U)})$, and $\mathbf{V}_{(U)}(\widehat{\boldsymbol{\theta}}_{(U)})$, with unambiguous notation.

Predicted Values, PRESS Residual, and PRESS Statistic

An unconditional predicted value is $\widehat{y}_i = \mathbf{x}_i'\widehat{\boldsymbol{\beta}}$, where the vector $\mathbf{x}_i$ is the ith row of $\mathbf{X}$. The (raw) residual is given as $\widehat{\epsilon}_i = y_i - \widehat{y}_i$, and the PRESS *residual* is

$$\widehat{\epsilon}_{i(U)} = y_i - \mathbf{x}_i'\widehat{\boldsymbol{\beta}}_{(U)}$$

The PRESS *statistic* is the sum of the squared PRESS residuals,

$$PRESS = \sum_{i \in U} \widehat{\epsilon}^2_{i(U)}$$

where the sum is over the observations in U.

If EFFECT=, SIZE=, or KEEP= is not specified, PROC MIXED computes the PRESS residual for each observation selected through SELECT= (or all observations if SELECT= is not given). If EFFECT=, SIZE=, or KEEP= is specified, the procedure computes PRESS.

Leverage

For the general mixed model, leverage can be defined through the projection matrix that results from a transformation of the model with the inverse of the Cholesky decomposition of $\mathbf{V}$, or through an oblique projector. The MIXED procedure follows the latter path in the computation of influence diagnostics. The leverage value reported for the ith observation is the ith diagonal entry of the matrix

$$\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}$$

which is the weight of the observation in contributing to its own predicted value, $\mathbf{H} = d\widehat{\mathbf{Y}}/d\mathbf{Y}$.

While $\mathbf{H}$ is idempotent, it is generally not symmetric and thus not a projection matrix in the narrow sense.

The properties of these leverages are generalizations of the properties in models with diagonal variance-covariance matrices. For example, $\widehat{\mathbf{Y}} = \mathbf{HY}$, and in a model with intercept and $\mathbf{V} = \sigma^2\mathbf{I}$, the leverage values

$$h_{ii} = \mathbf{x}_i'(\mathbf{X}'\mathbf{X})^{-}\mathbf{x}_i$$

are $h_{ii}^l = 1/n \le h_{ii} \le 1 = h_{ii}^u$ and $\sum_{i=1}^n h_{ii} = \mathrm{rank}(\mathbf{X})$. The lower bound for h_{ii} is achieved in an intercept-only model, and the upper bound is achieved in a saturated model. The trace of $\mathbf{H}$ equals the rank of $\mathbf{X}$.

If ν_{ij} denotes the element in row i, column j of $\mathbf{V}^{-1}$, then for a model containing only an intercept the diagonal elements of $\mathbf{H}$ are

$$h_{ii} = \frac{\sum_{j=1}^n \nu_{ij}}{\sum_{i=1}^n \sum_{j=1}^n \nu_{ij}}$$

Because $\sum_{j=1}^n \nu_{ij}$ is a sum of elements in the ith row of the *inverse* variance-covariance matrix, h_{ii} can be negative, even if the correlations among data points are nonnegative. In case of a saturated model with $\mathbf{X} = \mathbf{I}$, $h_{ii} = 1.0$.

Internally and Externally Studentized Residuals

See the section "Residual Diagnostics" on page 6860 for the distinction between standardization, studentization, and scaling of residuals. Internally studentized marginal and conditional residuals are computed with the RESIDUAL option of the MODEL statement. The INFLUENCE option computes internally and externally studentized marginal residuals.

The computation of internally studentized residuals relies on the diagonal entries of $\mathbf{V}(\widehat{\boldsymbol{\theta}}) - \mathbf{Q}(\widehat{\boldsymbol{\theta}})$, where $\mathbf{Q}(\widehat{\boldsymbol{\theta}}) = \mathbf{X}(\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{X})^{-}\mathbf{X}'$. Externally studentized residuals require iterative influence analysis or a profiled residual variance. In the former case the studentization is based on $\mathbf{V}(\widehat{\boldsymbol{\theta}}_U)$; in the latter case it is based on $\sigma^2_{(U)}\mathbf{V}(\widehat{\boldsymbol{\theta}}^*)$.

Cook's D

Cook's D statistic is an invariant norm that measures the influence of observations in U on a vector of parameter estimates (Cook 1977). In case of the fixed-effects coefficients, let

$$\boldsymbol{\delta}_{(U)} = \widehat{\boldsymbol{\beta}} - \widehat{\boldsymbol{\beta}}_{(U)}$$

Then the MIXED procedure computes

$$D(\boldsymbol{\beta}) = \boldsymbol{\delta}'_{(U)}\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}]^{-}\boldsymbol{\delta}_{(U)}/\mathrm{rank}(\mathbf{X})$$

where $\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}]^{-}$ is the matrix that results from sweeping $(\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}})^{-1}\mathbf{X})^{-}$.

If $\mathbf{V}$ is known, Cook's D can be calibrated according to a chi-square distribution with degrees of freedom equal to the rank of $\mathbf{X}$ (Christensen, Pearson, and Johnson 1992). For estimated $\mathbf{V}$ the calibration can be carried out according to an $F(\mathrm{rank}(\mathbf{X}), n - \mathrm{rank}(\mathbf{X}))$ distribution. To interpret D on a familiar scale, Cook (1979) and Cook and Weisberg (1982, p. 116) refer to the 50th percentile of the reference distribution. If D is equal to that percentile, then removing the points in U moves the fixed-effects coefficient vector from the center of the confidence region to the 50% confidence ellipsoid (Myers 1990, p. 262).

In the case of iterative influence analysis, the MIXED procedure also computes a D-type statistic for the covariance parameters. If $\boldsymbol{\Gamma}$ is the asymptotic variance-covariance matrix of $\widehat{\boldsymbol{\theta}}$, then MIXED computes

$$D_{\boldsymbol{\theta}} = (\widehat{\boldsymbol{\theta}} - \widehat{\boldsymbol{\theta}}_{(U)})'\widehat{\boldsymbol{\Gamma}}^{-1}(\widehat{\boldsymbol{\theta}} - \widehat{\boldsymbol{\theta}}_{(U)})$$

DFFITS and MDFFITS

A DFFIT measures the change in predicted values due to removal of data points. If this change is standardized by the externally estimated standard error of the predicted value in the full data, the DFFITS statistic of Belsley, Kuh, and Welsch (1980, p. 15) results:

$$\mathrm{DFFITS}_i = (\widehat{y}_i - \widehat{y}_{i(u)})/\mathrm{ese}(\widehat{y}_i)$$

The MIXED procedure computes DFFITS when the EFFECT= or SIZE= modifier of the INFLUENCE option is not in effect. In general, an external estimate of the estimated standard error is used. When ITER > 0, the estimate is

$$\mathrm{ese}(\widehat{y}_i) = \sqrt{\mathbf{x}_i'(\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}}_{(u)})^{-}\mathbf{X})^{-1}\mathbf{x}_i}$$

When ITER=0 and σ^2 is profiled, then

$$\mathrm{ese}(\widehat{y}_i) = \widehat{\sigma}_{(u)}\sqrt{\mathbf{x}_i'(\mathbf{X}'\mathbf{V}(\widehat{\boldsymbol{\theta}}^*)^{-1}\mathbf{X})^{-}\mathbf{x}_i}$$

When the EFFECT=, SIZE=, or KEEP= modifier is specified, the MIXED procedure computes a multivariate version suitable for the deletion of multiple data points. The statistic, termed MDFFITS after the MDFFIT statistic of Belsley, Kuh, and Welsch (1980, p. 32), is closely related to Cook's D. Consider the case $\mathbf{V} = \sigma^2\mathbf{V}(\boldsymbol{\theta}^*)$ so that

$$\mathrm{Var}[\widehat{\boldsymbol{\beta}}] = \sigma^2(\mathbf{X}'\mathbf{V}(\boldsymbol{\theta}^*)^{-1}\mathbf{X})^{-}$$

and let $\widetilde{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}_{(U)}]$ be an estimate of $\mathrm{Var}[\widehat{\boldsymbol{\beta}}_{(U)}]$ that does not use the observations in U. The MDFFITS statistic is then computed as

$$\mathrm{MDFFITS}(\boldsymbol{\beta}) = \boldsymbol{\delta}_{(U)}'\widetilde{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}_{(U)}]^{-}\boldsymbol{\delta}_{(U)}/\mathrm{rank}(\mathbf{X})$$

If ITER=0 and σ^2 is profiled, then $\widetilde{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}_{(U)}]^{-}$ is obtained by sweeping

$$\widehat{\sigma}^2_{(U)}(\mathbf{X}_{(U)}'\mathbf{V}_{(U)}(\widehat{\boldsymbol{\theta}}^*)^{-}\mathbf{X}_{(U)})^{-}$$

The underlying idea is that if $\boldsymbol{\theta}^*$ were known, then

$$(\mathbf{X}_{(U)}'\mathbf{V}_{(U)}(\boldsymbol{\theta}^*)^{-1}\mathbf{X}_{(U)})^{-}$$

would be $\mathrm{Var}[\widehat{\boldsymbol{\beta}}]/\sigma^2$ in a generalized least squares regression with all but the data in U.

In the case of iterative influence analysis, $\widetilde{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}_{(U)}]$ is evaluated at $\widehat{\boldsymbol{\theta}}_{(U)}$. Furthermore, a MDFFITS-type statistic is then computed for the covariance parameters:

$$\mathrm{MDFFITS}(\boldsymbol{\theta}) = (\widehat{\boldsymbol{\theta}} - \widehat{\boldsymbol{\theta}}_{(U)})'\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\theta}}_{(U)}]^{-1}(\widehat{\boldsymbol{\theta}} - \widehat{\boldsymbol{\theta}}_{(U)})$$

Covariance Ratio and Trace

These statistics depend on the availability of an external estimate of $\mathbf{V}$, or at least of σ^2. Whereas Cook's D and MDFFITS measure the impact of data points on a vector of parameter estimates, the covariance-based statistics measure impact on their precision. Following Christensen, Pearson, and Johnson (1992), the MIXED procedure computes

$$\mathrm{CovTrace}(\boldsymbol{\beta}) = |\mathrm{trace}(\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}]^{-}\ \widetilde{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}_{(U)}]) - \mathrm{rank}(\mathbf{X})|$$

$$\mathrm{CovRatio}(\boldsymbol{\beta}) = \frac{\det_{ns}(\widetilde{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}_{(U)}])}{\det_{ns}(\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\beta}}])}$$

where $\det_{ns}(\mathbf{M})$ denotes the determinant of the nonsingular part of matrix $\mathbf{M}$.

In the case of iterative influence analysis these statistics are also computed for the covariance parameter estimates. If q denotes the rank of $\mathrm{Var}[\widehat{\boldsymbol{\theta}}]$, then

$$\mathrm{CovTrace}(\boldsymbol{\theta}) = |\mathrm{trace}(\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\theta}}]^{-}\ \widehat{\mathrm{Var}}[\widehat{\boldsymbol{\theta}}_{(U)}]) - q|$$

$$\mathrm{CovRatio}(\boldsymbol{\theta}) = \frac{\det_{ns}(\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\theta}}_{(U)}])}{\det_{ns}(\widehat{\mathrm{Var}}[\widehat{\boldsymbol{\theta}}])}$$

Likelihood Distances

The log-likelihood function l and restricted log-likelihood function l_R of the linear mixed model are given in the section "Estimating Covariance Parameters in the Mixed Model" on page 6849. Denote as $\boldsymbol{\psi}$ the collection of all parameters, i.e., the fixed effects $\boldsymbol{\beta}$ and the covariance parameters $\boldsymbol{\theta}$. Twice the difference between the (restricted) log-likelihood evaluated at the full-data estimates $\widehat{\boldsymbol{\psi}}$ and at the reduced-data estimates $\widehat{\boldsymbol{\psi}}_{(U)}$ is known as the (restricted) likelihood distance:

$$\mathrm{RLD}_{(U)} = 2\{l_R(\widehat{\boldsymbol{\psi}}) - l_R(\widehat{\boldsymbol{\psi}}_{(U)})\}$$

$$\mathrm{LD}_{(U)} = 2\{l(\widehat{\boldsymbol{\psi}}) - l(\widehat{\boldsymbol{\psi}}_{(U)})\}$$

Cook and Weisberg (1982, Ch. 5.2) refer to these differences as *likelihood distances*, Beckman, Nachtsheim, and Cook (1987) call the measures *likelihood displacements*. If the number of elements in $\boldsymbol{\psi}$ that are subject to updating following point removal is q, then likelihood displacements can be compared against cutoffs from a chi-square distribution with q degrees of freedom. Notice that this reference distribution does not depend on the number of observations removed from the analysis, but rather on the number of model parameters that are updated. The likelihood displacement gives twice the amount by which the log likelihood of the full data changes if one were to use an estimate based on fewer data points. It is thus a global, summary measure of the influence of the observations in U jointly on all parameters.

Unless METHOD=ML, the MIXED procedure computes the likelihood displacement based on the residual (=restricted) log likelihood, even if METHOD=MIVQUE0 or METHOD=TYPE1, TYPE2, or TYPE3.

Noniterative Update Formulas

Update formulas that do not require refitting of the model are available for the cases where $\mathbf{V} = \sigma^2\mathbf{I}$, $\mathbf{V}$ is known, or $\mathbf{V}^*$ is known. When ITER=0 and these update formulas can be invoked, the MIXED procedure uses the computational devices that are outlined in the following paragraphs. It is then assumed that the variance-covariance matrix of the fixed effects has the form $(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}$. When DDFM=KENWARDROGER or DDFM=KENWARDROGER2, this is not the case; the estimated variance-covariance matrix is then inflated to better represent the uncertainty in the estimated covariance parameters. Influence statistics when DDFM=KENWARDROGER should iteratively update the covariance parameters (ITER > 0). The dependence of $\mathbf{V}$ on $\boldsymbol{\theta}$ is suppressed in the sequel for brevity.

Updating the Fixed Effects Denote by $\mathbf{U}$ the $(n \times k)$ matrix that is assembled from k columns of the identity matrix. Each column of $\mathbf{U}$ corresponds to the removal of one data point. The point being targeted by the ith column of $\mathbf{U}$ corresponds to the row in which a 1 appears. Furthermore, define

$$\begin{aligned}
\boldsymbol{\Omega} &= (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-} \\
\mathbf{Q} &= \mathbf{X}\boldsymbol{\Omega}\mathbf{X}' \\
\mathbf{P} &= \mathbf{V}^{-1}(\mathbf{V}-\mathbf{Q})\mathbf{V}^{-1}
\end{aligned}$$

The change in the fixed-effects estimates following removal of the observations in U is

$$\widehat{\boldsymbol{\beta}} - \widehat{\boldsymbol{\beta}}_{(U)} = \boldsymbol{\Omega}\mathbf{X}'\mathbf{V}^{-1}\mathbf{U}(\mathbf{U}'\mathbf{P}\mathbf{U})^{-1}\mathbf{U}'\mathbf{V}^{-1}(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}})$$

Using results in Cook and Weisberg (1982, A2) you can further compute

$$\widetilde{\boldsymbol{\Omega}} = (\mathbf{X}'_{(U)}\mathbf{V}^{-1}_{(U)}\mathbf{X}_{(U)})^{-} = \boldsymbol{\Omega} + \boldsymbol{\Omega}\mathbf{X}'\mathbf{V}^{-1}\mathbf{U}(\mathbf{U}'\mathbf{P}\mathbf{U})^{-1}\mathbf{U}'\mathbf{V}^{-1}\mathbf{X}\boldsymbol{\Omega}$$

If $\mathbf{X}$ is $(n \times p)$ of rank $m < p$, then $\boldsymbol{\Omega}$ is deficient in rank and the MIXED procedure computes needed quantities in $\widetilde{\boldsymbol{\Omega}}$ by sweeping (Goodnight 1979). If the rank of the $(k \times k)$ matrix $\mathbf{U}'\mathbf{P}\mathbf{U}$ is less than k, the removal of the observations introduces a new singularity, whether $\mathbf{X}$ is of full rank or not. The solution vectors $\widehat{\boldsymbol{\beta}}$ and $\widehat{\boldsymbol{\beta}}_{(U)}$ then do not have the same expected values and should not be compared. When the MIXED procedure encounters this situation, influence diagnostics that depend on the choice of generalized inverse are not computed. The procedure also monitors the singularity criteria when sweeping the rows of $(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}$ and of $(\mathbf{X}'_{(U)}\mathbf{V}^{-1}_{(U)}\mathbf{X}_{(U)})^{-}$. If a new singularity is encountered or a former singularity disappears, no influence statistics are computed.

Residual Variance When σ^2 is profiled out of the marginal variance-covariance matrix, a closed-form estimate of σ^2 that is based on only the remaining observations can be computed provided $\mathbf{V}^* = \mathbf{V}(\widehat{\boldsymbol{\theta}}^*)$ is known. Hurtado (1993, Thm. 5.2) shows that

$$(n-q-r)\widehat{\sigma}^2_{(U)} = (n-q)\widehat{\sigma}^2 - \widehat{\boldsymbol{\epsilon}}'_U(\widehat{\sigma}^2\mathbf{U}'\mathbf{P}\mathbf{U})^{-1}\widehat{\boldsymbol{\epsilon}}_U$$

and $\widehat{\boldsymbol{\epsilon}}_U = \mathbf{U}'\mathbf{V}^{*-1}(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}})$. In the case of maximum likelihood estimation $q = 0$ and for REML estimation $q = \mathrm{rank}(\mathbf{X})$. The constant r equals the rank of $(\mathbf{U}'\mathbf{P}\mathbf{U})$ for REML estimation and the number of effective observations that are removed if METHOD=ML.

Likelihood Distances For noniterative methods the following computational devices are used to compute (restricted) likelihood distances provided that the residual variance σ^2 is profiled.

The log likelihood function $l(\widehat{\boldsymbol{\theta}})$ evaluated at the full-data and reduced-data estimates can be written as

$$\begin{aligned}
l(\widehat{\boldsymbol{\psi}}) &= -\frac{n}{2}\log(\widehat{\sigma}^2) - \frac{1}{2}\log|\mathbf{V}^*| - \frac{1}{2}(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}})'\mathbf{V}^{*-1}(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}})/\widehat{\sigma}^2 - \frac{n}{2}\log(2\pi) \\
l(\widehat{\boldsymbol{\psi}}_{(U)}) &= -\frac{n}{2}\log(\widehat{\sigma}^2_{(U)}) - \frac{1}{2}\log|\mathbf{V}^*| - \frac{1}{2}(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}}_{(U)})'\mathbf{V}^{*-1}(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}}_{(U)})/\widehat{\sigma}^2_{(U)} - \frac{n}{2}\log(2\pi)
\end{aligned}$$

Notice that $l(\widehat{\boldsymbol{\theta}}_{(U)})$ evaluates the log likelihood for n data points at the reduced-data estimates. It is not the log likelihood obtained by fitting the model to the reduced data. The likelihood distance is then

$$\mathrm{LD}_{(U)} = n\log\left\{\frac{\widehat{\sigma}^2_{(U)}}{\widehat{\sigma}^2}\right\} - n + \left(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}}_{(U)}\right)'\mathbf{V}^{*-1}\left(\mathbf{y}-\mathbf{X}\widehat{\boldsymbol{\beta}}_{(U)}\right)/\widehat{\sigma}^2_{(U)}$$

Expressions for $\mathrm{RLD}_{(U)}$ in noniterative influence analysis are derived along the same lines.

Default Output

The following sections describe the output PROC MIXED produces by default. This output is organized into various tables, and they are discussed in order of appearance.

Model Information

The "Model Information" table describes the model, some of the variables it involves, and the method used in fitting it. It also lists the method (profile, factor, parameter, or none) for handling the residual variance in the model. The *profile* method concentrates the residual variance out of the optimization problem, whereas the *parameter* method retains it as a parameter in the optimization. The *factor* method keeps the residual fixed, and *none* is displayed when a residual variance is not part of the model.

The "Model Information" table also has a row labeled Fixed Effects SE Method. This row describes the method used to compute the approximate standard errors for the fixed-effects parameter estimates and related functions of them. The two possibilities for this row are Model-Based, which is the default method, and Empirical, which results from using the EMPIRICAL option in the PROC MIXED statement.

The ODS name of the "Model Information" table is ModelInfo.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. You should check this information to make sure the data are correct. You can adjust the order of the CLASS variable levels with the ORDER= option in the PROC MIXED statement. The ODS name of the "Class Level Information" table is ClassLevels.

Dimensions

The "Dimensions" table lists the sizes of relevant matrices. This table can be useful in determining CPU time and memory requirements. The ODS name of the "Dimensions" table is Dimensions.

Number of Observations

The "Number of Observations" table shows the number of observations read from the data set and the number of observations used in fitting the model.

Iteration History

The "Iteration History" table describes the optimization of the residual log likelihood or log likelihood. The function to be minimized (the *objective function*) is $-2l$ for ML and $-2l_R$ for REML; the column name of the objective function in the "Iteration History" table is "-2 Log Like" for ML and "-2 Res Log Like" for REML. The minimization is performed by using a ridge-stabilized Newton-Raphson algorithm, and the rows of this table describe the iterations that this algorithm takes in order to minimize the objective function.

The Evaluations column of the "Iteration History" table tells how many times the objective function is evaluated during each iteration.

The Criterion column of the "Iteration History" table is, by default, a relative Hessian convergence quantity given by

$$\frac{\mathbf{g}_k' \mathbf{H}_k^{-1} \mathbf{g}_k}{|f_k|}$$

where f_k is the value of the objective function at iteration k, $\mathbf{g}_k$ is the gradient (first derivative) of f_k, and $\mathbf{H}_k$ is the Hessian (second derivative) of f_k. If $\mathbf{H}_k$ is singular, then PROC MIXED uses the following relative quantity:

$$\frac{\mathbf{g}_k' \mathbf{g}_k}{|f_k|}$$

To prevent the division by $|f_k|$, use the ABSOLUTE option in the PROC MIXED statement. To use a relative function or gradient criterion, use the CONVF or CONVG option, respectively.

The Hessian criterion is considered superior to function and gradient criteria because it measures orthogonality rather than lack of progress (Bates and Watts 1988). Provided the initial estimate is feasible and the maximum number of iterations is not exceeded, the Newton-Raphson algorithm is considered to have converged when the criterion is less than the tolerance specified with the CONVF, CONVG, or CONVH option in the PROC MIXED statement. The default tolerance is 1E–8. If convergence is not achieved, PROC MIXED displays the estimates of the parameters at the last iteration.

A convergence criterion that is missing indicates that a boundary constraint has been dropped; it is usually not a cause for concern.

If you specify the ITDETAILS option in the PROC MIXED statement, then the covariance parameter estimates at each iteration are included as additional columns in the "Iteration History" table.

The ODS name of the "Iteration History" table is IterHistory.

Convergence Status

The "Convergence Status" table informs about the status of the iterative estimation process at the end of the Newton-Raphson optimization. It appears as a message in the listing, and this message is repeated in the log. The ODS object ConvergenceStatus also contains several nonprinting columns that can be helpful in checking the success of the iterative process, in particular during batch processing or when analyzing BY groups. The Status variable takes on the value 0 for a successful convergence (even if the Hessian matrix might not be positive definite). The values 1 and 2 of the Status variable indicate lack of convergence and infeasible initial parameter values, respectively. The variables pdG and pdH can be used to check whether the **G** and **H** (Hessian) matrices are positive definite.

For models that are not fit iteratively, such as models without random effects or when the NOITER option is in effect, the "Convergence Status" is not produced.

Covariance Parameter Estimates

The "Covariance Parameter Estimates" table contains the estimates of the parameters in **G** and **R** (see the section "Estimating Covariance Parameters in the Mixed Model" on page 6849). Their values are labeled in the table along with Subject and Group information if applicable. The estimates are displayed in the Estimate column and are the results of one of the following estimation methods: REML, ML, MIVQUE0, SSCP, Type1, Type2, or Type3.

If you specify the RATIO option in the PROC MIXED statement, the Ratio column is added to the table listing the ratio of each parameter estimate to that of the residual variance.

Specifying the COVTEST option in the PROC MIXED statement produces the "Std Error," "Z Value," and "Pr Z" columns. The "Std Error" column contains the approximate standard errors of the covariance parameter estimates. These are the square roots of the diagonal elements of the observed inverse Fisher information matrix, which equals $2\mathbf{H}^{-1}$, where **H** is the Hessian matrix. The **H** matrix consists of the second derivatives of the objective function with respect to the covariance parameters; see Wolfinger, Tobias, and Sall (1994) for formulas. When you use the SCORING= option and PROC MIXED converges without stopping the scoring algorithm, PROC MIXED uses the expected Hessian matrix to compute the covariance matrix instead of the observed Hessian. The observed or expected inverse Fisher information matrix can be viewed as an asymptotic covariance matrix of the estimates.

The "Z Value" column is the estimate divided by its approximate standard error, and the "Pr Z" column is the one- or two-tailed area of the standard Gaussian density outside of the Z-value. The MIXED procedure computes one-sided p-values for the residual variance and for covariance parameters with a lower bound of 0. The procedure computes two-sided p-values otherwise. These statistics constitute Wald tests of the covariance parameters, and they are valid only asymptotically.

CAUTION: Wald tests can be unreliable in small samples.

The ODS name of the "Covariance Parameter Estimates" table is CovParms.

Fit Statistics

The "Fit Statistics" table provides some statistics about the estimated mixed model. Expressions for the –2 times the log likelihood are provided in the section "Estimating Covariance Parameters in the Mixed Model" on page 6849. If the log likelihood is an extremely large number, then PROC MIXED has deemed the estimated **V** matrix to be singular. In this case, all subsequent results should be viewed with caution.

In addition, the "Fit Statistics" table lists three information criteria: AIC, AICC, and BIC, all in smaller-is-better form. Expressions for these criteria are described under the IC option.

The ODS name of the "Model Fitting Information" table is FitStatistics.

Null Model Likelihood Ratio Test

If one covariance model is a submodel of another, you can carry out a likelihood ratio test for the significance of the more general model by computing –2 times the difference between their log likelihoods. Then compare this statistic to the χ^2 distribution with degrees of freedom equal to the difference in the number of parameters for the two models.

The results of this test are reported in the "Null Model Likelihood Ratio Test" table to determine whether it is necessary to model the covariance structure of the data at all. The χ^2 value is –2 times the log likelihood from the null model minus –2 times the log likelihood from the fitted model. The null model is the model that

contains only the fixed effects that are listed in the MODEL statement, and $\mathbf{R} = \sigma^2\mathbf{I}$ or $\mathbf{R} = \sigma^2 \sum_{k=1}^{q}(A_k)$ when TYPE=LIN(q) in the REPEATED statement. This statistic has an asymptotic χ^2 distribution with $q-1$ degrees of freedom, where q is the effective number of covariance parameters (those not estimated to be on a boundary constraint). The Pr > ChiSq column contains the upper-tail area from this distribution. This p-value can be used to assess the significance of the model fit.

This test is not produced for cases where the null hypothesis lies on the boundary of the parameter space, which is typically true of variance component models. This is because the standard asymptotic theory does not apply in such cases (Self and Liang 1987, Case 5).

If you specify a PARMS statement, PROC MIXED constructs a likelihood ratio test between the best model from the grid search and the final fitted model and reports the results in the "Parameter Search" table.

The ODS name of the "Null Model Likelihood Ratio Test" table is LRT.

Type 3 Tests of Fixed Effects

The "Type 3 Tests of Fixed Effects" table contains hypothesis tests for the significance of each of the fixed effects—that is, those effects that you specify in the MODEL statement. By default, PROC MIXED performs these tests by first constructing a Type 3 $\mathbf{L}$ matrix (see Chapter 16, "The Four Types of Estimable Functions") for each effect. This $\mathbf{L}$ matrix is then used to compute the following F statistic,

$$F = \frac{\widehat{\boldsymbol{\beta}}'\mathbf{L}'[\mathbf{L}(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}\mathbf{L}']^{-}\mathbf{L}\widehat{\boldsymbol{\beta}}}{r}$$

where $r = \mathrm{rank}(\mathbf{L}(\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X})^{-}\mathbf{L}')$. A p-value for the test is computed as the tail area beyond this statistic from an F distribution with NDF and DDF degrees of freedom. The numerator degrees of freedom (NDF) are the row rank of $\mathbf{L}$, and the denominator degrees of freedom (DDF) are computed by using one of the methods described under the DDFM= option. Small p-values (typically less than 0.05 or 0.01) indicate a significant effect.

You can use the HTYPE= option in the MODEL statement to obtain tables of Type 1 (sequential) tests and Type 2 (adjusted) tests in addition to or instead of the table of Type 3 (partial) tests.

You can use the CHISQ option in the MODEL statement to obtain Wald χ^2 tests of the fixed effects. These are carried out by using the numerator of the F statistic and comparing it with the χ^2 distribution with NDF degrees of freedom. It is more liberal than the F test because it effectively assumes infinite denominator degrees of freedom.

The ODS names of the "Type 1 Tests of Fixed Effects" through the "Type 3 Tests of Fixed Effects" tables are Tests1 through Tests3, respectively.

ODS Table Names

Each table created by PROC MIXED has a name associated with it, and you must use this name to reference the table when using ODS statements. These names are listed in Table 84.26.

Table 84.26 ODS Tables Produced by PROC MIXED

Table Name	Description	Required Statement / Option
AccRates	Acceptance rates for posterior sampling	PRIOR
AsyCorr	Asymptotic correlation matrix of covariance parameters	PROC MIXED / ASYCORR
AsyCov	Asymptotic covariance matrix of covariance parameters	PROC MIXED / ASYCOV
Base	Base densities used for posterior sampling	PRIOR
Bound	Computed bound for posterior rejection sampling	PRIOR
CholG	Cholesky root of the estimated **G** matrix	RANDOM / GC
CholR	Cholesky root of blocks of the estimated **R** matrix	REPEATED / RC
CholV	Cholesky root of blocks of the estimated **V** matrix	RANDOM / VC
ClassLevels	Level information from the CLASS statement	Default output
Coef	**L** matrix coefficients	E option in MODEL, CONTRAST, ESTIMATE, or LSMEANS
Contrasts	Results from the CONTRAST statements	CONTRAST
ConvergenceStatus	Convergence status	Default
CorrB	Approximate correlation matrix of fixed-effects parameter estimates	MODEL / CORRB
CovB	Approximate covariance matrix of fixed-effects parameter estimates	MODEL / COVB
CovParms	Estimated covariance parameters	Default output
Diffs	Differences of LS-means	LSMEANS / DIFF or ADJUST
Dimensions	Dimensions of the model	Default output
Estimates	Results from ESTIMATE statements	ESTIMATE
FitStatistics	Fit statistics	Default
G	Estimated **G** matrix	RANDOM / G
GCorr	Correlation matrix from the estimated **G** matrix	RANDOM / GCORR
HLM1	Type 1 Hotelling-Lawley-McKeon tests of fixed effects	MODEL / HTYPE=1 and REPEATED / HLM TYPE=UN

Table 84.26 *continued*

Table Name	Description	Required Statement / Option
HLM2	Type 2 Hotelling-Lawley-McKeon tests of fixed effects	MODEL / HTYPE=2 and REPEATED / HLM TYPE=UN
HLM3	Type 3 Hotelling-Lawley-McKeon tests of fixed effects	REPEATED / HLM TYPE=UN
HLPS1	Type 1 Hotelling-Lawley-Pillai-Samson tests of fixed effects	MODEL / HTYPE=1 and REPEATED / HLPS TYPE=UN
HLPS2	Type 2 Hotelling-Lawley-Pillai-Samson tests of fixed effects	MODEL / HTYPE=1 and REPEATED / HLPS TYPE=UN
HLPS3	Type 3 Hotelling-Lawley-Pillai-Samson tests of fixed effects	REPEATED / HLPS TYPE=UN
Influence	Influence diagnostics	MODEL / INFLUENCE
InfoCrit	Information criteria	PROC MIXED / IC
InvCholG	Inverse Cholesky root of the estimated **G** matrix	RANDOM / GCI
InvCholR	Inverse Cholesky root of blocks of the estimated **R** matrix	REPEATED / RCI
InvCholV	Inverse Cholesky root of blocks of the estimated **V** matrix	RANDOM / VCI
InvCovB	Inverse of approximate covariance matrix of fixed-effects parameter estimates	MODEL / COVBI
InvG	Inverse of the estimated **G** matrix	RANDOM / GI
InvR	Inverse of blocks of the estimated **R** matrix	REPEATED / RI
InvV	Inverse of blocks of the estimated **V** matrix	RANDOM / VI
IterHistory	Iteration history	Default output
LComponents	Single-degree-of-freedom estimates that correspond to rows of the **L** matrix for fixed effects	MODEL / LCOMPONENTS
LRT	Likelihood ratio test	Default output
LSMeans	LS-means	LSMEANS
MMEq	Mixed model equations	PROC MIXED / MMEQ
MMEqSol	Mixed model equations solution	PROC MIXED / MMEQSOL
ModelInfo	Model information	Default output
NObs	Number of observations read and used	Default output
ParmSearch	Parameter search values	PARMS
Posterior	Posterior sampling information	PRIOR
Ranks	Ranks of design matrices **X** and (**XZ**)	PROC MIXED / RANKS
R	Blocks of the estimated **R** matrix	REPEATED / R

Table 84.26 *continued*

Table Name	Description	Required Statement / Option
RCorr	Correlation matrix from blocks of the estimated **R** matrix	REPEATED / RCORR
Search	Posterior density search table	PRIOR / PSEARCH
SimDetails	Details of difference quantile simulation	LSMEANS / ADJUST=SIMULATE(REPORT)
SimResults	Evaluation of difference quantile simulation	LSMEANS / ADJUST=SIMULATE(REPORT)
Slices	Tests of LS-means slices	LSMEANS / SLICE=
SolutionF	Fixed-effects solution vector	MODEL / S
SolutionR	Random-effects solution vector	RANDOM / S
Tests1	Type 1 tests of fixed effects	MODEL / HTYPE=1
Tests2	Type 2 tests of fixed effects	MODEL / HTYPE=2
Tests3	Type 3 tests of fixed effects	Default output
Type1	Type 1 analysis of variance	PROC MIXED / METHOD=TYPE1
Type2	Type 2 analysis of variance	PROC MIXED / METHOD=TYPE2
Type3	Type 3 analysis of variance	PROC MIXED / METHOD=TYPE3
Trans	Transformation of covariance parameters	PRIOR / PTRANS
V	Blocks of the estimated **V** matrix	RANDOM / V
VCorr	Correlation matrix from blocks of the estimated **V** matrix	RANDOM / VCORR

In Table 84.26, "Coef" refers to multiple tables produced by the E, E1, E2, or E3 option in the MODEL statement and the E option in the CONTRAST, ESTIMATE, and LSMEANS statements. You can create one large data set of these tables with a statement similar to the following:

```
ods output Coef=c;
```

To create separate data sets, use the following statement:

```
ods output Coef(match_all)=c;
```

Here the resulting data sets are named C, C1, C2, etc. The same principles apply to data sets created from the R, CholR, InvCholR, RCorr, InvR, V, CholV, InvCholV, VCorr, and InvV tables.

In Table 84.26, the following changes have occurred from SAS 6. The Predicted, PredMeans, and Sample tables from SAS 6 no longer exist and have been replaced by output data sets; see descriptions of the MODEL statement options OUTP= and OUTPM= and the PRIOR statement option OUT= for more details. The ML and REML tables from SAS 6 have been replaced by the IterHistory table. The Tests, HLM, and HLPS tables from SAS 6 have been renamed Tests3, HLM3, and HLPS3, respectively.

Table 84.27 lists the variable names associated with the data sets created when you use the ODS OUTPUT option in conjunction with the preceding tables. In Table 84.27, *n* is used to denote a generic number that depends on the particular data set and model you select, and it can assume a different value each time it is used (even within the same table). The phrase *model specific* appears in rows of the affected tables to indicate that columns in these tables depend on the variables you specify in the model.

CAUTION: There is a danger of name collisions with the variables in the *model specific* tables in Table 84.27 and variables in your input data set. You should avoid using input variables with the same names as the variables in these tables.

Table 84.27 Variable Names for the ODS Tables Produced in PROC MIXED

Table Name	Variables
AsyCorr	Row, CovParm, CovP1–CovP*n*
AsyCov	Row, CovParm, CovP1–CovP*n*
Base	Type, Parm1–Parm*n*
Bound	Technique, Converge, Iterations, Evaluations, LogBound, CovP1–CovP*n*, TCovP1–TCovP*n*
CholG	*Model specific*, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Row, Col1–Col*n*
CholR	Index, Row, Col1–Col*n*
CholV	Index, Row, Col1–Col*n*
ClassLevels	Class, Levels, Values
Coef	*Model specific*, LMatrix, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Row1–Row*n*
Contrasts	Label, NumDF, DenDF, ChiSquare, FValue, ProbChiSq, ProbF
CorrB	*Model specific*, Effect, Row, Col1–Col*n*
CovB	*Model specific*, Effect, Row, Col1–Col*n*
CovParms	CovParm, Subject, Group, Estimate, StandardError, ZValue, ProbZ, Alpha, Lower, Upper
Diffs	*Model specific*, Effect, Margins, ByLevel, AT variables, Diff, StandardError, DF, tValue, Tails, Probt, Adjustment, Adjp, Alpha, Lower, Upper, AdjLow, AdjUpp
Dimensions	Descr, Value
Estimates	Label, Estimate, StandardError, DF, tValue, Tails, Probt, Alpha, Lower, Upper
FitStatistics	Descr, Value
G	*Model specific*, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Row, Col1–Col*n*
GCorr	*Model specific*, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Row, Col1–Col*n*
HLM1	Effect, NumDF, DenDF, FValue, ProbF
HLM2	Effect, NumDF, DenDF, FValue, ProbF
HLM3	Effect, NumDF, DenDF, FValue, ProbF
HLPS1	Effect, NumDF, DenDF, FValue, ProbF
HLPS2	Effect, NumDF, DenDF, FValue, ProbF
HLPS3	Effect, NumDF, DenDF, FValue, ProbF
Influence	*Dependent on option modifiers*, Effect, Tuple, Obs1–Obs*k*, Level, Iter, Index, Predicted, Residual, Leverage, PressRes, PRESS, Student, RMSE, RStudent, CookD, DFFITS, MDFFITS, CovRatio, CovTrace, CookDCP, MDFFITSCP, CovRatioCP, CovTraceCP, LD, RLD, Parm1–Parm*p*, CovP1–CovP*q*, Notes
InfoCrit	Neg2LogLike, Parms, AIC, AICC, HQIC, BIC, CAIC

Table 84.27 *continued*

Table Name	Variables
InvCholG	*Model specific*, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Row, Col1–Col*n*
InvCholR	Index, Row, Col1–Col*n*
InvCholV	Index, Row, Col1–Col*n*
InvCovB	*Model specific*, Effect, Row, Col1–Col*n*
InvG	*Model specific*, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Row, Col1–Col*n*
InvR	Index, Row, Col1–Col*n*
InvV	Index, Row, Col1–Col*n*
IterHistory	CovP1–CovP*n*, Iteration, Evaluations, M2ResLogLike, M2LogLike, Criterion
LComponents	Effect, TestType, LIndex, Estimate, StdErr, DF, tValue, Probt
LRT	DF, ChiSquare, ProbChiSq
LSMeans	*Model specific*, Effect, Margins, ByLevel, AT variables, Estimate, StandardError, DF, tValue, Probt, Alpha, Lower, Upper, Cov1–Cov*n*, Corr1–Corr*n*
MMEq	*Model specific*, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Row, Col1–Col*n*
MMEqSol	*Model specific*, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Row, Col1–Col*n*
ModelInfo	Descr, Value
Nobs	Label, N, NObsRead, NObsUsed, SumFreqsRead, SumFreqsUsed
ParmSearch	CovP1–CovP*n*, Var, ResLogLike, M2ResLogLike2, LogLike, M2LogLike, LogDetH
Posterior	Descr, Value
R	Index, Row, Col1–Col*n*
RCorr	Index, Row, Col1–Col*n*
Search	Parm, TCovP1–TCovP*n*, Posterior
Slices	*Model specific*, Effect, Margins, ByLevel, AT variables, NumDF, DenDF, FValue, ProbF
SolutionF	*Model specific*, Effect, Estimate, StandardError, DF, tValue, Probt, Alpha, Lower, Upper
SolutionR	*Model specific*, Effect, Subject, Sub1–Sub*n*, Group, Group1–Group*n*, Estimate, StdErrPred, DF, tValue, Probt, Alpha, Lower, Upper
Tests1	Effect, NumDF, DenDF, ChiSquare, FValue, ProbChiSq, ProbF
Tests2	Effect, NumDF, DenDF, ChiSquare, FValue, ProbChiSq, ProbF
Tests3	Effect, NumDF, DenDF, ChiSquare, FValue, ProbChiSq, ProbF
Type1	Source, DF, SS, MS, EMS, ErrorTerm, ErrorDF, FValue, ProbF
Type2	Source, DF, SS, MS, EMS, ErrorTerm, ErrorDF, FValue, ProbF
Type3	Source, DF, SS, MS, EMS, ErrorTerm, ErrorDF, FValue, ProbF
Trans	Prior, TCovP, CovP1–CovP*n*
V	Index, Row, Col1–Col*n*
VCorr	Index, Row, Col1–Col*n*

Some of the variables listed in Table 84.27 are created only when you specify certain *options* in the relevant PROC MIXED statements.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

Some graphs are produced by default; other graphs are produced by using statements and options.

ODS Graph Names

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC MIXED generates are listed in Table 84.28, along with the required statements and options.

Table 84.28 Graphs Produced by PROC MIXED

ODS Graph Name	Plot Description	Statement or Option
Boxplot	Box plots	PLOTS=BOXPLOT
CovRatioPlot	CovRatio statistics for fixed effects or covariance parameters	PLOTS=INFLUENCESTATPANEL(UNPACK) and MODEL / INFLUENCE
CooksDPlot	Cook's *D* for fixed effects or covariance parameters	PLOTS=INFLUENCESTATPANEL(UNPACK) and MODEL / INFLUENCE
DistancePlot	Likelihood or restricted likelihood distance	MODEL / INFLUENCE
InfluenceEstPlot	Panel of deletion estimates	MODEL / INFLUENCE(EST) or PLOTS=INFLUENCEESTPLOT and MODEL / INFLUENCE
InfluenceEstPlot	Parameter estimates after removing observation or sets of observations	PLOTS=INFLUENCEESTPLOT(UNPACK) and MODEL / INFLUENCE
InfluenceStatPanel	Panel of influence statistics	MODEL / INFLUENCE
PearsonBoxPlot	Box plot of Pearson residuals	PLOTS=PEARSONPANEL(UNPACK BOX)
PearsonByPredicted	Pearson residuals vs. predicted	PLOTS=PEARSONPANEL(UNPACK)
PearsonHistogram	Histogram of Pearson residuals	PLOTS=PEARSONPANEL(UNPACK)

Table 84.28 *continued*

ODS Graph Name	Plot Description	Statement or Option
PearsonPanel	Panel of Pearson residuals	MODEL / RESIDUAL
PearsonQQplot	*Q-Q* plot of Pearson residuals	PLOTS=PEARSONPANEL(UNPACK)
PressPlot	Plot of PRESS residuals or PRESS statistic	PLOTS=PRESS and MODEL / INFLUENCE
ResidualBoxplot	Box plot of (raw) residuals	PLOTS=RESIDUALPANEL(UNPACK BOX)
ResidualByPredicted	Residuals vs. predicted	PLOTS=RESIDUALPANEL(UNPACK)
ResidualHistogram	Histogram of raw residuals	PLOTS=RESIDUALPANEL(UNPACK)
ResidualPanel	Panel of (raw) residuals	MODEL / RESIDUAL
ResidualQQplot	*Q-Q* plot of raw residuals	PLOTS=RESIDUALPANEL(UNPACK)
ScaledBoxplot	Box plot of scaled residuals	PLOTS=VCIRYPANEL(UNPACK BOX)
ScaledByPredicted	Scaled residuals vs. predicted	PLOTS=VCIRYPANEL(UNPACK)
ScaledHistogram	Histogram of scaled residuals	PLOTS=VCIRYPANEL(UNPACK)
ScaledQQplot	*Q-Q* plot of scaled residuals	PLOTS=VCIRYPANEL(UNPACK)
StudentBoxplot	Box plot of studentized residuals	PLOTS=STUDENTPANEL(UNPACK BOX)
StudentByPredicted	Studentized residuals vs. predicted	PLOTS=STUDENTPANEL(UNPACK)
StudentHistogram	Histogram of studentized residuals	PLOTS=STUDENTPANEL(UNPACK)
StudentPanel	Panel of studentized residuals	MODEL / RESIDUAL
StudentQQplot	*Q-Q* plot of studentized residuals	PLOTS=STUDENTPANEL(UNPACK)
VCIRYPanel	Panel of scaled residuals	MODEL / VCIRY

When ODS Graphics is enabled, the LSMESTIMATE and SLICE statements can produce plots that are associated with their analyses. For information about these plots, see the sections "LSMESTIMATE Statement" on page 513 and "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

Residual Plots

The MIXED procedure can generate panels of residual diagnostics. Each panel consists of a plot of residuals versus predicted values, a histogram with normal density overlaid, a Q-Q plot, and summary residual and fit statistics (Figure 84.15). The plots are produced even if the OUTP= and OUTPM= options in the MODEL statement are not specified. Residual panels can be generated for marginal and conditional raw, studentized, and Pearson residuals as well as for scaled residuals (see the section "Residual Diagnostics" on page 6860).

Recall the example in the section "Getting Started: MIXED Procedure" on page 6766. The following statements generate several 2×2 panels of residual graphs:

```
ods graphics on;
proc mixed data=heights plots=studentpanel(marginal conditional);
   class Family Gender;
   model Height = Gender / residual;
   random Family Family*Gender;
run;
ods graphics off;
```

The graphs are created when ODS Graphics is enabled. The panel of the studentized marginal residuals is shown in Figure 84.15, and the panel of the studentized conditional residuals is shown in Figure 84.16.

Figure 84.15 Panel of the Studentized (Marginal) Residuals

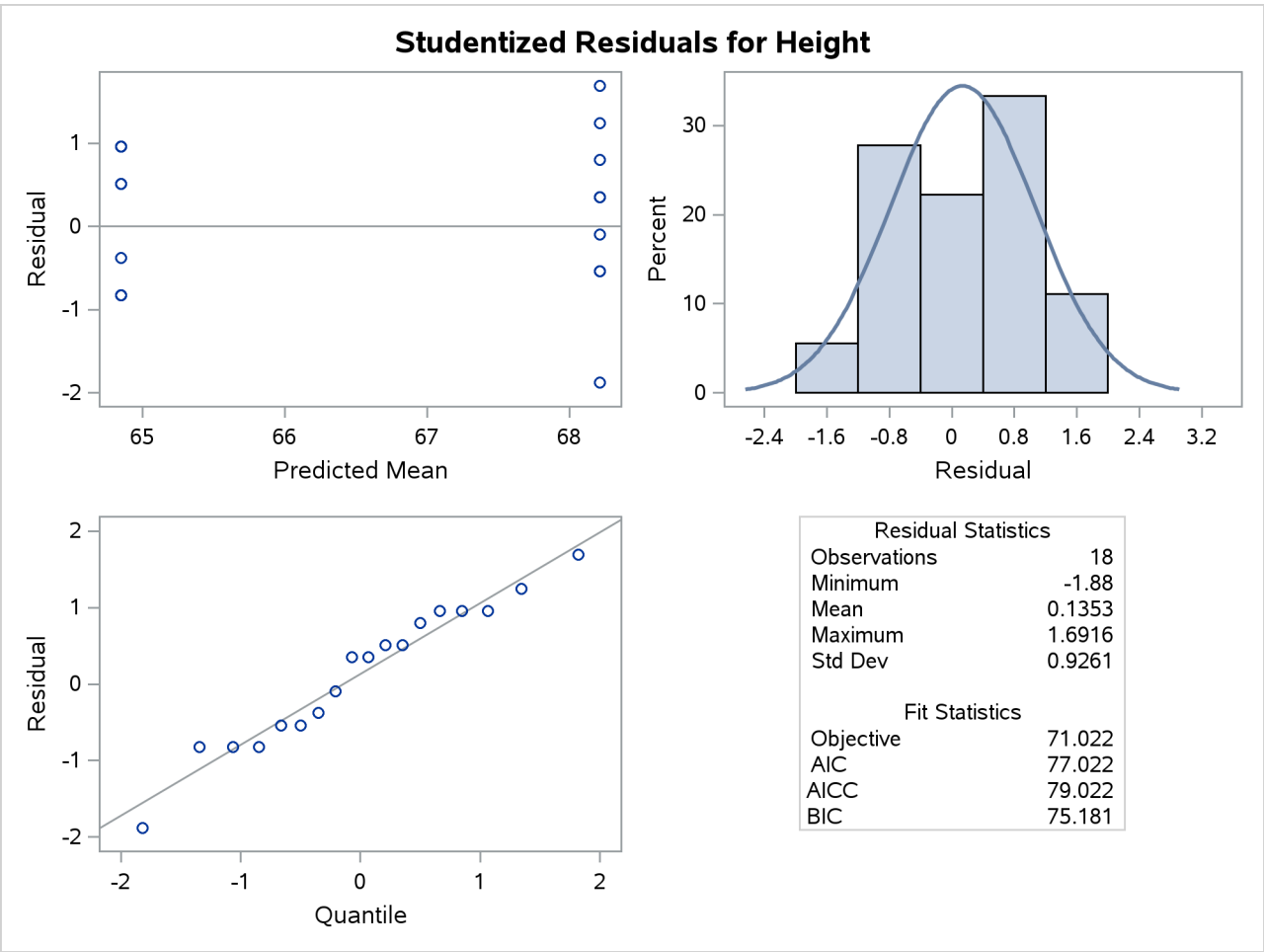

Since the fixed-effects part of the model comprises only an intercept and the gender effect, the marginal mean takes on only two values, one for each gender. The "Residual Statistics" inset in the lower-right corner provides descriptive statistics for the set of residuals that is displayed. Note that residuals in a mixed model do not necessarily sum to zero, even if the model contains an intercept.

Figure 84.16 Panel of the Conditional Studentized Residuals

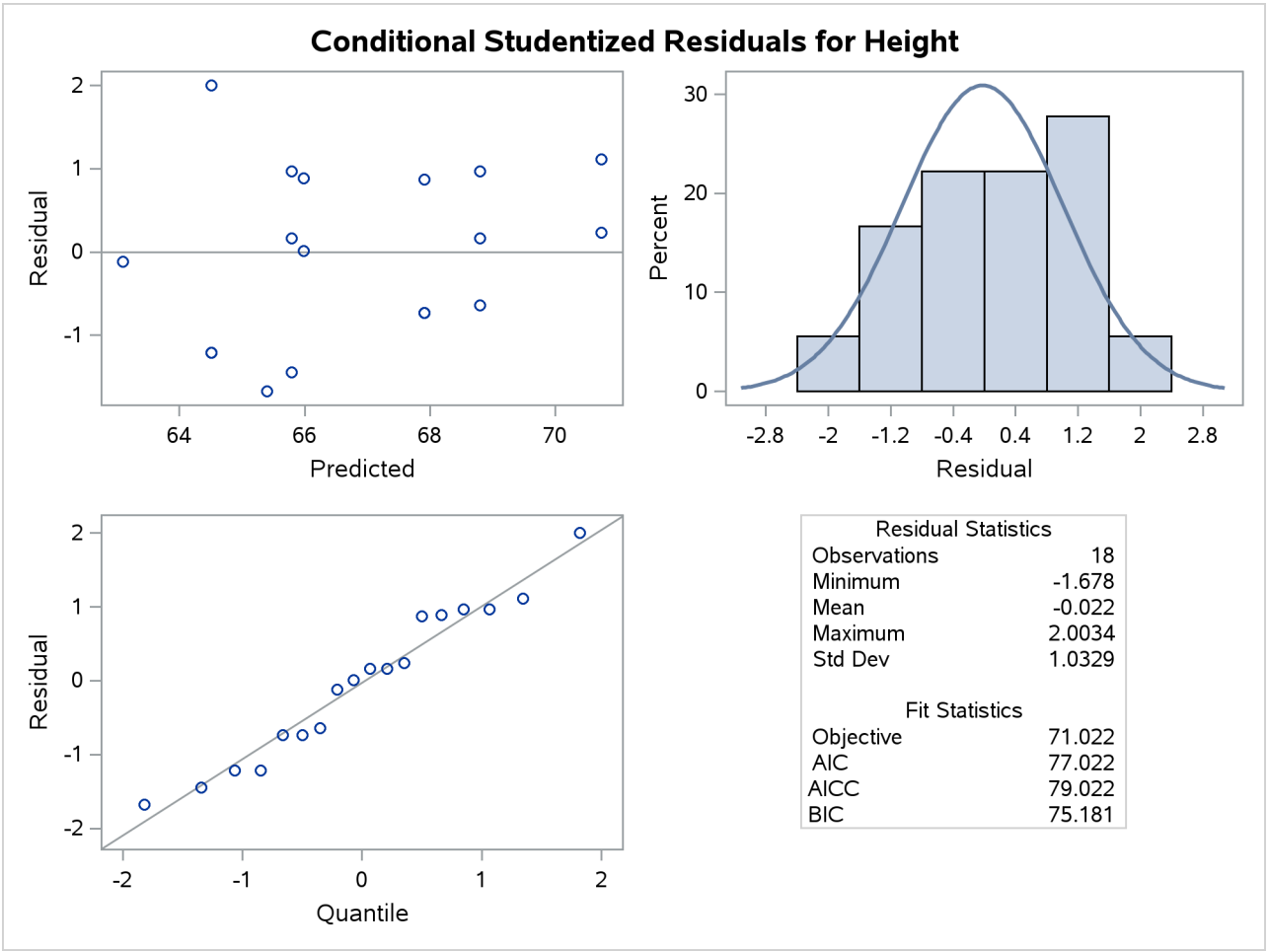

Influence Plots

The graphical features of the MIXED procedure enable you to generate plots of influence diagnostics and of deletion estimates. The type and number of plots produced depend on your modifiers of the INFLUENCE option in the MODEL statement and on the PLOTS= option in the PROC MIXED statement. Plots related to covariance parameters are produced only when diagnostics are computed by iterative methods (ITER=). The estimates of the fixed effects—and covariance parameters when updates are iterative—are plotted when you specify the ESTIMATES modifier or when you request PLOTS=INFLUENCEESTPLOT.

Two basic types of influence panels are shown in Figure 84.17 and Figure 84.18. The diagnostics panel shows Cook's *D* and CovRatio statistics for the fixed effects and the covariance parameters. For the SAS statements that produce these influence panels, see Example 84.8. In this example, the impact of subjects (Person) on the analysis is assessed. The Cook's *D* statistic measures a subject's impact on the estimates, and the CovRatio statistic measures a subject's impact on the precision of the estimates. Separate statistics are computed for the fixed effects and the covariance parameters. The CovRatio statistic has a threshold of 1.0. Values larger than 1.0 indicate that precision of the estimates is lost by exclusion of the observations in question. Values smaller than 1.0 indicate that precision is gained by exclusion of the observations from the analysis. For example, it is evident from Output 84.17 that person 20 has considerable impact on the

covariance parameter estimates and moderate influence on the fixed-effects estimates. Furthermore, exclusion of this subject from the analysis increases the precision of the covariance parameters, whereas the effect on the precision of the fixed effects is minor.

Output 84.18 shows another type of influence plot, a panel of the deletion estimates. Each plot within the panel corresponds to one of the model parameters. A reference line is drawn at the estimate based on the full data.

Figure 84.17 Influence Diagnostics

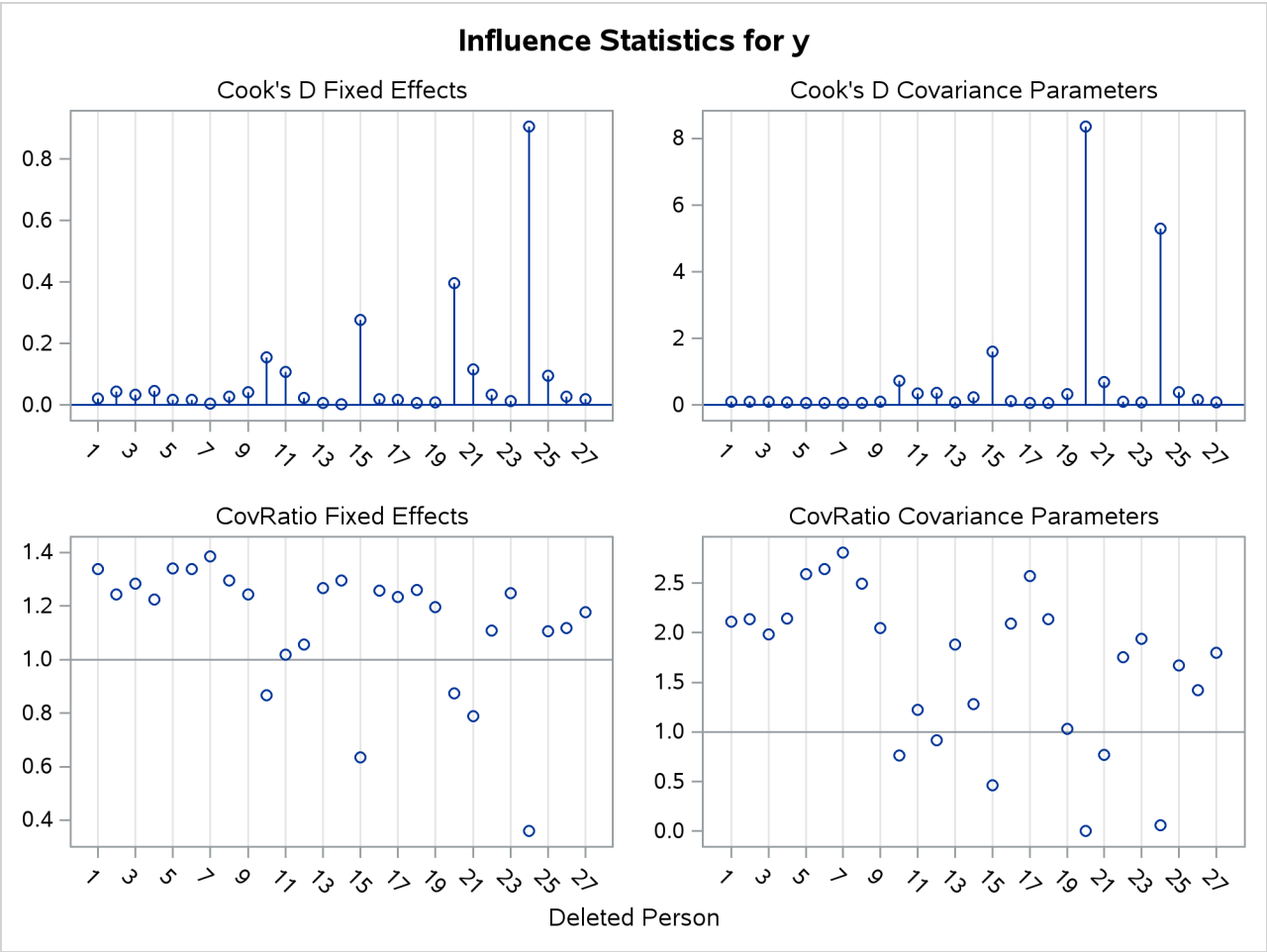

Figure 84.18 Deletion Estimates

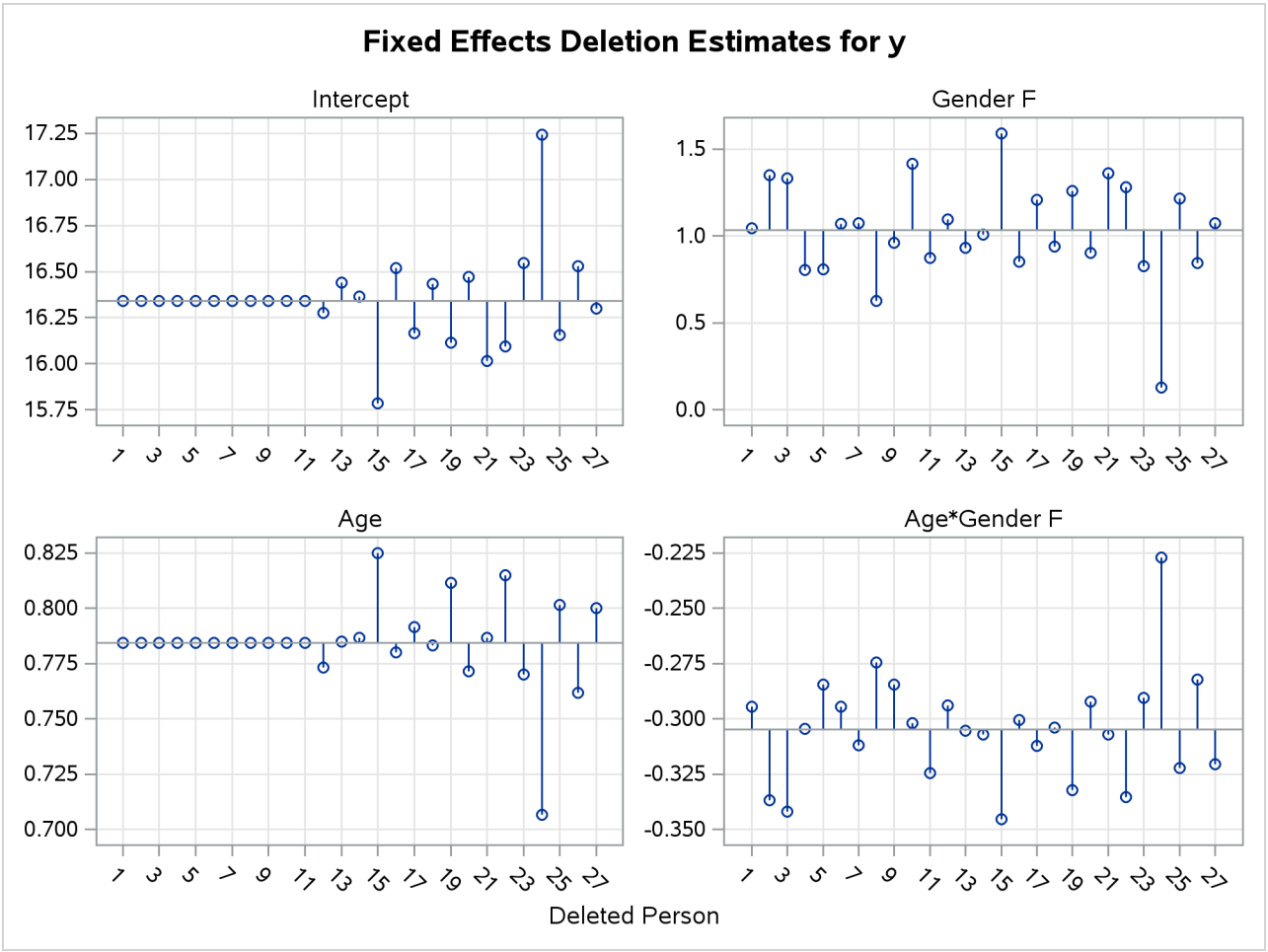

Computational Issues

Computational Method

In addition to numerous matrix-multiplication routines, PROC MIXED frequently uses the sweep operator (Goodnight 1979) and the Cholesky root (Golub and Van Loan 1989). The routines perform a modified W transformation (Goodnight and Hemmerle 1979) for **G**-side likelihood calculations and a direct method for **R**-side likelihood calculations. For the Type 3 F tests, PROC MIXED uses the algorithm described in Chapter 53, "The GLM Procedure."

PROC MIXED uses a ridge-stabilized Newton-Raphson algorithm to optimize either a full (ML) or residual (REML) likelihood function. The Newton-Raphson algorithm is preferred to the EM algorithm (Lindstrom and Bates 1988). PROC MIXED profiles the likelihood with respect to the fixed effects and also with respect to the residual variance whenever it appears reasonable to do so. The residual profiling can be avoided by using the NOPROFILE option of the PROC MIXED statement. PROC MIXED uses the MIVQUE0 method (Rao 1972; Giesbrecht 1989) to compute initial values.

The likelihoods that PROC MIXED optimizes are usually well-defined continuous functions with a single optimum. The Newton-Raphson algorithm typically performs well and finds the optimum in a few iterations. It is a quadratically converging algorithm, meaning that the error of the approximation near the optimum is squared at each iteration. The quadratic convergence property is evident when the convergence criterion drops to zero by factors of 10 or more.

Table 84.29 Notation for Order Calculations

Symbol	Number
p	Columns of $\mathbf{X}$
g	Columns of $\mathbf{Z}$
N	Observations
q	Covariance parameters
t	Maximum observations per subject
S	Subjects

Using the notation from Table 84.29, the following are estimates of the computational speed of the algorithms used in PROC MIXED. For likelihood calculations, the crossproducts matrix construction is of order $N(p+g)^2$ and the sweep operations are of order $(p+g)^3$. The first derivative calculations for parameters in $\mathbf{G}$ are of order qg^3 for ML and $q(g^3 + pg^2 + p^2g)$ for REML. If you specify a subject effect in the RANDOM statement and if you are not using the REPEATED statement, then replace g with g/S and q with qS in these calculations. The first derivative calculations for parameters in $\mathbf{R}$ are of order $qS(t^3 + gt^2 + g^2t)$ for ML and $qS(t^3 + (p+g)t^2 + (p^2+g^2)t)$ for REML. For the second derivatives, replace q with $q(q+1)/2$ in the first derivative expressions. When you specify both $\mathbf{G}$- and $\mathbf{R}$-side parameters (that is, when you use both the RANDOM and REPEATED statements), then additional calculations are required of an order equal to the sum of the orders for $\mathbf{G}$ and $\mathbf{R}$. Considerable execution times can result in this case.

For further details about the computational techniques used in PROC MIXED, see Wolfinger, Tobias, and Sall (1994).

Parameter Constraints

By default, some covariance parameters are assumed to satisfy certain boundary constraints during the Newton-Raphson algorithm. For example, variance components are constrained to be nonnegative, and autoregressive parameters are constrained to be between –1 and 1. You can remove these constraints with the NOBOUND option in the PARMS statement (or with the NOBOUND option in the PROC MIXED statement), but this can lead to estimates that produce an infinite likelihood. You can also introduce or change boundary constraints with the LOWERB= and UPPERB= options in the PARMS statement.

During the Newton-Raphson algorithm, a parameter might be set equal to one of its boundary constraints for a few iterations and then it might move away from the boundary. You see a missing value in the Criterion column of the "Iteration History" table whenever a boundary constraint is dropped.

For some data sets the final estimate of a parameter might equal one of its boundary constraints. This is usually not a cause for concern, but it might lead you to consider a different model. For instance, a variance component estimate can equal zero; in this case, you might want to drop the corresponding random effect from the model. However, be aware that changing the model in this fashion can affect degrees-of-freedom calculations.

Convergence Problems

For some data sets, the Newton-Raphson algorithm can fail to converge. Nonconvergence can result from a number of causes, including flat or ridged likelihood surfaces and ill-conditioned data.

It is also possible for PROC MIXED to converge to a point that is not the global optimum of the likelihood, although this usually occurs only with the spatial covariance structures.

If you experience convergence problems, the following points might be helpful:

- One useful tool is the PARMS statement, which lets you input initial values for the covariance parameters and performs a grid search over the likelihood surface.
- Sometimes the Newton-Raphson algorithm does not perform well when two of the covariance parameters are on a different scale—that is, when they are several orders of magnitude apart. This is because the Hessian matrix is processed jointly for the two parameters, and elements of it corresponding to one of the parameters can become close to internal tolerances in PROC MIXED. In this case, you can improve stability by rescaling the effects in the model so that the covariance parameters are on the same scale.
- Data that are extremely large or extremely small can adversely affect results because of the internal tolerances in PROC MIXED. Rescaling it can improve stability.
- For stubborn problems, you might want to specify ODS OUTPUT COVPARMS=*data-set-name* to output the "Covariance Parameter Estimates" table as a precautionary measure. That way, if the problem does not converge, you can read the final parameter values back into a new run with the PARMSDATA= option in the PARMS statement.
- Fisher scoring can be more robust than Newton-Raphson with poor MIVQUE(0) starting values. Specifying a SCORING= value of 5 or so might help to recover from poor starting values.
- Tuning the singularity options SINGULAR=, SINGCHOL=, and SINGRES= in the MODEL statement can improve the stability of the optimization process.
- Tuning the MAXITER= and MAXFUNC= options in the PROC MIXED statement can save resources. Also, the ITDETAILS option displays the values of all the parameters at each iteration.
- Using the NOPROFILE and NOBOUND options in the PROC MIXED statement might help convergence, although they can produce unusual results.
- Although the CONVH convergence criterion usually gives the best results, you might want to try CONVF or CONVG, possibly along with the ABSOLUTE option.
- If the convergence criterion reaches a relatively small value such as 1E–7 but never gets lower than 1E–8, you might want to specify CONVH=1E–6 in the PROC MIXED statement to get results; however, interpret the results with caution.
- An infinite likelihood during the iteration process means that the Newton-Raphson algorithm has stepped into a region where either the **R** or **V** matrix is nonpositive definite. This is usually no cause for concern as long as iterations continue. If PROC MIXED stops because of an infinite likelihood, recheck your model to make sure that no observations from the same subject are producing identical rows in **R** or **V** and that you have enough data to estimate the particular covariance structure you have selected. Any time that the final estimated likelihood is infinite, subsequent results should be interpreted with caution.

- A nonpositive definite Hessian matrix can indicate a surface saddlepoint or linear dependencies among the parameters.
- A warning message about the singularities of $\mathbf{X}$ changing indicates that there is some linear dependency in the estimate of $\mathbf{X}'\widehat{\mathbf{V}}^{-1}\mathbf{X}$ that is not found in $\mathbf{X}'\mathbf{X}$. This can adversely affect the likelihood calculations and optimization process. If you encounter this problem, make sure that your model specification is reasonable and that you have enough data to estimate the particular covariance structure you have selected. Rearranging effects in the MODEL statement so that the most significant ones are first can help, because PROC MIXED sweeps the estimate of $\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}$ in the order of the MODEL effects and the sweep is more stable if larger pivots are dealt with first. If this does not help, specifying starting values with the PARMS statement can place the optimization on a different and possibly more stable path.
- Lack of convergence can indicate model misspecification or a violation of the normality assumption.

Memory

Let p be the number of columns in $\mathbf{X}$, and let g be the number of columns in $\mathbf{Z}$. For large models, most of the memory resources are required for holding symmetric matrices of order p, g, and $p + g$. The approximate memory requirement in bytes is

$$40(p^2 + g^2) + 32(p + g)^2$$

If you have a large model that exceeds the memory capacity of your computer, see the suggestions listed under "Computing Time."

Computing Time

PROC MIXED is computationally intensive, and execution times can be long. In addition to the CPU time used in collecting sums and crossproducts and in solving the mixed model equations (as in PROC GLM), considerable CPU time is often required to compute the likelihood function and its derivatives. These latter computations are performed for every Newton-Raphson iteration.

If you have a model that takes too long to run, the following suggestions can be helpful:

- Examine the "Model Information" table to find out the number of columns in the $\mathbf{X}$ and $\mathbf{Z}$ matrices. A large number of columns in either matrix can greatly increase computing time. You might want to eliminate some higher-order effects if they are too large.
- If you have a $\mathbf{Z}$ matrix with a lot of columns, use the DDFM=BW option in the MODEL statement to eliminate the time required for the containment method.
- If possible, "factor out" a common effect from the effects in the RANDOM statement and make it the SUBJECT= effect. This creates a block-diagonal $\mathbf{G}$ matrix and can often speed calculations.
- If possible, use the same or nested SUBJECT= effects in all RANDOM and REPEATED statements.
- If your data set is very large, you might want to analyze it in pieces. The BY statement can help implement this strategy.

- In general, specify random effects with a lot of levels in the REPEATED statement and those with a few levels in the RANDOM statement.
- The METHOD=MIVQUE0 option runs faster than either the METHOD=REML or METHOD=ML option because it is noniterative.
- You can specify known values for the covariance parameters by using the HOLD= or NOITER option in the PARMS statement or the GDATA= option in the RANDOM statement. This eliminates the need for iteration.
- The LOGNOTE option in the PROC MIXED statement writes periodic messages to the SAS log concerning the status of the calculations. It can help you diagnose where the slowdown is occurring.

Examples: MIXED Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

The following are basic examples of the use of PROC MIXED. More examples and details can be found in Littell et al. (2006); Wolfinger (1997); Verbeke and Molenberghs (1997, 2000); Murray (1998); Singer (1998); Sullivan, Dukes, and Losina (1999), and Brown and Prescott (1999).

Example 84.1: Split-Plot Design

(View the complete code for this example (mixex1.sas) in the example repository.)

PROC MIXED can fit a variety of mixed models. One of the most common mixed models is the split-plot design. The split-plot design involves two experimental factors, A and B. Levels of A are randomly assigned to whole plots (main plots), and levels of B are randomly assigned to split plots (subplots) within each whole plot. The design provides more precise information about B than about A, and it often arises when A can be applied only to large experimental units. An example is where A represents irrigation levels for large plots of land and B represents different crop varieties planted in each large plot.

Consider the following data from Stroup (1989a), which arise from a balanced split-plot design with the whole plots arranged in a randomized complete-block design. The variable A is the whole-plot factor, and the variable B is the subplot factor. A traditional analysis of these data involves the construction of the whole-plot error (A*Block) to test A and the pooled residual error (B*Block and A*B*Block) to test B and A*B. To carry out this analysis with PROC GLM, you must use a TEST statement to obtain the correct F test for A.

Performing a mixed model analysis with PROC MIXED eliminates the need for the error term construction. PROC MIXED estimates variance components for Block, A*Block, and the residual, and it automatically incorporates the correct error terms into test statistics.

The following statements create a DATA set for a split-plot design with four blocks, three whole-plot levels, and two subplot levels:

```
data sp;
   input Block A B Y @@;
   datalines;
1 1 1  56  1 1 2  41
1 2 1  50  1 2 2  36
1 3 1  39  1 3 2  35
2 1 1  30  2 1 2  25
2 2 1  36  2 2 2  28
2 3 1  33  2 3 2  30
3 1 1  32  3 1 2  24
3 2 1  31  3 2 2  27
3 3 1  15  3 3 2  19
4 1 1  30  4 1 2  25
4 2 1  35  4 2 2  30
4 3 1  17  4 3 2  18
;
```

The following statements fit the split-plot model assuming random block effects:

```
proc mixed;
   class A B Block;
   model Y = A B A*B;
   random Block A*Block;
run;
```

The variables A, B, and Block are listed as classification variables in the CLASS statement. The columns of model matrix **X** consist of indicator variables corresponding to the levels of the fixed effects A, B, and A*B listed on the right side of the MODEL statement. The dependent variable Y is listed on the left side of the MODEL statement.

The columns of the model matrix **Z** consist of indicator variables corresponding to the levels of the random effects Block and A*Block. The **G** matrix is diagonal and contains the variance components of Block and A*Block. The **R** matrix is also diagonal and contains the residual variance.

The SAS statements produce Output 84.1.1–Output 84.1.8.

The "Model Information" table in Output 84.1.1 lists basic information about the split-plot model. REML is used to estimate the variance components, and the residual variance is profiled from the optimization.

Output 84.1.1 Results for Split-Plot Analysis

The Mixed Procedure

Model Information	
Data Set	WORK.SP
Dependent Variable	Y
Covariance Structure	Variance Components
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

The "Class Level Information" table in Output 84.1.2 lists the levels of all variables specified in the CLASS

statement. You can check this table to make sure that the data are correct.

Output 84.1.2 Split-Plot Example (*continued*)

Class Level Information		
Class	**Levels**	**Values**
A	3	1 2 3
B	2	1 2
Block	4	1 2 3 4

The "Dimensions" table in Output 84.1.3 lists the magnitudes of various vectors and matrices. The **X** matrix is seen to be 24×12, and the **Z** matrix is 24×16.

Output 84.1.3 Split-Plot Example (*continued*)

Dimensions	
Covariance Parameters	3
Columns in X	12
Columns in Z	16
Subjects	1
Max Obs per Subject	24

The "Number of Observations" table in Output 84.1.4 shows that all observations read from the data set are used in the analysis.

Output 84.1.4 Split-Plot Example (*continued*)

Number of Observations	
Number of Observations Read	24
Number of Observations Used	24
Number of Observations Not Used	0

PROC MIXED estimates the variance components for Block, A*Block, and the residual by REML. The REML estimates are the values that maximize the likelihood of a set of linearly independent error contrasts, and they provide a correction for the downward bias found in the usual maximum likelihood estimates. The objective function is –2 times the logarithm of the restricted likelihood, and PROC MIXED minimizes this objective function to obtain the estimates.

The minimization method is the Newton-Raphson algorithm, which uses the first and second derivatives of the objective function to iteratively find its minimum. The "Iteration History" table in Output 84.1.5 records the steps of that optimization process. For this example, only one iteration is required to obtain the estimates. The Evaluations column reveals that the restricted likelihood is evaluated once for each of the iterations. A criterion of 0 indicates that the Newton-Raphson algorithm has converged.

Output 84.1.5 Split-Plot Analysis (*continued*)

Iteration History			
Iteration	**Evaluations**	**-2 Res Log Like**	**Criterion**
0	1	139.81461222	
1	1	119.76184570	0.00000000

Convergence criteria met.

The REML estimates for the variance components of Block, A*Block, and the residual are 62.40, 15.38, and 9.36, respectively, as listed in the Estimate column of the "Covariance Parameter Estimates" table in Output 84.1.6.

Output 84.1.6 Split-Plot Analysis (*continued*)

Covariance Parameter Estimates	
Cov Parm	**Estimate**
Block	62.3958
A*Block	15.3819
Residual	9.3611

The "Fit Statistics" table in Output 84.1.7 lists several pieces of information about the fitted mixed model, including the residual log likelihood. The Akaike (AIC) and Bayesian (BIC) information criteria can be used to compare different models; the ones with smaller values are preferred. The AICC information criteria is a small-sample bias-adjusted form of the Akaike criterion (Hurvich and Tsai 1989).

Output 84.1.7 Split-Plot Analysis (*continued*)

Fit Statistics	
-2 Res Log Likelihood	119.8
AIC (Smaller is Better)	125.8
AICC (Smaller is Better)	127.5
BIC (Smaller is Better)	123.9

Finally, the fixed effects are tested by using Type 3 estimable functions (Output 84.1.8).

Output 84.1.8 Split-Plot Analysis (*continued*)

Type 3 Tests of Fixed Effects				
Effect	**Num DF**	**Den DF**	**F Value**	**Pr > F**
A	2	6	4.07	0.0764
B	1	9	19.39	0.0017
A*B	2	9	4.02	0.0566

The tests match the one obtained from the following PROC GLM statements:

```
proc glm data=sp;
   class A B Block;
   model Y = A B A*B Block A*Block;
   test h=A e=A*Block;
run;
```

You can continue this analysis by producing solutions for the fixed and random effects and then testing various linear combinations of them by using the CONTRAST and ESTIMATE statements. If you use the same CONTRAST and ESTIMATE statements with PROC GLM, the test statistics correspond to the fixed-effects-only model. The test statistics from PROC MIXED incorporate the random effects.

The various "inference space" contrasts given by Stroup (1989a) can be implemented via the ESTIMATE statement. Consider the following examples:

```
proc mixed data=sp;
   class A B Block;
   model Y = A B A*B;
   random Block A*Block;
   estimate 'a1 mean narrow'
             intercept 1 A 1 B .5 .5 A*B .5 .5 |
             Block     .25 .25 .25 .25
             A*Block   .25 .25 .25 .25 0 0 0 0 0 0 0 0;

   estimate 'a1 mean intermed'
             intercept 1 A 1 B .5 .5 A*B .5 .5 |
             Block     .25 .25 .25 .25;
   estimate 'a1 mean broad'
             intercept 1 a 1 b .5 .5 A*B .5 .5;
run;
```

These statements result in Output 84.1.9.

Output 84.1.9 Inference Space Results

The Mixed Procedure

Estimates					
Label	Estimate	Standard Error	DF	t Value	Pr > |t|
a1 mean narrow	32.8750	1.0817	9	30.39	<.0001
a1 mean intermed	32.8750	2.2396	9	14.68	<.0001
a1 mean broad	32.8750	4.5403	9	7.24	<.0001

Note that all the estimates are equal, but their standard errors increase with the size of the inference space. The narrow inference space consists of the observed levels of Block and A*Block, and the t-statistic value of 30.39 applies only to these levels. This is the same t statistic computed by PROC GLM, because it computes standard errors from the narrow inference space. The intermediate inference space consists of the observed levels of Block and the entire population of levels from which A*Block are sampled. The t-statistic value of 14.68 applies to this intermediate space. The broad inference space consists of arbitrary random levels of both Block and A*Block, and the t-statistic value of 7.24 is appropriate. Note that the larger the inference space, the weaker the conclusion. However, the broad inference space is usually the one of interest, and even in this space conclusive results are common. The highly significant p-value for 'a1 mean broad' is an

example. You can also obtain the 'a1 mean broad' result by specifying A in an LSMEANS statement. For more discussion of the inference space concept, see McLean, Sanders, and Stroup (1991).

The following statements illustrate another feature of the RANDOM statement. Recall that the basic statements for a split-plot design with whole plots arranged in randomized blocks are as follows.

```
proc mixed;
   class A B Block;
   model Y = A B A*B;
   random Block A*Block;
run;
```

An equivalent way of specifying this model is as follows:

```
/* equivalent model */
proc mixed data=sp;
   class A B Block;
   model Y = A B A*B;
   random intercept A / subject=Block;
run;
```

In general, if all of the effects in the RANDOM statement can be nested within one effect, you can specify that one effect by using the SUBJECT= option. The subject effect is, in a sense, "factored out" of the random effects. The specification that uses the SUBJECT= effect can result in faster execution times for large problems because PROC MIXED is able to perform the likelihood calculations separately for each subject.

Example 84.2: Repeated Measures

(View the complete code for this example (mixex2.sas) in the example repository.)

The following data are from Pothoff and Roy (1964) and consist of growth measurements for 11 girls and 16 boys at ages 8, 10, 12, and 14. Some of the observations are suspect (for example, the third observation for person 20); however, all of the data are used here for comparison purposes.

The analysis strategy employs a linear growth curve model for the boys and girls as well as a variance-covariance model that incorporates correlations for all of the observations arising from the same person. The data are assumed to be Gaussian, and their likelihood is maximized to estimate the model parameters. For overviews of this approach to repeated measures, see Jennrich and Schluchter (1986); Louis (1988); Crowder and Hand (1990); Diggle, Liang, and Zeger (1994); Everitt (1995). Jennrich and Schluchter present results for the Pothoff and Roy data from various covariance structures. The PROC MIXED statements to fit an unstructured variance matrix (their Model 2) are as follows:

```
data pr;
   input Person Gender $ y1 y2 y3 y4;
   y=y1; Age=8;  output;
   y=y2; Age=10; output;
   y=y3; Age=12; output;
   y=y4; Age=14; output;
   drop y1-y4;
   datalines;
 1   F   21.0    20.0    21.5    23.0
 2   F   21.0    21.5    24.0    25.5
```

```
 3   F   20.5    24.0    24.5    26.0
 4   F   23.5    24.5    25.0    26.5
 5   F   21.5    23.0    22.5    23.5
 6   F   20.0    21.0    21.0    22.5
 7   F   21.5    22.5    23.0    25.0
 8   F   23.0    23.0    23.5    24.0
 9   F   20.0    21.0    22.0    21.5
10   F   16.5    19.0    19.0    19.5
11   F   24.5    25.0    28.0    28.0
12   M   26.0    25.0    29.0    31.0
13   M   21.5    22.5    23.0    26.5
14   M   23.0    22.5    24.0    27.5
15   M   25.5    27.5    26.5    27.0
16   M   20.0    23.5    22.5    26.0
17   M   24.5    25.5    27.0    28.5
18   M   22.0    22.0    24.5    26.5
19   M   24.0    21.5    24.5    25.5
20   M   23.0    20.5    31.0    26.0
21   M   27.5    28.0    31.0    31.5
22   M   23.0    23.0    23.5    25.0
23   M   21.5    23.5    24.0    28.0
24   M   17.0    24.5    26.0    29.5
25   M   22.5    25.5    25.5    26.0
26   M   23.0    24.5    26.0    30.0
27   M   22.0    21.5    23.5    25.0
;

proc mixed data=pr method=ml covtest;
   class Person Gender;
   model y = Gender Age Gender*Age / s;
   repeated / type=un subject=Person r;
run;
```

To follow Jennrich and Schluchter, this example uses maximum likelihood (METHOD=ML) instead of the default REML to estimate the unknown covariance parameters. The COVTEST option requests asymptotic tests of all the covariance parameters.

The MODEL statement first lists the dependent variable Y. The fixed effects are then listed after the equal sign. The variable Gender requests a different intercept for the girls and boys, Age models an overall linear growth trend, and Gender*Age makes the slopes different over time. It is actually not necessary to specify Age separately, but doing so enables PROC MIXED to carry out a test for heterogeneous slopes. The SOLUTION option requests the display of the fixed-effects solution vector.

The REPEATED statement contains no effects, taking advantage of the default assumption that the observations are ordered similarly for each subject. The TYPE=UN option requests an unstructured block for each SUBJECT=Person. The **R** matrix is, therefore, block diagonal with 27 blocks, each block consisting of identical 4×4 unstructured matrices. The 10 parameters of these unstructured blocks make up the covariance parameters estimated by maximum likelihood. The R option requests that the first block of **R** be displayed.

The results from this analysis are shown in Output 84.2.1–Output 84.2.9.

Output 84.2.1 Repeated Measures Analysis with Unstructured Covariance Matrix

The Mixed Procedure

Model Information	
Data Set	WORK.PR
Dependent Variable	y
Covariance Structure	Unstructured
Subject Effect	Person
Estimation Method	ML
Residual Variance Method	None
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Between-Within

In Output 84.2.1, the covariance structure is listed as "Unstructured," and no residual variance is used with this structure. The default degrees-of-freedom method here is "Between-Within."

Output 84.2.2 Repeated Measures Analysis (*continued*)

Class Level Information		
Class	**Levels**	**Values**
Person	27	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
Gender	2	F M

In Output 84.2.2, note that Person has 27 levels and Gender has 2.

Output 84.2.3 Repeated Measures Analysis (*continued*)

Dimensions	
Covariance Parameters	10
Columns in X	6
Columns in Z	0
Subjects	27
Max Obs per Subject	4

In Output 84.2.3, the 10 covariance parameters result from the 4×4 unstructured blocks of $\mathbf{R}$. There is no $\mathbf{Z}$ matrix for this model, and each of the 27 subjects has a maximum of 4 observations.

Output 84.2.4 Repeated Measures Analysis (*continued*)

Number of Observations	
Number of Observations Read	108
Number of Observations Used	108
Number of Observations Not Used	0

Iteration History			
Iteration	**Evaluations**	**-2 Log Like**	**Criterion**
0	1	478.24175986	
1	2	419.47721707	0.00000152
2	1	419.47704812	0.00000000

Output 84.2.4 *continued*

Convergence criteria met.

Three Newton-Raphson iterations are required to find the maximum likelihood estimates (Output 84.2.4). The default relative Hessian criterion has a final value less than 1E–8, indicating the convergence of the Newton-Raphson algorithm and the attainment of an optimum.

Output 84.2.5 Repeated Measures Analysis (*continued*)

Estimated R Matrix for Person 1				
Row	**Col1**	**Col2**	**Col3**	**Col4**
1	5.1192	2.4409	3.6105	2.5222
2	2.4409	3.9279	2.7175	3.0624
3	3.6105	2.7175	5.9798	3.8235
4	2.5222	3.0624	3.8235	4.6180

The 4×4 matrix in Output 84.2.5 is the estimated unstructured covariance matrix. It is the estimate of the first block of **R**, and the other 26 blocks all have the same estimate.

Output 84.2.6 Repeated Measures Analysis (*continued*)

Covariance Parameter Estimates					
Cov Parm	**Subject**	**Estimate**	**Standard Error**	**Z Value**	**Pr Z**
UN(1,1)	Person	5.1192	1.4169	3.61	0.0002
UN(2,1)	Person	2.4409	0.9835	2.48	0.0131
UN(2,2)	Person	3.9279	1.0824	3.63	0.0001
UN(3,1)	Person	3.6105	1.2767	2.83	0.0047
UN(3,2)	Person	2.7175	1.0740	2.53	0.0114
UN(3,3)	Person	5.9798	1.6279	3.67	0.0001
UN(4,1)	Person	2.5222	1.0649	2.37	0.0179
UN(4,2)	Person	3.0624	1.0135	3.02	0.0025
UN(4,3)	Person	3.8235	1.2508	3.06	0.0022
UN(4,4)	Person	4.6180	1.2573	3.67	0.0001

The "Covariance Parameter Estimates" table in Output 84.2.6 lists the 10 estimated covariance parameters in order; note their correspondence to the first block of **R** displayed in Output 84.2.5. The parameter estimates are labeled according to their location in the block in the Cov Parm column, and all of these estimates are associated with Person as the subject effect. The Std Error column lists approximate standard errors of the covariance parameters obtained from the inverse Hessian matrix. These standard errors lead to approximate Wald Z statistics, which are compared with the standard normal distribution The results of these tests indicate that all the parameters are significantly different from 0; however, the Wald test can be unreliable in small samples.

To carry out Wald tests of various linear combinations of these parameters, use the following procedure. First, run the statements again, adding the ASYCOV option and an ODS statement:

```
ods output CovParms=cp AsyCov=asy;
proc mixed data=pr method=ml covtest asycov;
   class Person Gender;
   model y = Gender Age Gender*Age / s;
   repeated / type=un subject=Person r;
run;
```

This creates two data sets, cp and asy, which contain the covariance parameter estimates and their asymptotic variance covariance matrix, respectively. Then read these data sets into the SAS/IML matrix programming language as follows:

```
proc iml;
   use cp;
   read all var {Estimate} into est;
   use asy;
   read all var ('CovP1':'CovP10') into asy;
quit;
```

You can then construct your desired linear combinations and corresponding quadratic forms with the asy matrix.

Output 84.2.7 Repeated Measures Analysis (*continued*)

Fit Statistics	
-2 Log Likelihood	419.5
AIC (Smaller is Better)	447.5
AICC (Smaller is Better)	452.0
BIC (Smaller is Better)	465.6

Null Model Likelihood Ratio Test		
DF	Chi-Square	Pr > ChiSq
9	58.76	<.0001

The null model likelihood ratio test (LRT) in Output 84.2.7 is highly significant for this model, indicating that the unstructured covariance matrix is preferred to the diagonal matrix of the ordinary least squares null model. The degrees of freedom for this test is 9, which is the difference between 10 and the 1 parameter for the null model's diagonal matrix.

Output 84.2.8 Repeated Measures Analysis (*continued*)

Solution for Fixed Effects						
Effect	Gender	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		15.8423	0.9356	25	16.93	<.0001
Gender	F	1.5831	1.4658	25	1.08	0.2904
Gender	M	0	.	.	.	.
Age		0.8268	0.07911	25	10.45	<.0001
Age*Gender	F	-0.3504	0.1239	25	-2.83	0.0091
Age*Gender	M	0	.	.	.	.

The "Solution for Fixed Effects" table in Output 84.2.8 lists the solution vector for the fixed effects. The estimate of the boys' intercept is 15.8423, while that for the girls is $15.8423 + 1.5831 = 17.0654$. Similarly, the estimate for the boys' slope is 0.8268, while that for the girls is $0.8268 - 0.3504 = 0.4764$. Thus the girls' starting point is larger than that for the boys, but their growth rate is about half that of the boys.

Note that two of the estimates equal 0; this is a result of the overparameterized model used by PROC MIXED. You can obtain a full-rank parameterization by using the following MODEL statement:

```
model y = Gender Gender*Age / noint s;
```

Here, the NOINT option causes the different intercepts to be fit directly as the two levels of Gender. However, this alternative specification results in different tests for these effects.

Output 84.2.9 Repeated Measures Analysis (*continued*)

Type 3 Tests of Fixed Effects

Effect	Num DF	Den DF	F Value	Pr > F
Gender	1	25	1.17	0.2904
Age	1	25	110.54	<.0001
Age*Gender	1	25	7.99	0.0091

The "Type 3 Tests of Fixed Effects" table in Output 84.2.9 displays Type 3 tests for all of the fixed effects. These tests are partial in the sense that they account for all of the other fixed effects in the model. In addition, you can use the HTYPE= option in the MODEL statement to obtain Type 1 (sequential) or Type 2 (also partial) tests of effects.

It is usually best to consider higher-order terms first, and in this case the Age*Gender test reveals a difference between the slopes that is statistically significant at the 1% level. Note that the p-value for this test (0.0091) is the same as the p-value in the "Age*Gender F" row in the "Solution for Fixed Effects" table (Output 84.2.8) and that the F statistic (7.99) is the square of the t statistic (–2.83), ignoring rounding error. Similar connections are evident among the other rows in these two tables.

The Age test is one for an overall growth curve accounting for possible heterogeneous slopes, and it is highly significant. Finally, the Gender row tests the null hypothesis of a common intercept, and this hypothesis cannot be rejected from these data.

As an alternative to the F tests shown here, you can carry out likelihood ratio tests of various hypotheses by fitting the reduced models, subtracting –2 log likelihoods, and comparing the resulting statistics with χ^2 distributions.

Since the different levels of the repeated effect represent different years, it is natural to try fitting a time series model to the data within each subject. To obtain time series structures in $\mathbf{R}$, you can replace TYPE=UN with TYPE=AR(1) or TYPE=TOEP to obtain the first- or nth-order autoregressive covariance matrices, respectively. For example, the statements to fit an AR(1) structure are as follows:

```
/* first-order autoregressive */
proc mixed data=pr method=ml;
   class Person Gender;
   model y = Gender Age Gender*Age / s;
   repeated / type=ar(1) sub=Person r;
run;
```

To fit a random coefficients model, use the following statements:

```
/* random coefficients model */
proc mixed data=pr method=ml;
   class Person Gender;
   model y = Gender Age Gender*Age / s;
   random intercept Age / type=un sub=Person g;
run;
```

This specifies an unstructured covariance matrix for the random intercept and slope. In mixed model notation, $\mathbf{G}$ is block diagonal with identical 2×2 unstructured blocks for each person. By default, $\mathbf{R}$ becomes $\sigma^2\mathbf{I}$. See Example 84.5 for further information about this model.

Finally, you can fit a compound symmetry structure by using TYPE=CS, as follows:

```
proc mixed data=pr method=ml covtest;
   class Person Gender;
   model y = Gender Age Gender*Age / s;
   repeated / type=cs subject=Person r;
run;
```

The results from this analysis are shown in Output 84.2.10–Output 84.2.17.

The "Model Information" table in Output 84.2.10 is the same as before except for the change in "Covariance Structure."

Output 84.2.10 Repeated Measures Analysis with Compound Symmetry Structure

The Mixed Procedure

Model Information	
Data Set	WORK.PR
Dependent Variable	y
Covariance Structure	Compound Symmetry
Subject Effect	Person
Estimation Method	ML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Between-Within

The "Dimensions" table in Output 84.2.11 shows that there are only two covariance parameters in the compound symmetry model; this covariance structure has common variance and common covariance.

Output 84.2.11 Analysis with Compound Symmetry (*continued*)

Class Level Information

Class	Levels	Values
Person	27	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
Gender	2	F M

Output 84.2.11 *continued*

Dimensions	
Covariance Parameters	2
Columns in X	6
Columns in Z	0
Subjects	27
Max Obs per Subject	4

Number of Observations	
Number of Observations Read	108
Number of Observations Used	108
Number of Observations Not Used	0

Since the data are balanced, only one step is required to find the estimates (Output 84.2.12).

Output 84.2.12 Analysis with Compound Symmetry (*continued*)

Iteration History			
Iteration	Evaluations	-2 Log Like	Criterion
0	1	478.24175986	
1	1	428.63905802	0.00000000

Convergence criteria met.

Output 84.2.13 displays the estimated **R** matrix for the first subject. Note the compound symmetry structure here, which consists of a common covariance with a diagonal enhancement.

Output 84.2.13 Analysis with Compound Symmetry (*continued*)

Estimated R Matrix for Person 1				
Row	Col1	Col2	Col3	Col4
1	4.9052	3.0306	3.0306	3.0306
2	3.0306	4.9052	3.0306	3.0306
3	3.0306	3.0306	4.9052	3.0306
4	3.0306	3.0306	3.0306	4.9052

The common covariance is estimated to be 3.0306, as listed in the CS row of the "Covariance Parameter Estimates" table in Output 84.2.14, and the residual variance is estimated to be 1.8746, as listed in the Residual row. You can use these two numbers to estimate the intraclass correlation coefficient (ICC) for this model. Here, the ICC estimate equals $3.0306/(3.0306 + 1.8746) = 0.6178$. You can also obtain this number by adding the RCORR option to the REPEATED statement.

Output 84.2.14 Analysis with Compound Symmetry (*continued*)

Covariance Parameter Estimates					
Cov Parm	Subject	Estimate	Standard Error	Z Value	Pr Z
CS	Person	3.0306	0.9552	3.17	0.0015
Residual		1.8746	0.2946	6.36	<.0001

In the case shown in Output 84.2.15, the null model LRT has only one degree of freedom, corresponding to the common covariance parameter. The test indicates that modeling this extra covariance is superior to fitting the simple null model.

Output 84.2.15 Analysis with Compound Symmetry (*continued*)

Fit Statistics	
-2 Log Likelihood	428.6
AIC (Smaller is Better)	440.6
AICC (Smaller is Better)	441.5
BIC (Smaller is Better)	448.4

Null Model Likelihood Ratio Test		
DF	Chi-Square	Pr > ChiSq
1	49.60	<.0001

Note that the fixed-effects estimates and their standard errors (Output 84.2.16) are not very different from those in the preceding unstructured example (Output 84.2.8).

Output 84.2.16 Analysis with Compound Symmetry (*continued*)

Solution for Fixed Effects						
Effect	Gender	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		16.3406	0.9631	25	16.97	<.0001
Gender	F	1.0321	1.5089	25	0.68	0.5003
Gender	M	0	.	.	.	.
Age		0.7844	0.07654	79	10.25	<.0001
Age*Gender	F	-0.3048	0.1199	79	-2.54	0.0130
Age*Gender	M	0	.	.	.	.

The F tests shown in Output 84.2.17 are also similar to those from the preceding unstructured example (Output 84.2.9). Again, the slopes are significantly different but the intercepts are not.

Output 84.2.17 Analysis with Compound Symmetry (*continued*)

Type 3 Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
Gender	1	25	0.47	0.5003
Age	1	79	111.10	<.0001
Age*Gender	1	79	6.46	0.0130

You can fit the same compound symmetry model with the following specification by using the RANDOM statement:

```
proc mixed data=pr method=ml;
   class Person Gender;
   model y = Gender Age Gender*Age / ddfm=bw s;
   random int / subject=Person;
```

```
run;
```

Compound symmetry is the structure that Jennrich and Schluchter deemed best among the ones they fit. To carry the analysis one step further, you can use the GROUP= option as follows to specify heterogeneity of this structure across girls and boys:

```
proc mixed data=pr method=ml;
   class Person Gender;
   model y = Gender Age Gender*Age / s;
   repeated / type=cs subject=Person group=Gender;
run;
```

The results from this analysis are shown in Output 84.2.18–Output 84.2.24. Note that in Output 84.2.18 Gender is listed as a "Group Effect."

Output 84.2.18 Repeated Measures Analysis with Heterogeneous Structures

The Mixed Procedure

Model Information	
Data Set	WORK.PR
Dependent Variable	y
Covariance Structure	Compound Symmetry
Subject Effect	Person
Group Effect	Gender
Estimation Method	ML
Residual Variance Method	None
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Between-Within

The four covariance parameters listed in Output 84.2.19 result from the two compound symmetry structures corresponding to the two levels of Gender.

Output 84.2.19 Analysis with Heterogeneous Structures (*continued*)

Class Level Information

Class	Levels	Values
Person	27	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27
Gender	2	F M

Dimensions	
Covariance Parameters	4
Columns in X	6
Columns in Z	0
Subjects	27
Max Obs per Subject	4

Number of Observations	
Number of Observations Read	108
Number of Observations Used	108
Number of Observations Not Used	0

As Output 84.2.20 shows, even with the heterogeneity, only one iteration is required for convergence.

Output 84.2.20 Analysis with Heterogeneous Structures (*continued*)

Iteration History			
Iteration	**Evaluations**	**-2 Log Like**	**Criterion**
0	1	478.24175986	
1	1	408.81297228	0.00000000

Convergence criteria met.

The "Covariance Parameter Estimates" table in Output 84.2.21 lists the heterogeneous estimates. Note that both the common covariance and the diagonal enhancement differ between girls and boys.

Output 84.2.21 Analysis with Heterogeneous Structures (*continued*)

Covariance Parameter Estimates			
Cov Parm	**Subject**	**Group**	**Estimate**
Variance	Person	Gender F	0.5900
CS	Person	Gender F	3.8804
Variance	Person	Gender M	2.7577
CS	Person	Gender M	2.4463

As Output 84.2.22 shows, both Akaike's information criterion (424.8) and Schwarz's Bayesian information criterion (435.2) are smaller for this model than for the homogeneous compound symmetry model (440.6 and 448.4, respectively). This indicates that the heterogeneous model is more appropriate. To construct the likelihood ratio test between the two models, subtract the –2 log likelihood values: $428.6 - 408.8 = 19.8$. Comparing this value with the χ^2 distribution with two degrees of freedom yields a p-value less than 0.0001, again favoring the heterogeneous model.

Output 84.2.22 Analysis with Heterogeneous Structures (*continued*)

Fit Statistics	
-2 Log Likelihood	408.8
AIC (Smaller is Better)	424.8
AICC (Smaller is Better)	426.3
BIC (Smaller is Better)	435.2

Null Model Likelihood Ratio Test		
DF	**Chi-Square**	**Pr > ChiSq**
3	69.43	<.0001

Note that the fixed-effects estimates shown in Output 84.2.23 are the same as in the homogeneous case, but the standard errors are different.

Output 84.2.23 Analysis with Heterogeneous Structures (*continued*)

Solution for Fixed Effects						
Effect	Gender	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		16.3406	1.1130	25	14.68	<.0001
Gender	F	1.0321	1.3890	25	0.74	0.4644
Gender	M	0	.	.	.	.
Age		0.7844	0.09283	79	8.45	<.0001
Age*Gender	F	-0.3048	0.1063	79	-2.87	0.0053
Age*Gender	M	0	.	.	.	.

The fixed-effects tests shown in Output 84.2.24 are similar to those from previous models, although the *p*-values do change as a result of specifying a different covariance structure. It is important for you to select a reasonable covariance structure in order to obtain valid inferences for your fixed effects.

Output 84.2.24 Analysis with Heterogeneous Structures (*continued*)

Type 3 Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
Gender	1	25	0.55	0.4644
Age	1	79	141.37	<.0001
Age*Gender	1	79	8.22	0.0053

Example 84.3: Plotting the Likelihood

(View the complete code for this example (mixex3.sas) in the example repository.)

The data for this example are from Hemmerle and Hartley (1973) and are also used for an example in the VARCOMP procedure. The response variable consists of measurements from an oven experiment, and the model contains a fixed effect A and random effects B and A*B.

The SAS statements are as follows:

```
data hh;
   input a b y @@;
   datalines;
1 1 237   1 1 254    1 1 246
1 2 178   1 2 179
2 1 208   2 1 178    2 1 187
2 2 146   2 2 145    2 2 141
3 1 186   3 1 183
3 2 142   3 2 125    3 2 136
;

ods output ParmSearch=parms;
proc mixed data=hh asycov mmeq mmeqsol covtest;
   class a b;
   model y = a / outp=predicted;
   random b a*b;
```

```
   lsmeans a;
   parms (17 to 20 by .1) (.3 to .4 by .005) (1.0);
run;
proc print data=predicted;
run;
```

The ASYCOV option in the PROC MIXED statement requests the asymptotic variance matrix of the covariance parameter estimates. This matrix is the observed inverse Fisher information matrix, which equals $2\mathbf{H}^{-1}$, where $\mathbf{H}$ is the Hessian matrix of the objective function evaluated at the final covariance parameter estimates. The MMEQ and MMEQSOL options in the PROC MIXED statement request that the mixed model equations and their solution be displayed.

The OUTP= option in the MODEL statement produces the data set predicted, containing the predicted values. Least squares means (LSMEANS) are requested for A. The PARMS and ODS statements are used to construct a data set containing the likelihood surface.

The results from this analysis are shown in Output 84.3.1–Output 84.3.13.

The "Model Information" table in Output 84.3.1 lists details about this variance components model.

Output 84.3.1 Model Information

The Mixed Procedure

Model Information	
Data Set	WORK.HH
Dependent Variable	y
Covariance Structure	Variance Components
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

The "Class Level Information" table in Output 84.3.2 lists the levels for A and B.

Output 84.3.2 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
a	3	1 2 3
b	2	1 2

The "Dimensions" table in Output 84.3.3 reveals that $\mathbf{X}$ is 16×4 and $\mathbf{Z}$ is 16×8. Since there are no SUBJECT= effects, PROC MIXED considers the data to be effectively from one subject with 16 observations.

Output 84.3.3 Model Dimensions and Number of Observations

Dimensions	
Covariance Parameters	3
Columns in X	4
Columns in Z	8
Subjects	1
Max Obs per Subject	16

Number of Observations	
Number of Observations Read	16
Number of Observations Used	16
Number of Observations Not Used	0

Only a portion of the "Parameter Search" table is shown in Output 84.3.4 because the full listing has 651 rows.

Output 84.3.4 Selected Results of Parameter Search

The Mixed Procedure

CovP1	CovP2	CovP3	Variance	Res Log Like	-2 Res Log Like
17.0000	0.3000	1.0000	80.1400	-52.4699	104.9399
17.0000	0.3050	1.0000	80.0466	-52.4697	104.9393
17.0000	0.3100	1.0000	79.9545	-52.4694	104.9388
17.0000	0.3150	1.0000	79.8637	-52.4692	104.9384
17.0000	0.3200	1.0000	79.7742	-52.4691	104.9381
17.0000	0.3250	1.0000	79.6859	-52.4690	104.9379
17.0000	0.3300	1.0000	79.5988	-52.4689	104.9378
17.0000	0.3350	1.0000	79.5129	-52.4689	104.9377
17.0000	0.3400	1.0000	79.4282	-52.4689	104.9377
17.0000	0.3450	1.0000	79.3447	-52.4689	104.9378
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
20.0000	0.3550	1.0000	78.2003	-52.4683	104.9366
20.0000	0.3600	1.0000	78.1201	-52.4684	104.9368
20.0000	0.3650	1.0000	78.0409	-52.4685	104.9370
20.0000	0.3700	1.0000	77.9628	-52.4687	104.9373
20.0000	0.3750	1.0000	77.8857	-52.4689	104.9377
20.0000	0.3800	1.0000	77.8096	-52.4691	104.9382
20.0000	0.3850	1.0000	77.7345	-52.4693	104.9387
20.0000	0.3900	1.0000	77.6603	-52.4696	104.9392
20.0000	0.3950	1.0000	77.5871	-52.4699	104.9399
20.0000	0.4000	1.0000	77.5148	-52.4703	104.9406

As Output 84.3.5 shows, convergence occurs quickly because PROC MIXED starts from the best value from the grid search.

Output 84.3.5 Iteration History and Convergence Status

Iteration History			
Iteration	**Evaluations**	**-2 Res Log Like**	**Criterion**
1	2	104.93416367	0.00000000

Convergence criteria met.

The "Covariance Parameter Estimates" table in Output 84.3.6 lists the variance components estimates. Note that B is much more variable than A*B.

Output 84.3.6 Estimated Covariance Parameters

Covariance Parameter Estimates				
Cov Parm	**Estimate**	**Standard Error**	**Z Value**	**Pr > Z**
b	1464.36	2098.01	0.70	0.2426
a*b	26.9581	59.6570	0.45	0.3257
Residual	78.8426	35.3512	2.23	0.0129

The asymptotic covariance matrix in Output 84.3.7 also reflects the large variability of B relative to A*B.

Output 84.3.7 Asymptotic Covariance Matrix of Covariance Parameters

Asymptotic Covariance Matrix of Estimates				
Row	**Cov Parm**	**CovP1**	**CovP2**	**CovP3**
1	b	4401640	1.2831	-273.32
2	a*b	1.2831	3558.96	-502.84
3	Residual	-273.32	-502.84	1249.71

As Output 84.3.8 shows, the PARMS likelihood ratio test (LRT) compares the best model from the grid search with the final fitted model. Since these models are nearly the same, the LRT is not significant.

Output 84.3.8 Fit Statistics and Likelihood Ratio Test

Fit Statistics	
-2 Res Log Likelihood	104.9
AIC (Smaller is Better)	110.9
AICC (Smaller is Better)	113.6
BIC (Smaller is Better)	107.0

PARMS Model Likelihood Ratio Test		
DF	**Chi-Square**	**Pr > ChiSq**
2	0.00	1.0000

The mixed model equations are analogous to the normal equations in the standard linear model. As Output 84.3.9 shows, for this example, rows 1–4 correspond to the fixed effects, rows 5–12 correspond to the random effects, and Col13 corresponds to the dependent variable.

Output 84.3.9 Mixed Model Equations

Mixed Model Equations														
Row	Effect	a	b	Col1	Col2	Col3	Col4	Col5	Col6	Col7	Col8	Col9	Col10	Col11	Col12
1	Intercept			0.2029	0.06342	0.07610	0.06342	0.1015	0.1015	0.03805	0.02537	0.03805	0.03805	0.02537	0.03805
2	a	1		0.06342	0.06342			0.03805	0.02537	0.03805	0.02537				
3	a	2		0.07610		0.07610		0.03805	0.03805			0.03805	0.03805		
4	a	3		0.06342			0.06342	0.02537	0.03805					0.02537	0.03805
5	b		1	0.1015	0.03805	0.03805	0.02537	0.1022		0.03805		0.03805		0.02537	
6	b		2	0.1015	0.02537	0.03805	0.03805		0.1022		0.02537		0.03805		0.03805
7	a*b	1	1	0.03805	0.03805			0.03805		0.07515					
8	a*b	1	2	0.02537	0.02537				0.02537		0.06246				
9	a*b	2	1	0.03805		0.03805		0.03805				0.07515			
10	a*b	2	2	0.03805		0.03805			0.03805				0.07515		
11	a*b	3	1	0.02537			0.02537	0.02537						0.06246	
12	a*b	3	2	0.03805			0.03805		0.03805						0.07515

Mixed Model Equations	
Row	Col13
1	36.4143
2	13.8757
3	12.7469
4	9.7917
5	21.2956
6	15.1187
7	9.3477
8	4.5280
9	7.2676
10	5.4793
11	4.6802
12	5.1115

The solution matrix in Output 84.3.10 results from sweeping all but the last row of the mixed model equations matrix. The final column contains a solution vector for the fixed and random effects. The first four rows correspond to fixed effects and the last eight correspond to random effects.

Output 84.3.10 Solutions of the Mixed Model Equations

Mixed Model Equations Solution														
Row	Effect	a	b	Col1	Col2	Col3	Col4	Col5	Col6	Col7	Col8	Col9	Col10	Col11
1	Intercept			761.84	-29.7718	-29.6578		-731.14	-733.22	-0.4680	0.4680	-0.5257	0.5257	-12.4663
2	a	1		-29.7718	59.5436	29.7718		-2.0764	2.0764	-14.0239	-12.9342	1.0514	-1.0514	12.9342
3	a	2		-29.6578	29.7718	56.2773		-1.0382	1.0382	0.4680	-0.4680	-12.9534	-14.0048	12.4663
4	a	3												
5	b		1	-731.14	-2.0764	-1.0382		741.63	722.73	-4.2598	4.2598	-4.7855	4.7855	-4.2598
6	b		2	-733.22	2.0764	1.0382		722.73	741.63	4.2598	-4.2598	4.7855	-4.7855	4.2598
7	a*b	1	1	-0.4680	-14.0239	0.4680		-4.2598	4.2598	22.8027	4.1555	2.1570	-2.1570	1.9200
8	a*b	1	2	0.4680	-12.9342	-0.4680		4.2598	-4.2598	4.1555	22.8027	-2.1570	2.1570	-1.9200
9	a*b	2	1	-0.5257	1.0514	-12.9534		-4.7855	4.7855	2.1570	-2.1570	22.5560	4.4021	2.1570
10	a*b	2	2	0.5257	-1.0514	-14.0048		4.7855	-4.7855	-2.1570	2.1570	4.4021	22.5560	-2.1570
11	a*b	3	1	-12.4663	12.9342	12.4663		-4.2598	4.2598	1.9200	-1.9200	2.1570	-2.1570	22.8027
12	a*b	3	2	-14.4918	14.0239	14.4918		4.2598	-4.2598	-1.9200	1.9200	-2.1570	2.1570	4.1555

Mixed Model Equations Solution		
Row	Col12	Col13
1	-14.4918	159.61
2	14.0239	53.2049
3	14.4918	7.8856
4		
5	4.2598	26.8837
6	-4.2598	-26.8837
7	-1.9200	3.0198
8	1.9200	-3.0198
9	-2.1570	-1.7134
10	2.1570	1.7134
11	4.1555	-0.8115
12	22.8027	0.8115

The A factor is significant at the 5% level (Output 84.3.11).

Output 84.3.11 Tests of Fixed Effects

Type 3 Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
a	2	2	28.00	0.0345

Output 84.3.12 shows that the significance of A appears to be from the difference between its first level and its other two levels.

Output 84.3.12 Least Squares Means for A Effect

Least Squares Means						
Effect	a	Estimate	Standard Error	DF	t Value	Pr > |t|
a	1	212.82	27.6014	2	7.71	0.0164
a	2	167.50	27.5463	2	6.08	0.0260
a	3	159.61	27.6014	2	5.78	0.0286

Output 84.3.13 lists the predicted values from the model. These values are the sum of the fixed-effects estimates and the empirical best linear unbiased predictors (EBLUPs) of the random effects.

Output 84.3.13 Predicted Values

Obs	a	b	y	Pred	StdErrPred	DF	Alpha	Lower	Upper	Resid
1	1	1	237	242.723	4.72563	10	0.05	232.193	253.252	-5.7228
2	1	1	254	242.723	4.72563	10	0.05	232.193	253.252	11.2772
3	1	1	246	242.723	4.72563	10	0.05	232.193	253.252	3.2772
4	1	2	178	182.916	5.52589	10	0.05	170.603	195.228	-4.9159
5	1	2	179	182.916	5.52589	10	0.05	170.603	195.228	-3.9159
6	2	1	208	192.670	4.70076	10	0.05	182.196	203.144	15.3297
7	2	1	178	192.670	4.70076	10	0.05	182.196	203.144	-14.6703
8	2	1	187	192.670	4.70076	10	0.05	182.196	203.144	-5.6703
9	2	2	146	142.330	4.70076	10	0.05	131.856	152.804	3.6703
10	2	2	145	142.330	4.70076	10	0.05	131.856	152.804	2.6703
11	2	2	141	142.330	4.70076	10	0.05	131.856	152.804	-1.3297
12	3	1	186	185.687	5.52589	10	0.05	173.374	197.999	0.3134
13	3	1	183	185.687	5.52589	10	0.05	173.374	197.999	-2.6866
14	3	2	142	133.542	4.72563	10	0.05	123.013	144.072	8.4578
15	3	2	125	133.542	4.72563	10	0.05	123.013	144.072	-8.5422
16	3	2	136	133.542	4.72563	10	0.05	123.013	144.072	2.4578

To plot the likelihood surface by using ODS Graphics, use the following statements:

```
proc template;
   define statgraph surface;
      begingraph;
         layout overlay3d;
            surfaceplotparm x=CovP1 y=CovP2 z=ResLogLike;
         endlayout;
      endgraph;
   end;
run;
proc sgrender data=parms template=surface;
run;
```

The results from this plot are shown in Output 84.3.14. The peak of the surface is the REML estimates for the B and A*B variance components.

Output 84.3.14 Plot of Likelihood Surface

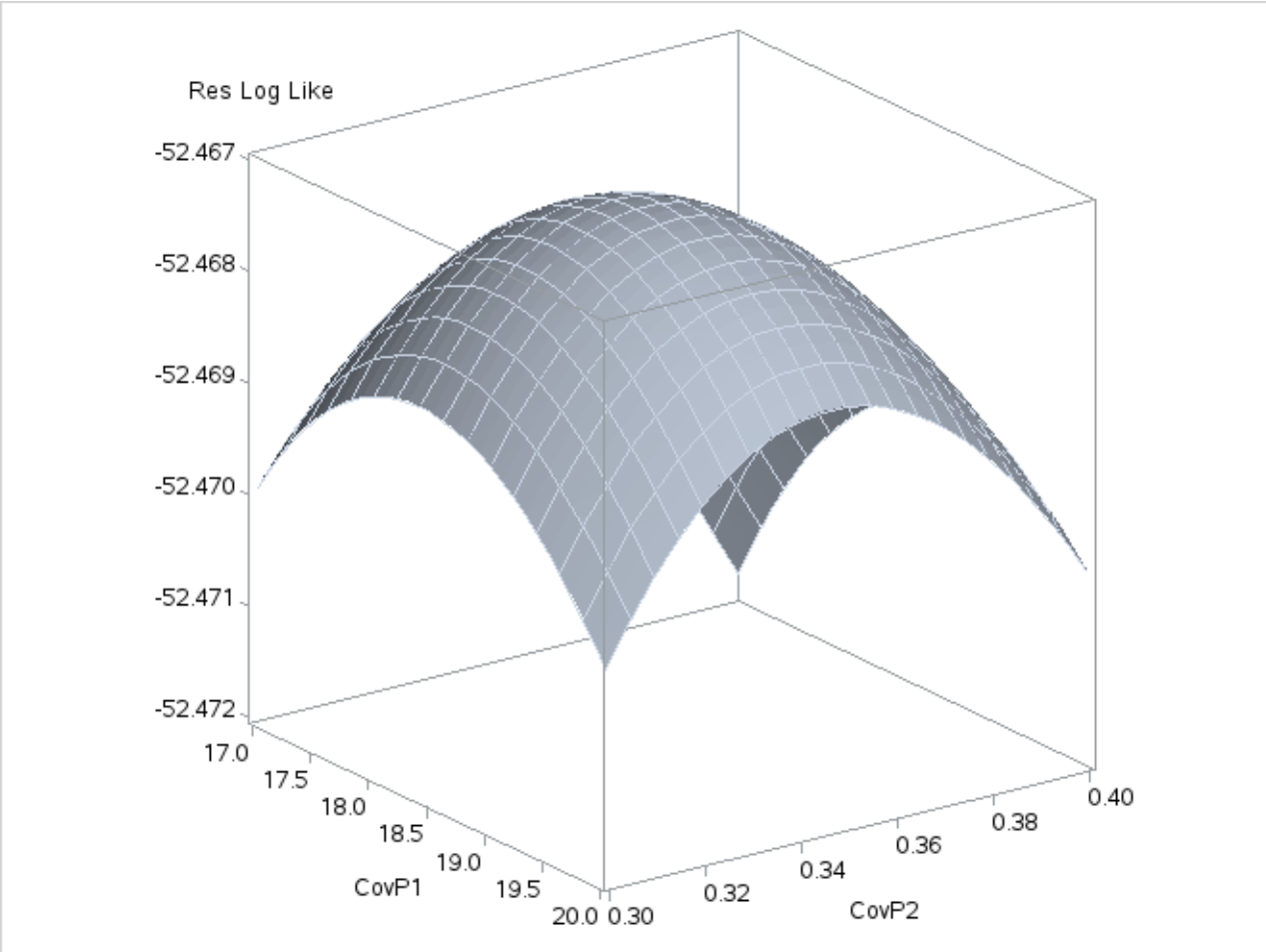

Example 84.4: Known G and R

(View the complete code for this example (mixex4.sas) in the example repository.)

This animal breeding example from Henderson (1984, p. 48) considers multiple traits. The data are artificial and consist of measurements of two traits on three animals, but the second trait of the third animal is missing. Assuming an additive genetic model, you can use PROC MIXED to predict the breeding value of both traits on all three animals and also to predict the second trait of the third animal. The data are as follows:

```
data h;
   input Trait Animal Y;
   datalines;
1 1 6
1 2 8
1 3 7
2 1 9
2 2 5
2 3 .
;
```

Both **G** and **R** are known.

$$\mathbf{G} = \begin{bmatrix} 2 & 1 & 1 & 2 & 1 & 1 \\ 1 & 2 & .5 & 1 & 2 & .5 \\ 1 & .5 & 2 & 1 & .5 & 2 \\ 2 & 1 & 1 & 3 & 1.5 & 1.5 \\ 1 & 2 & .5 & 1.5 & 3 & .75 \\ 1 & .5 & 2 & 1.5 & .75 & 3 \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} 4 & 0 & 0 & 1 & 0 & 0 \\ 0 & 4 & 0 & 0 & 1 & 0 \\ 0 & 0 & 4 & 0 & 0 & 1 \\ 1 & 0 & 0 & 5 & 0 & 0 \\ 0 & 1 & 0 & 0 & 5 & 0 \\ 0 & 0 & 1 & 0 & 0 & 5 \end{bmatrix}$$

In order to read **G** into PROC MIXED by using the GDATA= option in the RANDOM statement, perform the following DATA step:

```
data g;
   input Row Col1-Col6;
   datalines;
1  2  1  1  2   1     1
2  1  2 .5  1   2    .5
3  1 .5  2  1    .5  2
4  2  1  1  3   1.5  1.5
5  1  2 .5  1.5 3     .75
6  1 .5  2  1.5  .75 3
;
```

The preceding data are in the dense representation for a GDATA= data set. You can also construct a data set with the sparse representation by using Row, Col, and Value variables, although this would require 21 observations instead of 6 for this example.

The PROC MIXED statements are as follows:

```
proc mixed data=h mmeq mmeqsol;
   class Trait Animal;
   model Y = Trait / noint s outp=predicted;
   random Trait*Animal / type=un gdata=g g gi s;
   repeated / type=un sub=Animal r ri;
   parms (4) (1) (5) / noiter;
run;
proc print data=predicted;
run;
```

The MMEQ and MMEQSOL options request the mixed model equations and their solution. The variables Trait and Animal are classification variables, and Trait defines the entire **X** matrix for the fixed-effects portion of the model, since the intercept is omitted with the NOINT option. The fixed-effects solution vector and predicted values are also requested by using the S and OUTP= options, respectively.

The random effect Trait*Animal leads to a **Z** matrix with six columns, the first five corresponding to the identity matrix and the last consisting of 0s. An unstructured **G** matrix is specified by using the TYPE=UN

option, and it is read into PROC MIXED from a SAS data set by using the GDATA=G specification. The G and GI options request the display of $\mathbf{G}$ and $\mathbf{G}^{-1}$, respectively. The S option requests that the random-effects solution vector be displayed.

Note that the preceding $\mathbf{R}$ matrix is block diagonal if the data are sorted by animals. The REPEATED statement exploits this fact by requesting $\mathbf{R}$ to have unstructured 2×2 blocks corresponding to animals, which are the subjects. The R and RI options request that the estimated 2×2 blocks for the first animal and its inverse be displayed. The PARMS statement lists the parameters of this 2×2 matrix. Note that the parameters from $\mathbf{G}$ are not specified in the PARMS statement because they have already been assigned by using the GDATA= option in the RANDOM statement. The NOITER option prevents PROC MIXED from computing residual (restricted) maximum likelihood estimates; instead, the known values are used for inferences.

The results from this analysis are shown in Output 84.4.1–Output 84.4.12.

The "Unstructured" covariance structure (Output 84.4.1) applies to both $\mathbf{G}$ and $\mathbf{R}$ here. The levels of Trait and Animal have been specified correctly.

Output 84.4.1 Model and Class Level Information

The Mixed Procedure

Model Information	
Data Set	WORK.H
Dependent Variable	Y
Covariance Structure	Unstructured
Subject Effect	Animal
Estimation Method	REML
Residual Variance Method	None
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

Class Level Information		
Class	**Levels**	**Values**
Trait	2	1 2
Animal	3	1 2 3

The three covariance parameters indicated in Output 84.4.2 correspond to those from the $\mathbf{R}$ matrix. Those from $\mathbf{G}$ are considered fixed and known because of the GDATA= option.

Output 84.4.2 Model Dimensions and Number of Observations

Dimensions	
Covariance Parameters	3
Columns in X	2
Columns in Z	6
Subjects	1
Max Obs per Subject	5

Output 84.4.2 *continued*

Number of Observations	
Number of Observations Read	6
Number of Observations Used	5
Number of Observations Not Used	1

Because starting values for the covariance parameters are specified in the PARMS statement, the MIXED procedure prints the residual (restricted) log likelihood at the starting values. Because of the NOITER option in the PARMS statement, this is also the final log likelihood in this analysis (Output 84.4.3).

Output 84.4.3 REML Log Likelihood

Parameter Search				
CovP1	CovP2	CovP3	Res Log Like	-2 Res Log Like
4.0000	1.0000	5.0000	-7.3731	14.7463

The block of **R** corresponding to the first animal and the inverse of this block are shown in Output 84.4.4.

Output 84.4.4 Inverse R Matrix

Estimated R Matrix for Animal 1		
Row	Col1	Col2
1	4.0000	1.0000
2	1.0000	5.0000

Estimated Inv(R) Matrix for Animal 1		
Row	Col1	Col2
1	0.2632	-0.05263
2	-0.05263	0.2105

The **G** matrix as specified in the GDATA= data set and its inverse are shown in Output 84.4.5 and Output 84.4.6.

Output 84.4.5 G Matrix

Estimated G Matrix									
Row	Effect	Trait	Animal	Col1	Col2	Col3	Col4	Col5	Col6
1	Trait*Animal	1	1	2.0000	1.0000	1.0000	2.0000	1.0000	1.0000
2	Trait*Animal	1	2	1.0000	2.0000	0.5000	1.0000	2.0000	0.5000
3	Trait*Animal	1	3	1.0000	0.5000	2.0000	1.0000	0.5000	2.0000
4	Trait*Animal	2	1	2.0000	1.0000	1.0000	3.0000	1.5000	1.5000
5	Trait*Animal	2	2	1.0000	2.0000	0.5000	1.5000	3.0000	0.7500
6	Trait*Animal	2	3	1.0000	0.5000	2.0000	1.5000	0.7500	3.0000

Output 84.4.6 Inverse G Matrix

Estimated Inv(G) Matrix									
Row	**Effect**	**Trait**	**Animal**	**Col1**	**Col2**	**Col3**	**Col4**	**Col5**	**Col6**
1	Trait*Animal	1	1	2.5000	-1.0000	-1.0000	-1.6667	0.6667	0.6667
2	Trait*Animal	1	2	-1.0000	2.0000		0.6667	-1.3333	
3	Trait*Animal	1	3	-1.0000		2.0000	0.6667		-1.3333
4	Trait*Animal	2	1	-1.6667	0.6667	0.6667	1.6667	-0.6667	-0.6667
5	Trait*Animal	2	2	0.6667	-1.3333		-0.6667	1.3333	
6	Trait*Animal	2	3	0.6667		-1.3333	-0.6667		1.3333

The table of covariance parameter estimates in Output 84.4.7 displays only the parameters in **R**. Because of the GDATA= option in the RANDOM statement, the G-side parameters do not participate in the parameter estimation process. Because of the NOITER option in the PARMS statement, however, the R-side parameters in this output are identical to their starting values.

Output 84.4.7 R-Side Covariance Parameters

Covariance Parameter Estimates		
Cov Parm	**Subject**	**Estimate**
UN(1,1)	Animal	4.0000
UN(2,1)	Animal	1.0000
UN(2,2)	Animal	5.0000

The coefficients of the mixed model equations in Output 84.4.8 agree with Henderson (1984, p. 55). Recall from Output 84.4.1 that there are 2 columns in **X** and 6 columns in **Z**. The first 8 columns of the mixed model equations correspond to the **X** and **Z** components. Column 9 represents the Y border.

Output 84.4.8 Mixed Model Equations with Y Border

Mixed Model Equations												
Row	**Effect**	**Trait**	**Animal**	**Col1**	**Col2**	**Col3**	**Col4**	**Col5**	**Col6**	**Col7**	**Col8**	**Col9**
1	Trait	1		0.7763	-0.1053	0.2632	0.2632	0.2500	-0.05263	-0.05263		4.6974
2	Trait	2		-0.1053	0.4211	-0.05263	-0.05263		0.2105	0.2105		2.2105
3	Trait*Animal	1	1	0.2632	-0.05263	2.7632	-1.0000	-1.0000	-1.7193	0.6667	0.6667	1.1053
4	Trait*Animal	1	2	0.2632	-0.05263	-1.0000	2.2632		0.6667	-1.3860		1.8421
5	Trait*Animal	1	3	0.2500		-1.0000		2.2500	0.6667		-1.3333	1.7500
6	Trait*Animal	2	1	-0.05263	0.2105	-1.7193	0.6667	0.6667	1.8772	-0.6667	-0.6667	1.5789
7	Trait*Animal	2	2	-0.05263	0.2105	0.6667	-1.3860		-0.6667	1.5439		0.6316
8	Trait*Animal	2	3			0.6667		-1.3333	-0.6667		1.3333	

The solution to the mixed model equations also matches that given by Henderson (1984, p. 55). After solving the augmented mixed model equations, you can find the solutions for fixed and random effects in the last column (Output 84.4.9).

Output 84.4.9 Solutions of the Mixed Model Equations with Y Border

Mixed Model Equations Solution												
Row	Effect	Trait	Animal	Col1	Col2	Col3	Col4	Col5	Col6	Col7	Col8	Col9
1	Trait	1		2.5508	1.5685	-1.3047	-1.1775	-1.1701	-1.3002	-1.1821	-1.1678	6.9909
2	Trait	2		1.5685	4.5539	-1.4112	-1.3534	-0.9410	-2.1592	-2.1055	-1.3149	6.9959
3	Trait*Animal	1	1	-1.3047	-1.4112	1.8282	1.0652	1.0206	1.8010	1.0925	1.0070	0.05450
4	Trait*Animal	1	2	-1.1775	-1.3534	1.0652	1.7589	0.7085	1.0900	1.7341	0.7209	-0.04955
5	Trait*Animal	1	3	-1.1701	-0.9410	1.0206	0.7085	1.7812	1.0095	0.7197	1.7756	0.02230
6	Trait*Animal	2	1	-1.3002	-2.1592	1.8010	1.0900	1.0095	2.7518	1.6392	1.4849	0.2651
7	Trait*Animal	2	2	-1.1821	-2.1055	1.0925	1.7341	0.7197	1.6392	2.6874	0.9930	-0.2601
8	Trait*Animal	2	3	-1.1678	-1.3149	1.0070	0.7209	1.7756	1.4849	0.9930	2.7645	0.1276

The solutions for the fixed and random effects in Output 84.4.10 correspond to the last column in Output 84.4.9. Note that the standard errors for the fixed effects and the prediction standard errors for the random effects are the square root values of the diagonal entries in the solution of the mixed model equations (Output 84.4.9).

Output 84.4.10 Solutions for Fixed and Random Effects

Solution for Fixed Effects						
Effect	Trait	Estimate	Standard Error	DF	t Value	Pr > |t|
Trait	1	6.9909	1.5971	3	4.38	0.0221
Trait	2	6.9959	2.1340	3	3.28	0.0465

Solution for Random Effects							
Effect	Trait	Animal	Estimate	Std Err Pred	DF	t Value	Pr > |t|
Trait*Animal	1	1	0.05450	1.3521	0	0.04	.
Trait*Animal	1	2	-0.04955	1.3262	0	-0.04	.
Trait*Animal	1	3	0.02230	1.3346	0	0.02	.
Trait*Animal	2	1	0.2651	1.6589	0	0.16	.
Trait*Animal	2	2	-0.2601	1.6393	0	-0.16	.
Trait*Animal	2	3	0.1276	1.6627	0	0.08	.

The estimates for the two traits are nearly identical, but the standard error of the second trait is larger because of the missing observation.

The Estimate column in the "Solution for Random Effects" table lists the best linear unbiased predictions (BLUPs) of the breeding values of both traits for all three animals. The p-values are missing because the default containment method for computing degrees of freedom results in zero degrees of freedom for the random effects parameter tests.

Output 84.4.11 Significance Test Comparing Traits

Type 3 Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
Trait	2	3	10.59	0.0437

The two estimated traits are significantly different from zero at the 5% level (Output 84.4.11).

Output 84.4.12 displays the predicted values of the observations based on the trait and breeding value estimates—that is, the fixed and random effects.

Output 84.4.12 Predicted Observations

Obs	Trait	Animal	Y	Pred	StdErrPred	DF	Alpha	Lower	Upper	Resid
1	1	1	6	7.04542	1.33027	0	0.05	.	.	-1.04542
2	1	2	8	6.94137	1.39806	0	0.05	.	.	1.05863
3	1	3	7	7.01321	1.41129	0	0.05	.	.	-0.01321
4	2	1	9	7.26094	1.72839	0	0.05	.	.	1.73906
5	2	2	5	6.73576	1.74077	0	0.05	.	.	-1.73576
6	2	3	.	7.12015	2.99088	0	0.05	.	.	.

The predicted values are not the predictions of future records in the sense that they do not contain a component corresponding to a new observational error. See Henderson (1984) for information about predicting future records. The Lower and Upper columns usually contain confidence limits for the predicted values; they are missing here because the random-effects parameter degrees of freedom equals 0.

Example 84.5: Random Coefficients

(View the complete code for this example (mixex5.sas) in the example repository.)

This example comes from a pharmaceutical stability data simulation performed by Obenchain (1990). The observed responses are replicate assay results, expressed in percent of label claim, at various shelf ages, expressed in months. The desired mixed model involves three batches of product that differ randomly in intercept (initial potency) and slope (degradation rate). This type of model is also known as a hierarchical or multilevel model (Singer 1998; Sullivan, Dukes, and Losina 1999).

The SAS statements are as follows:

```
data rc;
   input Batch Month @@;
   Monthc = Month;
   do i = 1 to 6;
      input Y @@;
      output;
   end;
   datalines;
 1   0  101.2 103.3 103.3 102.1 104.4 102.4
 1   1   98.8  99.4  99.7  99.5    .     .
 1   3   98.4  99.0  97.3  99.8    .     .
 1   6  101.5 100.2 101.7 102.7    .     .
 1   9   96.3  97.2  97.2  96.3    .     .
 1  12   97.3  97.9  96.8  97.7  97.7  96.7
 2   0  102.6 102.7 102.4 102.1 102.9 102.6
 2   1   99.1  99.0  99.9 100.6    .     .
 2   3  105.7 103.3 103.4 104.0    .     .
 2   6  101.3 101.5 100.9 101.4    .     .
 2   9   94.1  96.5  97.2 95.6     .     .
```

```
 2  12   93.1  92.8  95.4 92.2   92.2  93.0
 3   0  105.1 103.9 106.1 104.1 103.7 104.6
 3   1  102.2 102.0 100.8  99.8    .     .
 3   3  101.2 101.8 100.8 102.6    .     .
 3   6  101.1 102.0 100.1 100.2    .     .
 3   9  100.9  99.5 102.2 100.8    .     .
 3  12   97.8  98.3  96.9  98.4  96.9  96.5
;

proc mixed data=rc;
   class Batch;
   model Y = Month / s;
   random Int Month / type=un sub=Batch s;
run;
```

In the DATA step, Monthc is created as a duplicate of Month in order to enable both a continuous and a classification version of the same variable. The variable Monthc is used in a subsequent analysis

In the PROC MIXED statements, Batch is listed as the only classification variable. The fixed effect Month in the MODEL statement is not declared as a classification variable; thus it models a linear trend in time. An intercept is included as a fixed effect by default, and the S option requests that the fixed-effects parameter estimates be produced.

The two random effects are Int and Month, modeling random intercepts and slopes, respectively. Note that Intercept and Month are used as both fixed and random effects. The TYPE=UN option in the RANDOM statement specifies an unstructured covariance matrix for the random intercept and slope effects. In mixed model notation, **G** is block diagonal with unstructured 2×2 blocks. Each block corresponds to a different level of Batch, which is the SUBJECT= effect. The unstructured type provides a mechanism for estimating the correlation between the random coefficients. The S option requests the production of the random-effects parameter estimates.

The results from this analysis are shown in Output 84.5.1–Output 84.5.9. The "Unstructured" covariance structure in Output 84.5.1 applies to **G** here.

Output 84.5.1 Model Information in Random Coefficients Analysis

The Mixed Procedure

Model Information	
Data Set	WORK.RC
Dependent Variable	Y
Covariance Structure	Unstructured
Subject Effect	Batch
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

Batch is the only classification variable in this analysis, and it has three levels (Output 84.5.2).

Output 84.5.2 Random Coefficients Analysis (*continued*)

Class Level Information		
Class	**Levels**	**Values**
Batch	3	1 2 3

The "Dimensions" table in Output 84.5.3 indicates that there are three subjects (corresponding to batches). The 24 observations not used correspond to the missing values of Y in the input data set.

Output 84.5.3 Random Coefficients Analysis (*continued*)

Dimensions	
Covariance Parameters	4
Columns in X	2
Columns in Z per Subject	2
Subjects	3
Max Obs per Subject	28

Number of Observations	
Number of Observations Read	108
Number of Observations Used	84
Number of Observations Not Used	24

As Output 84.5.4 shows, only one iteration is required for convergence.

Output 84.5.4 Random Coefficients Analysis (*continued*)

Iteration History			
Iteration	**Evaluations**	**-2 Res Log Like**	**Criterion**
0	1	367.02768461	
1	1	350.32813577	0.00000000

Convergence criteria met.

The Estimate column in Output 84.5.5 lists the estimated elements of the unstructured 2×2 matrix comprising the blocks of **G**. Note that the random coefficients are negatively correlated.

Output 84.5.5 Random Coefficients Analysis (*continued*)

Covariance Parameter Estimates		
Cov Parm	**Subject**	**Estimate**
UN(1,1)	Batch	0.9768
UN(2,1)	Batch	-0.1045
UN(2,2)	Batch	0.03717
Residual		3.2932

The null model likelihood ratio test indicates a significant improvement over the null model consisting of no random effects and a homogeneous residual error (Output 84.5.6).

Output 84.5.6 Random Coefficients Analysis (*continued*)

Fit Statistics	
-2 Res Log Likelihood	350.3
AIC (Smaller is Better)	358.3
AICC (Smaller is Better)	358.8
BIC (Smaller is Better)	354.7

Null Model Likelihood Ratio Test		
DF	Chi-Square	Pr > ChiSq
3	16.70	0.0008

The fixed-effects estimates represent the estimated means for the random intercept and slope, respectively (Output 84.5.7).

Output 84.5.7 Random Coefficients Analysis (*continued*)

Solution for Fixed Effects

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	102.70	0.6456	2	159.08	<.0001
Month	-0.5259	0.1194	2	-4.41	0.0478

The random-effects estimates represent the estimated deviation from the mean intercept and slope for each batch (Output 84.5.8). Therefore, the intercept for the first batch is close to $102.7 - 1 = 101.7$, while the intercepts for the other two batches are greater than 102.7. The second batch has a slope less than the mean slope of –0.526, while the other two batches have slopes greater than –0.526.

Output 84.5.8 Random Coefficients Analysis (*continued*)

Solution for Random Effects

Effect	Batch	Estimate	Std Err Pred	DF	t Value	Pr > |t|
Intercept	1	-1.0010	0.6842	78	-1.46	0.1474
Month	1	0.1287	0.1245	78	1.03	0.3047
Intercept	2	0.3934	0.6842	78	0.58	0.5669
Month	2	-0.2060	0.1245	78	-1.65	0.1021
Intercept	3	0.6076	0.6842	78	0.89	0.3772
Month	3	0.07731	0.1245	78	0.62	0.5365

The F statistic in the "Type 3 Tests of Fixed Effects" table in Output 84.5.9 is the square of the t statistic used in the test of Month in the preceding "Solution for Fixed Effects" table (compare Output 84.5.7 and Output 84.5.9). Both statistics test the null hypothesis that the slope assigned to Month equals 0, and this hypothesis can barely be rejected at the 5% level.

Output 84.5.9 Random Coefficients Analysis (*continued*)

Type 3 Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
Month	1	2	19.41	0.0478

It is also possible to fit a random coefficients model with error terms that follow a nested structure (Fuller and Battese 1973). The following SAS statements represent one way of doing this:

```
proc mixed data=rc;
   class Batch Monthc;
   model Y = Month / s;
   random Int Month Monthc / sub=Batch s;
run;
```

The variable Monthc is added to the CLASS and RANDOM statements, and it models the nested errors. Note that Month and Monthc are continuous and classification versions of the same variable. Also, the TYPE=UN option is dropped from the RANDOM statement, resulting in the default variance components model instead of correlated random coefficients. The results from this analysis are shown in Output 84.5.10.

Output 84.5.10 Random Coefficients with Nested Errors Analysis

The Mixed Procedure

Model Information	
Data Set	WORK.RC
Dependent Variable	Y
Covariance Structure	Variance Components
Subject Effect	Batch
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

Class Level Information		
Class	Levels	Values
Batch	3	1 2 3
Monthc	6	0 1 3 6 9 12

Dimensions	
Covariance Parameters	4
Columns in X	2
Columns in Z per Subject	8
Subjects	3
Max Obs per Subject	28

Number of Observations	
Number of Observations Read	108
Number of Observations Used	84
Number of Observations Not Used	24

Output 84.5.10 *continued*

Iteration History			
Iteration	**Evaluations**	**-2 Res Log Like**	**Criterion**
0	1	367.02768461	
1	4	277.51945360	.
2	1	276.97551718	0.00104208
3	1	276.90304909	0.00003174
4	1	276.90100316	0.00000004
5	1	276.90100092	0.00000000

Convergence criteria met.

Covariance Parameter Estimates		
Cov Parm	**Subject**	**Estimate**
Intercept	Batch	0
Month	Batch	0.01243
Monthc	Batch	3.7411
Residual		0.7969

For this analysis, the Newton-Raphson algorithm requires five iterations and nine likelihood evaluations to achieve convergence. The missing value in the Criterion column in iteration 1 indicates that a boundary constraint has been dropped.

The estimate for the Intercept variance component equals 0. This occurs frequently in practice and indicates that the restricted likelihood is maximized by setting this variance component equal to 0. Whenever a zero variance component estimate occurs, the following note appears in the SAS log:

```
NOTE: Estimated G matrix is not positive definite.
```

The remaining variance component estimates are positive, and the estimate corresponding to the nested errors (MONTHC) is much larger than the other two.

A comparison of AIC and BIC for this model with those of the previous model favors the nested error model (compare Output 84.5.11 and Output 84.5.6). Strictly speaking, a likelihood ratio test cannot be carried out between the two models because one is not contained in the other; however, a cautious comparison of likelihoods can be informative.

Output 84.5.11 Random Coefficients with Nested Errors Analysis (*continued*)

Fit Statistics	
-2 Res Log Likelihood	276.9
AIC (Smaller is Better)	282.9
AICC (Smaller is Better)	283.2
BIC (Smaller is Better)	280.2

The better-fitting covariance model affects the standard errors of the fixed-effects parameter estimates more than the estimates themselves (Output 84.5.12).

Output 84.5.12 Random Coefficients with Nested Errors Analysis (*continued*)

Solution for Fixed Effects					
Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	102.56	0.7287	2	140.74	<.0001
Month	-0.5003	0.1259	2	-3.97	0.0579

The random-effects solution provides the empirical best linear unbiased predictions (EBLUPs) for the realizations of the random intercept, slope, and nested errors (Output 84.5.13). You can use these values to compare batches and months.

Output 84.5.13 Random Coefficients with Nested Errors Analysis (*continued*)

Solution for Random Effects							
Effect	Batch	Monthc	Estimate	Std Err Pred	DF	t Value	Pr > |t|
Intercept	1		0	.	.	.	.
Month	1		-0.00028	0.09268	66	-0.00	0.9976
Monthc	1	0	0.2191	0.7896	66	0.28	0.7823
Monthc	1	1	-2.5690	0.7571	66	-3.39	0.0012
Monthc	1	3	-2.3067	0.6865	66	-3.36	0.0013
Monthc	1	6	1.8726	0.7328	66	2.56	0.0129
Monthc	1	9	-1.2350	0.9300	66	-1.33	0.1888
Monthc	1	12	0.7736	1.1992	66	0.65	0.5211
Intercept	2		0	.	.	.	.
Month	2		-0.07571	0.09268	66	-0.82	0.4169
Monthc	2	0	-0.00621	0.7896	66	-0.01	0.9938
Monthc	2	1	-2.2126	0.7571	66	-2.92	0.0048
Monthc	2	3	3.1063	0.6865	66	4.53	<.0001
Monthc	2	6	2.0649	0.7328	66	2.82	0.0064
Monthc	2	9	-1.4450	0.9300	66	-1.55	0.1250
Monthc	2	12	-2.4405	1.1992	66	-2.04	0.0459
Intercept	3		0	.	.	.	.
Month	3		0.07600	0.09268	66	0.82	0.4152
Monthc	3	0	1.9574	0.7896	66	2.48	0.0157
Monthc	3	1	-0.8850	0.7571	66	-1.17	0.2466
Monthc	3	3	0.3006	0.6865	66	0.44	0.6629
Monthc	3	6	0.7972	0.7328	66	1.09	0.2806
Monthc	3	9	2.0059	0.9300	66	2.16	0.0347
Monthc	3	12	0.002293	1.1992	66	0.00	0.9985

Output 84.5.14 Random Coefficients with Nested Errors Analysis (*continued*)

Type 3 Tests of Fixed Effects				
Effect	Num DF	Den DF	F Value	Pr > F
Month	1	2	15.78	0.0579

The test of Month is similar to that from the previous model, although it is no longer significant at the 5% level (Output 84.5.14).

Example 84.6: Line-Source Sprinkler Irrigation

(View the complete code for this example (mixex6.sas) in the example repository.)

These data appear in Hanks et al. (1980); Johnson, Chaudhuri, and Kanemasu (1983); Stroup (1989b). Three cultivars (Cult) of winter wheat are randomly assigned to rectangular plots within each of three blocks (Block). The nine plots are located side by side, and a line-source sprinkler is placed through the middle. Each plot is subdivided into twelve subplots—six to the north of the line source, six to the south (Dir). The two plots closest to the line source represent the maximum irrigation level (Irrig=6), the two next-closest plots represent the next-highest level (Irrig=5), and so forth.

This example is a case where both **G** and **R** can be modeled. One of Stroup's models specifies a diagonal **G** containing the variance components for Block, Block*Dir, and Block*Irrig, and a Toeplitz **R** with four bands. The SAS statements to fit this model and carry out some further analyses follow.

CAUTION: This analysis can require considerable CPU time.

```
data line;
   length Cult$ 8;
   input Block Cult$ @;
   row = _n_;
   do Sbplt=1 to 12;
      if Sbplt le 6 then do;
         Irrig = Sbplt;
         Dir = 'North';
      end; else do;
         Irrig = 13 - Sbplt;
         Dir = 'South';
      end;
      input Y @; output;
   end;
   datalines;
 1 Luke     2.4 2.7 5.6 7.5 7.9 7.1 6.1 7.3 7.4 6.7 3.8 1.8
 1 Nugaines 2.2 2.2 4.3 6.3 7.9 7.1 6.2 5.3 5.3 5.2 5.4 2.9
 1 Bridger  2.9 3.2 5.1 6.9 6.1 7.5 5.6 6.5 6.6 5.3 4.1 3.1
 2 Nugaines 2.4 2.2 4.0 5.8 6.1 6.2 7.0 6.4 6.7 6.4 3.7 2.2
 2 Bridger  2.6 3.1 5.7 6.4 7.7 6.8 6.3 6.2 6.6 6.5 4.2 2.7
 2 Luke     2.2 2.7 4.3 6.9 6.8 8.0 6.5 7.3 5.9 6.6 3.0 2.0
 3 Nugaines 1.8 1.9 3.7 4.9 5.4 5.1 5.7 5.0 5.6 5.1 4.2 2.2
 3 Luke     2.1 2.3 3.7 5.8 6.3 6.3 6.5 5.7 5.8 4.5 2.7 2.3
 3 Bridger  2.7 2.8 4.0 5.0 5.2 5.2 5.9 6.1 6.0 4.3 3.1 3.1
;

proc mixed;
   class Block Cult Dir Irrig;
   model Y = Cult|Dir|Irrig@2;
   random Block Block*Dir Block*Irrig;
   repeated / type=toep(4) sub=Block*Cult r;
   lsmeans Cult|Irrig;
```

```
    estimate 'Bridger vs Luke' Cult 1 -1 0;
    estimate 'Linear Irrig' Irrig -5 -3 -1 1 3 5;
    estimate 'B vs L x Linear Irrig' Cult*Irrig
             -5 -3 -1 1 3 5 5 3 1 -1 -3 -5;
run;
```

The preceding statements use the bar operator (|) and the at sign (@) to specify all two-factor interactions between Cult, Dir, and Irrig as fixed effects.

The RANDOM statement sets up the **Z** and **G** matrices corresponding to the random effects Block, Block*Dir, and Block*Irrig.

In the REPEATED statement, the TYPE=TOEP(4) option sets up the blocks of the **R** matrix to be Toeplitz with four bands below and including the main diagonal. The subject effect is Block*Cult, and it produces nine 12×12 blocks. The R option requests that the first block of **R** be displayed.

Least squares means (LSMEANS) are requested for Cult, Irrig, and Cult*Irrig, and a few ESTIMATE statements are specified to illustrate some linear combinations of the fixed effects.

The results from this analysis are shown in Output 84.6.1.

The "Covariance Structures" row in Output 84.6.1 reveals the two different structures assumed for **G** and **R**.

Output 84.6.1 Model Information in Line-Source Sprinkler Analysis

The Mixed Procedure

Model Information	
Data Set	WORK.LINE
Dependent Variable	Y
Covariance Structures	Variance Components, Toeplitz
Subject Effect	Block*Cult
Estimation Method	REML
Residual Variance Method	Profile
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Containment

The levels of each classification variable are listed as a single string in the Values column, regardless of whether the levels are numeric or character (Output 84.6.2).

Output 84.6.2 Class Level Information

Class Level Information		
Class	**Levels**	**Values**
Block	3	1 2 3
Cult	3	Bridger Luke Nugaines
Dir	2	North South
Irrig	6	1 2 3 4 5 6

Even though there is a SUBJECT= effect in the REPEATED statement, the analysis considers all of the data to be from one subject because there is no corresponding SUBJECT= effect in the RANDOM statement (Output 84.6.3).

Output 84.6.3 Model Dimensions and Number of Observations

Dimensions	
Covariance Parameters	7
Columns in X	48
Columns in Z	27
Subjects	1
Max Obs per Subject	108

Number of Observations	
Number of Observations Read	108
Number of Observations Used	108
Number of Observations Not Used	0

The Newton-Raphson algorithm converges successfully in seven iterations (Output 84.6.4).

Output 84.6.4 Iteration History and Convergence Status

Iteration History			
Iteration	Evaluations	-2 Res Log Like	Criterion
0	1	226.25427252	
1	4	187.99336173	.
2	3	186.62579299	0.10431081
3	1	184.38218213	0.04807260
4	1	183.41836853	0.00886548
5	1	183.25111475	0.00075353
6	1	183.23809997	0.00000748
7	1	183.23797748	0.00000000

Convergence criteria met.

The first block of the estimated **R** matrix has the TOEP(4) structure, and the observations that are three plots apart exhibit a negative correlation (Output 84.6.5).

Output 84.6.5 Estimated R Matrix for the First Subject

Estimated R Matrix for Block*Cult 1 Bridger												
Row	Col1	Col2	Col3	Col4	Col5	Col6	Col7	Col8	Col9	Col10	Col11	Col12
1	0.2850	0.007986	0.001452	-0.09253								
2	0.007986	0.2850	0.007986	0.001452	-0.09253							
3	0.001452	0.007986	0.2850	0.007986	0.001452	-0.09253						
4	-0.09253	0.001452	0.007986	0.2850	0.007986	0.001452	-0.09253					
5		-0.09253	0.001452	0.007986	0.2850	0.007986	0.001452	-0.09253				
6			-0.09253	0.001452	0.007986	0.2850	0.007986	0.001452	-0.09253			
7				-0.09253	0.001452	0.007986	0.2850	0.007986	0.001452	-0.09253		
8					-0.09253	0.001452	0.007986	0.2850	0.007986	0.001452	-0.09253	
9						-0.09253	0.001452	0.007986	0.2850	0.007986	0.001452	-0.09253
10							-0.09253	0.001452	0.007986	0.2850	0.007986	0.001452
11								-0.09253	0.001452	0.007986	0.2850	0.007986
12									-0.09253	0.001452	0.007986	0.2850

Output 84.6.6 lists the estimated covariance parameters from both **G** and **R**. The first three are the variance components making up the diagonal **G**, and the final four make up the Toeplitz structure in the blocks of **R**. The Residual row corresponds to the variance of the Toeplitz structure, and it represents the parameter profiled out during the optimization process.

Output 84.6.6 Estimated Covariance Parameters

Covariance Parameter Estimates		
Cov Parm	**Subject**	**Estimate**
Block		0.2194
Block*Dir		0.01768
Block*Irrig		0.03539
TOEP(2)	Block*Cult	0.007986
TOEP(3)	Block*Cult	0.001452
TOEP(4)	Block*Cult	-0.09253
Residual		0.2850

The "–2 Res Log Likelihood" value in Output 84.6.7 is the same as the final value listed in the "Iteration History" table (Output 84.6.4).

Output 84.6.7 Fit Statistics Based on the Residual Log Likelihood

Fit Statistics	
-2 Res Log Likelihood	183.2
AIC (Smaller is Better)	197.2
AICC (Smaller is Better)	198.8
BIC (Smaller is Better)	190.9

Every fixed effect except for Dir and Cult*Irrig is significant at the 5% level (Output 84.6.8).

Output 84.6.8 Tests for Fixed Effects

Type 3 Tests of Fixed Effects				
Effect	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Cult	2	68	7.98	0.0008
Dir	1	2	3.95	0.1852
Cult*Dir	2	68	3.44	0.0379
Irrig	5	10	102.60	<.0001
Cult*Irrig	10	68	1.91	0.0580
Dir*Irrig	5	68	6.12	<.0001

The "Estimates" table lists the results from the various linear combinations of fixed effects specified in the ESTIMATE statements (Output 84.6.9). Bridger is not significantly different from Luke, and Irrig possesses a strong linear component. This strength appears to be influencing the significance of the interaction.

Output 84.6.9 Estimates

Estimates					
Label	Estimate	Standard Error	DF	t Value	Pr > |t|
Bridger vs Luke	-0.03889	0.09524	68	-0.41	0.6843
Linear Irrig	30.6444	1.4412	10	21.26	<.0001
B vs L x Linear Irrig	-9.8667	2.7400	68	-3.60	0.0006

The least squares means shown in Output 84.6.10 are useful in comparing the levels of the various fixed effects. For example, it appears that irrigation levels 5 and 6 have virtually the same effect.

Output 84.6.10 Least Squares Means for Cult, Irrig, and Their Interaction

Least Squares Means							
Effect	Cult	Irrig	Estimate	Standard Error	DF	t Value	Pr > |t|
Cult	Bridger		5.0306	0.2874	68	17.51	<.0001
Cult	Luke		5.0694	0.2874	68	17.64	<.0001
Cult	Nugaines		4.7222	0.2874	68	16.43	<.0001
Irrig		1	2.4222	0.3220	10	7.52	<.0001
Irrig		2	3.1833	0.3220	10	9.88	<.0001
Irrig		3	5.0556	0.3220	10	15.70	<.0001
Irrig		4	6.1889	0.3220	10	19.22	<.0001
Irrig		5	6.4000	0.3140	10	20.38	<.0001
Irrig		6	6.3944	0.3227	10	19.81	<.0001
Cult*Irrig	Bridger	1	2.8500	0.3679	68	7.75	<.0001
Cult*Irrig	Bridger	2	3.4167	0.3679	68	9.29	<.0001
Cult*Irrig	Bridger	3	5.1500	0.3679	68	14.00	<.0001
Cult*Irrig	Bridger	4	6.2500	0.3679	68	16.99	<.0001
Cult*Irrig	Bridger	5	6.3000	0.3463	68	18.19	<.0001
Cult*Irrig	Bridger	6	6.2167	0.3697	68	16.81	<.0001
Cult*Irrig	Luke	1	2.1333	0.3679	68	5.80	<.0001
Cult*Irrig	Luke	2	2.8667	0.3679	68	7.79	<.0001
Cult*Irrig	Luke	3	5.2333	0.3679	68	14.22	<.0001
Cult*Irrig	Luke	4	6.5500	0.3679	68	17.80	<.0001
Cult*Irrig	Luke	5	6.8833	0.3463	68	19.87	<.0001
Cult*Irrig	Luke	6	6.7500	0.3697	68	18.26	<.0001
Cult*Irrig	Nugaines	1	2.2833	0.3679	68	6.21	<.0001
Cult*Irrig	Nugaines	2	3.2667	0.3679	68	8.88	<.0001
Cult*Irrig	Nugaines	3	4.7833	0.3679	68	13.00	<.0001
Cult*Irrig	Nugaines	4	5.7667	0.3679	68	15.67	<.0001
Cult*Irrig	Nugaines	5	6.0167	0.3463	68	17.37	<.0001
Cult*Irrig	Nugaines	6	6.2167	0.3697	68	16.81	<.0001

An interesting exercise is to fit other variance-covariance models to these data and to compare them to this one by using likelihood ratio tests, Akaike's information criterion, or Schwarz's Bayesian information criterion. In particular, some spatial models are worth investigating (Marx and Thompson 1987; Zimmerman and Harville 1991). The following is one example of spatial model statements:

```
proc mixed;
   class Block Cult Dir Irrig;
   model Y = Cult|Dir|Irrig@2;
   repeated / type=sp(pow)(Row Sbplt) sub=intercept;
run;
```

The TYPE=SP(POW)(Row Sbplt) option in the REPEATED statement requests the spatial power structure, with the two defining coordinate variables being Row and Sbplt. The SUBJECT=INTERCEPT option indicates that the entire data set is to be considered as one subject, thereby modeling $\mathbf{R}$ as a dense 108×108 covariance matrix. See Wolfinger (1993) for further discussion of this example and additional analyses.

Example 84.7: Influence in Heterogeneous Variance Model

(View the complete code for this example (mixex7.sas) in the example repository.)

In this example from Snedecor and Cochran (1980, p. 216), a one-way classification model with heterogeneous variances is fit. The data, shown in the following DATA step, represent amounts of different types of fat absorbed by batches of doughnuts during cooking, measured in grams.

```
data absorb;
   input FatType Absorbed @@;
   datalines;
 1 164  1 172  1 168  1 177  1 156  1 195
 2 178  2 191  2 197  2 182  2 185  2 177
 3 175  3 193  3 178  3 171  3 163  3 176
 4 155  4 166  4 149  4 164  4 170  4 168
;
```

The statistical model for these data can be written as

$$
\begin{aligned}
Y_{ij} &= \mu + \tau_i + \epsilon_{ij} \\
i &= 1,\ldots,t = 4 \\
j &= 1,\ldots,r = 6 \\
\epsilon_{ij} &= N(0,\sigma_i^2)
\end{aligned}
$$

where Y_{ij} is the amount of fat absorbed by the jth batch of the ith fat type, and τ_i denotes the fat-type effects. A quick glance at the data suggests that observations 6, 9, 14, and 21 might be influential on the analysis, because these are extreme observations for the respective fat types.

The following SAS statements fit this model and request influence diagnostics for the fixed effects and covariance parameters. ODS Graphics is used to create plots of the influence diagnostics in addition to the tabular output. The ESTIMATES suboption requests plots of "leave-one-out" estimates for the fixed effects and group variances.

```
ods graphics on;

proc mixed data=absorb asycov;
   class FatType;
   model Absorbed = FatType / s
                    influence(iter=10 estimates);
```

```
   repeated / group=FatType;
   ods output Influence=inf;
run;

ods graphics off;
```

The "Influence" table is output to the SAS data set inf so that parameter estimates can be printed subsequently. Results from this analysis are shown in Output 84.7.1.

Output 84.7.1 Heterogeneous Variance Analysis

The Mixed Procedure

Model Information	
Data Set	WORK.ABSORB
Dependent Variable	Absorbed
Covariance Structure	Variance Components
Group Effect	FatType
Estimation Method	REML
Residual Variance Method	None
Fixed Effects SE Method	Model-Based
Degrees of Freedom Method	Between-Within

Covariance Parameter Estimates

Cov Parm	Group	Estimate
Residual	FatType 1	178.00
Residual	FatType 2	60.4000
Residual	FatType 3	97.6000
Residual	FatType 4	67.6000

Solution for Fixed Effects

Effect	FatType	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept		162.00	3.3566	20	48.26	<.0001
FatType	1	10.0000	6.3979	20	1.56	0.1337
FatType	2	23.0000	4.6188	20	4.98	<.0001
FatType	3	14.0000	5.2472	20	2.67	0.0148
FatType	4	0	.	.	.	.

The fixed-effects solutions correspond to estimates of the following parameters:

$$\begin{aligned}
\text{Intercept} &: \mu + \tau_4 \\
\text{FatType1} &: \tau_1 - \tau_4 \\
\text{FatType2} &: \tau_2 - \tau_4 \\
\text{FatType3} &: \tau_3 - \tau_4 \\
\text{FatType4} &: 0
\end{aligned}$$

You can easily verify that these estimates are simple functions of the arithmetic means $\overline{y}_{i.}$ in the groups. For example, $\widehat{\mu + \tau_4} = \overline{y}_{4.} = 162.0$, $\widehat{\tau_1 - \tau_4} = \overline{y}_{1.} - \overline{y}_{4.} = 10.0$, and so forth. The covariance parameter estimates are the sample variances in the groups and are uncorrelated.

The variances in the four groups are shown in the "Covariance Parameter Estimates" table (Output 84.7.1). The estimated variance in the first group is two to three times larger than the variance in the other groups.

Output 84.7.2 Asymptotic Variances of Group Variance Estimates

Asymptotic Covariance Matrix of Estimates					
Row	**Cov Parm**	**CovP1**	**CovP2**	**CovP3**	**CovP4**
1	Residual	12674			
2	Residual		1459.26		
3	Residual			3810.30	
4	Residual				1827.90

In groups where the residual variance estimate is large, the precision of the estimate is also small (Output 84.7.2).

The following statements print the "leave-one-out" estimates for fixed effects and covariance parameters that were written to the inf data set with the ESTIMATES suboption (Output 84.7.3):

```
proc print data=inf label;
   var parm1-parm5 covp1-covp4;
run;
```

Output 84.7.3 Leave-One-Out Estimates

Obs	Intercept	FatType 1	FatType 2	FatType 3	FatType 4	Residual FatType 1	Residual FatType 2	Residual FatType 3	Residual FatType 4
1	162.00	11.600	23.000	14.000	0	203.30	60.400	97.60	67.600
2	162.00	10.000	23.000	14.000	0	222.47	60.400	97.60	67.600
3	162.00	10.800	23.000	14.000	0	217.68	60.400	97.60	67.600
4	162.00	9.000	23.000	14.000	0	214.99	60.400	97.60	67.600
5	162.00	13.200	23.000	14.000	0	145.70	60.400	97.60	67.600
6	162.00	5.400	23.000	14.000	0	63.80	60.400	97.60	67.600
7	162.00	10.000	24.400	14.000	0	178.00	60.795	97.60	67.600
8	162.00	10.000	21.800	14.000	0	178.00	64.691	97.60	67.600
9	162.00	10.000	20.600	14.000	0	178.00	32.296	97.60	67.600
10	162.00	10.000	23.600	14.000	0	178.00	72.797	97.60	67.600
11	162.00	10.000	23.000	14.000	0	178.00	75.490	97.60	67.600
12	162.00	10.000	24.600	14.000	0	178.00	56.285	97.60	67.600
13	162.00	10.000	23.000	14.200	0	178.00	60.400	121.68	67.600
14	162.00	10.000	23.000	10.600	0	178.00	60.400	35.30	67.600
15	162.00	10.000	23.000	13.600	0	178.00	60.400	120.79	67.600
16	162.00	10.000	23.000	15.000	0	178.00	60.400	114.50	67.600
17	162.00	10.000	23.000	16.600	0	178.00	60.400	71.30	67.600
18	162.00	10.000	23.000	14.000	0	178.00	60.400	121.98	67.600
19	163.40	8.600	21.600	12.600	0	178.00	60.400	97.60	69.799
20	161.20	10.800	23.800	14.800	0	178.00	60.400	97.60	79.698
21	164.60	7.400	20.400	11.400	0	178.00	60.400	97.60	33.800
22	161.60	10.400	23.400	14.400	0	178.00	60.400	97.60	83.292
23	160.40	11.600	24.600	15.600	0	178.00	60.400	97.60	65.299
24	160.80	11.200	24.200	15.200	0	178.00	60.400	97.60	73.677

The graphical displays in Output 84.7.4 and Output 84.7.5 are created when ODS Graphics is enabled. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the MIXED procedure, see the section "ODS Graphics" on page 6877.

Output 84.7.4 Fixed-Effects Deletion Estimates

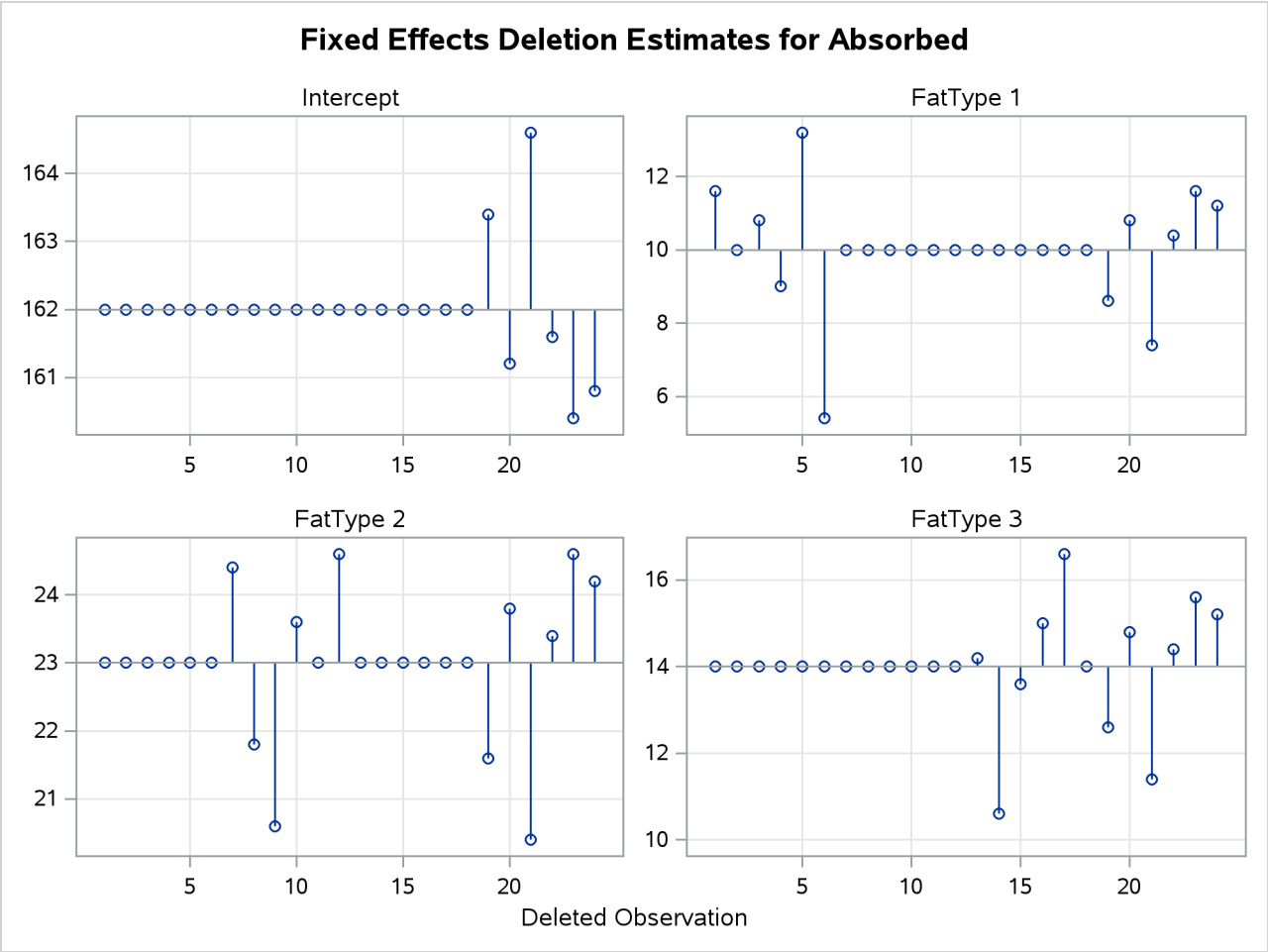

Output 84.7.5 Covariance Parameter Deletion Estimates

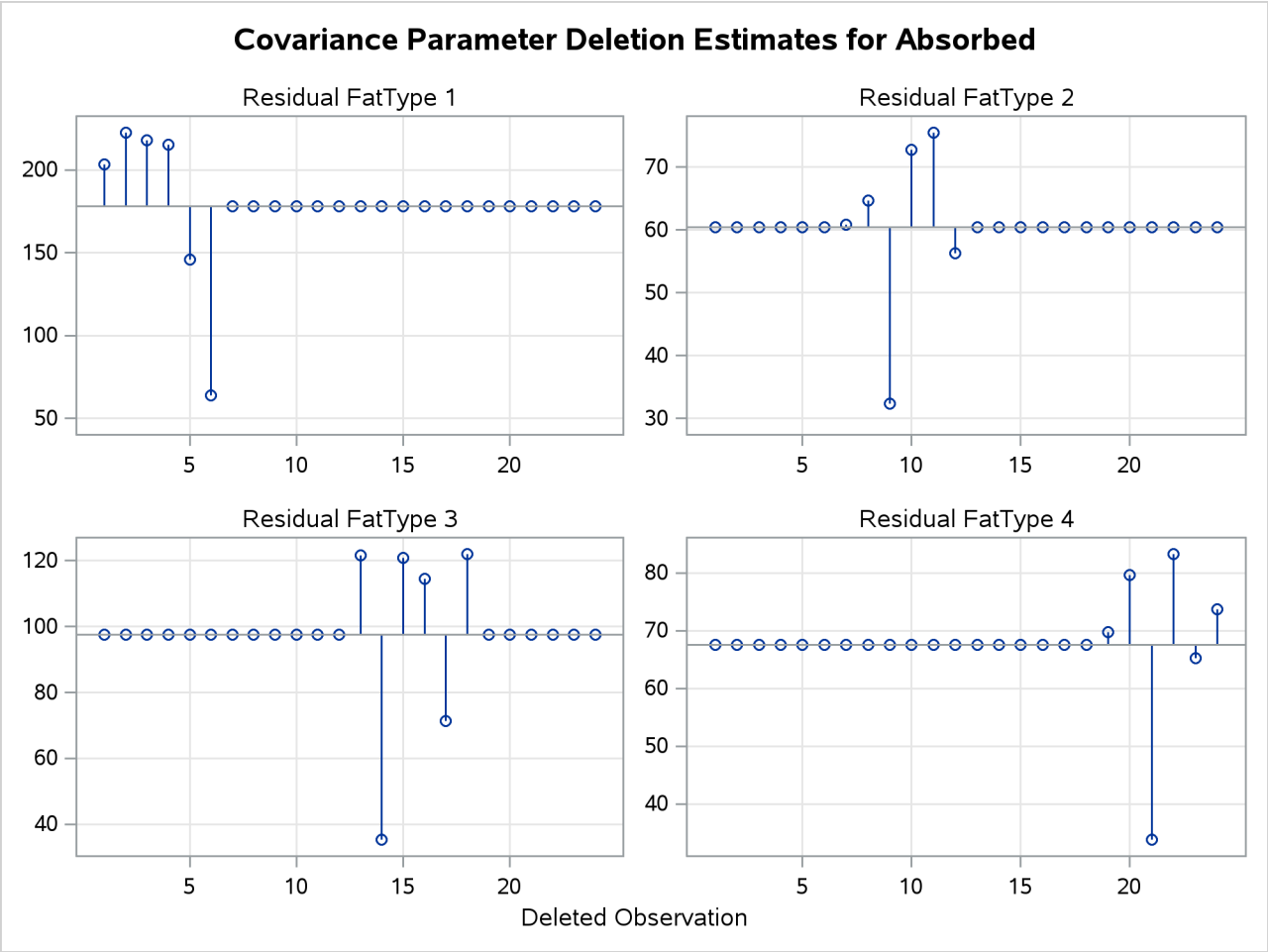

The estimate of the intercept is affected only when observations from the last group are removed. The estimate of the "FatType 1" effect reacts to removal of observations in the first and last group (Output 84.7.4).

While observations can affect one or more fixed-effects solutions in this model, they can affect only one covariance parameter, the variance in their group (Output 84.7.5). Observations 6, 9, 14, and 21, which are extreme in their group, reduce the group variance considerably.

Diagnostics related to residuals and predicted values are printed with the following statements:

```
proc print data=inf label;
   var observed predicted residual pressres
       student Rstudent;
run;
```

Output 84.7.6 Residual Diagnostics

Obs	Observed Value	Predicted Mean	Residual	PRESS Residual	Internally Studentized Residual	Externally Studentized Residual
1	164	172.0	-8.000	-9.600	-0.6569	-0.6146
2	172	172.0	0.000	0.000	0.0000	0.0000
3	168	172.0	-4.000	-4.800	-0.3284	-0.2970
4	177	172.0	5.000	6.000	0.4105	0.3736
5	156	172.0	-16.000	-19.200	-1.3137	-1.4521
6	195	172.0	23.000	27.600	1.8885	3.1544
7	178	185.0	-7.000	-8.400	-0.9867	-0.9835
8	191	185.0	6.000	7.200	0.8457	0.8172
9	197	185.0	12.000	14.400	1.6914	2.3131
10	182	185.0	-3.000	-3.600	-0.4229	-0.3852
11	185	185.0	0.000	-0.000	0.0000	0.0000
12	177	185.0	-8.000	-9.600	-1.1276	-1.1681
13	175	176.0	-1.000	-1.200	-0.1109	-0.0993
14	193	176.0	17.000	20.400	1.8850	3.1344
15	178	176.0	2.000	2.400	0.2218	0.1993
16	171	176.0	-5.000	-6.000	-0.5544	-0.5119
17	163	176.0	-13.000	-15.600	-1.4415	-1.6865
18	176	176.0	0.000	0.000	0.0000	0.0000
19	155	162.0	-7.000	-8.400	-0.9326	-0.9178
20	166	162.0	4.000	4.800	0.5329	0.4908
21	149	162.0	-13.000	-15.600	-1.7321	-2.4495
22	164	162.0	2.000	2.400	0.2665	0.2401
23	170	162.0	8.000	9.600	1.0659	1.0845
24	168	162.0	6.000	7.200	0.7994	0.7657

Observations 6, 9, 14, and 21 have large studentized residuals (Output 84.7.6). That the externally studentized residuals are much larger than the internally studentized residuals for these observations indicates that the variance estimate in the group shrinks when the observation is removed. Also important to note is that comparisons based on raw residuals in models with heterogeneous variance can be misleading. Observation 5, for example, has a larger residual but a smaller studentized residual than observation 21. The variance for the first fat type is much larger than the variance in the fourth group. A “large” residual is more “surprising” in the groups with small variance.

A measure of the overall influence on the analysis is the (restricted) likelihood distance, shown in Output 84.7.7. Observations 6, 9, 14, and 21 clearly displace the REML solution more than any other observations.

Output 84.7.7 Restricted Likelihood Distance

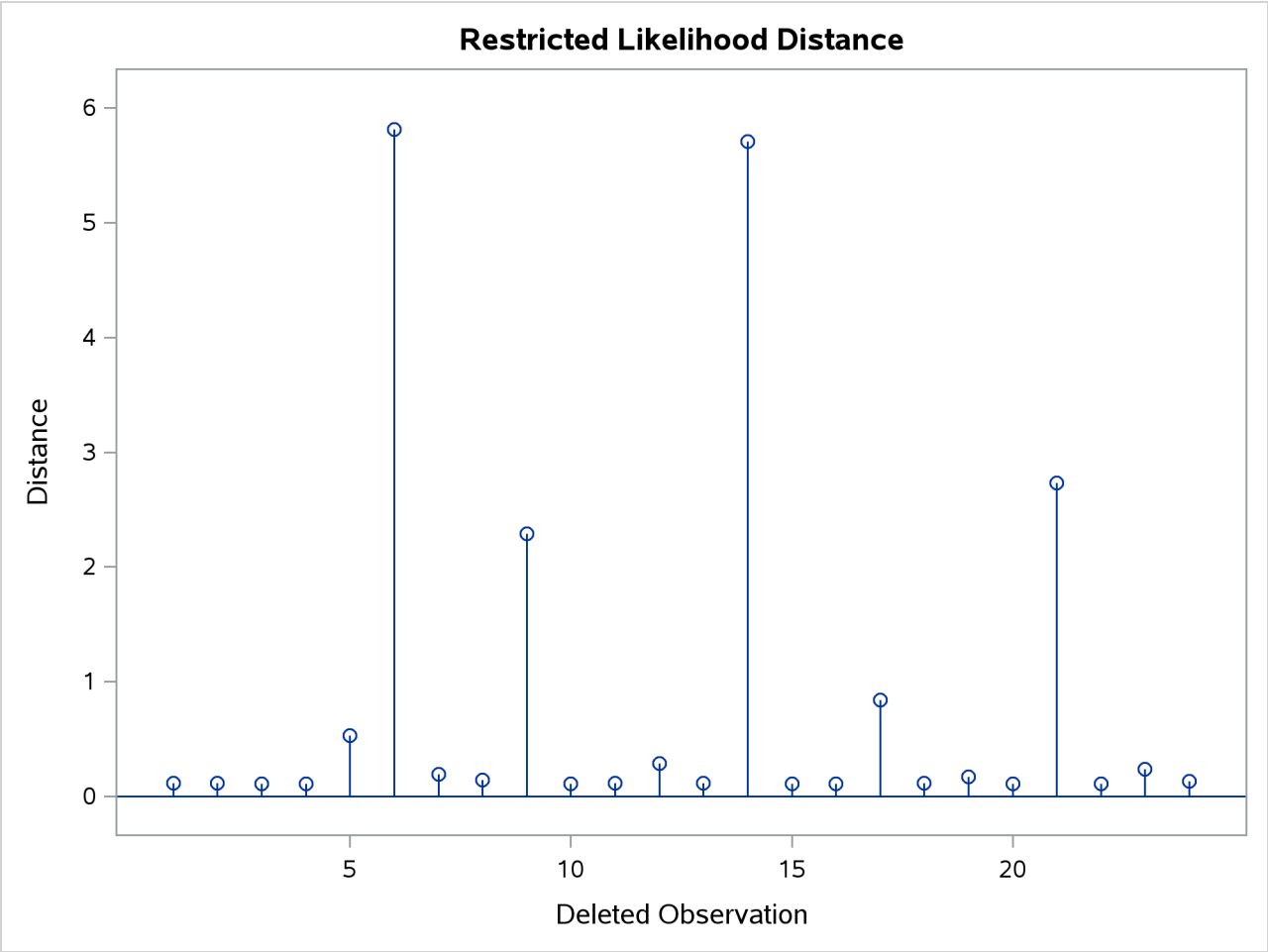

The following statements list the restricted likelihood distance and various diagnostics related to the fixed-effects estimates (Output 84.7.8):

```
proc print data=inf label;
   var leverage observed CookD DFFITS CovRatio RLD;
run;
```

Output 84.7.8 Restricted Likelihood Distance and Fixed-Effects Diagnostics

Obs	Leverage	Observed Value	Cook's D	DFFITS	COVRATIO	Restr. Likelihood Distance
1	0.167	164	0.02157	-0.27487	1.3706	0.1178
2	0.167	172	0.00000	-0.00000	1.4998	0.1156
3	0.167	168	0.00539	-0.13282	1.4675	0.1124
4	0.167	177	0.00843	0.16706	1.4494	0.1117
5	0.167	156	0.08629	-0.64938	0.9822	0.5290
6	0.167	195	0.17831	1.41069	0.4301	5.8101
7	0.167	178	0.04868	-0.43982	1.2078	0.1935
8	0.167	191	0.03576	0.36546	1.2853	0.1451
9	0.167	197	0.14305	1.03446	0.6416	2.2909
10	0.167	182	0.00894	-0.17225	1.4463	0.1116
11	0.167	185	0.00000	-0.00000	1.4998	0.1156
12	0.167	177	0.06358	-0.52239	1.1183	0.2856
13	0.167	175	0.00061	-0.04441	1.4961	0.1151
14	0.167	193	0.17766	1.40175	0.4340	5.7044
15	0.167	178	0.00246	0.08915	1.4851	0.1139
16	0.167	171	0.01537	-0.22892	1.4078	0.1129
17	0.167	163	0.10389	-0.75423	0.8766	0.8433
18	0.167	176	0.00000	0.00000	1.4998	0.1156
19	0.167	155	0.04349	-0.41047	1.2390	0.1710
20	0.167	166	0.01420	0.21950	1.4148	0.1124
21	0.167	149	0.15000	-1.09545	0.6000	2.7343
22	0.167	164	0.00355	0.10736	1.4786	0.1133
23	0.167	170	0.05680	0.48500	1.1592	0.2383
24	0.167	168	0.03195	0.34245	1.3079	0.1353

In this example, observations with large likelihood distances also have large values for Cook's *D* and values of CovRatio far less than one (Output 84.7.8). The latter indicates that the fixed effects are estimated more precisely when these observations are removed from the analysis.

The following statements print the values of the *D* statistic and the CovRatio for the covariance parameters:

```
proc print data=inf label;
   var iter CookDCP CovRatioCP;
run;
```

The same conclusions as for the fixed-effects estimates hold for the covariance parameter estimates. Observations 6, 9, 14, and 21 change the estimates and their precision considerably (Output 84.7.9, Output 84.7.10). All iterative updates converged within at most four iterations.

Output 84.7.9 Covariance Parameter Diagnostics

Obs	Iterations	Cook's D CovParms	COVRATIO CovParms
1	3	0.05050	1.6306
2	3	0.15603	1.9520
3	3	0.12426	1.8692
4	3	0.10796	1.8233
5	4	0.08232	0.8375
6	4	1.02909	0.1606
7	1	0.00011	1.2662
8	2	0.01262	1.4335
9	3	0.54126	0.3573
10	3	0.10531	1.8156
11	3	0.15603	1.9520
12	2	0.01160	1.0849
13	3	0.15223	1.9425
14	4	1.01865	0.1635
15	3	0.14111	1.9141
16	3	0.07494	1.7203
17	3	0.18154	0.6671
18	3	0.15603	1.9520
19	2	0.00265	1.3326
20	3	0.08008	1.7374
21	1	0.62500	0.3125
22	3	0.13472	1.8974
23	2	0.00290	1.1663
24	2	0.02020	1.4839

Output 84.7.10 displays the standard panel of influence diagnostics that is obtained when influence analysis is iterative. The Cook's *D* and CovRatio statistics are displayed for each deletion set for both fixed-effects and covariance parameter estimates. This provides a convenient summary of the impact on the analysis for each deletion set, since Cook's *D* statistic measures impact on the estimates and the CovRatio statistic measures impact on the precision of the estimates.

Output 84.7.10 Influence Diagnostics

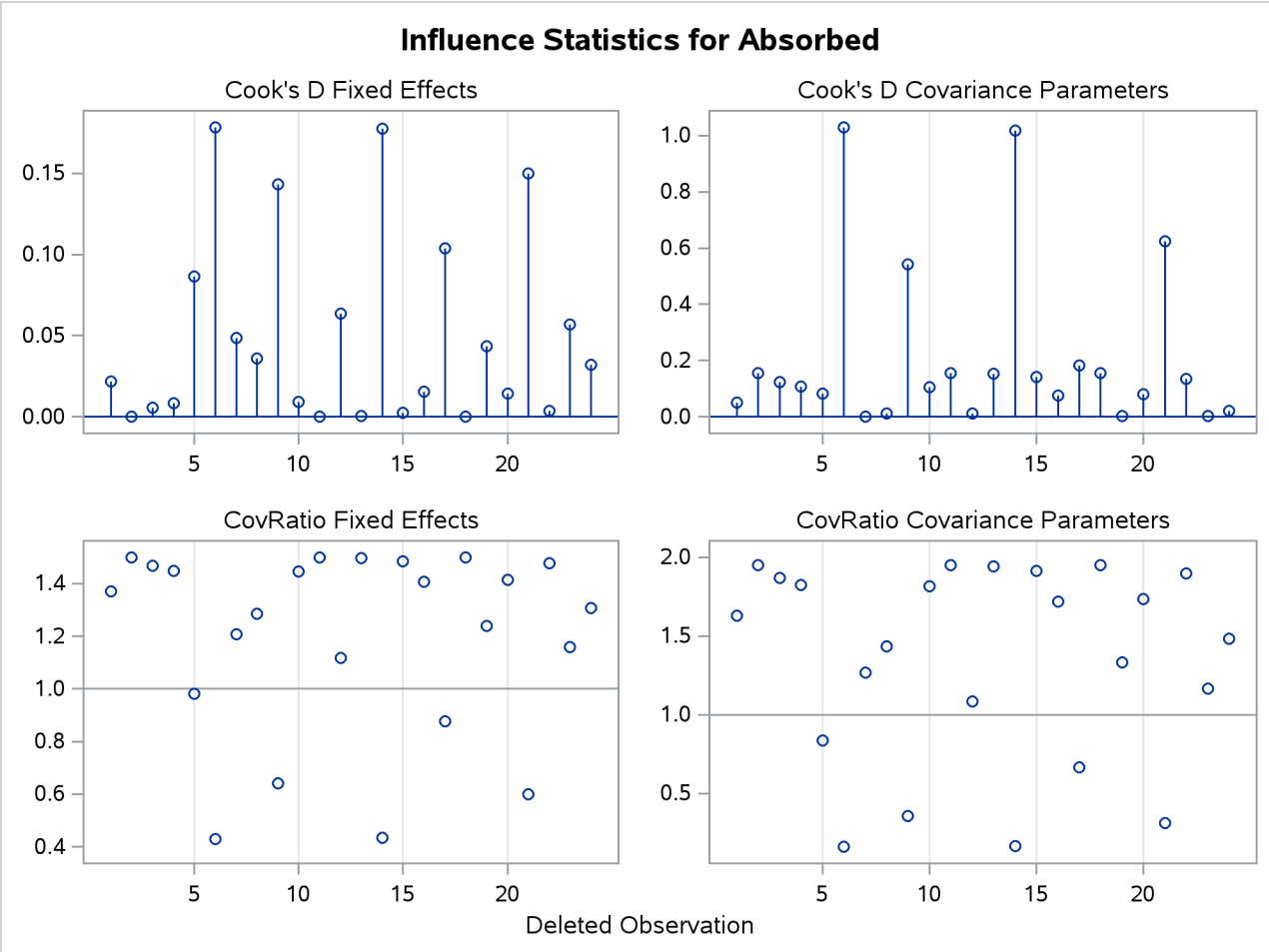

Observations 6, 9, 14, and 21 have considerable impact on estimates and precision of fixed effects and covariance parameters. This is not necessarily the case. Observations can be influential on only some aspects of the analysis, as shown in the next example.

Example 84.8: Influence Analysis for Repeated Measures Data

(View the complete code for this example (mixex8.sas) in the example repository.)

This example revisits the repeated measures data of Pothoff and Roy (1964) that were analyzed in Example 84.2. Recall that the data consist of growth measurements at ages 8, 10, 12, and 14 for 11 girls and 16 boys. The model being fit contains fixed effects for Gender and Age and their interaction.

The earlier analysis of these data indicated some unusual observations in this data set. Because of the clustered data structure, it is of interest to study the influence of clusters (children) on the analysis rather than the influence of individual observations. A cluster comprises the repeated measurements for each child.

The repeated measures are first modeled with an unstructured within-child variance-covariance matrix. A residual variance is not profiled in this model. A noniterative influence analysis will update the fixed

effects only. The following statements request this noniterative maximum likelihood analysis and produce Output 84.8.1:

```
proc mixed data=pr method=ml;
   class person gender;
   model y = gender age gender*age /
                   influence(effect=person);
   repeated / type=un subject=person;
   ods select influence;
run;
```

Output 84.8.1 Default Influence Statistics in Noniterative Analysis

The Mixed Procedure

Influence Diagnostics for Levels of Person

Person	Number of Observations in Level	PRESS Statistic	Cook's D
1	4	10.1716	0.01539
2	4	3.8187	0.03988
3	4	10.8448	0.02891
4	4	24.0339	0.04515
5	4	1.6900	0.01613
6	4	11.8592	0.01634
7	4	1.1887	0.00521
8	4	4.6717	0.02742
9	4	13.4244	0.03949
10	4	85.1195	0.13848
11	4	67.9397	0.09728
12	4	40.6467	0.04438
13	4	13.0304	0.00924
14	4	6.1712	0.00411
15	4	24.5702	0.12727
16	4	20.5266	0.01026
17	4	9.9917	0.01526
18	4	7.9355	0.01070
19	4	15.5955	0.01982
20	4	42.6845	0.01973
21	4	95.3282	0.10075
22	4	13.9649	0.03778
23	4	4.9656	0.01245
24	4	37.2494	0.15094
25	4	4.3756	0.03375
26	4	8.1448	0.03470
27	4	20.2913	0.02523

Each observation in the "Influence Diagnostics for Levels of Person" table in Output 84.8.1 represents the removal of four observations. The subjects 10, 15, and 24 have the greatest impact on the fixed effects (Cook's D), and subject 10 and 21 have large PRESS statistics. The 21st child has a large PRESS statistic, and its D statistic is not that extreme. This is an indication that the model fits rather poorly for this child,

whether it is part of the data or not.

The previous analysis does not take into account the effect on the covariance parameters when a subject is removed from the analysis. If you also update the covariance parameters, the impact of observations on these can amplify or allay their effect on the fixed effects. To assess the overall influence of subjects on the analysis and to compute separate statistics for the fixed effects and covariance parameters, an iterative analysis is obtained by adding the INFLUENCE suboption ITER=, as follows:

```
ods graphics on;

proc mixed data=pr method=ml;
   class person gender;
   model y = gender age gender*age /
                   influence(effect=person iter=5);
   repeated / type=un subject=person;
run;
```

The number of additional iterations following removal of the observations for a particular subject is limited to five. Graphical displays of influence diagnostics are created when ODS Graphics is enabled. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the MIXED procedure, see the section "ODS Graphics" on page 6877.

The MIXED procedure produces a plot of the restricted likelihood distance (Output 84.8.2) and a panel of diagnostics for fixed effects and covariance parameters (Output 84.8.3).

Output 84.8.2 Restricted Likelihood Distance

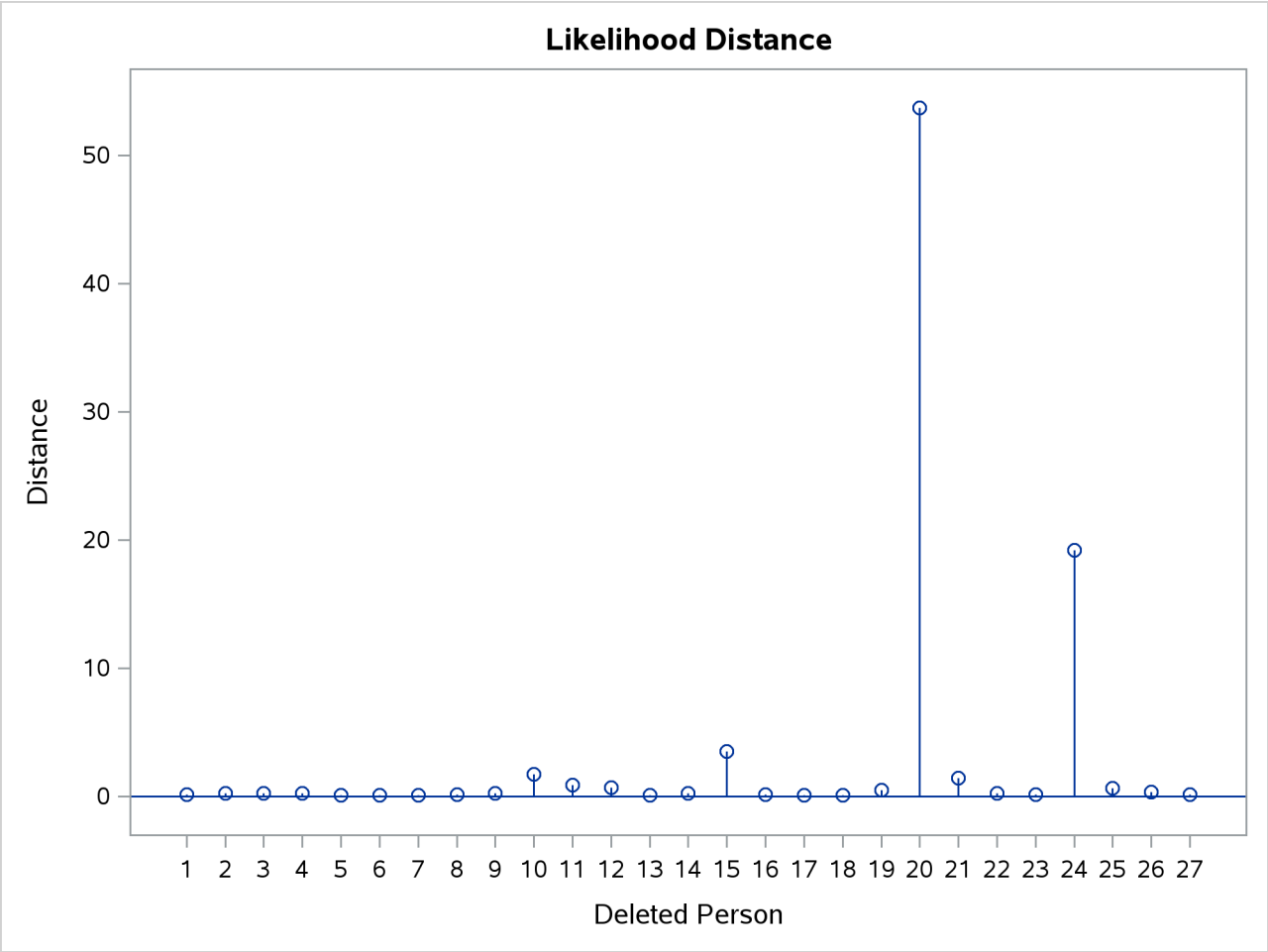

Output 84.8.3 Influence Diagnostics Panel

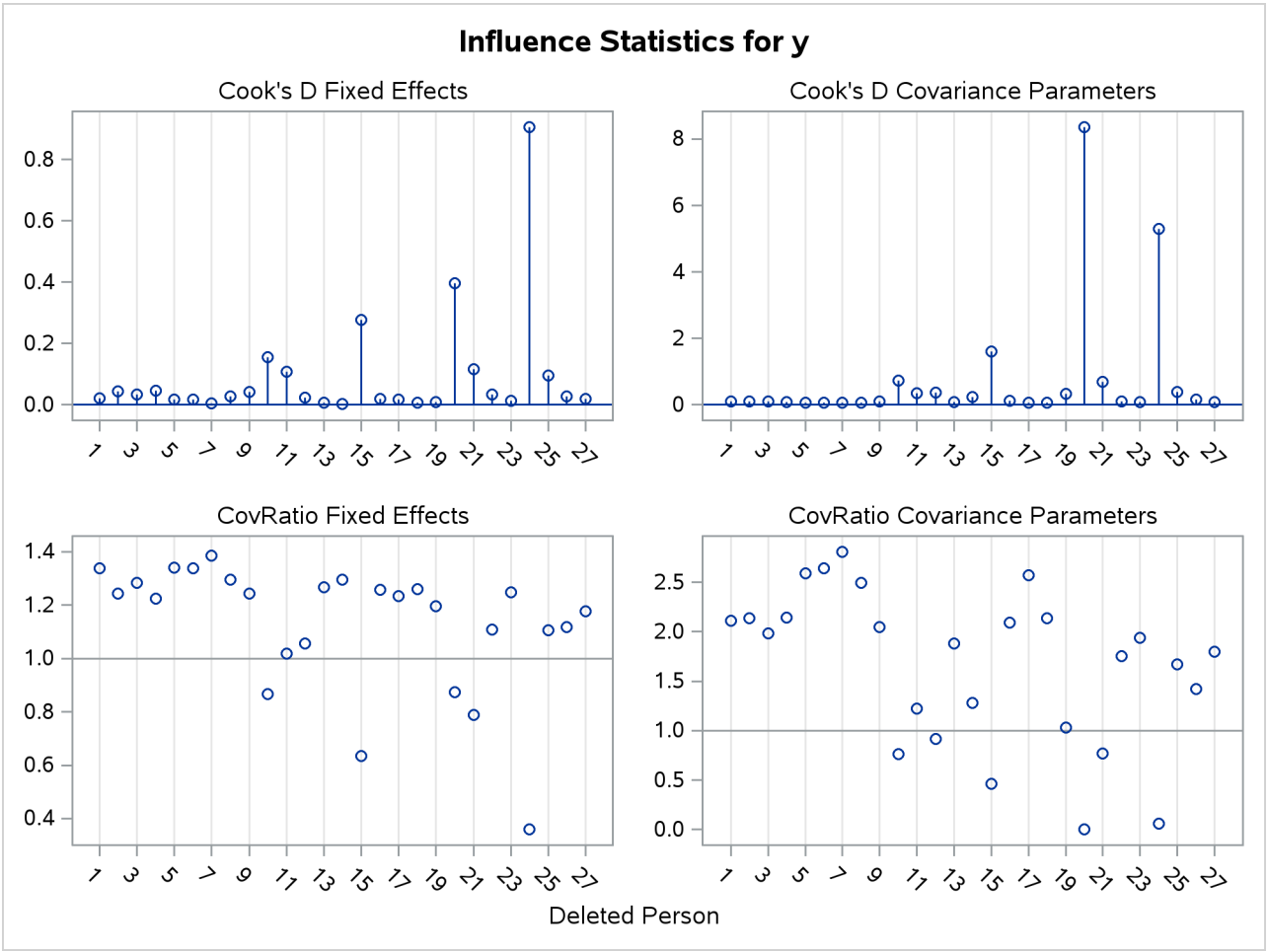

As judged by the restricted likelihood distance, subjects 20 and 24 clearly have the most influence on the overall analysis (Output 84.8.2).

Output 84.8.3 displays Cook's *D* and CovRatio statistics for the fixed effects and covariance parameters. Clearly, subject 20 has a dramatic effect on the estimates of variances and covariances. This subject also affects the precision of the covariance parameter estimates more than any other subject in Output 84.8.3 (CovRatio near 0).

The child who exerts the greatest influence on the fixed effects is subject 24. Maybe surprisingly, this subject affects the variance-covariance matrix of the fixed effects more than subject 20 (small CovRatio in Output 84.8.3).

The final model investigated for these data is a random coefficient model as in Stram and Lee (1994) with random effects for the intercept and age effect. The following statements examine the estimates for fixed effects and the entries of the unstructured 2×2 variance matrix of the random coefficients graphically:

```
proc mixed data=pr method=ml
           plots(only)=InfluenceEstPlot;
   class person gender;
   model y = gender age gender*age /
```

```
        influence(iter=5 effect=person est);
    random intercept age / type=un subject=person;
run;
```

The PLOTS(ONLY)=INFLUENCEESTPLOT option restricts the graphical output from this PROC MIXED run to only the panels of deletion estimates (Output 84.8.4 and Output 84.8.5).

Output 84.8.4 Fixed-Effects Deletion Estimates

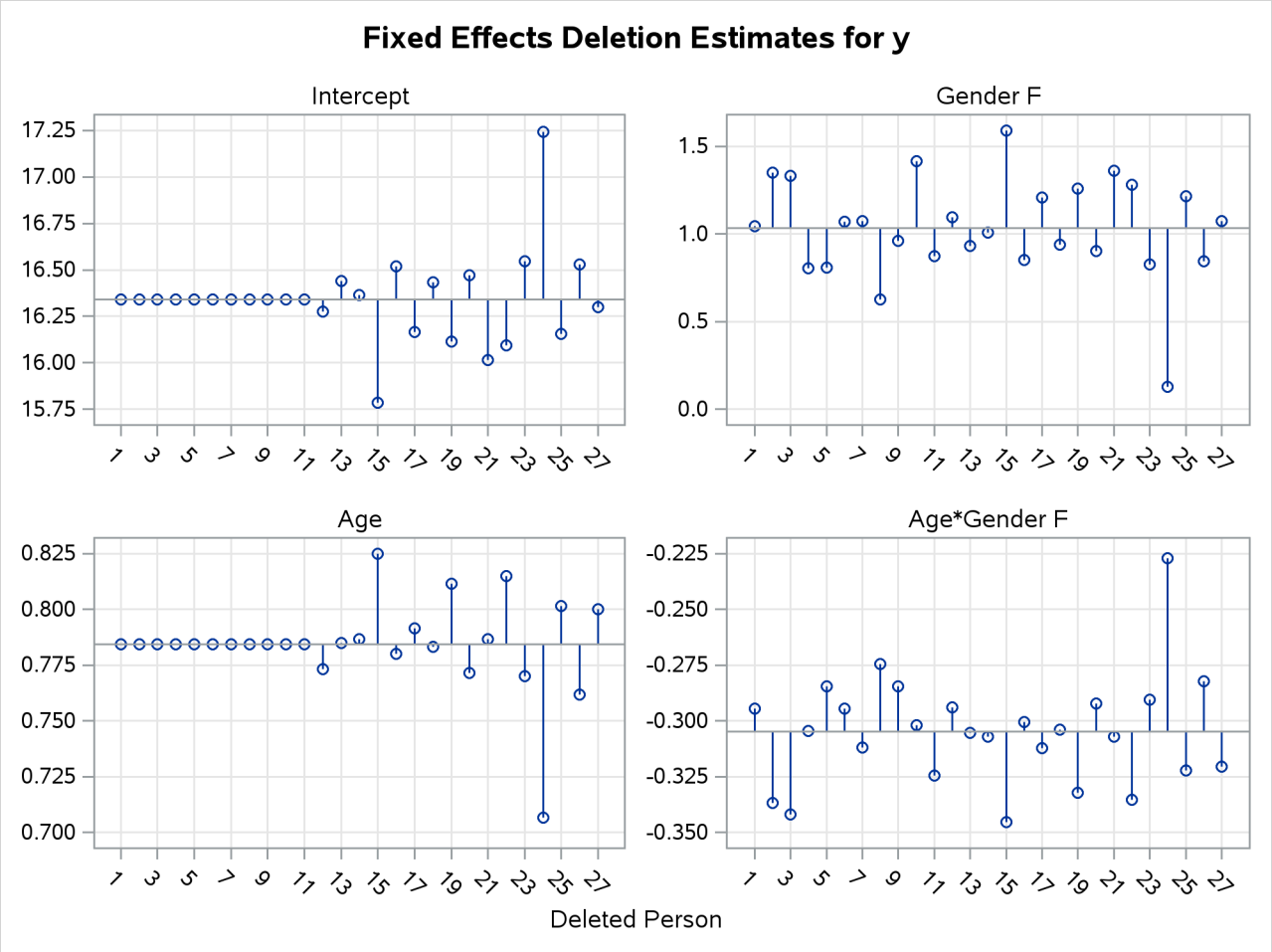

In Output 84.8.4 the graphs on the left side of the panel represent the intercept and slope estimate for boys; the graphs on the right side represent the difference in intercept and slope between boys and girls. Removing any one of the first eleven children, who are girls, does not alter the intercept or slope in the group of boys. The difference in these parameters between boys and girls is altered by the removal of any child. Subject 24 changes the fixed effects considerably, subject 20 much less so.

Output 84.8.5 Covariance Parameter Deletion Estimates

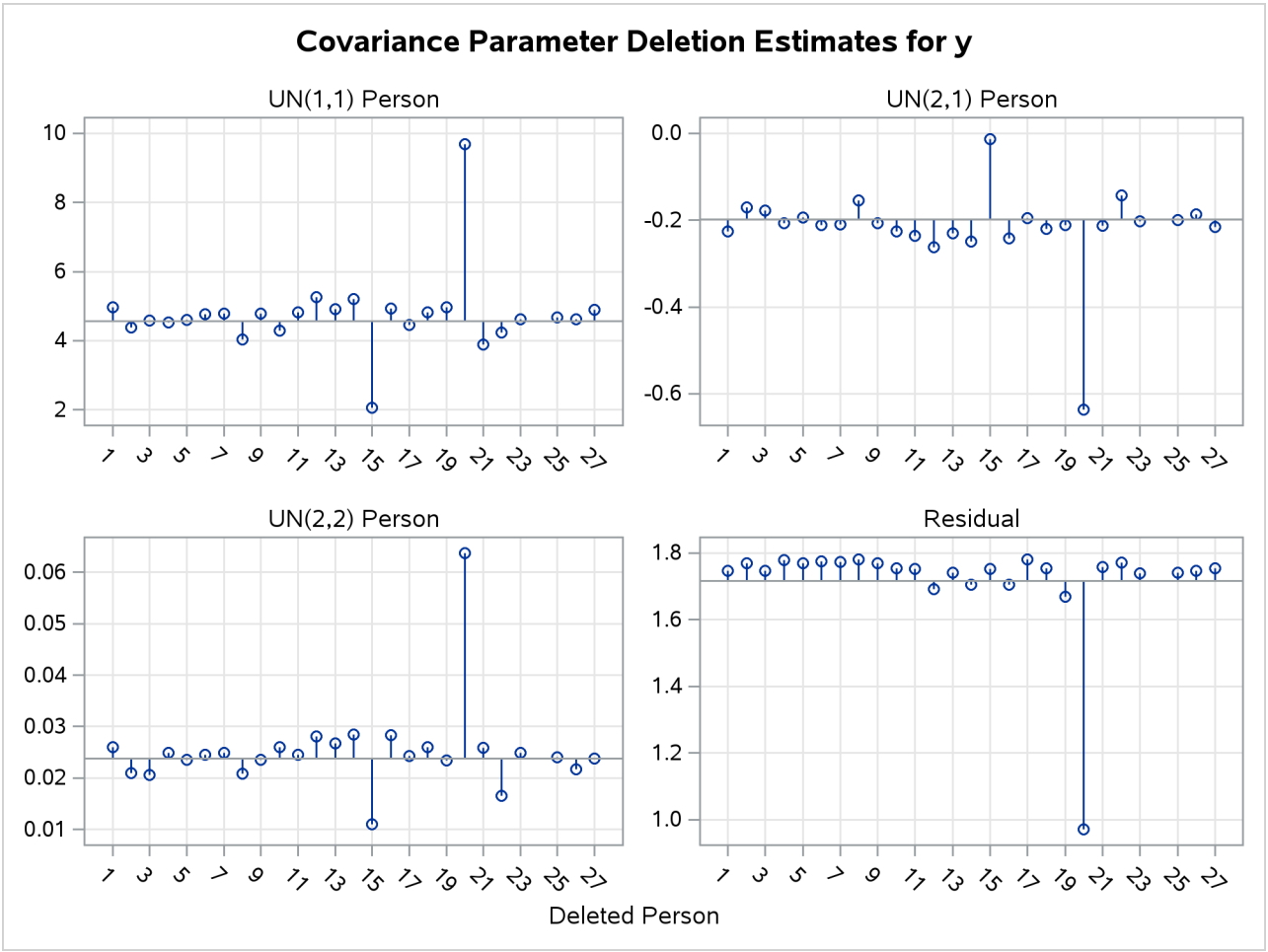

The covariance parameter deletion estimates in Output 84.8.5 show several important features.

- The panels do not contain information about subject 24. Estimation of the **G** matrix following removal of that child did not yield a positive definite matrix. As a consequence, covariance parameter diagnostics are not produced for this subject.
- Subject 20 has great impact on the four covariance parameters. Removing this child from the analysis increases the variance of the random intercept and random slope and reduces the residual variance by almost 80%. The repeated measurements of this child exhibit an up-and-down behavior.
- The variance of the random intercept and slope are reduced when child 15 is removed from the analysis. This child's growth measurements oscillate about 27.0 from age 10 on.

Examining observed and residual values by levels of classification variables is also a useful tool to diagnose the adequacy of the model and unusual observations. Box plots for effects in the model that consist of only classification variables can be requested with the BOXPLOT option of the PLOTS= option in the PROC MIXED statement. For example, the following statements produce box plots for the SUBJECT= effects in the model:

```
ods graphics on;
proc mixed data=pr method=ml
       plot=boxplot(observed marginal conditional subject);
   class person gender;
   model y = gender age gender*age;
   random intercept age / type=un subject=person;
run;
```

The specific boxplot options request a plot of the observed data (Output 84.8.6), the marginal residuals (Output 84.8.7), and the conditional residuals (Output 84.8.8). Box plots of the observed values show the variation within and between children clearly. The group of girls (subjects 1–11) is distinguishable from the group of boys by somewhat lesser average growth and lesser within-child variation (Output 84.8.6). After adjusting for overall (population-averaged) gender and age effects, the residual within-child variation is reduced but substantial differences in the means remain (Output 84.8.7). If child-specific inferences are desired, a model accounting for only Gender, Age, and Gender*Age effects is not adequate for these data.

Output 84.8.6 Distribution of Observed Values

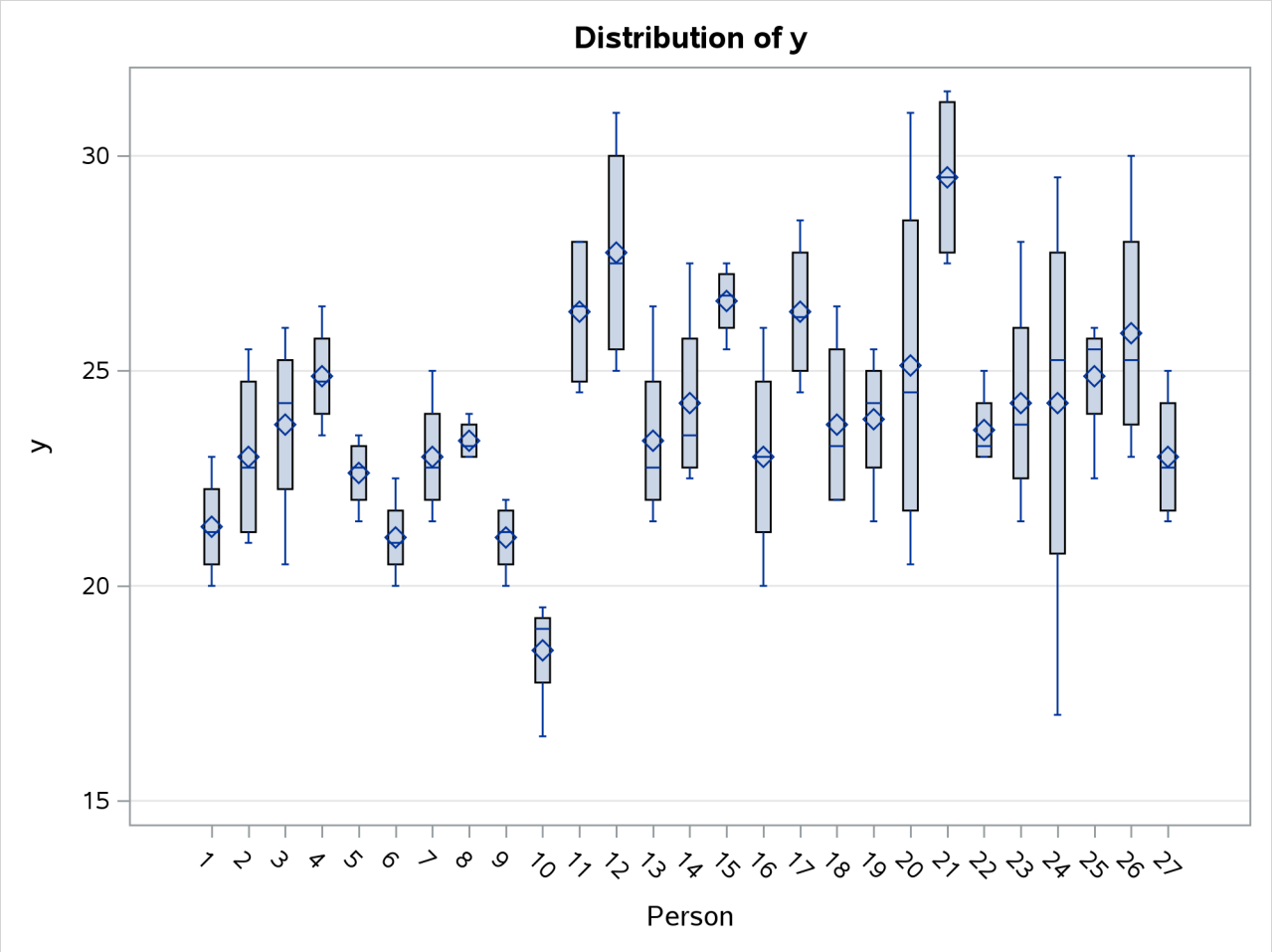

Output 84.8.7 Distribution of Marginal Residuals

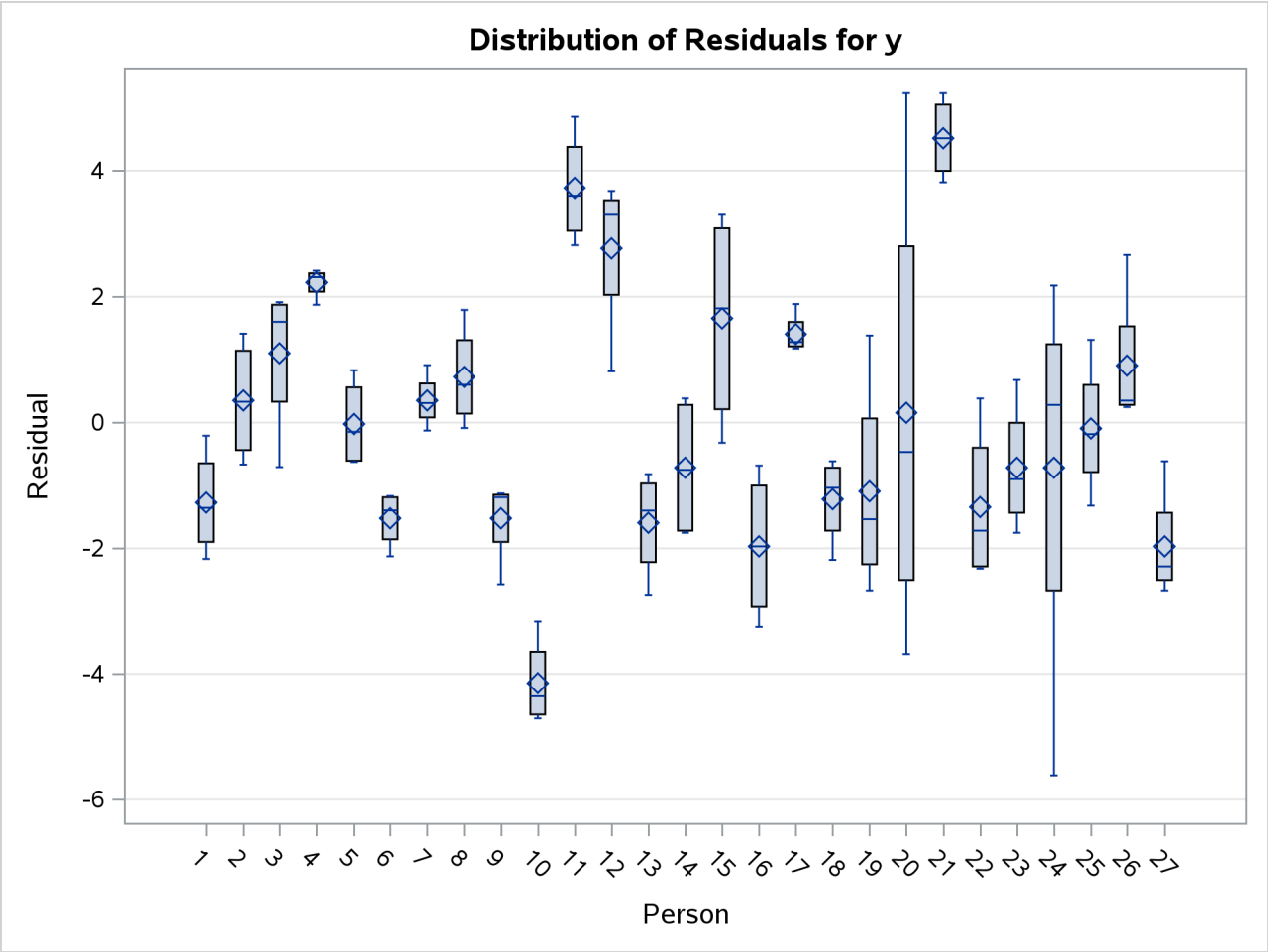

The conditional residuals incorporate the EBLUPs for each child and enable you to examine whether the subject-specific model is adequate (Output 84.8.8). By using each child "as its own control," the residuals are now centered near zero. Subjects 20 and 24 stand out as unusual in all three sets of box plots.

Output 84.8.8 Distribution of Conditional Residuals

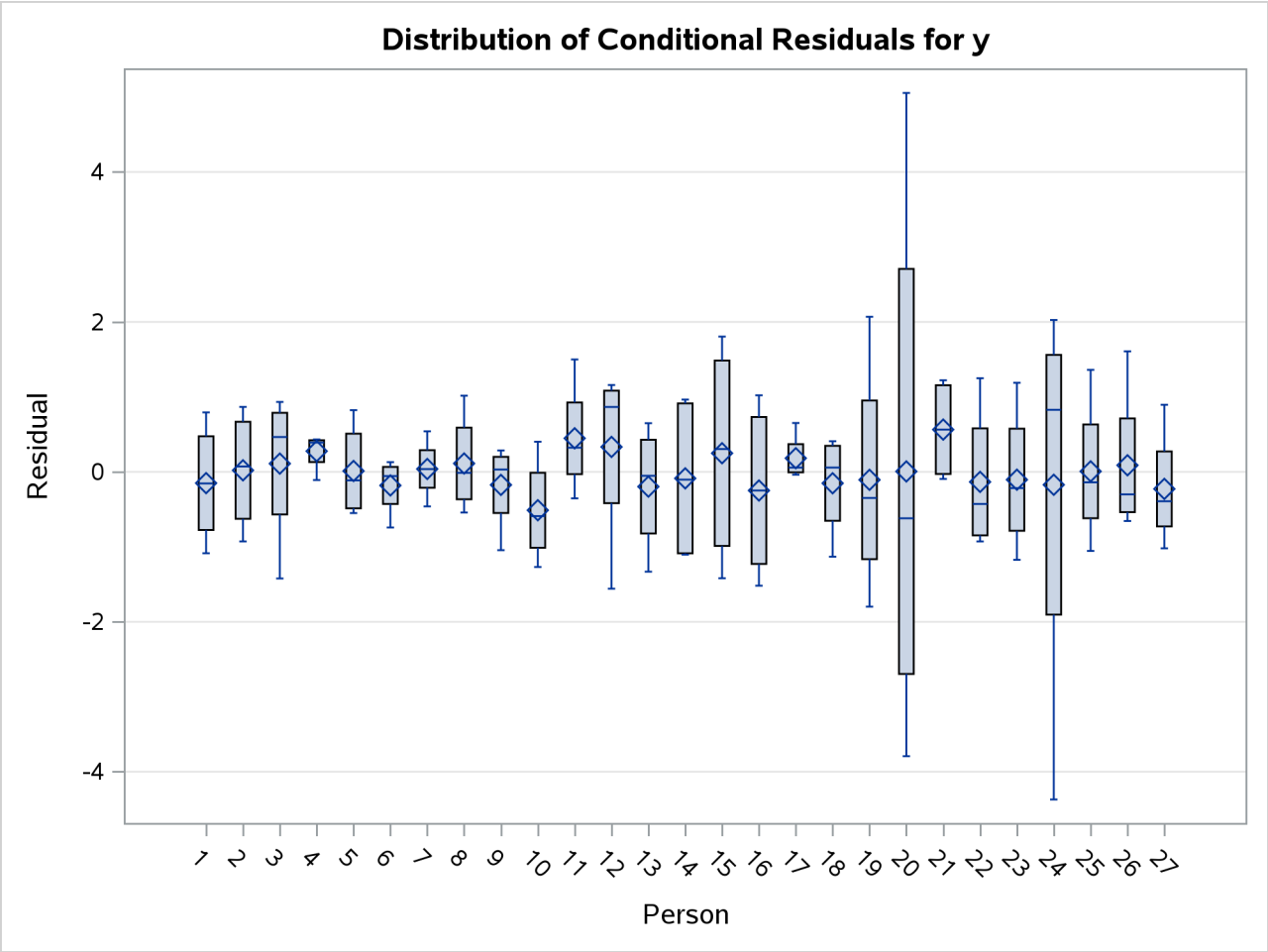

Example 84.9: Examining Individual Test Components

(View the complete code for this example (mixex9.sas) in the example repository.)

The LCOMPONENTS option in the MODEL statement enables you to perform single-degree-of-freedom tests for individual rows of the **L** matrix. Such tests are useful to identify interaction patterns. In a balanced layout, Type 3 components of **L** associated with A*B interactions correspond to simple contrasts of cell mean differences.

The first example revisits the data from the split-plot design by Stroup (1989a) that was analyzed in Example 84.1. Recall that variables A and B in the following statements represent the whole-plot and subplot factors, respectively:

```
proc mixed data=sp;
   class a b block;
   model y = a b a*b / LComponents e3;
   random block a*block;
run;
```

The MIXED procedure constructs a separate **L** matrix for each of the three fixed-effects components. The matrices are displayed in Output 84.9.1. The tests for fixed effects are shown in Output 84.9.2.

Output 84.9.1 Coefficients of Type 3 Estimable Functions

The Mixed Procedure

Type 3 Coefficients for A				
Effect	**A**	**B**	**Row1**	**Row2**
Intercept				
A	1		1	
A	2			1
A	3		-1	-1
B		1		
B		2		
A*B	1	1	0.5	
A*B	1	2	0.5	
A*B	2	1		0.5
A*B	2	2		0.5
A*B	3	1	-0.5	-0.5
A*B	3	2	-0.5	-0.5

Type 3 Coefficients for B			
Effect	**A**	**B**	**Row1**
Intercept			
A	1		
A	2		
A	3		
B		1	1
B		2	-1
A*B	1	1	0.3333
A*B	1	2	-0.333
A*B	2	1	0.3333
A*B	2	2	-0.333
A*B	3	1	0.3333
A*B	3	2	-0.333

Output 84.9.1 *continued*

Type 3 Coefficients for A*B				
Effect	**A**	**B**	**Row1**	**Row2**
Intercept				
A	1			
A	2			
A	3			
B		1		
B		2		
A*B	1	1	1	
A*B	1	2	-1	
A*B	2	1		1
A*B	2	2		-1
A*B	3	1	-1	-1
A*B	3	2	1	1

Output 84.9.2 Type 3 Tests in Split-Plot Example

Type 3 Tests of Fixed Effects				
Effect	**Num DF**	**Den DF**	**F Value**	**Pr > F**
A	2	6	4.07	0.0764
B	1	9	19.39	0.0017
A*B	2	9	4.02	0.0566

If $\mu_{i.}$ denotes a whole-plot main effect mean, $\mu_{.j}$ denotes a subplot main effect mean, and μ_{ij} denotes a cell mean, the five components shown in Output 84.9.3 correspond to tests of the following:

- $H_0 : \mu_{1.} = \mu_{3.}$
- $H_0 : \mu_{2.} = \mu_{3.}$
- $H_0 : \mu_{.1} = \mu_{.2}$
- $H_0 : \mu_{11} - \mu_{12} = \mu_{31} - \mu_{32}$
- $H_0 : \mu_{21} - \mu_{22} = \mu_{31} - \mu_{32}$

Output 84.9.3 Type 3 L Components Table

L Components of Type 3 Tests of Fixed Effects						
Effect	**L Index**	**Estimate**	**Standard Error**	**DF**	**t Value**	**Pr > |t|**
A	**1**	7.1250	3.1672	6	2.25	0.0655
A	**2**	8.3750	3.1672	6	2.64	0.0383
B	**1**	5.5000	1.2491	9	4.40	0.0017
A*B	**1**	7.7500	3.0596	9	2.53	0.0321
A*B	**2**	7.2500	3.0596	9	2.37	0.0419

The first three components are comparisons of marginal means. The fourth component compares the effect of factor B at the first whole-plot level against the effect of B at the third whole-plot level. Finally, the last component tests whether the factor B effect changes between the second and third whole-plot level.

The Type 3 component tests can also be produced with these corresponding ESTIMATE statements:

```
proc mixed data=sp;
   class a b block ;
   model y = a b a*b;
   random block a*block;
   estimate 'a    1' a 1 0 -1;
   estimate 'a    2' a 0 1 -1;
   estimate 'b    1' b   1 -1;
   estimate 'a*b  1' a*b 1 -1 0  0 -1 1;
   estimate 'a*b  2' a*b 0  0 1 -1 -1 1;
   ods select Estimates;
run;
```

The results are shown in Output 84.9.4.

Output 84.9.4 Results from ESTIMATE Statements

The Mixed Procedure

Estimates

Label	Estimate	Standard Error	DF	t Value	Pr > |t|
a 1	7.1250	3.1672	6	2.25	0.0655
a 2	8.3750	3.1672	6	2.64	0.0383
b 1	5.5000	1.2491	9	4.40	0.0017
a*b 1	7.7500	3.0596	9	2.53	0.0321
a*b 2	7.2500	3.0596	9	2.37	0.0419

A second useful application of the LCOMPONENTS option is in polynomial models, where Type 1 tests are often used to test the entry of model terms sequentially. The SOLUTION option in the MODEL statement displays the regression coefficients that correspond to a Type 3 analysis. That is, the coefficients represent the partial coefficients you would get by adding the regressor variable last in a model containing all other effects, and the tests are identical to those in the "Type 3 Tests of Fixed Effects" table.

Consider the following DATA step and the fit of a third-order polynomial regression model.

```
data polynomial;
   do x=1 to 20; input y@@; output; end;
   datalines;
1.092   1.758   1.997   3.154   3.880
3.810   4.921   4.573   6.029   6.032
6.291   7.151   7.154   6.469   7.137
6.374   5.860   4.866   4.155   2.711
;

proc mixed data=polynomial;
   model y = x x*x x*x*x / s lcomponents htype=1,3;
run;
```

The *t* tests displayed in the "Solution for Fixed Effects" table are Type 3 tests, sometimes referred to as partial tests. They measure the contribution of a regressor in the presence of all other regressor variables in the model.

Output 84.9.5 Parameter Estimates in Polynomial Model

The Mixed Procedure

Solution for Fixed Effects

Effect	Estimate	Standard Error	DF	t Value	Pr > |t|
Intercept	0.7837	0.3545	16	2.21	0.0420
x	0.3726	0.1426	16	2.61	0.0189
x*x	0.04756	0.01558	16	3.05	0.0076
x*x*x	-0.00306	0.000489	16	-6.27	<.0001

The Type 3 L components are identical to the tests in the "Solutions for Fixed Effects" table shown in Output 84.9.5. The Type 1 table yields the following:

- sequential (Type 1) tests of regression variables that test the significance of a regressor given all other variables preceding it in the model list
- the regression coefficients for sequential submodels

Output 84.9.6 Type 1 and Type 3 L Components

L Components of Type 1 Tests of Fixed Effects

Effect	L Index	Estimate	Standard Error	DF	t Value	Pr > |t|
x	1	0.1763	0.01259	16	14.01	<.0001
x*x	1	-0.04886	0.002449	16	-19.95	<.0001
x*x*x	1	-0.00306	0.000489	16	-6.27	<.0001

L Components of Type 3 Tests of Fixed Effects

Effect	L Index	Estimate	Standard Error	DF	t Value	Pr > |t|
x	1	0.3726	0.1426	16	2.61	0.0189
x*x	1	0.04756	0.01558	16	3.05	0.0076
x*x*x	1	-0.00306	0.000489	16	-6.27	<.0001

The estimate of 0.1763 is the regression coefficient in a simple linear regression of Y on X. The estimate of –0.04886 is the partial coefficient for the quadratic term when it is added to a model containing only a linear component. Similarly, the value –0.00306 is the partial coefficient for the cubic term when it is added to a model containing a linear and quadratic component. The last Type 1 component is always identical to the corresponding Type 3 component.

Example 84.10: Isotonic Contrasts for Ordered Mean Values

(View the complete code for this example (mixex10.sas) in the example repository.)

It is often of interest to test whether the mean values of the dependent variable increases or decreases monotonically with certain factors. Hirotsu and Srivastava (2000) demonstrate one approach by using data (Moriguchi 1976). The data consist of ferrite cores subjected to four increasing temperatures. The response variable is the magnetic force of each core.

```
data FerriteCores;
   do Temp = 1 to 4;
      do rep = 1 to 5; drop rep;
         input MagneticForce @@;
         output;
      end;
   end;
   datalines;
10.8  9.9 10.7 10.4  9.7
10.7 10.6 11.0 10.8 10.9
11.9 11.2 11.0 11.1 11.3
11.4 10.7 10.9 11.3 11.7
;
```

The method presented by Hirotsu and Srivastava (2000) to test whether the magnetic force of the cores rises monotonically with temperature depends on the lower confidence limits of the *isotonic contrasts* of the force means at each temperature, adjusted for multiplicity. The corresponding isotonic contrast compares the average of a particular group and the preceding groups with the average of the succeeding groups. You can compute adjusted confidence intervals for isotonic contrasts by using the LSMESTIMATE statement.

The following statements analyze the FerriteCores data as a one-way design and multiplicity-adjusted lower confidence limits for the isotonic contrasts. For the multiplicity adjustment, the LSMESTIMATE statement employs simulation, which provides adjusted p-values and lower confidence limits that are exact up to Monte Carlo error.

```
proc mixed data=FerriteCores;
   class Temp;
   model MagneticForce = Temp;
   lsmestimate Temp
        'avg(1:1)<avg(2:4)' -3  1  1  1 divisor=3,
        'avg(1:2)<avg(3:4)' -1 -1  1  1 divisor=2,
        'avg(1:3)<avg(4:4)' -1 -1 -1  3 divisor=3
        / adjust=simulate(seed=1) cl upper;
   ods select LSMestimates;
run;
```

The results are shown in Output 84.10.1.

Output 84.10.1 Analysis of LS-Means with Isotonic Contrasts

The Mixed Procedure

Least Squares Means Estimates
Adjustment for Multiplicity: Simulated

Effect	Label	Estimate	Standard Error	DF	t Value	Tails	Pr > t	Adj P	Alpha	Lower	Upper	Adj Lower	Adj Upper
Temp	avg(1:1)<avg(2:4)	0.8000	0.1906	16	4.20	Upper	0.0003	0.0010	0.05	0.4672	Infty	0.3771	Infty
Temp	avg(1:2)<avg(3:4)	0.7000	0.1651	16	4.24	Upper	0.0003	0.0009	0.05	0.4118	Infty	0.3337	Infty
Temp	avg(1:3)<avg(4:4)	0.4000	0.1906	16	2.10	Upper	0.0260	0.0625	0.05	0.06721	Infty	-0.02291	Infty

With an adjusted p-value of 0.001, the magnetic force at the first temperature is significantly less than the average of the other temperatures. Likewise, the average of the first two temperatures is significantly less than the average of the last two ($p = 0.0009$). However, the magnetic force at the last temperature is not significantly greater than the average magnetic force of the others ($p = 0.0625$). These results indicate a significant monotone increase over the first three temperatures, but not across all four temperatures.

References

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Akritas, M. G., Arnold, S. F., and Brunner, E. (1997). "Nonparametric Hypotheses and Rank Statistics for Unbalanced Factorial Designs." *Journal of the American Statistical Association* 92:258–265.

Allen, D. M. (1974). "The Relationship between Variable Selection and Data Augmentation and a Method of Prediction." *Technometrics* 16:125–127.

Bates, D. M., and Watts, D. G. (1988). *Nonlinear Regression Analysis and Its Applications*. New York: John Wiley & Sons.

Beckman, R. J., Nachtsheim, C. J., and Cook, R. D. (1987). "Diagnostics for Mixed-Model Analysis of Variance." *Technometrics* 29:413–426.

Belsley, D. A., Kuh, E., and Welsch, R. E. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. New York: John Wiley & Sons.

Box, G. E. P., and Tiao, G. C. (1973). *Bayesian Inference in Statistical Analysis*. New York: John Wiley & Sons.

Bozdogan, H. (1987). "Model Selection and Akaike's Information Criterion (AIC): The General Theory and Its Analytical Extensions." *Psychometrika* 52:345–370.

Brown, H., and Prescott, R. (1999). *Applied Mixed Models in Medicine*. New York: John Wiley & Sons.

Brownie, C., Bowman, D. T., and Burton, J. W. (1993). "Estimating Spatial Variation in Analysis of Data from Yield Trials: A Comparison of Methods." *Agronomy Journal* 85:1244–1253.

Brownie, C., and Gumpertz, M. L. (1997). "Validity of Spatial Analysis of Large Field Trials." *Journal of Agricultural, Biological, and Environmental Statistics* 2:1–23.

Brunner, E., Dette, H., and Munk, A. (1997). "Box-Type Approximations in Nonparametric Factorial Designs." *Journal of the American Statistical Association* 92:1494–1502.

Brunner, E., Domhof, S., and Langer, F. (2002). *Nonparametric Analysis of Longitudinal Data in Factorial Experiments*. New York: John Wiley & Sons.

Burdick, R. K., and Graybill, F. A. (1992). *Confidence Intervals on Variance Components*. New York: Marcel Dekker.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Carlin, B. P., and Louis, T. A. (1996). *Bayes and Empirical Bayes Methods for Data Analysis*. London: Chapman & Hall.

Carroll, R. J., and Ruppert, D. (1988). *Transformation and Weighting in Regression*. London: Chapman & Hall.

Chilès, J.-P., and Delfiner, P. (1999). *Geostatistics: Modeling Spatial Uncertainty*. New York: John Wiley & Sons.

Christensen, R., Pearson, L. M., and Johnson, W. (1992). "Case-Deletion Diagnostics for Mixed Models." *Technometrics* 34:38–45.

Cook, R. D. (1977). "Detection of Influential Observations in Linear Regression." *Technometrics* 19:15–18.

Cook, R. D. (1979). "Influential Observations in Linear Regression." *Journal of the American Statistical Association* 74:169–174.

Cook, R. D., and Weisberg, S. (1982). *Residuals and Influence in Regression*. New York: Chapman & Hall.

Cressie, N. (1993). *Statistics for Spatial Data*. Rev. ed. New York: John Wiley & Sons.

Crowder, M. J., and Hand, D. J. (1990). *Analysis of Repeated Measures*. New York: Chapman & Hall.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society, Series B* 39:1–38.

Diggle, P. J. (1988). "An Approach to the Analysis of Repeated Measurements." *Biometrics* 44:959–971.

Diggle, P. J., Liang, K.-Y., and Zeger, S. L. (1994). *Analysis of Longitudinal Data*. Oxford: Clarendon Press.

Dunnett, C. W. (1980). "Pairwise Multiple Comparisons in the Unequal Variance Case." *Journal of the American Statistical Association* 75:796–800.

Edwards, D., and Berry, J. J. (1987). "The Efficiency of Simulation-Based Multiple Comparisons." *Biometrics* 43:913–928.

Everitt, B. S. (1995). "The Analysis of Repeated Measures: A Practical Review with Examples." *Journal of the Royal Statistical Society, Series D* 44:113–135. https://doi.org/10.2307/2348622.

Fai, A. H. T., and Cornelius, P. L. (1996). "Approximate F-Tests of Multiple Degree of Freedom Hypotheses in Generalized Least Squares Analyses of Unbalanced Split-Plot Experiments." *Journal of Statistical Computation and Simulation* 54:363–378.

Federer, W. T., and Wolfinger, R. D. (1998). "SAS Code for Recovering Intereffect Information in Experiments with Incomplete Block and Lattice Rectangle Designs." *Agronomy Journal* 90:545–551.

Fuller, W. A. (1976). *Introduction to Statistical Time Series*. New York: John Wiley & Sons.

Fuller, W. A., and Battese, G. E. (1973). "Transformations for Estimation of Linear Models with Nested Error Structure." *Journal of the American Statistical Association* 68:626–632.

Galecki, A. T. (1994). "General Class of Covariance Structures for Two or More Repeated Factors in Longitudinal Data Analysis." *Communications in Statistics—Theory and Methods* 23:3105–3109.

Games, P. A., and Howell, J. F. (1976). "Pairwise Multiple Comparison Procedures with Unequal n's and/or Variances: A Monte Carlo Study." *Journal of Educational Statistics* 1:113–125.

Gelfand, A. E., Hills, S. E., Racine-Poon, A., and Smith, A. F. M. (1990). "Illustration of Bayesian Inference in Normal Data Models Using Gibbs Sampling." *Journal of the American Statistical Association* 85:972–985.

Ghosh, M. (1992). "Discussion of Schervish, M., 'Bayesian Analysis of Linear Models'." In *Bayesian Statistics*, vol. 4, edited by J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, 432–433. Oxford: Clarendon Press.

Giesbrecht, F. G. (1989). *A General Structure for the Class of Mixed Linear Models*. Southern Cooperative Series Bulletin 343, Louisiana Agricultural Experiment Station, Baton Rouge.

Giesbrecht, F. G., and Burns, J. C. (1985). "Two-Stage Analysis Based on a Mixed Model: Large-Sample Asymptotic Theory and Small-Sample Simulation Results." *Biometrics* 41:477–486.

Golub, G. H., and Van Loan, C. F. (1989). *Matrix Computations*. 2nd ed. Baltimore: Johns Hopkins University Press.

Goodnight, J. H. (1978). *Tests of Hypotheses in Fixed-Effects Linear Models*. Technical Report R-101, SAS Institute Inc., Cary, NC.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Goodnight, J. H., and Hemmerle, W. J. (1979). "A Simplified Algorithm for the W-Transformation in Variance Component Estimation." *Technometrics* 21:265–267.

Gotway, C. A., and Stroup, W. W. (1997). "A Generalized Linear Model Approach to Spatial Data and Prediction." *Journal of Agricultural, Biological, and Environmental Statistics* 2:157–187.

Greenhouse, S. W., and Geisser, S. (1959). "On Methods in the Analysis of Profile Data." *Psychometrika* 32:95–112.

Gregoire, T. G., Schabenberger, O., and Barrett, J. P. (1995). "Linear Modelling of Irregularly Spaced, Unbalanced, Longitudinal Data from Permanent Plot Measurements." *Canadian Journal of Forest Research* 25:137–156.

Handcock, M. S., and Stein, M. L. (1993). "A Bayesian Analysis of Kriging." *Technometrics* 35:403–410.

Handcock, M. S., and Wallis, J. R. (1994). "An Approach to Statistical Spatial-Temporal Modeling of Meteorological Fields." *Journal of the American Statistical Association* 89:368–390. With discussion.

Hanks, R. J., Sisson, D. V., Hurst, R. L., and Hubbard, K. G. (1980). "Statistical Analysis of Results from Irrigation Experiments Using the Line-Source Sprinkler System." *Soil Science Society American Journal* 44:886–888.

Hannan, E. J., and Quinn, B. G. (1979). "The Determination of the Order of an Autoregression." *Journal of the Royal Statistical Society, Series B* 41:190–195.

Hartley, H. O., and Rao, J. N. K. (1967). "Maximum-Likelihood Estimation for the Mixed Analysis of Variance Model." *Biometrika* 54:93–108.

Harville, D. A. (1977). "Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems." *Journal of the American Statistical Association* 72:320–338.

Harville, D. A. (1988). "Mixed-Model Methodology: Theoretical Justifications and Future Directions." In *Proceedings of the Statistical Computing Section*, 41–49. Alexandria, VA: American Statistical Association.

Harville, D. A. (1990). "BLUP (Best Linear Unbiased Prediction), and Beyond." In *Advances in Statistical Methods for Genetic Improvement of Livestock*, edited by D. Gianola and K. Hammond, 239–276. Vol. 18 of Advanced Series in Agricultural Sciences. Berlin: Springer-Verlag.

Harville, D. A., and Jeske, D. R. (1992). "Mean Squared Error of Estimation or Prediction under a General Linear Model." *Journal of the American Statistical Association* 87:724–731.

Hemmerle, W. J., and Hartley, H. O. (1973). "Computing Maximum Likelihood Estimates for the Mixed AOV Model Using the W-Transformation." *Technometrics* 15:819–831.

Henderson, C. R. (1984). *Applications of Linear Models in Animal Breeding*. Guelph, ON: University of Guelph.

Henderson, C. R. (1990). "Statistical Method in Animal Improvement: Historical Overview." In *Advances in Statistical Methods for Genetic Improvement of Livestock*, 1–14. New York: Springer-Verlag.

Hirotsu, C., and Srivastava, M. (2000). "Simultaneous Confidence Intervals Based on One-Sided Max *t* Test." *Statistics and Probability Letters* 49:25–37.

Hsu, J. C. (1992). "The Factor Analytic Approach to Simultaneous Inference in the General Linear Model." *Journal of Computational and Graphical Statistics* 1:151–168.

Huber, P. J. (1967). "The Behavior of Maximum Likelihood Estimates under Nonstandard Conditions." *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* 1:221–233.

Hurtado, G. I. (1993). "Detection of Influential Observations in Linear Mixed Models." Ph.D. diss., Department of Statistics, North Carolina State University.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Huynh, H., and Feldt, L. S. (1970). "Conditions Under Which Mean Square Ratios in Repeated Measurements Designs Have Exact F-Distributions." *Journal of the American Statistical Association* 65:1582–1589.

Jennrich, R. I., and Schluchter, M. D. (1986). "Unbalanced Repeated-Measures Models with Structured Covariance Matrices." *Biometrics* 42:805–820.

Johnson, D. E., Chaudhuri, U. N., and Kanemasu, E. T. (1983). "Statistical Analysis of Line-Source Sprinkler Irrigation Experiments and Other Nonrandomized Experiments Using Multivariate Methods." *Soil Science Society American Journal* 47:309–312.

Jones, R. H., and Boadi-Boateng, F. (1991). "Unequally Spaced Longitudinal Data with AR(1) Serial Correlation." *Biometrics* 47:161–175.

Kackar, R. N., and Harville, D. A. (1984). "Approximations for Standard Errors of Estimators of Fixed and Random Effects in Mixed Linear Models." *Journal of the American Statistical Association* 79:853–862.

Kass, R. E., and Steffey, D. (1989). "Approximate Bayesian Inference in Conditionally Independent Hierarchical Models (Parametric Empirical Bayes Models)." *Journal of the American Statistical Association* 84:717–726.

Kenward, M. G. (1987). "A Method for Comparing Profiles of Repeated Measurements." *Journal of the Royal Statistical Society, Series C* 36:296–308.

Kenward, M. G., and Roger, J. H. (1997). "Small Sample Inference for Fixed Effects from Restricted Maximum Likelihood." *Biometrics* 53:983–997.

Kenward, M. G., and Roger, J. H. (2009). "An Improved Approximation to the Precision of Fixed Effects from Restricted Maximum Likelihood." *Computational Statistics and Data Analysis* 53:2583–2595.

Keselman, H. J., Algina, J., Kowalchuk, R. K., and Wolfinger, R. D. (1998). "A Comparison of Two Approaches for Selecting Covariance Structures in the Analysis of Repeated Measures." *Communications in Statistics—Simulation and Computation* 27:591–604.

Keselman, H. J., Algina, J., Kowalchuk, R. K., and Wolfinger, R. D. (1999). "A Comparison of Recent Approaches to the Analysis of Repeated Measurements." *British Journal of Mathematical and Statistical Psychology* 52:63–78.

Kramer, C. Y. (1956). "Extension of Multiple Range Tests to Group Means with Unequal Numbers of Replications." *Biometrics* 12:307–310.

Laird, N. M., Lange, N. T., and Stram, D. O. (1987). "Maximum Likelihood Computations with Repeated Measures: Application of the EM Algorithm." *Journal of the American Statistical Association* 82:97–105.

Laird, N. M., and Ware, J. H. (1982). "Random-Effects Models for Longitudinal Data." *Biometrics* 38:963–974.

LaMotte, L. R. (1973). "Quadratic Estimation of Variance Components." *Biometrics* 29:311–330.

Liang, K.-Y., and Zeger, S. L. (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika* 73:13–22.

Lindsey, J. K. (1993). *Models for Repeated Measurements*. Oxford: Clarendon Press.

Lindstrom, M. J., and Bates, D. M. (1988). "Newton-Raphson and EM Algorithms for Linear Mixed-Effects Models for Repeated-Measures Data." *Journal of the American Statistical Association* 83:1014–1022.

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and Schabenberger, O. (2006). *SAS for Mixed Models*. 2nd ed. Cary, NC: SAS Institute Inc.

Little, R. J. A. (1995). "Modeling the Drop-Out Mechanism in Repeated-Measures Studies." *Journal of the American Statistical Association* 90:1112–1121.

Louis, T. A. (1988). "General Methods for Analyzing Repeated Measures." *Statistics in Medicine* 7:29–45.

Macchiavelli, R. E., and Arnold, S. F. (1994). "Variable Order Ante-dependence Models." *Communications in Statistics—Theory and Methods* 23:2683–2699.

Marx, D., and Thompson, K. (1987). *Practical Aspects of Agricultural Kriging*. Bulletin 903, Arkansas Agricultural Experiment Station, Fayetteville.

Matérn, B. (1986). *Spatial Variation*. 2nd ed. New York: Springer-Verlag.

McKeon, J. J. (1974). "F Approximations to the Distribution of Hotelling's T_0^2." *Biometrika* 61:381–383.

McLean, R. A., and Sanders, W. L. (1988). "Approximating Degrees of Freedom for Standard Errors in Mixed Linear Models." In *Proceedings of the Statistical Computing Section*, 50–59. Alexandria, VA: American Statistical Association.

McLean, R. A., Sanders, W. L., and Stroup, W. W. (1991). "A Unified Approach to Mixed Linear Models." *American Statistician* 45:54–64.

Milliken, G. A., and Johnson, D. E. (1992). *Designed Experiments.* Vol. 1 of Analysis of Messy Data. Reprint edition. New York: Chapman & Hall.

Moriguchi, S., ed. (1976). *Statistical Method for Quality Control*. Tokyo: Japan Standards Association. In Japanese.

Murray, D. M. (1998). *Design and Analysis of Group-Randomized Trials*. New York: Oxford University Press.

Myers, R. H. (1990). *Classical and Modern Regression with Applications*. 2nd ed. Belmont, CA: PWS-Kent.

Obenchain, R. L. (1990). *STABLSIM.EXE, Version 9010.* Unpublished C code. Indianapolis: Eli Lilly.

Patel, H. I. (1991). "Analysis of Incomplete Data from a Clinical Trial with Repeated Measurements." *Biometrika* 78:609–619.

Patterson, H. D., and Thompson, R. (1971). "Recovery of Inter-block Information When Block Sizes Are Unequal." *Biometrika* 58:545–554.

Pillai, K. C. S., and Samson, P., Jr. (1959). "On Hotelling's Generalization of T^2." *Biometrika* 46:160–168.

Pothoff, R. F., and Roy, S. N. (1964). "A Generalized Multivariate Analysis of Variance Model Useful Especially for Growth Curve Problems." *Biometrika* 51:313–326.

Prasad, N. G. N., and Rao, J. N. K. (1990). "The Estimation of Mean Squared Error of Small-Area Estimators." *Journal of the American Statistical Association* 85:163–171.

Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.

Rao, C. R. (1972). "Estimation of Variance and Covariance Components in Linear Models." *Journal of the American Statistical Association* 67:112–115.

Ripley, B. D. (1987). *Stochastic Simulation*. New York: John Wiley & Sons.

Robinson, G. K. (1991). "That BLUP Is a Good Thing: The Estimation of Random Effects." *Statistical Science* 6:15–51.

Rubin, D. B. (1976). "Inference and Missing Data." *Biometrika* 63:581–592.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989). "Design and Analysis of Computer Experiments." *Statistical Science* 4:409–435.

Schabenberger, O., and Gotway, C. A. (2005). *Statistical Methods for Spatial Data Analysis*. Boca Raton, FL: Chapman & Hall/CRC.

Schervish, M. J. (1992). "Bayesian Analysis of Linear Models." In *Bayesian Statistics*, vol. 4, edited by J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, 419–434. Oxford: Clarendon Press.

Schluchter, M. D., and Elashoff, J. D. (1990). "Small-Sample Adjustments to Tests with Unbalanced Repeated Measures Assuming Several Covariance Structures." *Journal of Statistical Computation and Simulation* 37:69–87.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Searle, S. R. (1982). *Matrix Algebra Useful for Statisticians*. New York: John Wiley & Sons.

Searle, S. R. (1988). "Mixed Models and Unbalanced Data: Wherefrom, Whereat, and Whereto?" *Communications in Statistics—Theory and Methods* 17:935–968.

Searle, S. R., Casella, G., and McCulloch, C. E. (1992). *Variance Components*. New York: John Wiley & Sons.

Self, S. G., and Liang, K.-Y. (1987). "Asymptotic Properties of Maximum Likelihood Estimators and Likelihood Ratio Tests under Nonstandard Conditions." *Journal of the American Statistical Association* 82:605–610.

Serfling, R. J. (1980). *Approximation Theorems of Mathematical Statistics*. New York: John Wiley & Sons.

Simpson, S. L., Edwards, L. J., Muller, K. E., Sen, P. K., and Styner, M. A. (2010). "A Linear Exponent AR(1) Family of Correlation Structures." *Statistics in Medicine* 29:1825–1838.

Singer, J. D. (1998). "Using SAS PROC MIXED to Fit Multilevel Models, Hierarchical Models, and Individual Growth Models." *Journal of Educational and Behavioral Statistics* 23:323–355.

Smith, A. F. M., and Gelfand, A. E. (1992). "Bayesian Statistics without Tears: A Sampling-Resampling Perspective." *American Statistician* 46:84–88.

Snedecor, G. W., and Cochran, W. G. (1980). *Statistical Methods*. 7th ed. Ames: Iowa State University Press.

Steel, R. G. D., Torrie, J. H., and Dickey, D. A. (1997). *Principles and Procedures of Statistics: A Biometrical Approach*. 3rd ed. New York: McGraw-Hill.

Stram, D. O., and Lee, J. W. (1994). "Variance Components Testing in the Longitudinal Mixed Effects Model." *Biometrics* 50:1171–1177.

Stroup, W. W. (1989a). "Predictable Functions and Prediction Space in the Mixed Model Procedure." *Applications of Mixed Models in Agriculture and Related Disciplines,* 39–48. Southern Cooperative Series Bulletin No. 343, Louisiana Agricultural Experiment Station, Baton Rouge.

Stroup, W. W. (1989b). "Use of Mixed Model Procedure to Analyze Spatially Correlated Data: An Example Applied to a Line-Source Sprinkler Irrigation Experiment." *Applications of Mixed Models in Agriculture and Related Disciplines,* 104–122. Southern Cooperative Series Bulletin No. 343, Louisiana Agricultural Experiment Station, Baton Rouge.

Stroup, W. W., Baenziger, P. S., and Mulitze, D. K. (1994). "Removing Spatial Variation from Wheat Yield Trials: A Comparison of Methods." *Crop Science* 86:62–66.

Sullivan, L. M., Dukes, K. A., and Losina, E. (1999). "An Introduction to Hierarchical Linear Modelling." *Statistics in Medicine* 18:855–888.

Swallow, W. H., and Monahan, J. F. (1984). "Monte Carlo Comparison of ANOVA, MIVQUE, REML, and ML Estimators of Variance Components." *Technometrics* 26:47–57.

Tamhane, A. C. (1979). "A Comparison of Procedures for Multiple Comparisons of Means with Unequal Variances." *Journal of the American Statistical Association* 74:471–480.

Tierney, L. (1994). "Markov Chains for Exploring Posterior Distributions." *Annals of Statistics* 22:1701–1762.

Verbeke, G., and Molenberghs, G., eds. (1997). *Linear Mixed Models in Practice: A SAS-Oriented Approach*. New York: Springer.

Verbeke, G., and Molenberghs, G. (2000). *Linear Mixed Models for Longitudinal Data*. New York: Springer.

Westfall, P. H., Tobias, R. D., Rom, D., Wolfinger, R. D., and Hochberg, Y. (1999). *Multiple Comparisons and Multiple Tests Using the SAS System*. Cary, NC: SAS Institute Inc.

Westfall, P. H., and Young, S. S. (1993). *Resampling-Based Multiple Testing: Examples and Methods for* p-*Value Adjustment*. New York: John Wiley & Sons.

White, H. (1980). "A Heteroskedasticity-Consistent Covariance Matrix Estimator and a Direct Test for Heteroskedasticity." *Econometrica* 48:817–838.

Whittle, P. (1954). "On Stationary Processes in the Plane." *Biometrika* 41:434–449.

Winer, B. J. (1971). *Statistical Principles in Experimental Design*. 2nd ed. New York: McGraw-Hill.

Wolfinger, R. D. (1993). "Covariance Structure Selection in General Mixed Models." *Communications in Statistics—Simulation and Computation* 22:1079–1106.

Wolfinger, R. D. (1996). "Heterogeneous Variance-Covariance Structures for Repeated Measures." *Journal of Agricultural, Biological, and Environmental Statistics* 1:205–230.

Wolfinger, R. D. (1997). "An Example of Using Mixed Models and PROC MIXED for Longitudinal Data." *Journal of Biopharmaceutical Statistics* 7:481–500.

Wolfinger, R. D., and Chang, M. (1995). "Comparing the SAS GLM and MIXED Procedures for Repeated Measures." In *Proceedings of the Twentieth Annual SAS Users Group Conference*, 1172–1182. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI95/Sugi-95-198%20Wolfinger%20Chang.pdf.

Wolfinger, R. D., Tobias, R. D., and Sall, J. (1991). "Mixed Models: A Future Direction." In *Proceedings of the Sixteenth Annual SAS Users Group Conference*, 1380–1388. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI91/Sugi-91-249%20Wolfinger%20Tobias%20Sall.pdf.

Wolfinger, R. D., Tobias, R. D., and Sall, J. (1994). "Computing Gaussian Likelihoods and Their Derivatives for General Linear Mixed Models." *SIAM Journal on Scientific Computing* 15:1294–1310.

Wright, S. P. (1994). *Adjusted* F *Tests for Repeated Measures with the MIXED Procedure*. Knoxville: Statistics Department, University of Tennessee.

Zimmerman, D. L., and Harville, D. A. (1991). "A Random Field Approach to the Analysis of Field-Plot Experiments and Other Spatial Experiments." *Biometrics* 47:223–239.

Chapter 85
The MODECLUS Procedure

Contents

Overview: MODECLUS Procedure **6962**
Getting Started: MODECLUS Procedure **6963**
Syntax: MODECLUS Procedure **6968**
- PROC MODECLUS Statement 6968
- BY Statement 6975
- FREQ Statement 6975
- ID Statement 6976
- VAR Statement 6976

Details: MODECLUS Procedure **6976**
- Density Estimation 6976
- Clustering Methods 6979
- Significance Tests 6982
- Computational Resources 6987
- Missing Values 6987
- Output Data Sets 6988
- Displayed Output 6990
- ODS Table Names 6993

Examples: MODECLUS Procedure **6994**
- Example 85.1: Cluster Analysis of Samples from Univariate Distributions 6994
- Example 85.2: Cluster Analysis of Flying Mileages between Ten American Cities . . 7017
- Example 85.3: Cluster Analysis with Significance Tests 7024
- Example 85.4: Cluster Analysis: Hertzsprung-Russell Plot 7030
- Example 85.5: Using the TRACE Option When METHOD=6 7034

References **7038**

Overview: MODECLUS Procedure

The MODECLUS procedure clusters observations in a SAS data set by using any of several algorithms based on nonparametric density estimates. The data can be numeric coordinates or distances. PROC MODECLUS can perform approximate significance tests for the number of clusters and can hierarchically join nonsignificant clusters. The significance tests are empirically validated by simulations with sample sizes ranging from 20 to 2000.

PROC MODECLUS produces output data sets containing density estimates and cluster membership, various cluster statistics including approximate *p*-values, and a summary of the number of clusters generated by various algorithms, smoothing parameters, and significance levels.

Most clustering methods are biased toward finding clusters possessing certain characteristics related to size (number of members), shape, or dispersion. Methods based on the least squares criterion (Sarle 1982), such as *k*-means and Ward's minimum variance method, tend to find clusters with roughly the same number of observations in each cluster. Average linkage (see Chapter 40, "The CLUSTER Procedure") is somewhat biased toward finding clusters of equal variance. Many clustering methods tend to produce compact, roughly hyperspherical clusters and are incapable of detecting clusters with highly elongated or irregular shapes. The methods with the least bias are those based on nonparametric density estimation (Silverman 1986, pp. 130–146; Scott 1992, pp. 125–190) such as density linkage (see Chapter 40, "The CLUSTER Procedure"), Wong and Lane (1983); Wong and Schaack (1982). The biases of many commonly used clustering methods are discussed in Chapter 12, "Introduction to Clustering Procedures."

PROC MODECLUS implements several clustering methods by using nonparametric density estimation. Such clustering methods are referred to hereafter as *nonparametric clustering methods*. The methods in PROC MODECLUS are related to, but not identical to, methods developed by Gitman (1973); Huizinga (1978); Koontz and Fukunaga (1972a, b); Koontz, Narendra, and Fukunaga (1976); Mizoguchi and Shimura (1980); Wong and Lane (1983).

Details of the algorithms are provided in the section "Clustering Methods" on page 6979.

For nonparametric clustering methods, a cluster is loosely defined as a region surrounding a local maximum of the probability density function (see the section "Significance Tests" on page 6982 for a more rigorous definition). Given a sufficiently large sample, nonparametric clustering methods are capable of detecting clusters of unequal size and dispersion and with highly irregular shapes. Nonparametric methods can also obtain good results for compact clusters of equal size and dispersion, but they naturally require larger sample sizes for good recovery than clustering methods that are biased toward finding such "nice" clusters.

For coordinate data, nonparametric clustering methods are less sensitive to changes in scale of the variables or to affine transformations of the variables than are most other commonly used clustering methods. Nevertheless, it is necessary to consider questions of scaling and transformation, since variables with large variances tend to have more of an effect on the resulting clusters than those with small variances. If two or more variables are not measured in comparable units, some type of standardization or scaling is necessary; otherwise, the distances used by the procedure might be based on inappropriate apples-and-oranges computations. For variables with comparable units of measurement, standardization or scaling might still be desirable if the scale estimates of the variables are not related to their expected importance for defining clusters. If you want two variables to have equal importance in the analysis, they should have roughly equal scale estimates. If you want one variable to have more of an effect than another, the former should be scaled to have a greater scale estimate than the latter. The STD option in the PROC MODECLUS statement scales all variables to equal variance. However, the variance is not necessarily the most appropriate scale estimate for cluster analysis. In

particular, outliers should be removed before using PROC MODECLUS with the STD option. A variety of scale estimators including robust estimators are provided in the STDIZE procedure (for detailed information, see Chapter 115, "The STDIZE Procedure"). Additionally, the ACECLUS procedure provides another way to transform the variables to try to improve the separation of clusters.

Since clusters are defined in terms of local maxima of the probability density function, nonlinear transformations of the data can change the number of population clusters. The variables should be transformed so that equal differences are of equal practical importance. An interval scale of measurement is required. Ordinal or ranked data are generally inappropriate, since monotone transformations can produce any arbitrary number of modes.

Unlike the methods in the CLUSTER procedure, the methods in the MODECLUS procedure are not inherently hierarchical. However, PROC MODECLUS can do approximate nonparametric significance tests for the number of clusters by obtaining an approximate p-value for each cluster, and it can hierarchically join nonsignificant clusters.

Another important difference between the MODECLUS procedure and many other clustering methods is that you do not tell PROC MODECLUS how many clusters you want. Instead, you specify a *smoothing parameter* (see the section "Density Estimation" on page 6976) and, optionally, a significance level, and PROC MODECLUS determines the number of clusters. You can specify a list of smoothing parameters, and PROC MODECLUS performs a separate cluster analysis for each value in the list.

Getting Started: MODECLUS Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/modeclus.sas.)

This section illustrates how PROC MODECLUS can be used to examine the clusters of data in the following artificial data set.

```
data example;
   input x y @@;
   datalines;
18 18  20 22  21 20  12 23  17 12  23 25  25 20  16 27
20 13  28 22  80 20  75 19  77 23  81 26  55 21  64 24
72 26  70 35  75 30  78 42  18 52  27 57  41 61  48 64
59 72  69 72  80 80  31 53  51 69  72 81
;
```

It is a good practice to plot the data to check for obvious clusters or pathologies prior to the analysis. In this example, with only two variables and a small sample size, the SGPLOT procedure in the following statements produces a scatter plot:

```
proc sgplot;
   scatter y=y x=x;
run;
```

Figure 85.1 suggests three clusters. Of these clusters, the one in the lower-left corner is the most compact, while the lower-right cluster is more dispersed.

The upper cluster is elongated and would be difficult for most clustering algorithms to identify as a single cluster. The plot also suggests that a Euclidean distance of 10 or 20 is a good initial guess for the neighborhood size in density estimation and clustering.

Figure 85.1 Scatter Plot of Data

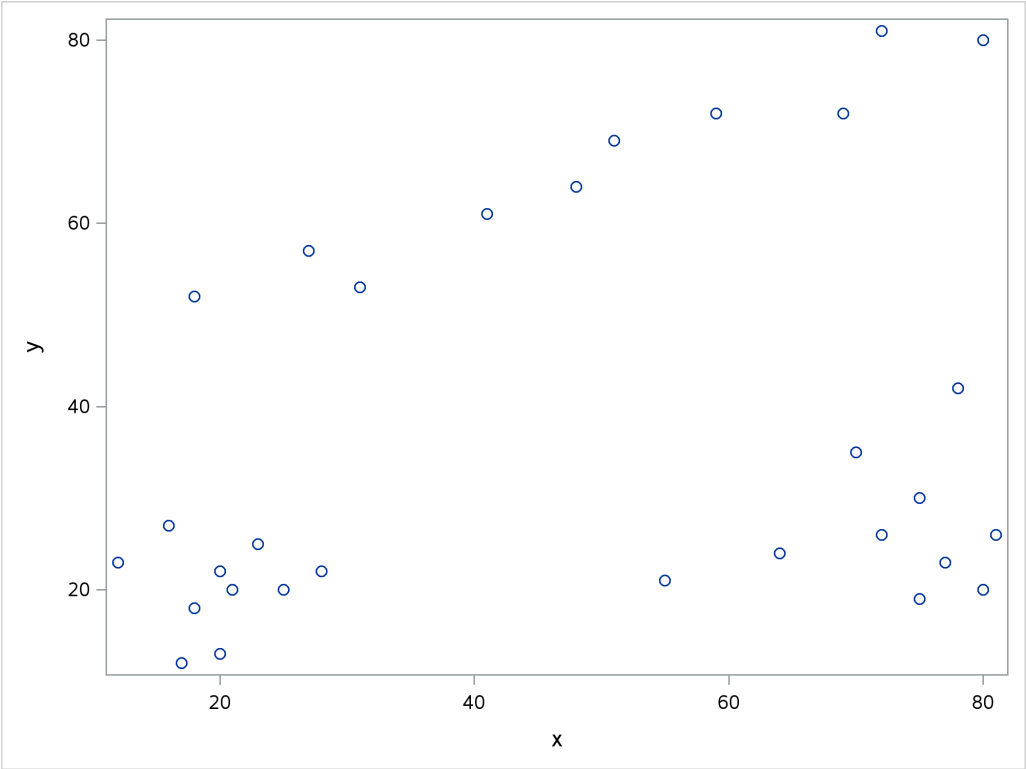

To obtain a cluster analysis in PROC MODECLUS, you must specify the METHOD= option; for most purposes, METHOD=1 is recommended. The cluster analysis can be performed with a list of radii (R=10 15 35), as shown in the following PROC MODECLUS statement. An output data set containing the cluster membership is created with the OUT= option. The following statements produce Figure 85.2 through Figure 85.5:

```
proc modeclus data=example method=1 r=10 15 35 out=out;
run;
```

For each cluster solution, PROC MODECLUS produces a table of cluster statistics including the cluster number, the number of observations in the cluster, the maximum estimated density within the cluster, the number of observations in the cluster having a neighbor that belongs to a different cluster, and the estimated saddle density of the cluster. The results are displayed in Figure 85.2, Figure 85.3, and Figure 85.4 for three different radii. A smaller radius (R=10) yields a larger number of clusters (6), as displayed in Figure 85.2; a larger radius (R=35) includes all observations in a single cluster, as displayed in Figure 85.4. Note that all clusters in these three figures are “isolated” since their corresponding boundary frequencies are all zeros. Consequently, all the estimated saddle densities are missing. A table summarizing each cluster solution is then produced at the end, as displayed in Figure 85.5.

Figure 85.2 Results from PROC MODECLUS for METHOD=1 and R=10

The MODECLUS Procedure
R=10 METHOD=1

Cluster Statistics				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density
1	10	0.00106103	0	.
2	9	0.00084883	0	.
3	7	0.00031831	0	.
4	2	0.00021221	0	.
5	1	0.0001061	0	.
6	1	0.0001061	0	.

Figure 85.3 Results from PROC MODECLUS for METHOD=1 and R=15

The MODECLUS Procedure
R=15 METHOD=1

Cluster Statistics				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density
1	10	0.00047157	0	.
2	10	0.00042441	0	.
3	10	0.00023579	0	.

Figure 85.4 Results from PROC MODECLUS for METHOD=1 and R=35

The MODECLUS Procedure
R=35 METHOD=1

Cluster Statistics				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density
1	30	0.00012126	0	.

Figure 85.5 Summary Table

The MODECLUS Procedure

Cluster Summary		
R	Number of Clusters	Frequency of Unclassified Objects
10	6	0
15	3	0
35	1	0

The OUT= data set contains a complete copy of the input data set for each cluster solution. By using a BY statement in the following PROC SGPLOT statement, you can examine the differences in cluster memberships for each radius as shown in Figure 85.6 through Figure 85.8:

```
proc sgplot data=out noautolegend;
   scatter y=y x=x / group=cluster markerchar=cluster;
   by _r_;
run;
```

Figure 85.6 Scatter Plots of Cluster Memberships with _R_=10

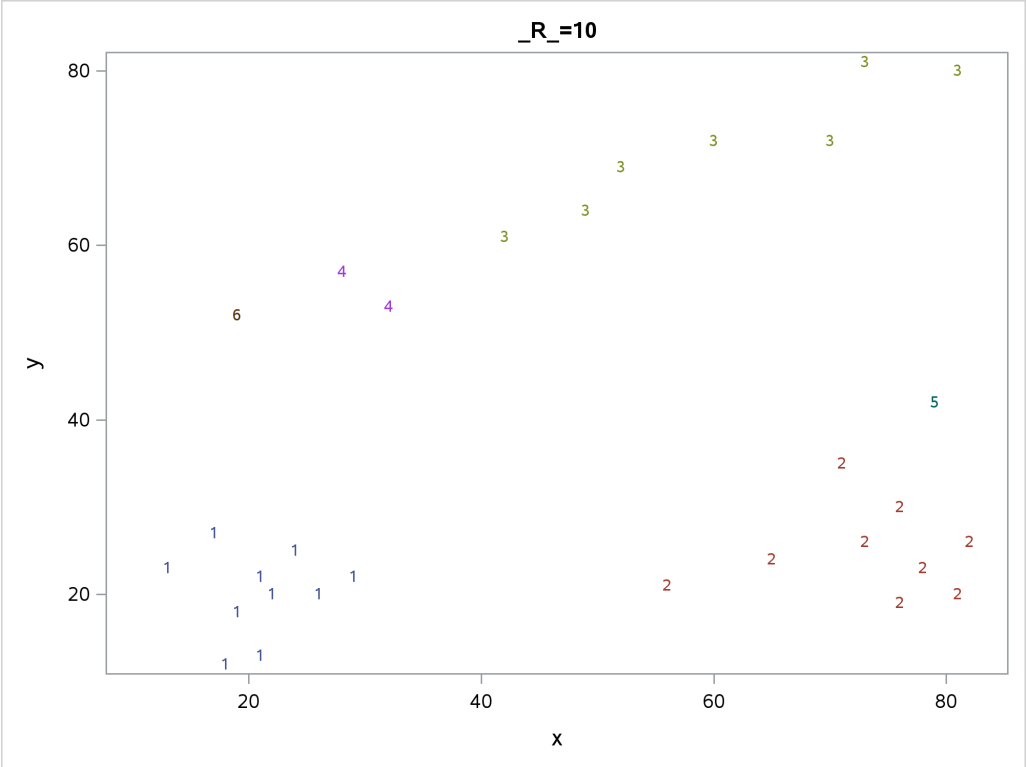

Figure 85.7 Scatter Plots of Cluster Memberships with _R_=15

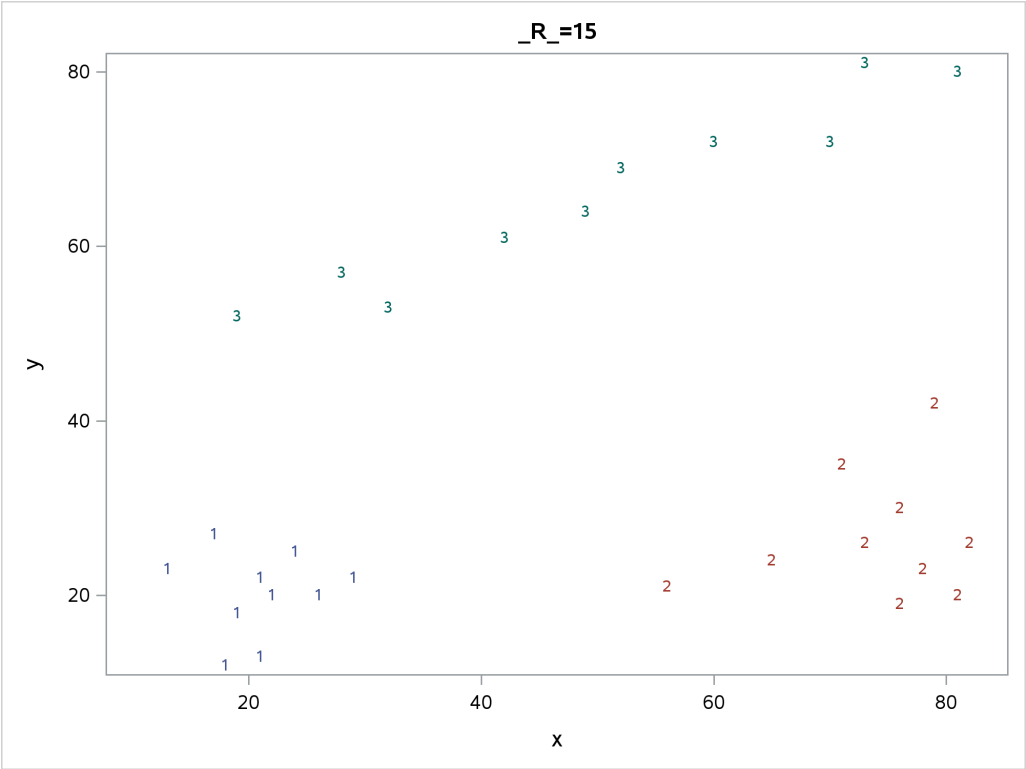

Figure 85.8 Scatter Plots of Cluster Memberships with _R_=35

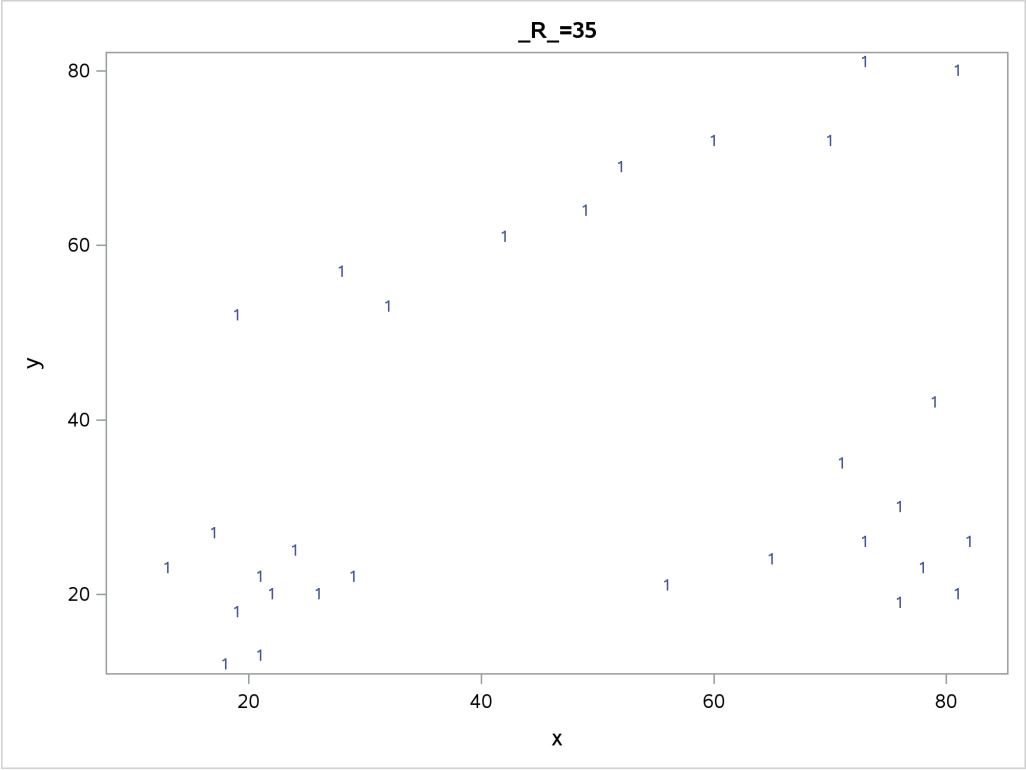

Syntax: MODECLUS Procedure

The following statements are available in the MODECLUS procedure:

```
PROC MODECLUS <options> ;
    BY variables ;
    FREQ variable ;
    ID variable ;
    VAR variables ;
```

The PROC MODECLUS statement is required. All other statements are optional.

PROC MODECLUS Statement

```
PROC MODECLUS <options> ;
```

The PROC MODECLUS statement invokes the MODECLUS procedure. Table 85.1 summarizes the options available in the PROC MODECLUS statement. These options are discussed in the following sections.

Table 85.1 Summary of PROC MODECLUS Statement Options

Option	Description
Specify Input and Output Data Sets	
DATA=	Specifies input data set name
OUT=	Specifies output data set name for observations
OUTCLUS=	Specifies output data set name for clusters
OUTSUM=	Specifies output data set name for cluster solutions
Specify Variables in Output Data Sets	
CLUSTER=	Specifies variable in the OUT= and OUTCLUS= data sets identifying clusters
DENSITY=	Specifies variable in the OUT= data set containing density estimates
OUTLENGTH=	Specifies length of variables in the output data sets
Summarize and Process Coordinate Data before Clustering	
SIMPLE	Requests simple statistics
STANDARD	Standardizes the variables to mean 0 and standard deviation 1
Specify Smoothing Parameters	
DK=	Specifies number of neighbors to use for *k*th-nearest-neighbor density estimation
CK=	Specifies number of neighbors to use for clustering
K=	Specifies number of neighbors to use for *k*th-nearest-neighbor density estimation and clustering
DR=	Specifies radius of the sphere of support for uniform-kernel density estimation
CR=	Specifies radius of the neighborhood for clustering
R=	Specifies radius of the sphere of support for uniform-kernel density estimation and the neighborhood clustering
Specify Density Estimation Options	
CASCADE=	Specifies number of times the density estimates are to be cascaded

Table 85.1 *continued*

Option	Description
DIMENSION=	Specifies dimensionality to be used when computing density estimates
AM	Uses arithmetic means for cascading density estimates
HM	Uses harmonic means for cascading density estimates
SUM	Uses sums for cascading density estimates
Specify Clustering Methods and Options	
DOCK=	Dissolves clusters with *n* or fewer members
EARLY	Stops the analysis after obtaining a solution with either no cluster or a single cluster
JOIN=	Requests that nonsignificant clusters be hierarchically joined
MAXCLUSTERS=	Specifies maximum number of clusters to be obtained with METHOD=6
METHOD=	Specifies clustering method to use
MODE=	Specifies minimum members for either cluster to be designated a modal cluster when two clusters are joined using METHOD=5
POWER=	Specifies power of the density used with METHOD=6
TEST	Specifies approximate significance tests for the number of clusters
THRESHOLD=	Specifies assignment threshold used with METHOD=6
Specify the Output Display Options	
ALL	Produces all optional output
BOUNDARY	Displays the density and cluster membership of observations with neighbors belonging to a different cluster
CORE	Retains the neighbor lists for each observation in memory
CROSS	Displays the estimated cross validated log density of each observation
CROSSLIST	Displays the estimated density and cluster membership of each observation
LOCAL	Displays estimates of local dimensionality and writes them to the OUT=data set
NEIGHBOR	Displays the neighbors of each observation
NOPRINT	Suppresses the display of the output
NOSUMMARY	Suppresses the display of the summary of the number of clusters, number of unassigned observations, and maximum *p*-value for each analysis
SHORT	Suppresses the display of statistics for each cluster
TRACE	Traces the cluster assignments when METHOD=6

You can specify at least one of the following options for smoothing parameters for density estimation: DK=, K=, DR=, or R=. To obtain a cluster analysis, you can specify the METHOD= option and at least one of the following smoothing parameters for clustering: CK=, K=, CR=, or R=. If you want significance tests for the number of clusters, you should specify either the DR= or R= option. If none of the smoothing parameters is specified, the MODECLUS procedure provides a default value for the R= option. See the section "Density Estimation" on page 6976 for the formula of a reasonable first guess for R= and a discussion of smoothing parameters.

You can specify lists of values for the DK=, CK=, K=, DR=, CR=, and R= options. Numbers in the lists can be separated by blanks or commas. You can include in the lists one or more items of the form *start* TO *stop* BY *increment*. Each list can contain either one value or the same number of values as in every other list that contains more than one value. If a list has only one value, that value is used in combination with all the

values in longer lists. If two or more lists have more than one value, then one analysis is done by using the first value in each list, another analysis is done by using the second value in each list, and so on.

You can specify the following options in the PROC MODECLUS statement.

ALL
produces all optional output.

AM
specifies arithmetic means for cascading density estimates. See the description of the CASCADE= option.

BOUNDARY
displays the density and cluster membership of observations with neighbors belonging to a different cluster.

CASCADE=*n*
CASC=*n*
specifies the number of times the density estimates are to be cascaded (see the section "Density Estimation" on page 6976). The default value 0 performs no cascading.

You can specify a list of values for the CASCADE= option. Each value in the list is combined with each combination of smoothing parameters to produce a separate analysis.

CK=*n*
specifies the number of neighbors to use for clustering. The number of neighbors should be at least two but less than the number of observations. See the section "Density Estimation" on page 6976 for details.

CLUSTER=*name*
provides a name for the variable in the OUT= and OUTCLUS= data sets identifying clusters. The default name is CLUSTER.

CORE
keeps the neighbor lists for each observation in the computer memory to make small problems run faster.

CR=*n*
specifies the radius of the neighborhood for clustering. See the section "Density Estimation" on page 6976 for details.

CROSS
computes the likelihood cross validation criterion (Silverman 1986, pp. 52–55). This option appears to be of limited usefulness. See the section "Density Estimation" on page 6976 for details.

CROSSLIST
displays the cross validated log density of each observation.

DATA=*SAS-data-set*

specifies the input data set containing observations to be clustered. If you omit the DATA= option, the most recently created SAS data set is used.

If the data set is TYPE=DISTANCE, the data are interpreted as a distance matrix. The number of variables must equal the number of observations in the data set or in each BY group. The distances are assumed to be Euclidean, but the procedure accepts other types of distances or dissimilarities. Unlike the CLUSTER procedure, PROC MODECLUS uses the entire distance matrix, not just the lower triangle; the distances are not required to be symmetric. The neighbors of a given observation are determined solely from the distances in that observation. Missing values are considered infinite. Various distance measures can be computed from coordinate data by using the DISTANCE procedure (for detailed information, see Chapter 43, "The DISTANCE Procedure").

If the data set is not TYPE=DISTANCE, the data are interpreted as coordinates in a Euclidean space, and Euclidean distances are computed. The variables can be discrete or continuous and should be at the interval level of measurement.

Data set types such as TYPE=DISTANCE do not persist when you copy or modify a data set. You must specify the TYPE= data set option for the new data set, as in the following example:

```
data dist2(type=distance);
   set dist;
run;
```

If you do not specify the TYPE=DISTANCE data set option, the new data set is the default TYPE=DATA. If you use the new data set in a procedure that accepts both TYPE=DATA or TYPE=DISTANCE data sets (such as PROC CLUSTER or PROC MODECLUS), the results will be incorrect.

DENSITY=*name*

provides a name for the variable in the OUT= data set containing density estimates. The default name is DENSITY.

DIMENSION=*n*

DIM=*n*

specifies the dimensionality to be used when computing density estimates. The default is the number of variables if the data are coordinates; the default is 1 if the data are distances.

DK=*n*

specifies the number of neighbors to use for *k*th-nearest-neighbor density estimation. The number of neighbors should be at least two but less than the number of observations. See the section "Density Estimation" on page 6976 for details.

DOCK=*n*

dissolves clusters with *n* or fewer members by making the members unassigned.

DR=*n*

specifies the radius of the sphere of support for uniform-kernel density estimation. See the section "Density Estimation" on page 6976 for details.

EARLY

stops the cluster analysis after obtaining either a solution with no cluster or a solution with one cluster to which all observations are assigned. The smoothing parameters should be specified in increasing order. This can reduce the computer time required for the analysis but might occasionally miss some multiple-cluster solutions.

HM

uses harmonic means for cascading density estimates. See the description of the CASCADE= option for details.

JOIN< =*p* >

requests that nonsignificant clusters be hierarchically joined. The JOIN option implies the TEST option. After each solution is obtained, the cluster with the largest approximate *p*-value is either joined to a neighboring cluster or, if there is no neighboring cluster, dissolved by making all of its members unassigned. After two clusters are joined, an analysis of the remaining clusters is displayed.

If you do not specify a *p*-value with the JOIN= option, joining continues until only one cluster remains, and the results are written to the output data sets after each analysis. If you specify a *p*-value with the JOIN= option, joining continues until the greatest approximate *p*-value is less than the value given in the JOIN= option, and only if there is more than one cluster are the results for that analysis written to the output data sets.

Any value of *p* less than 1E–8 is set to 1E–8.

K=*n*

specifies the number of neighbors to use for *k*th-nearest-neighbor density estimation and clustering. The number of neighbors should be at least two but less than the number of observations. Specifying K=*n* is equivalent to specifying both DK=*n* and CK=*n*. See the section "Density Estimation" on page 6976 for details.

LIST

displays the estimated density and cluster membership of each observation.

LOCAL

requests estimates of local dimensionality (Tukey and Tukey 1981, pp. 236–237).

MAXCLUSTERS=*n*

MAXC=*n*

specifies the maximum number of clusters to be obtained with the METHOD=6 option. By default, there is no fixed limit.

METHOD=*n*

MET=*n*

M=*n*

specifies what clustering method to use. Since these methods do not have widely recognized names, the methods are indicated by numbers from 0 to 6. The methods are described in the section "Clustering Methods" on page 6979. For most purposes, METHOD=1 is recommended, although METHOD=6

might occasionally produce better results in return for considerably greater computer time and space requirements. METHOD=1 is not good for discrete coordinate data with only a few equally spaced values. In this case, METHOD=6 or METHOD=3 works better. METHOD=4 or METHOD=5 is less desirable than other methods when there are ties, since a general characteristic of agglomerative hierarchical clustering methods is that the results are indeterminate in the presence of ties.

You must specify the METHOD= option to obtain a cluster analysis.

You can specify a list of values for the METHOD= option. Each value in the list is combined with each combination of smoothing and cascading parameters to produce a separate cluster analysis.

MODE=*n*

specifies that when two clusters are joined using the METHOD=5 option (no other methods are affected by the MODE= option), each must have at least *n* members for either cluster to be designated a modal cluster. In any case, each cluster must also have a maximum density greater than the fusion density for either cluster to be designated a modal cluster. If you specify the K= option, the default value of the MODE= option is the same as the value of the K= option because the use of *k*th-nearest-neighbor density estimation limits the resolution that can be obtained for clusters with fewer than *k* members. If you do not specify the K= option, the default is MODE=2. If you specify MODE=0, the default value is used instead of 0. If you specify a FREQ statement, the MODE= value is compared to the number of observations in each cluster, not to the sum of the frequencies.

NEIGHBOR

displays the neighbors of each observation in a table called "Nearest Neighbor List." See Nearest Neighbor List for information displayed in the table.

NOPRINT

suppresses the display of the output. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

NOSUMMARY

suppresses the display of the summary of the number of clusters, number of unassigned observations, and maximum *p*-value for each analysis.

OUT=*SAS-data-set*

specifies the output data set containing the input data plus density estimates, cluster membership, and variables identifying the type of solution. There is an output observation corresponding to each input observation for each solution. Therefore, the OUT= data set can be very large.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For details about OUT= data sets, see the section "Output Data Sets" on page 6988.

OUTCLUS=*SAS-data-set*

OUTC=*SAS-data-set*

specifies the output data set containing an observation corresponding to each cluster in each solution. The variables identify the solution and contain statistics describing the clusters.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For details about OUTCLUS= data sets, see the section "Output Data Sets" on page 6988.

OUTSUM=*SAS-data-set*

OUTS=*SAS-data-set*

specifies the output data set containing an observation corresponding to each cluster solution, giving the number of clusters and the number of unclassified observations for that solution.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For details about OUTSUM= data sets, see the section "Output Data Sets" on page 6988.

OUTLENGTH=*n*

OUTL=*n*

specifies the length of those output variables that are not copied from the input data set but are created by PROC MODECLUS.

The OUTLENGTH= option applies only to the following variables that appear in all of the output data sets: _K_, _DK_, _CK_, _R_, _DR_, _CR_, _CASCAD_, _METHOD_, _NJOIN_, and _LOCAL_.

The minimum value is 2 or 3, depending on the operating system. The maximum value is 8. The default value is 8.

POWER=*n*

POW=*n*

specifies the power of the density used with the METHOD=6 option. The default value is 2.

R=*n*

specifies the radius of the sphere of support for uniform-kernel density estimation and the neighborhood for clustering. Specifying R=*n* is equivalent to specifying both DR=*n* and CR=*n*. See the section "Density Estimation" on page 6976 for details.

SHORT

suppresses the display of statistics for each cluster.

SIMPLE

S

displays means, standard deviations, skewness, kurtosis, and a coefficient of bimodality. The SIMPLE option applies only to coordinate data.

STANDARD

STD

standardizes the variables to mean 0 and standard deviation 1. The STANDARD option applies only to coordinate data.

SUM

uses sums for cascading density estimates. See the description of the CASCADE= option for details.

TEST

performs approximate significance tests for the number of clusters. The R= or DR= option must also be specified with a nonzero value to obtain significance tests.

The significance tests performed by PROC MODECLUS are valid only for simple random samples, and they require at least 20 observations per cluster to have enough power to be of any use. See the section "Significance Tests" on page 6982 for details.

THRESHOLD=*n*

THR=*n*

specifies the assignment threshold used with the METHOD=6 option. The default is 0.5.

TRACE

traces the process of cluster assignments when METHOD=6 is specified.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC MODECLUS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the MODECLUS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If one variable in the input data set represents the frequency of occurrence for other values in the observation, specify the variable's name in a FREQ statement. PROC MODECLUS then treats the data set as if each observation appeared *n* times, where *n* is the value of the FREQ variable for the observation. Nonintegral values of the FREQ variable are truncated to the largest integer less than the FREQ value.

ID Statement

ID *variable* **;**

The values of the ID variable identify observations in the displayed results and in the OUT= data set. If you omit the ID statement, each observation is identified by its observation number, and a variable called _OBS_ is written to the OUT= data set containing the original observation numbers.

VAR Statement

VAR *variables* **;**

The VAR statement specifies numeric variables to be used in the cluster analysis. If you omit the VAR statement, all numeric variables not specified in other statements are used.

Details: MODECLUS Procedure

Density Estimation

See Silverman (1986) or Scott (1992) for an introduction to nonparametric density estimation.

PROC MODECLUS uses hyperspherical uniform kernels of fixed or variable radius. The density estimate at a point is computed by dividing the number of observations within a sphere centered at the point by the product of the sample size and the volume of the sphere. The size of the sphere is determined by the smoothing parameters that you are required to specify.

For fixed-radius kernels, specify the radius as a Euclidean distance with either the DR= or R= option. For variable-radius kernels, specify the number of neighbors desired within the sphere with either the DK= or K= option; the radius is then the smallest radius that contains at least the specified number of observations including the observation at which the density is being estimated. If you specify both the DR= or R= option and the DK= or K= option, the radius used is the maximum of the two indicated radii; this is useful for dealing with outliers.

It is convenient to refer to the sphere of support of the kernel at observation x_i as the *neighborhood* of x_i. The observations within the neighborhood of x_i are the *neighbors* of x_i. In some contexts, x_i is considered a neighbor of itself, but in other contexts it is not. The following notation is used in this chapter:

x_i	the ith observation
d(x,y)	the distance between points x and y
n	the total number of observations in the sample
n_i	the number of observations within the neighborhood of x_i, including x_i itself
n_i^-	the number of observations within the neighborhood of x_i, not including x_i itself
N_i	the set of indices of neighbors of x_i, including i

N_i^-	the set of indices of neighbors of x_i, not including i
v_i	the volume of the neighborhood of x_i
r_i	the radius of the neighborhood of x_i
$\hat{f}_i$	the estimated density at x_i
$\hat{f}_i^-$	the cross validated density estimate at x_i
C_k	the set of indices of observations assigned to cluster k
v	the number of variables or the dimensionality
s_l	standard deviation of the lth variable

The estimated density at x_i is

$$\hat{f}_i = \frac{n_i}{n v_i}$$

which indicates the number of neighbors of x_i divided by the product of the sample size and the volume of the neighborhood at x_i, where

$$v_i = \frac{\pi^{\frac{v}{2}} {r_i}^v}{\Gamma(\frac{v}{2}+1)}$$

and Γ can be computed in a DATA step by using the GAMMA function. Note that $v = 1$ for distance data.

The density estimates provided by uniform kernels are not quite as good as those provided by some other types of kernels, but they are quite satisfactory for clustering. The significance tests for the number of clusters require the use of fixed-size uniform kernels.

There is no simple answer to the question of which smoothing parameter to use (Silverman 1986, pp. 43–61, 84–88, 98–99). It is usually necessary to try several different smoothing parameters. A reasonable first guess for the K= option is in the range of 0.1 to 1 times $n^{4/(v+4)}$, smaller values being suitable for higher dimensionalities. A reasonable first guess for the R= option in many coordinate data sets is given by

$$\left[\frac{2^{v+2}(v+2)\Gamma(\frac{v}{2}+1)}{n v^2}\right]^{1/(v+4)} \sqrt{\sum_{l=1}^{v} s_l^2}$$

which can be computed in a DATA step by using the GAMMA function for Γ. The MODECLUS procedure also provides this first guess as a default smoothing parameter if none of the options (DR=, CR=, R=, DK=, CK=, and K=) is specified. This formula is derived under the assumption that the data are sampled from a multivariate normal distribution and, therefore, tend to be too large (oversmooth) if the true distribution is multimodal. Robust estimates of the standard deviations might be preferable if there are outliers. If the data are distances, the factor $\sqrt{\sum {s_l}^2}$ can be replaced by an average root-mean-squared Euclidean distance divided by $\sqrt{2}$. To prevent outliers from appearing as separate clusters, you can also specify K=2 or CK=2 or, more generally, K=m or CK=m, $m \geq 2$, which in most cases forces clusters to have at least m members.

If the variables all have unit variance (for example, if you specify the STD option), you can use Table 85.2 to obtain an initial guess for the R= option.

Table 85.2 Reasonable First Guess for R= for Standardized Data

Number of Obs	Number of Variables 1	2	3	4	5	6	7	8	9	10
20	1.01	1.36	1.77	2.23	2.73	3.25	3.81	4.38	4.98	5.60
35	0.91	1.24	1.64	2.08	2.56	3.08	3.62	4.18	4.77	5.38
50	0.84	1.17	1.56	1.99	2.46	2.97	3.50	4.06	4.64	5.24
75	0.78	1.09	1.47	1.89	2.35	2.85	3.38	3.93	4.50	5.09
100	0.73	1.04	1.41	1.82	2.28	2.77	3.29	3.83	4.40	4.99
150	0.68	0.97	1.33	1.73	2.18	2.66	3.17	3.71	4.27	4.85
200	0.64	0.93	1.28	1.67	2.11	2.58	3.09	3.62	4.17	4.75
350	0.57	0.85	1.18	1.56	1.98	2.44	2.93	3.45	4.00	4.56
500	0.53	0.80	1.12	1.49	1.91	2.36	2.84	3.35	3.89	4.45
750	0.49	0.74	1.06	1.42	1.82	2.26	2.74	3.24	3.77	4.32
1000	0.46	0.71	1.01	1.37	1.77	2.20	2.67	3.16	3.69	4.23
1500	0.43	0.66	0.96	1.30	1.69	2.11	2.57	3.06	3.57	4.11
2000	0.40	0.63	0.92	1.25	1.63	2.05	2.50	2.99	3.49	4.03

One data-based method for choosing the smoothing parameter is likelihood cross validation (Silverman 1986, pp. 52–55). The cross validated density estimate at an observation is obtained by omitting the observation from the computations:

$$\hat{f}_i^- = \frac{n_i^-}{n v_i}$$

The (log) likelihood cross validation criterion is then computed as

$$\sum_{i=1}^{n} \log \hat{f}_i^-$$

The suggested smoothing parameter is the one that maximizes this criterion. With fixed-radius kernels, likelihood cross validation oversmooths long-tailed distributions; for purposes of clustering, it tends to undersmooth short-tailed distributions. With k-nearest-neighbor density estimation, likelihood cross validation is useless because it almost always indicates k=2.

Cascaded density estimates are obtained by computing initial kernel density estimates and then, at each observation, taking the arithmetic mean, harmonic mean, or sum of the initial density estimates of the observations within the neighborhood. The cascaded density estimates can, in turn, be cascaded, and so on. Let ${}_k\hat{f}_i$ be the density estimate at x_i cascaded k times. For all types of cascading, ${}_0\hat{f}_i = \hat{f}_i$. If the cascading is done by arithmetic means, then, for $k \geq 0$,

$${}_{k+1}\hat{f}_i = \sum_{j \in N_i} {}_k\hat{f}_j / n_i$$

For harmonic means,

$${}_{k+1}\hat{f}_i = \left(\sum_{j \in N_i} {}_k\hat{f}_j^{-1} / n_i \right)^{-1}$$

and for sums,

$$ {}_{k+1}\hat{f}_i = \left(\sum_{j \in N_i} {}_k\hat{f}_j^{k+1} \right)^{\frac{1}{k+2}} $$

To avoid cluttering formulas, the symbol $\hat{f}_i$ is used in the rest of the chapter to denote the density estimate at x_i whether cascaded or not, since the clustering methods and significance tests do not depend on the degree of cascading.

Cascading increases the smoothness of the estimates with less computation than would be required by increasing the smoothing parameters to yield a comparable degree of smoothness. For population densities with bounded support and discontinuities at the boundaries, cascading improves estimates near the boundaries. Cascaded estimates, especially using sums, might be more sensitive to the local covariance structure of the distribution than are the uncascaded kernel estimates. Cascading seems to be useful for detecting very nonspherical clusters. Cascading was suggested by Tukey and Tukey (1981, p. 237). Additional research into the properties of cascaded density estimates is needed.

Clustering Methods

The number of clusters is a function of the smoothing parameters. The number of clusters tends to decrease as the smoothing parameters increase, but the relationship is not strictly monotonic. Generally, you should specify several different values of the smoothing parameters to see how the number of clusters varies.

The clustering methods used by PROC MODECLUS use spherical clustering neighborhoods of fixed or variable radius that are similar to the spherical kernels used for density estimation. For fixed-radius neighborhoods, specify the radius as a Euclidean distance with either the CR= or R= option. For variable-radius neighborhoods, specify the number of neighbors desired within the sphere with either the CK= or K= option; the radius is then the smallest radius that contains at least the specified number of observations including the observation for which the neighborhood is being determined. However, in the following descriptions of clustering methods, an observation is not considered to be one of its own neighbors. If you specify both the CR= or R= option and the CK= or K= option, the radius used is the maximum of the two indicated radii; this is useful for dealing with outliers. In this section, the symbols N_i, N_i^-, n_i, and n_i^- refer to clustering neighborhoods, not density estimation neighborhoods.

METHOD=0

Begin with each observation in a separate cluster. For each observation and each of its neighbors, join the cluster to which the observation belongs with the cluster to which the neighbor belongs. This method does not use density estimates. With a fixed clustering radius, the clusters are those obtained by cutting the single linkage tree at the specified radius (see Chapter 40, "The CLUSTER Procedure").

METHOD=1

Begin with each observation in a separate cluster. For each observation, find the nearest neighbor with a greater estimated density. If such a neighbor exists, join the cluster to which the observation belongs with the cluster to which the specified neighbor belongs.

Next, consider each observation with density estimates equal to that of one or more neighbors but not less than the estimate at any neighbor. Join the cluster containing the observation with (1) each cluster containing a neighbor of the observation such that the maximum density estimate in the cluster equals the density estimate at the observation and (2) the cluster containing the nearest neighbor of the observation such that the maximum density estimate in the cluster exceeds the density estimate at the observation.

This method is similar to the classification or assignment stage of algorithms described by Gitman (1973) and Huizinga (1978).

METHOD=2

Begin with each observation in a separate cluster. For each observation, find the neighbor with the greatest estimated density exceeding the estimated density of the observation. If such a neighbor exists, join the cluster to which the observation belongs with the cluster to which the specified neighbor belongs.

Observations with density estimates equal to that of one or more neighbors but not less than the estimate at any neighbor are treated the same way as they are in METHOD=1.

This method is similar to the first stage of an algorithm proposed by Mizoguchi and Shimura (1980).

METHOD=3

Begin with each observation in a separate cluster. For each observation, find the neighbor with greater estimated density such that the slope of the line connecting the point on the estimated density surface at the observation with the point on the estimated density surface at the neighbor is a maximum. That is, for observation x_i, find a neighbor x_j such that $(\hat{f}_j - \hat{f}_i)/\mathrm{d}(x_j, x_i)$ is a maximum. If this slope is positive, join the cluster to which observation x_i belongs with the cluster to which the specified neighbor x_j belongs. This method was invented by Koontz, Narendra, and Fukunaga (1976).

Observations with density estimates equal to that of one or more neighbors but not less than the estimate at any neighbor are treated the same way as they are in METHOD=1. The algorithm suggested for this situation by Koontz, Narendra, and Fukunaga (1976) might fail for flat areas in the estimated density that contain four or more observations.

METHOD=4

This method is equivalent to the first stage of two-stage density linkage (see Chapter 40, "The CLUSTER Procedure") without the use of the MODE=option.

METHOD=5

This method is equivalent to the first stage of two-stage density linkage (see Chapter 40, "The CLUSTER Procedure") with the use of the MODE=option.

METHOD=6

Begin with all observations unassigned.

Step 1: Form a list of seeds, each seed being a single observation such that the estimated density of the observation is not less than the estimated density of any of its neighbors. If you specify the MAXCLUSTERS=*n* option, retain only the *n* seeds with the greatest estimated densities.

Step 2: Consider each seed in decreasing order of estimated density, as follows:

1. If the current seed has already been assigned, proceed to the next seed. Otherwise, form a new cluster consisting of the current seed.
2. Add to the cluster any unassigned seed that is a neighbor of a member of the cluster or that shares a neighbor with a member of the cluster; repeat until no unassigned seed satisfies these conditions.
3. Add to the cluster all neighbors of seeds that belong to the cluster.
4. Consider each unassigned observation. Compute the ratio of the sum of the $p-1$ powers of the estimated density of the neighbors that belong to the current cluster to the sum of the $p-1$ powers of the estimated density of all of its neighbors, where p is specified by the POWER= option and is 2 by default. Let x_i be the current observation, and let k be the index of the current cluster. Then this ratio is

$$r_{ik} = \frac{\sum_{j \in N_i \cap C_k} \hat{f}_j^{p-1}}{\sum_{j \in N_i} \hat{f}_j^{p-1}}$$

(The sum of the $p-1$ powers of the estimated density of the neighbors of an observation is an estimate of the integral of the pth power of the density over the neighborhood.) If r_{ik} exceeds the maximum of 0.5 and the value of the THRESHOLD= option, add the observation x_i to the current cluster k. Repeat until no more observations can be added to the current cluster.

Step 3: (This step is performed only if the value of the THRESHOLD= option is less than 0.5.) Form a list of unassigned observations in decreasing order of estimated density. Repeat the following actions until the list is empty:

1. Remove the first observation from the list, such as observation x_i.
2. For each cluster k, compute r_{ik}.
3. If the maximum over clusters of r_{ik} exceeds the value of the THRESHOLD= option, assign observation x_i to the corresponding cluster and insert all observations of which the current observation is a neighbor into the list, keeping the list in decreasing order of estimated density.

METHOD=6 is related to a method invented by Koontz and Fukunaga (1972a) and discussed by Koontz and Fukunaga (1972b).

Significance Tests

Significance tests require that a fixed-radius kernel be specified for density estimation via the DR= or R= option. You can also specify the DK= or K= option, but only the fixed radius is used for the significance tests.

The purpose of the significance tests is as follows: given a simple random sample of objects from a population, obtain an estimate of the number of clusters in the population such that the probability in repeated sampling that the estimate exceeds the true number of clusters is not much greater than α, $1\% \leq \alpha \leq 10\%$. In other words, a sequence of null hypotheses of the form

$$H_0^{(i)}: \text{The number of population clusters is } i \text{ or less}$$

where $i = 1, 2, \ldots, n$, is tested against the alternatives such as

$$H_a^{(i)}: \text{The number of population clusters exceeds } i$$

with a maximum experimentwise error rate of approximately α. The tests protect you from overestimating the number of population clusters. It is impossible to protect against underestimating the number of population clusters without introducing much stronger assumptions than are used here, since the number of population clusters could conceivably exceed the sample size.

The method for conducting significance tests is as follows:

1. Estimate densities by using fixed-radius uniform kernels.
2. Obtain preliminary clusters by a "valley-seeking" method. Other clustering methods could be used but would yield less power.
3. Compute an approximate p-value for each cluster by comparing the estimated maximum density in the cluster with the estimated maximum density on the cluster boundary.
4. Repeatedly join the least significant cluster with a neighboring cluster until all remaining clusters are significant.
5. Estimate the number of population clusters as the number of significant sample clusters.
6. The preceding steps can be repeated for any number of different radii, and the estimate of the number of population clusters can be taken to be the maximum number of significant sample clusters for any radius.

This method has the following useful features:

- No distributional assumptions are required.
- The choice of smoothing parameter is not critical since you can try any number of different values.

- The data can be coordinates or distances.
- Time and space requirements for the significance tests are no worse than those for obtaining the clusters.
- The power is high enough to be useful for practical purposes.

The method for computing the p-values is based on a series of plausible approximations. There are as yet no rigorous proofs that the method is infallible. Neither are there any asymptotic results. However, simulations for sample sizes ranging from 20 to 2000 indicate that the p-values are almost always conservative. The only case discovered so far in which the p-values are liberal is a uniform distribution in one dimension for which the simulated error rates exceed the nominal significance level only slightly for a limited range of sample sizes.

To make inferences regarding population clusters, it is first necessary to define what is meant by a cluster. For clustering methods that use nonparametric density estimation, a cluster is usually loosely defined as a region surrounding a local maximum of the probability density function or a maximal connected set of local maxima. This definition might not be satisfactory for very rough densities with many local maxima. It is not applicable at all to discrete distributions for which the density does not exist. As another example in which this definition is not intuitively reasonable, consider a uniform distribution in two dimensions with support in the shape of a figure eight (including the interior). This density might be considered to contain two clusters even though it does not have two distinct modes.

These difficulties can be avoided by defining clusters in terms of the local maxima of a smoothed probability density or mass function. For example, define the neighborhood distribution function (NDF) with radius r at a point x as the probability that a randomly selected point will lie within a radius r of x—that is, the probability integral over a hypersphere of radius r centered at x:

$$s(x) \quad = \quad P(\mathrm{d}(x, X) <= r)$$

where X is the random variable being sampled, r is a user-specified radius, and $\mathrm{d}(x,y)$ is the distance between points x and y.

The NDF exists for all probability distributions. You can select the radius according to the degree of resolution required. The minimum-variance unbiased estimate of the NDF at a point x is proportional to the uniform-kernel density estimate with corresponding support.

You can define a *modal region* as a maximal connected set of local maxima of the NDF. A cluster is a connected set containing exactly one modal region. This definition seems to give intuitively reasonable results in most cases. An exception is a uniform density on the perimeter of a square. The NDF has four local maxima. There are eight local maxima along the perimeter, but running PROC MODECLUS with the R= option would yield four clusters since the two local maxima at each corner are separated by a distance equal to the radius. While this density does indeed have four distinctive features (the corners), it is not obvious that each corner should be considered a cluster.

The number of population clusters depends on the radius of the NDF. The significance tests in PROC MODECLUS protect against overestimating the number of clusters at any specified radius. It is often useful to look at the clustering results across a range of radii. A plot of the number of sample clusters as a function of the radius is a useful descriptive display, especially for high-dimensional data (Wong and Schaack 1982).

If a population has two clusters, it must have two modal regions. If there are two modal regions, there must be a "valley" between them. It seems intuitively desirable that the boundary between the two clusters should follow the bottom of this valley. All the clustering methods in PROC MODECLUS are designed to locate the estimated cluster boundaries in this way, although methods 1 and 6 seem to be much more successful at this than the others. Regardless of the precise location of the cluster boundary, it is clear that the maximum of the NDF along the boundary between two clusters must be strictly less than the value of the NDF in either modal region; otherwise, there would be only a single modal region; according to Hartigan and Hartigan (1985), there must be a "dip" between the two modes. PROC MODECLUS assesses the significance of a sample cluster by comparing the NDF in the modal region with the maximum of the NDF along the cluster boundary. If the NDF has second-order derivatives in the region of interest and if the boundary between the two clusters is indeed at the bottom of the valley, then the maximum value of the NDF along the boundary occurs at a saddle point. Hence, this test is called a *saddle test*. This term is intended to describe any test for clusters that compares modal densities with saddle densities, not just the test currently implemented in the MODECLUS procedure.

The obvious estimate of the maximum NDF in a sample cluster is the maximum estimated NDF at an observation in the cluster. Let $m(k)$ be the index of the observation for which the maximum is attained in cluster k.

Estimating the maximum NDF on the cluster boundary is more complicated. One approach is to take the maximum NDF estimate at an observation in the cluster that has a neighbor belonging to another cluster. This method yields excessively large estimates when the neighborhood is large. Another approach is to try to choose an object closer to the boundary by taking the observation with the maximum sum of estimated densities of neighbors belonging to a different cluster. After some experimentation, it is found that a combination of these two methods works well. Let B_k be the set of indices of observations in cluster k that have neighbors belonging to a different cluster, and compute

$$\max_{i \in B_k} \left(0.2 \hat{f}_i n_i + \sum_{j \in N_i - C_k} \hat{f}_j \right)$$

Let $s(k)$ be the index of the observation for which the maximum is attained.

Using the notation $\#(S)$ for the cardinality of set S, let

$$\begin{aligned}
n_{ij}^{-} &= \#(N_i^{-} \cap N_j^{-}) \\
c_m(k) &= n_{m(k)}^{-} - n_{m(k)s(k)}^{-} \\
c_s(k) &= n_{s(k)}^{-} - n_{m(k)s(k)}^{-} \text{ if } B_k \neq \emptyset, \\
&= 0 \text{ otherwise} \\
q_k &= 1/2 \text{ if } B_k \neq \emptyset, \\
&= 2/3 \text{ otherwise} \\
z_k &= \frac{c_m(k) - q_k(c_m(k) + c_s(k)) - 1/2}{\sqrt{q_k(1-q_k)(c_m(k) + c_s(k))}} \\
u &= \left\lceil (0.2 + 0.05\sqrt{n}) \sum_{i:n_i>1} \frac{1}{n_i + 1} \right\rceil
\end{aligned}$$

Let $R(u)$ be a random variable distributed as the range of a random sample of u observations from a standard normal distribution. Then the approximate p-value p_k for cluster k is

$$p_k = \Pr(z_k > R(u)/\sqrt{2})$$

If points $m(k)$ and $s(k)$ are fixed a priori, z_k would be the usual approximately normal test statistic for comparing two binomial random variables. In fact, $m(k)$ and $s(k)$ are selected in such a way that $c_m(k)$ tends to be large and $c_s(k)$ tends to be small. For this reason, and because there might be a large number of clusters, each with its own z_k to be tested, each z_k is referred to the distribution of $R(u)$ instead of a standard normal distribution. If the tests are conducted for only one radius and if u is chosen equal to n, then the p-values are very conservative because (1) you are not making all possible pairwise comparisons of observations in the sample and (2) n_i^- and n_j^- are positively correlated if the neighborhoods overlap. In the formula for u, the summation overcorrects somewhat for the conservativeness due to correlated n_i^-'s. The factor $0.2 + 0.05\sqrt{n}$ is empirically estimated from simulation results to adjust for the use of more than one radius.

If the JOIN option is specified, the least significant cluster (the cluster with the smallest z_k) is either dissolved or joined with a neighboring cluster. If no members of the cluster have neighbors belonging to a different cluster, all members of the cluster are unassigned. Otherwise, the cluster is joined to the neighboring cluster such that the sum of density estimates of neighbors of the estimated saddle point belonging to it is a maximum. Joining clusters increases the power of the saddle test. For example, consider a population with two well-separated clusters. Suppose that, for a certain radius, each population cluster is divided into two sample clusters. None of the four sample clusters is likely to be significant, but after the two sample clusters corresponding to each population cluster are joined, the remaining two clusters can be highly significant.

The saddle test implemented in PROC MODECLUS has been evaluated by simulation from known distributions. Some results are given in the following three tables. In Table 85.3, samples of 20 to 2000 observations are generated from a one-dimensional uniform distribution. For sample sizes of 1000 or less, 2000 samples are generated and analyzed by PROC MODECLUS. For a sample size of 2000, only 1000 samples are generated. The analysis is done with at least 20 different values of the R= option spread across the range of radii most likely to yield significant results. The six central columns of the table give the observed error rates at the nominal error rates (α) at the head of each column. The standard errors of the observed error rates are given at the bottom of the table. The observed error rates are conservative for $\alpha \leq 5\%$, but they increase with α and become slightly liberal for sample sizes in the middle of the range tested.

Table 85.3 Observed Error Rates (%) for Uniform Distribution

Sample	Nominal Type 1 Error Rate						Number of
Size	1	2	5	10	15	20	Simulations
20	0.00	0.00	0.00	0.60	11.65	27.05	2000
50	0.35	0.70	4.50	10.95	20.55	29.80	2000
100	0.35	0.85	3.90	11.05	18.95	28.05	2000
200	0.30	1.35	4.00	10.50	18.60	27.05	2000
500	0.45	1.05	4.35	9.80	16.55	23.55	2000
1000	0.70	1.30	4.65	9.55	15.45	19.95	2000
2000	0.40	1.10	3.00	7.40	11.50	16.70	1000
Standard	0.22	0.31	0.49	0.67	0.80	0.89	2000
Error	0.31	0.44	0.69	0.95	1.13	1.26	1000

All unimodal distributions other than the uniform that have been tested, including normal, Cauchy, and exponential distributions and uniform mixtures, have produced much more conservative results. Table 85.4 displays results from a unimodal mixture of two normal distributions with equal variances and equal sampling probabilities and with means separated by two standard deviations. Any greater separation would produce a bimodal distribution. The observed error rates are quite conservative.

Table 85.4 Observed Error Rates (%) for Normal Mixture with 2σ Separation

Sample	Nominal Type 1 Error Rate						Number of
Size	1	2	5	10	15	20	Simulations
100	0.0	0.0	0.0	1.0	2.0	4.0	200
200	0.0	0.0	0.0	2.0	3.0	3.0	200
500	0.0	0.0	0.5	0.5	0.5	0.5	200

All distributions in two or more dimensions that have been tested yield extremely conservative results. For example, a uniform distribution on a circle yields observed error rates that are never more than one-tenth of the nominal error rates for sample sizes up to 1000. This conservatism is due to the fact that, as the dimensionality increases, more and more of the probability lies in the tails of the distribution (Silverman 1986, p. 92), and the saddle test used by PROC MODECLUS is more conservative for distributions with pronounced tails. This applies even to a uniform distribution on a hypersphere because, although the density does not have tails, the NDF does.

Since the formulas for the significance tests do not involve the dimensionality, no problems are created when the data are linearly dependent. Simulations of data in nonlinear subspaces (the circumference of a circle or surface of a sphere) have also yielded conservative results.

Table 85.5 displays results in terms of power for identifying two clusters in samples from a bimodal mixture of two normal distributions with equal variances and equal sampling probabilities separated by four standard deviations. In this simulation, PROC MODECLUS never indicated more than two significant clusters.

Table 85.5 Power (%) for Normal Mixture with 4σ Separation

Sample	Nominal Type 1 Error Rate						Number of
Size	1	2	5	10	15	20	Simulations
20	0.0	0.0	0.0	2.0	37.5	68.5	200
35	0.0	13.5	38.5	48.5	64.0	75.5	200
50	17.5	26.0	51.5	67.0	78.5	84.0	200
75	25.5	36.0	58.5	77.5	85.5	89.5	200
100	40.0	54.5	72.5	84.5	91.5	92.5	200
150	70.5	80.0	92.0	97.0	100.0	100.0	200
200	89.0	96.0	99.5	100.0	100.0	100.0	200

The saddle test is not as efficient as excess-mass tests for multimodality (Müller and Sawitzki 1991; Polonik 1993). However, there is not yet a general approximation for the distribution of excess-mass statistics to circumvent the need for simulations to do significance tests. See Minnotte (1992) for a review of tests for multimodality.

Computational Resources

The MODECLUS procedure stores coordinate data in memory if there is enough space. For distance data, only one observation at a time is in memory.

PROC MODECLUS constructs lists of the neighbors of each observation. The total space required is $12 \sum n_i$ bytes, where n_i is based on the largest neighborhood required by any analysis. The lists are stored in a SAS utility data set unless you specify the CORE option. You might get an error message from the SAS System or from the operating system if there is not enough disk space for the utility data set. Clustering method 6 requires a second list that is always stored in memory.

For coordinate data, the time required to construct the neighbor lists is roughly proportional to $v(\log n)(\sum n_i) \log(\sum n_i / n)$. For distance data, the time is roughly proportional to $n^2 \log(\sum n_i / n)$.

The time required for density estimation is proportional to $\sum n_i$ and is usually small compared to the time required for constructing the neighbor lists.

Clustering methods 0 through 3 are quite efficient, requiring time proportional to $\sum n_i$. Methods 4 and 5 are slower, requiring time roughly proportional to $(\sum n_i) \log(\sum n_i)$. Method 6 can also be slow, but the time requirements depend very much on the data and the particular options specified. Methods 4, 5, and 6 also require more memory than the other methods.

The time required for significance tests is roughly proportional to $g \sum n_i$, where g is the number of clusters.

PROC MODECLUS can process data sets of several thousand observations if you specify reasonable smoothing parameters. Very small smoothing values produce many clusters, whereas very large values produce many neighbors; either case can require excessive time or space.

Missing Values

If the data are coordinates, observations with missing values are excluded from the analysis.

If the data are distances, missing values are treated as infinite. The neighbors of each observation are determined solely by the distances in that observation. The distances are not required to be symmetric, and there is no check for symmetry; the neighbors of each observation are determined only from the distances in that observation. This treatment of missing values is quite different from that of the CLUSTER procedure, which ignores the upper triangle of the distance matrix.

Output Data Sets

The OUT= data set contains one complete copy of the input data set for each cluster solution. There are additional variables identifying each solution and giving information about individual observations. Solutions with only one remaining cluster when JOIN=*p* is specified are omitted from the OUT= data set (see the description of the JOIN= option). The OUT= data set can be extremely large, so it is advisable to specify the DROP= data set option to exclude unnecessary variables.

The OUTCLUS= or OUTC= data set contains one observation for each cluster in each cluster solution. The variables identify the solution and provide statistics describing the cluster.

The OUTSUM= or OUTS= data set contains one observation for each cluster solution. The variables identify the solution and provide information about the solution as a whole.

The following variables can appear in all of the output data sets:

- _K_, which is the value of the K= option for the current solution. This variable appears only if you specify the K= option.
- _DK_, which is the value of the DK= option for the current solution. This variable appears only if you specify the DK= option.
- _CK_, which is the value of the CK= option for the current solution. This variable appears only if you specify the CK= option.
- _R_, which is the value of the R= option for the current solution. This variable appears only if you specify the R= option.
- _DR_, which is the value of the DR= option for the current solution. This variable appears only if you specify the DR= option.
- _CR_, which is the value of the CR= option for the current solution. This variable appears only if you specify the CR= option.
- _CASCAD_, which is the number of times the density estimates have been cascaded for the current solution. This variable appears only if you specify the CASCADE= option.
- _METHOD_, which is the value of the METHOD= option for the current solution. This variable appears only if you specify the METHOD= option.
- _NJOIN_, which is the number of clusters that are joined or dissolved in the current solution. This variable appears only if you specify the JOIN option.
- _LOCAL_, which is the local dimensionality estimate of the observation. This variable appears only if you specify the LOCAL option.

The OUT= data set contains the following variables:

- the variables from the input data set

- _OBS_, which is the observation number from the input data set. This variable appears only if you omit the ID statement.
- DENSITY, which is the estimated density at the observation. This variable can be renamed by the DENSITY= option.
- CLUSTER, which is the number of the cluster to which the observation is assigned. This variable can be renamed by the CLUSTER= option.

The OUTCLUS= data set contains the following variables:

- the BY variables, if any
- _NCLUS_, which is the number of clusters in the solution
- CLUSTER, which is the number of the current cluster
- _FREQ_, which is the number of observations in the cluster
- _MODE_, which is the maximum estimated density in the cluster
- _BFREQ_, which is the number of observations in the cluster with neighbors belonging to a different cluster
- _SADDLE_, which is the estimated saddle density for the cluster
- _MC_, which is the number of observations within the fixed-radius density-estimation neighborhood of the modal observation. This variable appears only if you specify the TEST or JOIN option.
- _SC_, which is the number of observations within the fixed-radius density-estimation neighborhood of the saddle observation. This variable appears only if you specify the TEST or JOIN option.
- _OC_, which is the number of observations within the overlap of the two previous neighborhoods. This variable appears only if you specify the TEST or JOIN option.
- _Z_, which is the approximate z statistic for the cluster. This variable appears only if you specify the TEST or JOIN option.
- _P_, which is the approximate p-value for the cluster. This variable appears only if you specify the TEST or JOIN option.

The OUTSUM= data set contains the following variables:

- the BY variables, if any
- _NCLUS_, which is the number of clusters in the solution
- _UNCL_, which is the number of unclassified observations
- _CROSS_, which is the likelihood cross validation criterion if you specify the CROSS or CROSSLIST option

Displayed Output

If you specify the SIMPLE option and the data are coordinates, PROC MODECLUS displays the following simple descriptive statistics for each variable:

- the mean
- the standard deviation
- the skewness
- the kurtosis
- a coefficient of bimodality (see Chapter 40, "The CLUSTER Procedure")

If you specify the NEIGHBOR option, PROC MODECLUS displays a list of neighbors for each observation. The table contains the following items:

- the observation number or ID value of the observation
- the observation number or ID value of each of its neighbors
- the distance to each neighbor

If you specify the CROSSLIST option, PROC MODECLUS produces a table of information regarding cross validation of the density estimates. Each table has a row for each observation. For each observation, the following are displayed:

- the observation number or ID value of the observation
- the radius of the neighborhood
- the number of neighbors
- the estimated log density
- the estimated cross validated log density

If you specify the LOCAL option, PROC MODECLUS produces a table of information regarding estimates of local dimensionality. Each table has a row for each observation. For each observation, the following are displayed:

- the observation number or ID value of the observation
- the radius of the neighborhood
- the estimated local dimensionality

If you specify the LIST option, PROC MODECLUS produces a table listing the observations within each cluster. The table includes the following items:

- the cluster number
- the observation number or ID value of the observation
- the estimated density
- the sum of the density estimates of observations within the neighborhood that belong to the same cluster
- the sum of the density estimates of observations within the neighborhood that belong to a different cluster
- the sum of the density estimates of all the observations within the neighborhood
- the ratio of the sum of the density estimates for the same cluster to the sum of all the density estimates in the neighborhood

If you specify the LIST option and there are unassigned objects, PROC MODECLUS produces a table listing those observations. The table includes the following items:

- the observation number or ID value of the observation
- the estimated density
- the ratio of the sum of the density estimates for the same cluster to the sum of the density estimates in the neighborhood for all other clusters

If you specify the BOUNDARY option, PROC MODECLUS produces a table listing the observations in each cluster that have a neighbor belonging to a different cluster. The table includes the following items:

- the observation number or ID value of the observation
- the estimated density
- the cluster number
- the ratio of the sum of the density estimates for the same cluster to the sum of the density estimates in the neighborhood for all other clusters

If you do not specify the SHORT option, PROC MODECLUS produces a table of cluster statistics including the following items:

- the cluster number
- the cluster frequency (the number of observations in the cluster)
- the maximum estimated density within the cluster

- the number of observations in the cluster having a neighbor that belongs to a different cluster
- the estimated saddle density of the cluster

If you specify the TEST or JOIN option, the table of cluster statistics includes the following items pertaining to the saddle test:

- the number of observations within the fixed-radius density-estimation neighborhood of the modal observation
- the number of observations within the fixed-radius density-estimation neighborhood of the saddle observation
- the number of observations within the overlap of the two preceding neighborhoods
- the z statistic for comparing the preceding counts
- the approximate p-value

If you do not specify the NOSUMMARY option, PROC MODECLUS produces a table summarizing each cluster solution containing the following items:

- the smoothing parameters and cascade value
- the number of clusters
- the frequency of unclassified objects
- the likelihood cross validation criterion if you specify the CROSS or CROSSLIST option

If you specify the JOIN option, the summary table also includes the following items:

- the number of clusters joined
- the maximum p-value of any cluster in the solution

If you specify the TRACE option, PROC MODECLUS produces a table for each cluster solution that lists each observation along with its cluster membership as it is reassigned from the "Old" cluster to the "New" cluster. This reassignment is described in **Step 1** through **Step 3** of the section "METHOD=6" on page 6981. Each table has a row for each observation. For each observation, the following are displayed:

- the observation number or ID value of the observation
- the estimated density
- the "Old" cluster membership. 0 represents an unassigned observation and –1 represents a seed.
- the "New" cluster membership
- "Ratio," which is documented in the section "METHOD=6" on page 6981. The following character values can also be displayed:

"M" means the observation is a mode

"S" means the observation is a seed

"N" means the neighbor of a mode or seed, for which the ratio is not computed

ODS Table Names

PROC MODECLUS assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 85.6.

For more information about ODS, see Chapter 23, "Using the Output Delivery System."

All of the ODS tables in Table 85.6 are created by specifying the PROC MODECLUS statement.

Table 85.6 ODS Tables Produced by PROC MODECLUS

ODS Table Name	Description	Option
BoundaryFreq	Boundary objects information	BOUNDARY or ALL
ClusterList	Cluster listing, cluster ID, frequency, density, etc.	LIST or ALL
ClusterStats	Cluster statistics	Default
	Cluster statistics, significance test statistics	TEST, JOIN, or ALL
ClusterSummary	Cluster summary	Default
	Cluster summary, cross validation criterion	CROSS, CROSSLIS, or ALL
	Cluster summary, clusters joined information	JOIN or ALL
CrossList	Cross validated log density	CROSSLIST
ListLocal	Local dimensionality estimates	LOCAL
Neighbor	Nearest neighbor list	NEIGHBOR or ALL
SimpleStatistics	Simple statistics	SIMPLE or ALL
Trace	Trace of clustering algorithm (METHOD=6 only)	TRACE or ALL when METHOD=6
UnassignObjects	Information about unassigned objects	LIST or ALL

Examples: MODECLUS Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 85.1: Cluster Analysis of Samples from Univariate Distributions

(View the complete code for this example (modecex1.sas) in the example repository.)

This example uses pseudo-random samples from a uniform distribution, an exponential distribution, and a bimodal mixture of two normal distributions. Results are presented in Output 85.1.1 through Output 85.1.18 as plots displaying both the true density and the estimated density, as well as cluster membership.

The following statements produce Output 85.1.1 through Output 85.1.4:

```
title 'Modeclus Example with Univariate Distributions';
title2 'Uniform Distribution';

data uniform;
   drop n;
   true=1;
   do n=1 to 100;
      x=ranuni(123);
      output;
   end;
run;

proc modeclus data=uniform m=1 k=10 20 40 60 out=out short;
   var x;
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 2 by 1.);
   yaxis values=(0 to 3 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _K_;
run;

proc modeclus data=uniform m=1 r=.05 .10 .20 .30 out=out short;
   var x;
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 2 by 1.);
   yaxis values=(0 to 2 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _R_;
run;
```

Output 85.1.1 Cluster Analysis of Sample from a Uniform Distribution

Modeclus Example with Univariate Distributions
Uniform Distribution

The MODECLUS Procedure

Cluster Summary		
K	**Number of Clusters**	**Frequency of Unclassified Objects**
10	6	0
20	3	0
40	2	0
60	1	0

Output 85.1.2 True Density, Estimated Density, and Cluster Membership by Various _K_ Values

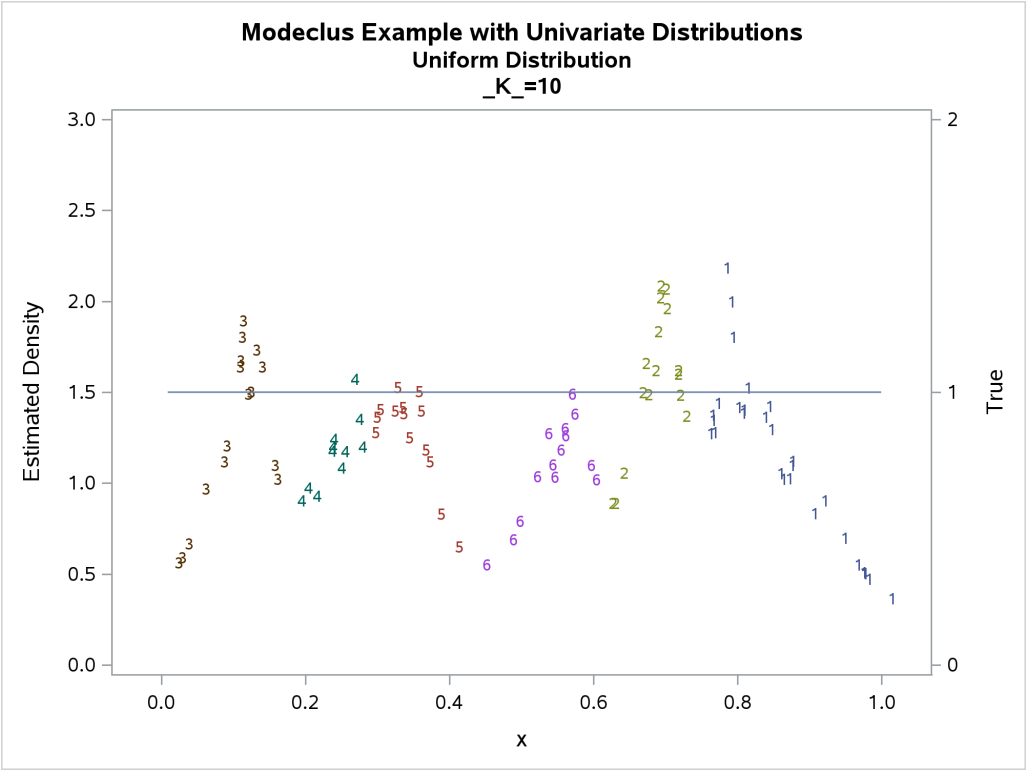

Output 85.1.2 *continued*

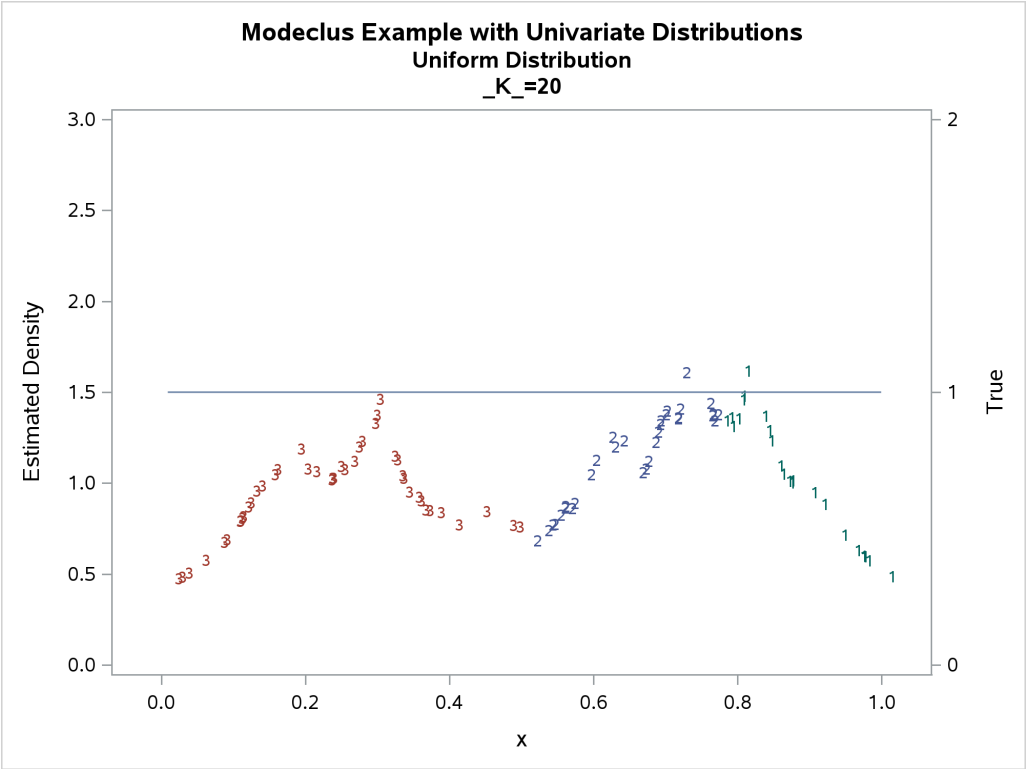

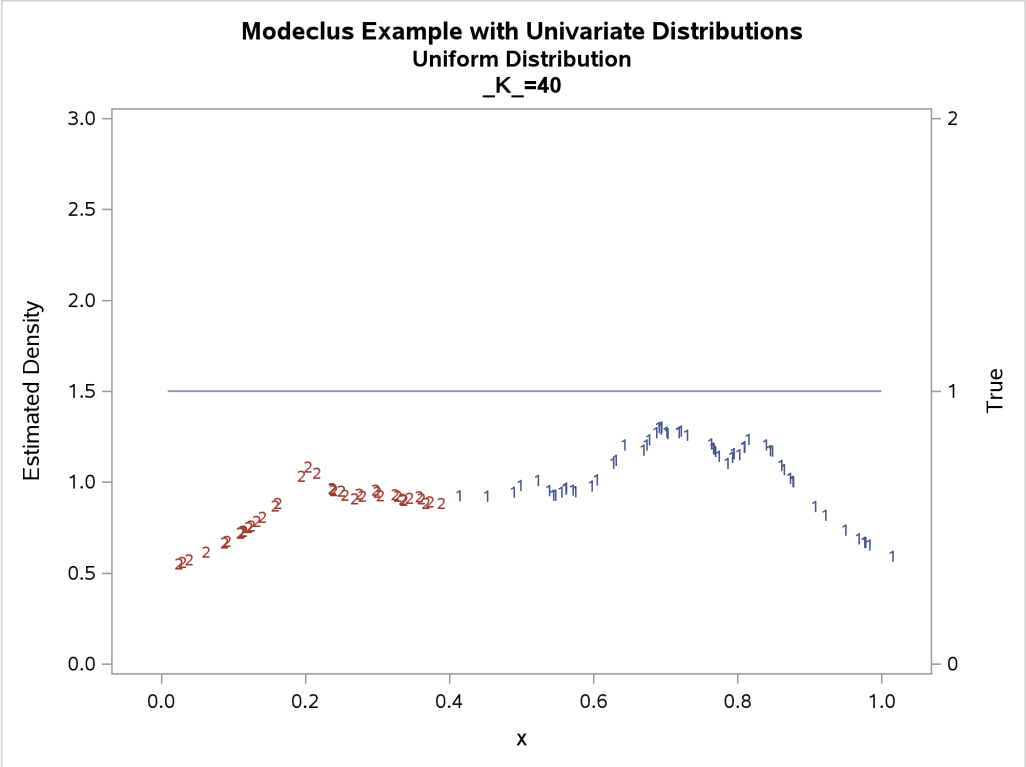

Output 85.1.2 *continued*

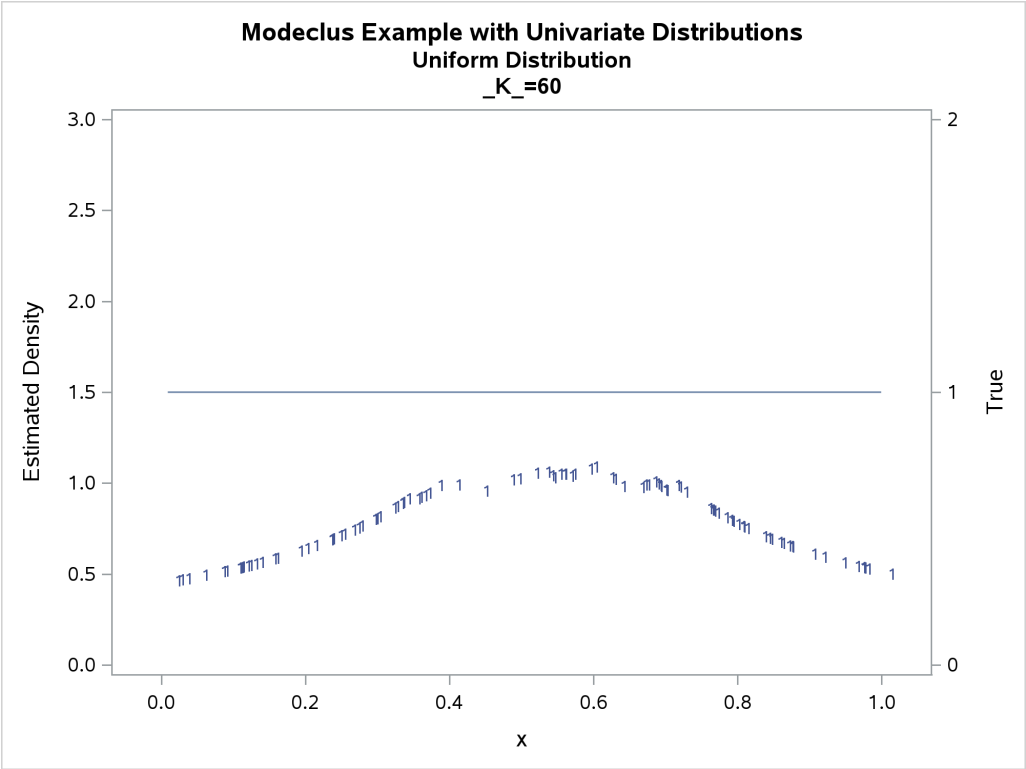

Output 85.1.3 Cluster Analysis of Sample from a Uniform Distribution

Modeclus Example with Univariate Distributions
Uniform Distribution

The MODECLUS Procedure

Cluster Summary		
R	**Number of Clusters**	**Frequency of Unclassified Objects**
0.05	4	0
0.1	2	0
0.2	2	0
0.3	1	0

Output 85.1.4 True Density, Estimated Density, and Cluster Membership by Various _R_ Values

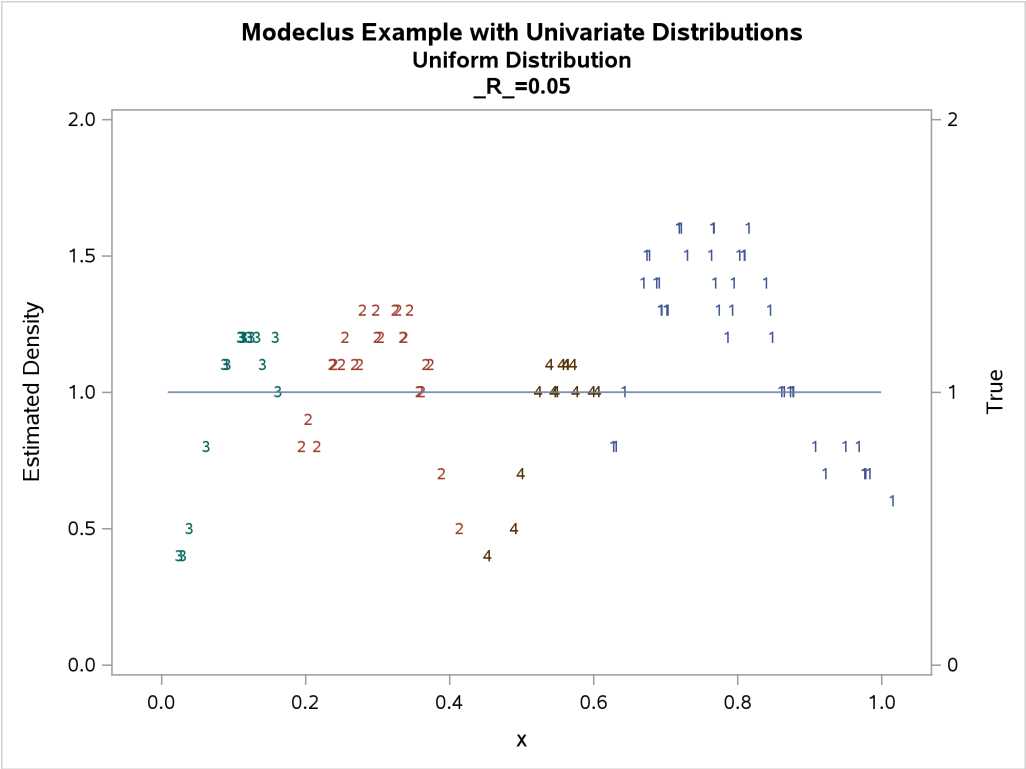

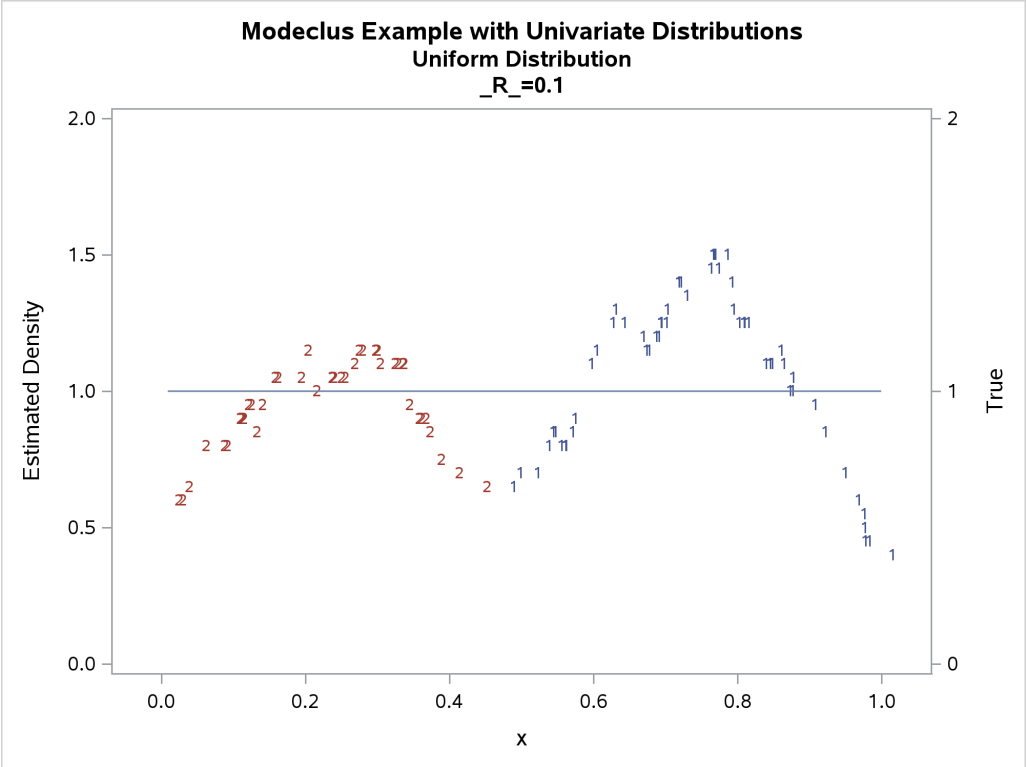

Output 85.1.4 *continued*

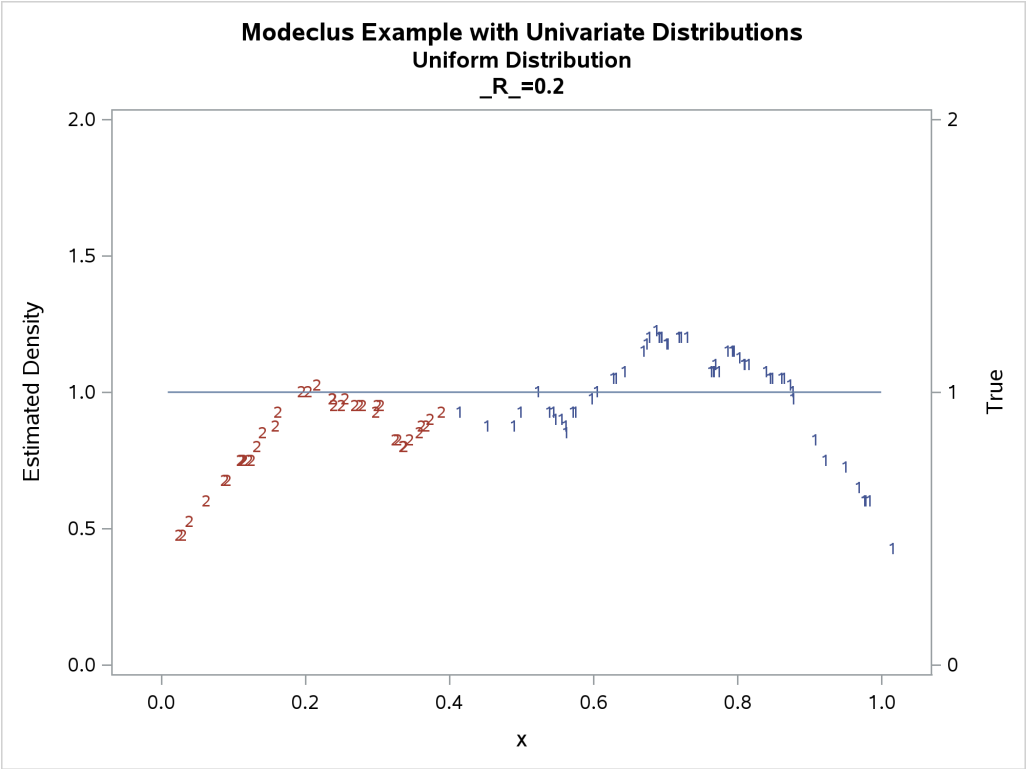

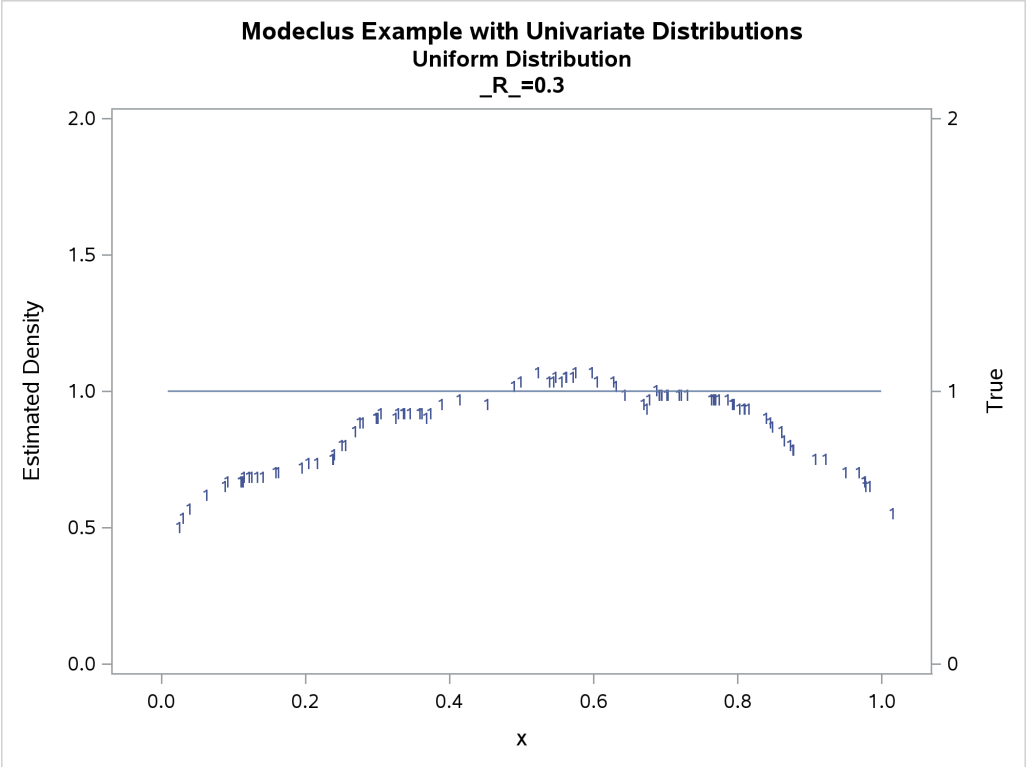

The following statements produce Output 85.1.5 through Output 85.1.12:

```
data expon;
   title2 'Exponential Distribution';
   drop n;
   do n=1 to 100;
      x=ranexp(123);
      true=exp(-x);
      output;
   end;
run;

proc modeclus data=expon m=1 k=10 20 40 out=out short;
   var x;
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 1 by .5);
   yaxis values=(0 to 2 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _K_;
run;

proc modeclus data=expon m=1 r=.20 .40 .80 out=out short;
   var x;
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 1 by .5);
   yaxis values=(0 to 1 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _R_;
run;

title3 'Different Density-Estimation and Clustering Windows';

proc modeclus data=expon m=1 r=.20 ck=10 20 40
              out=out short;
   var x;
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 1 by .5);
   yaxis values=(0 to 1 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _CK_;
run;

title3 'Cascaded Density Estimates Using Arithmetic Means';

proc modeclus data=expon m=1 r=.20 cascade=1 2 4 am out=out short;
   var x;
```

```
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 1 by .5);
   yaxis values=(0 to 1 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _R_ _CASCAD_;
run;
```

Output 85.1.5 Cluster Analysis of Sample from an Exponential Distribution

Modeclus Example with Univariate Distributions
Exponential Distribution

The MODECLUS Procedure

Cluster Summary		
K	Number of Clusters	Frequency of Unclassified Objects
10	5	0
20	3	0
40	1	0

Output 85.1.6 True Density, Estimated Density, and Cluster Membership by Various _K_ Values

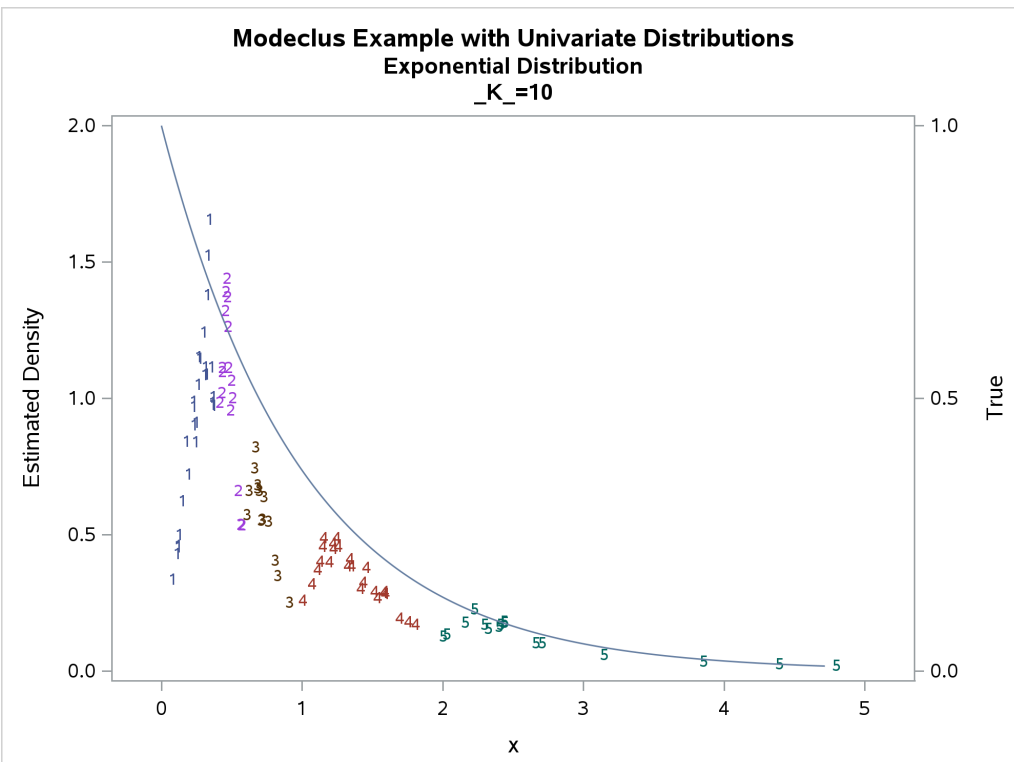

Output 85.1.6 *continued*

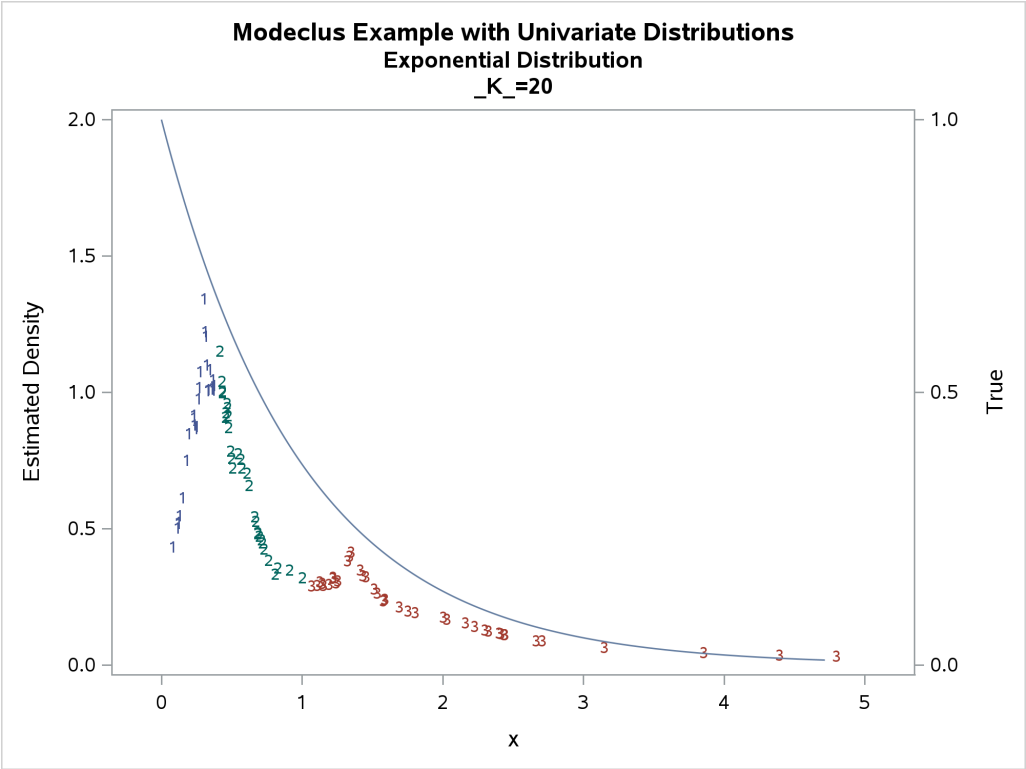

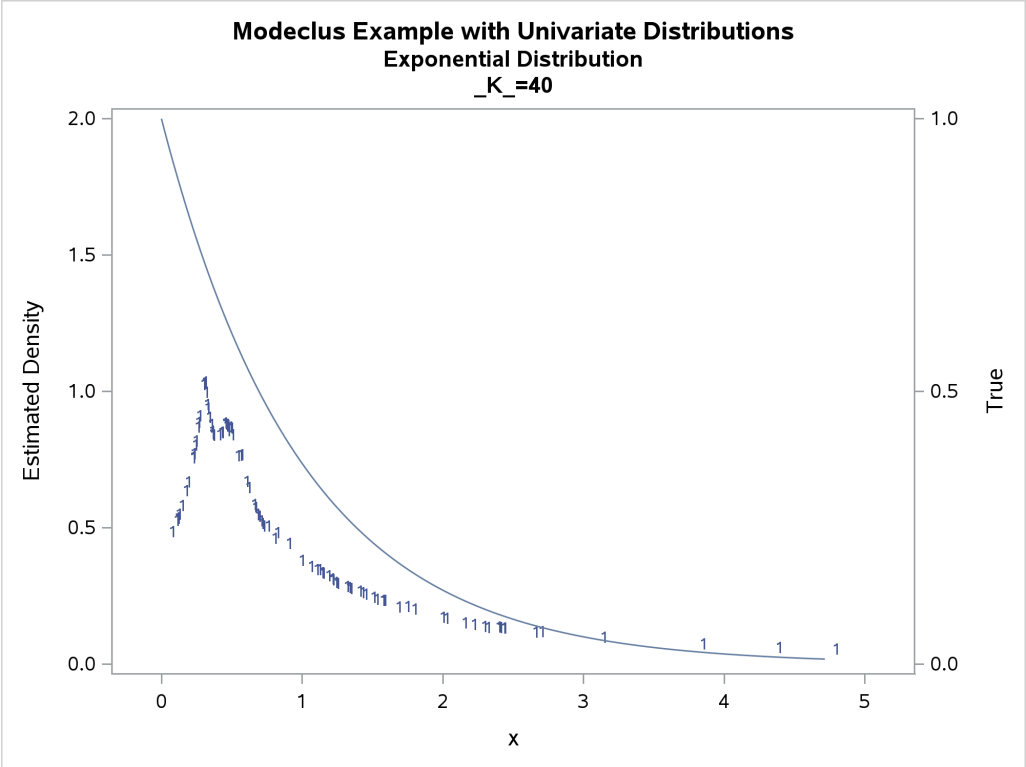

Output 85.1.7 Cluster Analysis of Sample from an Exponential Distribution

Modeclus Example with Univariate Distributions
Exponential Distribution

The MODECLUS Procedure

Cluster Summary		
R	**Number of Clusters**	**Frequency of Unclassified Objects**
0.2	8	0
0.4	6	0
0.8	1	0

Output 85.1.8 True Density, Estimated Density, and Cluster Membership by Various _R_ Values

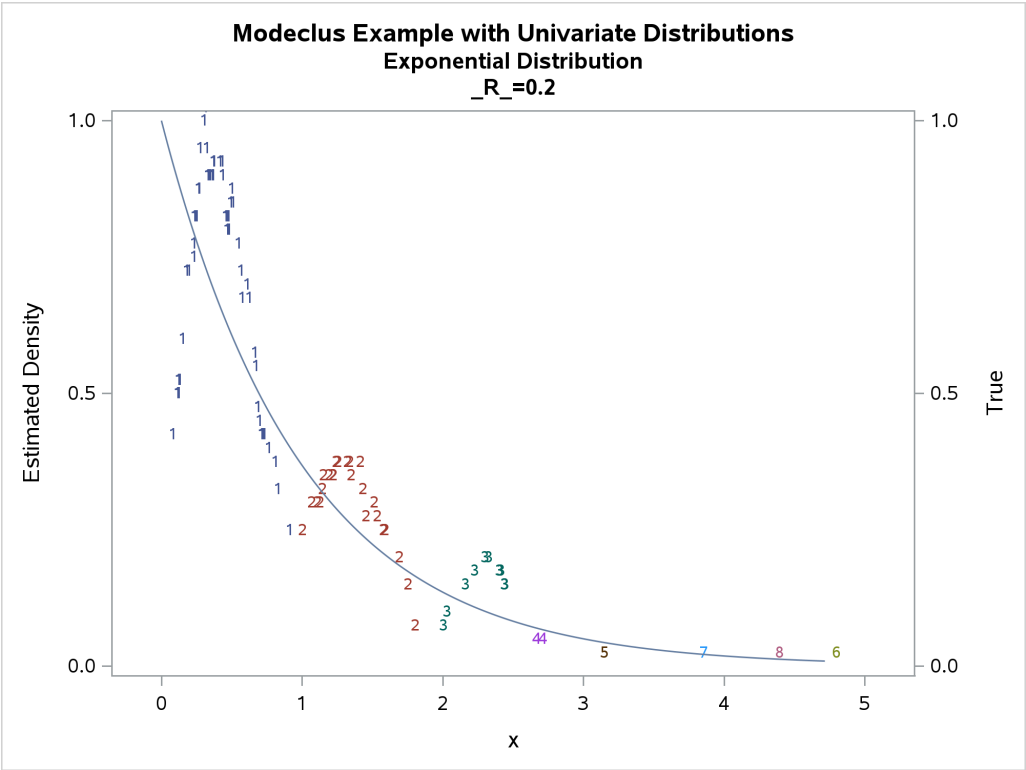

Output 85.1.8 *continued*

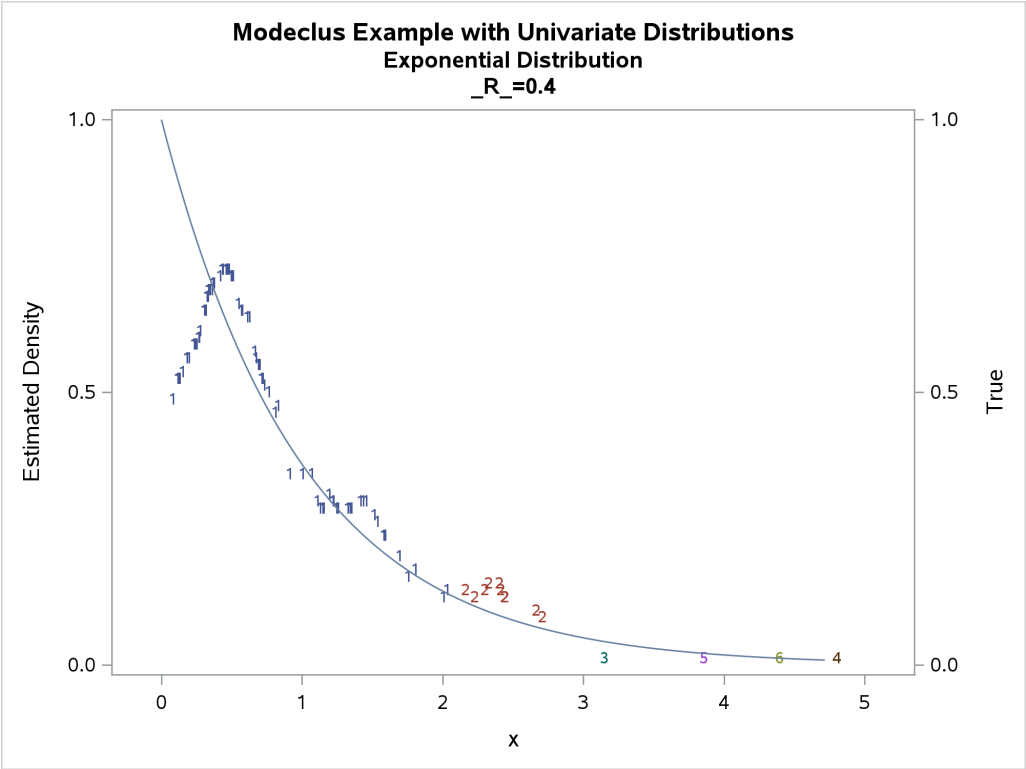

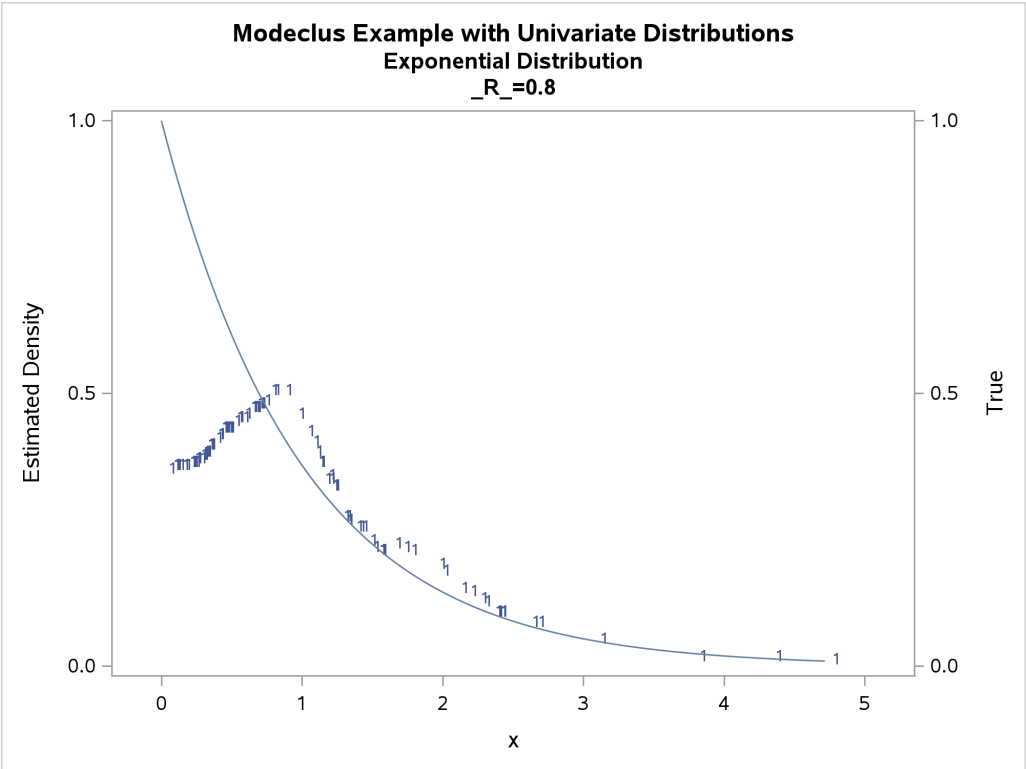

Output 85.1.9 Cluster Analysis of Sample from an Exponential Distribution

Modeclus Example with Univariate Distributions
Exponential Distribution
Different Density-Estimation and Clustering Windows

The MODECLUS Procedure

Cluster Summary

R	CK	Number of Clusters	Frequency of Unclassified Objects
0.2	10	3	0
0.2	20	2	0
0.2	40	1	0

Output 85.1.10 True Density, Estimated Density, and Cluster Membership by _R_=0.2 with Various _CK_ Values

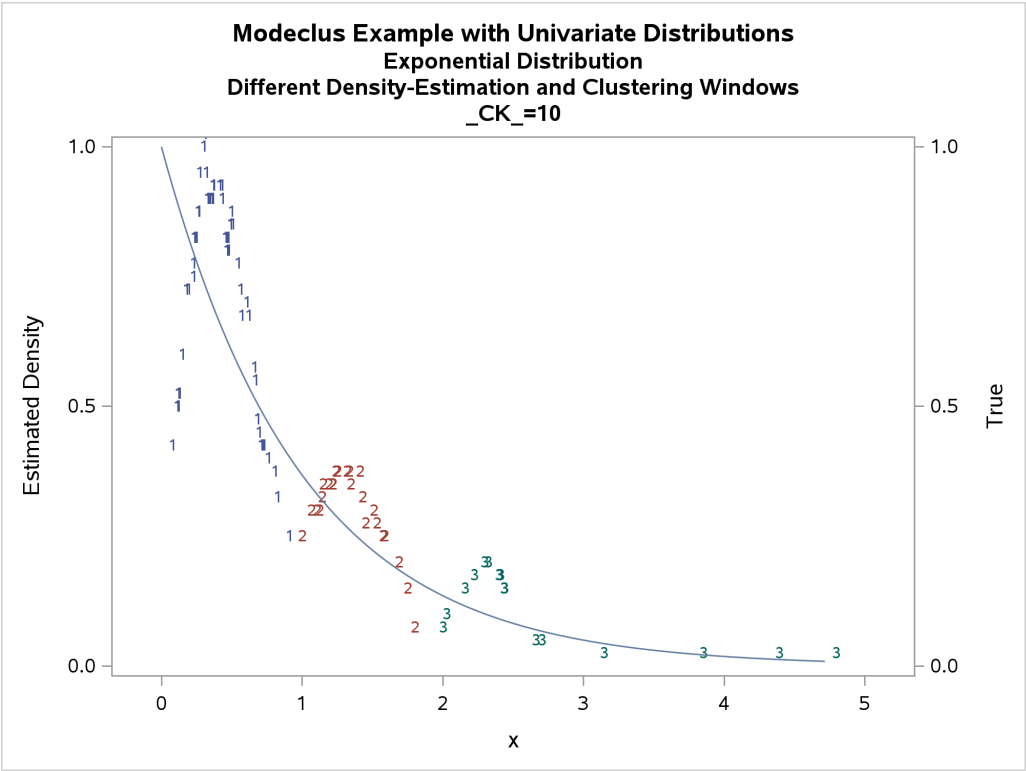

Output 85.1.10 *continued*

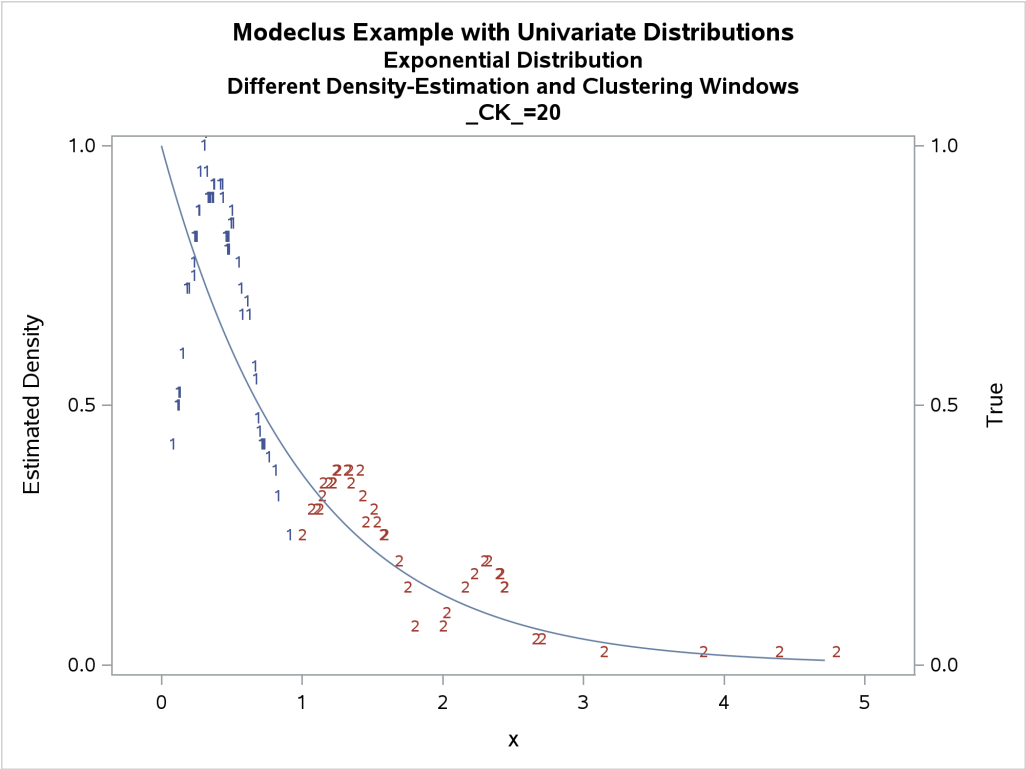

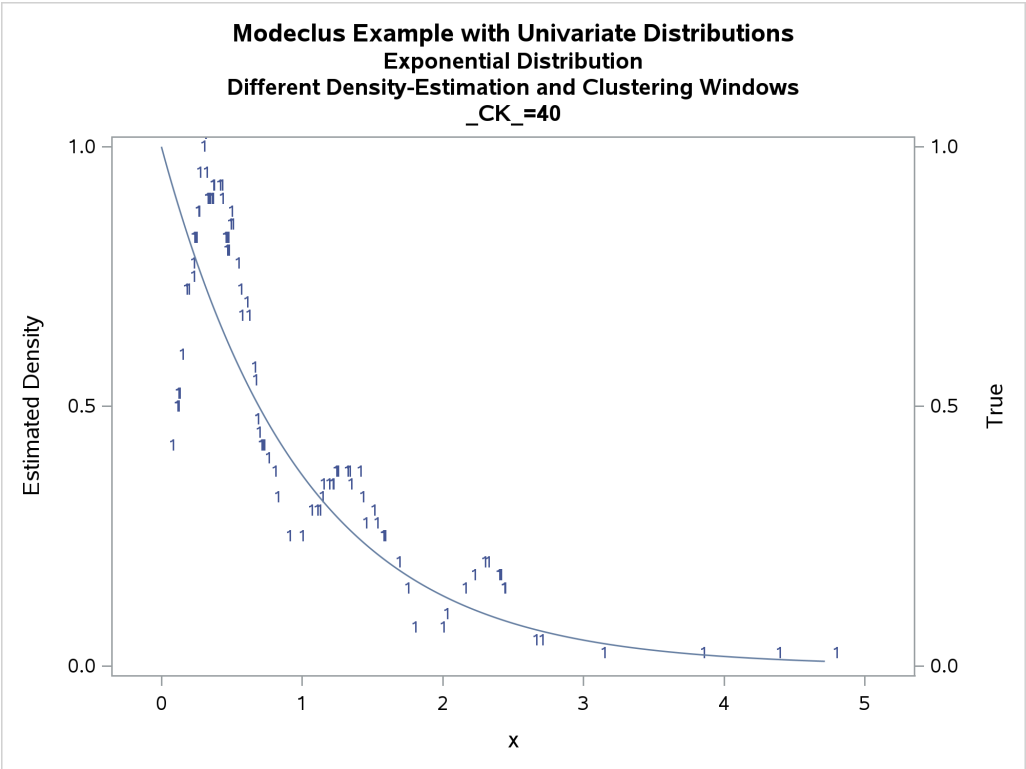

Output 85.1.11 Cluster Analysis of Sample from an Exponential Distribution

Modeclus Example with Univariate Distributions
Exponential Distribution
Cascaded Density Estimates Using Arithmetic Means

The MODECLUS Procedure

Cluster Summary

R	Cascade	Number of Clusters	Frequency of Unclassified Objects
0.2	1	8	0
0.2	2	8	0
0.2	4	7	0

Output 85.1.12 True Density, Estimated Density, and Cluster Membership by _R_=0.2 with Various _CASCAD_ Values

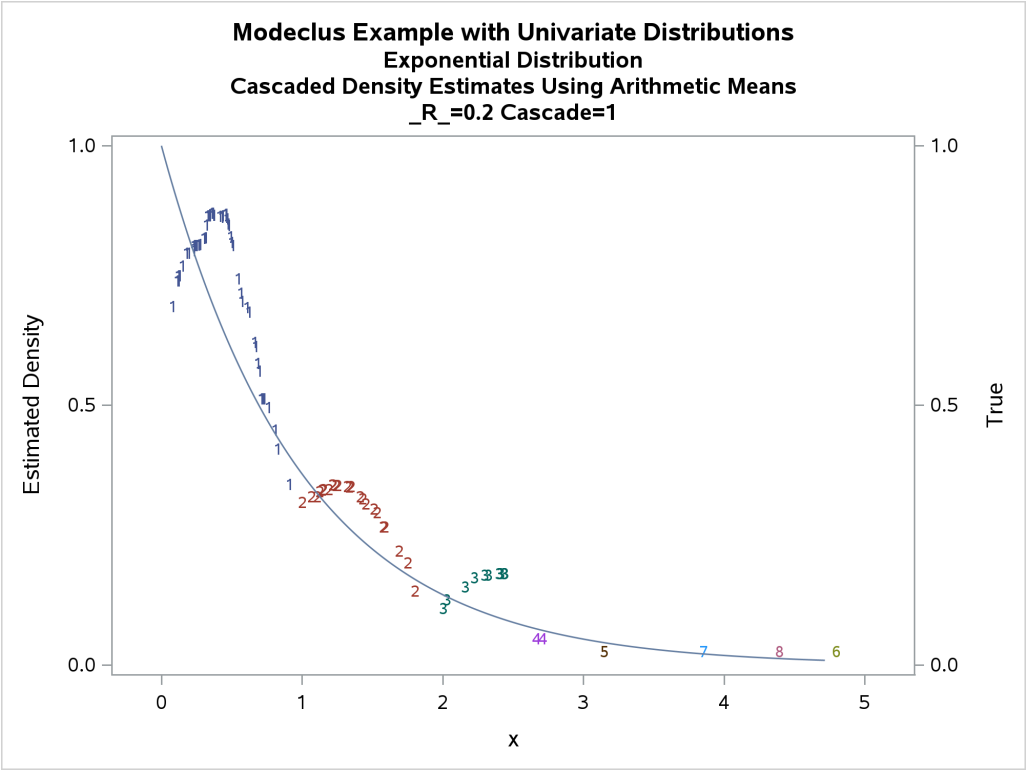

Output 85.1.12 *continued*

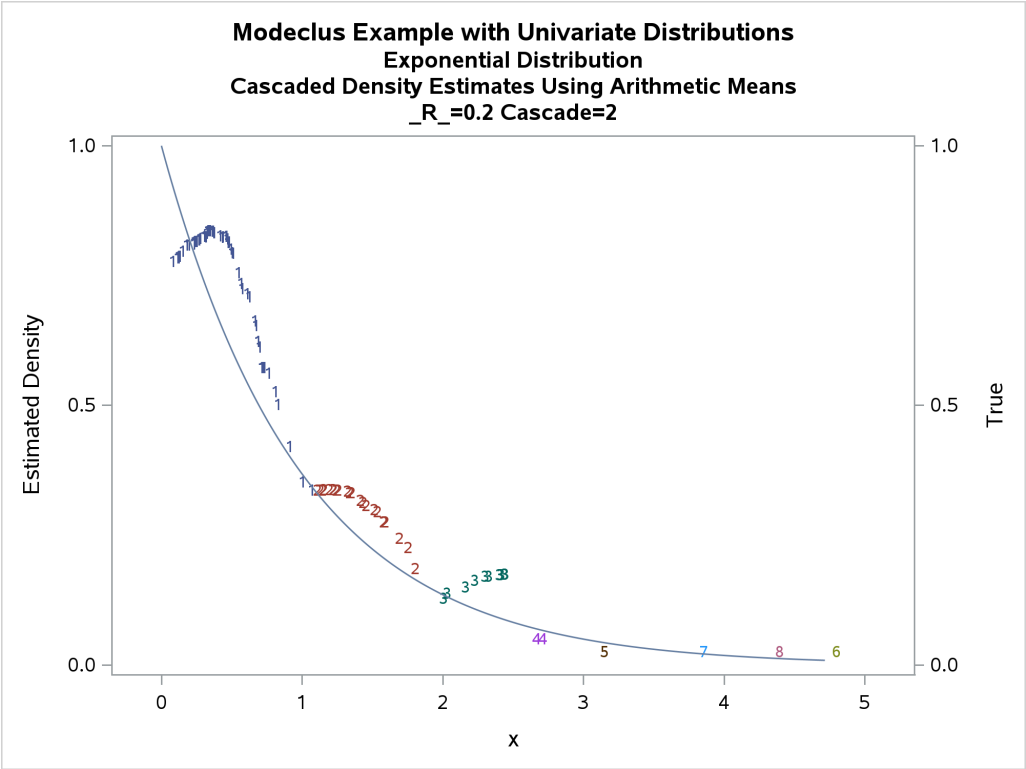

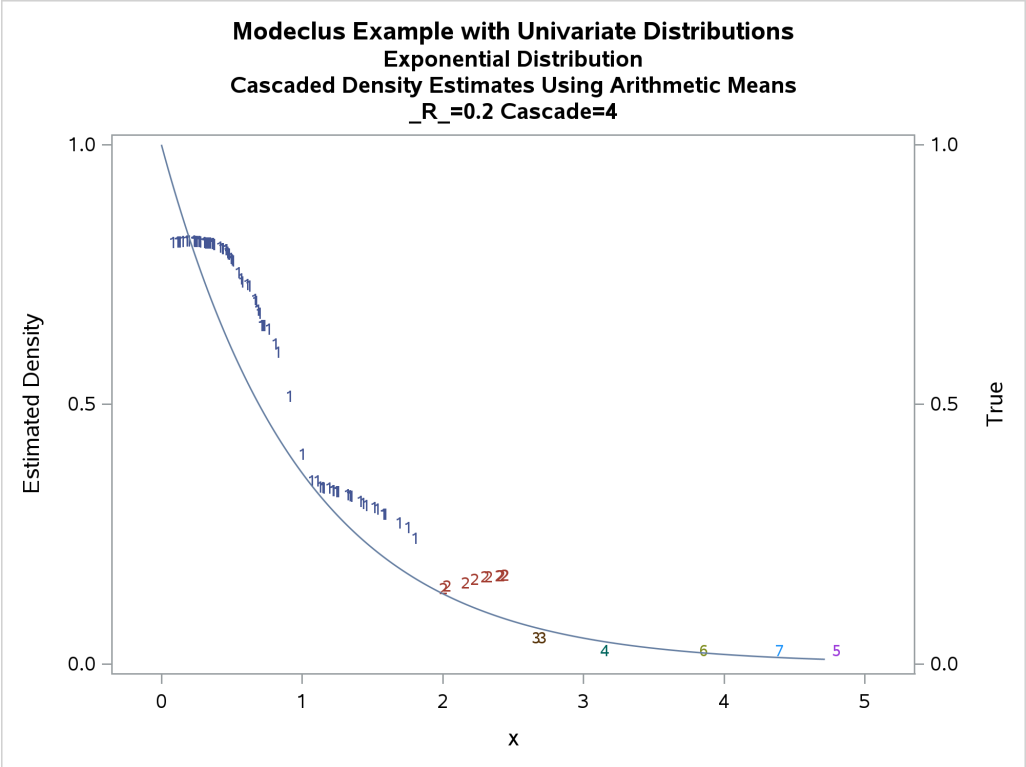

The following statements produce Output 85.1.13 through Output 85.1.18:

```
title2 'Normal Mixture Distribution';

data normix;
   drop n sigma;
   sigma=.125;
   do n=1 to 100;
      x=rannor(456)*sigma+mod(n,2)/2;
      true=exp(-.5*(x/sigma)**2)+exp(-.5*((x-.5)/sigma)**2);
      true=.5*true/(sigma*sqrt(2*3.1415926536));
      output;
   end;
run;

proc modeclus data=normix m=1 k=10 20 40 60 out=out short;
   var x;
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 1.6 by .1);
   yaxis values=(0 to 3 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _K_;
run;

proc modeclus data=normix m=1 r=.05 .10 .20 .30 out=out short;
   var x;
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 1.6 by .1);
   yaxis values=(0 to 3 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _R_;
run;

title3 'Cascaded Density Estimates Using Arithmetic Means';

proc modeclus data=normix m=1 r=.05 cascade=1 2 4 am out=out short;
   var x;
run;

proc sgplot data=out noautolegend;
   y2axis label='True' values=(0 to 1.6 by .1);
   yaxis values=(0 to 2 by 0.5);
   scatter y=density x=x / markerchar=cluster group=cluster;
   pbspline y=true x=x / y2axis nomarkers lineattrs=(thickness= 1);
   by _R_ _CASCAD_;
run;
```

Output 85.1.13 Cluster Analysis of Sample from a Bimodal Mixture of Two Normal Distributions

Modeclus Example with Univariate Distributions
Normal Mixture Distribution

The MODECLUS Procedure

Cluster Summary		
K	**Number of Clusters**	**Frequency of Unclassified Objects**
10	7	0
20	2	0
40	2	0
60	1	0

Output 85.1.14 True Density, Estimated Density, and Cluster Membership by Various _K_ Values

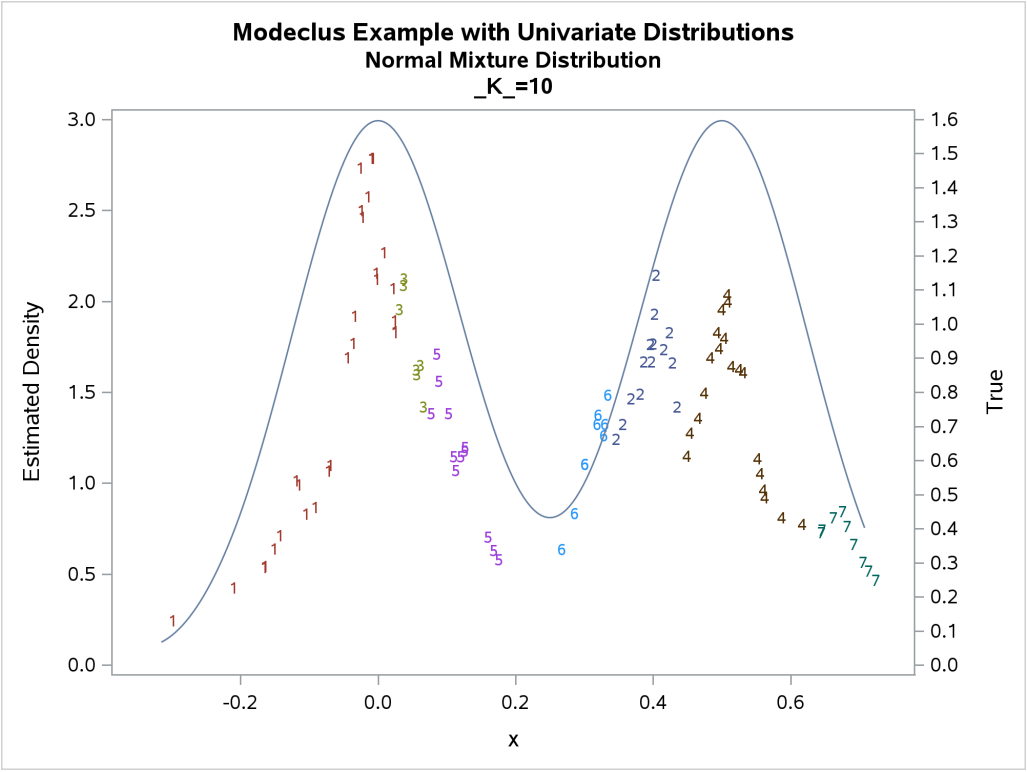

Output 85.1.14 *continued*

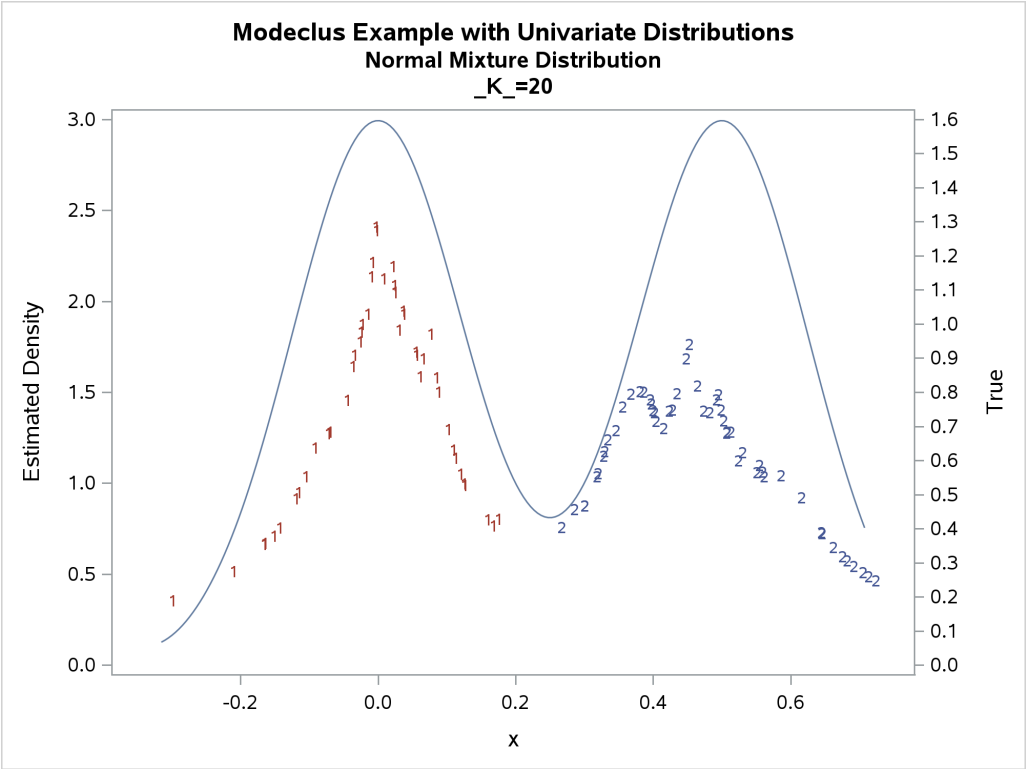

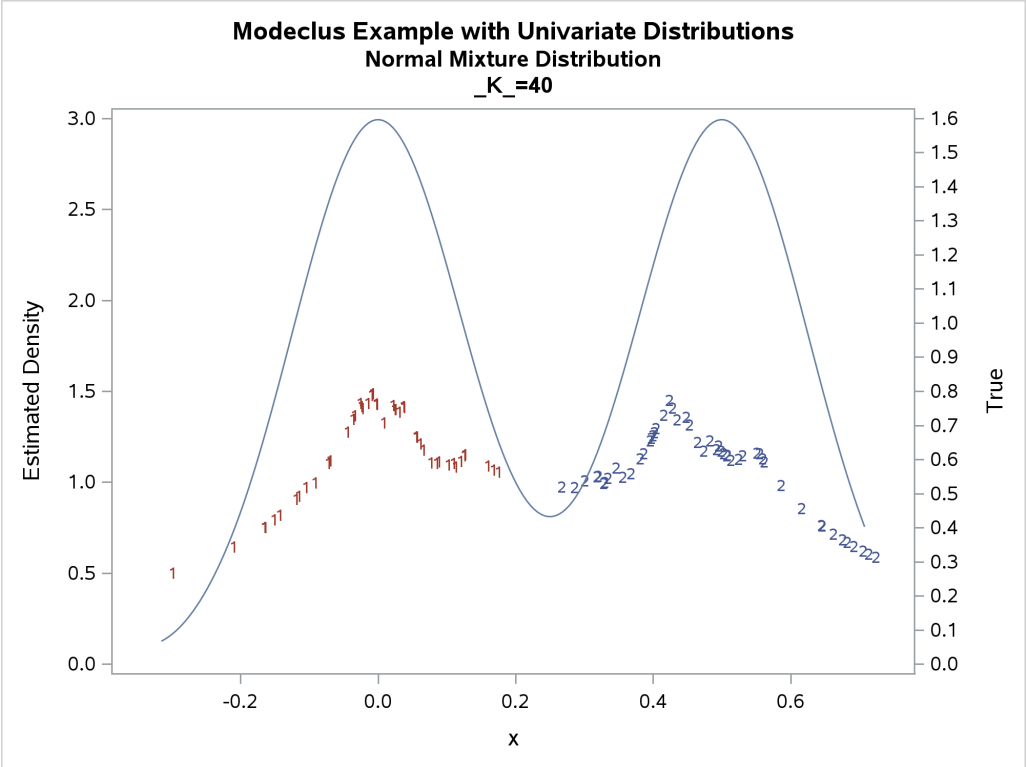

Output 85.1.14 *continued*

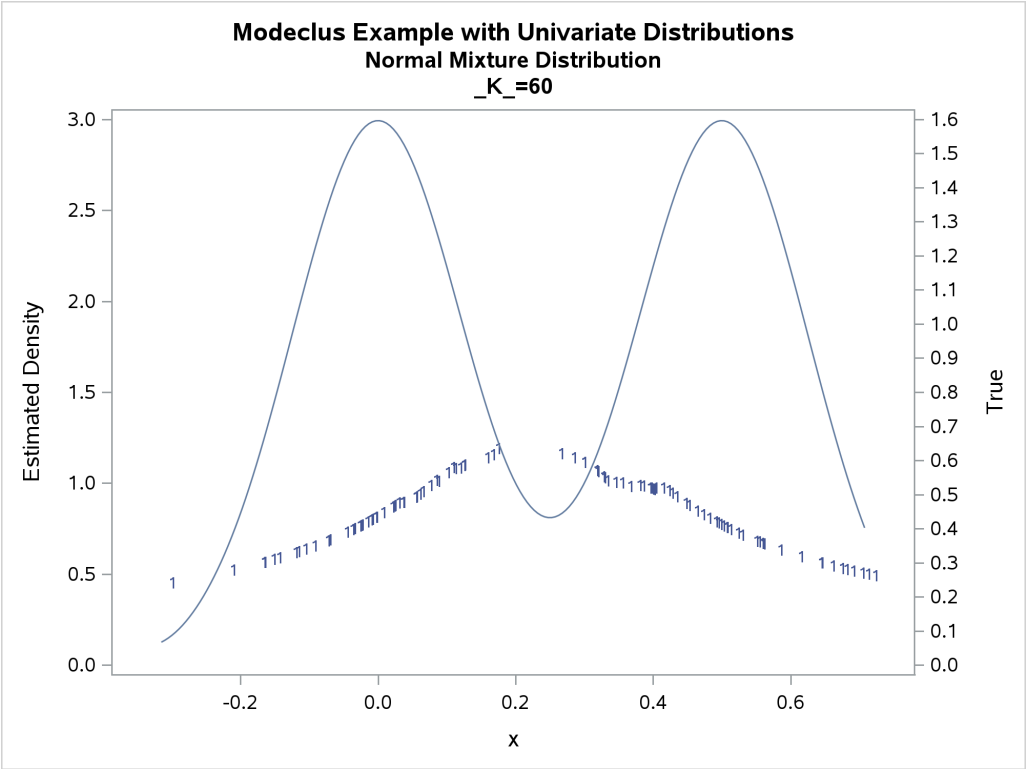

Output 85.1.15 Cluster Analysis of Sample from a Bimodal Mixture of Two Normal Distributions

Modeclus Example with Univariate Distributions
Normal Mixture Distribution

The MODECLUS Procedure

Cluster Summary		
R	**Number of Clusters**	**Frequency of Unclassified Objects**
0.05	5	0
0.1	2	0
0.2	2	0
0.3	1	0

Output 85.1.16 True Density, Estimated Density, and Cluster Membership by Various _R_= Values

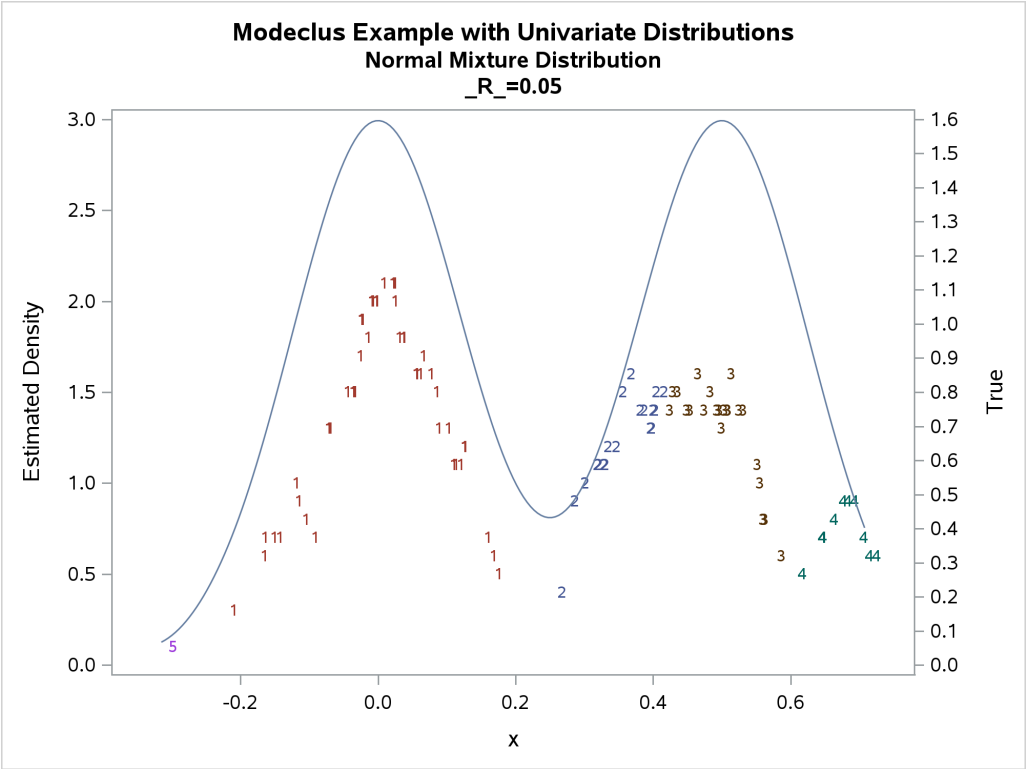

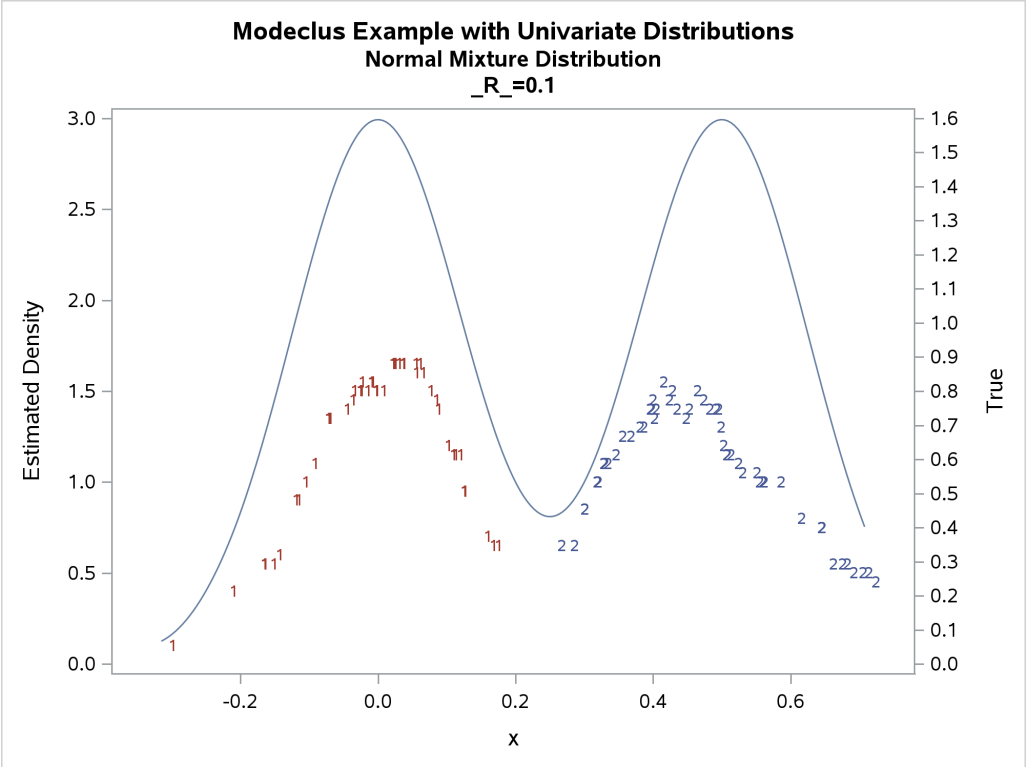

Output 85.1.16 *continued*

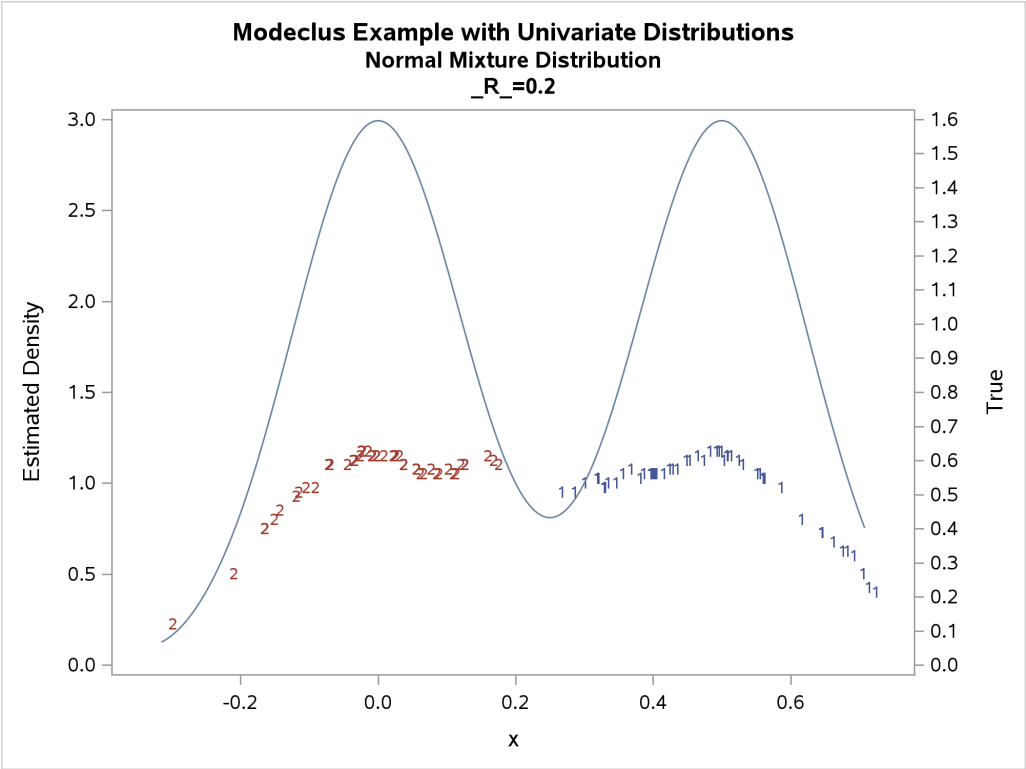

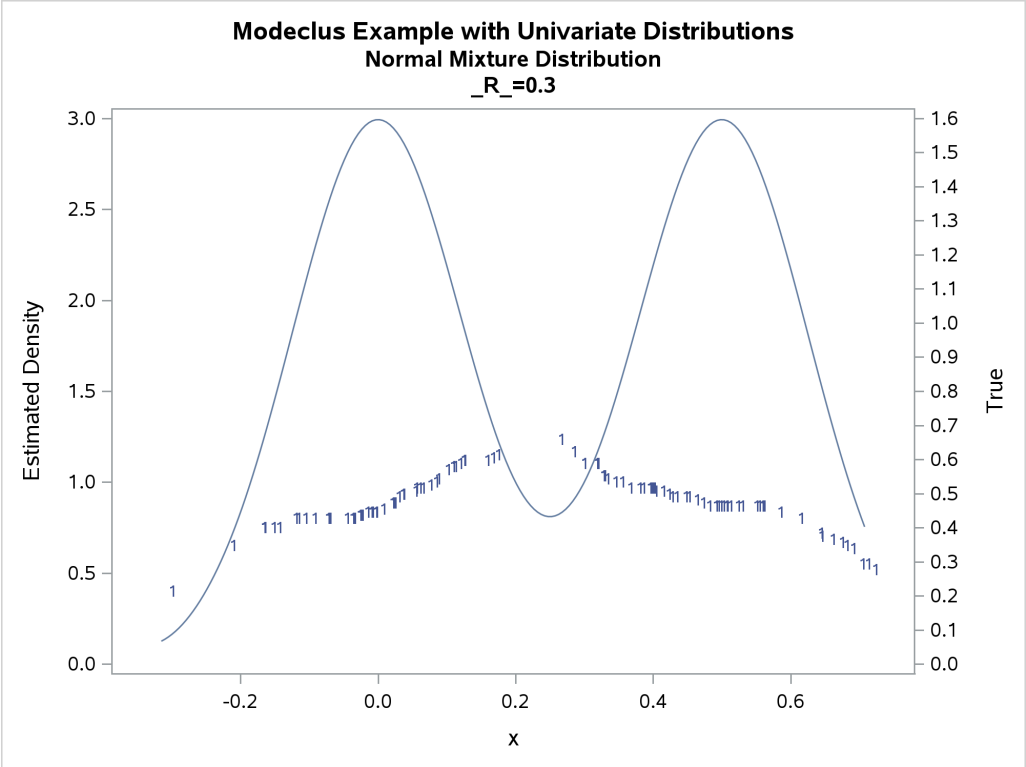

Output 85.1.17 Cluster Analysis of Sample from a Bimodal Mixture of Two Normal Distributions

Modeclus Example with Univariate Distributions
Normal Mixture Distribution
Cascaded Density Estimates Using Arithmetic Means

The MODECLUS Procedure

Cluster Summary

R	Cascade	Number of Clusters	Frequency of Unclassified Objects
0.05	1	5	0
0.05	2	4	0
0.05	4	4	0

Output 85.1.18 True Density, Estimated Density, and Cluster Membership by _R_=0.05 with Various _CASCAD_ Values

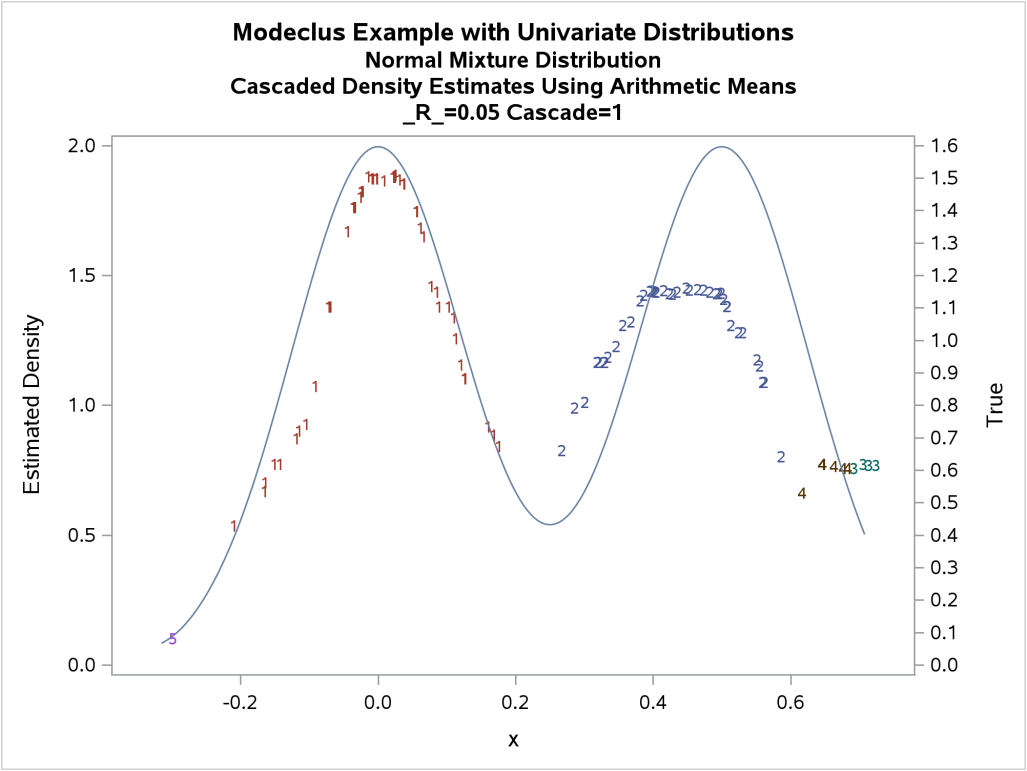

Output 85.1.18 *continued*

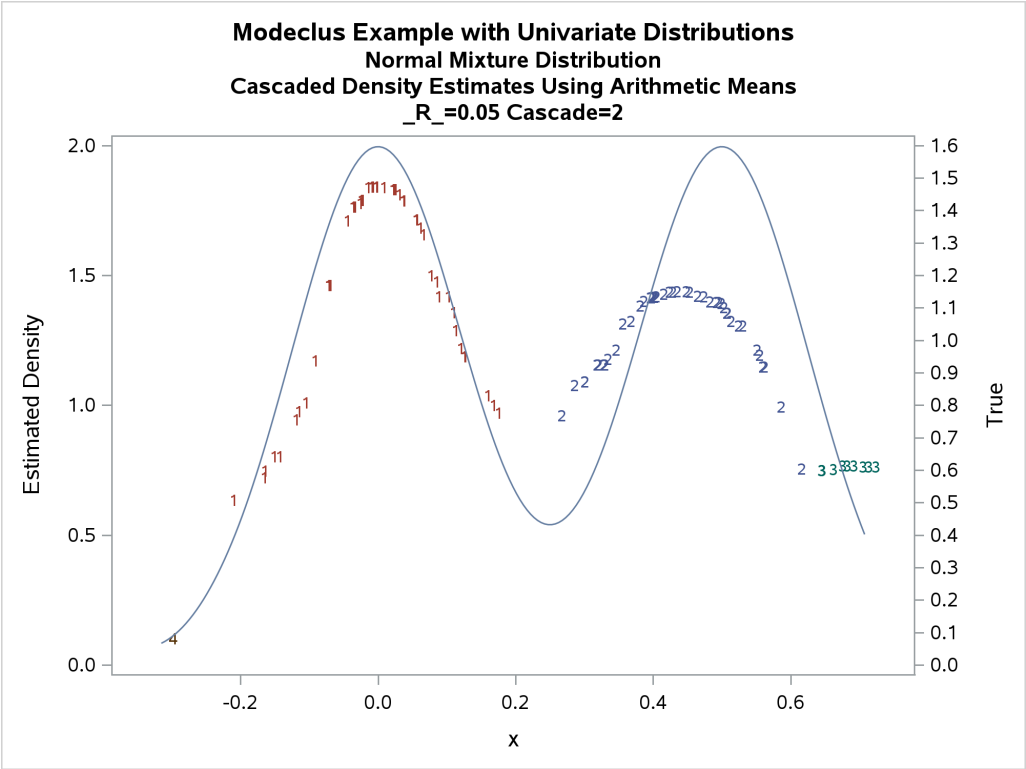

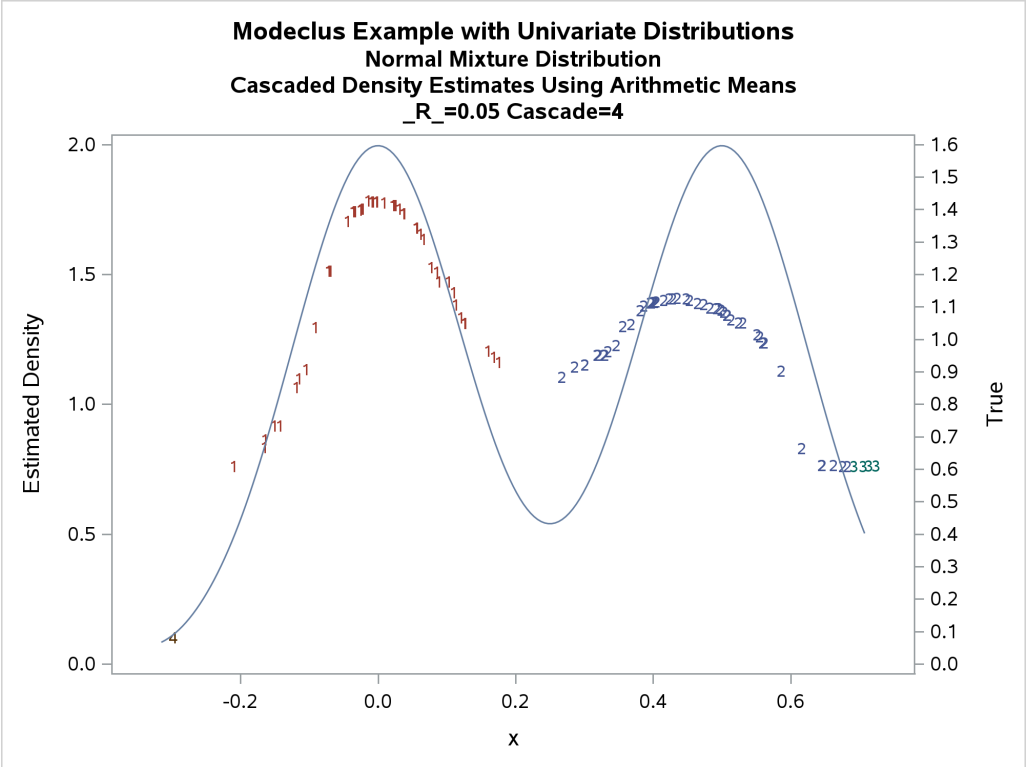

Example 85.2: Cluster Analysis of Flying Mileages between Ten American Cities

(View the complete code for this example (modecex2.sas) in the example repository.)

This example uses distance data and illustrates the use of the TRANSPOSE procedure and the DATA step to fill in the upper triangle of the distance matrix. A data set containing a table of flying mileages between 10 U.S. cities is available in the Sashelp library. The results are displayed in Output 85.2.1 through Output 85.2.3.

The following statements produce Output 85.2.1:

```
title 'Modeclus Analysis of 10 American Cities';
title2 'Based on Flying Mileages';

*-----Fill in Upper Triangle of Distance Matrix---------------;
proc transpose data=sashelp.mileages out=tran;
   copy city;
run;

data mileages(type=distance drop=col: _: i);
   merge sashelp.mileages tran;
   array var[10] atlanta--washingtondc;
   array col[10];
   do i = 1 to 10;
      var[i] = sum(var[i], col[i]);
   end;
run;

*-----Clustering with K-Nearest-Neighbor Density Estimates-----;
proc modeclus data=mileages all m=1 k=3;
   id CITY;
run;
```

Output 85.2.1 Clustering with K-Nearest-Neighbor Density Estimates

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure

Nearest Neighbor List		
City	**Neighbor**	**Distance**
Atlanta	Washington D.C.	543.0000000
	Chicago	587.0000000
Chicago	Atlanta	587.0000000
	Washington D.C.	597.0000000
Denver	Los Angeles	831.0000000
	Houston	879.0000000
Houston	Atlanta	701.0000000
	Denver	879.0000000
Los Angeles	San Francisco	347.0000000
	Denver	831.0000000
Miami	Atlanta	604.0000000
	Washington D.C.	923.0000000
New York	Washington D.C.	205.0000000
	Chicago	713.0000000
San Francisco	Los Angeles	347.0000000
	Seattle	678.0000000
Seattle	San Francisco	678.0000000
	Los Angeles	959.0000000
Washington D.C.	New York	205.0000000
	Atlanta	543.0000000

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure K=3 METHOD=1

Sums of Density Estimates Within Neighborhood						
Cluster	**City**	**Estimated Density**	**Same Cluster**	**Other Clusters**	**Total**	**Cluster Proportion Same/Total**
1	**Atlanta**	0.00025554	0.0005275	0	0.0005275	1.000
	Chicago	0.00025126	0.00053178	0	0.00053178	1.000
	Houston	0.00017065	0.00025554	0.00017065	0.00042619	0.600
	Miami	0.00016251	0.00053178	0	0.00053178	1.000
	New York	0.00021038	0.0005275	0	0.0005275	1.000
	Washington D.C.	0.00027624	0.00046592	0	0.00046592	1.000
2	**Denver**	0.00017065	0.00018051	0.00017065	0.00035115	0.514
	Los Angeles	0.00018051	0.00039189	0	0.00039189	1.000
	San Francisco	0.00022124	0.00033692	0	0.00033692	1.000
	Seattle	0.00015641	0.00040174	0	0.00040174	1.000

Output 85.2.1 *continued*

	Boundary Objects		-Cluster Proportions-	
City	**Density**	**Cluster**	**1**	**2**
Denver	0.0001706485	2	0.486	0.514
Houston	0.0001706485	1	0.600	0.400

Cluster Statistics

Cluster	**Frequency**	**Maximum Estimated Density**	**Boundary Frequency**	**Estimated Saddle Density**
1	6	0.00027624	1	0.00017065
2	4	0.00022124	1	0.00017065

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure

Cluster Summary

K	**Number of Clusters**	**Frequency of Unclassified Objects**
3	2	0

The following statements produce Output 85.2.2:

```
*------Clustering with Uniform-Kernel Density Estimates--------;
proc modeclus data=mileages all m=1 r=600 800;
   id CITY;
run;
```

Output 85.2.2 Clustering with Uniform-Kernel Density Estimates

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure

Nearest Neighbor List		
City	**Neighbor**	**Distance**
Atlanta	Washington D.C.	543.0000000
	Chicago	587.0000000
	Miami	604.0000000
	Houston	701.0000000
	New York	748.0000000
Chicago	Atlanta	587.0000000
	Washington D.C.	597.0000000
	New York	713.0000000
Houston	Atlanta	701.0000000
Los Angeles	San Francisco	347.0000000
Miami	Atlanta	604.0000000
New York	Washington D.C.	205.0000000
	Chicago	713.0000000
	Atlanta	748.0000000
San Francisco	Los Angeles	347.0000000
	Seattle	678.0000000
Seattle	San Francisco	678.0000000
Washington D.C.	New York	205.0000000
	Atlanta	543.0000000
	Chicago	597.0000000

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure
R=600 METHOD=1

Sums of Density Estimates Within Neighborhood						
Cluster	**City**	**Estimated Density**	**Same Cluster**	**Other Clusters**	**Total**	**Cluster Proportion Same/Total**
1	**Atlanta**	0.00025	0.00058333	0	0.00058333	1.000
	Chicago	0.00025	0.00058333	0	0.00058333	1.000
	New York	0.00016667	0.00033333	0	0.00033333	1.000
	Washington D.C.	0.00033333	0.00066667	0	0.00066667	1.000
2	**Los Angeles**	0.00016667	0.00016667	0	0.00016667	1.000
	San Francisco	0.00016667	0.00016667	0	0.00016667	1.000
3	**Denver**	0.00008333	0	0	0	.
4	**Houston**	0.00008333	0	0	0	.
5	**Miami**	0.00008333	0	0	0	.
6	**Seattle**	0.00008333	0	0	0	.

Output 85.2.2 *continued*

No Boundary Objects

Cluster Statistics				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density
1	4	0.00033333	0	.
2	2	0.00016667	0	.
3	1	0.00008333	0	.
4	1	0.00008333	0	.
5	1	0.00008333	0	.
6	1	0.00008333	0	.

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure
R=800 METHOD=1

Sums of Density Estimates Within Neighborhood						
Cluster	City	Estimated Density	Same Cluster	Other Clusters	Total	Cluster Proportion Same/Total
1	Atlanta	0.000375	0.001	0	0.001	1.000
	Chicago	0.00025	0.000875	0	0.000875	1.000
	Houston	0.000125	0.000375	0	0.000375	1.000
	Miami	0.000125	0.000375	0	0.000375	1.000
	New York	0.00025	0.000875	0	0.000875	1.000
	Washington D.C.	0.00025	0.000875	0	0.000875	1.000
2	Los Angeles	0.000125	0.0001875	0	0.0001875	1.000
	San Francisco	0.0001875	0.00025	0	0.00025	1.000
	Seattle	0.000125	0.0001875	0	0.0001875	1.000
3	Denver	0.0000625	0	0	0	.

No Boundary Objects

Cluster Statistics				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density
1	6	0.000375	0	.
2	3	0.0001875	0	.
3	1	0.0000625	0	.

Output 85.2.2 *continued*

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure

Cluster Summary		
R	Number of Clusters	Frequency of Unclassified Objects
600	6	0
800	3	0

The following statements produce Output 85.2.3:

```
*------Clustering Neighborhoods Extended to Nearest Neighbor--------;
proc modeclus data=mileages list m=1 ck=2 r=600 800;
   id CITY;
run;
```

Output 85.2.3 Uniform-Kernel Density Estimates, Clustering Neighborhoods Extended to Nearest Neighbor

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure CK=2 R=600 METHOD=1

Sums of Density Estimates Within Neighborhood						
Cluster	City	Estimated Density	Same Cluster	Other Clusters	Total	Cluster Proportion Same/Total
1	Atlanta	0.00025	0.00058333	0	0.00058333	1.000
	Chicago	0.00025	0.00058333	0	0.00058333	1.000
	Houston	0.00008333	0.00025	0	0.00025	1.000
	Miami	0.00008333	0.00025	0	0.00025	1.000
	New York	0.00016667	0.00033333	0	0.00033333	1.000
	Washington D.C.	0.00033333	0.00066667	0	0.00066667	1.000
2	Denver	0.00008333	0.00016667	0	0.00016667	1.000
	Los Angeles	0.00016667	0.00016667	0	0.00016667	1.000
	San Francisco	0.00016667	0.00016667	0	0.00016667	1.000
	Seattle	0.00008333	0.00016667	0	0.00016667	1.000

Cluster Statistics				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density
1	6	0.00033333	0	.
2	4	0.00016667	0	.

Output 85.2.3 *continued*

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure
CK=2 R=800 METHOD=1

Sums of Density Estimates Within Neighborhood

Cluster	City	Estimated Density	Same Cluster	Other Clusters	Total	Cluster Proportion Same/Total
1	Atlanta	0.000375	0.001	0	0.001	1.000
	Chicago	0.00025	0.000875	0	0.000875	1.000
	Houston	0.000125	0.000375	0	0.000375	1.000
	Miami	0.000125	0.000375	0	0.000375	1.000
	New York	0.00025	0.000875	0	0.000875	1.000
	Washington D.C.	0.00025	0.000875	0	0.000875	1.000
2	Denver	0.0000625	0.000125	0	0.000125	1.000
	Los Angeles	0.000125	0.0001875	0	0.0001875	1.000
	San Francisco	0.0001875	0.00025	0	0.00025	1.000
	Seattle	0.000125	0.0001875	0	0.0001875	1.000

Cluster Statistics

Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density
1	6	0.000375	0	.
2	4	0.0001875	0	.

Modeclus Analysis of 10 American Cities Based on Flying Mileages

The MODECLUS Procedure

Cluster Summary

R	CK	Number of Clusters	Frequency of Unclassified Objects
600	2	2	0
800	2	2	0

Example 85.3: Cluster Analysis with Significance Tests

(View the complete code for this example (modecex3.sas) in the example repository.)

This example uses artificial data containing two clusters. One cluster is from a circular bivariate normal distribution. The other is a ring-shaped cluster that completely surrounds the first cluster. Without significance tests, the ring is divided into several sample clusters for any degree of smoothing that yields reasonable density estimates. The JOIN= option puts the ring back together. Output 85.3.1 displays a short summary generated from the first PROC MODECLUS statement. Output 85.3.2 contains a series of tables produced from the second PROC MODECLUS statement. The lack of *p*-value in the JOIN= option makes joining continue until only one cluster remains (see the description of the JOIN= option). The cluster memberships are then plotted as displayed in Output 85.3.1 through Output 85.3.8.

The following statements produce Output 85.3.1 through Output 85.3.8:

```
title  'Modeclus Analysis with the JOIN= option';
title2 'A Normal Cluster Surrounded by a Ring Cluster';

data circle; keep x y;
   c=1;
   do n=1 to 30;
      x=rannor(5);
      y=rannor(5);
      output;
   end;

   c=2;
   do n=1 to 300;
      x=rannor(5);
      y=rannor(5);
      z=rannor(5)+8;
      l=z/sqrt(x**2+y**2);
      x=x*l;
      y=y*l;
      output;
   end;
run;

proc modeclus data=circle m=1 r=1 to 3.5 by .25 join=20 short;
run;

proc modeclus data=circle m=1 r=2.5 join out=out;
run;

proc sgplot data=out noautolegend;
   yaxis values=(-10 to 10 by 5);
   xaxis values=(-15 to 15 by 5);
   scatter y=y x=x / group=cluster Markerchar=cluster;
   by _NJOIN_;
run;
```

Output 85.3.1 Significance Tests with the JOIN=20 and SHORT Options

Modeclus Analysis with the JOIN= option
A Normal Cluster Surrounded by a Ring Cluster

The MODECLUS Procedure

Cluster Summary				
R	Number of Clusters Joined	Maximum P-value	Number of Clusters	Frequency of Unclassified Objects
1	36	0.9339	1	301
1.25	20	0.7131	1	301
1.5	10	0.3296	1	300
1.75	5	0.1990	2	0
2	5	0.0683	2	0
2.25	3	0.0504	2	0
2.5	4	0.0301	2	0
2.75	3	0.0585	2	0
3	5	0.0003	1	0
3.25	4	0.1923	2	0
3.5	4	0.0000	1	0

Output 85.3.2 Significance Tests with the JOIN Option

Modeclus Analysis with the JOIN= option
A Normal Cluster Surrounded by a Ring Cluster

The MODECLUS Procedure
R=2.5 METHOD=1

Cluster Statistics					-Saddle Test: Version 92.7-				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density	Mode Count	Saddle Count	Overlap Count	Z	Approx P-value
1	103	0.00617328	22	0.00308664	39	19	0	2.495	0.5055
2	71	0.00571029	20	0.0043213	36	27	9	1.193	0.999
3	53	0.00509296	18	0.00401263	32	25	10	0.986	0.9999
4	45	0.00478429	19	0.00354964	30	22	14	1.429	0.9924
5	30	0.00462996	0	.	29	0	.	3.611	0.0301
6	28	0.00370397	17	0.00354964	23	22	9	0.000	1

Cluster 6 with P-value 1.0000 will be joined to cluster 4.

Cluster Statistics					-Saddle Test: Version 92.7-				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density	Mode Count	Saddle Count	Overlap Count	Z	Approx P-value
1	103	0.00617328	22	0.00308664	39	19	0	2.495	0.5055
2	71	0.00571029	20	0.0043213	36	27	9	1.193	0.999
3	53	0.00509296	18	0.00401263	32	25	10	0.986	0.9999
4	73	0.00478429	13	0.00293231	30	18	0	1.588	0.9778
5	30	0.00462996	0	.	29	0	.	3.611	0.0301

Output 85.3.2 *continued*

Cluster 3 with P-value 0.9999 will be joined to cluster 1.

Cluster Statistics					-Saddle Test: Version 92.7-				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density	Mode Count	Saddle Count	Overlap Count	Z	Approx P-value
1	156	0.00617328	17	0.00246931	39	15	0	3.130	0.1318
2	71	0.00571029	20	0.0043213	36	27	9	1.193	0.999
3	73	0.00478429	13	0.00293231	30	18	0	1.588	0.9778
4	30	0.00462996	0	.	29	0	.	3.611	0.0301

Cluster 2 with P-value 0.9990 will be joined to cluster 3.

Cluster Statistics					-Saddle Test: Version 92.7-				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density	Mode Count	Saddle Count	Overlap Count	Z	Approx P-value
1	156	0.00617328	17	0.00246931	39	15	0	3.130	0.1318
2	144	0.00571029	14	0.00293231	36	18	0	2.313	0.6447
3	30	0.00462996	0	.	29	0	.	3.611	0.0301

Cluster 2 with P-value 0.6447 will be joined to cluster 1.

Cluster Statistics					-Saddle Test: Version 92.7-				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density	Mode Count	Saddle Count	Overlap Count	Z	Approx P-value
1	300	0.00617328	0	.	39	0	.	4.246	0.0026
2	30	0.00462996	0	.	29	0	.	3.611	0.0301

Cluster 2 with P-value 0.0301 will be dissolved.

Cluster Statistics					-Saddle Test: Version 92.7-				
Cluster	Frequency	Maximum Estimated Density	Boundary Frequency	Estimated Saddle Density	Mode Count	Saddle Count	Overlap Count	Z	Approx P-value
1	300	0.00617328	0	.	39	0	.	4.246	0.0026

30 observations were unassigned.

Cluster 1 with P-value 0.0026 will be dissolved.

Output 85.3.2 *continued*

Modeclus Analysis with the JOIN= option
A Normal Cluster Surrounded by a Ring Cluster

The MODECLUS Procedure

Cluster Summary				
R	Number of Clusters Joined	Maximum P-value	Number of Clusters	Frequency of Unclassified Objects
2.5	0	1.0000	6	0
2.5	1	0.9999	5	0
2.5	2	0.9990	4	0
2.5	3	0.6447	3	0
2.5	4	0.0301	2	0
2.5	5	0.0026	1	30

Output 85.3.3 Cluster Memberships When Number of Clusters Joined=0

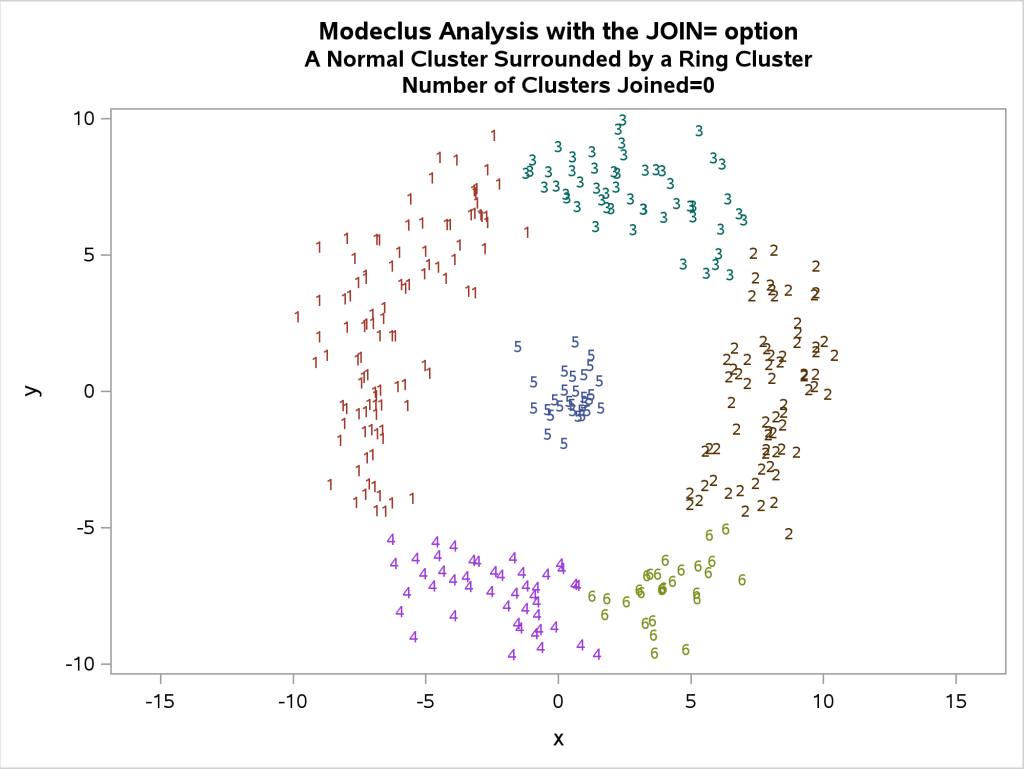

Output 85.3.4 Cluster Memberships When Number of Clusters Joined=1

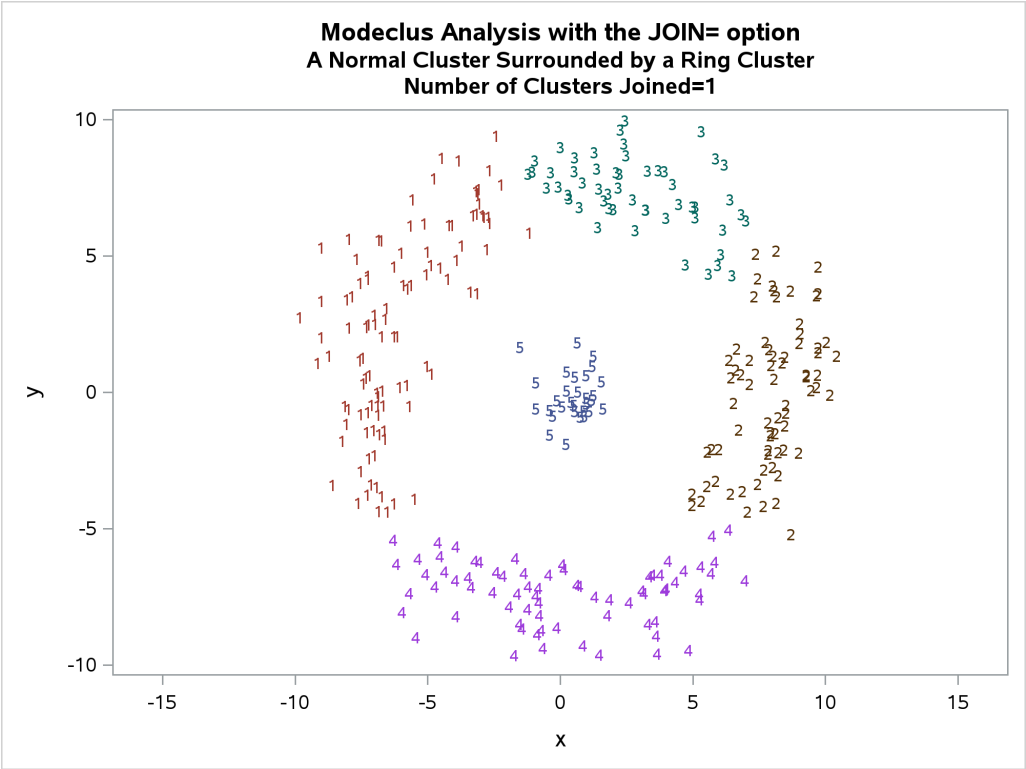

Output 85.3.5 Cluster Memberships When Number of Clusters Joined=2

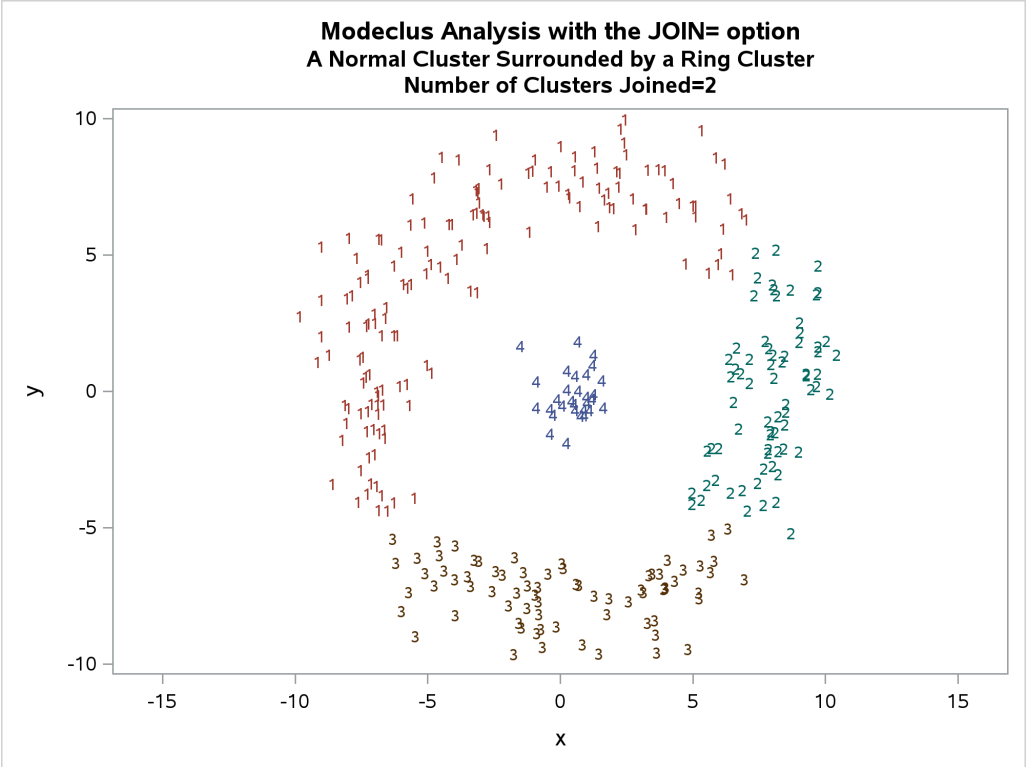

Output 85.3.6 Cluster Memberships When Number of Clusters Joined=3

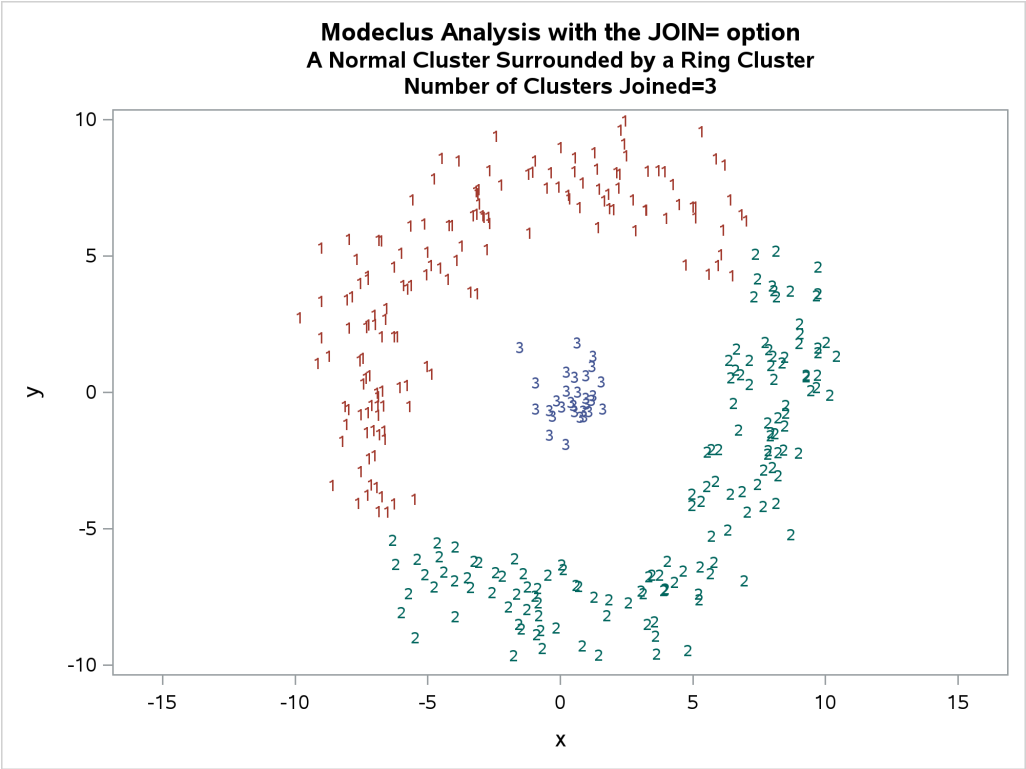

Output 85.3.7 Cluster Memberships When Number of Clusters Joined=4

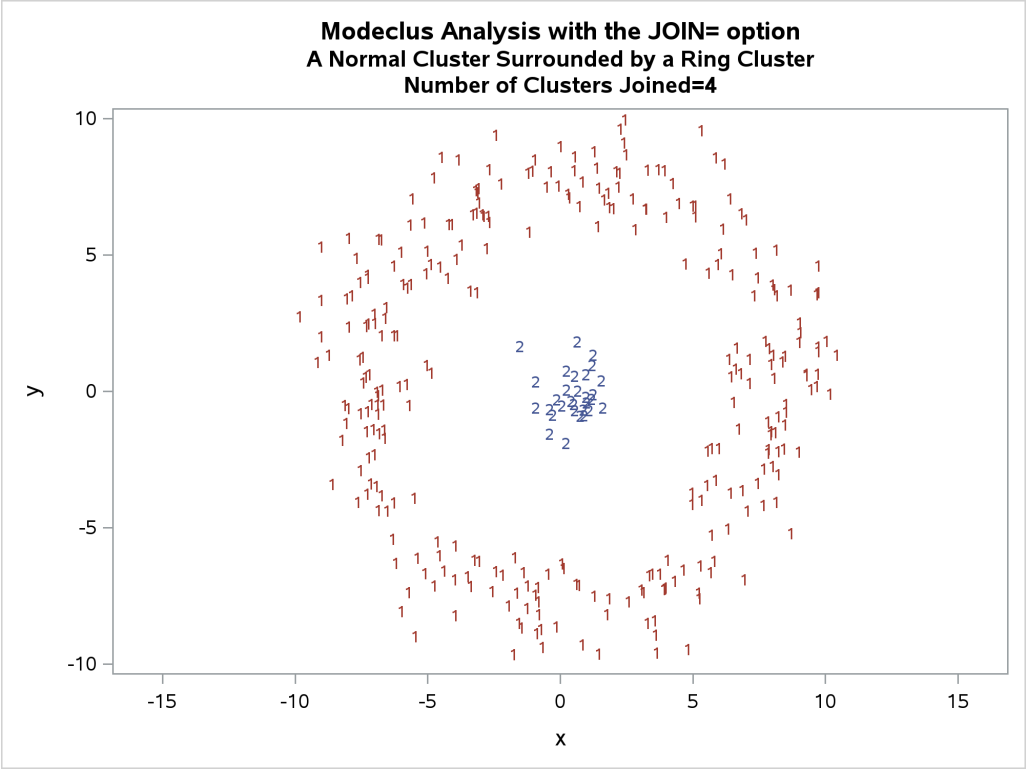

Output 85.3.8 Cluster Memberships When Number of Clusters Joined=5

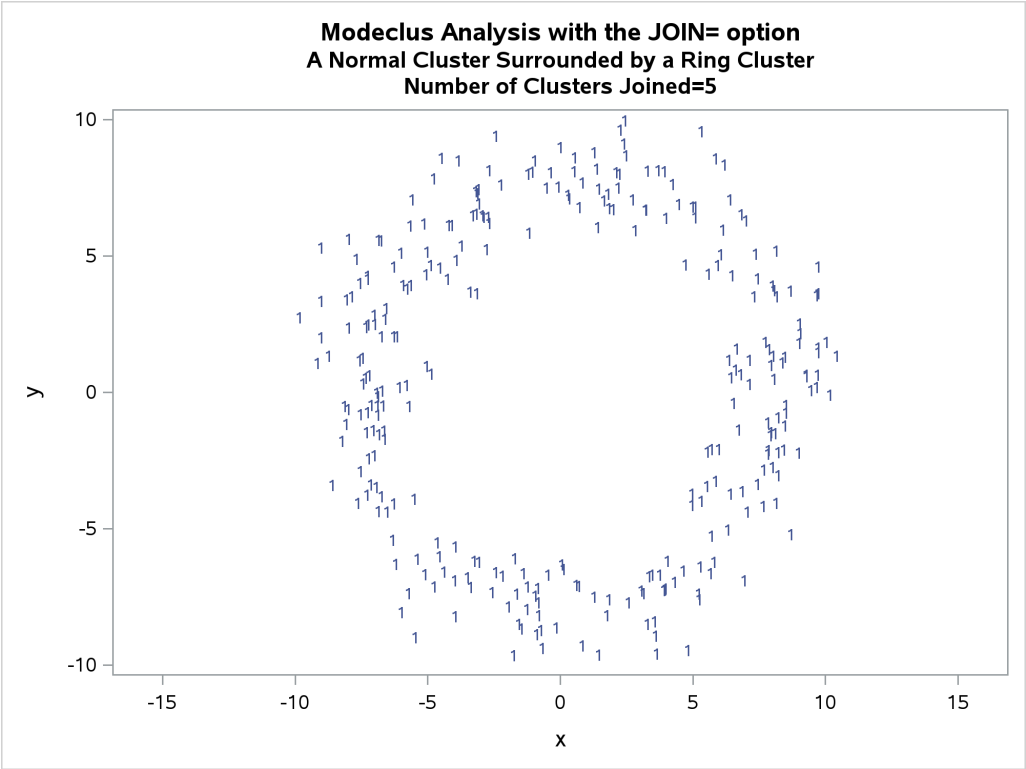

Example 85.4: Cluster Analysis: Hertzsprung-Russell Plot

(View the complete code for this example (modecex4.sas) in the example repository.)

This example uses computer-generated data to mimic a Hertzsprung-Russell plot (Struve and Zebergs 1962, p. 259) of the temperature and luminosity of stars. The data are plotted and displayed in Output 85.4.1. It appears that there are two main groups of stars and a collection of isolated stars. The long straggling group of points appearing diagonally across the figure represents the main group of stars; the more compact group in the top-right corner contains giant stars. The JOIN= option is specified at a 0.05 significance level with various smoothing parameters. The CK=5 option is specified in order to prevent the numerous outliers from forming separate clusters. The results from PROC MODECLUS is displayed in Output 85.4.2. The cluster memberships are then plotted by PROC SGPLOT, as displayed in Output 85.4.3 through Output 85.4.5.

Note that the graphic output from PROC SGPLOT in Output 85.4.3 is not available when _R_ = 2.5 because only one cluster remains after joining at a 5% significance level, and the results are not written to the OUT= data set. See the description of the JOIN= option). for more information.

The following statements produce Output 85.4.1 through Output 85.4.5:

```
title 'Hertzsprung-Russell Plot of Visible Stars';
title2 'Computer-Generated Simulated Data';

data hr;
   input x y @@;
```

```
   label x='-Temperature'
         y='-Luminosity';
   datalines;
1.0   12.8   0.9  13.7   0.9  12.9   1.0  12.3   1.0  12.2   2.6  10.9
2.4   10.9   2.5  11.2   2.3  11.5   2.6  12.0   2.4  12.1   2.3  10.9
2.6   11.5   2.5  11.9   2.4  11.0   3.4  11.1   3.3  11.2   3.4  11.1
3.4    9.9   3.2  10.4   3.5  10.8   3.4  11.0   3.3  11.2   3.3  10.8
3.5   10.0   3.5  10.2   3.4  10.2   3.6  10.6   3.7  10.4   3.7  10.1
3.4   10.7   3.4  10.8   3.3  11.0   3.6  10.8   3.5  10.1   4.5  10.3
4.6    9.4   4.3  10.3   4.6   9.4   4.4   9.9   4.5  10.4   4.4   9.9
4.6    9.4   4.4  10.7   4.4   9.3   4.4   9.5   4.1  10.6   4.4  10.6
4.5   10.3   4.4  10.0   4.2   9.8   4.5   9.5   4.2  13.4   4.6  10.4
4.5    9.8   5.8   8.8   5.6   8.4   5.6  13.9   5.7   9.5   5.6  14.5
5.6    9.2   5.7   8.7   5.7   9.4   5.7   9.3   5.6   9.4   5.8   9.8
5.5    8.8   5.8   8.9   5.7   9.4   5.6  12.1   5.4  10.1   5.8   9.3
5.9    9.0   5.7  10.0   5.6   9.3   6.6   8.6   6.7   8.5   6.7  12.5

   ... more lines ...

26.4  14.1  26.6  14.2  27.5  13.7  27.6  14.4  27.8  14.0  27.4  14.7
25.8  13.5  25.6  13.6  26.8  14.4  26.4  19.0  26.0  13.4  27.3  14.0
27.5  14.3  27.4  14.5  26.3  13.8  26.9  13.7  26.3  13.7  27.7  14.3
27.3  14.1  28.3  14.2  17.4  15.5  13.8  15.2  12.0  11.6  14.1  12.8
17.1  10.2  16.9  15.4  18.5  12.6  14.2  16.1  23.2   6.6  11.4  12.4
20.4  11.7  20.9   8.1  18.9  13.7  16.9   9.7  15.5   9.9  18.3  14.2
19.3  13.7  17.0  12.9  10.1  11.6  17.9  13.5  14.3   1.4  13.1  -0.8
8.1  -0.9  20.0   7.0  21.0   8.5  15.6  13.2
;

proc sgplot data=hr;
   scatter y=y x=x;
run;

proc modeclus data=hr m=1 r=1 1.5 2 2.5 ck=5
              join=.05 short out=out;
run;

title2 'MODECLUS Analysis';

proc sgplot data=out;
   scatter y=y x=x/group=cluster;
   by _R_;
run;
```

Output 85.4.1 Scatter Plot of Data

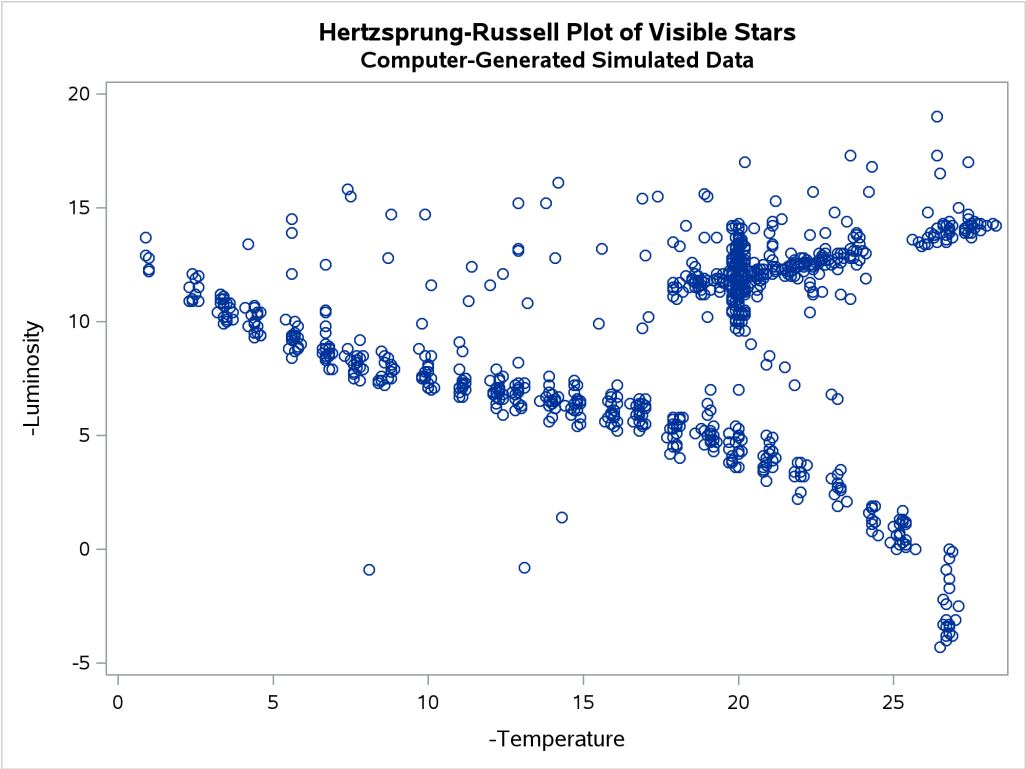

Output 85.4.2 Results from PROC MODECLUS

Hertzsprung-Russell Plot of Visible Stars
Computer-Generated Simulated Data

The MODECLUS Procedure

Cluster Summary

R	CK	Number of Clusters Joined	Maximum P-value	Number of Clusters	Frequency of Unclassified Objects
1	5	14	0.0001	2	0
1.5	5	6	0.0000	3	0
2	5	4	0.0000	2	0
2.5	5	2	0.0000	1	0

Output 85.4.3 Scatter Plots of Cluster Memberships by _R_= 1

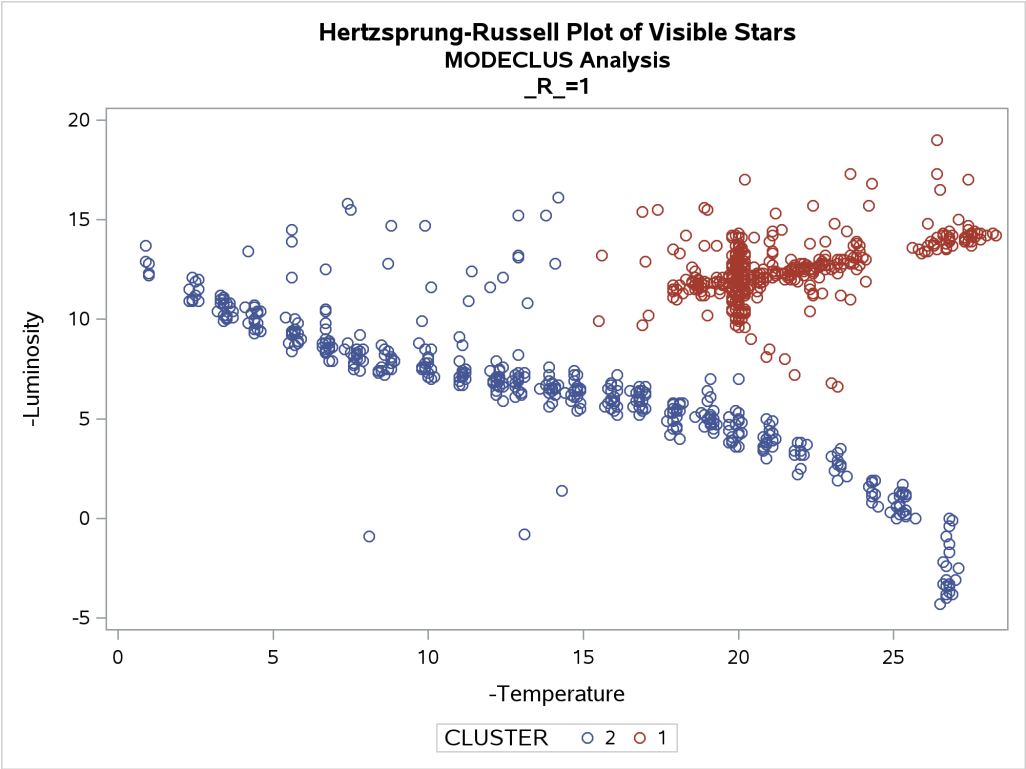

Output 85.4.4 Scatter Plots of Cluster Memberships by _R_= 1.5

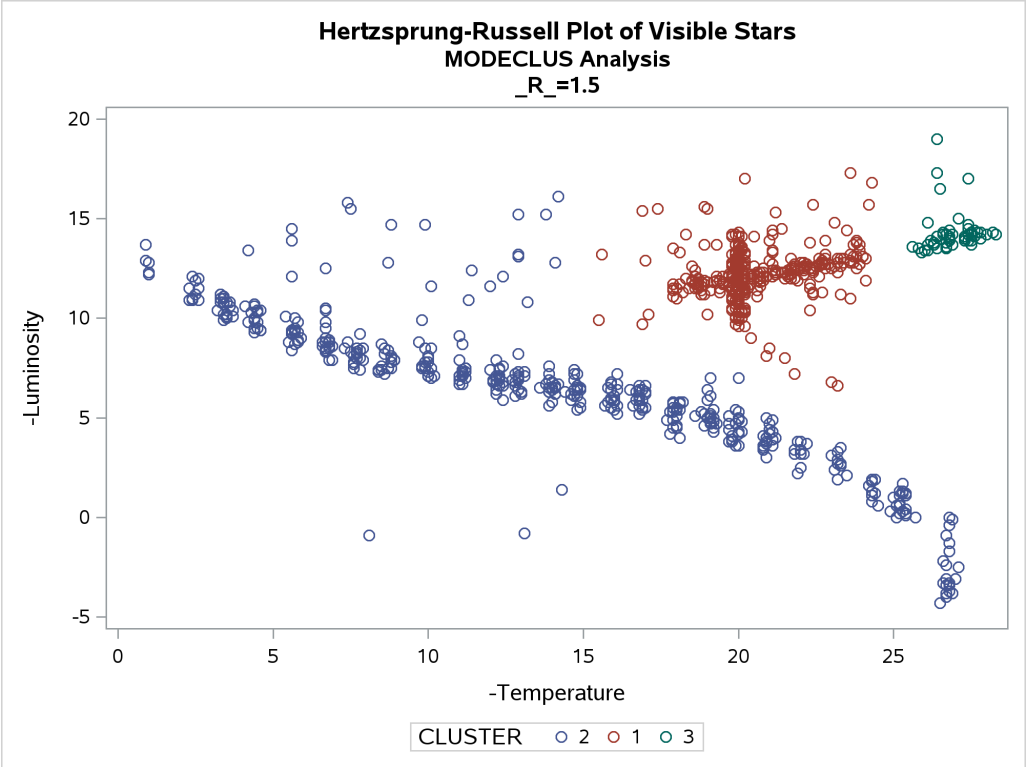

Output 85.4.5 Scatter Plots of Cluster Memberships by _R_=2

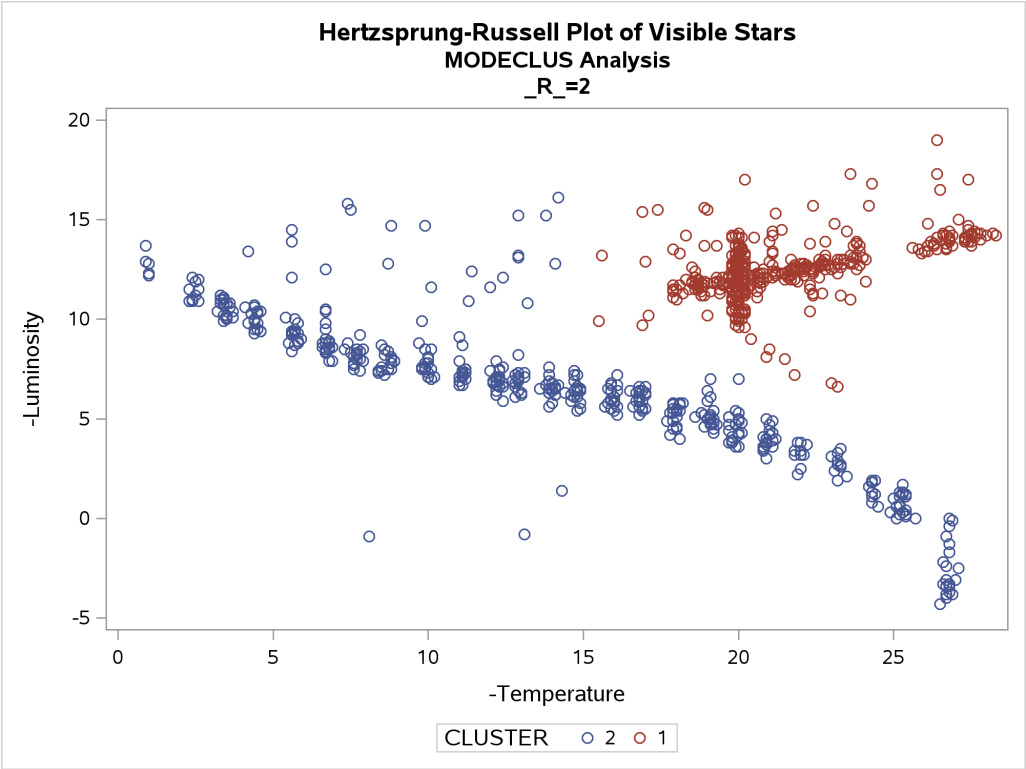

Example 85.5: Using the TRACE Option When METHOD=6

(View the complete code for this example (modecex5.sas) in the example repository.)

To illustrate how the TRACE option can help you to understand the clustering process when METHOD=6 is specified, the following data set is created with 12 observations:

```
data test;
   input x @@;
   datalines;
1 2 3 4 5 7.5 9 11.5 13 14.5 15 16
;
```

The first five observations seem to be close to each other, and the last five observations seem to be close to each other. Observation 6 is separated from the first five observations with a (Euclidean) distance of 2.5, and the same distance separates observation 7 from the last five observations. Observations 6 and 7 differ by 1.5.

Suppose METHOD=6 with a radius of 2.5 is chosen for the cluster analysis. You can specify the TRACE option to understand how each observation is assigned.

The following statements produce Output 85.5.1 and Output 85.5.2:

```
/*-- METHOD=6 with TRACE and THRESHOLD=0.5 (default) --*/
title 'METHOD=6 with TRACE and THRESHOLD=0.5 (default)';

proc modeclus data=test method=6 r=2.5 trace short out=out;
```

```
   var x;
run;

title2 'Plot of DENSITY*X=CLUSTER';

proc sgplot data=out;
   scatter y=density x=x / group=cluster datalabel=_obs_;
run;
```

Output 85.5.1 Partial Output of METHOD=6 with TRACE and Default THRESHOLD=

METHOD=6 with TRACE and THRESHOLD=0.5 (default)

The MODECLUS Procedure
R=2.5 METHOD=6

Trace of Clustering Algorithm				
		Cluster		
Obs	Density	Old	New	Ratio
3	0.0833333	-1	1	M
2	0.0666667	0	1	N
4	0.0666667	0	1	N
1	0.0500000	0	1	N
5	0.0666667	0	1	N
6	0.0500000	0	1	0.571
7	0.0500000	-1	1	0.500
9	0.0666667	-1	2	M
8	0.0500000	0	2	N
11	0.0666667	-1	2	S
12	0.0500000	0	2	N
10	0.0666667	-1	2	S

METHOD=6 with TRACE and THRESHOLD=0.5 (default)

The MODECLUS Procedure

Cluster Summary		
R	Number of Clusters	Frequency of Unclassified Objects
2.5	2	0

Output 85.5.2 Density Plot

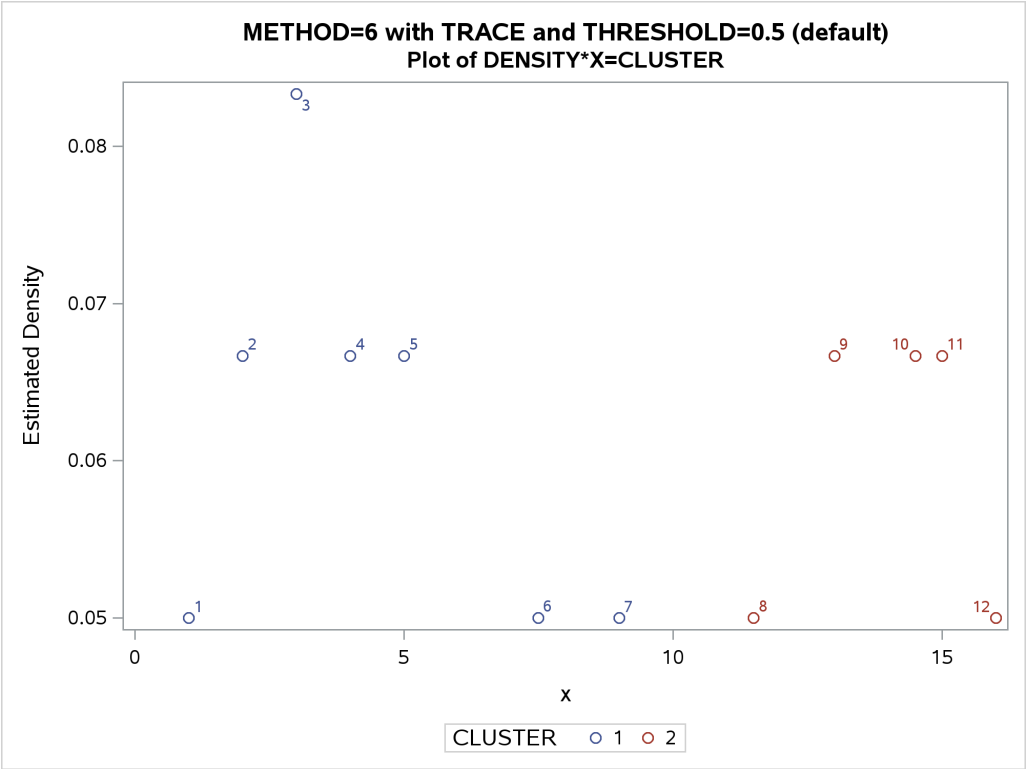

Note that in Output 85.5.1, observation 7 is originally a seed (indicated by a value of –1 in the "Old" column) and then assigned to cluster 1. This is because the ratio of observation 7 to cluster 1 is 0.5 and is not less than the default value of the THRESHOLD= option (0.5).

If the value of the THRESHOLD= option is increased to 0.55, observation 7 should be excluded from cluster 1 and the cluster membership of observation 7 is changed.

The following statements produce Output 85.5.3 and Output 85.5.4:

```
/*-- METHOD=6 with TRACE and THRESHOLD=0.55 --*/
title 'METHOD=6 with TRACE and THRESHOLD=0.55';

proc modeclus data=test method=6 r=2.5 trace threshold=0.55 short out=out;
   var x;
run;

title2 'Plot of DENSITY*X=CLUSTER with TRACE and THRESHOLD=0.55';

proc sgplot data=out;
   scatter y=density x=x / group=cluster datalabel=_obs_;
run;
```

Output 85.5.3 Partial Output of METHOD=6 with TRACE and THRESHOLD=.55

METHOD=6 with TRACE and THRESHOLD=0.55

The MODECLUS Procedure
R=2.5 METHOD=6

Trace of Clustering Algorithm				
		Cluster		
Obs	Density	Old	New	Ratio
3	0.0833333	-1	1	M
2	0.0666667	0	1	N
4	0.0666667	0	1	N
1	0.0500000	0	1	N
5	0.0666667	0	1	N
6	0.0500000	0	1	0.571
9	0.0666667	-1	2	M
8	0.0500000	0	2	N
11	0.0666667	-1	2	S
12	0.0500000	0	2	N
10	0.0666667	-1	2	S
7	0.0500000	-1	2	S

METHOD=6 with TRACE and THRESHOLD=0.55

The MODECLUS Procedure

Cluster Summary		
R	Number of Clusters	Frequency of Unclassified Objects
2.5	2	0

Output 85.5.4 Density Plot

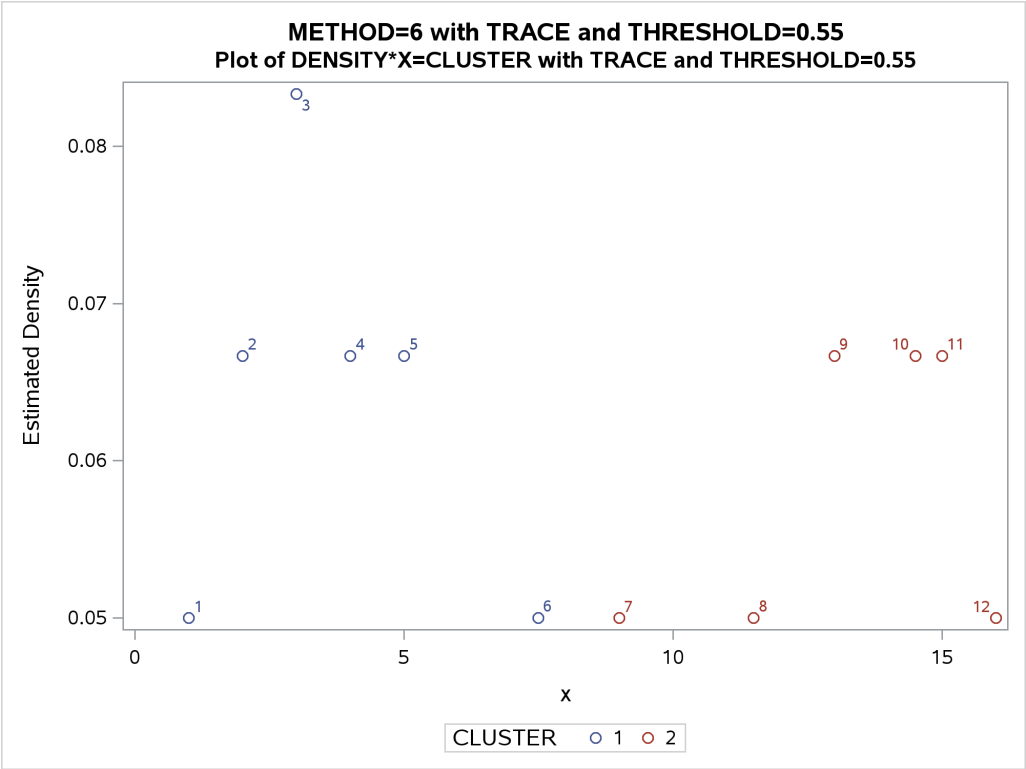

In Output 85.5.3, observation 7 is a seed that is excluded by cluster 1 because its ratio to cluster 1 is less than 0.55. Being a neighbor of a member (observation 8) of cluster 2, observation 7 eventually joins cluster 2 even though it remains a "SEED." (See **Step 2.2** in the section "METHOD=6" on page 6981.)

References

Gitman, I. (1973). "An Algorithm for Nonsupervised Pattern Classification." *IEEE Transactions on Systems, Man, and Cybernetics* SMC-3:66–74.

Hartigan, J. A., and Hartigan, P. M. (1985). "The Dip Test of Unimodality." *Annals of Statistics* 13:70–84.

Huizinga, D. H. (1978). *A Natural or Mode Seeking Cluster Analysis Algorithm*. Technical Report 78-1, Behavioral Research Institute, Boulder, CO.

Koontz, W. L. G., and Fukunaga, K. (1972a). "Asymptotic Analysis of a Nonparametric Clustering Technique." *IEEE Transactions on Computers* C-21:967–974.

Koontz, W. L. G., and Fukunaga, K. (1972b). "A Nonparametric Valley-Seeking Technique for Cluster Analysis." *IEEE Transactions on Computers* C-21:171–178.

Koontz, W. L. G., Narendra, P. M., and Fukunaga, K. (1976). "A Graph-Theoretic Approach to Nonparametric Cluster Analysis." *IEEE Transactions on Computers* C-25:936–944.

Minnotte, M. C. (1992). "A Test of Mode Existence with Applications to Multimodality." Ph.D. diss., Department of Statistics, Rice University.

Mizoguchi, R., and Shimura, M. (1980). "A Nonparametric Algorithm for Detecting Clusters Using Hierarchical Structure." *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-2:292–300.

Müller, D. W., and Sawitzki, G. (1991). "Excess Mass Estimates and Tests for Multimodality." *Journal of the American Statistical Association* 86:738–746.

Polonik, W. (1993). *Measuring Mass Concentrations and Estimating Density Contour Clusters—an Excess Mass Approach*. Technical Report 7, Beiträge zur Statistik, Universität Heidelberg.

Sarle, W. S. (1982). "Cluster Analysis by Least Squares." In *Proceedings of the Seventh Annual SAS Users Group International Conference*, 651–653. Cary, NC: SAS Institute Inc.

Scott, D. W. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization*. New York: John Wiley & Sons.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. New York: Chapman & Hall.

Struve, O., and Zebergs, V. (1962). *Astronomy of the Twentieth Century*. New York: Macmillan.

Tukey, P. A., and Tukey, J. W. (1981). "Data-Driven View Selection: Agglomeration and Sharpening." In *Interpreting Multivariate Data*, edited by V. Barnett, 215–243. Chichester, UK: John Wiley & Sons.

Wong, M. A., and Lane, T. (1983). "A kth Nearest Neighbor Clustering Procedure." *Journal of the Royal Statistical Society, Series B* 45:362–368.

Wong, M. A., and Schaack, C. (1982). "Using the kth Nearest Neighbor Clustering Procedure to Determine the Number of Subpopulations." *Proceedings of the Statistical Computing Section, American Statistical Association* 40–48.

Chapter 86
The MULTTEST Procedure

Contents

Overview: MULTTEST Procedure **7042**
Getting Started: MULTTEST Procedure **7043**
Drug Example 7043
Syntax: MULTTEST Procedure **7047**
PROC MULTTEST Statement 7047
BY Statement 7059
CLASS Statement 7059
CONTRAST Statement 7060
FREQ Statement 7061
ID Statement 7061
STRATA Statement 7061
TEST Statement 7062
Details: MULTTEST Procedure **7064**
Statistical Tests 7064
p-Value Adjustments 7072
Missing Values 7080
Computational Resources 7080
Output Data Sets 7081
Displayed Output 7083
ODS Table Names 7084
ODS Graphics 7085
Examples: MULTTEST Procedure **7086**
Example 86.1: Cochran-Armitage Test with Permutation Resampling 7086
Example 86.2: Freeman-Tukey and *t* Tests with Bootstrap Resampling 7089
Example 86.3: Peto Mortality-Prevalence Test 7093
Example 86.4: Fisher Test with Permutation Resampling 7096
Example 86.5: Inputting Raw *p*-Values 7099
Example 86.6: Adaptive Adjustments and ODS Graphics 7102
References **7110**

Overview: MULTTEST Procedure

The MULTTEST procedure addresses the multiple testing problem. This problem arises when you perform many hypothesis tests on the same data set. Carrying out multiple tests is often reasonable because of the cost of obtaining data, the discovery of new aspects of the data, and the many alternative statistical methods. However, a disadvantage of multiple testing is the greatly increased probability of declaring false significances.

For example, suppose you carry out 10 hypothesis tests at the 5% level, and you assume that the distributions of the p-values from these tests are uniform and independent. Then, the probability of declaring a particular test significant under its null hypothesis is 0.05, but the probability of declaring at least 1 of the 10 tests significant is 0.401. If you perform 20 hypothesis tests, the latter probability increases to 0.642. These high chances illustrate the danger of multiple testing.

PROC MULTTEST approaches the multiple testing problem by adjusting the p-values from a family of hypothesis tests. An adjusted p-value is defined as the smallest significance level for which the given hypothesis would be rejected, when the entire family of tests is considered. The decision rule is to reject the null hypothesis when the adjusted p-value is less than α. For most methods, this decision rule controls the *familywise error rate* at or below the α level. However, the *false discovery rate* controlling procedures control the false discovery rate at or below the α level.

PROC MULTTEST provides the following p-value adjustments:

- Bonferroni
- Šidák
- step-down methods
- Hochberg
- Hommel
- Fisher and Stouffer combination
- bootstrap
- permutation
- adaptive methods
- false discovery rate
- positive FDR

The Bonferroni and Šidák adjustments are simple functions of the raw p-values. They are computationally quick, but they can be too conservative. Step-down methods remove some conservativeness, as do the step-up methods of Hochberg (1988), and the adaptive methods. The bootstrap and permutation adjustments resample the data with and without replacement, respectively, to approximate the distribution of the minimum p-value of all tests. This distribution is then used to adjust the individual raw p-values. The bootstrap and permutation methods are computationally intensive but appealing in that, unlike the other methods, correlations and distributional characteristics are incorporated into the adjustments (Westfall and Young 1989; Westfall et al. 1999).

PROC MULTTEST handles data arising from a multivariate one-way ANOVA model, possibly stratified, with continuous and discrete response variables; it can also accept raw p-values as input data. You can perform a t test for the mean for continuous data with or without a homogeneity assumption, and the following statistical tests for discrete data:

- Cochran-Armitage linear trend test
- Freeman-Tukey double arcsine test
- Peto mortality-prevalence (log-rank) test
- Fisher exact test

The Cochran-Armitage and Peto tests have exact versions that use permutation distributions and asymptotic versions that use an optional continuity correction. Also, with the exception of the Fisher exact test, you can use a stratification variable to construct Mantel-Haenszel-type tests. All of the previously mentioned tests can be one- or two-sided.

As in the GLM procedure, you can specify linear contrasts that compare means or proportions of the treated groups. The output contains summary statistics and regular and multiplicity-adjusted *p*-values. You can create output data sets containing raw and adjusted *p*-values, test statistics and other intermediate calculations, permutation distributions, and resampling information.

The MULTTEST procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

The GLIMMIX, GLM, MIXED, and LIFETEST procedures, and other procedures that implement the ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements, also adjust their results for multiple tests. For more information, see the documentation for these procedures and statements, and Westfall et al. (1999).

Getting Started: MULTTEST Procedure

Drug Example

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/multgs.sas.)

Suppose you conduct a small study to test the effect of a drug on 15 subjects. You randomly divide the subjects into three balanced groups receiving 0 mg, 1 mg, and 2 mg of the drug, respectively. You carry out the experiment and record the presence or absence of 10 side effects for each subject. Your data set is as follows:

```
data Drug;
   input Dose$ SideEff1-SideEff10;
   datalines;
0MG  0   0   1   0   0   1   0   0   0   0
0MG  0   0   0   0   0   0   0   0   0   1
0MG  0   0   0   0   0   0   0   0   1   0
0MG  0   0   0   0   0   0   0   0   0   0
0MG  0   1   0   0   0   0   0   0   0   0
1MG  1   0   0   1   0   1   0   0   1   0
1MG  0   0   0   1   1   0   0   1   0   1
1MG  0   1   0   0   0   0   1   0   0   0
1MG  0   0   1   0   0   0   0   0   0   1
1MG  1   0   1   0   0   0   0   1   0   0
2MG  0   1   1   1   0   1   1   1   0   1
```

```
2MG  1   1   1   1   1   1   0   1   1   0
2MG  1   0   0   1   0   1   1   0   1   0
2MG  0   1   1   1   1   0   1   1   1   1
2MG  1   0   1   0   1   1   1   0   0   1
;
```

The increasing incidence of 1s for higher dosages in the preceding data set provides an initial visual indication that the drug has an effect. To explore this statistically, you perform an analysis in which the possibility of side effects increases linearly with drug level. You can analyze the data for each side effect separately, but you are concerned that, with so many tests, there might be a high probability of incorrectly declaring some drug effects significant. You want to correct for this multiplicity problem in a way that accounts for the discreteness of the data and for the correlations between observations on the same unit.

PROC MULTTEST addresses these concerns by processing all of the data simultaneously and adjusting the *p*-values. The following statements perform a typical analysis:

```
ods graphics on;
proc multtest bootstrap nsample=20000 seed=41287 notables
              plots=PByTest(vref=0.05 0.1);
   class Dose;
   test ca(SideEff1-SideEff10);
   contrast 'Trend' 0 1 2;
run;
ods graphics off;
```

This analysis uses the BOOTSTRAP option to adjust the *p*-values. The NSAMPLE= option requests 20,000 samples for the bootstrap analysis, and the starting seed for the random number generator is 41287. The NOTABLES option suppresses the display of summary statistics for each side effect and drug level combination. The PLOTS= option displays a visual summary of the unadjusted and adjusted *p*-values against each test, and the VREF= option adds reference lines to the display.

The CLASS statement is used to specify the grouping variable, Dose. The **ca(sideeff1-sideeff10)** specification in the TEST statement requests a Cochran-Armitage linear trend test for all 10 characteristics. The CONTRAST statement gives the coefficients for the linear trend test.

The "Model Information" table in Figure 86.1 describes the statistical tests performed by PROC MULTTEST. For this example, PROC MULTTEST carries out a two-tailed Cochran-Armitage linear trend test with no continuity correction or strata adjustment. This test is performed on the raw data and on 20,000 bootstrap samples.

Figure 86.1 Output Summary for the MULTTEST Procedure

The Multtest Procedure

Model Information	
Test for discrete variables	Cochran-Armitage
Z-score approximation used	Everywhere
Continuity correction	0
Tails for discrete tests	Two-tailed
Strata weights	None
P-value adjustment	Bootstrap
Number of resamples	20000
Seed	41287

The "Contrast Coefficients" table in Figure 86.2 displays the coefficients for the Cochran-Armitage test. They are 0, 1, and 2, as specified in the CONTRAST statement.

Figure 86.2 Coefficients Used in the MULTTEST Procedure

Contrast Coefficients			
	Dose		
Contrast	0MG	1MG	2MG
Trend	0	1	2

The "p-Values" table in Figure 86.3 lists the *p*-values for the drug example. The Raw column lists the *p*-values for the Cochran-Armitage test on the original data, and the Bootstrap column provides the bootstrap adjustment of the raw *p*-values.

Note that the raw *p*-values lead you to reject the null hypothesis of no linear trend for 3 of the 10 characteristics at the 5% level and 7 of the 10 characteristics at the 10% level. The bootstrap *p*-values, however, lead to this conclusion for 0 of the 10 characteristics at the 5% level and only 2 of the 10 characteristics at the 10% level; you can also see this in Figure 86.4.

Figure 86.3 Summary of *p*-Values for the MULTTEST Procedure

p-Values			
Variable	Contrast	Raw	Bootstrap
SideEff1	Trend	0.0519	0.3388
SideEff2	Trend	0.1949	0.8403
SideEff3	Trend	0.0662	0.5190
SideEff4	Trend	0.0126	0.0884
SideEff5	Trend	0.0382	0.2408
SideEff6	Trend	0.0614	0.4383
SideEff7	Trend	0.0095	0.0514
SideEff8	Trend	0.0519	0.3388
SideEff9	Trend	0.1949	0.8403
SideEff10	Trend	0.2123	0.9030

Figure 86.4 Adjusted p-Values

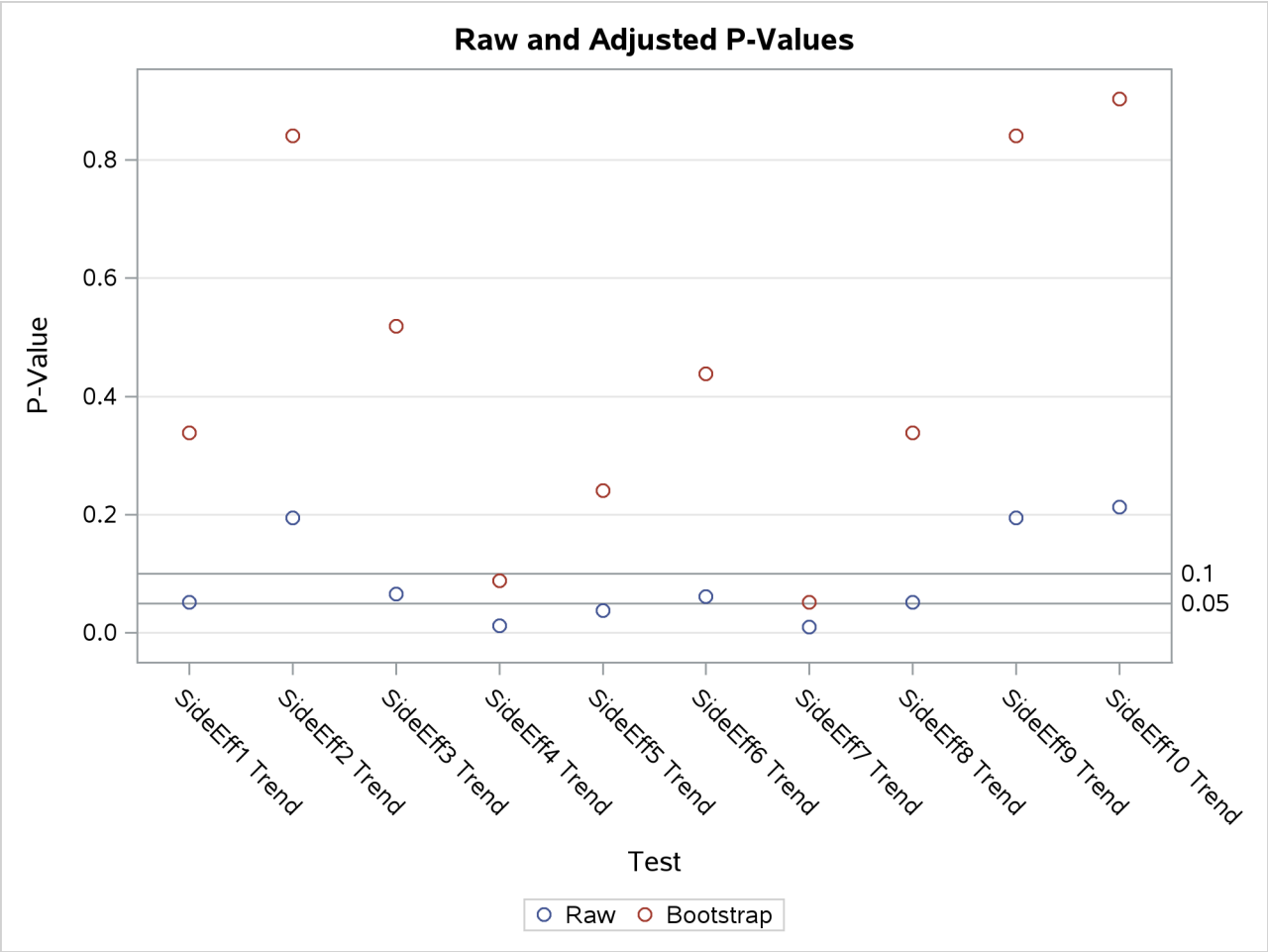

The bootstrap adjustment gives the probability of observing a p-value as extreme as each given p-value, considering all 10 tests simultaneously. This adjustment incorporates the correlation of the raw p-values, the discreteness of the data, and the multiple testing problem. Failure to account for these issues can certainly lead to misleading inferences for these data.

Syntax: MULTTEST Procedure

The following statements are available in the MULTTEST procedure:

```
PROC MULTTEST <options> ;
    BY variables ;
    CLASS variable ;
    CONTRAST 'label' values ;
    FREQ variable ;
    ID variables ;
    STRATA variable ;
    TEST name (variables < / options>) ;
```

Statements that follow the PROC MULTTEST statement can appear in any order. The CLASS and TEST statements are required unless the INPVALUES= option is specified in the PROC MULTTEST statement.

The following sections describe the PROC MULTTEST statement and then describe the other statements in alphabetical order.

PROC MULTTEST Statement

```
PROC MULTTEST <options> ;
```

The PROC MULTTEST statement invokes the MULTTEST procedure. It also specifies the p-value adjustments. Table 86.1 summarizes the *options* available in the PROC MULTTEST statement. These *options* are described in alphabetical order following the table.

Table 86.1 PROC MULTTEST Statement Options by Function

Option	Description
FWE-Controlling *p*-Value Adjustments	
ADAPTIVEHOLM	Computes the adaptive step-down Bonferroni adjustment
ADAPTIVEHOCHBERG	Computes the adaptive step-up Bonferroni adjustment
BONFERRONI	Computes the Bonferroni adjustment
BOOTSTRAP	Computes the bootstrap min-p adjustment
FISHER_C	Computes Fisher's combination adjustment
HOCHBERG	Computes the step-up Bonferroni adjustment
HOMMEL	Computes Hommel's adjustment
HOLM	Computes the step-down Bonferroni adjustment
PERMUTATION	Computes the permutation min-p adjustment
SIDAK	Computes Šidák's adjustment
STEPBON	Computes the step-down Bonferroni adjustment
STEPBOOT	Computes the step-down bootstrap adjustment
STEPPERM	Computes the step-down permutation adjustment
STEPSID	Computes the step-down Šidák adjustment
STOUFFER	Computes the Stouffer-Liptak combination adjustment

Table 86.1 *continued*

Option	Description
FDR-Controlling *p*-Value Adjustments	
ADAPTIVEFDR	Computes the adaptive linear step-up adjustment
DEPENDENTFDR	Computes the linear step-up adjustment under dependence
FDR	Computes the linear step-up adjustment
FDRBOOT	Computes the linear step-up bootstrap min-*p* adjustment
FDRPERM	Computes the linear step-up permutation min-*p* adjustment
PFDR	Computes the positive FDR adjustment
Input/Output Data Sets	
DATA=	Names the input data set
INPVALUES=	Names the input data set of raw *p*-values
OUT=	Names the output data set
OUTPERM=	Names the output permutation data set
OUTSAMP=	Names the output resample data set
Displayed Output Options	
NOPRINT	Suppresses all tables
NOTABLES	Suppresses variable tables
NOZEROS	Suppresses zero tables for CLASS variables
NOPVALUE	Suppresses the "p-Values" table
PLOTS	Requests ODS Graphics
Resampling Options	
CENTER	Mean-centers continuous variables before resampling
NOCENTER	Does not mean-center continuous variables before resampling
NSAMPLE=	Specifies the number of resamples
RANUNI	Specifies a different random number generator
SEED=	Specifies the seed for resampling
CLASS Variable Options	
NOZEROS	Suppresses zero tables for CLASS variables
ORDER=	Specifies CLASS variable order
Computational Options	
EPSILON=	Specifies the comparison value
NTRUENULL=	Specifies the estimation method for the number of true nulls
PTRUENULL=	Specifies the estimation method for the proportion of true nulls

You can specify the following *options* in the PROC MULTTEST statement.

ADAPTIVEHOCHBERG

AHOC

requests adjusted *p*-values by using the Hochberg and Benjamini (1990) adaptive step-up Bonferroni method. See the section "Adaptive Adjustments" on page 7076 for more details.

ADAPTIVEHOLM

AHOLM

requests adjusted p-values by using the Hochberg and Benjamini (1990) adaptive step-down Bonferroni method. See the section "Adaptive Adjustments" on page 7076 for more details.

ADAPTIVEFDR< (UNRESTRICT) >

AFDR< (UNRESTRICT) >

requests adjusted p-values by using the Benjamini and Hochberg (2000) adaptive linear step-up method (AFDR). The UNRESTRICT option estimates the AFDR as defined in Benjamini and Hochberg (2000), which allows the adjustment to reduce the raw p-value. By default, the AFDR is constrained to be greater than or equal to the raw p-value. See the section "Adaptive False Discovery Rate" on page 7078 for more details.

BONFERRONI

BON

specifies that the Bonferroni adjustments (number of tests $\times$ p-value) be computed for each test. These adjustments can be extremely conservative and should be viewed with caution. When exact tests are specified via the PERMUTATION= option in the TEST statement, the actual permutation distributions are used, resulting in a much less conservative version of this procedure (Westfall and Wolfinger 1997). See the section "Bonferroni" on page 7073 for more details.

BOOTSTRAP

BOOT

specifies that the p-values be adjusted by using the bootstrap method to resample vectors (Westfall and Young 1993). Resampling is performed with replacement and independently within levels of the STRATA variable. Continuous variables are mean-centered by default prior to resampling; specify the NOCENTER option to change this. See the section "Bootstrap" on page 7074 for more details. The BOOTSTRAP option is not allowed with the Peto test.

If the PERMUTATION= suboption is used with the CA test in the TEST statement, the exact permutation distribution is recomputed for each bootstrap sample. **CAUTION:** This can be very time-consuming. It is preferable to use permutation resampling when permutation base tests are used.

CENTER

requests that continuous variables be mean-centered prior to resampling. The default action is to mean-center for bootstrap resampling and not to mean-center for permutation resampling.

DATA=*SAS-data-set*

names the input SAS data set to be used by PROC MULTTEST. The default is to use the most recently created data set. The DATA= and INPVALUES= options cannot both be specified.

DEPENDENTFDR

DFDR

requests adjusted p-values by using the method of Benjamini and Yekutieli (2001). See the section "Dependent False Discovery Rate" on page 7077 for more details.

EPSILON=*number*

specifies the amount by which two p-values must differ to be declared unequal. The value *number* must be between 0 and 1; the default value is 1000 times the machine epsilon, which is approximately 1E–12. See Westfall and Young (1993, pp. 165–166) for more information.

FDR

LSU

requests adjusted p-values by using the linear step-up method of Benjamini and Hochberg (1995). These p-values do not control the familywise error rate, but they do control the false discovery rate in some cases. See the section "False Discovery Rate Controlling Adjustments" on page 7077 for more details.

FDRBOOT<(*β*)>

A bootstrap-resampling false discovery rate controlling method due to Yekutieli and Benjamini (1999). This method uses the same resampling algorithm as the BOOTSTRAP option. Every resample is saved in order to compute a quantile of the resampled p-values; therefore, this method can use a lot of memory. The parameter β designates that a $100(1-\beta)$ quantile is used in the computations for determining the adjustments; by default, $\beta = 0.05$. See the section "False Discovery Rate Resampling Adjustments" on page 7078 for details.

FDRPERM<(*β*)>

A permutation-resampling false discovery rate controlling method due to Yekutieli and Benjamini (1999). This method uses the same resampling algorithm as the PERMUTATION option. Every resample is saved in order to compute a quantile of the resampled p-values; therefore, this method can use a lot of memory. The parameter β designates that a $100(1-\beta)$ quantile is used in the computations for determining the adjustments; by default, $\beta = 0.05$. See the section "False Discovery Rate Resampling Adjustments" on page 7078 for details.

FISHER_C

FIC

requests adjusted p-values by using Fisher's combination method. See the section "Fisher Combination" on page 7076 for more details.

HOCHBERG

HOC

requests adjusted p-values by using the step-up Bonferroni method due to Hochberg (1988). See the section "Hochberg" on page 7075 for more details.

HOMMEL

HOM

requests adjusted p-values by using the method of Hommel (1988). See the section "Hommel" on page 7075 for more details.

HOLM

is an alias for the STEPBON adjustment.

INPVALUES< (*pvalue-name*) >=*SAS-data-set*

names an input SAS data set that includes a variable containing raw p-values. The MULTTEST procedure adjusts the collection of raw p-values for multiplicity. Resampling-based adjustments are not permitted with this type of data input. The CLASS, CONTRAST, FREQ, STRATA, and TEST statements are ignored when an INPVALUES= data set is specified. The INPVALUES= and DATA= options cannot both be specified. The *pvalue-name* enables you to specify the name of the p-value column from your data set. By default, *pvalue-name*='raw_p'. The INPVALUES= data set can contain variables in addition to the raw p-values variable; see Example 86.5 for an example.

LIPTAK

is an alias for the STOUFFER adjustment.

NOCENTER

requests that continuous variables not be mean-centered prior to resampling. The default action is to mean-center for bootstrap resampling and not to mean-center for permutation resampling.

NOPRINT

suppresses the normal display of results. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

NOPVALUE

suppresses the display of the "p-Values" table of raw and adjusted p-values. This option is most useful when you are adjusting many tests and need to create only an OUT= data set or display graphics.

NOTABLES

suppresses display of the "Discrete Variable Tabulations" and "Continuous Variable Tabulations" tables.

NOZEROS

suppresses display of tables having zero occurrences for all CLASS levels.

NSAMPLE=*number*

N=*number*

specifies the number of resamples for use with the resampling methods. The value *number* must be a positive integer; by default, 20,000 resamples are used. Large values of *number* (20,000 or more) are usually recommended for accuracy, but long execution times can result, particularly with large data sets.

NTRUENULL=*keyword* | *value*

M0=*keyword* | *value*

Controls the method used to estimate the number of true NULL hypotheses (m_0) for the adaptive methods. This option is ignored unless one of the adaptive methods is specified. By default, PROC MULTTEST uses the DECREASESLOPE method for the ADAPTIVEHOLM and ADAPTIVEHOCHBERG adjustments, and the LOWESTSLOPE method for ADAPTIVEFDR adjustment. For the PFDR adjustment, the SPLINE method is attempted first. If the estimate is nonpositive or if the slope of the spline at the last λ is greater than 0.1 times the range of the fitted spline values, then the BOOTSTRAP method is used.

You can specify a positive integer as the *value*, or you can specify one of the *keywords* in the following list. Alternatively, you can specify the proportion of true NULL hypotheses by using the PTRUENULL= option. Suppose you have m tests with ordered p-values $p_{(1)} \leq \ldots \leq p_{(m)}$, and define $q_{(i)} = 1 - p_{(i)}$.

BOOTSTRAP< (*bootstrap-options*) >

uses the bootstrap method of Storey and Tibshirani (2003). Compute the proportion of true null hypotheses $\hat{\pi}_0(\lambda) = \frac{m-N(\lambda)+f}{(1-\lambda)m}$ for $\lambda \in L = \{0, 0.05, \ldots, 0.95\}$, where $N(\lambda)$ is the number of p-values less than or equal to λ, and $f = 1$ for the finite-sample case; otherwise $f = 0$. For each λ, bootstrap on the p-values to form B bootstrap versions $\hat{\pi}_0^b(\lambda), b = 1, \ldots, B$, and choose the λ that yields the minimum $\widehat{\mathrm{MSE}}(\lambda) = \frac{1}{B}\sum_{b=1}^{B}(\hat{\pi}_0^b(\lambda) - \min_{\lambda' \in L}\hat{\pi}_0(\lambda'))^2$. The available *bootstrap-options* are as follows:

FINITE

the finite-sample case of the PFDR option, described on page 7054.

NBOOT=*B*

bootstrap resamples of the raw p-values for the λ computations. NBOOT=10,000 by default; *B* must be a positive integer.

NLAMBDA=*n*

"optimal" λ is the value in $\{0, \frac{1}{n}, \ldots, \frac{n-1}{n}\}$ that minimizes the MSE. NLAMBD=20 by default; *n* must be an integer greater than 1.

DECREASESLOPE

Schweder and Spjøtvoll (1982) as modified by Hochberg and Benjamini (1990). Let b_i be the slope of the least squares line fit to $\{q_{(m)}, \ldots, q_{(m-i+1)}\}$ and through the origin, for $i = 1, \ldots, m$. Find the first $i = m-1, m-2, \ldots, 1$ such that $b_i < b_{i+1}$. Then $\hat{m}_0 = \mathrm{ceil}(\frac{1}{b_{i+1}} - 1)$.

KSTEST< (β) >

uses the Kolmogorov-Smirnov uniformity test method of Turkheimer, Smith, and Schmidt (2001). Let $k_{min} = 1, k_{max} = m$, and the Kolmogorov-Smirnov statistic $D = \max(q_{(i)} - i/(m+1)(\sqrt{k} + 0.12 + 0.11/\sqrt{k}))$. If D is greater than the upper-tail probability (Press et al. 1992), then $k_{max} = k, k = \mathrm{floor}((k_{min} + k)/2)$; otherwise, let $k_{min} = k, k = \mathrm{floor}((k + k_{max})/2)$. Repeat until $k = k_{min}$. Next compute the slope b of the weighted least squares regression line on the k smallest $q_{(i)}$ by using weights $w_i = i(k-i+1)/((k+1)^2(k+2))$. Then $\hat{m}_0 = \mathrm{ceil}(\frac{1}{b} - 1)$.

LEASTSQUARES

uses a linear least squares method to search for the correct cutpoint. For each $i = 0, \ldots, m$ compute the SSE of the least squares line through the origin fitting $\{q_{(m)}, \ldots, q_{(m-i+1)}\}$, let b_i be the slope of this line, and add the SSE of the unconstrained least squares line through the rest of the qs. For $i = 0$ compute the SSE for the unconstrained line. The argument i that minimizes the SSE is the cutpoint: if $i = 0$ then $\hat{m}_0 = 0$; if $i = m$ then $\hat{m}_0 = m$; otherwise $\hat{m}_0 = \mathrm{ceil}(\frac{1}{b_i} - 1)$.

LOWESTSLOPE

uses the lowest slope method of Benjamini and Hochberg (2000). Find the first $i = 1, \ldots, m$ such that $b_i = q_{(i)}/(m-i+1)$ decreases. Then $\hat{m}_0 = \mathrm{floor}(\min(\frac{1}{b_i} + 1, m))$.

MEANDIFF

uses the mean of differences method of Hsueh, Chen, and Kodell (2003). Let $\bar{d}_i = \frac{q_{(m-i+1)}}{i}$ and estimate $\hat{m}_0^i = \frac{1}{\bar{d}_i} - 1$. Start from $i = m$ and proceed downward until the first time $\hat{m}_0^{i-1} \geq \hat{m}_0^i$ occurs.

SPLINE< (*spline-options*) >

uses the cubic spline method of Storey and Tibshirani (2003). For each $\lambda \in \{0, \frac{1}{n}, \frac{2}{n}, \ldots, \frac{n-1}{n}\}$ compute $\hat{\pi}_0(\lambda) = \frac{\#\{p_i > \lambda\}}{m(1-\lambda)}$. Let $\widehat{f}(\lambda)$ be the natural cubic spline with 3 degrees of freedom of $\hat{\pi}_0(\lambda)$ versus λ. Estimate $\hat{\pi}_0$ by taking the spline value at the last λ: $\hat{\pi}_0 = \hat{\pi}_0(\frac{n-1}{n})$, so that $\hat{m}_0 = m\hat{\pi}_0$. The available *spline-options* are as follows:

DF=*df*

sets the degrees of freedom of the spline, where *df* is a nonnegative integer. The default is DF=3.

DFCONV=*number*

specifies the absolute change in spline degrees of freedom value for concluding convergence. If $|df_i - df_{i+1}| <$ *number* (or if the SPCONV= criterion is satisfied), then convergence is declared. *number* must be between 0 and 1; by default, *number* is 1000 times the square root of machine epsilon, which is about 1E–5.

FINITE

computations for the finite-sample case of the PFDR option, described on page 7054.

MAXITER=*n*

specifies the maximum number of golden-search iterations used to find a spline with DF=df degrees of freedom. By default, MAXITER=100; *number* must be a nonnegative integer.

NLAMBDA=*n*

$\hat{\pi}_0(\lambda)$ for $\lambda \in \{0, \frac{1}{n}, \frac{2}{n}, \ldots, \frac{n-1}{n}\}$ for the spline fit. By default, NLAMBDA=20; *number* must be an integer greater than 1.

SPCONV=*number*

specifies the absolute change in smoothing parameter value for concluding convergence of the spline. If $|sp_i - sp_{i+1}| <$ *number* (or if the DFCONV= criterion is satisfied), then convergence is declared. By default, *number* equals the square root of the machine epsilon, which is about 1E–8.

In all cases $\hat{m}_0$ is constrained to lie between 0 and m; if the computed $\hat{m}_0 = 0$, then the adaptive adjustments do not produce results. If you specify $\hat{m}_0 > m$, then it is reduced to m. Values of $\hat{m}_0$ are displayed in the "Estimated Number of True Null Hypotheses" table.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement). This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUT=*SAS-data-set*

names the output SAS data set containing variable names, contrast names, intermediate calculations, and all associated p-values. See "OUT= Data Set" on page 7081 for more information.

OUTPERM=*SAS-data-set*

names the output SAS data set containing entire permutation distributions (upper-tail probabilities) for all tests when the PERMUTATION= option is specified. See "OUTPERM= Data Set" on page 7082 for more information. **CAUTION:** This data set can be very large.

OUTSAMP=*SAS-data-set*

names the output SAS data set containing information from the resampled data sets when resampling is performed. See "OUTSAMP= Data Set" on page 7082 for more information. **CAUTION:** This data set can be very large.

PDATA=*SAS-data-set*

is an alias for the INPVALUES= option.

PERMUTATION

PERM

computes adjusted p-values in identical fashion as the BOOTSTRAP option, with the exception that PROC MULTTEST resamples without replacement rather than with replacement. Resampling is performed independently within levels of the STRATA variable. Continuous variables are not mean-centered prior to resampling; specify the CENTER to change this. See the section "Bootstrap" on page 7074 for more details. The PERMUTATION option is not allowed with the Peto test.

PFDR<(*options***)>**

computes the "q-values" $\hat{q}_\lambda(p_i)$ of Storey (2002) and Storey, Taylor, and Siegmund (2004). PROC MULTTEST treats these "q-values" as adjusted p-values. The computations depend on selecting a parameter λ and an estimation method for the false discovery rate; see the section "Positive False Discovery Rate" on page 7079 for computational details. The available *options* for choosing the method are as follows:

FINITE
estimates the false discovery rate with $\widehat{\text{pFDR}}$ or $\widehat{\text{FDR}}$ for the finite-sample case with independent null p-values.

POSITIVE
estimates the false discovery rate with $\widehat{\text{pFDR}}$ instead of the default $\widehat{\text{FDR}}$.

UNRESTRICT
estimates the false discovery rate as defined in Storey (2002), which allows the adjustment to reduce the raw p-value. By default, the PFDR is constrained to be greater than or equal to the raw p-value.

The available *options* for controlling the λ search are the *bootstrap-options* (page 7052), the *spline-options* (page 7053), and the following *options*:

LAMBDA=*number*
specifies a $\lambda \in [0, 1)$ and does not perform the bootstrap or spline searches for an "optimal" λ.

MAXLAMBDA=*number*
stops the NLAMBDA= search sequence for the bootstrap and spline searches when this *number* is reached. The *number* must be in $[0, 1]$. This option is ignored if the LAMBDA= option is specified.

PLOTS<(*global-plot-options***)>=***plot-request***<(***options***)>**
PLOTS<(*global-plot-options***)>=(***plot-request***<(***options***)>< . . . *plot-request***<(***options***)>>)**
controls the plots produced through ODS Graphics. If you specify only one *plot-request*, you can omit the parentheses. For example, the following statements are valid specifications of the PLOTS= option:

```
plots = all
plots = (rawprob adjusted)
plots(sigonly) = (rawprob adjusted(unpack))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc multtest plots=adjusted inpvalues=a pfdr;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

By default, no graphs are created; you must specify the PLOTS= option to make graphs. You need at least two tests to produce a graph. If you are not using an INPVALUES= data set, then each test is given a name constructed as "variable-name contrast-label". If you specify a MEAN test in the TEST statement, the t-test names are prefixed with "Mean:". See Example 86.6 for examples of the ODS graphical displays.

The following *global-plot-options* are available:

UNPACKPANELS | UNPACK

suppresses paneling. By default, the plots produced with the ADJUSTED and RAWPROB options are grouped in a single display, called a *panel*. Specify UNPACK to display each plot separately.

SIGONLY< =*number* >

displays only those tests with adjusted p-values $\leq$ *number*, where $0 \leq$ *number* ≤ 1. By default, *number* = 0.05.

The following *plot-requests* are available:

ADJUSTED< (UNPACK) >

displays a 2×2 panel of adjusted p-value plots similar to those Storey and Tibshirani (2003) developed for use with the PFDR p-value adjustment method. The plots of the adjusted p-values by the raw p-values and the adjusted p-values by their rank show the effect of the adjustments. The plot of the proportion of adjusted p-values $\leq$ each adjusted p-value and the plot of the expected number of false positives (the proportion significant multiplied by the adjusted p-value) versus the proportion significant show the effect of choosing different significance levels. The UNPACK option unpanels the display.

ALL

produces all appropriate plots. You can specify other options with ALL; for example, to display all plots and unpack the RAWPROB plots you can specify **`plots=(all rawprob(unpack))`**.

LAMBDA

displays plots of the MSE and the estimated number of true nulls against the λ parameter when the NTRUENULL=SPLINE or NTRUENULL=BOOTSTRAP option is in effect.

MANHATTAN< (*options*) >

displays the Manhattan plot (a plot of $-\log_{10}$ of the adjusted p-values versus the tests). You can specify the following *options*:

GROUP=*variable*

specifies a variable to group the adjusted p-values in the display.

LABEL < =OBS >

labels the observations that have adjusted p-values that are less than the value specified in the VREF= option. By default, labels are created as follows: if an INPVALUES= data set and an ID statement are specified, then the observations are labeled with the ID values; if a DATA= data set is specified, then the observations are labeled with their constructed test name; otherwise, the observation or test number is displayed.

NOTESTNAME

displays the number of the test instead of the test name on the X-axis, which is useful when you have many tests.

UNPACK
suppresses paneling. By default, Manhattan plots are created for each requested p-value adjustment, and the results are grouped in a single display, called a *panel*. Specify UNPACK to display each plot separately.

VREF=*number* | **NONE**
displays a reference line at $-\log_{10}(\textit{number})$. The *number* must be between 0 and 1. By default, a reference line at $-\log_{10}(0.05)$ is displayed; it can be suppressed by specifying VREF=0 or VREF=NONE. If the LABEL option is also specified, then observations above this line are labeled with their ID variables, their observation number, their test name, or their test number.

NONE
suppresses all plots.

PBYTEST< (*options***) >**
displays the adjusted p-values for each test. The available options are as follows:

NOTESTNAME
displays the number of the test instead of the test name on the axis, which is useful when you have many tests.

VREF=
number-list displays reference lines at the p-values specified in the *number-list*. The values in the *number-list* must be between 0 and 1; otherwise they are ignored. You can specify a single value or a list of values; for example, `vref=0.1 0 to 0.05 by 0.01` displays reference lines at each of the values {0.01, 0.02, 0.03, 0.04, 0.05, and 0.1}.

RAWPROB< (UNPACK) >
displays a uniform probability plot of 1 minus the raw p-values (Schweder and Spjøtvoll 1982) along with a histogram. If m_0 is the number of true null hypotheses among the m tests, the points on the left side of the plot should be approximately linear with slope $\frac{1}{m_0+1}$. This graphic is displayed when an adaptive p-value adjustment method is requested in order to see if the NTRUENULL= estimate is appropriate. The UNPACK option unpanels the display.

PTRUENULL=*keyword* | *value*

PI0=*keyword* | *value*
is alias for the NTRUENULL= option, except that you can specify the proportion of true null hypotheses as a *value* between 0 and 1, instead of specifying the number of true null hypotheses. The available *keywords* are also the NTRUENULL= options described on page 7051.

RANUNI
uses an old random number generating algorithm. By default, the random number generator is the Mersenne Twister, which has better performance when bootstrapping.

SEED=*number*

S=*number*

specifies the initial seed for the random number generator used for resampling. The value for *number* must be an integer. If you do not specify a seed, or if you specify a value less than or equal to zero, then PROC MULTTEST uses the time of day from the computer's clock to generate an initial seed. For more details about seed values, see *SAS Programmers Guide: Essentials*.

SIDAK

SID

computes the Šidák adjustment for each test. These adjustments take the form

$$1 - (1 - p)^m$$

where p is the raw p-value and m is the number of tests. These are slightly less conservative than the Bonferroni adjustments, but they still should be viewed with caution. When exact tests are specified via the PERMUTATION= option in the TEST statement, the actual permutation distributions are used, resulting in a much less conservative version of this procedure (Westfall and Wolfinger 1997). See the section "Šidák" on page 7073 for more details.

STEPBON

HOLM

requests adjusted p-values by using the step-down Bonferroni method of Holm (1979). See the section "Step-Down Methods" on page 7074 for more details.

STEPBOOT

requests that adjusted p-values be computed by using bootstrap resampling as described under the BOOTSTRAP option, but in step-down fashion. See the section "Step-Down Methods" on page 7074 for more details.

STEPPERM

requests that adjusted p-values be computed by using permutation resampling as described under the PERMUTATION option, but in step-down fashion. See the section "Step-Down Methods" on page 7074 for more details.

STEPSID

requests adjusted p-values by using the Šidák method as described in the SIDAK option, but in step-down fashion. See the section "Step-Down Methods" on page 7074 for more details.

STOUFFER

LIPTAK

requests adjusted p-values by using the Stouffer-Liptak combination method. See the section "Stouffer-Liptak Combination" on page 7076 for more details.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC MULTTEST to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the MULTTEST procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

You can specify one or more *variables* in the input data set on the BY statement.

Since sorting the data changes the order in which PROC MULTTEST reads observations, this can affect the sort order for the levels of the CLASS variable if you have specified ORDER=DATA in the PROC MULTTEST statement. This, in turn, affects specifications in the CONTRAST statements.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < / **TRUNCATE** > **;**

The CLASS statement is required unless the INPVALUES= option is specified. The CLASS statement specifies a single variable (character or numeric) used to identify the groups for the analysis. For example, if the variable Treatment defines different levels of a treatment that you want to compare, then you would specify the following statements:

```
class Treatment;
```

The CLASS variable can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variable. The order of the class levels used by PROC MULTTEST corresponds to the order of their formatted values; this order can be changed with the ORDER= option in the PROC MULTTEST statement.

You can specify the TRUNCATE option to determine class levels by using no more than the first 16 characters of the formatted values.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS*

Formats and Informats: Reference. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC MULTTEST statement. You need to be aware of the order when using the CONTRAST statement, and you should check the "Contrast Coefficients" table to verify that it is suitable.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

CONTRAST Statement

CONTRAST '*label*' *values* **;**

This statement is used to identify tests between the levels of the CLASS variable; in particular, it is used to specify the coefficients for the trend tests. The *label* is a string naming the contrast; it contains a maximum of 21 characters. The *values* are scoring coefficients across the CLASS variable levels.

You can specify multiple CONTRAST statements, thereby specifying multiple contrasts for each variable. Multiplicity adjustments are computed for all contrasts and all variables simultaneously. The coefficients are applied to the ordered CLASS variables; this order can be changed with the ORDER= option in the PROC MULTTEST statement. For example, consider a four-group experiment with CLASS variable levels A1, A2, B1, and B2 denoting two levels of two treatments. The following statements produce three linear trend tests for each variable identified in the TEST statement. PROC MULTTEST computes the multiplicity adjustments over the entire collection of tests, which is three times the number of variables.

```
contrast 'a vs b'    -1 -1  1  1;
contrast 'a linear'  -1  1  0  0;
contrast 'b linear'   0  0 -1  1;
```

As another example, consider an animal carcinogenicity experiment with dose levels 0, 4, 8, 16, and 50. You can specify a trend test with the indicated scoring coefficients by using the following statement:

```
contrast 'arithmetic trend' 0 4 8 16 50;
```

Multiplicity-adjusted *p*-values are then computed over the collection of variables identified in the TEST statement. See Lagakos and Louis (1985) for guidelines on the selection of contrast-scoring values.

When a Fisher test is specified in the TEST statement, the CONTRAST statement coefficients are used to group the CLASS variable's levels. Groups with a –1 contrast coefficient are combined and compared with groups with a 1 contrast coefficient for each test, and groups with a 0 coefficient are not included in the contrast. For example, the following statements compute Fisher exact tests for (a) control versus the combined treatment groups, (b) control versus the first treatment group, and (c) control versus the third treatment group:

```
contrast 'c vs all' 1 -1 -1 -1;
contrast 'c vs t1'  1 -1  0  0;
contrast 'c vs t3'  1  0  0 -1;
```

Multiplicity adjustments are then computed over the entire collection of tests and variables. Only –1, 1, and 0 are acceptable CONTRAST coefficients when the Fisher test is specified; PROC MULTTEST ignores the CONTRAST statement if any other coefficients appear.

If you specify the FISHER test and no CONTRAST statements, then all contrasts of control versus treatment are automatically generated, with the first level of the CLASS variable deemed to be the control. In this case, the control level is assigned the value 1 in each contrast and the other treatment levels are assigned –1. You should therefore use the LOWERTAILED option to test for higher success rates in the treatment groups.

For tests other than FISHER, CONTRAST values are $0, 1, 2, \ldots$ by default. If you specify the CA or PETO test with the PERMUTATION= option, then your CONTRAST coefficients must be integer valued.

For *t* tests for the mean of continuous data (and for the FT tests), the contrast coefficients are centered to have mean $= 0$. The resulting centered scoring coefficients are then applied to the sample means (or to the double-arcsine-transformed proportions in the case of the FT tests).

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a variable that provides frequencies for each observation in the DATA= data set. Specifically, if n is the value of the FREQ variable for a given observation, then that observation is used n times.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

ID Statement

ID *variables* **;**

The ID statement names one or more variables for identifying observations in the output and in the plots. The statement requires an INPVALUES= data set. All ID variables are displayed in the "pValues" table. The ID variables are used as the X axis for the plots requested by the PLOTS=PBYTEST and PLOTS=MANHATTAN options in the PROC MULTTEST statement; they are also used to label points on the Manhattan plots. This option has no effect on the OUT= data set.

STRATA Statement

STRATA *variable* **;**

The STRATA statement identifies a single variable to use as a stratification variable in the analysis. This yields tests similar to those discussed in Mantel and Haenszel (1959) and Hoel and Walburg (1972) for binary data and pooled-means tests for continuous data. For example, when you test for prevalence in a carcinogenicity study, it is common to stratify on intervals of the time of death; the first level of the stratification variable might represent weeks 0–52, the second might represent weeks 53–80, and so on. In multicenter clinical studies, each level of the stratification variable might represent a particular center.

The following option is available in the STRATA statement after a slash (/):

WEIGHT=*keyword*

specifies the type of strata weighting to use when computing the Freeman-Tukey and *t* tests. Valid *keywords* are SAMPLESIZE, HARMONIC, and EQUAL. SAMPLESIZE requests weights proportional to the within-stratum sample sizes, and is the default method even if the WEIGHT= option is not specified. HARMONIC sets up weights equal to the harmonic mean of the nonmissing within-stratum CLASS sizes, and is similar to a Type 2 analysis in PROC GLM. EQUAL specifies equal weights, and is similar to a Type 3 analysis in PROC GLM.

TEST Statement

TEST *name* **(***variables* < / *options* >**) ;**

The TEST statement is required unless the INPVALUES= option is specified. The TEST statement identifies statistical tests to be performed and the discrete and continuous variables to be tested. Table 86.2 summarizes the names and options available in the TEST statement.

Table 86.2 TEST Statement Names and Options

Option	Description
TEST Names	
CA	Requests the Cochran-Armitage linear trend tests for group comparisons
FISHER	Requests Fisher exact tests
FT	Requests *Z*-score CA tests based upon the Freeman-Tukey double arcsine transformation
MEAN	Requests the *t* test for the mean
PETO	Requests the Peto mortality-prevalence test
TEST Options	
BINOMIAL	Uses the binomial variance estimate for CA and Peto tests
CONTINUITY=	Specifies a continuity correction
DDFM=	Specifies whether to use homogeneous or heterogeneous variances
LOWERTAILED	Makes all tests lower-tailed
PERMUTATION=	Computes *p*-values for the CA and Peto tests by using exact permutation distributions
TIME=	Identifies the Peto test variable containing the age at death
UPPERTAILED	Makes all tests upper-tailed

The following tests are permitted as *name* in the TEST statement.

CA

requests the Cochran-Armitage linear trend tests for group comparisons. The test variables should take the value 0 for a failure and 1 for a success. PERMUTATION= option can be used to request an exact permutation test; otherwise, a *Z*-score approximation is used. The CONTINUITY= option can be used to specify a continuity correction for the *Z*-score approximation.

FISHER

requests Fisher exact tests for comparing two treatment groups. The test variables should take the value 0 for a failure and 1 for a success.

FT

requests Z-score CA tests based upon the Freeman-Tukey double arcsine transformation of the frequencies. The test variables should take the value 0 for a failure and 1 for a success.

MEAN

requests the t test for the mean. The test variables can take on any numeric values.

PETO

requests the Peto mortality-prevalence test. The test variables should take the value 0 for a nonoccurrence, 1 for an incidental occurrence, and 2 for a fatal occurrence. The TIME= option should be used with the Peto test to specify an integer-valued variable giving the age at death. The CONTINUITY= option can be used to specify a continuity correction for the test.

If the value of a TEST variable is invalid, the observation is not used in the analysis. You can specify two tests only if one of them is MEAN. For example, the following statement is valid:

```
test ca(d1-d2) mean(c1-c2);
```

But specifying both CA and FT, as shown in the following statement, is invalid:

```
test ca(d1-d2) ft(d1-d2);
```

You can specify the following *options* in the TEST statement (some apply to only one test).

BINOMIAL

uses the binomial variance estimate for CA and Peto tests in their asymptotic normal approximations. The default is to use the hypergeometric variance.

CONTINUITY=*number*

C=*number*

specifies *number* as a particular continuity correction for the Z-score approximation in the CA and Peto tests. The default is 0.

LOWERTAILED

LOWER

is used to make all tests lower-tailed. All tests are two-tailed by default.

PERMUTATION=*number*

PERM=*number*

computes p-values for the CA and Peto tests by using exact permutation distributions when marginal success or failure totals within a stratum are *number* or less. You can specify *number* as a nonnegative integer. For totals greater than *number* (or when the PERMUTATION= option is omitted), PROC MULTTEST uses standard normal approximations with a continuity correction chosen to approximate the permutation distribution. PROC MULTTEST computes the appropriate convolution distributions when you use the STRATA statement along with the PERMUTATION= option.

DDFM= POOLED | SATTERTHWAITE
specifies whether the MEAN test uses a homogeneity assumption (DDFM=POOLED, the default) or deals with heterogeneous variances (DDFM=SATTERTHWAITE). See "*t* Test for the Mean" on page 7070 for more information.

TIME=*variable*
identifies the Peto test variable containing the age at death, which must be integer valued. If the TIME= option is omitted, all ages are assumed to equal 1.

UPPERTAILED
UPPER
is used to make all tests upper-tailed. All tests are two-tailed by default.

Details: MULTTEST Procedure

Statistical Tests

The following section discusses the statistical tests performed in the MULTTEST procedure. For continuous data, a *t* test for the mean (MEAN) is available. For discrete variables, available tests are the Cochran-Armitage linear trend test (CA), the Freeman-Tukey double arcsine test (FT), the Peto mortality-prevalence test (PETO), and the Fisher exact test (FISHER).

Throughout this section, the discrete and continuous variables are denoted by S_{vgsr} and X_{vgsr}, respectively, where v is the variable, g is the treatment group, s is the stratum, and r is the replication. Let m_{vgs} denote the sample size for a binary variable v within group g and stratum s. A plus sign (+) subscript denotes summation over an index. Note that the tests are invariant to the location and scale of the contrast coefficients t_g.

Cochran-Armitage Linear Trend Test

The Cochran-Armitage linear trend test (Cochran 1954; Armitage 1955; Agresti 2002) is implemented by using a *Z*-score approximation, an exact permutation distribution, or a combination of both.

Z-Score Approximation

The pooled probability estimate for variable v and stratum s is

$$p_{vs} = \frac{S_{v+s+}}{m_{v+s}}$$

The expected value (under constant within-stratum treatment probabilities) for variable v, group g, and stratum s is

$$E_{vgs} = m_{vgs} p_{vs}$$

Letting t_g denote the contrast trend coefficients specified by the CONTRAST statement, the test statistic for variable v has numerator

$$N_v = \sum_s \sum_g t_g (S_{vgs+} - E_{vgs})$$

The binomial variance estimate for this statistic is

$$V_v = \sum_s p_{vs}(1 - p_{vs}) \sum_g m_{vgs}(t_g - \bar{t}_{vs})^2$$

where

$$\bar{t}_{vs} = \sum_g \frac{m_{vgs} t_g}{m_{v+s}}$$

The hypergeometric variance estimate (the default) is

$$V_v = \sum_s \{m_{v+s}/(m_{v+s} - 1)\} p_{vs}(1 - p_{vs}) \sum_g m_{vgs}(t_g - \bar{t}_{vs})^2$$

For any strata s with $m_{v+s} \leq 1$, the contribution to the variance is taken to be zero.

PROC MULTTEST computes the Z-score statistic

$$Z_v = \frac{N_v}{\sqrt{V_v}}$$

The p-value for this statistic comes from the standard normal distribution. Whenever a 0 is computed for the denominator, the p-value is set to 1. This p-value approximates the probability obtained from the exact permutation distribution, discussed in the following text.

The Z-score statistic can be continuity-corrected to better approximate the permutation distribution. With continuity correction c, the upper-tailed p-value is computed from

$$Z_v = \frac{N_v - c}{\sqrt{V_v}}$$

For two-tailed, noncontinuity-corrected tests, PROC MULTTEST reports the p-value as $2\min(p, 1 - p)$, where p is the upper-tailed p-value. The same formula holds for the continuity-corrected test, with the exception that when the noncontinuity-corrected Z and the continuity-corrected Z have opposite signs, the two-tailed p-value is 1.

When the PERMUTATION= option is specified and no STRATA variable is specified, PROC MULTTEST uses a continuity correction selected to optimally approximate the upper-tail probability of permutation distributions with smaller marginal totals (Westfall and Lin 1988). Otherwise, the continuity correction is specified by the CONTINUITY= option in the TEST statement.

The CA Z-score statistic is the Hoel-Walburg (Mantel-Haenszel) statistic reported by Dinse (1985).

Exact Permutation Test

When you use the PERMUTATION= option for CA in the TEST statement, PROC MULTTEST computes the exact permutation distribution of the trend score

$$T_v = \sum_s \sum_g t_g S_{vgs+}$$

where the contrast trend coefficients t_g must be integer valued. The observed value of this trend is compared to the permutation distribution to obtain the p-value

$$p_v = \Pr(X \geq \text{ observed } T_v)$$

where X is a random variable from the permutation distribution and where upper-tailed tests are requested. This probability can be viewed as a binomial probability, where the within-stratum probabilities are constant and where the probability is conditional with respect to the marginal totals S_{v+s+}. It also can be considered a rerandomization probability.

Because the computations can be quite time-consuming with large data sets, specifying the PERMUTATION=*number* option in the TEST statement limits the situations where PROC MULTTEST computes the exact permutation distribution. When marginal total success or total failure frequencies exceed *number* for a particular stratum, the permutation distribution is approximated by a continuity-corrected normal distribution. You should be cautious when using the PERMUTATION= option in conjunction with bootstrap resampling because the permutation distribution is recomputed for each bootstrap sample. This recomputation is not necessary with permutation resampling.

The permutation distribution is computed in two steps:

1. The permutation distributions of the trend scores are computed within each stratum.
2. The distributions are convolved to obtain the distribution of the total trend.

As long as the total success or failure frequency does not exceed *number* for any stratum, the computed distributions are exact. In other words, if $S_{v+s+} \leq$ *number* or $(m_{v+s} - S_{v+s+}) \leq$ *number* for all s, then the permutation trend distribution for variable v is computed exactly.

In step 1, the distribution of the within-stratum trend

$$\sum_g t_g S_{vgs+}$$

is computed by using the multivariate hypergeometric distribution of the S_{vgs+}, provided *number* is not exceeded. This distribution can be written as

$$\Pr(S_{v1s+}, S_{v2s+}, \ldots, S_{vGs+}) = \prod_{g=1}^{G} \frac{\binom{m_{vgs}}{S_{vgs+}}}{\binom{m_{v+s}}{S_{v+s+}}}$$

The distribution of the within-stratum trend is then computed by summing these probabilities over appropriate configurations. For further information about this technique, see Bickis and Krewski (1986) and Westfall and Lin (1988). In step 2, the exact convolution distribution is obtained for the trend statistic summed over all strata having totals that meet the threshold criterion. This distribution is obtained by applying the fast Fourier transform to the exact within-stratum distributions. A description of this general method can be found in Pagano and Tritchler (1983) and Good (1987).

The convolution distribution of the overall trend is then computed by convolving the exact distribution with the distribution of the continuity-corrected standard normal approximation. To be more specific, let S_1 denote the subset of stratum indices that satisfy the threshold criterion, and let S_2 denote the subset of indices that do not satisfy the criterion. Let T_{v1} denote the combined trend statistic from the set S_1, which has an exact distribution obtained from Fourier analysis as previously outlined, and let T_{v2} denote the combined trend statistic from the set S_2. Then the distribution of the overall trend $T_v = T_{v1} + T_{v2}$ is obtained by convolving

the analytic distribution of T_{v1} with the continuity-corrected normal approximation for T_{v2}. Using the notation from the section "Z-Score Approximation" on page 7064, this convolution can be written as

$$\begin{aligned}\Pr(T_{v1} + T_{v2} \geq u) &= \sum_{u1} \Pr(T_{v1} + T_{v2} \geq u \mid T_{v1} = u1) \Pr(T_{v1} = u1) \\ &\approx \sum_{u1} \Pr(Z \geq z) \Pr(T_{v1} = u1)\end{aligned}$$

where Z is a standard normal random variable, and

$$z = \frac{1}{\sqrt{V_v}} \left(u - u1 - \sum_{S_2} p_{vs} \sum_g t_g m_{vgs} - c \right)$$

In this expression, the summation of s in V_v is over S_2, and c is the continuity correction discussed under the Z-score approximation.

When a two-tailed test is requested, the expected trend is computed

$$E_v = \sum_s \sum_g t_g E_{vgs}$$

The two-tailed p-value is reported as the permutation tail probability for the observed trend T_v plus the permutation tail probability for $2E_v - T_v$, the reflected trend.

Freeman-Tukey Double Arcsine Test

For this test, the contrast trend coefficients $t_1, \ldots, t_G$ are centered to the values $c_1, \ldots, c_G$, where $c_g = t_g - \bar{t}$, $\bar{t} = \sum_g t_g / G$, and G is the number of groups. The numerator of this test statistic is

$$N_v = \sum_s w_{vs} \sum_g c_g f(S_{vgs+}, m_{vgs})$$

where the weights w_{vs} take on three different types of values depending upon your specification of the WEIGHT= option in the STRATA statement. The default value is the within-strata sample size m_{v+s}, ensuring comparability with the ordinary CA trend statistic. WEIGHT=HARMONIC sets w_{vs} equal to the harmonic mean

$$\left[\left(\sum_g \frac{1}{m_{vgs}} \right) / G^* \right]^{-1}$$

where G^* is the number of nonmissing groups and the summation is over only the nonmissing elements. The harmonic means analysis places more weight on the smaller sample sizes than does the default sample size method, and is similar to a Type 2 analysis in PROC GLM. WEIGHT=EQUAL sets $w_{vs} = 1$ for all v and s, and is similar to a Type 3 analysis in PROC GLM.

The function $f(r, n)$ is the double arcsine transformation:

$$f(r, n) = \arcsin\left(\sqrt{\frac{r}{n+1}} \right) + \arcsin\left(\sqrt{\frac{r+1}{n+1}} \right)$$

The variance estimate is

$$V_v = \sum_s w_{vs}^2 \sum_g \frac{c_g^2}{m_{vgs} + \frac{1}{2}}$$

The test statistic is

$$Z_v = \frac{N_v}{\sqrt{V_v}}$$

The Freeman-Tukey transformation and its variance are described by Freeman and Tukey (1950) and Miller (1978). Since its variance is not weighted by the pooled probabilities, as is the CA test, the FT test can be more useful than the CA test for tests involving only a subset of the groups.

Peto Mortality-Prevalence Trend Test

The Peto test is a modified Cochran-Armitage procedure incorporating mortality and prevalence information. The Peto test is computed like two Cochran-Armitage Z-score approximations, one for prevalence and one for mortality (Peto et al. 1980). It represents a special case in PROC MULTTEST because the data structure requirements are different, and the resampling methods used for adjusting p-values are not valid. The TIME= option variable is required to specify "death" times or, more generally, times of occurrence. In addition, the test variables must assume one of the following three values:

- 0 = no occurrence
- 1 = incidental occurrence
- 2 = fatal occurrence

Use the TIME= option variable to define the mortality strata, and use the STRATA statement variable to define the prevalence strata.

In the following notation, the subscript v represents the variable, g represents the treatment group, s represents the stratum, and t represents the time. Recall that a plus sign (+) in a subscript location denotes summation over that subscript.

Let S_{vgs}^P be the number of incidental occurrences, and let m_{vgs}^P be the total sample size for variable v in group g, stratum s, excluding fatal tumors.

Let S_{vgt}^F be the number of fatal occurrences in time period t, and let m_{vgt}^F be the number of patients alive at the end of time $t - 1$.

The pooled probability estimates are given by

$$\begin{aligned} p_{vs}^P &= \frac{S_{v+s}^P}{m_{v+s}^P} \\ p_{vt}^F &= \frac{S_{v+t}^F}{m_{v+t}^F} \end{aligned}$$

The expected values are

$$
\begin{aligned}
E^P_{vgs} &= m^P_{vgs} p^P_{vs} \\
E^F_{vgt} &= m^F_{vgt} p^F_{vt}
\end{aligned}
$$

Let t_g denote a contrast trend coefficient, and define the numerator terms as follows:

$$
\begin{aligned}
N^P_v &= \sum_s \sum_g t_g \left(S^P_{vgs} - E^P_{vgs} \right) \\
N^F_v &= \sum_t \sum_g t_g \left(S^F_{vgt} - E^F_{vgt} \right)
\end{aligned}
$$

Define the denominator variance terms by using the binomial variance:

$$
\begin{aligned}
V^P_v &= \sum_s p^P_{vs} \left(1 - p^P_{vs} \right) \left[\left(\sum_g m^P_{vgs} {t_g}^2 \right) - \frac{1}{m^P_{v+s}} \left(\sum_g m^P_{vgs} t_g \right)^2 \right] \\
V^F_v &= \sum_s p^F_{vt} \left(1 - p^F_{vt} \right) \left[\left(\sum_g m^F_{vgt} {t_g}^2 \right) - \frac{1}{m^F_{v+t}} \left(\sum_g m^F_{vgt} t_g \right)^2 \right]
\end{aligned}
$$

The hypergeometric variances (the default) are calculated by weighting the within-strata variances as discussed in the section "Z-Score Approximation" on page 7064.

The Peto statistic is computed as

$$
Z_v = \frac{N^P_v + N^F_v - c}{\sqrt{V^P_v + V^F_v}}
$$

where c is a continuity correction. The p-value is determined from the standard normal distribution unless the PERMUTATION=*number* option is used. When you use the PERMUTATION= option for PETO in the TEST statement, PROC MULTTEST computes the "discrete approximation" permutation distribution described by Mantel (1980) and Soper and Tonkonoh (1993). Specifically, the permutation distribution of $\sum_s \sum_g t_g S^P_{vgs} + \sum_t \sum_g t_g S^F_{vgt}$ is computed, assuming that $\{\sum_g t_g S^P_{vgs}\}$ and $\{\sum_g t_g S^F_{vgt}\}$ are independent over all s and t. Note that the contrast trend coefficients t_g must be integer valued. The p-values are exact under this independence assumption. However, the independence assumption is valid only asymptotically, which is why these p-values are called "approximate."

An exact permutation distribution is available only under the assumption of equal risk of censoring in all treatment groups; even then, computing this distribution can be cumbersome. Soper and Tonkonoh (1993) describe situations where the discrete approximation distribution closely fits the exact permutation distribution.

Fisher Exact Test

The CONTRAST statement in PROC MULTTEST enables you to compute Fisher exact tests for two-group comparisons. No stratification variable is allowed for this test. Note, however, that the FISHER exact test is a special case of the exact permutation tests performed by PROC MULTTEST and that these permutation tests allow a stratification variable. Recall that contrast coefficients can be –1, 0, or 1 for the Fisher test. The frequencies and sample sizes of the groups scored as –1 are combined, as are the frequencies and sample sizes of the groups scored as 1. Groups scored as 0 are excluded. The –1 group is then compared with the 1 group by using the Fisher exact test.

Letting x and m denote the frequency and sample size of the 1 group, and letting y and n denote those of the –1 group, the p-value is calculated as

$$\Pr(X \geq x \mid X + Y = x + y) = \sum_{i=x}^{m} \frac{\binom{m}{i}\binom{n}{x+y-i}}{\binom{m+n}{x+y}}$$

where X and Y are independent binomially distributed random variables with sample sizes m and n and common probability parameters. The hypergeometric distribution is used to determine the stated probability; Yates (1984) discusses this technique. PROC MULTTEST computes the two-tailed p-values by adding probabilities from both tails of the hypergeometric distribution. The first tail is from the observed x and y, and the other tail is chosen so that the resulting probability is as large as possible without exceeding the probability from the first tail. If the variable being tested has only one level, then the p-value is set to 1.

t Test for the Mean

For continuous variables, PROC MULTTEST automatically centers the contrast trend coefficients, as in the Freeman-Tukey test. These centered coefficients c_g are then used to form a t statistic contrasting the within-group means. Let n_{vgs} denote the sample size within group g and stratum s; it depends on variable v only when there are missing values. Determine the weights w_{vs} as in the Freeman-Tukey test with n_{vgs} replacing m_{vgs}. Define

$$\bar{X}_{vgs+} = \frac{1}{n_{vgs}} \sum_r X_{vgsr}$$

as the sample mean within a group-and-stratum combination, and let μ_{vgs} denote the treatment means. Write the null hypothesis as

$$\sum_s w_{vs} \sum_g c_g \mu_{vgs} = 0$$

Also define

$$s_v^2 = \frac{\sum_s \sum_g \sum_r \left(X_{vgsr} - \bar{X}_{vgs+}\right)^2}{\sum_s \sum_g \left(n_{vgs} - 1\right)}$$

as the pooled sample variance.

Homogeneous Variance

Assuming constant variance for all group-and-stratum combinations, the t statistic for the mean is

$$M_v = \frac{\sum_s w_{vs} \sum_g c_g \bar{X}_{vgs+}}{\sqrt{s_v^2 \left(\sum_s w_{vs}^2 \sum_g \frac{c_g^2}{n_{vgs}} \right)}}$$

Then under the null hypothesis and assuming normality, independence, and homoscedasticity, M_v follows a t distribution with $df_p = \sum_s \sum_g \left(n_{vgs} - 1\right)$ degrees of freedom.

Whenever a denominator of 0 is computed, the p-value is set to 1. When missing data force $n_{vgs} = 0$, the contribution to the denominator of the pooled variance is 0 and not –1. This is also true for the degrees of freedom.

Heterogeneous Variance

If you do not assume constant variance for all group-and-stratum combinations, then the approximate t test is

$$M_v = \frac{\sum_s w_{vs} \sum_g c_g \bar{X}_{vgs+}}{\sqrt{\sum_s w_{vs}^2 \sum_g c_g^2 \frac{s_{vgs}^2}{n_{vgs}}}}$$

Under the null hypothesis and assuming normality and independence, the Satterthwaite (1946) approximation for the degrees of freedom of the t test is given by

$$df_s = \frac{\left(\sum_s w_{vs}^2 \sum_g c_g^2 \frac{s_{vgs}^2}{n_{vgs}} \right)^2}{\sum_s \sum_g \frac{\left(w_{vs}^2 c_g^2 \frac{s_{vgs}^2}{n_{vgs}} \right)^2}{n_{vgs} - 1}}$$

under the restriction $1 \leq df_s \leq \sum_s \sum_g n_{vgs}$.

Whenever a denominator of 0 for M_v is computed, the p-value is set to 1. If the denominator for df_s is computed as 0, then set $df_s = df_p$. When missing data force $n_{vgs} \leq 1$, that group-and-stratum combination does not contribute to the df_s computation.

p-Value Adjustments

Suppose you test m null hypotheses, $H_{01}, \ldots, H_{0m}$, and obtain the p-values $p_1, \ldots, p_m$. Denote the ordered p-values as $p_{(1)} \leq \cdots \leq p_{(m)}$ and order the tests appropriately: $H_{0(1)}, \ldots, H_{0(m)}$. Suppose you know m_0 of the null hypotheses are true and $m_1 = m - m_0$ are false. Let R indicate the number of null hypotheses rejected by the tests, where V of these are incorrectly rejected (that is, V tests are Type I errors) and $R - V$ are correctly rejected (so $m_1 - R + V$ tests are Type II errors). This information is summarized in the following table:

	Null Is Rejected	**Null Is Not Rejected**	**Total**
Null is True	V	$m_0 - V$	m_0
Null is False	$R - V$	$m_1 - R + V$	m_1
Total	R	$m - R$	m

The *familywise error rate* (FWE) is the overall Type I error rate for all the comparisons (possibly under some restrictions); that is, it is the maximum probability of incorrectly rejecting one or more null hypotheses:

$$\text{FWE} = \Pr(V > 0)$$

The FWE is also known as the *maximum experimentwise error rate* (MEER), as discussed in the section "Pairwise Comparisons" on page 4232 in Chapter 53, "The GLM Procedure."

The *false discovery rate* (FDR) is the expected proportion of incorrectly rejected hypotheses among all rejected hypotheses:

$$\begin{aligned}\text{FDR} &= E\left(\frac{V}{R}\right) \quad \text{where } \frac{V}{R} = 0 \text{ when } V = R = 0 \\ &= E\left(\frac{V}{R} \mid R > 0\right) \Pr(R > 0)\end{aligned}$$

Under the overall null hypothesis (all the null hypotheses are true), the FDR=FWE since $V = R$ gives $E\left(\frac{V}{R}\right) = 1 \times \Pr\left(\frac{V}{R} = 1\right) = \Pr(V > 0)$. Otherwise, FDR is always less than FWE, and an FDR-controlling adjustment also controls the FWE. Another definition used is the *positive* false discovery rate:

$$\text{pFDR} = E\left(\frac{V}{R} \mid R > 0\right)$$

The p-value adjustment methods discussed in the following sections attempt to correct the raw p-values while controlling either the FWE or the FDR. Note that the methods might impose some restrictions in order to achieve this; restrictions are discussed along with the methods in the following sections. Discussions and comparisons of some of these methods are given in Dmitrienko et al. (2005), Dudoit, Shaffer, and Boldrick (2003), Westfall et al. (1999), and Brown and Russell (1997).

Familywise Error Rate Controlling Adjustments

PROC MULTTEST provides several p-value adjustments to control the familywise error rate. *Single-step* adjustment methods are computed without reference to the other hypothesis tests under consideration. The available single-step methods are the Bonferroni and Šidák adjustments, which are simple functions of the raw p-values that try to distribute the significance level α across all the tests, and the bootstrap and permutation resampling adjustments, which require the raw data. The Bonferroni and Šidák methods are calculated from the permutation distributions when exact permutation tests are used with the CA or Peto test.

Stepwise tests, or *sequentially rejective* tests, order the hypotheses in *step-up* (least significant to most significant) or *step-down* fashion, then sequentially determine acceptance or rejection of the nulls. These tests are more powerful than the single-step tests, and they do not always require you to perform every test. However, PROC MULTTEST still adjusts every p-value. PROC MULTTEST provides the following stepwise p-value adjustments: step-down Bonferroni (Holm), step-down Šidák, step-down bootstrap and permutation resampling, Hochberg's (1988) step-up, Hommel's (1988), Fisher's combination method, and the Stouffer-Liptak combination method. Adaptive versions of Holm's step-down Bonferroni and Hochberg's step-up Bonferroni methods, which require an estimate of the number of true null hypotheses, are also available.

Liu (1996) shows that all single-step and stepwise tests based on marginal p-values can be used to construct a *closed* test (Marcus, Peritz, and Gabriel 1976; Dmitrienko et al. 2005). Closed testing methods not only control the familywise error rate at size α, but are also more powerful than the tests on which they are based. Westfall and Wolfinger (2000) note that several of the methods available in PROC MULTTEST are closed—namely, the step-down methods, Hommel's method, and Fisher's combination; see that reference for conditions and exceptions.

All methods except the resampling methods are calculated by simple functions of the raw p-values or marginal permutation distributions; the permutation and bootstrap adjustments require the raw data. Because the resampling techniques incorporate distributional and correlational structures, they tend to be less conservative than the other methods.

When a resampling (bootstrap or permutation) method is used with only one test, the adjusted p-value is the bootstrap or permutation p-value for that test, with no adjustment for multiplicity, as described by Westfall and Soper (1994).

Bonferroni

The Bonferroni p-value for test $i, i = 1, \ldots, m$ is simply $\tilde{p}_i = mp_i$. If the adjusted p-value exceeds 1, it is set to 1. The Bonferroni test is conservative but always controls the familywise error rate.

If the unadjusted p-values are computed by using exact permutation distributions, then the Bonferroni adjustment for p_i is $\tilde{p}_i = p_1^* + \cdots + p_m^*$, where p_j^* is the largest p-value from the permutation distribution of test j satisfying $p_j^* \leq p_i$, or 0 if all permutational p-values of test j are greater than p_i. These adjustments are much less conservative than the ordinary Bonferroni adjustments because they incorporate the discrete distributional characteristics. However, they remain conservative in that they do not incorporate correlation structures between multiple contrasts and multiple variables (Westfall and Wolfinger 1997).

Šidák

A technique slightly less conservative than Bonferroni is the Šidák p-value (Šidák 1967), which is $\tilde{p}_i = 1 - (1 - p_i)^m$. It is exact when all of the p-values are uniformly distributed and independent, and it is conservative when the test statistics satisfy the positive orthant dependence condition (Holland and Copenhaver 1987).

If the unadjusted p-values are computed by using exact permutation distributions, then the Šidák adjustment for p_i is $\tilde{p}_i = 1 - (1 - p_1^*) \cdots (1 - p_m^*)$, where the p_j^* are as described previously. These adjustments are less conservative than the corresponding Bonferroni adjustments, but they do not incorporate correlation structures between multiple contrasts and multiple variables (Westfall and Wolfinger 1997).

Bootstrap

The bootstrap method creates pseudo-data sets by sampling observations with replacement from each within-stratum pool of observations. An entire data set is thus created, and p-values for all tests are computed on this pseudo-data set. A counter records whether the minimum p-value from the pseudo-data set is less than or equal to the actual p-value for each base test. (If there are m tests, then there are m such counters.) This process is repeated a large number of times, and the proportion of resampled data sets where the minimum pseudo-p-value is less than or equal to an actual p-value is the adjusted p-value reported by PROC MULTTEST. The algorithms are described in Westfall and Young (1993).

In the case of continuous data, the pooling of the groups is not likely to re-create the shape of the null hypothesis distribution, since the pooled data are likely to be multimodal. For this reason, PROC MULTTEST automatically mean-centers all continuous variables prior to resampling. Such mean-centering is akin to resampling residuals in a regression analysis, as discussed by Freedman (1981). You can specify the NOCENTER option if you do not want to center the data.

The bootstrap method implicitly incorporates all sources of correlation, from both the multiple contrasts and the multivariate structure. The adjusted p-values incorporate all correlations and distributional characteristics. This method always provides weak control of the familywise error rate, and it provides strong control when the *subset pivotality* condition holds; that is, for any subset of the null hypotheses, the joint distribution of the p-values for the subset is identical to that under the complete null (Westfall and Young 1993).

Permutation

The permutation-style-adjusted p-values are computed in identical fashion as the bootstrap-adjusted p-values, with the exception that the within-stratum resampling is performed without replacement instead of with replacement. This produces a rerandomization analysis such as in Brown and Fears (1981) and Heyse and Rom (1988). In the spirit of rerandomization analyses, the continuous variables are not centered prior to resampling. This default can be overridden by using the CENTER option.

The permutation method implicitly incorporates all sources of correlation, from both the multiple contrasts and the multivariate structure. The adjusted p-values incorporate all correlations and distributional characteristics. This method always provides weak control of the familywise error rate, and it provides strong control of the familywise error rate under the *subset pivotality* condition, as described in the preceding section.

Step-Down Methods

Step-down testing is available for the Bonferroni, Šidák, bootstrap, and permutation methods. The benefit of using step-down methods is that the tests are made more powerful (smaller adjusted p-values) while, in most cases, maintaining strong control of the familywise error rate. The step-down method was pioneered by Holm (1979) and further developed by Shaffer (1986), Holland and Copenhaver (1987), and Hochberg and Tamhane (1987).

The Bonferroni step-down (Holm) p-values $\tilde{p}_{(1)}, \ldots, \tilde{p}_{(m)}$ are obtained from

$$\tilde{p}_{(i)} = \begin{cases} m p_{(1)} & \text{for } i = 1 \\ \max\left(\tilde{p}_{(i-1)}, (m - i + 1) p_{(i)}\right) & \text{for } i = 2, \ldots, m \end{cases}$$

As always, if any adjusted p-value exceeds 1, it is set to 1.

The Šidák step-down p-values are determined similarly:

$$\tilde{p}_{(i)} = \begin{cases} 1-(1-p_{(1)})^m & \text{for } i = 1 \\ \max\left(\tilde{p}_{(i-1)}, 1-(1-p_{(i)})^{m-i+1}\right) & \text{for } i = 2,\ldots,m \end{cases}$$

Step-down Bonferroni adjustments that use exact tests are defined as

$$\tilde{p}_{(i)} = \begin{cases} p^*_{(1)} + \cdots + p^*_{(m)} & \text{for } i = 1 \\ \max\left(\tilde{p}_{(i-1)}, p^*_{(i)} + \cdots + p^*_{(m)},\right) & \text{for } i = 2,\ldots,m \end{cases}$$

where the p^*_j are defined as before. Note that p^*_j is taken from the permutation distribution corresponding to the jth-smallest unadjusted p-value. Also, any $\tilde{p}_j$ greater than 1.0 is reduced to 1.0.

Step-down Šidák adjustments for exact tests are defined analogously by substituting $1-(1-p^*_{(j)})\cdots(1-p^*_{(m)})$ for $p^*_{(j)} + \cdots + p^*_{(m)}$.

The resampling-style step-down methods are analogous to the preceding step-down methods; the most extreme p-value is adjusted according to all m tests, the second-most extreme p-value is adjusted according to $(m-1)$ tests, and so on. The difference is that all correlational and distributional characteristics are incorporated when you use resampling methods. More specifically, assuming the same ordering of p-values as discussed previously, the resampling-style step-down-adjusted p-value for test i is the probability that the minimum pseudo-p-value of tests $i,\ldots,m$ is less than or equal to p_i.

This probability is evaluated by using Monte Carlo simulation, as are the previously described resampling-style-adjusted p-values. In fact, the computations for step-down-adjusted p-values are essentially no more time-consuming than the computations for the non-step-down-adjusted p-values. After Monte Carlo, the step-down-adjusted p-values are corrected to ensure monotonicity; this correction leaves the first adjusted p-values alone, then corrects the remaining ones as needed. The step-down method approximately controls the familywise error rate, and it is described in more detail by Westfall and Young (1993), Westfall et al. (1999), and Westfall and Wolfinger (2000).

Hommel

Hommel's (1988) method is a closed testing procedure based on Simes' test (Simes 1986). The Simes p-value for a joint test of any set of S hypotheses with p-values $p_{(1)} \le p_{(2)} \le \ldots \le p_{(S)}$ is $\min((S/1)p_{(1)}, (S/2)p_{(2)}, \ldots, (S/S)p_{(S)})$. The Hommel-adjusted p-value for test j is the maximum of all such Simes p-values, taken over all joint tests that include j as one of their components.

Hochberg-adjusted p-values are always as large or larger than Hommel-adjusted p-values. Sarkar and Chang (1997) shows that Simes' method is valid under independent or positively dependent p-values, so Hommel's and Hochberg's methods are also valid in such cases by the closure principle.

Hochberg

Assuming p-values are independent and uniformly distributed under their respective null hypotheses, Hochberg (1988) demonstrates that Holm's step-down adjustments control the familywise error rate even when calculated in *step-up* fashion. Since the adjusted p-values are uniformly smaller for Hochberg's method than for Holm's method, the Hochberg method is more powerful. However, this improved power comes at the cost of having to make the assumption of independence. Hochberg's method can be derived from Hommel's (Liu 1996), and is thus also derived from Simes' test (Simes 1986).

Hochberg-adjusted p-values are always as large or larger than Hommel-adjusted p-values. Sarkar and Chang (1997) showed that Simes' method is valid under independent or positively dependent p-values, so Hommel's and Hochberg's methods are also valid in such cases by the closure principle.

The Hochberg-adjusted p-values are defined in reverse order of the step-down Bonferroni:

$$\tilde{p}_{(i)} = \begin{cases} p_{(m)} & \text{for } i = m \\ \min\left(\tilde{p}_{(i+1)}, (m-i+1)\,p_{(i)}\right) & \text{for } i = m-1, \ldots, 1 \end{cases}$$

Fisher Combination

The FISHER_C option requests adjusted p-values by using closed tests, based on the idea of Fisher's combination test. The Fisher combination test for a joint test of any set of S hypotheses with p-values uses the chi-square statistic $\chi^2 = -2\sum \log(p_i)$, with $2S$ degrees of freedom. The FISHER_C adjusted p-value for test j is the maximum of all p-values for the combination tests, taken over all joint tests that include j as one of their components. Independence of p-values is required for the validity of this method.

Stouffer-Liptak Combination

The STOUFFER option requests adjusted p-values by using closed tests, based on the Stouffer-Liptak combination test. The Stouffer combination joint test of any set of S one-sided hypotheses with p-values, $p_1, \ldots, p_S$, yields the p-value, $1 - \Phi\left(\frac{1}{\sqrt{S}} \sum_i \Phi^{-1}(1-p_i)\right)$. The STOUFFER adjusted p-value for test j is the maximum of all p-values for the combination tests, taken over all joint tests that include j as one of their components.

Independence of the one-sided p-values is required for the validity of this method. Westfall (2005) shows that the Stouffer-Liptak adjustment might have more power than the Fisher combination and Simes' adjustments when the test results reinforce each other.

Adaptive Adjustments

Adaptive adjustments modify the FWE- and FDR-controlling procedures by taking an estimate of the number m_0 or proportion π_0 of true null hypotheses into account. The adjusted p-values for Holm's and Hochberg's methods involve the number of unadjusted p-values larger than (i), $m - i + 1$. So the minimal significance level at which the ith ordered p-value is rejected implies that the number of true null hypotheses is $m - i + 1$. However, if you know m_0, then you can replace $m - i + 1$ with $\min(m_0, m - i + 1)$, thereby obtaining more power while maintaining the original α-level significance.

Since m_0 is unknown, there are several methods used to estimate the value—see the NTRUENULL= option for more information. The estimation method described by Hochberg and Benjamini (1990) considers the graph of $1 - p_{(i)}$ versus i, where the $p_{(i)}$ are the ordered p-values of your tests. See Output 86.6.4 for an example. If all null hypotheses are actually true ($m_0 = m$), then the p-values behave like a sample from a uniform distribution and this graph should be a straight line through the origin. However, if points in the upper-right corner of this plot do not follow the initial trend, then some of these null hypotheses are probably false and $0 < m_0 < m$.

The ADAPTIVEHOLM option uses this estimate of m_0 to adjust the step-up Bonferroni method while the ADAPTIVEHOCHBERG option adjusts the step-down Bonferroni method. Both of these methods are due to Hochberg and Benjamini (1990). When m_0 is known, these procedures control the familywise error rate in the same manner as their nonadaptive versions but with more power; however, since m_0 must be estimated, the FWE control is only approximate. The ADAPTIVEFDR and PFDR options also use $\hat{m}_0$, and are described in the following section.

The adjusted p-values for the ADAPTIVEHOLM method are computed by

$$\tilde{p}_{(i)} = \begin{cases} \min(m, \hat{m}_0)\, p_{(1)} & \text{for } i = 1 \\ \max\left[\tilde{p}_{(i-1)}, \min(m - i + 1, \hat{m}_0)\, p_{(i)}\right] & \text{for } i = 2, \ldots, m \end{cases}$$

The adjusted p-values for the ADAPTIVEHOCHBERG method are computed by

$$\tilde{p}_{(i)} = \begin{cases} \min(1, \hat{m}_0)\, p_{(m)} & \text{for } i = m \\ \min\left[\tilde{p}_{(i+1)}, \min(m - i + 1, \hat{m}_0)\, p_{(i)}\right] & \text{for } i = m - 1, \ldots, 1 \end{cases}$$

False Discovery Rate Controlling Adjustments

Methods that control the *false discovery rate* (FDR) were described by Benjamini and Hochberg (1995). These adjustments do not necessarily control the familywise error rate (FWE). However, FDR-controlling methods are more powerful and more liberal, and hence reject more null hypotheses, than adjustments protecting the FWE. FDR-controlling methods are often used when you have a large number of null hypotheses. To control the FDR, Benjamini and Hochberg's (1995) linear step-up method is provided, as well as an adaptive version, a dependence version, and bootstrap and permutation resampling versions. Storey's (2002) pFDR methods are also provided.

The FDR option requests p-values that control the "false discovery rate" described by Benjamini and Hochberg (1995). These *linear step-up* adjustments are potentially much less conservative than the Hochberg adjustments.

The FDR-adjusted p-values are defined in step-up fashion, like the Hochberg adjustments, but with less conservative multipliers:

$$\tilde{p}_{(i)} = \begin{cases} p_{(m)} & \text{for } i = m \\ \min\left(\tilde{p}_{(i+1)}, \frac{m}{i} p_{(i)}\right) & \text{for } i = m - 1, \ldots, 1 \end{cases}$$

The FDR method is guaranteed to control the false discovery rate at level $\leq \frac{m_0}{m}\alpha \leq \alpha$ when you have independent p-values that are uniformly distributed under their respective null hypotheses. Benjamini and Yekutieli (2001) show that the false discovery rate is also controlled at level $\leq \frac{m_0}{m}\alpha$ when the *positive regression dependent* condition holds on the set of the true null hypotheses, and they provide several examples where this condition is true.

NOTE: The positive regression dependent condition on the set of the true null hypotheses holds if the joint distribution of the test statistics $\mathbf{X} = (X_1, \ldots, X_m)$ for the null hypotheses $H_{01}, \ldots, H_{0m}$ satisfies: $\Pr(\mathbf{X} \in A | X_i = x)$ is nondecreasing in x for each X_i where H_{0i} is true, for any increasing set A. The set A is increasing if $\mathbf{x} \in A$ and $\mathbf{y} \geq \mathbf{x}$ implies $\mathbf{y} \in A$.

Dependent False Discovery Rate

The DEPENDENTFDR option requests a false discovery rate controlling method that is always valid for p-values under any kind of dependency (Benjamini and Yekutieli 2001), but is thus quite conservative. Let $\gamma = \sum_{i=1}^{m} \frac{1}{i}$. The DEPENDENTFDR procedure always controls the false discovery rate at level $\leq \frac{m_0}{m}\alpha\gamma$. The adjusted p-values are computed as

$$\tilde{p}_{(i)} = \begin{cases} \gamma p_{(m)} & \text{for } i = m \\ \min\left(\tilde{p}_{(i+1)}, \gamma \frac{m}{i} p_{(i)}\right) & \text{for } i = m - 1, \ldots, 1 \end{cases}$$

False Discovery Rate Resampling Adjustments

Bootstrap and permutation resampling methods to control the false discovery rate are available with the FDRBOOT and FDRPERM options (Yekutieli and Benjamini 1999). These methods approximately control the false discovery rate when the *subset pivotality* condition holds, as discussed in the section "Bootstrap" on page 7074, and when the p-values corresponding to the true null hypotheses are independent of those for the false null hypotheses.

The resampling methodology for the BOOTSTRAP and PERMUTATION methods is used to create B resamples. For the bth resample, let $R^b(p)$ denote the number of p-values that are less than or equal to the observed p-value p. Let $r_\beta(p)$ be the $100(1-\beta)$ quantile of $\{R^1(p)\ldots R^b(p)\ldots R^B(p)\}$, and let $r(p)$ be the number of observed p-values less than or equal to p. Compute one of the following estimators:

$$\text{local estimator} \qquad Q_1(p) = \begin{cases} \dfrac{1}{B}\displaystyle\sum_{b=1}^{B} \dfrac{R^b(p)}{R^b(p) + r(p) - pm} & \text{if } r(p) - r_\beta(p) \geq pm \\ \#\{R^b(p) \geq 1\}/B & \text{otherwise} \end{cases}$$

$$\text{upper limit estimator} \qquad Q_\beta(p) = \begin{cases} \displaystyle\sup_{x\in[0,p]} \left(\frac{1}{B}\sum_{b=1}^{B} \frac{R^b(x)}{R^b(x) + r(x) - r_\beta(x)} \right) & \text{if } r(x) - r_\beta(x) \geq 0 \\ \#\{R^b(p) \geq 1\}/B & \text{otherwise} \end{cases}$$

where m is the number of tests and B is the number of resamples. Then for $Q = Q_1$ or Q_β, the adjusted p-values are computed as

$$\tilde{p}_{(i)} = \begin{cases} Q(p_{(m)}) & \text{for } i = m \\ \min\left(\tilde{p}_{(i+1)}, Q(p_{(i)})\right) & \text{for } i = m-1, \ldots, 1 \end{cases}$$

Adaptive False Discovery Rate

Since the FDR method controls the false discovery rate at $\leq \frac{m_0}{m}\alpha \leq \alpha$, knowledge of m_0 allows improvement of the power of the adjustment while still maintaining control of the false discovery rate. The ADAPTIVEFDR option requests adaptive adjusted p-values for approximate control of the false discovery rate, as discussed in Benjamini and Hochberg (2000). See the section "Adaptive Adjustments" on page 7076 for more details. These adaptive adjustments are also defined in step-up fashion but use an estimate $\hat{m}_0$ of the number of true null hypotheses:

$$\tilde{p}_{(i)} = \begin{cases} \frac{\hat{m}_0}{m} p_{(m)} & \text{for } i = m \\ \min\left(\tilde{p}_{(i+1)}, \frac{\hat{m}_0}{i} p_{(i)}\right) & \text{for } i = m-1, \ldots, 1 \end{cases}$$

Since $\hat{m}_0 \leq m$, the larger p-values are adjusted down. This means that, as defined, controlling the false discovery rate enables you to reject these tests at a level less than the observed p-value. However, by default, this reduction is prevented with an additional restriction: $\tilde{p}_{(i)} = \max\{\tilde{p}_{(i)}, p_{(i)}\}$.

To use this adjustment, Benjamini and Hochberg (2000) suggest first specifying the FDR option—if at least one test is rejected at your level, then apply the ADAPTIVEFDR adjustment. Alternatively, Benjamini, Krieger, and Yekutieli (2006) apply the FDR adjustment at level $\frac{\alpha}{\alpha+1}$, then specify the resulting number of true hypotheses with the NTRUENULL= option and apply the ADAPTIVEFDR adjustment; they show that this *two-stage linear step-up* procedure controls the false discovery rate at level α for independent test statistics.

Positive False Discovery Rate

The PFDR option computes the "q-values" $\hat{q}_\lambda(p_i)$ (Storey 2002; Storey, Taylor, and Siegmund 2004), which are adaptive adjusted p-values for strong control of the false discovery rate when the p-values corresponding to the true null hypotheses are independent and uniformly distributed. There are four versions of the PFDR available. Let $N(\lambda)$ be the number of observed p-values that are less than or equal to λ; let m be the number of tests; let $f = 1$ if the FINITE option is specified, and otherwise set $f = 0$; and denote the estimated proportion of true null hypotheses by

$$\hat{\pi}_0(\lambda) = \frac{m - N(\lambda) + f}{(1-\lambda)m}$$

The default estimate of FDR is

$$\widehat{\mathrm{FDR}}_\lambda(p) = \frac{\hat{\pi}_0(\lambda)\,p}{\max(N(p),1)/m}$$

If you set $\lambda = 0$, then this is identical to the FDR adjustment.

The positive FDR is estimated by

$$\widehat{\mathrm{pFDR}}_\lambda(p) = \frac{\widehat{\mathrm{FDR}}_\lambda(p)}{1-(1-p)^m}$$

The finite-sample versions of these two estimators for independent null p-values are given by

$$\begin{aligned}
\widehat{\mathrm{FDR}}^*_\lambda(p) &= \begin{cases} \dfrac{\hat{\pi}^*_0(\lambda)\,p}{\max(N(p),1)/m} & \text{if } p \le \lambda \\ 1 & \text{if } p > \lambda \end{cases} \\
\widehat{\mathrm{pFDR}}^*_\lambda(p) &= \frac{\widehat{\mathrm{FDR}}^*_\lambda(p)}{1-(1-p)^m}
\end{aligned}$$

Finally, the adjusted p-values are computed as

$$\tilde{p}_i = \hat{q}_\lambda(p_i) = \inf_{p \ge p_i} \mathrm{FDR}_\lambda(p) \quad i = 1, \ldots, m$$

This method can produce adjusted p-values that are smaller than the raw p-values. This means that, as defined, controlling the false discovery rate enables you to reject these tests at a level less than the observed p-value. However, by default, this reduction is prevented with an additional restriction: $\tilde{p}_i = \max\{\tilde{p}_i, p_i\}$.

Missing Values

If a CLASS or STRATA variable has a missing value, then PROC MULTTEST removes that observation from the analysis.

When there are missing values for test variables, the within-group-and-stratum sample sizes can differ from variable to variable. In most cases this is not a problem; however, it is possible for all data to be missing for a particular group within a particular stratum. For continuous variables and Freeman-Tukey tests, PROC MULTTEST re-centers the contrast trend coefficients within strata where all data for a particular group are missing. Re-centering the MEAN tests could redefine your *t* tests in an undesirable fashion; for example, if you specify coefficients to contrast the first and third groups (`contrast -1 0 1`) but the third group is missing, then the re-centered coefficients become –0.5 and 0.5, thus contrasting the first and second groups. If this is the case, you can run your *t* tests in separate PROC MULTTEST invocations, then combine the data and adjust the *p*-values by using the INPVALUES= option. However, you might find this re-centering acceptable for the Freeman-Tukey trend tests, since the contrast still tests for an increasing trend. The Cochran-Armitage and Peto tests are unaffected by this situation.

PROC MULTTEST uses missing values for resampling if they exist in the original data set. If all variables have missing values for any observation, then PROC MULTTEST removes the observation prior to resampling. Otherwise, PROC MULTTEST treats all missing values as ordinary observations in the resampling. This means that different resampled data sets can have different group sizes. In some cases it means that a resampled data set can have all missing values for a particular variable in a particular group/stratum combination, even when values exist for that combination in the original data. For this reason, PROC MULTTEST recomputes all quantities within each pseudo-data set, including such items as centered scoring coefficients and degrees of freedom for *p*-values.

While PROC MULTTEST does provide analyses in missing value cases, you should not feel that it completely solves the missing-value problem. If you are concerned about the adverse effects of missing data on a particular analysis, you should consider using imputation and sensitivity analyses to assess the effects of the missing data.

Computational Resources

PROC MULTTEST keeps all of the data in memory to expedite resampling. A large portion of the memory requirement is thus 8*NOBS*NVAR bytes, where NOBS is the number of observations in the data set, and NVAR is the number of variables analyzed, including CLASS, FREQ, and STRATA variables.

If you specify PERMUTATION=*number* (for exact permutation distributions), then PROC MULTTEST requires additional memory. This requirement is approximately 4*NTEST*NSTRATA*CMAX**number**(*number* + 1) bytes, where NTEST is the number of contrasts, NSTRATA is the number of STRATA levels, and CMAX is the maximum contrast coefficient.

If you specify the FDRBOOT or FDRPERM option, then saving all the resamples in memory requires 8*NSAMPLE*NOBS bytes, where NSAMPLE is the number of resamples used.

The execution time is linear in the number of resamples; that is, 10,000 resamples will take 10 times longer than 1,000 resamples. The execution time is also linear in the sample size; that is, 100 resamples of size *N* will take 10 times longer than 100 resamples of size 10*N*.

Output Data Sets

OUT= Data Set

The OUT= data set contains contrast names (_test_), variable names (_var_), the contrast label (_contrast_), raw *p*-values (raw_p or the value specified in the INPVALUES= option), and all requested adjusted *p*-values (bon_p, sid_p, boot_p, perm_p, stpbon_p, stpsid_p, stpbootp, stppermp, hom_p, hoc_p, fic_p, stouffer_p, aholm_p, ahoc_p, fdr_p, dfdr_p, fdrbootp, ufdbootp, fdrpermp, ufdpermp, afdr_p, or pfdr_p).

If a resampling-based adjusted *p*-value is requested, then the simulation standard error is included as either sim_se, stpsimse, fdrsimse, or ufdsimse, depending on whether single-step, step-down, or FDR adjustments are requested. The simulation standard errors are used to bound the true resampling-based adjusted *p*-value. For example, if the resampling-based estimate is 0.0312 and the simulation standard error is 0.00123, then a 95% confidence interval for the true adjusted *p*-value is $0.0312 \pm 1.96(0.00123)$, or 0.0288 to 0.0336.

Intermediate statistics used to calculate the *p*-values are also written to the OUT= data set. The statistics are separated by the _strat_ level. When _strat_ is reported as missing, the statistics refer to the pooled analysis over all _strat_ levels. The *p*-values are provided only for the pooled analyses and are therefore reported as missing for the strata-specific statistics.

For the Peto test, an additional variable, _tstrat_, is included to indicate whether the stratum is an incidental occurrence stratum (_tstrat_=0) or a fatal occurrence stratum (_tstrat_=1).

The statistic _value_ is the per-strata contribution to the numerator of the overall test statistic. In the case of the MEAN test, this is the contrast function of the sample means multiplied by the total number of observations within the stratum. For the FT test, _value_ is the contrast function of the double-arcsine transformed proportions, again multiplied by the total number of observations within the stratum. For the CA and Peto tests, _value_ is the observed value of the trend statistic within that stratum.

When either PETO or CA is requested, the variable _exp_ is included; this variable contains the expected value of the trend statistic for the given stratum.

The statistic _se_ is the square root of the variance of the per-strata _value_ statistic for any of the tests.

For MEAN tests, the variable _nval_ is included. When reported with an individual stratum level (that is, when the _strat_ value is nonmissing), the value _nval_ refers to the within-stratum sample size. For the combined analysis (that is, the value of the _strat_ is missing), the value _nval_ contains degrees of freedom of the *t* distribution used to compute the unadjusted *p*-value.

When the FISHER test is requested, the OUT= data set contains the variables _xval_, _mval_, _yval_, and _nval_, which define observations and sample sizes in the two groups defined by the CONTRAST statement.

For example, the OUT= data set from the drug example in the section "Getting Started: MULTTEST Procedure" on page 7043 is displayed in Figure 86.5.

Figure 86.5 Output Data for the MULTTEST Procedure

Obs	_test_	_var_	_contrast_	_value_	_exp_	_se_	raw_p	boot_p	sim_se
1	CA	SideEff1	Trend	8	5	1.54303	0.05187	0.33880	.003346749
2	CA	SideEff2	Trend	7	5	1.54303	0.19492	0.84030	.002590327
3	CA	SideEff3	Trend	10	7	1.63299	0.06619	0.51895	.003532994
4	CA	SideEff4	Trend	10	6	1.60357	0.01262	0.08840	.002007305
5	CA	SideEff5	Trend	7	4	1.44749	0.03821	0.24080	.003023370
6	CA	SideEff6	Trend	9	6	1.60357	0.06137	0.43825	.003508468
7	CA	SideEff7	Trend	9	5	1.54303	0.00953	0.05135	.001560660
8	CA	SideEff8	Trend	8	5	1.54303	0.05187	0.33880	.003346749
9	CA	SideEff9	Trend	7	5	1.54303	0.19492	0.84030	.002590327
10	CA	SideEff10	Trend	8	6	1.60357	0.21232	0.90300	.002092737

OUTPERM= Data Set

The OUTPERM= data set contains contrast names (_contrast_), variable names (_var_), and the associated permutation distributions (_value_ and upper_p). PROC MULTTEST computes the permutation distributions when you use the PERMUTATION= option with the CA or Peto test. The _value_ variable represents the support of the distributions, and upper_p represents their cumulative upper-tail probabilities. The size of this data set depends on the number of variables and the support of their permutation distributions.

For information about how this distribution is computed, see the section "Exact Permutation Test" on page 7065. For an illustration, see Example 86.1.

OUTSAMP= Data Set

The OUTSAMP= data set contains the data sets used in the resampling analysis, if such an analysis is requested. The variable _sample_ indicates the number of the resampled data set. This variable ranges from 1 to the value of the NSAMPLE= option. For each value of the _sample_ variable, an entire resampled data set is included, with _stratum_, _class_, and all other variables in the original data set. The values of the original variables are mean-centered for the mean test, if requested. The variable _obs_ indicates the observation's position in the original data set.

Each new data set is randomly drawn from the original data set, either with (bootstrap) or without (permutation) replacement. The size of this data set is, thus, the number of observations in the original data set times the number of samples.

Displayed Output

The output produced by PROC MULTTEST is divided into several tables. If the DATA= data set is specified, then the following tables are displayed:

- The "Model Information" table provides a list of the options and settings used for that particular invocation of the procedure. This table is not displayed if the INPVALUES= data set is specified. Included in this list are the following items:
  - statistical tests
  - support of the exact permutation distribution for the CA and Peto tests
  - continuity corrections used for the CA test
  - test tails
  - strata adjustment
  - *p*-value adjustments and specified suboptions
  - centering of continuous variables
  - number of samples and seed
- The "Contrast Coefficients" table lists the coefficients used in constructing the statistical tests. These coefficients are either specified in CONTRAST statements or generated by default. The coefficients apply to the levels of the CLASS statement variable. If a MEAN or FT test is specified in the TEST statement, the centered coefficients are displayed. Patterns of missing values in your data set might affect the coefficients used in your analysis; the displayed contrasts take missing value patterns into account. See the section "Missing Values" on page 7080 for more information.
- The "Variable Tabulations" tables provide summary statistics for each variable listed in the TEST statement. Included for discrete variables are the count, sample size, and percentage of occurrences. For continuous variables, the mean, sample standard deviation, and sample size are displayed. All of the previously mentioned statistics are computed for distinct combinations of the CLASS and STRATA statement variables.

If the INPVALUES= data set is specified, then the following tables are displayed:

- The "P-Value Adjustment Information" table provides a list of the specified *p*-value adjustments. If an adaptive adjustment is specified (see section "Adaptive Adjustments" on page 7076), then the following settings are also displayed when appropriate:
  - whether the finite-sample version of the PFDR is used (FINITE)
  - the number of tuning parameters to check (NLAMBDA=), the maximum tuning parameter (MAXLAMBDA=), or the specified tuning parameter (LAMBDA=)
  - the degrees of freedom of the spline (DF=) and the smoothing parameter
  - the number of bootstrap resamples (NBOOT=) and the seed (SEED=)

- If the bootstrap or spline method for estimating the number of true null hypotheses m_0 is used and the PLOTS= option is specified, the "Lambda Values" table displays the m_0 estimates as a function of the tuning parameter λ. If the bootstrap method is used, the table also displays the mean-squared errors, the minimum of which is used to select a specific λ. This table contains the values used in the "Lambda Functions" plot.
- The "Estimated Number of True Null Hypotheses" table displays the p-value adjustment, the method used to estimate the number of true nulls, and an estimate of the number and proportion of true null hypotheses in the data set.

The following table is displayed unless the NOPVALUE option is specified:

- The "p-Values" table is a collection of the raw and adjusted p-values from the run of PROC MULTTEST. The p-values are identified by variable and test.

ODS Table Names

PROC MULTTEST assigns a name to each table it creates, and you must use this name to reference the table when using the Output Delivery System (ODS). These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 86.3 ODS Tables Produced by PROC MULTTEST

ODS Table Name	Description	Statement or Option
Continuous	Continuous variable tabulations	TEST with MEAN
Contrasts	Contrast coefficients	Default
Discrete	Discrete variable tabulations	TEST with CA, FT, PETO, or FISHER
LambdaValues	True null estimates	AHOLM, AHOC, AFDR, or PFDR
ModelInfo	Model information	Default
NumTrueNull	Estimates of number of true nulls	AHOLM, AHOC, AFDR, or PFDR
pValues	p-values from the tests	Default
pValueInfo	p-value adjustment information	INPVALUES=

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You must also specify the options in the PROC MULTTEST statement that are indicated in Table 86.4.

PROC MULTTEST assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 86.4.

Table 86.4 Graphs Produced by PROC MULTTEST

ODS Graph Name	Plot Description	Option
AdjPlots	Panel of adjusted *p*-value plots	PLOTS=ADJUSTED
AdjByRawRank	Adjusted by rank of raw *p*-values	PLOTS=ADJUSTED(UNPACK)
AdjbyRawP	Adjusted by raw *p*-values	PLOTS=ADJUSTED(UNPACK)
AdjBySignificant	Proportion significant by adjusted	PLOTS=ADJUSTED(UNPACK)
FalsePosBySignificant	Expected number of false positives by proportion significant	PLOTS=ADJUSTED(UNPACK)
PByTest	*p*-values by test	PLOTS=PBYTEST
LambdaPlot	MSE or NTRUENULL by lambda	PLOTS=LAMBDA and (NTRUENULL=BOOTSTRAP or NTRUENULL=SPLINE or PFDR)
ManhattanPlots	$-\log_{10}$(adjusted *p*-values) by test	PLOTS=MANHATTAN
RawUniformPlot	Raw *p*-values by rank and histogram	PLOTS=RAWPROB or AHOLM or AHOC or AFDR or PFDR
RawUniformPlot	Raw *p*-values by rank	PLOTS=RAWPROB(UNPACK) and AHOLM or AHOC or AFDR or PFDR
RawUniformHist	Histogram of raw *p*-values with kernel density	PLOTS=RAWPROB(UNPACK) and AHOLM or AHOC or AFDR or PFDR

Examples: MULTTEST Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 86.1: Cochran-Armitage Test with Permutation Resampling

(View the complete code for this example (multex1.sas) in the example repository.)

This example, from Keith Soper at Merck, illustrates the exact permutation Cochran-Armitage test carried out on permutation resamples. In the following data set, each observation represents an animal. The binary variables S1 and S2 indicate two tumor types, with 0s indicating no tumor (failure) and 1 indicating a tumor (success); note that they have perfect negative association. The grouping variable is Dose.

```
data a;
   input S1 S2 Dose @@;
   datalines;
0 1 1   1 0 1   0 1 1
0 1 1   0 1 1   1 0 1
1 0 2   1 0 2   0 1 2
1 0 2   0 1 2   1 0 2
1 0 3   1 0 3   1 0 3
0 1 3   0 1 3   1 0 3
;

proc multtest data=a permutation nsample=10000 seed=36607 outperm=pmt;
   test ca(S1 S2 / permutation=10 uppertailed);
   class Dose;
   contrast 'CA Linear Trend' 0 1 2;
run;
proc print data=pmt;
run;
```

The PROC MULTTEST statement requests 10,000 permutation resamples. The OUTPERM= option creates an output SAS data set pmt used for the exact permutation distribution computed for the CA test.

The TEST statement specifies an upper-tailed Cochran-Armitage linear trend test for S1 and S2. The cutoff for exact permutation calculations is 10, as specified with the PERMUTATION= option in the TEST statement. Since S1 and S2 have 10 and 8 successes, respectively, PROC MULTTEST uses exact permutation distributions to compute the *p*-values for both variables.

The groups for the CA test are the levels of Dose from the CLASS statement. The trend coefficients applied to these groups are 0, 1, and 2, respectively, as specified in the CONTRAST statement.

Finally, the PROC PRINT statement displays the SAS data set pmt, which contains the permutation distributions.

The results from this analysis are displayed in Output 86.1.1 through Output 86.1.5. You should check the "Model Information" table to verify that the analysis specifications are correct.

Output 86.1.1 Cochran-Armitage Test with Permutation Resampling

The Multtest Procedure

Model Information	
Test for discrete variables	Cochran-Armitage
Exact permutation distribution used	Everywhere
Tails for discrete tests	Upper-tailed
Strata weights	None
P-value adjustment	Permutation
Number of resamples	10000
Seed	36607

The label and coefficients from the CONTRAST statement are shown in Output 86.1.2.

Output 86.1.2 Contrast Coefficients

Contrast Coefficients			
	Dose		
Contrast	**1**	**2**	**3**
CA Linear Trend	0	1	2

Output 86.1.3 displays summary statistics for the two test variables, S1 and S2. The Count column lists the number of successes for each level of the CLASS variable, Dose. The NumObs column lists the sample size, and the Percent column lists the percentage of successes in the sample.

Output 86.1.3 Summary Statistics

Discrete Variable Tabulations				
Variable	**Dose**	**Count**	**NumObs**	**Percent**
S1	1	2	6	33.33
S1	2	4	6	66.67
S1	3	4	6	66.67
S2	1	4	6	66.67
S2	2	2	6	33.33
S2	3	2	6	33.33

The Raw column in Output 86.1.4 contains the p-values from the CA test, and the Permutation column contains the permutation-adjusted p-values.

Output 86.1.4 Resulting p-Values

p-Values			
Variable	**Contrast**	**Raw**	**Permutation**
S1	CA Linear Trend	0.1993	0.4009
S2	CA Linear Trend	0.9220	1.0000

This table shows that, for S1, the adjusted p-value is approximately twice the raw p-value. In fact, resamples with small (large) p-values for S1 have large (small) p-values for S2 due to the perfect negative association

of the variables, and hence the permutation-adjusted p-value for S1 should be $2 \times 0.1993 = 0.3986$; the difference is due to resampling error. For the same reason, since the raw p-value for S2 is 0.9220, the adjusted p-value equals 1. The permutation p-values for S1 and S2 also happen to be the Bonferroni-adjusted p-values for this example.

The OUTPERM= data set is displayed in Output 86.1.5, which contains the exact permutation distributions for S1 and S2 in terms of cumulative probabilities.

Output 86.1.5 Exact Permutation Distribution

Obs	_contrast_	_var_	_value_	upper_p
1	CA Linear Trend	S1	0	1.00000
2	CA Linear Trend	S1	1	1.00000
3	CA Linear Trend	S1	2	1.00000
4	CA Linear Trend	S1	3	1.00000
5	CA Linear Trend	S1	4	1.00000
6	CA Linear Trend	S1	5	0.99966
7	CA Linear Trend	S1	6	0.99609
8	CA Linear Trend	S1	7	0.97827
9	CA Linear Trend	S1	8	0.92205
10	CA Linear Trend	S1	9	0.80070
11	CA Linear Trend	S1	10	0.61011
12	CA Linear Trend	S1	11	0.38989
13	CA Linear Trend	S1	12	0.19930
14	CA Linear Trend	S1	13	0.07795
15	CA Linear Trend	S1	14	0.02173
16	CA Linear Trend	S1	15	0.00391
17	CA Linear Trend	S1	16	0.00034
18	CA Linear Trend	S1	17	0.00000
19	CA Linear Trend	S1	18	0.00000
20	CA Linear Trend	S1	19	0.00000
21	CA Linear Trend	S1	20	0.00000
22	CA Linear Trend	S2	0	1.00000
23	CA Linear Trend	S2	1	1.00000
24	CA Linear Trend	S2	2	1.00000
25	CA Linear Trend	S2	3	0.99966
26	CA Linear Trend	S2	4	0.99609
27	CA Linear Trend	S2	5	0.97827
28	CA Linear Trend	S2	6	0.92205
29	CA Linear Trend	S2	7	0.80070
30	CA Linear Trend	S2	8	0.61011
31	CA Linear Trend	S2	9	0.38989
32	CA Linear Trend	S2	10	0.19930
33	CA Linear Trend	S2	11	0.07795
34	CA Linear Trend	S2	12	0.02173
35	CA Linear Trend	S2	13	0.00391
36	CA Linear Trend	S2	14	0.00034
37	CA Linear Trend	S2	15	0.00000
38	CA Linear Trend	S2	16	0.00000

Example 86.2: Freeman-Tukey and *t* Tests with Bootstrap Resampling

(View the complete code for this example (multex2.sas) in the example repository.)

The data for this example are the same as for Example 86.1, except that a continuous variable T, which indicates the time of death of the animal, has been added.

```
data a;
   input S1 S2 T Dose @@;
   datalines;
0 1 104 1   1 0  80 1   0 1 104 1
0 1 104 1   0 1 100 1   1 0 104 1
1 0  85 2   1 0  60 2   0 1  89 2
1 0  96 2   0 1  96 2   1 0  99 2
1 0  60 3   1 0  50 3   1 0  80 3
0 1  98 3   0 1  99 3   1 0  50 3
;

proc multtest data=a bootstrap nsample=10000 seed=37081 outsamp=res;
   test ft(S1 S2 / lowertailed) mean(T / lowertailed);
   class Dose;
   contrast 'Linear Trend' 0 1 2;
run;

proc print data=res(obs=36);
run;
```

The BOOTSTRAP option in the PROC MULTTEST statement requests bootstrap resampling, and NSAMPLE=10000 requests 10,000 bootstrap samples. The SEED=37081 option provides a starting value for the random number generator. The OUTSAMP=res option creates an output SAS data set res containing the 10,000 bootstrap samples.

The TEST statement specifies the Freeman-Tukey test for S1 and S2 and specifies the *t* test for T. Both tests are lower-tailed. The grouping variable in the CLASS statement is Dose, and the coefficients across the levels of Dose are 0, 1, and 2, as specified in the CONTRAST statement. The PROC PRINT statement displays the first 36 observations of the res data set containing the bootstrap samples.

The results from this analysis are listed in Output 86.2.1 through Output 86.2.5.

The "Model Information" table in Output 86.2.1 corresponds to the specifications in the invocation of PROC MULTTEST.

Output 86.2.1 FT and t tests with Bootstrap Resampling

The Multtest Procedure

Model Information	
Test for discrete variables	Freeman-Tukey
Test for continuous variables	Mean t-test
Degrees of Freedom Method	Pooled
Tails for discrete tests	Lower-tailed
Tails for continuous tests	Lower-tailed
Strata weights	None
P-value adjustment	Bootstrap
Center continuous variables	Yes
Number of resamples	10000
Seed	37081

The "Contrast Coefficients" table in Output 86.2.2 shows the coefficients from the CONTRAST statement after centering, and they model a linear trend.

Output 86.2.2 Contrast Coefficients

Contrast Coefficients			
		Dose	
Contrast	**1**	**2**	**3**
Linear Trend Centered	-1	0	1

The summary statistics are displayed in Output 86.2.3. The values for the discrete variables S1 and S2 are the same as those from Example 86.1. The mean, standard deviation, and sample size for the continuous variable T at each level of Dose are displayed in the "Continuous Variable Tabulations" table.

Output 86.2.3 Summary Statistics

Discrete Variable Tabulations				
Variable	**Dose**	**Count**	**NumObs**	**Percent**
S1	1	2	6	33.33
S1	2	4	6	66.67
S1	3	4	6	66.67
S2	1	4	6	66.67
S2	2	2	6	33.33
S2	3	2	6	33.33

Continuous Variable Tabulations				
Variable	**Dose**	**NumObs**	**Mean**	**Standard Deviation**
T	1	6	99.3333	9.6056
T	2	6	87.5000	14.4326
T	3	6	72.8333	22.7017

The p-values, displayed in Output 86.2.4, are from the Freeman-Tukey test for S1 and S2, and are from the t test for T.

Output 86.2.4 p-Values

	p-Values		
Variable	**Contrast**	**Raw**	**Bootstrap**
S1	Linear Trend	0.8547	1.0000
S2	Linear Trend	0.1453	0.4605
T	Linear Trend	0.0070	0.0281

The Raw column in Output 86.2.4 contains the results from the tests on the original data, while the Bootstrap column contains the bootstrap resampled adjustment to raw_p. Note that the adjusted p-values are larger than the raw p-values for all three variables. The adjusted p-values more accurately reflect the correlation of the raw p-values, the small size of the data, and the multiple testing.

Output 86.2.5 displays the first 36 observations of the SAS data set resulting from the OUTSAMP=RES option in the PROC MULTTEST statement. The entire data set has 180,000 observations, which is 10,000 times the number of observations in the data set.

Output 86.2.5 Resampling Data Set

Obs	_sample_	_class_	_obs_	S1	S2	T
1	1	1	17	0	1	26.1667
2	1	1	8	1	0	-27.5000
3	1	1	5	0	1	0.6667
4	1	1	9	0	1	1.5000
5	1	1	7	1	0	-2.5000
6	1	1	3	0	1	4.6667
7	1	2	12	1	0	11.5000
8	1	2	12	1	0	11.5000
9	1	2	14	1	0	-22.8333
10	1	2	17	0	1	26.1667
11	1	2	1	0	1	4.6667
12	1	2	15	1	0	7.1667
13	1	3	4	0	1	4.6667
14	1	3	17	0	1	26.1667
15	1	3	14	1	0	-22.8333
16	1	3	15	1	0	7.1667
17	1	3	15	1	0	7.1667
18	1	3	6	1	0	4.6667
19	2	1	6	1	0	4.6667
20	2	1	17	0	1	26.1667
21	2	1	8	1	0	-27.5000
22	2	1	13	1	0	-12.8333
23	2	1	9	0	1	1.5000
24	2	1	8	1	0	-27.5000
25	2	2	9	0	1	1.5000
26	2	2	18	1	0	-22.8333
27	2	2	15	1	0	7.1667
28	2	2	14	1	0	-22.8333
29	2	2	9	0	1	1.5000
30	2	2	17	0	1	26.1667
31	2	3	16	0	1	25.1667
32	2	3	11	0	1	8.5000
33	2	3	14	1	0	-22.8333
34	2	3	18	1	0	-22.8333
35	2	3	18	1	0	-22.8333
36	2	3	10	1	0	8.5000

The _sample_ variable is the sample indicator and _class_ indicates the resampling group—that is, the level of the CLASS variable Dose assigned to the new observation. The number of the observation in the original data set is represented by _obs_. Also listed are the values of the original test variables, S1 and S2, and the mean-centered values of T.

Example 86.3: Peto Mortality-Prevalence Test

(View the complete code for this example (multex3.sas) in the example repository.)

This example illustrates the use of the Peto mortality-prevalence test. The test is a combination of analyses about the prevalence of incidental tumors in the population and mortality due to fatal tumors.

In the following data set, each observation represents an animal. The variables S1–S3 are three tumor types, with a value of 0 indicating no tumor, 1 indicating an incidental (nonlethal) tumor, and 2 indicating a lethal tumor. The time variable T indicates the time of death of the animal, a strata variable B is constructed from T, and the grouping variable Dose is drug dosage.

```
data a;
   input S1-S3 T Dose @@;
   if T<=90 then B=1; else B=2;
   datalines;
0 0 0 104 0   2 0 1  80 0   0 0 1 104 0
0 0 0 104 0   0 2 0 100 0   1 0 0 104 0
2 0 0  85 1   2 1 0  60 1   0 1 0  89 1
2 0 1  96 1   0 0 0  96 1   2 0 1  99 1
2 1 1  60 2   2 0 0  50 2   2 0 1  80 2
0 0 2  98 2   0 0 1  99 2   2 1 1  50 2
;

proc multtest data=a notables out=p stepsid;
   test peto(S1-S3 / permutation=20 time=T uppertailed);
   class Dose;
   strata B;
   contrast 'mort-prev' 0  1  2;
run;
proc print data=p;
run;
```

The NOTABLES option in the PROC MULTTEST statement suppresses the display of the summary statistics for each variable. The OUT= option creates an output SAS data set p containing all *p*-values and intermediate statistics. The STEPSID option is used to adjust the *p*-values.

The TEST statement specifies an upper-tailed Peto test for S1–S3. The mortality strata are defined by TIME=T, the death times. The CLASS statement contains the grouping variable Dose. The prevalence strata are defined by the STRATA statement as the blocking variable B. The CONTRAST statement lists the default linear trend coefficients. The PROC PRINT statement displays the requested *p*-value data set.

The results from this analysis are listed in Output 86.3.1 through Output 86.3.4.

The "Model Information" table in Output 86.3.1 displays information corresponding to the PROC MULTTEST invocation. In this case the totals for all prevalence and fatality strata are less than 20, so exact permutation tests are used everywhere, and the STEPSID adjustments are computed from these permutation distributions.

Output 86.3.1 Peto Test

The Multtest Procedure

Model Information	
Test for discrete variables	Peto
Exact permutation distribution used	Everywhere
Tails for discrete tests	Upper-tailed
Strata weights	Sample size
P-value adjustment	Stepdown Sidak

The contrast trend coefficients are listed in Output 86.3.2. They happen to be the same as the levels of the Dose variable.

Output 86.3.2 Contrast Coefficients

Contrast Coefficients			
	Dose		
Contrast	**0**	**1**	**2**
mort-prev	0	1	2

In the "*p*-Values" table in Output 86.3.3, the *p*-values for the Peto tests are listed in the Raw column, and the step-down Šidák adjusted *p*-values are in the Stepdown Šidák column.

Output 86.3.3 *p*-Values

p-Values			
Variable	**Contrast**	**Raw**	**Stepdown Sidak**
S1	mort-prev	0.0681	0.0814
S2	mort-prev	0.5000	0.5000
S3	mort-prev	0.0363	0.0781

Significant *p*-values in the preceding table support the claim that higher dosage levels lead to higher mortality and prevalence. The raw Peto test is significant at the 5% level for S3, but the adjusted S3 test is no longer significant at 5%. The raw and adjusted *p*-values for S2 are the same because of the step-down technique.

The OUT= data set is displayed in Output 86.3.4.

Output 86.3.4 OUT= Data Set

Obs	_test_	_var_	_contrast_	_strat_	_tstrat_	_value_	_exp_	_se_	raw_p	stpsid_p
1	PETO	S1	mort-prev	1	0	0	0.00000	0.00000	.	.
2	PETO	S1	mort-prev	2	0	0	0.62500	0.85696	.	.
3	PETO	S1	mort-prev	50	1	4	2.00000	1.12022	.	.
4	PETO	S1	mort-prev	60	1	3	1.75000	1.06654	.	.
5	PETO	S1	mort-prev	80	1	2	1.57143	1.04978	.	.
6	PETO	S1	mort-prev	85	1	1	0.75000	0.72169	.	.
7	PETO	S1	mort-prev	96	1	1	0.70000	0.78102	.	.
8	PETO	S1	mort-prev	98	1	0	0.00000	0.00000	.	.
9	PETO	S1	mort-prev	99	1	1	0.42857	0.72843	.	.
10	PETO	S1	mort-prev	100	1	0	0.00000	0.00000	.	.
11	PETO	S2	mort-prev	1	0	6	5.50000	1.05221	.	.
12	PETO	S2	mort-prev	2	0	0	0.00000	0.00000	.	.
13	PETO	S2	mort-prev	50	1	0	0.00000	0.00000	.	.
14	PETO	S2	mort-prev	60	1	0	0.00000	0.00000	.	.
15	PETO	S2	mort-prev	80	1	0	0.00000	0.00000	.	.
16	PETO	S2	mort-prev	85	1	0	0.00000	0.00000	.	.
17	PETO	S2	mort-prev	96	1	0	0.00000	0.00000	.	.
18	PETO	S2	mort-prev	98	1	0	0.00000	0.00000	.	.
19	PETO	S2	mort-prev	99	1	0	0.00000	0.00000	.	.
20	PETO	S2	mort-prev	100	1	0	0.00000	0.00000	.	.
21	PETO	S3	mort-prev	1	0	6	5.50000	1.05221	.	.
22	PETO	S3	mort-prev	2	0	4	2.22222	1.08298	.	.
23	PETO	S3	mort-prev	50	1	0	0.00000	0.00000	.	.
24	PETO	S3	mort-prev	60	1	0	0.00000	0.00000	.	.
25	PETO	S3	mort-prev	80	1	0	0.00000	0.00000	.	.
26	PETO	S3	mort-prev	85	1	0	0.00000	0.00000	.	.
27	PETO	S3	mort-prev	96	1	0	0.00000	0.00000	.	.
28	PETO	S3	mort-prev	98	1	2	0.62500	0.85696	.	.
29	PETO	S3	mort-prev	99	1	0	0.00000	0.00000	.	.
30	PETO	S3	mort-prev	100	1	0	0.00000	0.00000	.	.
31	PETO	S1	mort-prev	.	.	12	7.82500	2.42699	0.06808	0.08140
32	PETO	S2	mort-prev	.	.	6	5.50000	1.05221	0.50000	0.50000
33	PETO	S3	mort-prev	.	.	12	8.34722	1.73619	0.03627	0.07811

The first 30 observations correspond to intermediate statistics used to compute the Peto p-values. The _test_ variable lists the name of the test, the _var_ variable lists the name of the TEST variables, and the _contrast_ variable lists the CONTRAST label. The _strat_ variable lists the level of the STRATA variable, and the _tstrat_ variable indicates whether or not the stratum corresponds to values of the TIME= variable. The _value_ variable is the observed contrast for a stratum, and the _exp_ variable is its expected value. The variable _se_ contains the square root of the variance terms summed to form the denominator of the Peto statistics.

The final three observations correspond to the three Peto tests, with their p-values listed under the raw_p variable. The stpsid_p variable contains the step-down Šidák-adjusted p-values.

Example 86.4: Fisher Test with Permutation Resampling

(View the complete code for this example (multex4.sas) in the example repository.)

The following data, from Brown and Fears (1981), are the results of an 80-week carcinogenesis bioassay with female mice. Six tissue sites are examined at necropsy; 1 indicates the presence of a tumor and 0 the absence. A frequency variable Freq is included. A control and four different doses of a drug (in parts per milliliter) make up the levels of the grouping variable Dose.

```
data a;
   input Liver Lung Lymph Cardio Pitui Ovary Freq Dose$ @@;
   datalines;
1 0 0 0 0 0 8  CTRL   0 1 0 0 0 0 7  CTRL   0 0 1 0 0 0 6  CTRL
0 0 0 1 0 0 1  CTRL   0 0 0 0 0 1 2  CTRL   1 1 0 0 0 0 4  CTRL
1 0 1 0 0 0 1  CTRL   1 0 0 0 0 1 1  CTRL   0 1 1 0 0 0 1  CTRL
0 0 0 0 0 0 18 CTRL
1 0 0 0 0 0 9  4PPM   0 1 0 0 0 0 4  4PPM   0 0 1 0 0 0 7  4PPM
0 0 0 1 0 0 1  4PPM   0 0 0 0 1 0 2  4PPM   0 0 0 0 0 1 1  4PPM
1 1 0 0 0 0 4  4PPM   1 0 1 0 0 0 3  4PPM   1 0 0 0 1 0 1  4PPM
0 1 1 0 0 0 1  4PPM   0 1 0 1 0 0 1  4PPM   1 0 1 1 0 0 1  4PPM
0 0 0 0 0 0 15 4PPM
1 0 0 0 0 0 8  8PPM   0 1 0 0 0 0 3  8PPM   0 0 1 0 0 0 6  8PPM
0 0 0 1 0 0 3  8PPM   1 1 0 0 0 0 1  8PPM   1 0 1 0 0 0 2  8PPM
1 0 0 1 0 0 1  8PPM   1 0 0 0 1 0 1  8PPM   1 1 0 1 0 0 2  8PPM
1 1 0 0 0 1 2  8PPM   0 0 0 0 0 0 19 8PPM
1 0 0 0 0 0 4  16PPM  0 1 0 0 0 0 2  16PPM  0 0 1 0 0 0 9  16PPM
0 0 0 0 1 0 1  16PPM  0 0 0 0 0 1 1  16PPM  1 1 0 0 0 0 4  16PPM
1 0 1 0 0 0 1  16PPM  0 1 1 0 0 0 1  16PPM  0 1 0 1 0 0 1  16PPM
0 1 0 0 0 1 1  16PPM  0 0 1 1 0 0 1  16PPM  0 0 1 0 1 0 1  16PPM
1 1 1 0 0 0 2  16PPM  0 0 0 0 0 0 14 16PPM
1 0 0 0 0 0 8  50PPM  0 1 0 0 0 0 4  50PPM  0 0 1 0 0 0 8  50PPM
0 0 0 1 0 0 1  50PPM  0 0 0 0 0 1 4  50PPM  1 1 0 0 0 0 3  50PPM
1 0 1 0 0 0 1  50PPM  0 1 1 0 0 0 1  50PPM  0 1 0 0 1 1 1  50PPM
0 0 0 0 0 0 19 50PPM
;

proc multtest data=a order=data notables out=p
              permutation nsample=1000 seed=764511;
   test fisher(Liver Lung Lymph Cardio Pitui Ovary /
               lowertailed);
   class Dose;
   freq Freq;
run;
proc print data=p;
run;
```

In the PROC MULTTEST statement, the ORDER=DATA option is required to keep the levels of Dose in the order in which they appear in the data set. Without this option, the levels are sorted by their formatted value, resulting in an alphabetic ordering. The NOTABLES option suppresses the display of summary statistics, and the OUT= option produces an output data set p containing the *p*-values. The PERMUTATION option specifies permutation resampling, NSAMPLE=1000 requests 1000 samples, and SEED=764511 option provides a starting value for the random number generator. You should specify a seed if you need to duplicate

resampling results.

To test for higher rates of tumor occurrence in the treatment groups compared to the control group, the LOWERTAILED option is specified in the FISHER option of the TEST statement to produce a lower-tailed Fisher exact test for the six tissue sites. The Fisher test is appropriate for comparing a treatment and a control, but multiple testing can be a problem. Brown and Fears (1981) use a multivariate permutation to evaluate the entire collection of tests. PROC MULTTEST adjusts the *p*-values by simulation.

The treatments make up the levels of the grouping variable Dose, listed in the CLASS statement. Since no CONTRAST statement is specified, PROC MULTTEST uses the default pairwise contrasts with the first level of Dose. The FREQ statement is used since these are summary data containing frequency counts of occurrences.

The results from this analysis are listed in Output 86.4.1 through Output 86.4.4. First, the PROC MULTTEST specifications are displayed in Output 86.4.1.

Output 86.4.1 Fisher Test with Permutation Resampling

The Multtest Procedure

Model Information	
Test for discrete variables	Fisher
Tails for discrete tests	Lower-tailed
Strata weights	None
P-value adjustment	Permutation
Number of resamples	1000
Seed	764511

The default contrasts for the Fisher test are displayed in Output 86.4.2. Note that each dose is compared with the control.

Output 86.4.2 Default Contrast Coefficients

Contrast Coefficients					
	Dose				
Contrast	**CTRL**	**4PPM**	**8PPM**	**16PPM**	**50PPM**
CTRL vs. 4PPM	1	-1	0	0	0
CTRL vs. 8PPM	1	0	-1	0	0
CTRL vs. 16PPM	1	0	0	-1	0
CTRL vs. 50PPM	1	0	0	0	-1

The “p-Values” table in Output 86.4.3 displays *p*-values for the Fisher exact tests and their permutation-based adjustments.

Output 86.4.3 p-Values

		p-Values	
Variable	**Contrast**	**Raw**	**Permutation**
Liver	CTRL vs. 4PPM	0.2828	0.9610
Liver	CTRL vs. 8PPM	0.3069	0.9670
Liver	CTRL vs. 16PPM	0.7102	1.0000
Liver	CTRL vs. 50PPM	0.7718	1.0000
Lung	CTRL vs. 4PPM	0.7818	1.0000
Lung	CTRL vs. 8PPM	0.8858	1.0000
Lung	CTRL vs. 16PPM	0.5469	0.9990
Lung	CTRL vs. 50PPM	0.8498	1.0000
Lymph	CTRL vs. 4PPM	0.2423	0.9280
Lymph	CTRL vs. 8PPM	0.5898	1.0000
Lymph	CTRL vs. 16PPM	0.0350	0.2680
Lymph	CTRL vs. 50PPM	0.4161	0.9930
Cardio	CTRL vs. 4PPM	0.3163	0.9710
Cardio	CTRL vs. 8PPM	0.0525	0.3710
Cardio	CTRL vs. 16PPM	0.4506	0.9960
Cardio	CTRL vs. 50PPM	0.7576	1.0000
Pitui	CTRL vs. 4PPM	0.1250	0.7540
Pitui	CTRL vs. 8PPM	0.4948	0.9970
Pitui	CTRL vs. 16PPM	0.2157	0.9080
Pitui	CTRL vs. 50PPM	0.5051	0.9970
Ovary	CTRL vs. 4PPM	0.9437	1.0000
Ovary	CTRL vs. 8PPM	0.8126	1.0000
Ovary	CTRL vs. 16PPM	0.7760	1.0000
Ovary	CTRL vs. 50PPM	0.3689	0.9930

As noted by Brown and Fears, only one of the 24 tests is significant at the 5% level (Lymph, CTRL vs. 16PPM). Brown and Fears report a 12% chance of observing at least one significant raw p-value for 16PPM and a 9% chance of observing at least one significant raw p-value for Lymph (both at the 5% level). Adjusted p-values exhibit much lower chances of false significances. For this example, none of the adjusted p-values are close to significant.

The OUT= data set is displayed in Output 86.4.4.

Output 86.4.4 OUT= Data Set

Obs	_test_	_var_	_contrast_	_xval_	_mval_	_yval_	_nval_	raw_p	perm_p	sim_se
1	FISHER	Liver	CTRL vs. 4PPM	14	49	18	50	0.28282	0.961	0.006122
2	FISHER	Liver	CTRL vs. 8PPM	14	49	17	48	0.30688	0.967	0.005649
3	FISHER	Liver	CTRL vs. 16PPM	14	49	11	43	0.71022	1.000	0.000000
4	FISHER	Liver	CTRL vs. 50PPM	14	49	12	50	0.77175	1.000	0.000000
5	FISHER	Lung	CTRL vs. 4PPM	12	49	10	50	0.78180	1.000	0.000000
6	FISHER	Lung	CTRL vs. 8PPM	12	49	8	48	0.88581	1.000	0.000000
7	FISHER	Lung	CTRL vs. 16PPM	12	49	11	43	0.54685	0.999	0.000999
8	FISHER	Lung	CTRL vs. 50PPM	12	49	9	50	0.84978	1.000	0.000000
9	FISHER	Lymph	CTRL vs. 4PPM	8	49	12	50	0.24228	0.928	0.008174
10	FISHER	Lymph	CTRL vs. 8PPM	8	49	8	48	0.58977	1.000	0.000000
11	FISHER	Lymph	CTRL vs. 16PPM	8	49	15	43	0.03498	0.268	0.014006
12	FISHER	Lymph	CTRL vs. 50PPM	8	49	10	50	0.41607	0.993	0.002636
13	FISHER	Cardio	CTRL vs. 4PPM	1	49	3	50	0.31631	0.971	0.005307
14	FISHER	Cardio	CTRL vs. 8PPM	1	49	6	48	0.05254	0.371	0.015276
15	FISHER	Cardio	CTRL vs. 16PPM	1	49	2	43	0.45061	0.996	0.001996
16	FISHER	Cardio	CTRL vs. 50PPM	1	49	1	50	0.75758	1.000	0.000000
17	FISHER	Pitui	CTRL vs. 4PPM	0	49	3	50	0.12496	0.754	0.013619
18	FISHER	Pitui	CTRL vs. 8PPM	0	49	1	48	0.49485	0.997	0.001729
19	FISHER	Pitui	CTRL vs. 16PPM	0	49	2	43	0.21572	0.908	0.009140
20	FISHER	Pitui	CTRL vs. 50PPM	0	49	1	50	0.50505	0.997	0.001729
21	FISHER	Ovary	CTRL vs. 4PPM	3	49	1	50	0.94372	1.000	0.000000
22	FISHER	Ovary	CTRL vs. 8PPM	3	49	2	48	0.81260	1.000	0.000000
23	FISHER	Ovary	CTRL vs. 16PPM	3	49	2	43	0.77596	1.000	0.000000
24	FISHER	Ovary	CTRL vs. 50PPM	3	49	5	50	0.36889	0.993	0.002636

The _test_, _var_, and _contrast_ variables provide the TEST name, TEST variable, and CONTRAST label, respectively. The _xval_, _mval_, _yval_, and _nval_ variables contain the components used to compute the Fisher exact tests from the hypergeometric distribution. The raw_p variable contains the p-values from the Fisher exact tests, and the perm_p variable contains their permutation-based adjustments. The variable sim_se is the simulation standard error from the permutation resampling.

Example 86.5: Inputting Raw p-Values

(View the complete code for this example (multex5.sas) in the example repository.)

This example illustrates how to use PROC MULTTEST to multiplicity-adjust a collection of raw p-values obtained from some other source. This is a valuable option for those cases where PROC MULTTEST cannot compute the raw p-values directly. The data set a, which follows, contains the unadjusted p-values computed in Example 86.4. Note that the data set needs to have one variable containing the p-values, but the data set can contain other variables as well.

```
data a;
   input Test$ Raw_P @@;
   datalines;
test01  0.28282    test02  0.30688    test03  0.71022
test04  0.77175    test05  0.78180    test06  0.88581
```

```
test07  0.54685    test08  0.84978    test09  0.24228
test10  0.58977    test11  0.03498    test12  0.41607
test13  0.31631    test14  0.05254    test15  0.45061
test16  0.75758    test17  0.12496    test18  0.49485
test19  0.21572    test20  0.50505    test21  0.94372
test22  0.81260    test23  0.77596    test24  0.36889
;

proc multtest inpvalues=a holm hoc fdr;
run;
```

Note that the PROC MULTTEST statement is the only statement that can be specified with the *p*-value input mode. In this example, the raw *p*-values are adjusted by the Holm, Hochberg, and FDR methods.

The "P-Value Adjustment Information" table, displayed in Output 86.5.1, provides information about the requested adjustments and replaces the usual "Model Information" table. The adjusted *p*-values are displayed in Output 86.5.2

Output 86.5.1 Inputting Raw *p*-Values

The Multtest Procedure

P-Value Adjustment Information	
P-Value Adjustment	Stepdown Bonferroni
P-Value Adjustment	Hochberg
P-Value Adjustment	False Discovery Rate

Output 86.5.2 p-Values

	p-Values			
Test	Raw	Stepdown Bonferroni	Hochberg	False Discovery Rate
1	0.2828	1.0000	0.9437	0.9243
2	0.3069	1.0000	0.9437	0.9243
3	0.7102	1.0000	0.9437	0.9243
4	0.7718	1.0000	0.9437	0.9243
5	0.7818	1.0000	0.9437	0.9243
6	0.8858	1.0000	0.9437	0.9243
7	0.5469	1.0000	0.9437	0.9243
8	0.8498	1.0000	0.9437	0.9243
9	0.2423	1.0000	0.9437	0.9243
10	0.5898	1.0000	0.9437	0.9243
11	0.0350	0.8395	0.8395	0.6305
12	0.4161	1.0000	0.9437	0.9243
13	0.3163	1.0000	0.9437	0.9243
14	0.0525	1.0000	0.9437	0.6305
15	0.4506	1.0000	0.9437	0.9243
16	0.7576	1.0000	0.9437	0.9243
17	0.1250	1.0000	0.9437	0.9243
18	0.4949	1.0000	0.9437	0.9243
19	0.2157	1.0000	0.9437	0.9243
20	0.5051	1.0000	0.9437	0.9243
21	0.9437	1.0000	0.9437	0.9437
22	0.8126	1.0000	0.9437	0.9243
23	0.7760	1.0000	0.9437	0.9243
24	0.3689	1.0000	0.9437	0.9243

Note that the adjusted p-values for the Hochberg method are less than or equal to those for the Holm (Step-down Bonferroni) method. In turn, the adjusted p-values for the FDR method (False Discovery Rate) are less than or equal to those for the Hochberg method. These comparisons hold generally for all p-value configurations. The FDR method controls the false discovery rate and not the familywise error rate. The Hochberg method controls the familywise error rate under independence. The Holm method controls the familywise error rate without assuming independence.

Example 86.6: Adaptive Adjustments and ODS Graphics

(View the complete code for this example (multex6.sas) in the example repository.)

An experiment was performed using Affymetrix gene chips on the CD4 lymphocyte white blood cells of patients with and without a hereditary allergy (atopy) and possibly with asthma. The Asthma-Atopy microarray data set and analysis are discussed in Gibson and Wolfinger (2004): a one-way ANOVA model of the log2mas5 variable ($\log_2$(MAS 5.0 summary statistics) is fit against a classification variable trt describing different asthma-atopy combinations in the patients, and the least squares means of the trt variable are computed.

For this example, a 1% random sample of least squares means having p-values exceeding 1E–6 is taken. The resulting data are recorded in the test data set, where the Probe_Set_ID variable identifies the probe and the Probt variable contains the p-values for the $m = 121$ tests, as follows:

```
data test;
   length Probe_Set_ID $9.;
   input Probe_Set_ID $ Probt @@;
   datalines;
200973_s_ .963316  201059_at .462754  201563_at .000409  201733_at .000819
201951_at .000252  202944_at .106550  203107_x_ .040396  203372_s_ .010911
203469_s_ .987234  203641_s_ .019296  203795_s_ .002276  204055_s_ .002328
205020_s_ .008628  205199_at .608129  205373_at .005209  205384_at .742381
205428_s_ .870533  205653_at .621671  205686_s_ .396440  205760_s_ .000002
206032_at .024661  206159_at .997627  206223_at .003702  206398_s_ .191682
206623_at .010030  206852_at .000004  207072_at .000214  207371_at .000013
207789_s_ .023623  207861_at .000002  207897_at .000007  208022_s_ .251999
208086_s_ .000361  208406_s_ .040182  208464_at .161468  209055_s_ .529824
209125_at .142276  209369_at .240079  209748_at .071750  209894_at .000042
209906_at .223282  210130_s_ .192187  210199_at .101623  210477_x_ .300038
210491_at .000078  210531_at .000784  210734_x_ .202931  210755_at .009644
210782_x_ .000011  211320_s_ .022896  211329_x_ .486869  211362_s_ .881798
211369_at .000030  211399_at .000008  211572_s_ .269788  211647_x_ .001301
213072_at .005019  213143_at .008711  213238_at .004824  213391_at .316133
213468_at .000172  213636_at .097133  213823_at .001678  213854_at .001921
213976_at .000299  214006_s_ .014616  214063_s_ .000361  214407_x_ .609880
214445_at .000009  214570_x_ .000002  214648_at .001255  214684_at .288156
214991_s_ .006695  215012_at .000499  215117_at .000136  215201_at .045235
215304_at .000816  215342_s_ .973786  215392_at .112937  215557_at .000007
215608_at .006204  215935_at .000027  215980_s_ .037382  216010_x_ .000354
216051_x_ .000003  216086_at .002310  216092_s_ .000056  216511_s_ .294776
216733_s_ .004823  216747_at .002902  216874_at .000117  216969_s_ .001614
217133_x_ .056851  217198_x_ .169196  217557_s_ .002966  217738_at .000005
218601_at .023817  218818_at .027554  219302_s_ .000039  219441_s_ .000172
219574_at .193737  219612_s_ .000075  219697_at .046476  219700_at .003049
219945_at .000066  219964_at .000684  220234_at .130064  220473_s_ .000017
220575_at .030223  220633_s_ .058460  220925_at .252465  221256_s_ .721731
221314_at .002307  221589_s_ .001810  221995_s_ .350859  222071_s_ .000062
222113_s_ .000023  222208_s_ .100961  222303_at .049265  37226_at  .000749
60474_at  .000423
run;
```

The following statements adjust the p-values in the test data set by using the adaptive adjustments (ADAP-

TIVEHOLM, ADAPTIVEHOCHBERG, ADAPTIVEFDR, and PFDR), which require an estimate of the number of true null hypotheses ($\hat{m}_0$) or proportion of true null hypotheses ($\hat{\pi}_0$). This example illustrates some of the features and graphics for computing and evaluating these estimates. The NOPVALUE option is specified to suppress the display of the "p-Values" table.

```
ods graphics on;
proc multtest inpvalues(Probt)=test plots=all seed=518498000
              aholm ahoc afdr pfdr(positive) nopvalue;
   id Probe_Set_ID;
run;
ods graphics off;
```

Output 86.6.1 lists the requested p-value adjustments, along with the selected value of the "Lambda" tuning parameter and the seed (specified with the SEED= option) used in the bootstrap method of estimating the number of true null hypotheses. The "Lambda Values" table lists the estimated number of true nulls for each value of λ, where you can see that the minimum MSE (0.002315) occurs at $\lambda = 0.4$. Output 86.6.2 shows that the SPLINE method failed due to a large slope at $\lambda = 0.95$, so the bootstrap method is used and the MSE plot is displayed.

Output 86.6.1 p and Lambda Values

The Multtest Procedure

P-Value Adjustment Information	
P-Value Adjustment	Adaptive Holm
P-Value Adjustment	Adaptive Hochberg
P-Value Adjustment	Adaptive FDR
P-Value Adjustment	pFDR Q-Value
Lambda	0.4
Seed	518498000

Output 86.6.1 *continued*

Lambda Values			
Lambda	**MSE**	**NTrueNull Observed**	**NTrueNull Spline**
0	0.657880	121.000000	67.318707
0.050000	0.030212	43.157895	59.812885
0.100000	0.024897	41.111111	52.636271
0.150000	0.014904	36.470588	46.033846
0.200000	0.008580	32.500000	40.172642
0.250000	0.006476	30.666667	35.157768
0.300000	0.002719	25.714286	31.046105
0.350000	0.002471	24.615385	27.861153
0.400000	0.002378	23.333333	25.595089
0.450000	0.003285	25.454545	24.217908
0.500000	0.003036	24.000000	23.687690
0.550000	0.003567	24.444444	23.965745
0.600000	0.005813	27.500000	25.016579
0.650000	0.004118	22.857143	26.809774
0.700000	0.006647	26.666667	29.321876
0.750000	0.006260	24.000000	32.512203
0.800000	0.013242	30.000000	36.315191
0.850000	0.037624	40.000000	40.618909
0.900000	0.046906	40.000000	45.274355
0.950000	0.332183	80.000000	50.117369

Output 86.6.2 Tuning Parameter Plots

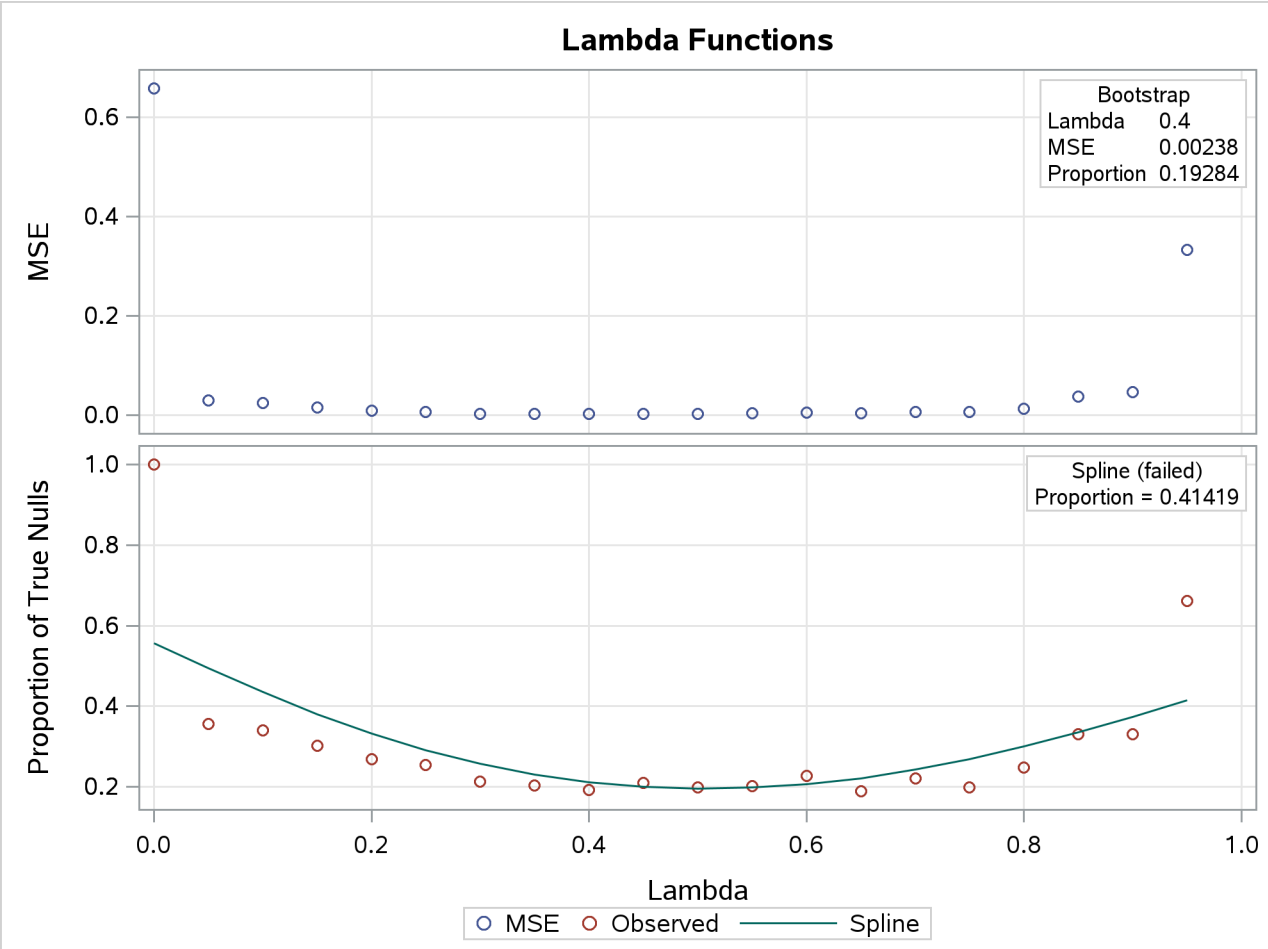

Output 86.6.3 also shows that the bootstrap estimate is used for the PFDR adjustment. The other adjustments have different default methods for estimating the number of true nulls.

Output 86.6.3 Adjustments and Their Default Estimation Method

Estimated Number of True Null Hypotheses			
P-Value Adjustment	**Method**	**Estimate**	**Proportion**
Adaptive Holm	**Decreased Slope**	26	0.21488
Adaptive Hochberg	**Decreased Slope**	26	0.21488
Adaptive FDR	**Lowest Slope**	43	0.35537
Positive FDR	**Bootstrap**	23.3333	0.19284

Output 86.6.4 displays the estimated number of true nulls $\hat{m}_0$ against a uniform probability plot of the unadjusted p-values (if the p-values are distributed uniformly, the points on the plot will all lie on a straight line). According to Schweder and Spjøtvoll (1982) and Hochberg and Benjamini (1990), the points on the left side of the plot should be approximately linear with slope $\frac{1}{m_0+1}$, so you can use this plot to evaluate whether your estimate of $\hat{m}_0$ seems reasonable.

Output 86.6.4 p-Value Distribution

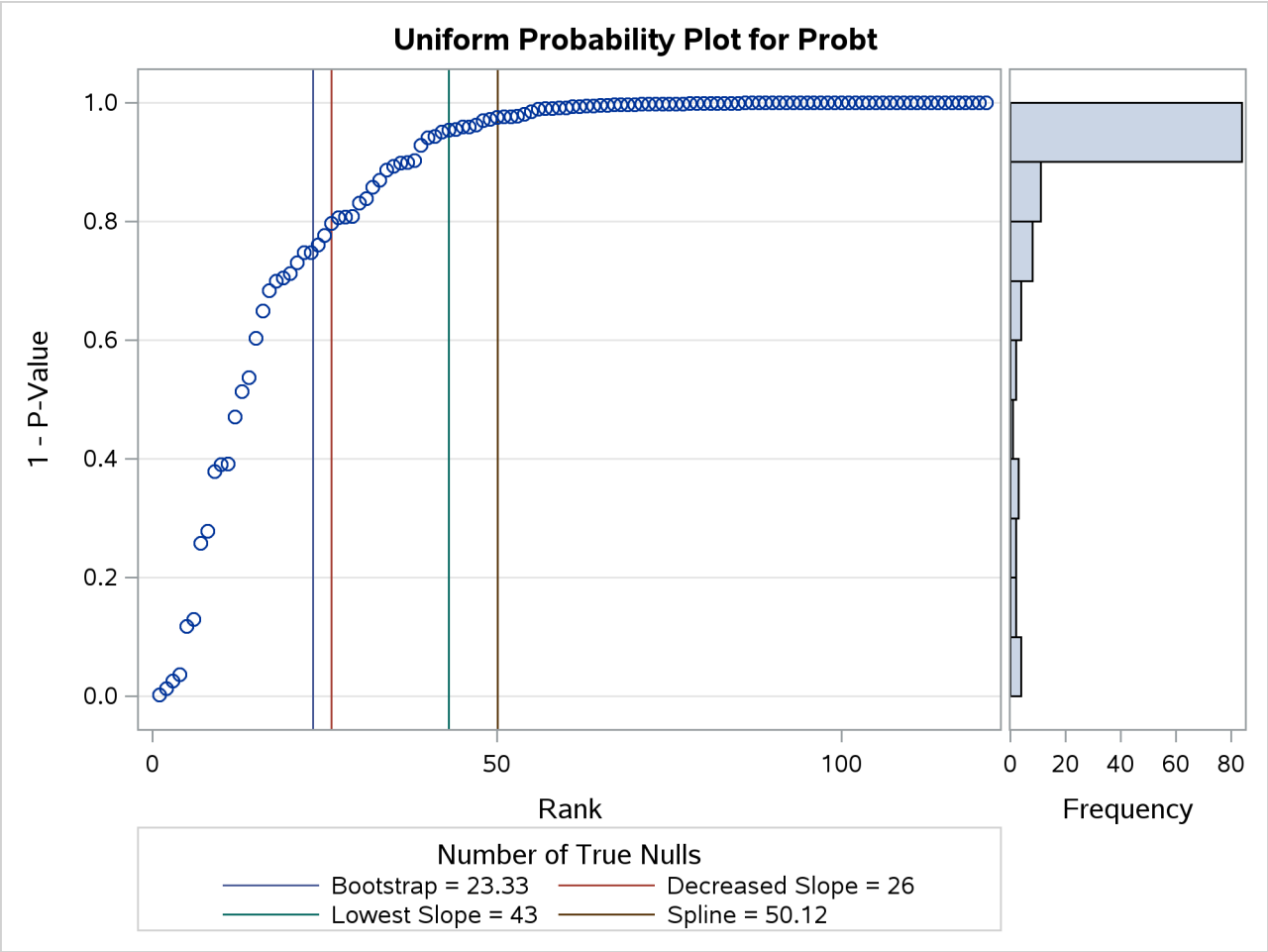

The NTRUENULL= option provides several methods for estimating the number of true null hypotheses; the following table displays each method and its estimate for this example:

NTRUENULL=	Estimate
BOOTSTRAP	23.3
DECREASESLOPE	26
KSTEST	35
LEASTSQUARES	28
LOWESTSLOPE	43
MEANDIFF	42
SPLINE	50.1

Another method of estimating the number of true null hypotheses fits a finite mixture model (mixing a uniform with a beta) to the distribution of the unadjusted p-values (Allison et al. 2002). Osborne (2006) provides the following PROC NLMIXED statements to fit this model:

```
proc nlmixed data=test;
   parameters pi0=0.5 a=.1 b=.1;
   pi1= 1-pi0;
   bounds 0 <= pi0 <= 1;
   loglikelihood= log(pi0+pi1*pdf('beta',Probt,a,b));
   model Probt ~ general(loglikelihood);
run;
```

You might have to change the initial parameter values in the PARAMETERS statement to achieve convergence; see Chapter 89, "The NLMIXED Procedure," for more information. This mixture model estimates $\hat{\pi}_0 = 0$, meaning that the distribution of p-values is completely specified by a single beta distribution. If the estimate were, say, $\hat{\pi}_0 = 0.10$, you could then specify it as follows:

```
proc multtest inpvalues(Probt)=test ptruenull=0.10
              aholm ahoc afdr pfdr(positive) nopvalue;
   id Probe_Set_ID;
run;
```

A plot of the unadjusted and adjusted p-values for each test is also produced. Due to the large number of tests and adjustments, the plot is not very informative and is not displayed here.

The top two plots in Output 86.6.5 show how the adjusted values compare with each other and the unadjusted p-values. The PFDR and AFDR adjustments are eventually smaller than the unadjusted p-values since they control the false discovery rate. The adaptive Holm and Hochberg adjustments are almost identical, so the adaptive Holm values are mostly obscured in all four plots. The plot of the Proportion Significant versus the Adjusted p-values tells you how many of the tests are significant for each cutoff, while the plot of the number of false positives (FPN) versus the Proportion Significant tells you how many false positives you can expect for that cutoff.

Output 86.6.5 Adjustment Diagnostics

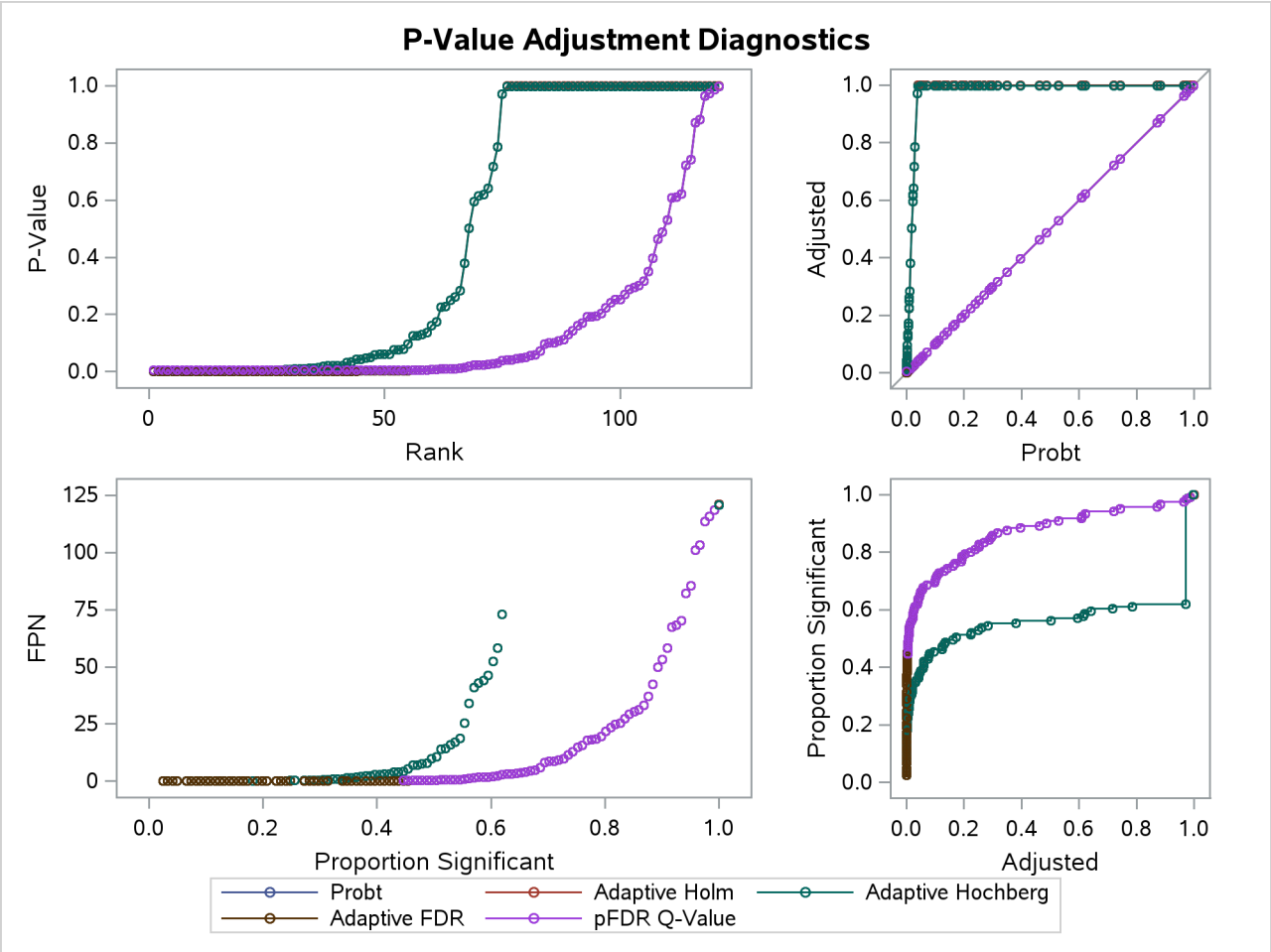

A Manhattan plot displays $-\log_{10}$ of the adjusted p-values, so the most significant tests stand out at the top of the plot. The default plot is not displayed here. The following statements create a Manhattan plot of the adaptive FDR p-values, with the most significant tests labeled with their observation number. The ID values are displayed on the X axis, and the VREF= option specifies the significance level. This plot is typically created with many more p-values, and special ODS Graphics options such as the LABELMAX= option might be required to display the graph. If memory usage is an issue, you might want to store your p-values and use the SGPLOT procedure to create a similar graph.

```
ods graphics on / labelmax=1000;
proc multtest inpvalues(Probt)=test afdr nopvalue
   plots=Manhattan(label=obs vref=0.0001);
   id Probe_Set_ID;
run;
ods graphics off;
```

Output 86.6.6 Manhattan Plot

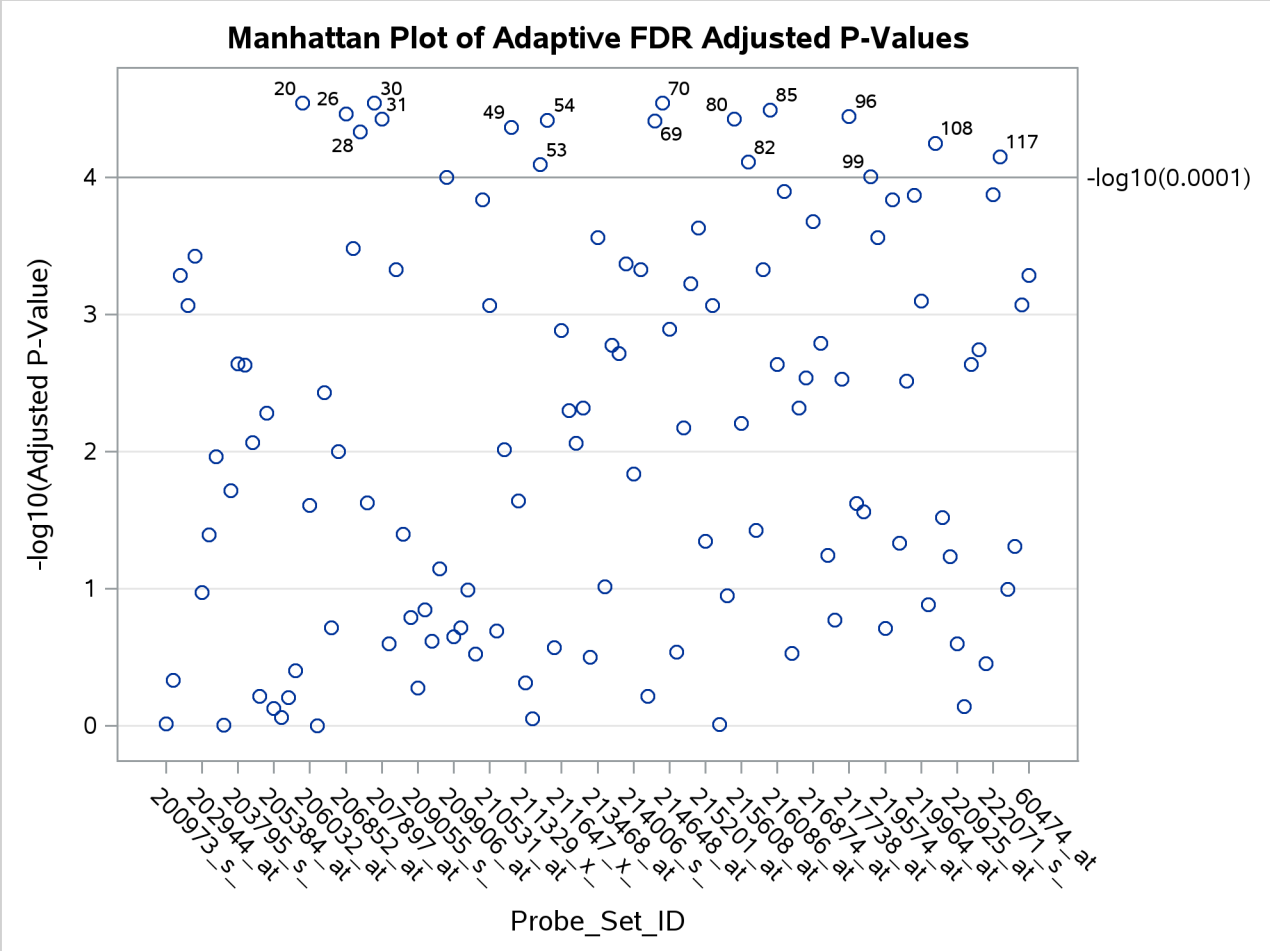

If you have a lot of tests, the "Raw and Adjusted *p*-Values" and "P-Value Adjustment Diagnostics" plots can be more informative if you suppress some of the tests. In the following statements, the SIGONLY=0.001 option selects tests with adjusted *p*-values < 0.001 for display. Output 86.6.7 displays tests with their "significant" adjusted *p*-values:

```
ods graphics on;
proc multtest inpvalues(Probt)=test plots(sigonly=0.001)=PByTest
              aholm ahoc afdr pfdr(positive) nopvalue;
run;
ods graphics off;
```

Output 86.6.7 Raw and Adjusted p-Values

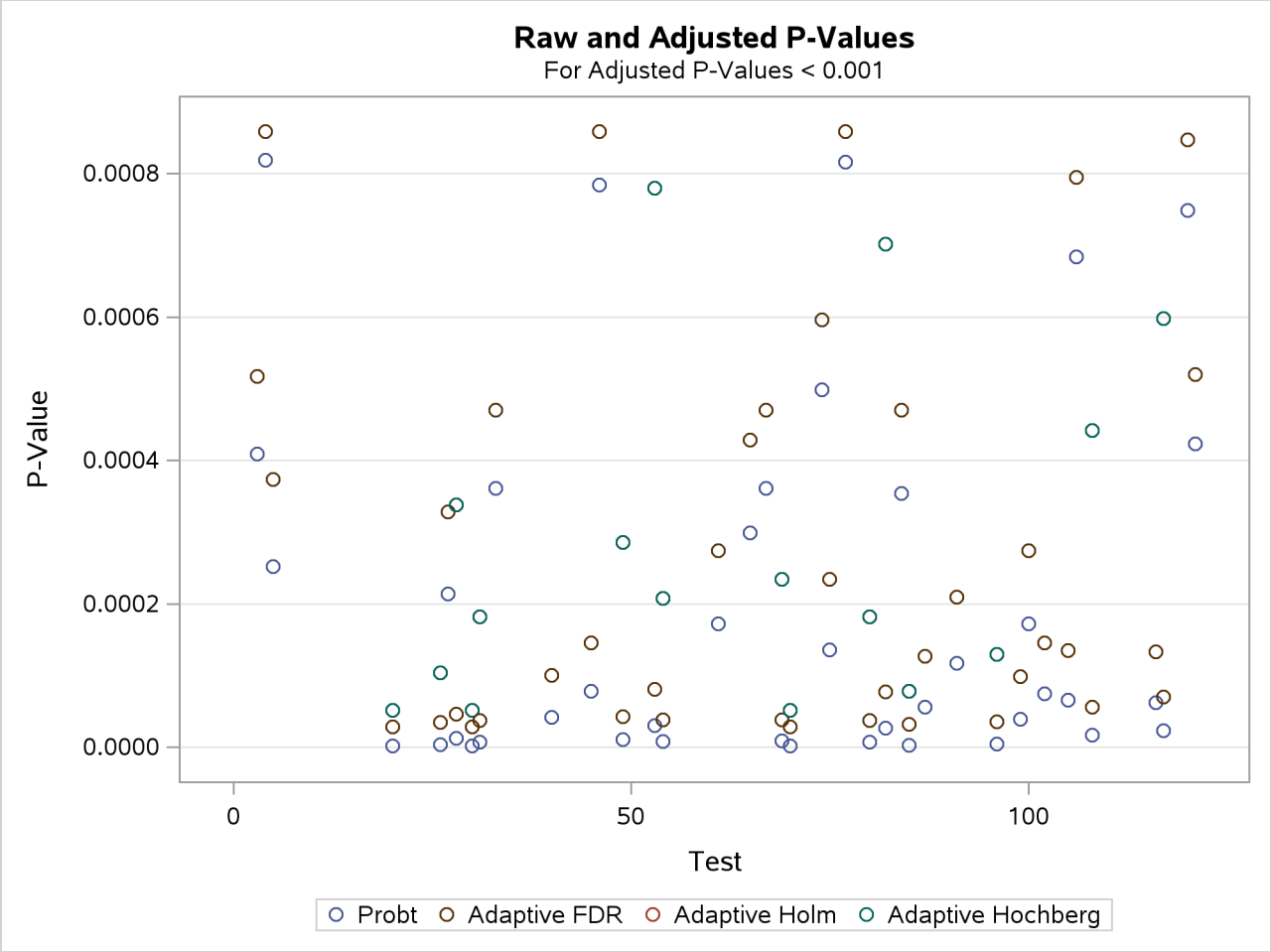

References

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Allison, D. B., Gadbury, G. L., Moonseong, H., Fernández, J. R., Lee, C., Prolla, T. A., and Weindruch, R. (2002). "A Mixture Model Approach for the Analysis of Microarray Gene Expression Data." *Computational Statistics and Data Analysis* 39:1–20.

Armitage, P. (1955). "Tests for Linear Trend in Proportions and Frequencies." *Biometrics* 11:375–386.

Benjamini, Y., and Hochberg, Y. (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society, Series B* 57:289–300.

Benjamini, Y., and Hochberg, Y. (2000). "On the Adaptive Control of the False Discovery Rate in Multiple Testing with Independent Statistics." *Journal of Educational and Behavioral Statistics* 25:60–83.

Benjamini, Y., Krieger, A. M., and Yekutieli, D. (2006). "Adaptive Linear Step-Up False Discovery Rate Controlling Procedures." *Biometrika* 93:491–507.

Benjamini, Y., and Yekutieli, D. (2001). "The Control of the False Discovery Rate in Multiple Testing under Dependency." *Annals of Statistics* 29:1165–1188.

Bickis, M., and Krewski, D. (1986). "Statistical Issues in the Analysis of the Long Term Carcinogenicity Bioassay in Small Rodents: An Empirical Evaluation of Statistical Decision Rules." Environmental Health Directorate, Health Protection Branch, Health and Welfare Canada, Ottawa, Ontario.

Brown, B. W., and Russell, K. (1997). "Methods Correcting for Multiple Testing: Operating Characteristics." *Statistics in Medicine* 16:2511–2528.

Brown, C. C., and Fears, T. R. (1981). "Exact Significance Levels for Multiple Binomial Testing with Application to Carcinogenicity Screens." *Biometrics* 37:763–774.

Cochran, W. G. (1954). "Some Methods for Strengthening the Common χ^2 Tests." *Biometrics* 10:417–451.

Dinse, G. E. (1985). "Testing for Trend in Tumor Prevalence Rates, Part 1: Nonlethal Tumors." *Biometrics* 41:751–770.

Dmitrienko, A., Molenberghs, G., Chuang-Stein, C., and Offen, W. (2005). *Analysis of Clinical Trials Using SAS: A Practical Guide*. Cary, NC: SAS Institute Inc.

Dudoit, S., Shaffer, J. P., and Boldrick, J. C. (2003). "Multiple Hypothesis Testing in Microarray Experiments." *Statistical Science* 18:71–103.

Freedman, D. A. (1981). "Bootstrapping Regression Models." *Annals of Statistics* 9:1218–1228.

Freeman, M. F., and Tukey, J. W. (1950). "Transformations Related to the Angular and the Square Root." *Annals of Mathematical Statistics* 21:607–611.

Gibson, G., and Wolfinger, R. D. (2004). "Gene Expression Profiling Using Mixed Models." In *Genetic Analysis of Complex Traits Using SAS*, edited by A. M. Saxton, 251–278. Cary, NC: SAS Institute Inc.

Good, I. J. (1987). "A Survey of the Use of the Fast Fourier Transform for Computing Distributions." *Journal of Statistical Computation and Simulation* 28:87–93.

Heyse, J., and Rom, D. (1988). "Adjusting for Multiplicity of Statistical Tests in the Analysis of Carcinogenicity Studies." *Biometrical Journal* 30:883–896.

Hochberg, Y. (1988). "A Sharper Bonferroni Procedure for Multiple Significance Testing." *Biometrika* 75:800–803.

Hochberg, Y., and Benjamini, Y. (1990). "More Powerful Procedures for Multiple Significance Testing." *Statistics in Medicine* 9:811–818.

Hochberg, Y., and Tamhane, A. C. (1987). *Multiple Comparison Procedures*. New York: John Wiley & Sons.

Hoel, D. G., and Walburg, H. E. (1972). "Statistical Analysis of Survival Experiments." *Journal of the National Cancer Institute* 49:361–372.

Holland, B. S., and Copenhaver, M. D. (1987). "An Improved Sequentially Rejective Bonferroni Test Procedure." *Biometrics* 43:417–424.

Holm, S. (1979). "A Simple Sequentially Rejective Multiple Test Procedure." *Scandinavian Journal of Statistics* 6:65–70.

Hommel, G. (1988). "A Comparison of Two Modified Bonferroni Procedures." *Biometrika* 75:383–386.

Hsueh, H., Chen, J. J., and Kodell, R. L. (2003). "Comparison of Methods for Estimating the Number of True Null Hypotheses in Multiplicity Testing." *Journal of Biopharmaceutical Statistics* 13:675–689.

Lagakos, S. W., and Louis, T. A. (1985). "The Statistical Analysis of Rodent Tumorigenicity Experiments." In *Toxicological Risk Assessment*, edited by D. B. Clayson, D. Krewski, and I. Munro, 144–163. Boca Raton, FL: CRC Press.

Liu, W. (1996). "Multiple Tests of a Non-hierarchical Finite Family of Hypotheses." *Journal of the Royal Statistical Society, Series B* 58:455–461.

Mantel, N. (1980). "Assessing Laboratory Evidence for Neoplastic Activity." *Biometrics* 36:381–399.

Mantel, N., and Haenszel, W. (1959). "Statistical Aspects of Analysis of Data from Retrospective Studies of Disease." *Journal of the National Cancer Institute* 22:719–748.

Marcus, R., Peritz, E., and Gabriel, K. R. (1976). "On Closed Testing Procedures with Special Reference to Ordered Analysis of Variance." *Biometrika* 63:655–660.

Miller, J. J. (1978). "The Inverse of the Freeman-Tukey Double Arcsine Transformation." *American Statistician* 32:138.

Osborne, J. A. (2006). "Estimating the False Discovery Rate Using SAS." In *Proceedings of the Thirty-First Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi31/190-31.pdf.

Pagano, M., and Tritchler, D. (1983). "On Obtaining Permutation Distributions in Polynomial Time." *Journal of the American Statistical Association* 78:435–440.

Peto, R., Pike, M. C., Day, N. E., Gray, R. G., Lee, P. N., Parish, S., Peto, J., Richards, S., and Wahrendorf, J. (1980). "Guidelines for Simple, Sensitive Significance Tests for Carcinogenic Effects in Long-Term Animal Experiments." In *Suppl. 2: Long-Term and Short-Term Screening Assays for Carcinogens—a Critical Appraisal*, 311–426. IARC Monographs on the Evaluation of Carcinogenic Risks to Humans. Lyon: International Agency for Research on Cancer.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical Recipes in C: The Art of Scientific Computing*. 2nd ed. Cambridge: Cambridge University Press.

Sarkar, S. K., and Chang, C.-K. (1997). "The Simes Method for Multiple Hypothesis Testing with Positively Dependent Test Statistics." *Journal of the American Statistical Association* 92:1601–1608.

Satterthwaite, F. E. (1946). "An Approximate Distribution of Estimates of Variance Components." *Biometrics Bulletin* 2:110–114.

Schweder, T., and Spjøtvoll, E. (1982). "Plots of *P*-Values to Evaluate Many Tests Simultaneously." *Biometrika* 69:493–502.

Shaffer, J. P. (1986). "Modified Sequentially Rejective Multiple Test Procedures." *Journal of the American Statistical Association* 81:826–831.

Šidák, Z. (1967). "Rectangular Confidence Regions for the Means of Multivariate Normal Distributions." *Journal of the American Statistical Association* 62:626–633.

Simes, R. J. (1986). "An Improved Bonferroni Procedure for Multiple Tests of Significance." *Biometrika* 73:751–754.

Soper, K. A., and Tonkonoh, N. (1993). "The Discrete Distribution Used for the Log-Rank Test Can Be Inaccurate." *Biometrical Journal* 35:291–298.

Storey, J. D. (2002). "A Direct Approach to False Discovery Rates." *Journal of the Royal Statistical Society, Series B* 64:479–498.

Storey, J. D., Taylor, J. E., and Siegmund, D. (2004). "Strong Control, Conservative Point Estimation, and Simultaneous Conservative Consistency of False Discovery Rates: A Unified Approach." *Journal of the Royal Statistical Society, Series B* 66:187–205.

Storey, J. D., and Tibshirani, R. (2003). "Statistical Significance for Genomewide Studies." *Proceedings of the National Academy of Sciences of the United States of America* 100:9440–9445.

Turkheimer, F. E., Smith, C. B., and Schmidt, K. (2001). "Estimation of the Number of 'True' Null Hypotheses in Multivariate Analysis of Neuroimaging Data." *NeuroImage* 13:920–930.

Westfall, P. H. (2005). "Combining P Values." In *Encyclopedia of Biostatistics*, 2nd ed., edited by P. Armitage and T. Colton. Chichester, UK: John Wiley & Sons.

Westfall, P. H., and Lin, Y. (1988). "Estimating Optimal Continuity Corrections in Run Time." In *Proceedings of the Statistical Computing Section*, 297–298. Alexandria, VA: American Statistical Association.

Westfall, P. H., and Soper, K. A. (1994). "Nonstandard Uses of PROC MULTTEST: Permutational Peto Tests; Permutational and Unconditional t and Binomial Tests." In *Proceedings of the Nineteenth Annual SAS Users Group International Conference*, 986–989. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI94/Sugi-94-173%20Westfall%20Soper.pdf.

Westfall, P. H., Tobias, R. D., Rom, D., Wolfinger, R. D., and Hochberg, Y. (1999). *Multiple Comparisons and Multiple Tests Using the SAS System*. Cary, NC: SAS Institute Inc.

Westfall, P. H., and Wolfinger, R. D. (1997). "Multiple Tests with Discrete Distributions." *American Statistician* 51:3–8.

Westfall, P. H., and Wolfinger, R. D. (2000). "Closed Multiple Testing Procedures and PROC MULTTEST." *Observations* (June):21 pages. Paper available at http://support.sas.com/kb/22/addl/fusion22950_1_multtest.pdf.

Westfall, P. H., and Young, S. S. (1989). "P-Value Adjustments for Multiple Tests in Multivariate Binomial Models." *Journal of the American Statistical Association* 84:780–786.

Westfall, P. H., and Young, S. S. (1993). *Resampling-Based Multiple Testing: Examples and Methods for* p*-Value Adjustment*. New York: John Wiley & Sons.

Yates, F. (1984). "Tests of Significance for 2×2 Contingency Tables." *Journal of the Royal Statistical Society, Series A* 147:426–463.

Yekutieli, D., and Benjamini, Y. (1999). "Resampling-Based False Discovery Rate Controlling Multiple Test Procedures for Correlated Test Statistics." *Journal of Statistical Planning and Inference* 82:171–196.

Chapter 87
The NESTED Procedure

Contents

Overview: NESTED Procedure **7116**
Contrasted with Other SAS Procedures 7116
Getting Started: NESTED Procedure **7117**
Reliability of Automobile Models 7117
Syntax: NESTED Procedure **7119**
PROC NESTED Statement 7119
BY Statement 7120
CLASS Statement 7120
VAR Statement 7121
Details: NESTED Procedure **7121**
Missing Values 7121
Unbalanced Data 7121
General Random-Effects Model 7121
Analysis of Covariation 7122
Error Terms in *F* Tests 7122
Computational Method 7123
Displayed Output 7123
ODS Table Names 7125
Example: NESTED Procedure **7125**
Example 87.1: Variability of Calcium Concentration in Turnip Greens 7125
References **7127**

Overview: NESTED Procedure

The NESTED procedure performs random-effects analysis of variance for data from an experiment with a nested (hierarchical) structure.[1] A random-effects model for data from a completely nested design with two factors has the form

$$y_{ijr} = \mu + \alpha_i + \beta_{ij} + \epsilon_{ijr}$$

where

y_{ijr}	is the value of the dependent variable observed at the *r*th replication with the first factor at its *i*th level and the second factor at its *j*th level.
μ	is the overall (fixed) mean of the sampling population.
$\alpha_i, \beta_{ij}, \epsilon_{ijr}$	are mutually uncorrelated random effects with zero means and respective variances σ_α^2, σ_β^2, and σ_ϵ^2 (the variance components).

This model is appropriate for an experiment with a multistage nested sampling design. An example of this is given in Example 87.1, where four turnip plants are randomly chosen (the first factor), then three leaves are randomly chosen from each plant (the second factor nested within the first), and then two samples are taken from each leaf (the different replications at fixed levels of the two factors).

Note that PROC NESTED is appropriate for models with only classification effects; it does not handle models that contain continuous covariates. For random effects models with covariates, use either the GLM or MIXED procedure.

Contrasted with Other SAS Procedures

The NESTED procedure performs a computationally efficient analysis of variance for data with a nested design, estimating the different components of variance and also testing for their significance if the design is balanced (see the section "Unbalanced Data" on page 7121). Although other procedures (such as GLM and MIXED) provide similar analyzes, PROC NESTED is both easier to use and computationally more efficient for this special type of design. This is especially true when the design involves a large number of factors, levels, or observations.

For example, to specify a four-factor completely nested design in the GLM procedure, you use the following form:

```
class a b c d;
model y=a b(a) c(a b) d(a b c);
```

However, to specify the same design in PROC NESTED, you simply use the following form:

[1] PROC NESTED is modeled after the General Purpose Nested Analysis of Variance program of the Dairy Cattle Research Branch of the United States Department of Agriculture. That program was originally written by M. R. Swanson, Statistical Reporting Service, United States Department of Agriculture.

```
class a b c d;
var y;
```

In addition, other procedures require TEST statements to perform appropriate tests, whereas the NESTED procedure produces the appropriate tests automatically. However, PROC NESTED makes one assumption about the input data that the other procedures do not: **PROC NESTED assumes that the input data set is sorted by the classification (CLASS) variables defining the effects.** If you use PROC NESTED on data that are not sorted by the CLASS variables, then the results might not be valid.

Getting Started: NESTED Procedure

Reliability of Automobile Models

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/nestgs1.sas.)

A study is performed to compare the reliability of several models of automobiles. Three different automobile models (Model) from each of four U.S. automobile manufacturers (Make) are tested. Three different cars of each make and model are subjected to a reliability test and given a score between 1 and 100 (Score), where higher scores indicate greater reliability.

The following statements create the SAS data set auto.

```
title1 'Reliability of Automobile Models';
data auto;
   input Make $ Model Score @@;
   datalines;
a 1 62  a 2 77  a 3 59
a 1 67  a 2 73  a 3 64
a 1 60  a 2 79  a 3 60
b 1 72  b 2 58  b 3 80
b 1 75  b 2 63  b 3 84
b 1 69  b 2 57  b 3 89
c 1 94  c 2 76  c 3 81
c 1 90  c 2 75  c 3 85
c 1 88  c 2 78  c 3 85
d 1 69  d 2 73  d 3 90
d 1 72  d 2 88  d 3 87
d 1 76  d 2 87  d 3 92
;
```

The Make variable contains the make of the automobile, represented here by ‘a’, ‘b’, ‘c’, or ‘d’, while the Model variable represents the automobile model with a ‘1’, ‘2’, or ‘3’. The Score variable contains the reliability scores given to the three sampled cars from each Make-Model group. Since the automobile models are nested within their makes, the NESTED procedure is used to analyze these data. The NESTED procedure requires the data to be sorted by Make and, within Make, by Model, so the following statements execute a PROC SORT before completing the analysis.

```
proc sort data=auto;
   by Make Model;
run;

title1 'Reliability of Automobile Models';
proc nested data=auto;
   class Make Model;
   var Score;
run;
```

The Model variable appears after the Make variable in the CLASS statement because it is nested within Make. The VAR statement specifies the response variable. The output is displayed in Figure 87.1.

Figure 87.1 Output from PROC NESTED

Reliability of Automobile Models

The NESTED Procedure

Coefficients of Expected Mean Squares			
Source	**Make**	**Model**	**Error**
Make	9	3	1
Model	0	3	1
Error	0	0	1

Nested Random Effects Analysis of Variance for Variable Score

Variance Source	DF	Sum of Squares	F Value	Pr > F	Error Term	Mean Square	Variance Component	Percent of Total
Total	35	4177.888889				119.368254	131.876543	100.0000
Make	3	1709.000000	2.15	0.1719	Model	569.666667	33.867284	25.6811
Model	8	2118.888889	18.16	<.0001	Error	264.861111	83.425926	63.2606
Error	24	350.000000				14.583333	14.583333	11.0583

Score Mean	75.94444444
Standard Error of Score Mean	3.97794848

Figure 87.1 first displays the coefficients of the variance components that make up each of the expected mean squares, and then it displays the ANOVA table. The results do not indicate significant variation between the different automobile makes ($F = 2.15$, $p = 0.1719$). However, they do suggest that there is significant variation between the different models within the makes ($F = 18.16$, $p < 0.0001$). This is evident in the fact that the make of car accounts for only 25.7% of the total variation in the data, while the car model accounts for 63.3% (as shown in the Percent of Total column). The estimated variance components are shown in the Variance Component column.

Syntax: NESTED Procedure

The following statements are available in the NESTED procedure:

PROC NESTED < *options* > **;**
  CLASS *variables* < / *option* > **;**
  VAR *variables* **;**
  BY *variables* **;**

The PROC NESTED and CLASS statements are required. The BY, CLASS, and VAR statements are described after the PROC NESTED statement.

PROC NESTED Statement

PROC NESTED < *options* > **;**

The PROC NESTED statement invokes the NESTED procedure. Table 87.1 summarizes the options available in the PROC NESTED statement.

Table 87.1 PROC NESTED Statement Options

Option	Description
AOV	Displays only the analysis of variance statistics
DATA=	Names the SAS data set

The PROC NESTED statement has the following options:

AOV
displays only the analysis of variance statistics when there is more than one dependent variable. The "analysis of covariation" statistics are suppressed (see the section "Analysis of Covariation" on page 7122).

DATA=*SAS-data-set*
names the SAS data set to be used by PROC NESTED. By default, the procedure uses the most recently created SAS data set.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC NESTED to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the NESTED procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / *option* > **;**

You must include a CLASS statement with PROC NESTED specifying the classification variables for the analysis.

Values of a variable in the CLASS statement denote the levels of an effect. The name of that variable is also the name of the corresponding effect. The second effect is assumed to be nested within the first effect, the third effect is assumed to be nested within the second effect, and so on.

By default, class levels are determined from the entire formatted values of the CLASS variables. Note that this represents a slight change from previous releases in the way in which class levels are determined. Prior to SAS 9, class levels were determined using no more than the first eight characters of the formatted values, except for numeric variables with no explicit format, for which class levels were determined from the raw numeric values. If you want to revert to this previous behavior, you can use the TRUNCATE option in the CLASS statement. In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide*, and the discussions for the FORMAT statement and SAS formats in *SAS DATA Step Statements: Reference*.

NOTE: The data set must be sorted by the classification variables in the order in which they are given in the CLASS statement. Use PROC SORT to sort the data if they are not already sorted.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

VAR Statement

VAR *variables* **;**

The VAR statement lists the dependent variables for the analysis. The dependent variables must be numeric variables. If you do not specify a VAR statement, PROC NESTED performs an analysis of variance for all numeric variables in the data set, except those already specified in the CLASS statement.

Details: NESTED Procedure

Missing Values

An observation with missing values for any of the variables used by PROC NESTED is omitted from the analysis. Blank values of CLASS character variables are treated as missing values.

Unbalanced Data

A completely nested design is defined to be unbalanced if the groups corresponding to the levels of some classification variable are not all of the same size. The NESTED procedure can compute unbiased estimates for the variance components in an unbalanced design, but because the sums of squares on which these estimates are based no longer have χ^2 distributions under a Gaussian model for the data, F tests for the significance of the variance components cannot be computed. PROC NESTED checks to see that the design is balanced. If it is not, a warning to that effect appears in the log, and the columns corresponding to the F tests in the analysis of variance are left blank.

General Random-Effects Model

A random-effects model for data from a completely nested design with n factors has the general form

$$y_{i_1 i_2 \cdots i_n r} = \mu + \alpha_{i_1} + \beta_{i_1 i_2} + \cdots + \epsilon_{i_1 i_2 \cdots i_n r}$$

where

$y_{i_1 i_2 \cdots i_n r}$	is the value of the dependent variable observed at the rth replication with factor j at level i_j, for $j = 1, \ldots, n$.
μ	is the overall (fixed) mean of the sampled population.

$\alpha_{i_1}, \beta_{i_1 i_2}, \ldots, \epsilon_{i_1 i_2 \cdots i_n r}$ are mutually uncorrelated random effects with zero means and respective variances $\sigma^2_\alpha, \sigma^2_\beta, \ldots, \sigma^2_\epsilon$.

Analysis of Covariation

When you specify more than one dependent variable, the NESTED procedure produces a descriptive analysis of the covariance between each pair of dependent variables in addition to a separate analysis of variance for each variable. The analysis of covariation is computed under the basic random-effects model for each pair of dependent variables:

$$\begin{aligned} y_{i_1 i_2 \cdots i_n r} &= \mu + \alpha_{i_1} + \beta_{i_1 i_2} + \cdots + \epsilon_{i_1 i_2 \cdots i_n r} \\ y'_{i_1 i_2 \cdots i_n r} &= \mu' + \alpha'_{i_1} + \beta'_{i_1 i_2} + \cdots + \epsilon'_{i_1 i_2 \cdots i_n r} \end{aligned}$$

where the notation is the same as that used in the preceding general random-effects model.

There is an additional assumption that all the random effects in the two models are mutually uncorrelated except for corresponding effects, for which

$$\begin{aligned} \mathrm{Corr}(\alpha_{i_1}, \alpha'_{i_1}) &= \rho_\alpha \\ \mathrm{Corr}(\beta_{i_1 i_2}, \beta'_{i_1 i_2}) &= \rho_\beta \\ &\vdots \\ \mathrm{Corr}(\epsilon_{i_1 i_2 \cdots i_n r}, \epsilon'_{i_1 i_2 \cdots i_n r}) &= \rho_\epsilon \end{aligned}$$

Error Terms in F Tests

Random-effects ANOVAs are distinguished from fixed-effects ANOVAs by which error mean squares are used as the denominator for F tests. Under a fixed-effects model, there is only one true error term in the model, and the corresponding mean square is used as the denominator for all tests. This is how the usual analysis is computed in PROC ANOVA, for example. However, in a random-effects model for a nested experiment, mean squares are compared sequentially. The correct denominator in the test for the first factor is the mean square due to the second factor; the correct denominator in the test for the second factor is the mean square due to the third factor; and so on. Only the mean square due to the last factor, the one at the bottom of the nesting order, should be compared to the error mean square.

Computational Method

The building blocks of the analysis are the sums of squares for the dependent variables for each classification variable within the factors that precede it in the model, corrected for the factors that follow it. For example, for a two-factor nested design, PROC NESTED computes the following sums of squares:

$$\text{Total SS} \qquad \sum_{ijr}(y_{ijr} - y_{...})^2$$

$$\text{SS for Factor 1} \qquad \sum_{i} n_{i\cdot}\left(\frac{y_{i\cdot\cdot}}{n_{i\cdot}} - \frac{y_{...}}{n_{\cdot\cdot}}\right)^2$$

$$\text{SS for Factor 2 within Factor 1} \qquad \sum_{ij} n_{ij}\left(\frac{y_{ij\cdot}}{n_{ij}} - \frac{y_{i\cdot\cdot}}{n_{i\cdot}}\right)^2$$

$$\text{Error SS} \qquad \sum_{ijr}\left(y_{ijr} - \frac{y_{ij\cdot}}{n_{ij}}\right)^2$$

where y_{ijr} is the rth replication, n_{ij} is the number of replications at level i of the first factor and level j of the second, and a dot as a subscript indicates summation over the corresponding index. If there is more than one dependent variable, PROC NESTED also computes the corresponding sums of crossproducts for each pair. The expected value of the sum of squares for a given classification factor is a linear combination of the variance components corresponding to this factor and to the factors that are nested within it. For each factor, the coefficients of this linear combination are computed. (The efficiency of PROC NESTED is partly due to the fact that these various sums can be accumulated with just one pass through the data, assuming that the data have been sorted by the classification variables.) Finally, estimates of the variance components are derived as the solution to the set of linear equations that arise from equating the mean squares to their expected values.

Displayed Output

PROC NESTED displays the following items for each dependent variable:

- Coefficients of Expected Mean Squares, which are the coefficients of the $n + 1$ variance components making up the expected mean square. Denoting the element in the ith row and jth column of this matrix by C_{ij}, the expected value of the mean square due to the ith classification factor is

  $$C_{i1}\sigma_1^2 + \cdots + C_{in}\sigma_n^2 + C_{i,n+1}\sigma_\epsilon^2$$

  C_{ij} is always zero for $i > j$, and if the design is balanced, C_{ij} is equal to the common size of all classification groups of the jth factor for $i \le j$. Finally, the mean square for error is always an unbiased estimate of σ_ϵ^2. In other words, $C_{n+1,n+1} = 1$.

For every dependent variable, PROC NESTED displays an analysis of variance table. Each table contains the following:

- each Variance Source in the model (the different components of variance) and the total variance
- degrees of freedom (DF) for the corresponding sum of squares
- Sum of Squares for each classification factor. The sum of squares for a given classification factor is the sum of squares in the dependent variable within the factors that precede it in the model, corrected for the factors that follow it. (See the section "Computational Method" on page 7123.)
- *F* Value for a factor, which is the ratio of its mean square to the appropriate error mean square. The next column, labeled PR > F, gives the significance levels that result from testing the hypothesis that each variance component equals zero.
- the appropriate Error Term for an *F* test, which is the mean square due to the next classification factor in the nesting order. (See the section "Error Terms in *F* Tests" on page 7122.)
- Mean Square due to a factor, which is the corresponding sum of squares divided by the degrees of freedom
- estimates of the Variance Components. These are computed by equating the mean squares to their expected values and solving for the variance terms. (See the section "Computational Method" on page 7123.)
- Percent of Total, the proportion of variance due to each source. For the *i*th factor, the value is

$$100 \times \frac{\text{source variance component}}{\text{total variance component}}$$

- Mean, the overall average of the dependent variable. This gives an unbiased estimate of the mean of the population. Its variance is estimated by a certain linear combination of the estimated variance components, which is identical to the mean square due to the first factor in the model divided by the total number of observations when the design is balanced.

If there is more than one dependent variable, then the NESTED procedure displays an "analysis of covariation" table for each pair of dependent variables (unless the AOV option is specified in the PROC NESTED statement). See the section "Analysis of Covariation" on page 7122 for details. For each source of variation, this table includes the following:

- Degrees of Freedom
- Sum of Products
- Mean Products
- Covariance Component, the estimate of the covariance component

Items in the analysis of covariation table are computed analogously to their counterparts in the analysis of variance table. The analysis of covariation table also includes the following:

- Variance Component Correlation for a given factor. This is an estimate of the correlation between corresponding effects due to this factor. This correlation is the ratio of the covariance component for this factor to the square root of the product of the variance components for the factor for the two different dependent variables. (See the section "Analysis of Covariation" on page 7122.)

- Mean Square Correlation for a given classification factor. This is the ratio of the Mean Products for this factor to the square root of the product of the Mean Squares for the factor for the two different dependent variables.

ODS Table Names

PROC NESTED assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 87.2 ODS Tables Produced by PROC NESTED

ODS Table Name	Description	Statement
ANCOVA	Analysis of covariance	Default with more than one dependent variable
ANOVA	Analysis of variance	Default
EMSCoef	Coefficients of expected mean squares	Default
Statistics	Overall statistics for fit	Default

Example: NESTED Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 87.1: Variability of Calcium Concentration in Turnip Greens

(View the complete code for this example (nestex1.sas) in the example repository.)

In the following example from Snedecor and Cochran (1967), an experiment is conducted to study the variability of calcium concentration in turnip greens. Four plants are selected at random; then three leaves are randomly selected from each plant. Two 100-mg samples are taken from each leaf. The amount of calcium is determined by microchemical methods.

Because the data are read in sorted order, it is not necessary to use PROC SORT on the CLASS variables. Leaf is nested in Plant; Sample is nested in Leaf and is left for the residual term. All the effects are random effects. The following statements read the data and invoke PROC NESTED. These statements produce Output 87.1.1.

```
title 'Calcium Concentration in Turnip Leaves--Nested Random Model';
title2 'Snedecor and Cochran, ''Statistical Methods'', 1967, p. 286';
data Turnip;
   do Plant=1 to 4;
      do Leaf=1 to 3;
         do Sample=1 to 2;
            input Calcium @@;
            output;
         end;
      end;
   end;
   datalines;
3.28 3.09 3.52 3.48 2.88 2.80 2.46 2.44
1.87 1.92 2.19 2.19 2.77 2.66 3.74 3.44
2.55 2.55 3.78 3.87 4.07 4.12 3.31 3.31
;

proc nested data=Turnip;
   class plant leaf;
   var calcium;
run;
```

Output 87.1.1 Analysis of Calcium Concentration in Turnip Greens Using PROC NESTED

Calcium Concentration in Turnip Leaves--Nested Random Model
Snedecor and Cochran, 'Statistical Methods', 1967, p. 286

The NESTED Procedure

Coefficients of Expected Mean Squares			
Source	**Plant**	**Leaf**	**Error**
Plant	6	2	1
Leaf	0	2	1
Error	0	0	1

Nested Random Effects Analysis of Variance for Variable Calcium

Variance Source	DF	Sum of Squares	F Value	Pr > F	Error Term	Mean Square	Variance Component	Percent of Total
Total	23	10.270396				0.446539	0.532938	100.0000
Plant	3	7.560346	7.67	0.0097	Leaf	2.520115	0.365223	68.5302
Leaf	8	2.630200	49.41	<.0001	Error	0.328775	0.161060	30.2212
Error	12	0.079850				0.006654	0.006654	1.2486

Calcium Mean	3.01208333
Standard Error of Calcium Mean	0.32404445

The results indicate that there is significant (nonzero) variation from plant to plant (Pr > F is 0.0097) and from leaf to leaf within a plant (Pr > F is less than 0.0001). Notice that the variance component for Plant uses the Leaf mean square as an error term in the model rather than the error mean square.

References

Snedecor, G. W., and Cochran, W. G. (1967). *Statistical Methods*. 6th ed. Ames: Iowa State University Press.

Steel, R. G. D., and Torrie, J. H. (1980). *Principles and Procedures of Statistics*. 2nd ed. New York: McGraw-Hill.

Chapter 88
The NLIN Procedure

Contents

Overview: NLIN Procedure **7130**
Getting Started: NLIN Procedure **7132**
- Nonlinear or Linear Model 7132
- Notation for Nonlinear Regression Models 7133
- Estimating the Parameters in the Nonlinear Model 7134

Syntax: NLIN Procedure **7138**
- PROC NLIN Statement 7139
- BOOTSTRAP Statement 7150
- BOUNDS Statement 7152
- BY Statement 7153
- CONTROL Statement 7153
- CONTRAST Statement 7154
- DER Statement 7154
- ESTIMATE Statement 7155
- ID Statement 7155
- MODEL Statement 7155
- OUTPUT Statement 7156
- PARAMETERS Statement 7160
- PROFILE Statement 7163
- RETAIN Statement 7164
- Other Programming Statements 7165

Details: NLIN Procedure **7166**
- Automatic Derivatives 7166
- Measures of Nonlinearity, Diagnostics and Inference 7168
- Missing Values 7176
- Special Variables 7176
- Troubleshooting 7178
- Computational Methods 7180
- Output Data Sets 7185
- Confidence Intervals 7186
- Covariance Matrix of Parameter Estimates 7187
- Convergence Measures 7188
- Displayed Output 7189
- ODS Table Names 7190
- ODS Graphics 7191

Examples: NLIN Procedure **7193**

Example 88.1: Segmented Model 7193
Example 88.2: Iteratively Reweighted Least Squares 7197
Example 88.3: Probit Model with Likelihood Function 7201
Example 88.4: Affecting Curvature through Parameterization 7204
Example 88.5: Comparing Nonlinear Trends among Groups 7210
Example 88.6: ODS Graphics and Diagnostics 7219
Example 88.7: Parameter Profiling and Bootstrapping 7223
References **7233**

Overview: NLIN Procedure

The NLIN procedure fits nonlinear regression models and estimates the parameters by nonlinear least squares or weighted nonlinear least squares. You specify the model with programming statements. This gives you great flexibility in modeling the relationship between the response variable and independent (regressor) variables. It does, however, require additional coding compared to model specifications in linear modeling procedures such as the REG, GLM, and MIXED procedures.

Estimating parameters in a nonlinear model is an iterative process that commences from starting values. You need to declare the parameters in your model and supply their initial values for the NLIN procedure. You do not need to specify derivatives of the model equation with respect to the parameters. Although facilities for specifying first and second derivatives exist in the NLIN procedure, it is not recommended that you specify derivatives this way. Obtaining derivatives from user-specified expressions predates the high-quality automatic differentiator that is now used by the NLIN procedure.

Nonlinear least-squares estimation involves finding those values in the parameter space that minimize the (weighted) residual sum of squares. In a sense, this is a “distribution-free” estimation criterion since the distribution of the data does not need to be fully specified. Instead, the assumption of homoscedastic and uncorrelated model errors with zero mean is sufficient. You can relax the homoscedasticity assumption by using a weighted residual sum of squares criterion. The assumption of uncorrelated errors (independent observations) cannot be relaxed in the NLIN procedure. In summary, the primary assumptions for analysis with the NLIN procedure are as follows:

- The structure in the response variable can be decomposed additively into a mean function and an error component.
- The model errors are uncorrelated and have zero mean. Unless a weighted analysis is performed, the errors are also assumed to be homoscedastic (have equal variance).
- The mean function consists of known regressor (independent) variables and unknown constants (the parameters).

Fitting nonlinear models can be a difficult undertaking. There is no closed-form solution for the parameter estimates, and the process is iterative. There can be one or more local minima in the residual sum of squares surface, and the process depends on the starting values supplied by the user. You can reduce the dependence on the starting values and reduce the chance to arrive at a local minimum by specifying a grid of starting values. The NLIN procedure then computes the residual sum of squares at each point on the grid and starts the

iterative process from the point that yields the lowest sum of squares. Even in this case, however, convergence does not guarantee that a global minimum has been found.

The numerical behavior of a model and a model–data combination can depend on the way in which you parameterize the model—for example, whether parameters are expressed on the logarithmic scale or not. Parameterization also has bearing on the interpretation of the estimated quantities and the statistical properties of the parameter estimators. Inferential procedures in nonlinear regression models are typically approximate in that they rely on the asymptotic properties of the parameter estimators that are obtained as the sample size grows without bound. Such asymptotic inference can be questionable in small samples, especially if the behavior of the parameter estimators is "far-from-linear." Reparameterization of the model can yield parameters whose behavior is akin to that of estimators in linear models. These parameters exhibit close-to-linear behavior.

The NLIN procedure solves the nonlinear least squares problem by one of the following four algorithms (methods):

- steepest-descent or gradient method
- Newton method
- modified Gauss-Newton method
- Marquardt method

These methods use derivatives or approximations to derivatives of the SSE with respect to the parameters to guide the search for the parameters producing the smallest SSE. Derivatives computed automatically by the NLIN procedure are analytic, unless the model contains functions for which an analytic derivative is not available.

Using PROC NLIN, you can also do the following:

- confine the estimation procedure to a certain range of values of the parameters by imposing bounds on the estimates
- produce new SAS data sets containing predicted values, parameter estimates, residuals and other model diagnostics, estimates at each iteration, and so forth.

You can use the NLIN procedure for segmented models (see Example 88.1) or robust regression (see Example 88.2). You can also use it to compute maximum-likelihood estimates for certain models (see Jennrich and Moore 1975; Charnes, Frome, and Yu 1976). For maximum likelihood estimation in a model with a linear predictor and binomial error distribution, see the LOGISTIC, PROBIT, GENMOD, GLIMMIX, and CATMOD procedures. For a linear model with a Poisson, gamma, or inverse Gaussian error distribution, see the GENMOD and GLIMMIX procedures. For likelihood estimation in a linear model with a normal error distribution, see the MIXED, GENMOD, and GLIMMIX procedures. The PHREG and LIFEREG procedures fit survival models by maximum likelihood. For general maximum likelihood estimation, see the NLP procedure in the *SAS/OR User's Guide: Mathematical Programming* and the NLMIXED procedure. These procedures are recommended over the NLIN procedure for solving maximum likelihood problems.

PROC NLIN uses the Output Delivery System (ODS). ODS enables you to convert any of the output from PROC NLIN into a SAS data set. See the section "ODS Table Names" on page 7190 for a listing of the ODS tables that are produced by the NLIN procedure.

PROC NLIN exploits all the available cores on a multicore machine during bootstrap estimation and parameter profiling. It does so by performing multiple optimizations in parallel for these tasks. see the BOOTSTRAP and PROFILE statements.

In addition, PROC NLIN can produce graphs when ODS Graphics is enabled. For more information, see the PLOTS option and the section "ODS Graphics" on page 7191 for a listing of the ODS graphs.

Getting Started: NLIN Procedure

Nonlinear or Linear Model

The NLIN procedure performs univariate nonlinear regression by using the least squares method. Nonlinear regression analysis is indicated when the functional relationship between the response variable and the predictor variables is nonlinear. Nonlinearity in this context refers to a nonlinear relationship in the *parameters*. Many linear regression models exhibit a relationship in the regressor (predictor) variables that is not simply a straight line. This does not make the models nonlinear. A model is nonlinear in the parameters if the derivative of the model with respect to a parameter depends on this or other parameters.

Consider, for example the models

$$\mathrm{E}[Y|x] = \beta_0 + \beta_1 x$$
$$\mathrm{E}[Y|x] = \beta_0 + \beta_1 x + \beta_2 x^2$$
$$\mathrm{E}[Y|x] = \beta + x/\alpha$$

In these expressions, $\mathrm{E}[Y|x]$ denotes the expected value of the response variable Y at the fixed value of x. (The conditioning on x simply indicates that the predictor variables are assumed to be non-random in models fit by the NLIN procedure. Conditioning is often omitted for brevity in this chapter.)

Only the third model is a nonlinear model. The first model is a simple linear regression. It is linear in the parameters β_0 and β_1 since the model derivatives do not depend on unknowns:

$$\frac{\partial}{\beta_0}(\beta_0 + \beta_1 x) = 1$$
$$\frac{\partial}{\beta_1}(\beta_0 + \beta_1 x) = x$$

The model is also linear in its relationship with x (a straight line). The second model is also linear in the parameters, since

$$\frac{\partial}{\beta_0}\left(\beta_0 + \beta_1 x + \beta_2 x^2\right) = 1$$
$$\frac{\partial}{\beta_1}\left(\beta_0 + \beta_1 x + \beta_2 x^2\right) = x$$
$$\frac{\partial}{\beta_2}\left(\beta_0 + \beta_1 x + \beta_2 x^2\right) = x^2$$

It is a *curvilinear* model since it exhibits a curved relationship when plotted against x. The third model, finally, is a nonlinear model since

$$\frac{\partial}{\beta}(\beta + x/\alpha) = 1$$
$$\frac{\partial}{\alpha}(\beta + x/\alpha) = -\frac{x}{\alpha^2}$$

The second of these derivatives depends on a parameter α. A model is nonlinear if it is not linear in at least one parameter.

Notation for Nonlinear Regression Models

This section briefly introduces the basic notation for nonlinear regression models that applies in this chapter. Additional notation is introduced throughout as needed.

The $(n \times 1)$ vector of observed responses is denoted as $\mathbf{y}$. This vector is the realization of an $(n \times 1)$ random vector $\mathbf{Y}$. The NLIN procedure assumes that the variance matrix of this random vector is $\sigma^2\mathbf{I}$. In other words, the observations have equal variance (are homoscedastic) and are uncorrelated. By defining the special variable _WEIGHT_ in your NLIN programming statements, you can introduce heterogeneous variances. If a _WEIGHT_ variable is present, then $\mathrm{Var}[\mathbf{Y}] = \sigma^2\mathbf{W}^{-1}$, where $\mathbf{W}$ is a diagonal matrix containing the values of the _WEIGHT_ variable.

The mean of the random vector is represented by a nonlinear model that depends on parameters $\beta_1, \ldots, \beta_p$ and regressor (independent) variables $z_1, \ldots, z_k$:

$$\mathrm{E}[Y_i] = f\left(\beta_1, \beta_2, \ldots, \beta_p; z_{i1}, \ldots, z_{ik}\right)$$

In contrast to linear models, the number of regressor variables (k) does not necessarily equal the number of parameters (p) in the mean function $f(\,)$. For example, the model fitted in the next subsection contains a single regressor and two parameters.

To represent the mean of the vector of observations, boldface notation is used in an obvious extension of the previous equation:

$$\mathrm{E}[\mathbf{Y}] = \mathbf{f}(\boldsymbol{\beta}; \mathbf{z}_1, \ldots, \mathbf{z}_k)$$

The vector $\mathbf{z}_1$, for example, is an $(n \times 1)$ vector of the values for the first regressor variables. The explicit dependence of the mean function on $\boldsymbol{\beta}$ and/or the $\mathbf{z}$ vectors is often omitted for brevity.

In summary, the stochastic structure of models fit with the NLIN procedure is mathematically captured by

$$\begin{aligned} \mathbf{Y} &= \mathbf{f}(\boldsymbol{\beta}; \mathbf{z}_1, \ldots, \mathbf{z}_k) + \boldsymbol{\epsilon} \\ \mathrm{E}[\boldsymbol{\epsilon}] &= \mathbf{0} \\ \mathrm{Var}[\boldsymbol{\epsilon}] &= \sigma^2\mathbf{I} \end{aligned}$$

Note that the residual variance σ^2 is typically also unknown. Since it is not estimated in the same fashion as the other p parameters, it is often not counted in the number of parameters of the nonlinear regression. An estimate of σ^2 is obtained after the model fit by the method of moments based on the residual sum of squares.

A matrix that plays an important role in fitting nonlinear regression models is the $(n \times p)$ matrix of the first partial derivatives of the mean function $\mathbf{f}$ with respect to the p model parameters. It is frequently denoted as

$$\mathbf{X} = \frac{\partial \mathbf{f}\left(\boldsymbol{\beta}; \mathbf{z}_1, \ldots, \mathbf{z}_k\right)}{\partial \boldsymbol{\beta}}$$

The use of the symbol $\mathbf{X}$—common in linear statistical modeling—is no accident here. The first derivative matrix plays a similar role in nonlinear regression to that of the $\mathbf{X}$ matrix in a linear model. For example, the asymptotic variance of the nonlinear least-squares estimators is proportional to $(\mathbf{X}'\mathbf{X})^{-1}$, and projection-type matrices in nonlinear regressions are based on $\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$. Also, fitting a nonlinear regression model can be cast as an iterative process where a nonlinear model is approximated by a series of linear models in which the derivative matrix is the regressor matrix. An important difference between linear and nonlinear models is that the derivatives in a linear model do not depend on any parameters (see previous subsection). In contrast, the derivative matrix $\partial \mathbf{f}(\boldsymbol{\beta})/\partial \boldsymbol{\beta}$ is a function of at least one element of $\boldsymbol{\beta}$. It is this dependence that lies at the core of the fact that estimating the parameters in a nonlinear model cannot be accomplished in closed form, but it is an iterative process that commences with user-supplied starting values and attempts to continually improve on the parameter estimates.

Estimating the Parameters in the Nonlinear Model

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/nlings.sas.)

As an example of a nonlinear regression analysis, consider the following theoretical model of enzyme kinetics. The model relates the initial velocity of an enzymatic reaction to the substrate concentration.

$$f(\mathbf{x}, \boldsymbol{\theta}) = \frac{\theta_1 x_i}{\theta_2 + x_i}, \quad \text{for } i = 1, 2, \ldots, n$$

where x_i represents the amount of substrate for n trials and $f(\mathbf{x}, \boldsymbol{\theta})$ is the velocity of the reaction. The vector $\boldsymbol{\theta}$ contains the rate parameters. This model is known as the Michaelis-Menten model in biochemistry (Ratkowsky 1990, p. 59). The model exists in many parameterizations. In the form shown here, θ_1 is the maximum velocity of the reaction that is theoretically attainable. The parameter θ_2 is the substrate concentration at which the velocity is 50% of the maximum.

Suppose that you want to study the relationship between concentration and velocity for a particular enzyme/substrate pair. You record the reaction rate (velocity) observed at different substrate concentrations.

A SAS data set is created for this experiment in the following DATA step:

```
data Enzyme;
   input Concentration Velocity @@;
   datalines;
0.26 124.7   0.30 126.9
0.48 135.9   0.50 137.6
0.54 139.6   0.68 141.1
0.82 142.8   1.14 147.6
1.28 149.8   1.38 149.4
1.80 153.9   2.30 152.5
2.44 154.5   2.48 154.7
;
```

The SAS data set Enzyme contains the two variables Concentration (substrate concentration) and Velocity (reaction rate). The following statements fit the Michaelis-Menten model by nonlinear least squares:

```
proc nlin data=Enzyme method=marquardt hougaard;
   parms theta1=155
         theta2=0 to 0.07 by 0.01;
   model Velocity = theta1*Concentration / (theta2 + Concentration);
run;
```

The DATA= option specifies that the SAS data set Enzyme be used in the analysis. The METHOD= option directs PROC NLIN to use the MARQUARDT iterative method. The HOUGAARD option requests that a skewness measure be calculated for the parameters.

The PARMS statement declares the parameters and specifies their initial values. Suppose that V represents the velocity and C represents the substrate concentration. In this example, the initial estimates listed in the PARMS statement for θ_1 and θ_2 are obtained as follows:

θ_1: Because the model is a monotonic increasing function in C, and because

$$\lim_{C\to\infty}\left(\frac{\theta_1 C}{\theta_2 + C}\right) = \theta_1$$

you can take the largest observed value of the variable Velocity (154.7) as the initial value for the parameter Theta1. Thus, the PARMS statement specifies 155 as the initial value for Theta1, which is approximately equal to the maximum observed velocity.

θ_2: To obtain an initial value for the parameter θ_2, first rearrange the model equation to solve for θ_2:

$$\theta_2 = \frac{\theta_1 C}{V} - C$$

By substituting the initial value of Theta1 for θ_1 and taking each pair of observed values of Concentration and Velocity for C and V, respectively, you obtain a set of possible starting values for Theta2 ranging from about 0.01 to 0.07.

You can choose any value within this range as a starting value for Theta2, or you can direct PROC NLIN to perform a preliminary search for the best initial Theta2 value within that range of values. The PARMS statement specifies a range of values for Theta2, resulting in a search over the grid points from 0 to 0.07 in increments of 0.01.

The MODEL statement specifies the enzymatic reaction model

$$V = \frac{\theta_1 C}{\theta_2 + C}$$

in terms of the data set variables Velocity and Concentration and in terms of the parameters in the PARMS statement.

The results from this PROC NLIN invocation are displayed in the following figures.

PROC NLIN evaluates the model at each point on the specified grid for the Theta2 parameter. Figure 88.1 displays the calculations resulting from the grid search.

Figure 88.1 Nonlinear Least-Squares Grid Search

The NLIN Procedure
Dependent Variable Velocity

Grid Search		
theta1	theta2	Sum of Squares
155.0	0	3075.4
155.0	0.0100	2074.1
155.0	0.0200	1310.3
155.0	0.0300	752.0
155.0	0.0400	371.9
155.0	0.0500	147.2
155.0	0.0600	58.1130
155.0	0.0700	87.9662

The parameter Theta1 is held constant at its specified initial value of 155, the grid is traversed, and the residual sum of squares is computed at each point. The "best" starting value is the point that corresponds to the smallest value of the residual sum of squares. The best set of starting values is obtained for $\theta_1 = 155, \theta_2 = 0.06$ (Figure 88.1). PROC NLIN uses this point from which to start the following, iterative phase of nonlinear least-squares estimation.

Figure 88.2 displays the iteration history. Note that the first entry in the "Iterative Phase" table echoes the starting values and the residual sum of squares for the best value combination in Figure 88.1. The subsequent rows of the table show the updates of the parameter estimates and the improvement (decrease) in the residual sum of squares. For this data-and-model combination, the first iteration yielded a large improvement in the sum of squares (from 58.113 to 19.7017). Further steps were necessary to improve the estimates in order to achieve the convergence criterion. The NLIN procedure by default determines convergence by using R, the relative offset measure of Bates and Watts (1981). Convergence is declared when this measure is less than 10^{-5}—in this example, after three iterations.

Figure 88.2 Iteration History and Convergence Status

The NLIN Procedure
Dependent Variable Velocity
Method: Marquardt

Iterative Phase			
Iter	theta1	theta2	Sum of Squares
0	155.0	0.0600	58.1130
1	158.0	0.0736	19.7017
2	158.1	0.0741	19.6606
3	158.1	0.0741	19.6606

NOTE: Convergence criterion met.

Figure 88.3 Estimation Summary

Estimation Summary	
Method	Marquardt
Iterations	3
R	5.861E-6
PPC(theta2)	8.569E-7
RPC(theta2)	0.000078
Object	2.902E-7
Objective	19.66059
Observations Read	14
Observations Used	14
Observations Missing	0

A summary of the estimation including several convergence measures (R, PPC, RPC, and Object) is displayed in Figure 88.3.

The "R" measure in Figure 88.3 is the relative offset convergence measure of Bates and Watts. A "PPC" value of 8.569E–7 indicates that the parameter Theta2 (which has the largest PPC value of the parameters) would change by that relative amount, if PROC NLIN were to take an additional iteration step. The "RPC" value indicates that Theta2 changed by 0.000078, relative to its value in the last iteration. These changes are measured before step length adjustments are made. The "Object" measure indicates that the objective function changed by 2.902E–7 in relative value from the last iteration.

Figure 88.4 displays the analysis of variance table for the model. The table displays the degrees of freedom, sums of squares, and mean squares along with the model F test.

Figure 88.4 Nonlinear Least-Squares Analysis of Variance

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	290116	145058	88537.2	<.0001
Error	12	19.6606	1.6384		
Uncorrected Total	14	290135			

Figure 88.5 Parameter Estimates and Approximate 95% Confidence Intervals

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness
theta1	158.1	0.6737	156.6	159.6	0.0152
theta2	0.0741	0.00313	0.0673	0.0809	0.0362

Figure 88.5 displays the estimates for each parameter, the associated asymptotic standard error, and the upper and lower values for the asymptotic 95% confidence interval. PROC NLIN also displays the asymptotic correlations between the estimated parameters (not shown).

The skewness measures of 0.0152 and 0.0362 indicate that the parameter estimators exhibit close-to-linear

behavior and that their standard errors and confidence intervals can be safely used for inferences.

Thus, the estimated nonlinear model that relates reaction velocity and substrate concentration can be written as

$$\widehat{V} = \frac{158.1C}{0.0741 + C}$$

where $\widehat{V}$ represents the predicted velocity or rate of the reaction and C represents the substrate concentration.

Syntax: NLIN Procedure

The following statements are available in the NLIN procedure:

```
PROC NLIN <options> ;
    BOOTSTRAP </ options> ;
    BOUNDS inequality <, ..., inequality> ;
    BY variables ;
    CONTROL variable <=values> <... variable <=values>> ;
    CONTRAST 'label' expression <, expression> <option> ;
    DER. parameter=expression ;
    DER. parameter.parameter=expression ;
    ESTIMATE 'label' expression <options> ;
    ID variables ;
    MODEL dependent = expression ;
    OUTPUT OUT=SAS-data-set keyword=names <... keyword=names> ;
    PARAMETERS <parameter-specification> <, ..., parameter-specification>
                    </ PDATA=SAS-data-set> ;
    PROFILE parameter <... parameter> </ options> ;
    RETAIN variable <=values> <... variable <=values>> ;
    Programming statements ;
```

The statements in the NLIN procedure, in addition to the PROC NLIN statement, are as follows:

BOOTSTRAP	requests bootstrap resampling and estimation
BOUNDS	constrains the parameter estimates within specified bounds
BY	specifies variables to define subgroups for the analysis
DER	specifies the first or second partial derivatives
ID	specifies additional variables to add to the output data set
MODEL	defines the relationship between the dependent and independent variables (the mean function)
OUTPUT	creates an output data set containing observation-wise statistics
PARAMETERS	identifies parameters to be estimated and their starting values
PROFILE	identifies parameters to be profiled

Programming Statements includes assignment statements, ARRAY statements, DO loops, and other program control statements. These are valid SAS expressions that can appear in the DATA step. PROC NLIN enables you to create new variables within the procedure and use them in the nonlinear analysis. These programming statements can appear anywhere in the PROC NLIN code, but new variables must be created before they appear in other statements. The NLIN procedure automatically creates several variables that are also available for use in the analysis. See the section "Special Variables" on page 7176 for more information.

The PROC NLIN, PARAMETERS, and MODEL statements are required.

PROC NLIN Statement

PROC NLIN < *options* > **;**

The PROC NLIN statement invokes the NLIN procedure.

Table 88.1 summarizes the *options* available in the PROC NLIN statement. All *options* are subsequently discussed in alphabetical order.

Table 88.1 Summary of Options in PROC NLIN Statement

Option	Description
Options Related to Data Sets	
DATA=	Specifies the input data set
OUTEST=	Specifies the output data set for parameter estimates, covariance matrix, and so on
SAVE	Requests that final estimates be added to the OUTEST= data set
Optimization Options	
BEST=	Limits display of grid search
METHOD=	Chooses the optimization method
MAXITER=	Specifies the maximum number of iterations
MAXSUBIT=	Specifies the maximum number of step halvings
NOHALVE	Allows the objective function to increase between iterations
RHO=	Controls the step-size search
SMETHOD=	Specifies the step-size search method
TAU=	Controls the step-size search
G4	Uses the Moore-Penrose inverse
UNCORRECTEDDF	Does not expense degrees of freedom when bounds are active
SIGSQ=	Specifies the fixed value for residual variance
Singularity and Convergence Criteria	
CONVERGE=	Tunes the convergence criterion
CONVERGEOBJ=	Uses the change in loss function as the convergence criterion and tunes its value
CONVERGEPARM=	Uses the maximum change in parameter estimates as the convergence criterion and tunes its value

Table 88.1 *continued*

Option	Description
SINGULAR=	Tunes the singularity criterion used in matrix inversions
ODS Graphics Options	
PLOTS=	Produces ODS graphical displays
Displayed Output	
ALPHA=	Specifies α for confidence limits
DF=	Specifies the degrees of freedom for p-values and confidence limits
ECORR	Requests the correlation matrix of additional estimates
ECOV	Requests the covariance matrix of additional estimates
EDER	Requests derivatives of additional estimates
HOUGAARD	Adds Hougaard's skewness measure to the "Parameter Estimates" table
BIAS	Adds Box's bias measure to the "Parameter Estimates" table
NOITPRINT	Suppresses the "Iteration History" table
NOPRINT	Suppresses displayed output
LIST	Displays the model program and variable list
LISTALL	Selects the LIST, LISTDEP, LISTDER, and LISTCODE options
LISTCODE	Displays the model program code
LISTDEP	Displays dependencies of model variables
LISTDER	Displays the derivative table
NLINMEASURES	Displays the global nonlinearity measures table
TOTALSS	Adds the uncorrected or corrected total sum of squares to the analysis of variance table
XREF	Displays the cross-reference of variables
Trace Model Execution	
FLOW	Displays execution messages for program statements
PRINT	Displays results of statements in model program
TRACE	Displays results of operations in model program

ALPHA=*α*

specifies the level of significance α used in the construction of $100(1-\alpha)\%$ confidence intervals. The value must be strictly between 0 and 1; the default value of $\alpha = 0.05$ results in 95% intervals. This value is used as the default confidence level for limits computed in the "Parameter Estimates" table and with the LCLM, LCL, UCLM, and UCL options in the OUTPUT statement.

BEST=*n*

requests that PROC NLIN display the residual sums of squares only for the best *n* combinations of possible starting values from the grid. If you do not specify the BEST= option, PROC NLIN displays the residual sum of squares for every combination of possible parameter starting values.

BIAS

adds Box's bias and percentage bias measures to the "Parameter Estimates" table (Box 1971). Box's bias measure, along with Hougaard's measure of skewness, is used for assessing a parameter estimator's close-to-linear behavior (Ratkowsky 1983, 1990). Hence, it is useful for identifying problematic parameters (Seber and Wild 1989, sec. 4.7.1). When you specify the BIAS option, Box's bias measure (Box 1971) and the percentage bias (the bias expressed as a percentage of the least-squares estimator) are added for each parameter to the "Parameter Estimates" table. Ratkowsky (1983, p. 21) takes a percentage bias in excess of 1% to be a good rule of thumb for indicating nonlinear behavior.

See the section "Box's Measure of Bias" on page 7168 for further details. Example 88.4 shows how to use this measure, along with Hougaard's measure of skewness, to evaluate changes in the parameterization of a nonlinear model. Computation of the Box's bias measure requires first and second derivatives. If you do not provide derivatives with the DER statement—and it is recommended that you do not—the analytic derivatives are computed for you.

CONVERGE=*c*

specifies the convergence criterion for PROC NLIN. For all iterative methods the relative offset convergence measure of Bates and Watts is used by default to determine convergence. This measure is labeled "R" in the "Estimation Summary" table. The iterations are said to have converged for CONVERGE=*c* if

$$\sqrt{\frac{\mathbf{r}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{r}}{\mathrm{LOSS}^{(i)}}} < c$$

where $\mathbf{r}$ is the residual vector and $\mathbf{X}$ is the $(n \times p)$ matrix of first derivatives with respect to the parameters. The default LOSS function is the sum of squared errors (SSE), and $\mathrm{LOSS}^{(i)}$ denotes the value of the loss function at the ith iteration. By default, CONVERGE=10^{-5}. The R convergence measure cannot be computed accurately in the special case of a perfect fit (residuals close to zero). When the SSE is less than the value of the SINGULAR= criterion, convergence is assumed.

CONVERGEOBJ=*c*

uses the change in the LOSS function as the convergence criterion and tunes the criterion. The iterations are said to have converged for CONVERGEOBJ=*c* if

$$\frac{|\mathrm{LOSS}^{(i-1)} - \mathrm{LOSS}^{(i)}|}{|\mathrm{LOSS}^{(i-1)} + 10^{-6}|} < c$$

where $\mathrm{LOSS}^{(i)}$ is the LOSS for the ith iteration. The default LOSS function is the sum of squared errors (SSE), the residual sum of squares. The constant *c* should be a small positive number. For more details about the LOSS function, see the section "Special Variable Used to Determine Convergence Criteria" on page 7177. For more details about the computational methods in the NLIN procedure, see the section "Computational Methods" on page 7180.

If you specify the CONVERGEOBJ= option, the CONVERGE= option is disabled.

CONVERGEPARM=*c*

uses the maximum change among parameter estimates as the convergence criterion and tunes the criterion. The iterations are said to have converged for CONVERGEPARM=*c* if

$$\max_j \left(\frac{|\beta_j^{(i-1)} - \beta_j^{(i)}|}{|\beta_j^{(i-1)}|} \right) < c$$

where $\beta_j^{(i)}$ is the value of the *j*th parameter at the *i*th iteration.

The default convergence criterion is CONVERGE. If you specify CONVERGEPARM=*c*, the maximum change in parameters is used as the convergence criterion. If you specify both the CONVERGEOBJ= and CONVERGEPARM= options, PROC NLIN continues to iterate until the decrease in LOSS is sufficiently small (as determined by the CONVERGEOBJ= option) and the maximum change among the parameters is sufficiently small (as determined by the CONVERGEPARM= option).

DATA=*SAS-data-set*

specifies the input SAS data set to be analyzed by PROC NLIN. If you omit the DATA= option, the most recently created SAS data set is used.

DF=*d*

specifies the default number of degrees of freedom to use in the calculation of *p*-values and confidence limits for additional parameter estimates.

ECOV

requests the approximate covariance matrix for all expressions that are specified in ESTIMATE statements.

ECORR

requests the approximate correlation matrix for all expressions that are specified in ESTIMATE statements.

EDER

requests the derivatives of all expressions that are specified in ESTIMATE statements with respect to each of the model parameters.

FLOW

displays a message for each statement in the model program as it is executed. This debugging option is rarely needed, and it produces large amounts of output.

G4

uses a Moore-Penrose inverse (g_4-inverse) in parameter estimation. See Kennedy and Gentle (1980) for details.

HOUGAARD

adds Hougaard's measure of skewness to the "Parameter Estimates" table (Hougaard 1982, 1985). The skewness measure is one method of assessing a parameter estimator's close-to-linear behavior in the sense of Ratkowsky (1983, 1990). The behavior of estimators that are close to linear approaches that of least squares estimators in linear models, which are unbiased and have minimum variance. When you specify the HOUGAARD option, the standardized skewness measure of Hougaard (1985) is added for each parameter to the "Parameter Estimates" table. Because of the linkage between nonlinear behavior

of a parameter estimator in nonlinear regression and the nonnormality of the estimator's sampling distribution, Ratkowsky (1990, p. 28) provides the following rules to interpret the (standardized) Hougaard skewness measure:

- Values less than 0.1 in absolute value indicate very close-to-linear behavior.
- Values between 0.1 and 0.25 in absolute value indicate reasonably close-to-linear behavior.
- The nonlinear behavior is apparent for absolute values above 0.25 and is considerable for absolute values above 1.

See the section "Hougaard's Measure of Skewness" on page 7169 for further details. Example 88.4 shows how to use this measure to evaluate changes in the parameterization of a nonlinear model. Computation of the Hougaard skewness measure requires first and second derivatives. If you do not provide derivatives with the DER statement—and it is recommended that you do not—the analytic derivatives are computed for you. For weighted least squares, the NLIN procedure ignores the weights for computing the Hougaard skewness measure. This can be a strong assumption as the formulation in Hougaard (1985) assumes homoscedastic errors.

LIST

displays the model program and variable lists, including the statements added by macros. Note that the expressions displayed by the LIST option do not necessarily represent the way the expression is actually calculated—because intermediate results for common subexpressions can be reused—but are shown in expanded form. To see how the expression is actually evaluated, use the LISTCODE option.

LISTALL

selects the LIST, LISTDEP, LISTDER, and LISTCODE options.

LISTCODE

displays the derivative tables and the compiled model program code. The LISTCODE option is a debugging feature and is not normally needed.

LISTDEP

produces a report that lists, for each variable in the model program, the variables that depend on it and the variables on which it depends.

LISTDER

displays a table of derivatives. The derivatives table lists each nonzero derivative computed for the problem. The derivative listed can be a constant, a variable in the model program, or a special derivative variable created to hold the result of an expression.

MAXITER=*n*

specifies the maximum number n of iterations in the optimization process. The default is $n = 100$.

MAXSUBIT=*n*

places a limit on the number of step halvings. The value of MAXSUBIT must be a positive integer and the default value is $n = 30$.

METHOD=GAUSS | MARQUARDT | NEWTON | GRADIENT

specifies the iterative method that the NLIN procedure uses to solve the nonlinear least squares problem. The Gauss, Marquardt, and Newton methods are more robust than the gradient method. If you omit this option, the Gauss method is used. See the section "Computational Methods" on page 7180 for more information.

NLINMEASURES

displays the global nonlinearity measures table. These measures include the maximum intrinsic and parameter-effects curvatures (Bates and Watts 1980), the root mean square (RMS) intrinsic and parameter-effects curvatures, and the critical curvature value (Bates and Watts 1980). The estimates of residual variance that are computed by using the ordinary and projected residuals are also included. According to Bates and Watts, both intrinsic and parameter-effects curvatures are deemed negligible if their values are less than the critical curvature value. This value is given by $1/(\sqrt{F})$, where $F = F(p, n-p; \alpha)$. The value $1/\sqrt{F}$ can be considered to be the radius of curvature of the $100(1-\alpha)$ percent confidence region (Bates and Watts 1980). For weighted least squares, PROC NLIN ignores the weights when it computes the curvature measures. This can be a strong assumption, because the original derivation by Bates and Watts assumes homoscedastic errors.

NOHALVE

removes the restriction that the objective value must decrease at every iteration. Step halving is still used to satisfy the bounds that you specify in the BOUNDS statement and to ensure that the number of observations that can be evaluated does not decrease. This option can be useful in weighted nonlinear least squares problems where the weights depend on the parameters, such as in iteratively reweighted least squares (IRLS) fitting. See Example 88.2 for an application of IRLS fitting.

NOITPRINT

suppresses the display of the "Iteration History" table.

NOPRINT

suppresses the display of the output. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OUTEST=*SAS-data-set*

specifies an output data set that contains the parameter estimates produced at each iteration. See the section "Output Data Sets" for details. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PLOTS < (*global-plot-option*) > < = (*plot-request*< (*options*) > < . . . *plot-request*< (*options*) > >) >

controls most of the plots that are produced through ODS Graphics (other plots are controlled by the BOOTSTRAP and PROFILE statements). When you specify only one *plot-request*, you can omit the parentheses around it. Here are some examples:

```
plots
plots        = none
plots        = diagnostics(unpack)
plots        = fit(stats=none)
plots        = residuals(residualtype=proj unpack smooth)
plots(stats=all)  = (diagnostics(stats=(maxincurv maxpecurv)) fit)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc nlin plots=diagnostics(stats=all);
   model y = alpha - beta*(gamma**x);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled and if you specify the PLOTS option without any *global-plot-option* or *plot-requests*, PROC NLIN produces the plots listed in Table 88.2 with the default set of statistics and options. If you do not specify the PLOTS option, PROC NLIN does not produce any of these graphs.

Table 88.2 Graphs Produced When the PLOTS Option Is Specified

Plot	Conditional On
ContourFitPlot	Model with two regressors
FitDiagnosticsPanel	Unconditional
FitPlot	Model with one regressor
LeveragePlot	Unconditional
LocalInfluencePlot	Unconditional
ResidualPanel	Unconditional

You can request additional plots by specifying *plot-requests*. For a listing of all the plots that PROC NLIN produces, see the section "ODS Graphics" on page 7191. Each *global-plot-option* applies to all plots that the procedure generates, except for plots that are controlled by the BOOTSTRAP and PROFILE statements. The *global-plot-option* can be overridden by a specific *option* after a *plot-request*.

You can specify the following *global-plot-options*:

MAXPOINTS=*value*
specifies the threshold value to produce a heat map instead of a scatter plot for the fit plot. By default, MAXPOINTS=5000.

RESIDUALTYPE=RAW | PROJ | BOTH
specifies the residual type to be plotted in the fit diagnostics and residual plots. RESIDUALTYPE=RAW requests that only the ordinary residuals be included in the plots; RESIDUALTYPE=PROJ sets the choice to projected residuals. By default, both residual types are included; you can also include both types by specifying RESIDUALTYPE=BOTH. See the section "Residuals in Nonlinear Regression" on page 7171 for details about the properties of ordinary and projected residuals in nonlinear regression.

STATS=ALL | DEFAULT | NONE | (*plot-statistics*)

requests the statistics to be included in all plots, except the ResidualPlots and the unpacked diagnostics plots. Table 88.3 lists the statistics that you can request. STATS=ALL requests all these statistics, STATS=NONE suppresses all statistics, and STATS=DEFAULT selects the default statistics. You request statistics in addition to the default set by including the keyword DEFAULT in the *plot-statistics* list.

Table 88.3 Statistics Available in Plots

Keyword	Default	Description
DEFAULT		All default statistics
MAXINCURV		Maximum intrinsic curvature
MAXPECURV		Maximum parameter-effects curvature
MSE	x	Mean square error, estimated or set by the SIGSQ option
NOBS	x	Number of observations used
NPARM	x	Number of parameters in the model
PVAR	x	Estimate of residual variance by using the projected residuals
RMSINCURV		Root mean square intrinsic curvature
RMSPECURV		Root mean square parameter-effects curvature
VAR	x	Estimate of residual variance by using the ordinary residuals

Along with the maximum intrinsic and parameter-effects curvatures, the critical curvature (CURVCRIT) value—that is, $1/\sqrt{F}$, where $F = F(p, n-p; \alpha)$—is also displayed. You do not need to specify any option for it. See the section "Relative Curvature Measures of Nonlinearity" on page 7170 for details about curvature measures of nonlinearity.

UNPACK

suppresses paneling.

You can specify the following *plot-requests* in the PLOTS= option:

ALL

produces all appropriate plots.

NONE

suppresses all plots.

DIAGNOSTICS < (*diagnostics-options*) >

produces a summary panel of fit diagnostics, leverage plots, and local-influence plots. The fit diagnostics panel includes the following plots:

- histogram of the ordinary residuals
- histogram of the projected residuals
- response variable values versus the predicted values
- expectation or mean of the ordinary residuals versus the predicted values
- ordinary and projected residuals versus the predicted values

- standardized ordinary and projected residuals versus the predicted values
- standardized ordinary and projected residuals versus the tangential leverage
- standardized ordinary and projected residuals versus the Jacobian leverage
- box plot of the ordinary and projected residuals if you specify the STATS=NONE suboption

The leverage and local influence plots are produced separately. The leverage plot is an index plot of the tangential and Jacobian leverages (by observation), and the local-influence plot contains the local influence by observation for a perturbation of the response variable. See the sections "Leverage in Nonlinear Regression" on page 7170 and "Local Influence in Nonlinear Regression" on page 7171 for details about leverages and local influence in nonlinear regression.

You can specify the following *diagnostics-options*:

LABELOBS
specifies that the leverage and local-influence plots be labeled with the observation number. Only these two plots are affected by this option.

LEVERAGETYPE=TAN | JAC | BOTH
specifies the leverage type to be plotted in the leverage plot. LEVERAGETYPE=TAN specifies that only the tangential leverage be included in the leverage plot, and LEVERAGETYPE=JAC specifies that only the Jacobian leverage be included. By default, both are displayed in the leverage plot. You can achieve the same result by specifying LEVERAGETYPE=BOTH. Only the leverage plot is affected by this option.

RESIDUALTYPE=RAW | PROJ | BOTH
specifies the residual type to be plotted in the panel. See the RESIDUALTYPE= *global-plot-option* for details. This *diagnostics-option* overrides the PLOTS RESIDUALTYPE *global-plot-option*. Only the plots that overlay both ordinary and projected residuals in the same plot are affected by this option.

STATS=*stats-options*
determines which statistics are included in the panel. See the STATS= *global-plot-option* for details. This *diagnostics-option* overrides the PLOTS STATS *global-plot-option*.

UNPACK
produces the plots in the diagnostics panel as individual plots. The statistics panel is not included in the individual plots, even if you specify the STATS= *global-plot-option* or the STATS= *diagnostics-option* or both.

FITPLOT | FIT < (*fit-options***) >**
produces, depending on the number of regressors, a scatter or contour fit plot. For a single-regressor model, a scatter plot of the data overlaid with the regression curve, confidence, and prediction bands is produced. However, if the number of observations exceeds 5,000, then PROC NLIN produces a heat map instead of a scatter plot. You can change this threshold value by using the MAXPOINTS= *global-plot-option*. For two-regressor models, a contour fit plot of the model with overlaid data is produced. If the model contains more than two regressors, no fit plot is produced.

You can specify the following *fit-options*:

NOCLI
suppresses the prediction limits for single-regressor models.

NOCLM
suppresses the confidence limits for single-regressor models.

NOLIMITS
suppresses the confidence and prediction limits for single-regressor models.

OBS=GRADIENT | NONE | OUTLINE | OUTLINEGRADIENT
controls how the observations are displayed. The *suboptions* are as follows:

GRADIENT specifies that observations be displayed as circles colored by the observed response. The same color gradient is used to display the fitted surface and the observations. Observations for which the predicted response is close to the observed response have similar colors—the greater the contrast between the color of an observation and the surface, the larger the residual is at that point. OBS=GRADIENT is the default.

NONE suppresses the observations.

OUTLINE specifies that observations be displayed as circles with a border but with a completely transparent fill.

OUTLINEGRADIENT is the same as OBS=GRADIENT except that a border is shown around each observation. This option is useful for identifying the location observations for which the residuals are small, because at these points the color of the observations and the color of the surface are indistinguishable.

CONTLEG
specifies that a continuous legend be included in the contour fit plot of a two-regressor model.

STATS=*stats-options*
determines which model fit statistics are included in the panel. See the STATS= *global-plot-option* for details. This *fit-option* overrides the PLOTS STATS *global-plot-option*.

RESIDUALS < (*residual-options***) >**
produces panels of the ordinary and projected residuals versus the regressors in the model. Each panel contains at most six plots, and multiple panels are used in the case where there are more than six regressors in the model.

The following *residual-options* are available:

RESIDUALTYPE=RAW | PROJ | BOTH
specifies the residual type to be plotted in the panel. See the RESIDUALTYPE= *global-plot-option* for details. This *residual-option* overrides the PLOTS RESIDUALTYPE *global-plot-option*.

SMOOTH
requests a nonparametric smooth of the residuals for each regressor. Each nonparametric fit is a loess fit that uses local linear polynomials, linear interpolation, and a smoothing parameter selected that yields a local minimum of the corrected Akaike information criterion (AICC). See Chapter 78, "The LOESS Procedure," for details.

UNPACK
suppresses paneling.

PRINT
displays the result of each statement in the program as it is executed. This option is a debugging feature that produces large amounts of output and is normally not needed.

RHO=*value*
specifies a value that controls the step-size search. By default RHO=0.1, except when METHOD=MARQUARDT. In that case, RHO=10. See the section "Step-Size Search" on page 7185 for more details.

SAVE
specifies that, when the iteration limit is exceeded, the parameter estimates from the final iteration be output to the OUTEST= data set. These parameter estimates are associated with the observation for which _TYPE_="FINAL". If you omit the SAVE option, the parameter estimates from the final iteration are not output to the data set unless convergence has been attained.

SIGSQ=*value*
specifies a value to use as the estimate of the residual variance in lieu of the estimated mean-square error. This value is used in computing the standard errors of the estimates. Fixing the value of the residual variance can be useful, for example, in maximum likelihood estimation.

SINGULAR=*s*
specifies the singularity criterion, *s*, which is the absolute magnitude of the smallest pivot value allowed when inverting the Hessian or the approximation to the Hessian. The default value is 1E4 times the machine epsilon; this product is approximately 1E-12 on most computers.

SMETHOD=HALVE | GOLDEN | CUBIC
specifies the step-size search method. The default is SMETHOD=HALVE. See the section "Step-Size Search" on page 7185 for details.

TAU=*value*
specifies a value that is used to control the step-size search. The default is TAU=1, except when METHOD=MARQUARDT. In that case the default is TAU=0.01. See the section "Step-Size Search" on page 7185 for details.

TOTALSS
adds to the analysis of variance table the uncorrected total sum of squares in models that have an (implied) intercept, and adds the corrected total sum of squares in models that do not have an (implied) intercept.

TRACE
displays the result of each operation in each statement in the model program as it is executed, in addition to the information displayed by the FLOW and PRINT options. This debugging option is needed very rarely, and it produces even more output than the FLOW and PRINT options.

XREF
displays a cross-reference of the variables in the model program showing where each variable is referenced or given a value. The XREF listing does not include derivative variables.

UNCORRECTEDDF
specifies that no degrees of freedom be lost when a bound is active. When the UNCORRECTEDDF option is not specified, an active bound is treated as if a restriction were applied to the set of parameters, so one parameter degree of freedom is deducted.

BOOTSTRAP Statement

BOOTSTRAP <*/ options*> **;**

A BOOTSTRAP statement requests bootstrap estimation of confidence intervals, the covariance matrix, and the correlation matrix of parameter estimates. To produce the plots that are controlled by the BOOTSTRAP statement, ODS Graphics must be enabled. If the main data set contains observations that PROC NLIN deems unusable, the procedure issues a message that these observations are excluded from the bootstrap resampling. PROC NLIN ignores the BOOTSTRAP statement for nonconvergent and singular models.

Table 88.4 summarizes the *options* available in the BOOTSTRAP statement.

Table 88.4 Summary of Options in BOOTSTRAP Statement

Option	Description
BOOTCI	Produces bootstrap confidence intervals of the parameters
BOOTCORR	Produces a bootstrap correlation matrix estimate table
BOOTCOV	Produces a bootstrap covariance matrix estimate table
BOOTDATA=	Specifies the bootstrap output data set
BOOTPLOTS	Produces plots of the bootstrap parameter estimates
DGP=	Specifies the bootstrap data generating process (DGP)
NSAMPLES=	Specifies the number of bootstrap sample data sets (replicates)
SEED=	Provides the seed that initializes the random number stream

BOOTCI <(BC | NORMAL | PERC | ALL)>
produces bootstrap-based confidence intervals for the parameters and adds columns that contain these values to the "Parameter Estimates" table. You can specify the following types of bootstrap confidence intervals:

BC produces bias-corrected confidence intervals.

NORMAL produces confidence intervals based on the assumption that bootstrap parameter estimates follow a normal distribution.

PERC produces percentile-based confidence intervals.

ALL produces all three confidence intervals.

The ALPHA= option in the PROC NLIN statement sets the level of significance that is used in constructing these bootstrap confidence intervals. By default, without the BOOTCI option, PROC NLIN produces bias-corrected (BC) confidence intervals and adds a column that contains the standard deviation of the bootstrap parameter estimates to the "Parameter Estimates" table. For more information, see the section "Bootstrap Resampling and Estimation" on page 7173.

BOOTCORR

produces the "Bootstrap Correlation Matrix" table, which contains a bootstrap estimate of the correlation matrix of the parameter estimates. For more information, see the section "Bootstrap Resampling and Estimation" on page 7173.

BOOTCOV

produces the "Bootstrap Covariance Matrix" table, which contains a bootstrap estimate of the covariance matrix of the parameter estimates. For more information, see the section "Bootstrap Resampling and Estimation" on page 7173.

BOOTDATA=*SAS-data-set*

specifies the SAS data set that contains the bootstrap sample data when you use a BOOTSTRAP statement. For more information about this data set, see the section "Output Data Sets" on page 7185.

BOOTPLOTS < (HIST | SCATTER | ALL) >

produces ODS graphics of bootstrap parameter estimates. You can specify the following types of plots:

HIST produces histograms of the bootstrap parameter estimates.

SCATTER produces pairwise scatter plots of the bootstrap parameter estimates.

ALL produces both plots.

By default, if ODS Graphics is enabled, PROC NLIN produces histograms of the bootstrap parameter estimates.

DGP=RESIDUAL < (*scaling-option***) > | WILD**

specifies the bootstrap data generating process (DGP). DGP=RESIDUAL requests the residual bootstrap, and DGP=WILD requests the wild bootstrap. The *scaling-option* determines the type of residual scaling to be performed for DGP=RESIDUAL.

Table 88.5 Scaling Options Available for DGP=RESIDUAL

Keyword	Description
ADJSSE	Simple uniform scaling
JAC	Scaling based on Jacobian leverage
RAW	No scaling
TAN	Scaling based on tangential leverage

By default, if the BOOTSTRAP statement is specified with no DGP= option or if no *scaling-option*

is specified for DGP=RESIDUAL, PROC NLIN performs a residual bootstrap with simple scaling (ADJSSE) for unweighted least squares and a wild bootstrap (WILD) for weighted least squares. For more information, see the section "Bootstrap Resampling and Estimation" on page 7173.

NSAMPLES=*n*

specifies the number of bootstrap sample data sets (replicates). By default, NSAMPLES=1000. For more information, see the section "Bootstrap Resampling and Estimation" on page 7173.

SEED=*n*

provides the seed that initializes the random number stream for generating the bootstrap sample data sets (replicates). If you do not specify the SEED= value or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer's clock. The largest possible value for the seed is $2^{31} - 1$. The _SEED_ in the data set that is produced by the BOOTDATA= option contains the value that is used as the initial seed for a particular replicate.

You can use the SYSRANDOM and SYSRANEND macro variables after a PROC NLIN run to query the initial and final seed values. However, using the final seed value as the starting seed for a subsequent analysis does not continue the random number stream where the previous analysis left off. The SYSRANEND macro variable provides a mechanism to pass on seed values to ensure that the sequence of random numbers is the same every time you run an entire program.

BOUNDS Statement

BOUNDS *inequality* <, . . . , *inequality*> ;

The BOUNDS statement restricts the parameter estimates so that they lie within specified regions. In each BOUNDS statement, you can specify a series of boundary values separated by commas. The series of bounds is applied simultaneously. Each boundary specification consists of a list of parameters, an inequality comparison operator, and a value. In a single-bounded expression, these three elements follow one another in the order described. The following are examples of valid single-bounded expressions:

```
bounds a1-a10 <= 20;
bounds c > 30;
bounds a b c > 0;
```

Multiple-bounded expressions are also permitted. For example:

```
bounds 0 <= B<= 10;
bounds 15 < x1 <= 30;
bounds r <= s <= p < q;
```

If you need to restrict an expression involving several parameters (for example, $\alpha + \beta < 1$), you can reparameterize the model so that the expression becomes a parameter or so that the boundary constraint can be expressed as a simple relationship between two parameters. For example, the boundary constraint $\alpha + \beta < 1$ in the model

```
model y = alpha + beta*x;
```

can be achieved by parameterizing $\theta = 1 - \beta$ as follows:

```
bounds alpha < theta;
model y = alpha + (1-theta)*x;
```

Lagrange multipliers are reported for all bounds that are enforced (active) when the estimation terminates. In the "Parameter Estimates" table, the Lagrange multiplier estimates are identified with names *Bound1*, *Bound2*, and so forth. An active bound is treated as if a restriction were applied to the set of parameters so that one parameter degree of freedom is deducted. You can use the UNCORRECTEDDF option to prevent the loss of degrees of freedom when bounds are active.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC NLIN to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the NLIN procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CONTROL Statement

CONTROL *variable* <**=***values*> <*. . . variable* <**=***values*>> **;**

The CONTROL statement declares control variables and specifies their values. A control variable is like a retained variable (see the section "RETAIN Statement" on page 7164) except that it is retained *across* iterations, and the derivative of the model with respect to a control variable is always zero.

CONTRAST Statement

CONTRAST '*label*' *expression* < **,** *expression* > < *option* > **;**

The CONTRAST statement enables you to conduct a statistical test that several expressions simultaneously equal zero. The expressions are typically contrasts—that is, differences whose expected values equal zero under the hypothesis of interest.

In the CONTRAST statement, you must provide a quoted string to identify the contrast and then a list of valid SAS expressions separated by commas. Multiple CONTRAST statements are permitted, and results from all statements are listed in a common table. PROC NLIN constructs approximate F tests for each statement by using the delta method (Cox 1998) to approximate the variance-covariance matrix of the constituent expressions.

You can specify the following *option*:

DF=*d*

specifies the denominator degrees of freedom to be used in computing p values for the F statistics. By default, the value of *d* is the value of the DF= option in the PROC NLIN statement.

DER Statement

DER. *parameter***=***expression* **;**

DER. *parameter.parameter***=***expression* **;**

The DER statement specifies first or second partial derivatives. By default, analytical derivatives are automatically computed. However, you can specify the derivatives yourself by using the DER.parm syntax. Use the first form shown to specify first partial derivatives, and use the second form to specify second partial derivatives. Note that the DER.parm syntax is retained for backward compatibility. The automatic analytical derivatives are, in general, a better choice. For additional information about automatic analytical derivatives, see the section "Automatic Derivatives" on page 7166.

For most of the computational methods, you need only specify the first partial derivative with respect to each parameter to be estimated. For the NEWTON method, specify both the first and the second derivatives. If any needed derivatives are not specified, they are automatically computed.

The expression can be an algebraic representation of the partial derivative of the expression in the MODEL statement with respect to the parameter or parameters that appear on the left side of the DER statement. Numerical derivatives can also be used. The expression in the DER statement must conform to the rules for a valid SAS expression, and it can include any quantities that the MODEL statement expression contains.

ESTIMATE Statement

ESTIMATE '*label*' *expression* < *options* > **;**

The ESTIMATE statement enables you to compute an additional estimate that is a function of the parameter values. You must provide a quoted string to identify the estimate and then a valid SAS expression. Multiple ESTIMATE statements are permitted, and results from all statements are listed in a common table. PROC NLIN computes approximate standard errors for the estimates by using the delta method (Billingsley 1986). It uses these standard errors to compute corresponding t statistics, p-values, and confidence limits.

If you specify the ECOV option in the PROC NLIN statement, the procedure also produces a table that contains the approximate covariance matrix of all the additional estimates you specify. Similarly, the ECORR option produces a table of the corresponding correlation matrix, and the EDER option produces a table of the derivatives of the additional estimates with respect to each of the model parameters.

You can specify the following *options*:

ALPHA=α
specifies the alpha level to be used in computing confidence limits. By default, the value of α is the value of the ALPHA= option in the PROC NLIN statement.

DF=*d*
specifies the degrees of freedom to be used in computing p-values and confidence limits. By default, the value of *d* is the value of the DF= option in the PROC NLIN statement.

ID Statement

ID *variables* **;**

The ID statement specifies additional variables to place in the output data set created by the OUTPUT statement. Any variable on the left side of any assignment statement is eligible. Also, the special variables created by the procedure can be specified. Variables in the input data set do not need to be specified in the ID statement since they are automatically included in the output data set.

MODEL Statement

MODEL *dependent* **=** *expression* **;**

The MODEL statement defines the prediction equation by declaring the dependent variable and defining an expression that evaluates predicted values. The expression can be any valid SAS expression that yields a numeric result. The expression can include parameter names, variables in the data set, and variables created by programming statements in the NLIN procedure. Any operators or functions that can be used in a DATA step can also be used in the MODEL statement.

A statement such as

```
model y=expression;
```

is translated into the form

model.y=*expression*;

using the compound variable name model.y to hold the predicted value. You can use this assignment directly as an alternative to the MODEL statement. Either a MODEL statement or an assignment to a compound variable such as model.y must appear.

OUTPUT Statement

OUTPUT OUT=*SAS-data-set keyword***=***names* < . . . *keyword***=***names* > < / *options* > **;**

The OUTPUT statement specifies an output data set to contain statistics calculated for each observation. For each statistic, specify the *keyword*, an equal sign, and a variable name for the statistic in the output data set. All of the names appearing in the OUTPUT statement must be valid SAS names, and none of the new variable names can match a variable already existing in the data set to which PROC NLIN is applied.

If an observation includes a missing value for one of the independent variables, both the predicted value and the residual value are missing for that observation. If the iterations fail to converge, all the values of all the variables named in the OUTPUT statement are missing values.

Table 88.6 summarizes the *options* available in the OUTPUT statement.

Table 88.6 OUTPUT Statement Options

Option	Description
Output Data Set and OUTPUT Statement Options	
ALPHA=	Specifies the level of significance α
DER	Saves the first derivatives of the model
OUT=	Specifies the output data set
Keyword Options	
H=	Specifies the tangential leverage
J=	Specifies the Jacobian leverage
L95=	Specifies the lower bound of an approximate 95% prediction interval
L95M=	Specifies the lower bound of an approximate 95% confidence interval for the mean
LCL=	Specifies the lower bound of an approximate $100(1-\alpha)\%$ prediction interval
LCLM=	Specifies the lower bound of an approximate $100(1-\alpha)\%$ confidence interval for the mean
LMAX=	Specifies the direction of maximum local influence
PARMS=	Specifies the parameter estimates
PREDICTED=	Specifies the predicted values
PROJRES=	Specifies the projected residuals
PROJSTUDENT=	Specifies the standardized projected residuals
RESEXPEC=	Specifies the mean of the residuals
RESIDUAL=	Specifies the residuals
SSE=	Specifies the residual sum of squares

Table 88.6 *continued*

Option	Description
STDI=	Specifies the standard error of the individual predicted value
STDP=	Specifies the standard error of the mean predicted value
STDR=	Specifies the standard error of the residual
STUDENT=	Specifies the standardized residuals
U95=	Specifies the upper bound of an approximate 95% prediction interval
U95M=	Specifies the upper bound of an approximate 95% confidence interval for the mean
UCL=	Specifies the upper bound of an approximate $100(1-\alpha)\%$ prediction interval
UCLM=	Specifies the upper bound of an approximate $100(1-\alpha)\%$ confidence interval for the mean
WEIGHT=	Specifies the special variable _WEIGHT_

You can specify the following *options* in the OUTPUT statement. For a description of computational formulas, see Chapter 4, "Introduction to Regression Procedures."

OUT=*SAS-data-set*
specifies the SAS data set to be created by PROC NLIN when an OUTPUT statement is included. The new data set includes the variables in the input data set. Also included are any ID variables specified in the ID statement, plus new variables with names that are specified in the OUTPUT statement.

The following values can be calculated and output to the new data set.

H=*name*
specifies a variable that contains the tangential leverage. See the section "Leverage in Nonlinear Regression" on page 7170 for details.

J=*name*
specifies a variable that contains the Jacobian leverage. See the section "Leverage in Nonlinear Regression" on page 7170 for details.

L95=*name*
specifies a variable that contains the lower bound of an approximate 95% confidence interval for an individual prediction. This includes the variance of the error as well as the variance of the parameter estimates. See also the description for the U95= option later in this section.

L95M=*name*
specifies a variable that contains the lower bound of an approximate 95% confidence interval for the expected value (mean). See also the description for the U95M= option later in this section.

LCL=*name*
specifies a variable that contains the lower bound of an approximate $100(1-\alpha)\%$ confidence interval for an individual prediction. The α level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the value of the ALPHA= option in the PROC NLIN statement. If neither of these options is specified, then $\alpha = 0.05$ by default, resulting in a lower bound for an approximate 95% confidence interval. For the corresponding upper bound, see the UCL keyword.

LCLM=*name*

specifies a variable that contains the lower bound of an approximate $100(1-\alpha)\%$ confidence interval for the expected value (mean). The α level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the value of the ALPHA= option in the PROC NLIN statement. If neither of these options is specified, then $\alpha = 0.05$ by default, resulting in a lower bound for an approximate 95% confidence interval. For the corresponding lower bound, see the UCLM keyword.

LMAX=*name*

specifies a variable that contains the direction of maximum local influence of an additive perturbation of the response variable. See the section "Local Influence in Nonlinear Regression" on page 7171 for details.

PARMS=*names*

specifies variables in the output data set that contains parameter estimates. These can be the same variable names that are listed in the PARAMETERS statement; however, you can choose new names for the parameters identified in the sequence from the parameter estimates table. A note in the log indicates which variable in the output data set is associated with which parameter name. Note that, for each of these new variables, the values are the same for every observation in the new data set.

PREDICTED=*name*

P=*name*

specifies a variable in the output data set that contains the predicted values of the dependent variable.

PROJRES=*name*

specifies a variable that contains the projected residuals obtained by projecting the residuals (ordinary residuals) into the null space of $(X|H)$. For the ordinary residuals, see the RESIDUAL= option later in this section. The section "Residuals in Nonlinear Regression" on page 7171 describes the statistical properties of projected residuals in nonlinear regression.

PROJSTUDENT=*name*

specifies a variable that contains the standardized projected residuals. See the section "Residuals in Nonlinear Regression" on page 7171 for details and the STUDENT= option later in this section.

RESEXPEC=*name*

specifies a variable that contains the mean of the residuals. In contrast to linear regressions where the mean of the residuals is zero, in nonlinear regression the residuals have a nonzero mean value and show a negative covariance with the mean response. See the section "Residuals in Nonlinear Regression" on page 7171 for details.

RESIDUAL=*name*

R=*name*

specifies a variable in the output data set that contains the residuals. See also the description of PROJRES= option stated previously in this section and the section "Residuals in Nonlinear Regression" on page 7171 for the statistical properties of residuals and projected residuals.

SSE=*name*

ESS=*name*

specifies a variable in the output data set that contains the residual sum of squares finally determined by the procedure. The value of the variable is the same for every observation in the new data set.

STDI=*name*

specifies a variable that contains the standard error of the individual predicted value.

STDP=*name*

specifies a variable that contains the standard error of the mean predicted value.

STDR=*name*

specifies a variable that contains the standard error of the residual.

STUDENT=*name*

specifies a variable that contains the standardized residuals. These are residuals divided by their estimated standard deviation. See the PROJSTUDENT= option defined previously in this section and the section "Residuals in Nonlinear Regression" on page 7171 for the statistical properties of residuals and projected residuals.

U95=*name*

specifies a variable that contains the upper bound of an approximate 95% confidence interval for an individual prediction. See also the description for the L95= option.

U95M=*name*

specifies a variable that contains the upper bound of an approximate 95% confidence interval for the expected value (mean). See also the description for the L95M= option.

UCL=*name*

specifies a variable that contains the upper bound of an approximate $100(1-\alpha)\%$ confidence interval an individual prediction. The α level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the value of the ALPHA= option in the PROC NLIN statement. If neither of these options is specified, then $\alpha = 0.05$ by default, resulting in an upper bound for an approximate 95% confidence interval. For the corresponding lower bound, see the LCL keyword.

UCLM=*name*

specifies a variable that contains the upper bound of an approximate $100(1-\alpha)\%$ confidence interval for the expected value (mean). The α level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the value of the ALPHA= option in the PROC NLIN statement. If neither of these options is specified, then $\alpha = 0.05$ by default, resulting in an upper bound for an approximate 95% confidence interval. For the corresponding lower bound, see the LCLM keyword.

WEIGHT=*name*

specifies a variable in the output data set that contains values of the special variable _WEIGHT_.

You can specify the following *options* in the OUTPUT statement after a slash (/) :

ALPHA=α

specifies the level of significance α for $100(1-\alpha)\%$ confidence intervals. By default, α is equal to the value of the ALPHA= option in the PROC NLIN statement or 0.05 if that option is not specified. You can supply values that are strictly between 0 and 1.

DER

saves the first derivatives of the model with respect to the parameters to the OUTPUT data set. The derivatives are named DER_parmname, where "parmname" is the name of the model parameter in your NLIN statements. You can use the DER option to extract the $\mathbf{X} = \partial\mathbf{f}/\partial\boldsymbol{\beta}$ matrix into a SAS data set. For example, the following statements create the data set nlinX, which contains the $\mathbf{X}$ matrix:

```
proc nlin;
   parms theta1=155 theta3=0.04;
   model V = theta1*c / (theta3 + c);
   output out=nlinout / der;
run;

data nlinX;
   set nlinout(keep=DER_:);
run;
```

The derivatives are evaluated at the final parameter estimates.

PARAMETERS Statement

PARAMETERS < *parameter-specification* > < **,** . . . **,** *parameter-specification* >
< / **PDATA=***SAS-data-set* > **;**

PARMS < *parameter-specification* > < **,** . . . **,** *parameter-specification* >
< / **PDATA=***SAS-data-set* > **;**

A PARAMETERS (or PARMS) statement is required. The purpose of this statement is to provide starting values for the NLIN procedure. You can provide values that define a point in the parameter space or a set of points.

All parameters must be assigned starting values through the PARAMETERS statement. The NLIN procedure does not assign default starting values to parameters in your model that do not appear in the PARAMETERS statement. However, it is not necessary to supply parameters and starting values explicitly through a *parameter-specification*. Starting values can also be provided through a data set. The names assigned to parameters must be valid SAS names and must not coincide with names of variables in the input data set (see the DATA= option in the PROC NLIN statement). Parameters that are assigned starting values through the PARAMETERS statement can be omitted from the estimation, for example, if the expression in the MODEL statement does not depend on them.

Assigning Starting Values with Parameter-Specification

A *parameter-specification* has the general form

name **=** *value-list*

where *name* identifies the parameter and *value-list* provides the set of starting values for the parameter.

Very often, the *value-list* contains only a single value, but more general and flexible list specifications are possible:

m	a single value
$m1, m2, \ldots, mn$	several values
m TO n	a sequence where m equals the starting value, n equals the ending value, and the increment equals 1
m TO n BY i	a sequence where m equals the starting value, n equals the ending value, and the increment is i
$m1, m2$ TO $m3$	mixed values and sequences

When you specify more than one value for a parameter, PROC NLIN sorts the values in ascending sequence and removes duplicate values from the parameter list before forming the grid for the parameter search. If you specify several values for each parameter, PROC NLIN evaluates the model at each point on the grid. The iterations then commence from the point on the grid that yields the smallest objective function value.

For example, the following PARMS statement specifies five parameters and sets their possible starting values as shown in the table:

```
parms  b0 = 0
       b1 = 4 to 8
       b2 = 0 to .6 by .2
       b3 = 1, 10, 100
       b4 = 0, .5, 1 to 4;
```

Possible Starting Values

B0	B1	B2	B3	B4
0	4	0.0	1	0.0
	5	0.2	10	0.5
	6	0.4	100	1.0
	7	0.6		2.0
	8			3.0
				4.0

Residual sums of squares are calculated for each of the $1 \times 5 \times 4 \times 3 \times 6 = 360$ combinations of possible starting values. (Specifying a large grid of values can take considerable computing time.)

If you specify a starting value with a *parameter-specification*, any starting values provided for this parameter through the PDATA= data set are ignored. The *parameter-specification* overrides the information in the PDATA= data set. When you specify a BY statement, the same *parameter-specification* is applied in each

BY group. To vary starting values with BY groups, use the PDATA= option in the PARAMETERS statement as described in the following paragraphs.

Assigning Starting Values from a SAS Data Set

The PDATA= option in the PARAMETERS statement enables you to assign starting values for parameters through a SAS data set. The data set must contain at least two variables that identify the parameter and contain starting values, respectively. The *character* variable identifying the parameters must be named Parameter or Parm. The *numeric* variable containing the starting value must be named Estimate or Est. This enables you, for example, to use the contents of the ParameterEstimates table from one PROC NLIN run to supply starting values for a subsequent run, as in the following example:

```
proc nlin data=FirstData;
   parameters alpha=100 beta=3 gamma=4;
   < other NLIN statements >
   model y = ... ;
   ods output ParameterEstimates=pest;
run;

proc nlin data=SecondData;
   parameters / pdata=pest;
   Switch = 1/(1+gamma*exp(beta*log(dose)));
   model y = alpha*Switch;
run;
```

You can specify multiple values for a parameter in the PDATA= data set, and the parameters can appear in any order. The starting values are collected by parameter and arranged in ascending order, and duplicate values are removed. The parameter names in the PDATA= data set are not case sensitive. For example, the following DATA step defines starting values for three parameters and a starting grid with $1 \times 3 \times 1 = 3$ points:

```
data test;
   input Parameter $ Estimate;
   datalines;
alpha  100
 BETA   4
 beta   4.1
beta    4.2
beta    4.1
 gamma 30
;
```

If starting values for a parameter are provided through the PDATA= data set and through an explicit *parameter-specification*, the latter takes precedence.

When you specify a BY statement, you can control whether the same starting values are applied to each BY group or whether the starting values are varied. If the BY variables are not present in the PDATA= data set, the entire contents of the PDATA= data set are applied in each BY group. If the BY variables are present in the PDATA= data set, then BY-group-specific starting values are assigned.

PROFILE Statement

PROFILE <*. . . parameter*> <*/ options*> **;**

A PROFILE statement selects parameters for profiling to assess their nonlinear characteristics. It can also gauge the influence of each observation on the selected parameters. If none of the parameters is specified in the PROFILE statement, all parameters are selected for profiling. To produce the plots that are controlled by the PROFILE statement, ODS Graphics must be enabled. If there are observations in the main data set that PROC NLIN deems unusable, the procedure issues a message stating that these observations are excluded from the parameter profiling. PROC NLIN ignores PROFILE statements for nonconvergent and singular models.

Table 88.8 summarizes the *options* available in the PROFILE statement.

Table 88.8 Summary of Options in PROFILE Statement

Option	Description
ALL	Produces all profiling related plots and statistics
CONFCURV	Produces confidence curves for the listed parameters
JACKKNIFE	Produces plots of the absolute relative percentage change by observation for the listed parameters
PROFDATA=	Specifies the profile output data set
RANGE=	Specifies a profiling range
TPLOT	Produces profile *t* plots for the listed parameters

ALL

produces all appropriate plots for the parameters that are listed in the PROFILE statement. This option overrides all other *options*. If the PROFDATA= option is specified, the values of the corresponding statistics are written to the data set.

CONFCURV

produces confidence curves for each parameter listed in the PROFILE statement. If the PROFDATA= option is specified, the values of the corresponding statistics are written to the data set. See the section "Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates" on page 7172 for details.

JACKKNIFE

produces, for each parameter listed in the PROFILE statement, plots of absolute relative percentage change in the value of the parameter that results from a fit to a jackknife-resampled datum versus observation number. The jackknife resampling is performed with a leave-one-out method. If the PROFDATA= option is specified, the values of the corresponding statistics are written to the data set. See the section "Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates" on page 7172 for details.

PROFDATA=*SAS-data-set*

specifies the SAS data set to be created by PROC NLIN when a PROFILE statement is included. See the section "Output Data Sets" on page 7185 for details about this data set.

RANGE=γ_f *to* γ_t *by* γ_b

specifies a profiling range for each parameter listed in the PROFILE statement. γ_f, γ_t and γ_b represent the starting, final, and consecutive incremental values of the range in terms of the standard errors of the parameters to be profiled. If you do not specify the RANGE= option, PROC NLIN uses –2, 2, and 0.2 as default values for γ_f, γ_t, and γ_b, respectively. For example, if parameter β is in the PROFILE statement, then during the profiling of β, PROC NLIN performs approximately $(\gamma_t - \gamma_f)/\gamma_b$ number of constrained optimizations. At the ith constrained optimization, the value of β is constrained to β_i,

$$\beta_i = \hat{\beta} + \text{stderr}_\beta \left(\gamma_f + i \, \frac{(\gamma_t - \gamma_f)}{\gamma_b} \right)$$

and the rest of the parameters are freely optimized. In the preceding expression, $\hat{\beta}$ and stderr_β are the estimated value and standard error, respectively, for β. If parameter bounds are specified with BOUNDS statements, PROC NLIN enforces those bounds during the profiling optimizations.

NOTE: The profiling optimizations are performed using quasi-Newton algorithm. During these optimizations, any optimization options specified in the PROC NLIN statement and the _LOSS_ variable are ignored.

TPLOT

produces profile t plots for each parameter listed in the PROFILE statement. If the PROFDATA= option is specified, the values of the corresponding statistics are written to the data set. See the section "Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates" on page 7172 for details.

RETAIN Statement

RETAIN *variable* <=*values*> <. . . *variable* <=*values*>> ;

The RETAIN statement declares retained variables and specifies their values. A retained variable is like a control variable (see the section "CONTROL Statement" on page 7153) except that it is retained only *within* iterations. An iteration involves a single pass through the data set.

Other Programming Statements

PROC NLIN supports many statements that are similar to SAS programming statements used in a DATA step.

Several SAS programming statements can be used after the PROC NLIN statement. These statements can appear anywhere in the PROC NLIN code, but new variables must be created before they appear in other statements. For example, the following statements are valid since they create the variable temp before it is used in the MODEL statement:

```
proc nlin;
   parms b0=0 to 2 by 0.5 b1=0.01 to 0.09 by 0.01;
   temp = exp(-b1*x);
   model y=b0*(1-temp);
run;
```

The following statements result in missing values for y because the variable temp is undefined before it is used:

```
proc nlin;
   parms b0=0 to 2 by 0.5 b1=0.01 to 0.09 by 0.01;
   model y = b0*(1-temp);
   temp = exp(-b1*x);
run;
```

PROC NLIN can process assignment statements, explicitly or implicitly subscripted ARRAY statements, explicitly or implicitly subscripted array references, IF statements, SAS functions, and program control statements. You can use programming statements to create new SAS variables for the duration of the procedure. These variables are not included in the data set to which PROC NLIN is applied. Program statements can include variables in the DATA= data set, parameter names, variables created by preceding programming statements within PROC NLIN, and special variables used by PROC NLIN. The following SAS programming statements can be used in PROC NLIN:

ABORT;
ARRAY *arrayname* < **[** *dimensions* **]** > < **$** > < *variables-and-constants* >**;**
CALL *name* < **(***expression* < **,** *expression* . . . >**)** >**;**
DELETE;
DO < *variable* **=** *expression* < **TO** *expression* > < **BY** *expression* > >
< **,** *expression* < **TO** *expression* > < **BY** *expression* > > . . .
< **WHILE** *expression* > < **UNTIL** *expression* >**;**
END;
GOTO *statement-label***;**
IF *expression***;**
IF *expression* **THEN** *program-statement***;**
ELSE *program-statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement-label***;**
PUT < *variable* > < **=** > . . . **;**
RETURN;
SELECT < **(***expression***)** >**;**
STOP;
SUBSTR(*variable***,** *index***,** *length***)=** *expression***;**
WHEN (*expression***)***program-statement***;**
OTHERWISE *program-statement***;**

These statements can use the special variables created by PROC NLIN. See the section "Special Variables" on page 7176 for more information.

Details: NLIN Procedure

Automatic Derivatives

Depending on the optimization method you select, analytical first- and second-order derivatives are computed automatically. Derivatives can still be supplied using the DER.parm syntax. These DER.parm derivatives are not verified by the differentiator. If any needed derivatives are not supplied, they are computed and added to the programming statements. To view the computed derivatives, use the LISTDER or LIST option.

The following model is solved using Newton's method. Analytical first- and second-order derivatives are automatically computed. The compiled program code is shown in Figure 88.6.

```
proc nlin data=Enzyme method=newton list;
   parms x1=4 x2=2;
   model Velocity = x1 * exp (x2 * Concentration);
run;
```

Figure 88.6 Model and Derivative Code Output

The NLIN Procedure

Listing of Compiled Program Code		
Stmt	**Line:Col**	**Statement as Parsed**
1	1711:4	MODEL.Velocity = x1 * EXP(x2 * Concentration);
1	1711:4	@MODEL.Velocity/@x1 = EXP(x2 * Concentration);
1	1711:4	@MODEL.Velocity/@x2 = x1 * Concentration * EXP(x2 * Concentration);
1	1711:4	@@MODEL.Velocity/@x1/@x2 = Concentration * EXP(x2 * Concentration);
1	1711:4	@@MODEL.Velocity/@x2/@x1 = Concentration * EXP(x2 * Concentration);
1	1711:4	@@MODEL.Velocity/@x2/@x2 = x1 * Concentration * Concentration * EXP(x2 * Concentration);

Note that all the derivatives require the evaluation of EXP(X2 * Concentration). The actual machine-level code is displayed if you specify the LISTCODE option, as in the following statements:

```
proc nlin data=Enzyme method=newton listcode;
   parms x1=4 x2=2;
   model Velocity = x1 * exp (x2 * Concentration);
run;
```

Note that, in the generated code, only one exponentiation is performed (Figure 88.7). The generated code reuses previous operations to be more efficient.

Figure 88.7 LISTCODE Output

The NLIN Procedure

Code Listing		
1 Stmt MODEL	line 1718 column 4. (1) arg=MODEL.Velocity argsave=MODEL.Velocity	
	Source Text:	model Velocity = x1 * exp (x2 * Concentration);
Oper *	at 1718:34 (30,0,2).	* : ##dbl1 <- x2 Concentration
Oper EXP	at 1718:30 (103,0,1).	EXP : ##dbl2 <- ##dbl1
Oper *	at 1718:24 (30,0,2).	* : MODEL.Velocity <- x1 ##dbl2
Oper eeocf	at 1718:24 (18,0,1).	eeocf : _DER_ <- _DER_
Oper =	at 1718:24 (1,0,1).	= : @MODEL.Velocity/@x1 <- ##dbl2
Oper *	at 1718:30 (30,0,2).	* : @1dt1_1 <- Concentration ##dbl2
Oper *	at 1718:24 (30,0,2).	* : @MODEL.Velocity/@x2 <- x1 @1dt1_1
Oper =	at 1718:24 (1,0,1).	= : @@MODEL.Velocity/@x1/@x2 <- @1dt1_1
Oper =	at 1718:24 (1,0,1).	= : @@MODEL.Velocity/@x2/@x1 <- @1dt1_1
Oper *	at 1718:30 (30,0,2).	* : @2dt1_1 <- Concentration @1dt1_1
Oper *	at 1718:24 (30,0,2).	* : @@MODEL.Velocity/@x2/@x2 <- x1 @2dt1_1

Measures of Nonlinearity, Diagnostics and Inference

A "close-to-linear" nonlinear regression model, in the sense of Ratkowsky (1983, 1990), is a model in which parameter estimators have properties similar to those in a linear regression model. That is, the least squares estimators of the parameters are close to being unbiased and normally distributed, and they have minimum variance.

A nonlinear regression model sometimes fails to be close to linear due to the properties of one or several parameters. When this occurs, bias in the parameter estimates can render inferences that use the reported standard errors and confidence limits invalid.

PROC NLIN provides various measures of nonlinearity. To assess the nonlinearity of a model-data combination, you can use both of the following complementary sets of measures:

- Box's bias (Box 1971) and Hougaard's skewness (Hougaard 1982, 1985) of the least squares parameter estimates
- curvature measures of nonlinearity (Bates and Watts 1980).

Furthermore, PROC NLIN provides residual, leverage, and local-influence diagnostics (St. Laurent and Cook 1993).

In the following several sections, these nonlinearity measures and diagnostics are discussed. For this material, several basic definitions are required. Let $\mathbf{X}$ be the Jacobian matrix for the model, $\mathbf{X} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\beta}}$, and let $\mathbf{Q}$ and $\mathbf{R}$ be the components of the QR decomposition of $\mathbf{X} = \mathbf{QR}$ of $\mathbf{X}$, where $\mathbf{Q}$ is an $(n \times n)$ orthogonal matrix. Finally, let $\mathbf{B}$ be the inverse of the matrix constructed from the first p rows of the $(n \times p)$ dimensional matrix $\mathbf{R}$ (that is, $\mathbf{B} = \mathbf{R}_p^{-1}$). Next define

$$
\begin{aligned}
\left[\mathbf{H}_j\right]_{kl} &= \frac{\partial^2 \mathbf{f}_j}{\partial \boldsymbol{\beta}_k \partial \boldsymbol{\beta}_l} \\
\left[\mathbf{U}_j\right]_{kl} &= \sum_{mn} \mathbf{B}'_{km} \left[\mathbf{H}_j\right]_{mn} \mathbf{B}_{nl} \\
\left[\mathbf{A}_j\right]_{kl} &= \sqrt{p \times \mathrm{mse}} \sum_{m} \mathbf{Q}'_{jm} \left[\mathbf{U}_m\right]_{kl} \ ,
\end{aligned}
$$

where $\mathbf{H}$, $\mathbf{U}$ and the acceleration array $\mathbf{A}$ are three-dimensional $(n \times p \times p)$ matrices. The first p faces of the acceleration array constitute a $(p \times p \times p)$ parameter-effects array and the last $(n - p)$ faces constitute the $(n - p \times p \times p)$ intrinsic curvature array (Bates and Watts 1980). The previous and subsequent quantities are computed at the least squares parameter estimators.

Box's Measure of Bias

The degree to which parameter estimators exhibit close-to-linear behavior can be assessed with Box's bias (Box 1971) and Hougaard's measure of skewness (Hougaard 1982, 1985). The bias and percentage bias measures are available through the BIAS option in the PROC NLIN statement. Box's bias measure is defined as

$$
\widehat{\mathrm{E}}\left[\widehat{\beta} - \beta\right] = -\frac{\sigma^2}{2} \left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1} \sum_{i=1}^{n} w_i \mathbf{x}'_i \operatorname{Tr}\left(\left(\mathbf{X}'\mathbf{W}\mathbf{X}\right)^{-1} \left[\mathbf{H}_i\right]\right)
$$

where $\sigma^2 = \text{mse}$ if the SIGSQ option is not set. Otherwise, σ^2 is the value you set with the SIGSQ option. $\mathbf{W}$ is the diagonal weight matrix specified with the _WEIGHT_ variable (or the identity matrix if _WEIGHT_ is not defined) and $[\mathbf{H}_i]$ is the $(p \times p)$ Hessian matrix at the ith observation. In the case of unweighted least squares, the bias formula can be expressed in terms of the acceleration array $\mathbf{A}$,

$$\widehat{\mathrm{E}}\left[\widehat{\beta}_i - \beta_i\right] = -\frac{\sigma^2}{2p \times \text{mse}} \sum_{j,k=1}^{p} \mathbf{B}_{ij} \left[\mathbf{A}_j\right]_{kk}$$

As the preceding formulas illustrate, the bias depends solely on the parameter-effects array, thereby permitting its reduction through reparameterization. Example 88.4 shows how changing the parameterization of a four-parameter logistic model can reduce the bias. Ratkowsky (1983, p. 21) recommends that you consider reparameterization if the percentage bias exceeds 1%.

Hougaard's Measure of Skewness

In addition to Box's bias, Hougaard's measure of skewness, g_{1i} (Hougaard 1982, 1985), is also provided in PROC NLIN to assess the close-to-linear behavior of parameter estimators. This measure is available through the HOUGAARD option in the PROC NLIN statement. Hougaard's skewness measure for the ith parameter is based on the third central moment, defined as

$$\mathrm{E}\left[\widehat{\beta}_i - \mathrm{E}\left(\widehat{\beta}_i\right)\right]^3 = -\left(\sigma^2\right)^2 \sum_{jkl} [\mathbf{L}]_{ij} [\mathbf{L}]_{ik} [\mathbf{L}]_{il} \left(\left[\mathbf{V}_j\right]_{kl} + [\mathbf{V}_k]_{jl} + [\mathbf{V}_l]_{jk}\right)$$

where the sum is a triple sum over the number of parameters and

$$\mathbf{L} = \left(\mathbf{X}'\mathbf{X}\right)^{-1} = \left(\frac{\partial \mathbf{f}}{\partial \boldsymbol{\beta}'} \frac{\partial \mathbf{f}}{\partial \boldsymbol{\beta}}\right)^{-1}$$

The term $[\mathbf{L}]_{ij}$ denotes the value in row i, column j of the matrix $\mathbf{L}$. (Hougaard (1985) uses superscript notation to denote elements in this inverse.) The matrix $\mathbf{V}$ is a three-dimensional $(p \times p \times p)$ array

$$\left[\mathbf{V}_j\right]_{kl} = \sum_{m=1}^{n} \frac{\partial F_m}{\partial \beta_j} \frac{\partial^2 F_m}{\partial \beta_k \partial \beta_l}$$

The third central moment is then normalized using the standard error as

$$G_{1i} = \mathrm{E}\left[\widehat{\beta}_i - \mathrm{E}(\widehat{\beta}_i)\right]^3 / \left(\sigma^2 \times [\mathbf{L}]_{ii}\right)^{3/2}$$

The previous expressions depend on the unknown values of the parameters and on the residual variance σ^2. In order to evaluate the Hougaard measure in a particular data set, the NLIN procedure computes

$$g_{1i} = \widehat{\mathrm{E}}\left[\widehat{\beta}_i - \mathrm{E}(\widehat{\beta}_i)\right]^3 / \left(\text{mse} \times [\widehat{\mathbf{L}}]_{ii}\right)^{3/2}$$

$$\widehat{\mathrm{E}}\left[\widehat{\beta}_i - \mathrm{E}(\widehat{\beta}_i)\right]^3 = -\text{mse}^2 \sum_{jkl} [\widehat{\mathbf{L}}]_{ij} [\widehat{\mathbf{L}}]_{ik} [\widehat{\mathbf{L}}]_{il} \left([\widehat{\mathbf{V}}_j]_{kl} + [\widehat{\mathbf{V}}_k]_{jl} + [\widehat{\mathbf{V}}_l]_{jk}\right)$$

Following Ratkowsky (1990, p. 28), the parameter β_i is considered to be very close to linear, reasonably close, skewed, or quite nonlinear according to the absolute value of the Hougaard measure $|g_{1i}|$ being less than 0.1, between 0.1 and 0.25, between 0.25 and 1, or greater than 1, respectively.

Relative Curvature Measures of Nonlinearity

Bates and Watts (1980) formulated the maximum parameter-effects and maximum intrinsic curvature measures of nonlinearity to assess the close-to-linear behavior of nonlinear models. Ratkowsky (1990) notes that of the two curvature components in a nonlinear model, the parameter-effects curvature is typically larger. It is this component that you can affect by changing the parameterization of a model. PROC NLIN provides these two measures of curvature both through the STATS *plot-option* and through the NLINMEASURES option in the PROC NLIN statement.

The maximum parameter-effects and intrinsic curvatures are defined, in a compact form, as

$$\mathbf{T}^{\tau} = \max_{\beta} \left|\left|\beta' \mathbf{A}^{\tau} \beta\right|\right|$$
$$\mathbf{T}^{\eta} = \max_{\beta} \left|\left|\beta' \mathbf{A}^{\eta} \beta\right|\right|$$

where $\mathbf{T}^{\tau}$ and $\mathbf{T}^{\eta}$ denote the maximum parameter-effects and intrinsic curvatures, while $\mathbf{A}^{\tau}$ and $\mathbf{A}^{\eta}$ stand for the parameter-effects and intrinsic curvature arrays. The maximization is carried out over a unit-vector of the parameter values (Bates and Watts 1980). In line with Bates and Watts (1980), PROC NLIN takes 10^{-4} as the convergence tolerance for the maximum intrinsic and parameter-effects curvatures. Note that the preceding matrix products involve contraction of the faces of the three-dimensional acceleration arrays with the normalized parameter vector, β. The corresponding expressions for the RMS (root mean square) parameter-effects and intrinsic curvatures can be found in Bates and Watts (1980).

The statistical significance of $\mathbf{T}^{\tau}$ and $\mathbf{T}^{\eta}$ and the corresponding RMS values can be assessed by comparing these values with $1/\sqrt{F}$, where F is the upper $\alpha \times 100\%$ quantile of an F distribution with p and $n - p$ degrees of freedom (Bates and Watts 1980).

One motivation for fitting a nonlinear model in a different parameterization is to obtain a particular interpretation and to give parameter estimators more close-to-linear behavior. Example 88.4 shows how changing the parameterization of a four-parameter logistic model can reduce the parameter-effects curvature and can yield a useful parameter interpretation at the same time. In addition, Example 88.6 shows a nonlinear model with a high intrinsic curvature and the corresponding diagnostics.

Leverage in Nonlinear Regression

In contrast to linear regression, there are several measures of leverage in nonlinear regression. Furthermore, in nonlinear regression, the effect of a change in the ith response on the ith predicted value might depend on both the size of the change and the ith response itself (St. Laurent and Cook 1992). As a result, some observations might show superleverage —namely, leverages in excess of one (St. Laurent and Cook 1992).

PROC NLIN provides two measures of leverages: tangential and Jacobian leverages through the PLOTS option in the PROC NLIN statement and the H= and J= options of OUTPUT statement. Tangential leverage, $\mathbf{H}_i$, is based on approximating the nonlinear model with a linear model that parameterizes the tangent plane at the least squares parameter estimators. In contrast, Jacobian leverage, $\mathbf{J}_i$, is simply defined as the instantaneous rate of change in the ith predicted value with respect to the ith response (St. Laurent and Cook 1992).

The mathematical formulas for tangential and Jacobian leverages are

$$\mathbf{H}_i = w_i \mathbf{x}_i (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1} \mathbf{x}_i'$$
$$\mathbf{J}_i = w_i \mathbf{x}_i (\mathbf{X}'\mathbf{W}\mathbf{X} - [\mathbf{W}\,\mathbf{e}][\mathbf{H}])^{-1} \mathbf{x}_i' ,$$

where $\mathbf{e}$ is the vector of residuals, $\mathbf{W}$ is the diagonal weight matrix if you specify the special variable _WEIGHT_ and otherwise the identity matrix, i indexes the corresponding quantities for the ith observation, and the $\mathbf{X}$ matrix is defined in the section "Notation for Nonlinear Regression Models" on page 7133. The brackets [.][.] indicate column multiplication as defined in Bates and Watts (1980). The preceding formula for tangential leverage holds if the gradient, Marquardt, or Gauss methods are used. For the Newton method, the tangential leverage is set equal to the Jacobian leverage.

In a model with a large intrinsic curvature, the Jacobian and tangential leverages can be very different. In fact, the two leverages are identical only if the model provides an exact fit to the data ($\mathbf{e} = 0$) or the model is intrinsically linear (St. Laurent and Cook 1993). This is also illustrated by the leverage plot and nonlinearity measures provided in Example 88.6.

Local Influence in Nonlinear Regression

St. Laurent and Cook (1993) suggest using $l_{\max}$, the direction that yields the maximum normal curvature, to assess the local influence of an additive perturbation to the response variable on the estimation of the parameters and variance of a nonlinear model. Defining the normal curvature components

$$\mathbf{C}_\beta = \max_l \frac{2}{\sigma^2} l' \mathbf{J} l$$
$$\mathbf{C}_\sigma = \max_l \frac{4}{\sigma^2} l' \mathbf{P}_e l$$

where $\mathbf{J}$ is the Jacobian leverage matrix and $\mathbf{P}_e = \mathbf{e}\mathbf{e}'/(\mathbf{e}'\mathbf{e})$, you choose the $l_{\max}$ that results in the maximum of the two curvature components (St. Laurent and Cook 1993). PROC NLIN provides $l_{\max}$ through the PLOTS option in the PROC NLIN statement and the LMAX= option in the OUTPUT statement. Example 88.6 shows a plot of $l_{\max}$ for a model with high intrinsic curvature.

Residuals in Nonlinear Regression

If a nonlinear model is intrinsically nonlinear, using the residuals $\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}}$ for diagnostics can be misleading (Cook and Tsai 1985). This is due to the fact that in correctly specified intrinsically nonlinear models, the residuals have nonzero means and different variances, even when the original error terms have identical variances. Furthermore, the covariance between the residuals and the predicted values tends to be negative semidefinite, complicating the interpretation of plots based on $\mathbf{e}$ (Cook and Tsai 1985).

Projected residuals are proposed by Cook and Tsai (1985) to overcome these shortcomings of residuals, which are henceforth called raw (ordinary) residuals to differentiate them from their projected counterparts. Projected residuals have zero means and are uncorrelated with the predicted values. In fact, projected residuals are identical to the raw residuals in intrinsically linear models.

PROC NLIN provides raw and projected residuals, along with their standardized forms. In addition, the mean or expectation of the raw residuals is available. These can be accessed with the PLOTS option in the PROC NLIN statement and the OUTPUT statement options PROJRES=, PROJSTUDENT=, RESEXPEC=, RESIDUAL= and STUDENT=.

Denote the projected residuals by $\mathbf{e}_p$ and the expectation of the raw residuals by $\mathrm{E}[\mathbf{e}]$. Then

$$\mathbf{e}_p = (\mathbf{I}_n - \mathbf{P}_{xh})\,\mathbf{e}$$
$$\mathrm{E}[\mathbf{e}_i] = -\frac{\sigma^2}{2} \sum_{j=1}^{n} \tilde{\mathbf{P}}_{x,ij} \mathrm{Tr}\left([\mathbf{H}_j]\, (\mathbf{X}'\mathbf{X})^{-1} \right)$$

where $\mathbf{e}_i$ is the ith observation raw residual, $\mathbf{I}_n$ is an n-dimensional identity matrix, $\mathbf{P}_{xh}$ is the projector onto the column space of $(\mathbf{X}|\mathbf{H})$, and $\tilde{\mathbf{P}}_x = \mathbf{I}_n - \mathbf{P}_x$. The preceding formulas are general, and the projectors are defined accordingly to take the weighting into consideration. In unweighted least squares, $\mathrm{E}[\mathbf{e}]$ reduces to

$$\mathrm{E}[\mathbf{e}] = -\frac{1}{2}\sigma^2\tilde{\mathbf{Q}}\,\mathbf{a}$$

where $\tilde{\mathbf{Q}}$ is the last $n-p$ columns of the $\mathbf{Q}$ matrix in the QR decomposition of $\mathbf{X}$, and the $(n-p)$ dimensional vector $\mathbf{a}$ is defined in terms of the intrinsic acceleration array

$$\mathbf{a}_i = \sum_{j=1}^{p} \left[\mathbf{A}_{i+p}\right]_{jj}$$

The standardized raw residuals, which are denoted by $\tilde{\mathbf{e}}$, are defined as

$$\tilde{\mathbf{e}} = \frac{\sqrt{w_i}\,\mathbf{e}}{\sqrt{\sigma^2(1-\mathbf{P}_{x,ii})}}$$

where σ^2 is the residual variance that is estimated by using the mean square error.

Similarly, the standardized projected residuals, which are denoted by $\tilde{\mathbf{e}}_p$, are defined as

$$\tilde{\mathbf{e}}_p = \frac{\sqrt{w_i}\,\mathbf{e}_p}{\sqrt{\sigma^2(1-\mathbf{P}_{xh,ii})}}$$

where the estimate of σ^2 is computed by using the following formula (Cook and Tsai 1985):

$$\hat{\sigma}^2 = \frac{\mathbf{e}_p'\mathbf{e}_p}{\mathrm{Tr}\,(\mathbf{I}_n - \mathbf{P}_{xh})}$$

The use of raw and projected residuals for diagnostics in nonlinear regression is illustrated in Example 88.6.

Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates

The global measures of nonlinearity, which are discussed in the preceding section, are very useful for assessing the overall nonlinearity of the model. However, the impact of global nonlinearity on inference regarding subsets of the parameter set cannot be easily determined (Cook and Tsai 1985). The impact of the nonlinearity on the uncertainty of individual parameters can be efficiently described by profile t plots and confidence curves (Bates and Watts 1988; Cook and Weisberg 1990).

A profile t plot for parameter β is a plot of the likelihood ratio pivotal statistic, $\tau(\beta)$, and the corresponding Wald pivotal statistic, $\sigma(\beta)$ (Bates and Watts 1988). The likelihood ratio pivotal statistic is defined as

$$\tau(\beta) = \mathrm{sign}(\beta - \hat{\beta})\,\mathrm{L}(\beta)$$

where

$$\mathrm{L}(\beta) = \left(\frac{\mathrm{SSE}(\beta,\tilde{\Theta}) - \mathrm{SSE}(\hat{\beta},\hat{\Theta})}{\mathrm{mse}}\right)^{1/2}$$

where β is the profile parameter and Θ refers to the remaining parameters. $\mathrm{SSE}(\beta, \tilde{\Theta})$ is the sum of square errors where the profile parameter β is constrained at a given value and $\tilde{\Theta}$ is the least squares estimate of Θ conditional on a given value of β. In contrast, $\mathrm{SSE}(\hat{\beta}, \hat{\Theta})$ is the sum of square errors for the full model. For linear models, $\tau(\beta)$ matches $\sigma(\beta)$, which is defined as

$$\sigma(\beta) = \frac{\beta - \hat{\beta}}{\mathrm{stderr}_\beta}$$

where β is the constrained value, $\hat{\beta}$ is the estimated value for the parameter, and stderr_β is the standard error for the parameter. Usually a profile *t* plot is overlaid with a reference line that passes through the origin and has a slope of one. PROC NLIN follows this convention.

A confidence curve for a particular parameter is useful for validating Wald-based confidence intervals for the parameter against likelihood-based confidence intervals (Cook and Weisberg 1990). A confidence curve contains a scatter plot of the constrained parameter value versus $\mathrm{L}(\beta)$. The Wald-based confidence intervals are overlaid as two straight lines that pass through $(0, \hat{\beta})$ with a slope of $\pm\mathrm{stderr}_\beta$. Hence, for different levels of significance, you can easily compare the Wald-based confidence intervals against the corresponding confidence intervals that are based on the likelihood ratio. Cook and Weisberg (1990) recommend that you report a single Wald-based confidence interval only if there is a good agreement between the Wald and likelihood intervals up to at least the 99% confidence level.

Compared to local influence, the leave-one-out method performs a more complete analysis of the influence of observations on the values of parameter estimates. In this method, jackknife resampling removes each observation in turn and fits the model to the remaining data set. Hence, a data set with *n* observations will have *n* corresponding data sets with *n*–1 observations. The impact of each observation on a parameter is assessed by the absolute relative percentage change in the value of the parameter compared with the reference value from the full data.

Bootstrap Resampling and Estimation

Bootstrap resampling and estimation methods can be used to produce confidence intervals and covariance matrix estimates that have second-order, $\mathcal{O}(1/n)$ accuracy, where *n* is the number of observations (DiCiccio and Efron 1996). In contrast, the standard Wald-based confidence interval has first-order, $\mathcal{O}(1/\sqrt{n})$ accuracy. Bootstrap methods achieve this higher accuracy at the cost of an orders-of-magnitude increase in numerical computation compared to standard asymptotic approximations. However, dramatic increases in performance and decreases in the cost of numerical computation have made bootstrap methods very attractive for routine statistical inference (MacKinnon 2002).

PROC NLIN samples as many bootstrap sample data sets (replicates) as you specify in the NSAMPLES= option and performs least squares fit on each replicate. For each least squares fit, PROC NLIN uses the original parameter estimates as starting values for the model parameters. The statistics from the least squares fits that converge are collected and used to produce confidence intervals, covariance and correlation matrices, and histograms and scatter plots of the bootstrap parameter estimates.

Each replicate is obtained by sampling the residuals instead of the input data set. The sampled residuals are used to simulate the response by using a bootstrap data generating process (DGP). PROC NLIN's bootstrap DGP produces replicates that contain the same number of observations as the number of usable observations in the input data set. In fact, the bootstrap DGP for a particular replicate starts by discarding the input data set observations that PROC NLIN deems unusable during the original least squares estimation. The next step of the bootstrap DGP for a replicate can be represented by the formula

$$\tilde{Y}_i = f(\hat{\boldsymbol{\beta}}; \mathbf{z}_i') + \tilde{\epsilon}_i$$

where $\tilde{Y}_i$ is the ith simulated response, $\hat{\boldsymbol{\beta}}$ is the original least squares parameter estimate, $\mathbf{z}_i'$ is the ith regressor vector, and $\tilde{\epsilon}_i$ is the ith simulated error.

PROC NLIN makes several bootstrap DGP types available in the DGP= option of the BOOTSTRAP statement. These bootstrap DGP types differ only in how they obtain the ith simulated error, $\tilde{\epsilon}_i$. In general, $\tilde{\epsilon}_i$ can be represented as

$$\tilde{\epsilon}_i = \mathbf{s}_r \hat{\mathbf{e}}_r$$

where r is a uniform random integer between 1 and the number of usable observations, $\mathbf{s}_r$ is a scale factor that depends on the chosen bootstrap DGP options, and $\hat{\mathbf{e}}_r$ is the rth residual obtained from the rth usable observation of the original least squares fit. The scale factor, $\mathbf{s}_r$, that captures the differences among the various bootstrap DGP types takes one of the following values:

$$\mathbf{s}_r = \begin{cases} 1 & \text{if DGP=RESIDUAL(RAW)} \\ \sqrt{\frac{\mathbf{n}}{\mathbf{n}-\mathbf{p}}} & \text{if DGP=RESIDUAL(ADJSSE)} \\ \frac{1}{\sqrt{1-\mathbf{H}_r}} & \text{if DGP=RESIDUAL(TAN)} \\ \frac{1}{\sqrt{1-\mathbf{J}_r}} & \text{if DGP=RESIDUAL(JAC)} \\ \frac{\gamma}{\sqrt{1-\mathbf{H}_r}} & \text{if DGP=WILD} \end{cases}$$

In the preceding formula, $\mathbf{n}$ is the number of usable observations, $\mathbf{p}$ is the number of model parameters, and $\mathbf{H}_r$ and $\mathbf{J}_r$ are the rth tangential and Jacobian leverages, respectively. For the WILD bootstrap DGP (Wu 1986), which is the only bootstrap DGP type exclusively available for weighted least squares, γ is a random number given by

$$\gamma = \begin{cases} -\frac{\sqrt{5}-1}{2} & \text{with probability } \frac{\sqrt{5}+1}{2\sqrt{5}} \\ \frac{\sqrt{5}+1}{2} & \text{with probability } \frac{\sqrt{5}-1}{2\sqrt{5}} \end{cases}$$

PROC NLIN makes three types of bootstrap confidence intervals available in the BOOTCI option in the BOOTSTRAP statement. These confidence intervals are the percentile, normal, and bias-corrected bootstrap confidence intervals. The computational details of these confidence intervals for β_i, the ith model parameter, follow. For simplicity of notation, denote β_i as β. Also, let B represent the number of replicates for which the least squares fit converges

The option that computes the percentile bootstrap confidence interval, BOOTCI(PERC), does so by computing the $100(\alpha/2)$th and $100(1-\alpha/2)$th percentiles from the bootstrap parameter estimates. α is from the ALPHA= option in the PROC NLIN statement. These percentiles are computed as follows. Let $\tilde{\beta}_1, \tilde{\beta}_2, \ldots, \tilde{\beta}_B$ represent the ordered values of the bootstrap estimates for β. Let the kth weighted average percentile be y, set $p = \frac{k}{100}$, and let

$$np = j + g$$

where j is the integer part of np and g is the fractional part of np. Then the kth percentile, y, is given by

$$y = \begin{cases} \frac{1}{2}\left(\tilde{\beta}_j + \tilde{\beta}_{j+1}\right) & \text{if} \quad g = 0 \\ \tilde{\beta}_{j+1} & \text{if} \quad g > 0 \end{cases}$$

which corresponds to the default percentile definition of the UNIVARIATE procedure.

In contrast, the BOOTCI(NORMAL) option in the BOOTSTRAP statement computes the normal bootstrap confidence interval by approximating the distribution of the bootstrap parameter estimates as a normal distribution. Consequently, the normal bootstrap confidence interval, for α level, is given by

$$\hat{\beta} \pm \mathrm{stdb}_{\tilde{\beta}} \times \mathrm{z}^{(1-\alpha/2)}$$

where $\hat{\beta}$ is the original least squares parameter estimate, $\mathrm{stdb}_{\tilde{\beta}}$ is the standard deviation of the bootstrap parameter estimates, $\tilde{\beta}$; and $\mathrm{z}^{(1-\alpha/2)}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution.

The BOOTSTRAP statement option that computes the bias-corrected bootstrap confidence interval, BOOTCI(BC), does so by making use of the cumulative distribution function (CDF), $G(\tilde{\beta})$, of the bootstrap parameter estimates to correct for the upper and lower endpoints of the α level. The bias-corrected bootstrap confidence interval is given by

$$G^{-1}\big(\Phi(2z_0 \pm z_\alpha)\big)$$

where Φ is the standard normal CDF, $z_\alpha = \Phi^{-1}(\alpha)$, and z_0 is a bias correction given by

$$z_0 = \Phi^{-1}\left(\frac{\mathrm{N}\big(\tilde{\beta} \le \hat{\beta}\big)}{\mathrm{B}}\right)$$

where $\hat{\beta}$ is the estimate of β from the original least squares fit and $\mathrm{N}(\tilde{\beta} \le \hat{\beta})$ is the number of bootstrap estimates, $\tilde{\beta}$, that are less than or equal to $\hat{\beta}$.

In addition, PROC NLIN produces bootstrap estimates of the covariance and correlation matrices. For the *i*th and *j*th model parameters, the covariance is estimated by

$$\mathrm{Cov}(\tilde{\beta}_i, \tilde{\beta}_j) = \sum_{k=1}^{k=\mathrm{B}} \frac{(\tilde{\beta}_{ik} - \bar{\tilde{\beta}}_i)(\tilde{\beta}_{jk} - \bar{\tilde{\beta}}_j)}{\mathrm{B}-1}$$

where the sum runs over the nonmissing bootstrap parameter estimates. The bootstrap correlation matrix is estimated by scaling the bootstrap covariance matrix

$$\mathrm{Corr}(\tilde{\beta}_i, \tilde{\beta}_j) = \frac{\mathrm{Cov}(\tilde{\beta}_i, \tilde{\beta}_j)}{\mathrm{stdb}_{\tilde{\beta}_i}\mathrm{stdb}_{\tilde{\beta}_j}}$$

where $\mathrm{stdb}_{\tilde{\beta}_i}$ and $\mathrm{stdb}_{\tilde{\beta}_j}$ are the bootstrap standard deviations for the *i*th and *j*th parameters.

Missing Values

If the value of any one of the SAS variables involved in the model is missing from an observation, PROC NLIN deems that observation unusable and omits it from the analysis. If only the value of the dependent variable is missing, that observation has a predicted value calculated for it when you use an OUTPUT statement and specify the PREDICTED= option.

If an observation includes a missing value for one of the independent variables, both the predicted value and the residual value are missing for that observation. If the iterations fail to converge, the values for all variables named in the OUTPUT statement are set to missing.

Special Variables

Several special variables are created automatically and can be used in PROC NLIN programming statements.

Special Variables with Values That Are Set by PROC NLIN

The values of the following six special variables are set by PROC NLIN and should not be reset to a different value by programming statements:

ERROR is set to 1 if a numerical error or invalid argument to a function occurs during the current execution of the program. It is reset to 0 before each new execution.

ITER represents the current iteration number. The variable _ITER_ is set to –1 during the grid search phase.

MODEL is set to 1 for passes through the data when only the predicted values are needed, and not the derivatives. It is 0 when both predicted values and derivatives are needed. If your derivative calculations consume a lot of time, you can save resources by using the following statement after your MODEL statement but before your derivative calculations:

```
if _model_ then return;
```

The derivatives generated by PROC NLIN do this automatically.

_N_ indicates the number of times the PROC NLIN step has been executed. It is never reset for successive passes through the data set.

OBS indicates the observation number in the data set for the current program execution. It is reset to 1 to start each pass through the data set (unlike the _N_ variable).

SSE contains the error sum of squares of the last iteration. During the grid search phase, the _SSE_ variable is set to 0. For iteration 0, the _SSE_ variable is set to the SSE associated with the point chosen from the grid search.

PROC NLIN always sets the preceding special variables. In contrast, it sets the following special variables only when the BOOTSTRAP or PROFILE statements are invoked:

__COMPUTED_DEP__ is used to define _P_WEIGHTED_SSE_.

_P_WEIGHTED_SSE_ contains the error sum of squares for the optimization tasks that are required for profiling parameters, assessing the influence of each observation on the estimate of each profiled parameter, and bootstrap estimation.

Special Variable Used to Determine Convergence Criteria

The special variable _LOSS_ can be used to determine the criterion function for convergence and step shortening. PROC NLIN looks for the variable _LOSS_ in the programming statements and, if it is defined, uses the (weighted) sum of this value instead of the residual sum of squares to determine the criterion function for convergence and step shortening. This feature is useful in certain types of maximum-likelihood estimation.

NOTE: Even if you specify the _LOSS_ variable in your programming statements, the NLIN procedure continues to solve a least squares problem. The specified _LOSS_ variable does **not** define or alter the objective function for parameter estimation.

Weighted Regression with the Special Variable _WEIGHT_

To obtain weighted nonlinear least squares estimates of parameters, make an assignment to the _WEIGHT_ variable as in the following statement:

```
_weight_ = expression;
```

When this statement is included, the expression on the right side of the assignment statement is evaluated for each observation in the data set. The values multiplied by $1/\sigma^2$ are then taken as inverse elements of the diagonal variance-covariance matrix of the dependent variable.

When a variable name is given after the equal sign, the values of the variable are taken as the inverse elements of the variance-covariance matrix. The larger the _WEIGHT_ value, the more importance the observation is given.

The _WEIGHT_ variable can be a function of the estimated parameters. For estimation purposes, the derivative of the _WEIGHT_ variable with respect to the parameters is not included in the gradient and the Hessian of the loss function. This is normally the desired approach for iteratively reweighted least squares estimation. When the _WEIGHT_ variable is a function of the parameters, the gradient and the Hessian used can lead to poor convergence or nonconvergence of the requested estimation. To have the derivative of the _WEIGHT_ variable with respect to the parameters included in the gradient and the Hessian of the loss function, do not use the _WEIGHT_ variable. Instead, redefine the model as

$$(y - f(x, \beta)) \times \sqrt{\mathrm{wgt}(\beta)}$$

where y is the original dependent variable, $f(x, \beta)$ is the nonlinear model, and $\mathrm{wgt}(\beta)$ is the weight that is a function of the parameters.

If the _WEIGHT_= statement is not used, the default value of 1 is used, and regular least squares estimates are obtained.

Troubleshooting

This section describes a number of problems that might occur in your analysis with PROC NLIN.

Excessive Computing Time

If you specify a grid of starting values that contains many points, the analysis might take excessive time since the procedure must go through the entire data set for each point on the grid.

The analysis might also take excessive time if your problem takes many iterations to converge, since each iteration requires as much time as a linear regression with predicted values and residuals calculated.

Dependencies

The matrix of partial derivatives can be singular, possibly indicating an overparameterized model. For example, if b0 starts at zero in the following model, the derivatives for b1 are all zero for the first iteration:

```
parms b0=0 b1=.022;
model pop=b0*exp(b1*(year-1790));
der.b0=exp(b1*(year-1790));
der.b1=(year-1790)*b0*exp(b1*(year-1790));
```

The first iteration changes a subset of the parameters; then the procedure can make progress in succeeding iterations. This singularity problem is local. The next example displays a global problem. The term b2 in the exponent is not identifiable since it trades roles with b0.

```
parms b0=3.9 b1=.022 b2=0;
model pop=b0*exp(b1*(year-1790)+b2);
der.b0 = exp(b1*(year-1790)+b2);
der.b1 = (year-1790)*b0*exp(b1*(year-1790)+b2);
der.b2 = b0*exp(b1*(year-1790)+b2);
```

Unable to Improve

The method can lead to steps that do not improve the estimates even after a series of step halvings. If this happens, the procedure issues a message stating that it is unable to make further progress, and it then displays the following warning message:

```
PROC NLIN failed to converge
```

Then it displays the results. This means that the procedure has not converged. If you provided your own derivatives, check them carefully and then examine the residual sum of squares surface. If PROC NLIN has not converged, try a different set of starting values, a different METHOD= specification, the G4 option, or a different model.

Divergence

The iterative process might diverge, resulting in overflows in computations. It is also possible that parameters enter a space where arguments to such functions as LOG and SQRT become invalid. For example, consider the following model:

```
parms b=0;
model y = x / b;
```

Suppose that y contains only zeros, and suppose that the values for variable x are not zero. There is no least squares estimate for b since the SSE declines as b approaches infinity or minus infinity. To avoid the problem, the same model could be parameterized as **y = a*x**.

If you have divergence problems, try reparameterizing the model, selecting different starting values, increasing the maximum allowed number of iterations (the MAXITER= option), specifying an alternative METHOD= option, or including a BOUNDS statement.

Local Minimum

The program might converge to a local rather than a global minimum. For example, consider the following model:

```
parms a=1 b=-1;
model y=(1-a*x)*(1-b*x);
```

Once a solution is found, an equivalent solution with the same SSE can be obtained by swapping the values of a and b.

Discontinuities

The computational methods assume that the model is a continuous and smooth function of the parameters. If this is not true, the method does not work. For example, the following models do not work:

```
model y=a+int(b*x);

model y=a+b*x+4*(z>c);
```

Responding to Trouble

PROC NLIN does not necessarily produce a good solution the first time. Much depends on specifying good initial values for the parameters. You can specify a grid of values in the PARMS statement to search for good starting values. While most practical models should give you no trouble, other models can require switching to a different iteration method or a different computational method for matrix inversion. Specifying the option METHOD=MARQUARDT sometimes works when the default method (Gauss-Newton) does not work.

Computational Methods

Nonlinear Least Squares

Recall the basic notation for the nonlinear regression model from the section "Notation for Nonlinear Regression Models" on page 7133. The parameter vector $\boldsymbol{\beta}$ belongs to $\boldsymbol{\Omega}$, a subset of R^p. Two points of this set are of particular interest: the true value $\widetilde{\boldsymbol{\beta}}$ and the least squares estimate $\widehat{\boldsymbol{\beta}}$. The general nonlinear model fit with the NLIN procedure is represented by the equation

$$\mathbf{Y} = \mathbf{f}(\widetilde{\beta_0}, \widetilde{\beta_1}, \ldots, \widetilde{\beta_p}; \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_k) + \boldsymbol{\epsilon} = \mathbf{f}(\widetilde{\boldsymbol{\beta}}; \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_k) + \boldsymbol{\epsilon}$$

where $\mathbf{z}_j$ denotes the $(n \times 1)$ vector of the jth regressor (independent) variable, $\widetilde{\boldsymbol{\beta}}$ is the true value of the parameter vector, and $\boldsymbol{\epsilon}$ is the $(n \times 1)$ vector of homoscedastic and uncorrelated model errors with zero mean.

To write the model for the ith observation, the ith elements of $\mathbf{z}_1, \ldots, \mathbf{z}_k$ are collected in the row vector $\mathbf{z}_i'$, and the model equation becomes

$$Y_i = f(\boldsymbol{\beta}; \mathbf{z}_i') + \epsilon_i$$

The shorthand $f_i(\boldsymbol{\beta})$ will also be used throughout to denote the mean of the ith observation.

For any given value $\boldsymbol{\beta}$ we can compute the residual sum of squares

$$\begin{aligned}\mathrm{SSE}(\boldsymbol{\beta}) &= \sum_{i=1}^{n} \left(y_i - f\left(\boldsymbol{\beta}; \mathbf{z}_i'\right)\right)^2 \\ &= \sum_{i=1}^{n} \left(y_i - f_i\left(\boldsymbol{\beta}\right)\right)^2 = \mathbf{r}(\boldsymbol{\beta})'\mathbf{r}(\boldsymbol{\beta})\end{aligned}$$

The aim of nonlinear least squares estimation is to find the value $\widehat{\boldsymbol{\beta}}$ that minimizes $\mathrm{SSE}(\boldsymbol{\beta})$. Because $\mathbf{f}$ is a nonlinear function of $\boldsymbol{\beta}$, a closed-form solution does not exist for this minimization problem. An iterative process is used instead. The iterative techniques that PROC NLIN uses are similar to a series of linear regressions involving the matrix $\mathbf{X}$ and the residual vector $\mathbf{r} = \mathbf{y} - \mathbf{f}(\boldsymbol{\beta})$, evaluated at the current values of $\boldsymbol{\beta}$.

It is more insightful, however, to describe the algorithms in terms of their approach to minimizing the residual sum of squares and in terms of their updating formulas. If $\widehat{\boldsymbol{\beta}}^{(u)}$ denotes the value of the parameter estimates at the uth iteration, and $\widehat{\boldsymbol{\beta}}^{(0)}$ are your starting values, then the NLIN procedure attempts to find values k and $\boldsymbol{\Delta}$ such that

$$\widehat{\boldsymbol{\beta}}^{(u+1)} = \widehat{\boldsymbol{\beta}}^{(u)} + k\boldsymbol{\Delta}$$

and

$$\mathrm{SSE}\left(\widehat{\boldsymbol{\beta}}^{(u+1)}\right) < \mathrm{SSE}\left(\widehat{\boldsymbol{\beta}}^{(u)}\right)$$

The various methods to fit a nonlinear regression model—which you can select with the METHOD= option in the PROC NLIN statement—differ in the calculation of the update vector $\boldsymbol{\Delta}$.

The gradient and Hessian of the residual sum of squares with respect to individual parameters and pairs of parameters are, respectively,

$$\mathbf{g}(\beta_j) = \frac{\partial \mathrm{SSE}(\boldsymbol{\beta})}{\partial \beta_j} = -2\sum_{i=1}^{n}(y_i - f_i(\boldsymbol{\beta}))\,\frac{\partial f_i(\boldsymbol{\beta})}{\partial \beta_j}$$

$$[\mathbf{H}(\boldsymbol{\beta})]_{jk} = \frac{\partial \mathrm{SSE}(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k} = 2\sum_{i=1}^{n}\frac{\partial f_i(\boldsymbol{\beta})}{\partial \beta_j}\frac{\partial f_i(\boldsymbol{\beta})}{\partial \beta_k} - (y_i - f_i(\boldsymbol{\beta}))\,\frac{\partial^2 f_i(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k}$$

Denote as $\mathbf{H}_i^*(\boldsymbol{\beta})$ the Hessian matrix of the mean function,

$$\left[\mathbf{H}_i^*(\boldsymbol{\beta})\right]_{jk} = \left[\frac{\partial^2 f_i(\boldsymbol{\beta})}{\partial \beta_j \partial \beta_k}\right]_{jk}$$

Collecting the derivatives across all parameters leads to the expressions

$$\mathbf{g}(\boldsymbol{\beta}) = \frac{\partial \mathrm{SSE}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\mathbf{X}'\mathbf{r}(\boldsymbol{\beta})$$

$$\mathbf{H}(\boldsymbol{\beta}) = \frac{\partial^2 \mathrm{SSE}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}'} = 2\left(\mathbf{X}'\mathbf{X} - \sum_{i=1}^{n} r_i(\boldsymbol{\beta})\mathbf{H}_i^*(\boldsymbol{\beta})\right)$$

The change in the vector of parameter estimates is computed as follows, depending on the estimation method:

$$\text{Gauss-Newton: } \boldsymbol{\Delta} = (-\mathrm{E}[\mathbf{H}(\boldsymbol{\beta})])^{-}\,\mathbf{g}(\boldsymbol{\beta}) = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{r}$$

$$\text{Marquardt: } \boldsymbol{\Delta} = \left(\mathbf{X}'\mathbf{X} + \lambda \mathrm{diag}(\mathbf{X}'\mathbf{X})\right)^{-}\mathbf{X}'\mathbf{r}$$

$$\text{Newton: } \boldsymbol{\Delta} = -\mathbf{H}(\boldsymbol{\beta})^{-}\mathbf{g}(\boldsymbol{\beta}) = \mathbf{H}(\boldsymbol{\beta})^{-}\mathbf{X}'\mathbf{r}$$

$$\text{Steepest descent: } \boldsymbol{\Delta} = -\frac{1}{2}\frac{\partial \mathrm{SSE}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \mathbf{X}'\mathbf{r}$$

The Gauss-Newton and Marquardt iterative methods regress the residuals onto the partial derivatives of the model with respect to the parameters until the estimates converge. You can view the Marquardt algorithm as a Gauss-Newton algorithm with a ridging penalty. The Newton iterative method regresses the residuals onto a function of the first and second derivatives of the model with respect to the parameters until the estimates converge. Analytical first- and second-order derivatives are automatically computed as needed.

The default method used to compute $(\mathbf{X}'\mathbf{X})^{-}$ is the sweep (Goodnight 1979). It produces a reflexive generalized inverse (a g_2-inverse, Pringle and Rayner 1971). In some cases it might be preferable to use a Moore-Penrose inverse (a g_4-inverse) instead. If you specify the G4 option in the PROC NLIN statement, a g_4-inverse is used to calculate $\boldsymbol{\Delta}$ on each iteration.

The four algorithms are now described in greater detail.

Algorithmic Details

Gauss-Newton and Newton Methods

From the preceding set of equations you can see that the Marquardt method is a ridged version of the Gauss-Newton method. If the ridge parameter λ equals zero, the Marquardt step is identical to the Gauss-Newton step. An important difference between the Newton methods and the Gauss-Newton-type algorithms lies in the use of second derivatives. To motivate this distinctive element between Gauss-Newton and the Newton method, focus first on the objective function in nonlinear least squares. To numerically find the minimum of

$$\mathrm{SSE}(\boldsymbol{\beta}) = \mathbf{r}(\boldsymbol{\beta})'\mathbf{r}(\boldsymbol{\beta})$$

you can approach the problem by approximating the sum of squares criterion by a criterion for which you can compute a closed-form solution. Following Seber and Wild (1989, Sect. 2.1.3), we can achieve that by doing the following:

- approximating the model and substituting the approximation into $\mathrm{SSE}(\boldsymbol{\beta})$
- approximating $\mathrm{SSE}(\boldsymbol{\beta})$ directly

The first method, approximating the nonlinear model with a first-order Taylor series, is the purview of the Gauss-Newton method. Approximating the residual sum of squares directly is the realm of the Newton method.

The first-order Taylor series of the residual $\mathbf{r}(\boldsymbol{\beta})$ at the point $\widehat{\boldsymbol{\beta}}$ is

$$\mathbf{r}(\boldsymbol{\beta}) \approx \mathbf{r}(\widehat{\boldsymbol{\beta}}) - \widehat{\mathbf{X}}\left(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\right)$$

Substitution into $\mathrm{SSE}(\boldsymbol{\beta})$ leads to the objective function for the Gauss-Newton step:

$$\mathrm{SSE}\,(\boldsymbol{\beta}) \approx S_G(\boldsymbol{\beta}) = \mathbf{r}(\widehat{\boldsymbol{\beta}})' - 2\mathbf{r}'(\widehat{\boldsymbol{\beta}})\widehat{\mathbf{X}}\left(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\right) + \left(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\right)'\left(\widehat{\mathbf{X}}'\widehat{\mathbf{X}}\right)\left(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\right)$$

"Hat" notation is used here to indicate that the quantity in question is evaluated at $\widehat{\boldsymbol{\beta}}$.

To motivate the Newton method, take a second-order Taylor series of $\mathrm{SSE}(\boldsymbol{\beta})$ around the value $\widehat{\boldsymbol{\beta}}$:

$$\mathrm{SSE}\,(\boldsymbol{\beta}) \approx S_N(\boldsymbol{\beta}) = \mathrm{SSE}\left(\widehat{\boldsymbol{\beta}}\right) + \mathbf{g}(\widehat{\boldsymbol{\beta}})'\left(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\right) + \frac{1}{2}\left(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\right)'\mathbf{H}(\widehat{\boldsymbol{\beta}})\left(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\right)$$

Both $S_G(\boldsymbol{\beta})$ and $S_N(\boldsymbol{\beta})$ are quadratic functions in $\boldsymbol{\beta}$ and are easily minimized. The minima occur when

$$\text{Gauss-Newton: } \boldsymbol{\beta} - \widehat{\boldsymbol{\beta}} = \left(\widehat{\mathbf{X}}'\widehat{\mathbf{X}}\right)^{-}\widehat{\mathbf{X}}'\mathbf{r}(\widehat{\boldsymbol{\beta}})$$

$$\text{Newton}: \boldsymbol{\beta} - \widehat{\boldsymbol{\beta}} = -\mathbf{H}(\widehat{\boldsymbol{\beta}})^{-}\mathbf{g}(\widehat{\boldsymbol{\beta}})$$

and these terms define the preceding $\boldsymbol{\Delta}$ update vectors.

Gauss-Newton Method

Since the Gauss-Newton method is based on an approximation of the model, you can also derive the update vector by first considering the "normal" equations of the nonlinear model

$$\mathbf{X}'\mathbf{f}(\boldsymbol{\beta}) = \mathbf{X}'\mathbf{Y}$$

and then substituting the Taylor approximation

$$\mathbf{f}(\boldsymbol{\beta}) \approx \mathbf{f}(\widehat{\boldsymbol{\beta}}) + \mathbf{X}\left(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}\right)$$

for $\mathbf{f}(\boldsymbol{\beta})$. This leads to

$$\begin{aligned}
\mathbf{X}'\left(\mathbf{f}(\widehat{\boldsymbol{\beta}}) + \mathbf{X}(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}})\right) &= \mathbf{X}'\mathbf{Y} \\
(\mathbf{X}'\mathbf{X})(\boldsymbol{\beta} - \widehat{\boldsymbol{\beta}}) &= \mathbf{X}'\mathbf{Y} - \mathbf{X}'\mathbf{f}(\widehat{\boldsymbol{\beta}}) \\
(\mathbf{X}'\mathbf{X})\boldsymbol{\Delta} &= \mathbf{X}'\mathbf{r}(\widehat{\boldsymbol{\beta}})
\end{aligned}$$

and the update vector becomes

$$\boldsymbol{\Delta} = \left(\mathbf{X}'\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{r}(\widehat{\boldsymbol{\beta}})$$

NOTE: If $\mathbf{X}'\mathbf{X}$ is singular or becomes singular, PROC NLIN computes $\boldsymbol{\Delta}$ by using a generalized inverse for the iterations after singularity occurs. If $\mathbf{X}'\mathbf{X}$ is still singular for the last iteration, the solution should be examined.

Newton Method

The Newton method uses the second derivatives and solves the equation

$$\boldsymbol{\Delta} = \mathbf{H}^{-}\mathbf{X}'\mathbf{r}$$

If the automatic variables _WEIGHT_, _WGTJPJ_, and _RESID_ are used, then

$$\boldsymbol{\Delta} = \mathbf{H}^{-}\mathbf{X}'\mathbf{W}^{\mathrm{SSE}}\mathbf{r}^{*}$$

is the direction, where

$$\mathbf{H} = \mathbf{X}'\mathbf{W}^{\mathrm{XPX}}\mathbf{X} - \sum_{i=1}^{n}\mathbf{H}_i^{*}(\boldsymbol{\beta})w_i^{\mathrm{XPX}}r_i^{*}$$

and

$\mathbf{W}^{\mathrm{SSE}}$ is an $n \times n$ diagonal matrix with elements w_i^{SSE} of weights from the _WEIGHT_ variable. Each element w_i^{SSE} contains the value of _WEIGHT_ for the ith observation.

$\mathbf{W}^{\mathrm{XPX}}$ is an $n \times n$ diagonal matrix with elements w_i^{XPX} of weights from the _WGTJPJ_ variable. Each element w_i^{XPX} contains the value of _WGTJPJ_ for the ith observation.

$\mathbf{r}^{*}$ is a vector with elements r_i^{*} from the _RESID_ variable. Each element r_i^{*} contains the value of _RESID_ evaluated for the ith observation.

Marquardt Method

The updating formula for the Marquardt method is as follows:

$$\mathbf{\Delta} = (\mathbf{X}'\mathbf{X} + \lambda \mathrm{diag}(\mathbf{X}'\mathbf{X}))^{-}\mathbf{X}'\mathbf{e}$$

The Marquardt method is a compromise between the Gauss-Newton and steepest descent methods (Marquardt 1963). As $\lambda \to 0$, the direction approaches Gauss-Newton. As $\lambda \to \infty$, the direction approaches steepest descent.

Marquardt's studies indicate that the average angle between Gauss-Newton and steepest descent directions is about 90°. A choice of λ between 0 and infinity produces a compromise direction.

By default, PROC NLIN chooses $\lambda = 10^{-7}$ to start and computes $\mathbf{\Delta}$. If $\mathrm{SSE}(\boldsymbol{\beta}_0 + \mathbf{\Delta}) < \mathrm{SSE}(\boldsymbol{\beta}_0)$, then $\lambda = \lambda/10$ for the next iteration. Each time $\mathrm{SSE}(\boldsymbol{\beta}_0 + \mathbf{\Delta}) > \mathrm{SSE}(\boldsymbol{\beta}_0)$, then $\lambda = 10\lambda$.

NOTE: If the SSE decreases on each iteration, then $\lambda \to 0$, and you are essentially using the Gauss-Newton method. If SSE does not improve, then λ is increased until you are moving in the steepest descent direction.

Marquardt's method is equivalent to performing a series of ridge regressions, and it is useful when the parameter estimates are highly correlated or the objective function is not well approximated by a quadratic.

Steepest Descent (Gradient) Method

The steepest descent method is based directly on the gradient of $0.5\mathbf{r}(\boldsymbol{\beta})'\mathbf{r}(\boldsymbol{\beta})$:

$$\frac{1}{2}\frac{\partial \mathrm{SSE}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -\mathbf{X}'\mathbf{r}$$

The quantity $-\mathbf{X}'\mathbf{r}$ is the gradient along which $\boldsymbol{\epsilon}'\boldsymbol{\epsilon}$ increases. Thus $\mathbf{\Delta} = \mathbf{X}'\mathbf{r}$ is the direction of steepest descent.

If the automatic variables _WEIGHT_ and _RESID_ are used, then

$$\mathbf{\Delta} = \mathbf{X}'\mathbf{W}^{\mathrm{SSE}}\mathbf{r}^*$$

is the direction, where

$\mathbf{W}^{\mathrm{SSE}}$ is an $n \times n$ diagonal matrix with elements w_i^{SSE} of weights from the _WEIGHT_ variable. Each element w_i^{SSE} contains the value of _WEIGHT_ for the ith observation.

$\mathbf{r}^*$ is a vector with elements r_i^* from _RESID_. Each element r_i^* contains the value of _RESID_ evaluated for the ith observation.

Using the method of steepest descent, let

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \alpha\mathbf{\Delta}$$

where the scalar α is chosen such that

$$\mathrm{SSE}(\boldsymbol{\beta}_i + \alpha\mathbf{\Delta}) < \mathrm{SSE}(\boldsymbol{\beta}_i)$$

NOTE: The steepest-descent method can converge very slowly and is therefore not generally recommended. It is sometimes useful when the initial values are poor.

Step-Size Search

The default method of finding the step size k is step halving by using SMETHOD=HALVE. If $\mathrm{SSE}(\boldsymbol{\beta}^{(u)} + \boldsymbol{\Delta}) > \mathrm{SSE}(\boldsymbol{\beta}^{(u)})$, compute $\mathrm{SSE}(\boldsymbol{\beta}^{(u)} + 0.5\boldsymbol{\Delta})$, $\mathrm{SSE}(\boldsymbol{\beta}^{(u)} + 0.25\boldsymbol{\Delta}), \ldots$, until a smaller SSE is found.

If you specify SMETHOD=GOLDEN, the step size k is determined by a golden section search. The parameter TAU determines the length of the initial interval to be searched, with the interval having length TAU (or $2 \times \mathrm{TAU}$), depending on $\mathrm{SSE}(\boldsymbol{\beta}^{(u)} + \boldsymbol{\Delta})$. The RHO parameter specifies how fine the search is to be. The SSE at each endpoint of the interval is evaluated, and a new subinterval is chosen. The size of the interval is reduced until its length is less than RHO. One pass through the data is required each time the interval is reduced. Hence, if RHO is very small relative to TAU, a large amount of time can be spent determining a step size. For more information about the golden section search, see Kennedy and Gentle (1980).

If you specify SMETHOD=CUBIC, the NLIN procedure performs a cubic interpolation to estimate the step size. If the estimated step size does not result in a decrease in SSE, step halving is used.

Output Data Sets

The data set produced by the OUTEST= option in the PROC NLIN statement contains the parameter estimates on each iteration, including the grid search.

The variable _ITER_ contains the iteration number. The variable _TYPE_ denotes whether the observation contains iteration parameter estimates (“ITER”), final parameter estimates (“FINAL”), or covariance estimates (“COVB”). The variable _NAME_ contains the parameter name for covariances, and the variable _SSE_ contains the objective function value for the parameter estimates. The variable _STATUS_ indicates whether the estimates have converged.

The data set produced by the OUTPUT statement contains statistics calculated for each observation. In addition, the data set contains the variables from the input data set and any ID variables that are specified in the ID statement.

The data set that is produced by the PROFDATA= option in the PROFILE statement contains statistics that are calculated during parameter profiling. The variable _TYPE_ denotes whether the observation is from a profiling task (set as PROF) or jackknife resampling task (set as JACK). The variable _PROF_PARM_ refers to the name of the profiled parameter for profiling tasks. The variable _OBS_INF_MEAS_ is the observation index of the deleted observation for jackknife resampling tasks, the variable _SSE_ stands for the sum of squared errors, and the variables _L_STAT_ and _W_STAT_ contain the likelihood ratio pivotal statistic and the Wald pivotal statistic, respectively. The rest of the variables are the parameter estimates. Missing values for _SSE_ indicate that the least squares fit for that particular profiling or jackknife task does not converge.

The data set that is produced by the BOOTDATA= option in the BOOTSTRAP statement contains statistics that are calculated during bootstrap resampling and estimation. The variable _REPLICATE_ indexes the bootstrap sample data sets. The variable _SEED_ refers to the initial seed value that is used in obtaining each sample data set, the variable _SSE_ stands for the sum of squared errors, and the rest of the variables are the parameter estimates for the model that is fit to the bootstrap sample data set. Missing values for _SSE_ indicate that the least squares fit for that particular bootstrap sample data set does not converge.

Confidence Intervals

Parameter Confidence Intervals

By default, the parameter confidence intervals are computed using the Wald-based formula:

$$\widehat{\beta}_i \pm \text{stderr}_i \times t(n-p, 1-\alpha/2)$$

where $\widehat{\beta}_i$ is the ith parameter estimate, stderr_i is its estimated standard error, $t(n-p, 1-\alpha/2)$ is a t statistic with $n-p$ degrees of freedom, n is the number of usable observations, and p is the number of parameters. You can specify the significance level α used in constructing these confidence limits with the ALPHA= option in the PROC NLIN statement; the default value is $\alpha = 0.05$.

Because Wald-based confidence intervals are only asymptotically valid, you can use the BOOTSTRAP statement to request bootstrap-based confidence intervals. Furthermore, you can augment these confidence intervals with confidence curves for a more complete inference on confidence intervals. For more information, see the section "Bootstrap Resampling and Estimation" on page 7173 and "Profiling Parameters and Assessing the Influence of Observations on Parameter Estimates" on page 7172.

Model Confidence Intervals

Model confidence intervals are output when an OUT= data set is specified and one or more of the *keywords* LCLM, UCLM, LCL, UCL, L95M=, U95M=, L95=, and U95= is specified. The expressions for these terms are as follows:

$$\begin{aligned}
\text{LCLM} &= f(\boldsymbol{\beta}, \mathbf{z}_i) - \sqrt{\text{MSE} \times h_i/w_i} \times t(n-p, 1-\alpha/2) \\
\text{UCLM} &= f(\boldsymbol{\beta}, \mathbf{z}_i) + \sqrt{\text{MSE} \times h_i/w_i} \times t(n-p, 1-\alpha/2) \\
\text{LCL} &= f(\boldsymbol{\beta}, \mathbf{z}_i) - \sqrt{\text{MSE}(h_i + 1/w_i)} \times t(n-p, 1-\alpha/2) \\
\text{UCL} &= f(\boldsymbol{\beta}, \mathbf{z}_i) + \sqrt{\text{MSE}(h_i + 1/w_i)} \times t(n-p, 1-\alpha/2) \\
\text{L95M} &= f(\boldsymbol{\beta}, \mathbf{z}_i) - \sqrt{\text{MSE} \times h_i/w_i} \times t(n-p, 1-0.05/2) \\
\text{U95M} &= f(\boldsymbol{\beta}, \mathbf{z}_i) + \sqrt{\text{MSE} \times h_i/w_i} \times t(n-p, 1-0.05/2) \\
\text{L95} &= f(\boldsymbol{\beta}, \mathbf{z}_i) - \sqrt{\text{MSE}(h_i + 1/w_i)} \times t(n-p, 1-0.05/2) \\
\text{U95} &= f(\boldsymbol{\beta}, \mathbf{z}_i) + \sqrt{\text{MSE}(h_i + 1/w_i)} \times t(n-p, 1-0.05/2)
\end{aligned}$$

where $h_i = w_i \mathbf{x}_i (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1} \mathbf{x}_i'$ is the leverage, $\mathbf{X} = \partial \mathbf{f}/\partial \boldsymbol{\beta}$, and $\mathbf{x}_i$ is the ith row of $\mathbf{X}$. These results are derived for linear systems. The intervals are approximate for nonlinear models. The value α in the preceding formulas for LCLM, UCLM, LCL, and UCL can be set with the ALPHA= option in the PROC NLIN statement or with the ALPHA= option in the OUTPUT statement. If both ALPHA= options are specified, the option in the OUTPUT takes precedence.

Covariance Matrix of Parameter Estimates

By default, PROC NLIN estimates the covariance matrix of parameter estimates based on the following first order asymptotic approximation. For unconstrained estimates (no active bounds), the covariance matrix of the parameter estimates is

$$\mathrm{mse} \times (\mathbf{X}'\mathbf{X})^{-1}$$

for the gradient, Marquardt, and Gauss methods and

$$\mathrm{mse} \times \mathbf{H}^{-1}$$

for the Newton method. Recall that $\mathbf{X}$ is the matrix of the first partial derivatives of the nonlinear model with respect to the parameters. The matrices are evaluated at the final parameter estimates. The mean square error, the estimate of the residual variance σ^2, is computed as

$$\mathrm{mse} = \mathbf{r}'\mathbf{r}/(n-p)$$

where n is the number of nonmissing (used) observations and p is the number of estimable parameters. The standard error reported for the parameter estimates is the square root of the corresponding diagonal element of this matrix. If you specify a value for the residual variance with the SIGSQ= option, then that value replaces mse in the preceding expressions.

Now suppose that restrictions or bounds are active. Equality restrictions can be written as a vector function, $h(\boldsymbol{\theta}) = \mathbf{0}$. Inequality restrictions are either active or inactive. When an inequality restriction is active, it is treated as an equality restriction.

Assume that the vector $h(\boldsymbol{\theta})$ contains the current active restrictions. The constraint matrix $\mathbf{A}$ is then

$$\mathbf{A}(\widehat{\boldsymbol{\theta}}) = \frac{\partial h(\widehat{\boldsymbol{\theta}})}{\partial \widehat{\boldsymbol{\theta}}}$$

The covariance matrix for the restricted parameter estimates is computed as

$$\mathbf{Z}(\mathbf{Z}'\mathbf{H}\mathbf{Z})^{-1}\mathbf{Z}'$$

where $\mathbf{H}$ is the Hessian (or approximation to the Hessian) and $\mathbf{Z}$ collects the last $(p - n_c)$ columns of $\mathbf{Q}$ from an LQ factorization of the constraint matrix. Further, n_c is the number of active constraints, and p denotes the number of parameters. See Gill, Murray, and Wright (1981) for more details about the LQ factorization. The covariance matrix for the Lagrange multipliers is computed as

$$\left(\mathbf{A}\mathbf{H}^{-1}\mathbf{A}'\right)^{-1}$$

In addition to the default covariance matrix estimate, the BOOTSTRAP statement can be used to produce bootstrap covariance matrix estimate. See the section "Bootstrap Resampling and Estimation" on page 7173 for details.

Convergence Measures

The NLIN procedure computes and reports four convergence measures, labeled R, PPC, RPC, and OBJECT.

R is the primary convergence measure for the parameters. It measures the degree to which the residuals are orthogonal to the columns of **X**, and it approaches 0 as the gradient of the objective function becomes small. R is defined as

$$\sqrt{\frac{\mathbf{r}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{r}}{\mathrm{LOSS}^i}}$$

PPC is the prospective parameter change measure. PPC measures the maximum relative change in the parameters implied by the parameter-change vector computed for the next iteration. At the kth iteration, PPC is the maximum over the parameters

$$\frac{|\widehat{\beta}_i^{(k+1)} - \widehat{\beta}_i^{(k)}|}{|\widehat{\beta}_i^{(k)}| + 1\mathrm{E} - 6}$$

where $\widehat{\beta}_i^{(k)}$ is the current value of the ith parameter and $\widehat{\beta}_i^{(k+1)}$ is the prospective value of this parameter after adding the change vector computed for the next iteration. These changes are measured before step length adjustments are made. The parameter with the maximum prospective relative change is displayed with the value of PPC, unless the PPC is nearly 0.

RPC is the retrospective parameter change measure. RPC measures the maximum relative change in the parameters from the previous iteration. At the kth iteration, RPC is the maximum over i of

$$\frac{|\widehat{\beta}_i^{(k)} - \widehat{\beta}_i^{(k-1)}|}{|\widehat{\beta}_i^{(k-1)}| + 1\mathrm{E} - 6}$$

where $\widehat{\beta}_i^{(k)}$ is the current value of the ith parameter and $\widehat{\beta}_i^{k-1}$ is the previous value of this parameter. These changes are measured before step length adjustments are made. The name of the parameter with the maximum retrospective relative change is displayed with the value of RPC, unless the RPC is nearly 0.

OBJECT measures the relative change in the objective function value between iterations:

$$\frac{|O^{(k)} - O^{(k-1)}|}{|O^{(k-1)} + 1\mathrm{E} - 6|}$$

where $O^{(k-1)}$ is the value of the objective function ($O^{(k)}$) from the previous iteration. This is the old CONVERGEOBJ= criterion.

Displayed Output

In addition to the output data sets, PROC NLIN also produces the following output objects:

- the residual sums of squares associated with all or some of the combinations of possible starting values of the parameters
- the estimates of the parameters and the residual sums of squares at each iteration
- the estimation summary table, which displays information about the estimation method, the number of observations in the analysis, the objective function, and convergence measures
- the analysis of variance table, including sums of squares for the "Model," "Residual," and "Total" sources of variation ("Corrected Total" or "Uncorrected Total"), and the model F test. Note that, only the uncorrected total SS is reported and the respective F test is based on the uncorrected total SS if PROC NLIN determines the model does not include an intercept. If PROC NLIN determines the model does include an intercept, only the corrected total SS is reported and the respective F test is based on the corrected total SS.
- the table of parameter estimates, which contains for each parameter in the model its estimate, the approximate standard error of the estimate, and a 95% confidence interval based on the approximate standard error. The confidence level can be changed with the ALPHA= option in the PROC NLIN statement. The HOUGAARD option in the PROC NLIN statement requests that Hougaard's skewness measure be added for each parameter. The standard errors and confidence limits are labeled approximate because they are valid asymptotically as the number of observations grows. If your model is linear in the parameters, the standard errors and confidence intervals are not approximate.
- the approximate correlation matrix of the parameter estimates. This correlation matrix is labeled approximate because it is computed from the approximate covariance matrix of the parameter estimates. If your model is linear in the parameters, the correlation matrix is not approximate.
- the table of additional estimates, which displays the results of all ESTIMATE statements that you specify. In addition, if you specify the ECOV and ECORR options in the PROC NLIN statement, the procedure also produces tables that display the approximate covariance and correlation matrices, respectively, of the additional estimates. They are computed using the delta method (Billingsley 1986; Cox 1998). Similarly, the EDER option in the PROC NLIN statement produces a table that displays the derivatives of the additional estimates with respect to the model parameters that are evaluated at their final estimated values.
- the table of contrasts, which displays the results of all CONTRAST statements that you specify.

ODS Table Names

PROC NLIN assigns a name to each table it creates. You can use these names to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 88.9. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 88.9 ODS Tables Produced by PROC NLIN

ODS Table Name	Description	Statement	Option
AdditionalEstimates	Results from ESTIMATE statements	ESTIMATE	
ANOVA	Analysis of variance	Default	
BootstrapCorr	Bootstrap estimate of correlation matrix of the parameters	BOOTSTRAP	
BootstrapCov	Bootstrap estimate of covariance matrix of the parameters	BOOTSTRAP	
CodeDependency	Variable cross reference	PROC NLIN	LISTDEP
CodeList	Listing of program statements	PROC NLIN	LISTCODE
Contrasts	Results from CONTRAST statements	CONTRAST	
ConvergenceStatus	Convergence status	Default	
CorrB	Correlation of the parameters	Default	
CorrMatAddEst	Correlation matrix of additional estimates	ESTIMATE	ECORR
CovMatAddEst	Covariance matrix of additional estimates	ESTIMATE	ECOV
DerAddEst	Derivatives of additional estimates	ESTIMATE	EDER
EstSummary	Summary of the estimation	Default	
FirstDerivatives	First derivative table	PROC NLIN	LISTDER
IterHistory	Iteration output	Default	
MissingValues	Missing values generated by the program	Default	
NonlinearityMeasures	Global nonlinearity measures	PROC NLIN	NLINMEASURES
ParameterEstimates	Parameter estimates	Default	

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC NLIN assigns a name to each graph it creates using ODS. You can use these names to refer to the graphs when using ODS. The graphs that are controlled by the PLOTS option in the PROC NLIN statement are listed in Table 88.10, those that are controlled by the options in PROFILE statement are in Table 88.11.

Table 88.10 Graphs Controlled by the PLOTS option in the PROC NLIN Statement

ODS Graph Name	Plot Description	PLOTS Option
ContourFitPlot	Contour fit plot for models with two regressors	FIT
FitPlot	Fit plot for models with one regressor	FIT
FitDiagnosticsPanel	Panel of fit diagnostics	DIAGNOSTICS
LeveragePlot	Tangential and Jacobian leverages versus observation number	DIAGNOSTICS
LocalInfluencePlot	Local influence versus observation number	DIAGNOSTICS
ObservedByPredictedPlot	Dependent variable versus predicted values	DIAGNOSTICS(UNPACK)
ProjectedResidualHistogram	A histogram of the projected residuals	DIAGNOSTICS(UNPACK)
RawResidualExpectationPlot	Raw residual expectation versus predicted values	DIAGNOSTICS(UNPACK)
RawResidualHistogram	A histogram of the raw residuals	DIAGNOSTICS(UNPACK)
ResidualBoxPlot	A box plot of the raw and projected residuals	DIAGNOSTICS(UNPACK)
ResidualPanel	A panel of the raw and projected residuals versus the regressors	RESIDUALS
ResidualPlot	A plot of the raw and projected residuals versus the regressors	RESIDUALS(UNPACK)
ResidualByPredictedPlot	Raw and projected residuals versus the predicted values	DIAGNOSTICS(UNPACK)
RStudentByJacLeveragePlot	Standardized raw and projected residuals versus Jacobian leverage	DIAGNOSTICS(UNPACK)
RStudentByPredictedPlot	Standardized raw and projected residuals versus the predicted values	DIAGNOSTICS(UNPACK)
RStudentByTanLeveragePlot	Standardized raw and projected residuals versus tangential leverage	DIAGNOSTICS(UNPACK)

Table 88.11 Graphs Controlled by the PROFILE Statement

ODS Graph Name	Plot Description	PROFILE Option
ConfidenceCurve	Parameter value versus *t* value	CONFCURV
JackknifePlot	Absolute relative percentage difference versus observation number	JACKKNIFE
ProfiletPlot	Likelihood ratio pivotal statistic versus Wald pivotal statistic	TPLOT

Table 88.12 Graphs Controlled by the BOOTSTRAP Statement

ODS Graph Name	Plot Description	BOOTSTRAP Option
BootstrapHistPlot	A histogram of bootstrap parameter estimates	BOOTPLOTS(HIST)
BootstrapScatterPlot	Pairwise scatter plot of bootstrap parameter estimates	BOOTPLOTS(SCATTER)

Convergence Status Table

The "Convergence Status" table can be used to programmatically check the status of an estimation. This table contains the Status variable that takes on the value 0, 1, 2, or 3. If Status takes on a value less than 3, the convergence criterion was met. Specifically, the values mean the following:

Status=0 indicates that the convergence criterion was met and no warning or error messages were issued during the PROC NLIN run. Also, no notes that could indicate a problem with the model were issued.

Status=1 indicates that the convergence criterion was met and notes were written to the log that might indicate a problem with the model.

Status=2 indicates that the convergence criterion was met and one or more warning messages were produced during the PROC NLIN run.

Status=3 indicates that the convergence criterion was not met.

The following sample program demonstrates how the "Convergence Status" table can be used:

```
ods output ConvergenceStatus=ConvStatus;
proc nlin data=YourData;
   parameters a=1 b=1 c=1;
   model wgt = a + x / (b*y+c*z);
run;

data _null_;
   set ConvStatus;
   if status > 0 then put "A problem occurred";
run;
```

Examples: NLIN Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 88.1: Segmented Model

(View the complete code for this example (nlinex1.sas) in the example repository.)

Suppose you are interested in fitting a model that consists of two segments that connect in a smooth fashion. For example, the following model states that for values of x less than x_0 the mean of Y is a quadratic function in x, and for values of x greater than x_0 the mean of Y is constant:

$$\mathrm{E}[Y|x] = \begin{cases} \alpha + \beta x + \gamma x^2 & \text{if } x < x_0 \\ c & \text{if } x \geq x_0 \end{cases}$$

In this model equation α, β, and γ are the coefficients of the quadratic segment, and c is the plateau of the mean function. The NLIN procedure can fit such a segmented model even when the join point, x_0, is unknown.

We also want to impose conditions on the two segments of the model. First, the curve should be continuous—that is, the quadratic and the plateau section need to meet at x_0. Second, the curve should be smooth—that is, the first derivative of the two segments with respect to x need to coincide at x_0.

The continuity condition requires that

$$p = \mathrm{E}[Y|x_0] = \alpha + \beta x_0 + \gamma x_0^2$$

The smoothness condition requires that

$$\frac{\partial \mathrm{E}[Y|x_0]}{\partial x} = \beta + 2\gamma x_0 \equiv 0$$

If you solve for x_0 and substitute into the expression for c, the two conditions jointly imply that

$$\begin{aligned} x_0 &= -\beta/2\gamma \\ c &= \alpha - \beta^2/4\gamma \end{aligned}$$

Although there are apparently four unknowns, the model contains only three parameters. The continuity and smoothness restrictions together completely determine one parameter given the other three.

The following DATA step creates the SAS data set for this example:

```
data a;
   input y x @@;
   datalines;
.46 1  .47  2 .57  3 .61  4 .62  5 .68  6 .69  7
.78 8  .70  9 .74 10 .77 11 .78 12 .74 13 .80 13
.80 15 .78 16
;
```

The following PROC NLIN statements fit this segmented model:

```
title 'Quadratic Model with Plateau';
proc nlin data=a;
   parms alpha=.45 beta=.05 gamma=-.0025;

   x0 = -.5*beta / gamma;

   if (x < x0) then
        mean = alpha + beta*x  + gamma*x*x;
   else mean = alpha + beta*x0 + gamma*x0*x0;
   model y = mean;

   if _obs_=1 and _iter_ =.  then do;
      plateau =alpha + beta*x0 + gamma*x0*x0;
      put /  x0= plateau=  ;
   end;
   output out=b predicted=yp;
run;
```

The parameters of the model are α, β, and γ, respectively. They are represented in the PROC NLIN statements by the variables alpha, beta, and gamma, respectively. In order to model the two segments, a conditional statement is used that assigns the appropriate expression to the mean function depending on the value of x_0. A PUT statement is used to print the constrained parameters every time the program is executed for the first observation. The OUTPUT statement computes predicted values for plotting and saves them to data set b.

Note that there are other ways in which you can write the conditional expressions for this model. For example, you could formulate a condition with two model statements, as follows:

```
proc nlin data=a;
   parms alpha=.45 beta=.05 gamma=-.0025;
   x0 = -.5*beta / gamma;
   if (x < x0) then
        model y = alpha+beta*x+gamma*x*x;
   else model y = alpha+beta*x0+gamma*x0*x0;
run;
```

Or you could use a single expression with a conditional evaluation, as in the following statements:

```
proc nlin data=a;
   parms alpha=.45 beta=.05 gamma=-.0025;
   x0 = -.5*beta / gamma;
   model y = (x <x0)*(alpha+beta*x +gamma*x*x) +
             (x>=x0)*(alpha+beta*x0+gamma*x0*x0);
run;
```

The results from fitting this model with PROC NLIN are shown in Output 88.1.1–Output 88.1.3. The iterative optimization converges after six iterations (Output 88.1.1). Output 88.1.1 indicates that the join point is 12.747 and the plateau value is 0.777.

Output 88.1.1 Nonlinear Least-Squares Iterative Phase

Quadratic Model with Plateau

The NLIN Procedure
Dependent Variable y
Method: Gauss-Newton

Iterative Phase				
Iter	alpha	beta	gamma	Sum of Squares
0	0.4500	0.0500	-0.00250	0.0562
1	0.3881	0.0616	-0.00234	0.0118
2	0.3930	0.0601	-0.00234	0.0101
3	0.3922	0.0604	-0.00237	0.0101
4	0.3921	0.0605	-0.00237	0.0101
5	0.3921	0.0605	-0.00237	0.0101
6	0.3921	0.0605	-0.00237	0.0101

NOTE: Convergence criterion met.

Output 88.1.2 Results from Put Statement

```
x0=12.747669162 plateau=0.7774974276
```

Output 88.1.3 Least-Squares Analysis for the Quadratic Model

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	0.1769	0.0884	114.22	<.0001
Error	13	0.0101	0.000774		
Corrected Total	15	0.1869			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
alpha	0.3921	0.0267	0.3345	0.4497
beta	0.0605	0.00842	0.0423	0.0787
gamma	-0.00237	0.000551	-0.00356	-0.00118

The following statements produce a graph of the observed and predicted values with reference lines for the join point and plateau estimates (Output 88.1.4):

```
proc sgplot data=b noautolegend;
   yaxis label='Observed or Predicted';
   refline 0.777  / axis=y label="Plateau"    labelpos=min;
   refline 12.747 / axis=x label="Join point" labelpos=min;
   scatter y=y  x=x;
   series  y=yp x=x;
run;
```

Output 88.1.4 Observed and Predicted Values for the Quadratic Model

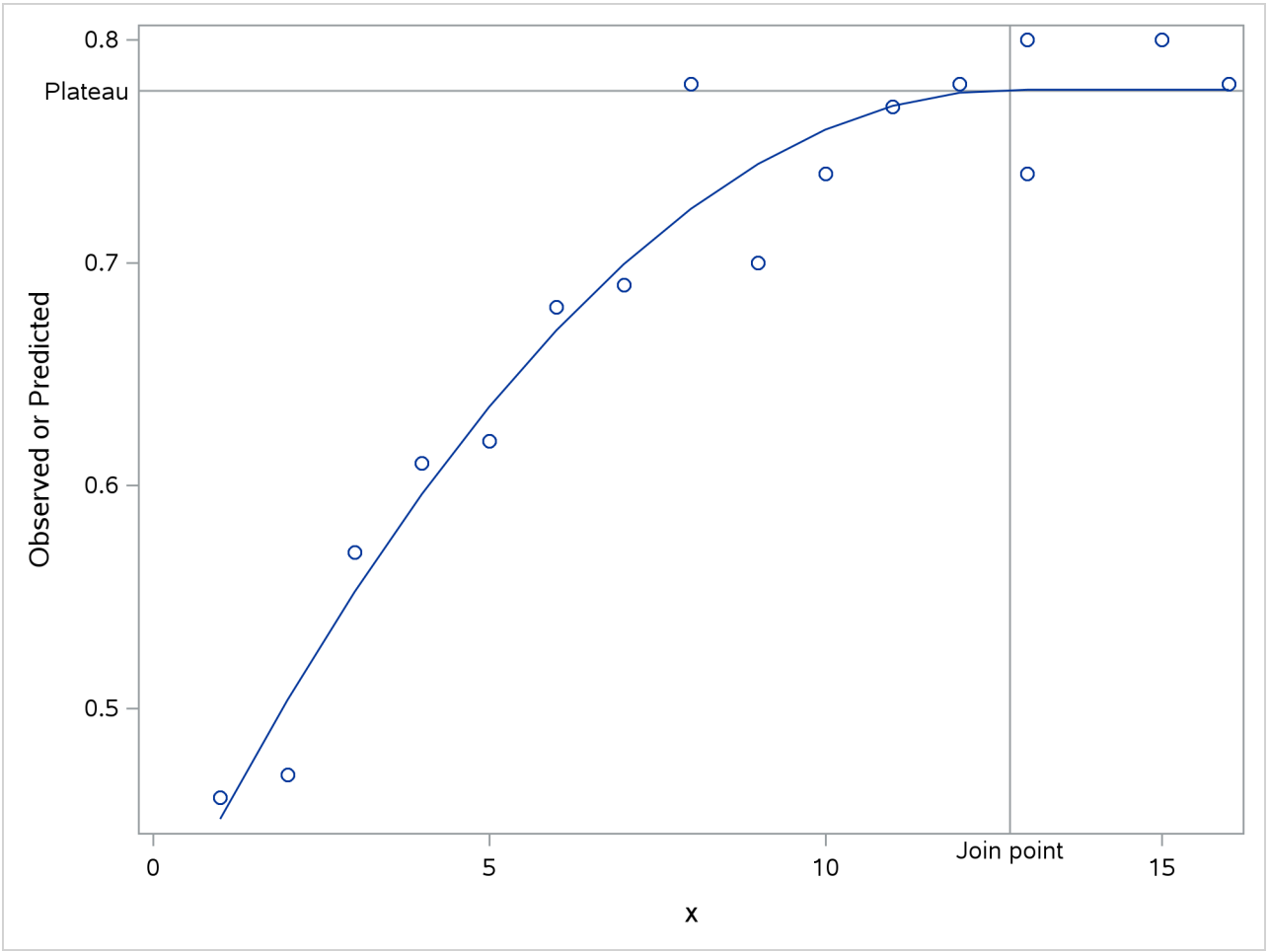

If you want to estimate the join point directly, you can use the relationship between the parameters to change the parameterization of the model in such a way that the mean function depends directly on x_0. Using the smoothness condition that relates x_0 to γ,

$$x_0 = -\beta/2\gamma$$

you can express γ as a function of β and x_0:

$$\gamma = -\beta/(2x_0)$$

Substituting for γ in the model equation

$$\mathrm{E}[Y|x] = \begin{cases} \alpha + \beta x + \gamma x^2 & \text{if } x < x_0 \\ \alpha - \beta^2/(4\gamma) & \text{if } x \geq x_0 \end{cases}$$

yields the reparameterized model

$$\mathrm{E}[Y|x] = \begin{cases} \alpha + \beta x(1 - x/(2x_0)) & \text{if } x < x_0 \\ \alpha + \beta x_0/2 & \text{if } x \geq x_0 \end{cases}$$

This model is fit with the following PROC NLIN statements:

```
proc nlin data=a;
   parms alpha=.45 beta=.05 x0=10;
   if (x<x0) then
        mean = alpha + beta*x *(1-x/(2*x0));
   else mean = alpha + beta*x0/2;
   model y = mean;
run;
```

Output 88.1.5 Results from Reparameterized Model

The NLIN Procedure
Dependent Variable y
Method: Gauss-Newton

NOTE: Convergence criterion met.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	0.1769	0.0884	114.22	<.0001
Error	13	0.0101	0.000774		
Corrected Total	15	0.1869			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
alpha	0.3921	0.0267	0.3345	0.4497
beta	0.0605	0.00842	0.0423	0.0787
x0	12.7477	1.2781	9.9864	15.5089

The analysis of variance table in the reparameterized model is the same as in the earlier analysis (compare Output 88.1.5 and Output 88.1.3). Changing the parameterization of a model does not affect the fit. The "Parameter Estimates" table now shows x0 as a parameter in the model. The estimate agrees with the earlier result that uses the PUT statement (Output 88.1.2). Since x_0 is now a model parameter, the NLIN procedure also reports its asymptotic standard error and its approximate 95% confidence interval.

Example 88.2: Iteratively Reweighted Least Squares

(View the complete code for this example (nlinex2.sas) in the example repository.)

With the NLIN procedure you can perform weighted nonlinear least squares regression in situations where the weights are functions of the parameters. To minimize a weighted sum of squares, you assign an expression to the _WEIGHT_ variable in your PROC NLIN statements. When the _WEIGHT_ variable depends on the model parameters, the estimation technique is known as iteratively reweighted least squares (IRLS). In this situation you should employ the NOHALVE option in the PROC NLIN statement. Because the weights change from iteration to iteration, it is not reasonable to expect the *weighted* residual sum of squares to decrease between iterations. The NOHALVE option removes that restriction.

Examples where IRLS estimation is used include robust regression via M-estimation (Huber 1964, 1973), generalized linear models (McCullagh and Nelder 1989), and semivariogram fitting in spatial statistics

(Schabenberger and Pierce 2002, Sect. 9.2). There are dedicated SAS/STAT procedures for robust regression (the ROBUSTREG procedure) and generalized linear models (the GENMOD and GLIMMIX procedures). Examples of weighted least squares fitting of a semivariogram function can be found in Chapter 131, "The VARIOGRAM Procedure."

In this example we show an application of PROC NLIN for M-estimation only to illustrate the connection between robust regression and weighted least squares. The ROBUSTREG procedure is the appropriate tool to fit these models with SAS/STAT software.

M-estimation was introduced by Huber (1964, 1973) to estimate location parameters robustly. Beaton and Tukey (1974) applied the idea of M-estimation in regression models and introduced the biweight (or bisquare) weight function. See Holland and Welsch (1977) for this and other robust methods. Consider a linear regression model of the form

$$\mathrm{E}[Y_i|x] = \mathbf{x}_i'\boldsymbol{\beta} + \epsilon_i$$

In weighted least squares estimation you seek the parameters $\widehat{\boldsymbol{\beta}}$ that minimize

$$\sum_{i=1}^{n} w_i \left(y_i - \mathbf{x}_i'\boldsymbol{\beta}\right)^2 = \sum_{i=1}^{n} w_i e_i^2$$

where w_i is the weight associated with the ith observation. The *normal* equations of this minimization problem can be written as

$$\sum_{i=1}^{n} w_i e_i \mathbf{x}_i = \mathbf{0}$$

In M-estimation the corresponding equations take on the form

$$\sum_{i=1}^{n} \psi(e_i)\mathbf{x}_i = \mathbf{0}$$

where $\psi(\cdot)$ is a weighing function. The Beaton-Tukey biweight, for example, can be written as

$$\psi(e_i) = \begin{cases} e_i\left(1 - \left(\frac{e_i}{\sigma k}\right)^2\right)^2 & |e_i/\sigma| \le k \\ 0 & |e_i/\sigma| > k \end{cases}$$

Substitution into the estimating equation for M-estimation yields weighted least squares equations

$$\sum_{i=1}^{n} \psi(e_i)\mathbf{x}_i = \sum_{i=1}^{n} w_i e_i \mathbf{x}_i = \mathbf{0}$$

$$w_i = \begin{cases} \left(1 - \left(\frac{e_i}{\sigma k}\right)^2\right)^2 & |e_i/\sigma| \le k \\ 0 & |e_i/\sigma| > k \end{cases}$$

The biweight function involves two constants, σ and k. The scale σ can be fixed or estimated from the fit in the previous iteration. If σ is estimated, a robust estimator of scale is typically used. In this example σ is fixed at 2. A common value for the constant k is $k = 4.685$.

The following DATA step creates a SAS data set of the population of the United States (in millions), recorded at 10-year intervals starting in 1790 and ending in 1990. The aim is to fit a quadratic linear model to the population over time.

```
title 'U.S. Population Growth';
data uspop;
   input pop :6.3 @@;
   retain year 1780;
   year   = year+10;
   yearsq = year*year;
   datalines;
3929 5308 7239 9638 12866 17069 23191 31443 39818 50155
62947 75994 91972 105710 122775 131669 151325 179323 203211
226542 248710
;
```

The PROC NLIN code that follows fits this linear model by M-estimation and IRLS. The weight function is set to a zero or nonzero value depending on the value of the scaled residual. The NOHALVE option removes the requirement that the (weighted) residual sum of squares must decrease between iterations.

```
title 'Beaton/Tukey Biweight Robust Regression using IRLS';
proc nlin data=uspop nohalve;
   parms b0=20450.43 b1=-22.7806 b2=.0063456;
   model pop=b0+b1*year+b2*year*year;
   resid = pop-model.pop;
   sigma = 2;
   k     = 4.685;
   if abs(resid/sigma)<=k then _weight_=(1-(resid / (sigma*k))**2)**2;
   else _weight_=0;
   output out=c r=rbi;
run;
```

Parameter estimates from this fit are shown in Output 88.2.1, and the computed weights at the final iteration are displayed in Output 88.2.2. The observations for 1940 and 1950 are highly discounted because of their large residuals.

Output 88.2.1 Nonlinear Least-Squares Analysis

Beaton/Tukey Biweight Robust Regression using IRLS

The NLIN Procedure

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	2	113564	56782.0	49454.5	<.0001
Error	18	20.6670	1.1482		
Corrected Total	20	113585			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
b0	20828.7	259.4	20283.8	21373.6
b1	-23.2004	0.2746	-23.7773	-22.6235
b2	0.00646	0.000073	0.00631	0.00661

Output 88.2.2 Listing of Computed Weights from PROC NLIN

Obs	pop	year	yearsq	rbi	sigma	k	_weight_
1	3.929	1790	3204100	-0.93711	2	4.685	0.98010
2	5.308	1800	3240000	0.46091	2	4.685	0.99517
3	7.239	1810	3276100	1.11853	2	4.685	0.97170
4	9.638	1820	3312400	0.95176	2	4.685	0.97947
5	12.866	1830	3348900	0.32159	2	4.685	0.99765
6	17.069	1840	3385600	-0.62597	2	4.685	0.99109
7	23.191	1850	3422500	-0.94692	2	4.685	0.97968
8	31.443	1860	3459600	-0.43027	2	4.685	0.99579
9	39.818	1870	3496900	-1.08302	2	4.685	0.97346
10	50.155	1880	3534400	-1.06615	2	4.685	0.97427
11	62.947	1890	3572100	0.11332	2	4.685	0.99971
12	75.994	1900	3610000	0.25539	2	4.685	0.99851
13	91.972	1910	3648100	2.03607	2	4.685	0.90779
14	105.710	1920	3686400	0.28436	2	4.685	0.99816
15	122.775	1930	3724900	0.56725	2	4.685	0.99268
16	131.669	1940	3763600	-8.61325	2	4.685	0.02403
17	151.325	1950	3802500	-8.32415	2	4.685	0.04443
18	179.323	1960	3841600	-0.98543	2	4.685	0.97800
19	203.211	1970	3880900	0.95088	2	4.685	0.97951
20	226.542	1980	3920400	1.03780	2	4.685	0.97562
21	248.710	1990	3960100	-1.33067	2	4.685	0.96007

You can obtain this analysis more conveniently with PROC ROBUSTREG. The procedure re-estimates the scale parameter robustly between iterations. To obtain an analysis with a fixed scale parameter as in this example, use the following PROC ROBUSTREG statements:

```
proc robustreg data=uspop method=m(scale=2);
   model pop = year year*year;
   output out=weights weight=w;
run;

proc print data=weights;
run;
```

Note that the computation of standard errors in the ROBUSTREG procedure is different from the calculations in the NLIN procedure.

Example 88.3: Probit Model with Likelihood Function

(View the complete code for this example (nlinex3.sas) in the example repository.)

The data in this example, taken from Lee (1974), consist of patient characteristics and a variable indicating whether cancer remission occurred. This example demonstrates how to use PROC NLIN with a likelihood function. In this case, twice the negative of the log-likelihood function is to be minimized. This is the objective function for the analysis:

$$-2\log L = -2\sum_{i=1}^{n}\log\{\pi_i(y_i, \mathbf{x}_i)\}$$

In this expression, π_i denotes the success probability of the n Bernoulli trials, and $\log L$ is the log likelihood of n independent binary (Bernoulli) observations. The probability π_i depends on the observations through the linear predictor η_i,

$$\pi_i(y_i, \mathbf{x}_i) = \begin{cases} 1-\Phi(\eta_i) & y_i = 1 \\ \Phi(\eta_i) & y_i = 0 \end{cases}$$

The linear predictor takes the form of a regression equation that is linear in the parameters,

$$\eta_i = \beta_0 + \beta_1 z_{1i} + \beta_2 z_{2i} + \cdots + \beta_k z_{ki} = \mathbf{z}_i\boldsymbol{\beta}$$

Despite this linearity of η in the z variables, the probit model is nonlinear, because the linear predictor appears inside the nonlinear probit function.

In order to use the NLIN procedure to minimize the function, the estimation problem must be cast in terms of a nonlinear least squares problem with objective function

$$\sum_{i=1}^{n}(y_i - f(\boldsymbol{\beta}, \mathbf{z}_i')^2$$

This can be accomplished by setting $y_i = 0$ and $f(\boldsymbol{\beta}, \mathbf{z}_i') = \sqrt{(}-2\log\{\pi_i\})$. Because $0 \le \pi \le 1$, the function $-2\log\{\pi_i\}$ is strictly positive and the square root can be taken safely.

The following DATA step creates the data for the probit analysis. The variable like is created in the DATA step, and it contains the value 0 throughout. This variable serves as the "dummy" response variable in the PROC NLIN step. The variable remiss indicates whether cancer remission occurred. It is the binary outcome variable of interest and is used to determine the relevant probability for observation i as the success or failure probability of a Bernoulli experiment.

```
data remiss;
   input remiss cell smear infil li blast temp;
   label remiss = 'complete remission';
   like = 0;
   label like = 'dummy variable for nlin';
   datalines;
 1 0.8  .83 .66 1.9 1.10   .996
 1 0.9  .36 .32 1.4 0.74   .992
 0 0.8  .88 .70 0.8 0.176  .982
 0 1    .87 .87 0.7 1.053  .986
```

```
 1 0.9  .75 .68 1.3 0.519  .980
 0 1    .65 .65 0.6 0.519  .982
 1 0.95 .97 .92 1   1.23   .992
 0 0.95 .87 .83 1.9 1.354 1.020
 0 1    .45 .45 0.8 0.322  .999
 0 0.95 .36 .34 0.5 0     1.038
 0 0.85 .39 .33 0.7 0.279  .988
 0 0.7  .76 .53 1.2 0.146  .982
 0 0.8  .46 .37 0.4 0.38  1.006
 0 0.2  .39 .08 0.8 0.114  .990
 0 1    .90 .90 1.1 1.037  .990
 1 1    .84 .84 1.9 2.064 1.020
 0 0.65 .42 .27 0.5 0.114 1.014
 0 1    .75 .75 1   1.322 1.004
 0 0.5  .44 .22 0.6 0.114  .990
 1 1    .63 .63 1.1 1.072  .986
 0 1    .33 .33 0.4 0.176 1.010
 0 0.9  .93 .84 0.6 1.591 1.020
 1 1    .58 .58 1   0.531 1.002
 0 0.95 .32 .30 1.6 0.886  .988
 1 1    .60 .60 1.7 0.964  .990
 1 1    .69 .69 0.9 0.398  .986
 0 1    .73 .73 0.7 0.398  .986
;
```

The following NLIN statements fit the probit model:

```
proc nlin data=remiss method=newton sigsq=1;
   parms int=-10 a = -2 b = -1 c=6;

   linp = int + a*cell + b*li + c*temp;
   p   = probnorm(linp);

   if (remiss = 1) then pi = 1-p;
                   else pi = p;

   model.like = sqrt(- 2 * log(pi));
   output out=p p=predict;
run;
```

The assignment to the variable linp creates the linear predictor of the generalized linear model,

$$\eta = \beta_0 + \beta_1 \mathrm{cell}_i + \beta_2 \mathrm{li}_i + \beta_3 * \mathrm{temp}_i$$

In this example, the variables cell, li, and temp are used as regressors.

By default, the NLIN procedure computes the covariance matrix of the parameter estimates based on the nonlinear least squares assumption. That is, the procedure computes the estimate of the residual variance as the mean square error and uses that to multiply the inverse crossproduct matrix or the inverse Hessian matrix. (See the section "Covariance Matrix of Parameter Estimates" on page 7187 for details.) In the probit model, there is no residual variance. In addition, standard errors in maximum likelihood estimation are based on the inverse Hessian matrix. The METHOD=NEWTON option in the PROC NLIN statement is used to employ the Hessian matrix in computing the covariance matrix of the parameter estimates. The SIGSQ=1 option replaces the residual variance estimate that PROC NLIN would use by default as a multiplier of the inverse Hessian with the value 1.0.

Output 88.3.1 shows the results of this analysis. The analysis of variance table shows an apparently strange result. The total sum of squares is zero, and the model sum of squares is negative. Recall that the values of the response variable were set to zero and the mean function was constructed as $-2\log\{\pi_i\}$ in order for the NLIN procedure to minimize a log-likelihood function in terms of a nonlinear least squares problem. The value 21.9002 shown as the "Error" sum of squares is the value of the function $-2\log L$.

Output 88.3.1 Nonlinear Least-Squares Analysis from PROC NLIN

The NLIN Procedure

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	4	-21.9002	-5.4750	-5.75	.
Error	23	21.9002	0.9522		
Uncorrected Total	27	0			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
int	-36.7548	32.3607	-103.7	30.1885
a	-5.6298	4.6376	-15.2235	3.9639
b	-2.2513	0.9790	-4.2764	-0.2262
c	45.1815	34.9095	-27.0343	117.4

The problem can be more simply solved using dedicated procedures for generalized linear models:

```
proc glimmix data=remiss;
   model remiss = cell li temp / dist=binary link=probit s;
run;

proc genmod data=remiss;
   model remiss = cell li temp / dist=bin link=probit;
run;

proc logistic data=remiss;
   model remiss = cell li temp / link=probit technique=newton;
run;
```

Example 88.4: Affecting Curvature through Parameterization

(View the complete code for this example (nlinex4.sas) in the example repository.)

The work of Ratkowsky (1983, 1990) has brought into focus the importance of close-to-linear behavior of parameters in nonlinear regression models. The curvature in a nonlinear model consists of two components: the intrinsic curvature and the parameter-effects curvature. See the section "Relative Curvature Measures of Nonlinearity" on page 7170 for details. Intrinsic curvature expresses the degree to which the nonlinear model bends as values of the *parameters* change. This is not the same as the curviness of the model as a function of the covariates (the x variables). Intrinsic curvature is a function of the type of model you are fitting and the data. This curvature component cannot be affected by reparameterization of the model. According to Ratkowsky (1983), the intrinsic curvature component is typically smaller than the parameter-effects curvature, which can be affected by altering the parameterization of the model.

In models with low curvature, the nonlinear least squares parameter estimators behave similarly to least squares estimators in linear regression models, which have a number of desirable properties. If the model is correct, they are best linear unbiased estimators and are normally distributed if the model errors are normal (otherwise they are asymptotically normal). As you lower the curvature of a nonlinear model, you can expect that the parameter estimators approach the behavior of the linear regression model estimators: they behave "close to linear."

This example uses a simple data set and a commonly applied model for dose-response relationships to examine how the parameter-effects curvature can be reduced. The statistics by which an estimator's behavior is judged are Box's bias (Box 1971) and Hougaard's measure of skewness (Hougaard 1982, 1985).

The log-logistic model

$$\mathrm{E}[Y|x] = \delta + \frac{\alpha - \delta}{1 + \gamma \exp\{\beta \ln(x)\}}$$

is a popular model to express the response Y as a function of dose x. The response is bounded between the asymptotes α and δ. The term in the denominator governs the transition between the asymptotes and depends on two parameters, γ and β. The log-logistic model can be viewed as a member of a broader class of dose-response functions, those relying on *switch-on* or *switch-off* mechanisms (see, for example, Schabenberger and Pierce 2002, sec. 5.8.6). A switch function is usually a monotonic function $S(x, \boldsymbol{\theta})$ that takes values between 0 and 1. A switch-on function increases in x; a switch-off function decreases in x. In the log-logistic case, the function

$$S(x, [\beta, \gamma]) = \frac{1}{1 + \gamma \exp\{\beta \ln(x)\}}$$

is a switch-off function for $\beta > 0$ and a switch-on function for $\beta < 0$. You can write general dose-response functions with asymptotes simply as

$$\mathrm{E}[Y|x] = \mu_{\min} + (\mu_{\max} - \mu_{\min}) S(x, \boldsymbol{\theta})$$

The following DATA step creates a small data set from a dose-response experiment with response y:

```
data logistic;
   input dose y;
   logdose = log(dose);
   datalines;
0.009  106.56
0.035   94.12
0.07    89.76
0.15    60.21
0.20    39.95
0.28    21.88
0.50     7.46
;
```

A graph of these data is produced with the following statements:

```
proc sgplot data=logistic;
   scatter y=y x=dose;
   xaxis type=log logstyle=linear;
run;
```

Output 88.4.1 Observed Data in Dose-Response Experiment

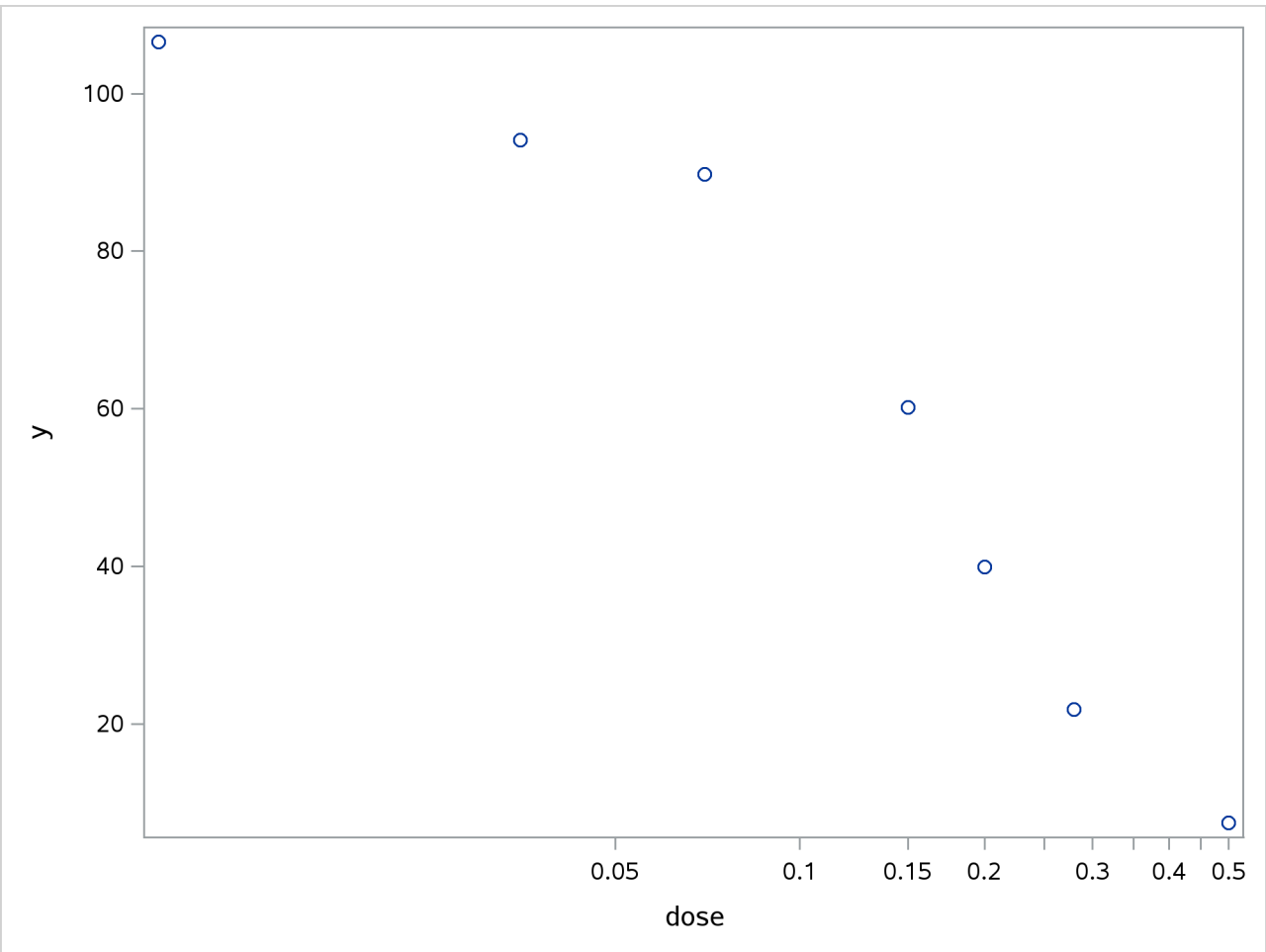

When dose is expressed on the log scale, the sigmoidal shape of the dose-response relationship is clearly visible (Output 88.4.1). The log-logistic switching model in the preceding parameterization is fit with the following statements in the NLIN procedure:

```
proc nlin data=logistic bias hougaard nlinmeasures;
   parameters alpha=100 beta=3 gamma=300;
   delta = 0;
   Switch = 1/(1+gamma*exp(beta*log(dose)));
   model y = delta + (alpha - delta)*Switch;
run;
```

The lower asymptote δ is assumed to be 0 in this case. Since δ is not listed in the PARAMETERS statement and is assigned a value in the program, it is assumed to be constant. Note that the term Switch is the switch-off function in the log-logistic model. The BIAS and HOUGAARD options in the PROC NLIN statement request that Box's bias, percentage bias, and Hougaard's skewness measure be added to the table of parameter estimates, and the NLINMEASURES option requests that the global nonlinearity measures be produced.

The NLIN procedure converges after 10 iterations and achieves a residual mean square error of 15.1869 (Output 88.4.2). This value is not that important by itself, but it is worth noting since this model fit is compared to the fit with other parameterizations later on.

Output 88.4.2 Iteration History and Analysis of Variance

The NLIN Procedure
Dependent Variable y
Method: Gauss-Newton

Iterative Phase				
Iter	alpha	beta	gamma	Sum of Squares
0	100.0	3.0000	300.0	386.4
1	100.4	2.8011	162.8	129.1
2	100.8	2.6184	101.4	69.2710
3	101.3	2.4266	69.7579	68.2167
4	101.7	2.3790	69.0358	60.8223
5	101.8	2.3621	67.3709	60.7516
6	101.8	2.3582	67.0044	60.7477
7	101.8	2.3573	66.9150	60.7475
8	101.8	2.3571	66.8948	60.7475
9	101.8	2.3570	66.8902	60.7475
10	101.8	2.3570	66.8892	60.7475

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	3	33965.4	11321.8	745.50	<.0001
Error	4	60.7475	15.1869		
Uncorrected Total	7	34026.1			

The table of parameter estimates displays the estimates of the three model parameters, their approximate

standard errors, 95% confidence limits, Hougaard's skewness measure, Box's bias, and percentage bias (Output 88.4.3). Parameters for which the skewness measure is less than 0.1 in absolute value and with percentage bias less than 1% exhibit very close-to-linear behavior, and skewness values less than 0.25 in absolute value indicate reasonably close-to-linear behavior (Ratkowsky 1990). According to these rules, the estimators $\widehat{\beta}$ and $\widehat{\gamma}$ suffer from substantial curvature. The estimator $\widehat{\gamma}$ is especially "far-from-linear." Inferences that involve $\widehat{\gamma}$ and rely on the reported standard errors or confidence limits (or both) for this parameter might be questionable.

Output 88.4.3 Parameter Estimates, Hougaard's Skewness, and Box's Bias

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness	Bias	Percent Bias
alpha	101.8	3.0034	93.4751	110.2	0.1415	0.1512	0.15
beta	2.3570	0.2928	1.5440	3.1699	0.4987	0.0303	1.29
gamma	66.8892	31.6146	-20.8870	154.7	1.9200	10.9230	16.3

The related global nonlinearity measures output table (Output 88.4.4) shows that both the maximum and RMS parameter-effects curvature are substantially larger than the critical curvature value recommended by Bates and Watts (1980). In contrast, the intrinsic curvatures of the model are less than the critical value. This implies that most of the nonlinearity can be removed by reparameterization.

Output 88.4.4 Global Nonlinearity Measures

Global Nonlinearity Measures	
Max Intrinsic Curvature	0.2397
RMS Intrinsic Curvature	0.1154
Max Parameter-Effects Curvature	4.0842
RMS Parameter-Effects Curvature	1.8198
Curvature Critical Value	0.3895
Raw Residual Variance	15.187
Projected Residual Variance	5.922

One method of reducing the parameter-effects curvature, and thereby reduce the bias and skewness of the parameter estimators, is to replace a parameter with its expected-value parameterization. Schabenberger et al. (1999) and Schabenberger and Pierce (2002, sec. 5.7.2) refer to this method as *reparameterization through defining relationships*. A defining relationship is obtained by equating the mean response at a chosen value of x (say, x^*) to the model:

$$\mathrm{E}[Y|x^*] = \delta + \frac{\alpha - \delta}{1 + \gamma \exp\{\beta \ln(x^*)\}}$$

This equation is then solved for a parameter that is subsequently replaced in the original equation. This method is particularly useful if x^* has an interesting interpretation. For example, let λ_K denote the value that reduces the response by $K \times 100\%$,

$$\mathrm{E}[Y|\lambda_K] = \delta + \left(\frac{100 - K}{100}\right)(\alpha - \delta)$$

Because γ exhibits large bias and skewness, it is the target in the first round of reparameterization. Setting the expression for the conditional mean at λ_K equal to the mean function when $x = \lambda_K$ yields the following

expression:

$$\delta + \left(\frac{100-K}{100}\right)(\alpha-\delta) = \delta + \frac{\alpha-\delta}{1+\gamma\exp\{\beta\ln(\lambda_K)\}}$$

This expression is solved for γ, and the result is substituted back into the model equation. This leads to a log-logistic model in which γ is replaced by the parameter λ_K, the dose at which the response was reduced by $K \times 100\%$. The new model equation is

$$\mathrm{E}[Y|x] = \delta + \frac{\alpha-\delta}{1+K/(100-K)\exp\{\beta\ln(x/\lambda_K)\}}$$

A particularly interesting choice is $K = 50$, since λ_{50} is the dose at which the response is halved. In studies of mortality, this concentration is also known as the LD50. For the special case of λ_{50} the model equation becomes

$$\mathrm{E}[Y|x] = \delta + \frac{\alpha-\delta}{1+\exp\{\beta\ln(x/\lambda_{50})\}}$$

You can fit the model in the LD50 parameterization with the following statements:

```
proc nlin data=logistic bias hougaard;
   parameters alpha=100 beta=3 LD50=0.15;
   delta = 0;
   Switch = 1/(1+exp(beta*log(dose/LD50)));
   model y = delta + (alpha - delta)*Switch;
   output out=nlinout pred=p lcl=lcl ucl=ucl;
run;
```

Partial results from this NLIN run are shown in Output 88.4.5. The analysis of variance tables in Output 88.4.2 and Output 88.4.5 are identical. Changing the parameterization of a model does not affect the model fit. It does, however, affect the interpretation of the parameters and the statistical properties (close-to-linear behavior) of the parameter estimators. The skewness and bias measures of the parameter LD50 is considerably reduced compared to those values for the parameter γ in the previous parameterization. Also, γ has been replaced by a parameter with a useful interpretation, the dose that yields a 50% reduction in mean response. Also notice that the bias and skewness measures of α and β are not affected by the $\gamma \rightarrow$ LD50 reparameterization.

Output 88.4.5 ANOVA Table and Parameter Estimates in LD50 Parameterization

The NLIN Procedure
Dependent Variable y
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	3	33965.4	11321.8	745.50	<.0001
Error	4	60.7475	15.1869		
Uncorrected Total	7	34026.1			

Output 88.4.5 *continued*

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness	Bias	Percent Bias
alpha	101.8	3.0034	93.4752	110.2	0.1415	0.1512	0.15
beta	2.3570	0.2928	1.5440	3.1699	0.4987	0.0303	1.29
LD50	0.1681	0.00915	0.1427	0.1935	-0.0605	-0.00013	-0.08

To reduce the parameter-effects curvature of the β parameter, you can use the technique of defining relationships again. This can be done generically, by solving

$$\mu^* = \delta + \frac{\alpha - \delta}{1 + \exp\{\beta \ln(x/\lambda)\}}$$

for β, treating μ^* as the new parameter (in lieu of β), and choosing a value for x^* that leads to low skewness. This results in the expected-value parameterization of β. Solving for β yields

$$\beta = \frac{\log\left(\frac{\alpha-\mu^*}{\mu^*-\delta}\right)}{\log\left(x^*/\lambda\right)}$$

The interpretation of the parameter μ^* that replaces β in the model equation is simple: it is the mean dose response when the dose is x^*. Fixing $x^* = 0.3$, the following PROC NLIN statements fit this model:

```
proc nlin data=logistic bias hougaard nlinmeasures;
   parameters alpha=100 mustar=20 LD50=0.15;
   delta   = 0;
   xstar   = 0.3;
   beta    = log((alpha - mustar)/(mustar - delta)) / log(xstar/LD50);
   Switch  = 1/(1+exp(beta*log(dose/LD50)));
   model y = delta + (alpha - delta)*Switch;
   output out=nlinout pred=p lcl=lcl ucl=ucl;
run;
```

Note that the switch-off function continues to be written in terms of β and the LD50. The only difference from the previous model is that β is now expressed as a function of the parameter μ^*. Using expected-value parameterizations is a simple mechanism to lower the curvature in a model and to arrive at starting values. The starting value for μ^* can be gleaned from Output 88.4.1 at $x = 0.3$.

Output 88.4.6 shows selected results from this NLIN run. The ANOVA table is again unaffected by the change in parameterization. The skewness for μ^* is significantly reduced in comparison to those of the β parameter in the previous model (compare Output 88.4.6 and Output 88.4.5), while its bias remains on the same scale from Output 88.4.5 to Output 88.4.6. Also note the substantial reduction in the parameter-effects curvature values. As expected, the intrinsic curvature values remain intact.

Output 88.4.6 ANOVA Table and Parameter Estimates in Expected-Value Parameterization

The NLIN Procedure
Dependent Variable y
Method: Gauss-Newton

NOTE: Convergence criterion met.

Output 88.4.6 *continued*

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	3	33965.4	11321.8	745.50	<.0001
Error	4	60.7475	15.1869		
Uncorrected Total	7	34026.1			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness	Bias	Percent Bias
alpha	101.8	3.0034	93.4752	110.2	0.1415	0.1512	0.15
mustar	20.7073	2.6430	13.3693	28.0454	-0.0572	-0.0983	-0.47
LD50	0.1681	0.00915	0.1427	0.1935	-0.0605	-0.00013	-0.08

Global Nonlinearity Measures	
Max Intrinsic Curvature	0.2397
RMS Intrinsic Curvature	0.1154
Max Parameter-Effects Curvature	0.2925
RMS Parameter-Effects Curvature	0.1500
Curvature Critical Value	0.3895
Raw Residual Variance	15.187
Projected Residual Variance	5.9219

Example 88.5: Comparing Nonlinear Trends among Groups

(View the complete code for this example (nlinex5.sas) in the example repository.)

When you model nonlinear trends in the presence of group (classification) variables, two questions often arise: whether the trends should be varied by group, and how to decide which parameters should be varied across groups. A large battery of tools is available on linear statistical models to test hypotheses involving the model parameters, especially to test linear hypotheses. To test similar hypotheses in nonlinear models, you can draw on analogous tools. Especially important in this regard are comparisons of nested models by contrasting their residual sums of squares.

In this example, a two-group model from a pharmacokinetic application is fit to data that are in part based on the theophylline data from Pinheiro and Bates (1995) and the first example in the documentation for the NLMIXED procedure. In a pharmacokinetic application you study how a drug is dispersed through a living organism. The following data represent concentrations of the drug theophylline over a 25-hour period following oral administration. The data are derived by collapsing and averaging the subject-specific data from Pinheiro and Bates (1995) in a particular, yet unimportant, way. The purpose of arranging the data in this way is purely to demonstrate the methodology.

```
data theop;
   input time dose conc @@;
   if (dose = 4) then group=1; else group=2;
   datalines;
 0.00   4  0.1633  0.25   4   2.045
```

```
  0.27   4     4.4  0.30   4    7.37
  0.35   4    1.89  0.37   4    2.89
  0.50   4    3.96  0.57   4    6.57
  0.58   4     6.9  0.60   4     4.6
  0.63   4    9.03  0.77   4    5.22
  1.00   4    7.82  1.02   4   7.305
  1.05   4    7.14  1.07   4     8.6
  1.12   4    10.5  2.00   4    9.72
  2.02   4    7.93  2.05   4    7.83
  2.13   4    8.38  3.50   4    7.54
  3.52   4    9.75  3.53   4    5.66
  3.55   4   10.21  3.62   4     7.5
  3.82   4    8.58  5.02   4   6.275
  5.05   4    9.18  5.07   4    8.57
  5.08   4     6.2  5.10   4    8.36
  7.02   4    5.78  7.03   4    7.47
  7.07   4   5.945  7.08   4    8.02
  7.17   4    4.24  8.80   4    4.11
  9.00   4     4.9  9.02   4    5.33
  9.03   4    6.11  9.05   4    6.89
  9.38   4    7.14 11.60   4    3.16
 11.98   4    4.19 12.05   4    4.57
 12.10   4    5.68 12.12   4    5.94
 12.15   4     3.7 23.70   4    2.42
 24.15   4    1.17 24.17   4    1.05
 24.37   4    3.28 24.43   4    1.12
 24.65   4    1.15  0.00   5   0.025
  0.25   5    2.92  0.27   5   1.505
  0.30   5    2.02  0.50   5   4.795
  0.52   5    5.53  0.58   5    3.08
  0.98   5   7.655  1.00   5   9.855
  1.02   5    5.02  1.15   5    6.44
  1.92   5    8.33  1.98   5    6.81
  2.02   5  7.8233  2.03   5    6.32
  3.48   5    7.09  3.50   5   7.795
  3.53   5    6.59  3.57   5    5.53
  3.60   5    5.87  5.00   5     5.8
  5.02   5  6.2867  5.05   5    5.88
  6.98   5    5.25  7.00   5    4.02
  7.02   5    7.09  7.03   5   4.925
  7.15   5    4.73  9.00   5    4.47
  9.03   5    3.62  9.07   5    4.57
  9.10   5     5.9  9.22   5    3.46
 12.00   5    3.69 12.05   5    3.53
 12.10   5    2.89 12.12   5    2.69
 23.85   5    0.92 24.08   5    0.86
 24.12   5    1.25 24.22   5    1.15
 24.30   5     0.9 24.35   5    1.57
;
```

The following code plots the theophylline concentration data over time for the two groups (Output 88.5.1). In each group the concentration tends to rise sharply right after the drug is administered, followed by a prolonged tapering of the concentration.

```
proc sgplot data=theop;
   scatter x=time y=conc / group=group;
   yaxis label='Concentration';
   xaxis label='Time';
run;
```

Output 88.5.1 Observed Responses in Two Groups

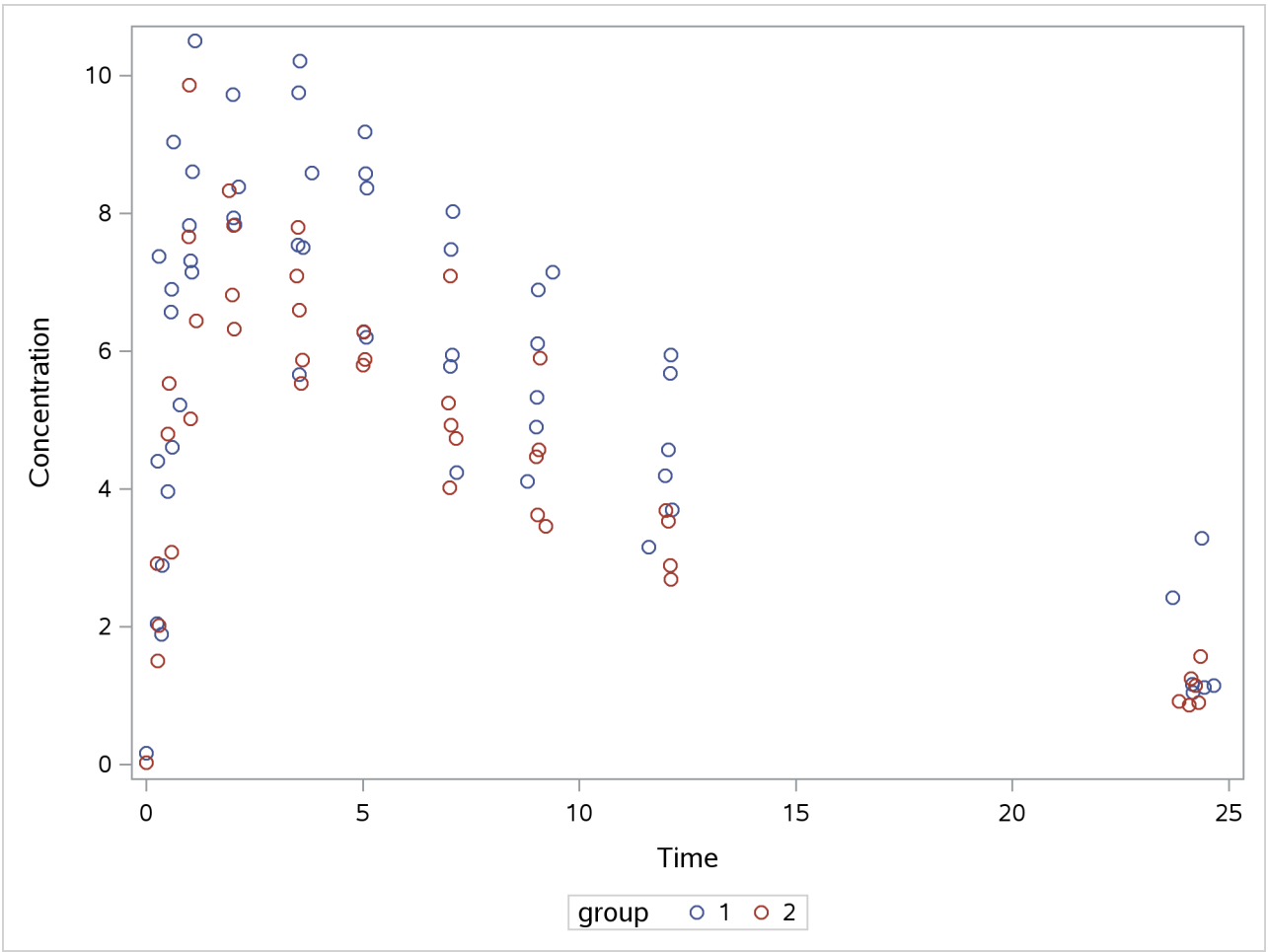

In the context of nonlinear mixed models, Pinheiro and Bates (1995) consider a first-order compartment model for these data. In terms of two fixed treatment groups, the model can be written as

$$C_{it} = \frac{Dk_{e_i}k_{a_i}}{Cl_i(k_{a_i} - k_{e_i})}[\exp(-k_{e_i}t) - \exp(-k_{a_i}t)] + \epsilon_{it}$$

where C_{it} is the observed concentration in group i at time t, D is the dose of theophylline, k_{e_i} is the elimination rate in group i, k_{a_i} is the absorption rate in group i, Cl_i is the clearance in group i, and ϵ_{it} denotes the model error. Because the rates and the clearance must be positive, you can parameterize the

model in terms of log rates and the log clearance:

$$\begin{aligned} Cl_i &= \exp\{\beta_{1i}\} \\ k_{a_i} &= \exp\{\beta_{2i}\} \\ k_{e_i} &= \exp\{\beta_{3i}\} \end{aligned}$$

In this parameterization the model contains six parameters, and the rates and clearance vary by group. This produces two separate response profiles, one for each group. On the other extreme, you could model the trends as if there were no differences among the groups:

$$\begin{aligned} Cl_i &= \exp\{\beta_1\} \\ k_{a_i} &= \exp\{\beta_2\} \\ k_{e_i} &= \exp\{\beta_3\} \end{aligned}$$

In between these two extremes lie other models, such as a model where both groups have the same absorption and elimination rate, but different clearances. The question then becomes how to go about building a model in an organized manner.

To test hypotheses about nested nonlinear models, you can apply the idea of a "Sum of Squares Reduction Test." A reduced model is nested within a full model if you can impose q constraints on the full model to obtain the reduced model. Then, if SSE_r and SSE_f denote the residual sum of squares in the reduced and the full model, respectively, the test statistic is

$$F_R = \frac{\left(\mathrm{SSE}_r - \mathrm{SSE}_f\right)/q}{\mathrm{SSE}_f/(n-p)} = \frac{\left(\mathrm{SSE}_r - \mathrm{SSE}_f\right)/q}{\mathrm{MSE}_f}$$

where n are the number of observations used and p are the number of parameters in the full model. The numerator of the F_R statistic is the average reduction in residual sum of squares per constraint. The mean square error of the full model is used to scale this average because it is less likely to be a biased estimator of the residual variance than the variance estimate from a constrained (reduced) model. The F_R statistic is then compared against quantiles from an F distribution with q numerator and $n-p$ denominator degrees of freedom. Schabenberger and Pierce (2002) discuss the justification for this test and compare it to other tests in nonlinear models.

In the present application we might phrase the initial question akin to the overall F test for a factor in a linear model: Should any parameters be varied between the two groups? The corresponding null hypothesis is

$$H: \begin{cases} \beta_{11} = \beta_{12} \\ \beta_{21} = \beta_{22} \\ \beta_{31} = \beta_{32} \end{cases}$$

where the first subscript identifies the type of the parameter and the second subscript identifies the group. Note that this hypothesis implies

$$H: \begin{cases} Cl_1 = Cl_2 \\ k_{a_1} = k_{a_2} \\ k_{e_1} = k_{e_2} \end{cases}$$

If you fail to reject this hypothesis, there is no need to further examine individual parameter differences.

The reduced model—the model subject to the null hypothesis—is fit with the following PROC NLIN statements:

```
proc nlin data=theop;
   parms beta1=-3.22 beta2=0.47 beta3=-2.45;
   cl   = exp(beta1);
   ka   = exp(beta2);
   ke   = exp(beta3);
   mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
   model conc = mean;
   ods output Anova=aovred(rename=(ss=ssred ms=msred df=dfred));
run;
```

The clearance, the rates, and the mean function are formed independently of the group membership. The analysis of variance table is saved to the data set aovred, and some of its variables are renamed. This is done so that the data set can be merged easily with the analysis of variance table for the full model (see following).

The converged model has a residual sum of square of $\mathrm{SSE}_r = 286.4$ and a mean square error of 3.0142 (Output 88.5.2). The table of parameter estimates gives the values for the estimated log clearance ($\widehat{\beta}_1 = -3.2991$), the estimated log absorption rate ($\widehat{\beta}_2 = 0.4769$), and the estimated log elimination rate ($\widehat{\beta}_3 = -2.5555$).

Output 88.5.2 Fit Results for the Reduced Model

The NLIN Procedure
Dependent Variable conc
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	3	3100.5	1033.5	342.87	<.0001
Error	95	286.4	3.0142		
Uncorrected Total	98	3386.8			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1	-3.2991	0.0956	-3.4888	-3.1094
beta2	0.4769	0.1640	0.1512	0.8025
beta3	-2.5555	0.1410	-2.8354	-2.2755

The full model, in which all three parameters are varied by group, can be fit with the following statements in the NLIN procedure:

```
proc nlin data=theop;
   parms beta1_1=-3.22 beta2_1=0.47 beta3_1=-2.45
         beta1_2=-3.22 beta2_2=0.47 beta3_2=-2.45;
   if (group=1) then do;
      cl   = exp(beta1_1);
      ka   = exp(beta2_1);
      ke   = exp(beta3_1);
   end; else do;
```

```
      cl   = exp(beta1_2);
      ka   = exp(beta2_2);
      ke   = exp(beta3_2);
   end;
   mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
   model conc = mean;
   ods output Anova=aovfull;
run;
```

Separate parameters for the groups are now specified in the PARMS statement, and the value of the model variables cl, ka, and ke is assigned conditional on the group membership of an observation. Notice that the same expression as in the previous run can be used to model the mean function.

The results from this PROC NLIN run are shown in Output 88.5.3. The residual sum of squares in the full model is only $\mathrm{SSE}_f = 138.9$, compared to $\mathrm{SSE}_r = 286.4$ in the reduced model (Output 88.5.3).

Output 88.5.3 Fit Results for the Full Model

The NLIN Procedure
Dependent Variable conc
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	6	3247.9	541.3	358.56	<.0001
Error	92	138.9	1.5097		
Uncorrected Total	98	3386.8			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1_1	-3.5671	0.0864	-3.7387	-3.3956
beta2_1	0.4421	0.1349	0.1742	0.7101
beta3_1	-2.6230	0.1265	-2.8742	-2.3718
beta1_2	-3.0111	0.1061	-3.2219	-2.8003
beta2_2	0.3977	0.1987	0.00305	0.7924
beta3_2	-2.4442	0.1618	-2.7655	-2.1229

Whether this reduction in sum of squares is sufficient to declare that the full model provides a significantly better fit than the reduced model depends on the number of constraints imposed on the full model and on the variability in the data. In other words, before drawing any conclusions, you have to take into account how many parameters have been dropped from the model and how much variation around the regression trends the data exhibit. The F_R statistic sets these quantities into relation. The following macro merges the analysis of variance tables from the full and reduced model, and computes F_R and its p-value:

```
%macro SSReductionTest;
   data aov; merge aovred aovfull;
      if (Source='Error') then do;
         Fstat  = ((SSred-SS)/(dfred-df))/ms;
```

```
         pvalue = 1-Probf(Fstat,dfred-df,df);
         output;
      end;
   run;
   proc print data=aov label noobs;
      label Fstat  = 'F Value'
            pValue = 'Prob > F';
      format pvalue pvalue8.;
      var Fstat pValue;
   run;
%mend;
%SSReductionTest;
```

Output 88.5.4 F Statistic and P-value for Hypothesis of Equal Trends

F Value	Prob > F
32.5589	<.000001

There is clear evidence that the model with separate trends fits these data significantly better (Output 88.5.4). To decide whether all parameters should be varied between the groups or only one or two of them, we first refit the model in a slightly different parameterization:

```
proc nlin data=theop;
   parms beta1_1=-3.22 beta2_1=0.47 beta3_1=-2.45
         beta1_diff=0 beta2_diff=0 beta3_diff=0;
   if (group=1) then do;
      cl   = exp(beta1_1);
      ka   = exp(beta2_1);
      ke   = exp(beta3_1);
   end; else do;
      cl   = exp(beta1_1 + beta1_diff);
      ka   = exp(beta2_1 + beta2_diff);
      ke   = exp(beta3_1 + beta3_diff);
   end;
   mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
   model conc = mean;
run;
```

In the preceding statements, the parameters in the second group were expressed using offsets from parameters in the first group. For example, the parameter beta1_diff measures the change in log clearance between group 2 and group 1.

This simple reparameterization does not affect the model fit. The analysis of variance tables in Output 88.5.5 and Output 88.5.3 are identical. It does, however, affect the interpretation of the estimated quantities. Since the parameter beta1_diff measures the change in the log clearance rates between the groups, you can use the approximate 95% confidence limits in Output 88.5.5 to assess whether that quantity in the pharmacokinetic equation varies between groups. Only the confidence interval for the difference in the log clearances excludes 0. The intervals for beta2_diff and beta3_diff include 0.

Output 88.5.5 Fit Results for the Full Model in Difference Parameterization

The NLIN Procedure
Dependent Variable conc
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	6	3247.9	541.3	358.56	<.0001
Error	92	138.9	1.5097		
Uncorrected Total	98	3386.8			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1_1	-3.5671	0.0864	-3.7387	-3.3956
beta2_1	0.4421	0.1349	0.1742	0.7101
beta3_1	-2.6230	0.1265	-2.8742	-2.3718
beta1_diff	0.5560	0.1368	0.2842	0.8278
beta2_diff	-0.0444	0.2402	-0.5214	0.4326
beta3_diff	0.1788	0.2054	-0.2291	0.5866

This suggests as the final model one where the absorption and elimination rates are the same for both groups and only the clearances are varied. The following statements fit this model and perform the sum of squares reduction test:

```
proc nlin data=theop;
   parms beta1_1=-3.22 beta2_1=0.47 beta3_1=-2.45
         beta1_diff=0;
   ka = exp(beta2_1);
   ke = exp(beta3_1);
   if (group=1) then do;
      cl = exp(beta1_1);
   end; else do;
      cl = exp(beta1_1 + beta1_diff);
   end;
   mean = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
   model conc = mean;
   ods output Anova=aovred(rename=(ss=ssred ms=msred df=dfred));
   output out=predvals predicted=p;
run;

%SSReductionTest;
```

The results for this model with common absorption and elimination rates are shown in Output 88.5.6. The sum-of-squares reduction test comparing this model against the full model with six parameters shows—as expected—that the full model does not fit the data significantly better ($p = 0.6193$, Output 88.5.7).

Output 88.5.6 Fit Results for Model with Common Rates

The NLIN Procedure
Dependent Variable conc
Method: Gauss-Newton

NOTE: Convergence criterion met.

Note: An intercept was not specified for this model.

Source	DF	Sum of Squares	Mean Square	F Value	Approx Pr > F
Model	4	3246.5	811.6	543.60	<.0001
Error	94	140.3	1.4930		
Uncorrected Total	98	3386.8			

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits	
beta1_1	-3.5218	0.0681	-3.6570	-3.3867
beta2_1	0.4226	0.1107	0.2028	0.6424
beta3_1	-2.5571	0.0988	-2.7532	-2.3610
beta1_diff	0.4346	0.0454	0.3444	0.5248

Output 88.5.7 F Statistic and P-value for Hypothesis of Common Rates

F Value	Prob > F
0.48151	0.619398

A plot of the observed and predicted values for this final model is obtained with the following statements:

```
proc sgplot data=predvals;
   scatter x=time y=conc / group=group;
   series  x=time y=p    / group=group name='fit';
   keylegend 'fit'       / across=2 title='Group';
   yaxis label='Concentration';
   xaxis label='Time';
run;
```

The plot is shown in Output 88.5.8.

Output 88.5.8 Observed and Fitted Values for Theophylline Data

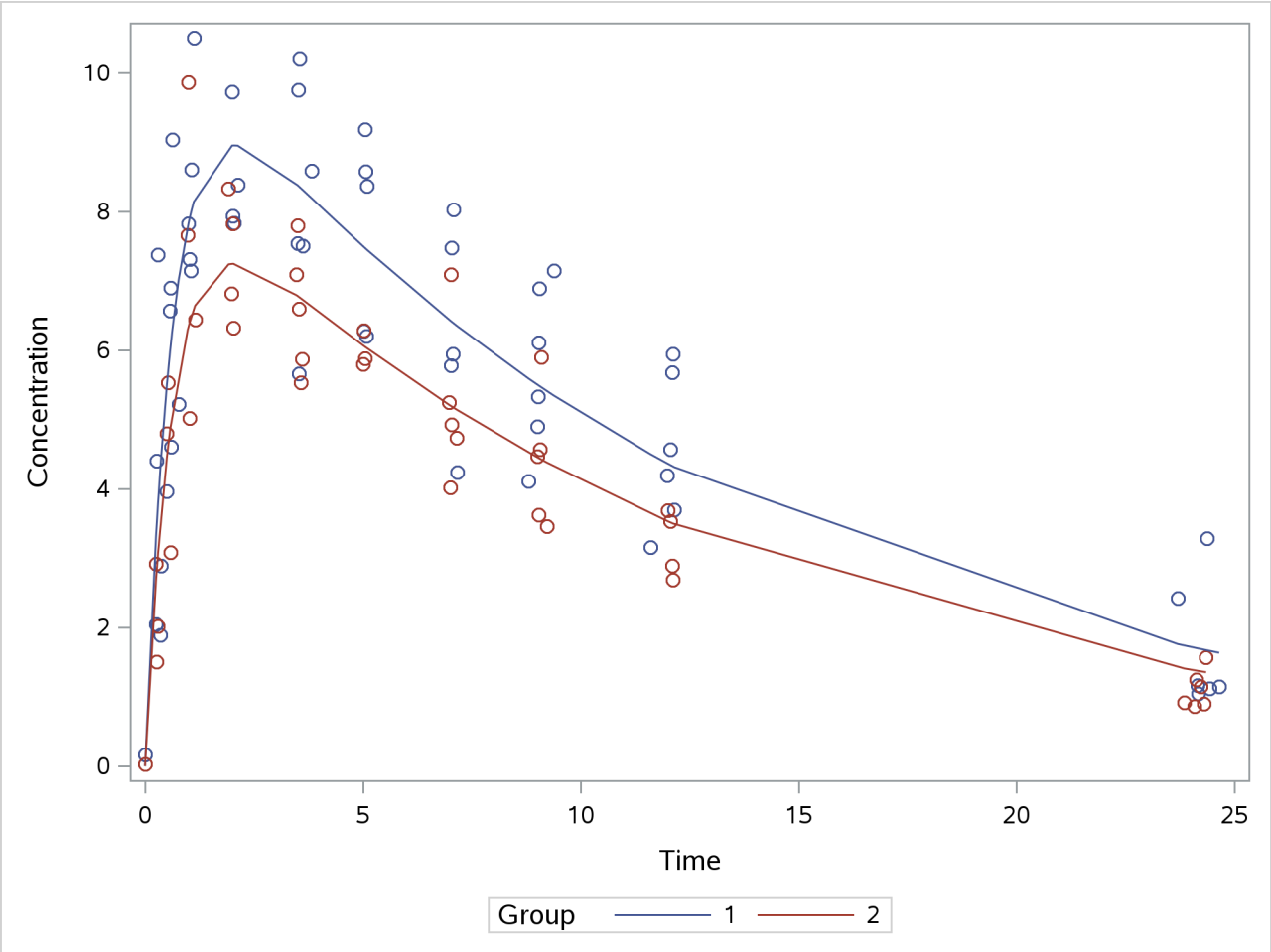

The sum-of-squares reduction test is not the only possibility of performing linear-model style hypothesis testing in nonlinear models. You can also perform Wald-type testing of linear hypotheses about the parameter estimates. See Example 52.17 in Chapter 52, "The GLIMMIX Procedure," for an application of this example that uses the NLIN and GLIMMIX procedures to compare the parameters across groups and adjusts *p*-values for multiplicity.

Example 88.6: ODS Graphics and Diagnostics

(View the complete code for this example (nlinex6.sas) in the example repository.)

The model in this example, taken from St. Laurent and Cook (1993), shows an unusual behavior in that the intrinsic curvature is substantially larger than the parameter-effects curvature. This example demonstrates how the diagnostics features of PROC NLIN can be used to perform postconvergence diagnostics.

The model takes the form

$$\mathrm{E}[Y\,|x_1, x_2] = \alpha x_1 + \exp\{\gamma x_2\}$$

The following DATA step creates a small data set to be used in this example:

```
data contrived;
   input x1 x2 y;
   datalines;
-4.0   -2.5  -10.0
-3.0   -2.0   -5.0
-2.0   -1.5   -2.0
-1.0   -1.0   -1.0
 0.0    0.0    1.5
 1.0    1.0    4.0
 2.0    1.5    5.0
 3.0    2.0    6.0
 4.0    2.5    7.0
-3.5   -2.2   -7.1
-3.5   -1.7   -5.1
 3.5    0.7    6.1
 2.5    1.2    7.5
;
```

The model is fit with the following statements in the NLIN procedure:

```
ods graphics on;
proc nlin data=contrived bias hougaard
        NLINMEASURES plots(stats=all)=(diagnostics);
   parms alpha=2.0
         gamma=0.0;
   model y = alpha*x1 + exp(gamma*x2);
run;
ods graphics off;
```

Output 88.6.1 Bias, Skewness, and Global Nonlinearity Measures

The NLIN Procedure

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Skewness	Bias	Percent Bias
alpha	1.9378	0.4704	0.9024	2.9733	6.6491	0.5763	29.7
gamma	0.0718	0.7923	-1.6720	1.8156	-7.5596	-0.9982	-1390

Global Nonlinearity Measures	
Max Intrinsic Curvature	6.6007
RMS Intrinsic Curvature	4.0421
Max Parameter-Effects Curvature	3.5719
RMS Parameter-Effects Curvature	2.1873
Curvature Critical Value	0.5011
Raw Residual Variance	2.1722
Projected Residual Variance	1.4551

The bias, skewness, and both the maximum and RMS intrinsic curvatures, compared to the critical curvature value, show that the model is highly nonlinear (Output 88.6.1). As such, performing diagnostics with the raw residuals can be problematic because they might have undesirable statistical properties: a nonzero mean and a negative semidefinite (instead of zero) covariance with the predicted values and different variances. In

addition, the use of tangential leverage is questionable in this case.

The partial results from this NLIN run are shown in Output 88.6.2, Output 88.6.3, and Output 88.6.4. The diagnostics plots corroborate the previously mentioned expectations: highly correlated raw residuals (with the predicted values), significant differences between tangential and Jacobian leverages and projected residuals which overcome some of the shortcomings of the raw residuals. Finally, considering the large intrinsic curvature, reparameterization might not make the model close-to-linear, perhaps necessitating the construction of another model.

Output 88.6.2 Diagnostics Panel

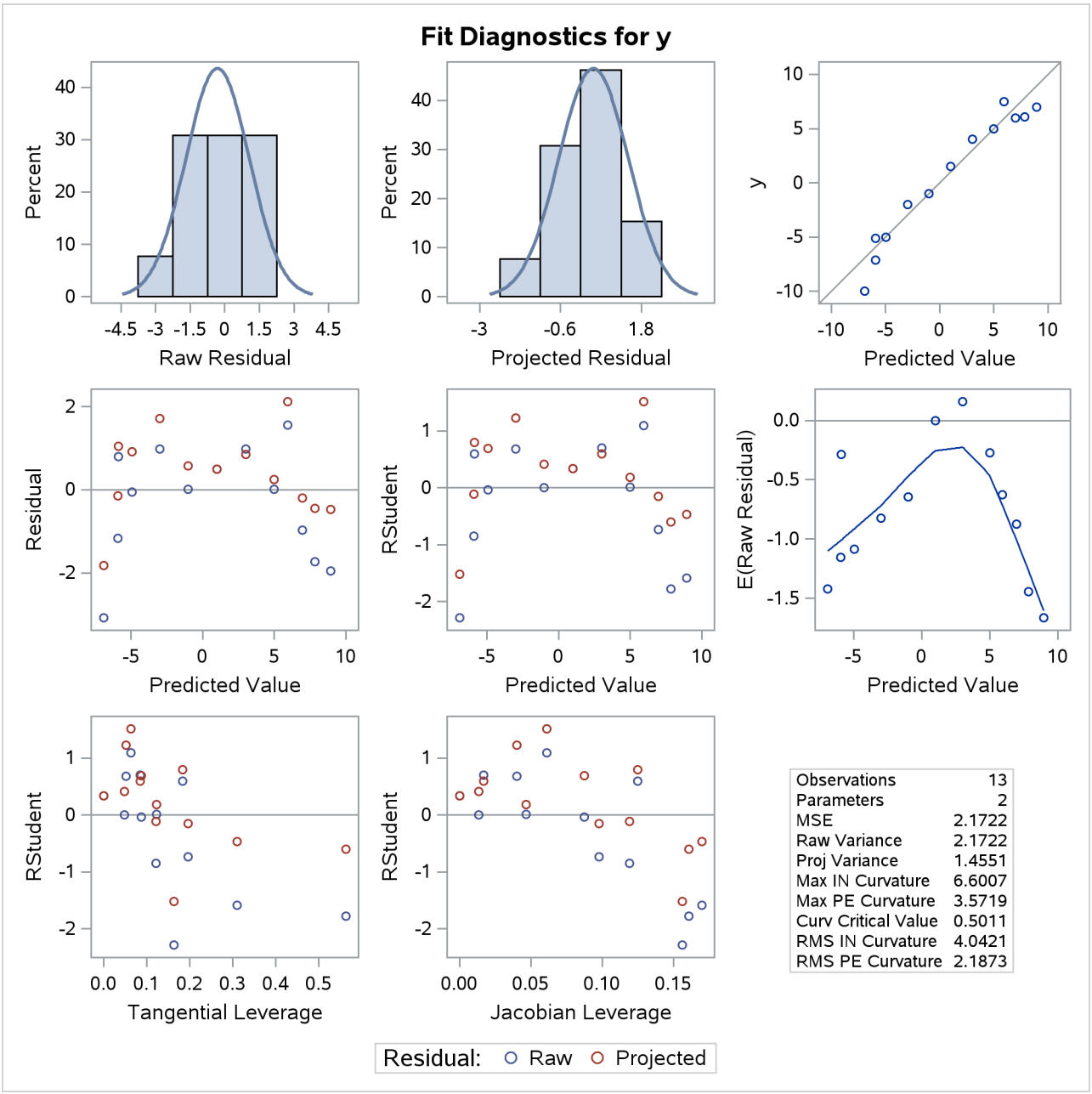

Output 88.6.3 Leverage Plots

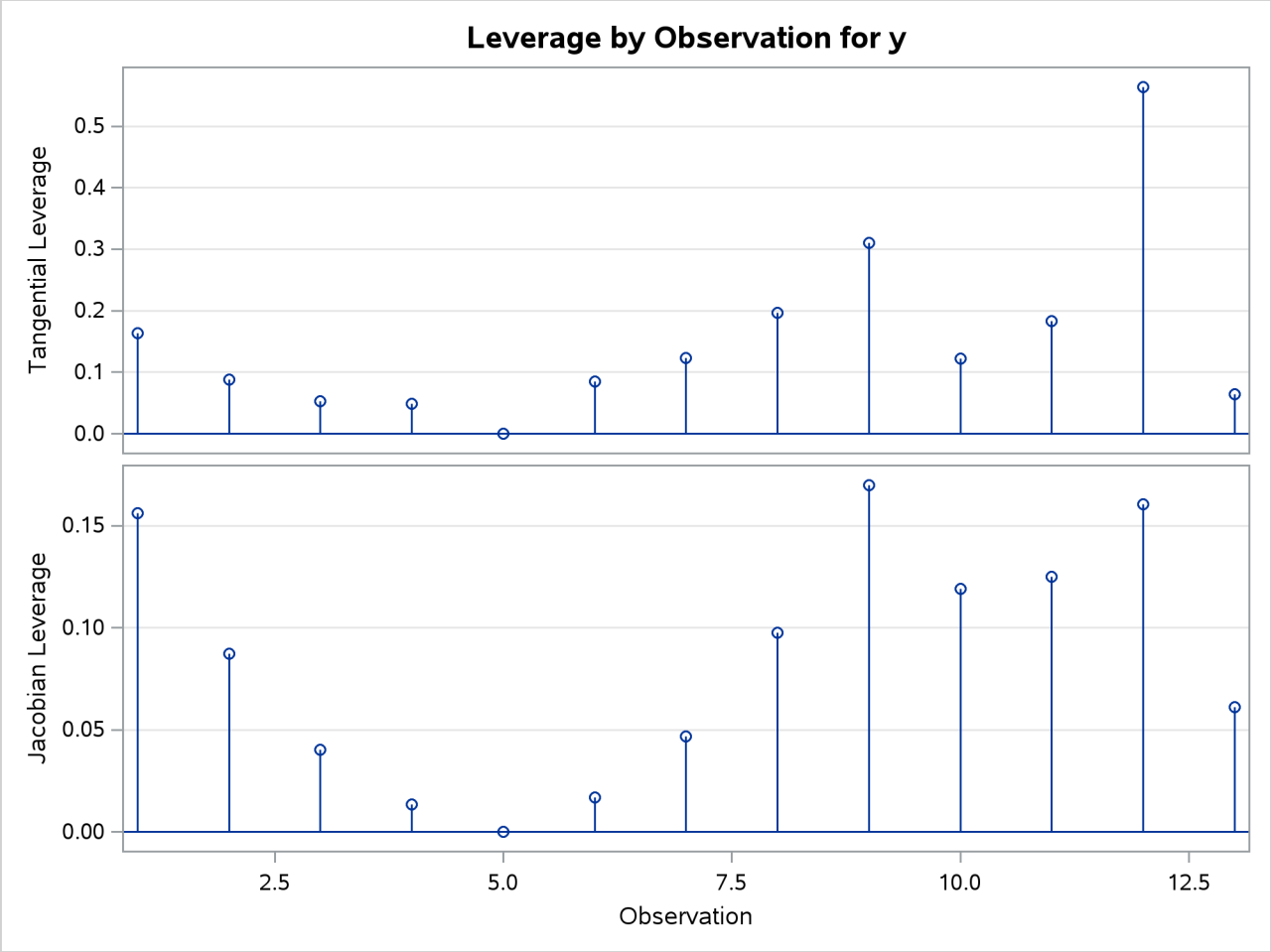

Output 88.6.4 Local Influence Plot

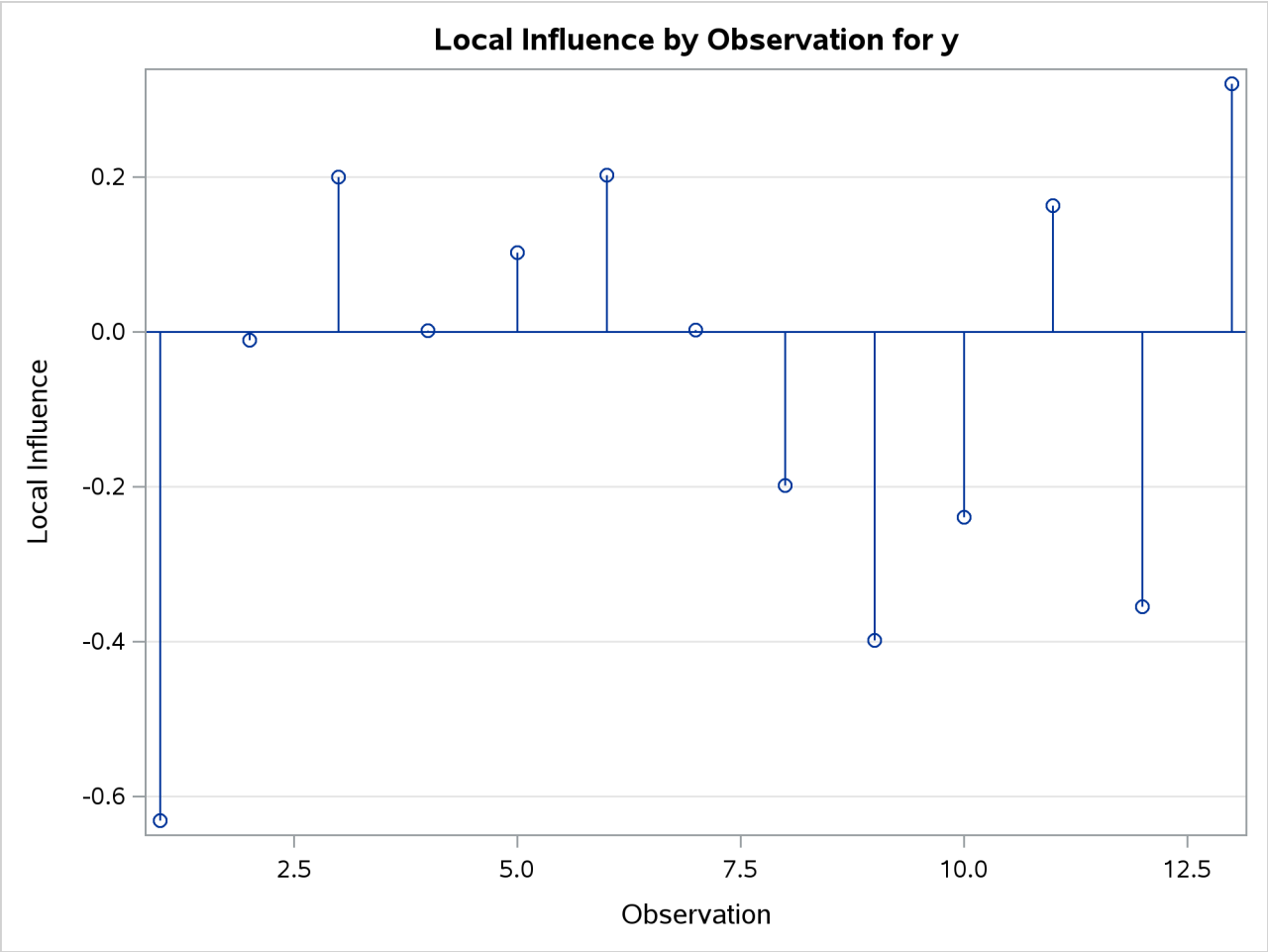

Example 88.7: Parameter Profiling and Bootstrapping

(View the complete code for this example (nlinex7.sas) in the example repository.)

This example, which uses a model from Clarke (1987), shows why it is difficult to intuitively reason about the characteristics of the parameters that occur in a nonlinear model. As a consequence, the example demonstrates the indispensability of parameter profiling and confidence curves to identify the nonlinear characteristics of model parameters. The parameter profiling is also augmented with a plot that shows the influence of each observation on the parameter estimates. Furthermore, the BOOTSTRAP statement in PROC NLIN is invoked in order to study the sampling distribution of parameter estimates and make more accurate statistical inferences.

The model takes the form

$$\mathrm{E}[Y|x] = \theta_3 + \theta_2 \exp\{\theta_1 x\}$$

The data set in this example is from Clarke (1987). The following DATA step creates this data set:

```
data clarke1987a;
   input  x y;
   datalines;
1    3.183
2    3.059
3    2.871
4    2.622
5    2.541
6    2.184
7    2.110
8    2.075
9    2.018
10   1.903
11   1.770
12   1.762
13   1.550
;
```

The model is fit by using the following statements in the NLIN procedure:

```
ods graphics on;
proc nlin data=clarke1987a plots(stats=none)=diagnostics;
   parms theta1=-0.15
         theta2=2.0
         theta3=0.80;
   profile theta1 theta3 / range = -6 to 2 by 0.2 all;
   bootstrap / nsamples = 2000 seed=123 bootplots bootci bootcov;
   model y = theta3 + theta2*exp(theta1*x);
run;
ods graphics off;
```

Output 88.7.1 Parameter Estimates and Bootstrap Estimate of the Covariance Matrix

The NLIN Procedure

Parameter	Estimate	Approx Std Error	Approximate 95% Confidence Limits		Bootstrap Std Dev	Bootstrap Bias-Corrected 95% Confidence Limits	
theta1	-0.1031	0.0255	-0.1599	-0.0462	0.0253	-0.1505	-0.0531
theta2	2.5190	0.2658	1.9268	3.1112	0.4449	2.1716	3.5325
theta3	0.9631	0.3216	0.2466	1.6797	0.4907	-0.2565	1.3697

Bootstrap Covariance Matrix Estimate			
	theta1	**theta2**	**theta3**
theta1	0.00063801	0.00834109	-.01006267
theta2	0.00834109	0.19795970	-.21564243
theta3	-.01006267	-.21564243	0.24076093

By default, the BOOTCI option in the BOOTSTRAP statement adds a column that contains the bias-corrected bootstrap confidence limits to the "Parameter Estimates" table (Output 88.7.1). The BOOTCOV option produces the "Bootstrap Covariance Matrix" table (Output 88.7.1). The discrepancy between the Wald-based

confidence limits and the bias-corrected bootstrap confidence limits is much greater for θ_3 than for θ_1. The profile t plot, confidence curve, and histograms of the bootstrap estimates show why this is the case.

The profile t plot for parameter θ_3 in Output 88.7.3 shows a definite deviation from the linear reference line that has a slope of 1 and passes through the origin. Hence, Wald-based inference for θ_3 is not appropriate. In contrast, the profile t plot for parameter θ_1 in Output 88.7.2 shows that Wald-based inference for θ_1 might be sufficient.

Output 88.7.4 and Output 88.7.5 show the confidence curves for θ_1 and θ_3, respectively. For θ_3, you can see a significant difference between the Wald-based confidence interval and the corresponding likelihood-based interval. In such cases, the likelihood-based intervals are preferred because their coverage rate is much closer to the nominal values than the coverage rate of the Wald-based intervals (Donaldson and Schnabel 1987; Cook and Weisberg 1990).

Output 88.7.6 depicts the influence of each observation on the value of θ_3. Observations 6 and 13 have the most influence on the value of this parameter. The plot is generated using the leave-one-out method and should be contrasted with the local influence plot in Output 88.7.7, which is based on assessing the influence of an additive perturbation of the response variable.

Output 88.7.8 and Output 88.7.9 are histograms that show the distribution of the bootstrap parameter estimates for θ_1 and θ_3, respectively. These histograms complement the information that is obtained about θ_1 and θ_3 from the profile t plots. Specifically, they show that the bootstrap parameter estimate of θ_1 has a distribution close to normal, whereas that of θ_3 has a distribution that deviates significantly from normal. Again, this leads to the conclusion that inferences based on linear approximations, such as Wald-based confidence intervals, work better for θ_1 than for θ_3.

Finally, this example shows that the adequacy of a linear approximation with regard to a certain parameter cannot be inferred directly from the model. If it could, then θ_3, which enters the model linearly, would have a completely linear behavior, whereas θ_1 would have a highly nonlinear behavior. However, the diagnostics that are based on the profile t plot and confidence curves, and the histograms of the bootstrap parameter estimates, show that the opposite holds. For a detailed discussion about this issue, see Cook and Weisberg (1990).

Output 88.7.2 Profile *t* Plot

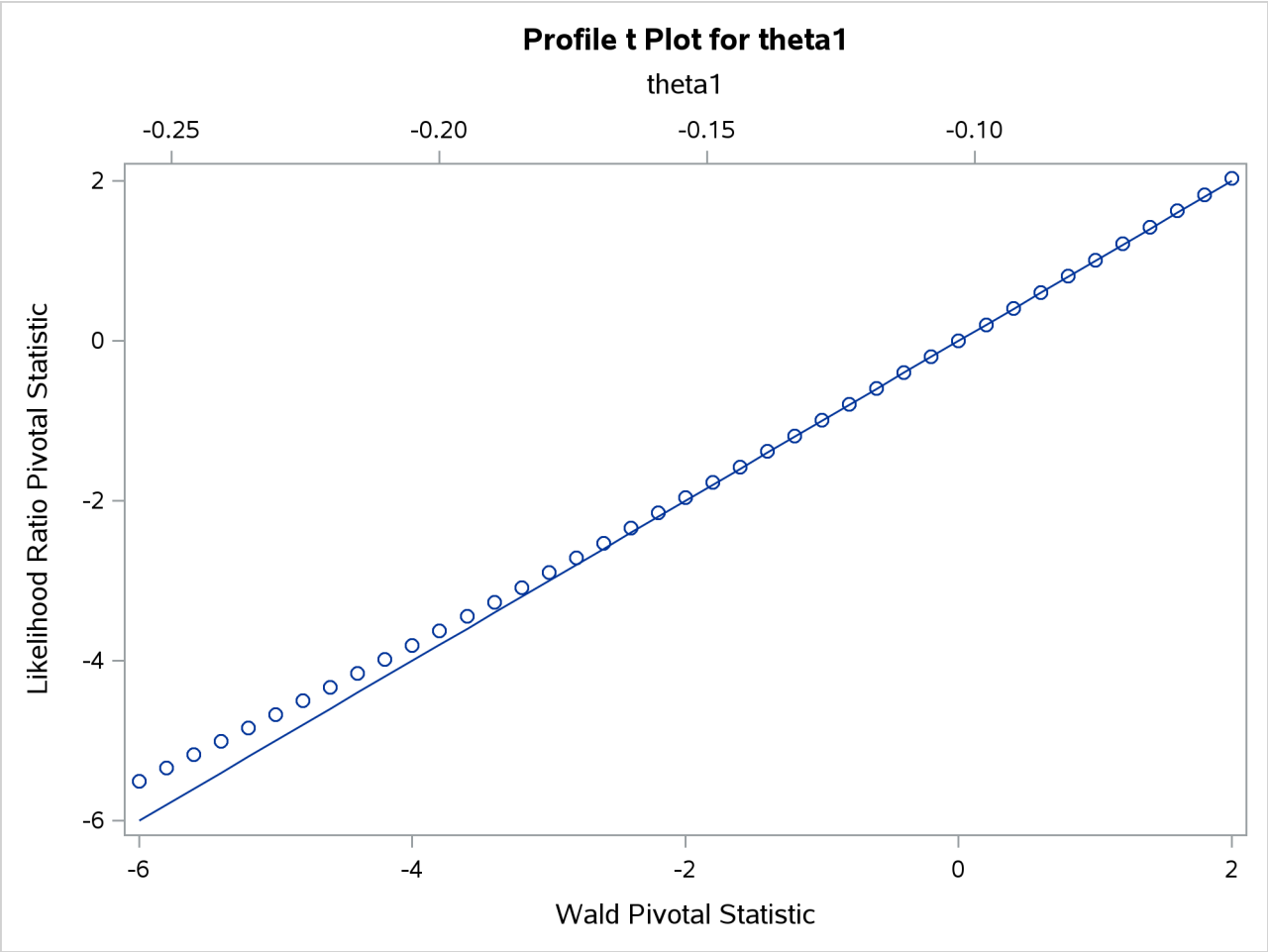

Output 88.7.3 Profile *t* Plot

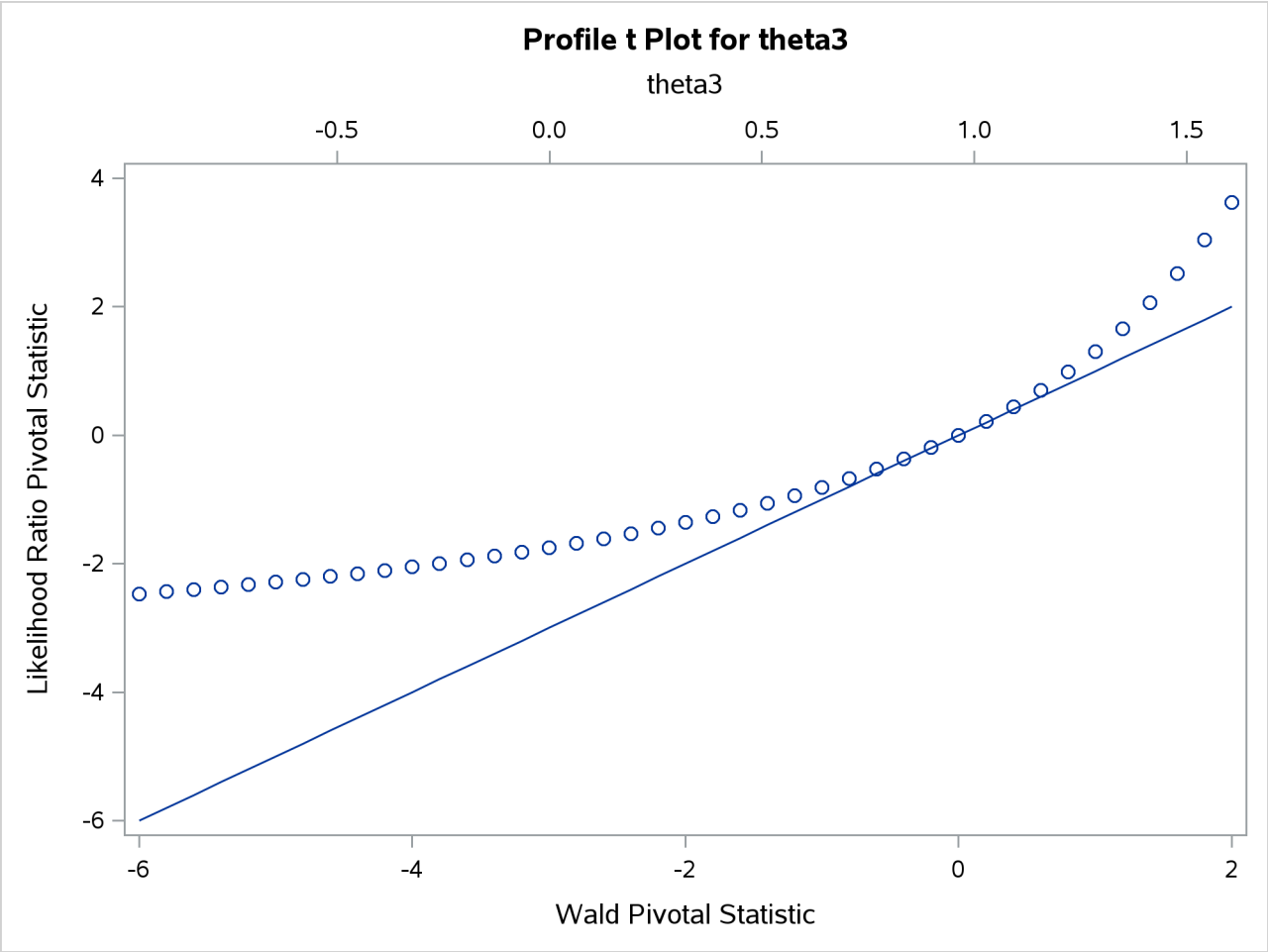

Output 88.7.4 Confidence Curve

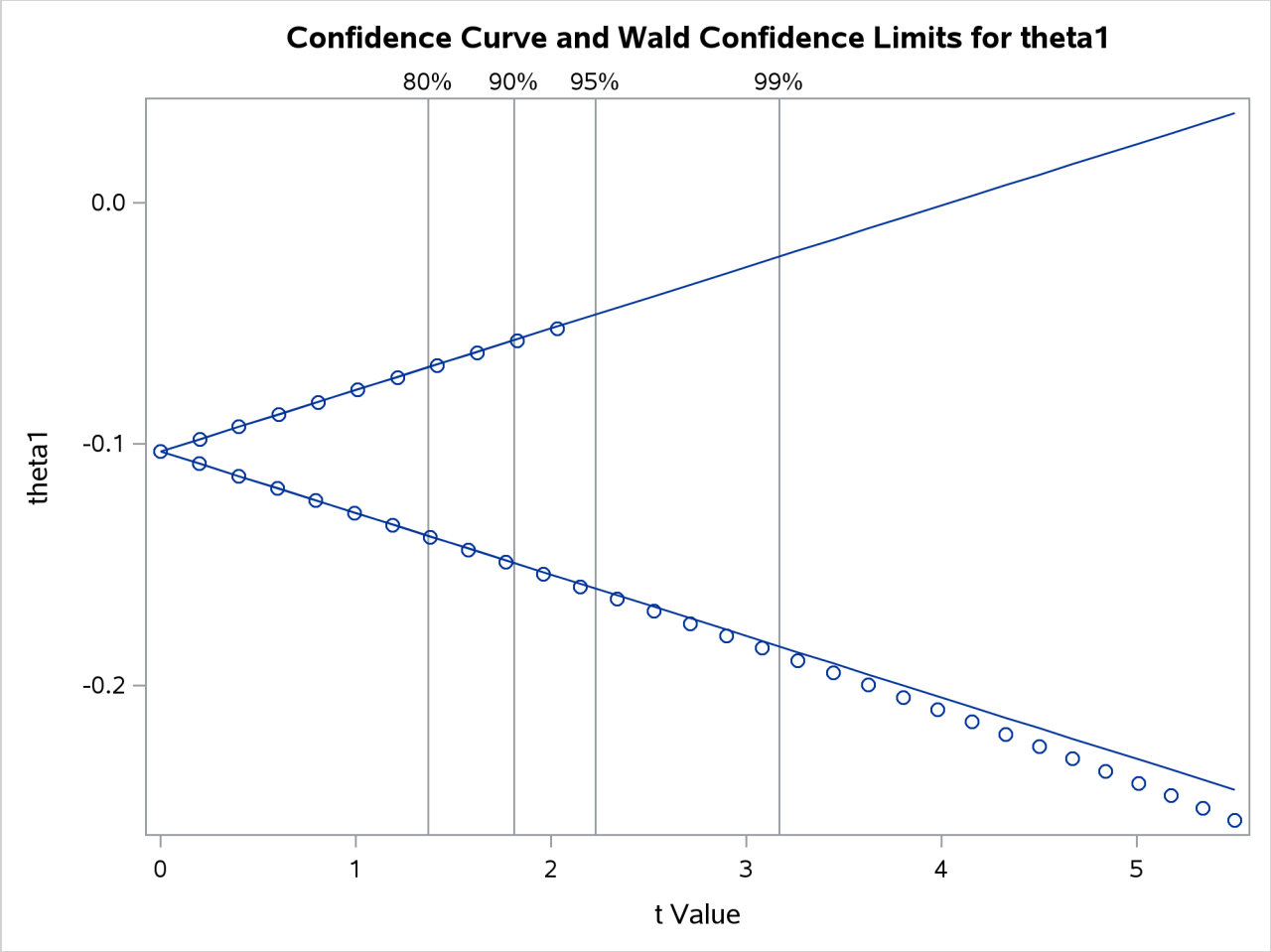

Output 88.7.5 Confidence Curve

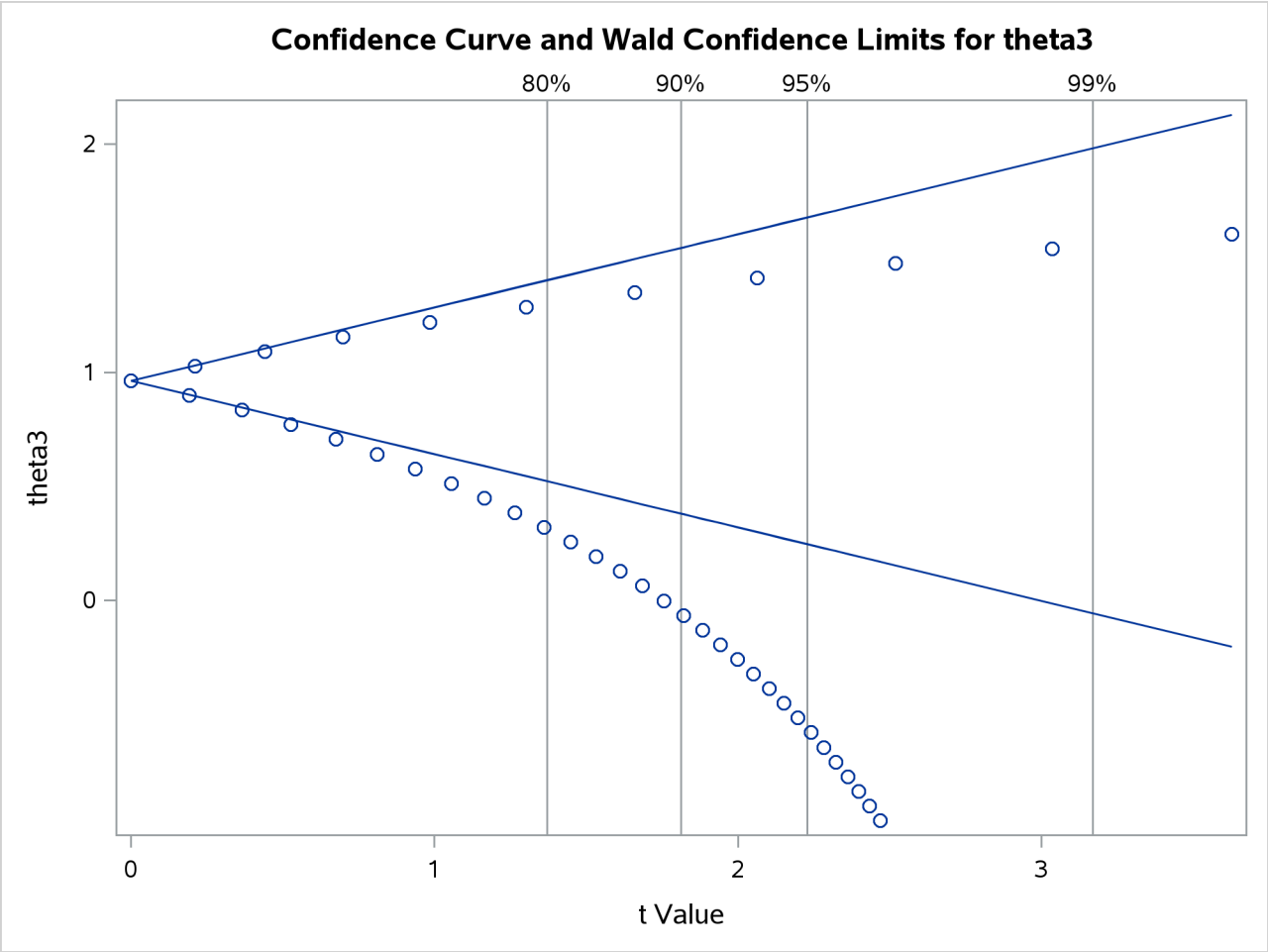

Output 88.7.6 Observation Influence on Parameter Estimate

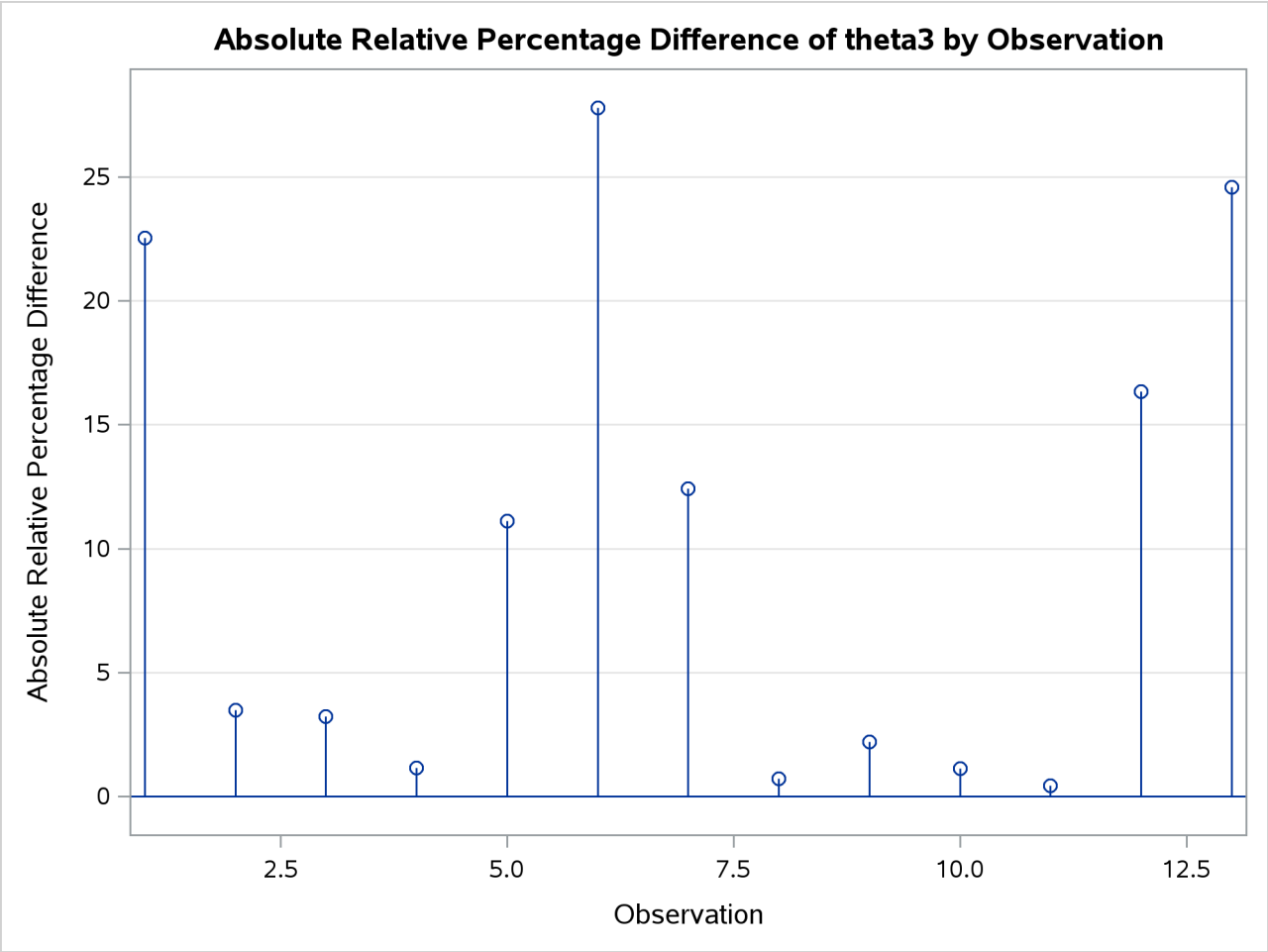

Output 88.7.7 Local Influence Plot

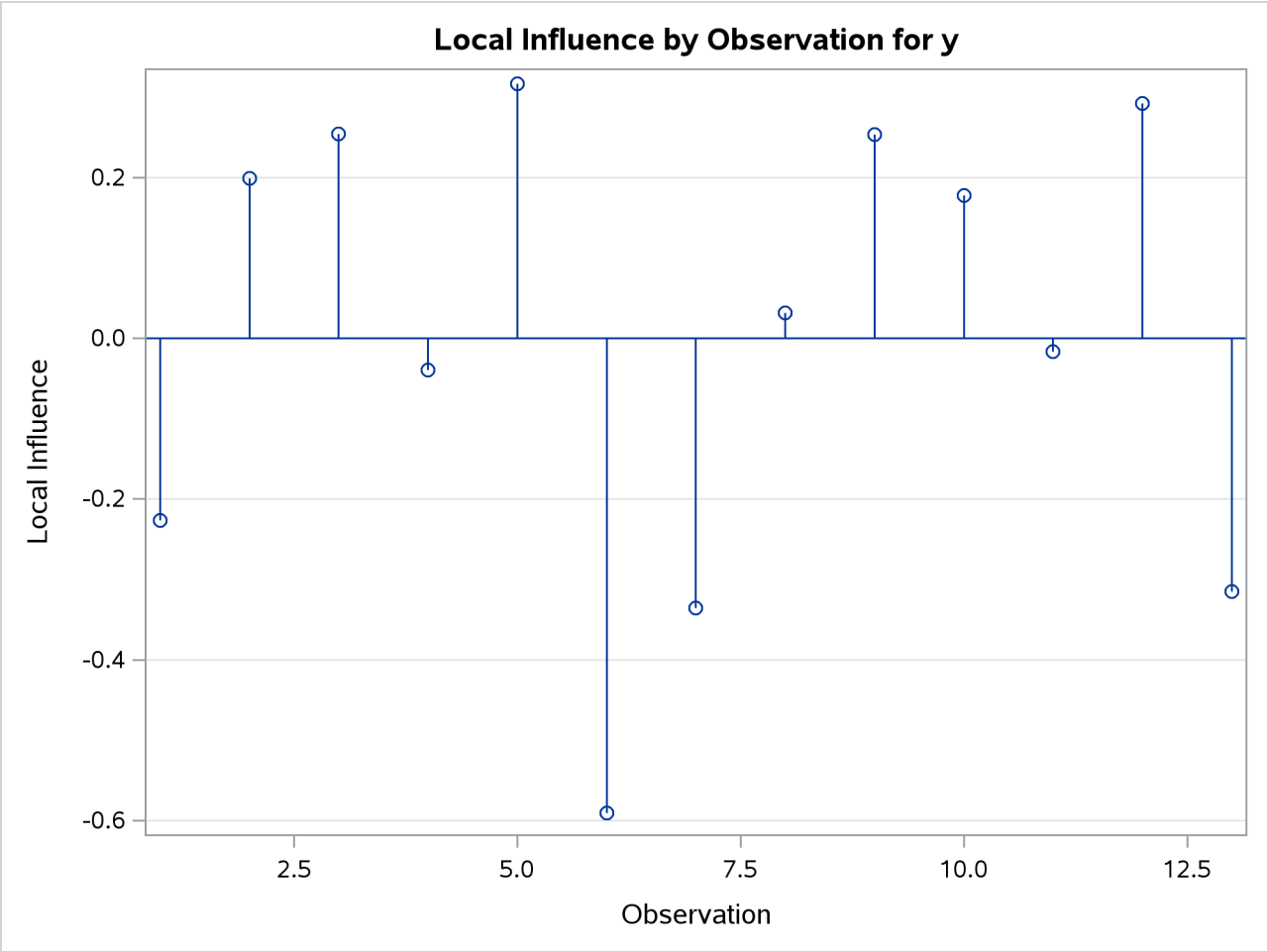

Output 88.7.8 Histogram of Bootstrap Parameter Estimates

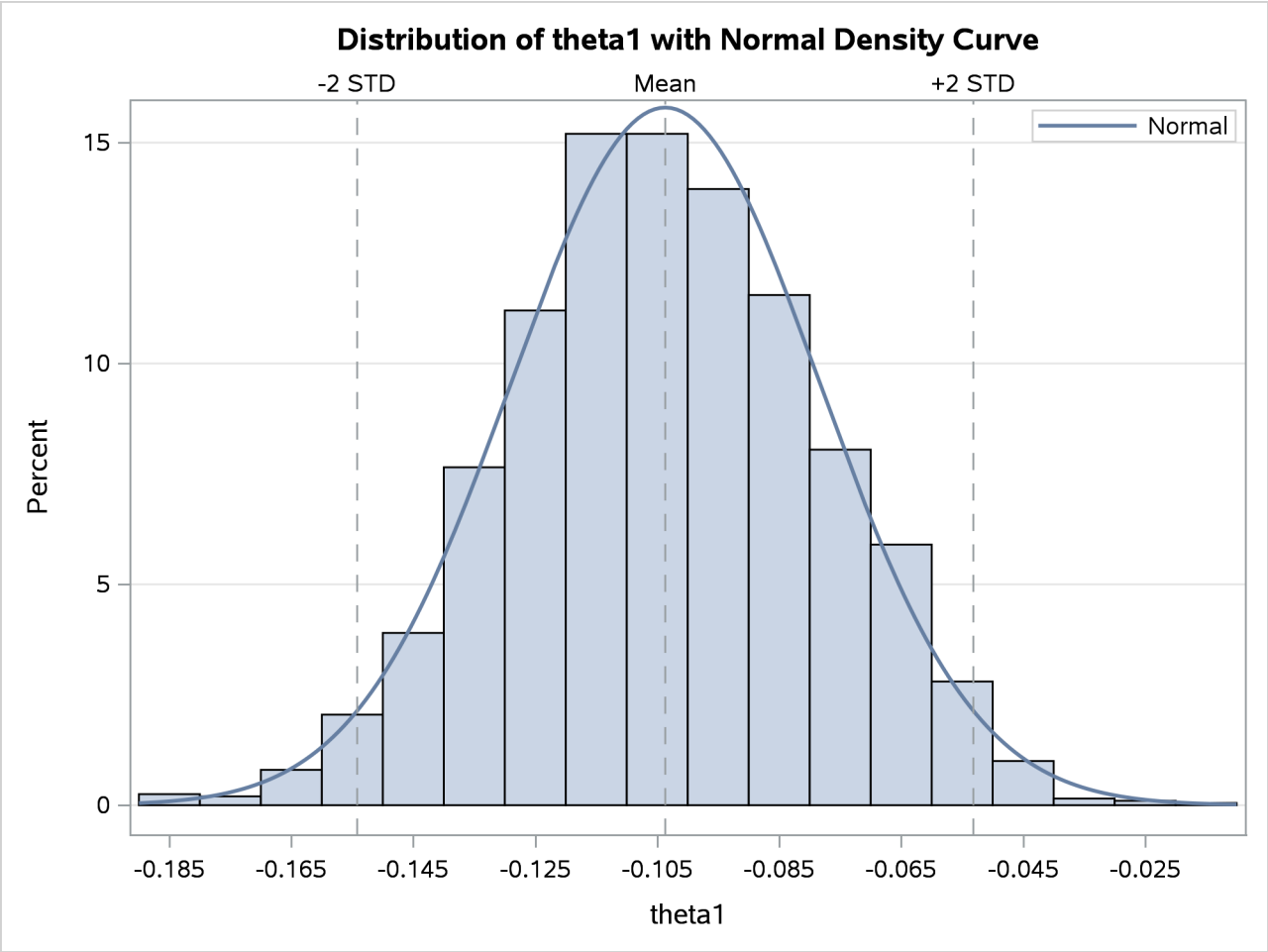

Output 88.7.9 Histogram of Bootstrap Parameter Estimates

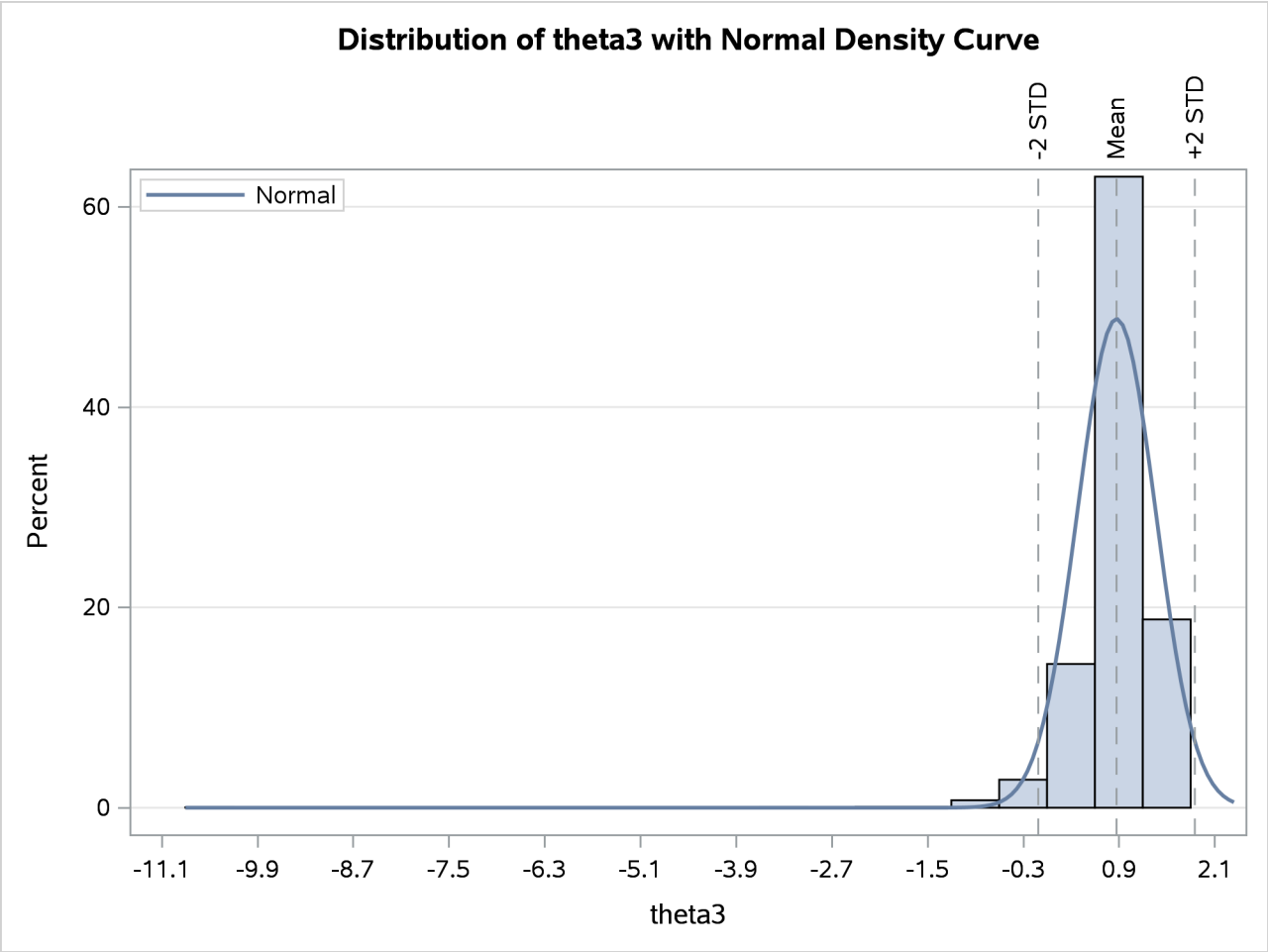

References

Bard, Y. (1970). "Comparison of Gradient Methods for the Solution of the Nonlinear Parameter Estimation Problem." *SIAM Journal on Numerical Analysis* 7:157–186.

Bard, Y. (1974). *Nonlinear Parameter Estimation*. New York: Academic Press.

Bates, D. M., and Watts, D. G. (1980). "Relative Curvature Measures of Nonlinearity." *Journal of the Royal Statistical Society, Series B* 42:1–25. With discussion.

Bates, D. M., and Watts, D. G. (1981). "A Relative Offset Orthogonality Convergence Criterion for Nonlinear Least Squares." *Technometrics* 23:179–183.

Bates, D. M., and Watts, D. G. (1988). *Nonlinear Regression Analysis and Its Applications*. New York: John Wiley & Sons.

Beaton, A. E., and Tukey, J. W. (1974). "The Fitting of Power Series, Meaning Polynomials, Illustrated on Band-Spectroscopic Data." *Technometrics* 16:147–185.

Billingsley, P. (1986). *Probability and Measure*. 2nd ed. New York: John Wiley & Sons.

Box, M. J. (1971). "Bias in Nonlinear Estimation." *Journal of the Royal Statistical Society, Series B* 33:171–201. With discussion.

Charnes, A., Frome, E. L., and Yu, P. L. (1976). "The Equivalence of Generalized Least Squares and Maximum Likelihood Estimation in the Exponential Family." *Journal of the American Statistical Association* 71:169–171.

Clarke, G. P. Y. (1987). "Approximate Confidence Limits for a Parameter Function in Nonlinear Regression." *Journal of the American Statistical Association* 82:221–230.

Cook, R. D., and Tsai, C.-L. (1985). "Residuals in Nonlinear Regression." *Biometrika* 72:23–29.

Cook, R. D., and Weisberg, S. (1990). "Confidence Curves in Nonlinear Regression." *Journal of the American Statistical Association* 85:544–551.

Cox, C. (1998). "Delta Method." In *Encyclopedia of Biostatistics*, edited by P. Armitage and T. Colton. New York: John Wiley & Sons.

Cox, D. R. (1970). *Analysis of Binary Data*. London: Metheun.

DiCiccio, T. J., and Efron, B. (1996). "Bootstrap Confidence Intervals." *Statistical Science* 11:189–212.

Donaldson, J. R., and Schnabel, R. B. (1987). "Computational Experience with Confidence Regions and Confidence Intervals for Nonlinear Least Squares." *Technometrics* 29:67–82.

Finney, D. J. (1971). *Probit Analysis*. 3rd ed. Cambridge: Cambridge University Press.

Gallant, A. R. (1975). "Nonlinear Regression." *American Statistician* 29:73–81.

Gill, P. E., Murray, W., and Wright, M. H. (1981). *Practical Optimization*. New York: Academic Press.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Hartley, H. O. (1961). "The Modified Gauss-Newton Method for the Fitting of Non-linear Regression Functions by Least Squares." *Technometrics* 3:269–280.

Holland, P. W., and Welsch, R. E. (1977). "Robust Regression Using Iteratively Reweighted Least-Squares." *Communications in Statistics—Theory and Methods* 6:813–827.

Hougaard, P. (1982). "Parameterizations of Nonlinear Models." *Journal of the Royal Statistical Society, Series B* 44:244–252.

Hougaard, P. (1985). "The Appropriateness of the Asymptotic Distribution in a Nonlinear Regression Model in Relation to Curvature." *Journal of the Royal Statistical Society, Series B* 47:103–114.

Huber, P. J. (1964). "Robust Estimation of a Location Parameter." *Annals of Mathematical Statistics* 35:73–101.

Huber, P. J. (1973). "Robust Regression: Asymptotics, Conjectures, and Monte Carlo." *Annals of Statistics* 1:799–821.

Jennrich, R. I. (1969). "Asymptotic Properties of Nonlinear Least Squares Estimators." *Annals of Mathematical Statistics* 40:633–643.

Jennrich, R. I., and Moore, R. H. (1975). "Maximum Likelihood Estimation by Means of Nonlinear Least Squares." In *Proceedings of the Statistical Computing Section*, 57–65. Washington, DC: American Statistical Association.

Jennrich, R. I., and Sampson, P. F. (1968). "Application of Stepwise Regression to Nonlinear Estimation." *Technometrics* 10:63–72.

Judge, G. G., Griffiths, W. E., Hill, R. C., and Lee, T.-C. (1980). *The Theory and Practice of Econometrics*. New York: John Wiley & Sons.

Kennedy, W. J., Jr., and Gentle, J. E. (1980). *Statistical Computing*. New York: Marcel Dekker.

Lee, E. T. (1974). "A Computer Program for Linear Logistic Regression Analysis." *Computer Programs in Biomedicine* 4:80–92.

MacKinnon, J. G. (2002). "Bootstrap Inference in Econometrics." *Canadian Journal of Economics* 35:615–645.

Marquardt, D. W. (1963). "An Algorithm for Least-Squares Estimation of Nonlinear Parameters." *Journal of the Society for Industrial and Applied Mathematics* 11:431–441.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Pinheiro, J. C., and Bates, D. M. (1995). "Approximations to the Log-Likelihood Function in the Nonlinear Mixed-Effects Model." *Journal of Computational and Graphical Statistics* 4:12–35.

Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.

Ratkowsky, D. (1983). *Nonlinear Regression Modeling*. New York: Marcel Dekker.

Ratkowsky, D. (1990). *Handbook of Nonlinear Regression Models*. New York: Marcel Dekker.

Schabenberger, O., and Pierce, F. J. (2002). *Contemporary Statistical Models for the Plant and Soil Sciences*. Boca Raton, FL: CRC Press.

Schabenberger, O., Tharp, B. E., Kells, J. J., and Penner, D. (1999). "Statistical Tests for Hormesis and Effective Dosages in Herbicide Dose Response." *Agronomy Journal* 91:713–721.

Seber, G. A. F., and Wild, C. J. (1989). *Nonlinear Regression*. New York: John Wiley & Sons.

St. Laurent, R. T., and Cook, R. D. (1992). "Leverages and Superleverages in Nonlinear Regression." *Journal of the American Statistical Association* 87:985–990.

St. Laurent, R. T., and Cook, R. D. (1993). "Leverages, Local Influence, and Curvature in Nonlinear Regression." *Biometrika* 80:99–106.

Wu, C. F. J. (1986). "Jackknife, Bootstrap and Other Resampling Methods in Regression Analysis." *Annals of Statistics* 80:1261–1295.

Chapter 89
The NLMIXED Procedure

Contents

Overview: NLMIXED Procedure . . . **7238**
- Introduction . . . 7238
- Literature on Nonlinear Mixed Models . . . 7239
- PROC NLMIXED Compared with Other SAS Procedures and Macros . . . 7239

Getting Started: NLMIXED Procedure . . . **7240**
- Nonlinear Growth Curves with Gaussian Data . . . 7240
- Logistic-Normal Model with Binomial Data . . . 7243

Syntax: NLMIXED Procedure . . . **7247**
- PROC NLMIXED Statement . . . 7247
- ARRAY Statement . . . 7265
- BOUNDS Statement . . . 7266
- BY Statement . . . 7266
- CMPTMODEL Statement . . . 7267
- CONTRAST Statement . . . 7270
- ESTIMATE Statement . . . 7270
- ID Statement . . . 7271
- MODEL Statement . . . 7271
- PARMS Statement . . . 7272
- PREDICT Statement . . . 7273
- RANDOM Statement . . . 7273
- REPLICATE Statement . . . 7275
- Programming Statements . . . 7275

Details: NLMIXED Procedure . . . **7277**
- Modeling Assumptions and Notation . . . 7277
- Integral Approximations . . . 7279
- Built-In Log-Likelihood Functions . . . 7281
- Compartment Models . . . 7283
- Hierarchical Model Specification . . . 7293
- Optimization Algorithms . . . 7295
- Finite-Difference Approximations of Derivatives . . . 7300
- Hessian Scaling . . . 7302
- Active Set Methods . . . 7302
- Line-Search Methods . . . 7304
- Restricting the Step Length . . . 7305
- Computational Problems . . . 7306
- Covariance Matrix . . . 7309

Prediction 7310
Computational Resources 7311
Displayed Output 7312
ODS Table Names 7314
Examples: NLMIXED Procedure **7315**
Example 89.1: One-Compartment Model with Pharmacokinetic Data 7315
Example 89.2: Probit-Normal Model with Binomial Data 7319
Example 89.3: Probit-Normal Model with Ordinal Data 7322
Example 89.4: Poisson-Normal Model with Count Data 7326
Example 89.5: Failure Time and Frailty Model 7329
Example 89.6: Simulated Nested Linear Random-Effects Model 7338
Example 89.7: Overdispersion Hierarchical Nonlinear Mixed Model 7342
References **7348**

Overview: NLMIXED Procedure

Introduction

The NLMIXED procedure fits nonlinear mixed models—that is, models in which both fixed and random effects enter nonlinearly. These models have a wide variety of applications, two of the most common being pharmacokinetics and overdispersed binomial data. PROC NLMIXED enables you to specify a conditional distribution for your data (given the random effects) having either a standard form (normal, binomial, Poisson) or a general distribution that you code using SAS programming statements.

PROC NLMIXED fits nonlinear mixed models by maximizing an approximation to the likelihood integrated over the random effects. Different integral approximations are available, the principal ones being adaptive Gaussian quadrature and a first-order Taylor series approximation. A variety of alternative optimization techniques are available to carry out the maximization; the default is a dual quasi-Newton algorithm.

Successful convergence of the optimization problem results in parameter estimates along with their approximate standard errors based on the second derivative matrix of the likelihood function. PROC NLMIXED enables you to use the estimated model to construct predictions of arbitrary functions by using empirical Bayes estimates of the random effects. You can also estimate arbitrary functions of the nonrandom parameters, and PROC NLMIXED computes their approximate standard errors by using the delta method.

Literature on Nonlinear Mixed Models

Davidian and Giltinan (1995) and Vonesh and Chinchilli (1997) provide good overviews as well as general theoretical developments and examples of nonlinear mixed models. Pinheiro and Bates (1995) is a primary reference for the theory and computational techniques of PROC NLMIXED. They describe and compare several different integrated likelihood approximations and provide evidence that adaptive Gaussian quadrature is one of the best methods. Davidian and Gallant (1993) also use Gaussian quadrature for nonlinear mixed models, although the smooth nonparametric density they advocate for the random effects is currently not available in PROC NLMIXED.

Traditional approaches to fitting nonlinear mixed models involve Taylor series expansions, expanding around either zero or the empirical best linear unbiased predictions of the random effects. The former is the basis for the well-known first-order method (Beal and Sheiner 1982, 1988; Sheiner and Beal 1985), and it is optionally available in PROC NLMIXED. The latter is the basis for the estimation method of Lindstrom and Bates (1990), and it is not available in PROC NLMIXED. However, the closely related Laplacian approximation is an option; it is equivalent to adaptive Gaussian quadrature with only one quadrature point. The Laplacian approximation and its relationship to the Lindstrom-Bates method are discussed by: Beal and Sheiner (1992); Wolfinger (1993); Vonesh (1992, 1996); Vonesh and Chinchilli (1997); Wolfinger and Lin (1997).

A parallel literature exists in the area of generalized linear mixed models, in which random effects appear as a part of the linear predictor inside a link function. Taylor-series methods similar to those just described are discussed in articles such as: Harville and Mee (1984); Stiratelli, Laird, and Ware (1984); Gilmour, Anderson, and Rae (1985); Goldstein (1991); Schall (1991); Engel and Keen (1992); Breslow and Clayton (1993); Wolfinger and O'Connell (1993); McGilchrist (1994), but such methods have not been implemented in PROC NLMIXED because they can produce biased results in certain binary data situations (Rodriguez and Goldman 1995; Lin and Breslow 1996). Instead, a numerical quadrature approach is available in PROC NLMIXED, as discussed in: Pierce and Sands (1975); Anderson and Aitkin (1985); Hedeker and Gibbons (1994); Crouch and Spiegelman (1990); Longford (1994); McCulloch (1994); Liu and Pierce (1994); Diggle, Liang, and Zeger (1994).

Nonlinear mixed models have important applications in pharmacokinetics, and Roe (1997) provides a wide-ranging comparison of many popular techniques. Yuh et al. (1994) provide an extensive bibliography on nonlinear mixed models and their use in pharmacokinetics.

PROC NLMIXED Compared with Other SAS Procedures and Macros

The models fit by PROC NLMIXED can be viewed as generalizations of the random coefficient models fit by the MIXED procedure. This generalization allows the random coefficients to enter the model nonlinearly, whereas in PROC MIXED they enter linearly. With PROC MIXED you can perform both maximum likelihood and restricted maximum likelihood (REML) estimation, whereas PROC NLMIXED implements only maximum likelihood. This is because the analog to the REML method in PROC NLMIXED would involve a high-dimensional integral over all of the fixed-effects parameters, and this integral is typically not available in closed form. Finally, PROC MIXED assumes the data to be normally distributed, whereas PROC NLMIXED enables you to analyze data that are normal, binomial, or Poisson or that have any likelihood programmable with SAS statements.

PROC NLMIXED does not implement the same estimation techniques available with the NLINMIX macro or the default estimation method of the GLIMMIX procedure. These are based on the estimation methods

of: Lindstrom and Bates (1990); Breslow and Clayton (1993); Wolfinger and O'Connell (1993), and they iteratively fit a set of generalized estimating equations (see Chapters 14 and 15 of Littell et al. 2006; Wolfinger 1997). In contrast, PROC NLMIXED directly maximizes an approximate integrated likelihood. This remark also applies to the SAS/IML macros MIXNLIN (Vonesh and Chinchilli 1997) and NLMEM (Galecki 1998).

The GLIMMIX procedure also fits mixed models for nonnormal data with nonlinearity in the conditional mean function. In contrast to the NLMIXED procedure, PROC GLIMMIX assumes that the model contains a linear predictor that links covariates to the conditional mean of the response. The NLMIXED procedure is designed to handle general conditional mean functions, whether they contain a linear component or not. As mentioned earlier, the GLIMMIX procedure by default estimates parameters in generalized linear mixed models by pseudo-likelihood techniques, whereas PROC NLMIXED by default performs maximum likelihood estimation by adaptive Gauss-Hermite quadrature. This estimation method is also available with the GLIMMIX procedure (METHOD=QUAD in the PROC GLIMMIX statement).

PROC NLMIXED has close ties with the NLP procedure in SAS/OR software. PROC NLMIXED uses a subset of the optimization code underlying PROC NLP and has many of the same optimization-based options. Also, the programming statement functionality used by PROC NLMIXED is the same as that used by PROC NLP and the MODEL procedure in SAS/ETS software.

Getting Started: NLMIXED Procedure

Nonlinear Growth Curves with Gaussian Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/nlmgs1.sas.)

As an introductory example, consider the orange tree data of Draper and Smith (1981). These data consist of seven measurements of the trunk circumference (in millimeters) on each of five orange trees. You can input these data into a SAS data set as follows:

```
data tree;
   input tree day y;
   datalines;
1  118   30
1  484   58
1  664   87

   ... more lines ...

5 1582  177
;
```

Lindstrom and Bates (1990) and Pinheiro and Bates (1995) propose the following logistic nonlinear mixed model for these data:

$$y_{ij} = \frac{b_1 + u_{i1}}{1 + \exp[-(d_{ij} - b_2)/b_3]} + e_{ij}$$

Here, y_{ij} represents the jth measurement on the ith tree ($i = 1, \ldots, 5$; $j = 1, \ldots, 7$), d_{ij} is the corresponding day, b_1, b_2, b_3 are the fixed-effects parameters, u_{i1} are the random-effect parameters assumed to be iid

$N(0, \sigma_u^2)$, and e_{ij} are the residual errors assumed to be iid $N(0, \sigma_e^2)$ and independent of the u_{i1}. This model has a logistic form, and the random-effect parameters u_{i1} enter the model linearly.

The statements to fit this nonlinear mixed model are as follows:

```
proc nlmixed data=tree;
   parms b1=190 b2=700 b3=350 s2u=1000 s2e=60;
   num = b1+u1;
   ex  = exp(-(day-b2)/b3);
   den = 1 + ex;
   model y ~ normal(num/den,s2e);
   random u1 ~ normal(0,s2u) subject=tree;
run;
```

The PROC NLMIXED statement invokes the procedure and inputs the tree data set. The PARMS statement identifies the unknown parameters and their starting values. Here there are three fixed-effects parameters (b1, b2, b3) and two variance components (s2u, s2e).

The next three statements are SAS programming statements specifying the logistic mixed model. A new variable u1 is included to identify the random effect. These statements are evaluated for every observation in the data set when the NLMIXED procedure computes the log likelihood function and its derivatives.

The MODEL statement defines the dependent variable and its conditional distribution given the random effects. Here a normal (Gaussian) conditional distribution is specified with mean num/den and variance s2e.

The RANDOM statement defines the single random effect to be u1, and specifies that it follow a normal distribution with mean 0 and variance s2u. The SUBJECT= argument in the RANDOM statement defines a variable indicating when the random effect obtains new realizations; in this case, it changes according to the values of the tree variable. PROC NLMIXED assumes that the input data set is clustered according to the levels of the tree variable; that is, all observations from the same tree occur sequentially in the input data set.

The output from this analysis is as follows.

Figure 89.1 Model Specifications

The NLMIXED Procedure

Specifications	
Data Set	WORK.TREE
Dependent Variable	y
Distribution for Dependent Variable	Normal
Random Effects	u1
Distribution for Random Effects	Normal
Subject Variable	tree
Optimization Technique	Dual Quasi-Newton
Integration Method	Adaptive Gaussian Quadrature

The "Specifications" table lists basic information about the nonlinear mixed model you have specified (Figure 89.1). Included are the input data set, the dependent and subject variables, the random effects, the relevant distributions, and the type of optimization. The "Dimensions" table lists various counts related to the model, including the number of observations, subjects, and parameters (Figure 89.2). These quantities are useful for checking that you have specified your data set and model correctly. Also listed is the number of quadrature points that PROC NLMIXED has selected based on the evaluation of the log likelihood at the

starting values of the parameters. Here, only one quadrature point is necessary because the random-effect parameters u_{i1} enter the model linearly. (The Gauss-Hermite quadrature with a single quadrature point results in the Laplace approximation of the log likelihood.)

Figure 89.2 Dimensions Table for Growth Curve Model

Dimensions	
Observations Used	35
Observations Not Used	0
Total Observations	35
Subjects	5
Max Obs per Subject	7
Parameters	5
Quadrature Points	1

Figure 89.3 Starting Values of Parameter Estimates and Negative Log Likelihood

Initial Parameters					
b1	b2	b3	s2u	s2e	Negative Log Likelihood
190	700	350	1000	60	132.491787

The "Parameters" table lists the parameters to be estimated, their starting values, and the negative log likelihood evaluated at the starting values (Figure 89.3).

Figure 89.4 Iteration History for Growth Curve Model

Iteration History					
Iteration	Calls	Negative Log Likelihood	Difference	Maximum Gradient	Slope
1	8	131.6867	0.805045	0.010269	-0.63300
2	12	131.6447	0.042082	0.014783	-0.01820
3	16	131.6141	0.030583	0.009809	-0.02796
4	20	131.5725	0.041555	0.001186	-0.01344
5	22	131.5719	0.000627	0.000200	-0.00121
6	25	131.5719	5.549E-6	0.000092	-7.68E-6
7	28	131.5719	1.096E-6	6.097E-6	-1.29E-6

NOTE: GCONV convergence criterion satisfied.

The "Iteration History" table records the history of the minimization of the negative log likelihood (Figure 89.4). For each iteration of the quasi-Newton optimization, values are listed for the number of function calls, the value of the negative log likelihood, the difference from the previous iteration, the absolute value of the largest gradient, and the slope of the search direction. The note at the bottom of the table indicates that the algorithm has converged successfully according to the GCONV convergence criterion, a standard criterion computed using a quadratic form in the gradient and the inverse Hessian.

The final maximized value of the log likelihood as well as the information criterion of Akaike (AIC), its

small sample bias corrected version (AICC), and the Bayesian information criterion (BIC) in the "smaller is better" form appear in the "Fit Statistics" table (Figure 89.5). These statistics can be used to compare different nonlinear mixed models.

Figure 89.5 Fit Statistics for Growth Curve Model

Fit Statistics	
-2 Log Likelihood	263.1
AIC (smaller is better)	273.1
AICC (smaller is better)	275.2
BIC (smaller is better)	271.2

Figure 89.6 Parameter Estimates at Convergence

Parameter Estimates								
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
b1	192.05	15.6473	4	12.27	0.0003	148.61	235.50	1.154E-6
b2	727.90	35.2474	4	20.65	<.0001	630.04	825.76	5.289E-6
b3	348.07	27.0793	4	12.85	0.0002	272.88	423.25	-6.1E-6
s2u	999.88	647.44	4	1.54	0.1974	-797.71	2797.46	-3.84E-6
s2e	61.5139	15.8832	4	3.87	0.0179	17.4150	105.61	2.892E-6

The maximum likelihood estimates of the five parameters and their approximate standard errors computed using the final Hessian matrix are displayed in the "Parameter Estimates" table (Figure 89.6). Approximate t-values and Wald-type confidence limits are also provided, with degrees of freedom equal to the number of subjects minus the number of random effects. You should interpret these statistics cautiously for variance parameters like s2u and s2e. The final column in the output shows the gradient vector at the optimization solution. Each element appears to be sufficiently small to indicate a stationary point.

Since the random-effect parameters u_{i1} enter the model linearly, you can obtain equivalent results by using the first-order method (specify METHOD=FIRO in the PROC NLMIXED statement).

Logistic-Normal Model with Binomial Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/nlmgs2.sas.)

This example analyzes the data from Beitler and Landis (1985), which represent results from a multi-center clinical trial investigating the effectiveness of two topical cream treatments (active drug, control) in curing an infection. For each of eight clinics, the number of trials and favorable cures are recorded for each treatment. The SAS data set is as follows.

```
data infection;
   input clinic t x n;
   datalines;
1 1 11 36
1 0 10 37
```

```
2 1 16 20
2 0 22 32
3 1 14 19
3 0  7 19
4 1  2 16
4 0  1 17
5 1  6 17
5 0  0 12
6 1  1 11
6 0  0 10
7 1  1  5
7 0  1  9
8 1  4  6
8 0  6  7
;
```

Suppose n_{ij} denotes the number of trials for the ith clinic and the jth treatment ($i = 1, \ldots, 8$; $j = 0, 1$), and x_{ij} denotes the corresponding number of favorable cures. Then a reasonable model for the preceding data is the following logistic model with random effects:

$$x_{ij}|u_i \sim \text{Binomial}(n_{ij}, p_{ij})$$

and

$$\eta_{ij} = \log\left(\frac{p_{ij}}{1 - p_{ij}}\right) = \beta_0 + \beta_1 t_j + u_i$$

The notation t_j indicates the jth treatment, and the u_i are assumed to be iid $N(0, \sigma_u^2)$.

The PROC NLMIXED statements to fit this model are as follows:

```
proc nlmixed data=infection;
   parms beta0=-1 beta1=1 s2u=2;
   eta    = beta0 + beta1*t + u;
   expeta = exp(eta);
   p      = expeta/(1+expeta);
   model x ~ binomial(n,p);
   random u ~ normal(0,s2u) subject=clinic;
   predict eta out=eta;
   estimate '1/beta1' 1/beta1;
run;
```

The PROC NLMIXED statement invokes the procedure, and the PARMS statement defines the parameters and their starting values. The next three statements define p_{ij}, and the MODEL statement defines the conditional distribution of x_{ij} to be binomial. The RANDOM statement defines u to be the random effect with subjects defined by the clinic variable.

The PREDICT statement constructs predictions for each observation in the input data set. For this example, predictions of η_{ij} and approximate standard errors of prediction are output to a data set named eta. These predictions include empirical Bayes estimates of the random effects u_i.

The ESTIMATE statement requests an estimate of the reciprocal of β_1.

The output for this model is as follows.

Figure 89.7 Model Information and Dimensions for Logistic-Normal Model

The NLMIXED Procedure

Specifications	
Data Set	WORK.INFECTION
Dependent Variable	x
Distribution for Dependent Variable	Binomial
Random Effects	u
Distribution for Random Effects	Normal
Subject Variable	clinic
Optimization Technique	Dual Quasi-Newton
Integration Method	Adaptive Gaussian Quadrature

Dimensions	
Observations Used	16
Observations Not Used	0
Total Observations	16
Subjects	8
Max Obs per Subject	2
Parameters	3
Quadrature Points	5

The "Specifications" table provides basic information about the nonlinear mixed model (Figure 89.7). For example, the distribution of the response variable, conditional on normally distributed random effects, is binomial. The "Dimensions" table provides counts of various variables. You should check this table to make sure the data set and model have been entered properly. PROC NLMIXED selects five quadrature points to achieve the default accuracy in the likelihood calculations.

Figure 89.8 Starting Values of Parameter Estimates

Initial Parameters			
beta0	**beta1**	**s2u**	**Negative Log Likelihood**
-1	1	2	37.5945925

The "Parameters" table lists the starting point of the optimization and the negative log likelihood at the starting values (Figure 89.8).

Figure 89.9 Iteration History and Fit Statistics for Logistic-Normal Model

Iteration History

Iteration	Calls	Negative Log Likelihood	Difference	Maximum Gradient	Slope
1	4	37.3622692	0.232323	2.88208	-19.3762
2	6	37.1460375	0.216232	0.92193	-0.82852
3	9	37.0300936	0.115944	0.31590	-0.59175
4	11	37.0223017	0.007792	0.019060	-0.01615
5	13	37.0222472	0.000054	0.001743	-0.00011
6	16	37.0222466	6.57E-7	0.000091	-1.28E-6
7	19	37.0222466	5.38E-10	2.078E-6	-1.1E-9

NOTE: GCONV convergence criterion satisfied.

Fit Statistics	
-2 Log Likelihood	74.0
AIC (smaller is better)	80.0
AICC (smaller is better)	82.0
BIC (smaller is better)	80.3

The "Iteration History" table indicates successful convergence in seven iterations (Figure 89.9). The "Fit Statistics" table lists some useful statistics based on the maximized value of the log likelihood.

Figure 89.10 Parameter Estimates for Logistic-Normal Model

Parameter Estimates

Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
beta0	-1.1974	0.5561	7	-2.15	0.0683	-2.5123	0.1175	-3.1E-7
beta1	0.7385	0.3004	7	2.46	0.0436	0.02806	1.4488	-2.08E-6
s2u	1.9591	1.1903	7	1.65	0.1438	-0.8555	4.7737	-2.48E-7

The "Parameter Estimates" table indicates marginal significance of the two fixed-effects parameters (Figure 89.10). The positive value of the estimate of β_1 indicates that the treatment significantly increases the chance of a favorable cure.

Figure 89.11 Table of Additional Estimates

Additional Estimates

Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
1/beta1	1.3542	0.5509	7	2.46	0.0436	0.05	0.05146	2.6569

The "Additional Estimates" table displays results from the ESTIMATE statement (Figure 89.11). The estimate of $1/\beta_1$ equals $1/0.7385 = 1.3542$ and its standard error equals $0.3004/0.7385^2 = 0.5509$ by the delta method (Billingsley 1986; Cox 1998). Note that this particular approximation produces a t-statistic identical to that for the estimate of β_1. Not shown is the eta data set, which contains the original 16 observations and predictions of the η_{ij}.

Syntax: NLMIXED Procedure

The following statements are available in the NLMIXED procedure:

PROC NLMIXED < *options* > ;
  ARRAY *array-specification* ;
  BOUNDS *boundary-constraints* ;
  BY *variables* ;
  CMPTMODEL *required-options conditionally-required-options* < *options* > ;
  CONTRAST '*label*' *expression* < , *expression* > < *option* > ;
  ESTIMATE '*label*' *expression* < *options* > ;
  ID *names* ;
  MODEL *model-specification* ;
  PARMS *parameters-and-starting-values* ;
  PREDICT *expression* **OUT=***SAS-data-set* < *options* > ;
  RANDOM *random-effects-specification* ;
  REPLICATE *variable* ;
  Programming statements ;

The following sections provide a detailed description of each of these statements.

PROC NLMIXED Statement

PROC NLMIXED < *options* > ;

The PROC NLMIXED statement invokes the NLMIXED procedure. Table 89.1 summarizes the options available in the PROC NLMIXED statement.

Table 89.1 PROC NLMIXED Statement Options

Option	Description
Basic Options	
DATA=	Specifies the input data set
METHOD=	Specifies the integration method
NOSORTSUB	Requests that the unique SUBJECT= variable values not be used
NTHREADS=	Specifies the number of threads to use
Displayed Output Specifications	
ALPHA=	Specifies α for confidence limits
CORR	Requests the correlation matrix
COV	Requests the covariance matrix
DF=	Specifies the degrees of freedom for p-values and confidence limits
ECORR	Requests the correlation matrix of additional estimates
ECOV	Requests the covariance matrix of additional estimates
EDER	Requests derivatives of additional estimates
EMPIRICAL	Requests the empirical (“sandwich”) estimator of covariance matrix
HESS	Requests the Hessian matrix

Table 89.1 *continued*

Option	Description
ITDETAILS	Requests iteration details
START	Specifies the gradient at starting values
Debugging Output	
FLOW	Displays the model execution messages
LISTCODE	Displays compiled model program
LISTDEP	Produces a model dependency listing
LISTDER	Displays the model derivatives
LIST	Displays the model program and variables
TRACE	Displays detailed model execution messages
XREF	Displays the model cross references
Quadrature Options	
NOADSCALE	Requests no adaptive scaling
NOAD	Requests no adaptive centering
OUTQ=	Displays output data set
QFAC=	Specifies the search factor
QMAX=	Specifies the maximum points
QPOINTS=	Specifies the number of points
QSCALEFAC=	Specifies the scale factor
QTOL=	Specifies the tolerance
Empirical Bayes Options	
EBOPT	Requests comprehensive optimization
EBSSFRAC=	Specifies the step-shortening fraction
EBSSTOL=	Specifies the step-shortening tolerance
EBSTEPS=	Specifies the number of Newton steps
EBSUBSTEPS=	Specifies the number of substeps
EBTOL=	Specifies the convergence tolerance
EBZSTART	Requests zero as the starting values
OUTR=	Displays an output data set that contains empirical Bayes estimates of random effects and their approximate standard errors
Optimization Specifications	
HESCAL=	Specifies the type of Hessian scaling
INHESSIAN<=>	Specifies the start for approximated Hessian
LINESEARCH=	Specifies the line-search method
LSPRECISION=	Specifies the line-search precision
OPTCHECK<=>	Checks optimality in a neighborhood
RESTART=	Specifies the iteration number for update restart
TECHNIQUE=	Specifies the minimization technique
UPDATE=	Specifies the update technique
Derivatives Specifications	
DIAHES	Uses only the diagonal of Hessian
FDHESSIAN<=>	Specifies the finite-difference second derivatives

Table 89.1 *continued*

Option	Description
FD<=>	Specifies the finite-difference derivatives
Constraint Specifications	
LCDEACT=	Specifies the Lagrange multiplier tolerance for deactivating
LCEPSILON=	Specifies the range for active constraints
LCSINGULAR=	Specifies the tolerance for dependent constraints
Termination Criteria Specifications	
ABSCONV=	Specifies the absolute function convergence criterion
ABSFCONV=	Specifies the absolute function difference convergence criterion
ABSGCONV=	Specifies the absolute gradient convergence criterion
ABSXCONV=	Specifies the absolute parameter convergence criterion
FCONV=	Specifies the relative function convergence criterion
FCONV2=	Specifies another relative function convergence criterion
FDIGITS=	Specifies the number accurate digits in objective function
FSIZE=	Specifies the FSIZE parameter of the relative function and relative gradient termination criteria
GCONV=	Specifies the relative gradient convergence criterion
MAXFUNC=	Specifies the maximum number of function calls
MAXITER=	Specifies the maximum number of iterations
MAXTIME=	Specifies the upper limit seconds of CPU time
MINITER=	Specifies the minimum number of iterations
XCONV=	Specifies the relative parameter convergence criterion
XSIZE=	Used in XCONV criterion
Step Length Specifications	
DAMPSTEP<=>	Specifies the damped steps in line search
INSTEP=	Specifies the initial trust-region radius
MAXSTEP=	Specifies the maximum trust-region radius
Singularity Tolerances	
SINGCHOL=	Specifies the tolerance for Cholesky roots
SINGHESS=	Specifies the tolerance for Hessian
SINGSWEEP=	Specifies the tolerance for sweep
SINGVAR=	Specifies the tolerance for variances
Covariance Matrix Tolerances	
ASINGULAR=	Specifies the absolute singularity for inertia
CFACTOR=	Specifies the multiplication factor for COV matrix
COVSING=	Specifies the tolerance for singular COV matrix
G4=	Specifies the threshold for Moore-Penrose inverse
MSINGULAR=	Specifies the relative M singularity for inertia
VSINGULAR=	Specifies the relative V singularity for inertia

These options are described in alphabetical order. For a description of the mathematical notation used in the following sections, see the section "Modeling Assumptions and Notation" on page 7277.

ABSCONV=*r*
ABSTOL=*r*

specifies an absolute function convergence criterion. For minimization, termination requires $f(\boldsymbol{\theta}^{(k)}) \leq r$. The default value of *r* is the negative square root of the largest double-precision value, which serves only as a protection against overflows.

ABSFCONV=*r***<[***n***]>**
ABSFTOL=*r***<[***n***]>**

specifies an absolute function difference convergence criterion. For all techniques except NMSIMP, termination requires a small change of the function value in successive iterations:

$$|f(\boldsymbol{\theta}^{(k-1)}) - f(\boldsymbol{\theta}^{(k)})| \leq r$$

The same formula is used for the NMSIMP technique, but $\boldsymbol{\theta}^{(k)}$ is defined as the vertex with the lowest function value, and $\boldsymbol{\theta}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default value is $r = 0$. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

ABSGCONV=*r***<[***n***]>**
ABSGTOL=*r***<[***n***]>**

specifies an absolute gradient convergence criterion. Termination requires the maximum absolute gradient element to be small:

$$\max_j |g_j(\boldsymbol{\theta}^{(k)})| \leq r$$

This criterion is not used by the NMSIMP technique. The default value is r = 1E–5. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated. If you specify more than one RANDOM statement, the default value is r = 1E–3.

ABSXCONV=*r***<[***n***]>**
ABSXTOL=*r***<[***n***]>**

specifies an absolute parameter convergence criterion. For all techniques except NMSIMP, termination requires a small Euclidean distance between successive parameter vectors,

$$\| \boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k-1)} \|_2 \leq r$$

For the NMSIMP technique, termination requires either a small length $\alpha^{(k)}$ of the vertices of a restart simplex,

$$\alpha^{(k)} \leq r$$

or a small simplex size,

$$\delta^{(k)} \leq r$$

where the simplex size $\delta^{(k)}$ is defined as the L1 distance from the simplex vertex $\boldsymbol{\xi}^{(k)}$ with the smallest function value to the other *n* simplex points $\boldsymbol{\theta}_l^{(k)} \neq \boldsymbol{\xi}^{(k)}$:

$$\delta^{(k)} = \sum_{\boldsymbol{\theta}_l \neq y} \| \boldsymbol{\theta}_l^{(k)} - \boldsymbol{\xi}^{(k)} \|_1$$

The default is r = 1E–8 for the NMSIMP technique and $r = 0$ otherwise. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

ALPHA=α

specifies the alpha level to be used in computing confidence limits. The default value is 0.05.

ASINGULAR=*r*

ASING=*r*

specifies an absolute singularity criterion for the computation of the inertia (number of positive, negative, and zero eigenvalues) of the Hessian and its projected forms. The default value is the square root of the smallest positive double-precision value.

CFACTOR=*f*

specifies a multiplication factor *f* for the estimated covariance matrix of the parameter estimates.

CORR

requests the approximate correlation matrix for the parameter estimates.

COV

requests the approximate covariance matrix for the parameter estimates.

COVSING=*r*> 0

specifies a nonnegative threshold that determines whether the eigenvalues of a singular Hessian matrix are considered to be zero.

DAMPSTEP<=*r***>**

DS<=*r***>**

specifies that the initial step-size value $\alpha^{(0)}$ for each line search (used by the QUANEW, CONGRA, or NEWRAP technique) cannot be larger than *r* times the step-size value used in the former iteration. If you specify the DAMPSTEP option without factor *r*, the default value is $r = 2$. The DAMPSTEP=*r* option can prevent the line-search algorithm from repeatedly stepping into regions where some objective functions are difficult to compute or where they could lead to floating-point overflows during the computation of objective functions and their derivatives. The DAMPSTEP=*r* option can save time-costly function calls that result in very small step sizes α. For more details on setting the start values of each line search, see the section "Restricting the Step Length" on page 7305.

DATA=*SAS-data-set*

specifies the input data set. Observations in this data set are used to compute the log likelihood function that you specify with PROC NLMIXED statements.

DF=*d*

specifies the degrees of freedom to be used in computing *p* values and confidence limits. PROC NLMIXED calculates the default degrees of freedom as follows:

- When there is no RANDOM statement in the model, the default value is the number of observations.
- When only one RANDOM statement is specified, the default value is the number of subjects minus the number of random effects for random-effects models.
- When multiple RANDOM statements are specified, the default degrees of freedom is the number of subjects in the lowest nested level minus the total number of random effects. For example, if the highest level of hierarchy is specified by SUBJECT=S1 and the next level of hierarchy (nested within S1) is specified by SUBJECT=S2(S1), then the degrees of freedom is computed as the total number of subjects from S2(S1) minus the total number of random-effects variables in the model.

If the degrees of freedom computation leads to a nonpositive value, then the default value is the total number of observations.

DIAHES
specifies that only the diagonal of the Hessian be used.

EBOPT
requests that a more comprehensive optimization be carried out if the default empirical Bayes optimization fails to converge. If you specify more than one RANDOM statement, this option is ignored.

EBSSFRAC=*r* > 0
specifies the step-shortening fraction to be used while computing empirical Bayes estimates of the random effects. The default value is 0.8. If you specify more than one RANDOM statement, this option is ignored.

EBSSTOL=*r* ≥ 0
specifies the objective function tolerance for determining the cessation of step-shortening while computing empirical Bayes estimates of the random effects. The default value is r = 1E–8. If you specify more than one RANDOM statement, this option is ignored.

EBSTEPS=*n* ≥ 0
specifies the maximum number of Newton steps for computing empirical Bayes estimates of random effects. The default value is $n = 50$. If you specify more than one RANDOM statement, this option is ignored.

EBSUBSTEPS=*n* ≥ 0
specifies the maximum number of step-shortenings for computing empirical Bayes estimates of random effects. The default value is $n = 20$. If you specify more than one RANDOM statement, this option is ignored.

EBTOL=*r* ≥ 0
specifies the convergence tolerance for empirical Bayes estimation. The default value is $r = \epsilon\mathrm{E}4$, where ϵ is the machine precision. This default value equals approximately 1E–12 on most machines. If you specify more than one RANDOM statement, this option is ignored.

EBZSTART
requests that a zero be used as starting values during empirical Bayes estimation. By default, the starting values are set equal to the estimates from the previous iteration (or zero for the first iteration).

ECOV
requests the approximate covariance matrix for all expressions specified in ESTIMATE statements.

ECORR
requests the approximate correlation matrix for all expressions specified in ESTIMATE statements.

EDER
requests the derivatives of all expressions specified in ESTIMATE statements with respect to each of the model parameters.

EMPIRICAL

requests that the covariance matrix of the parameter estimates be computed as a likelihood-based empirical ("sandwich") estimator (White 1982). If $f(\boldsymbol{\theta}) = -\log\{m(\boldsymbol{\theta})\}$ is the objective function for the optimization and $m(\boldsymbol{\theta})$ denotes the marginal log likelihood (see the section "Modeling Assumptions and Notation" on page 7277 for notation and further definitions) the empirical estimator is computed as

$$\mathbf{H}(\hat{\boldsymbol{\theta}})^{-1}\left(\sum_{i=1}^{s}\mathbf{g}_i(\hat{\boldsymbol{\theta}})\mathbf{g}_i(\hat{\boldsymbol{\theta}})'\right)\mathbf{H}(\hat{\boldsymbol{\theta}})^{-1}$$

where $\mathbf{H}$ is the second derivative matrix of f and $\mathbf{g}_i$ is the first derivative of the contribution to f by the ith subject. If you choose the EMPIRICAL option, this estimator of the covariance matrix of the parameter estimates replaces the model-based estimator $\mathbf{H}(\hat{\boldsymbol{\theta}})^{-1}$ in subsequent calculations. You can output the subject-specific gradients $\mathbf{g}_i$ to a SAS data set with the SUBGRADIENT option in the PROC NLMIXED statement.

The EMPIRICAL option requires the presence of a RANDOM statement and is available for METHOD=GAUSS and METHOD=ISAMP only.

If you specify more than one RANDOM statement, this option is ignored.

FCONV=*r*< [*n*] >

FTOL=*r*< [*n*] >

specifies a relative function convergence criterion. For all techniques except NMSIMP, termination requires a small relative change of the function value in successive iterations,

$$\frac{|f(\boldsymbol{\theta}^{(k)}) - f(\boldsymbol{\theta}^{(k-1)})|}{\max(|f(\boldsymbol{\theta}^{(k-1)})|, \text{FSIZE})} \le r$$

where FSIZE is defined by the FSIZE= option. The same formula is used for the NMSIMP technique, but $\boldsymbol{\theta}^{(k)}$ is defined as the vertex with the lowest function value, and $\boldsymbol{\theta}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex. The default is $r = 10^{-\text{FDIGITS}}$, where FDIGITS is the value of the FDIGITS= option. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

FCONV2=*r*< [*n*] >

FTOL2=*r*< [*n*] >

specifies another function convergence criterion. For all techniques except NMSIMP, termination requires a small predicted reduction

$$df^{(k)} \approx f(\boldsymbol{\theta}^{(k)}) - f(\boldsymbol{\theta}^{(k)} + \mathbf{s}^{(k)})$$

of the objective function. The predicted reduction

$$\begin{aligned} df^{(k)} &= -\mathbf{g}^{(k)\prime}\mathbf{s}^{(k)} - \frac{1}{2}\mathbf{s}^{(k)\prime}\mathbf{H}^{(k)}\mathbf{s}^{(k)} \\ &= -\frac{1}{2}\mathbf{s}^{(k)\prime}\mathbf{g}^{(k)} \\ &\le r \end{aligned}$$

is computed by approximating the objective function f by the first two terms of the Taylor series and substituting the Newton step:

$$\mathbf{s}^{(k)} = -[\mathbf{H}^{(k)}]^{-1}\mathbf{g}^{(k)}$$

For the NMSIMP technique, termination requires a small standard deviation of the function values of the $n+1$ simplex vertices $\boldsymbol{\theta}_l^{(k)}$, $l = 0, \ldots, n$,

$$\sqrt{\frac{1}{n+1}\sum_l \left[f(\boldsymbol{\theta}_l^{(k)}) - \overline{f}(\boldsymbol{\theta}^{(k)})\right]^2} \le r$$

where $\overline{f}(\boldsymbol{\theta}^{(k)}) = \frac{1}{n+1}\sum_l f(\boldsymbol{\theta}_l^{(k)})$. If there are n_{act} boundary constraints active at $\boldsymbol{\theta}^{(k)}$, the mean and standard deviation are computed only for the $n+1-n_{act}$ unconstrained vertices. The default value is r = 1E–6 for the NMSIMP technique and r = 0 otherwise. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate.

FD < = FORWARD | CENTRAL | *r* **>**

specifies that all derivatives be computed using finite difference approximations. The following specifications are permitted:

FD is equivalent to FD=100.

FD=CENTRAL uses central differences.

FD=FORWARD uses forward differences.

FD=*r* uses central differences for the initial and final evaluations of the gradient and for the Hessian. During iteration, start with forward differences and switch to a corresponding central-difference formula during the iteration process when one of the following two criteria is satisfied:

- The absolute maximum gradient element is less than or equal to r times the ABSGCONV= threshold.
- The normalized predicted function reduction (see the GTOL option) is less than or equal to $\max(1\mathrm{E}-6,\ r \times \mathrm{GTOL})$. The 1E–6 ensures that the switch is done, even if you set the GTOL threshold to zero.

Note that the FD and FDHESSIAN options cannot apply at the same time. The FDHESSIAN option is ignored when only first-order derivatives are used. See the section "Finite-Difference Approximations of Derivatives" on page 7300 for more information.

FDHESSIAN< =FORWARD | CENTRAL >

FDHES< =FORWARD | CENTRAL >

FDH< =FORWARD | CENTRAL >

specifies that second-order derivatives be computed using finite difference approximations based on evaluations of the gradients.

FDHESSIAN=FORWARD uses forward differences.

FDHESSIAN=CENTRAL uses central differences.

FDHESSIAN uses forward differences for the Hessian except for the initial and final output.

Note that the FD and FDHESSIAN options cannot apply at the same time. See the section "Finite-Difference Approximations of Derivatives" on page 7300 for more information.

FDIGITS=*r*

specifies the number of accurate digits in evaluations of the objective function. Fractional values such as FDIGITS=4.7 are allowed. The default value is $r = -\log_{10}\epsilon$, where ϵ is the machine precision. The value of *r* is used to compute the interval size *h* for the computation of finite-difference approximations of the derivatives of the objective function and for the default value of the FCONV= option. If you specify more than one RANDOM statement, the default value is $r = 0.5(-\log_{10}\epsilon)$.

FLOW

displays a message for each statement in the model program as it is executed. This debugging option is very rarely needed and produces voluminous output.

FSIZE=*r*

specifies the FSIZE parameter of the relative function and relative gradient termination criteria. The default value is *r* = 0. For more information, see the FCONV= and GCONV= options.

G4=*n* > 0

specifies a dimension to determine the type of generalized inverse to use when the approximate covariance matrix of the parameter estimates is singular. The default value of *n* is 60. See the section "Covariance Matrix" on page 7309 for more information.

GCONV=*r***< [***n***] >**

GTOL=*r***< [***n***] >**

specifies a relative gradient convergence criterion. For all techniques except CONGRA and NMSIMP, termination requires that the normalized predicted function reduction is small,

$$\frac{\mathbf{g}(\boldsymbol{\theta}^{(k)})'[\mathbf{H}^{(k)}]^{-1}\mathbf{g}(\boldsymbol{\theta}^{(k)})}{\max(|f(\boldsymbol{\theta}^{(k)})|, \mathrm{FSIZE})} \leq r$$

where FSIZE is defined by the FSIZE= option. For the CONGRA technique (where a reliable Hessian estimate *H* is not available), the following criterion is used:

$$\frac{\| \mathbf{g}(\boldsymbol{\theta}^{(k)}) \|_2^2 \quad \| \mathbf{s}(\boldsymbol{\theta}^{(k)}) \|_2}{\| \mathbf{g}(\boldsymbol{\theta}^{(k)}) - \mathbf{g}(\boldsymbol{\theta}^{(k-1)}) \|_2 \max(|f(\boldsymbol{\theta}^{(k)})|, \mathrm{FSIZE})} \leq r$$

This criterion is not used by the NMSIMP technique.

The default value is *r* = 1E–8. The optional integer value *n* specifies the number of successive iterations for which the criterion must be satisfied before the process can terminate. If you specify more than one RANDOM statement, the default value is *r* = 1E–6.

HESCAL= 0 | 1 | 2 | 3

HS=0 | 1 | 2 | 3

specifies the scaling version of the Hessian matrix used in NRRIDG, TRUREG, NEWRAP, or DBLDOG optimization.

If HS is not equal to 0, the first iteration and each restart iteration sets the diagonal scaling matrix $\mathbf{D}^{(0)} = \mathrm{diag}(d_i^{(0)})$:

$$d_i^{(0)} = \sqrt{\max(|H_{i,i}^{(0)}|, \epsilon)}$$

where $H_{i,i}^{(0)}$ are the diagonal elements of the Hessian. In every other iteration, the diagonal scaling matrix $\mathbf{D}^{(0)} = \mathrm{diag}(d_i^{(0)})$ is updated depending on the HS option:

0 specifies that no scaling is done.

1 specifies the Moré (1978) scaling update:

$$d_i^{(k+1)} = \max\left[d_i^{(k)}, \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}\right]$$

2 specifies the Dennis, Gay, and Welsch (1981) scaling update:

$$d_i^{(k+1)} = \max\left[0.6 * d_i^{(k)}, \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}\right]$$

3 specifies that d_i is reset in each iteration:

$$d_i^{(k+1)} = \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}$$

In each scaling update, ϵ is the relative machine precision. The default value is HS=0. Scaling of the Hessian can be time-consuming in the case where general linear constraints are active.

HESS

requests the display of the final Hessian matrix after optimization. If you also specify the START option, then the Hessian at the starting values is also printed.

INHESSIAN< =*r* >

INHESS< =*r* >

specifies how the initial estimate of the approximate Hessian is defined for the quasi-Newton techniques QUANEW and DBLDOG. There are two alternatives:

- If you do not use the *r* specification, the initial estimate of the approximate Hessian is set to the Hessian at $\boldsymbol{\theta}^{(0)}$.
- If you do use the *r* specification, the initial estimate of the approximate Hessian is set to the multiple of the identity matrix, $r\mathbf{I}$.

By default, if you do not specify the option INHESSIAN=*r*, the initial estimate of the approximate Hessian is set to the multiple of the identity matrix $r\mathbf{I}$, where the scalar r is computed from the magnitude of the initial gradient.

INSTEP=*r*

reduces the length of the first trial step during the line search of the first iterations. For highly nonlinear objective functions, such as the EXP function, the default initial radius of the trust-region algorithm TRUREG or DBLDOG or the default step length of the line-search algorithms can result in arithmetic overflows. If this occurs, you should specify decreasing values of $0 < r < 1$ such as INSTEP=1E–1, INSTEP=1E–2, INSTEP=1E–4, and so on, until the iteration starts successfully.

- For trust-region algorithms (TRUREG, DBLDOG), the INSTEP= option specifies a factor $r > 0$ for the initial radius $\Delta^{(0)}$ of the trust region. The default initial trust-region radius is the length of the scaled gradient. This step corresponds to the default radius factor of $r = 1$.
- For line-search algorithms (NEWRAP, CONGRA, QUANEW), the INSTEP= option specifies an upper bound for the initial step length for the line search during the first five iterations. The default initial step length is $r = 1$.
- For the Nelder-Mead simplex algorithm, using TECH=NMSIMP, the INSTEP=*r* option defines the size of the start simplex.

For more details, see the section "Computational Problems" on page 7306.

ITDETAILS

requests a more complete iteration history, including the current values of the parameter estimates, their gradients, and additional optimization statistics. For further details, see the section "Iterations" on page 7312.

LCDEACT=*r*

LCD=*r*

specifies a threshold r for the Lagrange multiplier that determines whether an active inequality constraint remains active or can be deactivated. During minimization, an active inequality constraint can be deactivated only if its Lagrange multiplier is less than the threshold value $r < 0$. The default value is

$$r = -\min(0.01, \max(0.1 \times \mathrm{ABSGCONV}, 0.001 \times \mathrm{gmax}^{(k)}))$$

where ABSGCONV is the value of the absolute gradient criterion, and $\mathrm{gmax}^{(k)}$ is the maximum absolute element of the (projected) gradient $\mathbf{g}^{(k)}$ or $\mathbf{Z}'\mathbf{g}^{(k)}$. (See the section "Active Set Methods" for a definition of $\mathbf{Z}$.)

LCEPSILON=*r* > 0

LCEPS=*r* > 0

LCE=*r* > 0

specifies the range for active and violated boundary constraints. The default value is r = 1E–8. During the optimization process, the introduction of rounding errors can force PROC NLMIXED to increase the value of r by a factor of $10, 100, \ldots$. If this happens, it is indicated by a message displayed in the log.

LCSINGULAR=*r* > 0

LCSING=*r* > 0

LCS=*r* > 0

specifies a criterion r, used in the update of the QR decomposition, that determines whether an active constraint is linearly dependent on a set of other active constraints. The default value is r = 1E–8. The larger r becomes, the more the active constraints are recognized as being linearly dependent. If the value of r is larger than 0.1, it is reset to 0.1.

LINESEARCH=*i*

LIS=*i*

specifies the line-search method for the CONGRA, QUANEW, and NEWRAP optimization techniques. See Fletcher (1987) for an introduction to line-search techniques. The value of i can be $1, \ldots, 8$. For CONGRA, QUANEW and NEWRAP, the default value is i = 2.

1 specifies a line-search method that needs the same number of function and gradient calls for cubic interpolation and cubic extrapolation; this method is similar to one used by the Harwell subroutine library.

2 specifies a line-search method that needs more function than gradient calls for quadratic and cubic interpolation and cubic extrapolation; this method is implemented as shown in Fletcher (1987) and can be modified to an exact line search by using the LSPRECISION= option.

3 specifies a line-search method that needs the same number of function and gradient calls for cubic interpolation and cubic extrapolation; this method is implemented as shown in Fletcher (1987) and can be modified to an exact line search by using the LSPRECISION= option.

4 specifies a line-search method that needs the same number of function and gradient calls for stepwise extrapolation and cubic interpolation.

5 specifies a line-search method that is a modified version of LIS=4.

6 specifies golden section line search (Polak 1971), which uses only function values for linear approximation.

7 specifies bisection line search (Polak 1971), which uses only function values for linear approximation.

8 specifies the Armijo line-search technique (Polak 1971), which uses only function values for linear approximation.

LIST

displays the model program and variable lists. The LIST option is a debugging feature and is not normally needed.

LISTCODE

displays the derivative tables and the compiled program code. The LISTCODE option is a debugging feature and is not normally needed.

LISTDEP

produces a report that lists, for each variable in the program, the variables that depend on it and on which it depends. The LISTDEP option is a debugging feature and is not normally needed.

LISTDER

displays a table of derivatives. This table lists each nonzero derivative computed for the problem. The LISTDER option is a debugging feature and is not normally needed.

LOGNOTE< =*n*>

writes periodic notes to the log that describe the current status of computations. It is designed for use with analyses requiring extensive CPU resources. The optional integer value n specifies the desired level of reporting detail. The default is $n = 1$. Choosing $n = 2$ adds information about the objective function values at the end of each iteration. The most detail is obtained with $n = 3$, which also reports the results of function evaluations within iterations.

LSPRECISION=*r*

LSP=*r*

specifies the degree of accuracy that should be obtained by the line-search algorithms LIS=2 and LIS=3. Usually an imprecise line search is inexpensive and successful. For more difficult optimization problems, a more precise and expensive line search might be necessary (Fletcher 1987). The second line-search method (which is the default for the NEWRAP, QUANEW, and CONGRA techniques) and the third line-search method approach exact line search for small LSPRECISION= values. If you have numerical problems, you should try to decrease the LSPRECISION= value to obtain a more precise line search. The default values are shown in the following table.

TECH=	UPDATE=	LSP Default
QUANEW	DBFGS, BFGS	$r = 0.4$
QUANEW	DDFP, DFP	$r = 0.06$
CONGRA	all	$r = 0.1$
NEWRAP	no update	$r = 0.9$

For more details, see Fletcher (1987).

MAXFUNC=*i*

MAXFU=*i*

specifies the maximum number (i) of function calls in the optimization process. The default value is $\max(10 \times j,\ n)$, where j is the value of the MAXITER= option and n is shown in the following table.

TECH=	n
TRUREG, NRRIDG, NEWRAP	125
QUANEW, DBLDOG	500
CONGRA	1000
NMSIMP	3000

Note that the optimization can terminate only after completing a full iteration. Therefore, the number of function calls that is actually performed can exceed the number that is specified by the MAXFUNC= option.

MAXITER=*j*

MAXIT=*j*

specifies the maximum number (j) of iterations in the optimization process. The default values are shown in the following table.

TECH=	j
TRUREG, NRRIDG, NEWRAP	50
QUANEW, DBLDOG	200
CONGRA	400
NMSIMP	1000

These default values are also valid when j is specified as a missing value.

MAXSTEP=*r***< [***n***] >**

specifies an upper bound for the step length of the line-search algorithms during the first n iterations. By default, r is the largest double-precision value and n is the largest integer available. Setting this option can improve the speed of convergence for the CONGRA, QUANEW, and NEWRAP techniques.

MAXTIME=*r*

specifies an upper limit of *r* seconds of CPU time for the optimization process. The time checked only at the end of each iteration. Therefore, the actual run time might be longer than the specified time. By default, CPU time is not limited. The actual running time includes the rest of the time needed to finish the iteration and the time needed to generate the output of the results.

METHOD=*value*

specifies the method for approximating the integral of the likelihood over the random effects. Valid values are as follows:

FIRO

specifies the first-order method of Beal and Sheiner (1982). When using METHOD=FIRO, you must specify the NORMAL distribution in the MODEL statement and you must also specify a RANDOM statement.

GAUSS

specifies adaptive Gauss-Hermite quadrature (Pinheiro and Bates 1995). You can prevent the adaptation with the NOAD option or prevent adaptive scaling with the NOADSCALE option. This is the default integration method.

HARDY

specifies Hardy quadrature based on an adaptive trapezoidal rule. This method is available only for one-dimensional integrals; that is, you must specify only one random effect.

ISAMP

specifies adaptive importance sampling (Pinheiro and Bates 1995). You can prevent the adaptation with the NOAD option or prevent adaptive scaling with the NOADSCALE option. You can use the SEED= option to specify a starting seed for the random number generation used in the importance sampling. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock.

MINITER=*i*

MINIT=*i*

specifies the minimum number of iterations. The default value is 0. If you request more iterations than are actually needed for convergence to a stationary point, the optimization algorithms can behave strangely. For example, the effect of rounding errors can prevent the algorithm from continuing for the required number of iterations.

MSINGULAR=*r* > 0

MSING=*r* > 0

specifies a relative singularity criterion for the computation of the inertia (number of positive, negative, and zero eigenvalues) of the Hessian and its projected forms. The default value is 1E–12 if you do not specify the SINGHESS= option; otherwise, the default value is $\max(10\epsilon, (1\mathrm{E}-4) \times \mathrm{SINGHESS})$. See the section "Covariance Matrix" on page 7309 for more information.

NOAD

requests that the Gaussian quadrature be nonadaptive; that is, the quadrature points are centered at zero for each of the random effects and the current random-effects variance matrix is used as the scale matrix.

NOADSCALE

requests nonadaptive scaling for adaptive Gaussian quadrature; that is, the quadrature points are centered at the empirical Bayes estimates for the random effects, but the current random-effects variance matrix is used as the scale matrix. By default, the observed Hessian from the current empirical Bayes estimates is used as the scale matrix.

NOSORTSUB

requests that the data be processed sequentially and forms a new subject whenever the value of the SUBJECT= variable changes from the previous observation. This option enables PROC NLMIXED to construct the clusters by sequential processing.

If you specify more than one RANDOM statement, this option is ignored.

NTHREADS=*n* > -2

specifies the number of threads to use for the log-likelihood calculation that uses the Gaussian quadrature method. When calculating the approximated marginal log likelihood by using the Gaussian quadrature method, PROC NLMIXED allocates data to different threads and calculates the objective function by accumulating values from each thread. NTHREADS=–1 sets the number of available threads to the number of hyperthreaded cores available on the system. By default, NTHREADS=1. This option is valid only when METHOD=GAUSS.

OPTCHECK<=*r* > 0**>**

computes the function values $f(\boldsymbol{\theta}_l)$ of a grid of points $\boldsymbol{\theta}_l$ in a ball of radius of r about $\boldsymbol{\theta}^*$. If you specify the OPTCHECK option without factor r, the default value is $r = 0.1$ at the starting point and $r = 0.01$ at the terminating point. If a point $\boldsymbol{\theta}_l^*$ is found that has a better function value than $f(\boldsymbol{\theta}^*)$, then optimization is restarted at $\boldsymbol{\theta}_l^*$.

OUTQ=*SAS-data-set*

specifies an output data set that contains the quadrature points used for numerical integration.

OUTR=*SAS-data-set*

specifies an output data set that contains empirical Bayes estimates of the random effects of all hierarchies and their approximate standard errors.

QFAC=*r* > 0

specifies the additive factor used to adaptively search for the number of quadrature points. For METHOD=GAUSS, the search sequence is 1, 3, 5, 7, 9, 11, $11 + r$, $11 + 2r$, ..., where the default value of r is 10. For METHOD=ISAMP, the search sequence is 10, $10 + r$, $10 + 2r$, ..., where the default value of r is 50.

QMAX=*r* > 0

specifies the maximum number of quadrature points permitted before the adaptive search is aborted. The default values are 31 for adaptive Gaussian quadrature, 61 for nonadaptive Gaussian quadrature, 160 for adaptive importance sampling, and 310 for nonadaptive importance sampling.

QPOINTS=*n* > 0

specifies the number of quadrature points to be used during evaluation of integrals. For METHOD=GAUSS, *n* equals the number of points used in each dimension of the random effects, resulting in a total of n^r points, where *r* is the number of dimensions. For METHOD=ISAMP, *n* specifies the total number of quadrature points regardless of the dimension of the random effects. By default, the number of quadrature points is selected adaptively, and this option disables the adaptive search.

QSCALEFAC=*r* > 0

specifies a multiplier for the scale matrix used during quadrature calculations. The default value is 1.0.

QTOL=*r* > 0

specifies the tolerance used to adaptively select the number of quadrature points. When the relative difference between two successive likelihood calculations is less than *r*, then the search terminates and the lesser number of quadrature points is used during the subsequent optimization process. The default value is 1E–4.

RESTART=*i* > 0

REST=*i* > 0

specifies that the QUANEW or CONGRA algorithm is restarted with a steepest descent/ascent search direction after, at most, *i* iterations. Default values are as follows:

- CONGRA: UPDATE=PB: restart is performed automatically, *i* is not used.
- CONGRA: UPDATE≠PB: $i = \min(10n, 80)$, where *n* is the number of parameters.
- QUANEW: *i* is the largest integer available.

SEED=*i*

specifies the random number seed for METHOD=ISAMP. If you do not specify a seed, or if you specify a value less than or equal to zero, the seed is generated from reading the time of day from the computer clock. The value must be less than $2^{31} - 1$.

SINGCHOL=*r* > 0

specifies the singularity criterion *r* for Cholesky roots of the random-effects variance matrix and scale matrix for adaptive Gaussian quadrature. The default value is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGHESS=*r* > 0

specifies the singularity criterion *r* for the inversion of the Hessian matrix. The default value is 1E–8. See the ASINGULAR, MSINGULAR=, and VSINGULAR= options for more information.

SINGSWEEP=*r* > 0

specifies the singularity criterion *r* for inverting the variance matrix in the first-order method and the empirical Bayes Hessian matrix. The default value is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGVAR=*r* > 0
specifies the singularity criterion *r* below which statistical variances are considered to equal zero. The default value is 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

START
requests that the gradient of the log likelihood at the starting values be displayed. If you also specify the HESS option, then the starting Hessian is displayed as well.

SUBGRADIENT=*SAS-data-set*
SUBGRAD=*SAS-data-set*
specifies a SAS data set that contains subgradients. In models that use the RANDOM statement, the data set contains the subject-specific gradients of the integrated, marginal log likelihood with respect to all parameters. The sum of the subject-specific gradients equals the gradient that is reported in the "Parameter Estimates" table. The data set contains a variable that identifies the subjects.

In models that do not use the RANDOM statement, the data set contains the observation-wise gradient. The variable identifying the SUBJECT= is then replaced with the Observation. This observation counter includes observations not used in the analysis and is reset in each BY group.

Saving disaggregated gradient information by specifying the SUBGRADIENT= option requires that you also specify METHOD=GAUSS or METHOD=ISAMP.

If you specify more than one RANDOM statement, this option is ignored.

TECHNIQUE=*value*
TECH=*value*
specifies the optimization technique. By default, TECH = QUANEW. Valid values are as follows:

- **CONGRA**
  performs a conjugate-gradient optimization, which can be more precisely specified with the UPDATE= option and modified with the LINESEARCH= option. When you specify this option, UPDATE=PB by default.
- **DBLDOG**
  performs a version of double-dogleg optimization, which can be more precisely specified with the UPDATE= option. When you specify this option, UPDATE=DBFGS by default.
- **NMSIMP**
  performs a Nelder-Mead simplex optimization.
- **NONE**
  does not perform any optimization. This option can be used as follows:
  - to perform a grid search without optimization
  - to compute estimates and predictions that cannot be obtained efficiently with any of the optimization techniques
- **NEWRAP**
  performs a Newton-Raphson optimization combining a line-search algorithm with ridging. The line-search algorithm LIS=2 is the default method.
- **NRRIDG**
  performs a Newton-Raphson optimization with ridging.

- **QUANEW**
  performs a quasi-Newton optimization, which can be defined more precisely with the UPDATE= option and modified with the LINESEARCH= option. This is the default estimation method.
- **TRUREG**
  performs a trust region optimization.

TRACE
displays the result of each operation in each statement in the model program as it is executed. This debugging option is very rarely needed, and it produces voluminous output.

UPDATE=*method*
UPD=*method*
specifies the update method for the quasi-Newton, double-dogleg, or conjugate-gradient optimization technique. Not every update method can be used with each optimizer. See the section "Optimization Algorithms" on page 7295 for more information.

Valid methods are as follows:

- **BFGS**
  performs the original Broyden, Fletcher, Goldfarb, and Shanno (BFGS) update of the inverse Hessian matrix.
- **DBFGS**
  performs the dual BFGS update of the Cholesky factor of the Hessian matrix. This is the default update method.
- **DDFP**
  performs the dual Davidon, Fletcher, and Powell (DFP) update of the Cholesky factor of the Hessian matrix.
- **DFP**
  performs the original DFP update of the inverse Hessian matrix.
- **PB**
  performs the automatic restart update method of Powell (1977) and Beale (1972).
- **FR**
  performs the Fletcher-Reeves update (Fletcher 1987).
- **PR**
  performs the Polak-Ribiere update (Fletcher 1987).
- **CD**
  performs a conjugate-descent update of Fletcher (1987).

VSINGULAR=*r* > 0
VSING=*r* > 0
specifies a relative singularity criterion for the computation of the inertia (number of positive, negative, and zero eigenvalues) of the Hessian and its projected forms. The default value is r = 1E–8 if the SINGHESS= option is not specified, and it is the value of SINGHESS= option otherwise. See the section "Covariance Matrix" on page 7309 for more information.

XCONV=*r***< [***n***] >**

XTOL=*r***< [***n***] >**

specifies the relative parameter convergence criterion. For all techniques except NMSIMP, termination requires a small relative parameter change in subsequent iterations:

$$\frac{\max_j |\theta_j^{(k)} - \theta_j^{(k-1)}|}{\max(|\theta_j^{(k)}|, |\theta_j^{(k-1)}|, \text{XSIZE})} \leq r$$

For the NMSIMP technique, the same formula is used, but $\boldsymbol{\theta}^{(k)}$ is defined as the vertex with the lowest function value and $\boldsymbol{\theta}^{(k-1)}$ is defined as the vertex with the highest function value in the simplex.

The default value is r = 1E–8 for the NMSIMP technique and r = 0 otherwise. The optional integer value n specifies the number of successive iterations for which the criterion must be satisfied before the process can be terminated.

XREF

displays a cross-reference of the variables in the program showing where each variable is referenced or given a value. The XREF listing does not include derivative variables. This option is a debugging feature and is not normally needed.

XSIZE=*r* > 0

specifies the XSIZE parameter of the relative parameter termination criterion. The default value is r = 0. For more details, see the XCONV= option.

ARRAY Statement

ARRAY *arrayname* **[** *dimensions* **]** < **$** > < *variables-and-constants* > **;**

The ARRAY statement is similar to, but not exactly the same as, the ARRAY statement in the SAS DATA step, and it is exactly the same as the ARRAY statements in the NLIN, NLP, and MODEL procedures. The ARRAY statement is used to associate a name (of no more than eight characters) with a list of variables and constants. The array name is used with subscripts in the program to refer to the array elements. The following statements illustrate this:

```
array r[8] r1-r8;

do i = 1 to 8;
   r[i] = 0;
end;
```

The ARRAY statement does not support all the features of the ARRAY statement in the DATA step. It cannot be used to assign initial values to array elements. Implicit indexing of variables cannot be used; all array references must have explicit subscript expressions. Only exact array dimensions are allowed; lower-bound specifications are not supported. A maximum of six dimensions is allowed.

On the other hand, the ARRAY statement does allow both variables and constants to be used as array elements. (Constant array elements cannot have values assigned to them.) Both dimension specification and the list of elements are optional, but at least one must be specified. When the list of elements is not specified or fewer elements than the size of the array are listed, array variables are created by suffixing element numbers to the array name to complete the element list.

BOUNDS Statement

BOUNDS *b-con* < , *b-con* . . . > ;

where	*b-con* :=	number *operator* parameter_list *operator* number
or	*b-con* :=	number *operator* parameter_list
or	*b-con* :=	parameter_list *operator* number
and	*operator* :=	<=, <, >=, or >

Boundary constraints are specified with a BOUNDS statement. One- or two-sided boundary constraints are allowed. The list of boundary constraints are separated by commas. For example:

```
bounds 0 <= a1-a9 X <= 1, -1 <= c2-c5;
bounds b1-b10 y >= 0;
```

You can specify more than one BOUNDS statement. If you specify more than one lower (upper) bound for the same parameter, the maximum (minimum) of these is taken.

If the maximum l_j of all lower bounds is larger than the minimum of all upper bounds u_j for the same parameter θ_j, the boundary constraint is replaced by $\theta_j := l_j := \min(u_j)$ defined by the minimum of all upper bounds specified for θ_j.

BY Statement

BY *variables* ;

You can specify a BY statement in PROC NLMIXED to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the NLMIXED procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

An optimization problem is solved for each BY group separately unless the TECH=NONE option is specified.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CMPTMODEL Statement

CMPTMODEL *required-options conditionally-required-options* < *options* > **;**

The CMPTMODEL statement computes predicted concentrations from a specified one-, two-, or three-compartment model. The CMPTMODEL statement includes three types of options:

- The following *required-options* are required: PCONC=, TIME=, NCOMPS=, ADMTYPE=, and PARMTYPE=.
- The following *conditionally-required-options* are conditionally required depending on the specification of the *required-options*: CL*n*=, VOL*n*=, K12=, K21=, K13=, K31=, KA=, K*n*0=, RATE=, and DURN=.
- The following *options* are optional: DOSE*n*=, SCALE*n*=, PCONC0=, PCONC2=, and PCONC3=.

You must specify the following *required-options*:

ADMTYPE=IVB | INF | ORAL
specifies the administration type. You can specify the following values:

IVB specifies the bolus type of administration.

INF specifies the infusion type of administration.

ORAL specifies the oral type of administration.

NCOMPS=1 | 2 | 3
specifies the number of compartments in a model.

PARMTYPE=1 | 2
PTYPE=1 | 2
specifies the parameterization type. You can specify the following values:

1 parameterizes the compartment model in terms of elimination and transfer rate constants.

2 parameterizes the compartment model in terms of clearance and volume constants.

PCONC=*variable*
PCONC1=*variable*
specifies the outcome variable that is the predicted concentration in the first or central compartment at each time point.

TIME=*variable*
specifies the time value at which the predicted concentrations are computed, where *variable* can be a data set variable or a defined variable.

You can combine the NCOMPS=, ADMTYPE=, and PARMTYPE= options to fit a compartment model (one-, two-, or three- compartments) with different administration methods (bolus, infusion, or oral) using different parameterizations. The PCONC= variable is the outcome that is the evaluated predicted concentration at the TIME= value. You can use this variable in model specification in your program.

You might need to specify one or more of the following *conditionally-required-options* based on what you specify in the *required-options*:

CL*n***=***variable*
specifies the clearance value for the *n*th compartment, where *n* takes the values of 1, 2, or 3. (CL is an alias for CL1.) This option is valid only in models that require clearance (PARMTYPE=2). The *variable* can be a data set variable or a defined variable.

- You must specify a CL1= option when you specify a one-compartment model (NCOMPS=1)
- You must specify CL1= and CL2= options when you specify a two-compartment model (NCOMPS=2)
- You must specify CL1=, CL2=, and CL3= options when you specify a three-compartment model (NCOMPS=3)

DURN=*variable*
specifies the duration of the infusion. This option is valid only in infusion models (ADMTYPE=INF). The *variable* can be a data set variable or a defined variable. In an infusion model, you must specify either this option or the RATE= option.

KA=*variable*
specifies the absorption rate constant. This option is valid only for the oral type of compartment models (ADMTYPE=ORAL).

K12=*variable*
specifies the transfer rate constant from the first (central) compartment to the second compartment. This option is valid only in models that require transfer rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable. You must specify this option in a two- or three-compartment model.

K21=*variable*
specifies the transfer rate constant from the second compartment to the first (central) compartment. This option is valid only in models that require transfer rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable. You must specify this option in a two- or three- compartment model.

K13=*variable*
specifies the transfer rate constant from the first (central) compartment to the third compartment. This option is valid only in models that require transfer rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable. You must specify this option in a three-compartment model.

K31=*variable*
specifies the transfer rate constant from the third compartment to the first (central) compartment. This option is valid only in models that require transfer rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable. You must specify this option in a three-compartment model.

K*n***0=***variable*
specifies the elimination rate constant for the *n*th compartment, where *n* takes the values of 1, 2, or 3. (Ke is an alias for K10.) This option is valid only in models that require elimination rate (PARMTYPE=1). The *variable* can be a data set variable or a defined variable.

- You must specify a K10= option when you specify a one-compartment model (NCOMPS=1)
- You must specify a K10= option and you can optionally specify a K20= option when you specify a two-compartment model (NCOMPS=2)

- You must specify a K10= option and you can optionally specify K20= and K30= options when you specify a three-compartment model (NCOMPS=3)

RATE=*variable*
specifies the rate of the infusion. This option is valid only in infusion models (ADMTYPE=INF). The *variable* can be a data set variable or a defined variable. In an infusion model, you must specify either this option or the DURN= option.

VOL*n***=***variable*
specifies the apparent volume of distribution of a drug in the *n*th compartment, where *n* takes the values of 1, 2, or 3. (VOL is an alias for VOL1.) This option is valid only in models that require volume (PARMTYPE=2). The *variable* can be a data set variable or a defined variable.

- You must specify a VOL1= option when you specify a one-compartment model (NCOMPS=1)
- You must specify VOL1= and VOL2= options when you specify a two-compartment model (NCOMPS=2)
- You must specify VOL1=, VOL2=, and VOL3= options when you specify a three-compartment model (NCOMPS=3)

You can also specify the following optional *options*:

DOSE*n***=***variable*
specifies the amount of the dose for the *n*th compartment, where *n* takes the values of 0, 1, 2, or 3. (DOSE is an alias for DOSE1.) The DOSE0= option specifies the amount of dose in the zero (depot) compartment, and is valid only in oral models (ADMTYPE=ORAL). The *variable* can be a data set variable or a defined variable.

- DOSE1= is optional and valid for one-, two-, and three-compartment models
- DOSE2= is optional and valid only for two-, and three-compartment models
- DOSE3= is optional and valid only for a three-compartment model

When ADMTYPE=ORAL is specified, DOSE0=1, DOSE1=0, DOSE2=0, and DOSE3=0 by default. Otherwise, DOSE1=1, DOSE2=0, and DOSE3=0 by default.

PCONC0=*variable*
specifies the outcome *variable* that is the predicted concentration of the zero (depot) compartment. This option is valid only when the ADMTYPE= ORAL option is specified.

PCONC2=*variable*
specifies the outcome *variable* that is the predicted concentration of the second compartment. This option is valid only in two- and three-compartment models.

PCONC3=*variable*
specifies the outcome *variable* that is the predicted concentration of the second compartment. This option is valid only in three-compartment models.

SCALE*n***=***variable*

specifies the scale value for the *n*th compartment. The *variable* can be a data set variable or a defined variable. SCALE0= option is valid only in oral models (ADMTYPE=ORAL).

- SCALE1= is optional and valid for one-, two-, and three-compartment models
- SCALE2= is optional and valid only for two-, and three-compartment models
- SCALE3= is optional and valid only for a three-compartment model

By default, all optional SCALE*n* values are set to 1.

For more information, see the section "Compartment Models" on page 7283.

CONTRAST Statement

CONTRAST '*label*' *expression* < **,** *expression* > < *option* > **;**

The CONTRAST statement enables you to conduct a statistical test that several expressions simultaneously equal zero. The expressions are typically contrasts—that is, differences whose expected values equal zero under the hypothesis of interest.

In the CONTRAST statement, you must provide a quoted string to identify the contrast and then a list of valid SAS expressions separated by commas. Multiple CONTRAST statements are permitted, and results from all statements are listed in a common table. PROC NLMIXED constructs approximate F tests for each statement by using the delta method (Cox 1998) to approximate the variance-covariance matrix of the constituent expressions.

You can specify the following *option*:

DF=*d*

specifies the denominator degrees of freedom to be used in computing p values for the F statistics. By default, the value of *d* is the value of the DF= option in the PROC NLMIXED statement.

ESTIMATE Statement

ESTIMATE '*label*' *expression* < *options* > **;**

The ESTIMATE statement enables you to compute an additional estimate that is a function of the parameter values. You must provide a quoted string to identify the estimate and then a valid SAS expression. Multiple ESTIMATE statements are permitted, and results from all statements are listed in a common table. PROC NLMIXED computes approximate standard errors for the estimates by using the delta method (Billingsley 1986). It uses these standard errors to compute corresponding t statistics, p-values, and confidence limits.

If you specify the ECOV option in the PROC NLMIXED statement, the procedure also produces a table that contains the approximate covariance matrix of all the additional estimates you specify. Similarly, the ECORR option produces a table of the corresponding correlation matrix, and the EDER option produces a table of the derivatives of the additional estimates with respect to each of the model parameters.

You can specify the following *options*:

ALPHA=α
specifies the alpha level to be used in computing confidence limits. By default, the value of α is the value of the ALPHA= option in the PROC NLMIXED statement.

DF=*d*
specifies the degrees of freedom to be used in computing p-values and confidence limits. By default, the value of the *d* is the value of the DF= option in the PROC NLMIXED statement.

ID Statement

ID *names* **;**

The ID statement identifies additional quantities to be included in the OUT= data set of the PREDICT statement. These can be any symbols you have defined with SAS programming statements.

MODEL Statement

MODEL *dependent-variable* ∼ *distribution* **;**

The MODEL statement is the mechanism for specifying the conditional distribution of the data given the random effects. You must specify a single dependent variable from the input data set, a tilde ∼, and then a distribution with its parameters. Valid distributions are as follows.

- **normal**(m,v) specifies a normal (Gaussian) distribution with mean m and variance v.
- **binary**(p) specifies a binary (Bernoulli) distribution with probability p.
- **binomial**(n,p) specifies a binomial distribution with count n and probability p.
- **gamma**(a,b) specifies a gamma distribution with shape a and scale b.
- **negbin**(n,p) specifies a negative binomial distribution with count n and probability p.
- **poisson**(m) specifies a Poisson distribution with mean m.
- **general**(ll) specifies a general log likelihood function that you construct using SAS programming statements.

The MODEL statement must follow any SAS programming statements you specify for computing parameters of the preceding distributions. See the section "Built-In Log-Likelihood Functions" on page 7281 for expressions of the built-in conditional log-likelihood functions.

PARMS Statement

PARMS < *name-list* < **=***numbers* > > < **,** *name-list* < **=***numbers* > . . . >
< / *options* > **;**

The PARMS statement lists names of parameters and specifies initial values, possibly over a grid. You can specify the parameters and values directly in a list, or you can provide the name of a SAS data set that contains them by using the DATA= option.

While the PARMS statement is not required, you are encouraged to use it to provide PROC NLMIXED with accurate starting values. Parameters not listed in the PARMS statement are assigned an initial value of 1. PROC NLMIXED considers all symbols not assigned values to be parameters, so you should specify your modeling statements carefully and check the output from the "Parameters" table to make sure the proper parameters are identified.

A list of parameter names in the PARMS statement is not separated by commas and is followed by an equal sign and a list of numbers. If the number list consists of only one number, this number defines the initial value for all the parameters listed to the left of the equal sign.

If the number list consists of more than one number, these numbers specify the grid locations for each of the parameters listed to the left of the equal sign. You can use the TO and BY keywords to specify a number list for a grid search. If you specify a grid of points in a PARMS statement, PROC NLMIXED computes the objective function value at each grid point and chooses the best (feasible) grid point as an initial point for the optimization process. You can use the BEST= option to save memory for the storing and sorting of all grid point information.

The following options are available in the PARMS statement after a slash (/):

BEST=*i* > 0
specifies the maximum number of points displayed in the "Parameters" table, selected as the points with the maximum likelihood values. By default, all grid values are displayed.

BYDATA
enables you to assign different starting values for each BY group by using the DATA=*SAS-data-set* option during BY processing. By default, BY groups are ignored in the PARMS data set. For the BYDATA option to be effective, the DATA= data set must contain the BY variables and the same BY groups as the primary input data set. When you supply a grid of starting values with the DATA= data set and the BYDATA option is in effect, the size of the grid is determined by the first BY group.

DATA=*SAS-data-set*
specifies a SAS data set containing parameter names and starting values. The data set should be in one of two forms: narrow or wide. The narrow-form data set contains the variables Parameter and Estimate, with parameters and values listed as distinct observations. The wide-form data set has the parameters themselves as variables, and each observation provides a different set of starting values. By default, BY groups are ignored in this data set, so the same starting grid is evaluated for each BY group. You can vary the starting values for BY groups by using the BYDATA option.

PREDICT Statement

PREDICT *expression* **OUT=***SAS-data-set* < *options* > **;**

The PREDICT statement enables you to construct predictions of an expression across all of the observations in the input data set. Any valid SAS programming expression involving the input data set variables, parameters, and random effects is valid. Predicted values are computed using the parameter estimates and empirical Bayes estimates of the random effects. Standard errors of prediction are computed using the delta method (Billingsley 1986; Cox 1998). Results are placed in an output data set that you specify with the OUT= option. Besides all variables from the input data set, the OUT= data set contains the following variables: Pred, StdErrPred, DF, tValue, Probt, Alpha, Lower, Upper. You can also add other computed quantities to this data set with the ID statement.

The following options are available in the PREDICT statement:

ALPHA=*α*

specifies the alpha level to be used in computing t statistics and intervals. The default value corresponds to the ALPHA= option in the PROC NLMIXED statement.

DER

requests that derivatives of the predicted expression with respect to all parameters be included in the OUT= data set. The variable names for the derivatives are the same as the parameter names with the prefix "Der_" appended. All of the derivatives are evaluated at the final estimates of the parameters and the empirical Bayes estimates of the random effects.

DF=*d*

specifies the degrees of freedom to be used in computing t statistics and intervals in the OUT= data set. The default value corresponds to the DF= option in the PROC NLMIXED statement.

RANDOM Statement

RANDOM *random-effects* ∼ *distribution* **SUBJECT=***variable* < *options* > **;**

The RANDOM statement defines the random effects and their distribution. The random effects must be represented by symbols that appear in your SAS programming statements. The random effects usually influence the mean value of the distribution that is specified in the MODEL statement. The RANDOM statement consists of a list of the random effects (usually just one or two symbols), a tilde (∼), the distribution of the random effects, and then a SUBJECT= variable.

The only distribution available for the random effects is normal(m,v), with mean m and variance v.

This syntax is illustrated as follows for one effect:

```
random u ~ normal(0,s2u) subject=clinic;
```

For multiple effects, you should specify bracketed vectors for m and v, the latter consisting of the lower triangle of the random-effects variance matrix listed in row order. This is illustrated for two random effects as follows:

```
random b1 b2 ~ normal([0,0],[g11,g21,g22]) subject=person;
```

Similarly, the syntax for three random effects is illustrated as follows:

```
random b1 b2 b3 ~ normal([0,0,0],[g11,g21,g22,g31,g32,g33])
   subject=person;
```

The SUBJECT= variable determines the unique realizations of the random effects.

PROC NLMIXED constructs the subject clusters based on unique values in the SUBJECT= variable. The input data set does not need to be sorted by SUBJECT= variable.

You can specify multiple RANDOM statements. When you specify more than one RANDOM statement, PROC NLMIXED assumes that the SUBJECT= variable from each RANDOM statement forms a containment hierarchy. For more information, see the section "Hierarchical Model Specification" on page 7293. The syntax for two RANDOM statements is illustrated as follows:

```
random r11 ~ normal(0,sd1) subject = school;
random r21 ~ normal(0,sd2) subject = class(school);
```

You can specify the following options in the RANDOM statement:

ALPHA=α

specifies the alpha level to be used in computing t statistics and intervals. The default value corresponds to the value of the ALPHA= option in the PROC NLMIXED statement.

DF=*d*

specifies the degrees of freedom to be used in computing t statistics and intervals in the OUT= data set. PROC NLMIXED calculates the default degrees of freedom as follows:

- When only one or no RANDOM statement is specified, the default value corresponds to the value of the DF= option in the PROC NLMIXED statement.
- When multiple RANDOM statements are specified, the default value is the number of subjects minus the number of random-effects variables in the corresponding RANDOM statement.

OUT=*SAS-data-set*

requests an output data set that contains empirical Bayes estimates of the random effects and their approximate standard errors of prediction.

REPLICATE Statement

REPLICATE *variable* **;**

The REPLICATE statement provides a way to accommodate models in which different subjects have identical data. This occurs most commonly when the dependent variable is binary. When you specify a REPLICATE variable, PROC NLMIXED treats its value as the number of subjects that have data identical to the data for the current value of the SUBJECT= variable (specified in the RANDOM statement). Only the last observation of the REPLICATE variable for each subject is used, and the replicate variable must have only positive values.

The function of a REPLICATE statement is related to but not quite the same as the function of either a FREQ or a WEIGHT statement in other statistical modeling procedures, such as the GLM, GENMOD, GLIMMIX, and LOGISTIC procedures. A FREQ or a WEIGHT value essentially multiplies the log likelihood or sum of squares contribution for each observation. On the other hand, a REPLICATE value multiplies the log likelihood contribution of each subject, which consists of one or more observations. When no SUBJECT= variable is specified, the REPLICATE value behaves like a weight, multiplying each observation's log likelihood contribution.

NOTE: The REPLICATE statement is not allowed when there is more than one RANDOM statement.

Programming Statements

This section lists the programming statements that are used to code the log-likelihood function in PROC NLMIXED. It also documents the differences between programming statements in PROC NLMIXED and programming statements in the SAS DATA step. The syntax of programming statements used in PROC NLMIXED is identical to that used in the CALIS and GENMOD procedures (see Chapter 33 and Chapter 51, respectively), and the MODEL procedure (see the *SAS/ETS User's Guide*). Most of the programming statements that can be used in the SAS DATA step can also be used in the NLMIXED procedure. See *SAS DATA Step Statements: Reference* for a description of SAS programming statements. The following are valid statements:

ABORT;
ARRAY *arrayname* < **[** *dimensions* **]** > < **$** > < *variables-and-constants* >**;**
CALL *name* < **(***expression* < **,** *expression* . . . >**)** >**;**
DELETE;
DO < *variable* **=** *expression* < **TO** *expression* > < **BY** *expression* > >
< **,** *expression* < **TO** *expression* > < **BY** *expression* > > . . .
< **WHILE** *expression* > < **UNTIL** *expression* >**;**
END;
GOTO *statement-label***;**
IF *expression***;**
IF *expression* **THEN** *program-statement***;**
ELSE *program-statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement-label***;**
PUT < *variable* > < **=** > . . . **;**
RETURN;
SELECT < **(***expression***)** >**;**
STOP;
SUBSTR(*variable***,** *index***,** *length***)=** *expression***;**
WHEN (*expression***)***program-statement***;**
OTHERWISE *program-statement***;**

For the most part, the SAS programming statements work the same as they do in the SAS DATA step, as documented in *SAS Programmers Guide: Essentials*; however, there are the following differences:

- The ABORT statement does not allow any arguments.
- The DO statement does not allow a character index variable. Thus

  ```
  do i = 1,2,3;
  ```

  is supported, but the following statement is not supported:

  ```
  do i = 'A','B','C';
  ```

- The LAG function does work appropriately with PROC NLMIXED, but you can use the ZLAG function instead.
- The PUT statement, used mostly for program debugging in PROC NLMIXED, supports only some of the features of the DATA step PUT statement, and it has some new features that the DATA step PUT statement does not.
  - The PROC NLMIXED PUT statement does not support line pointers, factored lists, iteration factors, overprinting, _INFILE_, the colon (:) format modifier, or "$".
  - The PROC NLMIXED PUT statement does support expressions, but the expression must be enclosed in parentheses. For example, the following statement displays the square root of x:

    ```
    put  (sqrt(x));
    ```

- The PROC NLMIXED PUT statement supports the item _PDV_ to display a formatted listing of all variables in the program. For example, the following statement displays a much more readable listing of the variables than the _ALL_ print item:

  ```
  put  _pdv_;
  ```

- The WHEN and OTHERWISE statements enable you to specify more than one target statement. That is, DO/END groups are not necessary for multiple statement WHENs. For example, the following syntax is valid:

```
select;
   when (exp1) stmt1;
               stmt2;
   when (exp2) stmt3;
               stmt4;
end;
```

When coding your programming statements, you should avoid defining variables that begin with an underscore (_), because they might conflict with internal variables created by PROC NLMIXED. The MODEL statement must come after any SAS programming statements that define or modify terms used in the construction of the log-likelihood.

Details: NLMIXED Procedure

This section contains details about the underlying theory and computations of PROC NLMIXED.

Modeling Assumptions and Notation

PROC NLMIXED operates under the following general framework for nonlinear mixed models. Assume that you have an observed data vector $\mathbf{y}_i$ for each of i subjects, $i = 1, \ldots, s$. The $\mathbf{y}_i$ are assumed to be independent across i, but within-subject covariance is likely to exist because each of the elements of $\mathbf{y}_i$ is measured on the same subject. As a statistical mechanism for modeling this within-subject covariance, assume that there exist latent random-effect vectors $\mathbf{u}_i$ of small dimension (typically one or two) that are also independent across i. Assume also that an appropriate model linking $\mathbf{y}_i$ and $\mathbf{u}_i$ exists, leading to the joint probability density function

$$p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)q(\mathbf{u}_i|\boldsymbol{\xi})$$

where $\mathbf{X}_i$ is a matrix of observed explanatory variables and $\boldsymbol{\phi}$ and $\boldsymbol{\xi}$ are vectors of unknown parameters.

Let $\boldsymbol{\theta} = [\boldsymbol{\phi}, \boldsymbol{\xi}]$ and assume that it is of dimension n. Then inferences about $\boldsymbol{\theta}$ are based on the marginal likelihood function

$$m(\boldsymbol{\theta}) = \prod_{i=1}^{s} \int p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)q(\mathbf{u}_i|\boldsymbol{\xi})d\mathbf{u}_i$$

In particular, the function

$$f(\boldsymbol{\theta}) = -\log m(\boldsymbol{\theta})$$

is minimized over $\boldsymbol{\theta}$ numerically in order to estimate $\boldsymbol{\theta}$, and the inverse Hessian (second derivative) matrix at the estimates provides an approximate variance-covariance matrix for the estimate of $\boldsymbol{\theta}$. The function $f(\boldsymbol{\theta})$ is referred to both as the negative log likelihood function and as the objective function for optimization.

As an example of the preceding general framework, consider the nonlinear growth curve example in the section "Getting Started: NLMIXED Procedure" on page 7240. Here, the conditional distribution $p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, u_i)$ is normal with mean

$$\frac{b_1 + u_{i1}}{1 + \exp[-(d_{ij} - b_2)/b_3]}$$

and variance σ_e^2; thus $\boldsymbol{\phi} = [b_1, b_2, b_3, \sigma_e^2]$. Also, u_i is a scalar and $q(u_i|\xi)$ is normal with mean 0 and variance σ_u^2; thus $\xi = \sigma_u^2$.

The following additional notation is also found in this chapter. The quantity $\boldsymbol{\theta}^{(k)}$ refers to the parameter vector at the kth iteration, the vector $\mathbf{g}(\boldsymbol{\theta})$ refers to the gradient vector $\nabla f(\boldsymbol{\theta})$, and the matrix $\mathbf{H}(\boldsymbol{\theta})$ refers to the Hessian $\nabla^2 f(\boldsymbol{\theta})$. Other symbols are used to denote various constants or option values.

Nested Multilevel Nonlinear Mixed Models

The general framework for nested multilevel nonlinear mixed models in cases of two levels can be explained as follows. Let $\mathbf{y}_{j(i)}$ be the response vector observed on subject j that is nested within subject i, where j is commonly referred as the second-level subject and i is the first-level subject. There are s first-level subjects, and each has s_i second-level subjects that are nested within. An example is $\mathbf{y}_{j(i)}$, which are the heights of students in class j of school i, where $j = 1, \ldots, s_i$ for each i and $i = 1, \ldots, s$. Suppose there exist latent random-effect vectors $\mathbf{v}_{j(i)}$ and $\mathbf{v}_i$ of small dimensions for modeling within subject covariance. Assume also that an appropriate model that links $\mathbf{y}_{j(i)}$ and $(\mathbf{v}_{j(i)}, \mathbf{v}_i)$ exists, and if you use the notation $\mathbf{y}_i = (\mathbf{y}_{1(i)}, \ldots, \mathbf{y}_{s_i(i)})$, $\mathbf{u}_i = (\mathbf{v}_i, \mathbf{v}_{1(i)}, \ldots, \mathbf{v}_{s_i(i)})$, and $\boldsymbol{\xi} = (\boldsymbol{\xi}_1, \boldsymbol{\xi}_2)$, the joint density function in terms of the first-level subject can be expressed as

$$p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)q(\mathbf{u}_i|\boldsymbol{\xi}) = \left(\prod_{j=1}^{s_i} p(\mathbf{y}_{j(i)}|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{v}_i, \mathbf{v}_{j(i)})q_2(\mathbf{v}_{j(i)}|\boldsymbol{\xi}_2)\right) q_1(\mathbf{v}_i|\boldsymbol{\xi}_1)$$

As defined in the previous section, the marginal likelihood function where $\boldsymbol{\theta} = [\boldsymbol{\phi}, \boldsymbol{\xi}]$ is

$$m(\boldsymbol{\theta}) = \prod_{i=1}^{s} \int p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)q(\mathbf{u}_i|\boldsymbol{\xi})d\mathbf{u}_i$$

Again, the function

$$f(\boldsymbol{\theta}) = -\log m(\boldsymbol{\theta})$$

is minimized over $\boldsymbol{\theta}$ numerically in order to estimate $\boldsymbol{\theta}$. Models that have more than two levels follow similar notation.

Integral Approximations

An important part of the marginal maximum likelihood method described previously is the computation of the integral over the random effects. The default method in PROC NLMIXED for computing this integral is adaptive Gaussian quadrature as described in Pinheiro and Bates (1995). Another approximation method is the first-order method of Beal and Sheiner (1982, 1988). A description of these two methods follows.

Adaptive Gaussian Quadrature

A quadrature method approximates a given integral by a weighted sum over predefined abscissas for the random effects. A good approximation can usually be obtained with an adequate number of quadrature points as well as appropriate centering and scaling of the abscissas. Adaptive Gaussian quadrature for the integral over $\mathbf{u}_i$ centers the integral at the empirical Bayes estimate of $\mathbf{u}_i$, defined as the vector $\widehat{\mathbf{u}}_i$ that minimizes

$$-\log\left[p(\mathbf{y}_i|\mathbf{X}_i,\boldsymbol{\phi},\mathbf{u}_i)q(\mathbf{u}_i|\boldsymbol{\xi})\right]$$

with $\boldsymbol{\phi}$ and $\boldsymbol{\xi}$ set equal to their current estimates. The final Hessian matrix from this optimization can be used to scale the quadrature abscissas.

Suppose $(z_j, w_j; j = 1,\ldots,p)$ denote the standard Gauss-Hermite abscissas and weights (Golub and Welsch 1969, or Table 25.10 of Abramowitz and Stegun 1972). The adaptive Gaussian quadrature integral approximation is as follows:

$$\int p(\mathbf{y}_i|\mathbf{X}_i,\boldsymbol{\phi},\mathbf{u}_i)q(\mathbf{u}_i|\boldsymbol{\xi})d\mathbf{u}_i \approx$$
$$2^{r/2}\,|\boldsymbol{\Gamma}(\mathbf{X}_i,\boldsymbol{\theta})|^{-1/2}\sum_{j_1=1}^{p}\cdots\sum_{j_r=1}^{p}\left[p(\mathbf{y}_i|\mathbf{X}_i,\boldsymbol{\phi},\mathbf{a}_{j_1,\ldots,j_r})q(\mathbf{a}_{j_1,\ldots,j_r}|\boldsymbol{\xi})\prod_{k=1}^{r}w_{j_k}\exp z_{j_k}^2\right]$$

where r is the dimension of u_i, $\boldsymbol{\Gamma}(\mathbf{X}_i,\boldsymbol{\theta})$ is the Hessian matrix from the empirical Bayes minimization, $\mathbf{z}_{j_1,\ldots,j_r}$ is a vector with elements $(z_{j_1},\ldots,z_{j_r})$, and

$$\mathbf{a}_{j_1,\ldots,j_r} = \widehat{\mathbf{u}}_i + 2^{1/2}\boldsymbol{\Gamma}(\mathbf{X}_i,\boldsymbol{\theta})^{-1/2}\mathbf{z}_{j_1,\ldots,j_r}$$

PROC NLMIXED selects the number of quadrature points adaptively by evaluating the log-likelihood function at the starting values of the parameters until two successive evaluations have a relative difference less than the value of the QTOL= option. The specific search sequence is described under the QFAC= option. Using the QPOINTS= option, you can adjust the number of quadrature points p to obtain different levels of accuracy. Setting $p = 1$ results in the Laplacian approximation as described in: Beal and Sheiner (1992); Wolfinger (1993); Vonesh (1992, 1996); Vonesh and Chinchilli (1997); Wolfinger and Lin (1997).

The NOAD option in the PROC NLMIXED statement requests nonadaptive Gaussian quadrature. Here all $\widehat{\mathbf{u}}_i$ are set equal to zero, and the Cholesky root of the estimated variance matrix of the random effects is substituted for $\boldsymbol{\Gamma}(\mathbf{X}_i,\boldsymbol{\theta})^{-1/2}$ in the preceding expression for $\mathbf{a}_{j_1,\ldots,j_r}$. In this case derivatives are computed using the algorithm of Smith (1995). The NOADSCALE option requests the same scaling substitution but with the empirical Bayes $\widehat{\mathbf{u}}_i$.

When there is one RANDOM statement, the dimension of $\mathbf{u}_i$ is the number of random effects that are specified in that RANDOM statement. For nested multilevel nonlinear mixed models, the dimension

of $\mathbf{u}_i$ could increase significantly. In this case, the dimension of $\mathbf{u}_i$ is equal to the sum of all nested subjects times the corresponding random-effects dimension. For example, consider a three-level nested nonlinear mixed model that contains s first-level subjects, where each first-level subject has s_i second-level subjects that are nested within and s_{ij} third-level subjects that are nested within each combination of first-level subject i and second-level subject j. Then, based on notation in the previous section, $\mathbf{u}_i = (\mathbf{v}_i, \mathbf{v}_{1(i)}, \mathbf{v}_{1(i1)}, \ldots, \mathbf{v}_{s_i1(i1)}, \ldots, \mathbf{v}_{s_i(i)}, \mathbf{v}_{1(is_i)}, \ldots, \mathbf{v}_{s_{is_i}(is_i)})$. Suppose r_i is the random-effects dimension at level i. Then the dimension of $\mathbf{u}_i$, denoted as r, can be computed as $r = r_1 + (r_2 \times s_i) + \left(r_3 \times \sum_{j=1}^{s_i} s_{ij}\right)$. Hence the joint estimation of $\mathbf{u}_i$ is more costly.

When only one RANDOM statement is specified for the model, by default PROC NLMIXED uses Newton-Raphson optimization for empirical Bayes minimization of random effects for each subject. If you specify the EBOPT option in the PROC NLMIXED statement, then the procedure uses Newton-Raphson ridge optimization. But when multiple RANDOM statements are specified, quasi-Newton optimization is used for the empirical Bayes minimization of random effects for each subject. Therefore, all the options that are related to the Newton-Raphson method (such as the EBOPT, EBSSFRAC=, EBSSTOL=, EBSTEPS=, EBSUBSTEPS=, and EBTOL= options) are ignored.

PROC NLMIXED computes the derivatives of the adaptive Gaussian quadrature approximation when carrying out the default dual quasi-Newton optimization. For nested multilevel nonlinear mixed models, PROC NLMIXED does not explicitly compute these derivatives. In this case, it ignores the EMPIRICAL and SUBGRADIENT= options in the PROC NLMIXED statement. Also, nested multilevel nonlinear mixed models use METHOD=GAUSS only. In addition to these changes, for nested multilevel nonlinear mixed models, the default values of GTOL and ABSGTOL are changed to 1E–6 and 1E–3, respectively.

First-Order Method

Another integral approximation available in PROC NLMIXED is the first-order method of Beal and Sheiner (1982, 1988) and Sheiner and Beal (1985). This approximation is used only in the case where $p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)$ is normal—that is,

$$p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i) = (2\pi)^{-n_i/2} |\mathbf{R}_i(\mathbf{X}_i, \boldsymbol{\phi})|^{-1/2} \\ \exp\left\{-(1/2)\left[\mathbf{y}_i - \mathbf{m}_i(\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)\right]' \mathbf{R}_i(\mathbf{X}_i, \boldsymbol{\phi})^{-1} \left[\mathbf{y}_i - \mathbf{m}_i(\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)\right]\right\}$$

where n_i is the dimension of $\mathbf{y}_i$, $\mathbf{R}_i$ is a diagonal variance matrix, and $\mathbf{m}_i$ is the conditional mean vector of $\mathbf{y}_i$.

The first-order approximation is obtained by expanding $\mathbf{m}_i(\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)$ with a one-term Taylor series expansion about $\mathbf{u}_i = \mathbf{0}$, resulting in the approximation

$$p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i) \approx (2\pi)^{-n_i/2} |\mathbf{R}_i(\mathbf{X}_i, \boldsymbol{\phi})|^{-1/2} \\ \exp\Big(-(1/2)\left[\mathbf{y}_i - \mathbf{m}_i(\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{0}) - \mathbf{Z}_i(\mathbf{X}_i, \boldsymbol{\phi})\mathbf{u}_i\right]' \\ \mathbf{R}_i(\mathbf{X}_i, \boldsymbol{\phi})^{-1} \left[\mathbf{y}_i - \mathbf{m}_i(\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{0}) - \mathbf{Z}_i(\mathbf{X}_i, \boldsymbol{\phi})\mathbf{u}_i\right]\Big)$$

where $\mathbf{Z}_i(\mathbf{X}_i, \boldsymbol{\phi})$ is the Jacobian matrix $\partial \mathbf{m}_i(\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i)/\partial \mathbf{u}_i$ evaluated at $\mathbf{u}_i = \mathbf{0}$.

Assuming that $q(\mathbf{u}_i|\boldsymbol{\xi})$ is normal with mean $\mathbf{0}$ and variance matrix $\mathbf{G}(\boldsymbol{\xi})$, the first-order integral approximation is computable in closed form after completing the square:

$$\int p(\mathbf{y}_i|\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{u}_i) q(\mathbf{u}_i|\boldsymbol{\xi})\, d\mathbf{u}_i \approx (2\pi)^{-n_i/2} |\mathbf{V}_i(\mathbf{X}_i, \boldsymbol{\theta})|^{-1/2} \\ \exp\Big(-(1/2)\left[\mathbf{y}_i - \mathbf{m}_i(\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{0})\right]' \mathbf{V}_i(\mathbf{X}_i, \boldsymbol{\theta})^{-1} \left[\mathbf{y}_i - \mathbf{m}_i(\mathbf{X}_i, \boldsymbol{\phi}, \mathbf{0})\right]\Big)$$

where $\mathbf{V}_i(\mathbf{X}_i, \boldsymbol{\theta}) = \mathbf{Z}_i(\mathbf{X}_i, \boldsymbol{\phi})\mathbf{G}(\boldsymbol{\xi})\mathbf{Z}_i(\mathbf{X}_i, \boldsymbol{\phi})' + \mathbf{R}_i(\mathbf{X}_i, \boldsymbol{\phi})$. The resulting approximation for $f(\boldsymbol{\theta})$ is then minimized over $\boldsymbol{\theta} = [\boldsymbol{\phi}, \boldsymbol{\xi}]$ to obtain the first-order estimates. PROC NLMIXED uses finite-difference derivatives of the first-order integral approximation when carrying out the default dual quasi-Newton optimization.

Built-In Log-Likelihood Functions

This section displays the basic formulas used by the NLMIXED procedure to compute the conditional log-likelihood functions of the data given the random effects. Note, however, that in addition to these basic equations, the NLMIXED procedure employs a number of checks for missing values and floating-point arithmetic. You can see the entire program used by the NLMIXED procedure to compute the conditional log-likelihood functions $l(\boldsymbol{\phi}; y)$ by adding the LIST debugging option to the PROC NLMIXED statement.

$Y \sim \text{normal}(m, v)$

$$
\begin{aligned}
l(m, v; y) &= -\frac{1}{2}\left(\log\{2\pi\} + \frac{(y-m)^2}{v} + \log\{v\}\right) \\
\mathrm{E}[Y] &= m \\
\mathrm{Var}[Y] &= v \\
v &> 0
\end{aligned}
$$

$Y \sim \text{binary}(p)$

$$
\begin{aligned}
l_1(p; y) &= \begin{cases} y \ \log\{p\} & y > 0 \\ 0 & \text{otherwise} \end{cases} \\
l_2(p; y) &= \begin{cases} (1-y) \ \log\{1-p\} & y < 1 \\ 0 & \text{otherwise} \end{cases} \\
l(p; y) &= l_1(p; y) + l_2(p; y) \\
\mathrm{E}[Y] &= p \\
\mathrm{Var}[Y] &= p\,(1-p) \\
0 &< p < 1
\end{aligned}
$$

$Y \sim \text{binomial}(n, p)$

$$
\begin{aligned}
l_c &= \log\{\Gamma(n+1)\} - \log\{\Gamma(y+1)\} - \log\{\Gamma(n-y+1)\} \\
l_1(n, p; y) &= \begin{cases} y \ \log\{p\} & y > 0 \\ 0 & \text{otherwise} \end{cases} \\
l_2(n, p; y) &= \begin{cases} (n-y) \ \log\{1-p\} & n-y > 0 \\ 0 & \text{otherwise} \end{cases} \\
l(n, p; y) &= l_c + l_1(n, p; y) + l_2(n, p; y) \\
\mathrm{E}[Y] &= n\,p \\
\mathrm{Var}[Y] &= n\,p\,(1-p) \\
0 &< p < 1
\end{aligned}
$$

$Y \sim \text{gamma}(a, b)$

$$
\begin{aligned}
l(a, b; y) &= -a \log\{b\} - \log\{\Gamma(a)\} + (a-1)\log\{y\} - y/b \\
\text{E}[Y] &= ab \\
\text{Var}[Y] &= ab^2 \\
a &> 0 \\
b &> 0
\end{aligned}
$$

This parameterization of the gamma distribution differs from the parameterization used in the GLIMMIX and GENMOD procedures. The following statements show the equivalent reparameterization in the NLMIXED procedure that fits a generalized linear model for gamma-distributed data in the parameterization of the GLIMMIX procedure:

```
proc glimmix;
   model y = x / dist=gamma s;
run;


proc nlmixed;
   parms b0=1 b1=0 scale=14;
   linp = b0 + b1*x;
   mu   = exp(linp);
   b    = mu/scale;
   model y ~ gamma(scale,b);
run;
```

$Y \sim \text{negbin}(n, p)$

$$
\begin{aligned}
l(n, p; y) &= \log\{\Gamma(n+y)\} - \log\{\Gamma(n)\} - \log\{\Gamma(y+1)\} \\
&\quad + n\log\{p\} + y\log\{1-p\} \\
\text{E}[Y] &= nP = n\left(\frac{1-p}{p}\right) \\
\text{Var}[Y] &= nP(1-P) = n\left(\frac{1-p}{p}\right)\frac{1}{p} \\
n &\geq 0 \\
0 &< p < 1
\end{aligned}
$$

This form of the negative binomial distribution is one of the many parameterizations in which the mass function or log-likelihood function appears. Another common parameterization uses

$$
\begin{aligned}
l(n, p; y) &= \log\{\Gamma(n+y)\} - \log\{\Gamma(n)\} - \log\{\Gamma(y+1)\} \\
&\quad + n\log\{1 - P/(1+P)\} + y\log\{P/(1+P)\}
\end{aligned}
$$

with $P = (1-p)/p$, $P > 0$.

Note that the parameter n can be real-numbered; it does not have to be integer-valued. The parameterization of the negative binomial distribution in the NLMIXED procedure differs from that in the GLIMMIX and GENMOD procedures. The following statements show the equivalent formulations for maximum likelihood estimation in the GLIMMIX and NLMIXED procedures in a negative binomial regression model:

```
proc glimmix;
   model y = x / dist=negbin s;
run;


proc nlmixed;
   parms b0=3, b1=1, k=0.8;
   linp = b0 + b1*x;
   mu = exp(linp);
   p  = 1/(1+mu*k);
   model y ~ negbin(1/k,p);
run;
```

$Y \sim \text{Poisson}(m)$

$$
\begin{aligned}
l(m; y) &= y \log\{m\} - m - \log\{\Gamma(y+1)\} \\
\text{E}[Y] &= m \\
\text{Var}[Y] &= m \\
m &> 0
\end{aligned}
$$

Compartment Models

Pharmacokinetics (PK) is a branch of medicine that models the movement of a drug through the body (Gabrielsson and Weiner 2006). PK is sometimes referred to as the study of what the body does to a drug. Compartment models are basic building blocks of PK models. In a study, a body is divided in to several *compartments*, groups of organs or tissues that are kinetically homogeneous. The main interest of PK is to model how a drug moves through these compartments—for example, to estimate the amount of a drug and further concentrations of the drug that are present in a compartment at any given time. The concentrations of the drug, over time, are typically modeled via a set of differential equations that depend on a variety of variables, such as the amount of drug given, elimination rates, transfer rates, and so on. The sets of differential equations can also depend on how the drug is administered to the body, distributed through the body, and eliminated from the body. These models are known as compartment models, which are divided based on the number of compartments. Not all analytical solutions to multiple-compartment differential equations models are known, but Abuhelwa, Foster, and Upton (2015) and Fisher and Shafer (2007) provide closed-form solutions for those that correspond to the one-, two-, and three- compartments. These are the models that are handled by the CMPTMODEL statement.

Routes of Drug Administration

There are three types of drug administration methods: intravenous bolus, intravenous infusion, and extravascular dose administrations. A bolus medication typically has no or a very short time lag for the drug to enter a compartment. Intravenous infusions are given over a period of time, and the drug enters into the body at a constant rate. In infusion, the two quantities, *rate* and *duration* of the infusion, are of interest and are used in calculating the amount of drug present in the body. The rate and the duration of the infusion are related, so knowing one determines the other. For compartment models that have infusion type of administration, it is sufficient to provide either the rate or the duration information.

One-, Two-, and Three-Compartment Models for Intravenous Administration

This section lists a number of assumptions that are made on the set of compartment models that the CMPTMODEL statement supports. The one-, two-, and three-compartment models all have a central compartment and can have one or more peripheral compartments that are linked only to the central compartment but not to each other. After a drug is administered to an administration site, it is distributed to the central compartment and then to other peripheral compartments. The rates at which the drug moves from the central compartment to and from the other peripheral compartments are characterized by transfer rate constants. The rates at which the drug is eliminated from the central or the peripheral compartments are characterized by elimination rate constants.

Schematic representation of the one-, two-, and three- compartment models are given in Figure 89.12, Figure 89.13, and Figure 89.14, respectively. In each figure, Dose is the dosage of the drug that is given intravenously, and k_{10} represents the rate at which the drug is eliminated from compartment 1. In Figure 89.13 and Figure 89.14, k_{12} represents the transfer rate constant from compartment 1 to compartment 2 and k_{21} represents the transfer rate constant from compartment 2 to compartment 1. Similarly, in Figure 89.14, k_{13} represents the transfer rate constant from compartment 1 to compartment 3 and k_{31} represents the transfer rate constant from compartment 3 to compartment 1. In these scenarios, compartment 1 is the central compartment and compartments 2 and 3 are the peripheral compartments. In compartment models, a drug can be administered only to and eliminated only from the central compartment.

Figure 89.12 One-Compartment Model

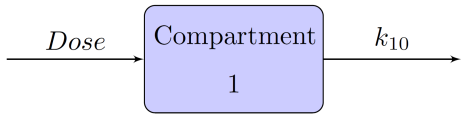

Figure 89.13 Two-Compartment Model

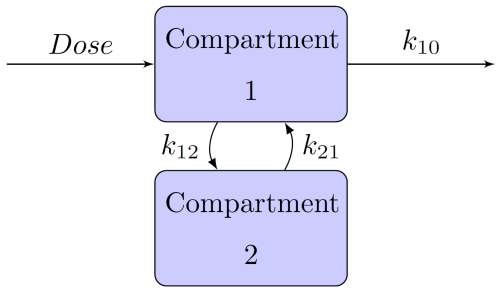

Figure 89.14 Three-Compartment Model

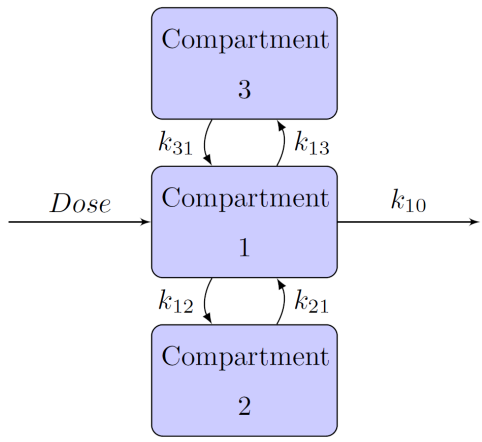

One-, Two-, and Three-Compartment Models for Extravascular Administration

Extravascular administrations, such as oral administration, are different from bolus and infusion in the sense that there is an absorption phase before the drug enters the central compartment. Schematic representation of the one-, two-, and three- compartment models when the drug is administered using an oral method are given in Figure 89.15, Figure 89.16, and Figure 89.17, respectively. In these models, compartment 0 is also called the *depot* compartment.

Figure 89.15 One-Compartment Model with Absorption Phase

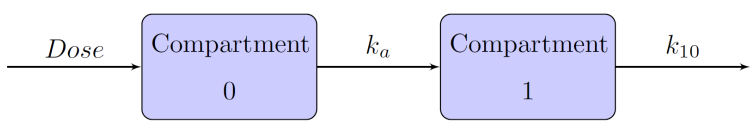

Figure 89.16 Two-Compartment Model with Absorption Phase

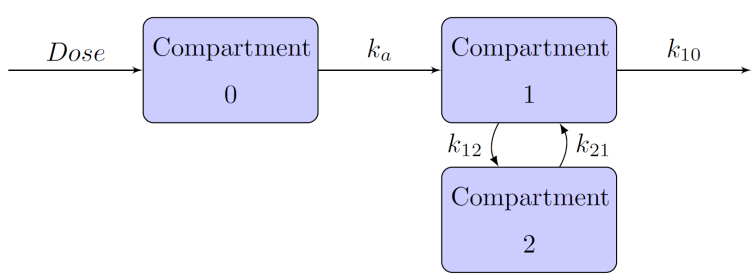

Figure 89.17 Three-Compartment Model with Absorption Phase

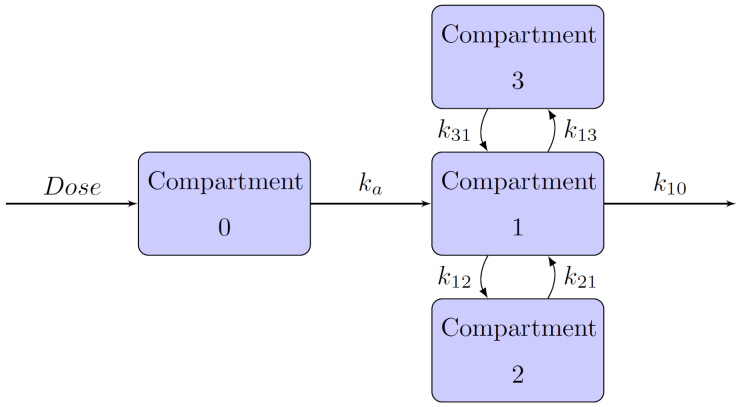

Compartment Models Specification

The CMPTMODEL statement in PROC NLMIXED enables you to fit one-, two-, and three-compartment models that use bolus, infusion, or oral types of drug administration. The CMPTMODEL statement computes the predicted concentrations of the drug for all compartments at each time point.

The CMPTMODEL statement supports a large number of options, which are grouped into three types: required, conditionally-required, and optional options. You use the required options to specify the type of compartment model that you want to fit (the NCOMPS=, ADMTYPE=, and PARMTYPE= options) and to provide the input time variable (in the TIME= option) and the outcome variable (in the PCONC= option) that is the predicted concentration in the first (central) compartment at each time point.

The statement supports the following nine types of compartment models:

1. one-compartment model with bolus dose administration
2. one-compartment model with infusion type of dose administration
3. one-compartment model with oral dose administration
4. two-compartment model with bolus dose administration
5. two-compartment model with infusion type of dose administration
6. two-compartment model with oral dose administration
7. three-compartment model with bolus dose administration
8. three-compartment model with infusion type of dose administration
9. three-compartment model with oral dose administration.

Each of these nine models can be parameterized in two ways: in terms of elimination and transfer rate constants of each compartment, or in terms of clearance and volume parameters of each compartment. The relationship between rate constants and the clearance and volumes of each compartment are:

$$K_{10} = CL1/VOL1$$
$$K_{12} = CL2/VOL1$$
$$K_{13} = CL3/VOL1$$
$$K_{21} = CL2/VOL2$$
$$K_{31} = CL3/VOL3$$

where CL*n* and VOL*n* are the clearance and volume of the *n*th compartment for *n*=1,2,3. You can use the PARMTYPE= option in the CMPTMODEL statement to specify these parameterizations. In total, you can specify 18 types of compartment models by using combinations of the NCOMPS=, ADMTYPE=, and PARMTYPE= options in the CMPTMODEL statement.

The conditionally required options define the specifics needed in some of the compartment models. For example, if you specify a two-compartment model with a bolus type of drug administration in terms of rate constants by using NCOMPS=2, ADMTYPE=IVB, and PARMTYPE=1, then you must specify the K12=, K21=, and K10= options. Here is an example:

```
PROC NLMIXED data=TwoComp;

...other statements...

erate10= exp(beta1);
trate12= exp(beta2);
trate21= exp(beta3);

CMPTMODEL ncomps=2 admtype=ivb parmtype=1 pconc=pred time=time
          k10=erate10 k12=trate12 k21=trate21
          dose1=ddose scale1=vol/1000;
model conc ~ normal(pred,s2);

...other statements...

run;
```

The K12= and K21= options specify the transfer rate constants, and the K10= option specifies the elimination rate constant from the central compartment. In addition, you can specify dosage information (DOSE*n*=) or scaling information (SCALE*n*=) in the syntax. For more information, see the section "CMPTMODEL Statement" on page 7267.

Table 89.2, Table 89.3, and Table 89.4 list all conditionally required options and valid optional options for each of the 18 compartment models.

Table 89.2 One-Compartment Models

Model	Required	Conditionally Required	Optional
One compartment with bolus dose	NCOMPS=1 ADMTYPE=IVB PARMTYPE=1	K10=	DOSE1= SCALE1=
One compartment with bolus dose	NCOMPS=1 ADMTYPE=IVB PARMTYPE=2	CL1= VOL1=	DOSE1= SCALE1=
One compartment with infusion dose	NCOMPS=1 ADMTYPE=INF PARMTYPE=1	K10= RATE= (or) DURN=	DOSE1= SCALE1=
One compartment with infusion dose	NCOMPS=1 ADMTYPE=INF PARMTYPE=2	CL1= VOL1= RATE= (or) DURN=	DOSE1= SCALE1=
One compartment with oral dose	NCOMPS=1 ADMTYPE=ORAL PARMTYPE=1	K10= Ka=	DOSE1= SCALE1=
One compartment with oral dose	NCOMPS=1 ADMTYPE=ORAL PARMTYPE=2	CL1= VOL1= Ka=	DOSE1= SCALE1=

Table 89.3 Two-Compartment Models

Model	Required	Conditionally Required	Optional
Two compartments with bolus dose	NCOMPS=2 ADMTYPE=IVB PARMTYPE=1	K10= K12= K21=	DOSE1= DOSE2= SCALE1= SCALE2= K20= PCONC2=
Two compartments with bolus dose	NCOMPS=2 ADMTYPE=IVB PARMTYPE=2	CL1= VOL1= CL2= VOL2=	DOSE1= DOSE2= SCALE1= SCALE2= K20= PCONC2=
Two compartments with infusion dose	NCOMPS=2 ADMTYPE=INF PARMTYPE=1	K10= K12= K21= RATE= (or) DURN=	DOSE1= SCALE1=
Two compartments with infusion dose	NCOMPS=2 ADMTYPE=INF PARMTYPE=2	CL1= VOL1= CL2= VOL2= RATE= (or) DURN=	DOSE1= SCALE1=
Two compartments with oral dose	NCOMPS=2 ADMTYPE=ORAL PARMTYPE=1	K10= K12= K21= Ka=	DOSE1= DOSE2= SCALE1= SCALE2= K20= PCONC2=
Two compartments with oral dose	NCOMPS=2 ADMTYPE=ORAL PARMTYPE=2	CL1= VOL1= CL2= VOL2= Ka=	DOSE1= DOSE2= SCALE1= SCALE2= K20= PCONC2=

Table 89.4 Three-Compartment Models

Model	Required	Conditionally Required	Optional
Three compartments with bolus dose	NCOMPS=3 ADMTYPE=IVB PARMTYPE=1	K10= K12= K21= K13= K31=	DOSE1= DOSE2= DOSE3= SCALE1= SCALE2= SCALE3= K20= K30= PCONC2= PCONC3=
Three compartments with bolus dose	NCOMPS=3 ADMTYPE=IVB PARMTYPE=2	CL1= VOL1= CL2= VOL2= CL3= VOL3=	DOSE1= DOSE2= DOSE3= SCALE1= SCALE2= SCALE3= K20= K30= PCONC2= PCONC3=
Three compartments with infusion dose	NCOMPS=3 ADMTYPE=INF PARMTYPE=1	K10= K12= K21= K13= K31= RATE= (or) DURN=	DOSE1= SCALE1=
Three compartments with infusion dose	NCOMPS=3 ADMTYPE=INF PARMTYPE=2	CL1= VOL1= CL2= VOL2= CL3= VOL3= RATE= (or) DURN=	DOSE1= SCALE1=
Three compartments with oral dose	NCOMPS=3 ADMTYPE=ORAL PARMTYPE=1	K10= K12= K21= K13= K31= Ka=	DOSE1= DOSE2= DOSE3= SCALE1= SCALE2= SCALE3= K20= K30= PCONC2= PCONC3=
Three compartments with oral dose	NCOMPS=3 ADMTYPE=ORAL PARMTYPE=2	CL1= VOL1= CL2= VOL2= CL3= VOL3= Ka=	DOSE1= DOSE2= DOSE3= SCALE1= SCALE2= SCALE3= K20= K30= PCONC2= PCONC3=

One-Compartment Model with an Oral Dose Administration

Consider the example "Example 89.1: One-Compartment Model with Pharmacokinetic Data" on page 7315, which studies the dispersion of the theophylline drug through a living individual. In this example, Pinheiro and Bates (1995) considered an one-compartment model with an oral dose. The paper mentions the solution to the one-compartment model. Instead of using the explicit solution, you can use the CMPTMODEL statement to fit the same model as follows:

```
proc nlmixed data=theoph;
   parms beta1=-3.22 beta2=0.47 beta3=-2.45
   s2b1 =0.03 cb12 =0 s2b2 =0.4 s2=0.5;
   cl = exp(beta1 + b1);
   ka = exp(beta2 + b2);
```

```
    ke = exp(beta3);
    vl = cl/ke;
    CMPTMODEL ncomps=1 admtype=oral time=time pconc=predConc
              parmtype=1  ka=ka k10=ke dose0=dose scale1=vl;
    model conc ~ normal(predConc,s2);
    random b1 b2 ~ normal([0,0],[s2b1,cb12,s2b2]) subject=subject;
run;
```

The NCOMPS=1 and ADMTYPE=ORAL options specify a one-compartment model with oral administration. Time is a data set variable that indicates time, and predConc is an outcome variable that contains predicted concentration. The PARMTYPE=1 option requests the compartment model using absorption and elimination rate constants, with KA=ka and K10=ke indicating the absorption and elimination rate constants, respectively. The dosage value for each patient in the depot compartment is specified in DOSE0=option, where dose is a data set variable. Lastly, SCALE1=vl scales the predicted concentrations by vl.

Multiple Doses

In PK field experiments or observational studies, a patient can often receive a drug multiple times, either continually or periodically over a period of time. The patient can also receive multiple types of drug in a study. For example, a patient might receive a bolus injection in the morning and an infusion drug at a constant rate in the evening. Calculation of predicted concentration in each compartment in the presence of these multiple doses or multiple types of dosing is different from single-dosing compartment models. For differential equations of a number of multiple-dosage scenarios and solutions to predicted concentration for the central compartment, see Gabrielsson and Weiner (2006).

The CMPTMODEL statement handles multiple doses in one-, two-, or three-compartment models and computes predicted concentrations in the central compartment. The syntax specification does not change from single-dose models, and the structure and content of the input data set are understood by the statement to fit various types of multiple-dosage models.

Within the biopharmaceutical industry, data in multiple-dosage studies are often structured by following a convention that has been popularized by NONMEM software (Beal et al. 2011). This convention is used to name variables and specify variable values; see Owen and Fiedler-Kelly (2014). If your data are available in a SAS data set that follows this convention, the data set must be converted to a SAS data set that is suitable to be analyzed using PROC NLMIXED. You can use the autocall macro %PKCONVRT for this purpose.

The %PKCONVRT macro takes two arguments: an input data set (specified by the first argument) and an output data set (specified by the second argument). The organization of the input data set should follow the convention. The output data set can serve as the DATA= data set for PROC NLMIXED to fit single-dose or multiple-dose models. Here is a simple example:

```
data pk_ex;
input ID  TIME  AMT DV  EVID;
datalines;
1     0.00    60000        .       1
1     0.20    50000        .       1
1     0.25        0    1126.1      0
1     0.50        0     869.9      0
1     0.75        0     883.6      0
1     1.00        0    1244.0      0
1     1.50        0     995.2      0
2     0.00    70000        .       1
```

```
2     0.25        0    1126.1      0
2     0.50        0     869.9      0
2     0.75        0     883.6      0
2     1.00        0    1244.0      0

   .... more lines ...
;
run;

%pkconvrt(data=pk_ex, out=out_ex);
```

If you want to fit a two-compartment model for the preceding multiple dosage data, use the following NLMIXED program with the out_ex data set that is produced from the %PKCONVRT macro:

```
proc nlmixed data=out_ex;
   parms beta1=2 beta2=2 beta3=2 beta4=2 s2=0.6
         s2b1=0.4 s2b2=0.4 s2b3=0.4 s2b4=0.4 cb21=0.001
         cb31=0.001 cb32=0.001 cb41=0.001 cb42=0.001 cb43=0.001;
   bounds s2b1>0, s2b2 >0, s2b3>0, s2b4>0, s2>0;
   cl1 = exp(beta1+b1);
   vl1 = exp(beta2+b2);
   cl2 = exp(beta3+b3);
   vl2 = exp(beta4+b4);
   CMPTMODEL ncomps=2 admtype=ivb parmtype=2 pconc=pred time=time
             vol1=vl1 cl1=cl1 vol2=vl2 cl2=cl2
             scale1=vl1;
   model conc ~ normal(pred,s2);
   random b1 b2 b3 b4 ~ normal([0,0,0,0],[s2b1,
                                          cb21,s2b2,
                                          cb31,cb32,s2b3,
                                          cb41,cb42,cb43,s2b4]) subject = id;
run;
```

The CMPTMODEL statement computes the predictions only for the central compartment for multiple dosage data. Therefore, options PCONC2=, PCONC3=, and PCONC0= are not valid in a multiple dose model. In addition, the K20=, K30=, DOSE2=, DOSE3=, SCALE2=, SCALE3=, SCALE0= options are ignored.

The %PKCONVRT autocall macro computes the elapsed time (from the time that a dose is administered to the time that a concentration is measured) in scenarios for both single and multiple continuous doses. The macro writes the computed time in the data set that is specified in the OUT= option. This data set also contains all the dosage information for all the time points at which the concentrations are measured. When you specify this data set in the PROC NLMIXED statement and you specify a CMPTMODEL statement, the elapsed times and the dosage information from the data set are used directly in computing the predicted concentrations in the central compartment. As a result, the CMPTMODEL statement ignores any specified DOSE1= and TIME= options. Similarly, for absorption models (ADMTYPE = ORAL), the DOSE0= and TIME= options in the CMPTMODEL statement are ignored and overwritten with the elapsed times and the dosage information from the OUT= data set. For examples that use the %PKCONVRT autocall macro and the CMPTMODEL statement to fit various compartment models, see Kurada and Chen (2018).

In addition, the CMPTMODEL statement supports steady-state dosing scenarios (Owen and Fiedler-Kelly 2014) in one-, two-, and three-compartment models. The %PKCONVRT macro keeps track of the elapsed time and dosing information that are required in the CMPTMODEL statement to fit steady-state dose models. The %PKCONVRT autocall macro supports input data sets that have combinations of steady-state

and regular dosing, single and multiple dosing. For example, your data set can contain dosing history information on patients who receive multiple regular bolus doses and others who receive a single steady-state dose. When you use the output data set from the %PKCONVRT macro as input to PROC NLMIXED, the CMPTMODEL statement properly interprets all these distinct scenarios and computes the predicted concentrations accordingly.

The %PKCONVRT macro creates new variables (such as the _CMPT_ variable) in the OUT= data set. According to SAS conventions, it is best to avoid having variables whose name begins with an underscore _ in the DATA = data set. Also programming variables in the NLMIXED program that start with _CMPT_ are reserved for the internal computations; hence, avoid programming variables names that begin with _CMPT_ .

Hierarchical Model Specification

PROC NLMIXED supports multiple RANDOM statements to accommodate nested multilevel nonlinear mixed models.If you use multiple RANDOM statements, PROC NLMIXED assumes that the SUBJECT= variable from each RANDOM statement forms a containment hierarchy. In the containment hierarchy, each SUBJECT= variable is contained by another SUBJECT= variable, and the SUBJECT= variable that is contained by all SUBJECT= variables is considered "the" SUBJECT= variable. For example, consider the following three-level nested model that has three SUBJECT= variables:

A	B	C
1	1	1
1	1	2
1	2	1
1	2	2
2	1	1
2	1	2
2	1	3
2	2	1
2	2	2

Suppose you specify a three-level nested model by using the following three RANDOM statements:

```
random r11 ~ normal(0,sd1) subject = A;
random r21 ~ normal(0,sd2) subject = B(A);
random r31 ~ normal(0,sd3) subject = C(A*B);
```

Then PROC NLMIXED assumes the containment hierarchy as follows. The first-level hierarchy is defined using "the" SUBJECT= variable A. Similarly, the second-level hierarchy is defined using SUBJECT= variable B, which is nested within first level. Finally, SUBJECT= variable C defines the third-level hierarchy that is nested within both the first and second levels. In short, the SUBJECT= variable A is "the" SUBJECT= variable. B is contained in A, and C is contained in both A and B. In this example, there are two first-level subjects that are determined using "the" SUBJECT= variable A.

Based on the preceding hierarchy specification, PROC NLMIXED's indexing of nested subjects for each first-level subject can be visualized as follows:

$$\overbrace{\overbrace{\overbrace{C_{1(11)}C_{2(11)}}^{B_{1(1)}}\overbrace{C_{1(12)}C_{2(12)}}^{B_{2(1)}}}^{A_1}\overbrace{\overbrace{C_{1(21)}C_{2(21)}C_{3(21)}}^{B_{1(2)}}\overbrace{C_{1(22)}C_{2(22)}}^{B_{2(2)}}}^{A_2}}$$

The ith subject from the first level is denoted as A_i, the second-level nested subjects are denoted as $\mathsf{B}_{j(i)}$, and the third-level nested subjects are denoted as $\mathsf{C}_{k(ij)}$.

You can specify any nested structure by using the SUBJECT= syntax in PROC NLMIXED. For example, using the following three RANDOM statements, PROC NLMIXED fits a different model:

```
random r11 ~ normal(0,sd1) subject = C;
random r21 ~ normal(0,sd2) subject = B(C);
random r31 ~ normal(0,sd3) subject = A(C*B);
```

In this case, PROC NLMIXED processes the subjects by using SUBJECT= variable C as "the" SUBJECT= variable, and the containment hierarchy is changed as follows:

A	B	C
1	1	1
1	1	2
1	2	1
1	2	2
2	1	1
2	1	2
2	1	3
2	2	1
2	2	2

$\Rightarrow$

C	B	A
1	1	1
1	1	2
1	2	1
1	2	2
2	1	1
2	1	2
2	2	1
2	2	2
3	1	2

Again, PROC NLMIXED's indexing of the nested subjects in this containment hierarchy can be visualized as follows:

$$\overbrace{\overbrace{\overbrace{A_{1(11)}A_{2(11)}}^{B_{1(1)}}\overbrace{A_{1(12)}A_{2(12)}}^{B_{2(1)}}}^{C_1}\overbrace{\overbrace{A_{1(21)}A_{2(21)}}^{B_{1(2)}}\overbrace{A_{1(22)}A_{2(22)}}^{B_{2(2)}}}^{C_2}\overbrace{\overbrace{A_{1(31)}}^{B_{1(3)}}}^{C_3}}$$

Here, PROC NLMIXED assumes that C is "the" SUBJECT= variable. B is contained in C, and A is contained in C and B. In this case, there are three first-level subjects that are determined using "the" SUBJECT= variable C.

As explained before, in this case, the ith subject from the first level is denoted as C_i, the second-level nested subjects are denoted as $\mathsf{B}_{j(i)}$ and for third level, the nested subjects are denoted as $\mathsf{A}_{k(ij)}$.

Note that the containment hierarchy could potentially create more subjects than the unique number of subjects. For example, consider the following table, in which A is "the" SUBJECT= variable and B is nested within the subject A:

A	B
a	1
a	2
b	1
b	2
c	1
c	2

Even though the SUBJECT = B variable has only two unique subjects (1 and 2), when the containment hierarchy that is specified along with B is nested within A, PROC NLMIXED creates six nested B subjects. These nested subjects can be denoted as 1(a), 2(a), 1(b), 2(b), 1(c), and 2(c).

PROC NLMIXED does not support noncontainment hierarchy (or non-nested) models. For example, the following statements are not supported, because subject C is not nested within B and A:

```
random r11 ~ normal(0,sd1) subject = A;
random r21 ~ normal(0,sd2) subject = B(A);
random r31 ~ normal(0,sd3) subject = C;
```

Optimization Algorithms

There are several optimization techniques available in PROC NLMIXED. You can choose a particular optimizer with the TECH= option in the PROC NLMIXED statement.

Algorithm	TECH=
trust region method	TRUREG
Newton-Raphson method with line search	NEWRAP
Newton-Raphson method with ridging	NRRIDG
quasi-Newton methods (DBFGS, DDFP, BFGS, DFP)	QUANEW
double-dogleg method (DBFGS, DDFP)	DBLDOG
conjugate gradient methods (PB, FR, PR, CD)	CONGRA
Nelder-Mead simplex method	NMSIMP

No algorithm for optimizing general nonlinear functions exists that always finds the global optimum for a general nonlinear minimization problem in a reasonable amount of time. Since no single optimization technique is invariably superior to others, PROC NLMIXED provides a variety of optimization techniques that work well in various circumstances. However, you can devise problems for which none of the techniques in PROC NLMIXED can find the correct solution. Moreover, nonlinear optimization can be computationally expensive in terms of time and memory, so you must be careful when matching an algorithm to a problem.

All optimization techniques in PROC NLMIXED use $O(n^2)$ memory except the conjugate gradient methods, which use only $O(n)$ of memory and are designed to optimize problems with many parameters. Since the techniques are iterative, they require the repeated computation of the following:

- the function value (optimization criterion)

- the gradient vector (first-order partial derivatives)
- for some techniques, the (approximate) Hessian matrix (second-order partial derivatives)

However, since each of the optimizers requires different derivatives, some computational efficiencies can be gained. The following table shows, for each optimization technique, which derivatives are required (FOD: first-order derivatives; SOD: second-order derivatives).

Algorithm	FOD	SOD
TRUREG	x	x
NEWRAP	x	x
NRRIDG	x	x
QUANEW	x	-
DBLDOG	x	-
CONGRA	x	-
NMSIMP	-	-

Each optimization method employs one or more convergence criteria that determine when it has converged. The various termination criteria are listed and described in the "PROC NLMIXED Statement" section. An algorithm is considered to have converged when any one of the convergence criteria is satisfied. For example, under the default settings, the QUANEW algorithm will converge if ABSGCONV< 1E–5, FCONV< $10^{-\text{FDIGITS}}$, or GCONV< 1E–8.

Choosing an Optimization Algorithm

The factors that go into choosing a particular optimization technique for a particular problem are complex and can involve trial and error.

For many optimization problems, computing the gradient takes more computer time than computing the function value, and computing the Hessian sometimes takes *much* more computer time and memory than computing the gradient, especially when there are many decision variables. Unfortunately, optimization techniques that do not use some kind of Hessian approximation usually require many more iterations than techniques that do use a Hessian matrix, and as a result the total run time of these techniques is often longer. Techniques that do not use the Hessian also tend to be less reliable. For example, they can more easily terminate at stationary points rather than at global optima.

A few general remarks about the various optimization techniques follow:

- The second-derivative methods TRUREG, NEWRAP, and NRRIDG are best for small problems where the Hessian matrix is not expensive to compute. Sometimes the NRRIDG algorithm can be faster than the TRUREG algorithm, but TRUREG can be more stable. The NRRIDG algorithm requires only one matrix with $n(n+1)/2$ double words; TRUREG and NEWRAP require two such matrices.
- The first-derivative methods QUANEW and DBLDOG are best for medium-sized problems where the objective function and the gradient are much faster to evaluate than the Hessian. The QUANEW and DBLDOG algorithms, in general, require more iterations than TRUREG, NRRIDG, and NEWRAP, but each iteration can be much faster. The QUANEW and DBLDOG algorithms require only the gradient to update an approximate Hessian, and they require slightly less memory than TRUREG or NEWRAP (essentially one matrix with $n(n+1)/2$ double words). QUANEW is the default optimization method.

- The first-derivative method CONGRA is best for large problems where the objective function and the gradient can be computed much faster than the Hessian and where too much memory is required to store the (approximate) Hessian. The CONGRA algorithm, in general, requires more iterations than QUANEW or DBLDOG, but each iteration can be much faster. Since CONGRA requires only a factor of n double-word memory, many large applications of PROC NLMIXED can be solved only by CONGRA.
- The no-derivative method NMSIMP is best for small problems where derivatives are not continuous or are very difficult to compute.

Algorithm Descriptions

Some details about the optimization techniques follow.

Trust Region Optimization (TRUREG)

The trust region method uses the gradient $\mathbf{g}(\boldsymbol{\theta}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\theta}^{(k)})$; thus, it requires that the objective function $f(\boldsymbol{\theta})$ have continuous first- and second-order derivatives inside the feasible region.

The trust region method iteratively optimizes a quadratic approximation to the nonlinear objective function within a hyperelliptic trust region with radius Δ that constrains the step size corresponding to the quality of the quadratic approximation. The trust region method is implemented using: Dennis, Gay, and Welsch (1981); Gay (1983); Moré and Sorensen (1983).

The trust region method performs well for small- to medium-sized problems, and it does not need many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the (dual) quasi-Newton or conjugate gradient algorithms might be more efficient.

Newton-Raphson Optimization with Line Search (NEWRAP)

The NEWRAP technique uses the gradient $\mathbf{g}(\boldsymbol{\theta}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\theta}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region. If second-order derivatives are computed efficiently and precisely, the NEWRAP method can perform well for medium-sized to large problems, and it does not need many function, gradient, and Hessian calls.

This algorithm uses a pure Newton step when the Hessian is positive definite and when the Newton step reduces the value of the objective function successfully. Otherwise, a combination of ridging and line search is performed to compute successful steps. If the Hessian is not positive definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive definite (Eskow and Schnabel 1991).

In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation (LINESEARCH=2).

Newton-Raphson Ridge Optimization (NRRIDG)

The NRRIDG technique uses the gradient $\mathbf{g}(\boldsymbol{\theta}^{(k)})$ and the Hessian matrix $\mathbf{H}(\boldsymbol{\theta}^{(k)})$; thus, it requires that the objective function have continuous first- and second-order derivatives inside the feasible region.

This algorithm uses a pure Newton step when the Hessian is positive definite and when the Newton step reduces the value of the objective function successfully. If at least one of these two conditions is not satisfied, a multiple of the identity matrix is added to the Hessian matrix.

The NRRIDG method performs well for small- to medium-sized problems, and it does not require many function, gradient, and Hessian calls. However, if the computation of the Hessian matrix is computationally expensive, one of the (dual) quasi-Newton or conjugate gradient algorithms might be more efficient.

Since the NRRIDG technique uses an orthogonal decomposition of the approximate Hessian, each iteration of NRRIDG can be slower than that of the NEWRAP technique, which works with Cholesky decomposition. Usually, however, NRRIDG requires fewer iterations than NEWRAP.

Quasi-Newton Optimization (QUANEW)

The (dual) quasi-Newton method uses the gradient $\mathbf{g}(\boldsymbol{\theta}^{(k)})$, and it does not need to compute second-order derivatives since they are approximated. It works well for medium to moderately large optimization problems where the objective function and the gradient are much faster to compute than the Hessian; but, in general, it requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which compute second-order derivatives. QUANEW is the default optimization algorithm because it provides an appropriate balance between the speed and stability required for most nonlinear mixed model applications.

The QUANEW technique is one of the following, depending on the value of the UPDATE= option.

- the original quasi-Newton algorithm, which updates an approximation of the inverse Hessian
- the dual quasi-Newton algorithm, which updates the Cholesky factor of an approximate Hessian (default)

You can specify four update formulas with the UPDATE= option:

- DBFGS performs the dual Broyden, Fletcher, Goldfarb, and Shanno (BFGS) update of the Cholesky factor of the Hessian matrix. This is the default.
- DDFP performs the dual Davidon, Fletcher, and Powell (DFP) update of the Cholesky factor of the Hessian matrix.
- BFGS performs the original BFGS update of the inverse Hessian matrix.
- DFP performs the original DFP update of the inverse Hessian matrix.

In each iteration, a line search is performed along the search direction to find an approximate optimum. The default line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α satisfying the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit of the step size. Violating the left-side Goldstein condition can affect the positive definiteness of the quasi-Newton update. In that case, either the update is skipped or the iterations are restarted with an identity matrix, resulting in the steepest descent or ascent search direction. You can specify line-search algorithms other than the default with the LINESEARCH= option.

The QUANEW algorithm uses its own line-search technique. No options and parameters (except the INSTEP= option) controlling the line search in the other algorithms apply here. In several applications, large steps in the first iterations are troublesome. You can use the INSTEP= option to impose an upper bound for the step size α during the first five iterations. You can also use the INHESSIAN=*r* option to specify a different starting approximation for the Hessian. If you specify only the INHESSIAN option, the Cholesky factor of a (possibly ridged) finite difference approximation of the Hessian is used to initialize the quasi-Newton update process. The values of the LCSINGULAR=, LCEPSILON=, and LCDEACT= options, which control the

processing of linear and boundary constraints, are valid only for the quadratic programming subroutine used in each iteration of the QUANEW algorithm.

Double-Dogleg Optimization (DBLDOG)

The double-dogleg optimization method combines the ideas of the quasi-Newton and trust region methods. In each iteration, the double-dogleg algorithm computes the step $\mathrm{s}^{(k)}$ as the linear combination of the steepest descent or ascent search direction $\mathrm{s}_1^{(k)}$ and a quasi-Newton search direction $\mathrm{s}_2^{(k)}$:

$$\mathrm{s}^{(k)} = \alpha_1 \mathrm{s}_1^{(k)} + \alpha_2 \mathrm{s}_2^{(k)}$$

The step is requested to remain within a prespecified trust region radius; see Fletcher (1987, p. 107). Thus, the DBLDOG subroutine uses the dual quasi-Newton update but does not perform a line search. You can specify two update formulas with the UPDATE= option:

- DBFGS performs the dual Broyden, Fletcher, Goldfarb, and Shanno update of the Cholesky factor of the Hessian matrix. This is the default.
- DDFP performs the dual Davidon, Fletcher, and Powell update of the Cholesky factor of the Hessian matrix.

The double-dogleg optimization technique works well for medium to moderately large optimization problems where the objective function and the gradient are much faster to compute than the Hessian. The implementation is based on Dennis and Mei (1979) and Gay (1983), but it is extended for dealing with boundary and linear constraints. The DBLDOG technique generally requires more iterations than the TRUREG, NEWRAP, and NRRIDG techniques, which require second-order derivatives; however, each of the DBLDOG iterations is computationally cheap. Furthermore, the DBLDOG technique requires only gradient calls for the update of the Cholesky factor of an approximate Hessian.

Conjugate Gradient Optimization (CONGRA)

Second-order derivatives are not required by the CONGRA algorithm and are not even approximated. The CONGRA algorithm can be expensive in function and gradient calls, but it requires only $O(n)$ memory for unconstrained optimization. In general, many iterations are required to obtain a precise solution, but each of the CONGRA iterations is computationally cheap. You can specify four different update formulas for generating the conjugate directions by using the UPDATE= option:

- PB performs the automatic restart update method of Powell (1977) and Beale (1972). This is the default.
- FR performs the Fletcher-Reeves update (Fletcher 1987).
- PR performs the Polak-Ribiere update (Fletcher 1987).
- CD performs a conjugate-descent update of Fletcher (1987).

The default, UPDATE=PB, behaved best in most test examples. You are advised to avoid the option UPDATE=CD, which behaved worst in most test examples.

The CONGRA subroutine should be used for optimization problems with large n. For the unconstrained or boundary constrained case, CONGRA requires only $O(n)$ bytes of working memory, whereas all other

optimization methods require order $O(n^2)$ bytes of working memory. During n successive iterations, uninterrupted by restarts or changes in the working set, the conjugate gradient algorithm computes a cycle of n conjugate search directions. In each iteration, a line search is performed along the search direction to find an approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation to obtain a step size α satisfying the Goldstein conditions. One of the Goldstein conditions can be violated if the feasible region defines an upper limit for the step size. Other line-search algorithms can be specified with the LINESEARCH= option.

Nelder-Mead Simplex Optimization (NMSIMP)

The Nelder-Mead simplex method does not use any derivatives and does not assume that the objective function has continuous derivatives. The objective function itself needs to be continuous. This technique is quite expensive in the number of function calls, and it might be unable to generate precise results for $n \gg 40$.

The original Nelder-Mead simplex algorithm is implemented and extended to boundary constraints. This algorithm does not compute the objective for infeasible points, but it changes the shape of the simplex adapting to the nonlinearities of the objective function, which contributes to an increased speed of convergence. It uses a special termination criterion.

Finite-Difference Approximations of Derivatives

The FD= and FDHESSIAN= options specify the use of finite-difference approximations of the derivatives. The FD= option specifies that all derivatives are approximated using function evaluations, and the FDHESSIAN= option specifies that second-order derivatives are approximated using gradient evaluations.

Computing derivatives by finite-difference approximations can be very time-consuming, especially for second-order derivatives based only on values of the objective function (FD= option). If analytical derivatives are difficult to obtain (for example, if a function is computed by an iterative process), you might consider one of the optimization techniques that use first-order derivatives only (QUANEW, DBLDOG, or CONGRA). In the expressions that follow, $\boldsymbol{\theta}$ denotes the parameter vector, h_i denotes the step size for the ith parameter, and $\mathbf{e}_i$ is a vector of zeros with a 1 in the ith position.

Forward-Difference Approximations

The forward-difference derivative approximations consume less computer time, but they are usually not as precise as approximations that use central-difference formulas.

- For first-order derivatives, n additional function calls are required:

$$g_i = \frac{\partial f}{\partial \theta_i} \approx \frac{f(\boldsymbol{\theta} + h_i \mathbf{e}_i) - f(\boldsymbol{\theta})}{h_i}$$

- For second-order derivatives based on function calls only (Dennis and Schnabel 1983, p. 80), $n + n^2/2$ additional function calls are required for dense Hessian:

$$\frac{\partial^2 f}{\partial \theta_i \partial \theta_j} \approx \frac{f(\boldsymbol{\theta} + h_i \mathbf{e}_i + h_j \mathbf{e}_j) - f(\boldsymbol{\theta} + h_i \mathbf{e}_i) - f(\boldsymbol{\theta} + h_j \mathbf{e}_j) + f(\boldsymbol{\theta})}{h_i h_j}$$

- For second-order derivatives based on gradient calls (Dennis and Schnabel 1983, p. 103), n additional gradient calls are required:

$$\frac{\partial^2 f}{\partial\theta_i\partial\theta_j} \approx \frac{g_i(\boldsymbol{\theta} + h_j\mathbf{e}_j) - g_i(\boldsymbol{\theta})}{2h_j} + \frac{g_j(\boldsymbol{\theta} + h_i\mathbf{e}_i) - g_j(\boldsymbol{\theta})}{2h_i}$$

Central-Difference Approximations

Central-difference approximations are usually more precise, but they consume more computer time than approximations that use forward-difference derivative formulas.

- For first-order derivatives, $2n$ additional function calls are required:

$$g_i = \frac{\partial f}{\partial\theta_i} \approx \frac{f(\boldsymbol{\theta} + h_i\mathbf{e}_i) - f(\boldsymbol{\theta} - h_i\mathbf{e}_i)}{2h_i}$$

- For second-order derivatives based on function calls only (Abramowitz and Stegun 1972, p. 884), $2n + 4n^2/2$ additional function calls are required.

$$\frac{\partial^2 f}{\partial\theta_i^2} \approx \frac{-f(\boldsymbol{\theta} + 2h_i\mathbf{e}_i) + 16f(\boldsymbol{\theta} + h_i\mathbf{e}_i) - 30f(\boldsymbol{\theta}) + 16f(\boldsymbol{\theta} - h_{\mathbf{i}}e_i) - f(\boldsymbol{\theta} - 2h_i\mathbf{e}_i)}{12h_i^2}$$

$$\frac{\partial^2 f}{\partial\theta_i\partial\theta_j} \approx \frac{f(\boldsymbol{\theta} + h_i\mathbf{e}_i + h_j\mathbf{e}_j) - f(\boldsymbol{\theta} + h_i\mathbf{e}_i - h_j\mathbf{e}_j) - f(\boldsymbol{\theta} - h_i\mathbf{e}_i + h_j\mathbf{e}_j) + f(\boldsymbol{\theta} - h_i\mathbf{e}_i - h_j\mathbf{e}_j)}{4h_ih_j}$$

- For second-order derivatives based on gradient calls, $2n$ additional gradient calls are required:

$$\frac{\partial^2 f}{\partial\theta_i\partial\theta_j} \approx \frac{g_i(\boldsymbol{\theta} + h_j\mathbf{e}_j) - g_i(\boldsymbol{\theta} - h_j\mathbf{e}_j)}{4h_j} + \frac{g_j(\boldsymbol{\theta} + h_i\mathbf{e}_i) - g_j(\boldsymbol{\theta} - h_i\mathbf{e}_i)}{4h_i}$$

You can use the FDIGITS= option to specify the number of accurate digits in the evaluation of the objective function. This specification is helpful in determining an appropriate interval size h to be used in the finite-difference formulas.

The step sizes h_j, $j = 1, \ldots, n$ are defined as follows:

- For the forward-difference approximation of first-order derivatives that use function calls and second-order derivatives that use gradient calls, $h_j = \sqrt[2]{\eta}(1 + |\theta_j|)$.
- For the forward-difference approximation of second-order derivatives that use only function calls and all central-difference formulas, $h_j = \sqrt[3]{\eta}(1 + |\theta_j|)$.

The value of η is defined by the FDIGITS= option:

- If you specify the number of accurate digits by using FDIGITS=r, η is set to 10^{-r}.
- If you do not specify the FDIGITS= option, η is set to the machine precision ϵ.

Hessian Scaling

The rows and columns of the Hessian matrix can be scaled when you use the trust region, Newton-Raphson, and double-dogleg optimization techniques. Each element $H_{i,j}, i, j = 1, \ldots, n$ is divided by the scaling factor $d_i d_j$, where the scaling vector $d = (d_1, \ldots, d_n)$ is iteratively updated in a way specified by the HESCAL=*i* option, as follows:

$i = 0:$ No scaling is done (equivalent to $d_i = 1$).

$i \neq 0:$ First iteration and each restart iteration sets:

$$d_i^{(0)} = \sqrt{\max(|H_{i,i}^{(0)}|, \epsilon)}$$

$i = 1:$ See Moré (1978):

$$d_i^{(k+1)} = \max\left[d_i^{(k)}, \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}\right]$$

$i = 2:$ See Dennis, Gay, and Welsch (1981):

$$d_i^{(k+1)} = \max\left[0.6 d_i^{(k)}, \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}\right]$$

$i = 3:$ d_i is reset in each iteration:

$$d_i^{(k+1)} = \sqrt{\max(|H_{i,i}^{(k)}|, \epsilon)}$$

In the preceding equations, ϵ is the relative machine precision or, equivalently, the largest double-precision value that, when added to 1, results in 1.

Active Set Methods

The parameter vector $\boldsymbol{\theta} \in \mathcal{R}^n$ can be subject to a set of m linear equality and inequality constraints:

$$\sum_{j=1}^{n} a_{ij}\theta_j = b_i \qquad i = 1, \ldots, m_e$$
$$\sum_{j=1}^{n} a_{ij}\theta_j \geq b_i \qquad i = m_e + 1, \ldots, m$$

The coefficients a_{ij} and right-hand sides b_i of the equality and inequality constraints are collected in the $m \times n$ matrix $\mathbf{A}$ and the m vector $\mathbf{b}$.

The m linear constraints define a feasible region $\mathcal{G}$ in $\mathcal{R}^n$ that must contain the point $\boldsymbol{\theta}_*$ that minimizes the problem. If the feasible region $\mathcal{G}$ is empty, no solution to the optimization problem exists.

In PROC NLMIXED, all optimization techniques use *active set methods*. The iteration starts with a feasible point $\boldsymbol{\theta}^{(0)}$, which you can provide or which can be computed by the Schittkowski and Stoer (1979) algorithm

implemented in PROC NLMIXED. The algorithm then moves from one feasible point $\boldsymbol{\theta}^{(k-1)}$ to a better feasible point $\boldsymbol{\theta}^{(k)}$ along a feasible search direction $\mathrm{s}^{(k)}$,

$$\boldsymbol{\theta}^{(k)} = \boldsymbol{\theta}^{(k-1)} + \alpha^{(k)}\mathrm{s}^{(k)}, \quad \alpha^{(k)} > 0$$

Theoretically, the path of points $\boldsymbol{\theta}^{(k)}$ never leaves the feasible region $\mathcal{G}$ of the optimization problem, but it can reach its boundaries. The active set $\mathcal{A}^{(k)}$ of point $\boldsymbol{\theta}^{(k)}$ is defined as the index set of all linear equality constraints and those inequality constraints that are satisfied at $\boldsymbol{\theta}^{(k)}$. If no constraint is active $\boldsymbol{\theta}^{(k)}$, the point is located in the interior of $\mathcal{G}$, and the active set $\mathcal{A}^{(k)} = \emptyset$ is empty. If the point $\boldsymbol{\theta}^{(k)}$ in iteration k hits the boundary of inequality constraint i, this constraint i becomes active and is added to $\mathcal{A}^{(k)}$. Each equality constraint and each active inequality constraint reduce the dimension (degrees of freedom) of the optimization problem.

In practice, the active constraints can be satisfied only with finite precision. The LCEPSILON=*r* option specifies the range for active and violated linear constraints. If the point $\boldsymbol{\theta}^{(k)}$ satisfies the condition

$$\left| \sum_{j=1}^{n} a_{ij}\theta_j^{(k)} - b_i \right| \leq t$$

where $t = r(|b_i| + 1)$, the constraint i is recognized as an active constraint. Otherwise, the constraint i is either an inactive inequality or a violated inequality or equality constraint. Due to rounding errors in computing the projected search direction, error can be accumulated so that an iterate $\boldsymbol{\theta}^{(k)}$ steps out of the feasible region.

In those cases, PROC NLMIXED might try to pull the iterate $\boldsymbol{\theta}^{(k)}$ back into the feasible region. However, in some cases the algorithm needs to increase the feasible region by increasing the LCEPSILON=*r* value. If this happens, a message is displayed in the log output.

If the algorithm cannot improve the value of the objective function by moving from an active constraint back into the interior of the feasible region, it makes this inequality constraint an equality constraint in the next iteration. This means that the active set $\mathcal{A}^{(k+1)}$ still contains the constraint i. Otherwise, it releases the active inequality constraint and increases the dimension of the optimization problem in the next iteration.

A serious numerical problem can arise when some of the active constraints become (nearly) linearly dependent. PROC NLMIXED removes linearly dependent equality constraints before starting optimization. You can use the LCSINGULAR= option to specify a criterion r used in the update of the QR decomposition that determines whether an active constraint is linearly dependent relative to a set of other active constraints.

If the solution $\boldsymbol{\theta}^*$ is subjected to n_{act} linear equality or active inequality constraints, the QR decomposition of the $n \times n_{act}$ matrix $\hat{\mathbf{A}}'$ of the linear constraints is computed by $\hat{\mathbf{A}}' = \mathbf{QR}$, where $\mathbf{Q}$ is an $n \times n$ orthogonal matrix and $\mathbf{R}$ is an $n \times n_{act}$ upper triangular matrix. The n columns of matrix $\mathbf{Q}$ can be separated into two matrices, $\mathbf{Q} = [\mathbf{Y}, \mathbf{Z}]$, where $\mathbf{Y}$ contains the first n_{act} orthogonal columns of $\mathbf{Q}$ and $\mathbf{Z}$ contains the last $n - n_{act}$ orthogonal columns of $\mathbf{Q}$. The $n \times (n - n_{act})$ column-orthogonal matrix $\mathbf{Z}$ is also called the *null-space matrix* of the active linear constraints $\hat{\mathbf{A}}'$. The $n - n_{act}$ columns of the $n \times (n - n_{act})$ matrix $\mathbf{Z}$ form a basis orthogonal to the rows of the $n_{act} \times n$ matrix $\hat{\mathbf{A}}$.

At the end of the iterating, PROC NLMIXED computes the *projected gradient* g_Z,

$$\mathrm{g}_Z = \mathbf{Z}'\mathrm{g}$$

In the case of boundary-constrained optimization, the elements of the projected gradient correspond to the gradient elements of the free parameters. A necessary condition for $\boldsymbol{\theta}^*$ to be a local minimum of the

optimization problem is

$$\mathbf{g}_Z(\boldsymbol{\theta}^*) = \mathbf{Z}'\mathbf{g}(\boldsymbol{\theta}^*) = \mathbf{0}$$

The symmetric $n_{act} \times n_{act}$ matrix $\mathbf{G}_Z$,

$$\mathbf{G}_Z = \mathbf{Z}'\mathbf{G}\mathbf{Z}$$

is called a *projected Hessian matrix*. A second-order necessary condition for $\boldsymbol{\theta}^*$ to be a local minimizer requires that the projected Hessian matrix is positive semidefinite.

Those elements of the n_{act} vector of first-order estimates of *Lagrange multipliers*,

$$\lambda = (\hat{\mathbf{A}}\hat{\mathbf{A}}')^{-1}\hat{\mathbf{A}}\mathbf{Z}\mathbf{Z}'\mathbf{g}$$

that correspond to active inequality constraints indicate whether an improvement of the objective function can be obtained by releasing this active constraint. For minimization, a significant negative Lagrange multiplier indicates that a possible reduction of the objective function can be achieved by releasing this active linear constraint. The LCDEACT=*r* option specifies a threshold *r* for the Lagrange multiplier that determines whether an active inequality constraint remains active or can be deactivated. (In the case of boundary-constrained optimization, the Lagrange multipliers for active lower (upper) constraints are the negative (positive) gradient elements corresponding to the active parameters.)

Line-Search Methods

In each iteration k, the (dual) quasi-Newton, conjugate gradient, and Newton-Raphson minimization techniques use iterative line-search algorithms that try to optimize a linear, quadratic, or cubic approximation of f along a feasible descent search direction $\mathbf{s}^{(k)}$,

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \alpha^{(k)}\mathbf{s}^{(k)}, \quad \alpha^{(k)} > 0$$

by computing an approximately optimal scalar $\alpha^{(k)}$.

Therefore, a line-search algorithm is an iterative process that optimizes a nonlinear function $f(\alpha)$ of one parameter (α) within each iteration k of the optimization technique. Since the outside iteration process is based only on the approximation of the objective function, the inside iteration of the line-search algorithm does not have to be perfect. Usually, it is satisfactory that the choice of α significantly reduces (in a minimization) the objective function. Criteria often used for termination of line-search algorithms are the Goldstein conditions (see Fletcher 1987).

You can select various line-search algorithms by specifying the LINESEARCH= option. The line-search method LINESEARCH=2 seems to be superior when function evaluation consumes significantly less computation time than gradient evaluation. Therefore, LINESEARCH=2 is the default method for Newton-Raphson, (dual) quasi-Newton, and conjugate gradient optimizations.

You can modify the line-search methods LINESEARCH=2 and LINESEARCH=3 to be exact line searches by using the LSPRECISION= option and specifying the σ parameter described in Fletcher (1987). The line-search methods LINESEARCH=1, LINESEARCH=2, and LINESEARCH=3 satisfy the left-side and right-side Goldstein conditions (see Fletcher 1987). When derivatives are available, the line-search methods LINESEARCH=6, LINESEARCH=7, and LINESEARCH=8 try to satisfy the right-side Goldstein condition; if derivatives are not available, these line-search algorithms use only function calls.

Restricting the Step Length

Almost all line-search algorithms use iterative extrapolation techniques that can easily lead them to (feasible) points where the objective function f is no longer defined or is difficult to compute. Therefore, PROC NLMIXED provides options restricting the step length α or trust region radius Δ, especially during the first main iterations.

The inner product $\mathbf{g}'\mathbf{s}$ of the gradient $\mathbf{g}$ and the search direction $\mathbf{s}$ is the slope of $f(\alpha) = f(\boldsymbol{\theta} + \alpha\mathbf{s})$ along the search direction $\mathbf{s}$. The default starting value $\alpha^{(0)} = \alpha^{(k,0)}$ in each line-search algorithm $(\min_{\alpha>0} f(\boldsymbol{\theta} + \alpha\mathbf{s}))$ during the main iteration k is computed in three steps:

1. The first step uses either the difference $|\Delta f| = |f^{(k)} - f^{(k-1)}|$ of the function values during the last two consecutive iterations or the final step-size value α^- of the last iteration $k-1$ to compute a first value of $\alpha_1^{(0)}$.

   - If the DAMPSTEP option is not used,

     $$\alpha_1^{(0)} = \begin{cases} step & \text{if } 0.1 \le step \le 10 \\ 10 & \text{if } step > 10 \\ 0.1 & \text{if } step < 0.1 \end{cases}$$

     with

     $$step = \begin{cases} |\Delta f|/|\mathbf{g}'\mathbf{s}| & \text{if } |\mathbf{g}'\mathbf{s}| \ge \epsilon \max(100 \times |\Delta f|, 1) \\ 1 & \text{otherwise} \end{cases}$$

     This value of $\alpha_1^{(0)}$ can be too large and can lead to a difficult or impossible function evaluation, especially for highly nonlinear functions such as the EXP function.

   - If the DAMPSTEP=*r* option is used,

     $$\alpha_1^{(0)} = \min(1, r\alpha^-)$$

     The initial value for the new step length can be no larger than r times the final step length α^- of the former iteration. The default value is $r = 2$.

2. During the first five iterations, the second step enables you to reduce $\alpha_1^{(0)}$ to a smaller starting value $\alpha_2^{(0)}$ by using the INSTEP=*r* option:

   $$\alpha_2^{(0)} = \min(\alpha_1^{(0)}, r)$$

   After more than five iterations, $\alpha_2^{(0)}$ is set to $\alpha_1^{(0)}$.

3. The third step can further reduce the step length by

   $$\alpha_3^{(0)} = \min(\alpha_2^{(0)}, \min(10, u))$$

   where u is the maximum length of a step inside the feasible region.

The INSTEP=r option enables you to specify a smaller or larger radius Δ of the trust region used in the first iteration of the trust region and double-dogleg algorithms. The default initial trust region radius $\Delta^{(0)}$ is the length of the scaled gradient (Moré 1978). This step corresponds to the default radius factor of $r = 1$. In most practical applications of the TRUREG and DBLDOG algorithms, this choice is successful. However, for bad initial values and highly nonlinear objective functions (such as the EXP function), the default start radius can result in arithmetic overflows. If this happens, you can try decreasing values of INSTEP=r, $0 < r < 1$, until the iteration starts successfully. A small factor r also affects the trust region radius $\Delta^{(k+1)}$ of the next steps because the radius is changed in each iteration by a factor $0 < c \leq 4$, depending on the ratio ρ expressing the goodness of quadratic function approximation. Reducing the radius Δ corresponds to increasing the ridge parameter λ, producing smaller steps aimed more closely toward the (negative) gradient direction.

Computational Problems

Floating-Point Errors and Overflows

Numerical optimization of a numerically integrated function is a difficult task, and the computation of the objective function and its derivatives can lead to arithmetic exceptions and overflows. A typical cause of these problems is parameters with widely varying scales. If the scaling of your parameters varies by more than a few orders of magnitude, the numerical stability of the optimization problem can be seriously reduced and can result in computational difficulties. A simple remedy is to rescale each parameter so that its final estimated value has a magnitude near 1.

If parameter rescaling does not help, consider the following actions:

- Specify the ITDETAILS option in the PROC NLMIXED statement to obtain more detailed information about when and where the problem is occurring.
- Provide different initial values or try a grid search of values.
- Use boundary constraints to avoid the region where overflows can happen.
- Delete outlying observations or subjects from the input data, if this is reasonable.
- Change the algorithm (specified in programming statements) that computes the objective function.

The line-search algorithms that work with cubic extrapolation are especially sensitive to arithmetic overflows. If an overflow occurs during a line search, you can use the INSTEP= option to reduce the length of the first trial step during the first five iterations, or you can use the DAMPSTEP or MAXSTEP option to restrict the step length of the initial α in subsequent iterations. If an arithmetic overflow occurs in the first iteration of the trust region or double-dogleg algorithm, you can use the INSTEP= option to reduce the default trust region radius of the first iteration. You can also change the optimization technique or the line-search method.

Long Run Times

PROC NLMIXED can take a long time to run for problems involving complex models, many parameters, or large input data sets. Although the optimization techniques used by PROC NLMIXED are some of the best ones available, they are not guaranteed to converge quickly for all problems. Ill-posed or misspecified models can cause the algorithms to use more extensive calculations designed to achieve convergence, and this can result in longer run times. So first make sure that your model is specified correctly, that your parameters are scaled to be of the same order of magnitude, and that your data reasonably match the model you are contemplating.

If you are using the default adaptive Gaussian quadrature algorithm and no iteration history is printing at all, then PROC NLMIXED might be bogged down trying to determine the number of quadrature points at the first set of starting values. Specifying the QPOINTS= option will bypass this stage and proceed directly to iterations; however, be aware that the likelihood approximation might not be accurate if there are too few quadrature points.

PROC NLMIXED can also have difficulty determining the number of quadrature points if the initial starting values are far from the optimum values. To obtain more accurate starting values for the model parameters, one easy method is to fit a model with no RANDOM statement. You can then use these estimates as starting values, although you will still need to specify values for the random-effects distribution. For normal-normal models, another strategy is to use METHOD=FIRO. If you can obtain estimates by using this approximate method, then they can be used as starting values for more accurate likelihood approximations.

If you are running PROC NLMIXED multiple times, you will probably want to include a statement like the following in your program:

```
ods output ParameterEstimates=pe;
```

This statement creates a SAS data set named PE upon completion of the run. In your next invocation of PROC NLMIXED, you can then specify

```
parms / data=pe;
```

to read in the previous estimates as starting values.

To speed general computations, you should double-check your programming statements to minimize the number of floating-point operations. Using auxiliary variables and factoring amenable expressions can be useful changes in this regard.

Problems Evaluating Code for Objective Function

The starting point $\boldsymbol{\theta}^{(0)}$ must be a point for which the programming statements can be evaluated. However, during optimization, the optimizer might iterate to a point $\boldsymbol{\theta}^{(k)}$ where the objective function or its derivatives cannot be evaluated. In some cases, the specification of boundary for parameters can avoid such situations. In many other cases, you can indicate that the point $\boldsymbol{\theta}^{(0)}$ is a bad point simply by returning an extremely large value for the objective function. In these cases, the optimization algorithm reduces the step length and stays closer to the point that has been evaluated successfully in the former iteration.

No Convergence

There are a number of things to try if the optimizer fails to converge.

- Change the initial values by using a grid search specification to obtain a set of good feasible starting values.
- Change or modify the update technique or the line-search algorithm.

  This method applies only to TECH=QUANEW and TECH=CONGRA. For example, if you use the default update formula and the default line-search algorithm, you can do the following:

  - change the update formula with the UPDATE= option
  - change the line-search algorithm with the LINESEARCH= option
  - specify a more precise line search with the LSPRECISION= option, if you use LINESEARCH=2 or LINESEARCH=3
- Change the optimization technique.

  For example, if you use the default option, TECH=QUANEW, you can try one of the second-derivative methods if your problem is small or the conjugate gradient method if it is large.
- Adjust finite-difference derivatives.

  The forward-difference derivatives specified with the FD= or FDHESSIAN= option might not be precise enough to satisfy strong gradient termination criteria. You might need to specify the more expensive central-difference formulas. The finite-difference intervals might be too small or too big, and the finite-difference derivatives might be erroneous.
- Double-check the data entry and program specification.

Convergence to Stationary Point

The gradient at a stationary point is the null vector, which always leads to a zero search direction. This point satisfies the first-order termination criterion. Search directions that are based on the gradient are zero, so the algorithm terminates. There are two ways to avoid this situation:

- Use the PARMS statement to specify a grid of feasible initial points.
- Use the OPTCHECK=*r* option to avoid terminating at the stationary point.

The signs of the eigenvalues of the (reduced) Hessian matrix contain the following information regarding a stationary point:

- If all of the eigenvalues are positive, the Hessian matrix is positive definite, and the point is a minimum point.
- If some of the eigenvalues are positive and all remaining eigenvalues are zero, the Hessian matrix is positive semidefinite, and the point is a minimum or saddle point.
- If all of the eigenvalues are negative, the Hessian matrix is negative definite, and the point is a maximum point.

- If some of the eigenvalues are negative and all remaining eigenvalues are zero, the Hessian matrix is negative semidefinite, and the point is a maximum or saddle point.
- If all of the eigenvalues are zero, the point can be a minimum, maximum, or saddle point.

Precision of Solution

In some applications, PROC NLMIXED can result in parameter values that are not precise enough. Usually, this means that the procedure terminated at a point too far from the optimal point. The termination criteria define the size of the termination region around the optimal point. Any point inside this region can be accepted for terminating the optimization process. The default values of the termination criteria are set to satisfy a reasonable compromise between the computational effort (computer time) and the precision of the computed estimates for the most common applications. However, there are a number of circumstances in which the default values of the termination criteria specify a region that is either too large or too small.

If the termination region is too large, then it can contain points with low precision. In such cases, you should determine which termination criterion stopped the optimization process. In many applications, you can obtain a solution with higher precision simply by using the old parameter estimates as starting values in a subsequent run in which you specify a smaller value for the termination criterion that was satisfied at the former run.

If the termination region is too small, the optimization process might take longer to find a point inside such a region, or it might not even find such a point due to rounding errors in function values and derivatives. This can easily happen in applications in which finite-difference approximations of derivatives are used and the GCONV and ABSGCONV termination criteria are too small to respect rounding errors in the gradient values.

Covariance Matrix

The estimated covariance matrix of the parameter estimates is computed as the inverse Hessian matrix, and for unconstrained problems it should be positive definite. If the final parameter estimates are subjected to $n_{act} > 0$ active linear inequality constraints, the formulas of the covariance matrices are modified similar to Gallant (1987) and Cramer (1986, p. 38) and additionally generalized for applications with singular matrices.

There are several steps available that enable you to tune the rank calculations of the covariance matrix.

1. You can use the ASINGULAR=, MSINGULAR=, and VSINGULAR= options to set three singularity criteria for the inversion of the Hessian matrix **H**. The singularity criterion used for the inversion is

   $$|d_{j,j}| \leq \max(\text{ASING}, \text{VSING} * |H_{j,j}|, \text{MSING} * \max(|H_{1,1}|, \ldots, |H_{n,n}|))$$

   where $d_{j,j}$ is the diagonal pivot of the matrix **H**, and ASING, VSING, and MSING are the specified values of the ASINGULAR=, VSINGULAR=, and MSINGULAR= options, respectively. The default values are as follows:

   - ASING: the square root of the smallest positive double-precision value
   - MSING: 1E–12 if you do not specify the SINGHESS= option and $\max(10\epsilon, 1\text{E}{-4}\times\text{SINGHESS})$ otherwise, where ϵ is the machine precision
   - VSING: 1E–8 if you do not specify the SINGHESS= option and the value of SINGHESS otherwise

Note that, in many cases, a normalized matrix $\mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-1}$ is decomposed, and the singularity criteria are modified correspondingly.

2. If the matrix $\mathbf{H}$ is found to be singular in the first step, a generalized inverse is computed. Depending on the G4= option, either a generalized inverse satisfying all four Moore-Penrose conditions is computed (a g_4-inverse) or a generalized inverse satisfying only two Moore-Penrose conditions is computed (a g_2-inverse, Pringle and Rayner 1971). If the number of parameters n of the application is less than or equal to G4=i, a g_4-inverse is computed; otherwise, only a g_2-inverse is computed. The g_4-inverse is computed by the (computationally very expensive but numerically stable) eigenvalue decomposition, and the g_2-inverse is computed by Gauss transformation. The g_4-inverse is computed using the eigenvalue decomposition $\mathbf{A} = \mathbf{Z}\mathbf{\Lambda}\mathbf{Z}'$, where $\mathbf{Z}$ is the orthogonal matrix of eigenvectors and $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues, $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$. The g_4-inverse of $\mathbf{H}$ is set to

$$\mathbf{A}^- = \mathbf{Z}\mathbf{\Lambda}^-\mathbf{Z}'$$

where the diagonal matrix $\mathbf{\Lambda}^- = \mathrm{diag}(\lambda_1^-, \ldots, \lambda_n^-)$ is defined using the COVSING= option:

$$\lambda_i^- = \begin{cases} 1/\lambda_i & \text{if } |\lambda_i| > \mathrm{COVSING} \\ 0 & \text{if } |\lambda_i| \le \mathrm{COVSING} \end{cases}$$

If you do not specify the COVSING= option, the nr smallest eigenvalues are set to zero, where nr is the number of rank deficiencies found in the first step.

For optimization techniques that do not use second-order derivatives, the covariance matrix is computed using finite-difference approximations of the derivatives.

Prediction

The nonlinear mixed model is a useful tool for statistical prediction. Assuming a prediction is to be made regarding the ith subject, suppose that $f(\boldsymbol{\theta}, \mathbf{u}_i)$ is a differentiable function predicting some quantity of interest. Recall that $\boldsymbol{\theta}$ denotes the vector of unknown parameters and $\mathbf{u}_i$ denotes the vector of random effects for the ith subject. A natural point prediction is $f(\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{u}}_i)$, where $\widehat{\boldsymbol{\theta}}$ is the maximum likelihood estimate of $\boldsymbol{\theta}$ and $\widehat{\mathbf{u}}_i$ is the empirical Bayes estimate of $\mathbf{u}_i$ described previously in the section "Integral Approximations" on page 7279.

An approximate prediction variance matrix for $(\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{u}}_i)$ is

$$\mathbf{P} = \begin{bmatrix} \widehat{\mathbf{H}}^{-1} & \widehat{\mathbf{H}}^{-1}\left(\frac{\partial \widehat{\mathbf{u}}_i}{\partial \boldsymbol{\theta}}\right)' \\ \left(\frac{\partial \widehat{\mathbf{u}}_i}{\partial \boldsymbol{\theta}}\right)\widehat{\mathbf{H}}^{-1} & \widehat{\mathbf{\Gamma}}^{-1} + \left(\frac{\partial \widehat{\mathbf{u}}_i}{\partial \boldsymbol{\theta}}\right)\widehat{\mathbf{H}}^{-1}\left(\frac{\partial \widehat{\mathbf{u}}_i}{\partial \boldsymbol{\theta}}\right)' \end{bmatrix}$$

where $\widehat{\mathbf{H}}$ is the approximate Hessian matrix from the optimization for $\widehat{\boldsymbol{\theta}}$, $\widehat{\mathbf{\Gamma}}$ is the approximate Hessian matrix from the optimization for $\widehat{\mathbf{u}}_i$, and $(\partial\widehat{\mathbf{u}}_i/\partial\boldsymbol{\theta})$ is the derivative of $\widehat{\mathbf{u}}_i$ with respect to $\boldsymbol{\theta}$, evaluated at $(\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{u}}_i)$. The approximate variance matrix for $\widehat{\boldsymbol{\theta}}$ is the standard one discussed in the previous section, and that for $\widehat{\mathbf{u}}_i$ is an approximation to the conditional mean squared error of prediction described by Booth and Hobert (1998).

The prediction variance for a general scalar function $f(\boldsymbol{\theta}, \mathbf{u}_i)$ is defined as the expected squared difference $E[f(\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{u}}_i) - f(\boldsymbol{\theta}, \mathbf{u}_i)]^2$. PROC NLMIXED computes an approximation to it as follows. The derivative of $f(\boldsymbol{\theta}, \mathbf{u}_i)$ is computed with respect to each element of $(\boldsymbol{\theta}, \mathbf{u}_i)$ and evaluated at $(\widehat{\boldsymbol{\theta}}, \widehat{\mathbf{u}}_i)$. If $\mathbf{a}_i$ is the resulting vector, then the approximate prediction variance is $\mathbf{a}_i'\mathbf{P}\mathbf{a}_i$. This approximation is known as the delta method (Billingsley 1986; Cox 1998).

Computational Resources

Since nonlinear optimization is an iterative process that depends on many factors, it is difficult to estimate how much computer time is necessary to find an optimal solution satisfying one of the termination criteria. You can use the MAXTIME=, MAXITER=, and MAXFUNC= options to restrict the amount of CPU time, the number of iterations, and the number of function calls in a single run of PROC NLMIXED.

In each iteration k, the NRRIDG technique uses a symmetric Householder transformation to decompose the $n \times n$ Hessian matrix $\mathbf{H}$,

$$\mathbf{H} = \mathbf{V}'\mathbf{T}\mathbf{V}, \qquad \mathbf{V}\text{: orthogonal}, \quad \mathbf{T}\text{: tridiagonal}$$

to compute the (Newton) search direction $\mathbf{s}$,

$$\mathbf{s}^{(k)} = -[\mathbf{H}^{(k)}]^{-1}\mathbf{g}^{(k)} \qquad k = 1, 2, 3, \ldots$$

The TRUREG and NEWRAP techniques use the Cholesky decomposition to solve the same linear system while computing the search direction. The QUANEW, DBLDOG, CONGRA, and NMSIMP techniques do not need to invert or decompose a Hessian matrix; thus, they require less computational resources than the other techniques.

The larger the problem, the more time is needed to compute function values and derivatives. Therefore, you might want to compare optimization techniques by counting and comparing the respective numbers of function, gradient, and Hessian evaluations.

Finite-difference approximations of the derivatives are expensive because they require additional function or gradient calls:

- forward-difference formulas
  - For first-order derivatives, n additional function calls are required.
  - For second-order derivatives based on function calls only, for a dense Hessian, $n+n^2/2$ additional function calls are required.
  - For second-order derivatives based on gradient calls, n additional gradient calls are required.
- central-difference formulas
  - For first-order derivatives, $2n$ additional function calls are required.
  - For second-order derivatives based on function calls only, for a dense Hessian, $2n+2n^2$ additional function calls are required.
  - For second-order derivatives based on gradient calls, $2n$ additional gradient calls are required.

Many applications need considerably more time for computing second-order derivatives (Hessian matrix) than for computing first-order derivatives (gradient). In such cases, a dual quasi-Newton technique is recommended, which does not require second-order derivatives.

Displayed Output

This section describes the displayed output from PROC NLMIXED. See the section "ODS Table Names" on page 7314 for details about how this output interfaces with the Output Delivery System.

Specifications

The NLMIXED procedure first displays the "Specifications" table, listing basic information about the nonlinear mixed model that you have specified. It includes the principal variables and estimation methods.

Dimensions

The "Dimensions" table lists counts of important quantities in your nonlinear mixed model, including the number of observations, subjects, parameters, and quadrature points.

Parameters

The "Parameters" table displays the information you provided with the PARMS statement and the value of the negative log-likelihood function evaluated at the starting values.

Starting Gradient and Hessian

The START option in the PROC NLMIXED statement displays the gradient of the negative log-likelihood function at the starting values of the parameters. If you also specify the HESS option, then the starting Hessian is displayed as well.

Iterations

The iteration history consists of one line of output for each iteration in the optimization process. The iteration history is displayed by default because it is important that you check for possible convergence problems. The default iteration history includes the following variables:

- Iter, the iteration number
- Calls, the number of function calls
- NegLogLike, the value of the objective function
- Diff, the difference between adjacent function values
- MaxGrad, the maximum of the absolute (projected) gradient components (except NMSIMP)
- Slope, the slope $\mathbf{g}'\mathbf{s}$ of the search direction $\mathbf{s}$ at the current parameter iterate $\boldsymbol{\theta}^{(k)}$ (QUANEW only)
- Rho, the ratio between the achieved and predicted values of Diff (NRRIDG only)
- Radius, the radius of the trust region (TRUREG only)
- StdDev, the standard deviation of the simplex values (NMSIMP only)
- Delta, the vertex length of the simplex (NMSIMP only)
- Size, the size of the simplex (NMSIMP only)

For the QUANEW method, the value of Slope should be significantly negative. Otherwise, the line-search algorithm has difficulty reducing the function value sufficiently. If this difficulty is encountered, an asterisk (*) appears after the iteration number. If there is a tilde ∼ after the iteration number, the BFGS update is skipped, and very high values of the Lagrange function are produced. A backslash (\ ) after the iteration number indicates that Powell's correction for the BFGS update is used.

For methods using second derivatives, an asterisk (*) after the iteration number means that the computed Hessian approximation was singular and had to be ridged with a positive value.

For the NMSIMP method, only one line is displayed for several internal iterations. This technique skips the output for some iterations because some of the termination tests (StdDev and Size) are rather time-consuming compared to the simplex operations, and they are performed only every five simplex operations.

The ITDETAILS option in the PROC NLMIXED statement provides a more detailed iteration history. Besides listing the current values of the parameters and their gradients, the ITDETAILS option provides the following values in addition to the default output:

- Restart, the number of iteration restarts
- Active, the number of active constraints
- Lambda, the value of the Lagrange multiplier (TRUREG and DBLDOG only)
- Ridge, the ridge value (NRRIDG only)
- Alpha, the line-search step size (QUANEW only)

An apostrophe (') trailing the number of active constraints indicates that at least one of the active constraints was released from the active set due to a significant Lagrange multiplier.

Convergence Status

The "Convergence Status" table contains a status message describing the reason for termination of the optimization. The ODS name of this table is ConvergenceStatus, and you can query the nonprinting numeric variable Status to check for a successful optimization. This is useful in batch processing, or when processing BY groups, for example, in simulations. Successful convergence is indicated by Status=0.

Fitting Information

The "Fitting Information" table lists the final minimized value of –2 times the log likelihood as well as the information criteria of Akaike (AIC) and Schwarz (BIC), as well as a finite-sample corrected version of AIC (AICC). The criteria are computed as follows:

$$\begin{aligned} \text{AIC} &= 2f(\widehat{\boldsymbol{\theta}}) + 2p \\ \text{AICC} &= 2f(\widehat{\boldsymbol{\theta}}) + 2pn/(n-p-1) \\ \text{BIC} &= 2f(\widehat{\boldsymbol{\theta}}) + p\log(s) \end{aligned}$$

where $f()$ is the negative of the marginal log-likelihood function, $\widehat{\boldsymbol{\theta}}$ is the vector of parameter estimates, p is the number of parameters, n is the number of observations, and s is the number of subjects. See Hurvich and Tsai (1989) and Burnham and Anderson (1998) for additional details.

Parameter Estimates

The "Parameter Estimates" table lists the estimates of the parameter values after successful convergence of the optimization problem or the final values of the parameters under nonconvergence. If the problem did converge, standard errors are computed from the final Hessian matrix. The ratio of the estimate with its standard error produces a t value, with approximate degrees of freedom computed as the number of subjects minus the number of random effects. A p-value and confidence limits based on this t distribution are also provided. Finally, the gradient of the negative log-likelihood function is displayed for each parameter, and you should verify that they each are sufficiently small for nonconstrained parameters.

Covariance and Correlation Matrices

Following standard maximum likelihood theory (for example, Serfling 1980), the asymptotic variance-covariance matrix of the parameter estimates equals the inverse of the Hessian matrix. You can display this matrix with the COV option in the PROC NLMIXED statement. The corresponding correlation form is available with the CORR option.

Additional Estimates

The "Additional Estimates" table displays the results of all ESTIMATE statements that you specify, with the same columns as the "Parameter Estimates" table. The ECOV and ECORR options in the PROC NLMIXED statement produce tables displaying the approximate covariance and correlation matrices of the additional estimates. They are computed using the delta method (Billingsley 1986; Cox 1998). The EDER option in the PROC NLMIXED statement produces a table that displays the derivatives of the additional estimates with respect to the model parameters evaluated at their final estimated values.

ODS Table Names

PROC NLMIXED assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 89.5. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 89.5 ODS Tables Produced by PROC NLMIXED

ODS Table Name	Description	Statement or Option
AdditionalEstimates	Results from ESTIMATE statements	ESTIMATE
Contrasts	Results from CONTRAST statements	CONTRAST
ConvergenceStatus	Convergence status	default
CorrMatAddEst	Correlation matrix of additional estimates	ECORR
CorrMatParmEst	Correlation matrix of parameter estimates	CORR
CovMatAddEst	Covariance matrix of additional estimates	ECOV
CovMatParmEst	Covariance matrix of parameter estimates	COV
DerAddEst	Derivatives of additional estimates	EDER
Dimensions	Dimensions of the problem	default
FitStatistics	Fit statistics	default
Hessian	Second derivative matrix	HESS
IterHistory	Iteration history	default
Parameters	Initial parameters	default
ParameterEstimates	Parameter estimates	default

Table 89.5 *continued*

ODS Table Name	Description	Statement or Option
Specifications	Model specifications	default
StartingHessian	Starting Hessian matrix	START HESS
StartingValues	Starting values and gradient	START

Examples: NLMIXED Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 89.1: One-Compartment Model with Pharmacokinetic Data

(View the complete code for this example (nlmex1.sas) in the example repository.)

A popular application of nonlinear mixed models is in the field of pharmacokinetics, which studies how a drug disperses through a living individual. This example considers the theophylline data from Pinheiro and Bates (1995). Serum concentrations of the drug theophylline are measured in 12 subjects over a 25-hour period after oral administration. The data are as follows.

```
data theoph;
   input subject time conc dose wt;
   datalines;
 1  0.00  0.74 4.02 79.6
 1  0.25  2.84 4.02 79.6
 1  0.57  6.57 4.02 79.6
 1  1.12 10.50 4.02 79.6
 1  2.02  9.66 4.02 79.6
 1  3.82  8.58 4.02 79.6
 1  5.10  8.36 4.02 79.6
 1  7.03  7.47 4.02 79.6

   ... more lines ...

12 24.15  1.17 5.30 60.5
;
```

Pinheiro and Bates (1995) consider the following first-order compartment model for these data:

$$C_{it} = \frac{Dk_{e_i}k_{a_i}}{Cl_i(k_{a_i} - k_{e_i})}[\exp(-k_{e_i}t) - \exp(-k_{a_i}t)] + e_{it}$$

where C_{it} is the observed concentration of the ith subject at time t, D is the dose of theophylline, k_{e_i} is the elimination rate constant for subject i, k_{a_i} is the absorption rate constant for subject i, Cl_i is the clearance

for subject i, and e_{it} are normal errors. To allow for random variability between subjects, they assume

$$
\begin{aligned}
Cl_i &= \exp(\beta_1 + b_{i1}) \\
k_{a_i} &= \exp(\beta_2 + b_{i2}) \\
k_{e_i} &= \exp(\beta_3)
\end{aligned}
$$

where the βs denote fixed-effects parameters and the b_is denote random-effects parameters with an unknown covariance matrix.

The PROC NLMIXED statements to fit this model are as follows:

```
proc nlmixed data=theoph;
   parms beta1=-3.22 beta2=0.47 beta3=-2.45
         s2b1 =0.03  cb12 =0    s2b2 =0.4 s2=0.5;
   cl   = exp(beta1 + b1);
   ka   = exp(beta2 + b2);
   ke   = exp(beta3);
   pred = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
   model conc ~ normal(pred,s2);
   random b1 b2 ~ normal([0,0],[s2b1,cb12,s2b2]) subject=subject;
run;
```

The PARMS statement specifies starting values for the three βs and four variance-covariance parameters. The clearance and rate constants are defined using SAS programming statements, and the conditional model for the data is defined to be normal with mean pred and variance s2. The two random effects are b1 and b2, and their joint distribution is defined in the RANDOM statement. Brackets are used in defining their mean vector (two zeros) and the lower triangle of their variance-covariance matrix (a general 2×2 matrix). The SUBJECT= variable is subject.

The results from this analysis are as follows.

Output 89.1.1 Model Specification for One-Compartment Model

The NLMIXED Procedure

Specifications	
Data Set	WORK.THEOPH
Dependent Variable	conc
Distribution for Dependent Variable	Normal
Random Effects	b1 b2
Distribution for Random Effects	Normal
Subject Variable	subject
Optimization Technique	Dual Quasi-Newton
Integration Method	Adaptive Gaussian Quadrature

The "Specifications" table lists the setup of the model (Output 89.1.1). The "Dimensions" table indicates that there are 132 observations, 12 subjects, and 7 parameters. PROC NLMIXED selects 5 quadrature points for each random effect, producing a total grid of 25 points over which quadrature is performed (Output 89.1.2).

Output 89.1.2 Dimensions Table for One-Compartment Model

Dimensions	
Observations Used	132
Observations Not Used	0
Total Observations	132
Subjects	12
Max Obs per Subject	11
Parameters	7
Quadrature Points	5

The "Parameters" table lists the 7 parameters, their starting values, and the initial evaluation of the negative log likelihood using adaptive Gaussian quadrature (Output 89.1.3). The "Iteration History" table indicates that 10 steps are required for the dual quasi-Newton algorithm to achieve convergence.

Output 89.1.3 Starting Values and Iteration History

Initial Parameters							
beta1	**beta2**	**beta3**	**s2b1**	**cb12**	**s2b2**	**s2**	**Negative Log Likelihood**
-3.22	0.47	-2.45	0.03	0	0.4	0.5	177.789945

Iteration History					
Iteration	**Calls**	**Negative Log Likelihood**	**Difference**	**Maximum Gradient**	**Slope**
1	7	177.7762	0.013697	2.87337	-63.0744
2	11	177.7643	0.011948	1.69814	-4.75239
3	14	177.7573	0.007036	1.29744	-1.97311
4	17	177.7557	0.001576	1.44141	-0.49772
5	20	177.7467	0.008988	1.13228	-0.82230
6	24	177.7464	0.000299	0.83129	-0.00244
7	27	177.7463	0.000083	0.72420	-0.00789
8	31	177.7457	0.000578	0.18002	-0.00583
9	34	177.7457	3.88E-6	0.017958	-8.25E-6
10	37	177.7457	3.222E-8	0.000143	-6.51E-8

NOTE: GCONV convergence criterion satisfied.

Output 89.1.4 Fit Statistics for One-Compartment Model

Fit Statistics	
-2 Log Likelihood	355.5
AIC (smaller is better)	369.5
AICC (smaller is better)	370.4
BIC (smaller is better)	372.9

The "Fit Statistics" table lists the final optimized values of the log-likelihood function and information criteria in the "smaller is better" form (Output 89.1.4).

The "Parameter Estimates" table contains the maximum likelihood estimates of the parameters (Output 89.1.5). Both s2b1 and s2b2 are marginally significant, indicating between-subject variability in the clearances and absorption rate constants, respectively. There does not appear to be a significant covariance between them, as seen by the estimate of cb12.

Output 89.1.5 Parameter Estimates for One-Compartment Model

Parameter Estimates

Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
beta1	-3.2268	0.05950	10	-54.23	<.0001	-3.3594	-3.0942	-0.00009
beta2	0.4806	0.1989	10	2.42	0.0363	0.03745	0.9238	3.645E-7
beta3	-2.4592	0.05126	10	-47.97	<.0001	-2.5734	-2.3449	0.000039
s2b1	0.02803	0.01221	10	2.30	0.0446	0.000828	0.05524	-0.00014
cb12	-0.00127	0.03404	10	-0.04	0.9710	-0.07712	0.07458	-0.00007
s2b2	0.4331	0.2005	10	2.16	0.0561	-0.01354	0.8798	-6.98E-6
s2	0.5016	0.06837	10	7.34	<.0001	0.3493	0.6540	6.133E-6

The estimates of β_1, β_2, and β_3 are close to the adaptive quadrature estimates listed in Table 3 of Pinheiro and Bates (1995). However, Pinheiro and Bates use a Cholesky-root parameterization for the random-effects variance matrix and a logarithmic parameterization for the residual variance. The PROC NLMIXED statements using their parameterization are as follows, and results are similar.

```
proc nlmixed data=theoph;
   parms ll1=-1.5 l2=0 ll3=-0.1 beta1=-3 beta2=0.5 beta3=-2.5 ls2=-0.7;
   s2   = exp(ls2);
   l1   = exp(ll1);
   l3   = exp(ll3);
   s2b1 = l1*l1*s2;
   cb12 = l2*l1*s2;
   s2b2 = (l2*l2 + l3*l3)*s2;
   cl   = exp(beta1 + b1);
   ka   = exp(beta2 + b2);
   ke   = exp(beta3);
   pred = dose*ke*ka*(exp(-ke*time)-exp(-ka*time))/cl/(ka-ke);
   model conc   ~ normal(pred,s2);
   random b1 b2 ~ normal([0,0],[s2b1,cb12,s2b2]) subject=subject;
run;
```

Example 89.2: Probit-Normal Model with Binomial Data

(View the complete code for this example (nlmex2.sas) in the example repository.)

For this example, consider the data from Weil (1970), also studied by: Williams (1975); Ochi and Prentice (1984); McCulloch (1994). In this experiment 16 pregnant rats receive a control diet and 16 receive a chemically treated diet, and the litter size for each rat is recorded after 4 and 21 days. The SAS data set follows:

```
data rats;
   input trt $ m x @@;
   if (trt='c') then do;
      x1 = 1;
      x2 = 0;
   end;
   else do;
      x1 = 0;
      x2 = 1;
   end;
   litter = _n_;
   datalines;
c 13 13   c 12 12   c  9  9   c  9  9   c  8  8   c  8  8   c 13 12   c 12 11
c 10  9   c 10  9   c  9  8   c 13 11   c  5  4   c  7  5   c 10  7   c 10  7
t 12 12   t 11 11   t 10 10   t  9  9   t 11 10   t 10  9   t 10  9   t  9  8
t  9  8   t  5  4   t  9  7   t  7  4   t 10  5   t  6  3   t 10  3   t  7  0
;
```

Here, m represents the size of the litter after 4 days, and x represents the size of the litter after 21 days. Also, indicator variables x1 and x2 are constructed for the two treatment levels.

Following McCulloch (1994), assume a latent survival model of the form

$$y_{ijk} = t_i + \alpha_{ij} + e_{ijk}$$

where i indexes treatment, j indexes litter, and k indexes newborn rats within a litter. The t_i represent treatment means, the α_{ij} represent random litter effects assumed to be iid $N(0, s_i^2)$, and the e_{ijk} represent iid residual errors, all on the latent scale.

Instead of observing the survival times y_{ijk}, assume that only the binary variable indicating whether y_{ijk} exceeds 0 is observed. If x_{ij} denotes the sum of these binary variables for the ith treatment and the jth litter, then the preceding assumptions lead to the following generalized linear mixed model:

$$x_{ij}|\alpha_{ij} \sim \text{Binomial}(m_{ij}, p_{ij})$$

where m_{ij} is the size of each litter after 4 days and

$$p_{ij} = \Phi(t_i + \alpha_{ij})$$

The PROC NLMIXED statements to fit this model are as follows:

```
proc nlmixed data=rats;
   parms t1=1 t2=1 s1=.05 s2=1;
   eta = x1*t1 + x2*t2 + alpha;
   p   = probnorm(eta);
   model x ~ binomial(m,p);
   random alpha ~ normal(0,x1*s1*s1+x2*s2*s2) subject=litter;
   estimate 'gamma2' t2/sqrt(1+s2*s2);
   predict p out=p;
run;
```

As in Example 89.1, the PROC NLMIXED statement invokes the procedure and the PARMS statement defines the parameters. The parameters for this example are the two treatment means, t1 and t2, and the two random-effect standard deviations, s1 and s2.

The indicator variables x1 and x2 are used in the program to assign the proper mean to each observation in the input data set as well as the proper variance to the random effects. Note that programming expressions are permitted inside the distributional specifications, as illustrated by the random-effects variance specified here.

The ESTIMATE statement requests an estimate of $\gamma_2 = t_2/\sqrt{1+s_2^2}$, which is a location-scale parameter from Ochi and Prentice (1984).

The PREDICT statement constructs predictions for each observation in the input data set. For this example, predictions of p and approximate standard errors of prediction are output to a SAS data set named P. These predictions are functions of the parameter estimates and the empirical Bayes estimates of the random effects α_i.

The output for this model is as follows.

Output 89.2.1 Specifications, Dimensions, and Starting Values

The NLMIXED Procedure

Specifications	
Data Set	WORK.RATS
Dependent Variable	x
Distribution for Dependent Variable	Binomial
Random Effects	alpha
Distribution for Random Effects	Normal
Subject Variable	litter
Optimization Technique	Dual Quasi-Newton
Integration Method	Adaptive Gaussian Quadrature

Dimensions	
Observations Used	32
Observations Not Used	0
Total Observations	32
Subjects	32
Max Obs per Subject	1
Parameters	4
Quadrature Points	7

Output 89.2.1 *continued*

Initial Parameters				
t1	t2	s1	s2	Negative Log Likelihood
1	1	0.05	1	54.9362323

The "Specifications" table provides basic information about this nonlinear mixed model (Output 89.2.1). The "Dimensions" table provides counts of various variables. Note that each observation in the data comprises a separate subject. Using the starting values in the "Parameters" table, PROC NLMIXED determines that the log-likelihood function can be approximated with sufficient accuracy with a seven-point quadrature rule.

Output 89.2.2 Iteration History for Probit-Normal Model

Iteration History					
Iteration	Calls	Negative Log Likelihood	Difference	Maximum Gradient	Slope
1	4	53.9933934	0.942839	11.0326	-81.9428
2	6	52.8753530	1.11804	2.14895	-2.86277
3	9	52.6350386	0.240314	0.32996	-1.05049
4	11	52.6319939	0.003045	0.12293	-0.00672
5	14	52.6313583	0.000636	0.028246	-0.00352
6	18	52.6313174	0.000041	0.013551	-0.00023
7	21	52.6313115	5.839E-6	0.000603	-0.00001
8	24	52.6313115	9.45E-9	0.000022	-1.68E-8

NOTE: GCONV convergence criterion satisfied.

The "Iteration History" table indicates successful convergence in 8 iterations (Output 89.2.2).

Output 89.2.3 Fit Statistics for Probit-Normal Model

Fit Statistics	
-2 Log Likelihood	105.3
AIC (smaller is better)	113.3
AICC (smaller is better)	114.7
BIC (smaller is better)	119.1

The "Fit Statistics" table lists useful statistics based on the maximized value of the log likelihood (Output 89.2.3).

Output 89.2.4 Parameter Estimates for Probit-Normal Model

Parameter Estimates								
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
t1	1.3063	0.1685	31	7.75	<.0001	0.9626	1.6499	-0.00002
t2	0.9475	0.3055	31	3.10	0.0041	0.3244	1.5705	9.283E-6
s1	0.2403	0.3015	31	0.80	0.4315	-0.3746	0.8552	0.000014
s2	1.0292	0.2988	31	3.44	0.0017	0.4198	1.6386	-3.16E-6

The "Parameter Estimates" table indicates significance of all the parameters except S1 (Output 89.2.4).

Output 89.2.5 Additional Estimates

Additional Estimates								
Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
gamma2	0.6603	0.2165	31	3.05	0.0047	0.05	0.2186	1.1019

The "Additional Estimates" table displays results from the ESTIMATE statement (Output 89.2.5). The estimate of γ_2 equals 0.66, agreeing with that obtained by McCulloch (1994). The standard error 0.22 is computed using the delta method (Billingsley 1986; Cox 1998).

Not shown is the P data set, which contains the original 32 observations and predictions of the p_{ij}.

Example 89.3: Probit-Normal Model with Ordinal Data

(View the complete code for this example (nlmex3.sas) in the example repository.)

The data for this example are from Ezzet and Whitehead (1991), who describe a crossover experiment on two groups of patients using two different inhaler devices (A and B). Patients from group 1 used device A for one week and then device B for another week. Patients from group 2 used the devices in reverse order. The data entered as a SAS data set are as follows:

```
data inhaler;
   input clarity group time freq @@;
   gt = group*time;
   sub = floor((_n_+1)/2);
   datalines;
1 0 0 59   1 0 1 59   1 0 0 35   2 0 1 35   1 0 0  3   3 0 1  3   1 0 0  2
4 0 1  2   2 0 0 11   1 0 1 11   2 0 0 27   2 0 1 27   2 0 0  2   3 0 1  2
2 0 0  1   4 0 1  1   4 0 0  1   1 0 1  1   4 0 0  1   2 0 1  1   1 1 0 63
1 1 1 63   1 1 0 13   2 1 1 13   2 1 0 40   1 1 1 40   2 1 0 15   2 1 1 15
3 1 0  7   1 1 1  7   3 1 0  2   2 1 1  2   3 1 0  1   3 1 1  1   4 1 0  2
1 1 1  2   4 1 0  1   3 1 1  1
;
```

The response measurement, clarity, is the patients' assessment on the clarity of the leaflet instructions for the devices. The clarity variable is on an ordinal scale, with 1=easy, 2=only clear after rereading, 3=not very clear, and 4=confusing. The group variable indicates the treatment group, and the time variable indicates the

time of measurement. The freq variable indicates the number of patients with exactly the same responses. A variable gt is created to indicate a group-by-time interaction, and a variable sub is created to indicate patients.

As in the previous example and in Hedeker and Gibbons (1994), assume an underlying latent continuous variable, here with the form

$$y_{ij} = \beta_0 + \beta_1 g_i + \beta_2 t_j + \beta_3 g_i t_j + u_i + e_{ij}$$

where i indexes patient and j indexes the time period, g_i indicates groups, t_j indicates time, u_i is a patient-level normal random effect, and e_{ij} are iid normal errors. The βs are unknown coefficients to be estimated.

Instead of observing y_{ij}, however, you observe only whether it falls in one of the four intervals: $(-\infty, 0)$, $(0, I1)$, $(I1, I1 + I2)$, or $(I1 + I2, \infty)$, where $I1$ and $I2$ are both positive. The resulting category is the value assigned to the clarity variable.

The following code sets up and fits this ordinal probit model:

```
proc nlmixed data=inhaler corr ecorr;
   parms b0=0 b1=0 b2=0 b3=0 sd=1 i1=1 i2=1;
   bounds i1 > 0, i2 > 0;
   eta = b0 + b1*group + b2*time + b3*gt + u;
   if (clarity=1) then p = probnorm(-eta);
   else if (clarity=2) then
      p = probnorm(i1-eta) - probnorm(-eta);
   else if (clarity=3) then
      p = probnorm(i1+i2-eta) - probnorm(i1-eta);
   else p = 1 - probnorm(i1+i2-eta);
   if (p > 1e-8) then ll = log(p);
   else ll = -1e20;
   model clarity ~ general(ll);
   random u ~ normal(0,sd*sd) subject=sub;
   replicate freq;
   estimate 'thresh2' i1;
   estimate 'thresh3' i1 + i2;
   estimate 'icc' sd*sd/(1+sd*sd);
run;
```

The PROC NLMIXED statement specifies the input data set and requests correlations both for the parameter estimates (CORR option) and for the additional estimates specified with ESTIMATE statements (ECORR option).

The parameters as defined in the PARMS statement are as follows: b0 (overall intercept), b1 (group main effect), B2 (time main effect), b3 (group-by-time interaction), sd (standard deviation of the random effect), i1 (increment between first and second thresholds), and i2 (increment between second and third thresholds). The BOUNDS statement restricts i1 and i2 to be positive.

The SAS programming statements begin by defining the linear predictor eta, which is a linear combination of the b parameters and a single random effect u. The next statements define the ordinal likelihood according to the clarity variable, eta, and the increment variables. An error trap is included in case the likelihood becomes too small.

A general log-likelihood specification is used in the MODEL statement, and the RANDOM statement defines the random effect u to have standard deviation sd and subject variable sub. The REPLICATE statement indicates that data for each subject should be replicated according to the freq variable.

The ESTIMATE statements specify the second and third thresholds in terms of the increment variables (the first threshold is assumed to equal zero for model identifiability). Also computed is the intraclass correlation.

The output is as follows.

Output 89.3.1 Specifications for Ordinal Data Model

The NLMIXED Procedure

Specifications	
Data Set	WORK.INHALER
Dependent Variable	clarity
Distribution for Dependent Variable	General
Random Effects	u
Distribution for Random Effects	Normal
Subject Variable	sub
Replicate Variable	freq
Optimization Technique	Dual Quasi-Newton
Integration Method	Adaptive Gaussian Quadrature

The "Specifications" table echoes some primary information specified for this nonlinear mixed model (Output 89.3.1). Because the log-likelihood function was expressed with SAS programming statements, the distribution is displayed as *General* in the "Specifications" table.

The "Dimensions" table reveals a total of 286 subjects, which is the sum of the values of the FREQ variable for the second time point. Five quadrature points are selected for log-likelihood evaluation (Output 89.3.2).

Output 89.3.2 Dimensions Table for Ordinal Data Model

Dimensions	
Observations Used	38
Observations Not Used	0
Total Observations	38
Subjects	286
Max Obs per Subject	2
Parameters	7
Quadrature Points	5

Output 89.3.3 Parameter Starting Values and Negative Log Likelihood

Initial Parameters							
b0	**b1**	**b2**	**b3**	**sd**	**i1**	**i2**	**Negative Log Likelihood**
0	0	0	0	1	1	1	538.484276

The "Parameters" table lists the simple starting values for this problem (Output 89.3.3). The "Iteration History" table indicates successful convergence in 13 iterations (Output 89.3.4).

Output 89.3.4 Iteration History

Iteration History					
Iteration	Calls	Negative Log Likelihood	Difference	Maximum Gradient	Slope
1	4	476.3825	62.10176	43.7506	-1431.40
2	7	463.2282	13.15431	14.2465	-106.753
3	9	458.5281	4.70008	48.3132	-33.0389
4	11	450.9757	7.552383	22.6010	-40.9954
5	14	448.0127	2.963033	14.8688	-16.7453
6	17	447.2452	0.767549	7.77419	-2.26743
7	19	446.7277	0.517483	3.79353	-1.59278
8	22	446.5183	0.209396	0.86864	-0.37801
9	26	446.5145	0.003745	0.32857	-0.02356
10	29	446.5133	0.001187	0.056778	-0.00183
11	32	446.5133	0.000027	0.010785	-0.00004
12	35	446.5133	3.956E-6	0.004922	-5.41E-6
13	38	446.5133	1.989E-7	0.000470	-4E-7

NOTE: GCONV convergence criterion satisfied.

Output 89.3.5 Fit Statistics for Ordinal Data Model

Fit Statistics	
-2 Log Likelihood	893.0
AIC (smaller is better)	907.0
AICC (smaller is better)	910.8
BIC (smaller is better)	932.6

The "Fit Statistics" table lists the log likelihood and information criteria for model comparisons (Output 89.3.5).

Output 89.3.6 Parameter Estimates at Convergence

Parameter Estimates								
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
b0	-0.6364	0.1342	285	-4.74	<.0001	-0.9006	-0.3722	0.000470
b1	0.6007	0.1770	285	3.39	0.0008	0.2523	0.9491	0.000265
b2	0.6015	0.1582	285	3.80	0.0002	0.2900	0.9129	0.000080
b3	-1.4817	0.2385	285	-6.21	<.0001	-1.9512	-1.0122	0.000102
sd	0.6599	0.1312	285	5.03	<.0001	0.4017	0.9181	-0.00009
i1	1.7450	0.1474	285	11.84	<.0001	1.4548	2.0352	0.000202
i2	0.5985	0.1427	285	4.19	<.0001	0.3176	0.8794	0.000087

The "Parameter Estimates" table indicates significance of all the parameters (Output 89.3.6).

Output 89.3.7 Threshold and Intraclass Correlation Estimates

Additional Estimates								
Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
thresh2	1.7450	0.1474	285	11.84	<.0001	0.05	1.4548	2.0352
thresh3	2.3435	0.2073	285	11.31	<.0001	0.05	1.9355	2.7516
icc	0.3034	0.08402	285	3.61	0.0004	0.05	0.1380	0.4687

The "Additional Estimates" table displays results from the ESTIMATE statements (Output 89.3.7).

Example 89.4: Poisson-Normal Model with Count Data

(View the complete code for this example (nlmex4.sas) in the example repository.)

This example uses the pump failure data of Gaver and O'Muircheartaigh (1987). The number of failures and the time of operation are recorded for 10 pumps. Each of the pumps is classified into one of two groups corresponding to either continuous or intermittent operation. The data are as follows:

```
data pump;
   input y t group;
   pump = _n_;
   logtstd = log(t) - 2.4564900;
   datalines;
 5  94.320 1
 1  15.720 2
 5  62.880 1
14 125.760 1
 3   5.240 2
19  31.440 1
 1   1.048 2
 1   1.048 2
 4   2.096 2
22  10.480 2
;
```

Each row denotes data for a single pump, and the variable logtstd contains the centered operation times.

Letting y_{ij} denote the number of failures for the jth pump in the ith group, Draper (1996) considers the following hierarchical model for these data:

$$
\begin{aligned}
y_{ij}|\lambda_{ij} &\sim \text{Poisson}(\lambda_{ij}) \\
\log \lambda_{ij} &= \alpha_i + \beta_i(\log t_{ij} - \overline{\log t}) + e_{ij} \\
e_{ij}|\sigma^2 &\sim \text{Normal}(0, \sigma^2)
\end{aligned}
$$

The model specifies different intercepts and slopes for each group, and the random effect is a mechanism for accounting for overdispersion.

The corresponding PROC NLMIXED statements are as follows:

```
proc nlmixed data=pump;
   parms logsig 0 beta1 1 beta2 1 alpha1 1 alpha2 1;
   if (group = 1) then eta = alpha1 + beta1*logtstd + e;
   else eta = alpha2 + beta2*logtstd + e;
   lambda = exp(eta);
   model y ~ poisson(lambda);
   random e ~ normal(0,exp(2*logsig)) subject=pump;
   estimate 'alpha1-alpha2' alpha1-alpha2;
   estimate 'beta1-beta2' beta1-beta2;
run;
```

The selected output is as follows.

Output 89.4.1 Dimensions Table for Poisson-Normal Model

The NLMIXED Procedure

Dimensions	
Observations Used	10
Observations Not Used	0
Total Observations	10
Subjects	10
Max Obs per Subject	1
Parameters	5
Quadrature Points	5

The "Dimensions" table indicates that data for 10 pumps are used with one observation for each (Output 89.4.1).

Output 89.4.2 Iteration History for Poisson-Normal Model

Iteration History

Iteration	Calls	Negative Log Likelihood	Difference	Maximum Gradient	Slope
1	4	30.6986932	2.162768	5.10725	-91.6020
2	9	30.0255468	0.673146	2.76174	-11.0489
3	12	29.7263250	0.299222	2.99040	-2.36048
4	16	28.7390263	0.987299	2.07443	-3.93678
5	18	28.3161933	0.422833	0.61253	-0.63084
6	21	28.0956400	0.220553	0.46216	-0.52684
7	24	28.0438024	0.051838	0.40505	-0.10018
8	27	28.0357134	0.008089	0.13506	-0.01875
9	30	28.0339250	0.001788	0.026279	-0.00514
10	33	28.0338744	0.000051	0.004020	-0.00012
11	36	28.0338727	1.681E-6	0.002864	-5.09E-6
12	39	28.0338724	3.199E-7	0.000147	-6.87E-7
13	42	28.0338724	2.532E-9	0.000017	-5.75E-9

NOTE: GCONV convergence criterion satisfied.

The "Iteration History" table indicates successful convergence in 13 iterations (Output 89.4.2).

Output 89.4.3 Fit Statistics for Poisson-Normal Model

Fit Statistics	
-2 Log Likelihood	56.1
AIC (smaller is better)	66.1
AICC (smaller is better)	81.1
BIC (smaller is better)	67.6

The "Fit Statistics" table lists the final log likelihood and associated information criteria (Output 89.4.3).

Output 89.4.4 Parameter Estimates and Additional Estimates

Parameter Estimates

Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
logsig	-0.3161	0.3213	9	-0.98	0.3508	-1.0429	0.4107	-0.00002
beta1	-0.4256	0.7473	9	-0.57	0.5829	-2.1162	1.2649	-0.00002
beta2	0.6097	0.3814	9	1.60	0.1443	-0.2530	1.4725	-1.61E-6
alpha1	2.9644	1.3826	9	2.14	0.0606	-0.1632	6.0921	-5.25E-6
alpha2	1.7992	0.5492	9	3.28	0.0096	0.5568	3.0415	-5.73E-6

Additional Estimates

Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Alpha	Lower	Upper
alpha1-alpha2	1.1653	1.4855	9	0.78	0.4529	0.05	-2.1952	4.5257
beta1-beta2	-1.0354	0.8389	9	-1.23	0.2484	0.05	-2.9331	0.8623

The "Parameter Estimates" and "Additional Estimates" tables list the maximum likelihood estimates for each of the parameters and two differences (Output 89.4.4). The point estimates for the mean parameters agree fairly closely with the Bayesian posterior means reported by Draper (1996); however, the likelihood-based standard errors are roughly half the Bayesian posterior standard deviations. This is most likely due to the fact that the Bayesian standard deviations account for the uncertainty in estimating σ^2, whereas the likelihood values plug in its estimated value. This downward bias can be corrected somewhat by using the t_9 distribution shown here.

Example 89.5: Failure Time and Frailty Model

(View the complete code for this example (nlmex5.sas) in the example repository.)

In this example an accelerated failure time model with proportional hazard is fitted with and without random effects. The data are from the "Getting Started" example of PROC LIFEREG; see Chapter 76, "The LIFEREG Procedure." Thirty-eight patients are divided into two groups of equal size, and different pain relievers are assigned to each group. The outcome reported is the time in minutes until headache relief. The variable censor indicates whether relief was observed during the course of the observation period (censor = 0) or whether the observation is censored (censor = 1). The SAS DATA step for these data is as follows:

```
data headache;
   input minutes group censor @@;
   patient = _n_;
   datalines;
11  1  0    12  1  0    19  1  0    19  1  0
19  1  0    19  1  0    21  1  0    20  1  0
21  1  0    21  1  0    20  1  0    21  1  0
20  1  0    21  1  0    25  1  0    27  1  0
30  1  0    21  1  1    24  1  1    14  2  0
16  2  0    16  2  0    21  2  0    21  2  0
23  2  0    23  2  0    23  2  0    23  2  0
25  2  1    23  2  0    24  2  0    24  2  0
26  2  1    32  2  1    30  2  1    30  2  0
32  2  1    20  2  1
;
```

In modeling survival data, censoring of observations must be taken into account carefully. In this example, only right censoring occurs. If $g(t, \boldsymbol{\beta})$, $h(t, \boldsymbol{\beta})$, and $G(t, \boldsymbol{\beta})$ denote the density of failure, the hazard function, and the survival distribution function at time t, respectively, then the log likelihood can be written as

$$\begin{aligned} l(\boldsymbol{\beta}; \mathbf{t}) &= \sum_{i \in U_u} \log g(t_i, \boldsymbol{\beta}) + \sum_{i \in U_c} \log G(t_i, \boldsymbol{\beta}) \\ &= \sum_{i \in U_u} \log h(t_i, \boldsymbol{\beta}) + \sum_{i=1}^{n} \log G(t_i, \boldsymbol{\beta}) \end{aligned}$$

(See Cox and Oakes 1984, Ch. 3.) In these expressions U_u is the set of uncensored observations, U_c is the set of censored observations, and n denotes the total sample size.

The proportional hazards specification expresses the hazard in terms of a baseline hazard, multiplied by a constant. In this example the hazard is that of a Weibull model and is parameterized as $h(t, \boldsymbol{\beta}) = \gamma\alpha(\alpha t)^{\gamma-1}$ and $\alpha = \exp\{-\mathbf{x}'\boldsymbol{\beta}\}$.

The linear predictor is set equal to the intercept in the reference group (group = 2); this defines the baseline hazard. The corresponding distribution of survival past time t is $G(t, \boldsymbol{\beta}) = \exp\{-(\alpha t)^{\gamma}\}$. See Cox and Oakes (1984, Table 2.1) and the section "Supported Distributions" in Chapter 76, "The LIFEREG Procedure," for this and other survival distribution models and various parameterizations.

The following NLMIXED statements fit this accelerated failure time model and estimate the cumulative distribution function of time to headache relief:

```
proc nlmixed data=headache;
   bounds gamma > 0;
   linp  = b0 - b1*(group-2);
   alpha = exp(-linp);
   G_t   = exp(-(alpha*minutes)**gamma);
   g     = gamma*alpha*((alpha*minutes)**(gamma-1))*G_t;
   ll    = (censor=0)*log(g) + (censor=1)*log(G_t);
   model minutes ~ general(ll);
   predict 1-G_t out=cdf;
run;
```

Output 89.5.1 Specifications Table for Fixed-Effects Failure Time Model

The NLMIXED Procedure

Specifications	
Data Set	WORK.HEADACHE
Dependent Variable	minutes
Distribution for Dependent Variable	General
Optimization Technique	Dual Quasi-Newton
Integration Method	None

The "Specifications" table shows that no integration is required, since the model does not contain random effects (Output 89.5.1).

Output 89.5.2 Negative Log Likelihood with Default Starting Values

Initial Parameters			
gamma	**b0**	**b1**	**Negative Log Likelihood**
1	1	1	263.990327

No starting values were given for the three parameters. The NLMIXED procedure assigns the default value of 1.0 in this case. The negative log likelihood based on these starting values is shown in Output 89.5.2.

Output 89.5.3 Iteration History for Fixed-Effects Failure Time Model

Iteration History					
Iteration	Calls	Negative Log Likelihood	Difference	Maximum Gradient	Slope
1	4	169.2443	94.74602	22.5599	-2230.83
2	8	142.8735	26.3708	14.8863	-3.64643
3	11	140.6337	2.239814	11.2523	-9.49454
4	15	122.8907	17.74304	19.4496	-2.50807
5	17	121.3970	1.493699	13.8558	-4.55427
6	20	120.6238	0.773116	13.6706	-1.38064
7	22	119.2782	1.345647	15.7801	-1.69072
8	26	116.2713	3.006871	26.9403	-3.25290
9	30	109.4274	6.843925	19.8838	-6.92890
10	35	103.2981	6.129298	12.1565	-4.96054
11	40	101.6862	1.611863	14.2487	-4.34059
12	42	100.0279	1.658364	11.6985	-13.2049
13	46	99.9189048	0.108971	3.60255	-0.55176
14	49	99.8738836	0.045021	0.17071	-0.16645
15	52	99.8736392	0.000244	0.050822	-0.00041
16	55	99.8736351	4.071E-6	0.000705	-6.9E-6
17	58	99.8736351	6.1E-10	4.768E-6	-1.23E-9

NOTE: GCONV convergence criterion satisfied.

The "Iteration History" table shows that the procedure converges after 17 iterations and 34 evaluations of the objective function (Output 89.5.3).

Output 89.5.4 Fit Statistics and Parameter Estimates

Fit Statistics	
-2 Log Likelihood	199.7
AIC (smaller is better)	205.7
AICC (smaller is better)	206.5
BIC (smaller is better)	210.7

Parameter Estimates								
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
gamma	4.7128	0.6742	38	6.99	<.0001	3.3479	6.0777	5.327E-8
b0	3.3091	0.05885	38	56.23	<.0001	3.1900	3.4283	-4.77E-6
b1	-0.1933	0.07856	38	-2.46	0.0185	-0.3523	-0.03426	-1.22E-6

The parameter estimates and their standard errors shown in Output 89.5.4 are identical to those obtained with the LIFEREG procedure and the following statements:

```
proc lifereg data=headache;
   class group;
   model minutes*censor(1) = group / dist=weibull;
   output out=new cdf=prob;
run;
```

The t statistic and confidence limits are based on 38 degrees of freedom. The LIFEREG procedure computes z intervals for the parameter estimates.

For the two groups you obtain

$$\widehat{\alpha}(\text{group} = 1) = \exp\{-3.3091 + 0.1933\} = 0.04434$$
$$\widehat{\alpha}(\text{group} = 2) = \exp\{-3.3091\} = 0.03655$$

The probabilities of headache relief by t minutes are estimated as

$$1 - G(t, \text{group} = 1) = 1 - \exp\{-(0.04434 * t)^{4.7128}\}$$
$$1 - G(t, \text{group} = 2) = 1 - \exp\{-(0.03655 * t)^{4.7128}\}$$

These probabilities, calculated at the observed times, are shown for the two groups in Output 89.5.5 and printed with the following statements:

```
proc print data=cdf;
   var group censor patient minutes pred;
run;
```

Since the slope estimate is negative with p-value of 0.0185, you can infer that pain reliever 1 leads to overall significantly faster relief, but the estimated probabilities give no information about patient-to-patient variation within and between groups. For example, while pain reliever 1 provides faster relief overall, some patients in group 2 might respond more quickly than some patients in group 1. A frailty model enables you to accommodate and estimate patient-to-patient variation in health status by introducing random effects into a subject's hazard function.

Output 89.5.5 Estimated Cumulative Distribution Function

Obs	group	censor	patient	minutes	Pred
1	1	0	1	11	0.03336
2	1	0	2	12	0.04985
3	1	0	3	19	0.35975
4	1	0	4	19	0.35975
5	1	0	5	19	0.35975
6	1	0	6	19	0.35975
7	1	0	7	21	0.51063
8	1	0	8	20	0.43325
9	1	0	9	21	0.51063
10	1	0	10	21	0.51063
11	1	0	11	20	0.43325
12	1	0	12	21	0.51063
13	1	0	13	20	0.43325
14	1	0	14	21	0.51063
15	1	0	15	25	0.80315
16	1	0	16	27	0.90328
17	1	0	17	30	0.97846
18	1	1	18	21	0.51063
19	1	1	19	24	0.73838
20	2	0	20	14	0.04163
21	2	0	21	16	0.07667
22	2	0	22	16	0.07667
23	2	0	23	21	0.24976
24	2	0	24	21	0.24976
25	2	0	25	23	0.35674
26	2	0	26	23	0.35674
27	2	0	27	23	0.35674
28	2	0	28	23	0.35674
29	2	1	29	25	0.47982
30	2	0	30	23	0.35674
31	2	0	31	24	0.41678
32	2	0	32	24	0.41678
33	2	1	33	26	0.54446
34	2	1	34	32	0.87656
35	2	1	35	30	0.78633
36	2	0	36	30	0.78633
37	2	1	37	32	0.87656
38	2	1	38	20	0.20414

The following statements model the hazard for patient i in terms of $\alpha_i = \exp\{-\mathbf{x}_i'\boldsymbol{\beta} - z_i\}$, where z_i is a (normal) random patient effect. Notice that the only difference from the previous NLMIXED statements are the RANDOM statement and the addition of z in the linear predictor. The empirical Bayes estimates of the random effect (RANDOM statement), the parameter estimates (ODS OUTPUT statement), and the estimated cumulative distribution function (PREDICT statement) are saved to subsequently graph the patient-specific distribution functions.

```
ods output ParameterEstimates=est;
proc nlmixed data=headache;
   bounds gamma > 0;
   linp  = b0 - b1*(group-2) + z;
   alpha = exp(-linp);
   G_t   = exp(-(alpha*minutes)**gamma);
   g     = gamma*alpha*((alpha*minutes)**(gamma-1))*G_t;
   ll = (censor=0)*log(g) + (censor=1)*log(G_t);
   model minutes ~ general(ll);
   random z ~ normal(0,exp(2*logsig)) subject=patient out=EB;
   predict 1-G_t out=cdf;
run;
```

Output 89.5.6 Specifications for Random Frailty Model

The NLMIXED Procedure

Specifications	
Data Set	WORK.HEADACHE
Dependent Variable	minutes
Distribution for Dependent Variable	General
Random Effects	z
Distribution for Random Effects	Normal
Subject Variable	patient
Optimization Technique	Dual Quasi-Newton
Integration Method	Adaptive Gaussian Quadrature

The "Specifications" table shows that the objective function is computed by adaptive Gaussian quadrature because of the presence of random effects (compare Output 89.5.6 and Output 89.5.1). The "Dimensions" table reports that nine quadrature points are being used to integrate over the random effects (Output 89.5.7).

Output 89.5.7 Dimensions Table for Random Frailty Model

Dimensions	
Observations Used	38
Observations Not Used	0
Total Observations	38
Subjects	38
Max Obs per Subject	1
Parameters	4
Quadrature Points	9

Output 89.5.8 Iteration History for Random Frailty Model

Iteration History					
Iteration	Calls	Negative Log Likelihood	Difference	Maximum Gradient	Slope
1	9	142.1214	28.82225	12.1448	-88.8664
2	12	136.4404	5.681042	25.9310	-65.7217
3	16	122.9720	13.46833	46.5655	-146.887
4	19	120.9048	2.067216	23.7794	-94.2862
5	23	109.2241	11.68068	57.6549	-92.4075
6	26	105.0647	4.159411	4.82465	-19.5879
7	28	101.9022	3.162526	14.1287	-6.33767
8	31	99.6907395	2.211468	7.67682	-3.42364
9	34	99.3654033	0.325336	5.68920	-0.93978
10	37	99.2602178	0.105185	0.31764	-0.23408
11	40	99.2544340	0.005784	1.17351	-0.00556
12	42	99.2456973	0.008737	0.24741	-0.00871
13	45	99.2445445	0.001153	0.10494	-0.00218
14	48	99.2444958	0.000049	0.005646	-0.00010
15	51	99.2444957	9.147E-8	0.000271	-1.84E-7

NOTE: GCONV convergence criterion satisfied.

The procedure converges after 15 iterations (Output 89.5.8). The achieved –2 log likelihood is only 1.2 less than that in the model without random effects (compare Output 89.5.9 and Output 89.5.4). Compared to a chi-square distribution with one degree of freedom, the addition of the random effect appears not to improve the model significantly. You must exercise care, however, in interpreting likelihood ratio tests when the value under the null hypothesis falls on the boundary of the parameter space (see, for example, Self and Liang 1987).

Output 89.5.9 Fit Statistics for Random Frailty Model

Fit Statistics	
-2 Log Likelihood	198.5
AIC (smaller is better)	206.5
AICC (smaller is better)	207.7
BIC (smaller is better)	213.0

Output 89.5.10 Parameter Estimates

Parameter Estimates								
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
gamma	6.2867	2.1334	37	2.95	0.0055	1.9640	10.6093	-1.89E-7
b0	3.2786	0.06576	37	49.86	<.0001	3.1453	3.4118	0.000271
b1	-0.1761	0.08264	37	-2.13	0.0398	-0.3436	-0.00867	0.000111
logsig	-1.9027	0.5273	37	-3.61	0.0009	-2.9710	-0.8343	0.000027

The estimate of the Weibull parameter has changed drastically from the model without random effects (compare Output 89.5.10 and Output 89.5.4). The variance of the patient random effect is $\exp\{-2\times 1.9027\} = 0.02225$. The listing in Output 89.5.11 shows the empirical Bayes estimates of the random effects. These are the adjustments made to the linear predictor in order to obtain a patient's survival distribution. The listing is produced with the following statements:

```
proc print data=eb;
   var Patient Effect Estimate StdErrPred;
run;
```

Output 89.5.11 Empirical Bayes Estimates of Random Effects

Obs	patient	Effect	Estimate	StdErrPred
1	1	z	-0.13597	0.23249
2	2	z	-0.13323	0.22793
3	3	z	-0.06294	0.13813
4	4	z	-0.06294	0.13813
5	5	z	-0.06294	0.13813
6	6	z	-0.06294	0.13813
7	7	z	-0.02568	0.11759
8	8	z	-0.04499	0.12618
9	9	z	-0.02568	0.11759
10	10	z	-0.02568	0.11759
11	11	z	-0.04499	0.12618
12	12	z	-0.02568	0.11759
13	13	z	-0.04499	0.12618
14	14	z	-0.02568	0.11759
15	15	z	0.05980	0.11618
16	16	z	0.10458	0.12684
17	17	z	0.17147	0.14550
18	18	z	0.06471	0.13807
19	19	z	0.11157	0.14604
20	20	z	-0.13406	0.22899
21	21	z	-0.12698	0.21666
22	22	z	-0.12698	0.21666
23	23	z	-0.08506	0.15701
24	24	z	-0.08506	0.15701
25	25	z	-0.05797	0.13294
26	26	z	-0.05797	0.13294
27	27	z	-0.05797	0.13294
28	28	z	-0.05797	0.13294
29	29	z	0.06420	0.13956
30	30	z	-0.05797	0.13294
31	31	z	-0.04266	0.12390
32	32	z	-0.04266	0.12390
33	33	z	0.07618	0.14132
34	34	z	0.16292	0.16459
35	35	z	0.13193	0.15528
36	36	z	0.06327	0.12124
37	37	z	0.16292	0.16459
38	38	z	0.02074	0.14160

The predicted values and patient-specific survival distributions can be plotted with the SAS code that follows:

```
proc transpose data=est(keep=estimate)
    out=trest(rename=(col1=gamma col2=b0 col3=b1));
run;

data pred;
   merge eb(keep=estimate) headache(keep=patient group);
   array pp{2} pred1-pred2;
   if _n_ = 1 then set trest(keep=gamma b0 b1);
   do time=11 to 32;
      linp      = b0 - b1*(group-2) + estimate;
      pp{group} = 1-exp(- (exp(-linp)*time)**gamma);
      symbolid  = patient+1;
      output;
   end;
   keep pred1 pred2 time patient;
run;

data pred;
   merge pred
         cdf(where  = (group=1)
             rename = (pred=pcdf1 minutes=minutes1)
             keep   = pred minutes group)
         cdf(where  = (group=2)
             rename = (pred=pcdf2 minutes=minutes2)
             keep   = pred minutes group);
   drop group;
run;

proc sgplot data=pred noautolegend;
   label minutes1='Minutes to Headache Relief'
         pcdf1   ='Estimated Patient-specific CDF';
   series  x=time     y=pred1  /
           group=patient
           lineattrs=(pattern=solid color=black);
   series  x=time     y=pred2  /
           group=patient
           lineattrs=(pattern=dash color=black);
   scatter x=minutes1 y=pcdf1  /
           markerattrs=(symbol=CircleFilled size=9);
   scatter x=minutes2 y=pcdf2  /
           markerattrs=(symbol=Circle       size=9);
run;
```

The separation of the distribution functions by groups is evident in Output 89.5.12. Most of the distributions of patients in the first group are to the left of the distributions in the second group. The separation is not complete, however. Several patients who are assigned the second pain reliever experience headache relief more quickly than patients assigned to the first group.

Output 89.5.12 Patient-Specific CDFs and Predicted Values. Pain Reliever 1: Solid Lines, Closed Circles; Pain Reliever 2: Dashed Lines, Open Circles.

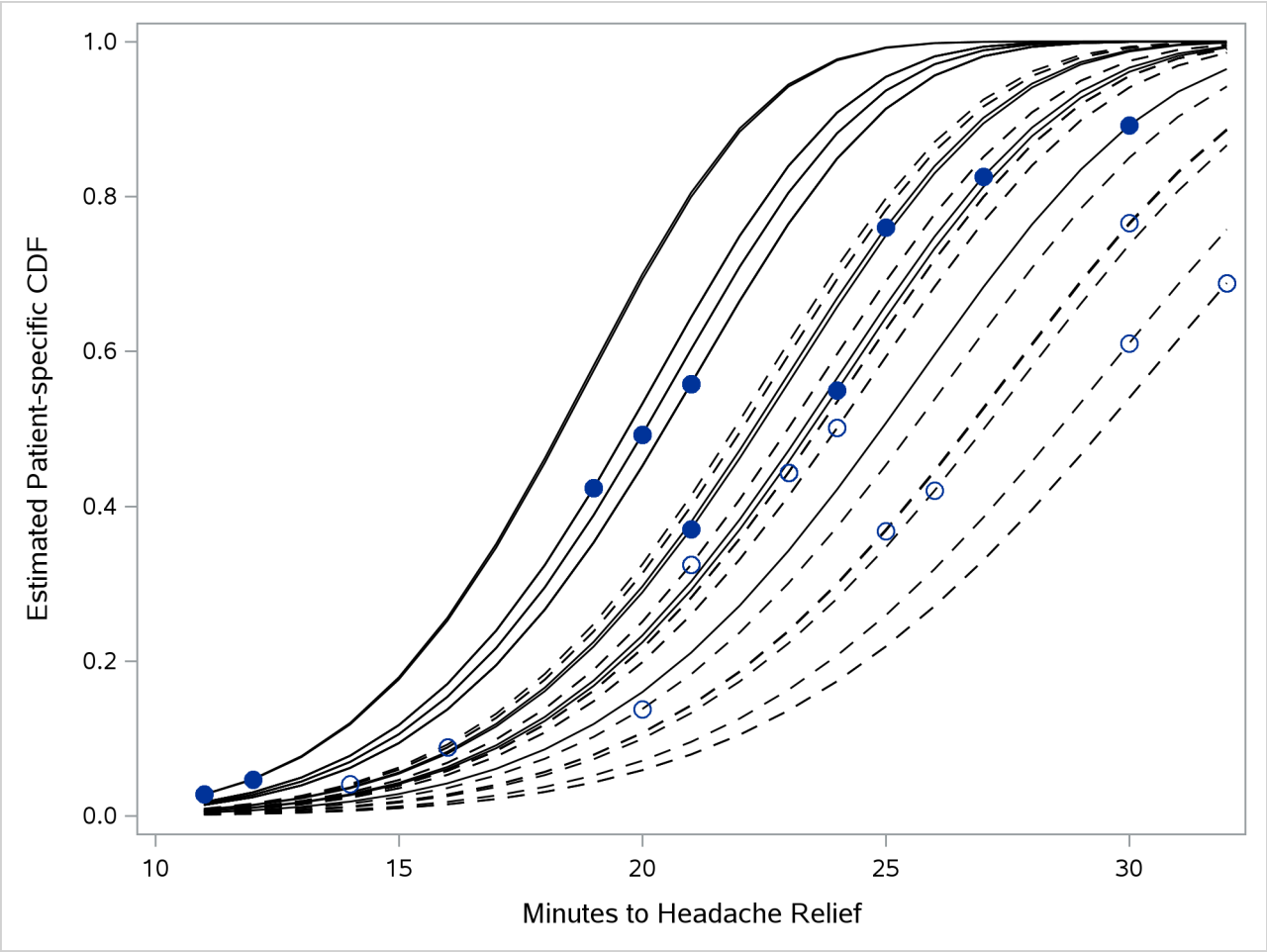

Example 89.6: Simulated Nested Linear Random-Effects Model

(View the complete code for this example (nlmex6.sas) in the example repository.)

This simulation study example demonstrates how to fit a hierarchical model with PROC NLMIXED by using a simple two-level nested linear model. In this example, the data are generated from a simple two-level nested hierarchical normal model (Littell et al. 2006). Here the conditional distribution of the response variable, given the random effects, is normal. The mean of this conditional normal distribution is a linear function of a fixed parameter and the random effects. In this simulation, only the intercept is used as a fixed parameter. Also, the first-level random effects are assumed to follow a normal distribution. Similarly, the second-level random effects that are nested within the first level follow a normal distribution. The model can

be represented as follows:

$$
\begin{aligned}
y_{ij}|\beta_0, v_i, v_{j(i)} &\sim N(\mu_{ij}, \sigma^2) \\
\mu_{ij} &= \beta_0 + v_i + v_{j(i)} \\
v_i &\sim N(0, \sigma_a^2) \\
v_{j(i)} &\sim N(0, \sigma_b^2)
\end{aligned}
$$

The simulation code is as follows:

```
%let na = 100;
%let nb = 5;
%let nr = 2;
data nested;
   do A = 1 to &na;
      err1 = 3*rannor(339205);
      do B = 1 to &nb;
         err2 = 2*rannor(0);
         do rep = 1 to &nr;
            err3 = 1*rannor(0);
            resp = 10 + err1 + err2 + err3;
            output;
         end;
      end;
   end;
run;
```

You can use PROC NLMIXED to fit the preceding nested model as follows by using two RANDOM statements:

```
proc nlmixed data = nested;
   bounds vara >=0, varb_a >=0;
   mean = intercept + aeffect + beffect;
   model resp ~ normal (mean, s2);
   random aeffect ~ normal(0,vara) subject = A;
   random beffect ~ normal(0,varb_a) subject = B(A);
run;
```

The MODEL statement specifies the response variable distribution as normal. The linear relationship between the mean of this normal distribution and the fixed and random effects is specified by using a SAS programming statement. The BOUNDS statement specifies the nonnegative constraints on the variance parameters.

PROC NLMIXED with multiple RANDOM statements is used to fit hierarchical (nested) models. In this example, two RANDOM statements are used to fit a two-level nested model. The first RANDOM statement identifies aeffect as the first-level random effect, v_i, and it is followed by the subject variable A. The subject variable A indexes the number of distinct and independent subjects at the first level. Then the second RANDOM statement identifies the beffect, $v_{j(i)}$, as the second-level random effect, which is nested within the first level. The subject variable specification in the second RANDOM statement defines the nested structure. In this example, the subject variable B is nested within A.

The results from the analysis follow.

Output 89.6.1 Model Specifications for a Two-Level Nested Model

The NLMIXED Procedure

Specifications	
Data Set	WORK.NESTED
Dependent Variable	resp
Distribution for Dependent Variable	Normal
Random Effects	aeffect, beffect
Distribution for Random Effects	Normal, Normal
Subject Variables	A, B
Optimization Technique	Dual Quasi-Newton
Integration Method	Adaptive Gaussian Quadrature

The "Specifications" table (Output 89.6.1) lists the setup of the nested model. It lists all the random effects and their distribution along with the SUBJECT= variable in the nested sequence. Random effects that are listed in the specifications table are separated by a comma, indicating that aeffect is the first-level random effect, followed by the second-level random effect, beffect, which is nested within the first level. The same scheme applies to the distribution and subject items in the table. In this example, aeffect is the random effect for the first level and is specified using the subject variable A, which follows a normal distribution. Then beffect is the random effect for the second level, which is nested within the first level; it is specified using the subject variable B, which also follows a normal distribution.

The "Dimensions" table (Output 89.6.2) follows the same nested flow to exhibit the model dimensions. It indicates that 1,000 observations are used. There are 100 first-level subjects and 500 second-level subjects, because each first-level subject is contained by five nested second-level subjects. For this model, PROC NLMIXED selects one quadrature point to fit the model.

Output 89.6.2 Dimensions Table for a Two-Level Nested Model

Dimensions	
Observations Used	1000
Observations Not Used	0
Total Observations	1000
Subjects [A]	100
Max Obs per Subject	10
Subjects [B]	500
Max Obs per Subject	2
Parameters	4
Quadrature Points	1

A total of 15 steps are required to achieve convergence. They are shown in the "Iteration History" table (Output 89.6.3).

Output 89.6.3 Iteration History

Iteration History					
Iteration	Calls	Negative Log Likelihood	Difference	Maximum Gradient	Slope
1	12	2982.0141	2695.598	215.363	-87641.4
2	37	2462.0899	519.9243	518.335	-346.648
3	43	2144.0047	318.0852	60.9954	-2483.38
4	49	2108.7720	35.23265	56.3198	-64.4603
5	61	2098.5627	10.20936	43.0728	-7.79282
6	67	2094.2091	4.353503	34.1536	-20.0831
7	73	2090.2802	3.928938	26.4378	-5.86557
8	80	2089.3443	0.93594	15.0337	-2.41721
9	86	2088.1106	1.233621	20.3763	-0.59126
10	98	2072.7699	15.34077	5.02632	-1.53249
11	105	2068.5425	4.227401	28.6085	-2.25953
12	112	2066.4439	2.098605	9.01350	-2.53322
13	119	2065.8127	0.631184	3.73626	-0.50717
14	126	2065.7590	0.053645	1.02923	-0.07232
15	133	2065.7550	0.00406	0.36112	-0.00578
16	139	2065.7502	0.004798	0.27899	-0.00171

NOTE: GCONV convergence criterion satisfied.

The "Parameter Estimates" table (Output 89.6.4) contains the maximum likelihood estimates of the parameters. You can see from this table that the intercept and the s2 variables are very close to the simulation parameters. Also, both variances are very close to the simulation parameters.

Output 89.6.4 Parameter Estimates for a Two-Level Nested Model

Parameter Estimates								
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
vara	9.2198	1.4299	498	6.45	<.0001	6.4104	12.0291	0.022933
varb_a	3.6997	0.2973	498	12.44	<.0001	3.1156	4.2839	0.011981
intercept	10.1546	0.3171	498	32.02	<.0001	9.5315	10.7776	-0.03401
s2	0.9661	0.06102	498	15.83	<.0001	0.8462	1.0860	-0.27897

Example 89.7: Overdispersion Hierarchical Nonlinear Mixed Model

(View the complete code for this example (nlmex7.sas) in the example repository.)

This example describes the overdispersion hierarchical nonlinear mixed model that is given in Ghebretinsae et al. (2013). In this experiment, 24 rats are divided into four groups and receive a daily oral dose of 1,2-dimethylhydrazine dihydrochloride in three dose levels (low, medium, and high) and a vehicle control. In addition, an extra group of three animals receive a positive control. All 27 animals are euthanized 3 hours after the last dose is administered. For each animal, a cell suspension is prepared, and from each cell suspension, three replicate samples are prepared for scoring. Using a semi-automated scoring system, 50 randomly selected cells from each sample are scored per animal for DNA damage. The comet assay methodology is used to detect this DNA damage. In this methodology, the DNA damage is measured by the distance of DNA migration from the perimeter of the comet head to the last visible point in the tail. This distance is defined as the tail length and is observed for the 150 cells from each animal. The data are available in the Sashelp library. In order to use the dose levels as regressors in PROC NLMIXED, you need to create indicator variables for each dose level. The following DATA step creates indicator variables for the dose levels:

```
data comet;
   set sashelp.comet;
   L = 0; M = 0; H = 0; PC = 0;
   if (dose = 1.25) then L  = 1;
   if (dose = 2.50) then M  = 1;
   if (dose = 5.00) then H  = 1;
   if (dose = 200)  then PC = 1;
run;
```

Ghebretinsae et al. (2013) suggest a Weibull distribution to model the outcomes (tail lengths). Note that the data exhibit the nested nature because of the three replicate samples from each cell. Also, the 50 cells from each rat are correlated. To account for the dependency of these 150 correlated cells from each rat, two levels of normally distributed nested random effects are used. Let y_{ijk} be the tail length that is observed in the *k*th cell from the *j*th sample of the *i*th rat. Then the suggested two-level nested model can be described as follows:

$$
\begin{aligned}
y_{ijk}|r_i, s_{j(i)} &\sim \text{Weibull}(\rho, \kappa_{ijk}) \\
\kappa_{ijk} &= \exp(\beta_0 + \beta_1 \mathrm{L}_{ijk} + \beta_2 \mathrm{M}_{ijk} + \beta_3 \mathrm{H}_{ijk} + \beta_4 \mathrm{PC}_i + r_i + s_{j(i)}) \\
r_i &\sim N(0, d_1) \\
s_{j(i)} &\sim N(0, d_2)
\end{aligned}
$$

This model is referred to as the nested Weibull model in the rest of the example. The parameters of the Weibull distribution, ρ and κ_{ijk}, are known as the shape and scale parameters, respectively. Based on this model specification, the conditional expectation of the response variable, given the random effects, is a nonlinear function of fixed and random effects. The nonlinear relationship can be written as

$$
E(y_{ijk}|r_i, s_{j(i)}) = \frac{\Gamma\left(\frac{1}{\rho}+1\right)}{\kappa_{ijk}^{1/\rho}} = \frac{\Gamma\left(\frac{1}{\rho}+1\right)}{\left(e^{(\beta_0+\beta_1 \mathrm{L}_{ijk}+\beta_2 \mathrm{M}_{ijk}+\beta_3 \mathrm{H}_{ijk}+\beta_4 \mathrm{PC}_i+r_i+s_{j(i)})}\right)^{1/\rho}}
$$

These expected values can be obtained using the PREDICT statement in PROC NLMIXED, and they are useful for validating the model fitting. The specified nested Weibull model can be fitted using the following PROC NLMIXED statements:

```
proc nlmixed data = comet;
   parms b0 = -25  b1 = -10 b2 = -10 b3 = -10 b4 = -10 rho = 10;
   bounds sd1 >=0, sd2 >=0;
   mu = b0+ b1*L+ b2*M+ b3*H+ b4*PC + rateff + sampeff;
   term = (length**rho)*exp(mu);
   llike = log(rho) + (rho-1)*log(length) + mu - term;
   model length ~ general(llike);
   random ratEff ~ normal(0,sd1) subject = rat;
   random sampEff ~ normal(0,sd2) subject = sample(rat);
   predict gamma(1+(1/rho))/(exp(mu)**(1/rho)) out = p1;
run;
```

Note that the two-parameter Weibull density function is given by

$$f(y;\rho,\kappa) = \kappa\rho y^{\rho-1}e^{-\kappa y^{\rho}}$$

This implies that the log likelihood is

$$l(\rho,\kappa;y) = \log(\kappa) + \log(\rho) + (\rho-1)\log(y) - \kappa y^{\rho}$$

A general log-likelihood specification is used in the MODEL statement to specify the preceding log of the Weibull density. The linear combination of the parameters, b_0 to b_4, and the two random effects, RatEff and SampEff, is defined using the linear predictor MU in the first SAS programming statement. Then, the next two SAS programming statements together define the log of the Weibull density function. The first RANDOM statement defines the first level of the random effect, r_i, by using RatEff. Then, the second RANDOM statement defines the second level of the random effect, $s_{j(i)}$, nested within the first level by using SampEff. Both random-effects distributions are specified as normal with mean 0 and variance sd1 and sd2 for the first and second levels, respectively. The selected output from this model is shown in Output 89.7.1, Output 89.7.2, and Output 89.7.3.

Output 89.7.1 Model Specification for Nested Weibull Model

The NLMIXED Procedure

Specifications	
Data Set	WORK.COMET
Dependent Variable	Length
Distribution for Dependent Variable	General
Random Effects	rateff, sampeff
Distribution for Random Effects	Normal, Normal
Subject Variables	Rat, Sample
Optimization Technique	Dual Quasi-Newton
Integration Method	Adaptive Gaussian Quadrature

The "Specifications" table shows that RatEff is the first-level random effect, specified by the subject variable Rat, and it follows a normal distribution (Output 89.7.1). Similarly, SampEff is the second-level random effect, nested in RatEff and specified by the subject variable Sample, and it follows a normal distribution.

Output 89.7.2 Dimensions Table for Nested Weibull Model

Dimensions	
Observations Used	4050
Observations Not Used	0
Total Observations	4050
Subjects [Rat]	27
Max Obs per Subject	150
Subjects [Sample]	81
Max Obs per Subject	50
Parameters	8
Quadrature Points	1

The "Dimensions" table indicates that 4,050 observations are used in the analysis (Output 89.7.2). It also indicates that there are 27 rats and that 81 samples are nested within the rats. As explained earlier, three samples are selected randomly from each rat, so a total of 81 (27 times 3) samples are used in the analysis.

Output 89.7.3 Parameter Estimates for Nested Weibull Model

Parameter Estimates								
Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
b0	-15.6348	0.2661	79	-58.76	<.0001	-16.1644	-15.1052	0.12535
b1	-4.4971	0.2402	79	-18.72	<.0001	-4.9753	-4.0190	0.070248
b2	-4.6009	0.2236	79	-20.58	<.0001	-5.0459	-4.1560	-0.37771
b3	-4.8276	0.2240	79	-21.55	<.0001	-5.2734	-4.3818	-0.20049
b4	-3.4972	0.2712	79	-12.89	<.0001	-4.0371	-2.9574	-0.18402
rho	4.9580	0.05873	79	84.42	<.0001	4.8411	5.0749	0.64013
sd1	0.02502	0.04830	79	0.52	0.6059	-0.07112	0.1212	0.29146
sd2	0.3908	0.07321	79	5.34	<.0001	0.2451	0.5365	0.88853

The "Parameter Estimates" table list the maximum likelihood estimates of the parameters along with their standard errors (Output 89.7.3). The estimates are fairly close to the estimates that are given in Ghebretinsae et al. (2013), who use a gamma distribution for the Sample random effects instead of the normal distribution as in the preceding model. Note that the parameters β_1, β_2, and β_3 represent the effects of low, medium, and high doses, respectively, when compared to the vehicle control. Similarly, β_4 is the parameter that corresponds to the effect of positive control when compared to vehicle control. The p-value indicates that the point estimates of all these parameters, β_1 to β_4, are significant; this implies that the toxicity of 1,2-dimethylhydrazine dihydrochloride is significant at different dose levels.

In the nested Weibull model, the scale parameter, κ_{ijk}, is random because it is a function of random effects. This implies that the model assumes that each observation is drawn from a different Weibull distribution. Molenberghs et al. (2010) indicate that this way of modeling leads to an overdispersed Weibull model. Ghebretinsae et al. (2013) use the random-effects mechanism to account for this overdispersion. The

overdispersed Weibull model can be written as follows:

$$
\begin{aligned}
y_{ijk}|r_i, s_{j(i)}, \theta_{ijk} &\sim \text{Weibull}(\rho, \kappa_{ijk}) \\
\kappa_{ijk} &= \theta_{ijk} \exp(\beta_0 + \beta_1 \mathrm{L}_{ijk} + \beta_2 \mathrm{M}_{ijk} + \beta_3 \mathrm{H}_{ijk} + \beta_4 \mathrm{PC}_i + r_i + s_{j(i)}) \\
\theta_{ijk} &\sim \text{gamma}(\alpha, 1/\alpha) \\
r_i &\sim N(0, d_1) \\
s_{j(i)} &\sim N(0, d_2)
\end{aligned}
$$

This model is referred to as the nested Weibull overdispersion model in the rest of this example. Again, in this model, the shape parameter, κ_{ijk}, is the function of the normally distributed random effects, r_i and $s_{j(i)}$, along with other random effects, θ_{ijk}. Further, the model assumes that θ_{ijk} follow a gamma distribution. Note that the random effects, r_i and $s_{j(i)}$, account for the cluster effect of the observations from the same sample that is nested within a rat, whereas θ_{ijk} account for the overdispersion in the data. Molenberghs et al. (2010) integrate out the overdispersion random effects, θ_{ijk}, from the joint distribution $f(y_{ijk}|\theta_{ijk}, r_i, s_{j(i)})g(\theta_{ijk}|\alpha)$ and provide the conditional density of $y_{ijk}|r_i, s_{j(i)}$. The conditional density follows:

$$
f(y_{ijk}|r_i, s_{j(i)}) = \frac{\rho y_{ij}^{\rho-1} e^{(\beta_0+\beta_1 \mathrm{L}_{ijk}+\beta_2 \mathrm{M}_{ijk}+\beta_3 \mathrm{H}_{ijk}+\beta_4 \mathrm{PC}_i+r_i+s_{j(i)})}}{\left(1 + \frac{1}{\alpha} y_{ij}^{\rho} e^{(\beta_0+\beta_1 \mathrm{L}_{ijk}+\beta_2 \mathrm{M}_{ijk}+\beta_3 \mathrm{H}_{ijk}+\beta_4 \mathrm{PC}_i+r_i+s_{j(i)})}\right)^{\alpha+1}}
$$

Again, in this case, the conditional expectation of the response variable, given the random effects, is a nonlinear function of the fixed and random effects. Molenberghs et al. (2010) provide this nonlinear relationship as follows:

$$
E(y_{ijk}|r_i, s_{j(i)}) = \frac{\Gamma\left(\alpha - \frac{1}{\rho}\right)\Gamma\left(\frac{1}{\rho} + 1\right)}{\left(\frac{1}{\alpha} e^{(\beta_0+\beta_1 \mathrm{L}_{ijk}+\beta_2 \mathrm{M}_{ijk}+\beta_3 \mathrm{H}_{ijk}+\beta_4 \mathrm{PC}_i+r_i+s_{j(i)})}\right)^{1/\rho} \Gamma(\alpha)}
$$

As indicated previously, these values are useful for validating the fit of the model, and they can be obtained using a PREDICT statement in PROC NLMIXED. Using the preceding conditional distribution of response, given the random effects and the random-effects distribution, PROC NLMIXED fits the two-level nested Weibull overdispersion model by using the following statements:

```
ods select ParameterEstimates;
proc nlmixed data = comet;
   parms b0 = -25  b1 = -10 b2 = -10 b3 = -10 b4 = -10 alpha = 1 rho = 10;
   bounds sd1 >=0, sd2 >=0;
   mu = b0+ b1*L+ b2*M+ b3*H+ b4*PC + rateff + sampeff;
   num = rho*((length)**(rho-1))*exp(mu);
   den = (1+(((length)**rho)*exp(mu))/alpha)**(alpha+1);
   llike = log(num/den);
   model length ~ general(llike);
   random rateff ~ normal(0,sd1) subject = rat;
   random sampeff ~ normal(0,sd2) subject = sample(rat);
   predict (gamma(alpha-(1/rho))*gamma(1+(1/rho)))
           /(gamma(alpha)*((exp(mu)/alpha)**(1/rho))) out = p2;
run;
```

The parameter estimates that are obtained from the nested Weibull overdispersion model are given in Output 89.7.1. The "Parameter Estimates" table indicates that the point estimates are still significant, as they are in the previous model in which the overdispersion is not taken into consideration. So the same conclusion about the dose levels holds: the toxicity of 1,2-dimethylhydrazine dihydrochloride is significant at different dose levels.

Output 89.7.4 Parameter Estimates for Nested Weibull Overdispersion Model

The NLMIXED Procedure

Parameter Estimates

Parameter	Estimate	Standard Error	DF	t Value	Pr > |t|	95% Confidence Limits		Gradient
b0	-31.0039	0.8151	79	-38.04	<.0001	-32.6263	-29.3815	-0.16634
b1	-11.9098	0.5158	79	-23.09	<.0001	-12.9365	-10.8831	0.002898
b2	-12.1190	0.5172	79	-23.43	<.0001	-13.1485	-11.0895	0.10644
b3	-12.5512	0.5216	79	-24.06	<.0001	-13.5893	-11.5130	0.30625
b4	-9.6301	0.5554	79	-17.34	<.0001	-10.7355	-8.5246	-0.11915
alpha	0.8925	0.05046	79	17.69	<.0001	0.7921	0.9930	2.03622
rho	10.7187	0.2790	79	38.41	<.0001	10.1633	11.2740	0.54432
sd1	0.1908	0.1548	79	1.23	0.2214	-0.1173	0.4988	0.52252
sd2	0.7925	0.1651	79	4.80	<.0001	0.4638	1.1212	-0.01877

Recall that, from both models, the conditional expected values of the response variable, given the random effects, are obtained using the PREDICT statement. These predicted values are stored in the data sets p1 and p2 for the nested Weibull and nested Weibull overdispersion models, respectively. Because, in this example, the regressor variables are only indicators, the prediction values for all the observations from the same sample that is nested within a rat are equal. Further, compute the average predicted value for each sample that is nested within a rat for both models. These values predict the average tail length of each sample that is nested within a rat based on the fitted model. In addition, compute the observed sample averages of the responses from each cluster, which can be viewed as the crude estimate of the average population tail length for each cluster. The following statements combine all these average tail lengths, both observed and predicted, from the two fitted models into a single data set:

```
proc means data = p1 mean noprint;
   class sample;
   var pred;
   output out = p11 (keep = sample weibull_mean)mean = weibull_mean;
run;

proc means data = p2 mean noprint;
   class sample;
   var pred;
   output out = p21 (keep = sample weibull_gamma_mean)mean = weibull_gamma_mean;
run;

proc means data = comet mean noprint;
   class sample;
   var length;
   id dose rat;
   output out = p31(keep = dose rat sample observed_mean) mean = observed_mean;
```

```
run;

data average;
   merge p11 p21 p31;
   by sample;
   if sample = . then delete;
   label observed_mean = "Observed Average";
   label weibull_gamma_mean = "Weibull Gamma Average";
   label weibull_mean = "Weibull Average";
   label sample = "Sample Index";
   if dose = 0 then dosage = "0 Vehicle Control ";
   if dose = 1.25 then dosage = "1 Low";
   if dose = 2.50 then dosage = "2 Medium";
   if dose = 5.00 then dosage = "3 High";
   if dose = 200  then dosage = "4 Positive Control";
run;
```

In order to validate the fit of the models, the observed average tail length values are plotted along with predicted average tail lengths from each model that is fitted using PROC NLMIXED. Moreover, the plots are created for each dosage level by using a PANELBY statement in PROC SGPANEL. The code follows:

```
proc sgpanel data = average;
   panelby dosage/onepanel layout = columnlattice novarname uniscale = row;
   rowaxis values=(10 to 80 by 5) label="Average Tail Length";
   series x = sample y = observed_mean;
   series x = sample y = weibull_mean;
   series x = sample y = weibull_gamma_mean;
run;
```

The resulting plots are given in Output 89.7.5.

Output 89.7.5 Comparison of Nested Weibull Model and Nested Weibull Overdispersion Model

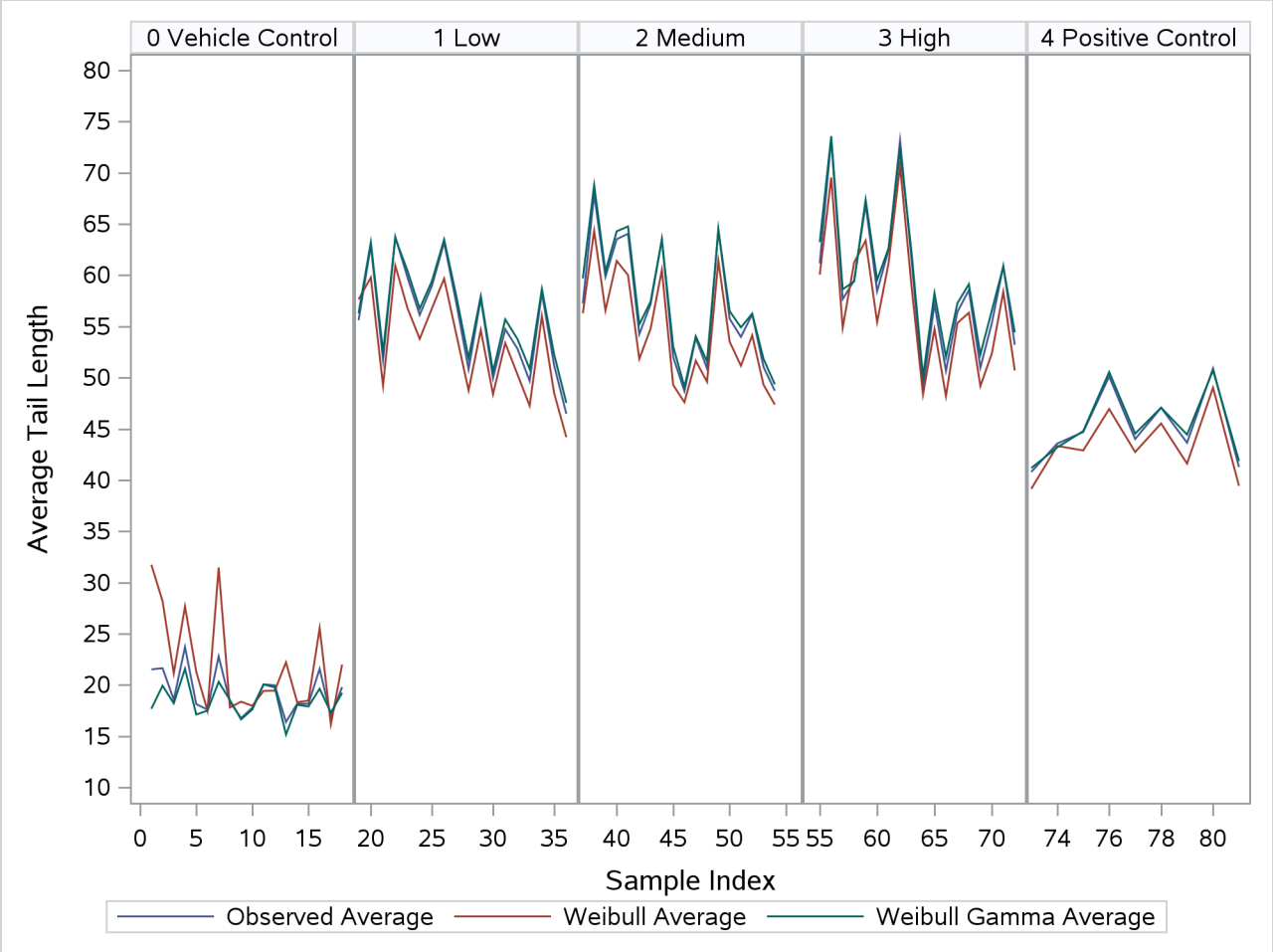

You can see from the plots that the observed averages of the tail lengths from each sample that is nested within a rat are closer to the predicted averages of the nested Weibull overdispersion model than to the predicted averages of the nested Weibull model.

References

Abramowitz, M., and Stegun, I. A., eds. (1972). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables.* 10th printing. New York: Dover.

Abuhelwa, A. Y., Foster, D. J., and Upton, R. N. (2015). "ADVAN-Style Analytical Solutions for Common Pharmacokinetic Models." *Journal of Pharmacological and Toxicological Methods* 73:42–48.

Anderson, D. A., and Aitkin, M. (1985). "Variance Component Models with Binary Response: Interviewer Variability." *Journal of the Royal Statistical Society, Series B* 47:203–210.

Beal, S. L., and Sheiner, L. B. (1982). "Estimating Population Kinetics." *CRC Critical Reviews in Biomedical Engineering* 8:195–222.

Beal, S. L., and Sheiner, L. B. (1988). "Heteroscedastic Nonlinear Regression." *Technometrics* 30:327–338.

Beal, S. L., and Sheiner, L. B. (1992). *NONMEM User's Guide*. San Francisco: NONMEM Project Group, University of California, San Francisco.

Beal, S. L., Sheiner, L. B., Boeckmann, A. J., and Bauer, R. J., eds. (2011). *NONMEM User's Guides (1989–2011)*. Hanover, MD: Icon Development Solutions.

Beale, E. M. L. (1972). "A Derivation of Conjugate Gradients." In *Numerical Methods for Nonlinear Optimization*, edited by F. A. Lootsma, 39–43. London: Academic Press.

Beitler, P. J., and Landis, J. R. (1985). "A Mixed-Effects Model for Categorical Data." *Biometrics* 41:991–1000.

Billingsley, P. (1986). *Probability and Measure*. 2nd ed. New York: John Wiley & Sons.

Booth, J. G., and Hobert, J. P. (1998). "Standard Errors of Prediction in Generalized Linear Mixed Models." *Journal of the American Statistical Association* 93:262–272.

Breslow, N. E., and Clayton, D. G. (1993). "Approximate Inference in Generalized Linear Mixed Models." *Journal of the American Statistical Association* 88:9–25.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Cox, C. (1998). "Delta Method." In *Encyclopedia of Biostatistics*, edited by P. Armitage and T. Colton. New York: John Wiley & Sons.

Cox, D. R., and Oakes, D. (1984). *Analysis of Survival Data*. London: Chapman & Hall.

Cramer, J. S. (1986). *Econometric Applications of Maximum Likelihood Methods*. Cambridge: Cambridge University Press.

Crouch, E. A. C., and Spiegelman, D. (1990). "The Evaluation of Integrals of the Form $\int_{-\infty}^{\infty} f(t)\exp(-t^2)dt$: Application to Logistic-Normal Models." *Journal of the American Statistical Association* 85:464–469.

Davidian, M., and Gallant, A. R. (1993). "The Nonlinear Mixed Effects Model with a Smooth Random Effects Density." *Biometrika* 80:475–488.

Davidian, M., and Giltinan, D. M. (1995). *Nonlinear Models for Repeated Measurement Data*. New York: Chapman & Hall.

Dennis, J. E., Gay, D. M., and Welsch, R. E. (1981). "An Adaptive Nonlinear Least-Squares Algorithm." *ACM Transactions on Mathematical Software* 7:348–368.

Dennis, J. E., and Mei, H. H. W. (1979). "Two New Unconstrained Optimization Algorithms Which Use Function and Gradient Values." *Journal of Optimization Theory and Applications* 28:453–482.

Dennis, J. E., and Schnabel, R. B. (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Englewood Cliffs, NJ: Prentice-Hall.

Diggle, P. J., Liang, K.-Y., and Zeger, S. L. (1994). *Analysis of Longitudinal Data*. Oxford: Clarendon Press.

Draper, D. (1996). "Discussion of the Paper by Lee and Nelder." *Journal of the Royal Statistical Society, Series B* 58:662–663.

Draper, N. R., and Smith, H. (1981). *Applied Regression Analysis*. 2nd ed. New York: John Wiley & Sons.

Engel, B., and Keen, A. (1992). *A Simple Approach for the Analysis of Generalized Linear Mixed Models*, vol. LWA-92-6. Wageningen, Netherlands: Agricultural Mathematics Group (GLW-DLO).

Eskow, E., and Schnabel, R. B. (1991). "Algorithm 695: Software for a New Modified Cholesky Factorization." *ACM Transactions on Mathematical Software* 17:306–312.

Ezzet, F., and Whitehead, J. (1991). "A Random Effects Model for Ordinal Responses from a Crossover Trial." *Statistics in Medicine* 10:901–907.

Fisher, D., and Shafer, S. (2007). "Pharmacokinetic and Pharmacodynamic Analysis with NONMEM: Basic Concepts." Course materials for Fisher/Shafer NONMEM Workshop, March 7–11, Ghent, Belgium.

Fletcher, R. (1987). *Practical Methods of Optimization*. 2nd ed. Chichester, UK: John Wiley & Sons.

Gabrielsson, J., and Weiner, D. (2006). *Pharmacokinetic and Pharmacodynamic Data Analysis: Concepts and Applications*. 4th ed. Stockholm: Swedish Pharmaceutical Press.

Galecki, A. T. (1998). "NLMEM: New SAS/IML Macro for Hierarchical Nonlinear Models." *Computer Methods and Programs in Biomedicine* 55:207–216.

Gallant, A. R. (1987). *Nonlinear Statistical Models*. New York: John Wiley & Sons.

Gaver, D. P., and O'Muircheartaigh, I. G. (1987). "Robust Empirical Bayes Analysis of Event Rates." *Technometrics* 29:1–15.

Gay, D. M. (1983). "Subroutines for Unconstrained Minimization." *ACM Transactions on Mathematical Software* 9:503–524.

Ghebretinsae, A. H., Faes, C., Molenberghs, G., De Boeck, M., and Geys, H. (2013). "A Bayesian, Generalized Frailty Model for Comet Assays." *Journal of Biopharmaceutical Statistics* 23:618–636.

Gilmour, A. R., Anderson, R. D., and Rae, A. L. (1985). "The Analysis of Binomial Data by Generalized Linear Mixed Model." *Biometrika* 72:593–599.

Goldstein, H. (1991). "Nonlinear Multilevel Models, with an Application to Discrete Response Data." *Biometrika* 78:45–51.

Golub, G. H., and Welsch, J. H. (1969). "Calculation of Gaussian Quadrature Rules." *Mathematical Computing* 23:221–230.

Harville, D. A., and Mee, R. W. (1984). "A Mixed-Model Procedure for Analyzing Ordered Categorical Data." *Biometrics* 40:393–408.

Hedeker, D., and Gibbons, R. D. (1994). "A Random Effects Ordinal Regression Model for Multilevel Analysis." *Biometrics* 50:933–944.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Kurada, R. R., and Chen, F. (2018). "Fitting Compartment Models Using PROC NLMIXED." In *Proceedings of the SAS Global Forum 2018 Conference*. Cary, NC: SAS Institute Inc. https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2018/1883-2018.pdf.

Lin, X., and Breslow, N. E. (1996). "Bias Correction in Generalized Linear Mixed Models with Multiple Components of Dispersion." *Journal of the American Statistical Association* 91:1007–1016.

Lindstrom, M. J., and Bates, D. M. (1990). "Nonlinear Mixed Effects Models for Repeated Measures Data." *Biometrics* 46:673–687.

Littell, R. C., Milliken, G. A., Stroup, W. W., Wolfinger, R. D., and Schabenberger, O. (2006). *SAS for Mixed Models*. 2nd ed. Cary, NC: SAS Institute Inc.

Liu, Q., and Pierce, D. A. (1994). "A Note on Gauss-Hermite Quadrature." *Biometrika* 81:624–629.

Longford, N. T. (1994). "Logistic Regression with Random Coefficients." *Computational Statistics and Data Analysis* 17:1–15.

McCulloch, C. E. (1994). "Maximum Likelihood Variance Components Estimation for Binary Data." *Journal of the American Statistical Association* 89:330–335.

McGilchrist, C. A. (1994). "Estimation in Generalized Mixed Models." *Journal of the Royal Statistical Society, Series B* 56:61–69.

Molenberghs, G., Verbeke, G., Demetrio, C. G. B., and Vieira, A. M. C. (2010). "A Family of Generalized Linear Models for Repeated Measures with Normal and Conjugate Random Effects." *Statistical Science* 25:325–347.

Moré, J. J. (1978). "The Levenberg-Marquardt Algorithm: Implementation and Theory." In *Lecture Notes in Mathematics*, vol. 30, edited by G. A. Watson, 105–116. Berlin: Springer-Verlag.

Moré, J. J., and Sorensen, D. C. (1983). "Computing a Trust-Region Step." *SIAM Journal on Scientific and Statistical Computing* 4:553–572.

Ochi, Y., and Prentice, R. L. (1984). "Likelihood Inference in a Correlated Probit Regression Model." *Biometrika* 71:531–543.

Owen, J. S., and Fiedler-Kelly, J. (2014). *Introduction to Population Pharmacokinetic/Pharmacodynamic Analysis with Nonlinear Mixed Effects Models*. Hoboken, NJ: John Wiley & Sons.

Pierce, D. A., and Sands, B. R. (1975). *Extra-Bernoulli Variation in Binary Data*. Technical Report 46, Department of Statistics, Oregon State University.

Pinheiro, J. C., and Bates, D. M. (1995). "Approximations to the Log-Likelihood Function in the Nonlinear Mixed-Effects Model." *Journal of Computational and Graphical Statistics* 4:12–35.

Polak, E. (1971). *Computational Methods in Optimization*. New York: Academic Press.

Powell, M. J. D. (1977). "Restart Procedures for the Conjugate Gradient Method." *Mathematical Programming* 12:241–254.

Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.

Rodriguez, G., and Goldman, N. (1995). "An Assessment of Estimation Procedures for Multilevel Models with Binary Response." *Journal of the Royal Statistical Society, Series A* 158:73–89.

Roe, D. J. (1997). "Comparison of Population Pharmacokinetic Modeling Methods Using Simulated Data: Results from the Population Modeling Workgroup." *Statistics in Medicine* 16:1241–1262.

Schall, R. (1991). "Estimation in Generalized Linear Models with Random Effects." *Biometrika* 78:719–727.

Schittkowski, K., and Stoer, J. (1979). "A Factorization Method for the Solution of Constrained Linear Least Squares Problems Allowing Subsequent Data Changes." *Numerische Mathematik* 31:431–463.

Self, S. G., and Liang, K.-Y. (1987). "Asymptotic Properties of Maximum Likelihood Estimators and Likelihood Ratio Tests under Nonstandard Conditions." *Journal of the American Statistical Association* 82:605–610.

Serfling, R. J. (1980). *Approximation Theorems of Mathematical Statistics*. New York: John Wiley & Sons.

Sheiner, L. B., and Beal, S. L. (1980). "Evaluation of Methods for Estimating Population Pharmacokinetic Parameters, Part I: Michaelis-Menten Model—Routine Clinical Pharmacokinetic Data." *Journal of Pharmacokinetics and Biopharmaceutics* 8:553–571.

Sheiner, L. B., and Beal, S. L. (1985). "Pharmacokinetic Parameter Estimates from Several Least Squares Procedures: Superiority of Extended Least Squares." *Journal of Pharmacokinetics and Biopharmaceutics* 13:185–201.

Smith, S. P. (1995). "Differentiation of the Cholesky Algorithm." *Journal of Computational and Graphical Statistics* 4:134–147.

Stiratelli, R., Laird, N. M., and Ware, J. H. (1984). "Random Effects Models for Serial Observations with Binary Response." *Biometrics* 40:961–971.

Vonesh, E. F. (1992). "Nonlinear Models for the Analysis of Longitudinal Data." *Statistics in Medicine* 11:1929–1954.

Vonesh, E. F. (1996). "A Note on Laplace's Approximation for Nonlinear Mixed-Effects Models." *Biometrika* 83:447–452.

Vonesh, E. F., and Chinchilli, V. M. (1997). *Linear and Nonlinear Models for the Analysis of Repeated Measurements*. New York: Marcel Dekker.

Weil, C. S. (1970). "Selection of the Valid Number of Sampling Units and Consideration of Their Combination in Toxicological Studies Involving Reproduction, Teratogenesis, or Carcinogenesis." *Food and Cosmetic Toxicology* 8:177–182.

White, H. (1982). "Maximum Likelihood Estimation of Misspecified Models." *Econometrica* 50:1–25.

Williams, D. A. (1975). "The Analysis of Binary Responses from Toxicological Experiments Involving Reproduction and Teratogenicity." *Biometrics* 31:949–952.

Wolfinger, R. D. (1993). "Laplace's Approximation for Nonlinear Mixed Models." *Biometrika* 80:791–795.

Wolfinger, R. D. (1997). "Comment: Experiences with the SAS Macro NLINMIX." *Statistics in Medicine* 16:1258–1259.

Wolfinger, R. D., and Lin, X. (1997). "Two Taylor-Series Approximation Methods for Nonlinear Mixed Models." *Computational Statistics and Data Analysis* 25:465–490.

Wolfinger, R. D., and O'Connell, M. A. (1993). "Generalized Linear Mixed Models: A Pseudo-likelihood Approach." *Journal of Statistical Computation and Simulation* 48:233–243.

Yuh, L., Beal, S. L., Davidian, M., Harrison, F., Hester, A., Kowalski, K., Vonesh, E., and Wolfinger, R. D. (1994). "Population Pharmacokinetic/Pharmacodynamic Methodology and Applications: A Bibliography." *Biometrics* 50:566–575.

Chapter 90
The NPAR1WAY Procedure

Contents

Overview: NPAR1WAY Procedure . . . **7356**
Getting Started: NPAR1WAY Procedure . . . **7356**
Syntax: NPAR1WAY Procedure . . . **7365**
- PROC NPAR1WAY Statement . . . 7365
- BY Statement . . . 7374
- CLASS Statement . . . 7375
- EXACT Statement . . . 7375
- FREQ Statement . . . 7380
- OUTPUT Statement . . . 7380
- STRATA Statement . . . 7383
- VAR Statement . . . 7385

Details: NPAR1WAY Procedure . . . **7385**
- Missing Values . . . 7385
- Tied Values . . . 7386
- Statistical Computations . . . 7386
  - Simple Linear Rank Tests for Two-Sample Data . . . 7386
  - One-Way ANOVA Tests . . . 7388
  - Scores for Linear Rank and One-Way ANOVA Tests . . . 7389
  - Stratified Analysis . . . 7391
  - Hodges-Lehmann Estimation of Location Shift . . . 7392
  - Fligner-Policello Test . . . 7394
  - Ratio Mean Deviations (RMD) Exact Test . . . 7396
  - Multiple Comparisons Based on Pairwise Rankings . . . 7397
  - Empirical Distribution Function Tests . . . 7397
  - Exact Tests . . . 7399
- Contents of the Output Data Set . . . 7403
- Displayed Output . . . 7409
- ODS Table Names . . . 7417
- ODS Graphics . . . 7419

Examples: NPAR1WAY Procedure . . . **7420**
- Example 90.1: Two-Sample Location Tests and Plots . . . 7420
- Example 90.2: EDF Statistics and EDF Plot . . . 7424
- Example 90.3: Exact Wilcoxon Two-Sample Test . . . 7425
- Example 90.4: Hodges-Lehmann Estimation . . . 7427
- Example 90.5: Exact Savage Multisample Test . . . 7428

References . . . **7429**

Overview: NPAR1WAY Procedure

The NPAR1WAY procedure performs nonparametric tests for location and scale differences across a one-way classification. PROC NPAR1WAY also provides a standard analysis of variance on the raw data, empirical distribution function statistics, pairwise multiple comparison analysis, and stratified analysis.

PROC NPAR1WAY performs tests for location and scale differences based on the following rank-based scores of a response variable: Wilcoxon, median, Van der Waerden (normal), Savage, Siegel-Tukey, Ansari-Bradley, Klotz, Mood, and Conover. In addition, PROC NPAR1WAY provides tests that use the raw input data as scores. When the data are classified into two samples, tests are based on simple linear rank statistics. When the data are classified into more than two samples, tests are based on one-way analysis of variance (ANOVA) statistics. Both asymptotic and exact *p*-values are available for these tests. PROC NPAR1WAY also provides Hodges-Lehmann estimation of the location shift (with exact confidence limits), the Fligner-Policello test, and the RMD exact scale test for two-sample data.

PROC NPAR1WAY provides stratified analysis for two-sample data based on the following scores: Wilcoxon, median, Van der Waerden (normal), Savage, and raw data scores. Rank-based scores can be computed by using within-stratum ranks or overall ranks; strata can be weighted by stratum size or by equal weights. PROC NPAR1WAY also provides alignment by strata.

PROC NPAR1WAY computes empirical distribution function (EDF) statistics, which test whether the distribution of a variable is the same across different groups. These statistics include the Kolmogorov-Smirnov test, the Cramér–von Mises test, and the Kuiper test. Exact *p*-values are available for the two-sample Kolmogorov-Smirnov test.

PROC NPAR1WAY uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For more information about the statistical graphics that PROC NPAR1WAY produces, see the PLOTS= option in the PROC NPAR1WAY statement and the section "ODS Graphics" on page 7419.

Getting Started: NPAR1WAY Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/m-n/npar1gs.sas`.)

This example illustrates how you can use PROC NPAR1WAY to perform a one-way nonparametric analysis. The data from Halverson and Sherwood (1930) consist of weight gain measurements for five different levels of gossypol additive in animal feed. Gossypol is a substance contained in cottonseed shells, and these data were collected to study the effect of gossypol on animal nutrition.

The following DATA step statements create the SAS data set Gossypol:

```
data Gossypol;
   input Dose n;
   do i=1 to n;
      input Gain @@;
      output;
   end;
```

```
   datalines;
0 16
228 229 218 216 224 208 235 229 233 219 224 220 232 200 208 232
.04 11
186 229 220 208 228 198 222 273 216 198 213
.07 12
179 193 183 180 143 204 114 188 178 134 208 196
.10 17
130 87 135 116 118 165 151 59 126 64 78 94 150 160 122 110 178
.13 11
154 130 130 118 118 104 112 134 98 100 104
;
```

The data set Gossypol contains the variable Dose, which represents the amount of gossypol additive, and the variable Gain, which represents the weight gain.

Researchers are interested in whether there is a difference in weight gain among animals receiving the different dose levels of gossypol. The following statements invoke the NPAR1WAY procedure to perform a nonparametric analysis of this problem:

```
proc npar1way data=Gossypol;
   class Dose;
   var Gain;
run;
```

The variable Dose is the CLASS variable, and the VAR statement specifies the variable Gain is the response variable. The CLASS statement is required, and you must name only one CLASS variable. You can name one or more analysis variables in the VAR statement. If you omit the VAR statement, PROC NPAR1WAY analyzes all numeric variables in the data set except the CLASS variable, the FREQ variable, and the BY variables.

When no analysis options are specified in the PROC NPAR1WAY statement, the ANOVA, WILCOXON, MEDIAN, VW, SAVAGE, and EDF options are invoked by default. The tables in the following figures show the results of these analyses.

The tables in Figure 90.1 are produced by the ANOVA option. For each level of the CLASS variable Dose, PROC NPAR1WAY displays the number of observations and the mean of the analysis variable Gain. PROC NPAR1WAY displays a standard analysis of variance on the raw data. This gives the same results as the GLM and ANOVA procedures. The p-value for the F test is <0.0001, which indicates that Dose accounts for a significant portion of the variability of the dependent variable Gain.

Figure 90.1 Analysis of Variance

The NPAR1WAY Procedure

Analysis of Variance for Variable Gain Classified by Variable Dose

Dose	N	Mean
0	16	222.187500
0.04	11	217.363636
0.07	12	175.000000
0.1	17	120.176471
0.13	11	118.363636

Figure 90.1 *continued*

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Among	4	140082.986077	35020.74652	55.8143	<.0001
Within	62	38901.998997	627.45160		

Average scores were used for ties.

The WILCOXON option produces the output in Figure 90.2. PROC NPAR1WAY first provides a summary of the Wilcoxon scores for the analysis variable Gain by class level. For each level of the CLASS variable Dose, PROC NPAR1WAY displays the following information: number of observations, sum of the Wilcoxon scores, expected sum under the null hypothesis of no difference among class levels, standard deviation under the null hypothesis, and mean score.

Next PROC NPAR1WAY displays the one-way ANOVA statistic, which for Wilcoxon scores is known as the Kruskal-Wallis test. The statistic is 52.6656, with 4 degrees of freedom, which is the number of class levels minus 1. The *p*-value (probability of a larger statistic under the null hypothesis) is <0.0001. This leads to rejection of the null hypothesis that there is no difference in location for Gain among the levels of Dose. This *p*-value is asymptotic, computed from the asymptotic chi-square distribution of the test statistic. For certain data sets it might also be useful to compute the exact *p*-value—for example, for small data sets or for data sets that are sparse, skewed, or heavily tied. You can use the EXACT statement to request exact *p*-values for any of the location or scale tests available in PROC NPAR1WAY.

Figure 90.2 Wilcoxon Score Analysis

Wilcoxon Scores (Rank Sums) for Variable Gain Classified by Variable Dose

Dose	N	Sum of Scores	Expected Under H0	Std Dev Under H0	Mean Score
0	16	890.50	544.0	67.978966	55.656250
0.04	11	555.00	374.0	59.063588	50.454545
0.07	12	395.50	408.0	61.136622	32.958333
0.1	17	275.50	578.0	69.380741	16.205882
0.13	11	161.50	374.0	59.063588	14.681818

Average scores were used for ties.

Kruskal-Wallis Test

Chi-Square	DF	Pr > ChiSq
52.6656	4	<.0001

Figure 90.3 through Figure 90.5 display the analyses produced by the MEDIAN, VW, and SAVAGE options. For each score type, PROC NPAR1WAY provides a summary of scores and the one-way ANOVA statistic, as previously described for Wilcoxon scores. Other score types available in PROC NPAR1WAY are Siegel-Tukey, Ansari-Bradley, Klotz, and Mood, which can be used to test for scale differences. Conover scores can be used to test for differences in both location and scale. Additionally, you can specify the SCORES=DATA option, which uses the input data as scores. This option gives you the flexibility to construct any scores for your data with the DATA step and then analyze these scores with PROC NPAR1WAY.

Figure 90.3 Median Score Analysis

Median Scores (Number of Points Above Median) for Variable Gain Classified by Variable Dose					
Dose	**N**	**Sum of Scores**	**Expected Under H0**	**Std Dev Under H0**	**Mean Score**
0	16	16.0	7.880597	1.757902	1.00
0.04	11	11.0	5.417910	1.527355	1.00
0.07	12	6.0	5.910448	1.580963	0.50
0.1	17	0.0	8.373134	1.794152	0.00
0.13	11	0.0	5.417910	1.527355	0.00
Average scores were used for ties.					

Median One-Way Analysis		
Chi-Square	**DF**	**Pr > ChiSq**
54.1765	4	<.0001

Figure 90.4 Van der Waerden (Normal) Score Analysis

Van der Waerden Scores (Normal) for Variable Gain Classified by Variable Dose					
Dose	**N**	**Sum of Scores**	**Expected Under H0**	**Std Dev Under H0**	**Mean Score**
0	16	16.116474	0.0	3.325957	1.007280
0.04	11	8.340899	0.0	2.889761	0.758264
0.07	12	-0.576674	0.0	2.991186	-0.048056
0.1	17	-14.688921	0.0	3.394540	-0.864054
0.13	11	-9.191777	0.0	2.889761	-0.835616
Average scores were used for ties.					

Van der Waerden One-Way Analysis		
Chi-Square	**DF**	**Pr > ChiSq**
47.2972	4	<.0001

Figure 90.5 Savage Score Analysis

Savage Scores (Exponential) for Variable Gain Classified by Variable Dose					
Dose	**N**	**Sum of Scores**	**Expected Under H0**	**Std Dev Under H0**	**Mean Score**
0	16	16.074391	0.0	3.385275	1.004649
0.04	11	7.693099	0.0	2.941300	0.699373
0.07	12	-3.584958	0.0	3.044534	-0.298746
0.1	17	-11.979488	0.0	3.455082	-0.704676
0.13	11	-8.203044	0.0	2.941300	-0.745731
Average scores were used for ties.					

Savage One-Way Analysis		
Chi-Square	**DF**	**Pr > ChiSq**
39.4908	4	<.0001

The tables in Figure 90.6 display the empirical distribution function statistics, comparing the distribution of Gain for the different levels of Dose. These tables are produced by the EDF option, and they include Kolmogorov-Smirnov statistics and Cramér–von Mises statistics.

Figure 90.6 Empirical Distribution Function Analysis

Kolmogorov-Smirnov Test for Variable Gain Classified by Variable Dose

Dose	N	EDF at Maximum	Deviation from Mean at Maximum
0	16	0.000000	-1.910448
0.04	11	0.000000	-1.584060
0.07	12	0.333333	-0.499796
0.1	17	1.000000	2.153861
0.13	11	1.000000	1.732565
Total	67	0.477612	

Maximum Deviation Occurred at Observation 36

Value of Gain at Maximum = 178.0

Kolmogorov-Smirnov Statistics (Asymptotic)

KS	0.457928	**KSa**	3.748300

Cramer-von Mises Test for Variable Gain Classified by Variable Dose

Dose	N	Summed Deviation from Mean
0	16	2.165210
0.04	11	0.918280
0.07	12	0.348227
0.1	17	1.497542
0.13	11	1.335745

Cramer-von Mises Statistics (Asymptotic)

CM	0.093508	**CMa**	6.265003

PROC NPAR1WAY uses ODS Graphics to create graphs as part of its output. The following statements produce a box plot of Wilcoxon scores for Gain classified by Dose. ODS Graphics must be enabled before producing graphs.

```
ods graphics on;
proc npar1way data=Gossypol plots(only)=wilcoxonboxplot;
   class Dose;
   var Gain;
run;
ods graphics off;
```

Figure 90.7 displays the box plot of Wilcoxon scores. This graph corresponds to the Wilcoxon scores analysis shown in Figure 90.2. To remove the *p*-value from the box plot display, you can specify the NOSTATS plot option in parentheses after the WILCOXONBOXPLOT option.

Box plots are available for all PROC NPAR1WAY score types except median scores, which are displayed

in a stacked bar chart. If ODS Graphics is enabled but you do not specify the PLOTS= option, PROC NPAR1WAY produces all plots that are associated with the analyses that you request.

Figure 90.7 Box Plot of Wilcoxon Scores

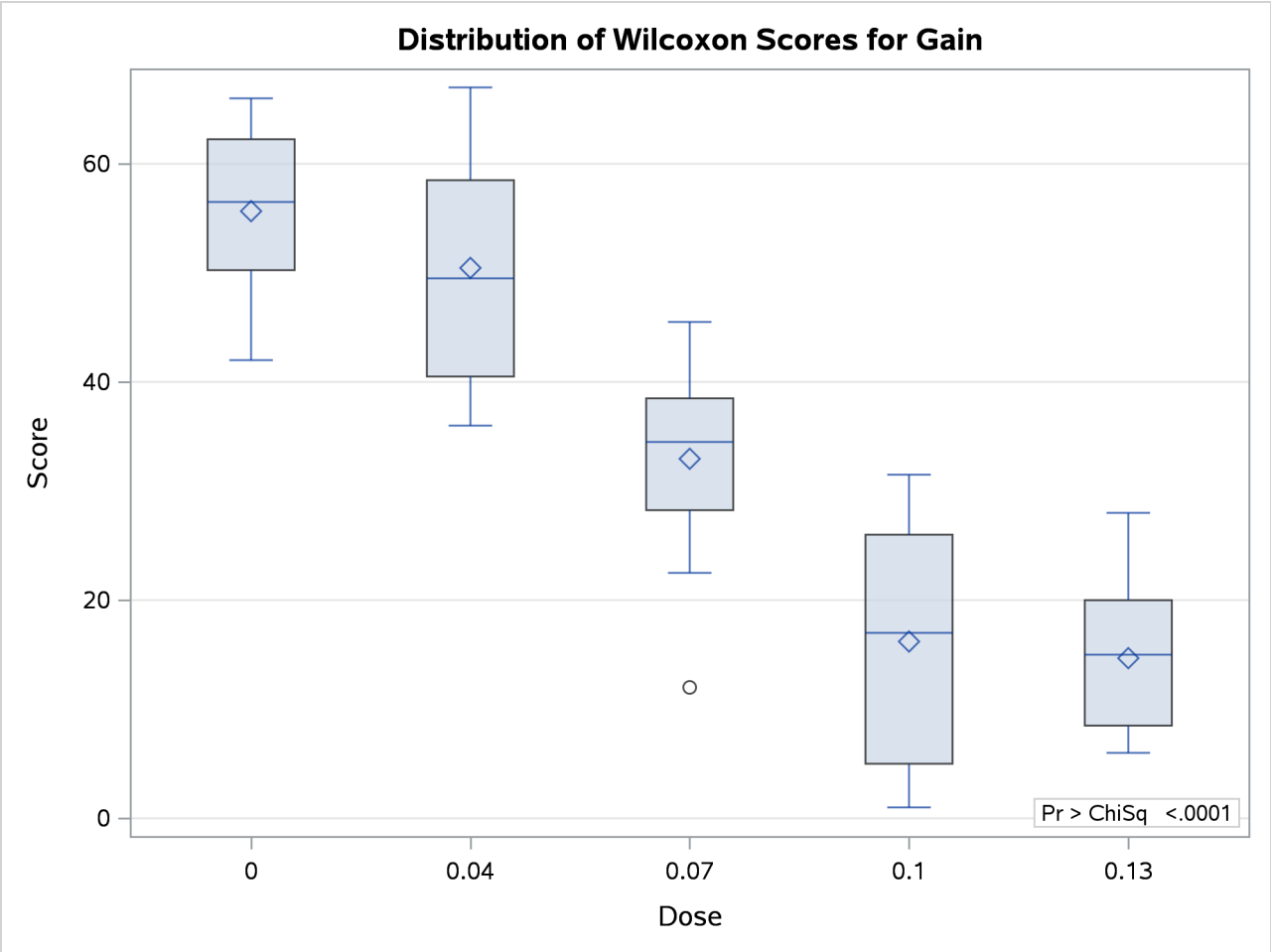

In the preceding example, the CLASS variable Dose has five levels, and the analyses examine possible differences among these five levels (samples). The following statements invoke the NPAR1WAY procedure to perform a nonparametric analysis of the two lowest levels of Dose:

```
proc npar1way data=Gossypol;
   where Dose <= .04;
   class Dose;
   var Gain;
run;
```

The tables in the following figures show the results of this two-sample analysis. The tables in Figure 90.8 are produced by the ANOVA option.

Figure 90.8 Analysis of Variance for Two-Sample Data

The NPAR1WAY Procedure

Analysis of Variance for Variable Gain Classified by Variable Dose		
Dose	**N**	**Mean**
0	16	222.187500
0.04	11	217.363636

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Among	1	151.683712	151.683712	0.5587	0.4617
Within	25	6786.982955	271.479318		

Average scores were used for ties.

Figure 90.9 displays the output produced by the WILCOXON option. PROC NPAR1WAY provides a summary of the Wilcoxon scores for the analysis variable Gain for each of the two class levels. Because there are two levels, PROC NPAR1WAY displays the two-sample test, which is based on the simple linear rank statistic with Wilcoxon scores. The normal approximation includes a continuity correction. To remove the continuity correction, you can specify the CORRECT=NO option. PROC NPAR1WAY also gives a t approximation for the Wilcoxon two-sample test. Like the multisample analysis, PROC NPAR1WAY computes a one-way ANOVA statistic, which for Wilcoxon scores is known as the Kruskal-Wallis test. All these p-values show no difference in Gain for the two Dose levels at the 0.05 level of significance.

Figure 90.10 through Figure 90.12 display the two-sample analyses produced by the MEDIAN, VW, and SAVAGE options.

Figure 90.9 Wilcoxon Two-Sample Analysis

Wilcoxon Scores (Rank Sums) for Variable Gain Classified by Variable Dose					
Dose	**N**	**Sum of Scores**	**Expected Under H0**	**Std Dev Under H0**	**Mean Score**
0	16	253.50	224.0	20.221565	15.843750
0.04	11	124.50	154.0	20.221565	11.318182

Average scores were used for ties.

Wilcoxon Two-Sample Test					
				t Approximation	
Statistic	**Z**	**Pr < Z**	**Pr > |Z|**	**Pr < Z**	**Pr > |Z|**
124.5000	-1.4341	0.0758	0.1515	0.0817	0.1635

Z includes a continuity correction of 0.5.

Kruskal-Wallis Test		
Chi-Square	**DF**	**Pr > ChiSq**
2.1282	1	0.1446

Figure 90.10 Median Two-Sample Analysis

Median Scores (Number of Points Above Median) for Variable Gain Classified by Variable Dose					
Dose	**N**	**Sum of Scores**	**Expected Under H0**	**Std Dev Under H0**	**Mean Score**
0	16	9.0	7.703704	1.299995	0.562500
0.04	11	4.0	5.296296	1.299995	0.363636
Average scores were used for ties.					

Median Two-Sample Test			
Statistic	**Z**	**Pr < Z**	**Pr > |Z|**
4.0000	-0.9972	0.1593	0.3187

Median One-Way Analysis		
Chi-Square	**DF**	**Pr > ChiSq**
0.9943	1	0.3187

Figure 90.11 Van der Waerden (Normal) Two-Sample Analysis

Van der Waerden Scores (Normal) for Variable Gain Classified by Variable Dose					
Dose	**N**	**Sum of Scores**	**Expected Under H0**	**Std Dev Under H0**	**Mean Score**
0	16	3.346520	0.0	2.320336	0.209157
0.04	11	-3.346520	0.0	2.320336	-0.304229
Average scores were used for ties.					

Van der Waerden Two-Sample Test			
Statistic	**Z**	**Pr < Z**	**Pr > |Z|**
-3.3465	-1.4423	0.0746	0.1492

Van der Waerden One-Way Analysis		
Chi-Square	**DF**	**Pr > ChiSq**
2.0801	1	0.1492

Figure 90.12 Savage Two-Sample Analysis

Savage Scores (Exponential) for Variable Gain Classified by Variable Dose					
Dose	**N**	**Sum of Scores**	**Expected Under H0**	**Std Dev Under H0**	**Mean Score**
0	16	1.834554	0.0	2.401839	0.114660
0.04	11	-1.834554	0.0	2.401839	-0.166778
Average scores were used for ties.					

Savage Two-Sample Test			
Statistic	**Z**	**Pr < Z**	**Pr > |Z|**
-1.8346	-0.7638	0.2225	0.4450

Figure 90.12 *continued*

Savage One-Way Analysis		
Chi-Square	DF	Pr > ChiSq
0.5834	1	0.4450

The tables in Figure 90.13 display the empirical distribution function statistics, comparing the distribution of Gain for the two levels of Dose. The *p*-value for the Kolmogorov-Smirnov two-sample test is 0.6199, which indicates no rejection of the null hypothesis that the Gain distributions are identical for the two levels of Dose.

Figure 90.13 EDF Two-Sample Analysis

Kolmogorov-Smirnov Test for Variable Gain Classified by Variable Dose			
Dose	N	EDF at Maximum	Deviation from Mean at Maximum
0	16	0.250000	-0.481481
0.04	11	0.545455	0.580689
Total	27	0.370370	
Maximum Deviation Occurred at Observation 4			
Value of Gain at Maximum = 216.0			

Kolmogorov-Smirnov Two-Sample Test (Asymptotic)			
KS	0.145172	**D**	0.295455
KSa	0.754337	**Pr > KSa**	0.6199

Cramer-von Mises Test for Variable Gain Classified by Variable Dose		
Dose	N	Summed Deviation from Mean
0	16	0.098638
0.04	11	0.143474

Cramer-von Mises Statistics (Asymptotic)			
CM	0.008967	**CMa**	0.242112

Kuiper Test for Variable Gain Classified by Variable Dose		
Dose	N	Deviation from Mean
0	16	0.090909
0.04	11	0.295455

Kuiper Two-Sample Test (Asymptotic)					
K	0.386364	**Ka**	0.986440	**Pr > Ka**	0.8383

Syntax: NPAR1WAY Procedure

The following statements are available in the NPAR1WAY procedure:

```
PROC NPAR1WAY <options> ;
    BY variables ;
    CLASS variable ;
    EXACT statistic-options </ computation-options> ;
    FREQ variable ;
    OUTPUT <OUT=SAS-data-set> <output-options> ;
    STRATA variables </ options> ;
    VAR variables ;
```

The PROC NPAR1WAY statement invokes the NPAR1WAY procedure. The PROC NPAR1WAY and CLASS statements are required. Table 90.1 summarizes the statements available in the NPAR1WAY procedure.

The rest of this section provides detailed syntax information for each of these statements, beginning with the PROC NPAR1WAY statement. The remaining statements are described in alphabetical order.

Table 90.1 Summary of PROC NPAR1WAY Statements

Statement	Description
BY	Provides separate analyses for each BY group
CLASS	Identifies the classification variable
EXACT	Requests exact tests
FREQ	Identifies a frequency variable
OUTPUT	Requests an output data set
STRATA	Identifies strata variables
VAR	Identifies analysis variables

PROC NPAR1WAY Statement

```
PROC NPAR1WAY <options> ;
```

The PROC NPAR1WAY statement invokes the NPAR1WAY procedure. Optionally, it identifies the input data set and requests analyses. By default, the procedure uses the most recently created SAS data set. If you do not specify any analysis options or a STRATA statement, PROC NPAR1WAY performs an analysis of variance (ANOVA option), tests for location differences (WILCOXON, MEDIAN, SAVAGE, and VW options), and empirical distribution function tests (EDF option).

Table 90.2 lists the *options* available in the PROC NPAR1WAY statement. Descriptions of the *options* follow in alphabetical order.

Table 90.2 PROC NPAR1WAY Statement Options

Option	Description
Input Data Set	
DATA=	Names the input SAS data set
Missing Values	
MISSING	Treats missing values as a valid level
Analyses	
AB	Requests analysis of Ansari-Bradley scores
ANOVA	Requests standard analysis of variance
CONOVER	Requests analysis of Conover scores
D	Requests one-sided Kolmogorov-Smirnov statistics
DSCF	Requests pairwise multiple comparison analysis
EDF | KS	Requests empirical distribution function statistics
FP	Requests the Fligner-Policello test
HL	Requests Hodges-Lehmann estimation for two-sample data
KLOTZ	Requests analysis of Klotz scores
MEDIAN	Requests analysis of median scores
MOOD	Requests analysis of Mood scores
RMD	Requests the ratio mean deviations (RMD) test
SAVAGE	Requests analysis of Savage scores
SCORES=DATA	Requests analysis of input data scores
ST	Requests analysis of Siegel-Tukey scores
VW | NORMAL	Requests analysis of Van der Waerden (normal) scores
WILCOXON	Requests analysis of Wilcoxon scores
Analysis Details	
ADJUST	Requests median adjustment for scale analyses
ALIGN=STRATA	Requests alignment by strata
ALPHA=	Specifies the confidence level for Hodges-Lehmann estimation
CORRECT=NO	Suppresses the continuity correction for Wilcoxon and Siegel-Tukey two-sample tests
Displayed Output	
NOPRINT	Suppresses the display of all output
ORDER=	Specifies the display order of CLASS variable levels
TABLES=RESTORE	Displays score test tables in factoid form
VARINFO	Displays variable information
Plots	
PLOTS=	Requests plots from ODS Graphics

You can specify the following *options*:

AB <(ADJUST)>

requests an analysis of Ansari-Bradley scores. For more information, see the section "Ansari-Bradley Scores" on page 7390. The ADJUST suboption subtracts class medians from the input observations before performing the analysis.

ADJUST

adjusts for location differences among classes before performing tests for scale differences. This option applies to the following analyses: Ansari-Bradley (AB), Klotz (KLOTZ), Mood (MOOD), and Siegel-Tukey (ST). If you specify the ADJUST option, PROC NPAR1WAY subtracts the corresponding class median from each observation before scoring the data and performing the scale tests that you request.

You can also request adjustment for an individual scale test by specifying ADJUST in parentheses after the scale test option; for example, you can specify ST(ADJUST) to adjust by class medians before performing the Siegel-Tukey test.

ALIGN=STRATA <(*option*)>

aligns the response variable values by strata (Hodges and Lehmann 1962; Mehrotra, Lu, and Li 2010) before performing the analyses that you request. When you specify this option, you must also specify a STRATA statement to define the strata. PROC NPAR1WAY subtracts the stratum location measure from each response value in the stratum. You can specify a location measure *option* in parentheses after the ALIGN=STRATA option; by default, PROC NPAR1WAY uses the stratum median (MEDIAN) as the stratum location measure.

You can specify one of the following *options*:

HL

specifies the Hodges-Lehmann shift as the stratum location measure. The stratum Hodges-Lehmann shift is the median of all pairwise (Walsh) averages of response values in the stratum.

MEAN

specifies the mean (average) of the response values in the stratum as the stratum location measure.

MEDIAN

specifies the median of the response values in the stratum as the stratum location measure. This is the default location measure for ALIGN=STRATA.

ALPHA=α

specifies the level of the confidence limits for the Hodges-Lehmann location shift, which you can request by specifying the HL option. The value of α must be between 0 and 1; a confidence level of α produces $100(1-\alpha)\%$ confidence limits. By default, ALPHA=0.05, which produces 95% confidence limits for the location shift.

ANOVA

requests a standard analysis of variance on the raw data.

CONOVER

requests an analysis of Conover scores. For more information, see the section "Conover Scores" on page 7391.

CORRECT=NO

suppresses the continuity correction for the Wilcoxon two-sample test and the Siegel-Tukey two-sample test. For more information, see the section "Continuity Correction" on page 7388.

D

requests the one-sided Kolmogorov-Smirnov $D+$ and $D-$ statistics and their asymptotic p-values, in addition to the two-sided D statistic that the EDF option produces for two-sample data. The D option invokes the EDF option. If you request exact Kolmogorov-Smirnov tests by specifying the KS option in the EXACT statement for two-sample data, PROC NPAR1WAY provides D+ and D– by default. For more information about Kolmogorov-Smirnov statistics, see the section "Empirical Distribution Function Tests" on page 7397.

DATA=*SAS-data-set*

names the *SAS-data-set* to be analyzed by PROC NPAR1WAY. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

DSCF

requests the Dwass, Steel, Critchlow-Fligner multiple comparison procedure, which is based on pairwise two-sample rankings. This procedure is available for multisample data, where the number of CLASS variable levels is greater than 2. For more information, see the section "Multiple Comparisons Based on Pairwise Rankings" on page 7397.

EDF

KS

requests statistics based on the empirical distribution function. These include the Kolmogorov-Smirnov and Cramér–von Mises tests and, if there are only two classification levels, the Kuiper test. For more information, see the section "Empirical Distribution Function Tests" on page 7397.

The EDF option produces the Kolmogorov-Smirnov D statistic for two-sample data. You can request the one-sided D+ and D– statistics for two-sample data by specifying the D option.

FP < (REFCLASS=*class-number* | '*class-value*') >

requests the Fligner-Policello test for two-sample data. For more information, see the section "Fligner-Policello Test" on page 7394.

The REFCLASS= suboption specifies which of the two CLASS variable levels (samples) to use as the reference class X in the difference between class placement sums, $P_y - P_x$. REFCLASS=1 specifies the first class that is listed in the "Fligner-Policello Placements" table, and REFCLASS=2 specifies the second class. (By default, the table displays class levels in the order in which they appear in the input data set. If you specify the ORDER=FORMATTED option, the table displays class levels in order of their formatted value.) REFCLASS='*class-value*' identifies the reference class by the formatted value of the CLASS variable.

By default, PROC NPAR1WAY uses the larger of the two classes as the reference class X. If both classes have the same number of observations, PROC NPAR1WAY uses the class that appears second in the "Fligner-Policello Placements" table as the reference class.

HL < (REFCLASS=*class-number* | '*class-value*') >

requests Hodges-Lehmann estimation of the location shift for two-sample data. This option also provides asymptotic confidence limits for the location shift (which are sometimes known as Moses confidence limits). For more information, see the section "Hodges-Lehmann Estimation of Location Shift" on page 7392. You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits for the location shift.

The REFCLASS= suboption specifies which of the two CLASS variable levels (samples) to use as the reference class X in the location shift, $Y - X$. REFCLASS=1 specifies the first class that is listed in the "Wilcoxon Scores" table, and REFCLASS=2 specifies the second class. (By default, the table displays class levels in the order in which they appear in the input data set. If you specify the ORDER=FORMATTED option, the table displays class levels in order of their formatted value.) REFCLASS='*class-value*' identifies the reference class by the formatted value of the CLASS variable.

By default, PROC NPAR1WAY uses the larger of the two classes as the reference class X. If both classes have the same number of observations, PROC NPAR1WAY uses the class that appears second in the "Wilcoxon Scores" table as the reference class.

KLOTZ < (ADJUST) >

requests an analysis of Klotz scores. For more information, see the section "Klotz Scores" on page 7390. If you specify the ADJUST suboption, PROC NPAR1WAY subtracts the corresponding class median from each observation before performing the analysis.

MEDIAN

requests an analysis of median scores. When there are two classification levels, this option produces the two-sample median test. When there are more than two samples, this option produces the multisample median test, which is also known as the Brown-Mood test. For more information, see the section "Median Scores" on page 7389.

MISSING

treats missing values as a valid nonmissing level for CLASS and STRATA variables. By default, PROC NPAR1WAY excludes an observations from the analysis if the value of the CLASS variable is missing. By default, PROC NPAR1WAY excludes an observations from the stratified analysis if the value of the STRATA variable is missing. For more information, see the section "Missing Values" on page 7385.

MOOD < (ADJUST) >

requests an analysis of Mood scores. For more information, see the section "Mood Scores" on page 7391. If you specify the ADJUST suboption, PROC NPAR1WAY subtracts the corresponding class median from each observation before performing the analysis.

NOPRINT

suppresses the display of all output. You can use the NOPRINT option when you only want to create an output data set. This option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

ORDER=DATA | FORMATTED

specifies the sort order of the CLASS variable levels in the displayed output.

You can specify one of the following orders:

DATA
orders CLASS variable levels by order of appearance in the input data set.

FORMATTED
orders CLASS variable levels by external formatted value, except for numeric variables with no explicit format, which are ordered by their unformatted (internal) value.

By default, ORDER=DATA when the value of the DATA= option is a SAS data set. When the value of the DATA= option is a CAS table, ORDER=DATA is not allowed and PROC NPAR1WAY uses ORDER=FORMATTED.

The FORMATTED order is machine-dependent. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*plot-option*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*plot-option*) > < . . . *plot-request* < (*plot-option*) > >) >
controls the plots that are produced through ODS Graphics. *Plot-requests* specify the plots to produce, and *plot-options* control the appearance and content of the plots. You can specify *plot-options* in parentheses after a *plot-request*. A *global-plot-option* applies to all plots for which it is available, unless it is altered by a specific *plot-option*. You can specify *global-plot-options* in parentheses after the PLOTS option.

When you specify only one *plot-request*, you can omit the parentheses around the request. For example:

```
plots=all
plots=wilcoxonboxplot
plots=(wilcoxonboxplot edfplot)
plots(only)=(medianplot normalboxplot)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc npar1way plots=wilcoxonboxplot;
   variable response;
   class treatment;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, PROC NPAR1WAY produces all plots that are associated with the analyses that you request. If you request a plot by specifying the PLOTS= option but do not request the corresponding analysis, PROC NPAR1WAY automatically invokes that analysis. For example, if you specify PLOTS=CONOVERBOXPLOT but do not also specify the CONOVER option in the PROC NPAR1WAY statement, PROC NPAR1WAY produces the Conover scores analysis in addition to the box plot.

You can suppress default plots and request specific plots by using the PLOTS(ONLY)= option; PLOTS(ONLY)=(*plot-requests*) produces only the plots that are specified as *plot-requests*. You can suppress all plots by specifying the PLOTS=NONE option. The PLOTS= option has no effect when you specify the NOPRINT option.

PROC NPAR1WAY provides box plots of scored data, median plots, and empirical distribution plots. See Figure 90.7, Output 90.1.2, Output 90.1.4, and Output 90.2.2 for examples of plots that PROC NPAR1WAY produces. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Global Plot Options

A *global-plot-option* applies to all plots for which the option is available unless it is altered by a specific *plot-option*. You can specify the following *global-plot-options* in parentheses after the PLOTS option. You cannot specify both the STATS and the NOSTATS *global-plot-options* in the same statement.

NOSTATS
suppresses the *p*-values that are displayed by default in the plots.

ONLY
suppresses the default plots and requests only the plots that are specified as *plot-requests*.

STATS
displays *p*-values in the plots. This is the default.

Plot Requests

You can specify the following *plot-requests*:

ABBOXPLOT
AB
requests a box plot of Ansari-Bradley scores. This plot is associated with the Ansari-Bradley analysis, which you request by specifying the AB option.

ALL
requests all plots that are associated with the specified analyses. This is the default if you do not specify the ONLY *global-plot-option*.

ANOVABOXPLOT
ANOVA
requests a box plot of the raw data. This plot is associated with the analysis of variance based on the raw data, which you request by specifying the ANOVA option.

CONOVERBOXPLOT
CONOVER
requests a box plot of Conover scores. This plot is associated with the Conover analysis, which you request by specifying the CONOVER option.

DATASCORESBOXPLOT

DATASCORES

requests a box plot of raw data scores. This plot is associated with the analysis that uses input data as scores, which you request by specifying the SCORES=DATA option.

EDFPLOT

EDF

requests an empirical distribution plot. This plot is associated with the analyses based on the empirical distribution function, which you request by specifying the EDF option.

FPBOXPLOT

FP

requests a box plot of Fligner-Policello placements. This plot is associated with the Fligner-Policello analysis, which you request by specifying the FP option.

KLOTZBOXPLOT

KLOTZ

requests a box plot of Klotz scores. This plot is associated with the Klotz analysis, which you request by specifying the KLOTZ option.

MEDIANPLOT

MEDIAN

requests a stacked bar chart showing the frequencies above and below the overall median. This plot is associated with the median score analysis, which you request by specifying the MEDIAN option.

MOODBOXPLOT

MOOD

requests a box plot of Mood scores. This plot is associated with the Mood analysis, which you request by specifying the MOOD option.

NONE

suppresses all plots.

SAVAGEBOXPLOT

SAVAGE

requests a box plot of Savage scores. This plot is associated with the Savage analysis, which you request by specifying the SAVAGE option.

STBOXPLOT

ST

requests a box plot of Siegel-Tukey scores. This plot is associated with the Siegel-Tukey analysis, which you request by specifying the ST option.

VWBOXPLOT

VW

NORMALBOXPLOT

NORMAL

requests a box plot of Van der Waerden (normal) scores. This plot is associated with the Van der Waerden analysis, which you request by specifying the VW (NORMAL) option.

WILCOXONBOXPLOT

WILCOXON

requests a box plot of Wilcoxon scores. This plot is associated with the Wilcoxon analysis, which you request by specifying the WILCOXON option.

Plot Options

The following *plot-options* are available for any *plot-request*. You cannot specify both the STATS and the NOSTATS *plot-options* for the same plot. If you specify both the NOSTATS *global-plot-option* and the STATS *plot-option* for an individual *plot-request*, the STATS *plot-option* overrides the *global-plot-option* and displays statistics in the individual plot.

NOSTATS

suppresses the *p*-values that are displayed in the plot by default.

STATS

displays *p*-values in the plot. This is the default.

RMD

requests the ratio mean deviations (RMD) exact test of scale for two-sample data. For more information, see the section "Ratio Mean Deviations (RMD) Exact Test" on page 7396.

You can also request the RMD test by specifying the RMD option in the EXACT statement, which enables you to specify *computation-options* and request the following additional statistics: the exact mid *p*-value, the exact point probability, and Monte Carlo estimates of the exact *p*-values.

SAVAGE

requests an analysis of Savage scores. For more information, see the section "Savage Scores" on page 7390.

SCORES=DATA < (ADJUST) >

PERM < (ADJUST) >

requests an analysis that uses input data as scores. This option gives you the flexibility to construct any scores for your data with the DATA step and then analyze these scores with PROC NPAR1WAY. For more information, see the section "Scores for Linear Rank and One-Way ANOVA Tests" on page 7389.

To produce the two-sample permutation test that is known as Pitman's test, provide raw (unscored) data in the input data set and specify the SCORES=DATA option in the EXACT statement. For more information, see the section "Exact Tests" on page 7399.

If you specify the ADJUST suboption, PROC NPAR1WAY subtracts the corresponding class median from each observation before performing the analysis.

ST < (ADJUST) >

requests an analysis of Siegel-Tukey scores. For more information, see the section "Siegel-Tukey Scores" on page 7390. If you specify the ADJUST suboption, PROC NPAR1WAY subtracts the corresponding class median from each observation before performing the analysis.

TABLES=RESTORE

displays score test tables in legacy, factoid (label-value) format. By default, PROC NPAR1WAY displays score test tables in tabular format.

Score test tables include the two-sample test, one-way analysis, and exact test tables that PROC NPAR1WAY produces when you specify any of the following options in the PROC NPAR1WAY or EXACT statement: AB, CONOVER, FP, KLOTZ, MEDIAN, MOOD, SAVAGE, SCORES=DATA, ST, VW | NORMAL, or WILCOXON. For more information, see the sections "Displayed Output" on page 7409 and "ODS Table Names" on page 7417.

VARINFO < (TIES) >

displays a "Variable Information" table for each analysis variable that you specify in the VAR statement. This table provides the following data summary information: number of observations and number of classes used in the analysis, number of missing observations and number of empty classes, sum of frequencies used (if you specify a FREQ statement), and number of strata (if you specify the ALIGN=STRATA option).

If you specify the TIES suboption, the "Variable Information" table also includes the number of variable levels and the number of tied observations.

VW

NORMAL

requests an analysis of Van der Waerden (normal) scores. For more information, see the section "Van der Waerden (Normal) Scores" on page 7389.

WILCOXON

requests an analysis of Wilcoxon scores. When there are two classification levels (samples), this option produces the Wilcoxon rank-sum test. For any number of classification levels, this option produces the Kruskal-Wallis test. For more information, see the section "Wilcoxon Scores" on page 7389.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC NPAR1WAY to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the NPAR1WAY procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* **;**

The CLASS statement, which is required, names one and only one classification variable. The variable can be character or numeric. The CLASS variable identifies groups (samples) in the data, and PROC NPAR1WAY provides analyses to examine differences among these groups. There can be two or more groups in the data.

EXACT Statement

EXACT *statistic-options* < / *computation-options* > **;**

The EXACT statement requests exact tests and confidence limits for selected statistics. The *statistic-options* identify which statistics to compute, and the *computation-options* specify options for computing exact statistics. For more information, see the section "Exact Tests" on page 7399.

NOTE: PROC NPAR1WAY computes exact tests by using fast and efficient algorithms that are superior to direct enumeration. Exact tests are appropriate when a data set is small, sparse, skewed, or heavily tied. For some large problems, computation of exact tests might require a large amount of time and memory. Consider using asymptotic tests for such problems. Alternatively, when asymptotic methods might not be sufficient for such large problems, consider using Monte Carlo estimation of exact p-values. You can request Monte Carlo estimation by specifying the MC *computation-option* in the EXACT statement. For more information, see the section "Computational Resources" on page 7401.

Statistic Options

The *statistic-options* specify which exact tests to compute. Table 90.3 lists the available *statistic-options* and the exact statistics that are computed. Descriptions of the *statistic-options* follow Table 90.3 in alphabetical order.

PROC NPAR1WAY provides exact p-values for the following nonparametric location and scale tests: Wilcoxon, median, Van der Waerden (normal), Savage, Siegel-Tukey, Ansari-Bradley, Klotz, Mood, and Conover. The procedure also provides exact p-values for tests that use the raw input data as scores. These exact tests are available when the data are classified into two levels (two-sample tests) and when the data are classified into more than two levels (multisample tests). The two-sample exact tests are based on simple linear rank statistics, and the multisample exact tests are based on one-way ANOVA statistics.

For two-sample data, PROC NPAR1WAY also provides the exact Kolmogorov-Smirnov test, the exact ratio mean deviations (RMD) test, and exact Hodges-Lehmann confidence limits for the location shift.

If you list no *statistic-options* in the EXACT statement, PROC NPAR1WAY computes all available exact *p*-values for those tests that you request in the PROC NPAR1WAY statement.

Table 90.3 EXACT Statement Statistic Options

Statistic Option	Exact Test
AB	Ansari-Bradley test
CONOVER	Conover test
HL	Hodges-Lehmann confidence limits
KLOTZ	Klotz test
KS | EDF	Two-sample Kolmogorov-Smirnov test
MEDIAN	Median test
MOOD	Mood test
RMD	Ratio mean deviations (RMD) test
SAVAGE	Savage test
SCORES=DATA	Test with input data as scores
ST	Siegel-Tukey test
VW | NORMAL	Van der Waerden (normal scores) test
WILCOXON	Wilcoxon test for two-sample data or Kruskal-Wallis test for multisample data

You can specify the following *statistic-options*:

AB

requests an exact Ansari-Bradley test. For more information, see the sections "Ansari-Bradley Scores" on page 7390 and "Exact Tests" on page 7399. The AB option in the PROC NPAR1WAY statement provides Ansari-Bradley score analysis and asymptotic tests.

CONOVER

requests an exact Conover test. For more information, see the sections "Conover Scores" on page 7391 and "Exact Tests" on page 7399. The CONOVER option in the PROC NPAR1WAY statement provides Conover score analysis and asymptotic tests.

HL

requests exact Hodges-Lehmann confidence limits for the location shift for two-sample data. For more information, see the section "Hodges-Lehmann Estimation of Location Shift" on page 7392. The HL option in the PROC NPAR1WAY statement provides asymptotic Hodges-Lehmann confidence limits.

You can specify the level of the confidence limits in the ALPHA= option in the PROC NPAR1WAY statement. By default, ALPHA=0.05, which produces 95% confidence limits for the location shift.

KLOTZ

requests an exact Klotz test. For more information, see the sections "Klotz Scores" on page 7390 and "Exact Tests" on page 7399. The KLOTZ option in the PROC NPAR1WAY statement provides Klotz score analysis and asymptotic tests.

KS

EDF

requests an exact Kolmogorov-Smirnov two-sample test. For more information, see the section "Empirical Distribution Function Tests" on page 7397. The EDF option in the PROC NPAR1WAY statement provides the asymptotic Kolmogorov-Smirnov test and other statistics that are based on the empirical distribution function. The D option in the PROC NPAR1WAY statement provides the asymptotic one-sided Kolmogorov-Smirnov tests for two-sample data.

MEDIAN

requests an exact median test. For more information, see the sections "Median Scores" on page 7389 and "Exact Tests" on page 7399. The MEDIAN option in the PROC NPAR1WAY statement provides median score analysis and asymptotic tests.

MOOD

requests an exact Mood test. For more information, see the sections "Mood Scores" on page 7391 and "Exact Tests" on page 7399. The MOOD option in the PROC NPAR1WAY statement provides Mood score analysis and asymptotic tests.

RMD

requests the ratio mean deviations (RMD) exact test for two-sample data. For more information, see the sections "Ratio Mean Deviations (RMD) Exact Test" on page 7396 and "Exact Tests" on page 7399.

SAVAGE

requests an exact Savage test. For more information, see the sections "Savage Scores" on page 7390 and "Exact Tests" on page 7399. The SAVAGE option in the PROC NPAR1WAY statement provides Savage score analysis and asymptotic tests.

SCORES=DATA

PERM

requests an exact test that uses the input data as scores. For two-sample data, the test is based on the rank-sum statistic. For multisample data, the test is based on the one-way ANOVA statistic. For more information, see the sections "Scores for Linear Rank and One-Way ANOVA Tests" on page 7389 and "Exact Tests" on page 7399. The SCORES=DATA option in the PROC NPAR1WAY statement provides analysis of the data scores and the corresponding asymptotic test.

ST

requests an exact Siegel-Tukey test. For more information, see the sections "Siegel-Tukey Scores" on page 7390 and "Exact Tests" on page 7399. The ST option in the PROC NPAR1WAY statement provides analysis of Siegel-Tukey scores and asymptotic tests.

VW

NORMAL

requests an exact Van der Waerden (normal scores) test. For more information, see the sections "Van der Waerden (Normal) Scores" on page 7389 and "Exact Tests" on page 7399. The VW (NORMAL) option in the PROC NPAR1WAY statement provides analysis of Van der Waerden (normal) scores and asymptotic tests.

WILCOXON

requests an exact Wilcoxon test. When the data consist of two classification levels (samples), the exact test is based on the Wilcoxon rank-sum statistic. When the data consist of more than two levels (multisample data), the exact test is based on the one-way ANOVA statistic for Wilcoxon scores, which is the Kruskal-Wallis statistic. For more information, see the sections "Wilcoxon Scores" on page 7389 and "Exact Tests" on page 7399. The WILCOXON option in the PROC NPAR1WAY statement provides analysis of Wilcoxon scores and the asymptotic Wilcoxon and Kruskal-Wallis tests.

Computation Options

Computation-options specify options for computing exact statistics. You can specify the following *computation-options*:

ALPHA=α

specifies the level of the confidence limits for Monte Carlo p-value estimates. The value of α must be between 0 and 1; a confidence level of α produces $100(1-\alpha)\%$ confidence limits. By default, ALPHA=0.01, which produces 99% confidence limits for the exact p-values.

This option invokes the MC option.

MAXTIME=*value*

specifies the maximum clock time (in seconds) that PROC NPAR1WAY can use to compute an exact p-value. If the procedure does not complete the computation within the specified time, the computation terminates. The maximum time value must be a positive number. This option is available for exact p-value computation and for Monte Carlo estimation of exact p-values. For more information, see the section "Computational Resources" on page 7401.

MC

requests Monte Carlo estimation of exact p-values instead of direct exact p-value computation. Monte Carlo estimation can be useful for large problems where exact computations require a substantial amount of time and memory but asymptotic approximations might not be sufficient. For more information, see the section "Monte Carlo Estimation" on page 7402.

This option is available for all EXACT *statistic-options* except the HL option, which produces exact Hodges-Lehmann confidence limits. The ALPHA=, N=, and SEED= options invoke the MC option.

MIDP

requests exact mid p-values for the exact tests. The exact mid p-value is defined as the exact p-value minus half the exact point probability. For two-sample data, PROC NPAR1WAY provides exact mid p-values for the one-sided tests of the linear rank statistics. For multisample data, PROC NPAR1WAY provides exact mid p-values for the one-way ANOVA tests. For more information, see the section "Definition of p-Values" on page 7400.

The MIDP option is available for all EXACT statement *statistic-options* except the HL option, which produces exact Hodges-Lehmann confidence limits. You cannot specify both the MIDP option and the MC option.

N=*n*

specifies the number of samples for Monte Carlo estimation. The value of *n* must be a positive integer. Larger values of *n* produce more precise estimates of exact *p*-values. Because larger values of *n* generate more samples, the computation time increases. By default, N=10,000.

This option invokes the MC option.

PFORMAT=*format-name* | **EXACT**

specifies the display format for exact *p*-values. PROC NPAR1WAY applies this format to one- and two-sided exact *p*-values, exact point probabilities, and exact mid *p*-values. By default, PROC NPAR1WAY displays exact *p*-values in the PVALUE6.4 format.

You can provide a *format-name* or you can specify PFORMAT=EXACT to control the format of exact *p*-values. The value of *format-name* can be any standard SAS numeric format or a user-defined format. The format length must not exceed 24. For information about formats, see the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS format in *SAS Formats and Informats: Reference*.

If you specify PFORMAT=EXACT, PROC NPAR1WAY uses the 6.4 format to display exact *p*-values that are greater than or equal to 0.001; the procedure uses the E10.3 format to display values that are between 0.000 and 0.001.

POINT

requests exact point probabilities for the exact tests. The exact point probability is the exact probability that the test statistic equals the observed value. For two-sample data, PROC NPAR1WAY provides exact point probabilities for the one-sided tests of the linear rank statistics. For multisample data, PROC NPAR1WAY provides exact point probabilities for the one-way ANOVA tests. For more information, see the section "Definition of *p*-Values" on page 7400.

The POINT option is available for all EXACT statement *statistic-options* except the HL option, which produces exact Hodges-Lehmann confidence limits. You cannot specify both the POINT option and the MC option.

SEED=*number*

specifies the initial seed for random number generation for Monte Carlo estimation. The value of the SEED= option must be an integer. If you do not specify the SEED= option or if the SEED= value is negative or 0, PROC NPAR1WAY uses the time of day from the computer's clock to obtain the initial seed.

This option invokes the MC option.

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a numeric variable that represents a frequency of occurrence for each observation in the input data set. PROC NPAR1WAY treats the data set as if each observation appears *n* times, where *n* is the value of the FREQ variable for the observation. The sum of the FREQ variable values represents the total number of observations, and the analysis is based on this expanded number of observations.

Noninteger values of the FREQ variable are truncated to the largest integer less than the FREQ value. An observation is used in the analysis only if the value of the FREQ variable is greater than or equal to 1.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*> < *output-options*> **;**

The OUTPUT statement creates a SAS data set that contains statistics that PROC NPAR1WAY computes. Table 90.4 lists the statistics that can be stored in the output data set. You identify which statistics to include by specifying *output-options*.

The output data set contains one observation for each analysis variable that you name in the VAR statement. If you use a BY statement, the output data set contains an observation or set of observations for each BY group. For more information, see the section "Contents of the Output Data Set" on page 7403.

As an alternative to the OUTPUT statement, you can use the Output Delivery System (ODS) to store statistics that PROC NPAR1WAY computes. ODS can create a SAS data set from any table that PROC NPAR1WAY produces. For more information, see the section "ODS Table Names" on page 7417 and Chapter 23, "Using the Output Delivery System."

The OUTPUT statement data set does not include Monte Carlo estimates of exact *p*-values, multiple comparison statistics, or stratified analysis statistics. You can use ODS to store these statistics in output data sets.

You can specify the following *options*:

OUT=*SAS-data-set*

names the output data set. When you use an OUTPUT statement but do not specify the OUT= option, PROC NPAR1WAY creates a data set and names it by using the DATA*n* convention.

output-options

specifies the statistics to include in the output data set. Table 90.4 lists the *output-options* that are available in the OUTPUT statement. Descriptions of the *output-options* follow the table in alphabetical order.

When you specify an *output-option*, the output data set includes the test statistic and associated values from the analysis that you specify. The associated values might include standardized statistics, one- and two-sided *p*-values, exact *p*-values, degrees of freedom, and confidence limits. See the section "Contents of the Output Data Set" on page 7403 for a list of output data set variables that each *output-option* produces.

If you do not specify any *output-options* in the OUTPUT statement, the output data set includes all available statistics from the analyses that you request in the PROC NPAR1WAY statement. If you

request a statistic in the OUTPUT statement but do not request the corresponding analysis in the PROC NPAR1WAY statement, PROC NPAR1WAY performs the corresponding analysis.

Table 90.4 OUTPUT Statement Output Options

Output Option	Output Data Set Statistics
AB	Ansari-Bradley test
ANOVA	Analysis of variance
CONOVER	Conover test
EDF | KS	Kolmogorov-Smirnov test, Cramér–von Mises test, and Kuiper test for two-sample data
FP	Fligner-Policello test
HL	Hodges-Lehmann estimates
KLOTZ	Klotz test
MEDIAN	Median test
MOOD	Mood test
RMD	Ratio mean deviations (RMD) test
SAVAGE	Savage test
SCORES=DATA	Test with input data as scores
ST	Siegel-Tukey test
VW | NORMAL	Van der Waerden (normal scores) test
WILCOXON	Wilcoxon test for two-sample data and Kruskal-Wallis test

You can specify the following *output-options*:

AB

includes statistics from the Ansari-Bradley analysis in the output data set. The AB option in the PROC NPAR1WAY statement requests analysis of Ansari-Bradley scores. For more information, see the section "Ansari-Bradley Scores" on page 7390.

ANOVA

includes analysis of variance statistics in the output data set. The ANOVA option in the PROC NPAR1WAY statement requests a standard analysis of variance for the raw data.

CONOVER

includes statistics from the Conover analysis in the output data set. The CONOVER option in the PROC NPAR1WAY statement requests analysis of Conover scores. For more information, see the section "Conover Scores" on page 7391.

EDF

KS

includes empirical distribution function statistics in the output data set. The EDF option in the PROC NPAR1WAY statement requests computation of these statistics, which include the Kolmogorov-Smirnov test, the Cramér–von Mises test, and the Kuiper test for two sample-data. For more information, see the section "Empirical Distribution Function Tests" on page 7397.

FP

includes the Fligner-Policello test in the output data set. The FP option in the PROC NPAR1WAY statement requests the Fligner-Policello test, which is available for two-sample data. For more information, see the section "Fligner-Policello Test" on page 7394.

HL

includes the Hodges-Lehmann estimate and confidence limits in the output data set. The HL option in the PROC NPAR1WAY statement requests Hodges-Lehmann estimation, which is available for two-sample data. For more information, see the section "Hodges-Lehmann Estimation of Location Shift" on page 7392.

KLOTZ

includes statistics from the Klotz analysis in the output data set. The KLOTZ option in the PROC NPAR1WAY statement requests analysis of Klotz scores. For more information, see the section "Klotz Scores" on page 7390.

MEDIAN

includes statistics from the median analysis in the output data set. The MEDIAN option in the PROC NPAR1WAY statement requests analysis of median scores. For more information, see the section "Median Scores" on page 7389.

MOOD

includes statistics from the Mood analysis in the output data set. The MOOD option in the PROC NPAR1WAY statement requests analysis of Mood scores. For more information, see the section "Mood Scores" on page 7391.

RMD

includes statistics from the ratio mean deviations (RMD) test in the output data set. The RMD option in the PROC NPAR1WAY statement and the RMD option in the EXACT statement request the RMD test. For more information, see the section "Ratio Mean Deviations (RMD) Exact Test" on page 7396.

SAVAGE

includes statistics from the Savage analysis in the output data set. The SAVAGE option in the PROC NPAR1WAY statement requests analysis of SAVAGE scores. For more information, see the section "Savage Scores" on page 7390.

SCORES=DATA

PERM

includes statistics from the analysis of data scores in the output data set. The SCORES=DATA option in the PROC NPAR1WAY statement requests an analysis that uses the raw input data as scores. For more information, see the section "Scores for Linear Rank and One-Way ANOVA Tests" on page 7389.

ST

includes statistics from the Siegel-Tukey analysis in the output data set. The ST option in the PROC NPAR1WAY statement requests analysis of Siegel-Tukey scores. For more information, see the section "Siegel-Tukey Scores" on page 7390.

VW

NORMAL

includes statistics from the Van der Waerden (normal scores) analysis in the output data set. The VW (NORMAL) option in the PROC NPAR1WAY statement requests analysis of Van der Waerden scores. For more information, see the section "Van der Waerden (Normal) Scores" on page 7389.

WILCOXON

includes statistics from the Wilcoxon analysis in the output data set. The WILCOXON option in the PROC NPAR1WAY statement requests analysis of Wilcoxon scores. For two-sample data, this analysis includes the Wilcoxon rank-sum test. For all data (two-sample and multisample), the analysis includes the Kruskal-Wallis test. For more information, see the section "Wilcoxon Scores" on page 7389.

STRATA Statement

STRATA *variables* < / *options* > **;**

The STRATA statement names one or more *variables* that define the strata. The levels of the STRATA variables combine to form the strata, which are nonoverlapping subgroups.

The STRATA variables can be either character or numeric. PROC NPAR1WAY uses the formatted values of the STRATA variables to determine the STRATA variable levels. Thus, you can use formats to group values into levels. For more information, see the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

PROC NPAR1WAY excludes an observation from the stratified analysis if it has a missing value for any STRATA variable unless you specify the MISSING option in the PROC NPAR1WAY statement. When you specify the MISSING option, PROC NPAR1WAY treats missing values of the STRATA variables as valid, nonmissing levels. For more information, see the section "Missing Values" on page 7385.

You can request stratified analyses by specifying one or more score type *options* (WILCOXON, MEDIAN, VW | NORMAL, SAVAGE, and SCORES=DATA) in the STRATA statement. If you do not specify a score type *option* in the STRATA statement (or the ALIGN=STRATA option in the PROC NPAR1WAY statement), PROC NPAR1WAY performs stratified analysis of Wilcoxon scores by default. The statistic options in the PROC NPAR1WAY statement request unstratified analyses; the statistic options in the STRATA statement request stratified analyses.

Stratified analysis is available for two-sample data (when the data are classified into two levels by the CLASS variable). For more information, see the section "Stratified Analysis" on page 7391.

If you specify the ALIGN=STRATA option in the PROC NPAR1WAY statement along with a STRATA statement, PROC NPAR1WAY aligns the analysis variable values by strata before performing the unstratified analyses that you request (by specifying options in the PROC NPAR1WAY statement). When you specify the ALIGN=STRATA option to compute aligned analysis, you cannot also specify STRATA statement *options* to perform stratified analysis.

You can specify the following *options*:

MEDIAN

requests a stratified analysis of median scores. For more information, see the sections "Median Scores" on page 7389 and "Stratified Analysis" on page 7391.

RANKS=STRATUM | OVERALL

specifies the type of ranking to be used for stratified analysis. RANKS=STRATUM ranks (and scores) the observations separately within each stratum. RANKS=OVERALL ranks (and scores) the observations overall (without regard to stratum classification) before stratified analysis is performed. By default, RANKS=STRATUM. For more information, see the section "Stratified Analysis" on page 7391.

SAVAGE

requests a stratified analysis of Savage scores. For more information, see the sections "Savage Scores" on page 7390 and "Stratified Analysis" on page 7391.

SCORES=DATA

PERM

requests a stratified analysis of the raw (unscored) values of the analysis variable. This option enables you to construct any scores for your data (for example, by using the DATA step) and analyze the scores by using PROC NPAR1WAY. For more information, see the sections "Scores for Linear Rank and One-Way ANOVA Tests" on page 7389 and "Stratified Analysis" on page 7391.

VW

NORMAL

requests a stratified analysis of Van der Waerden (normal) scores. For more information, see the sections "Van der Waerden (Normal) Scores" on page 7389 and "Stratified Analysis" on page 7391.

WEIGHTS=STRATUM | EQUAL

specifies the stratum weights for stratified analysis. WEIGHTS=STRATUM weights the stratum statistics by stratum size; the weight of stratum i is $1/(n_i + 1)$, where n_i is the number of observations (frequency) in stratum i. WEIGHTS=EQUAL weights the stratum statistics equally; the weight of each stratum is 1. For more information, see the section "Stratified Analysis" on page 7391.

When you specify RANKS=OVERALL, WEIGHTS=EQUAL by default; otherwise, WEIGHTS=STRATUM by default.

WILCOXON

requests a stratified analysis of Wilcoxon scores. For more information, see the sections "Wilcoxon Scores" on page 7389 and "Stratified Analysis" on page 7391. By default (when WEIGHTS=STRATUM and RANKS=STRATUM), the stratified Wilcoxon test is the van Elteren test (Van Elteren 1960).

VAR Statement

VAR *variables* **;**

The VAR statement names the response (dependent) variables to be included in the analysis. These variables must be numeric. If you omit the VAR statement, PROC NPAR1WAY includes all numeric variables in the data set except the BY, CLASS, FREQ, and STRATA variables.

Details: NPAR1WAY Procedure

Missing Values

If an observation has a missing value for a response (VAR) variable, PROC NPAR1WAY excludes the observation from the analysis of that variable. If an observation has a missing or nonpositive value for the FREQ variable, PROC NPAR1WAY excludes that observation from the analysis.

To display the number of missing response values, you can specify the VARINFO option in the PROC NPAR1WAY statement. This provides a "Variable Information" table for each response variable. The table includes the number of observations that are used in the analysis and the number of missing values.

If an observation has a missing value for the CLASS variable, PROC NPAR1WAY excludes that observation from the analysis unless you specify the MISSING option in the PROC NPAR1WAY statement. If you specify the MISSING option, PROC NPAR1WAY treats missing values of the CLASS variable as a valid level.

If an observation has a missing value for a STRATA variable, PROC NPAR1WAY excludes that observation from the stratified analyses unless you specify the MISSING option in the PROC NPAR1WAY statement. If you specify the MISSING option, PROC NPAR1WAY treats missing values of the STRATA variables as valid levels.

When you specify a STRATA statement, PROC NPAR1WAY displays a "Stratum Information" table, which lists the stratum variable levels and the number of observations (frequency) in each stratum. This table summarizes the valid observations in the input data set (or BY group). An observation is valid if it has nonmissing values of the CLASS and STRATA variables (unless you specify the MISSING option) and a positive value of the FREQ variable. PROC NPAR1WAY then analyzes each response (VAR) variable separately; each analysis includes valid observations that have nonmissing values for the individual response variable.

PROC NPAR1WAY treats missing BY variable values like any other BY variable value. The missing values form a separate, valid BY group.

Tied Values

Tied values occur when two or more observations are equal, whether the observations occur in the same sample or in different samples. In theory, nonparametric tests were developed for continuous distributions where the probability of a tie is zero. In practice, however, ties often occur. PROC NPAR1WAY uses the same method to handle ties for all score types. The procedure computes the scores as if there were no ties, averages the scores for tied observations, and assigns this average score to each observation that has the same value.

To display the number of tied response values, you can specify the VARINFO(TIES) option in the PROC NPAR1WAY statement. This provides a "Variable Information" table for each response variable. The table includes the number of variable levels and the number of tied observations.

When there are tied values, PROC NPAR1WAY first sorts the observations in ascending order and assigns ranks as if there were no ties. Then the procedure computes the scores based on these ranks by using the formula for the specified score type. The procedure averages the scores for tied observations and assigns this average score to each of the tied observations. Thus, all equal data values have the same score value. PROC NPAR1WAY then computes the test statistic from these scores.

The asymptotic tests might be less accurate when the distribution of the data is heavily tied. For such data, it might be appropriate to use the exact tests provided by PROC NPAR1WAY as described in the section "Exact Tests" on page 7399.

When computing empirical distribution function statistics for data with ties, PROC NPAR1WAY uses the formulas given in the section "Empirical Distribution Function Tests" on page 7397. No special handling of ties is necessary.

PROC NPAR1WAY bases its computations on the internal numeric values of the analysis variables; the procedure does not format or round these values before analysis. When values differ in their internal representation, even slightly, PROC NPAR1WAY does not treat them as tied values. If this is a concern for your data, round the analysis variables by an appropriate amount before invoking PROC NPAR1WAY. For information about the ROUND function, see the discussion in *SAS Functions and CALL Routines: Reference*.

Statistical Computations

Simple Linear Rank Tests for Two-Sample Data

Statistics of the form

$$S = \sum_{j=1}^{n} c_j \, a(R_j)$$

are called *simple linear rank statistics*, where

R_j	is the rank of observation j
$a(R_j)$	is the score based on the rank of observation j
c_j	indicates the class of observation j
n	is the total number of observations

For two-sample data (where the observations are classified into two levels), PROC NPAR1WAY calculates simple linear rank statistics for the scores that you specify. The section "Scores for Linear Rank and One-Way ANOVA Tests" on page 7389 describes the available scores, which you can use to test for differences in location and differences in scale.

To compute the linear rank statistic S, PROC NPAR1WAY sums the scores of the observations in the smaller of the two samples. If both samples have the same number of observations, PROC NPAR1WAY sums those scores for the sample (level) that appears first in the displayed output. (By default, class levels are displayed in the order in which they appear in the input data set. If you specify the ORDER=FORMATTED option, class levels are displayed in order of their formatted value.)

For each score that you specify, PROC NPAR1WAY computes an asymptotic test of the null hypothesis of no difference between the two classification levels. Exact tests are also available for these two-sample linear rank statistics. PROC NPAR1WAY computes exact tests for each score type that you specify in the EXACT statement. For more information, see the section "Exact Tests" on page 7399.

To compute an asymptotic test for a linear rank sum statistic, PROC NPAR1WAY uses a standardized test statistic z, which has an asymptotic standard normal distribution under the null hypothesis. The standardized test statistic is computed as

$$z = (S - \mathrm{E}_0(S)) \,/\, \sqrt{\mathrm{Var}_0(S)}$$

where $\mathrm{E}_0(S)$ is the expected value of S under the null hypothesis, and $\mathrm{Var}_0(S)$ is the variance under the null hypothesis. As shown in Randles and Wolfe (1979),

$$\mathrm{E}_0(S) = \frac{n_1}{n} \sum_{j=1}^{n} a(R_j)$$

where n_1 is the number of observations in the first (smaller) class level (sample), n_2 is the number of observations in the other class level, and

$$\mathrm{Var}_0(S) = \frac{n_1 n_2}{n(n-1)} \sum_{j=1}^{n} \left(a(R_j) - \bar{a}\right)^2$$

where $\bar{a}$ is the average score,

$$\bar{a} = \frac{1}{n} \sum_{j=1}^{n} a(R_j)$$

Definition of p-Values

PROC NPAR1WAY computes one-sided and two-sided asymptotic p-values for each two-sample linear rank test. When the test statistic z is greater than its null hypothesis expected value of 0, PROC NPAR1WAY computes the right-sided p-value, which is the probability of a larger value of the statistic occurring under the null hypothesis. When the test statistic is less than or equal to 0, PROC NPAR1WAY computes the left-sided p-value, which is the probability of a smaller value of the statistic occurring under the null hypothesis. The one-sided p-value $P_1(z)$ can be expressed as

$$P_1(z) = \begin{cases} \mathrm{Prob}(Z > z) & \text{if } z > 0 \\ \mathrm{Prob}(Z < z) & \text{if } z \le 0 \end{cases}$$

where Z has a standard normal distribution. The two-sided p-value $P_2(z)$ is computed as

$$P_2(z) = \mathrm{Prob}(|Z| > |z|)$$

Continuity Correction

PROC NPAR1WAY uses a continuity correction for the asymptotic two-sample Wilcoxon and Siegel-Tukey tests by default. You can remove the continuity correction by specifying the CORRECT=NO option. PROC NPAR1WAY incorporates the continuity correction when computing the standardized test statistic z by subtracting 0.5 from the numerator $(S - \mathrm{E}_0(S))$ if it is greater than 0. If the numerator is less than 0, PROC NPAR1WAY adds 0.5. Some sources recommend a continuity correction for nonparametric tests that use a continuous distribution to approximate a discrete distribution. For more information, see Sheskin (1997).

If you specify CORRECT=NO, PROC NPAR1WAY does not use a continuity correction for any test.

One-Way ANOVA Tests

PROC NPAR1WAY computes a one-way ANOVA test for each score type that you specify. Under the null hypothesis of no difference among class levels (samples), this test statistic has an asymptotic chi-square distribution with $r - 1$ degrees of freedom, where r is the number of class levels. For Wilcoxon scores, this test is known as the Kruskal-Wallis test.

Exact one-way ANOVA tests are also available for multisample data (where the data are classified into more than two levels). For two-sample data, exact simple linear rank tests are available. PROC NPAR1WAY computes exact tests for each score type that you specify in the EXACT statement. For more information, see the section "Exact Tests" on page 7399.

PROC NPAR1WAY computes the one-way ANOVA test statistic as

$$C = \left(\sum_{i=1}^{r} (T_i - \mathrm{E}_0(T_i))^2 \,/\, n_i \right) \,/\, S^2$$

where T_i is the total of scores for class level i, $\mathrm{E}_0(T_i)$ is the expected total for level i under the null hypothesis of no difference among levels, n_i is the number of observations in level i, and S^2 is the sample variance of the scores. The total of scores for class level i is given by

$$T_i = \sum_{j=1}^{n} c_{ij}\, a(R_j)$$

where $a(R_j)$ is the score for observation j, and c_{ij} indicates whether observation j is in level i. The expected total of scores for class level i under the null hypothesis is equal to

$$\mathrm{E}_0(T_i) = \frac{n_i}{n} \sum_{j=1}^{n} a(R_j)$$

The sample variance of the scores is computed as

$$S^2 = \frac{1}{(n-1)} \sum_{j=1}^{n} \left(a(R_j) - \bar{a} \right)^2$$

where $\bar{a}$ is the average score,

$$\bar{a} = \frac{1}{n} \sum_{j=1}^{n} a(R_j)$$

Scores for Linear Rank and One-Way ANOVA Tests

For each score type that you specify, PROC NPAR1WAY computes a one-way ANOVA statistic and also a linear rank statistic for two-sample data. The following score types are used primarily to test for differences in location: Wilcoxon, median, Van der Waerden (normal), and Savage. The following scores types are used to test for scale differences: Siegel-Tukey, Ansari-Bradley, Klotz, and Mood. Conover scores can be used to test for differences in both location and scale. This section gives formulas for the score types available in PROC NPAR1WAY. For further information about the formulas and the applicability of each score, see Randles and Wolfe (1979), Gibbons and Chakraborti (2010), Conover (1999), and Hollander and Wolfe (1999).

In addition to the score types described in this section, you can specify the SCORES=DATA option to use the input data observations as scores. This enables you to produce a wide variety of tests. You can construct any scores by using the DATA step, and then you can use PROC NPAR1WAY to compute the corresponding linear rank and one-way ANOVA tests for these scores. You can also analyze raw (unscored) data by using the SCORES=DATA option; for two-sample data, the corresponding exact test is a permutation test that is known as Pitman's test.

Wilcoxon Scores

Wilcoxon scores are the ranks of the observations, which can be written as

$$a(R_j) = R_j$$

where R_j is the rank of observation j, and $a(R_j)$ is the score of observation j.

Using Wilcoxon scores in the linear rank statistic for two-sample data produces the rank sum statistic of the Mann-Whitney-Wilcoxon test. Using Wilcoxon scores in the one-way ANOVA statistic produces the Kruskal-Wallis test. Wilcoxon scores are locally most powerful for location shifts of a logistic distribution.

When computing the asymptotic Wilcoxon two-sample test, PROC NPAR1WAY uses a continuity correction by default, as described in the section "Continuity Correction" on page 7388. If you specify the CORRECT=NO option in the PROC NPAR1WAY statement, the procedure does not use a continuity correction.

Median Scores

Median scores equal 1 for observations greater than the median, and 0 otherwise. In terms of the observation ranks, median scores are defined as

$$a(R_j) = \begin{cases} 1 & \text{if } R_j > (n+1)/2 \\ 0 & \text{if } R_j \le (n+1)/2 \end{cases}$$

Using median scores in the linear rank statistic for two-sample data produces the two-sample median test. Using median scores in the one-way ANOVA statistic for multisample data produces the Brown-Mood test. Median scores are particularly powerful for distributions that are symmetric and heavy-tailed.

Van der Waerden (Normal) Scores

Van der Waerden scores are the quantiles of a standard normal distribution and are also known as *quantile normal scores*. Van der Waerden scores are computed as

$$a(R_j) = \Phi^{-1}\left(\frac{R_j}{n+1}\right)$$

where Φ is the cumulative distribution function of a standard normal distribution. These scores are powerful for normal distributions.

Savage Scores

Savage scores are expected values of order statistics from the exponential distribution, with 1 subtracted to center the scores around 0. Savage scores are computed as

$$a(R_j) = \sum_{i=1}^{R_j} \left(\frac{1}{n-i+1} \right) - 1$$

Savage scores are powerful for comparing scale differences in exponential distributions or location shifts in extreme value distributions (Hajek 1969, p. 83).

Siegel-Tukey Scores

Siegel-Tukey scores are defined as

$$\begin{aligned} &a(1) = 1, \quad a(n) = 2, \qquad a(n-1) = 3, \quad a(2) = 4, \\ &a(3) = 5, \quad a(n-2) = 6, \quad a(n-3) = 7, \quad a(4) = 8, \ \ldots \end{aligned}$$

where the score values continue to increase in this pattern toward the middle ranks until all observations have been assigned a score.

When computing the asymptotic Siegel-Tukey two-sample test, PROC NPAR1WAY uses a continuity correction by default, as described in the section "Continuity Correction" on page 7388. If you specify the CORRECT=NO option in the PROC NPAR1WAY statement, the procedure does not use a continuity correction.

Ansari-Bradley Scores

Ansari-Bradley scores are similar to Siegel-Tukey scores, but Ansari-Bradley scoring assigns the same score value to corresponding extreme ranks. (Siegel-Tukey scores are a permutation of the ranks $1, 2, \ldots, n$.) Ansari-Bradley scores are defined as

$$\begin{aligned} &a(1) = 1, \quad a(n) = 1, \\ &a(2) = 2, \quad a(n-1) = 2, \ \ldots \end{aligned}$$

Equivalently, Ansari-Bradley scores are equal to

$$a(R_j) = \frac{n+1}{2} - \left| R_j - \frac{n+1}{2} \right|$$

Klotz Scores

Klotz scores are the squares of the Van der Waerden (normal) scores. Klotz scores are computed as

$$a(R_j) = \left(\Phi^{-1} \left(\frac{R_j}{n+1} \right) \right)^2$$

where Φ is the cumulative distribution function of a standard normal distribution.

Mood Scores

Mood scores are computed as the square of the difference between the observation rank and the average rank. Mood scores can be written as

$$a(R_j) = \left(R_j - \frac{n+1}{2}\right)^2$$

Conover Scores

Conover scores are based on the squared ranks of the absolute deviations from the sample means. For observation j the absolute deviation from the mean is computed as

$$U_j = |X_{j(i)} - \bar{X}_i|$$

where $X_{j(i)}$ is the value of observation j, observation j belongs to sample i, and $\bar{X}_i$ is the mean of sample i. The values of U_j are ranked, and the Conover score for observation j is computed as

$$a(U_j) = \big(\mathrm{Rank}(U_j)\big)^2$$

Following Conover (1999), when there are ties among the values of U_j, PROC NPAR1WAY assigns the average rank to each of the tied observations. To compute the average rank, PROC NPAR1WAY first ranks the U_j as if there were no ties and then averages the ranks of the tied observations.

The Conover score test is also known as the squared ranks test for variances. For more information, see Conover (1999).

Stratified Analysis

PROC NPAR1WAY provides stratified analysis of two-sample data for the following score types: Wilcoxon, median, Van der Waerden (normal), Savage, and data scores. The procedure computes the stratified test statistic by combining the stratum-level linear rank statistics for the score type that you specify. The section "Scores for Linear Rank and One-Way ANOVA Tests" on page 7389 describes the computation of the rank-based scores. For stratified analysis, the scores can be computed by using within-stratum ranks or overall ranks. By default, or if you specify the RANKS=STRATUM option, PROC NPAR1WAY ranks and scores the response values separately within each stratum. If you specify the RANKS=OVERALL option, PROC NPAR1WAY ranks and scores the response values overall (without regard to stratum classification). For more information, see Mehrotra, Lu, and Li (2010), Lehmann and D'Abrera (2006), and Van Elteren (1960).

The stratified rank sum statistic T is computed as

$$T = \sum_{k=1}^{K} w_k \, S_k$$

where S_k is the sum of the rank-based scores for stratum k (as described in the section "Simple Linear Rank Tests for Two-Sample Data" on page 7386), w_k is the weight of stratum k, and K is the total number of strata. By default, or if you specify the WEIGHTS=STRATUM option, the stratum weights are computed as $w_k = 1/(n_k+1)$, where n_k is the number of observations in stratum k. If you specify the WEIGHTS=EQUAL option, all stratum weights are 1 ($w_k = 1$ for all k).

The expected value and variance of T are similarly computed by combining stratum-level values as

$$\mathrm{E}_0(T) = \sum_{k=1}^{K} w_k \, \mathrm{E}_0(S_k)$$

$$\mathrm{Var}_0(T) = \sum_{k=1}^{K} w_k^2 \, \mathrm{Var}_0(S_k)$$

where $\mathrm{E}_0(S_k)$ and $\mathrm{Var}_0(S_k)$ are the expected value and variance of the rank sum statistic for stratum k. For more information, see the section "Simple Linear Rank Tests for Two-Sample Data" on page 7386.

The stratified test statistic is computed as

$$z = (T - \mathrm{E}_0(T)) \, / \sqrt{\mathrm{Var}_0(T)}$$

Under the null hypothesis of no difference between the two classification levels, the test statistic has a standard normal distribution. PROC NPAR1WAY provides one-sided and two-sided asymptotic p-values for the test as described in the section "Definition of p-Values" on page 7387.

PROC NPAR1WAY defines the reference class level (sample) for the stratified two-sample analysis by considering the number of observations (total frequency) across all strata. The reference group is defined as the smaller of the two samples. If both samples have the same number of observations, then by default, the reference level is defined as the level that appears first in the input data set; if you specify ORDER=FORMATTED, the reference level is defined as the first level in order of formatted value. Each stratum rank sum statistic is then computed as the sum of the scores for the observations in the reference group. PROC NPAR1WAY lists the reference group first in the "Class Information" table.

Hodges-Lehmann Estimation of Location Shift

If you specify the HL option, PROC NPAR1WAY computes the Hodges-Lehmann estimate of location shift for two-sample data. This option also provides asymptotic confidence limits for the location shift (which are sometimes known as Moses confidence limits). You can specify the confidence level in the ALPHA= option in the PROC NPAR1WAY statement. By default, ALPHA=0.05, which produces 95% confidence limits. Additionally, you can request exact confidence limits for the location shift by specifying the HL option in the EXACT statement.

The Hodges-Lehmann estimator of location shift is associated with the Wilcoxon linear rank statistic. For more information, see Hollander and Wolfe (1999) and Hodges and Lehmann (1983).

PROC NPAR1WAY computes the Hodges-Lehmann estimate $\hat{\Delta}$ as the median of all paired differences between observations in the two samples (classes), which can be written as

$$\hat{\Delta} = \mathrm{median}\left((Y_j - X_i) \quad \text{where } \; j = 1, 2, \ldots, n_1; \; i = 1, 2, \ldots, n_2 \right)$$

The Y_j are observations in class 1, the X_i are observations in class 2, and n_1 and n_2 denote the number of observations in class 1 and class 2, respectively.

By default, PROC NPAR1WAY uses the larger of the two classes as the reference class X (class 2 in the Hodges-Lehmann difference). If both classes have the same number of observations, PROC NPAR1WAY uses the class that appears second in the "Wilcoxon Scores" table as the reference class. (By default, the table displays class levels in the order in which they appear in the input data set. If you specify the ORDER=FORMATTED option, the table displays class levels in order of their formatted value.)

You can specify the reference class by using the HL(REFCLASS=) option. REFCLASS=1 specifies the first class that is listed in the "Wilcoxon Scores" table, and REFCLASS=2 specifies the second class. REFCLASS='*class-value*' identifies the reference class by the formatted value of the CLASS variable.

Let m denote the total number of differences $(n_1 \times n_2)$, and let $U^{(k)}$ denote the kth value of $(Y_j - X_i)$ among the ordered differences. When m is an odd number, the median difference is the value that has rank $(m+1)/2$,

$$\hat{\Delta} = U^{(k)} \quad \text{where } k = (m+1)/2$$

When m is an even number, the median difference is the average of the values that have ranks $(m/2)$ and $((m/2)+1)$,

$$\hat{\Delta} = \left(U^{(k)} + U^{(k+1)}\right)/2 \quad \text{where } k = m/2$$

Following Hollander and Wolfe (1999), the asymptotic lower and upper confidence limits for the location shift are

$$\left(\Delta_L = U^{(C_\alpha)},\ \ \Delta_U = U^{(m+1-C_\alpha)}\right)$$

where C_α is the largest integer less than or equal to $C_\alpha{}^*$, which is computed as

$$C_\alpha{}^* = \mathrm{E}_0(S) - z_{\alpha/2}\sqrt{\mathrm{Var}_0(S)}$$

where $\mathrm{E}_0(S)$ and $\mathrm{Var}_0(S)$ are the expected value and variance, respectively, of the Wilcoxon statistic S under the null hypothesis (as described in the section "Simple Linear Rank Tests for Two-Sample Data" on page 7386), and $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. For Wilcoxon rank scores,

$$\mathrm{E}_0(S) = n_1 n_2/2$$

When there are no tied values, $\mathrm{Var}_0(S)$ for Wilcoxon scores equals

$$\mathrm{Var}_0(S) = n_1 n_2 (n_1 + n_2 + 1)/12$$

PROC NPAR1WAY displays the midpoint of the confidence interval (Δ_L, Δ_U), which can also be used as an estimate of location shift. For more information, see Lehmann (1963). Additionally, PROC NPAR1WAY provides an estimate of the asymptotic standard error of $\hat{\Delta}$ based on the length of the confidence interval, which is computed as

$$\mathrm{se}(\hat{\Delta}) = (\Delta_U - \Delta_L)\,/\,2\,z_{\alpha/2}$$

Exact Confidence Limits

If you specify the HL option in the EXACT statement, PROC NPAR1WAY computes exact confidence limits for the location shift between the two samples. You can specify the level of the confidence limits in the ALPHA= option in the PROC NPAR1WAY statement. By default, ALPHA=0.05, which produces 95% confidence limits.

PROC NPAR1WAY computes exact confidence limits for the location shift as described in Randles and Wolfe (1979, p. 180). PROC NPAR1WAY first generates the exact conditional distribution of the Mann-Whitney

U statistic, which is the number of pairwise differences $(Y_j - X_i)$ that are positive plus half the number of pairwise differences that are 0. The Mann-Whitney statistic is defined as

$$M = \sum_{j=1}^{n_1} \sum_{i=1}^{n_2} \phi\left(Y_j, X_i\right)$$

where

$$\phi(Y_j, X_i) = \begin{cases} 1 & \text{if } Y_j > X_i \\ 1/2 & \text{if } Y_j = X_i \\ 0 & \text{otherwise} \end{cases}$$

From the exact conditional distribution of the Mann-Whitney statistic M, PROC NPAR1WAY chooses $C^*_{L,\alpha}$ as the smallest value such that $\mathrm{Prob}(M \geq C^*_{L,\alpha}) \leq \alpha/2$. Rounding $C^*_{L,\alpha}$ up to the nearest integer $C_{L,\alpha}$, the lower confidence limit is the difference $(Y_i - X_j)$ that has a rank of $(n_1 n_2 - C_{L,\alpha} + 1)$.

To find the upper confidence limit, PROC NPAR1WAY chooses $C^*_{U,\alpha}$ as the largest Mann-Whitney value such that $\mathrm{Prob}(M \leq C^*_{U,\alpha}) \leq \alpha/2$. Rounding $C^*_{U,\alpha}$ down to the nearest integer $C_{U,\alpha}$, the upper confidence limit is the difference $(Y_i - X_j)$ that has a rank of $(n_1 n_2 - C_{U,\alpha})$.

Because this is a discrete problem, the confidence coefficient is not exactly $(1 - \alpha)$ but is at least $(1 - \alpha)$; thus, these confidence limits are conservative.

Fligner-Policello Test

If you specify the FP option, PROC NPAR1WAY computes the Fligner-Policello location test for two-sample data (Fligner and Policello 1981). The null hypothesis for the test is H_0: $\theta_x = \theta_y$, where θ_x and θ_y are the population medians of the two classes. The Fligner-Policello test assumes that the distribution in each class is symmetric around the class median, but it does not require that the two class distributions have the same form or that the class variances be equal. For more information, see Hollander and Wolfe (1999) and Juneau (2007).

The Fligner-Policello test is based on placement scores (Orban and Wolfe 1979). The placement of an observation X_i from class X, $P(X_i)$, is defined as the number of observations in class Y that are less than X_i. If there are ties, the placement of X_i is adjusted by adding half the number of observations in class Y that are equal to X_i. The placement of an observation Y_j from class Y, $P(Y_j)$, is defined in the same way. The placements can be expressed as

$$\begin{aligned} P(X_i) &= \sum_{j=1}^{n_y} \left(\mathrm{I}(Y_j < X_i) + 0.5\, \mathrm{I}(Y_j = X_i) \right) \\ P(Y_j) &= \sum_{i=1}^{n_x} \left(\mathrm{I}(X_i < Y_j) + 0.5\, \mathrm{I}(X_i = Y_j) \right) \end{aligned}$$

where $\mathrm{I}(\cdot)$ is an indicator function and n_x and n_y denote the number of observations in class X and class Y, respectively.

The average placements for class X and class Y are computed as

$$\bar{P}_x = \left(\sum_{i=1}^{n_x} P(X_i) \right) / n_x$$

$$\bar{P}_y = \left(\sum_{j=1}^{n_y} P(Y_j) \right) / n_y$$

The Fligner-Policello test statistic is computed as

$$z = \left(\sum_{j=1}^{n_y} P(Y_j) - \sum_{i=1}^{n_x} P(X_i) \right) / \left(2\sqrt{V_x + V_y + \bar{P}_x \bar{P}_y} \right)$$

where

$$V_x = \sum_{i=1}^{n_x} (P(X_i) - \bar{P}_x)^2$$

$$V_y = \sum_{j=1}^{n_y} (P(Y_j) - \bar{P}_y)^2$$

and the standard deviation of the placements is $\sqrt{V_x/(n_x - 1)}$ for class X and $\sqrt{V_y/(n_y - 1)}$ for class Y.

Under the null hypothesis, the Fligner-Policello statistic has an asymptotic standard normal distribution. PROC NPAR1WAY provides one- and two-sided asymptotic p-values for the Fligner-Policello test. For the one-sided test, PROC NPAR1WAY displays the right-sided p-value when the test statistic z is greater than its null hypothesis expected value of 0. PROC NPAR1WAY displays the left-sided p-value when the test statistic z is less than or equal to 0. The one-sided p-value $P_1(z)$ can be expressed as

$$P_1(z) = \begin{cases} \text{Prob}(Z > z) & \text{if } z > 0 \\ \text{Prob}(Z < z) & \text{if } z \leq 0 \end{cases}$$

where Z has a standard normal distribution. The two-sided p-value $P_2(z)$ is computed as Prob($|Z| > |z|$).

When you specify the FP option, PROC NPAR1WAY displays a "Fligner-Policello Placements" table and a "Fligner-Policello Test" table. The "Fligner-Policello Placements" table contains the following information for each of the two classes: number of observations, sum of the placements, average placement, and standard deviation of the placements. The "Fligner-Policello Test" table contains the test statistic z and the corresponding one- and two-sided p-values. This table also displays the difference between the class placement sums, which is the numerator of the test statistic. When ODS Graphics is enabled and you specify the FP or PLOTS=FPBOXPLOT option, PROC NPAR1WAY provides a box plot of the Fligner-Policello placements.

PROC NPAR1WAY computes the Fligner-Policello difference (the numerator of the test statistic) as the placement sum for class Y minus the placement sum for class X (the reference class). By default, PROC NPAR1WAY uses the larger of the two classes as the reference class X. If both classes have the same number of observations, PROC NPAR1WAY uses the class that appears second in the "Fligner-Policello Placements"

table as the reference class. (By default, the table displays class levels in the order in which they appear in the input data set. If you specify the ORDER=FORMATTED option, the table displays class levels in order of their formatted value.)

You can specify the reference class by using the FP(REFCLASS=) option. REFCLASS=1 specifies the first class that is listed in the "Fligner-Policello Placements" table, and REFCLASS=2 specifies the second class. REFCLASS='*class-value*' identifies the reference class by the formatted value of the CLASS variable.

Ratio Mean Deviations (RMD) Exact Test

If you specify the RMD option in the PROC NPAR1WAY statement or the EXACT statement, PROC NPAR1WAY computes the ratio mean deviations (RMD) exact test of scale differences for two-sample data (Higgins 2004). This test is based on the ratio of absolute deviations of the observations (response variable values) from their class medians. For more information, see Butar and Park (2008) and Richter and McCann (2017).

The RMD test statistic is computed as

$$\mathrm{RMD} = \left(\sum_{j=1}^{n_1} d_{1j}/n_1 \right) / \left(\sum_{j=1}^{n_2} d_{2j}/n_2 \right)$$

where the absolute deviations for class i are

$$d_{ij} = |X_{ij} - m_i|$$

and where m_i is the sample median of class i, X_{ij} are the observations in class i ($j = 1, \ldots, n_i$), and n_i is the number of observations in class i ($i = 1, 2$).

PROC NPAR1WAY uses the first class that is displayed in the "Deviations" table as class 1 and the second class as class 2. By default, the table displays class levels in the order in which they appear in the input data set. If you specify the ORDER=FORMATTED option, the table displays class levels in order of their formatted value.

The RMD p-value is computed by using an exact permutation test that is conditional on the observed deviations (Higgins 2004). PROC NPAR1WAY displays the exact one-sided p-value as the left-sided p-value (Pr <= RMD) when the value of the RMD statistic is less than or equal to 1; otherwise, PROC NPAR1WAY displays the right-sided p-value (Pr >= RMD).

If you specify the MIDP *computation-option* in the EXACT statement, PROC NPAR1WAY also provides the exact mid p-value for the RMD test. If you specify the POINT *computation-option* in the EXACT statement, PROC NPAR1WAY provides the exact point probability for the RMD test.

PROC NPAR1WAY computes the modified RMD two-sided statistic as

$$\mathrm{RMD2} = \max(\ \mathrm{RMD},\ 1/\mathrm{RMD}\ )$$

The corresponding exact two-sided p-value is computed by using a permutation test for the RMD2 statistic.

To obtain Monte Carlo estimates of exact p-values for the RMD test, you can specify the MC *computation-option* together with the RMD *statistic-option* in the EXACT statement. For more information, see the section "Exact Tests" on page 7399.

Multiple Comparisons Based on Pairwise Rankings

If you specify the DSCF option, PROC NPAR1WAY computes the Dwass, Steel, Critchlow-Fligner (DSCF) multiple comparison analysis, which is based on pairwise two-sample Wilcoxon comparisons (Dwass 1960; Steel 1960; Critchlow and Fligner 1991). The DSCF analysis is available when the number of CLASS variable levels (samples) is greater than 2. There are $r(r-1)/2$ pairs of samples, where r is the total number of samples.

For each pair of samples, PROC NPAR1WAY ranks the observations and computes the standardized Wilcoxon test statistic, which is described in the sections "Simple Linear Rank Tests for Two-Sample Data" on page 7386 and "Wilcoxon Scores" on page 7389. The procedure does not include a continuity correction in the Wilcoxon test statistics that it computes for the DSCF analysis. The DSCF statistic for a pair of samples is computed as $\sqrt{2}\,z$, where z is the two-sample standardized Wilcoxon statistic. Under the null hypothesis of no location differences among the r samples, the distribution of the DSCF statistics can be approximated by the studentized range distribution for r independent standard normal variables. The p-value for a two-sample DSCF comparison is the percentile of the studentized range distribution that corresponds to the value of the DSCF statistic. For more information, see Hollander and Wolfe (1999), Juneau (2007), and Juneau (2004).

When you specify the DSCF option, PROC NPAR1WAY displays the "Pairwise Two-Sided Multiple Comparison Analysis" table, which contains the following information for each two-sample comparison: comparison identification (two CLASS variable levels), two-sample Wilcoxon test statistic, DSCF statistic, and two-sided DSCF p-value. You can store these statistics in a SAS data set by using the Output Delivery System (ODS). The ODS name for this table is DSCF. For more information, see the section "ODS Table Names" on page 7417 and Chapter 23, "Using the Output Delivery System."

Empirical Distribution Function Tests

If you specify the EDF option, PROC NPAR1WAY computes tests based on the empirical distribution function. These include the Kolmogorov-Smirnov and Cramér–von Mises tests, and also the Kuiper test for two-sample data. This section gives formulas for these test statistics. For further information about the formulas and the interpretation of EDF statistics, see Hollander and Wolfe (1999) and Gibbons and Chakraborti (2010). For information about the k-sample analogs of the Kolmogorov-Smirnov and Cramér–von Mises statistics, see Kiefer (1959).

The *empirical distribution function* (EDF) of a sample $\{x_j\}$, $j = 1, 2, \ldots, n$, is defined as

$$F(x) = \frac{1}{n}(\text{number of } x_j \le x) = \frac{1}{n}\sum_{j=1}^{n} I(x_j \le x)$$

where $I(\cdot)$ is an indicator function. PROC NPAR1WAY uses the subsample of values within the ith class level to generate an EDF for the class, F_i. The EDF for the overall sample, pooled over classes, can also be expressed as

$$F(x) = \frac{1}{n}\sum_{i} n_i\, F_i(x)$$

where n_i is the number of observations in the ith class level, and n is the total number of observations.

Kolmogorov-Smirnov Test

The Kolmogorov-Smirnov statistic measures the maximum deviation of the EDF within the classes from the pooled EDF. PROC NPAR1WAY computes the Kolmogorov-Smirnov statistic as

$$\mathrm{KS} = \max_j \sqrt{\frac{1}{n} \sum_i n_i \left(F_i(x_j) - F(x_j)\right)^2} \quad \text{where } j = 1, 2, \ldots, n$$

The asymptotic Kolmogorov-Smirnov statistic is computed as

$$\mathrm{KS}_a = \mathrm{KS} \times \sqrt{n}$$

For each class level i and overall, PROC NPAR1WAY displays the value of F_i at the maximum deviation from F and the value $\sqrt{n_i}\,(F_i - F)$ at the maximum deviation from F. PROC NPAR1WAY also gives the observation where the maximum deviation occurs.

If there are only two class levels, PROC NPAR1WAY computes the two-sample Kolmogorov-Smirnov test statistic D as

$$D = \max_j \left|F_1(x_j) - F_2(x_j)\right| \quad \text{where } j = 1, 2, \ldots, n$$

The p-value for this test is the probability that D is greater than the observed value d under the null hypothesis of no difference between class levels (samples). PROC NPAR1WAY computes the asymptotic p-value for D by using the approximation

$$\mathrm{Prob}(D > d) = 2 \sum_{i=1}^{\infty} (-1)^{(i-1)}\, e^{(-2i^2 z^2)}$$

where

$$z = d\sqrt{n_1 n_2 / n}$$

For more information, see Hodges (1957).

If you specify the D option, or if you request exact Kolmogorov-Smirnov p-values by specifying the KS option in the EXACT statement, PROC NPAR1WAY also computes the one-sided Kolmogorov-Smirnov statistics $D+$ and $D-$ for two-sample data as

$$D+ = \max_j \left(F_1(x_j) - F_2(x_j)\right) \quad \text{where } j = 1, 2, \ldots, n$$

$$D- = \max_j \left(F_2(x_j) - F_1(x_j)\right) \quad \text{where } j = 1, 2, \ldots, n$$

The asymptotic probability that $D+$ is greater than the observed value d^+, under the null hypothesis of no difference between the two class levels, is computed as

$$\mathrm{Prob}(D+ > d^+) = e^{-2z^2} \quad \text{where } z = d^+\sqrt{n_1 n_2 / n}$$

Similarly, the asymptotic probability that $D-$ is greater than the observed value d^- is computed as

$$\mathrm{Prob}(D- > d^-) = e^{-2z^2} \quad \text{where } z = d^-\sqrt{n_1 n_2 / n}$$

To request exact p-values for the Kolmogorov-Smirnov statistics, you can specify the KS option in the EXACT statement. For more information, see the section "Exact Tests" on page 7399.

Cramér–von Mises Test

The Cramér–von Mises statistic is defined as

$$\mathrm{CM} = \frac{1}{n^2} \sum_i \left(n_i \sum_{j=1}^{p} t_j \left(F_i(x_j) - F(x_j) \right)^2 \right)$$

where t_j is the number of ties at the jth distinct value and p is the number of distinct values. The asymptotic value is computed as

$$\mathrm{CM}_a = \mathrm{CM} \times n$$

PROC NPAR1WAY displays the contribution of each class level to the sum CM_a.

Kuiper Test

For data with two class levels, PROC NPAR1WAY computes the Kuiper statistic, its scaled value for the asymptotic distribution, and the asymptotic p-value. The Kuiper statistic is computed as

$$K = \max_j \left(F_1(x_j) - F_2(x_j) \right) - \min_j \left(F_1(x_j) - F_2(x_j) \right) \quad \text{where } j = 1, 2, \ldots, n$$

The asymptotic value is

$$K_a = K \sqrt{n_1 n_2 / n}$$

PROC NPAR1WAY displays the value of $(\max_j \left| F_1(x_j) - F_2(x_j) \right|)$ for each class level.

The p-value for the Kuiper test is the probability of observing a larger value of K_a under the null hypothesis of no difference between the two classes. PROC NPAR1WAY computes this p-value according to Owen (1962, p. 441).

Exact Tests

PROC NPAR1WAY provides exact p-values for the following nonparametric location and scale tests: Wilcoxon, median, van der Waerden (normal), Savage, Siegel-Tukey, Ansari-Bradley, Klotz, Mood, and Conover. The procedure also provides exact p-values for tests that use the raw input data as scores. These exact tests are available for two-sample and multisample data. The two-sample exact tests are based on simple linear rank statistics, and the multisample exact tests are based on one-way ANOVA statistics.

For two-sample data, PROC NPAR1WAY also provides the exact Kolmogorov-Smirnov test, the exact ratio mean deviations (RMD) test, and exact Hodges-Lehmann confidence limits for the location shift.

Exact tests can be useful in situations where the asymptotic assumptions are not met and therefore the asymptotic p-values might not be close approximations for the true p-values. Standard asymptotic methods involve the assumption that the test statistic follows a particular distribution when the sample size is sufficiently large. When the sample size is not large, asymptotic results might not be valid. Asymptotic results might also be unreliable when the distribution of the data is sparse, skewed, or heavily tied. For more information, see Agresti (2007) and Bishop, Fienberg, and Holland (1975). Exact computations are based on the statistical theory of exact conditional inference for contingency tables, which is reviewed by Agresti (1992).

In addition to the computation of exact *p*-values, PROC NPAR1WAY provides the option to estimate exact *p*-values by Monte Carlo simulation. This can be useful for large problems where exact computations require a substantial amount of time and memory but asymptotic approximations might not be sufficient.

The following sections summarize the exact computational algorithms, define the exact *p*-values that PROC NPAR1WAY computes, discuss the computational resource requirements, and describe the Monte Carlo estimation option.

Computational Algorithms

PROC NPAR1WAY computes exact *p*-values for nonparametric location and scale tests by using the network algorithm, which was developed by Mehta and Patel (1983). This algorithm provides a substantial advantage over direct enumeration, which can be very time-consuming and feasible only for small problems. See Agresti (1992) for a review of algorithms for computation of exact *p*-values, and see Mehta, Patel, and Tsiatis (1984) and Mehta, Patel, and Senchaudhuri (1991) for information about the performance of the network algorithm.

To implement the network algorithm, PROC NPAR1WAY constructs a contingency table from the input data; the contingency table rows correspond to the levels of the classification variable and the table columns correspond to the levels of the response variable. The reference set for the observed contingency table includes all tables that have the same marginal row and column sums as the observed table. Corresponding to this reference set, the network algorithm forms a directed acyclic network that consists of nodes in a number of stages. A path through the network corresponds to a distinct table in the reference set. The distances between nodes are defined so that the total distance of a path through the network is the corresponding value of the test statistic. At each node, the algorithm computes the shortest and longest path distances for all paths that pass through that node. For the two-sample linear rank statistics, which can be expressed as linear combinations of cell frequencies multiplied by increasing row and column scores, PROC NPAR1WAY computes shortest and longest path distances by using the algorithm of Agresti, Mehta, and Patel (1990). For the multisample one-way test statistics, PROC NPAR1WAY computes an upper bound for the longest path and a lower bound for the shortest path by following the approach of Valz and Thompson (1994).

The longest and shortest path distances (bounds) for a node are compared to the value of the observed test statistic to determine whether all paths through the node contribute to the *p*-value, no paths through the node contribute to the *p*-value, or neither of these situations occurs. If all paths through the node contribute, the *p*-value is incremented accordingly, and these paths are eliminated from further analysis. If no paths contribute, these paths are eliminated from further analysis. Otherwise, the algorithm continues to process this node and the associated paths. The algorithm finishes when all nodes have been accounted for.

PROC NPAR1WAY performs the network algorithm by using full numerical precision to represent all statistics, row and column scores, and other quantities in the computations. Although it is possible to use rounding to improve the speed and memory requirements of the algorithm, PROC NPAR1WAY does not use rounding because it might reduce the accuracy of the results.

Definition of *p*-Values

For two-sample linear rank tests, PROC NPAR1WAY computes exact one-sided and two-sided *p*-values for each test that you specify in the EXACT statement. For one-sided tests, PROC NPAR1WAY displays the right-sided *p*-value when the observed value of the test statistic is greater than its expected value. The right-sided *p*-value is the sum of probabilities for those tables with a test statistic that is greater than or equal to the observed test statistic. Otherwise, when the observed test statistic is less than or equal to its expected value, PROC NPAR1WAY displays the left-sided *p*-value. The left-sided *p*-value is the sum of probabilities

for those tables with a test statistic that is less than or equal to the observed value. The one-sided p-value P_1 can be expressed as

$$P_1(t) = \begin{cases} \text{Prob(Test Statistic} \geq t \text{)} & \text{if } t > \mathrm{E}_0(T) \\ \text{Prob(Test Statistic} \leq t \text{)} & \text{if } t \leq \mathrm{E}_0(T) \end{cases}$$

where t is the observed value of the test statistic and $\mathrm{E}_0(T)$ is the expected value of the test statistic under the null hypothesis. PROC NPAR1WAY computes the two-sided p-value as the sum of the one-sided p-value and the corresponding area in the opposite tail of the distribution of the statistic, equidistant from the expected value. The two-sided p-value P_2 can be expressed as

$$P_2(t) = \mathrm{Prob}\left(|\text{Test Statistic} - \mathrm{E}_0(T)| \geq |t - \mathrm{E}_0(T)|\right)$$

Tests for multisample data are based on one-way ANOVA statistics. For a test of this form, large values of the test statistic indicate a departure from the null hypothesis; the test is inherently two-sided. The exact p-value is the sum of probabilities for those tables having a test statistic greater than or equal to the value of the observed test statistic.

If you specify the POINT option in the EXACT statement, PROC NPAR1WAY provides point probabilities for the exact tests. The point probability is the exact probability that the test statistic equals the observed value. For two-sample data, PROC NPAR1WAY provides point probabilities for the one-sided tests of the linear rank statistics. For multisample data, PROC NPAR1WAY provides point probabilities for the one-way ANOVA tests.

If you specify the MIDP option in the EXACT statement, PROC NPAR1WAY provides exact mid p-values. The exact mid p-value is defined as the exact p-value minus half the exact point probability, which equals the average of $\mathrm{Prob}(\text{Test Statistic} \geq t)$ and $\mathrm{Prob}(\text{Test Statistic} > t)$ for a right-sided test. The exact mid p-value is smaller and less conservative than the nonadjusted exact p-value. For more information, see Agresti (2013, section 1.1.4) and Hirji (2006, sections 2.5 and 2.11.1). For two-sample data, PROC NPAR1WAY provides mid p-values for the one-sided tests of the linear rank statistics. For multisample data, PROC NPAR1WAY provides mid p-values for the one-way ANOVA tests.

Computational Resources

PROC NPAR1WAY uses relatively fast and efficient algorithms for exact computations. These algorithms, together with improvements in computing power, make it feasible to perform exact computations for data where previously only asymptotic methods could be applied. Nevertheless, depending on your available computing resources, exact computations for some very large problems might require a prohibitive amount of time and memory. For such large problems, consider whether exact methods are really needed or whether asymptotic methods might give results that are very close to the exact results while requiring much less computing time and memory. When asymptotic methods might not be sufficient for such large problems, consider using Monte Carlo estimation of exact p-values, which is described in the section "Monte Carlo Estimation" on page 7402.

There is no formula that can predict in advance how much time and memory are needed to compute an exact p-value for a specific data set and test. The time and memory requirements depend on several factors, which include the following: the total number of observations, the number of rows (class levels) and columns (response levels), the particular arrangement of the observations, and the test to be performed. Generally, larger problems (in terms of total sample size, number of rows, and number of columns) tend to require more time and memory. For a fixed total sample size, time and memory requirements tend to increase as

the number of rows and columns increase because this corresponds to an increase in the number of tables in the reference set. For a fixed total sample size, time and memory requirements also tend to increase as the marginal row and column sums become more homogeneous. For more information, see Agresti, Mehta, and Patel (1990) and Gail and Mantel (1977).

While PROC NPAR1WAY is computing an exact *p*-value, you can terminate the computation by pressing the system interrupt key sequence and choosing to stop computations. For more information, see the *SAS Companion* for your system. After you terminate an exact computation, PROC NPAR1WAY completes all other remaining tasks. The procedure reports missing values for any exact *p*-values that were not computed before termination.

To limit the amount of time that PROC NPAR1WAY uses for exact computations, you can specify the MAXTIME= option in the EXACT statement. This option sets the maximum amount of clock time (in seconds) that PROC NPAR1WAY can use to compute an exact *p*-value. If PROC NPAR1WAY does not finish an exact computation in the time that you specify, the procedure terminates the computation and completes the remaining tasks.

Monte Carlo Estimation

When you specify the MC option in the EXACT statement, PROC NPAR1WAY computes Monte Carlo estimates of exact *p*-values. Monte Carlo estimation can be useful for large problems where exact computations require a substantial amount of time and memory but asymptotic approximations might not be sufficient.

To describe the precision of a Monte Carlo estimate, PROC NPAR1WAY provides the asymptotic standard error and $100(1-\alpha)\%$ confidence limits. You can specify the confidence level α in the ALPHA= option in the EXACT statement; by default, ALPHA=0.01, which produces 99% confidence limits.

You can specify the number of Monte Carlo samples by using the N=*n* option in the EXACT statement. By default, PROC NPAR1WAY uses 10,000 samples to compute a Monte Carlo estimate. To improve the precision of the Monte Carlo estimates, you can specify a larger value of *n*; this increases the computation time because more samples are generated. To reduce the computation time, you can specify a smaller value of *n*.

PROC NPAR1WAY computes a Monte Carlo estimate of an exact *p*-value by generating a random sample of tables from the reference set for the exact test. The reference set is based on the contingency table of the input data; the contingency table rows correspond to the levels of the classification variable and the table columns correspond to the levels of the response variable. The reference set includes all tables that have the same marginal row and column sums as the observed table.

PROC NPAR1WAY generates a random sample of tables from the reference set by using the algorithm of Agresti, Wackerly, and Boyett (1979), which generates tables in proportion to their hypergeometric probabilities conditional on the marginal frequencies. For each sample table, PROC NPAR1WAY computes the value of the test statistic and compares it to the value of the observed test statistic. To estimate a right-sided *p*-value, PROC NPAR1WAY counts all sample tables for which the test statistic is greater than or equal to the observed test statistic. The estimate of the *p*-value is the number of these tables divided by the total number of sample tables, which can be expressed as

$$\begin{aligned}
\hat{p}_{\mathrm{mc}} &= m \,/\, n \\
m &= \text{number of samples for which (test statistic} \geq t_o) \\
n &= \text{total number of samples} \\
t_o &= \text{observed test statistic}
\end{aligned}$$

PROC NPAR1WAY computes estimates of left-sided and two-sided exact p-values similarly. For left-sided exact p-values, PROC NPAR1WAY evaluates whether the sample test statistics are less than or equal to the observed test statistic. For two-sided exact p-values, PROC NPAR1WAY compares sample test statistics to the observed test statistic by using the definition of the two-sided p-value (P_2) for the test. For more information, see the section "Definition of p-Values" on page 7400 and descriptions of the individual tests.

The variable m has a binomial distribution with n trials and success probability p. The asymptotic standard error of the Monte Carlo estimate is

$$\mathrm{se}(\hat{p}_{\mathrm{mc}}) = \sqrt{\hat{p}_{\mathrm{mc}}\,(1-\hat{p}_{\mathrm{mc}})\,/\,(n-1)}$$

PROC NPAR1WAY constructs asymptotic confidence limits for the exact p-value as

$$\hat{p}_{\mathrm{mc}} \;\pm\; \left(z_{\alpha/2} \times \mathrm{se}(\hat{p}_{\mathrm{mc}})\right)$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution, and the confidence level α is determined by the ALPHA= option in the EXACT statement.

When the Monte Carlo estimate $\hat{p}_{\mathrm{mc}}$ is 0, PROC NPAR1WAY computes confidence limits for the p-value as

$$\left(0,\;\; 1-\alpha^{(1/n)}\right)$$

When the Monte Carlo estimate $\hat{p}_{\mathrm{mc}}$ is 1, PROC NPAR1WAY computes confidence limits for the p-value as

$$\left(\alpha^{(1/n)},\;\; 1\right)$$

Contents of the Output Data Set

The OUTPUT statement creates a SAS data set that contains statistics that PROC NPAR1WAY computes. You identify which statistics to store in the output data set by specifying *output-options* in the OUTPUT statement. For more information, see the description of the OUTPUT statement.

The output data set contains one observation for each analysis variable for each BY group. (You can specify the analysis variables in the VAR statement.) The OUTPUT data set includes the following variables:

- BY variables, if you use a BY statement
- _VAR_, which identifies the analysis variable
- variables that contain the statistics

When you specify an *output-option* in the OUTPUT statement, the output data set contains the test statistic and associated values from the analysis that you specify. The associated values might include standardized statistics, one- and two-sided p-values, exact p-values, degrees of freedom, and confidence limits. If you request an exact test for the specified analysis by using the EXACT statement, the output data set includes the exact p-values. The statistics that are included also depend on the classification of the data. Some statistics are available only for two-sample data (where the CLASS variable groups the data into two classes); other statistics are available only for multisample data.

Table 90.5 lists variable names and descriptions or the statistics that are available in the output data set.

Monte Carlo estimates of exact *p*-values, multiple comparison statistics, and stratified analysis statistics are not available in this output data set. You can store any table that PROC NPAR1WAY produces in a SAS data set by using the Output Delivery System (ODS). For more information, see the section "ODS Table Names" on page 7417 and Chapter 23, "Using the Output Delivery System."

Table 90.5 Output Data Set Variable Names and Descriptions

Output Option	Output Variables		Variable Descriptions
AB	_AB_	*	Two-sample Ansari-Bradley statistic
	Z_AB	*	Ansari-Bradley statistic, standardized
	PL_AB	*	*p*-value (left-sided), Ansari-Bradley test
	PR_AB	*	*p*-value (right-sided), Ansari-Bradley test
	P2_AB	*	*p*-value (two-sided), Ansari-Bradley test
	XPL_AB	*	Exact *p*-value (left-sided), Ansari-Bradley test
	XPR_AB	*	Exact *p*-value (right-sided), Ansari-Bradley test
	XPT_AB	*	Exact point probability, Ansari-Bradley test
	XMP_AB	*	Exact mid *p*-value, Ansari-Bradley test
	XP2_AB	*	Exact *p*-value (two-sided), Ansari-Bradley test
	CHAB		Ansari-Bradley chi-square
	DF_CHAB		Degrees of freedom, Ansari-Bradley chi-square
	P_CHAB		*p*-value, Ansari-Bradley chi-square test
	XP_CHAB	**	Exact *p*-value, Ansari-Bradley chi-square test
	XPT_CHAB	**	Exact point probability, Ansari-Bradley chi-square test
	XMP_CHAB	**	Exact mid *p*-value, Ansari-Bradley chi-square test
ANOVA	_MSA_		ANOVA effect mean square, among MS
	MSE		ANOVA error mean square, within MS
	_F_		*F* statistic for ANOVA
	P_F		*p*-value, *F* statistic for ANOVA
CONOVER	_CON_	*	Two-sample Conover statistic
	Z_CON	*	Conover statistic, standardized
	PL_CON	*	*p*-value (left-sided), Conover test
	PR_CON	*	*p*-value (right-sided), Conover test
	P2_CON	*	*p*-value (two-sided), Conover test
	XPL_CON	*	Exact *p*-value (left-sided), Conover test
	XPR_CON	*	Exact *p*-value (right-sided), Conover test
	XPT_CON	*	Exact point probability, Conover test
	XMP_CON	*	Exact mid *p*-value, Conover test
	XP2_CON	*	Exact *p*-value (two-sided), Conover test
	CHCON		Conover chi-square
	DF_CHCON		Degrees of freedom, Conover chi-square
	P_CHCON		*p*-value, Conover chi-square test
	XP_CHCON	**	Exact *p*-value, Conover chi-square test
	XPT_CHCO	**	Exact point probability, Conover chi-square test
	XMP_CHCON	**	Exact mid *p*-value, Conover chi-square test

Table 90.5 *continued*

Output Option	Output Variables		Variable Descriptions
EDF	_KS_		Kolmogorov-Smirnov statistic
	KSA		Kolmogorov-Smirnov statistic (asymptotic)
	_Dp_	*	Two-sample Kolmogorov-Smirnov *D*+
	P_Dp	*	*p*-value, *D*+
	_Dm_	*	Two-sample Kolmogorov-Smirnov *D*–
	P_Dm	*	*p*-value, *D*–
	_D_	*	Two-sample Kolmogorov-Smirnov statistic *D*
	P_KSA	*	*p*-value, *D*
	XP_Dp	*	Exact *p*-value, *D*+
	XPT_Dp	*	Exact point probability, *D*+
	XMP_Dp	*	Exact mid *p*-value, *D*+
	XP_Dm	*	Exact *p*-value, *D*–
	XPT_Dm	*	Exact point probability, *D*–
	XMP_Dm	*	Exact mid *p*-value, *D*–
	XP_D	*	Exact *p*-value, *D*
	XPT_D	*	Exact point probability, *D*
	XMP_D	*	Exact mid *p*-value, *D*
	_CM_		Cramér–von Mises statistic
	CMA		Cramér–von Mises statistic (asymptotic)
	_K_	*	Kuiper two-sample statistic
	_KA_	*	Kuiper two-sample statistic (asymptotic)
	P_KA	*	*p*-value, two-sample Kuiper test
FP	_FP_	*	Fligner-Policello statistic
	PL_FP	*	*p*-value (left-sided), Fligner-Policello test
	PR_FP	*	*p*-value (right-sided), Fligner-Policello test
	P2_FP	*	*p*-value (two-sided), Fligner-Policello test
HL	_HL_	*	Hodges-Lehmann estimate, location shift
	L_HL	*	Lower confidence limit, Hodges-Lehmann
	U_HL	*	Upper confidence limit, Hodges-Lehmann
	M_HL	*	Confidence limit midpoint, Hodges-Lehmann
	E_HL	*	ASE of Hodges-Lehmann estimate
	XL_HL	*	Exact lower confidence limit, Hodges-Lehmann
	XU_HL	*	Exact upper confidence limit, Hodges-Lehmann
	XM_HL	*	Exact confidence limit midpoint
KLOTZ	_KLOTZ_	*	Two-sample Klotz statistic
	Z_K	*	Klotz statistic, standardized
	PL_K	*	*p*-value (left-sided), Klotz test
	PR_K	*	*p*-value (right-sided), Klotz test
	P2_K	*	*p*-value (two-sided), Klotz test
	XPL_K	*	Exact *p*-value (left-sided), Klotz test
	XPR_K	*	Exact *p*-value (right-sided), Klotz test

Table 90.5 *continued*

Output Option	Output Variables		Variable Descriptions
	XPT_K	*	Exact point probability, Klotz test
	XMP_K	*	Exact mid *p*-value, Klotz test
	XP2_K	*	Exact *p*-value (two-sided), Klotz test
	CHK		Klotz chi-square
	DF_CHK		Degrees of freedom, Klotz chi-square
	P_CHK		*p*-value, Klotz chi-square test
	XP_CHK	**	Exact *p*-value, Klotz chi-square test
	XPT_CHK	**	Exact point probability, Klotz chi-square test
	XMP_CHK	**	Exact mid *p*-value, Klotz chi-square test
MEDIAN	_MED_	*	Two-sample median statistic
	Z_MED	*	Median statistic, standardized
	PL_MED	*	*p*-value (left-sided), median test
	PR_MED	*	*p*-value (right-sided), median test
	P2_MED	*	*p*-value (two-sided), median test
	XPL_MED	*	Exact *p*-value (left-sided), median test
	XPR_MED	*	Exact *p*-value (right-sided), median test
	XPT_MED	*	Exact point probability, median test
	XMP_MED	*	Exact mid *p*-value, median test
	XP2_MED	*	Exact *p*-value (two-sided), median test
	CHMED		Median chi-square (Brown-Mood test)
	DF_CHMED		Degrees of freedom, median chi-square
	P_CHMED		*p*-value, median chi-square test
	XP_CHMED	**	Exact *p*-value, median chi-square test
	XPT_CHME	**	Exact point probability, median chi-square test
	XMP_CHMED	**	Exact mid *p*-value, median chi-square test
MOOD	_MOOD_	*	Two-sample Mood statistic
	Z_MOOD	*	Mood statistic, standardized
	PL_MOOD	*	*p*-value (left-sided), Mood test
	PR_MOOD	*	*p*-value (right-sided), Mood test
	P2_MOOD	*	*p*-value (two-sided), Mood test
	XPL_MOOD	*	Exact *p*-value (left-sided), Mood test
	XPR_MOOD	*	Exact *p*-value (right-sided), Mood test
	XPT_MOOD	*	Exact point probability, Mood test
	XMP_MOOD	*	Exact mid *p*-value, Mood test
	XP2_MOOD	*	Exact *p*-value (two-sided), Mood test
	CHMOOD		Mood chi-square
	DF_CHMOO		Degrees of freedom, Mood chi-square
	P_CHMOOD		*p*-value, Mood chi-square test
	XP_CHMOO	**	Exact *p*-value, Mood chi-square test
	XPT_CHMO	**	Exact point probability, Mood chi-square test
	XMP_CHMOOD	**	Exact mid *p*-value, Mood chi-square test

Table 90.5 *continued*

Output Option	Output Variables		Variable Descriptions
RMD	_RMD_	*	Ratio mean deviations (RMD) statistic
	XPL_RMD	*	Exact *p*-value (left-sided), RMD test
	XPR_RMD	*	Exact *p*-value (right-sided), RMD test
	XPT_RMD	*	Exact point probability, RMD test
	XMP_RMD	*	Exact mid *p*-value, RMD test
	RMD2	*	RMD2 two-sided statistic
	XP2_RMD	*	Exact *p*-value (two-sided), RMD test
SAVAGE	_SAV_	*	Two-sample Savage statistic
	Z_SAV	*	Savage statistic, standardized
	PL_SAV	*	*p*-value (left-sided), Savage test
	PR_SAV	*	*p*-value (right-sided), Savage test
	P2_SAV	*	*p*-value (two-sided), Savage test
	XPL_SAV	*	Exact *p*-value (left-sided), Savage test
	XPR_SAV	*	Exact *p*-value (right-sided), Savage test
	XPT_SAV	*	Exact point probability, Savage test
	XMP_SAV	*	Exact mid *p*-value, Savage test
	XP2_SAV	*	Exact *p*-value (two-sided), Savage test
	CHSAV		Savage chi-square
	DF_CHSAV		Degrees of freedom, Savage chi-square
	P_CHSAV		*p*-value, Savage chi-square test
	XP_CHSAV	**	Exact *p*-value, Savage chi-square test
	XPT_CHSA	**	Exact point probability, Savage chi-square test
	XMP_CHSAV	**	Exact mid *p*-value, Savage chi-square test
SCORES=DATA	_DATA_	*	Two-sample data scores statistic
	Z_DATA	*	Data scores statistic, standardized
	PL_DATA	*	*p*-value (left-sided), data scores test
	PR_DATA	*	*p*-value (right-sided), data scores test
	P2_DATA	*	*p*-value (two-sided), data scores test
	XPL_DATA	*	Exact *p*-value (left-sided), data scores test
	XPR_DATA	*	Exact *p*-value (right-sided), data scores test
	XPT_DATA	*	Exact point probability, data scores test
	XMP_DATA	*	Exact mid *p*-value, data scores test
	XP2_DATA	*	Exact *p*-value (two-sided), data scores test
	CHDATA		Data scores chi-square
	DF_CHDAT		Degrees of freedom, data scores chi-square
	P_CHDATA		*p*-value, data scores chi-square test
	XP_CHDAT	**	Exact *p*-value, data scores chi-square test
	XPT_CHDA	**	Exact point probability, data scores chi-square test
	XMP_CHDATA	**	Exact mid *p*-value, data scores chi-square test
ST	_ST_	*	Two-sample Siegel-Tukey statistic
	Z_ST	*	Siegel-Tukey statistic, standardized
	PL_ST	*	*p*-value (left-sided), Siegel-Tukey test

Table 90.5 *continued*

Output Option	Output Variables		Variable Descriptions
	PR_ST	*	*p*-value (right-sided), Siegel-Tukey test
	P2_ST	*	*p*-value (two-sided), Siegel-Tukey test
	XPL_ST	*	Exact *p*-value (left-sided), Siegel-Tukey test
	XPR_ST	*	Exact *p*-value (right-sided), Siegel-Tukey test
	XPT_ST	*	Exact point probability, Siegel-Tukey test
	XMP_ST	*	Exact mid *p*-value, Siegel-Tukey test
	XP2_ST	*	Exact *p*-value (two-sided), Siegel-Tukey test
	CHST		Siegel-Tukey chi-square
	DF_CHST		Degrees of freedom, Siegel-Tukey chi-square
	P_CHST		*p*-value, Siegel-Tukey chi-square test
	XP_CHST	**	Exact *p*-value, Siegel-Tukey chi-square test
	XPT_CHST	**	Exact point probability, Siegel-Tukey chi-square test
	XMP_CHST	**	Exact mid *p*-value, Siegel-Tukey chi-square test
VW | NORMAL	_VW_	*	Two-sample Van der Waerden statistic
	Z_VW	*	Van der Waerden statistic, standardized
	PL_VW	*	*p*-value (left-sided), Van der Waerden test
	PR_VW	*	*p*-value (right-sided), Van der Waerden test
	P2_VW	*	*p*-value (two-sided), Van der Waerden test
	XPL_VW	*	Exact *p*-value (left-sided), Van der Waerden test
	XPR_VW	*	Exact *p*-value (right-sided), Van der Waerden test
	XPT_VW	*	Exact point probability, Van der Waerden test
	XMP_VW	*	Exact mid *p*-value, Van der Waerden test
	XP2_VW	*	Exact *p*-value (two-sided), Van der Waerden test
	CHVW		Van der Waerden chi-square
	DF_CHVW		Degrees of freedom, Van der Waerden chi-square
	P_CHVW		*p*-value, Van der Waerden chi-square test
	XP_CHVW	**	Exact *p*-value, Van der Waerden chi-square test
	XPT_CHVW	**	Exact point probability, Van der Waerden chi-square test
	XMP_CHVW	**	Exact mid *p*-value, Van der Waerden chi-square test
WILCOXON	_WIL_	*	Two-sample Wilcoxon statistic
	Z_WIL	*	Wilcoxon statistic, standardized
	PL_WIL	*	*p*-value (left-sided), Wilcoxon test
	PR_WIL	*	*p*-value (right-sided), Wilcoxon test
	P2_WIL	*	*p*-value (two-sided), Wilcoxon test
	PTL_WIL	*	*p*-value (left-sided), Wilcoxon *t* approximation
	PTR_WIL	*	*p*-value (right-sided), Wilcoxon *t* approximation
	PT2_WIL	*	*p*-value (two-sided), Wilcoxon *t* approximation
	XPL_WIL	*	Exact *p*-value (left-sided), Wilcoxon test
	XPR_WIL	*	Exact *p*-value (right-sided), Wilcoxon test
	XPT_WIL	*	Exact point probability, Wilcoxon test
	XMP_WIL	*	Exact mid *p*-value, Wilcoxon test
	XP2_WIL	*	Exact *p*-value (two-sided), Wilcoxon test

Table 90.5 *continued*

Output Option	Output Variables		Variable Descriptions
	_KW_		Kruskal-Wallis statistic
	DF_KW		Degrees of freedom, Kruskal-Wallis test
	P_KW		*p*-value, Kruskal-Wallis test
	XP_KW	**	Exact *p*-value, Kruskal-Wallis test
	XPT_KW	**	Exact point probability, Kruskal-Wallis test
	XMP_KW	**	Exact mid *p*-value, Kruskal-Wallis test

*Statistic included only for two-sample data.

**Statistic included only for multisample data.

Displayed Output

If you specify the VARINFO option, PROC NPAR1WAY displays a "Variable Information" table for each analysis variable that you specify in the VAR statement. This table includes the following information:

- Number of Observations Used
- Number of Missing Observations
- Sum of Frequencies Used, if you specify a FREQ statement
- Number of Class Levels Used
- Number of Empty Class Levels Excluded
- Number of Strata, if you specify the ALIGN=STRATA option
- Number of Variable Levels, if you specify the VARINFO(TIES) option
- Number of Tied Observations, if you specify the VARINFO(TIES) option

If you specify the ANOVA option, PROC NPAR1WAY displays a "Class Means" table and an "Analysis of Variance" table for each response variable. The "Class Means" table includes the following information for each CLASS variable value (level):

- N, which is the number of observations
- Mean of the response variable

The "Analysis of Variance" table includes the following information for each Source of variation (Among classes and Within classes):

- DF, which is the degrees of freedom associated with the source

- Sum of Squares
- Mean Square, which is the sum of squares divided by the degrees of freedom

The "Analysis of Variance" table also includes the following:

- F Value for testing the hypothesis that the class means are equal, which is computed by dividing the Mean Square (Among) by the Mean Square (Within)
- Pr > F, which is the significance probability corresponding to the F Value

For each score type that you specify, PROC NPAR1WAY displays a "Class Scores" table. The available score types include Wilcoxon, median, Van der Waerden (normal), Savage, Siegel-Tukey, Ansari-Bradley, Klotz, Mood, Conover, and raw data scores. PROC NPAR1WAY computes the scores for the response variable values and classifies the scored observations according to the CLASS variable values. The "Class Scores" table includes the following information for each CLASS variable level:

- N, which is the number of observations
- Sum of Scores
- Expected Under H0, which is the expected sum of scores under the null hypothesis of no difference among classes
- Std Dev Under H0, which is the standard deviation under the null hypothesis
- Mean Score

When there are two levels of the CLASS variable, PROC NPAR1WAY displays a "Two-Sample Test" table for each analysis of scores. The "Two-Sample Test" table includes the following information:

- Statistic, which is the sum of scores for the class with the smaller sample size
- Z, which is the standardized test statistic and has an asymptotic standard normal distribution under the null hypothesis
- One-Sided Pr < Z or One-Sided Pr > Z, which is the asymptotic one-sided p-value. This is displayed as Pr < Z or Pr > Z, depending on whether Z is ≤ 0 or > 0.
- Two-Sided Pr > |Z|, which is the asymptotic two-sided p-value

For Wilcoxon scores, the "Two-Sample Test" table also includes a t Approximation for the Wilcoxon two-sample test.

If you request an exact test by specifying the score type in the EXACT statement, the "Two-Sample Test" table also includes the following exact p-values:

- One-Sided Pr $\leq$ S or One-Sided Pr $\geq$ S, which is the exact one-sided p-value. This is displayed as Pr $\leq$ S or Pr $\geq$ S, depending on whether S $\leq$ Mean or S > Mean, where S is the test statistic and Mean is its expected value under the null hypothesis.

- Point Pr = S, which is the exact point probability. This is displayed if you specify the POINT option in the EXACT statement.
- Mid p-Value, which is displayed if you specify the MIDP option in the EXACT statement
- Two-Sided Pr ≥ |S - Mean|, which is the exact two-sided *p*-value

If you request Monte Carlo estimates for a two-sample exact test by specifying the MC option in the EXACT statement, PROC NPAR1WAY displays the "Monte Carlo Estimates for the Exact Test" table, which includes the following information:

- Estimate of One-Sided Pr ≤ S or One-Sided Pr ≥ S, which is the exact one-sided *p*-value, together with the Lower and Upper Confidence Limits
- Estimate of Two-Sided Pr ≥ |S - Mean|, which is the exact two-sided *p*-value, together with the Lower and Upper Confidence Limits
- Number of Samples used to compute the Monte Carlo estimates
- Initial Seed used to compute the Monte Carlo estimates

For both two-sample and multisample data, PROC NPAR1WAY displays a "One-Way Analysis" table, which includes the following information:

- Chi-Square, which is the one-way ANOVA statistic for testing the null hypothesis of no difference among classes
- DF, which is the degrees of freedom
- Pr > Chi-Square, which is the asymptotic *p*-value

For multisample data, if you request an exact test by specifying the score type in the EXACT statement, the "One-Way Analysis" table also displays the exact *p*-value as follows:

- Exact Pr ≥ Chi-Square
- Exact Pr = Chi-Square, which is the point probability. This is displayed if you specify the POINT option in the EXACT statement.
- Exact Mid p-Value, which is displayed if you specify the MIDP option in the EXACT statement

For multisample data, if you specify the MC option in the EXACT statement, PROC NPAR1WAY displays the following information in the "Monte Carlo Estimate for the Exact Test" table:

- Estimate of Exact Pr ≥ Chi-Square, together with the Lower and Upper Confidence Limits
- Number of Samples used to compute the Monte Carlo estimate
- Initial Seed used to compute the Monte Carlo estimate

If you specify the HL option for two-sample data, PROC NPAR1WAY produces a "Hodges-Lehmann Estimation" table, which includes the following information:

- Location Shift estimate
- Confidence Limits for the Location Shift
- Confidence Interval Midpoint
- Asymptotic Standard Error estimate, which is based on the confidence interval

If you request exact Hodges-Lehmann confidence limits by specifying the HL option in the EXACT statement, the "Hodges-Lehmann Estimation" table also includes Exact Confidence Limits and the exact Interval Midpoint.

If you specify the FP option for two-sample data, PROC NPAR1WAY produces the "Fligner-Policello Placements" table and the "Fligner-Policello Test" table. The "Fligner-Policello Placements" table includes the following information for each CLASS variable level:

- N, which is the number of observations
- Sum, which is the sum of the placements
- Mean, which is the average placement
- Std Dev, which is the standard deviation of the placements

The "Fligner-Policello Test" table includes the following information:

- Difference, which is the difference in placement sums between the two CLASS levels (samples)
- Statistic (Z), which is the standardized test statistic and has an asymptotic standard normal distribution under the null hypothesis
- One-Sided Pr < Z or One-Sided Pr > Z, which is the one-sided *p*-value. This is displayed as Pr < Z or Pr > Z, depending on whether Z is ≤ 0 or > 0.
- Two-Sided Pr > |Z|, which is the two-sided *p*-value

If you specify the RMD option for two-sample data, PROC NPAR1WAY produces the "Deviations" table and the "Ratio Mean Deviations (RMD) Exact Test" table. The "Deviations" table includes the following information for each CLASS variable level:

- N, which is the number of observations
- Median, which is the sample median
- Sum of Deviations, which is the sum of the absolute deviations from the median
- Mean Deviation, which is the average of the absolute deviations

The "Ratio Mean Deviations (RMD) Exact Test" table includes the following information:

- RMD, which is the ratio mean deviations statistic
- Pr $\leq$ RMD or Pr $\geq$ RMD, which is the exact one-sided p-value. PROC NPAR1WAY displays Pr $\leq$ RMD when RMD is ≤ 1 and Pr $\geq$ RMD otherwise.
- RMD2, which is the modified RMD two-sided statistic
- Pr $\geq$ RMD2, which is the exact two-sided p-value

If you specify the DSCF option for multisample data, PROC NPAR1WAY produces the "Pairwise Two-Sided Multiple Comparison Analysis" table, which includes the following information for each two-sample comparison:

- Comparison, which identifies the two CLASS levels that are compared
- Wilcoxon Z, which is the standardized two-sample Wilcoxon statistic
- DSCF Value, which is the Dwass, Steel, Critchlow-Fligner statistic
- Pr > DSCF, which is the two-sided p-value

If you specify the EDF option, PROC NPAR1WAY produces tables for the Kolmogorov-Smirnov test, the Cramér–von Mises test, and for two-sample data only, the Kuiper test.

The "Kolmogorov-Smirnov Test" table includes the following information for each CLASS variable level:

- N, which is the number of observations
- EDF at Maximum, which is the value of the class EDF (empirical distribution function) at its maximum deviation from the pooled EDF
- Deviation from Mean at Maximum, which is the value of $\sqrt{n_i}\,(F_i - F)$ at its maximum, where n_i is the class sample size, F_i is the class EDF, and F is the pooled EDF

The "Kolmogorov-Smirnov Test" table displays the following statistics:

- KS, which is the Kolmogorov-Smirnov statistic
- KSa, which is the asymptotic Kolmogorov-Smirnov statistic, KSa = $\sqrt{n}$ KS

For two-sample data, the "Kolmogorov-Smirnov Test" table also displays the following statistics:

- Pr > KSa, which is the asymptotic p-value for KSa and equals Pr > D
- D, which is the two-sample Kolmogorov-Smirnov statistic, $\max_j |F_1(x_j) - F_2(x_j)|$

If you specify the D option for two-sample data, PROC NPAR1WAY displays the following one-sided Kolmogorov-Smirnov statistics and their asymptotic *p*-values in the "Kolmogorov-Smirnov Two-Sample Test" table:

- D+, which is $\max_j (F_1(x_j) - F_2(x_j))$
- Pr > D+
- D-, which is $\max_j (F_2(x_j) - F_1(x_j))$
- Pr > D-

For two-sample data, if you request exact Kolmogorov-Smirnov tests by specifying the KS option in the EXACT statement, PROC NPAR1WAY displays the following exact *p*-values in the "Kolmogorov-Smirnov Two-Sample Test" table:

- Exact Pr ≥ D
- Exact Pr ≥ D+
- Exact Pr ≥ D-
- Exact Point Pr = D, Exact Point Pr = D+, and Exact Point Pr = D-, if you specify the POINT option in the EXACT statement
- Exact Mid p-Values for D, D+, and D-, if you specify the MIDP option in the EXACT statement

If you request Monte Carlo estimates for the two-sample exact Kolmogorov-Smirnov test, PROC NPAR1WAY displays the following information in the "Kolmogorov-Smirnov Two-Sample Test" table:

- Estimate of Exact Pr ≥ D, together with the Lower and Upper Confidence Limits
- Estimate of Exact Pr ≥ D+, together with the Lower and Upper Confidence Limits
- Estimate of Exact Pr ≥ D-, together with the Lower and Upper Confidence Limits
- Number of Samples used to compute the Monte Carlo estimates
- Initial Seed used to compute the Monte Carlo estimates

The "Cramér–von Mises Test" table includes the following information for each CLASS variable level:

- N, which is the number of observations
- Summed Deviation from Mean, which is $(n_i/n) \sum_{j=1}^{p} t_j \left(F_i(x_j) - F(x_j)\right)^2$

The "Cramér–von Mises Statistics" table displays the following statistics:

- CM, which is the Cramér–von Mises statistic

- CMa, which is the asymptotic Cramér–von Mises statistic, CMa = n CM

For two-sample data, PROC NPAR1WAY displays the "Kuiper Test" table, which includes the following information for each CLASS variable level:

- N, which is the number of observations
- Deviation from Mean, which is $\max_j |F_1(x_j) - F_2(x_j)|$

The "Kuiper Two-Sample Statistics" table displays the following statistics:

- K, which is the Kuiper two-sample test statistic
- Ka, which is the asymptotic Kuiper two-sample test statistic, Ka = K $\sqrt{n_1 n_2 / n}$
- Pr > Ka

If you specify a STRATA statement along with the ALIGN=STRATA option in the PROC NPAR1WAY statement, PROC NPAR1WAY displays a "Stratum Alignment" table for each analysis variable. This table includes the following information for each stratum:

- Stratum index, which is a sequential stratum identification number
- STRATA variables, which list the levels of STRATA variables
- N Obs, which is the number of observations in the stratum
- Median, Mean, or HL Shift

If you specify a STRATA statement and do not specify the ALIGN=STRATA option, PROC NPAR1WAY displays the "Stratum Information," "Stratified Analysis Method," and "Class Information" tables.

The "Stratum Information" table displays the following information for each stratum:

- Stratum index, which is a sequential stratum identification number
- STRATA variables, which list the levels of STRATA variables
- N Obs, which is the number of observations in the stratum

The "Stratified Analysis Method" table displays the following information:

- Stratum Weights (Stratum Size or Equal)
- Ranks (Within Strata or Overall)

The "Class Information" tables displays the following information for each class:

- Class index, which is a sequential class identification number

- CLASS variable level
- N Obs, which is the number of observations in the class

If you request stratified analysis by using the STRATA statement, PROC NPAR1WAY displays the "Scores by Strata" and "Stratified Test" tables for each analysis variable and score type that you specify.

The "Scores by Strata" table includes the following information for each stratum:

- Stratum index, which is a sequential stratum identification number
- N Obs, which is the number of observations in the stratum
- Class N Obs, which is the number of observations in class 1
- Statistic, which is the sum of scores in class 1
- Expected, which is the expected value under H0
- Std Dev, which is the standard deviation under H0
- Mean score

The "Stratified Test" table includes the following information:

- N Obs, which is the total number of observations
- N Strata, which is the number of strata
- Statistic, which is the stratified sum of scores
- Expected, which is the expected value under H0
- Std Dev, which is the standard deviation under H0
- Z, which is the standardized test statistic
- Pr < Z or Pr > Z, which is the one-sided *p*-value
- Pr > |Z|, which is the two-sided *p*-value

ODS Table Names

PROC NPAR1WAY assigns a name to each table that it creates. You can use these names to refer to tables when you use the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 90.6 lists the ODS table names together with their descriptions and the options required to produce the tables. If you do not specify any analysis options in the PROC NPAR1WAY statement and do not specify a STRATA statement, PROC NPAR1WAY provides the ANOVA, WILCOXON, MEDIAN, VW (NORMAL), SAVAGE, and EDF analyses by default.

Table 90.6 ODS Tables Produced by PROC NPAR1WAY

ODS Table Name	Description	Statement	Option
ABAnalysis	Ansari-Bradley one-way analysis	PROC	AB
ABMC	Monte Carlo estimates for the Ansari-Bradley exact test	EXACT	AB / MC
ABScores	Ansari-Bradley scores	PROC	AB
ABTest*	Ansari-Bradley two-sample test	PROC	AB
ANOVA	Analysis of variance	PROC	ANOVA
ClassInfo*	Class information	STRATA	
ClassMeans	Class means	PROC	ANOVA
ConoverAnalysis	Conover one-way analysis	PROC	CONOVER
ConoverMC	Monte Carlo estimates for the Conover exact test	EXACT	CONOVER / MC
ConoverScores	Conover scores	PROC	CONOVER
ConoverTest*	Conover two-sample test	PROC	CONOVER
CVMStats	Cramér–von Mises statistics	PROC	EDF
CVMTest	Cramér–von Mises test	PROC	EDF
DataScores	Data scores	PROC	SCORES=DATA
DataScoresAnalysis	Data scores one-way analysis	PROC	SCORES=DATA
DataScoresMC	Monte Carlo estimates for the data scores exact test	EXACT	SCORES=DATA / MC
DataScoresStrata*	Data scores by strata	STRATA	SCORES=DATA
DataScoresStrataTest*	Stratified data scores test	STRATA	SCORES=DATA
DataScoresTest*	Data scores two-sample test	PROC	SCORES=DATA
DSCF**	DSCF multiple comparison analysis	PROC	DSCF
FPPlacements*	Fligner-Policello placements	PROC	FP
FPTest*	Fligner-Policello test	PROC	FP
HodgesLehmann*	Hodges-Lehmann estimation	PROC	HL
KlotzAnalysis	Klotz one-way analysis	PROC	KLOTZ
KlotzMC	Monte Carlo estimates for the Klotz exact test	EXACT	KLOTZ / MC
KlotzScores	Klotz scores	PROC	KLOTZ
KlotzTest*	Klotz two-sample test	PROC	KLOTZ
KruskalWallisMC**	Monte Carlo estimates for the Kruskal-Wallis exact test	EXACT	WILCOXON / MC

Table 90.6 *continued*

ODS Table Name	Description	Statement	Option
KruskalWallisTest	Kruskal-Wallis test	PROC	WILCOXON
KS2Stats*	Kolmogorov-Smirnov two-sample statistics	PROC	EDF
KSExactTest*	Kolmogorov-Smirnov exact test	EXACT	KS | EDF
KSMC*	Monte Carlo estimates for the Kolmogorov-Smirnov exact test	EXACT	KS | EDF / MC
KSStats**	Kolmogorov-Smirnov statistics	PROC	EDF
KSTest	Kolmogorov-Smirnov test	PROC	EDF
KuiperStats*	Kuiper two-sample statistics	PROC	EDF
KuiperTest*	Kuiper test	PROC	EDF
MedianAnalysis	Median one-way analysis	PROC	MEDIAN
MedianMC	Monte Carlo estimates for the median exact test	EXACT	MEDIAN / MC
MedianScores	Median scores	PROC	MEDIAN
MedianStrata*	Median scores by strata	STRATA	MEDIAN
MedianStrataTest*	Stratified median test	STRATA	MEDIAN
MedianTest*	Median two-sample test	PROC	MEDIAN
MoodAnalysis	Mood one-way analysis	PROC	MOOD
MoodMC	Monte Carlo estimates for the Mood exact test	EXACT	MOOD / MC
MoodScores	Mood scores	PROC	MOOD
MoodTest*	Mood two-sample test	PROC	MOOD
RMDDeviations*	RMD deviations	PROC or EXACT	RMD RMD
RMDMC*	Monte Carlo estimates for the RMD exact test	EXACT	RMD / MC
RMDTest*	RMD two-sample test	PROC or EXACT	RMD RMD
SavageAnalysis	Savage one-way analysis	PROC	SAVAGE
SavageMC	Monte Carlo estimates for the Savage exact test	EXACT	SAVAGE / MC
SavageScores	Savage scores	PROC	SAVAGE
SavageStrata*	Savage scores by strata	STRATA	SAVAGE
SavageStrataTest*	Stratified Savage test	STRATA	SAVAGE
SavageTest*	Savage two-sample test	PROC	SAVAGE
STAnalysis	Siegel-Tukey one-way analysis	PROC	ST
STMC	Monte Carlo estimates for the Siegel-Tukey exact test	EXACT	ST / MC
StrataAlign	Stratum alignment	PROC	ALIGN=STRATA
StrataInfo	Stratum information	STRATA	
StrataMethod*	Stratified analysis method	STRATA	
STScores	Siegel-Tukey scores	PROC	ST
STTest*	Siegel-Tukey two-sample test	PROC	ST
VarInfo	Variable information	PROC	VARINFO

Table 90.6 *continued*

ODS Table Name	Description	Statement	Option
VWAnalysis	Van der Waerden one-way analysis	PROC	VW | NORMAL
VWMC	Monte Carlo estimates for the Van der Waerden exact test	EXACT	VW | NORMAL / MC
VWScores	Van der Waerden scores	PROC	VW | NORMAL
VWStrata*	Van der Waerden scores by strata	STRATA	VW | NORMAL
VWStrataTest*	Stratified Van der Waerden test	STRATA	VW | NORMAL
VWTest*	Van der Waerden two-sample test	PROC	VW | NORMAL
WilcoxonMC*	Monte Carlo estimates for the Wilcoxon two-sample exact test	EXACT	WILCOXON / MC
WilcoxonScores	Wilcoxon scores	PROC	WILCOXON
WilcoxonStrata*	Wilcoxon scores by strata	STRATA	WILCOXON
WilcoxonStrataTest*	Stratified Wilcoxon test	STRATA	WILCOXON
WilcoxonTest*	Wilcoxon two-sample test	PROC	WILCOXON

*Table produced only for two-sample data.

**Table produced only for multisample data.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, you can request specific plots by specifying the PLOTS= option in the PROC NPAR1WAY statement. If you do not specify the PLOTS= option but have enabled ODS Graphics, PROC NPAR1WAY produces all plots that are associated with the analyses that you request.

PROC NPAR1WAY assigns a name to each graph that it creates by using ODS Graphics. You can use these names to refer to the graphs. Table 90.7 lists the names of the graphs that PROC NPAR1WAY generates together with their descriptions and the options that are required to produce the graphs.

Table 90.7 Graphs Produced by PROC NPAR1WAY

ODS Graph Name	Description	Option
ABBoxPlot	Box plot of Ansari-Bradley scores	AB
ANOVABoxPlot	Box plot of raw data	ANOVA
ConoverBoxPlot	Box plot of Conover scores	CONOVER
DataScoresBoxPlot	Box plot of data scores	SCORES=DATA
EDFPlot	Empirical distribution function plot	EDF
FPBoxPlot	Box plot of Fligner-Policello placements	FP
KlotzBoxPlot	Box plot of Klotz scores	KLOTZ
MedianPlot	Median plot	MEDIAN
MoodBoxPlot	Box plot of Mood scores	MOOD
SavageBoxPlot	Box plot of Savage scores	SAVAGE
STBoxPlot	Box plot of Siegel-Tukey scores	ST
VWBoxPlot	Box plot of Van der Waerden scores	VW | NORMAL
WilcoxonBoxPlot	Box plot of Wilcoxon scores	WILCOXON

Examples: NPAR1WAY Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-stat`.

Example 90.1: Two-Sample Location Tests and Plots

(View the complete code for this example (npar1ex1.sas) in the example repository.)

Fifty-nine female patients with rheumatoid arthritis who participated in a clinical trial were assigned to two groups, active and placebo. The response status (excellent=5, good=4, moderate=3, fair=2, poor=1) of each patient was recorded.

The following SAS statements create the data set Arthritis, which contains the observed status values for all the patients. The variable Treatment denotes the treatment received by a patient, and the variable Response contains the response status of the patient. The variable Freq contains the frequency of the observation, which is the number of patients with the Treatment and Response combination.

```
data Arthritis;
   input Treatment $ Response Freq @@;
   datalines;
Active 5 5 Active 4 11 Active 3 5 Active 2 1 Active 1 5
Placebo 5 2 Placebo 4 4 Placebo 3 7 Placebo 2 7 Placebo 1 12
;
```

The following PROC NPAR1WAY statements test the null hypothesis that there is no difference in the patient response status against the alternative hypothesis that the patient response status differs in the two treatment groups. The WILCOXON option requests the Wilcoxon test for difference in location, and the MEDIAN

option requests the median test for difference in location. The variable Treatment is the CLASS variable, and the VAR statement specifies that the variable Response is the analysis variable.

The PLOTS= option requests a box plot of the Wilcoxon scores and a median plot for Response classified by Treatment. ODS Graphics must be enabled before producing plots.

```
ods graphics on;
proc npar1way data=Arthritis wilcoxon median
              plots=(wilcoxonboxplot medianplot);
   class Treatment;
   var Response;
   freq Freq;
run;
ods graphics off;
```

Output 90.1.1 shows the results of the Wilcoxon analysis. The Wilcoxon two-sample test statistic equals 999.0, which is the sum of the Wilcoxon scores for the smaller sample (Active). This sum is greater than 810.0, which is the expected value under the null hypothesis of no difference between the two samples, Active and Placebo. The one-sided p-value is 0.0016, which indicates that the patient response for the Active treatment is significantly more than for the Placebo group.

Output 90.1.1 Wilcoxon Two-Sample Test

The NPAR1WAY Procedure

Wilcoxon Scores (Rank Sums) for Variable Response Classified by Variable Treatment

Treatment	N	Sum of Scores	Expected Under H0	Std Dev Under H0	Mean Score
Active	27	999.0	810.0	63.972744	37.000000
Placebo	32	771.0	960.0	63.972744	24.093750

Average scores were used for ties.

Wilcoxon Two-Sample Test

Statistic	Z	Pr > Z	Pr > |Z|	t Approximation Pr > Z	t Approximation Pr > |Z|
999.0000	2.9466	0.0016	0.0032	0.0023	0.0046

Z includes a continuity correction of 0.5.

Kruskal-Wallis Test

Chi-Square	DF	Pr > ChiSq
8.7284	1	0.0031

Output 90.1.2 displays the box plot of Wilcoxon scores classified by Treatment, which corresponds to the Wilcoxon analysis in Output 90.1.1. To remove the p-values from the box plot display, you can specify the NOSTATS plot option in parentheses after the WILCOXONBOXPLOT option.

Output 90.1.2 Box Plot of Wilcoxon Scores

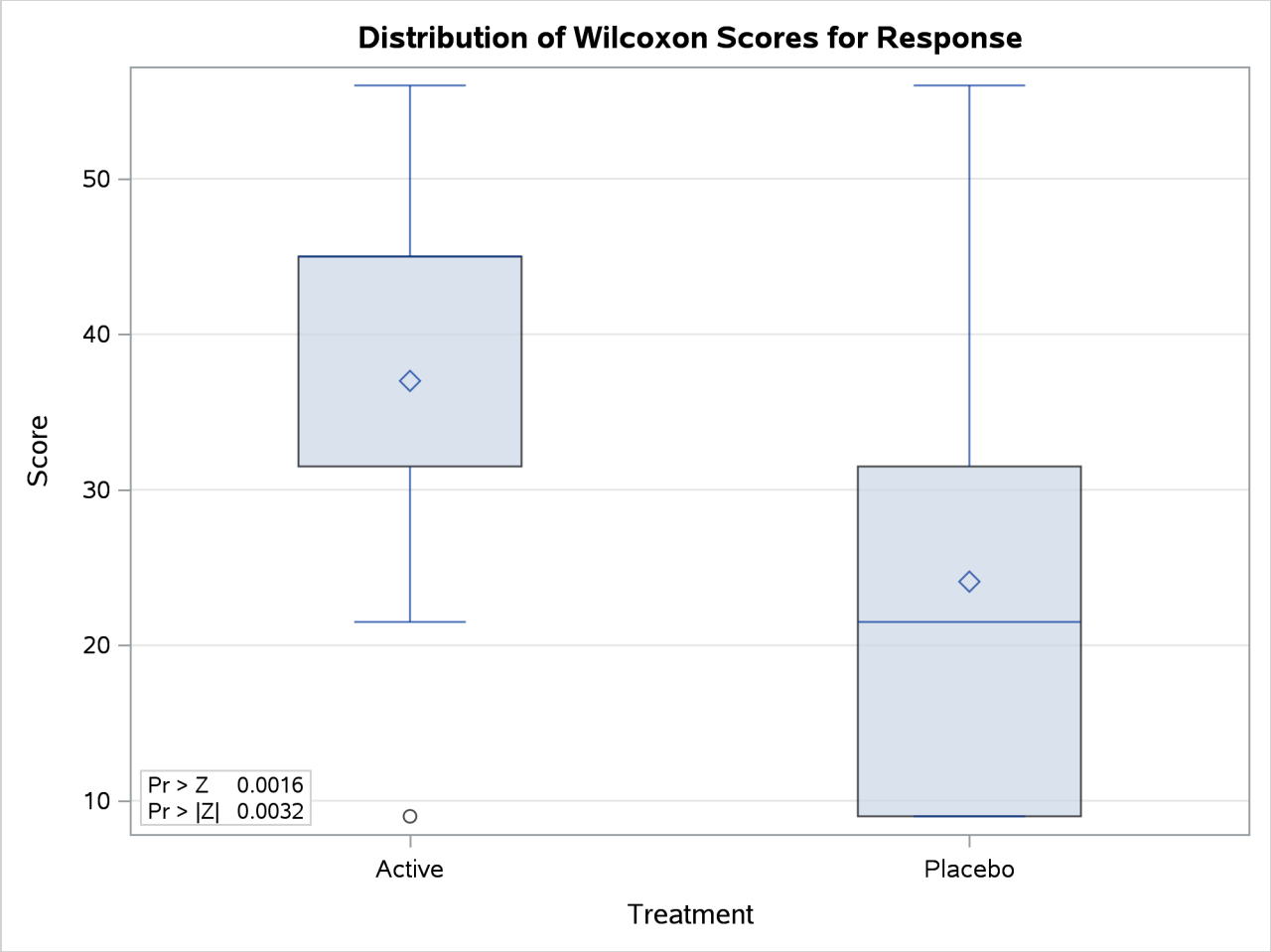

Output 90.1.3 shows the results of the median two-sample test. The value of the test statistic is 18.9167, and its standardized Z value is 3.2667. The one-sided p-value Pr > Z is 0.0005. This supports the alternative hypothesis that the effect of the Active treatment is greater than that of the Placebo.

Output 90.1.4 displays the median plot for the analysis of Response classified by Treatment. The median plot is a stacked bar chart showing the frequencies above and below the overall median. This plot corresponds to the median scores analysis in Output 90.1.3.

Output 90.1.3 Median Two-Sample Test

Median Scores (Number of Points Above Median) for Variable Response Classified by Variable Treatment

Treatment	N	Sum of Scores	Expected Under H0	Std Dev Under H0	Mean Score
Active	27	18.916667	13.271186	1.728195	0.700617
Placebo	32	10.083333	15.728814	1.728195	0.315104

Average scores were used for ties.

Output 90.1.3 *continued*

Median Two-Sample Test			
Statistic	**Z**	**Pr > Z**	**Pr > |Z|**
18.9167	3.2667	0.0005	0.0011

Median One-Way Analysis		
Chi-Square	**DF**	**Pr > ChiSq**
10.6713	1	0.0011

Output 90.1.4 Median Plot

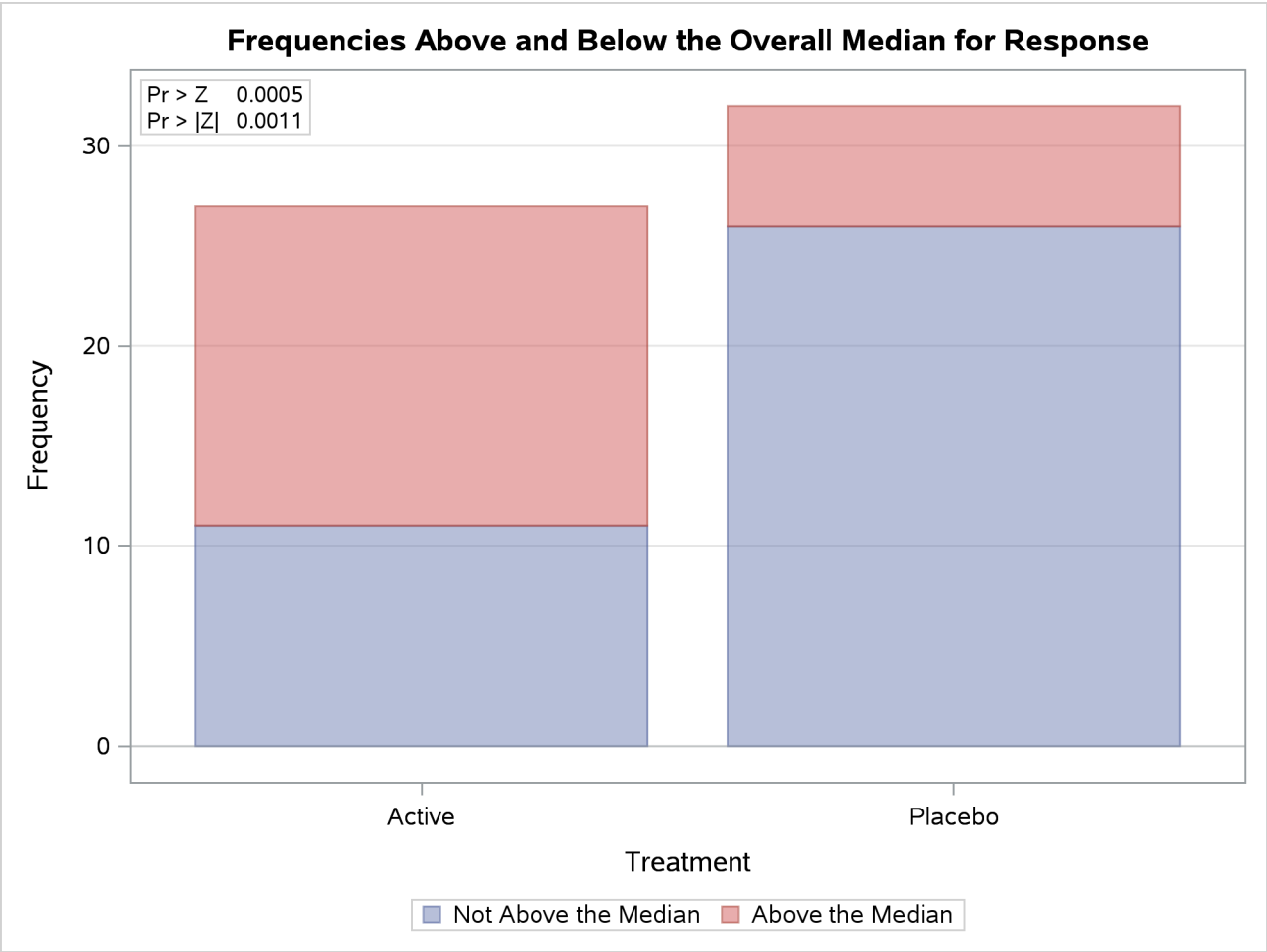

Example 90.2: EDF Statistics and EDF Plot

(View the complete code for this example (npar1ex2.sas) in the example repository.)

This example uses the SAS data set Arthritis created in Example 90.1. The data set contains the variable Treatment, which denotes the treatment received by a patient, and the variable Response, which contains the response status of the patient. The variable Freq contains the frequency of the observation, which is the number of patients with the Treatment and Response combination.

The following statements request empirical distribution function (EDF) statistics, which test whether the distribution of a variable is the same across different groups. The EDF option requests the EDF analysis. The variable Treatment is the CLASS variable, and the variable Response specified in the VAR statement is the analysis variable. The FREQ statement names Freq as the frequency variable.

The PLOTS= option requests an EDF plot for Response classified by Treatment. ODS Graphics must be enabled before producing plots.

```
ods graphics on;
proc npar1way edf plots=edfplot data=Arthritis;
   class Treatment;
   var Response;
   freq Freq;
run;
ods graphics off;
```

Output 90.2.1 shows EDF statistics that compare the two levels of Treatment, Active and Placebo. The asymptotic *p*-value for the Kolmogorov-Smirnov test is 0.0164. This supports rejection of the null hypothesis that the distributions are the same for the two samples.

Output 90.2.2 shows the EDF plot for Response classified by Treatment.

Output 90.2.1 Empirical Distribution Function Statistics

The NPAR1WAY Procedure

Kolmogorov-Smirnov Test for Variable Response Classified by Variable Treatment

Treatment	N	EDF at Maximum	Deviation from Mean at Maximum
Active	27	0.407407	-1.141653
Placebo	32	0.812500	1.048675
Total	59	0.627119	

Maximum Deviation Occurred at Observation 3

Value of Response at Maximum = 3.0

Kolmogorov-Smirnov Two-Sample Test (Asymptotic)

KS	0.201818	D	0.405093
KSa	1.550191	Pr > KSa	0.0164

Output 90.2.2 Empirical Distribution Function Plot

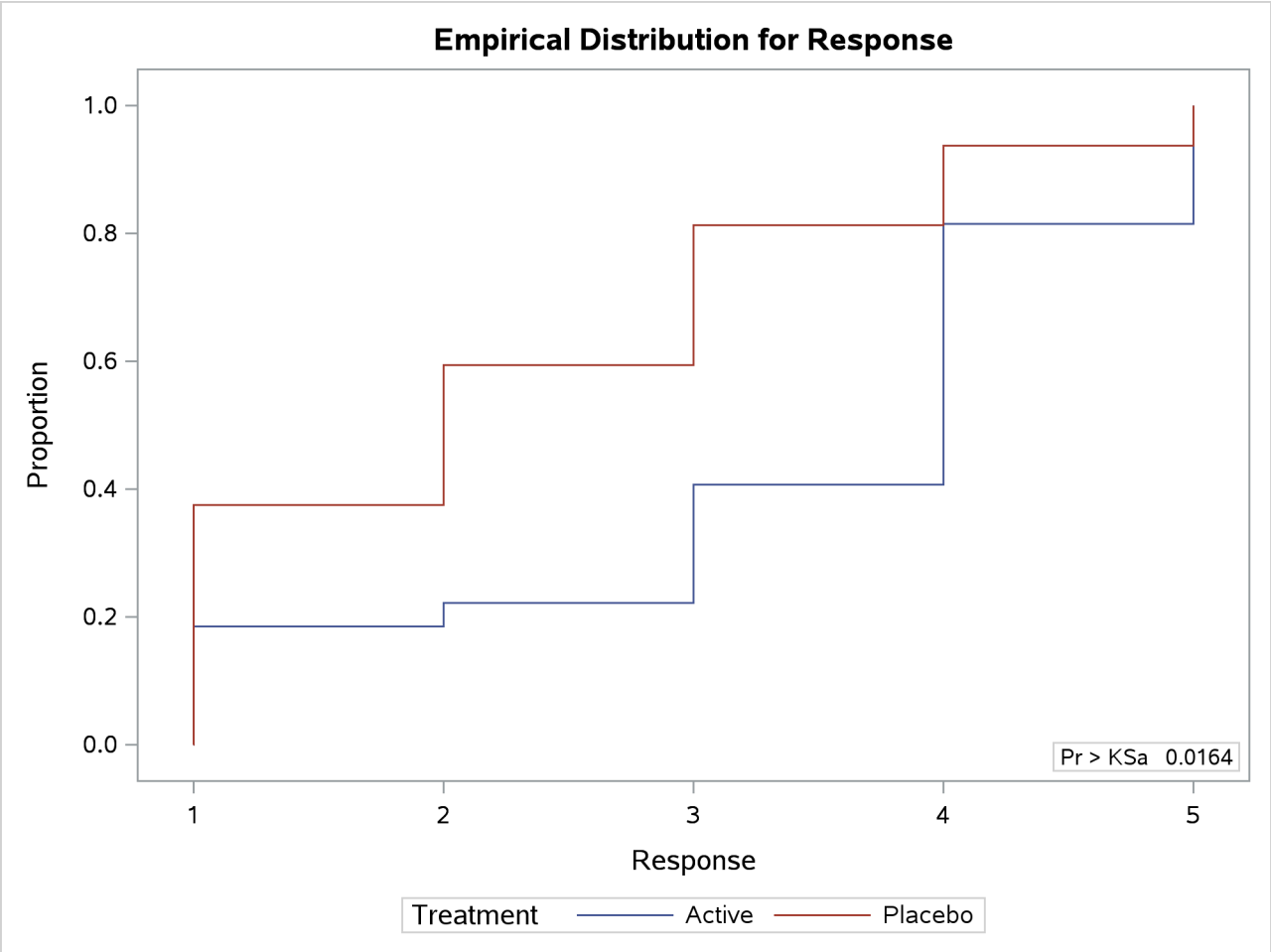

Example 90.3: Exact Wilcoxon Two-Sample Test

(View the complete code for this example (npar1ex3.sas) in the example repository.)

Researchers conducted an experiment to compare the effects of two stimulants. Thirteen randomly selected subjects received the first stimulant, and six randomly selected subjects received the second stimulant. The reaction times (in minutes) were measured while the subjects were under the influence of the stimulants.

The following SAS statements create the data set React, which contains the observed reaction times for each stimulant. The variable Stim represents Stimulant 1 or 2. The variable Time contains the reaction times observed for subjects under the stimulant.

```
data React;
   input Stim Time @@;
   datalines;
1 1.94   1 1.94   1 2.92   1 2.92   1 2.92   1 2.92   1 3.27
1 3.27   1 3.27   1 3.27   1 3.70   1 3.70   1 3.74
2 3.27   2 3.27   2 3.27   2 3.70   2 3.70   2 3.74
;
```

The following statements request a Wilcoxon test of the null hypothesis that there is no difference between the effects of the two stimulants. Stim is the CLASS variable, and Time is the analysis variable. The WILCOXON option requests an analysis of Wilcoxon scores. The CORRECT=NO option removes the continuity correction from the computation of the standardized z test statistic. The WILCOXON option in the EXACT statement requests exact p-values for the Wilcoxon test. Because the sample size is small, the large-sample normal approximation might not be adequate, and it is appropriate to compute the exact test. These statements produce the results shown in Output 90.3.1.

```
proc npar1way wilcoxon correct=no data=React;
   class Stim;
   var Time;
   exact wilcoxon;
run;
```

Output 90.3.1 displays the results of the Wilcoxon two-sample test. The value of the Wilcoxon statistic is 79.50. Because this value is greater than 60.0 (the expected value under the null hypothesis), PROC NPAR1WAY displays the right-sided p-values. The normal approximation for the Wilcoxon two-sample test yields a one-sided p-value of 0.0382 and a two-sided p-value of 0.0764. For the exact Wilcoxon test, the one-sided p-value is 0.0527, and the two-sided p-value is 0.1054.

Output 90.3.1 Exact Wilcoxon Two-Sample Test

The NPAR1WAY Procedure

Wilcoxon Scores (Rank Sums) for Variable Time Classified by Variable Stim

Stim	N	Sum of Scores	Expected Under H0	Std Dev Under H0	Mean Score
1	13	110.50	130.0	11.004784	8.500
2	6	79.50	60.0	11.004784	13.250

Average scores were used for ties.

Wilcoxon Two-Sample Test

				t Approximation			Exact
Statistic (S)	Z	Pr > Z	Pr > |Z|	Pr > Z	Pr > |Z|	Pr >= S	Pr >= |S-Mean|
79.5000	1.7720	0.0382	0.0764	0.0467	0.0933	0.0527	0.1054

Kruskal-Wallis Test

Chi-Square	DF	Pr > ChiSq
3.1398	1	0.0764

Example 90.4: Hodges-Lehmann Estimation

(View the complete code for this example (npar1ex4.sas) in the example repository.)

This example uses the SAS data set React created in Example 90.3. The data set contains the variable Stim, which represents Stimulant 1 or 2, and the variable Time, which contains the reaction times observed for subjects under the stimulant.

The following statements request Hodges-Lehmann estimation of the location shift between the two groups. Stim is the CLASS variable, and Time is the analysis variable. The HL option requests Hodges-Lehmann estimation. The ALPHA= option sets the confidence level for the Hodges-Lehmann confidence limits. The HL option in the EXACT statement requests exact confidence limits for the estimate of location shift. The ODS SELECT statement selects which tables to display. Output 90.4.1 shows the Hodges-Lehmann results.

```
proc npar1way hl alpha=.02 data=React;
   class Stim;
   var Time;
   exact hl;
   ods select WilcoxonScores HodgesLehmann;
run;
```

The HL option invokes the WILCOXON option, which produces a table of Wilcoxon scores (Output 90.4.1). The Hodges-Lehmann estimate of location shift is 0.35, and the asymptotic confidence limits are 0.00 and 0.82. The confidence interval midpoint is 0.41, which can also be used as an estimate of the location shift. The ASE estimate of 0.1762 is based on the length of the confidence interval. The exact confidence limits are 0.00 and 1.33.

Output 90.4.1 Hodges-Lehmann Estimate of Location Shift

The NPAR1WAY Procedure

Wilcoxon Scores (Rank Sums) for Variable Time Classified by Variable Stim

Stim	N	Sum of Scores	Expected Under H0	Std Dev Under H0	Mean Score
1	13	110.50	130.0	11.004784	8.500
2	6	79.50	60.0	11.004784	13.250

Average scores were used for ties.

Hodges-Lehmann Estimation

Location Shift (2 - 1) 0.3500

Type	98% Confidence Limits		Interval Midpoint	Asymptotic Standard Error
Asymptotic (Moses)	0.0000	0.8200	0.4100	0.1762
Exact	0.0000	1.3300	0.6650	

Example 90.5: Exact Savage Multisample Test

(View the complete code for this example (npar1ex5.sas) in the example repository.)

A researcher conducting a laboratory experiment randomly assigned 15 mice to receive one of three drugs. The survival time (in days) was then recorded.

The following SAS statements create the data set Mice, which contains the observed survival times for the mice. The variable Treatment denotes the treatment received. The variable Days contains the number of days the mouse survived.

```
data Mice;
   input Treatment $ Days @@;
   datalines;
1 1 1 1 1 3 1 3 1 4
2 3 2 4 2 4 2 4 2 15
3 4 3 4 3 10 3 10 3 26
;
```

The following statements request a Savage test of the null hypothesis that there is no difference in survival time among the three drugs. Treatment is the CLASS variable, and Days is the analysis variable. The SAVAGE option requests an analysis of Savage scores. The SAVAGE option in the EXACT statement requests exact *p*-values for the Savage test. Because the sample size is small, the large-sample normal approximation might not be adequate, and it is appropriate to compute the exact test.

PROC NPAR1WAY tests the null hypothesis that there is no difference in the survival times among the three drugs against the alternative hypothesis of difference among the drugs. The SAVAGE option specifies an analysis based on Savage scores. The variable Treatment is the CLASS variable, and the variable Days is the response variable. The EXACT statement requests the exact Savage test.

```
proc npar1way savage data=Mice;
   class Treatment;
   var Days;
   exact savage;
run;
```

Output 90.5.1 shows the results of the Savage test. The exact *p*-value is 0.0445, which supports a difference in survival times among the drugs at the 0.05 level. The asymptotic *p*-value based on the chi-square approximation is 0.0638.

Output 90.5.1 Exact Savage Multisample Test

The NPAR1WAY Procedure

Savage Scores (Exponential) for Variable Days Classified by Variable Treatment					
Treatment	N	Sum of Scores	Expected Under H0	Std Dev Under H0	Mean Score
1	5	-3.367980	0.0	1.634555	-0.673596
2	5	0.095618	0.0	1.634555	0.019124
3	5	3.272362	0.0	1.634555	0.654472
Average scores were used for ties.					

Output 90.5.1 *continued*

Savage One-Way Analysis			
Chi-Square	DF	Pr > ChiSq	Exact Pr >= ChiSq
5.5047	2	0.0638	0.0445

References

Agresti, A. (1992). "A Survey of Exact Inference for Contingency Tables." *Statistical Science* 7:131–177.

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Agresti, A. (2007). *An Introduction to Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Agresti, A. (2013). *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Agresti, A., Mehta, C. R., and Patel, N. R. (1990). "Exact Inference for Contingency Tables with Ordered Categories." *Journal of the American Statistical Association* 85:453–458.

Agresti, A., Wackerly, D., and Boyett, J. M. (1979). "Exact Conditional Tests for Cross-Classifications: Approximation of Attained Significance Levels." *Psychometrika* 44:75–83.

Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1975). *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, MA: MIT Press.

Butar, F. B., and Park, J.-W. (2008). "Permutation Tests for Comparing Two Populations." *Journal of Mathematical Sciences and Mathematics Education* 3:19–30.

Conover, W. J. (1999). *Practical Nonparametric Statistics*. 3rd ed. New York: John Wiley & Sons.

Critchlow, D. E., and Fligner, M. A. (1991). "On Distribution-Free Multiple Comparisons in the One-Way Analysis of Variance." *Communications in Statistics—Theory and Methods* 20:127–139.

Dwass, M. (1960). "Some k-Sample Rank-Order Tests." In *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, edited by I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow, and H. B. Mann, 198–202. Stanford, CA: Stanford University Press.

Fligner, M. A., and Policello, G. E. (1981). "Robust Rank Procedures for the Behrens-Fisher Problem." *Journal of the American Statistical Association* 76:162–168.

Gail, M. H., and Mantel, N. (1977). "Counting the Number of $r \times c$ Contingency Tables with Fixed Margins." *Journal of the American Statistical Association* 72:859–862.

Gibbons, J. D., and Chakraborti, S. (2010). *Nonparametric Statistical Inference*. 5th ed. New York: Chapman & Hall.

Hajek, J. (1969). *A Course in Nonparametric Statistics*. San Francisco: Holden-Day.

Halverson, J. O., and Sherwood, F. W. (1930). "Investigations in the Feeding of Cottonseed Meal to Cattle." *North Carolina Agricultural Experiment Station Technical Bulletin* 39, 158 pages.

Higgins, J. J. (2004). *An Introduction to Modern Nonparametric Statistics*. Belmont, CA: Brooks/Cole.

Hirji, K. F. (2006). *Exact Analysis of Discrete Data*. Boca Raton, FL: Chapman & Hall/CRC.

Hodges, J. L., Jr. (1957). "The Significance Probability of the Smirnov Two-Sample Test." *Arkiv för Matematik* 3:469–486.

Hodges, J. L., Jr., and Lehmann, E. L. (1962). "Rank Methods for Combination of Independent Experiments in Analysis of Variance." *Annals of Mathematical Statistics* 33:482–497.

Hodges, J. L., Jr., and Lehmann, E. L. (1983). "Hodges-Lehmann Estimators." In *Encyclopedia of Statistical Sciences.* vol. 3, edited by S. Kotz, N. L. Johnson, and C. B. Read. New York: John Wiley & Sons.

Hollander, M., and Wolfe, D. A. (1999). *Nonparametric Statistical Methods*. 2nd ed. New York: John Wiley & Sons.

Juneau, P. (2004). "Simultaneous Nonparametric Inference in a One-Way Layout Using the SAS System." In *Proceedings of PharmaSUG 2004 (Pharmaceutical Industry SAS Users Group).* Paper no. 04. Cary, NC: SAS Institute Inc.

Juneau, P. (2007). "Nonparametric Methods in Pharmaceutical Statistics." In *Pharmaceutical Statistics Using SAS: A Practical Guide*, edited by A. Dmitrienko, C. Chuang-Stein, and R. D'Agostino, 117–150. Cary, NC: SAS Institute Inc.

Kiefer, J. (1959). "K-Sample Analogues of the Kolmogorov-Smirnov and Cramér–von Mises Tests." *Annals of Mathematical Statistics* 30:420–447.

Lehmann, E. L. (1963). "Nonparametric Confidence Intervals for a Shift Parameter." *Annals of Mathematical Statistics* 34:1507–1512.

Lehmann, E. L., and D'Abrera, H. J. M. (2006). *Nonparametrics: Statistical Methods Based on Ranks*. Rev. ed. New York: Springer.

Mehrotra, D. V., Lu, X., and Li, X. (2010). "Rank-Based Analyses of Stratified Experiments: Alternatives to the van Elteren Test." *American Statistician* 64:121–130.

Mehta, C. R., and Patel, N. R. (1983). "A Network Algorithm for Performing Fisher's Exact Test in $r \times c$ Contingency Tables." *Journal of the American Statistical Association* 78:427–434.

Mehta, C. R., Patel, N. R., and Senchaudhuri, P. (1991). "Exact Stratified Linear Rank Tests for Binary Data." In *Computing Science and Statistics: Proceedings of the Twenty-Third Symposium on the Interface*, edited by E. M. Keramidas, 200–207. Fairfax Station, VA: Interface Foundation.

Mehta, C. R., Patel, N. R., and Tsiatis, A. A. (1984). "Exact Significance Testing to Establish Treatment Equivalence with Ordered Categorical Data." *Biometrics* 40:819–825.

Orban, J., and Wolfe, D. A. (1979). *A Class of Distribution-Free Two-Sample Tests Based on Placements*. Technical Report 178, Department of Statistics, Ohio State University.

Owen, D. B. (1962). *Handbook of Statistical Tables*. Reading, MA: Addison-Wesley.

Quade, D. (1966). "On Analysis of Variance for the k-Sample Problem." *Annals of Mathematical Statistics* 37:1747–1758.

Randles, R. H., and Wolfe, D. A. (1979). *Introduction to the Theory of Nonparametric Statistics*. New York: John Wiley & Sons.

Richter, S. J., and McCann, M. H. (2017). "Permutation Tests of Scale Using Deviances." *Communications in Statistics—Simulation and Computation* 46:5553–5565.

Sheskin, D. J. (1997). *Handbook of Parametric and Nonparametric Statistical Procedures*. Boca Raton, FL: CRC Press.

Steel, R. G. D. (1960). "A Rank Sum Test for Comparing All Pairs of Treatments." *Technometrics* 2:197–207.

Valz, P. D., and Thompson, M. E. (1994). "Exact Inference for Kendall's S and Spearman's ρ with Extensions to Fisher's Exact Test in $r \times c$ Contingency Tables." *Journal of Computational and Graphical Statistics* 3:459–472.

Van Elteren, P. H. (1960). "On the Combination of Independent Two-Sample Tests of Wilcoxon." *Bulletin of the International Statistical Institute* 37:351–361.

Chapter 91
The ORTHOREG Procedure

Contents

Overview: ORTHOREG Procedure **7434**
Getting Started: ORTHOREG Procedure **7434**
Longley Data 7434
Syntax: ORTHOREG Procedure **7437**
PROC ORTHOREG Statement 7438
BY Statement 7439
CLASS Statement 7440
EFFECT Statement 7441
EFFECTPLOT Statement 7442
ESTIMATE Statement 7443
LSMEANS Statement 7444
LSMESTIMATE Statement 7446
MODEL Statement 7447
SLICE Statement 7447
STORE Statement 7447
TEST Statement 7448
WEIGHT Statement 7448
Details: ORTHOREG Procedure **7449**
Missing Values 7449
Output Data Set 7449
Displayed Output 7449
ODS Table Names 7450
ODS Graphics 7450
Examples: ORTHOREG Procedure **7451**
Example 91.1: Precise Analysis of Variance 7451
Example 91.2: Wampler Data 7454
Example 91.3: Fitting Polynomials 7456
References **7460**

Overview: ORTHOREG Procedure

The ORTHOREG procedure fits general linear models by the method of least squares. Other SAS/STAT software procedures, such as the GLM and REG procedures, fit the same types of models, but PROC ORTHOREG can produce more accurate estimates than other regression procedures when your data are ill-conditioned. Instead of collecting crossproducts, PROC ORTHOREG uses Gentleman-Givens transformations to update and compute the upper triangular matrix **R** of the QR decomposition of the data matrix, with special care for scaling (Gentleman 1972, 1973). This method has the advantage over other orthogonalization methods (for example, Householder transformations) of not requiring the data matrix to be stored in memory.

The standard SAS regression procedures (PROC REG and PROC GLM) are very accurate for most problems. However, if you have very ill-conditioned data, these procedures can produce estimates that yield an error sum of squares very close to the minimum but still different from the exact least squares estimates. Normally, this coincides with estimates that have very high standard errors. In other words, the numerical error is much smaller than the statistical standard error.

PROC ORTHOREG fits models by the method of linear least squares, minimizing the sum of the squared residuals for predicting the responses—that is, the distance between the regression line and the observed Ys. The "ORTHO" in the name of the procedure refers to the orthogonalization approach to solving the least squares equations. In particular, PROC ORTHOREG does *not* perform the modeling method known as "orthogonal regression," which minimizes a different criterion (namely, the distance between the regression line and the X/Y points taken together.)

Getting Started: ORTHOREG Procedure

Longley Data

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/orthin1.sas.)

The labor statistics data set of Longley (1967) is noted for being ill-conditioned. Both the ORTHOREG and GLM procedures are applied for comparison (only portions of the PROC GLM results are shown).

NOTE: The results from this example vary from machine to machine, depending on floating-point configuration.

The following statements read the data into the SAS data set Longley:

```
title 'PROC ORTHOREG used with Longley data';
data Longley;
   input Employment Prices GNP Jobless Military PopSize Year;
   datalines;
60323  83.0 234289 2356 1590 107608 1947
61122  88.5 259426 2325 1456 108632 1948
60171  88.2 258054 3682 1616 109773 1949
61187  89.5 284599 3351 1650 110929 1950
63221  96.2 328975 2099 3099 112075 1951
```

```
63639  98.1 346999 1932 3594 113270 1952
64989  99.0 365385 1870 3547 115094 1953
63761 100.0 363112 3578 3350 116219 1954
66019 101.2 397469 2904 3048 117388 1955
67857 104.6 419180 2822 2857 118734 1956
68169 108.4 442769 2936 2798 120445 1957
66513 110.8 444546 4681 2637 121950 1958
68655 112.6 482704 3813 2552 123366 1959
69564 114.2 502601 3931 2514 125368 1960
69331 115.7 518173 4806 2572 127852 1961
70551 116.9 554894 4007 2827 130081 1962
;
```

The data set contains one dependent variable, Employment (total derived employment), and six independent variables: Prices (GNP implicit price deflator normalized to the value 100 in 1954), GNP (gross national product), Jobless (unemployment), Military (size of armed forces), PopSize (noninstitutional population aged 14 and over), and Year (year).

The following statements use the ORTHOREG procedure to model the Longley data by using a quadratic model in each independent variable, without interaction:

```
proc orthoreg data=Longley;
   model Employment = Prices   Prices*Prices
                      GNP      GNP*GNP
                      Jobless  Jobless*Jobless
                      Military Military*Military
                      PopSize  PopSize*PopSize
                      Year     Year*Year;
run;
```

Figure 91.1 shows the resulting analysis.

Figure 91.1 PROC ORTHOREG Results

PROC ORTHOREG used with Longley data

The ORTHOREG Procedure

Dependent Variable: Employment

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	12	184864508.5	15405375.71	320.24	0.0003
Error	3	144317.4957	48105.83189		
Corrected Total	15	185008826.0			

Root MSE	219.33041717
R-Square	0.9992199426

Figure 91.1 *continued*

Parameter	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	186931078.64012	154201839.7	1.21	0.3122
Prices	1	1324.506793624790	916.1745583	1.45	0.2440
Prices**2	1	-6.619239228454	4.789144565	-1.38	0.2609
GNP	1	-0.127686421562	0.0738897784	-1.73	0.1824
GNP**2	1	3.1369569286211E-8	8.7167753E-8	0.36	0.7428
Jobless	1	-4.355076535587	1.385179240	-3.14	0.0515
Jobless**2	1	0.000221329441	0.0001763541	1.26	0.2983
Military	1	4.911620145609	1.826715856	2.69	0.0745
Military**2	1	-0.001137071467	0.0003539971	-3.21	0.0489
PopSize	1	-0.030399723432	5.927253824	-0.01	0.9962
PopSize**2	1	-0.000001212511415	0.0000237262	-0.05	0.9625
Year	1	-194907.13904	157739.2876	-1.24	0.3045
Year**2	1	50.806760353824	40.27987894	1.26	0.2963

The estimates in Figure 91.1 compare very well with the best estimates available; for additional information, see Longley (1967) and Beaton, Rubin, and Barone (1976).

The following statements request the same analysis from the GLM procedure:

```
proc glm data=Longley;
   model Employment = Prices   Prices*Prices
                      GNP      GNP*GNP
                      Jobless  Jobless*Jobless
                      Military Military*Military
                      PopSize  PopSize*PopSize
                      Year     Year*Year;
   ods select OverallANOVA
              FitStatistics
              ParameterEstimates
              Notes;
run;
```

Figure 91.2 contains the overall ANOVA table and the parameter estimates produced by PROC GLM. Notice that the PROC ORTHOREG fit achieves a somewhat smaller root mean square error (RMSE) and also that the GLM procedure detects spurious singularities.

Figure 91.2 Partial PROC GLM Results

PROC ORTHOREG used with Longley data

The GLM Procedure

Dependent Variable: Employment

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	11	184791061.6	16799187.4	308.58	<.0001
Error	4	217764.4	54441.1		
Corrected Total	15	185008826.0			

Figure 91.2 *continued*

R-Square	Coeff Var	Root MSE	Employment Mean
0.998823	0.357221	233.3262	65317.00

Parameter	Estimate		Standard Error	t Value	Pr > |t|
Intercept	-3598851.832	B	1327335.647	-2.71	0.0535
Prices	523.802		688.979	0.76	0.4894
Prices*Prices	-2.326		3.507	-0.66	0.5434
GNP	-0.138		0.078	-1.76	0.1526
GNP*GNP	0.000		0.000	0.24	0.8218
Jobless	-4.599		1.459	-3.15	0.0344
Jobless*Jobless	0.000		0.000	1.14	0.3183
Military	4.994		1.942	2.57	0.0619
Military*Military	-0.001		0.000	-3.15	0.0346
PopSize	-4.246		5.156	-0.82	0.4565
PopSize*PopSize	0.000	B	0.000	0.81	0.4655
Year	0.000	B	.	.	.
Year*Year	1.038		0.419	2.48	0.0683

Note: The X'X matrix has been found to be singular, and a generalized inverse was used to solve the normal equations. Terms whose estimates are followed by the letter 'B' are not uniquely estimable.

Syntax: ORTHOREG Procedure

The following statements are available in the ORTHOREG procedure:

```
PROC ORTHOREG <options> ;
   CLASS variables </ option> ;
   MODEL dependent-variable = independent-effects </ option> ;
   BY variables ;
   EFFECT name = effect-type (variables </ options>) ;
   EFFECTPLOT <plot-type <(plot-definition-options)>> </ options> ;
   ESTIMATE <'label'> estimate-specification </ options> ;
   LSMEANS <model-effects> </ options> ;
   LSMESTIMATE model-effect lsmestimate-specification </ options> ;
   SLICE model-effect </ options> ;
   STORE <OUT=>item-store-name </ LABEL='label'> ;
   TEST <model-effects> </ options> ;
   WEIGHT variable ;
```

The BY, CLASS, MODEL, and WEIGHT statements are described in full after the PROC ORTHOREG statement in alphabetical order. The EFFECT, EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, SLICE, STORE, and TEST statements are common to many procedures. Summary descriptions of functionality and syntax for these statements are also given after the PROC ORTHOREG statement in alphabetical order, and full documentation about them is available in Chapter 20, "Shared Concepts and Topics."

PROC ORTHOREG Statement

PROC ORTHOREG < *options* > **;**

The PROC ORTHOREG statement invokes the ORTHOREG procedure. Table 91.1 summarizes the *options* available in the PROC ORTHOREG statement.

Table 91.1 PROC ORTHOREG Statement Options

Option	Description
DATA=	Specifies the input SAS data set
NAMELEN=	Specifies the maximum length of effect names
NOPRINT	Suppresses the normal display of results
ORDER=	Specifies the order in which to sort class levels
OUTEST=	Produces an output data set
SINGULAR=	Specifies the singularity criterion

The PROC ORTHOREG statement has the following *options*:

DATA=*SAS-data-set*

specifies the input SAS data set to use. By default, the procedure uses the most recently created SAS data set. The specified data set cannot be a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set.

NAMELEN=*number*

specifies the maximum length of effect names in tables and output data sets to be *number* characters, where *number* is a value between 20 and 2,000. By default, NAMELEN=20.

NOPRINT

suppresses the normal display of results. This option temporarily disables the Output Delivery System (ODS); for more information, see Chapter 23, "Using the Output Delivery System."

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This ordering determines which parameters in the model correspond to each level in the data, so the ORDER= option may be useful when you use ESTIMATE statement. This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUTEST=*SAS-data-set*
produces an output data set that contains the parameter estimates, the BY variables, and the special variables _TYPE_ (value "**PARMS**"), _NAME_ (blank), and _RMSE_ (root mean squared error).

SINGULAR=*s*
specifies a singularity criterion ($s \geq 0$) for the inversion of the triangular matrix $\mathbf{R}$. By default, SINGULAR=1E–12.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC ORTHOREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the ORTHOREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(REF=** *option***)** > . . . < *variable* < **(REF=** *option***)** > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC ORTHOREG statement.

You can specify the following REF= option to indicate how the levels of an individual classification variable are to be ordered by enclosing it in parentheses after the variable name:

REF=*'level'* **| FIRST | LAST**

specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC ORTHOREG's singular parameterization. You can specify the *level* of the variable to use as the reference level; specify a value that corresponds to the formatted value of the variable if a format is assigned. Alternatively, you can specify REF=FIRST to designate that the first ordered level serve as the reference, or REF=LAST to designate that the last ordered level serve as the reference. To specify that REF=FIRST or REF=LAST be used for all classification variables, use the REF= *global-option* after the slash (/) in the CLASS statement.

You can specify the following *global-options* in the CLASS statement after a slash (/):

REF=FIRST | LAST

specifies a level of all classification variables to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC ORTHOREG's singular parameterization. Specify REF=FIRST to designate that the first ordered level for each classification variable serve as the reference. Specify REF=LAST to designate that the last ordered level serve as the reference. This option applies to all the variables specified in the CLASS statement. To specify different reference levels for different classification variables, use REF= options for individual variables.

TRUNCATE

specifies that class levels be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 91.2 summarizes the *options* available in the EFFECT statement.

Table 91.2 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.

Table 91.2 *continued*

Option	Description
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

EFFECTPLOT Statement

EFFECTPLOT < *plot-type* < *(plot-definition-options)* > > < / *options* > **;**

The EFFECTPLOT statement produces a display of the fitted model and provides options for changing and enhancing the displays. Table 91.3 describes the available *plot-types* and their *plot-definition-options*.

Table 91.3 *Plot-Types* and *Plot-Definition-Options*

Plot-Type and Description	Plot-Definition-Options
BOX	
Displays a box plot of continuous response data at each level of a CLASS effect, with predicted values superimposed and connected by a line. This is an alternative to the INTERACTION *plot-type*.	PLOTBY= variable or CLASS effect X= CLASS variable or effect
CONTOUR	
Displays a contour plot of predicted values against two continuous covariates.	PLOTBY= variable or CLASS effect X= continuous variable Y= continuous variable

Table 91.3 *continued*

Plot-Type and Description	Plot-Definition-Options
FIT Displays a curve of predicted values versus a continuous variable.	PLOTBY= variable or CLASS effect X= continuous variable
INTERACTION Displays a plot of predicted values (possibly with error bars) versus the levels of a CLASS effect. The predicted values are connected with lines and can be grouped by the levels of another CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= CLASS variable or effect
MOSAIC Displays a mosaic plot of predicted values using up to three CLASS effects.	PLOTBY= variable or CLASS effect X= CLASS effects
SLICEFIT Displays a curve of predicted values versus a continuous variable grouped by the levels of a CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= continuous variable

For full details about the syntax and options of the EFFECTPLOT statement, see the section "EFFECTPLOT Statement" on page 447 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 91.4 summarizes the *options* available in the ESTIMATE statement.

Table 91.4 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference

Table 91.4 *continued*

Option	Description
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected *p*-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the **L** matrix
JOINT	Produces a joint *F* or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement" on page 475 in Chapter 20, "Shared Concepts and Topics."

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 91.5 summarizes the *options* available in the LSMEANS statement.

Table 91.5 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 91.6 summarizes the *options* available in the LSMESTIMATE statement.

Table 91.6 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

MODEL *dependent-variable* **=** *independent-effects* < / *option* > **;**

The MODEL statement names the dependent variable and the independent effects. Only one MODEL statement is allowed. The specification of effects and the parameterization of the linear model are the same as in the GLM procedure; for more information, see Chapter 53, "The GLM Procedure."

The following *option* can be used in the MODEL statement:

NOINT

omits the intercept term from the model. Often, this omission also changes the total sum of squares in the ANOVA and the value of R square to forms of these statistics that are not corrected for the mean. However, if the model is determined to contain an implicit intercept, in the sense that the all-ones intercept vector is in the column space of the design, then the usual mean-corrected forms of these statistics are used.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

TEST Statement

TEST < *model-effects* > < / *options* > **;**

The TEST statement enables you to perform *F* tests for model effects that test Type I, Type II, or Type III hypotheses. For more information about constructing Type I, II, and III estimable functions, see Chapter 16, "The Four Types of Estimable Functions."

Table 91.7 summarizes the *options* available in the TEST statement.

Table 91.7 TEST Statement Options

Option	Description
CHISQ	Requests chi-square tests
DDF=	Specifies denominator degrees of freedom for fixed effects
E	Requests Type I, Type II, and Type III coefficients
E1	Requests Type I coefficients
E2	Requests Type II coefficients
E3	Requests Type III coefficients
HTYPE=	Indicates the type of hypothesis test to perform
INTERCEPT	Adds a row that corresponds to the overall intercept

For more information about the syntax of the TEST statement, see the section "TEST Statement" on page 556 in Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

A WEIGHT statement names a variable in the input data set whose values are relative weights for a weighted least squares regression. If the weight value is proportional to the reciprocal of the variance for each observation, the weighted estimates are the best linear unbiased estimates (BLUE). For a more complete description of the WEIGHT statement, see the section "WEIGHT Statement" on page 4206 in Chapter 53, "The GLM Procedure."

Details: ORTHOREG Procedure

Missing Values

If there is a missing value for any model variable in an observation, the entire observation is dropped from the analysis.

Output Data Set

The OUTEST= option produces a TYPE=EST output SAS data set that contains the BY variables, parameter estimates, and four special variables. For each new value of the BY variables, PROC ORTHOREG outputs an observation to the OUTEST= data set. The variables in the data set are as follows:

- parameter estimates for all variables listed in the MODEL statement
- BY variables
- _TYPE_, which is a character variable with the value PARMS for every observation
- _NAME_, which is a character variable left blank for every observation
- _RMSE_, which is the root mean square error (the estimate of the standard deviation of the true errors)
- Intercept, which is the estimated intercept. This variable does not exist in the OUTEST= data set if the NOINT option is specified.

Displayed Output

PROC ORTHOREG displays the parameter estimates and associated statistics. These include the following:

- overall model analysis of variance, including the error mean square, which is an estimate of σ^2 (the variance of the true errors), and the overall F test for a model effect.
- root mean square error, which is an estimate of the standard deviation of the true errors. It is calculated as the square root of the mean squared error.
- R square (R^2) measures how much variation in the dependent variable can be accounted for by the model. R square, which can range from 0 to 1, is the ratio of the sum of squares for the model to the corrected total sum of squares. In general, the larger the value of R square, the better the model's fit.
- estimates for the parameters in the linear model

The table of parameter estimates consists of the following:

- the terms used as regressors, including the intercept.
- degrees of freedom (DF) for the variable. There is one degree of freedom for each parameter being estimated unless the model is not full rank.
- estimated linear coefficients.
- estimates of the standard errors of the parameter estimates.
- the critical t values for testing whether the parameters are This is computed as the parameter estimate divided by its standard error.
- the two-sided p-value for the t test, which is the probability that a t statistic would obtain a greater absolute value than that observed given that the true parameter is zero.

ODS Table Names

PROC ORTHOREG assigns a name to each table it creates. You can use these names to reference the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 91.8. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Each of the EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements also creates tables, which are not listed in Table 91.8. For information about these tables, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Table 91.8 ODS Tables Produced by PROC ORTHOREG

ODS Table Name	Description	Statement
ANOVA	Analysis of variance	Default
FitStatistics	Overall statistics for fit	Default
Levels	Table of class levels	CLASS statement
ParameterEstimates	Parameter estimates	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, then each of the EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements can produce plots associated with their analyses. For information about these plots, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Examples: ORTHOREG Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 91.1: Precise Analysis of Variance

(View the complete code for this example (orthex1.sas) in the example repository.)

The data for the following example are from Powell, Murphy, and Gramlich (1982). In order to calibrate an instrument for measuring atomic weight, 24 replicate measurements of the atomic weight of silver (chemical symbol Ag) are made with the new instrument and with a reference instrument.

NOTE: The results from this example vary from machine to machine, depending on floating-point configuration.

The following statements read the measurements for the two instruments into the SAS data set AgWeight:

```
title 'Atomic Weight of Silver by Two Different Instruments';
data AgWeight;
   input Instrument AgWeight @@;
   datalines;
1 107.8681568   1 107.8681465   1 107.8681572   1 107.8681785
1 107.8681446   1 107.8681903   1 107.8681526   1 107.8681494
1 107.8681616   1 107.8681587   1 107.8681519   1 107.8681486
1 107.8681419   1 107.8681569   1 107.8681508   1 107.8681672
1 107.8681385   1 107.8681518   1 107.8681662   1 107.8681424
1 107.8681360   1 107.8681333   1 107.8681610   1 107.8681477
2 107.8681079   2 107.8681344   2 107.8681513   2 107.8681197
2 107.8681604   2 107.8681385   2 107.8681642   2 107.8681365
2 107.8681151   2 107.8681082   2 107.8681517   2 107.8681448
2 107.8681198   2 107.8681482   2 107.8681334   2 107.8681609
2 107.8681101   2 107.8681512   2 107.8681469   2 107.8681360
2 107.8681254   2 107.8681261   2 107.8681450   2 107.8681368
;
```

Notice that the variation in the atomic weight measurements is several orders of magnitude less than their mean. This is a situation that can be difficult for standard, regression-based analysis-of-variance procedures to handle correctly.

The following statements invoke the ORTHOREG procedure to perform a simple one-way analysis of variance, testing for differences between the two instruments:

```
proc orthoreg data=AgWeight;
   class Instrument;
   model AgWeight = Instrument;
run;
```

Output 91.1.1 shows the resulting analysis.

Output 91.1.1 PROC ORTHOREG Results for Atomic Weight Example

Atomic Weight of Silver by Two Different Instruments

The ORTHOREG Procedure

Class Level Information		
Factor	Levels	Values
Instrument	2	1 2

Atomic Weight of Silver by Two Different Instruments

The ORTHOREG Procedure

Dependent Variable: AgWeight

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	3.6383419E-9	3.6383419E-9	15.95	0.0002
Error	46	1.0495173E-8	2.281559E-10		
Corrected Total	47	1.4133515E-8			

Root MSE	0.0000151048
R-Square	0.2574265445

Parameter	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	107.868136354167	3.0832608E-6	3.499E7	<.0001
(Instrument='1')	1	0.000017412499999	4.3603893E-6	3.99	0.0002
(Instrument='2')	0	0	.	.	.

The mean difference between instruments is about 1.74×10^{-5} (the value of the `(Instrument='1')` parameter in the parameter estimates table), whereas the level of background variation in the measurements is about 1.51×10^{-5} (the value of the root mean square error). At this level of error, the difference is significant, with a p-value of 0.0002.

The National Institute of Standards and Technology (1998) has provided certified ANOVA values for this data set. The following statements use ODS to examine the ANOVA values produced by ORTHOREG more precisely, for comparison with the NIST-certified values:

```
ods exclude all;

proc orthoreg data=AgWeight;
   class Instrument;
   model AgWeight = Instrument;
   ods output ANOVA         = OrthoregANOVA
```

```
              FitStatistics = OrthoregFitStat;
run;

ods exclude none;

data _null_;
   set OrthoregANOVA  (in=inANOVA)
       OrthoregFitStat(in=inFitStat);
   if (inANOVA) then do;
      if (Source = 'Model') then put "Model SS: " ss e20.;
      if (Source = 'Error') then put "Error SS: " ss e20.;
   end;
   if (inFitStat) then do;
      if (Statistic = 'Root MSE') then
                           put "Root MSE: " nValue1 e20.;
      if (Statistic = 'R-Square') then
                         put "R-Square: " nValue1 best20.;
   end;
run;
```

Table 91.9 and Table 91.10 compare the ANOVA values certified by NIST with those produced by ORTHOREG. As you can see, the agreement is quite good.

Table 91.9 Accuracy Comparison for Sums of Squares

Values	Model SS	Error SS
NIST-certified	3.6383418750000E–09	1.0495172916667E–08
ORTHOREG	3.6383418747907E–09	1.0495172916797E–08

Table 91.10 Accuracy Comparison for Fit Statistics

Values	Root MSE	R Square
NIST-certified	1.5104831444641E–05	0.25742654453832
ORTHOREG	1.5104831444735E–05	0.25742654452494

Example 91.2: Wampler Data

(View the complete code for this example (orthex2.sas) in the example repository.)

This example applies the ORTHOREG procedure to a collection of data sets noted for being ill-conditioned. The OUTEST= data set is used to collect the results for comparison with values certified to be correct by the National Institute of Standards and Technology (1998).

NOTE: The results from this example vary from machine to machine, depending on floating-point configuration.

The data are from Wampler (1970). The independent variates for all five data sets are x^i, $i = 1, \ldots 5$, for $x = 0, 1, \ldots, 20$. Two of the five dependent variables are exact linear functions of the independent terms:

$$\begin{aligned} y_1 &= 1 + x + x^2 + x^3 + x^4 + x^5 \\ y_2 &= 1 + 0.1x + 0.01x^2 + 0.001x^3 + 0.0001x^4 + 0.00001x^5 \end{aligned}$$

The other three dependent variables have the same mean value as y_1, but with nonzero errors:

$$\begin{aligned} y_3 &= y_1 + \mathbf{e} \\ y_4 &= y_1 + 100\mathbf{e} \\ y_5 &= y_1 + 10000\mathbf{e} \end{aligned}$$

where $\mathbf{e}$ is a vector of values with standard deviation $\sim$2044, chosen to be orthogonal to the mean model for y_1.

The following statements create a SAS data set Wampler that contains the Wampler data, run a SAS macro program that uses PROC ORTHOREG to fit a fifth-order polynomial in x to each of the Wampler dependent variables, and collect the results in a data set named ParmEst:

```
data Wampler;
   do x=0 to 20;
      input e @@;
      y1 = 1 +       x    +        x**2 +      x**3
             +       x**4 +        x**5;
      y2 = 1 + .1   *x    + .01   *x**2 + .001*x**3
             + .0001*x**4 + .00001*x**5;
      y3 = y1 +       e;
      y4 = y1 +   100*e;
      y5 = y1 + 10000*e;
      output;
   end;
   datalines;
759 -2048 2048 -2048 2523 -2048 2048 -2048 1838 -2048 2048
-2048 1838 -2048 2048 -2048 2523 -2048 2048 -2048 759
;
```

```
%macro WTest;
   data ParmEst; if (0); run;
   %do i = 1 %to 5;
      proc orthoreg data=Wampler outest=ParmEst&i noprint;
         model y&i = x x*x x*x*x x*x*x*x x*x*x*x*x;
      data ParmEst&i; set ParmEst&i; Dep = "y&i";
      data ParmEst; set ParmEst ParmEst&i;
         label Col1='x'     Col2='x**2'  Col3='x**3'
               Col4='x**4'  Col5='x**5';
      run;
   %end;
%mend;
%WTest;
```

Instead of displaying the raw values of the RMSE and parameter estimates, use an additional DATA step as follows to compute the deviations from the values certified to be correct by the National Institute of Standards and Technology (1998):

```
data ParmEst; set ParmEst;
   if      (Dep = 'y1') then
      _RMSE_ = _RMSE_ - 0.00000000000000;
   else if (Dep = 'y2') then
      _RMSE_ = _RMSE_ - 0.00000000000000;
   else if (Dep = 'y3') then
      _RMSE_ = _RMSE_ - 2360.14502379268;
   else if (Dep = 'y4') then
      _RMSE_ = _RMSE_ - 236014.502379268;
   else if (Dep = 'y5') then
      _RMSE_ = _RMSE_ - 23601450.2379268;
   if (Dep ^= 'y2') then do;
      Intercept = Intercept - 1.00000000000000;
      Col1      = Col1      - 1.00000000000000;
      Col2      = Col2      - 1.00000000000000;
      Col3      = Col3      - 1.00000000000000;
      Col4      = Col4      - 1.00000000000000;
      Col5      = Col5      - 1.00000000000000;
   end;
   else do;
      Intercept = Intercept - 1.00000000000000;
      Col1      = Col1      - 0.100000000000000;
      Col2      = Col2      - 0.100000000000000e-1;
      Col3      = Col3      - 0.100000000000000e-2;
      Col4      = Col4      - 0.100000000000000e-3;
      Col5      = Col5      - 0.100000000000000e-4;
   end;
run;

proc print data=ParmEst label noobs;
   title 'Wampler data: Deviations from Certified Values';
   format _RMSE_ Intercept Col1-Col5 e9.;
   var Dep _RMSE_ Intercept Col1-Col5;
run;
```

The results, shown in Output 91.2.1, indicate that the values computed by PROC ORTHOREG are quite

close to the NIST-certified values.

Output 91.2.1 Wampler Data: Deviations from Certified Values

Wampler data: Deviations from Certified Values

Dep	_RMSE_	Intercept	x	x**2	x**3	x**4	x**5
y1	0.00E+00	-2.65E-10	7.28E-12	2.91E-11	-5.34E-12	3.55E-13	-7.44E-15
y2	0.00E+00	1.11E-15	-1.53E-15	1.04E-15	-1.48E-16	9.36E-18	-2.00E-19
y3	-3.87E-11	1.27E-10	-1.20E-10	3.37E-11	-3.64E-12	2.34E-13	-5.33E-15
y4	-3.78E-10	1.89E-09	-4.23E-09	1.51E-09	-1.96E-10	1.07E-11	-2.07E-13
y5	-3.73E-08	2.27E-07	-4.30E-07	1.52E-07	-1.98E-08	1.08E-09	-2.08E-11

Example 91.3: Fitting Polynomials

(View the complete code for this example (orthex3.sas) in the example repository.)

The extra accuracy of the regression algorithm used by PROC ORTHOREG is most useful when the model contains near-singularities that you want to be able to distinguish from true singularities. This example demonstrates this usefulness in the context of fitting polynomials of high degree.

NOTE: The results from this example vary from machine to machine, depending on floating-point configuration.

The following DATA step computes a response y as an exact ninth-degree polynomial function of a predictor x evaluated at 0, 0.01, 0.02, . . . , 1.

```
title 'Polynomial Data';
data Polynomial;
   do i = 1 to 101;
      x = (i-1)/(101-1);
      y = 10**(9/2);
      do j = 0 to 8;
         y = y * (x - j/8);
      end;
      output;
   end;
run;
```

The polynomial is constructed in such a way that its zeros lie at $x = i/8$ for $i = 0, \ldots, 8$. The following statements use the EFFECT statement to fit a ninth-degree polynomial to this data with PROC ORTHOREG. The EFFECT statement makes it easy to specify complicated polynomial models.

```
ods graphics on;

proc orthoreg data=Polynomial;
   effect xMod = polynomial(x / degree=9);
   model y = xMod;
   effectplot fit / obs;
   store OStore;
run;
```

```
ods graphics off;
```

The effect xMod defined by the EFFECT statement refers to all nine degrees of freedom in the ninth-degree polynomial (excluding the intercept term). The resulting output is shown in Output 91.3.1. Note that the R square for the fit is 1, indicating that the ninth-degree polynomial has been correctly fit.

Output 91.3.1 PROC ORTHOREG Results for Ninth-Degree Polynomial

Polynomial Data

The ORTHOREG Procedure

Dependent Variable: y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	9	15.52718006	1.725242228	4.14E22	<.0001
Error	91	3.788943E-21	4.163673E-23		
Corrected Total	100	15.52718006			

Root MSE	6.452653E-12
R-Square	1

Parameter	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-5.46580558591E-11	5.125258E-12	-10.66	<.0001
x	1	75.997731243698	3.094014E-10	2.46E11	<.0001
x^2	1	-1652.407813609130	6.002412E-9	-275E9	<.0001
x^3	1	14249.45398	5.2496523E-8	2.71E11	<.0001
x^4	1	-64932.46157	2.4632391E-7	-264E9	<.0001
x^5	1	173315.35936	6.7372187E-7	2.57E11	<.0001
x^6	1	-280158.03646	1.1068403E-6	-253E9	<.0001
x^7	1	269781.81288	1.0751352E-6	2.51E11	<.0001
x^8	1	-142302.49471	5.6865866E-7	-25E10	<.0001
x^9	1	31622.77660	1.2617109E-7	2.51E11	<.0001

The fit plot produced by the EFFECTPLOT statement, Output 91.3.2, also demonstrates the perfect fit.

Output 91.3.2 PROC ORTHOREG Fit Plot for Ninth-Degree Polynomial

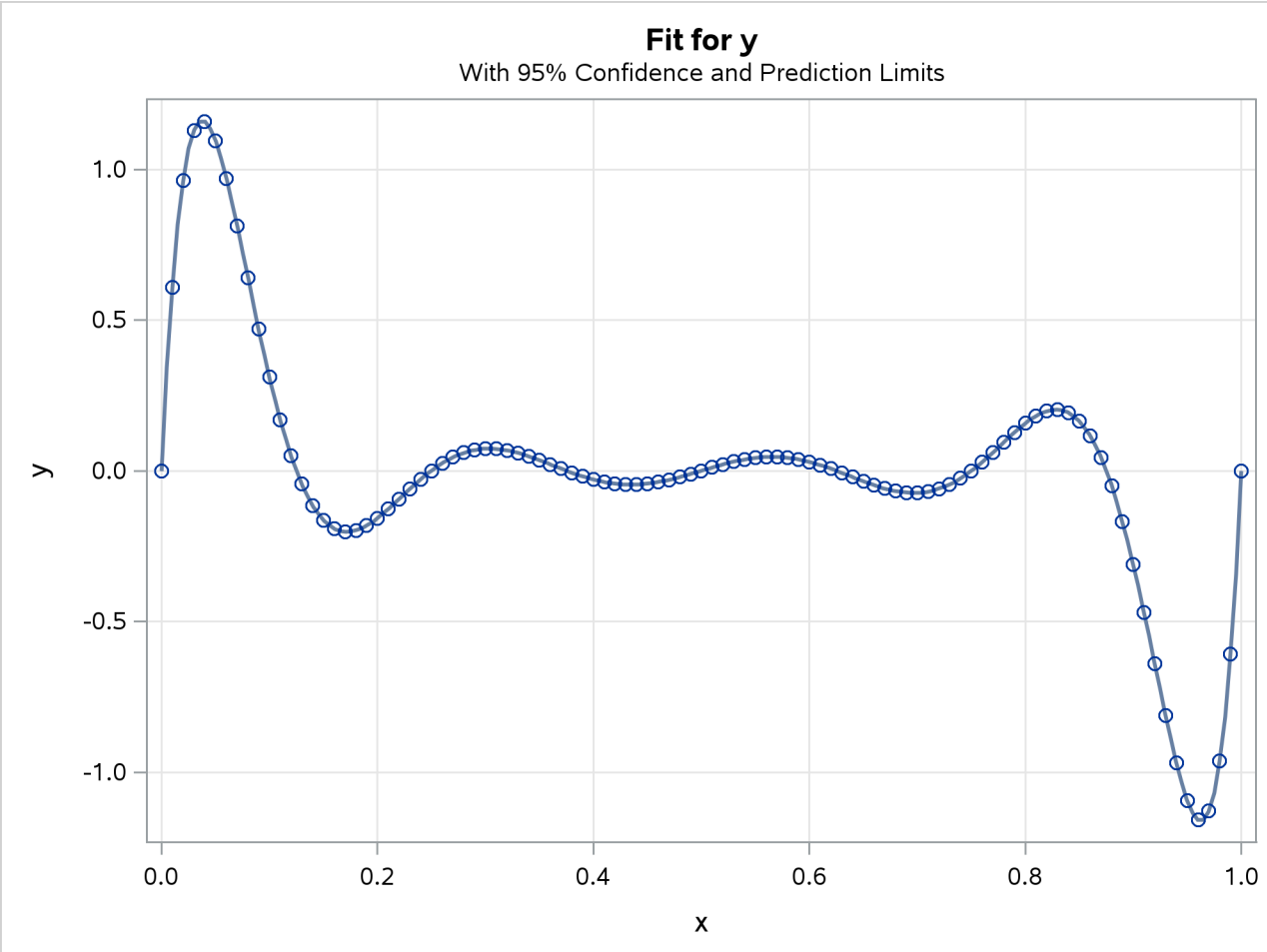

Finally, you can use the PLM procedure with the fit model saved by the STORE statement in the item store OStore to check the predicted values for the known zeros of the polynomial, as shown in the following statements:

```
data Zeros(keep=x);
   do j = 0 to 8;
      x = j/8;
      output;
   end;
run;

proc plm restore=OStore noprint;
   score data=Zeros out=OZeros pred=OPred;
run;

proc print noobs;
run;
```

The predicted values of the zeros, shown in Output 91.3.3, are again all minuscule.

Output 91.3.3 Predicted Zeros for Ninth-Degree Polynomial

Polynomial Data

x	OPred
0.000	-5.4658E-11
0.125	-2.8932E-11
0.250	-3.4226E-12
0.375	2.05178E-11
0.500	5.07114E-11
0.625	9.11768E-11
0.750	1.56888E-10
0.875	2.89219E-10
1.000	4.69299E-10

To compare these results with those from a least squares fit produced by an alternative algorithm, consider fitting a polynomial to this data using the GLM procedure. PROC GLM does not have an EFFECT statement, but the familiar bar notation can still be used to specify a ninth-degree polynomial fairly succinctly, as shown in the following statements:

```
proc glm data=Polynomial;
   model y = x|x|x|x|x|x|x|x|x;
   store GStore;
run;
```

Partial results are shown in Output 91.3.4. In this case, the R square for the fit is only about 0.83, indicating that the full ninth-degree polynomial was not correctly fit.

Output 91.3.4 PROC GLM for Ninth-Degree Polynomial

Polynomial Data

The GLM Procedure

Dependent Variable: y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	8	12.91166714	1.61395839	56.77	<.0001
Error	92	2.61551291	0.02842949		
Corrected Total	100	15.52718006			

R-Square	Coeff Var	Root MSE	y Mean
0.831553	-6.6691E17	0.168610	-0.000000

The following statements, which use the PLM procedure to compute predictions based on the GLM fit at the true zeros of the polynomial, also confirm that PROC GLM is not able to correctly fit a polynomial of this degree, as shown in Output 91.3.5.

```
proc plm restore=GStore noprint;
   score data=Zeros out=GZeros pred=GPred;
run;

data Zeros;
   merge OZeros GZeros;
run;

proc print noobs;
run;
```

Output 91.3.5 Predicted Zeros for Ninth-Degree Polynomial

Polynomial Data

x	OPred	GPred
0.000	-5.4658E-11	0.44896
0.125	-2.8932E-11	0.22087
0.250	-3.4226E-12	-0.19037
0.375	2.05178E-11	0.12710
0.500	5.07114E-11	-0.00000
0.625	9.11768E-11	-0.12710
0.750	1.56888E-10	0.19037
0.875	2.89219E-10	-0.22087
1.000	4.69299E-10	-0.44896

References

Beaton, A. E., Rubin, D. B., and Barone, J. L. (1976). "The Acceptability of Regression Solutions: Another Look at Computational Accuracy." *Journal of the American Statistical Association* 71:158–168.

Gentleman, W. M. (1972). *Basic Procedures for Large, Sparse, or Weighted Least Squares Problems*. Technical Report CSRR-2068, University of Waterloo, Ontario.

Gentleman, W. M. (1973). "Least Squares Computations by Givens Transformations without Square Roots." *Journal of the Institute of Mathematics and Its Applications* 12:329–336.

Lawson, C. L., and Hanson, R. J. (1974). *Solving Least Squares Problems*. Englewood Cliffs, NJ: Prentice-Hall.

Longley, J. W. (1967). "An Appraisal of Least Squares Programs for the Electronic Computer from the Point of View of the User." *Journal of the American Statistical Association* 62:819–841.

National Institute of Standards and Technology (1998). "Statistical Reference Data Sets." Accessed June 6, 2011. http://www.itl.nist.gov/div898/strd/general/dataarchive.html.

Powell, L. J., Murphy, T. J., and Gramlich, J. W. (1982). "The Absolute Isotopic Abundance and Atomic Weight of a Reference Sample of Silver." *NBS Journal of Research* 87:9–19.

Wampler, R. H. (1970). "A Report of the Accuracy of Some Widely Used Least Squares Computer Programs." *Journal of the American Statistical Association* 65:549–563.

Chapter 92
The PHREG Procedure

Contents

Overview: PHREG Procedure . . . **7465**
Getting Started: PHREG Procedure . . . **7467**
- Classical Method of Maximum Likelihood . . . 7468
- Bayesian Analysis . . . 7471

Syntax: PHREG Procedure . . . **7475**
- PROC PHREG Statement . . . 7476
- ASSESS Statement . . . 7485
- BASELINE Statement . . . 7486
- BAYES Statement . . . 7494
- BY Statement . . . 7508
- CLASS Statement . . . 7508
- CONTRAST Statement . . . 7512
- EFFECT Statement . . . 7515
- ESTIMATE Statement . . . 7516
- FREQ Statement . . . 7518
- HAZARDRATIO Statement . . . 7518
- ID Statement . . . 7521
- LSMEANS Statement . . . 7521
- LSMESTIMATE Statement . . . 7522
- MODEL Statement . . . 7524
- OUTPUT Statement . . . 7534
- Programming Statements . . . 7538
- RANDOM Statement . . . 7540
- ROC Statement . . . 7542
- STRATA Statement . . . 7542
- SLICE Statement . . . 7543
- STORE Statement . . . 7543
- TEST Statement . . . 7543
- WEIGHT Statement . . . 7545

Details: PHREG Procedure . . . **7545**
- Failure Time Distribution . . . 7545
- Time and CLASS Variables Usage . . . 7546
- Partial Likelihood Function for the Cox Model . . . 7550
- Counting Process Style of Input . . . 7551
- Left-Truncation of Failure Times . . . 7553
- The Multiplicative Hazards Model . . . 7553

Proportional Rates/Means Models for Recurrent Events 7554
The Frailty Model 7555
Analysis of Competing-Risks Data 7556
Hazard Ratios 7563
Newton-Raphson Method 7566
Firth's Modification for Maximum Likelihood Estimation 7567
Robust Sandwich Variance Estimate 7568
Testing the Global Null Hypothesis 7569
Type 3 Tests and Joint Tests 7570
Confidence Limits for a Hazard Ratio 7571
Using the TEST Statement to Test Linear Hypotheses 7573
Analysis of Multivariate Failure Time Data 7574
Model Fit Statistics 7582
Schemper-Henderson Predictive Measure 7583
Residuals 7585
Diagnostics Based on Weighted Residuals 7587
Influence of Observations on Overall Fit of the Model 7588
Concordance Statistics 7589
Time-Dependent ROC Curves 7591
Survivor Function Estimators 7596
Caution about Using Survival Data with Left Truncation 7601
Effect Selection Methods 7604
Assessment of the Proportional Hazards Model 7605
The Penalized Partial Likelihood Approach for Fitting Frailty Models 7607
Specifics for Bayesian Analysis 7610
Input and Output Data Sets 7621
Displayed Output 7624
ODS Table Names 7633
ODS Graphics 7636
Examples: PHREG Procedure **7638**
Example 92.1: Stepwise Regression 7638
Example 92.2: Best Subset Selection 7644
Example 92.3: Modeling with Categorical Predictors 7645
Example 92.4: Firth's Correction for Monotone Likelihood 7652
Example 92.5: Conditional Logistic Regression for m:n Matching 7654
Example 92.6: Model Using Time-Dependent Explanatory Variables 7658
Example 92.7: Time-Dependent Repeated Measurements of a Covariate 7664
Example 92.8: Survival Curves 7671
Example 92.9: Analysis of Residuals 7678
Example 92.10: Analysis of Recurrent Events Data 7680
Example 92.11: Analysis of Clustered Data 7690
Example 92.12: Model Assessment Using Cumulative Sums of Martingale Residuals . 7695
Example 92.13: Bayesian Analysis of the Cox Model 7707
Example 92.14: Bayesian Analysis of Piecewise Exponential Model 7717

Example 92.15: Analysis of Competing-Risks Data 7722
Example 92.16: Concordance and ROC Curves 7730
References **7734**

Overview: PHREG Procedure

The analysis of survival data requires special techniques because the data are almost always incomplete and familiar parametric assumptions might be unjustifiable. Investigators follow subjects until they reach a prespecified endpoint (for example, death). However, subjects sometimes withdraw from a study, or the study is completed before the endpoint is reached. In these cases, the survival times (also known as failure times) are *censored*; subjects survived to a certain time beyond which their status is unknown. The uncensored survival times are sometimes referred to as *event* times. Methods of survival analysis must account for both censored and uncensored data.

Many types of models have been used for survival data. Two of the more popular types of models are the accelerated failure time model (Kalbfleisch and Prentice 1980) and the Cox proportional hazards model (Cox 1972). Each has its own assumptions about the underlying distribution of the survival times. Two closely related functions often used to describe the distribution of survival times are the survivor function and the hazard function. See the section "Failure Time Distribution" on page 7545 for definitions. The accelerated failure time model assumes a parametric form for the effects of the explanatory variables and usually assumes a parametric form for the underlying survivor function. The Cox proportional hazards model also assumes a parametric form for the effects of the explanatory variables, but it allows an unspecified form for the underlying survivor function.

The PHREG procedure performs regression analysis of survival data based on the Cox proportional hazards model. Cox's semiparametric model is widely used in the analysis of survival data to explain the effect of explanatory variables on hazard rates.

The survival time of each member of a population is assumed to follow its own hazard function, $\lambda_i(t)$, expressed as

$$\lambda_i(t) = \lambda(t; \mathbf{Z}_i) = \lambda_0(t) \exp(\mathbf{Z}_i' \boldsymbol{\beta})$$

where $\lambda_0(t)$ is an arbitrary and unspecified baseline hazard function, $\mathbf{Z}_i$ is the vector of explanatory variables for the ith individual, and $\boldsymbol{\beta}$ is the vector of unknown regression parameters that is associated with the explanatory variables. The vector $\boldsymbol{\beta}$ is assumed to be the same for all individuals. The survivor function can be expressed as

$$S(t; \mathbf{Z}_i) = [S_0(t)]^{\exp(\mathbf{Z}_i' \boldsymbol{\beta})}$$

where $S_0(t) = \exp(-\int_0^t \lambda_0(u)du)$ is the baseline survivor function. To estimate $\boldsymbol{\beta}$, Cox (1972, 1975) introduced the partial likelihood function, which eliminates the unknown baseline hazard $\lambda_0(t)$ and accounts for censored survival times.

The partial likelihood of Cox also allows time-dependent explanatory variables. An explanatory variable is time-dependent if its value for any given individual can change over time. Time-dependent variables have many useful applications in survival analysis. You can use a time-dependent variable to model the effect of subjects changing treatment groups. Or you can include time-dependent variables such as blood pressure or

blood chemistry measures that vary with time during the course of a study. You can also use time-dependent variables to test the validity of the proportional hazards model.

An alternative way to fit models with time-dependent explanatory variables is to use the counting process style of input. The counting process formulation enables PROC PHREG to fit a superset of the Cox model, known as the multiplicative hazards model. This extension also includes recurrent events data and left-truncation of failure times. The theory of these models is based on the counting process pioneered by Andersen and Gill (1982), and the model is often referred to as the Andersen-Gill model.

Multivariate failure-time data arise when each study subject can potentially experience several events (for example, multiple infections after surgery) or when there exists some natural or artificial clustering of subjects (for example, a litter of mice) that induces dependence among the failure times of the same cluster. Data in the former situation are referred to as multiple events data, which include recurrent events data as a special case; data in the latter situation are referred to as clustered data. You can use PROC PHREG to carry out various methods of analyzing these data.

The population under study can consist of a number of subpopulations, each of which has its own baseline hazard function. PROC PHREG performs a stratified analysis to adjust for such subpopulation differences. Under the stratified model, the hazard function for the *j*th individual in the *i*th stratum is expressed as

$$\lambda_{ij}(t) = \lambda_{i0}(t)\exp(\mathbf{Z}'_{ij}\boldsymbol{\beta})$$

where $\lambda_{i0}(t)$ is the baseline hazard function for the *i*th stratum and $\mathbf{Z}_{ij}$ is the vector of explanatory variables for the individual. The regression coefficients are assumed to be the same for all individuals across all strata.

Ties in the failure times can arise when the time scale is genuinely discrete or when survival times that are generated from the continuous-time model are grouped into coarser units. The PHREG procedure includes four methods of handling ties. The *discrete* logistic model is available for discrete time-scale data. The other three methods apply to continuous time-scale data. The *exact* method computes the exact conditional probability under the model that the set of observed tied event times occurs before all the censored times with the same value or before larger values. *Breslow* and *Efron* methods provide approximations to the exact method.

Variable selection is a typical exploratory exercise in multiple regression when the investigator is interested in identifying important prognostic factors from a large number of candidate variables. The PHREG procedure provides four selection methods: forward selection, backward elimination, stepwise selection, and best subset selection. The best subset selection method is based on the likelihood score statistic. This method identifies a specified number of best models that contain one, two, or three variables and so on, up to the single model that contains all of the explanatory variables.

The PHREG procedure also enables you to do the following: include an offset variable in the model; weight the observations in the input data; test linear hypotheses about the regression parameters; perform conditional logistic regression analysis for matched case-control studies; output survivor function estimates, residuals, and regression diagnostics; and estimate and plot the survivor function for a new set of covariates.

PROC PHREG can also be used to fit the multinomial logit choice model to discrete choice data. See `http://support.sas.com/resources/papers/tnote/tnote_marketresearch.html` for more information about discrete choice modeling and the multinomial logit model. Look for the "Discrete Choice" report.

The PHREG procedure uses ODS Graphics to create graphs as part of its output. For example, the ASSESS statement uses a graphical method that uses ODS Graphics to check the adequacy of the model. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

For both the BASELINE and OUTPUT statements, the default method of estimating a survivor function has changed to the Breslow (1972) estimator—that is, METHOD=CH. The option NOMEAN that was available in the BASELINE statement prior to SAS/STAT 9.2 has become obsolete—that is, requested statistics at the sample average values of the covariates are no longer computed and added to the OUT= data set. However, if the COVARIATES= data set is not specified, the requested statistics are computed and output for the covariate set that consists of the reference levels for the CLASS variables and sample averages for the continuous variable. In addition to the requested statistics, the OUT= data set also contains all variables in the COVARIATES= data set.

The remaining sections of this chapter contain information about how to use PROC PHREG, information about the underlying statistical methodology, and some sample applications of the procedure. The section "Getting Started: PHREG Procedure" on page 7467 introduces PROC PHREG with two examples. The section "Syntax: PHREG Procedure" on page 7475 describes the syntax of the procedure. The section "Details: PHREG Procedure" on page 7545 summarizes the statistical techniques used in PROC PHREG. The section "Examples: PHREG Procedure" on page 7638 includes 16 additional examples of useful applications. Experienced SAS/STAT software users might decide to proceed to the "Syntax" section, while other users might choose to read both the "Getting Started" and "Examples" sections before proceeding to "Syntax" and "Details."

Getting Started: PHREG Procedure

This section uses the two-sample vaginal cancer mortality data from Kalbfleisch and Prentice (1980, p. 2) in two examples to illustrate some of the basic features of PROC PHREG. The first example carries out a classical Cox regression analysis and the second example performs a Bayesian analysis of the Cox model.

Two groups of rats received different pretreatment regimes and then were exposed to a carcinogen. Investigators recorded the survival times of the rats from exposure to mortality from vaginal cancer. Four rats died of other causes, so their survival times are censored. Interest lies in whether the survival curves differ between the two groups.

The following DATA step creates the data set Rats, which contains the variable Days (the survival time in days), the variable Status (the censoring indicator variable: 0 if censored and 1 if not censored), and the variable Group (the pretreatment group indicator).

```
data Rats;
   label Days  ='Days from Exposure to Death';
   input Days Status Group @@;
   datalines;
143 1 0   164 1 0   188 1 0   188 1 0
190 1 0   192 1 0   206 1 0   209 1 0
213 1 0   216 1 0   220 1 0   227 1 0
230 1 0   234 1 0   246 1 0   265 1 0
304 1 0   216 0 0   244 0 0   142 1 1
156 1 1   163 1 1   198 1 1   205 1 1
232 1 1   232 1 1   233 1 1   233 1 1
233 1 1   233 1 1   239 1 1   240 1 1
261 1 1   280 1 1   280 1 1   296 1 1
296 1 1   323 1 1   204 0 1   344 0 1
;
```

By using ODS Graphics, PROC PHREG allows you to plot the survival curve for Group=0 and the survival curve for Group=1, but first you must save these two covariate values in a SAS data set as in the following DATA step:

```
data Regimes;
   Group=0;
   output;
   Group=1;
   output;
run;
```

Classical Method of Maximum Likelihood

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/phrgs1.sas.)

PROC PHREG fits the Cox model by maximizing the partial likelihood and computes the baseline survivor function by using the Breslow (1972) estimate. The following statements produce Figure 92.1 and Figure 92.2:

```
ods graphics on;
proc phreg data=Rats plot(overlay)=survival;
   model Days*Status(0)=Group;
   baseline covariates=regimes out=_null_;
run;
```

In the MODEL statement, the response variable, Days, is crossed with the censoring variable, Status, with the value that indicates censoring is enclosed in parentheses. The values of Days are considered censored if the value of Status is 0; otherwise, they are considered event times.

Graphs are produced when ODS Graphics is enabled. The survival curves for the two observations in the data set Regime, specified in the COVARIATES= option in the BASELINE statement, are requested through the PLOTS= option with the OVERLAY option for overlaying both survival curves in the same plot.

Figure 92.2 shows a typical printed output of a classical analysis. Since Group takes only two values, the null hypothesis for no difference between the two groups is identical to the null hypothesis that the regression coefficient for Group is 0. All three tests in the "Testing Global Null Hypothesis: BETA=0" table (see the section "Testing the Global Null Hypothesis" on page 7569) suggest that the survival curves for the two pretreatment groups might not be the same. In this model, the hazard ratio (or risk ratio) for Group, defined as the exponentiation of the regression coefficient for Group, is the ratio of the hazard functions between the two groups. The estimate is 0.551, implying that the hazard function for Group=1 is smaller than that for Group=0. In other words, rats in Group=1 lived longer than those in Group=0. This conclusion is also revealed in the plot of the survivor functions in Figure 92.2.

Figure 92.1 Comparison of Two Survival Curves

The PHREG Procedure

Model Information		
Data Set	WORK.RATS	
Dependent Variable	Days	Days from Exposure to Death
Censoring Variable	Status	
Censoring Value(s)	0	
Ties Handling	BRESLOW	

Number of Observations Read	40
Number of Observations Used	40

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
40	36	4	10.00

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Without Covariates	With Covariates
-2 LOG L	204.317	201.438
AIC	204.317	203.438
SBC	204.317	205.022

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	2.8784	1	0.0898
Score	3.0001	1	0.0833
Wald	2.9254	1	0.0872

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
Group	1	-0.59590	0.34840	2.9254	0.0872	0.551

Figure 92.2 Survivorship for the Two Pretreatment Regimes

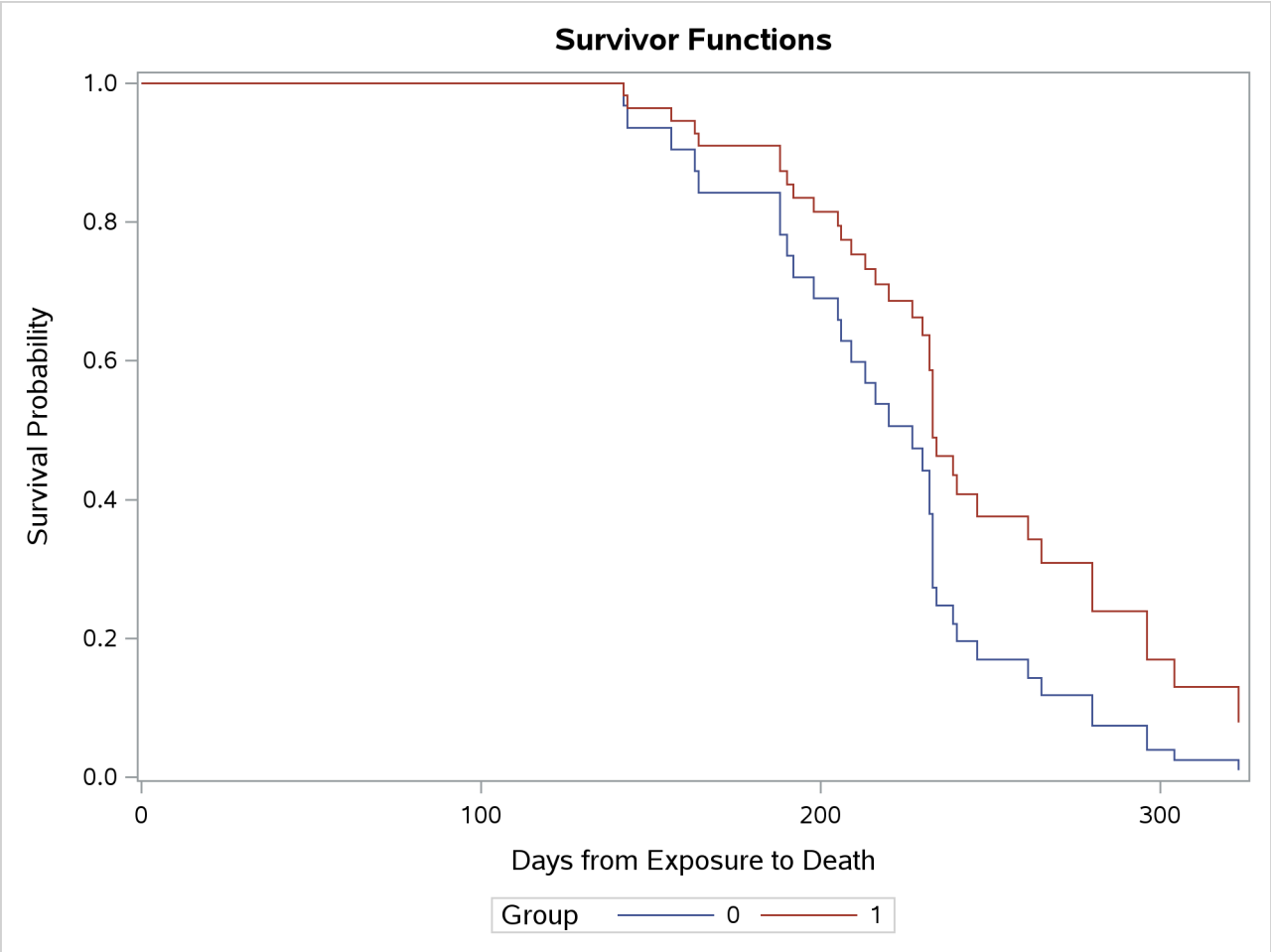

In this example, the comparison of two survival curves is put in the form of a proportional hazards model. This approach is essentially the same as the log-rank (Mantel-Haenszel) test. In fact, if there are no ties in the survival times, the likelihood score test in the Cox regression analysis is identical to the log-rank test. The advantage of the Cox regression approach is the ability to adjust for the other variables by including them in the model. For example, the present model could be expanded by including a variable that contains the initial body weights of the rats.

Next, consider a simple test of the validity of the proportional hazards assumption. The proportional hazards model for comparing the two pretreatment groups is given by the following:

$$\lambda(t) = \begin{cases} \lambda_0(t) & \text{if GROUP} = 0 \\ \lambda_0(t)e^{\beta_1} & \text{if GROUP} = 1 \end{cases}$$

The ratio of hazards is e^{β_1}, which does not depend on time. If the hazard ratio changes with time, the proportional hazards model assumption is invalid. Simple forms of departure from the proportional hazards model can be investigated with the following time-dependent explanatory variable $x = x(t)$:

$$x(t) = \begin{cases} 0 & \text{if GROUP} = 0 \\ \log(t) - 5.4 & \text{if GROUP} = 1 \end{cases}$$

Here, $\log(t)$ is used instead of t to avoid numerical instability in the computation. The constant, 5.4, is the average of the logs of the survival times and is included to improve interpretability. The hazard ratio in the two groups then becomes $e^{\beta_1 - 5.4\beta_2} t^{\beta_2}$, where β_2 is the regression parameter for the time-dependent variable x. The term e^{β_1} represents the hazard ratio at the geometric mean of the survival times. A nonzero value of β_2 would imply an increasing ($\beta_2 > 0$) or decreasing ($\beta_2 < 0$) trend in the hazard ratio with time.

The following statements implement this simple test of the proportional hazards assumption. The MODEL statement includes the time-dependent explanatory variable X, which is defined subsequently by the programming statement. At each event time, subjects in the risk set (those alive just before the event time) have their X values changed accordingly.

```
proc phreg data=Rats;
   model Days*Status(0)=Group X;
   X=Group*(log(Days) - 5.4);
run;
```

The analysis of the parameter estimates is displayed in Figure 92.3. The Wald chi-square statistic for testing the null hypothesis that $\beta_2 = 0$ is 0.0158. The statistic is not statistically significant when compared to a chi-square distribution with one degree of freedom ($p = 0.8999$). Thus, you can conclude that there is no evidence of an increasing or decreasing trend over time in the hazard ratio.

Figure 92.3 A Simple Test of Trend in the Hazard Ratio

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
Group	1	-0.59976	0.34837	2.9639	0.0851	0.549
X	1	-0.22952	1.82489	0.0158	0.8999	0.795

Bayesian Analysis

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/phrgs2.sas.)

PROC PHREG uses the partial likelihood of the Cox model as the likelihood and generates a chain of posterior distribution samples by the Gibbs Sampler. Summary statistics, convergence diagnostics, and diagnostic plots are provided for each parameter. The following statements generate Figure 92.4–Figure 92.10:

```
ods graphics on;
proc phreg data=Rats;
   model Days*Status(0)=Group;
   bayes seed=1 outpost=Post;
run;
```

The BAYES statement invokes the Bayesian analysis. The SEED= option is specified to maintain reproducibility; the OUTPOST= option saves the posterior distribution samples in a SAS data set for post-processing; no other options are specified in the BAYES statement. By default, a uniform prior distribution is assumed on the regression coefficient Group. The uniform prior is a flat prior on the real line with a distribution that reflects ignorance of the location of the parameter, placing equal probability on all possible values the regression

coefficient can take. Using the uniform prior in the following example, you would expect the Bayesian estimates to resemble the classical results of maximizing the likelihood. If you can elicit an informative prior on the regression coefficients, you should use the COEFFPRIOR= option to specify it.

You should make sure that the posterior distribution samples have achieved convergence before using them for Bayesian inference. PROC PHREG produces three convergence diagnostics by default. If ODS Graphics is enabled before calling PROC PHREG as in the preceding program, diagnostics plots are also displayed.

The results of this analysis are shown in the following figures.

The "Model Information" table in Figure 92.4 summarizes information about the model you fit and the size of the simulation.

Figure 92.4 Model Information

The PHREG Procedure

Bayesian Analysis

Model Information		
Data Set	WORK.RATS	
Dependent Variable	Days	Days from Exposure to Death
Censoring Variable	Status	
Censoring Value(s)	0	
Model	Cox	
Ties Handling	BRESLOW	
Sampling Algorithm	ARMS	
Burn-In Size	2000	
MC Sample Size	10000	
Thinning	1	

PROC PHREG first fits the Cox model by maximizing the partial likelihood. The only parameter in the model is the regression coefficient of Group. The maximum likelihood estimate (MLE) of the parameter and its 95% confidence interval are shown in Figure 92.5.

Figure 92.5 Classical Parameter Estimates

Maximum Likelihood Estimates					
Parameter	**DF**	**Estimate**	**Standard Error**	**95% Confidence Limits**	
Group	1	-0.5959	0.3484	-1.2788	0.0870

Since no prior is specified for the regression coefficient, the default uniform prior is used. This information is displayed in the "Uniform Prior for Regression Coefficients" table in Figure 92.6.

Figure 92.6 Coefficient Prior

Uniform Prior for Regression Coefficients	
Parameter	**Prior**
Group	Constant

The "Fit Statistics" table in Figure 92.7 lists information about the fitted model. The table displays the DIC (deviance information criterion) and pD (effective number of parameters). For more information, see the section "Fit Statistics" on page 7619.

Figure 92.7 Fit Statistics

Fit Statistics	
DIC (smaller is better)	203.444
pD (Effective Number of Parameters)	1.003

Summary statistics of the posterior samples are displayed in the "Posterior Summaries and Intervals" table in Figure 92.8. Note that the mean and standard deviation of the posterior samples are comparable to the MLE and its standard error, respectively, because of the use of the uniform prior.

Figure 92.8 Summary Statistics

The PHREG Procedure

Bayesian Analysis

Posterior Summaries and Intervals					
Parameter	N	Mean	Standard Deviation	95% HPD Interval	
Group	10000	-0.5998	0.3511	-1.2984	0.0756

PROC PHREG provides diagnostics to assess the convergence of the generated Markov chain. Figure 92.9 shows the effective sample size diagnostic. There is no indication that the Markov chain has not reached convergence. For information about interpreting these diagnostics, see the section "Statistical Diagnostic Tests" on page 166 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Figure 92.9 Convergence Diagnostics

The PHREG Procedure

Bayesian Analysis

Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
Group	10000.0	1.0000	1.0000

You can also assess the convergence of the generated Markov chain by examining the trace plot, the autocorrelation function plot, and the posterior density plot. Figure 92.10 displays a panel of these three plots for the parameter Group. This graphical display is automatically produced when ODS Graphics is enabled. Note that the trace of the samples centers on –0.6 with only small fluctuations, the autocorrelations are quite small, and the posterior density appears bell-shaped—all exemplifying the behavior of a converged Markov chain.

Figure 92.10 Diagnostic Plots

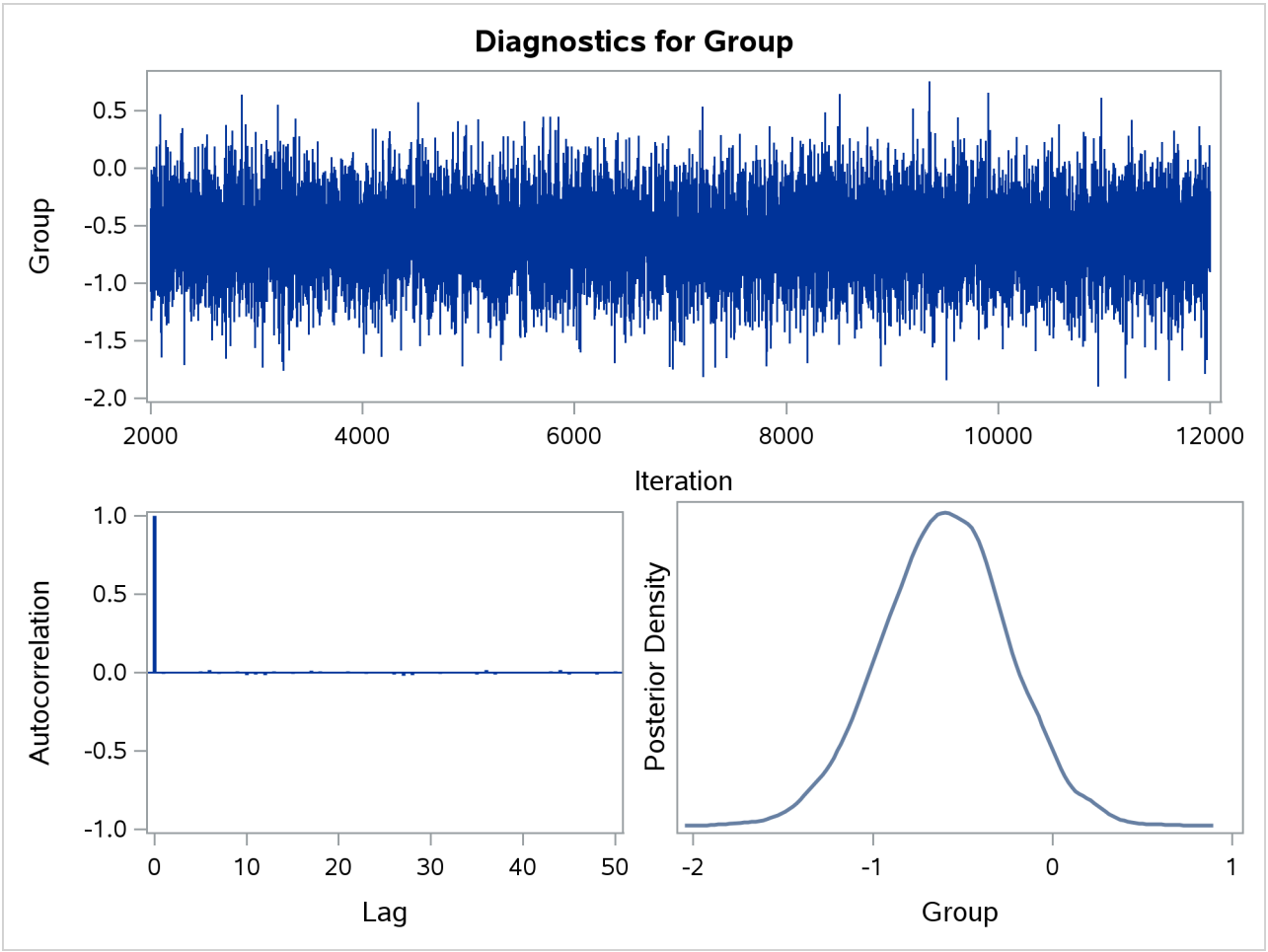

The proportional hazards model for comparing the two pretreatment groups is

$$\lambda(t) = \begin{cases} \lambda_0(t) & \text{if Group=0} \\ \lambda_0(t)\mathrm{e}^{\beta} & \text{if Group=1} \end{cases}$$

The probability that the hazard of Group=0 is greater than that of Group=1 is

$$\Pr(\lambda_0(t) > \lambda_0(t)\mathrm{e}^{\beta}) = \Pr(\beta < 0)$$

This probability can be enumerated from the posterior distribution samples by computing the fraction of samples with a coefficient less than 0. The following DATA step and PROC MEANS perform this calculation:

```
data New;
   set Post;
   Indicator=(Group < 0);
   label Indicator='Group < 0';
run;
proc means data=New(keep=Indicator) n mean;
run;
```

Figure 92.11 Prob(Hazard(Group=0) > Hazard(Group=1))

The MEANS Procedure

Analysis Variable : Indicator Group < 0	
N	Mean
10000	0.9581000

The PROC MEANS results are displayed in Figure 92.11. There is a 95.8% chance that the hazard rate of Group=0 is greater than that of Group=1. The result is consistent with the fact that the average survival time of Group=0 is less than that of Group=1.

Syntax: PHREG Procedure

The following statements are available in the PHREG procedure. Items within < > are optional.

```
PROC PHREG <options> ;
    ASSESS keyword </ options> ;
    BASELINE <OUT=SAS-data-set> <COVARIATES=SAS-data-set>
                <keyword=name ... keyword=name> </ options> ;
    BAYES <options> ;
    BY variables ;
    CLASS variable <(options)> <... variable <(options)>> </ options> ;
    CONTRAST <'label'> effect values <, ..., effect values> </ options> ;
    FREQ variable ;
    EFFECT name = effect-type (variables </ options>) ;
    ESTIMATE <'label'> estimate-specification </ options> ;
    HAZARDRATIO <'label'> variable </ options> ;
    ID variables ;
    LSMEANS <model-effects> </ options> ;
    LSMESTIMATE model-effect lsmestimate-specification </ options> ;
    MODEL response <* censor(list)> = <effects> </ options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=name ... keyword=name> </ options> ;
    Programming statements ;
    RANDOM variable </ options> ;
    ROC <'label'> specification ;
    SLICE model-effect </ options> ;
    STORE <OUT=> item-store-name </ LABEL='label'> ;
    STRATA variable <(list)> <... variable <(list)>> </ option> ;
    <label:> TEST equation <, ..., equation> </ options> ;
    WEIGHT variable </ option> ;
```

The PROC PHREG and MODEL statements are required. The CLASS statement, if present, must precede the MODEL statement, and the ASSESS or CONTRAST statement, if present, must come after the MODEL statement. The BAYES statement, that invokes a Bayesian analysis, is not compatible with the ASSESS,

CONTRAST, ID, OUTPUT, and TEST statements, as well as a number of options in the PROC PHREG and MODEL statements. For more information, see the section "Specifics for Bayesian Analysis" on page 7610.

The rest of this section provides detailed syntax information for each statement, beginning with the PROC PHREG statement. The remaining statements are covered in alphabetical order.

PROC PHREG Statement

PROC PHREG < *options* > **;**

The PROC PHREG statement invokes the PHREG procedure. Table 92.1 summarizes the *options* available in the PROC PHREG statement.

Table 92.1 PROC PHREG Statement Options

Option	Description
ALPHA=	Specifies the level of significance
ATRISK	Displays a table that contains the number of units and the corresponding number of events in the risk sets
CONCORDANCE	Computes concordance statistics
COVM	Uses the model-based covariance matrix in the analysis
COVOUT	Adds the estimated covariance matrix to the OUTEST= data set
COVSANDWICH	Requests the robust sandwich estimate for the covariance matrix
DATA=	Names the SAS data set to be analyzed
EV	Requests the Schemper-Henderson predictive measures
FAST	Uses a fast algorithm for large data with start/stop input
INEST=	Names the SAS data set that contains initial estimates
MULTIPASS	Recompiles the risk sets
NAMELEN=	Specifies the length of effect names
NOPRINT	Suppresses all displayed output
NOSUMMARY	Suppresses the summary display observation frequencies
OUTEST=	Creates an output SAS data set containing estimates of the regression coefficients
PLOTS=	Controls the plots that are produced through ODS Graphics
ROCOPTIONS	Specifies options for receiver operating characteristic analysis
SIMPLE	Displays simple descriptive statistics
TAU=	Specifies upper time limit for Uno's concordance statistic and the integrated area under the curve
ZPH	Requests diagnostics based on weighted residuals for checking the proportional hazards assumption

You can specify the following *options* in the PROC PHREG statement.

ALPHA=*number*

specifies the level of significance α for $100(1-\alpha)\%$ confidence intervals. The value *number* must be between 0 and 1; the default value is 0.05, which results in 95% intervals. This value is used as the default confidence level for limits computed by the BASELINE, BAYES, CONTRAST, HAZARDRATIO, and MODEL statements. You can override this default by specifying the ALPHA= option in the separate statements.

ATRISK

displays a table that contains the number of units at risk at each event time and the corresponding number of events in the risk sets. For example, the following risk set information is displayed if the ATRISK option is specified in the example in the section "Getting Started: PHREG Procedure" on page 7467.

Risk Set Information		
	Number of Units	
Days	At Risk	Event
142	40	1
143	39	1
156	38	1
⋮	⋮	⋮
296	5	2
304	3	1
323	2	1

CONCORDANCE < =*method*< (*options*) > >

computes concordance statistics for the model specified in the MODEL statement (unless the NOFIT option is specified) and for each model specified in an ROC statement. For more information, see the section "Concordance Statistics" on page 7589. You can specify the following *methods*:

HARRELL < (SE) >

requests Harrell's concordance statistic (Harrell et al. 1984). The SE suboption, if specified, produces standard error for the concordance statistic by the method of Kang et al. (2015).

UNO < (*uno-options***) >**

computes the concordance statistic of Uno et al. (2011). You can specify the following *uno-options*:

SE

computes the standard error for the concordance statistic by using the perturbation resampling approach as described in Uno et al. (2011).

SEED=*n*

specifies an integer seed for the random number generator to generate the perturbation samples.

ITER=*n*
specifies the number of perturbation samples to use.

ALPHA=*number*
specifies the significance level of the confidence interval for the concordance probability.

DIFF
calculates paired differences in the estimated concordance statistics among all identified ROC models. If the SE suboption is also specified, standard errors and confidence limits are also computed.

Specifying the CONCORDANCE option without any suboptions is equivalent to specifying CONCORDANCE=HARRELL.

COVOUT
adds the estimated covariance matrix of the parameter estimates to the OUTEST= data set. The COVOUT option has no effect unless the OUTEST= option is specified.

COVM
requests that the model-based covariance matrix (which is the inverse of the observed information matrix) be used in the analysis if the COVS option is also specified. The COVM option has no effect if the COVS option is not specified.

COVSANDWICH < (AGGREGATE) >
COVS < (AGGREGATE) >
requests the robust sandwich estimate of Lin and Wei (1989) for the covariance matrix. When this option is specified, this robust sandwich estimate is used in the Wald tests for testing the global null hypothesis, null hypotheses of individual parameters, and the hypotheses in the CONTRAST and TEST statements. In addition, a modified score test is computed in the testing of the global null hypothesis, and the parameter estimates table has an additional StdErrRatio column, which contains the ratios of the robust estimate of the standard error relative to the corresponding model-based estimate. Optionally, you can specify the keyword AGGREGATE enclosed in parentheses after the COVSANDWICH (or COVS) option, which requests a summing up of the score residuals for each distinct ID pattern in the computation of the robust sandwich covariance estimate. This AGGREGATE option has no effect if the ID statement is not specified.

DATA=*SAS-data-set*
names the SAS data set that contains the data to be analyzed. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

EV
requests the Schemper-Henderson measure (Schemper and Henderson 2000) of the proportion of variation that is explained by a Cox regression. This measure of explained variation (EV) is the ratio of distance measures between the 1/0 survival processes and the fitted survival curves with and without covariates information. The distance measure is referred to as the predictive inaccuracy, because the smaller the predictive inaccuracy, the better the prediction. When you specify this option, PROC PHREG creates a table that has three columns: one presents the predictive inaccuracy without covariates (D); one presents the predictive inaccuracy with covariates (Dz); and one presents the EV measure, computed according to $100\frac{D-Dz}{D}\%$.

FAST

uses an alternative algorithm to speed up the fitting of the Cox regression for a large data set that has the counting process style of input. Simonsen (2014) has demonstrated the efficiency of this algorithm when the data set contains a large number of observations and many distinct event times. The algorithm requires only one pass through the data to compute the Breslow or Efron partial log-likelihood function and the corresponding gradient and Hessian. PROC PHREG ignores the FAST option if you specify a TIES= option value other than BRESLOW or EFRON, or if you specify programming statements for time-varying covariates. You might not see much improvement in the optimization time if your data set has only a moderate number of observations.

INEST=*SAS-data-set*

names the SAS data set that contains initial estimates for all the parameters in the model. BY-group processing is allowed in setting up the INEST= data set. For more information, see the section "INEST= Input Data Set" on page 7622.

MULTIPASS

requests that, for each Newton-Raphson iteration, PROC PHREG recompile the risk sets that correspond to the event times for the (start,stop) style of response and recomputes the values of the time-dependent variables defined by the programming statements for each observation in the risk sets. If the MULTIPASS option is not specified, PROC PHREG computes all risk sets and all the variable values and saves them in a utility file. The MULTIPASS option decreases required disk space at the expense of increased execution time; however, for very large data, it might actually save time, because it is time-consuming to write and read large utility files. This option has an effect only when the (start,stop) style of response is used or when there are time-dependent explanatory variables.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 200. The default length is 20 characters.

NOPRINT

suppresses all displayed output. Note that this option temporarily disables the Output Delivery System (ODS); for more information about ODS, see Chapter 23, "Using the Output Delivery System."

NOSUMMARY

suppresses the summary display of the event and censored observation frequencies.

OUTEST=*SAS-data-set*

creates an output SAS data set that contains estimates of the regression coefficients. The data set also contains the convergence status and the log likelihood. If you use the COVOUT option, the data set also contains the estimated covariance matrix of the parameter estimators. For more information, see the section "OUTEST= Output Data Set" on page 7621.

PLOTS< (*global-plot-options***) > = *plot-request***

PLOTS< (*global-plot-options***) > = (***plot-request* < . . . < *plot-request* > >**)**

controls the plots that are produced through ODS Graphics. Two types of plots can be produced: plots related to the receiver operating characteristic curves (such as AUC, AUCDIFF, and ROC) and the baseline function plots (such as CIF, CUMHAZ, MCF, and SURVIVAL).

For the baseline function plots, each observation in the COVARIATES= data set in the BASELINE statement represents a set of covariates for which a curve is produced for each *plot-request* and for

each stratum. You can use the ROWID= option in the BASELINE statement to specify a variable in the COVARIATES= data set for identifying the curves that are produced for the covariate sets. If the ROWID= option is not specified, the produced curves are identified by the covariate values if there is only a single covariate or by the observation numbers of the COVARIATES= data set if the model has two or more covariates. If the COVARIATES= data set is not specified, a reference set of covariates that consists of the reference levels for the CLASS variables and the average values for the continuous variables is used. For plotting more than one curve, you can use the OVERLAY= *global-plot-option* to group the curves in separate plots.

When you specify one *plot-request*, you can omit the parentheses around the plot request. Here are some examples:

```
plots=survival
plots=(survival auc)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc phreg plots(cl)=survival;
   model Time*Status(0)=X1-X5;
   baseline covariates=One;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The *global-plot-options* include the following:

CL< =EQTAIL | HPD >

displays the pointwise interval limits for the specified curves. For the classical analysis, CL displays the confidence limits. For the Bayesian analysis, CL=EQTAIL displays the equal-tail credible limits and CL=HPD displays the HPD limits. Specifying just CL in a Bayesian analysis defaults to CL=HPD.

OVERLAY < =*overlay-option* >

specifies how the curves for the various strata and covariate sets are overlaid. If the STRATA statement is not specified, specifying OVERLAY without any option will overlay the curves for all the covariate sets. The available *overlay-options* are as follows:

BYGROUP

GROUP

overlays, for each stratum, all curves for the covariate sets that have the same GROUP= value in the COVARIATES= data set in the same plot.

INDIVIDUAL

IND

displays, for each stratum, a separate plot for each covariate set.

BYROW

ROW

displays, for each covariate set, a separate plot containing the curves for all the strata.

BYSTRATUM

STRATUM

displays, for each stratum, a separate plot containing the curves for all sets of covariates.

The default is OVERLAY=BYGROUP if the GROUP= option is specified in the BASELINE statement or if the COVARIATES= data set contains the _GROUP_ variable; otherwise the default is OVERLAY=INDIVIDUAL.

TIMERANGE=(< *min*> < *,max*>**)**

TIMERANGE=< *min*> < *,max*>

RANGE=(< *min*> < *,max*>**)**

RANGE=< *min*> < *,max*>

specifies the range of values on the time axis to clip the display. The *min* and *max* values are the lower and upper bounds of the range. By default, *min* is 0 and *max* is the largest event time.

You can specify the following *plot-requests*:

AUC

plots the time-dependent area under the curve (AUC) for each identified model. For a particular time *t*, the AUC is the area under the receiver operating characteristic (ROC) curve at *t*. This option is ignored if time points are specified in the AT= *suboption* in the ROCOPTIONS *option*.

AUCDIFF

plots the difference between two AUC curves for each pair of identified models. This option is ignored if time points are specified in the AT= *suboption* in the ROCOPTIONS *option*.

CIF

plots the estimated cumulative incidence function (CIF) for each set of covariates in the COVARIATES= data set in the BASELINE statement. If the COVARIATES= data set is not specified, the estimated CIF is plotted for the reference set of covariates, which consists of reference levels for the CLASS variables and average values for the continuous variables.

CUMHAZ

plots the estimated cumulative hazard function for each set of covariates in the COVARIATES= data set in the BASELINE statement. If the COVARIATES= data set is not specified, the estimated cumulative hazard function is plotted for the reference set of covariates, which consists of reference levels for the CLASS variables and average values for the continuous variables.

MCF

plots the estimated mean cumulative function for each set of covariates in the COVARIATES= data set in the BASELINE statement. If the COVARIATES= data set is not specified, the estimated mean cumulative function is plotted for the reference set of covariates, which consists of reference levels for the CLASS variables and average values for the continuous variables.

NONE
suppresses all the plots in the procedure. Specifying this option is equivalent to disabling ODS Graphics for the entire procedure.

ROC< (TICK) >
plots the time-dependent receiver operating characteristic (ROC) curves. You must specify the time points at which the ROC curves are calculated by using the AT= suboption in the ROCOPTIONS option. A plot is produced for each time point. Grid lines are displayed for both axes at 0.0, 0.2, 0.4, 0.6, 0.8, and 1.0. By default, the plots are displayed in panels, with each panel containing up to six plots. Tick values are not shown in the panel plots unless you specify the keyword TICK to show the tick values at the grid lines. You can specify the OVERLAY=INDIVIDUAL *global-plot-option* to display each plot individually.

SURVIVAL
plots the estimated survivor function for each set of covariates in the COVARIATES= data set in the BASELINE statement. If COVARIATES= data set is not specified, the estimated survivor function is plotted for the reference set of covariates, which consists of reference levels for the CLASS variables and average values for the continuous variables.

ROCOPTIONS (*options*)
specifies *options* that apply to the analysis of receiver operating characteristic (ROC) curves for the model that is specified in the MODEL statement (unless the NOFIT option is specified) and for each model specified in a ROC statement. You can specify the following *options*:

AT=*number-list*
specifies the list of time points at which the ROC curves are calculated. The *number-list* can be a list of numbers separated by blanks, or of the form 10 to 30 by 5, or a combination of both.

AUC
displays the area under the curve at time points specified in the AT= *suboption* or at the event times if the AT= *suboption* is not specified.

AUCDIFF
displays the differences between the AUC functions for each pair of identified models at the time points specified in the AT= *suboption* or at the event times if no time points are specified.

IAUC
displays the integrated area under the curve (IAUC), computed as a weighted average of the AUC values at all the event times if the TAU= option is not specified or at the event times less than or equal to the value specified in the TAU= option. The weights that are used are jumps of the Kaplan-Meier estimate of the survivor function.

METHOD=*method* < *(options)* >
specifies the method to calculate ROC curves and AUC statistics. For more information, see the section "Time-Dependent ROC Curves" on page 7591. You can specify the following *methods* along with any applicable *options*:

IPCW< *(ipcw-options)* >

UNO< *(ipcw-options)* >

uses the inverse probability of censoring weighting (IPCW) technique of Uno et al. (2007). Event observations are weighted according to their probabilities of being censored. The following *ipcw-options* apply only to AUC calculations:

ALPHA=*value*

specifies the significance level of the confidence interval for the AUC.

CL

computes the pointwise confidence limits for the AUC based on perturbation resampling.

ITER=*n*

specifies the number of perturbation samples.

SEED=*n*

specifies an integer seed for the random number generator to generate perturbation samples.

KM

uses the conditional Kaplan-Meier method of Heagerty, Lumley, and Pepe (2000).

NNE < *(nne-options)* >

uses the nearest-neighbors technique of Heagerty, Lumley, and Pepe (2000). You can specify the following *nne-options*:

ASYM

uses asymmetric kernels in the estimation.

SPAN=*value*

specifies the proportion of observations to use in deriving the bandwidth for the kernel-based estimation. By default, SPAN=0.05.

RECURSIVE

CHAMBLESS

uses the method of Chambless and Diao (2006). The calculation is recursive, and it is performed on the ordered event times sequentially from the smallest to the largest.

WKKM < *(wkkm-options)* >

uses the weighted kernel Kaplan-Meier method of Li, Greene, and Hu (2018). You can specify the following *wkkm-options*:

ASYM

uses asymmetric kernels in the estimation.

SPAN=*value*

specifies the proportion of observations to use in deriving the bandwidth for the kernel-based estimation. By default, SPAN=0.05.

By default, METHOD=NNE.

OUTAUC=*SAS-data-set*

names the output data set to contain the data necessary to produce the AUC plot. Each curve is identified by the corresponding model label. For the list of variables in this data set, see the section "OUTAUC= Output Data Set in the ROCOPTIONS Option" on page 7623.

OUTROC=*SAS-data-set*

names the output data set to contain the data necessary to produce the ROC plots. Each ROC curve corresponds to a block of observations that are identified by a time point and a model label. For the list of variables in this data set, see the section "OUTROC= Output Data Set in the ROCOPTIONS Option" on page 7623.

SIMPLE

displays simple descriptive statistics (mean, standard deviation, minimum, and maximum) for each explanatory variable in the MODEL statement.

TAU=*value*

specifies the upper time limit for computing the integrated area under the curve (IAUC) statistic and Uno's concordance statistic. Only event times that do not exceed the specified *value* are used in the calculation. The default *value* is the largest event time.

ZPH< (*zph-options***) >**

requests diagnostics based on the weighted Schoenfeld residuals for checking the proportional hazards assumption (for more information, see the section "ZPH Diagnostics" on page 7587). For each predictor, PROC PHREG presents a plot of the time-varying coefficients in addition to a correlation test between the weighted residuals and failure times in a given scale. You can specify the following *zph-options*:

FIT=NONE | LOESS | SPLINE

displays a fitted smooth curve in a plot of time-varying coefficients. FIT=LOESS displays a loess curve. FIT=SPLINE fits a penalized B-spline curve. If you do not want to display a fitted curve, specify FIT=NONE. By default, FIT=SPLINE.

GLOBAL

computes the global correlation test.

NOPLOT

suppresses the plots of the time-varying coefficients $\beta(t)$.

NOTEST

suppresses the correlation tests.

OUT=*SAS-data-set*

names the output data set that contains the time-varying coefficients $\boldsymbol{\beta}(t)$, one row per event time. The variables that contain $\boldsymbol{\beta}(t)$ have the same names as the predictors. The data set also contains the transformed event times $g(t)$.

TRANSFORM=IDENTITY | KM | LOG | RANK

specifies how the failure times should be transformed in the diagnostic plots and correlation tests. You can choose from the following transformations:

IDENTITY	specify the identity transformation, $g(t) = t$.
KM	specifies the complement of the Kaplan-Meier estimate transformation, $g(t) = 1 - \mathrm{KM}(t)$.
LOG	specifies the log transformation, $g(t) = \log(t)$.
RANK	specifies the rank transformation, $g(t) = \mathrm{rank}(t)$.

By default, TRANSFORM=RANK.

ASSESS Statement

ASSESS < **VAR=(***list***)** > < **PH** > < / *options* > **;**

The ASSESS statement performs the graphical and numerical methods of Lin, Wei, and Ying (1993) for checking the adequacy of the Cox regression model. The methods are derived from cumulative sums of martingale residuals over follow-up times or covariate values. You can assess the functional form of a covariate or you can check the proportional hazards assumption for each covariate in the Cox model. PROC PHREG uses ODS Graphics for the graphical displays. You must specify at least one of the following *options* to create an analysis.

VAR=(*variable-list***)**

specifies the list of explanatory variables for which their functional forms are assessed. For each variable on the list, the observed cumulative martingale residuals are plotted against the values of the explanatory variable along with 20 (or *n* if NPATHS=*n* is specified) simulated residual patterns.

PROPORTIONALHAZARDS

PH

requests the checking of the proportional hazards assumption. For each explanatory variable in the model, the observed score process component is plotted against the follow-up time along with 20 (or *n* if NPATHS=*n* is specified) simulated patterns.

The following *options* can be specified after a slash (/):

NPATHS=*n*

specifies the number of simulated residual patterns to be displayed in a cumulative martingale residual plot or a score process plot. The default is *n*=20.

CRPANEL

requests that a plot with four panels, each containing the observed cumulative martingale residuals and two simulated residual patterns, be created.

RESAMPLE <=*n*>
requests that the Kolmogorov-type supremum test be computed on 1,000 simulated patterns or on *n* simulated patterns if *n* is specified.

SEED=*n*
specifies an integer seed for the random number generator used in creating simulated realizations for plots and for the Kolmogorov-type supremum tests. Specifying a seed enables you to reproduce identical graphs and *p*-values for the model assessments from the same PHREG specification. If you omit the SEED= option, or if you specify a nonpositive seed, a random seed is derived from the time of day on the computer's clock.

BASELINE Statement

BASELINE <**OUT=***SAS-data-set*> <**OUTDIFF=***SAS-data-set*> <**COVARIATES=***SAS-data-set*> <**TIMELIST=***list*> <*keyword*=*name* . . . *keyword*=*name*> </ *options*> ;

The BASELINE statement creates a SAS data set (named by the OUT= option) that contains the baseline function estimates at the event times of each stratum for every set of covariates in the COVARIATES= data set. If the COVARIATES= data set is not specified, a reference set of covariates consisting of the reference levels for the CLASS variables and the average values for the continuous variables is used. You can use the DIRADJ option to obtain the direct adjusted survival curve that averages the estimated survival curves for the observations in the COVARIATES= data set. No BASELINE data set is created if the model contains a time-dependent variable defined by means of programming statement. Table 92.2 summarizes the *options* available in the BASELINE statement.

Table 92.2 BASELINE Statement Options

Option	Description
Data Set and Time Points Options	
OUT=	Specifies the output BASELINE data set
OUTDIFF=	Specifies the output data set that contains differences of direct adjusted survival curves
COVARIATES=	Specifies the SAS data set that contains the explanatory variables
TIMELIST=	Specifies a list of time points at which the requested statistics are output
Keyword Options	
CIF=	Specifies the cumulative incidence estimate
CMF=	Specifies the cumulative mean function estimate
CUMHAZ=	Specifies the cumulative hazard function estimate
LOGLOGS=	Specifies the log of the negative log of SURVIVAL
LOGSURV=	Specifies the log of SURVIVAL
LOWERCIF=	Specifies the lower pointwise confidence limit for CIF
LOWERCMF=	Specifies the lower pointwise confidence limit for CMF
LOWERCUMHAZ=	Specifies the lower pointwise confidence limit for CUMHAZ
LOWERHPDCUMHAZ=	Specifies the lower limit of the HPD interval for CUMHAZ
LOWERHPD=	Specifies the lower limit of the HPD interval for SURVIVAL
LOWER=	Specifies the lower pointwise confidence limit for SURVIVAL

Table 92.2 *continued*

Options	Description
STDCIF=	Specifies the estimated standard error of CIF
STDCMF=	Specifies the estimated standard error of CMF
STDCUMHAZ=	Specifies the estimated standard error of CUMHAZ
STDERR=	Specifies the standard error of SURVIVAL
STDXBETA=	Specifies the estimated standard error of the linear predictor estimator
SURVIVAL=	Specifies the survivor function estimate
UPPERCIF=	Specifies the upper pointwise confidence limit for CIF
UPPERCMF=	Specifies the upper pointwise confidence limit for CMF
UPPERCUMHAZ=	Specifies the upper pointwise confidence limit for CUMHAZ
UPPERHPDCUMHAZ=	Specifies the upper limit of the HPD interval for CUMHAZ
UPPERHPD=	Specifies the upper limit of the HPD interval for SURVIVAL
UPPER=	Specifies the upper pointwise confidence limit for SURVIVAL
XBETA=	Specifies the estimate of the linear predictor $\mathbf{x}'\boldsymbol{\beta}$
Other Options	
ALPHA=	Specifies the significance level of the confidence interval for the survivor function
CLTYPE=	Specifies the transformation used to compute the confidence limits
DIRADJ	Computes direct adjusted survival curves
GROUP=	Names a variable whose values are used to identify or group the survival curves
METHOD=	Specifies the method used to compute the survivor function estimates
NORMALSAMPLE=	Specifies the number of normal random samples for CIF confidence limits
ROWID=	Names the variable in the COVARIATES= data set for identifying the baseline functions curves in the plots
SEED=	Specifies the random number generator seed

The following *options* are available in the BASELINE statement.

OUT=*SAS-data-set*

names the output BASELINE data set. If you omit the OUT= option, the data set is created and given a default name by using the DATA*n* convention. For more information, see the section "OUT= Output Data Set in the BASELINE Statement" on page 7622.

OUTDIFF=*SAS-data-set*

names the output data set that contains all pairwise differences of direct adjusted probabilities between groups if the GROUP= variable is specified, or between strata if the GROUP= variable is not specified. It is required that the DIRADJ option be specified to use the OUTDIFF= option.

COVARIATES=*SAS-data-set*

names the SAS data set that contains the sets of explanatory variable values for which the quantities of interest are estimated. All variables in the COVARIATES= data set are copied to the OUT= data set. Thus, any variable in the COVARIATES= data set can be used to identify the covariate sets in the OUT= data set.

TIMELIST=*list*

TIMEPOINT=*list*

specifies a *list* of time points at which the requested prediction statistics are output to the OUT= and the OUTDIFF= data sets. The prediction statistics in the OUT= data set include the survivor function estimates, the direct adjusted survivor function estimates, the cumulative hazard function estimates, and the cumulative incidence function estimates. If the *list* contains time points that are greater than the largest observed time, the requested prediction statistics might not be defined for these time points and are not output. The following specifications are equivalent:

```
timelist=5,20 to 50 by 10
timelist=5 20 30 40 50
```

If the TIMELIST= option is not specified, the OUT= and the OUTDIFF= data sets include the requested prediction statistics at all event times.

keyword=name

specifies the statistics to be included in the OUT= data set and assigns names to the variables that contain these statistics. Specify a *keyword* for each desired statistic, an equal sign, and the name of the variable for the statistic. Not all *keywords* listed in Table 92.3 (and discussed in the text that follows) are appropriate for both the classical analysis and the Bayesian analysis; and the table summaries the choices for each analysis.

Table 92.3 Summary of the Keyword Choices

Keyword	Classical	Bayesian
Survivor Function		
SURVIVAL=	x	x
STDERR=	x	x
LOWER=	x	x
UPPER=	x	x
LOWERHPD=		x
UPPERHPD=		x
Cumulative Hazard Function		
CUMHAZ=	x	x
STDCUMHAZ=	x	x
LOWERCUMHAZ=	x	x
UPPERCUMHAZ=	x	x
LOWERHPDCUMHAZ=		x
UPPERHPDCUMHAZ=		x

Table 92.3 *continued*

Options	Classical	Bayesian
Cumulative Incidence Function		
CIF=	x	
STDCIF=	x	
LOWERCIF=	x	
UPPERCIF=	x	
Cumulative Mean Function		
CMF=	x	
STDCMF=	x	
LOWERCMF=	x	
UPPERCMF=	x	
Others		
XBETA=	x	x
STDXBETA=	x	x
LOGSURV=	x	
LOGLOGS=	x	

You can specify the following *keywords*:

CIF=

specifies the cumulative incidence function estimate for competing risks data. Specifying CIF=_ALL_ is equivalent to specifying CIF=CIF, STDCIF=StdErrCIF, LOWERCIF=LowerCIF, and UPPERCIF=UpperCIF.

CMF=

MCF=

specifies the cumulative mean function estimate for recurrent events data. Specifying CMF=_ALL_ is equivalent to specifying CMF=CMF, STDCMF=StdErrCMF, LOWERCMF=LowerCMF, and UPPERCMF=UpperCMF. Nelson (2003) refers to the mean function estimate as MCF (mean cumulative function).

CUMHAZ=

specifies the cumulative hazard function estimate. Specifying CUMHAZ=_ALL_ is equivalent to specifying CUMHAZ=CumHaz, STDCUMHAZ=StdErrCumHaz, LOWERCUMHAZ=LowerCumHaz, and UPPERCUMHAZ=UpperCumHaz. For a Bayesian analysis, CUMHAZ=_ALL_ also includes LOWERHPDCUMHAZ=LowerHPDCumHaz and UpperHPDCUMHAZ=UpperHPDCumHaz.

LOGLOGS=

specifies the log of the negative log of SURVIVAL=.

LOGSURV=
specifies the log of SURVIVAL=.

LOWER=
L=
specifies the lower pointwise confidence limit for the survivor function. For a Bayesian analysis, this is the lower limit of the equal-tail credible interval for the survivor function. The confidence level is determined by the ALPHA= option.

LOWERCIF=
specifies the lower pointwise confidence limit for the cumulative incidence function. The confidence level is determined by the ALPHA= option.

LOWERCMF=
LOWERMCF=
specifies the lower pointwise confidence limit for the cumulative mean function. The confidence level is determined by the ALPHA= option.

LOWERHPD=
specifies the lower limit of the HPD interval for the survivor function. The confidence level is determined by the ALPHA= option.

LOWERHPDCUMHAZ=
specifies the lower limit of the HPD interval for the cumulative hazard function. The confidence level is determined by the ALPHA= option.

LOWERCUMHAZ=
specifies the lower pointwise confidence limit for the cumulative hazard function. For a Bayesian analysis, this is the lower limit of the equal-tail credible interval for the cumulative hazard function. The confidence level is determined by the ALPHA= option.

STDERR=
specifies the standard error of the survivor function estimator. For a Bayesian analysis, this is the standard deviation of the posterior distribution of the survivor function.

STDCIF=
specifies the estimated standard error of the cumulative incidence function estimator.

STDCMF=
STDMCF=
specifies the estimated standard error of the cumulative mean function estimator.

STDCUMHAZ=
specifies the estimated standard error of the cumulative hazard function estimator. For a Bayesian analysis, this is the standard deviation of the posterior distribution of the cumulative hazard function.

STDXBETA=
specifies the estimated standard error of the linear predictor estimator. For a Bayesian analysis, this is the standard deviation of the posterior distribution of the linear predictor.

SURVIVAL=
specifies the survivor function ($S(t) = [S_0(t)]^{\exp(\boldsymbol{\beta}'\mathbf{x})}$) estimate. Specifying SURVIVAL=_ALL_ is equivalent to specifying SURVIVAL=Survival, STDERR=StdErrSurvival, LOWER=LowerSurvival, and UPPER=UpperSurvival; and for a Bayesian analysis, SURVIVAL=_ALL_ also specifies LOWERHPD=LowerHPDSurvival and UPPERHPD=UpperHPDSurvival.

UPPER=
U=
specifies the upper pointwise confidence limit for the survivor function. For a Bayesian analysis, this is the upper limit of the equal-tail credible interval for the survivor function. The confidence level is determined by the ALPHA= option.

UPPERCIF=
specifies the upper pointwise confidence limit for the cumulative incidence function. The confidence level is determined by the ALPHA= option.

UPPERCMF=
UPPERMCF=
specifies the upper pointwise confidence limit for the cumulative mean function. The confidence level is determined by the ALPHA= option.

UPPERCUMHAZ=
specifies the upper pointwise confidence limit for the cumulative hazard function. For a Bayesian analysis, this is the upper limit of the equal-tail credible interval for the cumulative hazard function. The confidence level is determined by the ALPHA= option.

UPPERHPD=
specifies the upper limit of the equal-tail credible interval for the survivor function. The confidence level is determined by the ALPHA= option.

UPPERHPDCUMHAZ=
specifies the upper limit of the equal-tail credible interval for the cumulative hazard function. The confidence level is determined by the ALPHA= option.

XBETA=
specifies the estimate of the linear predictor $\mathbf{x}'\boldsymbol{\beta}$.

The following *options* can appear in the BASELINE statement after a slash (/). The METHOD= and CLTYPE= options apply only to the estimate of the survivor function in the classical analysis. For the Bayesian analysis, the survivor function is estimated by the Breslow (1972) method.

ALPHA=*value*
specifies the significance level of the confidence interval for the survivor function. The value must be between 0 and 1. The default is the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified.

CLTYPE=*method*

specifies the transformation used to compute the confidence limits for $S(t, \mathbf{z})$, the survivor function for a subject with a fixed covariate vector $\mathbf{z}$ at event time t. The CLTYPE= option can take the following values:

LOG

specifies that the confidence limits for $\log(S(t, \mathbf{z}))$ be computed using the normal theory approximation. The confidence limits for $S(t, \mathbf{z})$ are obtained by back-transforming the confidence limits for $\log(S(t, \mathbf{z}))$.

LOGLOG

specifies that the confidence limits for the $\log(-\log(S(t, \mathbf{z})))$ be computed using normal theory approximation. The confidence limits for $S(t, \mathbf{z})$ are obtained by back-transforming the confidence limits for $\log(-\log(S(t, \mathbf{z})))$.

NORMAL

IDENTITY

specifies that the confidence limits for $S(t, \mathbf{z})$ be computed directly using normal theory approximation.

The default is CLTYPE=LOG.

DIRADJ

computes direct adjusted survival curves (Makuch 1982; Gail and Byar 1986; Zhang et al. 2007) by averaging the estimated survival curves for the observations in the COVARIATES= data set. For more information about the adjusted survival curves, see the section "Direct Adjusted Survival Curves" on page 7599 and Example 92.8. If the COVARIATES= data set is not specified, the input data set specified in the DATA= option in the PROC PHREG statement is used instead. If you also specify the GROUP= option, PROC PHREG computes an adjusted survival curve for each value of the GROUP= variable.

GROUP=*variable*

names a variable whose values identify or group the estimated survival curves. The behavior of this option depends on whether you also specify the DIRADJ option:

- If you also specify the DIRADJ option, *variable* must be a CLASS variable in the model. A direct adjusted survival curve is computed for each value of *variable* in the input data. The *variable* does not have to be a variable in the COVARIATES= data set. Each direct adjusted survival curve is the average of the survival curves of all individuals in the COVARIATES= data set with their value of *variable* set to a specific value.
- If you do not specify the DIRADJ option, *variable* is required to be a numeric variable in the COVARIATES= data set. Survival curves for the observations with the same value of the *variable* are overlaid in the same plot.

METHOD=*method*

specifies the method used to compute the survivor function estimates. For more information, see the section "Survivor Function Estimators" on page 7596. You can specify the following *methods*:

BRESLOW

CH

EMP

specifies that the Breslow (1972) estimator be used to compute the survivor function—that is, that the survivor function be estimated by exponentiating the negative empirical cumulative hazard function.

FH

specifies that the Fleming-Harrington (FH) estimates be computed. The FH estimator is a tie-breaking modification of the Breslow estimator. If there are no tied event times, this estimator is the same as the Breslow estimator.

PL

specifies that the product-limit estimates of the survivor function be computed. This estimator is not available if you use the model syntax that allows two time variables for counting process style of input; in such a case the Breslow estimator (METHOD=BRESLOW) is used instead.

The default is METHOD=BRESLOW.

NORMALSAMPLE=*n*

specifies the number of sets of normal random samples to simulate the Gaussian process in the estimation of the confidence limits for the cumulative incidence function. By default, NORMALSAMPLE=100.

ROWID=*variable*

ID=*variable*

ROW=*variable*

names a variable in the COVARIATES= data set for identifying the baseline function curves in the plots. This option has no effect if the PLOTS= option in the PROC PHREG statement is not specified. Values of this variable are used to label the curves for the corresponding rows in the COVARIATES= data set. You can specify ROWID=_OBS_ to use the observation numbers in the COVARIATES= data set for identification.

SEED=*n*

specifies an integer seed, ranging from 1 to $2^{31}-1$, to simulate the distribution of the Gaussian process in the estimation of the confidence limits for the cumulative incidence function. Specifying a seed enables you to reproduce identical confidence limits from the same PROC PHREG specification. If you omit the SEED= option, or if you specify a nonpositive seed, a random seed is derived from the time of day on the computer's clock.

For recurrent events data, both CMF= and CUMHAZ= statistics are the Nelson estimators, but their standard error are not the same. Confidence limits for the cumulative mean function and cumulative hazard function are based on the log transform.

BAYES Statement

BAYES < *options* > **;**

The BAYES statement requests a Bayesian analysis of the regression model by using Gibbs sampling. The Bayesian posterior samples (also known as the chain) for the regression parameters can be output to a SAS data set. Table 92.4 summarizes the *options* available in the BAYES statement.

Table 92.4 BAYES Statement Options

Option	Description
Monte Carlo Options	
INITIAL=	Specifies initial values of the chain
NBI=	Specifies the number of burn-in iterations
NMC=	Specifies the number of iterations after burn-in
SAMPLING=	Specifies the sampling algorithm
SEED=	Specifies the random number generator seed
THINNING=	Controls the thinning of the Markov chain
Model and Prior Options	
COEFFPRIOR=	Specifies the prior of the regression coefficients
DISPERSIONPRIOR=	Specifies the prior of the dispersion parameter for frailties
PIECEWISE=	Specifies details of the piecewise exponential model
SPLINE=	Specifies details of the proportional hazards spline model
Summaries and Diagnostics of the Posterior Samples	
DIAGNOSTICS=	Displays convergence diagnostics
PLOTS=	Displays diagnostic plots
STATISTICS=	Displays summary statistics
Posterior Samples	
OUTPOST=	Names a SAS data set for the posterior samples

The following list describes these *options* and their *suboptions*.

COEFFPRIOR=UNIFORM | NORMAL < **(***normal-option***)** > **| ZELLNER** < **(***zellner-option***)** >

CPRIOR=UNIFORM | NORMAL < **(***normal-option***)** > **| ZELLNER** < **(***zellner-option***)** >

COEFF=UNIFORM | NORMAL < **(***normal-option***)** > **| ZELLNER** < **(***zellner-option***)** >

specifies the prior distribution for the regression coefficients. The default is COEFFPRIOR=UNIFORM. The following prior distributions are available:

UNIFORM

specifies a flat prior—that is, the prior that is proportional to a constant ($p(\beta_1, \ldots, \beta_k) \propto 1$ for all $-\infty < \beta_i < \infty$).

NORMAL< (*normal-option*) >
specifies a normal distribution. The *normal-options* include the following:

INPUT=*SAS-data-set*
specifies a SAS data set that contains the mean and covariance information of the normal prior. The data set must contain the _TYPE_ variable to identify the observation type, and it must contain a variable to represent each regression coefficient. If the data set also contains the _NAME_ variable, values of this variable are used to identify the covariances for the _TYPE_='COV' observations; otherwise, the _TYPE_='COV' observations are assumed to be in the same order as the explanatory variables in the MODEL statement. PROC PHREG reads the mean vector from the observation with _TYPE_='MEAN' and the covariance matrix from observations with _TYPE_='COV'. For an independent normal prior, the variances can be specified with _TYPE_='VAR'; alternatively, the precisions (inverse of the variances) can be specified with _TYPE_='PRECISION'.

RELVAR < =*c* >
specifies a normal prior $N(\mathbf{0}, c\mathbf{J})$, where $\mathbf{J}$ is a diagonal matrix with diagonal elements equal to the variances of the corresponding ML estimator. By default, *c*=1E6.

VAR=*c*
specifies the normal prior $N(\mathbf{0}, c\mathbf{I})$, where $\mathbf{I}$ is the identity matrix.

If you do not specify a *normal-option*, the normal prior $N(\mathbf{0}, 10^6\mathbf{I})$, where $\mathbf{I}$ is the identity matrix, is used. For more information, see the section "Normal Prior" on page 7615.

ZELLNER< (*zellner-option*) >
specifies the Zellner g-prior for the regression coefficients. The g-prior is a normal prior distribution with mean zero and covariance matrix equal to $(g\mathbf{X}'\mathbf{X})^{-1}$, where $\mathbf{X}$ is the design matrix and g can be a constant or a parameter with a gamma prior. The *zellner-options* include the following:

G=*number*
specifies a constant *number* for g.

GAMMA < (SHAPE=*a* ISCALE=*b*) >
specifies that g has a gamma prior distribution $G(a, b)$ with density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$. By default, *a*=*b*=1E–4.

If you do not specify a *zellner-option*, the default is ZELLNER(g=1E–6).

DISPERSIONPRIOR=GAMMA< (*gamma-options*) > | IGAMMA< (*igamma-options*) > | IMPROPER
DPRIOR=GAMMA< (*gamma-options*) > | IGAMMA< (*igamma-options*) > | IMPROPER
specifies the prior distribution of the dispersion parameter. For gamma frailty, the dispersion parameter is the variance of the gamma frailty; for lognormal frailty, the dispersion parameter is the variance of the normal random component. The default is DISPERSIONPRIOR=IMPROPER.

You can specify the following values for this *option*:

GAMMA< (*gamma-options*) >

specifies the gamma prior. A gamma prior $G(a, b)$ with hyperparameters a and b has density $f(\theta) = \frac{b^a \theta^{a-1} \mathrm{e}^{-b\theta}}{\Gamma(a)}$, where a is the shape parameter and b is the inverse-scale parameter. You can use the following *gamma-options* enclosed in parentheses to specify the hyperparameters:

SHAPE=*a*

ISCALE=*b*

results in a $G(a, b)$ prior when both *gamma-options* are specified.

SHAPE=*c*

results in a $G(c, c)$ prior when specified alone.

ISCALE=*c*

results in a $G(c, c)$ prior when specify alone.

The default is SHAPE=1E–4 and ISCALE=1E–4.

IGAMMA< (*igamma-options*) >

specifies the inverse gamma prior. An inverse gamma prior $IG(a, b)$ with hyperparameters a and b has a density $f(\theta) = \frac{b^a \theta^{-(a+1)} \mathrm{e}^{-\frac{b}{\theta}}}{\Gamma(a)}$, where a is the shape parameter and b is the scale parameter. You can use the following *igamma-options* enclosed in parentheses to specify the hyperparameters:

SHAPE=*a*

SCALE=*b*

results in a $IG(a, b)$ prior when both *igamma-options* are specified.

SHAPE=*c*

results in a $IG(c, c)$ prior when specified alone.

SCALE=*c*

results in a $IG(c, c)$ prior when specified alone.

The default is SHAPE=2.001 AND SCALE=0.01.

IMPROPER

specifies the improper prior, which has a density $f(\theta)$ proportional to θ^{-1}.

DIAGNOSTICS=ALL | NONE | *keyword* **|** (*keyword-list*)

DIAG=ALL | NONE | *keyword* **|** (*keyword-list*)

controls the number of diagnostics produced. You can request all the diagnostics in the following list by specifying DIAGNOSTICS=ALL. If you do not want any of these diagnostics, you specify DIAGNOSTICS=NONE. If you want some but not all of the diagnostics, or if you want to change certain settings of these diagnostics, you specify a subset of the following *keywords*. The default is DIAGNOSTICS=(AUTOCORR GEWEKE ESS).

AUTOCORR < (LAGS= *numeric-list*) >

computes the autocorrelations of lags given by LAGS= list for each parameter. Elements in the list are truncated to integers and repeated values are removed. If the LAGS= option is not specified, autocorrelations of lags 1, 5, 10, and 50 are computed for each variable. For more information, see the section "Autocorrelations" on page 174 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

ESS

computes the effective sample size of Kass et al. (1998), the correlation time, and the efficiency of the chain for each parameter. For more information, see the section "Effective Sample Size" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

MCSE

MCERROR

computes the Monte Carlo standard error for each parameter. The Monte Caro standard error, which measures the simulation accuracy, is the standard error of the posterior mean estimate and is calculated as the posterior standard deviation divided by the square root of the effective sample size. For more information. see the section "Standard Error of the Mean Estimate" on page 176 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

HEIDELBERGER < (*heidel-options***) >**

computes the Heidelberger and Welch tests for each parameter. For more information, see the section "Heidelberger and Welch Diagnostics" on page 170 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The tests consist of a stationary test and a halfwidth test. The former tests the null hypothesis that the sample values form a stationary process. If the stationarity test is passed, a halfwidth test is then carried out. Optionally, you can specify one or more of the following *heidel-options*:

SALPHA=*value*

specifies the α level ($0 < \alpha < 1$) for the stationarity test. The default is the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified.

HALPHA=*value*

specifies the α level ($0 < \alpha < 1$) for the halfwidth test. The default is the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified.

EPS=*value*

specifies a small positive number ϵ such that if the halfwidth is less than ϵ times the sample mean of the retaining samples, the halfwidth test is passed.

GELMAN < (*gelman-options***) >**

computes the Gelman and Rubin convergence diagnostics. For more information, see the section "Gelman and Rubin Diagnostics" on page 167 in Chapter 8, "Introduction to Bayesian Analysis Procedures." You can specify one or more of the following *gelman-options*:

NCHAIN=*number*

N=*number*

specifies the number of parallel chains used to compute the diagnostic and has to be 2 or larger. The default is NCHAIN=3. The NCHAIN= option is ignored when the INITIAL= option is specified in the BAYES statement, and in such a case, the number of parallel chains is determined by the number of valid observations in the INITIAL= data set.

ALPHA=*value*

specifies the significance level for the upper bound. The default is the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified (resulting in a 97.5% bound).

GEWEKE < *geweke-options* >

computes the Geweke diagnostics. For more information, see the section "Geweke Diagnostics" on page 169 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The diagnostic is essentially a two-sample *t*-test between the first f_1 portion and the last f_2 portion of the chain. The default is f_1=0.1 and f_2=0.5, but you can choose other fractions by using the following *geweke-options*:

FRAC1=*value*

specifies the early f_1 fraction of the Markov chain.

FRAC2=*value*

specifies the latter f_2 fraction of the Markov chain.

RAFTERY < (*raftery-options*) >

computes the Raftery and Lewis diagnostics. For more information, see the section "Raftery and Lewis Diagnostics" on page 172 in Chapter 8, "Introduction to Bayesian Analysis Procedures." The diagnostic evaluates the accuracy of the estimated quantile ($\hat{\theta}_Q$ for a given $Q \in (0, 1)$) of a chain. $\hat{\theta}_Q$ can achieve any degree of accuracy when the chain is allowed to run for a long time. A stopping criterion is when the estimated probability $\hat{P}_Q = \Pr(\theta \leq \hat{\theta}_Q)$ reaches within $\pm R$ of the value Q with probability S; that is, $\Pr(Q - R \leq \hat{P}_Q \leq Q + R) = S$. The following *raftery-options* enable you to specify Q, R, S, and a precision level ϵ for a stationary test.

QUANTILE=*value*

Q=*value*

specifies the order (a value between 0 and 1) of the quantile of interest. The default is 0.025.

ACCURACY=*value*

R=*value*

specifies a small positive number as the margin of error for measuring the accuracy of estimation of the quantile. The default is 0.005.

PROBABILITY=*value*

S=*value*

specifies the probability of attaining the accuracy of the estimation of the quantile. The default is 0.95.

EPSILON=*value*

EPS=*value*

specifies the tolerance level (a small positive number) for the test. The default is 0.001.

INITIAL=*SAS-data-set*

specifies the SAS data set that contains the initial values of the Markov chains. The INITIAL= data set must contain a variable for each parameter in the model. You can specify multiple rows as the initial values of the parallel chains for the Gelman-Rubin statistics, but posterior summary statistics, diagnostics, and plots are computed only for the first chain.

NBI=*number*

specifies the number of burn-in iterations before the chains are saved. The default is 2000.

NMC=*number*

specifies the number of iterations after the burn-in. The default is 10000.

OUTPOST=*SAS-data-set*

OUT=*SAS-data-set*

names the SAS data set that contains the posterior samples. For more information, see the section "OUTPOST= Output Data Set in the BAYES Statement" on page 7623. Alternatively, you can output the posterior samples into a data set, as shown in the following example in which the data set is named PostSamp.

```
ODS OUTPUT PosteriorSample = PostSamp;
```

PIECEWISE < =*keyword* **< (< NINTERVAL=***number* **> < INTERVALS=(***numeric-list***) > < PRIOR=***option* **>) > >**

specifies that the piecewise constant baseline hazard model be used in the Bayesian analysis. You can specify one of the following two *keywords*:

HAZARD

models the baseline hazard parameters in the original scale. The hazard parameters are named Lambda1, Lambda2, . . ., and so on.

LOGHAZARD

models the baseline hazard parameters in the log scale. The log-hazard parameters are named Alpha1, Alpha2, . . ., and so on.

Specifying PIECEWISE by itself is the same as specifying PIECEWISE=LOGHAZARD.

You can choose one of the following two options to specify the partition of the time axis into intervals of constant baseline hazards:

NINTERVAL=*number*

N=*number*

specifies the number of intervals with constant baseline hazard rates. PROC PHREG partitions the time axis into the given number of intervals with approximately equal number of events in each interval.

INTERVALS=(*numeric-list*)

INTERVAL=(*numeric-list*)

specifies the list of numbers that partition the time axis into disjoint intervals with constant baseline hazard in each interval. For example, INTERVALS=(100, 150, 200, 250, 300) specifies a model with a constant hazard in the intervals [0,100), [100,150), [150,200), [200,250), [250,300), and [300,∞). Each interval must contain at least one event; otherwise, the posterior distribution can be improper, and inferences cannot be derived from an improper posterior distribution.

If neither NINTERVAL= nor INTERVALS= is specified, the default is NINTERVAL=8.

To specify the prior for the baseline hazards $(\lambda_1, \ldots, \lambda_J)$ in the original scale, you specify the following:

PRIOR = IMPROPER | UNIFORM | GAMMA< (*gamma-option*) > | ARGAMMA< (*argamma-option*) >

The default is PRIOR=IMPROPER. The available prior options include the following:

IMPROPER

specifies the noninformative and improper prior $p(\lambda_1, \ldots, \lambda_J) \propto \prod_i \lambda_i{}^{-1}$ for all $\lambda_i > 0$.

UNIFORM

specifies a uniform prior on the real line; that is, $p(\lambda_i) \propto 1$ for all $\lambda_i > 0$.

GAMMA < (*gamma-option*) >

specifies an independent gamma prior $G(a, b)$ with density $f(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}$, which can be followed by one of the following *gamma-options* enclosed in parentheses. The hyperparameters a and b are the shape and inverse-scale parameters of the gamma distribution, respectively. For more information, see the section "Independent Gamma Prior" on page 7614. The default is $G(10^{-4}, 10^{-4})$ for each λ_j, setting the prior mean to 1 with variance 1E4. This prior is proper and reasonably noninformative.

INPUT=*SAS-data-set*

specifies a data set containing the hyperparameters of the independent gamma prior. The data set must contain the _TYPE_ variable to identify the observation type, and it must contain the variables named Lambda1, Lambda2, ..., and so forth, to represent the hazard parameters. The observation with _TYPE_='SHAPE' identifies the shape parameters, and the observation with _TYPE_='ISCALE' identifies the inverse-scale parameters.

RELSHAPE< =*c* >

specifies independent $G(c\hat{\lambda}_j, c)$ distribution, where $\hat{\lambda}_j$'s are the MLEs of the hazard rates. This prior has mean $\hat{\lambda}_j$ and variance $\frac{\hat{\lambda}_j}{c}$. By default, *c*=1E–4.

SHAPE=*a* and ISCALE=*b*

together specify the $G(a, b)$ prior.

SHAPE=*c*

ISCALE=*c*

specifies the $G(c, c)$ prior.

ARGAMMA < (*argamma-option***) >**

specifies an autoregressive gamma prior of order 1, which can be followed by one of the following *argamma-options*. For more information, see the section "AR1 Prior" on page 7614.

INPUT=*SAS-data-set*

specifies a data set containing the hyperparameters of the correlated gamma prior. The data set must contain the _TYPE_ variable to identify the observation type, and it must contain the variables named Lambda1, Lambda2, ..., and so forth, to represent the hazard parameters. The observation with _TYPE_='SHAPE' identifies the shape parameters, and the observation with _TYPE_='ISCALE' identifies the *relative* inverse-scale parameters; that is, if a_j and b_j are, respectively, the SHAPE and ISCALE values for $\lambda_j, 1 \leq j \leq J$, then $\lambda_1 \sim G(a_1, b_1)$, and $\lambda_j \sim G(a_j, b_j/\lambda_{j-1})$ for $2 \leq j \leq J$.

SHAPE=*a* **and SCALE=***b*

together specify that $\lambda_1 \sim G(a, b)$ and $\lambda_j \sim G(a, b/\lambda_{j-1})$ for $2 \leq j \leq J$.

SHAPE=*c*

ISCALE=*c*

specifies that $\lambda_1 \sim G(c, c)$ and $\lambda_j \sim G(c, c/\lambda_{j-1})$ for $2 \leq j \leq J$.

To specify the prior for $\alpha_1, \ldots, \alpha_J$, the hazard parameters in the log scale, you specifying the following:

PRIOR=UNIFORM | NORMAL< (*normal-option***) >**

specifies the prior for the loghazard parameters. The default is PRIOR=UNIFORM. The available PRIOR= options are as follows:

UNIFORM

specifies the uniform prior on the real line; that is, $\alpha_i \propto 1$ for all $-\infty < \alpha_i < \infty$.

NORMAL< (*normal-option***) >**

specifies a normal prior distribution on the log-hazard parameters. The *normal-options* include the following. If you do not specify a *normal-option*, the normal prior $N(\mathbf{0}, 10^6\mathbf{I})$, where $\mathbf{I}$ is the identity matrix, is used.

INPUT=*SAS-data-set*

specifies a SAS data set containing the mean and covariance information of the normal prior. The data set must contain the _TYPE_ variable to identify the observation type, and it must contain variables named Alpha1, Alpha2, ..., and so forth, to represent the log-hazard parameters. If the data set also contains the _NAME_ variable, the value of this variable will be used to identify the covariances for the _TYPE_='COV' observations; otherwise, the _TYPE_='COV' observations are assumed to be in the same order as the explanatory variables in the MODEL statement. PROC PHREG reads the mean vector from the observation with _TYPE_='MEAN' and the covariance matrix from observations with _TYPE_='COV'. For more information, see the section "Normal Prior" on page 7615. For an independent normal prior, the variances can be specified with _TYPE_='VAR'; alternatively, the precisions (inverse of the variances) can be specified with _TYPE_='PRECISION'.

If you have a joint normal prior for the log-hazard parameters and the regression coefficients, specify the same data set containing the mean and covariance information of the multivariate normal distribution in both the COEFFPRIOR=NORMAL(INPUT=) and the PIECEWISE=LOGHAZARD(PRIOR=NORMAL(INPUT=)) options. For more information, see the section "Joint Multivariate Normal Prior for Log-Hazards and Regression Coefficients" on page 7615.

RELVAR <=*c*>
specifies the normal prior $N(\mathbf{0}, c\mathbf{J})$, where $\mathbf{J}$ is a diagonal matrix with diagonal elements equal to the variances of the corresponding ML estimator. By default, *c*=1E6.

VAR=*c*
specifies the normal prior $N(\mathbf{0}, c\mathbf{I})$, where $\mathbf{I}$ is the identity matrix.

PLOTS <(*global-plot-options*)> = *plot-request*

PLOTS <(*global-plot-options*)> = (*plot-requests*)

controls the diagnostic plots produced through ODS Graphics. Three types of plots can be requested: trace plots, autocorrelation function plots, and kernel density plots. By default, the plots are displayed in panels unless the global plot option UNPACK is specified. If you specify more than one type of plots, the plots are displayed by parameters unless the global plot option GROUPBY=TYPE is specified. When you specify only one plot request, you can omit the parentheses around the plot request. For example:

```
plots=none
plots(unpack)=trace
plots=(trace autocorr)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc phreg;
   model y=x;
   bayes plots=trace;
run;
```

If ODS Graphics is enabled but you do not specify the PLOTS= option in the BAYES statement, then PROC PHREG produces, for each parameter, a panel that contains the trace plot, the autocorrelation function plot, and the density plot. This is equivalent to specifying **`plots=(trace autocorr density)`**.

The *global-plot-options* include the following:

FRINGE
creates a fringe plot on the X axis of the density plot.

GROUPBY = PARAMETER | TYPE

specifies how the plots are to be grouped when there is more than one type of plots. The choices are as follows:

TYPE

specifies that the plots be grouped by type.

PARAMETER

specifies that the plots be grouped by parameter.

GROUPBY=PARAMETER is the default.

SMOOTH

displays a fitted penalized B-spline curve each trace plot.

UNPACKPANEL

UNPACK

specifies that all paneled plots be unpacked, meaning that each plot in a panel is displayed separately.

The *plot-requests* include the following:

ALL

specifies all types of plots. PLOTS=ALL is equivalent to specifying PLOTS=(TRACE AUTOCORR DENSITY).

AUTOCORR

displays the autocorrelation function plots for the parameters.

DENSITY

displays the kernel density plots for the parameters.

NONE

suppresses all diagnostic plots.

TRACE

displays the trace plots for the parameters. For more information, see the section "Visual Analysis via Trace Plots" on page 163 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

Consider a model with four parameters, X1–X4. Displays for various specification are depicted as follows.

1. PLOTS=(TRACE AUTOCORR) displays the trace and autocorrelation plots for each parameter side by side with two parameters per panel:

Display 1	Trace(X1)	Autocorr(X1)
	Trace(X2)	Autocorr(X2)
Display 2	Trace(X3)	Autocorr(X3)
	Trace(X4)	Autocorr(X4)

2. PLOTS(GROUPBY=TYPE)=(TRACE AUTOCORR) displays all the paneled trace plots, followed by panels of autocorrelation plots:

Display 1	Trace(X1)	
	Trace(X2)	
Display 2	Trace(X3)	
	Trace(X4)	
Display 3	Autocorr(X1)	Autocorr(X2)
	Autocorr(X3)	Autocorr(X4)

3. PLOTS(UNPACK)=(TRACE AUTOCORR) displays a separate trace plot and a separate correlation plot, parameter by parameter:

Display 1	Trace(X1)
Display 2	Autocorr(X1)
Display 3	Trace(X2)
Display 4	Autocorr(X2)
Display 5	Trace(X3)
Display 6	Autocorr(X3)
Display 7	Trace(X4)
Display 8	Autocorr(X4)

4. PLOTS(UNPACK GROUPBY=TYPE) = (TRACE AUTOCORR) displays all the separate trace plots followed by the separate autocorrelation plots:

Display 1	Trace(X1)
Display 2	Trace(X2)
Display 3	Trace(X3)
Display 4	Trace(X4)
Display 5	Autocorr(X1)
Display 6	Autocorr(X2)
Display 7	Autocorr(X3)
Display 8	Autocorr(X4)

SAMPLING=*keyword*

specifies the sampling algorithm used in the Markov chain Monte Carlo (MCMC) simulations. Two sampling algorithms are available:

ARMS

GIBBS

requests the use of the adaptive rejection Metropolis sampling (ARMS) algorithm to draw the Gibbs samples.

RWM

requests the use of the random walk Metropolis (RWM) algorithm to draw the samples.

Both sampling algorithms can be applied to the Cox model and the piecewise constant hazards model; the default for these models is SAMPLING=ARMS. Only the RWM algorithm can be applied to the proportional hazards spline model and the frailty model. For more information about the MCMC sampling algorithms, see the section "Markov Chain Monte Carlo Method" on page 155 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

SEED=*number*

specifies an integer seed that ranges from 1 to 2^{31}–1 for the random number generator in the simulation. Specifying a seed enables you to reproduce identical Markov chains for the same specification. If you omit the SEED= option, or if you specify a nonpositive seed, a random seed is derived from the time of day on the computer's clock.

SPLINE < (< DF=*number* **> < PRIOR=***option* **>) >**

CUBICSPLINE < (< DF=*number* **> < PRIOR=***option* **>) >**

used the proportional hazards spline model of Royston and Parmar (2002) in the Bayesian analysis. This is a proportional hazards model in which the logarithm of the baseline cumulative hazard function is a natural cubic spline function of the log time,

$$\log[H_0(t)] = s(y, \boldsymbol{\gamma}) \equiv \gamma_0 + \gamma_1 y + \gamma_2 v_1(y) + \cdots + \gamma_{m+1} v_m(y)$$

where $y = \log(t)$ represents the time on a log scale. The v_j are the basis functions, which are computed as

$$v_j(y) = (y - k_j)_+^3 - e_j(y - k_{\min})_+^3 - (1 - e_j)(y - k_{\max})_+^3$$

where

$$\begin{aligned} e_j &= \frac{k_{\max} - k_j}{k_{\max} - k_{\min}} \\ (x - a)_+ &= \max(0, x - a) \end{aligned}$$

Here, $k_{\min}$ and $k_{\max}$ are two terminal knots, and $k_1 < \cdots < k_m$ are m internal knots that are placed between $k_{\min}$ and $k_{\max}$. The degrees of freedom equals $m + 1$. When $m = 0$, the log of the baseline hazard becomes $\gamma_0 + \gamma_1 x$, which corresponds to a common form of the Weibull model.

You can specify the degrees of freedom in the DF=*number* option, where *number* must be an integer. The number of knots equals *number* plus 1. By default, DF=2.

To specify the prior for $\gamma_0, \ldots, \gamma_{m+1}$, the parameters for the cubic splines, you specify the following option:

PRIOR=UNIFORM | NORMAL< (*normal-option*) >
specifies the prior for the cubic splines parameters. You can specify the following values:

UNIFORM
specifies the uniform prior on the real line; that is, $\gamma_i \propto 1$ for all $-\infty < \gamma_i < \infty$.

NORMAL< (*normal-option*) >
specifies a normal prior distribution on the cubic spline parameters. If you do not specify a *normal-option*, the procedure uses the normal prior $N(\mathbf{0}, 10^6\mathbf{I})$, where $\mathbf{I}$ is the identity matrix. You can specify the following *normal-options*:

INPUT=*SAS-data-set*
specifies a SAS data set that contains the mean and covariance information of the normal prior. The data set must contain the _TYPE_ variable to identify the observation type, and it must contain variables named Gamma1, Gamma2, . . . , and so on, to represent the log-hazard parameters. If the data set also contains the _NAME_ variable, the value of this variable is used to identify the covariances for the _TYPE_='COV' observations; otherwise, the _TYPE_='COV' observations are assumed to be in the same order as the explanatory variables in the MODEL statement. PROC PHREG reads the mean vector from the observation for which _TYPE_='MEAN' and reads the covariance matrix from observations for which _TYPE_='COV'. For more information, see the section "Normal Prior" on page 7615. For an independent normal prior, you can specify the variances by using _TYPE_='VAR'; alternatively, you can specify the precisions (inverse of the variances) by using _TYPE_='PRECISION'.

If you have a joint normal prior for the cubic splines parameters and the regression coefficients, specify the same data set containing the mean and covariance information of the multivariate normal distribution in both the COEFFPRIOR=NORMAL(INPUT=) and SPLINE(PRIOR=NORMAL(INPUT=)) options. For more information, see the section "Joint Multivariate Normal Prior for Cubic Spline Parameters and Regression Coefficients" on page 7615.

RELVAR < =*c* >
specifies the normal prior $N(\mathbf{0}, c\mathbf{J})$, where $\mathbf{J}$ is a diagonal matrix whose diagonal elements are equal to the variances of the corresponding ML estimator. By default, *c*=1E6.

VAR=*c*
specifies the normal prior $N(\mathbf{0}, c\mathbf{I})$, where $\mathbf{I}$ is the identity matrix.

By default, PRIOR=UNIFORM.

STATISTICS < (*global-options*) > = ALL | NONE | *keyword* | (*keyword-list*)

STATS < (*global-options*) > = ALL | NONE | *keyword* | (*keyword-list*)
controls the number of posterior statistics produced. Specifying STATISTICS=ALL is equivalent to specifying STATISTICS=(SUMMARY INTERVAL COV CORR). If you do not want any posterior statistics, you specify STATISTICS=NONE. The default is STATISTICS=(SUMMARY INTERVAL). For more information, see the section "Summary Statistics" on page 175 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

You can specify the following *global-options*:

ALPHA=*numeric-list*
controls the probabilities of the credible intervals. The ALPHA= values must be between 0 and 1. Each ALPHA= value produces a pair of 100(1–ALPHA)% equal-tail and HPD intervals for each parameters. The default is the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified (yielding the 95% credible intervals for each parameter).

PERCENT=*numeric-list*
requests the percentile points of the posterior samples. The PERCENT= values must be between 0 and 100. The default is PERCENT= 25, 50, 75, which yield the 25th, 50th, and 75th percentile points for each parameter.

You can specify the following values for a *keyword* or as part of a *keyword-list*. To specify a list, place parentheses around multiple *keywords* that are separated by spaces.

CORR
produces the posterior correlation matrix.

COV
produces the posterior covariance matrix.

SUMMARY
produces the means, standard deviations, and percentile points for the posterior samples. The default is to produce the 25th, 50th, and 75th percentile points, but you can use the global PERCENT= option to request specific percentile points.

INTERVAL
produces equal-tail credible intervals and HPD intervals. The default is to produce the 95% equal-tail credible intervals and 95% HPD intervals, but you can use the global ALPHA= option to request intervals of any probabilities.

THINNING=*number*
THIN=*number*
controls the thinning of the Markov chain. Only one in every k samples is used when THINNING=k, and if NBI=n_0 and NMC=n, the number of samples kept is

$$\left[\frac{n_0 + n}{k}\right] - \left[\frac{n_0}{k}\right]$$

where $[a]$ represents the integer part of the number a. The default is THINNING=1.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC PHREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the PHREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. Response variables do not need to be specified in the CLASS statement.

The CLASS statement must precede the MODEL statement. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*. You can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING
DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC PHREG orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (., ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values of the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so this option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC PHREG interprets values of the ORDER= option:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. You can specify any of the *keywords* shown in the following table. The default is PARAM=REF. Design matrix columns are created from CLASS variables according to the corresponding coding schemes.

Value of PARAM=	Coding
EFFECT	Effect coding
GLM	Less-than-full-rank reference cell coding (this *keyword* can be used only in a global option)
ORDINAL **THERMOMETER**	Cumulative parameterization for an ordinal CLASS variable
POLYNOMIAL **POLY**	Polynomial coding
REFERENCE **REF**	Reference cell coding
ORTHEFFECT	Orthogonalizes PARAM=EFFECT coding
ORTHORDINAL **ORTHOTHERM**	Orthogonalizes PARAM=ORDINAL coding
ORTHPOLY	Orthogonalizes PARAM=POLYNOMIAL coding
ORTHREF	Orthogonalizes PARAM=REFERENCE coding

All parameterizations are full rank, except for the GLM parameterization. The REF= option in the CLASS statement determines the reference level for EFFECT and REFERENCE coding and for their orthogonal parameterizations. It also indirectly determines the reference level for a singular GLM parameterization through the order of levels.

If a PARAM= option is specified as a variable option for some variables, then any variables for which PARAM= is not specified use either the EFFECT parameterization if the global PARAM= option is not specified, or the full-rank parameterization indicated in the global PARAM= option if specified. If the global PARAM=GLM option is specified and PARAM= is also specified for some variables, GLM parameterization is used for all variables.

If PARAM=ORTHPOLY or PARAM=POLY and the classification variable is numeric, then the ORDER= option in the CLASS statement is ignored, and the internal unformatted values are used. For more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

REF=*'level'* | *keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*:

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n*>

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

Class Variable Naming Convention

Parameter names for a CLASS predictor variable are constructed by concatenating the CLASS variable name with the CLASS levels. However, for the POLYNOMIAL and orthogonal parameterizations, parameter names are formed by concatenating the CLASS variable name and keywords that reflect the parameterization. For examples and more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

Class Variable Parameterization with Unbalanced Designs

PROC PHREG initially parameterizes the CLASS variables by looking at the levels of the variables across the complete data set. If you have an *unbalanced* replication of levels across variables or BY groups, then the design matrix and the parameter interpretation might be different from what you expect. For example, suppose you have a model that has one CLASS variable A with three levels (1, 2, and 3) and another CLASS variable B with two levels (1 and 2). If the third level of A occurs only with the first level of B, if you use the EFFECT parameterization, and if your model contains the effect A(B) and an intercept, then the design for A within the second level of B is not a differential effect. In particular, the design looks like the following:

		Design Matrix			
		A(B=1)		A(B=2)	
B	**A**	**A1**	**A2**	**A1**	**A2**
1	1	1	0	0	0
1	2	0	1	0	0
1	3	–1	–1	0	0
2	1	0	0	1	0
2	2	0	0	0	1

PROC PHREG detects linear dependency among the last two design variables and sets the parameter for A2(B=2) to zero, resulting in an interpretation of these parameters as if they were reference- or dummy-coded. The REFERENCE or GLM parameterization might be more appropriate for such problems.

CONTRAST Statement

CONTRAST '*label*' *row-description* <, . . . *row-description*> </ *options*> ;

The CONTRAST statement provides a mechanism for obtaining customized hypothesis tests. It is similar to the CONTRAST statement in PROC GLM and PROC CATMOD, depending on the coding schemes used with any categorical variables involved.

The CONTRAST statement enables you to specify a matrix, $\mathbf{L}$, for testing the hypothesis $\mathbf{L}\boldsymbol{\beta} = \mathbf{0}$. You must be familiar with the details of the model parameterization that PROC PHREG uses (for more information, see the PARAM= option in the section "CLASS Statement" on page 7508). Optionally, the CONTRAST statement enables you to estimate each row, $\boldsymbol{l}_i'\boldsymbol{\beta}$, of $\mathbf{L}\boldsymbol{\beta}$ and test the hypothesis $\boldsymbol{l}_i'\boldsymbol{\beta} = 0$. Computed statistics are based on the asymptotic chi-square distribution of the Wald statistic.

There is no limit to the number of CONTRAST statements that you can specify, but they must appear after the MODEL statement.

The syntax of a *row-description* is:

effect values <,. . . ,*effect values*>

The following parameters are specified in the CONTRAST statement:

label identifies the contrast on the output. A label is required for every contrast specified, and it must be enclosed in quotes.

effect identifies an effect that appears in the MODEL statement. You do not need to include all effects that are included in the MODEL statement.

values are constants that are elements of the $\mathbf{L}$ matrix associated with the effect. To correctly specify your contrast, it is crucial to know the ordering of parameters within each effect and the variable levels associated with any parameter. The "Class Level Information" table shows the ordering of levels within variables. The E option, described later in this section, enables you to verify the proper correspondence of *values* to parameters.

The rows of $\mathbf{L}$ are specified in order and are separated by commas. Multiple degree-of-freedom hypotheses can be tested by specifying multiple *row-descriptions*. For any of the full-rank parameterizations, if an effect is not specified in the CONTRAST statement, all of its coefficients in the $\mathbf{L}$ matrix are set to 0. If too many values are specified for an effect, the extra ones are ignored. If too few values are specified, the remaining ones are set to 0.

When you use effect coding (by specifying PARAM=EFFECT in the CLASS statement), all parameters are directly estimable (involve no other parameters). For example, suppose an effect coded CLASS variable A has four levels. Then there are three parameters $(\alpha_1, \alpha_2, \alpha_3)$ representing the first three levels, and the fourth parameter is represented by

$$-\alpha_1 - \alpha_2 - \alpha_3$$

To test the first versus the fourth level of A, you would test

$$\alpha_1 = -\alpha_1 - \alpha_2 - \alpha_3$$

or, equivalently,

$$2\alpha_1 + \alpha_2 + \alpha_3 = 0$$

which, in the form $\mathbf{L}\boldsymbol{\beta} = 0$, is

$$\begin{bmatrix} 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = 0$$

Therefore, you would use the following CONTRAST statement:

```
contrast '1 vs. 4' A 2 1 1;
```

To contrast the third level with the average of the first two levels, you would test

$$\frac{\alpha_1 + \alpha_2}{2} = \alpha_3$$

or, equivalently,

$$\alpha_1 + \alpha_2 - 2\alpha_3 = 0$$

Therefore, you would use the following CONTRAST statement:

```
contrast '1&2 vs. 3' A 1 1 -2;
```

Other CONTRAST statements involving classification variables with PARAM=EFFECT are constructed similarly. For example:

```
contrast '1 vs. 2    '  A  1 -1  0;
contrast '1&2 vs. 4  '  A  3  3  2;
contrast '1&2 vs. 3&4'  A  2  2  0;
contrast 'Main Effect'  A  1  0  0,
                        A  0  1  0,
                        A  0  0  1;
```

When you use the less-than-full-rank parameterization (by specifying PARAM=GLM in the CLASS statement), each row is checked for estimability. If PROC PHREG finds a contrast to be nonestimable, it displays missing values in corresponding rows in the results. PROC PHREG handles missing level combinations of categorical variables in the same manner as PROC GLM. Parameters corresponding to missing level combinations are not included in the model. This convention can affect the way in which you specify the **L** matrix in your CONTRAST statement. If the elements of **L** are not specified for an effect that contains a specified effect, then the elements of the specified effect are distributed over the levels of the higher-order effect just as the GLM procedure does for its CONTRAST and ESTIMATE statements. For example, suppose that the model contains effects A and B and their interaction A*B. If you specify a CONTRAST statement involving A alone, the **L** matrix contains nonzero terms for both A and A*B, since A*B contains A.

The Cox model contains no explicit intercept parameter, so it is not valid to specify one in the CONTRAST statement. As a consequence, you can test or estimate only homogeneous linear combinations (those with zero-intercept coefficients, such as contrasts that represent group differences) for the GLM parameterization.

The degrees of freedom are the number of linearly independent constraints implied by the CONTRAST statement—that is, the rank of **L**.

You can specify the following *options* after a slash (/).

ALPHA= *p*

specifies the level of significance *p* for the $100(1-p)\%$ confidence interval for each contrast when the ESTIMATE option is specified. The value *p* must be between 0 and 1. By default, *p* is equal to the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified.

E

requests that the **L** matrix be displayed.

ESTIMATE=*keyword*

requests that each individual contrast (that is, each row, $l_i'\boldsymbol{\beta}$, of $\mathbf{L}\boldsymbol{\beta}$) or exponentiated contrast ($\mathrm{e}^{l_i'\boldsymbol{\beta}}$) be estimated and tested. PROC PHREG displays the point estimate, its standard error, a Wald confidence interval, and a Wald chi-square test for each contrast. The significance level of the confidence interval is controlled by the ALPHA= option. You can estimate the contrast or the exponentiated contrast ($\mathrm{e}^{l_i'\boldsymbol{\beta}}$), or both, by specifying one of the following *keywords*:

PARM specifies that the contrast itself be estimated.

EXP specifies that the exponentiated contrast be estimated.

BOTH specifies that both the contrast and the exponentiated contrast be estimated.

SINGULAR=*number*

tunes the estimability check. This option is ignored when the full-rank parameterization is used. If $\mathbf{v}$ is a vector, define $\mathrm{ABS}(\mathbf{v})$ to be the largest absolute value of the elements of $\mathbf{v}$. For a row vector $\boldsymbol{l}'$ of the contrast matrix $\mathbf{L}$, define c to be equal to $\mathrm{ABS}(\boldsymbol{l})$ if $\mathrm{ABS}(\boldsymbol{l})$ is greater than 0; otherwise, c equals 1. If $\mathrm{ABS}(\boldsymbol{l}'-\boldsymbol{l}'\mathbf{T})$ is greater than $c * \mathit{number}$, then $\boldsymbol{l}$ is declared nonestimable. The $\mathbf{T}$ matrix is the Hermite form matrix $\mathbf{I}_0^{-}\mathbf{I}_0$, where $\mathbf{I}_0^{-}$ represents a generalized inverse of the information matrix $\mathbf{I}_0$ of the null model. The value for *number* must be between 0 and 1; the default value is 1E–4.

TEST< (*keywords*) **>**

requests a Type 3 test for each contrast. The default is to use the Wald statistic, but you can requests other statistics by specifying one or more of the following *keywords*:

ALL

requests the likelihood ratio tests, the score tests, and the Wald tests. Specifying TEST(ALL) is equivalent to specifying TEST=(LR SCORE WALD).

NONE

suppresses the Type 3 analysis. Even if the TEST option is not specified, PROC PHREG displays the Wald test results for each model effect if a CLASS variable is involved in a MODEL effect. The NONE option can be used to suppress such display.

LR

requests the likelihood ratio tests. This request is not honored if the COVS option is also specified.

SCORE

requests the score tests. This request is not honored if the COVS option is also specified.

WALD
requests the Wald tests.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 92.5 summarizes the *options* available in the EFFECT statement.

Table 92.5 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.

Table 92.5 *continued*

Option	Description
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, <'*label*'> *estimate-specification* <(*divisor*=*n*)>> <, . . . >
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations. Because the Cox model contains no explicit intercept parameter, you can estimate only homogeneous linear combinations (such as group differences) for the GLM parameterization. For more information, see the section "CONTRAST Statement" on page 7512.

Table 92.6 summarizes *options* available in the ESTIMATE statement. If you specify the BAYES statement, the ADJUST=, STEPDOWN, TESTVALUE, LOWER, UPPER, and JOINT options are ignored. The PLOTS= option is not available for the maximum likelihood analysis. It is available only for the Bayesian analysis.

Table 92.6 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the **L** matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For details about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement" on page 475 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* < / *option* > **;**

The FREQ statement identifies the *variable* (in the input data set) that contains the frequency of occurrence of each observation. PROC PHREG treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If not an integer, the frequency value is truncated to an integer. If the frequency value is missing, the observation is not used in the estimation of the regression parameters.

The following *option* can be specified in the FREQ statement after a slash (/):

NOTRUNCATE

NOTRUNC

specifies that frequency values are not truncated to integers.

HAZARDRATIO Statement

HAZARDRATIO < '*label*' > *variable* < / *options* > **;**

The HAZARDRATIO statement enables you to request hazard ratios for any variable in the model at customized settings. For example, if the model contains the interaction of a CLASS variable A and a continuous variable X, the following specification displays a table of hazard ratios comparing the hazards of each pair of levels of A at X=3:

```
hazardratio A / at (X=3);
```

The HAZARDRATIO statement identifies the variable whose hazard ratios are to be evaluated. If the variable is a continuous variable, the hazard ratio compares the hazards for a given change (by default, a increase of 1 unit) in the variable. For a CLASS variable, a hazard ratio compares the hazards of two levels of the variable. More than one HAZARDRATIO statement can be specified, and an optional label (specified as a quoted string) helps identify the output.

Table 92.7 summarizes the *options* available in the HAZARDRATIO statement.

Table 92.7 HAZARDRATIO Statement Options

Option	Description
ALPHA=	Specifies the alpha level
AT	Specifies the variables that interact with the variable of interest
CL=	Specifies confidence limits
DIFF=	Specifies which differences to consider
E	Displays the log-hazard ratio
PLCONV=	Controls the convergence criterion
PLMAXIT=	Specifies the maximum number of iterations to achieve the convergence
PLSINGULAR=	Specifies the tolerance for testing the singularity
UNITS=	Specifies the units of change

Options for the HAZARDRATIO statement are as follows.

ALPHA=*number*
specifies the alpha level of the interval estimates for the hazard ratios. The value must be between 0 and 1. The default is the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified.

AT (*variable***=ALL | REF |** *list* **< . . .** *variable***=ALL | REF |** *list* **>)**
specifies the variables that interact with the variable of interest and the corresponding values of the interacting variables. If the interacting variable is continuous and a numeric list is specified after the equal sign, hazard ratios are computed for each value in the list. If the interacting variable is a CLASS variable, you can specify, after the equal sign, a list of quoted strings corresponding to various levels of the CLASS variable, or you can specify the keyword ALL or REF. Hazard ratios are computed at each value of the list if the list is specified, or at each level of the interacting variable if ALL is specified, or at the reference level of the interacting variable if REF is specified.

If this option is not specified, PROC PHREG finds all the variables that interact with the variable of interest. If an interacting variable is a CLASS variable, *variable*= ALL is the default; if the interacting variable is continuous, *variable*=*m* is the default, where *m* is the average of all the sampled values of the continuous variable.

Suppose the model contains two interactions: an interaction A*B of CLASS variables A and B, and another interaction A*X of A with a continuous variable X. If 3.5 is the average of the sampled values of X, the following two HAZARDRATIO statements are equivalent:

```
hazardratio A;
hazardratio A / at (B=ALL X=3.5);
```

CL=WALD | PL | BOTH
specifies whether to create the Wald or profile-likelihood confidence limits, or both for the classical analysis. By default, Wald confidence limits are produced. This option is not applicable to a Bayesian analysis.

DIFF=*diff-request*
specifies which differences to consider for the level comparisons of a CLASS variable. This option is ignored in the estimation of hazard ratios for a continuous variable. The *diff-requests* include the following:

DISTINCT

DISTINCTPAIRS

ALL
requests all comparisons of only the distinct combinations of pairs

PAIRWISE
requests all possible pairwise comparisons of levels

REF

requests comparisons between the reference level and all other levels of the CLASS variable.

For example, let A be a CLASS variable with 3 levels (A1, A2, and A3), and A3 is specified as the reference level. The following table depicts the hazard ratios displayed for the three alternatives of the DIFF= option:

	Hazard Ratios Displayed					
DIFF=*option*	**A1 vs A2**	**A2 vs A1**	**A1 vs A3**	**A3 vs A1**	**A2 vs A3**	**A3 vs A2**
DISTINCT	√		√		√	
PAIRWISE	√	√	√	√	√	√
REF			√		√	

The default is DIFF=DISTINCT.

E

displays the vector $\mathbf{h}$ of linear coefficients such that $\mathbf{h}'\boldsymbol{\beta}$ is the log-hazard ratio, with $\boldsymbol{\beta}$ being the vector of regression coefficients.

PLCONV=*value*

controls the convergence criterion for the profile-likelihood confidence limits. The quantity *value* must be a positive number, with a default value of 1E–4. The PLCONV= option has no effect if profile-likelihood confidence intervals (CL=PL) are not requested.

PLMAXIT=*n*

specifies the maximum number of iterations to achieve the convergence of the profile-likelihood confidence limits. By default, PLMAXITER=25. If convergence is not attained in *n* iterations, the corresponding profile-likelihood confidence limit for the hazard ratio is set to missing. The PLMAXITER= option has no effect if profile-likelihood confidence intervals (CL=PL) are not requested.

PLSINGULAR=*value*

specifies the tolerance for testing the singularity of the Hessian matrix in the computation of the profile-likelihood confidence limits. The test requires that a pivot for sweeping this matrix be at least this number times a norm of the matrix. Values of the PLSINGULAR= option must be numeric. By default, *value* is the machine epsilon times 1E7, which is approximately 1E–9. The PLSINGULAR= option has no effect if profile-likelihood confidence intervals (CL=PL) are not requested.

UNITS=*numeric-list*

specifies the units of change in the continuous explanatory variable for which the customized hazard ratio is estimated. The default is UNITS=1. This option is ignored in the computation of the hazard ratios for a CLASS variable.

ID Statement

ID *variables* **;**

The ID statement specifies additional variables for identifying observations in the input data. These variables are placed in the OUT= data set created by the OUTPUT statement. In the computation of the COVSANDWICH estimate, you can aggregate over distinct values of these ID variables.

Only variables in the input data set can be included in the ID statement.

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs. Because the Cox model contains no explicit intercept parameter, you can estimate only the group differences of LS-means. Therefore, in contrast with many other procedures, the DIFF and NOMEANS options in the LSMEANS statement are the default options that display the differences of the LS-means but not the LS-means themselves.

Table 92.8 summarizes the *options* available in the LSMEANS statement. If you specify the BAYES statement, the ADJUST=, STEPDOWN, and LINES options are ignored. The PLOTS= option is not available for the maximum likelihood analysis. It is available only for the Bayesian analysis.

Table 92.8 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion

Table 92.8 *continued*

Option	Description
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For details about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , < '*label*' > *values* < *divisor*=*n* > > < , . . . >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 92.9 summarizes the *options* available in the LSMESTIMATE statement. If the BAYES statement is specified, the ADJUST=, STEPDOWN, TESTVALUE, LOWER, UPPER, and JOINT options are ignored. The PLOTS= option is not available for the maximum likelihood analysis. It is available only for the Bayesian analysis.

Table 92.9 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the $\mathbf{L}$ matrix
ELSM	Prints the $\mathbf{K}$ matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For details about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

MODEL *response* < ∗ *censor* (*list*) > **=** *effects* < / *options* > **;**

MODEL (*t1*, *t2*)< ∗ *censor*(*list*) > **=** *effects* < / *options* > **;**

The MODEL statement identifies the variables to be used as the failure time variables, the optional censoring variable, and the explanatory effects, including covariates, main effects, interactions, nested effects; for more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure." A note of caution: specifying the effect T*A in the MODEL statement, where T is the time variable and A is a CLASS variable, does not make the effect time-dependent. For more information, see the section "Time and CLASS Variables Usage" on page 7546.

Two forms of MODEL syntax can be specified; the first form allows one time variable, and the second form allows two time variables for the counting process style of input (for more information, see the section "Counting Process Style of Input" on page 7551).

In the first MODEL statement, the name of the failure time variable precedes the equal sign. This name can optionally be followed by an asterisk, the name of the censoring variable, and a list of censoring values (separated by blanks or commas if there is more than one) enclosed in parentheses. If the censoring variable takes on one of these values, the corresponding failure time is considered to be censored. Following the equal sign are the explanatory effects (sometimes called independent variables or covariates) for the model.

Instead of a single failure-time variable, the second MODEL statement identifies a pair of failure-time variables. Their names are enclosed in parentheses, and they signify the endpoints of a semiclosed interval $(t1, t2]$ during which the subject is at risk. If the censoring variable takes on one of the censoring values, the time $t2$ is considered to be censored.

The censoring variable must be numeric and the failure-time variables must contain nonnegative values. Any observation with a negative failure time is excluded from the analysis, as is any observation with a missing value for any of the variables listed in the MODEL statement. Failure-time variables with a SAS date format are not recommended because the dates might be translated into negative numbers and consequently the corresponding observation would be discarded.

Table 92.10 summarizes the *options* available in the MODEL statement. These *options* can be specified after a slash (/). Four convergence criteria are allowed for the maximum likelihood optimization: ABSFCONV=, FCONV=, GCONV=, and XCONV=. If you specify more than one convergence criterion, the optimization is terminated as soon as one of the criteria is satisfied. If none of the criteria is specified, the default is GCONV=1E–8.

Table 92.10 MODEL Statement Options

Option	Description
Model Specification Options	
EVENTCODE=	Requests an analysis of competing-risks data
NOFIT	Suppresses model fitting
OFFSET=	Specifies offset variable
SELECTION=	Specifies effect selection method
Effect Selection Options	
BEST=	Controls the number of models displayed for best subset selection

Table 92.10 *continued*

Option	Description
DETAILS	Requests detailed results at each step
HIERARCHY=	Specifies whether and how hierarchy is maintained and whether a single effect or multiple effects are allowed to enter or leave the model per step
INCLUDE=	Specifies number of effects included in every model
MAXSTEP=	Specifies maximum number of steps for stepwise selection
SEQUENTIAL	Adds or deletes effects in sequential order
SLENTRY=	Specifies significance level for entering effects
SLSTAY=	Specifies significance level for removing effects
START=	Specifies number of variables in first model
STOP=	Specifies number of variables in final model
STOPRES	Adds or deletes variables by residual chi-square criterion
Maximum Likelihood Optimization Options	
ABSFCONV=	Specifies absolute function convergence criterion
FCONV=	Specifies relative function convergence criterion
FIRTH	Specifies Firth's penalized likelihood method
GCONV=	Specifies relative gradient convergence criterion
XCONV=	Specifies relative parameter convergence criterion
MAXITER=	Specifies maximum number of iterations
RIDGEINIT=	Specifies the initial ridging value
RIDGING=	Specifies the technique to improve the log likelihood function when its value is worse than that of the previous step
SINGULAR=	Specifies tolerance for testing singularity
Confidence Interval Options	
ALPHA=	Specifies α for the $100(1-\alpha)\%$ confidence intervals
PLCONV=	Specifies profile-likelihood convergence criterion
RISKLIMITS=	Computes confidence intervals for hazard ratios
Display Options	
CORRB	Displays correlation matrix
COVB	Displays covariance matrix
ITPRINT	Displays iteration history
NODUMMYPRINT	Suppresses "Class Level Information" table
TYPE1	Displays Type 1 analysis
TYPE3	Displays Type 3 tests or joint tests of effects
Miscellaneous Options	
ENTRYTIME=	Specifies the delayed entry time variable
ROCLABEL=	Specifies a label to identify the model that is specified in the MODEL statement model for an ROC analysis
TIES=	Specifies the method of handling ties in failure times

ALPHA=*value*

sets the significance level used for the confidence limits for the hazard ratios. The quantity *value* must be between 0 and 1. The default is the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified. This option has no effect unless the RISKLIMITS option is specified.

ABSFCONV=*value*

CONVERGELIKE=*value*

specifies the absolute function convergence criterion. Termination requires a small change in the objective function (log partial likelihood function) in subsequent iterations,

$$|l_k - l_{k-1}| < \textit{value}$$

where l_k is the value of the objective function at iteration k.

BEST=*n*

requests that *n* models with the highest score chi-square statistics be displayed for each model size. This option is used exclusively with the best subset selection (SELECTION=SCORE). If the option is omitted and there are no more than 10 explanatory variables, then all possible models are listed for each model size. If the option is omitted and there are more than 10 explanatory variables, then the number of models selected for each model size is at most equal to the number of explanatory variables listed in the MODEL statement.

See Example 92.2 for an illustration of the best subset selection method and the BEST= option.

CORRB

displays the estimated correlation matrix of the parameter estimates.

COVB

displays the estimated covariance matrix of the parameter estimates.

DETAILS

produces a detailed display at each step of the model-building process. It produces an "Analysis of Variables Not in the Model" table before displaying the variable selected for entry for forward or stepwise selection. For each model fitted, it produces the "Analysis of Maximum Likelihood Estimates" table.

See Example 92.1 for a discussion of these tables.

ENTRYTIME=*variable*

ENTRY=*variable*

specifies the name of the variable that represents the left-truncation time. This option has no effect when the counting process style of input is specified. For more information, see the section "Left-Truncation of Failure Times" on page 7553.

EVENTCODE< (*compete-risk-option***) >=***number*

FAILCODE< (*compete-risk-option***) >=***number*

requests an analysis of competing-risks data. You can specify one of the following *compete-risk-options* to choose between two commonly used regression approaches to analyze the competing-risks data:

COX

models the cause-specific hazard function for each event type separately, by applying the Cox regression that targets each event type and censors all other observations. Based on all the models for the cause-specific hazards, the cumulative incidence function for an event of interest is estimated according to Andersen et al. (1992, chap. VII). For more information, see the section "Modeling the Cause-Specific Hazard" on page 7557.

FG< (BRESLOW) >

requests the Fine and Gray (1999) analysis that models the hazard function of the cumulative incidence function. The model is often referred to as the proportional subdistribution hazards model. For more information, see the section "Modeling the Cumulative Incidence" on page 7559. By default, the Kaplan-Meier estimate of the survival function of the censoring variable is used in the inversely probability censoring weighting (IPCW) technique to construct the pseudo-likelihood. You can replace the Kaplan-Meier estimate by the Breslow estimate in the IPCW technique by specifying the keyword BRESLOW as a suboption; for example,

```
EVENTCODE(FG(BRESLOW)) = 1
```

If you do not specify a *compete-risk-option*, PROC PHREG fits the proportional subdistribution hazard model of Fine and Gray (1999).

The *number* after the equal sign specifies the value of the censoring variable that represents the event of interest for the competing-risks analysis. The following specification requests a cause-specific analysis of which cumulative incidence curves of Status=2 can be requested for specific covariate values:

```
model T*Status(0)= X1-X5 / eventcode(COX)=2;
```

The following specification fits a proportional subdistribution hazard model for Status=2:

```
model T*Status(0)= X1-X5 / eventcode=2;
```

FCONV=*value*

specifies the relative function convergence criterion. Termination requires a small relative change in the objective function (log partial likelihood function) in subsequent iterations,

$$\frac{|l_k - l_{k-1}|}{|l_{k-1}| + 1\mathrm{E} - 6} < \mathit{value}$$

where l_k is the value of the objective function at iteration k.

FIRTH

performs Firth's penalized maximum likelihood estimation to reduce bias in the parameter estimates (Heinze and Schemper 2001; Firth 1993). This method is useful when the likelihood is monotone—that is, the likelihood converges to finite value while at least one estimate diverges to infinity.

GCONV=*value*

specifies the relative gradient convergence criterion. Termination requires that the normalized prediction function reduction is small,

$$\frac{\mathbf{g}_k \mathbf{H}_k^{-1} \mathbf{g}_k}{|l_k| + 1\mathrm{E} - 6} < \textit{value}$$

where l_k is the log partial likelihood, $\mathbf{g}_k$ is the gradient vector (first partial derivatives of the log partial likelihood), and $\mathbf{H}_k$ is the negative Hessian matrix (second partial derivatives of the log partial likelihood), all at iteration k.

HIERARCHY=*keyword*

HIER=*keyword*

specifies whether and how the model hierarchy requirement is applied and whether a single effect or multiple effects are allowed to enter or leave the model in one step. You can specify that only CLASS variable effects, or both CLASS and continuous variable effects, be subject to the hierarchy requirement. The HIERARCHY= option is ignored unless you also specify the forward, backward, or stepwise selection method.

Model hierarchy refers to the requirement that, for any term to be in the model, all effects contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must be in the model. Likewise, neither effect A nor B can leave the model while the interaction A*B is in the model.

You can specify any of the following *keywords* in the HIERARCHY= option:

NONE

indicates that the model hierarchy is not maintained. Any single effect can enter or leave the model at any given step of the selection process.

SINGLE

indicates that only one effect can enter or leave the model at one time, subject to the model hierarchy requirement. For example, suppose that you specify the main effects A and B and the interaction of A*B in the model. In the first step of the selection process, either A or B can enter the model. In the second step, the other main effect can enter the model. The interaction effect can enter the model only when both main effects have already been entered. Also, before A or B can be removed from the model, the A*B interaction must first be removed. All effects (CLASS and continuous variables) are subject to the hierarchy requirement.

SINGLECLASS

is the same as HIERARCHY=SINGLE except that only CLASS effects are subject to the hierarchy requirement.

MULTIPLE

indicates that more than one effect can enter or leave the model at one time, subject to the model hierarchy requirement. In a forward selection step, a single main effect can enter the model, or an interaction can enter the model together with all the effects that are contained in the interaction. In a backward elimination step, an interaction itself, or the interaction together with all the effects that the interaction contains, can be removed. All effects (CLASS and continuous variable) are subject to the hierarchy requirement.

MULTIPLECLASS
is the same as HIERARCHY=MULTIPLE except that only CLASS effects are subject to the hierarchy requirement.

The default value is HIERARCHY=SINGLE, which means that model hierarchy is to be maintained for all effects (that is, both CLASS and continuous variable effects) and that only a single effect can enter or leave the model at each step.

INCLUDE=*n*
includes the first *n* effects in the MODEL statement in every model. By default, INCLUDE=0. The INCLUDE= option has no effect when SELECTION=NONE.

ITPRINT
displays the iteration history, including the last evaluation of the gradient vector.

MAXITER=*n*
specifies the maximum number of iterations allowed. The default value for *n* is 25. If convergence is not attained in *n* iterations, the displayed output and all data sets created by PROC PHREG contain results that are based on the last maximum likelihood iteration.

MAXSTEP=*n*
specifies the maximum number of times the explanatory variables can move in and out of the model before the stepwise model-building process ends. The default value for *n* is twice the number of explanatory variables in the MODEL statement. The option has no effect for other model selection methods.

NODUMMYPRINT

NODESIGNPRINT

NODP
suppresses the "Class Level Information" table, which shows how the design matrix columns for the CLASS variables are coded.

NOFIT
performs the global score test, which tests the joint significance of all the explanatory variables in the MODEL statement. No parameters are estimated. If the NOFIT option is specified along with other MODEL statement options, NOFIT takes precedence, and all other options are ignored except the TIES= option.

OFFSET=*name*
specifies the name of an offset variable, which is an explanatory variable with a regression coefficient fixed as one. You can also use the OFFSET= option to restrict parameters to a fixed value. For example, if you want to restrict the parameter for variable X1 to 1 and the parameter for X2 to 2, compute Restrict = X1 + 2 * X2 in a DATA step, specify the option offset=Restrict, and leave X1 and X2 out of the model. This option can be used to incorporate risk weights for the likelihood function.

PLCONV=*value*

controls the convergence criterion for confidence intervals based on the profile-likelihood function. The quantity *value* must be a positive number, with a default value of 1E–4. The PLCONV= option has no effect if profile-likelihood based confidence intervals are not requested.

RIDGING=*keyword*

specifies the technique to improve the log likelihood when its value is worse than that of the previous step. The available *keywords* are as follows:

ABSOLUTE

specifies that the diagonal elements of the negative (expected) Hessian be inflated by adding the ridge value.

RELATIVE

specifies that the diagonal elements be inflated by the factor equal to 1 plus the ridge value.

NONE

specifies the crude line-search method of taking half a step be used instead of ridging.

The default is RIDGING=RELATIVE.

RIDGEINIT=*value*

specifies the initial ridge value. The maximum ridge value is 2000 times the maximum of 1 and the initial ridge value. The initial ridge value is raised to 1E–4 if it is less than 1E–4. By default, RIDGEINIT=1E–4. This option has no effect for RIDGING=ABSOLUTE.

RISKLIMITS< =*keyword* **>**

RL< =*keyword* **>**

produces confidence intervals for hazard ratios of main effects not involved in interactions or nestings. Computation of these confidence intervals is based on the profile likelihood or based on individual Wald tests. The confidence coefficient can be specified with the ALPHA= option. You can specify one of the following *keywords*:

PL

requests profile-likelihood confidence limits.

WALD

requests confidence limits based on the Wald tests.

BOTH

request both profile-likelihood and Wald confidence limits.

Classification main effects that use parameterizations other than REF, EFFECT, or GLM are ignored. If you need to compute hazard ratios for an effect involved in interactions or nestings, or using some other parameterization, then you should specify a HAZARDRATIO statement for that effect.

ROCLABEL='*label***'**

specifies a label to identify the model that is specified in the MODEL statement (or the final model if a selection method is used) when a concordance analysis or an ROC analysis is conducted. By default, ROCLABEL='Model'.

SELECTION=*method*

specifies the method used to select the model. The *methods* available are as follows:

BACKWARD

B

requests backward elimination.

FORWARD

F

requests forward selection.

NONE

N

fits the complete model specified in the MODEL statement. This is the default value.

SCORE

requests best subset selection, which identifies a specified (in the BEST= option) number of models with the highest score chi-square statistic for all possible model sizes ranging from one explanatory variable to the total number of explanatory variables listed in the MODEL statement. This option is not allowed if an explanatory effect in the MODEL statement contains a CLASS variable.

STEPWISE

S

requests stepwise selection.

For more information, see the section "Effect Selection Methods" on page 7604.

SEQUENTIAL

forces variables to be added to the model in the order specified in the MODEL statement or to be eliminated from the model in the reverse order of that specified in the MODEL statement.

SINGULAR=*value*

specifies the singularity criterion for determining linear dependencies in the set of explanatory variables. The default value is 1E–12.

SLENTRY=*value*

SLE=*value*

specifies the significance level (a value between 0 and 1) for entering an explanatory variable into the model in the FORWARD or STEPWISE method. For all variables not in the model, the one with the smallest p-value is entered if the p-value is less than or equal to the specified significance level. The default value is 0.05.

SLSTAY=*value*

SLS=*value*

specifies the significance level (a value between 0 and 1) for removing an explanatory variable from the model in the BACKWARD or STEPWISE method. For all variables in the model, the one with the largest p-value is removed if the p-value exceeds the specified significance level. The default value is 0.05.

START=*n*

begins the FORWARD, BACKWARD, or STEPWISE selection process with the first *n* effects listed in the MODEL statement. The value of *n* ranges from 0 to *s*, where *s* is the total number of effects in the MODEL statement. The default value of *n* is *s* for the BACKWARD method and 0 for the FORWARD and STEPWISE methods. Note that START=*n* specifies only that the first *n* effects appear in the first model, while INCLUDE=*n* requires that the first *n* effects be included in every model. For the SCORE method, START=*n* specifies that the smallest models contain *n* effects, where *n* ranges from 1 to *s*; the default value is 1. The START= option has no effect when SELECTION=NONE.

STOP=*n*

specifies the maximum (FORWARD method) or minimum (BACKWARD method) number of effects to be included in the final model. The effect selection process is stopped when *n* effects are found. The value of *n* ranges from 0 to *s*, where *s* is the total number of effects in the MODEL statement. The default value of *n* is *s* for the FORWARD method and 0 for the BACKWARD method. For the SCORE method, STOP=*n* specifies that the smallest models contain *n* effects, where *n* ranges from 1 to *s*; the default value of *n* is *s*. The STOP= option has no effect when SELECTION=NONE or STEPWISE.

STOPRES

SR

specifies that the addition and deletion of variables be based on the result of the likelihood score test for testing the joint significance of variables not in the model. This score chi-square statistic is referred to as the residual chi-square. In the FORWARD method, the STOPRES option enters the explanatory variables into the model one at a time until the residual chi-square becomes insignificant (that is, until the *p*-value of the residual chi-square exceeds the SLENTRY= value). In the BACKWARD method, the STOPRES option removes variables from the model one at a time until the residual chi-square becomes significant (that is, until the *p*-value of the residual chi-square becomes less than the SLSTAY= value). The STOPRES option has no effect for the STEPWISE method.

TYPE1

requests that a Type 1 (sequential) analysis of likelihood ratio test be performed. This consists of sequentially fitting models, beginning with the null model and continuing up to the model specified in the MODEL statement. The likelihood ratio statistic for each successive pair of models is computed and displayed in a table.

TYPE3 < (*keywords*) >

requests a Type 3 test or a joint test for each effect that is specified in the MODEL statement. For more information, see the section "Type 3 Tests and Joint Tests" on page 7570. The default is to use the Wald statistic, but you can requests other statistics by specifying one or more of the following *keywords*:

ALL

requests the likelihood ratio tests, the score tests, and the Wald tests. Specifying TYPE3(ALL) is equivalent to specifying TYPE3=(LR SCORE WALD).

NONE

suppresses the Type 3 analysis. Even if the TYPE3 option is not specified, PROC PHREG displays the Wald test results for each model effect if a CLASS variable is involved in a MODEL effect. The NONE option can be used to suppress such display.

LR
requests the likelihood ratio tests. This request is not honored if the COVS option is also specified.

SCORE
requests the score tests. This request is not honored if the COVS option is also specified.

WALD
requests the Wald tests.

TIES=*method*
specifies how to handle ties in the failure time. The following *methods* are available:

BRESLOW
uses the approximate likelihood of Breslow (1974). This is the default value.

DISCRETE
replaces the proportional hazards model by the discrete logistic model

$$\frac{\lambda(t;\mathbf{z})}{1-\lambda(t;\mathbf{z})} = \frac{\lambda_0(t)}{1-\lambda_0(t)}\exp(\mathbf{z}'\boldsymbol{\beta})$$

where $\lambda_0(t)$ and $h(t;\mathbf{z})$ are discrete hazard functions.

EFRON
uses the approximate likelihood of Efron (1977).

EXACT
computes the exact conditional probability under the proportional hazards assumption that all tied event times occur before censored times of the same value or before larger values. This is equivalent to summing all terms of the marginal likelihood for $\boldsymbol{\beta}$ that are consistent with the observed data (Kalbfleisch and Prentice 1980; DeLong, Guirguis, and So 1994).

TIES=EXACT can take a considerable amount of computer resources. If ties are not extensive, TIES=EFRON and TIES=BRESLOW methods provide satisfactory approximations to TIES=EXACT for the continuous time-scale model. In general, Efron's approximation gives results that are much closer to the exact method results than Breslow's approximation does. If the time scale is genuinely discrete, you should use TIES=DISCRETE. TIES=DISCRETE is also required in the analysis of case-control studies when there is more than one case in a matched set. If there are no ties, all four methods result in the same likelihood and yield identical estimates. The default, TIES=BRESLOW, is the most efficient method when there are no ties.

XCONV=*value*
CONVEREPARM=*value*
specifies the relative parameter convergence criterion. Termination requires a small relative parameter change in subsequent iterations,

$$\max_i |\delta_k^{(i)}| < value$$

where

$$\delta_k^{(i)} = \begin{cases} \theta_k^{(i)} - \theta_{k-1}^{(i)} & |\theta_{k-1}^{(i)}| < .01 \\ \frac{\theta_k^{(i)} - \theta_{k-1}^{(i)}}{\theta_{k-1}^{(i)}} & \text{otherwise} \end{cases}$$

where $\theta_k^{(i)}$ is the estimate of the ith parameter at iteration k.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*> < *keyword*=*name* . . . *keyword*=*name*> < / *option*> ;

The OUTPUT statement creates a new SAS data set that contains all the variables in the input data set and optionally contains variables for the estimated linear predictor and its standard error estimate, survival estimates, residuals, and influence statistics.

The estimated linear predictor and its standard error estimate are computed for all observations in which the explanatory variables have no missing values, even if the observed time is missing; if the observed time is not missing, the predicted probability is computed at the observed time. By adding observations that have a missing censoring indicator to the input data set, you can compute predicted probabilities for new observations or for settings of explanatory variables and observed times that are not present in the data without affecting the model. Alternatively, you can use the BASELINE statement to compute predicted survival probabilities for new observations.

No data set is created for the OUTPUT statement if the model contains any time-dependent variables that are defined by programming statements.

Table 92.11 summarizes the options available in the OUTPUT statement. The statistic and diagnostic *keywords* specify the statistics to be included in the output data set and name the new variables that contain the statistics.

Table 92.11 OUTPUT Statement Options

Option	Description
METHOD=	Specifies the method to use to estimate the survival probabilities
OUT=	Names the output data set
Statistic Keywords	
ATRISK=	Names the variable that contains the number of subjects at risk
CIF=	Names the variable that contains the cumulative incidence probability
LOGLOGS=	Names the variable that contains the log of the negative log of survival probability
LOGSURV=	Names the variable that contains the log of survival probability
XBETA=	Names the variable that contains the linear predictor
STDXBETA=	Names the variable that contains standard error of the linear predictor
SURVIVAL=	Names the variable that contains the survival probability
Diagnostic Keywords	
DFBETA=	Requests the standardized deletion parameter differences
LD=	Names the variable that contains the likelihood displacement diagnostic
LMAX=	Names the variable that contains the relative influence diagnostic
RESDEV=	Names the variable that contains the deviance residuals
RESMART=	Names the variable that contains the martingale residuals
RESSCH=	Requests the Schoenfeld residuals
RESSCO=	Requests the score residuals
WTRESSCH=	Requests the weighted Schoenfeld residuals

OUT=*SAS-data-set*

names the output data set. If you omit the OUT= option, the OUTPUT data set is created and given a default name by using the DATA*n* convention. For more information, see the section "OUT= Output Data Set in the OUTPUT Statement" on page 7622.

METHOD=*method*

specifies the method used to compute the survivor function estimates. For more information, see the section "Survivor Function Estimators" on page 7596. This option appears in the OUTPUT statement after a slash (/). You can specify the following *methods*:

BRESLOW

CH

EMP

computes the cumulative hazard function estimate of the survivor function; that is, the survivor function is estimated by exponentiating the negative cumulative hazard function.

FH

computes the Fleming-Harrington (FH) estimates of the survivor function. The FH estimator is a tie-breaking modification of the Breslow estimator. If there are no tied event times, this estimator is the same as the Breslow estimator.

PL

computes the product-limit estimates of the survivor function. This estimator is not available if you use the model syntax that allows two time variables for the counting process style of input; in such a case, the Breslow estimator (METHOD=BRESLOW) is used instead.

By default, METHOD=BRESLOW.

The following list describes the statistic and diagnostic *keywords*:

ATRISK=*name*

names the variable that contains the number of subjects at risk at the observed time (or at the right endpoint of the at-risk interval when a counting-process specification is used in the MODEL statement, as described in the section "Counting Process Style of Input" on page 7551).

CIF=*name*

names the variable that contains the cumulative incidence probabilities at the observed times. For more information, see the section "Cumulative Incidence Prediction" on page 7562.

DFBETA=_ALL_ | *name-list*

requests the approximate changes in the parameter estimates $(\hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{(j)})$ when the jth observation is omitted. These variables are a weighted transform of the score residual variables and are useful in assessing local influence and in computing robust variance estimates. You can specify this option in one of the following ways:

name-list specifies up to s variable names, where s is the number of regression parameters of the model that is specified in the MODEL statement. The first variable contains the changes in the first regression parameter, the second variable contains the changes for the second regression parameter, and so on.

ALL requests the changes for all parameters and names them DFBETA_*xxx*, where *xxx* is the name of the model regression parameter that is formed from the input variable names (concatenated with the appropriate categories if a classification variable is used). For example, suppose that the model contains a continuous variable X and a CLASS variable Gender with two levels ("Female" and "Male") and that Gender has a GLM parameterization. Three statistics are produced: DFBETA_X, DFBETA_GenderFemale, and DFBETA_GenderMale.

If an effect that is specified in the MODEL statement is not included in the final model, the corresponding statistics are set to missing. For more information, see the section "Diagnostics Based on Weighted Residuals" on page 7587.

LD=*name*
names the variable that contains the approximate likelihood displacement when the observation is left out. This diagnostic can be used to assess the impact of each observation on the overall fit of the model. For more information, see the section "Influence of Observations on Overall Fit of the Model" on page 7588.

LMAX=*name*
names the variable that contains the relative influence of observations on the overall fit of the model. This diagnostic is useful in assessing the sensitivity of the model's fit to each observation. For more information, see the section "Influence of Observations on Overall Fit of the Model" on page 7588.

LOGLOGS=*name*
names the variable that contains the log of the negative log of the variable named in the SURVIVAL= option.

LOGSURV=*name*
names the variable that contains the log of variable named in the SURVIVAL= option.

RESDEV=*name*
names the variable that contains the deviance residuals. This variable is a transform of the variable named in the RESMART= option and can achieve a more symmetric distribution. For more information, see the section "Residuals" on page 7585.

RESMART=*name*
names the variable that contains the martingale residuals. The martingale residual at the observed time t can be interpreted as the difference over $[0, t]$ in the observed number of events minus the expected number of events. For more information, see the section "Residuals" on page 7585.

RESSCH=_ALL_ | *name-list*
requests Schoenfeld residuals, which are useful in assessing the proportional hazards assumption. Schoenfeld residuals are computed only at uncensored times and are missing for censored times. You can specify this option in one of the following ways:

name-list specifies up to s variable names, where s is the number of regression parameters of the model that is specified in the MODEL statement. The first variable contains the Schoenfeld residuals for the first regression parameter, the second variable contains the Schoenfeld residuals for the second regression parameter, and so on.

ALL requests Schoenfeld residuals for all regression parameters and names them RESSCH_*xxx*, where *xxx* is the name of the model regression parameter that is formed from the input variable names (concatenated with the appropriate categories if a classification variable is used). For example, suppose that the model contains a continuous variable X and a CLASS variable Gender with two levels ("Female" and "Male") and that Gender has a GLM parameterization. Three statistics are produced: RESSCH_X, RESSCH_GenderFemale, and RESSCH_GenderMale.

If an effect in the MODEL statement is not included in the final model, the corresponding Schoenfeld residuals are set to missing. For more information, see the section "Residuals" on page 7585.

RESSCO=_ALL_ | *name-list*

requests the score residuals, which are a decomposition of the first partial derivative of the log likelihood. These residuals can be used to assess the leverage that is exerted by each subject in the parameter estimates. They are also useful in constructing robust sandwich variance estimates. You can specify this option in one of the following ways:

name-list specifies up to *s* variable names, where *s* is the number of regression parameters of the model that is specified in the MODEL statement. The first variable contains the score residuals for the first regression parameter, the second variable contains the score residuals for the second parameter, and so on.

ALL requests score residuals for all regression parameters and names them RESSCO_*xxx*, where *xxx* is the name of the model regression parameter that is formed from the input variable names (concatenated with the appropriate categories if a classification variable is used). For example, suppose that the model contains a continuous variable X and a CLASS variable Gender with two levels ("Female" and "Male") and that Gender has a GLM parameterization. Three statistics are produced: RESSCO_X, RESSCO_GenderFemale, and RESSCO_GenderMale.

If an effect in the MODEL statement is not included in the final model, the corresponding score residuals are set to missing. For more information, see the section "Residuals" on page 7585.

STDXBETA=*name*

names the variable that contains the standard error estimates of linear predictor that is specified in the XBETA= option.

SURVIVAL=*name*

names the variable that contains the predicted survival probabilities at the observed times. For more information, see the section "Survivor Function Estimators" on page 7596.

WTRESSCH=_ALL_ | *name-list*

requests the weighted Schoenfeld residuals, which are useful in investigating the nature of nonproportionality if the proportional hazard assumption does not hold. You can specify this option in one of the following ways:

name-list specifies up to *s* variable names, where *s* is the number of regression parameters of the model that is specified in the MODEL statement. The first variable contains the weighted Schoenfeld residuals for the first regression parameter, the second variable contains the weighted Schoenfeld residuals for the second regression parameter, and so on.

ALL requests weighted Schoenfeld residuals for all regression parameters and names them WTRESSCH_*xxx*, where *xxx* is the name of the model regression parameter that is formed from the input variable names (concatenated with the appropriate categories if a classification variable is used). For example, suppose that the model contains a continuous variable X and a CLASS variable Gender with two levels ("Female" and "Male") and that Gender has a GLM parameterization. Three statistics are produced: WTRESSCH_X, WTRESSCH_GenderFemale, and WTRESSCH_GenderMale.

If an effect in the MODEL statement is not included in the final model, the corresponding weighted Schoenfeld residuals are set to missing. For more information, see the section "Diagnostics Based on Weighted Residuals" on page 7587.

XBETA=*name*
names the variable that contains the estimates of the linear predictor.

Programming Statements

Programming statements are used to create or modify the values of the explanatory variables in the MODEL statement. They are especially useful in fitting models with time-dependent explanatory variables. Programming statements can also be used to create explanatory variables that are not time dependent. For example, you can create indicator variables from a categorical variable and incorporate them into the model. PROC PHREG programming statements cannot be used to create or modify the values of the response variable, the censoring variable, the frequency variable, or the strata variables.

The following DATA step statements are available in PROC PHREG:

ABORT;
ARRAY *arrayname* < **[** *dimensions* **]** > < **$** > < *variables-and-constants* >**;**
CALL *name* < **(***expression* < **,** *expression . . .* >**)** >**;**
DELETE;
DO < *variable* **=** *expression* < **TO** *expression* > < **BY** *expression* > >
< **,** *expression* < **TO** *expression* > < **BY** *expression* > > . . .
< **WHILE** *expression* > < **UNTIL** *expression* >**;**
END;
GOTO *statement-label***;**
IF *expression***;**
IF *expression* **THEN** *program-statement***;**
ELSE *program-statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement-label***;**
PUT < *variable* > < **=** > . . . **;**
RETURN;
SELECT < **(***expression***)** >**;**
STOP;
SUBSTR(*variable***,** *index***,** *length***)=** *expression***;**
WHEN (*expression***)***program-statement***;**
OTHERWISE *program-statement***;**

By default, the PUT statement in PROC PHREG writes results to the Output window instead of the Log window. If you want the results of the PUT statements to go to the Log window, add the following statement before the PUT statements:

```
FILE LOG;
```

DATA step functions are also available. Use these programming statements the same way you use them in the DATA step. For detailed information, see *SAS DATA Step Statements: Reference*.

Consider the following example of using programming statements in PROC PHREG. Suppose blood pressure is measured at multiple times during the course of a study investigating the effect of blood pressure on some survival time. By treating the blood pressure as a time-dependent explanatory variable, you are able to use the value of the most recent blood pressure at each specific point of time in the modeling process rather than using the initial blood pressure or the final blood pressure. The values of the following variables are recorded for each patient, if they are available. Otherwise, the variables contain missing values.

Time	survival time
Censor	censoring indicator (with 0 as the censoring value)
BP0	blood pressure on entry to the study
T1	time 1
BP1	blood pressure at T1
T2	time 2
BP2	blood pressure at T2

The following programming statements create a variable BP. At each time T, the value of BP is the blood pressure reading for that time, if available. Otherwise, it is the last blood pressure reading.

```
proc phreg;
   model Time*Censor(0)=BP;
   BP = BP0;
   if Time>=T1 and T1^=. then BP=BP1;
   if Time>=T2 and T2^=. then BP=BP2;
run;
```

For other illustrations of using programming statements, see the section "Classical Method of Maximum Likelihood" on page 7468 and Example 92.6.

RANDOM Statement

RANDOM *variable* < / *options* > **;**

The RANDOM statement enables you to fit a shared frailty model for clustered data. For more information about the frailty model, see the section "The Frailty Model" on page 7555. The *variable* that represents the clusters must be a CLASS variable (declared in the CLASS statement).

The following *options* can be specified in the RANDOM statement:

ABSPCONV=*r*

specifies an absolute variance estimate convergence criterion for the doubly iterative estimation process. The PHREG procedure applies this criterion to the variance parameter estimate of the random effects. Suppose $\hat{\theta}^{(j)}$ denotes the estimate of the variance parameter at the *j*th optimization. By default, PROC PHREG examines the relative change in the variance estimate between optimizations (see the PCONV= option). The purpose of the ABSPCONV= criterion is to stop the doubly iterative process when the absolute change $|\hat{\theta}^{(j)} - \hat{\theta}^{(j-1)}|$ is less than the tolerance criterion *r*. This convergence criterion does not affect the convergence criteria applied within any individual optimization. In order to change the convergence behavior within an individual optimization, you can use the ABSCONV=, ABSFCONV=, ABSGCONV=, ABSXCONV=, FCONV=, or GCONV= option in the NLOPTIONS statement.

ALPHA=*value*

specifies the α level of the confidence limits for the random effects. The default is the value of the ALPHA= option in the PROC PHREG statement, or 0.05 if that option is not specified. This option is ignored if the SOLUTION option is not also specified.

DIST=GAMMA | LOGNORMAL

specifies the distribution of the shared frailty. DIST=GAMMA specifies a gamma frailty model. DIST=LOGNORMAL specifies a lognormal frailty model; that is, the log-frailty random variable has a normal distribution with mean zero. The default is DIST=LOGNORMAL.

METHOD=REML | ML

specifies the estimation method for the variance parameter of the normal random effects. METHOD=REML performs the residual maximum likelihood; METHOD=ML performs maximum likelihood. This option is ignored for the gamma frailty model. The default is METHOD=REML.

NOCLPRINT

suppresses the display of the "Class Level Information for Random Effects" table.

PCONV=*r*

specifies the variance estimate convergence criterion for the doubly iterative estimation process. The PHREG procedure applies this criterion to the variance estimate of the random effects. Suppose $\hat{\theta}^{(j)}$ denotes the estimate of variance at the *j*th optimization. The procedure terminates the doubly iterative process if the relative change

$$2 \times \frac{|\hat{\theta}^{(j)} - \hat{\theta}^{(j-1)}|}{|\hat{\theta}^{(j)}| + |\hat{\theta}^{(j-1)}|}$$

is less than *r*. To check an absolute convergence criterion in addition, you can specify the ABSPCONV= option in the RANDOM statement. The default value for *r* is 1E–4. This convergence criterion does not affect the convergence criteria applied within any individual optimization. In order to change the convergence behavior within an individual optimization, you can use the ABSCONV=, ABSFCONV=, ABSGCONV=, ABSXCONV=, FCONV=, or GCONV= option in the NLOPTIONS statement.

SOLUTION <(*number-list***)>**

displays statistical measures of the random-effect parameters. The behavior of this option depends on whether you also specify the BAYES statement:

- When you do not specify a BAYES statement, this option displays point estimates and confidence intervals for the random components and for the frailties.
- When you also specify a BAYES statement, this option displays the summary statistics and diagnostics of the random-effect parameters. Optionally, you can specify a *number-list* of indices of the random-effect parameters for which the summary statistics and diagnostics are to be displayed. For example, to display the summary statistics and diagnostics for the second to the fifth random-effect parameters, specify

  ```
  SOLUTION(2 to 5)
  ```

  If you specify SOLUTION without a list, summary statistics and diagnostics are displayed for each random-effect parameter.

INITIALVARIANCE=*value*

INITIAL=*value*

specifies an initial value of the dispersion parameter. For the lognormal frailty model, the dispersion parameter represents the variance of the normal random effect; for the gamma frailty model, it represents the variance of the gamma frailty. The default is INITIAL=1.

ROC Statement

ROC <'*label*'> *specification* **;**

The ROC statement specifies a model to be used in the concordance analysis or ROC analysis. There is no limit to the number of ROC statements that you can have. You can specify a *label* in quotation marks to identify an ROC statement. If you do not specify a *label*, the model in the *i*th ROC statement is labeled "ROC*i*".

You can specify one of the following *specifications*:

effect-list
specifies a list of effects that have previously been specified in the MODEL statement.

PRED=*variable*
specifies a *variable* in the input data set. The *variable* does not have to be specified in the MODEL statement. For an example, see Example 92.16.

SOURCE=*item-store-name*
specifies a one- or two-level name of an existing item store.

The PRED= option and the SOURCE= option enable you to input a model that is produced outside PROC PHREG; for example, you can fit a parametric survival model by using PROC LIFEREG, save the model in an item store, and then specify the item-store name in the SOURCE= option in a ROC statement to produce an ROC analysis of the imported model.

STRATA Statement

STRATA *variable* <(*list*)> <. . . *variable* <(*list*)>> </ *option*> **;**

The proportional hazards assumption might not be realistic for all data. If so, it might still be reasonable to perform a stratified analysis. The STRATA statement names the variables that determine the stratification. Strata are formed according to the nonmissing values of the STRATA variables unless the MISSING option is specified. In the STRATA statement, *variable* is a variable with values that are used to determine the strata levels, and *list* is an optional list of values for a numeric variable. Multiple variables can appear in the STRATA statement.

The values for *variable* can be formatted or unformatted. If the variable is a character variable, or if the variable is numeric and no list appears, then the strata are defined by the unique values of the variable. If the variable is numeric and is followed by a list, then the levels for that variable correspond to the intervals defined by the list. The corresponding strata are formed by the combination of levels and unique values. The list can include numeric values separated by commas or blanks, *value* to *value* by *value* range specifications, or combinations of these.

For example, the specification

```
strata age (5, 10 to 40 by 10) sex;
```

indicates that the levels for age are to be less than 5, 5 to 10, 10 to 20, 20 to 30, 30 to 40, and greater than 40. (Note that observations with exactly the cutpoint value fall into the interval preceding the cutpoint.) Thus, with the sex variable, this STRATA statement specifies 12 strata altogether.

The following *option* can be specified in the STRATA statement after a slash (/):

MISSING
allows missing values ('.' for numeric variables and blanks for character variables) as valid STRATA variable values. Otherwise, observations with missing STRATA variable values are deleted from the analysis.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

TEST Statement

< *label* **:** > **TEST** *equation* < **,** . . . **,** *equation* > < / *options* > **;**

The TEST statement tests linear hypotheses about the regression coefficients. PROC PHREG performs a Wald test for the joint hypothesis specified in a single TEST statement. Each equation specifies a linear hypothesis; multiple equations (rows of the joint hypothesis) are separated by commas. The *label*, which must be a valid SAS name, is used to identify the resulting output and should always be included. You can submit multiple TEST statements.

The form of an *equation* is as follows:

term < ± *term* . . . > < = < ± *term* < ± *term* . . . > > >

where *term* is a variable or a constant or a constant times a variable. The variable is any explanatory variable in the MODEL statement. When no equal sign appears, the expression is set to 0. The following program illustrates possible uses of the TEST statement:

```
proc phreg;
   model time= A1 A2 A3 A4;
   Test1: test A1, A2;
   Test2: test A1=0,A2=0;
   Test3: test A1=A2=A3;
   Test4: test A1=A2,A2=A3;
run;
```

Note that the first and second TEST statements are equivalent, as are the third and fourth TEST statements.

The following *options* can be specified in the TEST statement after a slash (/):

AVERAGE

enables you to assess the average effect of the variables in the given TEST statement. An overall estimate of the treatment effect is computed as a weighted average of the treatment coefficients as illustrated in the following statement:

```
TREATMENT: test trt1, trt2, trt3, trt4 / average;
```

Let β_1, β_2, β_3, and β_4 be corresponding parameters for trt1, trt2, trt3, and trt4, respectively. Let $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3, \hat{\beta}_4)'$ be the estimated coefficient vector and let $\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})$ be the corresponding variance estimate. Assuming $\beta_1 = \beta_2 = \beta_3 = \beta_4$, let $\bar{\beta}$ be the average treatment effect. The effect is estimated by $\mathbf{c}'\hat{\boldsymbol{\beta}}$, where $\mathbf{c} = [\mathbf{1}_4'\hat{\mathbf{V}}^{-1}(\hat{\boldsymbol{\beta}})\mathbf{1}_4]^{-1}\hat{\mathbf{V}}^{-1}(\hat{\boldsymbol{\beta}})\mathbf{1}_4$ and $\mathbf{1}_4 = (1, 1, 1, 1)'$. A test of the null hypothesis $H_0 : \bar{\beta} = 0$ is also included, which is more sensitive than the multivariate test for testing the null hypothesis $H_0 : \beta 1 = \beta 2 = \beta 3 = \beta 4 = 0$.

E

specifies that the linear coefficients and constants be printed. When the AVERAGE option is specified along with the E option, the optimal weights of the average effect are also printed in the same tables as the coefficients.

PRINT

displays intermediate calculations. This includes $\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{L}'$ bordered by $(\mathbf{L}\hat{\boldsymbol{\beta}} - \mathbf{c})$, and $[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{L}']^{-1}$ bordered by $[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{L}']^{-1}(\mathbf{L}\hat{\boldsymbol{\beta}} - \mathbf{c})$, where $\mathbf{L}$ is a matrix of linear coefficients and $\mathbf{c}$ is a vector of constants.

For more information, see the section "Using the TEST Statement to Test Linear Hypotheses" on page 7573.

WEIGHT Statement

WEIGHT *variable* < / *option* > **;**

The *variable* in the WEIGHT statement identifies the variable in the input data set that contains the case weights. When the WEIGHT statement appears, each observation in the input data set is weighted by the value of the WEIGHT variable. The WEIGHT values can be nonintegral and are not truncated. Observations with negative, zero, or missing values for the WEIGHT variable are not used in the model fitting. When the WEIGHT statement is not specified, each observation is assigned a weight of 1. The WEIGHT statement is available for TIES=BRESLOW and TIES=EFRON only.

The following *option* can be specified in the WEIGHT statement after a slash (/):

NORMALIZE

NORM

causes the weights specified by the WEIGHT *variable* to be normalized so that they add up the actual sample size. With this option, the estimated covariance matrix of the parameter estimators is invariant to the scale of the WEIGHT variable.

Details: PHREG Procedure

Failure Time Distribution

Let T be a nonnegative random variable representing the failure time of an individual from a homogeneous population. The survival distribution function (also known as the survivor function) of T is written as

$$S(t) = \Pr(T \geq t)$$

A mathematically equivalent way of specifying the distribution of T is through its hazard function. The hazard function $\lambda(t)$ specifies the instantaneous failure rate at t. If T is a continuous random variable, $\lambda(t)$ is expressed as

$$\lambda(t) = \lim_{\Delta t \to 0^+} \frac{\Pr(t \leq T < t + \Delta t \mid T \geq t)}{\Delta t} = \frac{f(t)}{S(t)}$$

where $f(t)$ is the probability density function of T. If T is discrete with masses at $x_1 < x_2 < \ldots$, then survivor function is given by

$$S(t) = \sum_{x_j \leq t} \Pr(T = x_j) = \sum_j \Pr(T = j)\delta(t - x_j)$$

where $\delta(u)$=0 if $u < 0$ and $\delta(u)$=1 otherwise. The discrete hazards are given by

$$\lambda_j = \Pr(T = x_j \mid T \geq x_j) = \frac{\Pr(T = x_j)}{S(x_j)}, \quad j = 1, 2, \ldots$$

Time and CLASS Variables Usage

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/phrtime.sas.)

The following DATA step creates an artificial data set, Test, to be used in this section. There are four variables in Test: the variable T contains the failure times; the variable Status is the censoring indicator variable with the value 1 for an uncensored failure time and the value 0 for a censored time; the variable A is a categorical variable with values 1, 2, and 3 representing three different categories; and the variable MirrorT is an exact copy of T.

```
data Test;
   input T Status A @@;
   MirrorT = T;
   datalines;
 23        1      1    7        0      1
 23        1      1   10        1      1
 20        0      1   13        0      1
 24        1      1   10        1      1
 18        1      2    6        1      2
 18        0      2    6        1      2
 13        0      2   13        1      2
  9        0      2   15        1      2
  8        1      3    6        1      3
 12        0      3    4        1      3
 11        1      3    8        1      1
  6        1      3    7        1      3
  7        1      3   12        1      3
  9        1      2   15        1      2
  3        1      2   14        0      3
  6        1      1   13        1      2
;
```

Time Variable on the Right Side of the MODEL Statement

When the time variable is explicitly used in an explanatory effect in the MODEL statement, the effect is *not* time-dependent. In the following specification, T is the time variable, but T does not play the role of the time variable in the explanatory effect T*A:

```
proc phreg data=Test;
   class A;
   model T*Status(0)=T*A;
run;
```

The parameter estimates of this model are shown in Figure 92.12.

Figure 92.12 T*A Effect

The PHREG Procedure

Analysis of Maximum Likelihood Estimates								
Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
T*A	**1**	1	-0.16549	0.05042	10.7734	0.0010	.	A 1 * T
T*A	**2**	1	-0.11852	0.04181	8.0344	0.0046	.	A 2 * T

To verify that the effect T*A in the MODEL statement is not time-dependent, T is replaced by MirrorT, which is an exact copy of T, as in the following statements:

```
proc phreg data=Test;
   class A;
   model T*Status(0)=A*MirrorT;
run;
```

The results of fitting this model (Figure 92.13) are identical to those of the previous model (Figure 92.12), except for the parameter names and labels. The effect A*MirrorT is not time-dependent, so neither is A*T.

Figure 92.13 T*A Effect

The PHREG Procedure

Analysis of Maximum Likelihood Estimates								
Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
MirrorT*A	**1**	1	-0.16549	0.05042	10.7734	0.0010	.	A 1 * MirrorT
MirrorT*A	**2**	1	-0.11852	0.04181	8.0344	0.0046	.	A 2 * MirrorT

CLASS Variables and Programming Statements

In PROC PHREG, the levels of CLASS variables are determined by the CLASS statement and the input data and are not affected by user-supplied programming statements. Consider the following statements, which produce the results in Figure 92.14. Variable A is declared as a CLASS variable in the CLASS statement. By default, the reference parameterization is used with A=3 as the reference level. Two regression coefficients are estimated for the two dummy variables of A.

```
proc phreg data=Test;
   class A;
   model T*Status(0)=A;
run;
```

Figure 92.14 shows the dummy variables of A and the regression coefficients estimates.

Figure 92.14 Design Variable and Regression Coefficient Estimates

The PHREG Procedure

Class Level Information			
Class	Value	Design Variables	
A	1	1	0
	2	0	1
	3	0	0

Analysis of Maximum Likelihood Estimates								
Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
A	1	1	-1.40925	0.64802	4.7293	0.0297	0.244	A 1
A	2	1	-0.65705	0.51764	1.6112	0.2043	0.518	A 2

Now consider the programming statement that attempts to change the value of the CLASS variable A as in the following specification:

```
proc phreg data=Test;
   class A;
   model T*Status(0)=A;
   if A=3 then A=2;
run;
```

Results of this analysis are shown in Figure 92.15 and are identical to those in Figure 92.14. The **`if A=3 then A=2`** programming statement has no effects on the design variables for A, which have already been determined.

Figure 92.15 Design Variable and Regression Coefficient Estimates

The PHREG Procedure

Class Level Information			
Class	Value	Design Variables	
A	1	1	0
	2	0	1
	3	0	0

Analysis of Maximum Likelihood Estimates								
Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
A	1	1	-1.40925	0.64802	4.7293	0.0297	0.244	A 1
A	2	1	-0.65705	0.51764	1.6112	0.2043	0.518	A 2

Additionally any variable used in a programming statement that has already been declared in the CLASS statement is *not* treated as a collection of the corresponding design variables. Consider the following statements:

```
proc phreg data=Test;
   class A;
   model T*Status(0)=A X;
   X=T*A;
run;
```

The CLASS variable A generates two design variables as explanatory variables. The variable X that is created by the **X=T*A** programming statement is a single time-dependent covariate whose values are evaluated using the exact values of A that exist in the data, not the dummy-coded values that represent the levels of A. In the Test data set, A assumes the values of 1, 2, and 3, and these are the exact values that are used in producing X. If A is a character variable, the programming statement converts the character value of A to numeric before carrying out the multiplication. If the values of A were Bird, Cat, and Dog, the programming statement X=T*A would produce an error for failing to convert the character values to numeric values.

Figure 92.16 Single Time-Dependent Variable X*A

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
A	**1**	1	0.15798	1.69338	0.0087	0.9257	1.171	A 1
A	**2**	1	0.00898	0.87573	0.0001	0.9918	1.009	A 2
X		1	0.09268	0.09535	0.9448	0.3311	1.097	

To generalize the simple test of proportional hazard assumption for the design variables of A (as in the section the "Classical Method of Maximum Likelihood" on page 7468), you specify the following statements, which are not the same as in the preceding program or as in the specification in the section "Time Variable on the Right Side of the MODEL Statement" on page 7546:

```
proc phreg data=Test;
   class A;
   model T*Status(0)=A X1 X2;
   X1= T*(A=1);
   X2= T*(A=2);
run;
```

The Boolean parenthetical expressions (A=1) and (A=2) resolve to a value of 1 or 0, depending on whether the expression is true or false, respectively.

Results of this test are shown in Figure 92.17.

Figure 92.17 Simple Test of Proportional Hazards Assumption

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
A	**1**	1	-0.00766	1.69435	0.0000	0.9964	0.992	A 1
A	**2**	1	-0.88132	1.64298	0.2877	0.5917	0.414	A 2
X1		1	-0.15522	0.20174	0.5920	0.4417	0.856	
X2		1	0.01155	0.18858	0.0037	0.9512	1.012	

In general, when your model contains a categorical explanatory variable that is time-dependent, it might be necessary to use hardcoded dummy variables to represent the categories of the categorical variable. Alternatively, you might consider using the counting-process style of input where you break up the covariate history of an individual into a number of records with nonoverlapping start and stop times and declare the categorical time-dependent variable in the CLASS statement.

Partial Likelihood Function for the Cox Model

Let $\mathbf{Z}_l(t)$ denote the vector explanatory variables for the lth individual at time t. Let $t_1 < t_2 < \ldots < t_k$ denote the k distinct, ordered event times. Let d_i denote the multiplicity of failures at t_i; that is, d_i is the size of the set $\mathcal{D}_i$ of individuals that fail at t_i. Let w_l be the weight associated with the lth individual. Using this notation, the likelihood functions used in PROC PHREG to estimate $\boldsymbol{\beta}$ are described in the following sections.

Continuous Time Scale

Let $\mathcal{R}_i$ denote the risk set just before the ith ordered event time t_i. Let $\mathcal{R}_i^*$ denote the set of individuals whose event or censored times exceed t_i or whose censored times are equal to t_i.

Exact Likelihood

$$L_1(\boldsymbol{\beta}) = \prod_{i=1}^{k} \left\{ \int_0^{\infty} \prod_{j \in \mathcal{D}_i} \left[1 - \exp\left(-\frac{e^{\boldsymbol{\beta}'\mathbf{Z}_j(t_i)}}{\sum_{l \in \mathcal{R}_i^*} e^{\boldsymbol{\beta}'\mathbf{Z}_l(t_i)}} t \right) \right] \exp(-t)\,dt \right\}$$

Breslow Likelihood

$$L_2(\boldsymbol{\beta}) = \prod_{i=1}^{k} \frac{e^{\boldsymbol{\beta}' \sum_{j \in \mathcal{D}_i} \mathbf{Z}_j(t_i)}}{\left[\sum_{l \in \mathcal{R}_i} e^{\boldsymbol{\beta}'\mathbf{Z}_l(t_i)} \right]^{d_i}}$$

Incorporating weights, the Breslow likelihood becomes

$$L_2(\boldsymbol{\beta}) = \prod_{i=1}^{k} \frac{e^{\boldsymbol{\beta}' \sum_{j \in \mathcal{D}_i} w_j \mathbf{Z}_j(t_i)}}{\left[\sum_{l \in \mathcal{R}_i} w_l e^{\boldsymbol{\beta}'\mathbf{Z}_l(t_i)} \right]^{\sum_{j \in \mathcal{D}_i} w_j}}$$

Efron Likelihood

$$L_3(\boldsymbol{\beta}) = \prod_{i=1}^{k} \frac{e^{\boldsymbol{\beta}' \sum_{j \in \mathcal{D}_i} \mathbf{Z}_j(t_i)}}{\prod_{j=1}^{d_i} \left(\sum_{l \in \mathcal{R}_i} e^{\boldsymbol{\beta}' \mathbf{Z}_l(t_i)} - \frac{j-1}{d_i} \sum_{l \in \mathcal{D}_i} e^{\boldsymbol{\beta}' \mathbf{Z}_l(t_i)} \right)}$$

Incorporating weights, the Efron likelihood becomes

$$L_3(\boldsymbol{\beta}) = \prod_{i=1}^{k} \frac{e^{\boldsymbol{\beta}' \sum_{j \in \mathcal{D}_i} w_j \mathbf{Z}_j(t_i)}}{\left[\prod_{j=1}^{d_i} \left(\sum_{l \in \mathcal{R}_i} w_l e^{\boldsymbol{\beta}' \mathbf{Z}_l(t_i)} - \frac{j-1}{d_i} \sum_{l \in \mathcal{D}_i} w_l e^{\boldsymbol{\beta}' \mathbf{Z}_l(t_i)} \right) \right]^{\frac{1}{d_i} \sum_{j \in \mathcal{D}_i} w_j}}$$

Discrete Time Scale

Let $\mathcal{Q}_i$ denote the set of all subsets of d_i individuals from the risk set $\mathcal{R}_i$. For each $\mathbf{q} \in \mathcal{Q}_i$, $\mathbf{q}$ is a d_i-tuple $(q_1, q_2, \ldots, q_{d_i})$ of individuals who might have failed at t_i.

Discrete Logistic Likelihood

$$L_4(\boldsymbol{\beta}) = \prod_{i=1}^{k} \frac{e^{\boldsymbol{\beta}' \sum_{j \in \mathcal{D}_i} \mathbf{Z}_j(t_i)}}{\sum_{\mathbf{q} \in \mathcal{Q}_i} e^{\boldsymbol{\beta}' \sum_{l=1}^{d_i} \mathbf{Z}_{q_l}(t_i)}}$$

The computation of $L_4(\boldsymbol{\beta})$ and its derivatives is based on an adaptation of the recurrence algorithm of Gail, Lubin, and Rubinstein (1981) to the logarithmic scale. When there are no ties on the event times (that is, $d_i \equiv 1$), all four likelihood functions $L_1(\boldsymbol{\beta})$, $L_2(\boldsymbol{\beta})$, $L_3(\boldsymbol{\beta})$, and $L_4(\boldsymbol{\beta})$ reduce to the same expression. In a stratified analysis, the partial likelihood is the product of the partial likelihood functions for the individual strata.

Counting Process Style of Input

In the counting process formulation, data for each subject are identified by a triple $\{N, Y, \mathbf{Z}\}$ of counting, at-risk, and covariate processes. Here, $N(t)$ indicates the number of events that the subject experiences over the time interval $(0, t]$; $Y(t)$ indicates whether the subject is at risk at time t (one if at risk and zero otherwise); and $\mathbf{Z}(t)$ is a vector of explanatory variables for the subject at time t. The sample path of N is a step function with jumps of size +1 at the event times, and $N(0) = 0$. Unless $\mathbf{Z}(t)$ changes continuously with time, the data for each subject can be represented by multiple observations, each identifying a semiclosed time interval $(t_1, t_2]$, the values of the explanatory variables over that interval, and the event status at t_2. The subject remains at risk during the interval $(t_1, t_2]$, and an event might occur at t_2. Values of the explanatory variables for the subject remain unchanged in the interval. This style of data input was originated by Therneau (1994).

For example, a patient has a tumor recurrence at weeks 3, 10, and 15 and is followed up to week 23. Consider three fixed explanatory variables Trt (treatment), Number (initial tumor number), and Size (initial tumor size),

and one time-dependent covariate Z that represents a hormone level. The value of Z might change during the follow-up period. The data for this patient are represented by the following four observations:

T1	T2	Status	Trt	Number	Size	Z
0	3	1	1	1	3	12.3
3	10	1	1	1	3	14.7
10	15	1	1	1	3	13.8
15	23	0	1	1	3	15.5

Here (T1,T2] contains the at-risk intervals. The variable Status indicates whether a recurrence has occurred at T2; a value of 1 indicates a tumor recurrence, and a value of 0 indicates nonrecurrence. The following statements fit a multiplicative hazards model with baseline covariates Trt, Number, and Size, and a time-varying covariate Z.

```
proc phreg;
   model (T1,T2) * Status(0) = Trt Number Size Z;
run;
```

Another useful application of the counting process formulation is delayed entry of subjects into the risk set. For example, in studying the mortality of workers exposed to a carcinogen, the survival time is chosen to be the worker's age at death by malignant neoplasm. Any worker joining the workplace at a later age than a given event failure time is not included in the corresponding risk set. The variables of a worker consist of Entry (age at which the worker entered the workplace), Age (age at death or age censored), Status (an indicator of whether the observation time is censored, with the value 0 identifying a censored time), and X1 and X2 (explanatory variables thought to be related to survival). The specification for such an application is as follows:

```
proc phreg;
   model (Entry, Age) * Status(0) = X1 X2;
run;
```

Alternatively, you can use a time-dependent variable to control the risk set, as illustrated in the following specification:

```
proc phreg;
   model Age * Status(0) = X1 X2;
   if Age < Entry then X1= .;
run;
```

Here, X1 becomes a time-dependent variable. At a given death time t, the value of X1 is reevaluated for each subject with Age $\geq t$; subjects with Entry $> t$ are given a missing value in X1 and are subsequently removed from the risk set. Computationally, this approach is not as efficient as the one that uses the counting process formulation.

Left-Truncation of Failure Times

Left-truncation occurs when individuals are not observed at the natural time origin of the phenomenon under study but come under observation at some known later time (called the left-truncation time). The risk set just prior to an event time does not include individuals whose left-truncation times exceed the given event time. Thus, any contribution to the likelihood must be conditional on the truncation limit having been exceeded.

You use the ENTRY= option to specify the variable that represents the left-truncation time. Suppose T1 and T2 represent the left-truncation time and the survival time, respectively. To account for left-truncation, you specify the following statements:

```
proc phreg;
   model T2*Dead(0)=X1-X10/entry=T1;
   title 'The ENTRY= option is Specified';
run;
```

Equivalently, you can use the counting process style of input for left-truncation:

```
proc phreg;
   model (T1,T2)*Dead(0)=X1-X10;
   title 'Counting Process Style of Input';
run;
```

Since the product-limit estimator of the survivor function is not available for the counting process style of input, you cannot use PROC PHREG to obtain the product-limit estimate of the survivor function if you have data with left-truncation times. In the preceding PROC PHREG calls, if you also specify METHOD=PL in a BASELINE statement or an OUTPUT statement, it is defaulted to METHOD=BRESLOW.

The Multiplicative Hazards Model

Consider a set of n subjects such that the counting process $N_i \equiv \{N_i(t), t \geq 0\}$ for the ith subject represents the number of observed events experienced over time t. The sample paths of the process N_i are step functions with jumps of size $+1$, with $N_i(0) = 0$. Let $\boldsymbol{\beta}$ denote the vector of unknown regression coefficients. The multiplicative hazards function $\Lambda(t, \mathbf{Z}_i(t))$ for N_i is given by

$$Y_i(t)d\Lambda(t, \mathbf{Z}_i(t)) = Y_i(t)\exp(\boldsymbol{\beta}'\mathbf{Z}_i(t))d\Lambda_0(t)$$

where

- $Y_i(t)$ indicates whether the ith subject is at risk at time t (specifically, $Y_i(t) = 1$ if at risk and $Y_i(t) = 0$ otherwise)
- $\mathbf{Z}_i(t)$ is the vector of explanatory variables for the ith subject at time t
- $\Lambda_0(t)$ is an unspecified baseline hazard function

See Fleming and Harrington (1991) and Andersen et al. (1992). The Cox model is a special case of this multiplicative hazards model, where $Y_i(t) = 1$ until the first event or censoring, and $Y_i(t) = 0$ thereafter.

The partial likelihood for n independent triplets $(N_i, Y_i, \mathbf{Z}_i), i = 1, \ldots, n$, has the form

$$\mathcal{L}(\boldsymbol{\beta}) = \prod_{i=1}^{n} \prod_{t \geq 0} \left\{ \frac{Y_i(t) \exp(\boldsymbol{\beta}' \mathbf{Z}_i(t))}{\sum_{j=1}^{n} Y_j(t) \exp(\boldsymbol{\beta}' \mathbf{Z}_j(t))} \right\}^{\Delta N_i(t)}$$

where $\Delta N_i(t) = 1$ if $N_i(t) - N_i(t-) = 1$, and $\Delta N_i(t) = 0$ otherwise.

Proportional Rates/Means Models for Recurrent Events

Let $N(t)$ be the number of events experienced by a subject over the time interval $(0, t]$. Let $dN(t)$ be the increment of the counting process N over $[t, t + dt)$. The rate function is given by

$$d\mu_{\mathbf{Z}}(t) = E[dN(t)|Z(t)] = \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}(t)} d\mu_0(t)$$

where $\mu_0(.)$ is an unknown continuous function. If the $\mathbf{Z}$ are time independent, the rate model is reduced to the mean model

$$\mu_Z(t) = \mathrm{e}^{\boldsymbol{\beta}' Z} \mu_0(t)$$

The partial likelihood for n independent triplets $(N_i, Y_i, \mathbf{Z}_i), i = 1, \ldots, n$, of counting, at-risk, and covariate processes is the same as that of the multiplicative hazards model. However, a robust sandwich estimate is used for the covariance matrix of the parameter estimator instead of the model-based estimate.

Let T_{ki} be the kth event time of the ith subject. Let C_i be the censoring time of the ith subject. The at-risk indicator and the failure indicator are, respectively,

$$Y_i(t) = I(C_i \geq t) \text{ and } \Delta_{ki} = I(T_{ki} \leq C_i)$$

Denote

$$S^{(0)}(\boldsymbol{\beta}, t) = \sum_{i=1}^{n} Y_i(t) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(t)} \text{ and } \bar{\mathbf{Z}}(\boldsymbol{\beta}, t) = \frac{\sum_{i=1}^{n} Y_i(t) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(t)} \mathbf{Z}_i(t)}{S^{(0)}(\boldsymbol{\beta}, t)}$$

Let $\hat{\boldsymbol{\beta}}$ be the maximum likelihood estimate of $\boldsymbol{\beta}$, and let $\mathcal{I}(\hat{\boldsymbol{\beta}})$ be the observed information matrix. The robust sandwich covariance matrix estimate is given by

$$\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \sum_{i=1}^{n} \left[W_i(\hat{\boldsymbol{\beta}}) W_i'(\hat{\boldsymbol{\beta}}) \right] \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})$$

where

$$\begin{aligned} W_i(\boldsymbol{\beta}) \quad = \quad & \sum_{k} \Delta_{ki} \left\{ Z_i(T_{ki}) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, T_{ki}) \right\} - \\ & \sum_{j=1}^{n} \sum_{l} \frac{\Delta_{lj} Y_i(T_{lj}) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(T_{lj})}}{S^0(\boldsymbol{\beta}, T_{lj})} \left\{ Z_i(T_{lj}) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, T_{lj}) \right\} \end{aligned}$$

For a given realization of the covariates $\boldsymbol{\xi}$, the Nelson estimator is used to predict the mean function

$$\hat{\mu}_{\boldsymbol{\xi}}(t) = \mathrm{e}^{\hat{\boldsymbol{\beta}}'\boldsymbol{\xi}} \sum_{i=1}^{n} \sum_{k} \frac{I(T_{ki} \le t)\Delta_{ki}}{S^{(0)}(\hat{\boldsymbol{\beta}}, T_{ki})}$$

with standard error estimate given by

$$\hat{\sigma}^2(\hat{\mu}_{\boldsymbol{\xi}}(t)) = \sum_{i=1}^{n} \left(\frac{1}{n} \hat{\Psi}_i(t, \boldsymbol{\xi}) \right)^2$$

where

$$\begin{aligned} \frac{1}{n}\hat{\Psi}_i(t, \boldsymbol{\xi}) \quad = \quad & \mathrm{e}^{\hat{\boldsymbol{\beta}}'\boldsymbol{\xi}} \left\{ \sum_{k} \frac{I(T_{ki} \le t)\Delta_{ki}}{S^{(0)}(\hat{\boldsymbol{\beta}}, T_{ki})} - \sum_{j=1}^{n} \sum_{k} \frac{Y_i(T_{kj}) \mathrm{e}^{\hat{\boldsymbol{\beta}}'\mathbf{Z}_i(T_{kj})} I(T_{kj} \le t)\Delta_{kj}}{[S^{(0)}(\hat{\boldsymbol{\beta}}, T_{kj})]^2} - \right. \\ & \left[\sum_{i=1}^{n} \sum_{k} \frac{I(T_{ki} \le t)\Delta_{ki}[\bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, T_{ki}) - \boldsymbol{\xi}]}{S^{(0)}(\hat{\boldsymbol{\beta}}, T_{ki})} \right] \\ & \left. \times \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \int_0^{\tau} [\mathbf{Z}_i(u) - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, u)] d\hat{M}_i(u) \right\} \end{aligned}$$

Since the cumulative mean function is always nonnegative, the log transform is used to compute confidence intervals. The $100(1-\alpha)\%$ pointwise confidence limits for $\mu_{\boldsymbol{\xi}}(t)$ are

$$\hat{\mu}_{\boldsymbol{\xi}}(t) \mathrm{e}^{\pm z_{\alpha/2} \frac{\hat{\sigma}(\hat{\mu}_{\boldsymbol{\xi}}(t))}{\hat{\mu}_{\boldsymbol{\xi}}(t)}}$$

where $z_{\alpha/2}$ is the upper $100\alpha/2$ percentage point of the standard normal distribution.

The Frailty Model

You can use the frailty model to model correlations between failures of the same cluster. The hazard rate for the jth individual in the ith cluster is

$$\lambda_{ij}(t) = \lambda_0(t) \mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_{ij}(t) + \gamma_i}$$

where $\lambda_0(t)$ is an arbitrary baseline hazard rate, $\mathbf{Z}_{ij}$ is the vector of (fixed-effect) covariates, $\boldsymbol{\beta}$ is the vector of regression coefficients, and γ_i is the random effect for cluster i.

Frailties are the exponential transformations of the random components, and the frailty model can be written as

$$\lambda_{ij}(t) = \lambda_0(t) \mathrm{e}^{\gamma_i} \mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_{ij}(t)}$$

The random components $\gamma_1, \ldots, \gamma_s$ (alternatively, the frailties $\mathrm{e}^{\gamma_1}, \ldots, \mathrm{e}^{\gamma_s}$) are assumed to be independent and identically distributed. Modeling is based on the random effects rather than on the frailties.

Two frailty distributions are available in PROC PHREG: gamma and lognormal. Use the DIST= option in the RANDOM statement to choose the distribution. Let θ be an unknown parameter. The frailty distributions are listed in Table 92.12.

Table 92.12 Frailty Distributions

Frailty Option	Distribution	Density f(γ_i)	Mean and Variance
DIST=GAMMA	$e^{\gamma_i} \sim G\left(\frac{1}{\theta}, \frac{1}{\theta}\right)$	$\frac{\exp(\frac{\gamma_i}{\theta})\exp\left(-\frac{\exp(\gamma_i)}{\theta}\right)}{\theta^{\frac{1}{\theta}}\Gamma(\frac{1}{\theta})}$	E(e^{γ_i})=1, V(e^{γ_i})=θ
DIST=LOGNORMAL	$\gamma_i \sim N(0, \theta)$	$\frac{1}{\sqrt{(2\pi\theta)}}\exp\left(-\frac{\gamma_i^2}{2\theta}\right)$	E(γ_i)=0, V(γ_i)=θ

The unknown parameter θ is a dispersion parameter. Each frailty distribution has a central tendency of 1 (the gamma frailty has a mean of 1, and the lognormal frailty has a median of 1). Thus, you can infer that individuals in cluster i with frailty $e^{\gamma_i} > 1$ (or $e^{\gamma_i} < 1$) tend to fail at a faster (or slower) rate than they fail under an independence model.

PROC PHREG estimates the regression coefficients $\boldsymbol{\beta}$, the random effects $\gamma_1, \ldots, \gamma_s$, and the dispersion parameter θ. The RANDOM statement in PROC PHREG enables you to fit a shared frailty model by a penalized partial likelihood approach. For more information, see the section "The Penalized Partial Likelihood Approach for Fitting Frailty Models" on page 7607. If you also specify the BAYES statement, PROC PHREG performs a Bayesian analysis of the shared frailty model. For more information, see the section "Frailty Model" on page 7616.

If the RANDOM statement is specified, any ASSESS, BASELINE, and OUTPUT statements are ignored. Also ignored are the COVS options in the PROC PHREG statement and the following *options* in the MODEL statement: BEST=, DETAILS, HIERARCHY=, INCLUDE=, NOFIT, PLCONV=, SELECTION=, SEQUENTIAL, SLENTRY=, SLSTAY=, TYPE1, and TYPE3(ALL, LR, SCORE). Profile likelihood confidence intervals for the hazard ratios are not available for the frailty model analysis.

Analysis of Competing-Risks Data

Competing risks arise in the analysis of time-to-event data when the event of interest can be impeded by a prior event of a different type. For example, a leukemia patient's relapse might be unobservable because the patient dies before relapse is diagnosed. In the presence of competing risks, the Kaplan-Meier method of estimating the survivor function is biased, because you can no longer assume that a subject will experience the event of interest if the follow-up period is long enough. An increasingly common practice of assessing the probability of a failure in competing-risks analysis is to estimate the cumulative incidence function (CIF), which is the probability subdistribution function of failure from a specific cause.

PROC PHREG provides two commonly used approaches of evaluating the relationship of the covariates to the cause-specific outcome in competing-risks data. One approach models the cause-specific hazard for each cause of failure separately, by applying the Cox regression to target each cause of failure and censor all other causes. This approach can be considered to be a direct generalization of the Cox model to the competing-risks setting, because the overall hazard function for any failure is the sum of the cause-specific hazard functions from all observable causes. Estimation of the CIF for a specific cause requires a complete analysis of all failure causes and entails fitting multiple cause-specific models, one for each observable cause. This method of analyzing competing-risks data is often called the cause-specific analysis or the Cox model. To request this method, specify the EVENTCODE(COX)= option in the MODEL statement.

Alternatively, Fine and Gray (1999) propose a proportional hazards model for the cumulative incidence of a failure cause of interest. The proportional hazard assumption is imposed on the subdistribution hazard, which is the hazard of the cumulative incidence. This method is often called as the proportional subdistribution hazard regression or the Fine and Gray model. To request this method, specify the EVENTCODE(FG)= option (or simply the EVENTCODE= option) in the MODEL statement.

You can request the CIF curves for a specified set of covariates by using the BASELINE statement. To request the cumulative incidence at specific time points, use the TIMELIST= option in the BASELINE statement. The PLOTS=CIF option in the PROC PHREG statement displays the CIF curves.

Modeling the Cause-Specific Hazard

For each observable cause of failure k, the cause-specific hazard for a subject that has a covariate vector Z follows the proportional hazards assumption,

$$\lambda_k(t|\mathbf{Z}) = \lambda_{k0}(t)\exp(\boldsymbol{\beta}_k'\mathbf{Z})$$

where $\lambda_{k0}(t)$ is an arbitrary and unspecified baseline cause-specific hazard function and $\boldsymbol{\beta}_k$ is the vector of regression coefficients. The regression coefficients for cause k, $\boldsymbol{\beta}_k$, are estimated by fitting a standard Cox model where observations that have the kth cause of failure are treated as event observations and all other observations are treated as censored observations.

You request a Cox regression analysis by specifying the EVENTCODE(COX)= option in the MODEL statement. When you specify this option, the following statements and options are ignored: the ASSESS, BAYES, HAZARDRATIO, RANDOM, STRATA, and WEIGHT statements; the ATRISK and COVS options in the PROC PHREG statement; and the following options in the MODEL statement: BEST=, DETAILS, ENTRY=, HIERARCHY=, INCLUDE=, NOFIT, PLCONV=, RISKLIMITS=PL, SELECTION=, SEQUENTIAL, SLENTRY=, SLSTAY=, TYPE1, and TYPE3(LR, SCORE).

Cumulative Incidence Prediction

For the ith subject, $i = 1, \ldots, n$, let X_i, Δ_i, ϵ_i, and $\mathbf{Z}_i(t)$ be the observed time, event indicator, cause of failure, and covariate vector at time t, respectively. Assume that K causes of failure are observable ($\epsilon_i \in (1, \ldots, K)$). Let

$$\begin{aligned} N_i^l(t) &= I(X_i \le t, \epsilon_i = l) \\ Y_i(t) &= 1 - \sum_{l=1}^{K} N_i^l(t-) \end{aligned}$$

Note that if $\epsilon_i = l$, then $N_i^l(t) = I(X_i \le t)$ and $Y_i(t) = I(X_i \ge t)$; if $\epsilon_i \ne l$, then $N_i^l(t) = 0$ and $Y_i(t) = I(X_i \ge t)$.

For the lth cause-specific Cox model, denote the maximum partial likelihood estimates as $\hat{\boldsymbol{\beta}}_l$ and the baseline cumulative hazard function as $\hat{\Lambda}_{l0}(t)$. For an individual that has covariates $\mathbf{Z} = \mathbf{z}_0$, the predicted cumulative hazard function estimate for cause l is

$$\hat{\Lambda}_l(t; \mathbf{z}_0) = \exp(\hat{\boldsymbol{\beta}}_l'\mathbf{z}_0)\hat{\Lambda}_{l0}(t)$$

The predicted survivor function is

$$\hat{S}(t; \mathbf{z}_0) = \prod_{s: X_i = s \le t, \Delta_i > 0} \left[1 - \hat{\Lambda}_0(s; \mathbf{z}_0)\right]$$

where $\hat{\Lambda}_0(t;\mathbf{z}_0) = \sum_{l=1}^{K} \hat{\Lambda}_l(t;\mathbf{z}_0)$

The predicted cumulative incidence for cause l is

$$\hat{F}_l(t;\mathbf{z}_0) = \int_0^t \hat{S}(s^-;\mathbf{z}_0)d\hat{\Lambda}_l(s;\mathbf{z}_0)$$

Based on the counting process and martingale theory (Andersen et al. 1992), the variance estimator of $\hat{F}_l(t;\mathbf{z}_0)$ is a sum of two terms,

$$\hat{\sigma}_l^2(t;z_0) = \hat{\sigma}_{l1}^2(t;z_0) + \hat{\sigma}_{l2}^2(t;z_0)$$

First, consider TIES=BRESLOW. Let

$$\begin{aligned}
S_l^{(0)}(\boldsymbol{\beta}_l,t) &= \sum_i Y_i(t)\exp(\boldsymbol{\beta}_l'\mathbf{Z}_i) \\
\mathbf{S}_l^{(1)}(\boldsymbol{\beta}_l,t) &= \sum_i Y_i(t)\exp(\boldsymbol{\beta}_l'\mathbf{Z}_i)\mathbf{Z}_i \\
\bar{\mathbf{Z}}_l(\boldsymbol{\beta}_l,t) &= \frac{\mathbf{S}_l^{(1)}(\boldsymbol{\beta}_l,t)}{S_l^{(0)}(\boldsymbol{\beta}_l,t)} \\
\mathbf{W}_l(t) &= \exp(\boldsymbol{\beta}_l'\mathbf{z}_0)\sum_i \int_0^t Y_i(s)(\mathbf{z}_0 - \bar{\mathbf{Z}}_l(\boldsymbol{\beta}_l,s))\frac{dN_i^l(s)}{S_l^{(0)}(\boldsymbol{\beta}_l,s)}
\end{aligned}$$

Replacing $\boldsymbol{\beta}_l$ by the maximum partial likelihood estimate $\hat{\boldsymbol{\beta}}_l$, let

$$\hat{\mathbf{W}}_0(t) = -\sum_{l=1}^{K}\hat{\mathbf{W}}_l(t)$$

and

$$\hat{\mathbf{H}}_l(t) = \sum_{h=0}^{K}\int_0^t \hat{S}(s^-;\mathbf{z}_0)\hat{Q}_{hl}(s,t;\mathbf{z}_0)d\hat{\mathbf{W}}_h(s)$$

where $\hat{Q}_{0l}(s,t;\mathbf{z}_0) = (\hat{F}_l(t;\mathbf{z}_0) - \hat{F}_l(s;\mathbf{z}_0))/\hat{S}(s^-;\mathbf{z}_0)$; and when $h > 0$, $\hat{Q}_{hl}(s,t;\mathbf{z}_0) = 1$ if $h = l$ and 0 otherwise.

The first term of the variance estimator is

$$\hat{\sigma}_{l1}^2(t;z_0) = \hat{\mathbf{H}}_l(t)'\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}_l)\hat{\mathbf{H}}_l(t)$$

where $\mathcal{I}(\hat{\boldsymbol{\beta}}_l)$ is the observed information matrix.

The second term of the variance estimator is

$$\hat{\sigma}_{l2}^2(t;z_0) = \sum_{h=1}^{K}\sum_i\int_0^t \left\{\hat{S}(s^-;\mathbf{z}_0)\left[I(h=l) - \hat{Q}_{0l}(s^-,t;\mathbf{z}_0)\right]\exp(\hat{\boldsymbol{\beta}}_l'\mathbf{z}_0)\right\}^2 \frac{dN_i^h(s)}{[S_h^{(0)}(\hat{\boldsymbol{\beta}}_h,s)]^2}$$

For TIES=EFRON, the preceding computation is modified to comply with the Efron partial likelihood. For a particular time t, let $\Delta_i(t)$=1 if the t is an event time of the ith subject and 0 otherwise. Let $d(t) = \sum_i \Delta_i(t)$, which is the number of subjects that have an event at t. For $1 \le k \le d(t)$, let

$$
\begin{aligned}
S_l^{(0)}(\boldsymbol{\beta}_l, k, t) &= \sum_i Y_i(t)\left\{1 - \frac{k-1}{d(t)}\Delta_i(t)\right\} \exp(\boldsymbol{\beta}_l' \mathbf{Z}_i) \\
\mathbf{S}_l^{(1)}(\boldsymbol{\beta}_l, k, t) &= \sum_i Y_i(t)\left\{1 - \frac{k-1}{d(t)}\Delta_i(t)\right\} \exp(\boldsymbol{\beta}_l' \mathbf{Z}_i)\mathbf{Z}_i \\
\bar{\mathbf{Z}}_l(\boldsymbol{\beta}_l, k, t) &= \frac{S_l^{(1)}(\boldsymbol{\beta}_l, k, t)}{S_l^{(0)}(\boldsymbol{\beta}_l, k, t)} \\
\mathbf{W}_l(t) &= \exp(\boldsymbol{\beta}_l' \mathbf{z}_0) \sum_i \int_0^t \frac{1}{d(s)} \sum_{k=1}^{d(s)} Y_i(s)(\mathbf{z}_0 - \bar{\mathbf{Z}}_l(\boldsymbol{\beta}_l, k, s)) \frac{dN_i^l(s)}{S_l^{(0)}(\boldsymbol{\beta}_l, k, s)}
\end{aligned}
$$

The second term of the variance estimator becomes

$$
\hat{\sigma}_{l2}^2(t; \mathbf{z}_0) = \sum_{h=1}^{K} \sum_i \int_0^t \frac{1}{d(s)} \sum_{k=1}^{d(s)} \left\{\hat{S}(s^-; \mathbf{z}_0)\left[I(h = l) - \hat{Q}_{0l}(s^-, t; \mathbf{z}_0)\right] \exp(\hat{\boldsymbol{\beta}}_l' \mathbf{z}_0)\right\}^2 Y_i(s) \frac{dN_i^h(s)}{\left[S_h^{(0)}(\hat{\boldsymbol{\beta}}_h, k, s)\right]^2}
$$

Confidence intervals for the cumulative incidence $F_l(t; \mathbf{z}_0)$ are computed from $\hat{F}_l(t; \mathbf{z}_0)$ and $\hat{\sigma}_{l1}^2(t; \mathbf{z}_0)$ in the same fashion as those of the survival function. The CLTYPE option in the BASELINE statement enables you to choose the LOG transformation, the LOGLOG (log of negative log) transformation, or no transformation to compute the confidence limits. For more information, see the section "Confidence Intervals for the Survivor Function" on page 7600.

Modeling the Cumulative Incidence

The proportional hazards model for the subdistribution that Fine and Gray (1999) propose aims to model the cumulative incidence of an event of interest. They define a subdistribution hazard,

$$
\bar{\lambda}_k(t) = -\frac{d}{dt} \log(1 - F_k(t))
$$

where $F_k(t)$ is the cumulative incidence function for the failure of cause k. They also impose a proportional hazards assumption on the subdistribution hazards:

$$
\bar{\lambda}_k(t|\mathbf{Z}) = \bar{\lambda}_{k,0}(t) \exp(\boldsymbol{\beta}_k' \mathbf{Z})
$$

The estimation of the regression coefficients is based on modified risk sets, where subjects that experience a competing event are retained after their event. The weight of subjects that are artificially retained in the risk sets is gradually reduced according to the conditional probability of being under follow-up if the competing event had not occurred.

You use PROC PHREG to fit the Fine and Gray (1999) model by specifying the EVENTCODE= option in the MODEL statement to indicate the event of interest. Maximum likelihood estimates of the regression coefficients are obtained by the Newton-Raphson algorithm. The covariance matrix of the parameter estimator

is computed as a sandwich estimate. You can request the CIF curves for a particular set of covariates by using the BASELINE statement. The PLOTS=CIF option in the PROC PHREG statement displays a plot of the curves. You can obtain Schoenfeld residuals and score residuals by using the OUTPUT statement.

To model the subdistribution hazards for clustered data (Zhou et al. 2012), you use the COVS(AGGREGATE) option in the PROC PHREG statement. You need to specify the ID statement to identify the clusters. To model the subdistribution hazards for stratified data (Zhou et al. 2011), you use the STRATA statement. PROC PHREG handles only regular stratified data that have a small number of large subject groups.

When you specify the EVENTCODE= option in the MODEL statement, the following statements and options are ignored: the ASSESS, BAYES, and RANDOM statements; the ATRISK and COVM options in the PROC PHREG statement; and the following options in the MODEL statement: BEST=, DETAILS, HIERARCHY=, INCLUDE=, NOFIT, PLCONV=, RISKLIMITS=PL, SELECTION=, SEQUENTIAL, SLENTRY=, SLSTAY=, TYPE1, and TYPE3(LR, SCORE). Profile likelihood confidence intervals for the hazard ratios are not available for the Fine and Gray competing-risks analysis.

Parameter Estimation

For the ith subject, $i = 1, \ldots, n$, let X_i, Δ_i, ϵ_i, and $\mathbf{Z}_i(t)$ be the observed time, event indicator, cause of failure, and covariate vector at time t, respectively. Assume that K causes of failure are observable ($\epsilon_i \in (1, \ldots, K)$). Consider failure from cause 1 to be the failure of interest, with failures of other causes as competing events. Let

$$\begin{aligned} N_i(t) &= I(X_i \le t, \epsilon_i = 1) \\ Y_i(t) &= 1 - N_i(t-) \end{aligned}$$

Note that if $\epsilon_i = 1$, then $N_i(t) = I(X_i \le t)$ and $Y_i(t) = I(X_i \ge t)$; if $\epsilon_i \ne 1$, then $N_i(t) = 0$ and $Y_i(t) = 1$. Let

$$\begin{aligned} r_i(t) &= I(C_i \ge T_i \wedge t) \\ w_i(t) &= r_i(t)\frac{G(t)}{G(X_i \wedge t)} \end{aligned}$$

where $G(t)$ is the Kaplan-Meier estimate (alternatively, the Breslow estimate) of the survivor function of the censoring variable, which is calculated using $\{X_i, 1 - \Delta_i, i = 1, 2, \ldots, n\}$. If $\Delta_i = 0$, then $r_i(t) = 1$ when $t \le X_i$ and is 0 otherwise; if $\Delta_i = 1$, then $r_i(t) = 1$. Table 92.13 displays the weight of a subject as a function of time.

Table 92.13 Weight for the ith Subject

t, X_i	**Status**	$r_i(t)$	$Y_i(t)$	$w_i(t)$
$t \le X_i$	$\Delta_i = 0$	1	1	1
	$\Delta_i\epsilon_i = 1$	1	1	1
	$\Delta_i\epsilon_i \ne 1$	1	1	1
$t > X_i$	$\Delta_i = 0$	0	1	0
	$\Delta_i\epsilon_i = 1$	1	0	$G(t)/G(X_i)$
	$\Delta_i\epsilon_i \ne 1$	1	1	$G(t)/G(X_i)$

The regression coefficients $\boldsymbol{\beta}$ are estimated by maximizing the pseudo-likelihood $L(\boldsymbol{\beta})$ with respect to $\boldsymbol{\beta}$:

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \left(\frac{\exp(\boldsymbol{\beta}'\mathbf{Z}_i(X_i))}{\sum_{j=1}^{n} Y_j(X_i) w_j(X_i) \exp(\boldsymbol{\beta}'\mathbf{Z}_j(X_i))} \right)^{I(\Delta_i \epsilon_i = 1)}$$

The variance-covariance matrix of the maximum likelihood estimator $\hat{\boldsymbol{\beta}}$ is approximated by a sandwich estimate.

With $\mathbf{a}^{(0)} = 1$, $\mathbf{a}^{(1)} = \mathbf{a}$, and $\mathbf{a}^{(2)} = \mathbf{a}\mathbf{a}'$, let

$$\begin{aligned} \mathbf{S}_2^{(r)}(\boldsymbol{\beta}, u) &= \sum_{j=1}^{n} w_j(u) Y_j(u) \mathbf{Z}_j(u)^{\otimes r} \exp(\boldsymbol{\beta}'\mathbf{Z}_j(u)), \quad r = 0, 1, 2 \\ \bar{\mathbf{Z}}(\boldsymbol{\beta}, u) &= \frac{\mathbf{S}_2^{(1)}(\boldsymbol{\beta}, u)}{S_2^{(0)}(\boldsymbol{\beta}, u)} \end{aligned}$$

The score function $\mathbf{U}_2(\boldsymbol{\beta})$ and the observed information matrix $\hat{\boldsymbol{\Omega}}$ are given by

$$\begin{aligned} \mathbf{U}_2(\hat{\boldsymbol{\beta}}) &= \sum_{i=1}^{n} \Big(\mathbf{Z}_i(X_i) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, X_i) \Big) I(\Delta_i \epsilon_i = 1) \\ \hat{\boldsymbol{\Omega}} &= -\frac{\partial \mathbf{U}_2(\hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}} = \sum_{i=1}^{n} \left(\frac{\mathbf{S}_2^{(2)}(\hat{\boldsymbol{\beta}}, X_i)}{S_2^{(0)}(\hat{\boldsymbol{\beta}}, X_i)} - \bar{\mathbf{Z}}^{\otimes 2}(\hat{\boldsymbol{\beta}}, X_i) \right) I(\Delta_i \epsilon_i = 1) \end{aligned}$$

The sandwich variance estimate of $\hat{\boldsymbol{\beta}}$ is

$$\widehat{\mathrm{var}}(\hat{\boldsymbol{\beta}}) = \hat{\boldsymbol{\Omega}}^{-1} \hat{\boldsymbol{\Sigma}} \hat{\boldsymbol{\Omega}}^{-1}$$

where $\hat{\boldsymbol{\Sigma}}$ is the estimate of the variance-covariance matrix of $\mathbf{U}_2(\hat{\boldsymbol{\beta}})$ that is given by

$$\hat{\boldsymbol{\Sigma}} = \sum_{i=1}^{n} (\hat{\boldsymbol{\eta}}_i + \hat{\boldsymbol{\psi}}_i)^{\otimes 2}$$

where

$$\hat{\boldsymbol{\eta}}_i = \int_0^{\infty} \Big(\mathbf{Z}_i(u) - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, u) \Big) w_i(u) d\hat{M}_i^1(u)$$

$$\hat{\boldsymbol{\psi}}_i = \int_0^{\infty} \frac{\hat{\mathbf{q}}(u)}{\pi(u)} d\hat{M}_i^c(u)$$

$$\hat{\mathbf{q}}(u) = -\sum_{i=1}^{n} \int_0^{\infty} \Big(\mathbf{Z}_i(s) - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, s) \Big) w_i(s) d\hat{M}_i^1 I(s \geq u > X_i)$$

$$\pi(u) = \sum_j I(X_j \geq u)$$

$$\hat{M}_i^1(t) = N_i(t) - \int_0^t Y_i(s) \exp(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i(s)) d\hat{\Lambda}_{10}(s)$$

$$\hat{M}_i^c(t) = I(X_i \le t, \Delta_i = 0) - \int_0^t I(X_i \ge u) d\hat{\Lambda}^c(u)$$

$$\hat{\Lambda}_{10}(t) = \sum_{i=1}^{n} \int_0^t \frac{1}{S_2^{(0)}(\hat{\boldsymbol{\beta}}, u)} w_i(u) dN_i(u)$$

$$\hat{\Lambda}^c(t) = \int_0^t \frac{\sum_i d\{I(X_i \le u, \Delta_i = 0)\}}{\sum_i I(X_i \ge u)}$$

Residuals

You can use the OUTPUT statement to output Schoenfeld residuals and score residuals to a SAS data set.

Schoenfeld residuals:	$\mathbf{Z}_i(X_i) - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, X_i), \Delta_i \epsilon_i = 1$	$1 \le i \le n$
Score residuals:	$\hat{\boldsymbol{\eta}}_i + \hat{\boldsymbol{\psi}}_i$	$1 \le i \le n$

Cumulative Incidence Prediction

For an individual that has covariates $\mathbf{Z} = \mathbf{z}_0$, the cumulative subdistribution hazard is estimated by

$$\hat{\Lambda}_1(t; \mathbf{z}_0) = \int_0^t \exp[\hat{\boldsymbol{\beta}}' \mathbf{z}_0] d\hat{\Lambda}_{10}(u)$$

and the predicted cumulative incidence is

$$\hat{F}_1(t; \mathbf{z}_0) = 1 - \exp[-\hat{\Lambda}_1(t; \mathbf{z}_0)]$$

To compute the confidence interval for the cumulative incidence, consider a monotone transformation $m(p)$ with first derivative $\dot{m}(p)$. Fine and Gray (1999, Section 5) give the following method to calculate pointwise confidence intervals. First, you generate B samples of normal random deviates $\{(A_{k1}, \ldots, A_{kn}), 1 \le k \le B\}$. You can specify the value of B by using the NORMALSAMPLE= option in the BASELINE statement. Then, you compute the estimate of $\mathrm{var}\{m[\hat{F}_1(t; z_0)] - m[F_1(t; z_0)]\}$ as

$$\hat{\sigma}^2(t; z_0) = \frac{1}{B} \sum_{k=1}^{B} \hat{J}_{1k}^2(t; z_0)$$

where

$$\begin{aligned} \hat{J}_{1k}(t; z_0) \quad = \quad & \dot{m}[\hat{F}_1(t; \mathbf{z}_0)] \exp[-\hat{\Lambda}_1(t; \mathbf{z}_0)] \sum_{i=1}^{n} A_{ki} \left\{ \int_0^t \frac{\exp(\hat{\boldsymbol{\beta}}' z_0)}{S_2^{(0)}(\hat{\boldsymbol{\beta}}, u)} w_i(u) d\hat{M}_i^1(u) \right. \\ & \left. + \hat{\boldsymbol{h}}'(t; z_0) \hat{\boldsymbol{\Omega}}^{-1} (\hat{\boldsymbol{\eta}}_i + \hat{\boldsymbol{\psi}}_i) + \int_0^\infty \frac{\hat{\boldsymbol{v}}(u, t, z_0)}{\hat{\pi}(u)} d\hat{M}_i^c(u) \right\} \end{aligned}$$

$$\hat{\mathbf{h}}(t; \mathbf{z}_0) = \exp(\hat{\boldsymbol{\beta}}' \mathrm{z}_0) \left\{ \hat{\Lambda}_{10}(t) \mathbf{z}_0 - \int_0^t \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, u) d\hat{\Lambda}_{10}(u) \right\}$$

$$\hat{v}(u, t, \mathbf{z}_0) = -\exp(\hat{\boldsymbol{\beta}}' \mathrm{z}_0) \sum_{i=1}^{n} \int_0^t \frac{1}{S_2^{(0)}(\hat{\boldsymbol{\beta}}, s)} w_i(s) d\hat{M}_i^1(s) I(s \ge u > X_i)$$

A 100(1–α)% confidence interval for $\hat{F}_1(t;\mathbf{z}_0)$ is given by

$$m^{-1}\left(m[\hat{F}_1(t;\mathbf{z}_0)] \pm z_\alpha \hat{\sigma}(t;\mathbf{z}_0)\right)$$

where z_α is the $100(1-\alpha/2)$ percentile of a standard normal distribution.

The CLTYPE= option in the BASELINE statement enables you to choose the LOG transformation, the LOGLOG (log of negative log) transformation, or the IDENTITY transformation. You can also output the standard error of the cumulative incidence, which is approximated by the delta method as follows:

$$\hat{\sigma}^2(\hat{F}(t;\mathbf{z}_0)) = \left(\dot{m}[\hat{F}(t;\mathbf{z}_0)]\right)^{-2} \hat{\sigma}^2(t;\mathbf{z}_0)$$

where $\dot{m}(x) = \frac{d}{dx}m(x)$. Table 92.14 displays the variance estimator for each transformation that is available in PROC PHREG.

Table 92.14 Variance Estimate of the CIF Predictor

CLTYPE=	Transformation	$\widehat{\mathrm{var}}(\hat{F}(t;\mathbf{z}_0))$
IDENTITY	$m(p) = p$	$\hat{\sigma}^2(t;\mathbf{z}_0)$
LOG	$m(p) = \log(p)$	$\left(\hat{F}_1(t;\mathbf{z}_0)\right)^2 \hat{\sigma}^2(t;\mathbf{z}_0)$
LOGLOG	$m(p) = \log(-\log(p))$	$\left(\hat{F}_1(t;\mathbf{z}_0)\log(\hat{F}_1(t;\mathbf{z}_0))\right)^2 \hat{\sigma}^2(t;\mathbf{z}_0)$

Hazard Ratios

Consider a dichotomous risk factor variable X that takes the value 1 if the risk factor is present and 0 if the risk factor is absent. The log-hazard function is given by

$$\log[\lambda(t|X)] = \log[\lambda_0(t)] + \beta_1 X$$

where $\lambda_0(t)$ is the baseline hazard function.

The hazard ratio ψ is defined as the ratio of the hazard for those with the risk factor ($X = 1$) to the hazard without the risk factor ($X = 0$). The log of the hazard ratio is given by

$$\log(\psi) \equiv \log[\psi(X=1, X=0)] = \log[\lambda(t|X=1)] - \log[\lambda(t|X=0)] = \beta_1$$

In general, the hazard ratio can be computed by exponentiating the difference of the log-hazard between any two population profiles. This is the approach taken by the HAZARDRATIO statement, so the computations are available regardless of parameterization, interactions, and nestings. However, as shown in the preceding equation for $\log(\psi)$, hazard ratios of main effects can be computed as functions of the parameter estimates, and the remainder of this section is concerned with this methodology.

The parameter, β_1, associated with X represents the change in the log-hazard rate from $X = 0$ to $X = 1$. So the hazard ratio is obtained by simply exponentiating the value of the parameter associated with the risk factor. The hazard ratio indicates how the hazard change as you change X from 0 to 1. For instance, $\psi = 2$ means that the hazard when $X = 1$ is twice the hazard when $X = 0$.

Suppose the values of the dichotomous risk factor are coded as constants a and b instead of 0 and 1. The hazard when $X = a$ becomes $\lambda(t)\exp(a\beta_1)$, and the hazard when $X = b$ becomes $\lambda(t)\exp(b\beta_1)$. The hazard ratio corresponding to an increase in X from a to b is

$$\psi = \exp[(b-a)\beta_1] = [\exp(\beta_1)]^{b-a} \equiv [\exp(\beta_1)]^c$$

Note that for any a and b such that $c = b - a = 1$, $\psi = \exp(\beta_1)$. So the hazard ratio can be interpreted as the change in the hazard for any increase of one unit in the corresponding risk factor. However, the change in hazard for some amount other than one unit is often of greater interest. For example, a change of one pound in body weight might be too small to be considered important, while a change of 10 pounds might be more meaningful. The hazard ratio for a change in X from a to b is estimated by raising the hazard ratio estimate for a unit change in X to the power of $c = b - a$ as shown previously.

For a polytomous risk factor, the computation of hazard ratios depends on how the risk factor is parameterized. For illustration, suppose that Cell is a risk factor with four categories: Adeno, Large, Small, and Squamous.

For the effect parameterization scheme (PARAM=EFFECT) with Squamous as the reference group, the design variables for Cell are as follows:

	Design Variables		
Cell	X_1	X_2	X_3
Adeno	1	0	0
Large	0	1	0
Small	0	0	1
Squamous	–1	–1	–1

The log-hazard for Adeno is

$$\begin{aligned}\log[\lambda(t|\text{Adeno})] &= \log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0)\\ &= \lambda_0(t) + \beta_1\end{aligned}$$

The log-hazard for Squamous is

$$\begin{aligned}\log[\lambda(t|\text{Squamous})] &= \log[\lambda_0(t)] + \beta_1(X_1 = -1) + \beta_2(X_2 = -1) + \beta_3(X_3 = -1))\\ &= \log[\lambda_0(t)] - \beta_1 - \beta_2 - \beta_3\end{aligned}$$

Therefore, the log-hazard ratio of Adeno versus Squamous

$$\begin{aligned}\log[\psi(\text{Adeno, Squamous})] &= \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})]\\ &= 2\beta_1 + \beta_2 + \beta_3\end{aligned}$$

For the reference cell parameterization scheme (PARAM=REF) with Squamous as the reference cell, the design variables for Cell are as follows:

	Design Variables		
Cell	X_1	X_2	X_3
Adeno	1	0	0
Large	0	1	0
Small	0	0	1
Squamous	0	0	0

The log-hazard ratio of Adeno versus Squamous is given by

$$\begin{aligned}
&\log(\psi(\text{Adeno}, \text{Squamous})) \\
&\quad = \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})] \\
&\quad = (\log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0)) - \\
&\qquad (\log[\lambda_0(t)] + \beta_1(X_1 = 0) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0)) \\
&\quad = \beta_1
\end{aligned}$$

For the GLM parameterization scheme (PARAM=GLM), the design variables are as follows:

	Design Variables			
Cell	X_1	X_2	X_3	X_4
Adeno	1	0	0	0
Large	0	1	0	0
Small	0	0	1	0
Squamous	0	0	0	1

The log-hazard ratio of Adeno versus Squamous is

$$\begin{aligned}
&\log(\psi(\text{Adeno}, \text{Squamous})) \\
&\quad = \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})] \\
&\quad = \log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) + \beta_4(X_4 = 0)) - \\
&\qquad (\log(\lambda_0(t)) + \beta_1(X_1 = 0) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) + \beta_4(X_4 = 1)) \\
&\quad = \beta_1 - \beta_4
\end{aligned}$$

Consider Cell as the only risk factor in the Cox regression in Example 92.3. The computation of hazard ratio of Adeno versus Squamous for various parameterization schemes is tabulated in Table 92.15.

Table 92.15 Hazard Ratio Comparing Adeno to Squamous

	Parameter Estimates				
PARAM=	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	**Hazard Ratio Estimates**
EFFECT	0.5494	–0.3618	0.4044		$\exp(2 \times 0.5494 - 0.3618 + 0.4044) = 3.131$
REF	1.1414	0.2302	0.9964		$\exp(1.1414) = 3.131$
GLM	1.1414	0.2302	0.9964	0.0000	$\exp(1.1414) = 3.131$

The fact that the log-hazard ratio ($\log(\psi)$) is a linear function of the parameters enables the HAZARDRATIO statement to compute the hazard ratio of the main effect even in the presence of interactions and nest effects. The section "Hazard Ratios" on page 7563 details the estimation of the hazard ratios in a classical analysis.

To customize hazard ratios for specific units of change for a continuous risk factor, you can use the UNITS= option in a HAZARDRATIO statement to specify a list of relevant units for each explanatory variable in the model. Estimates of these customized hazard ratios are given in a separate table. Let (L_j, U_j) be a confidence interval for $\log(\psi)$. The corresponding lower and upper confidence limits for the customized hazard ratio $\exp(c\beta_j)$ are $\exp(cL_j)$ and $\exp(cU_j)$, respectively (for $c > 0$), or $\exp(cU_j)$ and $\exp(cL_j)$, respectively (for $c < 0$).

Newton-Raphson Method

Let $L(\boldsymbol{\beta})$ be one of the likelihood functions described in the previous subsections. Let $l(\boldsymbol{\beta}) = \log L(\boldsymbol{\beta})$. Finding $\boldsymbol{\beta}$ such that $L(\boldsymbol{\beta})$ is maximized is equivalent to finding the solution $\hat{\boldsymbol{\beta}}$ to the likelihood equations

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = 0$$

With $\hat{\boldsymbol{\beta}}^0 = \mathbf{0}$ as the initial solution, the iterative scheme is expressed as

$$\hat{\boldsymbol{\beta}}^{j+1} = \hat{\boldsymbol{\beta}}^j - \left[\frac{\partial^2 l(\hat{\boldsymbol{\beta}}^j)}{\partial \boldsymbol{\beta}^2}\right]^{-1} \frac{\partial l(\hat{\boldsymbol{\beta}}^j)}{\partial \boldsymbol{\beta}}$$

The term after the minus sign is the Newton-Raphson step. If the likelihood function evaluated at $\hat{\boldsymbol{\beta}}^{j+1}$ is less than that evaluated at $\hat{\boldsymbol{\beta}}^j$, then $\hat{\boldsymbol{\beta}}^{j+1}$ is recomputed using half the step size. The iterative scheme continues until convergence is obtained—that is, until $\hat{\boldsymbol{\beta}}_{j+1}$ is sufficiently close to $\hat{\boldsymbol{\beta}}_j$. Then the maximum likelihood estimate of $\boldsymbol{\beta}$ is $\hat{\boldsymbol{\beta}} = \hat{\boldsymbol{\beta}}_{j+1}$.

The model-based variance estimate of $\hat{\boldsymbol{\beta}}$ is obtained by inverting the information matrix $\mathcal{I}(\hat{\boldsymbol{\beta}})$

$$\hat{\mathbf{V}}_m(\hat{\boldsymbol{\beta}}) = \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) = -\left[\frac{\partial^2 l(\hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}^2}\right]^{-1}$$

Firth's Modification for Maximum Likelihood Estimation

In fitting a Cox model, the phenomenon of monotone likelihood is observed if the likelihood converges to a finite value while at least one parameter diverges (Heinze and Schemper 2001).

Let $\mathbf{x}_l(t)$ denote the vector explanatory variables for the lth individual at time t. Let $t_1 < t_2 < \ldots < t_m$ denote the k distinct, ordered event times. Let d_j denote the multiplicity of failures at t_j; that is, d_j is the size of the set $\mathcal{D}_j$ of individuals that fail at t_j. Let $\mathcal{R}_j$ denote the risk set just before t_j. Let $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)'$ be the vector of regression parameters. The Breslow log partial likelihood is given by

$$l(\boldsymbol{\beta}) = \log L(\boldsymbol{\beta}) = \sum_{j=1}^{m} \left\{ \boldsymbol{\beta}' \sum_{l \in \mathcal{D}_j} \mathbf{x}_l(t_j) - d_j \log \sum_{h \in \mathcal{R}j} \mathrm{e}^{\boldsymbol{\beta}' \mathbf{x}_h(t_j)} \right\}$$

Denote

$$\mathbf{S}_j^{(a)}(\boldsymbol{\beta}) = \sum_{h \in \mathcal{R}j} \mathrm{e}^{\boldsymbol{\beta}' \mathbf{x}_h(t_j)} [\mathbf{x}_h(t_j)]^{\otimes a} \qquad a = 0, 1, 2$$

Then the score function is given by

$$\begin{aligned} \mathbf{U}(\boldsymbol{\beta}) &\equiv (U(\beta_1), \ldots, U(\beta_k))' \\ &= \frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \\ &= \sum_{j=1}^{m} \left\{ \sum_{l \in \mathcal{D}j} \mathbf{x}_l(t_j) - d_j \frac{\mathbf{S}_j^{(1)}(\boldsymbol{\beta})}{S_j^0(\boldsymbol{\beta})} \right\} \end{aligned}$$

and the Fisher information matrix is given by

$$\begin{aligned} \mathcal{I}(\boldsymbol{\beta}) &= -\frac{\partial^2 l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}^2} \\ &= \sum_{j=1}^{m} d_j \left\{ \frac{\mathbf{S}_j^{(2)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} - \left[\frac{\mathbf{S}_j^{(1)}(\boldsymbol{\beta})}{\mathbf{S}_j^{(0)}(\boldsymbol{\beta})} \right] \left[\frac{\mathbf{S}_j^{(1)}(\boldsymbol{\beta})}{\mathbf{S}_j^{(0)}(\boldsymbol{\beta})} \right]' \right\} \end{aligned}$$

Heinze (1999); Heinze and Schemper (2001) applied the idea of Firth (1993) by maximizing the penalized partial likelihood

$$l^*(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) + 0.5 \log(|\mathcal{I}(\boldsymbol{\beta})|)$$

The score function $\mathbf{U}(\boldsymbol{\beta})$ is replaced by the modified score function by $\mathbf{U}^*(\boldsymbol{\beta}) \equiv (U^*(\beta_1), \ldots, U^*(\beta_k))'$, where

$$U^*(\beta_r) = U(\beta_r) + 0.5 \mathrm{tr} \left\{ \mathcal{I}^{-1}(\boldsymbol{\beta}) \frac{\partial \mathcal{I}(\boldsymbol{\beta})}{\partial \beta_r} \right\} \qquad r = 1, \ldots, k$$

The Firth estimate is obtained iteratively as

$$\boldsymbol{\beta}^{(s+1)} = \boldsymbol{\beta}^{(s)} + \mathcal{I}^{-1}(\boldsymbol{\beta}^{(s)}) \mathbf{U}^*(\boldsymbol{\beta}^{(s)})$$

The covariance matrix $\hat{\mathbf{V}}$ is computed as $\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})$, where $\hat{\boldsymbol{\beta}}$ is the maximum penalized partial likelihood estimate.

Explicit formulae for $\frac{\partial \mathcal{I}(\boldsymbol{\beta})}{\partial \beta_r}, r = 1, \ldots, k$

Denote

$$\begin{aligned} \mathbf{x}_h(t) &= (x_{h1}(t), \ldots, x_{hk}(t))' \\ \mathbf{Q}_{jr}^{(a)}(\boldsymbol{\beta}) &= \sum_{h \in \mathcal{R}j} \mathrm{e}^{\boldsymbol{\beta}'\mathbf{x}_h(t_j)} x_{hr}(t_j)[\mathbf{x}_h(t_j)]^{\otimes a} \qquad a = 0, 1, 2; r = 1, \ldots, k \end{aligned}$$

Then

$$\begin{aligned} \frac{\partial \mathcal{I}(\boldsymbol{\beta})}{\partial \beta_r} = & \sum_{j=1}^{m} d_j \left\{ \left[\frac{\mathbf{Q}_{jr}^{(2)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} - \frac{\mathbf{Q}_{jr}^{(0)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} \frac{\mathbf{S}_j^{(2)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} \right] - \right. \\ & \left[\frac{\mathbf{Q}_{jr}^{(1)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} - \frac{\mathbf{Q}_{jr}^{(0)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} \frac{\mathbf{S}_j^{(1)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} \right] \left[\frac{\mathbf{S}_j^{(1)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} \right]' - \\ & \left. \left[\frac{\mathbf{S}_j^{(1)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} \right] \left[\frac{\mathbf{Q}_{jr}^{(1)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} - \frac{\mathbf{Q}_{jr}^{(0)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} \frac{\mathbf{S}_j^{(1)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} \right]' \right\} \qquad r = 1, \ldots, k \end{aligned}$$

Robust Sandwich Variance Estimate

For the ith subject, $i = 1, \ldots, n$, let X_i, w_i, and $\mathbf{Z}_i(t)$ be the observed time, weight, and the covariate vector at time t, respectively. Let Δ_i be the event indicator and let $Y_i(t) = I(X_i \geq t)$. Let

$$S^{(r)}(\boldsymbol{\beta}, t) = \sum_{j=1}^{n} w_j Y_j(t) \mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_j(t)} \mathbf{Z}_j^{\otimes r}(t), r = 0, 1$$

Let $\bar{\mathbf{Z}}(\boldsymbol{\beta}, t) = \frac{S^{(1)}(\boldsymbol{\beta}, t)}{S^{(0)}(\boldsymbol{\beta}, t)}$. The score residual for the ith subject is

$$\mathbf{L}_i(\boldsymbol{\beta}) = \Delta_i \left\{ \mathbf{Z}_i(X_i) - \bar{\mathbf{Z}}(\beta, X_i) \right\} - \sum_{j=1}^{n} \Delta_j \frac{w_j Y_i(X_j) \mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_i(X_j)}}{S^{(0)}(\boldsymbol{\beta}, X_j)} \left\{ \mathbf{Z}_i(X_j) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, X_j) \right\}$$

For TIES=EFRON, the computation of the score residuals is modified to comply with the Efron partial likelihood. For more information, see the section "Residuals" on page 7585.

The robust sandwich variance estimate of $\hat{\boldsymbol{\beta}}$ derived by Binder (1992), who incorporated weights into the analysis, is

$$\hat{\mathbf{V}}_s(\hat{\boldsymbol{\beta}}) = \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \left[\sum_{j=1}^{n} (w_j \mathbf{L}_j(\hat{\boldsymbol{\beta}}))^{\otimes 2} \right] \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})$$

where $\mathcal{I}(\hat{\boldsymbol{\beta}})$ is the observed information matrix, and $\mathbf{a}^{\otimes 2} = \mathbf{a}\mathbf{a}'$. Note that when $w_i \equiv 1$,

$$\hat{\mathbf{V}}_s(\hat{\boldsymbol{\beta}}) = \mathbf{D}'\mathbf{D}$$

where $\mathbf{D}$ is the matrix of DFBETA residuals. This robust variance estimate was proposed by Lin and Wei (1989) and Reid and Crépeau (1985).

Testing the Global Null Hypothesis

The following statistics can be used to test the global null hypothesis $H_0\colon \boldsymbol{\beta} = \mathbf{0}$. Under mild assumptions, each statistic has an asymptotic chi-square distribution with p degrees of freedom given the null hypothesis. The value p is the dimension of $\boldsymbol{\beta}$. For clustered data, the likelihood ratio test, the score test, and the Wald test assume independence of observations within a cluster, while the robust Wald test and the robust score test do not need such an assumption.

Likelihood Ratio Test

$$\chi^2_{\mathrm{LR}} = 2\left[l(\hat{\boldsymbol{\beta}}) - l(\mathbf{0})\right]$$

Score Test

$$\chi^2_S = \left[\frac{\partial l(\mathbf{0})}{\partial \boldsymbol{\beta}}\right]' \left[-\frac{\partial^2 l(\mathbf{0})}{\partial \boldsymbol{\beta}^2}\right]^{-1} \left[\frac{\partial l(\mathbf{0})}{\partial \boldsymbol{\beta}}\right]$$

Wald's Test

$$\chi^2_W = \hat{\boldsymbol{\beta}}' \left[-\frac{\partial^2 l(\hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}^2}\right] \hat{\boldsymbol{\beta}}$$

Robust Score Test

$$\chi^2_{\mathrm{RS}} = \left[\sum_i \mathbf{L}_i^0\right]' \left[\sum_i \mathbf{L}_i^0 \mathbf{L}_i^{0\prime}\right]^{-1} \left[\sum_i \mathbf{L}_i^0\right]$$

where $\mathbf{L}_i^0$ is the score residual of the ith subject at $\boldsymbol{\beta} = \mathbf{0}$; that is, $\mathbf{L}_i^0 = \mathbf{L}_i(\mathbf{0}, \infty)$, where the score process $\mathbf{L}_i(\boldsymbol{\beta}, t)$ is defined in the section "Residuals" on page 7585.

Robust Wald's Test

$$\chi^2_{\mathrm{RW}} = \hat{\boldsymbol{\beta}}'[\hat{\mathbf{V}}_s(\hat{\boldsymbol{\beta}})]^{-1}\hat{\boldsymbol{\beta}}$$

where $\hat{\mathbf{V}}_s(\hat{\boldsymbol{\beta}})$ is the sandwich variance estimate. For more information, see the section "Robust Sandwich Variance Estimate" on page 7568.

Type 3 Tests and Joint Tests

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, performing the Type 3 test of a main effect corresponds to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions." Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are zero. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), performing the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses reference parameterization (as specified by the PARAM=REF option in the CLASS statement), performing the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects. For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).

If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimable function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.

The following statistics can be used to test the null hypothesis $H_{0L}\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$, where $\mathbf{L}$ is a matrix of known coefficients. Under large sample likelihood theory (Kalbfleisch and Prentice 1980, Section 3.4), each of the following statistics has each of the following statistics has an asymptotic chi-square distribution with p degrees of freedom, where p is the rank of $\mathbf{L}$. Let $\tilde{\boldsymbol{\beta}}_{\mathbf{L}}$ be the maximum likelihood of $\boldsymbol{\beta}$ under the null hypothesis $H_{0\mathbf{L}}$; that is,

$$l(\tilde{\boldsymbol{\beta}}_{\mathbf{L}}) = \max_{\mathbf{L}\boldsymbol{\beta}=\mathbf{0}} l(\boldsymbol{\beta})$$

Likelihood Ratio Statistic

$$\chi^2_{\mathrm{LR}} = 2\left[l(\hat{\boldsymbol{\beta}}) - l(\tilde{\boldsymbol{\beta}}_{\mathbf{L}})\right]$$

Score Statistic

$$\chi^2_S = \left[\frac{\partial l(\tilde{\boldsymbol{\beta}}_{\mathbf{L}})}{\partial \boldsymbol{\beta}}\right]' \left[-\frac{\partial^2 l(\tilde{\boldsymbol{\beta}}_{\mathbf{L}})}{\partial \boldsymbol{\beta}^2}\right]^{-1} \left[\frac{\partial l(\tilde{\boldsymbol{\beta}}_{\mathbf{L}})}{\partial \boldsymbol{\beta}}\right]$$

Wald's Statistic

$$\chi^2_W = \left(\mathbf{L}\hat{\boldsymbol{\beta}}\right)' \left[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{L}'\right]^{-1} \left(\mathbf{L}\hat{\boldsymbol{\beta}}\right)$$

where $\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})$ is the estimated covariance matrix, which can be the model-based covariance matrix $\left[-\frac{\partial^2 l(\hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}^2}\right]^{-1}$ or the sandwich covariance matrix $V_S(\hat{\boldsymbol{\beta}})$. For more information, see the section "Robust Sandwich Variance Estimate" on page 7568.

Confidence Limits for a Hazard Ratio

Let $\mathbf{e}_j$ be the jth unit vector—that is, the jth entry of the vector is 1 and all other entries are 0. The hazard ratio for the explanatory variable with regression coefficient $\beta_j = \mathbf{e}'_j\boldsymbol{\beta}$ is defined as $\exp(\beta_j)$. In general, a log-hazard ratio can be written as $\mathbf{h}'\boldsymbol{\beta}$, a linear combination of the regression coefficients, and the hazard ratio $\exp(\mathbf{h}'\boldsymbol{\beta})$ is obtained by replacing $\mathbf{e}_j$ with $\mathbf{h}$.

Point Estimate

The hazard ratio $\exp(\mathbf{e}'_j\boldsymbol{\beta})$ is estimated by $\exp(\mathbf{e}'_j\hat{\beta})$, where $\hat{\boldsymbol{\beta}}$ is the maximum likelihood estimate of the $\boldsymbol{\beta}$.

Wald's Confidence Limits

The $100(1-\alpha)\%$ confidence limits for the hazard ratio are calculated as

$$\exp\left(\mathbf{e}'_j\hat{\beta} \pm z_{\alpha/2}\sqrt{\mathbf{e}'_j\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{e}_j}\right)$$

where $\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})$ is estimated covariance matrix, and $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile point of the standard normal distribution.

Profile-Likelihood Confidence Limits

The construction of the profile-likelihood-based confidence interval is derived from the asymptotic χ^2 distribution of the generalized likelihood ratio test of Venzon and Moolgavkar (1988). Suppose that the parameter vector is $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)'$ and you want to compute a confidence interval for β_j. The profile-likelihood function for $\beta_j = \gamma$ is defined as

$$l^*_j(\gamma) = \max_{\boldsymbol{\beta}\in\mathcal{B}_j(\gamma)} l(\boldsymbol{\beta})$$

where $\mathcal{B}_j(\gamma)$ is the set of all $\boldsymbol{\beta}$ with the jth element fixed at γ, and $l(\boldsymbol{\beta})$ is the log-likelihood function for $\boldsymbol{\beta}$. If $l_{\max} = l(\hat{\boldsymbol{\beta}})$ is the log likelihood evaluated at the maximum likelihood estimate $\hat{\boldsymbol{\beta}}$, then $2(l_{\max} - l^*_j(\beta_j))$

has a limiting chi-square distribution with one degree of freedom if β_j is the true parameter value. Let $l_0 = l_{\max} - 0.5\chi_1^2(1-\alpha)$, where $\chi_1^2(1-\alpha)$ is the $100(1-\alpha)$th percentile of the chi-square distribution with one degree of freedom. A $100(1-\alpha)\%$ confidence interval for β_j is

$$\{\gamma : l_j^*(\gamma) \geq l_0\}$$

The endpoints of the confidence interval are found by solving numerically for values of β_j that satisfy equality in the preceding relation. To obtain an iterative algorithm for computing the confidence limits, the log-likelihood function in a neighborhood of $\boldsymbol{\beta}$ is approximated by the quadratic function

$$\tilde{l}(\boldsymbol{\beta} + \boldsymbol{\delta}) = l(\boldsymbol{\beta}) + \boldsymbol{\delta}'\mathbf{g} + \frac{1}{2}\boldsymbol{\delta}'\mathbf{V}\boldsymbol{\delta}$$

where $\mathbf{g} = \mathbf{g}(\boldsymbol{\beta})$ is the gradient vector and $\mathbf{V} = \mathbf{V}(\boldsymbol{\beta})$ is the Hessian matrix. The increment $\boldsymbol{\delta}$ for the next iteration is obtained by solving the likelihood equations

$$\frac{d}{d\boldsymbol{\delta}}\{\tilde{l}(\boldsymbol{\beta} + \boldsymbol{\delta}) + \lambda(\mathbf{e}_j'\boldsymbol{\delta} - \gamma)\} = \mathbf{0}$$

where λ is the Lagrange multiplier, $\mathbf{e}_j$ is the jth unit vector, and γ is an unknown constant. The solution is

$$\boldsymbol{\delta} = -\mathbf{V}^{-1}(\mathbf{g} + \lambda\mathbf{e}_j)$$

By substituting this $\boldsymbol{\delta}$ into the equation $\tilde{l}(\boldsymbol{\beta} + \boldsymbol{\delta}) = l_0$, you can estimate λ as

$$\lambda = \pm\left(\frac{2(l_0 - l(\boldsymbol{\beta}) + \frac{1}{2}\mathbf{g}'\mathbf{V}^{-1}\mathbf{g})}{\mathbf{e}_j'\mathbf{V}^{-1}\mathbf{e}_j}\right)^{\frac{1}{2}}$$

The upper confidence limit for β_j is computed by starting at the maximum likelihood estimate of $\boldsymbol{\beta}$ and iterating with positive values of λ until convergence is attained. The process is repeated for the lower confidence limit, using negative values of λ.

Convergence is controlled by value ϵ specified with the PLCONV= option in the MODEL statement (the default value of ϵ is 1E–4). Convergence is declared on the current iteration if the following two conditions are satisfied:

$$|l(\boldsymbol{\beta}) - l_0| \leq \epsilon$$

and

$$(\mathbf{g} + \lambda\mathbf{e}_j)'\mathbf{V}^{-1}(\mathbf{g} + \lambda\mathbf{e}_j) \leq \epsilon$$

The profile-likelihood confidence limits for the hazard ratio $\exp(\mathbf{e}_j'\boldsymbol{\beta})$ are obtained by exponentiating these confidence limits.

Using the TEST Statement to Test Linear Hypotheses

Linear hypotheses for $\boldsymbol{\beta}$ are expressed in matrix form as

$$H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{c}$$

where $\mathbf{L}$ is a matrix of coefficients for the linear hypotheses, and $\mathbf{c}$ is a vector of constants. The Wald chi-square statistic for testing H_0 is computed as

$$\chi^2_W = \left(\mathbf{L}\hat{\boldsymbol{\beta}} - \mathbf{c}\right)' \left[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{L}'\right]^{-1} \left(\mathbf{L}\hat{\boldsymbol{\beta}} - \mathbf{c}\right)$$

where $\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})$ is the estimated covariance matrix. Under H_0, χ^2_W has an asymptotic chi-square distribution with r degrees of freedom, where r is the rank of $\mathbf{L}$.

Optimal Weights for the AVERAGE Option in the TEST Statement

Let $\boldsymbol{\beta}_0 = (\beta_{i_1}, \ldots, \beta_{i_s})'$, where $\{\beta_{i_1}, \ldots, \beta_{i_s}\}$ is a subset of s regression coefficients. For any vector $\mathbf{e} = (e_1, \ldots, e_s)'$ of length s,

$$\mathbf{e}'\hat{\boldsymbol{\beta}}_0 \sim N(\mathbf{e}'\boldsymbol{\beta}_0, \mathbf{e}'\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}}_0)\mathbf{e})$$

To find $\mathbf{e}$ such that $\mathbf{e}'\hat{\boldsymbol{\beta}}_0$ has the minimum variance, it is necessary to minimize $\mathbf{e}'\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}}_0)\mathbf{e}$ subject to $\sum_{i=1}^{k} e_i = 1$. Let $\mathbf{1}_s$ be a vector of 1's of length s. The expression to be minimized is

$$\mathbf{e}'\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}}_0)\mathbf{e} - \lambda(\mathbf{e}'\mathbf{1}_s - 1)$$

where λ is the Lagrange multiplier. Differentiating with respect to $\mathbf{e}$ and λ, respectively, yields

$$\begin{aligned} \hat{\mathbf{V}}(\hat{\boldsymbol{\beta}}_0)\mathbf{e} - \lambda\mathbf{1}_s &= \mathbf{0} \\ \mathbf{e}'\mathbf{1}_s - 1 &= 0 \end{aligned}$$

Solving these equations gives

$$\mathbf{e} = [\mathbf{1}_s'\hat{\mathbf{V}}^{-1}(\hat{\boldsymbol{\beta}}_0)\mathbf{1}_s]^{-1}\hat{\mathbf{V}}^{-1}(\hat{\boldsymbol{\beta}}_0)\mathbf{1}_s$$

This provides a one degree-of-freedom test for testing the null hypothesis $H_0 : \beta_{i_1} = \ldots = \beta_{i_s} = 0$ with normal test statistic

$$Z = \frac{\mathbf{e}'\hat{\boldsymbol{\beta}}_0}{\sqrt{\mathbf{e}'\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}}_0)\mathbf{e}}}$$

This test is more sensitive than the multivariate test specified by the TEST statement

```
Multivariate: test X1, ..., Xs;
```

where X1, ..., Xs are the variables with regression coefficients $\beta_{i_1}, \ldots, \beta_{i_s}$, respectively.

Analysis of Multivariate Failure Time Data

Multivariate failure time data arise when each study subject can potentially experience several events (for instance, multiple infections after surgery) or when there exists some natural or artificial clustering of subjects (for instance, a litter of mice) that induces dependence among the failure times of the same cluster. Data in the former situation are referred to as multiple events data, and data in the latter situation are referred to as clustered data. The multiple events data can be further classified into ordered and unordered data. For ordered data, there is a natural ordering of the multiple failures within a subject, which includes recurrent events data as a special case. For unordered data, the multiple event times result from several concurrent failure processes.

Multiple events data can be analyzed by the Wei, Lin, and Weissfeld (1989), or WLW, method based on the marginal Cox models. For the special case of recurrent events data, you can fit the intensity model (Andersen and Gill 1982), the proportional rates/means model (Pepe and Cai 1993; Lawless and Nadeau 1995; Lin et al. 2000), or the stratified models for total time and gap time proposed by Prentice, Williams, and Peterson (1981), or PWP. For clustered data, you can carry out the analysis of Lee, Wei, and Amato (1992) based on the marginal Cox model. To use PROC PHREG to perform these analyses correctly and effectively, you have to array your data in a specific way to produce the correct risk sets.

All examples described in this section can be found in the program *phrmult.sas* in the SAS/STAT sample library. Furthermore, the "Examples" section in this chapter contains two examples to illustrate the methods of analyzing recurrent events data and clustered data.

Marginal Cox Models for Multiple Events Data

Suppose there are n subjects and each subject can experience up to K potential events. Let $\mathbf{Z}_{ki}(.)$ be the covariate process associated with the kth event for the ith subject. The marginal Cox models are given by

$$\lambda_k(t;\mathbf{Z}_{ki}) = \lambda_{k0}\mathrm{e}^{\boldsymbol{\beta}_k'\mathbf{Z}_{ki}(t)}, k = 1,\ldots,K; i = 1,\ldots,n$$

where $\lambda_{k0}(t)$ is the (event-specific) baseline hazard function for the kth event and $\boldsymbol{\beta}_k$ is the (event-specific) column vector of regression coefficients for the kth event. WLW estimates $\boldsymbol{\beta}_1,\ldots,\boldsymbol{\beta}_K$ by the maximum partial likelihood estimates $\hat{\boldsymbol{\beta}}_1,\ldots,\hat{\boldsymbol{\beta}}_K$, respectively, and uses a robust sandwich covariance matrix estimate for $(\hat{\boldsymbol{\beta}}_1',\ldots,\hat{\boldsymbol{\beta}}_K')'$ to account for the dependence of the multiple failure times.

By using a properly prepared input data set, you can estimate the regression parameters for all the marginal Cox models and compute the robust sandwich covariance estimates in one PROC PHREG invocation. For convenience of discussion, suppose each subject can potentially experience $K = 3$ events and there are two explanatory variables Z1 and Z2. The event-specific parameters to be estimated are $\boldsymbol{\beta}_1 = (\beta_{11},\beta_{21})'$ for the first marginal model, $\boldsymbol{\beta}_2 = (\beta_{12},\beta_{22})'$ for the second marginal model, and $\boldsymbol{\beta}_3 = (\beta_{13},\beta_{23})'$ for the third marginal model. Inference of these parameters is based on the robust sandwich covariance matrix estimate of the parameter estimators. It is necessary that each row of the input data set represent the data for a potential event of a subject. The input data set should contain the following:

- an ID variable for identifying the subject so that all observations of the same subject have the same ID value
- an Enum variable to index the multiple events. For example, Enum=1 for the first event, Enum=2 for the second event, and so on.

- a Time variable to represent the observed time from some time origin for the event. For recurrence events data, it is the time from the study entry to each recurrence.
- a Status variable to indicate whether the Time value is a censored or uncensored time. For example, Status=1 indicates an uncensored time and Status=0 indicates a censored time.
- independent variables (Z1 and Z2)

The WLW analysis can be carried out by specifying the following:

```
proc phreg covs(aggregate);
   model Time*Status(0)=Z11 Z12 Z13 Z21 Z22 Z23;
   strata Enum;
   id ID;
   Z11= Z1 * (Enum=1);
   Z12= Z1 * (Enum=2);
   Z13= Z1 * (Enum=3);
   Z21= Z2 * (Enum=1);
   Z22= Z2 * (Enum=2);
   Z23= Z2 * (Enum=3);
run;
```

The variable Enum is specified in the STRATA statement so that there is one marginal Cox model for each distinct value of Enum. The variables Z11, Z12, Z13, Z21, Z22, and Z23 in the MODEL statement are event-specific variables derived from the independent variables Z1 and Z2 by the given programming statements. In particular, the variables Z11, Z12, and Z13 are event-specific variables for the explanatory variable Z1; the variables Z21, Z22, and Z23 are event-specific variables for the explanatory variable Z2. For $j = 1, 2$, and $k = 1, 2, 3$, variable Zjk contains the same values as the explanatory variable Zj for the rows that correspond to kth marginal model and the value 0 for all other rows; as such, β_{jk} is the regression coefficient for Zjk. You can avoid using the programming statements in PROC PHREG if you create these event-specific variables in the input data set by using the same programming statements in a DATA step.

The option COVS(AGGREGATE) is specified in the PROC PHREG statement to obtain the robust sandwich estimate of the covariance matrix, and the score residuals used in computing the middle part of the sandwich estimate are aggregated over identical ID values. You can also include TEST statements in the PROC PHREG code to test various linear hypotheses of the regression parameters based on the robust sandwich covariance matrix estimate.

Consider the AIDS study data in Wei, Lin, and Weissfeld (1989) from a randomized clinical trial to assess the antiretroviral capacity of ribavirin over time in AIDS patients. Blood samples were collected at weeks 4, 8, and 12 from each patient in three treatment groups (placebo, low dose of ribavirin, and high dose). For each serum sample, the failure time is the number of days before virus positivity was detected. If the sample was contaminated or it took a longer period of time than was achievable in the laboratory, the sample was censored. For example:

- Patient #1 in the placebo group has uncensored times 9, 6, and 7 days (that is, it took 9 days to detect viral positivity in the first blood sample, 6 days for the second blood sample, and 7 days for the third blood sample).
- Patient #14 in the low-dose group of ribavirin has uncensored times of 16 and 17 days for the first and second sample, respectively, and a censored time of 21 days for the third blood sample.

- Patient #28 in the high-dose group has an uncensored time of 21 days for the first sample, no measurement for the second blood sample, and a censored time of 25 days for the third sample.

For a full-rank parameterization, two design variables are sufficient to represent three treatment groups. Based on the reference coding with placebo as the reference, the values of the two dummy explanatory variables Z1 and Z2 representing the treatments are as follows:

Treatment Group	Z1	Z2
Placebo	0	0
Low dose ribavirin	1	0
High dose ribavirin	0	1

The bulk of the task in using PROC PHREG to perform the WLW analysis lies in the preparation of the input data set. As discussed earlier, the input data set should contain the ID, Enum, Time, and Status variables, and event-specific independent variables Z11, Z12, Z13, Z21, Z22, and Z23. Data for the three patients described earlier are arrayed as follows:

ID	Time	Status	Enum	Z1	Z2
1	9	1	1	0	0
1	6	1	2	0	0
1	7	1	3	0	0
14	16	1	1	1	0
14	17	1	2	1	0
14	21	0	3	1	0
28	21	1	1	0	1
28	25	0	3	0	1

The first three rows are data for Patient #1 with event times at 9, 6, and 7 days, one row for each event. The next three rows are data for Patient #14, who has an uncensored time of 16 days for the first serum sample, an uncensored time of 17 days for the second sample, and a censored time of 21 days for the third sample. The last two rows are data for Patient #28 of the high-dose group (Z1=0 and Z2=1). Since the patient did not have a second serum sample, there are only two rows of data.

To perform the WLW analysis, you specify the following statements:

```
proc phreg covs(aggregate);
   model Time*Status(0)=Z11 Z12 Z13 Z21 Z22 Z23;
   strata Enum;
   id ID;
   Z11= Z1 * (Enum=1);
   Z12= Z1 * (Enum=2);
   Z13= Z1 * (Enum=3);
   Z21= Z2 * (Enum=1);
   Z22= Z2 * (Enum=2);
   Z23= Z2 * (Enum=3);
   EqualLowDose: test Z11=Z12, Z12=Z23;
   AverageLow: test Z11,Z12,Z13 / e average;
```

```
run;
```

Two linear hypotheses are tested using the TEST statements. The specification

```
EqualLowDose: test Z11=Z12, Z12=Z13;
```

tests the null hypothesis $\beta_{11} = \beta_{12} = \beta_{13}$ of identical low-dose effects across three marginal models. The specification

```
AverageLow: test Z11,Z12,Z13 / e average;
```

tests the null hypothesis of no low-dose effects (that is, $\beta_{11} = \beta_{12} = \beta_{13} = 0$). The AVERAGE option computes the optimal weights for estimating the average low-dose effect $\bar{\beta}_1 = \beta_{11} = \beta_{12} = \beta_{13}$ and performs a 1 DF test for testing the null hypothesis that $\bar{\beta}_1 = 0$. The E option displays the coefficients for the linear hypotheses, including the optimal weights.

Marginal Cox Models for Clustered Data

Suppose there are n clusters with K_i members in the ith cluster, $i = 1, \ldots, n$. Let $\mathbf{Z}_{ki}(.)$ be the covariate process associated with the kth member of the ith cluster. The marginal Cox model is given by

$$\lambda(t; \mathbf{Z}_{ki}) = \lambda_0(t) e^{\boldsymbol{\beta}' \mathbf{Z}_{ki}(t)} k = 1, \ldots, K_i; i = 1, \ldots, n$$

where $\lambda_0(t)$ is an arbitrary baseline hazard function and $\boldsymbol{\beta}$ is the vector of regression coefficients. Lee, Wei, and Amato (1992) estimate $\boldsymbol{\beta}$ by the maximum partial likelihood estimate $\hat{\boldsymbol{\beta}}$ under the independent working assumption, and use a robust sandwich covariance estimate to account for the intracluster dependence.

To use PROC PHREG to analyze the clustered data, each member of a cluster is represented by an observation in the input data set. The input data set to PROC PHREG should contain the following:

- an ID variable to identify the cluster so that members of the same cluster have the same ID value
- a Time variable to represent the observed survival time of a member of a cluster
- a Status variable to indicate whether the Time value is an uncensored or censored time. For example, Status=1 indicates an uncensored time and Status=0 indicates a censored time.
- the explanatory variables thought to be related to the failure time

Consider a tumor study in which one of three female rats of the same litter was randomly subjected to a drug treatment. The failure time is the time from randomization to the detection of tumor. If a rat died before the tumor was detected, the failure time was censored. For example:

- In litter #1, the drug-treated rat has an uncensored time of 101 days, one untreated rat has a censored time of 49 days, and the other untreated rat has a failure time of 104 days.
- In litter #3, the drug-treated rat has a censored time of 104 days, one untreated rat has a censored time of 102 days, and the other untreated rat has a censored time of 104 days.

In this example, a litter is a cluster and the rats of the same litter are members of the cluster. Let Trt be a 0-1 variable representing the treatment a rat received, with value 1 for drug treatment and 0 otherwise. Data for the two litters of rats described earlier contribute six observations to the input data set:

Litter	Time	Status	Trt
1	101	1	1
1	49	0	0
1	104	1	0
3	104	0	1
3	102	0	0
3	104	0	0

The analysis of Lee, Wei, and Amato (1992) can be performed by PROC PHREG as follows:

```
proc phreg covs(aggregate);
   model Time*Status(0)=Treatment;
   id Litter;
run;
```

Intensity and Rate/Mean Models for Recurrent Events Data

Suppose each subject experiences recurrences of the same phenomenon. Let $N(t)$ be the number of events a subject experiences over the interval [0,t] and let $\mathbf{Z}(.)$ be the covariate process of the subject.

The intensity model (Andersen and Gill 1982) is given by

$$\lambda_{\mathbf{Z}}(t)dt = E\{dN(t)|\mathcal{F}_{t-}\} = \lambda_0(t)\mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}(t)}dt$$

where $\mathcal{F}_t$ represents all the information of the processes N and $\mathbf{Z}$ up to time t, $\lambda_0(t)$ is an arbitrary baseline intensity function, and $\boldsymbol{\beta}$ is the vector of regression coefficients. This model consists of two components: (1) all the influence of the prior events on future recurrences, if there is any, is mediated through the time-dependent covariates, and (2) the covariates have multiplicative effects on the instantaneous rate of the counting process. If the covariates are time invariant, the risk of recurrences is unaffected by the past events.

The proportional rates and means models (Pepe and Cai 1993; Lawless and Nadeau 1995; Lin et al. 2000) assume that the covariates have multiplicative effects on the mean and rate functions of the counting process. The rate function is given by

$$d\mu_{\mathbf{Z}}(t) = E\{dN(t)|\mathbf{Z}(t)\} = \mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}(t)}d\mu_0(t)$$

where $\mu_0(t)$ is an unknown continuous function and $\boldsymbol{\beta}$ is the vector of regression parameters. If $\mathbf{Z}$ is time invariant, the mean function is given by

$$\mu_{\mathbf{Z}}(t) = E\{N(t)|\mathbf{Z}\} = \mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}}\mu_0(t)$$

For both the intensity and the proportional rates/means models, estimates of the regression coefficients are obtained by solving the partial likelihood score function. However, the covariance matrix estimate for the intensity model is computed as the inverse of the observed information matrix, while that for the proportional rates/means model is given by a sandwich estimate. For a given pattern of fixed covariates, the Nelson estimate for the cumulative intensity function is the same for the cumulative mean function, but their standard errors are not the same.

To fit the intensity or rate/mean model by using PROC PHREG, the counting process style of input is needed. A subject with K events contributes $K + 1$ observations to the input data set. The kth observation of the

subject identifies the time interval from the $(k - 1)$ event or time 0 (if $k = 1$) to the kth event, $k = 1, \ldots, K$. The $(K + 1)$ observation represents the time interval from the Kth event to time of censorship. The input data set should contain the following variables:

- a TStart variable to represent the $(k - 1)$ recurrence time or the value 0 if $k = 1$
- a TStop variable to represent the kth recurrence time or the follow-up time if $k = K + 1$
- a Status variable indicating whether the TStop time is a recurrence time or a censored time; for example, Status=1 for a recurrence time and Status=0 for censored time
- explanatory variables thought to be related to the recurrence times

If the rate/mean model is used, the input data should also contain an ID variable for identifying the subjects.

Consider the chronic granulomatous disease (CGD) data listed in Fleming and Harrington (1991). The disease is a rare disorder characterized by recurrent pyrogenic infections. The study is a placebo-controlled randomized clinical trial conducted by the International CGD Cooperative Study to assess the effect of gamma interferon to reduce the rate of infection. For each study patient the times of recurrent infections along with a number of prognostic factors were collected. For example:

- Patient #17404, age 38, in the gamma interferon group had a follow-up time of 293 without any infection.
- Patient #204001, age 12, in the placebo group had an infection at 219 days, a recurrent infection at 373 days, and was followed up to 414 days.

Let Trt be the variable representing the treatment status with value 1 for gamma interferon and value 2 for placebo. Let Age be a covariate representing the age of the CGD patient. Data for the two CGD patients described earlier are given in the following table.

ID	TStart	TStop	Status	Trt	Age
174054	0	293	0	1	38
204001	0	219	1	2	12
204001	219	373	1	2	12
204001	373	414	0	2	12

Since Patient #174054 had no infection through the end of the follow-up period (293 days), there is only one observation representing the period from time 0 to the end of the follow-up. Data for Patient #204001 are broken into three observations, since there are two infections. The first observation represents the period from time 0 to the first infection, the second observation represents the period from the first infection to the second infection, and the third time period represents the period from the second infection to the end of the follow-up.

The following specification fits the intensity model:

```
proc phreg;
   model (TStart,TStop)*Status(0)=Trt Age;
run;
```

You can predict the cumulative intensity function for a given pattern of fixed covariates by specifying the CUMHAZ= option in the BASELINE statement. Suppose you are interested in two fixed patterns, one for patients of age 30 in the gamma interferon group and the other for patients of age 1 in the placebo group. You first create the SAS data set as follows:

```
data Pattern;
   Trt=1; Age=30;
   output;
   Trt=2; Age=1;
   output;
run;
```

You then include the following BASELINE statement in the PROC PHREG specification. The CUMHAZ=_all_ option produces the cumulative hazard function estimates, the standard error estimates, and the lower and upper pointwise confidence limits.

```
baseline covariates=Pattern out=out1 cumhaz=_all_;
```

The following specification of PROC PHREG fits the mean model and predicts the cumulative mean function for the two patterns of covariates in the Pattern data set:

```
proc phreg covs(aggregate);
   model (Tstart,Tstop)*Status(0)=Trt Age;
   baseline covariates=Pattern out=out2 cmf=_all_;
   id ID;
run;
```

The COVS(AGGREGATE) option, along with the ID statement, computes the robust sandwich covariance matrix estimate. The CMF=_ALL_ option adds the cumulative mean function estimates, the standard error estimates, and the lower and upper pointwise confidence limits to the OUT=Out2 data set.

PWP Models for Recurrent Events Data

Let $N(t)$ be the number of events a subject experiences by time t. Let $\mathbf{Z}(t)$ be the covariate vectors of the subject at time t. For a subject who has K events before censorship takes place, let $t_0 = 0$, let t_k be the kth recurrence time, $k = 1, \ldots, K$, and let t_{K+1} be the censored time. Prentice, Williams, and Peterson (1981) consider two time scales, a total time from the beginning of the study and a gap time from immediately preceding failure. The PWP models are stratified Cox-type models that allow the shape of the hazard function to depend on the number of preceding events and possibly on other characteristics of $\{N(t)$ and $\mathbf{Z}(t)\}$. The total time and gap time models are given, respectively, as follows:

$$\lambda(t|\mathcal{F}_{t-}) = \lambda_{0k}(t)\mathrm{e}^{\boldsymbol{\beta}_k'\mathbf{Z}(t)}, \qquad t_{k-1} < t \le t_k$$
$$\lambda(t|\mathcal{F}_{t-}) = \lambda_{0k}(t - t_{k-1})\mathrm{e}^{\boldsymbol{\beta}_k'\mathbf{Z}(t)}, \qquad t_{k-1} < t \le t_k$$

where λ_{0k} is an arbitrary baseline intensity functions, and $\boldsymbol{\beta}_k$ is a vector of stratum-specific regression coefficients. Here, a subject moves to the kth stratum immediately after his $(k - 1)$ recurrence time and remains there until the kth recurrence occurs or until censorship takes place. For instance, a subject who

experiences only one event moves from the first stratum to the second stratum after the event occurs and remains in the second stratum until the end of the follow-up.

You can use PROC PHREG to carry out the analyses of the PWP models, but you have to prepare the input data set to provide the correct risk sets. The input data set for analyzing the total time is the same as the AG model with an additional variable to represent the stratum that the subject is in. A subject with K events contributes $K + 1$ observations to the input data set, one for each stratum that the subject moves to. The input data should contain the following variables:

- a TStart variable to represent the $(k - 1)$ recurrence time or the value 0 if $k = 1$
- a TStop variable to represent the kth recurrence time or the time of censorship if $k = K + 1$
- a Status variable with value 1 if the Time value is a recurrence time and value 0 if the Time value is a censored time
- an Enum variable representing the index of the stratum that the subject is in. For a subject who has only one event at t_1 and is followed to time t_c, Enum=1 for the first observation (where Time=t_1 and Status=1) and Enum=2 for the second observation (where Time=t_c and Status=0).
- explanatory variables thought to be related to the recurrence times

To analyze gap times, the input data set should also include a GapTime variable that is equal to (TStop – TStart).

Consider the data of two subjects in CGD data described in the previous section:

- Patients #174054, age 38, in the gamma interferon group had a follow-up time of 293 without any infection.
- Patient #204001, age 12, in the placebo group had an infection at 219 days, a recurrent infection at 373 days, and a follow-up time of 414 days.

To illustrate, suppose all subjects have at most two observed events. The data for the two subjects in the input data set are as follows:

ID	TStart	TStop	Gaptime	Status	Enum	Trt	Age
174054	0	293	293	0	1	1	38
204001	0	219	219	1	1	2	12
204001	219	373	154	1	2	2	12
204001	373	414	41	0	3	2	12

Subject #174054 contributes only one observation to the input data, since there is no observed event. Subject #204001 contributes three observations, since there are two observed events.

To fit the total time model of PWP with stratum-specific slopes, either you can create the stratum-specific explanatory variables (Trt1, Trt2, and Trt3 for Trt, and Age1, Age2, and Age3 for Age) in a DATA step, or you can specify them in PROC PHREG by using programming statements as follows:

```
proc phreg;
   model (TStart,TStop)*Status(0)=Trt1 Trt2 Trt3 Age1 Age2 Age3;
   strata Enum;
   Trt1= Trt * (Enum=1);
   Trt2= Trt * (Enum=2);
   Trt3= Trt * (Enum=3);
   Age1= Age * (Enum=1);
   Age2= Age * (Enum=2);
   Age3= Age * (Enum=3);
run;
```

To fit the total time model of PWP with the common regression coefficients, you specify the following:

```
proc phreg;
   model (TStart,TStop)*Status(0)=Trt Age;
   strata Enum;
run;
```

To fit the gap time model of PWP with stratum-specific regression coefficients, you specify the following:

```
proc phreg;
   model Gaptime*Status(0)=Trt1 Trt2 Trt3 Age1 Age2 Age3;
   strata Enum;
   Trt1= Trt * (Enum=1);
   Trt2= Trt * (Enum=2);
   Trt3= Trt * (Enum=3);
   Age1= Age * (Enum=1);
   Age2= Age * (Enum=2);
   Age3= Age * (Enum=3);
run;
```

To fit the gap time model of PWP with common regression coefficients, you specify the following:

```
proc phreg;
   model Gaptime*Status(0)=Trt Age;
   strata Enum;
run;
```

Model Fit Statistics

Suppose the model contains p regression parameters. The three statistics displayed by the PHREG procedure are calculated as follows:

- –2 log likelihood:

  $$-2 \text{ Log L} = -2 \log(L_n(\hat{\boldsymbol{\beta}}))$$

  where $L_n(.)$ is a partial likelihood function for the corresponding TIES= option as described in the section "Partial Likelihood Function for the Cox Model" on page 7550, and $\hat{\boldsymbol{\beta}}$ is the maximum likelihood estimate of the regression parameter vector.

- Akaike's information criterion (AIC):

  $$\text{AIC} = -2 \text{ Log L} + 2p$$

- Schwarz Bayesian criterion (SBC):

  $$\text{SBC} = -2\,\text{Log L} + p\log(d)$$

  where d is the number of uncensored observations in the data (Volinsky and Raftery 2000). The SBC statistic is also known as the Bayesian information criterion (BIC).

The –2 log likelihood ratio statistic has a chi-square distribution under the null hypothesis (that all the explanatory effects in the model are zero) and the procedure produces a p-value for this statistic. The AIC and SBC statistics offer two different ways of adjusting the –2 log likelihood statistic for the number of terms in the model and the number of observations used. It is recommended that you use these statistics when you compare different models for the same data (for example, when you use the METHOD=STEPWISE option in the MODEL statement); lower values of the statistic indicate a more desirable model.

Schemper-Henderson Predictive Measure

Measures of predictive accuracy of regression models quantify the extent to which covariates determine an individual outcome. Schemper and Henderson's (2000) proposed predictive accuracy measure is defined as the difference between individual processes and the fitted survivor function.

For the ith individual ($1 \le i \le n$), let l_i, X_i, Δ_i, and $\mathbf{Z}_i$ be the left-truncation time, observed time, event indicator (1 for death and 0 for censored), and covariate vector, respectively. If there is no delay entry, then $l_i = 0$. Let $t_{(1)} < \cdots < t_{(m)}$ be m distinct event times with d_j deaths at $t_{(j)}$. The survival process $Y_i(t)$ for the ith individual is

$$Y_i(t) = \begin{cases} 1 & l_i \le t < X_i \\ 0 & t \ge X_i \text{ and } \Delta_i = 1 \\ \text{undefined} & t \ge X_i \text{ and } \Delta_i = 0 \end{cases}$$

Let $\hat{S}(t)$ be the Kaplan-Meier estimate of the survivor function (assuming no covariates). Let $\hat{S}(t|\mathbf{Z})$ be the fitted survivor function with covariates $\mathbf{Z}$, and if you specify TIES=EFRON, then $\hat{S}(t|\mathbf{Z})$ is computed by the Efron method; otherwise, the Breslow estimate is used.

The predictive accuracy is defined as the difference between individual survival processes $Y_i(t)$ and the fitted survivor functions with ($\hat{S}(t|\mathbf{Z}_i)$) or without ($\hat{S}(t)$) covariates between 0 and τ, the largest observed time. For each death time $t_{(j)}$, define a mean absolute distance between the $Y_i(t)$ and the $\hat{S}(t)$ as

$$\begin{aligned} \hat{M}(t_{(j)}) &= \frac{1}{n_j}\sum_{i=1}^{n} I(l_i \le t_{(j)})\Big\{ I(X_i > t_{(j)} \ge l_i)(1-\hat{S}(t_{(j)})) + \Delta_i I(X_i \le t_{(j)})\hat{S}(t_{(j)}) \\ &\quad + (1-\Delta_i)I(X_i \le t_{(j)})\left[(1-\hat{S}(t_{(j)}))\frac{\hat{S}(t_{(j)})}{\hat{S}(X_i)} + \hat{S}(t_{(j)})\left(1-\frac{\hat{S}(t_{(j)})}{\hat{S}(X_i)}\right)\right]\Big\} \end{aligned}$$

where $n_j = \sum_{i=1}^{n} I(l_i \le t_{(j)})$. Let $\hat{M}(t_{(j)}|\mathbf{Z})$ be defined similarly to $\hat{M}(t_{(j)})$, but with $\hat{S}(t_{(j)})$ replaced by $\hat{S}(t_{(j)}|\mathbf{Z}_i)$ and $\hat{S}(X_i)$ replaced by $\hat{S}(X_i|\mathbf{Z}_i)$. Let $\hat{G}(t)$ be the Kaplan-Meier estimate of the censoring or

potential follow-up distribution, and let

$$w = \sum_{j=1}^{m} \frac{d_j}{\hat{G}(t_{(j)})}$$

The overall estimator of the predictive accuracy with ($\hat{D}_z$) and without ($\hat{D}$) covariates are weighted averages of $\hat{M}(t_{(j)}|\mathbf{Z})$ and $\hat{M}(t_{(j)})$, respectively, given by

$$\begin{aligned}
\hat{D}_z &= \frac{1}{w} \sum_{j=1}^{m} \frac{d_j}{\hat{G}(t_{(j)})} \hat{M}(t_{(j)}|\mathbf{Z}) \\
\hat{D} &= \frac{1}{w} \sum_{j=1}^{m} \frac{d_j}{\hat{G}(t_{(j)})} \hat{M}(t_{(j)})
\end{aligned}$$

The explained variation by the Cox regression is

$$V = 100 \left(1 - \frac{\hat{D}_z}{\hat{D}}\right) \%$$

Because the predictive accuracy measures $\hat{D}_z$ and $\hat{D}$ are based on differences between individual survival processes and fitted survivor functions, a smaller value indicates a better prediction. For this reason, $\hat{D}_z$ and $\hat{D}$ are also referred to as predictive inaccuracy measures.

These measures of predictive accuracy can be extended for use with a stratified Cox model (Fibrinogen Studies Collaboration 2009). Let K be the number of strata. Within each stratum $k(1 \leq k \leq K)$, let $t_{k(1)} < \cdots < t_{k(m_k)}$ be the distinct death times, and let d_{kj} be the number of deaths at $t_{k(j)}$. Comparisons are made at stratum-specific death times $t_{k(j)}(j = 1, \ldots, m_k)$, rather than the common failure points of all participants, to give, for each stratum k and at each strata-specific failure point, $M_k(t_{k(j)}$ and $M_k(t_{k(j)}|\mathbf{z})$. Weighting by the stratum-specific weights $w_k = \sum_{j=1}^{m_k} \hat{G}_k^{-1}(t_j) d_{kj}$, the overall estimator of predictive accuracy with and without covariates are, respectively, given by

$$\begin{aligned}
\hat{D}_z &= \frac{1}{\sum_k w_k} \sum_k \sum_{j=1}^{m_k} \frac{d_{kj}}{\hat{G}(t_{k(j)})} \hat{M}(t_{k(j)}|\mathbf{Z}) \\
\hat{D} &= \frac{1}{\sum_k w_k} \sum_k \sum_{j=1}^{m_k} \frac{d_{kj}}{\hat{G}(t_{k(j)})} \hat{M}(t_{k(j)})
\end{aligned}$$

Residuals

This section describes the computation of residuals (RESMART=, RESDEV=, RESSCH=, and RESSCO=) in the OUTPUT statement.

First, consider TIES=BRESLOW. Let

$$
\begin{aligned}
S^{(0)}(\boldsymbol{\beta}, t) &= \sum_i Y_i(t) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(t)} \\
S^{(1)}(\boldsymbol{\beta}, t) &= \sum_i Y_i(t) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(t)} \mathbf{Z}_i(t) \\
\bar{\mathbf{Z}}(\boldsymbol{\beta}, t) &= \frac{S^{(1)}(\boldsymbol{\beta}, t)}{S^{(0)}(\boldsymbol{\beta}, t)} \\
d\Lambda_0(\boldsymbol{\beta}, t) &= \sum_i \frac{dN_i(t)}{S^{(0)}(\boldsymbol{\beta}, t)} \\
dM_i(\boldsymbol{\beta}, t) &= dN_i(t) - Y_i(t) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(t)} d\Lambda_0(\boldsymbol{\beta}, t)
\end{aligned}
$$

The martingale residual at t is defined as

$$
\hat{M}_i(t) = \int_0^t dM_i(\hat{\boldsymbol{\beta}}, s) = N_i(t) - \int_0^t Y_i(s) \mathrm{e}^{\hat{\boldsymbol{\beta}}' \mathbf{Z}_i(s)} d\Lambda_0(\hat{\boldsymbol{\beta}}, s)
$$

Here $\hat{M}_i(t)$ estimates the difference over $(0, t]$ between the observed number of events for the ith subject and a conditional expected number of events. The quantity $\hat{M}_i \equiv \hat{M}_i(\infty)$ is referred to as the martingale residual for the ith subject. When the counting process MODEL specification is used, the RESMART= variable contains the component $(\hat{M}_i(t_2) - \hat{M}_i(t_1))$ instead of the martingale residual at t_2. The martingale residual for a subject can be obtained by summing up these component residuals within the subject. For the Cox model with no time-dependent explanatory variables, the martingale residual for the ith subject with observation time t_i and event status Δ_i is

$$
\hat{M}_i = \Delta_i - \mathrm{e}^{\hat{\boldsymbol{\beta}}' \mathbf{Z}_i} \int_0^{t_i} d\Lambda_0(\hat{\boldsymbol{\beta}}, s)
$$

The deviance residuals D_i are a transform of the martingale residuals:

$$
D_i = \mathrm{sign}(\hat{M}_i) \sqrt{2\left[-\hat{M}_i - N_i(\infty) \log\left(\frac{N_i(\infty) - \hat{M}_i}{N_i(\infty)}\right)\right]}
$$

The square root shrinks large negative martingale residuals, while the logarithmic transformation expands martingale residuals that are close to unity. As such, the deviance residuals are more symmetrically distributed around zero than the martingale residuals. For the Cox model, the deviance residual reduces to the form

$$
D_i = \mathrm{sign}(\hat{M}_i) \sqrt{2[-\hat{M}_i - \Delta_i \log(\Delta_i - \hat{M}_i)]}
$$

When the counting process MODEL specification is used, values of the RESDEV= variable are set to missing because the deviance residuals can be calculated only on a per-subject basis.

The Schoenfeld (1982) residual vector is calculated on a per-event-time basis. At the jth event time t_{i_j} of the ith subject, the Schoenfeld residual

$$\hat{\mathbf{U}}_i(t_{i_j}) = \mathbf{Z}_i(t_{i_j}) - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, t_{i_j})$$

is the difference between the ith subject covariate vector at t_{i_j} and the average of the covariate vectors over the risk set at t_{i_j}. Under the proportional hazards assumption, the Schoenfeld residuals have the sample path of a random walk; therefore, they are useful in assessing time trend or lack of proportionality. Harrell (1986) proposed a z-transform of the Pearson correlation between these residuals and the rank order of the failure time as a test statistic for nonproportional hazards.

The score process for the ith subject at time t is

$$\mathbf{L}_i(\boldsymbol{\beta}, t) = \int_0^t [\mathbf{Z}_i(s) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, s)] dM_i(\boldsymbol{\beta}, s)$$

The vector $\hat{\mathbf{L}}_i \equiv \mathbf{L}_i(\hat{\boldsymbol{\beta}}, \infty)$ is the score residual for the ith subject. When the counting process MODEL specification is used, the RESSCO= variables contain the components of $(\mathbf{L}_i(\hat{\boldsymbol{\beta}}, t_2) - \mathbf{L}_i(\hat{\boldsymbol{\beta}}, t_1))$ instead of the score process at t_2. The score residual for a subject can be obtained by summing up these component residuals within the subject.

The score residuals are a decomposition of the first partial derivative of the log likelihood. They are useful in assessing the influence of each subject on individual parameter estimates. Therneau, Grambsch, and Fleming (1990) have considered a Kolmogorov-type test based on the cumulative sum of the residuals for detecting nonproportional hazards. These residuals also play an important role in the computation of the robust sandwich variance estimators of Lin and Wei (1989) and Wei, Lin, and Weissfeld (1989).

For TIES=EFRON, the preceding computation is modified to comply with the Efron partial likelihood. For a given time t, let $\Delta_i(t)$=1 if the t is an event time of the ith subject and 0 otherwise. Let $d(t) = \sum_i \Delta_i(t)$, which is the number of subjects that have an event at t. For $1 \le k \le d(t)$, let

$$\begin{aligned}
S^{(0)}(\boldsymbol{\beta}, k, t) &= \sum_i Y_i(t) \left\{ 1 - \frac{k-1}{d(t)} \Delta_i(t) \right\} \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(t)} \\
S^{(1)}(\boldsymbol{\beta}, k, t) &= \sum_i Y_i(t) \left\{ 1 - \frac{k-1}{d(t)} \Delta_i(t) \right\} \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(t)} \mathbf{Z}_i(t) \\
\bar{\mathbf{Z}}(\boldsymbol{\beta}, k, t) &= \frac{S^{(1)}(\boldsymbol{\beta}, k, t)}{S^{(0)}(\boldsymbol{\beta}, k, t)} \\
d\Lambda_0(\boldsymbol{\beta}, k, t) &= \sum_i \frac{dN_i(t)}{S^{(0)}(\boldsymbol{\beta}, k, t)} \\
dM_i(\boldsymbol{\beta}, k, t) &= dN_i(t) - Y_i(t) \left(1 - \Delta_i(t) \frac{k-1}{d(t)} \right) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_i(t)} d\Lambda_0(\boldsymbol{\beta}, k, t)
\end{aligned}$$

The martingale residual at t for the ith subject is defined as

$$\hat{M}_i(t) = \int_0^t \frac{1}{d(s)} \sum_{k=1}^{d(s)} dM_i(\hat{\boldsymbol{\beta}}, k, s) = N_i(t) - \int_0^t \frac{1}{d(s)} \sum_{k=1}^{d(s)} Y_i(s) \left(1 - \Delta_i(s) \frac{k-1}{d(s)} \right) \mathrm{e}^{\hat{\boldsymbol{\beta}}' \mathbf{Z}_i(s)} d\Lambda_0(\hat{\boldsymbol{\beta}}, k, s)$$

Deviance residuals are computed by using the same transform on the corresponding martingale residuals as in TIES=BRESLOW.

The Schoenfeld residual vector for the ith subject at event time t_{i_j} is

$$\hat{\mathbf{U}}_i(t_{i_j}) = \mathbf{Z}_i(t_{i_j}) - \frac{1}{d(t_{i_j})} \sum_{k=1}^{d(t_{i_j})} \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, k, t_{i_j})$$

The score process for the ith subject at time t is given by

$$\mathbf{L}_i(\boldsymbol{\beta}, t) = \int_0^t \frac{1}{d(s)} \sum_{k=1}^{d(s)} \Big(\mathbf{Z}_i(s) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, k, s) \Big) dM_i(\boldsymbol{\beta}, k, s)$$

For TIES=DISCRETE or TIES=EXACT, it is difficult to come up with modifications that are consistent with the corresponding partial likelihood. Residuals for these TIES= methods are computed by using the same formulas as in TIES=BRESLOW.

Diagnostics Based on Weighted Residuals

ZPH Diagnostics

The vector of weighted Schoenfeld residuals, $\mathbf{r}_i$, is computed as

$$\mathbf{r}_i = n_e \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \hat{\mathbf{U}}_i(t_i)$$

where n_e is the total number of events and $\hat{\mathbf{U}}_i(t_i)$ is the vector of Schoenfeld residuals at event time t_i. The components of $\mathbf{r}_i$ are output to the WTRESSCH= variables in the OUTPUT statement.

The weighted Schoenfeld residuals are useful in assessing the proportional hazards assumption. The idea is that most of the common alternatives to the proportional hazards can be cast in terms of a time-varying coefficient model,

$$\lambda(t, \mathbf{Z}) = \lambda_0(t) \exp(\beta_1(t) Z_1 + \beta_2(t) Z_2 + \cdots)$$

where $\lambda(t, \mathbf{Z})$ and $\lambda_0(t)$ are hazard rates. Let $\hat{\beta}_j$ and r_{ij} be the jth component of $\hat{\boldsymbol{\beta}}$ and $\mathbf{r}_i$, respectively. Grambsch and Therneau (1994) suggest using a smoothed plot of $(\hat{\beta}_j + r_{ij})$ versus t_i to discover the functional form of the time-varying coefficient $\beta_j(t)$. A zero slope indicates that the coefficient does not vary with time.

DFBETA Diagnostics

The weighted score residuals are used more often than their unscaled counterparts in assessing local influence. Let $\hat{\boldsymbol{\beta}}_{(i)}$ be the estimate of $\boldsymbol{\beta}$ when the ith subject is left out, and let $\delta\hat{\boldsymbol{\beta}}_i = \hat{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{(i)}$. The jth component of $\delta\hat{\boldsymbol{\beta}}_i$ can be used to assess any untoward effect of the ith subject on $\hat{\beta}_j$. The exact computation of $\delta\hat{\boldsymbol{\beta}}_i$ involves refitting the model each time a subject is omitted. Cain and Lange (1984) derived the following approximation of $\boldsymbol{\Delta}_i$ as weighted score residuals:

$$\delta\hat{\boldsymbol{\beta}}_i = \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \hat{\mathbf{L}}_i$$

Here, $\hat{\mathbf{L}}_i$ is the vector of the score residuals for the ith subject. Values of $\delta\hat{\boldsymbol{\beta}}_i$ are output to the DFBETA= variables. Again, when the counting process MODEL specification is used, the DFBETA= variables contain the component $\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})(\mathbf{L}_i(\hat{\boldsymbol{\beta}}, t_2) - \mathbf{L}_i(\hat{\boldsymbol{\beta}}, t_1))$, where the score process $\mathbf{L}_i(\boldsymbol{\beta}, t)$ is defined in the section "Residuals" on page 7585. The vector $\delta\hat{\boldsymbol{\beta}}_i$ for the ith subject can be obtained by summing these components within the subject.

Note that these DFBETA statistics are a transform of the score residuals. In computing the robust sandwich variance estimators of Lin and Wei (1989) and Wei, Lin, and Weissfeld (1989), it is more convenient to use the DFBETA statistics than the score residuals (see Example 92.10).

Influence of Observations on Overall Fit of the Model

The LD statistic approximates the likelihood displacement, which is the amount by which minus twice the log likelihood ($-2\log L(\hat{\boldsymbol{\beta}})$), under a fitted model, changes when each subject in turn is left out. When the ith subject is omitted, the likelihood displacement is

$$2\log L(\hat{\boldsymbol{\beta}}) - 2\log L(\hat{\boldsymbol{\beta}}_{(i)})$$

where $\hat{\boldsymbol{\beta}}_{(i)}$ is the vector of parameter estimates obtained by fitting the model without the ith subject. Instead of refitting the model without the ith subject, Pettitt and Bin Daud (1989) propose that the likelihood displacement for the ith subject be approximated by

$$\mathrm{LD}_i = \hat{\mathbf{L}}_i' \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \hat{\mathbf{L}}_i$$

where $\hat{\mathbf{L}}_i$ is the score residual vector of the ith subject. This approximation is output to the LD= variable.

The LMAX statistic is another global influence statistic. This statistic is based on the symmetric matrix

$$\mathbf{B} = \mathbf{L}\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})\mathbf{L}'$$

where $\mathbf{L}$ is the matrix with rows that are the score residual vectors $\hat{\mathbf{L}}_i$. The elements of the eigenvector associated with the largest eigenvalue of the matrix $\mathbf{B}$, standardized to unit length, give a measure of the sensitivity of the fit of the model to each observation in the data. The influence of the ith subject on the global fit of the model is proportional to the magnitude of ζ_i, where ζ_i is the ith element of the vector $\boldsymbol{\zeta}$ that satisfies

$$\mathbf{B}\boldsymbol{\zeta} = \lambda_{\max}\boldsymbol{\zeta} \quad \text{and} \quad \boldsymbol{\zeta}'\boldsymbol{\zeta} = 1$$

with $\lambda_{\max}$ being the largest eigenvalue of $\mathbf{B}$. The sign of ζ_i is irrelevant, and its absolute value is output to the LMAX= variable.

When the counting process MODEL specification is used, the LD= and LMAX= variables are set to missing, because these two global influence statistics can be calculated on a per-subject basis only.

Concordance Statistics

The predictive accuracy of a statistical model can be measured by the agreement between observed and predicted outcomes. In the context of logistic regression with binary outcomes, the concordance statistic (also known as C-statistic) is the most commonly used measure of accuracy. The concept underlying concordance is that a subject who experiences a particular outcome has a higher predicted probability of that outcome than a subject who does not experience the outcome.

The C-statistic can be calculated as the proportion of pairs of subjects whose observed and predicted outcomes agree (are concordant) among all possible pairs in which one subject experiences the outcome of interest and the other one does not. The higher the C-statistic, the better the model can discriminate between subjects who do experience the outcome of interest and subjects who do not.

C-statistics can be formulated for any modeling approach that generates predicted values. In the context of survival analysis, various C-statistics have been formulated to deal with right-censored data. PROC PHREG provides concordance statistics that were introduced by Harrell (1986) and Uno et al. (2011). The following subsections discuss these statistics. In these subsections, $\boldsymbol{\beta}$ denotes the true regression parameters, and for a pair of subjects whose covariate vectors are $\mathbf{Z}_1$ and $\mathbf{Z}_2$ the survival times are denoted as T_1 and T_2 and the censoring times are denoted as D_1 and D_2, respectively. For the ith individual ($1 \le i \le n$) in a sample, let X_i, Δ_i, and $\mathbf{Z}_i$ be the observed time, event indicator (1 for death and 0 for censored), and covariate vector, respectively. Let $\hat{\boldsymbol{\beta}}$ denote the maximum partial likelihood estimates of $\boldsymbol{\beta}$.

Harrell's Concordance Statistic

Harrell (1986) proposes the following definition of the concordance probability:

$$C_H = \Pr(\boldsymbol{\beta}'\mathbf{Z}_1 > \boldsymbol{\beta}'\mathbf{Z}_2 | T_1 < T_2, T_1 < \min(D_1, D_2))$$

Assuming no ties in the event times and the predictor scores, C_H can be estimated by

$$\hat{C}_H = \frac{\sum_{i \neq j} \Delta_i I(X_i < X_j) I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i > \hat{\boldsymbol{\beta}}'\mathbf{Z}_j)}{\sum_{i \neq j} \Delta_i I(X_i < X_j)}$$

When there are ties in the predictor scores, the preceding calculation can be adjusted to be

$$\hat{C}_H = \frac{\sum_{i \neq j} \Delta_i I(X_i < X_j) \left[I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i > \hat{\boldsymbol{\beta}}'\mathbf{Z}_j) + 0.5 I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i = \hat{\boldsymbol{\beta}}'\mathbf{Z}_j) \right]}{\sum_{i \neq j} \Delta_i I(X_i < X_j)}$$

Assuming that the censoring time is independent of the event time, Kang et al. (2015) derive the standard errors estimator by using the delta method. Note this derivation assumes that $\hat{\boldsymbol{\beta}}$ is fixed, so it does not account for the variability in estimating $\boldsymbol{\beta}$. In order to show this condition more explicitly, the linear predictor $\boldsymbol{\beta}'\mathbf{Z}$ is replaced by a single variable Y. For a pair of subjects i and j, define the following quantities:

$$\mathrm{sgn}(Y_i, Y_j) = I(Y_i \ge Y_j) - I(Y_i \le Y_j)$$

$$\mathrm{csgn}(X_i, \Delta_i, X_j, \Delta_j) = I(X_i \ge X_j)\Delta_j - I(X_i \le X_j)\Delta_i$$

Let $t_{ijXY} = \mathrm{csgn}(X_i, \Delta_i, X_j, \Delta_j)\,\mathrm{sgn}(Y_i, Y_j)$, $t^*_{ijXX} = \mathrm{csgn}(X_i, \Delta_i, X_j, \Delta_j)^2$. Further define the following quantities:

$$t_{XY} = \frac{1}{n(n-1)} \sum_{i \neq j} t_{ijXY}$$

$$t^*_{XX} = \frac{1}{n(n-1)} \sum_{i \neq j} t^*_{ijXX}$$

Harrell's estimator can be rewritten as

$$\hat{C}_H = \frac{1}{2}\left(\frac{t_{XY}}{t^*_{XX}} + 1\right)$$

Applying the delta method, the variance of Harrell's C-statistic can be estimated by

$$\widehat{\mathrm{var}}(\hat{C}_H) = \begin{bmatrix} \frac{1}{t^*_{XX}} & -\frac{t_{XY}}{t^{*2}_{XX}} \end{bmatrix} \begin{bmatrix} \widehat{\mathrm{var}}(t_{XY}) & \widehat{\mathrm{cov}}(t^*_{XX}, t_{XY}) \\ \widehat{\mathrm{cov}}(t^*_{XX}, t_{XY}) & \widehat{\mathrm{var}}(t^*_{XX}) \end{bmatrix} \begin{bmatrix} \frac{1}{t^*_{XX}} & -\frac{t_{XY}}{t^{*2}_{XX}} \end{bmatrix}'$$

where

$$\widehat{\mathrm{var}}(t^*_{XX}) = \frac{4\sum_i \left(\sum_j t^*_{ijXX}\right)^2 - \sum_i \sum_j t^{*2}_{ijXX} - \frac{2(2n-3)}{n(n-1)} \left(\sum_i \sum_j t^{*2}_{ijXX}\right)^2}{n(n-1)(n-2)(n-3)}$$

$$\widehat{\mathrm{var}}(t_{XY}) = \frac{4\sum_i \left(\sum_j t_{ijXY}\right)^2 - \sum_i \sum_j t_{ijXY} - \frac{2(2n-3)}{n(n-1)} \left(\sum_i \sum_j t_{ijXY}\right)^2}{n(n-1)(n-2)(n-3)}$$

$$\widehat{\mathrm{cov}}(t^*_{XX}, t_{XY}) = \frac{4\sum_i \left(\sum_j t^*_{ijXX} \sum_{j'} t_{ij'XY}\right) - \sum_i \sum_j t^*_{ijXX} t_{ijXY} - \frac{2(2n-3)}{n(n-1)} \left(\sum_i \sum_j t^{*2}_{ijXX}\right)\left(\sum_i \sum_j t_{ijXY}\right)}{n(n-1)(n-2)(n-3)}$$

Uno's Concordance Statistic

Uno et al. (2011) propose the following method for estimating the concordance probability:

$$C_U = \Pr(\boldsymbol{\beta}'\mathbf{Z}_1 > \boldsymbol{\beta}'\mathbf{Z}_2 | T_1 < T_2)$$

If τ is a specified time point within the support of the censoring variable, Uno et al. (2011) also define a truncated version of the concordance probability as

$$C_U = \Pr(\boldsymbol{\beta}'\mathbf{Z}_1 > \boldsymbol{\beta}'\mathbf{Z}_2 | T_1 < T_2, T_1 < \tau)$$

You can specify a τ value in the TAU= option in the PROC PHREG statement. If the TAU= option is not specified, there is no truncation and the τ value is taken as the largest event time.

For the ith individual ($1 \le i \le n$), let X_i, Δ_i, and $\mathbf{Z}_i$ be the observed time, event indicator (1 for death and 0 for censored), and covariate vector, respectively. Let $\hat{G}(t)$ be the Kaplan-Meier estimate of the censoring distribution (assuming no covariates). C_U is consistently estimated by

$$\hat{C}_U = \frac{\sum_{i=1}^n \sum_{j=1}^n \Delta_i \hat{G}(X_i^-)^{-2} I(X_i < X_j, X_i < \tau)\left[I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i > \hat{\boldsymbol{\beta}}'\mathbf{Z}_j) + 0.5 * I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i = \hat{\boldsymbol{\beta}}'\mathbf{Z}_j)\right]}{\sum_{i=1}^n \sum_{j=1}^n \Delta_i \hat{G}(X_i^-)^{-2} I(X_i < X_j, X_i < \tau)}$$

Define $W = \sqrt{n}(\hat{C}_U - C_U)$. It can be shown that W is asymptotically distributed as a normal random variable with mean zero. The variance of W can be approximated by using the perturbation-resampling method. Specifically, let $\{\psi_i, i = 1, \ldots, n\}$ be a set of independent samples from an exponential distribution with mean of 1 and variance of 1. For a large n, W can be approximated by

$$\tilde{W} = \sum_{i<j} 0.5 * \left[\hat{V}_{ij}(\hat{\boldsymbol{\beta}}) + \hat{V}_{ji}(\hat{\boldsymbol{\beta}})\right] \psi_i \psi_j + \left[\hat{K}(G^*) - \hat{K}(\hat{G})\right] + \left[\hat{C}_U(\boldsymbol{\beta}^*) - \hat{C}_U(\hat{\boldsymbol{\beta}})\right]$$

where

$$\hat{K}(G) = \frac{\sum_{i<j} G(X_i^-)^{-2} I(X_i < X_j, X_i < \tau)\Delta_i \left[I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i > \hat{\boldsymbol{\beta}}'\mathbf{Z}_j) + 0.5 * I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i = \hat{\boldsymbol{\beta}}'\mathbf{Z}_j) - \hat{C}_\tau(\hat{\boldsymbol{\beta}})\right]}{\sum_{i<j} \hat{G}(X_i^-)^{-2} I(X_i < X_j, X_i < \tau)\Delta_i},$$

$$\hat{V}_{ij}(\hat{\boldsymbol{\beta}}) = \frac{G(X_i^-)^{-2} I(X_i < X_j, X_i < \tau)\Delta_i \left[I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i > \hat{\boldsymbol{\beta}}'\mathbf{Z}_j) + 0.5 * I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i = \hat{\boldsymbol{\beta}}'\mathbf{Z}_j) - \hat{C}_\tau(\hat{\boldsymbol{\beta}})\right]}{\sum_{i<j} \hat{G}(X_i^-)^{-2} I(X_i < X_j, X_i < \tau)\Delta_i}$$

and $G^*(\cdot)$ and $\boldsymbol{\beta}^*$ are the perturbed versions of $\hat{G}$ and $\hat{\boldsymbol{\beta}}$. $G^*(\cdot)$ is calculated as

$$G^*(t) = \hat{G}(t) - \hat{G}(t)\frac{2}{n(n-1)} \sum_{i<j} \int_0^t \frac{1}{n^{-1}\sum_i I(X_i \geq u)} \left[d\hat{M}(u) + d\hat{M}(u)\right] \psi_i \psi_j / 2$$

where $\hat{M}(t) = I(X_i \leq u, \Delta_i = 0) - \int_0^t I(X_i \geq u) d\hat{\Lambda}_C(u)$ and $\hat{\Lambda}_C(\cdot)$ is a consistent estimator of the cumulative hazard function for the censoring time variable. $\boldsymbol{\beta}^*$ is calculated as

$$\boldsymbol{\beta}^* = \hat{\boldsymbol{\beta}} + \frac{2}{n(n-1)} \sum_{i<j} \left\{\hat{H}(\hat{\boldsymbol{\beta}}) \left[U_i(\hat{\boldsymbol{\beta}}) + U_j(\hat{\boldsymbol{\beta}})\right] / 2\right\} \psi_i \psi_j$$

where $\hat{H}$ is the estimated variance-covariance matrix of $\hat{\boldsymbol{\beta}}$ divided by n and U_i is the contribution to the partial likelihood function from the ith individual. The third term of the formula for $\tilde{W}$ is dropped out if you use the PRED= option in an ROC statement to specify a variable that contains the prediction scores.

Suppose $\hat{\sigma}^2$ is the sample variance based on M realizations of $\tilde{W}$. The $100(1-\alpha)\%$ confidence limits for C_U are $\hat{C}_U \pm z_{\alpha/2}\hat{\sigma}$, where $z_{\alpha/2}$ is the upper $100\alpha/2$ percentile of the standard normal distribution.

Time-Dependent ROC Curves

In the context of logistic regression with binary outcomes, receiver operator characteristic (ROC) curves and AUC (area under the ROC curve) statistics are commonly used to assess the ability of the model to discriminate between the two outcomes. To adapt the concept of ROC curves to the survival setting, various definitions and estimators of time-dependent ROC curves and AUC functions have been proposed. See Blanche, Latouche, and Viallon (2013) for a comprehensive survey of different methods. Time-dependent ROC curves and AUC functions characterize how well the fitted model can distinguish between subjects who experience an event from subjects who are event-free.

Whereas C-statistics provide overall measures of predictive accuracy, time-dependent ROC curves and AUC functions summarize the predictive accuracy at specific times. In practice, it is common to use several time points within the support of the observed event times.

Let T denote the event-time variable, and let Y denote the continuous variable to be assessed. At time t, a binary outcome can be defined as follows:

$$D_t = I(T \le t)$$

Suppose c denotes a specific value within the support of Y. The sensitivity (SE) and specificity (SP) can be defined as

$$\mathrm{SE}_t(c) = \Pr(Y > c | D_t = 1)$$

$$\mathrm{SP}_t(c) = \Pr(Y \le c | D_t = 0)$$

The ROC curve at time t is defined to be

$$\mathrm{ROC}_t(p) = \mathrm{SE}_t(c_p), \quad c_p = \mathrm{SP}_t^{-1}(1-p)$$

This definition is often referred to as the "cumulative/dynamic" ROC curve in the literature. "Cumulative" means all events that occurred before time t are considered as "cases." Other types of time-dependent ROC curves are available in the literature—for example, in Heagerty and Zheng (2005).

The AUC statistic at time t is the area under the ROC curve at time t:

$$\mathrm{AUC}_t = \int \mathrm{ROC}_t(u)du$$

Let $\boldsymbol{\beta}$ denote the vector of regression parameters. For the ith individual ($1 \le i \le n$), let X_i, Δ_i, and $\mathbf{Z}_i$ be the observed time, event indicator (1 for death and 0 for censored), and covariate vector, respectively. Let $\hat{\boldsymbol{\beta}}$ denote the maximum partial likelihood estimates of $\boldsymbol{\beta}$. The estimated linear predictor for the ith individual is $Y_i = \boldsymbol{\beta}'\mathbf{Z}_i$. PROC PHREG supports the approaches that are described in the following sections for estimating time-dependent ROC curves.

Inverse Probability of Censoring Weighting Approach

Let $\hat{G}(t)$ be the Kaplan-Meier estimate of the censoring distribution (assuming no covariates). Assuming that the censoring distribution is independent of the failure time distribution, the sensitivity and specificity under a specific threshold value c can be consistently estimated by

$$\widehat{\mathrm{SE}}_t(c) = \frac{\sum_{i=1}^n \Delta_i I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i > c, X_i \le t)/\hat{G}(X_i)}{\sum_{i=1}^n \Delta_i I(X_i \le t)/\hat{G}(X_i)}$$

$$\widehat{\mathrm{SP}}_t(c) = \frac{\sum_{i=1}^n I(\hat{\boldsymbol{\beta}}'\mathbf{Z}_i \le c, X_i > t)}{\sum_{i=1}^n I(X_i > t)}$$

$\mathrm{ROC}_t(c)$ can be estimated by substituting in these estimated sensitivities and specificities. The estimated AUC_t is calculated by using the trapezoidal rule to integrate the estimated $\mathrm{ROC}_t(c)$ curve.

Uno et al. (2007) propose estimating the standard errors of the AUC_t estimator by using the perturbation-resampling method. Let $\{\psi_i, i = 1, \ldots, n\}$ be a set of independent samples from an exponential distribution with mean of 1 and variance of 1. The perturbed versions of $\widehat{\mathrm{SE}}_t(c)$ and $\widehat{\mathrm{SP}}_t(c)$ are

$$\widehat{\mathrm{SE}}_t^*(c) = \frac{\sum_{i=1}^n \Delta_i I(\boldsymbol{\beta}^{*\prime}\mathbf{Z}_i > c, X_i \le t)\psi_i/\hat{G}^*(X_i)}{\sum_{i=1}^n \Delta_i I(X_i \le t)\psi_i/\hat{G}^*(X_i)}$$

$$\widehat{\mathrm{SP}}_t^*(c) = \frac{\sum_{i=1}^n I(\boldsymbol{\beta}^{*'}\mathbf{Z}_i \le c, X_i > t)\psi_i}{\sum_{i=1}^n I(X_i > t)\psi_i}$$

where $G^*(\cdot)$ and $\boldsymbol{\beta}^*$ represent the perturbed versions of $\hat{G}(\cdot)$ and $\hat{\boldsymbol{\beta}}$. $G^*(\cdot)$ is calculated as

$$G^*(t) = \hat{G}(t) - \hat{G}(t)\frac{2}{n(n-1)}\sum_{i<j}\int_0^t \frac{1}{n^{-1}\sum_i I(X_i \ge u)}\left[d\hat{M}(u) + d\hat{M}(u)\right]\psi_i\psi_j/2$$

where $\hat{M}(t) = I(X_i \le u, \Delta_i = 0) - \int_0^t I(X_i \ge u)d\hat{\Lambda}_C(u)$ and $\hat{\Lambda}_C(\cdot)$ is a consistent estimator of the cumulative hazard function for the censoring time variable. $\boldsymbol{\beta}^*$ is calculated as

$$\boldsymbol{\beta}^* = \hat{\boldsymbol{\beta}} + \frac{2}{n(n-1)}\sum_{i<j}\left\{\hat{H}(\hat{\boldsymbol{\beta}})\left[U_i(\hat{\boldsymbol{\beta}}) + U_j(\hat{\boldsymbol{\beta}})\right]/2\right\}\psi_i\psi_j$$

where $\hat{H}$ is the estimated variance-covariance matrix of $\hat{\boldsymbol{\beta}}$ divided by n and U_i is the partial likelihood contribution from the ith individual.

The perturbed AUC_t estimate is obtained by substituting in the perturbed sensitivities and specificities. Suppose $\hat{\sigma}^2$ is the sample variance based on M realizations of the perturbed AUC_t. The $100(1-\alpha)\%$ confidence limits for AUC_t are $\widehat{\mathrm{AUC}}_t \pm z_{\alpha/2}\hat{\sigma}$, where $\widehat{\mathrm{AUC}}_t$ is the estimated AUC_t and $z_{\alpha/2}$ is the upper $100\alpha/2$ percentile of the standard normal distribution.

To choose this method of computing the time-dependent ROC curve, specify METHOD=IPCW in the ROCOPTIONS option in the PROC PHREG statement.

NOTE: This perturbation approach of estimating the standard error of the AUC_t statistic does not apply to a model that is specified by the PRED= option or SOURCE= option in an ROC statement.

Conditional Kaplan-Meier Approach

By using Bayes' theorem, sensitivity and the specificity can be written as

$$\mathrm{SE}_t(c) = \Pr(Y > c|D_t = 1) = \frac{[1 - S(t|Y > c)]\Pr(Y > c)}{1 - S(t)}$$

$$\mathrm{SP}_t(c) = \Pr(Y \le c|D_t = 0) = \frac{S(t|Y \le c)\Pr(Y \le c)}{S(t)}$$

where $S(\cdot)$ is the survivor function and $S(\cdot|Y > c)$ is the conditional survivor function for $Y > c$.

Heagerty, Lumley, and Pepe (2000) use the Kaplan-Meier method to estimate the survivor function $S(.)$ and the conditional survivor function $S(.|Y > c)$. The latter was estimated using subjects where the condition $Y > c$ is met. The sensitivity and the specificity are estimated by

$$\widehat{\mathrm{SE}}_t(c) = \frac{\left[1 - \hat{S}_{KM}(t|Y > c)\right][1 - \hat{F}_Y(c)]}{1 - \hat{S}_{KM}(t)}$$

$$\widehat{\mathrm{SP}}_t(c) = \frac{\hat{S}_{KM}(t|Y \le c)\hat{F}_Y(c)}{\hat{S}_{KM}(t)}$$

where $\hat{S}_{KM}(\cdot)$ is the Kaplan-Meier estimator and $\hat{F}_Y(c) = \sum_i I(Y_i \le c)/n$.

To choose this method of computing ROC curves, specify METHOD=KM in the ROCOPTIONS in the PROC PHREG statement.

Nearest Neighbors Approach

Following Akritas (1994), the bivariate survival function, $S(c,t) = \Pr(Y > c, T > t)$, can be estimated by

$$\hat{S}_{b_n}(c,t) = \frac{1}{n}\sum_i \hat{S}_{b_n}(t|Y = Y_i)I(Y_i > c)$$

where $\hat{S}_{b_n}(t|Y = Y_i)$ is a smoothed estimate of the conditional survival function. Define the weighted Kaplan-Meier estimator as

$$\hat{S}_{b_n}(t|Y = Y_i) = \prod_{s\in\{X_i:i=1,\ldots,n,\Delta_i=1\},s\le t}\left[1 - \frac{\sum_j K_{b_n}(Y_i,Y_j)I(X_j = s)\Delta_j}{\sum_j K_{b_n}(Y_i,Y_j)I(X_j \ge s)}\right]$$

where $K_{b_n}(Y_i,Y_j)$ is a kernel function that depends on the parameter b_n. Akritas (1994) uses the nearest neighbor kernel, $K_{b_n}(Y_i,Y_j) = I\{-b_n < \hat{F}_Y(Y_i) - \hat{F}_Y(Y_j) < b_n\}$, where $0 < 2b_n < 1$; this effectively selects the nearest $2b_n$ proportion of observations in the neighborhood. The default value for b_n is 0.05. You can specify a different value by using the SPAN= suboption in METHOD=NNE in the ROCOPTIONS option in the PHREG statement.

The sensitivity and specificity can then be estimated as

$$\widehat{\mathrm{SE}}_t(c) = \frac{1 - \hat{F}_Y(c) - \hat{S}_{b_n}(c,t)}{1 - \hat{S}_{b_n}(t)}$$

$$\widehat{\mathrm{SP}}_t(c) = 1 - \frac{\hat{S}_{b_n}(c,t)}{\hat{S}_{b_n}(t)}$$

where $\hat{S}_{b_n}(t) = \hat{S}_{b_n}(-\infty,t)$. For more information, see Heagerty, Lumley, and Pepe (2000).

To choose this method of computing time-dependent ROC curves, specify METHOD=NNE in the ROCOPTIONS option in the PROC PHREG statement.

Recursive Approach

Chambless and Diao (2006) propose estimating time-dependent ROC curves by using a recursive approach akin to the Kaplan-Meier method. Let $t_1 < t_2 < \cdots < t_M$ be the distinct event times in the data. The area under the curve at time t_m, $1 \le t_m \le M$, can be derived as

$$\mathrm{AUC}_{t_m} = \frac{\sum_{k=1}^m \gamma_k\lambda(t_k)(1-\lambda(t_k))S(t_{k-1}) - \sum_{k=1}^m \tau_k\lambda(t_k)(1-S(t_{k-1}))S(t_{k-1})}{S(t_m)(1-S(t_m))}$$

where $S(\cdot)$ is the survivor function, $\lambda(\cdot)$ is the hazard function, $t_0 = 0$, $\tau_0 = 0$, and

$$\tau_k = \Pr(\boldsymbol{\beta}'\mathbf{Z}_i > \boldsymbol{\beta}'\mathbf{Z}_j|X_i = t_k, \Delta_i = 1, X_j > t_k)$$

$$\gamma_k = \Pr(\boldsymbol{\beta}'\mathbf{Z}_i > \boldsymbol{\beta}'\mathbf{Z}_j|X_i = t_{k-1}, \Delta_i = 1, X_j = t_k, \Delta_j = 1)$$

In a recursive fashion, the sensitivity and specificity at time t_m can be shown to be

$$\mathrm{SE}_{t_m}(c) = \sum_{k=1}^m \rho_k(c)\lambda(t_k)S(t_{k-1})/[1 - S(t_m)]$$

$$\mathrm{SP}_{t_m}(c) = \frac{\Pr(\boldsymbol{\beta}'\mathbf{Z}_i \le c) - \sum_{k=1}^{m}[1-\rho_k(c)]\lambda(t_k)S(t_{k-1})}{S(t_m)}$$

where $\rho_k(c) = \Pr(\boldsymbol{\beta}'\mathbf{Z}_i > c | X_i = t_k, \Delta_i = 1)$.

Define $\mathcal{R}_k$ to be the risk set at time t_k, and let r_k be the number of subjects in $\mathcal{R}_k$. Let $\mathbf{Z}_{(k)}$ be the covariate vector for the subject whose event time is t_k. The unknown parameters τ_k, γ_k, and $\rho_k(c)$ can be estimated by

$$\hat{\tau}_k = \frac{1}{k-1}\sum_{i=1}^{k} I(\hat{\boldsymbol{\beta}}'Z_{(i)} > \hat{\boldsymbol{\beta}}'Z_{(k)})$$

$$\hat{\gamma}_k = \frac{1}{r_k-1}\sum_{j\in\mathcal{R}_k} I(\hat{\boldsymbol{\beta}}'Z_{(k)} > \hat{\boldsymbol{\beta}}'Z_j)$$

$$\hat{\rho}_k(c) = I(\hat{\boldsymbol{\beta}}'Z_{(k)} > c)$$

When there is only one event at each event time, $\lambda(t_k)$ is estimated by $\hat{\lambda}(t_k) = 1/r_k$ and $S(t_k)$ is estimated by the Kaplan-Meier method as $\hat{S}(t_k) = \hat{S}(t_{k-1})[1-\hat{\lambda}(t_{k-1})]$. In the case of a tie, the order of the events in the calculation is the same as the order of their appearance in the input data set.

To choose this method of computing time-dependent ROC curves, specify METHOD=RECURSIVE in the ROCOPTIONS option in the PROC PHREG statement.

Weighted Kernel Kaplan-Meier Approach

Let $S(t|Y) = \Pr(T > t|Y)$ denote the conditional survival function of T given Y. For the ith subject, define the weight W_i to be the probability of being a nonsurvivor at time t. Table 92.16 displays how W_i can be computed.

Table 92.16 Probability of Being a Nonsurvivor for the ith Subject

t, X_i	**Status**	W_i
$X_i > t$	$\Delta_i = 0$ or $\Delta_i = 1$	0
$X_i \le t$	$\Delta_i = 1$	1
$X_i = t$	$\Delta_i = 0$	0
$X_i < t$	$\Delta_i = 0$	$1 - \frac{S(t\|Y_i)}{S(X_i\|Y_i)}$

You can estimate $S(t|Y_i)$ by using a kernel-based Kaplan-Meier-type method:

$$\hat{S}_{b_n}(t|Y = Y_i) = \prod_{s\in\{X_i:i=1,\ldots,n,\Delta_i=1\},s\le t}\left[1 - \frac{\sum_j K_{b_n}(Y_i,Y_j)I(X_j = s)\Delta_j}{\sum_j K_{b_n}(Y_i,Y_j)I(X_j \ge s)}\right]$$

where $K_{b_n}(Y_i, Y_j)$ is a kernel function that depends on the parameter b_n. PROC PHREG uses the nearest-neighbors kernel, $K_{b_n}(Y_i, Y_j) = I\{-b_n < \hat{F}_Y(Y_i) - \hat{F}_Y(Y_j) < b_n\}$, where $0 < 2b_n < 1$; this effectively selects the nearest $2b_n$ proportion of observations in the neighborhood. The default value for b_n is 0.05. You can specify a different value by using the SPAN= suboption of the METHOD=WKKM option, which you specify in the ROCOPTIONS option in the PHREG statement.

At time t, you can estimate the sensitivity and specificity as

$$\widehat{\mathrm{SE}}_t(c) = \frac{\sum_{i=1}^{n} W_i I(Y_i > c)}{\sum_{i=1}^{n} W_i}$$

$$\widehat{\mathrm{SP}}_t(c) = \frac{\sum_{i=1}^{n} (1 - W_i) I(Y_i \le c)}{\sum_{i=1}^{n} (1 - W_i)}$$

Survivor Function Estimators

Three estimators of the survivor function are available: the Breslow (1972) estimator, which is based on the empirical cumulative hazard function, the Fleming and Harrington (1984) estimator, which is a tie-breaking modification of the Breslow estimator, and the product-limit estimator (Kalbfleisch and Prentice 1980, pp. 84–86).

Let $\{t_1 < \cdots < t_k\}$ be the distinct uncensored times of the survival data.

Breslow Estimator

To select this estimator, specify the METHOD=BRESLOW option in the BASELINE statement or OUTPUT statement. For the jth subject, let $\{(X_j, \Delta_j, \mathbf{Z}_j(.))\}$ represent the failure time, the event indicator, and the vector of covariate values, respectively. For $t \ge 0$, let

$$\begin{aligned} Y_j(t) &= I(X_j \ge t) \\ \Delta_j(t) &= \begin{cases} 1 & X_j = t \text{ and } \Delta_j = 1 \\ 0 & \text{otherwise} \end{cases} \\ d(t) &= \sum_j \Delta_j(t) \end{aligned}$$

Note that $d(t)$ is the number of subjects that have an event at t. Let

$$\begin{aligned} S^{(0)}(\boldsymbol{\beta}, t) &= \sum_j Y_j(t) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_j(t)} \\ S^{(1)}(\boldsymbol{\beta}, t) &= \sum_j Y_j(t) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{Z}_j(t)} \mathbf{Z}_j(t) \\ \bar{\mathbf{Z}}(\boldsymbol{\beta}, t) &= \frac{S^{(1)}(\boldsymbol{\beta}, t)}{S^{(0)}(\boldsymbol{\beta}, t)} \end{aligned}$$

For a given realization of the explanatory variables $\boldsymbol{\xi}$, the cumulative hazard function estimator at $\boldsymbol{\xi}$ is

$$\hat{\Lambda}_B(t, \boldsymbol{\xi}) = \mathrm{e}^{\hat{\boldsymbol{\beta}}' \boldsymbol{\xi}} \sum_{t_i \le t} \frac{d(t_i)}{S^{(0)}(\hat{\boldsymbol{\beta}}, t_i)}$$

with variance estimated by

$$\hat{\sigma}^2(\hat{\Lambda}_B(t, \boldsymbol{\xi})) = \mathrm{e}^{2\hat{\boldsymbol{\beta}}' \boldsymbol{\xi}} \sum_{t_i \le t} \frac{d(t_i)}{[S^{(0)}(\hat{\boldsymbol{\beta}}, t_i)]^2} + H(t, \boldsymbol{\xi})' [\mathcal{I}(\hat{\boldsymbol{\beta}})]^{-1} H(t, \boldsymbol{\xi})$$

where

$$H(t,\boldsymbol{\xi}) = \mathrm{e}^{\hat{\boldsymbol{\beta}}'\boldsymbol{\xi}} \sum_{t_i \le t} \frac{d(t_i)}{S^{(0)}(\hat{\boldsymbol{\beta}}, t_i)} \left(\bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, t_i) - \boldsymbol{\xi} \right)$$

For the marginal model, the variance estimator computation follows Spiekerman and Lin (1998).

The Breslow estimate of the survivor function for $\mathbf{Z} = \boldsymbol{\xi}$ is

$$\hat{S}_B(t,\boldsymbol{\xi}) = \exp(-\hat{\Lambda}_B(t,\boldsymbol{\xi}))$$

By the delta method, the standard error of $\hat{S}_B(t,\boldsymbol{\xi})$ is approximated by

$$\hat{\sigma}(\hat{S}_B(t,\boldsymbol{\xi})) = \hat{S}_B(t,\boldsymbol{\xi})\hat{\sigma}(\hat{\Lambda}_B(t,\boldsymbol{\xi}))$$

Fleming-Harrington Estimator

To select this estimator, specify the METHOD=FH option in the BASELINE statement or OUTPUT statement. With $Y_j(t)$ and $d(t)$ as defined in the section "Breslow Estimator" on page 7596 and for $1 \le k \le d(t)$, let

$$\begin{aligned}
S_E^{(0)}(\boldsymbol{\beta}, k, t) &= \sum_j Y_j(t) \left\{ 1 - \frac{k-1}{d(t)} \Delta_j(t) \right\} \mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_j(t)} \\
S_E^{(1)}(\boldsymbol{\beta}, k, t) &= \sum_j Y_j(t) \left\{ 1 - \frac{k-1}{d(t)} \Delta_j(t) \right\} \mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_j(t)} \mathbf{Z}_j(t) \\
\bar{\mathbf{Z}}_E \boldsymbol{\beta}, k, t) &= \frac{S_E^{(1)}(\boldsymbol{\beta}, k, t)}{S_E^{(0)}(\boldsymbol{\beta}, k, t)}
\end{aligned}$$

For a given realization of the explanatory variables, the Fleming-Harrington adjustment of the cumulative hazard function is

$$\hat{\Lambda}_F(t,\boldsymbol{\xi}) = \mathrm{e}^{\hat{\boldsymbol{\beta}}'\boldsymbol{\xi}} \sum_{t_i \le t} \left\{ \sum_{k=1}^{d(t_i)} \frac{1}{S_E^{(0)}(\hat{\boldsymbol{\beta}}, k, t_i)} \right\}$$

with variance estimated by

$$\hat{\sigma}^2(\hat{\Lambda}_F(t,\boldsymbol{\xi})) = \mathrm{e}^{2\hat{\boldsymbol{\beta}}'\boldsymbol{\xi}} \sum_{t_i \le t} \left\{ \sum_{k=1}^{d(t_i)} \frac{1}{[S_E^{(0)}(\hat{\boldsymbol{\beta}}, k, t_i)]^2} \right\} + H_E(t,\boldsymbol{\xi})'[\mathcal{I}(\hat{\boldsymbol{\beta}})]^{-1} H_E(t,\boldsymbol{\xi})$$

where

$$H_E(t,\boldsymbol{\xi}) = \mathrm{e}^{\hat{\boldsymbol{\beta}}'\boldsymbol{\xi}} \left\{ \left[\sum_{t_i \le t} \sum_{k=1}^{d(t_i)} \frac{1}{S_E^{(0)}(\hat{\boldsymbol{\beta}}, k, t_i)} \bar{\mathbf{Z}}_E(\hat{\boldsymbol{\beta}}, k, t_i) \right] - \hat{\Lambda}_F(t, \mathbf{0})\boldsymbol{\xi} \right\}$$

The Fleming-Harrington estimate of the survivor function for $\mathbf{Z} = \boldsymbol{\xi}$ is

$$\hat{S}_F(t,\boldsymbol{\xi}) = \exp(-\hat{\Lambda}_F(t,\boldsymbol{\xi}))$$

By the delta method, the standard error of $\hat{S}_B(t,\boldsymbol{\xi})$ is approximated by

$$\hat{\sigma}(\hat{S}_F(t,\boldsymbol{\xi})) = \hat{S}_F(t,\boldsymbol{\xi})\hat{\sigma}(\hat{\Lambda}_F(t,\boldsymbol{\xi}))$$

Product-Limit Estimator

To select this estimator, specify the METHOD=PL option in the BASELINE statement or OUTPUT statement. Let $\mathcal{D}_i$ denote the set of individuals that fail at t_i. Let $\mathcal{C}_i$ denote the set of individuals that are censored in the half-open interval $[t_i, t_{i+1})$, where $t_0 = 0$ and $t_{k+1} = \infty$. Let γ_l denote the censoring times in $[t_i, t_{i+1})$, where l ranges over $\mathcal{C}_i$.

The likelihood function for all individuals is given by

$$\mathcal{L} = \prod_{i=0}^{k} \left\{ \prod_{l \in \mathcal{D}_i} \left([S_0(t_i)]^{\exp(\mathbf{Z}_l'\boldsymbol{\beta})} - [S_0(t_i + 0)]^{\exp(\mathbf{Z}_l'\boldsymbol{\beta})} \right) \prod_{l \in \mathcal{C}_i} [S_0(\gamma_l + 0)]^{\exp(\mathbf{Z}_l'\boldsymbol{\beta})} \right\}$$

where $\mathcal{D}_0$ is empty. The likelihood $\mathcal{L}$ is maximized by taking $S_0(t) = S_0(t_i + 0)$ for $t_i < t \le t_{i+1}$ and allowing the probability mass to fall only on the observed event times $t_1, \ldots, t_k$. By considering a discrete model with hazard contribution $1 - \alpha_i$ at t_i, you take $S_0(t_i) = S_0(t_{i-1} + 0) = \prod_{j=0}^{i-1} \alpha_j$, where $\alpha_0 = 1$. Substitution into the likelihood function produces

$$\mathcal{L} = \prod_{i=0}^{k} \left\{ \prod_{j \in \mathcal{D}_i} \left(1 - \alpha_i^{\exp(\mathbf{Z}_j'\boldsymbol{\beta})} \right) \prod_{l \in \mathcal{R}_i - \mathcal{D}_i} \alpha_i^{\exp(\mathbf{z}_l'\boldsymbol{\beta})} \right\}$$

If you replace $\boldsymbol{\beta}$ with $\hat{\boldsymbol{\beta}}$ estimated from the partial likelihood function and then maximize with respect to $\alpha_1, \ldots, \alpha_k$, the maximum likelihood estimate $\hat{\alpha}_i$ of α_i becomes a solution of

$$\sum_{j \in \mathcal{D}_i} \frac{\exp(\mathbf{Z}_j'\hat{\boldsymbol{\beta}})}{1 - \hat{\alpha}_i^{\exp(\mathbf{Z}_j'\hat{\boldsymbol{\beta}})}} = \sum_{l \in \mathcal{R}_i} \exp(\mathbf{Z}_l'\hat{\boldsymbol{\beta}})$$

When only a single failure occurs at t_i, $\hat{\alpha}_i$ can be found explicitly. Otherwise, an iterative solution is obtained by the Newton method.

The baseline survival function is estimated by

$$\hat{S}_0(t) = \hat{S}_0(t_{i-1} + 0) = \prod_{j=0}^{i-1} \hat{\alpha}_j, t_{i-1} < t \le t_i$$

For a given realization of the explanatory variables $\boldsymbol{\xi}$, the product-limit estimate of the survival function at $\mathbf{Z} = \boldsymbol{\xi}$ is

$$\hat{S}_P(t, \boldsymbol{\xi}) = [\hat{S}_0(t)]^{\exp(\boldsymbol{\beta}'\boldsymbol{\xi})}$$

Approximating the variance of $-\log(S_P(t, \boldsymbol{\xi}))$ by the variance estimate of the Breslow estimator of the cumulative hazard function, the variance of the product-limit estimator at $\mathbf{Z} = \boldsymbol{\xi}$ is given by

$$\hat{\sigma}(\hat{S}_P(t, \boldsymbol{\xi})) = \hat{S}_P(t, \boldsymbol{\xi}) \hat{\sigma}(\hat{\Lambda}_B(t, \boldsymbol{\xi}))$$

Direct Adjusted Survival Curves

Consider the Breslow estimator of the survival function. For $j = 1, \ldots, n$, let $\boldsymbol{\xi}_j$ represent the covariate set of the jth patient. The direct adjusted survival curve averages the estimated survival curves for each patient:

$$\bar{S}(t) = \frac{1}{n} \sum_{j=1}^{n} \hat{S}(t, \boldsymbol{\xi}_j)$$

The variance of $\bar{S}(t)$ can be estimated by

$$\hat{\sigma}^2(\bar{S}(t)) = \frac{1}{n^2}\left(V^{(1)}t) + V^{(2)}(t)\right)$$

where

$$\begin{aligned} V^{(1)}(t) &= \left(\sum_{j=1}^{n} \mathrm{e}^{\hat{\boldsymbol{\beta}}'\boldsymbol{\xi}_j} \hat{S}(t, \boldsymbol{\xi}_j)\right)^2 \sum_{t_i \le t} \frac{d(t_i)}{[S^{(0)}(\hat{\boldsymbol{\beta}}, t_i)]^2} \\ V^{(2)}(t) &= \left(\sum_{j=1}^{n} \hat{S}(t, \boldsymbol{\xi}_j) H(t, \boldsymbol{\xi}_j)\right)' \left[\mathcal{I}(\hat{\boldsymbol{\beta}})\right]^{-1} \left(\sum_{j=1}^{n} \hat{S}(t, \boldsymbol{\xi}_j) H(t, \boldsymbol{\xi}_j)\right) \end{aligned}$$

Comparison of Direct Adjusted Probabilities of Two Strata

For a stratified Cox model, let k index the strata. For the jth patient, let $\hat{S}_k(t, \boldsymbol{\xi}_j)$ and $H_k(t, \boldsymbol{\xi}_j)$ be the estimated survival function and the $\mathbf{H}$ vector for the kth stratum. The direct adjusted survival curve for the kth stratum is

$$\bar{S}_k(t) = \frac{1}{n} \sum_{j=1}^{n} \hat{S}_k(t, \boldsymbol{\xi}_j)$$

The variance of $\bar{S}_1(t) - \bar{S}_2(t)$ can be estimated by

$$\hat{\sigma}^2(\bar{S}_1(t) - \bar{S}_2(t)) = \frac{1}{n^2}\left(U_1^{(1)}(t) + U_2^{(1)} + U_{12}^{(2)}(t)\right)$$

where

$$\begin{aligned} U_k^{(1)}(t) &= \left(\sum_{j=1}^{n} \mathrm{e}^{\boldsymbol{\beta}'\boldsymbol{\xi}_j} \hat{S}_k(t, \boldsymbol{\xi}_j)\right)^2 \sum_{t_i \le t} \frac{d(t_i)}{[S^{(0)}(\hat{\boldsymbol{\beta}}, t_i)]^2}, \quad k = 1, 2 \\ U_{12}^{(2)}(t) &= \left\{\sum_{j=1}^{n} \left[\hat{S}_1(t, \boldsymbol{\xi}_j) H_1(t, \boldsymbol{\xi}_j) - \hat{S}_2(t, \boldsymbol{\xi}_j) H_2(t, \boldsymbol{\xi}_j)\right]\right\}' \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \\ &\quad \left\{\sum_{j=1}^{n} \left[\hat{S}_1(t, \boldsymbol{\xi}_j) H_2(t, \boldsymbol{\xi}_j) - \hat{S}_1(t, \boldsymbol{\xi}_j) H(_2 t, \boldsymbol{\xi}_j)\right]\right\} \end{aligned}$$

Comparison of Direct Adjusted Survival Probabilities of Two Treatments

For $j = 1, \ldots, n$, let $\boldsymbol{\xi}_{jk}$ represent the covariate set of the jth patient with the kth treatment, $k = 1, 2$. The direct adjusted survival curve for the kth treatment is

$$\bar{S}_k(t) = \frac{1}{n} \sum_{i=j}^{n} \hat{S}(t, \boldsymbol{\xi}_{jk})$$

The variance of $\bar{S}_1(t) - \bar{S}_2(t)$ can be estimated by

$$\hat{\sigma}^2 \left(\bar{S}_1(t) - \bar{S}_2(t)\right) = \frac{1}{n^2}\left(V_{12}^{(1)}(t) + V_{12}^{(2)}(t)\right)$$

where

$$\begin{aligned}
V_{12}^{(1)}(t) &= \left\{\sum_{j=1}^{n}\left[\mathrm{e}^{\boldsymbol{\beta}'\boldsymbol{\xi}_{1j}}\hat{S}(t, \boldsymbol{\xi}_{1j}) - \mathrm{e}^{\boldsymbol{\beta}'\boldsymbol{\xi}_{2j}}\hat{S}(t, \boldsymbol{\xi}_{2j})\right]\right\}^2 \sum_{t_i \le t} \frac{d(t_i)}{[S^{(0)}(\hat{\boldsymbol{\beta}}, t_i)]^2} \\
V_{12}^{(2)}(t) &= \left\{\sum_{j=1}^{n}\left[\hat{S}(t, \boldsymbol{\xi}_{1j})H(t, \boldsymbol{\xi}_{1j}) - \hat{S}(t, \boldsymbol{\xi}_{2j})H(t, \boldsymbol{\xi}_{2j})\right]\right\}' \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \\
&\quad \left\{\sum_{j=1}^{n}\left[\hat{S}(t, \boldsymbol{\xi}_{1j})H(t, \boldsymbol{\xi}_{1j}) - \hat{S}(t, \boldsymbol{\xi}_{2j})H(t, \boldsymbol{\xi}_{2j})\right]\right\}
\end{aligned}$$

Confidence Intervals for the Survivor Function

When the computation of confidence limits for the survivor function $S(t)$ is based on the asymptotic normality of the survival estimator $\hat{S}(t)$—which can be the Breslow estimator $\hat{S}_B(t)$, the Fleming-Harrington estimator $\hat{S}_F(t)$, or the product-limit estimator $\hat{S}_P(t)$—the approximate confidence interval might include impossible values outside the range [0,1] at extreme values of t. This problem can be avoided by applying the asymptotic normality to a transformation of $S(t)$ for which the range is unrestricted. In addition, certain transformed confidence intervals for $S(t)$ perform better than the usual linear confidence intervals (Borgan and Liestøl 1990). The CLTYPE= option in the BASELINE statement enables you to choose one of the following transformations: the log-log function, the log function, and the linear function.

Let g be the transformation that is being applied to the survivor function $S(t)$. By the delta method, the standard error of $g(\hat{S}(t))$ is estimated by

$$\tau(t) = \hat{\sigma}\left[g(\hat{S}(t))\right] = g'\left(\hat{S}(t)\right)\hat{\sigma}[\hat{S}(t)]$$

where g' is the first derivative of the function g. The 100(1–α)% confidence interval for $S(t)$ is given by

$$g^{-1}\left\{g[\hat{S}(t)] \pm z_{\frac{\alpha}{2}} g'[\hat{S}(t)]\hat{\sigma}[\hat{S}(t)]\right\}$$

where g^{-1} is the inverse function of g. The choices for the transformation g are as follows:

- CLTYPE=NORMAL specifies linear transformation, which is the same as having no transformation in which g is the identity. The 100(1–α)% confidence interval for $S(t)$ is given by

$$\hat{S}(t) - z_{\frac{\alpha}{2}}\hat{\sigma}\left[\hat{S}(t)\right] \le S(t) \le \hat{S}(t) + z_{\frac{\alpha}{2}}\hat{\sigma}\left[\hat{S}(t)\right]$$

- CLTYPE=LOG specifies log transformation. The estimated variance of $\log(\hat{S}(t))$ is $\hat{\tau}^2(t) = \frac{\hat{\sigma}^2(\hat{S}(t))}{\hat{S}^2(t)}$. The 100(1–$\alpha$)% confidence interval for $S(t)$ is given by

$$\hat{S}(t)\exp\left(-z_{\frac{\alpha}{2}}\hat{\tau}(t)\right) \le S(t) \le \hat{S}(t)\exp\left(z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)$$

- CLTYPE=LOGLOG specifies log-log transformation. The estimated variance of $\log(-\log(\hat{S}(t))$ is $\hat{\tau}^2(t) = \frac{\hat{\sigma}^2[\hat{S}(t)]}{[\hat{S}(t)\log(\hat{S}(t))]^2}$. The 100(1–$\alpha$)% confidence interval for $S(t)$ is given by

$$\left[\hat{S}(t)\right]^{\exp\left(z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)} \le S(t) \le \left[\hat{S}(t)\right]^{\exp\left(-z_{\frac{\alpha}{2}}\hat{\tau}(t)\right)}$$

Caution about Using Survival Data with Left Truncation

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/phrcaut.sas.)

The product-limit estimator is used in a number of instances in the PHREG procedure, such as to transform the time values in the ZPH option in the PROC PHREG statement. The product-limit estimator is also used to construct the weights in the inverse probability of censoring weighting (IPCW) techniques, which are adapted to fit the proportional subdistribution model of Fine and Gray (1999) for competing-risks data and to assess the predictive accuracy of a model (Schemper and Henderson 2000). Although the product-limit estimator is the gold standard for estimating the survivor function of right-censored data, it might not be meaningful for right-censored data with left-truncation, as illustrated by Example 4.3 in Klein and Moeschberger (2003). In their example, 94 men and 365 women passed through the Channing House Retirement Center between January 1964 and July 1975. The outcome is the time to death, using the natural metric of age (in months).

The following statements create the data set Channing, which contains the following variables:

- Gender: female or male
- Age_entry: age at entry, in months
- Age_exit: age at exit (death or last follow-up), in months
- Death: death indicator, with the value 1 for death and 0 for censoring

```
data Channing;
   input Gender$ Age_entry Age_exit Death @@;
   datalines;
Female  1042 1172  1  Female   921 1040  1  Female   885 1003  1
Female   901 1018  1  Female   808  932  1  Female   915 1004  1
Female   901 1023  1  Female   852  908  1  Female   828  868  1
```

```
Female   968  990  1  Female   936 1033  1  Female   977 1056  1
Female   929  999  1  Female   936 1064  1  Female  1016 1122  1
Female   910 1020  1  Female  1140 1200  1  Female  1015 1056  1
Female   850  940  1  Female   895  996  1  Female   854  969  1

   ... more lines ...

  Male   751  777  1    Male   906  966  1    Male   835  907  1
  Male   946 1031  1    Male   759  781  1    Male   909  914  0
  Male   962  998  1    Male   984 1022  1    Male   891  932  1
  Male   835  898  1    Male  1039 1060  1    Male  1010 1044  1
;
```

The following statements use the PHREG procedure to save the product-limit estimate of the survivor function for each gender in the data set Outs. For each gender, the number of subjects at risk and the number of deaths at each death time are captured in the data set Atrisk. By merging these two data sets, Outs and Atrisk, you can conveniently display side by side the number of subjects at risk, the number of deaths, and the product-limit survival estimate at each death time.

```
ods graphics on;
proc phreg data=Channing plots(overlay=row)=survival atrisk;
   model Age_exit*Death(0)= /entrytime=Age_entry;
   strata Gender;
   baseline out=Outs survival=Probability / method=pl;
   ods output RiskSetInfo=Atrisk;
run;

data Outs;
   set Outs;
   if Gender="Female" then StratumNumber=1;
   else                    StratumNumber=2;
run;
data Outs;
   merge atrisk outs;
   by StratumNumber Age_exit;
run;

proc print data=Outs;
   id Gender;
   var Age_exit Atrisk Event Probability;
run;
```

Figure 92.18 displays two product-limit survival curves, one for women and one for men. The survival probabilities are tabulated in Figure 92.19 for women and in Figure 92.20 for men. Although the survival curve for women does not appear unusual, the survival curve for men looks odd, because the curve drops to 0 at 781 months even though the majority of men survive beyond 781 months. At 781 months, the risk set consists of a single time to death, rendering the product-limit estimate as 0 at 781 months and thereafter. The product-limit curve for men for these data is meaningless. Klein and Moeschberger (2003) suggest using only those observations in which the value of Age_exit exceeds 781 months.

Figure 92.18 Product-Limit Estimates for Women and Men

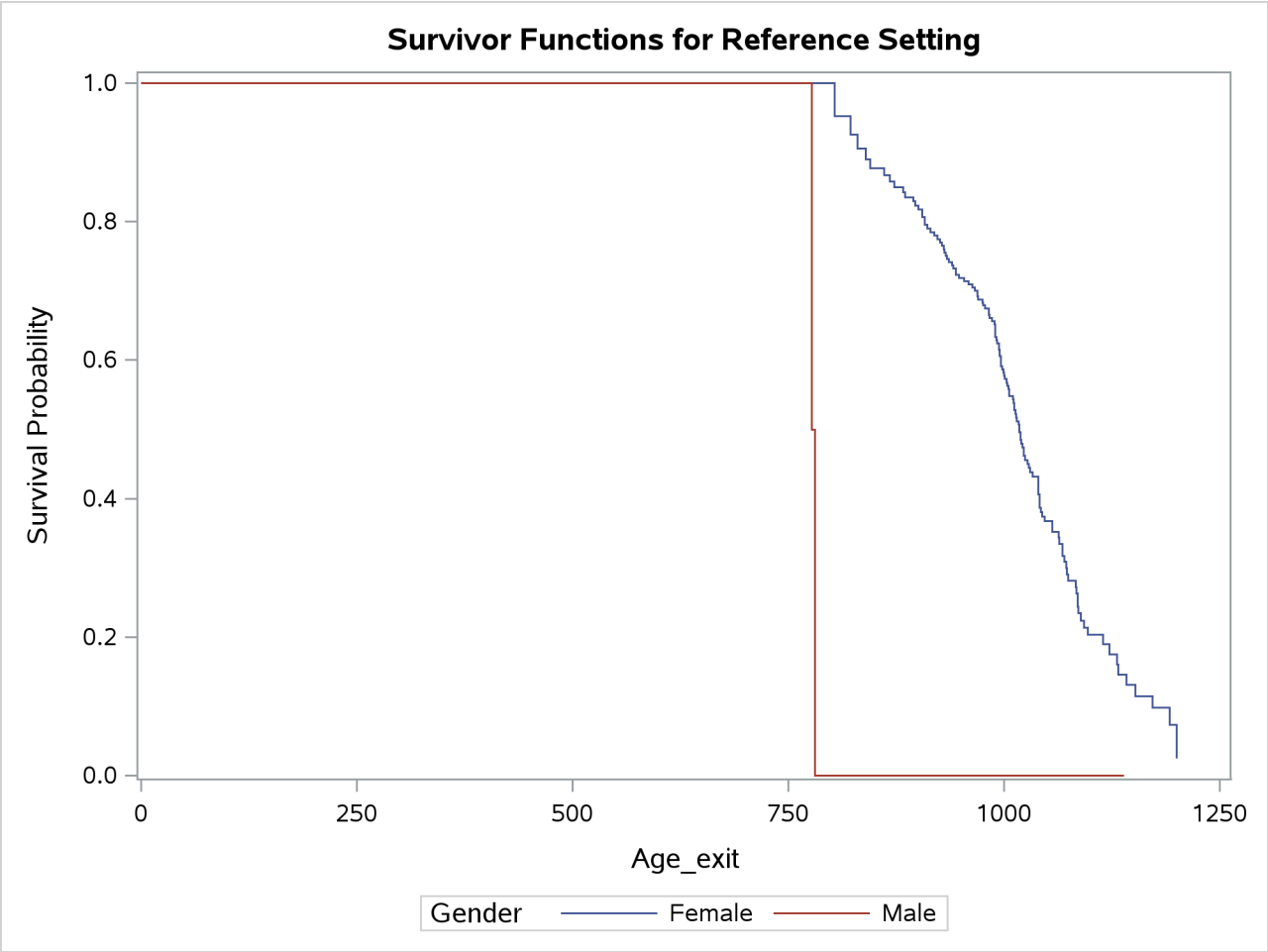

Figure 92.19 Product-Limit Survival Probabilities for Women

Gender	Age_exit	Atrisk	Event	Probability
Female	0	.	.	1.00000
Female	804	21	1	0.95238
Female	822	36	1	0.92593
Female	830	46	1	0.90580
Female	840	58	1	0.89018
Female	845	66	1	0.87669
Female	861	89	1	0.86684
	.	.	.	.
	.	.	.	.
	.	.	.	.
Female	1152	8	1	0.11493
Female	1172	7	1	0.09852
Female	1192	4	1	0.07389
Female	1200	3	2	0.02463

Figure 92.20 Product-Limit Survival Probabilities for Men

Gender	Age_exit	Atrisk	Event	Probability
Male	0	.	.	1.0
Male	777	2	1	0.5
Male	781	1	1	0.0
Male	869	24	1	0.0
Male	872	25	1	0.0
Male	876	25	1	0.0
Male	893	33	1	0.0
	.	.	.	.
	.	.	.	.
	.	.	.	.
Male	1085	10	1	0.0
Male	1094	8	2	0.0
Male	1128	3	1	0.0
Male	1139	2	1	0.0

PROC PHREG currently makes no attempt to circumvent the problem of the invalid product-limit estimator for left-truncated data.

Effect Selection Methods

Five effect selection methods are available. The simplest method (and the default) is SELECTION=NONE, for which PROC PHREG fits the complete model as specified in the MODEL statement. The other four methods are FORWARD for forward selection, BACKWARD for backward elimination, STEPWISE for stepwise selection, and SCORE for best subsets selection. These methods are specified with the SELECTION= option in the MODEL statement and are based on the score test or Wald test as described in the section "Type 3 Tests and Joint Tests" on page 7570.

SELECTION=FORWARD begins with the effects that are forced into the model. These are the first n effects in the MODEL statement, where n is the number specified by the START= or INCLUDE= option in the MODEL statement (n is zero by default). The forward selection process sequentially adds the effect that most improves the fit. The statistic that is used to determine whether to add an effect is the significant level of the score test (Kalbfleisch and Prentice 1980, Section 3.4.1) that reflects the effect's contribution to the model if it is included. At each step, the effect that is most significant is added. The process is repeated until none of the remaining effects meet the specified level for entry or until the STOP= value is reached.

SELECTION=BACKWARD starts with the full model, which includes all the explanatory effects unless the START= option is specified. In that case, only the first n effects in the MODEL statement are included in the initial model, where n is the number specified by the START= option. The backward elimination process sequentially drops the effects that have the least contribution to the fit. The statistic that is used to determine whether to drop an effect is the significance level of the Wald test (Kalbfleisch and Prentice 1980, Section 3.4.2). At any step, the least significant effect is dropped and the process continues until all effects that remain in the model are significant at the SLSTAY= level or until the STOP= value is reached.

SELECTION=STEPWISE modifies the forward selection technique by allowing effects already in the model to be removed. See Example 92.1 for an illustration of the stepwise selection process. The same entry and removal significance levels for the forward selection and backward elimination methods are used to assess

contributions of effects as they are added to or removed from a model. If, at a step of the stepwise method, any effect in the model is not significant at the SLSTAY= level, then the least significant of these effects is removed from the model and the algorithm proceeds to the next step. This ensures that no effect can be added to a model while some effect currently in the model is not deemed significant. Another effect can be added to the model only after all necessary deletions have been accomplished. In this case the effect whose addition is the most significant is added to the model and the algorithm proceeds to the next step. The stepwise process ends when none of the effects outside the model is significant at the SLENTRY= level and every effect in the model is significant at the SLSTAY= level. In some cases, neither of these two conditions for stopping is met and the sequence of models cycles. In this case, the stepwise method terminates at the end of the cycle.

SELECTION=SCORE uses the branch-and-bound algorithm of Furnival and Wilson (1974) to find a specified number of models with the highest score (chi-square) statistic for all possible model sizes, from 1, 2, or 3 variables, and so on, up to the single model that contains all of the explanatory variables. The number of models displayed for each model size is controlled by the BEST= option. You can use the START= option to impose a minimum model size, and you can use the STOP= option to impose a maximum model size. For instance, with BEST=3, START=2, and STOP=5, the SCORE selection method displays the best three models (that is, the three models with the highest score chi-squares) that contain 2, 3, 4, and 5 variables. One of the limitations of the branch-and-bound algorithm is that it works only when each explanatory effect contains exactly one parameter—the SELECTION=SCORE option is not allowed when an explanatory effect in the MODEL statement contains a CLASS variable.

The SEQUENTIAL and STOPRES options can alter the default criteria for adding variables to or removing variables from the model when they are used with the FORWARD, BACKWARD, or STEPWISE selection method.

Assessment of the Proportional Hazards Model

The proportional hazards model specifies that the hazard function for the failure time T associated with a $p \times 1$ column covariate vector $\mathbf{Z}$ takes the form

$$\lambda(t;\mathbf{Z}) = \lambda_0(t)\mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}}$$

where $\lambda_0(.)$ is an unspecified baseline hazard function and $\boldsymbol{\beta}$ is a $p \times 1$ column vector of regression parameters. Lin, Wei, and Ying (1993) present graphical and numerical methods for model assessment based on the cumulative sums of martingale residuals and their transforms over certain coordinates (such as covariate values or follow-up times). The distributions of these stochastic processes under the assumed model can be approximated by the distributions of certain zero-mean Gaussian processes whose realizations can be generated by simulation. Each observed residual pattern can then be compared, both graphically and numerically, with a number of realizations from the null distribution. Such comparisons enable you to assess objectively whether the observed residual pattern reflects anything beyond random fluctuation. These procedures are useful in determining appropriate functional forms of covariates and assessing the proportional hazards assumption. You use the ASSESS statement to carry out these model-checking procedures.

For a sample of n subjects, let $(X_i, \Delta_i, \mathbf{Z}_i)$ be the data of the ith subject; that is, X_i represents the observed failure time, Δ_i has a value of 1 if X_i is an uncensored time and 0 otherwise, and $\mathbf{Z}_i = (Z_{1i}, \ldots, Z_{pi})'$ is a p-vector of covariates. Let $N_i(t) = \Delta_i I(X_i \le t)$ and $Y_i(t) = I(X_i \ge t)$. Let

$$S^{(0)}(\boldsymbol{\beta}, t) = \sum_{i=1}^{n} Y_i(t)\mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_i} \quad \text{and} \quad \mathbf{Z}(\boldsymbol{\beta}, t) = \frac{\sum_{i=1}^{n} Y_i(t)\mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_i}\mathbf{Z}_i}{S^{(0)}(\boldsymbol{\beta}, t)}$$

Let $\hat{\boldsymbol{\beta}}$ be the maximum partial likelihood estimate of $\boldsymbol{\beta}$, and let $\mathcal{I}(\hat{\boldsymbol{\beta}})$ be the observed information matrix.

The martingale residuals are defined as

$$\hat{M}_i(t) = N_i(t) - \int_0^t Y_i(u)\mathrm{e}^{\hat{\boldsymbol{\beta}}'\mathbf{Z}_i}\, d\,\hat{\Lambda}_0(u), i = 1, \ldots, n$$

where $\hat{\Lambda}_0(t) = \int_0^t \frac{\sum_{i=1}^n dN_i(u)}{S^{(0)}(\hat{\boldsymbol{\beta}},u)}$.

The empirical score process $\mathbf{U}(\hat{\boldsymbol{\beta}}, t) = (U_1(\hat{\boldsymbol{\beta}}, t), \ldots, U_p(\hat{\boldsymbol{\beta}}, t))'$ is a transform of the martingale residuals:

$$\mathbf{U}(\hat{\boldsymbol{\beta}}, t) = \sum_{i=1}^n \mathbf{Z}_i \hat{M}_i(t)$$

Checking the Functional Form of a Covariate

To check the functional form of the *j*th covariate, consider the partial-sum process of $\hat{M}_i = \hat{M}_i(\infty)$:

$$W_j(z) = \sum_{i=1}^n I(Z_{ji} \le z)\hat{M}_i$$

Under that null hypothesis that the model holds, $W_j(z)$ can be approximated by the zero-mean Gaussian process

$$\begin{aligned}\hat{W}_j(z) \quad = \quad & \sum_{l=1}^n \Delta_l \left\{ I(Z_{jl} \le z) - \frac{\sum_{i=1}^n Y_i(X_l)\mathrm{e}^{\boldsymbol{\beta}'\mathbf{Z}_i} I(Z_{ij} \le z)}{S^{(0)}(\hat{\boldsymbol{\beta}}, X_l)} \right\} G_l - \\ & \sum_{k=1}^n \int_0^\infty Y_k(s)\mathrm{e}^{\hat{\boldsymbol{\beta}}'\mathbf{Z}_k} I(Z_{jk} \le z)[\mathbf{Z}_k - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, s)]' d\,\hat{\Lambda}_0(s) \\ & \times \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \sum_{l=1}^n \Delta_l [\mathbf{Z}_l - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, X_l)] G_l\end{aligned}$$

where $(G_1, \ldots, G_n)$ are independent standard normal variables that are independent of $(X_i, \Delta_i, \mathbf{Z}_i)$, $i = 1, \ldots, n$.

You can assess the functional form of the *j*th covariate by plotting a small number of realizations (the default is 20) of $\hat{W}_j(z)$ on the same graph as the observed $W_j(z)$ and visually comparing them to see how typical the observed pattern of $W_j(z)$ is of the null distribution samples. You can supplement the graphical inspection method with a Kolmogorov-type supremum test. Let s_j be the observed value of $S_j = \sup_z |W_j(z)|$ and let $\hat{S}_j = \sup_z |\hat{W}_j(z)|$. The *p*-value $\Pr(S_j \ge s_j)$ is approximated by $\Pr(\hat{S}_j \ge s_j)$, which in turn is approximated by generating a large number of realizations (1000 is the default) of $\hat{W}_j(.)$.

Checking the Proportional Hazards Assumption

Consider the standardized empirical score process for the jth component of $\mathbf{Z}$

$$U_j^*(t) = [\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})_{jj}]^{\frac{1}{2}} U_j(\hat{\boldsymbol{\beta}}, t),$$

Under the null hypothesis that the model holds, $U_j^*(t)$ can be approximated by

$$\begin{aligned}\hat{U}_j^*(t) \quad = \quad & [\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})_{jj}]^{\frac{1}{2}} \Bigg\{ \sum_{l=1}^{n} I(X_l \le t)\Delta_l [Z_{jl} - \bar{Z}_j(\hat{\boldsymbol{\beta}}, t)] G_l - \\ & \sum_{k=1}^{n} \int_0^t Y_k(s) \mathrm{e}^{\hat{\boldsymbol{\beta}}' \mathbf{Z}_k} Z_{jk} [\mathbf{Z}_k - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, s)]' d\,\hat{\Lambda}_0(s) \\ & \times \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \sum_{l=1}^{n} \Delta_l [\mathbf{Z}_l - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, X_l)] G_l \Bigg\}\end{aligned}$$

where $\bar{Z}_j(\hat{\boldsymbol{\beta}}, t)$ is the jth component of $\bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, t)$, and $(G_1, \ldots, G_n)$ are independent standard normal variables that are independent of $(X_i, \Delta_i, \mathbf{Z}_i, (i = 1, \ldots, n)$.

You can assess the proportional hazards assumption for the jth covariate by plotting a few realizations of $\hat{U}_j^*(t)$ on the same graph as the observed $U_j^*(t)$ and visually comparing them to see how typical the observed pattern of $U_j^*(t)$ is of the null distribution samples. Again you can supplement the graphical inspection method with a Kolmogorov-type supremum test. Let s_j^* be the observed value of $S_j^* = \sup_t |U_j^*(t)|$ and let $\hat{S}_j^* = \sup_t |\hat{U}_j^*(t)|$. The p-value $\Pr[S_j^* \ge s_j^*]$ is approximated by $\Pr[\hat{S}_j^* \ge s_j^*]$, which in turn is approximated by generating a large number of realizations (1000 is the default) of $\hat{U}_j^*(.)$.

The Penalized Partial Likelihood Approach for Fitting Frailty Models

Let $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_s)'$ be the vector of random components for the s clusters.

Gamma Frailty Model

Assume each e^{γ_i} has an independent and identically distributed gamma distribution with mean 1 and a common unknown variance θ; that is, e^{γ_i} is iid $G\left(\frac{1}{\theta}, \frac{1}{\theta}\right)$. The penalty function is

$$-\frac{1}{\theta} \sum_{i=1}^{s} \left(\gamma_i - \mathrm{e}^{\gamma_i} \right)$$

plus a function of θ. The penalized partial log likelihood is given by

$$l_p(\boldsymbol{\beta}, \boldsymbol{\gamma}, \theta) = l_{\text{partial}}(\boldsymbol{\beta}, \boldsymbol{\gamma}) + \frac{1}{\theta} \sum_{i=1}^{s} \left(\gamma_i - \mathrm{e}^{\gamma_i} \right)$$

where $l_{\text{partial}}(\boldsymbol{\beta}, \boldsymbol{\gamma})$ is the log of any of the partial likelihoods in the sections "Partial Likelihood Function for the Cox Model" on page 7550 and "The Multiplicative Hazards Model" on page 7553.

The profile marginal log-likelihood of this shared frailty model (Therneau and Grambsch 2000, pp. 257–258) is

$$l_m(\theta) = l_p(\hat{\boldsymbol{\beta}}(\theta), \hat{\boldsymbol{\gamma}}(\theta), \theta) + \sum_{i=1}^{s} \left\{ \theta^{-1} - (\theta^{-1} + d_i) \log\left[\theta^{-1} + d_i\right] + \theta^{-1} \log(\theta^{-1}) + \log\left[\frac{\Gamma(\theta^{-1} + d_i)}{\Gamma(\theta^{-1})}\right] \right\}$$

where d_i is the number of events in the ith cluster.

The maximization of this approximate likelihood is a doubly iterative process that alternates between the following two steps:

- For a provisional value of θ, the best linear unbiased predictors (BLUP) of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are computed by maximizing the penalized partial log-likelihood $l_p(\boldsymbol{\beta}, \boldsymbol{\gamma}, \theta)$. The marginal likelihood is evaluated. This constitutes the inner loop.
- A new value of θ is obtained by the golden section search based on the marginal likelihood of all the previous iterations. This constitutes the outer loop.

The outer loop is iterated until the bracketing interval of θ is small.

Lognormal Frailty Model

With each γ_i having a zero-mean normal distribution and a common variance θ, the penalty function is

$$\frac{1}{2\theta}\boldsymbol{\gamma}'\boldsymbol{\gamma}$$

plus a function of θ. The penalized partial log likelihood is given by

$$l_p(\boldsymbol{\beta}, \boldsymbol{\gamma}, \theta) = l_{\text{partial}}(\boldsymbol{\beta}, \boldsymbol{\gamma}) - \frac{1}{2\theta}\boldsymbol{\gamma}'\boldsymbol{\gamma}$$

where $l_{\text{partial}}(\boldsymbol{\beta}, \boldsymbol{\gamma})$ is the log of any of the partial likelihoods in the sections "Partial Likelihood Function for the Cox Model" on page 7550 and "The Multiplicative Hazards Model" on page 7553.

For a given θ, let $\mathbf{H}$ be the negative Hessian of the penalized partial log likelihood $l_p(\boldsymbol{\beta}, \boldsymbol{\gamma}, \theta)$; that is,

$$\mathbf{H} = \mathbf{H}(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \begin{bmatrix} \mathbf{H}_{11} & \mathbf{H}_{12} \\ \mathbf{H}_{21} & \mathbf{H}_{22} \end{bmatrix}$$

where $\mathbf{H}_{11} = -\frac{\partial^2 l_p(\boldsymbol{\beta},\boldsymbol{\gamma},\theta)}{\partial \boldsymbol{\beta}^2}$, $\mathbf{H}_{12} = \mathbf{H}'_{21} = -\frac{\partial^2 l_p(\boldsymbol{\beta},\boldsymbol{\gamma},\theta)}{\partial \boldsymbol{\beta}\partial \boldsymbol{\gamma}}$, and $\mathbf{H}_{22} = -\frac{\partial^2 l_p(\boldsymbol{\beta},\boldsymbol{\gamma},\theta)}{\partial \boldsymbol{\gamma}^2}$.

The marginal log likelihood of this shared frailty model is

$$l_m(\boldsymbol{\beta}, \theta) = -\frac{1}{2}\log(\theta^s) + \log\left[\int e^{l_p(\boldsymbol{\beta},\boldsymbol{\gamma},\theta)} d\boldsymbol{\gamma}\right]$$

Using a Laplace approximation to the integral as in Breslow and Clayton (1993), an approximate marginal log likelihood (Ripatti and Palmgren 2000; Therneau and Grambsch 2000) is given by

$$l_m(\boldsymbol{\beta}, \theta) \approx -\frac{1}{2}\log(\theta^s) - \frac{1}{2}\log(|\mathbf{H}_{22}(\boldsymbol{\beta}, \tilde{\boldsymbol{\gamma}}, \theta)|) - l_p(\boldsymbol{\beta}, \tilde{\boldsymbol{\gamma}}, \theta)$$

The maximization of this approximate likelihood is a doubly iterative process that alternates between the following two steps:

- For a provisional value of θ, PROC PHREG computes the best linear unbiased predictors (BLUP) of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ by maximizing the penalized partial log likelihood $l_p(\boldsymbol{\beta}, \boldsymbol{\gamma}, \theta)$. This constitutes the inner loop.
- For $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ fixed at the BLUP values, PROC PHREG estimates θ by maximizing the approximate marginal likelihood $l_m(\boldsymbol{\beta}, \theta)$. This constitutes the outer loop.

The outer loop is iterated until the difference between two successive estimates of θ is small.

The ML estimate of θ is

$$\hat{\theta} = \frac{\hat{\boldsymbol{\gamma}}'\hat{\boldsymbol{\gamma}} + \mathrm{trace}(\mathbf{H}_{22}^{-1})}{s}$$

The variance for $\hat{\theta}$ is

$$\mathrm{var}(\hat{\theta}) = 2\hat{\theta}\left[s + \frac{1}{\hat{\theta}^2}\mathrm{trace}(\mathbf{H}_{22}^{-1}\mathbf{H}_{22}^{-1}) - \frac{2}{\hat{\theta}}\mathrm{trace}(\mathbf{H}_{22}^{-1})\right]^{-1}$$

The REML estimation of θ is obtained by replacing $(\mathbf{H}_{22})^{-1}$ by $(\mathbf{H}^{-1})_{22}$.

The inverse of the final $\mathbf{H}$ matrix is used as the variance estimate of $(\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\gamma}})'$.

The final BLUP estimates of the random components $\gamma_1, \ldots, \gamma_s$ can be displayed by using the SOLUTION option in the RANDOM statement. Also displayed are estimates of the lognormal frailties, which are the exponentiated estimates of the BLUP estimates.

Wald-Type Tests for Penalized Models

Let $\mathbf{I}$ be the negative Hessian of the partial log likelihood $l_{\mathrm{partial}}(\boldsymbol{\beta}, \boldsymbol{\gamma})$:

$$\mathbf{I} = \begin{bmatrix} \mathbf{I}_{11} & \mathbf{I}_{12} \\ \mathbf{I}_{21} & \mathbf{I}_{22} \end{bmatrix}$$

where $\mathbf{I}_{11} = -\frac{\partial^2 l_{\mathrm{partial}}(\boldsymbol{\beta},\boldsymbol{\gamma})}{\partial \boldsymbol{\beta}^2}$, $\mathbf{I}_{12} = \mathbf{I}_{21}' = -\frac{\partial^2 l_{\mathrm{partial}}(\boldsymbol{\beta},\boldsymbol{\gamma})}{\partial \boldsymbol{\beta}\partial \boldsymbol{\gamma}}$, and $\mathbf{I}_{22} = -\frac{\partial^2 l_{\mathrm{partial}}(\boldsymbol{\beta},\boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}^2}$. Write $\boldsymbol{\tau}' = (\boldsymbol{\beta}', \boldsymbol{\gamma}')'$. The Wald-type chi-square statistic for testing $H_0 : \mathbf{C}\boldsymbol{\tau} = \mathbf{0}$ is

$$(\mathbf{C}\hat{\boldsymbol{\tau}})'(\mathbf{C}\mathbf{H}^{-1}\mathbf{C}')^{-1}(\mathbf{C}\hat{\boldsymbol{\tau}})$$

Let $\mathbf{H}$ be the negative Hessian of the penalized partial log likelihood $l_p(\boldsymbol{\beta}, \boldsymbol{\gamma}, \theta)$ at the ML estimate θ; that is, $\mathbf{H} = \frac{\partial^2}{\partial \boldsymbol{\beta}\partial \boldsymbol{\gamma}} l_p(\boldsymbol{\beta}, \boldsymbol{\gamma}, \hat{\theta})$. Let $\mathbf{V} = \mathbf{H}^{-1}\mathbf{I}\mathbf{H}^{-1}$. Gray (1992) recommends the following generalized degrees of freedom for the Wald test:

$$\mathrm{DF} = \mathrm{trace}[(\mathbf{C}\mathbf{H}^{-1}\mathbf{C}')^{-1}\mathbf{C}\mathbf{V}\mathbf{C}')]$$

See Therneau and Grambsch (2000, Section 5.8) for a discussion of this Wald-type test.

PROC PHREG uses the label “Adjusted DF” to represent this generalized degrees of freedom in the output.

Specifics for Bayesian Analysis

To request a Bayesian analysis, you specify the BAYES statement in addition to the PROC PHREG statement and the MODEL statement. You include a CLASS statement if you have effects that involve categorical variables. The FREQ or WEIGHT statement can be included if you have a frequency or weight variable, respectively, in the input data. You can use the STRATA statement to carry out a stratified analysis for the Cox model, but it is not allowed in the proportional hazards spline model or the piecewise constant baseline hazard model. Programming statements can be used to create time-dependent covariates for the Cox model, but they are not allowed in the proportional hazards spline model or the piecewise constant baseline hazard model. You can use the counting process style of input or the ENTRY= option in the MODEL statement to specify left truncation of failure times. The HAZARDRATIO statement enables you to request a hazard ratio analysis based on the posterior samples. The ASSESS, CONTRAST, ID, OUTPUT, and TEST statements, if specified, are ignored. Also ignored are the COVM and COVS options in the PROC PHREG statement and the following options in the MODEL statement: BEST=, CORRB, COVB, DETAILS, HIERARCHY=, INCLUDE=, MAXSTEP=, NOFIT, PLCONV=, SELECTION=, SEQUENTIAL, SLENTRY=, and SLSTAY=.

Proportional Hazards Spline Model

The proportional hazards spline model (Royston and Parmar 2002) with fixed covariate vector $\mathbf{x}$ has a cumulative hazard function at time t,

$$H(t;\mathbf{x}|\boldsymbol{\gamma}) = \mathrm{e}^{s(\log(t),\boldsymbol{\gamma})+\boldsymbol{\beta}'\mathbf{x}}$$

where $\boldsymbol{\beta}$ is the vector of regression coefficients and $s(\log(t),\boldsymbol{\gamma})$ is a cubic spline function as described for the SPLINE option in the BAYES statement. The number of knots is equal to one plus the degrees of freedom (which you specify in the DF= option). The knots are determined by the sequence of the distinct event times of the data. PROC PHREG places the terminal knots at the minimum and maximum of the sequence and selects the m internal knots as the kth percentiles of the sequence, where $k = 100\frac{1}{m+1}, \ldots, 100\frac{m}{m+1}$. For example, if you specify DF=4, the three internal knots are the 25th, 50th, and 75th percentiles of the distinct event times.

Let $\{(t_i, \mathbf{x}_i, \delta_i), i = 1, 2, \ldots, n\}$ be the observed data. The likelihood function of the proportional hazards spline model is given by

$$L_{RP}(\boldsymbol{\gamma},\beta) = \prod_i \left\{ \frac{1}{t_i}\frac{ds(y_i,\boldsymbol{\gamma})}{dy_i} \right\}^{\delta_i} \mathrm{e}^{\eta_i - \mathrm{e}^{\eta_i}}$$

where $y_i = \log(t_i)$ and $\eta_i = s(y_i,\boldsymbol{\gamma}) + \boldsymbol{\beta}'\mathbf{x_i}$. Note that

$$\frac{ds(y;\boldsymbol{\gamma})}{dy} = \gamma_1 + \sum_{j=2}^{m} \gamma_j[3(y-k_j)_+^2 - 3\lambda_j(x-k_{\min})_+^2 - 3(1-\lambda_j)(x-k_{\max})_+^2]$$

Piecewise Constant Baseline Hazard Model

Let $\{(t_i, \mathbf{x}_i, \delta_i), i = 1, 2, \ldots, n\}$ be the observed data. Let $a_0 = 0 < a_1 < \ldots < a_{J-1} < a_J = \infty$ be a partition of the time axis.

Hazards in Original Scale

The hazard function for subject i is

$$h(t|\mathbf{x}_i; \boldsymbol{\theta}) = h_0(t) \exp(\boldsymbol{\beta}'\mathbf{x}_i)$$

where

$$h_0(t) = \lambda_j \text{ if } a_{j-1} \le t < a_j, j = 1, \ldots, J$$

The baseline cumulative hazard function is

$$H_0(t) = \sum_{j=1}^{J} \lambda_j \Delta_j(t)$$

where

$$\Delta_j(t) = \begin{cases} 0 & t < a_{j-1} \\ t - a_{j-1} & a_{j-1} \le t < a_j \\ a_j - a_{j-1} & t \ge a_j \end{cases}$$

The log likelihood is given by

$$\begin{aligned} l(\boldsymbol{\lambda}, \boldsymbol{\beta}) &= \sum_{i=1}^{n} \delta_i \left[\sum_{j=1}^{J} I(a_{j-1} \le t_i < a_j) \log \lambda_j + \boldsymbol{\beta}'\mathbf{x}_i \right] - \sum_{i=1}^{n} \left[\sum_{j=1}^{J} \Delta_j(t_i) \lambda_j \right] \exp(\boldsymbol{\beta}'\mathbf{x}_i) \\ &= \sum_{j=1}^{J} d_j \log \lambda_j + \sum_{i=1}^{n} \delta_i \boldsymbol{\beta}'\mathbf{x}_i - \sum_{j=1}^{J} \lambda_j \left[\sum_{i=1}^{n} \Delta_j(t_i) \exp(\boldsymbol{\beta}'\mathbf{x}_i) \right] \end{aligned}$$

where $d_j = \sum_{i=1}^{n} \delta_i I(a_{j-1} \le t_i < a_j)$.

Note that for $1 \le j \le J$, the full conditional for λ_j is log-concave only when $d_j > 0$, but the full conditionals for the β's are always log-concave.

For a given $\boldsymbol{\beta}$, $\frac{\partial l}{\partial \boldsymbol{\lambda}} = 0$ gives

$$\tilde{\lambda}_j(\boldsymbol{\beta}) = \frac{d_j}{\sum_{i=1}^{n} \Delta_j(t_i) \exp(\boldsymbol{\beta}'\mathbf{x}_i)}, \quad j = 1, \ldots, J$$

Substituting these values into $l(\boldsymbol{\lambda}, \boldsymbol{\beta})$ gives the profile log likelihood for $\boldsymbol{\beta}$

$$l_p(\boldsymbol{\beta}) = \sum_{i=1}^{n} \delta_i \boldsymbol{\beta}'\mathbf{x}_i - \sum_{j=1}^{J} d_j \log \left[\sum_{l=1}^{n} \Delta_j(t_l) \exp(\boldsymbol{\beta}'\mathbf{x}_l) \right] + c$$

where $c = \sum_j (d_j \log d_j - d_j)$. Since the constant c does not depend on $\boldsymbol{\beta}$, it can be discarded from $l_p(\boldsymbol{\beta})$ in the optimization.

The MLE $\hat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ is obtained by maximizing

$$l_p(\boldsymbol{\beta}) = \sum_{i=1}^{n} \delta_i \boldsymbol{\beta}' \mathbf{x}_i - \sum_{j=1}^{J} d_j \log\left[\sum_{l=1}^{n} \Delta_j(t_l) \exp(\boldsymbol{\beta}' \mathbf{x}_l)\right]$$

with respect to $\boldsymbol{\beta}$, and the MLE $\hat{\boldsymbol{\lambda}}$ of $\boldsymbol{\lambda}$ is given by

$$\hat{\boldsymbol{\lambda}} = \tilde{\boldsymbol{\lambda}}(\hat{\boldsymbol{\beta}})$$

For $j = 1, \ldots, J$, let

$$\begin{aligned}
\mathbf{S}_j^{(r)}(\boldsymbol{\beta}) &= \sum_{l=1}^{n} \Delta_j(t_l) \mathrm{e}^{\boldsymbol{\beta}' \mathbf{x}_l} \mathbf{x}_l^{\otimes r}, \quad r = 0, 1, 2 \\
\mathbf{E}_j(\boldsymbol{\beta}) &= \frac{\mathbf{S}_j^{(1)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})}
\end{aligned}$$

The partial derivatives of $l_p(\boldsymbol{\beta})$ are

$$\begin{aligned}
\frac{\partial l_p(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} &= \sum_{i=1}^{n} \delta_i \mathbf{x}_i - \sum_{j=1}^{J} d_j \mathbf{E}_j(\boldsymbol{\beta}) \\
-\frac{\partial^2 l_p(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}^2} &= \sum_{j=1}^{J} d_j \left\{ \frac{\mathbf{S}_j^{(2)}(\boldsymbol{\beta})}{S_j^{(0)}(\boldsymbol{\beta})} - \Big[\mathbf{E}_j(\boldsymbol{\beta})\Big]\Big[\mathbf{E}_j(\boldsymbol{\beta})\Big]' \right\}
\end{aligned}$$

The asymptotic covariance matrix for $(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\beta}})$ is obtained as the inverse of the information matrix given by

$$\begin{aligned}
-\frac{\partial^2 l(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\lambda}^2} &= \mathcal{D}\left(\frac{d_1}{\hat{\lambda}_1^2}, \ldots, \frac{d_J}{\hat{\lambda}_J^2}\right) \\
-\frac{\partial^2 l(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}^2} &= \sum_{j=1}^{J} \hat{\lambda}_j \mathbf{S}_j^{(2)}(\hat{\boldsymbol{\beta}}) \\
-\frac{\partial^2 l(\hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\lambda} \partial \boldsymbol{\beta}} &= (\mathbf{S}_1^{(1)}(\hat{\boldsymbol{\beta}}), \ldots, \mathbf{S}_J^{(1)}(\hat{\boldsymbol{\beta}}))
\end{aligned}$$

See Example 6.5.1 in Lawless (2003) for details.

Hazards in Log Scale

By letting

$$\alpha_j = \log(\lambda_j), \quad j = 1, \ldots, J$$

you can build a prior correlation among the λ_j's by using a correlated prior $\boldsymbol{\alpha} \sim N(\boldsymbol{\alpha}_0, \Sigma_\alpha)$, where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_J)'$.

The log likelihood is given by

$$l(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{j=1}^{J} d_j \alpha_j + \sum_{i=1}^{n} \delta_i \boldsymbol{\beta}' \mathbf{x}_i - \sum_{j=1}^{J} \mathrm{e}^{\alpha_j} S_j^{(0)}(\boldsymbol{\beta})$$

Then the MLE of λ_j is given by

$$\mathrm{e}^{\hat{\alpha}_j} = \hat{\lambda}_j = \frac{d_j}{S_j^0(\hat{\boldsymbol{\beta}})}$$

Note that the full conditionals for α's and β's are always log-concave.

The asymptotic covariance matrix for $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}})$ is obtained as the inverse of the information matrix formed by

$$\begin{aligned}
-\frac{\partial^2 l(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\alpha}^2} &= \mathcal{D}\left(\mathrm{e}^{\hat{\alpha}_j} S_j^0(\hat{\boldsymbol{\beta}}), \ldots, \mathrm{e}^{\hat{\alpha}_J} S_J^0(\hat{\boldsymbol{\beta}}))\right) \\
-\frac{\partial^2 l(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\beta}^2} &= \sum_{j=1}^{J} \mathrm{e}^{\hat{\alpha}_j} \mathbf{S}_j^{(2)}(\hat{\boldsymbol{\beta}}) \\
-\frac{\partial^2 l(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}})}{\partial \boldsymbol{\alpha} \partial \boldsymbol{\beta}} &= (\mathrm{e}^{\hat{\alpha}_j} \mathbf{S}_1^{(1)}(\hat{\boldsymbol{\beta}}), \ldots, \mathrm{e}^{\hat{\alpha}_j} \mathbf{S}_J^{(1)}(\hat{\boldsymbol{\beta}}))
\end{aligned}$$

Priors for Model Parameters

For a Cox model, the model parameters are the regression coefficients. For a piecewise exponential model, the model parameters consist of the regression coefficients and the hazards or log-hazards. The priors for the hazards and the priors for the regression coefficients are assumed to be independent, while you can have a joint multivariate normal prior for the log-hazards and the regression coefficients. For a proportional hazards spline model, the model parameters consist of the regression coefficients and the cubic spline parameters. You can have a joint multivariate normal prior for the cubic spline parameters and the regression coefficients. Otherwise, the prior for the cubic spline parameters and the prior for the regression coefficients are assumed to be independent.

Cubic Spline Parameters

Uniform Prior The joint prior density is given by

$$p(\gamma_1, \ldots, \gamma_J) \propto 1, \forall -\infty < \gamma_i < \infty$$

Normal Prior Assume $\boldsymbol{\gamma}$ has a multivariate normal prior with mean vector $\boldsymbol{\gamma}_0$ and covariance matrix $\boldsymbol{\Psi}_0$. The joint prior density is given by

$$p(\gamma) \propto \mathrm{e}^{-\frac{1}{2}(\boldsymbol{\gamma}-\boldsymbol{\gamma}_0)' \boldsymbol{\Psi}_0^{-1} (\boldsymbol{\gamma}-\boldsymbol{\gamma}_0)}$$

Hazard Parameters

Let $\lambda_1, \ldots, \lambda_J$ be the constant baseline hazards.

Improper Prior The joint prior density is given by

$$p(\lambda_1, \ldots, \lambda_J) = \prod_{j=1}^{J} \frac{1}{\lambda_j} \text{ for all } \lambda_j > 0$$

This prior is improper (nonintegrable), but the posterior distribution is proper as long as there is at least one event time in each of the constant hazard intervals.

Uniform Prior The joint prior density is given by

$$p(\lambda_1, \ldots, \lambda_J) \propto 1 \text{ for all } \lambda_j > 0$$

This prior is improper (nonintegrable), but the posteriors are proper as long as there is at least one event time in each of the constant hazard intervals.

Gamma Prior The gamma distribution $G(a, b)$ has a PDF

$$f_{a,b}(t) = \frac{b(bt)^{a-1}\mathrm{e}^{-bt}}{\Gamma(a)}, t > 0$$

where a is the shape parameter and b^{-1} is the scale parameter. The mean is $\frac{a}{b}$ and the variance is $\frac{a}{b^2}$.

Independent Gamma Prior Suppose for $j = 1, \ldots, J$, λ_j has an independent $G(a_j, b_j)$ prior. The joint prior density is given by

$$p(\lambda_1, \ldots, \lambda_J) \propto \prod_{j=1}^{J} \left\{ \lambda_j^{a_j - 1} \mathrm{e}^{-b_j \lambda_j} \right\}, \forall \lambda_j > 0$$

AR1 Prior $\lambda_1, \ldots, \lambda_J$ are correlated as follows:

$$\begin{aligned}
\lambda_1 &\sim G(a_1, b_1) \\
\lambda_2 &\sim G\left(a_2, \frac{b_2}{\lambda_1}\right) \\
\ldots & \quad \ldots \\
\lambda_J &\sim G\left(a_J, \frac{b_J}{\lambda_{J-1}}\right)
\end{aligned}$$

The joint prior density is given by

$$p(\lambda_1, \ldots, \lambda_J) \propto \lambda_1^{a_1 - 1} \mathrm{e}^{-b_1 \lambda_1} \prod_{j=2}^{J} \left(\frac{b_j}{\lambda_{j-1}} \right)^{a_j} \lambda_j^{a_j - 1} \mathrm{e}^{-\frac{b_j}{\lambda_{j-1}} \lambda_j}$$

Log-Hazard Parameters

Write $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_J)' \equiv (\log \lambda_1, \ldots, \log \lambda_J)'$.

Uniform Prior The joint prior density is given by

$$p(\alpha_1, \ldots, \alpha_J) \propto 1, \forall -\infty < \alpha_i < \infty$$

Note that the uniform prior for the log-hazards is the same as the improper prior for the hazards.

Normal Prior Assume $\boldsymbol{\alpha}$ has a multivariate normal prior with mean vector $\boldsymbol{\alpha}_0$ and covariance matrix $\boldsymbol{\Psi}_0$. The joint prior density is given by

$$p(\boldsymbol{\alpha}) \propto \mathrm{e}^{-\frac{1}{2}(\boldsymbol{\alpha}-\boldsymbol{\alpha}_0)'\boldsymbol{\Psi}_0^{-1}(\boldsymbol{\alpha}-\boldsymbol{\alpha}_0)}$$

Regression Coefficients

Let $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)'$ be the vector of regression coefficients.

Uniform Prior The joint prior density is given by

$$p(\beta_1, \ldots, \beta_k) \propto 1, \forall -\infty < \beta_i < \infty$$

This prior is improper, but the posterior distributions for $\boldsymbol{\beta}$ are proper.

Normal Prior Assume $\boldsymbol{\beta}$ has a multivariate normal prior with mean vector $\boldsymbol{\beta}_0$ and covariance matrix $\boldsymbol{\Sigma}_0$. The joint prior density is given by

$$p(\boldsymbol{\beta}) \propto \mathrm{e}^{-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\beta}_0)'\boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\beta}-\boldsymbol{\beta}_0)}$$

Joint Multivariate Normal Prior for Log-Hazards and Regression Coefficients Assume $(\boldsymbol{\alpha}', \boldsymbol{\beta}')'$ has a multivariate normal prior with mean vector $(\alpha_0', \boldsymbol{\beta}_0')'$ and covariance matrix $\boldsymbol{\Phi}_0$. The joint prior density is given by

$$p(\boldsymbol{\alpha}, \boldsymbol{\beta}) \propto \mathrm{e}^{-\frac{1}{2}[(\boldsymbol{\alpha}-\boldsymbol{\alpha}_0)',(\boldsymbol{\beta}-\boldsymbol{\beta}_0)']\boldsymbol{\Phi}_0^{-1}[(\boldsymbol{\alpha}-\boldsymbol{\alpha}_0)',(\boldsymbol{\beta}-\boldsymbol{\beta}_0)']'}$$

Joint Multivariate Normal Prior for Cubic Spline Parameters and Regression Coefficients Assume $(\boldsymbol{\gamma}', \boldsymbol{\beta}')'$ has a multivariate normal prior with mean vector $(\gamma_0', \beta_0')'$ and covariance matrix $\boldsymbol{\Phi}_0$. The joint prior density is given by

$$p(\boldsymbol{\gamma}, \boldsymbol{\beta}) \propto \mathrm{e}^{-\frac{1}{2}[(\boldsymbol{\gamma}-\boldsymbol{\alpha}_0)',(\boldsymbol{\beta}-\boldsymbol{\beta}_0)']\boldsymbol{\Phi}_0^{-1}[(\boldsymbol{\gamma}-\boldsymbol{\alpha}_0)',(\boldsymbol{\beta}-\boldsymbol{\beta}_0)']'}$$

Zellner's *g*-Prior Assume $\boldsymbol{\beta}$ has a multivariate normal prior with mean vector $\mathbf{0}$ and covariance matrix $(g\mathbf{X}'\mathbf{X})^{-1}$, where $\mathbf{X}$ is the design matrix and g is either a constant or it follows a gamma prior with density $f(\tau) = \frac{b(b\tau)^{a-1}\mathrm{e}^{-b\tau}}{\Gamma(a)}$ where a and b are the SHAPE= and ISCALE= parameters. Let k be the rank of $\mathbf{X}$. The joint prior density with g being a constant c is given by

$$p(\boldsymbol{\beta}) \propto c^{\frac{k}{2}}\mathrm{e}^{-\frac{1}{2}\boldsymbol{\beta}'(c\mathbf{X}'\mathbf{X})^{-1}\boldsymbol{\beta}}$$

The joint prior density with g having a gamma prior is given by

$$p(\boldsymbol{\beta}, \tau) \propto \tau^{\frac{k}{2}}\mathrm{e}^{-\frac{1}{2}\boldsymbol{\beta}'(\tau\mathbf{X}'\mathbf{X})^{-1}\boldsymbol{\beta}}\frac{b(b\tau)^{a-1}\mathrm{e}^{-b\tau}}{\Gamma(a)}$$

Dispersion Parameter for Frailty Model

Improper Prior The density is

$$p(\theta) = \frac{1}{\theta}$$

Inverse Gamma Prior The inverse gamma distribution $IG(a, b)$ has a density

$$p(\theta|a, b) = \frac{b^a \theta^{-(a+1)} \mathrm{e}^{-\frac{b}{\theta}}}{\Gamma(a)}$$

where a and b are the SHAPE= and SCALE= parameters, respectively.

Gamma Prior The gamma distribution $G(a, b)$ has a density

$$p(\theta|a, b) = \frac{b^a \theta^{a-1} \mathrm{e}^{-b\theta}}{\Gamma(a)}$$

where a and b are the SHAPE= and ISCALE= parameters, respectively.

Posterior Distribution

Denote the observed data as D.

Cox Model

$$\pi(\boldsymbol{\beta}|D) \propto \underbrace{L(D|\boldsymbol{\beta})}_{\text{partial likelihood}} \overbrace{p(\boldsymbol{\beta})}^{\text{prior}}$$

Proportional Hazards Spline Model

$$\pi(\boldsymbol{\gamma}, \boldsymbol{\beta}|D) \propto \begin{cases} L_{\mathrm{RP}}(D|\boldsymbol{\gamma}, \boldsymbol{\beta}) p(\boldsymbol{\gamma}, \boldsymbol{\beta}) & \text{if} \quad (\gamma', \boldsymbol{\beta}')' \sim \mathrm{MVN} \\ L_{\mathrm{RP}}(D|\boldsymbol{\gamma}, \boldsymbol{\beta}) p(\boldsymbol{\gamma}) p(\boldsymbol{\beta}) & \text{otherwise} \end{cases}$$

where L_{RP} is the likelihood function with cubic spline parameters $\boldsymbol{\gamma}$ and regression coefficients $\boldsymbol{\beta}$ as parameters.

Frailty Model

Based on the framework of Sargent (1998),

$$\pi(\boldsymbol{\beta}, \boldsymbol{\gamma}, \theta|D) \propto \underbrace{L(D|\boldsymbol{\beta}, \boldsymbol{\gamma})}_{\text{partial likelihood}} \overbrace{g(\boldsymbol{\gamma}|\theta)}^{\text{random effects}} \underbrace{p(\boldsymbol{\beta}) p(\theta)}_{\text{priors}}$$

where the joint density of the random effects $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_s)'$ is given by

$$g(\boldsymbol{\gamma}|\theta) \propto \begin{cases} \prod_i \exp\left(\frac{\gamma_i}{\theta}\right) \exp\left(-\exp\left(\frac{\gamma_i}{\theta}\right)\right) & \text{gamma frailty} \\ \prod_i \exp\left(-\frac{\gamma_i^2}{2\theta}\right) & \text{lognormal frailty} \end{cases}$$

Piecewise Exponential Model

Hazard Parameters

$$\pi(\boldsymbol{\lambda}, \boldsymbol{\beta}|D) \propto L_H(D|\boldsymbol{\lambda}, \boldsymbol{\beta})p(\boldsymbol{\lambda})p(\boldsymbol{\beta})$$

where $L_H(D|\boldsymbol{\lambda}, \boldsymbol{\beta})$ is the likelihood function with hazards $\boldsymbol{\lambda}$ and regression coefficients $\boldsymbol{\beta}$ as parameters.

Log-Hazard Parameters

$$\pi(\boldsymbol{\alpha}, \boldsymbol{\beta}|D) \propto \begin{cases} L_{\mathrm{LH}}(D|\boldsymbol{\alpha}, \boldsymbol{\beta})p(\boldsymbol{\alpha}, \boldsymbol{\beta}) & \text{if} \quad (\boldsymbol{\alpha}', \boldsymbol{\beta}')' \sim \mathrm{MVN} \\ L_{\mathrm{LH}}(D|\boldsymbol{\alpha}, \boldsymbol{\beta})p(\boldsymbol{\alpha})p(\boldsymbol{\beta}) & \text{otherwise} \end{cases}$$

where $L_{\mathrm{LH}}(D|\boldsymbol{\alpha}, \boldsymbol{\beta})$ is the likelihood function with log-hazards $\boldsymbol{\alpha}$ and regression coefficients $\boldsymbol{\beta}$ as parameters.

Sampling from the Posterior Distribution

For the Gibbs sampler, PROC PHREG uses the ARMS (adaptive rejection Metropolis sampling) algorithm of Gilks, Best, and Tan (1995) to sample from the full conditionals. This is the default sampling scheme. Alternatively, you can requests the random walk Metropolis (RWM) algorithm to sample an entire parameter vector from the posterior distribution. For a general discussion of these algorithms, see section "Markov Chain Monte Carlo Method" on page 155 in Chapter 8, "Introduction to Bayesian Analysis Procedures."

You can output these posterior samples into a SAS data set by using the OUTPOST= option in the BAYES statement, or you can use the following SAS statement to output the posterior samples into the SAS data set Post:

```
ods output PosteriorSample=Post;
```

The output data set also includes the variables LogLike and LogPost, which represent the log of the likelihood and the log of the posterior log density, respectively.

Let $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)'$ be the parameter vector. For the Cox model, the θ_i's are the regression coefficients β_i's, and for the piecewise constant baseline hazard model, the θ_i's consist of the baseline hazards λ_i's (or log baseline hazards α_i's) and the regression coefficients β_j's. Let $L(D|\boldsymbol{\theta})$ be the likelihood function, where D is the observed data. Note that for the Cox model, the likelihood contains the infinite-dimensional baseline hazard function, and the gamma process is perhaps the most commonly used prior process (Ibrahim, Chen, and Sinha 2001). However, Sinha, Ibrahim, and Chen (2003) justify using the partial likelihood as the likelihood function for the Bayesian analysis. Let $p(\boldsymbol{\theta})$ be the prior distribution. The posterior $f\pi(\boldsymbol{\theta}|D)$ is proportional to the joint distribution $L(D|\boldsymbol{\theta})p(\boldsymbol{\theta})$.

Gibbs Sampler

The full conditional distribution of θ_i is proportional to the joint distribution; that is,

$$\pi(\theta_i|\theta_j, i \neq j, D) \propto L(D|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

For example, the one-dimensional conditional distribution of θ_1, given $\theta_j = \theta_j^*, 2 \leq j \leq k$, is computed as

$$\pi(\theta_1|\theta_j = \theta_j^*, 2 \leq j \leq k, D) = L(D|\boldsymbol{\theta} = (\theta_1, \theta_2^*, \ldots, \theta_k^*)')p(\boldsymbol{\theta} = (\theta_1, \theta_2^*, \ldots, \theta_k^*)')$$

Suppose you have a set of arbitrary starting values $\{\theta_1^{(0)}, \ldots, \theta_k^{(0)}\}$. Using the ARMS algorithm, an iteration of the Gibbs sampler consists of the following:

- draw $\theta_1^{(1)}$ from $\pi(\theta_1|\theta_2^{(0)},\ldots,\theta_k^{(0)},D)$
- draw $\theta_2^{(1)}$ from $\pi(\theta_2|\theta_1^{(1)},\theta_3^{(0)},\ldots,\theta_k^{(0)},D)$
- $\vdots$
- draw $\theta_k^{(1)}$ from $\pi(\theta_k|\theta_1^{(1)},\ldots,\theta_{k-1}^{(1)},D)$

After one iteration, you have $\{\theta_1^{(1)},\ldots,\theta_k^{(1)}\}$. After n iterations, you have $\{\theta_1^{(n)},\ldots,\theta_k^{(n)}\}$. Cumulatively, a chain of n samples is obtained.

Random Walk Metropolis Algorithm

PROC PHREG uses a multivariate normal proposal distribution $q(.|\boldsymbol{\theta})$ centered at $\boldsymbol{\theta}$. With an initial parameter vector $\boldsymbol{\theta}^{(0)}$, a new sample $\boldsymbol{\theta}^{(1)}$ is obtained as follows:

- sample $\boldsymbol{\theta}^*$ from $q(.|\boldsymbol{\theta}^{(0)})$
- calculate the quantity $r = \min\left\{\frac{\pi(\boldsymbol{\theta}^*|D)}{\pi(\boldsymbol{\theta}^{(0)}|D)}, 1\right\}$
- sample u from the uniform distribution $U(0,1)$
- set $\boldsymbol{\theta}^{(1)} = \boldsymbol{\theta}^*$ if $u < r$; otherwise set $\boldsymbol{\theta}^{(1)} = \boldsymbol{\theta}^{(0)}$

With $\boldsymbol{\theta}^{(1)}$ taking the role of $\boldsymbol{\theta}^{(0)}$, the previous steps are repeated to generate the next sample $\boldsymbol{\theta}^{(2)}$. After n iterations, a chain of n samples $\{\boldsymbol{\theta}^{(1)},\ldots,\boldsymbol{\theta}^{(n)}\}$ is obtained.

Starting Values of the Markov Chains

When the BAYES statement is specified, PROC PHREG generates one Markov chain that contains the approximate posterior samples of the model parameters. Additional chains are produced when the Gelman-Rubin diagnostics are requested. Starting values (initial values) can be specified in the INITIAL= data set in the BAYES statement. If the INITIAL= option is not specified, PROC PHREG picks its own initial values for the chains based on the maximum likelihood estimates of $\boldsymbol{\theta}$ and the prior information of $\boldsymbol{\theta}$.

Denote $[x]$ as the integral value of x.

Constant Baseline Hazard Parameters λ_i's

For the first chain that the summary statistics and diagnostics are based on, the initial values are

$$\lambda_i^{(0)} = \hat{\lambda}_i$$

For subsequent chains, the starting values are picked in two different ways according to the total number of chains specified. If the total number of chains specified is less than or equal to 10, initial values of the rth chain ($2 \le r \le 10$) are given by

$$\lambda_i^{(0)} = \hat{\lambda}_i \mathrm{e}^{\pm\left([\frac{r}{2}]+2\right)\hat{s}(\hat{\lambda}_i)}$$

with the plus sign for odd r and minus sign for even r. If the total number of chains is greater than 10, initial values are picked at random over a wide range of values. Let u_i be a uniform random number between 0 and 1; the initial value for λ_i is given by

$$\lambda_i^{(0)} = \hat{\lambda}_i \mathrm{e}^{16(u_i - 0.5)\hat{s}(\hat{\lambda}_i)}$$

Regression Coefficients and Log-Hazard Parameters θ_i's

The θ_i's are the regression coefficients β_i's, and in the piecewise exponential model, include the log-hazard parameters α_i's. For the first chain that the summary statistics and regression diagnostics are based on, the initial values are

$$\theta_i^{(0)} = \hat{\theta}_i$$

If the number of chains requested is less than or equal to 10, initial values for the rth chain ($2 \le r \le 10$) are given by

$$\theta_i^{(0)} = \hat{\theta}_i \pm \left(2 + \left[\frac{r}{2}\right]\right)\hat{s}(\hat{\theta}_i)$$

with the plus sign for odd r and minus sign for even r. When there are more than 10 chains, the initial value for the θ_i is picked at random over the range $(\hat{\theta}_i - 8\hat{s}(\hat{\theta}_i), \hat{\theta}_i + 8\hat{s}(\hat{\theta}_i))$; that is,

$$\theta_i^{(0)} = \hat{\theta}_i + 16(u_i - 0.5)\hat{s}(\hat{\theta}_i)$$

where u_i is a uniform random number between 0 and 1.

Fit Statistics

Denote the observed data by D. Let $\boldsymbol{\theta}$ be the vector of parameters of length k. Let $L(D|\boldsymbol{\theta})$ be the likelihood. The deviance information criterion (DIC) proposed in Spiegelhalter et al. (2002) is a Bayesian model assessment tool. Let $\mathrm{Dev}(\boldsymbol{\theta}) = -2\log L(D|\boldsymbol{\theta})$. Let $\overline{\mathrm{Dev}(\theta)}$ and $\bar{\boldsymbol{\theta}}$ be the corresponding posterior means of $\mathrm{Dev}(\boldsymbol{\theta})$ and $\boldsymbol{\theta}$, respectively. The deviance information criterion is computed as

$$\mathrm{DIC} = 2\overline{\mathrm{Dev}(\boldsymbol{\theta})} - \mathrm{Dev}(\bar{\boldsymbol{\theta}})$$

Also computed is

$$pD = \overline{\mathrm{Dev}(\boldsymbol{\theta})} - \mathrm{Dev}(\bar{\boldsymbol{\theta}})$$

where pD is interpreted as the effective number of parameters.

Note that $\mathrm{Dev}(\boldsymbol{\theta})$ defined here does not have the standardizing term as in the section "Deviance Information Criterion (DIC)" on page 177 in Chapter 8, "Introduction to Bayesian Analysis Procedures." Nevertheless, the DIC calculated here is still useful for variable selection.

Posterior Distribution for Quantities of Interest

Let $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_k)'$ be the parameter vector. For the Cox model, the θ_i's are the regression coefficients β_i's; for the proportional hazards spline model, the θ_i's consist of the cubic spline parameters γ_i's and the regression coefficients β_j's; for the piecewise constant baseline hazard model, the θ_i's consist of the baseline hazards λ_i's (or log baseline hazards α_i's) and the regression coefficients β_j's.

Let $\mathcal{S} = \{\boldsymbol{\theta}^{(r)}, r = 1, \ldots, N\}$ be the chain that represents the posterior distribution for $\boldsymbol{\theta}$.

Consider a quantity of interest τ that can be expressed as a function $f(\boldsymbol{\theta})$ of the parameter vector $\boldsymbol{\theta}$. You can construct the posterior distribution of τ by evaluating the function $f(\boldsymbol{\theta}^{(r)})$ for each $\boldsymbol{\theta}^{(r)}$ in $\mathcal{S}$. The posterior chain for τ is $\{f(\boldsymbol{\theta}^{(r)}), r = 1, \ldots, N\}$. Summary statistics such as mean, standard deviation, percentiles, and credible intervals are used to describe the posterior distribution of τ.

Hazard Ratio

As shown in the section "Hazard Ratios" on page 7563, a log-hazard ratio is a linear combination of the regression coefficients. Let $\mathbf{h}$ be the vector of linear coefficients. The posterior sample for this hazard ratio is the set $\{\exp(\mathrm{h}'\boldsymbol{\beta}^{(r)}), r = 1, \ldots, N\}$.

Survival Distribution

Let x be a covariate vector of interest.

Cox Model Let $\{(t_i, \mathbf{z}_i, \delta_i), i = 1, 2, \ldots, n\}$ be the observed data. Define

$$Y_i(t) = \begin{cases} 1 & t < t_i \\ 0 & \text{otherwise} \end{cases}$$

Consider the rth draw $\boldsymbol{\beta}^{(r)}$ of $\mathcal{S}$. The baseline cumulative hazard function at time t is given by

$$H_0(t|\boldsymbol{\beta}^{(r)}) = \sum_{i:t_i \le t} \frac{\delta_i}{\sum_{l=1}^{n} Y_l(t_i)\exp(\mathbf{z}_l'\boldsymbol{\beta}^{(r)})}$$

For the given covariate vector x, the cumulative hazard function at time t is

$$H(t; \mathbf{x}|\boldsymbol{\beta}^{(r)}) = H_0(t|\boldsymbol{\beta}^{(r)}) \exp(\mathbf{x}'\boldsymbol{\beta}^{(r)})$$

and the survival function at time t is

$$S(t; \mathbf{x}|\boldsymbol{\beta}^{(r)}) = \exp[-H(t; \mathbf{x}|\boldsymbol{\beta}^{(r)})]$$

Proportional Hazards Spline Model Consider the rth draw $\boldsymbol{\theta}^{(r)}$ in $\mathcal{S}$, where $\boldsymbol{\theta}^{(r)}$ consists of $\boldsymbol{\gamma}^{(r)} = (\gamma_1^{(r)}, \ldots, \gamma_J^{(r)})'$ and $\boldsymbol{\beta}^{(r)}$. The baseline cumulative hazard function at time t is

$$H_0(t|\boldsymbol{\gamma}^{(r)}) = \exp[s(\log(t), \boldsymbol{\gamma}^{(r)})]$$

where $s(\log(t), \boldsymbol{\gamma}^{(r)})$ is a cubic spline function as described for the SPLINE option in the BAYES statement. For the given covariate vector x, the cumulative hazard function at time t is

$$H(t; \mathbf{x}|\boldsymbol{\gamma}^{(r)}, \boldsymbol{\beta}^{(r)}) = H_0(t|\boldsymbol{\gamma}^{(r)}) \exp(\mathbf{x}'\boldsymbol{\beta}^{(r)})$$

and the survival function at time t is

$$S(t; \mathbf{x}|\boldsymbol{\gamma}^{(r)}, \boldsymbol{\beta}^{(r)}) = \exp[-H(t; \mathbf{x}|\boldsymbol{\gamma}^{(r)}, \boldsymbol{\beta}^{(r)})]$$

Piecewise Exponential Model Let $0 = a_0 < a_1 < \ldots < a_J < \infty$ be a partition of the time axis. Consider the rth draw $\boldsymbol{\theta}^{(r)}$ in $\mathcal{S}$, where $\boldsymbol{\theta}^{(r)}$ consists of $\boldsymbol{\lambda}^{(r)} = (\lambda_1^{(r)}, \ldots, \lambda_J^{(r)})'$ and $\boldsymbol{\beta}^{(r)}$. The baseline cumulative hazard function at time t is

$$H_0(t|\boldsymbol{\lambda}^{(r)}) = \sum_{j=1}^{J} \lambda_j^{(r)} \Delta_j(t)$$

where

$$\Delta_j(t) = \begin{cases} 0 & t < a_{j-1} \\ t - a_{j-1} & a_{j-1} \le t < a_j \\ a_j - a_{j-1} & t \ge a_j \end{cases}$$

For the given covariate vector $\mathbf{x}$, the cumulative hazard function at time t is

$$H(t; \mathbf{x}|\boldsymbol{\lambda}^{(r)}, \boldsymbol{\beta}^{(r)}) = H_0(t|\boldsymbol{\lambda}^{(r)}) \exp(\mathbf{x}'\boldsymbol{\beta}^{(r)})$$

and the survival function at time t is

$$S(t; \mathbf{x}|\boldsymbol{\lambda}^{(r)}, \boldsymbol{\beta}^{(r)}) = \exp[-H(t; \mathbf{x}|\boldsymbol{\lambda}^{(r)}, \boldsymbol{\beta}^{(r)})]$$

Input and Output Data Sets

OUTEST= Output Data Set

The OUTEST= data set contains one observation for each BY group containing the maximum likelihood estimates of the regression coefficients. If you also use the COVOUT option in the PROC PHREG statement, there are additional observations containing the rows of the estimated covariance matrix. If you specify SELECTION=FORWARD, BACKWARD, or STEPWISE, only the estimates of the parameters and covariance matrix for the final model are output to the OUTEST= data set.

Variables in the OUTEST= Data Set

The OUTEST= data set contains the following variables:

- any BY variables specified
- _TIES_, a character variable of length 8 with four possible values: BRESLOW, DISCRETE, EFRON, and EXACT. These are the four values of the TIES= option in the MODEL statement.
- _TYPE_, a character variable of length 8 with two possible values: PARMS for parameter estimates or COV for covariance estimates. If both the COVM and COVSANDWICH options are specified in the PROC PHREG statement along with the COVOUT option, _TYPE_='COVM' for the model-based covariance estimates and _TYPE_='COVS' for the robust sandwich covariance estimates.
- _STATUS_, a character variable indicating whether the estimates have converged
- _NAME_, a character variable containing the name of the TIME variable for the row of parameter estimates and the name of each explanatory variable to label the rows of covariance estimates

- one variable for each regression coefficient and one variable for the offset variable if the OFFSET= option is specified. If an explanatory variable is not included in the final model in a variable selection process, the corresponding parameter estimates and covariances are set to missing.
- _LNLIKE_, a numeric variable containing the last computed value of the log likelihood

Parameter Names in the OUTEST= Data Set

For continuous explanatory variables, the names of the parameters are the same as the corresponding variables. For CLASS variables, the parameter names are obtained by concatenating the corresponding CLASS variable name with the CLASS category; see the PARAM= option in the CLASS statement for more details. For interaction and nested effects, the parameter names are created by concatenating the names of each component effect.

INEST= Input Data Set

You can specify starting values for the maximum likelihood iterative algorithm in the INEST= data set. The INEST= data set has the same structure as the OUTEST= data set but is not required to have all the variables or observations that appear in the OUTEST= data set.

The INEST= data set must contain variables that represent the regression coefficients of the model. If BY processing is used, the INEST= data set should also include the BY variables, and there must be one observation for each BY group. If the INEST= data set also contains the _TYPE_ variable, only observations with _TYPE_ value 'PARMS' are used as starting values.

OUT= Output Data Set in the ZPH Option

The OUT= data set in the ZPH option contains the variable of event times and the variables that represent the time-varying coefficients, one for each parameter. If the transformation that you specify in the ZPH option is not an identity, the OUT= data set also contains a variable that represents the transformed event times.

OUT= Output Data Set in the OUTPUT Statement

The OUT= data set in the OUTPUT statement contains all the variables in the input data set, along with statistics you request by specifying *keyword=name* options. The new variables contain a variety of diagnostics that are calculated for each observation in the input data set.

OUT= Output Data Set in the BASELINE Statement

The OUT= data set in the BASELINE statement contains all the variables in the COVARIATES= data set, along with statistics you request by specifying *keyword=name* options. For unstratified input data, there are 1 + n observations in the OUT= data set for each observation in the COVARIATES= data set, where n is the number of distinct event times in the input data. For input data that are stratified into k strata, with n_i distinct events in the ith stratum, $i = 1, \ldots, k$, there are $1+n_i$ observations for the ith stratum in the OUT= data set for each observation in the COVARIATES= data set.

OUTAUC= Output Data Set in the ROCOPTIONS Option

The OUTAUC= data set contains data necessary for producing the AUC (area under the curve) plot; it can be created by specifying the OUTAUC= suboption in the ROCOPTIONS option in the PROC PHREG statement. This data set has the following variables:

- any specified BY variables
- _ID_, the sequence number of the model
- _SOURCE_, the model label
- the time variable, which identifies time point t at which the AUC statistic is calculated
- _AUC_, the AUC statistic, which is the area under the time-dependent ROC curve at each event time
- _STDERR_, the standard error of the AUC statistic
- _LOWERAUC_, the lower confidence limit for the AUC
- _UPPERAUC_, the upper confidence limit for the AUC

OUTIDFF= Output Data Set in the BASELINE Statement

The OUTDIFF= data set contains the differences of the direct adjusted survival probabilities between two treatments or two strata and their standard errors.

OUTPOST= Output Data Set in the BAYES Statement

The OUTPOST= data set contains the generated posterior samples. There are 3+n variables, where n is the number of model parameters. The variable Iteration represents the iteration number, the variable LogLike contains the log-likelihood values, and the variable LogPost contains the log-posterior-density values. The other n variables represent the draws of the Markov chain for the model parameters.

OUTROC= Output Data Set in the ROCOPTIONS Option

The OUTROC= data set contains data necessary for producing the ROC plot; it can be created by specifying the OUTROC= suboption in the ROCOPTIONS option in the PROC PHREG statement. This data set has the following variables:

- any specified BY variables
- the time variable, which identifies the time point at which the ROC curve is calculated
- _ID_, the sequence number of the model
- _SOURCE_, the model label
- _CUTOFF_, the cutpoint for predicting the response. Any observation whose predicted value exceeds or equals to the cutpoint is predicted to be an event; otherwise, it is predicted to be a nonevent.

- _SENSITIVITY_, the sensitivity, which is the proportion of event observations that were predicted to have an event response
- _SPECIFICITY_, the specificity, which is the proportion of nonevent observations that were predicted to have a nonevent response

Displayed Output

If you use the NOPRINT option in the PROC PHREG statement, the procedure does not display any output. Otherwise, PROC PHREG displays results of the analysis in a collection of tables. The tables are listed separately for the maximum likelihood analysis and for the Bayesian analysis.

Maximum Likelihood Analysis Displayed Output

Model Information

The "Model Information" table displays the two-level name of the input data set, the name and label of the failure time variable, the name and label of the censoring variable and the values indicating censored times, the model (either the Cox model or the piecewise constant baseline hazard model), the name and label of the OFFSET variable, the name and label of the FREQ variable, the name and label of the WEIGHT variable, and the method of handling ties in the failure time for the Cox model. The ODS name of the "Model Information" table is ModelInfo.

Number of Observations

The "Number of Observations" table displays the number of observations read and used in the analysis. The ODS name of the "Number of Observations" is NObs.

Class Level Information

The "Class Level Information" table is displayed when there are CLASS variables in the model. The table lists the categories of every CLASS variable that is used in the model and the corresponding design variable values. The ODS name of the "Class Level Information" table is ClassLevelInfo.

Class Level Information for Random Effects

The "Class Level Information for Random Effects" table is displayed when the RANDOM statement is specified. The table lists the categories of the classification variable specified in the RANDOM statement. The ODS name of the "Class Level Information for Random Effects" table is ClassLevelInfoR.

Summary of the Number of Event and Censored Values

The "Summary of the Number of Event and Censored Values" table displays, for each stratum, the breakdown of the number of events and censored values. The ODS name of the "Summary of the Number of Event and Censored Values" table is CensoredSummary.

Risk Sets Information

The "Risk Sets Information" table is displayed if you specify the ATRISK option in the PROC PHREG statement. The table displays, for each event time, the number of units at-risk and the number of units that experience the event. The ODS name of the "Risk Sets Information" table is RiskSetInfo.

Descriptive Statistics for Continuous Explanatory Variables

The "Simple Statistics for Continuous Explanatory Variables" table is displayed when you specify the SIMPLE option in the PROC PHREG statement. The table contains, for each stratum, the mean, standard deviation, and minimum and maximum for each continuous explanatory variable in the MODEL statement. The ODS name of the "Descriptive Statistics for Continuous Explanatory Variables" table is SimpleStatistics.

Frequency Distribution of CLASS Variables

The "Frequency Distribution of CLASS Variables" table is displayed if you specify the SIMPLE option in the PROC PHREG statement and there are CLASS variables in the model. The table lists the frequency of the levels of the CLASS variables. The ODS name of the "Frequency Distribution of CLASS Variables" table is ClassLevelFreq.

Maximum Likelihood Iteration History

The "Maximum Likelihood Iteration History" table is displayed if you specify the ITPRINT option in the MODEL statement. The table contains the iteration number, ridge value or step size, log likelihood, and parameter estimates at each iteration. The ODS name of the "Maximum Likelihood Iteration History" table is IterHistory.

Gradient of Last Iteration

The "Gradient of Last Iteration" table is displayed if you specify the ITPRINT option in the MODEL statement. The ODS name of the "Gradient of Last Iteration" table is LastGradient.

Convergence Status

The "Convergence Status" table displays the convergence status of the Newton-Raphson maximization. The ODS name of the "Convergence Status" table is ConvergenceStatus.

Model Fit Statistics

The "Model Fit Statistics" table displays the values of –2 log likelihood for the null model and the fitted model, the AIC, and SBC. The ODS name of the "Model Fit Statistics" table is FitStatistics.

Covariance Parameter Estimates

The "Covariance Parameter Estimates" table displays the estimate of the variance parameter of the random effect and the standard error estimate of the variance parameter estimator. The ODS name of the "Covariance Parameter Estimates" table is CovParms.

Testing Global Null Hypothesis: BETA=0

The "Testing Global Null Hypothesis: BETA=0" table displays results of the likelihood ratio test, the score test, and the Wald test for testing the hypothesis that all parameters are zero. For the frailty model, the score test is not displayed and an adjusted degrees of freedom is used (for more information, see the section "Wald-Type Tests for Penalized Models" on page 7609). For ODS purpose, the name of the "Testing Global Null Hypothesis: BETA=0" table is GlobalTests.

Likelihood Ratio Statistics for Type 1 Analysis

The "Likelihood Ratio Statistics for Type 1 Analysis" table is displayed if the TYPE1 option is specified in the MODEL statement. The table displays the degrees of freedom, the likelihood ratio chi-square statistic, and the p-value for each effect in the model. The ODS name of "Likelihood Ratio Statistics for Type 1 Analysis" is Type1.

Type 3 Tests

The "Type 3 Tests" table is displayed if the model contains a CLASS variable or if the TYPE3 option is specified in the MODEL statement. The table displays, for each specified statistic, the Type 3 chi-square, the degrees of freedom, and the *p*-value for each effect in the model. For the frailty model, the table also displays the adjusted Wald-type test results (for more information, see the section "Wald-Type Tests for Penalized Models" on page 7609). The ODS name of "Type 3 Tests" is Type3.

Analysis of Maximum Likelihood Estimates

The "Analysis of Maximum Likelihood Estimates" table displays the maximum likelihood estimate of the parameter; the estimated standard error, computed as the square root of the corresponding diagonal element of the estimated covariance matrix; the ratio of the robust standard error estimate to the model-based standard error estimate if you specify the COVS option in the PROC PHREG statement; the Wald Chi-Square statistic, computed as the square of the parameter estimate divided by its standard error estimate; the degrees of freedom of the Wald chi-square statistic, which has a value of 1 unless the corresponding parameter is redundant or infinite, in which case the value is 0; the *p*-value of the Wald chi-square statistic with respect to a chi-square distribution with one degree of freedom; the hazard ratio estimate; and the confidence limits for the hazard ratio if you specified the RISKLIMITS option in the MODEL statement. The ODS name of the "Analysis of Maximum Likelihood Estimates" table is ParameterEstimates.

Solution for Random Effects

The "Solution for Random Effects" table displays the BLUP estimates of the random effects, the estimated standard errors, the confidence intervals for the random effects, the exponentiated values of the BLUP estimates, and confidence intervals for the exponentiated random effects. The ODS name of the "Solution for Random Effects" table is SolutionR.

Regression Models Selected by Score Criterion

The "Regression Models Selected by Score Criterion" table is displayed if you specify SELECTION=SCORE in the MODEL statement. The table contains the number of explanatory variables in each model, the score chi-square statistic, and the names of the variables included in the model. The ODS name of the "Regression Models Selected by Score Criterion" table is BestSubsets.

Analysis of Effects Eligible for Entry

The "Analysis of Effects Eligible for Entry" table is displayed if you use the FORWARD or STEPWISE selection method and you specify the DETAILS option in the MODEL statement. The table contains the score chi-square statistic for testing the significance of each variable not in the model (after adjusting for the variables already in the model), and the *p*-value of the chi-square statistic with respect to a chi-square distribution with one degree of freedom. This table is produced before a variable is selected for entry in a forward selection step. The ODS name of the "Analysis of Effects Eligible for Entry" table is EffectsToEntry.

Analysis of Effects Eligible for Removal

The "Analysis of Effects Eligible for Removal" table is displayed if you use the BACKWARD or STEPWISE selection method and you specify the DETAILS option in the MODEL statement. The table contains the Wald chi-square statistic for testing the significance of each candidate effect for removal, the degrees of freedom of the Wald chi-square, and the corresponding *p*-value. This table is produced before an effect is selected for removal. The ODS name of the "Analysis of Effects Eligible for Removal" table is EffectsToRemoval.

Summary of Backward Elimination

The "Summary of Backward Elimination" table is displayed if you specify the SELECTION=BACKWARD option in the MODEL statement. The table contains the step number, the effects removed at each step, the corresponding chi-square statistic, the degrees of freedom, and the *p*-value. For ODS purpose, the name of the "Summary of Backward Elimination" table is ModelBuildingSummary.

Summary of Forward Selection

The "Summary of Forward Selection" table is displayed if you specify the SELECTION=FORWARD option in the MODEL statement. The table contains the step number, the effects entered at each step, the corresponding chi-square statistic, the degrees of freedom, and the *p*-value. For ODS purpose, the name of the "Summary of Forward Selection" table is ModelBuildingSummary.

Summary of Stepwise Selection

The "Summary of Stepwise Selection" table is displayed if you specify SELECTION=STEPWISE is specified in the MODEL statement. The table contains the step number, the effects entered or removed at each step, the corresponding chi-square statistic, the degrees of freedom, and the corresponding *p*-value. For ODS purpose, the name of the "Summary of Stepwise Selection" table is ModelBuildingSummary.

Covariance Matrix

The "Covariance Matrix" table is displayed if you specify the COVB option in the MODEL statement. The table contains the estimated covariance matrix for the parameter estimates. The ODS name of the "Covariance Matrix" table is CovB.

Correlation Matrix

The "Correlation Matrix" table is displayed if you specify the COVB option in the MODEL statement. The table contains the estimated correlation matrix for the parameter estimates. The ODS name of the "Correlation Matrix" table is CorrB.

Hazard Ratios for label

The "Hazard Ratios for *label*" table is displayed if you specify the HAZARDRATIO statement. The table displays the estimate and confidence limits for each hazard ratio. The ODS name of the "Hazard Ratios for *label*" table is HazardRatios.

Predictive Inaccuracy and Explained Variation

The "Predictive Inaccuracy and Explained Variation" table is displayed if you specify the EV option in the PROC PHREG statement. The table displays the predictive inaccuracy without covariates, the predictive inaccuracy with covariates, and the explained variation. If you specify the STRATA statement, the table also contains the stratum identification. The ODS name of the "Predictive Inaccuracy and Explained Variation" table is ExplainedVariation.

ZPH Tests of Nonproportional Hazards

The "ZPH Tests of Nonproportional Hazards" table is displayed if you specify the ZPH option in the PROC PHREG statement. For each parameter, the table displays the correlation between the time-varying coefficients and transformed times; the chi-square statistic and the corresponding *p*-value; and the *t* statistic and the corresponding *p*-value. The ODS name of the "ZPH Tests of Nonproportional Hazards" table is zphTest.

Coefficients of Contrast label

The "Coefficients of Contrast *label*" table is displayed if you specify the E option in the CONTRAST statement. The table displays the parameter names and the corresponding coefficients of each row of contrast *label*. The ODS name of the "Coefficients of Contrast *label*" table is ContrastCoeff.

Contrast Test Results

The "Contrast Test Results" table is displayed if you specify the CONTRAST statement. The table displays the degrees of freedom, test statistics, and the *p*-values for testing each contrast. The ODS name of the "Contrast Test Results" table is ContrastTest.

Contrast Estimation and Testing Results by Row

The "Contrast Estimation and Testing Results by Row" table is displayed if you specify the ESTIMATE option in the CONTRAST statement. The table displays, for each row, the estimate of the linear function of the coefficients, its standard error, and the confidence limits for the linear function. The ODS name of the "Contrast Estimation and Testing Results by Row" table is ContrastEstimate.

Linear Coefficients for label

The "Linear Coefficients *label*" table is displayed if you specify the E option in the TEST statement with *label* being the TEST statement label. The table contains the coefficients and constants of the linear hypothesis. The ODS name of the "Linear Coefficients for *label*" table is TestCoeff.

L[cov(b)]L' and Lb-c

The "L[cov(b)]L' and Lb-c" table is displayed if you specified the PRINT option in a TEST statement with *label* being the TEST statement label. The table displays the intermediate calculations of the Wald test. The ODS name of the "L[cov(b)]L' and Lb-c" table is TestPrint1.

Ginv(L[cov(b)]L') and Ginv(L[cov(b)]L')(Lb-c)

The "Ginv(L[cov(b)]L') and Ginv(L[cov(b)]L')(Lb-c)" table is displayed if you specified the PRINT option in a TEST statement with *label* being the TEST statement label. The table displays the intermediate calculations of the Wald test. The ODS name of the "Ginv(L[cov(b)]L') and Ginv(L[cov(b)]L')(Lb-c)" table is TestPrint2.

label Test Results

The "*label* Test Results" table is displayed if you specify a TEST statement with *label* being the TEST statement label. The table contains the Wald chi-square statistic, the degrees of freedom, and the *p*-value. The ODS name of "*label* Test Results" table is TestStmts.

Average Effect for label

The "Average Effect for *label*" table is displayed if the AVERAGE option is specified in a TEST statement with *label* being the TEST statement label. The table contains the weighted average of the parameter estimates for the variables in the TEST statement, the estimated standard error, the z-score, and the *p*-value. The ODS name of the "Average Effect for *label*" is TestAverage.

Reference Set of Covariates for Plotting

The "Reference Set of Covariates for Plotting" table is displayed if the PLOTS= option is requested without specifying the COVARIATES= data set in the BASELINE statement. The tables contains the values of the covariates for the reference set, where the reference levels are used for the CLASS variables and the sample averages for the continuous variables.

Harrell's Concordance Estimates

The "Harrell's Concordance Estimates" table is displayed if you specify CONCORDANCE=HARRELL in the PROC PHREG statement (or by default if you specify the CONCORDANCE option without specifying a method). The table displays the label of the identified model, the Harrell concordance estimate, the estimated standard error if the SE suboption is specified, the number of concordance pairs, the number of discordance pairs, the number of pairs that are tied in the predictor, and the number of pairs that are tied in time. The ODS name of the "Harrell's Concordance Estimates" table is Concordance.

Uno's Concordance Estimates

The "Uno's Concordance Estimates" table is displayed if you specify CONCORDANCE=UNO in the PROC PHREG statement. The table displays the label of the identified model, the Uno concordance estimate, and the estimated standard error if the SE suboption is specified. The ODS name of the "Uno's Concordance Estimates" table is Concordance.

Difference in Uno's Concordance Estimates

The "Difference in Uno's Concordance Estimates" table is displayed if you specify CONCORDANCE=UNO in the PROC PHREG statement and you specify both the SE and DIFF suboptions. The table identifies which pair of models is being compared and displays the difference of their concordance estimates, the estimated standard error of the difference, the Wald chi-square statistic (computed as the square of the difference divided by its standard error estimate), and the p-value of the Wald chi-square statistic with respect to a chi-square distribution with one degree of freedom. The ODS name of the "Difference in Uno's Concordance Estimates" table is ConcordanceDiff.

Time-Dependent Area under the Curve

The "Time-Dependent Area under the Curve" table is displayed if you specify the AUC suboption in the ROCOPTIONS option in the PROC PHREG statement. The table displays the label of the identified model, the time point at which the ROC curves are calculated, and the corresponding area under the ROC curve (AUC). If you specify METHOD=IPCW(CL) in the ROCOPTIONS option, the table also displays the estimated standard error and the confidence limits for the AUC. The ODS name of the "Time-Dependent Area under the Curve" table is AUC.

Differences in Time-Dependent Area under the Curve

The "Differences in Time-Dependent Area under the Curve" table is displayed if you specify the AUCDIFF suboption in the ROCOPTIONS option in the PROC PHREG statement. The table identifies which pair of models is being compared and displays the time point at which ROC curves are calculated and the difference in the AUC statistics. If you also specify METHOD=IPCW(CL) in ROCOPTIONS option, the table also displays the estimated standard error and the confidence limits for the difference in the AUC. The ODS name of the "Differences in Time-Dependent Area Under the Curve" table is AUCDiff.

Integrated Time-Dependent Area under the Curve

The "Integrated Time-Dependent Area under the Curve" table is displayed if you specify the IAUC suboption in the ROCOPTIONS option in the PROC PHREG statement. The table displays the label of the identified model and the integrated area under the curve (IAUC). The ODS name of the "Integrated Time-Dependent Area Under the Curve" table is IAUC.

Bayesian Analysis Displayed Output

Model Information

The "Model Information" table displays the two-level name of the input data set, the name and label of the failure time variable, the name and label of the censoring variable and the values indicating censored times, the model (either the Cox model or the piecewise constant baseline hazard model), the name and label of the OFFSET variable, the name and label of the FREQ variable, the name and label of the WEIGHT variable, the method of handling ties in the failure time, the number of burn-in iterations, the number of iterations after the burn-in, and the number of thinning iterations. The ODS name of the "Model Information" table is ModelInfo.

Number of Observations

The "Number of Observations" table displays the number of observations read and used in the analysis. The ODS name of the "Number of Observations" is NObs.

Class Level Information

The "Class Level Information" table is displayed if there are CLASS variables in the model. The table lists the categories of every CLASS variable in the model and the corresponding design variable values. The ODS name of the "Class Level Information" table is ClassLevelInfo.

Summary of the Number of Event and Censored Values

The "Summary of the Number of Event and Censored Values" table displays, for each stratum, the breakdown of the number of events and censored values. This table is not produced if the NONSUMMARY option is specified in the PROC PHREG statement. The ODS name of the "Summary of the Number of Event and Censored Values" table is CensoredSummary.

Descriptive Statistics for Continuous Explanatory Variables

The "Simple Statistics for Continuous Explanatory Variables" table is displayed when you specify the SIMPLE option in the PROC PHREG statement. The table contains, for each stratum, the mean, standard deviation, and minimum and maximum for each continuous explanatory variable in the MODEL statement. The ODS name of the "Descriptive Statistics for Continuous Explanatory Variables" table is SimpleStatistics.

Frequency Distribution of CLASS Variables

The "Frequency Distribution of CLASS Variables" table is displayed if you specify the SIMPLE option in the PROC PHREG statement and there are CLASS variables in the model. The table lists the frequency of the levels of the CLASS variables. The ODS name of the "Frequency Distribution of CLASS Variables" table is ClassLevelFreq.

Cubic Spline Parameter Information

The "Cubic Spline Parameter Information" table is displayed if you specify the SPLINE option in the BAYES statement. The table lists the name of the cubic spline parameters and the position of the knots. The ODS name of the "Cubic Spline Parameter Information" table is SplineKnots.

Regression Parameter Information

The "Regression Parameter Information" table displays the names of the parameters and the corresponding level information of effects containing the CLASS variables. The ODS name of the "Regression Parameter Information" table is ParmInfo.

Constant Baseline Hazard Time Intervals

The "Constant Baseline Hazard Time Intervals" table displays the intervals of constant baseline hazard and the corresponding numbers of failure times and event times. This table is produced only if you specify the PIECEWISE option in the BAYES statement. The ODS name of the "Constant Baseline Hazard Time Intervals" table is Interval.

Maximum Likelihood Estimates

The "Maximum Likelihood Estimates" table displays, for each parameter, the maximum likelihood estimate, the estimated standard error, and the 95% confidence limits. The ODS name of the "Maximum Likelihood Estimates" table is ParameterEstimates.

Hazard Prior

The "Hazard Prior" table is displayed if you specify the PIECEWISE=HAZARD option in the BAYES statement. It describes the prior distribution of the hazard parameters. The ODS name of the "Hazard Prior" table is HazardPrior.

Log-Hazard Prior

The "Log-Hazard Prior" table is displayed if you specify the PIECEWISE=LOGHAZARD option in the BAYES statement. It describes the prior distribution of the log-hazard parameters. The ODS name of the "Log-Hazard Prior" table is HazardPrior.

Spline Parameter Prior

The "Spline Parameter Prior" table is displayed if you specify the SPLINE option in the BAYES statement. It describes the prior distribution of the cubic spline parameters for the baseline hazard function. The ODS name of the "Spline Parameter Prior" table is HazardPrior.

Coefficient Prior

The "Coefficient Prior" table displays the prior distribution of the regression coefficients. The ODS name of the "Coefficient Prior" table is CoeffPrior.

Initial Values

The "Initial Values" table is displayed if you specify the INITIAL option in the BAYES statement. The table contains the initial values of the parameters for the Gibbs sampling. The ODS name of the "Initial Values" table is InitialValues.

Fit Statistics

The "Fit Statistics" table displays the DIC and pD statistics for each parameter. The ODS name of the "Fit Statistics" table is FitStatistics.

Posterior Summaries

The "Posterior Summaries" table displays the size of the posterior sample, the mean, the standard error, and the percentiles for each model parameter. The ODS name of the "Posterior Summaries" table is PostSummaries.

Posterior Intervals

The "Posterior Intervals" table displays the equal-tail interval and the HPD interval for each model parameter. The ODS name of the "Posterior Intervals" table is PostIntervals.

Posterior Covariance Matrix

The "Posterior Covariance Matrix" table is produced if you include COV in the SUMMARY= option in the BAYES statement. This tables displays the sample covariance of the posterior samples. The ODS name of the "Posterior Covariance Matrix" table is Cov.

Posterior Correlation Matrix

The "Posterior Correlation Matrix" table is displayed if you include CORR in the SUMMARY= option in the BAYES statement. The table contains the sample correlation of the posterior samples. The ODS name of the "Posterior Correlation Matrix" table is Corr.

Posterior Autocorrelations

The "Posterior Autocorrelations" table displays the lag 1, lag 5, lag 10, and lag 50 autocorrelations for each parameter. The ODS name of the "Posterior Autocorrelations" table is AutoCorr.

Gelman-Rubin Diagnostics

The "Gelman-Rubin Diagnostics" table is produced if you include GELMAN in the DIAGNOSTIC= option in the BAYES statement. This table displays the estimate of the potential scale reduction factor and its 97.5% upper confidence limit for each parameter. The ODS name of the "Gelman-Rubin Diagnostics" table is Gelman.

Geweke Diagnostics

The "Geweke Diagnostics" table displays the Geweke statistic and its p-value for each parameter. The ODS name of the "Geweke Diagnostics" table is Geweke.

Raftery-Lewis Diagnostics

The "Raftery-Lewis Diagnostics" tables is produced if you include RAFTERY in the DIAGNOSTIC= option in the BAYES statement. This table displays the Raftery and Lewis diagnostics for each variable. The ODS name of the "Raftery-Diagnostics" table is "Raftery."

Heidelberger-Welch Diagnostics

The "Heidelberger-Welch Diagnostics" table is displayed if you include HEIDELBERGER in the DIAGNOSTIC= option in the BAYES statement. This table describes the results of a stationary test and a halfwidth test for each parameter. The ODS name of the "Heidelberger-Welch Diagnostics" table is Heidelberger.

Effective Sample Sizes

The "Effective Sample Sizes" table displays, for each parameter, the effective sample size, the correlation time, and the efficiency. The ODS name of the "Effective Sample Sizes" table is ESS.

Hazard Ratios for label

The "Hazard Ratios for *label*" table is displayed if you specify the HAZARDRATIO statement. The table displays the posterior summary for each hazard ratio. The summary includes the mean, standard error, quartiles, and equal-tailed and HPD intervals. The ODS name of the "Hazard Ratios for *label*" table is HazardRatios.

Reference Set of Covariates for Plotting

The "Reference Set of Covariates for Plotting" table is displayed if the PLOTS= option is requested without specifying the COVARIATES= data set in the BASELINE statement. The table contains the values of the covariates for the reference set, where the reference levels are used for the CLASS variables and the sample averages for the continuous variables.

ODS Table Names

PROC PHREG assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed separately in Table 92.17 for the maximum likelihood analysis and in Table 92.18 for the Bayesian analysis. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Each of the EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements creates ODS tables, which are not listed in Table 92.17 and Table 92.18. For information about these tables, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Table 92.17 ODS Tables for a Maximum Likelihood Analysis Produced by PROC PHREG

ODS Table Name	Description	Statement	Option
AUC	Area under the ROC curve at specific time points	PROC	ROCOPTIONS(AUC)
AUCDiff	Differences in AUC between models	PROC	ROCOPTIONS(AUCDIFF)
BestSubsets	Best subset selection	MODEL	SELECTION=SCORE
CensoredSummary	Summary of event and censored observations	MODEL	Default
ClassLevelFreq	Frequency distribution of CLASS variables	PROC	SIMPLE (with CLASS vars)
ClassLevelInfo	CLASS variable levels and design variables	CLASS	Default
ClassLevelInfoR	Class levels for random effects	RANDOM	Default
ContrastCoeff	L matrix for contrasts	CONTRAST	E
ContrastEstimate	Individual contrast estimates	CONTRAST	ESTIMATE=
ContrastTest	Wald test for contrasts	CONTRAST	Default
ConvergenceStatus	Convergence status	MODEL	Default
CorrB	Estimated correlation matrix of parameter estimators	MODEL	CORRB
CovB	Estimated covariance matrix of parameter estimators	MODEL	COVB
CovParms	Variance estimates of the random effects	RANDOM	Default
EffectsToEnter	Analysis of effects for entry	MODEL	SELECTION=F|S
EffectsToRemove	Analysis of effects for removal	MODEL	SELECTION=B|S

Table 92.17 *continued*

ODS Table Name	Description	Statement	Option
ExplainedVariation	Schemper-Henderson predictive accuracy and explained variation	PROC	EV
FitStatistics	Model fit statistics	MODEL	Default
FunctionalFormSupTest	Supremum test for functional form	ASSESS	VAR=
GlobalScore	Global chi-square test	MODEL	NOFIT
GlobalTests	Tests of the global null hypothesis	MODEL	Default
HazardRatios	Hazard ratios and confidence limits	HAZARDRATIO	Default
IAUC	Integrated area under the curve	PROC	ROCOPTONS(IAUC)
IterHistory	Iteration history	MODEL	ITPRINT
LastGradient	Last evaluation of gradient	MODEL	ITPRINT
ModelANOVA	Type 3 tests or joint tests	MODEL	TYPE3 | CLASS
ModelBuildingSummary	Summary of model building	MODEL	SELECTION=B|F|S
ModelInfo	Model information	PROC	Default
NObs	Number of observations	PROC	Default
ParameterEstimates	Maximum likelihood estimates of model parameters	MODEL	Default
ProportionalHazardsSupTest	Supremum test for proportional hazards assumption	ASSESS	PH
ResidualChiSq	Residual chi-square	MODEL	SELECTION=F|B
ReferenceSet	Reference set of covariates for plotting	PROC	PLOTS=
RiskSetInfo	Risk set information	PROC	ATRISK
SimpleStatistics	Summary statistics of input continuous explanatory variables	PROC	SIMPLE
SolutionR	Solutions for random effects	RANDOM	SOLUTION
TestAverage	Average effect for test	TEST	AVERAGE
TestCoeff	Coefficients for linear hypotheses	TEST	E
TestPrint1	**L**[cov(**b**)]**L**' and **Lb-c**	TEST	PRINT
TestPrint2	Ginv(**L**[cov(**b**)]**L**') and Ginv(**L**[cov(**b**)]**L**')(**Lb-c**)	TEST	PRINT
TestStmts	Linear hypotheses testing results	TEST	Default
Type1	Type 1 likelihood ratio tests	MODEL	TYPE1
zphTest	Proportional hazards assumption tests based on scaled Schoenfeld residuals	PROC	ZPH

Table 92.18 ODS Table for a Bayesian Analysis Produced by PROC PHREG

ODS Table Name	Description	Statement	Option
AutoCorr	Autocorrelations of the posterior samples	BAYES	DIAGNOSTICS=AUTOCORR
CensoredSummary	Numbers of the event and censored observations	MODEL	default
ClassLevelFreq	Frequency distribution of CLASS variables	PROC	SIMPLE (with CLASS vars)
ClassLevelInfo	CLASS variable levels and design variables	CLASS	Default
CoeffPrior	Prior distribution of the regression coefficients	BAYES	Default
Corr	Posterior correlation matrix	BAYES	STATISTICS=CORR
Cov	Posterior covariance Matrix	BAYES	STATISTICS=COV
ESS	Effective sample sizes	BAYES	DIAGNOSTICS=ESS
FitStatistics	Fit statistics	BAYES	Default
Gelman	Gelman-Rubin convergence diagnostics	BAYES	DIAGNOSTICS=GELMAN
Geweke	Geweke convergence diagnostics	BAYES	DIAGNOSTICS=GEWEKE
HazardPrior	Prior distribution of the baseline hazards	BAYES	PIECEWISE | SPLINE
HazardRatios	Posterior summary statistics for hazard ratios	HAZARDRATIO	Default
Heidelberger	Heidelberger-Welch convergence diagnostics	BAYES	DIAGNOSTICS=HEIDELBERGER
InitialValues	Initial values of the Markov chains	BAYES	Default
MCError	Monte Carlo standard errors	BAYES	DIAGNOSTICS=MCERROR
ModelInfo	Model information	PROC	Default
NObs	Number of observations	PROC	Default
ParameterEstimates	Maximum likelihood estimates of model parameters	PROC	Default
ParmInfo	Names of regression coefficients	BAYES	Default
Partition	Partition of constant baseline hazard intervals	BAYES	PIECEWISE
PosteriorSample	Posterior samples	BAYES	(for ODS output data set only)
PostIntervals	Equal-tail and high probability density intervals of the posterior	BAYES	STATISTICS=INTERVAL
PostSumInt	Posterior summary statistics and HPD intervals	BAYES	(STATISTICS= not specified)
PostSummaries	Summary statistics of the posterior samples	BAYES	STATISTICS=SUMMARY
Raftery	Raftery-Lewis convergence diagnostics	BAYES	DIAGNOSTICS=RAFTERY

Table 92.18 *continued*

ODS Table Name	Description	Statement	Option
ReferenceSet	Reference set of covariates for plotting	PROC	PLOTS=
SimpleStatistics	Summary statistics of input continuous explanatory variables	PROC	SIMPLE
SplineKnots	Spline parameter names and knot positions	BAYES	SPLINE

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC PHREG generates are listed separately in Table 92.19 for the maximum likelihood analysis and in Table 92.20 for the Bayesian analysis. When the ODS Graphics are in effect in a Bayesian analysis, each of the ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements can produce plots associated with their analyses. For information of these plots, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Table 92.19 Graphs for a Maximum Likelihood Analysis Produced by PROC PHREG

ODS Graph Name	Plot Description	Statement	Option
AUCPlot	Area under the curve plot	PROC	PLOTS=AUC
AUCDiffPlot	Area under the curve difference plot	PROC	PLOTS=AUCDIFF
CIFPLOT	Cumulative incidence function plot	PROC	PLOTS=CIF
CumhazPlot	Cumulative hazard function plot	PROC	PLOTS=CUMHAZ
CumulativeResiduals	Cumulative martingale residual plot	ASSESS	VAR=
CumResidPanel	Panel plot of cumulative martingale residuals	ASSESS	VAR= & CRPANEL
MCFPlot	Mean cumulative function plot	PROC	PLOTS=MCF

Table 92.19 *continued*

ODS Graph Name	Plot Description	Statement	Option
ROCPanel	Receiver operating characteristic plots in panels	PROC	PLOTS=ROC
ROCPlot	Receiver operating characteristic plot	PROC	PLOTS(OVERLAY=IND)=ROC
ScoreProcess	Standardized score process plot	ASSESS	PH
SurvivalPlot	Survivor function plot	PROC	PLOTS=SURVIVAL

Table 92.20 Graphs for a Bayesian Analysis Produced by PROC PHREG

ODS Graph Name	Plot Description	Statement	Option
ADPanel	Autocorrelation function and density panel	BAYES	PLOTS=(AUTOCORR DENSITY)
AutocorrPanel	Autocorrelation function panel	BAYES	PLOTS= AUTOCORR
AutocorrPlot	Autocorrelation function plot	BAYES	PLOTS(UNPACK)=AUTOCORR
CumhazPlot	Cumulative hazard function plot	PROC	PLOTS=CUMHAZ
DensityPanel	Density panel	BAYES	PLOTS=DENSITY
DensityPlot	Density plot	BAYES	PLOTS(UNPACK)=DENSITY
SurvivalPlot	Survivor function plot	PROC	PLOTS=SURVIVAL
TAPanel	Trace and autocorrelation function panel	BAYES	PLOTS=(TRACE AUTOCORR)
TADPanel	Trace, density, and autocorrelation function panel	BAYES	PLOTS=(TRACE AUTOCORR DENSITY)
TDPanel	Trace and density panel	BAYES	PLOTS=(TRACE DENSITY)
TracePanel	Trace panel	BAYES	PLOTS=TRACE
TracePlot	Trace plot	BAYES	PLOTS(UNPACK)=TRACE

Examples: PHREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

This section contains 16 examples of using PROC PHREG. Example 92.13 and Example 92.14 illustrate Bayesian methodology, and the other examples use the classical method of maximum likelihood.

Example 92.1: Stepwise Regression

(View the complete code for this example (phrex1.sas) in the example repository.)

Krall, Uthoff, and Harley (1975) analyzed data from a study on multiple myeloma in which researchers treated 65 patients with alkylating agents. Of those patients, 48 died during the study and 17 survived. The following DATA step creates the data set Myeloma. The variable Time represents the survival time in months from diagnosis. The variable VStatus consists of two values, 0 and 1, indicating whether the patient was alive or dead, respectively, at the end of the study. If the value of VStatus is 0, the corresponding value of Time is censored. The variables thought to be related to survival are LogBUN (log(BUN) at diagnosis), HGB (hemoglobin at diagnosis), Platelet (platelets at diagnosis: 0=abnormal, 1=normal), Age (age at diagnosis, in years), LogWBC (log(WBC) at diagnosis), Frac (fractures at diagnosis: 0=none, 1=present), LogPBM (log percentage of plasma cells in bone marrow), Protein (proteinuria at diagnosis), and SCalc (serum calcium at diagnosis). Interest lies in identifying important prognostic factors from these nine explanatory variables.

```
data Myeloma;
   input Time VStatus LogBUN HGB Platelet Age LogWBC Frac
         LogPBM Protein SCalc;
   label Time='Survival Time'
         VStatus='0=Alive 1=Dead';
   datalines;
 1.25  1  2.2175   9.4  1  67  3.6628  1  1.9542  12  10
 1.25  1  1.9395  12.0  1  38  3.9868  1  1.9542  20  18
 2.00  1  1.5185   9.8  1  81  3.8751  1  2.0000   2  15
 2.00  1  1.7482  11.3  0  75  3.8062  1  1.2553   0  12
 2.00  1  1.3010   5.1  0  57  3.7243  1  2.0000   3   9
 3.00  1  1.5441   6.7  1  46  4.4757  0  1.9345  12  10
 5.00  1  2.2355  10.1  1  50  4.9542  1  1.6628   4   9
 5.00  1  1.6812   6.5  1  74  3.7324  0  1.7324   5   9
 6.00  1  1.3617   9.0  1  77  3.5441  0  1.4624   1   8
 6.00  1  2.1139  10.2  0  70  3.5441  1  1.3617   1   8
 6.00  1  1.1139   9.7  1  60  3.5185  1  1.3979   0  10
 6.00  1  1.4150  10.4  1  67  3.9294  1  1.6902   0   8
 7.00  1  1.9777   9.5  1  48  3.3617  1  1.5682   5  10
 7.00  1  1.0414   5.1  0  61  3.7324  1  2.0000   1  10
 7.00  1  1.1761  11.4  1  53  3.7243  1  1.5185   1  13
 9.00  1  1.7243   8.2  1  55  3.7993  1  1.7404   0  12
11.00  1  1.1139  14.0  1  61  3.8808  1  1.2788   0  10
11.00  1  1.2304  12.0  1  43  3.7709  1  1.1761   1   9
11.00  1  1.3010  13.2  1  65  3.7993  1  1.8195   1  10
11.00  1  1.5682   7.5  1  70  3.8865  0  1.6721   0  12
```

```
11.00  1  1.0792   9.6  1  51  3.5051  1  1.9031   0   9
13.00  1  0.7782   5.5  0  60  3.5798  1  1.3979   2  10
14.00  1  1.3979  14.6  1  66  3.7243  1  1.2553   2  10
15.00  1  1.6021  10.6  1  70  3.6902  1  1.4314   0  11
16.00  1  1.3424   9.0  1  48  3.9345  1  2.0000   0  10
16.00  1  1.3222   8.8  1  62  3.6990  1  0.6990  17  10
17.00  1  1.2304  10.0  1  53  3.8808  1  1.4472   4   9
17.00  1  1.5911  11.2  1  68  3.4314  0  1.6128   1  10
18.00  1  1.4472   7.5  1  65  3.5682  0  0.9031   7   8
19.00  1  1.0792  14.4  1  51  3.9191  1  2.0000   6  15
19.00  1  1.2553   7.5  0  60  3.7924  1  1.9294   5   9
24.00  1  1.3010  14.6  1  56  4.0899  1  0.4771   0   9
25.00  1  1.0000  12.4  1  67  3.8195  1  1.6435   0  10
26.00  1  1.2304  11.2  1  49  3.6021  1  2.0000  27  11
32.00  1  1.3222  10.6  1  46  3.6990  1  1.6335   1   9
35.00  1  1.1139   7.0  0  48  3.6532  1  1.1761   4  10
37.00  1  1.6021  11.0  1  63  3.9542  0  1.2041   7   9
41.00  1  1.0000  10.2  1  69  3.4771  1  1.4771   6  10
41.00  1  1.1461   5.0  1  70  3.5185  1  1.3424   0   9
51.00  1  1.5682   7.7  0  74  3.4150  1  1.0414   4  13
52.00  1  1.0000  10.1  1  60  3.8573  1  1.6532   4  10
54.00  1  1.2553   9.0  1  49  3.7243  1  1.6990   2  10
58.00  1  1.2041  12.1  1  42  3.6990  1  1.5798  22  10
66.00  1  1.4472   6.6  1  59  3.7853  1  1.8195   0   9
67.00  1  1.3222  12.8  1  52  3.6435  1  1.0414   1  10
88.00  1  1.1761  10.6  1  47  3.5563  0  1.7559  21   9
89.00  1  1.3222  14.0  1  63  3.6532  1  1.6232   1   9
92.00  1  1.4314  11.0  1  58  4.0755  1  1.4150   4  11
 4.00  0  1.9542  10.2  1  59  4.0453  0  0.7782  12  10
 4.00  0  1.9243  10.0  1  49  3.9590  0  1.6232   0  13
 7.00  0  1.1139  12.4  1  48  3.7993  1  1.8573   0  10
 7.00  0  1.5315  10.2  1  81  3.5911  0  1.8808   0  11
 8.00  0  1.0792   9.9  1  57  3.8325  1  1.6532   0   8
12.00  0  1.1461  11.6  1  46  3.6435  0  1.1461   0   7
11.00  0  1.6128  14.0  1  60  3.7324  1  1.8451   3   9
12.00  0  1.3979   8.8  1  66  3.8388  1  1.3617   0   9
13.00  0  1.6628   4.9  0  71  3.6435  0  1.7924   0   9
16.00  0  1.1461  13.0  1  55  3.8573  0  0.9031   0   9
19.00  0  1.3222  13.0  1  59  3.7709  1  2.0000   1  10
19.00  0  1.3222  10.8  1  69  3.8808  1  1.5185   0  10
28.00  0  1.2304   7.3  1  82  3.7482  1  1.6721   0   9
41.00  0  1.7559  12.8  1  72  3.7243  1  1.4472   1   9
53.00  0  1.1139  12.0  1  66  3.6128  1  2.0000   1  11
57.00  0  1.2553  12.5  1  66  3.9685  0  1.9542   0  11
77.00  0  1.0792  14.0  1  60  3.6812  0  0.9542   0  12
;
```

The stepwise selection process consists of a series of alternating forward selection and backward elimination steps. The former adds variables to the model, while the latter removes variables from the model.

The following statements use PROC PHREG to produce a stepwise regression analysis. Stepwise selection is requested by specifying the SELECTION=STEPWISE option in the MODEL statement. The option SLENTRY=0.25 specifies that a variable has to be significant at the 0.25 level before it can be entered into the model, while the option SLSTAY=0.15 specifies that a variable in the model has to be significant at

the 0.15 level for it to remain in the model. The DETAILS option requests detailed results for the variable selection process.

```
proc phreg data=Myeloma;
   model Time*VStatus(0)=LogBUN HGB Platelet Age LogWBC
                         Frac LogPBM Protein SCalc
                         / selection=stepwise slentry=0.25
                           slstay=0.15 details;
run;
```

Results of the stepwise regression analysis are displayed in Output 92.1.1 through Output 92.1.7.

Individual score tests are used to determine which of the nine explanatory variables is first selected into the model. In this case, the score test for each variable is the global score test for the model containing that variable as the only explanatory variable. Output 92.1.1 displays the chi-square statistics and the corresponding p-values. The variable LogBUN has the largest chi-square value (8.5164), and it is significant ($p = 0.0035$) at the SLENTRY=0.25 level. The variable LogBUN is thus entered into the model.

Output 92.1.1 Individual Score Test Results for All Variables

The PHREG Procedure

Model Information		
Data Set	WORK.MYELOMA	
Dependent Variable	Time	Survival Time
Censoring Variable	VStatus	0=Alive 1=Dead
Censoring Value(s)	0	
Ties Handling	BRESLOW	

Summary of the Number of Event and Censored Values			
Total	**Event**	**Censored**	**Percent Censored**
65	48	17	26.15

Analysis of Effects Eligible for Entry			
Effect	**DF**	**Score Chi-Square**	**Pr > ChiSq**
LogBUN	1	8.5164	0.0035
HGB	1	5.0664	0.0244
Platelet	1	3.1816	0.0745
Age	1	0.0183	0.8924
LogWBC	1	0.5658	0.4519
Frac	1	0.9151	0.3388
LogPBM	1	0.5846	0.4445
Protein	1	0.1466	0.7018
SCalc	1	1.1109	0.2919

Residual Chi-Square Test		
Chi-Square	**DF**	**Pr > ChiSq**
18.4550	9	0.0302

Output 92.1.2 displays the results of the first model. Since the Wald chi-square statistic is significant ($p =$

0.0039) at the SLSTAY=0.15 level, LogBUN stays in the model.

Output 92.1.2 First Model in the Stepwise Selection Process

Step 1. Effect LogBUN is entered. The model contains the following effects:

LogBUN

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Without Covariates	With Covariates
-2 LOG L	309.716	301.959
AIC	309.716	303.959
SBC	309.716	305.830

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	7.7572	1	0.0053
Score	8.5164	1	0.0035
Wald	8.3392	1	0.0039

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
LogBUN	1	1.74595	0.60460	8.3392	0.0039	5.731

The next step consists of selecting another variable to add to the model. Output 92.1.3 displays the chi-square statistics and p-values of individual score tests (adjusted for LogBUN) for the remaining eight variables. The score chi-square for a given variable is the value of the likelihood score test for testing the significance of the variable in the presence of LogBUN. The variable HGB is selected because it has the highest chi-square value (4.3468), and it is significant ($p = 0.0371$) at the SLENTRY=0.25 level.

Output 92.1.3 Score Tests Adjusted for the Variable LogBUN

Analysis of Effects Eligible for Entry			
Effect	DF	Score Chi-Square	Pr > ChiSq
HGB	1	4.3468	0.0371
Platelet	1	2.0183	0.1554
Age	1	0.7159	0.3975
LogWBC	1	0.0704	0.7908
Frac	1	1.0354	0.3089
LogPBM	1	1.0334	0.3094
Protein	1	0.5214	0.4703
SCalc	1	1.4150	0.2342

Residual Chi-Square Test		
Chi-Square	DF	Pr > ChiSq
9.3164	8	0.3163

Output 92.1.4 displays the fitted model containing both LogBUN and HGB. Based on the Wald statistics, neither LogBUN nor HGB is removed from the model.

Output 92.1.4 Second Model in the Stepwise Selection Process

Step 2. Effect HGB is entered. The model contains the following effects:

LogBUN HGB

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Without Covariates	With Covariates
-2 LOG L	309.716	297.767
AIC	309.716	301.767
SBC	309.716	305.509

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	11.9493	2	0.0025
Score	12.7252	2	0.0017
Wald	12.1900	2	0.0023

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
LogBUN	1	1.67440	0.61209	7.4833	0.0062	5.336
HGB	1	-0.11899	0.05751	4.2811	0.0385	0.888

Output 92.1.5 shows Step 3 of the selection process, in which the variable SCalc is added, resulting in the model with LogBUN, HGB, and SCalc as the explanatory variables. Note that SCalc has the smallest Wald chi-square statistic, and it is not significant ($p = 0.1782$) at the SLSTAY=0.15 level.

Output 92.1.5 Third Model in the Stepwise Regression

Step 3. Effect SCalc is entered. The model contains the following effects:

LogBUN HGB SCalc

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Without Covariates	With Covariates
-2 LOG L	309.716	296.078
AIC	309.716	302.078
SBC	309.716	307.692

Output 92.1.5 *continued*

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	13.6377	3	0.0034
Score	15.3053	3	0.0016
Wald	14.4542	3	0.0023

Analysis of Maximum Likelihood Estimates						
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**
LogBUN	1	1.63593	0.62359	6.8822	0.0087	5.134
HGB	1	-0.12643	0.05868	4.6419	0.0312	0.881
SCalc	1	0.13286	0.09868	1.8127	0.1782	1.142

The variable SCalc is then removed from the model in a step-down phase in Step 4 (Output 92.1.6). The removal of SCalc brings the stepwise selection process to a stop in order to avoid repeatedly entering and removing the same variable.

Output 92.1.6 Final Model in the Stepwise Regression

Step 4. Effect SCalc is removed. The model contains the following effects:

LogBUN HGB

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	**Without Covariates**	**With Covariates**
-2 LOG L	309.716	297.767
AIC	309.716	301.767
SBC	309.716	305.509

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	11.9493	2	0.0025
Score	12.7252	2	0.0017
Wald	12.1900	2	0.0023

Analysis of Maximum Likelihood Estimates						
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**
LogBUN	1	1.67440	0.61209	7.4833	0.0062	5.336
HGB	1	-0.11899	0.05751	4.2811	0.0385	0.888

Note: Model building terminates because the effect to be entered is the effect that was removed in the last step.

The procedure also displays a summary table of the steps in the stepwise selection process, as shown in Output 92.1.7.

Output 92.1.7 Model Selection Summary

Summary of Stepwise Selection							
	Effect						
Step	Entered	Removed	DF	Number In	Score Chi-Square	Wald Chi-Square	Pr > ChiSq
1	LogBUN		1	1	8.5164		0.0035
2	HGB		1	2	4.3468		0.0371
3	SCalc		1	3	1.8225		0.1770
4		SCalc	1	2		1.8127	0.1782

The stepwise selection process results in a model with two explanatory variables, LogBUN and HGB.

Example 92.2: Best Subset Selection

(View the complete code for this example (phrex2.sas) in the example repository.)

An alternative to stepwise selection of variables is best subset selection. This method uses the branch-and-bound algorithm of Furnival and Wilson (1974) to find a specified number of best models containing one, two, or three variables, and so on, up to the single model containing all of the explanatory variables. The criterion used to determine the "best" subset is based on the global score chi-square statistic. For two models A and B, each having the same number of explanatory variables, model A is considered to be better than model B if the global score chi-square statistic for A exceeds that for B.

In the following statements, best subset selection analysis is requested by specifying the SELECTION=SCORE option in the MODEL statement. The BEST=3 option requests the procedure to identify only the three best models for each size. In other words, PROC PHREG will list the three models having the highest score statistics of all the models possible for a given number of covariates.

```
proc phreg data=Myeloma;
   model Time*VStatus(0)=LogBUN HGB Platelet Age LogWBC
                         Frac LogPBM Protein SCalc
                         / selection=score best=3;
run;
```

Output 92.2.1 displays the results of this analysis. The number of explanatory variables in the model is given in the first column, and the names of the variables are listed on the right. The models are listed in descending order of their score chi-square values within each model size. For example, among all models containing two explanatory variables, the model that contains the variables LogBUN and HGB has the largest score value (12.7252), the model that contains the variables LogBUN and Platelet has the second-largest score value (11.1842), and the model that contains the variables LogBUN and SCalc has the third-largest score value (9.9962).

Output 92.2.1 Best Variable Combinations

The PHREG Procedure

Regression Models Selected by Score Criterion		
Number of Variables	Score Chi-Square	Variables Included in Model
1	8.5164	LogBUN
1	5.0664	HGB
1	3.1816	Platelet
2	12.7252	LogBUN HGB
2	11.1842	LogBUN Platelet
2	9.9962	LogBUN SCalc
3	15.3053	LogBUN HGB SCalc
3	13.9911	LogBUN HGB Age
3	13.5788	LogBUN HGB Frac
4	16.9873	LogBUN HGB Age SCalc
4	16.0457	LogBUN HGB Frac SCalc
4	15.7619	LogBUN HGB LogPBM SCalc
5	17.6291	LogBUN HGB Age Frac SCalc
5	17.3519	LogBUN HGB Age LogPBM SCalc
5	17.1922	LogBUN HGB Age LogWBC SCalc
6	17.9120	LogBUN HGB Age Frac LogPBM SCalc
6	17.7947	LogBUN HGB Age LogWBC Frac SCalc
6	17.7744	LogBUN HGB Platelet Age Frac SCalc
7	18.1517	LogBUN HGB Platelet Age Frac LogPBM SCalc
7	18.0568	LogBUN HGB Age LogWBC Frac LogPBM SCalc
7	18.0223	LogBUN HGB Platelet Age LogWBC Frac SCalc
8	18.3925	LogBUN HGB Platelet Age LogWBC Frac LogPBM SCalc
8	18.1636	LogBUN HGB Platelet Age Frac LogPBM Protein SCalc
8	18.1309	LogBUN HGB Platelet Age LogWBC Frac Protein SCalc
9	18.4550	LogBUN HGB Platelet Age LogWBC Frac LogPBM Protein SCalc

Example 92.3: Modeling with Categorical Predictors

(View the complete code for this example (phrex3.sas) in the example repository.)

Consider the data for the Veterans Administration lung cancer trial presented in Appendix 1 of Kalbfleisch and Prentice (1980). In this trial, males with advanced inoperable lung cancer were randomized to a standard therapy and a test chemotherapy. The primary endpoint for the therapy comparison was time to death in days, represented by the variable Time. Negative values of Time are censored values. The data include information about a number of explanatory variables: Therapy (type of therapy: standard or test), Cell (type of tumor cell: adeno, large, small, or squamous), Prior (prior therapy: 0=no, 10=yes), Age (age, in years), Duration (months from diagnosis to randomization), and Kps (Karnofsky performance scale). A censoring indicator variable, Censor, is created from the data, with the value 1 indicating a censored time and the value 0 indicating an event time. The following DATA step saves the data in the data set VALung.

```
proc format;
   value yesno 0='no' 10='yes';
run;

data VALung;
   drop check m;
   retain Therapy Cell;
   infile cards column=column;
   length Check $ 1;
   label Time='time to death in days'
         Kps='Karnofsky performance scale'
         Duration='months from diagnosis to randomization'
         Age='age in years'
         Prior='prior therapy'
         Cell='cell type'
         Therapy='type of treatment';
   format Prior yesno.;
   M=Column;
   input Check $ @@;
   if M>Column then M=1;
   if Check='s'|Check='t' then do;
      input @M Therapy $ Cell $;
      delete;
   end;
   else do;
      input @M Time Kps Duration Age Prior @@;
      Status=(Time>0);
      Time=abs(Time);
   end;
   datalines;
standard squamous
 72 60  7 69  0   411 70  5 64 10   228 60  3 38  0   126 60  9 63 10
118 70 11 65 10    10 20  5 49  0    82 40 10 69 10   110 80 29 68  0
314 50 18 43  0  -100 70  6 70  0    42 60  4 81  0     8 40 58 63 10
144 30  4 63  0   -25 80  9 52 10    11 70 11 48 10
standard small
 30 60  3 61  0   384 60  9 42  0     4 40  2 35  0    54 80  4 63 10
 13 60  4 56  0  -123 40  3 55  0   -97 60  5 67  0   153 60 14 63 10
 59 30  2 65  0   117 80  3 46  0    16 30  4 53 10   151 50 12 69  0
 22 60  4 68  0    56 80 12 43 10    21 40  2 55 10    18 20 15 42  0
139 80  2 64  0    20 30  5 65  0    31 75  3 65  0    52 70  2 55  0
287 60 25 66 10    18 30  4 60  0    51 60  1 67  0   122 80 28 53  0
 27 60  8 62  0    54 70  1 67  0     7 50  7 72  0    63 50 11 48  0
392 40  4 68  0    10 40 23 67 10
standard adeno
  8 20 19 61 10    92 70 10 60  0    35 40  6 62  0   117 80  2 38  0
132 80  5 50  0    12 50  4 63 10   162 80  5 64  0     3 30  3 43  0
 95 80  4 34  0
standard large
177 50 16 66 10   162 80  5 62  0   216 50 15 52  0   553 70  2 47  0
278 60 12 63  0    12 40 12 68 10   260 80  5 45  0   200 80 12 41 10
156 70  2 66  0  -182 90  2 62  0   143 90  8 60  0   105 80 11 66  0
103 80  5 38  0   250 70  8 53 10   100 60 13 37 10
```

```
test squamous
999 90 12 54 10   112 80  6 60  0   -87 80  3 48  0  -231 50  8 52 10
242 50  1 70  0   991 70  7 50 10   111 70  3 62  0     1 20 21 65 10
587 60  3 58  0   389 90  2 62  0    33 30  6 64  0    25 20 36 63  0
357 70 13 58  0   467 90  2 64  0   201 80 28 52 10     1 50  7 35  0
 30 70 11 63  0    44 60 13 70 10   283 90  2 51  0    15 50 13 40 10
test small
 25 30  2 69  0  -103 70 22 36 10    21 20  4 71  0    13 30  2 62  0
 87 60  2 60  0     2 40 36 44 10    20 30  9 54 10     7 20 11 66  0
 24 60  8 49  0    99 70  3 72  0     8 80  2 68  0    99 85  4 62  0
 61 70  2 71  0    25 70  2 70  0    95 70  1 61  0    80 50 17 71  0
 51 30 87 59 10    29 40  8 67  0
test adeno
 24 40  2 60  0    18 40  5 69 10   -83 99  3 57  0    31 80  3 39  0
 51 60  5 62  0    90 60 22 50 10    52 60  3 43  0    73 60  3 70  0
  8 50  5 66  0    36 70  8 61  0    48 10  4 81  0     7 40  4 58  0
140 70  3 63  0   186 90  3 60  0    84 80  4 62 10    19 50 10 42  0
 45 40  3 69  0    80 40  4 63  0
test large
 52 60  4 45  0   164 70 15 68 10    19 30  4 39 10    53 60 12 66  0
 15 30  5 63  0    43 60 11 49 10   340 80 10 64 10   133 75  1 65  0
111 60  5 64  0   231 70 18 67 10   378 80  4 65  0    49 30  3 37  0
;
```

The following statements use the PHREG procedure to fit the Cox proportional hazards model to these data. The variables Prior, Cell, and Therapy, which are categorical variables, are declared in the CLASS statement. By default, PROC PHREG parameterizes the CLASS variables by using the reference coding with the last category as the reference category. However, you can explicitly specify the reference category of your choice. Here, Prior=no is chosen as the reference category for prior therapy, Cell=large is chosen as the reference category for type of tumor cell, and Therapy=standard is chosen as the reference category for the type of therapy. In the MODEL statement, the term Prior|Therapy is just another way of specifying the main effects Prior, Therapy, and the Prior*Therapy interaction.

```
proc phreg data=VALung;
   class Prior(ref='no') Cell(ref='large') Therapy(ref='standard');
   model Time*Status(0) = Kps Duration Age Cell Prior|Therapy;
run;
```

Coding of the CLASS variables is displayed in Output 92.3.1. There is one dummy variable for Prior and one for Therapy, since both variables are binary. The dummy variable has a value of 0 for the reference category (Prior=no, Therapy=standard). The variable Cell has four categories and is represented by three dummy variables. Note that the reference category, Cell=large, has a value of 0 for all three dummy variables.

Output 92.3.1 Reference Coding of CLASS Variables

The PHREG Procedure

Class Level Information				
Class	Value	Design Variables		
Prior	no	0		
	yes	1		
Cell	adeno	1	0	0
	large	0	0	0
	small	0	1	0
	squamous	0	0	1
Therapy	standard	0		
	test	1		

The test results of individual model effects are shown in Output 92.3.2. There is a strong prognostic effect of Kps on patient's survivorship ($p < 0.0001$), and the survival times for patients of different Cell types differ significantly ($p = 0.0003$). The Prior*Therapy interaction is marginally significant ($p = 0.0416$)—that is, prior therapy might play a role in whether one treatment is more effective than the other.

Output 92.3.2 Wald Tests of Individual Effects

Joint Tests			
Effect	DF	Wald Chi-Square	Pr > ChiSq
Kps	1	35.5051	<.0001
Duration	1	0.1159	0.7335
Age	1	1.9772	0.1597
Cell	3	18.5339	0.0003
Prior	1	2.5296	0.1117
Therapy	1	5.2349	0.0221
Prior*Therapy	1	4.1528	0.0416

Note: Under full-rank parameterizations, Type 3 effect tests are replaced by joint tests. The joint test for an effect is a test that all of the parameters associated with that effect are zero. Such joint tests might not be equivalent to Type 3 effect tests under GLM parameterization.

In the Cox proportional hazards model, the effects of the covariates are to act multiplicatively on the hazard of the survival time, and therefore it is a little easier to interpret the corresponding hazard ratios than the regression parameters. For a parameter that corresponds to a continuous variable, the hazard ratio is the ratio of hazard rates for a increase of one unit of the variable. From Output 92.3.3, the hazard ratio estimate for Kps is 0.968, meaning that an increase of 10 units in Karnofsky performance scale will shrink the hazard rate by $1 - (0.968)^{10}$=28%. For a CLASS variable parameter, the hazard ratio presented in the Output 92.3.3 is the ratio of the hazard rates between the given category and the reference category. The hazard rate of Cell=adeno is 219% that of Cell=large, the hazard rate of Cell=small is 162% that of Cell=large, and the hazard rate of Cell=squamous is only 66% that of Cell=large. Hazard ratios for Prior and Therapy are missing since the model contains the Prior*Therapy interaction. You can use the HAZARDRATIO statement to obtain the hazard ratios for a main effect in the presence of interaction as shown later in this example.

Output 92.3.3 Parameters Estimates with Reference Coding

Analysis of Maximum Likelihood Estimates								
Parameter		**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**	**Label**
Kps		1	-0.03300	0.00554	35.5051	<.0001	0.968	Karnofsky performance scale
Duration		1	0.00323	0.00949	0.1159	0.7335	1.003	months from diagnosis to randomization
Age		1	-0.01353	0.00962	1.9772	0.1597	0.987	age in years
Cell	**adeno**	1	0.78356	0.30382	6.6512	0.0099	2.189	cell type adeno
Cell	**small**	1	0.48230	0.26537	3.3032	0.0691	1.620	cell type small
Cell	**squamous**	1	-0.40770	0.28363	2.0663	0.1506	0.665	cell type squamous
Prior	**yes**	1	0.45914	0.28868	2.5296	0.1117	.	prior therapy yes
Therapy	**test**	1	0.56662	0.24765	5.2349	0.0221	.	type of treatment test
Prior*Therapy	**yes test**	1	-0.87579	0.42976	4.1528	0.0416	.	prior therapy yes * type of treatment test

The following PROC PHREG statements illustrate the use of the backward elimination process to identify the effects that affect the survivorship of the lung cancer patients. The option SELECTION=BACKWARD is specified to carry out the backward elimination. The option SLSTAY=0.1 specifies the significant level for retaining the effects in the model.

```
proc phreg data=VALung;
   class Prior(ref='no') Cell(ref='large') Therapy(ref='standard');
   model Time*Status(0) = Kps Duration Age Cell Prior|Therapy
         / selection=backward slstay=0.1;
run;
```

Results of the backward elimination process are summarized in Output 92.3.4. The effect Duration was eliminated first and was followed by Age.

Output 92.3.4 Effects Eliminated from the Model

The PHREG Procedure

Summary of Backward Elimination						
Step	**Effect Removed**	**DF**	**Number In**	**Wald Chi-Square**	**Pr > ChiSq**	**Effect Label**
1	Duration	1	6	0.1159	0.7335	months from diagnosis to randomization
2	Age	1	5	2.0458	0.1526	age in years

Output 92.3.5 shows the Type 3 analysis of effects and the maximum likelihood estimates of the regression coefficients of the model. Without controlling for Age and Duration, KPS and Cell remain significant, but the Prior*Therapy interaction is less prominent than before ($p = 0.0871$) though still significant at 0.1 level.

Output 92.3.5 Type 3 Effects and Parameter Estimates for the Selected Model

Joint Tests			
Effect	DF	Wald Chi-Square	Pr > ChiSq
Kps	1	35.9218	<.0001
Cell	3	17.4134	0.0006
Prior	1	2.3113	0.1284
Therapy	1	3.8030	0.0512
Prior*Therapy	1	2.9269	0.0871

Note: Under full-rank parameterizations, Type 3 effect tests are replaced by joint tests. The joint test for an effect is a test that all of the parameters associated with that effect are zero. Such joint tests might not be equivalent to Type 3 effect tests under GLM parameterization.

Analysis of Maximum Likelihood Estimates								
Parameter			DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Kps			1	-0.03111	0.00519	35.9218	<.0001	0.969	Karnofsky performance scale
Cell	adeno		1	0.74907	0.30465	6.0457	0.0139	2.115	cell type adeno
Cell	small		1	0.44265	0.26168	2.8614	0.0907	1.557	cell type small
Cell	squamous		1	-0.41145	0.28309	2.1125	0.1461	0.663	cell type squamous
Prior	yes		1	0.41755	0.27465	2.3113	0.1284	.	prior therapy yes
Therapy	test		1	0.45670	0.23419	3.8030	0.0512	.	type of treatment test
Prior*Therapy	yes	test	1	-0.69443	0.40590	2.9269	0.0871	.	prior therapy yes * type of treatment test

Finally, the following statements refit the previous model and computes hazard ratios at settings beyond those displayed in the "Analysis of Maximum Likelihood Estimates" table. You can use either the HAZARDRATIO statement or the CONTRAST statement to obtain hazard ratios. Using the CONTRAST statement to compute hazard ratios for CLASS variables can be a daunting task unless you are familiar with the parameterization schemes (see the section "Parameterization of Model Effects" on page 412 in Chapter 20, "Shared Concepts and Topics"), but you have control over which specific hazard ratios you want to compute. HAZARDRATIO statements, on the other hand, are designed specifically to provide hazard ratios. They are easy to use and you can also request both the Wald confidence limits and the profile-likelihood confidence limits; the latter is not available for the CONTRAST statements. Three HAZARDRATIO statements are specified; each has the CL=BOTH option to request both the Wald confidence limits and the profile-likelihood limits. The first HAZARDRATIO statement, labeled 'H1', estimates the hazard ratio for an increase of 10 units in the KPS; the UNITS= option specifies the number of units increase. The second HAZARDRATIO statement, labeled 'H2' computes the hazard ratios for comparing any pairs of tumor Cell types. The third HAZARDRATIO statement, labeled 'H3', compares the test therapy with the standard therapy. The DIFF=REF option specifies that each nonreference category is compared to the reference category. The purpose of using DIFF=REF here is to ensure that the hazard ratio is comparing the test therapy to the standard therapy instead of the other way around. Three CONTRAST statements, labeled 'C1', 'C2', and 'C3', parallel to the HAZARDRATIO statements 'H1', 'H2', and 'H3', respectively, are specified. The ESTIMATE=EXP option specifies that the linear predictors be estimated in the exponential scale, which are precisely the hazard ratios.

```
proc phreg data=VALung;
   class Prior(ref='no') Cell(ref='large') Therapy(ref='standard');
   model Time*Status(0) = Kps Cell Prior|Therapy;
   hazardratio 'H1' Kps / units=10 cl=both;
   hazardratio 'H2' Cell / cl=both;
```

```
    hazardratio 'H3' Therapy / diff=ref cl=both;
    contrast 'C1' Kps 10 / estimate=exp;
    contrast 'C2' cell 1  0  0, /* adeno vs large    */
                  cell 1 -1  0, /* adeno vs small    */
                  cell 1  0 -1, /* adeno vs squamous */
                  cell 0 -1  0, /* large vs small    */
                  cell 0  0 -1, /* large vs Squamous */
                  cell 0  1 -1  /* small vs squamous */
                   / estimate=exp;
    contrast 'C3' Prior 0 Therapy 1  Prior*Therapy 0,
                  Prior 0 Therapy 1  Prior*Therapy 1  / estimate=exp;
  run;
```

Output 92.3.6 displays the results of the three HAZARDRATIO statements in separate tables. Results of the three CONTRAST statements are shown in one table in Output 92.3.7. However, point estimates and the Wald confidence limits for the hazard ratio agree in between the two outputs.

Output 92.3.6 Results from HAZARDRATIO Statements

The PHREG Procedure

H1: Hazard Ratios for Karnofsky performance scale

Description	Point Estimate	95% Wald Confidence Limits		95% Profile Likelihood Confidence Limits	
Kps Unit=10	0.733	0.662	0.811	0.662	0.811

H2: Hazard Ratios for cell type

Description	Point Estimate	95% Wald Confidence Limits		95% Profile Likelihood Confidence Limits	
Cell adeno vs large	2.115	1.164	3.843	1.162	3.855
Cell adeno vs small	1.359	0.798	2.312	0.791	2.301
Cell adeno vs squamous	3.192	1.773	5.746	1.770	5.768
Cell large vs small	0.642	0.385	1.073	0.380	1.065
Cell large vs squamous	1.509	0.866	2.628	0.863	2.634
Cell small vs squamous	2.349	1.387	3.980	1.399	4.030

H3: Hazard Ratios for type of treatment

Description	Point Estimate	95% Wald Confidence Limits		95% Profile Likelihood Confidence Limits	
Therapy test vs standard At Prior=no	1.579	0.998	2.499	0.998	2.506
Therapy test vs standard At Prior=yes	0.788	0.396	1.568	0.390	1.560

Output 92.3.7 Results from CONTRAST Statements

Contrast Estimation and Testing Results by Row									
Contrast	Type	Row	Estimate	Standard Error	Alpha	Confidence Limits		Wald Chi-Square	Pr > ChiSq
C1	EXP	1	0.7326	0.0380	0.05	0.6618	0.8111	35.9218	<.0001
C2	EXP	1	2.1150	0.6443	0.05	1.1641	3.8427	6.0457	0.0139
C2	EXP	2	1.3586	0.3686	0.05	0.7982	2.3122	1.2755	0.2587
C2	EXP	3	3.1916	0.9575	0.05	1.7727	5.7462	14.9629	0.0001
C2	EXP	4	0.6423	0.1681	0.05	0.3846	1.0728	2.8614	0.0907
C2	EXP	5	1.5090	0.4272	0.05	0.8664	2.6282	2.1125	0.1461
C2	EXP	6	2.3493	0.6318	0.05	1.3868	3.9797	10.0858	0.0015
C3	EXP	1	1.5789	0.3698	0.05	0.9977	2.4985	3.8030	0.0512
C3	EXP	2	0.7884	0.2766	0.05	0.3964	1.5680	0.4593	0.4980

Example 92.4: Firth's Correction for Monotone Likelihood

(View the complete code for this example (phrex4.sas) in the example repository.)

In fitting the Cox regression model by maximizing the partial likelihood, the estimate of an explanatory variable X will be infinite if the value of X at each uncensored failure time is the largest of all the values of X in the risk set at that time (Tsiatis 1981; Bryson and Johnson 1981). You can exploit this information to artificially create a data set that has the condition of monotone likelihood for the Cox regression. The following DATA step modifies the Myeloma data in Example 92.1 to create a new explanatory variable, Contrived, which has the value 1 if the observed time is less than or equal to 65 and has the value 0 otherwise. The phenomenon of monotone likelihood will be demonstrated in the new data set Myeloma2.

```
data Myeloma2;
   set Myeloma;
   Contrived= (Time <= 65);
run;
```

For illustration purposes, consider a Cox model with three explanatory variables, one of which is the variable Contrived. The following statements invoke PROC PHREG to perform the Cox regression. The IPRINT option is specified in the MODEL statement to print the iteration history of the optimization.

```
proc phreg data=Myeloma2;
   model Time*Vstatus(0)=LOGbun HGB Contrived / itprint;
run;
```

The symptom of monotonicity is demonstrated in Output 92.4.1. The log likelihood converges to the value of –136.56 while the coefficient for Contrived diverges. Although the Newton-Raphson optimization process did not fail, it is obvious that convergence is questionable. A close examination of the standard errors in the "Analysis of Maximum Likelihood Estimates" table reveals a very large value for the coefficient of Contrived. This is very typical of a diverged estimate.

Output 92.4.1 Monotone Likelihood Behavior Displayed

The PHREG Procedure

Maximum Likelihood Iteration History

Iter	Ridge	Log Likelihood	LogBUN	HGB	Contrived
0	0	-154.8579914384	0.0000000000	0.000000000	0.000000000
1	0	-140.6934052686	1.9948819671	-0.084318519	1.466331269
2	0	-137.7841629416	1.6794678962	-0.109067888	2.778361123
3	0	-136.9711897754	1.7140611684	-0.111564202	3.938095086
4	0	-136.7078932606	1.7181735043	-0.112273248	5.003053568
5	0	-136.6164264879	1.7187547532	-0.112369756	6.027435769
6	0	-136.5835200895	1.7188294108	-0.112382079	7.036444978
7	0	-136.5715152788	1.7188392687	-0.112383700	8.039763533
8	0	-136.5671126045	1.7188405904	-0.112383917	9.040984886
9	0	-136.5654947987	1.7188407687	-0.112383947	10.041434266
10	0	-136.5648998913	1.7188407928	-0.112383950	11.041599592
11	0	-136.5646810709	1.7188407960	-0.112383951	12.041660414
12	0	-136.5646005760	1.7188407965	-0.112383951	13.041682789
13	0	-136.5645709642	1.7188407965	-0.112383951	14.041691020
14	0	-136.5645600707	1.7188407965	-0.112383951	15.041694049
15	0	-136.5645560632	1.7188407965	-0.112383951	16.041695162
16	0	-136.5645545889	1.7188407965	-0.112383951	17.041695571

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
LogBUN	1	1.71884	0.58376	8.6697	0.0032	5.578
HGB	1	-0.11238	0.06090	3.4053	0.0650	0.894
Contrived	1	17.04170	1080	0.0002	0.9874	25183399

Next, the Firth correction was applied as shown in the following statements. Also, the profile-likelihood confidence limits for the hazard ratios are requested by using the RISKLIMITS=PL option.

```
proc phreg data=Myeloma2;
   model Time*Vstatus(0)=LogBUN HGB Contrived /
      firth risklimits=pl itprint;
run;
```

PROC PHREG uses the penalized likelihood maximum to obtain a finite estimate for the coefficient of Contrived (Output 92.4.2). The much preferred profile-likelihood confidence limits, as shown in (Heinze and Schemper 2001), are also displayed.

Output 92.4.2 Convergence Obtained with the Firth Correction

The PHREG Procedure

Maximum Likelihood Iteration History

Iter	Ridge	Log Likelihood	LogBUN	HGB	Contrived
0	0	-150.7361197494	0.0000000000	0.000000000	0.0000000000
1	0	-136.9933949142	2.0262484120	-0.086519138	1.4338859318
2	0	-134.5796594364	1.6770836974	-0.109172604	2.6221444778
3	0	-134.1572923217	1.7163408994	-0.111166227	3.4458043289
4	0	-134.1229607193	1.7209210332	-0.112007726	3.7923555412
5	0	-134.1228364805	1.7219588214	-0.112178328	3.8174197804
6	0	-134.1228355256	1.7220081673	-0.112187764	3.8151642206

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	95% Hazard Ratio Profile Likelihood Confidence Limits	
LogBUN	1	1.72201	0.58379	8.7008	0.0032	5.596	1.761	17.231
HGB	1	-0.11219	0.06059	3.4279	0.0641	0.894	0.794	1.007
Contrived	1	3.81516	1.55812	5.9955	0.0143	45.384	5.406	6005.404

Example 92.5: Conditional Logistic Regression for m:n Matching

(View the complete code for this example (phrex5.sas) in the example repository.)

Conditional logistic regression is used to investigate the relationship between an outcome and a set of prognostic factors in matched case-control studies. The outcome is whether the subject is a case or a control. If there is only one case and one control, the matching is 1:1. The *m:n* matching refers to the situation in which there is a varying number of cases and controls in the matched sets. You can perform conditional logistic regression with the PHREG procedure by using the discrete logistic model and forming a stratum for each matched set. In addition, you need to create dummy survival times so that all the cases in a matched set have the same event time value, and the corresponding controls are censored at later times.

Consider the following set of low infant birth-weight data extracted from Appendix 1 of Hosmer and Lemeshow (1989). These data represent 189 women, of whom 59 had low-birth-weight babies and 130 had normal-weight babies. Under investigation are the following risk factors: weight in pounds at the last menstrual period (LWT), presence of hypertension (HT), smoking status during pregnancy (Smoke), and presence of uterine irritability (UI). For HT, Smoke, and UI, a value of 1 indicates a "yes" and a value of 0 indicates a "no." The woman's age (Age) is used as the matching variable. The SAS data set LBW contains a subset of the data corresponding to women between the ages of 16 and 32.

```
data LBW;
   input id Age Low LWT Smoke HT UI @@;
   Time=2-Low;
   datalines;
 25  16   1   130   0  0  0    143  16   0   110   0  0  0
166  16   0   112   0  0  0    167  16   0   135   1  0  0
```

189	16	0	135	1	0	0	206	16	0	170	0	0	0
216	16	0	95	0	0	0	37	17	1	130	1	0	1
45	17	1	110	1	0	0	68	17	1	120	1	0	0
71	17	1	120	0	0	0	83	17	1	142	0	1	0
93	17	0	103	0	0	0	113	17	0	122	1	0	0
116	17	0	113	0	0	0	117	17	0	113	0	0	0
147	17	0	119	0	0	0	148	17	0	119	0	0	0
180	17	0	120	1	0	0	49	18	1	148	0	0	0
50	18	1	110	1	0	0	89	18	0	107	1	0	1
100	18	0	100	1	0	0	101	18	0	100	1	0	0
132	18	0	90	1	0	1	133	18	0	90	1	0	1
168	18	0	229	0	0	0	205	18	0	120	1	0	0
208	18	0	120	0	0	0	23	19	1	91	1	0	1
33	19	1	102	0	0	0	34	19	1	112	1	0	1
85	19	0	182	0	0	1	96	19	0	95	0	0	0
97	19	0	150	0	0	0	124	19	0	138	1	0	0
129	19	0	189	0	0	0	135	19	0	132	0	0	0
142	19	0	115	0	0	0	181	19	0	105	0	0	0
187	19	0	235	1	1	0	192	19	0	147	1	0	0
193	19	0	147	1	0	0	197	19	0	184	1	1	0
224	19	0	120	1	0	0	27	20	1	150	1	0	0
31	20	1	125	0	0	1	40	20	1	120	1	0	0
44	20	1	80	1	0	1	47	20	1	109	0	0	0
51	20	1	121	1	0	1	60	20	1	122	1	0	0
76	20	1	105	0	0	0	87	20	0	105	1	0	0
104	20	0	120	0	0	1	146	20	0	103	0	0	0
155	20	0	169	0	0	1	160	20	0	141	0	0	1
172	20	0	121	1	0	0	177	20	0	127	0	0	0
201	20	0	120	0	0	0	211	20	0	170	1	0	0
217	20	0	158	0	0	0	20	21	1	165	1	1	0
28	21	1	200	0	0	1	30	21	1	103	0	0	0
52	21	1	100	0	0	0	84	21	1	130	1	1	0
88	21	0	108	1	0	1	91	21	0	124	0	0	0
128	21	0	185	1	0	0	131	21	0	160	0	0	0
144	21	0	110	1	0	1	186	21	0	134	0	0	0
219	21	0	115	0	0	0	42	22	1	130	1	0	1
67	22	1	130	1	0	0	92	22	0	118	0	0	0
98	22	0	95	0	1	0	137	22	0	85	1	0	0
138	22	0	120	0	1	0	140	22	0	130	1	0	0
161	22	0	158	0	0	0	162	22	0	112	1	0	0
174	22	0	131	0	0	0	184	22	0	125	0	0	0
204	22	0	169	0	0	0	220	22	0	129	0	0	0
17	23	1	97	0	0	1	59	23	1	187	1	0	0
63	23	1	120	0	0	0	69	23	1	110	1	0	0
82	23	1	94	1	0	0	130	23	0	130	0	0	0
139	23	0	128	0	0	0	149	23	0	119	0	0	0
164	23	0	115	1	0	0	173	23	0	190	0	0	0
179	23	0	123	0	0	0	182	23	0	130	0	0	0
200	23	0	110	0	0	0	18	24	1	128	0	0	0
19	24	1	132	0	1	0	29	24	1	155	1	0	0
36	24	1	138	0	0	0	61	24	1	105	1	0	0
118	24	0	90	1	0	0	136	24	0	115	0	0	0
150	24	0	110	0	0	0	156	24	0	115	0	0	0
185	24	0	133	0	0	0	196	24	0	110	0	0	0

```
199  24   0   110   0  0  0    225  24   0   116   0  0  0
 13  25   1   105   0  1  0     15  25   1    85   0  0  1
 24  25   1   115   0  0  0     26  25   1    92   1  0  0
 32  25   1    89   0  0  0     46  25   1   105   0  0  0
103  25   0   118   1  0  0    111  25   0   120   0  0  1
120  25   0   155   0  0  0    121  25   0   125   0  0  0
169  25   0   140   0  0  0    188  25   0    95   1  0  1
202  25   0   241   0  1  0    215  25   0   120   0  0  0
221  25   0   130   0  0  0     35  26   1   117   1  0  0
 54  26   1    96   0  0  0     75  26   1   154   0  1  0
 77  26   1   190   1  0  0     95  26   0   113   1  0  0
115  26   0   168   1  0  0    154  26   0   133   1  0  0
218  26   0   160   0  0  0     16  27   1   150   0  0  0
 43  27   1   130   0  0  1    125  27   0   124   1  0  0
  4  28   1   120   1  0  1     79  28   1    95   1  0  0
105  28   0   120   1  0  0    109  28   0   120   0  0  0
112  28   0   167   0  0  0    151  28   0   140   0  0  0
159  28   0   250   1  0  0    212  28   0   134   0  0  0
214  28   0   130   0  0  0     10  29   1   130   0  0  1
 94  29   0   123   1  0  0    114  29   0   150   0  0  0
123  29   0   140   1  0  0    190  29   0   135   0  0  0
191  29   0   154   0  0  0    209  29   0   130   1  0  0
 65  30   1   142   1  0  0     99  30   0   107   0  0  1
141  30   0    95   1  0  0    145  30   0   153   0  0  0
176  30   0   110   0  0  0    195  30   0   137   0  0  0
203  30   0   112   0  0  0     56  31   1   102   1  0  0
107  31   0   100   0  0  1    126  31   0   215   1  0  0
163  31   0   150   1  0  0    222  31   0   120   0  0  0
 22  32   1   105   1  0  0    106  32   0   121   0  0  0
134  32   0   132   0  0  0    170  32   0   134   1  0  0
175  32   0   170   0  0  0    207  32   0   186   0  0  0
;
```

The variable Low is used to determine whether the subject is a case (Low=1, low-birth-weight baby) or a control (Low=0, normal-weight baby). The dummy time variable Time takes the value 1 for cases and 2 for controls.

The following statements produce a conditional logistic regression analysis of the data. The variable Time is the response, and Low is the censoring variable. Note that the data set is created so that all the cases have the same event time and the controls have later censored times. The matching variable Age is used in the STRATA statement so that each unique age value defines a stratum. The variables LWT, Smoke, HT, and UI are specified as explanatory variables. The TIES=DISCRETE option requests the discrete logistic model.

```
proc phreg data=LBW;
   model Time*Low(0)= LWT Smoke HT UI / ties=discrete;
   strata Age;
run;
```

The procedure displays a summary of the number of event and censored observations for each stratum. These are the number of cases and controls for each matched set shown in Output 92.5.1.

Output 92.5.1 Summary of Number of Case and Controls

The PHREG Procedure

Model Information	
Data Set	WORK.LBW
Dependent Variable	Time
Censoring Variable	Low
Censoring Value(s)	0
Ties Handling	DISCRETE

Summary of the Number of Event and Censored Values

Stratum	Age	Total	Event	Censored	Percent Censored
1	16	7	1	6	85.71
2	17	12	5	7	58.33
3	18	10	2	8	80.00
4	19	16	3	13	81.25
5	20	18	8	10	55.56
6	21	12	5	7	58.33
7	22	13	2	11	84.62
8	23	13	5	8	61.54
9	24	13	5	8	61.54
10	25	15	6	9	60.00
11	26	8	4	4	50.00
12	27	3	2	1	33.33
13	28	9	2	7	77.78
14	29	7	1	6	85.71
15	30	7	1	6	85.71
16	31	5	1	4	80.00
17	32	6	1	5	83.33
Total		174	54	120	68.97

Results of the conditional logistic regression analysis are shown in Output 92.5.2. Based on the Wald test for individual variables, the variables LWT, Smoke, and HT are statistically significant while UI is marginal.

The hazard ratios, computed by exponentiating the parameter estimates, are useful in interpreting the results of the analysis. If the hazard ratio of a prognostic factor is larger than 1, an increment in the factor increases the hazard rate. If the hazard ratio is less than 1, an increment in the factor decreases the hazard rate. Results indicate that women were more likely to have low-birth-weight babies if they were underweight in the last menstrual cycle, were hypertensive, smoked during pregnancy, or suffered uterine irritability.

Output 92.5.2 Conditional Logistic Regression Analysis for the Low-Birth-Weight Study

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Output 92.5.2 *continued*

Model Fit Statistics		
Criterion	Without Covariates	With Covariates
-2 LOG L	159.069	141.108
AIC	159.069	149.108
SBC	159.069	157.064

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	17.9613	4	0.0013
Score	17.3152	4	0.0017
Wald	15.5577	4	0.0037

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
LWT	1	-0.01498	0.00706	4.5001	0.0339	0.985
Smoke	1	0.80805	0.36797	4.8221	0.0281	2.244
HT	1	1.75143	0.73932	5.6120	0.0178	5.763
UI	1	0.88341	0.48032	3.3827	0.0659	2.419

For matched case-control studies with one case per matched set (1:n matching), the likelihood function for the conditional logistic regression reduces to that of the Cox model for the continuous time scale. For this situation, you can use the default TIES=BRESLOW.

Example 92.6: Model Using Time-Dependent Explanatory Variables

(View the complete code for this example (phrex6.sas) in the example repository.)

Time-dependent variables can be used to model the effects of subjects transferring from one treatment group to another. One example of the need for such strategies is the Stanford heart transplant program. Patients are accepted if physicians judge them suitable for heart transplant. Then, when a donor becomes available, physicians choose transplant recipients according to various medical criteria. A patient's status can be changed during the study from waiting for a transplant to being a transplant recipient. Transplant status can be defined by the time-dependent covariate function $z = z(t)$ as

$$z(t) = \begin{cases} 0 & \text{if the patient has not received the transplant at time } t \\ 1 & \text{if the patient has received the transplant at time } t \end{cases}$$

The Stanford heart transplant data that appear in Crowley and Hu (1977) consist of 103 patients, 69 of whom received transplants. The data are saved in a SAS data set called Heart in the following DATA step. For each patient in the program, there is a birth date (Bir_Date), a date of acceptance (Acc_Date), and a date last seen (Ter_Date). The survival time (Time) in days is defined as Time = Ter_Date – Acc_Date. The survival time is said to be uncensored (Status=1) or censored (Status=0), depending on whether Ter_Date is the date of death or the closing date of the study. The age, in years, at acceptance into the program is Acc_Age = (Acc_Date – Bir_Date) / 365. Previous open-heart surgery for each patient is indicated by the variable PrevSurg. For each transplant recipient, there is a date of transplant (Xpl_Date) and three measures (NMismatch, Antigen,

Mismatch) of tissue-type mismatching. The waiting period (WaitTime) in days for a transplant recipient is calculated as WaitTime = Xpl_Date – Acc_Date, and the age (in years) at transplant is Xpl_Age = (Xpl_Date – Bir_Date) / 365. For those who do not receive heart transplants, the WaitTime, Xpl_Age, NMismatch, Antigen, and Mismatch variables contain missing values.

The input data contain dates that have a two-digit year representation. The SAS option YEARCUTOFF=1900 is specified to ensure that a two-digit year xx is year 19xx.

```
options yearcutoff=1900;
data Heart;
   input ID
         @5  Bir_Date mmddyy8.
         @14 Acc_Date mmddyy8.
         @23 Xpl_Date mmddyy8.
         @32 Ter_Date mmddyy8.
         @41 Status 1.
         @43 PrevSurg 1.
         @45 NMismatch 1.
         @47 Antigen 1.
         @49 Mismatch 4.
         @54 Reject 1.
         @56 NotTyped $1.;
   label Bir_Date ='Date of birth'
         Acc_Date ='Date of acceptance'
         Xpl_Date ='Date of transplant'
         Ter_Date ='Date last seen'
         Status   =  'Dead=1 Alive=0'
         PrevSurg ='Previous surgery'
         NMismatch= 'No of mismatches'
         Antigen  = 'HLA-A2 antigen'
         Mismatch ='Mismatch score'
         NotTyped = 'y=not tissue-typed';
   Time= Ter_Date - Acc_Date;
   Acc_Age=int( (Acc_Date - Bir_Date)/365 );
   if ( Xpl_Date ne .) then do;
      WaitTime= Xpl_Date - Acc_Date;
      Xpl_Age= int( (Xpl_Date - Bir_Date)/365 );
   end;
   datalines;
  1 01 10 37 11 15 67          01 03 68 1 0
  2 03 02 16 01 02 68          01 07 68 1 0
  3 09 19 13 01 06 68 01 06 68 01 21 68 1 0 2 0 1.11 0
  4 12 23 27 03 28 68 05 02 68 05 05 68 1 0 3 0 1.66 0
  5 07 28 47 05 10 68          05 27 68 1 0
  6 11 08 13 06 13 68          06 15 68 1 0
  7 08 29 17 07 12 68 08 31 68 05 17 70 1 0 4 0 1.32 1
  8 03 27 23 08 01 68          09 09 68 1 0
  9 06 11 21 08 09 68          11 01 68 1 0
 10 02 09 26 08 11 68 08 22 68 10 07 68 1 0 2 0 0.61 1
 11 08 22 20 08 15 68 09 09 68 01 14 69 1 0 1 0 0.36 0
 12 07 09 15 09 17 68          09 24 68 1 0
 13 02 22 14 09 19 68 10 05 68 12 08 68 1 0 3 0 1.89 1
 14 09 16 14 09 20 68 10 26 68 07 07 72 1 0 1 0 0.87 1
 15 12 04 14 09 27 68          09 27 68 1 1
```

```
16 05 16 19 10 26 68 11 22 68 08 29 69 1 0 2 0 1.12 1
17 06 29 48 10 28 68          12 02 68 1 0
18 12 27 11 11 01 68 11 20 68 12 13 68 1 0 3 0 2.05 0
19 10 04 09 11 18 68          12 24 68 1 0
20 10 19 13 01 29 69 02 15 69 02 25 69 1 0 3 1 2.76 1
21 09 29 25 02 01 69 02 08 69 11 29 71 1 0 2 0 1.13 1
22 06 05 26 03 18 69 03 29 69 05 07 69 1 0 3 0 1.38 1
23 12 02 10 04 11 69 04 13 69 04 13 71 1 0 3 0 0.96 1
24 07 07 17 04 25 69 07 16 69 11 29 69 1 0 3 1 1.62 1
25 02 06 36 04 28 69 05 22 69 04 01 74 0 0 2 0 1.06 0
26 10 18 38 05 01 69          03 01 73 0 0
27 07 21 60 05 04 69          01 21 70 1 0
28 05 30 15 06 07 69 08 16 69 08 17 69 1 0 2 0 0.47 0
29 02 06 19 07 14 69          08 17 69 1 0
30 09 20 24 08 19 69 09 03 69 12 18 71 1 0 4 0 1.58 1
31 10 04 14 08 23 69          09 07 69 1 0
32 04 02 05 08 29 69 09 14 69 11 13 69 1 0 4 0 0.69 1
33 01 01 21 11 27 69 01 16 70 04 01 74 0 0 3 0 0.91 0
34 05 24 29 12 12 69 01 03 70 04 01 74 0 0 2 0 0.38 0
35 08 04 26 01 21 70          02 01 70 1 0
36 05 01 21 04 04 70 05 19 70 07 12 70 1 0 2 0 2.09 1
37 10 24 08 04 25 70 05 13 70 06 29 70 1 0 3 1 0.87 1
38 11 14 28 05 05 70 05 09 70 05 09 70 1 0 3 0 0.87 0
39 11 12 19 05 20 70 05 21 70 07 11 70 1 0            y
40 11 30 21 05 25 70 07 04 70 04 01 74 0 1 4 0 0.75 0
41 04 30 25 08 19 70 10 15 70 04 01 74 0 1 2 0 0.98 0
42 03 13 34 08 21 70          08 23 70 1 0
43 06 01 27 10 22 70          10 23 70 1 1
44 05 02 28 11 30 70          01 08 71 1 1
45 10 30 34 01 05 71 01 05 71 02 18 71 1 0 1 0 0.0  0
46 06 01 22 01 10 71 01 11 71 10 01 73 1 1 2 0 0.81 1
47 12 28 23 02 02 71 02 22 71 04 14 71 1 0 3 0 1.38 1
48 01 23 15 02 05 71          02 13 71 1 0
49 06 21 34 02 15 71 03 22 71 04 01 74 0 1 4 0 1.35 0
50 03 28 25 02 15 71 05 08 71 10 21 73 1 1            y
51 06 29 22 03 24 71 04 24 71 01 02 72 1 0 4 1 1.08 1
52 01 24 30 04 25 71          08 04 71 1 0
53 02 27 24 07 02 71 08 11 71 01 05 72 1 0            y
54 09 16 23 07 02 71          07 04 71 1 0
55 02 24 19 08 09 71 08 18 71 10 08 71 1 0 2 0 1.51 1
56 12 05 32 09 03 71 11 08 71 04 01 74 0 0 4 0 0.98 0
57 06 08 30 09 13 71          02 08 72 1 0
58 09 17 23 09 23 71 10 13 71 08 30 72 1 1 2 1 1.82 1
59 05 12 30 09 29 71 12 15 71 04 01 74 0 1 2 0 0.19 0
60 10 29 22 11 18 71 11 20 71 01 24 72 1 0 3 0 0.66 1
61 05 12 19 12 04 71          12 05 71 1 0
62 08 01 32 12 09 71          02 15 72 1 0
63 04 15 39 12 12 71 01 07 72 04 01 74 0 0 3 1 1.93 0
64 04 09 23 02 01 72 03 04 72 09 06 73 1 1 1 0 0.12 0
65 11 19 20 03 06 72 03 17 72 05 22 72 1 0 2 0 1.12 1
66 01 02 19 03 20 72          04 20 72 1 0
67 09 03 52 03 23 72 05 18 72 01 01 73 1 0 3 0 1.02 0
68 01 10 27 04 07 72 04 09 72 06 13 72 1 0 3 1 1.68 1
69 06 05 24 06 01 72 06 10 72 04 01 74 0 0 2 0 1.20 0
```

```
 70 06 17 19 06 17 72 06 21 72 07 16 72 1 0 3 1 1.68 1
 71 02 22 25 07 21 72 08 20 72 04 01 74 0 0 3 0 0.97 0
 72 11 22 45 08 14 72 08 17 72 04 01 74 0 0 3 1 1.46 0
 73 05 13 16 09 11 72 10 07 72 12 09 72 1 0 3 1 2.16 1
 74 07 20 43 09 18 72 09 22 72 10 04 72 1 0 1 0 0.61 0
 75 07 25 20 09 29 72          09 30 72 1 0
 76 09 03 20 10 04 72 11 18 72 04 01 74 0 1 3 1 1.70 0
 77 08 27 31 10 06 72          10 26 72 1 0
 78 02 20 24 11 03 72 05 31 73 04 01 74 0 0 3 0 0.81 0
 79 02 18 19 11 30 72 02 04 73 03 05 73 1 0 2 0 1.08 1
 80 06 27 26 12 06 72 12 31 72 04 01 74 0 1 3 0 1.41 0
 81 02 21 20 01 12 73 01 17 73 04 01 74 0 0 4 1 1.94 0
 82 08 19 42 11 01 71          01 01 73 0 0
 83 10 04 19 01 24 73 02 24 73 04 13 73 1 0 4 0 3.05 0
 84 05 13 30 01 30 73 03 07 73 12 29 73 1 0 4 0 0.60 1
 85 02 13 25 02 06 73          02 10 73 1 0
 86 03 30 24 03 01 73 03 08 73 04 01 74 0 0 3 1 1.44 0
 87 12 19 26 03 21 73 05 19 73 07 08 73 1 0 2 0 2.25 1
 88 11 16 18 03 28 73 04 27 73 04 01 74 0 0 3 0 0.68 0
 89 03 19 22 04 05 73 08 21 73 10 28 73 1 0 4 1 1.33 1
 90 03 25 21 04 06 73 09 12 73 10 08 73 1 1 3 1 0.82 0
 91 09 08 25 04 13 73          03 18 74 1 0
 92 05 03 28 04 27 73 03 02 74 04 01 74 0 0 1 0 0.16 0
 93 10 10 25 07 11 73 08 07 73 04 01 74 0 0 2 0 0.33 0
 94 11 11 29 09 14 73 09 17 73 02 25 74 1 1 3 0 1.20 1
 95 06 11 33 09 22 73 09 23 73 10 07 73 1 0            y
 96 02 09 47 10 04 73 10 16 73 04 01 74 0 0 2 0 0.46 0
 97 04 11 50 11 22 73 12 12 73 04 01 74 0 0 3 1 1.78 0
 98 04 28 45 12 14 73 03 19 74 04 01 74 0 0 4 1 0.77 0
 99 02 24 24 12 25 73          01 14 74 1 0
100 01 31 39 02 22 74 03 31 74 04 01 74 0 1 3 0 0.67 0
101 08 25 24 03 02 74          04 01 74 0 0
102 10 30 33 03 22 74          04 01 74 0 0
103 05 20 28 09 13 67          09 18 67 1 0
;
```

Crowley and Hu (1977) have presented a number of analyses to assess the effects of various explanatory variables on the survival of patients. This example fits two of the models that they have considered.

The first model consists of two explanatory variables—the transplant status and the age at acceptance. The transplant status (XStatus) is a time-dependent variable defined by the programming statements between the MODEL statement and the RUN statement. The XStatus variable takes the value 1 or 0 at time t (measured from the date of acceptance), depending on whether or not the patient has received a transplant at that time. Note that the value of XStatus changes for subjects in each risk set (subjects still alive just before each distinct event time); therefore, the variable cannot be created in the DATA step. The variable Acc_Age, which is not time dependent, accounts for the possibility that pretransplant risks vary with age. The following statements fit this model:

```
proc phreg data= Heart;
   model Time*Status(0)= XStatus Acc_Age;
   if (WaitTime = . or Time < WaitTime) then XStatus = 0;
   else  XStatus = 1;
run;
```

Results of this analysis are shown in Output 92.6.1. Transplantation appears to be associated with a slight decrease in risk, although the effect is not significant ($p = 0.8261$). The age at acceptance as a pretransplant risk factor adds significantly to the model ($p = 0.0289$). The risk increases significantly with age at acceptance.

Output 92.6.1 Heart Transplant Study Analysis I

The PHREG Procedure

Model Information		
Data Set	WORK.HEART	
Dependent Variable	Time	
Censoring Variable	Status	Dead=1 Alive=0
Censoring Value(s)	0	
Ties Handling	BRESLOW	

Number of Observations Read	103
Number of Observations Used	103

Summary of the Number of Event and Censored Values			
Total	**Event**	**Censored**	**Percent Censored**
103	75	28	27.18

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	**Without Covariates**	**With Covariates**
-2 LOG L	596.651	591.292
AIC	596.651	595.292
SBC	596.651	599.927

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	5.3593	2	0.0686
Score	4.8093	2	0.0903
Wald	4.7999	2	0.0907

Analysis of Maximum Likelihood Estimates						
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**
XStatus	1	-0.06720	0.30594	0.0482	0.8261	0.935
Acc_Age	1	0.03158	0.01446	4.7711	0.0289	1.032

The second model consists of three explanatory variables—the transplant status, the transplant age, and the mismatch score. Four transplant recipients who were not typed have no Mismatch values; they are excluded from the analysis by the use of a WHERE clause. The transplant age (XAge) and the mismatch score (XScore) are also time dependent and are defined in a fashion similar to that of XStatus. While the patient is waiting for a transplant, XAge and XScore have a value of 0. After the patient has migrated to the recipient population,

XAge takes on the value of Xpl_Age (transplant age for the recipient), and XScore takes on the value of Mismatch (a measure of the degree of dissimilarity between donor and recipient). The following statements fit this model:

```
proc phreg data= Heart;
   model Time*Status(0)= XStatus XAge XScore;
   where NotTyped ^= 'y';
   if (WaitTime = . or Time < WaitTime) then do;
      XStatus = 0;
      XAge = 0;
      XScore = 0;
   end;
   else do;
      XStatus = 1;
      XAge = Xpl_Age;
      XScore = Mismatch;
   end;
run;
```

Results of the analysis are shown in Output 92.6.2. Note that only 99 patients are included in this analysis, instead of 103 patients as in the previous analysis, since four transplant recipients who were not typed are excluded. The variable XAge is statistically significant ($p = 0.0143$), with a hazard ratio exceeding 1. Therefore, patients who had a transplant at younger ages lived longer than those who received a transplant later in their lives. The variable XScore has only minimal effect on the survival ($p = 0.1121$).

Output 92.6.2 Heart Transplant Study Analysis II

The PHREG Procedure

Model Information		
Data Set	WORK.HEART	
Dependent Variable	Time	
Censoring Variable	Status	Dead=1 Alive=0
Censoring Value(s)	0	
Ties Handling	BRESLOW	

Number of Observations Read	99
Number of Observations Used	99

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
99	71	28	28.28

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Output 92.6.2 *continued*

Model Fit Statistics		
Criterion	Without Covariates	With Covariates
-2 LOG L	561.680	551.874
AIC	561.680	557.874
SBC	561.680	564.662

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	9.8059	3	0.0203
Score	9.0521	3	0.0286
Wald	9.0554	3	0.0286

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
XStatus	1	-3.19837	1.18746	7.2547	0.0071	0.041
XAge	1	0.05544	0.02263	6.0019	0.0143	1.057
XScore	1	0.44490	0.28001	2.5245	0.1121	1.560

Example 92.7: Time-Dependent Repeated Measurements of a Covariate

(View the complete code for this example (phrex7.sas) in the example repository.)

Repeated determinations can be made during the course of a study of variables thought to be related to survival. Consider an experiment to study the dosing effect of a tumor-promoting agent. Forty-five rodents initially exposed to a carcinogen were randomly assigned to three dose groups. After the first death of an animal, the rodents were examined every week for the number of papillomas. Investigators were interested in determining the effects of dose on the carcinoma incidence after adjusting for the number of papillomas.

The input data set TUMOR consists of the following 19 variables:

- ID (subject identification)
- Time (survival time of the subject)
- Dead (censoring status where 1=dead and 0=censored)
- Dose (dose of the tumor-promoting agent)
- P1–P15 (number of papillomas at the 15 times that animals died. These 15 death times are weeks 27, 34, 37, 41, 43, 45, 46, 47, 49, 50, 51, 53, 65, 67, and 71. For instance, subject 1 died at week 47; it had no papilloma at week 27, five papillomas at week 34, six at week 37, eight at week 41, and 10 at weeks 43, 45, 46, and 47. For an animal that died before week 71, the number of papillomas is missing for those times beyond its death.)

The following SAS statements create the data set TUMOR:

```
data Tumor;
   infile datalines missover;
   input ID Time Dead Dose P1-P15;
   label ID='Subject ID';
   datalines;
 1 47 1  1.0  0  5  6  8 10 10 10 10
 2 71 1  1.0  0  0  0  0  0  0  0  0  1  1  1  1 1 1 1
 3 81 0  1.0  0  1  1  1  1  1  1  1  1  1  1  1 1 1 1
 4 81 0  1.0  0  0  0  0  0  0  0  0  0  0  0  0 0 0 0
 5 81 0  1.0  0  0  0  0  0  0  0  0  0  0  0  0 0 0 0
 6 65 1  1.0  0  0  0  1  1  1  1  1  1  1  1  1 1
 7 71 0  1.0  0  0  0  0  0  0  0  0  0  0  0  0 0 0 0
 8 69 0  1.0  0  0  0  0  0  0  0  0  0  0  0  0 0 0
 9 67 1  1.0  0  0  1  1  2  2  2  2  3  3  3  3 3 3
10 81 0  1.0  0  0  0  0  0  0  0  0  0  0  0  0 0 0 0
11 37 1  1.0  9  9  9
12 81 0  1.0  0  0  0  0  0  0  0  0  0  0  0  0 0 0 0
13 77 0  1.0  0  0  0  0  1  1  1  1  1  1  1  1 1 1 1
14 81 0  1.0  0  0  0  0  0  0  0  0  0  0  0  0 0 0 0
15 81 0  1.0  0  0  0  0  0  0  0  0  0  0  0  0 0 0 0
16 54 0  2.5  0  1  1  1  2  2  2  2  2  2  2  2
17 53 0  2.5  0  0  0  0  0  0  0  0  0  0  0  0
18 38 0  2.5  5 13 14
19 54 0  2.5  2  6  6  6  6  6  6  6  6  6  6  6
20 51 1  2.5 15 15 15 16 16 17 17 17 17 17 17
21 47 1  2.5 13 20 20 20 20 20 20 20
22 27 1  2.5 22
23 41 1  2.5  6 13 13 13
24 49 1  2.5  0  3  3  3  3  3  3  3  3
25 53 0  2.5  0  0  1  1  1  1  1  1  1  1  1  1
26 50 1  2.5  0  0  2  3  4  6  6  6  6  6
27 37 1  2.5  3 15 15
28 49 1  2.5  2  3  3  3  3  4  4  4  4
29 46 1  2.5  4  6  7  9  9  9  9
30 48 0  2.5 15 26 26 26 26 26 26 26
31 54 0 10.0 12 14 15 15 15 15 15 15 15 15 15 15
32 37 1 10.0 12 16 17
33 53 1 10.0  3  6  6  6  6  6  6  6  6  6  6  6
34 45 1 10.0  4 12 15 20 20 20
35 53 0 10.0  6 10 13 13 13 15 15 15 15 15 15 20
36 49 1 10.0  0  2  2  2  2  2  2  2  2
37 39 0 10.0  7  8  8
38 27 1 10.0 17
39 49 1 10.0  0  6  9 14 14 14 14 14 14
40 43 1 10.0 14 18 20 20 20
41 28 0 10.0  8
42 34 1 10.0 11 18
43 45 1 10.0 10 12 16 16 16 16
44 37 1 10.0  0  1  1
45 43 1 10.0  9 19 19 19 19
;
```

The number of papillomas (NPap) for each animal in the study was measured repeatedly over time. One

way of handling time-dependent repeated measurements in the PHREG procedure is to use programming statements to capture the appropriate covariate values of the subjects in each risk set. In this example, NPap is a time-dependent explanatory variable with values that are calculated by means of the programming statements shown in the following SAS statements:

```
proc phreg data=Tumor;
   model Time*Dead(0)=Dose NPap;
   array pp{*} P1-P14;
   array tt{*} t1-t15;
   t1=27;  t2=34;  t3=37;  t4=41;  t5=43;
   t6=45;  t7=46;  t8=47;  t9=49; t10=50;
   t11=51; t12=53; t13=65; t14=67; t15=71;
   if Time <  tt[1]  then NPap=0;
   else if time >= tt[15] then NPap=P15;
   else do i=1 to dim(pp);
      if tt[i] <= Time < tt[i+1] then NPap= pp[i];
   end;
run;
```

At each death time, the NPap value of each subject in the risk set is recalculated to reflect the actual number of papillomas at the given death time. For instance, subject one in the data set Tumor was in the risk sets at weeks 27 and 34; at week 27, the animal had no papilloma, while at week 34, it had five papillomas. Results of the analysis are shown in Output 92.7.1. After the number of papillomas is adjusted for, the dose effect of the tumor-promoting agent is not statistically significant.

Output 92.7.1 Cox Regression Analysis on the Survival of Rodents

The PHREG Procedure

Model Information	
Data Set	WORK.TUMOR
Dependent Variable	Time
Censoring Variable	Dead
Censoring Value(s)	0
Ties Handling	BRESLOW

Number of Observations Read	45
Number of Observations Used	45

Summary of the Number of Event and Censored Values			
Total	Event	Censored	Percent Censored
45	25	20	44.44

Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Output 92.7.1 *continued*

Model Fit Statistics		
Criterion	Without Covariates	With Covariates
-2 LOG L	166.793	143.269
AIC	166.793	147.269
SBC	166.793	149.707

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	23.5243	2	<.0001
Score	28.0498	2	<.0001
Wald	21.1646	2	<.0001

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
Dose	1	0.06885	0.05620	1.5010	0.2205	1.071
NPap	1	0.11714	0.02998	15.2705	<.0001	1.124

Another way to handle time-dependent repeated measurements in the PHREG procedure is to use the counting process style of input. Multiple records are created for each subject, one record for each distinct pattern of the time-dependent measurements. Each record contains a T1 value and a T2 value representing the time interval (T1,T2] during which the values of the explanatory variables remain unchanged. Each record also contains the censoring status at T2.

One advantage of using the counting process formulation is that you can easily obtain various residuals and influence statistics that are not available when programming statements are used to compute the values of the time-dependent variables. On the other hand, creating multiple records for the counting process formulation requires extra effort in data manipulation.

Consider a counting process style of input data set named Tumor1. It contains multiple observations for each subject in the data set Tumor. In addition to variables ID, Time, Dead, and Dose, four new variables are generated:

- T1 (left endpoint of the risk interval)
- T2 (right endpoint of the risk interval)
- NPap (number of papillomas in the time interval (T1,T2])
- Status (censoring status at T2)

For example, five observations are generated for the rodent that died at week 47 and that had no papilloma at week 27, five papillomas at week 34, six at week 37, eight at week 41, and 10 at weeks 43, 45, 46, and 47. The values of T1, T2, NPap, and Status for these five observations are (0,27,0,0), (27,34,5,0), (34,37,6,0), (37,41,8,0), and (41,47,10,1). Note that the variables ID, Time, and Dead are not needed for the estimation of the regression parameters, but they are useful for plotting the residuals.

The following SAS statements create the data set Tumor1:

```
data Tumor1(keep=ID Time Dead Dose T1 T2 NPap Status);
   array pp{*} P1-P14;
   array qq{*} P2-P15;
   array tt{1:15} _temporary_
      (27 34 37 41 43 45 46 47 49 50 51 53 65 67 71);
   set Tumor;
   T1 = 0;
   T2 = 0;
   Status = 0;
   if ( Time = tt[1] ) then do;
      T2 = tt[1];
      NPap = p1;
      Status = Dead;
      output;
   end;
   else do _i_=1 to dim(pp);
      if ( tt[_i_] = Time ) then do;
         T2= Time;
         NPap = pp[_i_];
         Status = Dead;
         output;
      end;
      else if (tt[_i_]  < Time ) then do;
         if (pp[_i_]  ^= qq[_i_] ) then do;
            if qq[_i_]  = . then T2= Time;
            else                 T2= tt[_i_];
            NPap= pp[_i_];
            Status= 0;
            output;
            T1 = T2;
         end;
      end;
   end;
   if ( Time >= tt[15] ) then do;
      T2 = Time;
      NPap = P15;
      Status = Dead;
      output;
   end;
run;
```

In the following SAS statements, the counting process MODEL specification is used. The DFBETA statistics are output to a SAS data set named Out1. Note that Out1 contains multiple observations for each subject—that is, one observation for each risk interval (T1,T2].

```
proc phreg data=Tumor1 noprint;
   model (T1,T2)*Status(0)=Dose NPap;
   output out=Out1 resmart=Mart dfbeta=db1-db2;
   id ID Time Dead;
run;
```

The output from PROC PHREG (not shown) is identical to Output 92.7.1 except for the "Summary of the Number of Event and Censored Values" table. The number of event observations remains unchanged between

the two specifications of PROC PHREG, but the number of censored observations differs due to the splitting of each subject's data into multiple observations for the counting process style of input.

Next, the MEANS procedure sums up the component statistics for each subject and outputs the results to a SAS data set named Out2:

```
proc means data=Out1 noprint;
   by ID Time Dead;
   var Mart db1-db2;
   output out=Out2 sum=Mart db_Dose db_NPap;
run;
```

Finally, DFBETA statistics are plotted against subject ID for easy identification of influential points:

```
title 'DfBetas for Dose';
proc sgplot data=Out2;
   yaxis label="DfBeta" grid;
   refline 0 / axis=y;
   scatter y=db_Dose x=ID;
run;
title 'DfBetas for NPap';
proc sgplot data=Out2;
   yaxis label="DfBeta" grid;
   refline 0 / axis=y;
   scatter y=db_NPap x=ID;
run;
```

The plots of the DFBETA statistics are shown in Output 92.7.2 and Output 92.7.3. Subject 30 appears to have a large influence on both the Dose and NPap coefficients. Subjects 31 and 35 have considerable influences on the DOSE coefficient, while subjects 22 and 44 have rather large influences on the NPap coefficient.

Output 92.7.2 Plot of DFBETA Statistic for DOSE versus Subject Number

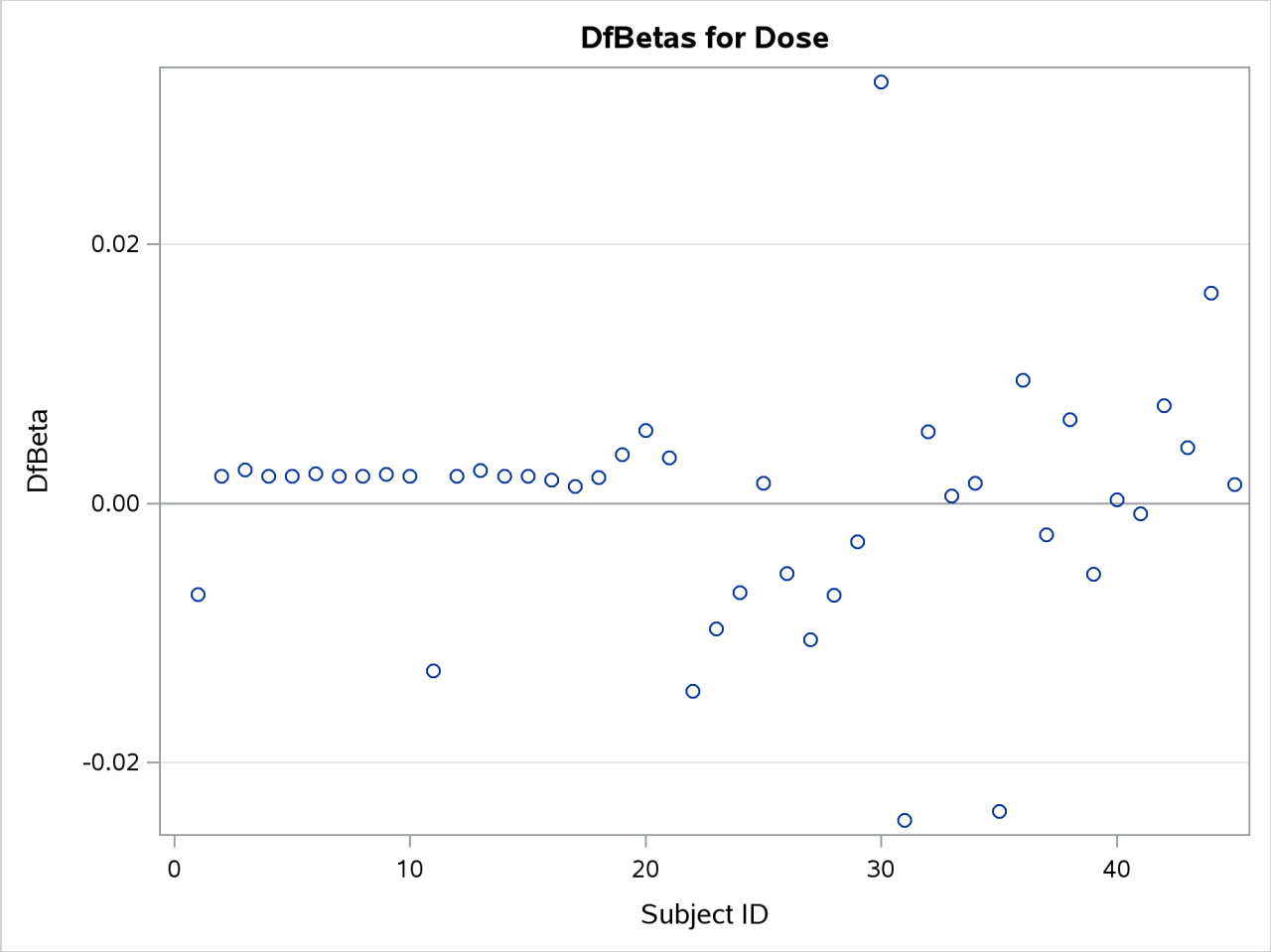

Output 92.7.3 Plot of DFBETA Statistic for NPAP versus Subject Number

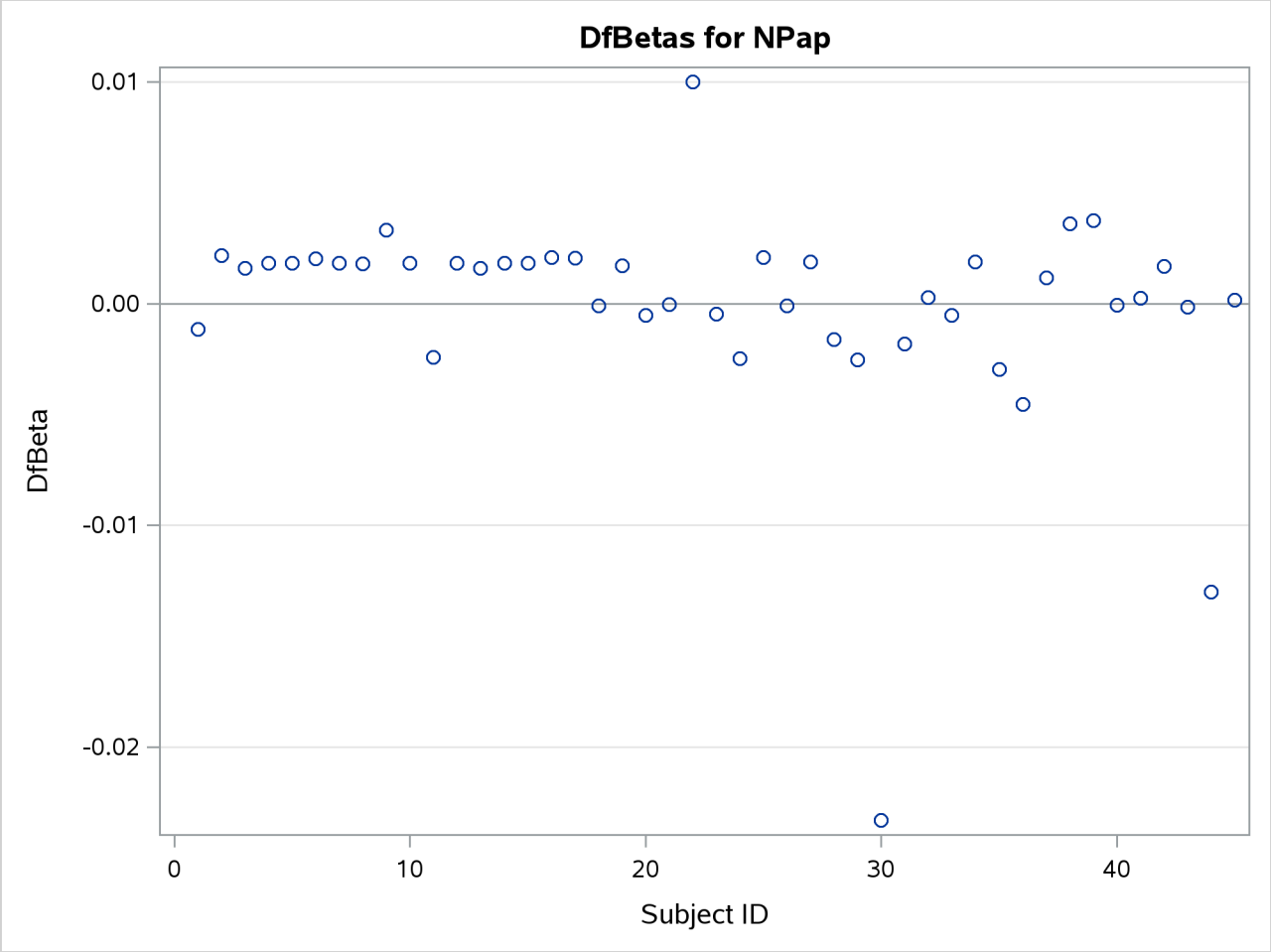

Example 92.8: Survival Curves

(View the complete code for this example (phrex8.sas) in the example repository.)

You might want to use your regression analysis results to predict the survivorship of subjects of specific covariate values. The COVARIATES= data set in the BASELINE statement enables you to specify the sets of covariate values for the prediction. On the other hand, you might want to summarize the survival experience of an average patient for a given population. The DIRADJ option in the BASELINE statement computes the direct adjusted survival curve that averages the estimated survival curves for patients whose covariates are represented in the COVARIATES= data set. By using the PLOTS= option in the PROC PHREG statement, you can use ODS Graphics to display the predicted survival curves. You can elect to output the predicted survival curves in a SAS data set by optionally specifying the OUT= option in the BASELINE statement. This example illustrates how to obtain the covariate-specific survival curves and the direct adjusted survival curve by using the Myeloma data set in Example 92.1, where variables LogBUN and HGB were identified as the most important prognostic factors. Suppose you want to compute the predicted survival curves for two sets of covariate values: (LogBUN=1.0, HGB=10) and (LogBUN=1.8, HGB=12). These values are saved in

the data set lnrisks in the following DATA step. Also created in this data set is the variable Id, whose values will be used in identifying the covariate sets in the survival plot.

```
data Inrisks;
   length Id $20;
   input LogBUN HGB Id $12-31;
   datalines;
1.00 10.0  logBUN=1.0 HGB=10
1.80 12.0  logBUN=1.8 HGB=12
;
```

The following statements plot the survival functions in Output 92.8.1 and save the survival estimates in the data set Pred1:

```
ods graphics on;
proc phreg data=Myeloma plots(overlay)=survival;
   model Time*VStatus(0)=LogBUN HGB;
   baseline covariates=Inrisks out=Pred1 survival=_all_/rowid=Id;
run;
```

The COVARIATES= option in the BASELINE statement specifies the data set that contains the set of covariates of interest. The PLOTS= option in the PROC PHREG statement creates the survival plot. The OVERLAY suboption overlays the two curves in the same plot. If the OVERLAY suboption is not specified, each curve is displayed in a separate plot. The ROWID= option in the BASELINE statement specifies that the values of the variable Id in the COVARIATES= data set be used to identify the curves in the plot. The SURVIVAL=_ALL_ option in the BASELINE statement requests that the estimated survivor function, standard error, and lower and upper confidence limits for the survivor function be output into the SAS data set that is specified in the OUT= option.

The survival Plot (Output 92.8.1) contains two curves, one for each of row of covariates in the data set lnrisks.

Output 92.8.1 Estimated Survivor Function Plot

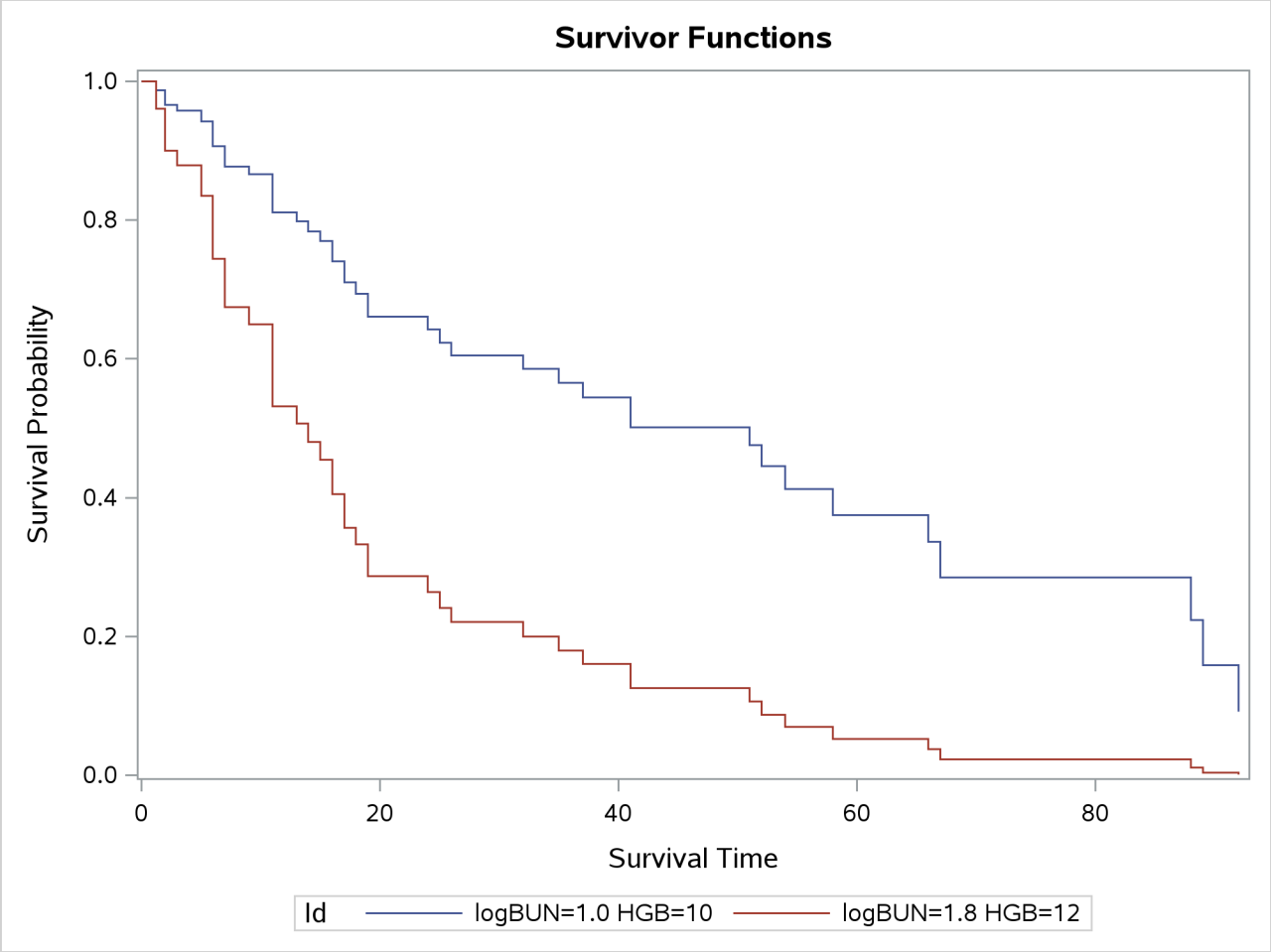

The following statements print out the observations in the data set Pred1 for the realization LogBUN=1.00 and HGB=10.0:

```
proc print data=Pred1(where=(logBUN=1 and HGB=10));
run;
```

As shown in Output 92.8.2, 32 observations represent the survivor function for the realization LogBUN=1.00 and HGB=10.0. The first observation has survival time 0 and survivor function estimate 1.0. Each of the remaining 31 observations represents a distinct event time in the input data set Myeloma. These observations are presented in ascending order of the event times. Note that all the variables in the COVARIATES=InRisks data set are included in the OUT=Pred1 data set. Likewise, you can print out the observations that represent the survivor function for the realization LogBUN=1.80 and HGB=12.0.

Output 92.8.2 Survivor Function Estimates for LogBUN=1.0 and HGB=10.0

Obs	Id	LogBUN	HGB	Time	Survival	StdErrSurvival	LowerSurvival	UpperSurvival
1	logBUN=1.0 HGB=10	1	10	0.00	1.00000	.	.	.
2	logBUN=1.0 HGB=10	1	10	1.25	0.98678	0.01043	0.96655	1.00000
3	logBUN=1.0 HGB=10	1	10	2.00	0.96559	0.01907	0.92892	1.00000
4	logBUN=1.0 HGB=10	1	10	3.00	0.95818	0.02180	0.91638	1.00000
5	logBUN=1.0 HGB=10	1	10	5.00	0.94188	0.02747	0.88955	0.99729
6	logBUN=1.0 HGB=10	1	10	6.00	0.90635	0.03796	0.83492	0.98389
7	logBUN=1.0 HGB=10	1	10	7.00	0.87742	0.04535	0.79290	0.97096
8	logBUN=1.0 HGB=10	1	10	9.00	0.86646	0.04801	0.77729	0.96585
9	logBUN=1.0 HGB=10	1	10	11.00	0.81084	0.05976	0.70178	0.93686
10	logBUN=1.0 HGB=10	1	10	13.00	0.79800	0.06238	0.68464	0.93012
11	logBUN=1.0 HGB=10	1	10	14.00	0.78384	0.06515	0.66601	0.92251
12	logBUN=1.0 HGB=10	1	10	15.00	0.76965	0.06779	0.64762	0.91467
13	logBUN=1.0 HGB=10	1	10	16.00	0.74071	0.07269	0.61110	0.89781
14	logBUN=1.0 HGB=10	1	10	17.00	0.71005	0.07760	0.57315	0.87966
15	logBUN=1.0 HGB=10	1	10	18.00	0.69392	0.07998	0.55360	0.86980
16	logBUN=1.0 HGB=10	1	10	19.00	0.66062	0.08442	0.51425	0.84865
17	logBUN=1.0 HGB=10	1	10	24.00	0.64210	0.08691	0.49248	0.83717
18	logBUN=1.0 HGB=10	1	10	25.00	0.62360	0.08921	0.47112	0.82542
19	logBUN=1.0 HGB=10	1	10	26.00	0.60523	0.09136	0.45023	0.81359
20	logBUN=1.0 HGB=10	1	10	32.00	0.58549	0.09371	0.42784	0.80122
21	logBUN=1.0 HGB=10	1	10	35.00	0.56534	0.09593	0.40539	0.78840
22	logBUN=1.0 HGB=10	1	10	37.00	0.54465	0.09816	0.38257	0.77542
23	logBUN=1.0 HGB=10	1	10	41.00	0.50178	0.10166	0.33733	0.74639
24	logBUN=1.0 HGB=10	1	10	51.00	0.47546	0.10368	0.31009	0.72901
25	logBUN=1.0 HGB=10	1	10	52.00	0.44510	0.10522	0.28006	0.70741
26	logBUN=1.0 HGB=10	1	10	54.00	0.41266	0.10689	0.24837	0.68560
27	logBUN=1.0 HGB=10	1	10	58.00	0.37465	0.10891	0.21192	0.66232
28	logBUN=1.0 HGB=10	1	10	66.00	0.33626	0.10980	0.17731	0.63772
29	logBUN=1.0 HGB=10	1	10	67.00	0.28529	0.11029	0.13372	0.60864
30	logBUN=1.0 HGB=10	1	10	88.00	0.22412	0.10928	0.08619	0.58282
31	logBUN=1.0 HGB=10	1	10	89.00	0.15864	0.10317	0.04435	0.56750
32	logBUN=1.0 HGB=10	1	10	92.00	0.09180	0.08545	0.01481	0.56907

Next, the DIRADJ option in the BASELINE statement is used to request a survival curve that represents the survival experience of an average patient in the population in which the COVARIATES= data set is sampled. When the DIRADJ option is specified, PROC PHREG computes the direct adjusted survival function by averaging the predicted survival functions for the rows in the COVARIATES= data set. The following statements plot the direct adjusted survival function in Output 92.8.3.

```
proc phreg data=Myeloma plots=survival;
   model Time*VStatus(0)=LogBUN HGB;
   baseline covariates=Myeloma survival=_all_/diradj;
run;
```

When the DIRADJ option is specified in the BASELINE statement, the default COVARIATES= data set is the input data set. For clarity, the COVARIATES=MYELOMA is specified in the BASELINE statement in the preceding PROC PHREG call.

Output 92.8.3 Average Survival Function for the Myeloma Data

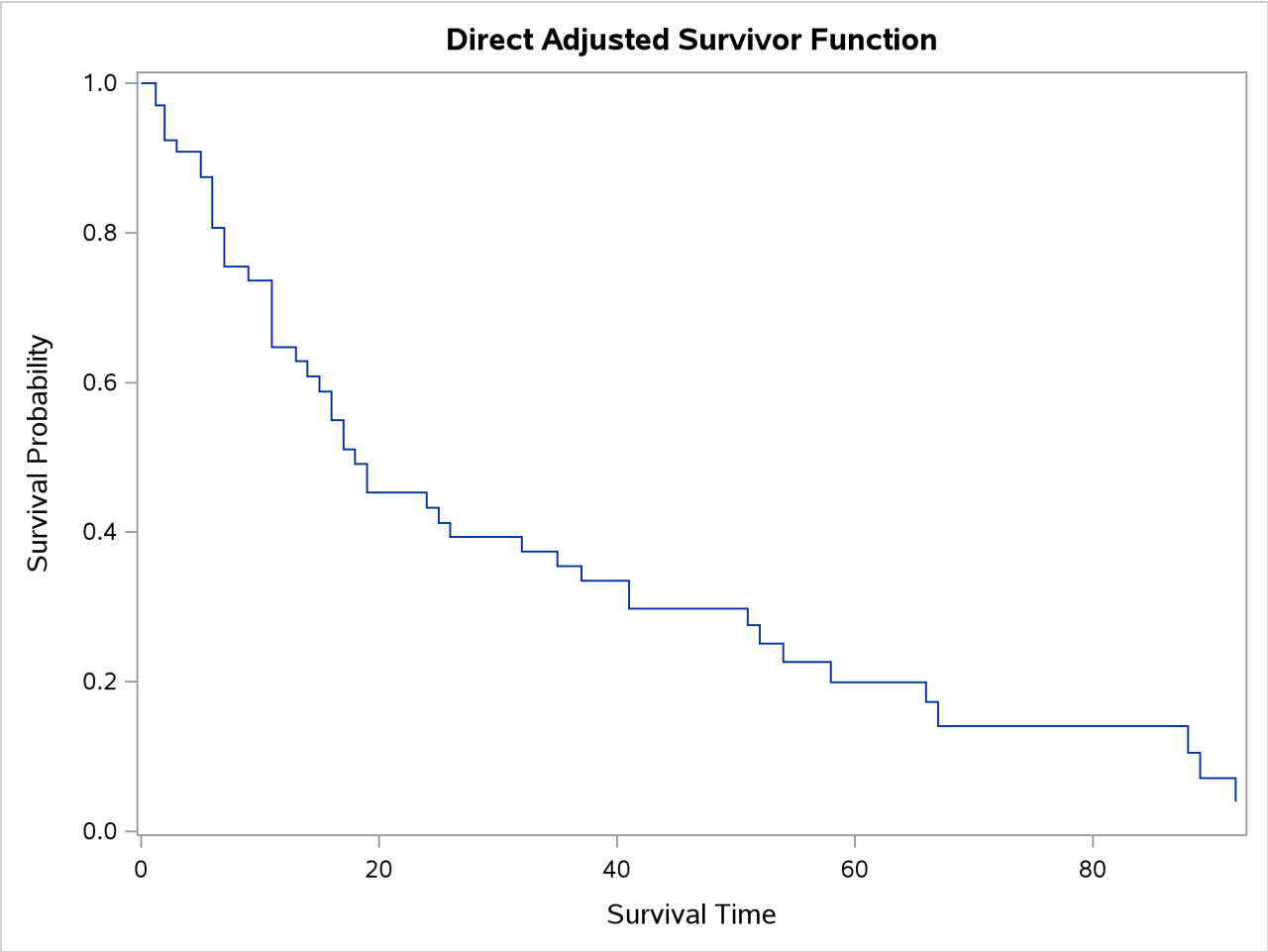

If neither the COVARIATES= data set nor the DIRADJ option is specified in the BASELINE statement, PROC PHREG computes a predicted survival curve based on $\bar{\mathbf{Z}}$, the average values of the covariate vectors in the input data (Neuberger et al. 1986). This curve represents the survival experience of a patient with an average prognostic index $\boldsymbol{\beta}'\bar{\mathbf{Z}}$ equal to the average prognostic index of all patients. This approach has a couple of drawbacks: it is possible that no patient could ever have such an average index, and it does not account for the variability in the prognostic factor from patient to patient.

The DIRADJ option is particularly useful if the model contains a categorical explanatory variable that represents different treatments of interest. By specifying this categorical variable in the GROUP= option, you obtain a direct adjusted survival curve for each category of the variable. In addition, you can use the OUTDIFF= option to save all pairwise differences of these direct adjusted survival probabilities in a data set. For illustration, consider a model that also includes the categorical variable Frac, which has a value 0 if a patient did not have a fracture at diagnosis and 1 otherwise, as an explanatory variable. The following statements plot the adjusted survival curves in Output 92.8.4 and save the differences of the direct adjusted survival probabilities in the data set Diff1:

```
proc phreg data=Myeloma plots(overlay)=survival;
   class Frac;
   model Time*VStatus(0)=LogBUN HGB Frac;
   baseline covariates=Myeloma outdiff=Diff1 survival=_all_/diradj group=Frac;
run;
```

Because the CLASS variable Frac is specified as the GROUP= variable, a separate direct adjusted survival curve is computed for each value of the variable Frac. Each direct adjusted survival curve is the average of the predicted survival curves for all the patients in the entire Myeloma data set with their Frac value set to a specific constant. For example, the direct adjusted survival curve for Frac=0 (no fracture at diagnosis) is computed as follows:

1. The value of the variable Frac is set to 0 for all observations in the Myeloma data set.
2. The survival curve for each observation in the modified data set is computed.
3. All the survival curves computed in step 2 are averaged.

Output 92.8.4 Average Survival by Fracture Status

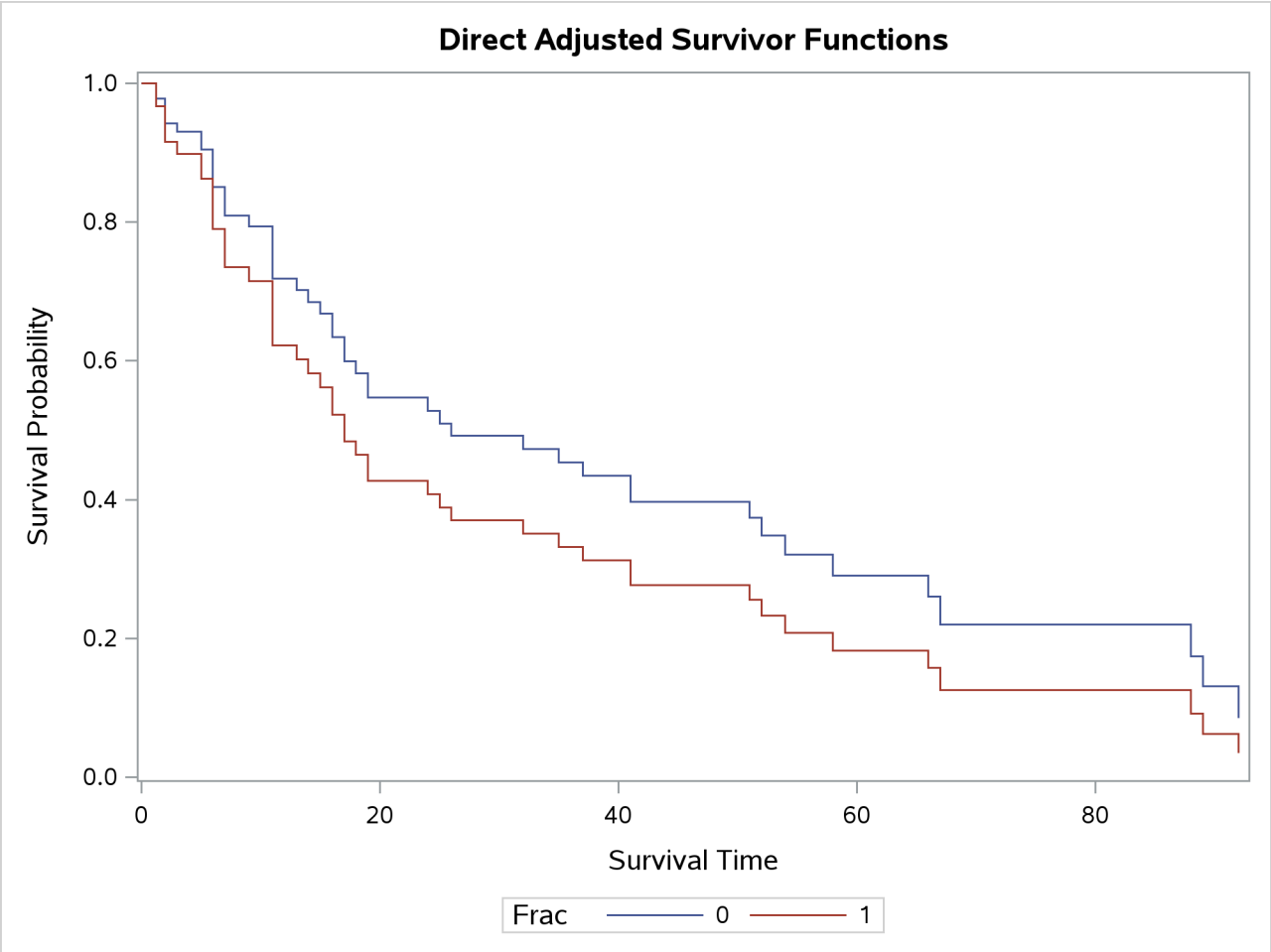

Output 92.8.4 shows that patients without fracture at diagnosis have better survival than those with fractures.

Differences in the survival probabilities and their standard errors are displayed in Output 92.8.5.

```
proc print data=Diff1;
run;
```

Output 92.8.5 Differences in the Survival between Fracture and Nonfracture

Obs	Frac	Frac2	Time	SurvDiff	StdErr
1	0	1	1.25	0.01074	0.01199
2	0	1	2.00	0.02653	0.02605
3	0	1	3.00	0.03165	0.03063
4	0	1	5.00	0.04191	0.03963
5	0	1	6.00	0.06115	0.05669
6	0	1	7.00	0.07416	0.06853
7	0	1	9.00	0.07854	0.07261
8	0	1	11.00	0.09669	0.09002
9	0	1	13.00	0.09998	0.09327
10	0	1	14.00	0.10319	0.09644
11	0	1	15.00	0.10611	0.09937
12	0	1	16.00	0.11117	0.10464
13	0	1	17.00	0.11532	0.10922
14	0	1	18.00	0.11704	0.11120
15	0	1	19.00	0.11969	0.11447
16	0	1	24.00	0.12072	0.11593
17	0	1	25.00	0.12145	0.11713
18	0	1	26.00	0.12189	0.11808
19	0	1	32.00	0.12208	0.11883
20	0	1	35.00	0.12197	0.11933
21	0	1	37.00	0.12155	0.11956
22	0	1	41.00	0.11983	0.11924
23	0	1	51.00	0.11821	0.11850
24	0	1	52.00	0.11580	0.11714
25	0	1	54.00	0.11262	0.11507
26	0	1	58.00	0.10824	0.11203
27	0	1	66.00	0.10301	0.10814
28	0	1	67.00	0.09451	0.10130
29	0	1	88.00	0.08248	0.09133
30	0	1	89.00	0.06847	0.08033
31	0	1	92.00	0.05038	0.06515

Example 92.9: Analysis of Residuals

(View the complete code for this example (phrex9.sas) in the example repository.)

Residuals are used to investigate the lack of fit of a model to a given subject. You can obtain martingale and deviance residuals for the Cox proportional hazards regression analysis by requesting that they be included in the OUTPUT data set. You can plot these statistics and look for outliers.

Consider the stepwise regression analysis performed in Example 92.1. The final model included variables LogBUN and HGB. You can generate residual statistics for this analysis by refitting the model containing those variables and including an OUTPUT statement as in the following invocation of PROC PHREG. The *keywords* XBETA, RESMART, and RESDEV identify new variables that contain the linear predictor scores $\mathbf{z}'_j\hat{\boldsymbol{\beta}}$, martingale residuals, and deviance residuals. These variables are xb, mart, and dev, respectively.

```
proc phreg data=Myeloma noprint;
   model Time*Vstatus(0)=LogBUN HGB;
   output out=Outp xbeta=Xb resmart=Mart resdev=Dev;
run;
```

The following statements plot the residuals against the linear predictor scores:

```
title "Myeloma Study";
proc sgplot data=Outp;
   yaxis grid;
   refline 0 / axis=y;
   scatter y=Mart x=Xb;
run;
proc sgplot data=Outp;
   yaxis grid;
   refline 0 / axis=y;
   scatter y=Dev x=Xb;
run;
```

The resulting plots are shown in Output 92.9.1 and Output 92.9.2. The martingale residuals are skewed because of the single event setting of the Cox model. The martingale residual plot shows an isolation point (with linear predictor score 1.09 and martingale residual –3.37), but this observation is no longer distinguishable in the deviance residual plot. In conclusion, there is no indication of a lack of fit of the model to individual observations.

Output 92.9.1 Martingale Residual Plot

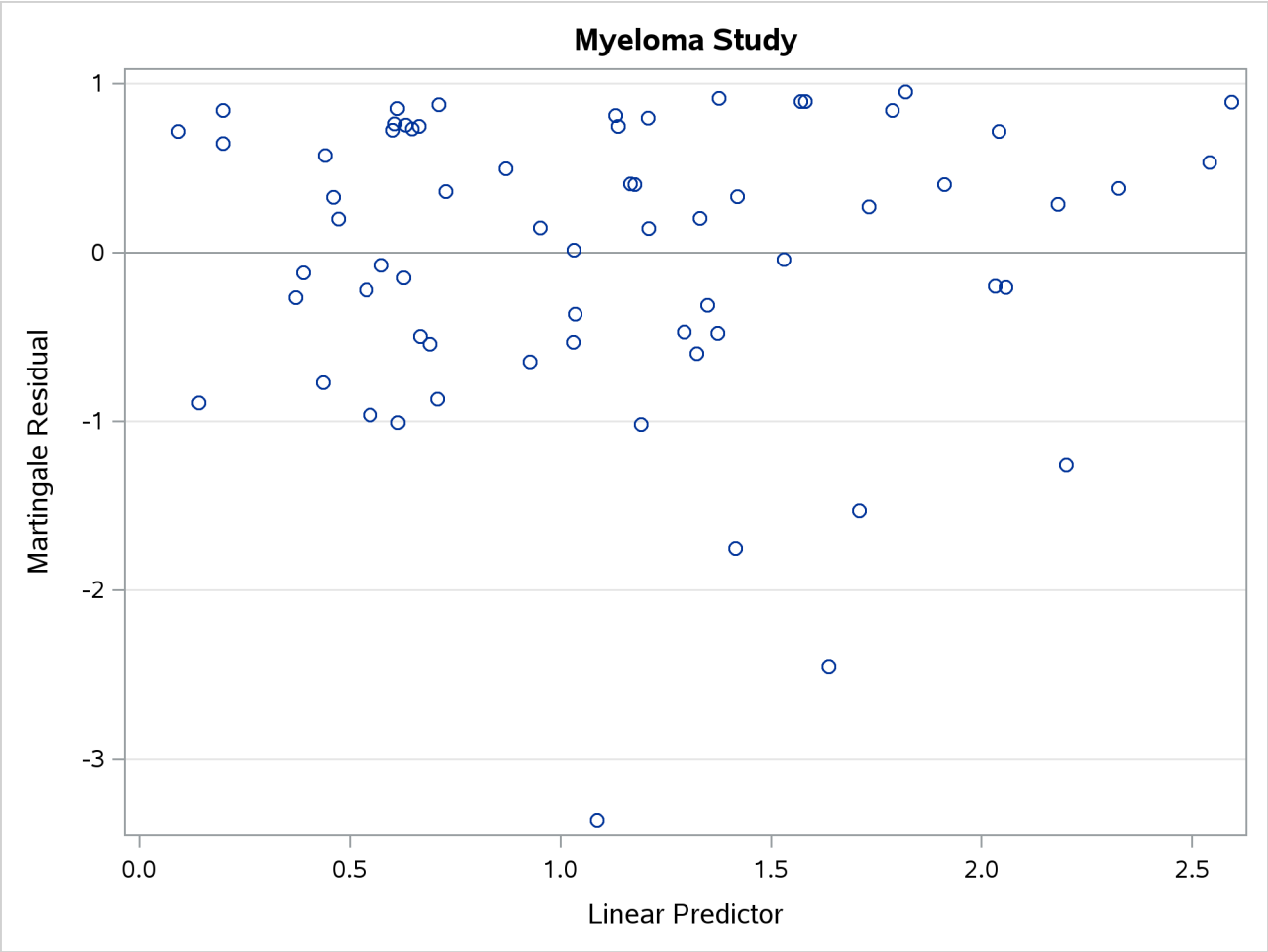

Output 92.9.2 Deviance Residual Plot

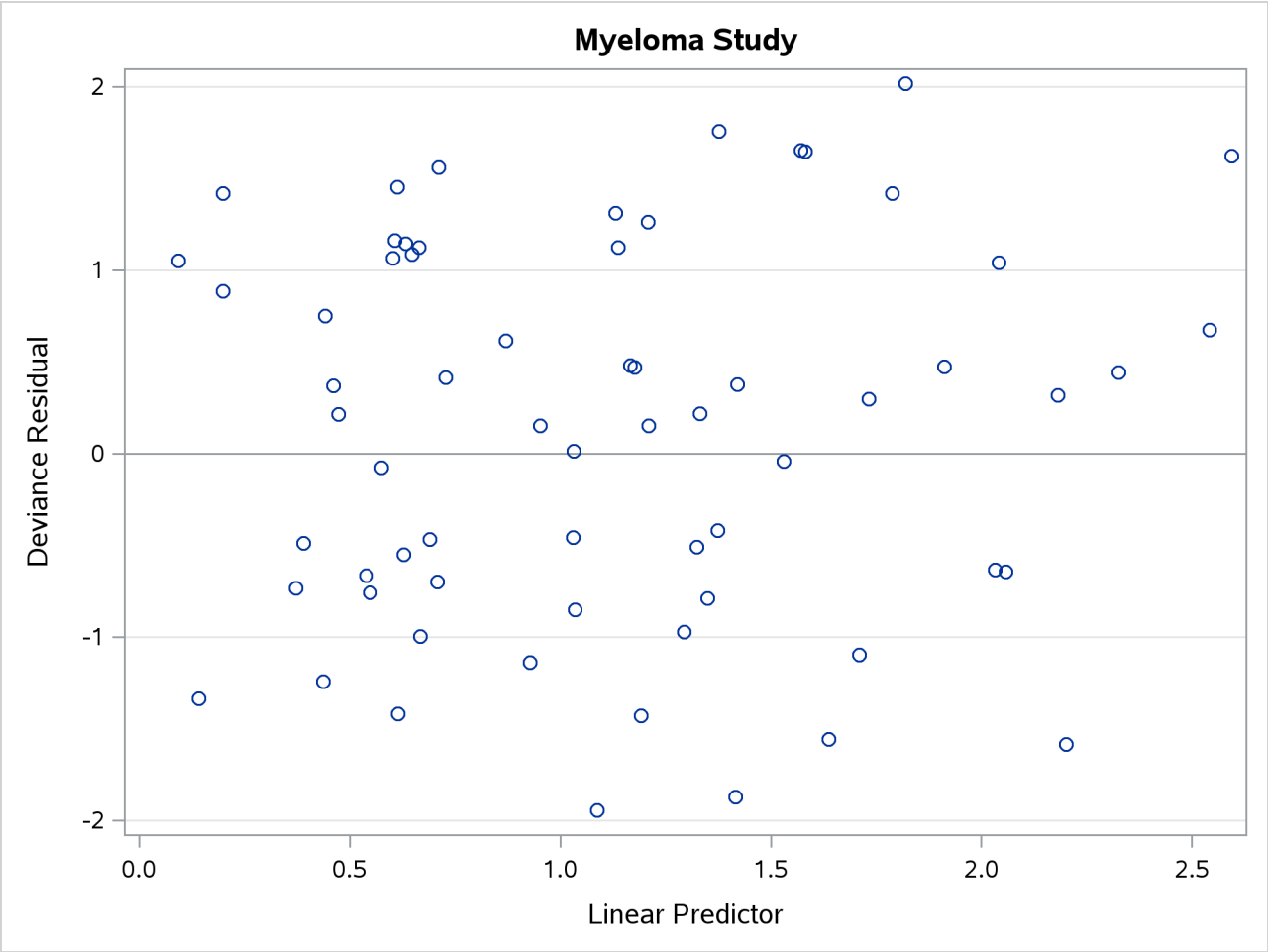

Example 92.10: Analysis of Recurrent Events Data

(View the complete code for this example (phrex10.sas) in the example repository.)

Recurrent events data consist of times to a number of repeated events for each sample unit—for example, times of recurrent episodes of a disease in patients. Various ways of analyzing recurrent events data are described in the section "Analysis of Multivariate Failure Time Data" on page 7574. The bladder cancer data listed in Wei, Lin, and Weissfeld (1989) are used here to illustrate these methods.

The data consist of 86 patients with superficial bladder tumors, which were removed when the patients entered the study. Of these patients, 48 were randomized into the placebo group, and 38 were randomized into the group receiving thiotepa. Many patients had multiple recurrences of tumors during the study, and new tumors were removed at each visit. The data set contains the first four recurrences of the tumor for each patient, and each recurrence time was measured from the patient's entry time into the study.

The data consist of the following eight variables:

- Trt, treatment group (1=placebo and 2=thiotepa)

- Time, follow-up time
- Number, number of initial tumors
- Size, initial tumor size
- T1, T2, T3, and T4, times of the four potential recurrences of the bladder tumor. A patient with only two recurrences has missing values in T3 and T4.

In the data set Bladder, four observations are created for each patient, one for each of the four potential tumor recurrences. In addition to values of Trt, Number, and Size for the patient, each observation contains the following variables:

- ID, patient's identification (which is the sequence number of the subject)
- Visit, visit number (with value k for the kth potential tumor recurrence)
- TStart, time of the $(k-1)$ recurrence for Visit=k, or the entry time 0 if VISIT=1, or the follow-up time if the $(k-1)$ recurrence does not occur
- TStop, time of the kth recurrence if Visit=k or follow-up time if the kth recurrence does not occur
- Status, event status of TStop (1=recurrence and 0=censored)

For instance, a patient with only one recurrence time at month 6 who was followed until month 10 will have values for Visit, TStart, TStop, and Status of (1,0,6,1), (2,6,10,0), (3,10,10,0), and (4,10,10,0), respectively. The last two observations are redundant for the intensity model and the proportional means model, but they are important for the analysis of the marginal Cox models. If the follow-up time is beyond the time of the fourth tumor recurrence, it is tempting to create a fifth observation with the time of the fourth tumor recurrence as the TStart value, the follow-up time as the TStop value, and a Status value of 0. However, Therneau and Grambsch (2000, Section 8.5) have warned against incorporating such observations into the analysis.

The following SAS statements create the data set Bladder:

```
data Bladder;
   keep ID TStart TStop Status Trt Number Size Visit;
   retain ID TStart 0;
   array tt[4] T1-T4;
   infile datalines missover;
   input Trt Time Number Size T1-T4;
   ID + 1;
   TStart=0;
   do Visit=1 to 4;
      if tt[Visit] = . then do;
         TStop=Time;
         Status=0;
      end;
      else do;
         TStop=tt[Visit];
         Status=1;
      end;
```

```
      output;
      TStart=TStop;
   end;
   if (TStart < Time) then delete;
   datalines;
1       0       1     1
1       1       1     3
1       4       2     1
1       7       1     1
1       10      5     1
1       10      4     1   6
1       14      1     1
1       18      1     1
1       18      1     3   5
1       18      1     1   12  16
1       23      3     3
1       23      1     3   10  15
1       23      1     1   3   16  23
1       23      3     1   3   9   21
1       24      2     3   7   10  16  24
1       25      1     1   3   15  25
1       26      1     2
1       26      8     1   1
1       26      1     4   2   26
1       28      1     2   25
1       29      1     4
1       29      1     2
1       29      4     1
1       30      1     6   28  30
1       30      1     5   2   17  22
1       30      2     1   3   6   8   12
1       31      1     3   12  15  24
1       32      1     2
1       34      2     1
1       36      2     1
1       36      3     1   29
1       37      1     2
1       40      4     1   9   17  22  24
1       40      5     1   16  19  23  29
1       41      1     2
1       43      1     1   3
1       43      2     6   6
1       44      2     1   3   6   9
1       45      1     1   9   11  20  26
1       48      1     1   18
1       49      1     3
1       51      3     1   35
1       53      1     7   17
1       53      3     1   3   15  46  51
1       59      1     1
1       61      3     2   2   15  24  30
1       64      1     3   5   14  19  27
1       64      2     3   2   8   12  13
2       1       1     3
```

```
2       1       1     1
2       5       8     1   5
2       9       1     2
2       10      1     1
2       13      1     1
2       14      2     6   3
2       17      5     3   1   3   5   7
2       18      5     1
2       18      1     3   17
2       19      5     1   2
2       21      1     1   17  19
2       22      1     1
2       25      1     3
2       25      1     5
2       25      1     1
2       26      1     1   6   12  13
2       27      1     1   6
2       29      2     1   2
2       36      8     3   26  35
2       38      1     1
2       39      1     1   22  23  27  32
2       39      6     1   4   16  23  27
2       40      3     1   24  26  29  40
2       41      3     2
2       41      1     1
2       43      1     1   1   27
2       44      1     1
2       44      6     1   2   20  23  27
2       45      1     2
2       46      1     4   2
2       46      1     4
2       49      3     3
2       50      1     1
2       50      4     1   4   24  47
2       54      3     4
2       54      2     1   38
2       59      1     3
;
```

First, consider fitting the intensity model (Andersen and Gill 1982) and the proportional means model (Lin et al. 2000). The counting process style of input is used in the PROC PHREG specification. For the proportional means model, inference is based on the robust sandwich covariance estimate, which is requested by the COVS(AGGREGATE) option in the PROC PHREG statement. The COVM option is specified for the analysis of the intensity model to use the model-based covariance estimate. Note that some of the observations in the data set Bladder have a degenerated interval of risk. The presence of these observations does not affect the results of the analysis since none of these observations are included in any of the risk sets. However, the procedure will run more efficiently without these observations; consequently, in the following SAS statements, the WHERE clause is used to eliminate these redundant observations:

```
title 'Intensity Model and Proportional Means Model';
proc phreg data=Bladder covm covs(aggregate);
   model (TStart, TStop) * Status(0) = Trt Number Size;
   id id;
```

```
   where TStart < TStop;
run;
```

Results of fitting the intensity model and the proportional means model are shown in Output 92.10.1 and Output 92.10.2, respectively. The robust sandwich standard error estimate for Trt is larger than its model-based counterpart, rendering the effect of thiotepa less significant in the proportional means model ($p = 0.0747$) than in the intensity model ($p = 0.0215$).

Output 92.10.1 Analysis of the Intensity Model

Intensity Model and Proportional Means Model

The PHREG Procedure

Analysis of Maximum Likelihood Estimates with Model-Based Variance Estimate

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
Trt	1	-0.45979	0.19996	5.2873	0.0215	0.631
Number	1	0.17165	0.04733	13.1541	0.0003	1.187
Size	1	-0.04256	0.06903	0.3801	0.5375	0.958

Output 92.10.2 Analysis of the Proportional Means Model

Analysis of Maximum Likelihood Estimates with Sandwich Variance Estimate

Parameter	DF	Parameter Estimate	Standard Error	StdErr Ratio	Chi-Square	Pr > ChiSq	Hazard Ratio
Trt	1	-0.45979	0.25801	1.290	3.1757	0.0747	0.631
Number	1	0.17165	0.06131	1.296	7.8373	0.0051	1.187
Size	1	-0.04256	0.07555	1.094	0.3174	0.5732	0.958

Next, consider the conditional models of Prentice, Williams, and Peterson (1981). In the PWP models, the risk set for the $(k + 1)$ recurrence is restricted to those patients who have experienced the first k recurrences. For example, a patient who experienced only one recurrence is an event observation for the first recurrence; this patient is a censored observation for the second recurrence and should not be included in the risk set for the third or fourth recurrence. The following DATA step eliminates those observations that should not be in the risk sets, forming a new input data set (named Bladder2) for fitting the PWP models. The variable Gaptime, representing the gap times between successive recurrences, is also created.

```
data Bladder2(drop=LastStatus);
   retain LastStatus;
   set Bladder;
   by ID;
   if first.id then LastStatus=1;
   if (Status=0 and LastStatus=0) then delete;
   LastStatus=Status;
   Gaptime=Tstop-Tstart;
run;
```

The following statements fit the PWP total time model. The variables Trt1, Trt2, Trt3, and Trt4 are visit-specific variables for Trt; the variables Number1, Number2, Numvber3, and Number4 are visit-specific variables for

Number; and the variables Size1, Size2, Size3, and Size4 are visit-specific variables for Size.

```
title 'PWP Total Time Model with Noncommon Effects';
proc phreg data=Bladder2;
   model (TStart,Tstop) * Status(0) = Trt1-Trt4 Number1-Number4
                                      Size1-Size4;
   Trt1= Trt * (Visit=1);
   Trt2= Trt * (Visit=2);
   Trt3= Trt * (Visit=3);
   Trt4= Trt * (Visit=4);
   Number1= Number * (Visit=1);
   Number2= Number * (Visit=2);
   Number3= Number * (Visit=3);
   Number4= Number * (Visit=4);
   Size1= Size * (Visit=1);
   Size2= Size * (Visit=2);
   Size3= Size * (Visit=3);
   Size4= Size * (Visit=4);
   strata Visit;
run;
```

Results of the analysis of the PWP total time model are shown in Output 92.10.3. There is no significant treatment effect on the total time in any of the four tumor recurrences.

Output 92.10.3 Analysis of the PWP Total Time Model with Noncommon Effects

PWP Total Time Model with Noncommon Effects

The PHREG Procedure

Summary of the Number of Event and Censored Values

Stratum	Visit	Total	Event	Censored	Percent Censored
1	1	85	47	38	44.71
2	2	46	29	17	36.96
3	3	27	22	5	18.52
4	4	20	14	6	30.00
Total		178	112	66	37.08

Output 92.10.3 *continued*

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
Trt1	1	-0.51757	0.31576	2.6868	0.1012	0.596
Trt2	1	-0.45967	0.40642	1.2792	0.2581	0.631
Trt3	1	0.11700	0.67183	0.0303	0.8617	1.124
Trt4	1	-0.04059	0.79251	0.0026	0.9592	0.960
Number1	1	0.23605	0.07607	9.6287	0.0019	1.266
Number2	1	-0.02044	0.09052	0.0510	0.8213	0.980
Number3	1	0.01219	0.18208	0.0045	0.9466	1.012
Number4	1	0.18915	0.24443	0.5989	0.4390	1.208
Size1	1	0.06790	0.10125	0.4498	0.5024	1.070
Size2	1	-0.15425	0.12300	1.5728	0.2098	0.857
Size3	1	0.14891	0.26299	0.3206	0.5713	1.161
Size4	1	0.0000732	0.34297	0.0000	0.9998	1.000

The following statements fit the PWP gap-time model:

```
title 'PWP Gap-Time Model with Noncommon Effects';
proc phreg data=Bladder2;
   model Gaptime * Status(0) = Trt1-Trt4 Number1-Number4
                               Size1-Size4;
   Trt1= Trt * (Visit=1);
   Trt2= Trt * (Visit=2);
   Trt3= Trt * (Visit=3);
   Trt4= Trt * (Visit=4);
   Number1= Number * (Visit=1);
   Number2= Number * (Visit=2);
   Number3= Number * (Visit=3);
   Number4= Number * (Visit=4);
   Size1= Size * (Visit=1);
   Size2= Size * (Visit=2);
   Size3= Size * (Visit=3);
   Size4= Size * (Visit=4);
   strata Visit;
run;
```

Results of the analysis of the PWP gap-time model are shown in Output 92.10.4. Note that the regression coefficients for the first tumor recurrence are the same as those of the total time model, since the total time and the gap time are the same for the first recurrence. There is no significant treatment effect on the gap times for any of the four tumor recurrences.

Output 92.10.4 Analysis of the PWP Gap-Time Model with Noncommon Effects

PWP Gap-Time Model with Noncommon Effects

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
Trt1	1	-0.51757	0.31576	2.6868	0.1012	0.596
Trt2	1	-0.25911	0.40511	0.4091	0.5224	0.772
Trt3	1	0.22105	0.54909	0.1621	0.6873	1.247
Trt4	1	-0.19498	0.64184	0.0923	0.7613	0.823
Number1	1	0.23605	0.07607	9.6287	0.0019	1.266
Number2	1	-0.00571	0.09667	0.0035	0.9529	0.994
Number3	1	0.12935	0.15970	0.6561	0.4180	1.138
Number4	1	0.42079	0.19816	4.5091	0.0337	1.523
Size1	1	0.06790	0.10125	0.4498	0.5024	1.070
Size2	1	-0.11636	0.11924	0.9524	0.3291	0.890
Size3	1	0.24995	0.23113	1.1695	0.2795	1.284
Size4	1	0.03557	0.29043	0.0150	0.9025	1.036

You can fit the PWP total time model with common effects by using the following SAS statements. However, the analysis is not shown here.

```
title2 'PWP Total Time Model with Common Effects';
proc phreg data=Bladder2;
   model (tstart,tstop) * status(0) = Trt Number Size;
   strata Visit;
run;
```

You can fit the PWP gap-time model with common effects by using the following statements. Again, the analysis is not shown here.

```
title2 'PWP Gap Time Model with Common Effects';
proc phreg data=Bladder2;
   model Gaptime * Status(0) = Trt Number Size;
   strata Visit;
run;
```

Recurrent events data are a special case of multiple events data in which the recurrence times are regarded as multivariate failure times and the marginal approach of Wei, Lin, and Weissfeld (1989) can be used. WLW fits a Cox model to each of the component times and makes statistical inference of the regression parameters based on a robust sandwich covariance matrix estimate. No specific correlation structure is imposed on the multivariate failure times. For the kth marginal model, let $\boldsymbol{\beta}_k$ denote the row vector of regression parameters, let $\hat{\boldsymbol{\beta}}_k$ denote the maximum likelihood estimate of $\boldsymbol{\beta}_k$, let $\hat{\mathbf{A}}_k$ denote the covariance matrix obtained by inverting the observed information matrix, and let $\mathbf{R}_k$ denote the matrix of score residuals. WLW showed that the joint distribution of $(\hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_4)'$ can be approximated by a multivariate normal distribution with

mean vector $(\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_4)'$ and robust covariance matrix

$$\begin{pmatrix} \mathbf{V}_{11} & \mathbf{V}_{12} & \mathbf{V}_{13} & \mathbf{V}_{14} \\ \mathbf{V}_{21} & \mathbf{V}_{22} & \mathbf{V}_{23} & \mathbf{V}_{24} \\ \mathbf{V}_{31} & \mathbf{V}_{32} & \mathbf{V}_{33} & \mathbf{V}_{34} \\ \mathbf{V}_{41} & \mathbf{V}_{42} & \mathbf{V}_{43} & \mathbf{V}_{44} \end{pmatrix}$$

with the submatrix $\mathbf{V}_{ij}$ given by

$$\mathbf{V}_{ij} = \hat{\mathbf{A}}_i (\mathbf{R}_i' \mathbf{R}_j) \hat{\mathbf{A}}_j$$

In this example, there are four marginal proportional hazards models, one for each potential recurrence time. Instead of fitting one model at a time, you can fit all four marginal models in one analysis by using the STRATA statement and model-specific covariates as in the following statements. Using Visit as the STRATA variable on the input data set Bladder, PROC PHREG simultaneously fits all four marginal models, one for each Visit value. The COVS(AGGREGATE) option is specified to compute the robust sandwich variance estimate by summing up the score residuals for each distinct pattern of ID value. The TEST statement TREATMENT is used to perform the global test of no treatment effect for each tumor recurrence, the AVERAGE option is specified to estimate the parameter for the common treatment effect, and the E option displays the optimal weights for the common treatment effect.

```
title 'Wei-Lin-Weissfeld Model';
proc phreg data=Bladder covs(aggregate);
   model TStop*Status(0)=Trt1-Trt4 Number1-Number4 Size1-Size4;
   Trt1= Trt * (Visit=1);
   Trt2= Trt * (Visit=2);
   Trt3= Trt * (Visit=3);
   Trt4= Trt * (Visit=4);
   Number1= Number * (Visit=1);
   Number2= Number * (Visit=2);
   Number3= Number * (Visit=3);
   Number4= Number * (Visit=4);
   Size1= Size * (Visit=1);
   Size2= Size * (Visit=2);
   Size3= Size * (Visit=3);
   Size4= Size * (Visit=4);
   strata Visit;
   id ID;
   TREATMENT: test trt1,trt2,trt3,trt4/average e;
run;
```

Out of the 86 patients, 47 patients have only one tumor recurrence, 29 patients have two recurrences, 22 patients have three recurrences, and 14 patients have four recurrences (Output 92.10.5). Parameter estimates for the four marginal models are shown in Output 92.10.6. The 4 DF Wald test (Output 92.10.7) indicates a lack of evidence of a treatment effect in any of the four recurrences ($p = 0.4105$). The optimal weights for estimating the parameter of the common treatment effect are 0.67684, 0.25723, –0.07547, and 0.14140 for Trt1, Trt2, Trt3, and Trt4, respectively, which gives a parameter estimate of –0.5489 with a standard error estimate of 0.2853. A more sensitive test for a treatment effect is the 1 DF test based on this common parameter; however, there is still insufficient evidence for such effect at the 0.05 level ($p = 0.0543$).

Output 92.10.5 Summary of Bladder Tumor Recurrences in 86 Patients

Wei-Lin-Weissfeld Model

The PHREG Procedure

Summary of the Number of Event and Censored Values

Stratum	Visit	Total	Event	Censored	Percent Censored
1	1	86	47	39	45.35
2	2	86	29	57	66.28
3	3	86	22	64	74.42
4	4	86	14	72	83.72
Total		344	112	232	67.44

Output 92.10.6 Analysis of Marginal Cox Models

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	StdErr Ratio	Chi-Square	Pr > ChiSq	Hazard Ratio
Trt1	1	-0.51762	0.30750	0.974	2.8336	0.0923	0.596
Trt2	1	-0.61944	0.36391	0.926	2.8975	0.0887	0.538
Trt3	1	-0.69988	0.41516	0.903	2.8419	0.0918	0.497
Trt4	1	-0.65079	0.48971	0.848	1.7661	0.1839	0.522
Number1	1	0.23599	0.07208	0.947	10.7204	0.0011	1.266
Number2	1	0.13756	0.08690	0.946	2.5059	0.1134	1.147
Number3	1	0.16984	0.10356	0.984	2.6896	0.1010	1.185
Number4	1	0.32880	0.11382	0.909	8.3453	0.0039	1.389
Size1	1	0.06789	0.08529	0.842	0.6336	0.4260	1.070
Size2	1	-0.07612	0.11812	0.881	0.4153	0.5193	0.927
Size3	1	-0.21131	0.17198	0.943	1.5097	0.2192	0.810
Size4	1	-0.20317	0.19106	0.830	1.1308	0.2876	0.816

Output 92.10.7 Tests of Treatment Effects

Wei-Lin-Weissfeld Model

The PHREG Procedure

Linear Coefficients for Test TREATMENT

Parameter	Row 1	Row 2	Row 3	Row 4	Average Effect
Trt1	1	0	0	0	0.67684
Trt2	0	1	0	0	0.25723
Trt3	0	0	1	0	-0.07547
Trt4	0	0	0	1	0.14140
Number1	0	0	0	0	0.00000
Number2	0	0	0	0	0.00000
Number3	0	0	0	0	0.00000
Number4	0	0	0	0	0.00000
Size1	0	0	0	0	0.00000
Size2	0	0	0	0	0.00000
Size3	0	0	0	0	0.00000
Size4	0	0	0	0	0.00000
CONSTANT	0	0	0	0	0.00000

Test TREATMENT Results

Wald Chi-Square	DF	Pr > ChiSq
3.9668	4	0.4105

Average Effect for Test TREATMENT

Estimate	Standard Error	z-Score	Pr > |z|
-0.5489	0.2853	-1.9240	0.0543

Example 92.11: Analysis of Clustered Data

(View the complete code for this example (phrex11.sas) in the example repository.)

When experimental units are naturally or artificially clustered, failure times of experimental units within a cluster are correlated. Two approaches can be taken to adjust for the intracluster correlation. In the marginal Cox model approach, Lee, Wei, and Amato (1992) estimate the regression parameters in the Cox model by the maximum partial likelihood estimates under an independent working assumption and use a robust sandwich covariance matrix estimate to account for the intracluster dependence. Lin (1994) illustrates this methodology by using a subset of data from the Diabetic Retinopathy Study (DRS). An alternative approach to account for the within-cluster correlation is to use a shared frailty model where cluster effects are incorporated into the model as independent and identically distributed random variables.

The following DATA step creates the data set Blind that represents 197 diabetic patients who have a high risk of experiencing blindness in both eyes as defined by DRS criteria. One eye of each patient is treated with laser photocoagulation. The hypothesis of interest is whether the laser treatment delays the occurrence

of blindness. Since juvenile and adult diabetes have very different courses, it is also desirable to examine how the age of onset of diabetes might affect the time of blindness. Since there are no biological differences between the left eye and the right eye, it is natural to assume a common baseline hazard function for the failure times of the left and right eyes.

Each patient is a cluster that contributes two observations to the input data set, one for each eye. The following variables are in the input data set Blind:

- ID, patient's identification
- Time, time to blindness
- Status, blindness indicator (0:censored and 1:blind)
- Treat, treatment received (Laser or Others)
- Type, type of diabetes (Juvenile: onset at age ≤ 20 or Adult: onset at age > 20)

```
proc format;
   value type 0='Juvenile' 1='Adult';
   value Rx  1='Laser' 0='Others';
run;

data Blind;
   input ID Time Status dty trt @@;
   Type= put(dty, type.);
   Treat= put(trt, Rx.);
   datalines;
   5 46.23 0 1 1    5 46.23 0 1 0   14 42.50 0 0 1   14 31.30 1 0 0
  16 42.27 0 0 1   16 42.27 0 0 0   25 20.60 0 0 1   25 20.60 0 0 0
  29 38.77 0 0 1   29  0.30 1 0 0   46 65.23 0 0 1   46 54.27 1 0 0

   ... more lines ...

1705  8.00 0 0 1 1705  8.00 0 0 0 1717 51.60 0 1 1 1717 42.33 1 1 0
1727 49.97 0 1 1 1727  2.90 1 1 0 1746 45.90 0 0 1 1746  1.43 1 0 0
1749 41.93 0 1 1 1749 41.93 0 1 0
;
```

As a preliminary analysis, PROC FREQ is used to summarize the number of eyes that developed blindness.

```
proc freq data=Blind;
   table Treat*Status;
run;
```

By the end of the study, 54 eyes treated with laser photocoagulation and 101 eyes treated by other means have developed blindness (Output 92.11.1).

Output 92.11.1 Distribution of Blindness

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Treat by Status		
		Status	
Treat	0	1	Total
Laser	143 36.29 72.59 59.83	54 13.71 27.41 34.84	197 50.00
Others	96 24.37 48.73 40.17	101 25.63 51.27 65.16	197 50.00
Total	239 60.66	155 39.34	394 100.00

The following statements use PROC PHREG to carry out the analysis of Lee, Wei, and Amato (1992). The explanatory variables in this Cox model are Treat, Type, and the Treat × Type interaction. The COVS(AGGREGATE) option is specified to compute the robust sandwich covariance matrix estimate. The ID statement identifies the variable that represents the clusters. The HAZARDRATIO statement requests hazard ratios for the treatments be displayed.

```
proc phreg data=Blind covs(aggregate);
   class Treat Type;
   model Time*Status(0)=Treat|Type;
   id ID;
   hazardratio 'Marginal Model Analysis' Treat;
run;
```

Results of the marginal model analysis are displayed in Output 92.11.2. The robust standard error estimates are smaller than the model-based counterparts, since the ratio of the robust standard error estimate relative to the model-based estimate is less than 1 for each parameter. Laser photocoagulation appears to be effective ($p = 0.0217$) in delaying the occurrence of blindness, although there is also a significant interaction effect between treatment and type of diabetes ($p = 0.0053$).

Output 92.11.2 Inference Based on the Marginal Model

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter		DF	Parameter Estimate	Standard Error	StdErr Ratio	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Treat	Laser	1	-0.42467	0.18497	0.850	5.2713	0.0217	.	Treat Laser
Type	Adult	1	0.34084	0.19558	0.982	3.0371	0.0814	.	Type Adult
Treat*Type	Laser Adult	1	-0.84566	0.30353	0.865	7.7622	0.0053	.	Treat Laser * Type Adult

Hazard ratio estimates of the laser treatment relative to nonlaser treatment are displayed in Output 92.11.3. For both types of diabetes, the 95% confidence interval for the hazard ratio lies below 1. This indicates that laser-photocoagulation treatment is more effective in delaying blindness regardless of the type of diabetes. However, the effect is more prominent for adult-onset diabetes than for juvenile-onset diabetes since the hazard ratio estimates for the former are less than those of the latter.

Output 92.11.3 Hazard Ratio Estimates for Marginal Model

Marginal Model Analysis: Hazard Ratios for Treat			
Description	Point Estimate	95% Wald Robust Confidence Limits	
Treat Laser vs Others At Type=Adult	0.281	0.175	0.451
Treat Laser vs Others At Type=Juvenile	0.654	0.455	0.940

Next, you analyze the same data by using a shared frailty model. The following statements use PROC PHREG to fit a shared frailty model to the Blind data set. The RANDOM statement identifies the variable ID as the variable that represents the clusters. You must declare the cluster variable as a classification variable in the CLASS statement.

```
proc phreg data=Blind;
   class ID Treat Type;
   model Time*Status(0)=Treat|Type;
   random ID;
   hazardratio 'Frailty Model Analysis' Treat;
run;
```

Selected results of this analysis are displayed in Output 92.11.4 to Output 92.11.6.

The "Random Class Level Information" table in Output 92.11.4 displays the 197 ID values of the patients. You can suppress the display of this table by using the NOCLPRINT option in the RANDOM statement.

Output 92.11.4 Unique Cluster Identification Values

The PHREG Procedure

Class Level Information for Random Effects		
Class	Levels	Values
ID	197	5 14 16 25 29 46 49 56 61 71 100 112 120 127 133 150 167 176 185 190 202 214 220 243 255 264 266 284 295 300 302 315 324 328 335 342 349 357 368 385 396 405 409 419 429 433 445 454 468 480 485 491 503 515 522 538 547 550 554 557 561 568 572 576 581 606 610 615 618 624 631 636 645 653 662 664 683 687 701 706 717 722 731 740 749 757 760 766 769 772 778 780 793 800 804 810 815 832 834 838 857 866 887 903 910 920 925 931 936 945 949 952 962 964 971 978 983 987 1002 1017 1029 1034 1037 1042 1069 1074 1098 1102 1112 1117 1126 1135 1145 1148 1167 1184 1191 1205 1213 1228 1247 1250 1253 1267 1281 1287 1293 1296 1309 1312 1317 1321 1333 1347 1361 1366 1373 1397 1410 1413 1425 1447 1461 1469 1480 1487 1491 1499 1503 1513 1524 1533 1537 1552 1554 1562 1572 1581 1585 1596 1600 1603 1619 1627 1636 1640 1643 1649 1666 1672 1683 1688 1705 1717 1727 1746 1749

The "Covariance Parameter Estimates" table in Output 92.11.5 displays the estimate and asymptotic estimated standard error of the common variance parameter of the normal random effects.

Output 92.11.5 Variance Estimate of the Normal Random Effects

Covariance Parameter Estimates		
Cov Parm	REML Estimate	Standard Error
ID	0.8308	0.2145

Output 92.11.6 displays the Wald tests for both the fixed effects and the random effects. The random effects are statistically significant ($p = 0.0042$). Results of testing the fixed effects are very similar to those based on the robust variance estimates. Laser photocoagulation appears to be effective ($p = 0.0252$) in delaying the occurrence of blindness, although there is also a significant treatment by diabetes type interaction effect ($p = 0.0071$).

Output 92.11.6 Inference Based on the Frailty Model

Type 3 Tests

Effect	Wald Chi-Square	DF	Pr > ChiSq	Adjusted DF	Adjusted Pr > ChiSq
Treat	4.8961	1	0.0269	0.9587	0.0252
Type	2.6395	1	0.1042	0.6795	0.0629
Treat*Type	7.1349	1	0.0076	0.9644	0.0071
ID	110.3922	.	.	74.2788	0.0042

Analysis of Maximum Likelihood Estimates

Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Treat	Laser	1	-0.49848	0.22528	4.8961	0.0269	.	Treat Laser
Type	Adult	1	0.39788	0.24490	2.6395	0.1042	.	Type Adult
Treat*Type	Laser Adult	1	-0.96540	0.36142	7.1349	0.0076	.	Treat Laser * Type Adult

Estimates of hazard ratios of the laser treatment relative to nonlaser treatment are displayed in Output 92.11.7. These estimates closely resemble those computed in analysis based on the marginal Cox model in Output 92.11.3, which leads to the same conclusion that laser photocoagulation is effective in delaying blindness for both types of diabetes, and more effective for the adult-onset diabetes than for juvenile-onset diabetes.

Output 92.11.7 Hazard Ratio Estimates for Frailty Model

Frailty Model Analysis: Hazard Ratios for Treat

Description	Point Estimate	95% Wald Confidence Limits	
Treat Laser vs Others At Type=Adult	0.231	0.133	0.403
Treat Laser vs Others At Type=Juvenile	0.607	0.391	0.945

Example 92.12: Model Assessment Using Cumulative Sums of Martingale Residuals

(View the complete code for this example (phrex12.sas) in the example repository.)

The Mayo liver disease example of Lin, Wei, and Ying (1993) is reproduced here to illustrate the checking of the functional form of a covariate and the assessment of the proportional hazards assumption. The data represent 418 patients with primary biliary cirrhosis (PBC), among whom 161 had died as of the date of data listing. A subset of the variables is saved in the SAS data set Liver. The data set contains the following variables:

- Time, follow-up time, in years
- Status, event indicator with value 1 for death time and value 0 for censored time
- Age, age in years from birth to study registration
- Albumin, serum albumin level, in g/dl
- Bilirubin, serum bilirubin level, in mg/dl
- Edema, edema presence
- Protime, prothrombin time, in seconds

The following statements create the data set Liver:

```
data Liver;
   input Time Status Age Albumin Bilirubin Edema Protime @@;
   label Time="Follow-up Time in Years";
   Time= Time / 365.25;
   datalines;
  400 1 58.7652 2.60 14.5 1.0 12.2 4500 0 56.4463 4.14  1.1 0.0 10.6
 1012 1 70.0726 3.48  1.4 0.5 12.0 1925 1 54.7406 2.54  1.8 0.5 10.3
 1504 0 38.1054 3.53  3.4 0.0 10.9 2503 1 66.2587 3.98  0.8 0.0 11.0
 1832 0 55.5346 4.09  1.0 0.0  9.7 2466 1 53.0568 4.00  0.3 0.0 11.0
 2400 1 42.5079 3.08  3.2 0.0 11.0   51 1 70.5599 2.74 12.6 1.0 11.5
 3762 1 53.7139 4.16  1.4 0.0 12.0  304 1 59.1376 3.52  3.6 0.0 13.6
 3577 0 45.6893 3.85  0.7 0.0 10.6 1217 1 56.2218 2.27  0.8 1.0 11.0
 3584 1 64.6461 3.87  0.8 0.0 11.0 3672 0 40.4435 3.66  0.7 0.0 10.8

   ... more lines ...

  989 0 35.0000 3.23  0.7 0.0 10.8  681 1 67.0000 2.96  1.2 0.0 10.9
 1103 0 39.0000 3.83  0.9 0.0 11.2 1055 0 57.0000 3.42  1.6 0.0  9.9
  691 0 58.0000 3.75  0.8 0.0 10.4  976 0 53.0000 3.29  0.7 0.0 10.6
;
```

Consider fitting a Cox model for the survival time of the PCB patients with the covariates Bilirubin, log(Protime), log(Albumin), Age, and Edema. The log transform, which is often applied to blood chemistry measurements, is deliberately not employed for Bilirubin. It is of interest to assess the functional form of the variable Bilirubin in the Cox model. The specifications are as follows:

```
ods graphics on;
proc phreg data=Liver;
   model Time*Status(0)=Bilirubin logProtime logAlbumin Age Edema;
   logProtime=log(Protime);
   logAlbumin=log(Albumin);
   assess var=(Bilirubin) /  resample seed=7548;
run;
```

The ASSESS statement creates a plot of the cumulative martingale residuals against the values of the covariate Bilirubin, which is specified in the VAR= option. The RESAMPLE option computes the *p*-value of a Kolmogorov-type supremum test based on a sample of 1,000 simulated residual patterns.

Parameter estimates of the model fit are shown in Output 92.12.1. The plot in Output 92.12.2 displays the observed cumulative martingale residual process for Bilirubin together with 20 simulated realizations from the null distribution. When ODS Graphics is enabled, this graphical display is requested by specifying the ASSESS statement. It is obvious that the observed process is atypical compared to the simulated realizations. Also, none of the 1,000 simulated realizations has an absolute maximum exceeding that of the observed cumulative martingale residual process. Both the graphical and numerical results indicate that a transform is deemed necessary for Bilirubin in the model.

Output 92.12.1 Cox Model with Bilirubin as a Covariate

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
Bilirubin	1	0.11733	0.01298	81.7567	<.0001	1.124
logProtime	1	2.77581	0.71482	15.0794	0.0001	16.052
logAlbumin	1	-3.17195	0.62945	25.3939	<.0001	0.042
Age	1	0.03779	0.00805	22.0288	<.0001	1.039
Edema	1	0.84772	0.28125	9.0850	0.0026	2.334

Output 92.12.2 Cumulative Martingale Residuals versus Bilirubin

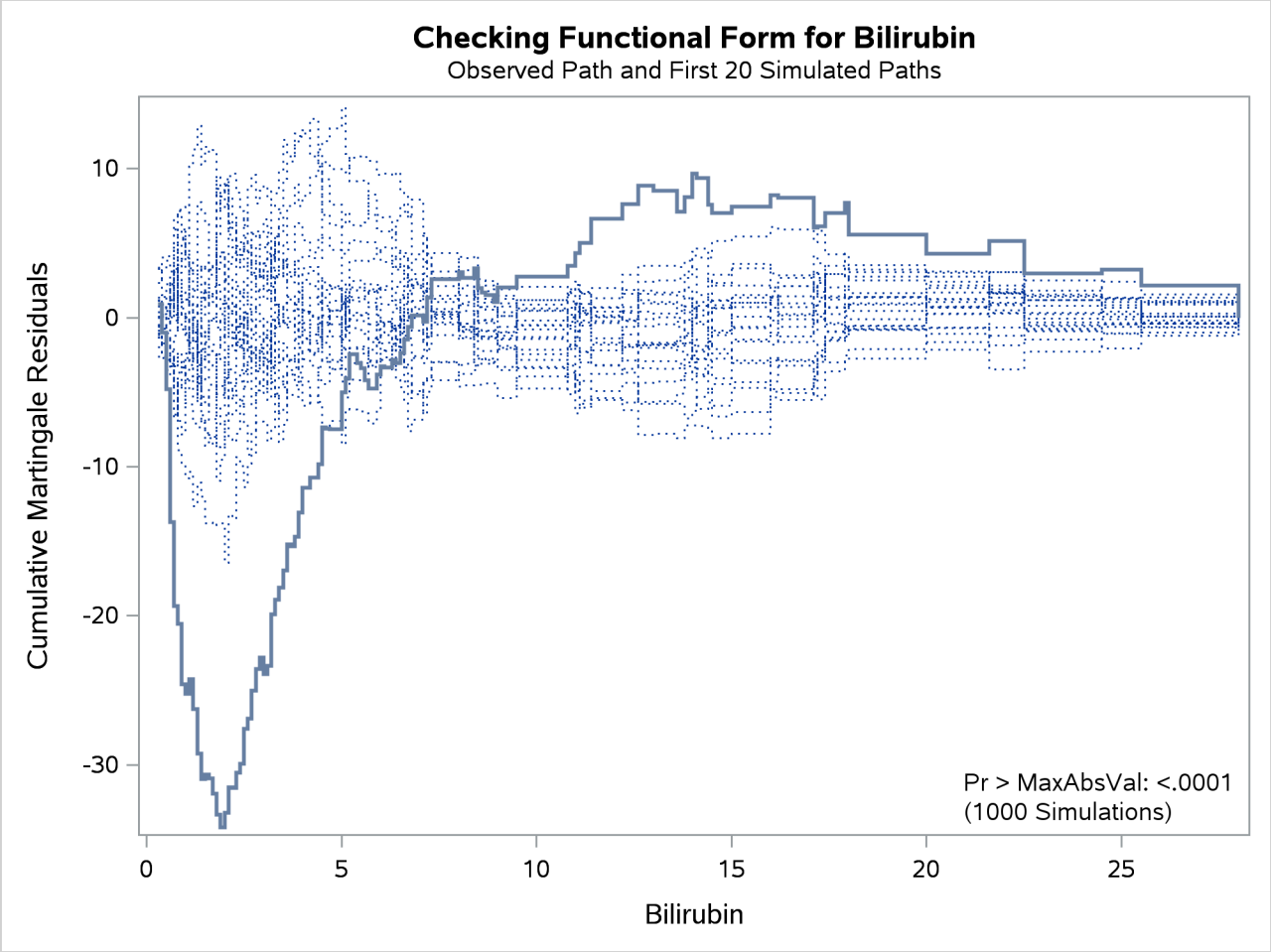

The cumulative martingale residual plots in Output 92.12.3 provide guidance in suggesting a more appropriate functional form for a covariate. The four curves were created from simple forms of misspecification by using 10,000 simulated times from a exponential model with 20% censoring. The true and fitted models are shown in Table 92.21. The following statements produce Output 92.12.3.

```
data sim(drop=tmp);
   p = 1 / 91;
   seed = 1;
   do n = 1 to 10000;
      x1 = rantbl( seed, p, p, p, p, p, p, p, p, p, p,
                            p, p, p, p, p, p, p, p, p, p,
                            p, p, p, p, p, p, p, p, p, p,
                            p, p, p, p, p, p, p, p, p, p,
                            p, p, p, p, p, p, p, p, p, p,
                            p, p, p, p, p, p, p, p, p, p,
                            p, p, p, p, p, p, p, p, p, p,
                            p, p, p, p, p, p, p, p, p, p,
                            p, p, p, p, p, p, p, p, p, p );

      x1 = 1 + ( x1 - 1 ) / 10;
```

```
      x2= x1 * x1;
      x3= x1 * x2;
      status= rantbl(seed, .8);
      tmp= log(1-ranuni(seed));
      t1= -exp(-log(x1)) * tmp;
      t2= -exp(-.1*(x1+x2)) * tmp;
      t3= -exp(-.01*(x1+x2+x3)) * tmp;
      tt= -exp(-(x1>5)) * tmp;
      output;
   end;
run;


proc sort data=sim;
   by x1;
run;

proc phreg data=sim noprint;
   model t1*status(2)=x1;
   output out=out1 resmart=resmart;
run;

proc phreg data=sim noprint;
   model t2*status(2)=x1;
   output out=out2 resmart=resmart;
run;


proc phreg data=sim noprint;
   model t3*status(2)=x1 x2;
   output out=out3 resmart=resmart;
run;

proc phreg data=sim noprint;
   model tt*status(2)=x1;
   output out=out4 resmart=resmart;
run;

data out1(keep=x1 cresid1);
   retain cresid1 0;
   set out1;
   by x1;
   cresid1 + resmart;
   if last.x1  then output;
run;

data out2(keep=x1 cresid2);
   retain cresid2 0;
   set out2;
   by x1;
   cresid2 + resmart;
   if last.x1  then output;
run;
```

```
data out3(keep=x1 cresid3);
   retain cresid3 0;
   set out3;
   by x1;
   cresid3 + resmart;
   if last.x1  then output;
run;

data out4(keep=x1 cresid4);
   retain cresid4 0;
   set out4;
   by x1;
   cresid4 + resmart;
   if last.x1  then output;
run;

data all;
   set out1;
   set out2;
   set out3;
   set out4;
run;

proc template;
   define statgraph MisSpecification;
      BeginGraph;
         entrytitle "Covariate Misspecification";
         layout lattice / columns=2 rows=2 columndatarange=unionall;

            columnaxes;
               columnaxis / display=(ticks tickvalues label) label="x";
               columnaxis / display=(ticks tickvalues label) label="x";
            endcolumnaxes;

            cell;
               cellheader;
                  entry "(a) Data: log(X), Model: X";
               endcellheader;
               layout overlay / xaxisopts=(display=none)
                                yaxisopts=(label="Cumulative Residual");
                  seriesplot y=cresid1 x=x1 / lineattrs=GraphFit;
               endlayout;
            endcell;

            cell;
               cellheader;
                  entry "(b) Data: X*X, Model: X";
               endcellheader;
               layout overlay / xaxisopts=(display=none)
                                yaxisopts=(label=" ");
                  seriesplot y=cresid2 x=x1 / lineattrs=GraphFit;
               endlayout;
            endcell;
```

```
            cell;
               cellheader;
                  entry "(c) Data: X*X*X, Model: X*X";
               endcellheader;
               layout overlay / xaxisopts=(display=none)
                                yaxisopts=(label="Cumulative Residual");
                  seriesplot y=cresid3 x=x1 / lineattrs=GraphFit;
               endlayout;
            endcell;

            cell;
               cellheader;
                  entry "(d) Data: I(X>5), Model: X";
               endcellheader;
               layout overlay / xaxisopts=(display=none)
                                yaxisopts=(label=" ");
                  seriesplot y=cresid4 x=x1 / lineattrs=GraphFit;
               endlayout;
            endcell;

         endlayout;
      EndGraph;
   end;
run;

proc sgrender data=all template=MisSpecification;
run;
```

Output 92.12.3 Typical Cumulative Residual Plot Patterns

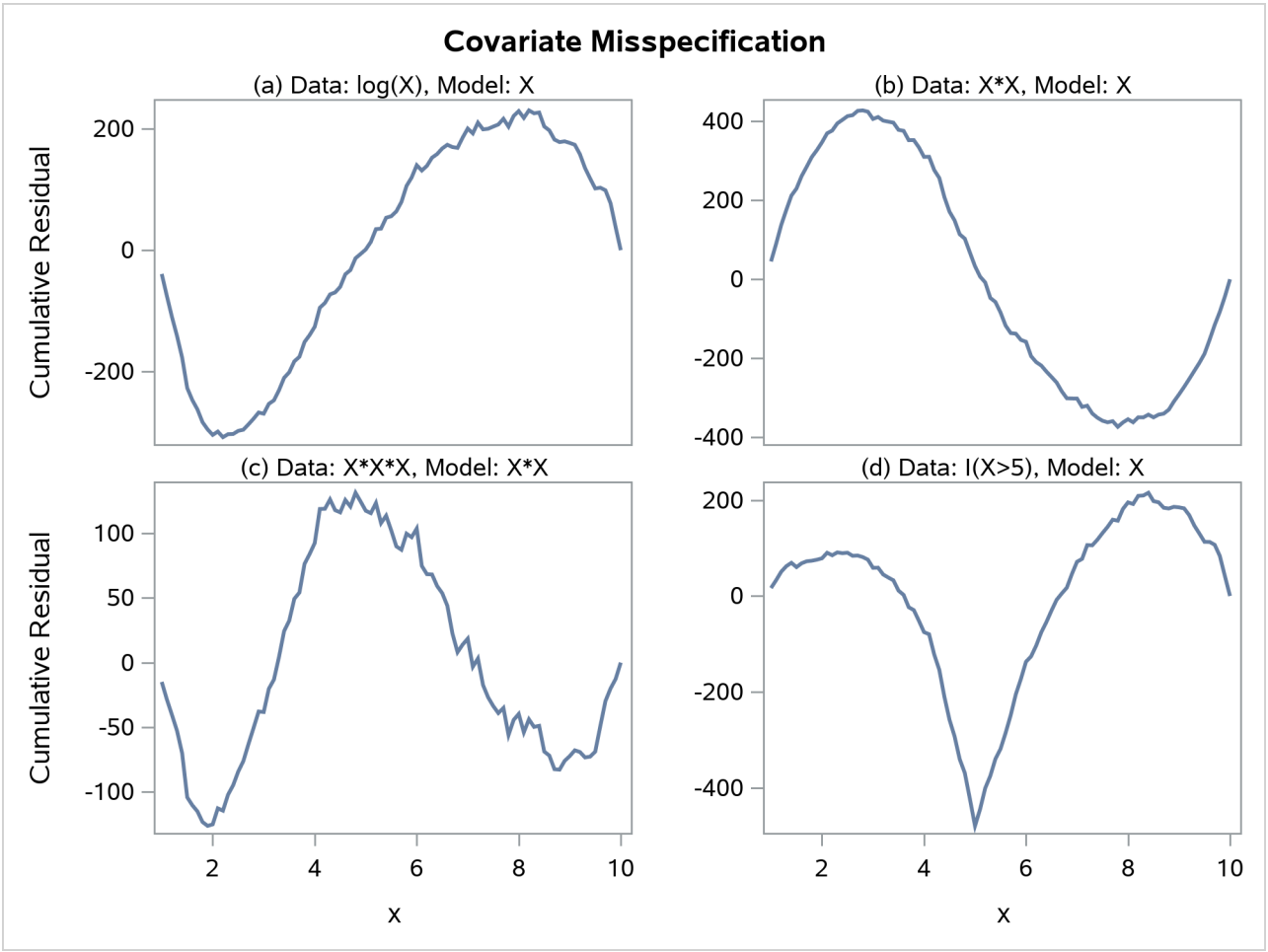

Table 92.21 Model Misspecifications

Plot	Data	Fitted Model
(a)	$\log(X)$	X
(b)	$\{X, X^2\}$	X
(c)	$\{X, X^2, X^3\}$	$\{X, X^2\}$
(d)	$I(X > 5)$	X

The curve of observed cumulative martingale residuals in Output 92.12.2 most resembles the behavior of the curve in plot (a) of Output 92.12.3, indicating that log(Bilirubin) might be a more appropriate term in the model than Bilirubin.

Next, the analysis of the natural history of the PBC is repeated with log(Bilirubin) replacing Bilirubin, and the functional form of log(Bilirubin) is assessed. Also assessed is the proportional hazards assumption for the Cox model. The analysis is carried out by the following statements:

```
proc phreg data=Liver;
   model Time*Status(0)=logBilirubin logProtime logAlbumin Age Edema;
   logBilirubin=log(Bilirubin);
   logProtime=log(Protime);
   logAlbumin=log(Albumin);
   assess var=(logBilirubin) ph / crpanel resample seed=19;
run;
```

The SEED= option specifies a integer seed for generating random numbers. The CRPANEL option in the ASSESS statement requests a panel of four plots. Each plot displays the observed cumulative martingale residual process along with two simulated realizations. The PH option checks the proportional hazards assumption of the model by plotting the observed standardized score process with 20 simulated realizations for each covariate in the model.

Output 92.12.4 displays the parameter estimates of the fitted model. The cumulative martingale residual plots in Output 92.12.5 and Output 92.12.6 show that the observed martingale residual process is more typical of the simulated realizations. The *p*-value for the Kolmogorov-type supremum test based on 1,000 simulations is 0.052, indicating that the log transform is a much improved functional form for Bilirubin.

Output 92.12.4 Model with log(Bilirubin) as a Covariate

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
logBilirubin	1	0.87072	0.08263	111.0484	<.0001	2.389
logProtime	1	2.37789	0.76674	9.6181	0.0019	10.782
logAlbumin	1	-2.53264	0.64819	15.2664	<.0001	0.079
Age	1	0.03940	0.00765	26.5306	<.0001	1.040
Edema	1	0.85934	0.27114	10.0447	0.0015	2.362

Output 92.12.5 Panel Plot of Cumulative Martingale Residuals versus log(Bilirubin)

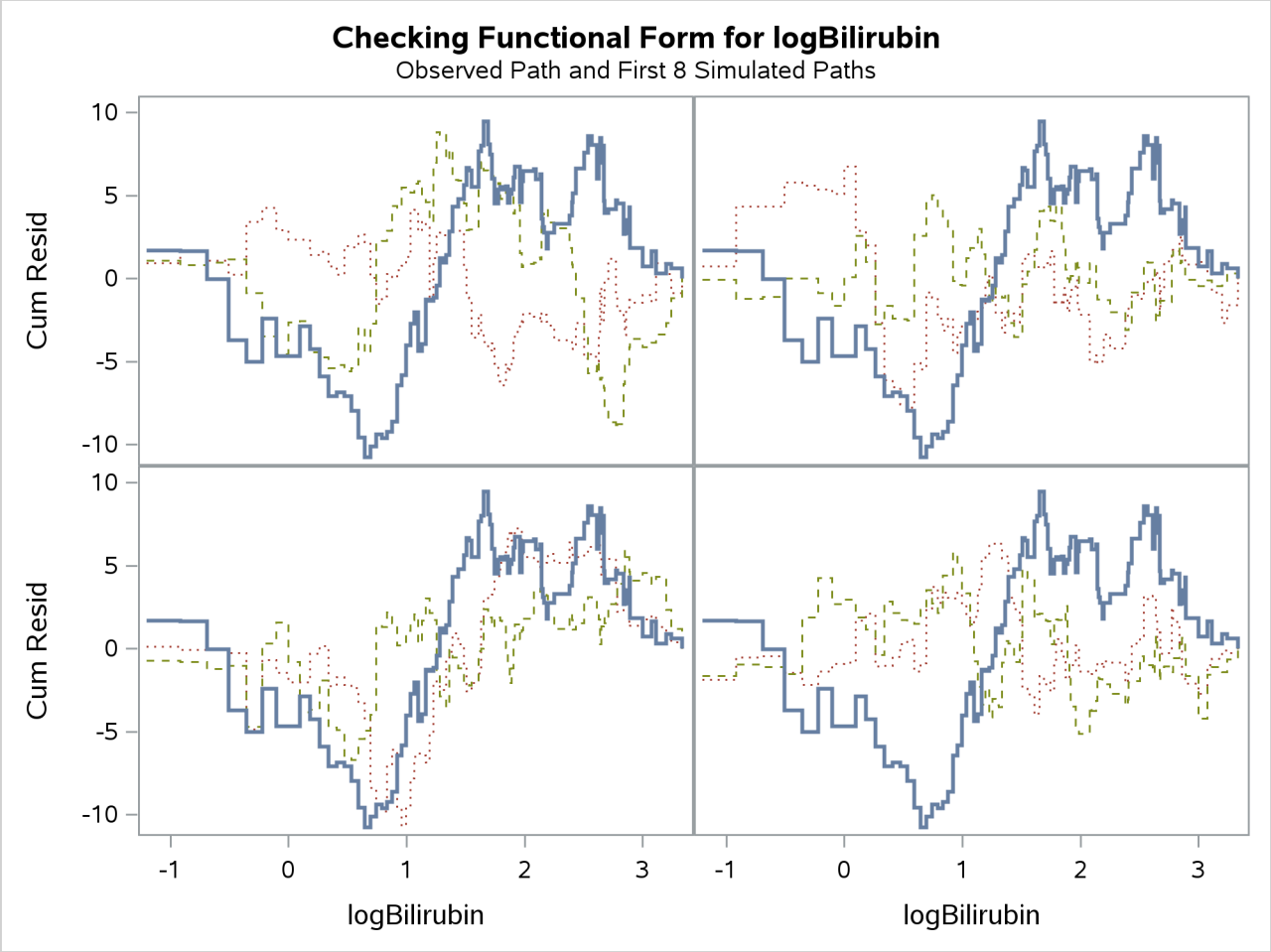

Output 92.12.6 Cumulative Martingale Residuals versus log(Bilirubin)

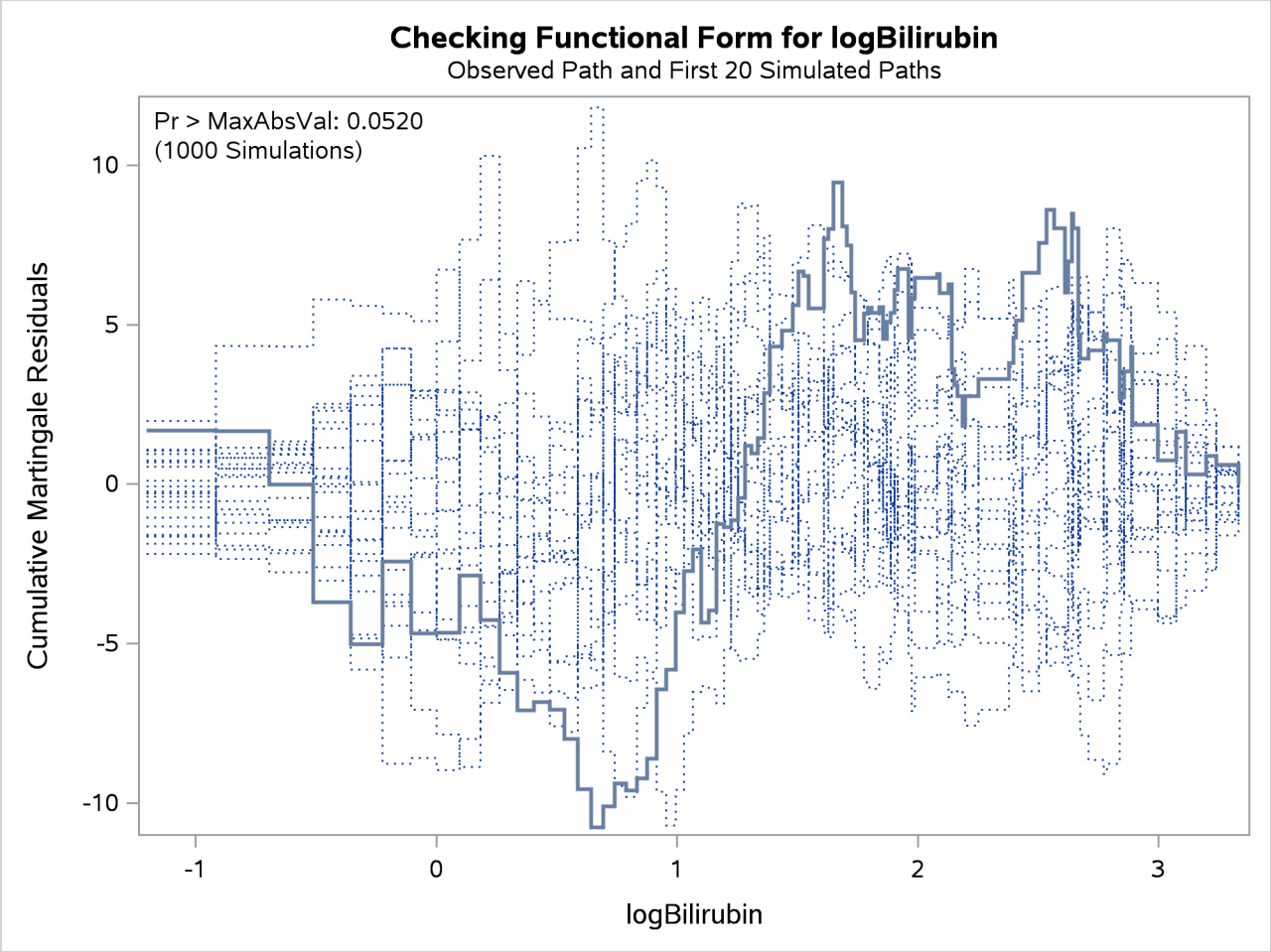

Output 92.12.7 and Output 92.12.8 display the results of proportional hazards assumption assessment for log(Bilirubin) and log(Protime), respectively. The latter plot reveals nonproportional hazards for log(Protime).

Output 92.12.7 Standardized Score Process for log(Bilirubin)

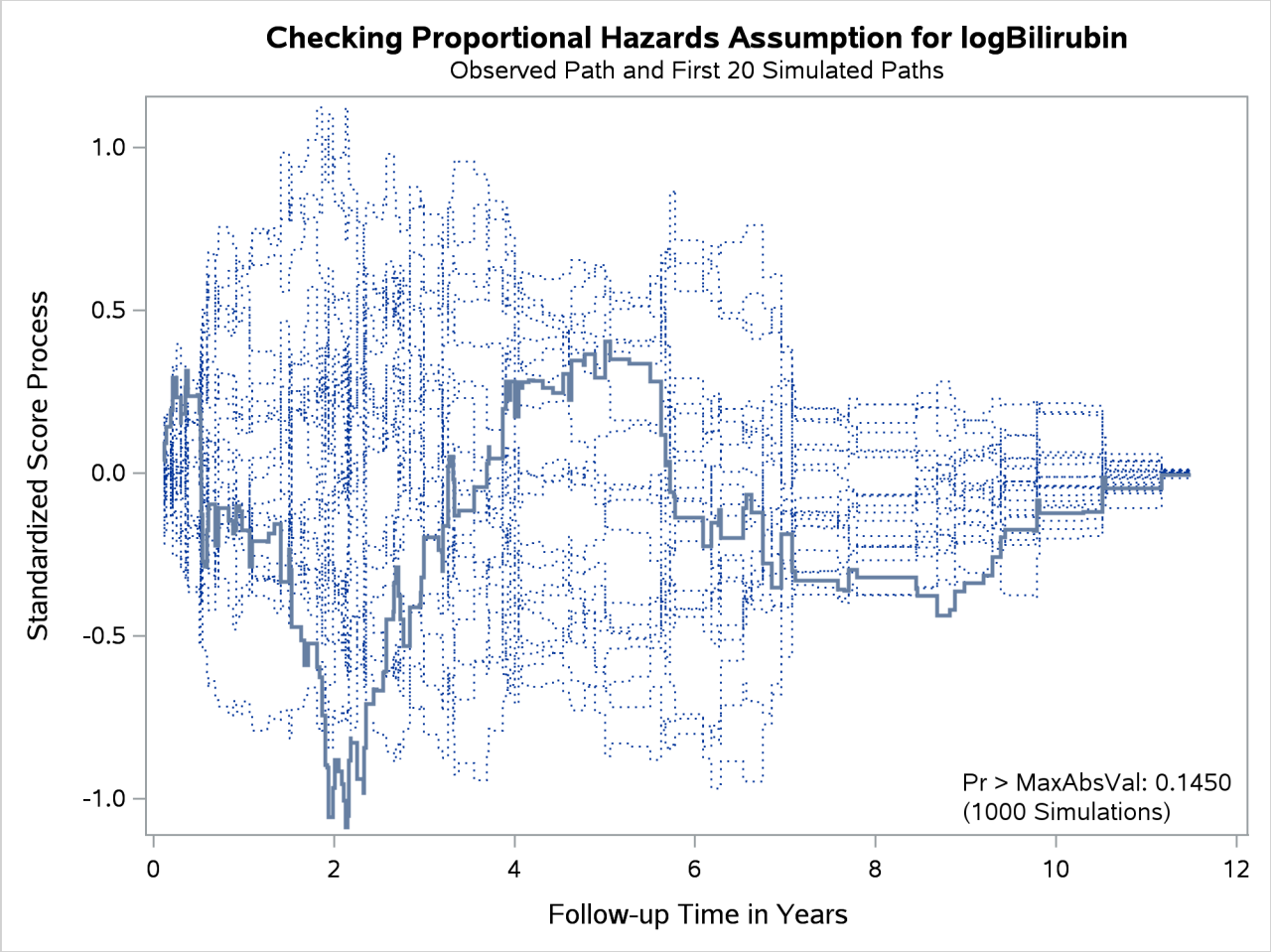

[

Output 92.12.8 Standardized Score Process for log(Protime)

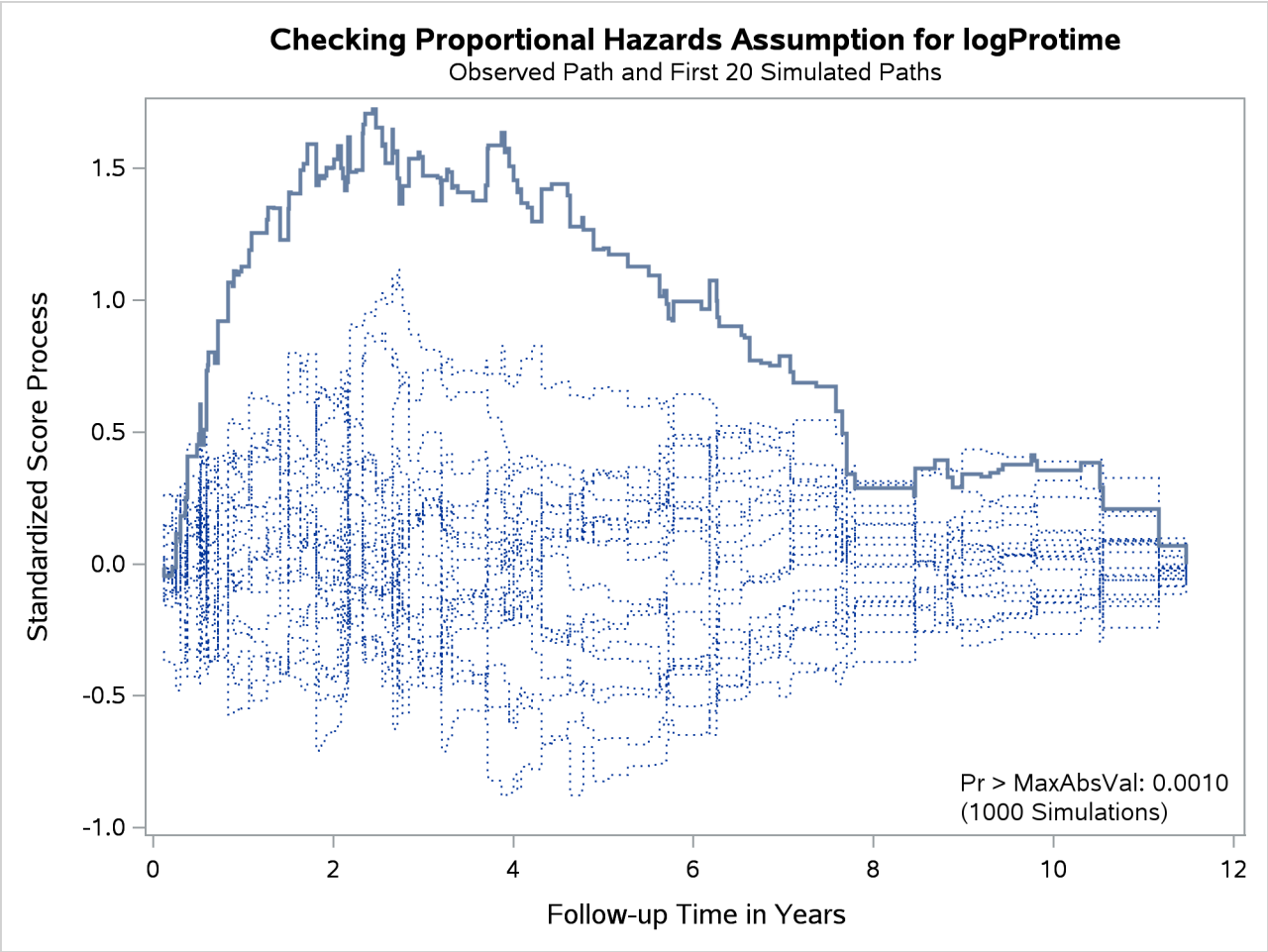

Plots for log(Albumin), Age, and Edema are not shown here. The Kolmogorov-type supremum test results for all the covariates are shown in Output 92.12.9. In addition to log(Protime), the proportional hazards assumption appears to be violated for Edema.

Output 92.12.9 Kolmogorov-Type Supremum Tests for Proportional Hazards Assumption

Supremum Test for Proportionals Hazards Assumption				
Variable	Maximum Absolute Value	Replications	Seed	Pr > MaxAbsVal
logBilirubin	1.0880	1000	19	0.1450
logProtime	1.7243	1000	19	0.0010
logAlbumin	0.8443	1000	19	0.4330
Age	0.7387	1000	19	0.4620
Edema	1.4350	1000	19	0.0330

Example 92.13: Bayesian Analysis of the Cox Model

(View the complete code for this example (phrex13.sas) in the example repository.)

This example illustrates the use of an informative prior. Hazard ratios, which are transformations of the regression parameters, are useful for interpreting survival models. This example also demonstrates the use of the HAZARDRATIO statement to obtain customized hazard ratios.

Consider the VALung data set in Example 92.3. In this example, the Cox model is used for the Bayesian analysis. The parameters are the coefficients of the continuous explanatory variables (Kps, Duration, and Age) and the coefficients of the design variables for the categorical explanatory variables (Prior, Cell, and Therapy). You use the CLASS statement in PROC PHREG to specify the categorical variables and their reference levels. Using the default reference parameterization, the design variables for the categorical variables are Prioryes (for Prior with Prior='no' as reference), Celladeno, Cellsmall, Cellsquamous (for Cell with Cell='large' as reference), and Therapytest (for Therapy='standard' as reference).

Consider the explanatory variable Kps. The Karnofsky performance scale index enables patients to be classified according to their functional impairment. The scale can range from 0 to 100—0 for dead, and 100 for a normal, healthy person with no evidence of disease. Recall that a flat prior was used for the regression coefficient in the example in the section "Bayesian Analysis" on page 7471. A flat prior on the Kps coefficient implies that the coefficient is as likely to be 0.1 as it is to be –100000. A coefficient of –5 means that a decrease of 20 points in the scale increases the hazard by $e^{-20\times-5}$(=2.68 $\times 10^{43}$)-fold, which is a rather unreasonable and unrealistic expectation for the effect of the Karnofsky index, much less than the value of –100000. Suppose you have a more realistic expectation: the effect is somewhat small and is more likely to be negative than positive, and a decrease of 20 points in the Karnofsky index will change the hazard from 0.9-fold (some minor positive effect) to 4-fold (a large negative effect). You can convert this opinion to a more informative prior on the Kps coefficient $\boldsymbol{\beta}_1$. Mathematically,

$$0.9 < e^{-20\boldsymbol{\beta}_1} < 4$$

which is equivalent to

$$-0.0693 < \boldsymbol{\beta}_1 < 0.0053$$

This becomes the plausible range that you believe the Kps coefficient can take. Now you can find a normal distribution that best approximates this belief by placing the majority of the prior distribution mass within this range. Assuming this interval is $\mu \pm 2\sigma$, where μ and σ are the mean and standard deviation of the normal prior, respectively, the hyperparameters μ and σ are computed as follows:

$$\begin{aligned}\mu &= \frac{-0.0693 + 0.0053}{2} = -0.032\\ \sigma &= \frac{0.0053 - (-0.0693)}{4} = 0.0186\end{aligned}$$

Note that a normal prior distribution with mean –0.0320 and standard deviation 0.0186 indicates that you believe, before looking at the data, that a decrease of 20 points in the Karnofsky index will probably change the hazard rate by 0.9-fold to 4-fold. This does not rule out the possibility that the Kps coefficient can take a more extreme value such as –5, but the probability of having such extreme values is very small.

Assume the prior distributions are independent for all the parameters. For the coefficient of Kps, you use a normal prior distribution with mean –0.0320 and variance 0.0186^2(=0.00035). For other parameters, you resort to using a normal prior distribution with mean 0 and variance 1E6, which is fairly noninformative. Means and variances of these independent normal distributions are saved in the data set Prior as follows:

```
proc format;
   value yesno 0='no' 10='yes';
run;

data VALung;
   drop check m;
   retain Therapy Cell;
   infile cards column=column;
   length Check $ 1;
   label Time='time to death in days'
         Kps='Karnofsky performance scale'
         Duration='months from diagnosis to randomization'
         Age='age in years'
         Prior='prior therapy'
         Cell='cell type'
         Therapy='type of treatment';
   format Prior yesno.;
   M=Column;
   input Check $ @@;
   if M>Column then M=1;
   if Check='s'|Check='t' then do;
      input @M Therapy $ Cell $;
      delete;
   end;
   else do;
      input @M Time Kps Duration Age Prior @@;
      Status=(Time>0);
      Time=abs(Time);
   end;
   datalines;
standard squamous
 72 60  7 69  0   411 70  5 64 10   228 60  3 38  0   126 60  9 63 10
118 70 11 65 10    10 20  5 49  0    82 40 10 69 10   110 80 29 68  0
314 50 18 43  0  -100 70  6 70  0    42 60  4 81  0     8 40 58 63 10
144 30  4 63  0   -25 80  9 52 10    11 70 11 48 10
standard small
 30 60  3 61  0   384 60  9 42  0     4 40  2 35  0    54 80  4 63 10
 13 60  4 56  0  -123 40  3 55  0   -97 60  5 67  0   153 60 14 63 10
 59 30  2 65  0   117 80  3 46  0    16 30  4 53 10   151 50 12 69  0
 22 60  4 68  0    56 80 12 43 10    21 40  2 55 10    18 20 15 42  0
139 80  2 64  0    20 30  5 65  0    31 75  3 65  0    52 70  2 55  0
287 60 25 66 10    18 30  4 60  0    51 60  1 67  0   122 80 28 53  0
 27 60  8 62  0    54 70  1 67  0     7 50  7 72  0    63 50 11 48  0
392 40  4 68  0    10 40 23 67 10
standard adeno
  8 20 19 61 10    92 70 10 60  0    35 40  6 62  0   117 80  2 38  0
132 80  5 50  0    12 50  4 63 10   162 80  5 64  0     3 30  3 43  0
 95 80  4 34  0
```

```
standard large
177 50 16 66 10   162 80  5 62  0   216 50 15 52  0   553 70  2 47  0
278 60 12 63  0    12 40 12 68 10   260 80  5 45  0   200 80 12 41 10
156 70  2 66  0  -182 90  2 62  0   143 90  8 60  0   105 80 11 66  0
103 80  5 38  0   250 70  8 53 10   100 60 13 37 10
test squamous
999 90 12 54 10   112 80  6 60  0   -87 80  3 48  0  -231 50  8 52 10
242 50  1 70  0   991 70  7 50 10   111 70  3 62  0     1 20 21 65 10
587 60  3 58  0   389 90  2 62  0    33 30  6 64  0    25 20 36 63  0
357 70 13 58  0   467 90  2 64  0   201 80 28 52 10     1 50  7 35  0
 30 70 11 63  0    44 60 13 70 10   283 90  2 51  0    15 50 13 40 10
test small
 25 30  2 69  0  -103 70 22 36 10    21 20  4 71  0    13 30  2 62  0
 87 60  2 60  0     2 40 36 44 10    20 30  9 54 10     7 20 11 66  0
 24 60  8 49  0    99 70  3 72  0     8 80  2 68  0    99 85  4 62  0
 61 70  2 71  0    25 70  2 70  0    95 70  1 61  0    80 50 17 71  0
 51 30 87 59 10    29 40  8 67  0
test adeno
 24 40  2 60  0    18 40  5 69 10   -83 99  3 57  0    31 80  3 39  0
 51 60  5 62  0    90 60 22 50 10    52 60  3 43  0    73 60  3 70  0
  8 50  5 66  0    36 70  8 61  0    48 10  4 81  0     7 40  4 58  0
140 70  3 63  0   186 90  3 60  0    84 80  4 62 10    19 50 10 42  0
 45 40  3 69  0    80 40  4 63  0
test large
 52 60  4 45  0   164 70 15 68 10    19 30  4 39 10    53 60 12 66  0
 15 30  5 63  0    43 60 11 49 10   340 80 10 64 10   133 75  1 65  0
111 60  5 64  0   231 70 18 67 10   378 80  4 65  0    49 30  3 37  0
;

data Prior;
   input _TYPE_ $ Kps Duration Age Prioryes Celladeno Cellsmall
         Cellsquamous Therapytest;
   datalines;
   Mean -0.0320 0 0 0 0 0 0 0
   Var  0.00035 1e6 1e6 1e6 1e6 1e6 1e6 1e6
   run;
```

In the following BAYES statement, COEFFPRIOR=NORMAL(INPUT=PRIOR) specifies the normal prior distribution for the regression coefficients whose details are contained in the data set Prior. Posterior summaries (means, standard errors, and quantiles) and intervals (equal-tailed and HPD) are requested by the STATISTICS= option. Autocorrelations and effective sample sizes are requested by the DIAGNOSTICS= option as convergence diagnostics along with the trace plots (PLOTS= option) for visual analysis. For comparisons of hazards, three HAZARDRATIO statements are specified—one for the variable Therapy, one for the variable Age, and one for the variable Cell.

```
ods graphics on;
proc phreg data=VALung;
   class Prior(ref='no') Cell(ref='large') Therapy(ref='standard');
   model Time*Status(0) = Kps Duration Age Prior Cell Therapy;
   bayes seed=1 coeffprior=normal(input=Prior) statistics=(summary interval)
         diagnostics=(autocorr ess) plots=trace;
   hazardratio 'Hazard Ratio Statement 1' Therapy;
   hazardratio 'Hazard Ratio Statement 2' Age / unit=10;
   hazardratio 'Hazard Ratio Statement 3' Cell;
```

```
run;
```

This analysis generates a posterior chain of 10,000 iterations after 2,000 iterations of burn-in, as depicted in Output 92.13.1.

Output 92.13.1 Model Information

The PHREG Procedure

Bayesian Analysis

Model Information		
Data Set	WORK.VALUNG	
Dependent Variable	Time	time to death in days
Censoring Variable	Status	
Censoring Value(s)	0	
Model	Cox	
Ties Handling	BRESLOW	
Sampling Algorithm	ARMS	
Burn-In Size	2000	
MC Sample Size	10000	
Thinning	1	

Output 92.13.2 displays the names of the parameters and their corresponding effects and categories.

Output 92.13.2 Parameter Names

Regression Parameter Information				
Parameter	**Effect**	**Prior**	**Cell**	**Therapy**
Kps	**Kps**			
Duration	**Duration**			
Age	**Age**			
Prioryes	**Prior**	yes		
Celladeno	**Cell**		adeno	
Cellsmall	**Cell**		small	
Cellsquamous	**Cell**		squamous	
Therapytest	**Therapy**			test

PROC PHREG computes the maximum likelihood estimates of regression parameters (Output 92.13.3). These estimates are used as the starting values for the simulation of posterior samples.

Output 92.13.3 Parameter Estimates

Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	95% Confidence Limits	
Kps	1	-0.0326	0.00551	-0.0434	-0.0218
Duration	1	-0.00009	0.00913	-0.0180	0.0178
Age	1	-0.00855	0.00930	-0.0268	0.00969
Prioryes	1	0.0723	0.2321	-0.3826	0.5273
Celladeno	1	0.7887	0.3027	0.1955	1.3819
Cellsmall	1	0.4569	0.2663	-0.0650	0.9787
Cellsquamous	1	-0.3996	0.2827	-0.9536	0.1544
Therapytest	1	0.2899	0.2072	-0.1162	0.6961

Output 92.13.4 displays the independent normal prior for the analysis.

Output 92.13.4 Coefficient Prior

Independent Normal Prior for Regression Coefficients		
Parameter	Mean	Precision
Kps	-0.032	2857.143
Duration	0	1E-6
Age	0	1E-6
Prioryes	0	1E-6
Celladeno	0	1E-6
Cellsmall	0	1E-6
Cellsquamous	0	1E-6
Therapytest	0	1E-6

Fit statistics are displayed in Output 92.13.5. These statistics are useful for variable selection.

Output 92.13.5 Fit Statistics

Fit Statistics	
DIC (smaller is better)	966.260
pD (Effective Number of Parameters)	7.934

Summary statistics of the posterior samples are shown in Output 92.13.6 and Output 92.13.7. These results are quite comparable to the classical results based on maximizing the likelihood as shown in Output 92.13.3, since the prior distribution for the regression coefficients is relatively flat.

Output 92.13.6 Summary Statistics

The PHREG Procedure

Bayesian Analysis

Posterior Summaries						
				Percentiles		
Parameter	N	Mean	Standard Deviation	25%	50%	75%
Kps	10000	-0.0326	0.00523	-0.0362	-0.0326	-0.0291
Duration	10000	-0.00159	0.00954	-0.00756	-0.00093	0.00504
Age	10000	-0.00844	0.00928	-0.0147	-0.00839	-0.00220
Prioryes	10000	0.0742	0.2348	-0.0812	0.0737	0.2337
Celladeno	10000	0.7881	0.3065	0.5839	0.7876	0.9933
Cellsmall	10000	0.4639	0.2709	0.2817	0.4581	0.6417
Cellsquamous	10000	-0.4024	0.2862	-0.5927	-0.4025	-0.2106
Therapytest	10000	0.2892	0.2038	0.1528	0.2893	0.4240

Output 92.13.7 Interval Statistics

Posterior Intervals					
		Equal-Tail			
Parameter	Alpha	Interval		HPD Interval	
Kps	0.050	-0.0429	-0.0222	-0.0433	-0.0226
Duration	0.050	-0.0220	0.0156	-0.0210	0.0164
Age	0.050	-0.0263	0.00963	-0.0265	0.00941
Prioryes	0.050	-0.3936	0.5308	-0.3832	0.5384
Celladeno	0.050	0.1879	1.3920	0.1764	1.3755
Cellsmall	0.050	-0.0571	1.0167	-0.0888	0.9806
Cellsquamous	0.050	-0.9687	0.1635	-0.9641	0.1667
Therapytest	0.050	-0.1083	0.6930	-0.1284	0.6710

With autocorrelations retreating quickly to 0 (Output 92.13.8) and large effective sample sizes (Output 92.13.9), both diagnostics indicate a reasonably good mixing of the Markov chain. The trace plots in Output 92.13.10 also confirm the convergence of the Markov chain.

Output 92.13.8 Autocorrelation Diagnostics

The PHREG Procedure

Bayesian Analysis

Posterior Autocorrelations				
Parameter	**Lag 1**	**Lag 5**	**Lag 10**	**Lag 50**
Kps	0.1442	-0.0016	0.0096	-0.0013
Duration	0.2672	-0.0054	-0.0004	-0.0011
Age	0.1374	-0.0044	0.0129	0.0084
Prioryes	0.2507	-0.0271	-0.0012	0.0004
Celladeno	0.4160	0.0265	-0.0062	0.0190
Cellsmall	0.5055	0.0277	-0.0011	0.0271
Cellsquamous	0.3586	0.0252	-0.0044	0.0107
Therapytest	0.2063	0.0199	-0.0047	-0.0166

Output 92.13.9 Effective Sample Size Diagnostics

Effective Sample Sizes			
		Autocorrelation	
Parameter	**ESS**	**Time**	**Efficiency**
Kps	7046.7	1.4191	0.7047
Duration	5790.0	1.7271	0.5790
Age	7426.1	1.3466	0.7426
Prioryes	6102.2	1.6388	0.6102
Celladeno	3673.4	2.7223	0.3673
Cellsmall	3346.4	2.9883	0.3346
Cellsquamous	4052.8	2.4674	0.4053
Therapytest	6870.8	1.4554	0.6871

Output 92.13.10 Trace Plots

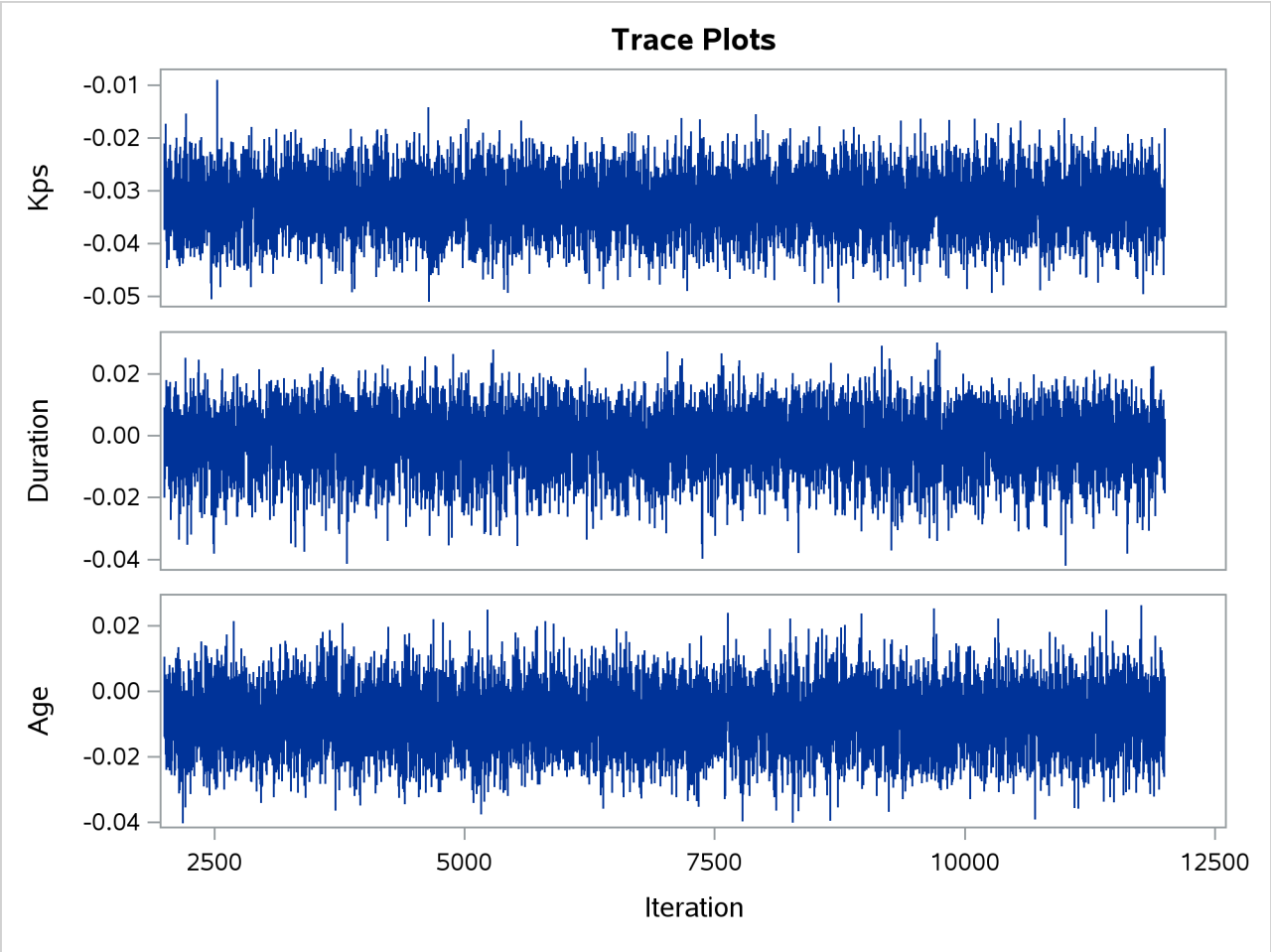

Output 92.13.10 *continued*

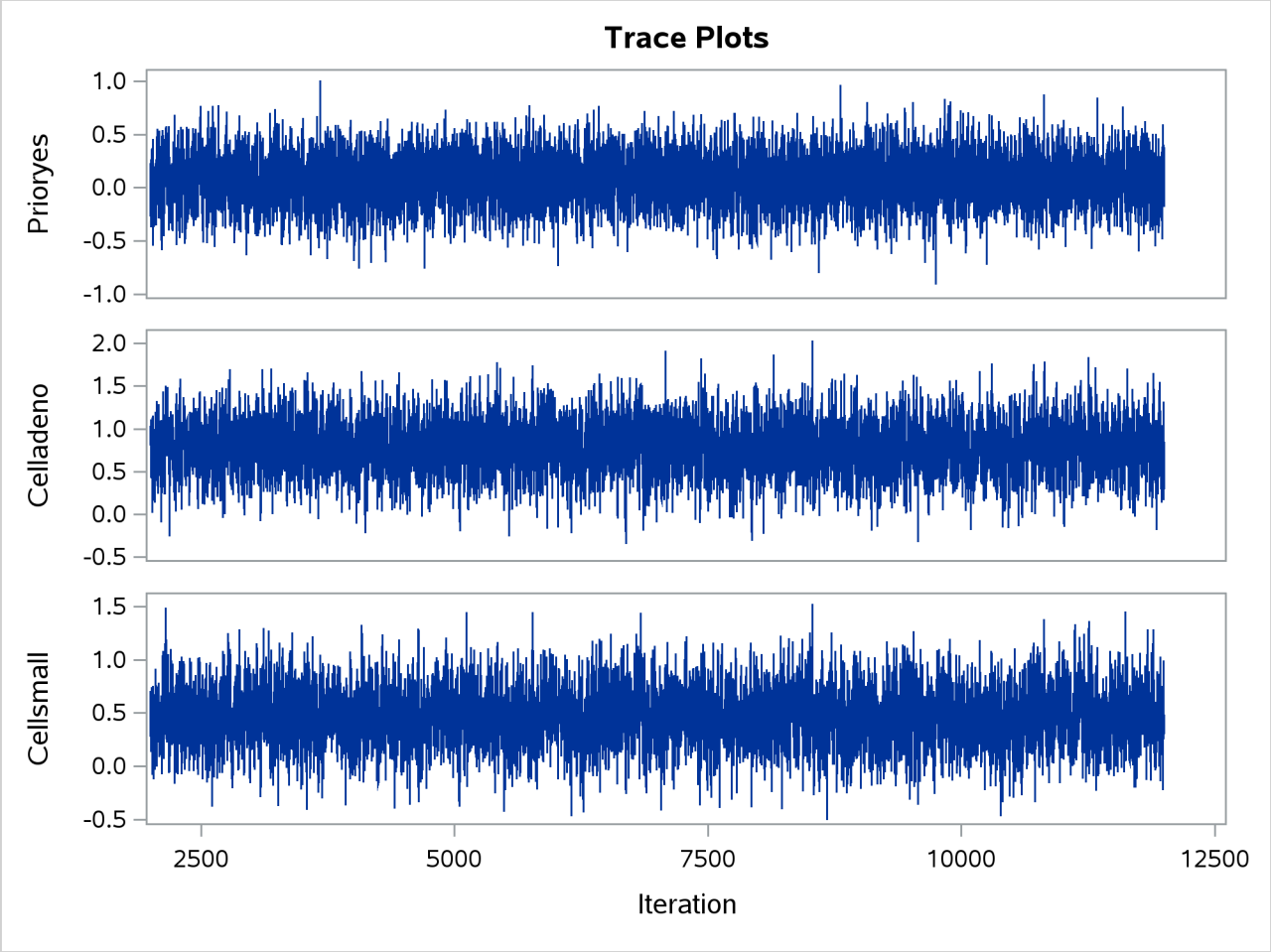

Output 92.13.10 *continued*

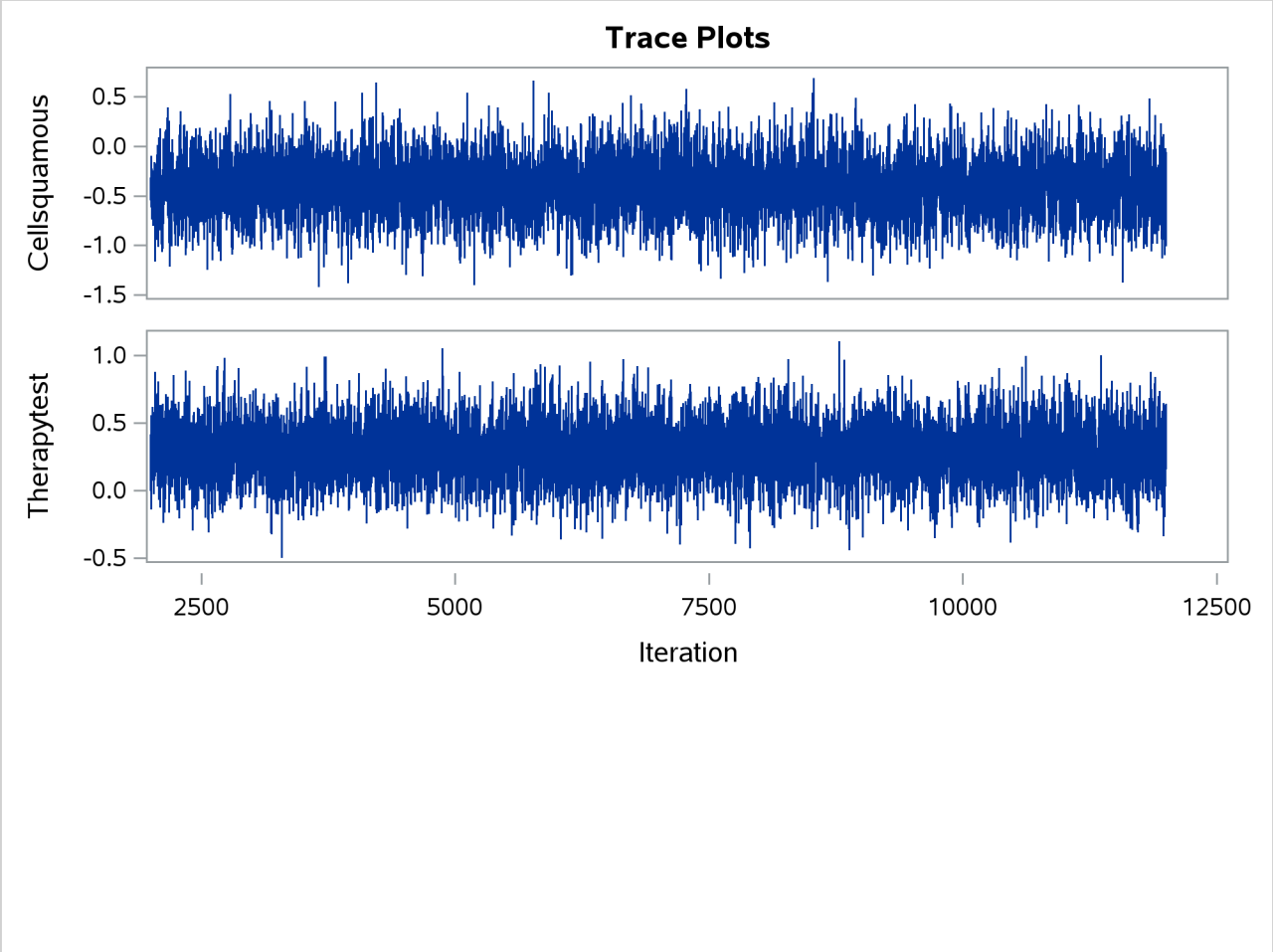

The first HAZARDRATIO statement compares the hazards between the standard therapy and the test therapy. Summaries of the posterior distribution of the corresponding hazard ratio are shown in Output 92.13.11. There is a 95% chance that the hazard ratio of standard therapy versus test therapy lies between 0.5 and 1.1.

Output 92.13.11 Hazard Ratio for Treatment

Hazard Ratio Statement 1: Hazard Ratios for type of treatment										
				Quantiles						
Description	N	Mean	Standard Deviation	25%	50%	75%	95% Equal-Tail Interval		95% HPD Interval	
Therapy standard vs test	10000	0.7645	0.1573	0.6544	0.7488	0.8583	0.5001	1.1143	0.4788	1.0805

The second HAZARDRATIO statement assesses the change of hazards for an increase in Age of 10 years. Summaries of the posterior distribution of the corresponding hazard ratio are shown in Output 92.13.12.

Output 92.13.12 Hazard Ratio for Age

Hazard Ratio Statement 2: Hazard Ratios for age in years										
				Quantiles						
Description	N	Mean	Standard Deviation	25%	50%	75%	95% Equal-Tail Interval		95% HPD Interval	
Age Unit=10	10000	0.9230	0.0859	0.8635	0.9195	0.9782	0.7685	1.1011	0.7650	1.0960

The third HAZARDRATIO statement compares the changes of hazards between two types of cells. For four types of cells, there are six different pairs of cell comparisons. The results are shown in Output 92.13.13.

Output 92.13.13 Hazard Ratios for Cell

Hazard Ratio Statement 3: Hazard Ratios for cell type										
				Quantiles						
Description	N	Mean	Standard Deviation	25%	50%	75%	95% Equal-Tail Interval		95% HPD Interval	
Cell adeno vs large	10000	2.3048	0.7224	1.7929	2.1982	2.7000	1.2067	4.0227	1.0053	3.7057
Cell adeno vs small	10000	1.4377	0.4078	1.1522	1.3841	1.6704	0.7930	2.3999	0.7309	2.2662
Cell adeno vs squamous	10000	3.4449	1.0745	2.6789	3.2941	4.0397	1.8067	5.9727	1.6303	5.5946
Cell large vs small	10000	0.6521	0.1780	0.5264	0.6325	0.7545	0.3618	1.0588	0.3331	1.0041
Cell large vs squamous	10000	1.5579	0.4548	1.2344	1.4955	1.8089	0.8492	2.6346	0.7542	2.4575
Cell small vs squamous	10000	2.4728	0.7081	1.9620	2.3663	2.8684	1.3789	4.1561	1.2787	3.9263

Example 92.14: Bayesian Analysis of Piecewise Exponential Model

(View the complete code for this example (phrex14.sas) in the example repository.)

This example illustrates using a piecewise exponential model in a Bayesian analysis. Consider the Rats data set in the section "Getting Started: PHREG Procedure" on page 7467. In the following statements, PROC PHREG is used to carry out a Bayesian analysis for the piecewise exponential model. In the BAYES statement, the option PIECEWISE stipulates a piecewise exponential model, and PIECEWISE=HAZARD requests that the constant hazards be modeled in the original scale. By default, eight intervals of constant hazards are used, and the intervals are chosen such that each has roughly the same number of events.

```
data Rats;
   label Days  ='Days from Exposure to Death';
   input Days Status Group @@;
   datalines;
143 1 0   164 1 0   188 1 0   188 1 0
190 1 0   192 1 0   206 1 0   209 1 0
213 1 0   216 1 0   220 1 0   227 1 0
230 1 0   234 1 0   246 1 0   265 1 0
304 1 0   216 0 0   244 0 0   142 1 1
156 1 1   163 1 1   198 1 1   205 1 1
232 1 1   232 1 1   233 1 1   233 1 1
233 1 1   233 1 1   239 1 1   240 1 1
261 1 1   280 1 1   280 1 1   296 1 1
```

```
296 1 1   323 1 1   204 0 1   344 0 1
;

proc phreg data=Rats;
   model Days*Status(0)=Group;
   bayes seed=1 piecewise=hazard statistics=(summary interval)
         diagnostics=(autocorr geweke ess);
run;
```

The "Model Information" table in Output 92.14.1 shows that the piecewise exponential model is being used.

Output 92.14.1 Model Information

The PHREG Procedure

Bayesian Analysis

Model Information		
Data Set	WORK.RATS	
Dependent Variable	Days	Days from Exposure to Death
Censoring Variable	Status	
Censoring Value(s)	0	
Model	Piecewise Exponential	
Sampling Algorithm	ARMS	
Burn-In Size	2000	
MC Sample Size	10000	
Thinning	1	

By default the time axis is partitioned into eight intervals of constant hazard. Output 92.14.2 details the number of events and observations in each interval. Note that the constant hazard parameters are named Lambda1,. . . , Lambda8. You can supply your own partition by using the INTERVALS= suboption within the PIECEWISE=HAZARD option.

Output 92.14.2 Interval Partition

Constant Hazard Time Intervals				
Interval				
[Lower,	Upper)	N	Event	Hazard Parameter
0	176	5	5	Lambda1
176	201.5	5	5	Lambda2
201.5	218	7	5	Lambda3
218	232.5	5	5	Lambda4
232.5	233.5	4	4	Lambda5
233.5	253.5	5	4	Lambda6
253.5	288	4	4	Lambda7
288	Infty	5	4	Lambda8

The model parameters consist of the eight hazard parameters Lambda1, . . . , Lambda8, and the regression coefficient Group. The maximum likelihood estimates are displayed in Output 92.14.3. Again, these estimates are used as the starting values for simulation of the posterior distribution.

Output 92.14.3 Maximum Likelihood Estimates

Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	95% Confidence Limits	
Lambda1	1	0.000953	0.000443	0.000084	0.00182
Lambda2	1	0.00794	0.00371	0.000672	0.0152
Lambda3	1	0.0156	0.00734	0.00120	0.0300
Lambda4	1	0.0236	0.0115	0.00112	0.0461
Lambda5	1	0.3669	0.1959	0	0.7509
Lambda6	1	0.0276	0.0148	0	0.0566
Lambda7	1	0.0262	0.0146	0	0.0548
Lambda8	1	0.0545	0.0310	0	0.1152
Group	1	-0.6223	0.3468	-1.3020	0.0573

Without using the PRIOR= suboption within the PIECEWISE=HAZARD option to specify the prior of the hazard parameters, the default is to use the noninformative and improper prior displayed in Output 92.14.4.

Output 92.14.4 Hazard Prior

Improper Prior for Hazards	
Parameter	Prior
Lambda1	1 / Lambda1
Lambda2	1 / Lambda2
Lambda3	1 / Lambda3
Lambda4	1 / Lambda4
Lambda5	1 / Lambda5
Lambda6	1 / Lambda6
Lambda7	1 / Lambda7
Lambda8	1 / Lambda8

The noninformative uniform prior is used for the regression coefficient Group (Output 92.14.5), as in the section "Bayesian Analysis" on page 7471.

Output 92.14.5 Coefficient Prior

Uniform Prior for Regression Coefficients	
Parameter	Prior
Group	Constant

Summary statistics for all model parameters are shown in Output 92.14.6 and Output 92.14.7.

Output 92.14.6 Summary Statistics

The PHREG Procedure

Bayesian Analysis

Posterior Summaries						
				Percentiles		
Parameter	N	Mean	Standard Deviation	25%	50%	75%
Lambda1	10000	0.000945	0.000444	0.000624	0.000876	0.00118
Lambda2	10000	0.00782	0.00363	0.00519	0.00724	0.00979
Lambda3	10000	0.0155	0.00735	0.0102	0.0144	0.0195
Lambda4	10000	0.0236	0.0116	0.0152	0.0217	0.0297
Lambda5	10000	0.3634	0.1965	0.2186	0.3266	0.4685
Lambda6	10000	0.0278	0.0153	0.0166	0.0249	0.0356
Lambda7	10000	0.0265	0.0151	0.0157	0.0236	0.0338
Lambda8	10000	0.0558	0.0323	0.0322	0.0488	0.0721
Group	10000	-0.6154	0.3570	-0.8569	-0.6186	-0.3788

Output 92.14.7 Interval Statistics

Posterior Intervals					
		Equal-Tail Interval		HPD Interval	
Parameter	Alpha				
Lambda1	0.050	0.000289	0.00199	0.000208	0.00182
Lambda2	0.050	0.00247	0.0165	0.00194	0.0152
Lambda3	0.050	0.00484	0.0331	0.00341	0.0301
Lambda4	0.050	0.00699	0.0515	0.00478	0.0462
Lambda5	0.050	0.0906	0.8325	0.0541	0.7469
Lambda6	0.050	0.00676	0.0654	0.00409	0.0580
Lambda7	0.050	0.00614	0.0648	0.00421	0.0569
Lambda8	0.050	0.0132	0.1368	0.00637	0.1207
Group	0.050	-1.3190	0.0893	-1.3379	0.0652

The requested diagnostics—namely, lag1, lag5, lag10, lag50 autocorrelations (Output 92.14.8), the Geweke diagnostics (Output 92.14.9), and the effective sample size diagnostics (Output 92.14.10)—show a good mixing of the Markov chain.

Output 92.14.8 Autocorrelations

The PHREG Procedure

Bayesian Analysis

Posterior Autocorrelations				
Parameter	**Lag 1**	**Lag 5**	**Lag 10**	**Lag 50**
Lambda1	0.0705	0.0015	0.0017	-0.0076
Lambda2	0.0909	0.0206	-0.0013	-0.0039
Lambda3	0.0861	-0.0072	0.0011	0.0002
Lambda4	0.1447	-0.0023	0.0081	0.0082
Lambda5	0.1086	0.0072	-0.0038	-0.0028
Lambda6	0.1281	0.0049	-0.0036	0.0048
Lambda7	0.1925	-0.0011	0.0094	-0.0011
Lambda8	0.2128	0.0322	-0.0042	-0.0045
Group	0.5638	0.0410	-0.0003	-0.0071

Output 92.14.9 Geweke Diagnostics

Geweke Diagnostics		
Parameter	**z**	**Pr > |z|**
Lambda1	-0.0705	0.9438
Lambda2	-0.4936	0.6216
Lambda3	0.5751	0.5652
Lambda4	1.0514	0.2931
Lambda5	0.8910	0.3729
Lambda6	0.2976	0.7660
Lambda7	1.6543	0.0981
Lambda8	0.6686	0.5038
Group	-1.2621	0.2069

Output 92.14.10 Effective Sample Size

Effective Sample Sizes			
		Autocorrelation	
Parameter	**ESS**	**Time**	**Efficiency**
Lambda1	7775.3	1.2861	0.7775
Lambda2	6874.8	1.4546	0.6875
Lambda3	7655.7	1.3062	0.7656
Lambda4	6337.1	1.5780	0.6337
Lambda5	6563.3	1.5236	0.6563
Lambda6	6720.8	1.4879	0.6721
Lambda7	5968.7	1.6754	0.5969
Lambda8	5137.2	1.9466	0.5137
Group	2980.4	3.3553	0.2980

The piecewise exponential model is an alternative to the Cox model. Another alternative model is the proportional hazards spline model of Royston and Parmar (2002). You can use the SPLINE option in the

BAYES statement to perform a Bayesian analysis of this model, as in the following statements:

```
proc phreg data=Rats;
   model Days*Status(0)=Group;
   bayes seed=1 spline statistics=(summary interval)
         diagnostics=(autocorr geweke ess);
run;
```

Example 92.15: Analysis of Competing-Risks Data

(View the complete code for this example (phrex15.sas) in the example repository.)

Bone marrow transplant (BMT) is a standard treatment for acute leukemia. Klein and Moeschberger (1997) present a set of BMT data for 137 patients, grouped into three risk categories based on their status at the time of transplant: acute lymphoblastic leukemia (ALL), acute myelocytic leukemia (AML) low-risk, and AML high-risk. During the follow-up period, some patients might relapse or some patients might die while in remission. Consider relapse to be the event of interest. Death is a competing risk because death impedes the occurrence of leukemia relapse. The Fine and Gray (1999) model is used to compare the risk categories on the disease-free survival.

The following DATA step creates the data set Bmt. The variable Disease represents the risk group of a patient, which is either ALL, AML-Low Risk, or AML-High Risk. The variable T represents the disease-free survival in days, which is the time to relapse, time to death, or censored. The variable Status has three values: 0 for censored observations, 1 for relapsed patients, and 2 for patients who die before experiencing a relapse.

```
proc format;
   value DiseaseGroup 1='ALL'
                      2='AML-Low Risk'
                      3='AML-High Risk';

data Bmt;
   input Disease T Status @@;
   label T='Disease-Free Survival in Days';
   format Disease DiseaseGroup.;
   datalines;
1   2081   0   1   1602   0   1   1496   0   1   1462   0   1   1433   0
1   1377   0   1   1330   0   1    996   0   1    226   0   1   1199   0
1   1111   0   1    530   0   1   1182   0   1   1167   0   1    418   2
1    383   1   1    276   2   1    104   1   1    609   1   1    172   2
1    487   2   1    662   1   1    194   2   1    230   1   1    526   2
1    122   2   1    129   1   1     74   1   1    122   1   1     86   2
1    466   2   1    192   1   1    109   1   1     55   1   1      1   2
1    107   2   1    110   1   1    332   2   2   2569   0   2   2506   0
2   2409   0   2   2218   0   2   1857   0   2   1829   0   2   1562   0
2   1470   0   2   1363   0   2   1030   0   2    860   0   2   1258   0
2   2246   0   2   1870   0   2   1799   0   2   1709   0   2   1674   0
2   1568   0   2   1527   0   2   1324   0   2    957   0   2    932   0
2    847   0   2    848   0   2   1850   0   2   1843   0   2   1535   0
2   1447   0   2   1384   0   2    414   2   2   2204   2   2   1063   2
2    481   2   2    105   2   2    641   2   2    390   2   2    288   2
2    421   1   2     79   2   2    748   1   2    486   1   2     48   2
2    272   1   2   1074   2   2    381   1   2     10   2   2     53   2
```

```
2     80   2   2     35   2   2    248   1   2    704   2   2    211   1
2    219   1   2    606   1   3   2640   0   3   2430   0   3   2252   0
3   2140   0   3   2133   0   3   1238   0   3   1631   0   3   2024   0
3   1345   0   3   1136   0   3    845   0   3    422   1   3    162   2
3     84   1   3    100   1   3      2   2   3     47   1   3    242   1
3    456   1   3    268   1   3    318   2   3     32   1   3    467   1
3     47   1   3    390   1   3    183   2   3    105   2   3    115   1
3    164   2   3     93   1   3    120   1   3     80   2   3    677   2
3     64   1   3    168   2   3     74   2   3     16   2   3    157   1
3    625   1   3     48   1   3    273   1   3     63   2   3     76   1
3    113   1   3    363   2
;
```

PROC PHREG enables you to plot the cumulative incidence function for each disease category, but first you must save these three Disease values in a SAS data set, as in the following DATA step:

```
data Risk;
   Disease=1; output;
   Disease=2; output;
   Disease=3; output;
   format Disease DiseaseGroup.;
   run;
```

The following statements use the PHREG procedure to fit the proportional subdistribution hazards model. To designate relapse (Status=1) as the event of interest, you specify EVENTCODE=1 in the MODEL statement. The HAZARDRATIO statement provides the hazard ratios for all pairs of disease groups. The COVARIATES= option in the BASELINE statement specifies the data set that contains the covariate settings for predicting cumulative incidence functions, and the OUT= option saves the prediction results in a SAS data set. The PLOTS= option in the PROC PHREG statement displays the cumulative incidence curves.

```
ods graphics on;
proc phreg data=Bmt plots(overlay=stratum)=cif;
   class Disease (order=internal ref=first);
   model T*Status(0)=Disease / eventcode=1;
   Hazardratio 'Pairwise' Disease / diff=pairwise;
   baseline covariates=Risk out=out1 cif=_all_ / seed=191;
run;
```

Output 92.15.1 displays the codes of different types of observations in the input data set. Relapse is the failure of interest with Status = 1, death is a competing failure with Status = 2, and censored observations are those with Status = 0. Out of the 137 transplant patients, 42 have a relapse, 41 die without experiencing a relapse, and 54 are censored (Output 92.15.2).

Output 92.15.1 Code for the Competing Failures and Censored Observations

The PHREG Procedure

Model Information		
Data Set	WORK.BMT	
Dependent Variable	T	Disease-Free Survival in Days
Status Variable	Status	
Event of Interest	1	
Competing Event	2	
Censored Value	0	

Output 92.15.2 Distribution of Events and Censored Observations

Summary of Failure Outcomes			
Total	Event of Interest	Competing Event	Censored
137	42	41	54

Output 92.15.3 shows a significant effect ($p = 0.0030$) of Disease on the disease-free survival. With the reference coding, the CLASS variable Disease is represented by two dummy variables. Parameter estimates and Wald tests for individual parameters are shown in Output 92.15.3.

Output 92.15.3 Wald Test of the Disease Effect

Type 3 Tests			
Effect	DF	Wald Chi-Square	Pr > ChiSq
Disease	2	11.6406	0.0030

Analysis of Maximum Likelihood Estimates								
Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Disease	AML-Low Risk	1	-0.80340	0.42846	3.5160	0.0608	0.448	Disease AML-Low Risk
Disease	AML-High Risk	1	0.50849	0.36618	1.9283	0.1649	1.663	Disease AML-High Risk

Hazard ratio estimates of one disease group relative to another disease group are displayed in Output 92.15.4. The hazard of relapse for the ALL patients is 2.2 times that for the AML-low risk patients, and the hazard for the AML-high risk patients is 1.7 times that for the ALL patients. It is expected that at any particular time after the transplant, an AML high-risk patient is more likely to relapse than an ALL patient, and an ALL patient is more likely to relapse than an AML low-risk patient. Such ordering of probabilities is revealed in the plot of the cumulative incidence functions in Output 92.15.5.

Output 92.15.4 Pairwise Comparison of Disease Group

Pairwise: Hazard Ratios for Disease			
Description	Point Estimate	95% Wald Confidence Limits	
Disease ALL vs AML-Low Risk	2.233	0.964	5.171
Disease AML-Low Risk vs ALL	0.448	0.193	1.037
Disease ALL vs AML-High Risk	0.601	0.293	1.233
Disease AML-High Risk vs ALL	1.663	0.811	3.408
Disease AML-Low Risk vs AML-High Risk	0.269	0.127	0.573
Disease AML-High Risk vs AML-Low Risk	3.713	1.745	7.900

Output 92.15.5 CIFs of the Three Disease Groups

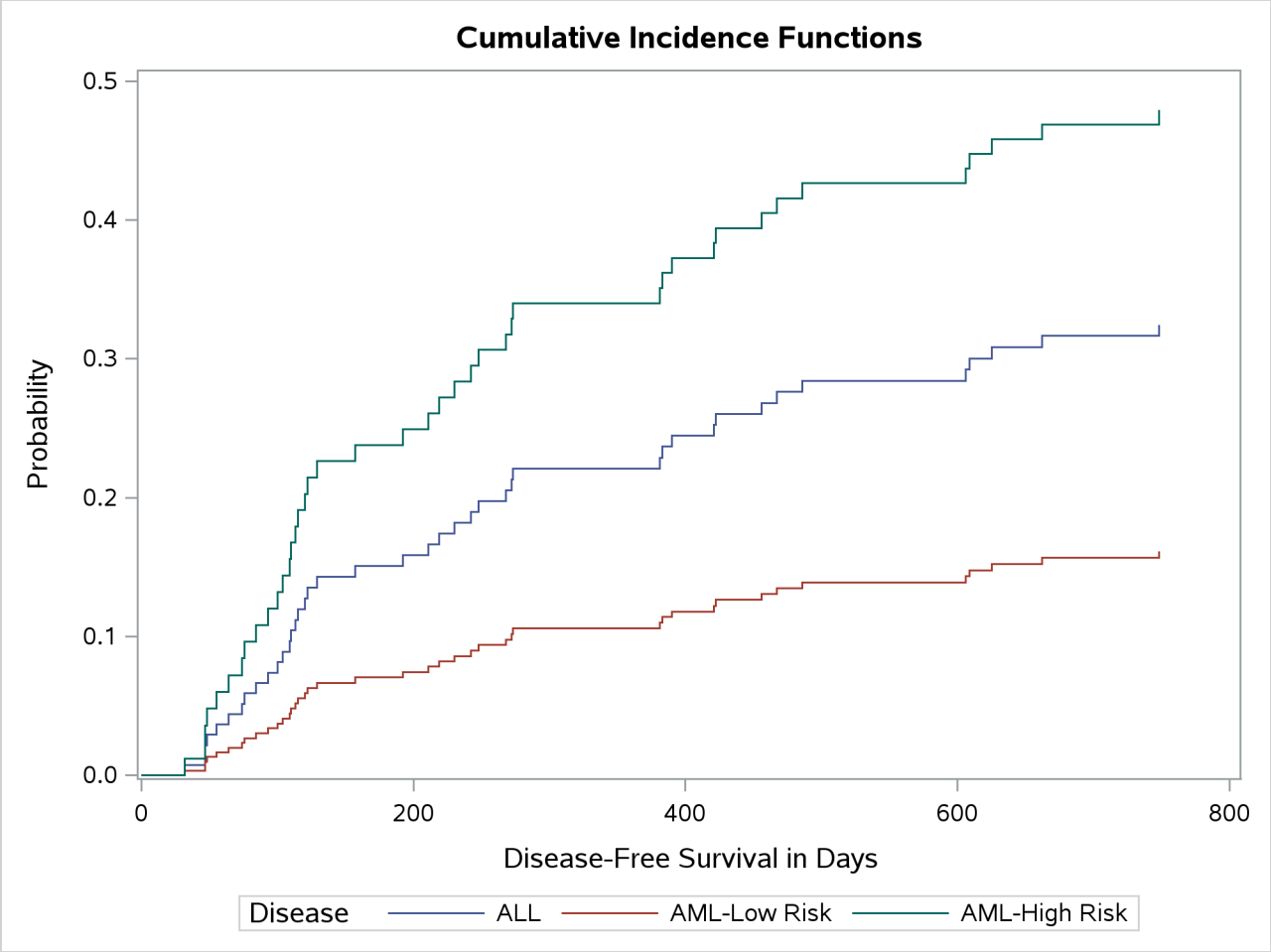

You use the following statements to display the cumulative incidence prediction for the ALL (Disease=1) risk group:

```
proc print data=Out1(where=(Disease=1));
   title 'CIFs Estimates and 95% Confidence limits for the ALL Group';
run;
```

Output 92.15.6 shows the point estimate and the confidence limits for the cumulative incidence at each distinct time when the event of interest occurred for the ALL patients. The predictions for the AML-low risk patients and AML-high risk patients are not shown.

Output 92.15.6 Cumulative Incidence Prediction for the ALL Group

CIFs Estimates and 95% Confidence limits for the ALL Group

Obs	Disease	T	CIF	StdErrCIF	LowerCIF	UpperCIF
1	ALL	0	0.00000	.	.	.
2	ALL	32	0.00727	0.007237	0.00103	0.05114
3	ALL	47	0.02183	0.014323	0.00604	0.07898
4	ALL	48	0.02922	0.017822	0.00884	0.09657
5	ALL	55	0.03663	0.019106	0.01318	0.10181
6	ALL	64	0.04405	0.019259	0.01870	0.10378
7	ALL	74	0.05151	0.019951	0.02411	0.11005
8	ALL	76	0.05897	0.025533	0.02524	0.13778
9	ALL	84	0.06646	0.025378	0.03145	0.14048
10	ALL	93	0.07400	0.025092	0.03807	0.14383
11	ALL	100	0.08158	0.030460	0.03924	0.16959
12	ALL	104	0.08920	0.029038	0.04712	0.16883
13	ALL	109	0.09682	0.033564	0.04907	0.19100
14	ALL	110	0.10443	0.035734	0.05341	0.20422
15	ALL	113	0.11205	0.041176	0.05453	0.23026
16	ALL	115	0.11972	0.037619	0.06467	0.22163
17	ALL	120	0.12742	0.036521	0.07266	0.22347
18	ALL	122	0.13518	0.042929	0.07254	0.25190
19	ALL	129	0.14293	0.041747	0.08063	0.25336
20	ALL	157	0.15068	0.046376	0.08243	0.27545
21	ALL	192	0.15848	0.051406	0.08392	0.29928
22	ALL	211	0.16628	0.058106	0.08383	0.32983
23	ALL	219	0.17404	0.056257	0.09236	0.32794
24	ALL	230	0.18185	0.053563	0.10210	0.32392
25	ALL	242	0.18967	0.065355	0.09653	0.37265
26	ALL	248	0.19753	0.057829	0.11128	0.35062
27	ALL	268	0.20535	0.054765	0.12176	0.34634
28	ALL	272	0.21322	0.058189	0.12489	0.36402
29	ALL	273	0.22105	0.061340	0.12832	0.38080
30	ALL	381	0.22893	0.061228	0.13554	0.38669
31	ALL	383	0.23677	0.062212	0.14147	0.39626
32	ALL	390	0.24461	0.063708	0.14682	0.40754
33	ALL	421	0.25250	0.070833	0.14571	0.43757
34	ALL	422	0.26035	0.063694	0.16118	0.42053
35	ALL	456	0.26825	0.067518	0.16379	0.43932
36	ALL	467	0.27621	0.073253	0.16424	0.46450
37	ALL	486	0.28422	0.066216	0.18004	0.44871
38	ALL	606	0.29233	0.067521	0.18590	0.45971
39	ALL	609	0.30039	0.079301	0.17905	0.50396
40	ALL	625	0.30845	0.067182	0.20128	0.47270
41	ALL	662	0.31657	0.070668	0.20439	0.49033
42	ALL	748	0.32469	0.082845	0.19692	0.53537

In contrast to the preceding proportional subdistribution hazard regression, the cause-specific analysis fits a separate Cox regression model that targets each event type and censors all other event types. The estimated

CIF for a specific event type is based on all the cause-specific hazard models.

The following statements use the PHREG procedure to perform the cause-specific analysis and estimate the CIF for the event of relapse (Status=1) by specifying the MODEL statement option EVENTCODE(COX)=1. The COVARIATES= option in the BASELINE statement specifies the data set that contains the covariate settings for predicting cumulative incidence functions for relapse, and the OUT= option saves the prediction results in a SAS data set. The PLOTS= option in the PROC PHREG statement displays the cumulative incidence curves for relapse.

```
ods graphics on;
title 'Cause-specific Analysis';
proc phreg data=Bmt plots(overlay)=cif;
   class Disease (order=internal ref=first);
   model T*Status(0)=Disease / eventcode(cox)=1;
   baseline covariates=Risk out=out2 cif=_all_;
run;
```

Output 92.15.7 and Output 92.15.8 show that the risk factor Disease has different effects on the failure outcomes.

Output 92.15.7 Parameter Estimates for Relapse (Status=1)

Cause-specific Analysis

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Disease	AML-Low Risk	1	-0.89403	0.44222	4.0872	0.0432	0.409	Disease AML-Low Risk
Disease	AML-High Risk	1	0.61009	0.36213	2.8384	0.0920	1.841	Disease AML-High Risk

Output 92.15.8 Parameter Estimates for Death (Status=2)

Analysis of Maximum Likelihood Estimates

Parameter		DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Disease	AML-Low Risk	1	-0.33593	0.38508	0.7610	0.3830	0.715	Disease AML-Low Risk
Disease	AML-High Risk	1	0.09667	0.40220	0.0578	0.8101	1.101	Disease AML-High Risk

Often the focus of a competing-risks analysis might not be on the parameter estimates, but on the cumulative incidence for specified covariate values. Output 92.15.9 displays the CIF of the event Relapse (Status=1) for the three Disease groups. The curves resemble those derived from the Fine and Gray model (Output 92.15.5).

Output 92.15.9 CIFs of the Three Disease Groups (Cause-Specific Analysis) for the All Group

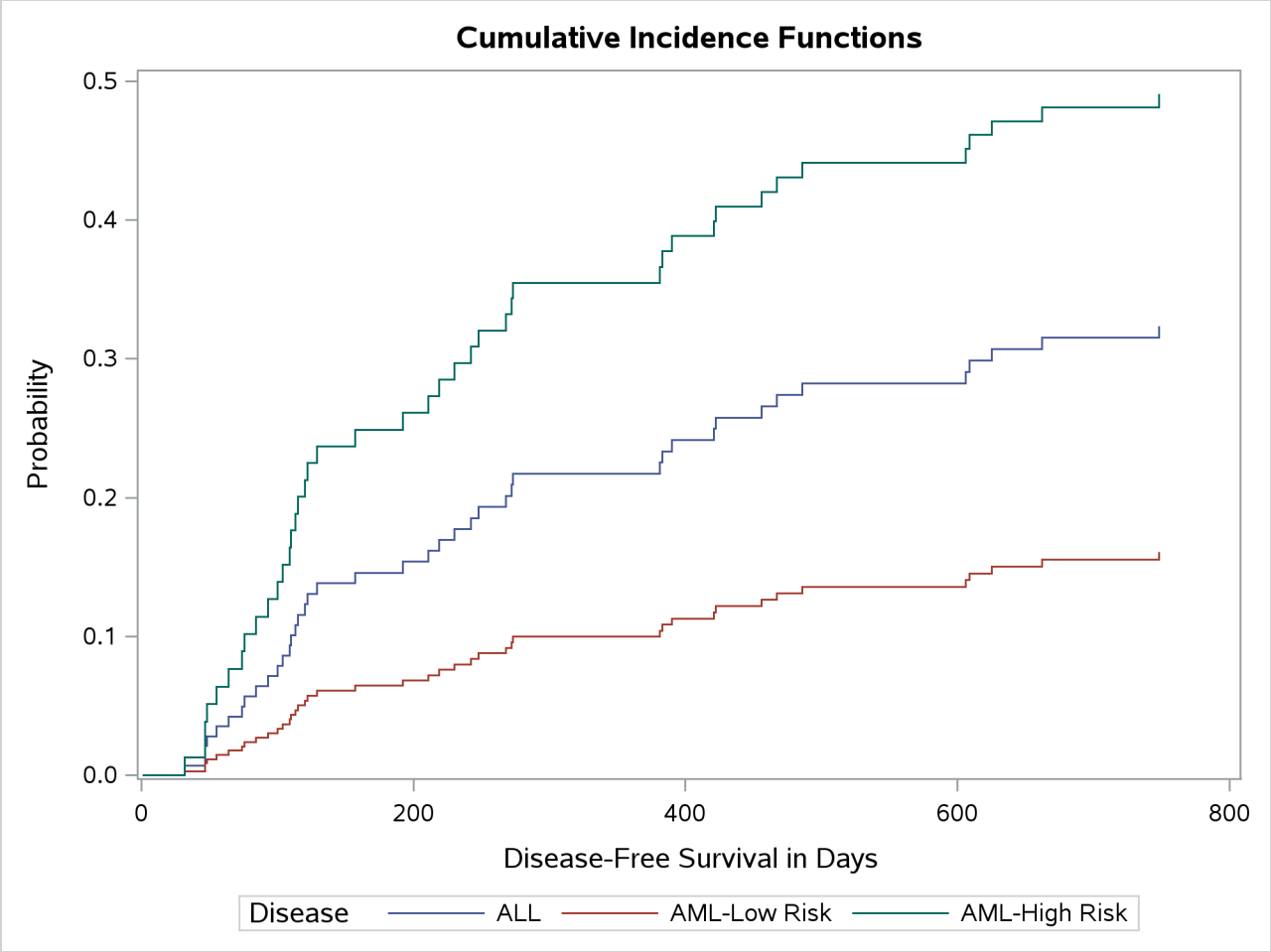

You use the following statements to display the cumulative incidence prediction for the ALL risk group (Disease=1):

```
proc print data=Out2(where=(Disease=1));
   title 'CIF Estimates and 95% Confidence limits for the ALL Group';
run;
```

Output 92.15.10 shows the point estimate and the confidence limits for the cumulative incidence at each distinct time when relapse (the event of interest) occurred for the ALL patients. It appears that the results are similar to the predictions from the Fine and Gray analysis.

Output 92.15.10 Cumulative Incidence Prediction (Cause-Specific Analysis) for the ALL Group

CIF Estimates and 95% Confidence limits for the ALL Group

Obs	Disease	T	CIF	StdErrCIF	LowerCIF	UpperCIF
1	ALL	0	0.00000	.	.	.
2	ALL	32	0.00703	0.007196	0.00094	0.05230
3	ALL	47	0.02109	0.013077	0.00626	0.07110
4	ALL	48	0.02821	0.015490	0.00962	0.08276
5	ALL	55	0.03530	0.017695	0.01322	0.09429
6	ALL	64	0.04243	0.019793	0.01701	0.10587
7	ALL	74	0.04961	0.021812	0.02096	0.11744
8	ALL	76	0.05683	0.023773	0.02503	0.12902
9	ALL	84	0.06407	0.025660	0.02922	0.14046
10	ALL	93	0.07135	0.027513	0.03351	0.15192
11	ALL	100	0.07868	0.029329	0.03789	0.16337
12	ALL	104	0.08607	0.031112	0.04238	0.17480
13	ALL	109	0.09346	0.032867	0.04691	0.18620
14	ALL	110	0.10085	0.034598	0.05149	0.19756
15	ALL	113	0.10825	0.036306	0.05609	0.20889
16	ALL	115	0.11570	0.037994	0.06078	0.22022
17	ALL	120	0.12321	0.039660	0.06556	0.23155
18	ALL	122	0.13079	0.041305	0.07043	0.24288
19	ALL	129	0.13836	0.042948	0.07530	0.25423
20	ALL	157	0.14594	0.044576	0.08020	0.26556
21	ALL	192	0.15385	0.046239	0.08536	0.27728
22	ALL	211	0.16176	0.047906	0.09053	0.28904
23	ALL	219	0.16962	0.049524	0.09571	0.30061
24	ALL	230	0.17752	0.051140	0.10093	0.31222
25	ALL	242	0.18541	0.052744	0.10617	0.32380
26	ALL	248	0.19339	0.054330	0.11151	0.33540
27	ALL	268	0.20131	0.055867	0.11686	0.34681
28	ALL	272	0.20932	0.057383	0.12231	0.35823
29	ALL	273	0.21727	0.058852	0.12777	0.36945
30	ALL	381	0.22542	0.060350	0.13339	0.38096
31	ALL	383	0.23351	0.061798	0.13900	0.39226
32	ALL	390	0.24159	0.063238	0.14464	0.40354
33	ALL	421	0.24964	0.064650	0.15027	0.41471
34	ALL	422	0.25760	0.066011	0.15589	0.42567
35	ALL	456	0.26568	0.067348	0.16165	0.43665
36	ALL	467	0.27387	0.068691	0.16752	0.44776
37	ALL	486	0.28211	0.069989	0.17348	0.45877
38	ALL	606	0.29042	0.071368	0.17941	0.47011
39	ALL	609	0.29863	0.072688	0.18533	0.48119
40	ALL	625	0.30685	0.074004	0.19126	0.49228
41	ALL	662	0.31511	0.075276	0.19729	0.50327
42	ALL	748	0.32342	0.076548	0.20338	0.51432

Example 92.16: Concordance and ROC Curves

(View the complete code for this example (phrex16.sas) in the example repository.)

The concordance statistic and ROC curves are popular diagnostic tools for a logistic regression model. For a survival model, ROC curves are time-sensitive. That is, you might have different ROC curves at different time points. Two different ways of computing the concordance statistic are available. Each method estimates a different measure of concordance probability.

The data set Liver, presented in Example 92.12, is used in this example to illustrate the concordance statistic and time-dependent ROC curves. The data set consists of 418 patients who have primary biliary cirrhosis (PBC). In the data set, the variable Time represents the follow-up time in years (which is the time from registration to the earlier of liver transplantation, death, or study termination), the variable Status is the censoring indicator (1 for death and 0 for censored), and the explanatory variables are Age (age in years), Albumin (serum albumin level in g/dl), Bilirubin (serum bilirubin level in mg/dl), Edema (presence of edema), and Protime (prothrombin time in seconds).

The following statements use the PHREG procedure to fit the Cox regression model that uses Bilirubin, Age, and Edema as explanatory variables. The CONCORDANCE option in the PROC PHREG statement requests that Harrell's concordance statistics be displayed. The PLOTS=ROC option plots the time-dependent ROC curves at time points 2, 4, 6, 8, and 10 years, which are specified in the AT= suboption in the ROCOPTIONS option.

```
ods graphics on;
proc phreg data=Liver concordance plots=roc rocoptions(at=2 to 10 by 2);
   model Time*Status(0)=Bilirubin Age Edema;
run;
```

Results of the Harrell concordance statistics are shown in Output 92.16.1. There are 34,798 concordance pairs, 8,884 discordance pairs, 2 pairs that are tied in the linear predictor, and 5 pairs that are tied in the follow-up time, which gives a concordance estimate of 0.7966. You can specify CONCORDANCE=HARRELL(SE) to compute the standard error of Harrell's concordance statistic, or you can use specify CONCORDANCE=UNO for an alternative concordance measure.

Output 92.16.1 Concordance Statistics

The PHREG Procedure

Harrell's Concordance Statistic					
		Comparable Pairs			
Source	Estimate	Concordance	Discordance	Tied in Predictor	Tied in Time
Model	0.7966	34798	8884	2	5

Output 92.16.2 shows the time-dependent ROC curves at the selected years. By default, these curves are computed by the nearest neighbors technique of Heagerty, Lumley, and Pepe (2000) and are displayed in a panel. It appears that among the five selected years, year 4 has the largest area under the curve (AUC), year 8 has the lowest AUC, and the others are in between. If you specify the OVERLAY=INDIVIDUAL global plot option to display individual plots, each plot also displays the area under the ROC curve.

Output 92.16.2 ROC Plot at Selected Time Points

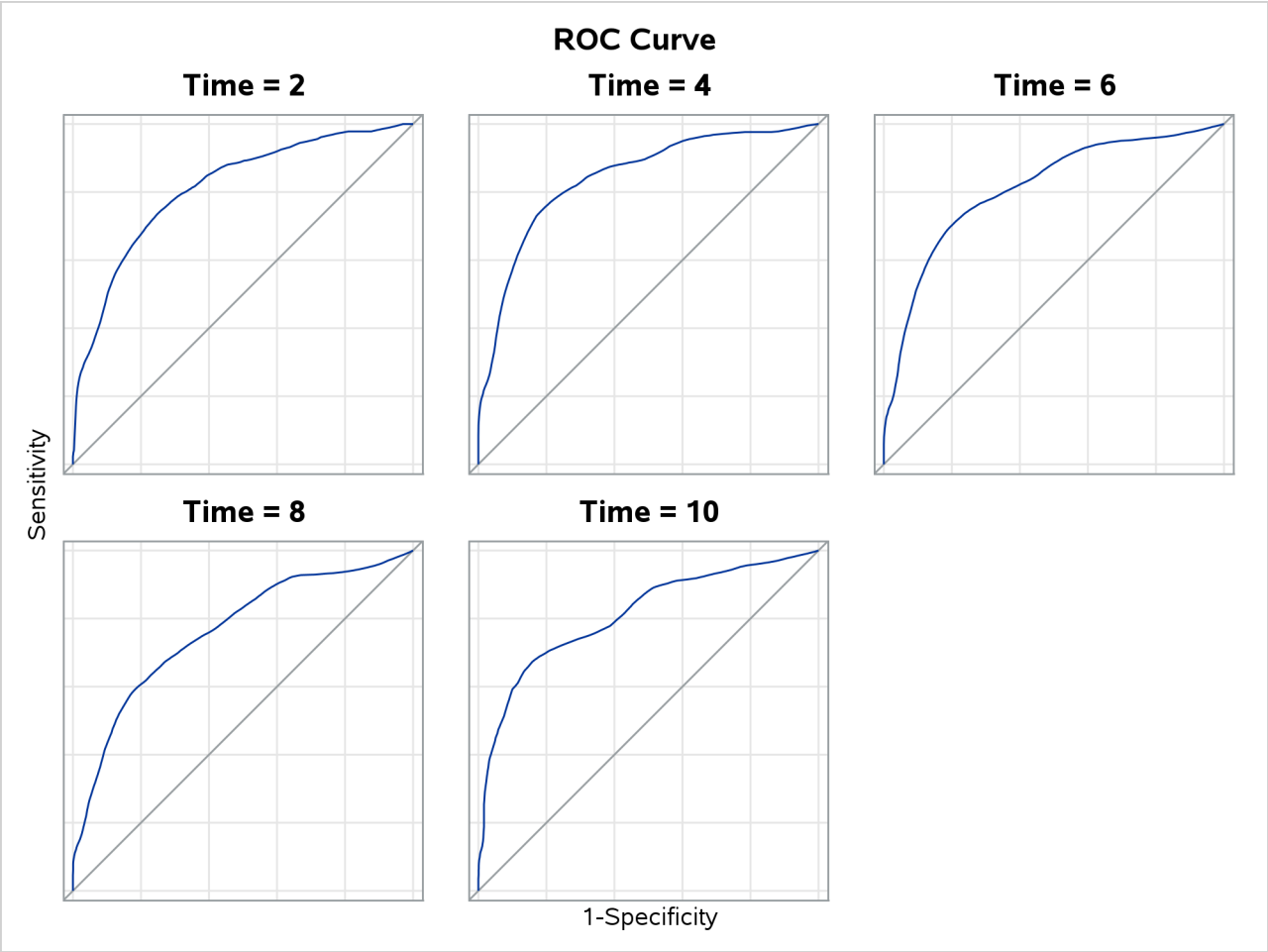

Time-dependent ROC curves change only at the distinct event times. You can examine the area under the curve at all distinct event times by plotting the curve of the AUC. The following statements plot the curve of the AUC of the fitted model and display the 95% pointwise confidence limits. The PLOTS=AUC option in the PROC PHREG statement plots the AUC curve. The ROCOPTIONS in the PROC PHREG statement enables you to specify the inverse probability of censoring weighting (IPCW) method to compute the ROC curves, and the CL suboption requests pointwise confidence limits for the AUC curve.

```
proc phreg data=Liver plots=auc rocoptions(method=ipcw(cl seed=1234));
   model Time*Status(0)=Bilirubin Age Edema;
run;
```

Output 92.16.3 displays the AUC curve and the 95% confidence limits for the fitted model. The AUC statistic reaches a high of 0.92 at Year 0.21 but mostly hovers around 0.8.

Output 92.16.3 AUC Plot with 95% Confidence Limits

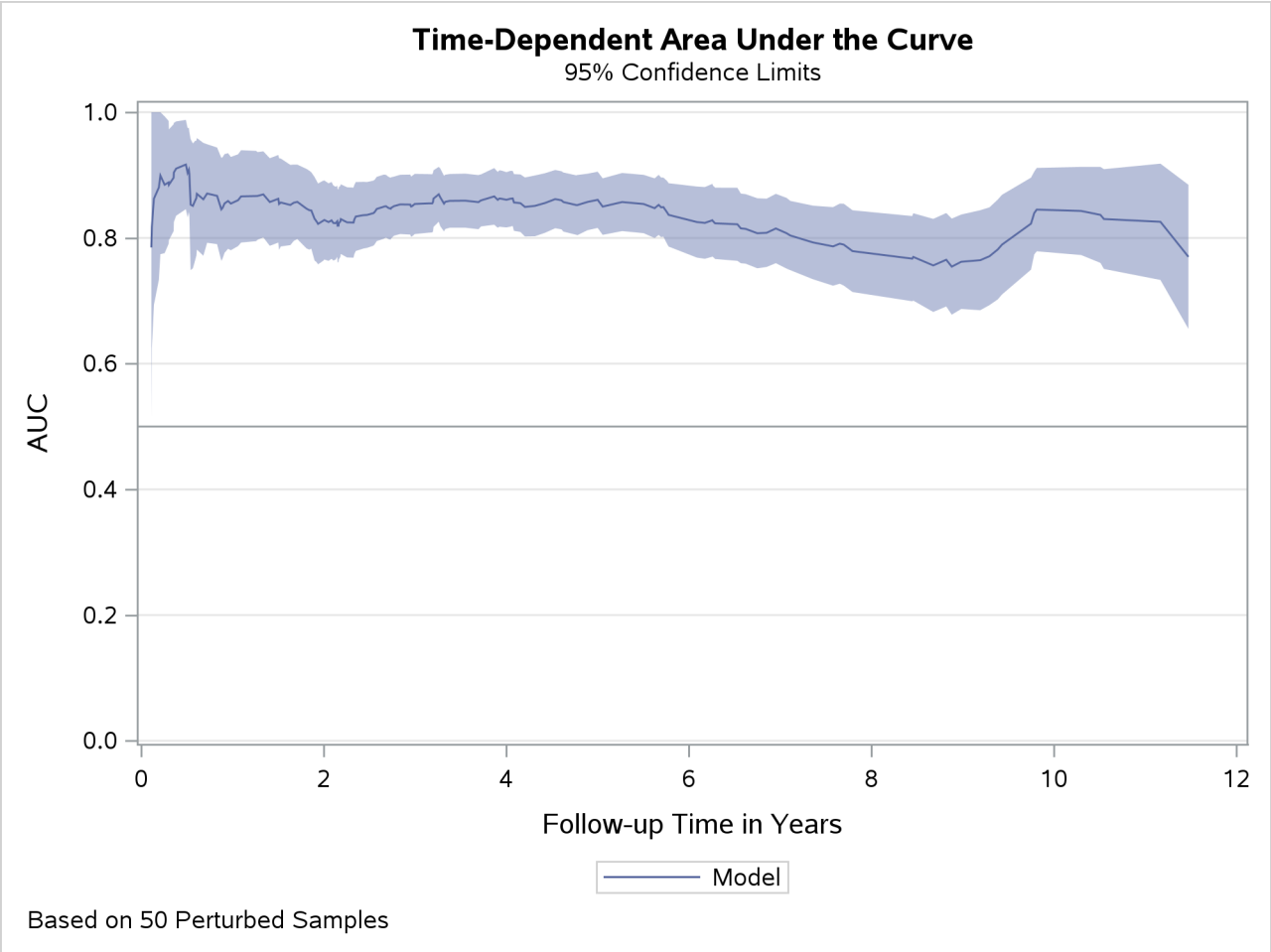

Consider three submodels of the previously fitted Cox model, each of which contains two of the three covariates: Bilirubin, Age, and Edema. You can use the Uno et al. (2011) methodology to assess the difference of the concordance probabilities between any two submodels. In the following statements, three ROC statements are specified, one for each submodel. The DIFF suboption in CONCORDANCE=UNO in the PROC PHREG statement requests that all pairwise differences be calculated. The SE suboption requests that standard error be computed for each pairwise difference, based on 100 perturbation samples as specified by the ITER= suboption. The seed of the random generator for the perturbation resampling is set to be 1234. The NOFIT option is specified in the MODEL statement, because there is no need to fit the specified model.

```
proc phreg data=Liver concordance=uno(diff se seed=1234 iter=100);
   model Time*Status(0)=Bilirubin Age Edema / nofit;
   roc 'Bilirubin+Age' Bilirubin Age;
   roc 'Age+Edema' Age Edema;
   roc 'Bilirubin+Edema' Bilirubin Edema;
run;
```

Output 92.16.4 displays results of the concordance analysis. It appears that the two submodels that contain Bilirubin have a significantly larger concordance probability than the submodel without Bilirubin. In other words, the two submodels that contain Bilirubin predict the survival outcomes better than the model without Bilirubin.

Output 92.16.4 Comparing Uno's Concordance Estimates

The PHREG Procedure

Differences in Uno's Concordance Statistic					
Source	**_Source**	**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Bilirubin+Age	**Age+Edema**	0.0972	0.0232	17.57	<.0001
Bilirubin+Age	**Bilirubin+Edema**	-0.0264	0.0231	1.31	0.2529
Age+Edema	**Bilirubin+Edema**	-0.1236	0.0287	18.51	<.0001

Example 92.12 demonstrated that the log transform is a much improved functional form for Bilirubin in a Cox regression model. It is expected that the model with Bilirubin in the log scale would have a better discriminating power than the model with Bilirubin in the original scale. In the following statements, PROC PHREG is used to fit the model with the log transform for Bilirubin. By using the OUTPUT statement, the linear predictor variable is saved as the variable Y in the output data set Liver2. With Liver2 as the input data set, PROC PHREG is called to fit the Cox model with Bilirubin in the original scale. The linear predictor variable Y is specified in the ROC statement as the PRED= variable.

```
proc phreg data=Liver;
   model Time*Status(0)=logBilirubin Age Edema;
   logBilirubin = log(Bilirubin);
   output out=Liver2 xbeta=Y;
run;
proc phreg data=Liver2 plots=roc rocoptions(at=2 to 10 by 2);
   model Time*Status(0)=Bilirubin Age Edema / roclabel='Bilirubin';
   roc 'logBilirubin' pred=Y;
run;
```

Output 92.16.5 displays the ROC curves of the two competing models. The ROC curve for the model with the log scale for Bilirubin essentially lies above that of its counterpart with the original scale for all the selected time points. This leads to the conclusion that the log transform for Bilirubin improves the predictive power of the model.

Output 92.16.5 ROC Plots to Evaluate the Log Transform for Bilirubin

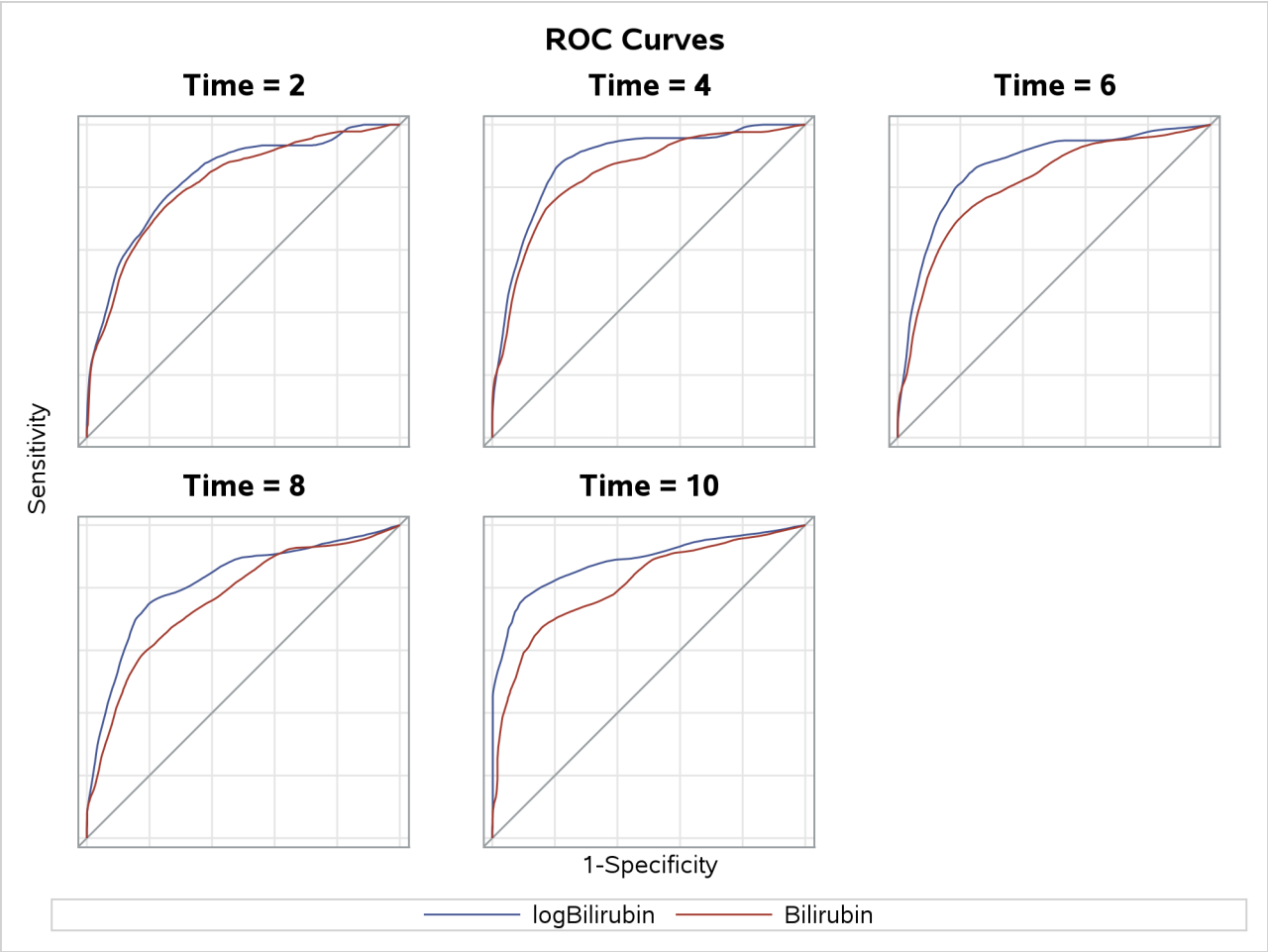

References

Akritas, M. G. (1994). "Nearest Neighbor Estimation of a Bivariate Distribution under Random Censoring." *Annals of Statistics* 22:1299–1327.

Andersen, P. K., Borgan, Ø., Gill, R. D., and Keiding, N. (1992). *Statistical Models Based on Counting Processes*. New York: Springer-Verlag.

Andersen, P. K., and Gill, R. D. (1982). "Cox's Regression Model Counting Process: A Large Sample Study." *Annals of Statistics* 10:1100–1120.

Binder, D. A. (1992). "Fitting Cox's Proportional Hazards Models from Survey Data." *Biometrika* 79:139–147.

Blanche, P., Latouche, A., and Viallon, V. (2013). "Time-Dependent AUC with Right-Censored Data: A Survey." In *Risk Assessment and Evaluation of Predictions*, edited by M.-L. T. Lee, M. Gail, R. Pfeiffer, G. Satten, T. Cai, and A. Gandy, 239–251. New York: Springer.

Borgan, Ø., and Liestøl, K. (1990). "A Note on Confidence Interval and Bands for the Survival Curves Based on Transformations." *Scandinavian Journal of Statistics* 18:35–41.

Breslow, N. E. (1972). "Discussion of Professor Cox's Paper." *Journal of the Royal Statistical Society, Series B* 34:216–217.

Breslow, N. E. (1974). "Covariance Analysis of Censored Survival Data." *Biometrics* 30:89–99.

Breslow, N. E., and Clayton, D. G. (1993). "Approximate Inference in Generalized Linear Mixed Models." *Journal of the American Statistical Association* 88:9–25.

Bryson, M. C., and Johnson, M. E. (1981). "The Incidence of Monotone Likelihood in the Cox Model." *Technometrics* 23:381–383.

Cain, K. C., and Lange, N. T. (1984). "Approximate Case Influence for the Proportional Hazards Regression Model with Censored Data." *Biometrics* 40:493–499.

Chambless, L. E., and Diao, G. (2006). "Estimation of Time-Dependent Area under the ROC Curve for Long-Term Risk Prediction." *Statistics in Medicine* 25:3474–3486.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220. With discussion.

Cox, D. R. (1975). "Partial Likelihood." *Biometrika* 62:269–276.

Crowley, J., and Hu, M. (1977). "Covariance Analysis of Heart Transplant Survival Data." *Journal of the American Statistical Association* 72:27–36.

DeLong, D. M., Guirguis, G. H., and So, Y. C. (1994). "Efficient Computation of Subset Selection Probabilities with Application to Cox Regression." *Biometrika* 81:607–611.

Efron, B. (1977). "The Efficiency of Cox's Likelihood Function for Censored Data." *Journal of the American Statistical Association* 72:557–565.

Fibrinogen Studies Collaboration (2009). "Measures to Assess the Prognostic Ability of the Stratified Cox Proportional Hazards Model." *Statistics in Medicine* 28:389–411.

Fine, J. P., and Gray, R. J. (1999). "A Proportional Hazards Model for the Subdistribution of a Competing Risk." *Journal of the American Statistical Association* 94:496–509.

Firth, D. (1993). "Bias Reduction of Maximum Likelihood Estimates." *Biometrika* 80:27–38.

Fleming, T. R., and Harrington, D. P. (1984). "Nonparametric Estimation of the Survival Distribution in Censored Data." *Communications in Statistics—Theory and Methods* 13:2469–2486.

Fleming, T. R., and Harrington, D. P. (1991). *Counting Processes and Survival Analysis*. New York: John Wiley & Sons.

Furnival, G. M., and Wilson, R. W. (1974). "Regression by Leaps and Bounds." *Technometrics* 16:499–511.

Gail, M. H., and Byar, D. P. (1986). "Variance Calculations for Direct Adjusted Survival Curves, with Applications to Testing for No Treatment Effect." *Biometrical Journal* 28:587–599.

Gail, M. H., Lubin, J. H., and Rubinstein, L. V. (1981). "Likelihood Calculations for Matched Case-Control Studies and Survival Studies with Tied Death Times." *Biometrika* 68:703–707.

Gilks, W. R., Best, N. G., and Tan, K. K. C. (1995). "Adaptive Rejection Metropolis Sampling within Gibbs Sampling." *Journal of the Royal Statistical Society, Series C* 44:455–472.

Grambsch, P. M., and Therneau, T. M. (1994). "Proportional Hazards Tests and Diagnostics Based on Weighted Residuals." *Biometrika* 81:515–526.

Gray, R. J. (1992). "Flexible Method for Analyzing Survival Data Using Splines, with Applications to Breast Cancer Prognosis." *Journal of the American Statistical Association* 87:942–951.

Harrell, F. E. (1986). "The PHGLM Procedure." In *SUGI Supplemental Library Guide, Version 5 Edition*. Cary, NC: SAS Institute Inc.

Harrell, F. E., Jr., Lee, K. L., Califf, R. M., Pryor, D. B., and Rosati, R. A. (1984). "Regression Modelling Strategies for Improved Prognostic Prediction." *Statistics in Medicine* 3:143–152.

Heagerty, P. J., Lumley, T., and Pepe, M. S. (2000). "Time-Dependent ROC Curves for Censored Survival Data and a Diagnostic Marker." *Biometrics* 56:337–344.

Heagerty, P. J., and Zheng, Y. (2005). "Survival Model Predictive Accuracy and ROC Curves." *Biometrics* 61:92–105.

Heinze, G. (1999). *The Application of Firth's Procedure to Cox and Logistic Regression*. Technical Report 10, updated January 2001, Department of Medical Computer Sciences, Section of Clinical Biometrics, University of Vienna.

Heinze, G., and Schemper, M. (2001). "A Solution to the Problem of Monotone Likelihood in Cox Regression." *Biometrics* 51:114–119.

Hosmer, D. W., Jr., and Lemeshow, S. (1989). *Applied Logistic Regression*. New York: John Wiley & Sons.

Ibrahim, J. G., Chen, M.-H., and Sinha, D. (2001). *Bayesian Survival Analysis*. New York: Springer-Verlag.

Kalbfleisch, J. D., and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley & Sons.

Kang, L., Chen, W., Petrick, N. A., and Gallas, B. D. (2015). "Comparing Two Correlated C Indices with Right-Censored Survival Outcome: A One-Shot Nonparametric Approach." *Statistics in Medicine* 34:685–703.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998). "Markov Chain Monte Carlo in Practice: A Roundtable Discussion." *American Statistician* 52:93–100.

Klein, J. P., and Moeschberger, M. L. (1997). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: Springer-Verlag.

Klein, J. P., and Moeschberger, M. L. (2003). *Survival Analysis: Techniques for Censored and Truncated Data*. 2nd ed. New York: Springer-Verlag.

Krall, J. M., Uthoff, V. A., and Harley, J. B. (1975). "A Step-Up Procedure for Selecting Variables Associated with Survival." *Biometrics* 31:49–57.

Lawless, J. F. (2003). *Statistical Model and Methods for Lifetime Data*. 2nd ed. New York: John Wiley & Sons.

Lawless, J. F., and Nadeau, C. (1995). "Some Simple Robust Methods for the Analysis of Recurrent Events." *Technometrics* 37:158–168.

Lee, E. W., Wei, L. J., and Amato, D. A. (1992). "Cox-Type Regression Analysis for Large Numbers of Small Groups of Correlated Failure Time Observations." In *Survival Analysis: State of the Art*, edited by J. P. Klein and P. K. Goel, 237–247. Dordrecht, Netherlands: Kluwer Academic.

Li, L., Greene, T., and Hu, B. (2018). "A Simple Method to Estimate the Time-Dependent Receiver Operating Characteristic Curve and the Area under the Curve with Right Censored Data." *Statistical Methods in Medical Research* 27:2264–2278.

Lin, D. Y. (1994). "Cox Regression Analysis of Multivariate Failure Time Data: The Marginal Approach." *Statistics in Medicine* 13:2233–2247.

Lin, D. Y., and Wei, L. J. (1989). "The Robust Inference for the Proportional Hazards Model." *Journal of the American Statistical Association* 84:1074–1078.

Lin, D. Y., Wei, L. J., Yang, I., and Ying, Z. (2000). "Semiparametric Regression for the Mean and Rate Functions of Recurrent Events." *Journal of the Royal Statistical Society, Series B* 62:711–730.

Lin, D. Y., Wei, L. J., and Ying, Z. (1993). "Checking the Cox Model with Cumulative Sums of Martingale-Based Residuals." *Biometrika* 80:557–572.

Littell, R. C., Freund, R. J., and Spector, P. C. (1991). *SAS System for Linear Models*. 3rd ed. Cary, NC: SAS Institute Inc.

Makuch, R. W. (1982). "Adjusted Survival Curve Estimation Using Covariates." *Journal of Chronic Diseases* 35:437–443.

Muller, K. E., and Fetterman, B. A. (2002). *Regression and ANOVA: An Integrated Approach Using SAS Software*. Cary, NC: SAS Institute Inc.

Nelson, W. (2003). *Recurrent Events Data Analysis for Product Repairs, Disease Recurrences, and Other Applications*. Philadelphia: SIAM.

Neuberger, J., Altman, D. G., Christensen, E., Tygstrup, N., and Williams, R. (1986). "Use of a Prognostic Index in Evaluation of Liver Transplantation for Primary Biliary Cirrhosis." *Transplantation* 41:713–716.

Pepe, M. S., and Cai, J. (1993). "Some Graphical Displays and Marginal Regression Analyses for Recurrent Failure Times and Time Dependent Covariates." *Journal of the American Statistical Association* 88:811–820.

Pettitt, A. N., and Bin Daud, I. (1989). "Case-Weighted Measures of Influence for Proportional Hazards Regression." *Journal of the Royal Statistical Society, Series C* 38:313–329.

Prentice, R. L., Williams, B. J., and Peterson, A. V. (1981). "On the Regression Analysis of Multivariate Failure Time Data." *Biometrika* 68:373–379.

Reid, N., and Crépeau, H. (1985). "Influence Functions for Proportional Hazards Regression." *Biometrika* 72:1–9.

Ripatti, S., and Palmgren, J. (2000). "Estimation of Multivariate Frailty Models Using Penalized Partial Likelihood." *Biometrics* 56:1016–1022.

Royston, P., and Parmar, M. K. B. (2002). "Flexible Parametric Proportional-Hazards and Proportional-Odds Models for Censored Survival Data, with Application to Prognostic Modelling and Estimation of Treatment Effects." *Statistics in Medicine* 21:2175–2197.

Sargent, D. J. (1998). "A General Framework for Random Effects Survival Analysis in the Cox Proportional Hazards Setting." *Biometrics* 54:1486–1497.

Schemper, M., and Henderson, R. (2000). "Predictive Accuracy and Explained Variation in Cox Regression." *Biometrics* 56:249–255.

Schoenfeld, D. A. (1982). "Partial Residuals for the Proportional Hazards Regression Model." *Biometrika* 69:239–241.

Simonsen, J. (2014). Statens Serum Institut, Copenhagen. Unpublished SAS macro.

Sinha, D., Ibrahim, J. G., and Chen, M.-H. (2003). "A Bayesian Justification of Cox's Partial Likelihood." *Biometrika* 90:629–641.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and Van der Linde, A. (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society, Series B* 64:583–616. With discussion.

Spiekerman, C. F., and Lin, D. Y. (1998). "Marginal Regression Models for Multivariate Failure Time Data." *Journal of the American Statistical Association* 93:1164–1175.

Therneau, T. M. (1994). *A Package for Survival Analysis in S*. Technical Report 53, Section of Biostatistics, Mayo Clinic, Rochester, MN.

Therneau, T. M., and Grambsch, P. M. (2000). *Modeling Survival Data: Extending the Cox Model*. New York: Springer-Verlag.

Therneau, T. M., Grambsch, P. M., and Fleming, T. R. (1990). "Martingale-Based Residuals and Survival Models." *Biometrika* 77:147–160.

Tsiatis, A. A. (1981). "A Large Sample Study of the Estimates for the Integrated Hazard Function in Cox's Regression Model for Survival Data." *Annals of Statistics* 9:93–108.

Uno, H., Cai, T., Pencina, M. J., D'Agostino, R. B., and Wei, L. J. (2011). "On the C-Statistics for Evaluating Overall Adequacy of Risk Prediction Procedures with Censored Survival Data." *Statistics in Medicine* 30:1105–1117.

Uno, H., Cai, T., Tian, L., and Wei, L. J. (2007). "Evaluating Prediction Rules for t-Year Survivors with Censored Regression Models." *Journal of the American Statistical Association* 102:527–537.

Venzon, D. J., and Moolgavkar, S. H. (1988). "A Method for Computing Profile-Likelihood-Based Confidence Intervals." *Journal of the Royal Statistical Society, Series C* 37:87–94.

Volinsky, C. T., and Raftery, A. E. (2000). "Bayesian Information Criterion for Censored Survival Models." *Biometrics* 56:256–262.

Wei, L. J., Lin, D. Y., and Weissfeld, L. (1989). "Regression Analysis of Multivariate Incomplete Failure Time Data by Modeling Marginal Distribution." *Journal of the American Statistical Association* 84:1065–1073.

Zhang, X., Loberiza, F. R., Klein, J. P., and Zhang, M. J. (2007). "A SAS Macro for Estimation of Direct Adjusted Survival Curves Based on a Stratified Cox Regression Model." *Computer Methods and Programs in Biomedicine* 88:95–111.

Zhou, B., Fine, J., Latouche, A., and Labopin, M. (2012). "Competing Risks Regression for Cluster Data." *Biostatistics* 13:371–383.

Zhou, B., Latouche, A., Rocha, V., and Fine, J. (2011). "Competing Risks Regression for Stratified Data." *Biometrics* 67:661–670.

Chapter 93
The PLAN Procedure

Contents

Overview: PLAN Procedure **7742**
Getting Started: PLAN Procedure **7743**
Three Replications with Four Factors 7743
Randomly Assigning Subjects to Treatments 7744
Syntax: PLAN Procedure **7746**
PROC PLAN Statement 7746
FACTORS Statement 7747
OUTPUT Statement 7750
TREATMENTS Statement 7752
Details: PLAN Procedure **7753**
Using PROC PLAN Interactively 7753
Output Data Sets 7753
Specifying Factor Structures 7755
Randomizing Designs 7757
Displayed Output 7757
ODS Table Names 7758
Examples: PLAN Procedure **7758**
Example 93.1: A Split-Plot Design 7758
Example 93.2: A Hierarchical Design 7759
Example 93.3: An Incomplete Block Design 7761
Example 93.4: A Latin Square Design 7763
Example 93.5: A Generalized Cyclic Incomplete Block Design 7765
Example 93.6: Permutations and Combinations 7766
Example 93.7: Crossover Designs 7770
References **7774**

Overview: PLAN Procedure

The PLAN procedure constructs designs and randomizes plans for factorial experiments, especially nested and crossed experiments and randomized block designs. PROC PLAN can also be used for generating lists of permutations and combinations of numbers. The PLAN procedure can construct the following types of experimental designs:

- full factorial designs, with and without randomization
- certain balanced and partially balanced incomplete block designs
- generalized cyclic incomplete block designs
- Latin square designs

For other kinds of experimental designs, especially fractional factorial, response surface, and orthogonal array designs, see the FACTEX and OPTEX procedures in SAS/QC software.

PROC PLAN generates designs by first generating a selection of the levels for the first factor. Then, for the second factor, PROC PLAN generates a selection of its levels for each level of the first factor. In general, for a given factor, the PLAN procedure generates a selection of its levels for all combinations of levels for the factors that precede it.

The selection can be done in five different ways:

- randomized selection, for which the levels are returned in a random order
- ordered selection, for which the levels are returned in a standard order every time a selection is generated
- cyclic selection, for which the levels returned are computed by cyclically permuting the levels of the previous selection
- permuted selection, for which the levels are a permutation of the integers $1, \ldots, n$
- combination selection, for which the m levels are selected as a combination of the integers $1, \ldots, n$ taken m at a time

The randomized selection method can be used to generate randomized plans. Also, by appropriate use of cyclic selection, any of the designs in the very wide class of generalized cyclic block designs (Jarrett and Hall 1978) can be generated.

There is no limit to the depth to which the different factors can be nested, and any number of randomized plans can be generated.

You can also declare a list of factors to be selected simultaneously with the lowest (that is, the most nested) factor. The levels of the factors in this list can be seen as constituting the treatment to be applied to the cells of the design. For this reason, factors in this list are called *treatments*. With this list, you can generate and randomize plans in one run of PROC PLAN.

Getting Started: PLAN Procedure

Three Replications with Four Factors

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/planin1.sas.)

Suppose you want to determine if the order in which four drugs are given affects the response of a subject. If you have only three subjects to test, you can use the following statements to design the experiment.

```
proc plan seed=27371;
   factors Replicate=3 ordered Drug=4;
run;
```

These statements produce a design with three replicates of the four levels of the factor Drug arranged in random order. The three levels of Replicate are arranged in order, as shown in Figure 93.1.

Figure 93.1 Three Replications and Four Factors

The PLAN Procedure

Factor	Select	Levels	Order
Replicate	3	3	Ordered
Drug	4	4	Random

Replicate	Drug			
1	3	2	4	1
2	1	2	4	3
3	4	1	2	3

You might also want to apply one of four different treatments to each cell of this plan (for example, applying different amounts of each drug). The following additional statements create the output shown in Figure 93.2:

```
   factors Replicate=3 ordered Drug=4;
   treatments Treatment=4;
run;
```

Figure 93.2 Using the TREATMENTS Statement

The PLAN Procedure

Plot Factors			
Factor	**Select**	**Levels**	**Order**
Replicate	3	3	Ordered
Drug	4	4	Random

Treatment Factors			
Factor	**Select**	**Levels**	**Order**
Treatment	4	4	Random

Figure 93.2 *continued*

Replicate	Drug				Treatment			
1	3	1	2	4	2	1	3	4
2	4	3	2	1	4	1	2	3
3	3	2	4	1	1	4	2	3

Randomly Assigning Subjects to Treatments

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/planin2.sas.)

You can use the PLAN procedure to design a completely randomized design. Suppose you have 12 experimental units, and you want to assign one of two treatments to each unit. Use a DATA step to store the unrandomized design in a SAS data set, and then call PROC PLAN to randomize it by specifying one factor with the default type of RANDOM, having 12 levels. The following statements produce Figure 93.3 and Figure 93.4:

```
title 'Completely Randomized Design';
/* The unrandomized design */

data Unrandomized;
   do Unit=1 to 12;
      if (Unit <= 6) then Treatment=1;
      else                Treatment=2;
      output;
   end;
run;

/* Randomize the design */

proc plan seed=27371;
   factors Unit=12;
   output data=Unrandomized out=Randomized;
run;

proc sort data=Randomized;
   by Unit;
run;
proc print;
run;
```

Figure 93.3 shows that the 12 levels of the unit factor have been randomly reordered and then lists the new ordering.

Figure 93.3 A Completely Randomized Design for Two Treatments

Completely Randomized Design

The PLAN Procedure

Factor	Select	Levels	Order
Unit	12	12	Random

Unit										
8	5	1	4	6	2	12	7	3	9	10	11

After the data set is sorted by the unit variable, the randomized design is displayed (Figure 93.4).

Figure 93.4 A Completely Randomized Design for Two Treatments

Completely Randomized Design

Obs	Unit	Treatment
1	1	1
2	2	1
3	3	2
4	4	1
5	5	1
6	6	1
7	7	2
8	8	1
9	9	2
10	10	2
11	11	2
12	12	2

You can also generate the plan by using a TREATMENTS statement instead of a DATA step. The following statements generate the same plan.

```
proc plan seed=27371;
   factors Unit=12;
   treatments Treatment=12 cyclic (1 1 1 1 1 1 2 2 2 2 2 2);
   output out=Randomized;
run;
```

Syntax: PLAN Procedure

The following statements are available in the PLAN procedure:

PROC PLAN < *options* > **;**
  FACTORS *factor-selections* < / **NOPRINT** > **;**
  OUTPUT OUT=*SAS-data-set* < *factor-value-settings* > **;**
  TREATMENTS *factor-selections* **;**

To use PROC PLAN, you need to specify the PROC PLAN statement and at least one FACTORS statement before the first RUN statement. The TREATMENTS statement, OUTPUT statement, and additional FACTORS statements can appear either before the first RUN statement or after it.

The rest of this section gives detailed syntax information for each of the statements, beginning with the PROC PLAN statement. The remaining statements are described in alphabetical order.

You can use PROC PLAN interactively by specifying multiple groups of statements, separated by RUN statements. For details, see the section "Using PROC PLAN Interactively" on page 7753.

PROC PLAN Statement

PROC PLAN < *options* > **;**

The PROC PLAN statement invokes the PLAN procedure. Optionally, you can also specify a random number seed or a default method for selecting levels of factors. By default, the procedure uses a random number seed generated from reading the time of day from the computer's clock and randomly selects levels of factors. These defaults can be modified with the SEED= and ORDERED options, respectively. Unlike many SAS/STAT procedures, the PLAN procedure does not have a DATA= option in the PROC PLAN statement; in this procedure, both the input and output data sets are specified in the OUTPUT statement.

Table 93.1 summarizes the options available in the PROC PLAN statement.

Table 93.1 PROC PLAN Statement Options

Option	Description
ORDERED	Selects the levels of the factor as the integers $1, 2, \ldots, m$, in order
SEED=	Specifies an integer used to start the pseudo-random number generator

You can specify the following options in the PROC PLAN statement:

ORDERED
selects the levels of the factor as the integers $1, 2, \ldots, m$, in order. For more detail, see "*Selection-Types*" on page 7748 and "Specifying Factor Structures" on page 7755.

SEED=*number*

specifies an integer used to start the pseudo-random number generator for selecting factor levels randomly. If you do not specify a seed, or if you specify a value less than or equal to zero, then by default the seed is generated by reading the time of day from the computer's clock.

FACTORS Statement

FACTORS *factor-selections* < / **NOPRINT** > **;**

The FACTORS statement specifies the factors of the plan and generates the plan. Taken together, the *factor-selections* specify the plan to be generated; you can specify more than one *factor-selection* request in a FACTORS statement. The form of a *factor-selection* is

name **=** *m* < **OF** *n* > < *selection-type* > < **NROW** > **;**

where

name is a valid SAS name. This gives the name of a factor in the design.

m is a positive integer that gives the number of values (factor levels) to be selected. If *n* is specified, the value of *m* must be less than or equal to *n*.

n is a positive integer that gives the number of values (factor levels) to be selected from.

selection-type specifies one of five methods for selecting *m* values. Possible values are COMB, CYCLIC, ORDERED, PERM, and RANDOM. The CYCLIC *selection-type* has additional optional specifications that enable you to specify an initial block of numbers to be cyclically permuted and an increment used to permute the numbers. By default, the *selection-type* is RANDOM, unless you use the ORDERED option in the PROC PLAN statement. In this case, the default *selection-type* is ORDERED. For details, see the following section, "*Selection-Types*"; for examples, see the section "Syntax Examples" on page 7748.

NROW sets the number of values (factor levels) to be equal to the number of all possible values (factor levels) of the next factor. This keyword is available only when the *selection-type* for the next factor is either COMB or PERM. For examples, see the section "Syntax Examples" on page 7748.

The following option can appear in the FACTORS statement after the slash:

NOPRINT

suppresses the display of the plan. This is particularly useful when you require only an output data set. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

Selection-Types

PROC PLAN interprets *selection-type* as follows:

RANDOM selects the m levels of the factor randomly without replacement from the integers $1, 2, \ldots, n$. Or, if n is not specified, RANDOM selects levels by randomly ordering the integers $1, 2, \ldots, m$.

ORDERED selects the levels of the factor as the integers $1, 2, \ldots, m$, in that order.

PERM selects the m levels of the factor as a permutation of the integers $1, \ldots m$ according to an algorithm that cycles through all $m!$ permutations. The permutations are produced in a sorted standard order; see Example 93.6.

COMB selects the m levels of the factor as a combination of the integers $1, \ldots, n$ taken m at a time, according to an algorithm that cycles through all $n!/(m!(n-m)!)$ combinations. The combinations are produced in a sorted standard order; see Example 93.6.

CYCLIC < (*initial-block*) > < *increment* > selects the levels of the factor by cyclically permuting the integers $1, 2, \ldots, n$. Wrapping occurs at m if n is not specified, and at n if n is specified. Additional optional specifications are as follows.

With the *selection-type* CYCLIC, you can optionally specify an *initial-block* and an *increment*. The *initial-block* must be specified within parentheses, and it specifies the block of numbers to permute. The first permutation is the block you specify, the second is the block permuted by 1 (or by the *increment* you specify), and so on. By default, the *initial-block* is the integers $1, 2, \ldots, m$. If you specify an *initial-block*, it must have m values. Values specified in the *initial-block* do not have to be given in increasing order.

The *increment* specifies the increment by which to permute the block of numbers. By default, the *increment* is 1.

Syntax Examples

This section gives some simple syntax examples. For more complex examples and details on how to generate various designs, see "Specifying Factor Structures" on page 7755. The examples in this section assume that you use the default random selection method and do not use the ORDERED option in the PROC PLAN statement.

The following specification generates a random permutation of the numbers 1, 2, 3, 4, and 5.

```
factors A=5;
```

The following specification generates a random permutation of five of the integers from 1 to 8, selected without replacement.

```
factors A=5 of 8;
```

Adding the ORDERED *selection-type* to the two previous specifications generates an ordered list of the integers 1 to 5. The following specification cyclically permutes the integers 1, 2, 3, and 4.

```
factors A=4 cyclic;
```

Since this simple request generates only one permutation of the numbers, the procedure generates an ordered list of the integers 1 to 4. The following specification cyclically permutes the integers 5 to 8.

```
factors A=4 of 8 cyclic (5 6 7 8);
```

In this case, because only one permutation is performed, the procedure generates an ordered list of the integers 5 to 8. The following specification produces an ordered list for A, the with values 1 and 2:

```
factors A=2 ordered B=4 of 8 cyclic (5 6 7 8) 2;
```

The associated factor levels for B are 5, 6, 7, and 8 for level 1 of A and 7, 8, 1, and 2 for level 2 of A.

Handling More Than One *Factor-Selection*

For cases with more than one *factor-selection* in the same FACTORS statement, PROC PLAN constructs the design as follows:

1. PROC PLAN first generates levels for the first *factor-selection*. These levels are permutations of integers (1, 2, and so on) appropriate for the selection type chosen. If you do not specify a selection type, PROC PLAN uses the default (RANDOM); if you specify the ORDERED option in the PROC PLAN statement, the procedure uses ORDERED as the default selection type.

2. For every integer generated for the first *factor-selection*, levels are generated for the second *factor-selection*. These levels are generated according to the specifications following the second equal sign.

3. This process is repeated until levels for all *factor-selections* have been generated.

The following statements give an example of generating a design with two random factors:

```
proc plan seed=27371;
   factors One=4 Two=3;
run;
```

The procedure first generates a random permutation of the integers 1 to 4 and then, for each integer, generates a random permutation of the integers 1 to 3. You can think of factor Two as being nested within factor One, where the levels of factor One are to be randomly assigned to four units.

As another example, six random permutations of the numbers 1, 2, and 3 can be generated when you specify the following statements:

```
proc plan seed=27371;
   factors a=6 ordered b=3;
run;
```

In the following two examples, the **NROW** keyword computes the number of levels for factor A by using the number of all possible values for factor B.

The following specification generates all permutations of the integers 1, 2, 3, and 4:

```
factors A=nrow B=4 perm;
```

The following specification generates all combinations of the integers 1, 2, 3, 4, and 5, taken three at a time:

```
factors A=nrow B=5 of 3 comb;
```

OUTPUT Statement

OUTPUT OUT=*SAS-data-set* < **DATA=***SAS-data-set* > < *factor-value-settings* > **;**

The OUTPUT statement applies only to the last plan generated. If you use PROC PLAN interactively, the OUTPUT statement for a given plan must be immediately preceded by the FACTORS statement (and the TREATMENTS statement, if appropriate) for the plan.

See "Output Data Sets" on page 7753 for more information about how output data sets are constructed.

You can specify the following options in the OUTPUT statement:

OUT=*SAS-data-set*

DATA=*SAS-data-set*

You can use the OUTPUT statement both to output the last plan generated and to use the last plan generated to randomize another SAS data set.

When you specify only the OUT= option in the OUTPUT statement, PROC PLAN saves the last plan generated to the specified data set. The output data set contains one variable for each factor in the plan and one observation for each cell in the plan. The value of a variable in a given observation is the level of the corresponding factor for that cell. The OUT= option is required.

When you specify both the DATA= and OUT= options in the OUTPUT statement, then PROC PLAN uses the last plan generated to randomize the input data set (DATA=), saving the results to the output data set (OUT=). The output data set has the same form as the input data set but has modified values for the variables that correspond to factors (see the section "Output Data Sets" on page 7753 for details). Values for variables not corresponding to factors are transferred without change.

factor-value-settings

specify the values input or output for the factors in the design. The form for *factor-value-settings* is different when only an OUT= data set is specified and when both OUT= and DATA= data sets are specified.

Both forms are discussed in the following section.

Factor-Value-Settings with Only an OUT= Data Set

When you specify only an OUT= data set, the form for each *factor-value-setting* specification is one of the following:

factor-name < **NVALS=***list-of-n-numbers* > < **ORDERED** | **RANDOM** > **;**

or

factor-name < **CVALS=***list-of-n-strings* > < **ORDERED** | **RANDOM** > **;**

where

factor-name is a factor in the last FACTORS statement preceding the OUTPUT statement.

NVALS= lists n numeric values for the factor. By default, the procedure uses NVALS=(1 2 3 $\cdots n$).

CVALS= lists n character strings for the factor. Each string can have up to 40 characters, and each string must be enclosed in quotes. **CAUTION:** When you use the CVALS= option, the variable created in the output data set has a length equal to the length of the longest string given as a value; shorter strings are padded with trailing blanks. For example, the values output for the first level of a two-level factor with the following two different specifications are not the same.

```
CVALS=('String 1' "String 2")

CVALS=('String 1' "A longer string")
```

The value output with the second specification is 'String 1' followed by seven blanks. In order to match two such values (for example, when merging two plans), you must use the TRIM function in the DATA step (see *SAS Functions and CALL Routines: Reference*).

ORDERED | RANDOM specifies how values (those given with the NVALS= or CVALS= option, or the default values) are associated with the levels of a factor (the integers $1, 2, \ldots, n$). The default association type is ORDERED, for which the first value specified is output for a factor level setting of 1, the second value specified is output for a level of 2, and so on. You can also specify an association type of RANDOM, for which the levels are associated with the values in a random order. Specifying RANDOM is useful for randomizing crossed experiments (see the section "Randomizing Designs" on page 7757).

The following statements give an example of using the OUTPUT statement with only an OUT= data set and with both the NVALS= and CVALS= specifications.

```
proc plan seed=27371;
   factors a=6 ordered b=3;
   output out=design a nvals=(10 to 60 by 10)
                     b cvals=('HSX' 'SB2' 'DNY');
run;
```

The DESIGN data set contains two variables, a and b. The values of the variable a are 10 when factor a equals 1, 20 when factor a equals 2, and so on. Values of the variable b are 'HSX' when factor b equals 1, 'SB2' when factor b equals 2, and 'DNY' when factor b equals 3.

Factor-Value-Settings with OUT= and DATA= Data Sets

If you specify an input data set with DATA=, then PROC PLAN assumes that each factor in the last plan generated corresponds to a variable in the input set. If the variable name is different from the name of the factor to which it corresponds, the two can be associated in the values specification by

input-variable-name **=** *factor-name* **;**

Then, the NVALS= or CVALS= specification can be used. The values given by NVALS= or CVALS= specify the input values as well as the output values for the corresponding variable.

Since the procedure assumes that the collection of input factor values constitutes a plan position description (see the section "Output Data Sets" on page 7753), the values must correspond to integers less than or equal

to *m*, the number of values selected for the associated factor. If any input values do not correspond, then the collection does not define a plan position, and the corresponding observation is output without changing the values of any of the factor variables.

The following statements demonstrate the use of *factor-value-settings*. The input SAS data set a contains variables Day and Hour, which are renamed Block and Plot, respectively.

```
proc plan seed=27371;
   factors Block=7 Plot=6;
   output data=a out=b
      Day = Block  cvals=('Mon' 'Tue' 'Wed' 'Thu'
                          'Fri' 'Sat' 'Sun'      )
      Hour = Plot;
run;
```

For another example of using both a DATA= and OUT= data set, see the section "Randomly Assigning Subjects to Treatments" on page 7744.

TREATMENTS Statement

TREATMENTS *factor-selections* **;**

The TREATMENTS statement specifies the treatments of the plan to generate, but it does not generate a plan. If you supply several FACTORS and TREATMENTS statements before the first RUN statement, the procedure uses only the last TREATMENTS specification and applies it to the plans that are generated by each of the FACTORS statements. The TREATMENTS statement must follow a FACTORS statement. The TREATMENTS statement has the same syntax as the FACTORS statement. The individual *factor-selections* also have the same form as in the FACTORS statement:

name **=** *m* < **OF** *n* > < *selection-type* > **;**

The procedure generates each treatment simultaneously with the lowest (that is, the most nested) factor in the last FACTORS statement. The *m* value for each treatment must be at least as large as the *m* for the most nested factor.

The following statements use both a FACTORS and a TREATMENTS statement. First the FACTORS statement sets up the rows and columns of a 3×3 square (factors r and c). Then, the TREATMENTS statement augments the square with two cyclic treatments. The resulting design is a 3×3 Graeco-Latin square, a type of design useful in main-effects factorial experiments.

```
proc plan;
   factors r=3 ordered c=3 ordered;
   treatments a=3 cyclic
              b=3 cyclic 2;
run;
```

The resulting Graeco-Latin square design is shown in Figure 93.5. Notice how the values of r and c are ordered (1, 2, 3) as requested.

Figure 93.5 A 3×3 Graeco-Latin Square

The PLAN Procedure

r	c			a			b	
1	1	2	3	1	2	3	1	2	3
2	1	2	3	2	3	1	3	1	2
3	1	2	3	3	1	2	2	3	1

Details: PLAN Procedure

Using PROC PLAN Interactively

After specifying a design with a FACTORS statement and running PROC PLAN with a RUN statement, you can generate additional plans and output data sets without invoking PROC PLAN again.

In PROC PLAN, all statements can be used interactively. You can execute statements singly or in groups by following the single statement or group of statements with a RUN statement.

If you use PROC PLAN interactively, you can end the procedure with a DATA step, another PROC step, an ENDSAS statement, or a QUIT statement. The syntax of the QUIT statement is

```
quit;
```

When you use PROC PLAN interactively, additional RUN statements do not end the procedure but tell PROC PLAN to execute additional statements.

Output Data Sets

To understand how PROC PLAN creates output data sets, you need to look at how the procedure represents a plan. A plan is a list of values for all the factors, the values being chosen according to the *factor-selection* requests you specify. For example, consider the plan produced by the following statements:

```
proc plan seed=12345;
   factors a=3 b=2;
run;
```

The plan as displayed by PROC PLAN is shown in Figure 93.6.

Figure 93.6 A Simple Plan

The PLAN Procedure

Factor	Select	Levels	Order
a	3	3	Random
b	2	2	Random

Figure 93.6 *continued*

a	b	
2	2	1
1	1	2
3	2	1

The first cell of the plan has a=2 and b=2, the second has a=2 and b=1, the third has a=1 and b=1, and so on. If you output the plan to a data set with the OUTPUT statement, by default the output data set contains a numeric variable with that factor's name; the values of this numeric variable are the numbers of the successive levels selected for the factor in the plan. For example, the following statements produce Figure 93.7.

```
proc plan seed=12345;
   factors a=3 b=2;
   output out=out;
run;
proc print data=out;
run;
```

Figure 93.7 Output Data Set from Simple Plan

Obs	a	b
1	2	2
2	2	1
3	1	1
4	1	2
5	3	2
6	3	1

Alternatively, you can specify the values that are output for a factor with the CVALS= or NVALS= option. Also, you can specify that the internal values be associated with the output values in random order with the RANDOM option. See the section "OUTPUT Statement" on page 7750.

If you also specify an input data set (DATA=), each factor is associated with a variable in the DATA= data set. This occurs either implicitly by the factor and variable having the same name or explicitly as described in the specifications for the OUTPUT statement. In this case, the values of the variables corresponding to the factors are first read and then interpreted as describing the position of a cell in the plan. Then the respective values taken by the factors at that position are assigned to the variables in the OUT= data set. For example, consider the data set defined by the following statements.

```
data in;
   input a b;
   datalines;
1 1
2 1
3 1
;
```

Suppose you specify this data set as an input data set for the OUTPUT statement.

```
proc plan seed=12345;
   factors a=3 b=2;
   output out=out data=in;
run;
proc print data=out;
run;
```

PROC PLAN interprets the first observation as referring to the cell in the first row and column of the plan, since a=1 and b=1; likewise, the second observation is interpreted as the cell in the second row and first column, and the third observation as the cell in the third row and first column. In the output data set, a and b have the values they have in the plan at these positions, as shown in Figure 93.8.

Figure 93.8 Output Form of Input Data Set from Simple Plan

Obs	a	b
1	2	2
2	1	1
3	3	2

When the factors are random, this has the effect of randomizing the input data set in the same manner as the plan produced (see the sections "Randomizing Designs" on page 7757 and "Randomly Assigning Subjects to Treatments" on page 7744).

Specifying Factor Structures

By appropriately combining features of the PLAN procedure, you can construct an extensive set of designs. The basic tools are the *factor-selections*, which are used in the FACTORS and TREATMENTS statements. Table 93.2 summarizes how the procedure interprets various *factor-selections* (assuming that the ORDERED option is not specified in the PROC PLAN statement).

Table 93.2 *Factor-Selection* Interpretation

Form of Request	Interpretation	Example	Results
name=*m*	Produce a random permutation of the integers $1, 2, \ldots, m$	`t=15`	Lists a random ordering of the numbers $1, 2, \ldots, 15$
name=*m* cyclic	Cyclically permute the integers $1, 2, \ldots, m$	`t=5 cyclic`	Selects the integers 1 to 5. On the next iteration, selects 2,3,4,5,1; then 3,4,5,1,2; and so on.
name=*m* of *n*	Choose a random sample of *m* integers (without replacement) from the set of integers $1, 2, \ldots, n$	`t=5 of 15`	Lists a random selection of 5 numbers from 1 to 15. First, the procedure selects 5 numbers and then arranges them in random order.
name=*m* of *n* ordered	Has the same effect as *name*=*m* ordered	`t=5 of 15 ordered`	Lists the integers 1 to 5 in increasing order (same as t=5 ordered)

Table 93.2 *continued*

Form of Request	Interpretation	Example	Results
name=*m* of *n* cyclic	Permute *m* of the *n* integers	`t=5 of 30 cyclic`	Selects the integers 1 to 5. On the next iteration, selects 2,3,4,5,6; then 3,4,5,6,7; and so on. The 30th iteration produces 30,1,2,3,4; the 31st iteration produces 1,2,3,4,5; and so on.
name=*m* perm	Produce a list of all permutations of *m* integers	`t=5 perm`	Lists the integers 1,2,3,4,5 on the first iteration; on the second lists 1,2,3,5,4; on the 119th iteration lists 5,4,3,1,2; and on the last (120th) lists 5,4,3,2,1.
name=*m* of *n* comb	Choose combinations of *m* integers from *n* integers	`t=3 of 5 comb`	Lists all combinations of 5 choose 3 integers. The first iteration is 1,2,3; the second is 1,2,4; the third is 1,2,5; and so on until the last iteration 3,4,5.
name=*m* of *n* cyclic (*initial-block*)	Permute *m* of the *n* integers, starting with the values specified in the *initial-block*	`t=4 of 30 cyclic (2 10 15 18)`	Selects the integers 2,10,15,18. On the next iteration, selects 3,11,16,19; then 4,12,17,20; and so on. The thirteenth iteration is 14,22,27,30; the fourteenth iteration is 15,23,28,1; and so on.
name=*m* of *n* cyclic (*initial-block*) *increment*	Permute *m* of the *n* integers. Start with the values specified in the *initial-block*, then add the *increment* to each value.	`t=4 of 30 cyclic (2 10 15 18) 2`	Selects the integers 2,10,15,18. On the next iteration, selects 4,12,17,20; then 6,14,19,22; and so on. The wrap occurs at the eighth iteration. The eighth iteration is 16,24,29,2; and so on.

In Table 93.2, in order for more than one iteration to appear in the plan, another *name*=*j* factor selection (with $j > 1$) must precede the example factor selection. For example, the following statements produce six of the iterations described in the last entry of Table 93.2.

```
proc plan;
   factors c=6 ordered t=4 of 30 cyclic (2 10 15 18) 2;
run;
```

The following statements create a randomized complete block design and output the design to a data set.

```
proc plan ordered seed=78390;
   factors blocks=3 cell=5;
   treatments t=5 random;
   output out=rcdb;
run;
```

Randomizing Designs

In many situations, proper randomization is crucial for the validity of any conclusions to be drawn from an experiment. Randomization is used both to neutralize the effect of any unknown systematic biases that might be involved in the design and to provide a basis for the assumptions underlying the analysis.

You can use PROC PLAN to randomize an already existing design: one produced by a previous call to PROC PLAN, perhaps, or a more specialized design taken from a standard reference such as Cochran and Cox (1957). The method is to specify the appropriate block structure in the FACTORS statement and then to specify the data set where the design is stored with the DATA= option in the OUTPUT statement. For an illustration of this method, see the section "Randomly Assigning Subjects to Treatments" on page 7744).

Two sorts of randomization are provided for, corresponding to the RANDOM factor selection and association types in the FACTORS and OUTPUT statements, respectively. Designs in which factors are completely nested (for example, block designs) should be randomized by specifying that the selection type of each factor is RANDOM in the FACTORS statement, which is the default (see Example 93.3). On the other hand, if the factors are crossed (for example, row-and-column designs), they should be randomized by one random reassignment of their values for the whole design. To do this, specify that the association type of each factor is RANDOM in the OUTPUT statement (see Example 93.4).

Displayed Output

The PLAN procedure displays the following output:

- the *m* value for each factor, which is the number of values to be selected
- the *n* value for each factor, which is the number of values to be selected from
- the selection type for each factor, as specified in the FACTORS statement
- the initial block and increment number for cyclic factors
- the *factor-value-selections* making up each plan

In addition, notes are written to the log that give the starting and ending values of the random number seed for each call to PROC PLAN.

ODS Table Names

PROC PLAN assigns a name to each table it creates. You can use these names to reference the table in the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 93.3 ODS Tables Produced by PROC PLAN

ODS Table Name	Description	Statements
FInfo	General factor information	FACTORS and no TREATMENTS
PFInfo	Plot factor information	FACTORS and TREATMENTS
Plan	Computed plan	Default
TFInfo	Treatment factor information	FACTORS and TREATMENTS

Examples: PLAN Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 93.1: A Split-Plot Design

(View the complete code for this example (planex1.sas) in the example repository.)

This plan is appropriate for a split-plot design with main plots forming a randomized complete block design. In this example, there are three blocks, four main plots per block, and two subplots per main plot. First, three random permutations (one for each of the blocks) of the integers 1, 2, 3, and 4 are produced. The four integers correspond to the four levels of the main plot factor a; the permutation determines how the levels of a are assigned to the main plots within a block. For each of these 12 numbers (four numbers per block for three blocks), a random permutation of the integers 1 and 2 is produced. Each two-integer permutation determines the assignment of the two levels of the subplot factor b within a main plot. The following statements produce Output 93.1.1:

```
title 'Split Plot Design';
proc plan seed=37277;
   factors Block=3 ordered a=4 b=2;
run;
```

Output 93.1.1 A Split-Plot Design

Split Plot Design

The PLAN Procedure

Factor	Select	Levels	Order
Block	3	3	Ordered
a	4	4	Random
b	2	2	Random

Block	a	b
1	4	2 1
	3	2 1
	1	2 1
	2	2 1
2	4	1 2
	3	1 2
	1	2 1
	2	1 2
3	4	2 1
	2	2 1
	3	2 1
	1	2 1

Example 93.2: A Hierarchical Design

(View the complete code for this example (planex2.sas) in the example repository.)

In this example, three plants are nested within four pots, which are nested within three houses. The FACTORS statement requests a random permutation of the numbers 1, 2, and 3 to choose Houses randomly. The second step requests a random permutation of the numbers 1, 2, 3, and 4 for each of those first three numbers to randomly assign Pots to Houses. Finally, the FACTORS statement requests a random permutation of 1, 2, and 3 for each of the 12 integers in the second set of permutations. This last step randomly assigns Plants to Pots. The following statements produce Output 93.2.1:

```
title 'Hierarchical Design';
proc plan seed=17431;
   factors Houses=3 Pots=4 Plants=3 / noprint;
   output out=nested;
run;

proc print data=nested;
run;
```

Output 93.2.1 A Hierarchical Design

Hierarchical Design

Obs	Houses	Pots	Plants
1	1	3	2
2	1	3	3
3	1	3	1
4	1	1	3
5	1	1	1
6	1	1	2
7	1	2	2
8	1	2	3
9	1	2	1
10	1	4	3
11	1	4	2
12	1	4	1
13	2	4	1
14	2	4	3
15	2	4	2
16	2	2	2
17	2	2	1
18	2	2	3
19	2	3	2
20	2	3	3
21	2	3	1
22	2	1	2
23	2	1	3
24	2	1	1
25	3	4	1
26	3	4	3
27	3	4	2
28	3	1	3
29	3	1	2
30	3	1	1
31	3	2	1
32	3	2	2
33	3	2	3
34	3	3	3
35	3	3	2
36	3	3	1

Example 93.3: An Incomplete Block Design

(View the complete code for this example (planex3.sas) in the example repository.)

Jarrett and Hall (1978) give an example of a generalized cyclic design with good efficiency characteristics. The design consists of two replicates of 52 treatments in 13 blocks of size 8. The following statements use the PLAN procedure to generate this design in an appropriately randomized form and store it in a SAS data set GCBD. Then the design is sorted and transposed to display in randomized order. The following statements produce Output 93.3.1 and Output 93.3.2:

```
title 'Generalized Cyclic Block Design';
proc plan seed=33373;
   factors Block=13 Plot=8;
   treatments Treatment=8 of 52 cyclic (1 2 3 4 32 43 46 49) 4;
   output out=GCBD;
quit;

proc sort data=GCBD out=GCBD;
   by Block Plot;
run;
proc transpose data= GCBD(rename=(Plot=_NAME_))
               out =tGCBD(drop=_NAME_);
   by Block;
   var Treatment;
run;
proc print data=tGCBD noobs;
run;
```

Output 93.3.1 A Generalized Cyclic Block Design

Generalized Cyclic Block Design

The PLAN Procedure

Plot Factors			
Factor	Select	Levels	Order
Block	13	13	Random
Plot	8	8	Random

Treatment Factors				
Factor	Select	Levels	Order	Initial Block / Increment
Treatment	8	52	Cyclic	(1 2 3 4 32 43 46 49) / 4

Output 93.3.1 *continued*

Block	Plot								Treatment							
10	7	4	8	1	2	3	5	6	1	2	3	4	32	43	46	49
8	1	2	4	3	8	6	5	7	5	6	7	8	36	47	50	1
9	2	5	4	7	3	1	8	6	9	10	11	12	40	51	2	5
6	4	2	6	8	3	7	1	5	13	14	15	16	44	3	6	9
7	4	7	6	3	1	2	8	5	17	18	19	20	48	7	10	13
4	4	8	1	5	3	6	7	2	21	22	23	24	52	11	14	17
2	6	2	3	8	7	5	1	4	25	26	27	28	4	15	18	21
3	6	2	3	1	7	4	5	8	29	30	31	32	8	19	22	25
1	1	2	7	8	5	6	3	4	33	34	35	36	12	23	26	29
5	5	7	6	8	4	3	1	2	37	38	39	40	16	27	30	33
12	5	8	1	4	7	3	6	2	41	42	43	44	20	31	34	37
13	3	5	1	8	4	2	6	7	45	46	47	48	24	35	38	41
11	4	1	5	2	3	8	6	7	49	50	51	52	28	39	42	45

Output 93.3.2 A Generalized Cyclic Block Design

Generalized Cyclic Block Design

Block	_1	_2	_3	_4	_5	_6	_7	_8
1	33	34	26	29	12	23	35	36
2	18	26	27	21	15	25	4	28
3	32	30	31	19	22	29	8	25
4	23	17	52	21	24	11	14	22
5	30	33	27	16	37	39	38	40
6	6	14	44	13	9	15	3	16
7	48	7	20	17	13	19	18	10
8	5	6	8	7	50	47	1	36
9	51	9	40	11	10	5	12	2
10	4	32	43	2	46	49	1	3
11	50	52	28	49	51	42	45	39
12	43	37	31	44	41	34	20	42
13	47	35	45	24	46	38	41	48

Example 93.4: A Latin Square Design

(View the complete code for this example (planex4.sas) in the example repository.)

All of the preceding examples involve designs with completely nested block structures, for which PROC PLAN was especially designed. However, by appropriate coordination of its facilities, PROC PLAN can accommodate a much wider class of designs. A Latin square design is based on experimental units that have a row-and-column block structure. The following statements use the CYCLIC option in the TREATMENTS statement to generate a simple 4×4 Latin square. The RANDOM option in the OUTPUT statement randomizes the generated Latin square by randomly permuting the row, column, and treatment values independently. This example also uses *factor-value-settings* in the OUTPUT statement.

```
title 'Latin Square Design';
proc plan seed=37430;
   factors Row=4 ordered Col=4 ordered / noprint;
   treatments Tmt=4 cyclic;
   output out=LatinSquare
          Row cvals=('Day 1' 'Day 2' 'Day 3' 'Day 4') random
          Col cvals=('Lab 1' 'Lab 2' 'Lab 3' 'Lab 4') random
          Tmt nvals=(      0     100     250     450) random;
quit;

proc sort data=LatinSquare out=LatinSquare;
   by Row Col;
run;
proc transpose data= LatinSquare(rename=(Col=_NAME_))
               out =tLatinSquare(drop=_NAME_);
   by Row;
   var Tmt;
run;
proc print data=tLatinSquare noobs;
run;
```

The preceding statements produce Output 93.4.1.

Output 93.4.1 A Randomized Latin Square Design

Latin Square Design

Row	Lab_1	Lab_2	Lab_3	Lab_4
Day 1	0	250	100	450
Day 2	250	450	0	100
Day 3	100	0	450	250
Day 4	450	100	250	0

You can use the PLAN procedure to randomize Latin squares from any transformation sets. See Kempthorne (1952) for definitions of transformation sets. In particular, the following DATA step and PROC PLAN statements demonstrate how to randomize a Latin square from the second transformation set defined in Kempthorne (1952). The following DATA step creates a 4×4 Latin square from the transformation set 2:

```
data Unrandomized;
   do Row = 1 to 4;
      do Col = 1 to 4;
         input Tmt @@;
         output;
      end;
   end;
   datalines;
1 2 3 4
2 1 4 3
3 4 1 2
4 3 2 1
;
```

The following PROC PLAN statements permute the rows and columns and randomly assign treatment levels.

```
proc plan seed=37430;
   factors Row = 4;
   output data=  Unrandomized  out=Randomized1;
run;
   factors Col = 4;
   output data=  Randomized1   out=Randomized2;
run;
   factors Tmt = 4;
   output data=  Randomized2   out=Randomized3;
run;

proc sort data=Randomized3;
   by Row Col;
run;
proc transpose data= Randomized3 out =tLatinSquare2(drop=_NAME_);
   by Row;
   var Tmt;
run;
proc print data=tLatinSquare2 noobs;
run;
```

Output 93.4.2 A Randomized Latin Square from a Different Transformation Set

The PLAN Procedure

Factor	Select	Levels	Order
Row	4	4	Random

Row
1 3 2 4

Factor	Select	Levels	Order
Col	4	4	Random

Col
3 2 4 1

Output 93.4.2 *continued*

Factor	Select	Levels	Order
Tmt	4	4	Random

Tmt
4 3 2 1

Row	COL1	COL2	COL3	COL4
1	1	3	4	2
2	3	1	2	4
3	2	4	3	1
4	4	2	1	3

Example 93.5: A Generalized Cyclic Incomplete Block Design

(View the complete code for this example (planex5.sas) in the example repository.)

The following statements depict how to create an appropriately randomized generalized cyclic incomplete block design for v treatments (given by the value of t) in b blocks (given by the value of b) of size k (with values of p indexing the cells within a block) with initial block $(e_1\ e_2\ \cdots\ e_k)$ and increment number i.

```
factors b=b p=k ;
treatments t=k of v cyclic (e1 e2 ··· ek ) i ;
```

For example, the specification

```
proc plan seed=37430;
   factors b=10 p=4;
   treatments t=4 of 30 cyclic (1 3 4 26) 2;
run;
```

generates the generalized cyclic incomplete block design given in Example 1 of Jarrett and Hall (1978), which is given by the rows and columns of the plan associated with the treatment factor t in Output 93.5.1.

Output 93.5.1 A Generalized Cyclic Incomplete Block Design

The PLAN Procedure

Plot Factors			
Factor	Select	Levels	Order
b	10	10	Random
p	4	4	Random

Treatment Factors				
Factor	Select	Levels	Order	Initial Block / Increment
t	4	30	Cyclic	(1 3 4 26) / 2

Output 93.5.1 *continued*

b	p				t			
2	2	3	1	4	1	3	4	26
1	3	2	4	1	3	5	6	28
3	2	3	4	1	5	7	8	30
10	4	2	3	1	7	9	10	2
9	4	1	2	3	9	11	12	4
4	1	3	2	4	11	13	14	6
5	1	2	4	3	13	15	16	8
8	3	2	4	1	15	17	18	10
7	2	4	1	3	17	19	20	12
6	2	1	4	3	19	21	22	14

Example 93.6: Permutations and Combinations

(View the complete code for this example (planex6.sas) in the example repository.)

Occasionally, you might need to generate all possible permutations of *n* things, or all possible combinations of *n* things taken *m* at a time.

For example, suppose you are planning an experiment in cognitive psychology where you want to present four successive stimuli to each subject. You want to observe each permutation of the four stimuli. The following statements use PROC PLAN to create a data set containing all possible permutations of four numbers in random order.

```
title 'All Permutations of 1,2,3,4';
proc plan seed=60359;
   factors    Subject  = 24
              Order    = 4  ordered;
   treatments Stimulus = 4  perm;
   output out=Psych;
run;

proc sort data=Psych out=Psych;
   by Subject Order;
run;
proc transpose data= Psych(rename=(Order=_NAME_))
               out =tPsych(drop=_NAME_);
   by Subject;
   var Stimulus;
run;
proc print data=tPsych noobs;
run;
```

The variable Subject is set at 24 levels because there are $4! = 24$ total permutations to be listed. If Subject > 24, the list repeats. Output 93.6.1 displays the PROC PLAN output. Note that the variable Subject is listed in random order.

Output 93.6.1 List of Permutations

All Permutations of 1,2,3,4

The PLAN Procedure

Plot Factors			
Factor	**Select**	**Levels**	**Order**
Subject	24	24	Random
Order	4	4	Ordered

Treatment Factors			
Factor	**Select**	**Levels**	**Order**
Stimulus	4	4	Perm

Subject	Order				Stimulus			
4	1	2	3	4	1	2	3	4
15	1	2	3	4	1	2	4	3
24	1	2	3	4	1	3	2	4
1	1	2	3	4	1	3	4	2
5	1	2	3	4	1	4	2	3
17	1	2	3	4	1	4	3	2
19	1	2	3	4	2	1	3	4
14	1	2	3	4	2	1	4	3
6	1	2	3	4	2	3	1	4
23	1	2	3	4	2	3	4	1
8	1	2	3	4	2	4	1	3
2	1	2	3	4	2	4	3	1
13	1	2	3	4	3	1	2	4
16	1	2	3	4	3	1	4	2
12	1	2	3	4	3	2	1	4
18	1	2	3	4	3	2	4	1
21	1	2	3	4	3	4	1	2
9	1	2	3	4	3	4	2	1
22	1	2	3	4	4	1	2	3
10	1	2	3	4	4	1	3	2
7	1	2	3	4	4	2	1	3
11	1	2	3	4	4	2	3	1
3	1	2	3	4	4	3	1	2
20	1	2	3	4	4	3	2	1

The output data set Psych contains 96 observations of the 3 variables (Subject, Order, and Stimulus). Sorting the output data set by Subject and by Order within Subject results in all possible permutations of Stimulus in random order. PROC TABULATE displays these permutations in Output 93.6.2.

Output 93.6.2 Randomized Permutations

All Permutations of 1,2,3,4

Subject	_1	_2	_3	_4
1	1	3	4	2
2	2	4	3	1
3	4	3	1	2
4	1	2	3	4
5	1	4	2	3
6	2	3	1	4
7	4	2	1	3
8	2	4	1	3
9	3	4	2	1
10	4	1	3	2
11	4	2	3	1
12	3	2	1	4
13	3	1	2	4
14	2	1	4	3
15	1	2	4	3
16	3	1	4	2
17	1	4	3	2
18	3	2	4	1
19	2	1	3	4
20	4	3	2	1
21	3	4	1	2
22	4	1	2	3
23	2	3	4	1
24	1	3	2	4

As another example, suppose you have six alternative treatments, any four of which can occur together in a block (in no particular order). The following statements use PROC PLAN to create a data set containing all possible combinations of six numbers taken four at a time. In this case, you use ODS to create the data set.

```
title 'All Combinations of (6 Choose 4) Integers';
proc plan;
   factors Block=15 ordered
           Treat= 4 of 6 comb;
   ods output Plan=Combinations;
run;

proc print data=Combinations noobs;
run;
```

The variable Block has 15 levels since there are a total of $6!/(4!2!) = 15$ combinations of four integers chosen from six integers. The data set formed by ODS from the displayed plan has one row for each block, with the four values of Treat corresponding to four different variables, as shown in Output 93.6.3 and Output 93.6.4.

Output 93.6.3 List of Combinations

All Combinations of (6 Choose 4) Integers

The PLAN Procedure

Factor	Select	Levels	Order
Block	15	15	Ordered
Treat	4	6	Comb

Block	Treat			
1	1	2	3	4
2	1	2	3	5
3	1	2	3	6
4	1	2	4	5
5	1	2	4	6
6	1	2	5	6
7	1	3	4	5
8	1	3	4	6
9	1	3	5	6
10	1	4	5	6
11	2	3	4	5
12	2	3	4	6
13	2	3	5	6
14	2	4	5	6
15	3	4	5	6

Output 93.6.4 Combinations Data Set Created by ODS

All Combinations of (6 Choose 4) Integers

Block	Treat1	Treat2	Treat3	Treat4
1	1	2	3	4
2	1	2	3	5
3	1	2	3	6
4	1	2	4	5
5	1	2	4	6
6	1	2	5	6
7	1	3	4	5
8	1	3	4	6
9	1	3	5	6
10	1	4	5	6
11	2	3	4	5
12	2	3	4	6
13	2	3	5	6
14	2	4	5	6
15	3	4	5	6

Example 93.7: Crossover Designs

(View the complete code for this example (planex7.sas) in the example repository.)

In *crossover* experiments, the same experimental units or subjects are given multiple treatments in sequence, and the model for the response at any one period includes an effect for the treatment applied in the previous period. A good design for a crossover experiment is therefore one that balances how often each treatment is preceded by each other treatment. Cox (1992) gives the following example of a balanced crossover experiment for paper production. In this experiment, the subjects are production runs of the mill, with the treatments being six different concentrations of pulp used in sequence. The following statements construct this design in a standard form:

```
proc plan;
   factors Run=6 ordered Period=6 ordered;
   treatments Treatment=6 cyclic (1 2 6 3 5 4);
run;
```

Output 93.7.1 shows the results of the preceding statements.

Output 93.7.1 Crossover Design for Six Treatments

The PLAN Procedure

Plot Factors			
Factor	Select	Levels	Order
Run	6	6	Ordered
Period	6	6	Ordered

Treatment Factors				
Factor	Select	Levels	Order	Initial Block / Increment
Treatment	6	6	Cyclic	(1 2 6 3 5 4) / 1

Run	Period						Treatment					
1	1	2	3	4	5	6	1	2	6	3	5	4
2	1	2	3	4	5	6	2	3	1	4	6	5
3	1	2	3	4	5	6	3	4	2	5	1	6
4	1	2	3	4	5	6	4	5	3	6	2	1
5	1	2	3	4	5	6	5	6	4	1	3	2
6	1	2	3	4	5	6	6	1	5	2	4	3

The construction method for this example is due to Williams (1949). The initial block for the treatment variable Treatment is defined as follows for $n = 6$:

$$(1 \quad 2 \;\; n \quad 3 \;\; n-1 \quad \ldots \quad n/2 \;\; n/2+2 \quad n/2)$$

This general form serves to generate a balanced crossover design for n treatments and n subjects in n periods when n is even. When n is odd, $2n$ subjects are required, with the following initial blocks, respectively for

odd and even n:

$$(1 \quad 2 \;\; n \quad 3 \;\; n-1 \quad \ldots \quad n/2+1 \;\; n/2)$$
$$(n/2 \;\; n/2+1 \quad \ldots \quad n-1 \;\; 3 \quad n \;\; 2 \quad 1)$$

In order to randomize Williams' crossover designs, the following statements randomly permute the subjects and treatments:

```
proc plan seed=136149876;
   factors Run=6 ordered Period=6 ordered / noprint;
   treatments Treatment=6 cyclic (1 2 6 3 5 4);
   output out=RandomizedDesign
      Run       random
      Treatment random
      ;
run;

/*
/ Relabel Period to obtain the same design as in Cox (1992).
/-----------------------------------------------------------------*/
data RandomizedDesign;
   set RandomizedDesign;
   Period = mod(Period+2,6)+1;
run;

proc sort data=RandomizedDesign;
   by Run Period;
run;
proc transpose data=RandomizedDesign out=tDesign(drop=_name_);
   by notsorted Run;
   var Treatment;
run;
data tDesign;
   set tDesign;
   rename COL1-COL6 = Period_1-Period_6;
run;
proc print data=tDesign noobs;
run;
```

In the preceding statements, Run and Treatment are randomized by using the RANDOM option in the OUTPUT statement, and new labels for Period are obtained in a subsequent DATA step. This Period relabeling is not necessary and might not be valid for Williams' designs in general; it is used in this example only to match results with those of Cox (1992). The SORT and TRANSPOSE steps then prepare the design to be printed in a standard form, shown in Output 93.7.2.

Output 93.7.2 Randomized Crossover Design

Run	Period_1	Period_2	Period_3	Period_4	Period_5	Period_6
1	3	6	2	5	4	1
2	5	3	4	6	1	2
3	1	4	5	2	6	3
4	2	1	6	4	3	5
5	6	5	1	3	2	4
6	4	2	3	1	5	6

The analysis of a crossover experiment requires for each observation a *carryover* variable whose values are the treatment in the preceding period. The following statements add such a variable to the randomized design constructed previously:

```
proc sort data=RandomizedDesign;
   by Run Period;
run;
data RandomizedDesign;
   set RandomizedDesign;
   by Run period;
   LagTreatment = lag(Treatment);
   if (first.Run) then LagTreatment = .;
run;

proc transpose data=RandomizedDesign out=tDesign(drop=_name_);
   by notsorted Run;
   var LagTreatment;
run;
data tDesign;
   set tDesign;
   rename COL1-COL6 = Period_1-Period_6;
run;
proc print data=tDesign noobs;
run;
```

Output 93.7.3 displays the values of the carryover variable for each run and period.

Output 93.7.3 Lag Treatment Effect in Crossover Design

Run	Period_1	Period_2	Period_3	Period_4	Period_5	Period_6
1	.	3	6	2	5	4
2	.	5	3	4	6	1
3	.	1	4	5	2	6
4	.	2	1	6	4	3
5	.	6	5	1	3	2
6	.	4	2	3	1	5

Of course, the carryover variable has no effect in the first period, which is why it is coded with a missing value in this case.

The LAG effect in the EFFECT statement in PROC ORTHOREG provides a convenient mechanism for incorporating the carryover effect into the analysis. The following statements first add the observed data

to the design to create the Mills data set. Then PROC ORTHOREG is invoked, and the carryover effect is defined as a lag effect with the relevant period and subject information specified. ODS is used to trim down the results to show only the parts that are usually of interest in crossover analysis. For more information about the EFFECTS statement in PROC ORTHOREG, see the section "EFFECT Statement" on page 7441 in Chapter 91, "The ORTHOREG Procedure."

```
data Responses;
   input Response @@;
   datalines;
56.7 53.8 54.4 54.4 58.9 54.5
58.5 60.2 61.3 54.4 59.1 59.8
55.7 60.7 56.7 59.9 56.6 59.6
57.3 57.7 55.2 58.1 60.2 60.2
53.7 57.1 59.2 58.9 58.9 59.6
58.1 55.7 58.9 56.6 59.6 57.5
;
data Mills;
   merge RandomizedDesign Responses;
run;

proc orthoreg data=Mills;
   class Run Period Treatment;
   effect CarryOver = lag(Treatment / period=Period within=Run);
   model Response = Run Period Treatment CarryOver;
   test Run Period Treatment CarryOver / htype=1;
   lsmeans Treatment CarryOver / diff=anom;
   ods select Tests1 LSMeans Diffs;
run;
```

Output 93.7.4 shows the carryover analysis that results from the preceding statements.

Output 93.7.4 Carryover Analysis for Crossover Experiment

The ORTHOREG Procedure

Dependent Variable: Response

Type I Tests of Model Effects

Effect	Num DF	Den DF	F Value	Pr > F
Run	5	15	13.76	<.0001
Period	5	15	7.19	0.0013
Treatment	5	15	22.95	<.0001
CarryOver	5	15	7.76	0.0009

Treatment Least Squares Means

Treatment	Estimate	Standard Error	DF	t Value	Pr > |t|
1	57.1954	0.3220	15	177.65	<.0001
2	57.6204	0.3220	15	178.97	<.0001
3	59.1919	0.3220	15	183.85	<.0001
4	59.2288	0.3220	15	183.97	<.0001
5	57.9829	0.3220	15	180.10	<.0001
6	55.0639	0.3220	15	171.03	<.0001

Output 93.7.4 *continued*

Differences of Treatment Least Squares Means						
Treatment	_Treatment	Estimate	Standard Error	DF	t Value	Pr > |t|
1	Avg	-0.5185	0.2948	15	-1.76	0.0990
2	Avg	-0.09345	0.2948	15	-0.32	0.7556
3	Avg	1.4780	0.2948	15	5.01	0.0002
4	Avg	1.5149	0.2948	15	5.14	0.0001
5	Avg	0.2690	0.2948	15	0.91	0.3758
6	Avg	-2.6500	0.2948	15	-8.99	<.0001

CarryOver Least Squares Means					
CarryOver	Estimate	Standard Error	DF	t Value	Pr > |t|
1	Non-est	.	.	.	.
2	Non-est	.	.	.	.
3	Non-est	.	.	.	.
4	Non-est	.	.	.	.
5	Non-est	.	.	.	.
6	Non-est	.	.	.	.

Differences of CarryOver Least Squares Means						
CarryOver	_CarryOver	Estimate	Standard Error	DF	t Value	Pr > |t|
1	Avg	0.3726	0.3284	15	1.13	0.2743
2	Avg	-0.2774	0.3284	15	-0.84	0.4116
3	Avg	0.6512	0.3284	15	1.98	0.0660
4	Avg	-1.3274	0.3284	15	-4.04	0.0011
5	Avg	1.3976	0.3284	15	4.26	0.0007
6	Avg	-0.8167	0.3284	15	-2.49	0.0252

The Type I analysis of variance indicates that all effects are significant—in particular, both the direct and the carryover effects of the treatment. In the presence of carryover effects, the LS-means need to be defined with some care. The LS-means for treatments computed using balanced margins for the carryover effect are inestimable; so the OBSMARGINS option is specified in the LSMEANS statement in order to use the observed margins instead. The observed margins take the absence of a carryover effect in the first period into account. Note that the LS-means themselves of the carryover effect are inestimable, but their differences are estimable. The LS-means of the direct effect of the treatment and the ANOM differences for the LS-means of their carryover effect match the "adjusted direct effects" and "adjusted residual effects," respectively, of Cox (1992).

References

Cochran, W. G., and Cox, G. M. (1957). *Experimental Designs*. 2nd ed. New York: John Wiley & Sons.

Cox, D. R. (1992). *Planning of Experiments*. New York: John Wiley & Sons.

Jarrett, R. G., and Hall, W. B. (1978). "Generalized Cyclic Incomplete Block Designs." *Biometrika* 65:397–401.

Kempthorne, O. (1952). *The Designs and Analysis of Experiments*. New York: John Wiley & Sons.

Williams, E. J. (1949). "Experimental Designs Balanced for the Estimation of Residual Effects of Treatments." *Australian Journal of Scientific Research, Series A* 2:149–168.

Chapter 94
The PLM Procedure

Contents

Overview: PLM Procedure . . . **7778**
- Basic Features . . . 7778
- PROC PLM Contrasted with Other SAS Procedures . . . 7779

Getting Started: PLM Procedure . . . **7780**

Syntax: PLM Procedure . . . **7787**
- PROC PLM Statement . . . 7788
- CODE Statement . . . 7792
- EFFECTPLOT Statement . . . 7793
- ESTIMATE Statement . . . 7794
- FILTER Statement . . . 7795
- LSMEANS Statement . . . 7796
- LSMESTIMATE Statement . . . 7797
- SCORE Statement . . . 7799
- SHOW Statement . . . 7801
- SLICE Statement . . . 7804
- TEST Statement . . . 7804
- WHERE Statement . . . 7805

Details: PLM Procedure . . . **7806**
- BY Processing and the PLM Procedure . . . 7806
- Analysis Based on Posterior Estimates . . . 7807
- Scoring Data Sets for Zero-Inflated Models . . . 7808
- User-Defined Formats and the PLM Procedure . . . 7809
- ODS Table Names . . . 7810
- ODS Graphics . . . 7811

Examples: PLM Procedure . . . **7811**
- Example 94.1: Scoring with PROC PLM . . . 7811
- Example 94.2: Working with Item Stores . . . 7813
- Example 94.3: Group Comparisons in an Ordinal Model . . . 7815
- Example 94.4: Posterior Inference for Binomial Data . . . 7817
- Example 94.5: BY-Group Processing . . . 7822
- Example 94.6: Comparing Multiple B-Splines . . . 7826
- Example 94.7: Linear Inference with Arbitrary Estimates . . . 7832

References . . . **7835**

Overview: PLM Procedure

The PLM procedure performs postfitting statistical analyses for the contents of a SAS item store that was previously created with the STORE statement in some other SAS/STAT procedure. An item store is a special SAS-defined binary file format used to store and restore information with a hierarchical structure.

The statements available in the PLM procedure are designed to reveal the contents of the source item store via the Output Delivery System (ODS) and to perform postfitting tasks such as the following:

- testing hypotheses
- computing confidence intervals
- producing prediction plots
- scoring a new data set

The use of item stores and PROC PLM enables you to separate common postprocessing tasks, such as testing for treatment differences and predicting new observations under a fitted model, from the process of model building and fitting. A numerically expensive model fitting technique can be applied once to produce a source item store. The PLM procedure can then be called multiple times and the results of the fitted model analyzed without incurring the model fitting expenditure again.

The PLM procedure offers the most advanced postprocessing techniques available in SAS/STAT software. These techniques include step-down multiplicity adjustments for p-values, F tests with order restrictions, analysis of means (ANOM), and sampling-based linear inference based on Bayes posterior estimates.

The following procedures support the STORE statement for the generation of item stores that can be processed with the PLM procedure: GEE, GENMOD, GLIMMIX, GLM, GLMSELECT, LIFEREG, LOGISTIC, MIXED, ORTHOREG, PHREG, PROBIT, SURVEYLOGISTIC, SURVEYPHREG, and SURVEYREG. The RELIABILITY procedure in SAS/QC software also supports the STORE statement. For details about the STORE statement, see the section "STORE Statement" on page 554 of Chapter 20, "Shared Concepts and Topics."

Basic Features

The PLM procedure, unlike most SAS/STAT procedures, does not operate primarily on an input data set. Instead, the procedure requires you to specify an item store with the RESTORE= option in the PROC PLM statement. The item store contains the necessary information and context about the statistical model that was fit when the store was created. SAS data sets are used only to provide input information in some circumstances. For example, when scoring a data set or when computing least squares means with specially defined population margins. In other words, instead of reading raw data and fitting a model, the PLM procedure reads the results of a model having been fit.

In order to interact with the item store and to reveal its contents, the PLM procedure supports the SHOW statement which converts item store information into standard ODS tables for viewing and further processing.

The PLM procedure is sensitive to the contents of the item store. For example, if a BAYES statement was in effect when the item store was created, the posterior parameter estimates are saved to the item store so

that the PLM procedure can perform postprocessing tasks by taking the posterior distribution of estimable functions into account. As another example, for item stores that are generated by a mixed model procedure using the Satterthwaite or Kenward-Roger (Kenward and Roger 1997) degrees-of-freedom method, these methods continue to be available when the item store contents are processed with the PLM procedure.

Because the PLM procedure does not read data and does not fit a model, the processing time of this procedure is usually considerably less than the processing time of the procedure that generates the item store.

PROC PLM Contrasted with Other SAS Procedures

In contrast to other analytic procedures in SAS/STAT software, the PLM procedure does not use an input data set. Instead, it retrieves information from an item store.

Some of the statements in the PLM procedure are also available as postprocessing statements in other procedures. Table 94.1 lists SAS/STAT procedures that support the same postprocessing statements as PROC PLM does.

Table 94.1 SAS/STAT Procedures with Postprocessing Statements Similar to Those of PROC PLM

	EFFECTPLOT	ESTIMATE	LSMEANS	LSMESTIMATE	SLICE	TEST
GEE	√	√	√	√	√	
GENMOD	√	√*	√	√	√	
GLIMMIX		√*	√*	√*	√	
GLM		√*	√*			√*
ICPHREG						√
LIFEREG	√	√	√	√	√	√
LOGISTIC	√	√	√	√	√	√*
MIXED		√*	√*	√	√	
ORTHOREG	√	√	√	√	√	√
PHREG		√	√	√	√	√*
PROBIT	√	√	√	√	√	√
SURVEYLOGISTIC		√	√	√	√	√*
SURVEYPHREG		√	√	√	√	√
SURVEYREG		√	√	√	√	√

Table entries marked with √ indicate procedures that support statements with the same functionality as in PROC PLM. Those entries marked with √* indicate procedures that support statements with same names but different syntaxes from PROC PLM. You can find the most comprehensive set of features for these statements in the PLM procedure. For example, the LSMEANS statement is available in all of the listed procedures. For example, the ESTIMATE statement available in the GENMOD, GLIMMIX, GLM and MIXED procedures does not support all options that PROC PLM supports, such as multiple rows and multiplicity adjustments.

The WHERE statement in other procedures enables you to conditionally select a subset of the observations from the input data set so that the procedure processes only the observations that meet the specified conditions. Since the PLM procedure does not use an input data set, the WHERE statement in the PLM procedure has different functionality. If the item store contains information about BY groups—that is, a BY statement was

in effect when the item store was created—you can use the WHERE statement to select specific BY groups for the analysis. You can also use the FILTER statement in the PLM procedure to filter results from the ODS output and output data sets.

Getting Started: PLM Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/plmgs.sas.)

The following DATA step creates a data set from a randomized block experiment with a factorial treatment structure of factors A and B:

```
data BlockDesign;
   input block a b y @@;
   datalines;
   1 1 1  56  1 1 2  41
   1 2 1  50  1 2 2  36
   1 3 1  39  1 3 2  35
   2 1 1  30  2 1 2  25
   2 2 1  36  2 2 2  28
   2 3 1  33  2 3 2  30
   3 1 1  32  3 1 2  24
   3 2 1  31  3 2 2  27
   3 3 1  15  3 3 2  19
   4 1 1  30  4 1 2  25
   4 2 1  35  4 2 2  30
   4 3 1  17  4 3 2  18
;
```

The GLM procedure is used in the following statements to fit the model and to create a source item store for the PLM procedure:

```
proc glm data=BlockDesign;
   class block a b;
   model y = block a b a*b / solution;
   store sasuser.BlockAnalysis / label='PLM: Getting Started';
run;
```

The CLASS statement identifies the variables Block, A, and B as classification variables. The MODEL statement specifies the response variable and the model effects. The block effect models the design effect, and the a, b, and a*b effects model the factorial treatment structure. The STORE statement requests that the context and results of this analysis be saved to an item store named sasuser.BlockAnalysis. Because the SASUSER library is specified as the library name of the item store, the store will be available after the SAS session completes. The optional label in the STORE statement identifies the store in subsequent analyses with the PLM procedure.

Note that having BlockDesign as the name of the output store would not create a conflict with the input data set name, because data sets and item stores are saved as files of different types.

Figure 94.1 displays the results from the GLM procedure. The “Class Level Information” table shows the number of levels and their values for the three classification variables. The “Parameter Estimates” table

shows the estimates and their standard errors along with *t* tests.

Figure 94.1 Class Variable Information, Fit Statistics, and Parameter Estimates

The GLM Procedure

Class Level Information		
Class	**Levels**	**Values**
block	4	1 2 3 4
a	3	1 2 3
b	2	1 2

R-Square	Coeff Var	Root MSE	y Mean
0.848966	15.05578	4.654747	30.91667

Parameter		Estimate		Standard Error	t Value	Pr > |t|
Intercept		20.41666667	B	2.85043856	7.16	<.0001
block	**1**	17.00000000	B	2.68741925	6.33	<.0001
block	**2**	4.50000000	B	2.68741925	1.67	0.1148
block	**3**	-1.16666667	B	2.68741925	-0.43	0.6704
block	**4**	0.00000000	B	.	.	.
a	**1**	3.25000000	B	3.29140294	0.99	0.3391
a	**2**	4.75000000	B	3.29140294	1.44	0.1695
a	**3**	0.00000000	B	.	.	.
b	**1**	0.50000000	B	3.29140294	0.15	0.8813
b	**2**	0.00000000	B	.	.	.
a*b	**1 1**	7.75000000	B	4.65474668	1.66	0.1167
a*b	**1 2**	0.00000000	B	.	.	.
a*b	**2 1**	7.25000000	B	4.65474668	1.56	0.1402
a*b	**2 2**	0.00000000	B	.	.	.
a*b	**3 1**	0.00000000	B	.	.	.
a*b	**3 2**	0.00000000	B	.	.	.

The following statements invoke the PLM procedure and use sasuser.BlockAnalysis as the source item store:

```
proc plm restore=sasuser.BlockAnalysis;
run;
```

These statements produce Figure 94.2. The “Store Information” table displays information that is gleaned from the source item store. For example, the store was created by the GLM procedure at the indicated time and date, and the input data set for the analysis was WORK.BLOCKDESIGN. The label used earlier in the STORE statement of the GLM procedure also appears as a descriptor in Figure 94.2.

Figure 94.2 Default Information

The PLM Procedure

Store Information	
Item Store	SASUSER.BLOCKANALYSIS
Label	PLM: Getting Started
Data Set Created From	WORK.BLOCKDESIGN
Created By	PROC GLM
Date Created	16APR20:01:36:56
Response Variable	y
Class Variables	block a b
Model Effects	Intercept block a b a*b

Class Level Information		
Class	**Levels**	**Values**
block	4	1 2 3 4
a	3	1 2 3
b	2	1 2

The "Store Information" table also echoes partial information about the variables and model effects that are used in the analysis. The "Class Level Information" table is produced by the PLM procedure by default whenever the model contains effects that depend on CLASS variables.

The following statements request a display of the fit statistics and the parameter estimates from the source item store and a test of the treatment main effects and their interactions:

```
proc plm restore=sasuser.BlockAnalysis;
   show fit parms;
   test a b a*b;
run;
```

The statements produce Figure 94.3. Notice that the estimates and standard errors in the "Parameter Estimates" table agree with the results displayed earlier by the GLM procedure, except for small differences in formatting.

Figure 94.3 Fit Statistics, Parameter Estimates, and Tests of Effects

The PLM Procedure

Fit Statistics	
MSE	21.66667
Error df	15

Figure 94.3 *continued*

Parameter Estimates					
Effect	block	a	b	Estimate	Standard Error
Intercept				20.4167	2.8504
block	1			17.0000	2.6874
block	2			4.5000	2.6874
block	3			-1.1667	2.6874
block	4			0	.
a		1		3.2500	3.2914
a		2		4.7500	3.2914
a		3		0	.
b			1	0.5000	3.2914
b			2	0	.
a*b		1	1	7.7500	4.6547
a*b		1	2	0	.
a*b		2	1	7.2500	4.6547
a*b		2	2	0	.
a*b		3	1	0	.
a*b		3	2	0	.

Type III Tests of Model Effects				
Effect	Num DF	Den DF	F Value	Pr > F
a	2	15	7.54	0.0054
b	1	15	8.38	0.0111
a*b	2	15	1.74	0.2097

Since the main effects, but not the interaction are significant in this experiment, the subsequent analysis focuses on the main effects, in particular on the effect of variable A.

The following statements request the least squares means of the A effect along with their pairwise differences:

```
proc plm restore=sasuser.BlockAnalysis seed=3;
   lsmeans     a          / diff;
   lsmestimate a -1 1,
                  1  1 -2 / uppertailed ftest;
run;
```

The LSMESTIMATE statement tests two linear combinations of the A least squares means: equality of the first two levels and whether the sum of the first two level effects equals twice the effect of the third level. The FTEST option in the LSMESTIMATE statement requests a joint F test for this two-row contrast. The UPPERTAILED option requests that the F test also be carried out under one-sided order restrictions. Since F tests under order restrictions (chi-bar-square statistic) require a simulation-based approach for the calculation of p-values, the random number stream is initialized with a seed value through the SEED= option in the PROC PLM statement.

The results of the LSMEANS and the LSMESTIMATE statement are shown in Figure 94.4.

Figure 94.4 LS-Means Related Inference for A Effect

The PLM Procedure

a Least Squares Means

a	Estimate	Standard Error	DF	t Value	Pr > |t|
1	32.8750	1.6457	15	19.98	<.0001
2	34.1250	1.6457	15	20.74	<.0001
3	25.7500	1.6457	15	15.65	<.0001

Differences of a Least Squares Means

a	_a	Estimate	Standard Error	DF	t Value	Pr > |t|
1	2	-1.2500	2.3274	15	-0.54	0.5991
1	3	7.1250	2.3274	15	3.06	0.0079
2	3	8.3750	2.3274	15	3.60	0.0026

Least Squares Means Estimates

Effect	Label	Estimate	Standard Error	DF	t Value	Tails	Pr > t
a	Row 1	1.2500	2.3274	15	0.54	Upper	0.2995
a	Row 2	15.5000	4.0311	15	3.85	Upper	0.0008

F Test for Least Squares Means Estimates

Effect	Num DF	Den DF	F Value	Pr > F	ChiBarSq Value	Pr > ChiBarSq
a	2	15	7.54	0.0054	15.07	0.0001

The least squares means for the three levels of variable A are 32.875, 34.125, and 25.75. The differences between the third level and the first and second levels are statistically significant at the 5% level (p-values of 0.0079 and 0.0026, respectively). There is no significant difference between the first two levels. The first row of the "Least Squares Means Estimates" table also displays the difference between the first two levels of factor A. Although the (absolute value of the) estimate and its standard error are identical to those in the "Differences of a Least Squares Means" table, the p-values do not agree because one-sided tests were requested in the LSMESTIMATE statement.

The "F Test" table in Figure 94.4 shows the two degree-of-freedom test for the linear combinations of the LS-means. The F value of 7.54 with p-value of 0.0054 represents the usual (two-sided) F test. Under the one-sided right-tailed order restriction imposed by the UPPERTAILED option, the ChiBarSq value of 15.07 represents the observed value of the chi-bar-square statistic of Silvapulle and Sen (2004). The associated p-value of 0.0001 was obtained by simulation.

Now suppose that you are interested in analyzing the relationship of the interaction cell means. (Typically this would not be the case in this example since the a*b interaction is not significant; see Figure 94.3.) The SLICE statement in the following PROC PLM run produces an F test of equality and all pair-wise differences of the interaction means for the subset (partition) where variable B is at level '1'. With ODS Graphics enabled, the pairwise differences are displayed in a diffogram.

```
ods graphics on;
proc plm restore=sasuser.BlockAnalysis;
   slice a*b / sliceby(b='1') diff;
run;
ods graphics off;
```

The results are shown in Figure 94.5. Since variable A has three levels, the test of equality of the A means at level '1' of B is a two-degree comparison. This comparison is statistically significant (*p*-value equals 0.0040). You can conclude that the three levels of A are not the same for the first level of B.

Figure 94.5 Results from Analyzing an Interaction Partition

The PLM Procedure

F Test for a*b Least Squares Means Slice

Slice	Num DF	Den DF	F Value	Pr > F
b 1	2	15	8.18	0.0040

Simple Differences of a*b Least Squares Means

Slice	a	_a	Estimate	Standard Error	DF	t Value	Pr > |t|
b 1	1	2	-1.0000	3.2914	15	-0.30	0.7654
b 1	1	3	11.0000	3.2914	15	3.34	0.0045
b 1	2	3	12.0000	3.2914	15	3.65	0.0024

The table of "Simple Differences" was produced by the DIFF option in the SLICE statement. As is the case with the marginal comparisons in Figure 94.4, there are significant differences against the third level of A if variable B is held fixed at '1'.

Figure 94.6 shows the diffogram that displays the three pairwise least squares mean differences and their significance. Each line segment corresponds to a comparison. It centers at the least squares means in the pair with its length corresponding to the projected width of a confidence interval for the difference. If the variable B is held fixed at '1', both the first two levels are significantly different from the third level, but the difference between the first and the second level is not significant.

Figure 94.6 LS-Means Difference Diffogram

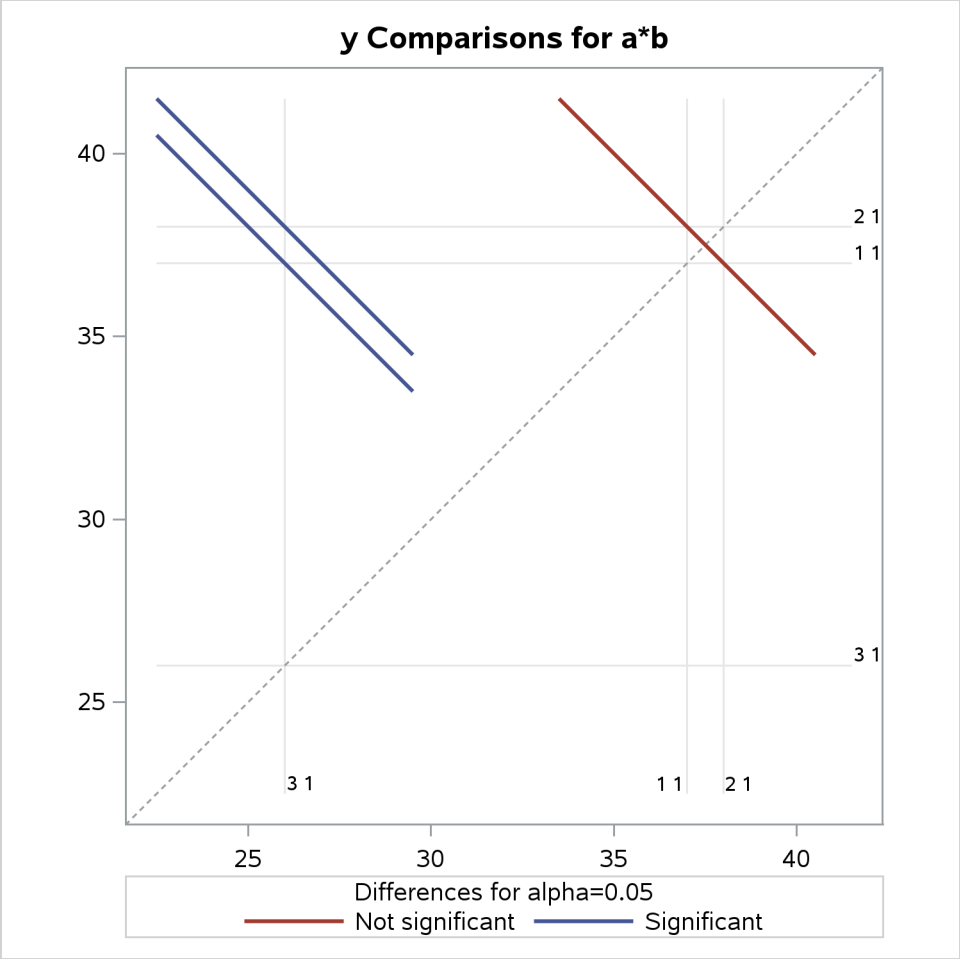

Syntax: PLM Procedure

The following statements are available in the PLM procedure:

```
PROC PLM RESTORE=item-store-specification < options > ;
    CODE < options > ;
    EFFECTPLOT < plot-type < (plot-definition-options) > > < / options > ;
    ESTIMATE < 'label' > estimate-specification < (divisor=n) >
              < , . . . < 'label' > estimate-specification < (divisor=n) > > < / options > ;
    FILTER expression ;
    LSMEANS  < model-effects > < / options > ;
    LSMESTIMATE model-effect < 'label' > values < divisor=n >
                  < , . . . < 'label' > values < divisor=n > > < / options > ;
    SCORE DATA=SAS-data-set < OUT=SAS-data-set >
          < keyword< =name > > . . .
          < keyword< =name > > < / options > ;
    SHOW options ;
    SLICE  model-effect < / options > ;
    TEST  < model-effects > < / options > ;
    WHERE expression ;
```

With the exception of the PROC PLM statement and the FILTER statement, any statement can appear multiple times and in any order. The default order in which the statements are processed by the PLM procedure depends on the specification in the item store and can be modified with the STMTORDER= option in the PROC PLM statement.

In contrast to many other SAS/STAT modeling procedures, the PLM procedure does not have common modeling statements such as the CLASS and MODEL statements. This is because the information about classification variables and model effects is contained in the source item store that is passed to the procedure in the PROC PLM statement. All subsequent statements are checked for consistency with the stored model. For example, the statement

```
lsmeans c / diff;
```

is detected as not valid unless all of the following conditions were true at the time when the source store was created:

- The effect C was used in the model.
- C was specified in the CLASS statement.
- The CLASS variables in the model had a GLM parameterization.

The FILTER, SCORE, SHOW, and WHERE statements are described in full after the PROC PLM statement in alphabetical order. The CODE EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, SLICE, and TEST statements are also used by many other procedures. Summary descriptions of functionality and syntax for these statements are also given after the PROC PLM statement in alphabetical order, but full documentation about them is available in Chapter 20, “Shared Concepts and Topics.”

PROC PLM Statement

PROC PLM RESTORE=*item-store-specification* < *options* > **;**

The PROC PLM statement invokes the PLM procedure. The RESTORE= option with *item-store-specification* is required. Table 94.2 summarizes the *options* available in the PROC PLM statement.

Table 94.2 PROC PLM Statement Options

Option	Description
Basic Options	
RESTORE=	Specifies the source item store for processing
SEED=	Specifies the random number seed
STMTORDER=	Affects the order in which statements are grouped during processing
FORMAT=	Specifies how the PLM procedure handles user-defined formats
WHEREFORMAT	Specifies the constants (literals) in terms of the formatted values of the BY variables
Computational Options	
ALPHA=	Specifies the nominal significance level
DDFMETHOD=	Specifies the method for determining denominator degrees of freedom
PERCENTILES=	Supplies a list of percentiles for the construction of HPD intervals
Displayed Output	
MAXLEN=	Determines the maximum length of informational strings
NOCLPRINT	Suppresses the display of the "Class Level Information" table
NOINFO	Suppresses the display of the "Store Information" table
NOPRINT	Suppresses tabular and graphical output
PLOT	Controls the plots produced through ODS Graphics
Singularity Tolerances	
ESTEPS=	Specifies the tolerance value used in determining the estimability of linear functions
SINGCHOL=	Tunes the singularity criterion in Cholesky decompositions
SINGRES=	Sets the tolerance for which the residual variance or scale parameter is considered to be zero
SINGULAR=	Tunes the general singularity criterion
ZETA=	Tunes the sensitivity in forming Type III functions

You can specify the following *options*:

ALPHA=α

specifies the nominal significance level for multiplicity corrections and for the construction of confidence intervals. The value of α must be between 0 and 1. The default is the value specified in the source item store, or 0.05 if the item store does not provide a value. The confidence level based on α is $1-\alpha$.

DDFMETHOD=RESIDUAL | RES | ERROR

DDFMETHOD=NONE

DDFMETHOD=KENROG | KR | KENWARDROGER

DDFMETHOD=SATTERTH | SAT | SATTERTHWAITE

specifies the method for determining denominator degrees of freedom for tests and confidence intervals. The default degree-of-freedom method is determined by the contents of the item store. You can override the default to some extent with the DDFMETHOD= option.

If you choose DDFMETHOD=NONE, then infinite denominator degrees of freedom are assumed for tests and confidence intervals. This essentially produces z tests and intervals instead of t tests and intervals and chi-square tests instead of F tests.

The KENWARDROGER and SATTERTHWAITE methods require that the source item store contain information about these methods. This information is currently available for item stores that were created with the MIXED or GLIMMIX procedures when the appropriate DDFM= option was in effect.

ESTEPS=ϵ

specifies the tolerance value used in determining the estimability of linear functions. The default value is determined by the contents of the source item store; it is usually 1E–4.

FORMAT=NOLOAD | RELOAD

specifies how the PLM procedure handles user-defined formats, which are not permanent. When the item store is created, user-defined formats are stored. When the PLM procedure opens an item store, it uses this option as follows. If FORMAT=RELOAD (the default), the stored formats are loaded again from the item store and formats that already exist in your SAS session are replaced by the reloaded formats. If FORMAT=NOLOAD, stored formats are not loaded from the item store and existing formats are not replaced.

With FORMAT=NOLOAD, you prevent the PLM procedure from reloading the format from the item store. As a consequence, PLM statements might fail if a format was present at the item store creation and is not available in your SAS session. Also, if you modify the format that was used in the item store creation and use FORMAT=NOLOAD, you might obtain unexpected results because levels of classification variables are remapped.

The "Class Level Information" table always displays the formatted values of classification variables that were used in fitting the model, regardless of the FORMAT= option. For more details about using formats with the PLM procedure, see "User-Defined Formats and the PLM Procedure" on page 7809.

MAXLEN=*n*

determines the maximum length of informational strings in the "Store Information" table. This table displays, for example, lists of classification or BY variables and lists of model effects. The value of *n* determines the truncation length for these strings. The minimum and maximum values for *n* are 20 and 256, respectively. The default is *n* = 100.

NOCLPRINT< =*number* >

suppresses the display of the "Class Level Information" table if you do not specify *number*. If you specify *number*, only levels with totals that are less than *number* are listed in the table. The PLM procedure produces the "Class Level Information" table by default when the model contains effects that depend on classification variables.

NOINFO
suppresses the display of the "Store Information" table.

NOPRINT
suppresses the generation of tabular and graphical output. When the NOPRINT option is in effect, ODS tables are also not produced.

PERCENTILES=*value-list*
PERCENTILE=*value-list*
supplies a list of percentiles for the construction of highest posterior density (HPD) intervals when the PLM procedure performs a sampling-based analysis (for example, when processing an item store that contains posterior parameter estimates from a Bayesian analysis). The default set of percentiles depends on the contents of the source item store; it is typically PERCENTILES=25, 50, 75. The entries in *value-list* must be strictly between 0 and 100.

PLOTS < (*global-plot-option***) > < =***specific-plot-options***>**
controls the plots produced through ODS Graphics. ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc plm plots=all;
   lsmeans a/diff;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

Global Plot Option

The following *global-plot-option* applies to all plots produced by PROC PLM.

UNPACKPANEL
UNPACK
suppresses paneling. (By default, multiple plots can appear in some output panels.) Specify UNPACK to display each plot separately.

Specific Plot Options

You can specify the following *specific-plot-options*:

ALL
requests that all the appropriate plots be produced.

NONE
suppresses all plots.

SEED=*number*
specifies the random number seed for analyses that depend on a random number stream. You can also specify the random number seed through some PLM statements (for example, through the SEED= options in the ESTIMATE, LSMEANS, and LSMESTIMATE statements). However, note that there is only a single random number stream per procedure run. Specifying the SEED= option in the PROC PLM statement initializes the stream for all subsequent statements. If you do not specify a random number seed, the source item store might supply one for you. If a seed is in effect when the PLM procedure opens the source store, the "Store Information" table displays its value.

If the random number seed is less than or equal to zero, the seed is generated from reading the time of day from the computer clock and a log message indicates the chosen seed value.

SINGCHOL=*number*
tunes the singularity criterion in Cholesky decompositions. The default value depends on the contents of the source item store. The default value is typically 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGRES=*number*
sets the tolerance for which the residual variance or scale parameter is considered to be zero. The default value depends on the contents of the source item store. The default value is typically 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

SINGULAR=*number*
tunes the general singularity criterion applied by the PLM procedure in divisions and inversions. The default value used by the PLM procedure depends on the contents of the item store. The default value is typically 1E4 times the machine epsilon; this product is approximately 1E–12 on most computers.

RESTORE=*item-store-specification*
specifies the source item store for processing. This option is required because, in contrast to SAS data sets, there is no default item store. An *item-store-specification* consists of a one- or two-level name as with SAS data sets. As with data sets, the default library association of an item store is with the WORK library, and any stores created in this library are deleted when the SAS session concludes.

STMTORDER=SYNTAX | GROUP
STMT=SYNTAX | GROUP
affects the order in which statements are grouped during processing. The default behavior depends on the contents of the source item store and can be modified with the STMTORDER= option. If STMTORDER=SYNTAX is in effect, the statements are processed in the order in which they appear. Note that this precludes the hierarchical grouping of ODS objects. If STMTORDER=GROUP is in effect, the statements are processed in groups and in the following order: SHOW, TEST, LSMEANS, SLICE, LSMESTIMATE, ESTIMATE, SCORE, EFFECTPLOT, and CODE.

WHEREFORMAT

specifies that the constants (literals) specified in WHERE expressions for group selection are in terms of the formatted values of the BY variables. By default, WHERE expressions are specified in terms of the unformatted (raw) values of the BY variables, as in the SAS DATA step.

ZETA=*number*

tunes the sensitivity in forming Type III functions. Any element in the estimable function basis with an absolute value less than *number* is set to 0. The default depends on the contents of the source item store; it usually is 1E–8.

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 94.3 summarizes the *options* available in the CODE statement.

Table 94.3 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

EFFECTPLOT Statement

EFFECTPLOT < *plot-type* < *(plot-definition-options)* > > < / *options* > **;**

The EFFECTPLOT statement produces a display of the fitted model and provides options for changing and enhancing the displays. Table 94.4 describes the available *plot-types* and their *plot-definition-options*.

Table 94.4 *Plot-Types* and *Plot-Definition-Options*

Plot-Type and Description	Plot-Definition-Options
CONTOUR Displays a contour plot of predicted values against two continuous covariates.	PLOTBY= variable or CLASS effect X= continuous variable Y= continuous variable
FIT Displays a curve of predicted values versus a continuous variable.	PLOTBY= variable or CLASS effect X= continuous variable
INTERACTION Displays a plot of predicted values (possibly with error bars) versus the levels of a CLASS effect. The predicted values are connected with lines and can be grouped by the levels of another CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= CLASS variable or effect
MOSAIC Displays a mosaic plot of predicted values using up to three CLASS effects.	PLOTBY= variable or CLASS effect X= CLASS effects
SLICEFIT Displays a curve of predicted values versus a continuous variable grouped by the levels of a CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= continuous variable

For full details about the syntax and options of the EFFECTPLOT statement, see the section "EFFECTPLOT Statement" on page 447 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 94.5 summarizes the *options* available in the ESTIMATE statement.

Table 94.5 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the $\mathbf{L}$ matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement"

on page 475 in Chapter 20, "Shared Concepts and Topics."

FILTER Statement

FILTER *expression* **;**

The FILTER statement enables you to filter the results of the PLM procedure, specifically the contents of ODS tables and the output data sets. There can be at most one FILTER statement per PROC PLM run, and the filter is applied to all BY groups and to all queries generated through WHERE expressions.

A filter *expression* follows the same pattern as a *where-expression* in the WHERE statement. The expressions consist of operands and operators. For more information about specifying *where-expressions*, see the WHERE statement for the PLM procedure and *SAS Programmers Guide: Essentials*.

Valid *keywords* for the formation of operands in the FILTER statement are shown in Table 94.6.

Table 94.6 Keywords for Filtering Results

Keyword	Description
Prob	Regular (unadjusted) *p*-values from *t*, *F*, or chi-square tests
ProbChi	Regular (unadjusted) *p*-values from chi-square tests
ProbF	Regular (unadjusted) *p*-values from *F* tests
ProbT	Regular (unadjusted) *p*-values from *t* tests
AdjP	Adjusted *p*-values
Estimate	Results displayed in "Estimates" column of ODS tables
Pred	Predicted values in SCORE output data sets
Resid	Residuals in SCORE output data sets.
Std	Standard errors in ODS tables and in SCORE results
Mu	Results displayed in the "Mean" column of ODS tables (this column is typically produced by the ILINK option)
tValue	The value of the usual *t* statistic
FValue	The value of the usual *F* statistic
Chisq	The value of the chi-square statistic
testStat	The value of the test statistic (a generic *keyword* for the 'tValue', 'FValue', and 'Chisq' tokens)
Lower	The lower confidence limit displayed in ODS tables
Upper	The upper confidence limit displayed in ODS tables
AdjLower	The adjusted lower confidence limit displayed in ODS tables
AdjUpper	The adjusted upper confidence limit displayed in ODS tables
LowerMu	The lower confidence limit for the mean displayed in ODS tables
UpperMu	The upper confidence limit for the mean displayed in ODS tables
AdjLowerMu	The adjusted lower confidence limit for the mean displayed in ODS tables
AdjUpperMu	The adjusted upper confidence limit for the mean displayed in ODS tables

When you write filtering expressions, be advised that filtering variables that are not used in the results are

typically set to missing values. For example, the following statements select all results (filter nothing) because no adjusted *p*-values are computed:

```
proc plm restore=MyStore;
   lsmeans a / diff;
   filter adjp < 0.05;
run;
```

If the adjusted *p*-values are set to missing values, the condition **adjp < 0.05** is true in each case (missing values always compare smaller than the smallest nonmissing value).

See "Example 94.6: Comparing Multiple B-Splines" on page 7826 for an example of using the FILTER statement.

Filtering results has no affect on the item store contents that are displayed with the SHOW statement. However, BY-group selection with the WHERE statement can limit the amount of information that is displayed by the SHOW statements.

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 94.7 summarizes the *options* available in the LSMEANS statement.

Table 94.7 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion

Table 94.7 *continued*

Option	Description
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 94.8 summarizes the *options* available in the LSMESTIMATE statement.

Table 94.8 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

SCORE Statement

SCORE DATA=*SAS-data-set* < **OUT=***SAS-data-set* >
< *keyword*< **=***name* > > . . .
< *keyword*< **=***name* > > < / *options* > **;**

The SCORE statement applies the contents of the source item store to compute predicted values and other observation-wise statistics for a SAS data set.

You can specify the following syntax elements in the SCORE statement before the option slash (/):

DATA=*SAS-data-set*
specifies the input data set for scoring. This option is required, and the data set is examined for congruity with the previously fitted (and stored) model. For example, all necessary variables to form a row of the **X** matrix must be present in the input data set and must be of the correct type and format. The following variables do not have to be present in the input data set:

- the response variable
- the *events* and *trials* variables used in the *events/trials* syntax for binomial data
- variables used in WEIGHT or FREQ statements

OUT=*SAS-data-set*
specifies the name of the output data set. If you do not specify an output data set with the OUT= option, the PLM procedure uses the DATA*n* convention to name the output data set.

keyword< **=***name* >
specifies a statistic to be included in the OUT= data set and optionally assigns the statistic the variable name *name*. Table 94.9 lists the *keywords* and the default names assigned by the PLM procedure if you do not specify a *name*.

Table 94.9 Keywords for Output Statistics

Keyword	Description	Expression	Name
PREDICTED	Linear predictor	$\widehat{\eta} = \mathbf{x}\widehat{\boldsymbol{\beta}}$	Predicted
STDERR	Standard deviation of linear predictor	$\sqrt{\mathrm{Var}(\widehat{\eta})}$	StdErr
RESIDUAL	Residual	$y - g^{-1}(\widehat{\eta})$	Resid
LCLM	Lower confidence limit for the linear predictor		LCLM
UCLM	Upper confidence limit for the linear predictor		UCLM
LCL	Lower prediction limit for the linear predictor		LCL
UCL	Upper prediction limit for the linear predictor		UCL
PZERO	zero-inflation probability for zero-inflated models	$g_z^{-1}(\mathbf{z}\widehat{\boldsymbol{\gamma}})$	PZERO

Prediction limits (LCL, UCL) are available only for statistical models that allow such limits, typically

regression-type models for normally distributed data with an identity link function. Zero-inflation probability (PZERO) is available only for zero-inflated models. For details on how PROC PLM computes statistics for zero-inflated models, see "Scoring Data Sets for Zero-Inflated Models" on page 7808.

You can specify the following *options* in the SCORE statement after a slash (/):

ALPHA=*number*
determines the coverage probability for two-sided confidence and prediction intervals. The coverage probability is computed as 1 – *number*. The value of *number* must be between 0 and 1; the default is 0.05.

DF=*number*
specifies the degrees of freedom to use in the construction of prediction and confidence limits.

ILINK
requests that predicted values be inversely linked to produce predictions on the data scale. By default, predictions are produced on the linear scale where covariate effects are additive.

NOOFFSET
requests that the offset values not be added to the prediction if the offset variable is used in the fitted model.

NOUNIQUE
requests that names not be made unique in the case of naming conflicts. By default, the PLM procedure avoids naming conflicts by assigning a unique name to each output variable. If you specify the NOUNIQUE option, variables with conflicting names are not renamed. In that case, the first variable added to the output data set takes precedence.

NOVAR
requests that variables from the input data set not be added to the output data set.

OBSCAT
requests that statistics in models for multinomial data be written to the output data set only for the response level that corresponds to the observed level of the observation.

SAMPLE
requests that the sample of parameter estimates in the item store be used to form scoring statistics. This option is useful when the item store contains the results of a Bayesian analysis and a posterior sample of parameter estimates. The predicted value is then computed as the average predicted value across the posterior estimates, and the standard error measures the standard deviation of these estimates. For example, let $\widehat{\boldsymbol{\beta}}_1, \ldots, \widehat{\boldsymbol{\beta}}_k$ denote the k posterior sample estimates of $\boldsymbol{\beta}$, and let $\mathbf{x}_i$ denote the x-vector for the ith observation in the scoring data set. If the SAMPLE option is in effect, the output statistics for the predicted value, the standard error, and the residual of the ith observation are computed as

$$
\begin{aligned}
\eta_{ij} &= \mathbf{x}_i \widehat{\boldsymbol{\beta}}_j \\
\mathrm{PRED}_i &= \overline{\eta}_i = \frac{1}{k}\sum_{j=1}^{k} \eta_{ij} \\
\mathrm{STDERR}_i &= \left(\frac{1}{k-1} \sum_{j=1}^{k} \left(\eta_{ij} - \overline{\eta}_i\right)^2 \right)^{1/2} \\
\mathrm{RESIDUAL}_i &= y_i - g^{-1}\left(\overline{\eta}_i\right)
\end{aligned}
$$

where $g^{-1}(\cdot)$ denotes the inverse link function.

If, in addition, the ILINK option is in effect, the calculations are as follows:

$$
\begin{aligned}
\eta_{ij} &= \mathbf{x}_i \widehat{\boldsymbol{\beta}}_j \\
\mathrm{PRED}_i &= \frac{1}{k}\sum_{j=1}^{k} g^{-1}\left(\eta_{ij}\right) \\
\mathrm{STDERR}_i &= \left(\frac{1}{k-1} \sum_{j=1}^{k} \left(g^{-1}(\eta_{ij}) - \mathrm{PRED}_i\right)^2 \right)^{1/2} \\
\mathrm{RESIDUAL}_i &= y_i - \mathrm{PRED}_i
\end{aligned}
$$

The LCL and UCL statistics are not available with the SAMPLE option. When the LCLM and UCLM statistics are requested, the SAMPLE option yields the lower $100(\alpha/2)$th and upper $100(1-\alpha/2)$th percentiles of the predicted values under the sample (posterior) distribution. When you request residuals with the SAMPLE option, the calculation depends on whether the ILINK option is specified.

SHOW Statement

SHOW *options* **;**

The SHOW statement uses the Output Delivery System to display contents of the item store. This statement is useful for verifying that the contents of the item store apply to the analysis and for generating ODS tables. Table 94.10 summarizes the *options* available in the SHOW statement.

Table 94.10 SHOW Statement Options

Option	Description
ALL	Displays all applicable contents
BYVAR	Displays information about the BY variables
CLASSLEVELS	Displays the “Class Level Information” table
CORRELATION	Produces the correlation matrix of the parameter estimates
COVARIANCE	Produces the covariance matrix of the parameter estimates

Table 94.10 *continued*

Option	Description
EFFECTS	Displays information about the constructed effects
FITSTATS	Displays the fit statistics
HESSIAN	Displays the Hessian matrix
HERMITE	Generates the Hermite matrix $\mathbf{H} = (\mathbf{X}'\mathbf{X})^{-}(\mathbf{X}'\mathbf{X})$
PARAMETERS	Displays the parameter estimates
PROGRAM	Displays the SAS program that generated the item store
XPX	Displays the crossproduct matrix $\mathbf{X}'\mathbf{X}$
XPXI	Displays the generalized inverse of the crossproduct matrix $\mathbf{X}'\mathbf{X}$

You can specify the following *options* after the SHOW statement:

ALL | _ALL_
displays all applicable contents.

BYVAR | BY
displays information about the BY variables in the source item store. If a BY statement was present when the item store was created, the PLM procedure performs the analysis separately for each BY group.

CLASSLEVELS | CLASS
displays the "Class Level Information" table. This table is produced by the PLM procedure by default if the model contains effects that depend on classification variables.

CORRELATION | CORR | CORRB
produces the correlation matrix of the parameter estimates. If the source item store contains a posterior sample of parameter estimates, the computed matrix is the correlation matrix of the sample covariance matrix.

COVARIANCE | COV | COVB
produces the covariance matrix of the parameter estimates. If the source item store contains a posterior sample of parameter estimates, the PLM procedure computes the empirical sample covariance matrix from the posterior estimates. You can convert this matrix into a sample correlation matrix with the CORRELATION option in the SHOW statement.

EFFECTS
displays information about the constructed effects in the model. Constructed effects are those that were created with the EFFECT statement in the procedure run that generated the source item store.

FITSTATS | FIT
displays the fit statistics from the item store.

HESSIAN | HESS

displays the Hessian matrix.

HERMITE | HERM

generates the Hermite matrix $\mathbf{H} = (\mathbf{X}'\mathbf{X})^{-}(\mathbf{X}'\mathbf{X})$. The PLM procedure chooses a reflexive, g_2-inverse for the generalized inverse of the crossproduct matrix $\mathbf{X}'\mathbf{X}$. See "Important Linear Algebra Concepts" on page 44 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," for information about generalized inverses and the sweep operator.

PARAMETERS< =*n*>

PARMS< =*n*>

displays the parameter estimates. The structure of the display depends on whether a posterior sample of parameter estimates is available in the source item store. If such a sample is present, up to the first 20 parameter vectors are shown in wide format. You can modify this number with the *n* argument.

If no posterior sample is present, the single vector of parameter estimates is shown in narrow format. If the store contains information about the covariance matrix of the parameter estimates, then standard errors are added.

PROGRAM< (WIDTH=*n*)>

PROG< (WIDTH=*n*)>

displays the SAS program that generated the item store, provided that this was stored at store generation time. The program does not include comments, titles, or some other global statements. The optional width parameter *n* determines the display width of the source code.

The PROGRAM option is supported only by the following procedures: GLMSELECT, LIFEREG, LOGISTIC, MIXED, ORTHOREG, PROBIT, SURVEYLOGISTIC, SURVEYREG in SAS/STAT; RELIABILITY in SAS/QC; and COUNTREG in SAS/ETS.

XPX | CROSSPRODUCT

displays the crossproduct matrix $\mathbf{X}'\mathbf{X}$.

XPXI

displays the generalized inverse of the crossproduct matrix $\mathbf{X}'\mathbf{X}$. The PLM procedure obtains a reflexive g_2-inverse by sweeping. See "Important Linear Algebra Concepts" on page 44 in Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," for information about generalized inverses and the sweep operator.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

TEST Statement

TEST < *model-effects* > < / *options* > **;**

The TEST statement enables you to perform *F* tests for model effects that test Type I, Type II, or Type III hypotheses. For more information about constructing Type I, II, and III estimable functions, see Chapter 16, "The Four Types of Estimable Functions."

Table 94.11 summarizes the *options* available in the TEST statement.

Table 94.11 TEST Statement Options

Option	Description
CHISQ	Requests chi-square tests
DDF=	Specifies denominator degrees of freedom for fixed effects
E	Requests Type I, Type II, and Type III coefficients
E1	Requests Type I coefficients
E2	Requests Type II coefficients
E3	Requests Type III coefficients
HTYPE=	Indicates the type of hypothesis test to perform
INTERCEPT	Adds a row that corresponds to the overall intercept

For more information about the syntax of the TEST statement, see the section "TEST Statement" on page 556 in Chapter 20, "Shared Concepts and Topics."

WHERE Statement

WHERE *expression* **;**

You can use the WHERE statement in the PLM procedure when the item store contains BY-variable information and you want to apply the PROC PLM statements to only a subset of the BY groups.

A WHERE expression is a type of SAS expression that defines a condition. In the DATA step and in procedures that use SAS data sets as the input source, the WHERE expression is used to select observations for inclusion in the DATA step or in the analysis. In the PLM procedure, which does not accept a SAS data set but rather takes an item store that was created by a qualifying SAS/STAT procedure, the WHERE statement is also used to specify conditions. The conditional selection does not apply to observations in PROC PLM, however. Instead, you use the WHERE statement in the PLM procedure to select a subset of BY groups from the item store to which to apply the PROC PLM statements.

The general syntax of the WHERE statement is

WHERE *operand* < *operator operand* > < **AND** | **OR** *operand* < *operator operand* >. . . > **;**

where

operand is something to be operated on. The operand can be the name of a BY variable in the item store, a SAS function, a constant, or a predefined name to identify columns in result tables.

operator is a symbol that requests a comparison, logical operation, or arithmetic calculation. All SAS expression operators are valid for a WHERE expression.

For more details about how to specify general WHERE expressions, see *SAS Programmers Guide: Essentials*. Notice that the FILTER statement accepts similar expressions that are specified in terms of predefined *keywords*. Expressions in the WHERE statement of the PLM procedure are written in terms of BY variables.

There is no limit to the number of WHERE statements in the PLM procedure. When you specify multiple WHERE statements, the statements are *not* cumulative. Each WHERE statement is executed separately. You can think of each selection WHERE statement as one analytic query to the item store: the WHERE statement defines the query, and the PLM procedure is the querying engine. For example, suppose that the item store contains results for the numeric BY variables A and B. The following statements define two separate queries of the item store:

```
WHERE a = 4;
WHERE (b < 3) and (a > 4);
```

The PLM procedure first applies the requested analysis to all BY groups where a equals 4 (irrespective of the value of variable b). The analysis is then repeated for all BY groups where b is less than 3 and a is greater than 4.

Group selection with WHERE statements is possible only if the item store contains BY variables. You can use the BYVAR option in the SHOW statement to display the BY variables in the item store.

Note that WHERE expressions in the SAS DATA step and in many procedures are specified in terms of the unformatted values of data set variables, even if a format was applied to the variable. If you specify the WHEREFORMAT option in the PROC PLM statement, the PLM procedure evaluates WHERE expressions for BY variables in terms of the formatted values. For example, assume that the following format was applied to the variable tx when the item store was created:

```
proc format;
   value bf 1 = 'Control'
            2 = 'Treated';
run;
```

Then the following two PROC PLM runs are equivalent:

```
proc plm restore=MyStore;
   show parms;
   where b = 2;
run;

proc plm restore=MyStore whereformat;
   show parms;
   where b = 'Treated';
run;
```

Details: PLM Procedure

BY Processing and the PLM Procedure

When a BY statement is in effect for the analysis that creates an item store, the information about BY variables and BY-group-specific modeling results are transferred to the item store. In this case, the PLM procedure automatically assumes a processing mode for the item store that is akin to BY processing, with the PLM statements being applied in turn for each of the BY groups. Also, you can then obtain a table of BY groups with the BYVAR option in the SHOW statement. The "Source Information" table also displays the variable names of the BY variables if BY groups are present. The WHERE statement can be used to restrict the analysis to specific BY groups that meet the conditions of the WHERE expression.

See Example 94.4 for an example that uses BY-group-specific information in the source item store.

As with procedures that operate on input data sets, the BY variable information is added automatically to any output data sets and ODS tables produced by the PLM procedure.

When you score a data set with the SCORE statement and the item store contains BY variables, three situations can arise:

- None of the BY variables are present in the scoring data set. In this situation the results of the BY groups in the item store are applied in turn to the entire scoring data set. For example, if the scoring data set contains 50 observations and no BY-variable information, the number of observations in the output data set of the SCORE statement equals 50 times the number of BY groups.
- The scoring data set contains only a part of the BY variables, or the variables have different type or format. The PLM procedure does not process such an incompatible scoring data set.
- All BY variables are in the scoring data set in the same type and format as when the item store was created. The BY-group-specific results are applied to each observation in the scoring data set.

The scoring data set does not have to be sorted or grouped by the BY variables. However, it is computationally more efficient if the scoring data set is arranged by groups of the BY variables.

Analysis Based on Posterior Estimates

If an item store is saved from a Bayesian analysis (by PROC GENMOD or PROC PHREG or PROC LIFEREG), then PROC PLM can perform sampling-based inference based on Bayes posterior estimates that are saved in the item store. For example, the following statements request a Bayesian analysis and save the results to an item store named sasuser.gmd. For the Bayesian analysis, the random number generator seed is set to 1. By default, a noninformative distribution is set as the prior distribution for the regression coefficients and the posterior sample size is 10,000.

```
proc genmod data=gs;
   class a b;
   model y = a b;
   bayes seed=1;
   store sasuser.gmd / label='Bayesian Analysis';
run;
```

When the PLM procedure opens the item store sasuser.gmd, it detects that the results were saved from a Bayesian analysis. The posterior sample of regression coefficient estimates are then loaded to perform statistical inference tasks.

The majority of postprocessing tasks involve inference based on an estimable linear function $\mathbf{L}\widehat{\boldsymbol{\beta}}$, which often requires its mean and variance. When the standard frequentist analyses are performed, the mean and variance have explicit forms because the parameter estimate $\widehat{\boldsymbol{\beta}}$ is analytically tractable. However, explicit forms are not usually available when Bayesian models are fitted. Instead, empirical means and variance-covariance matrices for the estimable function are constructed from the posterior sample.

Let $\widehat{\boldsymbol{\beta}}_i, i = 1, \ldots, N_p$ denote the N_p vectors of posterior sample estimates of $\boldsymbol{\beta}$ saved in sasuser.gmd. Use these vectors to construct the posterior sample of estimable functions $\mathbf{L}\boldsymbol{\beta}_i$. The posterior mean of the estimable function is thus

$$\overline{\mathbf{L}\widehat{\boldsymbol{\beta}}} = \frac{1}{N_p}\sum_{i=1}^{N_p}\mathbf{L}\widehat{\boldsymbol{\beta}}_i$$

and the posterior variance of the estimable function is

$$\mathbf{V}\left(\mathbf{L}\widehat{\boldsymbol{\beta}}\right) = \frac{1}{N_p - 1}\sum_{i=1}^{N_p}\left(\mathbf{L}\widehat{\boldsymbol{\beta}}_i - \overline{\mathbf{L}\widehat{\boldsymbol{\beta}}}\right)^2$$

Sometimes statistical inference on a transformation of $\mathbf{L}\widehat{\boldsymbol{\beta}}$ is requested. For example, the EXP option for the ESTIMATE and LSMESTIMATE statements requests analysis based on $\exp(\mathbf{L}\widehat{\boldsymbol{\beta}})$, exponentiation of the estimable function. If this type of analysis is requested, the posterior sample of transformed estimable functions is constructed by transforming each of the estimable function evaluated at the posterior sample: $f(\mathbf{L}\widehat{\boldsymbol{\beta}}_i), i = 1, \ldots, N_p$. The posterior mean and variance for $f(\mathbf{L}\widehat{\boldsymbol{\beta}})$ are then computed from the constructed

sample to make the inference:

$$\overline{f(\mathbf{L}\widehat{\boldsymbol{\beta}})} = \frac{1}{N_p}\sum_{i=1}^{N_p} f(\mathbf{L}\widehat{\boldsymbol{\beta}}_i)$$

$$\mathbf{V}\left(f(\mathbf{L}\widehat{\boldsymbol{\beta}})\right) = \frac{1}{N_p - 1}\sum_{i=1}^{N_p}\left(f(\mathbf{L}\widehat{\boldsymbol{\beta}}_i) - \overline{f(\mathbf{L}\widehat{\boldsymbol{\beta}})}\right)^2$$

After obtaining the posterior mean and variance, the PLM procedure proceeds to perform statistical inference based on them.

Scoring Data Sets for Zero-Inflated Models

The PLM procedure can score new observations for zero-inflated models with the SCORE statement. If you specify the ILINK option, the computed statistics are for estimated counts.

In the following formula, $\mathbf{x}$ is the design row for covariates that correspond to the Poisson or negative binomial component, $\hat{\boldsymbol{\beta}}$ is the column vector of the fitted regression parameters; $\mathbf{z}$ is the design row for covariates that correspond to the zero inflation component, $\hat{\boldsymbol{\gamma}}$ is the column vector of the fitted regression parameters; g and g^{-1} are the link and inverse link functions for the Poisson or negative binomial component, g_z and g_z^{-1} are the link and inverse link functions for the zero inflation component; Φ is the standard normal cumulative distribution function and α is the nominal significance level. Let

$$\begin{array}{lclcl}
\mu & = & g^{-1}(\eta) & = & g^{-1}(\mathbf{x}\hat{\boldsymbol{\beta}}) \\
\mu_z & = & g_z^{-1}(\eta_z) & = & g_z^{-1}(\mathbf{z}\hat{\boldsymbol{\gamma}}) \\
v_1 & = & (\frac{\mathrm{d}\mu}{\mathrm{d}\eta})^2\mathbf{x}\hat{\mathbf{V}}_{\boldsymbol{\beta},\boldsymbol{\beta}}\mathbf{x}' & & \\
v_2 & = & (\frac{\mathrm{d}\mu_z}{\mathrm{d}\eta_z})^2\mathbf{z}\hat{\mathbf{V}}_{\boldsymbol{\gamma},\boldsymbol{\gamma}}\mathbf{z}' & & \\
v_{12} & = & -(\frac{\mathrm{d}\mu}{\mathrm{d}\eta})(\frac{\mathrm{d}\mu_z}{\mathrm{d}\eta_z})\mathbf{x}\hat{\mathbf{V}}_{\boldsymbol{\beta},\boldsymbol{\gamma}}\mathbf{z}' & &
\end{array}$$

The formula for statistics in the SCORE statement for zero-inflated models are listed as follows.

$$\begin{array}{lclcl}
\text{PZERO} & = & \omega & = & \mu_z \\
\text{PRED/ILINK} & = & p_c & = & \mu(1-\omega) \\
\text{STD/ILINK} & = & s_c & = & \sqrt{p_c^2 v_2 + (1-\omega)^2 v_1 + v_1 v_2 + 2p_c(1-\omega)v_{12} + v_{12}^2} \\
\text{UCLM/ILINK} & = & u_c & = & p_c \exp(\Phi^{-1}(1-\alpha/2)s_c/p_c) \\
\text{LCLM/ILINK} & = & l_c & = & p_c/\exp(\Phi^{-1}(1-\alpha/2)s_c/p_c) \\
\text{PRED} & = & p_l & = & g(p_c) \\
\text{STD} & = & s_l & = & g(s_c)/(\frac{\mathrm{d}\mu}{\mathrm{d}\eta}) \\
\text{UCLM} & = & u_l & = & g(u_c) \\
\text{LCLM} & = & l_l & = & g(l_c)
\end{array}$$

User-Defined Formats and the PLM Procedure

The PLM procedure does not support a FORMAT statement because it operates without an input data set, and also because changing the format properties of variables could alter the interpretation of parameter estimates, thus creating a dissonance with variable properties in effect when the item store was created. Instead, user-defined formats that are applied to classification variables when the item store is created are saved to the store and are by default reloaded by the PLM procedure. When the PLM procedure loads a format, notes are issued to the log.

You can change the load behavior for formats with the FORMAT= option in the PROC PLM statement.

User-defined formats do not need to be supplied in a new SAS session. However, when a user-defined format with the same name as a stored format exists and the default FORMAT=RELOAD option is in effect, the format definition loaded from the item store replaces the format currently in effect.

In the following statements, the format AFORM is created and applied to the variable a in the PROC GLM step. This format definition is transferred to the item store sasuser.glm through the STORE statement.

```
proc format;
   value aform 1='One' 2='Two' 3='Three';
run;
proc glm data=sp;
   format a aform.;
   class block a b;
   model y = block a b x;
   store sasuser.glm;
   weight x;
run;
```

The following statements replace the format definition of the AFORM format. The PLM step then reloads the AFORM format (from the item store) and thereby restores its original state.

```
proc format;
   value aform 1='Un' 2='Deux' 3='Trois';
run;
proc plm restore=sasuser.glm;
   show class;
   score data=sp out=plmout lcl lclm ucl uclm;
run;
```

The following notes, issued by the PLM procedure, inform you that the procedure loaded the format, the format already existed, and the existing format was replaced:

```
NOTE: The format AFORM was loaded from item store SASUSER.GLM.
NOTE: Format AFORM is already on the library.
NOTE: Format AFORM has been output.
```

After the PROC PLM run, the definition that is in effect for the format AFORM corresponds to the following SAS statements:

```
proc format;
   value aform 1='One' 2='Two' 3='Three';
run;
```

ODS Table Names

PROC PLM assigns a name to each table it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 94.12. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Each of the EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements also creates tables, which are not listed in Table 94.12. For information about these tables, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Table 94.12 ODS Tables Produced by PROC PLM

Table Name	Description	Required Option
ByVarInfo	Information about BY variables in source item store (if present)	SHOW BYVAR
ClassLevels	Level information from the CLASS statement	Default output when model effects depend on CLASS variables
Corr	Correlation matrix of parameter estimates	SHOW CORR
Cov	Covariance matrix of parameter estimates	SHOW COV
FitStatistics	Fit statistics	SHOW FIT
Hessian	Hessian matrix	SHOW HESSIAN
Hermite	Hermite matrix	SHOW HERMITE
ParameterEstimates	Parameter estimates	SHOW PARMS
ParameterSample	Sampled (posterior) parameter estimates	SHOW PARMS
Program[1]	Originating source code	SHOW PROGRAM
StoreInfo	Information about source item store	Default
XpX	$\mathbf{X}'\mathbf{X}$ matrix	SHOW XPX
XpXI	$(\mathbf{X}'\mathbf{X})^{-}$ matrix	SHOW XPXI

[1]Note: The ODS table is only available for item stores produced by the following procedures: GLMSELECT, LIFEREG, LOGISTIC, MIXED, ORTHOREG, PROBIT, SURVEYLOGISTIC, SURVEYREG in SAS/STAT, RELIABILITY in SAS/QC and COUNTREG in SAS/ETS.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, then each of the EFFECTPLOT, ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements can produce plots associated with their analyses. For information about these plots, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Examples: PLM Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 94.1: Scoring with PROC PLM

(View the complete code for this example (plmex1.sas) in the example repository.)

Logistic regression with model selection is often used to extract useful information and build interpretable models for classification problems with many variables. This example demonstrates how you can use PROC LOGISTIC to build a spline model on a simulated data set and how you can later use the fitted model to classify new observations.

The following DATA step creates a data set named SimuData, which contains 5,000 observations and 100 continuous variables:

```
%let nObs   = 5000;
%let nVars  = 100;
data SimuData;
   array x{&nVars};
   do obsNum=1 to &nObs;
      do j=1 to &nVars;
         x{j}=ranuni(1);
      end;

      linp =  10 + 11*x1 - 10*sqrt(x2) + 2/x3 - 8*exp(x4) + 7*x5*x5
             - 6*x6**1.5 + 5*log(x7) - 4*sin(3.14*x8) + 3*x9 - 2*x10;
      TrueProb = 1/(1+exp(-linp));

      if ranuni(1) < TrueProb then y=1;
```

```
                                    else y=0;
      output;
   end;
run;
```

The response is binary based on the inversely transformed logit values. The true logit is a function of only 10 of the 100 variables, including nonlinear transformations of seven variables, as follows:

$$\text{logit}(p) = 10+11x_1-10\sqrt{x_2}+\frac{2}{x_3}-8\exp(x_4)+7x_5^2-6x_6^{1.5}+5\log(x_7)-4\sin(3.14x_8)+3x_9-2x_{10}$$

Now suppose the true model is not known. With some exploratory data analysis, you determine that the dependency of the logit on some variables is nonlinear. Therefore, you decide to use splines to model this nonlinear dependence. Also, you want to use stepwise regression to remove unimportant variable transformations. The following statements perform the task:

```
proc logistic data=SimuData;
   effect splines = spline(x1-x&nVars/separate);
   model y = splines/selection=stepwise;
   store sasuser.SimuModel;
run;
```

By default, PROC LOGISTIC models the probability that $y = 0$. The EFFECT statement requests an effect named splines constructed by all predictors in the data. The SEPARATE option specifies that the spline basis for each variable be treated as a separate set so that model selection applies to each individual set. The SELECTION=STEPWISE specifies the stepwise regression as the model selection technique. The STORE statement requests that the fitted model be saved to an item store sasuser.SimuModel. See "Example 94.2: Working with Item Stores" on page 7813 for an example with more details about working with item stores.

The spline effect for each predictor produces seven columns in the design matrix, making stepwise regression computationally intensive. For example, a typical Pentium 4 workstation takes around ten minutes to run the preceding statements. Real data sets for classification can be much larger. See examples at UCI Machine Learning Repository (Asuncion and Newman 2007). If new observations about which you want to make predictions are available at model fitting time, you can add the SCORE statement in the LOGISTIC procedure. Consider the case in which observations to predict become available after fitting the model. With PROC PLM, you do not have to repeat the computationally intensive model-fitting processes multiple times. You can use the SCORE statement in the PLM procedure to score new observations based on the item store sasuser.SimuModel that was created during the initial model building. For example, to compute the probability of $y = 0$ for one new observation with all predictor values equal to 0.15 in the data set test, you can use the following statements:

```
data test;
   array x{&nVars};
   do j=1 to &nVars;
      x{j}=0.15;
   end;
   drop j;
   output;
run;
```

```
proc plm restore=sasuser.SimuModel;
   score data=test out=testout predicted / ilink;
run;
```

The ILINK option in the SCORE statement requests that predicted values be inversely transformed to the response scale. In this case, it is the predicted probability of $y = 0$. Output 94.1.1 shows the predicted probability for the new observation.

Output 94.1.1 Predicted Probability for One New Observation

Obs	Predicted
1	0.56649

Example 94.2: Working with Item Stores

(View the complete code for this example (plmex2.sas) in the example repository.)

This example demonstrates how procedures save statistical analysis context and results into item stores and how you can use PROC PLM to make post hoc inference based on saved item stores. The data are taken from McCullagh and Nelder (1989) and concern the effects on taste of various cheese additives. Four cheese additives were tested, and 52 response ratings for each additive were obtained. The response was measured on a scale of nine categories that range from strong dislike (1) to excellent taste (9). The following program saves the data in the data set Cheese. The variable y contains the taste rating, the variable Additive contains cheese additive types, and the variable freq contains the frequencies with which each additive received each rating.

```
data Cheese;
   do Additive = 1 to 4;
      do y = 1 to 9;
         input freq @@;
         output;
      end;
   end;
   label y='Taste Rating';
   datalines;
0  0  1  7  8  8 19  8 1
6  9 12 11  7  6  1  0 0
1  1  6  8 23  7  5  1 0
0  0  0  1  3  7 14 16 11
;
```

The response y is a categorical variable that contains nine ordered levels. You can use PROC LOGISTIC to fit an ordinal model to investigate the effects of the cheese additive types on taste ratings. Suppose you also want to save the ordinal model into an item store so that you can make statistical inference later. You can use the following statements to perform the tasks:

```
proc logistic data=cheese;
   freq freq;
   class additive y / param=glm;
   model y=additive;
```

```
   store sasuser.cheese;
   title 'Ordinal Model on Cheese Additives';
run;
```

By default, PROC LOGISTIC uses the cumulative logit model for the ordered categorical response. The STORE statement saves the fitted model to a SAS item store named sasuser.cheese. The name is a two-level SAS name of the form libname.membername. If libname is not specified in the STORE statement, the fitted results are saved in work.membername and the item store is deleted after the current SAS session ends. With this example, the fitted model is saved to an item store named sasuser.cheese in the Sasuser library. It is not deleted after the current SAS session ends. You can use PROC PLM to restore the results later.

The following statements use PROC PLM to load the saved model context and results by specifying RESTORE= with the target item store sasuser.cheese. Then they use two SHOW statements to display separate information saved in the item store. The first SHOW statement with the PROGRAM option displays the program that was used to generate the item store sasuser.cheese. The second SHOW statement with the PARMS option displays parameter estimates and associated statistics of the fitted ordinal model.

```
proc plm restore=sasuser.cheese;
   show program;
   show parms;
run;
```

Output 94.2.1 displays the program that generated the item store sasuser.cheese. Except for the title information, it matches the original program.

Output 94.2.1 Program Information from sasuser.cheese

Ordinal Model on Cheese Additives

The PLM Procedure

SAS Program Information
proc logistic data=cheese;
freq freq;
class additive y / param=glm;
model y=additive;
store sasuser.cheese;
run;

Output 94.2.2 displays estimates of the intercept terms and covariates and associated statistics. The intercept terms correspond to eight cumulative logits defined on taste ratings; that is, the ith intercept for ith logit is

$$\log\left(\frac{\sum_{j\le i} p_j}{1-\sum_{j\le i} p_j}\right)$$

Output 94.2.2 Parameter Estimates of the Ordinal Model

Parameter Estimates			
Parameter	Taste Rating	Estimate	Standard Error
Intercept	1	-7.0801	0.5624
Intercept	2	-6.0249	0.4755
Intercept	3	-4.9254	0.4272
Intercept	4	-3.8568	0.3902
Intercept	5	-2.5205	0.3431
Intercept	6	-1.5685	0.3086
Intercept	7	-0.06688	0.2658
Intercept	8	1.4930	0.3310
Additive 1		1.6128	0.3778
Additive 2		4.9645	0.4741
Additive 3		3.3227	0.4251
Additive 4		0	.

You can perform various statistical inference tasks from a saved item store, as long as the task is applicable under the model context. For example, you can perform group comparisons between different cheese additive types. See the next example for details.

Example 94.3: Group Comparisons in an Ordinal Model

(View the complete code for this example (plmex3.sas) in the example repository.)

This example continues the study of the effects on taste of various cheese additives. You have finished fitting an ordinal logistic model and saved it to an item store named sasuser.cheese in the previous example. Suppose you want to make comparisons between any pair of cheese additives. You can conduct the analysis by using the ESTIMATE statement and constructing an appropriate **L** matrix, or by using the LSMEANS statement to compute least squares means differences. For an ordinal logistic model with the cumulative logit link, the least squares means are predicted population margins of the cumulative logits. The following statements compute and display differences between least squares means of cheese additive:

```
ods graphics on;
proc plm restore=sasuser.cheese;
   lsmeans additive / cl diff oddsratio plot=diff;
run;
ods graphics off;
```

The LSMEANS statement contains four options. The DIFF option requests least squares means differences for cheese additives. Since the fitted model is an ordinal logistic model with the cumulative logit link, the least squares means differences represent log cumulative odds ratios. The ODDSRATIO option requests exponentiation of the LS-means differences which produces cumulative odds ratios. The CL option constructs confidence limits for the LS-means differences. When ODS Graphics is enabled, the PLOTS=DIFF option requests a display of all pairwise least squares means differences and their significance.

Output 94.3.1 displays the LS-means differences. The reported log odds ratios indicate the relative difference among the cheese additives. A negative log odds ratio indicates that the first category (displayed in the Additive column) having a lower taste rating is less likely than the second category (displayed in the _Additive

column) having a lower taste rating. For example, the log odds ratio between cheese additive 1 and 2 is –3.3517 and the corresponding odds ratio is 0.035. This means the odds of cheese additive 1 receiving a poor rating is 0.035 times the odds of cheese additive 2 receiving a poor rating. In addition to the highly significant p-value (< 0.0001), the confidence limits for both the log odds ratio and the odds ratio indicate that you can reject the null hypothesis that the odds of cheese additive 1 having a lower taste rating is the same as that of cheese additive 2 having a lower rating. Similarly, the odds of cheese additive 2 having a lower rating is 143.241 (with 95% confidence limits (56.558, 362.777)) times the odds of cheese additive 4 having a lower rating. With the same logic, you can conclude that the preference order for the four cheese types from the most favorable to the least favorable is: 4, 1, 3 and 2.

Output 94.3.1 LS-Means Differences of Additive

The PLM Procedure

Differences of Additive Least Squares Means

Additive	_Additive	Estimate	Standard Error	z Value	Pr > |z|	Alpha	Lower	Upper	Odds Ratio	Lower Confidence Limit for Odds Ratio	Upper Confidence Limit for Odds Ratio
1	2	-3.3517	0.4235	-7.91	<.0001	0.05	-4.1818	-2.5216	0.035	0.015	0.080
1	3	-1.7098	0.3731	-4.58	<.0001	0.05	-2.4410	-0.9787	0.181	0.087	0.376
1	4	1.6128	0.3778	4.27	<.0001	0.05	0.8724	2.3532	5.017	2.393	10.520
2	3	1.6419	0.3738	4.39	<.0001	0.05	0.9092	2.3746	5.165	2.482	10.746
2	4	4.9645	0.4741	10.47	<.0001	0.05	4.0353	5.8938	143.241	56.558	362.777
3	4	3.3227	0.4251	7.82	<.0001	0.05	2.4895	4.1558	27.734	12.055	63.805

Output 94.3.2 displays the DiffPlot. This shows that all pairs of LS-means differences, equivalent to log odds ratios in this case, are significant at the level of $\alpha = 0.05$. This means that the preference between any pair of the four cheese additive types are statistically significantly different.

Output 94.3.2 LS-Means Plot of Pairwise Differences

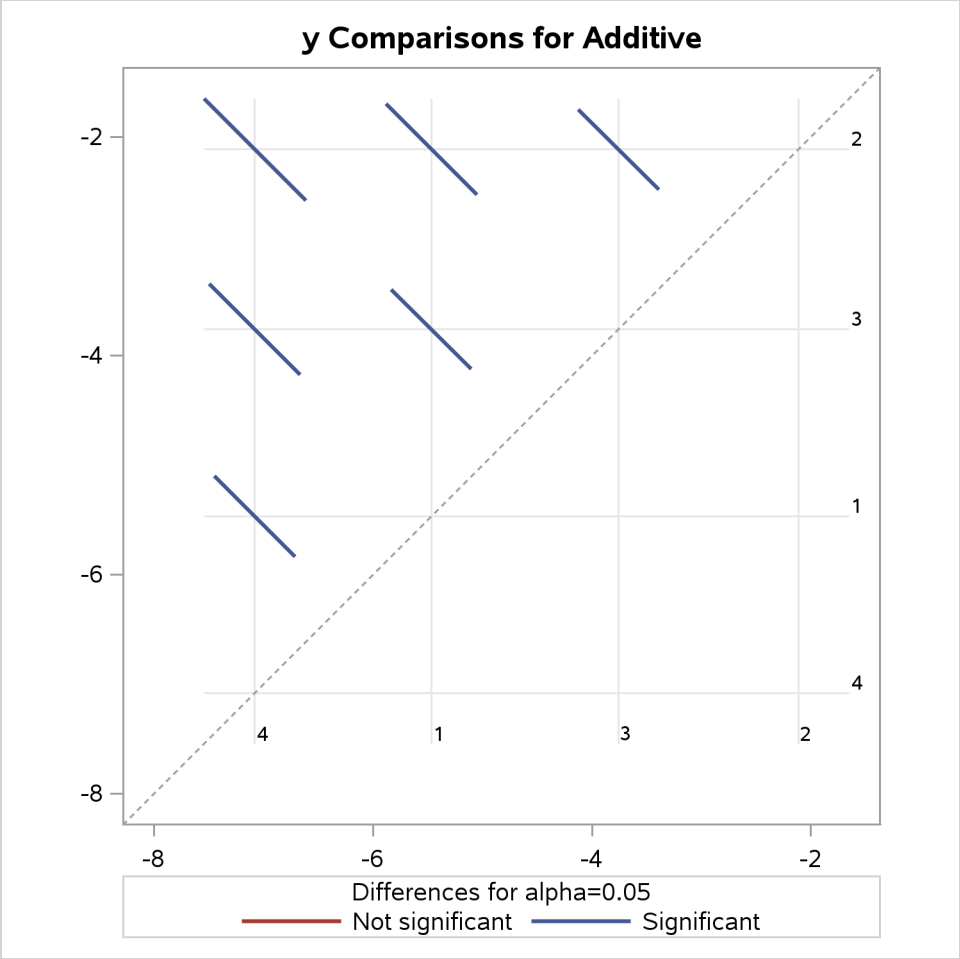

Example 94.4: Posterior Inference for Binomial Data

(View the complete code for this example (plmex4.sas) in the example repository.)

This example demonstrates how you can use PROC PLM to perform posterior inference from a Bayesian analysis. The data for this example are taken from Weisberg (1985) and concern the effect of small electrical currents on farm animals. The ultimate goal of the experiment was to understand the effects of high-voltage power lines on livestock and to better protect farm animals. Seven cows and six shock intensities were used in two experiments. In one experiment, each cow was given 30 electrical shocks with five at each shock intensity in random order. The number of shock responses was recorded for each cow at each shock level. The experiment was then repeated to investigate whether the response diminished due to fatigue of cows, or due to learning. So each cow received a total of 60 shocks. For the following analysis, the cow difference is ignored. The following DATA step lists the data where the variable current represents the shock level, the variable response represents the number of shock responses, the variable trial represents the total number of trials at each shock level, and the variable experiment represents the experiment number (1 for the initial experiment and 2 for the repeated one):

```
data cow;
   input current response trial experiment;
   datalines;
0  0 35 1
0  0 35 2
1  6 35 1
1  3 35 2
2 13 35 1
2  8 35 2
3 26 35 1
3 21 35 2
4 33 35 1
4 27 35 2
5 34 35 1
5 29 35 2
;
```

Suppose you are interested in modeling the distribution of the shock response based on the level of the current and the experiment number. You can use the GENMOD procedure to fit a frequentist logistic model for the data. However, if you have some prior information about parameter estimates, you can fit a Bayesian logistic regression model to take this prior information into account. In this case, suppose you believe the logit of response has a positive association with the shock level but you are uncertain about the ranges of other regression coefficients. To incorporate this prior information in the regression model, you can use the following statements:

```
data prior;
   input _type_$ current;
   datalines;
mean 100
var   50
;

proc genmod data=cow;
   class experiment;
   bayes coeffprior=normal(input=prior) seed=1;
   model response/trial = current|experiment / dist=binomial;
   store cowgmd;
   title 'Bayesian Logistic Model on Cow';
run;
```

The DATA step creates a data set prior that specifies the prior distribution for the variable current, which in this case is a normal distribution with mean 100 and variance 50. This reflects a rough belief in a positive coefficient in a moderate range for current. The prior distribution parameters are not specified for experiment and the interaction between experiment and current, and so PROC GENMOD assigns a default prior for them, which is a normal distribution with mean 0 and variance 1E6.

The BAYES statement in PROC GENMOD specifies that the regression coefficients follow a normal distribution with mean and variance specified in the input data set named prior. It also specifies 1 as the seed for the random number generator in the simulation of the posterior sample. The MODEL statement requests a logistic regression model with a logit link. The STORE statement requests that the fitted results be saved into an item store named cowgmd.

The convergence diagnostics in the output of PROC GENMOD indicate that the Markov chain has converged.

Output 94.4.1 displays summaries on the posterior sample of the regression coefficients. The posterior mean for the intercept is –3.5857 with a 95% HPD interval (−4.5226, −2.6303). The posterior mean of the coefficient for current is 1.1893 with a 95% HPD interval (0.8950, 1.4946), which indicates a positive association between the logit of response and the shock level. Further investigation about whether shock reaction was different between two experiment is warranted.

Output 94.4.1 Posterior Summaries on the Bayesian Logistic Model

Bayesian Logistic Model on Cow

The GENMOD Procedure

Bayesian Analysis

Posterior Summaries

Parameter	N	Mean	Standard Deviation	Percentiles 25%	50%	75%
Intercept	10000	-3.6047	0.4906	-3.9281	-3.5842	-3.2679
current	10000	1.1966	0.1561	1.0873	1.1927	1.2996
experiment1	10000	0.0350	0.7014	-0.4206	0.0347	0.4987
experiment1current	10000	0.3574	0.2503	0.1906	0.3520	0.5235

Posterior Intervals

Parameter	Alpha	Equal-Tail Interval		HPD Interval	
Intercept	0.050	-4.6233	-2.7074	-4.5611	-2.6581
current	0.050	0.9073	1.5152	0.9028	1.5064
experiment1	0.050	-1.3651	1.4004	-1.2995	1.4370
experiment1current	0.050	-0.1293	0.8580	-0.1287	0.8580

Bayesian model fitting typically involves a large amount of simulation. Using the item store and PROC PLM, you do not need to refit the model to perform further posterior inference. Suppose you want to determine whether the shock reaction for the current level is different between the two experiments. You can use PROC PLM with the ESTIMATE statement in the following statements:

```
proc plm restore=cowgmd;
   estimate
   'Diff at current 0' experiment 1 -1 current*experiment [1, 0 1] [-1, 0 2],
   'Diff at current 1' experiment 1 -1 current*experiment [1, 1 1] [-1, 1 2],
   'Diff at current 2' experiment 1 -1 current*experiment [1, 2 1] [-1, 2 2],
   'Diff at current 3' experiment 1 -1 current*experiment [1, 3 1] [-1, 3 2],
   'Diff at current 4' experiment 1 -1 current*experiment [1, 4 1] [-1, 4 2],
   'Diff at current 5' experiment 1 -1 current*experiment [1, 5 1] [-1, 5 2]
    / exp cl;
run;
```

Each line in the ESTIMATE statement compares the fits between the two groups at each current level. The nonpositional syntax is used for the interaction effect current*experiment. For example, the first line requests coefficient 1 for the interaction effect at current level 0 for the initial experiment, and coefficient –1 for the effect at current level 0 for the repeated experiment. The two terms are then added to derive the difference. For more details about the nonpositional syntax, see "Positional and Nonpositional Syntax for Coefficients in

Linear Functions" on page 486 in Chapter 20, "Shared Concepts and Topics."

The EXP option exponentiates log odds ratios to produce odds ratios. The CL option requests that confidence limits be constructed for both log odds ratios and odds ratios. Output 94.4.2 lists the posterior sample estimates for differences between experiments at different current levels.

Output 94.4.2 Comparisons between Experiments at Different Current Levels

Bayesian Logistic Model on Cow

The PLM Procedure

Sample Estimates											
				Percentiles							
Label	N	Estimate	Standard Deviation	25th	50th	75th	Alpha	Lower HPD	Upper HPD	Exponentiated	Standard Deviation of Exponentiated
Diff at current 0	10000	0.03500	0.7014	-0.4206	0.0347	0.4987	0.05	-1.2995	1.4370	1.3258	1.080632
Diff at current 1	10000	0.3924	0.4865	0.0700	0.3884	0.7096	0.05	-0.5287	1.3599	1.6672	0.873185
Diff at current 2	10000	0.7498	0.3266	0.5283	0.7468	0.9719	0.05	0.1300	1.3827	2.2328	0.753450
Diff at current 3	10000	1.1072	0.3194	0.8901	1.1001	1.3220	0.05	0.4772	1.7182	3.1863	1.068673
Diff at current 4	10000	1.4646	0.4718	1.1387	1.4559	1.7756	0.05	0.5369	2.3694	4.8511	2.562732
Diff at current 5	10000	1.8219	0.6844	1.3508	1.8079	2.2601	0.05	0.4881	3.1505	7.8941	6.668085

Sample Estimates					
	Percentiles for Exponentiated				
Label	25th	50th	75th	Lower HPD of Exponentiated	Upper HPD of Exponentiated
Diff at current 0	0.6566	1.0353	1.6466	0.1263	3.3077
Diff at current 1	1.0725	1.4746	2.0331	0.3809	3.3011
Diff at current 2	1.6961	2.1101	2.6429	1.0081	3.7135
Diff at current 3	2.4353	3.0045	3.7509	1.3990	5.1655
Diff at current 4	3.1227	4.2885	5.9040	1.3083	9.6902
Diff at current 5	3.8606	6.0976	9.5838	0.9081	19.4638

The sample statistics are constructed from the posterior sample saved in the item store cowgmd. From the output, the odds of a cow showing shock reaction at level 0 in the initial experiment is 1.2811 (with a 95% HPD interval (0.07387, 3.1001)) times the odds in the repeated experiment. The HPD interval for the odds ratio is constructed based on the mean and variance of the sample of the exponentiated log odds ratios, instead of based on the exponentiated mean and variance of the posterior sample of log odds ratios. The HPD interval suggests that there is not much evidence that the cows responded differently at current level 0 between the two experiments. Similar conclusions can be drawn for current level 1, 2, and 5. However, there is strong evidence that cows responded differently at current level 3 and 4 between the two experiments. The possible explanation is that, if the current level is so small that cows could hardly feel it or the current level is so strong that cows could hardly bear it, cows would respond consistently in the two experiment. If the current level is moderate, cows might get used to it and their response diminished in the repeated experiment.

You can visualize the distribution of the posterior sample of log odds ratios by specifying the PLOTS= option in the ESTIMATE statement. In the following statements, ODS Graphics is enabled by the ODS GRAPHICS ON statement, the PLOTS=BOXPLOT option requests a box plot of posterior distribution of log odds ratios. The suboption ORIENT=HORIZONTAL specifies a horizontal orientation of the boxes.

```
ods graphics on;
proc plm restore=cowgmd;
   estimate
   'Diff at current 0' experiment 1 -1 current*experiment [1, 0 1] [-1, 0 2],
   'Diff at current 1' experiment 1 -1 current*experiment [1, 1 1] [-1, 1 2],
   'Diff at current 2' experiment 1 -1 current*experiment [1, 2 1] [-1, 2 2],
   'Diff at current 3' experiment 1 -1 current*experiment [1, 3 1] [-1, 3 2],
   'Diff at current 4' experiment 1 -1 current*experiment [1, 4 1] [-1, 4 2],
   'Diff at current 5' experiment 1 -1 current*experiment [1, 5 1] [-1, 5 2]
    / plots=boxplot(orient=horizontal);
run;
ods graphics off;
```

Output 94.4.3 displays the box plot of the posterior sample of log odds ratios. The two boxes for differences at current level 3 and 4 show that the corresponding log odds ratios are significantly larger than the reference value $x = 0$. This indicate that there is obvious evidence that the probability of cow response is larger in the initial experiment than in the repeated one at the two current levels. The other four boxes show that the corresponding log odds ratios are not significantly different from 0, which suggests that there is no obvious reaction difference at current level 0, 1, 2, and 5 between the two experiments.

Output 94.4.3 Box Plot of Difference between Two Experiments

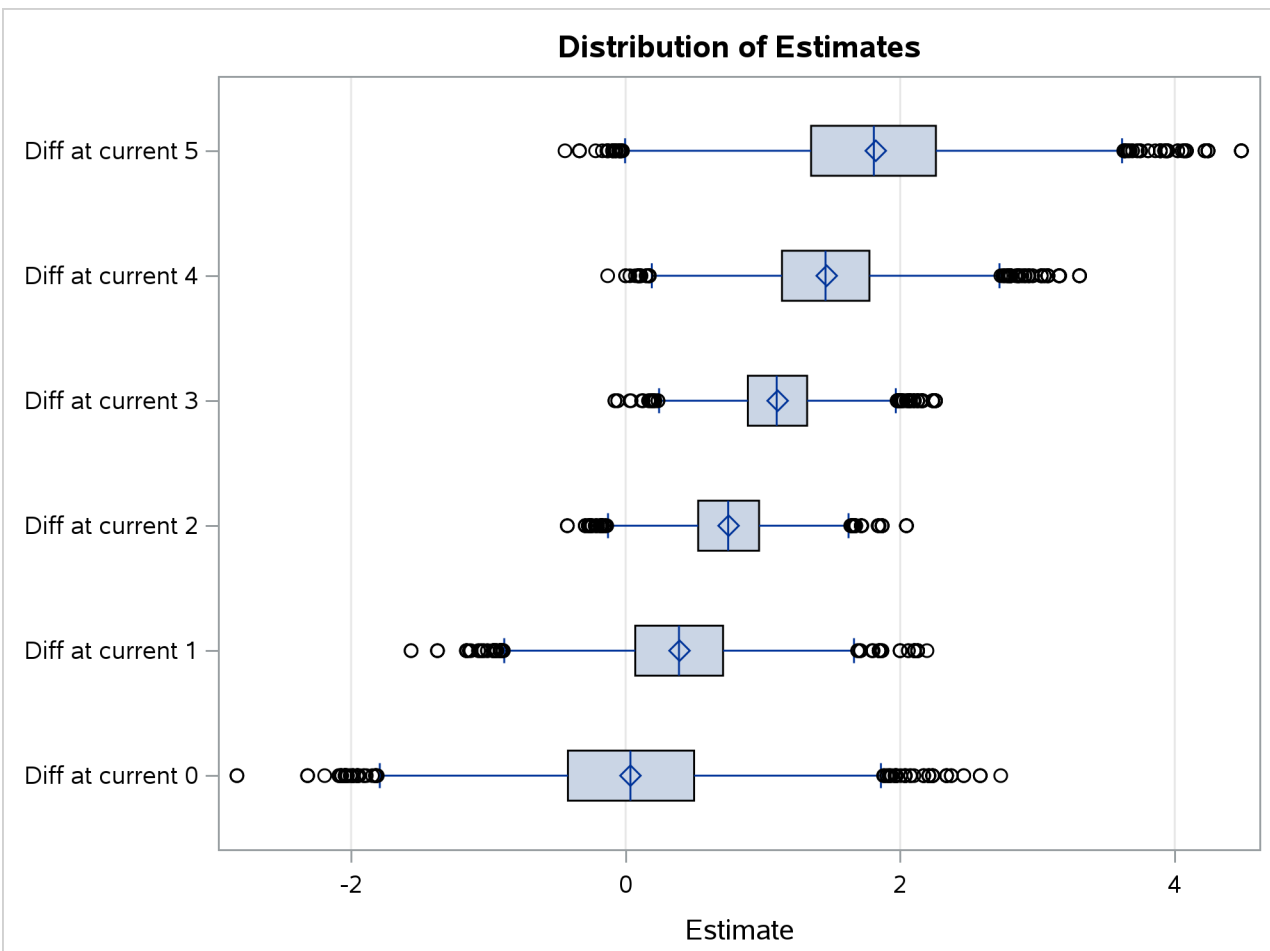

Example 94.5: BY-Group Processing

(View the complete code for this example (plmex5.sas) in the example repository.)

This example uses a data set on a study of the analgesic effects of treatments on elderly patients with neuralgia. The purpose of this example is to show how PROC PLM behaves under different situations when BY-group processing is present. Two test treatments and a placebo are compared to test whether the patient reported pain or not. For each patient, the information of age, gender, and the duration of complaint before the treatment began were recorded. The following DATA step creates the data set named Neuralgia:

```
Data Neuralgia;
   input Treatment $ Sex $ Age Duration Pain $ @@;
   datalines;
P F 68  1 No  B M 74 16 No  P F 67 30 No
P M 66 26 Yes B F 67 28 No  B F 77 16 No
A F 71 12 No  B F 72 50 No  B F 76  9 Yes
A M 71 17 Yes A F 63 27 No  A F 69 18 Yes
B F 66 12 No  A M 62 42 No  P F 64  1 Yes
A F 64 17 No  P M 74  4 No  A F 72 25 No
P M 70  1 Yes B M 66 19 No  B M 59 29 No
A F 64 30 No  A M 70 28 No  A M 69  1 No
B F 78  1 No  P M 83  1 Yes B F 69 42 No
B M 75 30 Yes P M 77 29 Yes P F 79 20 Yes
A M 70 12 No  A F 69 12 No  B F 65 14 No
B M 70  1 No  B M 67 23 No  A M 76 25 Yes
P M 78 12 Yes B M 77  1 Yes B F 69 24 No
P M 66  4 Yes P F 65 29 No  P M 60 26 Yes
A M 78 15 Yes B M 75 21 Yes A F 67 11 No
P F 72 27 No  P F 70 13 Yes A M 75  6 Yes
B F 65  7 No  P F 68 27 Yes P M 68 11 Yes
P M 67 17 Yes B M 70 22 No  A M 65 15 No
P F 67  1 Yes A M 67 10 No  P F 72 11 Yes
A F 74  1 No  B M 80 21 Yes A F 69  3 No
;
```

The data set contains five variables. Treatment is a classification variable that has three levels: A and B represent the two test treatments, and P represents the placebo treatment. Sex is a classification variable that indicates each patient's gender. Age is a continuous variable that indicates the age in years of each patient when a treatment began. Duration is a continuous variable that indicates the duration of complaint in months. The last variable Pain is the response variable with two levels: 'Yes' if pain was reported, 'No' if no pain was reported.

Suppose there is some preliminary belief that the dependency of pain on the explanatory variables is different for male and female patients, leading to separate models between genders. You believe there might be redundant information for predicting the probability of Pain. Thus, you want to perform model selection to eliminate unnecessary effects. You can use the following statements:

```
proc sort data=Neuralgia;
   by sex;
run;

proc logistic data=Neuralgia;
```

```
   class Treatment / param=glm;
   model pain = Treatment Age Duration / selection=backward;
   by sex;
   store painmodel;
   title 'Logistic Model on Neuralgia';
run;
```

PROC SORT is called to sort the data by variable Sex. The LOGISTIC procedure is then called to fit the probability of no pain. Three variables are specified for the full model: Treatment, Age, and Duration. Backward elimination is used as the model selection method. The BY statement fits separate models for male and female patients. Finally, the STORE statement specifies that the fitted results be saved to an item store named painmodel.

Output 94.5.1 lists parameter estimates from the two models after backward elimination is performed. From the model for female patients, Treatment is the only factor that affects the probability of no pain, and Treatment A and B have the same positive effect in predicting the probability of no pain. From the model for male patients, both Treatment and Age are included in the selected model. Treatment A and B have different positive effects, while Age has a negative effect in predicting the probability of no pain.

Output 94.5.1 Parameter Estimates for Male and Female Patients

Logistic Model on Neuralgia

The LOGISTIC Procedure

Sex=F

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	-0.4055	0.6455	0.3946	0.5299
Treatment A	1	2.6027	1.2360	4.4339	0.0352
Treatment B	1	2.6027	1.2360	4.4339	0.0352
Treatment P	0	0	.	.	.

Sex=M

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	20.6178	9.1638	5.0621	0.0245
Treatment A	1	3.9982	1.7333	5.3208	0.0211
Treatment B	1	4.5556	1.9252	5.5993	0.0180
Treatment P	0	0	.	.	.
Age	1	-0.3416	0.1408	5.8869	0.0153

Now the fitted models are saved to the item store painmodel. Suppose you want to use it to score several new observations. The following DATA steps create three data sets for scoring:

```
data score1;
   input Treatment $ Sex $ Age;
   datalines;
A F 20
B F 30
```

```
P F 40
A M 20
B M 30
P M 40
;

data score2;
   set score1(drop=sex);
run;

data score3;
   set score2(drop=Age);
run;
```

The first score data set score1 contains six observations and all the variables that are specified in the full model. The second score data set score2 is a duplicate of score1 except that Sex is dropped. The third score data set score3 is a duplicate of score2 except that Age is dropped. You can use the following statements to score the three data sets:

```
proc plm restore=painmodel;
   score data=score1 out=score1out predicted;
   score data=score2 out=score2out predicted;
   score data=score3 out=score3out predicted;
run;
```

Output 94.5.2 lists the store information that PROC PLM reads from the item store painmodel. The "Model Effects" entry lists all three variables that are specified in the full model before the BY-group processing.

Output 94.5.2 Item Store Information for painmodel

Logistic Model on Neuralgia

The PLM Procedure

Store Information	
Item Store	WORK.PAINMODEL
Data Set Created From	WORK.NEURALGIA
Created By	PROC LOGISTIC
Date Created	16APR20:01:38:40
By Variable	Sex
Response Variable	Pain
Link Function	Logit
Distribution	Binary
Class Variables	Treatment Pain
Model Effects	Intercept Treatment Age Duration

With the three SCORE statements, three data sets are thus produced: score1out, score2out, and score3out. They contain the linear predictors in addition to all original variables. The data set score1out contains the values shown in Output 94.5.3.

Output 94.5.3 Values of Data Set score1out

Logistic Model on Neuralgia

Obs	Treatment	Sex	Age	Predicted
1	A	F	20	2.1972
2	B	F	30	2.1972
3	P	F	40	-0.4055
4	A	M	20	17.7850
5	B	M	30	14.9269
6	P	M	40	6.9557

Linear predictors are computed for all six observations. Because the BY variable Sex is available in score1, PROC PLM uses separate models to score observations of male and female patients. So an observation with the same Treatment and Age has different linear predictors for different genders.

The data set score2out contains the values shown in Output 94.5.4.

Output 94.5.4 Values of Data Set score2out

Logistic Model on Neuralgia

Obs	Sex	Treatment	Age	Predicted
1	F	A	20	2.1972
2	F	B	30	2.1972
3	F	P	40	-0.4055
4	F	A	20	2.1972
5	F	B	30	2.1972
6	F	P	40	-0.4055
7	M	A	20	17.7850
8	M	B	30	14.9269
9	M	P	40	6.9557
10	M	A	20	17.7850
11	M	B	30	14.9269
12	M	P	40	6.9557

The second score data set score2 does not contain the BY variable Sex. PROC PLM continues to score the full data set two times. Each time the scoring is based on the fitted model for each corresponding BY-group. In the output data set, Sex is added at the first column as the BY-group indicator. The first six entries correspond to the model for female patients, and the next six entries correspond to the model for male patients. Age is not included in the first model, and Treatment A and B have the same parameter estimates, so observations 1, 2, 4, and 5 have the same linear predicted value.

The data set score3out contains the values shown in Output 94.5.5.

Output 94.5.5 Values of Data Set score3out

Logistic Model on Neuralgia

Obs	Sex	Treatment	Predicted
1	F	A	2.19722
2	F	B	2.19722
3	F	P	-0.40547
4	F	A	2.19722
5	F	B	2.19722
6	F	P	-0.40547
7	M	A	.
8	M	B	.
9	M	P	.
10	M	A	.
11	M	B	.
12	M	P	.

The third score data set score3 does not contain the BY variable Sex. PROC PLM scores the full data twice with separate models. Furthermore, it does not contain the variable Age, which is a selected variable for predicting the probability of no pain for male patients. Thus, PROC PLM computes linear predictor values for score3 by using the first model for female patients, and sets the linear predictor to missing when using the second model for male patients to score the data set.

Example 94.6: Comparing Multiple B-Splines

(View the complete code for this example (plmex6.sas) in the example repository.)

This example conducts an analysis similar to Example 15 in Chapter 52.33, "Examples: GLIMMIX Procedure." It uses simulated data to perform multiple comparisons among predicted values in a model with group-specific trends that are modeled through regression splines. The estimable functions are formed using nonpositional syntax with constructed effects. Consider the data in the following DATA step. Each of the 100 observations for the continuous response variable y is associated with one of two groups.

```
data spline;
   input group y @@;
   x = _n_;
   datalines;
 1    -.020 1    0.199 2    -1.36 1    -.026
 2    -.397 1    0.065 2    -.861 1    0.251
 1    0.253 2    -.460 2    0.195 2    -.108
 1    0.379 1    0.971 1    0.712 2    0.811
 2    0.574 2    0.755 1    0.316 2    0.961
 2    1.088 2    0.607 2    0.959 1    0.653
 1    0.629 2    1.237 2    0.734 2    0.299
 2    1.002 2    1.201 1    1.520 1    1.105
 1    1.329 1    1.580 2    1.098 1    1.613
 2    1.052 2    1.108 2    1.257 2    2.005
 2    1.726 2    1.179 2    1.338 1    1.707
 2    2.105 2    1.828 2    1.368 1    2.252
```

```
 1    1.984 2    1.867 1    2.771 1    2.052
 2    1.522 2    2.200 1    2.562 1    2.517
 1    2.769 1    2.534 2    1.969 1    2.460
 1    2.873 1    2.678 1    3.135 2    1.705
 1    2.893 1    3.023 1    3.050 2    2.273
 2    2.549 1    2.836 2    2.375 2    1.841
 1    3.727 1    3.806 1    3.269 1    3.533
 1    2.948 2    1.954 2    2.326 2    2.017
 1    3.744 2    2.431 2    2.040 1    3.995
 2    1.996 2    2.028 2    2.321 2    2.479
 2    2.337 1    4.516 2    2.326 2    2.144
 2    2.474 2    2.221 1    4.867 2    2.453
 1    5.253 2    3.024 2    2.403 1    5.498
;
```

The following statements fit a model with separate trends for the two groups; the trends are modeled as B-splines.

```
proc orthoreg data=spline;
   class group;
   effect spl = spline(x);
   model y = group spl*group / noint;
   store ortho_spline;
   title 'B-splines Comparisons';
run;
```

Results from this analysis are shown in Output 94.6.1. The "Parameter Estimates" table shows the estimates for the spline coefficients in the two groups.

Output 94.6.1 Results for Group-Specific Spline Model

B-splines Comparisons

The ORTHOREG Procedure

Dependent Variable: y

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	13	153.0175561	11.77058124	160.11	<.0001
Error	86	6.322380412	0.0735160513		
Corrected Total	99	159.3399365			

Root MSE	0.2711384357
R-Square	0.9603214326

Output 94.6.1 *continued*

Parameter	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
(group='1')	1	9.702654639620	3.134189999	3.10	0.0026
(group='2')	1	6.306192205636	2.629914777	2.40	0.0187
spl*group 1 1	1	-11.178645171804	3.700809740	-3.02	0.0033
spl*group 1 2	1	-20.194609274614	3.976504624	-5.08	<.0001
spl*group 2 1	1	-9.532736979953	3.257583205	-2.93	0.0044
spl*group 2 2	1	-5.856524965350	2.790611677	-2.10	0.0388
spl*group 3 1	1	-8.961183718933	3.071750881	-2.92	0.0045
spl*group 3 2	1	-5.556716052452	2.571671557	-2.16	0.0335
spl*group 4 1	1	-7.261532314788	3.243690314	-2.24	0.0278
spl*group 4 2	1	-4.367788897382	2.724680959	-1.60	0.1126
spl*group 5 1	1	-6.446152565109	2.961695536	-2.18	0.0323
spl*group 5 2	1	-4.038016189149	2.458883913	-1.64	0.1042
spl*group 6 1	1	-4.638169590941	3.709463632	-1.25	0.2146
spl*group 6 2	1	-4.302901043951	3.047854017	-1.41	0.1616
spl*group 7 1	0	0	.	.	.
spl*group 7 2	0	0	.	.	.

By default, the ORTHOREG procedure constructs B-splines with seven knots. Since B-spline coefficients satisfy a sum-to-one constraint and since the model contains group-specific intercepts, the last spline coefficient for each group is redundant and is set to 0.

The following statements make a prediction for the input data set by using the SCORE statement with PROC PLM and graph the observed and predicted values in the two groups:

```
proc plm restore=ortho_spline;
   score data=spline out=ortho_pred predicted=p;
run;

proc sgplot data=ortho_pred;
   series  y=p x=x / group=group name="fit";
   scatter y=y x=x / group=group;
   keylegend "fit" / title="Group";
run;
```

The prediction plot in Output 94.6.2 suggests that there is some separation of the group trends for small values of x and for values that exceed about $x = 40$.

Output 94.6.2 Observed Data and Predicted Values by Group

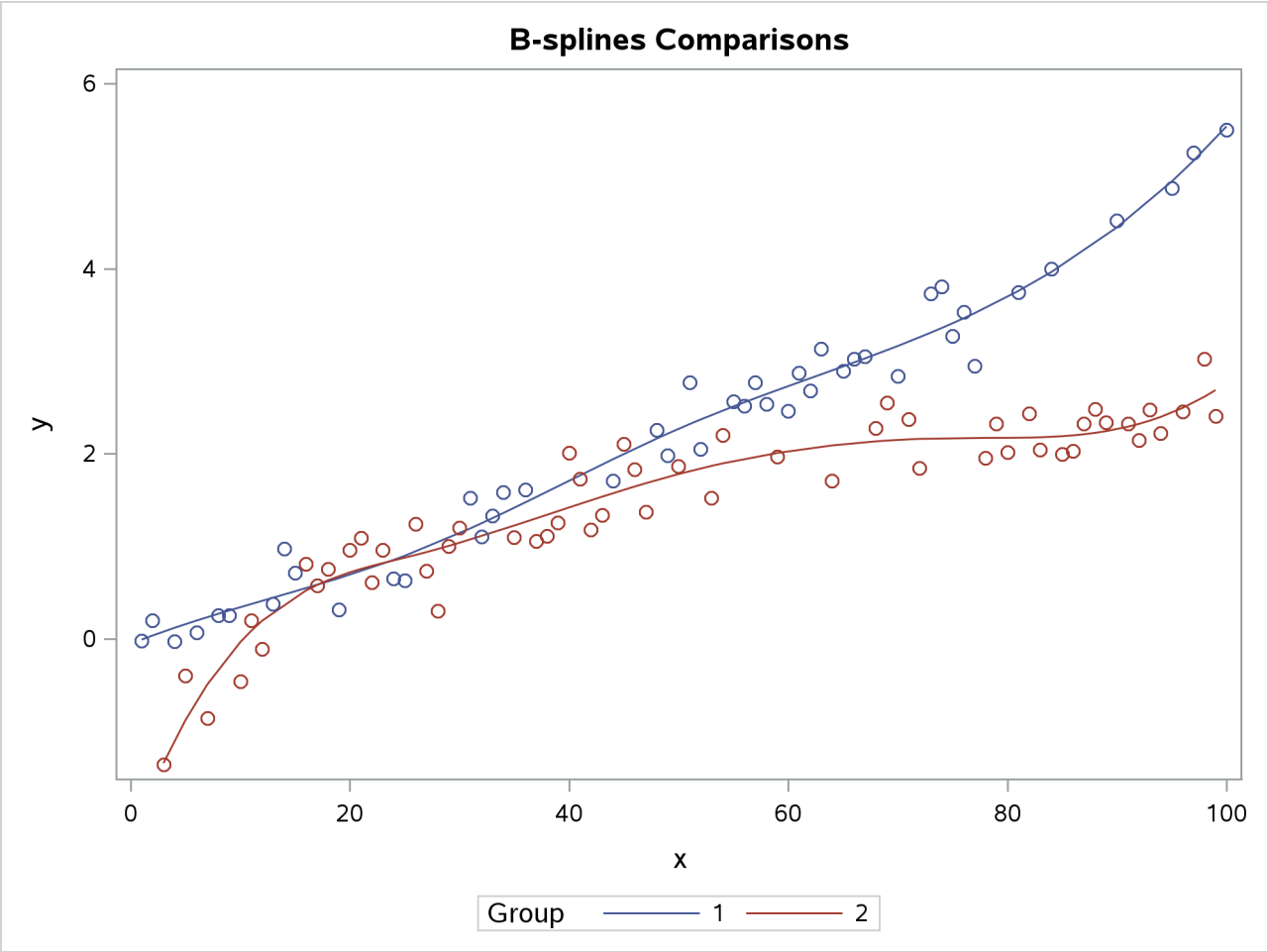

In order to determine the range on which the trends separate significantly, the PLM procedure is executed in the following statements with an ESTIMATE statement that applies group comparisons at a number of values for the spline variable x:

```
%macro GroupDiff;
   %do x=0 %to 75 %by 5;
      "Diff at x=&x" group 1 -1 group*spl [1,1  &x] [-1,2  &x],
   %end;
   'Diff at x=80' group 1 -1 group*spl [1,1 80] [-1,2 80]
%mend;

proc plm restore=ortho_spline;
   show effects;
   estimate %GroupDiff / adjust=simulate seed=1 stepdown;
run;
```

For example, the following ESTIMATE statement compares the trends between the two groups at $x = 25$:

```
estimate 'Diff at x=25' group 1 -1 group*spl [1,1 25] [-1,2 25];
```

The nonpositional syntax is used for the group*spl effect. For example, the specification [−1, 2 25] requests that the spline be computed at $x = 25$ for the second level of variable group. The resulting coefficients are added to the L vector for the estimate after being multiplied with –1.

Because comparisons are made at a large number of values for x, a multiplicity correction is in order to adjust the p-values to reflect familywise error control. Simulated p-values with step-down adjustment are used here.

Output 94.6.3 displays the "Store Information" for the item store and information about the spline effect (the result of the SHOW statement).

Output 94.6.3 Spline Details

B-splines Comparisons

The PLM Procedure

Store Information	
Item Store	WORK.ORTHO_SPLINE
Data Set Created From	WORK.SPLINE
Created By	PROC ORTHOREG
Date Created	16APR20:01:38:44
Response Variable	y
Class Variable	group
Constructed Effect	spl
Model Effects	group spl*group

B-splines Comparisons

The PLM Procedure

Knots for Spline Effect spl		
Knot Number	**Boundary**	**x**
1	*	-48.50000
2	*	-23.75000
3	*	1.00000
4		25.75000
5		50.50000
6		75.25000
7	*	100.00000
8	*	124.75000
9	*	149.50000

Output 94.6.3 *continued*

B-splines Comparisons

The PLM Procedure

Basis Details for Spline Effect spl			
Column	Support		Support Knots
1	-48.50000	25.75000	1-4
2	-48.50000	50.50000	1-5
3	-23.75000	75.25000	2-6
4	1.00000	100.00000	3-7
5	25.75000	124.75000	4-8
6	50.50000	149.50000	5-9
7	75.25000	149.50000	6-9

Output 94.6.4 displays the results from the ESTIMATE statement.

Output 94.6.4 Estimate Results with Multiplicity Correction

Estimates Adjustment for Multiplicity: Holm-Simulated						
Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Adj P
Diff at x=0	1.8809	0.3640	86	5.17	<.0001	<.0001
Diff at x=5	1.0376	0.1759	86	5.90	<.0001	<.0001
Diff at x=10	0.3778	0.1540	86	2.45	0.0162	0.0551
Diff at x=15	0.05822	0.1481	86	0.39	0.6952	0.9102
Diff at x=20	-0.02602	0.1243	86	-0.21	0.8346	0.9557
Diff at x=25	0.02014	0.1312	86	0.15	0.8783	0.9557
Diff at x=30	0.1023	0.1378	86	0.74	0.4600	0.7426
Diff at x=35	0.1924	0.1236	86	1.56	0.1231	0.2965
Diff at x=40	0.2883	0.1114	86	2.59	0.0113	0.0453
Diff at x=45	0.3877	0.1195	86	3.24	0.0017	0.0085
Diff at x=50	0.4885	0.1308	86	3.74	0.0003	0.0025
Diff at x=55	0.5903	0.1231	86	4.79	<.0001	<.0001
Diff at x=60	0.7031	0.1125	86	6.25	<.0001	<.0001
Diff at x=65	0.8401	0.1203	86	6.99	<.0001	<.0001
Diff at x=70	1.0147	0.1348	86	7.52	<.0001	<.0001
Diff at x=75	1.2400	0.1326	86	9.35	<.0001	<.0001
Diff at x=80	1.5237	0.1281	86	11.89	<.0001	<.0001

Notice that the "Store Information" in Output 94.6.3 displays the classification variables (from the CLASS statement in PROC ORTHOREG), the constructed effects (from the EFFECT statement in PROC ORTHOREG), and the model effects (from the MODEL statement in PROC ORTHOREG). Output 94.6.4 shows that at the 5% significance level the trends are significantly different for $x \leq 10$ and for $x \geq 40$. Between those values you cannot reject the hypothesis of trend congruity.

To see this effect more clearly, you can filter the results by adding the a filtering statement to the previous PROC PLM step:

```
proc plm restore=ortho_spline;
   estimate %GroupDiff / adjust=simulate seed=1 stepdown;
   filter adjp > 0.05;
run;
```

This produces Output 94.6.5, which displays the subset of the results in Output 94.6.4 that meets the condition in the FILTER expression.

Output 94.6.5 Filtered Estimate Results

B-splines Comparisons

The PLM Procedure

Estimates Adjustment for Multiplicity: Holm-Simulated						
Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Adj P
Diff at x=10	0.3778	0.1540	86	2.45	0.0162	0.0551
Diff at x=15	0.05822	0.1481	86	0.39	0.6952	0.9102
Diff at x=20	-0.02602	0.1243	86	-0.21	0.8346	0.9557
Diff at x=25	0.02014	0.1312	86	0.15	0.8783	0.9557
Diff at x=30	0.1023	0.1378	86	0.74	0.4600	0.7426
Diff at x=35	0.1924	0.1236	86	1.56	0.1231	0.2965

Example 94.7: Linear Inference with Arbitrary Estimates

(View the complete code for this example (plmex7.sas) in the example repository.)

Suppose that you have calculated a vector of parameter estimates of dimension $(p \times 1)$ and its associated variance-covariance matrix by some statistical method. You might want to use these results to perform linear inference, or to score a data set and calculate predicted values and their standard errors.

The following DATA steps create two SAS data sets. The first, called parms, contains six estimates that represent two uncorrelated groups. The data set cov contains the covariance matrix of the estimates. The lack of correlation between the two sets of three parameters is evident in the block-diagonal structure of the covariance matrix.

```
data parms;
   length name $6;
   input Name$ Value;
   datalines;
alpha1 -3.5671
beta1   0.4421
gamma1 -2.6230
alpha2 -3.0111
beta2   0.3977
gamma2 -2.4442
;
```

```
data cov;
   input Parm row col1-col6;
   datalines;
1 1  0.007462 -0.005222  0.010234  0.000000  0.000000  0.000000
1 2 -0.005222  0.048197 -0.010590  0.000000  0.000000  0.000000
1 3  0.010234 -0.010590  0.215999  0.000000  0.000000  0.000000
1 4  0.000000  0.000000  0.000000  0.031261 -0.009096  0.015785
1 5  0.000000  0.000000  0.000000 -0.009096  0.039487 -0.019996
1 6  0.000000  0.000000  0.000000  0.015785 -0.019996  0.126172
;
```

Suppose that you are interested in testing whether the parameters are homogeneous across groups—that is, whether $\alpha_1 = \alpha_2, \beta_1 = \beta_2, \gamma_1 = \gamma 2$. You are interested in testing the hypothesis jointly and separately with multiplicity adjustment.

To use the PLM procedure, you first need to create an item store that contains the necessary information as if the preceding parameter vector and covariance matrix were the result of a statistical modeling procedure. The following statements use the GLIMMIX procedure to create such an item store, by fitting a saturated linear model with the data set that contains the parameter estimates serving as the input data set:

```
proc glimmix data=parms order=data;
   class Name;
   model Value = Name / noint ddfm=none s;
   random _residual_ / type=lin(1) ldata=cov v;
   parms (1) / noiter;
   store ArtificialModel;
   title 'Linear Inference';
run;
```

The RANDOM statement is used to form the covariance structure for the estimates. The PARMS statement prevents iterative updates of the covariance parameters. The resulting marginal covariance matrix of the "data" is thus identical to the covariance matrix in the data set cov. The ORDER=DATA option in the PROC GLIMMIX statement is used to arrange the levels of the classification variable Name in the order in which they appear in the data set so that the order of the parameters matches that of the covariance matrix.

The results of this analysis are shown in Output 94.7.1. Note that the parameter estimates are identical to the values passed in the input data set and their standard errors equal the square root of the diagonal elements of the cov data set.

Output 94.7.1 "Fitted" Parameter Estimates and Covariance Matrix

Linear Inference

The GLIMMIX Procedure

Estimated V Matrix for Subject 1						
Row	**Col1**	**Col2**	**Col3**	**Col4**	**Col5**	**Col6**
1	0.007462	-0.00522	0.01023			
2	-0.00522	0.04820	-0.01059			
3	0.01023	-0.01059	0.2160			
4				0.03126	-0.00910	0.01579
5				-0.00910	0.03949	-0.02000
6				0.01579	-0.02000	0.1262

Output 94.7.1 *continued*

Solutions for Fixed Effects						
Effect	name	Estimate	Standard Error	DF	t Value	Pr > |t|
name	alpha1	-3.5671	0.08638	Infty	-41.29	<.0001
name	beta1	0.4421	0.2195	Infty	2.01	0.0440
name	gamma1	-2.6230	0.4648	Infty	-5.64	<.0001
name	alpha2	-3.0111	0.1768	Infty	-17.03	<.0001
name	beta2	0.3977	0.1987	Infty	2.00	0.0454
name	gamma2	-2.4442	0.3552	Infty	-6.88	<.0001

There are other ways to fit a saturated model with the GLIMMIX procedure. For example, you can use the TYPE=UN covariance structure in the RANDOM statement with a properly prepared input data set for the PDATA= option in the PARMS statement. See Example 17 in Chapter 52.33, "Examples: GLIMMIX Procedure," for details.

Once the item store exists, you can apply the linear inference capabilities of the PLM procedure. For example, the ESTIMATE statement in the following statements test the hypothesis of parameter homogeneity across groups:

```
proc plm restore=ArtificialModel;
   estimate
       'alpha1 = alpha2' Name 1  0  0 -1  0  0,
       'beta1  = beta2 ' Name 0  1  0  0 -1  0,
       'gamma1 = gamma2' Name 0  0  1  0  0 -1 /
                adjust=bon stepdown ftest(label='Homogeneity');
run;
```

Output 94.7.2 Results from the PLM Procedure

Linear Inference

The PLM Procedure

Estimates Adjustment for Multiplicity: Holm						
Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Adj P
alpha1 = alpha2	-0.5560	0.1968	Infty	-2.83	0.0047	0.0142
beta1 = beta2	0.04440	0.2961	Infty	0.15	0.8808	1.0000
gamma1 = gamma2	-0.1788	0.5850	Infty	-0.31	0.7599	1.0000

F Test for Estimates				
Label	Num DF	Den DF	F Value	Pr > F
Homogeneity	3	Infty	2.79	0.0389

The F test in Output 94.7.2 shows that the joint test of homogeneity is rejected. The individual tests with familywise control of the Type I error show that the overall difference is due to a significant change in the α parameters. The hypothesis of homogeneity across the two groups cannot be rejected for the β and γ parameters.

References

Asuncion, A., and Newman, D. J. (2007). "UCI Machine Learning Repository." http://archive.ics.uci.edu/ml/.

Kenward, M. G., and Roger, J. H. (1997). "Small Sample Inference for Fixed Effects from Restricted Maximum Likelihood." *Biometrics* 53:983–997.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Silvapulle, M. J., and Sen, P. K. (2004). *Constrained Statistical Inference: Order, Inequality, and Shape Constraints*. New York: John Wiley & Sons.

Weisberg, S. (1985). *Applied Linear Regression*. 2nd ed. New York: John Wiley & Sons.

Chapter 95
The PLS Procedure

Contents

Overview: PLS Procedure . . . **7838**
- Basic Features . . . 7838

Getting Started: PLS Procedure . . . **7839**
- Spectrometric Calibration . . . 7839

Syntax: PLS Procedure . . . **7848**
- PROC PLS Statement . . . 7848
- BY Statement . . . 7855
- CLASS Statement . . . 7855
- EFFECT Statement . . . 7856
- ID Statement . . . 7857
- MODEL Statement . . . 7858
- OUTPUT Statement . . . 7858

Details: PLS Procedure . . . **7859**
- Regression Methods . . . 7859
- Cross Validation . . . 7864
- Centering and Scaling . . . 7865
- Missing Values . . . 7866
- Displayed Output . . . 7866
- ODS Table Names . . . 7867
- ODS Graphics . . . 7868

Examples: PLS Procedure . . . **7870**
- Example 95.1: Examining Model Details . . . 7870
- Example 95.2: Examining Outliers . . . 7877
- Example 95.3: Choosing a PLS Model by Test Set Validation . . . 7879
- Example 95.4: Partial Least Squares Spline Smoothing . . . 7885

References . . . **7891**

Overview: PLS Procedure

The PLS procedure fits models by using any one of a number of linear predictive methods, including *partial least squares* (PLS). Ordinary least squares regression, as implemented in SAS/STAT procedures such as PROC GLM and PROC REG, has the single goal of minimizing sample response prediction error, seeking linear functions of the predictors that explain as much variation in each response as possible. The techniques implemented in the PLS procedure have the additional goal of accounting for variation in the predictors, under the assumption that directions in the predictor space that are well sampled should provide better prediction for *new* observations when the predictors are highly correlated. All of the techniques implemented in the PLS procedure work by extracting successive linear combinations of the predictors, called *factors* (also called *components*, *latent vectors*, or *latent variables*), which optimally address one or both of these two goals—explaining response variation and explaining predictor variation. In particular, the method of partial least squares balances the two objectives, seeking factors that explain both response and predictor variation.

Note that the name "partial least squares" also applies to a more general statistical method that is *not* implemented in this procedure. The partial least squares method was originally developed in the 1960s by the econometrician Herman Wold (1966) for modeling "paths" of causal relation between any number of "blocks" of variables. However, the PLS procedure fits only *predictive* partial least squares models, with one "block" of predictors and one "block" of responses. If you are interested in fitting more general path models, you should consider using the CALIS procedure.

Basic Features

The techniques implemented by the PLS procedure are as follows:

- principal components regression, which extracts factors to explain as much predictor sample variation as possible
- reduced rank regression, which extracts factors to explain as much response variation as possible. This technique, also known as (maximum) redundancy analysis, differs from multivariate linear regression only when there are multiple responses.
- partial least squares regression, which balances the two objectives of explaining response variation and explaining predictor variation. Two different formulations for partial least squares are available: the original predictive method of Wold (1966) and the SIMPLS method of De Jong (1993).

The number of factors to extract depends on the data. Basing the model on more extracted factors improves the model fit to the observed data, but extracting too many factors can cause *overfitting*—that is, tailoring the model too much to the current data, to the detriment of future predictions. The PLS procedure enables you to choose the number of extracted factors by *cross validation*—that is, fitting the model to part of the data, minimizing the prediction error for the unfitted part, and iterating with different portions of the data in the roles of fitted and unfitted. Various methods of cross validation are available, including one-at-a-time validation and splitting the data into blocks. The PLS procedure also offers test set validation, where the model is fit to the entire primary input data set and the fit is evaluated over a distinct test data set.

You can use the general linear modeling approach of the GLM procedure to specify a model for your design, allowing for general polynomial effects as well as classification or ANOVA effects. You can save the model fit by the PLS procedure in a data set and apply it to new data by using the SCORE procedure.

The PLS procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the PLS procedure, see the PLOTS options in the PROC PLS statements and the section "ODS Graphics" on page 7868.

Getting Started: PLS Procedure

Spectrometric Calibration

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/plsgs.sas.)

The example in this section illustrates basic features of the PLS procedure. The data are reported in Umetrics (1995); the original source is Lindberg, Persson, and Wold (1983). Suppose that you are researching pollution in the Baltic Sea, and you would like to use the spectra of samples of seawater to determine the amounts of three compounds present in samples from the Baltic Sea: lignin sulfonate (ls: pulp industry pollution), humic acids (ha: natural forest products), and optical whitener from detergent (dt). Spectrometric calibration is a type of problem in which partial least squares can be very effective. The predictors are the spectra emission intensities at different frequencies in sample spectrum, and the responses are the amounts of various chemicals in the sample.

For the purposes of calibrating the model, samples with known compositions are used. The calibration data consist of 16 samples of known concentrations of ls, ha, and dt, with spectra based on 27 frequencies (or, equivalently, wavelengths). The following statements create a SAS data set named Sample for these data.

```
data Sample;
   input obsnam $ v1-v27 ls ha dt @@;
   datalines;
EM1   2766 2610 3306 3630 3600 3438 3213 3051 2907 2844 2796
      2787 2760 2754 2670 2520 2310 2100 1917 1755 1602 1467
      1353 1260 1167 1101 1017         3.0110  0.0000   0.00
EM2   1492 1419 1369 1158  958  887  905  929  920  887  800
       710  617  535  451  368  296  241  190  157  128  106
        89   70   65   56   50         0.0000  0.4005   0.00
EM3   2450 2379 2400 2055 1689 1355 1109  908  750  673  644
       640  630  618  571  512  440  368  305  247  196  156
       120   98   80   61   50         0.0000  0.0000  90.63
EM4   2751 2883 3492 3570 3282 2937 2634 2370 2187 2070 2007
      1974 1950 1890 1824 1680 1527 1350 1206 1080  984  888
       810  732  669  630  582         1.4820  0.1580  40.00
EM5   2652 2691 3225 3285 3033 2784 2520 2340 2235 2148 2094
      2049 2007 1917 1800 1650 1464 1299 1140 1020  909  810
       726  657  594  549  507         1.1160  0.4104  30.45
EM6   3993 4722 6147 6720 6531 5970 5382 4842 4470 4200 4077
```

```
      4008 3948 3864 3663 3390 3090 2787 2481 2241 2028 1830
      1680 1533 1440 1314 1227         3.3970  0.3032  50.82
EM7   4032 4350 5430 5763 5490 4974 4452 3990 3690 3474 3357
      3300 3213 3147 3000 2772 2490 2220 1980 1779 1599 1440
      1320 1200 1119 1032  957         2.4280  0.2981  70.59
EM8   4530 5190 6910 7580 7510 6930 6150 5490 4990 4670 4490
      4370 4300 4210 4000 3770 3420 3060 2760 2490 2230 2060
      1860 1700 1590 1490 1380         4.0240  0.1153  89.39
EM9   4077 4410 5460 5857 5607 5097 4605 4170 3864 3708 3588
      3537 3480 3330 3192 2910 2610 2325 2064 1830 1638 1476
      1350 1236 1122 1044  963         2.2750  0.5040  81.75
EM10  3450 3432 3969 4020 3678 3237 2814 2487 2205 2061 2001
      1965 1947 1890 1776 1635 1452 1278 1128  981  867  753
       663  600  552  507  468         0.9588  0.1450 101.10
EM11  4989 5301 6807 7425 7155 6525 5784 5166 4695 4380 4197
      4131 4077 3972 3777 3531 3168 2835 2517 2244 2004 1809
      1620 1470 1359 1266 1167         3.1900  0.2530 120.00
EM12  5340 5790 7590 8390 8310 7670 6890 6190 5700 5380 5200
      5110 5040 4900 4700 4390 3970 3540 3170 2810 2490 2240
      2060 1870 1700 1590 1470         4.1320  0.5691 117.70
EM13  3162 3477 4365 4650 4470 4107 3717 3432 3228 3093 3009
      2964 2916 2838 2694 2490 2253 2013 1788 1599 1431 1305
      1194 1077  990  927  855         2.1600  0.4360  27.59
EM14  4380 4695 6018 6510 6342 5760 5151 4596 4200 3948 3807
      3720 3672 3567 3438 3171 2880 2571 2280 2046 1857 1680
      1548 1413 1314 1200 1119         3.0940  0.2471  61.71
EM15  4587 4200 5040 5289 4965 4449 3939 3507 3174 2970 2850
      2814 2748 2670 2529 2328 2088 1851 1641 1431 1284 1134
      1020  918  840  756  714         1.6040  0.2856 108.80
EM16  4017 4725 6090 6570 6354 5895 5346 4911 4611 4422 4314
      4287 4224 4110 3915 3600 3240 2913 2598 2325 2088 1917
      1734 1587 1452 1356 1257         3.1620  0.7012  60.00
;
```

Fitting a PLS Model

To isolate a few underlying spectral factors that provide a good predictive model, you can fit a PLS model to the 16 samples by using the following SAS statements:

```
proc pls data=sample;
   model ls ha dt = v1-v27;
run;
```

By default, the PLS procedure extracts at most 15 factors. The procedure lists the amount of variation accounted for by each of these factors, both individual and cumulative; this listing is shown in Figure 95.1.

Figure 95.1 PLS Variation Summary

The PLS Procedure

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	97.4607	97.4607	41.9155	41.9155
2	2.1830	99.6436	24.2435	66.1590
3	0.1781	99.8217	24.5339	90.6929
4	0.1197	99.9414	3.7898	94.4827
5	0.0415	99.9829	1.0045	95.4873
6	0.0106	99.9935	2.2808	97.7681
7	0.0017	99.9952	1.1693	98.9374
8	0.0010	99.9961	0.5041	99.4415
9	0.0014	99.9975	0.1229	99.5645
10	0.0010	99.9985	0.1103	99.6747
11	0.0003	99.9988	0.1523	99.8270
12	0.0003	99.9991	0.1291	99.9561
13	0.0002	99.9994	0.0312	99.9873
14	0.0004	99.9998	0.0065	99.9938
15	0.0002	100.0000	0.0062	100.0000

Note that all of the variation in both the predictors and the responses is accounted for by only 15 factors; this is because there are only 16 sample observations. More important, almost all of the variation is accounted for with even fewer factors—one or two for the predictors and three to eight for the responses.

Selecting the Number of Factors by Cross Validation

A PLS model is not complete until you choose the number of factors. You can choose the number of factors by using cross validation, in which the data set is divided into two or more groups. You fit the model to all groups except one, and then you check the capability of the model to predict responses for the group omitted. Repeating this for each group, you then can measure the overall capability of a given form of the model. The predicted residual sum of squares (PRESS) statistic is based on the residuals generated by this process.

To select the number of extracted factors by cross validation, you specify the CV= option with an argument that says which cross validation method to use. For example, a common method is split-sample validation, in which the different groups are composed of every *n*th observation beginning with the first, every *n*th observation beginning with the second, and so on. You can use the CV=SPLIT option to specify split-sample validation with $n = 7$ by default, as in the following SAS statements:

```
proc pls data=sample cv=split;
   model ls ha dt = v1-v27;
run;
```

The resulting output is shown in Figure 95.2 and Figure 95.3.

Figure 95.2 Split-Sample Validated PRESS Statistics for Number of Factors

The PLS Procedure

Split-sample Validation for the Number of Extracted Factors	
Number of Extracted Factors	Root Mean PRESS
0	1.107747
1	0.957983
2	0.931314
3	0.520222
4	0.530501
5	0.586786
6	0.475047
7	0.477595
8	0.483138
9	0.485739
10	0.48946
11	0.521445
12	0.525653
13	0.531049
14	0.531049
15	0.531049

Minimum root mean PRESS	0.4750
Minimizing number of factors	6

Figure 95.3 PLS Variation Summary for Split-Sample Validated Model

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	97.4607	97.4607	41.9155	41.9155
2	2.1830	99.6436	24.2435	66.1590
3	0.1781	99.8217	24.5339	90.6929
4	0.1197	99.9414	3.7898	94.4827
5	0.0415	99.9829	1.0045	95.4873
6	0.0106	99.9935	2.2808	97.7681

The absolute minimum PRESS is achieved with six extracted factors. Notice, however, that this is not much smaller than the PRESS for three factors. By using the CVTEST option, you can perform a statistical model comparison suggested by Van der Voet (1994) to test whether this difference is significant, as shown in the

following SAS statements:

```
proc pls data=sample cv=split cvtest(seed=12345);
   model ls ha dt = v1-v27;
run;
```

The model comparison test is based on a rerandomization of the data. By default, the seed for this randomization is based on the system clock, but it is specified here. The resulting output is shown in Figure 95.4 and Figure 95.5.

Figure 95.4 Testing Split-Sample Validation for Number of Factors

The PLS Procedure

Split-sample Validation for the Number of Extracted Factors

Number of Extracted Factors	Root Mean PRESS	T**2	Prob > T**2
0	1.107747	9.272858	0.0010
1	0.957983	10.62305	<.0001
2	0.931314	8.950878	0.0010
3	0.520222	5.133259	0.1440
4	0.530501	5.168427	0.1340
5	0.586786	6.437266	0.0150
6	0.475047	0	1.0000
7	0.477595	2.809763	0.4750
8	0.483138	7.189526	0.0110
9	0.485739	7.931726	0.0060
10	0.48946	6.612597	0.0150
11	0.521445	6.666235	0.0130
12	0.525653	7.092861	0.0080
13	0.531049	7.538298	0.0030
14	0.531049	7.538298	0.0030
15	0.531049	7.538298	0.0030

Minimum root mean PRESS	0.4750
Minimizing number of factors	6
Smallest number of factors with p > 0.1	3

Figure 95.5 PLS Variation Summary for Tested Split-Sample Validated Model

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	97.4607	97.4607	41.9155	41.9155
2	2.1830	99.6436	24.2435	66.1590
3	0.1781	99.8217	24.5339	90.6929

The p-value of 0.1430 in comparing the cross validated residuals from models with 6 and 3 factors indicates that the difference between the two models is insignificant; therefore, the model with fewer factors is preferred. The variation summary shows that over 99% of the predictor variation and over 90% of the response variation are accounted for by the three factors.

Predicting New Observations

Now that you have chosen a three-factor PLS model for predicting pollutant concentrations based on sample spectra, suppose that you have two new samples. The following SAS statements create a data set containing the spectra for the new samples:

```
data newobs;
   input obsnam $ v1-v27 @@;
   datalines;
EM17  3933 4518 5637 6006 5721 5187 4641 4149 3789
      3579 3447 3381 3327 3234 3078 2832 2571 2274
      2040 1818 1629 1470 1350 1245 1134 1050  987
EM25  2904 2997 3255 3150 2922 2778 2700 2646 2571
      2487 2370 2250 2127 2052 1713 1419 1200  984
       795  648  525  426  351  291  240  204  162
;
```

You can apply the PLS model to these samples to estimate pollutant concentration. To do so, append the new samples to the original 16, and specify that the predicted values for all 18 be output to a data set, as shown in the following statements:

```
data all;
   set sample newobs;
run;

proc pls data=all nfac=3;
   model ls ha dt = v1-v27;
   output out=pred p=p_ls p_ha p_dt;
run;

proc print data=pred;
   where (obsnam in ('EM17','EM25'));
   var obsnam p_ls p_ha p_dt;
run;
```

The new observations are not used in calculating the PLS model, since they have no response values. Their predicted concentrations are shown in Figure 95.6.

Figure 95.6 Predicted Concentrations for New Observations

Obs	obsnam	p_ls	p_ha	p_dt
17	EM17	2.54261	0.31877	81.4174
18	EM25	-0.24716	1.37892	46.3212

Finally, if ODS Graphics is enabled, PLS also displays by default a plot of the amount of variation accounted for by each factor, as well as a correlations loading plot that summarizes the first two dimensions of the PLS model. The following statements, which are the same as the previous split-sample validation analysis but with ODS Graphics enabled, additionally produce Figure 95.7 and Figure 95.8:

```
ods graphics on;

proc pls data=sample cv=split cvtest(seed=12345);
   model ls ha dt = v1-v27;
run;
```

Figure 95.7 Split-Sample Cross Validation Plot

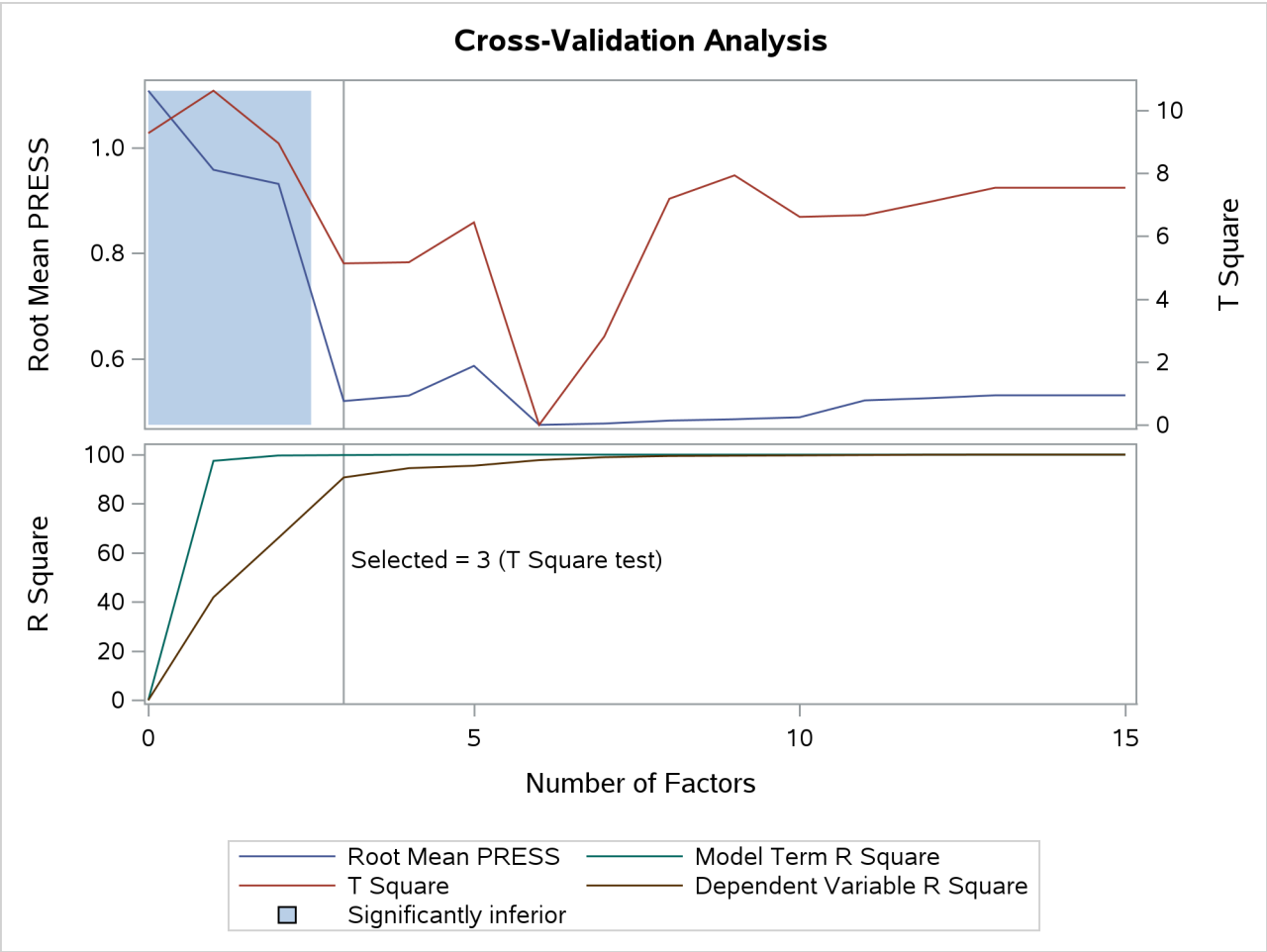

Figure 95.8 Correlation Loading Plot

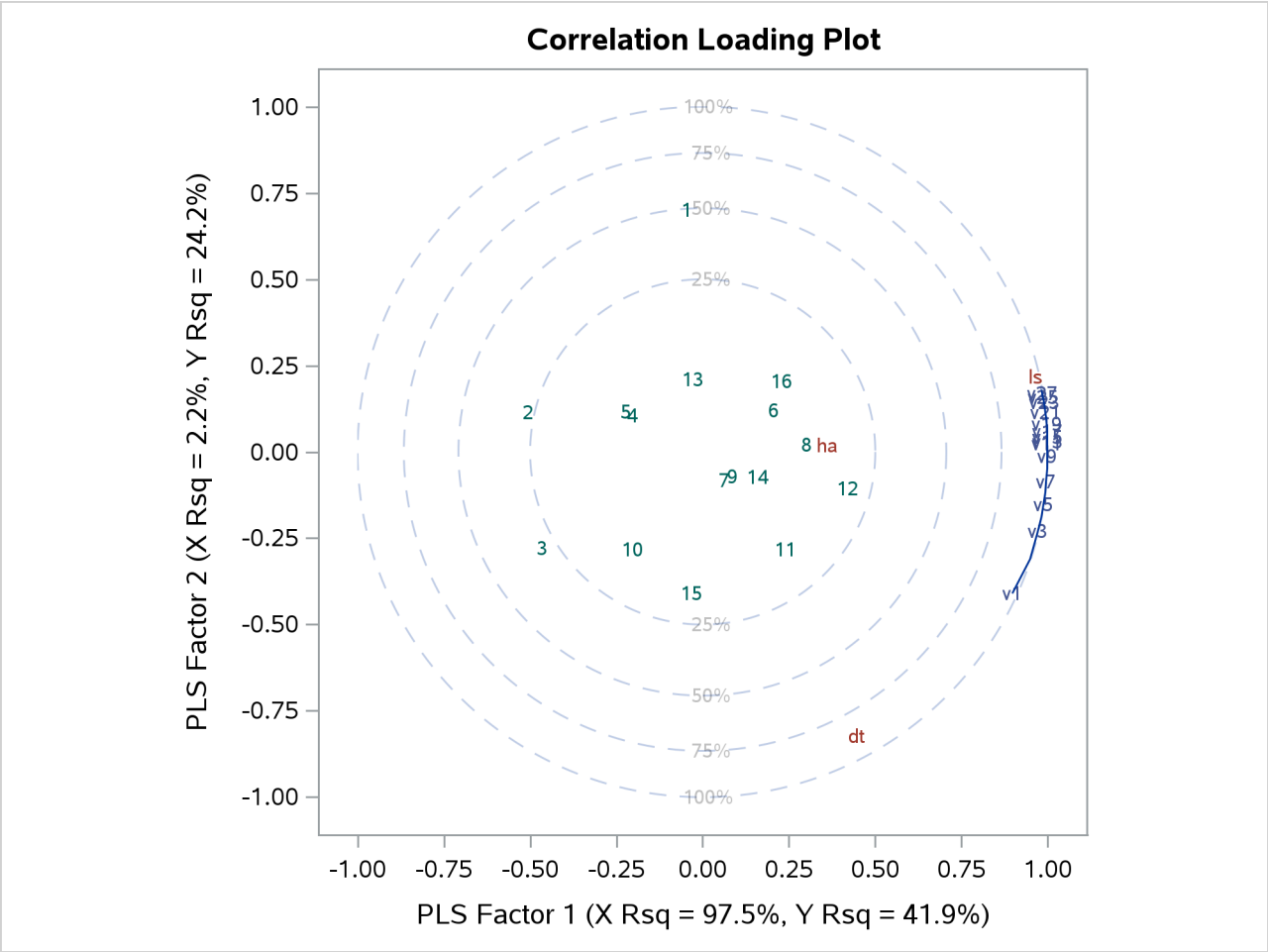

The cross validation plot in Figure 95.7 gives a visual representation of the selection of the optimum number of factors discussed previously. The correlation loading plot is a compact summary of many features of the PLS model. For example, it shows that the first factor is highly positively correlated with all spectral values, indicating that it is approximately an average of them all; the second factor is positively correlated with the lowest frequencies and negatively correlated with the highest, indicating that it is approximately a contrast between the two ends of the spectrum. The observations, represented by their number in the data set on this plot, are generally spaced well apart, indicating that the data give good information about these first two factors. For more details on the interpretation of the correlation loading plot, see the section "ODS Graphics" on page 7868 and Example 95.1.

The default correlation loading plot for just the first two factors depicts most of the model information. In order to see correlation loadings for all three of the selected factors, you use the NFAC= suboption for the PLOT=CORRLOAD option, as in the following:

```
proc pls data=sample nfac=3 plot=corrload(nfac=3);
   model ls ha dt = v1-v27;
run;
```

The resulting plot is shown in Figure 95.9.

Figure 95.9 Correlation Loading Plot Matrix

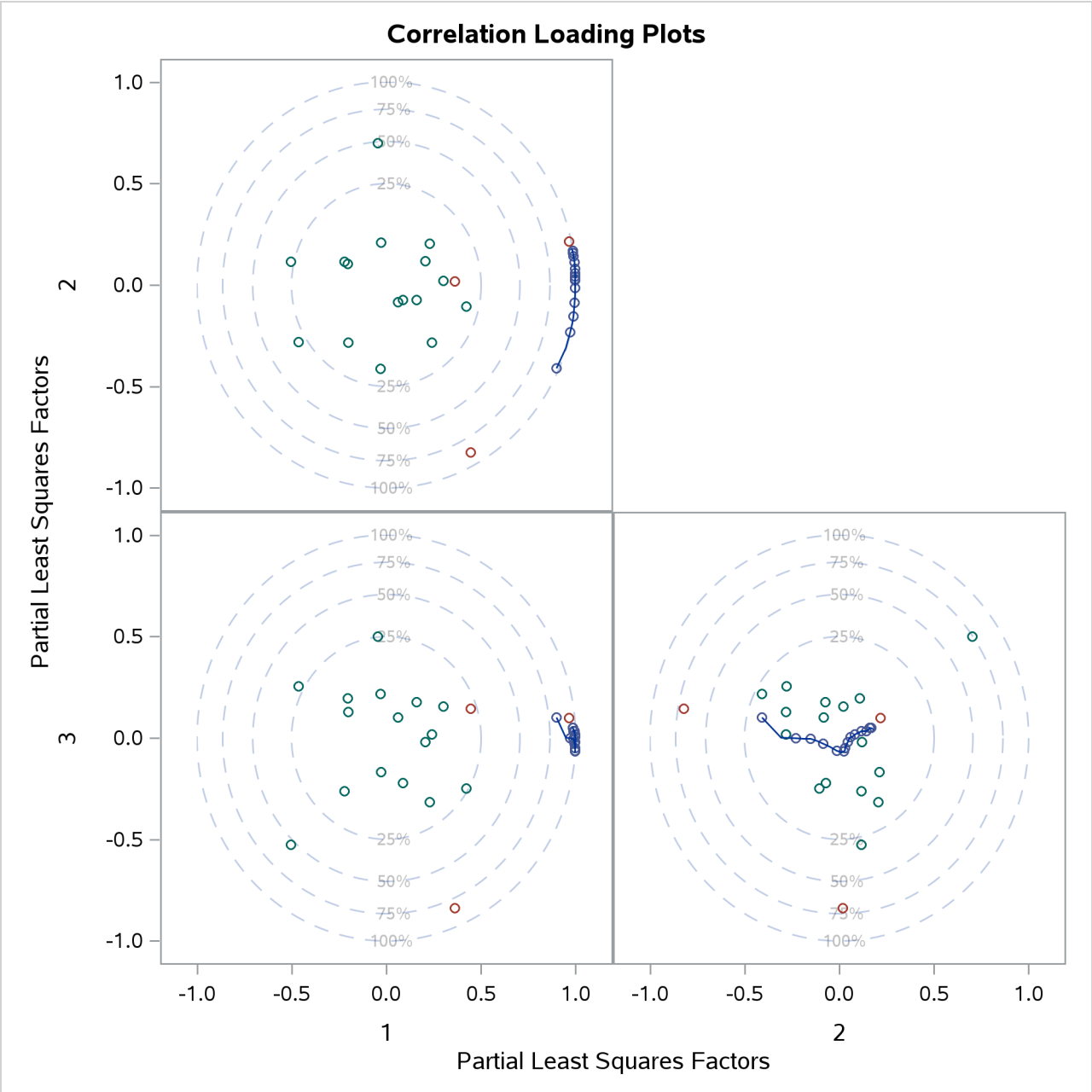

Syntax: PLS Procedure

The following statements are available in the PLS procedure. Items within the angle brackets are optional.

```
PROC PLS < options > ;
    BY variables ;
    CLASS variables < / option > ;
    EFFECT name=effect-type (variables< / options >) ;
    ID variables ;
    MODEL dependent-variables = effects < / options > ;
    OUTPUT OUT=SAS-data-set < options > ;
```

To analyze a data set, you must use the PROC PLS and MODEL statements. You can use the other statements as needed. CLASS and EFFECT statements, if present, must precede the MODEL statement.

PROC PLS Statement

```
PROC PLS < options > ;
```

The PROC PLS statement invokes the PLS procedure. Optionally, you can also indicate the analysis data and method in the PROC PLS statement. Table 95.1 summarizes the *options* available in the PROC PLS statement.

Table 95.1 PROC PLS Statement Options

Option	Description
CENSCALE	Displays the centering and scaling information
CV=	Specifies the cross validation method to be used
CVTEST	Specifies that van der Voet's (1994) randomization-based model comparison test be performed
DATA=	Names the SAS data set
DETAILS	Displays the details of the fitted model
METHOD=	Specifies the general factor extraction method to be used
MISSING=	Specifies how observations with missing values are to be handled in computing the fit
NFAC=	Specifies the number of factors to extract
NOCENTER	Suppresses centering of the responses and predictors before fitting
NOCVSTDIZE	Suppresses re-centering and rescaling of the responses and predictors when cross validating
NOPRINT	Suppresses the normal display of results
NOSCALE	Suppresses scaling of the responses and predictors before fitting
PLOTS	Controls the plots produced through ODS Graphics
VARSCALE	Specifies that continuous model variables be centered and scaled
VARSS	Displays the amount of variation accounted for in each response and predictor

The following *options* are available.

CENSCALE
lists the centering and scaling information for each response and predictor.

CV=ONE

CV=SPLIT < (*n*) >

CV=BLOCK < (*n*) >

CV=RANDOM < (*cv-random-opts*) >

CV=TESTSET(*SAS-data-set***)**
specifies the cross validation method to be used. By default, no cross validation is performed. The method CV=ONE requests one-at-a-time cross validation, CV=SPLIT requests that every *n*th observation be excluded, CV=BLOCK requests that *n* blocks of consecutive observations be excluded, CV=RANDOM requests that observations be excluded at random, and CV=TESTSET(*SAS-data-set*) specifies a test set of observations to be used for validation (formally, this is called "test set validation" rather than "cross validation"). You can, optionally, specify *n* for CV=SPLIT and CV=BLOCK; the default is $n = 7$. You can also specify the following optional *cv-random-options* in parentheses after the CV=RANDOM option:

NITER=*n*
specifies the number of random subsets to exclude. The default value is 10.

NTEST=*n*
specifies the number of observations in each random subset chosen for exclusion. The default value is one-tenth of the total number of observations.

SEED=*n*
specifies an integer used to start the pseudo-random number generator for selecting the random test set. If you do not specify a seed, or specify a value less than or equal to zero, the seed is by default generated from reading the time of day from the computer's clock.

CVTEST < (*cvtest-options*) >
specifies that van der Voet's (1994) randomization-based model comparison test be performed to test models with different numbers of extracted factors against the model that minimizes the predicted residual sum of squares; for more information, see the section "Cross Validation" on page 7864. You can also specify the following *cv-test-options* in parentheses after the CVTEST option:

PVAL=*n*
specifies the cutoff probability for declaring an insignificant difference. The default value is 0.10.

STAT=*test-statistic*
specifies the test statistic for the model comparison. You can specify either T2, for Hotelling's T^2 statistic, or PRESS, for the predicted residual sum of squares. The default value is T2.

NSAMP=*n*
specifies the number of randomizations to perform. The default value is 1000.

SEED=*n*
specifies the seed value for randomization generation (the clock time is used by default).

DATA=*SAS-data-set*
names the SAS data set to be used by PROC PLS. The default is the most recently created data set.

DETAILS
lists the details of the fitted model for each successive factor. The details listed are different for different extraction methods; for more information, see the section "Displayed Output" on page 7866.

METHOD=PLS< (*PLS-options* **) > | SIMPLS | PCR | RRR**
specifies the general factor extraction method to be used. The value PLS requests partial least squares, SIMPLS requests the SIMPLS method of De Jong (1993), PCR requests principal components regression, and RRR requests reduced rank regression. The default is METHOD=PLS. You can also specify the following optional *PLS-options* in parentheses after METHOD=PLS:

ALGORITHM=NIPALS | SVD | EIG | RLGW
names the specific algorithm used to compute extracted PLS factors. NIPALS requests the usual iterative NIPALS algorithm, SVD bases the extraction on the singular value decomposition of $\mathbf{X}'\mathbf{Y}$, EIG bases the extraction on the eigenvalue decomposition of $\mathbf{Y}'\mathbf{X}\mathbf{X}'\mathbf{Y}$, and RLGW is an iterative approach that is efficient when there are many predictors. ALGORITHM=SVD is the most accurate but least efficient approach; the default is ALGORITHM=NIPALS.

MAXITER=*n*
specifies the maximum number of iterations for the NIPALS and RLGW algorithms. The default value is 200.

EPSILON=*n*
specifies the convergence criterion for the NIPALS and RLGW algorithms. The default value is 10^{-12}.

MISSING=NONE | AVG | EM < (*EM-options* **) >**
specifies how observations with missing values are to be handled in computing the fit. The default is MISSING=NONE, for which observations with any missing variables (dependent or independent) are excluded from the analysis. MISSING=AVG specifies that the fit be computed by filling in missing values with the average of the nonmissing values for the corresponding variable. If you specify MISSING=EM, then the procedure first computes the model with MISSING=AVG and then fills in missing values by their predicted values based on that model and computes the model again. For both methods of imputation, the imputed values contribute to the centering and scaling values, and the difference between the imputed values and their final predictions contributes to the percentage of variation explained. You can also specify the following optional *EM-options* in parentheses after MISSING=EM:

MAXITER=*n*
specifies the maximum number of iterations for the imputation/fit loop. The default value is 1. If you specify a large value of MAXITER=, then the loop will iterate until it converges (as controlled by the EPSILON= option).

EPSILON=*n*
specifies the convergence criterion for the imputation/fit loop. The default value is 10^{-8}. This option is effective only if you specify a large value for the MAXITER= option.

NFAC=*n*
specifies the number of factors to extract. The default is $\min\{15, p, N\}$, where p is the number of predictors (the number of dependent variables for METHOD=RRR) and N is the number of runs (observations). This is probably more than you need for most applications. Extracting too many factors can lead to an overfit model, one that matches the training data too well, sacrificing predictive ability. Thus, if you use the default NFAC= specification, you should also either use the CV= option to select the appropriate number of factors for the final model or consider the analysis to be preliminary and examine the results to determine the appropriate number of factors for a subsequent analysis.

NOCENTER
suppresses centering of the responses and predictors before fitting. This is useful if the analysis variables are already centered and scaled. For more information, see the section "Centering and Scaling" on page 7865.

NOCVSTDIZE
suppresses re-centering and rescaling of the responses and predictors before each model is fit in the cross validation. For more information, see the section "Centering and Scaling" on page 7865.

NOPRINT
suppresses the normal display of results. This is useful when you want only the output statistics saved in a data set. Note that this option temporarily disables the Output Delivery System (ODS); for more information, see Chapter 23, "Using the Output Delivery System."

NOSCALE
suppresses scaling of the responses and predictors before fitting. This is useful if the analysis variables are already centered and scaled. For more information, see the section "Centering and Scaling" on page 7865.

PLOTS < (*global-plot-options*) > < = *plot-request*< (*options*) > >
PLOTS < (*global-plot-options*) > < = (*plot-request*< (*options*) > < . . . *plot-request*< (*options*) > >) >
controls the plots produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses from around the plot request. For example:

```
plots=none
plots=cvplot
plots=(diagnostics cvplot)
plots(unpack)=diagnostics
plots(unpack)=(diagnostics corrload(trace=off))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc pls data=pentaTrain;
   model log_RAI = S1-S5 L1-L5 P1-P5;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, then PROC PLS produces by default a plot of the R-square analysis and a correlation loading plot summarizing the first two factors.

The *global-plot-options* include the following:

FLIP
interchanges the X-axis and Y-axis dimensions for the score, weight, and loading plots.

ONLY
suppresses the default plots. Only plots specifically requested are displayed.

UNPACKPANEL
UNPACK
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACKPANEL to get each plot in a separate panel. You can specify PLOTS(UNPACKPANEL) to unpack only the default plots. You can also specify UNPACKPANEL as a suboption for certain specific plots, as discussed in the following.

The *plot-requests* include the following:

ALL
produces all appropriate plots. You can specify other options with ALL—for example, to request all plots and unpack only the residuals, specify PLOTS=(ALL RESIDUALS(UNPACK)).

CORRLOAD <(*options*)>
produces a correlation loading plot (default). You can specify the following *options*:

TRACE=OFF | ON
controls how points that correspond to the X-loadings are depicted. You can specify the following values:

OFF specifies that all X-loadings be depicted in the plot by their names plotted at the corresponding point on the graph.

ON specifies that the positions for all the X-loadings be depicted by a "trace" through the corresponding points.

By default, TRACE=ON if there are more than 20 predictors, and TRACE=OFF otherwise.

NFAC=*n*
specifies the number of factors for which to display correlation loading plots. By default, NFAC=2, which corresponds to a single plot for the first two factors. If you specify a value of n greater than 2, then the $n(n-1)/2$ plots are displayed together in a matrix of correlation loading plots. The maximum number of factors that can be displayed in such a matrix is 8.

UNPACK
requests that the $n(n-1)/2$ correlation loading plots be produced separately instead of in a matrix. This options has no effect unless the NFAC=*n* option is also specified, with a value of *n* greater than 2.

CVPLOT
produces a cross validation and R-square analysis. This plot requires the CV= option to be specified, and is displayed by default in this case.

DIAGNOSTICS < (UNPACK) >
produces a summary panel of the fit for each dependent variable. The summary by default consists of a panel for each dependent variable, with plots depicting the distribution of residuals and predicted values. You can use the UNPACK suboption to specify that the subplots be produced separately.

DMOD
produces the DMODX, DMODY, and DMODXY plots.

DMODX
produces a plot of the distance of each observation to the X model.

DMODXY
produces plots of the distance of each observation to the X and Y models.

DMODY
produces a plot of the distance of each observation to the Y model.

FIT
produces both the fit diagnostic panel and the ParmProfiles plot.

NONE
suppresses the display of graphics.

PARMPROFILES
produces profiles of the regression coefficients.

SCORES < (UNPACK | FLIP) >
produces the XScores, YScores, XYScores, and DModXY plots. You can use the UNPACK suboption to specify that the subplots for scores be produced separately, and the FLIP option to interchange their default X-axis and Y-axis dimensions.

RESIDUALS < (UNPACK) >
plots the residuals for each dependent variable against each independent variable. Residual plots are by default composed of multiple plots combined into a single panel. You can use the UNPACK suboption to specify that the subplots be produced separately.

VIP
produces profiles of variable importance factors.

WEIGHTS < (UNPACK | FLIP) >
produces all X and Y loading and weight plots, as well as the VIP plot. You can use the UNPACK suboption to specify that the subplots for weights and loadings be produced separately, and the FLIP option to interchange their default X-axis and Y-axis dimensions.

XLOADINGPLOT < (UNPACK | FLIP) >
produces a scatter plot matrix of X-loadings against each other. Loading scatter plot matrices are by default composed of multiple plots combined into a single panel. You can use the UNPACK suboption to specify that the subplots be produced separately, and the FLIP option to interchange the default X-axis and Y-axis dimensions.

XLOADINGPROFILES
produces profiles of the X-loadings.

XSCORES < (UNPACK | FLIP) >
produces a scatter plot matrix of X-scores against each other. Score scatter plot matrices are by default composed of multiple plots combined into a single panel. You can use the UNPACK suboption to specify that the subplots be produced separately, and the FLIP option to interchange the default X-axis and Y-axis dimensions.

XWEIGHTPLOT < (UNPACK | FLIP) >
produces a scatter plot matrix of X-weights against each other. Weight scatter plot matrices are by default composed of multiple plots combined into a single panel. You can use the UNPACK suboption to specify that the subplots be produced separately, and the FLIP option to interchange the default X-axis and Y-axis dimensions.

XWEIGHTPROFILES
produces profiles of the X-weights.

XYSCORES < (UNPACK) >
produces a scatter plot matrix of X-scores against Y-scores. Score scatter plot matrices are by default composed of multiple plots combined into a single panel. You can use the UNPACK suboption to specify that the subplots be produced separately.

YSCORES < (UNPACK | FLIP) >
produces a scatter plot matrix of Y-scores against each other. Score scatter plot matrices are by default composed of multiple plots combined into a single panel. You can use the UNPACK suboption to specify that the subplots be produced separately, and the FLIP option to interchange the default X-axis and Y-axis dimensions.

YWEIGHTPLOT < (UNPACK | FLIP) >
produces a scatter plot matrix of Y-weights against each other. Weight scatter plot matrices are by default composed of multiple plots combined into a single panel. You can use the UNPACK suboption to specify that the subplots be produced separately, and the FLIP option to interchange the default X-axis and Y-axis dimensions.

VARSCALE

specifies that continuous model variables be centered and scaled prior to centering and scaling the model effects in which they are involved. The rescaling specified by the VARSCALE option is sometimes more appropriate if the model involves crossproducts between model variables; however, the VARSCALE option still might not produce the model you expect. For more information, see the section "Centering and Scaling" on page 7865.

VARSS

lists, in addition to the average response and predictor sum of squares accounted for by each successive factor, the amount of variation accounted for in each response and predictor.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC PLS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the PLS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / **TRUNCATE** > **;**

The CLASS statement names the classification variables to be used in the model. The PLS procedure uses a singular, GLM parameterization for effects involving CLASS variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics." Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

You can use the TRUNCATE option to determine class levels by using no more than the first 16 characters of the formatted values.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables*< / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

The following *effect-types* are available.

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 95.2 summarizes the *options* available in the EFFECT statement.

Table 95.2 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned

Table 95.2 *continued*

Option	Description
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

ID Statement

ID *variables* **;**

The ID statement names variables whose values are used to label observations in plots. If you do not specify an ID statement, then each observations is labeled in plots by its corresponding observation number.

MODEL Statement

MODEL *response-variables* **=** *predictor-effects* < / *options* > **;**

The MODEL statement names the responses and the predictors, which determine the **Y** and **X** matrices of the model, respectively. Usually you simply list the names of the predictor variables as the model effects, but you can also use the effects notation of PROC GLM to specify polynomial effects and interactions; for more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure." The MODEL statement is required. You can specify only one MODEL statement (in contrast to the REG procedure, for example, which allows several MODEL statements in the same PROC REG run).

You can specify the following *options* in the MODEL statement after a slash (/).

INTERCEPT
By default, the responses and predictors are centered; thus, no intercept is required in the model. You can specify the INTERCEPT option to override the default.

SOLUTION
lists the coefficients of the final predictive model for the responses. The coefficients for predicting the centered and scaled responses based on the centered and scaled predictors are displayed, as well as the coefficients for predicting the raw responses based on the raw predictors.

OUTPUT Statement

OUTPUT OUT=*SAS-data-set keyword***=***names* < . . . *keyword***=***names* > **;**

You use the OUTPUT statement to specify a data set to receive quantities that can be computed for every input observation, such as extracted factors and predicted values. The following *keywords* are available:

PREDICTED	predicted values for responses
YRESIDUAL	residuals for responses
XRESIDUAL	residuals for predictors
XSCORE	extracted factors (X-scores, latent vectors, latent variables, T)
YSCORE	extracted responses (Y-scores, U)
STDY	standardized (centered and scaled) responses
STDX	standardized (centered and scaled) predictors
H	approximate leverage
PRESS	approximate predicted residuals
TSQUARE	scaled sum of squares of score values
STDXSSE	sum of squares of residuals for standardized predictors
STDYSSE	sum of squares of residuals for standardized responses

Suppose that there are N_x predictors and N_y responses and that the model has N_f selected factors.

- The keywords XRESIDUAL and STDX define an output variable for each predictor, so N_x names are required after each one.
- The keywords PREDICTED, YRESIDUAL, STDY, and PRESS define an output variable for each response, so N_y names are required after each of these *keywords*.
- The keywords XSCORE and YSCORE specify an output variable for each selected model factor. For these *keywords*, you provide only one base name, and the variables corresponding to each successive factor are named by appending the factor number to the base name. For example, if $N_f = 3$, then a specification of XSCORE=T would produce the variables T1, T2, and T3.
- Finally, the keywords H, TSQUARE, STDXSSE, and STDYSSE each specify a single output variable, so only one name is required after each of these *keywords*.

Details: PLS Procedure

Regression Methods

All of the predictive methods implemented in PROC PLS work essentially by finding linear combinations of the predictors (factors) to use to predict the responses linearly. The methods differ only in how the factors are derived, as explained in the following sections.

Partial Least Squares

Partial least squares (PLS) works by extracting one factor at a time. Let $\mathbf{X} = \mathbf{X}_0$ be the centered and scaled matrix of predictors and let $\mathbf{Y} = \mathbf{Y}_0$ be the centered and scaled matrix of response values. The PLS method starts with a linear combination $\mathbf{t} = \mathbf{X}_0\mathbf{w}$ of the predictors, where $\mathbf{t}$ is called a *score* vector and $\mathbf{w}$ is its associated *weight* vector. The PLS method predicts both $\mathbf{X}_0$ and $\mathbf{Y}_0$ by regression on $\mathbf{t}$:

$$\begin{aligned} \hat{\mathbf{X}}_0 &= \mathbf{t}\mathbf{p}', \quad \text{where} \quad \mathbf{p}' = (\mathbf{t}'\mathbf{t})^{-1}\mathbf{t}'\mathbf{X}_0 \\ \hat{\mathbf{Y}}_0 &= \mathbf{t}\mathbf{c}', \quad \text{where} \quad \mathbf{c}' = (\mathbf{t}'\mathbf{t})^{-1}\mathbf{t}'\mathbf{Y}_0 \end{aligned}$$

The vectors $\mathbf{p}$ and $\mathbf{c}$ are called the X- and Y-*loadings*, respectively.

The specific linear combination $\mathbf{t} = \mathbf{X}_0\mathbf{w}$ is the one that has maximum covariance $\mathbf{t}'\mathbf{u}$ with some response linear combination $\mathbf{u} = \mathbf{Y}_0\mathbf{q}$. Another characterization is that the X- and Y-weights $\mathbf{w}$ and $\mathbf{q}$ are proportional to the first left and right singular vectors of the covariance matrix $\mathbf{X}_0'\mathbf{Y}_0$ or, equivalently, the first eigenvectors of $\mathbf{X}_0'\mathbf{Y}_0\mathbf{Y}_0'\mathbf{X}_0$ and $\mathbf{Y}_0'\mathbf{X}_0\mathbf{X}_0'\mathbf{Y}_0$, respectively.

This accounts for how the first PLS factor is extracted. The second factor is extracted in the same way by replacing $\mathbf{X}_0$ and $\mathbf{Y}_0$ with the X- and Y-residuals from the first factor:

$$\begin{aligned} \mathbf{X}_1 &= \mathbf{X}_0 - \hat{\mathbf{X}}_0 \\ \mathbf{Y}_1 &= \mathbf{Y}_0 - \hat{\mathbf{Y}}_0 \end{aligned}$$

These residuals are also called the *deflated* $\mathbf{X}$ and $\mathbf{Y}$ blocks. The process of extracting a score vector and deflating the data matrices is repeated for as many extracted factors as are wanted.

SIMPLS

Note that each extracted PLS factor is defined in terms of different X-variables $\mathbf{X}_i$. This leads to difficulties in comparing different scores, weights, and so forth. The SIMPLS method of De Jong (1993) overcomes these difficulties by computing each score $\mathbf{t}_i = \mathbf{X}\mathbf{r}_i$ in terms of the original (centered and scaled) predictors $\mathbf{X}$. The SIMPLS X-weight vectors r_i are similar to the eigenvectors of $\mathbf{SS}' = \mathbf{X}'\mathbf{YY}'\mathbf{X}$, but they satisfy a different orthogonality condition. The $\mathbf{r}_1$ vector is just the first eigenvector $\mathbf{e}_1$ (so that the first SIMPLS score is the same as the first PLS score), but whereas the second eigenvector maximizes

$$\mathbf{e}_1'\mathbf{SS}'\mathbf{e}_2 \text{ subject to } \mathbf{e}_1'\mathbf{e}_2 = 0$$

the second SIMPLS weight $\mathbf{r}_2$ maximizes

$$\mathbf{r}_1' S S' \mathbf{r}_2 \text{ subject to } \mathbf{r}_1'\mathbf{X}'\mathbf{X}\mathbf{r}_2 = \mathbf{t}_1'\mathbf{t}_2 = 0$$

The SIMPLS scores are identical to the PLS scores for one response but slightly different for more than one response; see De Jong (1993) for details. The X- and Y-loadings are defined as in PLS, but since the scores are all defined in terms of $\mathbf{X}$, it is easy to compute the overall model coefficients $\mathbf{B}$:

$$\begin{aligned}
\hat{\mathbf{Y}} &= \sum_i \mathbf{t}_i \mathbf{c}_i' \\
&= \sum_i \mathbf{X}\mathbf{r}_i \mathbf{c}_i' \\
&= \mathbf{XB}, \text{ where } \mathbf{B} = \mathbf{RC}'
\end{aligned}$$

Principal Components Regression

Like the SIMPLS method, principal components regression (PCR) defines all the scores in terms of the original (centered and scaled) predictors $\mathbf{X}$. However, unlike both the PLS and SIMPLS methods, the PCR method chooses the X-weights/X-scores without regard to the response data. The X-scores are chosen to explain as much variation in $\mathbf{X}$ as possible; equivalently, the X-weights for the PCR method are the eigenvectors of the predictor covariance matrix $\mathbf{X}'\mathbf{X}$. Again, the X- and Y-loadings are defined as in PLS; but, as in SIMPLS, it is easy to compute overall model coefficients for the original (centered and scaled) responses $\mathbf{Y}$ in terms of the original predictors $\mathbf{X}$.

Reduced Rank Regression

As discussed in the preceding sections, partial least squares depends on selecting factors $\mathbf{t} = \mathbf{Xw}$ of the predictors and $\mathbf{u} = \mathbf{Yq}$ of the responses that have maximum covariance, whereas principal components regression effectively ignores $\mathbf{u}$ and selects $\mathbf{t}$ to have maximum variance, subject to orthogonality constraints. In contrast, reduced rank regression selects $\mathbf{u}$ to account for as much variation in the *predicted* responses as possible, effectively ignoring the predictors for the purposes of factor extraction. In reduced rank regression, the Y-weights $\mathbf{q}_i$ are the eigenvectors of the covariance matrix $\hat{\mathbf{Y}}_{\mathrm{LS}}'\hat{\mathbf{Y}}_{\mathrm{LS}}$ of the responses predicted by ordinary least squares regression; the X-scores are the projections of the Y-scores $\mathbf{Yq}_i$ onto the X space.

Relationships between Methods

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/plsrel.sas.)

When you develop a predictive model, it is important to consider not only the explanatory power of the model for current responses, but also how well sampled the predictive functions are, since this affects how well the model can extrapolate to future observations. All of the techniques implemented in the PLS procedure work by extracting successive factors, or linear combinations of the predictors, that optimally address one or both of these two goals—explaining response variation and explaining predictor variation. In particular, principal components regression selects factors that explain as much predictor variation as possible, reduced rank regression selects factors that explain as much response variation as possible, and partial least squares balances the two objectives, seeking for factors that explain both response and predictor variation.

To see the relationships between these methods, consider how each one extracts a single factor from the following artificial data set consisting of two predictors and one response:

```
data data;
   input x1 x2 y;
   datalines;
    3.37651  2.30716        0.75615
    0.74193 -0.88845        1.15285
    4.18747  2.17373        1.42392
    0.96097  0.57301        0.27433
   -1.11161 -0.75225       -0.25410
   -1.38029 -1.31343       -0.04728
    1.28153 -0.13751        1.00341
   -1.39242 -2.03615        0.45518
    0.63741  0.06183        0.40699
   -2.52533 -1.23726       -0.91080
    2.44277  3.61077       -0.82590
;

proc pls data=data nfac=1 method=rrr;
   model y = x1 x2;
run;

proc pls data=data nfac=1 method=pcr;
   model y = x1 x2;
run;

proc pls data=data nfac=1 method=pls;
   model y = x1 x2;
run;
```

The amount of model and response variation explained by the first factor for each method is shown in Figure 95.10 through Figure 95.12.

Figure 95.10 Variation Explained by First Reduced Rank Regression Factor

The PLS Procedure

Percent Variation Accounted for by Reduced Rank Regression Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	15.0661	15.0661	100.0000	100.0000

Figure 95.11 Variation Explained by First Principal Components Regression Factor

The PLS Procedure

Percent Variation Accounted for by Principal Components				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	92.9996	92.9996	9.3787	9.3787

Figure 95.12 Variation Explained by First Partial Least Squares Regression Factor

The PLS Procedure

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	88.5357	88.5357	26.5304	26.5304

Notice that, while the first reduced rank regression factor explains *all* of the response variation, it accounts for only about 15% of the predictor variation. In contrast, the first principal components regression factor accounts for most of the predictor variation (93%) but only 9% of the response variation. The first partial least squares factor accounts for only slightly less predictor variation than principal components but about three times as much response variation.

Figure 95.13 illustrates how partial least squares balances the goals of explaining response and predictor variation in this case.

Figure 95.13 Depiction of First Factors for Three Different Regression Methods

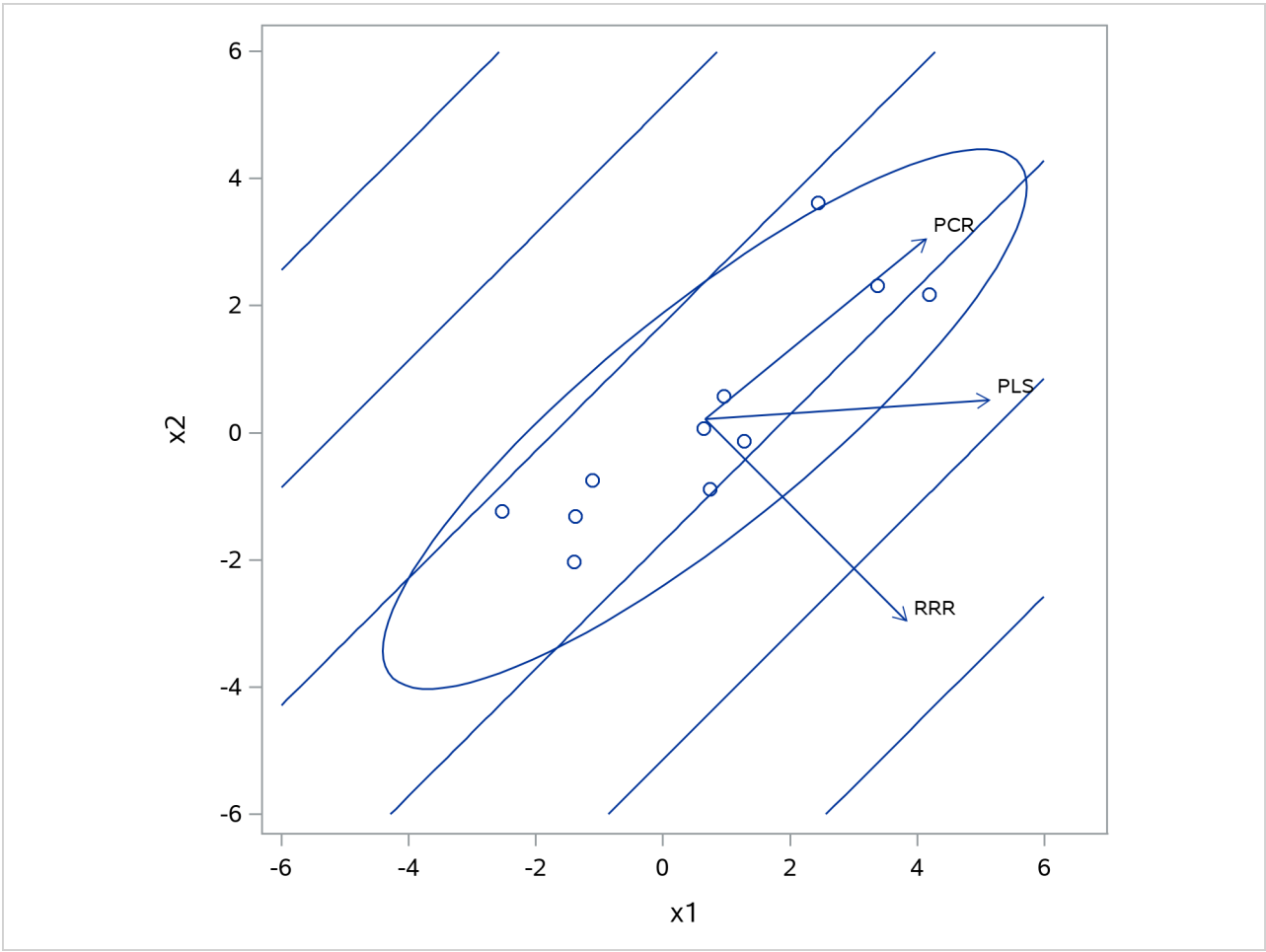

The ellipse shows the general shape of the 11 observations in the predictor space, with the contours of increasing y overlaid. Also shown are the directions of the first factor for each of the three methods. Notice that, while the predictors vary most in the x1 = x2 direction, the response changes most in the orthogonal x1 = -x2 direction. This explains why the first principal component accounts for little variation in the response and why the first reduced rank regression factor accounts for little variation in the predictors. The direction of the first partial least squares factor represents a compromise between the other two directions.

Cross Validation

None of the regression methods implemented in the PLS procedure fit the observed data any better than ordinary least squares (OLS) regression; in fact, all of the methods approach OLS as more factors are extracted. The crucial point is that, when there are many predictors, OLS can *overfit* the observed data; biased regression methods with fewer extracted factors can provide better predictability of *future* observations. However, as the preceding observations imply, the quality of the observed data fit cannot be used to choose the number of factors to extract; the number of extracted factors must be chosen on the basis of how well the model fits observations not involved in the modeling procedure itself.

One method of choosing the number of extracted factors is to fit the model to only part of the available data (the *training set*) and to measure how well models with different numbers of extracted factors fit the other part of the data (the *test set*). This is called *test set validation*. However, it is rare that you have enough data to make both parts large enough for pure test set validation to be useful. Alternatively, you can make several different divisions of the observed data into training set and test set. This is called *cross validation*, and there are several different types. In *one-at-a-time* cross validation, the first observation is held out as a single-element test set, with all other observations as the training set; next, the second observation is held out, then the third, and so on. Another method is to hold out successive blocks of observations as test sets—for example, observations 1 through 7, then observations 8 through 14, and so on; this is known as *blocked* validation. A similar method is *split-sample* cross validation, in which successive groups of widely separated observations are held out as the test set—for example, observations {1, 11, 21, . . . }, then observations {2, 12, 22, . . . }, and so on. Finally, test sets can be selected from the observed data randomly; this is known as *random sample* cross validation.

Which validation you should use depends on your data. Test set validation is preferred when you have enough data to make a division into a sizable training set and test set that represent the predictive population well. You can specify that the number of extracted factors be selected by test set validation by using the CV=TESTSET(*data set*) option, where *data set* is the name of the data set containing the test set. If you do not have enough data for test set validation, you can use one of the cross validation techniques. The most common technique is one-at-a-time validation (which you can specify with the CV=ONE option or just the CV option), unless the observed data are serially correlated, in which case either blocked or split-sample validation might be more appropriate (CV=BLOCK or CV=SPLIT); you can specify the number of test sets in blocked or split-sample validation with a number in parentheses after the CV= option. Note that CV=ONE is the most computationally intensive of the cross validation methods, since it requires a recomputation of the PLS model for every input observation. Also, note that using random subset selection with CV=RANDOM might lead two different researchers to produce different PLS models on the same data (unless the same seed is used).

Whichever validation method you use, the number of factors chosen is usually the one that minimizes the predicted residual sum of squares (PRESS); this is the default choice if you specify any of the CV methods with PROC PLS. However, often models with fewer factors have PRESS statistics that are only marginally larger than the absolute minimum. To address this, Van der Voet (1994) has proposed a statistical test for comparing the predicted residuals from different models; when you apply van der Voet's test, the number of factors chosen is the fewest with residuals that are insignificantly larger than the residuals of the model with minimum PRESS.

To see how van der Voet's test works, let $R_{i,jk}$ be the jth predicted residual for response k for the model with i extracted factors; the PRESS statistic is $\sum_{jk} R^2_{i,jk}$. Also, let $i_{\min}$ be the number of factors for which PRESS is minimized. The critical value for van der Voet's test is based on the differences between squared

predicted residuals

$$D_{i,jk} \quad = \quad R^2_{i,jk} - R^2_{i_{\min},jk}$$

One alternative for the critical value is $C_i = \sum_{jk} D_{i,jk}$, which is just the difference between the PRESS statistics for i and $i_{\min}$ factors; alternatively, van der Voet suggests Hotelling's T^2 statistic $C_i = \mathbf{d}'_{i,\cdot}\mathbf{S}_i^{-1}\mathbf{d}_{i,\cdot}$, where $\mathbf{d}_{i,\cdot}$ is the sum of the vectors $\mathbf{d}_{i,j} = \{D_{i,j1}, \ldots, D_{i,jN_y}\}'$ and $\mathbf{S}_i$ is the sum of squares and crossproducts matrix

$$\mathbf{S}_i \quad = \quad \sum_j \mathbf{d}_{i,j}\mathbf{d}'_{i,j}$$

Virtually, the significance level for van der Voet's test is obtained by comparing C_i with the distribution of values that result from randomly exchanging $R^2_{i,jk}$ and $R^2_{i_{\min},jk}$. In practice, a Monte Carlo sample of such values is simulated and the significance level is approximated as the proportion of simulated critical values that are greater than C_i. If you apply van der Voet's test by specifying the CVTEST option, then, by default, the number of extracted factors chosen is the least number with an approximate significance level that is greater than 0.10.

Centering and Scaling

By default, the predictors and the responses are centered and scaled to have mean 0 and standard deviation 1. Centering the predictors and the responses ensures that the criterion for choosing successive factors is based on how much *variation* they explain, in either the predictors or the responses or both. (See the section "Regression Methods" on page 7859 for more details on how different methods explain variation.) Without centering, both the mean variable value and the variation around that mean are involved in selecting factors. Scaling serves to place all predictors and responses on an equal footing relative to their variation in the data. For example, if Time and Temp are two of the predictors, then scaling says that a change of std(Time) in Time is roughly equivalent to a change of std(Temp) in Temp.

Usually, both the predictors and responses should be centered and scaled. However, if their values already represent variation around a nominal or target value, then you can use the NOCENTER option in the PROC PLS statement to suppress centering. Likewise, if the predictors or responses are already all on comparable scales, then you can use the NOSCALE option to suppress scaling.

Note that, if the predictors involve crossproduct terms, then, by default, the variables are *not* standardized before standardizing the crossproduct. That is, if the ith values of two predictors are denoted x_i^1 and x_i^2, then the default standardized ith value of the crossproduct is

$$\frac{x_i^1 x_i^2 - \mathrm{mean}_j(x_j^1 x_j^2)}{\mathrm{std}_j(x_j^1 x_j^2)}$$

If you want the crossproduct to be based instead on standardized variables

$$\frac{x_i^1 - m^1}{s^1} \times \frac{x_i^2 - m^2}{s^2}$$

where $m^k = \mathrm{mean}_j(x_j^k)$ and $s^k = \mathrm{std}_j(x_j^k)$ for $k = 1, 2$, then you should use the VARSCALE option in the PROC PLS statement. Standardizing the variables separately is usually a good idea, but unless the model also contains all crossproducts nested within each term, the resulting model might not be equivalent to a simple linear model in the same terms. To see this, note that a model involving the crossproduct of two standardized variables

$$\frac{x_i^1 - m^1}{s^1} \times \frac{x_i^2 - m^2}{s^2} = x_i^1 x_i^2 \frac{1}{s^1 s^2} - x_i^1 \frac{m^2}{s^1 s^2} - x_i^2 \frac{m^1}{s^1 s^2} + \frac{m^1 m^2}{s^1 s^2}$$

involves both the crossproduct term and the linear terms for the unstandardized variables.

When cross validation is performed for the number of effects, there is some disagreement among practitioners as to whether each cross validation training set should be retransformed. By default, PROC PLS does so, but you can suppress this behavior by specifying the NOCVSTDIZE option in the PROC PLS statement.

Missing Values

By default, PROC PLS handles missing values very simply. Observations with any missing independent variables (including all classification variables) are excluded from the analysis, and no predictions are computed for such observations. Observations with no missing independent variables but any missing dependent variables are also excluded from the analysis, but predictions are computed.

However, the MISSING= option in the PROC PLS statement provides more sophisticated ways of modeling in the presence of missing values. If you specify MISSING=AVG or MISSING=EM, then all observations in the input data set contribute to both the analysis and the OUTPUT OUT= data set. With MISSING=AVG, the fit is computed by filling in missing values with the average of the nonmissing values for the corresponding variable. With MISSING=EM, the procedure first computes the model with MISSING=AVG, then fills in missing values with their predicted values based on that model and computes the model again. Alternatively, you can specify MISSING=EM(MAXITER=*n*) with a large value of *n* in order to perform this imputation/fit loop until convergence.

Displayed Output

By default, PROC PLS displays just the amount of predictor and response variation accounted for by each factor.

If you perform a cross validation for the number of factors by specifying the CV option in the PROC PLS statement, then the procedure displays a summary of the cross validation for each number of factors, along with information about the optimal number of factors.

If you specify the DETAILS option in the PROC PLS statement, then details of the fitted model are displayed for each successive factor. These details for each number of factors include the following:

- the predictor loadings
- the predictor weights
- the response weights

- the coded regression coefficients (for METHOD=SIMPLS, PCR, or RRR)

If you specify the CENSCALE option in the PROC PLS statement, then centering and scaling information for each response and predictor is displayed.

If you specify the VARSS option in the PROC PLS statement, the procedure displays, in addition to the average response and predictor sum of squares accounted for by each successive factor, the amount of variation accounted for in each response and predictor.

If you specify the SOLUTION option in the MODEL statement, then PROC PLS displays the coefficients of the final predictive model for the responses. The coefficients for predicting the centered and scaled responses based on the centered and scaled predictors are displayed, as well as the coefficients for predicting the raw responses based on the raw predictors.

ODS Table Names

PROC PLS assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 95.3. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 95.3 ODS Tables Produced by PROC PLS

ODS Table Name	Description	Statement	Option
CVResults	Results of cross validation	PROC PLS	CV
CenScaleParms	Parameter estimates for centered and scaled data	MODEL	SOLUTION
CodedCoef	Coded coefficients	PROC PLS	DETAILS
MissingIterations	Iterations for missing value imputation	PROC PLS	MISSING=EM
ModelInfo	Model information	PROC PLS	Default
NObs	Number of observations	PROC PLS	Default
ParameterEstimates	Parameter estimates for raw data	MODEL	SOLUTION
PercentVariation	Variation accounted for by each factor	PROC PLS	Default
ResidualSummary	Residual summary from cross validation	PROC PLS	CV
XEffectCenScale	Centering and scaling information for predictor effects	PROC PLS	CENSCALE
XLoadings	Loadings for independents	PROC PLS	DETAILS
XVariableCenScale	Centering and scaling information for predictor variables	PROC PLS	CENSCALE and VARSCALE
XWeights	Weights for independents	PROC PLS	DETAILS
YVariableCenScale	Centering and scaling information for responses	PROC PLS	CENSCALE
YWeights	Weights for dependents	PROC PLS	DETAILS

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, by default the PLS procedure produces a plot of the variation accounted for by each extracted factor, as well as a *correlation loading plot* for the first two extracted factors (if the final model has at least two factors). The plot of the variation accounted for can take several forms:

- If the PLS analysis does not include cross validation, then the plot shows the total R square for both model effects and the dependent variables against the number of factors.
- If you specify the CV= option to select the number of factors in the final model by cross validation, then the plot shows the R-square analysis discussed previously as well as the root mean PRESS from the cross validation analysis, with the selected number of factors identified by a vertical line.

The correlation loading plot for the first two factors summarizes many aspects of the two most significant dimensions of the model. It consists of overlaid scatter plots of the scores of the first two factors, the loadings of the model effects, and the loadings of the dependent variables. The loadings are scaled so that the amount of variation in the variables that is explained by the model is proportional to the distance from the origin; circles indicating various levels of explained variation are also overlaid on the correlation loading plot. Also, the correlation between the model approximations for any two variables is proportional to the length of the projection of the point corresponding to one variable on a line through the origin passing through the point corresponding to the other variable; the sign of the correlation corresponds to which side of the origin the projected point falls on.

The R square and the first two correlation loadings are plotted by default when ODS Graphics is enabled, but you can produce many other plots for the PROC PLS analysis.

ODS Graph Names

PROC PLS assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 95.4.

Table 95.4 Graphs Produced by PROC PLS

ODS Graph Name	Plot Description	Option
CorrLoadPlot	Correlation loading plot (default)	PLOT=CORRLOAD(*option*)
CVPlot	Cross validation and R-square analysis (default, as appropriate)	CV=

Table 95.4 *continued*

ODS Graph Name	Plot Description	Option
DModXPlot	Distance of each observation to the X model	PLOT=DMODX
DModXYPlot	Distance of each observation to the X and Y models	PLOT=DMODXY
DModYPlot	Distance of each observation to the Y model	PLOT=DMODY
DiagnosticsPanel	Panel of diagnostic plots for the fit	PLOT=DIAGNOSTICS
AbsResidualByPredicted	Absolute residual by predicted values	PLOT=DIAGNOSTICS(UNPACK)
ObservedByPredicted	Observed by predicted	PLOT=DIAGNOSTICS(UNPACK)
QQPlot	Residual Q-Q plot	PLOT=DIAGNOSTICS(UNPACK)
ResidualByPredicted	Residual by predicted values	PLOT=DIAGNOSTICS(UNPACK)
ResidualHistogram	Residual histogram	PLOT=DIAGNOSTICS(UNPACK)
RFPlot	RF plot	PLOT=DIAGNOSTICS(UNPACK)
ParmProfiles	Profiles of regression coefficients	PLOT=PARMPROFILES
R2Plot	R-square analysis (default, as appropriate)	
ResidualPlots	Residuals for each dependent variable	PLOT=RESIDUALS
VariableImportancePlot	Profile of variable importance factors	PLOT=VIP
XLoadingPlot	Scatter plot matrix of X-loadings against each other	PLOT=XLOADINGPLOT
XLoadingProfiles	Profiles of the X-loadings	PLOT=XLOADINGPROFILES
XScorePlot	Scatter plot matrix of X-scores against each other	PLOT=XSCORES
XWeightPlot	Scatter plot matrix of X-weights against each other	PLOT=XWEIGHTPLOT
XWeightProfiles	Profiles of the X-weights	PLOT=XWEIGHTPROFILES
XYScorePlot	Scatter plot matrix of X-scores against Y-scores	PLOT=XYSCORES
YScorePlot	Scatter plot matrix of Y-scores against each other	PLOT=YSCORES
YWeightPlot	Scatter plot matrix of Y-weights against each other	PLOT=YWEIGHTPLOT

Examples: PLS Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 95.1: Examining Model Details

(View the complete code for this example (plsex1.sas) in the example repository.)

This example, from Umetrics (1995), demonstrates different ways to examine a PLS model. The data come from the field of drug discovery. New drugs are developed from chemicals that are biologically active. Testing a compound for biological activity is an expensive procedure, so it is useful to be able to predict biological activity from cheaper chemical measurements. In fact, computational chemistry makes it possible to calculate certain chemical measurements without even making the compound. These measurements include size, lipophilicity, and polarity at various sites on the molecule. The following statements create a data set named pentaTrain, which contains these data.

```
data pentaTrain;
   input obsnam $ S1 L1 P1 S2 L2 P2
                  S3 L3 P3 S4 L4 P4
                  S5 L5 P5  log_RAI @@;
   n = _n_;
   datalines;
VESSK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
          1.9607 -1.6324  0.5746  1.9607 -1.6324  0.5746
          2.8369  1.4092 -3.1398                   0.00
VESAK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
          1.9607 -1.6324  0.5746  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   0.28
VEASK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
          0.0744 -1.7333  0.0902  1.9607 -1.6324  0.5746
          2.8369  1.4092 -3.1398                   0.20
VEAAK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
          0.0744 -1.7333  0.0902  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   0.51
VKAAK    -2.6931 -2.5271 -1.2871  2.8369  1.4092 -3.1398
          0.0744 -1.7333  0.0902  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   0.11
VEWAK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
         -4.7548  3.6521  0.8524  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   2.73
VEAAP    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
          0.0744 -1.7333  0.0902  0.0744 -1.7333  0.0902
         -1.2201  0.8829  2.2253                   0.18
VEHAK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
          2.4064  1.7438  1.1057  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   1.53
VAAAK    -2.6931 -2.5271 -1.2871  0.0744 -1.7333  0.0902
          0.0744 -1.7333  0.0902  0.0744 -1.7333  0.0902
```

```
          2.8369  1.4092 -3.1398                  -0.10
GEAAK     2.2261 -5.3648  0.3049  3.0777  0.3891 -0.0701
          0.0744 -1.7333  0.0902  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                  -0.52
LEAAK    -4.1921 -1.0285 -0.9801  3.0777  0.3891 -0.0701
          0.0744 -1.7333  0.0902  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   0.40
FEAAK    -4.9217  1.2977  0.4473  3.0777  0.3891 -0.0701
          0.0744 -1.7333  0.0902  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   0.30
VEGGK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
          2.2261 -5.3648  0.3049  2.2261 -5.3648  0.3049
          2.8369  1.4092 -3.1398                  -1.00
VEFAK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
         -4.9217  1.2977  0.4473  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   1.57
VELAK    -2.6931 -2.5271 -1.2871  3.0777  0.3891 -0.0701
         -4.1921 -1.0285 -0.9801  0.0744 -1.7333  0.0902
          2.8369  1.4092 -3.1398                   0.59
;
```

You would like to study the relationship between these measurements and the activity of the compound, represented by the logarithm of the relative Bradykinin activating activity (log_RAI). Notice that these data consist of many predictors relative to the number of observations. Partial least squares is especially appropriate in this situation as a useful tool for finding a few underlying predictive factors that account for most of the variation in the response. Typically, the model is fit for part of the data (the "training" or "work" set), and the quality of the fit is judged by how well it predicts the other part of the data (the "test" or "prediction" set). For this example, the first 15 observations serve as the training set and the rest constitute the test set (see Ufkes et al. 1978, 1982).

When you fit a PLS model, you hope to find a few PLS factors that explain most of the variation in both predictors and responses. Factors that explain response variation provide good predictive models for new responses, and factors that explain predictor variation are well represented by the observed values of the predictors. The following statements fit a PLS model with two factors and save predicted values, residuals, and other information for each data point in a data set named outpls.

```
proc pls data=pentaTrain;
   model log_RAI = S1-S5 L1-L5 P1-P5;
run;
```

The PLS procedure displays a table, shown in Output 95.1.1, showing how much predictor and response variation is explained by each PLS factor.

Output 95.1.1 Amount of Training Set Variation Explained

The PLS Procedure

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	16.9014	16.9014	89.6399	89.6399
2	12.7721	29.6735	7.8368	97.4767
3	14.6554	44.3289	0.4636	97.9403
4	11.8421	56.1710	0.2485	98.1889
5	10.5894	66.7605	0.1494	98.3383
6	5.1876	71.9481	0.2617	98.6001
7	6.1873	78.1354	0.2428	98.8428
8	7.2252	85.3606	0.1926	99.0354
9	6.7285	92.0891	0.0725	99.1080
10	7.9076	99.9967	0.0000	99.1080
11	0.0033	100.0000	0.0099	99.1179
12	0.0000	100.0000	0.0000	99.1179
13	0.0000	100.0000	0.0000	99.1179
14	0.0000	100.0000	0.0000	99.1179
15	0.0000	100.0000	0.0000	99.1179

From Output 95.1.1, note that 97% of the response variation is already explained by just two factors, but only 29% of the predictor variation is explained.

The graphics in PROC PLS, available when ODS Graphics is enabled, make it easier to see features of the PLS model.

If ODS Graphics is enabled, then in addition to the tables discussed previously, PROC PLS displays a graphical depiction of the R-square analysis as well as a correlation loading plot summarizing the model based on the first two PLS factors. The following statements perform the previous analysis with ODS Graphics enabled, producing Output 95.1.2 and Output 95.1.3.

```
ods graphics on;

proc pls data=pentaTrain;
   model log_RAI = S1-S5 L1-L5 P1-P5;
run;
```

Output 95.1.2 Plot of Proportion of Variation Accounted For

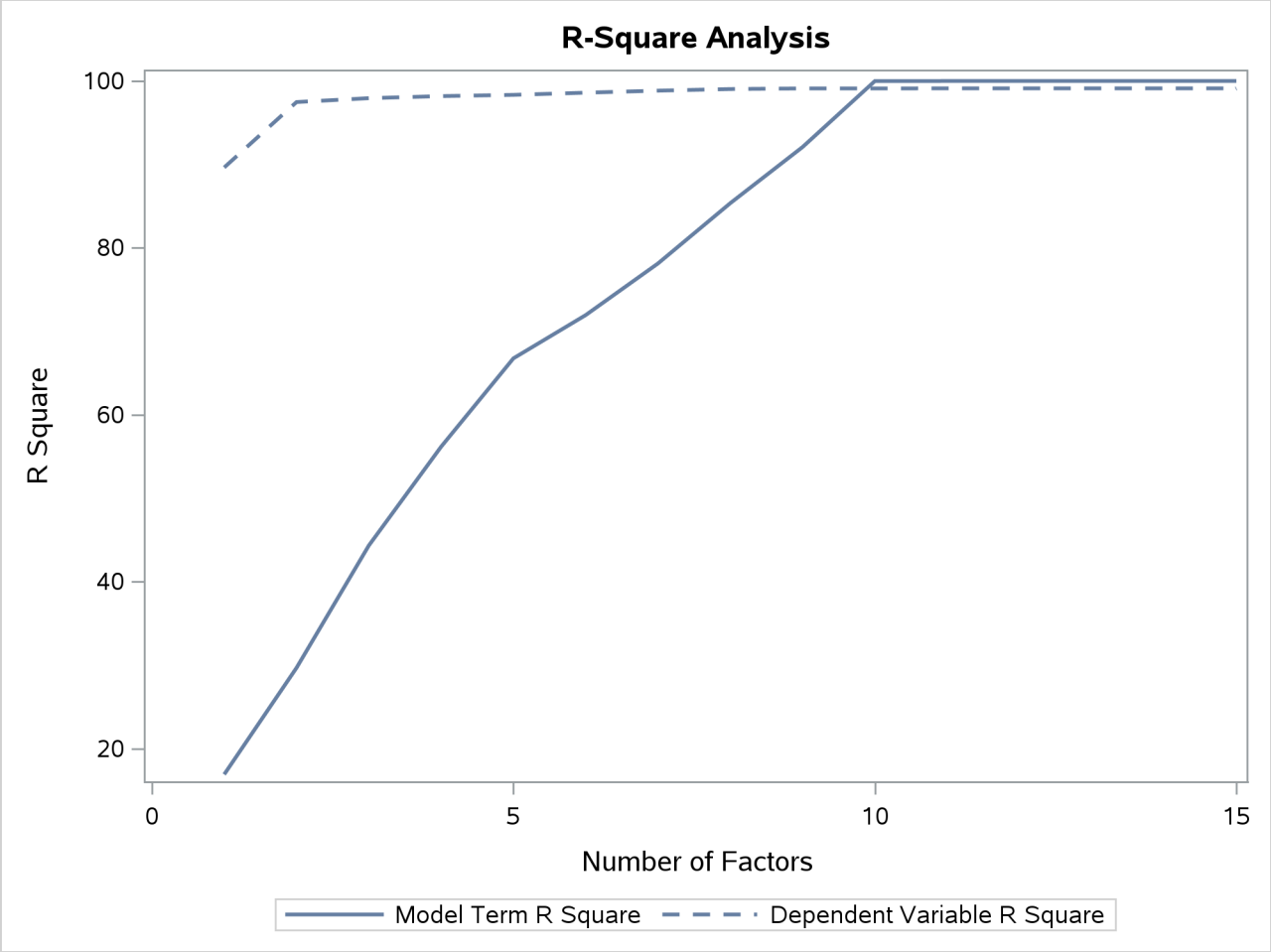

Output 95.1.3 Correlation Loading Plot

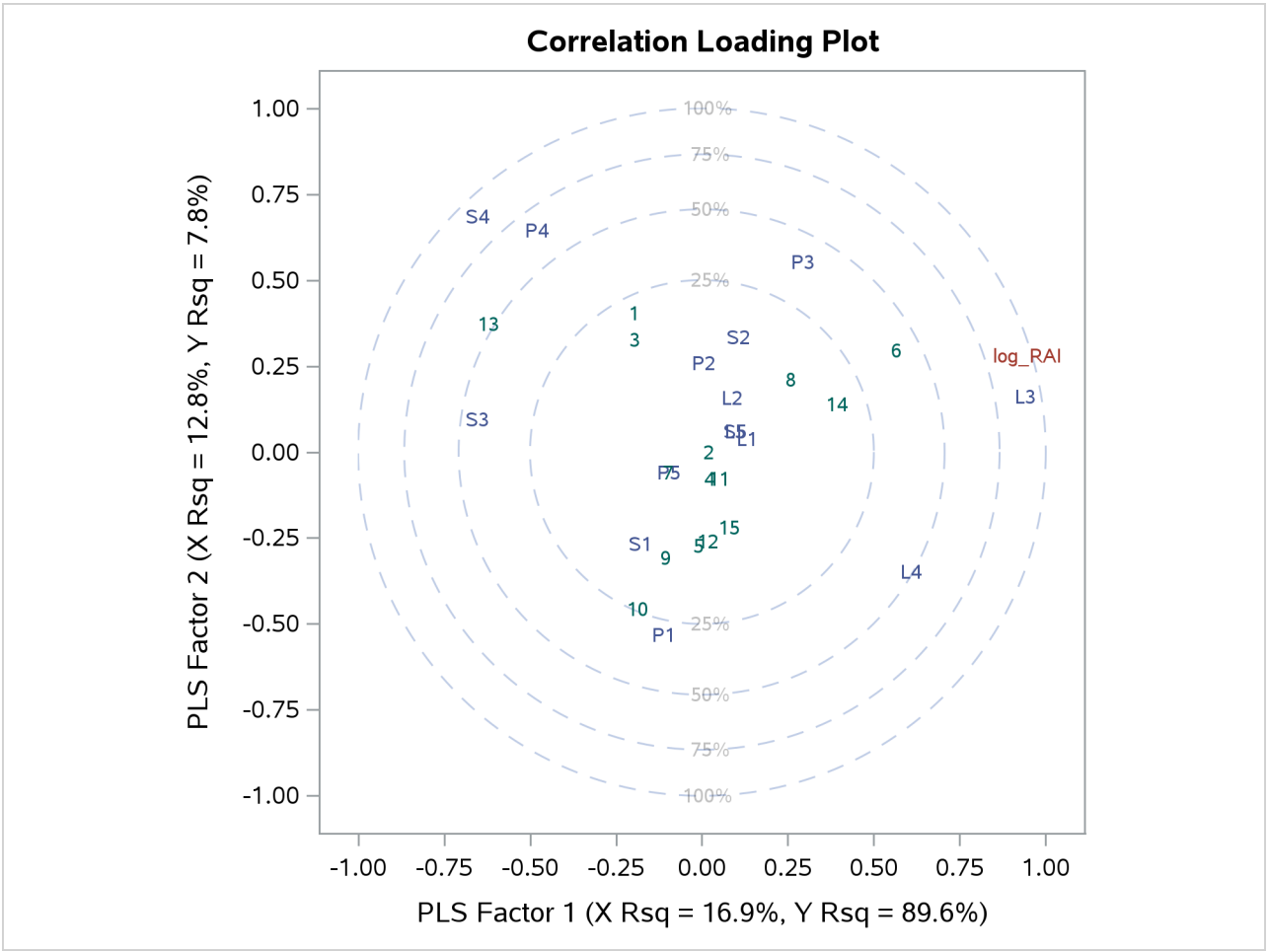

The plot in Output 95.1.2 of the proportion of variation explained (or R square) makes it clear that there is a plateau in the response variation after two factors are included in the model. The correlation loading plot in Output 95.1.3 summarizes many features of this two-factor model, including the following:

- The X-scores are plotted as numbers for each observation. You should look for patterns or clearly grouped observations. If you see a curved pattern, for example, you might want to add a quadratic term. Two or more groupings of observations indicate that it might be better to analyze the groups separately, perhaps by including classification effects in the model. This plot appears to show most of the observations close together, with a few being more spread out with larger positive X-scores for factor 2. There are no clear grouping patterns, but observation 13 stands out.
- The loadings show how much variation in each variable is accounted for by the first two factors, jointly by the distance of the corresponding point from the origin and individually by the distance for the projections of this point onto the horizontal and vertical axes. That the dependent variable is well explained by the model is reflected in the fact that the point for log_RAI is near the 100% circle.
- You can also use the projection interpretation to relate variables to each other. For example, projecting other variables' points onto the line that runs through the log_RAI point and the origin, you can see that the PLS approximation for the predictor L3 is highly positively correlated with log_RAI, S3 is

somewhat less correlated but in the negative direction, and several predictors including L1, L5, and S5 have very little correlation with log_RAI.

Other graphics enable you to explore more of the features of the PLS model. For example, you can examine the X-scores versus the Y-scores to explore how partial least squares chooses successive factors. For a good PLS model, the first few factors show a high correlation between the X- and Y-scores. The correlation usually decreases from one factor to the next. When ODS Graphics is enabled, you can plot the X-scores versus the Y-scores by using the PLOT=XYSCORES option, as shown in the following statements.

```
proc pls data=pentaTrain nfac=4 plot=XYScores;
   model log_RAI = S1-S5 L1-L5 P1-P5;
run;
```

The plot of the X-scores versus the Y-scores for the first four factors is shown in Output 95.1.4.

Output 95.1.4 X-Scores versus Y-Scores

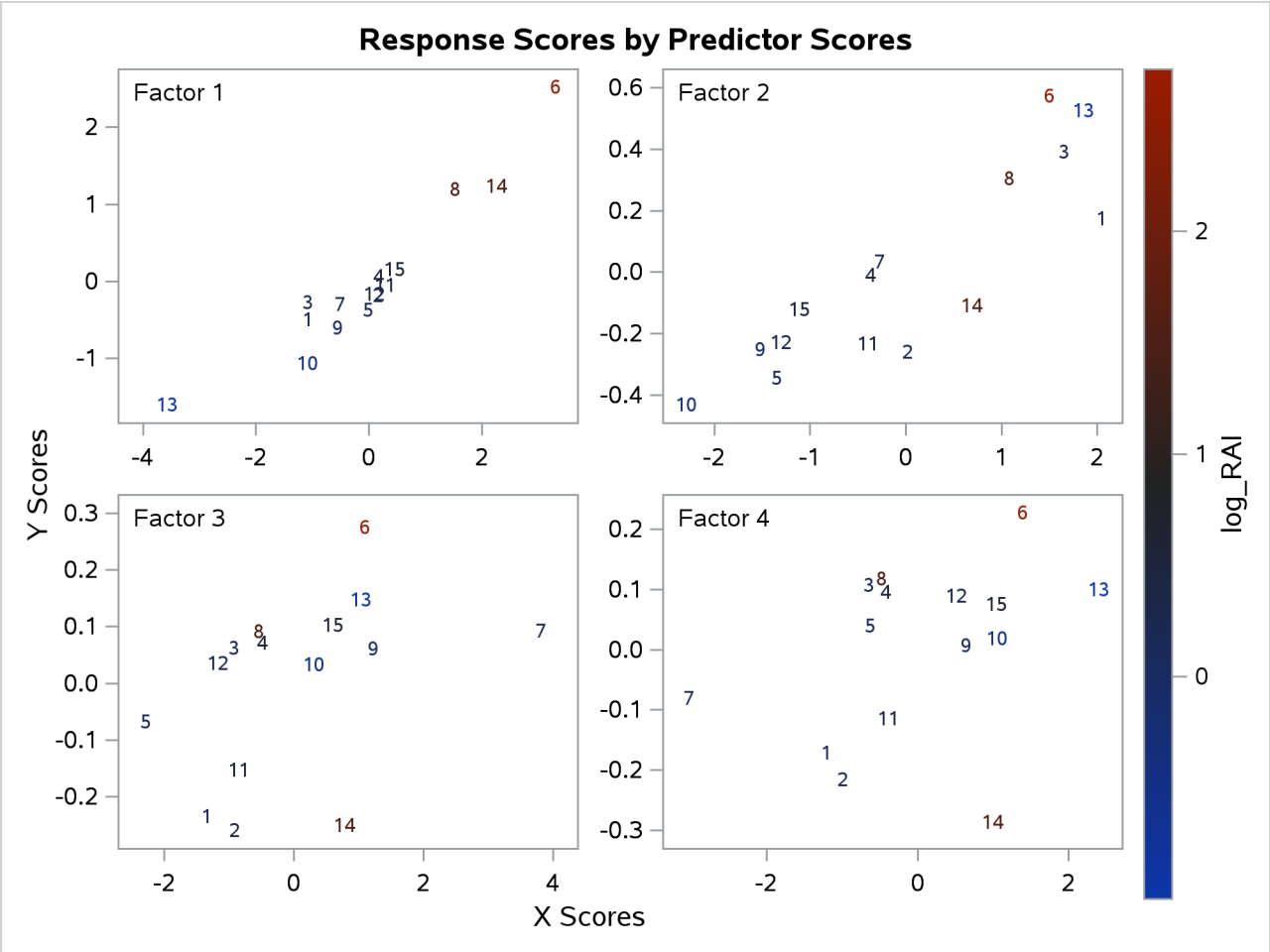

For this example, Output 95.1.4 shows high correlation between X- and Y-scores for the first factor but somewhat lower correlation for the second factor and sharply diminishing correlation after that. This adds strength to the judgment that NFAC=2 is the right number of factors for these data and this model. Note that observation 13 is again extreme in the first two plots. This run might be overly influential for the PLS

analysis; thus, you should check to make sure it is reliable.

As explained earlier, you can draw some inferences about the relationship between individual predictors and the dependent variable from the correlation loading plot. However, the regression coefficient profile and the variable importance plot give a more direct indication of which predictors are most useful for predicting the dependent variable. The regression coefficients represent the importance each predictor has in the prediction of just the response. The variable importance plot, on the other hand, represents the contribution of each predictor in fitting the PLS model for both predictors and response. It is based on the *Variable Importance for Projection* (VIP) statistic of Wold (1994), which summarizes the contribution a variable makes to the model. If a predictor has a relatively small coefficient (in absolute value) *and* a small value of VIP, then it is a prime candidate for deletion. Wold in Umetrics (1995) considers a value less than 0.8 to be "small" for the VIP. The following statements fit a two-factor PLS model and display these two additional plots.

```
proc pls data=pentaTrain nfac=2 plot=(ParmProfiles VIP);
   model log_RAI = S1-S5 L1-L5 P1-P5;
run;

ods graphics off;
```

The additional graphics are shown in Output 95.1.5 and Output 95.1.6.

Output 95.1.5 Variable Importance Plots

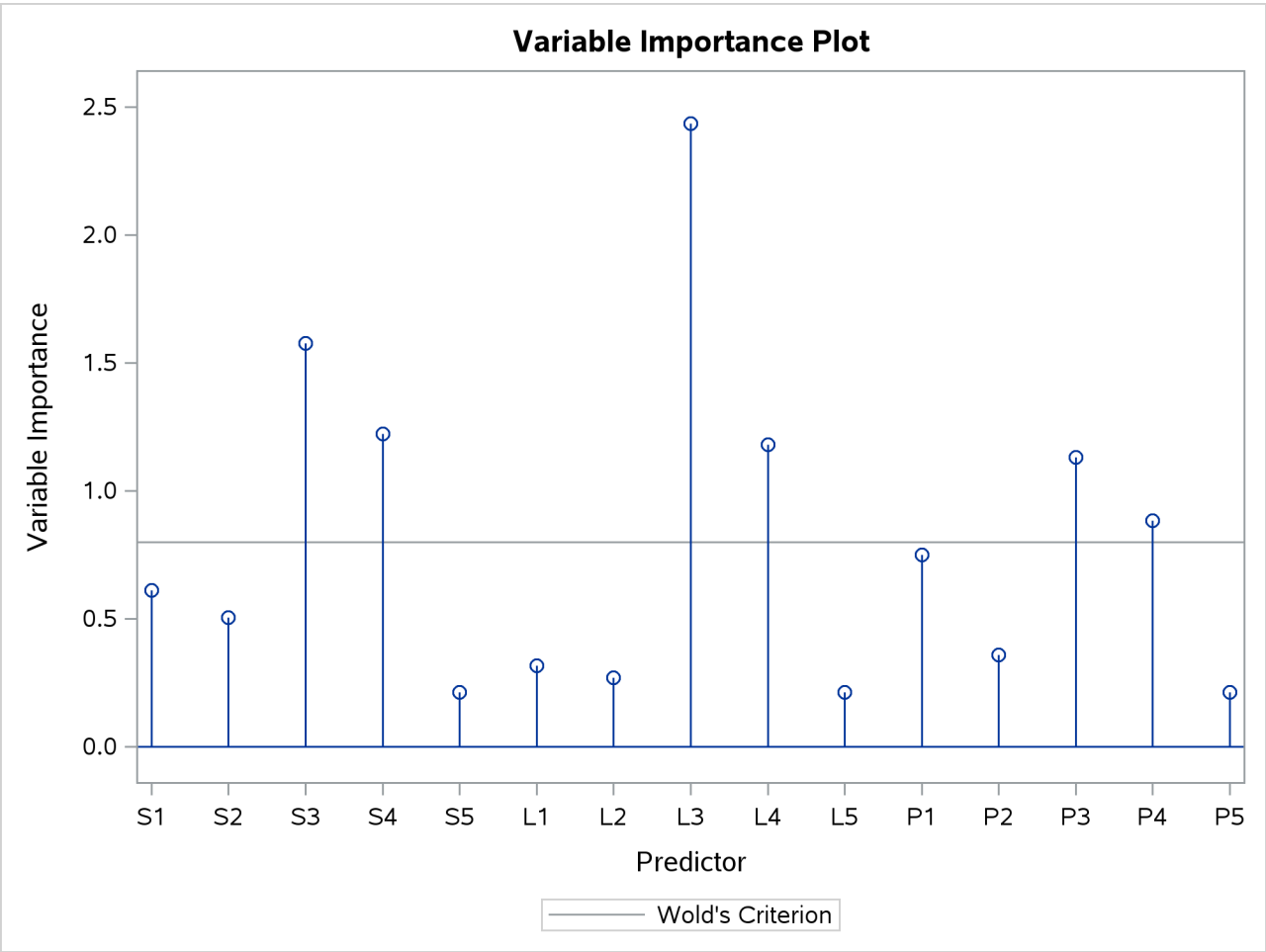

Output 95.1.6 Regression Parameter Profile

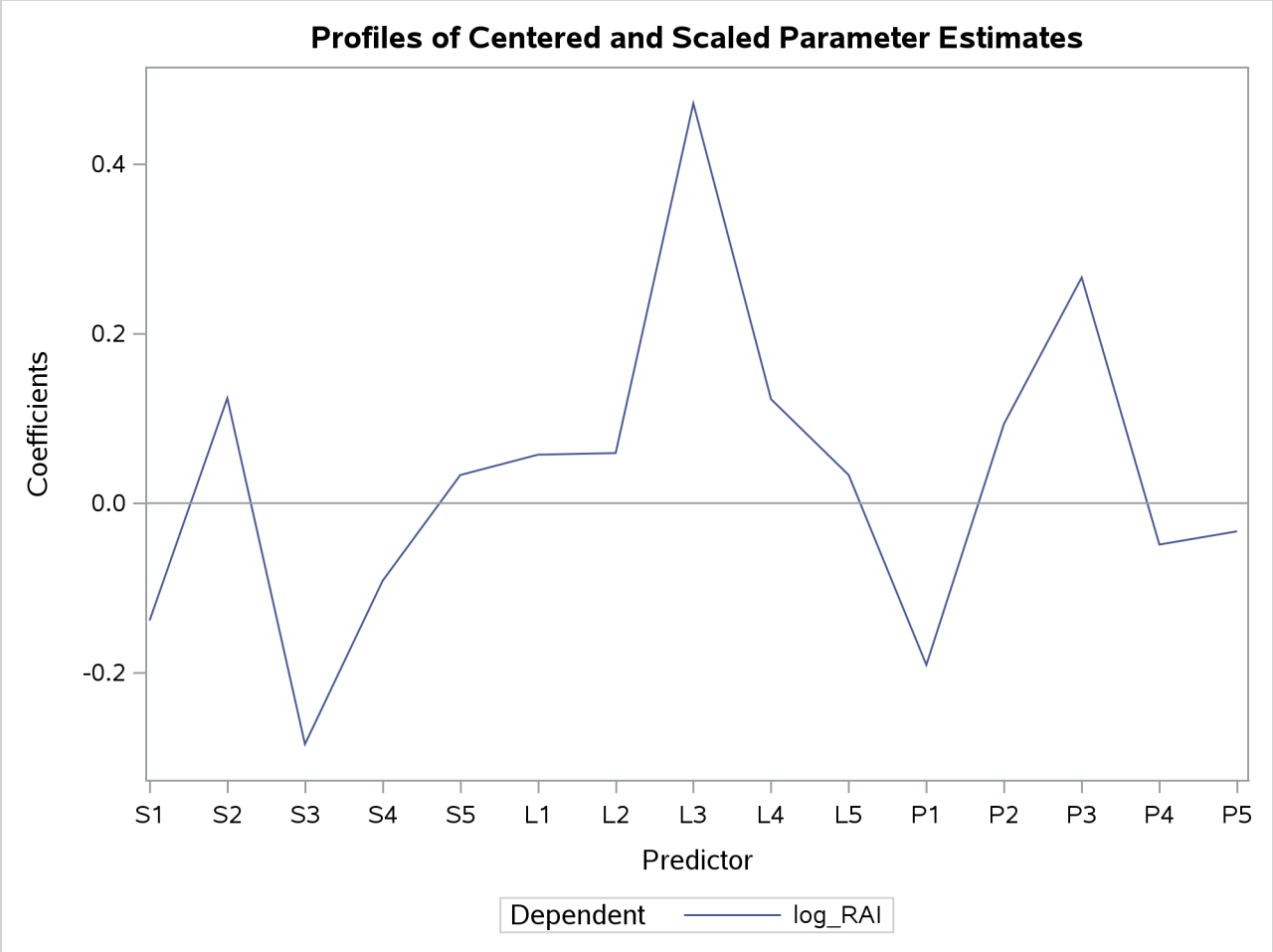

In these two plots, the variables L1, L2, P2, S5, L5, and P5 have small absolute coefficients and small VIP. Looking back at the correlation loading plot in Output 95.1.2, you can see that these variables tend to be the ones near zero for both PLS factors. You should consider dropping these variables from the model.

Example 95.2: Examining Outliers

(View the complete code for this example (plsex2.sas) in the example repository.)

This example is a continuation of Example 95.1.

Standard diagnostics for statistical models focus on the response, allowing you to look for patterns that indicate the model is inadequate or for outliers that do not seem to follow the trend of the rest of the data. However, partial least squares effectively models the predictors as well as the responses, so you should consider the pattern of the fit for both. The DModX and DModY statistics give the distance from each point to the PLS model with respect to the predictors and the responses, respectively, and ODS Graphics enables you to plot these values. No point should be dramatically farther from the model than the rest. If there is a group of points that are all farther from the model than the rest, they might have something in common, in which case they should be analyzed separately.

The following statements fit a reduced model to the data discussed in Example 95.1 and plot a panel of standard diagnostics as well as the distances of the observations to the model.

```
ods graphics on;

proc pls data=pentaTrain nfac=2 plot=(diagnostics dmod);
   model log_RAI = S1    P1
                   S2
                   S3 L3 P3
                   S4 L4   ;
run;
```

The plots are shown in Output 95.2.1 and Output 95.2.2.

Output 95.2.1 Model Fit Diagnostics

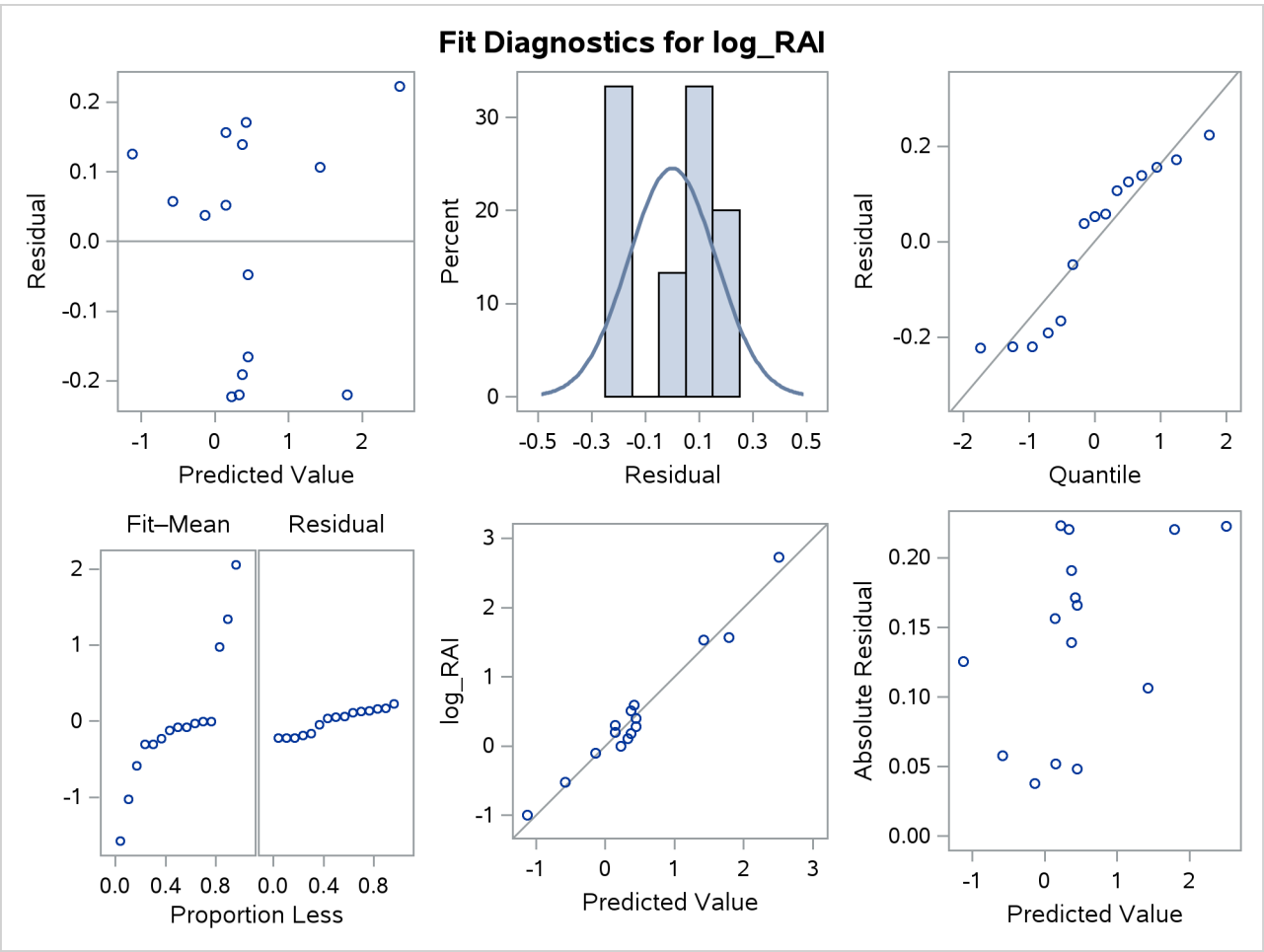

Output 95.2.2 Predictor versus Response Distances to the Model

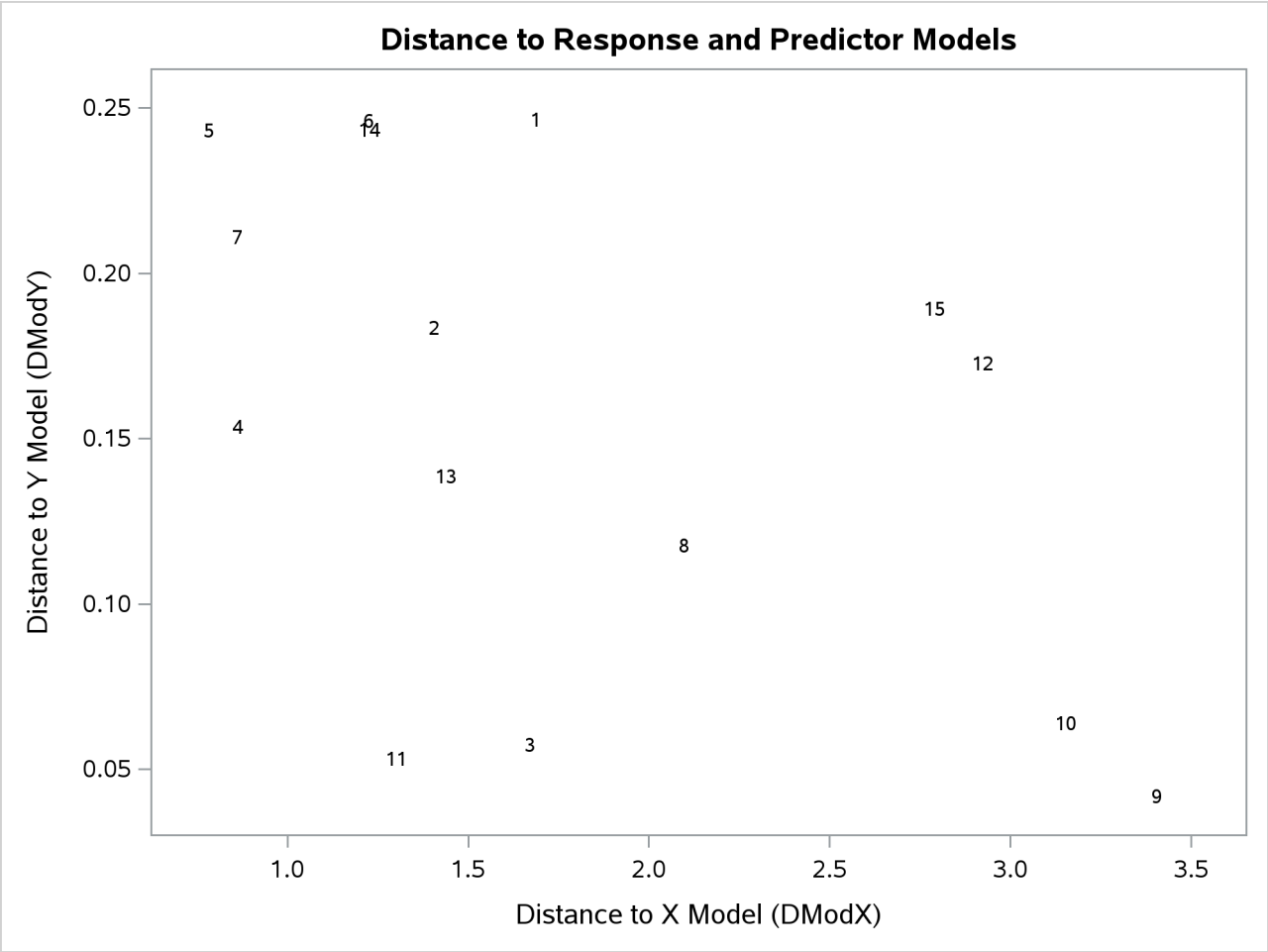

There appear to be no profound outliers in either the predictor space or the response space.

Example 95.3: Choosing a PLS Model by Test Set Validation

(View the complete code for this example (plsex3.sas) in the example repository.)

This example demonstrates issues in spectrometric calibration. The data (Umetrics 1995) consist of spectrographic readings on 33 samples containing known concentrations of two amino acids, tyrosine and tryptophan. The spectra are measured at 30 frequencies across the overall range of frequencies. For example, Figure 95.3.1 shows the observed spectra for three samples, one with only tryptophan, one with only tyrosine, and one with a mixture of the two, all at a total concentration of 10^{-6}.

Output 95.3.1 Spectra for Three Samples of Tyrosine and Tryptophan

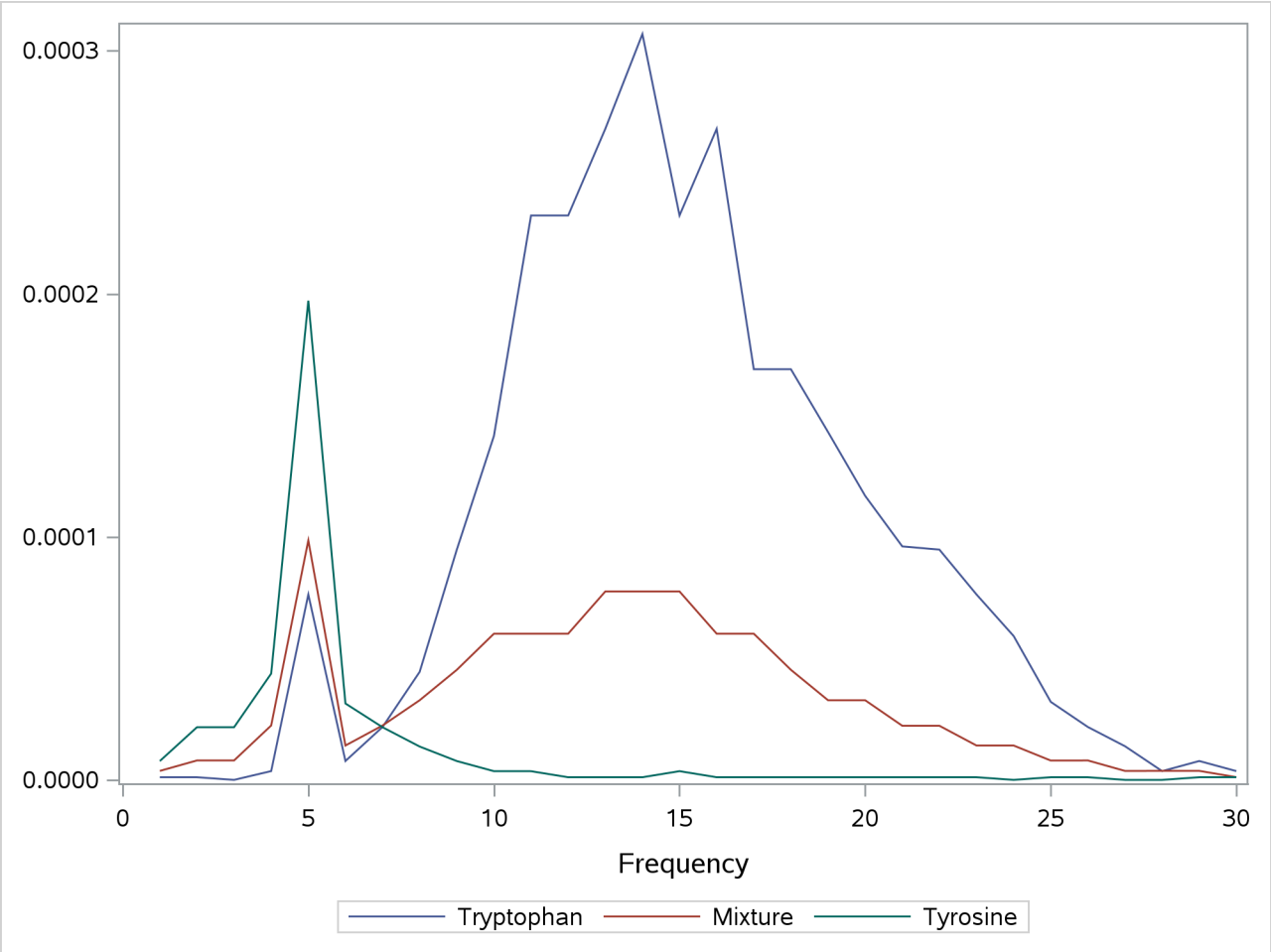

Of the 33 samples, 18 are used as a training set and 15 as a test set. The data originally appear in McAvoy et al. (1989).

These data were created in a lab, with the concentrations fixed in order to provide a wide range of applicability for the model. You want to use a linear function of the logarithms of the spectra to predict the logarithms of tyrosine and tryptophan concentration, as well as the logarithm of the total concentration. Actually, because of the possibility of zeros in both the responses and the predictors, slightly different transformations are used. The following statements create SAS data sets containing the training and test data, named ftrain and ftest, respectively.

```
data ftrain;
   input obsnam $ tot tyr f1-f30 @@;
   try = tot - tyr;
   if (tyr) then tyr_log = log10(tyr); else tyr_log = -8;
   if (try) then try_log = log10(try); else try_log = -8;
   tot_log = log10(tot);
   datalines;
17mix35 0.00003 0
 -6.215 -5.809 -5.114 -3.963 -2.897 -2.269 -1.675 -1.235
 -0.900 -0.659 -0.497 -0.395 -0.335 -0.315 -0.333 -0.377
```

```
 -0.453 -0.549 -0.658 -0.797 -0.878 -0.954 -1.060 -1.266
 -1.520 -1.804 -2.044 -2.269 -2.496 -2.714
19mix35 0.00003 3E-7
 -5.516 -5.294 -4.823 -3.858 -2.827 -2.249 -1.683 -1.218
 -0.907 -0.658 -0.501 -0.400 -0.345 -0.323 -0.342 -0.387
 -0.461 -0.554 -0.665 -0.803 -0.887 -0.960 -1.072 -1.272
 -1.541 -1.814 -2.058 -2.289 -2.496 -2.712
21mix35 0.00003 7.5E-7
 -5.519 -5.294 -4.501 -3.863 -2.827 -2.280 -1.716 -1.262
 -0.939 -0.694 -0.536 -0.444 -0.384 -0.369 -0.377 -0.421
 -0.495 -0.596 -0.706 -0.824 -0.917 -0.988 -1.103 -1.294
 -1.565 -1.841 -2.084 -2.320 -2.521 -2.729
23mix35 0.00003 1.5E-6

   ... more lines ...

mix6    0.0001 0.00009
 -1.140 -0.757 -0.497 -0.362 -0.329 -0.412 -0.513 -0.647
 -0.772 -0.877 -0.958 -1.040 -1.104 -1.162 -1.233 -1.317
 -1.425 -1.543 -1.661 -1.804 -1.877 -1.959 -2.034 -2.249
 -2.502 -2.732 -2.964 -3.142 -3.313 -3.576
;

data ftest;
   input obsnam $ tot tyr f1-f30 @@;
   try = tot - tyr;
   if (tyr) then tyr_log = log10(tyr); else tyr_log = -8;
   if (try) then try_log = log10(try); else try_log = -8;
   tot_log = log10(tot);
   datalines;
43trp6  1E-6 0
 -5.915 -5.918 -6.908 -5.428 -4.117 -5.103 -4.660 -4.351
 -4.023 -3.849 -3.634 -3.634 -3.572 -3.513 -3.634 -3.572
 -3.772 -3.772 -3.844 -3.932 -4.017 -4.023 -4.117 -4.227
 -4.492 -4.660 -4.855 -5.428 -5.103 -5.428
59mix6  1E-6 1E-7
 -5.903 -5.903 -5.903 -5.082 -4.213 -5.083 -4.838 -4.639
 -4.474 -4.213 -4.001 -4.098 -4.001 -4.001 -3.907 -4.001
 -4.098 -4.098 -4.206 -4.098 -4.213 -4.213 -4.335 -4.474
 -4.639 -4.838 -4.837 -5.085 -5.410 -5.410
51mix6  1E-6 2.5E-7
 -5.907 -5.907 -5.415 -4.843 -4.213 -4.843 -4.843 -4.483
 -4.343 -4.006 -4.006 -3.912 -3.830 -3.830 -3.755 -3.912
 -4.006 -4.001 -4.213 -4.213 -4.335 -4.483 -4.483 -4.642
 -4.841 -5.088 -5.088 -5.415 -5.415 -5.415
49mix6  1E-6 5E-7

   ... more lines ...

tyro2   0.0001 0.0001
 -1.081 -0.710 -0.470 -0.337 -0.327 -0.433 -0.602 -0.841
 -1.119 -1.423 -1.750 -2.121 -2.449 -2.818 -3.110 -3.467
 -3.781 -4.029 -4.241 -4.366 -4.501 -4.366 -4.501 -4.501
 -4.668 -4.668 -4.865 -4.865 -5.109 -5.111
```

```
;
```

The following statements fit a PLS model with 10 factors.

```
proc pls data=ftrain nfac=10;
   model tot_log tyr_log try_log = f1-f30;
run;
```

The table shown in Output 95.3.2 indicates that only three or four factors are required to explain almost all of the variation in both the predictors and the responses.

Output 95.3.2 Amount of Training Set Variation Explained

The PLS Procedure

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	81.1654	81.1654	48.3385	48.3385
2	16.8113	97.9768	32.5465	80.8851
3	1.7639	99.7407	11.4438	92.3289
4	0.1951	99.9357	3.8363	96.1652
5	0.0276	99.9634	1.6880	97.8532
6	0.0132	99.9765	0.7247	98.5779
7	0.0052	99.9817	0.2926	98.8705
8	0.0053	99.9870	0.1252	98.9956
9	0.0049	99.9918	0.1067	99.1023
10	0.0034	99.9952	0.1684	99.2707

In order to choose the optimal number of PLS factors, you can explore how well models based on the training data with different numbers of factors fit the test data. To do so, use the CV=TESTSET option, with an argument pointing to the test data set ftest. The following statements also employ the ODS Graphics features in PROC PLS to display the cross validation results in a plot.

```
ods graphics on;

proc pls data=ftrain nfac=10 cv=testset(ftest)
                             cvtest(stat=press seed=12345);
   model tot_log tyr_log try_log = f1-f30;
run;
```

The tabular results of the test set validation are shown in Output 95.3.3, and the graphical results are shown in Output 95.3.4. They indicate that, although five PLS factors give the minimum predicted residual sum of squares, the residuals for four factors are insignificantly different from those for five. Thus, the smaller model is preferred.

Output 95.3.3 Test Set Validation for the Number of PLS Factors

The PLS Procedure

Test Set Validation for the Number of Extracted Factors

Number of Extracted Factors	Root Mean PRESS	Prob > PRESS
0	3.056797	<.0001
1	2.630561	<.0001
2	1.00706	0.0070
3	0.664603	0.0020
4	0.521578	0.3800
5	0.500034	1.0000
6	0.513561	0.5100
7	0.501431	0.6870
8	1.055791	0.1530
9	1.435085	0.1010
10	1.720389	0.0320

Minimum root mean PRESS	0.5000
Minimizing number of factors	5
Smallest number of factors with p > 0.1	4

Percent Variation Accounted for by Partial Least Squares Factors

	Model Effects		Dependent Variables	
Number of Extracted Factors	**Current**	**Total**	**Current**	**Total**
1	81.1654	81.1654	48.3385	48.3385
2	16.8113	97.9768	32.5465	80.8851
3	1.7639	99.7407	11.4438	92.3289
4	0.1951	99.9357	3.8363	96.1652

Output 95.3.4 Test Set Validation Plot

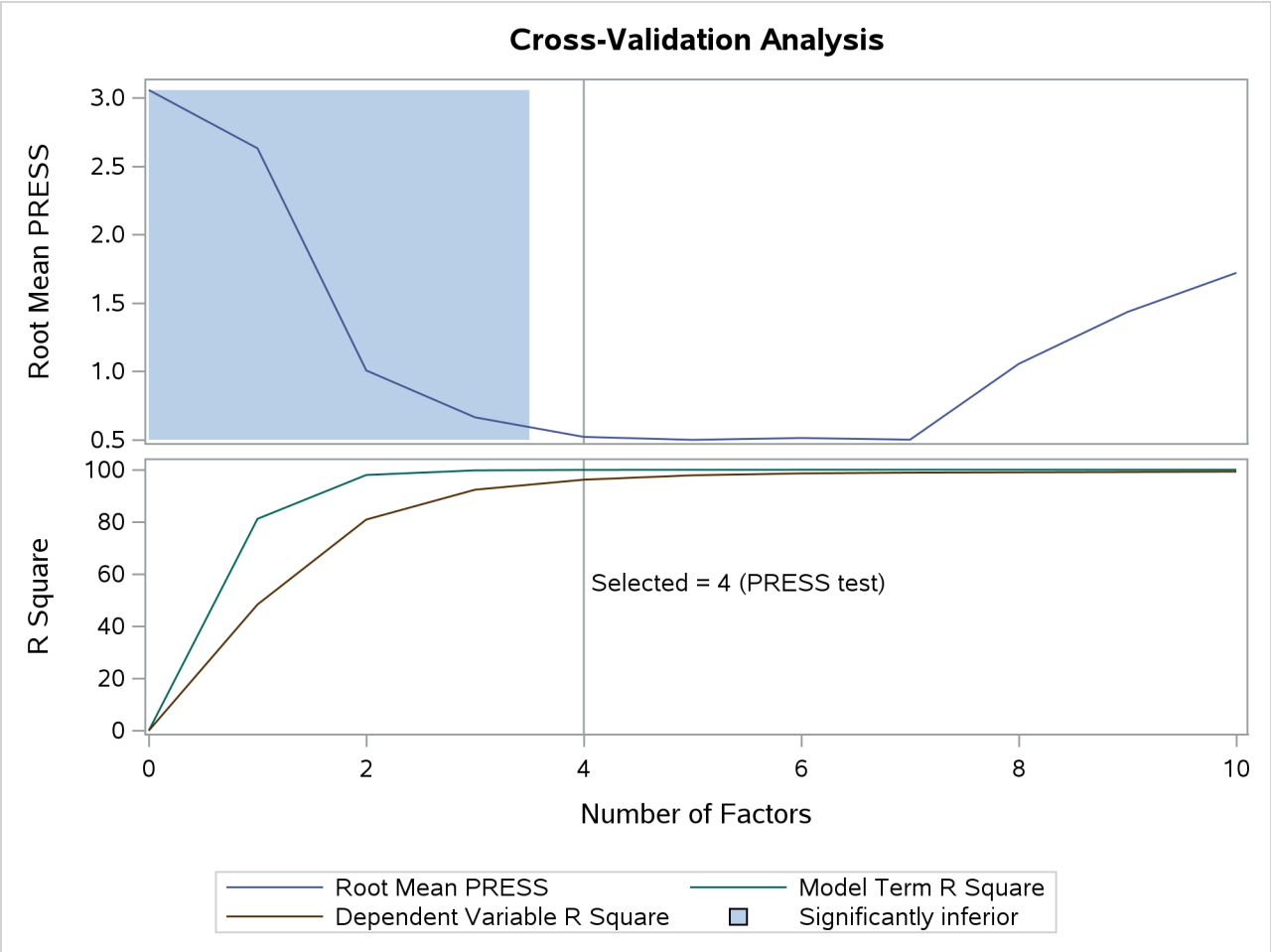

The factor loadings show how the PLS factors are constructed from the centered and scaled predictors. For spectral calibration, it is useful to plot the loadings against the frequency. In many cases, the physical meanings that can be attached to factor loadings help to validate the scientific interpretation of the PLS model. You can use ODS Graphics with PROC PLS to plot the loadings for the four PLS factors against frequency, as shown in the following statements.

```
proc pls data=ftrain nfac=4 plot=XLoadingProfiles;
   model tot_log tyr_log try_log = f1-f30;
run;

ods graphics off;
```

The resulting plot is shown in Output 95.3.5.

Output 95.3.5 Predictor Loadings across Frequencies

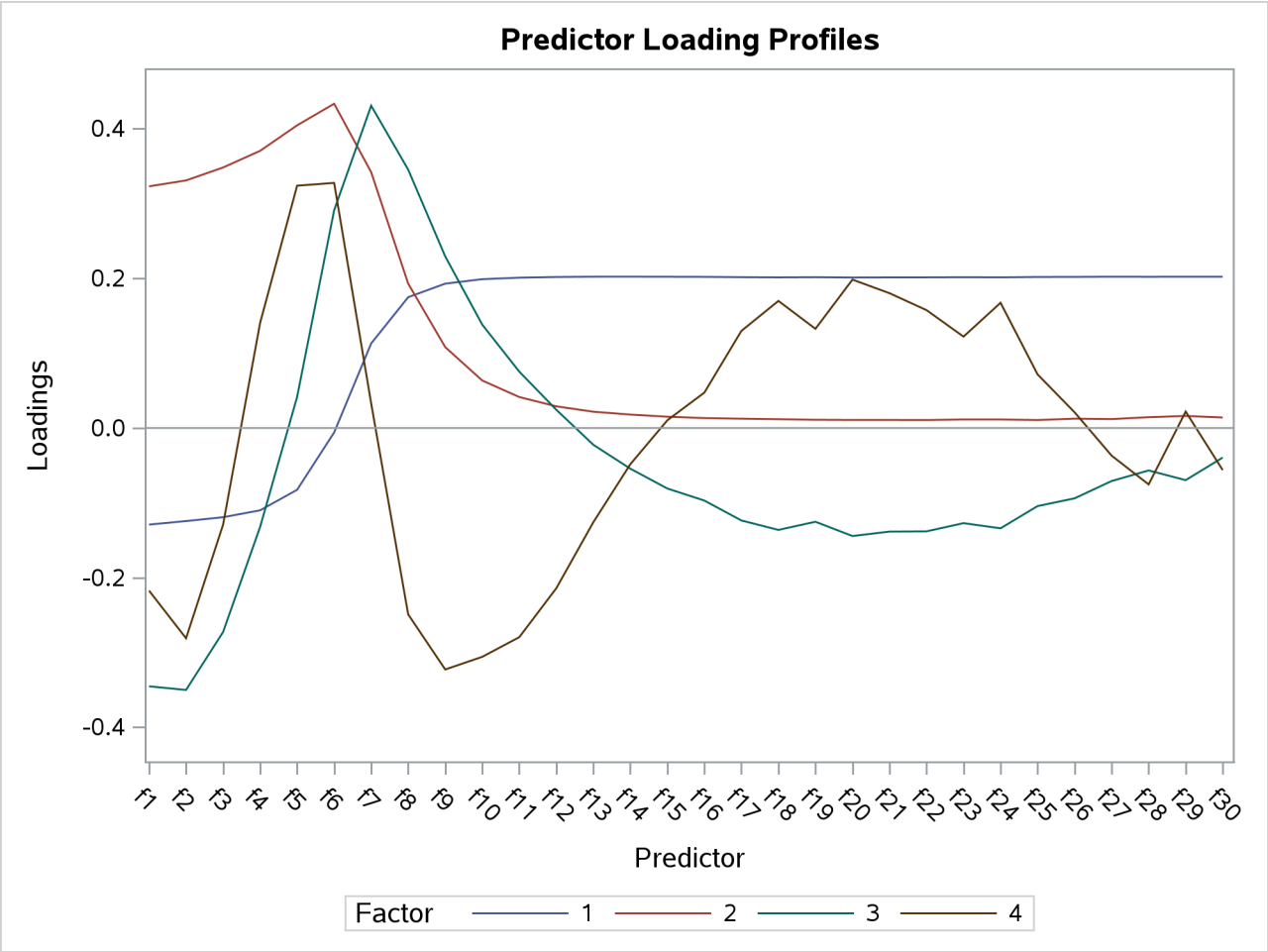

Notice that all four factors handle frequencies below and above about 7 or 8 differently. For example, the first factor is very nearly a simple contrast between the averages of the two sets of frequencies, and the second factor appears to be approximately a weighted sum of only the frequencies in the first set.

Example 95.4: Partial Least Squares Spline Smoothing

(View the complete code for this example (plsex4.sas) in the example repository.)

The EFFECT statement makes it easy to construct a wide variety of linear models. In particular, you can use the spline effect to add smoothing terms to a model. A particular benefit of using spline effects in PROC PLS is that, when operating on spline basis functions, the partial least squares algorithm effectively chooses the amount of smoothing automatically, especially if you combine it with cross validation for the selecting the number of factors. This example employs the EFFECT statement to demonstrate partial least squares spline smoothing of agricultural data.

Weibe (1935) presents data from a study of uniformity of wheat yields over a certain rectangular plot of land. The following statements read these wheat yield measurements, indexed by row and column distances, into the SAS data set Wheat:

```
data Wheat;
   keep Row Column Yield;
   input Yield @@;
   iRow = int((_N_-1)/12);
   iCol = mod( _N_-1 ,12);
   Column = iCol*15 + 1; /* Column distance, in feet */
   Row    = iRow* 1 + 1; /* Row    distance, in feet */
   Row = 125 - Row + 1;  /* Invert rows              */
   datalines;
 715 595 580 580 615 610 540 515 557 665 560 612
 770 710 655 675 700 690 565 585 550 574 511 618
 760 715 690 690 655 725 665 640 665 705 644 705
 665 615 685 555 585 630 550 520 553 616 573 570
 755 730 670 580 545 620 580 525 495 565 599 612
 745 670 585 560 550 710 590 545 538 587 600 664
 645 690 550 520 450 630 535 505 530 536 611 578
 585 495 455 470 445 555 500 450 420 461 531 559

   ... more lines ...

 570 585 635 765 550 675 765 620 608 705 677 660
 505 500 580 655 470 565 570 555 537 585 589 619
 465 430 510 680 460 600 670 615 620 594 616 784
;
```

The following statements use the PLS procedure to smooth these wheat yields using two spline effects, one for rows and another for columns, in addition to their crossproduct. Each spline effect has, by default, seven basis columns; thus their crossproduct has $49 = 7^2$ columns, for a total of 63 parameters in the full linear model. However, the predictive PLS model does not actually need to have 63 degrees of freedom. Rather, the degree of smoothing is controlled by the number of PLS factors, which in this case is chosen automatically by random subset validation with the CV=RANDOM option.

```
ods graphics on;

proc pls data=Wheat cv=random(seed=1) cvtest(seed=12345)
         plot(only)=contourfit(obs=gradient);
   effect splCol = spline(Column);
   effect splRow = spline(Row   );
   model Yield = splCol|splRow;
run;
```

These statements produce the output shown in Output 95.4.1 through Output 95.4.4.

Output 95.4.1 Default Spline Basis: Model and Data Information

The PLS Procedure

Data Set	WORK.WHEAT
Factor Extraction Method	Partial Least Squares
PLS Algorithm	NIPALS
Number of Response Variables	1
Number of Predictor Parameters	63
Missing Value Handling	Exclude
Maximum Number of Factors	15
Validation Method	10-fold Random Subset Validation
Random Subset Seed	1
Validation Testing Criterion	Prob T**2 > 0.1
Number of Random Permutations	1000
Random Permutation Seed	12345

Number of Observations Read	1500
Number of Observations Used	1500

Output 95.4.2 Default Spline Basis: Random Subset Validated PRESS Statistics for Number of Factors

Random Subset Validation for the Number of Extracted Factors

Number of Extracted Factors	Root Mean PRESS	T**2	Prob > T**2
0	1.066355	251.8793	<.0001
1	0.826177	123.8161	<.0001
2	0.745877	61.6035	<.0001
3	0.725181	44.99644	<.0001
4	0.701464	23.20199	<.0001
5	0.687164	8.369711	0.0030
6	0.683917	8.775847	0.0010
7	0.677969	2.907019	0.0830
8	0.676423	2.190871	0.1340
9	0.676966	3.191284	0.0600
10	0.675026	1.334638	0.2390
11	0.673906	0.556455	0.4470
12	0.673653	1.257292	0.2790
13	0.672669	0	1.0000
14	0.673596	2.386014	0.1190
15	0.672828	0.02962	0.8820

Minimum root mean PRESS	0.6727
Minimizing number of factors	13
Smallest number of factors with p > 0.1	8

Output 95.4.3 Default Spline Basis: PLS Variation Summary for Split-Sample Validated Model

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	11.5269	11.5269	40.2471	40.2471
2	7.2314	18.7583	10.4908	50.7379
3	6.9147	25.6730	2.6523	53.3902
4	3.8433	29.5163	2.8806	56.2708
5	6.4795	35.9958	1.3197	57.5905
6	7.6201	43.6159	1.1700	58.7605
7	7.3214	50.9373	0.7186	59.4790
8	4.8363	55.7736	0.4548	59.9339

Output 95.4.4 Default Spline Basis: Smoothed Yield

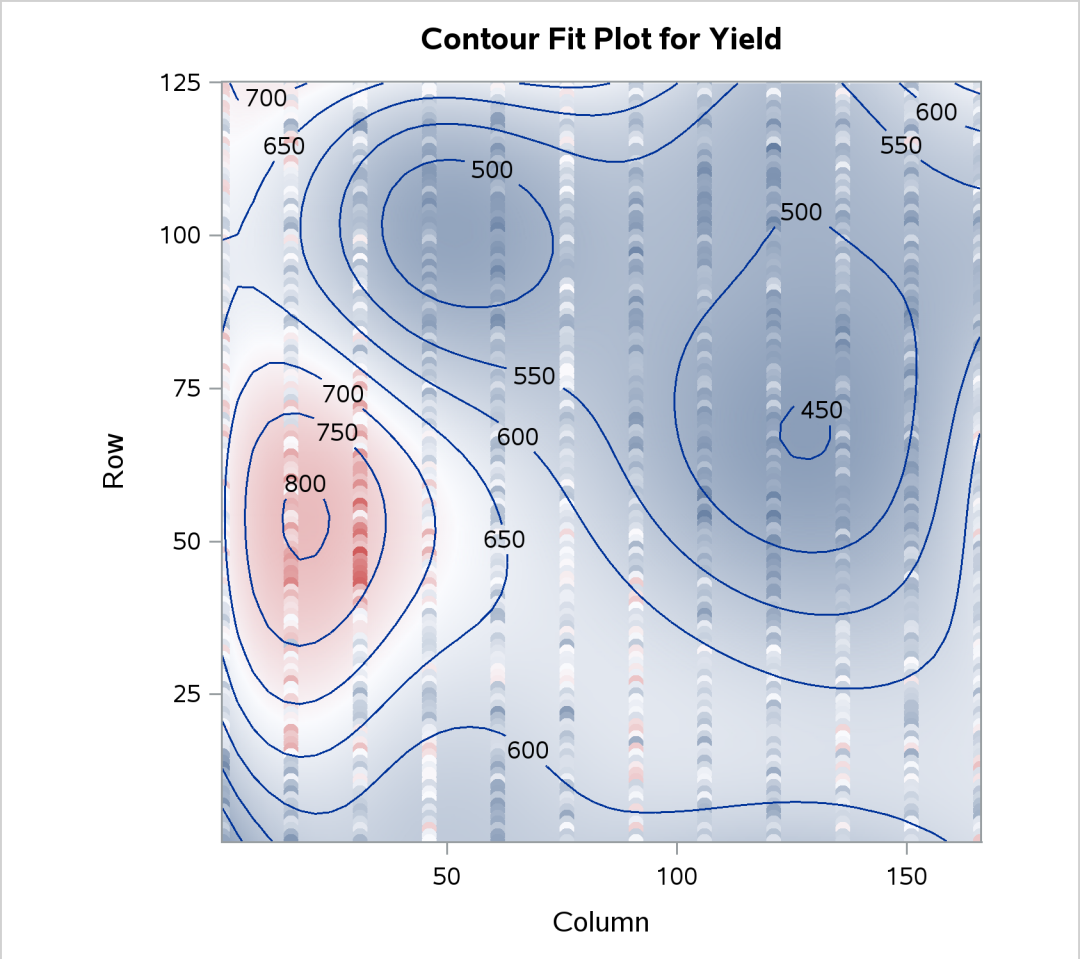

The cross validation results in Output 95.4.2 point to a model with eight PLS factors; this is the smallest

model whose predicted residual sum of squares (PRESS) is insignificantly different from the model with the absolute minimum PRESS. The variation summary in Output 95.4.3 shows that this model accounts for about 60% of the variation in the Yield values. The OBS=GRADIENT suboption for the PLOT=CONTOURFIT option specifies that the observations in the resulting plot, Output 95.4.4, be colored according to the same scheme as the surface of predicted yield. This coloration enables you to easily tell which observations are above the surface of predicted yield and which are below.

The surface of predicted yield is somewhat smoother than what Weibe (1935) settled on originally, with a predominance of simple, elliptically shaped contours. You can easily specify a potentially more granular model by increasing the number of knots in the spline bases. Even though the more granular model increases the number of predictor parameters, cross validation can still protect you from overfitting the data. The following statements are the same as those shown before, except that the spline effects now have twice as many basis functions:

```
ods graphics on;

proc pls data=Wheat cv=random(seed=1) cvtest(seed=12345)
         plot(only)=contourfit(obs=gradient);
   effect splCol = spline(Column / knotmethod=equal(14));
   effect splRow = spline(Row    / knotmethod=equal(14));
   model Yield = splCol|splRow;
run;
```

The resulting output is shown in Output 95.4.5 through Output 95.4.8.

Output 95.4.5 More Granular Spline Basis: Model and Data Information

The PLS Procedure

Data Set	WORK.WHEAT
Factor Extraction Method	Partial Least Squares
PLS Algorithm	NIPALS
Number of Response Variables	1
Number of Predictor Parameters	360
Missing Value Handling	Exclude
Maximum Number of Factors	15
Validation Method	10-fold Random Subset Validation
Random Subset Seed	1
Validation Testing Criterion	Prob T**2 > 0.1
Number of Random Permutations	1000
Random Permutation Seed	12345

Number of Observations Read	1500
Number of Observations Used	1500

Output 95.4.6 More Granular Spline Basis: Random Subset Validated PRESS Statistics for Number of Factors

Random Subset Validation for the Number of Extracted Factors			
Number of Extracted Factors	Root Mean PRESS	T**2	Prob > T**2
0	1.066355	247.9268	<.0001
1	0.652658	20.68858	<.0001
2	0.615087	0.074822	0.7740
3	0.614128	0	1.0000
4	0.615268	0.197678	0.6490
5	0.618001	1.372038	0.2340
6	0.622949	5.035504	0.0180
7	0.626482	7.296797	0.0080
8	0.633316	13.66045	<.0001
9	0.635239	16.16922	<.0001
10	0.636938	18.02295	<.0001
11	0.636494	16.9881	<.0001
12	0.63682	16.83341	<.0001
13	0.637719	16.74157	<.0001
14	0.637627	15.79342	<.0001
15	0.638431	16.12327	<.0001

Minimum root mean PRESS	0.6141
Minimizing number of factors	3
Smallest number of factors with p > 0.1	2

Output 95.4.7 More Granular Spline Basis: PLS Variation Summary for Split-Sample Validated Model

Percent Variation Accounted for by Partial Least Squares Factors				
	Model Effects		Dependent Variables	
Number of Extracted Factors	Current	Total	Current	Total
1	1.7967	1.7967	64.7792	64.7792
2	1.3719	3.1687	6.3163	71.0955

Output 95.4.8 More Granular Spline Basis: Smoothed Yield

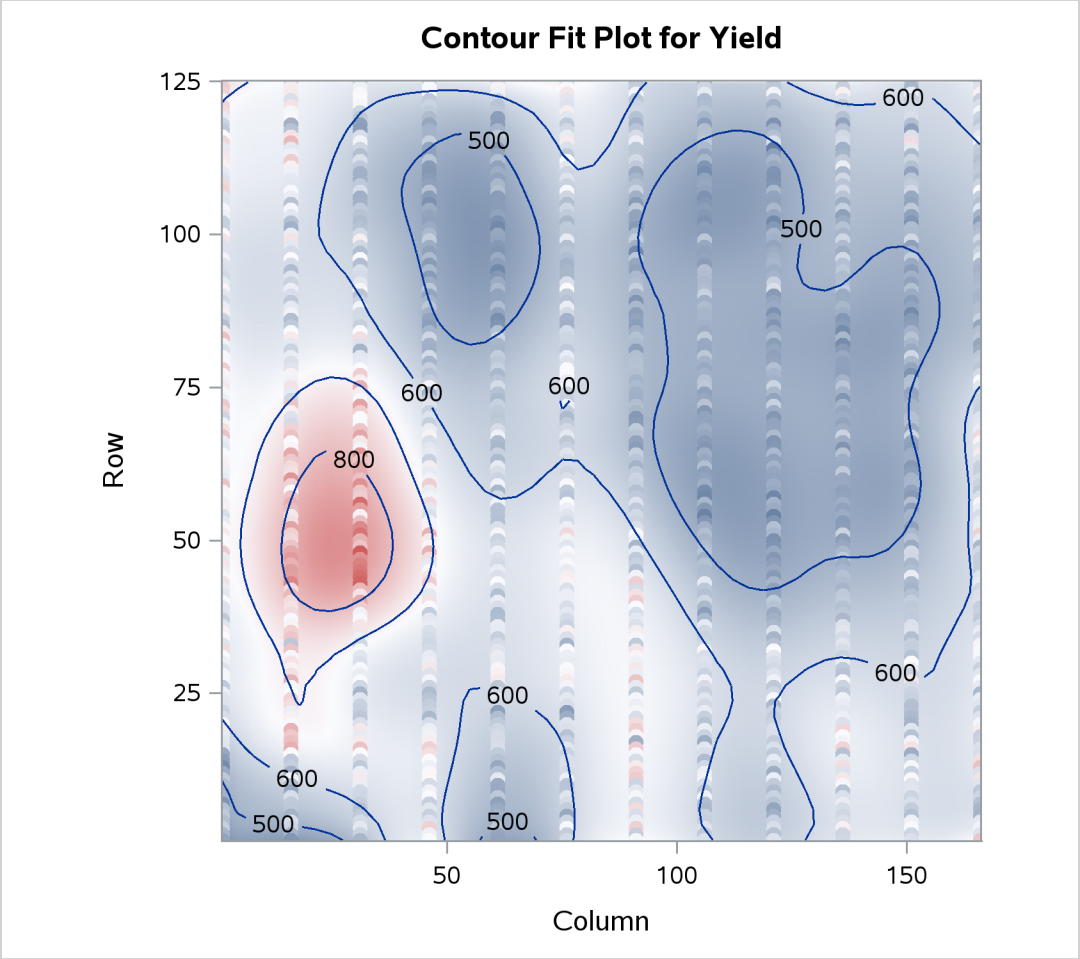

Output 95.4.5 shows that the model now has 360 parameters, many more than before. In Output 95.4.6 you can see that with more granular spline effects, fewer PLS factors are required—only two, in fact. However, Output 95.4.7 shows that this model now accounts for over 70% of the variation in the Yield values, and the contours of predicted values in Output 95.4.8 are less inclined to be simple elliptical shapes.

References

De Jong, S. (1993). "SIMPLS: An Alternative Approach to Partial Least Squares Regression." *Chemometrics and Intelligent Laboratory Systems* 18:251–263.

De Jong, S., and Kiers, H. (1992). "Principal Covariates Regression." *Chemometrics and Intelligent Laboratory Systems* 14:155–164.

Dijkstra, T. K. (1983). "Some Comments on Maximum Likelihood and Partial Least Squares Methods." *Journal of Econometrics* 22:67–90.

Dijkstra, T. K. (1985). *Latent Variables in Linear Stochastic Models: Reflections on Maximum Likelihood and Partial Least Squares Methods*. 2nd ed. Amsterdam: Sociometric Research Foundation.

Frank, I., and Friedman, J. (1993). "A Statistical View of Some Chemometrics Regression Tools." *Technometrics* 35:109–135.

Geladi, P., and Kowalski, B. (1986). "Partial Least-Squares Regression: A Tutorial." *Analytica Chimica Acta* 185:1–17.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. New York: Macmillan.

Helland, I. S. (1988). "On the Structure of Partial Least Squares Regression." *Communications in Statistics—Simulation and Computation* 17:581–607.

Hoerl, A. E., and Kennard, R. W. (1970). "Ridge Regression: Biased Estimation for Non-orthogonal Problems." *Technometrics* 12:55–67.

Lindberg, W., Persson, J.-A., and Wold, S. (1983). "Partial Least-Squares Method for Spectrofluorimetric Analysis of Mixtures of Humic Acid and Ligninsulfonate." *Analytical Chemistry* 55:643–648.

McAvoy, T. J., Wang, N. S., Naidu, S., Bhat, N., Gunter, J., and Simmons, M. (1989). "Interpreting Biosensor Data via Backpropagation." *International Joint Conference on Neural Networks* 1:227–233.

Naes, T., and Martens, H. (1985). "Comparison of Prediction Methods for Multicollinear Data." *Communications in Statistics—Simulation and Computation* 14:545–576.

Ränner, S., Lindgren, F., Geladi, P., and Wold, S. (1994). "A PLS Kernel Algorithm for Data Sets with Many Variables and Fewer Objects." *Journal of Chemometrics* 8:111–125.

Sarle, W. S. (1994). "Neural Networks and Statistical Models." In *Proceedings of the Nineteenth Annual SAS Users Group International Conference*, 1538–1550. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI94/Sugi-94-255%20Sarle.pdf.

Shao, J. (1993). "Linear Model Selection by Cross-Validation." *Journal of the American Statistical Association* 88:486–494.

Tobias, R. D. (1995). "An Introduction to Partial Least Squares Regression." In *Proceedings of the Twentieth Annual SAS Users Group International Conference*, 1250–1257. Cary, NC: SAS Institute Inc. http://support.sas.com/rnd/app/stat/papers/pls.pdf.

Ufkes, J. G. R., Visser, B. J., Heuver, G., and van der Meer, C. (1978). "Structure-Activity Relationships of Bradykinin Potentiating Peptides." *European Journal of Pharmacology* 50:119–122.

Ufkes, J. G. R., Visser, B. J., Heuver, G., Wynne, H. J., and van der Meer, C. (1982). "Further Studies on the Structure-Activity Relationships of Bradykinin-Potentiating Peptides." *European Journal of Pharmacology* 79:155–158.

Umetrics (1995). *Multivariate Analysis.* Three-day course. Winchester, MA: Umetrics.

Van den Wollenberg, A. L. (1977). "Redundancy Analysis: An Alternative to Canonical Correlation Analysis." *Psychometrika* 42:207–219.

Van der Voet, H. (1994). "Comparing the Predictive Accuracy of Models Using a Simple Randomization Test." *Chemometrics and Intelligent Laboratory Systems* 25:313–323.

Weibe, G. A. (1935). "Variation and Correlation in Grain Yield among 1,500 Wheat Nursery Plots." *Journal of Agricultural Research* 50:331–354.

Wold, H. (1966). "Estimation of Principal Components and Related Models by Iterative Least Squares." In *Multivariate Analysis*, edited by P. R. Krishnaiah, 391–420. New York: Academic Press.

Wold, S. (1994). "PLS for Multivariate Linear Modeling." In *QSAR: Chemometric Methods in Molecular Design; Methods and Principles in Medicinal Chemistry*, edited by H. van de Waterbeemd, 195–218. Weinheim, Germany: Verlag-Chemie.

Chapter 96
The POWER Procedure

Contents

Overview: POWER Procedure . . . **7897**
Getting Started: POWER Procedure . . . **7899**
Computing Power for a One-Sample *t* Test . . . 7899
Determining Required Sample Size for a Two-Sample *t* Test . . . 7902
Syntax: POWER Procedure . . . **7906**
PROC POWER Statement . . . 7907
COXREG Statement . . . 7908
CUSTOM Statement . . . 7912
LOGISTIC Statement . . . 7918
MULTREG Statement . . . 7925
ONECORR Statement . . . 7930
ONESAMPLEFREQ Statement . . . 7934
ONESAMPLEMEANS Statement . . . 7942
ONEWAYANOVA Statement . . . 7949
PAIREDFREQ Statement . . . 7954
PAIREDMEANS Statement . . . 7961
PLOT Statement . . . 7970
TWOSAMPLEFREQ Statement . . . 7975
TWOSAMPLEMEANS Statement . . . 7982
TWOSAMPLESURVIVAL Statement . . . 7992
TWOSAMPLEWILCOXON Statement . . . 8004
Details: POWER Procedure . . . **8009**
Overview of Power Concepts . . . 8009
Summary of Analyses . . . 8010
Specifying Value Lists in Analysis Statements . . . 8012
Keyword-Lists . . . 8012
Number-Lists . . . 8012
Grouped-Number-Lists . . . 8012
Name-Lists . . . 8014
Grouped-Name-Lists . . . 8014
Sample Size Adjustment Options . . . 8015
Error and Information Output . . . 8015
Displayed Output . . . 8017
ODS Table Names . . . 8018
Computational Resources . . . 8018
Memory . . . 8018

CPU Time . . . 8018
Computational Methods and Formulas . . . 8019
Common Notation . . . 8019
Analyses in the COXREG Statement . . . 8020
Analyses in the CUSTOM Statement . . . 8021
Analyses in the LOGISTIC Statement . . . 8024
Analyses in the MULTREG Statement . . . 8027
Analyses in the ONECORR Statement . . . 8029
Analyses in the ONESAMPLEFREQ Statement . . . 8031
Analyses in the ONESAMPLEMEANS Statement . . . 8049
Analyses in the ONEWAYANOVA Statement . . . 8053
Analyses in the PAIREDFREQ Statement . . . 8054
Analyses in the PAIREDMEANS Statement . . . 8058
Analyses in the TWOSAMPLEFREQ Statement . . . 8063
Analyses in the TWOSAMPLEMEANS Statement . . . 8068
Analyses in the TWOSAMPLESURVIVAL Statement . . . 8075
Analyses in the TWOSAMPLEWILCOXON Statement . . . 8078
ODS Graphics . . . 8081
Examples: POWER Procedure . . . **8081**
Example 96.1: One-Way ANOVA . . . 8081
Example 96.2: The Sawtooth Power Function in Proportion Analyses . . . 8086
Example 96.3: Simple AB/BA Crossover Designs . . . 8095
Example 96.4: Noninferiority Test with Lognormal Data . . . 8099
Example 96.5: Multiple Regression and Correlation . . . 8103
Example 96.6: Comparing Two Survival Curves . . . 8107
Example 96.7: Confidence Interval Precision . . . 8109
Example 96.8: Customizing Plots . . . 8112
Assigning Analysis Parameters to Axes . . . 8113
Fine-Tuning a Sample Size Axis . . . 8118
Adding Reference Lines . . . 8123
Linking Plot Features to Analysis Parameters . . . 8125
Choosing Key (Legend) Styles . . . 8130
Modifying Symbol Locations . . . 8134
Example 96.9: Binary Logistic Regression with Independent Predictors . . . 8136
Example 96.10: Wilcoxon-Mann-Whitney Test . . . 8138
Example 96.11: Logistic Regression Using the CUSTOM Statement . . . 8141
References . . . **8148**

Overview: POWER Procedure

Power and sample size analysis optimizes the resource usage and design of a study, improving chances of conclusive results with maximum efficiency. The POWER procedure performs prospective power and sample size analyses for a variety of goals, such as the following:

- determining the sample size required to get a significant result with adequate probability (power)
- characterizing the power of a study to detect a meaningful effect
- conducting what-if analyses to assess sensitivity of the power or required sample size to other factors

Here *prospective* indicates that the analysis pertains to planning for a future study. This is in contrast to *retrospective* power analysis for a past study, which is not supported by the procedure.

A variety of statistical analyses are covered:

- t tests, equivalence tests, and confidence intervals for means
- tests, equivalence tests, and confidence intervals for binomial proportions
- multiple regression
- tests of correlation and partial correlation
- one-way analysis of variance
- rank tests for comparing two survival curves
- Cox proportional hazards regression
- logistic regression with binary response
- Wilcoxon-Mann-Whitney (rank-sum) test
- extensions of existing analyses that involve the chi-square, F, t, or normal distribution, or the distribution of the correlation coefficient under multivariate normality

The extensions of existing analyses consist of scalar multipliers and custom values for primary noncentrality and degrees of freedom. Important use cases include the following:

- Wald and likelihood ratio tests in generalized linear models that have nominal, count, or ordinal responses
- sample size inflation that is caused by correlated predictors
- sample size deflation that is caused by correlation between the covariates and the response

Examples of generalized linear models include Poisson regression, logistic regression, and zero-inflated models.

For univariate and multivariate linear models, see Chapter 55, "The GLMPOWER Procedure."

Input for PROC POWER includes the components considered in study planning:

- design
- statistical model and test
- significance level (alpha)
- surmised effects and variability
- power
- sample size

You designate one of these components by a missing value in the input, in order to identify it as the result parameter. The procedure calculates this result value over one or more scenarios of input values for all other components. Power and sample size are the most common result values, but for some analyses the result can be something else. For example, you can solve for the sample size of a single group for a two-sample *t* test.

In addition to tabular results, PROC POWER produces graphs. You can produce the most common types of plots easily with default settings and use a variety of options for more customized graphics. For example, you can control the choice of axis variables, axis ranges, number of plotted points, mapping of graphical features (such as color, line style, symbol and panel) to analysis parameters, and legend appearance.

If ODS Graphics is enabled, then PROC POWER uses ODS Graphics to create graphs; otherwise, traditional graphs are produced.

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

For specific information about the statistical graphics and options available with the POWER procedure, see the PLOT statement and the section "ODS Graphics" on page 8081.

The POWER procedure is one of several tools available in SAS/STAT software for power and sample size analysis. PROC GLMPOWER supports more complex linear models. For more information about the GLMPOWER procedure, see Chapter 55, "The GLMPOWER Procedure." SAS Studio provides a user interface and implements many of the analyses that are supported in the procedures. For more information about the power and sample size steps available in SAS Studio flows, see the steps whose names end in "Statistical Power" and are documented in the "Generating Statistics" section of *SAS Studio: Working with Flows*.

The following sections of this chapter describe how to use PROC POWER and discuss the underlying statistical methodology. The section "Getting Started: POWER Procedure" on page 7899 introduces PROC POWER with simple examples of power computation for a one-sample *t* test and sample size determination for a two-sample *t* test. The section "Syntax: POWER Procedure" on page 7906 describes the syntax of the procedure. The section "Details: POWER Procedure" on page 8009 summarizes the methods employed by PROC POWER and provides details on several special topics. The section "Examples: POWER Procedure" on page 8081 illustrates the use of the POWER procedure with several applications.

For an overview of methodology and SAS tools for power and sample size analysis, see Chapter 19, "Introduction to Power and Sample Size Analysis." For more discussion and examples, see O'Brien and Castelloe (2007); Castelloe (2000); Castelloe and O'Brien (2001); Muller and Benignus (1992); O'Brien and Muller (1993); Lenth (2001).

Getting Started: POWER Procedure

Computing Power for a One-Sample *t* Test

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/pwrgs1.sas.)

Suppose you want to improve the accuracy of a machine used to print logos on sports jerseys. The logo placement has an inherently high variability, but the horizontal alignment of the machine can be adjusted. The operator agrees to pay for a costly adjustment if you can establish a nonzero mean horizontal displacement in either direction with high confidence. You have 150 jerseys at your disposal to measure, and you want to determine your chances of a significant result (power) by using a one-sample t test with a two-sided $\alpha = 0.05$.

You decide that 8 mm is the smallest displacement worth addressing. Hence, you will assume a true mean of 8 in the power computation. Experience indicates that the standard deviation is about 40.

Use the ONESAMPLEMEANS statement in the POWER procedure to compute the power. Indicate power as the result parameter by specifying the POWER= option with a missing value (.). Specify your conjectures for the mean and standard deviation by using the MEAN= and STDDEV= options and for the sample size by using the NTOTAL= option. The statements required to perform this analysis are as follows:

```
proc power;
   onesamplemeans
      mean   = 8
      ntotal = 150
      stddev = 40
      power  = .;
run;
```

Default values for the TEST=, DIST=, ALPHA=, NULLMEAN=, and SIDES= options specify a two-sided t test for a mean of 0, assuming a normal distribution with a significance level of $\alpha = 0.05$.

Figure 96.1 shows the output.

Figure 96.1 Sample Size Analysis for One-Sample *t* Test

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Mean	8
Standard Deviation	40
Total Sample Size	150
Number of Sides	2
Null Mean	0
Alpha	0.05

Computed Power
Power
0.682

The power is about 0.68. In other words, there is about a 2/3 chance that the *t* test will produce a significant result demonstrating the machine's average off-center displacement. This probability depends on the assumptions for the mean and standard deviation.

Now, suppose you want to account for some of your uncertainty in conjecturing the true mean and standard deviation by evaluating the power for four scenarios, using reasonable low and high values, 5 and 10 for the mean, and 30 and 50 for the standard deviation. Also, you might be able to measure more than 150 jerseys, and you would like to know under what circumstances you could get by with fewer. You want to plot power for sample sizes between 100 and 200 to visualize how sensitive the power is to changes in sample size for these four scenarios of means and standard deviations. The following statements perform this analysis:

```
ods graphics on;

proc power;
   onesamplemeans
      mean   = 5 10
      ntotal = 150
      stddev =  30 50
      power  = .;
   plot x=n min=100 max=200;
run;

ods graphics off;
```

The new mean and standard deviation values are specified by using the MEAN= and STDDEV= options in the ONESAMPLEMEANS statement. The PLOT statement with X=N produces a plot with sample size on the X axis. (The result parameter, in this case the power, is always plotted on the other axis.) The MIN= and MAX= options in the PLOT statement determine the sample size range. The ODS GRAPHICS ON statement enables ODS Graphics.

Figure 96.2 shows the output, and Figure 96.3 shows the plot.

Figure 96.2 Sample Size Analysis for One-Sample *t* Test with Input Ranges

The POWER Procedure
One-Sample t Test for Mean

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Total Sample Size	150
Number of Sides	2
Null Mean	0
Alpha	0.05

Computed Power

Index	Mean	Std Dev	Power
1	5	30	0.527
2	5	50	0.229
3	10	30	0.982
4	10	50	0.682

Figure 96.3 Plot of Power versus Sample Size for One-Sample t Test with Input Ranges

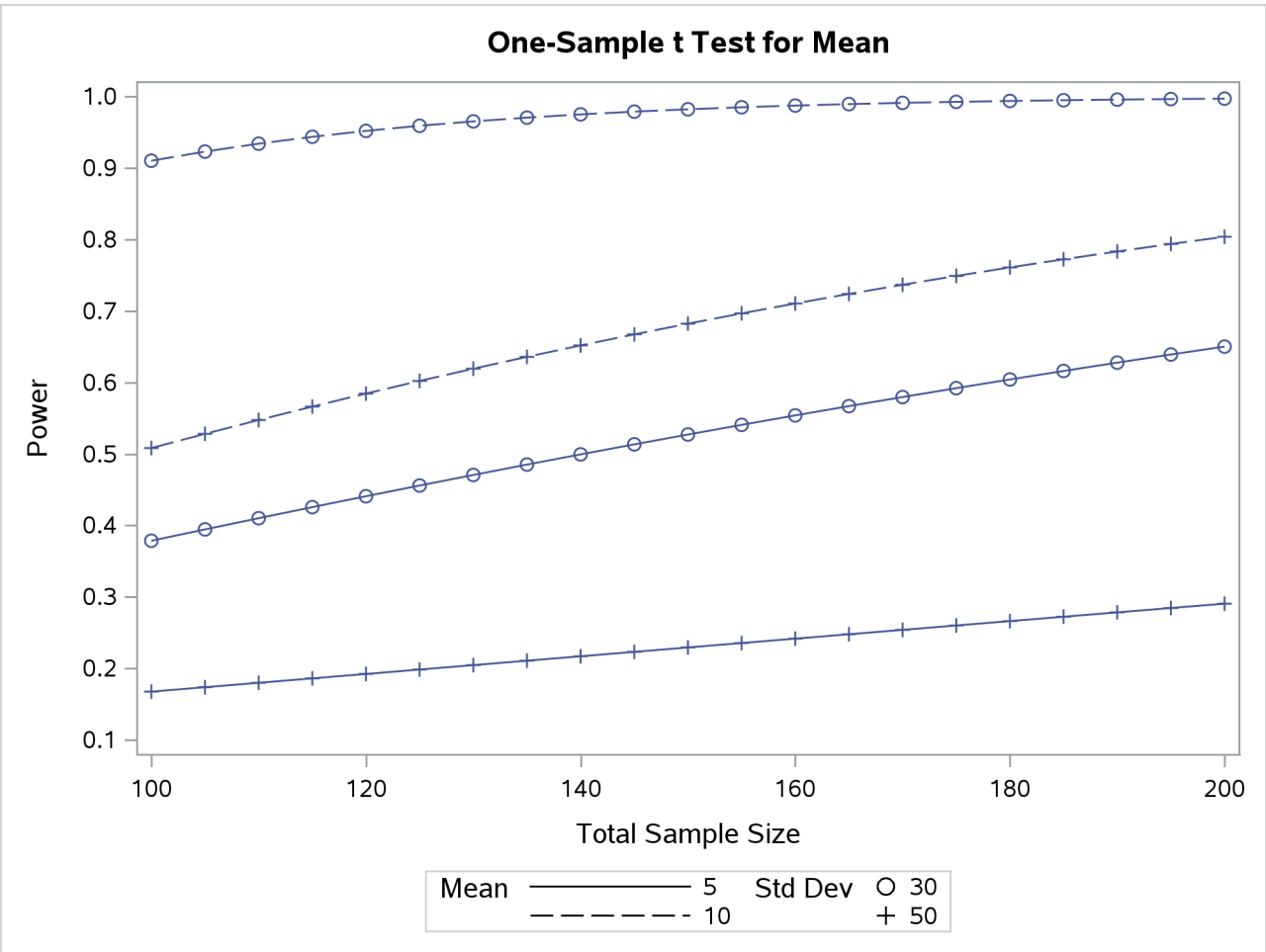

The power ranges from about 0.23 to 0.98 for a sample size of 150 depending on the mean and standard deviation. In Figure 96.3, the line style identifies the mean, and the plotting symbol identifies the standard deviation. The locations of plotting symbols indicate computed powers; the curves are linear interpolations of these points. The plot suggests sufficient power for a mean of 10 and standard deviation of 30 (for any of the sample sizes) but insufficient power for the other three scenarios.

Determining Required Sample Size for a Two-Sample *t* Test

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/pwrgs2.sas`.)

In this example you want to compare two physical therapy treatments designed to increase muscle flexibility. You need to determine the number of patients required to achieve a power of at least 0.9 to detect a group mean difference in a two-sample t test. You will use $\alpha = 0.05$ (two-tailed).

The mean flexibility with the standard treatment (as measured on a scale of 1 to 20) is well known to be about 13 and is thought to be between 14 and 15 with the new treatment. You conjecture three alternative scenarios for the means:

1. $\mu_1 = 13$, $\mu_2 = 14$
2. $\mu_1 = 13$, $\mu_2 = 14.5$
3. $\mu_1 = 13$, $\mu_2 = 15$

You conjecture two scenarios for the common group standard deviation:

1. $\sigma = 1.2$
2. $\sigma = 1.7$

You also want to try three weighting schemes:

1. equal group sizes (balanced, or 1:1)
2. twice as many patients with the new treatment (1:2)
3. three times as many patients with the new treatment (1:3)

This makes $3 \times 2 \times 3 = 18$ scenarios in all.

Use the TWOSAMPLEMEANS statement in the POWER procedure to determine the sample sizes required to give 90% power for each of these 18 scenarios. Indicate total sample size as the result parameter by specifying the NTOTAL= option with a missing value (.). Specify your conjectures for the means by using the GROUPMEANS= option. Using the "matched" notation (discussed in the section "Specifying Value Lists in Analysis Statements" on page 8012), enclose the two group means for each scenario in parentheses. Use the STDDEV= option to specify scenarios for the common standard deviation. Specify the weighting schemes by using the GROUPWEIGHTS= option. You could again use the matched notation. But for illustrative purposes, specify the scenarios for each group weight separately by using the "crossed" notation, with scenarios for each group weight separated by a vertical bar (|). The statements that perform the analysis are as follows:

```
proc power;
   twosamplemeans
      groupmeans   = (13 14) (13 14.5) (13 15)
      stddev       = 1.2 1.7
      groupweights = 1 | 1 2 3
      power        = 0.9
      ntotal       = .;
run;
```

Default values for the TEST=, DIST=, NULLDIFF=, ALPHA=, and SIDES= options specify a two-sided t test of group mean difference equal to 0, assuming a normal distribution with a significance level of $\alpha = 0.05$. The results are shown in Figure 96.4.

Figure 96.4 Sample Size Analysis for Two-Sample t Test Using Group Means

The POWER Procedure
Two-Sample t Test for Mean Difference

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Group 1 Weight	1
Nominal Power	0.9
Number of Sides	2
Null Difference	0
Alpha	0.05

Computed N Total						
Index	**Mean1**	**Mean2**	**Std Dev**	**Weight2**	**Actual Power**	**N Total**
1	13	14.0	1.2	1	0.907	64
2	13	14.0	1.2	2	0.908	72
3	13	14.0	1.2	3	0.905	84
4	13	14.0	1.7	1	0.901	124
5	13	14.0	1.7	2	0.905	141
6	13	14.0	1.7	3	0.900	164
7	13	14.5	1.2	1	0.910	30
8	13	14.5	1.2	2	0.906	33
9	13	14.5	1.2	3	0.916	40
10	13	14.5	1.7	1	0.900	56
11	13	14.5	1.7	2	0.901	63
12	13	14.5	1.7	3	0.908	76
13	13	15.0	1.2	1	0.913	18
14	13	15.0	1.2	2	0.927	21
15	13	15.0	1.2	3	0.922	24
16	13	15.0	1.7	1	0.914	34
17	13	15.0	1.7	2	0.921	39
18	13	15.0	1.7	3	0.910	44

The interpretation is that in the best-case scenario (large mean difference of 2, small standard deviation of

1.2, and balanced design), a sample size of $N = 18$ ($n_1 = n_2 = 9$) patients is sufficient to achieve a power of at least 0.9. In the worst-case scenario (small mean difference of 1, large standard deviation of 1.7, and a 1:3 unbalanced design), a sample size of $N = 164$ ($n_1 = 41$, $n_2 = 123$) patients is necessary. The Nominal Power of 0.9 in the "Fixed Scenario Elements" table represents the input target power, and the Actual Power column in the "Computed N Total" table is the power at the sample size (N Total) adjusted to achieve the specified sample weighting exactly.

Note the following characteristics of the analysis, and ways you can modify them if you want:

- The total sample sizes are rounded up to multiples of the weight sums (2 for the 1:1 design, 3 for the 1:2 design, and 4 for the 1:3 design) to ensure that each group size is an integer. To request raw fractional sample size solutions, use the NFRACTIONAL option.
- Only the group weight that varies (the one for group 2) is displayed as an output column, while the weight for group 1 appears in the "Fixed Scenario Elements" table. To display the group weights together in output columns, use the matched version of the value list rather than the crossed version.
- If you can specify only differences between group means (instead of their individual values), or if you want to display the mean differences instead of the individual means, use the MEANDIFF= option instead of the GROUPMEANS= option.

The following statements implement all of these modifications:

```
proc power;
   twosamplemeans
      nfractional
      meandiff     = 1 to 2 by 0.5
      stddev       = 1.2 1.7
      groupweights = (1 1) (1 2) (1 3)
      power        = 0.9
      ntotal       = .;
run;
```

Figure 96.5 shows the new results.

Figure 96.5 Sample Size Analysis for Two-Sample t Test Using Mean Differences

The POWER Procedure
Two-Sample t Test for Mean Difference

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Nominal Power	0.9
Number of Sides	2
Null Difference	0
Alpha	0.05

Figure 96.5 *continued*

Computed Ceiling N Total							
Index	Mean Diff	Std Dev	Weight1	Weight2	Fractional N Total	Actual Power	Ceiling N Total
1	1.0	1.2	1	1	62.507429	0.902	63
2	1.0	1.2	1	2	70.065711	0.904	71
3	1.0	1.2	1	3	82.665772	0.901	83
4	1.0	1.7	1	1	123.418482	0.901	124
5	1.0	1.7	1	2	138.598159	0.901	139
6	1.0	1.7	1	3	163.899094	0.900	164
7	1.5	1.2	1	1	28.961958	0.900	29
8	1.5	1.2	1	2	32.308867	0.906	33
9	1.5	1.2	1	3	37.893351	0.901	38
10	1.5	1.7	1	1	55.977156	0.900	56
11	1.5	1.7	1	2	62.717357	0.901	63
12	1.5	1.7	1	3	73.954291	0.900	74
13	2.0	1.2	1	1	17.298518	0.913	18
14	2.0	1.2	1	2	19.163836	0.913	20
15	2.0	1.2	1	3	22.282926	0.910	23
16	2.0	1.7	1	1	32.413512	0.905	33
17	2.0	1.7	1	2	36.195531	0.907	37
18	2.0	1.7	1	3	42.504535	0.903	43

Note that the Nominal Power of 0.9 applies to the raw computed sample size (Fractional N Total), and the Actual Power column applies to the rounded sample size (Ceiling N Total). Some of the adjusted sample sizes in Figure 96.5 are lower than those in Figure 96.4 because underlying group sample sizes are allowed to be fractional (for example, the first Ceiling N Total of 63 corresponding to equal group sizes of 31.5).

Syntax: POWER Procedure

The following statements are available in the POWER procedure:

PROC POWER < *options* > ;
  COXREG < *options* > ;
  CUSTOM < *options* > ;
  LOGISTIC < *options* > ;
  MULTREG < *options* > ;
  ONECORR < *options* > ;
  ONESAMPLEFREQ < *options* > ;
  ONESAMPLEMEANS < *options* > ;
  ONEWAYANOVA < *options* > ;
  PAIREDFREQ < *options* > ;
  PAIREDMEANS < *options* > ;
  PLOT < *plot-options* > < / *graph-options* > ;
  TWOSAMPLEFREQ < *options* > ;
  TWOSAMPLEMEANS < *options* > ;
  TWOSAMPLESURVIVAL < *options* > ;
  TWOSAMPLEWILCOXON < *options* > ;

The statements in the POWER procedure consist of the PROC POWER statement, a set of *analysis statements* (for requesting specific power and sample size analyses), and the PLOT statement (for producing graphs). The PROC POWER statement and at least one of the analysis statements are required. The analysis statements are COXREG, CUSTOM, LOGISTIC, MULTREG, ONECORR, ONESAMPLEFREQ, ONESAMPLEMEANS, ONEWAYANOVA, PAIREDFREQ, PAIREDMEANS, TWOSAMPLEFREQ, TWOSAMPLEMEANS, TWOSAMPLESURVIVAL, and TWOSAMPLEWILCOXON.

You can use multiple analysis statements and multiple PLOT statements. Each analysis statement produces a separate sample size analysis. Each PLOT statement refers to the previous analysis statement and generates a separate graph (or set of graphs).

The name of an analysis statement describes the framework of the statistical analysis for which sample size calculations are desired. You use options in the analysis statements to identify the result parameter to compute, to specify the statistical test and computational options, and to provide one or more scenarios for the values of relevant analysis parameters.

Table 96.1 summarizes the basic functions of each statement in PROC POWER. The syntax of each statement in Table 96.1 is described in the following pages.

Table 96.1 Statements in the POWER Procedure

Statement	Description
PROC POWER	Invokes the procedure
COXREG	Cox proportional hazards regression
CUSTOM	extensions of existing analyses that involve the chi-square, F, t, or normal distribution, or the distribution of the correlation coefficient under multivariate normality

Table 96.1 *continued*

Statement	Description
LOGISTIC	Likelihood ratio chi-square test of a single predictor in logistic regression with binary response
MULTREG	Tests of one or more coefficients in multiple linear regression
ONECORR	Fisher's z test and t test of (partial) correlation
ONESAMPLEFREQ	Tests, confidence interval precision, and equivalence tests of a single binomial proportion
ONESAMPLEMEANS	One-sample t test, confidence interval precision, or equivalence test
ONEWAYANOVA	One-way ANOVA including single-degree-of-freedom contrasts
PAIREDFREQ	McNemar's test for paired proportions
PAIREDMEANS	Paired t test, confidence interval precision, or equivalence test
PLOT	Displays plots for previous sample size analysis
TWOSAMPLEFREQ	Chi-square, likelihood ratio, and Fisher's exact tests for two independent proportions
TWOSAMPLEMEANS	Two-sample t test, confidence interval precision, or equivalence test
TWOSAMPLESURVIVAL	Log-rank, Gehan, and Tarone-Ware tests for comparing two survival curves
TWOSAMPLEWILCOXON	Wilcoxon-Mann-Whitney (rank-sum) test for 2 independent groups

See the section "Summary of Analyses" on page 8010 for a summary of the analyses available and the syntax required for them.

PROC POWER Statement

PROC POWER < *options* > **;**

The PROC POWER statement invokes the POWER procedure. You can specify the following option.

PLOTONLY

specifies that only graphical results from the PLOT statement should be produced.

COXREG Statement

COXREG < *options* > **;**

The COXREG statement performs power and sample size analyses for the score test of a single scalar predictor in Cox proportional hazards regression for survival data, possibly in the presence of one or more covariates that might be correlated with the tested predictor.

Summary of Options

Table 96.2 summarizes the *options* available in the COXREG statement.

Table 96.2 COXREG Statement Options

Option	Description
Define Analysis	
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
SIDES=	Specifies the number of sides and the direction of the statistical test
Specify Effects	
RSQUARE=	Specifies the R^2 value from the regression of the predictor of interest on the remaining predictors
HAZARDRATIO=	Specifies the hazard ratio
Specify Variability	
STDDEV=	Specifies the standard deviation of the predictor variable being tested
Specify Sample Size	
EVENTPROB=	Specifies the probability that an uncensored event occurs
EVENTSTOTAL=	Specifies the expected total number of events
NFRACTIONAL	Enables fractional input and output for sample sizes
NTOTAL=	Specifies the sample size
Specify Power	
POWER=	Specifies the desired power of the test
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.3 summarizes the valid result parameters for different analyses in the COXREG statement.

Table 96.3 Summary of Result Parameters in the COXREG Statement

Analyses	Solve For	Syntax
TEST=SCORE	Power Sample size	POWER=. NTOTAL=. EVENTSTOTAL=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

EVENTPROB=*number-list*

specifies the probability that an uncensored event occurs. You must specify this option when you use the NTOTAL= option. The EVENTPROB= option produces an error if you also specify the EVENTSTOTAL= option. If you are computing power, the input sample size is multiplied by the event probability to determine the number of events. If you are computing sample size, the internally computed number of events is divided by the event probability to compute the sample size. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

EVENTSTOTAL=*number-list*

EVENTTOTAL=*number-list*

EETOTAL=*number-list*

specifies the expected number of uncensored events, or requests a solution for this parameter by specifying a missing value (EVENTSTOTAL=.). The NFRACTIONAL option is automatically enabled when you use the EVENTSTOTAL= option. You must use either the EVENTSTOTAL= option or the NTOTAL= option, and you cannot use both. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

HAZARDRATIO=*number-list*

HR=*number-list*

specifies the hazard ratio for a one-unit increase in the predictor of interest x_1, holding any other predictors constant. The hazard ratio is equal to $\exp(\beta_1)$, where β_1 is the regression coefficient of x_1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. This option is automatically enabled when you use the EVENTSTOTAL= option. For information about the ramifications of the presence (and absence) of the NFRACTIONAL option, see the section "Sample Size Adjustment Options" on page 8015.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). You must use either the NTOTAL= option or the EVENTSTOTAL= option, and you cannot use both. The number of events is used internally in calculations, and the sample size is the ratio of the number of events (EVENTSTOTAL) and the event probability EVENTPROB. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=
- ALPHA=
- EVENTPROB=
- RSQUARE=
- HAZARDRATIO=
- STDDEV=
- NTOTAL=
- EVENTSTOTAL=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the COXREG statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the COXREG statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

RSQUARE=*number-list*

specifies the R^2 value from the regression of the predictor of interest on the remaining predictors. The sample size is either multiplied (if you are computing power) or divided (if you are computing sample size) by a factor of $(1 - R^2)$. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test or confidence interval. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. You can specify the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating a positive correlation between the tested predictor and survival—that is, a hazard ratio less than 1.

L specifies a lower one-sided test, with the alternative hypothesis indicating a negative correlation between the tested predictor and survival—that is, a hazard ratio greater than 1.

By default, SIDES=2.

STDDEV=*number-list*

STD=*number-list*

specifies the standard deviation of the predictor of interest. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TEST=SCORE

specifies the score test in Cox proportional hazards regression. This is the default test option.

Restrictions on Option Combinations

To specify the sample size, choose one of the following parameterizations:

- sample size (by using the NTOTAL= option) and event probability (by using the EVENTPROB= option)
- number of events (by using the EVENTSTOTAL= option)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the TWOSAMPLESURVIVAL statement.

Score Test for Cox Proportional Hazards Regression

You can use the NTOTAL= and EVENTPROB= options, as in the following statements. Default values for the SIDES=, ALPHA=, and TEST= options specify a two-sided score test with a significance level of 0.05.

```
proc power;
   coxreg
      hazardratio = 1.4
      rsquare = 0.15
      stddev = 1.2
      ntotal = 80
      eventprob = 0.8
      power = .
   ;
run;
```

You can also use the EVENTSTOTAL= option, as in the following statements:

```
proc power;
   coxreg
      hazardratio = 1.6
      rsquare = 0.2
      stddev = 1.1
      power = 0.9
      eventstotal = .
```

```
    ;
run;
```

CUSTOM Statement

CUSTOM < *options* > **;**

The CUSTOM statement performs power and sample size analyses for extensions of existing analyses that involve the chi-square, F, t, or normal distribution, or the distribution of the correlation coefficient under multivariate normality. The extensions consist of scalar multipliers and custom values for primary noncentrality and degrees of freedom. Important use cases include the following:

- Wald and likelihood ratio tests in generalized linear models that have nominal, count, or ordinal responses
- sample size inflation that is caused by correlated predictors
- sample size deflation that is caused by correlation between the covariates and the response

Examples of generalized linear models include Poisson regression, logistic regression, and zero-inflated models. See Example 96.11 for an example that involves logistic regression.

Summary of Options

Table 96.4 summarizes the *options* available in the CUSTOM statement.

Table 96.4 CUSTOM Statement Options

Option	Description
Define Analysis	
DIST=	Specifies the underlying distributional assumption
Specify Analysis Information	
ALPHA=	Specifies the significance level
MODELDF=	Specifies the model degrees of freedom
SIDES=	Specifies the number of sides and the direction of the statistical test
TESTDF=	Specifies the test degrees of freedom
Specify Effects	
CORR=	Specifies the correlation
CRITMULT=	Specifies the scalar multiplier of the critical value
PNCMULT=	Specifies the scalar multiplier of the primary noncentrality
PRIMNC=	Specifies the primary noncentrality
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NTOTAL=	Specifies the total sample size
Specify Power	
POWER=	Specifies the desired power

Table 96.4 *continued*

Option	Description
Control Ordering in Output	
OUTPUTORDER=	Controls the order of parameters

Table 96.5 shows which *options* apply to each distribution. Options not shown in the table apply to all distributions.

Table 96.5 Options That Can Be Used for Each Distribution in the CUSTOM Statement

	DIST= Value				
Option	CHISQUARE	CORR	F	NORMAL	T
CORR		•			
CRITMULT	•			•	
MODELDF		•	•		•
PNCMULT	•		•	•	•
PRIMNC	•		•	•	•
SIDES		•		•	•
TESTDF	•	•	•		

Table 96.6 summarizes the valid result parameters for different analyses in the CUSTOM statement.

Table 96.6 Summary of Result Parameters in the CUSTOM Statement

Solve For	Syntax
Power	POWER=.
Sample size	NTOTAL=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. By default, ALPHA=0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CORR=*number-list*

specifies the partial correlation between $p_1 \geq 1$ variables in a set X_1 and a variable Y, adjusting for $p - p_1$ variables in a (possibly empty) set X_{-1}. This option is required for DIST=CORR and not allowed for any other value of the DIST= option. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CRITMULT=*number-list*

specifies a scalar multiplier of the critical value in the power computation formula, where values in *number-list* must be nonnegative. This option is allowed only for DIST=CHISQUARE and DIST=NORMAL. Values must be nonnegative. By default, CRITMULT=1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DIST=CHISQUARE | CORR | F | NORMAL | T

specifies the underlying distribution assumed for the test statistic. You must specify one of the following values:

CHISQUARE specifies the chi-square distribution.

CORR specifies the distribution of the Pearson correlation coefficient under multivariate normality.

F specifies the F distribution.

NORMAL specifies the normal distribution.

T specifies the t distribution.

MODELDF=*number-list*

specifies the model degrees of freedom. This option is required for DIST=CORR, DIST=F, and DIST=T, and it is not allowed for any other value of the DIST= option. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. For information about the ramifications of the presence (and absence) of the NFRACTIONAL option, see the section "Sample Size Adjustment Options" on page 8015.

NTOTAL=*number-list*

specifies the sample size, or requests a solution for the sample size by specifying a missing value (NTOTAL=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. You can specify the following values:

INTERNAL arranges the parameters in the output according to the following order of their corresponding *options*:

1. SIDES=
2. TESTDF=
3. MODELDF=
4. PNCMULT=
5. CRITMULT=
6. ALPHA=
7. PRIMNC=
8. CORR=

9. NTOTAL=
10. POWER=

SYNTAX arranges the parameters in the output in the same order in which their corresponding *options* are specified in the CUSTOM statement.

REVERSE arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the CUSTOM statement.

By default, OUTPUTORDER=INTERNAL.

PNCMULT=*number-list*
specifies a scalar multiplier of the primary noncentrality in the power computation formula, where values in *number-list* must be nonnegative This option is allowed only for DIST=CHISQUARE, DIST=F, DIST=NORMAL, and DIST=T. By default, PNCMULT=1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

POWER=*number-list*
specifies the desired power of the test, or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability (a number between 0 and 1) rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PRIMNC=*number-list*
specifies the primary noncentrality of the distribution of the test statistic. This option is allowed only for DIST=CHISQUARE, DIST=F, DIST=NORMAL, and DIST=T. Values in *number-list* must be nonnegative for DIST=CHISQUARE and DIST=F. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*
specifies the number of sides (tails) and the direction of the statistical test. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. You can specify one or more of the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating an effect greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating an effect less than the null value.

By default, SIDES=2.

TESTDF=*number-list*
specifies the test degrees of freedom. This option is allowed only for DIST=CHISQUARE, DIST=CORR, and DIST=F. TESTDF must be equal to 1 if DIST=CORR; SIDES=1, U, or L; and the TESTDF= option is specified. By default, TESTDF=1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the CUSTOM statement.

Chi-Square Distribution

If you assume that the test statistic has a chi-square distribution, then you must specify the PRIMNC= option (primary noncentrality) and TESTDF= option (test degrees of freedom), as illustrated in the following statements:

```
proc power;
   custom
      dist   = chisquare
      primnc = 0.23
      testdf = 1
      ntotal = 30
      power  = .;
run;
```

The POWER=. option requests a solution for the power at a sample size of 30, as specified in the NTOTAL= option. You can also specify the CRITMULT= option to multiply the critical value in the power formula by a scalar or the PNCMULT= option to multiply the primary noncentrality by a scalar.

F Distribution

If you assume that the test statistic has an *F* distribution, then you must specify the PRIMNC= option (primary noncentrality), MODELDF= option (model degrees of freedom), and TESTDF= option (test degrees of freedom), as illustrated in the following statements:

```
proc power;
   custom
      dist   = f
      primnc = 0.04
      modeldf = 5
      testdf = 2
      ntotal = .
      power  = 0.9;
run;
```

The NTOTAL=. option requests a solution for the sample size that is required to achieve a power of at least 0.9, as specified in the POWER= option. You can also specify the PNCMULT= option to multiply the primary noncentrality by a scalar.

t Distribution

If you assume that the test statistic has a *t* distribution, then you must specify the PRIMNC= option (primary noncentrality) and MODELDF= option (model degrees of freedom), as illustrated in the following statements:

```
proc power;
   custom
      dist   = t
      primnc = 0.2
      modeldf = 2
      ntotal = 200
      power  = .;
```

```
run;
```

The POWER=. option requests a solution for the power at a sample size of 200, as specified in the NTOTAL= option. The default of SIDES=2 specifies a two-sided test. You can also specify the PNCMULT= option to multiply the primary noncentrality by a scalar.

Normal Distribution

If you assume that the test statistic has a normal distribution, then you must specify the PRIMNC= option (primary noncentrality), as illustrated in the following statements:

```
proc power;
   custom
      dist   = normal
      primnc = 0.2
      ntotal = .
      power  = 0.9;
run;
```

The NTOTAL=. option requests a solution for the sample size that is required to achieve a power of at least 0.9, as specified in the POWER= option. The default of SIDES=2 specifies a two-sided test. You can also specify the CRITMULT= option to multiply the critical value in the power formula by a scalar or the PNCMULT= option to multiply the primary noncentrality by a scalar.

Distribution of the Correlation Coefficient under Multivariate Normality

If you plan to use the Pearson correlation coefficient in your data analysis and you assume that the data have a multivariate normal distribution, then you must specify the CORR= option (partial correlation), MODELDF= option (model degrees of freedom), and TESTDF= option (test degrees of freedom), as illustrated in the following statements:

```
proc power;
   custom
      dist    = corr
      corr    = 0.3
      modeldf = 3
      testdf = 1
      ntotal  = 130
      power   = .;
run;
```

The POWER=. option requests a solution for the power at a sample size of 130, as specified in the NTOTAL= option. The default of SIDES=2 specifies a two-sided test.

LOGISTIC Statement

LOGISTIC < *options* > **;**

The LOGISTIC statement performs power and sample size analyses for the likelihood ratio chi-square test of a single predictor in binary logistic regression, possibly in the presence of one or more covariates that might be correlated with the tested predictor.

Summary of Options

Table 96.7 summarizes the *options* available in the LOGISTIC statement.

Table 96.7 LOGISTIC Statement Options

Option	Description
Define Analysis	
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
COVARIATES=	Specifies the distributions of predictor variables
TESTPREDICTOR=	Specifies the distribution of the predictor variable being tested
VARDIST=	Defines a distribution for a predictor variable
Specify Effects	
CORR=	Specifies the multiple correlation between the predictor and the covariates
COVODDSRATIOS=	Specifies the odds ratios for the covariates
COVREGCOEFFS=	Specifies the regression coefficients for the covariates
DEFAULTUNIT=	Specifies the default change in the predictor variables
INTERCEPT=	Specifies the intercept
RESPONSEPROB=	Specifies the response probability
TESTODDSRATIO=	Specifies the odds ratio being tested
TESTREGCOEFF=	Specifies the regression coefficient for the predictor variable
UNITS=	Specifies the changes in the predictor variables
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NTOTAL=	Specifies the sample size
Specify Power	
POWER=	Specifies the desired power of the test
Specify Computational Method	
DEFAULTNBINS=	Specifies the default number of categories for each predictor variable
NBINS=	Specifies the number of categories for predictor variables
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.8 summarizes the valid result parameters in the LOGISTIC statement.

Table 96.8 Summary of Result Parameters in the LOGISTIC Statement

Analyses	Solve For	Syntax
TEST=LRCHI	Power	POWER=.
	Sample size	NTOTAL=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CORR=*number-list*

specifies the multiple correlation (ρ) between the tested predictor and the covariates. If you also specify the COVARIATES= option, then the sample size is either multiplied (if you are computing power) or divided (if you are computing sample size) by a factor of $(1 - \rho^2)$. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

COVARIATES=*grouped-name-list*

specifies the distributions of any predictor variables in the model but not being tested, using labels specified with the VARDIST= option. The distributions are assumed to be independent of each other and of the tested predictor. If this option is omitted, then the tested predictor specified by the TESTEDPREDICTOR= option is assumed to be the only predictor in the model. For information about specifying the *grouped-name-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

COVODDSRATIOS=*grouped-number-list*

specifies the odds ratios for the covariates in the full model (including variables in the TESTPREDICTOR= and COVARIATES= options). The ordering of the values corresponds to the ordering in the COVARIATES= option. If the response variable is coded as $Y = 1$ for success and $Y = 0$ for failure, then the odds ratio for each covariate X is the odds of $Y = 1$ when $X = a$ divided by the odds of $Y = 1$ when $X = b$, where a and b are determined from the DEFAULTUNIT= and UNITS= options. Values must be greater than zero. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

COVREGCOEFFS=*grouped-number-list*

specifies the regression coefficients for the covariates in the full model including the test predictor (as specified by the TESTPREDICTOR= option). The ordering of the values corresponds to the ordering in the COVARIATES= option. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DEFAULTNBINS=*number*

specifies the default number of categories (or "bins") into which the distribution for each predictor variable is divided in internal calculations. Higher values increase computational time and memory requirements but generally lead to more accurate results. However, if the value is too high, then numerical instability can occur. Lower values are less likely to produce "No solution computed" errors. Each test predictor or covariate that is absent from the NBINS= option derives its bin number from the DEFAULTNBINS= option. The default value is DEFAULTNBINS=10.

There are two variable distributions for which the number of bins can be overridden internally:

- For an ordinal distribution, the number of ordinal values is always used as the number of bins.
- For a binomial distribution, if the requested number of bins is larger than $n + 1$, where n is the sample size parameter of the binomial distribution, then exactly $n + 1$ bins are used.

DEFAULTUNIT=*change-spec*

specifies the default change in the predictor variables assumed for odds ratios specified with the COVODDSRATIOS= and TESTODDSRATIO= options. Each test predictor or covariate that is absent from the UNITS= option derives its change value from the DEFAULTUNIT= option. The value must be nonzero. The default value is DEFAULTUNIT=1. This option can be used only if at least one of the COVODDSRATIOS= and TESTODDSRATIO= options is used.

Valid specifications for *change-spec* are as follows:

number defines the odds ratio as the ratio of the response variable odds when $X = a$ to the odds when $X = a - \mathit{number}$ for any constant a.

<+ | ->**SD** defines the odds ratio as the ratio of the odds when $X = a$ to the odds when $X = a - \sigma$ (or $X = a + \sigma$, if SD is preceded by a minus sign (–)) for any constant a, where σ is the standard deviation of X (as determined from the VARDIST= option).

*multiple****SD** defines the odds ratio as the ratio of the odds when $X = a$ to the odds when $X = a - \mathit{multiple} * \sigma$ for any constant a, where σ is the standard deviation of X (as determined from the VARDIST= option).

PERCENTILES(*p1*, *p2*) defines the odds ratio as the ratio of the odds when X is equal to its $p2\times$ 100th percentile to the odds when X is equal to its $p1\times$ 100th percentile (where the percentiles are determined from the distribution specified in the VARDIST= option). Values for *p1* and *p2* must be strictly between 0 and 1.

INTERCEPT=*number-list*

specifies the intercept in the full model (including variables in the TESTPREDICTOR= and COVARIATES= options). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NBINS=(*"name"* **=** *number* **<. . .** *"name"* **=** *number* **>)**

specifies the number of categories (or "bins") into which the distribution for each predictor variable (identified by its *name* from the VARDIST= option) is divided in internal calculations. Higher values increase computational time and memory requirements but generally lead to more accurate results. However, if the value is too high, then numerical instability can occur. Lower values are less likely to produce "No solution computed" errors. Each predictor variable that is absent from the NBINS= option derives its bin number from the DEFAULTNBINS= option.

There are two variable distributions for which the NBINS= value can be overridden internally:

- For an ordinal distribution, the number of ordinal values is always used as the number of bins.
- For a binomial distribution, if the requested number of bins is larger than $n + 1$, where n is the sample size parameter of the binomial distribution, then exactly $n + 1$ bins are used.

NFRACTIONAL
NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). Values must be at least one. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- DEFAULTNBINS=
- NBINS=
- ALPHA=
- RESPONSEPROB=
- INTERCEPT=
- TESTPREDICTOR=
- TESTODDSRATIO=
- TESTREGCOEFF=
- COVARIATES=
- COVODDSRATIOS=
- COVREGCOEFFS=
- CORR=
- NTOTAL=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the LOGISTIC statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the LOGISTIC statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

RESPONSEPROB=*number-list*

specifies the response probability in the full model when all predictor variables (including variables in the TESTPREDICTOR= and COVARIATES= options) are equal to their means. The log odds of this probability are equal to the intercept in the full model where all predictor are centered at their means. If the response variable is coded as $Y = 1$ for success and $Y = 0$ for failure, then this probability is equal to the mean of Y in the full model when all Xs are equal to their means. Values must be strictly between zero and one. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TEST=LRCHI

specifies the likelihood ratio chi-square test of a single model parameter in binary logistic regression. This is the default test option.

TESTODDSRATIO=*number-list*

specifies the odds ratio for the predictor variable being tested in the full model (including variables in the TESTPREDICTOR= and COVARIATES= options). If the response variable is coded as $Y = 1$ for success and $Y = 0$ for failure, then the odds ratio for the X being tested is the odds of $Y = 1$ when $X = a$ divided by the odds of $Y = 1$ when $X = b$, where a and b are determined from the DEFAULTUNIT= and UNITS= options. Values must be greater than zero. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TESTPREDICTOR=*name-list*

specifies the distribution of the predictor variable being tested, using labels specified with the VARDIST= option. This distribution is assumed to be independent of the distributions of the covariates as defined in the COVARIATES= option. For information about specifying the *name-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TESTREGCOEFF=*number-list*

specifies the regression coefficient for the predictor variable being tested in the full model including the covariates specified by the COVARIATES= option. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

UNITS=(*"name"* **=** *change-spec* **<...** *"name"* **=** *change-spec* **>)**

specifies the changes in the predictor variables assumed for odds ratios specified with the COVODDSRATIOS= and TESTODDSRATIO= options. Each predictor variable whose *name* (from the VARDIST= option) is absent from the UNITS option derives its change value from the DEFAULTUNIT= option. This option can be used only if at least one of the COVODDSRATIOS= and TESTODDSRATIO= options is used.

Valid specifications for *change-spec* are as follows:

number defines the odds ratio as the ratio of the response variable odds when $X = a$ to the odds when $X = a - \mathit{number}$ for any constant a.

<+ | ->**SD** defines the odds ratio as the ratio of the odds when $X = a$ to the odds when $X = a - \sigma$ (or $X = a + \sigma$, if SD is preceded by a minus sign (–)) for any constant a, where σ is the standard deviation of X (as determined from the VARDIST= option).

*multiple****SD** defines the odds ratio as the ratio of the odds when $X = a$ to the odds when $X = a - \mathit{multiple} \times \sigma$ for any constant a, where σ is the standard deviation of X (as determined from the VARDIST= option).

PERCENTILES(*p1*, *p2*) defines the odds ratio as the ratio of the odds when X is equal to its $p2\times$ 100th percentile to the odds when X is equal to its $p1\times$ 100th percentile (where the percentiles are determined from the distribution specified in the VARDIST= option). Values for *p1* and *p2* must be strictly between 0 and 1.

Each unit value must be nonzero.

VARDIST("*label***")=***distribution* **(***parameters***)**
defines a distribution for a predictor variable.

For the VARDIST= option,

label identifies the variable distribution in the output and with the COVARIATES= and TESTPREDICTOR= options.

distribution specifies the distributional form of the variable.

parameters specifies one or more parameters associated with the distribution.

The *distributions* and *parameters* are named and defined in the same way as the distributions and arguments in the CDF SAS function; for more information, see *SAS Functions and CALL Routines: Reference*. Choices for distributional forms and their parameters are as follows:

ORDINAL ((*values*) : (*probabilities*)) is an ordered categorical distribution. The *values* are any numbers separated by spaces. The *probabilities* are numbers between 0 and 1 (inclusive) separated by spaces. Their sum must be exactly 1. The number of *probabilities* must match the number of *values*.

BETA (*a*, *b* <, *l*, *r* >) is a beta distribution with shape parameters *a* and *b* and optional location parameters *l* and *r*. The values of *a* and *b* must be greater than 0, and *l* must be less than *r*. The default values for *l* and *r* are 0 and 1, respectively.

BINOMIAL (*p*, *n*) is a binomial distribution with probability of success *p* and number of independent Bernoulli trials *n*. The value of *p* must be greater than 0 and less than 1, and *n* must be an integer greater than 0. If $n = 1$, then the distribution is binary.

EXPONENTIAL (λ) is an exponential distribution with scale λ, which must be greater than 0.

GAMMA (a, λ) is a gamma distribution with shape a and scale λ. The values of a and λ must be greater than 0.

LAPLACE (θ, λ) is a Laplace distribution with location θ and scale λ. The value of λ must be greater than 0.

LOGISTIC (θ, λ) is a logistic distribution with location θ and scale λ. The value of λ must be greater than 0.

LOGNORMAL (θ, λ) is a lognormal distribution with location θ and scale λ. The value of λ must be greater than 0.

NORMAL (θ, λ) is a normal distribution with mean θ and standard deviation λ. The value of λ must be greater than 0.

POISSON (m) is a Poisson distribution with mean m. The value of m must be greater than 0.

UNIFORM (l, r) is a uniform distribution on the interval [l, r], where $l < r$.

Restrictions on Option Combinations

To specify the intercept in the full model, choose one of the following two parameterizations:

- intercept (using the INTERCEPT= options)
- Prob(Y = 1) when all predictors are equal to their means (using the RESPONSEPROB= option)

To specify the effect associated with the predictor variable being tested, choose one of the following two parameterizations:

- odds ratio (using the TESTODDSRATIO= options)
- regression coefficient (using the TESTREGCOEFFS= option)

To describe the effects of the covariates in the full model, choose one of the following two parameterizations:

- odds ratios (using the COVODDSRATIOS= options)
- regression coefficients (using the COVREGCOEFFS= options)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the LOGISTIC statement.

Likelihood Ratio Chi-Square Test for One Predictor

You can express effects in terms of response probability and odds ratios, as in the following statements:

```
proc power;
   logistic
      vardist("x1a") = normal(0, 2)
      vardist("x1b") = normal(0, 3)
      vardist("x2") = poisson(7)
      vardist("x3a") = ordinal((-5 0 5) : (.3 .4 .3))
      vardist("x3b") = ordinal((-5 0 5) : (.4 .3 .3))
      testpredictor = "x1a" "x1b"
      covariates = "x2" | "x3a" "x3b"
      responseprob = 0.15
      testoddsratio = 1.75
      covoddsratios = (2.1 1.4)
      ntotal = 100
      power = .;
run;
```

The VARDIST= options define the distributions of the predictor variables. The TESTPREDICTOR= option specifies two scenarios for the test predictor distribution, Normal(10,2) and Normal(10,3). The COVARIATES= option specifies two covariates, the first with a Poisson(7) distribution. The second covariate has an ordinal distribution on the values –5, 0, and 5 with two scenarios for the associated probabilities: (.3, .4, .3) and (.4, .3, .3). The response probability in the full model with all variables equal to their means is specified by the RESPONSEPROB= option as 0.15. The odds ratio for a unit decrease in the tested predictor is specified by the TESTODDSRATIO= option to be 1.75. Corresponding odds ratios for the two covariates

in the full model are specified by the COVODDSRATIOS= option to be 2.1 and 1.4. The POWER=. option requests a solution for the power at a sample size of 100 as specified by the NTOTAL= option.

Default values of the TEST= and ALPHA= options specify a likelihood ratio test of the first predictor with a significance level of 0.05. The default of DEFAULTUNIT=1 specifies that all odds ratios are defined in terms of unit changes in predictors. The default of DEFAULTNBINS=10 specifies that each of the three predictor variables is discretized into a distribution with 10 categories in internal calculations.

You can also express effects in terms of regression coefficients, as in the following statements:

```
proc power;
   logistic
      vardist("x1a") = normal(0, 2)
      vardist("x1b") = normal(0, 3)
      vardist("x2") = poisson(7)
      vardist("x3a") = ordinal((-5 0 5) : (.3 .4 .3))
      vardist("x3b") = ordinal((-5 0 5) : (.4 .3 .3))
      testpredictor = "x1a" "x1b"
      covariates = "x2" | "x3a" "x3b"
      intercept = -6.928162
      testregcoeff = 0.5596158
      covregcoeffs = (0.7419373 0.3364722)
      ntotal = 100
      power = .;
run;
```

The regression coefficients for the tested predictor (TESTREGCOEFF=0.5596158) and covariates (COVREGCOEFFS=(0.7419373 0.3364722)) are determined by taking the logarithm of the corresponding odds ratios. The intercept in the full model is specified as –6.928162 by the INTERCEPT= option. This number is calculated according to the formula at the end of "Analyses in the LOGISTIC Statement" on page 8024, which expresses the intercept in terms of the response probability, regression coefficients, and predictor means:

$$\text{Intercept} = \log\left(\frac{0.15}{1-0.15}\right) - (0.5596158(0) + 0.7419373(7) + 0.3364722(0))$$

MULTREG Statement

MULTREG < *options* > **;**

The MULTREG statement performs power and sample size analyses for Type III F tests of sets of predictors in multiple linear regression, assuming either fixed or normally distributed predictors.

Summary of Options

Table 96.9 summarizes the *options* available in the MULTREG statement.

Table 96.9 MULTREG Statement Options

Option	Description
Define Analysis	
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
MODEL=	Specifies the assumed distribution of the predictors
NFULLPREDICTORS=	Specifies the number of predictors in the full model
NOINT	Specifies a no-intercept model
NREDUCEDPREDICTORS=	Specifies the number of predictors in the reduced model
NTESTPREDICTORS=	Specifies the number of predictors being tested
Specify Effects	
PARTIALCORR=	Specifies the partial correlation
RSQUAREDIFF=	Specifies the difference in R^2
RSQUAREFULL=	Specifies the R^2 of the full model
RSQUAREREDUCED=	Specifies the R^2 of the reduced model
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NTOTAL=	Specifies the sample size
Specify Power	
POWER=	Specifies the desired power
Control Ordering in Output	
OUTPUTORDER=	Controls the order of parameters

Table 96.10 summarizes the valid result parameters in the MULTREG statement.

Table 96.10 Summary of Result Parameters in the MULTREG Statement

Analyses	Solve For	Syntax
TEST=TYPE3	Power	POWER=.
	Sample size	NTOTAL=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual 0.05 × 100% = 5% level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

MODEL=*keyword-list*

specifies the assumed distribution of the tested predictors. MODEL=FIXED indicates a fixed predictor distribution. MODEL=RANDOM (the default) indicates a joint multivariate normal distribution for the response and tested predictors. You can use the aliases CONDITIONAL for FIXED and UNCONDITIONAL for RANDOM. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

FIXED fixed predictors

RANDOM random (multivariate normal) predictors

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NFULLPREDICTORS=*number-list*

NFULLPRED=*number-list*

specifies the number of predictors in the full model, not counting the intercept. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NOINT

specifies a no-intercept model (for both full and reduced models). By default, the intercept is included in the model. If you want to test the intercept, you can specify the NOINT option and simply consider the intercept to be one of the predictors being tested. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NREDUCEDPREDICTORS=*number-list*

NREDUCEDPRED=*number-list*

NREDPRED=*number-list*

specifies the number of predictors in the reduced model, not counting the intercept. This is the same as the difference between values of the NFULLPREDICTORS= and NTESTPREDICTORS= options. Note that supplying a value of 0 is the same as specifying an F test of a Pearson correlation. This option cannot be used at the same time as the NTESTPREDICTORS= option. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NTESTPREDICTORS=*number-list*

NTESTPRED=*number-list*

specifies the number of predictors being tested. This is the same as the difference between values of the NFULLPREDICTORS= and NREDUCEDPREDICTORS= options. Note that supplying identical values for the NTESTPREDICTORS= and NFULLPREDICTORS= options is the same as specifying an F test of a Pearson correlation. This option cannot be used at the same time as the NREDUCEDPREDICTORS= option. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). The minimum acceptable value for the sample size depends on the MODEL=, NOINT, NFULLPREDICTORS=, NTESTPREDICTORS=, and NREDUCEDPREDICTORS= options. It ranges from $p + 1$ to $p + 3$, where p is the value of the NFULLPREDICTORS option. For further information about minimum NTOTAL values, see Table 96.36. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- MODEL=
- NFULLPREDICTORS=
- NTESTPREDICTORS=
- NREDUCEDPREDICTORS=
- ALPHA=
- PARTIALCORR=
- RSQUAREFULL=
- RSQUAREREDUCED=
- RSQUAREDIFF=
- NTOTAL=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the MULTREG statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the MULTREG statement.

PARTIALCORR=*number-list*

PCORR=*number-list*

specifies the partial correlation between the tested predictors and the response, adjusting for any other predictors in the model. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

RSQUAREDIFF=*number-list*

RSQDIFF=*number-list*

specifies the difference in R^2 between the full and reduced models. This is equivalent to the proportion of variation explained by the predictors you are testing. It is also equivalent to the squared semipartial correlation of the tested predictors with the response. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

RSQUAREFULL=*number-list*

RSQFULL=*number-list*

specifies the R^2 of the full model, where R^2 is the proportion of variation explained by the model. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

RSQUAREREDUCED=*number-list*

RSQREDUCED=*number-list*

RSQRED=*number-list*

specifies the R^2 of the reduced model, where R^2 is the proportion of variation explained by the model. If the reduced model is an empty or intercept-only model (in other words, if NREDUCEDPREDICTORS=0 or NTESTPREDICTORS=NFULLPREDICTORS), then RSQUAREREDUCED=0 is assumed. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TEST=TYPE3

specifies a Type III *F* test of a set of predictors adjusting for any other predictors in the model. This is the default test option.

Restrictions on Option Combinations

To specify the number of predictors, use any two of these three options:

- the number of predictors in the full model (NFULLPREDICTORS=)
- the number of predictors in the reduced model (NREDUCEDPREDICTORS=)
- the number of predictors being tested (NTESTPREDICTORS=)

To specify the effect, choose one of the following parameterizations:

- partial correlation (by using the PARTIALCORR= option)
- R^2 for the full and reduced models (by using any two of RSQUAREDIFF=, RSQUAREFULL=, and RSQUAREREDUCED=)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the MULTREG statement.

Type III F Test of a Set of Predictors

You can express effects in terms of partial correlation, as in the following statements. Default values of the TEST=, MODEL=, and ALPHA= options specify a Type III *F* test with a significance level of 0.05, assuming normally distributed predictors.

```
proc power;
   multreg
      model = random
      nfullpredictors = 7
      ntestpredictors = 3
      partialcorr = 0.35
      ntotal = 100
      power = .;
run;
```

You can also express effects in terms of R^2:

```
proc power;
   multreg
      model = fixed
      nfullpredictors = 7
      ntestpredictors = 3
      rsquarefull = 0.9
      rsquarediff = 0.1
      ntotal = .
      power = 0.9;
run;
```

ONECORR Statement

ONECORR < *options* > **;**

The ONECORR statement performs power and sample size analyses for tests of simple and partial Pearson correlation between two variables. Both Fisher's z test and the t test are supported.

Summary of Options

Table 96.11 summarizes the *options* available in the ONECORR statement.

Table 96.11 ONECORR Statement Options

Option	Description
Define Analysis	
DIST=	Specifies the underlying distribution assumed for the test statistic
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
MODEL=	Specifies the assumed distribution of the variables
NPARTIALVARS=	Specifies the number of variables adjusted for in the correlation
NULLCORR=	Specifies the null value of the correlation
SIDES=	Specifies the number of sides and the direction of the statistical test
Specify Effects	
CORR=	Specifies the correlation

Table 96.11 *continued*

Option	Description
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NTOTAL=	Specifies the sample size
Specify Power	
POWER=	Specifies the desired power of the test
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.12 summarizes the valid result parameters in the ONECORR statement.

Table 96.12 Summary of Result Parameters in the ONECORR Statement

Analyses	Solve For	Syntax
TEST=PEARSON	Power	POWER=.
	Sample size	NTOTAL=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CORR=*number-list*

specifies the correlation between two variables, possibly adjusting for other variables as determined by the NPARTIALVARS= option. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DIST=FISHERZ | T

specifies the underlying distribution assumed for the test statistic. FISHERZ corresponds to Fisher's z normalizing transformation of the correlation coefficient. T corresponds to the t transformation of the correlation coefficient. Note that DIST=T is equivalent to analyses in the MULTREG statement with NTESTPREDICTORS=1. The default value is FISHERZ.

MODEL=*keyword-list*

specifies the assumed distribution of the first variable when DIST=T. The second variable is assumed to have a normal distribution. MODEL=FIXED indicates a fixed distribution. MODEL=RANDOM (the default) indicates a joint bivariate normal distribution with the second variable. You can use the aliases CONDITIONAL for FIXED and UNCONDITIONAL for RANDOM. This option can be used only for DIST=T. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

FIXED fixed variables

RANDOM random (bivariate normal) variables

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NPARTIALVARS=*number-list*

NPVARS=*number-list*

specifies the number of variables adjusted for in the correlation between the two primary variables. The default value is 0, which corresponds to a simple correlation. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). Values for the sample size must be at least $p + 3$ when DIST=T and MODEL=CONDITIONAL, and at least $p + 4$ when either DIST=FISHER or when DIST=T and MODEL=UNCONDITIONAL, where p is the value of the NPARTIALVARS option. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLCORR=*number-list*

NULLC=*number-list*

specifies the null value of the correlation. The default value is 0. This option can be used only with the DIST=FISHERZ analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- MODEL=
- SIDES=
- NULLCORR=
- ALPHA=
- NPARTIALVARS=
- CORR=
- NTOTAL=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the ONECORR statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the ONECORR statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test. You can specify the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating a correlation greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating a correlation less than the null value.

By default, SIDES=2.

TEST=PEARSON

specifies a test of the Pearson correlation coefficient between two variables, possibly adjusting for other variables. This is the default test option.

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the ONECORR statement.

Fisher's z Test for Pearson Correlation

The following statements demonstrate a power computation for Fisher's z test for correlation. Default values of TEST=PEARSON, ALPHA=0.05, SIDES=2, and NPARTIALVARS=0 are assumed.

```
proc power;
   onecorr dist=fisherz
      nullcorr = 0.15
      corr = 0.35
      ntotal = 180
      power = .;
run;
```

t Test for Pearson Correlation

The following statements demonstrate a sample size computation for the *t* test for correlation. Default values of TEST=PEARSON, MODEL=RANDOM, ALPHA=0.05, and SIDES=2 are assumed.

```
proc power;
   onecorr dist=t
      npartialvars = 4
      corr = 0.45
      ntotal = .
      power = 0.85;
```

```
run;
```

ONESAMPLEFREQ Statement

ONESAMPLEFREQ < *options* > **;**

The ONESAMPLEFREQ statement performs power and sample size analyses for exact and approximate tests (including equivalence, noninferiority, and superiority) and confidence interval precision for a single binomial proportion.

Summary of Options

Table 96.13 summarizes the *options* available in the ONESAMPLEFREQ statement.

Table 96.13 ONESAMPLEFREQ Statement Options

Option	Description
Define Analysis	
CI=	Specifies an analysis of precision of a confidence interval
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
EQUIVBOUNDS=	Specifies the lower and upper equivalence bounds
LOWER=	Specifies the lower equivalence bound
MARGIN=	Specifies the equivalence or noninferiority or superiority margin
NULLPROPORTION=	Specifies the null proportion
SIDES=	Specifies the number of sides and the direction of the statistical test
UPPER=	Specifies the upper equivalence bound
Specify Effect	
HALFWIDTH=	Specifies the desired confidence interval half-width
PROPORTION=	Specifies the binomial proportion
Specify Variance Estimation	
VAREST=	Specifies how the variance is computed
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NTOTAL=	Specifies the sample size
Specify Power and Related Probabilities	
POWER=	Specifies the desired power of the test
PROBWIDTH=	Specifies the probability of obtaining a confidence interval half-width less than or equal to the value specified by HALFWIDTH=
Choose Computational Method	
METHOD=	Specifies the computational method
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.14 summarizes the valid result parameters for different analyses in the ONESAMPLEFREQ statement.

Table 96.14 Summary of Result Parameters in the ONESAMPLEFREQ Statement

Analyses	Solve For	Syntax
CI=WILSON	Prob(width)	PROBWIDTH=.
CI=AGRESTICOULL	Prob(width)	PROBWIDTH=.
CI=JEFFREYS	Prob(width)	PROBWIDTH=.
CI=EXACT	Prob(width)	PROBWIDTH=.
CI=WALD	Prob(width)	PROBWIDTH=.
CI=WALD_CORRECT	Prob(width)	PROBWIDTH=.
TEST=ADJZ METHOD=EXACT	Power	POWER=.
TEST=ADJZ METHOD=NORMAL	Power Sample size	POWER=. NTOTAL=.
TEST=EQUIV_ADJZ METHOD=EXACT	Power	POWER=.
TEST=EQUIV_ADJZ METHOD=NORMAL	Power Sample size	POWER=. NTOTAL=.
TEST=EQUIV_EXACT	Power	POWER=.
TEST=EQUIV_Z METHOD=EXACT	Power	POWER=.
TEST=EQUIV_Z METHOD=NORMAL	Power Sample size	POWER=. NTOTAL=.
TEST=EXACT	Power	POWER=.
TEST=Z METHOD=EXACT	Power	POWER=.
TEST=Z METHOD=NORMAL	Power Sample size	POWER=. NTOTAL=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For the equivalence analyses (TEST=EQUIV_ADJZ | EQUIV_EXACT | EQUIV_Z), a value of α corresponds to a $100(1 - 2\alpha)\%$ confidence interval, and the value must be less than 0.5. For all the other analyses, a value of α corresponds to a $100(1 - \alpha)\%$ confidence interval. If the CI= and SIDES=1 options are used, then the value must be less than 0.5. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CI

CI=AGRESTICOULL | AC

CI=JEFFREYS

CI=EXACT | CLOPPERPEARSON | CP

CI=WALD

CI=WALD_CORRECT

CI=WILSON | SCORE

specifies an analysis of precision of a confidence interval for the sample binomial proportion.

The value of the CI= option specifies the type of confidence interval. The CI=AGRESTICOULL option is a generalization of the "Adjusted Wald / add 2 successes and 2 failures" interval of Agresti and Coull (1998) and is presented in Brown, Cai, and DasGupta (2001). It corresponds to the TABLES / BINOMIAL (AGRESTICOULL) option in PROC FREQ. The CI=JEFFREYS option specifies the equal-tailed Jeffreys prior Bayesian interval, which corresponds to the TABLES / BINOMIAL (JEFFREYS) option in PROC FREQ. The CI=EXACT option specifies the exact Clopper-Pearson confidence interval based on enumeration, which corresponds to the TABLES / BINOMIAL (EXACT) option in PROC FREQ. The CI=WALD option specifies the confidence interval based on the Wald test (also commonly called the z test or normal-approximation test), which corresponds to the TABLES / BINOMIAL (WALD) option in PROC FREQ. The CI=WALD_CORRECT option specifies the confidence interval based on the Wald test with continuity correction, which corresponds to the TABLES / BINOMIAL (CORRECT WALD) option in PROC FREQ. The CI=WILSON option (the default) specifies the confidence interval based on the score statistic, which corresponds to the TABLES / BINOMIAL (WILSON) option in PROC FREQ.

Instead of power, the relevant probability for this analysis is the probability of achieving a desired precision. Specifically, it is the probability that the half-width of the confidence interval will be at most the value specified by the HALFWIDTH= option.

EQUIVBOUNDS=*grouped-number-list*

specifies the lower and upper equivalence bounds, representing the same information as the combination of the LOWER= and UPPER= options but grouping them together. The EQUIVBOUNDS= option can be used only with equivalence analyses (TEST=EQUIV_ADJZ | EQUIV_EXACT | EQUIV_Z). Values must be strictly between 0 and 1. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

HALFWIDTH=*number-list*

specifies the desired confidence interval half-width. The half-width for a two-sided interval is the length of the confidence interval divided by two. This option can be used only with the CI= analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

LOWER=*number-list*

specifies the lower equivalence bound for the binomial proportion. The LOWER= option can be used only with equivalence analyses (TEST=EQUIV_ADJZ | EQUIV_EXACT | EQUIV_Z). Values must be strictly between 0 and 1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

MARGIN=*number-list*

specifies the equivalence or noninferiority or superiority margin, depending on the analysis.

The MARGIN= option can be used with one-sided analyses (SIDES = 1 | U | L), in which case it specifies the margin added to the null proportion value in the hypothesis test, resulting in a noninferiority or superiority test (depending on the agreement between the effect and hypothesis directions and the sign of the margin). A test with a null proportion p_0 and a margin m is the same as a test with null proportion $p_0 + m$ and no margin.

The MARGIN= option can also be used with equivalence analyses (TEST=EQUIV_ADJZ | EQUIV_EXACT | EQUIV_Z) when the NULLPROPORTION= option is used, in which case it specifies the lower and upper equivalence bounds as $p_0 - m$ and $p_0 + m$, where p_0 is the value of the NULLPROPORTION= option and m is the value of the MARGIN= option.

The MARGIN= option cannot be used in conjunction with the SIDES=2 option. (Instead, specify an equivalence analysis by using TEST=EQUIV_ADJZ or TEST=EQUIV_EXACT or TEST=EQUIV_Z). Also, the MARGIN= option cannot be used with the CI= option.

Values must be strictly between –1 and 1. In addition, the sum of NULLPROPORTION and MARGIN must be strictly between 0 and 1 for one-sided analyses, and the derived lower equivalence bound (2 * NULLPROPORTION – MARGIN) must be strictly between 0 and 1 for equivalence analyses.

For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

METHOD=EXACT | NORMAL

specifies the computational method. METHOD=EXACT (the default) computes exact results by using the binomial distribution. METHOD=NORMAL computes approximate results by using the normal approximation to the binomial distribution.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. This option is invalid when the METHOD=EXACT option is specified. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLPROPORTION=*number-list*

NULLP=*number-list*

specifies the null proportion. A value of 0.5 corresponds to the sign test. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=

- NULLPROPORTION=
- ALPHA=
- PROPORTION=
- NTOTAL=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the ONESAMPLEFREQ statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the ONESAMPLEFREQ statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PROBWIDTH=*number-list*

specifies the desired probability of obtaining a confidence interval half-width less than or equal to the value specified by the HALFWIDTH= option. A missing value (PROBWIDTH=.) requests a solution for this probability. Values are expressed as probabilities (for example, 0.9) rather than percentages. This option can be used only with the CI= analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PROPORTION=*number-list*

P=*number-list*

specifies the binomial proportion—that is, the expected proportion of successes in the hypothetical binomial trial. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. You can specify the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating a proportion greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating a proportion less than the null value.

If the effect size is zero, then SIDES=1 is not permitted; instead, specify the direction of the test explicitly in this case with either SIDES=L or SIDES=U. By default, SIDES=2.

TEST=ADJZ | EQUIV_ADJZ | EQUIV_EXACT | EQUIV_Z | EXACT | Z
TEST

specifies the statistical analysis. TEST=ADJZ specifies a normal-approximate z test with continuity adjustment. TEST=EQUIV_ADJZ specifies a normal-approximate two-sided equivalence test based on the z statistic with continuity adjustment and a TOST (two one-sided tests) procedure. TEST=EQUIV_EXACT specifies the exact binomial two-sided equivalence test based on a TOST (two one-sided tests) procedure. TEST=EQUIV_Z specifies a normal-approximate two-sided equivalence test based on the z statistic without any continuity adjustment, which is the same as the chi-square statistic, and a TOST (two one-sided tests) procedure. TEST or TEST=EXACT (the default) specifies the exact binomial test. TEST=Z specifies a normal-approximate z test without any continuity adjustment, which is the same as the chi-square test when SIDES=2.

UPPER=*number-list*

specifies the upper equivalence bound for the binomial proportion. The UPPER= option can be used only with equivalence analyses (TEST=EQUIV_ADJZ | EQUIV_EXACT | EQUIV_Z). Values must be strictly between 0 and 1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

VAREST=*keyword-list*

specifies how the variance is computed in the test statistic for the TEST=Z, TEST=ADJZ, TEST=EQUIV_Z, and TEST=EQUIV_ADJZ analyses. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. Valid *keywords* are as follows:

NULL estimates the variance by using the null proportion(s) (specified by some combination of the NULLPROPORTION=, MARGIN=, LOWER=, and UPPER= options). For TEST=Z and TEST=ADJZ, the null proportion is the value of the NULLPROPORTION= option plus the value of the MARGIN= option (if it is used). For TEST=EQUIV_Z and TEST=EQUIV_ADJZ, there are two null proportions, which correspond to the lower and upper equivalence bounds, one for each test in the TOST (two one-sided tests) procedure.

SAMPLE estimates the variance by using the observed sample proportion.

By default, VAREST=NULL.

The VAREST= option is valid only if you also specify one of the following options: TEST=Z, TEST=ADJZ, TEST=EQUIV_Z, or TEST=EQUIV_ADJZ.

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the ONESAMPLEFREQ statement.

Exact Test of a Binomial Proportion

The following statements demonstrate a power computation for the exact test of a binomial proportion. Defaults for the SIDES= and ALPHA= options specify a two-sided test with a 0.05 significance level.

```
proc power;
   onesamplefreq test=exact
      nullproportion = 0.2
      proportion = 0.3
      ntotal = 100
      power = .;
run;
```

z Test

The following statements demonstrate a sample size computation for the z test of a binomial proportion. Defaults for the SIDES=, ALPHA=, and VAREST= options specify a two-sided test with a 0.05 significance level that uses the null variance estimate.

```
proc power;
   onesamplefreq test=z method=normal
      nullproportion = 0.8
      proportion = 0.85
      sides = u
      ntotal = .
      power = .9;
run;
```

z Test with Continuity Adjustment

The following statements demonstrate a sample size computation for the z test of a binomial proportion with a continuity adjustment. Defaults for the SIDES=, ALPHA=, and VAREST= options specify a two-sided test with a 0.05 significance level that uses the null variance estimate.

```
proc power;
   onesamplefreq test=adjz method=normal
      nullproportion = 0.15
      proportion = 0.1
      sides = l
      ntotal = .
      power = .9;
run;
```

Exact Equivalence Test for a Binomial Proportion

You can specify equivalence bounds by using the EQUIVBOUNDS= option, as in the following statements:

```
proc power;
   onesamplefreq test=equiv_exact
      proportion = 0.35
      equivbounds = (0.2 0.4)
      ntotal = 500
      power = .;
run;
```

You can also specify the combination of NULLPROPORTION= and MARGIN= options:

```
proc power;
   onesamplefreq test=equiv_exact
      proportion = 0.35
      nullproportion = 0.3
      margin = 0.1
      ntotal = 500
      power = .;
run;
```

Finally, you can specify the combination of LOWER= and UPPER= options:

```
proc power;
   onesamplefreq test=equiv_exact
      proportion = 0.35
      lower = 0.2
      upper = 0.4
      ntotal = 500
      power = .;
run;
```

Note that the three preceding analyses are identical.

Exact Noninferiority Test for a Binomial Proportion

A noninferiority test corresponds to an upper one-sided test with a negative-valued margin, as demonstrated in the following statements:

```
proc power;
   onesamplefreq test=exact
      sides = U
      proportion = 0.15
      nullproportion = 0.1
      margin = -0.02
      ntotal = 130
      power = .;
run;
```

Exact Superiority Test for a Binomial Proportion

A superiority test corresponds to an upper one-sided test with a positive-valued margin, as demonstrated in the following statements:

```
proc power;
   onesamplefreq test=exact
      sides = U
      proportion = 0.15
      nullproportion = 0.1
      margin = 0.02
      ntotal = 130
      power = .;
run;
```

Confidence Interval Precision

The following statements performs a confidence interval precision analysis for the Wilson score-based confidence interval for a binomial proportion. The default value of the ALPHA= option specifies a confidence level of 0.95.

```
proc power;
   onesamplefreq ci=wilson
      halfwidth = 0.1
      proportion = 0.3
      ntotal = 70
      probwidth = .;
run;
```

Restrictions on Option Combinations

To specify the equivalence bounds for TEST=EQUIV_ADJZ, TEST=EQUIV_EXACT, and TEST=EQUIV_Z, use any of these three option sets:

- lower and upper equivalence bounds, using the EQUIVBOUNDS= option
- lower and upper equivalence bounds, using the LOWER= and UPPER= options
- null proportion (NULLPROPORTION=) and margin (MARGIN=)

ONESAMPLEMEANS Statement

ONESAMPLEMEANS < *options* > **;**

The ONESAMPLEMEANS statement performs power and sample size analyses for *t* tests, equivalence tests, and confidence interval precision involving one sample.

Summary of Options

Table 96.15 summarizes the *options* available in the ONESAMPLEMEANS statement.

Table 96.15 ONESAMPLEMEANS Statement Options

Option	Description
Define Analysis	
CI=	Specifies an analysis of precision of the confidence interval for the mean
DIST=	Specifies the underlying distribution assumed for the test statistic
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
LOWER=	Specifies the lower equivalence bound for the mean
NULLMEAN=	Specifies the null mean
SIDES=	Specifies the number of sides and the direction of the statistical test
UPPER=	Specifies the upper equivalence bound for the mean

Table 96.15 *continued*

Option	Description
Specify Effect	
HALFWIDTH=	Specifies the desired confidence interval half-width
MEAN=	Specifies the mean
Specify Variability	
CV=	Specifies the coefficient of variation
STDDEV=	Specifies the standard deviation
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NTOTAL=	Specifies the sample size
Specify Power and Related Probabilities	
POWER=	Specifies the desired power of the test
PROBTYPE=	Specifies the type of probability for the PROBWIDTH= option
PROBWIDTH=	Specifies the probability of obtaining a confidence interval half-width less than or equal to the value specified by HALFWIDTH=
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.16 summarizes the valid result parameters for different analyses in the ONESAMPLEMEANS statement.

Table 96.16 Summary of Result Parameters in the ONESAMPLEMEANS Statement

Analyses	Solve For	Syntax
TEST=T DIST=NORMAL	Power	POWER=.
	Sample size	NTOTAL=.
	Alpha	ALPHA=.
	Mean	MEAN=.
	Standard Deviation	STDDEV=.
TEST=T DIST=LOGNORMAL	Power	POWER=.
	Sample size	NTOTAL=.
TEST=EQUIV	Power	POWER=.
	Sample size	NTOTAL=.
CI=T	Prob(width)	PROBWIDTH=.
	Sample size	NTOTAL=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test or requests a solution for alpha by specifying a missing value (ALPHA=.). The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For the TEST=EQUIV analysis, a value of α corresponds to a $100(1 - 2\alpha)\%$ confidence interval, and the value must be less than 0.5. For all the other analyses, a value of α corresponds to a $100(1 - \alpha)\%$ confidence interval. If the CI= and SIDES=1 options are used, then the value must be less than 0.5. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CI

CI=T

specifies an analysis of precision of the confidence interval for the mean. Instead of power, the relevant probability for this analysis is the probability of achieving a desired precision. Specifically, it is the probability that the half-width of the confidence interval will be at most the value specified by the HALFWIDTH= option. If neither the CI= option nor the TEST= option is used, the default is TEST=T.

CV=*number-list*

specifies the coefficient of variation, defined as the ratio of the standard deviation to the mean on the original data scale. You can use this option only with DIST=LOGNORMAL. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DIST=LOGNORMAL | NORMAL

specifies the underlying distribution assumed for the test statistic. NORMAL corresponds the normal distribution, and LOGNORMAL corresponds to the lognormal distribution. The default value is NORMAL.

HALFWIDTH=*number-list*

specifies the desired confidence interval half-width. The half-width is defined as the distance between the point estimate and a finite endpoint. This option can be used only with the CI=T analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

LOWER=*number-list*

specifies the lower equivalence bound for the mean. This option can be used only with the TEST=EQUIV analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

MEAN=*number-list*

specifies the mean, in the original scale, or requests a solution for the mean by specifying a missing value (MEAN=.). The mean is arithmetic if DIST=NORMAL and geometric if DIST=LOGNORMAL. This option can be used only with the TEST=T and TEST=EQUIV analyses. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLMEAN=*number-list*

NULLM=*number-list*

specifies the null mean, in the original scale (whether DIST=NORMAL or DIST=LOGNORMAL). The default value is 0 when DIST=NORMAL and 1 when DIST=LOGNORMAL. This option can be used only with the TEST=T analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=
- NULLMEAN=
- LOWER=
- UPPER=
- ALPHA=
- MEAN=
- HALFWIDTH=
- STDDEV=
- CV=
- NTOTAL=
- POWER=
- PROBTYPE=
- PROBWIDTH=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the ONESAMPLEMEANS statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the ONESAMPLEMEANS statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. This option can be used only with the TEST=T and TEST=EQUIV analyses. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PROBTYPE=*keyword-list*

specifies the type of probability for the PROBWIDTH= option. A value of CONDITIONAL (the default) indicates the conditional probability that the confidence interval half-width is at most the value specified by the HALFWIDTH= option, given that the true mean is captured by the confidence interval. A value of UNCONDITIONAL indicates the unconditional probability that the confidence interval half-width is at most the value specified by the HALFWIDTH= option. You can use the alias GIVENVALIDITY for CONDITIONAL. The PROBTYPE= option can be used only with the CI=T analysis. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CONDITIONAL width probability conditional on interval containing the mean

UNCONDITIONAL unconditional width probability

PROBWIDTH=*number-list*

specifies the desired probability of obtaining a confidence interval half-width less than or equal to the value specified by the HALFWIDTH= option. A missing value (PROBWIDTH=.) requests a solution for this probability. The type of probability is controlled with the PROBTYPE= option. Values are expressed as probabilities (for example, 0.9) rather than percentages. This option can be used only with the CI=T analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test or confidence interval. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. Valid *keywords* and their interpretation for the TEST= analyses are as follows:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating a mean greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating a mean less than the null value.

For confidence intervals, SIDES=U refers to an interval between the lower confidence limit and infinity, and SIDES=L refers to an interval between minus infinity and the upper confidence limit. For both of these cases and SIDES=1, the confidence interval computations are equivalent. The SIDES= option can be used only with the TEST=T and CI=T analyses. By default, SIDES=2.

STDDEV=*number-list*

STD=*number-list*

specifies the standard deviation, or requests a solution for the standard deviation by specifying a missing value (STDDEV=.). You can use this option only with DIST=NORMAL. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TEST=EQUIV | T
TEST

specifies the statistical analysis. TEST=EQUIV specifies an equivalence test of the mean by using a two one-sided tests (TOST) analysis (Schuirmann 1987). TEST or TEST=T (the default) specifies a *t* test on the mean. If neither the TEST= option nor the CI= option is used, the default is TEST=T.

UPPER=*number-list*

specifies the upper equivalence bound for the mean, in the original scale (whether DIST=NORMAL or DIST=LOGNORMAL). This option can be used only with the TEST=EQUIV analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

Restrictions on Option Combinations

To define the analysis, choose one of the following parameterizations:

- a statistical test (by using the TEST= option)
- confidence interval precision (by using the CI= option)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the ONESAMPLEMEANS statement.

One-Sample t Test

The following statements demonstrate a power computation for the one-sample *t* test. Default values for the DIST=, SIDES=, NULLMEAN=, and ALPHA= options specify a two-sided test for zero mean with a normal distribution and a significance level of 0.05.

```
proc power;
   onesamplemeans test=t
      mean = 7
      stddev = 3
      ntotal = 50
      power = .;
run;
```

One-Sample t Test with Lognormal Data

The following statements demonstrate a sample size computation for the one-sample *t* test for lognormal data. Default values for the SIDES=, NULLMEAN=, and ALPHA= options specify a two-sided test for unit mean with a significance level of 0.05.

```
proc power;
   onesamplemeans test=t dist=lognormal
      mean = 7
      cv = 0.8
      ntotal = .
      power = 0.9;
run;
```

Equivalence Test for Mean of Normal Data

The following statements demonstrate a power computation for the TOST equivalence test for a normal mean. Default values for the DIST= and ALPHA= options specify a normal distribution and a significance level of 0.05.

```
proc power;
   onesamplemeans test=equiv
      lower = 2
      upper = 7
      mean = 4
      stddev = 3
      ntotal = 100
      power = .;
run;
```

Equivalence Test for Mean of Lognormal Data

The following statements demonstrate a sample size computation for the TOST equivalence test for a lognormal mean. The default of ALPHA=0.05 specifies a significance level of 0.05.

```
proc power;
   onesamplemeans test=equiv dist=lognormal
      lower = 1
      upper = 5
      mean = 3
      cv = 0.6
      ntotal = .
      power = 0.85;
run;
```

Confidence Interval for Mean

By default CI=T analyzes the conditional probability of obtaining the desired precision, given that the interval contains the true mean, as in the following statements. The defaults of SIDES=2 and ALPHA=0.05 specify a two-sided interval with a confidence level of 0.95.

```
proc power;
   onesamplemeans ci = t
      halfwidth = 14
      stddev = 8
      ntotal = 50
      probwidth = .;
run;
```

ONEWAYANOVA Statement

ONEWAYANOVA < *options* > **;**

The ONEWAYANOVA statement performs power and sample size analyses for one-degree-of-freedom contrasts and the overall F test in one-way analysis of variance.

Summary of Options

Table 96.17 summarizes the *options* available in the ONEWAYANOVA statement.

Table 96.17 ONEWAYANOVA Statement Options

Option	Description
Define Analysis	
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
CONTRAST=	Specifies coefficients for single-degree-of-freedom hypothesis tests
NULLCONTRAST=	Specifies the null value of the contrast
SIDES=	Specifies the number of sides and the direction of the statistical test
Specify Effect	
GROUPMEANS=	Specifies the group means
Specify Variability	
STDDEV=	Specifies the error standard deviation
Specify Sample Size and Allocation	
GROUPNS=	Specifies the group sample sizes
GROUPWEIGHTS=	Specifies the sample size allocation weights for the groups
NFRACTIONAL	Enables fractional input and output for sample sizes
NPERGROUP=	Specifies the common sample size per group
NTOTAL=	Specifies the sample size
Specify Power	
POWER=	Specifies the desired power of the test
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.18 summarizes the valid result parameters for different analyses in the ONEWAYANOVA statement.

Table 96.18 Summary of Result Parameters in the ONEWAYANOVA Statement

Analyses	Solve For	Syntax
TEST=CONTRAST	Power	POWER=.
	Sample size	NTOTAL=.

Table 96.18 *continued*

Analyses	Solve For	Syntax
		NPERGROUP==.
TEST=OVERALL	Power Sample size	POWER=. NTOTAL=. NPERGROUP==.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CONTRAST= (*values*) < (... *values*) >

specifies coefficients for single-degree-of-freedom hypothesis tests. You must provide a coefficient for every mean appearing in the GROUPMEANS= option. Specify multiple contrasts either with additional sets of coefficients or with additional CONTRAST= options. For example, you can specify two different contrasts of five means by using the following:

```
CONTRAST = (1 -1 0 0 0) (1 0 -1 0 0)
```

GROUPMEANS=*grouped-number-list*

GMEANS=*grouped-number-list*

specifies the group means. This option is used to implicitly set the number of groups. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPNS=*grouped-number-list*

GNS=*grouped-number-list*

specifies the group sample sizes. The number of groups represented must be the same as with the GROUPMEANS= option. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPWEIGHTS=*grouped-number-list*

GWEIGHTS=*grouped-number-list*

specifies the sample size allocation weights for the groups. This option controls how the total sample size is divided between the groups. Each set of values across all groups represents relative allocation weights. Additionally, if the NFRACTIONAL option is not used, the total sample size is restricted to be equal to a multiple of the sum of the group weights (so that the resulting design has an integer sample size for each group while adhering exactly to the group allocation weights). The number of groups represented must be the same as with the GROUPMEANS= option. Values must be integers unless the NFRACTIONAL option is used. The default value is 1 for each group, amounting to a balanced design. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NPERGROUP=*number-list*

NPERG=*number-list*

specifies the common sample size per group or requests a solution for the common sample size per group by specifying a missing value (NPERGROUP==.). Use of this option implicitly specifies a balanced design. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLCONTRAST=*number-list*

NULLC=*number-list*

specifies the null value of the contrast. The default value is 0. This option can be used only with the TEST=CONTRAST analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- CONTRAST=
- SIDES=
- NULLCONTRAST=
- ALPHA=
- GROUPMEANS=
- STDDEV=
- GROUPWEIGHTS=
- NTOTAL=
- NPERGROUP==
- GROUPNS=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the ONEWAYANOVA statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the ONEWAYANOVA statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. You can specify the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating an effect greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating an effect less than the null value.

You can use this option only with the TEST=CONTRAST analysis. By default, SIDES=2.

STDDEV=*number-list*

STD=*number-list*

specifies the error standard deviation. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TEST=CONTRAST | OVERALL

specifies the statistical analysis. TEST=CONTRAST specifies a one-degree-of-freedom test of a contrast of means. The test is the usual F test for the two-sided case and the usual t test for the one-sided case. TEST=OVERALL specifies the overall F test of equality of all means. The default is TEST=CONTRAST if the CONTRAST= option is used, and TEST=OVERALL otherwise.

Restrictions on Option Combinations

To specify the sample size and allocation, choose one of the following parameterizations:

- sample size per group in a balanced design (by using the NPERGROUP== option)
- total sample size and allocation weights (by using the NTOTAL= and GROUPWEIGHTS= options)
- individual group sample sizes (by using the GROUPNS= option)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the ONEWAYANOVA statement.

One-Degree-of-Freedom Contrast

You can use the NPERGROUP== option in a balanced design, as in the following statements. Default values for the SIDES=, NULLCONTRAST=, and ALPHA= options specify a two-sided test for a contrast value of 0 with a significance level of 0.05.

```
proc power;
   onewayanova test=contrast
      contrast = (1 0 -1)
      groupmeans = 3 | 7 | 8
      stddev = 4
      npergroup = 50
      power = .;
run;
```

You can also specify an unbalanced design with the NTOTAL= and GROUPWEIGHTS= options:

```
proc power;
   onewayanova test=contrast
      contrast = (1 0 -1)
      groupmeans = 3 | 7 | 8
      stddev = 4
      groupweights = (1 2 2)
      ntotal = .
      power = 0.9;
run;
```

Another way to specify the sample sizes is with the GROUPNS= option:

```
proc power;
   onewayanova test=contrast
      contrast = (1 0 -1)
      groupmeans = 3 | 7 | 8
      stddev = 4
      groupns = (20 40 40)
      power = .;
run;
```

Overall F Test

The following statements demonstrate a power computation for the overall F test in a one-way ANOVA. The default of ALPHA=0.05 specifies a significance level of 0.05.

```
proc power;
   onewayanova test=overall
      groupmeans = 3 | 7 | 8
      stddev = 4
      npergroup = 50
      power = .;
run;
```

PAIREDFREQ Statement

PAIREDFREQ < *options* > **;**

The PAIREDFREQ statement performs power and sample size analyses for McNemar's test for paired proportions.

Summary of Options

Table 96.19 summarizes the *options* available in the PAIREDFREQ statement.

Table 96.19 PAIREDFREQ Statement Options

Option	Description
Define Analysis	
DIST=	Specifies the underlying distribution assumed for the test statistic
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
NULLDISCPROPRATIO=	Specifies the null value of the ratio of discordant proportions
SIDES=	Specifies the number of sides and the direction of the statistical test or confidence interval
Specify Effects	
CORR=	Specifies the correlation ϕ between members of a pair
DISCPROPDIFF=	Specifies the discordant proportion difference $p_{01} - p_{10}$
DISCPROPORTIONS=	Specifies the two discordant proportions, p_{10} and p_{01}
DISCPROPRATIO=	Specifies the ratio p_{01}/p_{10}
ODDSRATIO=	Specifies the odds ratio $[p_{\cdot 1}/(1 - p_{\cdot 1})] / [p_{1\cdot}/(1 - p_{1\cdot})]$
PAIREDPROPORTIONS=	Specifies the two paired proportions, $p_{1\cdot}$ and $p_{\cdot 1}$
PROPORTIONDIFF=	Specifies the proportion difference $p_{\cdot 1} - p_{1\cdot}$
REFPROPORTION=	Specifies either the reference first proportion $p_{1\cdot}$ or the reference discordant proportion p_{10}
RELATIVERISK=	Specifies the relative risk $p_{\cdot 1}/p_{1\cdot}$
TOTALPROPDISC=	Specifies the discordant proportion sum, $p_{10} + p_{01}$
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NPAIRS=	Specifies the total number of proportion pairs
Specify Power	
POWER=	Specifies the desired power of the test
Choose Computational Method	
METHOD=	Specifies the computational method
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.20 summarizes the valid result parameters in the PAIREDFREQ statement.

Table 96.20 Summary of Result Parameters in the PAIREDFREQ Statement

Analyses	Solve For	Syntax
TEST=MCNEMAR METHOD=CONNOR	Power Sample size	POWER=. NPAIRS=.
TEST=MCNEMAR METHOD=EXACT	Power	POWER=.
TEST=MCNEMAR METHOD=MIETTINEN	Power Sample size	POWER=. NPAIRS=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CORR=*number-list*

specifies the correlation ϕ between members of a pair. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DISCPROPORTIONS=*grouped-number-list*

DISCPS=*grouped-number-list*

specifies the two discordant proportions, p_{10} and p_{01}. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DISCPROPDIFF=*number-list*

DISCPDIFF=*number-list*

specifies the difference $p_{01} - p_{10}$ between discordant proportions. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DISCPROPRATIO=*number-list*

DISCPRATIO=*number-list*

specifies the ratio p_{01}/p_{10} of discordant proportions. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DIST=EXACT_COND | NORMAL

specifies the underlying distribution assumed for the test statistic. EXACT_COND corresponds to the exact conditional test, based on the exact binomial distribution of the two types of discordant pairs given the total number of discordant pairs. NORMAL corresponds to the conditional test based on the normal approximation to the binomial distribution of the two types of discordant pairs given the total number of discordant pairs. The default value is EXACT_COND.

METHOD=CONNOR | EXACT | MIETTINEN

specifies the computational method. METHOD=EXACT (the default) uses the exact binomial distributions of the total number of discordant pairs and the two types of discordant pairs. METHOD=CONNOR uses an approximation from Connor (1987), and METHOD=MIETTINEN uses an approximation from Miettinen (1968). The CONNOR and MIETTINEN methods are valid only for DIST=NORMAL.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option. This option cannot be used with METHOD=EXACT.

NPAIRS=*number-list*

specifies the total number of proportion pairs (concordant and discordant) or requests a solution for the number of pairs by specifying a missing value (NPAIRS=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLDISCPROPRATIO=*number-list*

NULLDISCPRATIO=*number-list*

NULLRATIO=*number-list*

NULLR=*number-list*

specifies the null value of the ratio of discordant proportions. The default value is 1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

ODDSRATIO=*number-list*

OR=*number-list*

specifies the odds ratio $[p_{\cdot 1}/(1-p_{\cdot 1})]\,/\,[p_{1\cdot}/(1-p_{1\cdot})]$. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=
- NULLDISCPROPRATIO=
- ALPHA=
- PAIREDPROPORTIONS=
- PROPORTIONDIFF=
- ODDSRATIO=
- RELATIVERISK=
- CORR=
- DISCPROPORTIONS=
- DISCPROPDIFF=
- TOTALPROPDISC=
- REFPROPORTION=
- DISCPROPRATIO=

- NPAIRS=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the PAIREDFREQ statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the PAIREDFREQ statement.

PAIREDPROPORTIONS=*grouped-number-list*

PPROPORTIONS=*grouped-number-list*

PAIREDPS=*grouped-number-list*

PPS=*grouped-number-list*

specifies the two paired proportions, $p_{1\cdot}$ and $p_{\cdot 1}$. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PROPORTIONDIFF=*number-list*

PDIFF=*number-list*

specifies the proportion difference $p_{\cdot 1} - p_{1\cdot}$. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

REFPROPORTION=*number-list*

REFP=*number-list*

specifies either the reference first proportion $p_{1\cdot}$ (when used in conjunction with the PROPORTIONDIFF=, ODDSRATIO=, or RELATIVERISK= option) or the reference discordant proportion p_{10} (when used in conjunction with the DISCPROPDIFF= or DISCPROPRATIO= option). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

RELATIVERISK=*number-list*

RR=*number-list*

specifies the relative risk $p_{\cdot 1}/p_{1\cdot}$. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test or confidence interval. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. You can specify the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating an effect greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating an effect less than the null value.

If the effect size is zero, then SIDES=1 is not permitted; instead, specify the direction of the test explicitly in this case with either SIDES=L or SIDES=U. By default, SIDES=2.

TEST=MCNEMAR
specifies the McNemar test of paired proportions. This is the default test option.

TOTALPROPDISC=*number-list*

TOTALPDISC=*number-list*

PDISC=*number-list*
specifies the sum of the two discordant proportions, $p_{10} + p_{01}$. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

Restrictions on Option Combinations

To specify the proportions, choose one of the following parameterizations:

- discordant proportions (using the DISCPROPORTIONS= option)
- difference and sum of discordant proportions (using the DISCPROPDIFF= and TOTALPROPDISC=options)
- difference of discordant proportions and reference discordant proportion (using the DISCPROPDIFF= and REFPROPORTION= options)
- ratio of discordant proportions and reference discordant proportion (using the DISCPROPRATIO= and REFPROPORTION= options)
- ratio and sum of discordant proportions (using the DISCPROPRATIO= and TOTALPROPDISC=options)
- paired proportions and correlation (using the PAIREDPROPORTIONS= and CORR= options)
- proportion difference, reference proportion, and correlation (using the PROPORTIONDIFF=, REFPROPORTION=, and CORR= options)
- odds ratio, reference proportion, and correlation (using the ODDSRATIO=, REFPROPORTION=, and CORR= options)
- relative risk, reference proportion, and correlation (using the RELATIVERISK=, REFPROPORTION=, and CORR= options)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the PAIREDFREQ statement.

McNemar Exact Conditional Test

You can express effects in terms of the individual discordant proportions, as in the following statements. Default values for the TEST=, SIDES=, ALPHA=, and NULLDISCPROPRATIO= options specify a two-sided McNemar test for no effect with a significance level of 0.05.

```
proc power;
   pairedfreq dist=exact_cond
      discproportions = 0.15 | 0.45
      npairs = 80
      power = .;
run;
```

You can also express effects in terms of the difference and sum of discordant proportions:

```
proc power;
   pairedfreq dist=exact_cond
      discpropdiff = 0.3
      totalpropdisc = 0.6
      npairs = 80
      power = .;
run;
```

You can also express effects in terms of the difference of discordant proportions and the reference discordant proportion:

```
proc power;
   pairedfreq dist=exact_cond
      discpropdiff = 0.3
      refproportion = 0.15
      npairs = 80
      power = .;
run;
```

You can also express effects in terms of the ratio of discordant proportions and the denominator of the ratio:

```
proc power;
   pairedfreq dist=exact_cond
      discpropratio = 3
      refproportion = 0.15
      npairs = 80
      power = .;
run;
```

You can also express effects in terms of the ratio and sum of discordant proportions:

```
proc power;
   pairedfreq dist=exact_cond
      discpropratio = 3
      totalpropdisc = 0.6
      npairs = 80
```

```
      power = .;
run;
```

You can also express effects in terms of the paired proportions and correlation:

```
proc power;
   pairedfreq dist=exact_cond
      pairedproportions = 0.6 | 0.8
      corr = 0.4
      npairs = 45
      power = .;
run;
```

You can also express effects in terms of the proportion difference, reference proportion, and correlation:

```
proc power;
   pairedfreq dist=exact_cond
      proportiondiff = 0.2
      refproportion = 0.6
      corr = 0.4
      npairs = 45
      power = .;
run;
```

You can also express effects in terms of the odds ratio, reference proportion, and correlation:

```
proc power;
   pairedfreq dist=exact_cond
      oddsratio = 2.66667
      refproportion = 0.6
      corr = 0.4
      npairs = 45
      power = .;
run;
```

You can also express effects in terms of the relative risk, reference proportion, and correlation:

```
proc power;
   pairedfreq dist=exact_cond
      relativerisk = 1.33333
      refproportion = 0.6
      corr = 0.4
      npairs = 45
      power = .;
run;
```

McNemar Normal Approximation Test

The following statements demonstrate a sample size computation for the normal-approximate McNemar test. The default value for the METHOD= option specifies an exact sample size computation. Default values for the TEST=, SIDES=, ALPHA=, and NULLDISCPROPRATIO= options specify a two-sided McNemar test for no effect with a significance level of 0.05.

```
proc power;
   pairedfreq dist=normal method=connor
      discproportions = 0.15 | 0.45
      npairs = .
      power = .9;
run;
```

PAIREDMEANS Statement

PAIREDMEANS < *options* > **;**

The PAIREDMEANS statement performs power and sample size analyses for t tests, equivalence tests, and confidence interval precision involving paired samples.

Summary of Options

Table 96.19 summarizes the *options* available in the PAIREDMEANS statement.

Table 96.21 PAIREDMEANS Statement Options

Option	Description
Define Analysis	
CI=	Specifies an analysis of precision of the confidence interval for the mean difference
DIST=	Specifies the underlying distribution assumed for the test statistic
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
LOWER=	Specifies the lower equivalence bound
NULLDIFF=	Specifies the null mean difference
NULLRATIO=	Specifies the null mean ratio
SIDES=	Specifies the number of sides and the direction of the statistical test or confidence interval
UPPER=	Specifies the upper equivalence bound
Specify Effects	
HALFWIDTH=	Specifies the desired confidence interval half-width
MEANDIFF=	Specifies the mean difference
MEANRATIO=	Specifies the geometric mean ratio, γ_2/γ_1
PAIREDMEANS=	Specifies the two paired means
Specify Variability	
CORR=	Specifies the correlation between members of a pair
CV=	Specifies the common coefficient of variation
PAIREDCVS=	Specifies the coefficient of variation for each member of a pair
PAIREDSTDDEVS=	Specifies the standard deviation of each member of a pair
STDDEV=	Specifies the common standard deviation

Table 96.21 *continued*

Option	Description
Specify Sample Size	
NFRACTIONAL	Enables fractional input and output for sample sizes
NPAIRS=	Specifies the number of pairs
Specify Power and Related Probabilities	
POWER=	Specifies the desired power of the test
PROBTYPE=	Specifies the type of probability for the PROBWIDTH= option
PROBWIDTH=	Specifies the probability of obtaining a confidence interval half-width less than or equal to the value specified by the HALFWIDTH=
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.22 summarizes the valid result parameters for different analyses in the PAIREDMEANS statement.

Table 96.22 Summary of Result Parameters in the PAIREDMEANS Statement

Analyses	Solve For	Syntax
TEST=DIFF	Power	POWER=.
	Sample size	NPAIRS=.
TEST=RATIO	Power	POWER=.
	Sample size	NPAIRS=.
TEST=EQUIV_DIFF	Power	POWER=.
	Sample size	NPAIRS=.
TEST=EQUIV_RATIO	Power	POWER=.
	Sample size	NPAIRS=.
CI=DIFF	Prob(width)	PROBWIDTH=.
	Sample size	NPAIRS=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual 0.05 × 100% = 5% level of significance. For the TEST=EQUIV_DIFF and TEST=EQUIV_RATIO analyses, a value of α corresponds to a $100(1 - 2\alpha)\%$ confidence interval, and the value must be less than 0.5. For all the other analyses, a value of α corresponds to a $100(1 - \alpha)\%$ confidence interval. If the CI= and SIDES=1 options are used, then the value must be less than 0.5. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CI

CI=DIFF

specifies an analysis of precision of the confidence interval for the mean difference. Instead of power, the relevant probability for this analysis is the probability of achieving a desired precision. Specifically, it is the probability that the half-width of the observed confidence interval will be at most the value specified by the HALFWIDTH= option. If neither the CI= option nor the TEST= option is used, the default is TEST=DIFF.

CORR=*number-list*

specifies the correlation between members of a pair. For tests that assume lognormal data (DIST=LOGNORMAL, or TEST=RATIO or TEST=EQUIV_RATIO), values of the CORR= option are restricted to the range (ρ_L, ρ_U), where

$$\rho_L = \frac{\exp\left(-\left[\log(\mathrm{CV}_1^2+1)\log(\mathrm{CV}_2^2+1)\right]^{\frac{1}{2}}\right)-1}{\mathrm{CV}_1\mathrm{CV}_2}$$

$$\rho_U = \frac{\exp\left(\left[\log(\mathrm{CV}_1^2+1)\log(\mathrm{CV}_2^2+1)\right]^{\frac{1}{2}}\right)-1}{\mathrm{CV}_1\mathrm{CV}_2}$$

and CV_1 are the CV_2 coefficient of variation values specified by the CV= or PAIREDCVS= option. See "Paired *t* Test for Mean Ratio with Lognormal Data (TEST=RATIO)" on page 8058 for more information about this restriction on correlation values. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CV=*number-list*

specifies the coefficient of variation that is assumed to be common to both members of a pair. The coefficient of variation is defined as the ratio of the standard deviation to the mean on the original data scale. You can use this option only with DIST=LOGNORMAL. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DIST=LOGNORMAL | NORMAL

specifies the underlying distribution assumed for the test statistic. NORMAL corresponds the normal distribution, and LOGNORMAL corresponds to the lognormal distribution. The default value (also the only acceptable value in each case) is NORMAL for TEST=DIFF, TEST=EQUIV_DIFF, and CI=DIFF; and LOGNORMAL for TEST=RATIO and TEST=EQUIV_RATIO.

HALFWIDTH=*number-list*

specifies the desired confidence interval half-width. The half-width is defined as the distance between the point estimate and a finite endpoint. This option can be used only with the CI=DIFF analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

LOWER=*number-list*

specifies the lower equivalence bound for the mean difference or mean ratio, in the original scale (whether DIST=NORMAL or DIST=LOGNORMAL). This option can be used only with the TEST=EQUIV_DIFF and TEST=EQUIV_RATIO analyses. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

MEANDIFF=*number-list*

specifies the mean difference, defined as the mean of the difference between the second and first members of a pair, $\mu_2 - \mu_1$. This option can be used only with the TEST=DIFF and TEST=EQUIV_DIFF analyses. When TEST=EQUIV_DIFF, the mean difference is interpreted as the treatment mean minus the reference mean. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

MEANRATIO=*number-list*

specifies the geometric mean ratio, defined as γ_2/γ_1. This option can be used only with the TEST=RATIO and TEST=EQUIV_RATIO analyses. When TEST=EQUIV_RATIO, the mean ratio is interpreted as the treatment mean divided by the reference mean. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NPAIRS=*number-list*

specifies the number of pairs or requests a solution for the number of pairs by specifying a missing value (NPAIRS=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLDIFF=*number-list*

NULLD=*number-list*

specifies the null mean difference. The default value is 0. This option can be used only with the TEST=DIFF analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLRATIO=*number-list*

NULLR=*number-list*

specifies the null mean ratio. The default value is 1. This option can be used only with the TEST=RATIO analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=
- NULLDIFF=
- NULLRATIO=
- LOWER=
- UPPER=
- ALPHA=
- PAIREDMEANS=
- MEANDIFF=

- MEANRATIO=
- HALFWIDTH=
- STDDEV=
- PAIREDSTDDEVS=
- CV=
- PAIREDCVS=
- CORR=
- NPAIRS=
- POWER=
- PROBTYPE=
- PROBWIDTH=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the PAIREDMEANS statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the PAIREDMEANS statement.

PAIREDCVS=*grouped-number-list*

specifies the coefficient of variation for each member of a pair. Unlike the CV= option, the PAIREDCVS= option supports different values for each member of a pair. The coefficient of variation is defined as the ratio of the standard deviation to the mean on the original data scale. Values must be nonnegative (unless both are equal to zero, which is permitted). This option can be used only with DIST=LOGNORMAL. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PAIREDMEANS=*grouped-number-list*

PMEANS=*grouped-number-list*

specifies the two paired means, in the original scale. The means are arithmetic if DIST=NORMAL and geometric if DIST=LOGNORMAL. This option cannot be used with the CI=DIFF analysis. When TEST=EQUIV_DIFF, the means are interpreted as the reference mean (first) and the treatment mean (second). For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PAIREDSTDDEVS=*grouped-number-list*

PAIREDSTDS=*grouped-number-list*

PSTDDEVS=*grouped-number-list*

PSTDS=*grouped-number-list*

specifies the standard deviation of each member of a pair. Unlike the STDDEV= option, the PAIREDSTDDEVS= option supports different values for each member of a pair. This option can be used only with DIST=NORMAL. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. This option cannot be used with the CI=DIFF analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PROBTYPE=*keyword-list*

specifies the type of probability for the PROBWIDTH= option. A value of CONDITIONAL (the default) indicates the conditional probability that the confidence interval half-width is at most the value specified by the HALFWIDTH= option, given that the true mean difference is captured by the confidence interval. A value of UNCONDITIONAL indicates the unconditional probability that the confidence interval half-width is at most the value specified by the HALFWIDTH= option. you can use the alias GIVENVALIDITY for CONDITIONAL. The PROBTYPE= option can be used only with the CI=DIFF analysis. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CONDITIONAL width probability conditional on interval containing the mean

UNCONDITIONAL unconditional width probability

PROBWIDTH=*number-list*

specifies the desired probability of obtaining a confidence interval half-width less than or equal to the value specified by the HALFWIDTH= option. A missing value (PROBWIDTH=.) requests a solution for this probability. The type of probability is controlled with the PROBTYPE= option. Values are expressed as probabilities (for example, 0.9) rather than percentages. This option can be used only with the CI=DIFF analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test or confidence interval. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. Valid *keywords* and their interpretation for the TEST= analyses are as follows:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating an effect greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating an effect less than the null value.

For confidence intervals, SIDES=U refers to an interval between the lower confidence limit and infinity, and SIDES=L refers to an interval between minus infinity and the upper confidence limit. For both of these cases and SIDES=1, the confidence interval computations are equivalent. You cannot use the SIDES= option with the TEST=EQUIV_DIFF and TEST=EQUIV_RATIO analyses. By default, SIDES=2.

STDDEV=*number-list*

STD=*number-list*

specifies the standard deviation assumed to be common to both members of a pair. This option can be used only with DIST=NORMAL. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TEST=DIFF | EQUIV_DIFF | EQUIV_RATIO | RATIO

TEST

specifies the statistical analysis. TEST or TEST=DIFF (the default) specifies a paired *t* test on the mean difference. TEST=EQUIV_DIFF specifies an additive equivalence test of the mean difference by using a two one-sided tests (TOST) analysis (Schuirmann 1987). TEST=EQUIV_RATIO specifies a multiplicative equivalence test of the mean ratio by using a TOST analysis. TEST=RATIO specifies a paired *t* test on the geometric mean ratio. If neither the TEST= option nor the CI= option is used, the default is TEST=DIFF.

UPPER=*number-list*

specifies the upper equivalence bound for the mean difference or mean ratio, in the original scale (whether DIST=NORMAL or DIST=LOGNORMAL). This option can be used only with the TEST=EQUIV_DIFF and TEST=EQUIV_RATIO analyses. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

Restrictions on Option Combinations

To define the analysis, choose one of the following parameterizations:

- a statistical test (by using the TEST= option)
- confidence interval precision (by using the CI= option)

To specify the means, choose one of the following parameterizations:

- individual means (by using the PAIREDMEANS= option)
- mean difference (by using the MEANDIFF= option)
- mean ratio (by using the MEANRATIO= option)

To specify the coefficient of variation, choose one of the following parameterizations:

- common coefficient of variation (by using the CV= option)
- individual coefficients of variation (by using the PAIREDCVS= option)

To specify the standard deviation, choose one of the following parameterizations:

- common standard deviation (by using the STDDEV= option)
- individual standard deviations (by using the PAIREDSTDDEVS= option)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the PAIREDMEANS statement.

Paired t Test

You can express effects in terms of the mean difference and variability in terms of a correlation and common standard deviation, as in the following statements. Default values for the DIST=, SIDES=, NULLDIFF=, and ALPHA= options specify a two-sided test for no difference with a normal distribution and a significance level of 0.05.

```
proc power;
   pairedmeans test=diff
      meandiff = 7
      corr = 0.4
      stddev = 12
      npairs = 50
      power = .;
run;
```

You can also express effects in terms of individual means and variability in terms of correlation and individual standard deviations:

```
proc power;
   pairedmeans test=diff
      pairedmeans = 8 | 15
      corr = 0.4
      pairedstddevs = (7 12)
      npairs = .
      power = 0.9;
run;
```

Paired t Test of Mean Ratio with Lognormal Data

You can express variability in terms of correlation and a common coefficient of variation, as in the following statements. Defaults for the DIST=, SIDES=, NULLRATIO= and ALPHA= options specify a two-sided test of mean ratio = 1 assuming a lognormal distribution and a significance level of 0.05.

```
proc power;
   pairedmeans test=ratio
      meanratio = 7
      corr = 0.3
      cv = 1.2
      npairs = 30
      power = .;
run;
```

You can also express variability in terms of correlation and individual coefficients of variation:

```
proc power;
   pairedmeans test=ratio
      meanratio = 7
      corr = 0.3
      pairedcvs = 0.8 | 0.9
```

```
      npairs = 30
      power = .;
run;
```

Additive Equivalence Test for Mean Difference with Normal Data

The following statements demonstrate a sample size computation for a TOST equivalence test for a normal mean difference. Default values for the DIST= and ALPHA= options specify a normal distribution and a significance level of 0.05.

```
proc power;
   pairedmeans test=equiv_diff
      lower = 2
      upper = 5
      meandiff = 4
      corr = 0.2
      stddev = 8
      npairs = .
      power = 0.9;
run;
```

Multiplicative Equivalence Test for Mean Ratio with Lognormal Data

The following statements demonstrate a power computation for a TOST equivalence test for a lognormal mean ratio. Default values for the DIST= and ALPHA= options specify a lognormal distribution and a significance level of 0.05.

```
proc power;
   pairedmeans test=equiv_ratio
      lower = 3
      upper = 7
      meanratio = 5
      corr = 0.2
      cv = 1.1
      npairs = 50
      power = .;
run;
```

Confidence Interval for Mean Difference

By default CI=DIFF analyzes the conditional probability of obtaining the desired precision, given that the interval contains the true mean difference, as in the following statements. The defaults of SIDES=2 and ALPHA=0.05 specify a two-sided interval with a confidence level of 0.95.

```
proc power;
   pairedmeans ci = diff
      halfwidth = 4
      corr = 0.35
      stddev = 8
      npairs = 30
      probwidth = .;
run;
```

PLOT Statement

PLOT < *plot-options* > < / *graph-options* > **;**

The PLOT statement produces a graph or set of graphs for the sample size analysis defined by the previous analysis statement. The *plot-options* define the plot characteristics, and the *graph-options* are SAS/GRAPH-style options. If ODS Graphics is enabled, then the PLOT statement uses ODS Graphics to create graphs. For example:

```
ods graphics on;

proc power;
   onesamplemeans
      mean   = 5 10
      ntotal = 150
      stddev =  30 50
      power  = .;
   plot x=n min=100 max=200;
run;

ods graphics off;
```

Otherwise, traditional graphics are produced. For example:

```
ods graphics off;

proc power;
   onesamplemeans
      mean   = 5 10
      ntotal = 150
      stddev =  30 50
      power  = .;
   plot x=n min=100 max=200;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

Options

You can specify the following *plot-options* in the PLOT statement.

INTERPOL=JOIN | NONE

specifies the type of curve to draw through the computed points. The INTERPOL=JOIN option connects computed points by straight lines. The INTERPOL=NONE option leaves computed points unconnected.

KEY=BYCURVE < **(** *bycurve-options* **)** >

KEY=BYFEATURE < **(** *byfeature-options* **)** >

KEY=ONCURVES

specifies the style of key (or "legend") for the plot. The default is KEY=BYFEATURE, which specifies a key with a column of entries for each plot feature (line style, color, and/or symbol). Each entry shows

the mapping between a value of the feature and the value(s) of the analysis parameter(s) linked to that feature. The KEY=BYCURVE option specifies a key with each row identifying a distinct curve in the plot. The KEY=ONCURVES option places a curve-specific label adjacent to each curve.

You can specify the following *bycurve-options* in parentheses after the KEY=BYCURVE option.

NUMBERS=OFF | ON
specifies how the key should identify curves. If NUMBERS=OFF, then the key includes symbol, color, and line style samples to identify the curves. If NUMBERS=ON, then the key includes numbers matching numeric labels placed adjacent to the curves. The default is NUMBERS=ON.

POS=BOTTOM | INSET
specifies the position of the key. The POS=BOTTOM option places the key below the X axis. The POS=INSET option places the key inside the plotting region and attempts to choose the least crowded corner. The default is POS=BOTTOM.

You can specify the following *byfeature-options* in parentheses after KEY=BYFEATURE option.

POS=BOTTOM | INSET
specifies the position of the key. The POS=BOTTOM option places the key below the X axis. The POS=INSET option places the key inside the plotting region and attempts to choose the least crowded corner. The default is POS=BOTTOM.

MARKERS=ANALYSIS | COMPUTED | NICE | NONE
specifies the locations for plotting symbols.

The MARKERS=ANALYSIS option places plotting symbols at locations that correspond to the values of the relevant input parameter from the analysis statement preceding the PLOT statement.

The MARKERS=COMPUTED option (the default) places plotting symbols at the locations of actual computed points from the sample size analysis.

The MARKERS=NICE option places plotting symbols at tick mark locations (corresponding to the argument axis).

The MARKERS=NONE option disables plotting symbols.

MAX=*number* | DATAMAX
specifies the maximum of the range of values for the parameter associated with the “argument” axis (the axis that is *not* representing the parameter being solved for). The default is DATAMAX, which specifies the maximum value that occurs for this parameter in the analysis statement that precedes the PLOT statement.

MIN=*number* | DATAMIN
specifies the minimum of the range of values for the parameter associated with the “argument” axis (the axis that is *not* representing the parameter being solved for). The default is DATAMIN, which specifies the minimum value that occurs for this parameter in the analysis statement that precedes the PLOT statement.

NPOINTS=*number*

NPTS=*number*

specifies the number of values for the parameter associated with the "argument" axis (the axis that is *not* representing the parameter being solved for). You cannot use the NPOINTS= and STEP= options simultaneously. The default value for typical situations is 20.

STEP=*number*

specifies the increment between values of the parameter associated with the "argument" axis (the axis that is *not* representing the parameter being solved for). You cannot use the STEP= and NPOINTS= options simultaneously. By default, the NPOINTS= option is used instead of the STEP= option.

VARY (*feature* **< BY** *parameter-list* **> <, . . . ,** *feature* **< BY** *parameter-list* **> >)**

specifies how plot features should be linked to varying analysis parameters. Available plot *features* are COLOR, LINESTYLE, PANEL, and SYMBOL. A "panel" refers to a separate plot with a heading identifying the subset of values represented in the plot.

The *parameter-list* is a list of one or more names separated by spaces. Each name must match the name of an analysis option used in the analysis statement preceding the PLOT statement. Also, the name must be the *primary* name for the analysis option—that is, the one listed first in the syntax description.

If you omit the < BY *parameter-list* > portion for a feature, then one or more multivalued parameters from the analysis will be automatically selected for you.

X=EFFECT | N | POWER

specifies a plot with the requested type of parameter on the X axis and the parameter being solved for on the Y axis. When X=EFFECT, the parameter assigned to the X axis is the one most representative of "effect size." When X=N, the parameter assigned to the X axis is the sample size. When X=POWER, the parameter assigned to the X axis is the one most representative of "power" (either power itself or a similar probability, such as Prob(Width) for confidence interval analyses). You cannot use the X= and Y= options simultaneously. The default is X=POWER, unless the result parameter is power or Prob(Width), in which case the default is X=N.

You can use the X=N option only when a scalar sample size parameter is used as input in the analysis. For example, X=N can be used with total sample size or sample size per group, or with two group sample sizes when one is being solved for.

Table 96.23 summarizes the parameters that represent effect size in different analyses.

Table 96.23 Effect Size Parameters for Different Analyses

Analysis Statement and Options	Effect Size Parameters
COXREG	Hazard ratio
CUSTOM	Primary noncentrality or correlation
LOGISTIC	None
MULTREG	Partial correlation or R^2 difference
ONECORR	Correlation

Table 96.23 *continued*

Analysis Statement and Options	Effect Size Parameters
ONESAMPLEFREQ TEST	Proportion
ONESAMPLEFREQ CI	CI half-width
ONESAMPLEMEANS TEST=T, ONESAMPLEMEANS TEST=EQUIV	Mean
ONESAMPLEMEANS CI=T	CI half-width
ONEWAYANOVA	None
PAIREDFREQ	Discordant proportion difference or ratio
PAIREDMEANS TEST=DIFF, PAIREDMEANS TEST=EQUIV_DIFF	Mean difference
PAIREDMEANS TEST=RATIO, PAIREDMEANS TEST=EQUIV_RATIO	Mean ratio
PAIREDMEANS CI=DIFF	CI half-width
TWOSAMPLEFREQ	Proportion difference, odds ratio, or relative risk
TWOSAMPLEMEANS TEST=DIFF, TWOSAMPLEMEANS TEST=DIFF_SATT, TWOSAMPLEMEANS TEST=EQUIV_DIFF	Mean difference
TWOSAMPLEMEANS TEST=RATIO, TWOSAMPLEMEANS TEST=EQUIV_RATIO	Mean ratio
TWOSAMPLEMEANS CI=DIFF	CI half-width
TWOSAMPLESURVIVAL	Hazard ratio if used, else none
TWOSAMPLEWILCOXON	None

XOPTS=(*x-options* **)**

specifies plot characteristics pertaining to the X axis.

You can specify the following *x-options* in parentheses.

CROSSREF=NO | YES

specifies whether the reference lines defined by the REF= *x-option* should be crossed with a reference line on the Y axis that indicates the solution point on the curve.

REF=*number-list*

specifies locations for reference lines extending from the X axis across the entire plotting region. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

Y=EFFECT | N | POWER

specifies a plot with the requested type of parameter on the Y axis and the parameter being solved for on the X axis. When Y=EFFECT, the parameter assigned to the Y axis is the one most representative of "effect size." When Y=N, the parameter assigned to the Y axis is the sample size. When Y=POWER, the parameter assigned to the Y axis is the one most representative of "power" (either power itself or a similar probability, such as Prob(Width) for confidence interval analyses). You cannot use the Y= and X= options simultaneously. By default, the X= option is used instead of the Y= option.

YOPTS=(*y-options* **)**

specifies plot characteristics pertaining to the Y axis.

You can specify the following *y-options* in parentheses.

CROSSREF=NO | YES

specifies whether the reference lines defined by the REF= *y-option* should be crossed with a reference line on the X axis that indicates the solution point on the curve.

REF=*number-list*

specifies locations for reference lines extending from the Y axis across the entire plotting region. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

You can specify the following *graph-options* in the PLOT statement after a slash (/).

DESCRIPTION='*string* **'**

specifies a descriptive string of up to 40 characters that appears in the "Description" field of the graphics catalog. The description does not appear on the plots. By default, PROC POWER assigns a description either of the form "*Y* versus *X*" (for a single-panel plot) or of the form "*Y* versus *X* (*S*)," where *Y* is the parameter on the Y axis, *X* is the parameter on the X axis, and *S* is a description of the subset represented on the current panel of a multipanel plot.

NAME='*string* **'**

specifies a name of up to eight characters for the catalog entry for the plot. The default name is PLOT*n*, where *n* is the number of the plot statement within the current invocation of PROC POWER. If the name duplicates the name of an existing entry, SAS/GRAPH software adds a number to the duplicate name to create a unique entry—for example, PLOT11 and PLOT12 for the second and third panels of a multipanel plot generated in the first PLOT statement in an invocation of PROC POWER.

TWOSAMPLEFREQ Statement

TWOSAMPLEFREQ < *options* > **;**

The TWOSAMPLEFREQ statement performs power and sample size analyses for tests of two independent proportions. The Farrington-Manning score, Pearson's chi-square, Fisher's exact, and likelihood ratio chi-square tests are supported.

Summary of Options

Table 96.24 summarizes the *options* available in the TWOSAMPLEFREQ statement.

Table 96.24 TWOSAMPLEFREQ Statement Options

Option	Description
Define Analysis	
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
NULLODDSRATIO=	Specifies the null odds ratio
NULLPROPORTIONDIFF=	Specifies the null proportion difference
NULLRELATIVERISK=	Specifies the null relative risk
SIDES=	Specifies the number of sides and the direction of the statistical test or confidence interval
Specify Effects	
GROUPPROPORTIONS=	Specifies the two independent proportions, p_1 and p_2
ODDSRATIO=	Specifies the odds ratio $[p_2/(1-p_2)]\,/\,[p_1/(1-p_1)]$
PROPORTIONDIFF=	Specifies the proportion difference $p_2 - p_1$
REFPROPORTION=	Specifies the reference proportion p_1
RELATIVERISK=	Specifies the relative risk p_2/p_1
Specify Sample Size and Allocation	
GROUPNS=	Specifies the two group sample sizes
GROUPWEIGHTS=	Specifies the sample size allocation weights for the two groups
NFRACTIONAL	Enables fractional input and output for sample sizes
NPERGROUP=	Specifies the common sample size per group
NTOTAL=	Specifies the sample size
Specify Power	
POWER=	Specifies the desired power of the test
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.25 summarizes the valid result parameters for different analyses in the TWOSAMPLEFREQ statement.

Table 96.25 Summary of Result Parameters in the TWOSAMPLEFREQ Statement

Analyses	Solve For	Syntax
TEST=FISHER	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.
TEST=FM	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.
TEST=FM_RR	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.
TEST=LRCHI	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.
TEST=PCHI	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPPROPORTIONS=*grouped-number-list*

GPROPORTIONS=*grouped-number-list*

GROUPPS=*grouped-number-list*

GPS=*grouped-number-list*

specifies the two independent proportions, p_1 and p_2. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPNS=*grouped-number-list*

GNS=*grouped-number-list*

specifies the two group sample sizes. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPWEIGHTS=*grouped-number-list*

GWEIGHTS=*grouped-number-list*

specifies the sample size allocation weights for the two groups. This option controls how the total sample size is divided between the two groups. Each pair of values for the two groups represents relative allocation weights. Additionally, if the NFRACTIONAL option is not used, the total sample size is restricted to be equal to a multiple of the sum of the two group weights (so that the resulting design has an integer sample size for each group while adhering exactly to the group allocation

weights). Values must be integers unless the NFRACTIONAL option is used. The default value is (1 1), a balanced design with a weight of 1 for each group. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NPERGROUP=*number-list*

NPERG=*number-list*

specifies the common sample size per group or requests a solution for the common sample size per group by specifying a missing value (NPERGROUP=.). Use of this option implicitly specifies a balanced design. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLODDSRATIO=*number-list*

NULLOR=*number-list*

specifies the null odds ratio. You can specify this option only if you also specify the ODDSRATIO= and TEST=PCHI options. The NULLODDSRATIO= option is inconsistent with TEST=PCHI, which tests the proportion difference rather than the odds ratio, and its value is converted internally to a NULLPROPORTIONDIFF value by fixing the reference proportion. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. By default, NULLOR=1.

NULLPROPORTIONDIFF=*number-list*

NULLPDIFF=*number-list*

specifies the null proportion difference. You can specify this option only if you also specify the GROUPPROPORTIONS= or PROPORTIONDIFF= option and the TEST=FM or TEST=PCHI option. If you are using a nondefault null value, then TEST=FM is recommended. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. By default, NULLPDIFF=0.

NULLRELATIVERISK=*number-list*

NULLRR=*number-list*

specifies the null relative risk. You can specify this option only if you also specify the GROUPPROPORTIONS= or RELATIVERISK= option and the TEST=FM_RR or TEST=PCHI option. If you are using a nondefault null value, then TEST=FM_RR is recommended. The NULLRELATIVERISK= option is inconsistent with TEST=PCHI, which tests the proportion difference rather than the relative risk, and its value is converted internally to a NULLPROPORTIONDIFF value by fixing the reference proportion. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. By default, NULLRR=1.

ODDSRATIO=*number-list*

OR=*number-list*

specifies the odds ratio $[p_2/(1-p_2)] \,/\, [p_1/(1-p_1)]$. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=
- NULLPROPORTIONDIFF=
- NULLODDSRATIO=
- NULLRELATIVERISK=
- ALPHA=
- GROUPPROPORTIONS=
- REFPROPORTION=
- PROPORTIONDIFF=
- ODDSRATIO=
- RELATIVERISK=
- GROUPWEIGHTS=
- NTOTAL=
- NPERGROUP=
- GROUPNS=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the TWOSAMPLEFREQ statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the TWOSAMPLEFREQ statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PROPORTIONDIFF=*number-list*

PDIFF=*number-list*

specifies the proportion difference $p_2 - p_1$. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

REFPROPORTION=*number-list*

REFP=*number-list*

specifies the reference proportion p_1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

RELATIVERISK=*number-list*
RR=*number-list*

specifies the relative risk p_2/p_1. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test or confidence interval. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. You can specify the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating an effect greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating an effect less than the null value.

If the effect size is zero, then SIDES=1 is not permitted; instead, specify the direction of the test explicitly in this case with either SIDES=L or SIDES=U. By default, SIDES=2.

TEST=FISHER | FM | FM_RR | LRCHI | PCHI

specifies the statistical analysis. You can specify the following values:

FISHER specifies Fisher's exact test.

FM specifies the score test of Farrington and Manning (1990) for proportion difference.

FM_RR specifies the score test of Farrington and Manning (1990) for relative risk.

LRCHI specifies the likelihood ratio chi-square test.

PCHI specifies Pearson's chi-square test for proportion difference.

If you are using a nondefault null value for a noninferiority or superiority test, then TEST=FM or TEST=FM_RR is the most appropriate choice. In the absence of any nondefault null values, the default is TEST=PCHI. If you specify at least one nonzero null difference by using the NULLPROPORTIONDIFF= option, then the default is TEST=FM. If you specify at least one null relative risk not equal to 1 by using the NULLRELATIVERISK= option, then the default is TEST=FM_RR. For information about the power and sample size computational methods and formulas, see the section "Analyses in the TWOSAMPLEFREQ Statement" on page 8063.

Restrictions on Option Combinations

To specify the proportions, choose one of the following parameterizations:

- individual proportions (by using the GROUPPROPORTIONS= option)
- difference between proportions and reference proportion (by using the PROPORTIONDIFF= and REFPROPORTION= options)
- odds ratio and reference proportion (by using the ODDSRATIO= and REFPROPORTION= options)

- relative risk and reference proportion (by using the RELATIVERISK= and REFPROPORTION= options)

To specify the sample size and allocation, choose one of the following parameterizations:

- sample size per group in a balanced design (by using the NPERGROUP= option)
- total sample size and allocation weights (by using the NTOTAL= and GROUPWEIGHTS= options)
- individual group sample sizes (by using the GROUPNS= option)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the TWOSAMPLEFREQ statement.

Pearson Chi-Square Test for Two Proportions

You can use the NPERGROUP= option in a balanced design and express effects in terms of the individual proportions, as in the following statements. Default values for the SIDES= and ALPHA= options specify a two-sided test with a significance level of 0.05.

```
proc power;
   twosamplefreq test=pchi
      groupproportions = (.15 .25)
      npergroup = 50
      power = .;
run;
```

You can also specify an unbalanced design by using the NTOTAL= and GROUPWEIGHTS= options and express effects in terms of the odds ratio. The default value of the NULLODDSRATIO= option specifies a test of no effect.

```
proc power;
   twosamplefreq test=pchi
      oddsratio = 2.5
      refproportion = 0.3
      groupweights = (1 2)
      ntotal = .
      power = 0.8;
run;
```

You can also specify sample sizes with the GROUPNS= option and express effects in terms of relative risks. The default value of the NULLRELATIVERISK= option specifies a test of no effect.

```
proc power;
   twosamplefreq test=pchi
      relativerisk = 1.5
      refproportion = 0.2
      groupns = 40 | 60
      power = .;
run;
```

You can also express effects in terms of the proportion difference. The default value of the NULLPROPORTIONDIFF= option specifies a test of no effect, and the default value of the GROUPWEIGHTS= option specifies a balanced design.

```
proc power;
   twosamplefreq test=pchi
      proportiondiff = 0.15
      refproportion = 0.4
      ntotal = 100
      power = .;
run;
```

Farrington-Manning Score Test for Proportion Difference

The following statements demonstrate a sample size computation for the Farrington-Manning score test for the difference of two independent proportions:

```
proc power;
   twosamplefreq test=fm
      proportiondiff = 0.06
      refproportion = 0.32
      nullproportiondiff = -0.02
      sides = u
      ntotal = .
      power = 0.85;
run;
```

Farrington-Manning Score Test for Relative Risk

The following statements demonstrate a sample size computation for the Farrington-Manning score test for the relative risk of two independent proportions:

```
proc power;
   twosamplefreq test=fm_rr
      relativerisk = 1.1
      refproportion = 0.32
      nullrelativerisk = 0.95
      sides = u
      ntotal = .
      power = 0.9;
run;
```

Fisher's Exact Conditional Test for Two Proportions

The following statements demonstrate a power computation for Fisher's exact conditional test for two proportions. Default values for the SIDES= and ALPHA= options specify a two-sided test with a significance level of 0.05.

```
proc power;
   twosamplefreq test=fisher
      groupproportions = (.35 .15)
      npergroup = 50
      power = .;
run;
```

Likelihood Ratio Chi-Square Test for Two Proportions

The following statements demonstrate a sample size computation for the likelihood ratio chi-square test for two proportions. Default values for the SIDES= and ALPHA= options specify a two-sided test with a significance level of 0.05.

```
proc power;
   twosamplefreq test=lrchi
      oddsratio = 2
      refproportion = 0.4
      npergroup = .
      power = 0.9;
run;
```

TWOSAMPLEMEANS Statement

TWOSAMPLEMEANS < *options* > **;**

The TWOSAMPLEMEANS statement performs power and sample size analyses for pooled and unpooled *t* tests, equivalence tests, and confidence interval precision involving two independent samples.

Summary of Options

Table 96.26 summarizes the *options* available in the TWOSAMPLEMEANS statement.

Table 96.26 TWOSAMPLEMEANS Statement Options

Option	Description
Define Analysis	
CI=	Specifies an analysis of precision of the confidence interval
DIST=	Specifies the underlying distribution assumed for the test statistic
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
LOWER=	Specifies the lower equivalence bound
NULLDIFF=	Specifies the null mean difference
NULLRATIO=	Specifies the null mean ratio
SIDES=	Specifies the number of sides and the direction of the statistical test or confidence interval
UPPER=	Specifies the upper equivalence bound
Specify Effects	
HALFWIDTH=	Specifies the desired confidence interval half-width
GROUPMEANS=	Specifies the two group means
MEANDIFF=	Specifies the mean difference
MEANRATIO=	Specifies the geometric mean ratio, γ_2/γ_1
Specify Variability	
CV=	Specifies the common coefficient of variation
GROUPSTDDEVS=	Specifies the standard deviation of each group

Table 96.26 *continued*

Option	Description
STDDEV=	Specifies the common standard deviation
Specify Sample Size and Allocation	
GROUPNS=	Specifies the two group sample sizes
GROUPWEIGHTS=	Specifies the sample size allocation weights for the two groups
NFRACTIONAL	Enables fractional input and output for sample sizes
NPERGROUP=	Specifies the common sample size per group
NTOTAL=	Specifies the sample size
Specify Power and Related Probabilities	
POWER=	Specifies the desired power of the test
PROBTYPE=	Specifies the type of probability for the PROBWIDTH= option
PROBWIDTH=	Specifies the desired probability of obtaining a confidence interval half-width less than or equal to the value specified
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.27 summarizes the valid result parameters for different analyses in the TWOSAMPLEMEANS statement.

Table 96.27 Summary of Result Parameters in the TWOSAMPLEMEANS Statement

Analyses	Solve For	Syntax
TEST=DIFF	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.
	Group sample size	GROUPNS= *n1* | .
		GROUPNS=. | *n2*
		GROUPNS= (*n1* .)
		GROUPNS= (. *n2*)
	Group weight	GROUPWEIGHTS= *w1* | .
		GROUPWEIGHTS=. | *w2*
		GROUPWEIGHTS= (*w1* .)
		GROUPWEIGHTS= (. *w2*)
	Alpha	ALPHA=.
	Group mean	GROUPMEANS= *mean1* | .
		GROUPMEANS=. | *mean2*
		GROUPMEANS= (*mean1* .)
		GROUPMEANS= (. *mean2*)
	Mean difference	MEANDIFF=.
	Standard deviation	STDDEV=.

Table 96.27 *continued*

Analyses	Solve For	Syntax
TEST=DIFF_SATT	Power Sample size	POWER=. NTOTAL=. NPERGROUP=.
TEST=RATIO	Power Sample size	POWER=. NTOTAL=. NPERGROUP=.
TEST=EQUIV_DIFF	Power Sample size	POWER=. NTOTAL=. NPERGROUP=.
TEST=EQUIV_RATIO	Power Sample size	POWER=. NTOTAL=. NPERGROUP=.
CI=DIFF	Prob(width) Sample size	PROBWIDTH=. NTOTAL=. NPERGROUP=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test or requests a solution for alpha by specifying a missing value (ALPHA=.). The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For the TEST=EQUIV_DIFF and TEST=EQUIV_RATIO analyses, a value of α corresponds to a $100(1 - 2\,\alpha)\%$ confidence interval, and the value must be less than 0.5. For all the other analyses, a value of α corresponds to a $100(1 - \alpha)\%$ confidence interval. If the CI= and SIDES=1 options are used, then the value must be less than 0.5. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CI

CI=DIFF

specifies an analysis of precision of the confidence interval for the mean difference, assuming equal variances. Instead of power, the relevant probability for this analysis is the probability that the interval half-width is at most the value specified by the HALFWIDTH= option. If neither the TEST= option nor the CI= option is used, the default is TEST=DIFF.

CV=*number-list*

specifies the coefficient of variation assumed to be common to both groups. The coefficient of variation is defined as the ratio of the standard deviation to the mean on the original data scale. You can use this option only with DIST=LOGNORMAL. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

DIST=LOGNORMAL | NORMAL

specifies the underlying distribution assumed for the test statistic. NORMAL corresponds the normal distribution, and LOGNORMAL corresponds to the lognormal distribution. The default value (also the only acceptable value in each case) is NORMAL for TEST=DIFF, TEST=DIFF_SATT, TEST=EQUIV_DIFF, and CI=DIFF; and LOGNORMAL for TEST=RATIO and TEST=EQUIV_RATIO.

GROUPMEANS=*grouped-number-list*

GMEANS=*grouped-number-list*

specifies the two group means or requests a solution for one group mean given the other. Means are in the original scale. They are arithmetic if DIST=NORMAL and geometric if DIST=LOGNORMAL. This option cannot be used with the CI=DIFF analysis. When TEST=EQUIV_DIFF, the means are interpreted as the reference mean (first) and the treatment mean (second). For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPNS=*grouped-number-list*

GNS=*grouped-number-list*

specifies the two group sample sizes or requests a solution for one group sample size given the other. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPSTDDEVS=*grouped-number-list*

GSTDDEVS=*grouped-number-list*

GROUPSTDS=*grouped-number-list*

GSTDS=*grouped-number-list*

specifies the standard deviation of each group. Unlike the STDDEV= option, the GROUPSTDDEVS== option supports different values for each group. It is valid only for the Satterthwaite *t* test (TEST=DIFF_SATT DIST=NORMAL). For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPWEIGHTS=*grouped-number-list*

GWEIGHTS=*grouped-number-list*

specifies the sample size allocation weights for the two groups, or requests a solution for one group weight given the other. This option controls how the total sample size is divided between the two groups. Each pair of values for the two groups represents relative allocation weights. Additionally, if the NFRACTIONAL option is not used, the total sample size is restricted to be equal to a multiple of the sum of the two group weights (so that the resulting design has an integer sample size for each group while adhering exactly to the group allocation weights). Values must be integers unless the NFRACTIONAL option is used. The default value is (1 1), a balanced design with a weight of 1 for each group. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

HALFWIDTH=*number-list*

specifies the desired confidence interval half-width. The half-width is defined as the distance between the point estimate and a finite endpoint. This option can be used only with the CI=DIFF analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

LOWER=*number-list*

specifies the lower equivalence bound for the mean difference or mean ratio, in the original scale (whether DIST=NORMAL or DIST=LOGNORMAL). Values must be greater than 0 when DIST=LOGNORMAL. This option can be used only with the TEST=EQUIV_DIFF and TEST=EQUIV_RATIO analyses. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

MEANDIFF=*number-list*

specifies the mean difference, defined as $\mu_2 - \mu_1$, or requests a solution for the mean difference by specifying a missing value (MEANDIFF=.). This option can be used only with the TEST=DIFF, TEST=DIFF_SATT, and TEST=EQUIV_DIFF analyses. When TEST=EQUIV_DIFF, the mean difference is interpreted as the treatment mean minus the reference mean. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

MEANRATIO=*number-list*

specifies the geometric mean ratio, defined as γ_2/γ_1. This option can be used only with the TEST=RATIO and TEST=EQUIV_RATIO analyses. When TEST=EQUIV_RATIO, the mean ratio is interpreted as the treatment mean divided by the reference mean. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NPERGROUP=*number-list*

NPERG=*number-list*

specifies the common sample size per group or requests a solution for the common sample size per group by specifying a missing value (NPERGROUP=.). Use of this option implicitly specifies a balanced design. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLDIFF=*number-list*

NULLD=*number-list*

specifies the null mean difference. The default value is 0. This option can be used only with the TEST=DIFF and TEST=DIFF_SATT analyses. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NULLRATIO=*number-list*

NULLR=*number-list*

specifies the null mean ratio. The default value is 1. This option can be used only with the TEST=RATIO analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=
- NULLDIFF=
- NULLRATIO=
- LOWER=
- UPPER=
- ALPHA=
- GROUPMEANS=
- MEANDIFF=
- MEANRATIO=
- HALFWIDTH=
- STDDEV=
- GROUPSTDDEVS==
- CV=
- GROUPWEIGHTS=
- NTOTAL=
- NPERGROUP=
- GROUPNS=
- POWER=
- PROBTYPE=
- PROBWIDTH=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the TWOSAMPLEMEANS statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the TWOSAMPLEMEANS statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. This option cannot be used with the CI=DIFF analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

PROBTYPE=*keyword-list*

specifies the type of probability for the PROBWIDTH= option. A value of CONDITIONAL (the default) indicates the conditional probability that the confidence interval half-width is at most the value specified by the HALFWIDTH= option, given that the true mean difference is captured by the confidence interval. A value of UNCONDITIONAL indicates the unconditional probability that the confidence interval half-width is at most the value specified by the HALFWIDTH= option. you can

use the alias GIVENVALIDITY for CONDITIONAL. The PROBTYPE= option can be used only with the CI=DIFF analysis. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CONDITIONAL width probability conditional on interval containing the mean

UNCONDITIONAL unconditional width probability

PROBWIDTH=*number-list*

specifies the desired probability of obtaining a confidence interval half-width less than or equal to the value specified by the HALFWIDTH= option. A missing value (PROBWIDTH=.) requests a solution for this probability. The type of probability is controlled with the PROBTYPE= option. Values are expressed as probabilities (for example, 0.9) rather than percentages. This option can be used only with the CI=DIFF analysis. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test or confidence interval. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. Valid *keywords* and their interpretation for the TEST= analyses are as follows:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating an effect greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating an effect less than the null value.

For confidence intervals, SIDES=U refers to an interval between the lower confidence limit and infinity, and SIDES=L refers to an interval between minus infinity and the upper confidence limit. For both of these cases and SIDES=1, the confidence interval computations are equivalent. You cannot use the SIDES= option with the TEST=EQUIV_DIFF and TEST=EQUIV_RATIO analyses. By default, SIDES=2.

STDDEV=*number-list*

STD=*number-list*

specifies the standard deviation assumed to be common to both groups, or requests a solution for the common standard deviation with a missing value (STDDEV=.). This option can be used only with DIST=NORMAL. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TEST=DIFF | DIFF_SATT | EQUIV_DIFF | EQUIV_RATIO | RATIO

TEST

specifies the statistical analysis. TEST or TEST=DIFF (the default) specifies a pooled t test on the mean difference, assuming equal variances. TEST=DIFF_SATT specifies a Satterthwaite unpooled t test on the mean difference, assuming unequal variances. TEST=EQUIV_DIFF specifies an additive equivalence test of the mean difference by using a two one-sided tests (TOST) analysis (Schuirmann

1987). TEST=EQUIV_RATIO specifies a multiplicative equivalence test of the mean ratio by using a TOST analysis. TEST=RATIO specifies a pooled *t* test on the mean ratio, assuming equal coefficients of variation. If neither the TEST= option nor the CI= option is used, the default is TEST=DIFF.

UPPER=*number-list*
specifies the upper equivalence bound for the mean difference or mean ratio, in the original scale (whether DIST=NORMAL or DIST=LOGNORMAL). This option can be used only with the TEST=EQUIV_DIFF and TEST=EQUIV_RATIO analyses. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

Restrictions on Option Combinations

To define the analysis, choose one of the following parameterizations:

- a statistical test (by using the TEST= option)
- confidence interval precision (by using the CI= option)

To specify the means, choose one of the following parameterizations:

- individual group means (by using the GROUPMEANS= option)
- mean difference (by using the MEANDIFF= option)
- mean ratio (by using the MEANRATIO= option)

To specify standard deviations in the Satterthwaite *t* test (TEST=DIFF_SATT), choose one of the following parameterizations:

- common standard deviation (by using the STDDEV= option)
- individual group standard deviations (by using the GROUPSTDDEVS== option)

To specify the sample sizes and allocation, choose one of the following parameterizations:

- sample size per group in a balanced design (by using the NPERGROUP= option)
- total sample size and allocation weights (by using the NTOTAL= and GROUPWEIGHTS= options)
- individual group sample sizes (by using the GROUPNS= option)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the TWOSAMPLEMEANS statement.

Two-Sample *t* Test Assuming Equal Variances

You can use the NPERGROUP= option in a balanced design and express effects in terms of the mean difference, as in the following statements. Default values for the DIST=, SIDES=, NULLDIFF=, and ALPHA= options specify a two-sided test for no difference with a normal distribution and a significance level of 0.05.

```
proc power;
   twosamplemeans test=diff
      meandiff = 7
      stddev = 12
      npergroup = 50
      power = .;
run;
```

You can also specify an unbalanced design by using the NTOTAL= and GROUPWEIGHTS= options and express effects in terms of individual group means:

```
proc power;
   twosamplemeans test=diff
      groupmeans = 8 | 15
      stddev = 4
      groupweights = (2 3)
      ntotal = .
      power = 0.9;
run;
```

Another way to specify the sample sizes is with the GROUPNS= option:

```
proc power;
   twosamplemeans test=diff
      groupmeans = 8 | 15
      stddev = 4
      groupns = (25 40)
      power = .;
run;
```

Two-Sample Satterthwaite *t* Test Assuming Unequal Variances

The following statements demonstrate a power computation for the two-sample Satterthwaite *t* test allowing unequal variances. Default values for the DIST=, SIDES=, NULLDIFF=, and ALPHA= options specify a two-sided test for no difference with a normal distribution and a significance level of 0.05.

```
proc power;
   twosamplemeans test=diff_satt
      meandiff = 3
      groupstddevs = 5 | 8
      groupweights = (1 2)
      ntotal = 60
      power = .;
run;
```

Two-Sample Pooled *t* Test of Mean Ratio with Lognormal Data

The following statements demonstrate a power computation for the pooled *t* test of a lognormal mean ratio. Default values for the DIST=, SIDES=, NULLRATIO=, and ALPHA= options specify a two-sided test of

mean ratio = 1 assuming a lognormal distribution and a significance level of 0.05.

```
proc power;
   twosamplemeans test=ratio
      meanratio = 7
      cv = 0.8
      groupns = 50 | 70
      power = .;
run;
```

Additive Equivalence Test for Mean Difference with Normal Data

The following statements demonstrate a sample size computation for the TOST equivalence test for a normal mean difference. A default value of GROUPWEIGHTS=(1 1) specifies a balanced design. Default values for the DIST= and ALPHA= options specify a significance level of 0.05 and an assumption of normally distributed data.

```
proc power;
   twosamplemeans test=equiv_diff
      lower = 2
      upper = 5
      meandiff = 4
      stddev = 8
      ntotal = .
      power = 0.9;
run;
```

Multiplicative Equivalence Test for Mean Ratio with Lognormal Data

The following statements demonstrate a power computation for the TOST equivalence test for a lognormal mean ratio. Default values for the DIST= and ALPHA= options specify a significance level of 0.05 and an assumption of lognormally distributed data.

```
proc power;
   twosamplemeans test=equiv_ratio
      lower = 3
      upper = 7
      meanratio = 5
      cv = 0.75
      npergroup = 50
      power = .;
run;
```

Confidence Interval for Mean Difference

By default CI=DIFF analyzes the conditional probability of obtaining the desired precision, given that the interval contains the true mean difference, as in the following statements. The defaults of SIDES=2 and ALPHA=0.05 specify a two-sided interval with a confidence level of 0.95.

```
proc power;
   twosamplemeans ci = diff
      halfwidth = 4
      stddev = 8
      groupns = (30 35)
      probwidth = .;
```

```
run;
```

TWOSAMPLESURVIVAL Statement

TWOSAMPLESURVIVAL < *options* > **;**

The TWOSAMPLESURVIVAL statement performs power and sample size analyses for comparing two survival curves. The log-rank, Gehan, and Tarone-Ware rank tests are supported.

Summary of Options

Table 96.28 summarizes the *options* available in the TWOSAMPLESURVIVAL statement.

Table 96.28 TWOSAMPLESURVIVAL Statement Options

Option	Description
Define Analysis	
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ACCRUALTIME=	Specifies the accrual time
ALPHA=	Specifies the significance level
FOLLOWUPTIME=	Specifies the follow-up time
SIDES=	Specifies the number of sides and the direction of the statistical test or confidence interval
TOTALTIME=	Specifies the total time
Specify Effects	
CURVE=	Defines a survival curve
GROUPMEDSURVTIMES=	Specifies the median survival times in each group
GROUPSURVEXPHAZARDS=	Specifies exponential hazard rates of the survival curve for each group
GROUPSURVIVAL=	Specifies the survival curve for each group
HAZARDRATIO=	Specifies the hazard ratio
REFSURVEXPHAZARD=	Specifies the exponential hazard rate of the survival curve for the reference group
REFSURVIVAL=	Specifies the survival curve for the reference group
Specify Loss Information	
GROUPLOSS=	Specifies the exponential loss survival curve for each group
GROUPLOSSEXPHAZARDS=	Specifies the exponential hazards of the loss in each group
GROUPMEDLOSSTIMES=	Specifies the median times of the loss in each group
Specify Sample Size and Allocation	
ACCRUALRATEPERGROUP=	Specifies the common accrual rate per group
ACCRUALRATETOTAL=	Specifies the total accrual rate
EVENTSTOTAL=	Specifies the expected total number of events
GROUPACCRUALRATES=	Specifies the accrual rate for each group
GROUPNS=	Specifies the two group sample sizes
GROUPWEIGHTS=	Specifies the sample size allocation weights for the two groups
NFRACTIONAL	Enables fractional input and output for sample sizes

Table 96.28 *continued*

Option	Description
NPERGROUP=	Specifies the common sample size per group
NTOTAL=	Specifies the sample size
Specify Power	
POWER=	Specifies the desired power of the test
Specify Computational Method	
NSUBINTERVAL=	Specifies the number of subintervals per unit time
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.29 summarizes the valid result parameters for different analyses in the TWOSAMPLESURVIVAL statement.

Table 96.29 Summary of Result Parameters in the TWOSAMPLESURVIVAL Statement

Analyses	Solve For	Syntax
TEST=GEHAN	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.
		EVENTSTOTAL=.
		ACCRUALRATETOTAL=.
		ACCRUALRATEPERGROUP=.
TEST=LOGRANK	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.
		EVENTSTOTAL=.
		ACCRUALRATETOTAL=.
		ACCRUALRATEPERGROUP=.
TEST=TARONEWARE	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.
		EVENTSTOTAL=.
		ACCRUALRATETOTAL=.
		ACCRUALRATEPERGROUP=.

Dictionary of Options

ACCRUALRATEPERGROUP=*number-list*

ACCRUALRATEPERG=*number-list*

ARPERGROUP=*number-list*

ARPERG=*number-list*

specifies the common accrual rate per group or requests a solution for the common accrual rate per group by specifying a missing value (ACCRUALRATEPERGROUP=.). The accrual rate per group is the number of subjects in each group that enters the study per time unit during the accrual period. Use of this option implicitly specifies a balanced design. The NFRACTIONAL option is automatically enabled when the ACCRUALRATEPERGROUP= option is used. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

ACCRUALRATETOTAL=*number-list*

ARTOTAL=*number-list*

specifies the total accrual rate or requests a solution for the accrual rate by specifying a missing value (ACCRUALRATETOTAL=.). The total accrual rate is the total number of subjects that enter the study per time unit during the accrual period. The NFRACTIONAL option is automatically enabled when the ACCRUALRATETOTAL= option is used. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

ACCRUALTIME=*number-list* | **MAX**

ACCTIME=*number-list* | **MAX**

ACCRUALT=*number-list* | **MAX**

ACCT=*number-list* | **MAX**

specifies the accrual time. Accrual is assumed to occur uniformly from time 0 to the time specified by the ACCRUALTIME= option. If the GROUPSURVIVAL= or REFSURVIVAL= option is used, then the value of the total time (the sum of accrual and follow-up times) must be less than or equal to the largest time in *each* multipoint (piecewise linear) survival curve. If the ACCRUALRATEPERGROUP=, ACCRUALRATETOTAL=, or GROUPACCRUALRATES= option is used, then the accrual time must be greater than 0. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

ACCRUALTIME=MAX can be used when each scenario in the analysis contains at least one piecewise linear survival curve (in the GROUPSURVIVAL= or REFSURVIVAL= option). It causes the accrual time to be automatically set, separately for each scenario, to the maximum possible time supported by the piecewise linear survival curve(s) in that scenario. It is not compatible with the FOLLOWUPTIME=MAX option or the TOTALTIME= option.

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

CURVE("*label***")=***points*

defines a survival curve.

For the CURVE= option,

label identifies the curve in the output and with the GROUPLOSS=, GROUPSURVIVAL=, and REFSURVIVAL= options.

points specifies one or more (time, survival) pairs on the curve, where the survival value denotes the probability of surviving until at least the specified time.

A single-point curve is interpreted as exponential, and a multipoint curve is interpreted as piecewise linear. Points can be expressed in either of two forms:

- a series of time:survival pairs separated by spaces. For example:

```
1:0.9 2:0.7 3:0.6
```

- a DOLIST of times enclosed in parentheses, followed by a colon (:), followed by a DOLIST of survival values enclosed in parentheses. For example:

```
(1 to 3 by 1):(0.9 0.7 0.6)
```

  The DOLIST format is the same as in the DATA step.

Points can also be expressed as combinations of the two forms. For example:

```
1:0.9 2:0.8 (3 to 6 by 1):(0.7 0.65 0.6 0.55)
```

The points have the following restrictions:

- The time values must be nonnegative and strictly increasing.
- The survival values must be strictly decreasing.
- The survival value at a time of 0 must be equal to 1.
- If there is only one point, then the time must be greater than 0, and the survival value cannot be 0 or 1.

EVENTSTOTAL=*number-list*

EVENTTOTAL=*number-list*

EETOTAL=*number-list*

specifies the expected total number of events—that is, deaths, whether observed or censored—during the entire study period, or requests a solution for this parameter by specifying a missing value (EVENTSTOTAL=.). The NFRACTIONAL option is automatically enabled when the EVENTSTOTAL= option is used. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

FOLLOWUPTIME=*number-list* | **MAX**

FUTIME=*number-list* | **MAX**

FOLLOWUPT=*number-list* | **MAX**

FUT=*number-list* | **MAX**

specifies the follow-up time, the amount of time in the study past the accrual time. If the GROUPSURVIVAL= or REFSURVIVAL= option is used, then the value of the total time (the sum of accrual and follow-up times) must be less than or equal to the largest time in *each* multipoint (piecewise linear) survival curve. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

FOLLOWUPTIME=MAX can be used when each scenario in the analysis contains at least one piecewise linear survival curve (in the GROUPSURVIVAL= or REFSURVIVAL= option). It causes the follow-up time to be automatically set, separately for each scenario, to the maximum possible time supported by the piecewise linear survival curve(s) in that scenario. It is not compatible with the ACCRUALTIME=MAX option or the TOTALTIME= option.

GROUPACCRUALRATES=*grouped-number-list*

GACCRUALRATES=*grouped-number-list*

GROUPARS=*grouped-number-list*

GARS=*grouped-number-list*

specifies the accrual rate for each group. The groupwise accrual rates are the numbers of subjects in each group that enters the study per time unit during the accrual period. The NFRACTIONAL option is automatically enabled when the GROUPACCRUALRATES= option is used. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPLOSS=*grouped-name-list*

GLOSS=*grouped-name-list*

specifies the exponential loss survival curve for each group, by using labels specified with the CURVE= option. Loss is assumed to follow an exponential curve, indicating the expected rate of censoring (in other words, loss to follow-up) over time. For information about specifying the *grouped-name-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPLOSSEXPHAZARDS=*grouped-number-list*

GLOSSEXPHAZARDS=*grouped-number-list*

GROUPLOSSEXPHS=*grouped-number-list*

GLOSSEXPHS=*grouped-number-list*

specifies the exponential hazards of the loss in each group. Loss is assumed to follow an exponential curve, indicating the expected rate of censoring (in other words, loss to follow-up) over time. If none of the GROUPLOSSEXPHAZARDS=, GROUPLOSS=, and GROUPMEDLOSSTIMES= options are used, the default of GROUPLOSSEXPHAZARDS=(0 0) indicates no loss to follow-up. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPMEDLOSSTIMES=*grouped-number-list*

GMEDLOSSTIMES=*grouped-number-list*

GROUPMEDLOSSTS=*grouped-number-list*

GMEDLOSSTS=*grouped-number-list*

specifies the median times of the loss in each group. Loss is assumed to follow an exponential curve, indicating the expected rate of censoring (in other words, loss to follow-up) over time. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPMEDSURVTIMES=*grouped-number-list*

GMEDSURVTIMES=*grouped-number-list*

GROUPMEDSURVTS=*grouped-number-list*

GMEDSURVTS=*grouped-number-list*

specifies the median survival times in each group. When the GROUPMEDSURVTIMES= option is used, the survival curve in each group is assumed to be exponential. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPNS=*grouped-number-list*

GNS=*grouped-number-list*

specifies the two group sample sizes. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPSURVEXPHAZARDS=*grouped-number-list*

GSURVEXPHAZARDS=*grouped-number-list*

GROUPSURVEXPHS=*grouped-number-list*

GEXPHS=*grouped-number-list*

specifies exponential hazard rates of the survival curve for each group. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPSURVIVAL=*grouped-name-list*

GSURVIVAL=*grouped-name-list*

GROUPSURV=*grouped-name-list*

GSURV=*grouped-name-list*

specifies the survival curve for each group, by using labels specified with the CURVE= option. For information about specifying the *grouped-name-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPWEIGHTS=*grouped-number-list*

GWEIGHTS=*grouped-number-list*

specifies the sample size allocation weights for the two groups. This option controls how the total sample size is divided between the two groups. Each pair of values for the two groups represents relative allocation weights. Additionally, if the NFRACTIONAL option is not used, the total sample size is restricted to be equal to a multiple of the sum of the two group weights (so that the resulting design has an integer sample size for each group while adhering exactly to the group allocation weights). Values must be integers unless the NFRACTIONAL option is used. The default value is (1 1), a balanced design with a weight of 1 for each group. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

HAZARDRATIO=*number-list*

HR=*number-list*

specifies the hazard ratio of the second group's survival curve to the first group's survival curve. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. This option is automatically enabled when any of the following options are used: ACCRUALRATEPERGROUP=, ACCRUALRATETOTAL=, EVENTSTOTAL=, and GROUPACCRUALRATES=. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NPERGROUP=*number-list*

NPERG=*number-list*

specifies the common sample size per group or requests a solution for the common sample size per group by specifying a missing value (NPERGROUP=.). Use of this option implicitly specifies a balanced design. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NSUBINTERVAL=*number-list*

NSUBINTERVALS=*number-list*

NSUB=*number-list*

NSUBS=*number-list*

specifies the number of subintervals per unit time to use in internal calculations. Higher values increase computational time and memory requirements but generally lead to more accurate results. The default value is 12. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=
- ACCRUALTIME=
- FOLLOWUPTIME=
- TOTALTIME=
- NSUBINTERVAL=
- ALPHA=
- REFSURVIVAL=
- GROUPSURVIVAL=

- REFSURVEXPHAZARD=
- HAZARDRATIO=
- GROUPSURVEXPHAZARDS=
- GROUPMEDSURVTIMES=
- GROUPLOSSEXPHAZARDS=
- GROUPLOSS=
- GROUPMEDLOSSTIMES=
- GROUPWEIGHTS=
- NTOTAL=
- ACCRUALRATETOTAL=
- EVENTSTOTAL=
- NPERGROUP=
- ACCRUALRATEPERGROUP=
- GROUPNS=
- GROUPACCRUALRATES=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the TWOSAMPLESURVIVAL statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the TWOSAMPLESURVIVAL statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

REFSURVEXPHAZARD=*number-list*

REFSURVEXPH=*number-list*

specifies the exponential hazard rate of the survival curve for the first (reference) group. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

REFSURVIVAL=*name-list*

REFSURV=*name-list*

specifies the survival curve for the first (reference) group, by using labels specified with the CURVE= option. For information about specifying the *name-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test or confidence interval. For information about specifying the *keyword-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012. You can specify the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis favoring better survival in the second group.

L specifies a lower one-sided test, with the alternative hypothesis favoring better survival in the first (reference) group.

By default, SIDES=2.

TEST=GEHAN | LOGRANK | TARONEWARE
specifies the statistical analysis. TEST=GEHAN specifies the Gehan rank test. TEST=LOGRANK (the default) specifies the log-rank test. TEST=TARONEWARE specifies the Tarone-Ware rank test.

TOTALTIME=*number-list* **| MAX**

TOTALT=*number-list* **| MAX**
specifies the total time, which is equal to the sum of accrual and follow-up times. If the GROUPSURVIVAL= or REFSURVIVAL= option is used, then the value of the total time must be less than or equal to the largest time in *each* multipoint (piecewise linear) survival curve. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

TOTALTIME=MAX can be used when each scenario in the analysis contains at least one piecewise linear survival curve (in the GROUPSURVIVAL= or REFSURVIVAL= option). It causes the total time to be automatically set, separately for each scenario, to the maximum possible time supported by the piecewise linear survival curve(s) in that scenario. It is not compatible with the ACCRUALTIME=MAX option or the FOLLOWUPTIME=MAX option.

Restrictions on Option Combinations

To specify the survival curves, choose one of the following parameterizations:

- arbitrary piecewise linear or exponential curves (by using the CURVE= and GROUPSURVIVAL= options)
- curves with proportional hazards (by using the CURVE=, REFSURVIVAL=, and HAZARDRATIO= options)
- exponential curves, by using one of the following parameterizations:
  - median survival times (by using the GROUPMEDSURVTIMES= option)
  - the hazard ratio and the hazard of the reference curve (by using the HAZARDRATIO= and REFSURVEXPHAZARD= options)
  - the individual hazards (by using the GROUPSURVEXPHAZARDS= option)

To specify the study time, use any two of the following three options:

- accrual time (by using the ACCRUALTIME= option)
- follow-up time (by using the FOLLOWUPTIME= option)
- total time, the sum of accrual and follow-up times (by using the TOTALTIME= option)

To specify the sample size and allocation, choose one of the following parameterizations:

- sample size per group in a balanced design (by using the NPERGROUP= option)
- accrual rate per group in a balanced design (by using the ACCRUALRATEPERGROUP= option)
- total sample size and allocation weights (by using the NTOTAL= and GROUPWEIGHTS= options)
- total accrual rate and allocation weights (by using the ACCRUALRATETOTAL= and GROUPWEIGHTS= options)
- expected total number of events and allocation weights (by using the EVENTSTOTAL= and GROUPWEIGHTS= options)
- individual group sample sizes (by using the GROUPNS= option)
- individual group accrual rates (by using the GROUPACCRUALRATES= option)

The values of parameters that involve expected number of events or accrual rate are converted internally to the analogous sample size parameterization (that is, the NPERGROUP=, NTOTAL=, or GROUPNS= option) for the purpose of sample size adjustments according to the presence or absence of the NFRACTIONAL option.

To specify the exponential loss curves, choose one of the following parameterizations:

- a point on the loss curve of each group (by using the CURVE= and GROUPLOSS= options)
- median loss times (by using the GROUPMEDLOSSTIMES= option)
- the individual loss hazards (by using the GROUPLOSSEXPHAZARDS= option)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the TWOSAMPLESURVIVAL statement.

Log-Rank Test for Two Survival Curves

You can use the NPERGROUP= option in a balanced design and specify piecewise linear or exponential survival curves by using the CURVE= and GROUPSURVIVAL= options, as in the following statements. Default values for the SIDES=, ALPHA=, NSUBINTERVAL=, and GROUPLOSSEXPHAZARDS= options specify a two-sided test with a significance level of 0.05, an assumption of no loss to follow-up, and the use of 12 subintervals per unit time in computations.

```
proc power;
   twosamplesurvival test=logrank
      curve("Control")   = (1 2 3):(0.8 0.7 0.6)
      curve("Treatment") = (5):(.6)
      groupsurvival = "Control" | "Treatment"
      accrualtime = 2
      followuptime = 1
      npergroup = 50
      power = .;
run;
```

In the preceding example, the “Control” curve is piecewise linear (since it has more than one point), and the “Treatment” curve is exponential (since it has only one point).

You can also specify an unbalanced design by using the NTOTAL= and GROUPWEIGHTS= options and specify piecewise linear or exponential survival curves with proportional hazards by using the CURVE=, REFSURVIVAL=, and HAZARDRATIO= options:

```
proc power;
   twosamplesurvival test=logrank
      curve("Control")   = (1 2 3):(0.8 0.7 0.6)
      refsurvival = "Control"
      hazardratio = 1.5
      accrualtime = 2
      followuptime = 1
      groupweights = (1 2)
      ntotal = .
      power = 0.8;
run;
```

Instead of computing sample size, you can compute the accrual rate by using the ACCRUALRATETOTAL= option:

```
proc power;
   twosamplesurvival test=logrank
      curve("Control")   = (1 2 3):(0.8 0.7 0.6)
      refsurvival = "Control"
      hazardratio = 1.5
      accrualtime = 2
      followuptime = 1
      groupweights = (1 2)
      accrualratetotal = .
      power = 0.8;
run;
```

or the expected number of events by using the EVENTSTOTAL= option:

```
proc power;
   twosamplesurvival test=logrank
      curve("Control")   = (1 2 3):(0.8 0.7 0.6)
      refsurvival = "Control"
      hazardratio = 1.5
      accrualtime = 2
      followuptime = 1
      groupweights = (1 2)
      eventstotal = .
      power = 0.8;
run;
```

You can also specify sample sizes with the GROUPNS= option and specify exponential survival curves in terms of median survival times:

```
proc power;
   twosamplesurvival test=logrank
      groupmedsurvtimes = (16 22)
      accrualtime = 6
```

```
      totaltime = 18
      groupns = 40 | 60
      power = .;
run;
```

You can also specify exponential survival curves in terms of the hazard ratio and reference hazard. The default value of the GROUPWEIGHTS= option specifies a balanced design.

```
proc power;
   twosamplesurvival test=logrank
      hazardratio = 1.2
      refsurvexphazard = 0.7
      accrualtime = 2
      totaltime = 4
      ntotal = 100
      power = .;
run;
```

You can also specify exponential survival curves in terms of the individual hazards, as in the following statements:

```
proc power;
   twosamplesurvival test=logrank
      groupsurvexphazards = 0.7 | 0.84
      accrualtime = 2
      totaltime = 4
      ntotal = .
      power = 0.9;
run;
```

Gehan Rank Test for Two Survival Curves

In addition to the log-rank test, you can also specify the Gehan tank test, as in the following statements. Default values for the SIDES=, ALPHA=, NSUBINTERVAL=, and GROUPLOSSEXPHAZARDS= options specify a two-sided test with a significance level of 0.05, an assumption of no loss to follow-up, and the use of 12 subintervals per unit time in computations.

```
proc power;
   twosamplesurvival test=gehan
      groupmedsurvtimes = 5 | 7
      accrualtime = 3
      totaltime = 6
      npergroup = .
      power = 0.8;
run;
```

Tarone-Ware Rank Test for Two Survival Curves

You can also specify the Tarone-Ware tank test, as in the following statements. Default values for the SIDES=, ALPHA=, NSUBINTERVAL=, and GROUPLOSSEXPHAZARDS= options specify a two-sided test with a significance level of 0.05, an assumption of no loss to follow-up, and the use of 12 subintervals per unit time in computations.

```
proc power;
   twosamplesurvival test=taroneware
      groupmedsurvtimes = 5 | 7
      accrualtime = 3
      totaltime = 6
      npergroup = 100
      power = .;
run;
```

TWOSAMPLEWILCOXON Statement

TWOSAMPLEWILCOXON < *options* > **;**

The TWOSAMPLEWILCOXON statement performs power and sample size analyses for the Wilcoxon-Mann-Whitney test (also called the Wilcoxon rank-sum test, Mann-Whitney-Wilcoxon test, or Mann-Whitney U test) for two independent groups.

Note that the O'Brien-Castelloe approach to computing power for the Wilcoxon test is approximate, based on asymptotic behavior as the total sample size gets large. The quality of the power approximation degrades for small sample sizes; conversely, the quality of the sample size approximation degrades if the two distributions are far apart, so that only a small sample is needed to detect a significant difference. But this degradation is rarely a problem in practical situations, in which experiments are usually performed for relatively close distributions.

Summary of Options

Table 96.30 summarizes the *options* available in the TWOSAMPLEWILCOXON statement.

Table 96.30 TWOSAMPLEWILCOXON Statement Options

Option	Description
Define Analysis	
TEST=	Specifies the statistical analysis
Specify Analysis Information	
ALPHA=	Specifies the significance level
SIDES=	Specifies the number of sides and the direction of the statistical test
Specify Distributions	
VARDIST=	Defines a distribution for a variable
VARIABLES=	Specifies the distributions of two or more variables
Specify Sample Size and Allocation	
GROUPNS=	Specifies the two group sample sizes
GROUPWEIGHTS=	Specifies the sample size allocation weights for the two groups
NFRACTIONAL	Enables fractional input and output for sample sizes
NPERGROUP=	Specifies the common sample size per group
NTOTAL=	Specifies the sample size

Table 96.30 *continued*

Option	Description
Specify Power	
POWER=	Specifies the desired power of the test
Specify Computational Options	
NBINS=	Specifies the number of categories for each variable
Control Ordering in Output	
OUTPUTORDER=	Controls the output order of parameters

Table 96.31 summarizes the valid result parameters in the TWOSAMPLEWILCOXON statement.

Table 96.31 Summary of Result Parameters in the TWOSAMPLEWILCOXON Statement

Analyses	Solve For	Syntax
TEST=WMW	Power	POWER=.
	Sample size	NTOTAL=.
		NPERGROUP=.

Dictionary of Options

ALPHA=*number-list*

specifies the level of significance of the statistical test. The default is 0.05, which corresponds to the usual $0.05 \times 100\% = 5\%$ level of significance. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPNS=*grouped-number-list*

GNS=*grouped-number-list*

specifies the two group sample sizes. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

GROUPWEIGHTS=*grouped-number-list*

GWEIGHTS=*grouped-number-list*

specifies the sample size allocation weights for the two groups. This option controls how the total sample size is divided between the two groups. Each pair of values for the two groups represents relative allocation weights. Additionally, if the NFRACTIONAL option is not used, the total sample size is restricted to be equal to a multiple of the sum of the two group weights (so that the resulting design has an integer sample size for each group while adhering exactly to the group allocation weights). Values must be integers unless the NFRACTIONAL option is used. The default value is (1 1), a balanced design with a weight of 1 for each group. For information about specifying the *grouped-number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NBINS=*number-list*

specifies the number of categories (or "bins") each variable's distribution is divided into (unless it is ordinal, in which case the categories remain intact) in internal calculations. Higher values increase computational time and memory requirements but generally lead to more accurate results. However, if the value is too high, then numerical instability can occur. Lower values are less likely to produce "No solution computed" errors. The default value is 1000. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NFRACTIONAL

NFRAC

enables fractional input and output for sample sizes. See the section "Sample Size Adjustment Options" on page 8015 for information about the ramifications of the presence (and absence) of the NFRACTIONAL option.

NPERGROUP=*number-list*

NPERG=*number-list*

specifies the common sample size per group or requests a solution for the common sample size per group by specifying a missing value (NPERGROUP=.). Use of this option implicitly specifies a balanced design. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

NTOTAL=*number-list*

specifies the sample size or requests a solution for the sample size by specifying a missing value (NTOTAL=.). For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

OUTPUTORDER=INTERNAL | REVERSE | SYNTAX

controls how the input and default analysis parameters are ordered in the output. OUTPUTORDER=INTERNAL (the default) arranges the parameters in the output according to the following order of their corresponding *options*:

- SIDES=
- NBINS=
- ALPHA=
- VARIABLES=
- GROUPWEIGHTS=
- NTOTAL=
- NPERGROUP=
- GROUPNS=
- POWER=

The OUTPUTORDER=SYNTAX option arranges the parameters in the output in the same order in which their corresponding options are specified in the TWOSAMPLEWILCOXON statement. The OUTPUTORDER=REVERSE option arranges the parameters in the output in the reverse of the order in which their corresponding *options* are specified in the TWOSAMPLEWILCOXON statement.

POWER=*number-list*

specifies the desired power of the test or requests a solution for the power by specifying a missing value (POWER=.). The power is expressed as a probability, a number between 0 and 1, rather than as a percentage. For information about specifying the *number-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

SIDES=*keyword-list*

specifies the number of sides (or tails) and the direction of the statistical test. You can specify the following *keywords*:

1 specifies a one-sided test, with the alternative hypothesis in the same direction as the effect.

2 specifies a two-sided test.

U specifies an upper one-sided test, with the alternative hypothesis indicating an effect greater than the null value.

L specifies a lower one-sided test, with the alternative hypothesis indicating an effect less than the null value.

By default, SIDES=2.

TEST=WMW

specifies the Wilcoxon-Mann-Whitney test for two independent groups This is the default test option.

VARDIST("*label***")=***distribution* **(***parameters***)**

defines a distribution for a variable.

For the VARDIST= option,

label identifies the variable distribution in the output and with the VARIABLES= option.

distribution specifies the distributional form of the variable.

parameters specifies one or more parameters associated with the distribution.

The *distributions* and *parameters* are named and defined in the same way as the distributions and arguments in the CDF SAS function; for more information, see *SAS Functions and CALL Routines: Reference*. Choices for distributional forms and their parameters are as follows:

ORDINAL ((*values*) : (*probabilities*)) is an ordered categorical distribution. The *values* are any numbers separated by spaces. The *probabilities* are numbers between 0 and 1 (inclusive) separated by spaces. Their sum must be exactly 1. The number of *probabilities* must match the number of *values*.

BETA (a, b <, l, r >) is a beta distribution with shape parameters a and b and optional location parameters l and r. The values of a and b must be greater than 0, and l must be less than r. The default values for l and r are 0 and 1, respectively.

BINOMIAL (p, n) is a binomial distribution with probability of success p and number of independent Bernoulli trials n. The value of p must be greater than 0 and less than 1, and n must be an integer greater than 0. If $n = 1$, then the distribution is binary.

EXPONENTIAL (λ) is an exponential distribution with scale λ, which must be greater than 0.

GAMMA (a, λ) is a gamma distribution with shape a and scale λ. The values of a and λ must be greater than 0.

LAPLACE (θ, λ) is a Laplace distribution with location θ and scale λ. The value of λ must be greater than 0.

LOGISTIC (θ, λ) is a logistic distribution with location θ and scale λ. The value of λ must be greater than 0.

LOGNORMAL (θ, λ) is a lognormal distribution with location θ and scale λ. The value of λ must be greater than 0.

NORMAL (θ, λ) is a normal distribution with mean θ and standard deviation λ. The value of λ must be greater than 0.

POISSON (m) is a Poisson distribution with mean m. The value of m must be greater than 0.

UNIFORM (l, r) is a uniform distribution on the interval [l, r], where $l < r$.

VARIABLES=*grouped-name-list*

VARS=*grouped-name-list*

specifies the distributions of two or more variables, using labels specified with the VARDIST= option. For information about specifying the *grouped-name-list*, see the section "Specifying Value Lists in Analysis Statements" on page 8012.

Restrictions on Option Combinations

To specify the sample size and allocation, choose one of the following parameterizations:

- sample size per group in a balanced design (using the NPERGROUP= option)
- total sample size and allocation weights (using the NTOTAL= and GROUPWEIGHTS= options)
- individual group sample sizes (using the GROUPNS= option)

Option Groups for Common Analyses

This section summarizes the syntax for the common analyses that are supported in the TWOSAMPLEWILCOXON statement.

Wilcoxon-Mann-Whitney Test for Comparing Two Distributions

The following statements performs a power analysis for Wilcoxon-Mann-Whitney tests comparing an ordinal variable with each other type of distribution. Default values for the ALPHA=, NBINS=, SIDES=, and TEST= options specify a two-sided test with a significance level of 0.05 and the use of 1000 categories per distribution when discretization is needed.

```
proc power;
   twosamplewilcoxon
      vardist("myordinal") = ordinal ((0 1 2) : (.2 .3 .5))
      vardist("mybeta1") = beta (1, 2)
      vardist("mybeta2") = beta (1, 2, 0, 2)
      vardist("mybinomial") = binomial (.3, 3)
```

```
        vardist("myexponential") = exponential (2)
        vardist("mygamma") = gamma (1.5, 2)
        vardist("mylaplace") = laplace (1, 2)
        vardist("mylogistic") = logistic (1, 2)
        vardist("mylognormal") = lognormal (1, 2)
        vardist("mynormal") = normal (3, 2)
        vardist("mypoisson") = poisson (2)
        vardist("myuniform") = uniform (0, 2)
        variables = "myordinal" | "mybeta1" "mybeta2" "mybinomial"
                                  "myexponential" "mygamma" "mylaplace"
                                  "mylogistic" "mylognormal" "mynormal"
                                  "mypoisson" "myuniform"
        ntotal = 40
        power = .;
run;
```

Details: POWER Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/pwrdet.sas.)

Overview of Power Concepts

In statistical hypothesis testing, you typically express the belief that some effect exists in a population by specifying an alternative hypothesis H_1. You state a null hypothesis H_0 as the assertion that the effect does *not* exist and attempt to gather evidence to reject H_0 in favor of H_1. Evidence is gathered in the form of sample data, and a statistical test is used to assess H_0. If H_0 is rejected but there really is *no* effect, this is called a *Type I error*. The probability of a Type I error is usually designated "alpha" or α, and statistical tests are designed to ensure that α is suitably small (for example, less than 0.05).

If there really is an effect in the population but H_0 is *not* rejected in the statistical test, then a *Type II error* has been made. The probability of a Type II error is usually designated "beta" or β. The probability $1 - \beta$ of avoiding a Type II error—that is, correctly rejecting H_0 and achieving statistical significance—is called the *power*. (**NOTE:** Another more general definition of power is the probability of rejecting H_0 for any given set of circumstances, even those that correspond to H_0 being true. The POWER procedure uses this more general definition.)

An important goal in study planning is to ensure an acceptably high level of power. Sample size plays a prominent role in power computations because the focus is often on determining a sufficient sample size to achieve a certain power, or assessing the power for a range of different sample sizes.

Some of the analyses in the POWER procedure focus on *precision* rather than power. An analysis of confidence interval precision is analogous to a traditional power analysis, with "CI Half-Width" taking the place of effect size and "Prob(Width)" taking the place of power. The *CI Half-Width* is the margin of error associated with the confidence interval, the distance between the point estimate and an endpoint. The *Prob(Width)* is the probability of obtaining a confidence interval with *at most* a target half-width.

Summary of Analyses

Table 96.32 gives a summary of the analyses supported in the POWER procedure. The name of the analysis statement reflects the type of data and design. The TEST=, CI=, and DIST= options specify the focus of the statistical hypothesis (in other words, the criterion on which the research question is based) and the test statistic to be used in data analysis.

Table 96.32 Summary of Analyses

Analysis	Statement	Options
Cox proportional hazards regression: score test	COXREG	
Various tests that involve the chi-square, F, t, or normal distribution, or the distribution of the correlation coefficient under multivariate normality	CUSTOM	DIST=CHISQUARE, DIST=CORR, DIST=F, DIST=NORMAL, DIST=T
Logistic regression: likelihood ratio chi-square test	LOGISTIC	
Multiple linear regression: Type III F test	MULTREG	
Correlation: Fisher's z test	ONECORR	DIST=FISHERZ
Correlation: t test	ONECORR	DIST=T
Binomial proportion: exact test	ONESAMPLEFREQ	TEST=EXACT
Binomial proportion: z test	ONESAMPLEFREQ	TEST=Z
Binomial proportion: z test with continuity adjustment	ONESAMPLEFREQ	TEST=ADJZ
Binomial proportion: exact equivalence test	ONESAMPLEFREQ	TEST=EQUIV_EXACT
Binomial proportion: z equivalence test	ONESAMPLEFREQ	TEST=EQUIV_Z
Binomial proportion: z test with continuity adjustment	ONESAMPLEFREQ	TEST=EQUIV_ADJZ
Binomial proportion: confidence interval	ONESAMPLEFREQ	CI=AGRESTICOULL, CI=JEFFREYS, CI=EXACT, CI=WALD, CI=WALD_CORRECT, CI=WILSON
One-sample t test	ONESAMPLEMEANS	TEST=T
One-sample t test with lognormal data	ONESAMPLEMEANS	TEST=T
DIST=LOGNORMAL		
One-sample equivalence test for mean of normal data	ONESAMPLEMEANS	TEST=EQUIV
One-sample equivalence test for mean of lognormal data	ONESAMPLEMEANS	TEST=EQUIV DIST=LOGNORMAL
Confidence interval for a mean	ONESAMPLEMEANS	CI=T

Table 96.32 *continued*

Analysis	Statement	Options
One-way ANOVA: one-degree-of-freedom contrast	ONEWAYANOVA	TEST=CONTRAST
One-way ANOVA: overall *F* test	ONEWAYANOVA	TEST=OVERALL
McNemar exact conditional test	PAIREDFREQ	
McNemar normal approximation test	PAIREDFREQ	DIST=NORMAL
Paired *t* test	PAIREDMEANS	TEST=DIFF
Paired *t* test of mean ratio with lognormal data	PAIREDMEANS	TEST=RATIO
Paired additive equivalence of mean difference with normal data	PAIREDMEANS	TEST=EQUIV_DIFF
Paired multiplicative equivalence of mean ratio with lognormal data	PAIREDMEANS	TEST=EQUIV_RATIO
Confidence interval for mean of paired differences	PAIREDMEANS	CI=DIFF
Farrington-Manning score test for proportion difference	TWOSAMPLEFREQ	TEST=FM
Farrington-Manning score test for relative risk	TWOSAMPLEFREQ	TEST=FM_RR
Pearson chi-square test for two independent proportions	TWOSAMPLEFREQ	TEST=PCHI
Fisher's exact test for two independent proportions	TWOSAMPLEFREQ	TEST=FISHER
Likelihood ratio chi-square test for two independent proportions	TWOSAMPLEFREQ	TEST=LRCHI
Two-sample *t* test assuming equal variances	TWOSAMPLEMEANS	TEST=DIFF
Two-sample Satterthwaite *t* test assuming unequal variances	TWOSAMPLEMEANS	TEST=DIFF_SATT
Two-sample pooled *t* test of mean ratio with lognormal data	TWOSAMPLEMEANS	TEST=RATIO
Two-sample additive equivalence of mean difference with normal data	TWOSAMPLEMEANS	TEST=EQUIV_DIFF
Two-sample multiplicative equivalence of mean ratio with lognormal data	TWOSAMPLEMEANS	TEST=EQUIV_RATIO
Two-sample confidence interval for mean difference	TWOSAMPLEMEANS	CI=DIFF
Log-rank test for comparing two survival curves	TWOSAMPLESURVIVAL	TEST=LOGRANK
Gehan rank test for comparing two survival curves	TWOSAMPLESURVIVAL	TEST=GEHAN
Tarone-Ware rank test for comparing two survival curves	TWOSAMPLESURVIVAL	TEST=TARONEWARE
Wilcoxon-Mann-Whitney (rank-sum) test	TWOSAMPLEWILCOXON	

Specifying Value Lists in Analysis Statements

To specify one or more scenarios for an analysis parameter (or set of parameters), you provide a list of values for the statement option that corresponds to the parameter(s). To identify the parameter you want to solve for, you place missing values in the appropriate list.

There are five basic types of such lists: *keyword-lists*, *number-lists*, *grouped-number-lists*, *name-lists*, and *grouped-name-lists*. Some parameters, such as the direction of a test, have values represented by one or more keywords in a *keyword-list*. Scenarios for scalar-valued parameters, such as power, are represented by a *number-list*. Scenarios for groups of scalar-valued parameters, such as group sample sizes in a multigroup design, are represented by a *grouped-number-list*. Scenarios for named parameters, such as reference survival curves, are represented by a *name-list*. Scenarios for groups of named parameters, such as group survival curves, are represented by a *grouped-name-list*.

The following subsections explain these five basic types of lists.

Keyword-Lists

A *keyword-list* is a list of one or more keywords, separated by spaces. For example, you can specify both two-sided and upper-tailed versions of a one-sample *t* test as follows:

```
SIDES = 2 U
```

Number-Lists

A *number-list* can be one of two things: a series of one or more numbers expressed in the form of one or more DOLISTs, or a missing value indicator (.).

The DOLIST format is the same as in the DATA step language. For example, for the one-sample *t* test you can specify four scenarios (30, 50, 70, and 100) for a total sample size in any of the following ways.

```
NTOTAL = 30 50 70 100
NTOTAL = 30 to 70 by 20 100
```

A missing value identifies a parameter as the result parameter; it is valid only with options representing parameters you can solve for in a given analysis. For example, you can request a solution for NTOTAL as follows:

```
NTOTAL = .
```

Grouped-Number-Lists

A *grouped-number-list* specifies multiple scenarios for numeric values in two or more groups, possibly including missing value indicators to solve for a specific group. The list can assume one of two general forms, a “crossed” version and a “matched” version.

Crossed Grouped-Number-Lists

The crossed version of a grouped number list consists of a series of *number-lists* (see the section "Number-Lists" on page 8012), one representing each group, with groups separated by a vertical bar (|). The values for each group represent multiple scenarios for that group, and the scenarios for each individual group are crossed to produce the set of all scenarios for the analysis option. For example, you can specify the following six scenarios for the sizes (n_1, n_2) of two groups

$$(20, 30)(20, 40)(20, 50)$$
$$(25, 30)(25, 40)(25, 50)$$

as follows:

```
GROUPNS = 20 25 | 30 40 50
```

If the analysis can solve for a value in one group given the other groups, then one of the *number-lists* in a *crossed grouped-number-list* can be a missing value indicator (.). For example, in a two-sample *t* test you can posit three scenarios for the group 2 sample size while solving for the group 1 sample size:

```
GROUPNS = . | 30 40 50
```

Some analyses can involve more than two groups. For example, you can specify $2 \times 3 \times 1 = 6$ scenarios for the means of three groups in a one-way ANOVA as follows:

```
GROUPMEANS = 10 12 | 10 to 20 by 5 | 24
```

Matched Grouped-Number-Lists

The matched version of a grouped number list consists of a series of numeric lists, each enclosed in parentheses. Each list consists of a value for each group and represents a single scenario for the analysis option. Multiple scenarios for the analysis option are represented by multiple lists. For example, you can express the crossed grouped-number-list

```
GROUPNS = 20 25 | 30 40 50
```

alternatively in a matched format:

```
GROUPNS = (20 30) (20 40) (20 50) (25 30) (25 40) (25 50)
```

The matched version is particularly useful when you want to include only a subset of all combinations of individual group values. For example, you might want to pair 20 only with 50, and 25 only with 30 and 40:

```
GROUPNS = (20 50) (25 30) (25 40)
```

If the analysis can solve for a value in one group given the other groups, then you can replace the value for that group with a missing value indicator (.). If used, the missing value indicator must occur in the same group in every scenario. For example, you can solve for the group 1 sample size (as in the section "Crossed Grouped-Number-Lists" on page 8013) by using a matched format:

```
GROUPNS = (. 30) (. 40) (. 50)
```

Some analyses can involve more than two groups. For example, you can specify two scenarios for the means of three groups in a one-way ANOVA:

```
GROUPMEANS = (15 24 32) (12 25 36)
```

Name-Lists

A *name-list* is a list of one or more names that are enclosed in single or double quotation marks and separated by spaces. For example, you can specify two scenarios for the reference survival curve in a log-rank test as follows:

```
REFSURVIVAL = "Curve A" "Curve B"
```

Grouped-Name-Lists

A *grouped-name-list* specifies multiple scenarios for names in two or more groups. The list can assume one of two general forms, a "crossed" version and a "matched" version.

Crossed Grouped-Name-Lists

The crossed version of a grouped name list consists of a series of *name-lists* (see the section "Name-Lists" on page 8014), one representing each group, with groups separated by a vertical bar (|). The values for each group represent multiple scenarios for that group, and the scenarios for each individual group are crossed to produce the set of all scenarios for the analysis option. For example, you can specify the following six scenarios for the survival curves (c_1, c_2) of two groups

("Curve A", "Curve C")("Curve A", "Curve D")("Curve A", "Curve E")
("Curve B", "Curve C")("Curve B", "Curve D")("Curve B", "Curve E")

as follows:

```
GROUPSURVIVAL = "Curve A" "Curve B" | "Curve C" "Curve D"
                                          "Curve E"
```

Matched Grouped-Name-Lists

The matched version of a grouped name list consists of a series of name lists, each enclosed in parentheses. Each list consists of a name for each group and represents a single scenario for the analysis option. Multiple scenarios for the analysis option are represented by multiple lists. For example, you can express the crossed grouped-name-list

```
GROUPSURVIVAL = "Curve A" "Curve B" | "Curve C" "Curve D"
                                          "Curve E"
```

alternatively in a matched format:

```
GROUPSURVIVAL = ("Curve A" "Curve C")
                ("Curve A" "Curve D")
                ("Curve A" "Curve E")
                ("Curve B" "Curve C")
                ("Curve B" "Curve D")
                ("Curve B" "Curve E")
```

The matched version is particularly useful when you want to include only a subset of all combinations of individual group values. For example, you might want to pair “Curve A” only with “Curve C”, and “Curve B” only with “Curve D” and “Curve E”:

```
GROUPSURVIVAL = ("Curve A" "Curve C")
                ("Curve B" "Curve D")
                ("Curve B" "Curve E")
```

Sample Size Adjustment Options

By default, PROC POWER rounds sample sizes conservatively (down in the input, up in the output) so that all total sizes (and individual group sample sizes, if a multigroup design) are integers. This is generally considered conservative because it selects the closest realistic design providing *at most* the power of the (possibly fractional) input or mathematically optimized design. In addition, in a multigroup design, all group sizes are adjusted to be multiples of the corresponding group weights. For example, if GROUPWEIGHTS = (2 6), then all group 1 sample sizes become multiples of 2, and all group 2 sample sizes become multiples of 6 (and all total sample sizes become multiples of 8).

With the NFRACTIONAL option, sample size input is not rounded, and sample size output (whether total or groupwise) are reported in two versions, a raw “fractional” version and a “ceiling” version rounded up to the nearest integer.

Whenever an input sample size is adjusted, both the original (“nominal”) and adjusted (“actual”) sample sizes are reported. Whenever computed output sample sizes are adjusted, both the original input (“nominal”) power and the achieved (“actual”) power at the adjusted sample size are reported.

Error and Information Output

The Error column in the main output table provides reasons for missing results and flags numerical results that are bounds rather than exact answers. For example, consider the sample size analysis implemented by the following statements:

```
proc power;
   twosamplefreq test=fm_rr
     relativerisk=1.0001
     refproportion=.4
     nullrelativerisk=1
     power=.9
     ntotal=.;
run;
```

Figure 96.6 Error Column

The POWER Procedure
Farrington-Manning Score Test for Relative Risk

Fixed Scenario Elements	
Distribution	Asymptotic normal
Method	Normal approximation
Null Relative Risk	1
Reference (Group 1) Proportion	0.4
Relative Risk	1.0001
Nominal Power	0.9
Number of Sides	2
Alpha	0.05
Group 1 Weight	1
Group 2 Weight	1

Computed N Total		
Actual Power	**N Total**	**Error**
0.473	2.15E+09	Solution is a lower bound

The output in Figure 96.6 reveals that the sample size to achieve a power of 0.9 could not be computed, but that the sample size 2.15E+09 achieves a power of 0.473.

The Info column provides further details about Error column entries, warnings about any boundary conditions detected, and notes about any adjustments to input. Note that the Info column is hidden by default in the main output. You can view it by using the ODS OUTPUT statement to save the output as a data set and the PRINT procedure. For example, the following SAS statements print both the Error and Info columns for a power computation in a two-sample *t* test:

```
proc power;
   twosamplemeans
      meandiff= 0 7
      stddev=2
      ntotal=2 5
      power=.;
   ods output output=Power;
run;

proc print noobs data=Power;
   var MeanDiff NominalNTotal NTotal Power Error Info;
run;
```

The output is shown in Figure 96.7.

Figure 96.7 Error and Info Columns

MeanDiff	NominalNTotal	NTotal	Power	Error	Info
0	2	2	.	Invalid input	N too small / No effect
0	5	4	0.050		Input N adjusted / No effect
7	2	2	.	Invalid input	N too small
7	5	4	0.477		Input N adjusted

The mean difference of 0 specified with the MEANDIFF= option leads to a "No effect" message to appear in the Info column. The sample size of 2 specified with the NTOTAL= option leads to an "Invalid input" message in the Error column and an "NTotal too small" message in the Info column. The sample size of 5 leads to an "Input N adjusted" message in the Info column because it is rounded down to 4 to produce integer group sizes of 2 per group.

Another error you might see is "No solution computed," which indicates that the complex numerical calculations involved could not converge for the input parameters. This situation can happen in extreme cases, but the complexity of the formulas makes it difficult to determine precise cutoffs for anticipating when it will happen. If you see this error when you are solving for sample size, then one potential remedy is to instead request the power for a variety of fixed sample sizes. This remedy can sometimes succeed when solving directly for the sample size fails.

Displayed Output

If you use the PLOTONLY option in the PROC POWER statement, the procedure displays only graphical output. Otherwise, the displayed output of the POWER procedure includes the following:

- the "Fixed Scenario Elements" table, which shows all applicable single-valued analysis parameters, in the following order: distribution, method, parameters that are input explicitly, and parameters that are supplied with defaults
- an output table that shows the following when applicable (in order): the index of the scenario, all multivalued input, ancillary results, the primary computed result, and error descriptions
- plots (if requested)

For each input parameter, the order of the input values is preserved in the output.

Ancillary results include the following:

- Actual Power, the achieved power, if it differs from the input (Nominal) power value
- Actual Prob(Width), the achieved precision probability, if it differs from the input (Nominal) probability value
- Actual Alpha, the achieved significance level, if it differs from the input (Nominal) alpha value
- fractional sample size, if the NFRACTIONAL option is used in the analysis statement

If sample size is the result parameter and the NFRACTIONAL option is used in the analysis statement, then both "Fractional" and "Ceiling" sample size results are displayed. Fractional sample sizes correspond to the "Nominal" values of power or precision probability. Ceiling sample sizes are simply the fractional sample sizes rounded up to the nearest integer; they correspond to "Actual" values of power or precision probability.

ODS Table Names

PROC POWER assigns a name to each table that it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 96.33. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 96.33 ODS Tables Produced by PROC POWER

ODS Table Name	Description	Statement
FixedElements	Factoid with single-valued analysis parameters	Default*
Output	All input and computed analysis parameters, error messages, and information messages for each scenario	Default
PlotContent	Data contained in plots, including analysis parameters and indices identifying plot features. (**NOTE:** This table is saved as a data set and not displayed in PROC POWER output.)	PLOT

*Depends on input.

Computational Resources

Memory

In the TWOSAMPLESURVIVAL statement, the amount of required memory is roughly proportional to the product of the number of subintervals (specified by the NSUBINTERVAL= option) and the total time of the study (specified by the ACCRUALTIME=, FOLLOWUPTIME=, and TOTALTIME= options). If you run out of memory, then you can try either specifying a smaller number of subintervals, changing the time scale to a use a longer time unit (for example, years instead of months), or both.

CPU Time

In the Satterthwaite *t* test analysis (TWOSAMPLEMEANS TEST=DIFF_SATT), the required CPU time grows as the mean difference decreases relative to the standard deviations. In the PAIREDFREQ statement, the required CPU time for the exact power computation (METHOD=EXACT) grows with the sample size.

Computational Methods and Formulas

This section describes the approaches that PROC POWER uses to compute power for each analysis. The first subsection defines some common notation. The following subsections describe the various power analyses, including discussions of the data, statistical test, and power formula for each analysis. Unless otherwise indicated, computed values for parameters besides power (for example, sample size) are obtained by solving power formulas for the desired parameters.

Common Notation

Table 96.34 displays notation for some of the more common parameters across analyses. The Associated Syntax column shows examples of relevant analysis statement options, where applicable.

Table 96.34 Common Notation

Symbol	Description	Associated Syntax
α	Significance level	ALPHA=
N	Total sample size	NTOTAL=, NPAIRS=
n_i	Sample size in ith group	NPERGROUP=, GROUPNS=
w_i	Allocation weight for ith group (standardized to sum to 1)	GROUPWEIGHTS=
μ	(Arithmetic) mean	MEAN=
μ_i	(Arithmetic) mean in ith group	GROUPMEANS=, PAIREDMEANS=
μ_{diff}	(Arithmetic) mean difference, $\mu_2 - \mu_1$ or $\mu_T - \mu_R$	MEANDIFF=
μ_0	Null mean or mean difference (arithmetic)	NULL=, NULLDIFF=
γ	Geometric mean	MEAN=
γ_i	Geometric mean in ith group	GROUPMEANS=, PAIREDMEANS=
γ_0	Null mean or mean ratio (geometric)	NULL=, NULLRATIO=
σ	Standard deviation (or common standard deviation per group)	STDDEV=
σ_i	Standard deviation in ith group	GROUPSTDDEVS=, PAIREDSTDDEVS=
σ_{diff}	Standard deviation of differences	
CV	Coefficient of variation, defined as the ratio of the standard deviation to the (arithmetic) mean on the original data scale	CV=, PAIREDCVS=
ρ	Correlation	CORR=
μ_T, μ_R	Treatment and reference (arithmetic) means for equivalence test	GROUPMEANS=, PAIREDMEANS=
γ_T, γ_R	Treatment and reference geometric means for equivalence test	GROUPMEANS=, PAIREDMEANS=
θ_L	Lower equivalence bound	LOWER=
θ_U	Upper equivalence bound	UPPER=
$t(\nu, \delta)$	t distribution with df ν and noncentrality δ	

Table 96.34 *continued*

Symbol	Description	Associated Syntax
$F(\nu_1, \nu_2, \lambda)$	F distribution with numerator *df* ν_1, denominator *df* ν_2, and noncentrality λ	
$t_{p;\nu}$	pth percentile of t distribution with *df* ν	
$F_{p;\nu_1,\nu_2}$	pth percentile of F distribution with numerator *df* ν_1 and denominator *df* ν_2	
$\mathrm{Bin}(N, p)$	Binomial distribution with sample size N and proportion p	

A "lower one-sided" test is associated with SIDES=L (or SIDES=1 with the effect smaller than the null value), and an "upper one-sided" test is associated with SIDES=U (or SIDES=1 with the effect larger than the null value).

Owen (1965) defines a function, known as Owen's Q, that is convenient for representing terms in power formulas for confidence intervals and equivalence tests:

$$Q_\nu(t, \delta; a, b) = \frac{\sqrt{2\pi}}{\Gamma(\frac{\nu}{2})2^{\frac{\nu-2}{2}}} \int_a^b \Phi\left(\frac{tx}{\sqrt{\nu}} - \delta\right) x^{\nu-1}\phi(x)\mathrm{d}x$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ are the density and cumulative distribution function of the standard normal distribution, respectively.

Analyses in the COXREG Statement

Score Test of a Single Scalar Predictor in Cox Proportional Hazards Regression (TEST=SCORE)

The power-computing formula is based on Hsieh and Lavori (2000, equation (2) and the section "Variance Inflation Factor" on page 556).

Define the following notation for a Cox proportional hazards regression analysis:

$$\begin{aligned}
N &= \#\text{subjects} \quad \text{(NTOTAL)} \\
K &= \#\text{predictors} \\
\mathbf{x} &= (x_1, \ldots, x_K)' = \text{vector of predictors} \\
x_1 &= \text{predictor of interest} \\
\mathbf{x}_{-1} &= (x_2, \ldots, x_K)' \\
h(t|\mathbf{x}) &= \text{hazard function for survival time given } \mathbf{x}\text{, evaluated at time } t \\
h_0(t) &= \text{baseline hazard at time } t \\
h_\mathrm{r} &= \text{hazard ratio for one-unit increase in } x_1 \quad \text{(HAZARDRATIO)} \\
p_e &= \mathrm{Prob}(\text{event is uncensored}) \quad \text{(EVENTPROB)} \\
\sigma &= \text{standard deviation of } x_1 \quad \text{(STDDEV)} \\
\rho &= \mathrm{Corr}(\mathbf{x}_{-1}, x_1) \\
R^2 &= \rho^2 = R^2 \text{ value from regression of } x_1 \text{ on } \mathbf{x}_{-1} \quad \text{(RSQUARE)}
\end{aligned}$$

The Cox proportional hazards regression model is

$$\begin{aligned}\log\left(h(t|\mathbf{x}/h_0(t)\right) &= \beta\mathbf{x} \\ &= \beta_1 x_1 + \cdots + \beta_K x_K\end{aligned}$$

You can convert a regression coefficient to a hazard ratio by using the equation $h_{\mathrm{r}} = \exp(\beta_1)$.

The hypothesis test of the first predictor variable is

$$H_0{:}h_{\mathrm{r}} = 1$$

$$H_1{:}\begin{cases} h_{\mathrm{r}} \neq 1, & \text{two-sided} \\ h_{\mathrm{r}} < 1, & \text{upper one-sided} \\ h_{\mathrm{r}} > 1, & \text{lower one-sided} \end{cases}$$

The upper and lower one-sided cases are expressed differently than in other analyses. This is because $h_{\mathrm{r}} > 1$ corresponds to a negative correlation between the tested predictor and survival and thus, by the convention used in PROC POWER for regression analyses, the lower side.

The approximate power is

$$\text{power} = \begin{cases} \Phi\left(z_\alpha - \sigma\sqrt{Np_e(1-R^2)}\log(h_{\mathrm{r}})\right), & \text{upper one-sided} \\ 1 - \Phi\left(z_{1-\alpha} - \sigma\sqrt{Np_e(1-R^2)}\log(h_{\mathrm{r}})\right), & \text{lower one-sided} \\ \Phi\left(z_{\frac{\alpha}{2}} - \sigma\sqrt{Np_e(1-R^2)}\log(h_{\mathrm{r}})\right) + 1 - \Phi\left(z_{1-\frac{\alpha}{2}} - \sigma\sqrt{Np_e(1-R^2)}\log(h_{\mathrm{r}})\right), & \text{two-sided} \end{cases}$$

For the one-sided cases, a closed-form inversion of the power equation yields an approximate total sample size

$$N = \left(\frac{\left(z_{\text{power}} + z_{1-\alpha}\right)^2}{p_e(1-R^2)\sigma^2\left(\log(h_{\mathrm{r}})\right)^2}\right)$$

For the two-sided case, the solution for N is obtained by numerically inverting the power equation.

Analyses in the CUSTOM Statement

The first of the following subsections specifies notation for the analyses in the CUSTOM statement, and the remaining subsections show power formulas and special cases equivalent to analyses elsewhere in PROC POWER or PROC GLMPOWER for each type of test statistic distribution.

Notation

Table 96.35 displays notation for the analyses in the CUSTOM statement.

Table 96.35 Common Notation

Symbol	Description	Associated Option
N	Total sample size	NTOTAL=
p_1	Test degrees of freedom	TESTDF=

Table 96.35 *continued*

Symbol	Description	Associated Option
p	Model degrees of freedom	MODELDF=
$\lambda^\star$	Primary noncentrality for χ^2 or F distribution	PRIMNC=
$\delta^\star$	Primary noncentrality for t or normal distribution	PRIMNC=
$\rho_{YX_1\|X_{-1}}$	Partial correlation between $p_1 \geq 1$ variables in a set X_1 and a variable Y, adjusting for $p - p_1$ variables in a (possibly empty) set X_{-1} and assuming multivariate normality	CORR=
m_1	Scalar multiplier of primary noncentrality	PNCMULT=
m_2	Scalar multiplier of critical value	CRITMULT=
α	Significance level	ALPHA=

Noncentral Chi-Square Distribution (DIST=CHISQUARE)

The power is computed as

$$\text{power} = P\left(\chi^2(p_1, Nm_1\lambda^\star) \geq m_2\chi^2_{1-\alpha}(p_1)\right)$$

The sample size is computed by numerically inverting the power formula.

The logistic regression analysis in the LOGISTIC statement is a special case in which $p_1 = 1$ and $m_2 = 1$. The two-sided Wilcoxon-Mann-Whitney test (TWOSAMPLEWILCOXON) is a special case in which $m_1 = 1$.

Correlation Coefficient Distribution Assuming Multivariate Normal Data (DIST=CORR)

The distribution of the correlation coefficient given the value of $\rho_{YX_1|X_{-1}}$ is shown in (Johnson, Kotz, and Balakrishnan 1995, Chapter 32). The one-sided cases are restricted to $p_1 = 1$.

The power is computed as

$$\text{power} = \begin{cases} P\left[R^2_{YX_1|X_{-1}} \geq \frac{F_{1-\alpha}(p_1,N-p)}{F_{1-\alpha}(p_1,N-p)+\frac{N-p}{p_1}} \mid \rho_{YX_1|X_{-1}}\right] & \text{two-sided} \\ P\left[R_{YX_1|X_{-1}} \geq \frac{t_{1-\alpha}(N-p-1)}{\left(t^2_{1-\alpha}(N-p-1)+N-p-1\right)^{\frac{1}{2}}} \mid \rho_{YX_1|X_{-1}}\right] & \text{upper one-sided} \\ P\left[R_{YX_1|X_{-1}} \leq \frac{t_{\alpha}(N-p-1)}{\left(t^2_{\alpha}(N-p-1)+N-p-1\right)^{\frac{1}{2}}} \mid \rho_{YX_1|X_{-1}}\right] & \text{lower one-sided} \end{cases}$$

The sample size is computed by numerically inverting the power formula.

The Pearson correlation analysis that assumes joint multivariate normality (ONECORR DIST=T MODEL=RANDOM) and the multiple regression analysis that assumes joint multivariate normal predictors (MULTREG MODEL=RANDOM) are special cases.

Noncentral F Distribution (DIST=F)

The power is computed as

$$\text{power} = P\left(F(p_1, N-p, Nm_1\lambda^\star) \geq F_{1-\alpha}(p_1, N-p)\right)$$

The sample size is computed by numerically inverting the power formula.

The two-sided contrast in one-way analysis of variance (ONEWAYANOVA) is a special case in which $p_1 = 1$ and $m_1 = 1$. The following analyses are special cases in which $m_1 = 1$:

- multiple regression that assumes fixed predictors (MULTREG MODEL=FIXED)
- overall F test in one-way analysis of variance (ONEWAYANOVA)
- Type III F test in a fixed-effect univariate linear model (PROC GLMPOWER)
- multivariate Type III F test in a fixed-effect multivariate linear model (PROC GLMPOWER)

Normal Distribution (DIST=NORMAL)

The power is computed as

$$\text{power} = \begin{cases} \Phi\left(m_2 z_\alpha - N^{\frac{1}{2}} m_1 \delta^\star\right) & \text{upper one-sided} \\ 1 - \Phi\left(m_2 z_{1-\alpha} - N^{\frac{1}{2}} m_1 \delta^\star\right) & \text{lower one-sided} \\ P\left(\chi^2(1, N(m_1\delta^\star)^2) \geq m_2^2 \chi^2_{1-\alpha}(1)\right) & \text{two-sided} \end{cases}$$

The sample size is computed by numerically inverting the power equation.

Cox proportional hazards regression (COXREG) and the log-rank test (TWOSAMPLESURVIVAL) are special cases in which $m_1 = 1$ and $m_2 = 1$. The Wilcoxon-Mann-Whitney test (TWOSAMPLEWILCOXON) is a special case in which $m_1 = 1$. Other special cases occur among the tests for binomial proportions (ONESAMPLEFREQ, PAIREDFREQ, and TWOSAMPLEFREQ).

Noncentral T Distribution (DIST=T)

The power is computed as

$$\text{power} = \begin{cases} P\left(F(1, N-p, N(m_1\delta^\star)^2) \geq F_{1-\alpha}(1, N-p)\right) & \text{two-sided} \\ P\left(t(N-p, N^{\frac{1}{2}} m_1 \delta^\star) \geq t_{1-\alpha}(N-p)\right) & \text{upper one-sided} \\ P\left(t(N-p, N^{\frac{1}{2}} m_1 \delta^\star) \leq t_{\alpha}(N-p)\right) & \text{lower one-sided} \end{cases}$$

The sample size is computed by numerically inverting the power formula.

The standard t tests in the ONESAMPLEMEANS and PAIREDMEANS statements are special cases in which $p = 1$ and $m_1 = 1$. The pooled t test (TWOSAMPLEMEANS) is a special case in which $p = 2$ and $m_1 = 1$. The Pearson correlation analysis that assumes fixed X values (ONECORR DIST=T MODEL=FIXED) is a special case in which $m_1 = 1$ and $p = p^\star + 2$, where $p^\star$ is the number of variables that are partialed out. The one-sided contrast in one-way analysis of variance (ONEWAYANOVA) is special case in which $m_1 = 1$ and $p_1 = G$, where G is the number of groups.

Analyses in the LOGISTIC Statement

Likelihood Ratio Chi-Square Test for One Predictor (TEST=LRCHI)

The power-computing formula is based on Shieh and O'Brien (1998); Shieh (2000); Self, Mauritsen, and Ohara (1992), and Hsieh (1989).

Define the following notation for a logistic regression analysis:

$$
\begin{aligned}
N &= \#\text{subjects} \quad \text{(NTOTAL)} \\
K &= \#\text{predictors (not counting intercept)} \\
\mathbf{x} &= (x_1, \ldots, x_K)' = \text{random variables for predictor vector} \\
\mathbf{x}_{-1} &= (x_2, \ldots, x_K)' \\
\boldsymbol{\mu} &= (\mu_1, \ldots, \mu_K)' = \mathrm{E}\mathbf{x} = \text{mean predictor vector} \\
\mathbf{x}_i &= (x_{i1}, \ldots, x_{iK})' = \text{predictor vector for subject } i \quad (i \in 1, \ldots, N) \\
Y &= \text{random variable for response (0 or 1)} \\
Y_i &= \text{response for subject } i \quad (i \in 1, \ldots, N) \\
p_i &= \mathrm{Prob}(Y_i = 1 | \mathbf{x}_i) \quad (i \in 1, \ldots, N) \\
\phi &= \mathrm{Prob}(Y_i = 1 | \mathbf{x}_i = \boldsymbol{\mu}) \quad \text{(RESPONSEPROB)} \\
U_j &= \text{unit change for } j\text{th predictor} \quad \text{(UNITS)} \\
\mathrm{OR}_j &= \mathrm{Odds}(Y_i = 1 | x_{ij} = c)/\mathrm{Odds}(Y_i = 1 | x_{ij} = c - U_j) \quad (c \text{ arbitrary}, i \in 1, \ldots, N, \\
&\quad j \in 1, \ldots, K) \quad \text{(TESTODDSRATIO if } j = 1, \text{COVODDSRATIOS if } j > 1) \\
\Psi_0 &= \text{intercept in full model (INTERCEPT)} \\
\boldsymbol{\Psi} &= (\Psi_1, \ldots, \Psi_K)' = \text{regression coefficients in full model} \\
&\quad (\Psi_1 = \text{TESTREGCOEFF, others = COVREGCOEFFS}) \\
\rho &= \mathrm{Corr}(\mathbf{x}_{-1}, x_1) \quad \text{(CORR)} \\
c_j &= \#\text{distinct possible values of } x_{ij} \quad (j \in 1, \ldots, K)(\text{for any } i) \quad \text{(NBINS)} \\
x^{\star}_{gj} &= g\text{th possible value of } x_{ij} \quad (g \in 1, \ldots, c_j)(j \in 1, \ldots, K) \\
&\quad (\text{for any } i) \quad \text{(VARDIST)}
\end{aligned}
$$

$$
\begin{aligned}
\pi_{gj} &= \mathrm{Prob}\left(x_{ij} = x^{\star}_{gj}\right) \quad (g \in 1, \ldots, c_j)(j \in 1, \ldots, K) \\
&\quad (\text{for any } i) \quad \text{(VARDIST)} \\
C &= \prod_{j=1}^{K} c_j = \#\text{possible values of } \mathbf{x}_i \quad (\text{for any } i) \\
\mathbf{x}^{\star}_m &= m\text{th possible value of } \mathbf{x}_i \quad (m \in 1, \ldots, C) \\
\pi_m &= \mathrm{Prob}\left(\mathbf{x}_i = \mathbf{x}^{\star}_m\right) \quad (m \in 1, \ldots, C)
\end{aligned}
$$

The logistic regression model is

$$
\log\left(\frac{p_i}{1 - p_i}\right) = \Psi_0 + \boldsymbol{\Psi}'\mathbf{x}_i
$$

The hypothesis test of the first predictor variable is

$$H_0{:}\Psi_1 = 0$$
$$H_1{:}\Psi_1 \neq 0$$

Assuming independence among all predictor variables, π_m is defined as follows:

$$\pi_m = \prod_{j=1}^{K} \pi_{h(m,j)j} \quad (m \in 1, \ldots, C)$$

where $h(m, j)$ is calculated according to the following algorithm:

$z = m;$
do $\quad j = K \quad$ to $\quad 1;$
$\quad h(m, j) = \mathrm{mod}(z - 1, c_j) + 1;$
$\quad z = \mathrm{floor}((z - 1)/c_j) + 1;$
end;

This algorithm causes the elements of the transposed vector $\{h(m, 1), \ldots, h(m, K)\}$ to vary fastest to slowest from right to left as m increases, as shown in the following table of $h(m, j)$ values:

$h(m, j)$		j				
		1	2	$\cdots$	$K-1$	K
	1	1	1	$\cdots$	1	1
	1	1	1	$\cdots$	1	2
	$\vdots$			$\vdots$		
	$\vdots$	1	1	$\cdots$	1	c_K
	$\vdots$	1	1	$\cdots$	2	1
	$\vdots$	1	1	$\cdots$	2	2
	$\vdots$			$\vdots$		
m	$\vdots$	1	1	$\cdots$	2	c_K
	$\vdots$			$\vdots$		
	$\vdots$	c_1	c_2	$\cdots$	c_{K-1}	1
	$\vdots$	c_1	c_2	$\cdots$	c_{K-1}	2
	$\vdots$			$\vdots$		
	C	c_1	c_2	$\cdots$	c_{K-1}	c_K

The $\mathbf{x}_m^\star$ values are determined in a completely analogous manner.

The discretization is handled as follows (unless the distribution is ordinal, or binomial with sample size parameter at least as large as requested number of bins): for x_j, generate c_j quantiles at evenly spaced

probability values such that each such quantile is at the midpoint of a bin with probability $\frac{1}{c_j}$. In other words,

$$x^{\star}_{gj} = \left(\frac{g-0.5}{c_j}\right) \text{th quantile of relevant distribution}$$
$$(g \in 1, \ldots, c_j)(j \in 1, \ldots, K)$$
$$\pi_{gj} = \frac{1}{c_j} \quad \text{(same for all } g\text{)}$$

The primary noncentrality for the power computation is

$$\Delta^{\star} = 2 \sum_{m=1}^{C} \pi_m \left[b'(\theta_m) \left(\theta_m - \theta^{\star}_m\right) - \left(b(\theta_m) - b(\theta^{\star}_m)\right)\right]$$

where

$$b'(\theta) = \frac{\exp(\theta)}{1+\exp(\theta)}$$
$$b(\theta) = \log\left(1+\exp(\theta)\right)$$
$$\theta_m = \Psi_0 + \boldsymbol{\Psi}'\mathbf{x}^{\star}_m$$
$$\theta^{\star}_m = \Psi^{\star}_0 + \boldsymbol{\Psi}^{\star\prime}\mathbf{x}^{\star}_m$$

where

$$\Psi^{\star}_0 = \Psi_0 + \Psi_1\mu_1 = \text{ intercept in reduced model, absorbing the tested predictor}$$
$$\boldsymbol{\Psi}^{\star} = (0, \Psi_2, \ldots, \Psi_K)' = \text{ coefficients in reduced model}$$

The power is

$$\text{power} = P\left(\chi^2(1, \Delta^{\star} N(1-\rho^2)) \geq \chi^2_{1-\alpha}(1)\right)$$

The factor $(1-\rho^2)$ is the adjustment for correlation between the predictor that is being tested and other predictors, from Hsieh (1989).

Alternative input parameterizations are handled by the following transformations:

$$\Psi_0 = \log\left(\frac{\phi}{1-\phi}\right) - \boldsymbol{\Psi}'\boldsymbol{\mu}$$
$$\Psi_j = \frac{\log(\mathrm{OR}_j)}{U_j} \quad (j \in 1, \ldots, K)$$

Analyses in the MULTREG Statement

Type III F Test in Multiple Regression (TEST=TYPE3)

Maxwell (2000) discusses a number of different ways to represent effect sizes (and to compute exact power based on them) in multiple regression. PROC POWER supports two of these, multiple partial correlation and R^2 in full and reduced models.

Let p denote the total number of predictors in the full model (excluding the intercept), and let Y denote the response variable. You are testing that the coefficients of $p_1 \geq 1$ predictors in a set X_1 are 0, controlling for all of the other predictors X_{-1}, which consists of $p - p_1 \geq 0$ variables.

The hypotheses can be expressed in two different ways. The first is in terms of $\rho_{YX_1|X_{-1}}$, the multiple partial correlation between the predictors in X_1 and the response Y adjusting for the predictors in X_{-1}:

$$H_0{:}\rho^2_{YX_1|X_{-1}} = 0$$
$$H_1{:}\rho^2_{YX_1|X_{-1}} > 0$$

The second is in terms of the multiple correlations in full ($\rho_{Y|(X_1,X_{-1})}$) and reduced ($\rho_{Y|X_{-1}}$) nested models:

$$H_0{:}\rho^2_{Y|(X_1,X_{-1})} - \rho^2_{Y|X_{-1}} = 0$$
$$H_1{:}\rho^2_{Y|(X_1,X_{-1})} - \rho^2_{Y|X_{-1}} > 0$$

Note that the squared values of $\rho_{Y|(X_1,X_{-1})}$ and $\rho_{Y|X_{-1}}$ are the population R^2 values for full and reduced models.

The test statistic can be written in terms of the sample multiple partial correlation $R_{YX_1|X_{-1}}$,

$$F = \begin{cases} (N-1-p)\frac{R^2_{YX_1|X_{-1}}}{1-R^2_{YX_1|X_{-1}}}, & \text{intercept} \\ (N-p)\frac{R^2_{YX_1|X_{-1}}}{1-R^2_{YX_1|X_{-1}}}, & \text{no intercept} \end{cases}$$

or the sample multiple correlations in full ($R_{Y|(X_1,X_{-1})}$) and reduced ($R_{Y|X_{-1}}$) models,

$$F = \begin{cases} (N-1-p)\frac{R^2_{Y|(X_1,X_{-1})}-R^2_{Y|X_{-1}}}{1-R^2_{Y|(X_1,X_{-1})}}, & \text{intercept} \\ (N-p)\frac{R^2_{Y|(X_1,X_{-1})}-R^2_{Y|X_{-1}}}{1-R^2_{Y|(X_1,X_{-1})}}, & \text{no intercept} \end{cases}$$

The test is the usual Type III F test in multiple regression:

$$\text{Reject} \quad H_0 \quad \text{if} \begin{cases} F \geq F_{1-\alpha}(p_1, N-1-p), & \text{intercept} \\ F \geq F_{1-\alpha}(p_1, N-p), & \text{no intercept} \end{cases}$$

Although the test is invariant to whether the predictors are assumed to be random or fixed, the power is affected by this assumption. If the response and predictors are assumed to have a joint multivariate normal

distribution, then the exact power is given by the following formula:

$$\text{power} = \begin{cases} P\left[\left(\frac{N-1-p}{p_1}\right)\left(\frac{R^2_{YX_1|X_{-1}}}{1-R^2_{YX_1|X_{-1}}}\right) \geq F_{1-\alpha}(p_1, N-1-p)\right], & \text{intercept} \\ P\left[\left(\frac{N-p}{p_1}\right)\left(\frac{R^2_{YX_1|X_{-1}}}{1-R^2_{YX_1|X_{-1}}}\right) \geq F_{1-\alpha}(p_1, N-p)\right], & \text{no intercept} \end{cases}$$

$$= \begin{cases} P\left[R^2_{YX_1|X_{-1}} \geq \frac{F_{1-\alpha}(p_1,N-1-p)}{F_{1-\alpha}(p_1,N-1-p)+\frac{N-1-p}{p_1}}\right], & \text{intercept} \\ P\left[R^2_{YX_1|X_{-1}} \geq \frac{F_{1-\alpha}(p_1,N-p)}{F_{1-\alpha}(p_1,N-p)+\frac{N-p}{p_1}}\right], & \text{no intercept} \end{cases}$$

The distribution of $R^2_{YX_1|X_{-1}}$ (for any $\rho^2_{YX_1|X_{-1}}$) is given in Chapter 32 of Johnson, Kotz, and Balakrishnan (1995). Sample size tables are presented in Gatsonis and Sampson (1989).

If the predictors are assumed to have fixed values, then the exact power is given by the noncentral F distribution. The noncentrality parameter is

$$\lambda = N\frac{\rho^2_{YX_1|X_{-1}}}{1-\rho^2_{YX_1|X_{-1}}}$$

or equivalently,

$$\lambda = N\frac{\rho^2_{Y|(X_1,X_{-1})} - \rho^2_{Y|X_{-1}}}{1-\rho^2_{Y|(X_1,X_{-1})}}$$

The power is

$$\text{power} = \begin{cases} P\left(F(p_1, N-1-p, \lambda) \geq F_{1-\alpha}(p_1, N-1-p)\right), & \text{intercept} \\ P\left(F(p_1, N-p, \lambda) \geq F_{1-\alpha}(p_1, N-p)\right), & \text{no intercept} \end{cases}$$

The minimum acceptable input value of N depends on several factors, as shown in Table 96.36.

Table 96.36 Minimum Acceptable Sample Size Values in the MULTREG Statement

Predictor Type	Intercept in Model?	$p_1 = 1$?	Minimum N
Random	Yes	Yes	$p + 3$
Random	Yes	No	$p + 2$
Random	No	Yes	$p + 2$
Random	No	No	$p + 1$
Fixed	Yes	Yes or No	$p + 2$
Fixed	No	Yes or No	$p + 1$

Analyses in the ONECORR Statement

Fisher's z Test for Pearson Correlation (TEST=PEARSON DIST=FISHERZ)

Fisher's z transformation (Fisher 1921) of the sample correlation $R_{Y|(X_1,X_{-1})}$ is defined as

$$z = \frac{1}{2}\log\left(\frac{1+R_{Y|(X_1,X_{-1})}}{1-R_{Y|(X_1,X_{-1})}}\right)$$

Fisher's z test assumes the approximate normal distribution $N(\mu,\sigma^2)$ for z, where

$$\mu = \frac{1}{2}\log\left(\frac{1+\rho_{Y|(X_1,X_{-1})}}{1-\rho_{Y|(X_1,X_{-1})}}\right) + \frac{\rho_{Y|(X_1,X_{-1})}}{2(N-1-p^\star)}$$

and

$$\sigma^2 = \frac{1}{N-3-p^\star}$$

where $p^\star$ is the number of variables partialed out (Anderson 1984, pp. 132–133) and $\rho_{Y|(X_1,X_{-1})}$ is the partial correlation between Y and X_1 adjusting for the set of zero or more variables X_{-1}.

The test statistic

$$z^\star = (N-3-p^\star)^{\frac{1}{2}}\left[z - \frac{1}{2}\log\left(\frac{1+\rho_0}{1-\rho_0}\right) - \frac{\rho_0}{2(N-1-p^\star)}\right]$$

is assumed to have a normal distribution $N(\delta,\nu)$, where ρ_0 is the null partial correlation and δ and ν are derived from Section 16.33 of Stuart and Ord (1994):

$$\delta = (N-3-p^\star)^{\frac{1}{2}}\left[\frac{1}{2}\log\left(\frac{1+\rho_{Y|(X_1,X_{-1})}}{1-\rho_{Y|(X_1,X_{-1})}}\right) + \frac{\rho_{Y|(X_1,X_{-1})}}{2(N-1-p^\star)}\left(1+\frac{5+\rho^2_{Y|(X_1,X_{-1})}}{4(N-1-p^\star)}+\right.\right.$$
$$\left.\left.\frac{11+2\rho^2_{Y|(X_1,X_{-1})}+3\rho^4_{Y|(X_1,X_{-1})}}{8(N-1-p^\star)^2}\right) - \frac{1}{2}\log\left(\frac{1+\rho_0}{1-\rho_0}\right) - \frac{\rho_0}{2(N-1-p^\star)}\right]$$
$$\nu = \frac{N-3-p^\star}{N-1-p^\star}\left[1+\frac{4-\rho^2_{Y|(X_1,X_{-1})}}{2(N-1-p^\star)} + \frac{22-6\rho^2_{Y|(X_1,X_{-1})}-3\rho^4_{Y|(X_1,X_{-1})}}{6(N-1-p^\star)^2}\right]$$

The approximate power is computed as

$$\text{power} = \begin{cases} \Phi\left(\frac{\delta - z_{1-\alpha}}{\nu^{\frac{1}{2}}}\right), & \text{upper one-sided} \\ \Phi\left(\frac{-\delta - z_{1-\alpha}}{\nu^{\frac{1}{2}}}\right), & \text{lower one-sided} \\ \Phi\left(\frac{\delta - z_{1-\frac{\alpha}{2}}}{\nu^{\frac{1}{2}}}\right) + \Phi\left(\frac{-\delta - z_{1-\frac{\alpha}{2}}}{\nu^{\frac{1}{2}}}\right), & \text{two-sided} \end{cases}$$

Because the test is biased, the achieved significance level might differ from the nominal significance level. The actual alpha is computed in the same way as the power, except that the correlation $\rho_{Y|(X_1,X_{-1})}$ is replaced by the null correlation ρ_0.

t Test for Pearson Correlation (TEST=PEARSON DIST=T)

The two-sided case is identical to multiple regression with an intercept and $p_1 = 1$, which is discussed in the section "Analyses in the MULTREG Statement" on page 8027.

Let $p^\star$ denote the number of variables partialed out. For the one-sided cases, the test statistic is

$$t = (N-2-p^\star)^{\frac{1}{2}} \frac{R_{YX_1|X_{-1}}}{\left(1-R^2_{YX_1|X_{-1}}\right)^{\frac{1}{2}}}$$

which is assumed to have a null distribution of $t(N-2-p^\star)$.

If the X and Y variables are assumed to have a joint multivariate normal distribution, then the exact power is given by the following formula:

$$\text{power} = \begin{cases} P\left[(N-2-p^\star)^{\frac{1}{2}} \frac{R_{YX_1|X_{-1}}}{\left(1-R^2_{YX_1|X_{-1}}\right)^{\frac{1}{2}}} \geq t_{1-\alpha}(N-2-p^\star)\right], & \text{upper one-sided} \\ P\left[(N-2-p^\star)^{\frac{1}{2}} \frac{R_{YX_1|X_{-1}}}{\left(1-R^2_{YX_1|X_{-1}}\right)^{\frac{1}{2}}} \leq t_{\alpha}(N-2-p^\star)\right], & \text{lower one-sided} \end{cases}$$

$$= \begin{cases} P\left[R_{Y|(X_1,X_{-1})} \geq \frac{t_{1-\alpha}(N-2-p^\star)}{\left(t^2_{1-\alpha}(N-2-p^\star)+N-2-p^\star\right)^{\frac{1}{2}}}\right], & \text{upper one-sided} \\ P\left[R_{Y|(X_1,X_{-1})} \leq \frac{t_{\alpha}(N-2-p^\star)}{\left(t^2_{\alpha}(N-2-p^\star)+N-2-p^\star\right)^{\frac{1}{2}}}\right], & \text{lower one-sided} \end{cases}$$

The distribution of $R_{Y|(X_1,X_{-1})}$ (given the underlying true correlation $\rho_{Y|(X_1,X_{-1})}$) is given in Chapter 32 of Johnson, Kotz, and Balakrishnan (1995).

If the X variables are assumed to have fixed values, then the exact power is given by the noncentral t distribution $t(N-2-p^\star, \delta)$, where the noncentrality is

$$\delta = N^{\frac{1}{2}} \frac{\rho_{YX_1|X_{-1}}}{\left(1-\rho^2_{YX_1|X_{-1}}\right)^{\frac{1}{2}}}$$

The power is

$$\text{power} = \begin{cases} P\left(t(N-2-p^\star, \delta) \geq t_{1-\alpha}(N-2-p^\star)\right), & \text{upper one-sided} \\ P\left(t(N-2-p^\star, \delta) \leq t_{\alpha}(N-2-p^\star)\right), & \text{lower one-sided} \end{cases}$$

Analyses in the ONESAMPLEFREQ Statement

Exact Test of a Binomial Proportion (TEST=EXACT)

Let X be distributed as $\mathrm{Bin}(N, p)$. The hypotheses for the test of the proportion p are as follows:

$$H_0{:}p = p_0$$

$$H_1{:}\begin{cases} p \neq p_0, & \text{two-sided} \\ p > p_0, & \text{upper one-sided} \\ p < p_0, & \text{lower one-sided} \end{cases}$$

The exact test assumes binomially distributed data and requires $N \geq 1$ and $0 < p_0 < 1$. The test statistic is

$$X = \text{number of successes} \sim \mathrm{Bin}(N, p)$$

The significance probability α is split symmetrically for two-sided tests, in the sense that each tail is filled with as much as possible up to $\alpha/2$.

Exact power computations are based on the binomial distribution and computing formulas such as the following from Johnson, Kotz, and Kemp (1992, equation 3.20):

$$P(X \geq C|N, p) = P\left(F_{\nu_1,\nu_2} \leq \frac{\nu_2 p}{\nu_1(1-p)}\right) \quad \text{where } \nu_1 = 2C \text{ and } \nu_2 = 2(N - C + 1)$$

Let C_L and C_U denote lower and upper critical values, respectively. Let α_a denote the achieved (actual) significance level, which for two-sided tests is the sum of the favorable major tail (α_M) and the opposite minor tail (α_m).

For the upper one-sided case,

$$\begin{aligned} C_U &= \min\{C : P(X \geq C|p_0) \leq \alpha\} \\ \text{Reject } H_0 \text{ if } & X \geq C_U \\ \alpha_a &= P(X \geq C_U|p_0) \\ \text{power} &= P(X \geq C_U|p) \end{aligned}$$

For the lower one-sided case,

$$\begin{aligned} C_L &= \max\{C : P(X \leq C|p_0) \leq \alpha\} \\ \text{Reject } H_0 \text{ if } & X \leq C_L \\ \alpha_a &= P(X \leq C_L|p_0) \\ \text{power} &= P(X \leq C_L|p) \end{aligned}$$

For the two-sided case,

$$\begin{aligned} C_L &= \max\{C : P(X \leq C|p_0) \leq \frac{\alpha}{2}\} \\ C_U &= \min\{C : P(X \geq C|p_0) \leq \frac{\alpha}{2}\} \\ \text{Reject } H_0 \text{ if } & X \leq C_L \text{ or } X \geq C_U \\ \alpha_a &= P(X \leq C_L \text{ or } X \geq C_U|p_0) \\ \text{power} &= P(X \leq C_L \text{ or } X \geq C_U|p) \end{aligned}$$

z Test for Binomial Proportion Using Null Variance (TEST=Z VAREST=NULL)

For the normal approximation test, the test statistic is

$$Z(X) = \frac{X - Np_0}{[Np_0(1-p_0)]^{\frac{1}{2}}}$$

For the METHOD=EXACT option, the computations are the same as described in the section "Exact Test of a Binomial Proportion (TEST=EXACT)" on page 8031 except for the definitions of the critical values.

For the upper one-sided case,

$$C_U = \min\{C : Z(C) \geq z_{1-\alpha}\}$$

For the lower one-sided case,

$$C_L = \max\{C : Z(C) \leq z_{\alpha}\}$$

For the two-sided case,

$$C_L = \max\{C : Z(C) \leq z_{\frac{\alpha}{2}}\}$$
$$C_U = \min\{C : Z(C) \geq z_{1-\frac{\alpha}{2}}\}$$

For the METHOD=NORMAL option, the test statistic $Z(X)$ is assumed to have the normal distribution

$$\mathrm{N}\left(\frac{N^{\frac{1}{2}}(p-p_0)}{[p_0(1-p_0)]^{\frac{1}{2}}}, \frac{p(1-p)}{p_0(1-p_0)}\right)$$

The approximate power is computed as

$$\text{power} = \begin{cases} \Phi\left(\dfrac{z_\alpha + \sqrt{N}\frac{p-p_0}{\sqrt{p_0(1-p_0)}}}{\sqrt{\frac{p(1-p)}{p_0(1-p_0)}}}\right), & \text{upper one-sided} \\ \Phi\left(\dfrac{z_\alpha - \sqrt{N}\frac{p-p_0}{\sqrt{p_0(1-p_0)}}}{\sqrt{\frac{p(1-p)}{p_0(1-p_0)}}}\right), & \text{lower one-sided} \\ \Phi\left(\dfrac{z_{\frac{\alpha}{2}} + \sqrt{N}\frac{p-p_0}{\sqrt{p_0(1-p_0)}}}{\sqrt{\frac{p(1-p)}{p_0(1-p_0)}}}\right) + \Phi\left(\dfrac{z_{\frac{\alpha}{2}} - \sqrt{N}\frac{p-p_0}{\sqrt{p_0(1-p_0)}}}{\sqrt{\frac{p(1-p)}{p_0(1-p_0)}}}\right), & \text{two-sided} \end{cases}$$

The approximate sample size is computed in closed form for the one-sided cases by inverting the power equation,

$$N = \left(\frac{z_{\text{power}}\sqrt{p(1-p)} + z_{1-\alpha}\sqrt{p_0(1-p_0)}}{p - p_0}\right)^2$$

and by numerical inversion for the two-sided case.

z Test for Binomial Proportion Using Sample Variance (TEST=Z VAREST=SAMPLE)

For the normal approximation test using the sample variance, the test statistic is

$$Z_s(X) = \frac{X - Np_0}{[N\hat{p}(1-\hat{p})]^{\frac{1}{2}}}$$

where $\hat{p} = X/N$.

For the METHOD=EXACT option, the computations are the same as described in the section "Exact Test of a Binomial Proportion (TEST=EXACT)" on page 8031 except for the definitions of the critical values.

For the upper one-sided case,

$$C_U = \min\{C : Z_s(C) \geq z_{1-\alpha}\}$$

For the lower one-sided case,

$$C_L = \max\{C : Z_s(C) \leq z_{\alpha}\}$$

For the two-sided case,

$$C_L = \max\{C : Z_s(C) \leq z_{\frac{\alpha}{2}}\}$$
$$C_U = \min\{C : Z_s(C) \geq z_{1-\frac{\alpha}{2}}\}$$

For the METHOD=NORMAL option, the test statistic $Z_s(X)$ is assumed to have the normal distribution

$$\mathrm{N}\left(\frac{N^{\frac{1}{2}}(p-p_0)}{[p(1-p)]^{\frac{1}{2}}}, 1\right)$$

(see Chow, Shao, and Wang (2003, p. 82)).

The approximate power is computed as

$$\text{power} = \begin{cases} \Phi\left(z_\alpha + \sqrt{N}\frac{p-p_0}{\sqrt{p(1-p)}}\right), & \text{upper one-sided} \\ \Phi\left(z_\alpha - \sqrt{N}\frac{p-p_0}{\sqrt{p(1-p)}}\right), & \text{lower one-sided} \\ \Phi\left(z_{\frac{\alpha}{2}} + \sqrt{N}\frac{p-p_0}{\sqrt{p(1-p)}}\right) + \Phi\left(z_{\frac{\alpha}{2}} - \sqrt{N}\frac{p-p_0}{\sqrt{p(1-p)}}\right), & \text{two-sided} \end{cases}$$

The approximate sample size is computed in closed form for the one-sided cases by inverting the power equation,

$$N = p(1-p)\left(\frac{z_{\text{power}} + z_{1-\alpha}}{p - p_0}\right)^2$$

and by numerical inversion for the two-sided case.

z Test for Binomial Proportion with Continuity Adjustment Using Null Variance (TEST=ADJZ VAREST=NULL)

For the normal approximation test with continuity adjustment, the test statistic is (Pagano and Gauvreau 1993, p. 295):

$$Z_c(X) = \frac{X - Np_0 + 0.5(1_{\{X<Np_0\}}) - 0.5(1_{\{X>Np_0\}})}{[Np_0(1-p_0)]^{\frac{1}{2}}}$$

For the METHOD=EXACT option, the computations are the same as described in the section "Exact Test of a Binomial Proportion (TEST=EXACT)" on page 8031 except for the definitions of the critical values.

For the upper one-sided case,

$$C_U = \min\{C : Z_c(C) \geq z_{1-\alpha}\}$$

For the lower one-sided case,

$$C_L = \max\{C : Z_c(C) \leq z_{\alpha}\}$$

For the two-sided case,

$$C_L = \max\{C : Z_c(C) \leq z_{\frac{\alpha}{2}}\}$$
$$C_U = \min\{C : Z_c(C) \geq z_{1-\frac{\alpha}{2}}\}$$

For the METHOD=NORMAL option, the test statistic $Z_c(X)$ is assumed to have the normal distribution $N(\mu, \sigma^2)$, where μ and σ^2 are derived as follows.

For convenience of notation, define

$$k = \frac{1}{2\sqrt{Np_0(1-p_0)}}$$

Then

$$E\left[Z_c(X)\right] = 2kNp - 2kNp_0 + kP(X < Np_0) - kP(X > Np_0)$$

and

$$\begin{aligned}\mathrm{Var}\left[Z_c(X)\right] = {} & 4k^2Np(1-p) + k^2\left[1 - P(X = Np_0)\right] - k^2\left[P(X < Np_0) - P(X > Np_0)\right]^2 \\ & + 4k^2\left[E\left(X1_{\{X<Np_0\}}\right) - E\left(X1_{\{X>Np_0\}}\right)\right] - 4k^2Np\left[P(X < Np_0) - P(X > Np_0)\right]\end{aligned}$$

The probabilities $P(X = Np_0)$, $P(X < Np_0)$, and $P(X > Np_0)$ and the truncated expectations $E\left(X1_{\{X<Np_0\}}\right)$ and $E\left(X1_{\{X>Np_0\}}\right)$ are approximated by assuming the normal-approximate distribution of X, $N(Np, Np(1-p))$. Letting $\phi(\cdot)$ and $\Phi(\cdot)$ denote the standard normal PDF and CDF, respectively, and defining d as

$$d = \frac{Np_0 - Np}{[Np(1-p)]^{\frac{1}{2}}}$$

the terms are computed as follows:

$$
\begin{aligned}
P(X = Np_0) &= 0 \\
P(X < Np_0) &= \Phi(d) \\
P(X > Np_0) &= 1 - \Phi(d) \\
E\left(X1_{\{X<Np_0\}}\right) &= Np\Phi(d) - [Np(1-p)]^{\frac{1}{2}}\,\phi(d) \\
E\left(X1_{\{X>Np_0\}}\right) &= Np\,[1-\Phi(d)] + [Np(1-p)]^{\frac{1}{2}}\,\phi(d)
\end{aligned}
$$

The mean and variance of $Z_c(X)$ are thus approximated by

$$\mu = k\,[2Np - 2Np_0 + 2\Phi(d) - 1]$$

and

$$\sigma^2 = 4k^2\left[Np(1-p) + \Phi(d)\,(1-\Phi(d)) - 2\,(Np(1-p))^{\frac{1}{2}}\,\phi(d)\right]$$

The approximate power is computed as

$$
\text{power} = \begin{cases}
\Phi\left(\frac{z_\alpha+\mu}{\sigma}\right), & \text{upper one-sided} \\
\Phi\left(\frac{z_\alpha-\mu}{\sigma}\right), & \text{lower one-sided} \\
\Phi\left(\frac{z_{\frac{\alpha}{2}}+\mu}{\sigma}\right) + \Phi\left(\frac{z_{\frac{\alpha}{2}}-\mu}{\sigma}\right), & \text{two-sided}
\end{cases}
$$

The approximate sample size is computed by numerical inversion.

z Test for Binomial Proportion with Continuity Adjustment Using Sample Variance (TEST=ADJZ VAREST=SAMPLE)

For the normal approximation test with continuity adjustment using the sample variance, the test statistic is

$$Z_{cs}(X) = \frac{X - Np_0 + 0.5(1_{\{X<Np_0\}}) - 0.5(1_{\{X>Np_0\}})}{[N\,\hat{p}(1-\hat{p})]^{\frac{1}{2}}}$$

where $\hat{p} = X/N$.

For the METHOD=EXACT option, the computations are the same as described in the section "Exact Test of a Binomial Proportion (TEST=EXACT)" on page 8031 except for the definitions of the critical values.

For the upper one-sided case,

$$C_U = \min\{C : Z_{cs}(C) \geq z_{1-\alpha}\}$$

For the lower one-sided case,

$$C_L = \max\{C : Z_{cs}(C) \leq z_\alpha\}$$

For the two-sided case,

$$C_L = \max\{C : Z_{cs}(C) \leq z_{\frac{\alpha}{2}}\}$$
$$C_U = \min\{C : Z_{cs}(C) \geq z_{1-\frac{\alpha}{2}}\}$$

For the METHOD=NORMAL option, the test statistic $Z_{cs}(X)$ is assumed to have the normal distribution $N(\mu, \sigma^2)$, where μ and σ^2 are derived as follows.

For convenience of notation, define

$$k = \frac{1}{2\sqrt{Np(1-p)}}$$

Then

$$E\left[Z_{cs}(X)\right] \approx 2kNp - 2kNp_0 + kP(X < Np_0) - kP(X > Np_0)$$

and

$$\begin{aligned}\mathrm{Var}\left[Z_{cs}(X)\right] \approx\ & 4k^2Np(1-p) + k^2\left[1 - P(X = Np_0)\right] - k^2\left[P(X < Np_0) - P(X > Np_0)\right]^2 \\ & + 4k^2\left[E\left(X1_{\{X<Np_0\}}\right) - E\left(X1_{\{X>Np_0\}}\right)\right] - 4k^2Np\left[P(X < Np_0) - P(X > Np_0)\right]\end{aligned}$$

The probabilities $P(X = Np_0)$, $P(X < Np_0)$, and $P(X > Np_0)$ and the truncated expectations $E\left(X1_{\{X<Np_0\}}\right)$ and $E\left(X1_{\{X>Np_0\}}\right)$ are approximated by assuming the normal-approximate distribution of X, $N(Np, Np(1-p))$. Letting $\phi(\cdot)$ and $\Phi(\cdot)$ denote the standard normal PDF and CDF, respectively, and defining d as

$$d = \frac{Np_0 - Np}{[Np(1-p)]^{\frac{1}{2}}}$$

the terms are computed as follows:

$$\begin{aligned}P(X = Np_0) &= 0 \\ P(X < Np_0) &= \Phi(d) \\ P(X > Np_0) &= 1 - \Phi(d) \\ E\left(X1_{\{X<Np_0\}}\right) &= Np\Phi(d) - [Np(1-p)]^{\frac{1}{2}}\phi(d) \\ E\left(X1_{\{X>Np_0\}}\right) &= Np\left[1 - \Phi(d)\right] + [Np(1-p)]^{\frac{1}{2}}\phi(d)\end{aligned}$$

The mean and variance of $Z_{cs}(X)$ are thus approximated by

$$\mu = k\left[2Np - 2Np_0 + 2\Phi(d) - 1\right]$$

and

$$\sigma^2 = 4k^2\left[Np(1-p) + \Phi(d)\left(1 - \Phi(d)\right) - 2\left(Np(1-p)\right)^{\frac{1}{2}}\phi(d)\right]$$

The approximate power is computed as

$$\text{power} = \begin{cases} \Phi\left(\frac{z_\alpha+\mu}{\sigma}\right), & \text{upper one-sided} \\ \Phi\left(\frac{z_\alpha-\mu}{\sigma}\right), & \text{lower one-sided} \\ \Phi\left(\frac{z_{\frac{\alpha}{2}}+\mu}{\sigma}\right) + \Phi\left(\frac{z_{\frac{\alpha}{2}}-\mu}{\sigma}\right), & \text{two-sided} \end{cases}$$

The approximate sample size is computed by numerical inversion.

Exact Equivalence Test of a Binomial Proportion (TEST=EQUIV_EXACT)

The hypotheses for the equivalence test are

$$H_0{:}p < \theta_L \quad \text{or} \quad p > \theta_U$$
$$H_1{:}\theta_L \le p \le \theta_U$$

where θ_L and θ_U are the lower and upper equivalence bounds, respectively.

The analysis is the two one-sided tests (TOST) procedure as described in Chow, Shao, and Wang (2003) on p. 84, but using exact critical values as on p. 116 instead of normal-based critical values.

Two different hypothesis tests are carried out:

$$H_{a0}{:}p < \theta_L$$
$$H_{a1}{:}p \ge \theta_L$$

and

$$H_{b0}{:}p > \theta_U$$
$$H_{b1}{:}p \le \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for p is contained completely within (θ_L, θ_U).

The test statistic for each of the two tests (H_{a0} versus H_{a1} and H_{b0} versus H_{b1}) is

$$X = \text{number of successes} \sim \text{Bin}(N, p)$$

Let C_U denote the critical value of the exact upper one-sided test of H_{a0} versus H_{a1}, and let C_L denote the critical value of the exact lower one-sided test of H_{b0} versus H_{b1}. These critical values are computed in the section "Exact Test of a Binomial Proportion (TEST=EXACT)" on page 8031. Both of these tests are rejected if and only if $C_U \le X \le C_L$. Thus, the exact power of the equivalence test is

$$\begin{aligned} \text{power} &= P\left(C_U \le X \le C_L\right) \\ &= P\left(X \ge C_U\right) - P\left(X \ge C_L + 1\right) \end{aligned}$$

The probabilities are computed using Johnson and Kotz (1970, equation 3.20).

z Equivalence Test for Binomial Proportion Using Null Variance (TEST=EQUIV_Z VAREST=NULL)

The hypotheses for the equivalence test are

$$H_0{:}p < \theta_L \quad \text{or} \quad p > \theta_U$$
$$H_1{:}\theta_L \le p \le \theta_U$$

where θ_L and θ_U are the lower and upper equivalence bounds, respectively.

The analysis is the two one-sided tests (TOST) procedure as described in Chow, Shao, and Wang (2003) on p. 84, but using the null variance instead of the sample variance.

Two different hypothesis tests are carried out:

$$H_{a0}{:}p < \theta_L$$
$$H_{a1}{:}p \ge \theta_L$$

and

$$H_{b0}{:}p > \theta_U$$
$$H_{b1}{:}p \le \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for p is contained completely within (θ_L, θ_U).

The test statistic for the test of H_{a0} versus H_{a1} is

$$Z_L(X) = \frac{X - N\theta_L}{[N\theta_L(1-\theta_L)]^{\frac{1}{2}}}$$

The test statistic for the test of H_{b0} versus H_{b1} is

$$Z_U(X) = \frac{X - N\theta_U}{[N\theta_U(1-\theta_U)]^{\frac{1}{2}}}$$

For the METHOD=EXACT option, let C_U denote the critical value of the exact upper one-sided test of H_{a0} versus H_{a1} using $Z_L(X)$. This critical value is computed in the section "z Test for Binomial Proportion Using Null Variance (TEST=Z VAREST=NULL)" on page 8032. Similarly, let C_L denote the critical value of the exact lower one-sided test of H_{b0} versus H_{b1} using $Z_U(X)$. Both of these tests are rejected if and only if $C_U \le X \le C_L$. Thus, the exact power of the equivalence test is

$$\begin{aligned}\text{power} &= P\left(C_U \le X \le C_L\right)\\ &= P\left(X \ge C_U\right) - P\left(X \ge C_L + 1\right)\end{aligned}$$

The probabilities are computed using Johnson and Kotz (1970, equation 3.20).

For the METHOD=NORMAL option, the test statistic $Z_L(X)$ is assumed to have the normal distribution

$$\mathrm{N}\left(\frac{N^{\frac{1}{2}}(p-\theta_L)}{[\theta_L(1-\theta_L)]^{\frac{1}{2}}}, \frac{p(1-p)}{\theta_L(1-\theta_L)}\right)$$

and the test statistic $Z_U(X)$ is assumed to have the normal distribution

$$\mathrm{N}\left(\frac{N^{\frac{1}{2}}(p-\theta_U)}{[\theta_U(1-\theta_U)]^{\frac{1}{2}}}, \frac{p(1-p)}{\theta_U(1-\theta_U)}\right)$$

(see Chow, Shao, and Wang (2003, p. 84)). The approximate power is computed as

$$\text{power} = \Phi\left(\frac{z_\alpha - \sqrt{N}\frac{p-\theta_U}{\sqrt{\theta_U(1-\theta_U)}}}{\sqrt{\frac{p(1-p)}{\theta_U(1-\theta_U)}}}\right) + \Phi\left(\frac{z_\alpha + \sqrt{N}\frac{p-\theta_L}{\sqrt{\theta_L(1-\theta_L)}}}{\sqrt{\frac{p(1-p)}{\theta_L(1-\theta_L)}}}\right) - 1$$

The approximate sample size is computed by numerically inverting the power formula, using the sample size estimate N_0 of Chow, Shao, and Wang (2003, p. 85) as an initial guess:

$$N_0 = p(1-p)\left(\frac{z_{1-\alpha} + z_{(1+\text{power})/2}}{0.5(\theta_U - \theta_L) - \mid p - 0.5(\theta_L + \theta_U)\mid}\right)^2$$

z Equivalence Test for Binomial Proportion Using Sample Variance (TEST=EQUIV_Z VAREST=SAMPLE)

The hypotheses for the equivalence test are

$$H_0{:}p < \theta_L \quad \text{or} \quad p > \theta_U$$
$$H_1{:}\theta_L \le p \le \theta_U$$

where θ_L and θ_U are the lower and upper equivalence bounds, respectively.

The analysis is the two one-sided tests (TOST) procedure as described in Chow, Shao, and Wang (2003) on p. 84.

Two different hypothesis tests are carried out:

$$H_{a0}{:}p < \theta_L$$
$$H_{a1}{:}p \ge \theta_L$$

and

$$H_{b0}{:}p > \theta_U$$
$$H_{b1}{:}p \le \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for p is contained completely within (θ_L, θ_U).

The test statistic for the test of H_{a0} versus H_{a1} is

$$Z_{sL}(X) = \frac{X - N\theta_L}{[N\hat{p}(1-\hat{p})]^{\frac{1}{2}}}$$

where $\hat{p} = X/N$.

The test statistic for the test of H_{b0} versus H_{b1} is

$$Z_{sU}(X) = \frac{X - N\theta_U}{[N\,\hat{p}(1-\hat{p})]^{\frac{1}{2}}}$$

For the METHOD=EXACT option, let C_U denote the critical value of the exact upper one-sided test of H_{a0} versus H_{a1} using $Z_{sL}(X)$. This critical value is computed in the section "z Test for Binomial Proportion Using Sample Variance (TEST=Z VAREST=SAMPLE)" on page 8033. Similarly, let C_L denote the critical value of the exact lower one-sided test of H_{b0} versus H_{b1} using $Z_{sU}(X)$. Both of these tests are rejected if and only if $C_U \le X \le C_L$. Thus, the exact power of the equivalence test is

$$\begin{aligned}\text{power} &= P\,(C_U \le X \le C_L)\\ &= P\,(X \ge C_U) - P\,(X \ge C_L + 1)\end{aligned}$$

The probabilities are computed using Johnson and Kotz (1970, equation 3.20).

For the METHOD=NORMAL option, the test statistic $Z_{sL}(X)$ is assumed to have the normal distribution

$$\mathrm{N}\left(\frac{N^{\frac{1}{2}}(p-\theta_L)}{[p(1-p)]^{\frac{1}{2}}}, 1\right)$$

and the test statistic $Z_{sU}(X)$ is assumed to have the normal distribution

$$\mathrm{N}\left(\frac{N^{\frac{1}{2}}(p-\theta_U)}{[p(1-p)]^{\frac{1}{2}}}, 1\right)$$

(see Chow, Shao, and Wang (2003), p. 84).

The approximate power is computed as

$$\text{power} = \Phi\left(z_\alpha - \sqrt{N}\frac{p-\theta_U}{\sqrt{p(1-p)}}\right) + \Phi\left(z_\alpha + \sqrt{N}\frac{p-\theta_L}{\sqrt{p(1-p)}}\right) - 1$$

The approximate sample size is computed by numerically inverting the power formula, using the sample size estimate N_0 of Chow, Shao, and Wang (2003, p. 85) as an initial guess:

$$N_0 = p(1-p)\left(\frac{z_{1-\alpha} + z_{(1+\text{power})/2}}{0.5(\theta_U - \theta_L) - \mid p - 0.5(\theta_L + \theta_U)\mid}\right)^2$$

z Equivalence Test for Binomial Proportion with Continuity Adjustment Using Null Variance (TEST=EQUIV_ADJZ VAREST=NULL)

The hypotheses for the equivalence test are

$$\begin{aligned}&H_0{:}p < \theta_L \quad \text{or} \quad p > \theta_U\\ &H_1{:}\theta_L \le p \le \theta_U\end{aligned}$$

where θ_L and θ_U are the lower and upper equivalence bounds, respectively.

The analysis is the two one-sided tests (TOST) procedure as described in Chow, Shao, and Wang (2003) on p. 84, but using the null variance instead of the sample variance.

Two different hypothesis tests are carried out:

$$H_{a0}{:}p < \theta_L$$
$$H_{a1}{:}p \geq \theta_L$$

and

$$H_{b0}{:}p > \theta_U$$
$$H_{b1}{:}p \leq \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for p is contained completely within (θ_L, θ_U).

The test statistic for the test of H_{a0} versus H_{a1} is

$$Z_{cL}(X) = \frac{X - N\theta_L + 0.5(1_{\{X<N\theta_L\}}) - 0.5(1_{\{X>N\theta_L\}})}{\left[N\hat{\theta_L}(1-\hat{\theta_L})\right]^{\frac{1}{2}}}$$

where $\hat{p} = X/N$.

The test statistic for the test of H_{b0} versus H_{b1} is

$$Z_{cU}(X) = \frac{X - N\theta_U + 0.5(1_{\{X<N\theta_U\}}) - 0.5(1_{\{X>N\theta_U\}})}{\left[N\hat{\theta_U}(1-\hat{\theta_U})\right]^{\frac{1}{2}}}$$

For the METHOD=EXACT option, let C_U denote the critical value of the exact upper one-sided test of H_{a0} versus H_{a1} using $Z_{cL}(X)$. This critical value is computed in the section "z Test for Binomial Proportion with Continuity Adjustment Using Null Variance (TEST=ADJZ VAREST=NULL)" on page 8034. Similarly, let C_L denote the critical value of the exact lower one-sided test of H_{b0} versus H_{b1} using $Z_{cU}(X)$. Both of these tests are rejected if and only if $C_U \leq X \leq C_L$. Thus, the exact power of the equivalence test is

$$\begin{aligned}\text{power} &= P\,(C_U \leq X \leq C_L)\\ &= P\,(X \geq C_U) - P\,(X \geq C_L + 1)\end{aligned}$$

The probabilities are computed using Johnson and Kotz (1970, equation 3.20).

For the METHOD=NORMAL option, the test statistic $Z_{cL}(X)$ is assumed to have the normal distribution $N(\mu_L, \sigma_L^2)$, and $Z_{cU}(X)$ is assumed to have the normal distribution $N(\mu_U, \sigma_U^2)$, where μ_L, μ_U, σ_L^2, and σ_U^2 are derived as follows.

For convenience of notation, define

$$k_L = \frac{1}{2\sqrt{N\theta_L(1-\theta_L)}}$$
$$k_U = \frac{1}{2\sqrt{N\theta_U(1-\theta_U)}}$$

Then

$$E\left[Z_{cL}(X)\right] \approx 2k_L Np - 2k_L N\theta_L + k_L P(X < N\theta_L) - k_L P(X > N\theta_L)$$
$$E\left[Z_{cU}(X)\right] \approx 2k_U Np - 2k_U N\theta_U + k_U P(X < N\theta_U) - k_U P(X > N\theta_U)$$

and

$$\begin{aligned}\mathrm{Var}\left[Z_{cL}(X)\right] \approx\ & 4k_L^2 Np(1-p) + k_L^2\left[1 - P(X = N\theta_L)\right] - k_L^2\left[P(X < N\theta_L) - P(X > N\theta_L)\right]^2 \\ & + 4k_L^2\left[E\left(X1_{\{X<N\theta_L\}}\right) - E\left(X1_{\{X>N\theta_L\}}\right)\right] - 4k_L^2 Np\left[P(X < N\theta_L) - P(X > N\theta_L)\right] \\ \mathrm{Var}\left[Z_{cU}(X)\right] \approx\ & 4k_U^2 Np(1-p) + k_U^2\left[1 - P(X = N\theta_U)\right] - k_U^2\left[P(X < N\theta_U) - P(X > N\theta_U)\right]^2 \\ & + 4k_U^2\left[E\left(X1_{\{X<N\theta_U\}}\right) - E\left(X1_{\{X>N\theta_U\}}\right)\right] - 4k_U^2 Np\left[P(X < N\theta_U) - P(X > N\theta_U)\right]\end{aligned}$$

The probabilities $P(X = N\theta_L)$, $P(X < N\theta_L)$, $P(X > N\theta_L)$, $P(X = N\theta_U)$, $P(X < N\theta_U)$, and $P(X > N\theta_U)$ and the truncated expectations $E\left(X1_{\{X<N\theta_L\}}\right)$, $E\left(X1_{\{X>N\theta_L\}}\right)$, $E\left(X1_{\{X<N\theta_L\}}\right)$, and $E\left(X1_{\{X>N\theta_L\}}\right)$ are approximated by assuming the normal-approximate distribution of X, $N(Np, Np(1-p))$. Letting $\phi(\cdot)$ and $\Phi(\cdot)$ denote the standard normal PDF and CDF, respectively, and defining d_L and d_U as

$$d_L = \frac{N\theta_L - Np}{[Np(1-p)]^{\frac{1}{2}}}$$
$$d_U = \frac{N\theta_U - Np}{[Np(1-p)]^{\frac{1}{2}}}$$

the terms are computed as follows:

$$\begin{aligned}P(X = N\theta_L) &= 0 \\ P(X = N\theta_U) &= 0 \\ P(X < N\theta_L) &= \Phi(d_L) \\ P(X < N\theta_U) &= \Phi(d_U) \\ P(X > N\theta_L) &= 1 - \Phi(d_L) \\ P(X > N\theta_U) &= 1 - \Phi(d_U) \\ E\left(X1_{\{X<N\theta_L\}}\right) &= Np\Phi(d_L) - [Np(1-p)]^{\frac{1}{2}}\,\phi(d_L) \\ E\left(X1_{\{X<N\theta_U\}}\right) &= Np\Phi(d_U) - [Np(1-p)]^{\frac{1}{2}}\,\phi(d_U) \\ E\left(X1_{\{X>N\theta_L\}}\right) &= Np\left[1 - \Phi(d_L)\right] + [Np(1-p)]^{\frac{1}{2}}\,\phi(d_L) \\ E\left(X1_{\{X>N\theta_U\}}\right) &= Np\left[1 - \Phi(d_U)\right] + [Np(1-p)]^{\frac{1}{2}}\,\phi(d_U)\end{aligned}$$

The mean and variance of $Z_{cL}(X)$ and $Z_{cU}(X)$ are thus approximated by

$$\mu_L = k_L\left[2Np - 2N\theta_L + 2\Phi(d_L) - 1\right]$$
$$\mu_U = k_U\left[2Np - 2N\theta_U + 2\Phi(d_U) - 1\right]$$

and

$$\sigma_L^2 = 4k_L^2 \left[Np(1-p) + \Phi(d_L)\,(1-\Phi(d_L)) - 2\,(Np(1-p))^{\frac{1}{2}}\,\phi(d_L) \right]$$
$$\sigma_U^2 = 4k_U^2 \left[Np(1-p) + \Phi(d_U)\,(1-\Phi(d_U)) - 2\,(Np(1-p))^{\frac{1}{2}}\,\phi(d_U) \right]$$

The approximate power is computed as

$$\text{power} = \Phi\left(\frac{z_\alpha - \mu_U}{\sigma_U}\right) + \Phi\left(\frac{z_\alpha + \mu_L}{\sigma_L}\right) - 1$$

The approximate sample size is computed by numerically inverting the power formula.

z Equivalence Test for Binomial Proportion with Continuity Adjustment Using Sample Variance (TEST=EQUIV_ADJZ VAREST=SAMPLE)

The hypotheses for the equivalence test are

$$H_0{:}p < \theta_L \quad \text{or} \quad p > \theta_U$$
$$H_1{:}\theta_L \le p \le \theta_U$$

where θ_L and θ_U are the lower and upper equivalence bounds, respectively.

The analysis is the two one-sided tests (TOST) procedure as described in Chow, Shao, and Wang (2003) on p. 84.

Two different hypothesis tests are carried out:

$$H_{a0}{:}p < \theta_L$$
$$H_{a1}{:}p \ge \theta_L$$

and

$$H_{b0}{:}p > \theta_U$$
$$H_{b1}{:}p \le \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for p is contained completely within (θ_L, θ_U).

The test statistic for the test of H_{a0} versus H_{a1} is

$$Z_{csL}(X) = \frac{X - N\theta_L + 0.5(1_{\{X<N\theta_L\}}) - 0.5(1_{\{X>N\theta_L\}})}{[N\,\hat{p}(1-\hat{p})]^{\frac{1}{2}}}$$

where $\hat{p} = X/N$.

The test statistic for the test of H_{b0} versus H_{b1} is

$$Z_{csU}(X) = \frac{X - N\theta_U + 0.5(1_{\{X<N\theta_U\}}) - 0.5(1_{\{X>N\theta_U\}})}{[N\,\hat{p}(1-\hat{p})]^{\frac{1}{2}}}$$

For the METHOD=EXACT option, let C_U denote the critical value of the exact upper one-sided test of H_{a0} versus H_{a1} using $Z_{csL}(X)$. This critical value is computed in the section "z Test for Binomial Proportion with Continuity Adjustment Using Sample Variance (TEST=ADJZ VAREST=SAMPLE)" on page 8035. Similarly, let C_L denote the critical value of the exact lower one-sided test of H_{b0} versus H_{b1} using $Z_{csU}(X)$. Both of these tests are rejected if and only if $C_U \leq X \leq C_L$. Thus, the exact power of the equivalence test is

$$\begin{aligned}\text{power} &= P\left(C_U \leq X \leq C_L\right)\\ &= P\left(X \geq C_U\right) - P\left(X \geq C_L + 1\right)\end{aligned}$$

The probabilities are computed using Johnson and Kotz (1970, equation 3.20).

For the METHOD=NORMAL option, the test statistic $Z_{csL}(X)$ is assumed to have the normal distribution $N(\mu_L, \sigma_L^2)$, and $Z_{csU}(X)$ is assumed to have the normal distribution $N(\mu_U, \sigma_U^2)$, where μ_L, μ_U, σ_L^2 and σ_U^2 are derived as follows.

For convenience of notation, define

$$k = \frac{1}{2\sqrt{Np(1-p)}}$$

Then

$$\begin{aligned}E\left[Z_{csL}(X)\right] &\approx 2kNp - 2kN\theta_L + kP(X < N\theta_L) - kP(X > N\theta_L)\\ E\left[Z_{csU}(X)\right] &\approx 2kNp - 2kN\theta_U + kP(X < N\theta_U) - kP(X > N\theta_U)\end{aligned}$$

and

$$\begin{aligned}\mathrm{Var}\left[Z_{csL}(X)\right] \approx\ & 4k^2Np(1-p) + k^2\left[1 - P(X = N\theta_L)\right] - k^2\left[P(X < N\theta_L) - P(X > N\theta_L)\right]^2\\ &+ 4k^2\left[E\left(X1_{\{X<N\theta_L\}}\right) - E\left(X1_{\{X>N\theta_L\}}\right)\right] - 4k^2Np\left[P(X < N\theta_L) - P(X > N\theta_L)\right]\\ \mathrm{Var}\left[Z_{csU}(X)\right] \approx\ & 4k^2Np(1-p) + k^2\left[1 - P(X = N\theta_U)\right] - k^2\left[P(X < N\theta_U) - P(X > N\theta_U)\right]^2\\ &+ 4k^2\left[E\left(X1_{\{X<N\theta_U\}}\right) - E\left(X1_{\{X>N\theta_U\}}\right)\right] - 4k^2Np\left[P(X < N\theta_U) - P(X > N\theta_U)\right]\end{aligned}$$

The probabilities $P(X = N\theta_L)$, $P(X < N\theta_L)$, $P(X > N\theta_L)$, $P(X = N\theta_U)$, $P(X < N\theta_U)$, and $P(X > N\theta_U)$ and the truncated expectations $E\left(X1_{\{X<N\theta_L\}}\right)$, $E\left(X1_{\{X>N\theta_L\}}\right)$, $E\left(X1_{\{X<N\theta_L\}}\right)$, and $E\left(X1_{\{X>N\theta_L\}}\right)$ are approximated by assuming the normal-approximate distribution of X, $N(Np, Np(1-p))$. Letting $\phi(\cdot)$ and $\Phi(\cdot)$ denote the standard normal PDF and CDF, respectively, and defining d_L and d_U as

$$\begin{aligned}d_L &= \frac{N\theta_L - Np}{[Np(1-p)]^{\frac{1}{2}}}\\ d_U &= \frac{N\theta_U - Np}{[Np(1-p)]^{\frac{1}{2}}}\end{aligned}$$

the terms are computed as follows:

$$
\begin{aligned}
P(X = N\theta_L) &= 0 \\
P(X = N\theta_U) &= 0 \\
P(X < N\theta_L) &= \Phi(d_L) \\
P(X < N\theta_U) &= \Phi(d_U) \\
P(X > N\theta_L) &= 1 - \Phi(d_L) \\
P(X > N\theta_U) &= 1 - \Phi(d_U) \\
E\left(X1_{\{X<N\theta_L\}}\right) &= Np\Phi(d_L) - [Np(1-p)]^{\frac{1}{2}}\,\phi(d_L) \\
E\left(X1_{\{X<N\theta_U\}}\right) &= Np\Phi(d_U) - [Np(1-p)]^{\frac{1}{2}}\,\phi(d_U) \\
E\left(X1_{\{X>N\theta_L\}}\right) &= Np\,[1-\Phi(d_L)] + [Np(1-p)]^{\frac{1}{2}}\,\phi(d_L) \\
E\left(X1_{\{X>N\theta_U\}}\right) &= Np\,[1-\Phi(d_U)] + [Np(1-p)]^{\frac{1}{2}}\,\phi(d_U)
\end{aligned}
$$

The mean and variance of $Z_{csL}(X)$ and $Z_{csU}(X)$ are thus approximated by

$$
\begin{aligned}
\mu_L &= k\,[2Np - 2N\theta_L + 2\Phi(d_L) - 1] \\
\mu_U &= k\,[2Np - 2N\theta_U + 2\Phi(d_U) - 1]
\end{aligned}
$$

and

$$
\begin{aligned}
\sigma_L^2 &= 4k^2\left[Np(1-p) + \Phi(d_L)\,(1-\Phi(d_L)) - 2\,(Np(1-p))^{\frac{1}{2}}\,\phi(d_L)\right] \\
\sigma_U^2 &= 4k^2\left[Np(1-p) + \Phi(d_U)\,(1-\Phi(d_U)) - 2\,(Np(1-p))^{\frac{1}{2}}\,\phi(d_U)\right]
\end{aligned}
$$

The approximate power is computed as

$$
\text{power} = \Phi\left(\frac{z_\alpha - \mu_U}{\sigma_U}\right) + \Phi\left(\frac{z_\alpha + \mu_L}{\sigma_L}\right) - 1
$$

The approximate sample size is computed by numerically inverting the power formula.

Wilson Score Confidence Interval for Binomial Proportion (CI=WILSON)

The two-sided $100(1-\alpha)\%$ confidence interval for p is

$$
\frac{X + \frac{z_{1-\alpha/2}^2}{2}}{N + z_{1-\alpha/2}^2} \quad \pm \quad \frac{z_{1-\alpha/2}N^{\frac{1}{2}}}{N + z_{1-\alpha/2}^2}\left(\hat{p}(1-\hat{p}) + \frac{z_{1-\alpha/2}^2}{4N}\right)^{\frac{1}{2}}
$$

So the half-width for the two-sided $100(1-\alpha)\%$ confidence interval is

$$
\text{half-width} = \frac{z_{1-\alpha/2}N^{\frac{1}{2}}}{N + z_{1-\alpha/2}^2}\left(\hat{p}(1-\hat{p}) + \frac{z_{1-\alpha/2}^2}{4N}\right)^{\frac{1}{2}}
$$

Prob(Width) is calculated exactly by adding up the probabilities of observing each $X \in \{1, \ldots, N\}$ that produces a confidence interval whose half-width is at most a target value h:

$$\text{Prob(Width)} = \sum_{i=0}^{N} P(X = i) 1_{\text{half-width}<h}$$

For references and more details about this and all other confidence intervals associated with the CI= option, see "Binomial Proportion" on page 3263 in Chapter 47, "The FREQ Procedure."

Agresti-Coull "Add k Successes and Failures" Confidence Interval for Binomial Proportion (CI=AGRESTICOULL)

The two-sided $100(1-\alpha)\%$ confidence interval for p is

$$\frac{X + \frac{z_{1-\alpha/2}^2}{2}}{N + z_{1-\alpha/2}^2} \quad \pm \quad z_{1-\alpha/2} \left(\frac{\frac{X+\frac{z_{1-\alpha/2}^2}{2}}{N+z_{1-\alpha/2}^2} \left(1 - \frac{X+\frac{z_{1-\alpha/2}^2}{2}}{N+z_{1-\alpha/2}^2}\right)}{N + z_{1-\alpha/2}^2} \right)^{\frac{1}{2}}$$

So the half-width for the two-sided $100(1-\alpha)\%$ confidence interval is

$$\text{half-width} = z_{1-\alpha/2} \left(\frac{\frac{X+\frac{z_{1-\alpha/2}^2}{2}}{N+z_{1-\alpha/2}^2} \left(1 - \frac{X+\frac{z_{1-\alpha/2}^2}{2}}{N+z_{1-\alpha/2}^2}\right)}{N + z_{1-\alpha/2}^2} \right)^{\frac{1}{2}}$$

Prob(Width) is calculated exactly by adding up the probabilities of observing each $X \in \{1, \ldots, N\}$ that produces a confidence interval whose half-width is at most a target value h:

$$\text{Prob(Width)} = \sum_{i=0}^{N} P(X = i) 1_{\text{half-width}<h}$$

Jeffreys Confidence Interval for Binomial Proportion (CI=JEFFREYS)

The two-sided $100(1-\alpha)\%$ confidence interval for p is

$$[L_J(X), U_J(X)]$$

where

$$L_J(X) = \begin{cases} 0, & X = 0 \\ \text{Beta}_{\alpha/2;X+1/2,N-X+1/2}, & X > 0 \end{cases}$$

and

$$U_J(X) = \begin{cases} \text{Beta}_{1-\alpha/2;X+1/2,N-X+1/2}, & X < N \\ 1, & X = N \end{cases}$$

The half-width of this two-sided $100(1-\alpha)\%$ confidence interval is defined as half the width of the full interval:

$$\text{half-width} = \frac{1}{2}\left(U_J(X) - L_J(X)\right)$$

Prob(Width) is calculated exactly by adding up the probabilities of observing each $X \in \{1, \ldots, N\}$ that produces a confidence interval whose half-width is at most a target value h:

$$\text{Prob(Width)} = \sum_{i=0}^{N} P(X = i) 1_{\text{half-width}<h}$$

Exact Clopper-Pearson Confidence Interval for Binomial Proportion (CI=EXACT)

The two-sided $100(1-\alpha)\%$ confidence interval for p is

$$[L_E(X), U_E(X)]$$

where

$$L_E(X) = \begin{cases} 0, & X = 0 \\ \text{Beta}_{\alpha/2;X,N-X+1}, & X > 0 \end{cases}$$

and

$$U_E(X) = \begin{cases} \text{Beta}_{1-\alpha/2;X+1,N-X}, & X < N \\ 1, & X = N \end{cases}$$

The half-width of this two-sided $100(1-\alpha)\%$ confidence interval is defined as half the width of the full interval:

$$\text{half-width} = \frac{1}{2}\left(U_E(X) - L_E(X)\right)$$

Prob(Width) is calculated exactly by adding up the probabilities of observing each $X \in \{1, \ldots, N\}$ that produces a confidence interval whose half-width is at most a target value h:

$$\text{Prob(Width)} = \sum_{i=0}^{N} P(X = i) 1_{\text{half-width}<h}$$

Wald Confidence Interval for Binomial Proportion (CI=WALD)

The two-sided $100(1-\alpha)\%$ confidence interval for p is

$$\hat{p} \quad \pm \quad z_{1-\alpha/2}\left(\frac{\hat{p}(1-\hat{p})}{N}\right)^{\frac{1}{2}}$$

So the half-width for the two-sided $100(1-\alpha)\%$ confidence interval is

$$\text{half-width} = z_{1-\alpha/2}\left(\frac{\hat{p}(1-\hat{p})}{N}\right)^{\frac{1}{2}}$$

Prob(Width) is calculated exactly by adding up the probabilities of observing each $X \in \{1,\ldots,N\}$ that produces a confidence interval whose half-width is at most a target value h:

$$\text{Prob(Width)} = \sum_{i=0}^{N} P(X=i)1_{\text{half-width}<h}$$

Continuity-Corrected Wald Confidence Interval for Binomial Proportion (CI=WALD_CORRECT)

The two-sided $100(1-\alpha)\%$ confidence interval for p is

$$\hat{p} \quad \pm \quad \left[z_{1-\alpha/2}\left(\frac{\hat{p}(1-\hat{p})}{N}\right)^{\frac{1}{2}} + \frac{1}{2N}\right]$$

So the half-width for the two-sided $100(1-\alpha)\%$ confidence interval is

$$\text{half-width} = z_{1-\alpha/2}\left(\frac{\hat{p}(1-\hat{p})}{N}\right)^{\frac{1}{2}} + \frac{1}{2N}$$

Prob(Width) is calculated exactly by adding up the probabilities of observing each $X \in \{1,\ldots,N\}$ that produces a confidence interval whose half-width is at most a target value h:

$$\text{Prob(Width)} = \sum_{i=0}^{N} P(X=i)1_{\text{half-width}<h}$$

Analyses in the ONESAMPLEMEANS Statement

One-Sample t Test (TEST=T)

The hypotheses for the one-sample t test are

$$H_0{:}\mu = \mu_0$$
$$H_1{:}\begin{cases} \mu \neq \mu_0, & \text{two-sided} \\ \mu > \mu_0, & \text{upper one-sided} \\ \mu < \mu_0, & \text{lower one-sided} \end{cases}$$

The test assumes normally distributed data and requires $N \geq 2$. The test statistics are

$$t = N^{\frac{1}{2}}\left(\frac{\bar{x} - \mu_0}{s}\right) \quad \sim \quad t(N-1, \delta)$$
$$t^2 \sim F(1, N-1, \delta^2)$$

where $\bar{x}$ is the sample mean, s is the sample standard deviation, and

$$\delta = N^{\frac{1}{2}}\left(\frac{\mu - \mu_0}{\sigma}\right)$$

The test is

$$\text{Reject} \quad H_0 \quad \text{if} \begin{cases} t^2 \geq F_{1-\alpha}(1, N-1), & \text{two-sided} \\ t \geq t_{1-\alpha}(N-1), & \text{upper one-sided} \\ t \leq t_{\alpha}(N-1), & \text{lower one-sided} \end{cases}$$

Exact power computations for t tests are discussed in O'Brien and Muller (1993, Section 8.2), although not specifically for the one-sample case. The power is based on the noncentral t and F distributions:

$$\text{power} = \begin{cases} P\left(F(1, N-1, \delta^2) \geq F_{1-\alpha}(1, N-1)\right), & \text{two-sided} \\ P\left(t(N-1, \delta) \geq t_{1-\alpha}(N-1)\right), & \text{upper one-sided} \\ P\left(t(N-1, \delta) \leq t_{\alpha}(N-1)\right), & \text{lower one-sided} \end{cases}$$

Solutions for N, α, and δ are obtained by numerically inverting the power equation. Closed-form solutions for other parameters, in terms of δ, are as follows:

$$\mu = \delta\sigma N^{-\frac{1}{2}} + \mu_0$$
$$\sigma = \begin{cases} \delta^{-1} N^{\frac{1}{2}}(\mu - \mu_0), & |\delta| > 0 \\ \text{undefined}, & \text{otherwise} \end{cases}$$

One-Sample t Test with Lognormal Data (TEST=T DIST=LOGNORMAL)

The lognormal case is handled by reexpressing the analysis equivalently as a normality-based test on the log-transformed data, by using properties of the lognormal distribution as discussed in Johnson, Kotz, and

Balakrishnan (1994, Chapter 14). The approaches in the section "One-Sample *t* Test (TEST=T)" on page 8049 then apply.

In contrast to the usual *t* test on normal data, the hypotheses with lognormal data are defined in terms of geometric means rather than arithmetic means. This is because the transformation of a null arithmetic mean of lognormal data to the normal scale depends on the unknown coefficient of variation, resulting in an ill-defined hypothesis on the log-transformed data. Geometric means transform cleanly and are more natural for lognormal data.

The hypotheses for the one-sample *t* test with lognormal data are

$$H_0: \frac{\gamma}{\gamma_0} = 1$$
$$H_1: \begin{cases} \frac{\gamma}{\gamma_0} \neq 1, & \text{two-sided} \\ \frac{\gamma}{\gamma_0} > 1, & \text{upper one-sided} \\ \frac{\gamma}{\gamma_0} < 1, & \text{lower one-sided} \end{cases}$$

Let $\mu^\star$ and $\sigma^\star$ be the (arithmetic) mean and standard deviation of the normal distribution of the log-transformed data. The hypotheses can be rewritten as follows:

$$H_0: \mu^\star = \log(\gamma_0)$$
$$H_1: \begin{cases} \mu^\star \neq \log(\gamma_0), & \text{two-sided} \\ \mu^\star > \log(\gamma_0), & \text{upper one-sided} \\ \mu^\star < \log(\gamma_0), & \text{lower one-sided} \end{cases}$$

where $\mu^\star = \log(\gamma)$.

The test assumes lognormally distributed data and requires $N \geq 2$.

The power is

$$\text{power} = \begin{cases} P\left(F(1, N-1, \delta^2) \geq F_{1-\alpha}(1, N-1)\right), & \text{two-sided} \\ P\left(t(N-1, \delta) \geq t_{1-\alpha}(N-1)\right), & \text{upper one-sided} \\ P\left(t(N-1, \delta) \leq t_{\alpha}(N-1)\right), & \text{lower one-sided} \end{cases}$$

where

$$\delta = N^{\frac{1}{2}} \left(\frac{\mu^\star - \log(\gamma_0)}{\sigma^\star} \right)$$
$$\sigma^\star = \left[\log(\text{CV}^2 + 1)\right]^{\frac{1}{2}}$$

Equivalence Test for Mean of Normal Data (TEST=EQUIV DIST=NORMAL)

The hypotheses for the equivalence test are

$$H_0: \mu < \theta_L \quad \text{or} \quad \mu > \theta_U$$
$$H_1: \theta_L \leq \mu \leq \theta_U$$

The analysis is the two one-sided tests (TOST) procedure of Schuirmann (1987). Two different hypothesis tests are carried out:

$$H_{a0}{:}\mu < \theta_L$$
$$H_{a1}{:}\mu \geq \theta_L$$

and

$$H_{b0}{:}\mu > \theta_U$$
$$H_{b1}{:}\mu \leq \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for μ is contained completely within (θ_L, θ_U).

The test assumes normally distributed data and requires $N \geq 2$. Phillips (1990) derives an expression for the exact power assuming a two-sample balanced design; the results are easily adapted to a one-sample design:

$$\text{power} = Q_{N-1}\left((-t_{1-\alpha}(N-1)), \frac{\mu-\theta_U}{\sigma N^{-\frac{1}{2}}}; 0, \frac{(N-1)^{\frac{1}{2}}(\theta_U-\theta_L)}{2\sigma N^{-\frac{1}{2}}(t_{1-\alpha}(N-1))}\right) - Q_{N-1}\left((t_{1-\alpha}(N-1)), \frac{\mu-\theta_L}{\sigma N^{-\frac{1}{2}}}; 0, \frac{(N-1)^{\frac{1}{2}}(\theta_U-\theta_L)}{2\sigma N^{-\frac{1}{2}}(t_{1-\alpha}(N-1))}\right)$$

where $Q_{\cdot}(\cdot,\cdot;\cdot,\cdot)$ is Owen's Q function, defined in the section "Common Notation" on page 8019.

Equivalence Test for Mean of Lognormal Data (TEST=EQUIV DIST=LOGNORMAL)

The lognormal case is handled by reexpressing the analysis equivalently as a normality-based test on the log-transformed data, by using properties of the lognormal distribution as discussed in Johnson, Kotz, and Balakrishnan (1994, Chapter 14). The approaches in the section "Equivalence Test for Mean of Normal Data (TEST=EQUIV DIST=NORMAL)" on page 8050 then apply.

In contrast to the additive equivalence test on normal data, the hypotheses with lognormal data are defined in terms of geometric means rather than arithmetic means. This is because the transformation of an arithmetic mean of lognormal data to the normal scale depends on the unknown coefficient of variation, resulting in an ill-defined hypothesis on the log-transformed data. Geometric means transform cleanly and are more natural for lognormal data.

The hypotheses for the equivalence test are

$$H_0{:}\gamma \leq \theta_L \quad \text{or} \quad \gamma \geq \theta_U$$
$$H_1{:}\theta_L < \gamma < \theta_U$$

$$\text{where} \quad 0 < \theta_L < \theta_U$$

The analysis is the two one-sided tests (TOST) procedure of Schuirmann (1987) on the log-transformed data. Two different hypothesis tests are carried out:

$$H_{a0}{:}\gamma < \theta_L$$
$$H_{a1}{:}\gamma \geq \theta_L$$

and

$$H_{b0}{:}\gamma > \theta_U$$
$$H_{b1}{:}\gamma \leq \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for γ is contained completely within (θ_L, θ_U).

The test assumes lognormally distributed data and requires $N \geq 2$. Diletti, Hauschke, and Steinijans (1991) derive an expression for the exact power assuming a crossover design; the results are easily adapted to a one-sample design:

$$\text{power} = Q_{N-1}\left((-t_{1-\alpha}(N-1)), \frac{\log(\gamma) - \log(\theta_U)}{\sigma^\star N^{-\frac{1}{2}}}; 0, \frac{(N-1)^{\frac{1}{2}}(\log(\theta_U) - \log(\theta_L))}{2\sigma^\star N^{-\frac{1}{2}}(t_{1-\alpha}(N-1))}\right) -$$
$$Q_{N-1}\left((t_{1-\alpha}(N-1)), \frac{\log(\gamma) - \log(\theta_L)}{\sigma^\star N^{-\frac{1}{2}}}; 0, \frac{(N-1)^{\frac{1}{2}}(\log(\theta_U) - \log(\theta_L))}{2\sigma^\star N^{-\frac{1}{2}}(t_{1-\alpha}(N-1))}\right)$$

where

$$\sigma^\star = \left[\log(\mathrm{CV}^2 + 1)\right]^{\frac{1}{2}}$$

is the standard deviation of the log-transformed data, and $Q_\cdot(\cdot,\cdot;\cdot,\cdot)$ is Owen's Q function, defined in the section "Common Notation" on page 8019.

Confidence Interval for Mean (CI=T)

This analysis of precision applies to the standard t-based confidence interval:

$$\begin{array}{ll}
\left[\bar{x} - t_{1-\frac{\alpha}{2}}(N-1)\frac{s}{\sqrt{N}}, \quad \bar{x} + t_{1-\frac{\alpha}{2}}(N-1)\frac{s}{\sqrt{N}}\right], & \text{two-sided} \\
\left[\bar{x} - t_{1-\alpha}(N-1)\frac{s}{\sqrt{N}}, \quad \infty\right), & \text{upper one-sided} \\
\left(-\infty, \quad \bar{x} + t_{1-\alpha}(N-1)\frac{s}{\sqrt{N}}\right], & \text{lower one-sided}
\end{array}$$

where $\bar{x}$ is the sample mean and s is the sample standard deviation. The "half-width" is defined as the distance from the point estimate $\bar{x}$ to a finite endpoint,

$$\text{half-width} = \begin{cases} t_{1-\frac{\alpha}{2}}(N-1)\frac{s}{\sqrt{N}}, & \text{two-sided} \\ t_{1-\alpha}(N-1)\frac{s}{\sqrt{N}}, & \text{one-sided} \end{cases}$$

A "valid" conference interval captures the true mean. The exact probability of obtaining at most the target confidence interval half-width h, unconditional or conditional on validity, is given by Beal (1989):

$$\Pr(\text{half-width} \leq h) = \begin{cases} P\left(\chi^2(N-1) \leq \frac{h^2 N(N-1)}{\sigma^2(t^2_{1-\frac{\alpha}{2}}(N-1))}\right), & \text{two-sided} \\ P\left(\chi^2(N-1) \leq \frac{h^2 N(N-1)}{\sigma^2(t^2_{1-\alpha}(N-1))}\right), & \text{one-sided} \end{cases}$$

$$\Pr(\text{half-width} \leq h \mid \text{validity}) = \begin{cases} \left(\frac{1}{1-\alpha}\right) 2\left[Q_{N-1}\left((t_{1-\frac{\alpha}{2}}(N-1)), 0; 0, b_1\right) - Q_{N-1}(0,0;0,b_1)\right], & \text{two-sided} \\ \left(\frac{1}{1-\alpha}\right) Q_{N-1}\left((t_{1-\alpha}(N-1)), 0; 0, b_1\right), & \text{one-sided} \end{cases}$$

where

$$b_1 = \frac{h(N-1)^{\frac{1}{2}}}{\sigma(t_{1-\frac{\alpha}{c}}(N-1))N^{-\frac{1}{2}}}$$
$$c = \text{number of sides}$$

and $Q_{\cdot}(\cdot,\cdot;\cdot,\cdot)$ is Owen's Q function, defined in the section "Common Notation" on page 8019.

A "quality" confidence interval is both sufficiently narrow (half-width $\leq h$) and valid:

$$\begin{aligned}\Pr(\text{quality}) &= \Pr(\text{half-width} \leq h \text{ and validity}) \\ &= \Pr(\text{half-width} \leq h | \text{validity})(1-\alpha)\end{aligned}$$

Analyses in the ONEWAYANOVA Statement

One-Degree-of-Freedom Contrast (TEST=CONTRAST)

The hypotheses are

$$H_0{:}c_1\mu_1 + \cdots + c_G\mu_G = c_0$$
$$H_1{:}\begin{cases} c_1\mu_1 + \cdots + c_G\mu_G \neq c_0, & \text{two-sided} \\ c_1\mu_1 + \cdots + c_G\mu_G > c_0, & \text{upper one-sided} \\ c_1\mu_1 + \cdots + c_G\mu_G < c_0, & \text{lower one-sided} \end{cases}$$

where G is the number of groups, $\{c_1, \ldots, c_G\}$ are the contrast coefficients, and c_0 is the null contrast value.

The test is the usual F test for a contrast in one-way ANOVA. It assumes normal data with common group variances and requires $N \geq G+1$ and $n_i \geq 1$.

O'Brien and Muller (1993, Section 8.2.3.2) give the exact power as

$$\text{power} = \begin{cases} P\left(F(1, N-G, \delta^2) \geq F_{1-\alpha}(1, N-G)\right), & \text{two-sided} \\ P\left(t(N-G, \delta) \geq t_{1-\alpha}(N-G)\right), & \text{upper one-sided} \\ P\left(t(N-G, \delta) \leq t_{\alpha}(N-G)\right), & \text{lower one-sided} \end{cases}$$

where

$$\delta = N^{\frac{1}{2}}\left(\frac{\sum_{i=1}^{G} c_i\mu_i - c_0}{\sigma\left(\sum_{i=1}^{G}\frac{c_i^2}{w_i}\right)^{\frac{1}{2}}}\right)$$

Overall F Test (TEST=OVERALL)

The hypotheses are

$$H_0{:}\mu_1 = \mu_2 = \cdots = \mu_G$$
$$H_1{:}\mu_i \neq \mu_j \text{ for some } i, j$$

where G is the number of groups.

The test is the usual overall F test for equality of means in one-way ANOVA. It assumes normal data with common group variances and requires $N \geq G + 1$ and $n_i \geq 1$.

O'Brien and Muller (1993, Section 8.2.3.1) give the exact power as

$$\text{power} = P\left(F(G-1, N-G, \lambda) \geq F_{1-\alpha}(G-1, N-G)\right)$$

where the noncentrality is

$$\lambda = N\left(\frac{\sum_{i=1}^{G} w_i(\mu_i - \bar{\mu})^2}{\sigma^2}\right)$$

and

$$\bar{\mu} = \sum_{i=1}^{G} w_i \mu_i$$

Analyses in the PAIREDFREQ Statement

Overview of Conditional McNemar tests

Notation:

		Case		
		Failure	Success	
Control	Failure	n_{00}	n_{01}	$n_{0\cdot}$
	Success	n_{10}	n_{11}	$n_{1\cdot}$
		$n_{\cdot 0}$	$n_{\cdot 1}$	N

$$\begin{aligned}
n_{00} &= \#\{\text{control=failure, case=failure}\} \\
n_{01} &= \#\{\text{control=failure, case=success}\} \\
n_{10} &= \#\{\text{control=success, case=failure}\} \\
n_{11} &= \#\{\text{control=success, case=success}\} \\
N &= n_{00} + n_{01} + n_{10} + n_{11} \\
n_D &= n_{01} + n_{10} \equiv \#\text{discordant pairs} \\
\hat{\pi}_{ij} &= \frac{n_{ij}}{N} \\
\pi_{ij} &= \text{theoretical population value of } \hat{\pi}_{ij} \\
\pi_{1\cdot} &= \pi_{10} + \pi_{11} \\
\pi_{\cdot 1} &= \pi_{01} + \pi_{11} \\
\phi &= \mathrm{Corr}(\text{control observation, case observation}) \quad \text{(within a pair)} \\
\mathrm{DPR} &= \text{"discordant proportion ratio"} = \frac{\pi_{01}}{\pi_{10}} \\
\mathrm{DPR}_0 &= \text{null DPR}
\end{aligned}$$

Power formulas are given here in terms of the discordant proportions π_{10} and π_{01}. If the input is specified in terms of $\{\pi_{1\cdot}, \pi_{\cdot 1}, \phi\}$, then it can be converted into values for $\{\pi_{10}, \pi_{01}\}$ as follows:

$$\pi_{01} = \pi_{\cdot 1}(1 - \pi_{1\cdot}) - \phi((1 - \pi_{1\cdot})\pi_{1\cdot}(1 - \pi_{\cdot 1})\pi_{\cdot 1})^{\frac{1}{2}}$$
$$\pi_{10} = \pi_{01} + \pi_{1\cdot} - \pi_{\cdot 1}$$

All McNemar tests covered in PROC POWER are *conditional*, meaning that n_D is assumed fixed at its observed value.

For the usual $\mathrm{DPR}_0 = 1$, the hypotheses are

$$H_0{:}\pi_{\cdot 1} = \pi_{1\cdot}$$
$$H_1{:}\begin{cases} \pi_{\cdot 1} \neq \pi_{1\cdot}, & \text{two-sided} \\ \pi_{\cdot 1} > \pi_{1\cdot}, & \text{upper one-sided} \\ \pi_{\cdot 1} < \pi_{1\cdot}, & \text{lower one-sided} \end{cases}$$

The test statistic for both tests covered in PROC POWER (DIST=EXACT_COND and DIST=NORMAL) is the McNemar statistic Q_M, which has the following form when $\mathrm{DPR}_0 = 1$:

$$Q_{M_0} = \frac{(n_{01} - n_{10})^2}{n_{01} + n_{10}}$$

For the conditional McNemar tests, this is equivalent to the square of the $Z(X)$ statistic for the test of a single proportion (normal approximation to binomial), where the proportion is $\frac{\pi_{01}}{\pi_{01}+\pi_{10}}$, the null is 0.5, and "$N$" is n_D (see, for example, Schork and Williams 1980):

$$\begin{aligned} Z(X) &= \frac{n_{01} - n_D(0.5)}{[n_D 0.5(1-0.5)]^{\frac{1}{2}}} \quad \dot{\sim} \mathrm{N}\left(\frac{n_D^{\frac{1}{2}}(\frac{\pi_{01}}{\pi_{01}+\pi_{10}} - 0.5)}{[0.5(1-0.5)]^{\frac{1}{2}}}, \frac{\frac{\pi_{01}}{\pi_{01}+\pi_{10}}\left(1 - \frac{\pi_{01}}{\pi_{01}+\pi_{10}}\right)}{0.5(1-0.5)} \right) \\ &= \frac{n_{01} - (n_{01} + n_{10})(0.5)}{[(n_{01} + n_{10})0.5(1-0.5)]^{\frac{1}{2}}} \\ &= \frac{n_{01} - n_{10}}{[n_{01} + n_{10}]^{\frac{1}{2}}} \\ &= \sqrt{Q_{M_0}} \end{aligned}$$

This can be generalized to a custom null for $\frac{\pi_{01}}{\pi_{01}+\pi_{10}}$, which is equivalent to specifying a custom null DPR:

$$\left[\frac{\pi_{01}}{\pi_{01} + \pi_{10}}\right]_0 \equiv \left[\frac{1}{1 + \frac{1}{\frac{\pi_{01}}{\pi_{10}}}}\right]_0 \equiv \frac{1}{1 + \frac{1}{\mathrm{DPR}_0}}$$

So, a conditional McNemar test (asymptotic or exact) with a custom null is equivalent to the test of a single proportion $p_1 \equiv \frac{\pi_{01}}{\pi_{01}+\pi_{10}}$ with a null value $p_0 \equiv \frac{1}{1+\frac{1}{\mathrm{DPR}_0}}$, with a sample size of n_D:

$$H_0{:}p_1 = p_0$$
$$H_1{:}\begin{cases} p_1 \neq p_0, & \text{two-sided} \\ p_1 > p_0, & \text{one-sided U} \\ p_1 < p_0, & \text{one-sided L} \end{cases}$$

which is equivalent to

$$H_0{:}\mathrm{DPR} = \mathrm{DPR}_0$$
$$H_1{:}\begin{cases} \mathrm{DPR} \neq \mathrm{DPR}_0, & \text{two-sided} \\ \mathrm{DPR} > \mathrm{DPR}_0, & \text{one-sided U} \\ \mathrm{DPR} < \mathrm{DPR}_0, & \text{one-sided L} \end{cases}$$

The general form of the test statistic is thus

$$Q_M = \frac{(n_{01} - n_D p_0)^2}{n_D p_0 (1 - p_0)}$$

The two most common conditional McNemar tests assume either the exact conditional distribution of Q_M (covered by the DIST=EXACT_COND analysis) or a standard normal distribution for Q_M (covered by the DIST=NORMAL analysis).

McNemar Exact Conditional Test (TEST=MCNEMAR DIST=EXACT_COND)

For DIST=EXACT_COND, the power is calculated assuming that the test is conducted by using the exact conditional distribution of Q_M (conditional on n_D). The power is calculated by first computing the conditional power for each possible n_D. The unconditional power is computed as a weighted average over all possible outcomes of n_D:

$$\text{power} = \sum_{n_D=0}^{N} P(n_D) P(\text{Reject } p_1 = p_0 | n_D)$$

where $n_D \sim \mathrm{Bin}(\pi_{01} + \pi_{10}, N)$, and $P(\text{Reject } p_1 = p_0 | n_D)$ is calculated by using the exact method in the section "Exact Test of a Binomial Proportion (TEST=EXACT)" on page 8031.

The achieved significance level, reported as "Actual Alpha" in the analysis, is computed in the same way except by using the actual alpha of the one-sample test in place of its power:

$$\text{actual alpha} = \sum_{n_D=0}^{N} P(n_D) \alpha^{\star}(p_1, p_0 | n_D)$$

where $\alpha^{\star}(p_1, p_0 | n_D)$ is the actual alpha calculated by using the exact method in the section "Exact Test of a Binomial Proportion (TEST=EXACT)" on page 8031 with proportion p_1, null p_0, and sample size n_D.

McNemar Normal Approximation Test (TEST=MCNEMAR DIST=NORMAL)

For DIST=NORMAL, power is calculated assuming the test is conducted by using the normal-approximate distribution of Q_M (conditional on n_D).

For the METHOD=EXACT option, the power is calculated in the same way as described in the section "McNemar Exact Conditional Test (TEST=MCNEMAR DIST=EXACT_COND)" on page 8056, except that $P(\text{Reject } p_1 = p_0 | n_D)$ is calculated by using the exact method in the section "z Test for Binomial Proportion Using Null Variance (TEST=Z VAREST=NULL)" on page 8032. The achieved significance level is calculated in the same way as described at the end of the section "McNemar Exact Conditional Test (TEST=MCNEMAR DIST=EXACT_COND)" on page 8056.

For the METHOD=MIETTINEN option, approximate sample size for the one-sided cases is computed according to equation (5.6) in Miettinen (1968):

$$N = \frac{\left\{ z_{1-\alpha}(p_{10} + p_{01}) + z_{power}\left[(p_{10} + p_{01})^2 - \frac{1}{4}(p_{01} - p_{10})^2(3 + p_{10} + p_{01})\right]^{\frac{1}{2}} \right\}^2}{(p_{10} + p_{01})(p_{01} - p_{10})^2}$$

Approximate power for the one-sided cases is computed by solving the sample size equation for power, and approximate power for the two-sided case follows easily by summing the one-sided powers each at $\alpha/2$:

$$\text{power} = \begin{cases} \Phi\left(\frac{(p_{01}-p_{10})[N(p_{10}+p_{01})]^{\frac{1}{2}} - z_{1-\alpha}(p_{10}+p_{01})}{\left[(p_{10}+p_{01})^2 - \frac{1}{4}(p_{01}-p_{10})^2(3+p_{10}+p_{01})\right]^{\frac{1}{2}}}\right), & \text{upper one-sided} \\ \Phi\left(\frac{-(p_{01}-p_{10})[N(p_{10}+p_{01})]^{\frac{1}{2}} - z_{1-\alpha}(p_{10}+p_{01})}{\left[(p_{10}+p_{01})^2 - \frac{1}{4}(p_{01}-p_{10})^2(3+p_{10}+p_{01})\right]^{\frac{1}{2}}}\right), & \text{lower one-sided} \\ \Phi\left(\frac{(p_{01}-p_{10})[N(p_{10}+p_{01})]^{\frac{1}{2}} - z_{1-\frac{\alpha}{2}}(p_{10}+p_{01})}{\left[(p_{10}+p_{01})^2 - \frac{1}{4}(p_{01}-p_{10})^2(3+p_{10}+p_{01})\right]^{\frac{1}{2}}}\right) + & \\ \quad \Phi\left(\frac{-(p_{01}-p_{10})[N(p_{10}+p_{01})]^{\frac{1}{2}} - z_{1-\frac{\alpha}{2}}(p_{10}+p_{01})}{\left[(p_{10}+p_{01})^2 - \frac{1}{4}(p_{01}-p_{10})^2(3+p_{10}+p_{01})\right]^{\frac{1}{2}}}\right), & \text{two-sided} \end{cases}$$

The two-sided solution for N is obtained by numerically inverting the power equation.

In general, compared to METHOD=CONNOR, the METHOD=MIETTINEN approximation tends to be slightly more accurate but can be slightly anticonservative in the sense of underestimating sample size and overestimating power (Lachin 1992, p. 1250).

For the METHOD=CONNOR option, approximate sample size for the one-sided cases is computed according to equation (3) in Connor (1987):

$$N = \frac{\left\{ z_{1-\alpha}(p_{10} + p_{01})^{\frac{1}{2}} + z_{power}\left[p_{10} + p_{01} - (p_{01} - p_{10})^2\right]^{\frac{1}{2}} \right\}^2}{(p_{01} - p_{10})^2}$$

Approximate power for the one-sided cases is computed by solving the sample size equation for power, and approximate power for the two-sided case follows easily by summing the one-sided powers each at $\alpha/2$:

$$\text{power} = \begin{cases} \Phi\left(\frac{(p_{01}-p_{10})N^{\frac{1}{2}} - z_{1-\alpha}(p_{10}+p_{01})^{\frac{1}{2}}}{\left[p_{10}+p_{01}-(p_{01}-p_{10})^2\right]^{\frac{1}{2}}}\right), & \text{upper one-sided} \\ \Phi\left(\frac{-(p_{01}-p_{10})N^{\frac{1}{2}} - z_{1-\alpha}(p_{10}+p_{01})^{\frac{1}{2}}}{\left[p_{10}+p_{01}-(p_{01}-p_{10})^2\right]^{\frac{1}{2}}}\right), & \text{lower one-sided} \\ \Phi\left(\frac{(p_{01}-p_{10})N^{\frac{1}{2}} - z_{1-\frac{\alpha}{2}}(p_{10}+p_{01})^{\frac{1}{2}}}{\left[p_{10}+p_{01}-(p_{01}-p_{10})^2\right]^{\frac{1}{2}}}\right) + & \\ \quad \Phi\left(\frac{-(p_{01}-p_{10})N^{\frac{1}{2}} - z_{1-\frac{\alpha}{2}}(p_{10}+p_{01})^{\frac{1}{2}}}{\left[p_{10}+p_{01}-(p_{01}-p_{10})^2\right]^{\frac{1}{2}}}\right), & \text{two-sided} \end{cases}$$

The two-sided solution for N is obtained by numerically inverting the power equation.

In general, compared to METHOD=MIETTINEN, the METHOD=CONNOR approximation tends to be slightly less accurate but slightly conservative in the sense of overestimating sample size and underestimating power (Lachin 1992, p. 1250).

Analyses in the PAIREDMEANS Statement

Paired t Test (TEST=DIFF)

The hypotheses for the paired t test are

$$H_0{:}\mu_{\text{diff}} = \mu_0$$
$$H_1{:}\begin{cases} \mu_{\text{diff}} \neq \mu_0, & \text{two-sided} \\ \mu_{\text{diff}} > \mu_0, & \text{upper one-sided} \\ \mu_{\text{diff}} < \mu_0, & \text{lower one-sided} \end{cases}$$

The test assumes normally distributed data and requires $N \geq 2$. The test statistics are

$$t = N^{\frac{1}{2}}\left(\frac{\bar{d} - \mu_0}{s_d}\right) \quad \sim t(N-1, \delta)$$
$$t^2 \sim F(1, N-1, \delta^2)$$

where $\bar{d}$ and s_d are the sample mean and standard deviation of the differences and

$$\delta = N^{\frac{1}{2}}\left(\frac{\mu_{\text{diff}} - \mu_0}{\sigma_{\text{diff}}}\right)$$

and

$$\sigma_{\text{diff}} = \left(\sigma_1^2 + \sigma_2^2 - 2\rho\sigma_1\sigma_2\right)^{\frac{1}{2}}$$

The test is

$$\text{Reject} \quad H_0 \quad \text{if} \begin{cases} t^2 \geq F_{1-\alpha}(1, N-1), & \text{two-sided} \\ t \geq t_{1-\alpha}(N-1), & \text{upper one-sided} \\ t \leq t_{\alpha}(N-1), & \text{lower one-sided} \end{cases}$$

Exact power computations for t tests are given in O'Brien and Muller (1993, Section 8.2.2):

$$\text{power} = \begin{cases} P\left(F(1, N-1, \delta^2) \geq F_{1-\alpha}(1, N-1)\right), & \text{two-sided} \\ P\left(t(N-1, \delta) \geq t_{1-\alpha}(N-1)\right), & \text{upper one-sided} \\ P\left(t(N-1, \delta) \leq t_{\alpha}(N-1)\right), & \text{lower one-sided} \end{cases}$$

Paired t Test for Mean Ratio with Lognormal Data (TEST=RATIO)

The lognormal case is handled by reexpressing the analysis equivalently as a normality-based test on the log-transformed data, by using properties of the lognormal distribution as discussed in Johnson, Kotz, and Balakrishnan (1994, Chapter 14). The approaches in the section "Paired t Test (TEST=DIFF)" on page 8058 then apply.

In contrast to the usual t test on normal data, the hypotheses with lognormal data are defined in terms of geometric means rather than arithmetic means.

The hypotheses for the paired t test with lognormal pairs $\{Y_1, Y_2\}$ are

$$H_0: \frac{\gamma_2}{\gamma_1} = \gamma_0$$
$$H_1: \begin{cases} \frac{\gamma_2}{\gamma_1} \neq \gamma_0, & \text{two-sided} \\ \frac{\gamma_2}{\gamma_1} > \gamma_0, & \text{upper one-sided} \\ \frac{\gamma_2}{\gamma_1} < \gamma_0, & \text{lower one-sided} \end{cases}$$

Let $\mu_1^\star$, $\mu_2^\star$, $\sigma_1^\star$, $\sigma_2^\star$, and $\rho^\star$ be the (arithmetic) means, standard deviations, and correlation of the bivariate normal distribution of the log-transformed data $\{\log Y_1, \log Y_2\}$. The hypotheses can be rewritten as follows:

$$H_0: \mu_2^\star - \mu_1^\star = \log(\gamma_0)$$
$$H_1: \begin{cases} \mu_2^\star - \mu_1^\star \neq \log(\gamma_0), & \text{two-sided} \\ \mu_2^\star - \mu_1^\star > \log(\gamma_0), & \text{upper one-sided} \\ \mu_2^\star - \mu_1^\star < \log(\gamma_0), & \text{lower one-sided} \end{cases}$$

where

$$\mu_1^\star = \log \gamma_1$$
$$\mu_2^\star = \log \gamma_2$$
$$\sigma_1^\star = \left[\log(\mathrm{CV}_1^2 + 1)\right]^{\frac{1}{2}}$$
$$\sigma_2^\star = \left[\log(\mathrm{CV}_2^2 + 1)\right]^{\frac{1}{2}}$$
$$\rho^\star = \frac{\log\{\rho \mathrm{CV}_1 \mathrm{CV}_2 + 1\}}{\sigma_1^\star \sigma_2^\star}$$

and CV_1, CV_2, and ρ are the coefficients of variation and the correlation of the original untransformed pairs $\{Y_1, Y_2\}$. The conversion from ρ to $\rho^\star$ is given by equation (44.36) on page 27 of Kotz, Balakrishnan, and Johnson (2000) and due to Jones and Miller (1966).

The valid range of ρ is restricted to (ρ_L, ρ_U), where

$$\rho_L = \frac{\exp\left(-\left[\log(\mathrm{CV}_1^2 + 1)\log(\mathrm{CV}_2^2 + 1)\right]^{\frac{1}{2}}\right) - 1}{\mathrm{CV}_1 \mathrm{CV}_2}$$
$$\rho_U = \frac{\exp\left(\left[\log(\mathrm{CV}_1^2 + 1)\log(\mathrm{CV}_2^2 + 1)\right]^{\frac{1}{2}}\right) - 1}{\mathrm{CV}_1 \mathrm{CV}_2}$$

These bounds are computed from equation (44.36) on page 27 of Kotz, Balakrishnan, and Johnson (2000) by observing that ρ is a monotonically increasing function of $\rho^\star$ and plugging in the values $\rho^\star = -1$ and $\rho^\star = 1$. Note that when the coefficients of variation are equal ($\mathrm{CV}_1 = \mathrm{CV}_2 = \mathrm{CV}$), the bounds simplify to

$$\rho_L = \frac{-1}{\mathrm{CV}^2 + 1}$$
$$\rho_U = 1$$

The test assumes lognormally distributed data and requires $N \geq 2$. The power is

$$\text{power} = \begin{cases} P\left(F(1, N-1, \delta^2) \geq F_{1-\alpha}(1, N-1)\right), & \text{two-sided} \\ P\left(t(N-1, \delta) \geq t_{1-\alpha}(N-1)\right), & \text{upper one-sided} \\ P\left(t(N-1, \delta) \leq t_{\alpha}(N-1)\right), & \text{lower one-sided} \end{cases}$$

where

$$\delta = N^{\frac{1}{2}} \left(\frac{\mu_1^\star - \mu_2^\star - \log(\gamma_0)}{\sigma^\star} \right)$$

and

$$\sigma^\star = \left(\sigma_1^{\star 2} + \sigma_2^{\star 2} - 2\rho^\star \sigma_1^\star \sigma_2^\star\right)^{\frac{1}{2}}$$

Additive Equivalence Test for Mean Difference with Normal Data (TEST=EQUIV_DIFF)

The hypotheses for the equivalence test are

$$H_0{:}\mu_{\text{diff}} < \theta_L \quad \text{or} \quad \mu_{\text{diff}} > \theta_U$$
$$H_1{:}\theta_L \leq \mu_{\text{diff}} \leq \theta_U$$

The analysis is the two one-sided tests (TOST) procedure of Schuirmann (1987). Two different hypothesis tests are carried out:

$$H_{a0}{:}\mu_{\text{diff}} < \theta_L$$
$$H_{a1}{:}\mu_{\text{diff}} \geq \theta_L$$

and

$$H_{b0}{:}\mu_{\text{diff}} > \theta_U$$
$$H_{b1}{:}\mu_{\text{diff}} \leq \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1 - 2\,\alpha)\%$ confidence interval for μ_{diff} is contained completely within (θ_L, θ_U).

The test assumes normally distributed data and requires $N \geq 2$. Phillips (1990) derives an expression for the exact power assuming a two-sample balanced design; the results are easily adapted to a paired design:

$$\text{power} = Q_{N-1}\left((-t_{1-\alpha}(N-1)), \frac{\mu_{\text{diff}} - \theta_U}{\sigma_{\text{diff}} N^{-\frac{1}{2}}}; 0, \frac{(N-1)^{\frac{1}{2}}(\theta_U - \theta_L)}{2\sigma_{\text{diff}} N^{-\frac{1}{2}}(t_{1-\alpha}(N-1))} \right) -$$
$$Q_{N-1}\left((t_{1-\alpha}(N-1)), \frac{\mu_{\text{diff}} - \theta_L}{\sigma_{\text{diff}} N^{-\frac{1}{2}}}; 0, \frac{(N-1)^{\frac{1}{2}}(\theta_U - \theta_L)}{2\sigma_{\text{diff}} N^{-\frac{1}{2}}(t_{1-\alpha}(N-1))} \right)$$

where

$$\sigma_{\text{diff}} = \left(\sigma_1^2 + \sigma_2^2 - 2\rho\sigma_1\sigma_2\right)^{\frac{1}{2}}$$

and $Q_{\cdot}(\cdot, \cdot; \cdot, \cdot)$ is Owen's Q function, defined in the section "Common Notation" on page 8019.

Multiplicative Equivalence Test for Mean Ratio with Lognormal Data (TEST=EQUIV_RATIO)

The lognormal case is handled by reexpressing the analysis equivalently as a normality-based test on the log-transformed data, by using properties of the lognormal distribution as discussed in Johnson, Kotz, and Balakrishnan (1994, Chapter 14). The approaches in the section "Additive Equivalence Test for Mean Difference with Normal Data (TEST=EQUIV_DIFF)" on page 8060 then apply.

In contrast to the additive equivalence test on normal data, the hypotheses with lognormal data are defined in terms of geometric means rather than arithmetic means.

The hypotheses for the equivalence test are

$$H_0: \frac{\gamma_T}{\gamma_R} \le \theta_L \quad \text{or} \quad \frac{\gamma_T}{\gamma_R} \ge \theta_U$$
$$H_1: \theta_L < \frac{\gamma_T}{\gamma_R} < \theta_U$$
$$\text{where} \quad 0 < \theta_L < \theta_U$$

The analysis is the two one-sided tests (TOST) procedure of Schuirmann (1987) on the log-transformed data. Two different hypothesis tests are carried out:

$$H_{a0}: \frac{\gamma_T}{\gamma_R} < \theta_L$$
$$H_{a1}: \frac{\gamma_T}{\gamma_R} \ge \theta_L$$

and

$$H_{b0}: \frac{\gamma_T}{\gamma_R} > \theta_U$$
$$H_{b1}: \frac{\gamma_T}{\gamma_R} \le \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for γ_T/γ_R is contained completely within (θ_L, θ_U).

The test assumes lognormally distributed data and requires $N \ge 2$. Diletti, Hauschke, and Steinijans (1991) derive an expression for the exact power assuming a crossover design; the results are easily adapted to a paired design:

$$\text{power} = Q_{N-1}\left((-t_{1-\alpha}(N-1)), \frac{\log\left(\frac{\gamma_T}{\gamma_R}\right) - \log(\theta_U)}{\sigma^\star N^{-\frac{1}{2}}}; 0, \frac{(N-1)^{\frac{1}{2}}(\log(\theta_U) - \log(\theta_L))}{2\sigma^\star N^{-\frac{1}{2}}(t_{1-\alpha}(N-1))}\right) -$$
$$Q_{N-1}\left((t_{1-\alpha}(N-1)), \frac{\log\left(\frac{\gamma_T}{\gamma_R}\right) - \log(\theta_L)}{\sigma^\star N^{-\frac{1}{2}}}; 0, \frac{(N-1)^{\frac{1}{2}}(\log(\theta_U) - \log(\theta_L))}{2\sigma^\star N^{-\frac{1}{2}}(t_{1-\alpha}(N-1))}\right)$$

where $\sigma^\star$ is the standard deviation of the differences between the log-transformed pairs (in other words, the standard deviation of $\log(Y_T) - \log(Y_R)$, where Y_T and Y_R are observations from the treatment and

reference, respectively), computed as

$$\sigma^\star = \left(\sigma_R^{\star 2} + \sigma_T^{\star 2} - 2\rho^\star \sigma_R^\star \sigma_T^\star\right)^{\frac{1}{2}}$$
$$\sigma_R^\star = \left[\log(\mathrm{CV}_R^2 + 1)\right]^{\frac{1}{2}}$$
$$\sigma_T^\star = \left[\log(\mathrm{CV}_T^2 + 1)\right]^{\frac{1}{2}}$$
$$\rho^\star = \frac{\log\{\rho \mathrm{CV}_R \mathrm{CV}_T + 1\}}{\sigma_R^\star \sigma_T^\star}$$

where CV_R, CV_T, and ρ are the coefficients of variation and the correlation of the original untransformed pairs $\{Y_T, Y_R\}$, and $Q_\cdot(\cdot,\cdot;\cdot,\cdot)$ is Owen's Q function. The conversion from ρ to $\rho^\star$ is given by equation (44.36) on page 27 of Kotz, Balakrishnan, and Johnson (2000) and due to Jones and Miller (1966), and Owen's Q function is defined in the section "Common Notation" on page 8019.

The valid range of ρ is restricted to (ρ_L, ρ_U), where

$$\rho_L = \frac{\exp\left(-\left[\log(\mathrm{CV}_R^2 + 1)\log(\mathrm{CV}_T^2 + 1)\right]^{\frac{1}{2}}\right) - 1}{\mathrm{CV}_R \mathrm{CV}_T}$$
$$\rho_U = \frac{\exp\left(\left[\log(\mathrm{CV}_R^2 + 1)\log(\mathrm{CV}_T^2 + 1)\right]^{\frac{1}{2}}\right) - 1}{\mathrm{CV}_R \mathrm{CV}_T}$$

These bounds are computed from equation (44.36) on page 27 of Kotz, Balakrishnan, and Johnson (2000) by observing that ρ is a monotonically increasing function of $\rho^\star$ and plugging in the values $\rho^\star = -1$ and $\rho^\star = 1$. Note that when the coefficients of variation are equal ($\mathrm{CV}_R = \mathrm{CV}_T = \mathrm{CV}$), the bounds simplify to

$$\rho_L = \frac{-1}{\mathrm{CV}^2 + 1}$$
$$\rho_U = 1$$

Confidence Interval for Mean Difference (CI=DIFF)

This analysis of precision applies to the standard t-based confidence interval:

$$\begin{array}{ll} \left[\bar{d} - t_{1-\frac{\alpha}{2}}(N-1)\frac{s_d}{\sqrt{N}},\quad \bar{d} + t_{1-\frac{\alpha}{2}}(N-1)\frac{s_d}{\sqrt{N}}\right], & \text{two-sided} \\ \left[\bar{d} - t_{1-\alpha}(N-1)\frac{s_d}{\sqrt{N}},\quad \infty\right), & \text{upper one-sided} \\ \left(-\infty,\quad \bar{d} + t_{1-\alpha}(N-1)\frac{s_d}{\sqrt{N}}\right], & \text{lower one-sided} \end{array}$$

where $\bar{d}$ and s_d are the sample mean and standard deviation of the differences. The "half-width" is defined as the distance from the point estimate $\bar{d}$ to a finite endpoint,

$$\text{half-width} = \begin{cases} t_{1-\frac{\alpha}{2}}(N-1)\frac{s_d}{\sqrt{N}}, & \text{two-sided} \\ t_{1-\alpha}(N-1)\frac{s_d}{\sqrt{N}}, & \text{one-sided} \end{cases}$$

A "valid" conference interval captures the true mean difference. The exact probability of obtaining at most the target confidence interval half-width h, unconditional or conditional on validity, is given by Beal (1989):

$$\Pr(\text{half-width} \le h) = \begin{cases} P\left(\chi^2(N-1) \le \frac{h^2 N(N-1)}{\sigma^2_{\text{diff}}(t^2_{1-\frac{\alpha}{2}}(N-1))}\right), & \text{two-sided} \\ P\left(\chi^2(N-1) \le \frac{h^2 N(N-1)}{\sigma^2_{\text{diff}}(t^2_{1-\alpha}(N-1))}\right), & \text{one-sided} \end{cases}$$

$$\Pr(\text{half-width} \le h| \text{ validity}) = \begin{cases} \left(\frac{1}{1-\alpha}\right) 2\left[Q_{N-1}\left((t_{1-\frac{\alpha}{2}}(N-1)), 0; 0, b_1\right) - Q_{N-1}(0,0;0,b_1)\right], & \text{two-sided} \\ \left(\frac{1}{1-\alpha}\right) Q_{N-1}\left((t_{1-\alpha}(N-1)), 0; 0, b_1\right), & \text{one-sided} \end{cases}$$

where

$$\sigma_{\text{diff}} = \left(\sigma_1^2 + \sigma_2^2 - 2\rho\sigma_1\sigma_2\right)^{\frac{1}{2}}$$

$$b_1 = \frac{h(N-1)^{\frac{1}{2}}}{\sigma_{\text{diff}}(t_{1-\frac{\alpha}{c}}(N-1))N^{-\frac{1}{2}}}$$

$$c = \text{number of sides}$$

and $Q.(\cdot,\cdot;\cdot,\cdot)$ is Owen's Q function, defined in the section "Common Notation" on page 8019.

A "quality" confidence interval is both sufficiently narrow (half-width $\le h$) and valid:

$$\begin{aligned} \Pr(\text{quality}) &= \Pr(\text{half-width} \le h \text{and validity}) \\ &= \Pr(\text{half-width} \le h|\text{validity})(1-\alpha) \end{aligned}$$

Analyses in the TWOSAMPLEFREQ Statement

Overview of the 2×2 *Table*

Notation:

		Outcome		
		Failure	Success	
Group	1	$n_1 - x_1$	x_1	n_1
	2	$n_2 - x_2$	x_2	n_2
		$N - m$	m	N

$$x_1 = \#\text{successes in group 1}$$

$$x_2 = \#\text{successes in group 2}$$

$$m = x_1 + x_2 = \text{ total } \#\text{successes}$$

$$\hat{p}_1 = \frac{x_1}{n_1}$$

$$\hat{p}_2 = \frac{x_2}{n_2}$$

$$\hat{p} = \frac{m}{N} = w_1\hat{p}_1 + w_2\hat{p}_2$$

The hypotheses are

$$H_0{:}p_2 - p_1 = p_0$$
$$H_1{:}\begin{cases} p_2 - p_1 \neq p_0, & \text{two-sided} \\ p_2 - p_1 > p_0, & \text{upper one-sided} \\ p_2 - p_1 < p_0, & \text{lower one-sided} \end{cases}$$

where p_0 is constrained to be 0 for the likelihood ratio and Fisher's exact tests. If $p_0 < 0$ in an upper one-sided test or $p_0 > 0$ in a lower one-sided test, then the test is a noninferiority test. If $p_0 > 0$ in an upper one-sided test or $p_0 < 0$ in a lower one-sided test, then the test is a superiority test. Although p_0 is unconstrained for the Pearson chi-square test, $p_0 \neq 0$ is not recommended for that test. The Farrington-Manning score test is a better choice when $p_0 \neq 0$.

Internal calculations are performed in terms of p_1, p_2, and p_0. An input set consisting of OR, p_1, and OR_0 is transformed as follows:

$$\begin{aligned} p_2 &= \frac{(\mathrm{OR})p_1}{1 - p_1 + (\mathrm{OR})p_1} \\ p_{10} &= p_1 \\ p_{20} &= \frac{\mathrm{OR}_0 p_{10}}{1 - p_{10} + (\mathrm{OR}_0)p_{10}} \\ p_0 &= p_{20} - p_{10} \end{aligned}$$

An input set consisting of RR, p_1, and RR_0 is transformed as follows:

$$\begin{aligned} p_2 &= (\mathrm{RR})p_1 \\ p_{10} &= p_1 \\ p_{20} &= (\mathrm{RR}_0)p_{10} \\ p_0 &= p_{20} - p_{10} \end{aligned}$$

The transformation of either OR_0 or RR_0 to p_0 is not unique. The chosen parameterization fixes the null value p_{10} at the input value of p_1. Some values of OR_0 or RR_0 might lead to invalid values of p_0 ($p_0 \leq 0$ or $p_0 \geq 1$), in which case an "Invalid input" error occurs.

Farrington-Manning Score Test for Proportion Difference (TEST=FM)

The Farrington-Manning score test for proportion difference is based on equations (1), (2), and (12) in Farrington and Manning (1990). The test statistic, which is assumed to have a null distribution of $N(0, 1)$ under H_0, is

$$z_{\mathrm{FMD}} = \frac{\hat{p}_2 - \hat{p}_1 - p_0}{\left[\frac{\tilde{p}_1(1-\tilde{p}_1)}{n_1} + \frac{\tilde{p}_2(1-\tilde{p}_2)}{n_2}\right]^{\frac{1}{2}}} = [N w_1 w_2]^{\frac{1}{2}} \frac{\hat{p}_2 - \hat{p}_1 - p_0}{[w_2 \tilde{p}_1(1-\tilde{p}_1) + w_1 \tilde{p}_2(1-\tilde{p}_2)]^{\frac{1}{2}}}$$

where $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of the proportions under the restriction $\tilde{p}_2 - \tilde{p}_1 = p_0$.

Sample size for the one-sided cases is given by equations (4) and (12) in Farrington and Manning (1990). One-sided power is computed by inverting the sample size formula. Power for the two-sided case is computed

by adding the lower-sided and upper-sided powers, each evaluated at $\alpha/2$. Sample size for the two-sided case is obtained by numerically inverting the power formula,

$$\text{power} = \begin{cases} \Phi\left(\frac{(p_2-p_1-p_0)(Nw_1w_2)^{\frac{1}{2}}-z_{1-\alpha}[w_2\tilde{p}_1(1-\tilde{p}_1)+w_1\tilde{p}_2(1-\tilde{p}_2)]^{\frac{1}{2}}}{[w_2p_1(1-p_1)+w_1p_2(1-p_2)]^{\frac{1}{2}}}\right), & \text{upper one-sided} \\ \Phi\left(\frac{-(p_2-p_1-p_0)(Nw_1w_2)^{\frac{1}{2}}-z_{1-\alpha}[w_2\tilde{p}_1(1-\tilde{p}_1)+w_1\tilde{p}_2(1-\tilde{p}_2)]^{\frac{1}{2}}}{[w_2p_1(1-p_1)+w_1p_2(1-p_2)]^{\frac{1}{2}}}\right), & \text{lower one-sided} \\ \Phi\left(\frac{(p_2-p_1-p_0)(Nw_1w_2)^{\frac{1}{2}}-z_{1-\frac{\alpha}{2}}[w_2\tilde{p}_1(1-\tilde{p}_1)+w_1\tilde{p}_2(1-\tilde{p}_2)]^{\frac{1}{2}}}{[w_2p_1(1-p_1)+w_1p_2(1-p_2)]^{\frac{1}{2}}}\right) + & \\ \quad \Phi\left(\frac{-(p_2-p_1-p_0)(Nw_1w_2)^{\frac{1}{2}}-z_{1-\frac{\alpha}{2}}[w_2\tilde{p}_1(1-\tilde{p}_1)+w_1\tilde{p}_2(1-\tilde{p}_2)]^{\frac{1}{2}}}{[w_2p_1(1-p_1)+w_1p_2(1-p_2)]^{\frac{1}{2}}}\right), & \text{two-sided} \end{cases}$$

where

$$\begin{aligned} \tilde{p}_2 &= 2u\cos(w) - b/(3a) \\ \tilde{p}_1 &= \tilde{p}_2 - p_0 \\ w &= (\pi + \cos^{-1}(v/u^3))/3 \\ v &= b^3/(3a)^3 - bc/(6a^2) + d/(2a) \\ u &= \text{sign}(v)\sqrt{b^2/(3a)^2 - c/(3a)} \\ a &= 1 + w_1/w_2 \\ b &= -[1 + w_1/w_2 + p_2 + (w_1/w_2)p_1 + p_0(w_1/w_2 + 2)] \\ c &= p_0^2 + p_0(2p_2 + w_1/w_2 + 1) + p_2 + (w_1/w_2)p_1 \\ d &= -p_2p_0(1 + p_0) \end{aligned}$$

For the one-sided cases, a closed-form inversion of the power equation yields an approximate total sample size of

$$N = \frac{\left[z_{1-\alpha}\{w_2\tilde{p}_1(1-\tilde{p}_1) + w_1\tilde{p}_2(1-\tilde{p}_2)\}^{\frac{1}{2}} + z_{\text{power}}\{w_2p_1(1-p_1) + w_1p_2(1-p_2)\}^{\frac{1}{2}}\right]^2}{w_1w_2(p_2-p_1-p_0)^2}$$

For the two-sided case, the solution for N is obtained by numerically inverting the power equation.

Farrington-Manning Score Test for Relative Risk (TEST=FM_RR)

The Farrington-Manning score test is based on equations (5), (6), and (13) in Farrington and Manning (1990). The test statistic, which is assumed to have a null distribution of $N(0,1)$ under H_0, is

$$z_{\text{FMR}} = \frac{\hat{p}_2 - \text{RR}_0\hat{p}_1}{\left[\frac{\text{RR}_0^2\tilde{p}_1(1-\tilde{p}_1)}{n_1} + \frac{\tilde{p}_2(1-\tilde{p}_2)}{n_2}\right]^{\frac{1}{2}}} = [Nw_1w_2]^{\frac{1}{2}}\frac{\hat{p}_2 - \text{RR}_0\hat{p}_1}{\left[w_2\text{RR}_0^2\tilde{p}_1(1-\tilde{p}_1) + w_1\tilde{p}_2(1-\tilde{p}_2)\right]^{\frac{1}{2}}}$$

where $\tilde{p}_1$ and $\tilde{p}_2$ are the maximum likelihood estimates of the proportions under the restriction $\tilde{p}_2/\tilde{p}_1 = \text{RR}_0$.

Sample size for the one-sided cases is given by equations (8) and (13) in Farrington and Manning (1990). One-sided power is computed by inverting the sample size formula. Power for the two-sided case is computed

by adding the lower-sided and upper-sided powers, each evaluated at $\alpha/2$. Sample size for the two-sided case is obtained by numerically inverting the power formula,

$$\text{power} = \begin{cases} \Phi\left(\dfrac{(\tilde{p_2}-\mathrm{RR}_0\tilde{p_1})(Nw_1w_2)^{\frac{1}{2}}-z_{1-\alpha}\left[w_2\mathrm{RR}_0^2\tilde{p_1}(1-\tilde{p_1})+w_1\tilde{p_2}(1-\tilde{p_2})\right]^{\frac{1}{2}}}{\left[w_2\mathrm{RR}_0^2p_1(1-p_1)+w_1p_2(1-p_2)\right]^{\frac{1}{2}}}\right), & \text{upper one-sided} \\ \Phi\left(\dfrac{-(\tilde{p_2}-\mathrm{RR}_0\tilde{p_1})(Nw_1w_2)^{\frac{1}{2}}-z_{1-\alpha}\left[w_2\mathrm{RR}_0^2\tilde{p_1}(1-\tilde{p_1})+w_1\tilde{p_2}(1-\tilde{p_2})\right]^{\frac{1}{2}}}{\left[w_2\mathrm{RR}_0^2p_1(1-p_1)+w_1p_2(1-p_2)\right]^{\frac{1}{2}}}\right), & \text{lower one-sided} \\ \Phi\left(\dfrac{(\tilde{p_2}-\mathrm{RR}_0\tilde{p_1})(Nw_1w_2)^{\frac{1}{2}}-z_{1-\frac{\alpha}{2}}\left[w_2\mathrm{RR}_0^2\tilde{p_1}(1-\tilde{p_1})+w_1\tilde{p_2}(1-\tilde{p_2})\right]^{\frac{1}{2}}}{\left[w_2\mathrm{RR}_0^2p_1(1-p_1)+w_1p_2(1-p_2)\right]^{\frac{1}{2}}}\right) + & \\ \quad\Phi\left(\dfrac{-(\tilde{p_2}-\mathrm{RR}_0\tilde{p_1})(Nw_1w_2)^{\frac{1}{2}}-z_{1-\frac{\alpha}{2}}\left[w_2\mathrm{RR}_0^2\tilde{p_1}(1-\tilde{p_1})+w_1\tilde{p_2}(1-\tilde{p_2})\right]^{\frac{1}{2}}}{\left[w_2\mathrm{RR}_0^2p_1(1-p_1)+w_1p_2(1-p_2)\right]^{\frac{1}{2}}}\right), & \text{two-sided} \end{cases}$$

where

$$\begin{aligned} \tilde{p_2} &= \frac{-b-(b^2-4ac)^{\frac{1}{2}}}{2a} \\ \tilde{p_1} &= \tilde{p_2}/\mathrm{RR}_0 \\ a &= 1+w_1/w_2 \\ b &= -\left[\mathrm{RR}_0\left(1+(w_1/w_2)p_1\right)+p_2+w_1/w_2\right] \\ c &= \mathrm{RR}_0\left(p_2+(w_1/w_2)p_1\right) \end{aligned}$$

For the one-sided cases, a closed-form inversion of the power equation yields an approximate total sample size of

$$N = \frac{\left[z_{1-\alpha}\left\{w_2\mathrm{RR}_0^2\tilde{p_1}(1-\tilde{p_1})+w_1\tilde{p_2}(1-\tilde{p_2})\right\}^{\frac{1}{2}}+z_{\text{power}}\left\{w_2\mathrm{RR}_0^2p_1(1-p_1)+w_1p_2(1-p_2)\right\}^{\frac{1}{2}}\right]^2}{w_1w_2(p_2-\mathrm{RR}_0p_1)^2}$$

For the two-sided case, the solution for N is obtained by numerically inverting the power equation.

Pearson Chi-Square Test for Two Proportions (TEST=PCHI)

The usual Pearson chi-square test is unconditional. The test statistic

$$z_P = \frac{\hat{p_2}-\hat{p_1}-p_0}{\left[\hat{p}(1-\hat{p})\left(\frac{1}{n_1}+\frac{1}{n_2}\right)\right]^{\frac{1}{2}}} = [Nw_1w_2]^{\frac{1}{2}}\,\frac{\hat{p_2}-\hat{p_1}-p_0}{[\hat{p}(1-\hat{p})]^{\frac{1}{2}}}$$

is assumed to have a null distribution of $N(0,1)$.

Sample size for the one-sided cases is given by equation (4) in Fleiss, Tytun, and Ury (1980). One-sided power is computed as suggested by Diegert and Diegert (1981) by inverting the sample size formula. Power for the two-sided case is computed by adding the lower-sided and upper-sided powers each evaluated at $\alpha/2$. Sample size for the two-sided case is obtained by numerically inverting the power formula. A custom null value p_0 for the proportion difference $p_2 - p_1$ is also supported, but it is not recommended. If you are using

a nondefault null value, then the Farrington-Manning score test is a better choice.

$$\text{power} = \begin{cases} \Phi\left(\dfrac{(p_2-p_1-p_0)(Nw_1w_2)^{\frac{1}{2}}-z_{1-\alpha}[(w_1p_1+w_2p_2)(1-w_1p_1-w_2p_2)]^{\frac{1}{2}}}{[w_2p_1(1-p_1)+w_1p_2(1-p_2)]^{\frac{1}{2}}}\right), & \text{upper one-sided} \\ \Phi\left(\dfrac{-(p_2-p_1-p_0)(Nw_1w_2)^{\frac{1}{2}}-z_{1-\alpha}[(w_1p_1+w_2p_2)(1-w_1p_1-w_2p_2)]^{\frac{1}{2}}}{[w_2p_1(1-p_1)+w_1p_2(1-p_2)]^{\frac{1}{2}}}\right), & \text{lower one-sided} \\ \Phi\left(\dfrac{(p_2-p_1-p_0)(Nw_1w_2)^{\frac{1}{2}}-z_{1-\frac{\alpha}{2}}[(w_1p_1+w_2p_2)(1-w_1p_1-w_2p_2)]^{\frac{1}{2}}}{[w_2p_1(1-p_1)+w_1p_2(1-p_2)]^{\frac{1}{2}}}\right)+ & \\ \quad\Phi\left(\dfrac{-(p_2-p_1-p_0)(Nw_1w_2)^{\frac{1}{2}}-z_{1-\frac{\alpha}{2}}[(w_1p_1+w_2p_2)(1-w_1p_1-w_2p_2)]^{\frac{1}{2}}}{[w_2p_1(1-p_1)+w_1p_2(1-p_2)]^{\frac{1}{2}}}\right), & \text{two-sided} \end{cases}$$

For the one-sided cases, a closed-form inversion of the power equation yields an approximate total sample size

$$N = \frac{\left[z_{1-\alpha}\left\{(w_1p_1+w_2p_2)(1-w_1p_1-w_2p_2)\right\}^{\frac{1}{2}} + z_{\text{power}}\left\{w_2p_1(1-p_1)+w_1p_2(1-p_2)\right\}^{\frac{1}{2}}\right]^2}{w_1w_2(p_2-p_1-p_0)^2}$$

For the two-sided case, the solution for N is obtained by numerically inverting the power equation.

Likelihood Ratio Chi-Square Test for Two Proportions (TEST=LRCHI)

The usual likelihood ratio chi-square test is unconditional. The test statistic

$$z_{\text{LR}} = (-1_{\{p_2<p_1\}})\sqrt{2N\sum_{i=1}^{2}\left[w_i\,\hat{p}_i\log\left(\frac{\hat{p}_i}{\hat{p}}\right)+w_i(1-\hat{p}_i)\log\left(\frac{1-\hat{p}_i}{1-\hat{p}}\right)\right]}$$

is assumed to have a null distribution of $N(0,1)$ and an alternative distribution of $N(\delta,1)$, where

$$\delta = N^{\frac{1}{2}}(-1_{\{p_2<p_1\}})\sqrt{2\sum_{i=1}^{2}\left[w_i\,p_i\log\left(\frac{p_i}{w_1p_1+w_2p_2}\right)+w_i(1-p_i)\log\left(\frac{1-p_i}{1-(w_1p_1+w_2p_2)}\right)\right]}$$

The approximate power is

$$\text{power} = \begin{cases} \Phi\left(\delta - z_{1-\alpha}\right), & \text{upper one-sided} \\ \Phi\left(-\delta - z_{1-\alpha}\right), & \text{lower one-sided} \\ \Phi\left(\delta - z_{1-\frac{\alpha}{2}}\right)+\Phi\left(-\delta - z_{1-\frac{\alpha}{2}}\right), & \text{two-sided} \end{cases}$$

For the one-sided cases, a closed-form inversion of the power equation yield an approximate total sample size

$$N = \left(\frac{z_{\text{power}}+z_{1-\alpha}}{\delta}\right)^2$$

For the two-sided case, the solution for N is obtained by numerically inverting the power equation.

Fisher's Exact Conditional Test for Two Proportions (Test=FISHER)

Fisher's exact test is conditional on the observed total number of successes m. Power and sample size computations are based on a test with similar power properties, the continuity-adjusted arcsine test. The test

statistic

$$z_A = (4Nw_1w_2)^{\frac{1}{2}}\left[\arcsin\left(\left[\hat{p}_2 + \frac{1}{2Nw_2}(1_{\{\hat{p}_2<\hat{p}_1\}} - 1_{\{\hat{p}_2>\hat{p}_1\}})\right]^{\frac{1}{2}}\right)\right.$$
$$\left.-\arcsin\left(\left[\hat{p}_1 + \frac{1}{2Nw_1}(1_{\{\hat{p}_1<\hat{p}_2\}} - 1_{\{\hat{p}_1>\hat{p}_2\}})\right]^{\frac{1}{2}}\right)\right]$$

is assumed to have a null distribution of $N(0,1)$ and an alternative distribution of $N(\delta,1)$, where

$$\delta = (4Nw_1w_2)^{\frac{1}{2}}\left[\arcsin\left(\left[p_2 + \frac{1}{2Nw_2}(1_{\{p_2<p_1\}} - 1_{\{p_2>p_1\}})\right]^{\frac{1}{2}}\right)\right.$$
$$\left.-\arcsin\left(\left[p_1 + \frac{1}{2Nw_1}(1_{\{p_1<p_2\}} - 1_{\{p_1>p_2\}})\right]^{\frac{1}{2}}\right)\right]$$

The approximate power for the one-sided balanced case is given by Walters (1979) and is easily extended to the unbalanced and two-sided cases:

$$\text{power} = \begin{cases} \Phi\left(\delta - z_{1-\alpha}\right), & \text{upper one-sided} \\ \Phi\left(-\delta - z_{1-\alpha}\right), & \text{lower one-sided} \\ \Phi\left(\delta - z_{1-\frac{\alpha}{2}}\right) + \Phi\left(-\delta - z_{1-\frac{\alpha}{2}}\right), & \text{two-sided} \end{cases}$$

The approximation is valid only for $N \geq 1/(2w_1w_2|p_1 - p_2|)$.

Analyses in the TWOSAMPLEMEANS Statement

Two-Sample t Test Assuming Equal Variances (TEST=DIFF)

The hypotheses for the two-sample t test are

$$H_0{:}\mu_{\text{diff}} = \mu_0$$
$$H_1{:}\begin{cases} \mu_{\text{diff}} \neq \mu_0, & \text{two-sided} \\ \mu_{\text{diff}} > \mu_0, & \text{upper one-sided} \\ \mu_{\text{diff}} < \mu_0, & \text{lower one-sided} \end{cases}$$

The test assumes normally distributed data and common standard deviation per group, and it requires $N \geq 3$, $n_1 \geq 1$, and $n_2 \geq 1$. The test statistics are

$$t = N^{\frac{1}{2}}(w_1w_2)^{\frac{1}{2}}\left(\frac{\bar{x}_2 - \bar{x}_1 - \mu_0}{s_p}\right) \quad \sim \quad t(N-2,\delta)$$
$$t^2 \sim F(1, N-2, \delta^2)$$

where $\bar{x}_1$ and $\bar{x}_2$ are the sample means and s_p is the pooled standard deviation, and

$$\delta = N^{\frac{1}{2}}(w_1w_2)^{\frac{1}{2}}\left(\frac{\mu_{\text{diff}} - \mu_0}{\sigma}\right)$$

The test is

$$\text{Reject} \quad H_0 \quad \text{if} \begin{cases} t^2 \geq F_{1-\alpha}(1, N-2), & \text{two-sided} \\ t \geq t_{1-\alpha}(N-2), & \text{upper one-sided} \\ t \leq t_{\alpha}(N-2), & \text{lower one-sided} \end{cases}$$

Exact power computations for t tests are given in O'Brien and Muller (1993, Section 8.2.1):

$$\text{power} = \begin{cases} P\left(F(1, N-2, \delta^2) \geq F_{1-\alpha}(1, N-2)\right), & \text{two-sided} \\ P\left(t(N-2, \delta) \geq t_{1-\alpha}(N-2)\right), & \text{upper one-sided} \\ P\left(t(N-2, \delta) \leq t_{\alpha}(N-2)\right), & \text{lower one-sided} \end{cases}$$

Solutions for N, n_1, n_2, α, and δ are obtained by numerically inverting the power equation. Closed-form solutions for other parameters, in terms of δ, are as follows:

$$\mu_{\text{diff}} = \delta\sigma(Nw_1w_2)^{-\frac{1}{2}} + \mu_0$$

$$\mu_1 = \delta\sigma(Nw_1w_2)^{-\frac{1}{2}} + \mu_0 - \mu_2$$

$$\mu_2 = \delta\sigma(Nw_1w_2)^{-\frac{1}{2}} + \mu_0 - \mu_1$$

$$\sigma = \begin{cases} \delta^{-1}(Nw_1w_2)^{\frac{1}{2}}(\mu_{\text{diff}} - \mu_0), & |\delta| > 0 \\ \text{undefined}, & \text{otherwise} \end{cases}$$

$$w_1 = \begin{cases} \frac{1}{2} \pm \frac{1}{2}\left[1 - \frac{4\delta^2\sigma^2}{N(\mu_{\text{diff}}-\mu_0)^2}\right]^{\frac{1}{2}}, & 0 < |\delta| \leq \frac{1}{2}N^{\frac{1}{2}}\frac{|\mu_{\text{diff}}-\mu_0|}{\sigma} \\ \text{undefined}, & \text{otherwise} \end{cases}$$

$$w_2 = \begin{cases} \frac{1}{2} \pm \frac{1}{2}\left[1 - \frac{4\delta^2\sigma^2}{N(\mu_{\text{diff}}-\mu_0)^2}\right]^{\frac{1}{2}}, & 0 < |\delta| \leq \frac{1}{2}N^{\frac{1}{2}}\frac{|\mu_{\text{diff}}-\mu_0|}{\sigma} \\ \text{undefined}, & \text{otherwise} \end{cases}$$

Finally, here is a derivation of the solution for w_1:

Solve the δ equation for w_1 (which requires the quadratic formula). Then determine the range of δ given w_1:

$$\min_{w_1}(\delta) = \begin{cases} 0, & \text{when} \quad w_1 = 0 \quad \text{or} \quad 1, \quad \text{if} \quad (\mu_{\text{diff}} - \mu_0) \geq 0 \\ \frac{1}{2}N^{\frac{1}{2}}\frac{(\mu_{\text{diff}}-\mu_0)}{\sigma}, & \text{when} \quad w_1 = \frac{1}{2}, \quad \text{if} \quad (\mu_{\text{diff}} - \mu_0) < 0 \end{cases}$$

$$\max_{w_1}(\delta) = \begin{cases} 0, & \text{when} \quad w_1 = 0 \quad \text{or} \quad 1, \quad \text{if} \quad (\mu_{\text{diff}} - \mu_0) < 0 \\ \frac{1}{2}N^{\frac{1}{2}}\frac{(\mu_{\text{diff}}-\mu_0)}{\sigma}, & \text{when} \quad w_1 = \frac{1}{2}, \quad \text{if} \quad (\mu_{\text{diff}} - \mu_0) \geq 0 \end{cases}$$

This implies

$$|\delta| \leq \frac{1}{2}N^{\frac{1}{2}}\frac{|\mu_{\text{diff}} - \mu_0|}{\sigma}$$

Two-Sample Satterthwaite t Test Assuming Unequal Variances (TEST=DIFF_SATT)

The hypotheses for the two-sample Satterthwaite t test are

$$H_0{:}\mu_{\text{diff}} = \mu_0$$

$$H_1{:}\begin{cases} \mu_{\text{diff}} \neq \mu_0, & \text{two-sided} \\ \mu_{\text{diff}} > \mu_0, & \text{upper one-sided} \\ \mu_{\text{diff}} < \mu_0, & \text{lower one-sided} \end{cases}$$

The test assumes normally distributed data and requires $N \geq 3$, $n_1 \geq 1$, and $n_2 \geq 1$. The test statistics are

$$t = \frac{\bar{x}_2 - \bar{x}_1 - \mu_0}{\left[\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right]^{\frac{1}{2}}} = N^{\frac{1}{2}} \frac{\bar{x}_2 - \bar{x}_1 - \mu_0}{\left[\frac{s_1^2}{w_1} + \frac{s_2^2}{w_2}\right]^{\frac{1}{2}}}$$

$$F = t^2$$

where $\bar{x}_1$ and $\bar{x}_2$ are the sample means and s_1 and s_2 are the sample standard deviations.

DiSantostefano and Muller (1995, p. 585) state, the test is based on assuming that under H_0, F is distributed as $F(1, \nu)$, where ν is given by Satterthwaite's approximation (Satterthwaite 1946),

$$\nu = \frac{\left[\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}\right]^2}{\frac{\left[\frac{\sigma_1^2}{n_1}\right]^2}{n_1-1} + \frac{\left[\frac{\sigma_2^2}{n_2}\right]^2}{n_2-1}} = \frac{\left[\frac{\sigma_1^2}{w_1} + \frac{\sigma_2^2}{w_2}\right]^2}{\frac{\left[\frac{\sigma_1^2}{w_1}\right]^2}{Nw_1-1} + \frac{\left[\frac{\sigma_2^2}{w_2}\right]^2}{Nw_2-1}}$$

Since ν is unknown, in practice it must be replaced by an estimate

$$\hat{\nu} = \frac{\left[\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right]^2}{\frac{\left[\frac{s_1^2}{n_1}\right]^2}{n_1-1} + \frac{\left[\frac{s_2^2}{n_2}\right]^2}{n_2-1}} = \frac{\left[\frac{s_1^2}{w_1} + \frac{s_2^2}{w_2}\right]^2}{\frac{\left[\frac{s_1^2}{w_1}\right]^2}{Nw_1-1} + \frac{\left[\frac{s_2^2}{w_2}\right]^2}{Nw_2-1}}$$

So the test is

$$\text{Reject} \quad H_0 \quad \text{if} \begin{cases} F \geq F_{1-\alpha}(1, \hat{\nu}), & \text{two-sided} \\ t \geq t_{1-\alpha}(\hat{\nu}), & \text{upper one-sided} \\ t \leq t_{\alpha}(\hat{\nu}), & \text{lower one-sided} \end{cases}$$

Exact solutions for power for the two-sided and upper one-sided cases are given in Moser, Stevens, and Watts (1989). The lower one-sided case follows easily by using symmetry. The equations are as follows:

$$\text{power} = \begin{cases} \int_0^\infty P\left(F(1, N-2, \lambda) > \right. \\ \quad \left. h(u) F_{1-\alpha}(1, v(u)) | u\right) f(u) \mathrm{d}u, & \text{two-sided} \\ \int_0^\infty P\left(t(N-2, \lambda^{\frac{1}{2}}) > \right. \\ \quad \left. [h(u)]^{\frac{1}{2}} t_{1-\alpha}(v(u)) | u\right) f(u) \mathrm{d}u, & \text{upper one-sided} \\ \int_0^\infty P\left(t(N-2, \lambda^{\frac{1}{2}}) < \right. \\ \quad \left. [h(u)]^{\frac{1}{2}} t_{\alpha}(v(u)) | u\right) f(u) \mathrm{d}u, & \text{lower one-sided} \end{cases}$$

where

$$h(u) = \frac{\left(\frac{1}{n_1} + \frac{u}{n_2}\right)(n_1 + n_2 - 2)}{\left[(n_1 - 1) + (n_2 - 1)\frac{u\sigma_1^2}{\sigma_2^2}\right]\left(\frac{1}{n_1} + \frac{\sigma_2^2}{\sigma_1^2 n_2}\right)}$$

$$v(u) = \frac{\left(\frac{1}{n_1} + \frac{u}{n_2}\right)^2}{\frac{1}{n_1^2(n_1-1)} + \frac{u^2}{n_2^2(n_2-1)}}$$

$$\lambda = \frac{(\mu_{\text{diff}} - \mu_0)^2}{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$$

$$f(u) = \frac{\Gamma\left(\frac{n_1+n_2-2}{2}\right)}{\Gamma\left(\frac{n_1-1}{2}\right)\Gamma\left(\frac{n_2-1}{2}\right)} \left[\frac{\sigma_1^2(n_2-1)}{\sigma_2^2(n_1-1)}\right]^{\frac{n_2-1}{2}} u^{\frac{n_2-3}{2}} \left[1 + \left(\frac{n_2-1}{n_1-1}\right)\frac{u\sigma_1^2}{\sigma_2^2}\right]^{-\left(\frac{n_1+n_2-2}{2}\right)}$$

The density $f(u)$ is obtained from the fact that

$$\frac{u\sigma_1^2}{\sigma_2^2} \sim F(n_2 - 1, n_1 - 1)$$

Because the test is biased, the achieved significance level might differ from the nominal significance level. The actual alpha is computed in the same way as the power, except that the mean difference μ_{diff} is replaced by the null mean difference μ_0.

Two-Sample Pooled t Test of Mean Ratio with Lognormal Data (TEST=RATIO)

The lognormal case is handled by reexpressing the analysis equivalently as a normality-based test on the log-transformed data, by using properties of the lognormal distribution as discussed in Johnson, Kotz, and Balakrishnan (1994, Chapter 14). The approaches in the section "Two-Sample *t* Test Assuming Equal Variances (TEST=DIFF)" on page 8068 then apply.

In contrast to the usual *t* test on normal data, the hypotheses with lognormal data are defined in terms of geometric means rather than arithmetic means. The test assumes equal coefficients of variation in the two groups.

The hypotheses for the two-sample t test with lognormal data are

$$H_0: \frac{\gamma_2}{\gamma_1} = \gamma_0$$
$$H_1: \begin{cases} \frac{\gamma_2}{\gamma_1} \neq \gamma_0, & \text{two-sided} \\ \frac{\gamma_2}{\gamma_1} > \gamma_0, & \text{upper one-sided} \\ \frac{\gamma_2}{\gamma_1} < \gamma_0, & \text{lower one-sided} \end{cases}$$

Let $\mu_1^\star$, $\mu_2^\star$, and $\sigma^\star$ be the (arithmetic) means and common standard deviation of the corresponding normal distributions of the log-transformed data. The hypotheses can be rewritten as follows:

$$H_0: \mu_2^\star - \mu_1^\star = \log(\gamma_0)$$
$$H_1: \begin{cases} \mu_2^\star - \mu_1^\star \neq \log(\gamma_0), & \text{two-sided} \\ \mu_2^\star - \mu_1^\star > \log(\gamma_0), & \text{upper one-sided} \\ \mu_2^\star - \mu_1^\star < \log(\gamma_0), & \text{lower one-sided} \end{cases}$$

where

$$\mu_1^\star = \log \gamma_1$$
$$\mu_2^\star = \log \gamma_2$$

The test assumes lognormally distributed data and requires $N \geq 3$, $n_1 \geq 1$, and $n_2 \geq 1$.

The power is

$$\text{power} = \begin{cases} P\left(F(1, N-2, \delta^2) \geq F_{1-\alpha}(1, N-2)\right), & \text{two-sided} \\ P\left(t(N-2, \delta) \geq t_{1-\alpha}(N-2)\right), & \text{upper one-sided} \\ P\left(t(N-2, \delta) \leq t_{\alpha}(N-2)\right), & \text{lower one-sided} \end{cases}$$

where

$$\delta = N^{\frac{1}{2}}(w_1 w_2)^{\frac{1}{2}} \left(\frac{\mu_2^\star - \mu_1^\star - \log(\gamma_0)}{\sigma^\star} \right)$$
$$\sigma^\star = \left[\log(\mathrm{CV}^2 + 1)\right]^{\frac{1}{2}}$$

Additive Equivalence Test for Mean Difference with Normal Data (TEST=EQUIV_DIFF)

The hypotheses for the equivalence test are

$$H_0: \mu_{\text{diff}} < \theta_L \quad \text{or} \quad \mu_{\text{diff}} > \theta_U$$
$$H_1: \theta_L \leq \mu_{\text{diff}} \leq \theta_U$$

The analysis is the two one-sided tests (TOST) procedure of Schuirmann (1987). Two different hypothesis tests are carried out:

$$H_{a0}: \mu_{\text{diff}} < \theta_L$$
$$H_{a1}: \mu_{\text{diff}} \geq \theta_L$$

and

$$H_{b0}{:}\mu_{\text{diff}} > \theta_U$$
$$H_{b1}{:}\mu_{\text{diff}} \le \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for μ_{diff} is contained completely within (θ_L, θ_U).

The test assumes normally distributed data and requires $N \ge 3$, $n_1 \ge 1$, and $n_2 \ge 1$. Phillips (1990) derives an expression for the exact power assuming a balanced design; the results are easily adapted to an unbalanced design:

$$\text{power} = Q_{N-2}\left((-t_{1-\alpha}(N-2)), \frac{\mu_{\text{diff}} - \theta_U}{\sigma N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}}; 0, \frac{(N-2)^{\frac{1}{2}}(\theta_U - \theta_L)}{2\sigma N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}(t_{1-\alpha}(N-2))}\right) - Q_{N-2}\left((t_{1-\alpha}(N-2)), \frac{\mu_{\text{diff}} - \theta_L}{\sigma N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}}; 0, \frac{(N-2)^{\frac{1}{2}}(\theta_U - \theta_L)}{2\sigma N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}(t_{1-\alpha}(N-2))}\right)$$

where $Q_{\cdot}(\cdot,\cdot;\cdot,\cdot)$ is Owen's Q function, defined in the section "Common Notation" on page 8019.

Multiplicative Equivalence Test for Mean Ratio with Lognormal Data (TEST=EQUIV_RATIO)

The lognormal case is handled by reexpressing the analysis equivalently as a normality-based test on the log-transformed data, by using properties of the lognormal distribution as discussed in Johnson, Kotz, and Balakrishnan (1994, Chapter 14). The approaches in the section "Additive Equivalence Test for Mean Difference with Normal Data (TEST=EQUIV_DIFF)" on page 8072 then apply.

In contrast to the additive equivalence test on normal data, the hypotheses with lognormal data are defined in terms of geometric means rather than arithmetic means.

The hypotheses for the equivalence test are

$$H_0{:}\frac{\gamma_T}{\gamma_R} \le \theta_L \quad \text{or} \quad \frac{\gamma_T}{\gamma_R} \ge \theta_U$$
$$H_1{:}\theta_L < \frac{\gamma_T}{\gamma_R} < \theta_U$$
$$\text{where} \quad 0 < \theta_L < \theta_U$$

The analysis is the two one-sided tests (TOST) procedure of Schuirmann (1987) on the log-transformed data. Two different hypothesis tests are carried out:

$$H_{a0}{:}\frac{\gamma_T}{\gamma_R} < \theta_L$$
$$H_{a1}{:}\frac{\gamma_T}{\gamma_R} \ge \theta_L$$

and

$$H_{b0}{:}\frac{\gamma_T}{\gamma_R} > \theta_U$$
$$H_{b1}{:}\frac{\gamma_T}{\gamma_R} \le \theta_U$$

If H_{a0} is rejected in favor of H_{a1} *and* H_{b0} is rejected in favor of H_{b1}, then H_0 is rejected in favor of H_1. Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for γ_T/γ_R is contained completely within (θ_L, θ_U).

The test assumes lognormally distributed data and requires $N \geq 3$, $n_1 \geq 1$, and $n_2 \geq 1$. Diletti, Hauschke, and Steinijans (1991) derive an expression for the exact power assuming a crossover design; the results are easily adapted to an unbalanced two-sample design:

$$\text{power} = Q_{N-2}\left((-t_{1-\alpha}(N-2)), \frac{\log\left(\frac{\gamma_T}{\gamma_R}\right) - \log(\theta_U)}{\sigma^\star N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}}; 0, \frac{(N-2)^{\frac{1}{2}}(\log(\theta_U) - \log(\theta_L))}{2\sigma^\star N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}(t_{1-\alpha}(N-2))}\right) - Q_{N-2}\left((t_{1-\alpha}(N-2)), \frac{\log\left(\frac{\gamma_T}{\gamma_R}\right) - \log(\theta_L)}{\sigma^\star N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}}; 0, \frac{(N-2)^{\frac{1}{2}}(\log(\theta_U) - \log(\theta_L))}{2\sigma^\star N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}(t_{1-\alpha}(N-2))}\right)$$

where

$$\sigma^\star = \left[\log(\mathrm{CV}^2 + 1)\right]^{\frac{1}{2}}$$

is the (assumed common) standard deviation of the normal distribution of the log-transformed data, and $Q_\cdot(\cdot,\cdot;\cdot,\cdot)$ is Owen's Q function, defined in the section "Common Notation" on page 8019.

Confidence Interval for Mean Difference (CI=DIFF)

This analysis of precision applies to the standard t-based confidence interval:

$$\begin{array}{ll}
\left[(\bar{x}_2 - \bar{x}_1) - t_{1-\frac{\alpha}{2}}(N-2)\frac{s_p}{\sqrt{N w_1 w_2}},\right. & \\
\quad \left.(\bar{x}_2 - \bar{x}_1) + t_{1-\frac{\alpha}{2}}(N-2)\frac{s_p}{\sqrt{N w_1 w_2}}\right], & \text{two-sided} \\
\left[(\bar{x}_2 - \bar{x}_1) - t_{1-\alpha}(N-2)\frac{s_p}{\sqrt{N w_1 w_2}}, \quad \infty\right), & \text{upper one-sided} \\
\left(-\infty, \quad (\bar{x}_2 - \bar{x}_1) + t_{1-\alpha}(N-2)\frac{s_p}{\sqrt{N w_1 w_2}}\right], & \text{lower one-sided}
\end{array}$$

where $\bar{x}_1$ and $\bar{x}_2$ are the sample means and s_p is the pooled standard deviation. The "half-width" is defined as the distance from the point estimate $\bar{x}_2 - \bar{x}_1$ to a finite endpoint,

$$\text{half-width} = \begin{cases} t_{1-\frac{\alpha}{2}}(N-2)\frac{s_p}{\sqrt{N w_1 w_2}}, & \text{two-sided} \\ t_{1-\alpha}(N-2)\frac{s_p}{\sqrt{N w_1 w_2}}, & \text{one-sided} \end{cases}$$

A "valid" conference interval captures the true mean. The exact probability of obtaining at most the target confidence interval half-width h, unconditional or conditional on validity, is given by Beal (1989):

$$\Pr(\text{half-width} \leq h) = \begin{cases} P\left(\chi^2(N-2) \leq \frac{h^2 N(N-2)(w_1 w_2)}{\sigma^2(t^2_{1-\frac{\alpha}{2}}(N-2))}\right), & \text{two-sided} \\ P\left(\chi^2(N-2) \leq \frac{h^2 N(N-2)(w_1 w_2)}{\sigma^2(t^2_{1-\alpha}(N-2))}\right), & \text{one-sided} \end{cases}$$

$$\Pr(\text{half-width} \leq h \mid \text{validity}) = \begin{cases} \left(\frac{1}{1-\alpha}\right) 2\left[Q_{N-2}\left((t_{1-\frac{\alpha}{2}}(N-2)), 0;\right.\right. \\ \quad \left.\left. 0, b_2\right) - Q_{N-2}(0, 0; 0, b_2)\right], & \text{two-sided} \\ \left(\frac{1}{1-\alpha}\right) Q_{N-2}\left((t_{1-\alpha}(N-2)), 0; 0, b_2\right), & \text{one-sided} \end{cases}$$

where

$$b_2 = \frac{h(N-2)^{\frac{1}{2}}}{\sigma(t_{1-\frac{\alpha}{c}}(N-2))N^{-\frac{1}{2}}(w_1 w_2)^{-\frac{1}{2}}}$$
$$c = \text{number of sides}$$

and $Q.(\cdot,\cdot;\cdot,\cdot)$ is Owen's Q function, defined in the section "Common Notation" on page 8019.

A "quality" confidence interval is both sufficiently narrow (half-width $\leq h$) and valid:

$$\begin{aligned}\Pr(\text{quality}) &= \Pr(\text{half-width} \leq h \text{and validity}) \\ &= \Pr(\text{half-width} \leq h|\text{validity})(1-\alpha)\end{aligned}$$

Analyses in the TWOSAMPLESURVIVAL Statement

Rank Tests for Two Survival Curves (TEST=LOGRANK, TEST=GEHAN, TEST=TARONEWARE)

The method is from Lakatos (1988) and Cantor (1997, pp. 83–92).

Define the following notation:

$$\begin{aligned}
X_j(i) &= i\text{th input time point on survival curve for group } j \\
S_j(i) &= \text{input survivor function value that corresponds to } X_j(i) \\
h_j(t) &= \text{hazard rate for group } j \text{ at time } t \\
\Psi_j(t) &= \text{loss hazard rate for group } j \text{ at time } t \\
\lambda_j &= \text{exponential hazard rate for group } j \\
R &= \text{hazard ratio of group 2 to group 1} \equiv \text{(assumed constant) value of } \frac{h_2(t)}{h_1(t)} \\
m_j &= \text{median survival time for group } j \\
b &= \text{number of subintervals per time unit} \\
T &= \text{accrual time} \\
\tau &= \text{follow-up time after accrual} \\
L_j &= \text{exponential loss rate for group } j \\
XL_j &= \text{input time point on loss curve for group } j \\
SL_j &= \text{input survivor function value that corresponds to } XL_j \\
mL_j &= \text{median survival time for group } j \\
r_i &= \text{rank for } i\text{th time point}
\end{aligned}$$

Each survival curve can be specified in one of several ways.

- For exponential curves:
  - a single point $(X_j(1), S_j(1))$ on the curve
  - median survival time

  - hazard rate
  - hazard ratio (for curve 2, with respect to curve 1)

- For piecewise linear curves with proportional hazards:
  - a set of points $\{(X_1(1), S_1(1)), (X_1(2), S_1(2)), \ldots\}$ (for curve 1)
  - hazard ratio (for curve 2, with respect to curve 1)

- For arbitrary piecewise linear curves:
  - a set of points $\{(X_j(1), S_j(1)), (X_j(2), S_j(2)), \ldots\}$

A total of $M + 1$ evenly spaced time points $\{t_0 = 0, t_1, t_2, \ldots, t_M = T + \tau\}$ are used in calculations, where

$$M = \text{floor}\left((T + \tau)b\right)$$

The hazard function is calculated for each survival curve at each time point. For an exponential curve, the (constant) hazard is given by one of the following, depending on the input parameterization:

$$h_j(t) = \begin{cases} \lambda_j \\ \lambda_1 R \\ \frac{-\log(\frac{1}{2})}{m_j} \\ \frac{-\log(S_j(1))}{X_j(1)} \\ \frac{-\log(S_1(1))}{X_1(1)} R \end{cases}$$

For a piecewise linear curve, define the following additional notation:

$$t_i^- = \text{largest input time } X \text{ such that } X \le t_i$$
$$t_i^+ = \text{smallest input time } X \text{ such that } X > t_i$$

The hazard is computed by using linear interpolation as follows:

$$h_j(t_i) = \frac{S_j(t_i^-) - S_j(t_i^+)}{\left[S_j(t_i^+) - S_j(t_i^-)\right]\left[t_i - t_i^-\right] + S_j(t_i^-)\left[t_i^+ - t_i^-\right]}$$

With proportional hazards, the hazard rate of group 2's curve in terms of the hazard rate of group 1's curve is

$$h_2(t) = h_1(t)R$$

Hazard function values $\{\Psi_j(t_i)\}$ for the loss curves are computed in an analogous way from $\{L_j, XL_j, SL_j, mL_j\}$.

The expected number at risk $N_j(i)$ at time i in group j is calculated for each group and time points 0 through $M - 1$, as follows:

$$N_j(0) = Nw_j$$
$$N_j(i+1) = N_j(i)\left[1 - h_j(t_i)\left(\frac{1}{b}\right) - \Psi_j(t_i)\left(\frac{1}{b}\right) - \left(\frac{1}{b(T + \tau - t_i)}\right) 1_{\{t_i > \tau\}}\right]$$

Define θ_i as the ratio of hazards and ϕ_i as the ratio of expected numbers at risk for time t_i:

$$\theta_i = \frac{h_2(t_i)}{h_1(t_i)}$$
$$\phi_i = \frac{N_2(i)}{N_1(i)}$$

The expected number of deaths in each subinterval is calculated as follows:

$$D_i = [h_1(t_i)N_1(i) + h_2(t_i)N_2(i)]\left(\frac{1}{b}\right)$$

The rank values are calculated as follows according to which test statistic is used:

$$r_i = \begin{cases} 1, & \text{log-rank} \\ N_1(i) + N_2(i), & \text{Gehan} \\ \sqrt{N_1(i) + N_2(i)}, & \text{Tarone-Ware} \end{cases}$$

The distribution of the test statistic is approximated by $N(E, 1)$ where

$$E = \frac{\sum_{i=0}^{M-1} D_i r_i \left[\frac{\phi_i \theta_i}{1+\phi_i \theta_i} - \frac{\phi_i}{1+\phi_i}\right]}{\sqrt{\sum_{i=0}^{M-1} D_i r_i^2 \frac{\phi_i}{(1+\phi_i)^2}}}$$

Note that $N^{\frac{1}{2}}$ can be factored out of the mean E, and so it can be expressed equivalently as

$$E = N^{\frac{1}{2}} E^\star = N^{\frac{1}{2}} \left[\frac{\sum_{i=0}^{M-1} D_i^\star r_i^\star \left[\frac{\phi_i \theta_i}{1+\phi_i \theta_i} - \frac{\phi_i}{1+\phi_i}\right]}{\sqrt{\sum_{i=0}^{M-1} D_i^\star r_i^{\star 2} \frac{\phi_i}{(1+\phi_i)^2}}}\right]$$

where $E^\star$ is free of N and

$$D_i^\star = \left[h_1(t_i)N_1^\star(i) + h_2(t_i)N_2^\star(i)\right]\left(\frac{1}{b}\right)$$
$$r_i^\star = \begin{cases} 1, & \text{log-rank} \\ N_1^\star(i) + N_2^\star(i), & \text{Gehan} \\ \sqrt{N_1^\star(i) + N_2^\star(i)}, & \text{Tarone-Ware} \end{cases}$$
$$N_j^\star(0) = w_j$$
$$N_j^\star(i+1) = N_j^\star(i)\left[1 - h_j(t_i)\left(\frac{1}{b}\right) - \Psi_j(t_i)\left(\frac{1}{b}\right) - \left(\frac{1}{b(T+\tau-t_i)}\right) 1_{\{t_i > \tau\}}\right]$$

The approximate power is

$$\text{power} = \begin{cases} \Phi\left(-N^{\frac{1}{2}} E^\star - z_{1-\alpha}\right), & \text{upper one-sided} \\ \Phi\left(N^{\frac{1}{2}} E^\star - z_{1-\alpha}\right), & \text{lower one-sided} \\ \Phi\left(-N^{\frac{1}{2}} E^\star - z_{1-\frac{\alpha}{2}}\right) + \Phi\left(N^{\frac{1}{2}} E^\star - z_{1-\frac{\alpha}{2}}\right), & \text{two-sided} \end{cases}$$

Note that the upper and lower one-sided cases are expressed differently than in other analyses. This is because $E^\star > 0$ corresponds to a higher survival curve in group 1 and thus, by the convention used in PROC power for two-group analyses, the lower side.

For the one-sided cases, a closed-form inversion of the power equation yield an approximate total sample size

$$N = \left(\frac{z_{\text{power}} + z_{1-\alpha}}{E^\star} \right)^2$$

For the two-sided case, the solution for N is obtained by numerically inverting the power equation.

Accrual rates are converted to and from sample sizes according to the equation $a_j = n_j / T$, where a_j is the accrual rate for group j.

Expected numbers of events—that is, deaths, whether observed or censored—are converted to and from sample sizes according to the equation

$$e_j = \begin{cases} n_j \left[1 - S_j(\tau) \right], & T = 0 \\ n_j \left[1 - \frac{1}{T} \int_0^T S_j(T + \tau - t) dt \right], & T > 0 \end{cases}$$

where e_j is the expected number of events in group j. For an exponential curve, the equation simplifies to

$$e_j = \begin{cases} n_j \left[1 - \exp(-\lambda_j \tau) \right], & T = 0 \\ n_j \left[1 - \frac{1}{\lambda_j T} \left(\exp(-\lambda_j \tau) - \exp(-\lambda_j (T + \tau)) \right) \right], & T > 0 \end{cases}$$

For a piecewise linear curve, first define K_j as the number of time points in the following collection: τ, $T + \tau$, and input time points for group j strictly between τ and $T + \tau$. Denote the ordered set of these points as $\{u_{j1}, \ldots, u_{jK_j}\}$. The survival function values $S_j(\tau)$ and $S_j(T + \tau)$ are calculated by linear interpolation between adjacent input time points if they do not coincide with any input time points. Then the equation for a piecewise linear curve simplifies to

$$e_j = \begin{cases} n_j \left[1 - S_j(\tau) \right], & T = 0 \\ n_j \left[1 - \frac{1}{2T} \sum_{i=1}^{K_j - 1} \left(u_{j,i+1} - u_{ji} \right) \left(S_j(u_{ji}) + S_j(u_{j,i+1}) \right) \right], & T > 0 \end{cases}$$

Analyses in the TWOSAMPLEWILCOXON Statement

Wilcoxon-Mann-Whitney Test for Comparing Two Distributions (TEST=WMW)

The power approximation in this section is applicable to the Wilcoxon-Mann-Whitney (WMW) test as invoked with the WILCOXON option in the PROC NPAR1WAY statement of the NPAR1WAY procedure. The approximation is based on O'Brien and Castelloe (2006) and an estimator called $\widehat{\text{WMW}}_{\text{odds}}$. See O'Brien and Castelloe (2006) for a definition of $\widehat{\text{WMW}}_{\text{odds}}$, which need not be derived in detail here for purposes of explaining the power formula.

Let Y_1 and Y_2 be independent observations from any two distributions that you want to compare using the WMW test. For purposes of deriving the asymptotic distribution of $\widehat{\text{WMW}}_{\text{odds}}$ (and consequently the power computation as well), these distributions must be formulated as ordered categorical ("ordinal") distributions.

If a distribution is continuous, it can be discretized using a large number of categories with negligible loss of accuracy. Each nonordinal distribution is divided into b categories, where b is the value of the NBINS parameter, with breakpoints evenly spaced on the probability scale. That is, each bin contains an equal

probability $1/b$ for that distribution. Then the breakpoints across both distributions are pooled to form a collection of C bins (heretofore called "categories"), and the probabilities of bin membership for each distribution are recalculated. The motivation for this method of binning is to avoid degenerate representations of the distributions—that is, small handfuls of large probabilities among mostly empty bins—as can be caused by something like an evenly spaced grid across raw values rather than probabilities.

After the discretization process just mentioned, there are now two ordinal distributions, each with a set of probabilities across a common set of C ordered categories. For simplicity of notation, assume (without loss of generality) the response values to be $1, \ldots, C$. Represent the conditional probabilities as

$$\tilde{p}_{ij} = \text{Prob}\left(Y_i = j \mid \text{group} = i\right), i \in \{1, 2\} \quad \text{and} \quad j \in \{1, \ldots, C\}$$

and the group allocation weights as

$$w_i = \frac{n_i}{N} = \text{Prob}\left(\text{group} = i\right), \quad i \in \{1, 2\}$$

The joint probabilities can then be calculated simply as

$$p_{ij} = \text{Prob}\left(\text{group} = i, Y_i = j\right) = w_i \tilde{p}_{ij}, i \in \{1, 2\} \quad \text{and} \quad j \in \{1, \ldots, C\}$$

The next step in the power computation is to compute the probabilities that a randomly chosen pair of observations from the two groups is concordant, discordant, or tied. It is useful to define these probabilities as functions of the terms Rs_{ij} and Rd_{ij}, defined as follows, where Y is a random observation drawn from the joint distribution across groups and categories:

$$\begin{aligned}
Rs_{ij} &= \text{Prob}\left(Y \text{ is concordant with cell}(i, j)\right) + \frac{1}{2}\text{Prob}\left(Y \text{ is tied with cell}(i, j)\right) \\
&= \text{Prob}\left((\text{group} < i \text{ and } Y < j) \text{ or } (\text{group} > i \text{ and } Y > j)\right) + \\
&\quad \frac{1}{2}\text{Prob}\left(\text{group} \neq i \text{ and } Y = j\right) \\
&= \sum_{g=1}^{2} \sum_{c=1}^{C} w_g \tilde{p}_{gc} \left[\mathrm{I}_{(g-i)(c-j)>0} + \frac{1}{2}\mathrm{I}_{g \neq i, c=j}\right]
\end{aligned}$$

and

$$\begin{aligned}
Rd_{ij} &= \text{Prob}\left(Y \text{ is discordant with cell}(i, j)\right) + \frac{1}{2}\text{Prob}\left(Y \text{ is tied with cell}(i, j)\right) \\
&= \text{Prob}\left((\text{group} < i \text{ and } Y > j) \text{ or } (\text{group} > i \text{ and } Y < j)\right) + \\
&\quad \frac{1}{2}\text{Prob}\left(\text{group} \neq i \text{ and } Y = j\right) \\
&= \sum_{g=1}^{2} \sum_{c=1}^{C} w_g \tilde{p}_{gc} \left[\mathrm{I}_{(g-i)(c-j)<0} + \frac{1}{2}\mathrm{I}_{g \neq i, c=j}\right]
\end{aligned}$$

For an independent random draw Y_1, Y_2 from the two distributions,

$$\begin{aligned}
P_c &= \text{Prob}\left(Y_1, Y_2 \text{ concordant}\right) + \frac{1}{2}\text{Prob}\left(Y_1, Y_2 \text{ tied}\right) \\
&= \sum_{i=1}^{2} \sum_{j=1}^{C} w_i \tilde{p}_{ij} Rs_{ij}
\end{aligned}$$

and

$$\begin{aligned} P_d &= \text{Prob}\,(Y_1, Y_2 \text{ discordant}) + \frac{1}{2}\text{Prob}\,(Y_1, Y_2 \text{ tied}) \\ &= \sum_{i=1}^{2}\sum_{j=1}^{C} w_i\,\tilde{p}_{ij}\,Rd_{ij} \end{aligned}$$

Then

$$\text{WMW}_{\text{odds}} = \frac{P_c}{P_d}$$

Proceeding to compute the theoretical standard error associated with WMW_{odds} (that is, the population analogue to the sample standard error),

$$\text{SE}(\text{WMW}_{\text{odds}}) = \frac{2}{P_d}\left[\sum_{i=1}^{2}\sum_{j=1}^{C} w_i\,\tilde{p}_{ij}\left(\text{WMW}_{\text{odds}}Rd_{ij} - Rs_{ij}\right)^2/N\right]^{\frac{1}{2}}$$

Converting to the natural log scale and using the delta method,

$$\text{SE}(\log(\text{WMW}_{\text{odds}})) = \frac{\text{SE}(\text{WMW}_{\text{odds}})}{\text{WMW}_{\text{odds}}}$$

The next step is to produce a "smoothed" version of the $2 \times C$ cell probabilities that conforms to the null hypothesis of the Wilcoxon-Mann-Whitney test (in other words, independence in the $2 \times C$ contingency table of probabilities). Let $\text{SE}_{H_0}(\log(\text{WMW}_{\text{odds}}))$ denote the theoretical standard error of $\log(\text{WMW}_{\text{odds}})$ assuming H_0.

Finally, compute the power using the noncentral chi-square and normal distributions:

$$\text{power} = \begin{cases} P\left(Z \geq \frac{\text{SE}_{H_0}(\log(\text{WMW}_{\text{odds}}))}{\text{SE}(\log(\text{WMW}_{\text{odds}}))} z_{1-\alpha} - \delta^\star N^{\frac{1}{2}}\right), & \text{upper one-sided} \\ P\left(Z \leq \frac{\text{SE}_{H_0}(\log(\text{WMW}_{\text{odds}}))}{\text{SE}(\log(\text{WMW}_{\text{odds}}))} z_{\alpha} - \delta^\star N^{\frac{1}{2}}\right), & \text{lower one-sided} \\ P\left(\chi^2(1, (\delta^\star)^2 N) \geq \left[\frac{\text{SE}_{H_0}(\log(\text{WMW}_{\text{odds}}))}{\text{SE}(\log(\text{WMW}_{\text{odds}}))}\right]^2 \chi^2_{1-\alpha}(1)\right), & \text{two-sided} \end{cases}$$

where

$$\delta^\star = \frac{\log(\text{WMW}_{\text{odds}})}{N^{\frac{1}{2}}\text{SE}(\log(\text{WMW}_{\text{odds}}))}$$

is the primary noncentrality—that is, the "effect size" that quantifies how much the two conjectured distributions differ. Z is a standard normal random variable, $\chi^2(df, nc)$ is a noncentral χ^2 random variable with degrees of freedom df and noncentrality nc, and N is the total sample size.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is not enabled, then PROC POWER creates traditional graphics.

You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC POWER generates are listed in Table 96.37, along with the required statements and options.

Table 96.37 Graphs Produced by PROC POWER

ODS Graph Name	Plot Description	Option
PowerPlot	Plot with two of the following three parameters on the X and Y axes: power, sample size, and effect size	PLOT
PowerAbort	Empty plot that shows an error message when a plot could not be produced	PLOT

Examples: POWER Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 96.1: One-Way ANOVA

(View the complete code for this example (pwrex01.sas) in the example repository.)

This example deals with the same situation as in Example 55.1 of Chapter 55, "The GLMPOWER Procedure."

Hocking (1985, p. 109) describes a study of the effectiveness of electrolytes in reducing lactic acid buildup for long-distance runners. You are planning a similar study in which you will allocate five different fluids to runners on a 10-mile course and measure lactic acid buildup immediately after the run. The fluids consist of water and two commercial electrolyte drinks, EZDure and LactoZap, each prepared at two concentrations, low (EZD1 and LZ1) and high (EZD2 and LZ2).

You conjecture that the standard deviation of lactic acid measurements given any particular fluid is about 3.75, and that the expected lactic acid values will correspond roughly to those in Table 96.38. You are least familiar with the LZ1 drink and hence decide to consider a range of reasonable values for that mean.

Table 96.38 Mean Lactic Acid Buildup by Fluid

Water	EZD1	EZD2	LZ1	LZ2
35.6	33.7	30.2	29 or 28	25.9

You are interested in four different comparisons, shown in Table 96.39 with appropriate contrast coefficients.

Table 96.39 Planned Comparisons

	Contrast Coefficients				
Comparison	Water	EZD1	EZD2	LZ1	LZ2
Water versus electrolytes	4	–1	–1	–1	–1
EZD versus LZ	0	1	1	–1	–1
EZD1 versus EZD2	0	1	–1	0	0
LZ1 versus LZ2	0	0	0	1	–1

For each of these contrasts you want to determine the sample size required to achieve a power of 0.9 for detecting an effect with magnitude in accord with Table 96.38. You are not yet attempting to choose a single sample size for the study, but rather checking the range of sample sizes needed for individual contrasts. You plan to test each contrast at $\alpha = 0.025$. In the interests of reducing costs, you will provide twice as many runners with water as with any of the electrolytes; in other words, you will use a sample size weighting scheme of 2:1:1:1:1. Use the ONEWAYANOVA statement in the POWER procedure to compute the sample sizes.

The statements required to perform this analysis are as follows:

```
proc power;
   onewayanova
      groupmeans = 35.6 | 33.7 | 30.2 | 29 28 | 25.9
      stddev = 3.75
      groupweights = (2 1 1 1 1)
      alpha = 0.025
      ntotal = .
      power = 0.9
      contrast = (4 -1 -1 -1 -1) (0  1  1 -1 -1)
                 (0  1 -1  0  0) (0  0  0  1 -1);
run;
```

The NTOTAL= option with a missing value (.) indicates total sample size as the result parameter. The GROUPMEANS= option with values from Table 96.38 specifies your conjectures for the means. With only one mean varying (the LZ1 mean), the "crossed" notation is simpler, showing scenarios for each group mean, separated by vertical bars (|). For more information about crossed and matched notations for grouped values, see the section "Specifying Value Lists in Analysis Statements" on page 8012. The contrasts in Table 96.39 are specified with the CONTRAST= option, by using the "matched" notation with each contrast enclosed in parentheses. The STDDEV=, ALPHA=, and POWER= options specify the error standard deviation,

significance level, and power. The GROUPWEIGHTS= option specifies the weighting schemes. Default values for the NULLCONTRAST= and SIDES= options specify a two-sided *t* test of the contrast equal to 0. See Output 96.1.1 for the results.

Output 96.1.1 Sample Sizes for One-Way ANOVA Contrasts

The POWER Procedure
Single DF Contrast in One-Way ANOVA

Fixed Scenario Elements	
Method	Exact
Alpha	0.025
Standard Deviation	3.75
Group Weights	2 1 1 1 1
Nominal Power	0.9
Number of Sides	2
Null Contrast Value	0

Computed N Total

Index	Contrast					Means					Actual Power	N Total
1	4	-1	-1	-1	-1	35.6	33.7	30.2	29	25.9	0.947	30
2	4	-1	-1	-1	-1	35.6	33.7	30.2	28	25.9	0.901	24
3	0	1	1	-1	-1	35.6	33.7	30.2	29	25.9	0.929	60
4	0	1	1	-1	-1	35.6	33.7	30.2	28	25.9	0.922	48
5	0	1	-1	0	0	35.6	33.7	30.2	29	25.9	0.901	174
6	0	1	-1	0	0	35.6	33.7	30.2	28	25.9	0.901	174
7	0	0	0	1	-1	35.6	33.7	30.2	29	25.9	0.902	222
8	0	0	0	1	-1	35.6	33.7	30.2	28	25.9	0.902	480

The sample sizes in Output 96.1.1 range from 24 for the comparison of water versus electrolytes to 480 for the comparison of LZ1 versus LZ2, both assuming the smaller LZ1 mean. The sample size for the latter comparison is relatively large because the small mean difference of 28 – 25.9 = 2.1 is hard to detect.

The Nominal Power of 0.9 in the "Fixed Scenario Elements" table in Output 96.1.1 represents the input target power, and the Actual Power column in the "Computed N Total" table is the power at the sample size (N Total) adjusted to achieve the specified sample weighting. Note that all of the sample sizes are rounded up to multiples of 6 to preserve integer group sizes (since the group weights add up to 6). You can use the NFRACTIONAL option in the ONEWAYANOVA statement to compute raw fractional sample sizes.

Suppose you want to plot the required sample size for the range of power values from 0.5 to 0.95. First, define the analysis by specifying the same statements as before, but add the PLOTONLY option to the PROC POWER statement to disable the nongraphical results. Next, specify the PLOT statement with X=POWER to request a plot with power on the X axis. (The result parameter, here sample size, is always plotted on the other axis.) Use the MIN= and MAX= options in the PLOT statement to specify the power range. The following statements produce the plot shown in Output 96.1.2.

```
ods graphics on;

proc power plotonly;
   onewayanova
      groupmeans = 35.6 | 33.7 | 30.2 | 29 28 | 25.9
```

```
        stddev = 3.75
        groupweights = (2 1 1 1 1)
        alpha = 0.025
        ntotal = .
        power = 0.9
        contrast = (4 -1 -1 -1 -1) (0  1  1 -1 -1)
                   (0  1 -1  0  0) (0  0  0  1 -1);
    plot x=power min=.5 max=.95;
run;
```

Output 96.1.2 Plot of Sample Size versus Power for One-Way ANOVA Contrasts

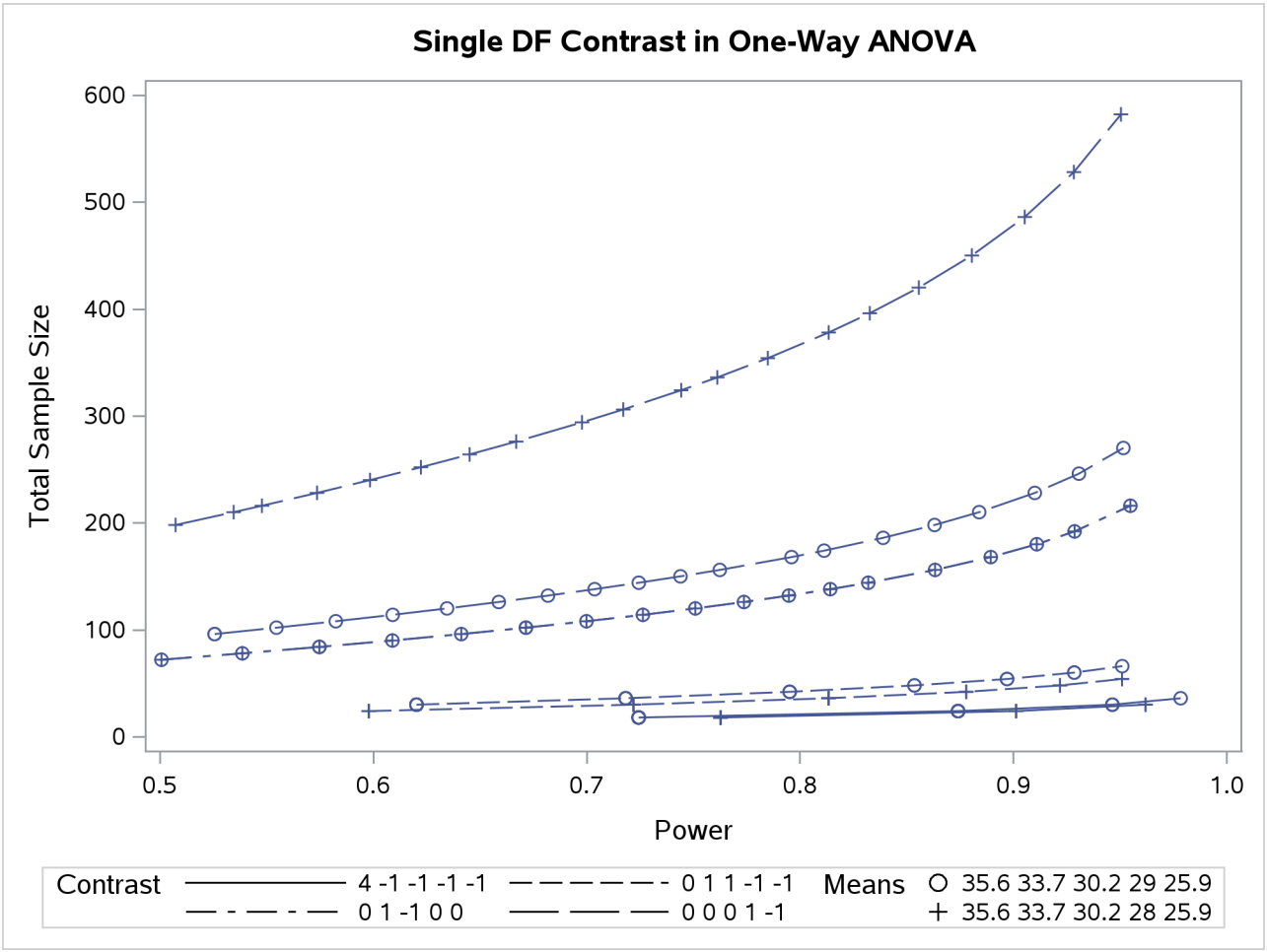

In Output 96.1.2, the line style identifies the contrast, and the plotting symbol identifies the group means scenario. The plot shows that the required sample size is highest for the (0 0 0 1 –1) contrast, which corresponds to the test of LZ1 versus LZ2 that was previously found to require the most resources, in either cell means scenario.

Note that some of the plotted points in Output 96.1.2 are unevenly spaced. This is because the plotted points are the *rounded* sample size results at their corresponding *actual* power levels. The range specified with the MIN= and MAX= values in the PLOT statement corresponds to *nominal* power levels. In some cases, actual power is substantially higher than nominal power. To obtain plots with evenly spaced points (but with *fractional* sample sizes at the computed points), you can use the NFRACTIONAL option in the analysis

statement preceding the PLOT statement.

Finally, suppose you want to plot the power for the range of sample sizes you will likely consider for the study (the range of 24 to 480 that achieves 0.9 power for different comparisons). In the ONEWAYANOVA statement, identify power as the result (POWER=.), and specify NTOTAL=24. The following statements produce the plot:

```
proc power plotonly;
   onewayanova
      groupmeans = 35.6 | 33.7 | 30.2 | 29 28 | 25.9
      stddev = 3.75
      groupweights = (2 1 1 1 1)
      alpha = 0.025
      ntotal = 24
      power = .
      contrast = (4 -1 -1 -1 -1) (0  1  1 -1 -1)
                 (0  1 -1  0  0) (0  0  0  1 -1);
   plot x=n min=24 max=480;
run;

ods graphics off;
```

The X=N option in the PLOT statement requests a plot with sample size on the X axis.

Note that the value specified with the NTOTAL=24 option is not used. It is overridden in the plot by the MIN= and MAX= options in the PLOT statement, and the PLOTONLY option in the PROC POWER statement disables nongraphical results. But the NTOTAL= option (along with a value) is still needed in the ONEWAYANOVA statement as a placeholder, to identify the desired parameterization for sample size.

Output 96.1.3 shows the resulting plot.

Output 96.1.3 Plot of Power versus Sample Size for One-Way ANOVA Contrasts

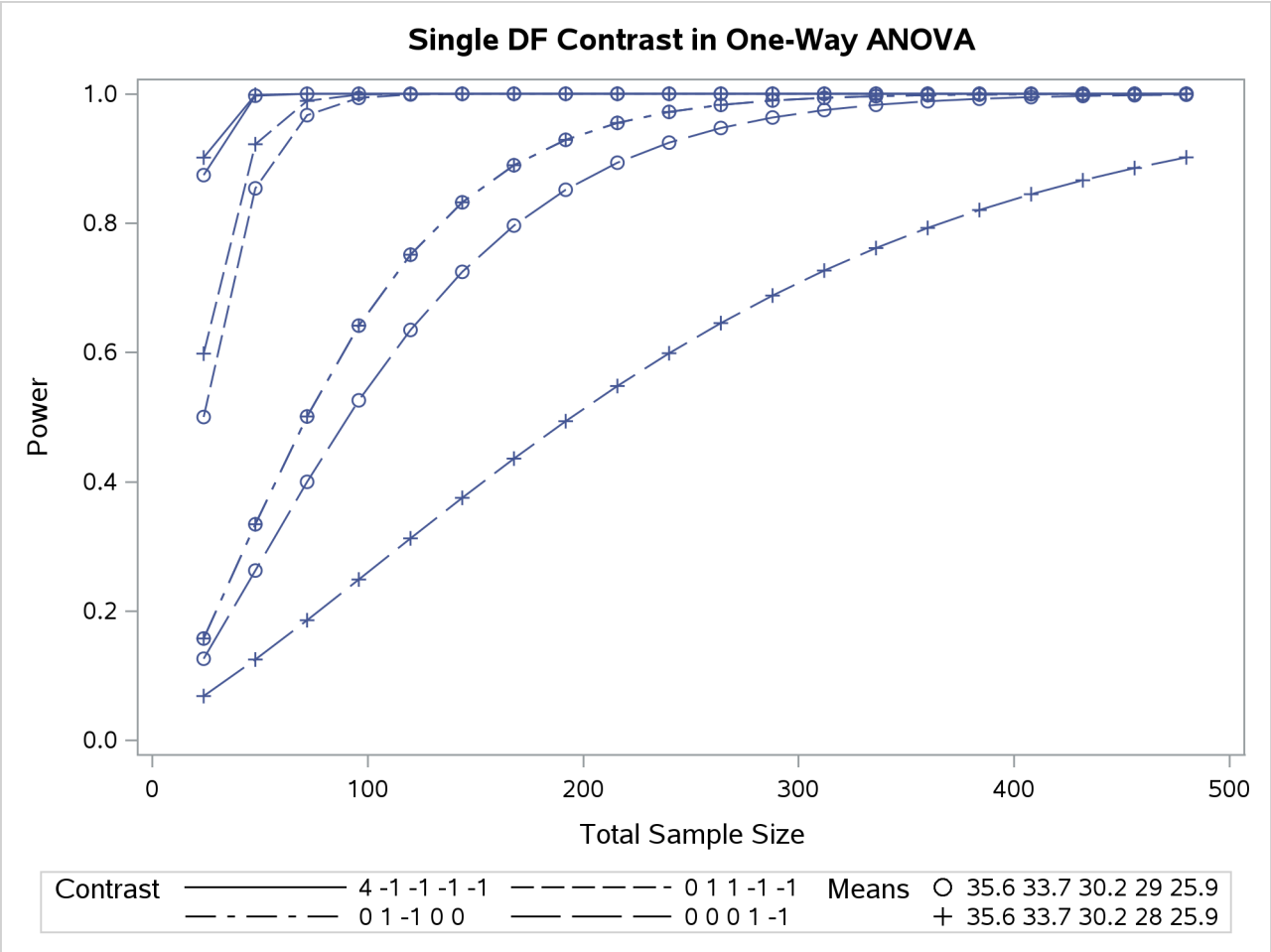

Although Output 96.1.2 and Output 96.1.3 surface essentially the same computations for practical power ranges, they each provide a different quick visual assessment. Output 96.1.2 reveals the range of required sample sizes for powers of interest, and Output 96.1.3 reveals the range of achieved powers for sample sizes of interest.

Example 96.2: The Sawtooth Power Function in Proportion Analyses

(View the complete code for this example (pwrex02.sas) in the example repository.)

For many common statistical analyses, the power curve is monotonically increasing: the more samples you take, the more power you achieve. However, in statistical analyses of discrete data, such as tests of proportions, the power curve is often nonmonotonic. A small increase in sample size can result in a *decrease* in power, a decrease that is sometimes substantial. The explanation is that the actual significance level (in other words, the achieved Type I error rate) for discrete tests strays below the target level and varies with sample size. The power loss from a decrease in the Type I error rate can outweigh the power gain from an increase in sample size. The example discussed here demonstrates this "sawtooth" phenomenon. For additional discussion on the topic, see Chernick and Liu (2002).

Suppose you have a new scheduling system for an airline, and you want to determine how many flights you must observe to have at least an 80% chance of establishing an improvement in the proportion of late arrivals on a specific travel route. You will use a one-sided exact binomial proportion test with a null proportion of 30%, the frequency of late arrivals under the previous scheduling system, and a nominal significance level of $\alpha = 0.05$. Well-supported predictions estimate the new late arrival rate to be about 20%, and you will base your sample size determination on this assumption.

The POWER procedure does not currently compute exact sample size directly for the exact binomial test. But you can get an initial estimate by computing the approximate sample size required for a z test. Use the ONESAMPLEFREQ statement in the POWER procedure with TEST=Z and METHOD=NORMAL to compute the approximate sample size to achieve a power of 0.8 by using the z test. The following statements perform the analysis:

```
proc power;
   onesamplefreq test=z method=normal
      sides          = 1
      alpha          = 0.05
      nullproportion = 0.3
      proportion     = 0.2
      ntotal         = .
      power          = 0.8;
run;
```

The NTOTAL= option with a missing value (.) indicates sample size as the result parameter. The SIDES=1 option specifies a one-sided test. The ALPHA=, NULLPROPORTION=, and POWER= options specify the significance level of 0.05, null value of 0.3, and target power of 0.8, respectively. The PROPORTION= option specifies your conjecture of 0.3 for the true proportion.

Output 96.2.1 Approximate Sample Size for z Test of a Proportion

The POWER Procedure
Z Test for Binomial Proportion

Fixed Scenario Elements	
Method	Normal approximation
Number of Sides	1
Null Proportion	0.3
Alpha	0.05
Binomial Proportion	0.2
Nominal Power	0.8
Variance Estimate	Null Variance

Computed N Total	
Actual Power	**N Total**
0.800	119

The results, shown in Output 96.2.1, indicate that you need to observe about $N = 119$ flights to have an 80% chance of rejecting the hypothesis of a late arrival proportion of 30% or higher, if the true proportion is 20%, by using the z test. A similar analysis (Output 96.2.2) reveals an approximate sample size of $N = 129$ for the z test with continuity correction, which is performed by using TEST=ADJZ:

```
proc power;
   onesamplefreq test=adjz method=normal
      sides          = 1
      alpha          = 0.05
      nullproportion = 0.3
      proportion     = 0.2
      ntotal         = .
      power          = 0.8;
run;
```

Output 96.2.2 Approximate Sample Size for z Test with Continuity Correction

The POWER Procedure
Z Test for Binomial Proportion with Continuity Adjustment

Fixed Scenario Elements	
Method	Normal approximation
Number of Sides	1
Null Proportion	0.3
Alpha	0.05
Binomial Proportion	0.2
Nominal Power	0.8
Variance Estimate	Null Variance

Computed N Total	
Actual Power	**N Total**
0.801	129

Based on the approximate sample size results, you decide to explore the power of the exact binomial test for sample sizes between 110 and 140. The following statements produce the plot:

```
ods graphics on;

proc power plotonly;
   onesamplefreq test=exact
      sides          = 1
      alpha          = 0.05
      nullproportion = 0.3
      proportion     = 0.2
      ntotal         = 119
      power          = .;
   plot x=n min=110 max=140 step=1
      yopts=(ref=.8) xopts=(ref=119 129);
run;
```

The TEST=EXACT option in the ONESAMPLEFREQ statement specifies the exact binomial test, and the missing value (.) for the POWER= option indicates power as the result parameter. The PLOTONLY option in the PROC POWER statement disables nongraphical output. The PLOT statement with X=N requests a plot with sample size on the X axis. The MIN= and MAX= options in the PLOT statement specify the sample size range. The YOPTS=(REF=) and XOPTS=(REF=) options add reference lines to highlight the approximate sample size results. The STEP=1 option produces a point at each integer sample size. The sample size value

specified with the NTOTAL= option in the ONESAMPLEFREQ statement is overridden by the MIN= and MAX= options in the PLOT statement. Output 96.2.3 shows the resulting plot.

Output 96.2.3 Plot of Power versus Sample Size for Exact Binomial Test

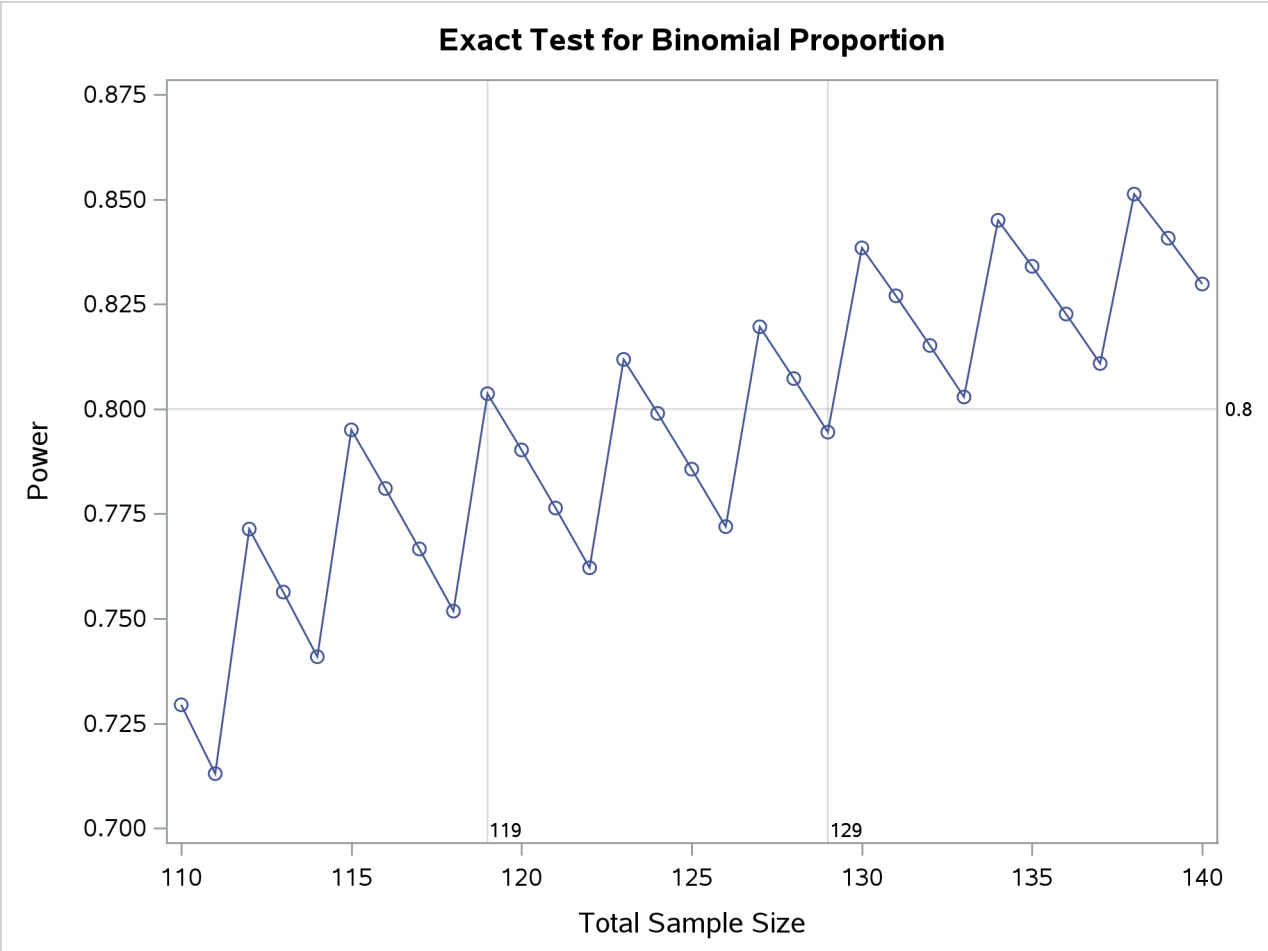

Note the sawtooth pattern in Output 96.2.3. Although the power surpasses the target level of 0.8 at N = 119, it decreases to 0.79 with N = 120 and further to 0.76 with N = 122 before rising again to 0.81 with N = 123. Not until N = 130 does the power stay above the 0.8 target. Thus, a more conservative sample size recommendation of 130 might be appropriate, depending on the precise goals of the sample size determination.

In addition to considering alternative sample sizes, you might also want to assess the sensitivity of the power to inaccuracies in assumptions about the true proportion. The following statements produce a plot including true proportion values of 0.18 and 0.22. They are identical to the previous statements except for the additional true proportion values specified with the PROPORTION= option in the ONESAMPLEFREQ statement.

```
proc power plotonly;
   onesamplefreq test=exact
      sides          = 1
      alpha          = 0.05
      nullproportion = 0.3
      proportion     = 0.18 0.2 0.22
```

```
      ntotal          = 119
      power           = .;
   plot x=n min=110 max=140 step=1
      yopts=(ref=.8) xopts=(ref=119 129);
run;
```

Output 96.2.4 shows the resulting plot.

Output 96.2.4 Plot for Assessing Sensitivity to True Proportion Value

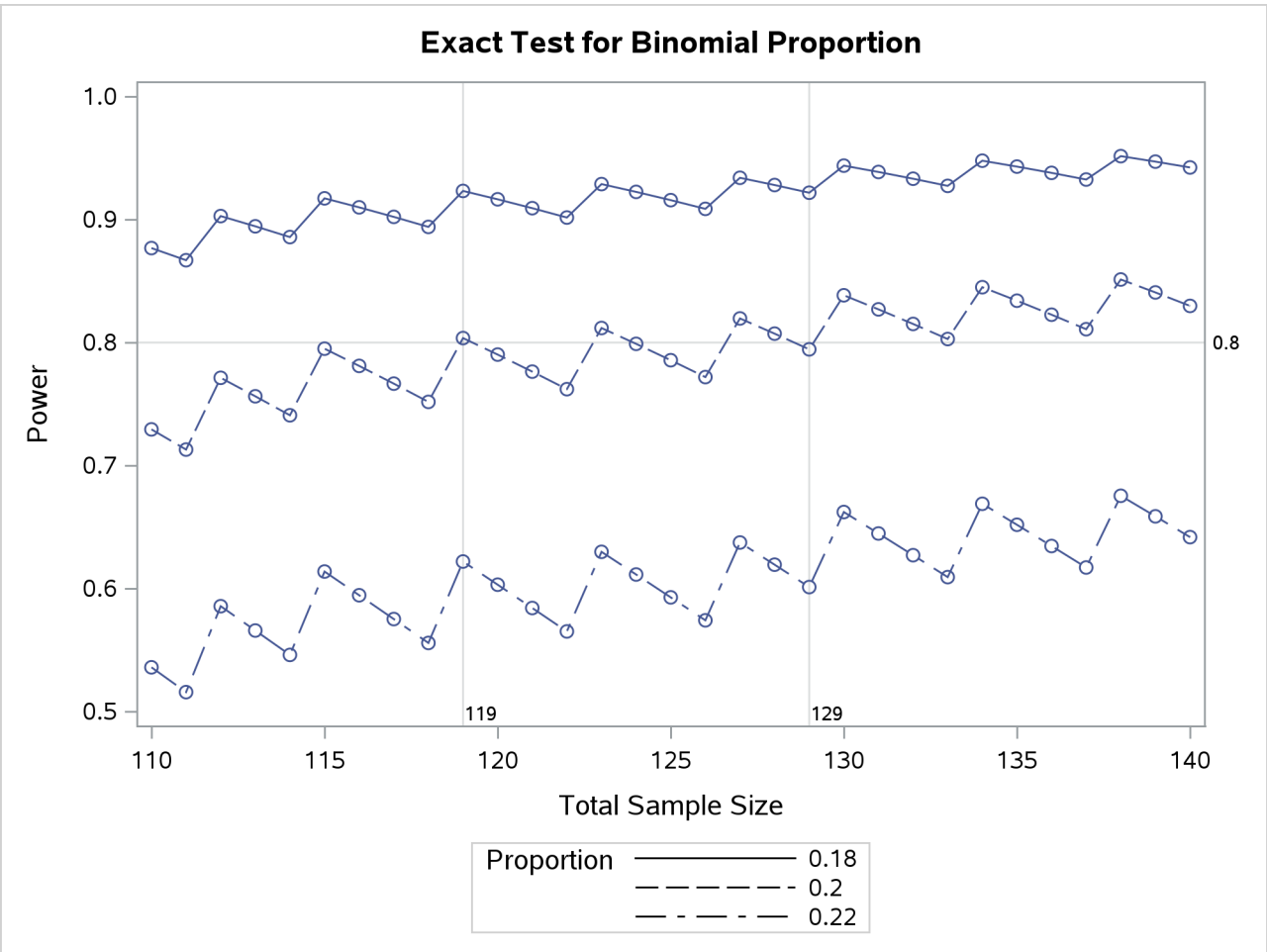

The plot reveals a dramatic sensitivity to the true proportion value. For N=119, the power is about 0.92 if the true proportion is 0.18, and as low as 0.62 if the proportion is 0.22. Note also that the power jumps occur at the same sample sizes in all three curves; the curves are only shifted and stretched vertically. This is because spikes and valleys in power curves are invariant to the true proportion value; they are due to changes in the critical value of the test.

A closer look at some ancillary output from the analysis sheds light on this property of the sawtooth pattern. You can add an ODS OUTPUT statement to save the plot content that corresponds to Output 96.2.3 to a data set:

```
proc power plotonly;
   ods output plotcontent=PlotData;
   onesamplefreq test=exact
      sides          = 1
      alpha          = 0.05
      nullproportion = 0.3
      proportion     = 0.2
      ntotal         = 119
      power          = .;
   plot x=n min=110 max=140 step=1
      yopts=(ref=.8) xopts=(ref=119 129);
run;
```

The PlotData data set contains parameter values for each point in the plot. The parameters include underlying characteristics of the putative test. The following statements print the critical value and actual significance level along with sample size and power:

```
proc print data=PlotData;
   var NTotal LowerCritVal Alpha Power;
run;
```

Output 96.2.5 shows the plot data.

Output 96.2.5 Numerical Content of Plot

Obs	NTotal	LowerCritVal	Alpha	Power
1	110	24	0.0356	0.729
2	111	24	0.0313	0.713
3	112	25	0.0446	0.771
4	113	25	0.0395	0.756
5	114	25	0.0349	0.741
6	115	26	0.0490	0.795
7	116	26	0.0435	0.781
8	117	26	0.0386	0.767
9	118	26	0.0341	0.752
10	119	27	0.0478	0.804
11	120	27	0.0425	0.790
12	121	27	0.0377	0.776
13	122	27	0.0334	0.762
14	123	28	0.0465	0.812
15	124	28	0.0414	0.799
16	125	28	0.0368	0.786
17	126	28	0.0327	0.772
18	127	29	0.0453	0.820
19	128	29	0.0404	0.807
20	129	29	0.0359	0.794
21	130	30	0.0493	0.838
22	131	30	0.0441	0.827
23	132	30	0.0394	0.815
24	133	30	0.0351	0.803
25	134	31	0.0480	0.845
26	135	31	0.0429	0.834
27	136	31	0.0384	0.823
28	137	31	0.0342	0.811
29	138	32	0.0466	0.851
30	139	32	0.0418	0.841
31	140	32	0.0374	0.830

Note that whenever the critical value changes, the actual α jumps up to a value close to the nominal $\alpha = 0.05$, and the power also jumps up. Then while the critical value stays constant, the actual α and power slowly decrease. The critical value is independent of the true proportion value. So you can achieve a locally maximal power by choosing a sample size corresponding to a spike on the sawtooth curve, and this choice is locally optimal *regardless* of the unknown value of the true proportion. Locally optimal sample sizes in this case include 115, 119, 123, 127, 130, and 134.

As a point of interest, the power does not always jump sharply and decrease gradually. The shape of the sawtooth depends on the direction of the test and the location of the null proportion relative to 0.5. For example, if the direction of the hypothesis in this example is reversed (by switching true and null proportion values) so that the rejection region is in the upper tail, then the power curve exhibits sharp decreases and gradual increases. The following statements are similar to those producing the plot in Output 96.2.3 but with values of the PROPORTION= and NULLPROPORTION= options switched:

```
proc power plotonly;
   onesamplefreq test=exact
      sides          = 1
      alpha          = 0.05
      nullproportion = 0.2
      proportion     = 0.3
      ntotal         = 119
      power          = .;
   plot x=n min=110 max=140 step=1;
run;
```

The resulting plot is shown in Output 96.2.6.

Output 96.2.6 Plot of Power versus Sample Size for Another One-sided Test

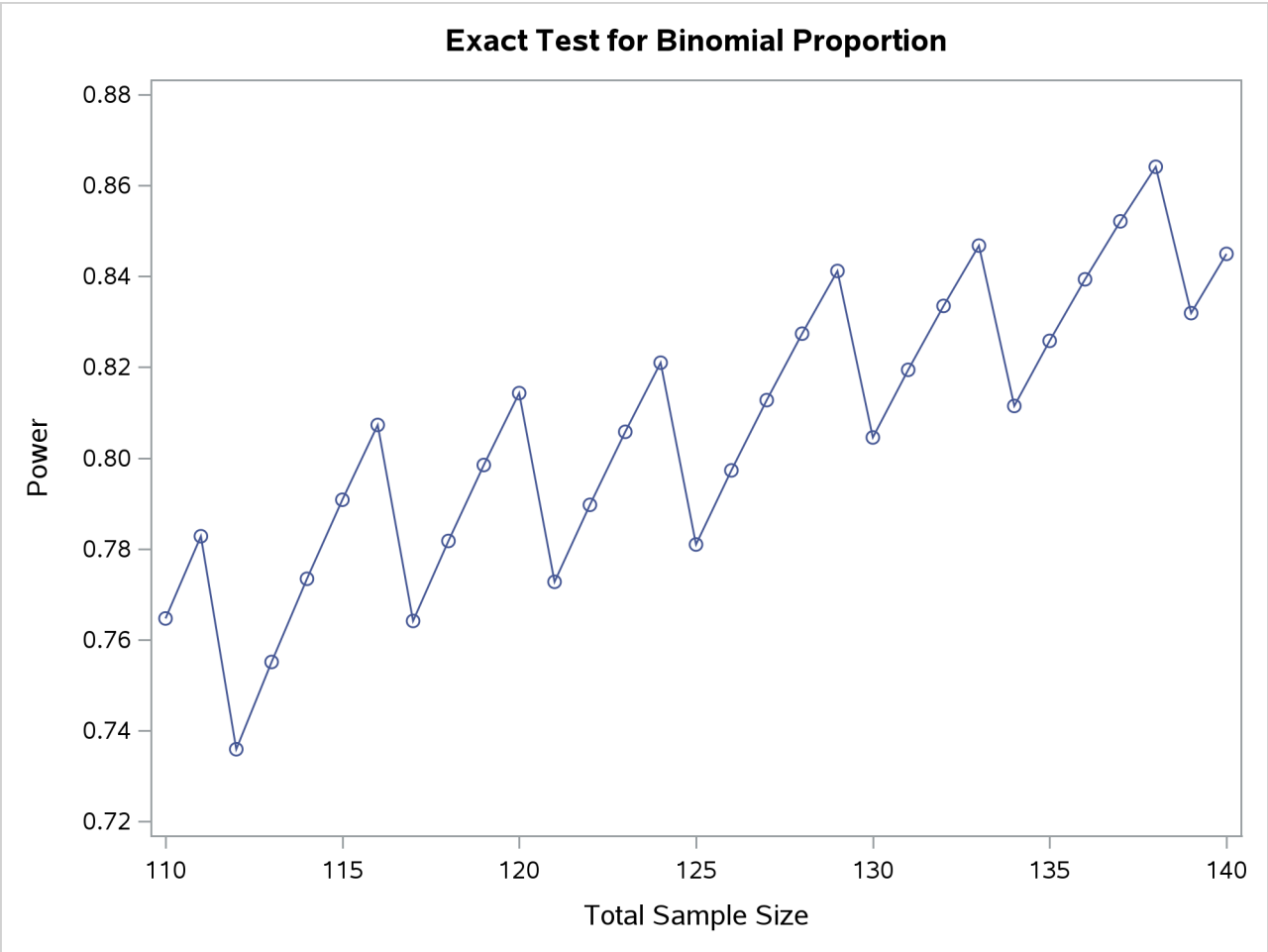

Finally, two-sided tests can lead to even more irregular power curve shapes, since changes in lower and upper critical values affect the power in different ways. The following statements produce a plot of power versus sample size for the scenario of a two-sided test with high alpha and a true proportion close to the null value:

```
proc power plotonly;
   onesamplefreq test=exact
      sides          = 2
      alpha          = 0.2
      nullproportion = 0.1
      proportion     = 0.09
      ntotal         = 10
      power          = .;
   plot x=n min=2 max=100 step=1;
run;

ods graphics off;
```

Output 96.2.7 shows the resulting plot.

Output 96.2.7 Plot of Power versus Sample Size for a Two-Sided Test

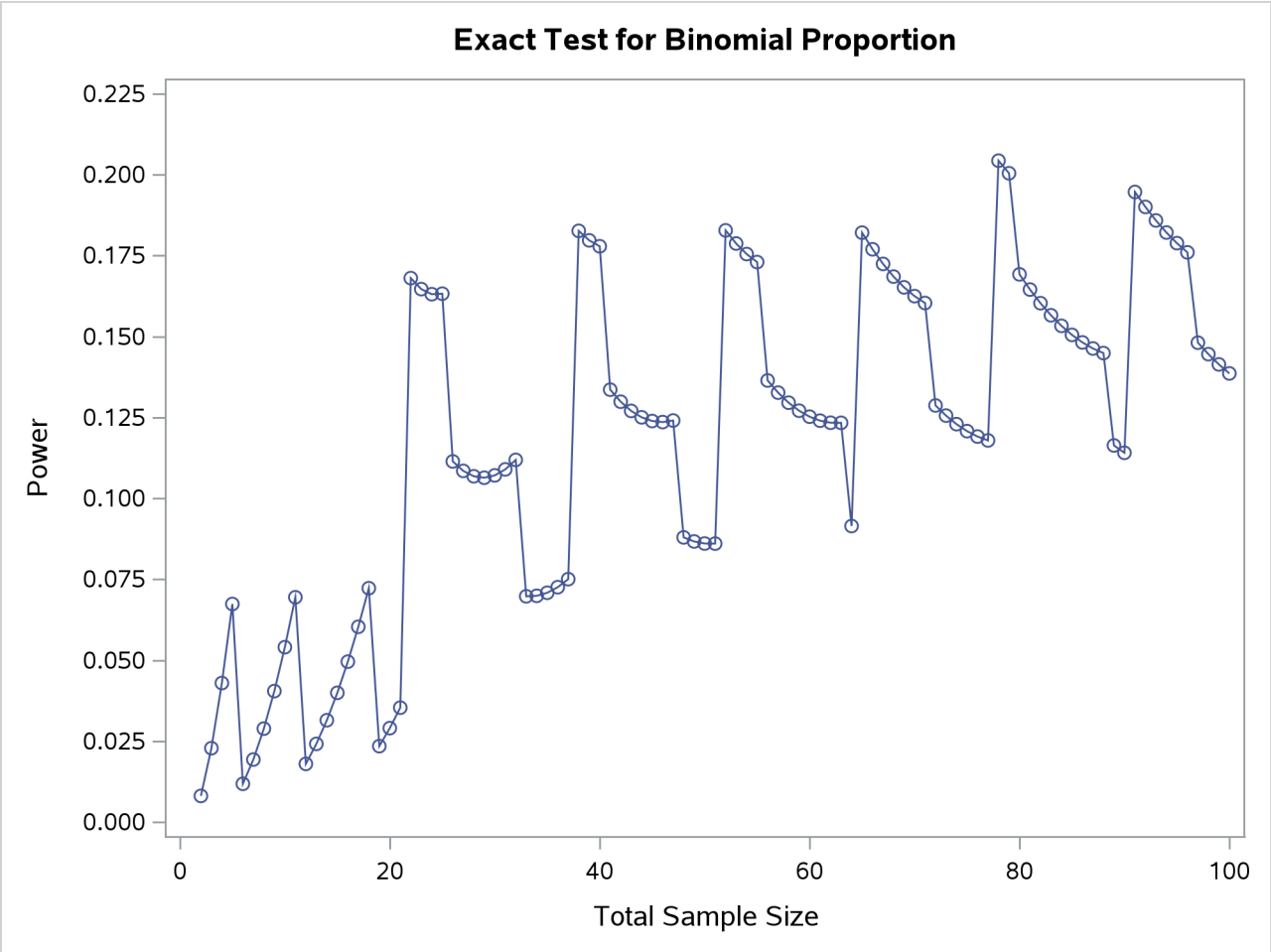

Due to the irregular shapes of power curves for proportion tests, the question "Which sample size should I use?" is often insufficient. A sample size solution produced directly in PROC POWER reveals the smallest possible sample size to achieve your target power. But as the examples in this section demonstrate, it is helpful to consult graphs for answers to questions such as the following:

- Which sample size will guarantee that all higher sample sizes also achieve my target power?
- Given a candidate sample size, can I increase it slightly to achieve locally maximal power, or perhaps even decrease it and get higher power?

Example 96.3: Simple AB/BA Crossover Designs

(View the complete code for this example (pwrex03.sas) in the example repository.)

Crossover trials are experiments in which each subject is given a sequence of different treatments. They are especially common in clinical trials for medical studies. The reduction in variability from taking multiple measurements on a subject allows for more precise treatment comparisons. The simplest such design is the AB/BA crossover, in which each subject receives each of two treatments in a randomized order.

Under certain simplifying assumptions, you can test the treatment difference in an AB/BA crossover trial by using either a paired or two-sample *t* test (or equivalence test, depending on the hypothesis). This example will demonstrate when and how you can use the PAIREDMEANS statement in PROC POWER to perform power analyses for AB/BA crossover designs.

Senn (1993, Chapter 3) discusses a study comparing the effects of two bronchodilator medications in treatment of asthma, by using an AB/BA crossover design. Suppose you want to plan a similar study comparing two new medications, "Xilodol" and "Brantium." Half of the patients would be assigned to sequence AB, getting a dose of Xilodol in the first treatment period, a wash-out period of one week, and then a dose of Brantium in the second treatment period. The other half would be assigned to sequence BA, following the same schedule but with the drugs reversed. In each treatment period you would administer the drugs in the morning and then measure peak expiratory flow (PEF) at the end of the day, with higher PEF representing better lung function.

You conjecture that the mean and standard deviation of PEF are about $\mu_A = 330$ and $\sigma_A = 40$ for Xilodol and $\mu_B = 310$ and $\sigma_B = 55$ for Brantium, and that each pair of measurements on the same subject will have a correlation of about 0.3. You want to compute the power of both one-sided and two-sided tests of mean difference, with a significance level of $\alpha = 0.01$, for a sample size of 100 patients and also plot the power for a range of 50 to 200 patients. Note that the allocation ratio of patients to the two sequences is irrelevant in this analysis.

The choice of statistical test depends on which assumptions are reasonable. One possibility is a *t* test. A paired or two-sample *t* test is valid when there is no carryover effect and no interactions between patients, treatments, and periods. See Senn (1993, Chapter 3) for more details. The choice between a paired or a two-sample test depends on what you assume about the period effect. If you assume no period effect, then a paired *t* test is the appropriate analysis for the design, with the first member of each pair being the Xilodol measurement (regardless of which sequence the patient belongs to). Otherwise, the two-sample *t* test approach is called for, since this analysis adjusts for the period effect by using an extra degree of freedom.

Suppose you assume no period effect. Then you can use the PAIREDMEANS statement in PROC POWER with the TEST=DIFF option to perform a sample size analysis for the paired *t* test. Indicate power as the result parameter by specifying the POWER= option with a missing value (.). Specify the conjectured means and standard deviations for each drug by using the PAIREDMEANS= and PAIREDSTDDEVS= options and the correlation by using the CORR= option. Specify both one- and two-sided tests by using the SIDES= option, the significance level by using the ALPHA= option, and the sample size (in terms of number of pairs) by using the NPAIRS= option. Generate a plot of power versus sample size by specifying the PLOT

statement with X=N to request a plot with sample size on the X axis. (The result parameter, here power, is always plotted on the other axis.) Use the MIN= and MAX= options in the PLOT statement to specify the sample size range (as numbers of pairs).

The following statements perform the sample size analysis:

```
ods graphics on;

proc power;
   pairedmeans test=diff
      pairedmeans   = (330 310)
      pairedstddevs = (40 55)
      corr          = 0.3
      sides         = 1 2
      alpha         = 0.01
      npairs        = 100
      power         = .;
   plot x=n min=50 max=200;
run;

ods graphics off;
```

Default values for the NULLDIFF= and DIST= options specify a null mean difference of 0 and the assumption of normally distributed data. The output is shown in Output 96.3.1 and Output 96.3.2.

Output 96.3.1 Power for Paired *t* Analysis of Crossover Design

The POWER Procedure
Paired t Test for Mean Difference

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Alpha	0.01
Mean 1	330
Mean 2	310
Standard Deviation 1	40
Standard Deviation 2	55
Correlation	0.3
Number of Pairs	100
Null Difference	0

Computed Power		
Index	**Sides**	**Power**
1	1	0.865
2	2	0.801

Output 96.3.2 Plot of Power versus Sample Size for Paired *t* Analysis of Crossover Design

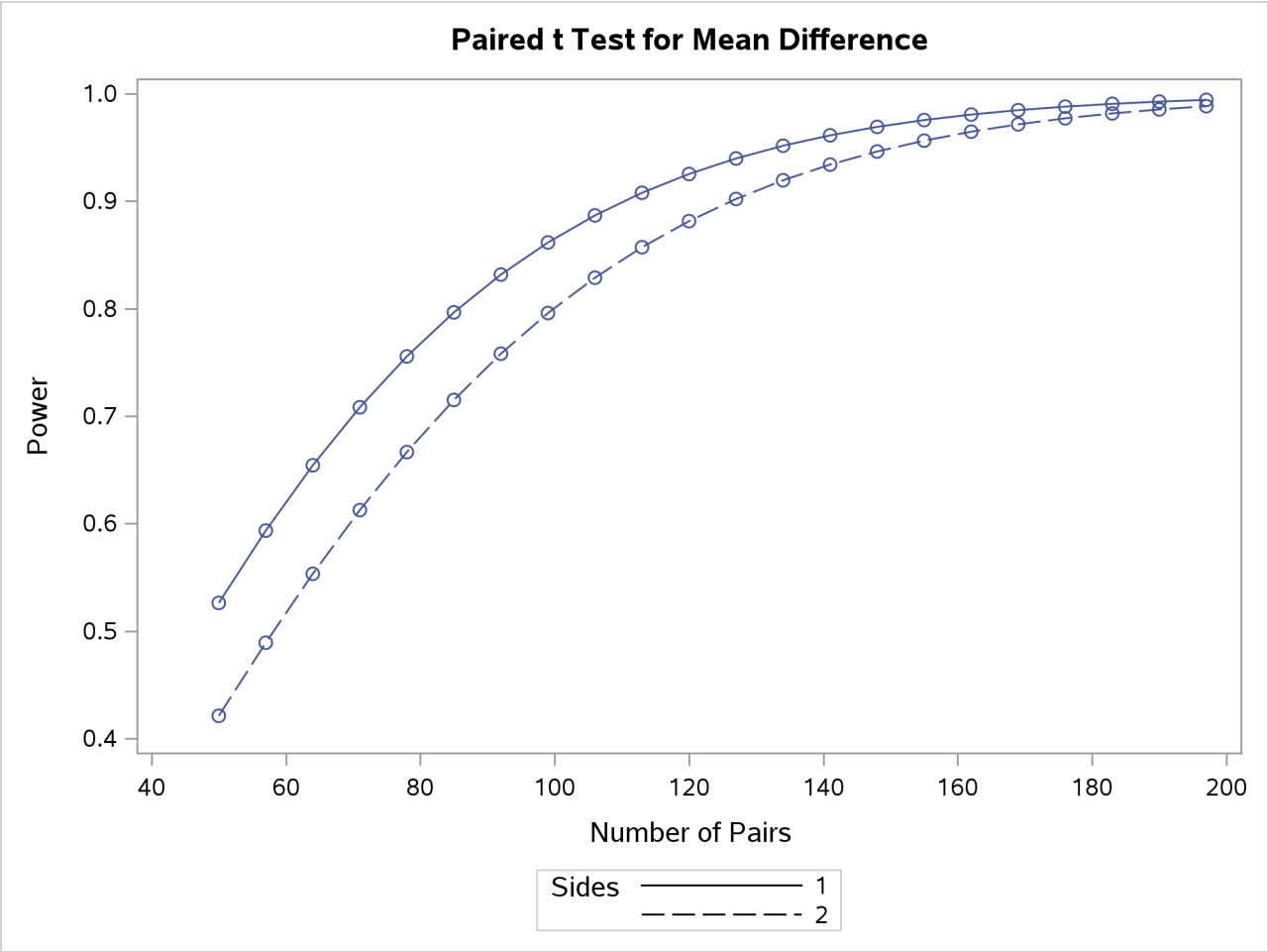

The "Computed Power" table in Output 96.3.1 shows that the power with 100 patients is about 0.8 for the two-sided test and 0.87 for the one-sided test with the alternative of larger Brantium mean. In Output 96.3.2, the line style identifies the number of sides of the test. The plotting symbols identify locations of actual computed powers; the curves are linear interpolations of these points. The plot demonstrates how much higher the power is in the one-sided test than in the two-sided test for the range of sample sizes.

Suppose now that instead of detecting a difference between Xilodol and Brantium, you want to establish that they are similar—in particular, that the absolute mean PEF difference is at most 35. You might consider this goal if, for example, one of the drugs has fewer side effects and if a difference of no more than 35 is considered clinically small. Instead of a standard *t* test, you would conduct an *equivalence test* of the treatment mean difference for the two drugs. You would test the hypothesis that the true difference is less than –35 or more than 35 against the alternative that the mean difference is between –35 and 35, by using an additive model and a two one-sided tests ("TOST") analysis.

Assuming no period effect, you can use the PAIREDMEANS statement with the TEST=EQUIV_DIFF option to perform a sample size analysis for the paired equivalence test. Indicate power as the result parameter by specifying the POWER= option with a missing value (.). Use the LOWER= and UPPER= options to specify the equivalence bounds of –35 and 35. Use the PAIREDMEANS=, PAIREDSTDDEVS=, CORR=, and ALPHA= options in the same way as in the *t* test at the beginning of this example to specify the remaining

parameters.

The following statements perform the sample size analysis:

```
proc power;
   pairedmeans test=equiv_add
      lower         = -35
      upper         = 35
      pairedmeans   = (330 310)
      pairedstddevs = (40 55)
      corr          = 0.3
      alpha         = 0.01
      npairs        = 100
      power         = .;
run;
```

The default option DIST=NORMAL specifies an assumption of normally distributed data. The output is shown in Output 96.3.3.

Output 96.3.3 Power for Paired Equivalence Test for Crossover Design

The POWER Procedure
Equivalence Test for Paired Mean Difference

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Lower Equivalence Bound	-35
Upper Equivalence Bound	35
Alpha	0.01
Reference Mean	330
Treatment Mean	310
Standard Deviation 1	40
Standard Deviation 2	55
Correlation	0.3
Number of Pairs	100

Computed Power
Power
0.598

The power for the paired equivalence test with 100 patients is about 0.6.

Example 96.4: Noninferiority Test with Lognormal Data

(View the complete code for this example (pwrex04.sas) in the example repository.)

The typical goal in noninferiority testing is to conclude that a new treatment or process or product is not appreciably worse than some standard. This is accomplished by convincingly rejecting a one-sided null hypothesis that the new treatment is appreciably worse than the standard. When designing such studies, investigators must define precisely what constitutes "appreciably worse."

You can use the POWER procedure for sample size analyses for a variety of noninferiority tests, by specifying custom, one-sided null hypotheses for common tests. This example illustrates the strategy (often called Blackwelder's scheme; Blackwelder 1982) by comparing the means of two independent lognormal samples. The logic applies to one-sample, two-sample, and paired-sample problems involving normally distributed measures and proportions.

Suppose you are designing a study hoping to show that a new (less expensive) manufacturing process does not produce appreciably more pollution than the current process. Quantifying "appreciably worse" as 10%, you seek to show that the mean pollutant level from the new process is less than 110% of that from the current process. In standard hypothesis testing notation, you seek to reject

$$H_0\colon \frac{\mu_{\text{new}}}{\mu_{\text{current}}} \geq 1.10$$

in favor of

$$H_A\colon \frac{\mu_{\text{new}}}{\mu_{\text{current}}} < 1.10$$

This is described graphically in Figure 96.8. Mean ratios below 100% are better levels for the new process; a ratio of 100% indicates absolute equivalence; ratios of 100–110% are "tolerably" worse; and ratios exceeding 110% are appreciably worse.

Figure 96.8 Hypotheses for the Pollutant Study

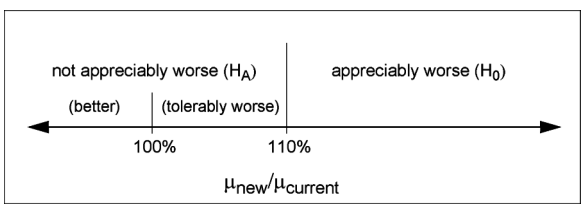

An appropriate test for this situation is the common two-group t test on log-transformed data. The hypotheses become

$$H_0\colon \log(\mu_{\text{new}}) - \log(\mu_{\text{current}}) \geq \log(1.10)$$
$$H_A\colon \log(\mu_{\text{new}}) - \log(\mu_{\text{current}}) < \log(1.10)$$

Measurements of the pollutant level will be taken by using laboratory models of the two processes and will be treated as independent lognormal observations with a coefficient of variation (σ/μ) between 0.5 and 0.6 for both processes. You will end up with 300 measurements for the current process and 180 for the new one. It is important to avoid a Type I error here, so you set the Type I error rate to 0.01. Your theoretical work suggests that the new process will actually reduce the pollutant by about 10% (to 90% of current), but you

need to compute and graph the power of the study if the new levels are actually between 70% and 120% of current levels.

Implement the sample size analysis by using the TWOSAMPLEMEANS statement in PROC POWER with the TEST=RATIO option. Indicate power as the result parameter by specifying the POWER= option with a missing value (.). Specify a series of scenarios for the mean ratio between 0.7 and 1.2 by using the MEANRATIO= option. Use the NULLRATIO= option to specify the null mean ratio of 1.10. Specify SIDES=L to indicate a one-sided test with the alternative hypothesis stating that the mean ratio is *lower* than the null value. Specify the significance level, scenarios for the coefficient of variation, and the group sample sizes by using the ALPHA=, CV=, and GROUPNS= options. Generate a plot of power versus mean ratio by specifying the PLOT statement with the X=EFFECT option to request a plot with mean ratio on the X axis. (The result parameter, here power, is always plotted on the other axis.) Use the STEP= option in the PLOT statement to specify an interval of 0.05 between computed points in the plot.

The following statements perform the desired analysis:

```
ods graphics on;

proc power;
   twosamplemeans test=ratio
      meanratio = 0.7 to 1.2 by 0.1
      nullratio = 1.10
      sides     = L
      alpha     = 0.01
      cv        = 0.5 0.6
      groupns   = (300 180)
      power     = .;
   plot x=effect step=0.05;
run;

ods graphics off;
```

Note the use of SIDES=L, which forces computations for cases that need a rejection region that is opposite to the one providing the most one-tailed power; in this case, it is the lower tail. Such cases will show power that is less than the prescribed Type I error rate. The default option DIST=LOGNORMAL specifies the assumption of lognormally distributed data. The default MIN= and MAX= options in the plot statement specify an X axis range identical to the effect size range in the TWOSAMPLEMEANS statement (mean ratios between 0.7 and 1.2).

Output 96.4.1 and Output 96.4.2 show the results.

Output 96.4.1 Power for Noninferiority Test of Ratio

The POWER Procedure
Two-Sample t Test for Mean Ratio

Fixed Scenario Elements	
Distribution	Lognormal
Method	Exact
Number of Sides	L
Null Geometric Mean Ratio	1.1
Alpha	0.01
Group 1 Sample Size	300
Group 2 Sample Size	180

Computed Power

Index	Geo Mean Ratio	CV	Power
1	0.7	0.5	>.999
2	0.7	0.6	>.999
3	0.8	0.5	>.999
4	0.8	0.6	>.999
5	0.9	0.5	0.985
6	0.9	0.6	0.933
7	1.0	0.5	0.424
8	1.0	0.6	0.306
9	1.1	0.5	0.010
10	1.1	0.6	0.010
11	1.2	0.5	<.001
12	1.2	0.6	<.001

Output 96.4.2 Plot of Power versus Mean Ratio for Noninferiority Test

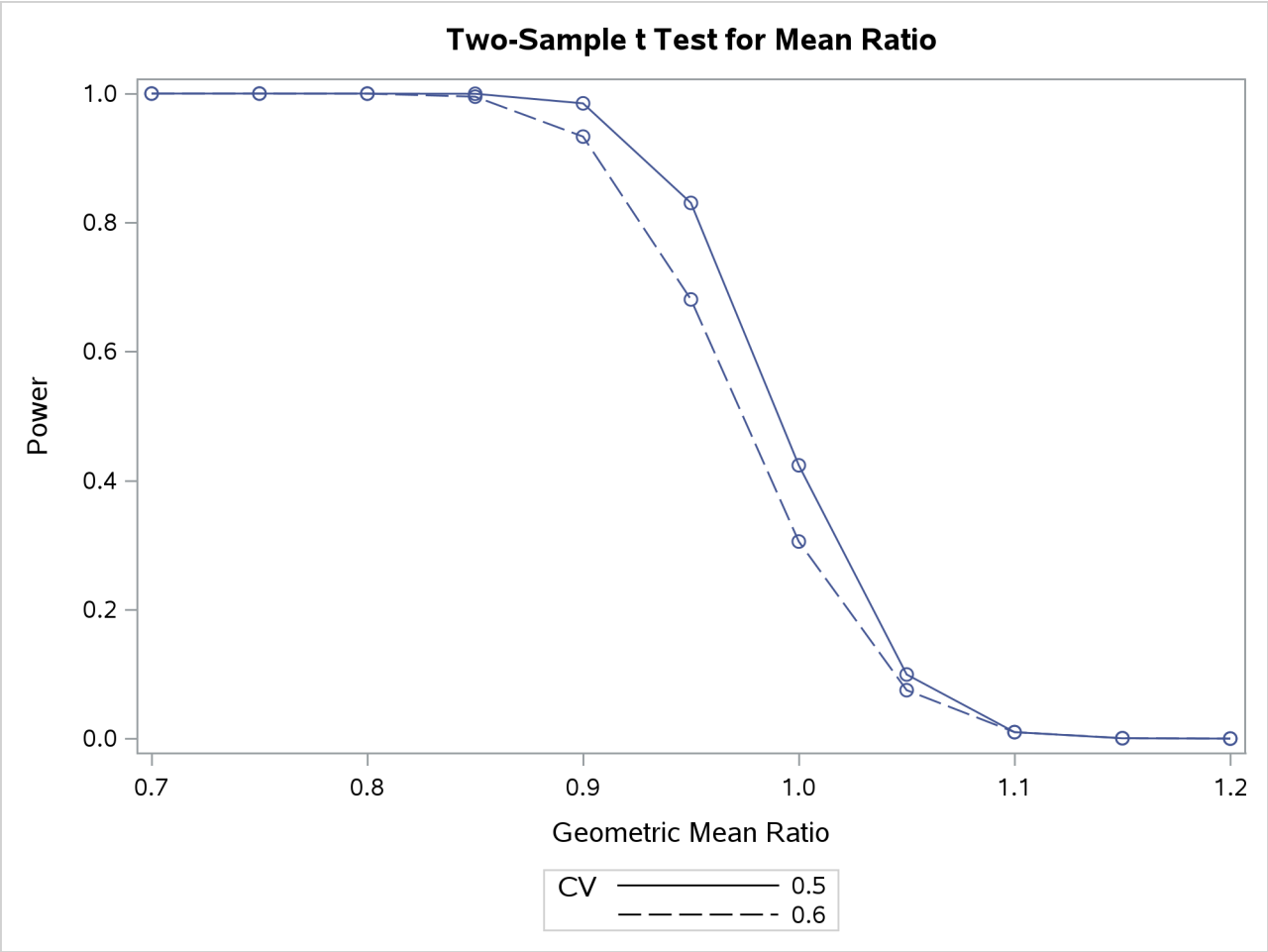

The "Computed Power" table in Output 96.4.1 shows that power exceeds 0.90 if the true mean ratio is 90% or less, as surmised. But power is unacceptably low (0.31–0.42) if the processes happen to be truly equivalent. Note that the power is identical to the alpha level (0.01) if the true mean ratio is 1.10 and below 0.01 if the true mean ratio is appreciably worse (>110%). In Output 96.4.2, the line style identifies the coefficient of variation. The plotting symbols identify locations of actual computed powers; the curves are linear interpolations of these points.

Example 96.5: Multiple Regression and Correlation

(View the complete code for this example (pwrex05.sas) in the example repository.)

You are working with a team of preventive cardiologists investigating whether elevated serum homocysteine levels are linked to atherosclerosis (plaque buildup) in coronary arteries. The planned analysis is an ordinary least squares regression to assess the relationship between total homocysteine level (tHcy) and a plaque burden index (PBI), adjusting for six other variables: age, gender, plasma levels of folate, vitamin B_6, vitamin B_{12}, and a serum cholesterol index. You will regress PBI on tHcy and the six other predictors (plus the intercept) and use a Type III F test to assess whether tHcy is a significant predictor after adjusting for the others. You wonder whether 100 subjects will provide adequate statistical power.

This is a correlational study at a single time. Subjects will be screened so that about half will have had a heart problem. All eight variables will be measured during one visit. Most clinicians are familiar with simple correlations between two variables, so you decide to pose the statistical problem in terms of estimating and testing the partial correlation between X_1 = tHcy and Y = PBI, controlling for the six other predictor variables ($R_{YX_1|X_{-1}}$). This greatly simplifies matters, especially the elicitation of the conjectured effect.

You use partial regression plots like that shown in Figure 96.9 to teach the team that the partial correlation between PBI and tHcy is the correlation of two sets of residuals obtained from ordinary regression models, one from regressing PBI on the six covariates and the other from regressing tHcy on the same covariates. Thus each subject has "expected" tHcy and PBI values based on the six covariates. The cardiologists believe that subjects whose tHcy is relatively higher than expected will also have a PBI that is relatively higher than expected. The partial correlation quantifies that adjusted association just as a standard simple correlation does with the unadjusted linear association between two variables.

Figure 96.9 Partial Regression Plot

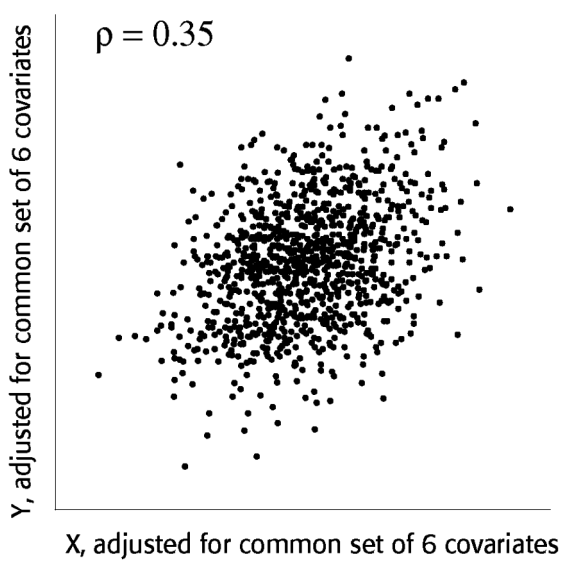

Based on previously published studies of various coronary risk factors and after viewing a set of scatterplots showing various correlations, the team surmises that the true partial correlation is likely to be at least 0.35.

You want to compute the statistical power for a sample size of $N = 100$ by using $\alpha = 0.05$. You also want to plot power for sample sizes between 50 and 150. Use the MULTREG statement to compute the power and the

PLOT statement to produce the graph. Since the predictors are observed rather than fixed in advanced, and a joint multivariate normal assumption seems tenable, use MODEL=RANDOM. The following statements perform the power analysis:

```
ods graphics on;

proc power;
   multreg
      model = random
      nfullpredictors = 7
      ntestpredictors = 1
      partialcorr = 0.35
      ntotal = 100
      power = .;
   plot x=n min=50 max=150;
run;

ods graphics off;
```

The POWER=. option identifies power as the parameter to compute. The NFULLPREDICTORS= option specifies seven total predictors (not including the intercept), and the NTESTPREDICTORS= option indicates that one of those predictors is being tested. The PARTIALCORR= and NTOTAL= options specify the partial correlation and sample size, respectively. The default value for the ALPHA= option sets the significance level to 0.05. The X=N option in the plot statement requests a plot of sample size on the X axis, and the MIN= and MAX= options specify the sample size range.

Output 96.5.1 shows the output, and Output 96.5.2 shows the plot.

Output 96.5.1 Power Analysis for Multiple Regression

The POWER Procedure
Type III F Test in Multiple Regression

Fixed Scenario Elements	
Method	Exact
Model	Random X
Number of Predictors in Full Model	7
Number of Test Predictors	1
Partial Correlation	0.35
Total Sample Size	100
Alpha	0.05

Computed Power
Power
0.939

Output 96.5.2 Plot of Power versus Sample Size for Multiple Regression

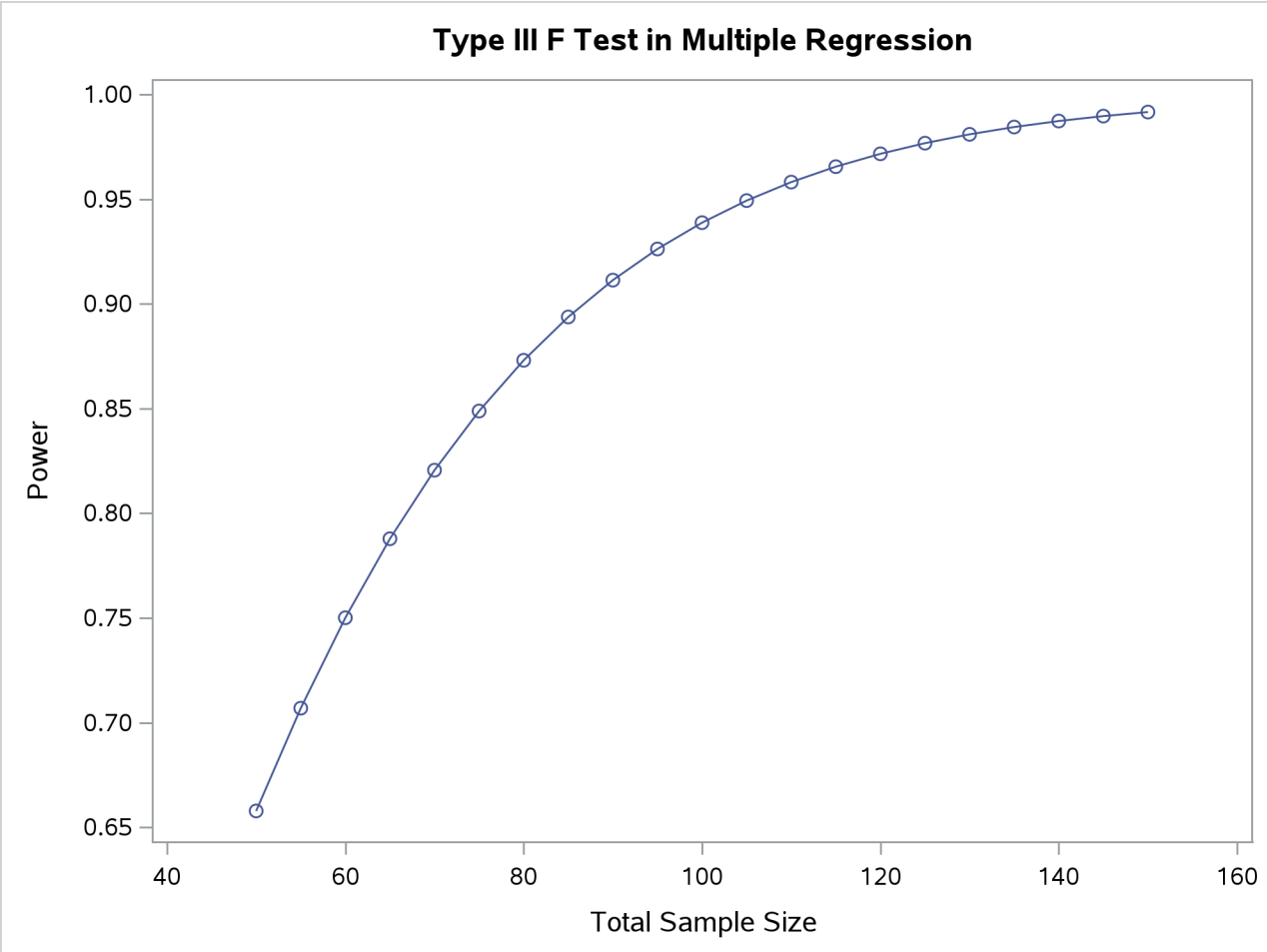

For the sample size $N = 100$, the study is almost balanced with respect to Type I and Type II error rates, with $\alpha = 0.05$ and $\beta = 1 - 0.937 = 0.063$. The study thus seems well designed at this sample size.

Now suppose that in a follow-up meeting with the cardiologists, you discover that their specific intent is to demonstrate that the (partial) correlation between PBI and tHcy is greater than 0.2. You suggest changing the planned data analysis to a one-sided Fisher's z test with a null correlation of 0.2. The following statements perform a power analysis for this test:

```
proc power;
   onecorr dist=fisherz
      npvars = 6
      corr = 0.35
      nullcorr = 0.2
      sides = 1
      ntotal = 100
      power = .;
run;
```

The DIST=FISHERZ option in the ONECORR statement specifies Fisher's z test. The NPARTIALVARS= option specifies that six additional variables are adjusted for in the partial correlation. The CORR= option

specifies the conjectured correlation of 0.35, and the NULLCORR= option indicates the null value of 0.2. The SIDES= option specifies a one-sided test.

Output 96.5.3 shows the output.

Output 96.5.3 Power Analysis for Fisher's z Test

The POWER Procedure
Fisher's z Test for Pearson Correlation

Fixed Scenario Elements	
Distribution	Fisher's z transformation of r
Method	Normal approximation
Number of Sides	1
Null Correlation	0.2
Number of Variables Partialled Out	6
Correlation	0.35
Total Sample Size	100
Nominal Alpha	0.05

	Computed Power
Actual Alpha	**Power**
0.05	0.466

The power for Fisher's z test is less than 50%, the decrease being mostly due to the smaller effect size (relative to the null value). When asked for a recommendation for a new sample size goal, you compute the required sample size to achieve a power of 0.95 (to balance Type I and Type II errors) and 0.85 (a threshold deemed to be minimally acceptable to the team). The following statements perform the sample size determination:

```
proc power;
   onecorr dist=fisherz
      npvars = 6
      corr = 0.35
      nullcorr = 0.2
      sides = 1
      ntotal = .
      power = 0.85 0.95;
run;
```

The NTOTAL=. option identifies sample size as the parameter to compute, and the POWER= option specifies the target powers.

Output 96.5.4 Sample Size Determination for Fisher's z Test

The POWER Procedure
Fisher's z Test for Pearson Correlation

Fixed Scenario Elements	
Distribution	Fisher's z transformation of r
Method	Normal approximation
Number of Sides	1
Null Correlation	0.2
Number of Variables Partialled Out	6
Correlation	0.35
Nominal Alpha	0.05

Computed N Total				
Index	**Nominal Power**	**Actual Alpha**	**Actual Power**	**N Total**
1	0.85	0.05	0.850	280
2	0.95	0.05	0.950	417

The results in Output 96.5.4 reveal a required sample size of 417 to achieve a power of 0.95 and a required sample size of 280 to achieve a power of 0.85.

Example 96.6: Comparing Two Survival Curves

(View the complete code for this example (pwrex06.sas) in the example repository.)

You are consulting for a clinical research group planning a trial to compare survival rates for proposed and standard cancer treatments. The planned data analysis is a log-rank test to nonparametrically compare the overall survival curves for the two treatments. Your goal is to determine an appropriate sample size to achieve a power of 0.8 for a two-sided test with $\alpha = 0.05$ by using a balanced design.

The survival curve for patients on the standard treatment is well known to be approximately exponential with a median survival time of five years. The research group conjectures that the new proposed treatment will yield a (nonexponential) survival curve similar to the dashed line in Figure 96.6.1.

Patients will be accrued uniformly over two years and then followed for an additional three years past the accrual period. Some loss to follow-up is expected, with roughly exponential rates that would result in about 50% loss with the standard treatment within 10 years. The loss to follow-up with the proposed treatment is more difficult to predict, but 50% loss would be expected to occur sometime between years 5 and 20.

Output 96.6.1 Survival Curves

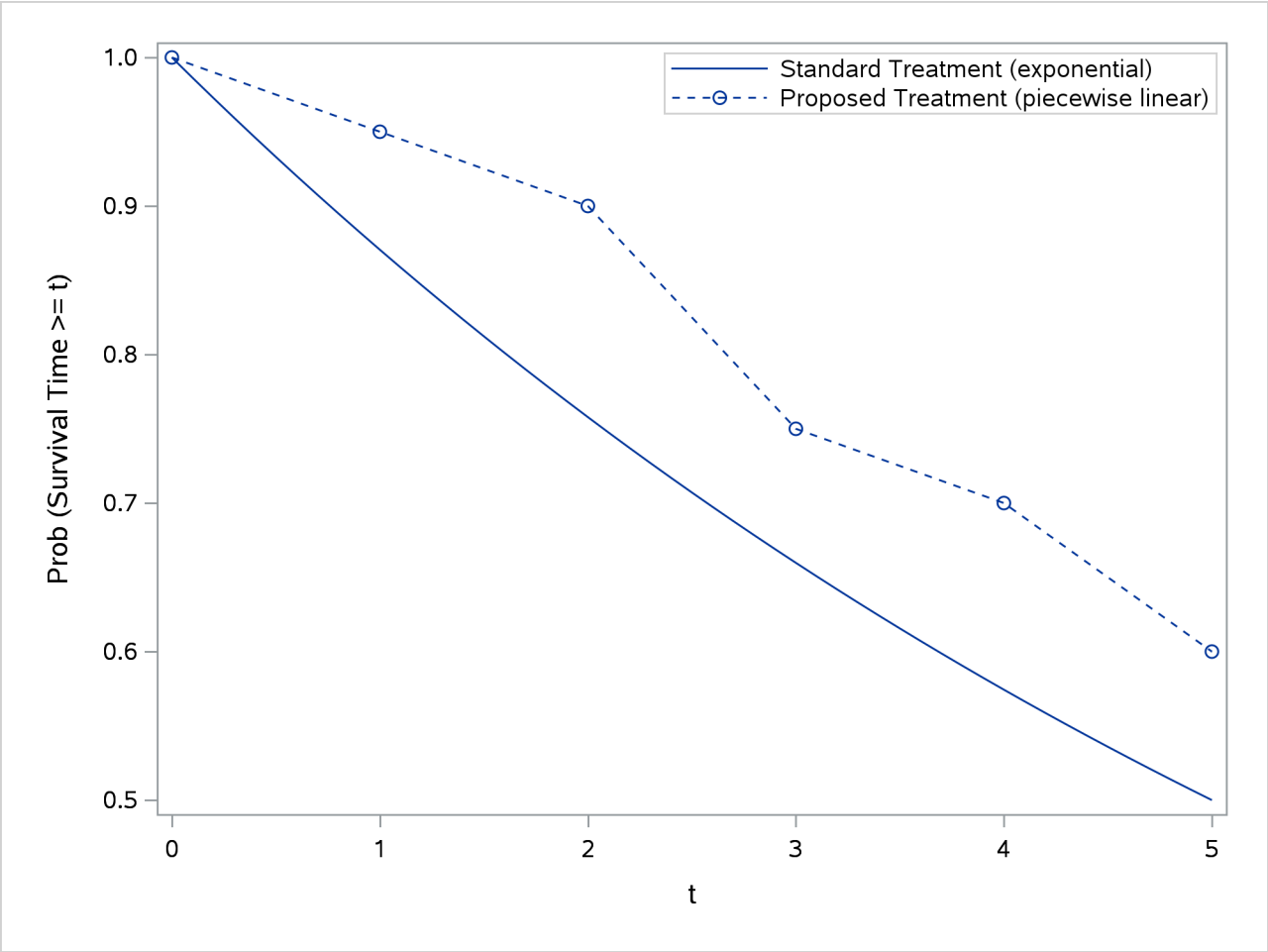

Use the TWOSAMPLESURVIVAL statement with the TEST=LOGRANK option to compute the required sample size for the log-rank test. The following statements perform the analysis:

```
proc power;
   twosamplesurvival test=logrank
      curve("Standard") = 5 : 0.5
      curve("Proposed") = (1 to 5 by 1):(0.95 0.9 0.75 0.7 0.6)
      groupsurvival = "Standard" | "Proposed"
      accrualtime = 2
      followuptime = 3
      groupmedlosstimes = 10 | 20 5
      power = 0.8
      npergroup = .;
run;
```

The CURVE= option defines the two survival curves. The “Standard” curve has only one point, specifying an exponential form with a survival probability of 0.5 at year 5. The “Proposed” curve is a piecewise linear curve defined by the five points shown in Figure 96.6.1. The GROUPSURVIVAL= option assigns the survival curves to the two groups, and the ACCRUALTIME= and FOLLOWUPTIME= options specify the accrual and follow-up times. The GROUPMEDLOSSTIMES= option specifies the years at which 50% loss is expected

to occur. The POWER= option specifies the target power, and the NPERGROUP=. option identifies sample size per group as the parameter to compute. Default values for the SIDES= and ALPHA= options specify a two-sided test with $\alpha = 0.05$.

Output 96.6.2 shows the results.

Output 96.6.2 Sample Size Determination for Log-Rank Test

The POWER Procedure
Log-Rank Test for Two Survival Curves

Fixed Scenario Elements	
Method	Lakatos normal approximation
Accrual Time	2
Follow-up Time	3
Group 1 Survival Curve	Standard
Form of Survival Curve 1	Exponential
Group 2 Survival Curve	Proposed
Form of Survival Curve 2	Piecewise Linear
Group 1 Median Loss Time	10
Nominal Power	0.8
Number of Sides	2
Number of Time Sub-Intervals	12
Alpha	0.05

Computed N per Group			
Index	**Median Loss Time 2**	**Actual Power**	**N per Group**
1	20	0.800	228
2	5	0.801	234

The required sample size per group to achieve a power of 0.8 is 228 if the median loss time is 20 years for the proposed treatment. Only six more patients are required in each group if the median loss time is as short as five years.

Example 96.7: Confidence Interval Precision

(View the complete code for this example (pwrex07.sas) in the example repository.)

An investment firm has hired you to help plan a study to estimate the success of a new investment strategy called IntuiVest. The study involves complex simulations of market conditions over time, and it tracks the balance of a hypothetical brokerage account starting with $50,000. Each simulation is very expensive in terms of computing time. You are asked to determine an appropriate number of simulations to estimate the average change in the account balance at the end of three years. The goal is to have a 95% chance of obtaining a 90% confidence interval whose half-width is at most $1,000. That is, the firm wants to have a 95% chance of being able to correctly claim at the end of the study that "Our research shows with 90% confidence that IntuiVest yields a profit of $X +/– $1,000 at the end of three years on an initial investment of $50,000 (under simulated market conditions)."

The probability of achieving the desired precision (that is, a small interval width) can be calculated either unconditionally or conditionally given that the true mean is captured by the interval. You decide to use the conditional form, considering two of its advantages:

- The conditional probability is usually lower than the unconditional probability for the same sample size, meaning that the conditional form is generally conservative.
- The overall probability of achieving the desired precision *and* capturing the true mean is easily computed as the product of the half-width probability and the confidence level. In this case, the overall probability is $0.95 \times 0.9 = 0.855$.

Based on some initial simulations, you expect a standard deviation between $25,000 and $45,000 for the ending account balance. You will consider both of these values in the sample size analysis.

As mentioned in the section "Overview of Power Concepts" on page 8009, an analysis of confidence interval precision is analogous to a traditional power analysis, with "CI Half-Width" taking the place of effect size and "Prob(Width)" taking the place of power. In this example, the target CI Half-Width is 1000, and the desired Prob(Width) is 0.95.

In addition to computing sample sizes for a half-width of $1,000, you are asked to plot the required number of simulations for a range of half-widths between $500 and $2,000. Use the ONESAMPLEMEANS statement with the CI=T option to implement the sample size determination. The following statements perform the analysis:

```
ods graphics on;

proc power;
   onesamplemeans ci=t
      alpha = 0.1
      halfwidth = 1000
      stddev = 25000 45000
      probwidth = 0.95
      ntotal = .;
   plot x=effect min=500 max=2000;
run;

ods graphics off;
```

The NTOTAL=. option identifies sample size as the parameter to compute. The ALPHA=0.1 option specifies a confidence level of $1 - \alpha = 0.9$. The HALFWIDTH= option specifies the target half-width, and the STDDEV= option specifies the conjectured standard deviation values. The PROBWIDTH= option specifies the desired probability of achieving the target precision. The default value PROBTYPE=CONDITIONAL specifies that this probability is conditional on the true mean being captured by the interval. The default of SIDES=2 indicates a two-sided interval.

Output 96.7.1 shows the output, and Output 96.7.2 shows the plot.

Output 96.7.1 Sample Size Determination for Confidence Interval Precision

The POWER Procedure
Confidence Interval for Mean

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Alpha	0.1
CI Half-Width	1000
Nominal Prob(Width)	0.95
Number of Sides	2
Prob Type	Conditional

Computed N Total			
Index	**Std Dev**	**Actual Prob(Width)**	**N Total**
1	25000	0.951	1788
2	45000	0.950	5652

Output 96.7.2 Plot of Sample Size versus Confidence Interval Half-Width

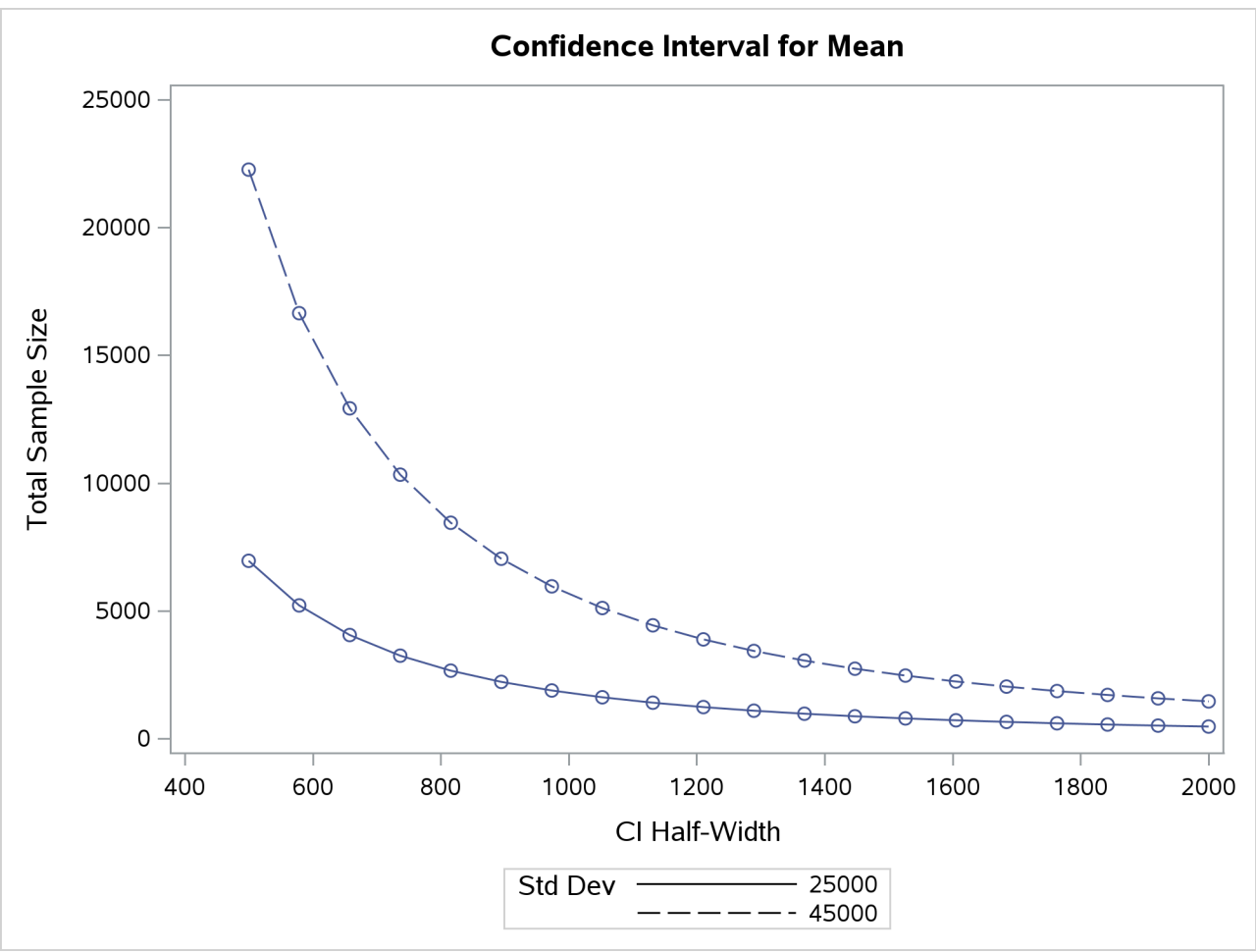

The number of simulations required in order to have a 95% chance of obtaining a half-width of at most 1000 is between 1788 and 5652, depending on the standard deviation. The plot reveals that more than 20,000 simulations would be required for a half-width of 500, assuming the higher standard deviation.

Example 96.8: Customizing Plots

(View the complete code for this example (pwrex08.sas) in the example repository.)

This example demonstrates various ways you can modify and enhance plots:

- assigning analysis parameters to axes
- fine-tuning a sample size axis
- adding reference lines
- linking plot features to analysis parameters
- choosing key (legend) styles
- modifying symbol locations

The example plots are all based on a sample size analysis for a two-sample t test of group mean difference. You start by computing the sample size required to achieve a power of 0.9 by using a two-sided test with $\alpha = 0.05$, assuming the first mean is 12, the second mean is either 15 or 18, and the standard deviation is either 7 or 9.

Use the TWOSAMPLEMEANS statement with the TEST=DIFF option to compute the required sample sizes. Indicate total sample size as the result parameter by supplying a missing value (.) with the NTOTAL= option. Use the GROUPMEANS=, STDDEV=, and POWER= options to specify values of the other parameters. The following statements perform the sample size computations:

```
proc power;
   twosamplemeans test=diff
      groupmeans   = 12 | 15 18
      stddev       = 7 9
      power        = 0.9
      ntotal       = .;
run;
```

Default values for the NULLDIFF=, SIDES=, GROUPWEIGHTS=, and DIST= options specify a null mean difference of 0, two-sided test, balanced design, and assumption of normally distributed data, respectively.

Output 96.8.1 shows that the required sample size ranges from 60 to 382, depending on the unknown standard deviation and second mean.

Output 96.8.1 Computed Sample Sizes

The POWER Procedure
Two-Sample t Test for Mean Difference

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Group 1 Mean	12
Nominal Power	0.9
Number of Sides	2
Null Difference	0
Alpha	0.05
Group 1 Weight	1
Group 2 Weight	1

Computed N Total				
Index	**Mean2**	**Std Dev**	**Actual Power**	**N Total**
1	15	7	0.902	232
2	15	9	0.901	382
3	18	7	0.904	60
4	18	9	0.904	98

Assigning Analysis Parameters to Axes

Use the PLOT statement to produce plots for all power and sample size analyses in PROC POWER. For the sample size analysis described at the beginning of this example, suppose you want to plot the required sample size on the Y axis against a range of powers between 0.5 and 0.95 on the X axis. The X= and Y= options specify which parameter to plot against the result and which axis to assign to this parameter. You can use either the X= or the Y= option, but not both. Use the X=POWER option in the PLOT statement to request a plot with power on the X axis. The result parameter, here total sample size, is always plotted on the other axis. Use the MIN= and MAX= options to specify the range of the axis indicated with either the X= or the Y= option. Here, specify MIN=0.5 and MAX=0.95 to specify the power range. The following statements produce the plot:

```
ods graphics on;

proc power plotonly;
   twosamplemeans test=diff
      groupmeans   = 12 | 15 18
      stddev       = 7 9
      power        = 0.9
      ntotal       = .;
   plot x=power min=0.5 max=0.95;
run;
```

Note that the value (0.9) of the POWER= option in the TWOSAMPLEMEANS statement is only a placeholder when the PLOTONLY option is used and both the MIN= and MAX= options are used, because the values of the MIN= and MAX= options override the value of 0.9. But the POWER= option itself is still required in the TWOSAMPLEMEANS statement, to provide a complete specification of the sample size analysis.

The resulting plot is shown in Output 96.8.2.

Output 96.8.2 Plot of Sample Size versus Power

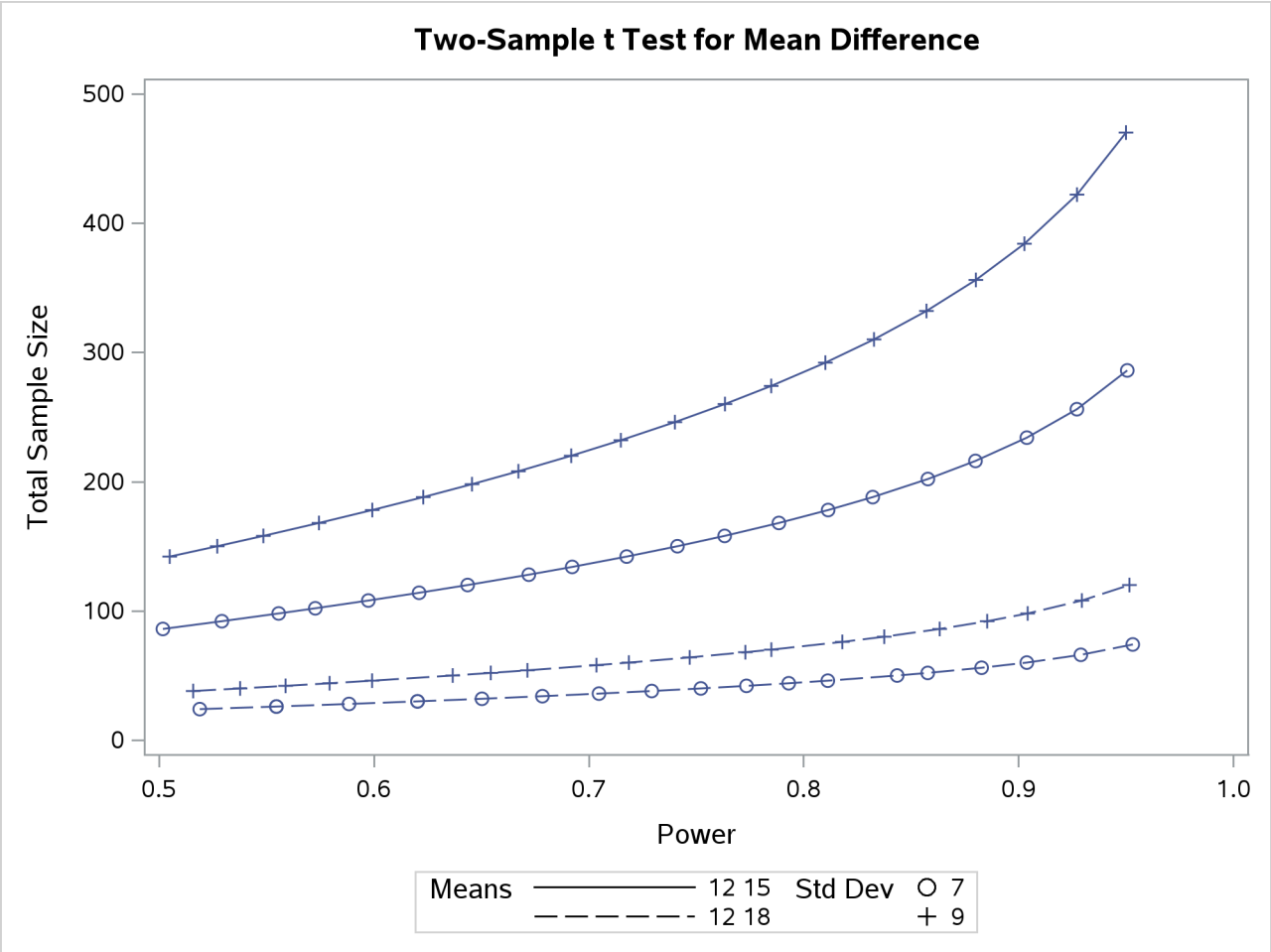

The line style identifies the group means scenario, and the plotting symbol identifies the standard deviation scenario. The locations of plotting symbols indicate computed sample sizes; the curves are linear interpolations of these points. By default, each curve consists of approximately 20 computed points (sometimes slightly more or less, depending on the analysis).

If you would rather plot power on the Y axis versus sample size on the X axis, you have two general strategies to choose from. One strategy is to use the Y= option instead of the X= option in the PLOT statement:

```
plot y=power min=0.5 max=0.95;
```

Output 96.8.3 Plot of Power versus Sample Size using First Strategy

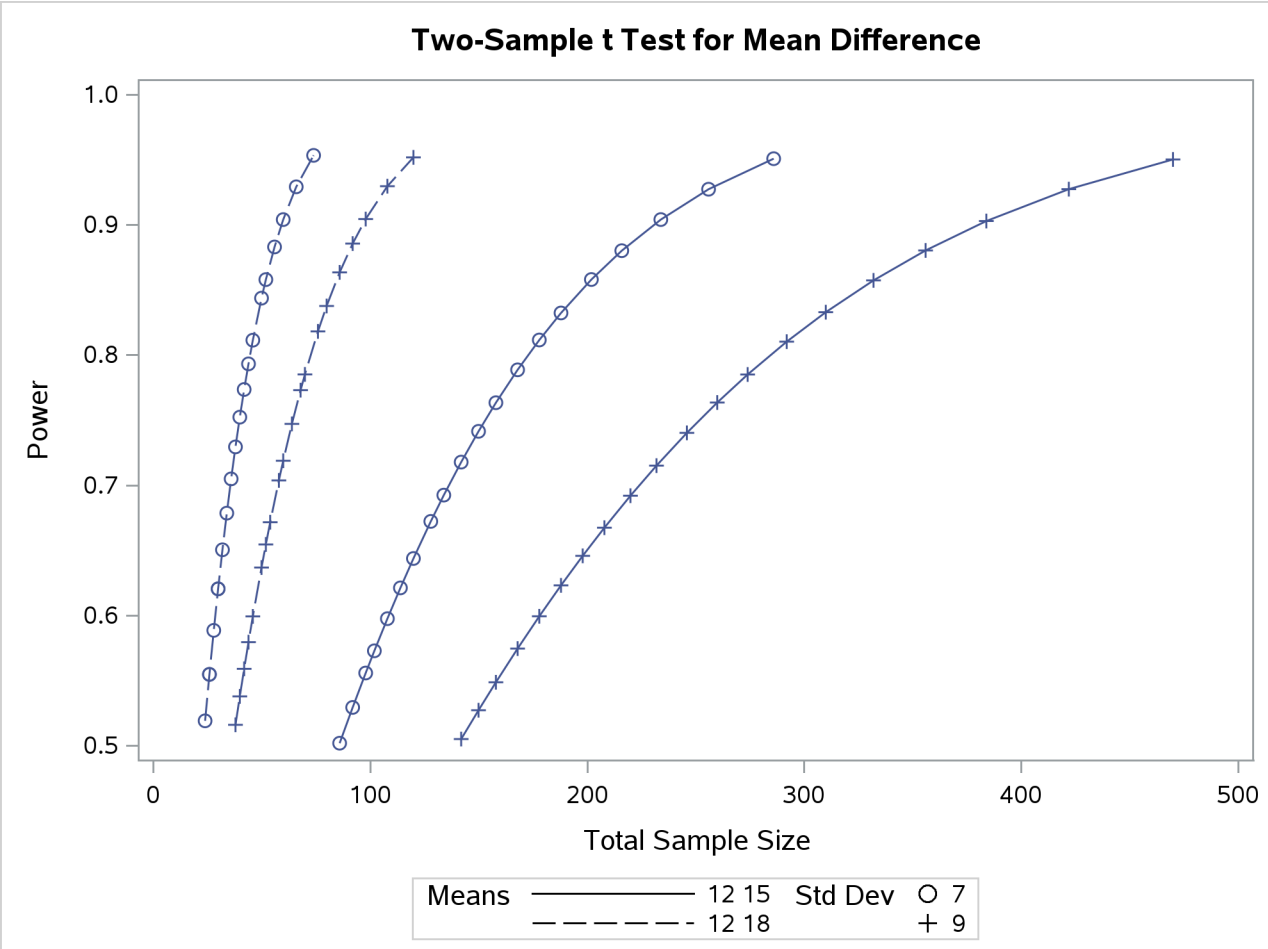

Note that the resulting plot (Output 96.8.3) is essentially a mirror image of Output 96.8.2. The axis ranges are set such that each curve in Output 96.8.3 contains similar values of Y instead of X. Each plotted point represents the computed value of the X axis at the input value of the Y axis.

A second strategy for plotting power versus sample size (when originally solving for sample size) is to invert the analysis and base the plot on computed power for a given range of sample sizes. This strategy works well for monotonic power curves (as is the case for the *t* test and most other continuous analyses). It is advantageous in the sense of preserving the traditional role of the Y axis as the computed parameter. A common way to implement this strategy is as follows:

- Determine the range of sample sizes sufficient to cover at the desired power range for all curves (where each "curve" represents a scenario for standard deviation and second group mean).
- Use this range for the X axis of a plot.

To determine the required sample sizes for target powers of 0.5 and 0.95, change the values in the POWER= option as follows to reflect this range:

```
proc power;
   twosamplemeans test=diff
      groupmeans   = 12 | 15 18
      stddev       = 7 9
      power        = 0.5 0.95
      ntotal       = .;
run;
```

Output 96.8.4 reveals that a sample size range of 24 to 470 is approximately sufficient to cover the desired power range of 0.5 to 0.95 for all curves ("approximately" because the actual power at the rounded sample size of 24 is slightly higher than the nominal power of 0.5).

Output 96.8.4 Computed Sample Sizes

The POWER Procedure
Two-Sample t Test for Mean Difference

Fixed Scenario Elements	
Distribution	Normal
Method	Exact
Group 1 Mean	12
Number of Sides	2
Null Difference	0
Alpha	0.05
Group 1 Weight	1
Group 2 Weight	1

Computed N Total					
Index	**Mean2**	**Std Dev**	**Nominal Power**	**Actual Power**	**N Total**
1	15	7	0.50	0.502	86
2	15	7	0.95	0.951	286
3	15	9	0.50	0.505	142
4	15	9	0.95	0.950	470
5	18	7	0.50	0.519	24
6	18	7	0.95	0.953	74
7	18	9	0.50	0.516	38
8	18	9	0.95	0.952	120

To plot power on the Y axis for sample sizes between 20 and 500, use the X=N option in the PLOT statement with MIN=20 and MAX=500:

```
proc power plotonly;
   twosamplemeans test=diff
      groupmeans   = 12 | 15 18
      stddev       = 7 9
      power        = .
      ntotal       = 200;
   plot x=n min=20 max=500;
run;
```

Each curve in the resulting plot in Output 96.8.5 covers at least a power range of 0.5 to 0.95.

Output 96.8.5 Plot of Power versus Sample Size Using Second Strategy

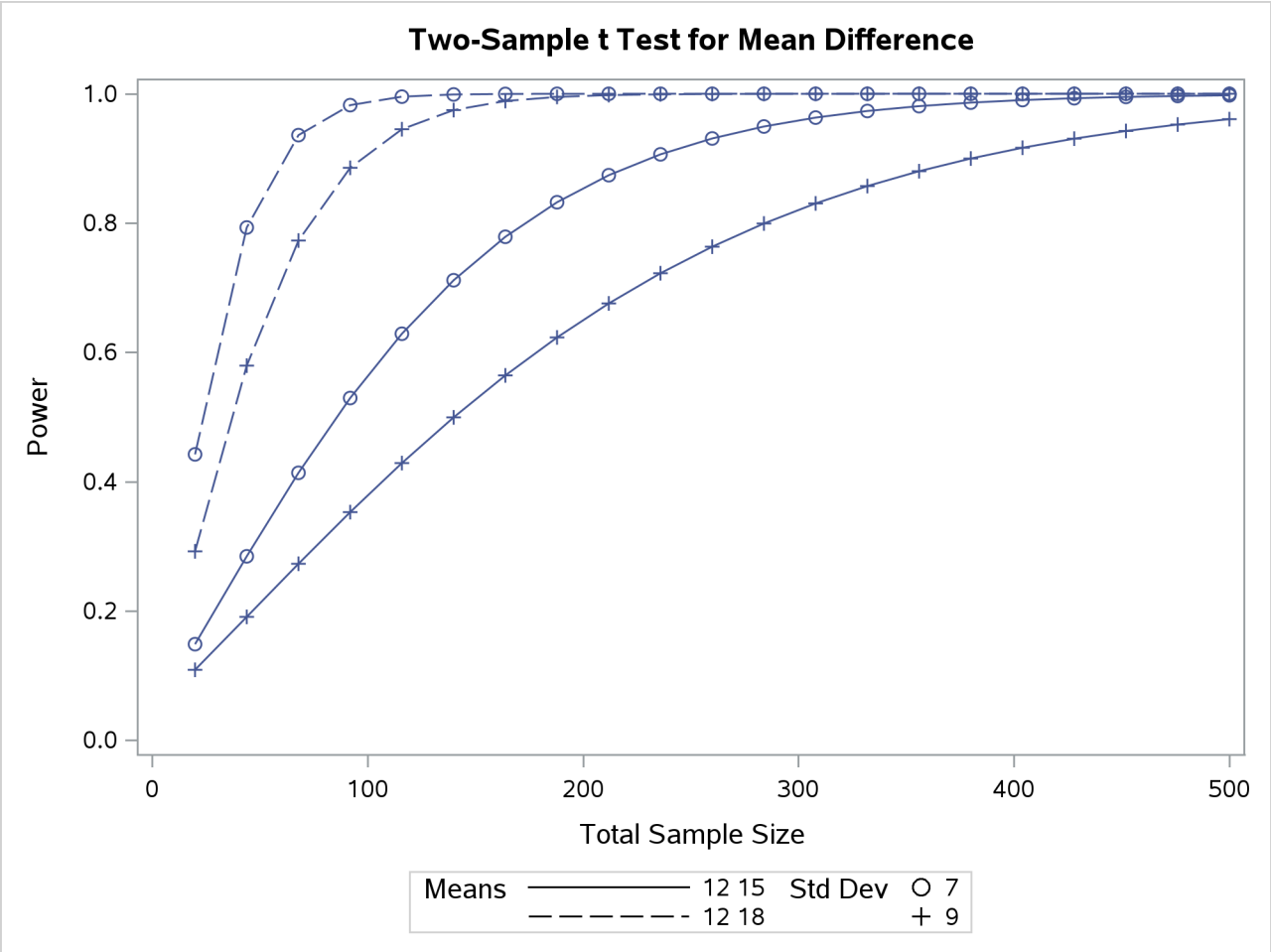

Finally, suppose you want to produce a plot of sample size versus effect size for a power of 0.9. In this case, the "effect size" is defined to be the mean difference. You need to reparameterize the analysis by using the MEANDIFF= option instead of the GROUPMEANS= option to produce a plot, since each plot axis must be represented by a scalar parameter. Use the X=EFFECT option in the PLOT statement to assign the mean difference to the X axis. The following statements produce a plot of required sample size to detect mean differences between 3 and 6:

```
proc power plotonly;
   twosamplemeans test=diff
      meandiff     = 3 6
      stddev       = 7 9
      power        = 0.9
      ntotal       = .;
   plot x=effect min=3 max=6;
run;
```

The resulting plot Output 96.8.6 shows how the required sample size decreases with increasing mean difference.

Output 96.8.6 Plot of Sample Size versus Mean Difference

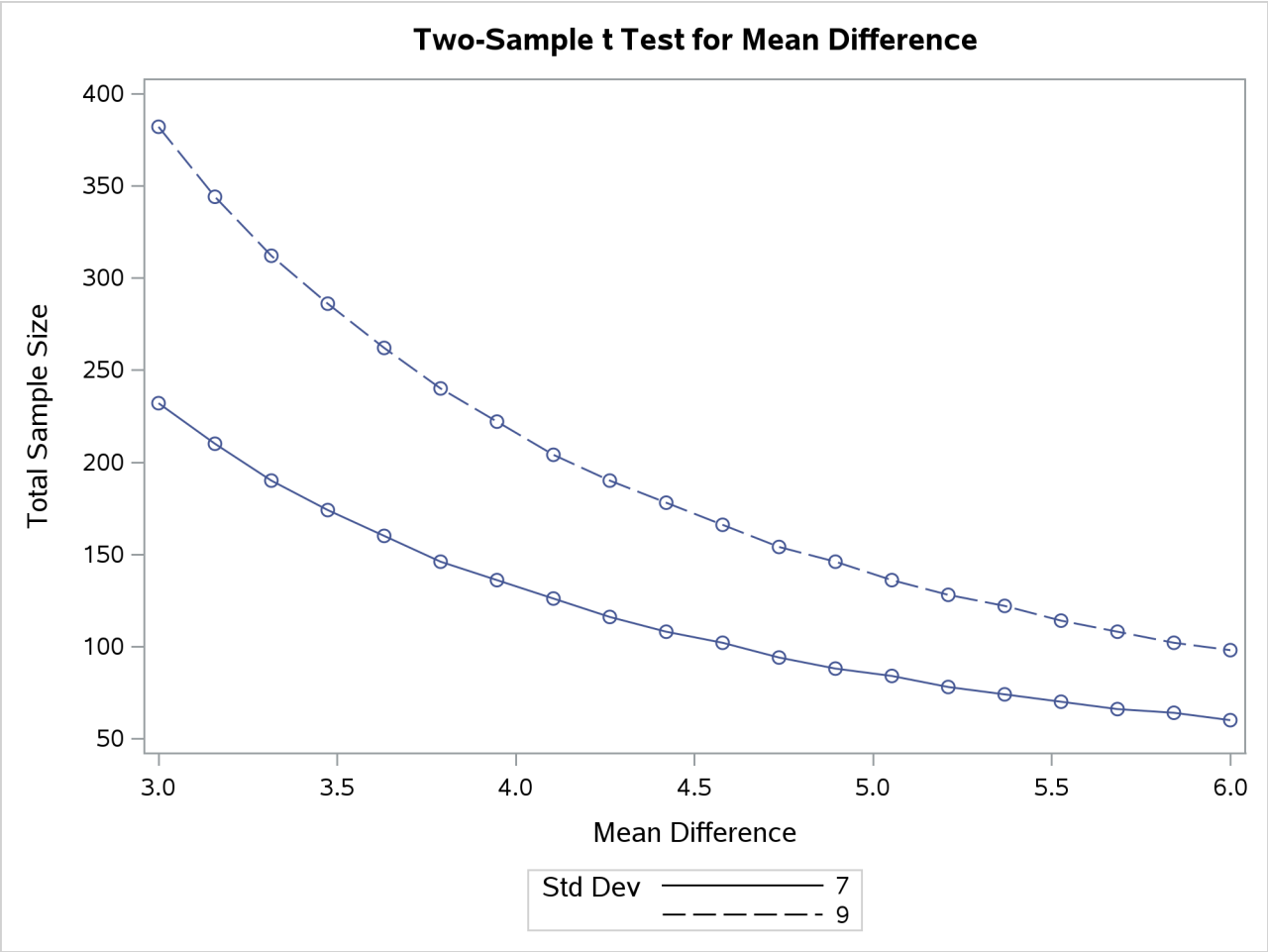

Fine-Tuning a Sample Size Axis

Consider the following plot request for a sample size analysis similar to the one in Output 96.8.1 but with only a single scenario, and with unbalanced sample size allocation of 2:1:

```
proc power plotonly;
   ods output plotcontent=PlotData;
   twosamplemeans test=diff
      groupmeans   = 12 | 18
      stddev       = 7
      groupweights = 2 | 1
      power        = .
      ntotal       = 20;
   plot x=n min=20 max=50 npoints=20;
run;
```

The MIN=, MAX=, and NPOINTS= options in the PLOT statement request a plot with 20 points between 20 and 50. But the resulting plot (Output 96.8.7) appears to have only 11 points, and they range from 18 to 48.

Output 96.8.7 Plot with Overlapping Points

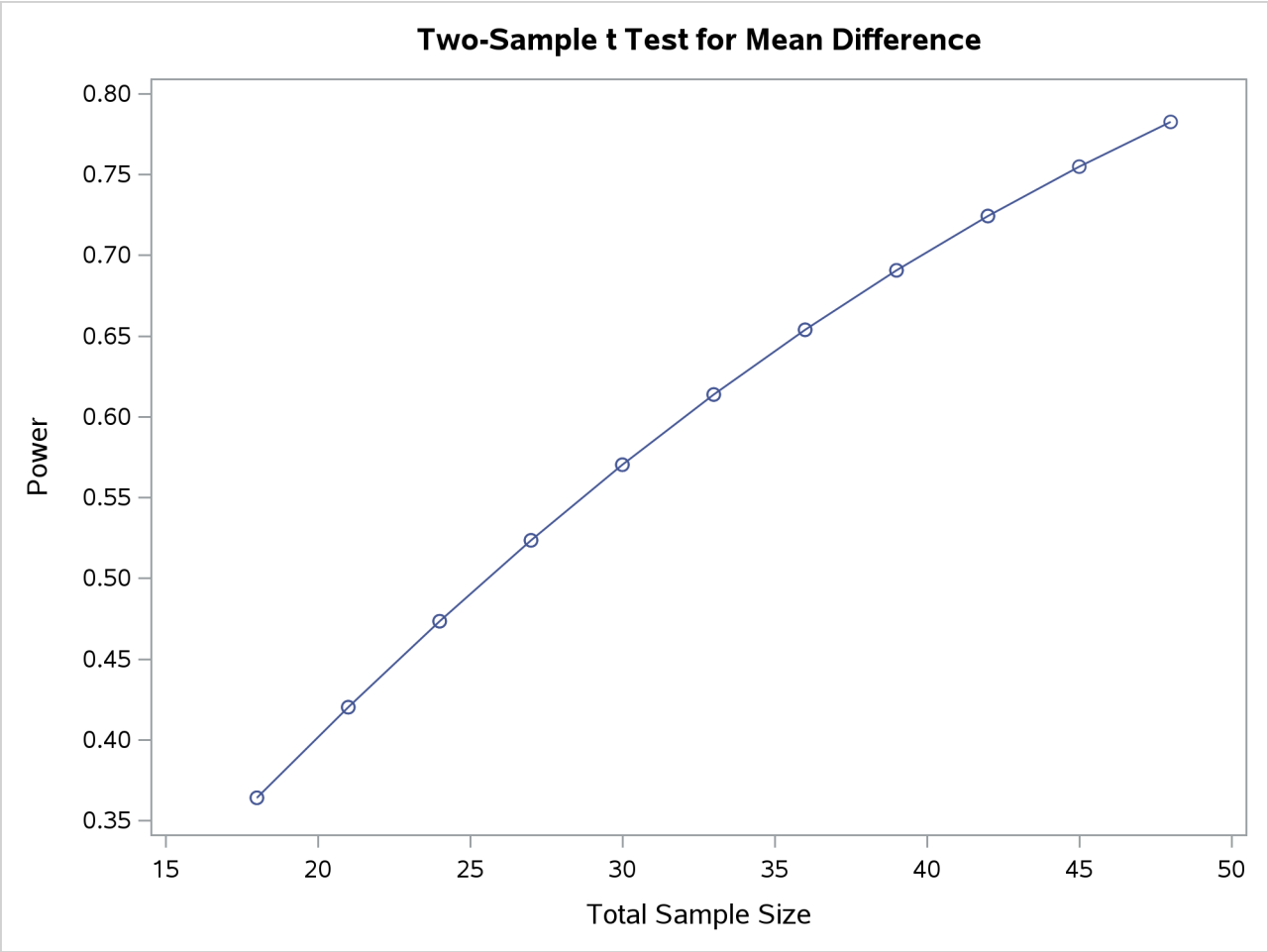

The reason that this plot has fewer points than usual is due to the rounding of sample sizes. If you do not use the NFRACTIONAL option in the analysis statement (here, the TWOSAMPLEMEANS statement), then the set of sample size points determined by the MIN=, MAX=, NPOINTS=, and STEP= options in the PLOT statement can be rounded to satisfy the allocation weights. In this case, they are rounded down to the nearest multiples of 3 (the sum of the weights), and many of the points overlap. To see the overlap, you can print the NominalNTotal (unadjusted) and NTotal (rounded) variables in the PlotContent ODS object (here saved to a data set called PlotData):

```
proc print data=PlotData;
   var NominalNTotal NTotal;
run;
```

The output is shown in Output 96.8.8.

Output 96.8.8 Sample Sizes

Obs	NominalNTotal	NTotal
1	18.0	18
2	19.6	18
3	21.2	21
4	22.7	21
5	24.3	24
6	25.9	24
7	27.5	27
8	29.1	27
9	30.6	30
10	32.2	30
11	33.8	33
12	35.4	33
13	36.9	36
14	38.5	36
15	40.1	39
16	41.7	39
17	43.3	42
18	44.8	42
19	46.4	45
20	48.0	48

Besides overlapping of sample size points, another peculiarity that might occur without the NFRACTIONAL option is unequal spacing—for example, in the plot in Output 96.8.9, created with the following statements:

```
proc power plotonly;
   twosamplemeans test=diff
      groupmeans   = 12 | 18
      stddev       = 7
      groupweights = 2 | 1
      power        = .
      ntotal       = 20;
   plot x=n min=20 max=50 npoints=5;
run;
```

Output 96.8.9 Plot with Unequally Spaced Points

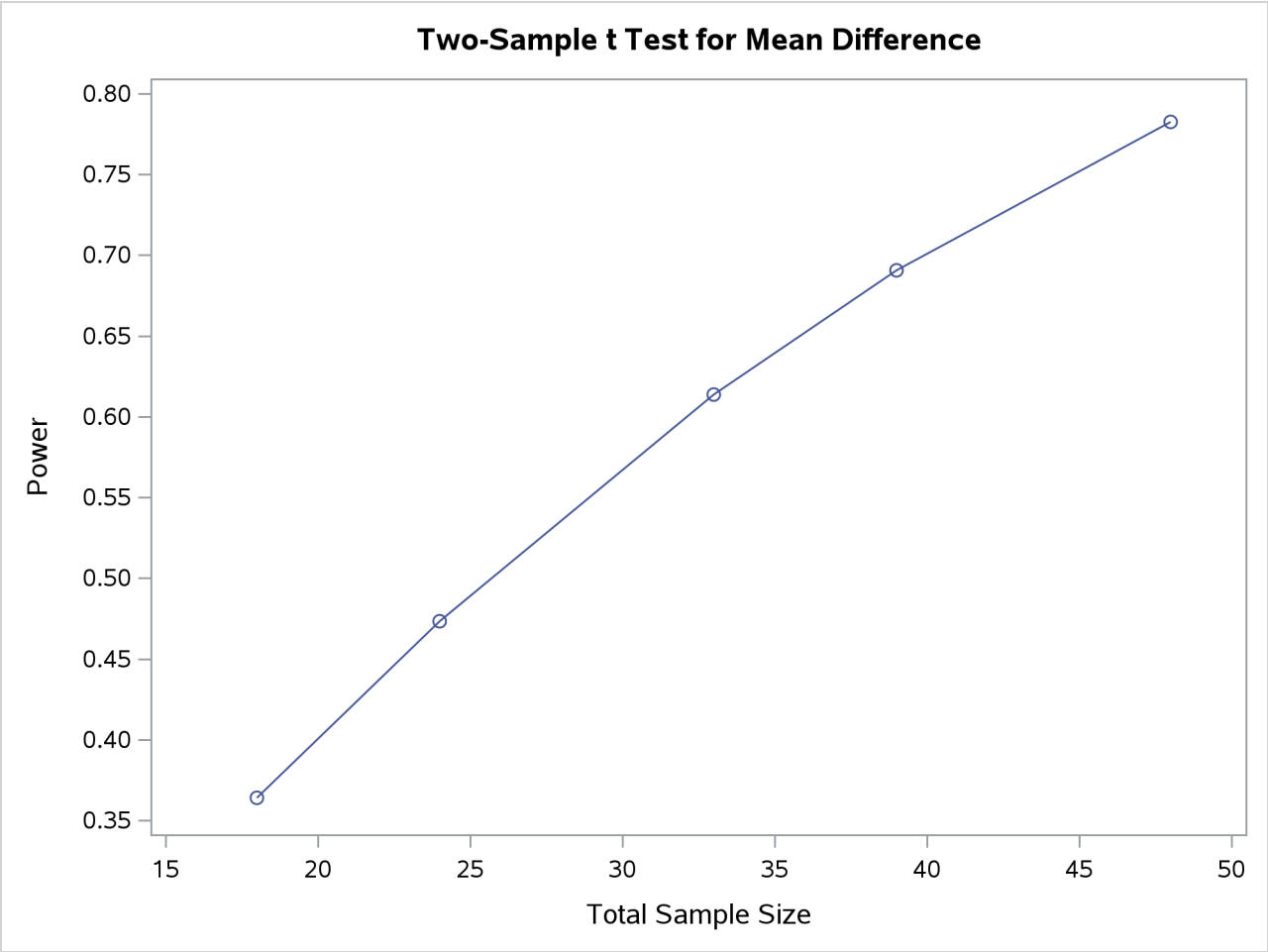

If you want to guarantee evenly spaced, nonoverlapping sample size points in your plots, you can either (1) use the NFRACTIONAL option in the analysis statement preceding the PLOT statement or (2) use the STEP= option and provide values for the MIN=, MAX=, and STEP= options in the PLOT statement that are multiples of the sum of the allocation weights. Note that this sum is simply 1 for one-sample and paired designs and 2 for balanced two-sample designs. So any integer step value works well for one-sample and paired designs, and any even step value works well for balanced two-sample designs. Both of these strategies will avoid rounding adjustments.

The following statements implement the first strategy to create the plot in Output 96.8.10, by using the NFRACTIONAL option in the TWOSAMPLEMEANS statement:

```
proc power plotonly;
   twosamplemeans test=diff
      nfractional
      groupmeans   = 12 | 18
      stddev       = 7
      groupweights = 2 | 1
      power        = .
      ntotal       = 20;
   plot x=n min=20 max=50 npoints=20;
```

```
run;
```

Output 96.8.10 Plot with Fractional Sample Sizes

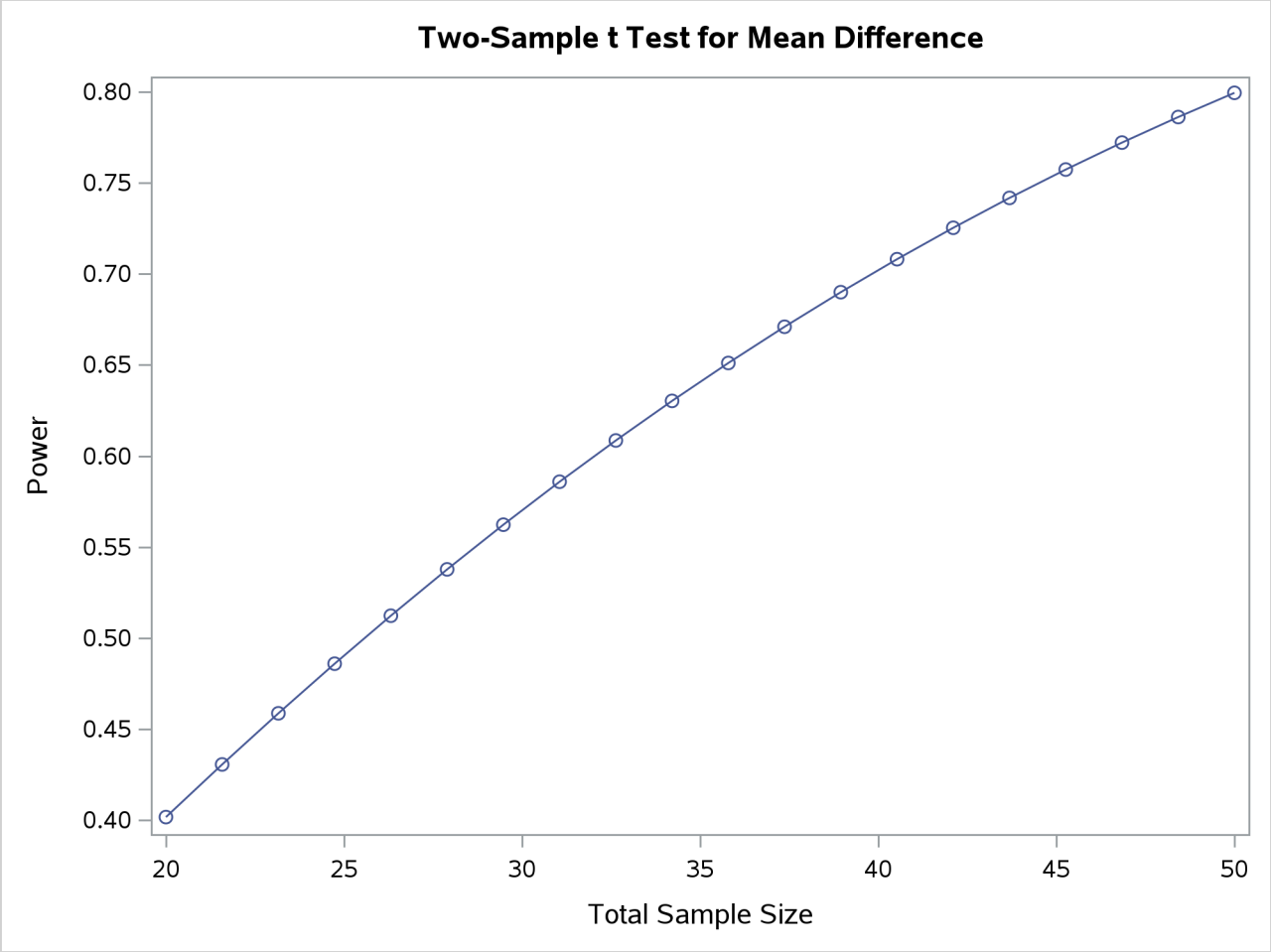

To implement the second strategy, use multiples of 3 for the STEP=, MIN=, and MAX= options in the PLOT statement (because the sum of the allocation weights is 2 + 1 = 3). The following statements use STEP=3, MIN=18, and MAX=48 to create a plot that looks identical to the plot in Output 96.8.7 but suffers no overlapping of points:

```
proc power plotonly;
   twosamplemeans test=diff
      groupmeans   = 12 | 18
      stddev       = 7
      groupweights = 2 | 1
      power        = .
      ntotal       = 20;
   plot x=n min=18 max=48 step=3;
run;
```

Adding Reference Lines

Suppose you want to add reference lines to highlight power=0.8 and power=0.9 on the plot in Output 96.8.5. You can add simple reference lines by using the YOPTS= option and REF= suboption in the PLOT statement to produce Output 96.8.11, with the following statements:

```
proc power plotonly;
   twosamplemeans test=diff
      groupmeans   = 12 | 15 18
      stddev       = 7 9
      power        = .
      ntotal       = 100;
   plot x=n min=20 max=500
      yopts=(ref=0.8 0.9);
run;
```

Output 96.8.11 Plot with Simple Reference Lines on Y Axis

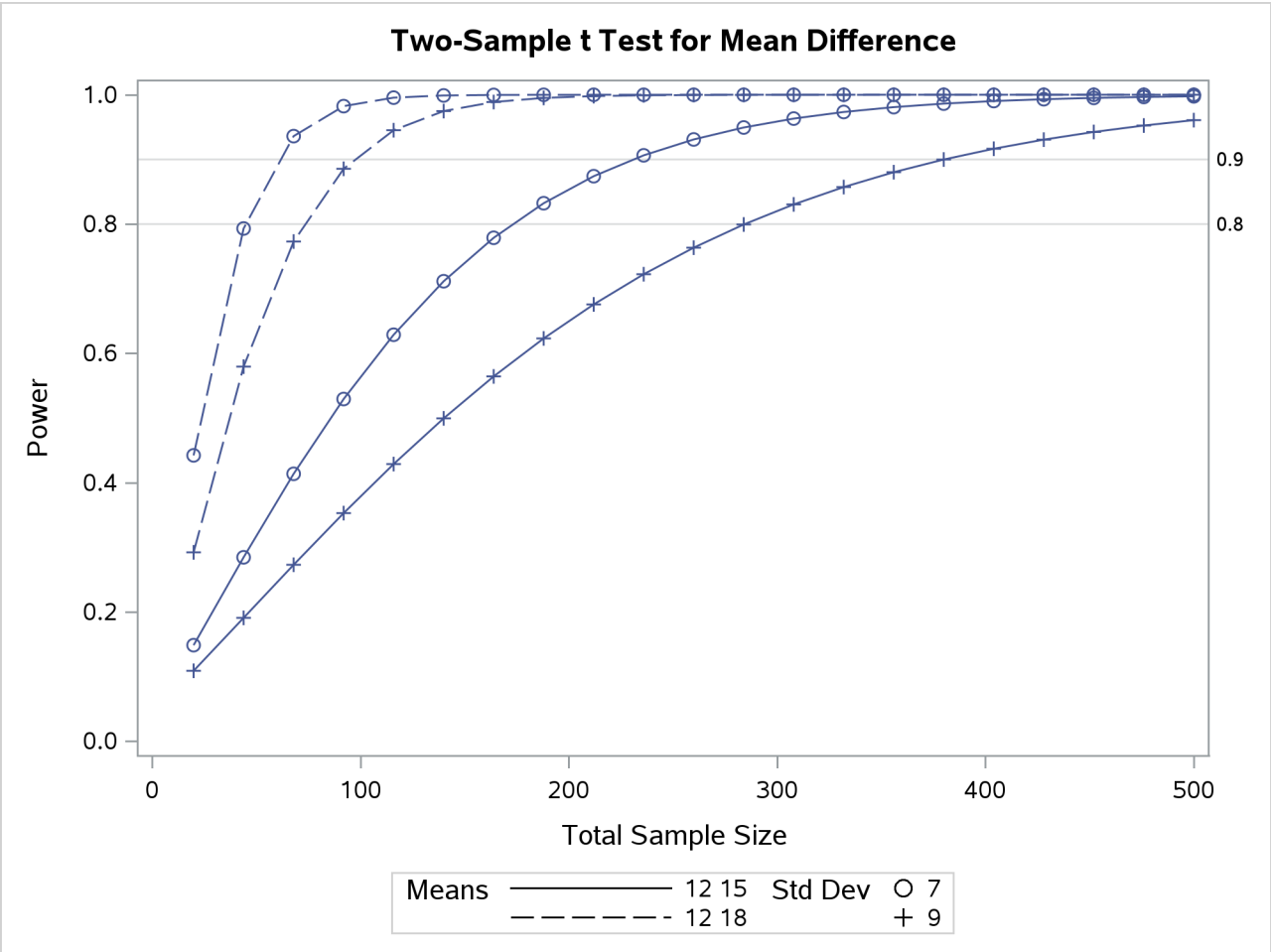

Or you can specify CROSSREF=YES to add reference lines that intersect each curve and cross over to the other axis:

```
plot x=n min=20 max=500
   yopts=(ref=0.8 0.9 crossref=yes);
```

The resulting plot is shown in Output 96.8.12.

Output 96.8.12 Plot with CROSSREF=YES Style Reference Lines from Y Axis

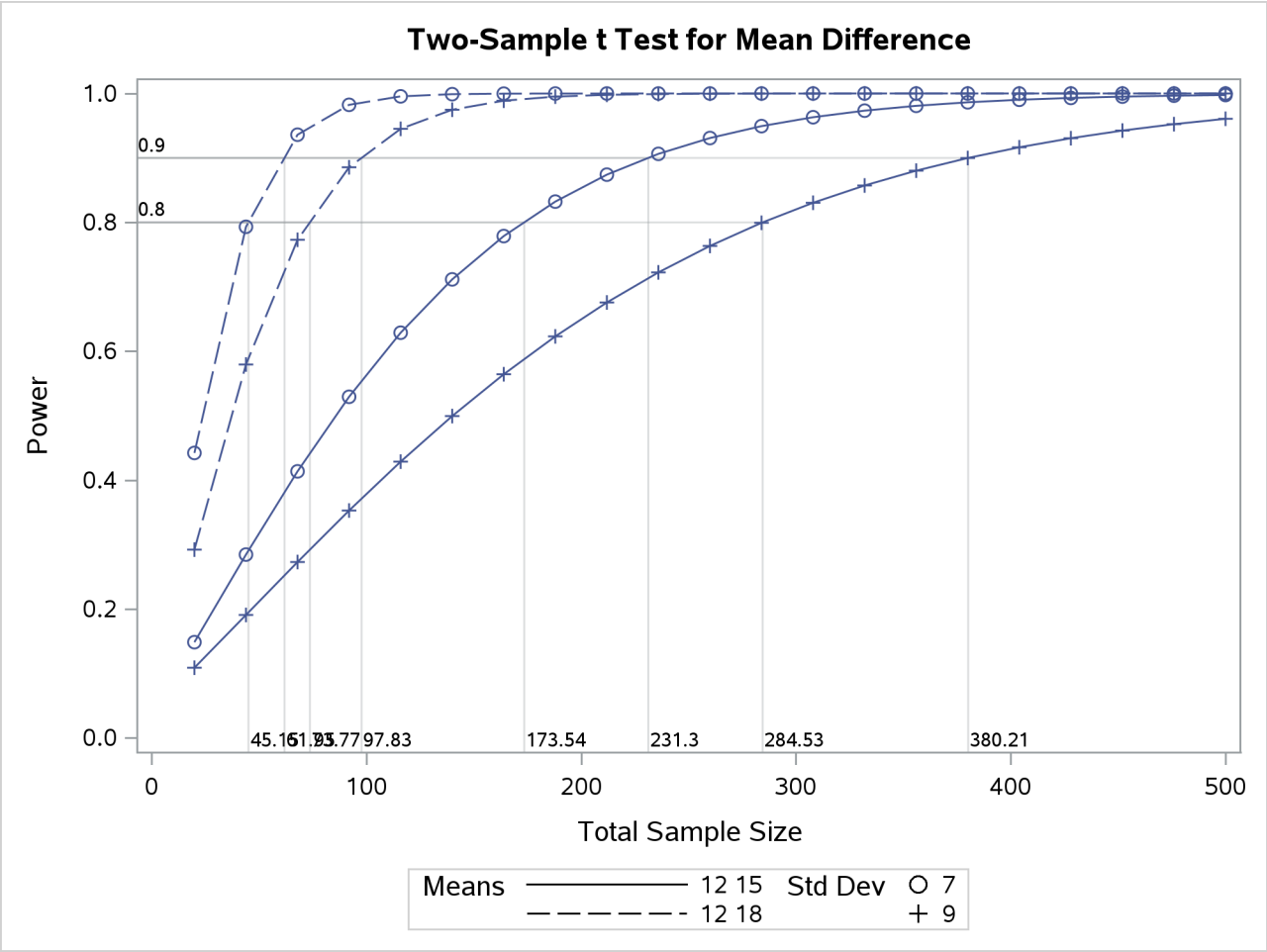

You can also add reference lines for the X axis by using the XOPTS= option instead of the YOPTS= option. For example, the following PLOT statement produces Output 96.8.13, which has crossing reference lines highlighting the sample size of 100:

```
plot x=n min=20 max=500
   xopts=(ref=100 crossref=yes);
```

Note that the values that label the reference lines at the X axis in Output 96.8.12 and at the Y axis in Output 96.8.13 are linearly interpolated from two neighboring points on the curves. Thus they might not exactly match corresponding values that are computed directly from the methods in the section "Computational Methods and Formulas" on page 8019—that is, computed by PROC POWER in the absence of a PLOT statement. The two ways of computing these values generally differ by a negligible amount.

Output 96.8.13 Plot with CROSSREF=YES Style Reference Lines from X Axis

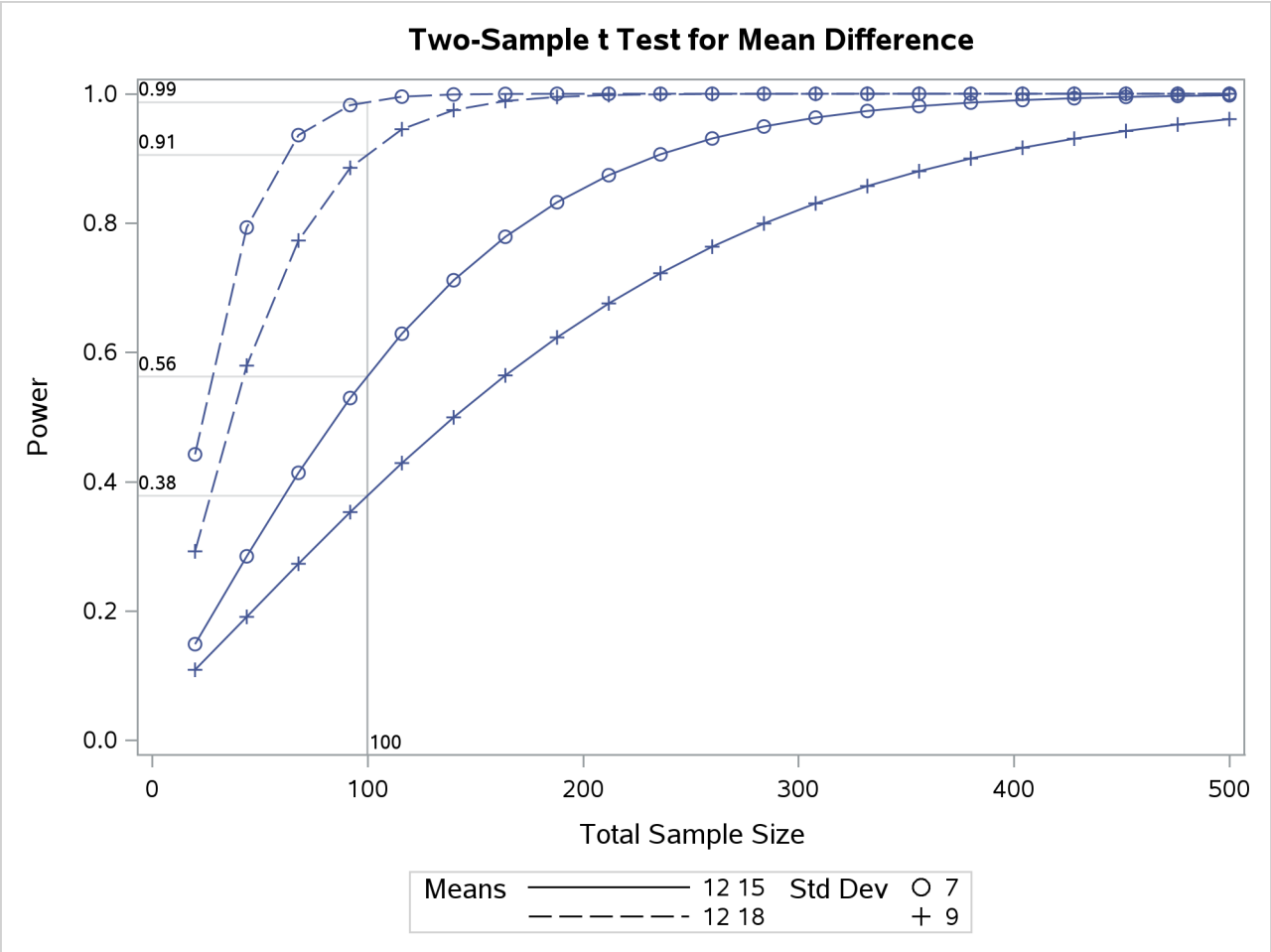

Linking Plot Features to Analysis Parameters

You can use the VARY option in the PLOT statement to specify which of the following features you want to associate with analysis parameters.

- line style
- plotting symbol
- color
- panel

You can specify mappings between each of these features and one or more analysis parameters, or you can simply choose a subset of these features to use (and rely on default settings to associate these features with multiple-valued analysis parameters).

Suppose you supplement the sample size analysis in Output 96.8.5 to include three values of alpha, by using the following statements:

```
proc power plotonly;
   twosamplemeans test=diff
      groupmeans   = 12 | 15 18
      stddev       = 7 9
      alpha        = 0.01 0.025 0.1
      power        = .
      ntotal       = 100;
   plot x=n min=20 max=500;
run;
```

The defaults for the VARY option in the PLOT statement specify line style varying by the ALPHA= parameter, plotting symbol varying by the GROUPMEANS= parameter, panel varying by the STDDEV= parameter, and color remaining constant. The resulting plot, consisting of two panels, is shown in Output 96.8.14.

Output 96.8.14 Plot with Default VARY Settings

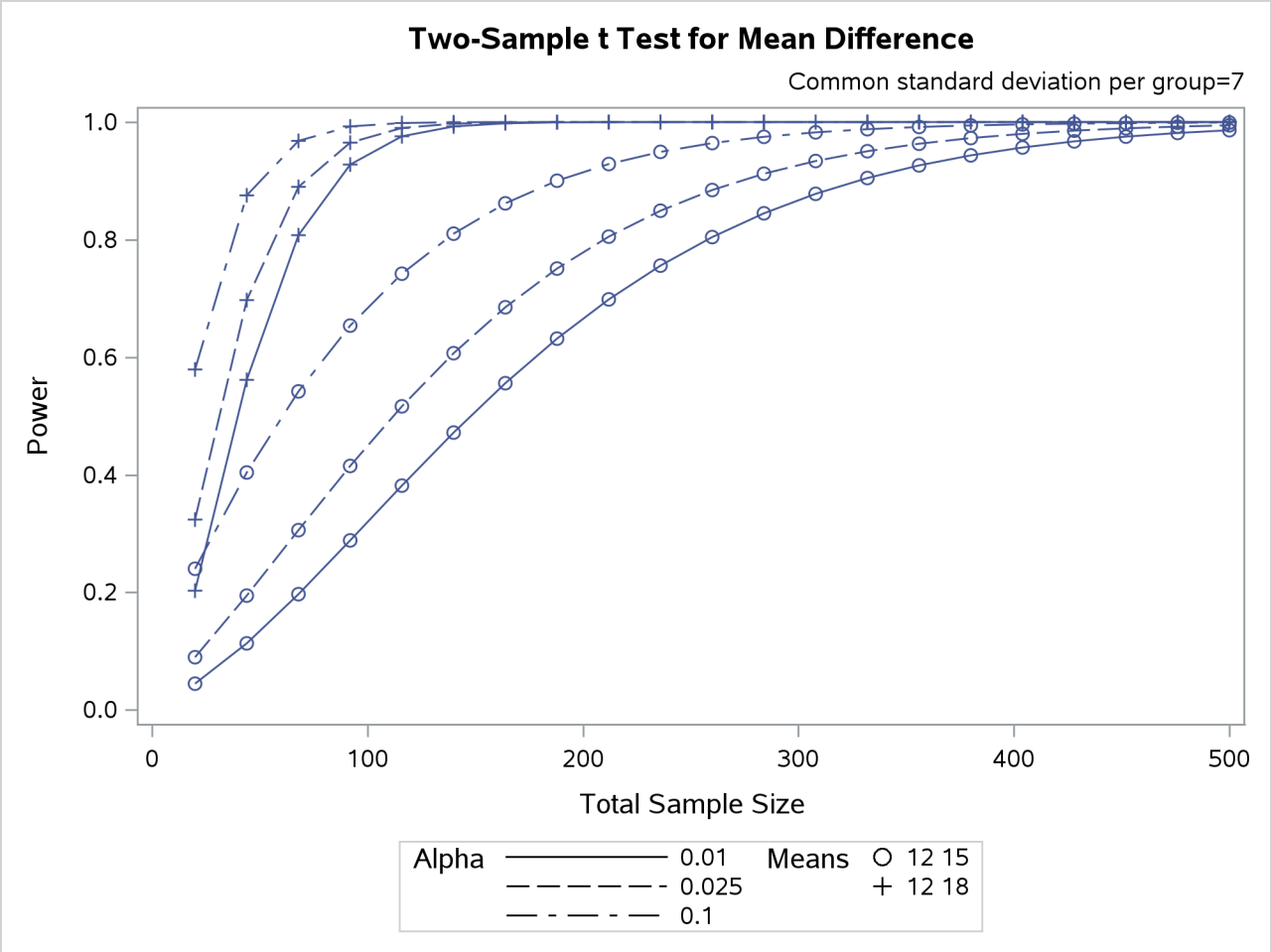

Output 96.8.14 *continued*

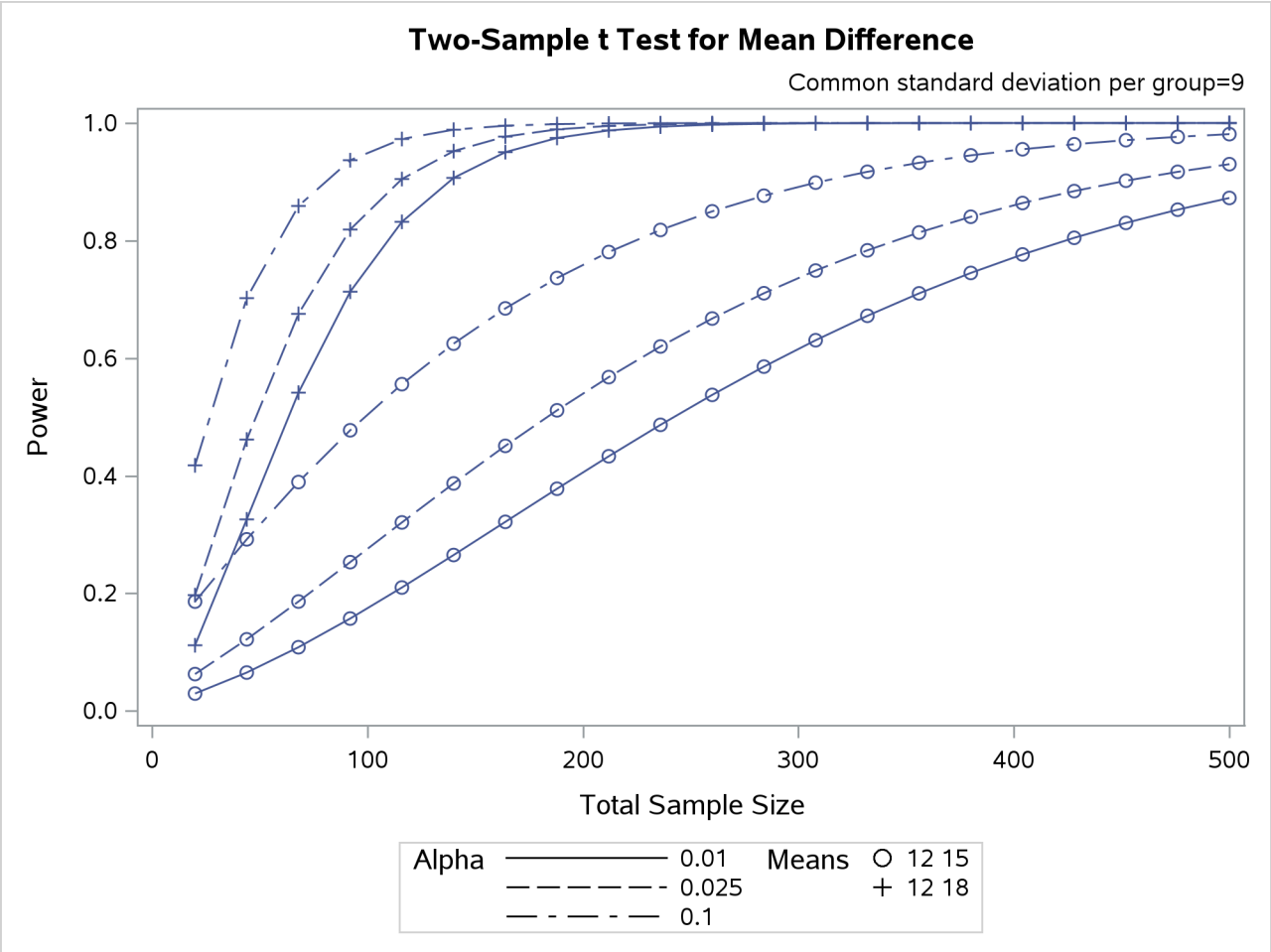

Suppose you want to produce a plot with only one panel that varies color in addition to line style and plotting symbol. Include the LINESTYLE, SYMBOL, and COLOR keywords in the VARY option in the PLOT statement, as follows, to produce the plot in Output 96.8.15:

```
plot x=n min=20 max=500
   vary (linestyle, symbol, color);
```

Output 96.8.15 Plot with Varying Color Instead of Panel

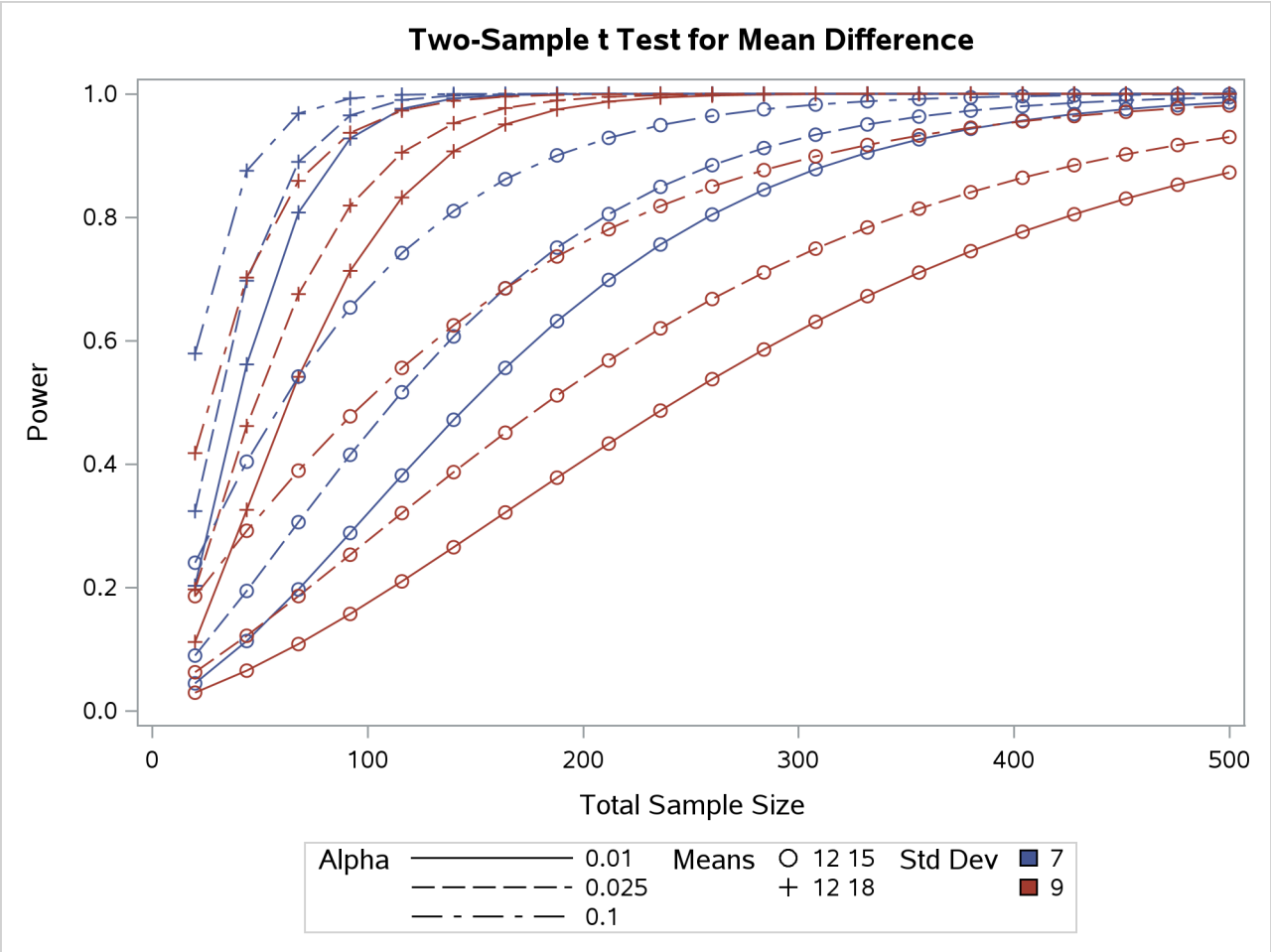

Finally, suppose you want to specify which features are used *and* which analysis parameters they are linked to. The following PLOT statement produces a two-panel plot (shown in Output 96.8.16) in which line style varies by standard deviation, plotting symbol varies by both alpha and sides, and panel varies by means:

```
plot x=n min=20 max=500
   vary (linestyle by stddev,
         symbol by alpha sides,
         panel by groupmeans);
```

Output 96.8.16 Plot with Features Explicitly Linked to Parameters

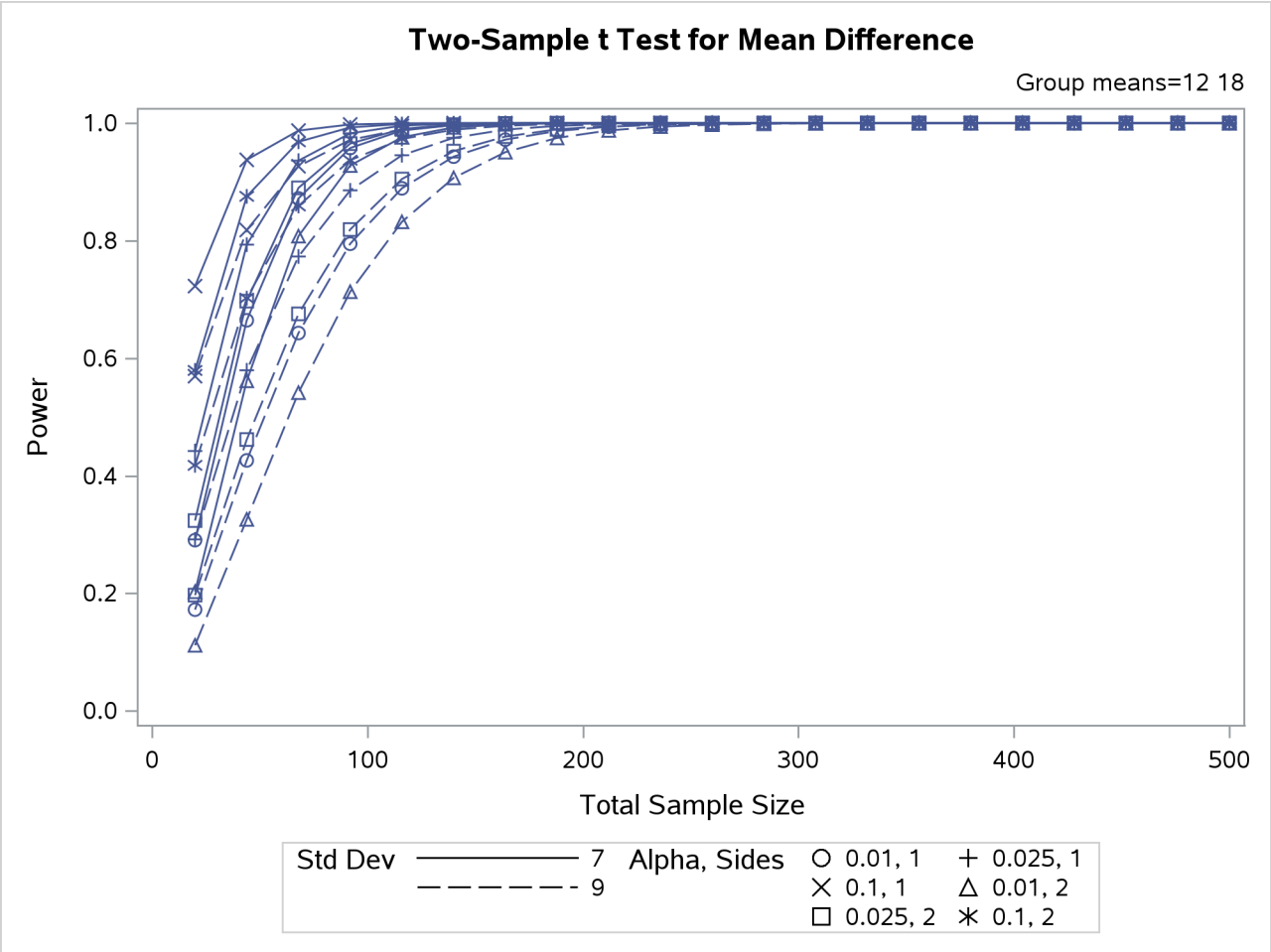

Output 96.8.16 *continued*

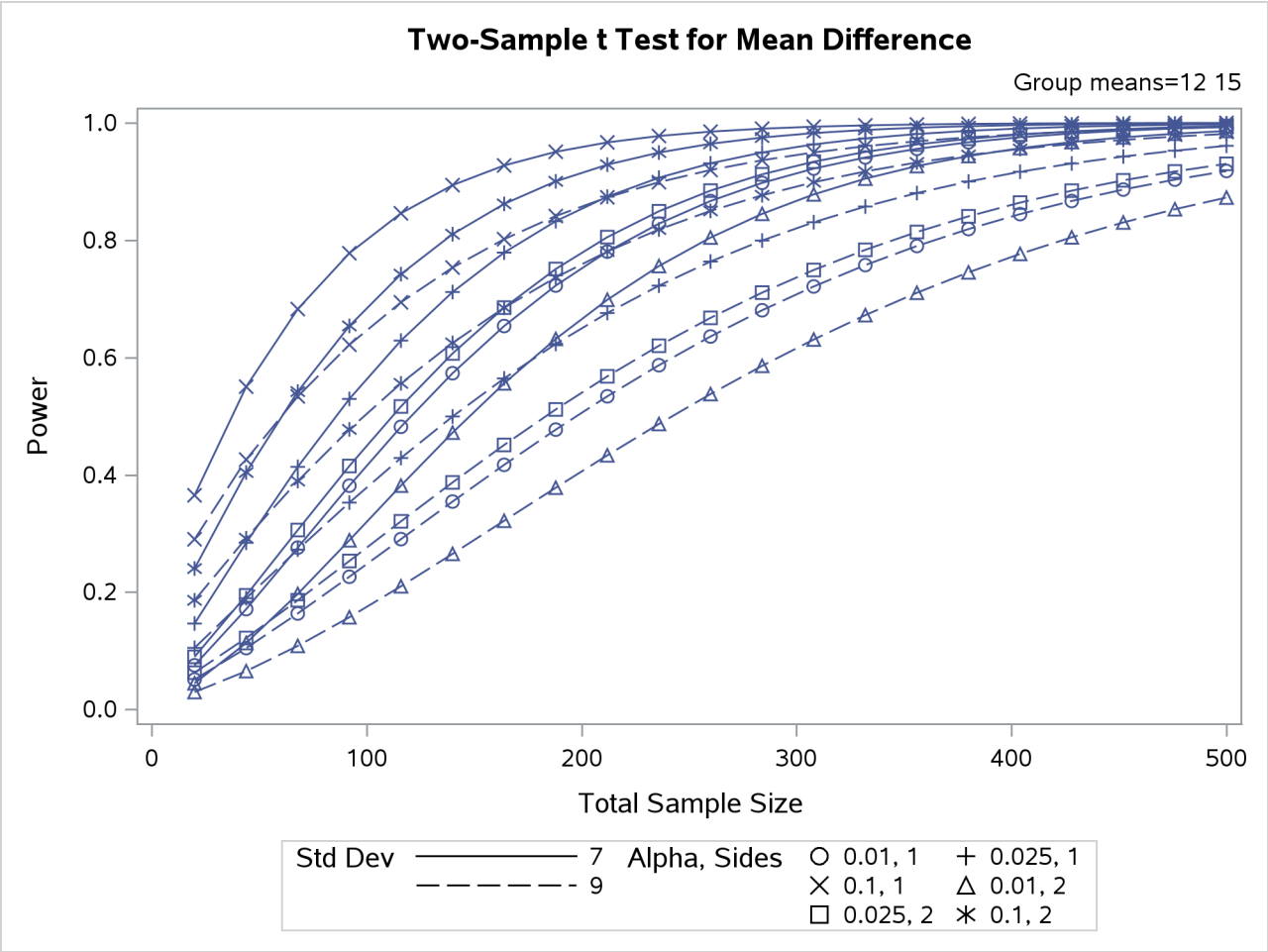

Choosing Key (Legend) Styles

The default style for the key (or "legend") is one that displays the association between levels of features and levels of analysis parameters, located below the X axis. For example, Output 96.8.5 demonstrates this style of key.

You can reproduce Output 96.8.5 with the same key but a different location, inside the plotting region, by using the POS=INSET option within the KEY=BYFEATURE option in the PLOT statement. The following statements product the plot in Output 96.8.17:

```
proc power plotonly;
   twosamplemeans test=diff
      groupmeans   = 12 | 15 18
      stddev       = 7 9
      power        = .
      ntotal       = 200;
   plot x=n min=20 max=500
      key = byfeature(pos=inset);
run;
```

Output 96.8.17 Plot with a By-Feature Key inside the Plotting Region

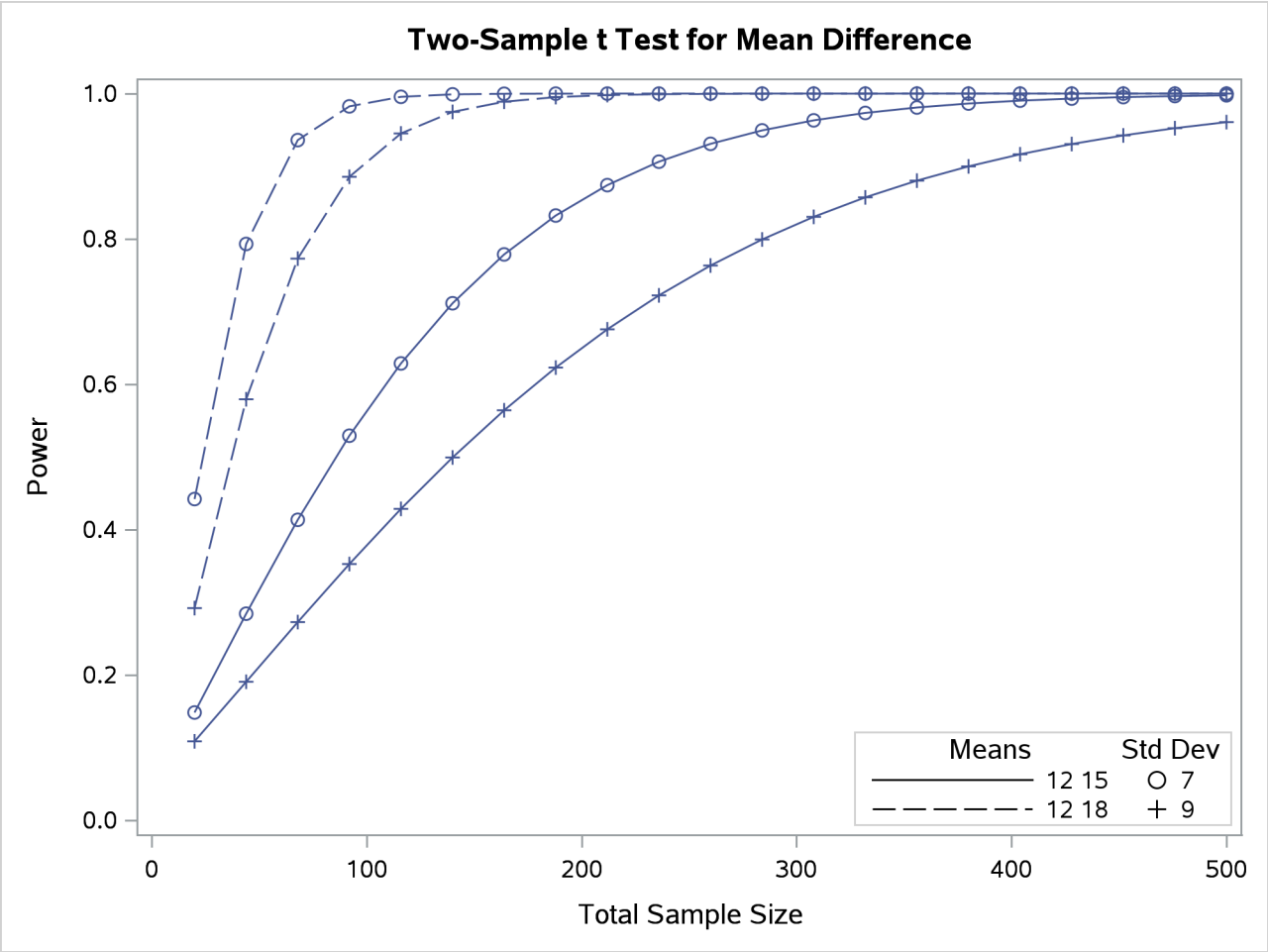

Alternatively, you can specify a key that identifies each individual curve separately by number by using the KEY=BYCURVE option in the PLOT statement:

```
plot x=n min=20 max=500
   key = bycurve;
```

The resulting plot is shown in Output 96.8.18.

Output 96.8.18 Plot with a Numbered By-Curve Key

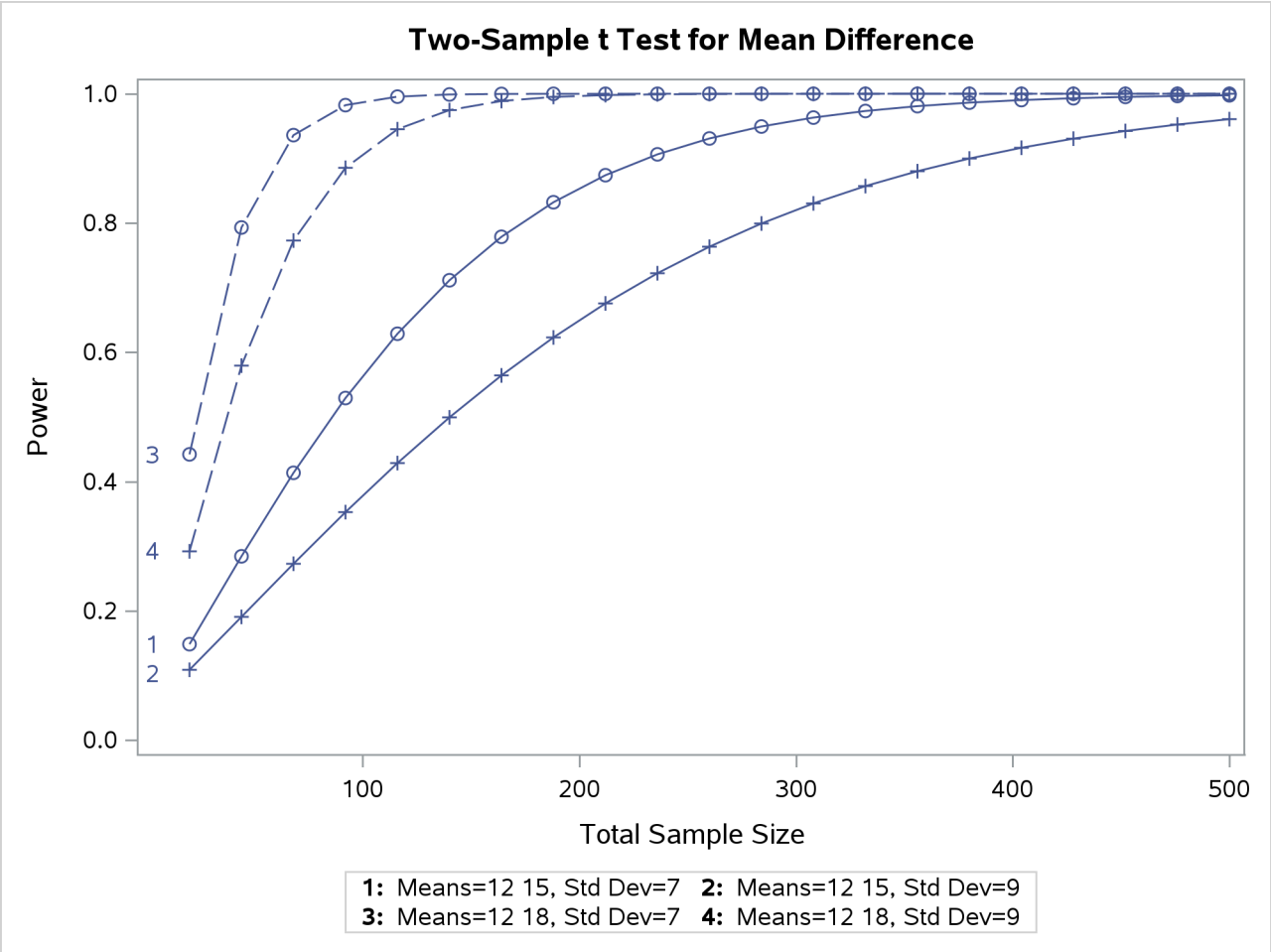

Use the NUMBERS=OFF option within the KEY=BYCURVE option to specify a nonnumbered key that identifies curves with samples of line styles, symbols, and colors:

```
plot x=n min=20 max=500
   key = bycurve(numbers=off pos=inset);
```

The POS=INSET suboption places the key within the plotting region. The resulting plot is shown in Output 96.8.19.

Output 96.8.19 Plot with a Nonnumbered By-Curve Key

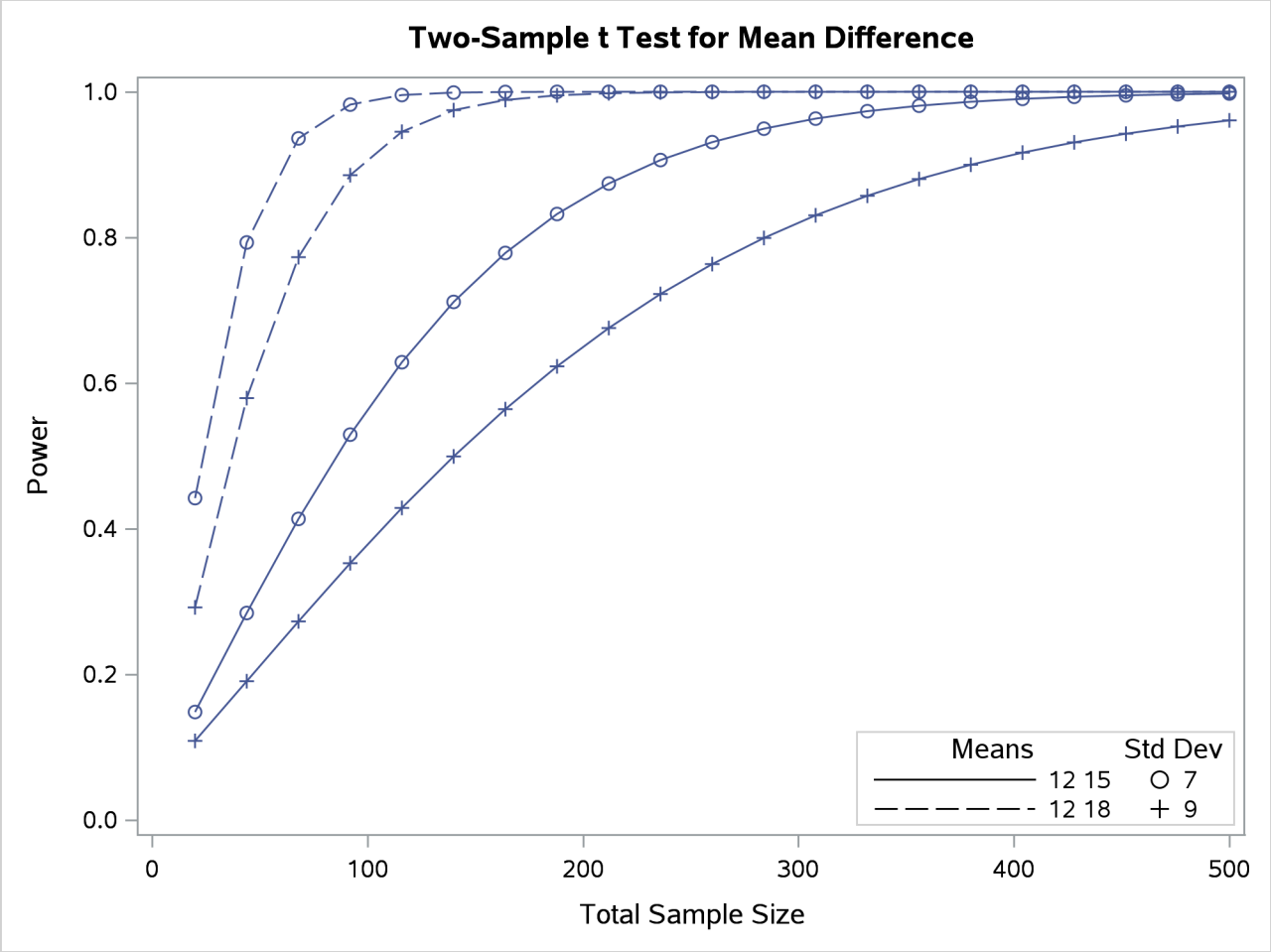

Finally, you can attach labels directly to curves with the KEY=ONCURVES option. The following PLOT statement produces Output 96.8.20:

```
plot x=n min=20 max=500
   key = oncurves;
```

Output 96.8.20 Plot with Directly Labeled Curves

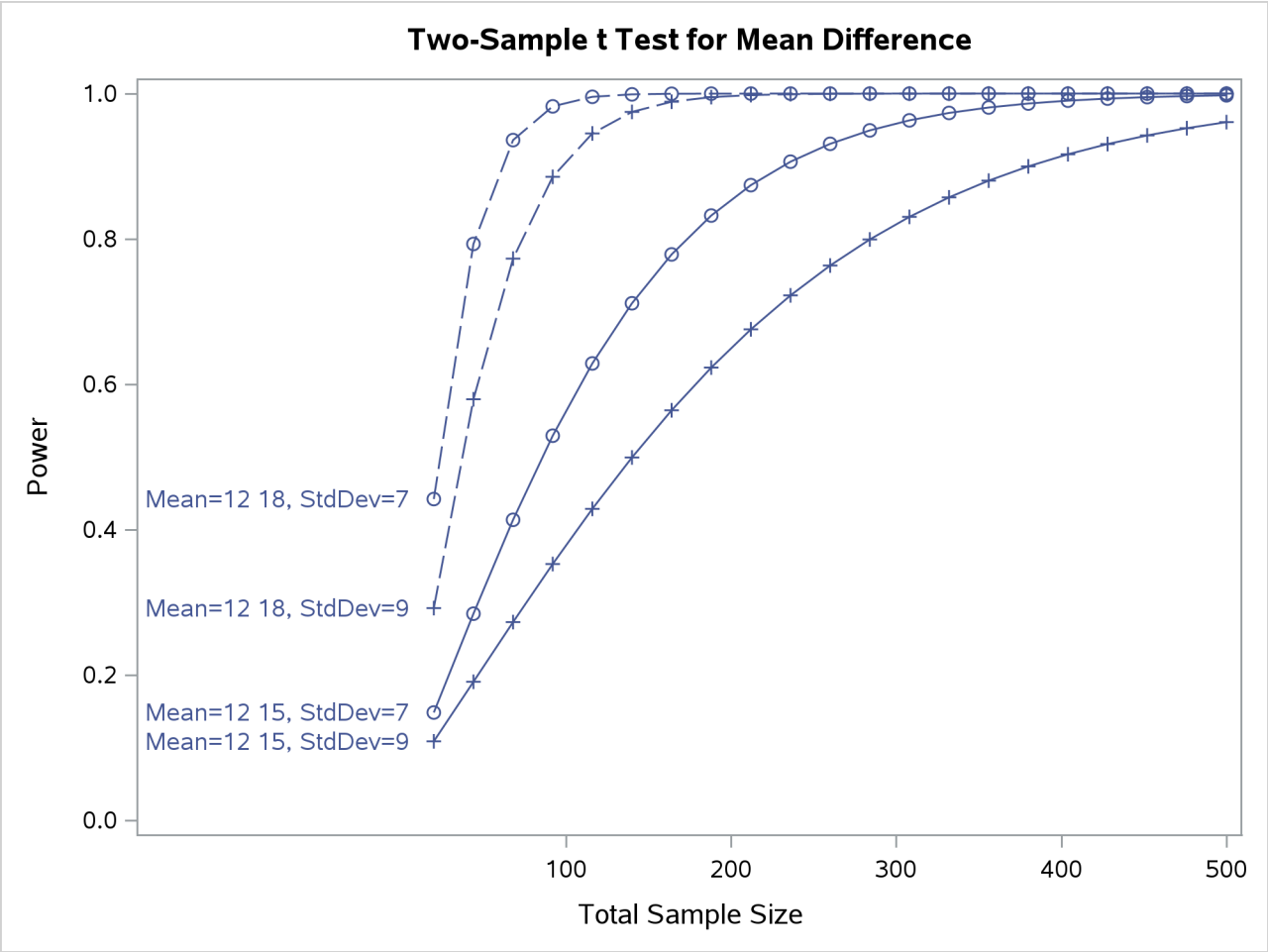

Modifying Symbol Locations

The default locations for plotting symbols are the points computed directly from the power and sample size algorithms. For example, Output 96.8.5 shows plotting symbols corresponding to computed points. The curves connecting these points are interpolated (as indicated by the INTERPOL= option in the PLOT statement).

You can modify the locations of plotting symbols by using the MARKERS= option in the PLOT statement. The MARKERS=ANALYSIS option places plotting symbols at locations corresponding to the input specified in the analysis statement preceding the PLOT statement. You might prefer this as an alternative to using reference lines to highlight specific points. For example, you can reproduce Output 96.8.5, but with the plotting symbols located at the sample sizes shown in Output 96.8.1, by using the following statements:

```
proc power plotonly;
   twosamplemeans test=diff
      groupmeans   = 12 | 15 18
      stddev       = 7 9
      power        = .
      ntotal       = 232 382 60 98;
```

```
    plot x=n min=20 max=500
       markers=analysis;
run;
```

The analysis statement here is the TWOSAMPLEMEANS statement. The MARKERS=ANALYSIS option in the PLOT statement causes the plotting symbols to occur at sample sizes specified by the NTOTAL= option in the TWOSAMPLEMEANS statement: 232, 382, 60, and 98. The resulting plot is shown in Output 96.8.21.

Output 96.8.21 Plot with MARKERS=ANALYSIS

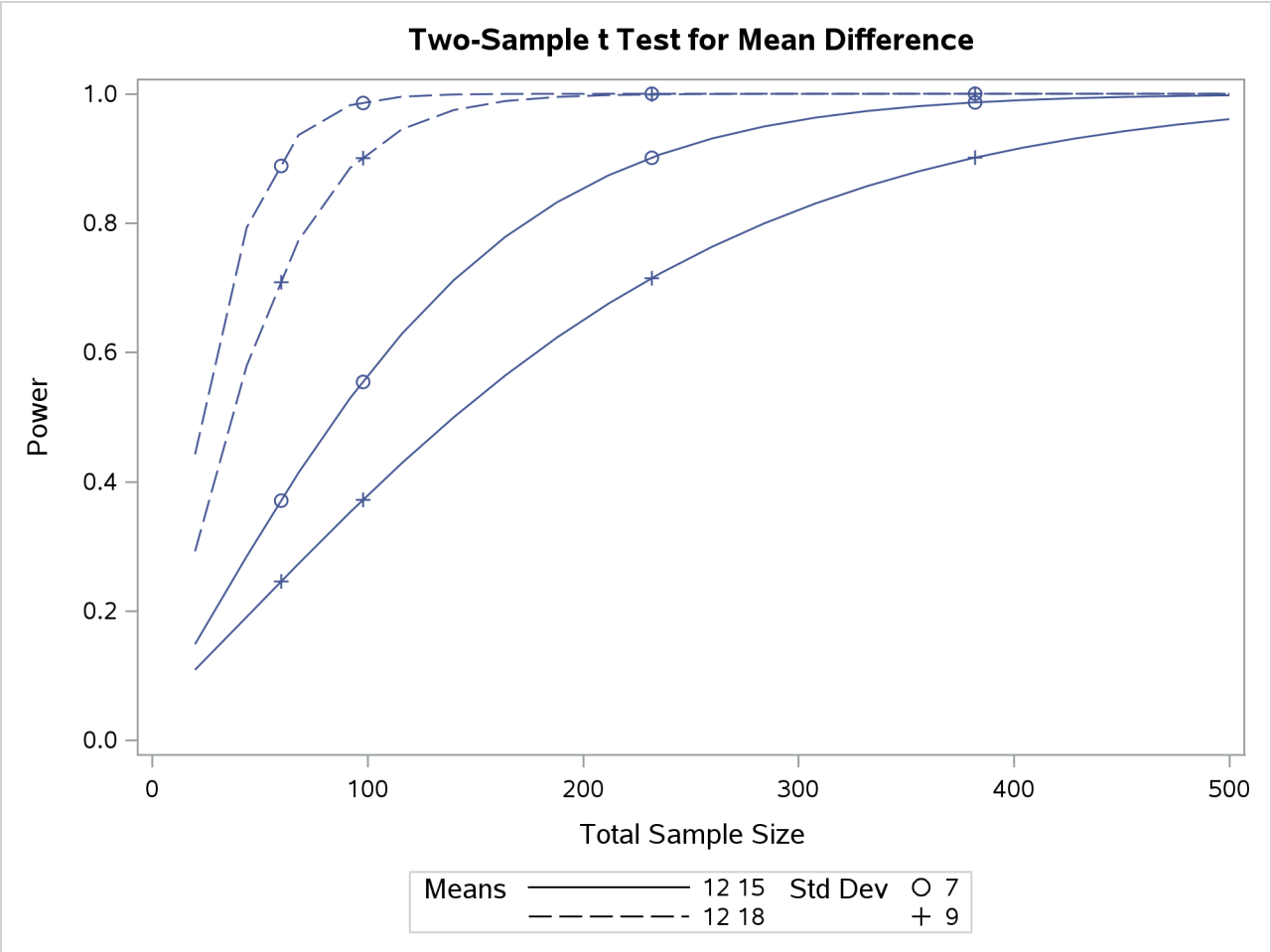

You can also use the MARKERS=NICE option to align symbols with the tick marks on one of the axes (the X axis when the X= option is used, or the Y axis when the Y= option is used):

```
    plot x=n min=20 max=500
       markers=nice;
```

The plot created by this PLOT statement is shown in Output 96.8.22.

Output 96.8.22 Plot with MARKERS=NICE

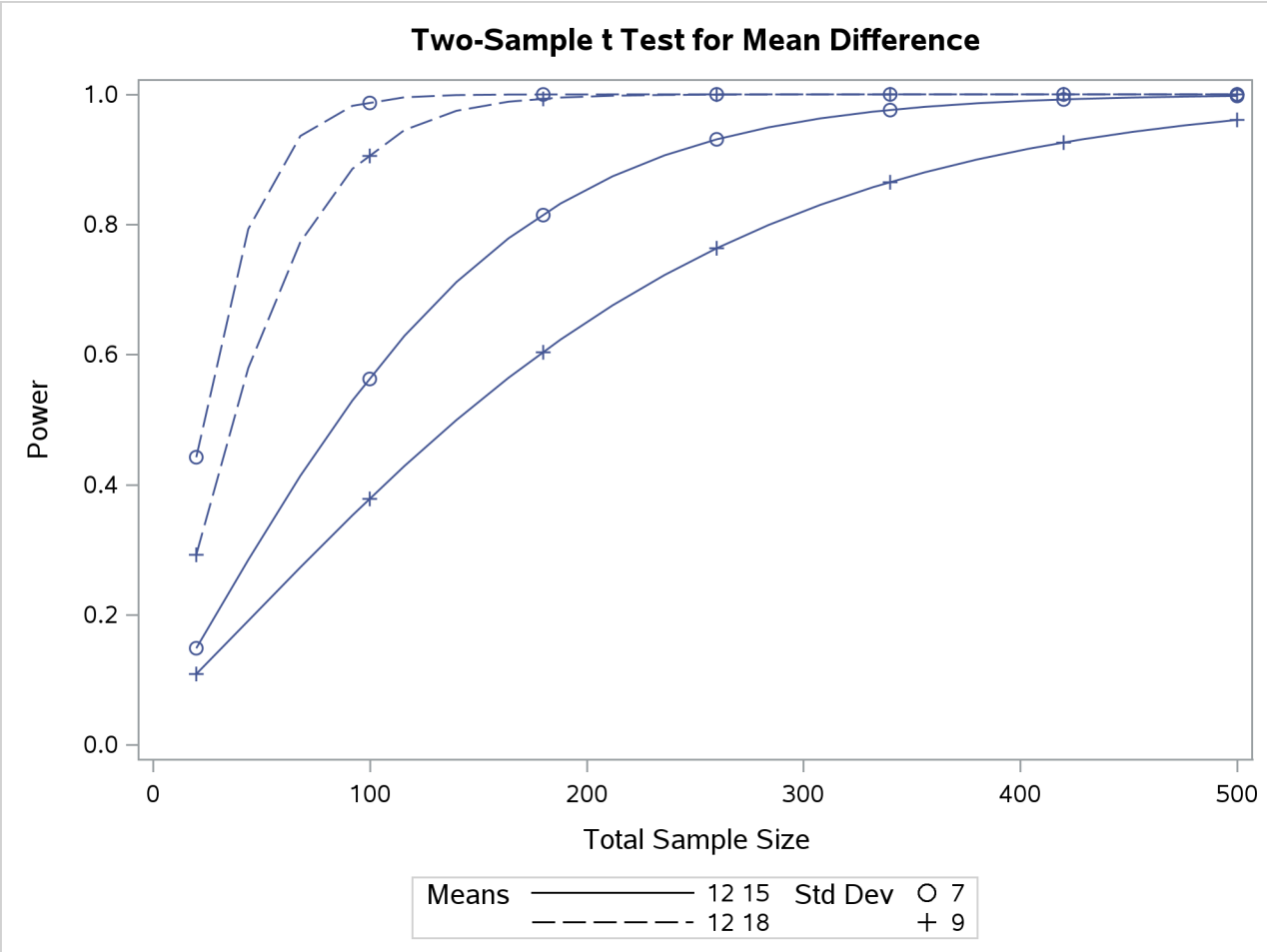

Note that the plotting symbols are aligned with the tick marks on the X axis because the X= option is specified.

Example 96.9: Binary Logistic Regression with Independent Predictors

(View the complete code for this example (pwrex09.sas) in the example repository.)

Suppose you are planning an industrial experiment similar to the analysis in "Getting Started: LOGISTIC Procedure" on page 5942 in Chapter 79, "The LOGISTIC Procedure," but for a different type of ingot. The primary test of interest is the likelihood ratio chi-square test of the effect of heating time on the readiness of the ingots for rolling. Ingots will be randomized independently into one of four different heating times (5, 10, 15, and 20 minutes) with allocation ratios 2:3:3:2 and three different soaking times (2, 4, and 6 minutes) with allocation ratios 2:2:1. The mass of each ingot will be measured as a covariate.

You want to know how many ingots you must sample to have a 90% chance of detecting an odds ratio as small as 1.2 for a five-minute heating time increase. The odds ratio is defined here as the odds of the ingot not being ready given a heating time of h minutes divided by the odds given a heating time of $h - 5$ minutes,

for any time h. You will use a significance level of $\alpha = 0.1$ to balance Type I and Type II errors since you consider their importance to be roughly equal.

The distributions of heating time and soaking time are determined by the design, but you must conjecture the distribution of ingot mass. Suppose you expect its distribution to be approximately normal with mean 4 kg and standard deviation between 1 kg and 2 kg.

You are powering the study for an odds ratio of 1.2 for the heating time, but you must also conjecture odds ratios for soaking time and mass. You suspect that the odds ratio for a unit increase in soaking time is about 1.4, and the odds ratio for a unit increase in mass is between 1 and 1.3.

Finally, you must provide a guess for the average probability of an ingot not being ready for rolling, averaged across all possible design profiles. Existing data suggest that this probability lies between 0.15 and 0.25.

You decide to evaluate sample size at the two extremes of each parameter for which you conjectured a range. Use the following statements to perform the sample size determination:

```
proc power;
   logistic
      vardist("Heat") = ordinal((5 10 15 20) : (0.2 0.3 0.3 0.2))
      vardist("Soak") = ordinal((2 4 6) : (0.4 0.4 0.2))
      vardist("Mass1") = normal(4, 1)
      vardist("Mass2") = normal(4, 2)
      testpredictor = "Heat"
      covariates = "Soak" | "Mass1" "Mass2"
      responseprob = 0.15 0.25
      testoddsratio = 1.2
      units= ("Heat" = 5)
      covoddsratios = 1.4 | 1 1.3
      alpha = 0.1
      power = 0.9
      ntotal = .;
run;
```

The VARDIST= option is used to define the distributions of the predictor variables. The distributions of heating and soaking times are defined by the experimental design, with ordinal probabilities derived from the allocation ratios. The two conjectured standard deviations for the ingot mass are represented in the Mass1 and Mass2 distributions. The TESTPREDICTOR= option identifies the predictor being tested, and the COVARIATES= option specifies the scenarios for the remaining predictors in the model (soaking time and mass). The RESPONSEPROB= option specifies the overall response probability, and the TESTODDSRATIO= and UNITS= options indicate the odds ratio and increment for heating time. The COVODDSRATIOS= option specifies the scenarios for the odds ratios of soaking time and mass. The default DEFAULTUNIT=1 option specifies a unit change for both of these odds ratios. The ALPHA= option sets the significance level, and the POWER= option defines the target power. Finally, the NTOTAL= option with a missing value (.) identifies the parameter to solve for.

Output 96.9.1 shows the results.

Output 96.9.1 Sample Sizes for Test of Heating Time in Logistic Regression

The POWER Procedure
Likelihood Ratio Chi-square Test for One Predictor

Fixed Scenario Elements	
Method	Shieh-O'Brien approximation
Alpha	0.1
Test Predictor	Heat
Odds Ratio for Test Predictor	1.2
Unit for Test Pred Odds Ratio	5
Nominal Power	0.9

Computed N Total

Index	Response Prob	Covariates	Cov ORs	Cov Units	Total N Bins	Actual Power	N Total
1	0.15	Soak Mass1	1.4 1.0	1 1	120	0.900	1878
2	0.15	Soak Mass1	1.4 1.3	1 1	120	0.900	1872
3	0.15	Soak Mass2	1.4 1.0	1 1	120	0.900	1878
4	0.15	Soak Mass2	1.4 1.3	1 1	120	0.900	1857
5	0.25	Soak Mass1	1.4 1.0	1 1	120	0.900	1342
6	0.25	Soak Mass1	1.4 1.3	1 1	120	0.900	1348
7	0.25	Soak Mass2	1.4 1.0	1 1	120	0.900	1342
8	0.25	Soak Mass2	1.4 1.3	1 1	120	0.900	1369

The required sample size ranges from 1342 to 1878, depending on the unknown true values of the overall response probability, mass standard deviation, and soaking time odds ratio. The overall response probability clearly has the largest influence among these parameters, with a sample size increase of almost 40% going from 0.25 to 0.15.

Example 96.10: Wilcoxon-Mann-Whitney Test

(View the complete code for this example (pwrex10.sas) in the example repository.)

Consider a hypothetical clinical trial to treat interstitial cystitis (IC), a painful, chronic inflammatory condition of the bladder with no known cause that most commonly affects women. Two treatments will be compared: lidocaine alone (“lidocaine”) versus lidocaine plus a fictitious experimental drug called Mironel (“Mir+lido”). The design is balanced, randomized, double-blind, and female-only. The primary outcome is a measure of overall improvement at week 4 of the study, measured on a seven-point Likert scale as shown in Table 96.40.

Table 96.40 Self-Report Improvement Scale

Compared to when I started this study, my condition is:	
Much worse	–3
Worse	–2
Slightly worse	–1
The same	0
Slightly better	+1
Better	+2
Much better	+3

The planned data analysis is a one-sided Wilcoxon-Mann-Whitney test with $\alpha = 0.05$ where the alternative hypothesis represents greater improvement for "Mir+lido."

You are asked to graphically assess the power of the planned trial for sample sizes between 100 and 250, assuming that the conditional outcome probabilities given treatment are equal to the values in Table 96.41.

Table 96.41 Conjectured Conditional Probabilities

	Response						
Treatment	–3	–2	–1	0	+1	+2	+3
Lidocaine	0.01	0.04	0.20	0.50	0.20	0.04	0.01
Mir+lido	0.01	0.03	0.15	0.35	0.30	0.10	0.06

Use the following statements to compute the power at sample sizes of 100 and 250 and generate a power curve:

```
ods graphics on;

proc power;
   twosamplewilcoxon
      vardist("lidocaine") = ordinal ((-3 -2 -1 0 1 2 3) :
                                      (.01 .04 .20 .50 .20 .04 .01))
      vardist("Mir+lido") = ordinal ((-3 -2 -1 0 1 2 3) :
                                      (.01 .03 .15 .35 .30 .10 .06))
      variables = "lidocaine" | "Mir+lido"
      sides = u
      ntotal = 100 250
      power = .;
   plot step=10;
run;

ods graphics off;
```

The VARDIST= option is used to define the distribution for each treatment, and the VARIABLES= option specifies the distributions to compare. The SIDES=U option corresponds to the alternative hypothesis that the second distribution ("Mir+lido") is more favorable. The NTOTAL= option specifies the total sample sizes

of interest, and the POWER= option with a missing value (.) identifies the parameter to solve for. The default GROUPWEIGHTS= and ALPHA= options specify a balanced design and significance level $\alpha = 0.05$.

The STEP=10 option in the PLOT statement requests a point for each sample size increment of 10. The default values for the X=, MIN=, and MAX= plot options specify a sample size range of 100 to 250 (the same as in the analysis) for the X axis.

The tabular and graphical results are shown in Output 96.10.1 and Output 96.10.2, respectively.

Output 96.10.1 Power Values for Wilcoxon-Mann-Whitney Test

The POWER Procedure
Wilcoxon-Mann-Whitney Test

Fixed Scenario Elements	
Method	O'Brien-Castelloe approximation
Number of Sides	U
Group 1 Variable	lidocaine
Group 2 Variable	Mir+lido
Pooled Number of Bins	7
Alpha	0.05
Group 1 Weight	1
Group 2 Weight	1
NBins per Group	1000

Computed Power		
	N	
Index	**Total**	**Power**
1	100	0.651
2	250	0.939

Output 96.10.2 Plot of Power versus Sample Size for Wilcoxon Power Analysis

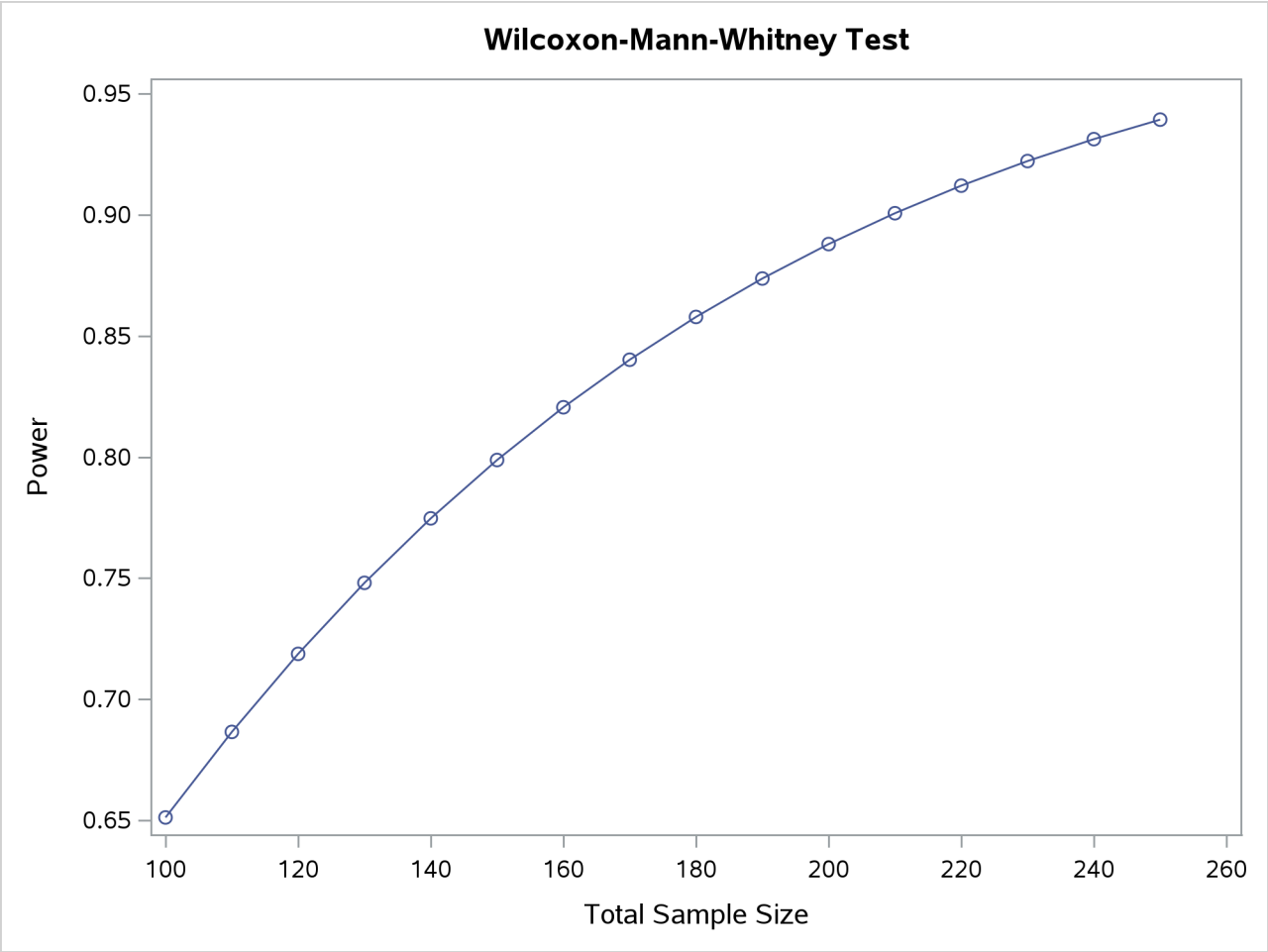

The achieved power ranges from 0.651 to 0.939, increasing with sample size.

Example 96.11: Logistic Regression Using the CUSTOM Statement

(View the complete code for this example (pwrex11.sas) in the example repository.)

The framework in Lyles, Lin, and Williamson (2007) provides an effective strategy for using the CUSTOM statement to compute power or sample size for generalized linear models. The process involves the following steps:

1. Specify the SAS code that you would use to perform the data analysis.
2. Create an exemplary data set that resembles the nature of the data you expect to obtain, covering as many reasonable values of the response variable as possible.
3. Run the analysis on the exemplary data set and extract the appropriate test statistic.
4. If you are planning for a likelihood ratio test, run the analysis again, but this time with a reduced model (lacking the predictors being tested). Again extract the appropriate test statistic.

5 Run PROC POWER with the CUSTOM statement, specifying the noncentral chi-square distribution with primary noncentrality equal to the appropriate transformation of the test statistics.

Although the power computation is conditional (assuming fixed predictors), you can approximate unconditional power by using a WEIGHT statement in the data analysis and constructing the exemplary data to be representative of the conjectured distribution of the predictors.

Suppose you want to compute the power of the Wald chi-square and likelihood ratio tests for the interaction term in a logistic regression model that has two binary covariates, as in Lyles, Lin, and Williamson (2007, section 3.2). You plan to use a sample size $N = 100$ and significance level $\alpha = 0.05$. You have conjectured values for the β_i regression coefficients as shown in Table 96.42.

Table 96.42 Conjectured Regression Coefficient Values

Source	Coefficient	Value
Intercept	β_0	0
X1	β_1	–1.5
X2	β_2	0.5
X1*X2	β_3	2

You expect the covariates to be distributed according to a multinomial distribution with probabilities as shown in Table 96.43.

Table 96.43 Conjectured Covariate Distribution

X1	X2	Probability
0	0	0.2
0	1	0.25
1	0	0.25
1	1	0.3

The strategy for the power computation is summarized as follows:

1 Create an exemplary data set to represent both the β regression coefficients and the covariate distribution.

2 Use the LOGISTIC procedure to compute the primary noncentrality values and degrees of freedom you need for the power calculation.

3 Fetch the primary noncentrality values and degrees of freedom from PROC LOGISTIC output and pass them into the CUSTOM statement in PROC POWER along with the other power analysis parameters.

The following steps describe this strategy in detail:

1 First create a data set that has one row for each unique design profile and compute the response probability that is associated with each profile according to your conjectured regression coefficients values. The following DATA step computes these probabilities as values of a variable PY:

```
* Compute P(Yj|Xi) values from conjectured betas;
data Exemplary;
   _B0 = 0; _B1 = -1.5; _B2 = 0.5; _B3 = 2;
   do X1 = 0 to 1;
      do X2 = 0 to 1;
         _pi = logistic(_B0 + _B1*X1 + _B2*X2 + _B3*X1*X2);
         do Y = 0 to 1;
            PY = _pi**Y * (1-_pi)**(1-Y);
            output;
         end;
      end;
   end;
   keep X1 X2 Y PY;
run;


proc print data=Exemplary;
run;
```

The output data set is shown in Output 96.11.1.

Output 96.11.1 Response Probabilities Produced from Regression Coefficients

Obs	X1	X2	Y	PY
1	0	0	0	0.50000
2	0	0	1	0.50000
3	0	1	0	0.37754
4	0	1	1	0.62246
5	1	0	0	0.81757
6	1	0	1	0.18243
7	1	1	0	0.26894
8	1	1	1	0.73106

Next expand this data set into an exemplary data set that represents your assumed covariate distribution. The following DATA step replicates each design profile the appropriate number of times according to the multinomial probabilities in Table 96.43.

```
* Expand according to relative allocations of design profiles;
data Exemplary;
   set Exemplary;
   if      (X1 = 0 and X2 = 0) then _alloc=4;
   else if (X1 = 0 and X2 = 1) then _alloc=5;
   else if (X1 = 1 and X2 = 0) then _alloc=5;
   else                             _alloc=6;
   do _i = 1 to _alloc;
      output;
   end;
   keep X1 X2 Y PY;
run;
```

```
proc print data=Exemplary;
run;
```

Output 96.11.2 shows the resulting exemplary data set.

Output 96.11.2 Exemplary Data Set

Obs	X1	X2	Y	PY
1	0	0	0	0.50000
2	0	0	0	0.50000
3	0	0	0	0.50000
4	0	0	0	0.50000
5	0	0	1	0.50000
6	0	0	1	0.50000
7	0	0	1	0.50000
8	0	0	1	0.50000
9	0	1	0	0.37754
10	0	1	0	0.37754
11	0	1	0	0.37754
12	0	1	0	0.37754
13	0	1	0	0.37754
14	0	1	1	0.62246
15	0	1	1	0.62246
16	0	1	1	0.62246
17	0	1	1	0.62246
18	0	1	1	0.62246
19	1	0	0	0.81757
20	1	0	0	0.81757
21	1	0	0	0.81757
22	1	0	0	0.81757
23	1	0	0	0.81757
24	1	0	1	0.18243
25	1	0	1	0.18243
26	1	0	1	0.18243
27	1	0	1	0.18243
28	1	0	1	0.18243
29	1	1	0	0.26894
30	1	1	0	0.26894
31	1	1	0	0.26894
32	1	1	0	0.26894
33	1	1	0	0.26894
34	1	1	0	0.26894
35	1	1	1	0.73106
36	1	1	1	0.73106
37	1	1	1	0.73106
38	1	1	1	0.73106
39	1	1	1	0.73106
40	1	1	1	0.73106

2 Next use the exemplary data as the input data set in PROC LOGISTIC, assigning PY as the WEIGHT variable. First fit the full model and save both the parameter estimates and fit statistics output tables as data sets. (You will later use a parameter estimate for the Wald chi-square power analysis and a fit statistic for the likelihood ratio test.)

```
* Run full model;
proc logistic data=Exemplary;
   ods output parameterestimates=fitFull fitstatistics=statsFull;
   weight PY;
   model Y(event='1') = X1 X2 X1*X2;
run;
```

Output 96.11.3 and Output 96.11.4 show these tables from the PROC LOGISTIC output.

Output 96.11.3 Parameter Estimates for Full Model

The LOGISTIC Procedure

Analysis of Maximum Likelihood Estimates					
Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept	1	1.69E-16	1.0000	0.0000	1.0000
X1	1	-1.4999	1.5300	0.9611	0.3269
X2	1	0.5000	1.3605	0.1351	0.7132
X1*X2	1	1.9999	2.0099	0.9901	0.3197

Output 96.11.4 Fit Statistics for Full Model

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	29.692	31.911
SC	31.381	38.666
-2 Log L	27.692	23.911

The numbers you need for the power analysis for the Wald chi-square test of X1*X2 are the degrees of freedom (1) and the Wald chi-square test statistic (0.9901). The number you need for the likelihood ratio test is the –2LogL value (23.911). The following DATA step saves these values to data sets:

```
* Fetch Wald chi-square statistic for test of X1*X2;
data fitFull;
   set fitFull;
   where Variable = "X1*X2";
   keep DF WaldChiSq;
run;
```

```
* Fetch -2LogL for full model;
data statsFull;
   set statsFull;
   where Criterion = "-2 Log L";
   Neg2LogLFull = InterceptAndCovariates;
   keep Neg2LogLFull;
run;
```

Next run the reduced model (dropping the interaction term, the one you want to test) and save the fit statistics table.

```
* Run reduced model;
proc logistic data=Exemplary;
   ods output fitstatistics=statsRed;
   weight PY;
   model Y = X1 X2;
run;
```

Output 96.11.5 shows the fit statistics table from the PROC LOGISTIC output.

Output 96.11.5 Fit Statistics for Reduced Model

The LOGISTIC Procedure

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	29.692	30.939
SC	31.381	36.005
-2 Log L	27.692	24.939

The number you need for the power analysis for the likelihood ratio test is the –2LogL value (24.939). The following DATA step saves these values to data sets:

```
* Fetch -2LogL for reduced model;
data statsRed;
   set statsRed;
   where Criterion = "-2 Log L";
   Neg2LogLRed = InterceptAndCovariates;
   keep Neg2LogLRed;
run;
```

Now you are ready to compute the primary noncentrality values. You need to divide both the Wald chi-square and –2LogL difference by the "effective sample size" of the exemplary data set—that is, the number of rows for each response value (20). The following statements compute the primary noncentrality and store all three numbers that you need for PROC POWER in macro variables:

```
* Compute primary noncentralities for Wald and LR tests;
data PrimNC;
   merge fitFull statsFull statsRed;
   WaldPrimNC = WaldChiSq / 20;
   LRPrimNC = -(Neg2LogLFull-Neg2LogLRed) / 20;
   keep DF WaldPrimNC LRPrimNC;
   call symput ("DF", DF);
   call symput ("WaldPrimNC", WaldPrimNC);
   call symput ("LRPrimNC", LRPrimNC);
run;


proc print data=PrimNC;
run;
```

The primary noncentrality values thus computed are shown in Output 96.11.6.

Output 96.11.6 Primary Noncentrality Values Derived from PROC LOGISTIC

Obs	DF	WaldPrimNC	LRPrimNC
1	1	0.049506	0.051396

3 Perform the power analysis by using the CUSTOM statement in PROC POWER as follows:

```
proc power;
   custom
      dist   = chisquare
      primnc = &WaldPrimNC &LRPrimNC
      testdf = &DF
      ntotal = 100
      power  = .;
run;
```

The computed powers for the two tests, shown in Output 96.11.7, match the results in Lyles, Lin, and Williamson (2007, section 3.2).

Output 96.11.7 Computed Power Values

The POWER Procedure
Custom Test

Fixed Scenario Elements	
Distribution	Chi-square
Method	Default
Test Degrees of Freedom	1
Total Sample Size	100
Alpha	0.05
Primary Noncentrality Multiplier	1
Critical Value Multiplier	1

Output 96.11.7 *continued*

Computed Power		
Index	Primary NC	Power
1	0.0495	0.605
2	0.0514	0.621

The following statements validate the PROC POWER results by using the QUANTILE and SDF functions in the DATA step:

```
data Power;
   set PrimNC;
   Crit = quantile("chisq", 1-0.05, DF);
   WaldPower = sdf("chisq", Crit, DF, 100 * WaldPrimNC);
   LRPower = sdf("chisq", Crit, DF, 100 * LRPrimNC);
   keep WaldPower LRPower;
run;

proc print data=Power;
run;
```

The computed powers for the two tests, shown in Output 96.11.8, match the results in Lyles, Lin, and Williamson (2007, section 3.2).

Output 96.11.8 Validation of Power Values

Obs	WaldPower	LRPower
1	0.60452	0.62063

References

Agresti, A. (1980). "Generalized Odds Ratios for Ordinal Data." *Biometrics* 36:59–67.

Agresti, A., and Coull, B. A. (1998). "Approximate Is Better Than 'Exact' for Interval Estimation of Binomial Proportions." *American Statistician* 52:119–126.

Anderson, T. W. (1984). *An Introduction to Multivariate Statistical Analysis*. 2nd ed. New York: John Wiley & Sons.

Beal, S. L. (1989). "Sample Size Determination for Confidence Intervals on the Population Means and on the Difference between Two Population Means." *Biometrics* 45:969–977.

Blackwelder, W. C. (1982). "'Proving the Null Hypothesis' in Clinical Trials." *Controlled Clinical Trials* 3:345–353.

Brown, L. D., Cai, T. T., and DasGupta, A. (2001). "Interval Estimation for a Binomial Proportion." *Statistical Science* 16:101–133.

Cantor, A. B. (1997). *Extending SAS Survival Analysis Techniques for Medical Research*. Cary, NC: SAS Institute Inc.

Castelloe, J. M. (2000). "Sample Size Computations and Power Analysis with the SAS System." In *Proceedings of the Twenty-Fifth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi25/25/st/25p265.pdf.

Castelloe, J. M., and O'Brien, R. G. (2001). "Power and Sample Size Determination for Linear Models." In *Proceedings of the Twenty-Sixth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi26/p240-26.pdf.

Chernick, M. R., and Liu, C. Y. (2002). "The Saw-Toothed Behavior of Power versus Sample Size and Software Solutions: Single Binomial Proportion Using Exact Methods." *American Statistician* 56:149–155.

Chow, S.-C., Shao, J., and Wang, H. (2003). *Sample Size Calculations in Clinical Research*. Boca Raton, FL: CRC Press.

Connor, R. J. (1987). "Sample Size for Testing Differences in Proportions for the Paired-Sample Design." *Biometrics* 43:207–211.

Diegert, C., and Diegert, K. V. (1981). "Note on Inversion of Casagrande-Pike-Smith Approximate Sample-Size Formula for Fisher-Irwin Test on 2×2 Tables." *Biometrics* 37:595.

Diletti, D., Hauschke, D., and Steinijans, V. W. (1991). "Sample Size Determination for Bioequivalence Assessment by Means of Confidence Intervals." *International Journal of Clinical Pharmacology, Therapy, and Toxicology* 29:1–8.

DiSantostefano, R. L., and Muller, K. E. (1995). "A Comparison of Power Approximations for Satterthwaite's Test." *Communications in Statistics—Simulation and Computation* 24:583–593.

Farrington, C. P., and Manning, G. (1990). "Test Statistics and Sample Size Formulae for Comparative Binomial Trials with Null Hypothesis of Non-zero Risk Difference or Non-unity Relative Risk." *Statistics in Medicine* 9:1447–1454.

Fisher, R. A. (1921). "On the 'Probable Error' of a Coefficient of Correlation Deduced from a Small Sample." *Metron* 1:3–32.

Fleiss, J. L., Tytun, A., and Ury, H. K. (1980). "A Simple Approximation for Calculating Sample Sizes for Comparing Independent Proportions." *Biometrics* 36:343–346.

Gatsonis, C., and Sampson, A. R. (1989). "Multiple Correlation: Exact Power and Sample Size Calculations." *Psychological Bulletin* 106:516–524.

Hocking, R. R. (1985). *The Analysis of Linear Models*. Monterey, CA: Brooks/Cole.

Hsieh, F. Y. (1989). "Sample Size Tables for Logistic Regression." *Statistics in Medicine* 8:795–802.

Hsieh, F. Y., and Lavori, P. W. (2000). "Sample-Size Calculations for the Cox Proportional Hazards Regression Model with Nonbinary Covariates." *Controlled Clinical Trials* 21:552–560.

Johnson, N. L., and Kotz, S. (1970). *Distributions in Statistics: Continuous Univariate Distributions*. New York: John Wiley & Sons.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1994). *Continuous Univariate Distributions*. 2nd ed. Vol. 1. New York: John Wiley & Sons.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1995). *Continuous Univariate Distributions.* 2nd ed. Vol. 2. New York: John Wiley & Sons.

Johnson, N. L., Kotz, S., and Kemp, A. W. (1992). *Univariate Discrete Distributions*. 2nd ed. New York: John Wiley & Sons.

Jones, R. M., and Miller, K. S. (1966). "On the Multivariate Lognormal Distribution." *Journal of Industrial Mathematics* 16:63–76.

Kolassa, J. E. (1995). "A Comparison of Size and Power Calculations for the Wilcoxon Statistic for Ordered Categorical Data." *Statistics in Medicine* 14:1577–1581.

Kotz, S., Balakrishnan, N., and Johnson, N. L. (2000). *Continuous Multivariate Distributions*. 2nd ed. New York: Wiley-Interscience.

Lachin, J. M. (1992). "Power and Sample Size Evaluation for the McNemar Test with Application to Matched Case-Control Studies." *Statistics in Medicine* 11:1239–1251.

Lakatos, E. (1988). "Sample Sizes Based on the Log-Rank Statistic in Complex Clinical Trials." *Biometrics* 44:229–241.

Lenth, R. V. (2001). "Some Practical Guidelines for Effective Sample Size Determination." *American Statistician* 55:187–193.

Lyles, R. H., Lin, H.-M., and Williamson, J. M. (2007). "A Practical Approach to Computing Power for Generalized Linear Models with Nominal, Count, or Ordinal Responses." *Statistics in Medicine* 26:1632–1648.

Maxwell, S. E. (2000). "Sample Size and Multiple Regression Analysis." *Psychological Methods* 5:434–458.

Miettinen, O. S. (1968). "The Matched Pairs Design in the Case of All-or-None Responses." *Biometrics* 24:339–352.

Moser, B. K., Stevens, G. R., and Watts, C. L. (1989). "The Two-Sample T Test versus Satterthwaite's Approximate F Test." *Communications in Statistics—Theory and Methods* 18:3963–3975.

Muller, K. E., and Benignus, V. A. (1992). "Increasing Scientific Power with Statistical Power." *Neurotoxicology and Teratology* 14:211–219.

O'Brien, R. G., and Castelloe, J. M. (2006). "Exploiting the Link between the Wilcoxon-Mann-Whitney Test and a Simple Odds Statistic." In *Proceedings of the Thirty-First Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi31/209-31.pdf.

O'Brien, R. G., and Castelloe, J. M. (2007). "Sample-Size Analysis for Traditional Hypothesis Testing: Concepts and Issues." In *Pharmaceutical Statistics Using SAS: A Practical Guide*, edited by A. Dmitrienko, C. Chuang-Stein, and R. D'Agostino, 237–271. Cary, NC: SAS Institute Inc.

O'Brien, R. G., and Muller, K. E. (1993). "Unified Power Analysis for t-Tests through Multivariate Hypotheses." In *Applied Analysis of Variance in Behavioral Science*, edited by L. K. Edwards, 297–344. New York: Marcel Dekker.

Owen, D. B. (1965). "A Special Case of a Bivariate Non-central t-Distribution." *Biometrika* 52:437–446.

Pagano, M., and Gauvreau, K. (1993). *Principles of Biostatistics*. Belmont, CA: Wadsworth.

Phillips, K. F. (1990). "Power of the Two One-Sided Tests Procedure in Bioequivalence." *Journal of Pharmacokinetics and Biopharmaceutics* 18:137–144.

Satterthwaite, F. E. (1946). "An Approximate Distribution of Estimates of Variance Components." *Biometrics Bulletin* 2:110–114.

Schork, M. A., and Williams, G. W. (1980). "Number of Observations Required for the Comparison of Two Correlated Proportions." *Communications in Statistics—Simulation and Computation* 9:349–357.

Schuirmann, D. J. (1987). "A Comparison of the Two One-Sided Tests Procedure and the Power Approach for Assessing the Equivalence of Average Bioavailability." *Journal of Pharmacokinetics and Biopharmaceutics* 15:657–680.

Self, S. G., Mauritsen, R. H., and Ohara, J. (1992). "Power Calculations for Likelihood Ratio Tests in Generalized Linear Models." *Biometrics* 48:31–39.

Senn, S. (1993). *Cross-Over Trials in Clinical Research*. New York: John Wiley & Sons.

Shieh, G. (2000). "A Comparison of Two Approaches for Power and Sample Size Calculations in Logistic Regression Models." *Communications in Statistics—Simulation and Computation* 29:763–791.

Shieh, G., and O'Brien, R. G. (1998). "A Simpler Method to Compute Power for Likelihood Ratio Tests in Generalized Linear Models." In *Proceedings of the Annual Joint Statistical Meetings of the American Statistical Association*. Alexandria, VA: ASA.

Stuart, A., and Ord, J. K. (1994). *Distribution Theory.* Vol. 1 of Kendall's Advanced Theory of Statistics. 6th ed. Baltimore: Edward Arnold.

Walters, D. E. (1979). "In Defence of the Arc Sine Approximation." *The Statistician* 28:219–232.

Wellek, S. (2003). *Testing Statistical Hypotheses of Equivalence*. Boca Raton, FL: Chapman & Hall/CRC.

Chapter 97
The Power and Sample Size Application

The functionality of the Power and Sample Size application has been replaced by a collection of steps in SAS Studio whose names end in “Statistical Power” and are documented in the “Generating Statistics” section of *SAS Studio: Working with Flows*.

Chapter 98
The PRINCOMP Procedure

Contents

Overview: PRINCOMP Procedure **8156**
Getting Started: PRINCOMP Procedure **8157**
Syntax: PRINCOMP Procedure **8162**
- PROC PRINCOMP Statement 8163
- BY Statement 8169
- FREQ Statement 8169
- ID Statement 8170
- PARTIAL Statement 8170
- VAR Statement 8170
- WEIGHT Statement 8170

Details: PRINCOMP Procedure **8171**
- Missing Values 8171
- Output Data Sets 8171
- Computational Resources 8173
- Displayed Output 8174
- ODS Table Names 8175
- ODS Graphics 8175

Examples: PRINCOMP Procedure **8176**
- Example 98.1: Analyzing Mean Temperatures of US Cities 8176
- Example 98.2: Analyzing Rankings of US College Basketball Teams 8179
- Example 98.3: Analyzing Job Ratings of Police Officers 8186

References **8197**

Overview: PRINCOMP Procedure

The PRINCOMP procedure performs principal component analysis. As input, you can use raw data, a correlation matrix, a covariance matrix, or a sum-of-squares-and-crossproducts (SSCP) matrix. You can create output data sets that contain eigenvalues, eigenvectors, and standardized or unstandardized principal component scores.

Principal component analysis is a multivariate technique for examining relationships among several quantitative variables. The choice between using factor analysis and using principal component analysis depends in part on your research objectives. You should use the PRINCOMP procedure if you are interested in summarizing data and detecting linear relationships. You can use principal component analysis to reduce the number of variables in regression, clustering, and so on. For a detailed comparison of the PRINCOMP and FACTOR procedures, see Chapter 10, "Introduction to Multivariate Procedures."

You can use ODS Graphics to display the scree plot, component pattern plot, component pattern profile plot, matrix plot of component scores, and component score plots. These plots are especially valuable tools in exploratory data analysis.

Principal component analysis was originated by Pearson (1901) and later developed by Hotelling (1933). The application of principal components is discussed by Rao (1964); Cooley and Lohnes (1971); Gnanadesikan (1977). Excellent statistical treatments of principal components are found in Kshirsagar (1972); Morrison (1976); Mardia, Kent, and Bibby (1979).

If you have a data set that contains p numeric variables, you can compute p principal components. Each principal component is a linear combination of the original variables, with coefficients equal to the eigenvectors of the correlation or covariance matrix. The eigenvectors are usually taken with unit length. The principal components are sorted by descending order of the eigenvalues, which are equal to the variances of the components.

Principal components have a variety of useful properties (Rao 1964; Kshirsagar 1972):

- The eigenvectors are orthogonal, so the principal components represent jointly perpendicular directions through the space of the original variables.
- The principal component scores are jointly uncorrelated. Note that this property is quite distinct from the previous one.
- The first principal component has the largest variance of any unit-length linear combination of the observed variables. The jth principal component has the largest variance of any unit-length linear combination orthogonal to the first $j-1$ principal components. The last principal component has the smallest variance of any linear combination of the original variables.
- The scores on the first j principal components have the highest possible generalized variance of any set of unit-length linear combinations of the original variables.
- The first j principal components provide a least squares solution to the model

  $$\mathbf{Y} = \mathbf{XB} + \mathbf{E}$$

  where $\mathbf{Y}$ is an $n \times p$ matrix of the centered observed variables; $\mathbf{X}$ is the $n \times j$ matrix of scores on the first j principal components; $\mathbf{B}$ is the $j \times p$ matrix of eigenvectors; $\mathbf{E}$ is an $n \times p$ matrix of residuals;

and you want to minimize trace($\mathbf{E}'\mathbf{E}$), the sum of all the squared elements in $\mathbf{E}$. In other words, the first j principal components are the best linear predictors of the original variables among all possible sets of j variables, although any nonsingular linear transformation of the first j principal components would provide equally good prediction. The same result is obtained if you want to minimize the determinant or the Euclidean (Schur, Frobenius) norm of $\mathbf{E}'\mathbf{E}$ rather than the trace.

- In geometric terms, the j-dimensional linear subspace that is spanned by the first j principal components provides the best possible fit to the data points as measured by the sum of squared perpendicular distances from each data point to the subspace. This contrasts with the geometric interpretation of least squares regression, which minimizes the sum of squared vertical distances. For example, suppose you have two variables. Then, the first principal component minimizes the sum of squared perpendicular distances from the points to the first principal axis. This contrasts with least squares, which would minimize the sum of squared vertical distances from the points to the fitted line.

Principal component analysis can also be used for exploring polynomial relationships and for multivariate outlier detection (Gnanadesikan 1977), and it is related to factor analysis, correspondence analysis, allometry, and biased regression techniques (Mardia, Kent, and Bibby 1979).

Getting Started: PRINCOMP Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/princgs.sas.)

The following data provide crime rates per 100,000 people in seven categories for each of the 50 US states in 1977. Because there are seven numeric variables, it is impossible to plot all the variables simultaneously. You can use principal components to summarize the data in two or three dimensions, and they help you visualize the data. The following statements produce Figure 98.1 through Figure 98.5:

```
title 'Crime Rates per 100,000 Population by State';

data Crime;
   input State $1-15 Murder Rape Robbery Assault
         Burglary Larceny Auto_Theft;
   datalines;
Alabama        14.2 25.2  96.8 278.3 1135.5 1881.9 280.7
Alaska         10.8 51.6  96.8 284.0 1331.7 3369.8 753.3
Arizona         9.5 34.2 138.2 312.3 2346.1 4467.4 439.5
Arkansas        8.8 27.6  83.2 203.4  972.6 1862.1 183.4
California     11.5 49.4 287.0 358.0 2139.4 3499.8 663.5
Colorado        6.3 42.0 170.7 292.9 1935.2 3903.2 477.1
Connecticut     4.2 16.8 129.5 131.8 1346.0 2620.7 593.2
Delaware        6.0 24.9 157.0 194.2 1682.6 3678.4 467.0
Florida        10.2 39.6 187.9 449.1 1859.9 3840.5 351.4
Georgia        11.7 31.1 140.5 256.5 1351.1 2170.2 297.9
Hawaii          7.2 25.5 128.0  64.1 1911.5 3920.4 489.4
Idaho           5.5 19.4  39.6 172.5 1050.8 2599.6 237.6
Illinois        9.9 21.8 211.3 209.0 1085.0 2828.5 528.6
Indiana         7.4 26.5 123.2 153.5 1086.2 2498.7 377.4
Iowa            2.3 10.6  41.2  89.8  812.5 2685.1 219.9
```

```
Kansas          6.6 22.0 100.7 180.5 1270.4 2739.3 244.3
Kentucky       10.1 19.1  81.1 123.3  872.2 1662.1 245.4
Louisiana      15.5 30.9 142.9 335.5 1165.5 2469.9 337.7
Maine           2.4 13.5  38.7 170.0 1253.1 2350.7 246.9
Maryland        8.0 34.8 292.1 358.9 1400.0 3177.7 428.5
Massachusetts   3.1 20.8 169.1 231.6 1532.2 2311.3 1140.1
Michigan        9.3 38.9 261.9 274.6 1522.7 3159.0 545.5
Minnesota       2.7 19.5  85.9  85.8 1134.7 2559.3 343.1
Mississippi    14.3 19.6  65.7 189.1  915.6 1239.9 144.4
Missouri        9.6 28.3 189.0 233.5 1318.3 2424.2 378.4
Montana         5.4 16.7  39.2 156.8  804.9 2773.2 309.2
Nebraska        3.9 18.1  64.7 112.7  760.0 2316.1 249.1
Nevada         15.8 49.1 323.1 355.0 2453.1 4212.6 559.2
New Hampshire   3.2 10.7  23.2  76.0 1041.7 2343.9 293.4
New Jersey      5.6 21.0 180.4 185.1 1435.8 2774.5 511.5
New Mexico      8.8 39.1 109.6 343.4 1418.7 3008.6 259.5
New York       10.7 29.4 472.6 319.1 1728.0 2782.0 745.8
North Carolina 10.6 17.0  61.3 318.3 1154.1 2037.8 192.1
North Dakota    0.9  9.0  13.3  43.8  446.1 1843.0 144.7
Ohio            7.8 27.3 190.5 181.1 1216.0 2696.8 400.4
Oklahoma        8.6 29.2  73.8 205.0 1288.2 2228.1 326.8
Oregon          4.9 39.9 124.1 286.9 1636.4 3506.1 388.9
Pennsylvania    5.6 19.0 130.3 128.0  877.5 1624.1 333.2
Rhode Island    3.6 10.5  86.5 201.0 1489.5 2844.1 791.4
South Carolina 11.9 33.0 105.9 485.3 1613.6 2342.4 245.1
South Dakota    2.0 13.5  17.9 155.7  570.5 1704.4 147.5
Tennessee      10.1 29.7 145.8 203.9 1259.7 1776.5 314.0
Texas          13.3 33.8 152.4 208.2 1603.1 2988.7 397.6
Utah            3.5 20.3  68.8 147.3 1171.6 3004.6 334.5
Vermont         1.4 15.9  30.8 101.2 1348.2 2201.0 265.2
Virginia        9.0 23.3  92.1 165.7  986.2 2521.2 226.7
Washington      4.3 39.6 106.2 224.8 1605.6 3386.9 360.3
West Virginia   6.0 13.2  42.2  90.9  597.4 1341.7 163.3
Wisconsin       2.8 12.9  52.2  63.7  846.9 2614.2 220.7
Wyoming         5.4 21.9  39.7 173.9  811.6 2772.2 282.0
;

ods graphics on;

proc princomp out=Crime_Components plots= score(ellipse ncomp=3);
   id State;
run;
```

Figure 98.1 displays the PROC PRINCOMP output, beginning with simple statistics and followed by the correlation matrix. By default, the PROC PRINCOMP statement requests principal components that are computed from the correlation matrix, so the total variance is equal to the number of variables, 7.

Figure 98.1 Number of Observations and Simple Statistics from the PRINCOMP Procedure

Crime Rates per 100,000 Population by State

The PRINCOMP Procedure

Observations	50
Variables	7

Simple Statistics

	Murder	Rape	Robbery	Assault	Burglary	Larceny	Auto_Theft
Mean	7.444000000	25.73400000	124.0920000	211.3000000	1291.904000	2671.288000	377.5260000
StD	3.866768941	10.75962995	88.3485672	100.2530492	432.455711	725.908707	193.3944175

Correlation Matrix

	Murder	Rape	Robbery	Assault	Burglary	Larceny	Auto_Theft
Murder	1.0000	0.6012	0.4837	0.6486	0.3858	0.1019	0.0688
Rape	0.6012	1.0000	0.5919	0.7403	0.7121	0.6140	0.3489
Robbery	0.4837	0.5919	1.0000	0.5571	0.6372	0.4467	0.5907
Assault	0.6486	0.7403	0.5571	1.0000	0.6229	0.4044	0.2758
Burglary	0.3858	0.7121	0.6372	0.6229	1.0000	0.7921	0.5580
Larceny	0.1019	0.6140	0.4467	0.4044	0.7921	1.0000	0.4442
Auto_Theft	0.0688	0.3489	0.5907	0.2758	0.5580	0.4442	1.0000

Figure 98.2 displays the eigenvalues. The first principal component accounts for about 58.8% of the total variance, the second principal component accounts for about 17.7%, and the third principal component accounts for about 10.4%. Note that the eigenvalues sum to the total variance.

The eigenvalues indicate that two or three components provide a good summary of the data: two components account for 76% of the total variance, and three components account for 87%. Subsequent components account for less than 5% each.

Figure 98.2 Results of Principal Component Analysis: PROC PRINCOMP

Eigenvalues of the Correlation Matrix

	Eigenvalue	Difference	Proportion	Cumulative
1	4.11495951	2.87623768	0.5879	0.5879
2	1.23872183	0.51290521	0.1770	0.7648
3	0.72581663	0.40938458	0.1037	0.8685
4	0.31643205	0.05845759	0.0452	0.9137
5	0.25797446	0.03593499	0.0369	0.9506
6	0.22203947	0.09798342	0.0317	0.9823
7	0.12405606		0.0177	1.0000

Figure 98.3 displays the eigenvectors. From the eigenvectors matrix, you can represent the first principal

component, Prin1, as a linear combination of the original variables:

$$\begin{aligned}\text{Prin1} = \quad & 0.300279 \times \text{Murder} \\ + \quad & 0.431759 \times \text{Rape} \\ + \quad & 0.396875 \times \text{Robbery} \\ & . \\ & . \\ & . \\ + \quad & 0.295177 \times \text{Auto_Theft}\end{aligned}$$

Similarly, the second principal component, Prin2, is

$$\begin{aligned}\text{Prin2} = \quad - \quad & 0.629174 \times \text{Murder} \\ - \quad & 0.169435 \times \text{Rape} \\ + \quad & 0.042247 \times \text{Robbery} \\ & . \\ & . \\ & . \\ - \quad & 0.502421 \times \text{Auto_Theft}\end{aligned}$$

where the variables are standardized.

Figure 98.3 Results of Principal Component Analysis: PROC PRINCOMP

Eigenvectors							
	Prin1	**Prin2**	**Prin3**	**Prin4**	**Prin5**	**Prin6**	**Prin7**
Murder	0.300279	-.629174	0.178245	-.232114	0.538123	0.259117	0.267593
Rape	0.431759	-.169435	-.244198	0.062216	0.188471	-.773271	-.296485
Robbery	0.396875	0.042247	0.495861	-.557989	-.519977	-.114385	-.003903
Assault	0.396652	-.343528	-.069510	0.629804	-.506651	0.172363	0.191745
Burglary	0.440157	0.203341	-.209895	-.057555	0.101033	0.535987	-.648117
Larceny	0.357360	0.402319	-.539231	-.234890	0.030099	0.039406	0.601690
Auto_Theft	0.295177	0.502421	0.568384	0.419238	0.369753	-.057298	0.147046

The first component is a measure of the overall crime rate because the first eigenvector shows approximately equal loadings on all variables. The second eigenvector has high positive loadings on the variables Auto_Theft and Larceny and high negative loadings on the variables Murder and Assault. There is also a small positive loading on the variable Burglary and a small negative loading on the variable Rape. This component seems to measure the preponderance of property crime compared to violent crime. The interpretation of the third component is not obvious.

The ODS GRAPHICS statement enables the creation of graphs. For more information, see Chapter 24, "Statistical Graphics Using ODS." The option PLOTS=SCORE(ELLIPSE NCOMP=3) in the PROC PRINCOMP statement requests the pairwise component score plots for the first three components, with a 95% prediction ellipse overlaid on each scatter plot. Figure 98.4 shows the plot of the first two components. You can identify

regional trends in the plot of the first two components. Nevada and California are at the extreme right, with high overall crime rates but an average ratio of property crime to violent crime. North Dakota and South Dakota are at the extreme left, with low overall crime rates. Southeastern states tend to be at the bottom of the plot, with a higher-than-average ratio of violent crime to property crime. New England states tend to be in the upper part of the plot, with a higher-than-average ratio of property crime to violent crime. Assuming that the first two components are from a bivariate normal distribution, the ellipse identifies Nevada as a possible outlier.

Figure 98.5 shows the plot of the first and third components. Assuming that the first and third components are from a bivariate normal distribution, the ellipse identifies Nevada, Massachusetts, and New York as possible outliers.

The most striking feature of the plot of the first and third principal components is that Massachusetts and New York are outliers on the third component.

Figure 98.4 Plot of the First Two Component Scores

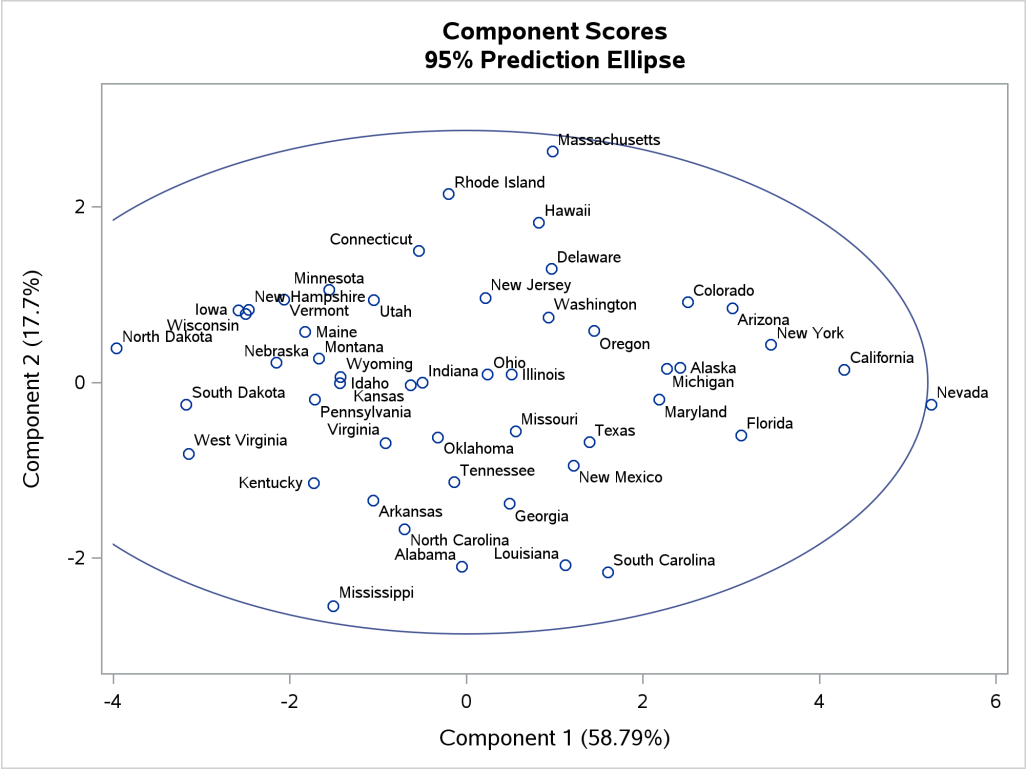

Figure 98.5 Plot of the First and Third Component Scores

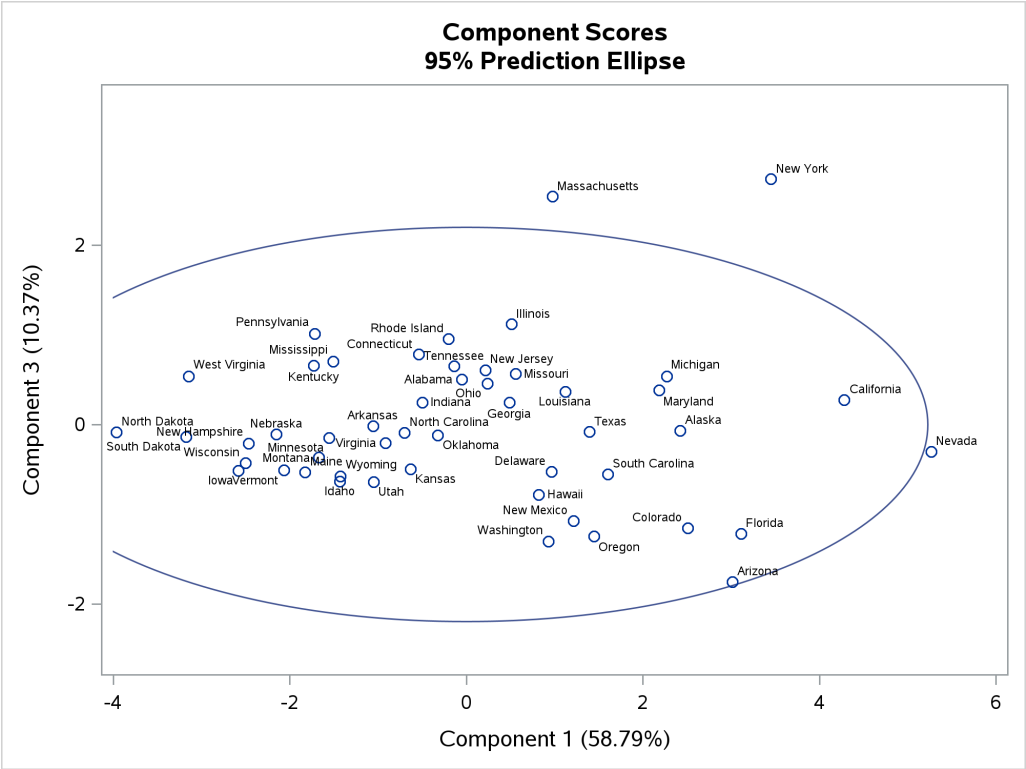

Syntax: PRINCOMP Procedure

The following statements are available in the PRINCOMP procedure:

PROC PRINCOMP < *options* > **;**
  BY *variables* **;**
  FREQ *variable* **;**
  ID *variables* **;**
  PARTIAL *variables* **;**
  VAR *variables* **;**
  WEIGHT *variable* **;**

Usually only the VAR statement is used in addition to the PROC PRINCOMP statement. The rest of this section provides detailed syntax information for each of the preceding statements, beginning with the PROC PRINCOMP statement. The remaining statements are described in alphabetical order.

PROC PRINCOMP Statement

PROC PRINCOMP < *options* > **;**

The PROC PRINCOMP statement invokes the PRINCOMP procedure. Optionally, it also identifies input and output data sets, specifies the analyses that are performed, and controls displayed output. Table 98.1 summarizes the options available in the PROC PRINCOMP statement.

Table 98.1 Summary of PROC PRINCOMP Statement Options

Option	Description
Specify Data Sets	
DATA=	Specifies the name of the input data set
OUT=	Specifies the name of the output data set
OUTSTAT=	Specifies the name of the output data set that contains various statistics
Specify Details of Analysis	
COV	Computes the principal components from the covariance matrix
N=	Specifies the number of principal components to be computed
NOINT	Omits the intercept from the model
PREFIX=	Specifies a prefix for naming the principal components
PARPREFIX=	Specifies a prefix for naming the residual variables
SINGULAR=	Specifies the singularity criterion
STD	Standardizes the principal component scores
VARDEF=	Specifies the divisor used in calculating variances and standard deviations
Suppress the Display of Output	
NOPRINT	Suppresses the display of all output
Specify ODS Graphics Details	
PLOTS=	Specifies options that control the details of the plots

The following list provides details about these *options*.

COVARIANCE

COV

computes the principal components from the covariance matrix. If you omit the COV option, the correlation matrix is analyzed. The COV option causes variables that have large variances to be more strongly associated with components that have large eigenvalues, it and causes variables that have small variances to be more strongly associated with components that have small eigenvalues. You should not specify the COV option unless the units in which the variables are measured are comparable or the variables are standardized in some way.

DATA=*SAS-data-set*

specifies the SAS data set to be analyzed. The data set can be an ordinary SAS data set or a TYPE=ACE, TYPE=CORR, TYPE=COV, TYPE=FACTOR, TYPE=SSCP, TYPE=UCORR, or TYPE=UCOV data set (see Appendix A, "Special SAS Data Sets"). Also, the PRINCOMP procedure can read the _TYPE_='COVB' matrix from a TYPE=EST data set. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

N=*number*

specifies the number of principal components to be computed. The default is the number of variables. The value of the N= option must be an integer greater than or equal to 0.

NOINT

omits the intercept from the model. In other words, the NOINT option requests that the covariance or correlation matrix not be corrected for the mean. When you specify the NOINT option, the covariance matrix and, hence, the standard deviations are not corrected for the mean.

If you use a TYPE=SSCP data set as input to the PRINCOMP procedure and list the variable Intercept in the VAR statement, the procedure acts as if you had also specified the NOINT option. If you use the NOINT option and also create an OUTSTAT= data set, the data set is TYPE=UCORR or TYPE=UCOV rather than TYPE=CORR or TYPE=COV.

NOPRINT

suppresses the display of all output. This option temporarily disables the Output Delivery System (ODS). For more information about ODS, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*

creates an output SAS data set to contain all the original data in addition to the principal component scores.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For information about OUT= data sets, see the section "Output Data Sets" on page 8171.

OUTSTAT=*SAS-data-set*

creates an output SAS data set to contain means, standard deviations, number of observations, correlations or covariances, eigenvalues, and eigenvectors. If you specify the COV option, the data set is TYPE=COV or TYPE=UCOV, depending on the NOINT option, and it contains covariances; otherwise, the data set is TYPE=CORR or TYPE=UCORR, depending on the NOINT option, and it contains correlations. If you specify the PARTIAL statement, the OUTSTAT= data set also contains R squares.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For more information about OUTSTAT= data sets, see the section "Output Data Sets" on page 8171.

PLOTS < (*global-plot-options*) > < = *plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < = (*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the plots that are produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=none
plots=(scatter pattern)
plots(unpack)=scree
plots(ncomp=3 flip)=(pattern(circles=0.5 1.0) score)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc princomp plots=all;
   var x1--x10;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, PROC PRINCOMP produces the scree plot by default.

You can specify the following *global-plot-options*:

FLIP

flips or interchanges the X-axis and Y-axis dimensions of the component score plots and the component pattern plots. For example, if you have three components, the default plots ($y * x$) are Component 2 * Component 1, Component 3 * Component 1, and Component 3 * Component 2. When you specify PLOTS(FLIP), the plots are Component 1 * Component 2, Component 1 * Component 3, and Component 2 * Component 3.

NCOMP=*n*

specifies the number of components $n(\geq 2)$ to be plotted for the component pattern plots and the component score plots. If you specify the NCOMP= option again in an individual plot, such as PLOTS=SCORE(NCOMP= m), the value m determines the number of components to be plotted in the component score plots. Be aware that the number of plots $(n \times (n-1)/2)$ that are produced grows quadratically when n increases. The default is 5 or the total number of components $m(\geq 2)$, whichever is smaller. If $n > m$, NCOMP=m is used.

ONLY

suppresses the default plots. Only plots that you specifically request are displayed.

UNPACKPANEL

UNPACK

suppresses paneling in the scree plot. By default, multiple plots can appear in an output panel. Specify UNPACKPANEL to get each plot to appear in a separate panel. You can specify PLOTS(UNPACKPANEL) to unpack the default plots. You can also specify UNPACKPANEL as a suboption with the SCREE option (such as PLOTS=SCREE(UNPACKPANEL)).

You can specify the following *plot-requests*:

ALL
produces all appropriate plots. You can specify other options along with ALL; for example, to request all plots and unpack only the scree plot, specify PLOTS=(ALL SCREE(UNPACKPANEL)).

EIGEN | EIGENVALUE | SCREE < (UNPACKPANEL) >
produces the scree plot of eigenvalues and proportion variance explained. By default, both plots appear in the same panel. Specify PLOTS= SCREE(UNPACKPANEL) to get each plot to appear in a separate panel.

MATRIX
produces the matrix plot of principal component scores.

NONE
suppresses the display of all graphics output.

PATTERN < (*pattern-options*) >
produces the pairwise component pattern plots. Each variable is plotted as an observation whose coordinates are correlations between the variable and the two corresponding components in the plot. Use the NCOMP= option (for instance, PLOTS=PATTERN(NCOMP=3)) as described in the following list to control the number of plots to display.

You can specify the following *pattern-options*:

CIRCLES < = *number-list* >
plots the variance percentage circles. For each number c ($0 < c \leq 1$) that is specified, a ($c \times 100\%$) variance circle is displayed. For each number c ($c > 1$) that is specified, a $c\%$ variance circle is displayed. You can specify either CIRCLES=0.05 1 or CIRCLES=5 100 to display 5% and 100% variance circles. PLOTS=PATTERN(CIRCLES) and PLOTS=PATTERN(VECTOR) both display a unit circle (100% variance). By default, no circle is displayed when you specify PLOTS=PATTERN.

FLIP
flips or interchanges the X-axis and Y-axis dimensions of the component pattern plots. Specify PLOTS=PATTERN(FLIP) to flip the X-axis and Y-axis dimensions.

NCOMP=*n*
specifies the number of components $n(\geq 2)$ to be plotted. The default is 5 or the total number of components $m(\geq 2)$, whichever is smaller. If $n > m$, NCOMP=m is used. Be aware that the number of plots $(n \times (n-1)/2)$ that are produced grows quadratically when n increases.

VECTOR
plots the pattern in a vector form.

PATTERNPROFILE | PROFILE
produces the pattern profile plot. Each component has its own profile. The Y-axis value represents the correlation between the variable (corresponding to the X-axis value) and the profiled principal component.

SCORE < (*score-options*) >

produces the pairwise component score plots. Use the NCOMP= option (for example, PLOTS=SCORE(NCOMP=3)) as described in the following list to control the number of plots to display.

You can specify the following *score-options*.

ALPHA=*number list*

specifies a list of numbers for the prediction ellipses to be displayed in the score plots. Each value (α) in the list must be greater than 0. If α is greater than or equal to 1, it is interpreted as a percentage and divided by 100; ALPHA=0.05 and ALPHA=5 are equivalent.

ELLIPSE

requests prediction ellipses for the principal component scores of a new observation to be created in the principal component score plots. For information about the computation of a prediction ellipse, see the section "Confidence and Prediction Ellipses" in "The CORR Procedure" (*Base SAS Procedures Guide: Statistical Procedures*).

FLIP

flips or interchanges the X-axis and Y-axis dimensions of the component score plots. Specify PLOTS=SCORE(FLIP) to flip the X-axis and Y-axis dimensions.

NCOMP=*n*

specifies the number of components $n(\geq 2)$ to be plotted. The default is 5 or the total number of components $m(\geq 2)$, whichever is smaller. If $n > m$, NCOMP=*m* is used. Be aware that the number of plots $(n \times (n-1)/2)$ that are produced grows quadratically when *n* increases.

PAINT < =*position* >

creates plots of component *i* versus component *j*, painted by component *k*. When you have at least three components, the PLOTS=SCORE option is specified, and the PAINT option is not specified, a painted score plot for component 3 versus component 2, painted by component 1, is produced. Use the PAINT option when you want to create painted score plots that involve other triples of components.

PLOTS=SCORE(PAINT), PLOTS=SCORE(PAINT=F), and PLOTS=SCORE(PAINT=FIRST) are all equivalent and create painted plots of $i \times j$, painted by k for triples (i, j, k), where $k < j < i$.

PLOTS=SCORE(PAINT=L) and PLOTS=SCORE(PAINT=LAST) are equivalent and create painted plots of $i \times j$, painted by k for triples (i, j, k), where $j < i < k$.

PLOTS=SCORE(PAINT=M) and PLOTS=SCORE(PAINT=MIDDLE) are equivalent and create painted plots of $i \times j$, painted by k for triples (i, j, k), where $j < k < i$.

PREFIX=*name*

specifies a prefix for naming the principal components. By default, the names are Prin1, Prin2, ..., Prin*n*. If you specify PREFIX=Abc, the components are named Abc1, Abc2, Abc3, and so on. The number of characters in the prefix plus the number of digits required to designate the variables should not exceed the current name length that is defined by the VALIDVARNAME= system option.

PARPREFIX=*name*

PPREFIX=*name*

RPREFIX=*name*

specifies a prefix for naming the residual variables in the OUT= data set and the OUTSTAT= data set. By default, the prefix is R_. The number of characters in the prefix plus the maximum length of the variable names should not exceed the current name length that is defined by the VALIDVARNAME= system option.

SINGULAR=*p*

SING=*p*

specifies the singularity criterion, where $0 < p < 1$. If a variable in a PARTIAL statement has an R square as large as $1 - p$ when predicted from the variables listed before it in the statement, the variable is assigned a standardized coefficient of 0. By default, SINGULAR=1E–8.

STANDARD

STD

standardizes the principal component scores in the OUT= data set and in plots to unit variance. If you omit the STANDARD option, the scores have variance equal to the corresponding eigenvalue. Note that the STANDARD option has no effect on the eigenvalues themselves.

VARDEF=DF | N | WDF | WEIGHT | WGT

specifies the divisor to be used in calculating variances and standard deviations. By default, VARDEF=DF. The following table displays the values and associated divisors:

Value	Divisor	Formula	
DF	Error degrees of freedom	$n - i$	(before partialing)
		$n - p - i$	(after partialing)
N	Number of observations	n	
WEIGHT | WGT	Sum of weights	$\sum_{j=1}^{n} w_j$	
WDF	Sum of weights minus one	$\left(\sum_{j=1}^{n} w_j\right) - i$	(before partialing)
		$\left(\sum_{j=1}^{n} w_j\right) - p - i$	(after partialing)

In the formulas for VARDEF=DF and VARDEF=WDF, p is the number of degrees of freedom of the variables in the PARTIAL statement, and i is 0 if the NOINT option is specified and 1 otherwise.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC PRINCOMP to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the PRINCOMP procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

The FREQ statement specifies a variable that provides frequencies for each observation in the DATA= data set. Specifically, if n is the value of the FREQ variable for a given observation, then that observation is used n times.

The analysis that you produce by using a FREQ statement reflects the expanded number of observations. The total number of observations is considered to be equal to the sum of the FREQ variable. You could produce the same analysis (without the FREQ statement) by first creating a new data set that contains the expanded number of observations. For example, if the value of the FREQ variable is 5 for the first observation, the first five observations in the new data set are identical. Each observation in the old data set would be replicated n_j times in the new data set, where n_j is the value of the FREQ variable for that observation.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

ID Statement

ID *variables* **;**

The ID statement labels observations by using values from the first ID variable in the principal component score plot. If one or more ID variables are specified, their values are displayed in tooltips of the component score plot and the matrix plot of component scores.

PARTIAL Statement

PARTIAL *variables* **;**

If you want to analyze a partial correlation or covariance matrix, specify the names of the numeric variables to be partialed out in the PARTIAL statement. The PRINCOMP procedure computes the principal components of the residuals from the prediction of the VAR variables by the PARTIAL variables. If you request an OUT= or OUTSTAT= data set, the residual variables are named by prefixing the characters R_ by default or the string specified in the PARPREFIX= option to the VAR variables.

VAR Statement

VAR *variables* **;**

The VAR statement lists the numeric variables to be analyzed. If you omit the VAR statement, all numeric variables not specified in other statements are analyzed. However, if the DATA= data set is TYPE=SSCP, the default set of variables used as VAR variables does not include Intercept so that the correlation or covariance matrix is constructed correctly. If you want to analyze Intercept as a separate variable, you should specify it in the VAR statement.

WEIGHT Statement

WEIGHT *variable* **;**

To use relative weights for each observation in the input data set, place the weights in a variable in the data set and specify the name in a WEIGHT statement. This is often done when the variance associated with each observation is different and the values of the weight variable are proportional to the reciprocals of the variances.

The observation is used in the analysis only if the value of the WEIGHT statement variable is nonmissing and is greater than 0.

Details: PRINCOMP Procedure

Missing Values

Observations that have missing values for any variable in the VAR, PARTIAL, FREQ, or WEIGHT statement are omitted from the analysis and are given missing values for principal component scores in the OUT= data set. If a correlation, covariance, or SSCP matrix is read, it can contain missing values as long as every pair of variables has at least one nonmissing entry.

Output Data Sets

OUT= Data Set

The OUT= data set contains all the variables in the original data set plus new variables that contain the principal component scores. The N= option determines the number of new variables. The names of the new variables are formed by concatenating the value given by the PREFIX= option (or Prin if PREFIX= is omitted) to the numbers 1, 2, 3, and so on. The new variables have mean 0 and variance equal to the corresponding eigenvalue, unless you specify the STANDARD option to standardize the scores to unit variance. Also, if you specify the COV option, PROC PRINCOMP computes the principal component scores from the corrected or uncorrected (if the NOINT option is specified) variables rather than from the standardized variables.

If you use a PARTIAL statement, the OUT= data set also contains the residuals from predicting the VAR variables from the PARTIAL variables.

You cannot create an OUT= data set if the DATA= data set is TYPE=ACE, TYPE=CORR, TYPE=COV, TYPE=EST, TYPE=FACTOR, TYPE=SSCP, TYPE=UCORR, or TYPE=UCOV.

OUTSTAT= Data Set

The OUTSTAT= data set is similar to the TYPE=CORR data set that the CORR procedure produces. The following table relates the TYPE= value for the OUTSTAT= data set to the options that are specified in the PROC PRINCOMP statement:

Options	TYPE=
(Default)	CORR
COV	COV
NOINT	UCORR
COV NOINT	UCOV

Note that the default (neither the COV nor NOINT option) produces a TYPE=CORR data set.

The new data set contains the following variables:

- the BY variables, if any

- two new variables, _TYPE_ and _NAME_, both character variables
- the variables that are analyzed (that is, those in the VAR statement); or, if there is no VAR statement, all numeric variables not listed in any other statement; or, if there is a PARTIAL statement, the residual variables as described in the section "OUT= Data Set" on page 8171

Each observation in the new data set contains some type of statistic, as indicated by the _TYPE_ variable. The values of the _TYPE_ variable are as follows:

TYPE	Contents
MEAN	mean of each variable. If you specify the PARTIAL statement, this observation is omitted.
STD	standard deviations. If you specify the COV option, this observation is omitted, so the SCORE procedure does not standardize the variables before computing scores. If you use the PARTIAL statement, the standard deviation of a variable is computed as its root mean squared error as predicted from the PARTIAL variables.
USTD	uncorrected standard deviations. When you specify the NOINT option in the PROC PRINCOMP statement, the OUTSTAT= data set contains standard deviations not corrected for the mean. However, if you also specify the COV option in the PROC PRINCOMP statement, this observation is omitted.
N	number of observations on which the analysis is based. This value is the same for each variable. If you specify the PARTIAL statement and the value of the VARDEF= option is DF or unspecified, then the number of observations is decremented by the degrees of freedom of the PARTIAL variables.
SUMWGT	the sum of the weights of the observations. This value is the same of each variable. If you specify the PARTIAL statement and VARDEF=WDF, then the sum of the weights is decremented by the degrees of freedom of the PARTIAL variables. This observation is output only if the value is different from that in the observation for which _TYPE_='N'.
CORR	correlations between each variable and the variable specified by the _NAME_ variable. The number of observations for which _TYPE_='CORR' is equal to the number of variables being analyzed. If you specify the COV option, no _TYPE_='CORR' observations are produced. If you use the PARTIAL statement, the partial correlations, not the raw correlations, are output.
UCORR	uncorrected correlation matrix. When you specify the NOINT option without the COV option in the PROC PRINCOMP statement, the OUTSTAT= data set contains a matrix of correlations not corrected for the means. However, if you also specify the COV option in the PROC PRINCOMP statement, this observation is omitted.
COV	covariances between each variable and the variable specified by the _NAME_ variable. _TYPE_='COV' observations are produced only if you specify the COV option. If you use the PARTIAL statement, the partial covariances, not the raw covariances, are output.
UCOV	uncorrected covariance matrix. When you specify the NOINT and COV options in the PROC PRINCOMP statement, the OUTSTAT= data set contains a matrix of covariances not corrected for the means.
EIGENVAL	eigenvalues. If the N= option requests fewer principal components than the maximum number, only the specified number of eigenvalues is produced, with missing values filling out the observation.

SCORE eigenvectors. The _NAME_ variable contains the name of the corresponding principal component as constructed from the PREFIX= option. The number of observations for which _TYPE_='SCORE' equals the number of principal components computed. The eigenvectors have unit length unless you specify the STD option, in which case the unit-length eigenvectors are divided by the square roots of the eigenvalues to produce scores that have unit standard deviations.

When you do not specify the COV option, you can produce the principal component scores by multiplying the standardized data by these coefficients. When you specify the COV option, you can produce the principal component scores by multiplying the centered data by these coefficients. You should use the means, obtained from the observation for which _TYPE_='MEAN', to center the data. You should use the standard deviations, obtained from the observation for which _TYPE_='STD', to standardize the data.

USCORE scoring coefficients to be applied without subtracting the mean from the raw variables. Observations for which _TYPE_='USCORE' are produced when you specify the NOINT option in the PROC PRINCOMP statement.

To obtain the principal component scores, these coefficients should be multiplied by the data that are standardized by the uncorrected standard deviations obtained from the observation for which _TYPE_='USTD'.

RSQUARED R squares for each VAR variable as predicted by the PARTIAL variables

B regression coefficients for each VAR variable as predicted by the PARTIAL variables. This observation is produced only if you specify the COV option.

STB standardized regression coefficients for each VAR variable as predicted by the PARTIAL variables. If you specify the COV option, this observation is omitted.

You can use the data set with the SCORE procedure to compute principal component scores, or you can use it as input to the FACTOR procedure and specify METHOD=SCORE to rotate the components. If you use the PARTIAL statement, the scoring coefficients should be applied to the residuals, not to the original variables.

Computational Resources

Let

$$
\begin{aligned}
n &= \text{number of observations} \\
v &= \text{number of VAR variables} \\
p &= \text{number of PARTIAL variables} \\
c &= \text{number of components}
\end{aligned}
$$

- The minimum allocated memory required (in bytes) is

$$232v + 120p + 48c + \max(8cv, 8vp + 4(v+p)(v+p+1))$$

- The time required to compute the correlation matrix is approximately proportional to

$$n(v+p)^2 + \frac{p}{2}(v+p)(v+p+1)$$

- The time required to compute eigenvalues is approximately proportional to v^3.
- The time required to compute eigenvectors is approximately proportional to cv^2.

Displayed Output

The PRINCOMP procedure displays the following items if the DATA= data set is not TYPE=CORR, TYPE=COV, TYPE=SSCP, TYPE=UCORR, or TYPE=UCOV:

- simple statistics, including the mean and standard deviation (StD) for each variable. If you specify the NOINT option, the uncorrected standard deviation (UStD) is displayed.
- the correlation or, if you specify the COV option, the covariance matrix

The PRINCOMP procedure displays the following items if you use the PARTIAL statement:

- regression statistics, giving the R square and root mean squared error (RMSE) for each VAR variable as predicted by the PARTIAL variables (not shown)
- standardized regression coefficients or, if you specify the COV option, regression coefficients for predicting the VAR variables from the PARTIAL variables (not shown)
- the partial correlation matrix or, if you specify the COV option, the partial covariance matrix (not shown)

The PRINCOMP procedure displays the following item if you specify the COV option:

- the total variance

The PRINCOMP procedure displays the following items unless you specify the NOPRINT option:

- eigenvalues of the correlation or covariance matrix, in addition to the difference between successive eigenvalues, the proportion of variance explained by each eigenvalue, and the cumulative proportion of variance explained
- the eigenvectors

ODS Table Names

PROC PRINCOMP assigns a name to each table that it creates. You can use these names to reference the ODS table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 98.2. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

All the tables are created by specifying the PROC PRINCOMP statement; a few tables need an additional PARTIAL statement.

Table 98.2 ODS Tables Produced by PROC PRINCOMP

ODS Table Name	Description	Statement / Option
Corr	Correlation matrix	Default
Cov	Covariance matrix	COV
Eigenvalues	Eigenvalues	Default
Eigenvectors	Eigenvectors	Default
NObsNVar	Number of observations, variables, and partial variables	Default
ParCorr	Partial correlation matrix	PARTIAL statement
ParCov	Uncorrected partial covariance matrix	PARTIAL statement and COV
RegCoef	Regression coefficients	PARTIAL statement and COV
RSquareRMSE	Regression statistics: R squares and RMSEs	PARTIAL statement
SimpleStatistics	Simple statistics	Default
StdRegCoef	Standardized regression coefficients	PARTIAL statement
TotalVariance	Total variance	COV

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

Some graphs are produced by default; other graphs are produced by using statements and options. You can reference every graph produced through ODS Graphics by name. The names of the graphs that PROC PRINCOMP generates are listed in Table 98.3, along with a description of each graph and the required statements and options.

Table 98.3 Graphs Produced by PROC PRINCOMP

ODS Graph Name	Plot Description	Statement and Option
PaintedScorePlot	Score plot of component i versus component j, painted by component k	PLOTS=SCORE when number of variables ≥ 3
PatternPlot	Component pattern plot	PLOTS=PATTERN
PatternProfilePlot	Component pattern profile plot	PLOTS=PATTERNPROFILE
ScoreMatrixPlot	Matrix plot of component scores	PLOTS=MATRIX
ScorePlot	Component score plot	PLOTS=SCORE
ScreePlot	Scree and variance plots	Default and PLOTS=SCREE
VariancePlot	Variance proportion explained plot	PLOTS=SCREE(UNPACKPANEL)

Examples: PRINCOMP Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 98.1: Analyzing Mean Temperatures of US Cities

(View the complete code for this example (princex1.sas) in the example repository.)

This example analyzes mean daily temperatures of selected US cities in January and July. Both the raw data and the principal components are plotted to illustrate that principal components are orthogonal rotations of the original variables.

The following statements create the Temperature data set:

```
data Temperature;
   length CityId $ 2;
   title 'Mean Temperature in January and July for Selected Cities';
   input City $ 1-15 January July;
   CityId = substr(City,1,2);
   datalines;
Mobile          51.2 81.6
Phoenix         51.2 91.2
Little Rock     39.5 81.4
Sacramento      45.1 75.2
Denver          29.9 73.0

   ... more lines ...

Cheyenne        26.6 69.1
;
```

The following statements plot the Temperature data set. The variable Cityid instead of City is used as a data label in the scatter plot to avoid label collisions.

```
title 'Mean Temperature in January and July for Selected Cities';
proc sgplot data=Temperature;
   scatter x=July y=January / datalabel=CityId;
run;
```

The results are displayed in Output 98.1.1, which shows a scatter plot of the 64 pairs of data points in which July temperatures are plotted against January temperatures.

Output 98.1.1 Plot of Raw Data

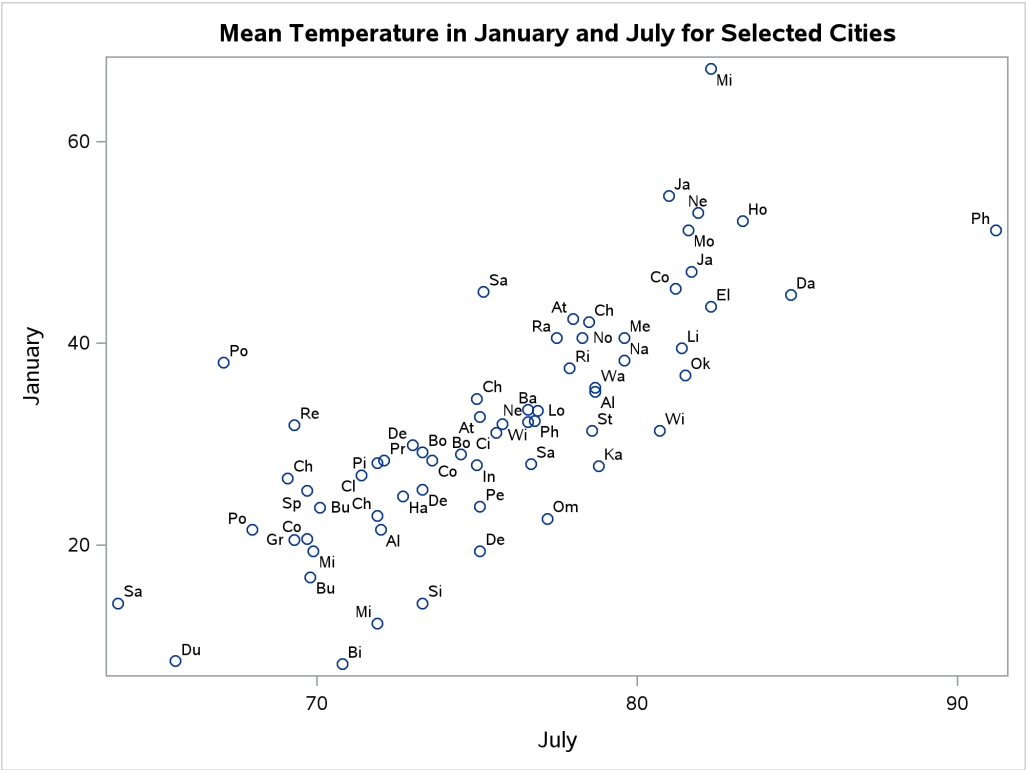

The following step requests a principal component analysis of the Temperature data set:

```
ods graphics on;

title 'Mean Temperature in January and July for Selected Cities';
proc princomp data=Temperature cov plots=score(ellipse);
   var July January;
   id CityId;
run;
```

Output 98.1.2 displays the PROC PRINCOMP output. The standard deviation of January (11.712) is higher than the standard deviation of July (5.128). The COV option in the PROC PRINCOMP statement requests that the principal components be computed from the covariance matrix. The total variance is 163.474. The first principal component accounts for about 94% of the total variance, and the second principal component accounts for only about 6%. The eigenvalues sum to the total variance.

Note that January receives a higher loading on Prin1 because it has a higher standard deviation than July. Also note that the PRINCOMP procedure calculates the scores by using the centered variables rather than the

standardized variables.

Output 98.1.2 Results of Principal Component Analysis

Mean Temperature in January and July for Selected Cities

The PRINCOMP Procedure

Observations	64
Variables	2

Simple Statistics

	July	January
Mean	75.60781250	32.09531250
StD	5.12761910	11.71243309

Covariance Matrix

	July	January
July	26.2924777	46.8282912
January	46.8282912	137.1810888

Total Variance	163.47356647

Eigenvalues of the Covariance Matrix

	Eigenvalue	Difference	Proportion	Cumulative
1	154.310607	145.147647	0.9439	0.9439
2	9.162960		0.0561	1.0000

Eigenvectors

	Prin1	Prin2
July	0.343532	0.939141
January	0.939141	-.343532

The PLOTS=SCORE option in the PROC PRINCOMP statement requests a plot of the second principal component against the first principal component, as shown in Output 98.1.3. It is clear from this plot that the principal components are orthogonal rotations of the original variables and that the first principal component has a larger variance than the second principal component. In fact, the first component has a larger variance than either of the original variables, July and January. The ellipse indicates that Miami, Phoenix, and Portland are possible outliers.

Output 98.1.3 Plot of Component 2 by Component 1

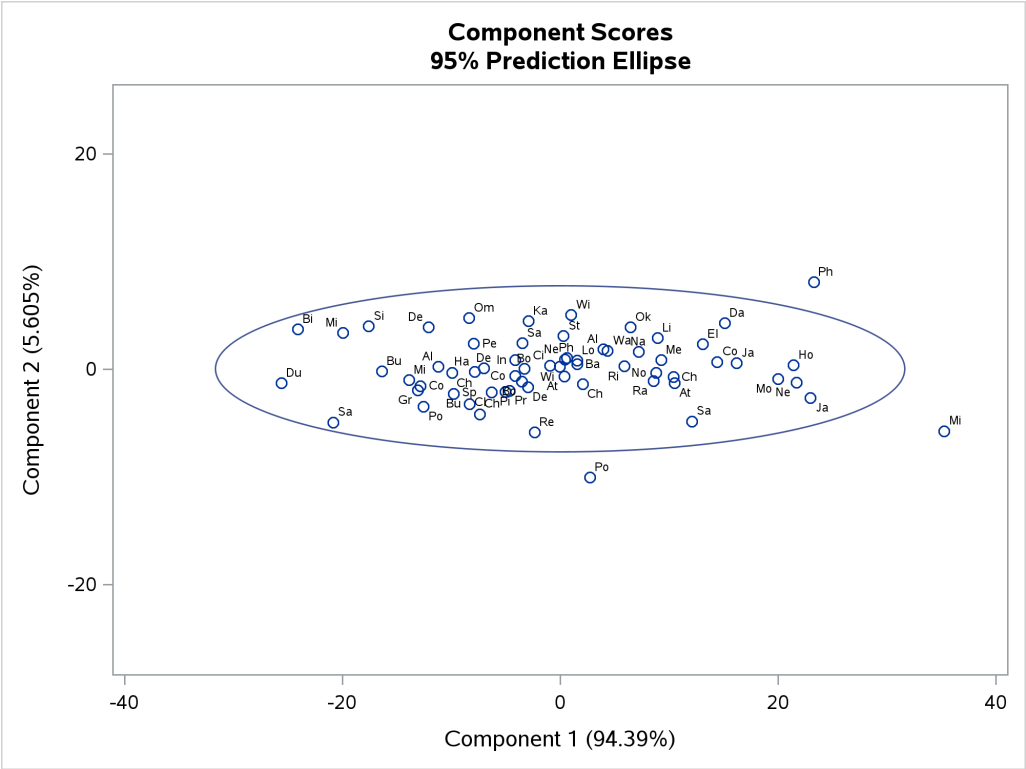

Example 98.2: Analyzing Rankings of US College Basketball Teams

(View the complete code for this example (princex2.sas) in the example repository.)

The data in this example are rankings of 35 US college basketball teams. The rankings were made before the start of the 1985–86 season by 10 news services. The purpose of the principal component analysis is to compute a single variable that best summarizes all 10 preseason rankings. Note that the various news services rank different numbers of teams, ranging from 20 to 30 (one of the variables, WashPost, has a missing rank). And, of course, not all news services rank the same teams, so there are missing values in these data. Each of the 35 teams is ranked by at least one news service.

The PRINCOMP procedure omits observations that have missing values. To obtain principal component scores for all the teams, you must replace the missing values. Because it is the best teams that are ranked, it is not appropriate to replace missing values with the mean of the nonmissing values. Instead, an ad hoc method is used that replaces missing values with the mean of the unassigned ranks. For example, if a news service ranks 20 teams, then ranks 21 through 35 are unassigned. The mean of ranks 21 through 35 is 28, so missing values for that variable are replaced by the value 28. To prevent the method of missing-value replacement from having an undue effect on the analysis, each observation is weighted according to the number of nonmissing values that it has. For an alternative analysis of these data, see Example 99.2 in Chapter 99, "The PRINQUAL Procedure."

Because the first principal component accounts for 78% of the variance, there is substantial agreement among the rankings. The eigenvector shows that all the news services are about equally weighted; this is also

suggested by the nearly horizontal line of the pattern profile plot in Output 98.2.3. So a simple average would work almost as well as the first principal component. The following statements produce Output 98.2.1.

```
/*-----------------------------------------------------------*/
/*                                                           */
/* Pre-season 1985 College Basketball Rankings               */
/* (rankings of 35 teams by 10 news services)                */
/*                                                           */
/* Note: (a) news services rank varying numbers of teams;    */
/*       (b) not all teams are ranked by all news services;  */
/*       (c) each team is ranked by at least one service;    */
/*       (d) rank 20 is missing for UPI.                     */
/*                                                           */
/*-----------------------------------------------------------*/

data HoopsRanks;
   input School $13. CSN DurSun DurHer WashPost USAToday
         Sport InSports UPI AP SI;
   label CSN      = 'Community Sports News (Chapel Hill, NC)'
         DurSun   = 'Durham Sun'
         DurHer   = 'Durham Morning Herald'
         WashPost = 'Washington Post'
         USAToday = 'USA Today'
         Sport    = 'Sport Magazine'
         InSports = 'Inside Sports'
         UPI      = 'United Press International'
         AP       = 'Associated Press'
         SI       = 'Sports Illustrated'
         ;
   format CSN--SI 5.1;
   datalines;
Louisville     1  8  1  9  8  9  6 10  9  9
Georgia Tech   2  2  4  3  1  1  1  2  1  1
Kansas         3  4  5  1  5 11  8  4  5  7
Michigan       4  5  9  4  2  5  3  1  3  2
Duke           5  6  7  5  4 10  4  5  6  5
UNC            6  1  2  2  3  4  2  3  2  3
Syracuse       7 10  6 11  6  6  5  6  4 10
Notre Dame     8 14 15 13 11 20 18 13 12  .
Kentucky       9 15 16 14 14 19 11 12 11 13
LSU           10  9 13  . 13 15 16  9 14  8
DePaul        11  . 21 15 20  . 19  .  . 19
Georgetown    12  7  8  6  9  2  9  8  8  4
Navy          13 20 23 10 18 13 15  . 20  .
Illinois      14  3  3  7  7  3 10  7  7  6
Iowa          15 16  .  . 23  .  . 14  . 20
Arkansas      16  .  .  . 25  .  .  .  . 16
Memphis State 17  . 11  . 16  8 20  . 15 12
Washington    18  .  .  .  .  .  . 17  .  .
UAB           19 13 10  . 12 17  . 16 16 15
UNLV          20 18 18 19 22  . 14 18 18  .
NC State      21 17 14 16 15  . 12 15 17 18
Maryland      22  .  .  . 19  .  .  . 19 14
Pittsburgh    23  .  .  .  .  .  .  .  .  .
```

```
Oklahoma      24 19 17 17 17 12 17  . 13 17
Indiana       25 12 20 18 21  .  .  .  .  .
Virginia      26  . 22  .  . 18  .  .  .  .
Old Dominion  27  .  .  .  .  .  .  .  .  .
Auburn        28 11 12  8 10  7  7 11 10 11
St. Johns     29  .  .  .  . 14  .  .  .  .
UCLA          30  .  .  .  .  .  . 19  .  .
St. Joseph's   .  . 19  .  .  .  .  .  .  .
Tennessee      .  . 24  .  . 16  .  .  .  .
Montana        .  .  . 20  .  .  .  .  .  .
Houston        .  .  .  . 24  .  .  .  .  .
Virginia Tech  .  .  .  .  .  . 13  .  .  .
;

/* PROC MEANS is used to output a data set containing the      */
/* maximum value of each of the newspaper and magazine         */
/* rankings.  The output data set, maxrank, is then used       */
/* to set the missing values to the next highest rank plus     */
/* thirty-six, divided by two (that is, the mean of the        */
/* missing ranks).  This ad hoc method of replacing missing    */
/* values is based more on intuition than on rigorous          */
/* statistical theory.  Observations are weighted by the       */
/* number of nonmissing values.                                */
/*                                                             */

title 'Pre-Season 1985 College Basketball Rankings';
proc means data=HoopsRanks;
   output out=MaxRank
          max=CSNMax DurSunMax DurHerMax
              WashPostMax USATodayMax SportMax
              InSportsMax UPIMax APMax SIMax;
run;
```

Output 98.2.1 Summary Statistics for Basketball Rankings from Using PROC MEANS

Pre-Season 1985 College Basketball Rankings

The MEANS Procedure

Variable	Label	N	Mean	Std Dev	Minimum	Maximum
CSN	Community Sports News (Chapel Hill, NC)	30	15.5000000	8.8034084	1.0000000	30.0000000
DurSun	Durham Sun	20	10.5000000	5.9160798	1.0000000	20.0000000
DurHer	Durham Morning Herald	24	12.5000000	7.0710678	1.0000000	24.0000000
WashPost	Washington Post	19	10.4210526	6.0673607	1.0000000	20.0000000
USAToday	USA Today	25	13.0000000	7.3598007	1.0000000	25.0000000
Sport	Sport Magazine	20	10.5000000	5.9160798	1.0000000	20.0000000
InSports	Inside Sports	20	10.5000000	5.9160798	1.0000000	20.0000000
UPI	United Press International	19	10.0000000	5.6273143	1.0000000	19.0000000
AP	Associated Press	20	10.5000000	5.9160798	1.0000000	20.0000000
SI	Sports Illustrated	20	10.5000000	5.9160798	1.0000000	20.0000000

The following statements produce Output 98.2.2 and Output 98.2.3:

```
data Basketball;
   set HoopsRanks;
   if _n_=1 then set MaxRank;
   array Services{10} CSN--SI;
   array MaxRanks{10} CSNMax--SIMax;
   keep School CSN--SI Weight;
   Weight=0;
   do i=1 to 10;
      if Services{i}=. then Services{i}=(MaxRanks{i}+36)/2;
      else Weight=Weight+1;
   end;
run;

ods graphics on;

proc princomp data=Basketball n=1 out=PCBasketball standard
              plots=patternprofile;
   var CSN--SI;
   weight Weight;
run;
```

Output 98.2.2 Principal Component Analysis of Basketball Rankings by Using PROC PRINCOMP

Pre-Season 1985 College Basketball Rankings

The PRINCOMP Procedure

Observations	35
Variables	10

Simple Statistics

	CSN	DurSun	DurHer	WashPost	USAToday	Sport
Mean	13.33640553	13.06451613	12.88018433	13.83410138	12.55760369	13.83870968
StD	22.08036285	21.66394183	21.38091837	23.47841791	20.48207965	23.37756267

Simple Statistics

	InSports	UPI	AP	SI
Mean	13.24423963	13.59216590	12.83410138	13.52534562
StD	22.20231526	23.25602811	21.40782406	22.93219584

Output 98.2.2 *continued*

Correlation Matrix

		CSN	DurSun	DurHer	WashPost	USAToday	Sport	InSports	UPI	AP	SI
CSN	Community Sports News (Chapel Hill, NC)	1.0000	0.6505	0.6415	0.6121	0.7456	0.4806	0.6558	0.7007	0.6779	0.6135
DurSun	Durham Sun	0.6505	1.0000	0.8341	0.7667	0.8860	0.6940	0.7702	0.9015	0.8437	0.7518
DurHer	Durham Morning Herald	0.6415	0.8341	1.0000	0.7035	0.8877	0.7788	0.7900	0.7676	0.8788	0.7761
WashPost	Washington Post	0.6121	0.7667	0.7035	1.0000	0.7984	0.6598	0.8717	0.6953	0.7809	0.5952
USAToday	USA Today	0.7456	0.8860	0.8877	0.7984	1.0000	0.7716	0.8475	0.8539	0.9479	0.8426
Sport	Sport Magazine	0.4806	0.6940	0.7788	0.6598	0.7716	1.0000	0.7176	0.6220	0.8217	0.7701
InSports	Inside Sports	0.6558	0.7702	0.7900	0.8717	0.8475	0.7176	1.0000	0.7920	0.8830	0.7332
UPI	United Press International	0.7007	0.9015	0.7676	0.6953	0.8539	0.6220	0.7920	1.0000	0.8436	0.7738
AP	Associated Press	0.6779	0.8437	0.8788	0.7809	0.9479	0.8217	0.8830	0.8436	1.0000	0.8212
SI	Sports Illustrated	0.6135	0.7518	0.7761	0.5952	0.8426	0.7701	0.7332	0.7738	0.8212	1.0000

Eigenvalues of the Correlation Matrix

	Eigenvalue	Difference	Proportion	Cumulative
1	7.88601647		0.7886	0.7886

Eigenvectors

		Prin1
CSN	Community Sports News (Chapel Hill, NC)	0.270205
DurSun	Durham Sun	0.326048
DurHer	Durham Morning Herald	0.324392
WashPost	Washington Post	0.300449
USAToday	USA Today	0.345200
Sport	Sport Magazine	0.293881
InSports	Inside Sports	0.324088
UPI	United Press International	0.319902
AP	Associated Press	0.342151
SI	Sports Illustrated	0.308570

Output 98.2.3 Pattern Profile Plot

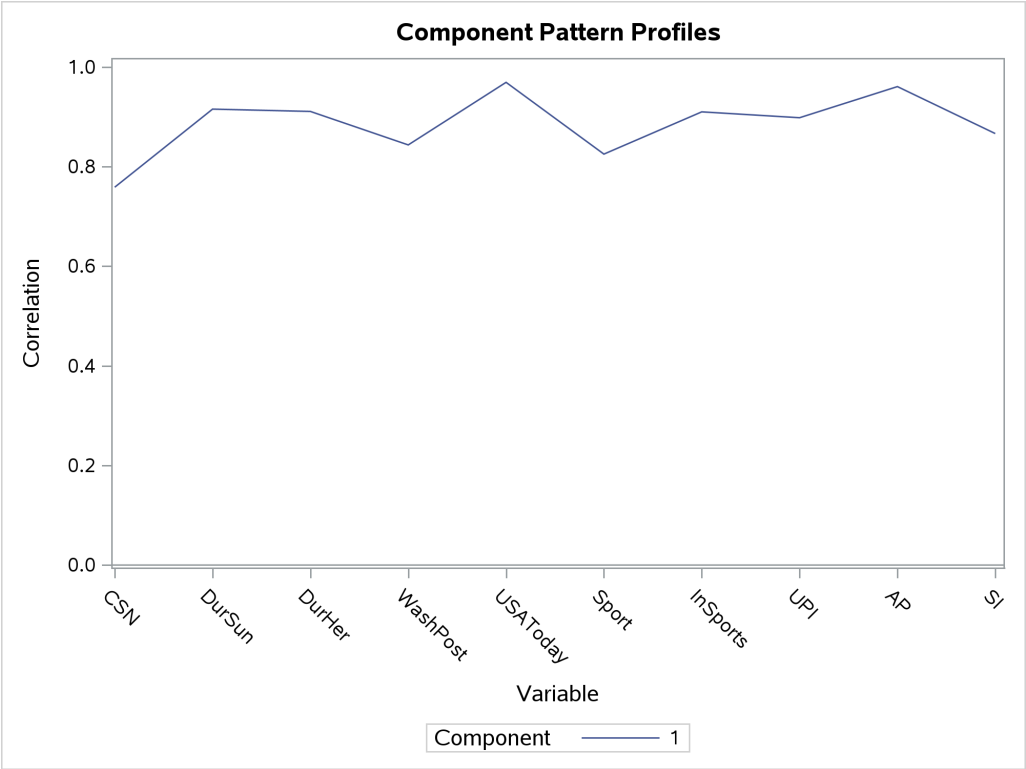

The following statements produce Output 98.2.4:

```
proc sort data=PCBasketball;
   by Prin1;
run;

proc print;
   var School Prin1;
   title 'Pre-Season 1985 College Basketball Rankings';
   title2 'College Teams as Ordered by PROC PRINCOMP';
run;
```

Output 98.2.4 Basketball Rankings from Using PROC PRINCOMP

Pre-Season 1985 College Basketball Rankings
College Teams as Ordered by PROC PRINCOMP

Obs	School	Prin1
1	Georgia Tech	-0.58068
2	UNC	-0.53317
3	Michigan	-0.47874
4	Kansas	-0.40285
5	Duke	-0.38464
6	Illinois	-0.33586
7	Syracuse	-0.31578
8	Louisville	-0.31489
9	Georgetown	-0.29735
10	Auburn	-0.09785
11	Kentucky	0.00843
12	LSU	0.00872
13	Notre Dame	0.09407
14	NC State	0.19404
15	UAB	0.19771
16	Oklahoma	0.23864
17	Memphis State	0.25319
18	Navy	0.28921
19	UNLV	0.35103
20	DePaul	0.43770
21	Iowa	0.50213
22	Indiana	0.51713
23	Maryland	0.55910
24	Arkansas	0.62977
25	Virginia	0.67586
26	Washington	0.67756
27	Tennessee	0.70822
28	St. Johns	0.71425
29	Virginia Tech	0.71638
30	St. Joseph's	0.73492
31	UCLA	0.73965
32	Pittsburgh	0.75078
33	Houston	0.75534
34	Montana	0.75790
35	Old Dominion	0.76821

Example 98.3: Analyzing Job Ratings of Police Officers

(View the complete code for this example (princex3.sas) in the example repository.)

This example uses the PRINCOMP procedure to analyze job performance. Police officers were rated by their supervisors in 14 categories as part of standard police department administrative procedure.

The following statements create the Jobratings data set:

```
options validvarname=any;
data Jobratings;
   input 'Communication Skills'n         'Problem Solving'n
         'Learning Ability'n             'Judgment Under Pressure'n
         'Observational Skills'n         'Willingness to Confront Problems'n
         'Interest in People'n           'Interpersonal Sensitivity'n
         'Desire for Self-Improvement'n  'Appearance'n
         'Dependability'n                'Physical Ability'n
         'Integrity'n                    'Overall Rating'n @@;
   datalines;
2 6 8 3 8 8 5 3 8 7 9 8 6 7 7 4 7 5 8 8 7 6 8 5 7 6 6 7 5 6 7 5 7 8 6 3 7 7 5
8 7 5 6 7 8 6 9 7 7 7 9 8 8 9 9 7 9 9 9 9 7 7 9 8 8 7 8 8 8 8 8 9 8 9 7 8 9 9
8 8 8 7 9 9 8 9 9 9 9 8 8 9 8 9 9 7 9 8 8 7 7 9 4 7 9 8 4 6 8 8 8 6 3 5 6 5 2

   ... more lines ...

7 8 9 9 7 9 9 7 9 9 9 9 8 9 9 8 9 9 8 9 9 8 9 9 7 6 6 5 6 3 9 9 5 6 7 4 8 6
;
```

The Jobratings data set contains 14 variables. Each variable contains the job ratings, which use a scale measurement from 1 to 10 (1=fail to comply, 10=exceptional). The last variable, Overall Rating, contains a score as an overall index of how each officer performs.

The following statements request a principal component analysis of the Jobratings data set, output the scores to the Scores data set (OUT= Scores), and produce default plots. Note that the variable Overall Rating is excluded from the analysis.

```
ods graphics on;

proc princomp data=Jobratings(drop='Overall Rating'n);
run;
```

Figure 98.3.1 and Figure 98.3.2 display the PROC PRINCOMP output, beginning with simple statistics and then the correlation matrix. By default, PROC PRINCOMP computes principal components from the correlation matrix, so the total variance is equal to the number of variables, 13. In this example, it would also be reasonable to use the COV option, which would cause variables that have a high variance (such as Dependability) to influence the results more than variables that have a low variance (such as Learning Ability). If you used the COV option, scores would be computed from centered rather than standardized variables.

Output 98.3.1 Simple Statistics and Correlation Matrix from Using PROC PRINCOMP

The PRINCOMP Procedure

Observations	103
Variables	13

Simple Statistics

	Communication Skills	Problem Solving	Learning Ability	Judgment Under Pressure	Observational Skills	Willingness to Confront Problems	Interest in People
Mean	6.650485437	6.631067961	6.990291262	6.737864078	6.932038835	7.291262136	6.708737864
StD	1.764068036	1.590352602	1.339411238	1.731830976	1.761584269	1.525155524	1.892353385

Simple Statistics

	Interpersonal Sensitivity	Desire for Self-Improvement	Appearance	Dependability	Physical Ability	Integrity
Mean	6.621359223	6.572815534	7.000000000	6.825242718	7.203883495	7.213592233
StD	1.760773587	1.729796212	1.798692335	1.917040123	1.555251845	1.845240223

Output 98.3.1 *continued*

Correlation Matrix

	Communication Skills	Problem Solving	Learning Ability	Judgment Under Pressure	Observational Skills	Willingness to Confront Problems	Interest in People
Communication Skills	1.0000	0.6280	0.5546	0.5538	0.5381	0.5265	0.4391
Problem Solving	0.6280	1.0000	0.5690	0.6195	0.4284	0.5015	0.3972
Learning Ability	0.5546	0.5690	1.0000	0.4892	0.6230	0.5245	0.2735
Judgment Under Pressure	0.5538	0.6195	0.4892	1.0000	0.3733	0.4004	0.6226
Observational Skills	0.5381	0.4284	0.6230	0.3733	1.0000	0.7300	0.2616
Willingness to Confront Problems	0.5265	0.5015	0.5245	0.4004	0.7300	1.0000	0.2233
Interest in People	0.4391	0.3972	0.2735	0.6226	0.2616	0.2233	1.0000
Interpersonal Sensitivity	0.5030	0.4398	0.1855	0.6134	0.1655	0.1291	0.8051
Desire for Self-Improvement	0.5642	0.4090	0.5737	0.4826	0.5985	0.5307	0.4857
Appearance	0.4913	0.3873	0.3988	0.2266	0.4177	0.4825	0.2679
Dependability	0.5471	0.4546	0.5110	0.5471	0.5626	0.4870	0.6074
Physical Ability	0.2192	0.3201	0.2269	0.3476	0.4274	0.4872	0.3768
Integrity	0.5081	0.3846	0.3142	0.5883	0.3906	0.3260	0.7452

Correlation Matrix

	Interpersonal Sensitivity	Desire for Self-Improvement	Appearance	Dependability	Physical Ability	Integrity
Communication Skills	0.5030	0.5642	0.4913	0.5471	0.2192	0.5081
Problem Solving	0.4398	0.4090	0.3873	0.4546	0.3201	0.3846
Learning Ability	0.1855	0.5737	0.3988	0.5110	0.2269	0.3142
Judgment Under Pressure	0.6134	0.4826	0.2266	0.5471	0.3476	0.5883
Observational Skills	0.1655	0.5985	0.4177	0.5626	0.4274	0.3906
Willingness to Confront Problems	0.1291	0.5307	0.4825	0.4870	0.4872	0.3260
Interest in People	0.8051	0.4857	0.2679	0.6074	0.3768	0.7452
Interpersonal Sensitivity	1.0000	0.3713	0.2600	0.5408	0.2182	0.6920
Desire for Self-Improvement	0.3713	1.0000	0.4474	0.5981	0.3752	0.5664
Appearance	0.2600	0.4474	1.0000	0.5089	0.3820	0.4135
Dependability	0.5408	0.5981	0.5089	1.0000	0.4461	0.6536
Physical Ability	0.2182	0.3752	0.3820	0.4461	1.0000	0.3810
Integrity	0.6920	0.5664	0.4135	0.6536	0.3810	1.0000

Figure 98.3.2 displays the eigenvalues. The first principal component accounts for about 50% of the total variance, the second principal component accounts for about 13.6%, and the third principal component accounts for about 7.7%. Note that the eigenvalues sum to the total variance. The eigenvalues indicate that three to five components provide a good summary of the data: three components account for about 71.7% of the total variance, and five components account for about 82.7%. Subsequent components account for less than 5% each.

Output 98.3.2 Eigenvalues and Eigenvectors from Using PROC PRINCOMP

Eigenvalues of the Correlation Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	6.54740242	4.77468744	0.5036	0.5036
2	1.77271499	0.76747933	0.1364	0.6400
3	1.00523565	0.26209665	0.0773	0.7173
4	0.74313901	0.06479499	0.0572	0.7745
5	0.67834402	0.22696368	0.0522	0.8267
6	0.45138034	0.06922167	0.0347	0.8614
7	0.38215866	0.08432613	0.0294	0.8908
8	0.29783254	0.02340663	0.0229	0.9137
9	0.27442591	0.01208809	0.0211	0.9348
10	0.26233782	0.01778332	0.0202	0.9550
11	0.24455450	0.04677622	0.0188	0.9738
12	0.19777828	0.05508241	0.0152	0.9890
13	0.14269586		0.0110	1.0000

Eigenvectors							
	Prin1	Prin2	Prin3	Prin4	Prin5	Prin6	Prin7
Communication Skills	0.303548	0.052039	-.329181	-.227039	0.181087	-.416563	0.143543
Problem Solving	0.278034	0.057046	-.400112	0.300476	0.453604	0.096750	0.048904
Learning Ability	0.266521	0.288152	-.354591	-.020735	-.219329	0.578388	-.114808
Judgment Under Pressure	0.294376	-.199458	-.255164	0.397306	-.030188	0.102087	0.068204
Observational Skills	0.276641	0.366979	0.065959	0.035711	-.325257	-.301254	-.297894
Willingness to Confront Problems	0.267580	0.392989	0.098723	0.184409	0.038278	-.458585	-.044796
Interest in People	0.278060	-.432916	0.118113	0.046047	-.111279	0.030870	-.011105
Interpersonal Sensitivity	0.253814	-.495662	-.064547	-.060000	0.107807	-.170305	-.088194
Desire for Self-Improvement	0.299833	0.099077	0.061097	-.211279	-.427477	0.105369	0.689011
Appearance	0.237358	0.190065	0.248353	-.544587	0.568044	0.221643	0.049267
Dependability	0.319480	-.049742	0.169476	-.156070	-.130575	0.202301	-.594850
Physical Ability	0.213868	0.097499	0.614959	0.514519	0.203995	0.173168	0.169247
Integrity	0.298246	-.301812	0.190222	-.169062	-.130757	-.100039	0.029456

Eigenvectors						
	Prin8	Prin9	Prin10	Prin11	Prin12	Prin13
Communication Skills	0.333846	-.430955	0.375983	0.028370	-.252778	-.122809
Problem Solving	0.199259	0.256098	-.372914	-.434417	0.069863	-.116642
Learning Ability	0.064088	0.224706	0.287031	0.210540	-.284355	0.248555
Judgment Under Pressure	-.591822	-.358618	0.178270	0.118318	0.306490	-.126636
Observational Skills	0.163484	0.258377	0.223793	-.079692	0.565290	-.168555
Willingness to Confront Problems	-.365684	0.129976	-.330710	0.275249	-.386151	0.177688
Interest in People	0.154829	0.321200	-.081470	0.393841	-.210915	-.610215
Interpersonal Sensitivity	0.192725	0.137468	-.074821	0.285447	0.276824	0.643410
Desire for Self-Improvement	0.087453	-.121474	-.363854	-.052085	0.151436	0.053834
Appearance	-.257497	0.087395	0.061890	0.168369	0.236655	-.113705
Dependability	0.081242	-.495598	-.377561	-.164909	-.090904	-.018094
Physical Ability	0.302536	-.149625	0.258321	-.006202	-.055828	0.133430
Integrity	-.317545	0.271060	0.297010	-.612497	-.276273	0.114965

PROC PRINCOMP produces the scree plot as shown in Figure 98.3.3 by default when ODS Graphics is enabled. You can obtain more plots by specifying the PLOTS= option in the PROC PRINCOMP statement.

The scree plot on the left shows that the eigenvalue of the first component is approximately 6.5 and the eigenvalue of the second component is largely decreased to under 2.0. The variance explained plot on the right shows that the first four principal components account for nearly 80% of the total variance.

Output 98.3.3 Scree Plot from Using PROC PRINCOMP

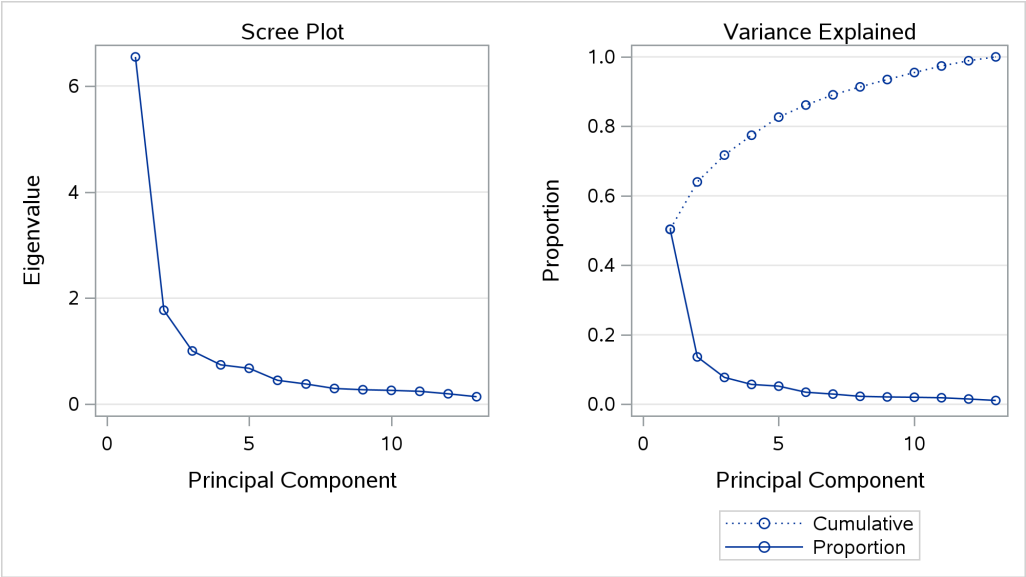

The first component reflects overall performance, because the first eigenvector shows approximately equal loadings on all variables. The second eigenvector has high positive loadings on the variables Observational Skills and Willingness to Confront Problems but even higher negative loadings on the variables Interest in People and Interpersonal Sensitivity. This component seems to reflect the ability to take action, but it also reflects a lack of interpersonal skills. The third eigenvector has a very high positive loading on the variable Physical Ability and high negative loadings on the variables Problem Solving and Learning Ability. This component seems to reflect physical strength, but it also shows poor learning and problem-solving skills.

In short, the three components represent the following:

First component:	overall performance
Second component:	smartness, toughness, and introversion
Third component:	superior strength and average intellect

PROC PRINCOMP also produces other plots besides the scree plot, that help interpret the results. The following statements request plots from the PRINCOMP procedure:

```
proc princomp data=Jobratings(drop='Overall Rating'n)
              n=5 plots(ncomp=3)=all;
run;
```

The N=5 option sets the number of principal components to five. The option PLOTS(NCOMP=3)=ALL produces all plots but limits to three the number of components that are displayed in the component pattern plots and the component score plots.

Output 98.3.4 shows a matrix plot of component scores for the first five principal components. The histogram of each component is displayed in the diagonal element of the matrix. The histograms indicate that the first principal component is skewed to the left and the second principal component is slightly skewed to the right.

Output 98.3.4 Matrix Plot of Component Scores

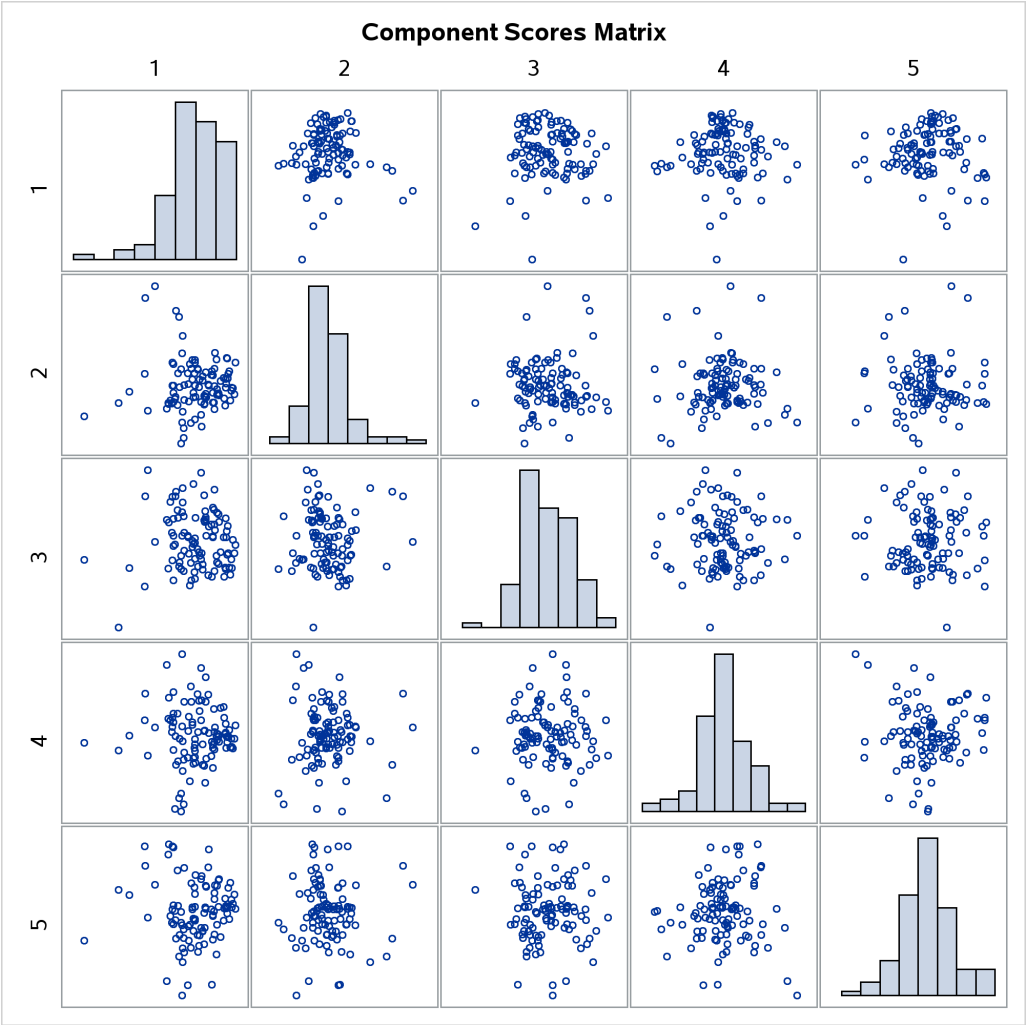

The pairwise component pattern plots are shown in Output 98.3.5 through Output 98.3.7. The pattern plots show the following:

- All variables positively and evenly correlate with the first principal component (Output 98.3.5 and Output 98.3.6).
- The variables Observational Skills and Willingness to Confront Problems correlate highly with the second component, and the variables Interest in People and Interpersonal Sensitivity correlate highly but negatively with the second component (Output 98.3.5).
- The variable Physical Ability correlates highly with the third component, and the variables Problem Solving and Learning Ability correlate highly but negatively with the third component (Output 98.3.6).

- The variables Observational Skills, Willingness to Confront Problems, Interest in People, and Interpersonal Sensitivity correlate highly (either positively or negatively) with the second component, but all these variables have very low correlations with the third component; the variables Physical Ability and Problem Solving correlate highly (either positively or negatively) with the third component, but both variables have very low correlations with the second component (Output 98.3.7).

Output 98.3.5 Pattern Plot of Component 2 by Component 1

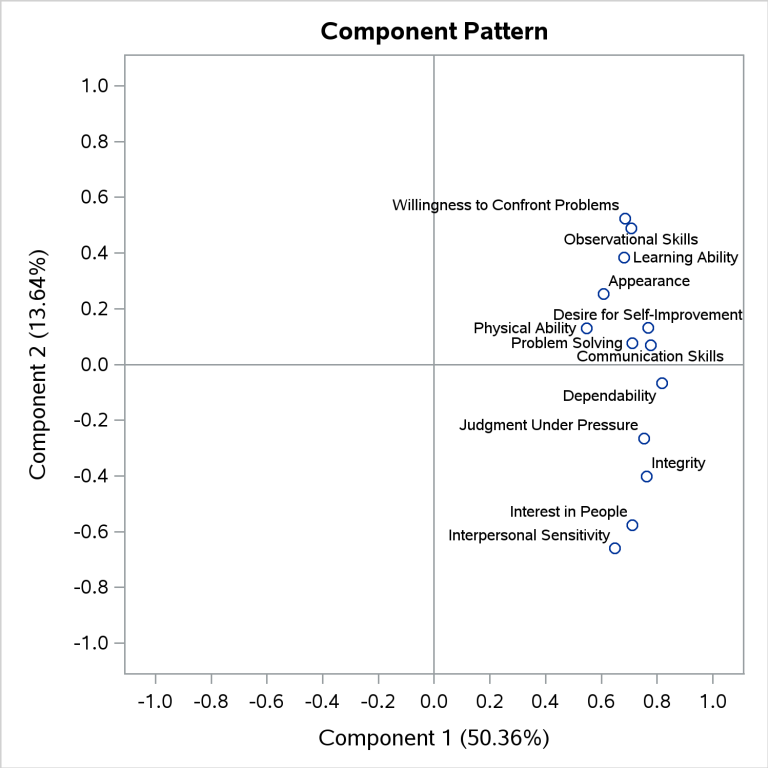

Output 98.3.6 Pattern Plot of Component 3 by Component 1

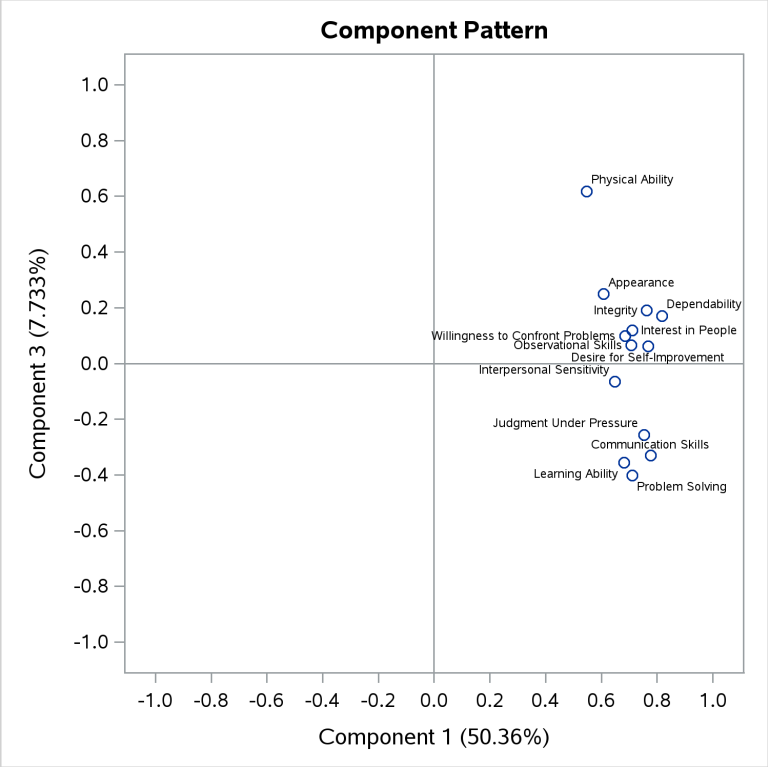

Output 98.3.7 Pattern Plot of Component 3 by Component 2

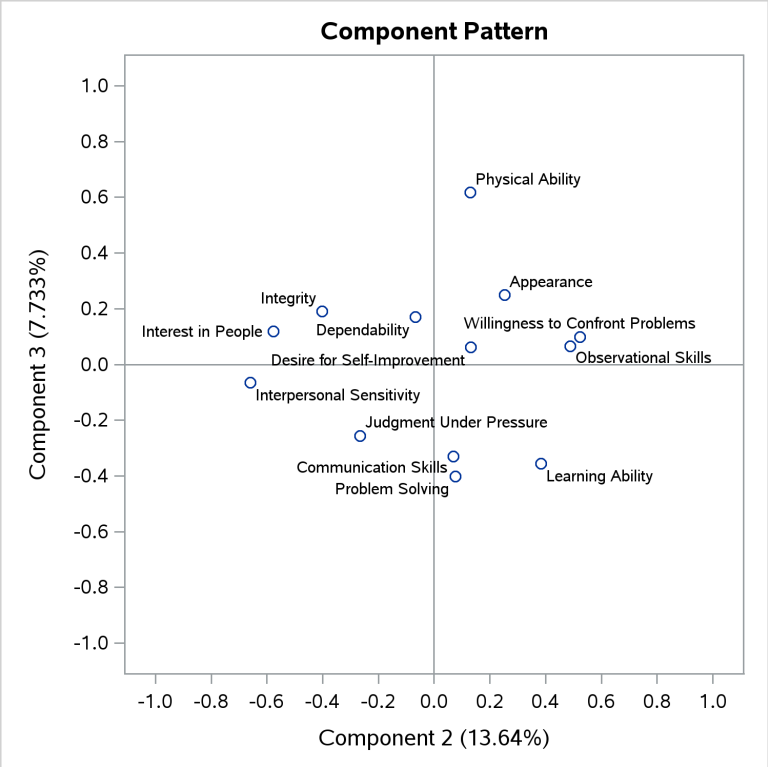

Output 98.3.8 shows a component pattern profile. As is shown in the pattern plots, the nearly horizontal profile from the first component indicates that the first component is mostly correlated evenly across all variables.

Output 98.3.8 Component Pattern Profile Plot from Using PROC PRINCOMP

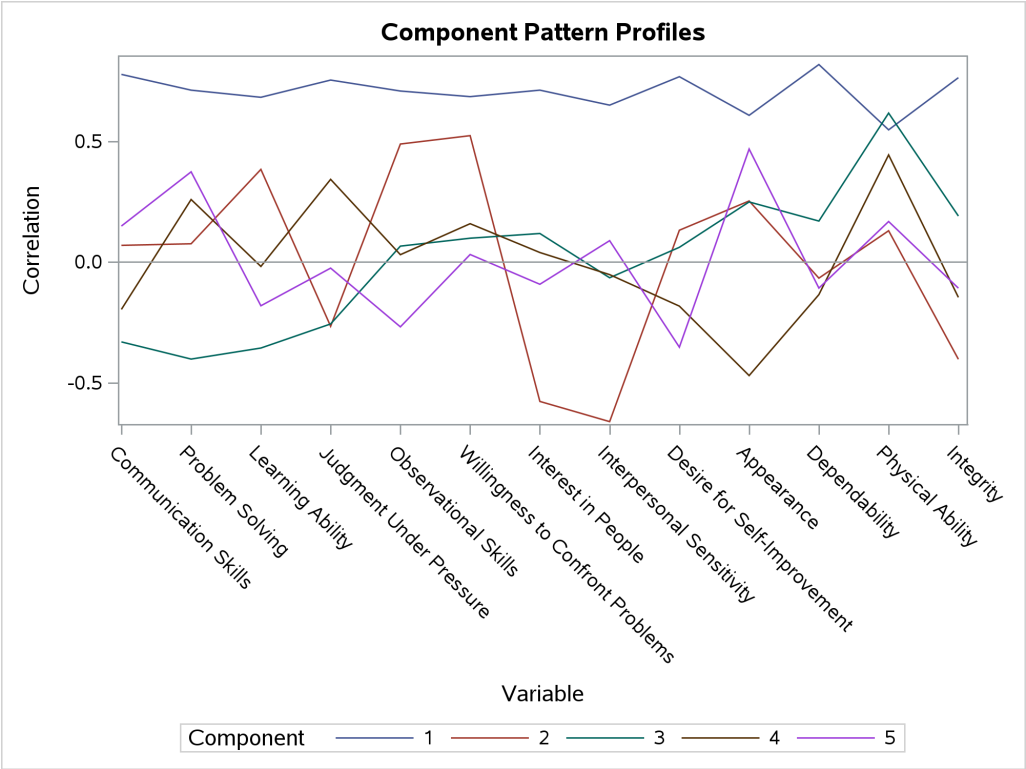

Output 98.3.9 through Output 98.3.11 display the pairwise component score plots. Observation numbers are used as the plotting symbol.

Output 98.3.9 shows a scatter plot of the first and second components. Observations 9, 82, and 84 seem like outliers on the first component. Observations 16 and 59 can be potential outliers on the second component.

Output 98.3.10 shows a scatter plot of the first and third components. Observations 9, 82, and 84 seem like outliers on the first component.

Output 98.3.11 shows a scatter plot of the second and third components. Observations 16 and 59 can be potential outliers on the second component.

Output 98.3.12 shows a scatter plot of the second and third components, displaying the first component in color. Color interpolation ranges from red (minimum) to blue (middle) to green (maximum).

Output 98.3.9 Component 2 versus Component 1

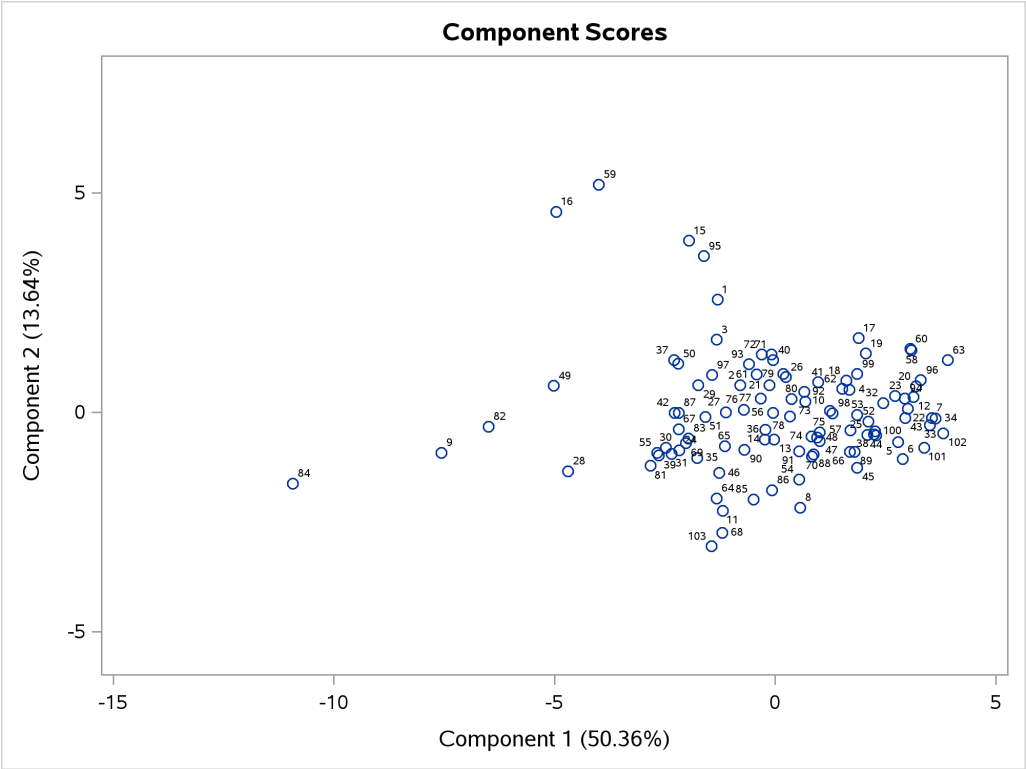

Output 98.3.10 Component 3 versus Component 1

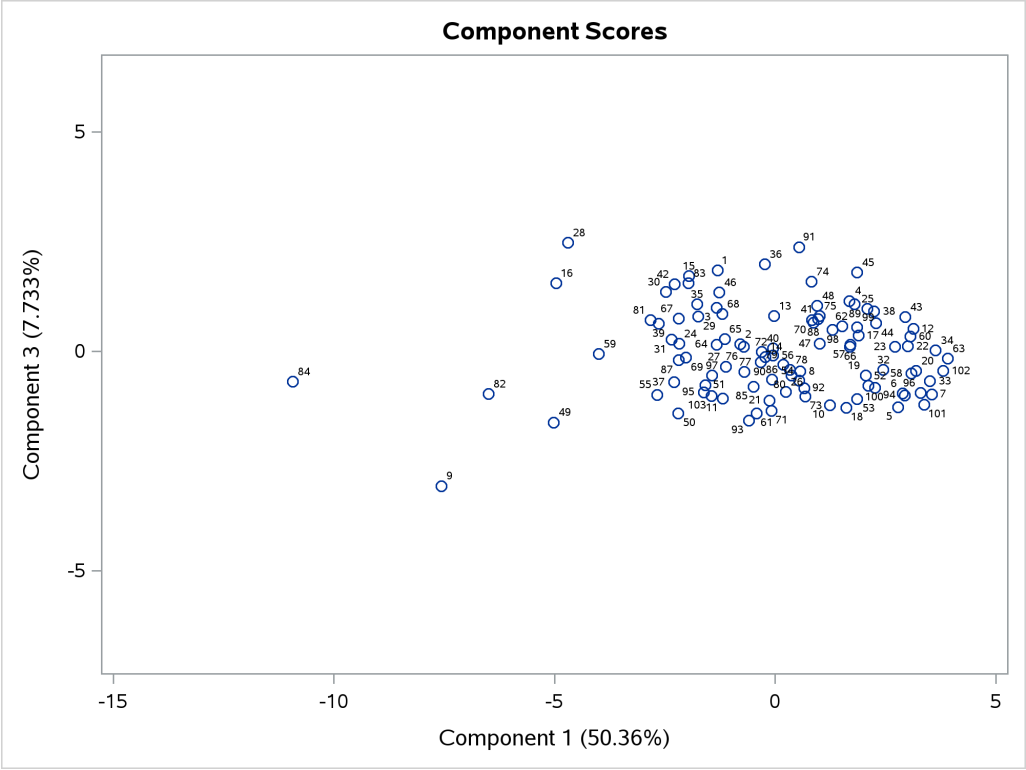

Output 98.3.11 Component 3 versus Component 2

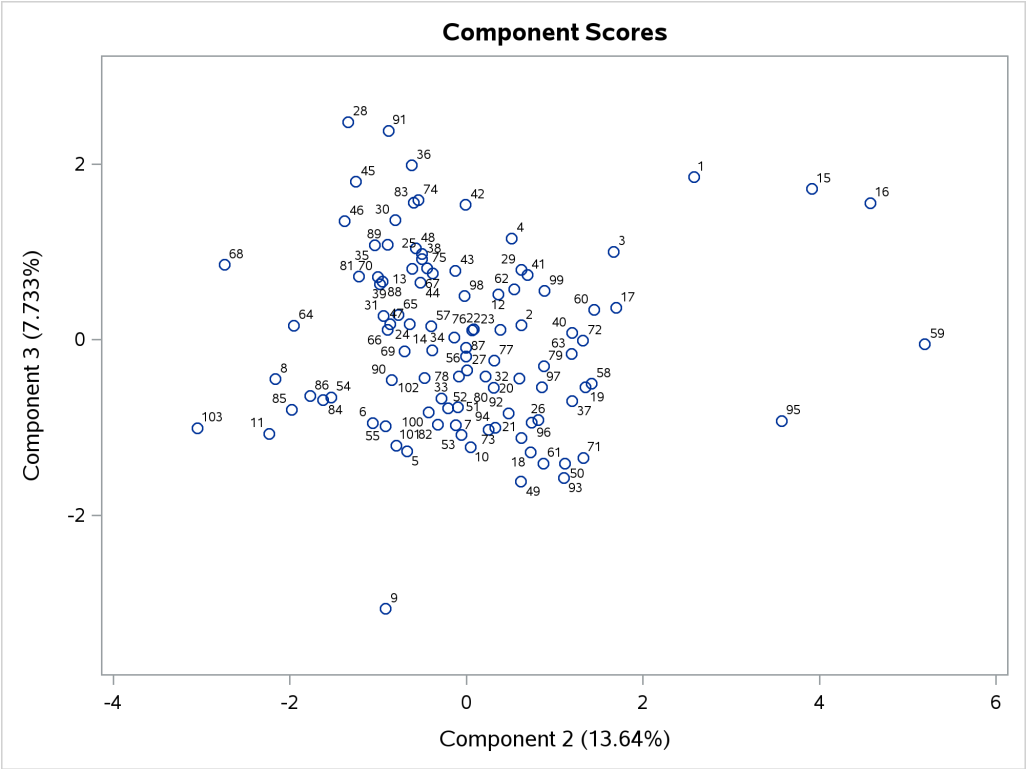

Output 98.3.12 Component 3 versus Component 2, Painted by Component 1

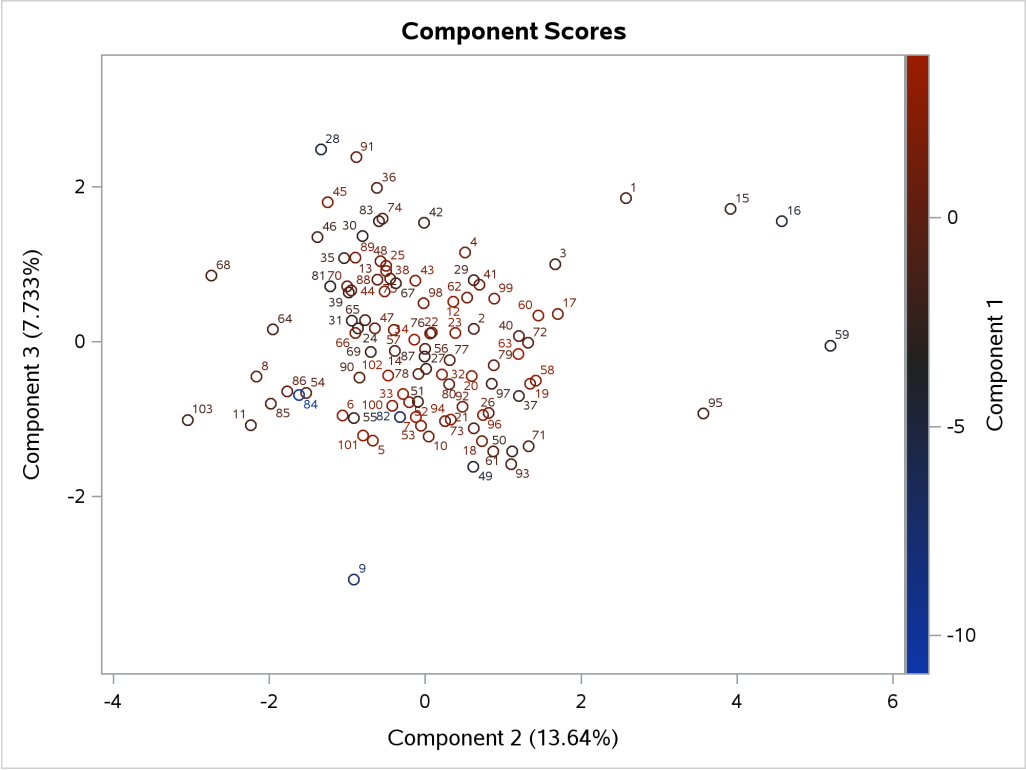

References

Cooley, W. W., and Lohnes, P. R. (1971). *Multivariate Data Analysis*. New York: John Wiley & Sons.

Gnanadesikan, R. (1977). *Methods for Statistical Data Analysis of Multivariate Observations*. New York: John Wiley & Sons.

Hotelling, H. (1933). "Analysis of a Complex of Statistical Variables into Principal Components." *Journal of Educational Psychology* 24:417–441, 498–520.

Kshirsagar, A. M. (1972). *Multivariate Analysis*. New York: Marcel Dekker.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Morrison, D. F. (1976). *Multivariate Statistical Methods*. 2nd ed. New York: McGraw-Hill.

Pearson, K. (1901). "On Lines and Planes of Closest Fit to Systems of Points in Space." *Philosophical Magazine* 6:559–572.

Rao, C. R. (1964). "The Use and Interpretation of Principal Component Analysis in Applied Research." *Sankhyā, Series A* 26:329–358.

Chapter 99
The PRINQUAL Procedure

Contents

Overview: PRINQUAL Procedure **8200**
Getting Started: PRINQUAL Procedure **8201**
Syntax: PRINQUAL Procedure **8206**
- PROC PRINQUAL Statement 8206
- BY Statement 8214
- FREQ Statement 8215
- ID Statement 8215
- TRANSFORM Statement 8215
- WEIGHT Statement 8223

Details: PRINQUAL Procedure **8223**
- The Three Methods of Variable Transformation 8223
- Understanding How PROC PRINQUAL Works 8224
- Splines 8228
- Missing Values 8229
- Controlling the Number of Iterations 8229
- Performing a Principal Component Analysis of Transformed Data 8230
- Using the MAC Method 8230
- Output Data Set 8231
- Avoiding Constant Transformations 8234
- Constant Variables 8234
- Character OPSCORE Variables 8234
- REITERATE Option Usage 8234
- Passive Observations 8235
- Computational Resources 8236
- Displayed Output 8237
- ODS Table Names 8238
- ODS Graphics 8238

Examples: PRINQUAL Procedure **8239**
- Example 99.1: Multidimensional Preference Analysis of Automobile Data 8239
- Example 99.2: Principal Components of Basketball Rankings 8247

References **8256**

Overview: PRINQUAL Procedure

The PRINQUAL procedure performs principal component analysis (PCA) of qualitative, quantitative, or mixed data. PROC PRINQUAL is based on the work of Kruskal and Shepard (1974); Young, Takane, and de Leeuw (1978); Young (1981); Winsberg and Ramsay (1983). PROC PRINQUAL finds linear and nonlinear transformations of variables, using the method of alternating least squares, that optimize properties of the transformed variables' correlation or covariance matrix. Nonoptimal transformations such as logarithm and rank are also available. You can use ODS Graphics to display the results. You can use PROC PRINQUAL to do the following:

- fit metric and nonmetric principal component analyses
- perform metric and nonmetric multidimensional preference (MDPREF) analyses (Carroll 1972)
- transform data prior to their use in other analyses
- reduce the number of variables for subsequent use in regression analyses, cluster analyses, and other analyses
- detect nonlinear relationships

PROC PRINQUAL provides three methods, each of which seeks to optimize a different property of the transformed variables' covariance or correlation matrix. These methods are as follows:

- maximum total variance, or MTV
- minimum generalized variance, or MGV
- maximum average correlation, or MAC

The MTV method is based on a PCA model, and it is the most commonly used method. All three methods attempt to find transformations that decrease the rank of the covariance matrix computed from the transformed variables. Transforming the variables to maximize the total variance accounted for by a few linear combinations locates the observations in a space with a dimensionality that approximates the stated number of linear combinations as much as possible, given the transformation constraints. Transforming the variables to minimize their generalized variance or maximize the average correlations also reduces the dimensionality, but without a stated target for the final dimensionality. See the section "The Three Methods of Variable Transformation" on page 8223 for more information about all three methods.

The data can contain variables measured on nominal, ordinal, interval, and ratio scales of measurement (Siegel 1956). Any mix is allowed with all methods. PROC PRINQUAL can do the following:

- transform nominal variables by optimally scoring the categories (Fisher 1938)
- transform ordinal variables monotonically by scoring the ordered categories so that order is weakly preserved (adjacent categories can be merged) and the covariance matrix is optimized. You can undo ties optimally or leave them tied (Kruskal 1964). You can also transform ordinal variables to ranks.

- transform interval and ratio scale of measurement variables linearly, or transform them nonlinearly with spline transformations (De Boor 1978; Van Rijckevorsel 1982) or monotone spline transformations (Winsberg and Ramsay 1983). In addition, nonoptimal transformations for logarithm, rank, exponential, power, logit, and inverse trigonometric sine are available.
- estimate missing data without constraint, with category constraints (missing values within the same group get the same value), and with order constraints (missing value estimates in adjacent groups can be tied to preserve a specified ordering). See Gifi (1990) and Young (1981).

The transformed qualitative (nominal and ordinal) variables can be thought of as being quantified by the analysis, with the quantification done in the context set by the algorithm. The data are quantified so that the proportion of variance accounted for by a stated number of principal components is locally maximized, the generalized variance of the variables is locally minimized, or the average of the correlations is locally maximized.

The PROC PRINQUAL iterations produce a set of transformed variables. Each variable's new scoring satisfies a set of constraints based on the original scoring of the variable and the specified transformation type. First, all variables are required to satisfy standardization constraints; that is, all variables have a fixed mean and variance. The other constraints include linear constraints, weak order constraints, category constraints, and smoothness constraints. The new set of scores is selected from the sets of possible scorings that do not violate the constraints so that the method criterion is locally optimized.

Getting Started: PRINQUAL Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/prqgs.sas.)

PROC PRINQUAL can be used to fit a principal component model with nonlinear transformations of the variables and graphically display the results. This example finds monotonic transformations of ratings of automobiles.

```
title 'Ratings for Automobiles Manufactured in 1980';

data cars;
   input Origin $ 1-8 Make $ 10-19 Model $ 21-36
         (MPG Reliability Acceleration Braking Handling Ride
          Visibility Comfort Quiet Cargo) (1.);
   datalines;
GMC      Buick      Century         3334444544
GMC      Buick      Electra         2434453555
GMC      Buick      Lesabre         2354353545
GMC      Buick      Regal           3244443424

   ... more lines ...

GMC      Pontiac    Sunbird         3134533234
;
```

```
ods graphics on;

proc prinqual data=cars plots=all maxiter=100;
   transform monotone(mpg -- cargo);
   id model;
run;
```

The PROC PRINQUAL statement names the input data set Cars. The ODS GRAPHICS statement, along with the PLOTS=ALL option, requests all graphical displays. The PLOTS=ALL option implies the MDPREF option and requests a PCA plot with the scores (automobiles) represented as points and the structure (variables) represented as vectors. By default, the vector lengths are increased by a factor of 2.5 to produce a better graphical display. If instead you were to specify MDPREF=1, you would get the actual vectors, and they would all be short and would end near the origin where there are a lot of points. It is often the case that increasing the vector lengths by a factor of 2 or 3 makes a better graphical display, so by default the vector lengths are increased by a factor of 2.5. Up to 100 iterations are requested with the MAXITER= option. All of the numeric variable are specified with a MONOTONE transformation, so their original values, 1 to 5, are optimally rescored to maximize fit to a two-component model while preserving the original order. The Model variable provides the labels for the row points in the plot.

The iteration history table is shown in Figure 99.1. The monotonic transformations allow the PCA to account for 5% more variance in two principal components than the ordinary PCA model applied to the untransformed data.

Figure 99.1 Automobile Ratings Iteration History

Ratings for Automobiles Manufactured in 1980

The PRINQUAL Procedure

PRINQUAL MTV Algorithm Iteration History

Iteration Number	Average Change	Maximum Change	Proportion of Variance	Criterion Change	Note
1	0.18087	1.24219	0.53742		
2	0.06916	0.77503	0.57244	0.03502	
3	0.04653	0.38237	0.57978	0.00734	
4	0.03387	0.18682	0.58300	0.00321	
5	0.02661	0.13506	0.58484	0.00185	
6	0.01730	0.09213	0.58600	0.00115	
7	0.00969	0.07107	0.58660	0.00061	
8	0.00705	0.04798	0.58685	0.00025	
9	0.00544	0.03482	0.58699	0.00014	
10	0.00442	0.02641	0.58708	0.00009	
11	0.00363	0.02062	0.58714	0.00006	
12	0.00298	0.01643	0.58717	0.00004	
13	0.00245	0.01325	0.58720	0.00002	
14	0.00201	0.01077	0.58721	0.00002	
15	0.00165	0.00880	0.58723	0.00001	
16	0.00136	0.00721	0.58723	0.00001	
17	0.00112	0.00591	0.58724	0.00001	
18	0.00092	0.00485	0.58724	0.00000	
19	0.00075	0.00399	0.58724	0.00000	
20	0.00062	0.00328	0.58725	0.00000	
21	0.00051	0.00269	0.58725	0.00000	
22	0.00042	0.00221	0.58725	0.00000	
23	0.00035	0.00182	0.58725	0.00000	
24	0.00028	0.00149	0.58725	0.00000	
25	0.00023	0.00123	0.58725	0.00000	
26	0.00019	0.00101	0.58725	0.00000	
27	0.00016	0.00083	0.58725	0.00000	
28	0.00013	0.00068	0.58725	0.00000	
29	0.00011	0.00056	0.58725	0.00000	
30	0.00009	0.00046	0.58725	0.00000	
31	0.00007	0.00038	0.58725	0.00000	
32	0.00006	0.00031	0.58725	0.00000	
33	0.00005	0.00025	0.58725	0.00000	
34	0.00004	0.00021	0.58725	0.00000	
35	0.00003	0.00017	0.58725	0.00000	
36	0.00003	0.00014	0.58725	0.00000	
37	0.00002	0.00012	0.58725	0.00000	
38	0.00002	0.00010	0.58725	0.00000	
39	0.00001	0.00008	0.58725	0.00000	
40	0.00001	0.00006	0.58725	0.00000	
41	0.00001	0.00005	0.58725	0.00000	
42	0.00001	0.00004	0.58725	0.00000	Converged

Figure 99.1 *continued*

Algorithm converged.

The PCA biplot in Figure 99.2 shows the transformed automobile ratings projected into the two-dimensional plane of the analysis. The automobiles on the left tend to be smaller than the autos on the right, and the autos at the top tend to be cheaper than the autos at the bottom. The vectors can help you interpret the plot of the scores. Longer vectors show the variables that better fit the two-dimensional model. A larger component of them is in the plane of the plot. In contrast, shorter vectors show the variables that do not fit the two-dimensional model as well. They tend to be located less in the plot and more away from the plot; hence their projection into the plot is shorter. To envision this, lay a pencil on your desk directly under a light, and slowly rotate it up to form a 90-degree angle with your desk. As you do so, the shadow or projection of the pencil onto your desk will get progressively shorter. The results show, for example, that the Chevette would be expected to do well on gas mileage but not well on quiet and acceleration. In contrast, the Corvette and the Firebird have the opposite pattern.

Figure 99.2 Automobile Ratings PCA Biplot

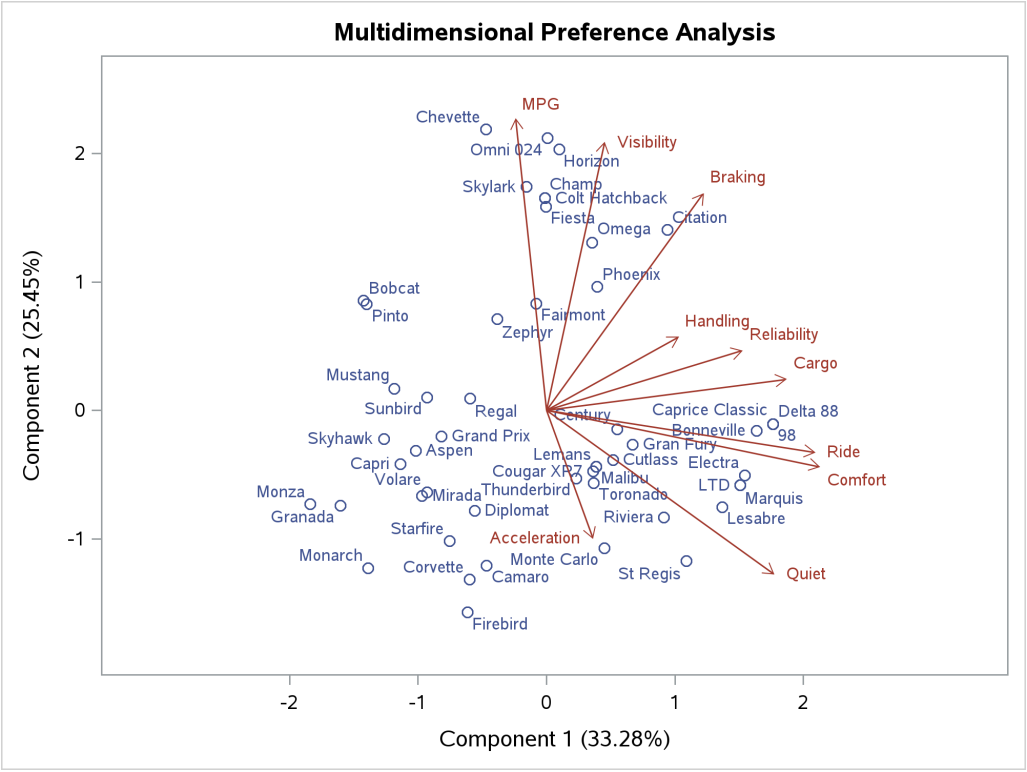

There are many patterns shown in the transformations in Figure 99.3. The transformation of Braking, for example, is not very different from the original scoring. The optimal scoring for other variables, such as Acceleration and Handling, is binary. Automobiles are differentiated by high versus everything else or low versus everything else.

Figure 99.3 Automobile Ratings Transformations

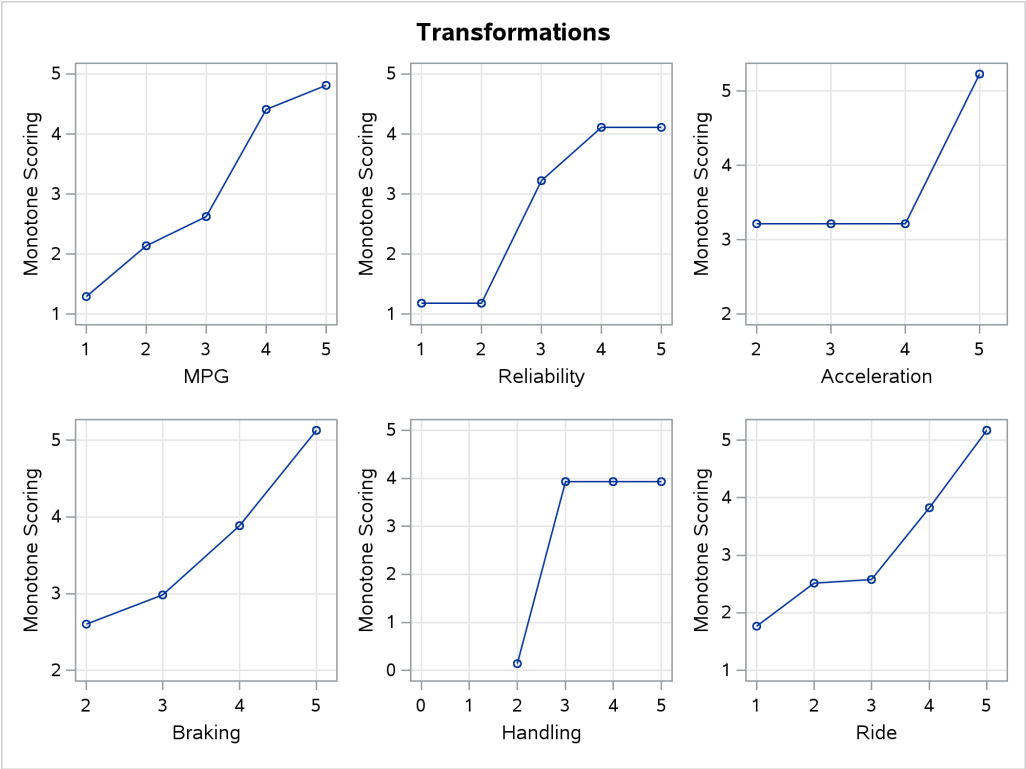

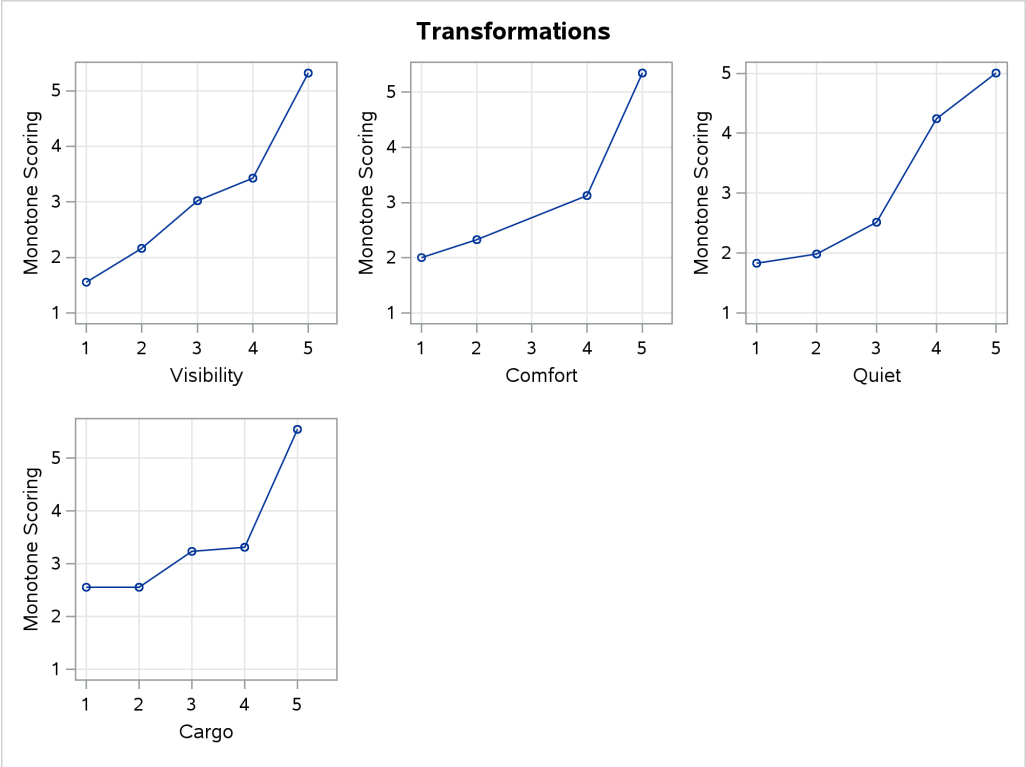

Syntax: PRINQUAL Procedure

The following statements are available in the PRINQUAL procedure:

```
PROC PRINQUAL < options > ;
    TRANSFORM   transform(variables < / t-options >)
                < transform(variables < / t-options >). . . > ;
    ID variables ;
    FREQ variable ;
    WEIGHT variable ;
    BY variables ;
```

To use PROC PRINQUAL, you need the PROC PRINQUAL and TRANSFORM statements. You can abbreviate all *options* and *t-options* to their first three letters. This is a special feature of PROC PRINQUAL and is not generally true of other SAS/STAT procedures.

The rest of this section provides detailed syntax information about each of the preceding statements, beginning with the PROC PRINQUAL statement. The remaining statements are described in alphabetical order.

PROC PRINQUAL Statement

```
PROC PRINQUAL < options > ;
```

The PROC PRINQUAL statement invokes the PRINQUAL procedure. Optionally, this statement identifies an input data set, creates an output data set, specifies the algorithm and other computational details, and controls displayed output. Table 99.1 summarizes the options available in the PROC PRINQUAL statement.

Table 99.1 Summary of PROC PRINQUAL Statement Options

Option	Description
Input Data Set Options	
DATA=	Specifies input SAS data set
Output Data Set Details	
APPROXIMATIONS	Outputs approximations to transformed variables
APREFIX=	Specifies prefix for approximation variables
CORRELATIONS	Outputs correlations and component structure matrix
MDPREF=	Specifies a multidimensional preference analysis
OUT=	Specifies output data set
PREFIX=	Specifies prefix for principal component scores
REPLACE	Replaces raw data with transformed data
SCORES	Outputs principal component scores
STANDARD	Standardizes principal component scores
TPREFIX=	Specifies prefix for transformed variables
TSTANDARD=	Specifies transformation standardization
Method and Iterations	
CCONVERGE=	Specifies minimum criterion change

Table 99.1 *continued*

Option	Description
CHANGE=	Specifies number of first iteration to be displayed
CONVERGE=	Specifies minimum data change
COVARIANCE	Analyzes covariances
DUMMY	Initializes using dummy variables
INITITER=	Specifies number of MAC initialization iterations
MAXITER=	Specifies maximum number of iterations
METHOD=	Specifies iterative algorithm
NOCHECK	Suppresses numerical error checking
N=	Specifies number of principal components
REFRESH=	Specifies number of MGV models before refreshing
REITERATE	Restarts iterations
SINGULAR=	Specifies singularity criterion
TYPE=	Specifies input observation type
Missing Data Handling	
MONOTONE=	Includes monotone special missing values
NOMISS	Excludes observations with missing values
UNTIE=	Unties special missing values
Control Displayed Output	
NOPRINT	Suppresses displayed output
PLOTS=	Specifies ODS Graphics details

The following list describes these options in alphabetical order.

APREFIX=*name*

APR=*name*

specifies a prefix for naming the approximation variables. By default, APREFIX=A. Specifying the APREFIX= option also implies the APPROXIMATIONS option.

APPROXIMATIONS

APPROX

APP

includes principal component approximations to the transformed variables (Eckart and Young 1936) in the output data set. Variable names are constructed from the value of the APREFIX= option and the input variable names. If you specify the APREFIX= option, then approximations are automatically included. If you specify the APPROXIMATIONS option and not the APREFIX= option, then the APPROXIMATIONS option uses the default, APREFIX=A, to construct the variable names.

CCONVERGE=*n*

CCO=*n*

specifies the minimum change in the criterion being optimized that is required to continue iterating. By default, CCONVERGE=0.0. The CCONVERGE= option is ignored for METHOD=MAC. For the MGV method, specify CCONVERGE=–2 to ensure data convergence.

CHANGE=*n*

CHA=*n*

specifies the number of the first iteration to be displayed in the iteration history table. The default is CHANGE=1. When you specify a larger value for *n*, the first $n - 1$ iterations are not displayed, thus speeding up the analysis. The CHANGE= option is most useful with the MGV method, which is much slower than the other methods.

CONVERGE=*n*

CON=*n*

specifies the minimum average absolute change in standardized variable scores that is required to continue iterating. By default, CONVERGE=0.00001. Average change is computed over only those variables that can be transformed by the iterations—that is, all LINEAR, OPSCORE, MONOTONE, UNTIE, SPLINE, MSPLINE, and SSPLINE variables and nonoptimal transformation variables with missing values. For more information, see the section "Optimal Transformations" on page 8218.

COVARIANCE

COV

computes the principal components from the covariance matrix. The variables are always centered to mean zero. If you do not specify the COVARIANCE option, the variables are also standardized to variance one, which means the analysis is based on the correlation matrix.

CORRELATIONS

COR

includes correlations and the component structure matrix in the output data set. By default, this information is not included.

DATA=*SAS-data-set*

specifies the SAS data set to be analyzed. The data set must be an ordinary SAS data set; it cannot be a TYPE=CORR or TYPE=COV data set. If you omit the DATA= option, PROC PRINQUAL uses the most recently created SAS data set.

DUMMY

DUM

expands variables specified for OPSCORE optimal transformations to dummy variables for the initialization (Tenenhaus and Vachette 1977). By default, the initial values of OPSCORE variables are the actual data values. The dummy variable nominal initialization requires considerable time and memory, so it might not be possible to use the DUMMY option with large data sets. No separate report of the initialization is produced. Initialization results are incorporated into the first iteration displayed in the iteration history table. For details, see the section "Optimal Transformations" on page 8218.

INITITER=*n*

INI=*n*

specifies the number of MAC iterations required to initialize the data before starting MTV or MGV iterations. By default, INITITER=0. The INITITER= option is ignored if METHOD=MAC.

MAXITER=*n*

MAX=*n*

specifies the maximum number of iterations. By default, MAXITER=30.

MDPREF<=*n***>**

MDP<=*n***>**

specifies a multidimensional preference analysis by implying the STANDARD, SCORES, and CORRELATIONS options. This option also suppresses warnings when there are more variables than observations.

When ODS Graphics is enabled, an MDPREF plot is produced with points for each row and vectors for each column. Often, the vectors are short, and a better graphical display is produced when the vectors are stretched. The absolute lengths of each vector can optionally be changed by specifying MDPREF=*n*. Then the vector coordinates are all multiplied by *n*. Usually, *n* will be a value such as 2, 2.5, or 3. The default is 2.5. Specify MDPREF=1 to see the vectors without any stretching. The relative lengths of the different vectors is important and interpretable, and these are preserved by the stretching.

METHOD=MAC | MGV | MTV

MET=MAC | MGV | MTV

specifies the optimization method. By default, METHOD=MTV. Values of the METHOD= option are MTV, for maximum total variance; MGV, for minimum generalized variance; and MAC, for maximum average correlation. You can use the MAC method when all variables are positively correlated or when no monotonicity constraints are placed on any transformations. See the section "The Three Methods of Variable Transformation" on page 8223 for more information.

MONOTONE=*two-letters*

MON=*two-letters*

specifies the first and last special missing value in the list of those special missing values to be estimated using within-variable order and category constraints. By default, there are no order constraints on missing value estimates. The *two-letters* value must consist of two letters in alphabetical order. For example, MONOTONE=DF means that the estimate of .D must be less than or equal to the estimate of .E, which must be less than or equal to the estimate of .F; no order constraints are placed on estimates of ._, .A through .C, and .G through .Z. For details, see the sections "Missing Values" on page 8229 and "Optimal Scaling" on page 10691 in Chapter 126, "The TRANSREG Procedure."

N=*n*

specifies the number of principal components to be computed. By default, N=2.

NOCHECK

NOC

turns off computationally intensive numerical error checking for the MGV method. If you do not specify the NOCHECK option, the procedure computes R square from the squared length of the predicted values vector and compares this value to the R square computed from the error sum of squares that is a byproduct of the sweep algorithm (Goodnight 1978). If the two values of R square differ by more than the square root of the value of the SINGULAR= option, a warning is displayed, the value of the REFRESH= option is halved, and the model is refit after refreshing. Specifying the NOCHECK option slightly speeds up the algorithm. Note that other less computationally intensive error checking is always performed.

NOMISS

NOM

excludes all observations with missing values from the analysis, but does not exclude them from the OUT= data set. If you omit the NOMISS option, PROC PRINQUAL simultaneously computes the optimal transformations of the nonmissing values and estimates the missing values that minimize squared error.

Casewise deletion of observations with missing values occurs when you specify the NOMISS option, when there are missing values in IDENTITY variables, when there are weights less than or equal to 0, or when there are frequencies less than 1. Excluded observations are output with a blank value for the _TYPE_ variable, and they have a weight of 0. They do not contribute to the analysis but are scored and transformed as *supplementary* or passive observations. See the sections "Passive Observations" on page 8235 and "Missing Values" on page 8229 for more information about excluded observations and missing data.

NOPRINT

NOP

suppresses the display of all output. This option disables the Output Delivery System (ODS), including ODS Graphics, for the duration of the procedure. For more information, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*

specifies an output SAS data set that contains results of the analysis. If you omit the OUT= option, PROC PRINQUAL still creates an output data set and names it by using the DATA*n* convention. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. You can use the REPLACE, APPROXIMATIONS, SCORES, and CORRELATIONS options to control what information is included in the output data set. For details, see the section "Output Data Set" on page 8231.

PLOTS < (*global-plot-options*) > < = *plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < = (*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses from around the plot request. Here are some examples:

```
plots=none
plots=transformation
plots(unpack)=transformation
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc prinqual plots=all;
   transformation spline(x1-x10);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled, but do not specify the PLOTS= option, then PROC PRINQUAL produces an MDPREF plot when the MDPREF option is specified.

The global plot options include the following:

FLIP

FLI

flips or interchanges the X-axis and Y-axis dimensions for MDPREF plots. The FLIP option can be specified either as a global plot option (for example, PLOTS(FLIP)) or with the MDPREF option (for example, PLOTS=MDPREF(FLIP)).

INTERPOLATE

INT

uses observations that are excluded from the analysis for interpolation in the fit and transformation plots. By default, observations with zero weight are excluded from all plots. These include observations with a zero, negative, or missing weight or frequency and observations excluded due to missing and invalid values. You can specify PLOTS(INTERPOLATE)=(*plot-requests*) to include some of these observations in the plots. You can use this option, for example, with sparse data sets to show smoother functions over the range of the data (see the section "The PLOTS(INTERPOLATE) Option" on page 10731 in Chapter 126, "The TRANSREG Procedure").

ONLY

ONL

suppresses the default plots. Only plots specifically requested are displayed.

UNPACKPANEL

UNPACK

UNP

suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACKPANEL to get each plot in a separate panel.

The plot requests include the following:

ALL

produces all appropriate plots.

TRANSFORMATION

TRA

TRANSFORMATION(UNPACK)

TRA(UNP)

plots the variable transformations. By default, multiple plots can appear in some output panels. Specify UNPACKPANEL to display each plot in a separate panel.

MDPREF

MDP

plots multidimensional preference analysis results. The MDPREF plot can also be requested by specifying the MDPREF option in the PROC PRINQUAL statement outside the PLOTS= option.

NONE

suppresses all plots.

PREFIX=*name*

PRE=*name*

specifies a prefix for naming the principal components. By default, PREFIX=Prin. As a result, the principal component default names are Prin1, Prin2,. . . , Prin*n*.

REFRESH=*n*

REF=*n*

specifies the number of variables to scale in the MGV method before computing a new inverse. By default, REFRESH=5. PROC PRINQUAL uses the REFRESH= option in the sweep algorithm of the MGV method. Large values for the REFRESH= option make the method run faster but with more numerical error. Small values make the method run more slowly but with more numerical accuracy.

REITERATE

REI

enables PROC PRINQUAL to use previous transformations as starting points. The REITERATE option affects only variables that are iteratively transformed (specified as LINEAR, SPLINE, MSPLINE, SSPLINE, UNTIE, OPSCORE, and MONOTONE). For iterative transformations, the REITERATE option requests a search in the input data set for a variable that consists of the value of the TPREFIX= option followed by the original variable name. If such a variable is found, it is used to provide the initial values for the first iteration. The final transformation is a member of the transformation family defined by the original variable, not the transformation family defined by the initialization variable. See the section "REITERATE Option Usage" on page 8234.

REPLACE

REP

replaces the original data with the transformed data in the output data set. The names of the transformed variables in the output data set correspond to the names of the original variables in the input data set. If you do not specify the REPLACE option, both original variables and transformed variables (with names constructed from the TPREFIX= option and the original variable names) are included in the output data set.

SCORES

SCO

includes principal component scores in the output data set. By default, scores are not included.

SINGULAR=*n*

SIN=*n*

specifies the largest value within rounding error of zero. By default, SINGULAR=1E–8. PROC PRINQUAL uses the value of the SINGULAR= option for checking $(1 - \mathrm{R}^2)$ when constructing full-rank matrices of predictor variables, checking denominators before dividing, and so on.

STANDARD

STD

standardizes the principal component scores in the output data set to mean zero and variance one instead of the default mean zero and variance equal to the corresponding eigenvalue. See the SCORES option.

TPREFIX=*name*

TPR=*name*

specifies a prefix for naming the transformed variables. By default, TPREFIX=T. The TPREFIX= option is ignored if you specify the REPLACE option.

TSTANDARD=CENTER | NOMISS | ORIGINAL | Z

TST=CEN | NOM | ORI | Z

specifies the standardization of the transformed variables in the OUT= data set. By default, TSTANDARD=ORIGINAL. When you specify the TSTANDARD= option in the PROC PRINQUAL statement, it the default standardization for all variables. When you specify TSTANDARD= as a *t-option*, it overrides the default standardization just for selected variables.

CENTER centers the output variables to mean zero, but the variances are the same as the variances of the input variables.

NOMISS sets the means and variances of the transformed variables in the OUT= data set, computed over all output values that correspond to nonmissing values in the input data set, to the means and variances computed from the nonmissing observations of the original variables. The TSTANDARD=NOMISS specification is useful with missing data. When a variable is linearly transformed, the final variable contains the original nonmissing values and the missing value estimates. In other words, the nonmissing values are unchanged. If your data have no missing values, TSTANDARD=NOMISS and TSTANDARD=ORIGINAL produce the same results.

ORIGINAL sets the means and variances of the transformed variables to the means and variances of the original variables. This is the default.

Z standardizes the variables to mean zero, variance one.

For nonoptimal variable transformations, the means and variances of the original variables are actually the means and variances of the nonlinearly transformed variables, unless you specify the ORIGINAL nonoptimal *t-option* in the TRANSFORM statement. For example, if a variable X with no missing values is specified as LOG, then, by default, the final transformation of X is simply LOG(X), not LOG(X) standardized to the mean of X and variance of X.

TYPE='*text* '|*name*

TYP='*text* '|*name*

specifies the valid value for the _TYPE_ variable in the input data set. If PROC PRINQUAL finds an input _TYPE_ variable, it uses only observations with a _TYPE_ value that matches the TYPE= value. This enables a PROC PRINQUAL OUT= data set containing correlations to be used as input to PROC PRINQUAL without requiring a WHERE statement to exclude the correlations. If a _TYPE_ variable is not in the data set, all observations are used. The default is TYPE='SCORE', so if you do not specify the TYPE= option, only observations with _TYPE_ = 'SCORE' are used.

PROC PRINQUAL displays a note when it reads observations with blank values of _TYPE_, but it does not automatically exclude those observations. Data sets created by the TRANSREG and PRINQUAL procedures have blank _TYPE_ values for those observations that were excluded from the analysis due to nonpositive weights, nonpositive frequencies, or missing data. When these observations are read again, they are excluded for the same reason that they were excluded from their original analysis, not because their _TYPE_ value is blank.

UNTIE=*two-letters*

UNT=*two-letters*

specifies the first and last special missing values in the list of those special missing values that are to be estimated with within-variable order constraints but no category constraints. The *two-letters* value must consist of two letters in alphabetical order. By default, there are category constraints but no order constraints on special missing value estimates. For details, see the section "Missing Values" on page 8229. Also, see the section "Optimal Scaling" on page 10691 in Chapter 126, "The TRANSREG Procedure."

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC PRINQUAL to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the PRINQUAL procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If one variable in the input data set represents the frequency of occurrence for other values in the observation, list the variable's name in a FREQ statement. PROC PRINQUAL then treats the data set as if each observation appeared *n* times, where *n* is the value of the FREQ variable for the observation. Noninteger values of the FREQ variable are truncated to the largest integer less than the FREQ value. The observation is used in the analysis only if the value of the FREQ statement variable is greater than or equal to 1.

ID Statement

ID *variables* **;**

The ID statement includes additional character or numeric variables in the output data set. The variables must be contained in the input data set.

TRANSFORM Statement

TRANSFORM *transform*(*variables* < / *t-options* >)< *transform*(*variables* < / *t-options* >)... > **;**

The TRANSFORM statement lists the variables to be analyzed (*variables*) and specifies the transformation (*transform*) to apply to each variable listed. You must specify a transformation for each variable list in the TRANSFORM statement. The variables are variables in the data set. The *t-options* are transformation options that provide details for the transformation; these depend on the *transform* chosen. The *t-options* are listed after a slash in the parentheses that enclose the variables.

For example, the following statements find a quadratic polynomial transformation of all variables in the data set:

```
proc prinqual;
   transform spline(_all_ / degree=2);
run;
```

Or, if N1 through N10 are nominal variables and M1 through M10 are ordinal variables, you can use the following statements:

```
proc prinqual;
   transform opscore(N1-N10) monotone(M1-M10);
run;
```

The following sections describe the transformations available (specified with *transform*) and the options available for some of the transformations (specified with *t-options*).

Families of Transformations

There are three types of transformation families: nonoptimal, optimal, and other. The families are described as follows:

Nonoptimal transformations preprocess the specified variables, replacing each one with a single new nonoptimal, nonlinear transformation.

Optimal transformations replace the specified variables with new, iteratively derived optimal transformation variables that fit the specified model better than the original variable (except in contrived cases where the transformation fits the model exactly as well as the original variable).

Other transformations are the IDENTITY and SSPLINE transformations. These do not fit into either of the preceding categories.

Table 99.2 summarizes the transformations in each family.

Table 99.2 Transformation Families

Transformation	Description
Nonoptimal Transformations	
ARSIN	Inverse trigonometric sine
EXP	Exponential
LOG	Logarithm
LOGIT	Logit
POWER	Raises variables to specified power
RANK	Transforms to ranks
Optimal Transformations	
LINEAR	Linear
MONOTONE	Monotonic, ties preserved
MSPLINE	Monotonic B-spline
OPSCORE	Optimal scoring
SPLINE	B-spline
UNTIE	Monotonic, ties not preserved
Other Transformations	
IDENTITY	Identity, no transformation
SSPLINE	Iterative smoothing spline

The *transform* is followed by a variable (or list of variables) enclosed in parentheses. Optionally, depending on the *transform*, the parentheses can also contain *t-options*, which follow the variables and a slash. For example, the following statement computes the LOG transformation of X and Y:

```
transform log(X Y);
```

A more complex example follows:

```
transform spline(Y / nknots=2) log(X1 X2 X3);
```

The preceding statement uses the SPLINE transformation of the variable Y and the LOG transformation of the variables X1, X2, and X3. In addition, it uses the NKNOTS= option with the SPLINE transformation and specifies two knots.

The rest of this section provides syntax details for members of the three families of transformations. The *t-options* are discussed in the section "Transformation Options (t-options)" on page 8219.

Nonoptimal Transformations

Nonoptimal transformations are computed before the iterative algorithm begins. Nonoptimal transformations create a single new transformed variable that replaces the original variable. The new variable is not transformed by the subsequent iterative algorithms (except for a possible linear transformation and missing value estimation).

The following list provides syntax and details for nonoptimal variable transformations.

ARSIN

ARS

finds an inverse trigonometric sine transformation. Variables specified in the ARSIN *transform* must be numeric and in the interval $(-1.0 \leq x \leq 1.0)$, and they are typically continuous.

EXP

exponentiates variables (x is transformed to a^x). To specify the value of a, use the PARAMETER= *t-option*. By default, a is the mathematical constant $e = 2.718\ldots$. Variables specified with the EXP *transform* must be numeric, and they are typically continuous.

LOG

transforms variables to logarithms (x is transformed to $\log_a(x)$). To specify the base of the logarithm, use the PARAMETER= *t-option*. The default is a natural logarithm with base $e = 2.718\ldots$. Variables specified with the LOG *transform* must be numeric and positive, and they are typically continuous.

LOGIT

finds a logit transformation on the variables. The logit of x is $\log(x/(1-x))$. Unlike other transformations, LOGIT does not have a three-letter abbreviation. Variables specified with the LOGIT *transform* must be numeric and in the interval $(0.0 < x < 1.0)$, and they are typically continuous.

POWER

POW

raises variables to a specified power (x is transformed to x^a). You must specify the power parameter a by specifying the PARAMETER= *t-option* following the variables.

```
power(variable / parameter=number)
```

You can use POWER for squaring variables (PARAMETER=2), reciprocal transformations (PARAMETER=–1), square roots (PARAMETER=0.5), and so on. Variables specified with the POWER *transform* must be numeric, and they are typically continuous.

RANK

RAN

transforms variables to ranks. Ranks are averaged within ties. The smallest input value is assigned the smallest rank. Variables specified with the RANK *transform* must be numeric.

Optimal Transformations

Optimal transformations are iteratively derived. Missing values for these types of variables can be optimally estimated (see the section "Missing Values" on page 8229). See the sections "OPSCORE, MONOTONE, UNTIE, and LINEAR Transformations" on page 10692 and "SPLINE and MSPLINE Transformations" on page 10693 in Chapter 126, "The TRANSREG Procedure," for more information about the optimal transformations.

The following list provides syntax and details for optimal transformations.

LINEAR

LIN

finds an optimal linear transformation of each variable. For variables with no missing values, the transformed variable is the same as the original variable. For variables with missing values, the transformed nonmissing values have a different scale and origin than the original values. Variables specified with the LINEAR *transform* must be numeric.

MONOTONE

MON

finds a monotonic transformation of each variable, with the restriction that ties are preserved. The Kruskal (1964) secondary least squares monotonic transformation is used. This transformation weakly preserves order and category membership (ties). Variables specified with the MONOTONE *transform* must be numeric, and they are typically discrete.

MSPLINE

MSP

finds a monotonically increasing B-spline transformation with monotonic coefficients (De Boor 1978; De Leeuw 1986) of each variable. You can specify the DEGREE=, KNOTS=, NKNOTS=, and EVENLY= *t-options* with MSPLINE. By default, PROC PRINQUAL uses a quadratic spline. Variables specified with the MSPLINE *transform* must be numeric, and they are typically continuous.

OPSCORE

OPS

finds an optimal scoring of each variable. The OPSCORE transformation assigns scores to each class (level) of the variable. The Fisher (1938) optimal scoring method is used. Variables specified with the OPSCORE *transform* can be either character or numeric; numeric variables should be discrete.

SPLINE

SPL

finds a B-spline transformation (De Boor 1978) of each variable. By default, PROC PRINQUAL uses a cubic polynomial transformation. You can specify the DEGREE=, KNOTS=, NKNOTS=, and EVENLY *t-options* with SPLINE. Variables specified with the SPLINE *transform* must be numeric, and they are typically continuous.

UNTIE

UNT

finds a monotonic transformation of each variable without the restriction that ties are preserved. PROC PRINQUAL uses the Kruskal (1964) primary least squares monotonic transformation method. This transformation weakly preserves order but not category membership (it might untie some previously tied values). Variables specified with the UNTIE *transform* must be numeric, and they are typically discrete.

Other Transformations

IDENTITY

IDE

specifies variables that are not changed by the iterations. The IDENTITY transformation is used for variables when no transformation and no missing data estimation are desired. However, the REFLECT, ADDITIVE, TSTANDARD=Z, and TSTANDARD=CENTER options can linearly transform all variables, including IDENTITY variables, after the iterations. Observations with missing values in IDENTITY variables are excluded from the analysis, and no optimal scores are computed for missing values in IDENTITY variables. Variables specified with the IDENTITY *transform* must be numeric.

SSPLINE

SSP

finds an iterative smoothing spline transformation of each variable. The SSPLINE transformation does not generally minimize squared error. You can specify the smoothing parameter with either the SM= *t-option* or the PARAMETER= *t-option*. The default smoothing parameter is SM=0. Variables specified with the SSPLINE *transform* must be numeric, and they are typically continuous.

Transformation Options (t-options)

If you use a nonoptimal, optimal, or other transformation, you can use *t-options*, which specify additional details of the transformation. The *t-options* are specified within the parentheses that enclose variables and are listed after a slash. For example:

```
proc prinqual;
   transform spline(X Y / nknots=3);
run;
```

The preceding statements find an optimal variable transformation (SPLINE) of the variables X and Y and use a *t-option* to specify the number of knots (NKNOTS=). The following is a more complex example:

```
proc prinqual;
   transform spline(Y / nknots=3) spline(X1 X2 / nknots=6);
run;
```

These statements use the SPLINE transformation for all three variables and use *t-options* as well; the NKNOTS= option specifies the number of knots for the spline.

The following sections discuss the *t-options* available for nonoptimal, optimal, and other transformations.

Table 99.3 summarizes the *t-options*.

Table 99.3 Transformation Options

Option	Description
Nonoptimal Transformation	
ORIGINAL	Uses original mean and variance
Parameter Specification	
PARAMETER=	Specifies miscellaneous parameters
SM	Specifies smoothing parameter
Spline	
DEGREE=	Specifies the degree of the spline
EVENLY	Spaces the knots evenly
KNOTS=	Specifies the interior knots or break points
NKNOTS=	Creates *n* knots
Other t-options	
NAME=	Renames variables
REFLECT	Reflects the variable around the mean
TSTANDARD=	Specifies transformation standardization

Nonoptimal Transformation t-options

ORIGINAL

ORI

matches the variable's final mean and variance to the mean and variance of the original variable. By default, the mean and variance are based on the transformed values. The ORIGINAL *t-option* is available for all of the nonoptimal transformations.

Parameter t-options

PARAMETER=*number*

PAR=*number*

specifies the transformation parameter. The PARAMETER= *t-option* is available for the EXP, LOG, POWER, SMOOTH, and SSPLINE transformations. For EXP, the parameter is the value to be exponentiated; for LOG, the parameter is the base value; and for POWER, the parameter is the power. For SMOOTH and SSPLINE, the parameter is the raw smoothing parameter. (See the SM= option for an alternative way to specify the smoothing parameter.) The default for the PARAMETER= *t-option* for the LOG and EXP transformations is $e = 2.718\ldots$. The default parameter for SSPLINE is computed from SM=0. For the POWER transformation, you must specify the PARAMETER= *t-option*; there is no default.

SM=*n*

specifies a smoothing parameter in the range 0 to 100, just like PROC GPLOT uses. For example, SM=50 in PROC PRINQUAL is equivalent to I=SM50 on the SYMBOL statement with PROC GPLOT. You can specify the SM= *t-option* only with the SSPLINE transformation. The smoothness of the function increases as the value of the smoothing parameter increases. By default, SM=0.

Spline t-options

The following *t-options* are available with the SPLINE and MSPLINE optimal transformations.

DEGREE=*n*

DEG=*n*

specifies the degree of the B-spline transformation. The degree must be a nonnegative integer. The defaults are DEGREE=3 for SPLINE variables and DEGREE=2 for MSPLINE variables.

The polynomial degree should be a small integer, usually 0, 1, 2, or 3. Larger values are rarely useful. If you have any doubt as to what degree to specify, use the default.

EVENLY<=*n***>**

EVE<=*n***>**

is used with the NKNOTS= *t-option* to space the knots evenly. The differences between adjacent knots are constant. If you specify NKNOTS=*k*, *k* knots are created at

$$\text{minimum} + i((\text{maximum} - \text{minimum})/(k+1))$$

for $i = 1, \ldots, k$. For example, if you specify

```
spline(X / knots=2 evenly)
```

and the variable X has a minimum of 4 and a maximum of 10, then the two interior knots are 6 and 8. Without the EVENLY *t-option*, the NKNOTS= *t-option* places knots at percentiles, so the knots are not evenly spaced.

By default for the SPLINE and MSPLINE transformations, the smaller exterior knots are all the same and just a little smaller than the minimum. Similarly, by default, the larger exterior knots are all the same and just a little larger than the maximum. However, if you specify EVENLY=*n*, then the *n* exterior knots are evenly spaced as well. The number of exterior knots must be greater than or equal to the degree. You can specify values larger than the degree when you want to interpolate slightly beyond the range or your data. The exterior knots must be less than the minimum or greater than the maximum, and hence the knots across all sets are not precisely equally spaced. For example, with data ranging from 0 to 10, and with EVENLY=3 and NKNOTS=4, the first exterior knots are –4.000000000001, –2.000000000001, and –0.000000000001, the interior knots are 2, 4, 6, and 8, and the second exterior knots are 10.000000000001, 12.000000000001, and 14.000000000001.

KNOTS=*number-list* | *n* **TO** *m* **BY** *p*

KNO=*number-list* | *n* **TO** *m* **BY** *p*

specifies the interior knots or break points. By default, there are no knots. The first time you specify a value in the knot list, it indicates a discontinuity in the *n*th (from DEGREE=*n*) derivative of the transformation function at the value of the knot. The second mention of a value indicates a discontinuity in the (n – 1) derivative of the transformation function at the value of the knot. Knots can be repeated any number of times to decrease the smoothness at the break points, but the values in the knot list can never decrease.

You cannot use the KNOTS= *t-option* with the NKNOTS= *t-option*. You should keep the number of knots small. (See the section "Specifying the Number of Knots" on page 10694 in Chapter 126, "The TRANSREG Procedure.")

NKNOTS=*n*

NKN=*n*

creates *n* knots, the first at the $100/(n+1)$th percentile, the second at the $200/(n+1)$th percentile, and so on. Knots are always placed at data values; there is no interpolation. For example, if NKNOTS=3, knots are placed at the 25th percentile, the median, and the 75th percentile. By default, NKNOTS=0. The NKNOTS= *t-option* must be ≥ 0.

You cannot use the NKNOTS= *t-option* with the KNOTS= *t-option*. You should keep the number of knots small. (See the section "Specifying the Number of Knots" on page 10694 in Chapter 126, "The TRANSREG Procedure.")

Other t-options

The following *t-options* are available for all transformations.

NAME=(*variable-list***)**

NAM=(*variable-list***)**

renames variables as they are used in the TRANSFORM statement. This option allows a variable to be used more than once. For example, if the variable X is a character variable, then the following step stores both the original character variable X and a numeric variable XC that contains category numbers in the output data set.

```
proc prinqual data=A n=1 out=B;
   transform linear(Y) opscore(X / name=(XC));
   id X;
run;
```

REFLECT

REF

reflects the transformation

$$y = -(y - \bar{y}) + \bar{y}$$

after the iterations are completed and before the final standardization and results calculations.

TSTANDARD=CENTER | NOMISS | ORIGINAL | Z

TST=CEN | NOM | ORI | Z

specifies the standardization of the transformed variables in the OUT= data set. By default, TSTANDARD=ORIGINAL. When the TSTANDARD= option is specified in the PROC PRINQUAL statement, it specifies the default standardization for all variables. When you specify TSTANDARD= as a *t-option*, it overrides the default standardization only for selected variables.

WEIGHT Statement

WEIGHT *variable* **;**

When you use a WEIGHT statement, a weighted residual sum of squares is minimized. The WEIGHT statement has no effect on degrees of freedom or number of observations, but the weights affect most other calculations. The observation is used in the analysis only if the value of the WEIGHT statement variable is greater than 0.

Details: PRINQUAL Procedure

The Three Methods of Variable Transformation

The three methods of variable transformation provided by PROC PRINQUAL are discussed in the following sections.

The Maximum Total Variance (MTV) Method

The MTV method (Young, Takane, and de Leeuw 1978) is based on the principal component model, and it attempts to maximize the sum of the first r eigenvalues of the covariance matrix. This method transforms variables to be (in a least squares sense) as similar to linear combinations of r principal component score variables as possible, where r can be much smaller than the number of variables. This maximizes the total variance of the first r components (the trace of the covariance matrix of the first r principal components). See *SAS Technical Report R-108*.

On each iteration, the MTV algorithm alternates classical principal component analysis (Hotelling 1933) with optimal scaling (Young 1981). When all variables are ordinal preference ratings, this corresponds to MDPREF analysis (Carroll 1972). You can request the dummy variable initialization method suggested by Tenenhaus and Vachette (1977), who independently propose the same iterative algorithm for nominal and interval scale-of-measurement variables.

The Minimum Generalized Variance (MGV) Method

The MGV method (Sarle 1984) uses an iterated multiple regression algorithm in an attempt to minimize the determinant of the covariance matrix of the transformed variables. This method transforms each variable to be (in a least squares sense) as similar to linear combinations of the remaining variables as possible. This locally minimizes the generalized variance of the transformed variables, the determinant of the covariance matrix, the volume of the parallelepiped defined by the transformed variables, and the sphericity (the extent to which a quadratic form in the optimized covariance matrix defines a sphere). See *SAS Technical Report R-108*.

On each iteration for each variable, the MGV algorithm alternates multiple regression with optimal scaling. The multiple regression involves predicting the selected variable from all other variables. You can request a dummy variable initialization by using a modification of the Tenenhaus and Vachette (1977) method that is appropriate with a regression algorithm. This method can be viewed as a way of investigating the nature of the linear and nonlinear dependencies in, and the rank of, a data matrix containing variables that can be

nonlinearly transformed. This method tries to create a less-than-full-rank data matrix. The matrix contains the transformation of each variable that is most similar to what the other transformed variables predict.

The Maximum Average Correlation (MAC) Method

The MAC method (De Leeuw 1985) uses an iterated constrained multiple regression algorithm in an attempt to maximize the average of the elements of the correlation matrix. This method transforms each variable to be (in a least squares sense) as similar to the average of the remaining variables as possible.

On each iteration for each variable, the MAC algorithm alternates computing an equally weighted average of the other variables with optimal scaling. The MAC method is similar to the MGV method in that each variable is scaled to be as similar to a linear combination of the other variables as possible, given the constraints on the transformation. However, optimal weights are not computed. You can use the MAC method when all variables are positively correlated or when no monotonicity constraints are placed on any transformations. Do not use this method with negatively correlated variables when some optimal transformations are constrained to be increasing because the signs of the correlations are not taken into account. The MAC method is useful as an initialization method for the MTV and MGV methods.

Understanding How PROC PRINQUAL Works

In the following example, PROC PRINQUAL uses the MTV method to linearize a curved scatter plot. Let

$$
\begin{aligned}
X_1 &= -1 \text{ to } 1 \text{ by } 0.02 \\
X_2 &= X_1^3 + \epsilon \\
X_3 &= X_2^5 + \epsilon
\end{aligned}
$$

where ϵ is normal error.

These three variables define a curved swarm of points in three-dimensional space. First, the SGSCATTER procedure is used to display two-dimensional views of these data. Next, PROC PRINQUAL is used to straighten the scatter plot, making it more one-dimensional by finding a smooth transformation of each variable. The N=1 option in the PROC PRINQUAL statement requests one principal component. The TRANSFORM statement requests a cubic spline transformation with nine knots. *Splines* are curves, which are usually required to be continuous and smooth. See the section "Splines" on page 8228 for more information about splines. See Smith (1979) for an excellent introduction to splines.

PROC PRINQUAL transforms each variable to be as much as possible like the first principal component (or more generally, to be close to the space defined by the first N= principal components). One component accounts for 92 percent of the variance of the untransformed data and over 99 percent of the variance of the transformed data (see Figure 99.5). Note that the results did not converge in the default 50 iterations, so more iterations were requested using the MAXITER= option. The transformations are requested by specifying PLOTS=TRANSFORMATION and are displayed in Figure 99.6.

PROC PRINQUAL creates an output data set that contains both the original and transformed variables. The original variables are named X1, X2, and X3, and the transformed variables are named TX1, TX2, and TX3. The transformed variables are displayed using the SGSCATTER procedure in Figure 99.7.

The following statements produce Figure 99.4 through Figure 99.7:

```
ods graphics on;

* Generate Three-Dimensional Data;
data X;
   do X1 = -1 to 1 by 0.02;
      X2 = X1 ** 3 + 0.05 * normal(7);
      X3 = X1 ** 5 + 0.05 * normal(7);
      output;
   end;
run;

proc sgscatter data=x;
   plot x1*x2 x1*x3 x3*x2;
run;

* Try to Straighten the Scatter Plot;
proc prinqual data=X n=1 maxiter=2000 plots=transformation out=results;
   title 'Linearize the Scatter Plot';
   transform spline(X1-X3 / nknots=9);
run;


* Plot the Linearized Scatter Plot;
proc sgscatter data=results;
   plot tx1*tx2 tx1*tx3 tx3*tx2;
run;
```

The three-dimensional data in Figure 99.4 and Figure 99.7 are displayed in three two-dimensional plots, arrayed as if they were three faces of a cube that was flattened as you might flatten a box.

Figure 99.4 Three-Dimensional Scatter Plot

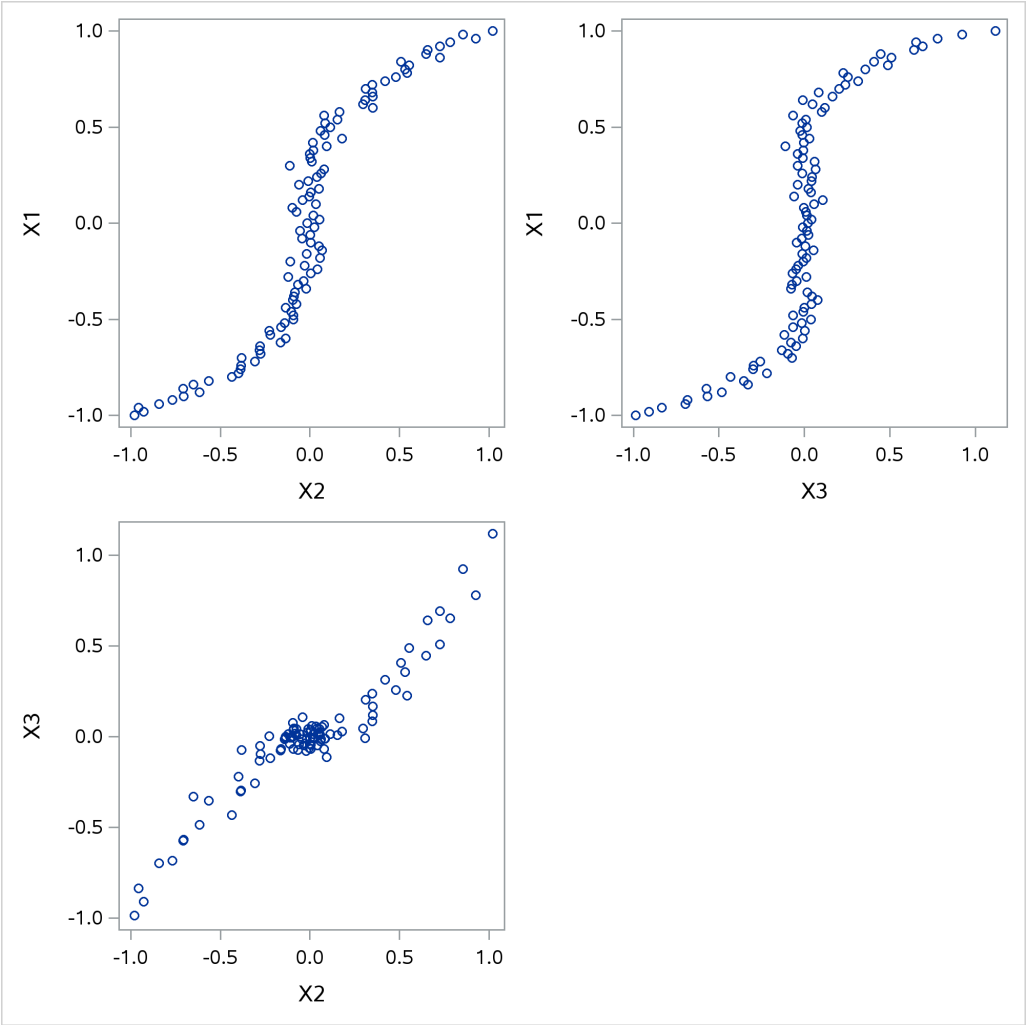

Figure 99.5 PRINQUAL Iteration History

Linearize the Scatter Plot

The PRINQUAL Procedure

PRINQUAL MTV Algorithm Iteration History

Iteration Number	Average Change	Maximum Change	Proportion of Variance	Criterion Change	Note
1	0.15125	0.93453	0.92376		
2	0.04589	0.14682	0.98030	0.05653	
3	0.03154	0.10125	0.98626	0.00596	
4	0.02258	0.06890	0.98890	0.00265	
5	0.01682	0.04777	0.99028	0.00137	
6	0.01297	0.03782	0.99106	0.00078	
7	0.01032	0.03029	0.99154	0.00048	
.					
.					
.					
1670	0.00001	0.00005	0.99371	0.00000	
1671	0.00001	0.00005	0.99371	0.00000	
1672	0.00001	0.00005	0.99371	0.00000	Converged

```
Algorithm converged.
```

Figure 99.6 Transformations

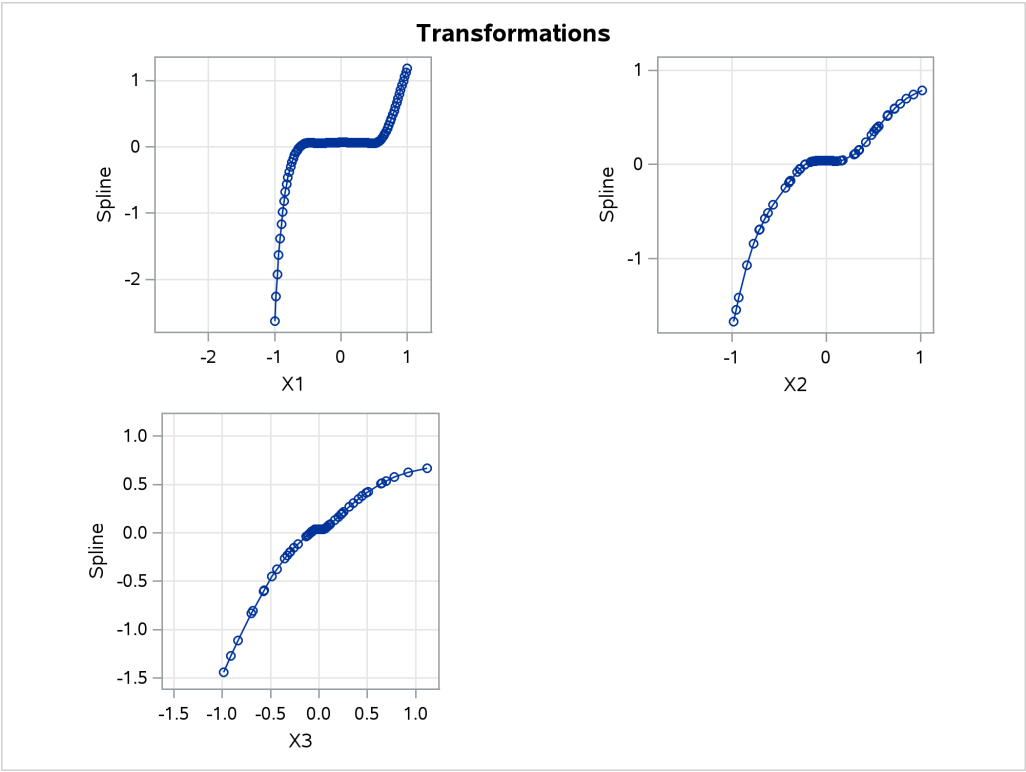

Figure 99.7 Linearized Scatter Plot

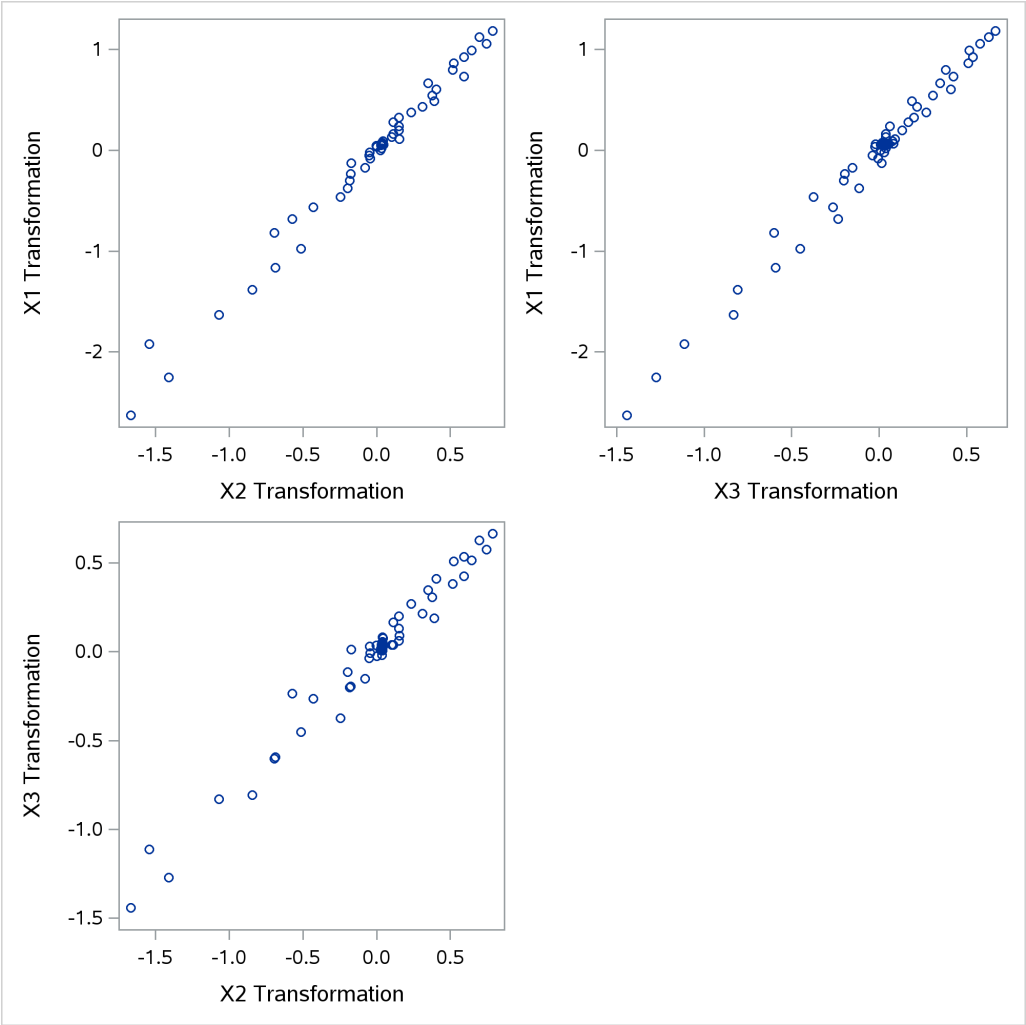

Splines

Splines are curves, and they are usually required to be continuous and smooth. Splines are usually defined as piecewise polynomials of degree n with function values and first $n - 1$ derivatives that agree at the points where they join. The abscissa values of the join points are called *knots*. The term "spline" is also used for polynomials (splines with no knots) and piecewise polynomials with more than one discontinuous derivative. Splines with no knots are generally smoother than splines with knots, which are generally smoother than splines with multiple discontinuous derivatives. Splines with few knots are generally smoother than splines with many knots; however, increasing the number of knots usually increases the fit of the spline function to the data. Knots give the curve freedom to bend to follow the data more closely. See Smith (1979) for an excellent introduction to splines. There are many examples and detailed discussions of splines in Chapter 126, "The TRANSREG Procedure." See the sections "Linear and Nonlinear Regression Functions" on page 10647, "Smoothing Splines" on page 10660, "SPLINE and MSPLINE Transformations" on page 10693, "Specifying the Number of Knots" on page 10694, "SPLINE, BSPLINE, and PSPLINE Comparisons" on page 10696,

"Linear and Nonlinear Regression Functions" on page 10647, "Simultaneously Fitting Two Regression Functions" on page 10651, and examples "Using Splines and Knots" on page 10632 and Example 126.1.

Missing Values

PROC PRINQUAL can estimate missing values, subject to optional constraints, so that the covariance matrix is optimized. The procedure provides several approaches for handling missing data. When you specify the NOMISS option in the PROC PRINQUAL statement, observations with missing values are excluded from the analysis. Otherwise, missing data are estimated, using variable means as initial estimates. Missing values for OPSCORE character variables are treated the same as any other category during the initialization. See the section "Missing Values" on page 10681 in Chapter 126, "The TRANSREG Procedure," for more information about missing data estimation.

Controlling the Number of Iterations

Several options in the PROC PRINQUAL statement control the number of iterations performed. Iteration terminates when any one of the following conditions is satisfied:

- The number of iterations equals the value of the MAXITER= option.
- The average absolute change in variable scores from one iteration to the next is less than the value of the CONVERGE= option.
- The criterion change is less than the value of the CCONVERGE= option.

With the MTV method, the change in the proportion of variance criterion can become negative when the data have converged so that it is numerically impossible, within machine precision, to increase the criterion. Because the MTV algorithm is convergent, a negative criterion change is the result of very small amounts of rounding error. The MGV method displays the average squared multiple correlation (which is not the criterion being optimized), so the criterion change can become negative well before convergence. The MAC method criterion (average correlation) is never computed, so the CCONVERGE= option is ignored for METHOD=MAC. You can specify a negative value for either convergence option if you want to define convergence only in terms of the other convergence option.

With the MGV method, iterations minimize the generalized variance (determinant), but the generalized variance is not reported for two reasons. First, in most data sets, the generalized variance is almost always near zero (or will be after one or two iterations), which is its minimum. This does not mean that iteration is complete; it simply means that at least one multiple correlation is at or near one. The algorithm continues minimizing the determinant in $(m-1)$, $(m-2)$ dimensions, and so on. Because the generalized variance is almost always near zero, it does not provide a good indication of how the iterations are progressing. The mean R square provides a better indication of convergence. The second reason for not reporting the generalized variance is that almost no additional time is required to compute R square values for each step. This is because the error sum of squares is a byproduct of the algorithm at each step. Computing the determinant at the end of each iteration adds more computations to an already computationally intensive algorithm.

You can increase the number of iterations to ensure convergence by increasing the value of the MAXITER= option and decreasing the value of the CONVERGE= option. Because the average absolute change in

standardized variable scores seldom decreases below 1E–11, you typically do not specify a value for the CONVERGE= option less than 1E–8 or 1E–10. Most of the data changes occur during the first few iterations, but the data can still change after 50 or even 100 iterations. You can try different combinations of values for the CONVERGE= and MAXITER= options to ensure convergence without extreme overiteration. If the data do not converge with the default specifications, specify the REITERATE option, or try CONVERGE=1E–8 and MAXITER=50, or CONVERGE=1E–10 and MAXITER=200.

Performing a Principal Component Analysis of Transformed Data

PROC PRINQUAL produces an iteration history table that displays (for each iteration) the iteration number, the maximum and average absolute change in standardized variable scores computed over the iteratively transformed variables, the criterion being optimized, and the criterion change. In order to examine the results of the analysis in more detail, you can analyze the information in the output data set by using other SAS procedures.

Specifically, use the PRINCOMP procedure to perform a principal components analysis on the transformed data. PROC PRINCOMP accepts the raw data from PROC PRINQUAL but issues a warning, because the PROC PRINQUAL output data set has _NAME_ and _TYPE_ variables but is not a TYPE=CORR data set. You can ignore this warning.

If the output data set contains both scores and correlations, you must subset it for analysis with PROC PRINCOMP. Otherwise, the correlation observations are treated as ordinary observations and the PROC PRINCOMP results are incorrect. For example, consider the following statements:

```
proc prinqual data=a out=b correlations replace;
   transform spline(var1-var50 / nknots=3);
run;

proc princomp data=b;
   where _TYPE_='SCORE';
run;
```

Also note that the proportion of variance accounted for, as reported by PROC PRINCOMP, can exceed the proportion of variance accounted for in the last PROC PRINQUAL iteration. This is because PROC PRINQUAL reports the variance accounted for by the components analysis that generated the current scaling of the data, not a components analysis of the current scaling of the data.

Using the MAC Method

You can use the MAC algorithm alone by specifying METHOD=MAC, or you can use it as an initialization algorithm for METHOD=MTV and METHOD=MGV analyses by specifying the iteration option INITITER=. If any variables are negatively correlated, do not use the MAC algorithm with monotonic transformations (MONOTONE, UNTIE, and MSPLINE) because the signs of the correlations among the variables are not used when computing variable approximations. If an approximation is negatively correlated with the original variable, monotone constraints would make the optimally scaled variable a constant, which is not allowed (see the section "Avoiding Constant Transformations" on page 8234). When used with other transformations, the MAC algorithm can reverse the scoring of the variables. So, for example, if variable X is designated LOG(X) with METHOD=MAC and TSTANDARD=ORIGINAL, the final transformation (for example, TX)

might not be LOG(X). If TX is not LOG(X), it has the same mean as LOG(X) and the same variance as LOG(X), and it is perfectly negatively correlated with LOG(X). PROC PRINQUAL displays a note for every variable that is reversed in this manner.

You can use the METHOD=MAC algorithm to reverse the scorings of some rating variables before a factor analysis. The correlations among bipolar ratings such as 'like - dislike', 'hot - cold', and 'fragile - monumental' are typically both positive and negative. If some items are reversed to say 'dislike - like', 'cold - hot', and 'monumental - fragile', some of the negative signs can be eliminated, and the factor pattern matrix would be cleaner. You can use PROC PRINQUAL with METHOD=MAC and LINEAR transformations to reverse some items, maximizing the average of the intercorrelations.

Output Data Set

PROC PRINQUAL produces an output data set by default. By specifying the OUT=, APPROXIMATIONS, SCORES, REPLACE, and CORRELATIONS options in the PROC PRINQUAL statement, you can name this data set and control its contents.

By default, the procedure creates an output data set that contains variables with _TYPE_='SCORE'. These observations contain original variables, transformed variables, components, or data approximations. If you specify the CORRELATIONS option in the PROC PRINQUAL statement, the data set also contains observations with _TYPE_='CORR'; these observations contain correlations or component structure information.

Structure and Content

The output data set can have 16 different forms, depending on the specified combinations of the REPLACE, SCORES, APPROXIMATIONS, and CORRELATIONS options. You can specify any combination of these options. To illustrate, assume that the data matrix consists of N observations and m variables, and n components are computed. Then define the following:

D the $N \times m$ matrix of original data with variable names that correspond to the names of the variables in the input data set. However, when you use the OPSCORE transformation on character variables, those variables are replaced by numeric variables that contain category numbers.

T the $N \times m$ matrix of transformed data with variable names constructed from the value of the TPREFIX= option (if you do not specify the REPLACE option) and the names of the variables in the input data set

S the $N \times n$ matrix of component scores with variable names constructed from the value of the PREFIX= option and integers

A the $N \times m$ matrix of data approximations with variable names constructed from the value of the APREFIX= option and the names of the variables in the input data set

$\mathbf{R_{TD}}$ the $m \times m$ matrix of correlations between the transformed variables and the original variables with variable names that correspond to the names of the variables in the input data set. When missing values exist, casewise deletion is used to compute the correlations.

$\mathbf{R_{TT}}$ the $m \times m$ matrix of correlations among the transformed variables with the variable names constructed from the value of the TPREFIX= option (if you do not specify the REPLACE option) and the names of the variables in the input data set

$\mathbf{R_{TS}}$ the $m \times n$ matrix of correlations between the transformed variables and the principal component scores (component structure matrix) with variable names constructed from the value of the PREFIX= option and integers

$\mathbf{R_{TA}}$ the $m \times m$ matrix of correlations between the transformed variables and the variable approximations with variable names constructed from the value of the APREFIX= option and the names of the variables in the input data set

To create a data set Work.A that contains all information, specify the following options in the PROC PRINQUAL statement:

```
proc prinqual scores approximations correlations out=a;
```

Also use a TRANSFORM statement appropriate for your data. Then the Work.A data set contains the following:

D	T	S	A
R_{TD}	R_{TT}	R_{TS}	R_{TA}

To eliminate the bottom partitions that contain the correlations and component structure, do not specify the CORRELATIONS option. For example, use the following PROC PRINQUAL statement with an appropriate TRANSFORM statement:

```
proc prinqual scores approximations out=a;
```

Then the Work.A data set contains the following:

D	T	S	A

Suppose you use the following PROC PRINQUAL statement (with an appropriate TRANSFORM statement):

```
proc prinqual out=a;
```

This creates a data set Work.A of the following form:

D	T

To output transformed data and component scores only, specify the following options in the PROC PRINQUAL statement:

```
proc prinqual replace scores out=a;
```

Then the Work.A data set contains the following:

T	S

TYPE and _NAME_ Variables

In addition to the preceding information, the output data set contains two character variables, the variable _TYPE_ (length 8) and the variable _NAME_ (the length depends on the VALIDVARNAME= system option).

The _TYPE_ variable has the value 'SCORE' if the observation contains variables, transformed variables, components, or data approximations; the _TYPE_ variable has the value 'CORR' if the observation contains correlations or component structure.

By default, the _NAME_ variable has values 'ROW1', 'ROW2', and so on, for the observations with _TYPE_='SCORE'. If you use an ID statement, the variable _NAME_ contains the formatted ID variable for SCORES observations. The values of the variable _NAME_ for observations with _TYPE_='CORR' are the names of the transformed variables.

Certain procedures, such as PROC PRINCOMP, which can use the PROC PRINQUAL output data set, issue a warning that the PROC PRINQUAL data set contains _NAME_ and _TYPE_ variables but is not a TYPE=CORR data set. You can ignore this warning.

Variable Names

The TPREFIX=, APREFIX=, and PREFIX= options specify prefixes for the transformed and approximation variable names and for principal component score variables, respectively. PROC PRINQUAL constructs transformed and approximation variable names from a prefix and the first characters of the original variable name. The number of characters in the prefix plus the number of characters in the original variable name (including the final digits, if any) required to uniquely designate the new variables should not exceed v, where v is the length from the VALIDVARNAME= system option (for example, 32 for VALIDVARNAME=V7). If the APREFIX= parameter that you specify is one character, PROC PRINQUAL adds the first $v - 1$ characters of the original variable name; if your prefix is four characters, only the first $v - 4$ characters of the original variable name are added.

Effect of the TSTANDARD= and COVARIANCE Options

The values in the output data set are affected by the TSTANDARD= and COVARIANCE options. If you specify TSTANDARD=NOMISS, the NOMISS standardization is performed on the transformed data after the iterations have been completed, but before the output data set is created. The new means and variances are used in creating the output data set. Then, if you do not specify the COVARIANCE option, the data are transformed to mean zero and variance one. The principal component scores and data approximations are computed from the resulting matrix. The data are then linearly transformed to have the mean and variance specified by the TSTANDARD= option. The data approximations are transformed so that the means within each pair of a transformed variable and its approximation are the same. The ratio of the variance of a variable approximation to the variance of the corresponding transformed variable equals the proportion of the variance of the variable that is accounted for by the components model.

If you specify the COVARIANCE option and do not specify TSTANDARD=Z, you can input the transformed data to PROC PRINCOMP, again specifying the COVARIANCE option, to perform a components analysis of the results of PROC PRINQUAL. Similarly, if you do not specify the COVARIANCE option with PROC PRINQUAL and you input the transformed data to PROC PRINCOMP without the COVARIANCE option, you receive the same report. However, some combinations of PROC PRINQUAL options, such as COVARIANCE and TSTANDARD=Z, while valid, produce approximations and scores that cannot be reproduced by PROC PRINCOMP.

The component scores in the output data set are computed from the correlations among the transformed variables, or from the covariances if you specified the COVARIANCE option. The component scores are computed after the TSTANDARD=NOMISS transformation, if specified. The means of the component scores in the output data set are always zero. The variances equal the corresponding eigenvalues, unless you specify the STANDARD option; then the variances are set to one.

Avoiding Constant Transformations

There are times when the optimal scaling produces a constant transformed variable. This can happen with the MONOTONE, UNTIE, and MSPLINE transformations when the target is negatively correlated with the original input variable. It can happen with all transformations when the target is uncorrelated with the original input variable. When this happens, the procedure modifies the target to avoid a constant transformation. This strategy avoids certain nonoptimal solutions.

If the transformation is monotonic and a constant transformed variable results, the procedure multiplies the target by –1 and tries the optimal scaling again. If the transformation is not monotonic or if the multiplication by –1 did not help, the procedure tries using a random target. If the transformation is still constant, the previous nonconstant transformation is retained. When a constant transformation is avoided by any strategy, this message is displayed: "A constant transformation was avoided for *name*."

Constant Variables

Constant and almost constant variables are zeroed and ignored.

Character OPSCORE Variables

Character OPSCORE variables are replaced by a numeric variable containing category numbers before the iterations, and the character values are discarded. Only the first eight characters are considered in determining category membership. If you want the original character variable in the output data set, give it a different name in the OPSCORE specification (OPSCORE(x / name=(x2)) and name the original variable in the ID statement (ID x;).

REITERATE Option Usage

You can use the REITERATE option to perform additional iterations when PROC PRINQUAL stops before the data have adequately converged. For example, suppose you execute the following code:

```
proc prinqual data=A cor out=B;
   transform mspline(X1-X5);
run;
```

If the transformations do not converge in the default 30 iterations, you can perform more iterations without repeating the first 30 iterations, as follows:

```
proc prinqual data=B reiterate cor out=B;
   transform mspline(X1-X5);
run;
```

Note that a WHERE statement is not necessary to exclude the correlation observations. They are automatically excluded because their _TYPE_ variable value is not 'SCORE'.

You can also use the REITERATE option to specify starting values other than the original values for the transformations. Providing alternate starting points might avoid local optima. Here are two examples.

```
proc prinqual data=A out=B;
   transform rank(X1-X5);
run;

proc prinqual data=B reiterate out=C;
   /* Use ranks as the starting point. */
   transform monotone(X1-X5);
run;

data B;
   set A;
   array TXS[5] TX1-TX5;
   do j = 1 to 5;
      TXS[j] = normal(0);
   end;
run;

proc prinqual data=B reiterate out=C;
   /* Use a random starting point. */
   transform monotone(X1-X5);
run;
```

Note that divergence with the REITERATE option, particularly in the second iteration, is not an error since the initial transformation is not required to be a valid member of the transformation family. When you specify the REITERATE option, the iteration does not terminate when the criterion change is negative during the first 10 iterations.

Passive Observations

Observations can be excluded from the analysis for several reasons, including zero weight, zero frequency, missing values in variables designated as IDENTITY, or missing values with the NOMISS option specified. These observations are passive in that they do not contribute to determining transformations, R square, total variance, and so on. However, some information can be computed for them, such as approximations, principal component scores, and transformed values. Passive observations in the output data set have a blank value for the variable _TYPE_.

Missing value estimates for passive observations might converge slowly with METHOD=MTV. In the following example, the missing value estimates should be 2, 5, and 8. Since the nonpassive observations do not change, the procedure converges in one iteration but the missing value estimates do not converge. The extra iterations produced by specifying CONVERGE=–1 and CCONVERGE=–1, as shown in the second PROC PRINQUAL step that follows, generate the expected results.

```
data A;
   input X Y;
   datalines;
1 1
2 .
3 3
4 4
5 .
6 6
7 7
8 .
9 9
;

proc prinqual nomiss data=A nomiss n=1 out=B method=mtv;
   transform lin(X Y);
run;

proc print;
run;

proc prinqual nomiss data=A nomiss n=1 out=B method=mtv converge=-1 cconverge=-1;
   transform lin(X Y);
run;

proc print;
run;
```

Computational Resources

This section provides information about the computational resources required to run PROC PRINQUAL.

Let

$$
\begin{aligned}
N &= \text{number of observations} \\
m &= \text{number of variables} \\
n &= \text{number of principal components} \\
k &= \text{maximum spline degree} \\
p &= \text{maximum number of knots}
\end{aligned}
$$

- For the MTV algorithm, more than

$$56m + 8Nm + 8\left(6N + (p + k + 2)(p + k + 11)\right)$$

  bytes of array space are required.

- For the MGV and MAC algorithms, more than $56m$ plus the maximum of the data matrix size and the optimal scaling work space bytes of array space are required. The data matrix size is $8Nm$ bytes. The optimal scaling work space requires less than $8\,(6N + (p + k + 2)(p + k + 11))$ bytes.
- For the MTV and MGV algorithms, more than $56m + 4m(m + 1)$ bytes of array space are required.
- PROC PRINQUAL tries to store the original and transformed data in memory. If there is not enough memory, a utility data set is used, potentially resulting in a large increase in execution time. The amount of memory for the preceding data formulas is an underestimate of the amount of memory needed to handle most problems. These formulas give an absolute minimum amount of memory required. If a utility data set is used, and if memory could be used with perfect efficiency, then roughly the amount of memory stated previously would be needed. In reality, most problems require at least two or three times the minimum.
- PROC PRINQUAL sorts the data once. The sort time is roughly proportional to $mN^{3/2}$.
- For the MTV algorithm, the time required to compute the variable approximations is roughly proportional to $2Nm^2 + 5m^3 + nm^2$.
- For the MGV algorithm, one regression analysis per iteration is required to compute model parameter estimates. The time required to accumulate the crossproduct matrix is roughly proportional to Nm^2. The time required to compute the regression coefficients is roughly proportional to m^3. For each variable for each iteration, the swept crossproduct matrix is updated with time roughly proportional to m(N+m). The swept crossproduct matrix is updated for each variable with time roughly proportional to m^2, until computations are refreshed, requiring all sweeps to be performed again.
- The only computationally intensive part of the MAC algorithm is the optimal scaling, since variable approximations are simple averages.
- Each optimal scaling is a multiple regression problem, although some transformations are handled with faster special-case algorithms. The number of regressors for the optimal scaling problems depends on the original values of the variable and the type of transformation. For each monotone spline transformation, an unknown number of multiple regressions is required to find a set of coefficients that satisfies the constraints. The B-spline basis is generated twice for each SPLINE and MSPLINE transformation for each iteration. The time required to generate the B-spline basis is roughly proportional to Nk^2.

Displayed Output

The main output from PROC PRINQUAL is the output data set. However, the procedure does produce displayed output in the form of an iteration history table that includes the following:

- iteration number
- the criterion being optimized
- criterion change
- maximum and average absolute change in standardized variable scores computed over variables that can be iteratively transformed
- notes
- final convergence status

ODS Table Names

PROC PRINQUAL assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 99.4 along with the PROC PRINQUAL statement options needed to produce the table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 99.4 ODS Tables Produced by PROC PRINQUAL

ODS Table Name	Description	Option
ConvergenceStatus	Convergence status	Default
Footnotes	Iteration history footnotes	Default
MAC	MAC iteration history	METHOD=MAC
MGV	MGV iteration history	METHOD=MGV
MTV	MTV iteration history	METHOD=MTV
PctVar	Percentage of variance	Nonprinting

The nonprinting PctVar table is not displayed and does not appear in the ODS trace output unless you specify it in an ODS OUTPUT statement, as in the following example:

```
ods output pctvar=pvardataset;
```

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

The plots are produced only when you specify the options shown in the table. You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC PRINQUAL generates are listed in Table 99.5, along with the required statements and options.

Table 99.5 Graphs Produced by PROC PRINQUAL

ODS Graph Name	Plot Description	Option
MDPrefPlot	Multidimensional preference analysis	MDPREF
TransformationPlot	Variable transformation	PLOTS=TRANSFORMATION

Examples: PRINQUAL Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 99.1: Multidimensional Preference Analysis of Automobile Data

(View the complete code for this example (prqex1.sas) in the example repository.)

This example uses PROC PRINQUAL to perform a nonmetric multidimensional preference (MDPREF) analysis (Carroll 1972). MDPREF analysis is a principal component analysis of a data matrix with columns that correspond to people and rows that correspond to objects. The data are ratings or rankings of each person's preference for each object. The data are the transpose of the usual multivariate data matrix. (In other words, the columns are people; in the more typical matrix the rows represent people.) The final result of an MDPREF analysis is a biplot (Gabriel 1981) of the resulting preference space. A biplot displays the judges and objects in a single plot by projecting them onto the plane in the transformed variable space that accounts for the most variance.

In 1980, 25 judges gave their preferences for each of 17 new automobiles. The ratings were made on a 0 to 9 scale, with 0 meaning very weak preference and 9 meaning very strong preference for the automobile. The following statements create a SAS data set with the manufacturer and model of each automobile along with the ratings:

```
title 'Preference Ratings for Automobiles Manufactured in 1980';

options validvarname=any;

data CarPref;
   input Make $ 1-10 Model $ 12-22 @25 ('1'n-'25'n) (1.);
   datalines;
Cadillac   Eldorado     8007990491240508971093809
Chevrolet  Chevette     0051200423451043003515698
Chevrolet  Citation     4053305814161643544747795
Chevrolet  Malibu       6027400723121345545668658
Ford       Fairmont     2024006715021443530648655
Ford       Mustang      5007197705021101850657555
Ford       Pinto        0021000303030201500514078
Honda      Accord       5956897609699952998975078
Honda      Civic        4836709507488852567765075
Lincoln    Continental  7008990592230409962091909
Plymouth   Gran Fury    7006000434101107333458708
Plymouth   Horizon      3005005635461302444675655
Plymouth   Volare       4005003614021602754476555
Pontiac    Firebird     0107895613201206958265907
Volkswagen Dasher       4858696508877795377895000
Volkswagen Rabbit       4858509709695795487885000
Volvo      DL           9989998909999987989919000
;
```

The following statements run PROC PRINCOMP and create a scree plot. The results of this step are shown in Output 99.1.1.

```
ods graphics on;
* Principal Component Analysis of the Original Data;
proc princomp data=CarPref;
   ods select ScreePlot;
   var '1'n-'25'n;
run;
```

Output 99.1.1 Eigenvalue Plot

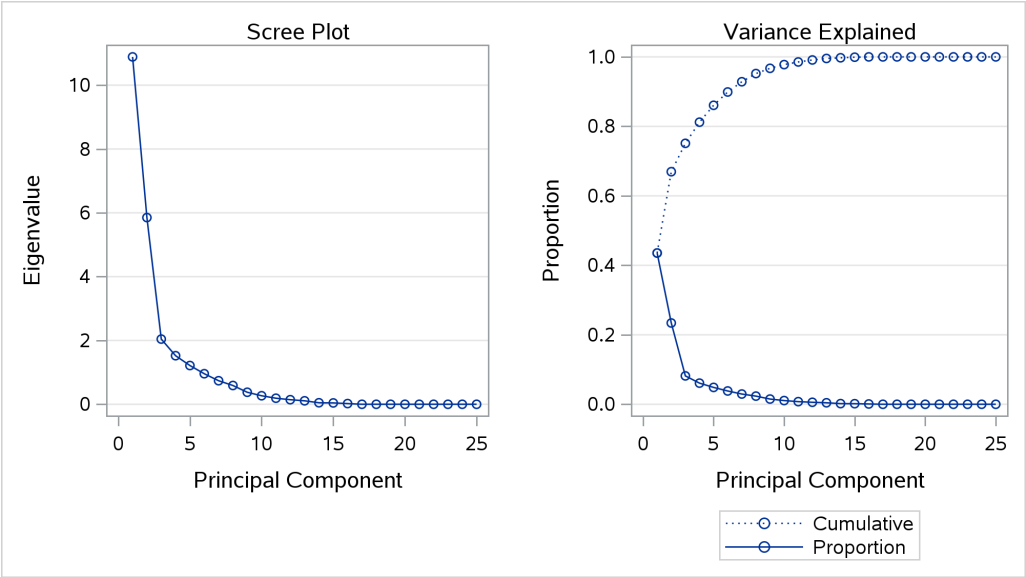

The scree or eigenvalue plot in Output 99.1.1 shows that two principal components should be retained. There is a clear separation between the first two components and the remaining components. There are eight eigenvalues that are precisely zero because there are eight fewer observations than variables in the data matrix. One additional eigenvalue is zero, for a total of nine zero eigenvalues, since the correlation matrix is based on centered data. The following statements create the data set and perform a principal component analysis of the original data.

PROC PRINQUAL fits the nonmetric MDPREF model. PROC PRINQUAL monotonically transforms the raw judgments to maximize the proportion of variance accounted for by the first two principal components. The MONOTONE option is specified in the TRANSFORM statement to request a nonmetric MDPREF analysis; alternatively, you can instead specify the IDENTITY option for a metric analysis. Several options are used in the PROC PRINQUAL statement. The option DATA=CarPref specifies the input data set, OUT=Results creates an output data set, and N=2 and the default METHOD=MTV transform the data to better fit a two-component model. The REPLACE option replaces the original data with the monotonically transformed data in the OUT= data set. The MDPREF option standardizes the component scores to variance one so that the geometry of the biplot is correct, and it creates two variables in the OUT= data set named Prin1 and Prin2. These variables contain the standardized principal component scores and structure matrix, which are used to make the biplot. If the variables in data matrix $\mathbf{X}$ are standardized to mean zero and variance one, and n is the number of rows in $\mathbf{X}$, then $\mathbf{X} = \mathbf{V}\boldsymbol{\Lambda}^{1/2}\mathbf{W}'$ is the principal component model, where $\mathbf{X}'\mathbf{X}/(n-1) = \mathbf{W}\boldsymbol{\Lambda}\mathbf{W}'$. The $\mathbf{W}$ and $\boldsymbol{\Lambda}$ contain the eigenvectors and eigenvalues of the correlation matrix of $\mathbf{X}$. The first two columns of $\mathbf{V}$, the standardized component scores, and $\mathbf{W}\boldsymbol{\Lambda}^{1/2}$, which is the structure

matrix, are output. The advantage of creating a biplot based on principal components is that coordinates do not depend on the sample size. The following statements transform the data and produce Output 99.1.2.

```
* Transform the Data to Better Fit a Two Component Model;
proc prinqual data=CarPref out=Results n=2 replace mdpref;
   title2 'Multidimensional Preference (MDPREF) Analysis';
   title3 'Optimal Monotonic Transformation of Preference Data';
   id model;
   transform monotone('1'n-'25'n);
run;
```

Output 99.1.2 PRINQUAL Iteration History

Preference Ratings for Automobiles Manufactured in 1980
Multidimensional Preference (MDPREF) Analysis
Optimal Monotonic Transformation of Preference Data

The PRINQUAL Procedure

PRINQUAL MTV Algorithm Iteration History

Iteration Number	Average Change	Maximum Change	Proportion of Variance	Criterion Change	Note
1	0.24994	1.28017	0.66946		
2	0.07223	0.36958	0.80194	0.13249	
3	0.04522	0.29026	0.81598	0.01404	
4	0.03096	0.25213	0.82178	0.00580	
5	0.02182	0.23045	0.82493	0.00315	
6	0.01602	0.19017	0.82680	0.00187	
7	0.01219	0.14748	0.82793	0.00113	
8	0.00953	0.11031	0.82861	0.00068	
9	0.00737	0.06461	0.82904	0.00043	
10	0.00556	0.04469	0.82930	0.00026	
11	0.00445	0.04087	0.82944	0.00014	
12	0.00381	0.03706	0.82955	0.00011	
13	0.00319	0.03348	0.82965	0.00009	
14	0.00255	0.02999	0.82971	0.00006	
15	0.00213	0.02824	0.82976	0.00005	
16	0.00183	0.02646	0.82980	0.00004	
17	0.00159	0.02472	0.82983	0.00003	
18	0.00139	0.02305	0.82985	0.00003	
19	0.00123	0.02145	0.82988	0.00002	
20	0.00109	0.01993	0.82989	0.00002	
21	0.00096	0.01850	0.82991	0.00001	
22	0.00086	0.01715	0.82992	0.00001	
23	0.00076	0.01588	0.82993	0.00001	
24	0.00067	0.01440	0.82994	0.00001	
25	0.00059	0.00871	0.82994	0.00001	
26	0.00050	0.00720	0.82995	0.00000	
27	0.00043	0.00642	0.82995	0.00000	
28	0.00037	0.00573	0.82995	0.00000	
29	0.00031	0.00510	0.82995	0.00000	
30	0.00027	0.00454	0.82995	0.00000	Not Converged

Output 99.1.2 *continued*

WARNING: Failed to converge, however criterion change is less than 0.0001.

The iteration history displayed by PROC PRINQUAL indicates that the proportion of variance is increased from an initial 0.66946 to 0.82995. The proportion of variance accounted for by PROC PRINQUAL on the first iteration equals the cumulative proportion of variance shown by PROC PRINCOMP for the first two principal components. PROC PRINQUAL's initial iteration performs a standard principal component analysis of the raw data. The columns labeled Average Change, Maximum Change, and Criterion Change contain values that always decrease, indicating that PROC PRINQUAL is improving the transformations at a monotonically decreasing rate over the iterations. This does not always happen, and when it does not, it suggests that the analysis might be converging to a degenerate solution. See Example 99.2 for a discussion of a degenerate solution. The algorithm does not converge in 30 iterations. However, the criterion change is small, indicating that more iterations are unlikely to have much effect on the results.

The biplot, shown in Output 99.1.3, is automatically displayed by PROC PRINQUAL when ODS Graphics is enabled and the MDPREF option is specified.

Output 99.1.3 Biplot Made with PRINQUAL

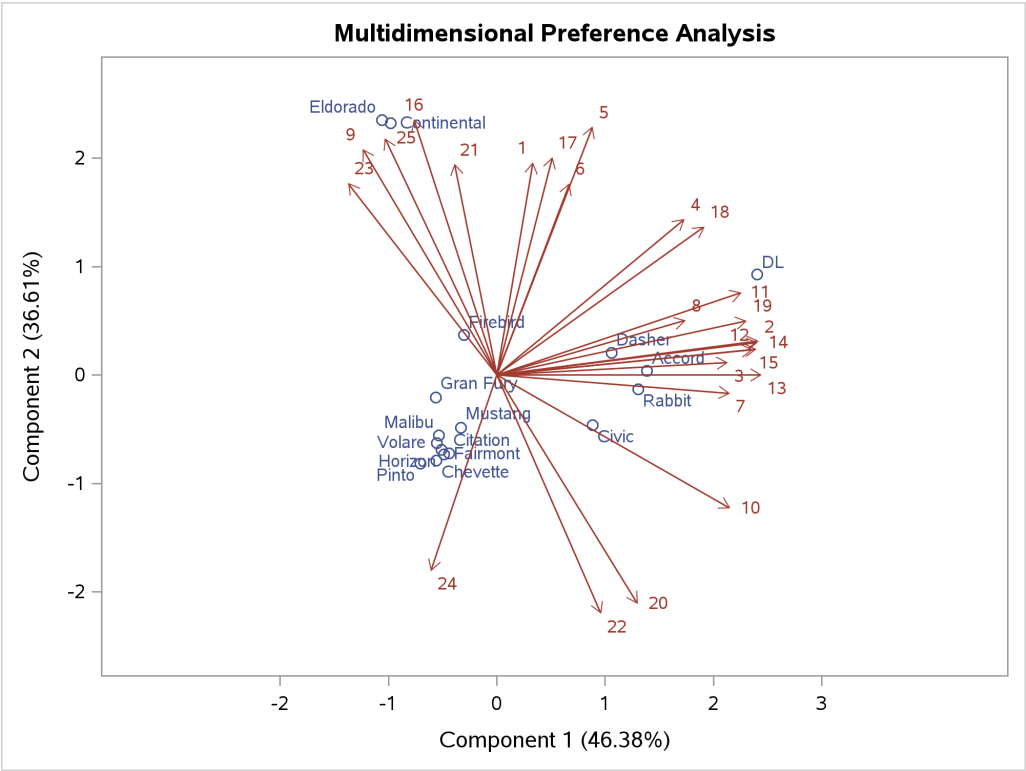

The second PROC PRINCOMP analysis is performed on the transformed data. The WHERE statement is used to retain only the monotonically transformed judgments. The scree plot shows that the first two eigenvalues are now much larger than the remaining smaller eigenvalues. The second eigenvalue has increased markedly at the expense of the next several eigenvalues. Two principal components seem to be necessary and sufficient to adequately describe these judges' preferences for these automobiles. The cumulative proportion of variance displayed by PROC PRINCOMP for the first two principal components is 0.83. The following

statements perform the analysis and produce Output 99.1.4:

```
* Final Principal Component Analysis;
proc princomp data=Results;
   ods select ScreePlot;
   var '1'n-'25'n;
   where _TYPE_='SCORE';
run;
```

Output 99.1.4 Transformed Data Eigenvalue Plot

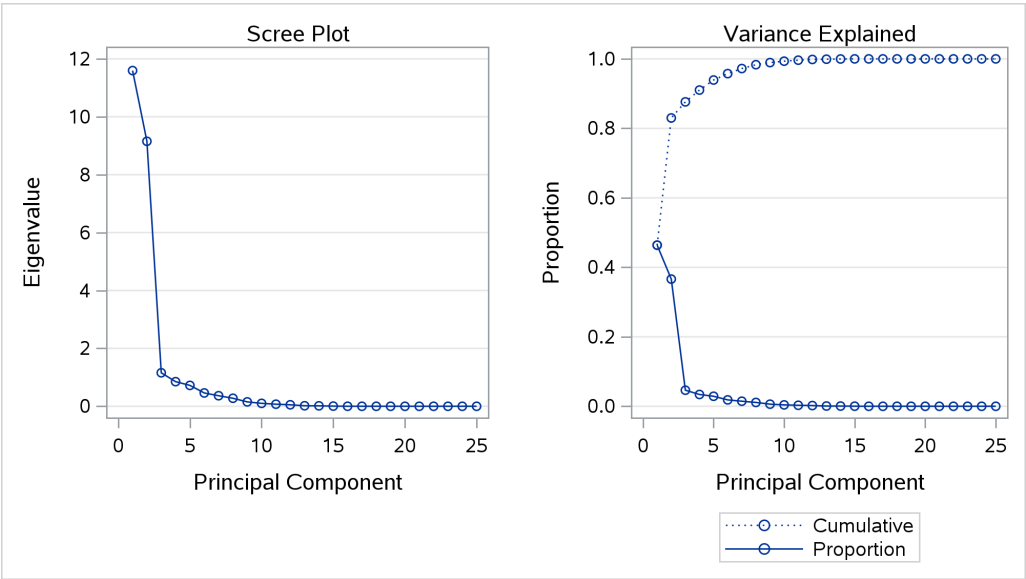

The remainder of the example discusses the MDPREF biplot. A biplot is a plot that displays the relation between the row points and the columns of a data matrix. The rows of $\mathbf{V}$, the standardized component scores, and $\mathbf{W}\mathbf{\Lambda}^{1/2}$, the structure matrix, contain enough information to reproduce $\mathbf{X}$. The (i, j) element of $\mathbf{X}$ is the product of row i of $\mathbf{V}$ and row j of $\mathbf{W}\mathbf{\Lambda}^{1/2}$. If all but the first two columns of $\mathbf{V}$ and $\mathbf{W}\mathbf{\Lambda}^{1/2}$ are discarded, the (i, j) element of $\mathbf{X}$ is approximated by the product of row i of $\mathbf{V}$ and row j of $\mathbf{W}\mathbf{\Lambda}^{1/2}$.

Since the MDPREF analysis is based on a principal component model, the dimensions of the MDPREF biplot are the first two principal components. The first principal component is the longest dimension through the MDPREF biplot. The first principal component is overall preference, which is the most salient dimension in the preference judgments. One end points in the direction that is on the average preferred most by the judges, and the other end points in the least preferred direction. The second principal component is orthogonal to the first principal component, and it is the orthogonal direction that is the second most salient. The interpretation of the second dimension varies from example to example.

With an MDPREF biplot, it is geometrically appropriate to represent each automobile (object) by a point and each judge by a vector. The automobile points have coordinates that are the scores of the automobile on the first two principal components. The judge vectors emanate from the origin of the space and go through a point whose coordinates are the coefficients of the judge (variable) on the first two principal components.

The absolute length of a vector is arbitrary. However, the relative lengths of the vectors indicate fit, with the squared lengths being proportional to the communalities that you can get in PROC FACTOR output. The direction of the vector indicates the direction that is most preferred by the individual judge, with preference increasing as the vector moves from the origin. Let $\mathbf{v}'$ be row i of $\mathbf{V}$, $\mathbf{u}'$ be row j of $\mathbf{U} = \mathbf{W}\mathbf{\Lambda}^{1/2}$, $\|\mathbf{v}\|$

be the length of $\mathbf{v}$, $\|\mathbf{u}\|$ be the length of $\mathbf{u}$, and θ be the angle between $\mathbf{v}$ and $\mathbf{u}$. The predicted degree of preference that an individual judge has for an automobile is $\mathbf{u}'\mathbf{v} = \|\mathbf{u}\|\ \|\mathbf{v}\| \cos\theta$. Each automobile point can be orthogonally projected onto the vector. The projection of automobile *i* on vector *j* is $\mathbf{u}((\mathbf{u}'\mathbf{v})/(\mathbf{u}'\mathbf{u}))$, and the length of this projection is $\|\mathbf{v}\| \cos\theta$. The automobile that projects farthest along a vector in the direction it points is that judge's most preferred automobile, since the length of this projection, $\|\mathbf{v}\| \cos\theta$, differs from the predicted preference, $\|\mathbf{u}\|\ \|\mathbf{v}\| \cos\theta$, only by $\|\mathbf{u}\|$, which is constant for each judge.

To interpret the biplot, look for directions through the plot that show a continuous change in some attribute of the automobiles, or look for regions in the plot that contain clusters of automobile points and determine what attributes the automobiles have in common. Points that are tightly clustered in a region of the plot represent automobiles that have the same preference patterns across the judges. Vectors that point in roughly the same direction represent judges who have similar preference patterns.

In the biplot, American automobiles are located at the left of the space, while European and Japanese automobiles are located at the right. At the top of the space are expensive American automobiles (Cadillac Eldorado, Lincoln Continental), while at the bottom are inexpensive ones (Ford Pinto, Chevrolet Chevette). The first principal component differentiates American from imported automobiles, and the second arranges automobiles by price and other associated characteristics.

The two expensive American automobiles form a cluster, the sporty automobile (Pontiac Firebird) is by itself, the Volvo DL is by itself, and the remaining imported autos form a cluster, as do the remaining American autos. It seems there are 5 prototypical automobiles in this set of 17, in terms of preference patterns among the 25 judges.

Most of the judges prefer the imported automobiles, especially the Volvo. There is also a fairly large minority that prefer the expensive autos, whether or not they are American (those with vectors that point toward one o'clock), or simply prefer expensive American automobiles (vectors that point toward eleven o'clock). There are two judges who prefer anything except expensive American autos (five o'clock vectors), and one who prefers inexpensive American autos (seven o'clock vector).

Several vectors point toward the upper-right corner of the plot, toward a region with no automobiles. This is the region between the European and Japanese autos at the right and the luxury autos at the top. This suggests that there is a market for luxury Japanese and European automobiles.

The next part of this example modifies the graph template for the MDPREF plot to display group information (the make of the automobile) in the MDPREF plot. First you need to run the PROC PRINQUAL step with ODS trace output enabled to find the name of the graph template for the MDPREF plot:

```
ods trace on;
proc prinqual data=CarPref out=Results n=2 replace mdpref;
   title2 'Multidimensional Preference (MDPREF) Analysis';
   title3 'Optimal Monotonic Transformation of Preference Data';
   id model;
   transform monotone('1'n-'25'n);
run;
```

The results are as follows:

```
Output Added:
-------------
Name:       MDPrefPlot
Label:      2 by 1
Template:   Stat.Prinqual.Graphics.MDPref
```

```
Path:       Prinqual.MDPREF.MDPrefPlot
-------------
```

The following step displays the template:

```
proc template;
   source Stat.Prinqual.Graphics.MDPref;
run;
```

The following template is displayed:

```
define statgraph Stat.Prinqual.Graphics.MDPref;
   notes "Multidimensional Preference Analysis Plot";
   dynamic xVar yVar xVec yVec ylab xlab yshortlab xshortlab xOri yOri
      stretch;
   begingraph;
      entrytitle "Multidimensional Preference Analysis";
      layout overlayequated / equatetype=fit xaxisopts=(label=XLAB shortlabel
         =XSHORTLAB offsetmin=0.1 offsetmax=0.1) yaxisopts=(label=YLAB
         shortlabel=YSHORTLAB offsetmin=0.1 offsetmax=0.1);
         scatterplot y=YVAR x=XVAR / datalabel=IDLAB1 rolename=(_tip1=
            OBSNUMVAR _id2=IDLAB2 _id3=IDLAB3 _id4=IDLAB4 _id5=IDLAB5 _id6=
            IDLAB6 _id7=IDLAB7 _id8=IDLAB8 _id9=IDLAB9 _id10=IDLAB10 _id11=
            IDLAB11 _id12=IDLAB12 _id13=IDLAB13 _id14=IDLAB14 _id15=IDLAB15
            _id16=IDLAB16 _id17=IDLAB17 _id18=IDLAB18 _id19=IDLAB19 _id20=
            IDLAB20) tip=(y x datalabel _tip1 _id2 _id3 _id4 _id5 _id6 _id7
            _id8 _id9 _id10 _id11 _id12 _id13 _id14 _id15 _id16 _id17 _id18
            _id19 _id20) datalabelattrs=GRAPHVALUETEXT (color=
            GraphData1:ContrastColor) markerattrs=GRAPHDATA1;
         vectorplot y=YVEC x=XVEC xorigin=0 yorigin=0 / datalabel=LABEL2VAR
            shaftprotected=false rolename=(_tip1=VNAME _tip2=VLABEL _tip3=
            YORI _tip4=XORI _tip5=LENGTH _tip6=LENGTH2) tip=(y x datalabel
            _tip1 _tip2 _tip3 _tip4 _tip5 _tip6) datalabelattrs=
            GRAPHVALUETEXT (color=GraphData2:ContrastColor) lineattrs=
            GRAPHDATA2 (pattern=solid) primary=true;
         if (0)
            entry "Vector Stretch = " STRETCH / autoalign=(topright topleft
               bottomright bottomleft right left top bottom);
         endif;
      endlayout;
   endgraph;
end;
```

The following step adds a PROC PRINQUAL statement and a RUN statement, removes attribute options from the SCATTERPLOT statement, adds a GROUP=IDLAB2 option to use the second ID variable as a group variable, and adds a NAME='s' option along with a DISCRETELEGEND statement to display the groups in a legend:

```
proc template;
   define statgraph Stat.Prinqual.Graphics.MDPref;
      notes "Multidimensional Preference Analysis Plot";
      dynamic xVar yVar xVec yVec ylab xlab yshortlab xshortlab xOri yOri
         stretch;
      begingraph;
```

```
            entrytitle "Multidimensional Preference Analysis";
            layout overlayequated / equatetype=fit xaxisopts=(label=XLAB shortlabel
               =XSHORTLAB offsetmin=0.1 offsetmax=0.1) yaxisopts=(label=YLAB
               shortlabel=YSHORTLAB offsetmin=0.1 offsetmax=0.1);
               scatterplot y=YVAR x=XVAR / datalabel=IDLAB1 rolename=(_tip1=
                  OBSNUMVAR _id2=IDLAB2 _id3=IDLAB3 _id4=IDLAB4 _id5=IDLAB5 _id6=
                  IDLAB6 _id7=IDLAB7 _id8=IDLAB8 _id9=IDLAB9 _id10=IDLAB10 _id11=
                  IDLAB11 _id12=IDLAB12 _id13=IDLAB13 _id14=IDLAB14 _id15=IDLAB15
                  _id16=IDLAB16 _id17=IDLAB17 _id18=IDLAB18 _id19=IDLAB19 _id20=
                  IDLAB20) tip=(y x datalabel _tip1 _id2 _id3 _id4 _id5 _id6 _id7
                  _id8 _id9 _id10 _id11 _id12 _id13 _id14 _id15 _id16 _id17 _id18
                  _id19 _id20)

                  group=idlab2 name='s'; *<==== add the group variable ====<<<<;

               vectorplot y=YVEC x=XVEC xorigin=0 yorigin=0 / datalabel=LABEL2VAR
                  shaftprotected=false rolename=(_tip1=VNAME _tip2=VLABEL _tip3=
                  YORI _tip4=XORI _tip5=LENGTH _tip6=LENGTH2) tip=(y x datalabel
                  _tip1 _tip2 _tip3 _tip4 _tip5 _tip6) datalabelattrs=
                  GRAPHVALUETEXT (color=GraphData2:ContrastColor) lineattrs=
                  GRAPHDATA2 (pattern=solid) primary=true;

               discretelegend 's'; *<==== add a legend ====<<<<;

               if (0)
                  entry "Vector Stretch = " STRETCH / autoalign=(topright topleft
                     bottomright bottomleft right left top bottom);
               endif;
            endlayout;
         endgraph;
      end;
   run;
```

The following step creates the MDPREF plot with the make of the automobile added as a second ID variable and displayed in the graph as a group variable:

```
   proc prinqual data=CarPref out=Results n=2 replace mdpref;
      title2 'Multidimensional Preference (MDPREF) Analysis';
      title3 'Optimal Monotonic Transformation of Preference Data';
      id model make;
      transform monotone('1'n-'25'n);
   run;
```

The results are displayed in Output 99.1.5.

Output 99.1.5 Biplot with a Group Variable

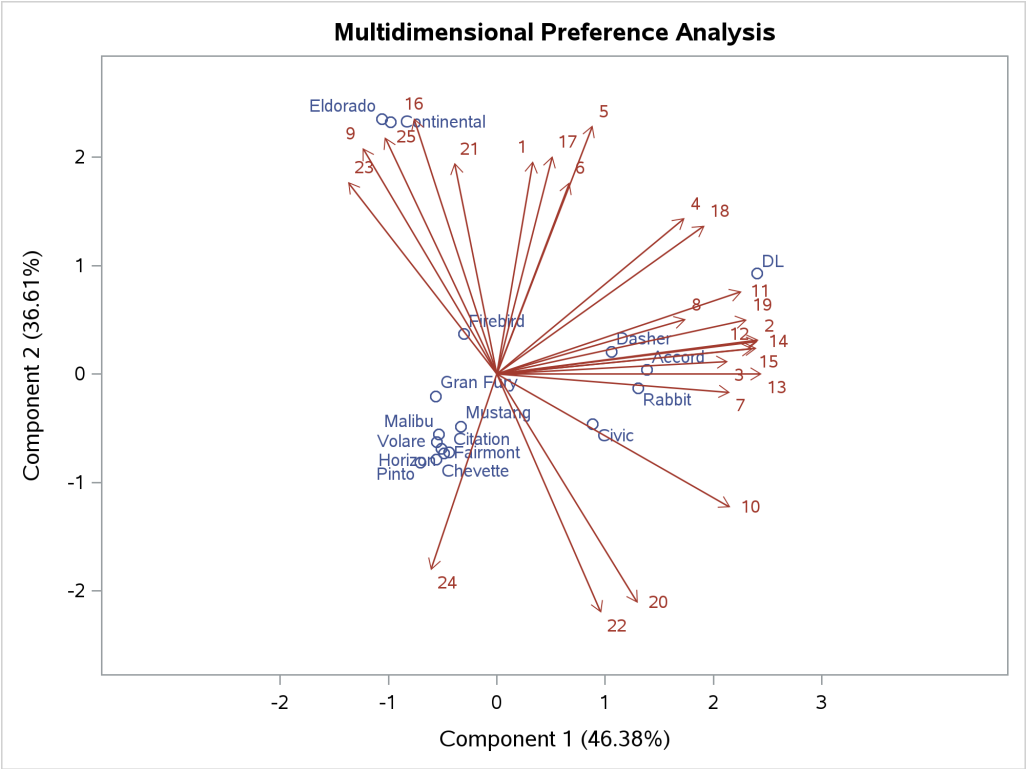

The following step restores the default template:

```
proc template;
   delete Stat.Prinqual.Graphics.MDPref / store=sasuser.templat;
run;
```

Example 99.2: Principal Components of Basketball Rankings

(View the complete code for this example (prqex2.sas) in the example repository.)

The data in this example are 1985–1986 preseason rankings of 35 U.S. college basketball teams by 10 different news services. The services do not all rank the same teams or the same number of teams, so there are missing values in these data. Each of the 35 teams in the data set is ranked by at least one news service. One way of summarizing these data is with a principal component analysis, since the rankings should all be related to a single underlying variable, the first principal component.

You can use PROC PRINQUAL to estimate the missing ranks and compute scores for all observations. You can formulate a PROC PRINQUAL analysis that assumes that the observed ranks are ordinal variables and replaces the ranks with new numbers that are monotonic with the ranks and better fit the one principal component model. The missing rank estimates need to be constrained since a news service would have positioned the unranked teams below the teams it ranked. PROC PRINQUAL should impose order constraints within the nonmissing values and between the missing and nonmissing values, but not within the missing

values. PROC PRINQUAL has sophisticated missing data handling facilities; however, these facilities cannot directly handle this problem. The solution requires reformulating the problem.

By performing some preliminary data manipulations, specifying the N=1 option in the PROC PRINQUAL statement, and specifying the UNTIE transformation in the TRANSFORM statement, you can make the missing value estimates conform to the requirements. The PROC MEANS step finds the largest rank for each variable. The next DATA step replaces missing values with a value that is one larger than the largest observed rank. The PROC PRINQUAL N=1 option specifies that the variables should be transformed to make them as one-dimensional as possible. The UNTIE transformation in the TRANSFORM statement monotonically transforms the ranks, untying any ties in an optimal way. Because the only ties are for the values that replace the missing values, and because these values are larger than the observed values, the rescoring of the data satisfies the preceding requirements.

The following statements create the data set and perform the transformations discussed previously. These statements produce Output 99.2.1 and Output 99.2.2.

```
*  Preseason 1985 College Basketball Rankings
*  (rankings of 35 teams by 10 news services)
*
*  Note:(a) Various news services rank varying numbers of teams.
*       (b) Not all 35 teams are ranked by all news services.
*       (c) Each team is ranked by at least one service.
*       (d) Rank 20 is missing for UPI.;

title1 '1985 Preseason College Basketball Rankings';

data bballm;
   input School $13. CSN DurhamSun DurhamHerald WashingtonPost
         USA_Today SportMagazine InsideSports UPI AP SportsIllustrated;
   label CSN               = 'Community Sports News (Chapel Hill, NC)'
         DurhamSun         = 'Durham Sun'
         DurhamHerald      = 'Durham Morning Herald'
         WashingtonPost    = 'Washington Post'
         USA_Today         = 'USA Today'
         SportMagazine     = 'Sport Magazine'
         InsideSports      = 'Inside Sports'
         UPI               = 'United Press International'
         AP                = 'Associated Press'
         SportsIllustrated = 'Sports Illustrated'
         ;
   format CSN--SportsIllustrated 5.1;
   datalines;
Louisville     1  8  1  9  8  9  6 10  9  9
Georgia Tech   2  2  4  3  1  1  1  2  1  1
Kansas         3  4  5  1  5 11  8  4  5  7
Michigan       4  5  9  4  2  5  3  1  3  2
Duke           5  6  7  5  4 10  4  5  6  5
UNC            6  1  2  2  3  4  2  3  2  3
Syracuse       7 10  6 11  6  6  5  6  4 10
Notre Dame     8 14 15 13 11 20 18 13 12  .
Kentucky       9 15 16 14 14 19 11 12 11 13
LSU           10  9 13  . 13 15 16  9 14  8
DePaul        11  . 21 15 20  . 19  .  . 19
```

```
Georgetown    12  7  8  6  9  2  9  8  8  4
Navy          13 20 23 10 18 13 15  . 20  .
Illinois      14  3  3  7  7  3 10  7  7  6
Iowa          15 16  .  . 23  .  . 14  . 20
Arkansas      16  .  .  . 25  .  .  .  . 16
Memphis State 17  . 11  . 16  8 20  . 15 12
Washington    18  .  .  .  .  .  . 17  .  .
UAB           19 13 10  . 12 17  . 16 16 15
UNLV          20 18 18 19 22  . 14 18 18  .
NC State      21 17 14 16 15  . 12 15 17 18
Maryland      22  .  .  . 19  .  .  . 19 14
Pittsburgh    23  .  .  .  .  .  .  .  .  .
Oklahoma      24 19 17 17 17 12 17  . 13 17
Indiana       25 12 20 18 21  .  .  .  .  .
Virginia      26  . 22  .  . 18  .  .  .  .
Old Dominion  27  .  .  .  .  .  .  .  .  .
Auburn        28 11 12  8 10  7  7 11 10 11
St. Johns     29  .  .  .  . 14  .  .  .  .
UCLA          30  .  .  .  .  .  . 19  .  .
St. Joseph's   .  . 19  .  .  .  .  .  .  .
Tennessee      .  . 24  .  . 16  .  .  .  .
Montana        .  .  . 20  .  .  .  .  .  .
Houston        .  .  .  . 24  .  .  .  .  .
Virginia Tech  .  .  .  .  .  . 13  .  .  .
;

* Find maximum rank for each news service and replace
* each missing value with the next highest rank.;

proc means data=bballm noprint;
   output out=maxrank
      max=mcsn mdurs mdurh mwas musa mspom mins mupi map mspoi;
run;

data bball;
   set bballm;
   if _n_=1 then set maxrank;
   array services[10] CSN--SportsIllustrated;
   array maxranks[10] mcsn--mspoi;
   keep School CSN--SportsIllustrated;
   do i=1 to 10;
      if services[i]=. then services[i]=maxranks[i]+1;
   end;
run;

* Assume that the ranks are ordinal and that unranked teams would have
* been ranked lower than ranked teams.  Monotonically transform all ranked
* teams while estimating the unranked teams.  Enforce the constraint that
* the missing ranks are estimated to be less than the observed ranks.
* Order the unranked teams optimally within this constraint.  Do this so
* as to maximize the variance accounted for by one linear combination.
* This makes the data as nearly rank one as possible, given the constraints.
*
* NOTE: The UNTIE transformation should be used with caution.
```

```
* It frequently produces degenerate results.;

ods graphics on;

proc prinqual data=bball out=tbball scores n=1 tstandard=z
   plots=transformations;
   title2 'Optimal Monotonic Transformation of Ranked Teams';
   title3 'with Constrained Estimation of Unranked Teams';
   transform untie(CSN -- SportsIllustrated);
   id School;
run;
```

Output 99.2.1 PRINQUAL Iteration History

1985 Preseason College Basketball Rankings
Optimal Monotonic Transformation of Ranked Teams
with Constrained Estimation of Unranked Teams

The PRINQUAL Procedure

PRINQUAL MTV Algorithm Iteration History

Iteration Number	Average Change	Maximum Change	Proportion of Variance	Criterion Change	Note
1	0.18563	0.76531	0.85850		
2	0.03225	0.14627	0.94362	0.08512	
3	0.02126	0.10530	0.94669	0.00307	
4	0.01467	0.07526	0.94801	0.00132	
5	0.01067	0.05282	0.94865	0.00064	
6	0.00800	0.03669	0.94899	0.00034	
7	0.00617	0.02862	0.94919	0.00020	
8	0.00486	0.02636	0.94932	0.00013	
9	0.00395	0.02453	0.94941	0.00009	
10	0.00327	0.02300	0.94947	0.00006	
11	0.00275	0.02166	0.94952	0.00005	
12	0.00236	0.02041	0.94956	0.00004	
13	0.00205	0.01927	0.94959	0.00003	
14	0.00181	0.01818	0.94962	0.00003	
15	0.00162	0.01719	0.94964	0.00002	
16	0.00147	0.01629	0.94966	0.00002	
17	0.00136	0.01546	0.94968	0.00002	
18	0.00128	0.01469	0.94970	0.00002	
19	0.00121	0.01398	0.94971	0.00001	
20	0.00115	0.01332	0.94973	0.00001	
21	0.00111	0.01271	0.94974	0.00001	
22	0.00105	0.01213	0.94975	0.00001	
23	0.00099	0.01155	0.94976	0.00001	
24	0.00095	0.01095	0.94977	0.00001	
25	0.00091	0.01038	0.94978	0.00001	
26	0.00088	0.00986	0.94978	0.00001	
27	0.00084	0.00936	0.94979	0.00001	
28	0.00081	0.00889	0.94980	0.00001	
29	0.00077	0.00846	0.94980	0.00000	
30	0.00073	0.00805	0.94980	0.00000	Not Converged

WARNING: Failed to converge, however criterion change is less than 0.0001.

Output 99.2.2 Transformations

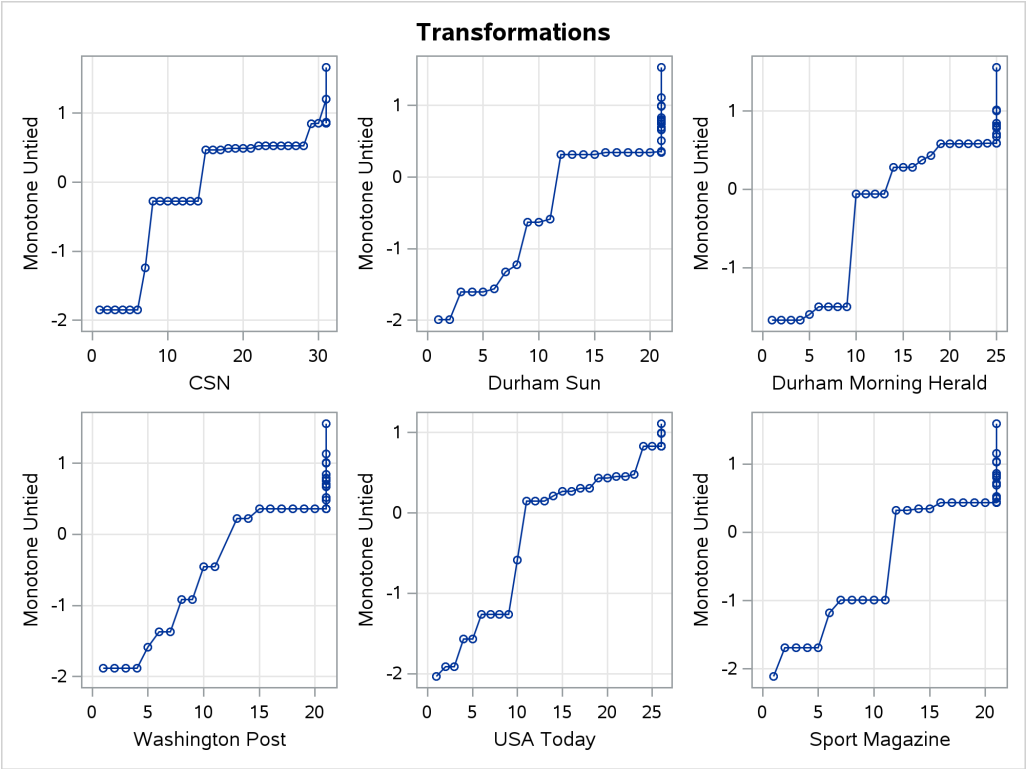

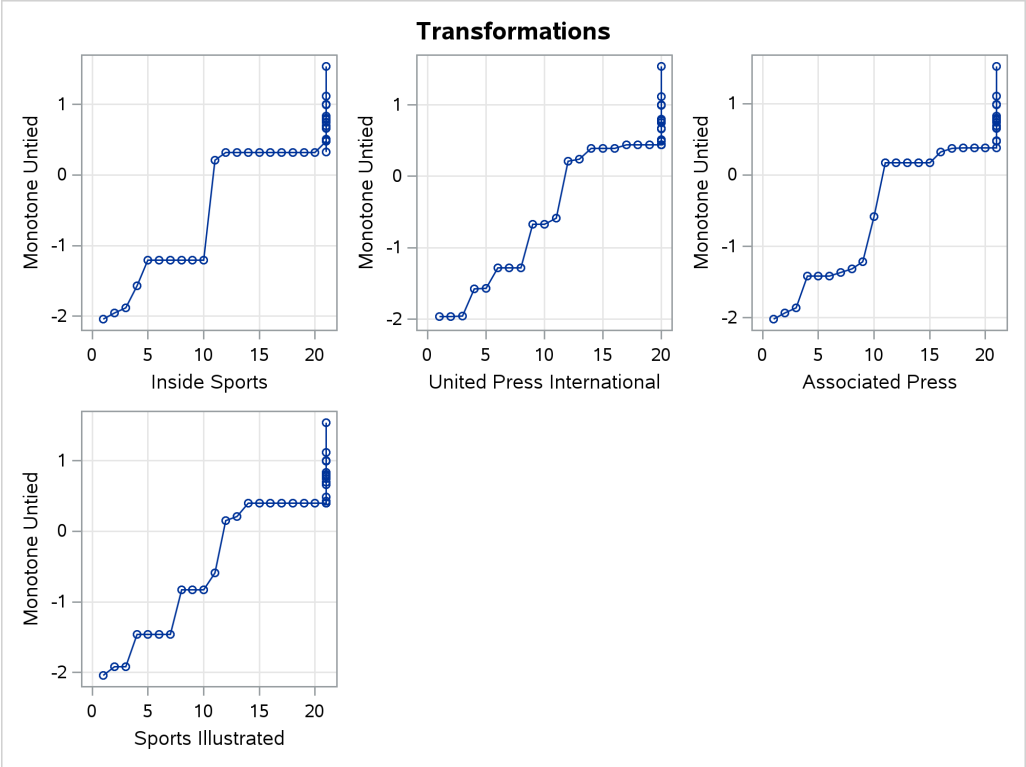

An alternative approach is to use the pairwise deletion option of the CORR procedure to compute the

correlation matrix and then use PROC PRINCOMP or PROC FACTOR to perform the principal component analysis. This approach has several disadvantages. The correlation matrix might not be positive semidefinite (PSD), an assumption required for principal component analysis. PROC PRINQUAL always produces a PSD correlation matrix. Even with pairwise deletion, PROC CORR removes the six observations that have only a single nonmissing value from this data set. Finally, it is still not possible to calculate scores on the principal components for those teams that have missing values.

You can compute the composite ranking by using PROC PRINCOMP and some preliminary data manipulations, similar to those discussed previously.

Chapter 98, "The PRINCOMP Procedure," contains an example where the average of the unused ranks in each poll is substituted for the missing values, and each observation is weighted by the number of nonmissing values. This method has much to recommend it. It is much faster and simpler than using PROC PRINQUAL. It is also much less prone to degeneracies and capitalization on chance. However, PROC PRINCOMP does not allow the nonmissing ranks to be monotonically transformed and the missing values untied to optimize fit.

PROC PRINQUAL monotonically transforms the observed ranks and estimates the missing ranks (within the constraints given previously) to account for almost 95 percent of the variance of the transformed data by just one dimension. PROC FACTOR is then used to report details of the principal component analysis of the transformed data. As shown by the Factor Pattern values in Output 99.2.3, nine of the ten news services have a correlation of 0.95 or larger with the scores on the first principal component after the data are optimally transformed. The scores are sorted and the composite ranking is displayed following the PROC FACTOR output. More confidence can be placed in the stability of the scores for teams that are ranked by the majority of the news services than in scores for teams that are seldom ranked.

The monotonic transformations are plotted for each of the ten news services. See Output 99.2.2. These plots show the values of the raw ranks (with the missing ranks replaced by the maximum rank plus one) versus the rescored (transformed) ranks. The transformations are the step functions that maximize the fit of the data to the principal component model. Smoother transformations could be found by using MSPLINE transformations, but MSPLINE transformations would not correctly handle the missing data problem.

The following statements perform the final analysis and produce Output 99.2.3:

```
* Perform the Final Principal Component Analysis;
proc factor nfactors=1 plots=scree;
   title4 'Principal Component Analysis';
   ods select factorpattern screeplot;
   var TCSN -- TSportsIllustrated;
run;

proc sort;
   by Prin1;
run;

* Display Scores on the First Principal Component;
proc print;
   title4 'Teams Ordered by Scores on First Principal Component';
   var School Prin1;
run;
```

Output 99.2.3 Principal Components of College Basketball Rankings

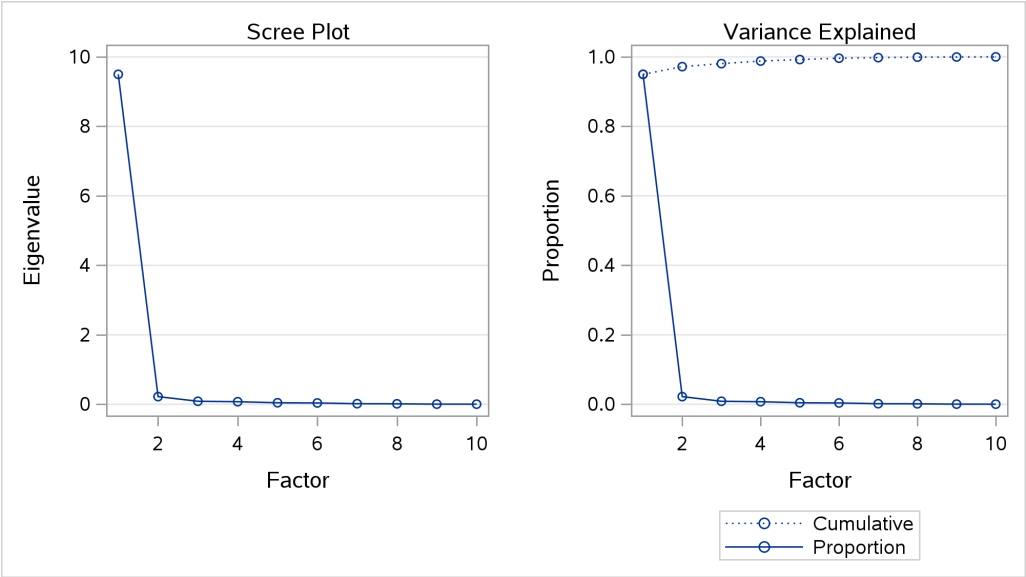

Output 99.2.3 *continued*

Factor Pattern		
		Factor1
TCSN	CSN Transformation	0.91136
TDurhamSun	DurhamSun Transformation	0.98887
TDurhamHerald	DurhamHerald Transformation	0.97402
TWashingtonPost	WashingtonPost Transformation	0.97408
TUSA_Today	USA_Today Transformation	0.98867
TSportMagazine	SportMagazine Transformation	0.95331
TInsideSports	InsideSports Transformation	0.98521
TUPI	UPI Transformation	0.98534
TAP	AP Transformation	0.99590
TSportsIllustrated	SportsIllustrated Transformation	0.98615

Output 99.2.3 *continued*

1985 Preseason College Basketball Rankings
Optimal Monotonic Transformation of Ranked Teams
with Constrained Estimation of Unranked Teams
Teams Ordered by Scores on First Principal Component

Obs	School	Prin1
1	Georgia Tech	-6.20315
2	UNC	-5.93314
3	Michigan	-5.71034
4	Kansas	-4.78699
5	Duke	-4.75896
6	Illinois	-4.19220
7	Georgetown	-4.02861
8	Louisville	-3.73087
9	Syracuse	-3.47497
10	Auburn	-1.78429
11	LSU	-0.35928
12	Memphis State	0.46737
13	Kentucky	0.63661
14	Notre Dame	0.71919
15	Navy	0.76187
16	UAB	0.98316
17	DePaul	1.09891
18	Oklahoma	1.12012
19	NC State	1.15144
20	UNLV	1.28766
21	Iowa	1.45260
22	Indiana	1.48123
23	Maryland	1.54935
24	Virginia	2.01385
25	Arkansas	2.02718
26	Washington	2.10878
27	Tennessee	2.27770
28	Virginia Tech	2.36103
29	St. Johns	2.37387
30	Montana	2.43502
31	UCLA	2.52481
32	Pittsburgh	3.00907
33	Old Dominion	3.03324
34	St. Joseph's	3.39259
35	Houston	4.69614

The ordinary PROC PRINQUAL missing data handling facilities do not work for these data because they do not constrain the missing data estimates properly. If you code the missing ranks as missing and specify linear transformations, then you can compute least squares estimates of the missing values without transforming the observed values. The first principal component then accounts for 92 percent of the variance after 20 iterations. However, Virginia Tech is ranked number 11 by its score even though it appeared in only one poll (Inside Sports ranked it number 13, anchoring it firmly in the middle). Specifying monotone transformations

is also inappropriate since they too allow unranked teams to move in between ranked teams.

With these data, the combination of monotone transformations and the freedom to score the missing ranks without constraint leads to degenerate transformations. PROC PRINQUAL tries to merge the 35 points into two points, producing a perfect fit in one dimension. There is evidence for this after 20 iterations when the Average Change, Maximum Change, and Criterion Change values are all increasing, instead of the more stable decreasing change rate seen in the analysis shown. The change rates all stop increasing after 41 iterations, and it is clear by 70 or 80 iterations that one component will account for 100 percent of the transformed variables variance after sufficient iteration. While this might seem desirable (after all, it is a perfect fit), you should, in fact, be on guard when this happens. Whenever convergence is slow, the rates of change increase, or the final data perfectly fit the model, the solution is probably degenerating because of too few constraints on the scorings.

PROC PRINQUAL can account for 100 percent of the variance by scoring Montana and UCLA with one positive value on all variables and scoring all the other teams with one negative value on all variables. This inappropriate analysis suggests that all ranked teams are equally good except for two teams that are less good. Both of these two teams are ranked by only one news service, and their only nonmissing rank is last in the poll. This accounts for the degeneracy.

References

Carroll, J. D. (1972). "Individual Differences and Multidimensional Scaling." In *Multidimensional Scaling: Theory and Applications in the Behavioral Sciences*, vol. 1, edited by R. N. Shepard, A. K. Romney, and S. B. Nerlove, 105–155. New York: Seminar Press.

De Boor, C. (1978). *A Practical Guide to Splines*. New York: Springer-Verlag.

De Leeuw, J. (1985). Personal communication, Leiden, Netherlands.

De Leeuw, J. (1986). *Regression with Optimal Scaling of the Dependent Variable*. Leiden, Netherlands: Department of Data Theory, University of Leiden.

Eckart, C., and Young, G. (1936). "The Approximation of One Matrix by Another of Lower Rank." *Psychometrika* 1:211–218.

Fisher, R. A. (1938). *Statistical Methods for Research Workers*. 10th ed. Edinburgh: Oliver & Boyd.

Gabriel, K. R. (1981). "Biplot Display of Multivariate Matrices for Inspection of Data and Diagnosis." In *Interpreting Multivariate Data*, edited by V. Barnett, 571–572. Chichester, UK: John Wiley & Sons.

Gifi, A. (1990). *Nonlinear Multivariate Analysis*. New York: John Wiley & Sons.

Goodnight, J. H. (1978). *The SWEEP Operator: Its Importance in Statistical Computing*. Technical Report R-106, SAS Institute Inc., Cary, NC.

Hotelling, H. (1933). "Analysis of a Complex of Statistical Variables into Principal Components." *Journal of Educational Psychology* 24:417–441, 498–520.

Kruskal, J. B. (1964). "Nonmetric Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis." *Psychometrika* 29:1–27.

Kruskal, J. B., and Shepard, R. N. (1974). "A Nonmetric Variety of Linear Factor Analysis." *Psychometrika* 38:123–157.

Sarle, W. S. (1984). Personal communication, Cary, NC.

Siegel, S. (1956). *Nonparametric Statistics*. New York: McGraw-Hill.

Smith, P. L. (1979). "Splines as a Useful and Convenient Statistical Tool." *American Statistician* 33:57–62.

Tenenhaus, M., and Vachette, J. L. (1977). "PRINQUAL: Un programme d'analyse en composantes principales d'un ensemble de variables nominales ou numériques [A program of principal component analysis of a set of nominal or numeric variables]." *Les Cahiers de Recherche* 68. CESA, Jouy-en-Josas, France.

Van Rijckevorsel, J. L. (1982). "Canonical Analysis with B-Splines." In *COMPUSTAT 1982, Part I*, edited by H. Caussinus, P. Ettinger, and R. Tomassone, 393–398. Vienna: Physica-Verlag.

Winsberg, S., and Ramsay, J. O. (1983). "Monotone Spline Transformations for Dimension Reduction." *Psychometrika* 48:575–595.

Young, F. W. (1981). "Quantitative Analysis of Qualitative Data." *Psychometrika* 46:357–388.

Young, F. W., Takane, Y., and de Leeuw, J. (1978). "The Principal Components of Mixed Measurement Level Multivariate Data: An Alternating Least Squares Method with Optimal Scaling Features." *Psychometrika* 43:279–281.

Chapter 100
The PROBIT Procedure

Contents

Overview: PROBIT Procedure **8260**
Getting Started: PROBIT Procedure **8261**
Estimating the Natural Response Threshold Parameter 8261
Syntax: PROBIT Procedure **8265**
PROC PROBIT Statement 8266
BY Statement 8271
CLASS Statement 8272
EFFECTPLOT Statement 8274
ESTIMATE Statement 8275
LSMEANS Statement 8276
LSMESTIMATE Statement 8277
MODEL Statement 8278
OUTPUT Statement 8284
SLICE Statement 8285
STORE Statement 8285
TEST Statement 8286
WEIGHT Statement 8286
Details: PROBIT Procedure **8286**
Missing Values 8286
Response Level Ordering 8286
Computational Method 8288
Distributions 8289
INEST= *SAS-data-set* 8289
Model Specification 8290
Lack-of-Fit Tests 8291
Rescaling the Covariance Matrix 8292
Tolerance Distribution 8292
Inverse Confidence Limits 8293
OUTEST= *SAS-data-set* 8294
XDATA= *SAS-data-set* 8294
Plot Options Superseded by ODS Graphics 8295
Displayed Output 8331
ODS Table Names 8333
ODS Graphics 8333
Examples: PROBIT Procedure **8337**
Example 100.1: Dosage Levels 8337

Example 100.2: Multilevel Response 8345
Example 100.3: Logistic Regression and Scoring New Data 8351
Example 100.4: An Epidemiology Study 8353
Example 100.5: Model Postfitting Analysis 8362
References **8364**

Overview: PROBIT Procedure

The PROBIT procedure calculates maximum likelihood estimates of regression parameters and the natural (or threshold) response rate for quantal response data from biological assays or other discrete event data. This includes probit, logit, ordinal logistic, and extreme value (or gompit) regression models.

Probit analysis developed from the need to analyze qualitative (dichotomous or polytomous) dependent variables within the regression framework. Many response variables are binary by nature (yes/no), while others are measured ordinally rather than continuously (degree of severity). Researchers have shown ordinary least squares (OLS) regression to be inadequate when the dependent variable is discrete (Collett 2003; Agresti 2002). Probit or logit analyses are more appropriate in this case.

The PROBIT procedure computes maximum likelihood estimates of the parameters $\boldsymbol{\beta}$ and C of the probit equation by using a modified Newton-Raphson algorithm. When the response Y is binary, with values 0 and 1, the probit equation is

$$p = \Pr(Y = 0) = C + (1 - C)F(\mathbf{x}'\boldsymbol{\beta})$$

where

$\boldsymbol{\beta}$ is a vector of parameter estimates

F is a cumulative distribution function (normal, logistic, or extreme value)

$\mathbf{x}$ is a vector of explanatory variables

p is the probability of a response

C is the natural (threshold) response rate

Notice that PROC PROBIT, by default, models the probability of the *lower* response levels. The choice of the distribution function F (normal for the probit model, logistic for the logit model, and extreme value or Gompertz for the gompit model) determines the type of analysis. For most problems, there is relatively little difference between the normal and logistic specifications of the model. Both distributions are symmetric about the value zero. The extreme value (or Gompertz) distribution, however, is not symmetric, approaching 0 on the left more slowly than it approaches 1 on the right. You can use the extreme value distribution where such asymmetry is appropriate.

For ordinal response models, the response, Y, of an individual or an experimental unit can be restricted to one of a (usually small) number, $k + 1 (k \geq 1)$, of ordinal values, denoted for convenience by $1, \ldots, k, k + 1$. For example, the severity of coronary disease can be classified into three response categories as 1=no disease, 2=angina pectoris, and 3=myocardial infraction. The PROBIT procedure fits a common slopes cumulative

model, which is a parallel-lines regression model based on the cumulative probabilities of the response categories rather than on their individual probabilities. The cumulative model has the form

$$\Pr(Y \le 1 \mid \mathbf{x}) = F(\mathbf{x}'\boldsymbol{\beta})$$

$$\Pr(Y \le i \mid \mathbf{x}) = F(\alpha_i + \mathbf{x}'\boldsymbol{\beta}), \quad 2 \le i \le k$$

where $\alpha_2, \ldots, \alpha_k$ are $k - 1$ intercept parameters. By default, the covariate vector $\mathbf{x}$ contains an overall intercept term.

You can set or estimate the natural (threshold) response rate C. Estimation of C can begin either from an initial value that you specify or from the rate observed in a control group. By default, the natural response rate is fixed at zero.

An observation in the data set analyzed by the PROBIT procedure might contain the response and explanatory values for one subject. Alternatively, it might provide the number of observed events from a number of subjects at a particular setting of the explanatory variables. In this case, PROC PROBIT models the probability of an event.

The PROBIT procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the PROBIT procedure, see the section "ODS Graphics" on page 8333.

Getting Started: PROBIT Procedure

The following example illustrates how you can use the PROBIT procedure to compute the threshold response rate and regression parameter estimates for quantal response data.

Estimating the Natural Response Threshold Parameter

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/probitgs.sas.)

Suppose you want to test the effect of a drug at 12 dosage levels. You randomly divide 180 subjects into 12 groups of 15—one group for each dosage level. You then conduct the experiment and, for each subject, record the presence or absence of a positive response to the drug. You summarize the data by counting the number of subjects responding positively in each dose group. Your data set is as follows:

```
data study;
   input Dose Respond @@;
   Number = 15;
   datalines;
0     3   1.1   4   1.3   4   2.0   3   2.2   5   2.8   4
3.7   5   3.9   9   4.4   8   4.8  11   5.9  12   6.8  13
;
```

The variable dose represents the amount of drug administered. The first group, receiving a dose level of 0, is the control group. The variable number represents the number of subjects in each group. All groups are equal

in size; hence, number has the value 15 for all observations. The variable respond represents the number of subjects responding to the associated drug dosage.

You can model the probability of positive response as a function of dosage by using the following statements:

```
ods graphics on;

proc probit data=study log10 optc plots=(predpplot ippplot);
   model respond/number=dose;
   output out=new p=p_hat;
run;
```

The DATA= option specifies that PROC PROBIT analyze the SAS data set study. The LOG10 option replaces the first continuous independent variable (dose) with its common logarithm. The OPTC option estimates the natural response rate. When you use the LOG10 option with the OPTC option, any observations with a dose value less than or equal to zero are used in the estimation as a control group.

The PLOTS= option in the PROC PROBIT statement, together with the ODS GRAPHICS statement, requests two plots for the estimated probability values and dosage levels. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the PROBIT procedure, see the section "ODS Graphics" on page 8333.

The MODEL statement specifies a proportional response by using the variables respond and number in *events/trials* syntax. The variable dose is the stimulus or explanatory variable.

The OUTPUT statement creates a new data set, new, that contains all the variables in the original data set, and a new variable, p_hat, that represents the predicted probabilities.

The results from this analysis are displayed in the following figures.

Figure 100.1 displays background information about the model fit. Included are the name of the input data set, the response variables used, and the number of observations, events, and trials. The last line in Figure 100.1 shows the final value of the log-likelihood function.

Figure 100.2 displays the table of parameter estimates for the model. The parameter C, which is the natural response threshold or the proportion of individuals responding at zero dose, is estimated to be 0.2409. Since both the intercept and the slope coefficient have significant p-values (0.0020, 0.0010), you can write the model for

$$\Pr(\text{response}) = C + (1 - C)F(\mathbf{x}'\beta)$$

as

$$\Pr(\text{response}) = 0.2409 + 0.7591(\Phi(-4.1439 + 6.2308 \times \log_{10}(\text{dose})))$$

where Φ is the normal cumulative distribution function.

Finally, PROC PROBIT specifies the resulting tolerance distribution by providing the mean MU and scale parameter SIGMA as well as the covariance matrix of the distribution parameters in Figure 100.3.

Figure 100.1 Model Fitting Information for the PROBIT Procedure

The Probit Procedure

Model Information	
Data Set	WORK.STUDY
Events Variable	Respond
Trials Variable	Number
Number of Observations	12
Number of Events	81
Number of Trials	180
Number of Events In Control Group	3
Number of Trials In Control Group	15
Name of Distribution	Normal
Log Likelihood	-104.3945783

Figure 100.2 Model Parameter Estimates for the PROBIT Procedure

Analysis of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-4.1438	1.3415	-6.7731	-1.5146	9.54	0.0020
Log10(Dose)	1	6.2308	1.8996	2.5076	9.9539	10.76	0.0010
_C_	1	0.2409	0.0523	0.1385	0.3433		

Figure 100.3 Tolerance Distribution Estimates for the PROBIT Procedure

Estimated Covariance Matrix for Tolerance Parameters

	MU	SIGMA	_C_
MU	0.001158	-0.000493	0.000954
SIGMA	-0.000493	0.002394	-0.000999
_C_	0.000954	-0.000999	0.002731

The PLOTS=PREDPPLOT option creates the plot in Figure 100.4, showing the relationship between dosage level, observed response proportions, and estimated probability values. The shaded region is the pointwise confidence band for the fitted probabilities, and a reference line is plotted at the estimated threshold value of 0.24.

The PLOTS=IPPPLOT option creates the plot in Figure 100.5, showing the inverse relationship between dosage level and observed response proportions/estimated probability values. The shaded region represents the pointwise fiducial limits for the predicted values of the dose variable, and a reference line is also plotted at the estimated threshold value of 0.24.

The two plot options can be put together with the PLOTS= option, as shown in the PROC PROBIT statement.

Figure 100.4 Plot of Observed and Fitted Probabilities versus Dose Level

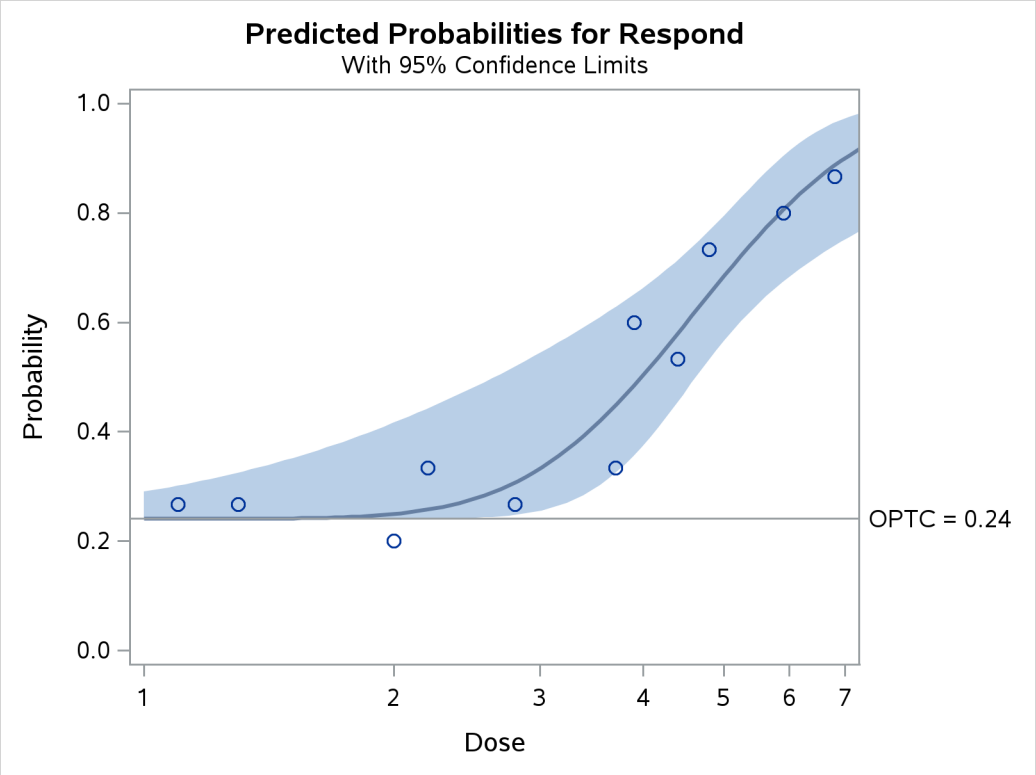

Figure 100.5 Inverse Predicted Probability Plot with Fiducial Limits

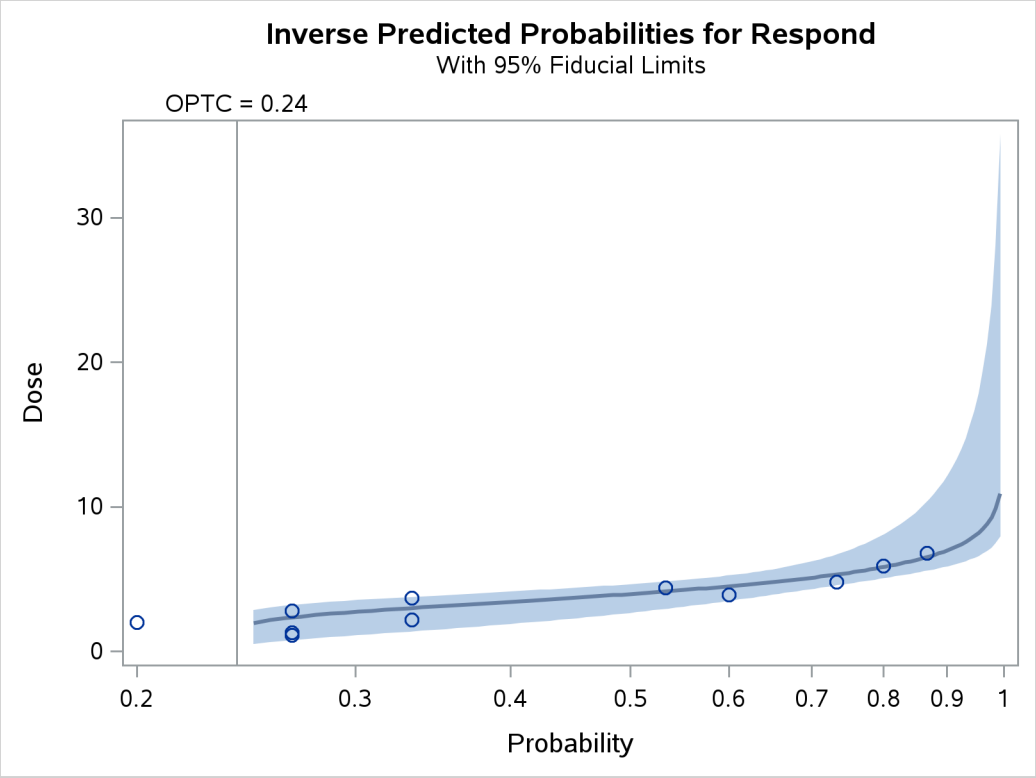

Syntax: PROBIT Procedure

The following statements are available in the PROBIT procedure:

```
PROC PROBIT <options> ;
    BY variables ;
    CDFPLOT <VAR=variable> <options> ;
    CLASS variables <(options)> ... <variable <(options)>> </options> ;
    ESTIMATE <'label'> estimate-specification <(divisor=n)>
                <, ... <'label'> estimate-specification <(divisor=n)>> </options> ;
    EFFECTPLOT <plot-type <(plot-definition-options)>> </options> ;
    INSET <keyword-list> </options> ;
    IPPPLOT <VAR=variable> <options> ;
    LPREDPLOT <VAR=variable> <options> ;
    LSMEANS <model-effects> </options> ;
    LSMESTIMATE model-effect <'label'> values <(divisor=n)>
                    <, ... <'label'> values <(divisor=n)>> </options> ;
    MODEL response <(response_options)> = independents </options> ;
    OUTPUT <OUT=SAS-data-set> <options> ;
    PREDPPLOT <VAR=variable> <options> ;
    SLICE model-effect </options> ;
    STORE <OUT=>item-store-name </ LABEL='label'> ;
    TEST <model-effects> </options> ;
    WEIGHT variable ;
```

A MODEL statement is required. Only a single MODEL statement can be used with one invocation of the PROBIT procedure. If multiple MODEL statements are present, only the last one is used. Main effects and higher-order terms can be specified in the MODEL statement, as in the GLM procedure. If a CLASS statement is used, it must precede the MODEL statement.

The CDFPLOT, INSET, IPPPLOT, LPREDPLOT, and PREDPPLOT statements are used to produce traditional graphical output. Use the PLOTS= option to create ODS Graphics.

The ESTIMATE, EFFECTPLOT, LSMEANS, LSMESTIMATE, SLICE, STORE, and TEST statements are common to many procedures. Summary descriptions of functionality and syntax for these statements are also given after the PROC PROBIT statement in alphabetical order, and full documentation about them is available in Chapter 20, “Shared Concepts and Topics.”

PROC PROBIT Statement

PROC PROBIT < *options* > **;**

The PROC PROBIT statement invokes the PROBIT procedure. Table 100.1 summarizes the *options* available in the PROC PROBIT statement.

Table 100.1 PROC PROBIT Statement Options

Option	Description
COVOUT	Writes the parameter estimate covariance matrix to the OUTEST= data set
C=	Controls how the natural response is handled
DATA=	Specifies the SAS data set to be used
HPROB=	Specifies a minimum probability level for the Pearson's chi-square
INEST=	Specifies an input SAS data set that contains initial estimates
INVERSECL	Computes confidence limits
LACKFIT	Performs two goodness-of-fit tests
LOG	Replaces the first continuous independent variable with its natural logarithm
LOG10	Replaces the first continuous independent variable with log to the base 10
NAMELEN=	Specifies the length of effect names to be *n* characters
NOPRINT	Suppresses the display of all output including graphics
OPTC	Controls how the natural response is handled
ORDER=	Specifies the sort order for the levels of the classification variables
OUTEST=	Specifies a SAS data set to contain the parameter estimates
PLOT | PLOTS	Controls the plots produced though ODS Graphics
XDATA=	Specifies an input SAS data set that contains values for all the independent variables

You can specify the following *options* in the PROC PROBIT statement.

COVOUT

writes the parameter estimate covariance matrix to the OUTEST= data set.

C=*rate*

OPTC

controls how the natural response is handled. Specify the OPTC option to request that the natural response rate *C* be estimated. Specify the C=*rate* option to set the natural response rate or to provide the initial estimate of the natural response rate. The natural response rate value must be a number between 0 and 1.

- If you specify neither the OPTC nor the C= option, a natural response rate of zero is assumed.
- If you specify both the OPTC and the C= option, the C= option should be a reasonable initial estimate of the natural response rate. For example, you could use the ratio of the number of responses to the number of subjects in a control group.
- If you specify the C= option but not the OPTC option, the natural response rate is set to the specified value and not estimated.

- If you specify the OPTC option but not the C= option, PROC PROBIT's action depends on the response variable, as follows:
  - If you specify either the LN or LOG10 option and some subjects have the first independent variable (dose) values less than or equal to zero, these subjects are treated as a control group. The initial estimate of C is then the ratio of the number of responses to the number of subjects in this group.
  - If you do not specify the LN or LOG10 option or if there is no control group, then one of the following occurs:
    - If all responses are greater than zero, the initial estimate of the natural response rate is the minimal response rate (the ratio of the number of responses to the number of subjects in a dose group) across all dose levels.
    - If one or more of the responses is zero (making the response rate zero in that dose group), the initial estimate of the natural rate is the reciprocal of twice the largest number of subjects in any dose group in the experiment.

DATA=*SAS-data-set*

specifies the SAS data set to be used by PROC PROBIT. By default, the procedure uses the most recently created SAS data set.

HPROB=*p*

specifies a minimum probability level for the Pearson's chi-square to indicate a good fit. The default value is 0.10. The LACKFIT option must also be specified for this option to have any effect. For Pearson's goodness-of-fit chi-square values with probability greater than the HPROB= value, the fiducial limits, if requested with the INVERSECL option, are computed by using a critical value of 1.96. For chi-square values with probability less than the value of the HPROB= option, the critical value is a 0.95 two-sided quantile value taken from the t distribution with degrees of freedom equal to $(k-1) \times m - q$, where k is the number of levels for the response variable, m is the number of different sets of independent variable values, and q is the number of parameters fit in the model. Note that the HPROB= option can also appear in the MODEL statement.

INEST=*SAS-data-set*

specifies an input SAS data set that contains initial estimates for all the parameters in the model. See the section "INEST= *SAS-data-set*" on page 8289 for a detailed description of the contents of the INEST= data set.

INVERSECL< (PROB=*rates***) >**

computes confidence limits for the values of the first continuous independent variable (such as dose) that yield selected response rates. You can optionally specify a list of response rates as *rates*. The response rates must be between zero and one, and can be a list separated by blanks, commas, or in the form of a DO list.

For example,

```
PROB = .1 TO .9 by .1
PROB = .1 .2 .3 .4
PROB = .01, .25, .75, .9
```

are valid lists of response rates.

If the algorithm fails to converge (this can happen when *C* is nonzero), missing values are reported for the confidence limits. See the section "Inverse Confidence Limits" on page 8293 for details. Note that the INVERSECL option can also appear in the MODEL statement.

LACKFIT

performs two goodness-of-fit tests (a Pearson's chi-square test and a log-likelihood ratio chi-square test) for the fitted model.

To compute the test statistics, proper grouping of the observations into subpopulations is needed. You can use the AGGREGATE or AGGREGATE= option for this end. See the entry for the AGGREGATE and AGGREGATE= options under the MODEL statement. If neither AGGREGATE nor AGGREGATE= is specified, PROC PROBIT assumes each observation is from a separate subpopulation and computes the goodness-of-fit test statistics only for the *events/trials* syntax.

NOTE: This test is not appropriate if the data are very sparse, with only a few values at each set of the independent variable values.

If the Pearson's chi-square test statistic is significant, then the covariance estimates and standard error estimates are adjusted. See the section "Lack-of-Fit Tests" on page 8291 for a description of the tests. Note that the LACKFIT option can also appear in the MODEL statement.

LOG

LN

analyzes the data by replacing the first continuous independent variable with its natural logarithm. This variable is usually the level of some treatment such as dosage. In addition to the usual output given by the INVERSECL option, the estimated dose values and 95% fiducial limits for dose are also displayed. If you specify the OPTC option, any observations with a dose value less than or equal to zero are used in the estimation as a control group. If you do not specify the OPTC option with the LOG or LN option, then any observations with the first continuous independent variable values less than or equal to zero are ignored.

LOG10

specifies an analysis like that of the LN or LOG option, except that the common logarithm (log to the base 10) of the dose value is used rather than the natural logarithm.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 200. The default length is 20 characters.

NOPRINT

suppresses the display of all output including graphics. Note that this option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OPTC

controls how the natural response is handled. See the description of the C= option on page 8266 for details.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

This order also applies to the levels of the response variable. Response level ordering is important because PROC PROBIT always models the probability of response levels at the beginning of the ordering. See the section "Response Level Ordering" on page 8286 for further details.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUTEST=*SAS-data-set*

specifies a SAS data set to contain the parameter estimates and, if the COVOUT option is specified, their estimated covariances. If you omit this option, the output data set is not created. The contents of the data set are described in the section "OUTEST= *SAS-data-set*" on page 8294.

PLOT | PLOTS < =*plot-request* **>**

PLOT | PLOTS < =(*plot-request* **< . . .** *plot-request* **>) >**

specifies options that control details of the plots created by ODS Graphics. These plots are related to a dose variable, which is identified as the first single continuous independent variable in the MODEL statement. If there are interaction terms with this variable in the model, the PROBIT procedure will not produce any plot.

You can specify more than one plot request within the parentheses after PLOTS=. For a single plot request, you can omit the parentheses.

ODS Graphics must be enabled before plots can be requested. For example:

```
proc probit plots=predplot;
   model r/n = dose;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

There are four options that you can use to request traditional ODS Graphics: CDFPLOT, IPPPLOT, LPREDPLOT, and PREDPPLOT. Some of these options apply only to either the binomial model or the multinomial model. Table 100.2 shows the availability of these statements for different models.

Table 100.2 PLOTS= Option Availability

Statement	Binomial	Multinomial
CDFPLOT	No	Yes
IPPPLOT	Yes	No
LPREDPLOT	Yes	Yes
PREDPPLOT	Yes	Yes

The CDFPLOT option creates a plot of predicted cumulative distribution function of the multinomial response variable as a function of a single continuous independent variable (dose variable). The PREDPPLOT option creates a plot that shows the relationship between dosage level, observed response proportions, and estimated probability values. The IPPPLOT statement creates a similar plot. The LPREDPLOT statement creates a linear predictor plot.

The following *plot-requests* are available.

ALL

creates all appropriate plots.

CDFPLOT< (LEVEL=(*character-list*)) >

requests the plot of predicted cumulative distribution function (CDF) of the multinomial response variable as a function of a single continuous independent variable (dose variable). This single continuous independent variable must be the first single continuous independent variable listed in the MODEL statement. You can request this plot only with a multinomial model.

The LEVEL= suboption specifies the levels of the multinomial response variable for which the CDF curves are requested. There are $k - 1$ curves for a k-level multinomial response variable (for the highest level, it is the constant line 1). You can specify any of them to be plotted by the LEVEL= suboption.

IPPPLOT

requests the inverse plot of the predicted probability against the first single continuous variable (dose variable) in the MODEL statement for the binomial model. You can request this plot only with a binomial model. The confidence limits for the predicted values of the dose variable are the computed fiducial limits, not the inverse of the confidence limits of the predicted probabilities. See the section "Inverse Confidence Limits" on page 8293 for more details.

LPREDPLOT< (LEVEL=(*character-list*)) >
requests the plot of the linear predictor $\mathbf{x}'\mathbf{b}$ against the first single continuous variable (dose variable) in the MODEL statement for either the binomial model or the multinomial model. The confidence limits for the predicted values are available only for the binomial model.

For the multinomial model, you can use the LEVEL= suboption to specify the levels for which the linear predictor lines are plotted.

NONE
suppresses all plots.

PREDPPLOT< (LEVEL=(*character-list*)) >
requests the plot of the predicted probability against the first single continuous variable (dose variable) in the MODEL statement for both the binomial model and the multinomial model. Confidence limits are available only for the binomial model.

For the multinomial model, you can use the LEVEL= suboption to specify the levels for which the linear predictor lines are plotted.

XDATA=*SAS-data-set*
specifies an input SAS data set that contains values for all the independent variables in the MODEL statement and variables in the CLASS statement. If there are covariates specified in a MODEL statement, you specify fixed values for the effects in the MODEL statement by the XDATA= data set when predicted values and/or fiducial limits for a single continuous variable (dose variable) are required. These specified values for the effects in the MODEL statement are also used for generating plots. See the section "XDATA= *SAS-data-set*" on page 8294 for a detailed description of the contents of the XDATA= data set.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC PROBIT to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the PROBIT procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *global-options* > ;

The CLASS statement names the classification variables to be used as explanatory variables in the analysis.

The CLASS statement must precede the MODEL statement. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*. You can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING

DESC

reverses the sort order of the classification variable. If both the DESCENDING and ORDER= options are specified, PROC PROBIT orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (., ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values for the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so the ORDER= option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC PROBIT interprets values of the ORDER= option.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

PARAM=GLM

specifies the parameterization method for the classification variable or variables. The default is GLM parameterization. Only GLM parameterization is currently available. The REF= option in the CLASS statement indirectly determines the reference level for a singular GLM parameterization through the order of levels.

REF=*'level'* | *keyword*

For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*.

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n* >

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

EFFECTPLOT Statement

EFFECTPLOT < *plot-type* < *(plot-definition-options)* > > < / *options* > **;**

The EFFECTPLOT statement produces a display of the fitted model and provides options for changing and enhancing the displays. Table 100.3 describes the available *plot-types* and their *plot-definition-options*.

Table 100.3 *Plot-Types* and *Plot-Definition-Options*

Plot-Type and Description	Plot-Definition-Options
BOX Displays a box plot of continuous response data at each level of a CLASS effect, with predicted values superimposed and connected by a line. This is an alternative to the INTERACTION *plot-type*.	PLOTBY= variable or CLASS effect X= CLASS variable or effect
CONTOUR Displays a contour plot of predicted values against two continuous covariates.	PLOTBY= variable or CLASS effect X= continuous variable Y= continuous variable
FIT Displays a curve of predicted values versus a continuous variable.	PLOTBY= variable or CLASS effect X= continuous variable
INTERACTION Displays a plot of predicted values (possibly with error bars) versus the levels of a CLASS effect. The predicted values are connected with lines and can be grouped by the levels of another CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= CLASS variable or effect
MOSAIC Displays a mosaic plot of predicted values using up to three CLASS effects.	PLOTBY= variable or CLASS effect X= CLASS effects
SLICEFIT Displays a curve of predicted values versus a continuous variable grouped by the levels of a CLASS effect.	PLOTBY= variable or CLASS effect SLICEBY= variable or CLASS effect X= continuous variable

For full details about the syntax and options of the EFFECTPLOT statement, see the section "EFFECTPLOT Statement" on page 447 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 100.4 summarizes the *options* available in the ESTIMATE statement.

Table 100.4 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the $\mathbf{L}$ matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement"

on page 475 in Chapter 20, "Shared Concepts and Topics."

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 100.5 summarizes the *options* available in the LSMEANS statement.

Table 100.5 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers

Table 100.5 *continued*

Option	Description
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 100.6 summarizes the *options* available in the LSMESTIMATE statement.

Table 100.6 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference

Table 100.6 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint *F* or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

< *label*: > **MODEL** *response*< (*response_options*) > **=** *effects* < / *options* > **;**

< *label*: > **MODEL** *events/trials* **=** *effects* < / *options* > **;**

The MODEL statement names the variables used as the response and the independent variables. Additionally, you can specify the distribution used to model the response, as well as other options. Only a single MODEL statement can be used with one invocation of the PROBIT procedure. If multiple MODEL statements are present, only the last is used. Main effects and interaction terms can be specified in the MODEL statement, as in the GLM procedure.

The optional *label*, which must be a valid SAS name, is used to label output from the matching MODEL statement.

The *response* can be a single variable whose value indicates the level of the observed response. The *response* variable can be numeric or character. For example, the *response* might be a variable called Symptoms that takes the value 'None,' 'Mild,' or 'Severe.' Variable options specific to the response variable can be specified in parentheses immediately after the response variable. Identifying the event level for binomial responses and ordering of response levels for multinomial responses is critical in these models. You can use the response variable options to do this.

Alternatively, the response can be specified as a pair of variable names separated by a slash (/). The value of the first variable, *events*, is the number of positive responses (or events). The value of the second variable,

trials, is the number of trials. Both variables must be numeric and nonnegative, and the ratio of the first variable value to the second variable value must be between 0 and 1, inclusive. For example, the variables might be hits, a variable containing the number of hits for a baseball player, and AtBats, a variable containing the number of times at bat. A model for hitting proportion (batting average) as a function of age could be specified as

```
model hits/AtBats=age;
```

The *effects* following the equal sign are the covariates in the model. Higher-order effects, such as interactions and nested terms, are allowed in the list, as in the GLM procedure. Variable names and combinations of variable names representing higher-order terms are allowed to appear in this list. Classification variables can be used as effects, and indicator variables are generated for the class levels. If you do not specify any covariates following the equal sign, an intercept-only model is fit.

Table 100.7 summarizes the *options* available in the MODEL statement.

Table 100.7 MODEL Statement Options

Option	Description
AGGREGATE	Specifies the subpopulations
ALPHA=	Sets the significance level
CONVERGE=	Specifies the convergence criterion
CORRB	Displays the estimated correlation matrix
COVB	Displays the estimated covariance matrix
DESCENDING	Reverses the order of the response categories
DISTRIBUTION=	Specifies the cumulative distribution function
EVENT=	Specifies the event category for the binary response model
HPROB=	Specifies a minimum probability level
INITIAL=	Sets initial values for the parameters
INTERCEPT=	Initializes the intercept parameter
INVERSECL	Computes confidence limits
ITPRINT	Displays the iteration history, the final evaluation of the gradient, and the second derivative matrix
LACKFIT	Performs two goodness-of-fit tests
MAXITER=	Specifies the maximum number of iterations
NOINT	Fits a model with no intercept parameter
ORDER=	Specifies the sort order for the levels of the response variable
REFERENCE=	Specifies the reference category for the binary response model
SCALE=	Specifies the method for estimating the dispersion parameter
SINGULAR=	Specifies the singularity criterion

Response Variable Options

DESCENDING | DESC

reverses the order of the response categories. If both the DESCENDING and ORDER= options are specified, PROC PROBIT orders the response categories according to the ORDER= option and then reverses that order. See the section "Response Level Ordering" on page 8286 for more detail.

EVENT='*category*' | *keyword*

specifies the event category for the binary response model. PROC PROBIT models the probability of the event category. The EVENT= option has no effect when there are more than two response categories. You can specify the value (formatted if a format is applied) of the event category in quotation marks, or you can specify one of the following *keywords*.

FIRST

designates the first ordered category as the event.

LAST

designates the last ordered category as the event.

By default, EVENT=FIRST.

One of the most common sets of response levels is {0,1}, where 1 represents the event for which the probability is to be modeled. Consider the example where Y takes the values 1 and 0 for event and nonevent, respectively, and Exposure is the explanatory variable. To specify the value 1 as the event category, use the following MODEL statement:

```
model Y(event='1') = Exposure;
```

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the response variable. The following table displays the available ORDER= options:

ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | *keyword*

REF='*category*' | *keyword*

specifies the reference category for the binary response model. Specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotation marks, or you can specify one of the following *keywords*:

FIRST designates the first ordered category as the reference.

LAST designates the last ordered category as the reference.

By default, REF=LAST.

Model Options

The following *options* are available in the MODEL statement.

AGGREGATE

AGGREGATE=*variable-list*

specifies the subpopulations on which the Pearson's chi-square test statistic and the log-likelihood ratio chi-square test statistic (deviance) are calculated if the LACKFIT option is specified. See the section "Rescaling the Covariance Matrix" on page 8292 for details of Pearson's chi-square and deviance calculations.

Observations with common values in the given list of variables are regarded as coming from the same subpopulation. Variables in the list can be any variables in the input data set. Specifying the AGGREGATE option is equivalent to specifying the AGGREGATE= option with a variable list that includes all independent variables in the MODEL statement. The PROBIT procedure sorts the input data set according to the variables specified in this list. Information for the sorted data set is reported in the "Response-Covariate Profile" table.

The deviance and Pearson's goodness-of-fit statistics are calculated if the LACKFIT option is specified in the MODEL statement. The calculated results are reported in the "Goodness-of-Fit" table. If the Pearson's chi-square test is significant with the test level specified by the HPROB= option, the fiducial limits, if required with the INVERSECL option in the MODEL statement, are modified (see the section "Inverse Confidence Limits" on page 8293 for details). Also, the covariance matrix is rescaled by the dispersion parameter when the SCALE= option is specified.

ALPHA=*value*

sets the significance level for the confidence intervals for regression parameters, fiducial limits for the predicted values, and confidence intervals for the predicted probabilities. The value must be between 0 and 1. The default value is ALPHA=0.05.

CONVERGE=*value*

specifies the convergence criterion. Convergence is declared when the maximum change in the parameter estimates between Newton-Raphson steps is less than the value specified. The change is a relative change if the parameter is greater than 0.01 in absolute value; otherwise, it is an absolute change.

By default, CONVERGE=1.0E–8.

CORRB
displays the estimated correlation matrix of the parameter estimates.

COVB
displays the estimated covariance matrix of the parameter estimates.

DISTRIBUTION=*distribution-type*
DIST=*distribution-type*
D=*distribution-type*
specifies the cumulative distribution function used to model the response probabilities. The distributions are described in the section "Details: PROBIT Procedure" on page 8286. Valid values for *distribution-type* are as follows:

NORMAL the normal distribution for the probit model

LOGISTIC the logistic distribution for the logit model

EXTREMEVALUE | EXTREME | GOMPERTZ the extreme value, or Gompertz distribution for the gompit model

By default, DISTRIBUTION=NORMAL.

HPROB=*p*
specifies a minimum probability level for the Pearson's chi-square to indicate a good fit. The default value is 0.10. The LACKFIT option must also be specified for this option to have any effect. For Pearson's goodness-of-fit chi-square values with probability greater than the HPROB= value, the fiducial limits, if requested with the INVERSECL option, are computed by using a critical value of 1.96. For chi-square values with probability less than the value of the HPROB= option, the critical value is a 0.95 two-sided quantile value taken from the t distribution with degrees of freedom equal to $(k-1) \times m - q$, where k is the number of levels for the response variable, m is the number of different sets of independent variable values, and q is the number of parameters fit in the model. If you specify the HPROB= option in both the PROC PROBIT and MODEL statements, the MODEL statement option takes precedence.

INITIAL=*values*
sets initial values for the parameters in the model other than the intercept. The values must be given in the order in which the variables are listed in the MODEL statement. If some of the independent variables listed in the MODEL statement are classification variables, then there must be as many values given for that variable as there are classification levels minus 1. The INITIAL option can be specified as follows.

Type of List	Specification
List separated by blanks	`initial=3 4 5`
List separated by commas	`initial=3,4,5`

By default, all parameters have initial estimates of zero.

NOTE: The INITIAL= option is overwritten by the INEST= option in the PROC PROBIT statement.

INTERCEPT=*value*

initializes the intercept parameter to *value*. By default, INTERCEPT=0.

INVERSECL< (PROB=*rates***) >**

computes confidence limits for the values of the first continuous independent variable (such as dose) that yield selected response rates. You can optionally specify a list of response rates as *rates*. The response rates must be between zero and one; they can be a list separated by blanks, commas, or in the form of a DO list. For example, the following expressions are all valid lists of response rates:

```
PROB = .1 TO .9 by .1
PROB = .1 .2 .3 .4
PROB = .01, .25, .75, .9
```

If the algorithm fails to converge (this can happen when C is nonzero), missing values are reported for the confidence limits. See the section "Inverse Confidence Limits" on page 8293 for details.

ITPRINT

displays the iteration history, the final evaluation of the gradient, and the second derivative matrix (Hessian).

LACKFIT

performs two goodness-of-fit tests (a Pearson's chi-square test and a log-likelihood ratio chi-square test) for the fitted model.

To compute the test statistics, proper grouping of the observations into subpopulations is needed. You can use the AGGREGATE or AGGREGATE= option for this purpose. See the entry for the AGGREGATE and AGGREGATE= options under the MODEL statement. If neither AGGREGATE nor AGGREGATE= is specified, PROC PROBIT assumes each observation is from a separate subpopulation and computes the goodness-of-fit test statistics only for the *events/trials* syntax.

NOTE: This test is not appropriate if the data are very sparse, with only a few values at each set of the independent variable values.

If the Pearson's chi-square test statistic is significant, then the covariance estimates and standard error estimates are adjusted. See the section "Lack-of-Fit Tests" on page 8291 for a description of the tests. Note that the LACKFIT option can also appear in the PROC PROBIT statement. See the section "PROC PROBIT Statement" on page 8266 for details.

MAXITER=*value*

MAXIT=*value*

specifies the maximum number of iterations to be performed in estimating the parameters. By default, MAXITER=50.

NOINT

fits a model with no intercept parameter. If the INTERCEPT= option is also specified, the intercept is fixed at the specified value; otherwise, it is set to zero. This is most useful when the response is binary. When the response has k levels, then $k - 1$ intercept parameters are fit. The NOINT option sets the intercept parameter corresponding to the lowest response level equal to zero. A Lagrange multiplier, or score, test for the restricted model is computed when the NOINT option is specified.

SCALE=*scale*

enables you to specify the method for estimating the dispersion parameter. To correct for overdispersion or underdispersion, the covariance matrix is multiplied by the estimate of the dispersion parameter. Valid values for *scale* are as follows:

D | DEVIANCE specifies that the dispersion parameter be estimated by the deviance divided by its degrees of freedom.

P | PEARSON specifies that the dispersion parameter be estimated by the Pearson's chi-square statistic divided by its degrees of freedom. This is set as the default method for estimating the dispersion parameter.

You can use the AGGREGATE= option to define the subpopulations for calculating the Pearson's chi-square statistic and the deviance.

The "Goodness-of-Fit " table includes the Pearson's chi-square statistic, the deviance, their degrees of freedom, the ratio of each statistic divided by its degrees of freedom, and the corresponding p-value.

SINGULAR=*value*

specifies the singularity criterion for determining linear dependencies in the set of independent variables. The sum of squares and crossproducts matrix of the independent variables is formed and swept. If the relative size of a pivot becomes less than the value specified, then the variable corresponding to the pivot is considered to be linearly dependent on the previous set of variables considered. By default, *value*=1E–12.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set keyword***=***name . . . keyword***=***name* > **;**

The OUTPUT statement creates a new SAS data set containing all variables in the input data set and, optionally, the fitted probabilities, the estimate of $\mathbf{x}'\boldsymbol{\beta}$, and the estimate of its standard error. Estimates of the probabilities, $\mathbf{x}'\boldsymbol{\beta}$, and the standard errors are computed for observations with missing response values as long as the values of all the explanatory variables are nonmissing. This enables you to compute these statistics for additional settings of the explanatory variables that are of interest but for which responses are not observed.

You can specify multiple OUTPUT statements. Each OUTPUT statement creates a new data set and applies only to the preceding MODEL statement. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

Details on the specifications in the OUTPUT statement are as follows:

keyword=name specifies the statistics to include in the output data set and assigns names to the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), an equal sign, and the variable to contain the statistic.

The *keywords* allowed and the statistics they represent are as follows:

PROB | P cumulative probability estimates

$$p = C + (1 - C)F(a_j + \mathbf{x}'\boldsymbol{\beta})$$

STD standard error estimates of $a_j + \mathbf{x}'\mathbf{b}$

XBETA estimates of $a_j + \mathbf{x}'\boldsymbol{\beta}$

OUT=*SAS-data-set* names the output data set. By default, the new data set is named by using the DATA*n* convention.

When the *single variable response* syntax is used, the _LEVEL_ variable is added to the output data set, and there are $k - 1$ output observations for each input observation, where k is the number of response levels. There is no observation output corresponding to the highest response level. For each of the $k - 1$ observations, the PROB variable contains the fitted probability of obtaining a response level up to the level indicated by the _LEVEL_ variable, the XBETA variable contains $a_j + \mathbf{x}'\mathbf{b}$, where j references the levels ($a_1 = 0$), and the STD variable contains the standard error estimate of the XBETA variable. See the section "Details: PROBIT Procedure" on page 8286 for the formulas for the parameterizations.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

TEST Statement

TEST *<model-effects> </options>* **;**

The TEST statement enables you to perform chi-square tests for model effects that test Type I, Type II, or Type III hypotheses. By default, the Type III tests are performed. For more information, see Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

A WEIGHT statement can be used with PROC PROBIT to weight each observation by the value of the variable specified. The contribution of each observation to the likelihood function is multiplied by the value of the weight variable. Observations with zero, negative, or missing weights are not used in model estimation.

Details: PROBIT Procedure

Missing Values

PROC PROBIT does not use any observations having missing values for any of the independent variables, the response variables, or the weight variable. If only the response variables are missing, statistics requested in the OUTPUT statement are computed.

Response Level Ordering

For binary response data, PROC PROBIT fits the following model by default:

$$\Phi^{-1}\left(\frac{p-C}{1-C}\right) = \mathrm{x}'\boldsymbol{\beta}$$

where p is the probability of the response level identified as the first level in the "Response Profile" table in the output and Φ is the normal cumulative distribution function. By default, the covariate vector $\mathbf{x}$ contains an intercept term. This is sometimes called Abbot's formula.

Because of the symmetry of the normal (and logistic) distribution, the effect of reversing the order of the two response values is to change the signs of $\boldsymbol{\beta}$ in the preceding equation.

By default, response levels appear in ascending, sorted order (that is, the lowest level appears first, and then the next lowest, and so on). There are a number of ways that you can control the sort order of the response categories and, therefore, which level is assigned the first ordered level. One of the most common sets of response levels is $\{0,1\}$, with 1 representing the event with the probability that is to be modeled.

Consider the example where Y takes the values 1 and 0 for event and nonevent, respectively, and EXPOSURE is the explanatory variable. By default, PROC PROBIT assigns the first ordered level to response level 0, causing the probability of the nonevent to be modeled. There are several ways to change this.

Besides recoding the variable Y, you can do the following:

- Explicitly state which response level is to be modeled by using the response variable option EVENT= in the MODEL statement:

```
model Y(event='1') = Exposure;
```

- Specify the nonevent category for the response variable in the response variable option REF= in the MODEL statement:

```
model Y(ref='0') = Exposure;
```

- Specify the response variable option DESCENDING in the MODEL statement to assign the lowest ordered value to Y=1:

```
model Y(descending)=Exposure;
```

- Assign a format to Y such that the first formatted value (when the formatted values are put in sorted order) corresponds to the event. For the following example, Y=0 could be assigned formatted value 'nonevent' and Y=1 could be assigned formatted value 'event.' Since ORDER=FORMATTED by default, Y=1 becomes the first ordered level. See Example 100.3 for an illustration of this method.

```
proc format;
   value disease 1='event' 0='nonevent';
run;
proc probit;
   model y=exposure;
   format y disease.;
run;
```

- Arrange the input data set so that Y=1 appears first and use the ORDER=DATA option in the PROC PROBIT statement. Because ORDER=DATA sorts levels in order of their appearance in the data set, Y=1 becomes the first ordered level. Note that this option causes classification variables to be sorted by their order of appearance in the data set, also.

Computational Method

The log-likelihood function is maximized by means of a ridge-stabilized Newton-Raphson algorithm. Initial regression parameter estimates are set to zero. The INITIAL= and INTERCEPT= options in the MODEL statement can be used to give nonzero initial estimates.

The log-likelihood function, L, is computed as

$$L = \sum_i w_i \ln(p_i)$$

where the sum is over the observations in the data set, w_i is the weight for the ith observation, and p_i is the modeled probability of the observed response. In the case of the events/trials syntax in the MODEL statement, each observation contributes two terms corresponding to the probability of the event and the probability of its complement:

$$L = \sum_i w_i [r_i \ln(p_i) + (n_i - r_i) \ln(1 - p_i)]$$

where r_i is the number of events and n_i is the number of trials for observation i. This log-likelihood function differs from the log-likelihood function for a binomial or multinomial distribution by additive terms consisting of the log of binomial or multinomial coefficients. These terms are parameter-independent and do not affect the model estimation or the standard errors and tests.

The estimated covariance matrix, $\mathbf{V}$, of the parameter estimates is computed as the negative inverse of the information matrix of second derivatives of L with respect to the parameters evaluated at the final parameter estimates. Thus, the estimated covariance matrix is derived from the observed information matrix rather than the expected information matrix (these are generally not the same). The standard error estimates for the parameter estimates are taken as the square roots of the corresponding diagonal elements of $\mathbf{V}$.

If convergence of the maximum likelihood estimates is attained, a Type III chi-square test statistic is computed for each effect, testing whether there is any contribution from any of the levels of the effect. This statistic is computed as a quadratic form in the appropriate parameter estimates by using the corresponding submatrix of the asymptotic covariance matrix estimate. See Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions," for more information about Type III estimable functions.

The asymptotic covariance matrix is computed as the inverse of the observed information matrix. Note that if the NOINT option is specified and classification variables are used, the first classification variable contains a contribution from an intercept term. The results are displayed in an ODS table named "Type3Analysis".

Chi-square tests for individual parameters are Wald tests based on the observed information matrix and the parameter estimates. If an effect has a single degree of freedom in the parameter estimates table, the chi-square test for this parameter is equivalent to the Type III test for this effect.

The theory behind these tests assumes large samples. If the samples are not large, it might be better to base the tests on log-likelihood ratios. These changes in log likelihood can be obtained by fitting the model twice, once with all the parameters of interest and once leaving out the parameters to be tested. See Cox and Oakes (1984) for a discussion of the merits of some possible test methods.

If some of the independent variables are perfectly correlated with the response pattern, then the theoretical parameter estimates can be infinite. Although fitted probabilities of 0 and 1 are not especially pathological, infinite parameter estimates are required to yield these probabilities. Due to the finite precision of computer

arithmetic, the actual parameter estimates are not infinite. Indeed, since the tails of the distributions allowed in the PROBIT procedure become small rapidly, an argument to the cumulative distribution function of around 20 becomes effectively infinite. In the case of such parameter estimates, the standard error estimates and the corresponding chi-square tests are not trustworthy.

Distributions

The distributions, $F(x)$, allowed in the PROBIT procedure are specified with the DISTRIBUTION= option in the MODEL statement. The cumulative distribution functions for the available distributions are

Cumulative Distribution Function	Distribution
$\int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{z^2}{2}\right)\, dz$	Normal
$\frac{1}{1+e^{-x}}$	Logistic
$1 - e^{-e^{x}}$	Extreme value or Gompertz

The variances of these three distributions are not all equal to 1, and their means are not all equal to zero. Their means and variances are shown in the following table, where γ is the Euler constant.

Distribution	Mean	Variance
Normal	0	1
Logistic	0	$\pi^2/3$
Extreme value or Gompertz	$-\gamma$	$\pi^2/6$

When comparing parameter estimates by using different distributions, you need to take into account the different scalings and, for the extreme value (or Gompertz) distribution, a possible shift in location. For example, if the fitted probabilities are in the neighborhood of 0.1 to 0.9, then the parameter estimates from the logistic model should be about $\pi/\sqrt{3}$ larger than the estimates from the probit model.

INEST= *SAS-data-set*

The INEST= data set names a SAS data set that specifies initial estimates for all the parameters in the model.

The INEST= data set must contain the intercept variables (named Intercept for binary response model and Intercept, Intercept2, Intercept3, and so forth, for multinomial response models) and all independent variables in the MODEL statement.

If BY processing is used, the INEST= data set should also include the BY variables, and there must be at least one observation for each BY group. If there is more than one observation in a BY group, the first one read is used for that BY group.

If the INEST= data set also contains the _TYPE_ variable, only observations with the _TYPE_ value "PARMS" are used as starting values. Combining the INEST= data set and the option MAXIT= in the MODEL statement, partial scoring can be done, such as predicting on a validation data set by using the model built from a training data set.

You can specify starting values for the iterative algorithm in the INEST= data set. This data set overwrites the INITIAL= option in the MODEL statement, which is a little difficult to use for models with multilevel interaction effects. The INEST= data set has the same structure as the "OUTEST= *SAS-data-set*" on page 8294, but it is not required to have all the variables or observations that appear in the OUTEST= data set. One simple use of the INEST= option is passing the previous OUTEST= data set directly to the next model as an INEST= data set, assuming that the two models have the same parameterization.

Model Specification

For a two-level response, the probability that the lesser response occurs is modeled by the probit equation as

$$p = C + (1 - C)F(\mathbf{x}'\mathbf{b})$$

The probability of the other (complementary) event is $1 - p$.

For a multilevel response with outcomes labeled l_i for $i = 1, 2, \ldots, k$, the probability, p_j, of observing level l_j is as follows:

$$\begin{aligned}
p_1 &= C + (1 - C)F(\mathbf{x}'\mathbf{b}) \\
p_2 &= (1 - C)\left(F(a_2 + \mathbf{x}'\mathbf{b}) - F(\mathbf{x}'\mathbf{b})\right) \\
&\vdots \\
p_j &= (1 - C)\left(F(a_j + \mathbf{x}'\mathbf{b}) - F(a_{j-1} + \mathbf{x}'\mathbf{b})\right) \\
&\vdots \\
p_k &= (1 - C)(1 - F(a_{k-1} + \mathbf{x}'\mathbf{b}))
\end{aligned}$$

Thus, for a k-level response, there are $k - 2$ additional parameters, $a_2, a_3, \ldots, a_{k-1}$, estimated. These parameters are denoted by Intercept*j*, $j = 2, 3, \ldots, k - 1$, in the output.

An intercept parameter is always added to the set of independent variables as the first term in the model unless the NOINT option is specified in the MODEL statement. If a classification variable taking on k levels is used as one of the independent variables, a set of k indicator variables is generated to model the effect of this variable. Because of the presence of the intercept term, there are at most $k - 1$ degrees of freedom for this effect in the model.

Lack-of-Fit Tests

Two goodness-of-fit tests can be requested from the PROBIT procedure: a Pearson's chi-square test and a log-likelihood ratio chi-square test.

To compute the test statistics, you can use the AGGREGATE or AGGREGATE= option grouping the observations into subpopulations. If neither AGGREGATE nor AGGREGATE= is specified, PROC PROBIT assumes that each observation is from a separate subpopulation and computes the goodness-of-fit test statistics only for the *events/trials* syntax.

If the Pearson's goodness-of-fit chi-square test is requested and the p-value for the test is too small, variances and covariances are adjusted by a heterogeneity factor (the goodness-of-fit chi-square divided by its degrees of freedom) and a critical value from the t distribution is used to compute the fiducial limits. The Pearson's chi-square test statistic is computed as

$$\chi^2_P = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{(r_{ij} - n_i \hat{p}_{ij})^2}{n_i \hat{p}_{ij}}$$

where the sum on i is over grouping, the sum on j is over levels of response, r_{ij} is the frequency of response level j for the ith grouping, n_i is the total frequency for the ith grouping, and $\hat{p}_{ij}$ is the fitted probability for the jth level at the ith grouping.

The likelihood ratio chi-square test statistic is computed as

$$\chi^2_D = 2 \sum_{i=1}^{m} \sum_{j=1}^{k} r_{ij} \ln \left(\frac{r_{ij}}{n_i \hat{p}_{ij}} \right)$$

This quantity is sometimes called the deviance. If the modeled probabilities fit the data, these statistics should be approximately distributed as chi-square with degrees of freedom equal to $(k-1) \times m - q$, where k is the number of levels of the multinomial or binomial response, m is the number of sets of independent variable values (covariate patterns), and q is the number of parameters fit in the model.

In order for the Pearson's statistic and the deviance to be distributed as chi-square, there must be sufficient replication within the groupings. When this is not true, the data are sparse, and the p-values for these statistics are not valid and should be ignored. Similarly, these statistics, divided by their degrees of freedom, cannot serve as indicators of overdispersion. A large difference between the Pearson's statistic and the deviance provides some evidence that the data are too sparse to use either statistic.

Rescaling the Covariance Matrix

One way of correcting overdispersion is to multiply the covariance matrix by a dispersion parameter. You can supply the value of the dispersion parameter directly, or you can estimate the dispersion parameter based on either the Pearson's chi-square statistic or the deviance for the fitted model.

The Pearson's chi-square statistic χ^2_P and the deviance χ^2_D are defined in the section "Lack-of-Fit Tests" on page 8291. If the SCALE= option is specified in the MODEL statement, the dispersion parameter is estimated by

$$\widehat{\sigma}^2 = \begin{cases} \chi^2_P/(m(k-1)-q) & \text{SCALE=PEARSON} \\ \chi^2_D/(m(k-1)-q) & \text{SCALE=DEVIANCE} \\ (\textit{constant})^2 & \text{SCALE=}\textit{constant} \end{cases}$$

In order for the Pearson's statistic and the deviance to be distributed as chi-square, there must be sufficient replication within the subpopulations. When this is not true, the data are sparse, and the p-values for these statistics are not valid and should be ignored. Similarly, these statistics, divided by their degrees of freedom, cannot serve as indicators of overdispersion. A large difference between the Pearson's statistic and the deviance provides some evidence that the data are too sparse to use either statistic.

You can use the AGGREGATE (or AGGREGATE=) option to define the subpopulation profiles. If you do not specify this option, each observation is regarded as coming from a separate subpopulation. For *events/trials* syntax, each observation represents n Bernoulli trials, where n is the value of the *trials* variable; for *single-trial* syntax, each observation represents a single trial. Without the AGGREGATE (or AGGREGATE=) option, the Pearson's chi-square statistic and the deviance are calculated only for *events/trials* syntax.

Note that the parameter estimates are not changed by this method. However, their standard errors are adjusted for overdispersion, affecting their significance tests.

Tolerance Distribution

For a single independent variable, such as a dosage level, the models for the probabilities can be justified on the basis of a population with mean μ and scale parameter σ of tolerances for the subjects. Then, given a dose x, the probability, P, of observing a response in a particular subject is the probability that the subject's tolerance is less than the dose or

$$P = F\left(\frac{x-\mu}{\sigma}\right)$$

Thus, in this case, the intercept parameter, b_0, and the regression parameter, b_1, are related to μ and σ by

$$b_0 = -\frac{\mu}{\sigma},\ b_1 = \frac{1}{\sigma}$$

NOTE: The parameter σ is not equal to the standard deviation of the population of tolerances for the logistic and extreme value distributions.

Inverse Confidence Limits

In bioassay problems, estimates of the values of the independent variables that yield a desired response are often needed. For instance, the value yielding a 50% response rate (called the ED50 or LD50) is often used. The INVERSECL option requests that confidence limits be computed for the value of the independent variable that yields a specified response. These limits are computed only for the first continuous variable effect in the model. The other variables are set either at their mean values if they are continuous or at the reference (last) level if they are discrete variables. For a discussion of inverse confidence limits, see Hubert, Bohidar, and Peace (1988).

For the PROBIT procedure, the response variable is a probability. An estimate of the first continuous variable value needed to achieve a response of p is given by

$$\hat{x}_1 = \frac{1}{b_1}\left(F^{-1}(p) - \mathbf{x}^{*\prime}\mathbf{b}^*\right)$$

where F is the cumulative distribution function used to model the probability, $\mathbf{x}^*$ is the vector of independent variables excluding the first one, which can be specified by the XDATA= option described in the section "XDATA= *SAS-data-set*" on page 8294, $\mathbf{b}^*$ is the vector of parameter estimates excluding the first one, and b_1 is the estimated regression coefficient for the independent variable of interest. This estimate assumes that there is no natural response rate ($C = 0$). When C is nonzero, the quantiles and confidence limits for the independent variable correspond to the adjusted probability $C + (1 - C)p$, rather than to p. As a result, an estimate of the value yielding response rate p is associated with the $(p - C)/(1 - C)$ quantile. For example, if $C = 0.1$ then an estimate of the LD50 is found corresponding to the 0.44 quantile. This value can be thought of as yielding 50% of the variable's effect, but a 44% response rate. For both binary and ordinal models, the INVERSECL option provides estimates of the value of x_1, which yields $\Pr(\text{first response level}) = p$, for various values of p.

This estimator is given as a ratio of random variables, such as $r = a/b$. Confidence limits for this ratio can be computed by using Fieller's theorem. A brief description of this theorem follows. See Finney (1971) for a more complete description of Fieller's theorem.

If the random variables a and b are thought to be distributed as jointly normal, then for any fixed value r the following probability statement holds if z is an $\alpha/2$ quantile from the standard normal distribution and $\mathbf{V}$ is the variance-covariance matrix of a and b:

$$\Pr\left((a - rb)^2 > z^2(V_{aa} - 2rV_{ab} + r^2V_{bb})\right) = \alpha$$

Usually the inequality can be solved for r to yield a confidence interval. The PROBIT procedure uses a value of 1.96 for z, corresponding to an α value of 0.05, unless the goodness-of-fit p-value is less than the specified value of the HPROB= option. When this happens, the covariance matrix is scaled by the heterogeneity factor, and a t distribution quantile is used for z.

It is possible for the roots of the equation for r to be imaginary or for the confidence interval to be all points outside of an interval. In these cases, the limits are set to missing by the PROBIT procedure.

Although the normal and logistic distribution give comparable fitted values of p if the empirically observed proportions are not too extreme, they can give appreciably different values when extrapolated into the tails. Correspondingly, the estimates of the confidence limits and dose values can be different for the two distributions even when they agree quite well in the body of the data. Extrapolation outside of the range of the actual data is often sensitive to model assumptions, and caution is advised if extrapolation is necessary.

OUTEST= *SAS-data-set*

The OUTEST= data set contains parameter estimates and the log likelihood for the model. You can specify a label in the MODEL statement to distinguish between the estimates for different models used by the PROBIT procedure. If you specify the COVOUT option, the OUTEST= data set also contains the estimated covariance matrix of the parameter estimates.

The OUTEST= data set contains each variable used as a dependent or independent variable in any MODEL statement. One observation consists of parameter values for the model with the dependent variable having the value -1. If you specify the COVOUT option, there are additional observations containing the rows of the estimated covariance matrix. For these observations, the dependent variable contains the parameter estimate for the corresponding row variable. The following variables are also added to the data set:

MODEL	a character variable containing the label of the MODEL statement, if present, or blank otherwise
NAME	a character variable containing the name of the dependent variable for the parameter estimates observations or the name of the row for the covariance matrix estimates
TYPE	a character variable containing the type of the observation, either PARMS for parameter estimates or COV for covariance estimates
DIST	a character variable containing the name of the distribution modeled
LNLIKE	a numeric variable containing the last computed value of the log likelihood
_C_	a numeric variable containing the estimated threshold parameter
INTERCEPT	a numeric variable containing the intercept parameter estimates and covariances

Any BY variables specified are also added to the OUTEST= data set.

XDATA= *SAS-data-set*

The XDATA= data set is used for specifying values for the effects in the MODEL statement when predicted values and/or fiducial limits for a single continuous variable (dose variable) are required. It is also used for plots specified by the PLOTS= options: PLOTS=CDFPLOT, PLOTS=IPPPLOT, PLOTS=LPREDPLOT, and PLOTS=PREDPPLOT.

The XDATA= data names a SAS data set that contains user input values for all the independent variables in the MODEL statement and the variables in the CLASS statement. The XDATA= data set has the same structure as the DATA= data set but is not required to have all the variables or observations that appear in the DATA= data set.

The XDATA= data set must contain all the independent variables in the MODEL statement and variables in the CLASS statement. Even though variables in the CLASS statement are not used in the MODEL statement, valid values are required for these variables in the XDATA= data set. Missing values are not allowed. For independent variables in the MODEL statement, although the dose variable's value is not used in the computing of predicted values and/or fiducial limits for the dose variable, missing values are not allowed in the XDATA= data set for any of the independent variables. Missing values are allowed for the dependent variables and other variables if they are included in the XDATA= data set and not listed in the CLASS statement.

If BY processing is used, the XDATA= data set should also include the BY variables, and there must be at least one valid observation for each BY group. If there is more than one valid observation in one BY group, the last one read is used for that BY group.

If there is no XDATA= data set in the PROC PROBIT statement, by default, the PROBIT procedure will use overall mean for effects containing continuous variable (or variables) and the highest level of a single classification variable as reference level. The rules are summarized as follows:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

Plot Options Superseded by ODS Graphics

You can select one of the following two types of graphics in PROC PROBIT: ODS and traditional. ODS Graphics is the preferred method of creating graphs, superseding traditional graphs.

When ODS Graphics is enabled, you can use the PLOTS= option in the PROC PROBIT statement to create plots by using ODS Graphics. For more information about ODS Graphics options, see the PLOTS= option.

There are four plot statements that you can use to request traditional high-resolution plots: CDFPLOT, IPPPLOT, LPREDPLOT, and PREDPPLOT. Some of these statements apply only to either the binomial model or the multinomial model. Table 100.8 shows the availability of these statements for different models.

Table 100.8 Plot Statement Availability

Statement	Binomial	Multinomial
CDFPLOT	No	Yes
IPPPLOT	Yes	No
LPREDPLOT	Yes	Yes
PREDPPLOT	Yes	Yes

The following example uses the data set study in the section "Estimating the Natural Response Threshold Parameter" on page 8261 to illustrate how to create high-resolution plots for the binomial model:

```
proc probit data=study log10 optc;
   model respond/number=dose;
   predpplot var=dose cfit=blue; inset;
   lpredplot var=dose cfit=blue; inset;
   ippplot   var=dose cfit=blue; inset/pos=se;
run;
```

All plot statements must follow the MODEL statement. The VAR= option specifies a continuous independent variable (dose variable) against which the predicted probability or the linear predictor is plotted. The INSET statement requests the inset box with summary information. See the section "INSET Statement" on page 8304 for more details.

The PREDPPLOT statement creates a plot that shows the relationship between dosage level, observed response proportions, and estimated probability values. See the section "PREDPPLOT Statement" on page 8323 for more details. The IPPPLOT statement creates a similar plot. See the section "IPPPLOT Statement" on page 8306 for details about this plot. The LPREDPLOT statement creates a linear predictor plot, which is described in the section "LPREDPLOT Statement" on page 8315.

The following example uses the data set multi from Example 100.2 to illustrate how to create high-resolution plots for the multinomial model:

```
proc probit data=multi order=data;
   class prep;
   model symptoms=prep ldose;
   cdfplot var=ldose  level=("None" "Mild" "Severe")
             cfit=blue cframe=ligr noconf;
   lpredplot var=ldose  level=("None" "Mild" "Severe")
             cfit=blue cframe=ligr;
   predpplot var=ldose  level=("None" "Mild" "Severe")
             cfit=blue cframe=ligr;
   weight n;
run;
```

The CDFPLOT statement creates a plot that shows the relationship between the cumulative response probabilities and the dose levels. The multinomial model plots are similar to those with the binomial model.

You can specify the following option in the PROC LIFEREG statement:

GOUT=*graphics-catalog*
specifies a graphics catalog in which to save graphics output.

CDFPLOT Statement

CDFPLOT < **VAR=***variable* > < *options* > **;**

The CDFPLOT statement plots the predicted cumulative distribution function (CDF) of the multinomial response variable as a function of a single continuous independent variable (dose variable). You can use this statement only after a multinomial model statement.

VAR=*variable*
specifies a single continuous variable (dose variable) in the independent variable list of the MODEL statement. If a VAR= *variable* is not specified, the first single continuous variable in the independent variable list of the MODEL statement is used. If such a variable does not exist in the independent variable list of the MODEL statement, an error is reported.

The predicted cumulative distribution function is defined as

$$\hat{F}_j(\mathbf{x}) = C + (1 - C)F(\hat{a}_j + \mathbf{x}'\hat{\mathbf{b}})$$

where $j = 1, \ldots, k$ are the indexes of the k levels of the multinomial response variable, F is the CDF of the distribution used to model the cumulative probabilities, $\hat{\mathbf{b}}$ is the vector of estimated parameters, $\mathbf{x}$ is the covariate vector, $\hat{a}_j$ are estimated ordinal intercepts with $\hat{a}_1 = 0$, and C is the threshold parameter, either known or estimated from the model. Let x_1 be the covariate corresponding to the

dose variable and $\mathbf{x}_{-1}$ be the vector of the rest of the covariates. Let the corresponding estimated parameters be $\hat{b}_1$ and $\hat{\mathbf{b}}_{-1}$. Then

$$\hat{F}_j(\mathbf{x}) = C + (1 - C)F(\hat{a}_j + x_1\hat{b}_1 + \mathbf{x}'_{-1}\hat{\mathbf{b}}_{-1})$$

To plot $\hat{F}_j$ as a function of x_1, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow these rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

options

specify the levels of the multinomial response variable for which the CDF curves are requested, and add features to the plot. There are $k - 1$ curves for a k-level multinomial response variable (for the highest level, it is the constant line 1). You can specify any of them to be plotted by the LEVEL= option in the CDFPLOT statement. See the LEVEL= option for how to specify the levels.

An attached box on the right side of the plot is used to label these curves with the names of their levels. You can specify the color of this box by using the CLABBOX= option.

You can use *options* in the CDFPLOT statement to do the following:

- superimpose specification limits
- specify the levels for which the CDF curves are requested
- specify graphical enhancements (such as color or text height)

Summary of Options

Table 100.9 through Table 100.15 summarize the *options* available in the CDFPLOT statement. The "Dictionary of Options" on page 8300 describes each option in detail.

Table 100.9 CDFPLOT Statement Options

LEVEL=(*character-list*)	Specifies the names of the levels for which the CDF curves are requested
NOTHRESH	Suppresses the threshold line
THRESHLABPOS=*value*	Specifies the position for the label of the threshold line

Table 100.10 Color Options

CAXIS=*color*	Specifies color for axis
CFIT=*color*	Specifies color for fitted curves
CFRAME=*color*	Specifies color for frame
CGRID=*color*	Specifies color for grid lines
CHREF=*color*	Specifies color for HREF= lines
CLABBOX=*color*	Specifies color for label box
CTEXT=*color*	Specifies color for text
CVREF=*color*	Specifies color for VREF= lines

Table 100.11 Options to Enhance Plots Produced on Graphics Devices

ANNOTATE=*SAS-data-set*	Specifies an Annotate data set
INBORDER	Requests a border around plot
LFIT=*linetype*	Specifies line style for fitted curves
LGRID=*linetype*	Specifies line style for grid lines
NOFIT	Suppresses CDF curves
NOFRAME	Suppresses the frame around plotting areas
NOGRID	Suppresses grid lines
NOHLABEL	Suppresses horizontal labels
NOHTICK	Suppresses horizontal ticks
NOVTICK	Suppresses vertical ticks
TURNVLABELS	Vertically strings out characters in vertical labels
WFIT=*n*	Specifies thickness for fitted curves
WGRID=*n*	Specifies thickness for grids
WREFL=*n*	Specifies thickness for reference lines

Table 100.12 Axis Options

HAXIS=*value1 to value2 < by value3 >*	Specifies tick mark values for horizontal axis
HLOWER=*value*	Specifies lower limit on horizontal axis scale
HOFFSET=*value*	Specifies offset for horizontal axis
HUPPER=*value*	Specifies upper limit on horizontal axis scale
NHTICK=*n*	Specifies number of ticks for horizontal axis
NVTICK=*n*	Specifies number of ticks for vertical axis
VAXIS=*value1 to value2 < by value3 >*	Specifies tick mark values for vertical axis
VAXISLABEL='*label*'	Specifies label for vertical axis
VLOWER=*value*	Specifies lower limit on vertical axis scale
VOFFSET=*value*	Specifies offset for vertical axis
VUPPER=*value*	Specifies upper limit on vertical axis scale
WAXIS=*n*	Specifies thickness for axis

Table 100.13 Graphics Catalog Options

DESCRIPTION='*string*'	Specifies description for graphics catalog member
NAME='*string*'	Specifies name for plot in graphics catalog

Table 100.14 Options for Text Enhancement

FONT=*font*	Specifies software font for text
HEIGHT=*value*	Specifies height of text outside framed areas
INHEIGHT=*value*	Specifies height of text inside framed areas

Table 100.15 Options for Reference Lines

HREF< (INTERSECT) >=value-list	Requests horizontal reference line
HREFLABELS=('label1',. . . ,'labeln')	Specifies labels for HREF= lines
HREFLABPOS=*n*	Specifies vertical position of labels for HREF= lines
LHREF=*linetype*	Specifies line style for HREF= lines
LVREF=*linetype*	Specifies line style for VREF= lines
VREF< (INTERSECT) >=value-list	Requests vertical reference line
VREFLABELS=('*label1*',. . . ,'*labeln*')	Specifies labels for VREF= lines
VREFLABPOS=*n*	Specifies horizontal position of labels for VREF= lines

Dictionary of Options

The following entries provide detailed descriptions of the *options* in the CDFPLOT statement.

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an Annotate data set, as described in *SAS/GRAPH: Reference*, that enables you to add features to the CDF plot. The ANNOTATE= data set you specify in the CDFPLOT statement is used for all plots created by the statement.

CAXIS=*color*

CAXES=*color*

specifies the color for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CFIT=*color*

specifies the color for the fitted CDF curves. The default is the first color in the device color list.

CFRAME=*color*

CFR=*color*

specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*

specifies the color for grid lines. The default is the first color in the device color list.

CLABBOX=*color*

specifies the color for the area enclosed by the label box for CDF curves. This area is not shaded by default.

CHREF=*color*

CH=*color*

specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*

specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*

CV=*color*

specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION="*string*"

DES="*string*"

specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*

specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HAXIS=*value1 to value2* **<** *by value3* **>**

specifies tick mark values for the horizontal axis; *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. If *value3* is omitted, a value of 1 is used.

Examples of HAXIS= lists follow:

```
haxis = 0 to 10
haxis = 2 to 10 by 2
haxis = 0 to 200 by 10
```

HEIGHT=*value*

specifies the height of text outside framed areas. The default value is 3.846 (in percentage).

HLOWER=*value*

specifies the lower limit on the horizontal axis scale. The HLOWER= option specifies *value* as the lower horizontal axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HOFFSET=*value*

specifies offset for horizontal axis. The default value is 1.

HREF < (INTERSECT) > =*value-list*

requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS='*label1*'**,. . .,**'*labeln*'

HREFLABEL='*label1*'**,. . .,**'*labeln*'

HREFLAB='*label1*'**,. . .,**'*labeln*'

specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*

specifies the vertical position of labels for HREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Top
2	Staggered from top
3	Bottom
4	Staggered from bottom
5	Alternating from top
6	Alternating from bottom

HUPPER=*value*
specifies *value* as the upper horizontal axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

INBORDER
requests a border around CDF plots.

INHEIGHT=*value*
specifies the height of text inside framed areas.

LEVEL=(*character-list***)**
ORDINAL=(*character-list***)**
specifies the names of the levels for which CDF curves are requested. Names should be quoted and separated by space. If there is no correct name provided, no CDF curve is plotted.

LFIT=*linetype*
specifies a line style for fitted curves. By default, fitted curves are drawn by connecting solid lines (*linetype* = 1).

LGRID=*linetype*
specifies a line style for all grid lines. *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

NAME='*string*'
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'PROBIT'.

NHTICK=*n*
specifies the number of ticks for the horizontal axis.

NOFIT
suppresses the fitted CDF curves.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOTHRESH
suppresses the threshold line.

NOVLABEL
suppresses vertical labels.

NVTICK=*n*
specifies the number of ticks for the vertical axis.

NOVTICK
suppresses vertical tick marks.

THRESHLABPOS=*n*
specifies the horizontal position of labels for the threshold line. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Left
2	Right

TURNVLABELS
vertically strings out characters in vertical labels.

VAXIS=*value1 to value2* **<** *by value3* **>**
specifies tick mark values for the vertical axis; *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. This method of specification of tick marks is not valid for logarithmic axes. If *value3* is omitted, a value of 1 is used.

Examples of VAXIS= lists follow:

```
vaxis = 0 to 10
vaxis = 0 to 2 by .1
```

VAXISLABEL='*string*'
specifies a label for the vertical axis.

VLOWER=*value*
specifies the lower limit on the vertical axis scale. The VLOWER= option specifies *value* as the lower vertical axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

VOFFSET=*value*
specifies the offset for the vertical axis.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS='*label1*',. . . ,'*labeln*'
VREFLABEL='*label1*',. . . ,'*labeln*'
VREFLAB='*label1*',. . . ,'*labeln*'
specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*
specifies the horizontal position of labels for VREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Left
2	Right

VUPPER=*value*
specifies the upper limit on the vertical axis scale. The VUPPER= option specifies *value* as the upper vertical axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

WAXIS=*n*
specifies line thickness for axes and frame. The default value is 1.

WFIT=*n*
specifies line thickness for fitted curves. The default value is 1.

WGRID=*n*
specifies line thickness for grids. The default value is 1.

WREFL=*n*
specifies line thickness for reference lines. The default value is 1.

INSET Statement

INSET < *keyword-list* > < / *options* > **;**

The box or table of summary information produced on plots made with the CDFPLOT, IPPPLOT, LPREDPLOT, or PREDPPLOT statement is called an *inset*. You can use the INSET statement to customize both the information that is printed in the inset box and the appearance of the inset box. To supply the information that is displayed in the inset box, you specify *keywords* corresponding to the information you want shown. For example, the following statements produce a predicted probability plot with the number of trials, the number of events, the name of the distribution, and the estimated optimum natural threshold in the inset.

```
proc probit data=epidemic;
   model r/n = dose;
   predpplot;
   inset nobs ntrials nevents dist optc;
run;
```

By default, inset entries are identified with appropriate labels. However, you can provide a customized label by specifying the *keyword* for that entry followed by the equal sign (=) and the label in quotes. For example, the following INSET statement produces an inset containing the number of observations and the name of the distribution, labeled “Sample Size” and “Distribution” in the inset.

```
inset nobs='Sample Size' dist='Distribution';
```

If you specify a *keyword* that does not apply to the plot you are creating, then the *keyword* is ignored.

The *options* control the appearance of the box.

If you specify more than one INSET statement, only the first one is used.

Keywords Used in the INSET Statement

Table 100.16 and Table 100.17 list *keywords* available in the INSET statement to display summary statistics, distribution parameters, and distribution fitting information.

Table 100.16 Summary Statistics

C	User-input threshold
NEVENTS	Number of events
NOBS	Number of observations
NRESPLEV	Number of levels of the response variable
NTRIALS	Number of trials
OPTC	Estimated natural threshold

Table 100.17 General Information

CONFIDENCE	Confidence coefficient for all confidence intervals
DIST	Name of the distribution

Options Used in the INSET Statement

Table 100.18 and Table 100.19 list the *options* available in the INSET statement.

Table 100.18 Color and Pattern Options

CFILL=*color*	Specifies color for filling box
CFILLH=*color*	Specifies color for filling box header
CFRAME=*color*	Specifies color for frame
CHEADER=*color*	Specifies color for text in header
CTEXT=*color*	Specifies color for text

Table 100.19 General Appearance Options

FONT=*font*	Specifies software font for text
HEADER="*quoted-string*'	Specifies text for header or box title
HEIGHT=*value*	Specifies height of text
NOFRAME	Omits frame around box
POS= *value* < DATA | PERCENT >	Determines the position of the inset. The *value* can be a compass point (N, NE, E, SE, S, SW, W, NW) or a pair of coordinates (x, y) enclosed in parentheses. The coordinates can be specified in axis percentage units or axis data units.
REFPOINT= *name*	Specifies the reference point for an inset that is positioned by a pair of coordinates with the POS= option. You use the REFPOINT= option in conjunction with the POS= coordinates. The REFPOINT= option specifies which corner of the inset frame you have specified with coordinates (x, y), and it can take the value of BR (bottom right), BL (bottom left), TR (top right), or TL (top left). The default is REFPOINT=BL. If the inset position is specified as a compass point, then the REFPOINT= option is ignored.

IPPPLOT Statement

IPPPLOT < *variable* > < *options* > **;**

The IPPPLOT statement plots the inverse of the predicted probability (IPP) against a single continuous variable (dose variable) in the MODEL statement for the binomial model. You can use this statement only after a binomial model statement. The confidence limits for the predicted values of the dose variable are the computed fiducial limits, not the inverse of the confidence limits of the predicted probabilities. See the section "Inverse Confidence Limits" on page 8293 for more details.

VAR= *variable*

specifies a single continuous variable (dose variable) in the independent variable list of the MODEL statement. If a VAR= variable is not specified, the first single continuous variable in the independent variable list of the MODEL statement is used. If such a variable does not exist in the independent variable list of the MODEL statement, an error is reported.

For the binomial model, the response variable is a probability. An estimate of the dose level $\hat{x}_1$ needed for a response of p is given by

$$\hat{x}_1 = (F^{-1}(p) - \mathbf{x}'_{-1}\hat{\mathbf{b}}_{-1})/\hat{b}_1$$

where F is the cumulative distribution function used to model the probability, $\mathbf{x}_{-1}$ is the vector of the rest of the covariates, $\hat{\mathbf{b}}_{-1}$ is the vector of the estimated parameters corresponding to $\mathbf{x}_{-1}$, and $\hat{b}_1$ is the estimated parameter for the dose variable of interest.

To plot $\hat{x}_1$ as a function of p, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow these rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

options

add features to the plot.

You can use *options* in the IPPPLOT statement to do the following:

- superimpose specification limits
- suppress or add the observed data points on the plot
- suppress or add the fiducial limits on the plot
- specify graphical enhancements (such as color or text height)

Summary of Options

Table 100.20 through Table 100.26 summarize the *options* available in the IPPPLOT statement. The "Dictionary of Options" on page 8310 describes each option in detail.

Table 100.20 Plot Layout Options for the IPPPLOT Statement

NOCONF	Suppresses fiducial limits
NODATA	Suppresses observed data points on the plot
NOTHRESH	Suppresses the threshold line
THRESHLABPOS=*value*	Specifies the position for the label of the threshold line

Table 100.21 Color Options

CAXIS=*color*	Specifies color for axis
CFIT=*color*	Specifies color for fitted curves
CFRAME=*color*	Specifies color for frame
CGRID=*color*	Specifies color for grid lines
CHREF=*color*	Specifies color for HREF= lines
CTEXT=*color*	Specifies color for text
CVREF=*color*	Specifies color for VREF= lines

Table 100.22 Options to Enhance Plots Produced on Graphics Devices

ANNOTATE=*SAS-data-set*	Specifies an Annotate data set
INBORDER	Requests a border around plot
LFIT=*linetype*	Specifies line style for fitted curves and confidence limits
LGRID=*linetype*	Specifies line style for grid lines
NOFIT	Suppresses fitted curves
NOFRAME	Suppresses the frame around plotting areas
NOGRID	Suppresses grid lines
NOHLABEL	Suppresses horizontal labels
NOHTICK	Suppresses horizontal ticks
NOVTICK	Suppresses vertical ticks
TURNVLABELS	Vertically strings out characters in vertical labels
WFIT=*n*	Specifies thickness for fitted curves
WGRID=*n*	Specifies thickness for grids
WREFL=*n*	Specifies thickness for reference lines

Table 100.23 Axis Options

HAXIS=*value1 to value2 < by value3 >*	Specifies tick mark values for horizontal axis
HLOWER=*value*	Specifies lower limit on horizontal axis scale
HOFFSET=*value*	Specifies offset for horizontal axis
HUPPER=*value*	Specifies upper limit on horizontal axis scale
NHTICK=*n*	Specifies number of ticks for horizontal axis
NVTICK=*n*	Specifies number of ticks for vertical axis
VAXIS=*value1 to value2 < by value3 >*	Specifies tick mark values for vertical axis
VAXISLABEL='*label*'	Specifies label for vertical axis
VLOWER=*value*	Specifies lower limit on vertical axis scale
VOFFSET=*value*	Specifies offset for vertical axis
VUPPER=*value*	Specifies upper limit on vertical axis scale
WAXIS=*n*	Specifies thickness for axis

Table 100.24 Options for Reference Lines

HREF< (INTERSECT) >=*value-list*	Requests horizontal reference line
HREFLABELS=('label1',. . . ,'labeln')	Specifies labels for HREF= lines
HREFLABPOS=*n*	Specifies vertical position of labels for HREF= lines
LHREF=*linetype*	Specifies line style for HREF= lines
LVREF=*linetype*	Specifies line style for VREF= lines
VREF< (INTERSECT) > =*value-list*	Requests vertical reference line
VREFLABELS=('label1',. . . ,'labeln')	Specifies labels for VREF= lines
VREFLABPOS=*n*	Specifies horizontal position of labels for VREF= lines

Table 100.25 Graphics Catalog Options

DESCRIPTION='*string*'	Specifies description for graphics catalog member
NAME='*string*'	Specifies name for plot in graphics catalog

Table 100.26 Options for Text Enhancement

FONT=*font*	Specifies software font for text
HEIGHT=*value*	Specifies height of text outside framed areas
INHEIGHT=*value*	Specifies height of text inside framed areas

Dictionary of Options

The following entries provide detailed descriptions of the *options* in the IPPPLOT statement.

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an Annotate data set, as described in *SAS/GRAPH: Reference*, that enables you to add features to the IPP plot. The ANNOTATE= data set you specify in the IPPPLOT statement is used for all plots created by the statement.

CAXIS=*color*

CAXES=*color*

specifies the color for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CFIT=*color*

specifies the color for the fitted IPP curves. The default is the first color in the device color list.

CFRAME=*color*

CFR=*color*

specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*

specifies the color for grid lines. The default is the first color in the device color list.

CHREF=*color*

CH=*color*

specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*

specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*

CV=*color*

specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string***'**

DES='*string***'**

specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*

specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HAXIS=*value1 to value2* **<** *by value3* **>**
specifies tick mark values for the horizontal axis; *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. If *value3* is omitted, a value of 1 is used.

Examples of HAXIS= lists follow:

```
haxis = 0 to 10
haxis = 2 to 10 by 2
haxis = 0 to 200 by 10
```

HEIGHT=*value*
specifies the height of text outside framed areas. The default value is 3.846 (in percentage).

HLOWER=*value*
specifies the lower limit on the horizontal axis scale. The HLOWER= option specifies *value* as the lower horizontal axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HOFFSET=*value*
specifies the offset for the horizontal axis. The default value is 1.

HREF < (INTERSECT) > =*value-list*
requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS='*label1*',**...**,'*labeln*'

HREFLABEL='*label1*',**...**,'*labeln*'

HREFLAB='*label1*',**...**,'*labeln*'
specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*
specifies the vertical position of labels for HREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Top
2	Staggered from top
3	Bottom
4	Staggered from bottom
5	Alternating from top
6	Alternating from bottom

HUPPER=*value*
specifies *value* as the upper horizontal axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

INBORDER
requests a border around IPP plots.

INHEIGHT=*value*
specifies the height of text inside framed areas.

LFIT=*linetype*
specifies a line style for fitted curves and confidence limits. By default, fitted curves are drawn by connecting solid lines (*linetype* = 1) and confidence limits are drawn by connecting dashed lines (*linetype* = 3).

LGRID=*linetype*
specifies a line style for all grid lines. The value for *linetype* must be between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

NAME='*string*'
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'PROBIT'.

NHTICK=*n*
specifies the number of ticks for the horizontal axis.

NOCONF
suppresses fiducial limits from the plot.

NODATA
suppresses observed data points from the plot.

NOFIT
suppresses the fitted IPP curves.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOTHRESH
suppresses the threshold line.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

NVTICK=*n*
specifies the number of ticks for the vertical axis.

THRESHLABPOS=*n*
specifies the vertical position of labels for the threshold line. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Top
2	Bottom

TURNVLABELS
vertically strings out characters in vertical labels.

VAXIS=*value1 to value2* **<** *by value3* **>**
specifies tick mark values for the vertical axis; *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. This method of specification of tick marks is not valid for logarithmic axes. If *value3* is omitted, a value of 1 is used.

Examples of VAXIS= lists follow:

```
vaxis = 0 to 10
vaxis = 0 to 2 by .1
```

VAXISLABEL='*string*'
specifies a label for the vertical axis.

VLOWER=*value*
specifies the lower limit on the vertical axis scale. The VLOWER= option specifies *value* as the lower vertical axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

VOFFSET=*value*
specifies the offset for the vertical axis.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS='*label1*',. . . ,'*labeln*'

VREFLABEL='*label1*',. . . ,'*labeln*'

VREFLAB='*label1*',. . . ,'*labeln*'
specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*
specifies the horizontal position of labels for VREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Left
2	Right

VUPPER=*value*
specifies the upper limit on the vertical axis scale. The VUPPER= option specifies *value* as the upper vertical axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

WAXIS=*n*
specifies line thickness for axes and frame. The default value is 1.

WFIT=*n*
specifies line thickness for fitted curves. The default value is 1.

WGRID=*n*
specifies line thickness for grids. The default value is 1.

WREFL=*n*
specifies line thickness for reference lines. The default value is 1.

LPREDPLOT Statement

LPREDPLOT < **VAR=***variable* > < *options* > **;**

The LPREDPLOT statement plots the linear predictor (LPRED) $\mathbf{x}'\mathbf{b}$ against a single continuous variable (dose variable) in the MODEL statement for either the binomial model or the multinomial model. The confidence limits for the predicted values are available only for the binomial model.

VAR= *variable*

specifies a single continuous variable (dose variable) in the independent variable list of the MODEL statement for which the linear predictor plot is plotted. If a VAR= variable is not specified, the first single continuous variable in the independent variable list of the MODEL statement is used. If such a variable does not exist in the independent variable list of the MODEL statement, an error is reported.

Let x_1 be the covariate of the dose variable, $\mathbf{x}_{-1}$ be the vector of the rest of the covariates, $\hat{\mathbf{b}}_{-1}$ be the vector of estimated parameters corresponding to $\mathbf{x}_{-1}$, and $\hat{b}_1$ be the estimated parameter for the dose variable of interest.

To plot $\hat{\mathbf{x}}'\mathbf{b}$ as a function of x_1, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow these rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

options

add features to the plot.

For the multinomial model, you can use the LEVEL= option to specify the levels for which the linear predictor lines are plotted. The lines are labeled by the names of their levels in the middle.

You can use *options* in the LPREDPLOT statement to do the following:

- superimpose specification limits
- suppress or add the observed data points on the plot for the binomial model
- suppress or add the confidence limits for the binomial model
- specify the levels for which the linear predictor lines are requested for the multinomial model
- specify graphical enhancements (such as color or text height)

Summary of Options

Table 100.27 through Table 100.33 list all *options* by function. The "Dictionary of Options" on page 8318 describes each option in detail.

Table 100.27 Plot Layout Options for the LPREDPLOT Statement

LEVEL=(*character-list*)	Specifies the names of the levels for which the linear predictor lines are requested (only for the multinomial model)
NOCONF	Suppresses fiducial limits (only for the binomial model)
NODATA	Suppresses observed data points on the plot (only for the binomial model)
NOTHRESH	Suppresses the threshold line
THRESHLABPOS=*value*	Specifies the position for the label of the threshold line

Table 100.28 Color Options

CAXIS=*color*	Specifies color for axis
CFIT=*color*	Specifies color for fitted curves
CFRAME=*color*	Specifies color for frame
CGRID=*color*	Specifies color for grid lines
CHREF=*color*	Specifies color for HREF= lines
CTEXT=*color*	Specifies color for text
CVREF=*color*	Specifies color for VREF= lines

Table 100.29 Options to Enhance Plots Produced on Graphics Devices

ANNOTATE=*SAS-data-set*	Specifies an Annotate data set
INBORDER	Requests a border around plot
LFIT=*linetype*	Specifies line style for fitted curves and confidence limits
LGRID=*linetype*	Specifies line style for grid lines
NOFIT	Suppresses fitted curves
NOFRAME	Suppresses the frame around plotting areas
NOGRID	Suppresses grid lines
NOHLABEL	Suppresses horizontal labels
NOHTICK	Suppresses horizontal ticks
NOVTICK	Suppresses vertical ticks
TURNVLABELS	Vertically strings out characters in vertical labels
WFIT=*n*	Specifies thickness for fitted curves
WGRID=*n*	Specifies thickness for grids
WREFL=*n*	Specifies thickness for reference lines

Table 100.30 Axis Options

HAXIS=*value1 to value2 < by value3 >*	Specifies tick mark values for horizontal axis
HLOWER=*value*	Specifies lower limit on horizontal axis scale
HOFFSET=*value*	Specifies offset for horizontal axis
HUPPER=*value*	Specifies upper limit on horizontal axis scale
NHTICK=*n*	Specifies number of ticks for horizontal axis
NVTICK=*n*	Specifies number of ticks for vertical axis
VAXIS=*value1 to value2 < by value3 >*	Specifies tick mark values for vertical axis
VAXISLABEL='*label*'	Specifies label for vertical axis
VLOWER=*value*	Specifies lower limit on vertical axis scale
VOFFSET=*value*	Specifies offset for vertical axis
VUPPER=*value*	Specifies upper limit on vertical axis scale
WAXIS=*n*	Specifies thickness for axis

Table 100.31 Graphics Catalog Options

DESCRIPTION='*string*'	Specifies description for graphics catalog member
NAME='*string*'	Specifies name for plot in graphics catalog

Table 100.32 Options for Text Enhancement

FONT=*font*	Specifies software font for text
HEIGHT=*value*	Specifies height of text outside framed areas
INHEIGHT=*value*	Specifies height of text inside framed areas

Table 100.33 Options for Reference Lines

HREF< (INTERSECT) >=*value-list*	Requests horizontal reference line
HREFLABELS=('label1',. . . ,'labeln')	Specifies labels for HREF= lines
HREFLABPOS=*n*	Specifies vertical position of labels for HREF= lines
LHREF=*linetype*	Specifies line style for HREF= lines
LVREF=*linetype*	Specifies line style for VREF= lines
VREF< (INTERSECT) >=*value-list*	Requests vertical reference line
VREFLABELS=('label1',. . . ,'labeln')	Specifies labels for VREF= lines
VREFLABPOS=*n*	Specifies horizontal position of labels for VREF= lines

Dictionary of Options

The following entries provide detailed descriptions of the *options* in the LPREDPLOT statement.

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an Annotate data set, as described in *SAS/GRAPH: Reference*, that enables you to add features to the LPRED plot. The ANNOTATE= data set you specify in the LPREDPLOT statement is used for all plots created by the statement.

CAXIS=*color*

CAXES=*color*

specifies the color for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CFIT=*color*

specifies the color for the fitted LPRED lines. The default is the first color in the device color list.

CFRAME=*color*

CFR=*color*

specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*

specifies the color for grid lines. The default is the first color in the device color list.

CHREF=*color*

CH=*color*

specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*

specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*

CV=*color*

specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string*'

DES='*string*'

specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*

specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HAXIS=*value1 to value2* **<** *by value3* **>**

specifies tick mark values for the horizontal axis; *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. If *value3* is omitted, a value of 1 is used.

Examples of HAXIS= lists follow:

```
haxis = 0 to 10
haxis = 2 to 10 by 2
haxis = 0 to 200 by 10
```

HEIGHT=*value*

specifies the height of text outside framed areas. The default value is 3.846 (in percentage).

HLOWER=*value*

specifies the lower limit on the horizontal axis scale. The HLOWER= option specifies *value* as the lower horizontal axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HOFFSET=*value*

specifies offset for horizontal axis. The default value is 1.

HREF < (INTERSECT) > =*value-list*

requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS='*label1*',**...**,'*labeln*'

HREFLABEL='*label1*',**...**,'*labeln*'

HREFLAB='*label1*',**...**,'*labeln*'

specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*

specifies the vertical position of labels for HREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Top
2	Staggered from top
3	Bottom
4	Staggered from bottom
5	Alternating from top
6	Alternating from bottom

HUPPER=*value*
specifies *value* as the upper horizontal axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

INBORDER
requests a border around LPRED plots.

INHEIGHT=*value*
specifies the height of text inside framed areas.

LEVEL=(*character-list***)**
ORDINAL=(*character-list***)**
specifies the names of the levels for which linear predictor lines are requested. Names should be quoted and separated by space. If there is no correct name provided, no LPRED line is plotted.

LFIT=*linetype*
specifies a line style for fitted curves and confidence limits. By default, fitted curves are drawn by connecting solid lines (*linetype* = 1) and confidence limits are drawn by connecting dashed lines (*linetype* = 3).

LGRID=*linetype*
specifies a line style for all grid lines. The value for *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

NAME='*string*'
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'PROBIT'.

NHTICK=*n*
specifies the number of ticks for the horizontal axis.

NOCONF
suppresses confidence limits from the plot. This works only for the binomial model. Confidence limits are not plotted for the multinomial model.

NODATA
suppresses observed data points from the plot. This works only for the binomial model. Data points are not plotted for the multinomial model.

NOFIT
suppresses the fitted LPRED lines.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOTHRESH
suppresses the threshold line.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

NVTICK=*n*
specifies the number of ticks for the vertical axis.

THRESHLABPOS=*n*
specifies the horizontal position of labels for the threshold line. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Left
2	Right

TURNVLABELS
vertically strings out characters in vertical labels.

VAXIS=*value1 to value2* **<** *by value3* **>**
specifies tick mark values for the vertical axis; *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. This method of specification of tick marks is not valid for logarithmic axes. If *value3* is omitted, a value of 1 is used.

Examples of VAXIS= lists follow:

```
vaxis = 0 to 10
vaxis = 0 to 2 by .1
```

VAXISLABEL='*string*'
specifies a label for the vertical axis.

VLOWER=*value*
specifies the lower limit on the vertical axis scale. The VLOWER= option specifies *value* as the lower vertical axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

VOFFSET=*value*
specifies the offset for the vertical axis.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS='*label1*',. . . ,'*labeln*'

VREFLABEL='*label1*',. . . ,'*labeln*'

VREFLAB='*label1*',. . . ,'*labeln*'
specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*
specifies the horizontal position of labels for VREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Left
2	Right

VUPPER=*number*
specifies the upper limit on the vertical axis scale. The VUPPER= option specifies *number* as the upper vertical axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

WAXIS=*n*
specifies line thickness for axes and frame. The default value is 1.

WFIT=*n*
specifies line thickness for fitted lines. The default value is 1.

WGRID=*n*
specifies line thickness for grids. The default value is 1.

WREFL=*n*
specifies line thickness for reference lines. The default value is 1.

PREDPPLOT Statement

PREDPPLOT < **VAR=***variable* > < *options* > **;**

The PREDPPLOT statement plots the predicted probability against a single continuous variable (dose variable) in the MODEL statement for both the binomial model and the multinomial model. Confidence limits are available only for the binomial model. An attached box on the right side of the plot is used to label predicted probability curves with the names of their levels for the multinomial model. You can specify the color of this box by using the CLABBOX= option.

VAR=*variable*
specifies a single continuous variable (dose variable) in the independent variable list of the MODEL statement. If a VAR= variable is not specified, the first single continuous variable in the independent variable list of the MODEL statement is used. If such a variable does not exist in the independent variable list of the MODEL statement, an error is reported.

The predicted probability is

$$\hat{p} = C + (1 - C)F(\mathbf{x}'\hat{\mathbf{b}})$$

for the binomial model and

$$\begin{aligned}
\hat{p}_1 &= C + (1 - C)F(\mathbf{x}'\hat{\mathbf{b}}) \\
\hat{p}_j &= (1 - C)(F(\hat{a}_j + \mathbf{x}'\hat{\mathbf{b}}) - F(\hat{a}_{j-1} + \mathbf{x}'\hat{\mathbf{b}})) \quad j = 2, \ldots, k - 1 \\
\hat{p}_k &= (1 - C)(1 - F(\hat{a}_{k-1} + \mathbf{x}'\hat{\mathbf{b}}))
\end{aligned}$$

for the multinomial model with k response levels, where F is the cumulative distribution function used to model the probability, $\mathbf{x}'$ is the vector of the covariates, $\hat{a}_j$ are the estimated ordinal intercepts with $\hat{a}_1 = 0$, C is the threshold parameter, either known or estimated from the model, and $\hat{\mathbf{b}}'$ is the vector of estimated parameters.

To plot $\hat{p}$ (or $\hat{p}_j$) as a function of a continuous variable x_1, the remaining covariates $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow these rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

options
enable you to plot the observed data and add features to the plot.

You can use *options* in the PREDPPLOT statement to do the following:

- superimpose specification limits
- suppress or add observed data points for the binomial model
- suppress or add confidence limits for the binomial model

- specify the levels for which predicted probability curves are requested for the multinomial model
- specify graphical enhancements (such as color or text height)

Summary of Options

Table 100.34 through Table 100.40 list all *options* by function. The "Dictionary of Options" on page 8326 describes each option in detail.

Table 100.34 Plot Layout Options for the PREDPPLOT Statement

LEVEL=(*character-list*)	Specifies the names of the levels for which the predicted probability curves are requested (only for the multinomial model)
NOCONF	Suppresses confidence limits
NODATA	Suppresses observed data points on the plot
NOTHRESH	Suppresses the threshold line
THRESHLABPOS=*value*	Specifies the position for the label of the threshold line

Table 100.35 Color Options

CAXIS=*color*	Specifies color for the axes
CFIT=*color*	Specifies color for fitted curves
CFRAME=*color*	Specifies color for frame
CGRID=*color*	Specifies color for grid lines
CHREF=*color*	Specifies color for HREF= lines
CLABBOX=*color*	Specifies color for label box
CTEXT=*color*	Specifies color for text
CVREF=*color*	Specifies color for VREF= lines

Table 100.36 Options to Enhance Plots Produced on Graphics Devices

ANNOTATE=*SAS-data-set*	Specifies an Annotate data set
INBORDER	Requests a border around plot
LFIT=*linetype*	Specifies line style for fitted curves and confidence limits
LGRID=*linetype*	Specifies line style for grid lines
NOFIT	Suppresses fitted curves
NOFRAME	Suppresses the frame around plotting areas
NOGRID	Suppresses grid lines
NOHLABEL	Suppresses horizontal labels
NOHTICK	Suppresses horizontal ticks
NOVTICK	Suppresses vertical ticks
TURNVLABELS	Vertically strings out characters in vertical labels
WFIT=*n*	Specifies thickness for fitted curves
WGRID=*n*	Specifies thickness for grids
WREFL=*n*	Specifies thickness for reference lines

Table 100.37 Axis Options

HAXIS=*value1 to value2 < by value3 >*	Specifies tick mark values for horizontal axis
HLOWER=*value*	Specifies lower limit on horizontal axis scale
HOFFSET=*value*	Specifies offset for horizontal axis
HUPPER=*value*	Specifies upper limit on horizontal axis scale
NHTICK=*n*	Specifies number of ticks for horizontal axis
NVTICK=*n*	Specifies number of ticks for vertical axis
VAXIS=*value1 to value2 < by value3 >*	Specifies tick mark values for vertical axis
VAXISLABEL='*label*'	Specifies label for vertical axis
VLOWER=*value*	Specifies lower limit on vertical axis scale
VOFFSET=*value*	Specifies offset for vertical axis
VUPPER=*value*	Specifies upper limit on vertical axis scale
WAXIS=*n*	Specifies thickness for axis

Table 100.38 Graphics Catalog Options

DESCRIPTION='*string*'	Specifies description for graphics catalog member
NAME='*string*'	Specifies name for plot in graphics catalog

Table 100.39 Options for Text Enhancement

FONT=*font*	Specifies software font for text
HEIGHT=*value*	Specifies height of text outside framed areas
INHEIGHT=*value*	Specifies height of text inside framed areas

Table 100.40 Options for Reference Lines

HREF< (INTERSECT) >=*value-list*	Requests horizontal reference line
HREFLABELS=('label1',. . . ,'labeln')	Specifies labels for HREF= lines
HREFLABPOS=*n*	Specifies vertical position of labels for HREF= lines
LHREF=*linetype*	Specifies line style for HREF= lines
LVREF=*linetype*	Specifies line style for VREF= lines
VREF< (INTERSECT) >=*value-list*	Requests vertical reference line
VREFLABELS=('label1',. . . ,'labeln')	Specifies labels for VREF= lines
VREFLABPOS=*n*	Specifies horizontal position of labels for VREF= lines

Dictionary of Options

The following entries provide detailed descriptions of the *options* in the PREDPPLOT statement.

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an Annotate data set, as described in *SAS/GRAPH: Reference*, that enables you to add features to the predicted probability plot. The ANNOTATE= data set you specify in the PREDPPLOT statement is used for all plots created by the statement.

CAXIS=*color*

CAXES=*color*

specifies the color for the axes and tick marks. This option overrides any COLOR= specifications in an AXIS statement. The default is the first color in the device color list.

CFIT=*color*

specifies the color for the fitted predicted probability curves. The default is the first color in the device color list.

CFRAME=*color*

CFR=*color*

specifies the color for the area enclosed by the axes and frame. This area is not shaded by default.

CGRID=*color*
specifies the color for grid lines. The default is the first color in the device color list.

CHREF=*color*
CH=*color*
specifies the color for lines requested by the HREF= option. The default is the first color in the device color list.

CTEXT=*color*
specifies the color for tick mark values and axis labels. The default is the color specified for the CTEXT= option in the most recent GOPTIONS statement.

CVREF=*color*
CV=*color*
specifies the color for lines requested by the VREF= option. The default is the first color in the device color list.

DESCRIPTION='*string*'
DES='*string*'
specifies a description, up to 40 characters, that appears in the PROC GREPLAY master menu. The default is the variable name.

FONT=*font*
specifies a software font for reference line and axis labels. You can also specify fonts for axis labels in an AXIS statement. The FONT= font takes precedence over the FTEXT= font specified in the most recent GOPTIONS statement. Hardware characters are used by default.

HAXIS=*value1 to value2* **<** *by value3* **>**
specifies tick mark values for the horizontal axis; *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. If *value3* is omitted, a value of 1 is used.

Examples of HAXIS= lists follow:

```
haxis = 0 to 10
haxis = 2 to 10 by 2
haxis = 0 to 200 by 10
```

HEIGHT=*value*
specifies the height of text outside framed areas.

HLOWER=*value*
specifies the lower limit on the horizontal axis scale. The HLOWER= option specifies *value* as the lower horizontal axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HOFFSET=*value*
specifies the offset for the horizontal axis. The default value is 1.

HUPPER=*value*
specifies *value* as the upper horizontal axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the HAXIS= option is used.

HREF < (INTERSECT) > =*value-list*
requests reference lines perpendicular to the horizontal axis. If (INTERSECT) is specified, a second reference line perpendicular to the vertical axis is drawn that intersects the fit line at the same point as the horizontal axis reference line. If a horizontal axis reference line label is specified, the intersecting vertical axis reference line is labeled with the vertical axis value. See also the CHREF=, HREFLABELS=, and LHREF= options.

HREFLABELS='*label1*',. . . ,'*labeln*'
HREFLABEL='*label1*',. . . ,'*labeln*'
HREFLAB='*label1*',. . . ,'*labeln*'
specifies labels for the lines requested by the HREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

HREFLABPOS=*n*
specifies the vertical position of labels for HREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Top
2	Staggered from top
3	Bottom
4	Staggered from bottom
5	Alternating from top
6	Alternating from bottom

INBORDER
requests a border around predicted probability plots.

INHEIGHT=*value*
specifies the height of text inside framed areas.

LEVEL=(*character-list***)**
ORDINAL= (*character-list***)**
specifies the names of the levels for which predicted probability curves are requested. Names should be quoted and separated by space. If there is no correct name provided, no fitted probability curve is plotted.

LFIT=*linetype*
specifies a line style for fitted curves and confidence limits. By default, fitted curves are drawn by connecting solid lines (*linetype* = 1) and confidence limits are drawn by connecting dashed lines (*linetype* = 3).

LGRID=*linetype*
specifies a line style for all grid lines. The value for *linetype* is between 1 and 46. The default is 35.

LHREF=*linetype*
LH=*linetype*
specifies the line type for lines requested by the HREF= option. The default is 2, which produces a dashed line.

LVREF=*linetype*
LV=*linetype*
specifies the line type for lines requested by the VREF= option. The default is 2, which produces a dashed line.

NAME='*string*'
specifies a name for the plot, up to eight characters, that appears in the PROC GREPLAY master menu. The default is 'PROBIT'.

NHTICK=*n*
specifies the number of ticks for the horizontal axis.

NOCONF
suppresses confidence limits from the plot. This works only for the binomial model. Confidence limits are not plotted for the multinomial model.

NODATA
suppresses observed data points from the plot. This works only for the binomial model. The data points are not plotted for the multinomial model.

NOFIT
suppresses the fitted predicted probability curves.

NOFRAME
suppresses the frame around plotting areas.

NOGRID
suppresses grid lines.

NOHLABEL
suppresses horizontal labels.

NOHTICK
suppresses horizontal tick marks.

NOTHRESH
suppresses the threshold line.

NOVLABEL
suppresses vertical labels.

NOVTICK
suppresses vertical tick marks.

NVTICK=*n*
specifies the number of ticks for the vertical axis.

THRESHLABPOS=*n*
specifies the horizontal position of labels for the threshold line. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Left
2	Right

TURNVLABELS
vertically strings out characters in vertical labels.

VAXIS=*value1 to value2* **<** *by value3* **>**
specifies tick mark values for the vertical axis; *value1*, *value2*, and *value3* must be numeric, and *value1* must be less than *value2*. The lower tick mark is *value1*. Tick marks are drawn at increments of *value3*. The last tick mark is the greatest value that does not exceed *value2*. This method of specification of tick marks is not valid for logarithmic axes. If *value3* is omitted, a value of 1 is used.

Examples of VAXIS= lists follow:

```
vaxis = 0 to 10
vaxis = 0 to 2 by .1
```

VAXISLABEL='*string*'
specifies a label for the vertical axis.

VLOWER=*value*
specifies the lower limit on the vertical axis scale. The VLOWER= option specifies *value* as the lower vertical axis tick mark. The tick mark interval and the upper axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

VOFFSET=*value*
specifies the offset for the vertical axis.

VREF=*value-list*
requests reference lines perpendicular to the vertical axis. If (INTERSECT) is specified, a second reference line perpendicular to the horizontal axis is drawn that intersects the fit line at the same point as the vertical axis reference line. If a vertical axis reference line label is specified, the intersecting horizontal axis reference line is labeled with the horizontal axis value. See also the CVREF=, LVREF=, and VREFLABELS= options.

VREFLABELS='*label1*',. . . ,'*labeln*'

VREFLABEL='*label1*',. . . ,'*labeln*'

VREFLAB='*label1*',. . . ,'*labeln*'

specifies labels for the lines requested by the VREF= option. The number of labels must equal the number of lines. Enclose each label in quotes. Labels can be up to 16 characters.

VREFLABPOS=*n*

specifies the horizontal position of labels for VREF= lines. The following table shows valid values for *n* and the corresponding label placements.

n	**Label Placement**
1	Left
2	Right

VUPPER=*value*

specifies the upper limit on the vertical axis scale. The VUPPER= option specifies *value* as the upper vertical axis tick mark. The tick mark interval and the lower axis limit are determined automatically. This option has no effect if the VAXIS= option is used.

WAXIS=*n*

specifies line thickness for axes and frame. The default value is 1.

WFIT=*n*

specifies line thickness for fitted curves. The default value is 1.

WGRID=*n*

specifies line thickness for grids. The default value is 1.

WREFL=*n*

specifies line thickness for reference lines. The default value is 1.

Displayed Output

If you request the iteration history (ITPRINT), PROC PROBIT displays the following:

- the current value of the log likelihood
- the ridging parameter for the modified Newton-Raphson optimization process
- the current estimate of the parameters
- the current estimate of the parameter C for a natural (threshold) model
- the values of the gradient and the Hessian on the last iteration

If you include classification variables, PROC PROBIT displays the following:

- the numbers of levels for each classification variable
- the (ordered) values of the levels
- the number of observations used

After the model is fit, PROC PROBIT displays the following:

- the name of the input data set
- the name of the dependent variables
- the number of observations used
- the number of events and the number of trials
- the final value of the log-likelihood function
- the parameter estimates
- the standard error estimates of the parameter estimates
- approximate chi-square test statistics for the test

If you specify the COVB or CORRB options, PROC PROBIT displays the following:

- the estimated covariance matrix for the parameter estimates
- the estimated correlation matrix for the parameter estimates

If you specify the LACKFIT option, PROC PROBIT displays the following:

- a count of the number of levels of the response and the number of distinct sets of independent variables
- a goodness-of-fit test based on the Pearson's chi-square
- a goodness-of-fit test based on the likelihood-ratio chi-square

If you specify only one independent variable, the normal distribution is used to model the probabilities, and the response is binary, then PROC PROBIT displays the following:

- the mean MU of the stimulus tolerance
- the scale parameter SIGMA of the stimulus tolerance
- the covariance matrix for MU, SIGMA, and the natural response parameter C

If you specify the INVERSECL options, PROC PROBIT also displays the following:

- the estimated dose along with the 95% fiducial limits for probability levels 0.01 to 0.10, 0.15 to 0.85 by 0.05, and 0.90 to 0.99

ODS Table Names

PROC PROBIT assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 100.41 ODS Tables Produced by PROC PROBIT

ODS Table Name	Description	Statement	Option
ClassLevels	Classification variable levels	CLASS	Default
ConvergenceStatus	Convergence status	MODEL	Default
CorrB	Parameter estimate correlation matrix	MODEL	CORRB
CovB	Parameter estimate covariance matrix	MODEL	COVB
CovTolerance	Covariance matrix for location and scale	MODEL	Default
GoodnessOfFit	Goodness-of-fit tests	MODEL	LACKFIT
Heterogeneity	Heterogeneity correction	MODEL	LACKFIT
IterHistory	Iteration history	MODEL	ITPRINT
LagrangeStatistics	Lagrange statistics	MODEL	NOINT
LastGrad	Last evaluation of the gradient	MODEL	ITPRINT
LastHess	Last evaluation of the Hessian	MODEL	ITPRINT
LogProbitAnalysis	Probit analysis for log dose	MODEL	INVERSECL
ModelInfo	Model information	MODEL	Default
MuSigma	Location and scale	MODEL	Default
NObs	Observations summary	PROC PROBIT	Default
ParameterEstimates	Parameter estimates	MODEL	Default
ParmInfo	Parameter indices	MODEL	Default
ProbitAnalysis	Probit analysis for linear dose	MODEL	INVERSECL
ResponseLevels	Response-covariate profile	MODEL	LACKFIT
ResponseProfiles	Counts for ordinal data	MODEL	Default
Type3Analysis	Type III tests	MODEL	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

These ODS graphs are controlled by the PLOTS= option in the PROC PROBIT statement. You can specify more than one graph request with the PLOTS= option. Table 100.42 summarizes these requests.

Table 100.42 Options for Plots

Option	Plot
ALL	All appropriate plots
CDFPLOT	Estimated cumulative probability
IPPPLOT	Inverse predicted probability
LPREDPLOT	Linear predictor
NONE	No plot
PREDPPLOT	Predicted probability

The following subsections provide information about these graphs.

ODS Graph Names

PROC PROBIT assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 100.43.

Table 100.43 Graphs Produced by PROC PROBIT

ODS Graph Name	Plot Description	Statement	PLOTS= Option
CDFPlot	Estimated cumulative probability	PROC PROBIT	CDFPLOT
IPPPlot	Inverse predicted probability	PROC PROBIT	IPPPLOT
LPredPlot	Linear predictor	PROC PROBIT	LPREDPLOT
PredPPlot	Predicted probability	PROC PROBIT	PREDPPLOT

CDF Plot

For a multinomial model, the predicted cumulative distribution function is defined as

$$\hat{F}_j(\mathbf{x}) = C + (1 - C)F(\hat{a}_j + \mathbf{x}'\hat{\mathbf{b}})$$

where $j = 1, \ldots, k$ are the indexes of the k levels of the multinomial response variable, F is the CDF of the distribution used to model the cumulative probabilities, $\hat{\mathbf{b}}$ is the vector of estimated parameters, $\mathbf{x}$ is the covariate vector, $\hat{a}_j$ are estimated ordinal intercepts with $\hat{a}_1 = 0$, and C is the threshold parameter, either known or estimated from the model. Let x_1 be the covariate corresponding to the dose variable and $\mathbf{x}_{-1}$ be the vector of the rest of the covariates. Let the corresponding estimated parameters be $\hat{b}_1$ and $\hat{\mathbf{b}}_{-1}$. Then

$$\hat{F}_j(\mathbf{x}) = C + (1 - C)F(\hat{a}_j + x_1\hat{b}_1 + \mathbf{x}'_{-1}\hat{\mathbf{b}}_{-1})$$

To plot $\hat{F}_j$ as a function of x_1, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow these rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.

- If the effect is a single classification variable, the highest level of the variable is used.

The LEVEL= suboption specify the levels of the multinomial response variable for which the CDF curves are requested. There are $k - 1$ curves for a k-level multinomial response variable (for the highest level, it is the constant line 1). You can specify any of them to be plotted by the LEVEL= suboption. See the plot in Output 100.2.6 for an example.

Inverse Predicted Probability Plot

For the binomial model, the response variable is a probability. An estimate of the dose level $\hat{x}_1$ needed for a response of p is given by

$$\hat{x}_1 = (F^{-1}(p) - \mathbf{x}'_{-1}\hat{\mathbf{b}}_{-1})/\hat{b}_1$$

where F is the cumulative distribution function used to model the probability, $\mathbf{x}_{-1}$ is the vector of the rest of the covariates, $\hat{\mathbf{b}}_{-1}$ is the vector of the estimated parameters corresponding to $\mathbf{x}_{-1}$, and $\hat{b}_1$ is the estimated parameter for the dose variable of interest.

To plot $\hat{x}_1$ as a function of p, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow these rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

Output 100.4.12 in Example 100.4 shows an inverse predicted probability plot.

Linear Predictor Plot

For both binomial models and multinomial models, the linear predictor $\mathbf{x}'\mathbf{b}$ can be plotted against the first single continuous variable (dose variable) in the MODEL statement.

Let x_1 be the covariate of the dose variable, $\mathbf{x}_{-1}$ be the vector of the rest of the covariates, $\hat{\mathbf{b}}_{-1}$ be the vector of estimated parameters corresponding to $\mathbf{x}_{-1}$, and $\hat{b}_1$ be the estimated parameter for the dose variable of interest.

To plot $\hat{\mathbf{x}}'\mathbf{b}$ as a function of x_1, $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow these rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

For the multinomial model, you can use the LEVEL= suboption to specify the levels for which the linear predictor lines are plotted.

The confidence limits for the predicted values are only available for the binomial model. Output 100.4.13 in Example 100.4 shows a linear predictor plot for a binomial model.

Predicted Probability Plot

The predicted probability is

$$\hat{p} = C + (1 - C)F(\mathbf{x}'\hat{\mathbf{b}})$$

for the binomial model and

$$\begin{aligned}
\hat{p}_1 &= C + (1 - C)F(\mathbf{x}'\hat{\mathbf{b}}) \\
\hat{p}_j &= (1 - C)(F(\hat{a}_j + \mathbf{x}'\hat{\mathbf{b}}) - F(\hat{a}_{j-1} + \mathbf{x}'\hat{\mathbf{b}})), \quad j = 2, \ldots, k - 1 \\
\hat{p}_k &= (1 - C)(1 - F(\hat{a}_{k-1} + \mathbf{x}'\hat{\mathbf{b}}))
\end{aligned}$$

for the multinomial model with k response levels, where F is the cumulative distribution function used to model the probability, $\mathbf{x}'$ is the vector of the covariates, $\hat{a}_j$ are the estimated ordinal intercepts with $\hat{a}_1 = 0$, C is the threshold parameter, either known or estimated from the model, and $\hat{\mathbf{b}}'$ is the vector of estimated parameters.

To plot $\hat{p}$ (or $\hat{p}_j$) as a function of a continuous variable x_1, the remaining covariates $\mathbf{x}_{-1}$ must be specified. You can use the XDATA= option to provide the values of $\mathbf{x}_{-1}$ (see the XDATA= option in the PROC PROBIT statement for details), or use the default values that follow these rules:

- If the effect contains a continuous variable (or variables), the overall mean of this effect is used.
- If the effect is a single classification variable, the highest level of the variable is used.

For the multinomial model, you can use the LEVEL= suboption to specify the levels for which the linear predictor lines are plotted.

Confidence limits are plotted only for the binomial model. Output 100.1.7 in Example 100.1 shows a predicted probability plot for a binomial model; and Output 100.2.3 in Example 100.2 shows a predicted probability plot for a multinomial model.

Examples: PROBIT Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 100.1: Dosage Levels

(View the complete code for this example (prbtex1.sas) in the example repository.)

In this example, Dose is a variable representing the level of a stimulus, N represents the number of subjects tested at each level of the stimulus, and Response is the number of subjects responding to that level of the stimulus. Both probit and logit response models are fit to the data. The LOG10 option in the PROC PROBIT statement requests that the log base 10 of Dose is used as the independent variable. Specifically, for a given level of Dose, the probability p of a positive response is modeled as

$$p = \Pr(\text{Response}) = F\left(b_0 + b_1 \times \log_{10}(\text{Dose})\right)$$

The probabilities are estimated first by using the normal distribution function (the default) and then by using the logistic distribution function. Note that, in this model specification, the natural rate is assumed to be zero.

The LACKFIT option specifies lack-of-fit tests and the INVERSECL option specifies inverse confidence limits.

In the DATA step that reads the data, a number of observations are generated that have a missing value for the response. Although the PROBIT procedure does not use the observations with the missing values to fit the model, it does give predicted values for all nonmissing sets of independent variables. These data points fill in the plot of fitted and observed values in the logistic model displayed in Output 100.1.7. The plot, requested with the PLOTS=PREDPPLOT option, displays the estimated logistic cumulative distribution function and the observed response rates.

The following statements produce Output 100.1.1:

```
data a;
   infile cards eof=eof;
   input Dose N Response @@;
   Observed= Response/N;
   output;
   return;
eof: do Dose=0.5 to 7.5 by 0.25;
        output;
     end;
   datalines;
1 10 1  2 12 2  3 10 4  4 10 5
5 12 8  6 10 8  7 10 10
;
```

```
ods graphics on;

proc probit log10;
   model Response/N=Dose / lackfit inversecl itprint;
   output out=B p=Prob std=std xbeta=xbeta;
run;
```

Output 100.1.1 Probit Analysis with Normal Distribution

The Probit Procedure

Iteration History for Parameter Estimates				
Iter	**Ridge**	**Loglikelihood**	**Intercept**	**Log10(Dose)**
0	0	-51.292891	0	0
1	0	-37.881166	-1.355817008	2.635206083
2	0	-37.286169	-1.764939171	3.3408954936
3	0	-37.280389	-1.812147863	3.4172391614
4	0	-37.280388	-1.812704962	3.418117919
5	0	-37.280388	-1.812704962	3.418117919

Model Information	
Data Set	WORK.A
Events Variable	Response
Trials Variable	N
Number of Observations	7
Number of Events	38
Number of Trials	74
Name of Distribution	Normal
Log Likelihood	-37.28038802

Last Evaluation of the Negative of the Gradient	
Intercept	**Log10(Dose)**
3.4349069E-7	-2.09809E-8

Last Evaluation of the Negative of the Hessian		
	Intercept	**Log10(Dose)**
Intercept	36.005280383	20.152675982
Log10(Dose)	20.152675982	13.078826305

Goodness-of-Fit Tests				
Statistic	**Value**	**DF**	**Value/DF**	**Pr > ChiSq**
Pearson Chi-Square	3.6497	5	0.7299	0.6009
L.R. Chi-Square	4.6381	5	0.9276	0.4616

Response-Covariate Profile	
Response Levels	2
Number of Covariate Values	7

Output 100.1.1 *continued*

Analysis of Maximum Likelihood Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-1.8127	0.4493	-2.6934	-0.9320	16.27	<.0001
Log10(Dose)	1	3.4181	0.7455	1.9569	4.8794	21.02	<.0001

Probit Model in Terms of Tolerance Distribution	
MU	SIGMA
0.53032254	0.29255866

Estimated Covariance Matrix for Tolerance Parameters		
	MU	SIGMA
MU	0.002418	-0.000409
SIGMA	-0.000409	0.004072

The *p*-values in the goodness-of-fit table of 0.6009 for the Pearson's chi-square and 0.4616 for the likelihood ratio chi-square indicate an adequate fit for the model fit with the normal distribution.

Tolerance distribution parameter estimates for the normal distribution indicate a mean tolerance for the population of 0.5303.

Output 100.1.2 displays probit analysis with the logarithm of dose levels. The LD50 (ED50 for log dose) is 0.5303, the dose corresponding to a probability of 0.5. This is the same as the mean tolerance for the normal distribution.

Output 100.1.2 Probit Analysis with Normal Distribution

The Probit Procedure

Probit Analysis on Log10(Dose)			
		95%	
Probability	Log10(Dose)	Fiducial Limits	
0.01	-0.15027	-0.69518	0.07710
0.02	-0.07052	-0.55766	0.13475
0.03	-0.01992	-0.47064	0.17156
0.04	0.01814	-0.40534	0.19941
0.05	0.04911	-0.35233	0.22218
0.06	0.07546	-0.30731	0.24165
0.07	0.09857	-0.26793	0.25881
0.08	0.11926	-0.23273	0.27425
0.09	0.13807	-0.20080	0.28837
0.10	0.15539	-0.17147	0.30142
0.15	0.22710	-0.05086	0.35631
0.20	0.28410	0.04369	0.40124
0.25	0.33299	0.12343	0.44116
0.30	0.37690	0.19348	0.47857
0.35	0.41759	0.25658	0.51504
0.40	0.45620	0.31429	0.55182
0.45	0.49356	0.36754	0.58999
0.50	0.53032	0.41693	0.63057
0.55	0.56709	0.46296	0.67451
0.60	0.60444	0.50618	0.72271
0.65	0.64305	0.54734	0.77603
0.70	0.68374	0.58745	0.83550
0.75	0.72765	0.62776	0.90265
0.80	0.77655	0.66999	0.98008
0.85	0.83354	0.71675	1.07279
0.90	0.90525	0.77313	1.19191
0.91	0.92257	0.78646	1.22098
0.92	0.94139	0.80083	1.25265
0.93	0.96208	0.81653	1.28759
0.94	0.98519	0.83394	1.32672
0.95	1.01154	0.85367	1.37149
0.96	1.04250	0.87669	1.42424
0.97	1.08056	0.90480	1.48928
0.98	1.13116	0.94189	1.57602
0.99	1.21092	0.99987	1.71321

Output 100.1.3 displays probit analysis with dose levels. The ED50 for dose is 3.39 with a 95% confidence interval of (2.61, 4.27).

Output 100.1.3 Probit Analysis with Normal Distribution

The Probit Procedure

Probit Analysis on Dose			
		95% Fiducial Limits	
Probability	**Dose**		
0.01	0.70750	0.20175	1.19427
0.02	0.85012	0.27691	1.36380
0.03	0.95517	0.33834	1.48444
0.04	1.04266	0.39324	1.58274
0.05	1.11971	0.44429	1.66793
0.06	1.18976	0.49282	1.74443
0.07	1.25478	0.53960	1.81473
0.08	1.31600	0.58515	1.88042
0.09	1.37427	0.62980	1.94252
0.10	1.43019	0.67380	2.00181
0.15	1.68696	0.88950	2.27147
0.20	1.92353	1.10584	2.51906
0.25	2.15276	1.32870	2.76161
0.30	2.38180	1.56128	3.01000
0.35	2.61573	1.80543	3.27374
0.40	2.85893	2.06200	3.56306
0.45	3.11573	2.33098	3.89038
0.50	3.39096	2.61175	4.27138
0.55	3.69051	2.90374	4.72619
0.60	4.02199	3.20759	5.28090
0.65	4.39594	3.52651	5.97077
0.70	4.82770	3.86765	6.84706
0.75	5.34134	4.24385	7.99189
0.80	5.97787	4.67724	9.55169
0.85	6.81617	5.20900	11.82480
0.90	8.03992	5.93105	15.55653
0.91	8.36704	6.11584	16.63320
0.92	8.73752	6.32165	17.89163
0.93	9.16385	6.55431	19.39034
0.94	9.66463	6.82245	21.21881
0.95	10.26925	7.13949	23.52275
0.96	11.02811	7.52816	26.56066
0.97	12.03830	8.03149	30.85201
0.98	13.52585	8.74763	37.67206
0.99	16.25233	9.99709	51.66627

The following statements request probit analysis of dosage levels with the logistic distribution:

```
proc probit log10 plot=predpplot;
   model Response/N=Dose / d=logistic inversecl;
   output out=B p=Prob std=std xbeta=xbeta;
run;
```

The regression parameter estimates in Output 100.1.4 for the logistic model of –3.22 and 5.97 are approximately $\pi/\sqrt{3}$ times as large as those for the normal model.

Output 100.1.4 Probit Analysis with Logistic Distribution

The Probit Procedure

Model Information	
Data Set	WORK.B
Events Variable	Response
Trials Variable	N
Number of Observations	7
Number of Events	38
Number of Trials	74
Name of Distribution	Logistic
Log Likelihood	-37.11065336

Analysis of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-3.2246	0.8861	-4.9613	-1.4880	13.24	0.0003
Log10(Dose)	1	5.9702	1.4492	3.1299	8.8105	16.97	<.0001

Output 100.1.5 and Output 100.1.6 show that both the ED50 and the LD50 are similar to those for the normal model.

Output 100.1.5 Probit Analysis with Logistic Distribution

The Probit Procedure

Probit Analysis on Log10(Dose)			
		95%	
Probability	**Log10(Dose)**	**Fiducial Limits**	
0.01	-0.22955	-0.97441	0.04234
0.02	-0.11175	-0.75158	0.12404
0.03	-0.04212	-0.62018	0.17265
0.04	0.00780	-0.52618	0.20771
0.05	0.04693	-0.45265	0.23533
0.06	0.07925	-0.39205	0.25826
0.07	0.10686	-0.34037	0.27796
0.08	0.13103	-0.29521	0.29530
0.09	0.15259	-0.25502	0.31085
0.10	0.17209	-0.21875	0.32498
0.15	0.24958	-0.07552	0.38207
0.20	0.30792	0.03092	0.42645
0.25	0.35611	0.11742	0.46451
0.30	0.39820	0.19143	0.49932
0.35	0.43644	0.25684	0.53275
0.40	0.47221	0.31588	0.56619
0.45	0.50651	0.36986	0.60089
0.50	0.54013	0.41957	0.63807
0.55	0.57374	0.46559	0.67894
0.60	0.60804	0.50846	0.72474
0.65	0.64381	0.54896	0.77673
0.70	0.68205	0.58815	0.83637
0.75	0.72414	0.62752	0.90582
0.80	0.77233	0.66915	0.98876
0.85	0.83067	0.71631	1.09242
0.90	0.90816	0.77562	1.23343
0.91	0.92766	0.79014	1.26931
0.92	0.94922	0.80607	1.30912
0.93	0.97339	0.82378	1.35391
0.94	1.00100	0.84384	1.40523
0.95	1.03332	0.86713	1.46546
0.96	1.07245	0.89511	1.53864
0.97	1.12237	0.93053	1.63228
0.98	1.19200	0.97952	1.76329
0.99	1.30980	1.06166	1.98569

Output 100.1.6 Probit Analysis with Logistic Distribution

The Probit Procedure

Probit Analysis on Dose			
		95%	
Probability	Dose	Fiducial Limits	
0.01	0.58945	0.10607	1.10241
0.02	0.77312	0.17718	1.33058
0.03	0.90757	0.23978	1.48817
0.04	1.01813	0.29773	1.61327
0.05	1.11413	0.35266	1.71922
0.06	1.20018	0.40546	1.81244
0.07	1.27896	0.45670	1.89654
0.08	1.35218	0.50675	1.97379
0.09	1.42100	0.55588	2.04572
0.10	1.48625	0.60430	2.11339
0.15	1.77656	0.84038	2.41030
0.20	2.03199	1.07379	2.66961
0.25	2.27043	1.31046	2.91416
0.30	2.50152	1.55393	3.15736
0.35	2.73172	1.80652	3.40996
0.40	2.96627	2.06957	3.68292
0.45	3.21006	2.34345	3.98927
0.50	3.46837	2.62768	4.34578
0.55	3.74746	2.92138	4.77466
0.60	4.05546	3.22451	5.30573
0.65	4.40366	3.53961	5.98041
0.70	4.80891	3.87391	6.86079
0.75	5.29836	4.24155	8.05044
0.80	5.92009	4.66820	9.74455
0.85	6.77126	5.20365	12.37149
0.90	8.09391	5.96508	17.11715
0.91	8.46559	6.16800	18.59129
0.92	8.89644	6.39837	20.37592
0.93	9.40575	6.66469	22.58957
0.94	10.02317	6.97977	25.42292
0.95	10.79732	7.36428	29.20549
0.96	11.81534	7.85438	34.56521
0.97	13.25466	8.52173	42.88232
0.98	15.55972	9.53941	57.98207
0.99	20.40815	11.52549	96.75820

The PLOTS=PREDPPLOT option together with the ODS GRAPHICS statement creates the plot of observed and fitted probabilities in Output 100.1.7. The shaded region is the pointwise confidence band for the probabilities.

Output 100.1.7 Plot of Observed and Fitted Probabilities

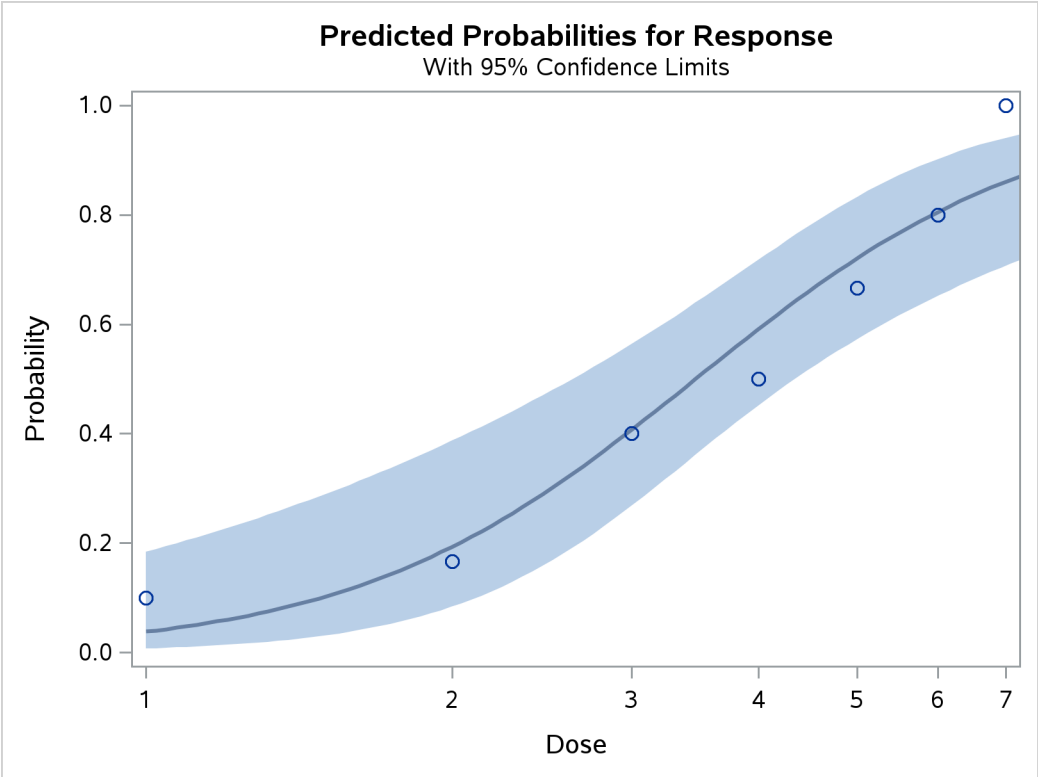

Example 100.2: Multilevel Response

(View the complete code for this example (prbtex2.sas) in the example repository.)

In this example, two preparations, a standard preparation and a test preparation, are each given at several dose levels to groups of insects. The symptoms are recorded for each insect within each group, and two multilevel probit models are fit. Because the natural sort order of the three levels is not the same as the response order, the ORDER=DATA response variable option is specified in the MODEL statement to get the desired order.

The following statements fit two models:

```
data multi;
   input Prep $ Dose Symptoms $ N;
   LDose=log10(Dose);
   if Prep='test' then PrepDose=LDose;
   else PrepDose=0;
   datalines;
stand     10      None       33
stand     10      Mild        7
stand     10      Severe     10
stand     20      None       17
stand     20      Mild       13
stand     20      Severe     17
stand     30      None       14
stand     30      Mild        3
```

```
stand     30      Severe     28
stand     40      None        9
stand     40      Mild        8
stand     40      Severe     32
test      10      None       44
test      10      Mild        6
test      10      Severe      0
test      20      None       32
test      20      Mild       10
test      20      Severe     12
test      30      None       23
test      30      Mild        7
test      30      Severe     21
test      40      None       16
test      40      Mild        6
test      40      Severe     19
;

proc probit data=multi;
   class Prep;
   nonpara: model Symptoms(order=data)=Prep LDose PrepDose / lackfit;
   weight N;
run;

proc probit data=multi;
   class Prep;
   parallel: model Symptoms(order=data)=Prep LDose / lackfit;
   weight N;
run;
```

Results of these two models are shown in Output 100.2.1 and Output 100.2.2. The first model allows for nonparallelism between the dose response curves for the two preparations by inclusion of an interaction between Prep and LDose. The interaction term is labeled PrepDose in the "Analysis of Parameter Estimates" table. The results of this first model indicate that the parameter for the interaction term is not significant, having a Wald chi-square of 0.73. Also, since the first model is a generalization of the second, a likelihood ratio test statistic for this same parameter can be obtained by multiplying the difference in log likelihoods between the two models by 2. The value obtained, $2 \times (-345.94 - (-346.31))$, is 0.73. This is in close agreement with the Wald chi-square from the first model. The lack-of-fit test statistics for the two models do not indicate a problem with either fit.

Output 100.2.1 Multilevel Response: Nonparallel Analysis

The Probit Procedure

Model Information	
Data Set	WORK.MULTI
Dependent Variable	Symptoms
Weight Variable	N
Number of Observations	23
Name of Distribution	Normal
Log Likelihood	-345.9401767

Output 100.2.1 *continued*

Class Level Information		
Name	**Levels**	**Values**
Prep	2	stand test
Symptoms	3	None Mild Severe

Analysis of Maximum Likelihood Parameter Estimates								
Parameter		**DF**	**Estimate**	**Standard Error**	**95% Confidence Limits**		**Chi-Square**	**Pr > ChiSq**
Intercept		1	3.8080	0.6252	2.5827	5.0333	37.10	<.0001
Intercept2		1	0.4684	0.0559	0.3589	0.5780	70.19	<.0001
Prep	stand	1	-1.2573	0.8190	-2.8624	0.3479	2.36	0.1247
Prep	test	0	0.0000	.	.	.	.	.
LDose		1	-2.1512	0.3909	-2.9173	-1.3851	30.29	<.0001
PrepDose		1	-0.5072	0.5945	-1.6724	0.6580	0.73	0.3935

Output 100.2.2 Multilevel Response: Parallel Analysis

The Probit Procedure

Model Information	
Data Set	WORK.MULTI
Dependent Variable	Symptoms
Weight Variable	N
Number of Observations	23
Name of Distribution	Normal
Log Likelihood	-346.306141

Class Level Information		
Name	**Levels**	**Values**
Prep	2	stand test
Symptoms	3	None Mild Severe

Analysis of Maximum Likelihood Parameter Estimates								
Parameter		**DF**	**Estimate**	**Standard Error**	**95% Confidence Limits**		**Chi-Square**	**Pr > ChiSq**
Intercept		1	3.4148	0.4126	2.6061	4.2235	68.50	<.0001
Intercept2		1	0.4678	0.0558	0.3584	0.5772	70.19	<.0001
Prep	stand	1	-0.5675	0.1259	-0.8142	-0.3208	20.33	<.0001
Prep	test	0	0.0000	.	.	.	.	.
LDose		1	-2.3721	0.2949	-2.9502	-1.7940	64.68	<.0001

The negative coefficient associated with LDose indicates that the probability of having no symptoms (Symptoms='None') or no or mild symptoms (Symptoms='None' or Symptoms='Mild') decreases as LDose increases; that is, the probability of a severe symptom increases with LDose. This association is apparent for both treatment groups.

The negative coefficient associated with the standard treatment group (Prep = stand) indicates that the

standard treatment is associated with more severe symptoms across all Ldose values.

The following statements use the PLOTS= option to create the plot shown in Output 100.2.3 and Output 100.2.4. Output 100.2.3 is the plot of the probabilities of the response taking on individual levels as a function of LDose. Since there are two covariates, LDose and Prep, the value of the classification variable Prep is fixed at the highest level, test. Instead of individual response level probabilities, the PLOTS=CDFPLOT option creates the plot of the cumulative response probabilities with confidence bands shown in Output 100.2.4.

```
proc probit data=multi
            plots=(predpplot(level=("None" "Mild" "Severe"))
                   cdfplot(level=("None" "Mild" "Severe")));
   class Prep;
   parallel: model Symptoms(order=data)=Prep LDose / lackfit;
   weight N;
run;
```

Output 100.2.3 Plot of Predicted Probabilities for the Test Preparation Group

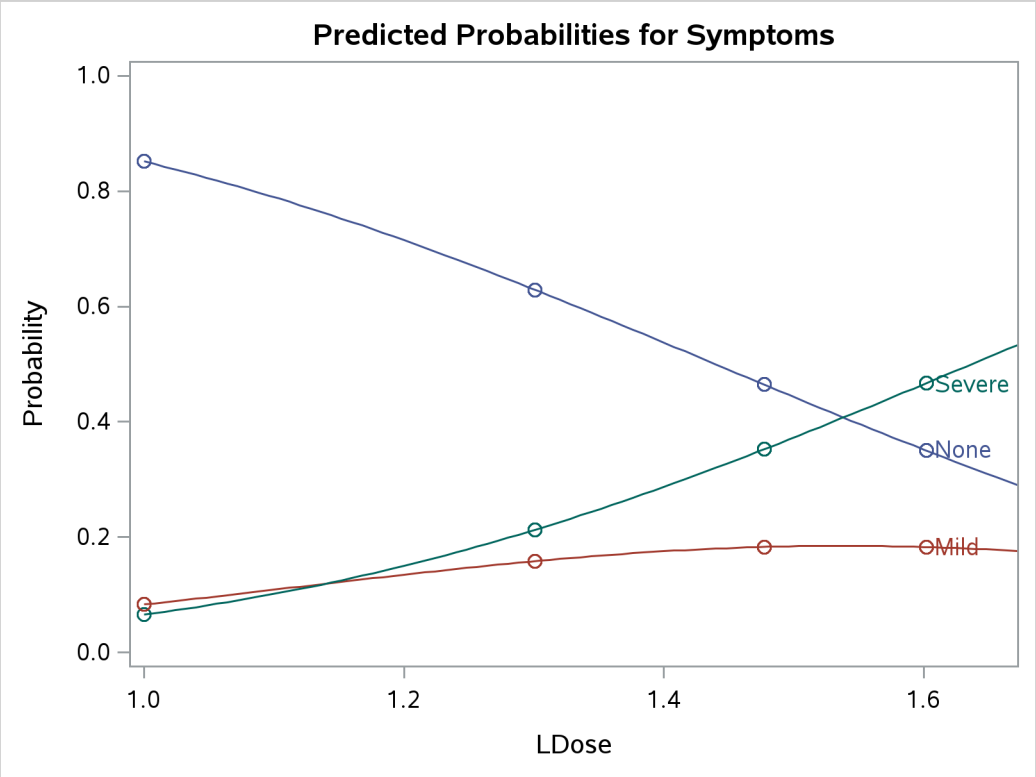

Output 100.2.4 Plot of Predicted Cumulative Probabilities for the Test Preparation Group

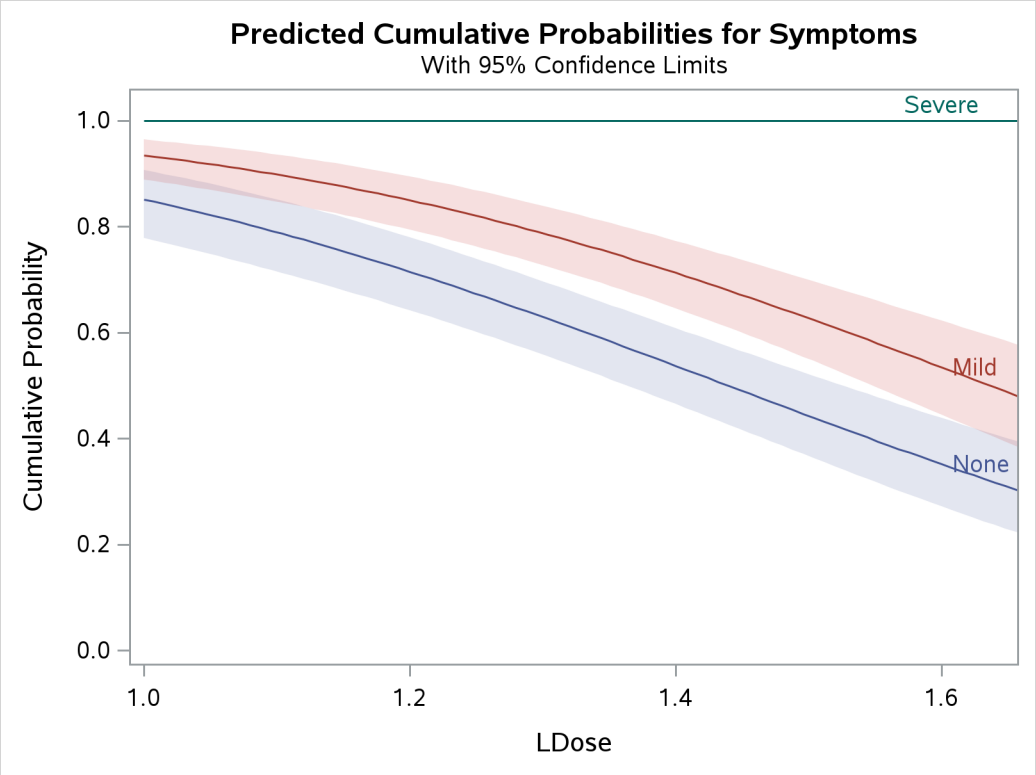

The following statements use the XDATA= data set to create plots of predicted probabilities and cumulative probabilities with Prep set to the stand level. The resulting plots are shown in Output 100.2.5 and Output 100.2.6.

```
data xrow;
   input Prep $ Dose Symptoms $ N;
   LDose=log10(Dose);
   datalines;
stand     40      Severe     32
run;

proc probit data=multi xdata=xrow
            plots=(predpplot(level=("None" "Mild" "Severe"))
                   cdfplot(level=("None" "Mild" "Severe")));
   class Prep;
   parallel: model Symptoms(order=data)=Prep LDose / lackfit;
   weight N;
run;
```

Output 100.2.5 Plot of Predicted Probabilities for the Standard Preparation Group

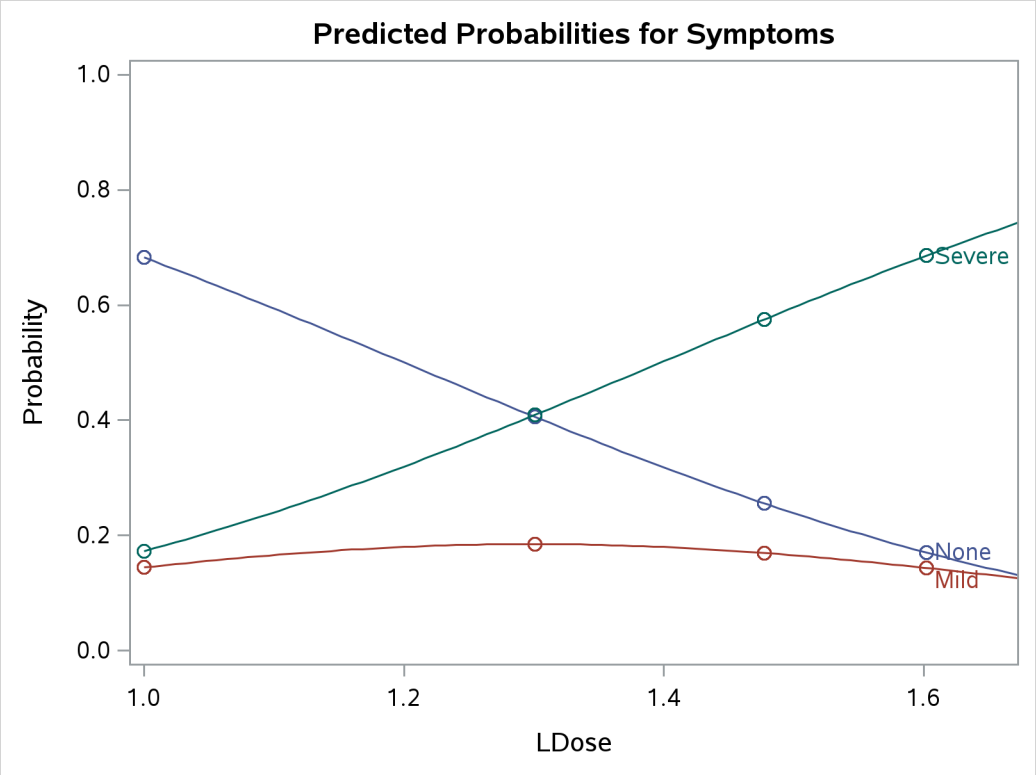

Output 100.2.6 Plot of Predicted Cumulative Probabilities for the Standard Preparation Group

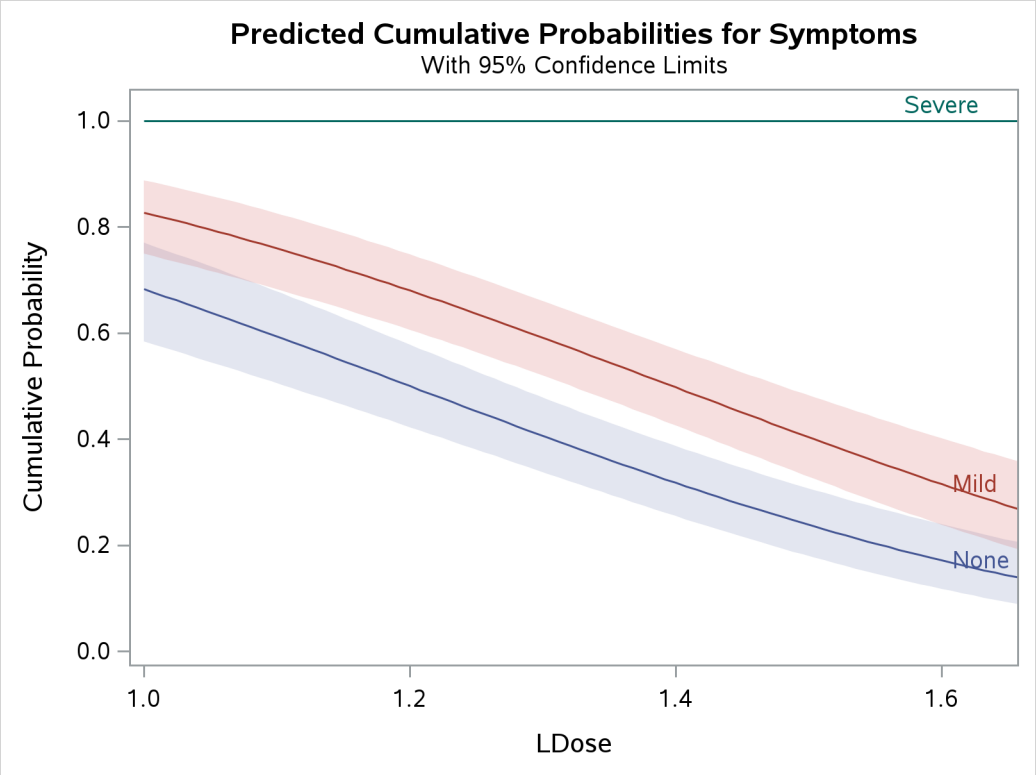

Example 100.3: Logistic Regression and Scoring New Data

(View the complete code for this example (prbtex3.sas) in the example repository.)

In this example, a series of people are asked whether or not they would subscribe to a new newspaper. For each person, the variables sex (Female, Male), age, and subs (1=yes, 0=no) are recorded. The PROBIT procedure is used to fit a logistic regression model to the probability of subscribing (subs = 1) as a function of the variables sex and age. Specifically, the probability of subscribing is modeled as

$$p = \Pr(\text{subs} = 1) = F\left(b_0 + b_1 \times \text{sex} + b_2 \times \text{age}\right)$$

where F is the cumulative logistic distribution function.

By default, the PROBIT procedure models the probability of the lower response level. The following statements format the values of subs as 1 = 'accept' and 0 = 'reject', and model $\Pr(\text{subs} = 1)$ by using the EVENT= response variable option. The STORE statement saves the fitted model in an item store named LogitModel. The results are shown in Output 100.3.1.

```
data news;
   input sex $ age subs @@;
   datalines;
Female     35    0   Male       44    0
Male       45    1   Female     47    1
Female     51    0   Female     47    0
Male       54    1   Male       47    1
Female     35    0   Female     34    0
Female     48    0   Female     56    1
Male       46    1   Female     59    1
Female     46    1   Male       59    1
Male       38    1   Female     39    0
Male       49    1   Male       42    1
Male       50    1   Female     45    0
Female     47    0   Female     30    1
Female     39    0   Female     51    0
Female     45    0   Female     43    1
Male       39    1   Male       31    0
Female     39    0   Male       34    0
Female     52    1   Female     46    0
Male       58    1   Female     50    1
Female     32    0   Female     52    1
Female     35    0   Female     51    0
;

proc format;
   value subscrib 1 = 'accept' 0 = 'reject';
run;

proc probit data=news;
   class sex;
   model subs(event="accept")=sex age / d=logistic itprint;
   format subs subscrib.;
   store out=LogitModel;
run;
```

Output 100.3.1 Logistic Regression of Subscription Status

The Probit Procedure

Iteration History for Parameter Estimates					
Iter	**Ridge**	**Loglikelihood**	**Intercept**	**sexFemale**	**age**
0	0	-27.725887	0	0	0
1	0	-20.142659	-3.634567629	-1.648455751	0.1051634384
2	0	-19.52245	-5.254865196	-2.234724956	0.1506493473
3	0	-19.490439	-5.728485385	-2.409827238	0.1639621828
4	0	-19.490303	-5.76187293	-2.422349862	0.1649007124
5	0	-19.490303	-5.7620267	-2.422407743	0.1649050312
6	0	-19.490303	-5.7620267	-2.422407743	0.1649050312

Model Information	
Data Set	WORK.NEWS
Dependent Variable	subs
Number of Observations	40
Name of Distribution	Logistic
Log Likelihood	-19.49030281

Class Level Information		
Name	**Levels**	**Values**
sex	2	Female Male
subs	2	accept reject

Last Evaluation of the Negative of the Gradient		
Intercept	**sexFemale**	**age**
-5.95457E-12	8.768328E-10	-1.636696E-8

Last Evaluation of the Negative of the Hessian			
	Intercept	**sexFemale**	**age**
Intercept	6.4597397447	4.6042218284	292.04051848
sexFemale	4.6042218284	4.6042218284	216.20829515
age	292.04051848	216.20829515	13487.329973

Analysis of Maximum Likelihood Parameter Estimates								
Parameter		**DF**	**Estimate**	**Standard Error**	**95% Confidence Limits**		**Chi-Square**	**Pr > ChiSq**
Intercept		1	-5.7620	2.7635	-11.1783	-0.3458	4.35	0.0371
sex	Female	1	-2.4224	0.9559	-4.2959	-0.5489	6.42	0.0113
sex	Male	0	0.0000	.	.	.	.	.
age		1	0.1649	0.0652	0.0371	0.2927	6.40	0.0114

Output 100.3.1 shows that there appears to be an effect due to both the variables sex and age. The positive coefficient for age indicates that older people are more likely to subscribe than younger people. The negative coefficient for sex indicates that females are less likely to subscribe than males.

You can use the SCORE statement in the PLM procedure to score new observations based on the fitted model saved by the STORE statement above. For example, to compute the probability of subscribing for one new

observation with sex = 'Female' and age = 35 in the data set test, you can use the following statements:

```
data test;
   input sex $ age;
   datalines;
Female     35
;

proc plm restore=LogitModel;
   score data=test out=testout predicted / ilink;
run;

proc print data=testout;
run;
```

The ILINK option in the SCORE statement applies the inverse of the logit link to provide an estimate on the mean (probability) scale. Output 100.3.2 shows the predicted probability for the new observation.

Output 100.3.2 Predicted Probability for One New Observation

Obs	sex	age	Predicted
1	Female	35	0.082205

Example 100.4: An Epidemiology Study

(View the complete code for this example (prbtex4.sas) in the example repository.)

The data in this example, which are from an epidemiology study, consist of five variables: the number, r, of individuals surviving after an epidemic, out of n treated, for combinations of medicine dosage (dose), treatment (treat = A, B), and sex (sex = 0(Female), 1(Male)).

To see whether the two treatments have different effects on male and female individual survival rates, the interaction term between the two variables treat and sex is included in the model.

The following invocation of PROC PROBIT fits the binary probit model to the grouped data:

```
data epidemic;
   input treat$ dose n r sex @@;
   label dose = Dose;
   datalines;
A  2.17 142 142  0   A   .57 132  47  1  A  1.68 128 105  1   A  1.08 126 100  0
A  1.79 125 118  0   B  1.66 117 115  1  B  1.49 127 114  0   B  1.17  51  44  1
B  2.00 127 126  0   B   .80 129 100  1
;

data xval;
   input treat $ dose sex;
   datalines;
B  2.  1
;
```

```
proc probit optc lackfit covout data=epidemic
            outest = out1 xdata = xval
            Plots=(predpplot ippplot lpredplot);
   class treat sex;
   model r/n = dose treat sex sex*treat/corrb covb inversecl;
   output out = out2 p =p;
run;
```

The results of this analysis are shown in the outputs that follow.

Output 100.4.1 displays the table of level information for *all* classification variables in the CLASS statement.

Output 100.4.1 Class Level Information

The Probit Procedure

Class Level Information		
Name	**Levels**	**Values**
treat	2	A B
sex	2	0 1

Output 100.4.2 displays the table of parameter information for the effects in the MODEL statement.

Output 100.4.2 Parameter Information

Parameter Information			
Parameter	**Effect**	**treat**	**sex**
Intercept	Intercept		
dose	dose		
treatA	treat	A	
treatB	treat	B	
sex0	sex		0
sex1	sex		1
treatAsex0	treat*sex	A	0
treatAsex1	treat*sex	A	1
treatBsex0	treat*sex	B	0
treatBsex1	treat*sex	B	1

Output 100.4.3 displays background information about the model fit. Included are the name of the input data set, the response variables used, the numbers of observations, events, and trials, the type of distribution, and the final value of the log-likelihood function.

Output 100.4.3 Model Information

The Probit Procedure

Model Information	
Data Set	WORK.EPIDEMIC
Events Variable	r
Trials Variable	n
Number of Observations	10
Number of Events	1011
Number of Trials	1204
Name of Distribution	Normal
Log Likelihood	-387.2467391

Output 100.4.4 displays the table of goodness-of-fit tests requested with the LACKFIT option in the PROC PROBIT statement. Two goodness-of-fit statistics, the Pearson's chi-square statistic and the likelihood ratio chi-square statistic, are computed. The grouping method for computing these statistics can be specified by the AGGREGATE= option. The details can be found in the AGGREGATE= option, and an example can be found in the second part of this example. By default, the PROBIT procedure uses the covariates in the MODEL statement to do grouping. Observations with the same values of the covariates in the MODEL statement are grouped into cells and the two statistics are computed according to these cells. The total number of cells and the number of levels for the response variable are reported next in the "Response-Covariate Profile."

In this example, neither the Pearson's chi-square nor the log-likelihood ratio chi-square tests are significant at the 0.1 level, which is the default test level used by the PROBIT procedure. That means that the model, which includes the interaction of treat and sex, is suitable for this epidemiology data set. (Further investigation shows that models without the interaction of treat and sex are not acceptable by either test.)

Output 100.4.4 Goodness-of-Fit Tests and Response-Covariate Profile

Goodness-of-Fit Tests				
Statistic	**Value**	**DF**	**Value/DF**	**Pr > ChiSq**
Pearson Chi-Square	4.9317	4	1.2329	0.2944
L.R. Chi-Square	5.7079	4	1.4270	0.2220

Response-Covariate Profile	
Response Levels	2
Number of Covariate Values	10

Output 100.4.5 displays the Type III test results for all effects specified in the MODEL statement, which include the degrees of freedom for the effect, the Wald Chi-Square test statistic, and the *p*-value.

Output 100.4.5 Type III Tests

Type III Analysis of Effects			
Effect	**DF**	**Wald Chi-Square**	**Pr > ChiSq**
dose	1	42.1691	<.0001
treat	1	16.1421	<.0001
sex	1	1.7710	0.1833
treat*sex	1	13.9343	0.0002

Output 100.4.6 displays the table of parameter estimates for the model. The PROBIT procedure displays information for all the parameters of an effect. Degenerate parameters are indicated by 0 degree of freedom. Confidence intervals are computed for all parameters with nonzero degrees of freedom, including the natural threshold C if the OPTC option is specified in the PROC PROBIT statement. The confidence level can be specified by the ALPHA= option in the MODEL statement. The default confidence level is 95%.

Output 100.4.6 Analysis of Parameter Estimates

Analysis of Maximum Likelihood Parameter Estimates

Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	-0.8871	0.3632	-1.5991	-0.1752	5.96	0.0146
dose		1	1.6774	0.2583	1.1711	2.1837	42.17	<.0001
treat	A	1	-1.2537	0.2616	-1.7664	-0.7410	22.97	<.0001
treat	B	0	0.0000	.	.	.	.	.
sex	0	1	-0.4633	0.2289	-0.9119	-0.0147	4.10	0.0429
sex	1	0	0.0000	.	.	.	.	.
treat*sex	A 0	1	1.2899	0.3456	0.6126	1.9672	13.93	0.0002
treat*sex	A 1	0	0.0000	.	.	.	.	.
treat*sex	B 0	0	0.0000	.	.	.	.	.
treat*sex	B 1	0	0.0000	.	.	.	.	.
_C_		1	0.2735	0.0946	0.0881	0.4589		

From Table 100.4.6, you can see the following results:

- The variable dose has a significant positive effect on the survival rate.
- Individuals under treatment A have a lower survival rate.
- Male individuals have a higher survival rate.
- Female individuals under treatment A have a higher survival rate.

Output 100.4.7 and Output 100.4.8 display tables of estimated covariance matrix and estimated correlation matrix for estimated parameters with a nonzero degree of freedom, respectively. They are computed by the inverse of the Hessian matrix of the estimated parameters.

Output 100.4.7 Estimated Covariance Matrix

Estimated Covariance Matrix

	Intercept	dose	treatA	sex0	treatAsex0	_C_
Intercept	0.131944	-0.087353	0.053551	0.030285	-0.067056	-0.028073
dose	-0.087353	0.066723	-0.047506	-0.034081	0.058620	0.018196
treatA	0.053551	-0.047506	0.068425	0.036063	-0.075323	-0.017084
sex0	0.030285	-0.034081	0.036063	0.052383	-0.063599	-0.008088
treatAsex0	-0.067056	0.058620	-0.075323	-0.063599	0.119408	0.019134
_C_	-0.028073	0.018196	-0.017084	-0.008088	0.019134	0.008948

Output 100.4.8 Estimated Correlation Matrix

Estimated Correlation Matrix						
	Intercept	**dose**	**treatA**	**sex0**	**treatAsex0**	**_C_**
Intercept	1.000000	-0.930998	0.563595	0.364284	-0.534227	-0.817027
dose	-0.930998	1.000000	-0.703083	-0.576477	0.656744	0.744699
treatA	0.563595	-0.703083	1.000000	0.602359	-0.833299	-0.690420
sex0	0.364284	-0.576477	0.602359	1.000000	-0.804154	-0.373565
treatAsex0	-0.534227	0.656744	-0.833299	-0.804154	1.000000	0.585364
_C_	-0.817027	0.744699	-0.690420	-0.373565	0.585364	1.000000

Output 100.4.9 displays the computed values and fiducial limits for the first single continuous variable dose in the MODEL statement, given the probability levels, without the effect of the natural threshold, and when the option INSERSECL in the MODEL statement is specified. If there is no single continuous variable in the MODEL specification but the INVERSECL option is specified, an error is reported.

Output 100.4.9 Probit Analysis on Dose

The Probit Procedure

Probit Analysis on dose			
		95%	
Probability	dose	Fiducial Limits	
0.01	-0.85801	-1.81301	-0.33743
0.02	-0.69549	-1.58167	-0.21116
0.03	-0.59238	-1.43501	-0.13093
0.04	-0.51482	-1.32476	-0.07050
0.05	-0.45172	-1.23513	-0.02130
0.06	-0.39802	-1.15888	0.02063
0.07	-0.35093	-1.09206	0.05742
0.08	-0.30877	-1.03226	0.09039
0.09	-0.27043	-0.97790	0.12040
0.10	-0.23513	-0.92788	0.14805
0.15	-0.08900	-0.72107	0.26278
0.20	0.02714	-0.55706	0.35434
0.25	0.12678	-0.41669	0.43322
0.30	0.21625	-0.29095	0.50437
0.35	0.29917	-0.17477	0.57064
0.40	0.37785	-0.06487	0.63387
0.45	0.45397	0.04104	0.69546
0.50	0.52888	0.14481	0.75654
0.55	0.60380	0.24800	0.81819
0.60	0.67992	0.35213	0.88157
0.65	0.75860	0.45879	0.94803
0.70	0.84151	0.56985	1.01942
0.75	0.93099	0.68770	1.09847
0.80	1.03063	0.81571	1.18970
0.85	1.14677	0.95926	1.30171
0.90	1.29290	1.12867	1.45386
0.91	1.32819	1.16747	1.49273
0.92	1.36654	1.20867	1.53590
0.93	1.40870	1.25284	1.58450
0.94	1.45579	1.30084	1.64012
0.95	1.50949	1.35397	1.70515
0.96	1.57258	1.41443	1.78353
0.97	1.65015	1.48626	1.88238
0.98	1.75326	1.57833	2.01720
0.99	1.91577	1.71776	2.23537

If the XDATA= option is used to input a data set for the independent variables in the MODEL statement, the PROBIT procedure uses these values for the independent variables other than the single continuous variable. Missing values are not permitted in the XDATA= data set for the independent variables, although the value for the single continuous variable is not used in the computing of the fiducial limits. A suitable valid value should be given. In the data set xval created by the SAS statements in Example 100.4, dose = 2. Only one observation from the XDATA= data set is used to produce a probit analysis table for a combination of classification variable levels. If more than one observation is present in the XDATA= data set, only the last

observation is used.

See the section "XDATA= *SAS-data-set*" on page 8294 for the default values for those effects other than the single continuous variable, for which the fiducial limits are computed.

In this example, there are two classification variables, treat and sex. Fiducial limits for the dose variable are computed for the highest level of the classification variables, treat = B and sex = 1, which is the default specification. Since these are the default values, you would get the same values and fiducial limits if you did not specify the XDATA= option in this example. The confidence level for the fiducial limits can be specified by the ALPHA= option in the MODEL statement. The default level is 95%.

If a LOG10 or LOG option is used in the PROC PROBIT statement, the values and the fiducial limits are computed for both the single continuous variable and its logarithm.

Output 100.4.10 displays the OUTEST= data set. All parameters for an effect are included. The name of a parameter is generated by combining the variable names and levels in the effect. The maximum length of a parameter name is 32.

Output 100.4.10 Outest Data Set for Epidemiology Study

Obs	_MODEL_	_NAME_	_TYPE_	_DIST_	_STATUS_	_LNLIKE_	r	Intercept	dose
1		r	PARMS	Normal	0 Converged	-387.247	-1.00000	-0.88714	1.67739
2		Intercept	COV	Normal	0 Converged	-387.247	-0.88714	0.13194	-0.08735
3		dose	COV	Normal	0 Converged	-387.247	1.67739	-0.08735	0.06672
4		treatA	COV	Normal	0 Converged	-387.247	-1.25367	0.05355	-0.04751
5		treatB	COV	Normal	0 Converged	-387.247	0.00000	0.00000	0.00000
6		sex0	COV	Normal	0 Converged	-387.247	-0.46329	0.03029	-0.03408
7		sex1	COV	Normal	0 Converged	-387.247	0.00000	0.00000	0.00000
8		treatAsex0	COV	Normal	0 Converged	-387.247	1.28991	-0.06706	0.05862
9		treatAsex1	COV	Normal	0 Converged	-387.247	0.00000	0.00000	0.00000
10		treatBsex0	COV	Normal	0 Converged	-387.247	0.00000	0.00000	0.00000
11		treatBsex1	COV	Normal	0 Converged	-387.247	0.00000	0.00000	0.00000
12		_C_	COV	Normal	0 Converged	-387.247	0.27347	-0.02807	0.01820

Obs	treatA	treatB	sex0	sex1	treatAsex0	treatAsex1	treatBsex0	treatBsex1	_C_
1	-1.25367	0	-0.46329	0	1.28991	0	0	0	0.27347
2	0.05355	0	0.03029	0	-0.06706	0	0	0	-0.02807
3	-0.04751	0	-0.03408	0	0.05862	0	0	0	0.01820
4	0.06843	0	0.03606	0	-0.07532	0	0	0	-0.01708
5	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
6	0.03606	0	0.05238	0	-0.06360	0	0	0	-0.00809
7	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
8	-0.07532	0	-0.06360	0	0.11941	0	0	0	0.01913
9	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
10	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
11	0.00000	0	0.00000	0	0.00000	0	0	0	0.00000
12	-0.01708	0	-0.00809	0	0.01913	0	0	0	0.00895

The plots in the following three outputs, Output 100.4.11, Output 100.4.12, and Output 100.4.13, are generated by the PLOTS= option. The first plot, specified with the PLOTS=PREDPPLOT option, is the plot of the predicted probability against the first single continuous variable dose in the MODEL statement. You

can specify values of other independent variables in the MODEL statement by using an XDATA= data set or by using the default values.

The second plot, specified with the PLOTS=IPPPLOT option, is the inverse of the predicted probability plot with the fiducial limits. It should be pointed out that the fiducial limits are *not* just the inverse of the confidence limits in the predicted probability plot; see the section "Inverse Confidence Limits" on page 8293 for the computation of these limits. The third plot, specified with the PLOTS=LPREDPLOT option, is the plot of the linear predictor $\mathbf{x}'_i$ against the first single continuous variable with the Wald confidence intervals.

Output 100.4.11 Predicted Probability Plot

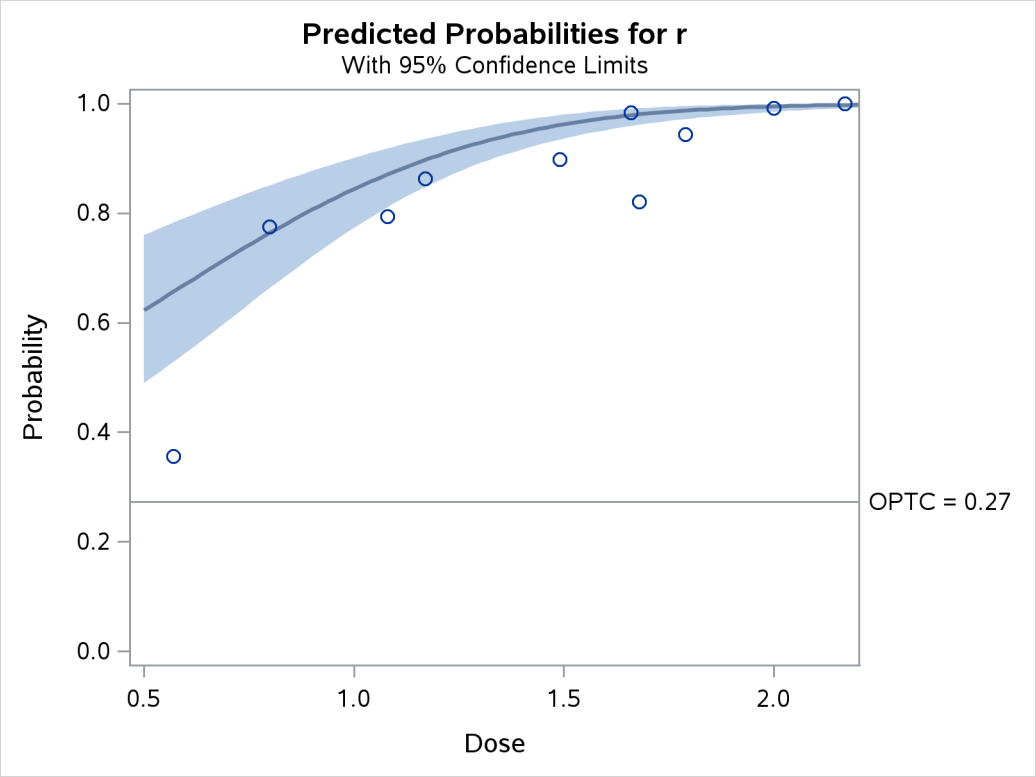

Output 100.4.12 Inverse Predicted Probability Plot

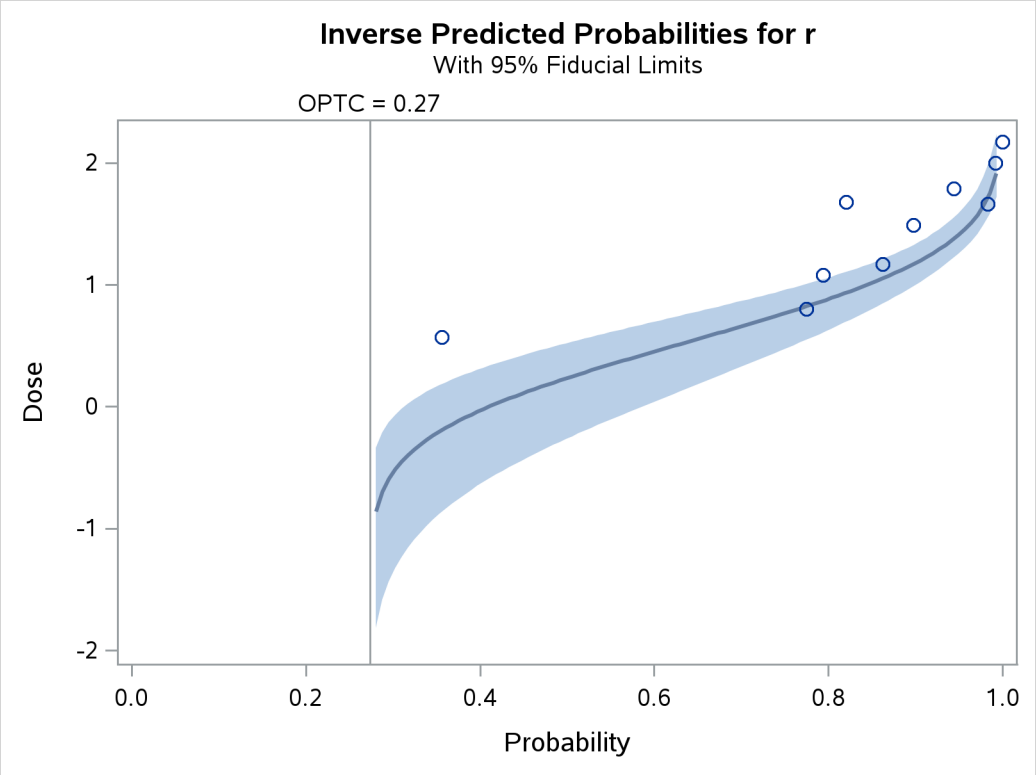

Output 100.4.13 Linear Predictor Plot

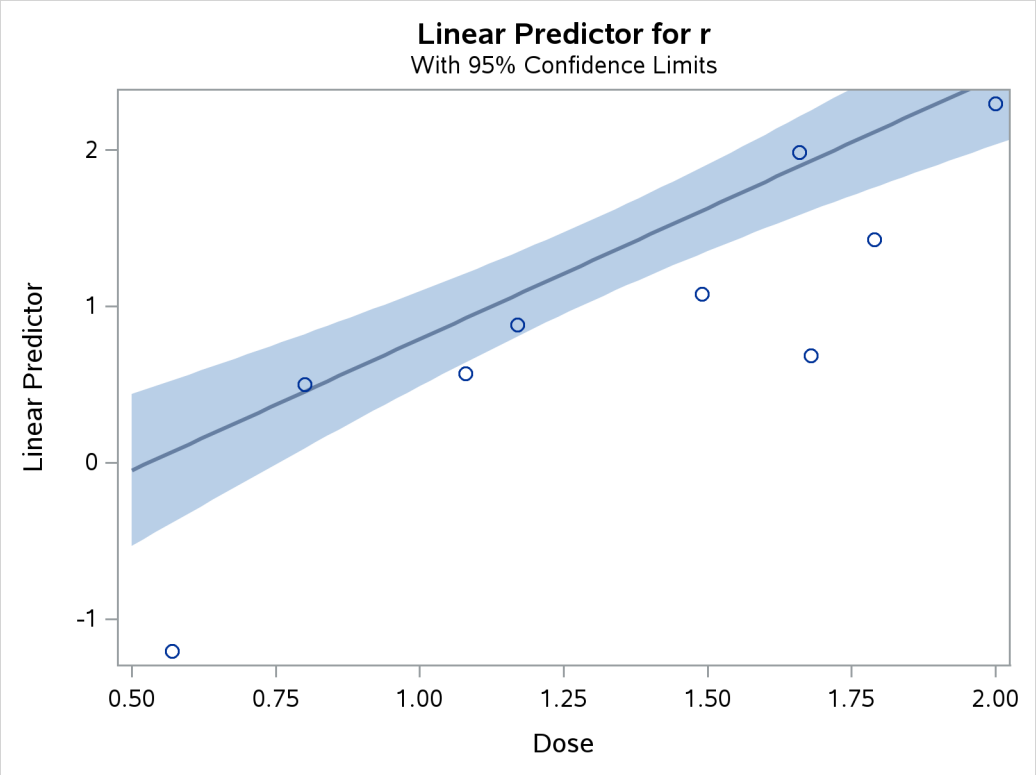

Example 100.5: Model Postfitting Analysis

(View the complete code for this example (prbtex5.sas) in the example repository.)

Recall the previous example of an epidemic study, in which the treat*sex interaction is statistically significant. Suppose you want to know whether such an effect is the same at different levels of the two categorical variables.

The following SAS statements fit a probit model and use the SLICE statement to request analysis of the two-way interaction term treat*sex:

```
proc probit data=epidemic;
   class treat sex;
   model r/n = dose treat sex treat*sex;
   slice treat*sex / diff;
   effectplot;
run;
```

Output 100.5.1 displays the test results for the interaction effect. As you can see, the difference between the two treatments is not significant among females.

Output 100.5.1 Tests Conditional on treat*sex

The Probit Procedure

Chi-Square Test for treat*sex Least Squares Means Slice			
Slice	Num DF	Chi-Square	Pr > ChiSq
treat A	1	18.52	<.0001

Chi-Square Test for treat*sex Least Squares Means Slice			
Slice	Num DF	Chi-Square	Pr > ChiSq
treat B	1	2.65	0.1035

Chi-Square Test for treat*sex Least Squares Means Slice			
Slice	Num DF	Chi-Square	Pr > ChiSq
sex 0	1	0.00	0.9579

Chi-Square Test for treat*sex Least Squares Means Slice			
Slice	Num DF	Chi-Square	Pr > ChiSq
sex 1	1	47.43	<.0001

The DIFF option computes effect differences between groups within the same slice. Results are displayed in Output 100.5.2.

Output 100.5.2 Effect Differences Conditional on treat*sex

Simple Differences of treat*sex Least Squares Means						
Slice	sex	_sex	Estimate	Standard Error	z Value	Pr > |z|
treat A	0	1	0.5957	0.1384	4.30	<.0001

Simple Differences of treat*sex Least Squares Means						
Slice	sex	_sex	Estimate	Standard Error	z Value	Pr > |z|
treat B	0	1	-0.2956	0.1816	-1.63	0.1035

Simple Differences of treat*sex Least Squares Means						
Slice	treat	_treat	Estimate	Standard Error	z Value	Pr > |z|
sex 0	A	B	-0.00899	0.1702	-0.05	0.9579

Simple Differences of treat*sex Least Squares Means						
Slice	treat	_treat	Estimate	Standard Error	z Value	Pr > |z|
sex 1	A	B	-0.9003	0.1307	-6.89	<.0001

The EFFECTPLOT statement produces a predicted probability plot for dose by the four groups that are formed by the treat*sex interaction. The plot is displayed in Output 100.5.3. The two overlapping curves represent the two treatment groups for females, suggesting no treatment effect. It appears that males tend to respond to the two treatments differently: those on treatment B have a better survival rate, and those on treatment A have a worse chance of survival.

Output 100.5.3 Predicted Probability versus Dose Level by treat*sex

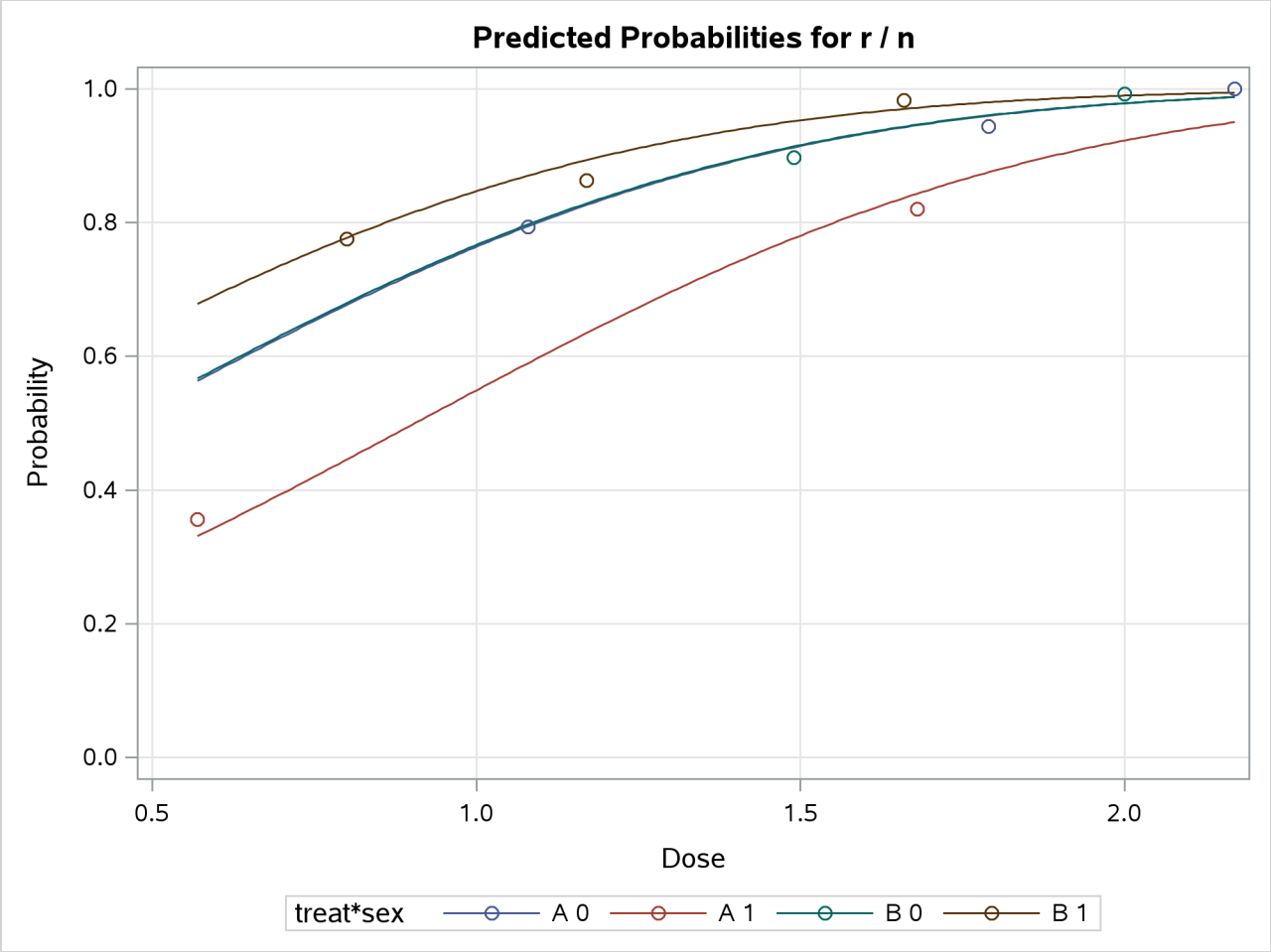

References

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Collett, D. (2003). *Modelling Binary Data*. 2nd ed. London: Chapman & Hall.

Cox, D. R., and Oakes, D. (1984). *Analysis of Survival Data*. London: Chapman & Hall.

Finney, D. J. (1971). *Probit Analysis*. 3rd ed. Cambridge: Cambridge University Press.

Hubert, J. J., Bohidar, N. R., and Peace, K. E. (1988). "Assessment of Pharmacological Activity." In *Biopharmaceutical Statistics for Drug Development*, edited by K. E. Peace, 83–145. New York: Marcel Dekker.

Chapter 101
The PSMATCH Procedure

Contents

Overview: PSMATCH Procedure . . . **8366**
- Process of Propensity Score Analysis . . . 8367
- Features of the PSMATCH Procedure . . . 8370

Getting Started: PSMATCH Procedure . . . **8372**

Syntax: PSMATCH Procedure . . . **8380**
- PROC PSMATCH Statement . . . 8380
- ASSESS Statement . . . 8383
- BY Statement . . . 8388
- CLASS Statement . . . 8388
- EWEIGHT Statement . . . 8389
- FREQ Statement . . . 8391
- ID Statement . . . 8391
- MATCH Statement . . . 8392
- OUTPUT Statement . . . 8399
- PSDATA Statement . . . 8400
- PSMODEL Statement . . . 8401
- PSWEIGHT Statement . . . 8402
- STRATA Statement . . . 8403

Details: PSMATCH Procedure . . . **8404**
- Observational Studies Contrasted with Randomized Trials . . . 8404
- Propensity Score Analysis . . . 8405
- Propensity Score Weighting . . . 8407
- Propensity Score Stratification . . . 8410
- Weighting after Stratification . . . 8411
- Matching Process . . . 8412
- Matching Metrics . . . 8413
- Matching Methods . . . 8414
- Weighting after Matching . . . 8416
- Variable Balance Assessment . . . 8418
- Sensitivity Analysis . . . 8421
- Entropy Balancing . . . 8423
- Table Output . . . 8426
- ODS Table Names . . . 8428
- Graphics Output . . . 8429
- ODS Graphics . . . 8431

Examples: PSMATCH Procedure . . . **8432**

Example 101.1: Propensity Score Weighting 8433
Example 101.2: Propensity Score Stratification 8442
Example 101.3: Optimal Variable Ratio Matching 8455
Example 101.4: Greedy Nearest Neighbor Matching 8461
Example 101.5: Outcome Analysis after Matching 8471
Example 101.6: Matching with Replacement 8474
Example 101.7: Mahalanobis Distance Matching 8479
Example 101.8: Matching with Precomputed Propensity Scores 8483
Example 101.9: Sensitivity Analysis after One-to-One Matching 8488
Example 101.10: Entropy Balance Weighting 8493
References **8494**

Overview: PSMATCH Procedure

In a randomized study, such as a randomized controlled trial, the subjects are randomly assigned to a treated (exposure) group or a control (non-exposure) group. Random assignment ensures that the distribution of the covariates is the same in both groups, and the treatment effect can be estimated by directly comparing the outcomes for the subjects in the two groups.

In contrast, the subjects in an observational study, such as a retrospective cohort study or a nonrandomized clinical trial, are not randomly assigned to the treated and control groups. Confounding can occur if some covariates are related to both the treatment assignment and the outcome. Consequently, there can be systematic differences between the treated subjects and the control subjects. In the presence of confounding, statistical approaches are required that remove the effects of confounding when estimating the effect of treatment.

One such approach is regression adjustment, which estimates the treatment effect after adjusting for differences in the baseline covariates. However, this approach has practical limitations, as discussed by Austin (2011a). Propensity score analysis is an alternative approach that circumvents many of these limitations.

The propensity score was defined by Rosenbaum and Rubin (1983, p. 47) as the probability of assignment to treatment conditional on a set of observed baseline covariates. Propensity score analysis minimizes the effects of confounding and offers some of the advantages of a randomized study. The basis for propensity score methods is the causal effect model introduced by Rubin (1974).

The PSMATCH procedure provides a variety of tools for propensity score analysis. The procedure either computes propensity scores or reads previously computed propensity scores, and it provides the following methods for using the scores to allow for valid estimation of the treatment effect in a subsequent outcome analysis:

- Inverse probability of treatment weighting and ATT weighting (weighting by odds): The procedure computes weights from the propensity scores. These weights can then be incorporated into a subsequent analysis that estimates the effect of treatment.
- Stratification: The procedure creates strata of observations that have similar propensity scores. In a subsequent analysis, the treatment effect can be estimated within each stratum, and the estimates can be combined across strata.

- Matching: The procedure matches each treated unit with one or more control units that have a similar value of the propensity score. In a subsequent analysis, the treatment effect can be estimated by comparing outcomes between treated and control subjects in the matched sample. If the outcome values for a study are not available prior to matching, only the matched units are needed for follow-up. Thus, the cost of the trial is reduced (Stuart 2010, p. 2).

In addition to performing propensity score analyses, the PSMATCH procedure also performs entropy balancing. Entropy balancing is a method of reweighting observations to ensure that the weighted moments of specified covariates are equal in the treated and control groups. As in the inverse probability of treatment weighting, you can incorporate the entropy balance weights in a subsequent analysis that estimates the effect of treatment. For more information about entropy balancing, see the section "Entropy Balancing" on page 8423.

The PSMATCH procedure also provides methods for assessing the balance of baseline covariates and other variables in the treated and control groups after matching, weighting, or stratification.

The PSMATCH procedure itself does not carry out the outcome analysis, nor does it make use of the outcome variable. After adequate variable balance has been achieved (as described in the section "Process of Propensity Score Analysis" on page 8367) and assuming that no other confounding variables are associated with both the treatment assignment and the outcome, the output data set that is created by the procedure serves as input for an appropriate statistical procedure for the outcome analysis.

Process of Propensity Score Analysis

A propensity score analysis usually involves the following steps (Guo and Fraser 2015, p. 131):

1. You specify a set of confounding variables that might be related to both the treatment assignment and the outcome.
2. You use this set of variables to fit a logistic regression model and compute propensity scores. The response is the probability of assignment to the treatment group.
3. If you are using weighting, you compute observation weights for estimating the treatment effect in a weighted outcome analysis.
4. If you are using stratification or matching, you specify the support region for observations. Observations outside this region are not included in the stratification or matching.
5. If you are using stratification, you specify the number of strata and create strata of observations that have similar propensity scores.
6. If you are using matching, you specify the distance metric for similarity of observations and the method for creating matched sets of observations. You can also compute weights for matched observations.
7. You assess the balance of variables by comparing the distributions between the treated and control groups.
8. To improve the balance, you can repeat the process with a different set of variables for the logistic regression model, a different region of support for stratification and matching, a different distance metric, or a different matching method.

9. When you are satisfied with the variable balance, you save the output data set for subsequent outcome analysis.

Propensity score analysis assumes that all the variables that affect both the outcome and the treatment assignment have been measured, but this assumption cannot be tested. In some cases, you can perform sensitivity analysis to examine this assumption; see the section "Sensitivity Analysis" on page 8421.

Note that the outcome variable is intentionally not used in these steps, and the selection of variables for the model should be made independently of the observed outcomes (Rubin 2001; Stuart 2010, p. 5). Furthermore, any variables that might have been affected by the treatment should not be included in the process (Rosenbaum and Rubin 1984; Stuart 2010, p. 5).

The flowchart in Figure 101.1 summarizes the steps.

After balance is achieved, you can add the response variable to the output data set that PROC PSMATCH created and perform an outcome analysis that mimics the analysis you would perform with data from a randomized study. For example, if you used matching, a simple univariate test or analysis might be sufficient to estimate the treatment effect.

Figure 101.1 Propensity Score Analysis

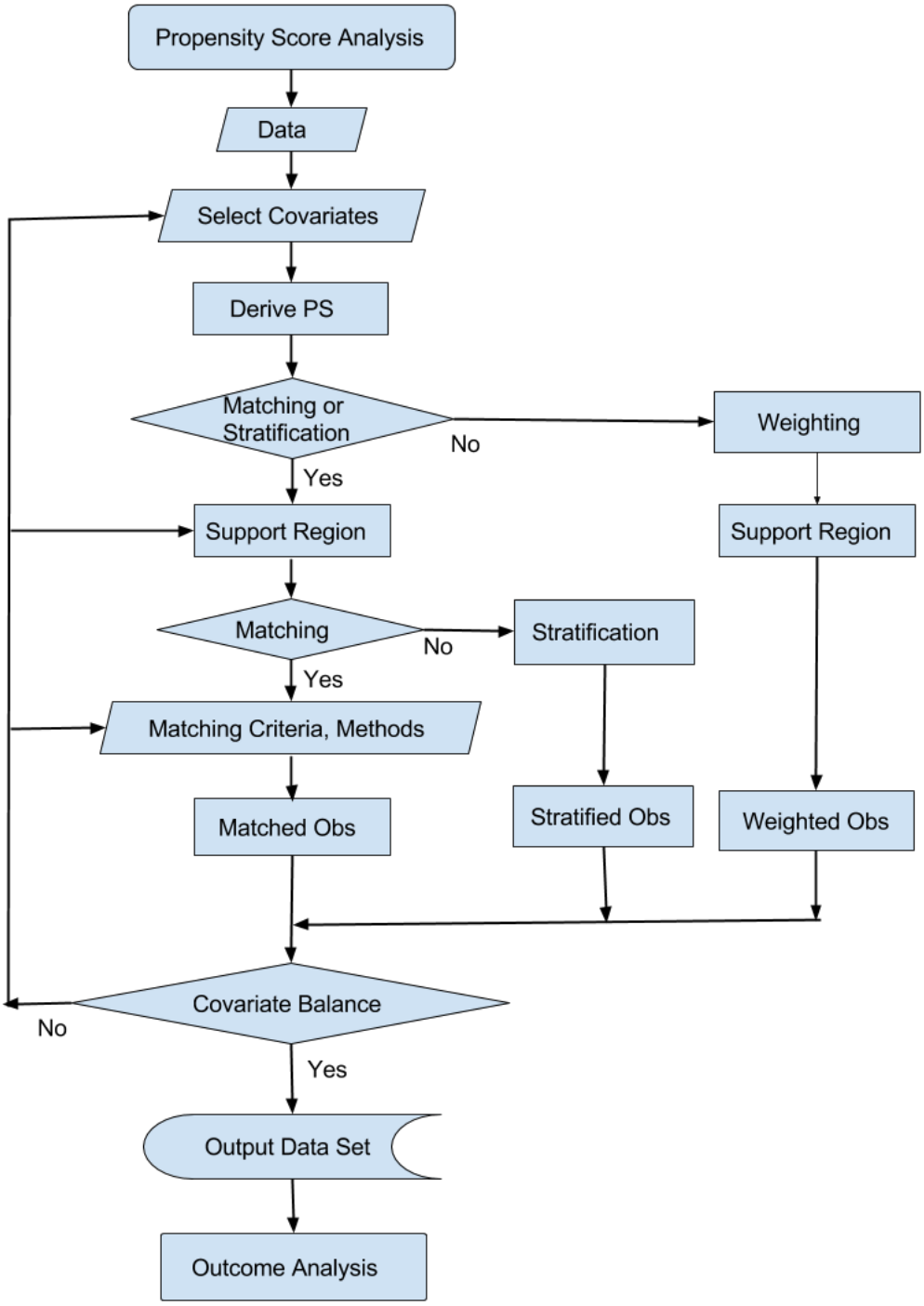

Features of the PSMATCH Procedure

You can use the PSMATCH procedure to create propensity scores (PS) for observations from treated and control groups by fitting a binary logistic regression model. Alternatively, you can input propensity scores that have already been created by using a different model or even a different approach such as a tree-based method. For example, you can input propensity scores that have been computed by the LOGISTIC procedure using a binary probit model or by the HPSPLIT procedure using a classification tree.

By default, the PSMATCH procedure uses the propensity scores to computes weights for the observations. Various types of weights are available, depending on whether the outcome analysis will use the weights to estimate the average treatment effect at the population level (ATE) or the average treatment effect for subjects who receive treatment (ATT). For more information about propensity score weighting, see the section "Propensity Score Weighting" on page 8407.

The PSMATCH procedure optionally creates strata of observations that have similar propensity scores. For more information, see the section "Propensity Score Stratification" on page 8410.

The PSMATCH procedure optionally matches observations in the treated and control groups. The procedure provides three strategies for propensity score matching.

- Greedy nearest neighbor matching selects the control unit nearest to each treated unit. Greedy nearest neighbor matching is done sequentially for treated units and without replacement.
- Optimal matching selects all control units that match each treated unit by minimizing the total absolute difference in propensity score across all matches. Optimal matching selects all matches simultaneously and without replacement. Three methods for optimal matching are available: fixed ratio matching, variable ratio matching, and full matching.
- Matching with replacement selects the control unit that best matches each treated unit. Each control unit can be matched to more than one treated unit, but it can only be matched to the same treated unit once.

For all three matching methods, you can specify a caliper width, which imposes a restriction on the quality of the matches. The difference in propensity score between the treated unit and its matching control unit must be less than or equal to the caliper width. For more information about these methods, see the section "Matching Methods" on page 8414.

Matching can be based on the difference in the logit of the propensity score (LPS), as well as the difference in the propensity score (PS). Furthermore, matching can be based on Mahalanobis distance that is computed from a set of continuous covariates (possibly including LPS and LS).

In addition to using the propensity score methods, you can use PROC PSMATCH to perform entropy balancing to reweight data. Entropy balancing directly targets balancing the moments of specified covariates and does not use the propensity score. For more information about entropy balancing, see the section "Entropy Balancing" on page 8423.

The PSMATCH procedure provides various ways to assess how well the distributions of variables are balanced between the treated and control groups. These variables include the propensity score, the logit of the propensity score, variables used in the logistic regression model, and other variables in the data set. The assessments include the following:

- differences in the distributions of the variables between the treated and control groups after weighting, stratification, and matching
- standardized mean differences in the variables between the treated and control groups after weighting, stratification, and matching
- percentage reductions of absolute differences after weighting, stratification, and matching

When you use stratification, the differences are also computed within each stratum. For more information about these statistics, see the section "Variable Balance Assessment" on page 8418.

The PSMATCH procedure also provides various plots for assessing balance. These plots include the following:

- bar charts for classification variables
- box plots for continuous variables
- CDF plots for continuous variables
- cloud plots for continuous variables, which are scatter plots in which the points are jittered to prevent overplotting
- cloud plots for inverse probability of treatment weights, ATT weights, and entropy balance weights
- a standardized mean differences plot that summarizes differences between the treated and control groups

When you use stratification, these plots are also produced for each stratum.

The PSMATCH procedures saves propensity scores and weights in an output data set that contains a sample that has been adjusted either by weighting, stratification, or matching. If the sample is stratified, you can save the strata identification in the output data set. If the sample is matched, you can save the matching identification in the output data set. If entropy balancing is used, propensity score values are not computed and are not included in the output data set. Instead, the procedure saves the entropy balance weights in an output data set.

Provided that the distributions of the variables in the adjusted sample are well balanced between the treated and control groups, the output data set serves as input for a subsequent outcome analysis that incorporates weights or strata or that is based on matched observations. Although the PSMATCH procedure itself does not provide this analysis, you can use many other SAS/STAT procedures for this purpose.

Getting Started: PSMATCH Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/psmchgs.sas.)

This example illustrates the use of the PSMATCH procedure to match observations for individuals in a treatment group with observations for individuals in a control group that have similar propensity scores. The matched observations are saved in an output data set that, with the addition of the outcome variable, can be used to provide an unbiased estimate of the treatment effect.

A pharmaceutical company is conducting a nonrandomized clinical trial to demonstrate the efficacy of a new treatment (Drug_X) by comparing it to an existing treatment (Drug_A). Patients in the trial can choose the treatment that they prefer; otherwise, physicians assign each patient to a treatment. The possibility of treatment selection bias is a concern because it can lead to systematic differences in the distributions of the baseline variables in the two groups, resulting in a biased estimate of treatment effect.

The data set Drugs contains baseline variable measurements for individuals from both treated and control groups. PatientID is the patient identification number, Drug is the treatment group indicator, Gender provides the gender, Age provides the age, and BMI provides the body mass index (a measure of body fat based on height and weight). Typically, more variables are used in a propensity score analysis, but for simplicity only a few variables are used in this example.

Figure 101.2 lists the first 10 observations.

Figure 101.2 Input Drug Data Set

Obs	PatientID	Drug	Gender	Age	BMI
1	284	Drug_X	Male	29	22.02
2	201	Drug_A	Male	45	26.68
3	147	Drug_A	Male	42	21.84
4	307	Drug_X	Male	38	22.71
5	433	Drug_A	Male	31	22.76
6	435	Drug_A	Male	43	26.86
7	159	Drug_A	Female	45	25.47
8	368	Drug_A	Female	49	24.28
9	286	Drug_A	Male	31	23.31
10	163	Drug_X	Female	39	25.34

Note that the Drugs data set does not contain a response variable, because the response variable is not used in a propensity score analysis. Instead, the response variable is added to the output data set that contains the matched observations, and the combined data set is then used for outcome analysis.

The following statements invoke the PSMATCH procedure and request optimal matching to match observations for patients in the treatment group with observations for patients in the control group:

```
ods graphics on;
proc psmatch data=drugs region=cs;
   class Drug Gender;
   psmodel Drug(Treated='Drug_X')= Gender Age BMI;
   match method=optimal(k=1) exact=Gender distance=lps caliper=0.25
```

```
        weight=none;
    assess lps allcov / plots=(barchart boxplot);
    output out(obs=match)=Outgs lps=_Lps matchid=_MatchID;
run;
```

The CLASS statement specifies the classification variables. The PSMODEL statement specifies the logistic regression model that creates the propensity score for each observation, which is the probability that the patient receives Drug_X. The Drug variable is the binary treatment indicator variable and TREATED='Drug_X' identifies Drug_X as the treated group. The Gender, Age, and BMI variables are included in the model because they are believed to be related to the assignment.

The REGION= option specifies which observations are used in stratification and matching. In this example, matching is requested by the MATCH statement, and the REGION=CS option requests that only those observations whose propensity scores (or equivalently, logits of propensity scores) lie in the common support region be used for matching. The common support region is defined as the largest interval that contains propensity scores for subjects in both groups. By default, the region is extended by 0.25 times a pooled estimate of the common standard deviation of the logit of the propensity score. For more information, see the description of the EXTEND= option on page 8382.

The MATCH statement specifies the criteria for matching. The DISTANCE=LPS option (which is the default) requests that the logit of the propensity score be used to compute differences between pairs of observations. The METHOD=OPTIMAL(K=1) option (which is the default) requests optimal matching of one control unit to each unit in the treated group in order to minimize the total within-pair difference, The EXACT=GENDER option forces the treated unit and its matched control unit to have the same value of the Gender variable.

The CALIPER=0.25 option specifies the caliper requirement for matching. This means that for a match to be made, the difference in the logits of the propensity scores for pairs of individuals from the two groups must be less than or equal to 0.25 times the pooled estimate of the common standard deviation of the logits of the propensity scores.

The "Data Information" table in Figure 101.3 displays information about the input and output data sets, the numbers of observations in the treated and control groups, the lower and upper limits for the propensity score support region, and the numbers of observations in the treated and control groups that fall within the support region. Of the 373 observations in the control group, 351 fall within the support region.

Figure 101.3 Data Information

The PSMATCH Procedure

Data Information	
Data Set	WORK.DRUGS
Output Data Set	WORK.OUTGS
Treatment Variable	Drug
Treated Group	Drug_X
All Obs (Treated)	113
All Obs (Control)	373
Support Region	Extended Common Support
Lower PS Support	0.050244
Upper PS Support	0.683999
Support Region Obs (Treated)	113
Support Region Obs (Control)	351

The "Propensity Score Information" table in Figure 101.4 displays summary statistics for propensity scores by treatment group on the basis of all observations, support region observations, and matched observations. When you specify the METHOD=OPTIMAL(K=1) option, all matched observations have the same weight—that is, each matched unit has a weight of 1. Therefore, all the propensity score summary statistics would remain unchanged if you applied these weights to the matched observations. In the example, the WEIGHT=NONE option suppresses the display of summary statistics for weighted matched observations.

Figure 101.4 Propensity Score Information

Propensity Score Information											
	Treated (Drug = Drug_X)					Control (Drug = Drug_A)					Treated - Control
Observations	N	Mean	Standard Deviation	Minimum	Maximum	N	Mean	Standard Deviation	Minimum	Maximum	Mean Difference
All	113	0.3108	0.1325	0.0602	0.6411	373	0.2088	0.1320	0.0202	0.6858	0.1020
Region	113	0.3108	0.1325	0.0602	0.6411	351	0.2176	0.1267	0.0510	0.6824	0.0932
Matched	113	0.3108	0.1325	0.0602	0.6411	113	0.3082	0.1310	0.0619	0.6824	0.0025

The "Matching Information" table in Figure 101.5 displays the matching criteria, the number of matched sets, the numbers of matched observations in the treated and control groups, and the total absolute difference in the logit of the propensity score for all matches.

Figure 101.5 Matching Information

Matching Information	
Distance Metric	Logit of Propensity Score
Method	Optimal Fixed Ratio Matching
Control/Treated Ratio	1
Caliper (Logit PS)	0.191862
Matched Sets	113
Matched Obs (Treated)	113
Matched Obs (Control)	113
Total Absolute Difference	2.941871

The ASSESS statement produces a table and plots that summarize differences in specified variables between treated and control groups. As specified by the LPS and ALLCOV options, these variables are the logit of the propensity score (LPS) and all the covariates in the PSMODEL statement: Gender, Age, and BMI. For a binary classification variable (Gender), the difference is in the proportion of the first ordered level (Female).

The "Standardized Mean Differences" table, shown in Figure 101.6, displays standardized mean differences for all observations, observations in the support region, and matched observations. The WEIGHT=NONE option suppresses the display of differences for weighted matched observations. Note that when one control unit is matched to each treated unit, the weights are all 1 for matched treated and control units and the results are identical for weighted matched observations and matched observations.

Figure 101.6 Standardized Mean Differences

The PSMATCH Procedure

Standardized Mean Differences (Treated - Control)						
Variable	**Observations**	**Mean Difference**	**Standard Deviation**	**Standardized Difference**	**Percent Reduction**	**Variance Ratio**
Logit Prop Score	**All**	0.63997	0.767449	0.83389		0.6517
	Region	0.54546		0.71074	14.77	0.8314
	Matched	0.01056		0.01375	98.35	1.0155
Age	**All**	-4.09509	6.079104	-0.67363		0.7076
	Region	-3.49368		-0.57470	14.69	0.8000
	Matched	0.16814		0.02766	95.89	1.1262
BMI	**All**	0.73930	1.923178	0.38441		0.8854
	Region	0.63257		0.32892	14.44	0.9288
	Matched	0.12425		0.06461	83.19	1.1967
Gender	**All**	-0.02482	0.496925	-0.04994		0.9892
	Region	-0.01651		-0.03323	33.46	0.9922
	Matched	0.00000		0.00000	100.00	1.0000
Standard deviation of All observations used to compute standardized differences						

By default, the standard deviations of the variables, pooled across the treated and control groups, are computed based on all observations. The pooled standard deviations are then used to compute standardized mean differences based on all observations, observations in the support region, and matched observations. You can request a different standard deviation with the STDDEV= option. In Figure 101.6 the standardized mean differences are significantly reduced in the matched observations. The largest of these differences in absolute value is 0.0646, which is less than the upper limit of 0.25 recommended by Rubin (2001, p. 174) and Stuart (2010, p. 11). However, many authors use an upper limit of 0.10 (Normand et al. 2001; Mamdani et al. 2005; Austin 2009).

The treated-to-control variance ratios between the two groups are between 1 and 1.1967 for all variables in the matched observations, which is within the recommended range of 0.5 to 2 (Rubin 2001, p. 174).

Note that the standardized mean difference for Gender is 0 in the matched observations because EXACT=GENDER is specified in the MATCH statement.

By default, when ODS Graphics is enabled, the PSMATCH procedure displays a standardized mean differences plot for the variables that are specified in the ASSESS statement, as shown in Figure 101.7.

Figure 101.7 Standardized Mean Differences Plot

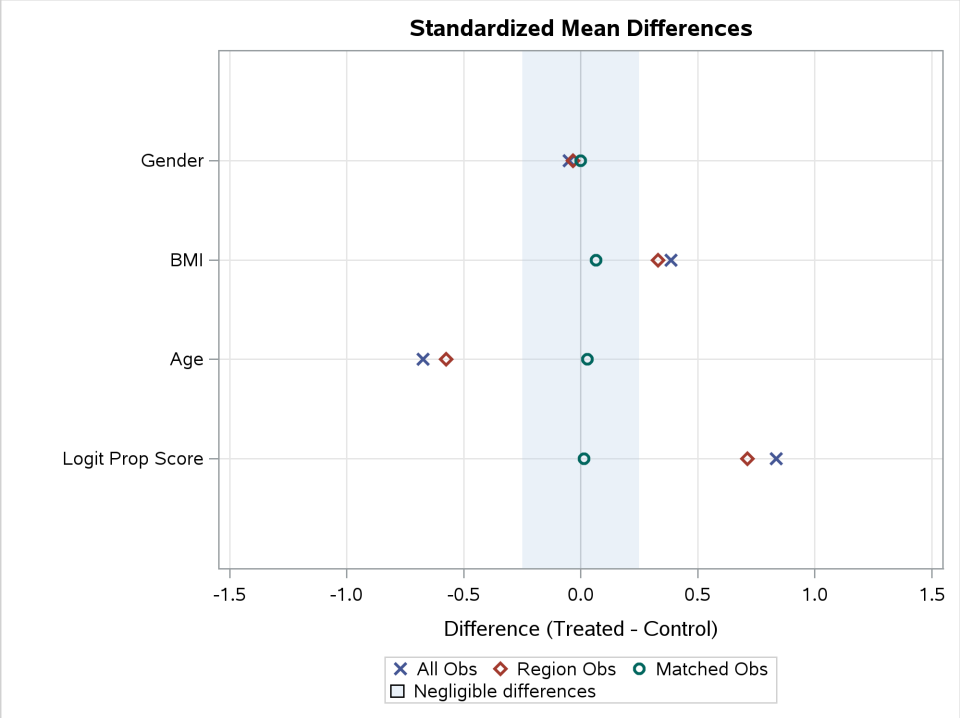

The "Standardized mean Differences Plot" displays the standardized mean differences in the "Standardized Mean Differences" table in Figure 101.6. All differences for the matched observations are within the recommended limits of –0.25 and 0.25, which are indicated by the shaded area. Again, note that many authors use limits of –0.10 and 0.10. (Normand et al. 2001; Mamdani et al. 2005; Austin 2009). You can use the PLOTS=STDDIFFPLOT(REF=) option to specify the limits for the shaded area.

The PLOTS=BARCHART option requests bar charts that compare the treated and control group distributions of binary classification variables that are specified in the ASSESS statement. The bar chart that is created for Gender is shown in Figure 101.8. The chart displays proportions by default, and it provides comparisons based on all observations, observations in the support region, and matched observations. The distributions of Gender are identical for matched observations because EXACT=GENDER is specified in the MATCH statement.

Figure 101.8 Gender Bar Chart

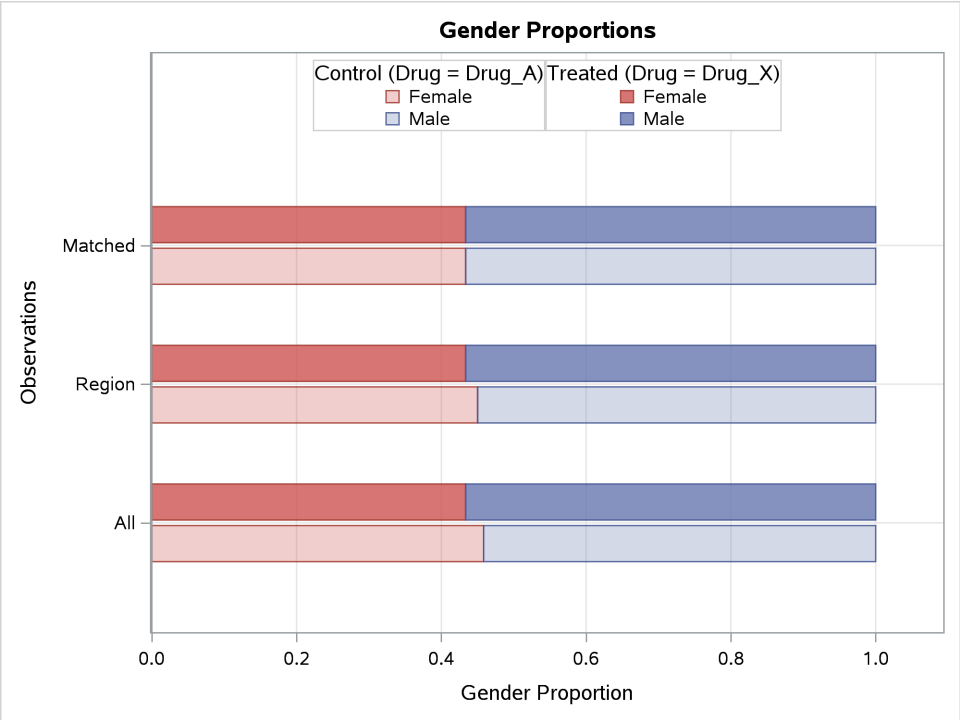

The PLOTS=BOXPLOT option requests box plots for the logit of the propensity score (LPS) and for the continuous variables that are specified in the ASSESS statement, as shown in Figure 101.9, Figure 101.10, and Figure 101.11. The box plots show good variable balance for the matched observations.

Figure 101.9 LPS Box Plot

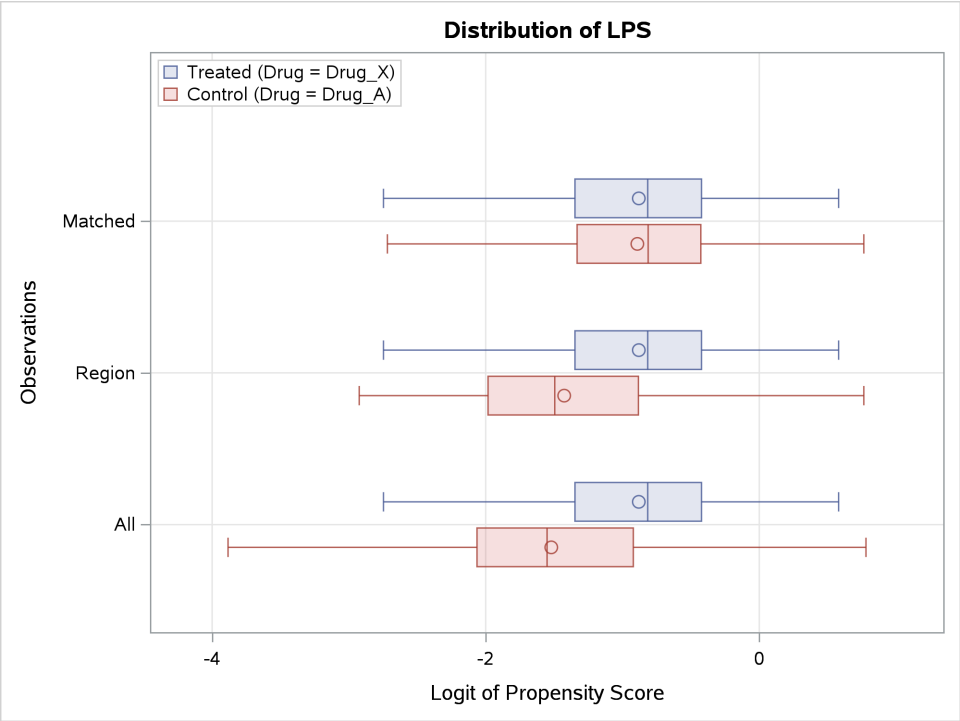

Figure 101.10 Age Box Plot

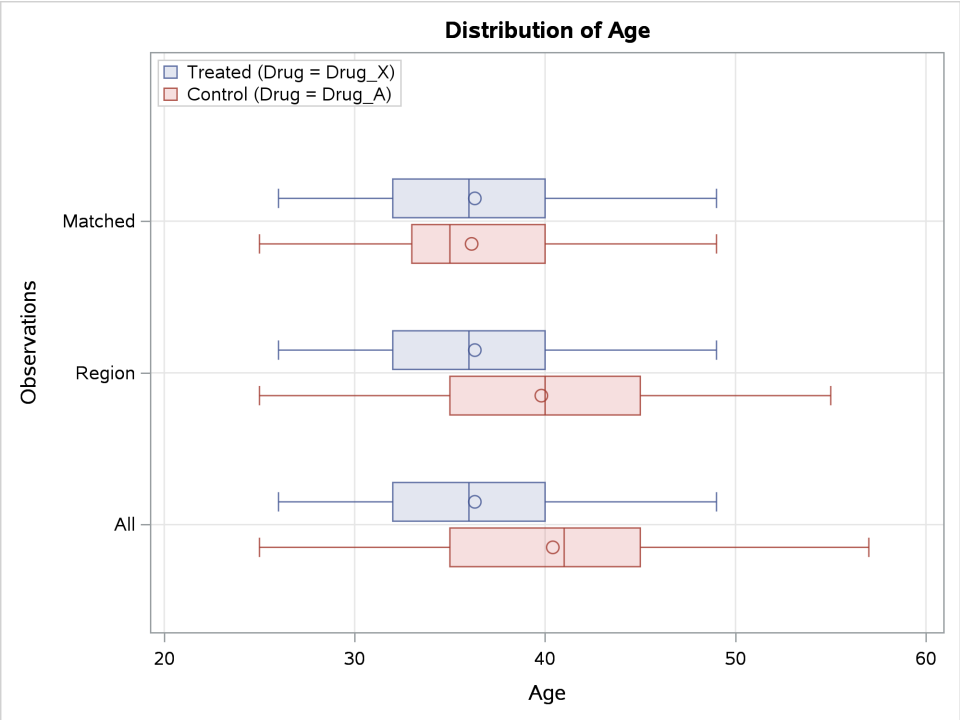

Figure 101.11 BMI Box Plot

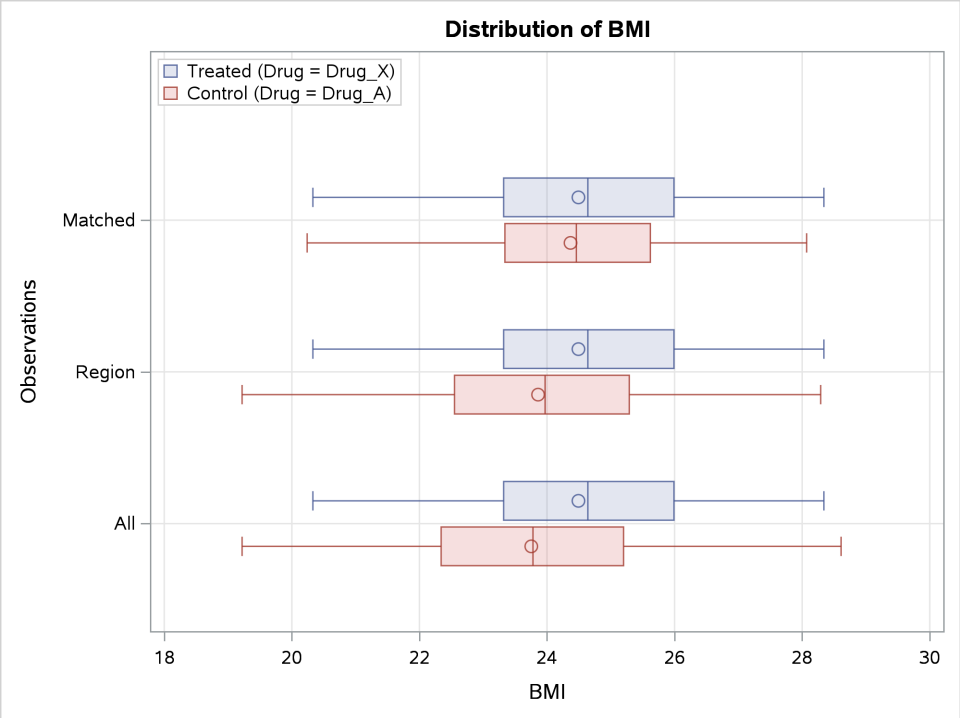

Because the matched observations in this example exhibit good balance, you can output them for subsequent

outcome analysis. In situations where you are not satisfied with the balance, you can do one or more of the following to improve the balance: you can select another set of variables for the propensity score model, you can modify the specification of the propensity score model (for example, by introducing nonlinear terms for the continuous variables or by adding interactions), you can modify the matching criteria, or you can choose another matching method.

The OUT(OBS=MATCH)= option in the OUTPUT statement creates an output data set named Outgs that contains the matched observations. By default, this data set includes the variable _PS_ (which provides the propensity score) and the variable _MATCHWGT_ (which provides matched observation weights). The weight for each treated unit is 1. The weight for each matched control unit is also 1 because one control unit is matched to each treated unit. The LPS=_LPS option adds a variable named _LPS that provides the logit of the propensity score, and the MATCHID=_MatchID option adds a variable named _MatchID that identifies the matched sets of observations.

The following statements list the observations in the first five matched sets, as shown in Figure 101.12.

```
proc sort data=outgs out=outgs1;
   by _MatchID;
run;

proc print data=outgs1(obs=10);
   var PatientID Drug Gender Age BMI _PS_ _LPS _MatchWgt_ _MatchID;
run;
```

Figure 101.12 Output Data Set with Matching Numbers

Obs	PatientID	Drug	Gender	Age	BMI	_PS_	_Lps	_MATCHWGT_	_MatchID
1	213	Drug_A	Female	49	23.24	0.06187	-2.71892	1	1
2	89	Drug_X	Female	44	20.75	0.06023	-2.74745	1	1
3	141	Drug_A	Female	43	20.55	0.06401	-2.68256	1	2
4	323	Drug_X	Female	46	22.22	0.06763	-2.62375	1	2
5	420	Drug_A	Male	45	22.08	0.08801	-2.33814	1	3
6	217	Drug_X	Male	49	23.96	0.08772	-2.34185	1	3
7	234	Drug_X	Female	41	21.11	0.08904	-2.32538	1	4
8	290	Drug_A	Female	40	20.57	0.08778	-2.34104	1	4
9	320	Drug_X	Female	46	24.17	0.10323	-2.16184	1	5
10	473	Drug_A	Female	45	23.76	0.10464	-2.14670	1	5

After the responses for the trial are observed and added to the matched data set Outgs, you can estimate the treatment effect by carrying out the same type of outcome analysis on Outgs that you would have used with the original data set Drugs (augmented with responses) as if it were a randomized trial (Ho et al. 2007, p. 223). This assumes that no other confounding variables are associated with both the response variable and the treatment group indicator Drug.

Syntax: PSMATCH Procedure

The following statements are available in the PSMATCH procedure:

```
PROC PSMATCH <options> ;
    ASSESS <ALLCOV> <LPS> <PS> <VAR=(var-list)> </ assess-options> ;
    BY variables ;
    CLASS variables ;
    EWEIGHT TREATVAR=treatvar <(trt-option)>
            BALANCEVARS=(variables </ moments-option>) <options> ;
    FREQ variable ;
    ID variable ;
    MATCH <options> ;
    OUTPUT OUT <(OBS=obs-value)>=SAS-data-set <keyword=name <keyword=name ... >> ;
    PSDATA TREATVAR=treatvar <(trt-option)> ps-option ;
    PSMODEL treatvar <(trt-option)> = <effects> </ WEIGHT=weight> ;
    PSWEIGHT <options> ;
    STRATA <options> ;
```

The PROC PSMATCH statement invokes the PSMATCH procedure. The CLASS statement and either a PSMODEL, PSDATA, or EWEIGHT statement are required. For propensity score analyses, either a PSMODEL or PSDATA statement is required. If you specify a PSMODEL statement, the CLASS statement must precede the PSMODEL statement.

The EWEIGHT, MATCH, PSWEIGHT, and STRATA statements perform entropy balancing, matching, propensity score–based weighting, and stratification, respectively. You can specify only one of these statements.

The following sections first describe the PROC PSMATCH statement and then describe the other statements in alphabetical order.

PROC PSMATCH Statement

The PROC PSMATCH statement invokes the PSMATCH procedure. Table 101.1 summarizes the options available in the PROC PSMATCH statement.

Table 101.1 Summary of PROC PSMATCH Options

Option	Description
DATA=	Specifies the input data set
REGION=	Specifies the support region of observations for stratification and matching

DATA=*SAS-data-set*

names the input SAS data set. If the propensity scores are to be derived from this data set, you must also include a PSMODEL statement to specify the binary logistic model. Otherwise, a PSDATA statement is required to identify the variable that contains either the propensity scores or the logits of the propensity scores. If you do not specify this option, the procedure uses the most recently created SAS data set.

REGION=*region* **<(***region-options***)>**

specifies an interval region of propensity scores (or equivalently, logits of propensity scores) that determines which observations are used in stratification and matching. Only those observations whose propensity scores lie in the region are used in stratification and matching. This option also determines which observations are included in the output data set if you specify the OUT(OBS=REGION) option in the OUTPUT statement (even when you omit the STRATA and MATCH statements). By default, REGION=TREATED if you specify a MATCH statement, and REGION=ALLOBS otherwise.

When you perform entropy balancing, all observations are used and the REGION= option is ignored.

You can specify the following *regions* along with their *region-options*:

REGION=ALLOBS <(*region-options***)>**

selects all available observations. You can specify the following *region-options* to select observations whose propensity scores lie in a specified range:

PSMIN=*pmin*

specifies the minimum propensity score in the support region, where $pmin \geq 0$. Observations whose propensity scores are less than *pmin* are excluded from the support region. By default, PSMIN=0, so that observations that have small propensity scores are not excluded.

PSMAX=*pmax*

specifies the maximum propensity score in the support region, where $pmax \leq 1$. Observations whose propensity scores are greater than *pmax* are excluded from the support region. By default, PSMAX=1, so that observations that have large propensity scores are not excluded.

You can also use the PSMIN= and PSMAX= options to exclude observations that have extreme propensity scores from the output data set.

REGION=CS <(*ext-option***)>**

selects observations whose propensity scores (or equivalently, logits of propensity scores) lie in the region of common support for the treated and control groups. This region is the largest interval that contains propensity scores (or logits of propensity scores) for subjects in both groups. The lower endpoint of the region is the larger of the minimum propensity scores (or logits of propensity scores) for the two groups. The upper endpoint is the smaller of the maximum propensity scores (or logits of propensity scores) for the two groups.

You can specify the following *ext-option*:

EXTEND <(*type-options*)> = *p* <(LOWER=p_l UPPER=p_u)>
extends the lower and upper ends of the common support region for the support region by *p*, where $p \geq 0$. By default, EXTEND=0.25.

You can use the following *type-options* to prescribe the extension requirement:

DISTANCE=LPS | PS
specifies the type of the distance that is used to extend the support region.

LPS extends the region by using the logit of the propensity score.

PS extends the region by using the propensity score.

By default, DISTANCE=LPS.

MULT=ONE | STDDEV
specifies the multiplier for the extension *p* to extend the support region.

ONE extends the region by *p*.

STDDEV extends the region by *p* times the pooled estimate of the standard deviation of either LPS (DISTANCE=LPS) or PS (DISTANCE=PS), where this estimate is computed as the square root of the average of the variances in the treated and control groups.

By default, MULT=STDDEV.

The DISTANCE= and MULT= *type-options* prescribe the extension requirement as follows:

- EXTEND(DISTANCE=PS MULT=ONE)=*p* extends the specified support region by *p* in propensity score. That is, if (R_l, R_u) denotes the propensity score interval region that is computed from the specified *region*, then the range of the extended support region is given by $(R_l - p,\ R_u + p)$.
- EXTEND(DISTANCE=PS MULT=STDDEV)=*p* extends the specified support region by $p \times \hat{\sigma}_{ps}$, the square root of the average variance of the propensity score in the treated and control groups. That is, if (R_l, R_u) denotes the propensity score interval region that is computed from the specified *region*, then the range of the extended support region is given by $(R_l - p\,\hat{\sigma}_{ps},\ R_u + p\,\hat{\sigma}_{ps})$.
- EXTEND(DISTANCE=LPS MULT=ONE)=*p* extends the specified support region by *p* in the logit of the propensity score.
- EXTEND(DISTANCE=LPS MULT=STDDEV)=*p* extends the specified support region by $p \times \hat{\sigma}_{lps}$, the square root of the average variance of the logit of the propensity score in the treated and control groups.

You can specify one of the following two options to use an extension other than *p*:

LOWER=p_l extends the lower end of the specified region by p_l, where $p_l \geq 0$.

UPPER=p_u extends the upper end of the specified region by p_u, where $p_u \geq 0$.

REGION=TREATED < (*ext-option*) >
selects observations whose propensity scores lie in the region of propensity scores for observations in the treated group.

You can specify the following *ext-option*:

EXTEND < (*type-options*) >= *p* < (**LOWER=**p_l **UPPER=**p_u) >
extends the lower and upper ends of the range of treated observations for the support region by p, where $p \geq 0$. By default, EXTEND=0.25.

You can use the *type-options* to prescribe the extension requirement, and these are identical to the *type-options* in the REGION=CS option. You can also specify the LOWER=p_l or UPPER=p_u suboption to use an extension other than p.

ASSESS Statement

ASSESS < **ALLCOV** > < **LPS** > < **PS** > < **VAR=***(var-list)* > < / *assess-options* > **;**

The ASSESS statement assesses variable differences between the treated and control groups for all observations and for observations in the support region that is specified in the REGION= option.

It also assesses variable differences for matched observations if you specify a MATCH statement, and it assesses variable differences for observations by stratum if you specify a STRATA statement. In addition, the ASSESS statement assesses variable differences for weighted observations provided that the WEIGHT=NONE suboption is not specified.

You can specify the variables for assessment by using the following options:

ALLCOV
requests an assessment of differences in the covariates that are specified in the PSMODEL statement. These variables must be binary classification variables or continuous variables in the input data set.

LPS
requests an assessment of differences in the logit of the propensity score. This option is ignored when you use entropy balancing.

PS
requests an assessment of differences in the propensity score. This option is ignored when you use entropy balancing.

VAR=(*var-list***)**
requests an assessment of differences in the specified list of variables. These variables are not limited to the variables that you specify in the PSMODEL statement. These variables can be any binary classification variables or continuous variables in the input data set.

If none of these options are specified, an assessment of differences in the propensity score is produced by default.

In addition, you can specify various *assess-options* after a slash (/). Table 101.2 summarizes these options.

Table 101.2 ASSESS Statement Options

Option	Description
PLOTS=	Requests plots for assessment of variable balance
STDBINVAR=	Specifies whether to standardize binary variables in the standardized mean differences table and plot
STDDEV=	Specifies the type of standard deviation to be used in the standardized mean difference computation
VARINFO	Displays variable information for the treated and control groups

PLOTS < (*global-option*) > < = *plot-request* >

PLOTS < (*global-option*) > = (*plot-request* < . . . *plot-request* >)

specifies options that control the plots.

You can specify the following *global-options*:

NODETAILS

displays plots for only two sets of observations: the set of all observations and a second set that depends on specified statements and options. This option does not apply to cloud plots.

If you specify a MATCH statement, the second set consists of matched observations (if neither WEIGHT=MATCHWGT nor WEIGHT=MATCHATEWGT is specified) or weighted matched observations (if WEIGHT=MATCHWGT or WEIGHT=MATCHATEWGT is specified). If you specify a STRATA statement, the second set consists of observations in the support region. If you specify neither a MATCH statement nor a STRATA statement, the second set consists of observations in the support region (if neither WEIGHT=ATEWGT nor WEIGHT=ATTWGT is specified) or the set of weighted observations in the support region (if WEIGHT=ATEWGT or WEIGHT=ATTWGT is specified).

ONLY

suppresses the default plots and displays only plots that are specifically requested.

ORIENT=HORIZONTAL | VERTICAL

controls the orientation of the plots.

HORIZONTAL places the lines and boxes horizontally for variable distribution plots, places the bar lengths horizontally for bar charts, places the variable values horizontally for cloud plots, places the standardized mean differences on the horizontal axis for the standardized mean differences plot, and places the graphs in a single column for CDF plots.

VERTICAL places the lines and boxes vertically for variable distribution plots, places the bar lengths vertically for bar charts, places the variable values vertically for cloud plots, places the standardized mean differences on the vertically axis for the standardized mean differences plot, and places the graphs in a single row (side-by-side) for CDF plots.

By default, ORIENT=HORIZONTAL.

You can specify the following *plot-requests*:

ALL

requests all applicable plots for all variables that are specified in the ASSESS statement. These plots include bar charts for binary classification variables; box plots, CDF plots, and cloud plots for continuous variables; and a combined standardized mean differences plot for all variables. If you specify a STRATA statement, then PROC PSMATCH also produces the plots by stratum.

BAR < (< DISPLAY=ALL | (*bar-list*) > < TYPE=FREQ | PROP >) >

BARCHART < (< DISPLAY=ALL | (*bar-list*) > < TYPE=FREQ | PROP >) >

requests comparative bar charts for binary classification variables. You can use the DISPLAY= option to select variables for which bar charts are to be displayed.

DISPLAY=ALL

requests bar charts for binary classification variables that are specified by the ALLCOV or VAR= option.

DISPLAY=(*bar-list*)

specifies a subset of the binary classification variables for which bar charts are to be displayed.

By default, DISPLAY=ALL.

You can use the TYPE= option to select either the frequencies or the proportions to be displayed in the bar charts.

TYPE=FREQ

displays frequencies of levels for the binary classification variable.

TYPE=PROP

displays proportions of levels for the binary classification variable.

By default, TYPE=PROP.

If you specify a STRATA statement, then the bar charts are also displayed by stratum.

BOX < (DISPLAY=ALL | (*box-list*)) >

BOXPLOT < (DISPLAY=ALL | (*box-list*)) >

requests box plots for continuous variables. You can use the DISPLAY= option to select variables for which box plots are to be displayed.

DISPLAY=ALL

requests box plots for all continuous variables that are specified by the ALLCOV or VAR= option. The option also requests box plots for logits of propensity scores if the LPS option is specified and propensity scores if the PS option is specified.

DISPLAY=(*box-list*)

specifies a subset of the continuous variables for which box plots are to be displayed.

By default, DISPLAY=ALL.

If you specify a STRATA statement, then the box plots are also displayed by stratum.

CDF < (DISPLAY=ALL | (*cdf-list*)) >

CDFPLOT < (DISPLAY=ALL | (*cdf-list*)) >

requests cumulative distribution (CDF) plots for continuous variables. You can use the DISPLAY= option to select variables for which CDF plots are to be displayed.

DISPLAY=ALL

requests CDF plots for all continuous variables that are specified by the ALLCOV or VAR= option. The option also requests CDF plots for logits of propensity scores if the LPS is specified and propensity scores if the PS option is specified.

DISPLAY=(*cdf-list*)

specifies a subset of the continuous variables for which CDF plots are to be displayed.

By default, DISPLAY=ALL.

If you specify a STRATA statement, then the CDF plots by stratum are also displayed.

CLOUD < (DISPLAY=ALL | (*cloud-list*)) >

CLOUDPLOT < (DISPLAY=ALL | (*cloud-list*)) >

requests cloud plots for continuous variables. The term cloud plot is used here to refer to scatter plots in which the points have been jittered by adding random noise to prevent overplotting. Jittering typically occurs when a continuous variable (such as age) is rounded to some convenient unit (such as years). You can use the DISPLAY= option to select variables for which cloud plots are to be displayed.

DISPLAY=ALL

requests cloud plots for all continuous variables that are specified by the ALLCOV or VAR= option. The option also requests cloud plots for logits of propensity scores if the LPS option is specified and propensity scores if the PS option is specified.

DISPLAY=(*cloud-list*)

specifies a subset of the continuous variables for which cloud plots are to be displayed.

By default, DISPLAY=ALL.

If you specify a STRATA statement, then cloud plots are also displayed by stratum.

NONE

suppresses all plots.

STDDIFF < (REF=*r*) >

STDDIFFPLOT < (REF=*r*) >

requests a standardized mean differences plot for all variables that are specified by the ALLCOV or VAR= option. The plot also includes logits of propensity scores if the LPS option is specified and propensity scores if the PS option is specified.

The REF=*r* option displays a shaded band that covers standardized mean differences from $-r$ to r, where $r > 0$. If you specify REF=0, the band is not displayed. By default, REF=0.25 (Rubin 2001, p. 174).

If you specify a STRATA statement, then standardized mean difference plots are also displayed by stratum. However, the shaded band is not displayed in the plot because recommended ranges for stratum-specific standardized mean differences are currently not available in the literature.

WGTCLOUD < (REF=*r*) >

WGTCLOUDPLOT < (REF=*r*) >

requests cloud plots for weights. The option is applicable if you specify the WEIGHT=ATEWGT or WEIGHT=ATTWGT option in the ASSESS statement. The term cloud plot is used here to refer to scatter plots in which the points have been jittered by adding random noise to prevent overplotting.

Observations that have large weights can be highly influential. Well-behaved ATE weights should be less than 10 times the expected weight (Stürmer et al. 2014, p. 578). For more information about expected weights, see the section "Propensity Score Weighting" on page 8407. The REF=*r* option displays a reference line at *r* times the expected weight. By default, REF=10.

Weight cloud plots for ATT weights and IPTW-ATE weights display distinct reference lines for weights for observations in the treated and control groups. For example, see Output 101.1.7. Weight cloud plots for stabilized IPTW-ATE weights display a single reference line at *r* because the expected weight is 1.

By default, PLOTS=STDDIFF.

STDBINVAR=YES | NO

specifies whether to display standardized binary variables in the standardized mean differences table and plot.

YES displays standardized binary variables in the standardized mean differences table and plot.

NO does not display standardized binary variables in the standardized mean differences table and plot, and displays raw binary variables in the standardized mean differences plot.

By default, STDBINVAR=YES.

STDDEV=POOLED < (ALLOBS=YES | NO) >

STDDEV=TREATED < (ALLOBS=YES | NO) >

specifies the standard deviation used in computing standardized mean differences.

POOLED uses the pooled standard deviation, which is computed as the square root of the average of the sample variances for the treated group and the sample variance for the control group.

TREATED uses the standard deviation of the variable values in the treated group only.

By default, STDDEV=POOLED.

The ALLOBS= option specifies the set of observations used to compute the variance.

YES uses the sample variances that are derived from all observations to compute the standardized mean differences for all observations, for observations in the support region, for stratified observations, and for matched observations.

NO uses the variance derived from all observations to compute the standardized mean differences for all observations, uses the variance derived from observations in the support region to compute the standardized mean differences for observations in the support region, uses the variance derived from stratified observations to compute the standardized mean differences for stratified observations, and uses the variance derived from matched observations to compute the standardized mean differences for matched observations.

By default, ALLOBS=YES.

VARINFO
requests a variable information table for the treated and control groups.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC PSMATCH to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the PSMATCH procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

CLASS Statement

CLASS *variables* **;**

The required CLASS statement specifies the following input variables, which are used as classification variables:

- the variable to use as the treatment indicator in the PSDATA, PSMODEL, and EWEIGHT statements
- the classification covariates in the logistic model in the PSMODEL statement
- the classification variables that you specify in the VAR= option in the ASSESS statement

If you specify a PSMODEL statement, the CLASS statement must precede the PSMODEL statement. Classification variables can be either character or numeric.

EWEIGHT Statement

EWEIGHT TREATVAR=*treatvar*< **(***trt-option***)** >
BALANCEVARS=(*variables* < */moments-option* >**)** < *options* > **;**

The EWEIGHT statement performs entropy balancing. This statement is ignored if you specify the MATCH, PSWEIGHT, or STRATA statement. If entropy balancing is performed, the PSMODEL statement, PSDATA statement, REGION= option, OBS= option, and all propensity score–related options are ignored.

You must specify the following options:

BALANCEVARS=(*variables* **< / MOMENTS= 1 | 2 >)**
specifies the *variables* to balance. These variables can be either continuous or categorical. For continuous variables, the first and possibly second moments are balanced between the treated and control groups. For categorical variables, the proportions are balanced between the treated and control groups.

The MOMENTS= suboption specifies the number of moments to balance. This suboption does not apply to categorical variables. By default, MOMENTS=2.

TREATVAR=*treatvar* **< (TREATED='***level***' |** *keyword***) >**
names the treatment indicator variable, *treatvar*, which must be a binary classification variable that you specify in the CLASS statement.

The TREATED= suboption specifies the treated level for the binary treatment variable. You can specify the value of the treated *level* in quotation marks, or you can specify one of the following *keywords*:

FIRST designates the first ordered level as the treated group.

LAST designates the last ordered level as the treated group.

By default, TREATED=FIRST.

In addition to the required BALANCEVARS= and TREATVAR= options, you can specify the *options* that are summarized in Table 101.3. The *options* are described fully in alphabetical order after the table.

Table 101.3 Summary of PROC PSMATCH Options

Option	Description
ABSGTOL=	Specifies an absolute gradient convergence criterion
BALTOL=	Specifies the tolerance to be used to evaluate the balance constraints
MAXFUNC=	Specifies the maximum number of iterations in the optimization process
MAXITER=	Specifies the maximum number of function calls in the optimization process
NLARGEST=	Displays observations that have the largest weights
REFWEIGHT=	Names the variable that contains the reference weights
TOTALWEIGHT=	Specifies the total of the entropy balancing weights computed for a treatment condition
WEIGHT=	Specifies the observations for which entropy balancing weights are computed

ABSGTOL=*r*

ABSGCONV=*r*

specifies the absolute gradient convergence criterion for the entropy balance dual optimization problem. Termination requires the maximum absolute gradient element for the dual problem to be small:

$$\max_j |g_j(\boldsymbol{\beta}^{(k)})| \leq r$$

Here, $\boldsymbol{\beta}$ is the vector of parameters in the optimization and $g_j(\cdot)$ is the gradient of the objective function with respect to the *j*th parameter. Note that the gradient of the objective function for the dual problem with respect to the *j*th parameter corresponds to the amount by which the *j*th balance constraint is violated for the primal problem. For more information about the entropy balance optimization problem, see the section "Entropy Balancing" on page 8423.

The value of *r* must be positive. By default, ABSGTOL=0.0001.

BALTOL=*r*

BALANCETOL=*r*

specifies the tolerance value to be used to evaluate the balance constraints and determine acceptance of the entropy balance weights.

If the entropy balance optimization problem fails to converge, the weights from the final iteration are accepted if the maximum absolute violation of the balance constraints is less than or equal to *r*. If the entropy balance optimization problem converges, the weights are rejected if a linearly dependent balance constraint is violated by a value greater than *r*.

The value of *r* must be positive and greater than or equal to the ABSGTOL= option. By default, BALTOL=0.0001.

MAXFUNC=*n*

MAXF=*n*

specifies the maximum number *n* of function calls in the optimization process. By default, MAXFUNC=1000.

MAXITER=*n*

MAXI=*n*

specifies the maximum number *n* of iterations in the optimization process. By default, MAXITER=100.

NLARGESTWGT=*n*

displays a table of the observations that have the *n* most extreme weights. If you specify an ID statement, the corresponding values of the ID variables are also displayed and identify the observations. By default, $n = 0$ and the table is not displayed.

REFWEIGHT< (WEIGHTMOMENTS=YES | NO) >=*variable*

specifies the variable in the DATA= data set that contains the reference weights to use for the entropy balance optimization problem. The reference weight values must be positive. If you do not specify a variable, uniform reference weights are used.

The WEIGHTMOMENTS= suboption specifies whether or not the target moments and proportions for the entropy balance optimization problem constraints should be weighted by the reference weights. By default, WEIGHTMOMENTS=YES.

TOTALWEIGHT=*r*

specifies the total of the entropy balance weights that are computed for a treatment condition. The value of *r* must be positive. If you do not specify an *r* value, the total of the weights is determined by the WEIGHT= option. When WEIGHT=CONTROL, the default total weight for the control units is equal to the number of treated units. When WEIGHT=ALLOBS, the default total weight for each group is equal to the number of treated and control units.

WEIGHT=CONTROL | ALLOBS

specifies the observations for which entropy balance weights are computed.

When WEIGHT=CONTROL, the procedure entropy balance weights only for the units in the control condition, and the target moments are determined by the covariate values of the treated units. For units in the treatment condition, the weights that are used in balance diagnostics and in the OUT= data set are set to the reference weight values if the REFWEIGHT= option is specified and set to 1 otherwise.

When WEIGHT=ALLOBS, the procedure computes entropy balance weights for both the treated and control units by solving two separate optimization problems. The target moments that define the constraints for both optimization problems are determined by the covariate values of all units.

By default, WEIGHT=CONTROL.

FREQ Statement

FREQ *variable* **;**

The FREQ statement identifies a *variable* that contains the frequency of occurrence of each observation. PROC PSMATCH treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. The FREQ statement is not allowed if a MATCH statement is specified.

ID Statement

ID *variables* **;**

The ID statement specifies one or more variables whose values identify the observations that are displayed in tables of extreme weights and most matches that are requested by the NLARGESTWGT= option in the EWEIGHT or PSWEIGHT statement and the NMATCHMOST= option in the MATCH statement.

MATCH Statement

MATCH < *options* > **;**

The MATCH statement matches observations in the control group to observations in the treatment group. The MATCH statement is not allowed if you specify a FREQ statement. The EWEIGHT, PSWEIGHT, and STRATA statements are ignored if you specify a MATCH statement.

Table 101.4 summarizes the *options* in the MATCH statement.

Table 101.4 MATCH Statement Options

Option	Description
CALIPER=	Specifies the caliper width requirement for matching
DISTANCE=	Specifies the distance for comparing treated units and control units
EXACT=	Requests exact matching for specified classification variables
METHOD=	Specifies the method to use for matching
NMATCHMOST=	Displays observations that have the greatest numbers of matches
WEIGHT=	Specifies the type of weight for matched observations

The flowchart in Figure 101.13 displays the steps in the propensity score matching process.

Figure 101.13 Propensity Score Matching Options

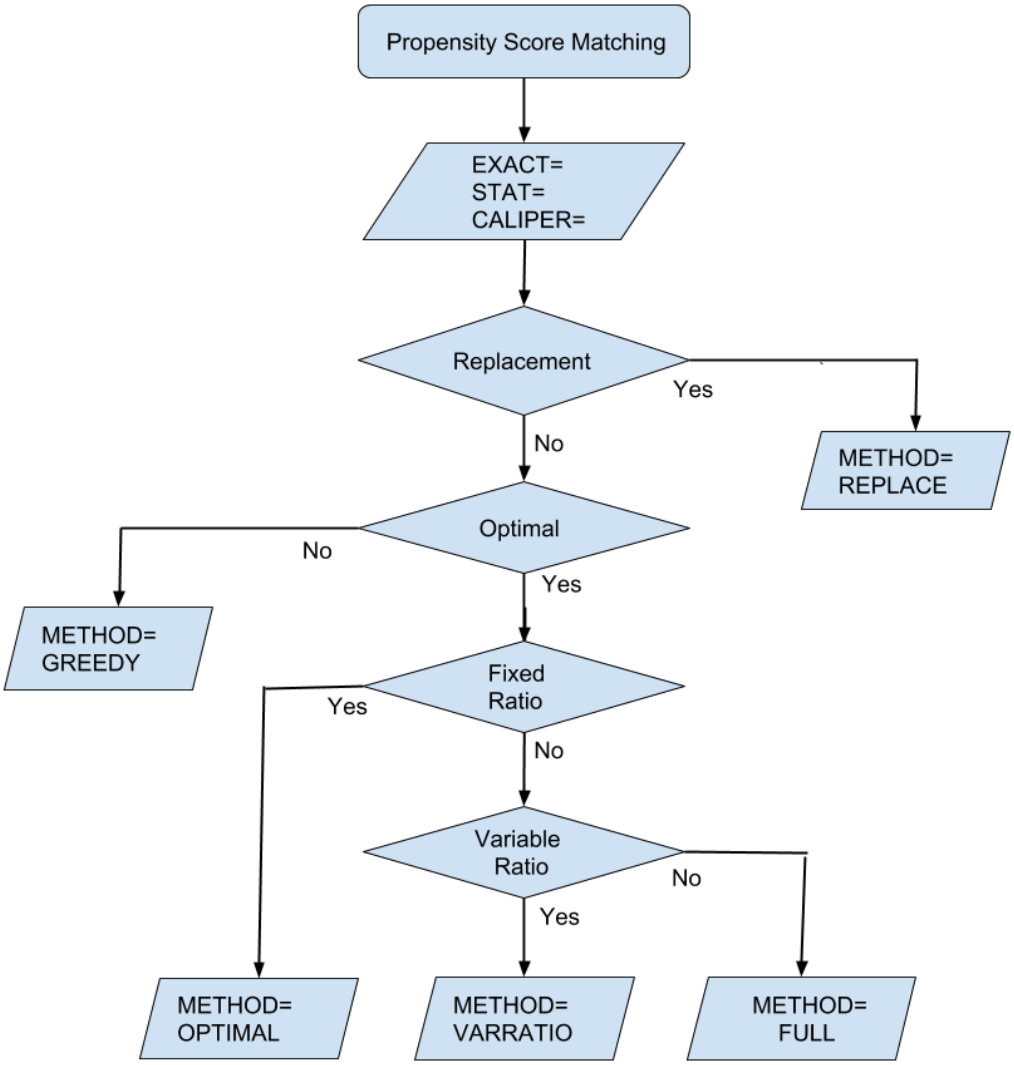

You can specify the following *options* in the MATCH statement:

CALIPER < (*caliper-options*) > = *r*

specifies the caliper width requirement for matching, where *r* is either missing or greater than 0. The difference in propensity scores (or logits of propensity scores) between the treated unit and its matching control unit must be less than or equal to *r*. If you specify CALIPER=., then the caliper requirement is ignored. By default, CALIPER=0.25 (Rosenbaum and Rubin 1985, p. 37). Austin (2011a) has shown that CALIPER=0.20 is optimal in many settings.

You can use the following two *caliper-options* to prescribe the caliper requirement:

MULT=ONE | STDDEV

specifies the multiplier for the specified caliper width *r*.

ONE uses *r* for the caliper width.

STDDEV uses *r* times the pooled estimate of the standard deviation of the logit of the propensity score (if you specify DISTANCE=LPS) or the propensity score (if you specify DISTANCE=PS), where this estimate is computed as the square root of the average of the variances in the treated and control groups.

By default, MULT=STDDEV.

MAHDISTANCE=LPS | PS

specifies the type of distance to be used in the caliper width computation if you specify the DISTANCE=MAH option in the MATCH statement.

LPS uses the logit of the propensity score.

PS uses the propensity score scale.

By default, MAHDISTANCE=LPS.

If you specify the DISTANCE=LPS or DISTANCE=PS option in the MATCH statement, the specified type of distance is used in the caliper width computation.

DISTANCE=*distance*

specifies the type of distance to be compared when treated units are matched to control units. If you specify the DISTANCE=LPS or DISTANCE=PS option, the specified type of distance is also used in the caliper width computation. By default, DISTANCE=LPS. You can specify the following values for *distance*:

LPS

specifies matching that minimizes the difference between the logits of the propensity scores for the two units.

PS

specifies matching that minimizes the difference between the propensity scores for the two units.

MAH (*var-options* **<** / *mah-options* **>)**

specifies matching that minimizes the Mahalanobis distance between the two units.

You use the following *var-options* to select at least one variable for computing the Mahalanobis distance:

LPS

includes the logit of the propensity score.

PS

includes the propensity score.

VAR=(*var-list*)
includes variables in the specified *var-list*. These variables must be continuous variables in the input data set.

You can also specify the following *mah-options*:

COV=CONTROL | IDENTITY | POOLED
specifies the type of covariance matrix in the Mahalanobis distance.

CONTROL uses the covariance matrix that is computed from observations in the control group.

IDENTITY uses the identity matrix, and the resulting distance is the Euclidean distance.

POOLED uses the pooled covariance matrix that is computed from observations in the treated group and observations in the control group.

By default, COV=CONTROL.

SQRT=YES | NO
specifies whether to apply the square root transformation to the Mahalanobis distance in the difference computation. This *mah-option* does not affect matching results for greedy nearest neighbor matching or matching with replacement. It affects only results for optimal matching that minimize the total absolute difference.

YES uses the square root of the Mahalanobis distance as the difference between treated and control units.

NO uses the Mahalanobis distance as the difference between treated and control units.

By default, SQRT=YES.

EXACT=*variable* | (*variables*)
specifies classification *variables* that are to be matched exactly. That is, observations in each matched set must have the same values for these variables. The *variables* must be specified in the CLASS statement.

METHOD=*method* < (*method-options*) >
specifies the *method* for the matching. You can specify the following *methods* and *method-options*. By default, METHOD=OPTIMAL.

METHOD=FULL (KMAX=*kmax* < *full-options* >)
requests optimal full matching. Each treated unit is matched with one or more control units, and each control unit (if matched) is matched with one or more treated units. If the specified total number of control units to be matched is less than the number of available control units, then constrained full matching is performed—that is, not all observations are matched.

You must specify the following suboption:

KMAX=*kmax*
specifies the maximum number of control units to be matched with each treated unit, where *kmax* $\geq$ 1.

You can also specify the following *full-options*:

KMAXTREATED=*kmaxtrt*

KMAXTRT=*kmaxtrt*
specifies the maximum number of treated units for each control, where *kmaxtrt* $\geq$ 1. By default, KMAXTREATED=2.

KMEAN=*kmean*
specifies the average number of control units to be matched with each treated unit. If the resulting number of control units is greater than the number of control units in the support region, the number of control units in the support region is used.

NCONTROL=*m*
specifies the number of control units to be matched. If *m* is greater than the number of control units in the support region, the number of control units in the support region is used.

PCTCONTROL=*p*
specifies the percentage of the total number of control units to be matched. If the resulting number of control units is greater than the total number of control units in the support region, the number of control units in the support region is used.

You can specify only one of the KMEAN=, NCONTROL=, and PCTCONTROL= options for the number of control units in the matched data set. If you do not specify any of the KMEAN=, NCONTROL=, and PCTCONTROL= options, KMEAN= (*kmax* + 1 / *kmaxtrt*) / 2 is used.

METHOD=GREEDY < (K=*k* **ORDER=***order-option***) >**
requests greedy nearest neighbor matching, in which each treated unit is sequentially matched with the *k* nearest control units. Matching depends on the ordering of the treated units, which you can specify in the ORDER= suboption.

You can specify the following suboptions:

K=*k*
specifies the number of matching control units, where *k* > 0, for each treated unit. PROC PSMATCH performs *k* separate loops of matching for treated units. In each loop, the nearest control unit is sequentially matched to each treated unit. By default, K=1 (one control unit for each treated unit).

ORDER=ASCENDING | DESCENDING | RANDOM < (SEED=*number***) >**
specifies the ordering of treated units that are used to find the matching control units. You can specify one of the following values:

ASCENDING
orders the treated units in ascending order of the propensity score.

DESCENDING
orders the treated units in descending order of the propensity score.

RANDOM < (SEED=*number*) >
orders the treated units in random order of the propensity score. The SEED= suboption specifies a positive integer to start the pseudorandom number generator. If the SEED= option is not specified, the value is generated from reading the time of day from the computer's clock.

By default, ORDER=DESCENDING.

METHOD=OPTIMAL < (K=*k*) >
requests optimal fixed ratio matching. The K=*k* suboption specifies the number of matching control units, where $k > 0$, for each treated unit. By default, K=1 (one control unit is matched with each treated unit).

METHOD=REPLACE < (K=*k*) >
requests a fixed number *k* of unique matching control units for each treated unit, where the matched control units are selected with replacement. This means that each control unit can be matched to more than one treated unit, but it can only be matched once to the same treated unit. The K=*k* suboption specifies the number of matching control units, where $k > 0$, for each treated unit. By default, K=1 (one control unit is matched with each treated unit).

METHOD=VARRATIO (KMAX=*kmax* < *vr-options* >)
requests optimal variable ratio matching. Each treated unit is matched with one or more control units.

You must specify the following suboption:

KMAX=*kmax*
specifies the maximum number of control units to be matched with each treated unit, where $kmax \geq 1$.

You can also specify the following *vr-options*:

KMEAN=*kmean*
specifies the average number of control units to be matched with each treated unit. If the resulting number of control units is greater than the total number of control units in the support region, the number of control units in the support region is used.

KMIN=*kmin*
specifies the minimum number of control units to be matched with each treated unit. By default, KMIN=1.

NCONTROL=*m*
specifies the total number of control units to be matched. If *m* is greater than the total number of control units in the support region, the number of control units in the support region is used.

PCTCONTROL=*p*
specifies the percentage of total control units to be matched. If the resulting number of control units is greater than the total number of control units in the support region, the number of control units in the support region is used.

You can specify only one of the KMEAN=, NCONTROL=, and PCTCONTROL= options for the number of control units in the matched data set. If you do not specify any of the KMEAN=, NCONTROL=, and PCTCONTROL= options, then KMEAN= (*kmin* + *kmax*) / 2 is used.

NMATCHMOST=*n*
displays a table of the observations that have the greatest numbers of matches, where $n \leq 50$. This option displays observation numbers and numbers of matches for the *n* observations that have the greatest numbers of matches in the treated and control groups. If an ID statement is also specified, the corresponding values of the ID variables are also displayed identify the observations. The option is not applicable to greedy matching (METHOD=GREEDY) and optimal fixed ratio matching (METHOD=OPTIMAL), where a fixed number of control units are matched to each treated unit. By default, $n = 0$ and the table is not displayed.

WEIGHT=ATEWGT | ATTWGT | EQUAL | MATCHATEWGT | MATCHATTWGT | MATCHWGT | NONE
specifies the type of weight for matched observations.

ATEWGT | MATCHATEWGT
weights the treatment group up to the total size of the matched set. That is, in each matched set, the total weight of treated units equals the total number of units in the matched set, and the total weight of control units also equals the total number of units in the matched set. This weighting is available only for an optimal full matching (METHOD=FULL), and it is appropriate for estimating the ATE. For more information about using match weighting to estimate the ATE, see the section "ATE Weighting after Full Matching" on page 8417.

ATTWGT | MATCHATTWGT | MATCHWGT
weights the control group up to the size of the treatment group in the matched set. That is, in each matched set, the total weight of control units equals the number of treated units in the matched set. This weighting is appropriate for estimating the ATT. For more information about using match weighting to estimate the ATT, see the sections "ATT Weighting after Matching without Replacement" on page 8416 and "ATT Weighting after Matching with Replacement" on page 8417.

EQUAL | NONE
uses the same weight for matched observations. That is, each matched unit has a weight of 1, regardless of the treatment group.

By default, WEIGHT=ATTWGT.

OUTPUT Statement

OUTPUT OUT < **(OBS=***obs-value***)** >**=***SAS-data-set* < *keyword***=***name* < *keyword***=***name* . . . >> **;**

The OUTPUT statement specifies the output data set and variables. You must specify the following option:

OUT < **(OBS=***obs-value***)** >**=***SAS-data-set*

names the output data set. The data set also includes the results of matching if you provide the MATCH statement. You can specify one of the following values for *obs-value*:

OBS=ALL requests that the output data set contain all observations.

OBS=REGION requests that the output data set contain only observations in the specified support region.

OBS=MATCH requests that the output data set contain only the matched treated units and control units. This option applies only if you specify the MATCH statement.

By default, OBS=ALL. The OBS= suboption is ignored if you specify the EWEIGHT statement.

Table 101.5 summarizes the *keywords* that are used to create and name the output variables.

Table 101.5 OUTPUT Statement Keywords

Keyword	Description
LPS	Specifies the variable that provides the logit of the propensity score
MATCHID	Specifies the variable that provides identification numbers for the matched units
PS	Specifies the variable that provides the propensity score
STRATA	Specifies the variable that numbers the strata
WEIGHT	Specifies the weight variable

LPS=*name*

creates and names the variable that provides the logit of the propensity score. This suboption is ignored if you specify the EWEIGHT statement.

MATCHID=*name*

creates and names the variable that provides identification numbers for the matched treated and control units. This suboption applies only if you specify the MATCH statement.

PS=*name*

creates and names the variable that provides the propensity score. This suboption is ignored if you specify the EWEIGHT statement.

If you omit this option and the PS= option in the PSDATA statement, then the variable that provides the propensity score is automatically created using *name*=_PS_.

STRATA=*name*

creates and names the variable that numbers the strata. This suboption applies only if you specify the STRATA statement.

If you omit this suboption but specify the STRATA statement, then the variable that numbers the strata is automatically created using *name*=_STRATA_.

WEIGHT=*name*

creates and names the weight variable. The output weight variable depends on the specified statements:

- If you specify an EWEIGHT statement, the weight variable contains weights as specified by the WEIGHT= option in the EWEIGHT statement.
- If you specify a MATCH statement, the weight variable contains weights of matched observations as specified by the WEIGHT= option in the MATCH statement.
- If you specify a PSWEIGHT statement, the weight variable contains weights of observations as specified by the WEIGHT= option in the PSWEIGHT statement.

This option is ignored if you specify a STRATA statement.

PSDATA Statement

PSDATA TREATVAR=*treatvar* < **(TREATED=**'*level*' | *keyword*) > *ps-option* **;**

You use the PSDATA statement when the DATA= data set contains precomputed propensity scores or logits of propensity scores and you want to base the propensity score analysis on these scores rather than using the PSMODEL statement to specify a logistic regression model for computing the scores. The PSDATA statement specifies the variable in the DATA= data set that is the treatment indicator variable and a variable that contains the propensity scores or logits of the propensity scores. If you omit the EWEIGHT statement, you must specify either the PSMODEL or PSDATA statement, but not both.

You must specify the following TREATVAR= option:

TREATVAR=*treatvar* **< (TREATED='***level*' | *keyword*) >**

names the treatment indicator variable, *treatvar*, which must be a binary classification variable that you specify in the CLASS statement.

The TREATED= suboption specifies the treated level for the binary treatment variable. You can specify the value of the treated *level* in quotation marks, or you can specify one of the following *keywords*:

FIRST designates the first ordered level as the treated group.

LAST designates the last ordered level as the treated group.

By default, TREATED=FIRST.

You must also specify one (and only one) of the following *ps-options*:

PS=*name*
names the variable that contains propensity scores, where the variable *name* must be a variable in the DATA= data set.

LPS=*name*
names the variable that contains logits of propensity scores, where the variable *name* must be a variable in the DATA= data set.

PSMODEL Statement

PSMODEL *treatvar* < **(***trt-option***)** > **=** < *effects* > < / **WEIGHT=***weight* > **;**

The PSMODEL statement specifies the logistic regression model for computing propensity scores. If you omit the EWEIGHT statement, you must specify either the PSMODEL or PSDATA statement, but not both.

The treatment indicator variable *treatvar* must be a binary classification variable that is listed in the CLASS statement, and the *effects* are the explanatory effects, which can include variables, main effects, interactions, and nested effects for the logistic regression model.

You can use the following *trt-option* to specify the treated level for the binary treatment variable:

TREATED='*level***'** | *keyword*
models the probability of the specified treated level. You can specify the value of the treated *level* in quotation marks, or you can specify one of the following *keywords*:

FIRST designates the first ordered level as the treated group.

LAST designates the last ordered level as the treated group.

By default, TREATED=FIRST.

You can specify the following option to fit a weighted logistic regression:

WEIGHT=*weight*
specifies a variable that contains the weight of each observation that is used in fitting the logistic regression model to derive the propensity scores. These weights should not be confused with weights that PROC PSMATCH derives from the propensity scores.

PSWEIGHT Statement

PSWEIGHT < *options* > **;**

The PSWEIGHT statement computes weights for the observations on the basis of propensity scores. The PSWEIGHT statement does not apply when you specify either a MATCH statement or a STRATA statement.

Table 101.6 summarizes the *options* in the PSWEIGHT statement.

Table 101.6 PSWEIGHT Statement Options

Option	Description
NLARGESTWGT=	Displays observations that have the largest weights
WEIGHT=	Specifies the type of weight for observations

NLARGESTWGT=*n*

displays a table of the observations that have the most extreme weights, where $n \leq 50$. This option displays observation numbers and weights for the *n* observations that have the largest weights in the treated and control groups. If you specify an ID statement, the corresponding values of the ID variables are also displayed and serve to identify the observations. By default, *n*=0 and the table is not displayed.

WEIGHT=ATEWGT | ATTWGT

specifies the type of weight for observations.

ATEWGT < (STABILIZE=YES | NO) >

uses inverse probability of treatment weighting (IPTW). These weights are appropriate for estimation of the ATE. For more information about IPTW-ATE weighting, see the section "Inverse Probability of Treatment Weighting" on page 8408.

YES requests a stabilized inverse probability of treatment weighting.

NO requests an inverse probability of treatment weighting.

By default, STABILIZE=NO.

ATTWGT

uses ATT weighting (also called weighting by odds) to weight the control group up to the treatment group. These weights are appropriate for estimation of the ATT. For more information about ATT weighting, see the section "ATT Weighting" on page 8409.

By default, WEIGHT=ATTWGT.

STRATA Statement

STRATA < *options* > **;**

The STRATA statement divides observations in the support region into strata based on propensity scores, where the support region is specified in the REGION= option in the PROC PSMATCH statement. The STRATA statement does not apply when you specify the MATCH statement.

Table 101.7 summarizes the *options* in the STRATA statement.

Table 101.7 STRATA Statement Options

Option	Description
KEY=	Specifies how to use the observations to construct the strata
NSTRATA=	Specifies the number of strata
STRATUMWGT=	Specifies the type of stratum weights to combine statistics across strata

KEY=TOTAL | TREATED

specifies how to use the observations to construct the strata.

TOTAL requests that each stratum contain approximately the same number of total units, which can be in either the treated group or the control group.

TREATED requests that each stratum contain approximately the same number of units in the treated group.

By default, KEY=TREATED. This option balances the number of treated units across strata, so that a reliable estimate of the treatment effect can be obtained for each stratum. However, a common alternative is to construct strata so that each stratum has the same number of total units. You can request this approach by specifying KEY=TOTAL. In either case, you should examine the number of treated units and the number of control units in each stratum to make sure that a reliable estimate can be obtained for each stratum.

NSTRATA=*n*

specifies the number of strata, where $n \geq 2$. Only observations in the support region are stratified. By default, NSTRATA=5.

STRATUMWGT=TOTAL | TREATED

STRATUMWEIGHT=TOTAL | TREATED

specifies the type of stratum weights that the PSMATCH procedure uses to combine stratum-specific statistics in the assessment of variable balance after stratification. For more information about stratum weights, see the section "Weighting after Stratification" on page 8411.

TOTAL uses the proportions of total units (treated and control units combined) in the strata as the weights. These weights sum up to one and are appropriate for estimating the ATE.

TREATED uses the proportions of treated units in the strata as the weights. These weights sum up to one and are appropriate for estimating the ATT.

By default, STRATUMWGT=TOTAL.

For more information, see the section "Propensity Score Stratification" on page 8410.

Details: PSMATCH Procedure

Observational Studies Contrasted with Randomized Trials

In a randomized study, such as a randomized controlled trial, the subjects are randomly assigned to a treated (exposure) group or a control (nonexposure) group. Random assignment ensures that the distribution of the covariates is the same in both groups, and the treatment effect can be estimated from a direct comparison of the outcomes for the subjects in the two groups.

In contrast, the subjects in an observational study are not randomly assigned to the treated and control groups. Confounding can occur if some covariates are related to both the treatment assignment and the outcome. Consequently, there can be systematic differences between the treated subjects and the control subjects. The presence of confounding requires statistical approaches that remove the effects of confounding when estimating the effect of treatment.

Observational studies are carried out when it is impractical or unethical to perform a randomized experiment. One example of an observational study is a retrospective cohort study that examines the relationship between a specific disease and a risk factor that occurred in the past; another example is a nonrandomized clinical trial that uses existing data such as control units that are extracted from a registry database.

The approach that the PSMATCH procedure uses and the following terminology are based on the potential outcomes framework for causal inference, which was introduced by Rubin (1974) and Rosenbaum and Rubin (1983). Under this framework, each individual typically has two potential outcomes in an observational study whose goal is to estimate the effect of a treatment:

- $Y(1)$, the outcome that would be observed if the individual receives the treatment.
- $Y(0)$, the outcome that would be observed if the individual does not receive the treatment under identical circumstances to those under which the subject would have received the treatment.

However, only one outcome can be observed.

The treatment effect is defined as $Y(1) - Y(0)$, and the average treatment effect is defined as:

$$\text{ATE} = E\left(Y(1) - Y(0)\right)$$

The average treatment effect for the treated (individuals who actually receive treatment) is defined as:

$$\text{ATT} = E\left(Y(1) - Y(0) \mid T = 1\right)$$

where T denotes the treatment assignment.

In a randomized trial, the potential outcomes $(Y(0), Y(1))$ and the treatment assignment (T) are independent:

$$(Y(0), Y(1)) \perp\!\!\!\perp T$$

Thus, the average treatment effect (ATE) is identical to the average treatment effect for the treated (ATT), which can be expressed as follows and can be estimated from the observed data:

$$E\left(Y(1) \mid T = 1\right) - E(Y(0) \mid T = 0)$$

In an observational study, the potential outcomes $(Y(0), Y(1))$ and the treatment assignment (T) might not be independent. In this case, the ATE and ATT are not the same. Furthermore, outcomes cannot be compared directly to estimate the treatment effect. In particular,

$$\begin{aligned} \text{ATT} &= E\left(Y(1) - Y(0) \mid T = 1\right) \\ &= E\left(Y(1) \mid T = 1\right) - E\left(Y(0) \mid T = 0\right) + E\left(Y(0) \mid T = 0\right) - E\left(Y(0) \mid T = 1\right) \end{aligned}$$

The following term can be estimated from the observed data:

$$E\left(Y(1) \mid T = 1\right) - E\left(Y(0) \mid T = 0\right)$$

However, the selection bias cannot be estimated from the observed data:

$$E\left(Y(0) \mid T = 0\right) - E\left(Y(0) \mid T = 1\right)$$

The selection bias is the average difference in the response that would be observed between individuals in the control group who do not receive treatment and individuals in the treatment group who do not receive treatment. Thus, the usual observed difference between the treated and control groups cannot be used to estimate the treatment effect. For subjects who are not randomly assigned to the treated and control groups, the baseline variables could be related to both the treatment assignment and the outcome, and consequently direct comparison of outcomes could result in biased estimates.

One strategy for correctly estimating the treatment effect is based on the propensity score, which is the conditional probability of the treatment assignment given the observed variables. You use propensity scores to account for confounding by weighting observations, by creating strata of subjects that have similar propensity scores, or by matching control subjects to treated subjects. This is done prior to the outcome analysis and without knowledge of the outcome variable (Rosenbaum and Rubin 1984; Stuart 2010, p. 5). The following section describes the propensity score approach.

Propensity Score Analysis

In a randomized study, the potential outcomes within treatment and control groups are unrelated to treatment assignment because individuals are randomly assigned to the groups. Consequently the treatment assignment given the variables X is strongly ignorable.

Rosenbaum and Rubin (1983) defined treatment assignment to be strongly ignorable when two conditions are met. The first condition (unconfoundedness) states that the potential outcomes $(Y(0), Y(1))$ and the treatment assignment (T) are conditionally independent given the observed baseline variables:

$$(Y(0), Y(1)) \perp\!\!\!\perp T \mid X = x$$

This condition is called the "no unmeasured confounders" assumption because it assumes that all the variables that affect both the outcome and the treatment assignment have been measured. The second

condition (probabilistic assignment) states that there is a positive probability that a subject receives each treatment:

$$0 < \Pr(T = 1 \mid X = x) < 1$$

When the treatment assignment in an observational study is assumed to be strongly ignorable, Rosenbaum and Rubin (1983, p. 43) showed that unbiased estimates of average treatment effects can be obtained by conditioning on the propensity score *e(x)*, which is the probability of the treatment assignment conditional on a set of observed variables *X*:

$$e(x) = \Pr(T = 1 \mid X = x)$$

At any value of the propensity score *e(x)*, the difference between the treatment and control means is an unbiased estimate of the average treatment effect at *e(x)*. Consequently, matching on the propensity score and propensity score stratification also produce unbiased estimates of treatment effects (Rosenbaum and Rubin 1983, p. 44).

Furthermore, the propensity score is a balancing score. At each value of the propensity score, the distributions of the variables X are the same in the treated and control groups (Rosenbaum and Rubin 1983, p. 44; Stuart 2010, p. 6). Thus, the treatment assignment *T* and observed variables *X* are conditionally independent given the propensity score Rosenbaum (2010, p. 72):

$$x \perp\!\!\!\perp T \mid e(x)$$

Propensity score analysis attempts to replicate the properties of a randomized trial with respect to the observed variables X. The steps involved in this analysis are described in the section "Process of Propensity Score Analysis" on page 8367.

The following subsections describe the support region and the propensity score methods that are available in the PSMATCH procedure.

Support Region

For stratification and matching, the PSMATCH procedure selects observations whose propensity scores lie in a support region that can be defined in several ways:

- Selecting all available observations. You can request this definition by specifying REGION=ALLOBS in the PROC PSMATCH statement.
- Selecting observations whose propensity scores lie in a specified range. You can request this definition by specifying REGION=ALLOBS and then additionally specifying range options.
- Selecting observations whose propensity scores lie in the region of common support for the propensity scores for observations in the treated and control groups. You can request this definition by specifying REGION=CS. This region can be extended by specifying the EXTEND suboption.
- Selecting observations whose propensity scores lie in the region of propensity scores for observations in the treated group. You can request this definition by specifying REGION=TREATED. This region can be extended by specifying the EXTEND suboption.

In combination with the REGION= option, you can specify the OUT(OBS=REGION) option in the OUTPUT statement to request that only observations in the support region be included in the output data set. You can specify this combination even without the use of stratification or matching. For example, you can use the REGION=ALLOBS(PMSIN=0.1 PSMAX=0.9) option to include only observations whose propensity scores are greater than or equal to 0.1 and less than or equal to 0.9 in the output data set.

Propensity Score Methods

You can use the propensity score methods in the PSMATCH procedure to create an output data set that contains a sample that has been adjusted (either by matching, stratification, or weighting) so that the distributions of the variables are balanced between the treated and control groups. The two groups differ only randomly in their observed or measured variables, as in a randomized study. You can then use the output data set in an outcome analysis to estimate the effect of the treatment.

The following propensity score methods are available in the PSMATCH procedure:

- weighting, which creates weights that are appropriate for estimating the ATE and ATT
- stratification, which creates strata based on propensity scores
- matching, which matches treated units with control units

Note that the outcome variable is not involved in these methods. For more information about these methods, see the sections "Propensity Score Weighting" on page 8407, "Propensity Score Stratification" on page 8410, and "Matching Process" on page 8412.

Propensity Score Weighting

The PSMATCH procedure provides the following methods for weighting observations when matching is not used:

- inverse probability of treatment weighting (IPTW), which is used to estimate the ATE
- stabilized IPTW-ATE weighting, which is used to estimate the ATE
- ATT weighting (also called weighting by odds), which is used to estimate the ATT

If an observation has a propensity score close to 0 or 1, its large IPTW-ATE or ATT weight might incorrectly affect the results in the subsequent weighted outcome analysis. You can use the PSMATCH procedure to examine the observations that have extreme weights.

The PSMATCH procedure also provides methods for weighting matched observations when matching is used (see the section "Weighting after Matching" on page 8416) and for weighting strata when stratification is used (see the section "Weighting after Stratification" on page 8411).

Inverse Probability of Treatment Weighting

Inverse probability of treatment weighting (IPTW) computes the weight for the *j*th observation with propensity score p_j as

$$w_j = \begin{cases} \frac{1}{p_j} & \text{for observations in the treated group} \\ \frac{1}{1-p_j} & \text{for observations in the control group} \end{cases}$$

These weights can be used in an outcome analysis to estimate the average treatment effect,

$$\text{ATE} = E(Y(1) - Y(0))$$

by weighting the two groups up to the full population. For example, for a treated unit with $p_j = 0.25$, the weight is 4, which represents four units in the full population.

Expected IPTW-ATE weights are given by

$$\bar{w} = \begin{cases} \frac{N_t+N_c}{N_t} = \frac{1}{p_t} & \text{for observations in the treated group} \\ \frac{N_t+N_c}{N_c} = \frac{1}{1-p_t} & \text{for observations in the control group} \end{cases}$$

where $p_t = N_t/(N_t + N_c)$ is the proportion of individuals in the treated group.

The PLOTS=WGTCLOUD option in the ASSESS statement requests cloud plots for weights. The plot displays a reference line at r/p_t for observations in the treated group and a reference line at $r/(1 - p_t)$ for observations in the control group, where *r*=10 by default. You can specify a different value for *r* in the PLOTS=WGTCLOUD(REF=*r*) option.

You can request inverse probability of treatment weighting by specifying the WEIGHT=ATEWGT option in the PSWEIGHT statement, and then by specifying the WEIGHT= option in the OUTPUT statement to create a variable that contains these weights.

Stabilized IPTW-ATE Weighting

If a treated unit has a propensity score close to 0 or a control unit has a propensity score close to 1, the resulting IPTW-ATE weight can be large. If a few observations have very large weights, the resulting IPTW-ATE estimator has a large variance and is not approximately normally distributed (Robins, Hernan, and Brumback 2000, p. 554).

In order to reduce large variances of this type, Robins, Hernan, and Brumback (2000, p. 554) replace the IPTW-ATE weights with stabilized IPTW-ATE weights:

$$w_j^* = \begin{cases} p_t w_j = \frac{p_t}{p_j} & \text{for observations in the treated group} \\ (1 - p_t) w_j = \frac{1-p_t}{1-p_j} & \text{for observations in the control group} \end{cases}$$

where $p_t = N_t/(N_t + N_c)$ is the proportion of individuals in the treated group.

That is, the stabilized IPTW-ATE weights are computed by multiplying the IPTW-ATE weights by the marginal probability of receiving the given treatment. Thus, the expected stabilized IPTW-ATE weight is 1 for observations in the treated group and for observations in the control group.

You can request stabilized inverse probability of treatment weighting by specifying the WEIGHT=ATEWGT(STABILIZE=YES) option in the PSWEIGHT statement, and then by specifying the WEIGHT= option in the OUTPUT statement to create a variable that contains these weights.

Observations that have large weights can be highly influential, and well-behaved stabilized weights should have a mean stabilized weight close to 1 and a maximum stabilized weight less than 10 (Stürmer et al. 2014, p. 578). That is, in each treatment group, ATE weights should have a mean IPTW-ATE weight close to their expected weight and a maximum IPTW-ATE weight less than 10 times their expected weight. For information about these expected weights, see the section "Inverse Probability of Treatment Weighting" on page 8408.

ATT Weighting

ATT weighting (also called weighting by odds) computes the weight for the *j*th observation with propensity score p_j as

$$w_j = \begin{cases} 1 & \text{for observations in the treated group} \\ \frac{p_j}{1-p_j} & \text{for observations in the control group} \end{cases}$$

These weights can be used in an outcome analysis to estimate the following average treatment effect for the treated units (individuals who actually receive treatment) by weighting the control group up to the treated group:

$$\text{ATT} = E(Y(1) - Y(0) \mid T = 1)$$

For example, for a control unit with $p_j = 0.75$, the weight is 3, which represents three units in the treated population.

The expected weight for observations in the control group is given by

$$\bar{w} = \frac{N_t}{N_c} = \frac{p_t}{1 - p_t}$$

where $p_t = N_t/(N_t + N_c)$ is the proportion of individuals in the treated group.

The PLOTS=WGTCLOUD option in the ASSESS statement requests cloud plots for weights. The plot displays a reference line at $r\,p_t/(1 - p_t)$ for observations in the control group, where *r*=10 by default. You can specify a different value for *r* with the PLOTS=WGTCLOUD(REF=*r*) option.

You can request ATT weighting by specifying the WEIGHT=ATTWGT option in the PSWEIGHT statement, and then by specifying the WEIGHT= option in the OUTPUT statement to create a variable that contains these weights.

Large Propensity Score Weights

For IPTW-ATE weighting, if a treated unit has a propensity score close to 0 or a control unit has a propensity score close to 1, the resulting weight can be large. Similarly, for ATT weighting, if a control unit has a propensity score close to 1, the resulting weight can also be large. If a few observations have very large weights, the resulting IPTW-ATE or ATT estimator has a large variance.

You can use the NLARGESTWGT=*n* option to request a table that displays the *n* largest IPTW-ATE or ATT weights in the treated and control groups. You can exclude observations that have extreme weights in the outcome analysis, and the inference is for the resulting subset of observations. You can examine the observations that have extreme weights, find the covariate values that are associated with these extreme weights, and exclude these observations based on covariate values for a more robust interpretation.

You can also specify the PSMIN= and PSMAX= suboptions in the REGION=ALLOBS option in the PROC PSMATCH statement and the OUT(OBS=REGION) option in the OUTPUT statement to exclude observations that have extreme weights from the output data set.

Propensity Score Stratification

Propensity stratification divides the observations into strata that have similar propensity scores, with the objective of balancing the observed variables between treated and control units within each stratum. The treatment effect can then be estimated by combining stratum-specific estimates of treatment effect. Rosenbaum and Rubin (1984, p. 521) show that an adjusted estimate of this type that is based on five strata can remove approximately 90% of the bias in the crude or unadjusted estimate.

The PSMATCH procedure performs stratification when you specify the STRATA statement, which divides the observations contained in the support region into strata (you specify the support region in the REGION= option in the PROC PSMATCH statement).

In general, when observations are stratified, it is common to require the same number of observations in each stratum. However, in the context of propensity score analysis, the number of units in the control group tends to be much larger than the number of units in the treated group. Consequently, this requirement can produce strata for which the number of units in the treated group is insufficient to compute reliable stratum-specific estimates of the treatment effect.

The KEY=TREATED option (which is the default) in the STRATA statement avoids this problem by allocating approximately the same number of treated units to each stratum. Alternatively, you can specify the KEY=TOTAL option to allocate approximately the same number of observations (for either treated or control units) to each stratum. Regardless of the method of allocation, you should ensure that the number of treated units and the number of control units in each stratum are sufficient to estimate the treatment effect.

To assess the variable balance after stratification, you can use the STRATUMWGT= option in the STRATA statement to specify the stratum weights, compute the weighted averages of stratum-specific variable averages in the treated group and in the control group, and then compare the resulting weighted averages between the treated and control groups.

In the outcome analysis, you can use the weighted average of the stratum-specific treatment estimates to estimate the treatment effect. You can estimate the ATT if you weight by the stratum-specific number of treated units, and you can estimate the ATE if you weight by the stratum-specific number of units (treated and control units combined) (Stuart 2010, p. 13; Guo and Fraser 2015, pp. 76–77).

The STRATUMWGT=TOTAL option uses the proportional size of the stratum as the stratum weight. The proportional size is the number of total units (treated and control) in the stratum divided by the total number of units. Stratum weights of this type are appropriate for estimating the ATE. The STRATUMWGT=TREATED option uses the proportional number of treated units as the stratum weight. This number is the number of treated units in the stratum divided by the total number of treated units. Stratum weights of this type are appropriate for estimating the ATT. The following section provides more details about weighting after stratification.

Weighting after Stratification

The STRATA statement creates strata of observations that have similar propensity scores. The NSTRATA= option specifies the number of strata. The KEY=TOTAL option allocates approximately the same number of total units to each stratum, and the KEY=TREATED option allocates the same number of treated units to each stratum.

After stratification, you can use the weighted average of the stratum-specific treatment estimates to estimate the treatment effect in the outcome analysis. The particular weights that you use depend on the estimator (ATE or ATT). Two commonly used stratum weights are weighting by the total units and weighting by the treated units.

The PSMATCH procedure provides the following stratum weights to assess the variable balance after stratification:

- STRATUMWGT=TOTAL (weighting by the number of total units in stratum), which is used to estimate the ATE
- STRATUMWGT=TREATED (weighting by the number of treated units in stratum), which is used to estimate the ATT

Thus, a stratum weight is computed as

$$w_g = \frac{w_{1g}}{\sum_g w_{1g}}$$

where g is the stratum index and

$$w_{1g} = \begin{cases} N_{t(g)} + N_{c(g)} & \text{if STRATUMWGT=TOTAL} \\ N_{t(g)} & \text{if STRATUMWGT=TREATED} \end{cases}$$

where $N_{t(g)}$ is the number of treated units and $N_{c(g)}$ is the number of control units in the gth stratum.

Let $\bar{x}_{\mathrm{t(g)}}$ be the mean of treated units in the gth stratum and $\bar{x}_{\mathrm{c(g)}}$ be the mean of control units in the gth stratum, with corresponding sample variances $V(x_{\mathrm{t(g)}})$ and $V(x_{\mathrm{c(g)}})$. Then the weighted stratum means for the treated and control groups are

$$\bar{x}_{\mathrm{t(S)}} = \sum_g w_g \bar{x}_{\mathrm{t(g)}}$$

$$\bar{x}_{\mathrm{c(S)}} = \sum_g w_g \bar{x}_{\mathrm{c(g)}}$$

The variances of the weighted stratum means $\bar{x}_{\mathrm{t(S)}}$ and $\bar{x}_{\mathrm{c(S)}}$ are then given by

$$V(x_{\mathrm{t(S)}}) = \sum_g w_g^2 V(x_{\mathrm{t(g)}})$$

$$V(x_{\mathrm{c(S)}}) = \sum_g w_g^2 V(x_{\mathrm{c(g)}})$$

These stratum weights are displayed in the “Standardized Mean Differences within Strata” table (see Output 101.2.7). You can use these weights for estimation of the treatment effect in an outcome analysis.

Matching Process

Except for matching with replacement in which multiple control units are matched to each treated unit, propensity score matching creates mutually exclusive sets of observations that have similar propensity scores. Each set has at least one treated unit and at least one control unit. The distribution of observed variables will be similar between treated units and control units in the matched sample.

For propensity score matching, Stuart (2010) reviews matching methods and provides guidance on their use. Austin (2014) provides a detailed comparison of algorithms for matching.

The flowchart in Figure 101.14 summarizes the steps in propensity score matching.

Figure 101.14 Steps in Propensity Score Matching

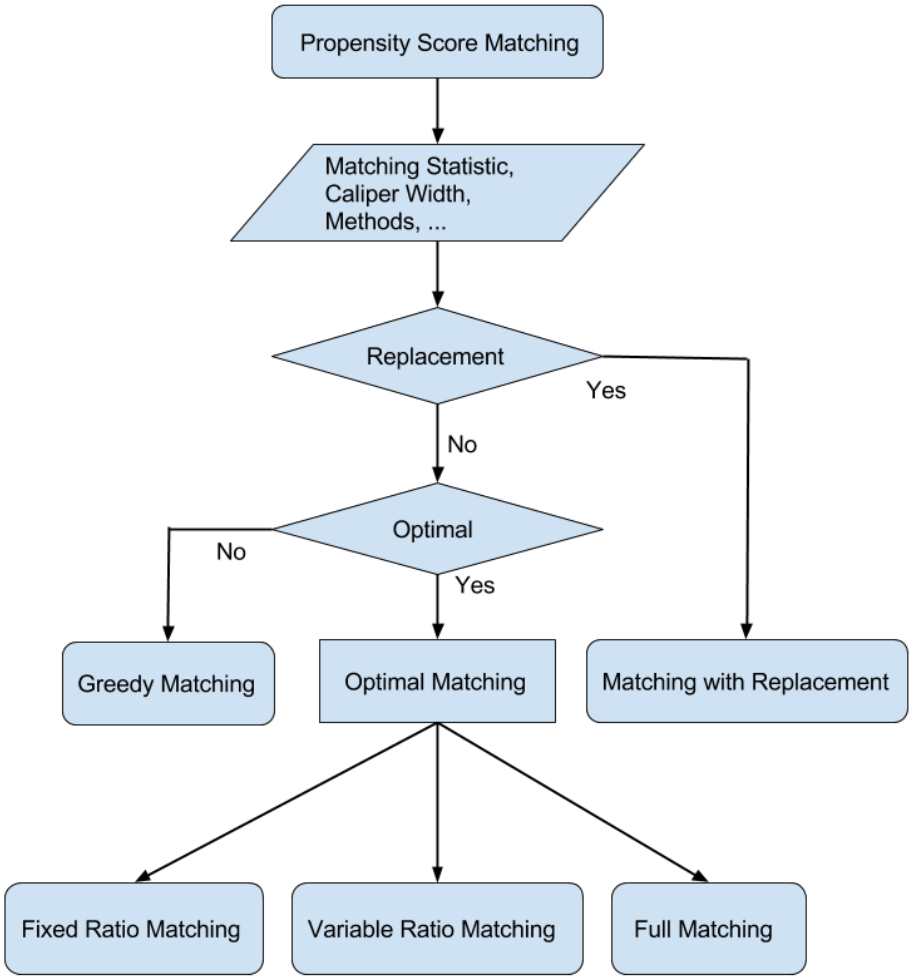

The PSMATCH procedure provides the following strategies for matching observations in the control group to observations in the treatment group:

- greedy nearest neighbor matching, which sequentially and without replacement selects the control unit

whose propensity score is closest to that of the particular treated unit

- optimal matching, which selects all matches simultaneously and without replacement to minimize the total absolute difference in propensity score across all matches (this approach includes fixed ratio matching, variable ratio matching, and full matching)
- matching with replacement, which selects with replacement the control unit whose propensity score is closest to that of each treated unit

In addition to the propensity score, you can also use the logit of the propensity score and Mahalanobis distance as the matching metric that is used to compare the closeness of two units. For more information, see the section "Matching Methods" on page 8414.

You can use the CALIPER= option in the MATCH statement to request that the difference in the propensity scores for a matched pair be less than or equal to a specified caliper width.

You can request exact matches of the levels of classification variables for treated and control units by specifying the EXACT= option in the MATCH statement.

Matching Metrics

The PSMATCH procedure provides the following metrics or distances for the purpose of matching observations in the treated group with observations in the control group:

- the difference in the logit of the propensity score (DISTANCE=LPS; this is the default)
- the difference in the propensity score (DISTANCE=PS)
- the Mahalanobis distance between sets of continuous variables (DISTANCE=MAH)

You specify the type of distance in the DISTANCE= option in the MATCH statement. Let p_{ti} and p_{cj} be the propensity scores of the *i*th treated unit and the *j*th control unit, respectively. When you specify DISTANCE=PS, matching is based on the absolute difference of propensity scores:

$$\left|p_{ti} - p_{cj}\right|$$

When you specify DISTANCE=LPS, matching is based on the absolute difference of logits of propensity scores:

$$\left|\mathrm{logit}(p_{ti}) - \mathrm{logit}(p_{cj})\right|$$

When you specify DISTANCE=MAH, two different versions of the Mahalanobis distance (d) can be computed, as specified in the SQRT= suboption of the DISTANCE=MAH option,

$$d = \begin{cases} \sqrt{(\mathbf{X}_{ti} - \mathbf{X}_{cj})'\ \mathbf{V}^{-1}\ (\mathbf{X}_{ti} - \mathbf{X}_{cj})} & \text{if SQRT=YES (this is the default)} \\ (\mathbf{X}_{ti} - \mathbf{X}_{cj})'\ \mathbf{V}^{-1}\ (\mathbf{X}_{ti} - \mathbf{X}_{cj}) & \text{if SQRT=NO} \end{cases}$$

where d is the Mahalanobis distance; $\mathbf{X}$ is the set of variables that include the logit of the propensity score if LPS is specified, the propensity score if PS is specified, and all continuous variables that are specified in the VAR= options; $\mathbf{X}_{ti}$ contains variable values of the ith treated unit; $\mathbf{X}_{cj}$ contains variable values of the jth control unit; and $\mathbf{V}$ is the covariance matrix of $\mathbf{X}$.

The SQRT= option does not affect the results for greedy nearest neighbor matching and matching with replacement; it affects only the results for optimal matching.

Three different covariance matrices can be used to compute the Mahalanobis distance (as specified in the COV= suboption of the DISTANCE=MAH option):

- the covariance matrix that is based on observations in the control group (COV=CONTROL; this is the default)
- the pooled covariance matrix that is based on observations in the treated and control groups (COV=POOLED)
- the identity matrix (COV=IDENTITY), which yields the Euclidean distance.

You can include the propensity score and the logit of the propensity among the variables that are used to compute the Mahalanobis distance. For example, when you specify DISTANCE=MAH(PS VAR=(X1 X2 X3) / COV=POOLED), the PSMATCH procedure computes the Mahalanobis distance between observations in the treated and control groups by using the propensity score and variables X1, X2, and X3. The covariance matrix is the pooled covariance matrix of the treated and control groups.

Matching Methods

When you specify the MATCH statement, the PSMATCH procedure matches observations in the control group to observations in the treatment group by using one of the methods that are described in the following subsections. You can request the method in the METHOD= option.

Greedy Nearest Neighbor Matching

Greedy nearest neighbor matching, requested by the METHOD=GREEDY option, selects the control unit whose propensity score best matches the propensity score of each treated unit. Greedy nearest neighbor matching is done sequentially and without replacement.

The following criteria are available for greedy nearest neighbor matching:

- the number of control units to be matched to each treated unit (you can specify this number in the K= suboption)
- the order of propensity scores of treated units, which can be ascending, descending, or random (you can specify the order in the ORDER= suboption)

Replacement Matching

Replacement matching, requested by the METHOD=REPLACE option, selects with replacement the control unit whose propensity score is closest to the propensity score for each treated unit. You can specify the number of control units to be matched to each treated unit in the K= suboption.

Optimal Matching

Optimal matching selects all matches simultaneously and without replacement to minimize the total absolute difference in propensity score across all matches. You can request the following optimal matching methods:

- fixed ratio matching, requested by the METHOD=OPTIMAL option, which matches a fixed number of control units to each treated unit
- variable ratio matching, requested by the METHOD=VARRATIO option, which matches one or more control units to each treated unit
- full matching, requested by the METHOD=FULL option, which matches each treated unit to one or more control units or matches each control unit to one or more treated units. By additionally specifying the KMEAN=, NCONTROL=, or PCTCONTROL= suboptions, you can request constrained full matching in which the number of matched control units is less than the total number of available controls.

As alternatives to matching on the propensity score, you can match on the logit of the propensity score or use the Mahalanobis distance to match on a set of variables (possibly including the PS or the LPS). All three of these methods minimize the total absolute difference across all matches in the matching metric, which is the total difference in the logit of the propensity score by default.

Table 101.8 lists the suboptions available for optimal matching. The symbol X indicates that the option applies for the specified method.

Table 101.8 Applicable Options for Optimal Matching

METHOD=	K=	KMIN=	KMAX=	KMAXTRT=	KMEAN= NCONTROL= PCTCONTROL=
OPTIMAL	X				
VARRATIO		X	X		X
FULL			X	X	X

- K= specifies the number of control units to be matched to each treated unit.
- KMIN= specifies the minimum number of control units to be matched to each treated unit.
- KMAX= specifies the maximum number of control units to be matched to each treated unit.
- KMAXTRT= specifies the maximum number of treated units to be matched to each matched control unit.

- KMEAN= specifies the average number of control units to be matched to each treated unit.
- NCONTROL= specifies the total number of control units to be matched.
- PCTCONTROL= specifies the percentage of control units to be matched.

You can specify only one of the KMEAN=, NCONTROL=, and PCTCONTROL= options.

Weighting after Matching

If the matched observations show good variable balance after matching, you can perform an outcome analysis to estimate the treatment effect by comparing outcomes between treated and control subjects in the matched sample. Except for the case of one-to-one matching without replacement, the matched observation weights should be used in the balance assessment and in the outcome analysis.

The PSMATCH procedure provides the following methods for weighting after matching:

- ATT weighting, which is used to estimate the ATT
- ATE weighting after full matching, which is used to estimate the ATE

ATT Weighting after Matching without Replacement

ATT weights for use after matching without replacement are computed as

$$w_{gj} = \begin{cases} 1 & \text{for treated units in the } g\text{th matched set} \\ \frac{N_{gt}}{N_{gc}} & \text{for control units in the } g\text{th matched set} \end{cases}$$

where N_{gt} is the number of treated units and N_{gc} is the number of control units in the gth matched set.

That is, in each matched set, each treated unit has a weight of 1 and each control unit has a weight that equals the number of treated units in the matched set divided by the number of control units in the matched set. Thus, with a one-to-one greedy or optimal matching, the weight is 1 for both the treated and control units. Under a different matching algorithm, if the gth matched set contains N_{gt}=1 treated unit and N_{gc}=3 control units, then the weight for each treated unit is 1 and the weight for each control unit is 1/3.

The total weight for the controls is equal to the total number of treated units in each matched group, and the total weight for the matched controls is equal to the total number of matched treated units.

You can request ATT weighting after matching by specifying the MATCHATTWGT option in the MATCH statement, and then by specifying the WEIGHT= option in the OUTPUT statement to create a variable that contains these weights.

ATT Weighting after Matching with Replacement

The PSMATCH procedure creates mutually exclusive sets of units after matching with replacement. In the matched set, each treated unit is connected to all control units either directly or indirectly. For example, assume that the treated group contains units $T_1, T_2, \ldots$, and the control group contains units $C_1, C_2, \ldots$. If T_1 is matched to C_1 and C_2 and T_2 is matched to C_2 and C_3, then T_1 is connected directly to C_1 and C_2 and is connected indirectly to C_3, Similarly, T_2 is connected directly to C_2 and C_3, and indirectly to C_1.

In each matched set, each treated unit has a weight of 1 and each control unit has a weight that is computed from contributions of its matched treated units. That is, if a treated unit has two matched control units, then each control unit has a weight of 1/2 from this treated unit.

For example, assume that T_1 is matched to C_1 and C_2 and T_2 is matched to C_2 and C_3, and the five units do not have other matches. Then C_1 has a weight of 1/2 (from T_1), C_2 has a weight of 1 (1/2 from T_1 and 1/2 from T_2), and C_3 has a weight of 1/2 (from T_2).

The total weight for the controls is equal to the total number of treated units in each matched group, and the total weight for the matched controls is equal to the total number of matched treated units.

You can request ATT weighting after matching by specifying the MATCHATTWGT option in the MATCH statement, and then by specifying the WEIGHT= option in the OUTPUT statement to create a variable that contains these weights.

ATE Weighting after Full Matching

When optimal full matching is done, ATE weights for use after matching are computed as

$$w_{gj} = \begin{cases} \frac{N_g}{N_{gt}} & \text{for treated units in the } g\text{th matched set} \\ \frac{N_g}{N_{gc}} & \text{for control units in the } g\text{th matched set} \end{cases}$$

where N_{gt} is the number of treated units, N_{gc} is the number of control units, and $N_g = N_{gt} + N_{gc}$ is the total number of units in the gth matched set.

That is, in each matched set, each treated unit has a weight that equals the total number of treated and control units divided by the number of treated units, and each control unit has a weight that equals the total number of treated and control units divided by the number of control units in the matched set. Thus, if a matched set contains N_{gt}=1 treated unit and N_{gc}=3 control units, then the treated unit has a weight of 4 and each control unit has a weight of 4/3; if a matched set contains N_{gt}=2 treated units and N_{gc}=1 control unit, then each treated unit has a weight of 3/2 and the control unit has a weight of 3.

The total weight for the treated units and the total weight for the control units are each equal to the combined number of treated and control units in each matched group. Thus, the total weight for matched treated units and the total weight for matched control units are each equal to the total number of matched units (treated and control units combined).

ATE weighting is available only for full matching (METHOD=FULL) and is appropriate only for unrestricted full matching (that is, when all available control units are matched).

You can request ATT weighting after matching by specifying the MATCHATEWGT option in the MATCH statement, and then by specifying the WEIGHT= option in the OUTPUT statement to create a variable that contains these weights.

Variable Balance Assessment

Propensity score analysis assumes that the true propensity scores are known. When the propensity scores are estimated—as is usually the case in practice—you need to assess how well the distributions of the propensity scores (or their logits) and the adjusted variables are balanced between the treatment group and the control group. PROC PSMATCH provides numerical and graphical tools that you can use to assess the balance of the propensity scores (or their logits) and the adjusted variables that are either continuous or binary.

The ASSESS statement in the PSMATCH procedure provides a variety of statistical measures and graphical displays for comparing these distributions. You can make these assessments for all the observations in the data set, the observations in the support region, or the matched observations (if you specify a MATCH statement).

Two statistical measures for balance assessment are the standardized mean difference between the treatment and control groups and the treated-to-control variance ratio. For good variable balance, the absolute standardized mean difference should be less than or equal to 0.25, and the variance ratio should be between 0.5 and 2 (Rubin 2001, p. 174; Stuart 2010, p. 11). Some authors have applied a smaller threshold of 0.1 to the absolute standardized mean difference (Normand et al. 2001; Mamdani et al. 2005; Austin 2009).

The standardized mean difference is computed by dividing the difference in the means of the variable in the two groups by an estimate of the standard deviation. Two estimates of the standard deviation are available in the PSMATCH procedure:

- the square root of the average of the variances in the treatment and control groups (Rosenbaum and Rubin 1985, p. 37),
- the standard deviation of observations in the treatment group only (Stuart 2010, p. 11)

For binary classification variables, the mean is taken to be the proportion p of units having the first classification level, and the variance is computed as $p(1-p)$ (Austin, Grootendorst, and Anderson 2007, p. 737).

If you specify a STRATA statement, then stratum-specific standardized mean differences are computed for observations in the support region.

The PSMATCH procedure displays the standardized mean differences in plots. You can also request box plots and cloud plots for continuous variables, and bar charts for binary classification variables. These plots are also produced for each stratum if you specify a STRATA statement.

The next three subsections describe how standardized mean differences and treated-to-control variance ratios are computed for all observations, observations in the support region, and matched observations.

Standardized Mean Differences for All Observations

For all observations in the data set, let $\bar{x}_{\mathrm{t(A)}}$ be the mean of the observations in the treatment group and let $\bar{x}_{\mathrm{c(A)}}$ be the mean of the observations in the control group, with corresponding sample variances $V(x_{\mathrm{t(A)}})$ and $V(x_{\mathrm{c(A)}})$. Then the standardized mean difference is

$$d_{\mathrm{(A)}} = \frac{\bar{x}_{\mathrm{t(A)}} - \bar{x}_{\mathrm{c(A)}}}{s_{\mathrm{(A)}}}$$

where the standard deviation is given by

$$s_{(\mathrm{A})} = \begin{cases} \sqrt{\frac{V(x_{\mathrm{t(A)}})+V(x_{\mathrm{c(A)}})}{2}} & \text{if STDDEV=POOLED} \\ \sqrt{V(x_{\mathrm{t(A)}})} & \text{if STDDEV=TREATED} \end{cases}$$

The treated-to-control variance ratio is

$$\frac{V(x_{\mathrm{t(A)}})}{V(x_{\mathrm{c(A)}})}$$

Standardized Mean Differences for Observations in the Support Region

For observations in the support region, let $\bar{x}_{\mathrm{t(R)}}$ be the mean of observations in the treatment group and $\bar{x}_{\mathrm{c(R)}}$ be the mean of observations in the control group, with corresponding sample variances $V(x_{\mathrm{t(R)}})$ and $V(x_{\mathrm{c(R)}})$. Then the standardized mean difference is

$$d_{(\mathrm{R})} = \begin{cases} \frac{\bar{x}_{\mathrm{t(R)}}-\bar{x}_{\mathrm{c(R)}}}{s_{(\mathrm{A})}} & \text{if STDDEV=POOLED(ALLOBS=YES)} \\ & \text{or STDDEV=TREATED(ALLOBS=YES)} \\ \frac{\bar{x}_{\mathrm{t(R)}}-\bar{x}_{\mathrm{c(R)}}}{s_{(\mathrm{R})}} & \text{if STDDEV=POOLED(ALLOBS=NO)} \\ & \text{or STDDEV=TREATED(ALLOBS=NO)} \end{cases}$$

where the standard deviation $s_{(\mathrm{R})}$ is given by

$$s_{(\mathrm{R})} = \begin{cases} \sqrt{\frac{V(x_{\mathrm{t(R)}})+V(x_{\mathrm{c(R)}})}{2}} & \text{if STDDEV=POOLED} \\ \sqrt{V(x_{\mathrm{t(R)}})} & \text{if STDDEV=TREATED} \end{cases}$$

That is, with ALLOBS=YES, the standard deviation that is derived from all observations in the data set is used to compute the standardized mean difference. With ALLOBS=NO, the standard deviation that is derived from observations in the support region is used to compute the standardized mean difference.

The treated-to-control variance ratio is

$$\frac{V(x_{\mathrm{t(R)}})}{V(x_{\mathrm{c(R)}})}$$

The percentage reduction in the standardized mean difference is computed as

$$100 \times \frac{\max(|d_{(\mathrm{A})}| - |d_{(\mathrm{R})}|, 0)}{|d_{(\mathrm{A})}|}$$

Pooled Standardized Mean Differences across the Strata

Let $\bar{x}_{\mathrm{t(S)}}$ be the weighted stratum mean of treated observations, and let $\bar{x}_{\mathrm{c(S)}}$ be the weighted stratum mean of control observations, with corresponding variances $V(x_{\mathrm{t(S)}})$ and $V(x_{\mathrm{c(S)}})$. For information about these statistics, see the section "Weighting after Stratification" on page 8411.

The standardized mean difference is

$$d_{(\mathrm{S})} = \begin{cases} \frac{\bar{x}_{\mathrm{t(S)}} - \bar{x}_{\mathrm{c(S)}}}{s_{(\mathrm{A})}} & \text{if STDDEV=POOLED(ALLOBS=YES)} \\ & \text{or STDDEV=TREATED(ALLOBS=YES)} \\ \frac{\bar{x}_{\mathrm{t(S)}} - \bar{x}_{\mathrm{c(S)}}}{s_{(\mathrm{S})}} & \text{if STDDEV=POOLED(ALLOBS=NO)} \\ & \text{or STDDEV=TREATED(ALLOBS=NO)} \end{cases}$$

where the standard deviation $s_{(\mathrm{S})}$ is given by

$$s_{(\mathrm{S})} = \begin{cases} \sqrt{\frac{V(x_{\mathrm{t(S)}}) + V(x_{\mathrm{c(S)}})}{2}} & \text{if STDDEV=POOLED} \\ \sqrt{V(x_{\mathrm{t(S)}})} & \text{if STDDEV=TREATED} \end{cases}$$

The treated-to-control variance ratio is

$$\frac{V(x_{\mathrm{t(S)}})}{V(x_{\mathrm{c(S)}})}$$

The percentage reduction for the standardized mean difference is computed as

$$100 \times \frac{\max(|d_{(\mathrm{A})}| - |d_{(\mathrm{S})}|, 0)}{|d_{(\mathrm{A})}|}$$

Standardized Mean Differences for Matched Observations

Let $\bar{x}_{\mathrm{t(M)}}$ be the mean of matched observations in the treatment group, and let $\bar{x}_{\mathrm{c(M)}}$ be the mean of matched observations in the control group, with corresponding sample variances $V(x_{\mathrm{t(M)}})$ and $V(x_{\mathrm{c(M)}})$. Then the standardized mean difference is

$$d_{(\mathrm{M})} = \begin{cases} \frac{\bar{x}_{\mathrm{t(M)}} - \bar{x}_{\mathrm{c(M)}}}{s_{(\mathrm{A})}} & \text{if STDDEV=POOLED(ALLOBS=YES)} \\ & \text{or STDDEV=TREATED(ALLOBS=YES)} \\ \frac{\bar{x}_{\mathrm{t(M)}} - \bar{x}_{\mathrm{c(M)}}}{s_{(\mathrm{M})}} & \text{if STDDEV=POOLED(ALLOBS=NO)} \\ & \text{or STDDEV=TREATED(ALLOBS=NO)} \end{cases}$$

where the standard deviation $s_{(\mathrm{M})}$ is given by

$$s_{(\mathrm{M})} = \begin{cases} \sqrt{\frac{V(x_{\mathrm{t(M)}}) + V(x_{\mathrm{c(M)}})}{2}} & \text{if STDDEV=POOLED} \\ \sqrt{V(x_{\mathrm{t(M)}})} & \text{if STDDEV=TREATED} \end{cases}$$

The treated-to-control variance ratio is

$$\frac{V(x_{\mathrm{t(M)}})}{V(x_{\mathrm{c(M)}})}$$

The percentage reduction for the standardized mean difference is computed as

$$100 \times \frac{\max(|d_{(\mathrm{A})}| - |d_{(\mathrm{M})}|, 0)}{|d_{(\mathrm{A})}|}$$

Sensitivity Analysis

Propensity score analysis assumes that all the confounders (variables that affect both the outcome and the treatment assignment) have been measured. If some confounders are unobserved, individuals that have the same observed covariates might not have the same probability of being assigned to the treated group. The assumption of no unmeasured confounders cannot be verified, so you should analyze the sensitivity of inferences to departures from this assumption. Sensitivity analysis considers how strong the unobserved covariates would have to be in order to negate the conclusion of the study (assuming that the initial analysis found a significant effect of the treatment).

Liu, Kuramoto, and Stuart (2013) describe seven commonly used techniques for sensitivity analysis. Based on the study objectives, these methods are classified into two groups. One group finds the tipping point that negates the statistical significance of the outcome-treatment association (Liu, Kuramoto, and Stuart 2013). The other group (not discussed here) derives the point estimate of the true outcome-treatment association with a 95% confidence interval (Liu, Kuramoto, and Stuart 2013).

Rosenbaum (2010, p. 77) provides a sensitivity analysis based on the odds ratio,

$$\frac{\pi_k/(1-\pi_k)}{\pi_l/(1-\pi_l)}$$

where π_k and π_l are the probabilities that the kth and lth individuals are assigned to the treated group, given that they have the same observed covariates, $x_k = x_l$.

For all kth and lth individuals with $x_k = x_l$, assume that the odds ratio is bounded by

$$\frac{1}{\Gamma} \leq \frac{\pi_k/(1-\pi_k)}{\pi_l/(1-\pi_l)} \leq \Gamma$$

where $\Gamma \geq 1$.

The parameter Γ measures the degree of hidden bias from unobserved confounders. For example, with $\Gamma = 2$,

$$\frac{\pi_k/(1-\pi_k)}{\pi_l/(1-\pi_l)} = 2$$

which indicates that even though they have the same values of the observed covariates, the kth individual is twice as likely as the lth individual to be in the treated group because of hidden bias.

Propensity score analysis assumes that if the kth and lth individuals have the same observed covariates, then $\pi_k = \pi_l$ and $\Gamma = 1$. When an outcome analysis leads to a significant result, Rosenbaum's sensitivity

analysis finds a tipping point, $\Gamma = \gamma$, that negates the conclusion of the study. A large value of γ is evidence that only a large departure from random treatment assignment can negate the conclusion of the study. If $\Gamma = \gamma$ is plausible, the study conclusion is not robust to hidden bias from an unobserved confounder.

For the case of one-to-one matched observations, Rosenbaum (2010, pp. 78–84) provides a sensitivity analysis that is based on paired observations. The following subsection describes this approach.

Sensitivity Analysis on Matched Observations

In a set of one-to-one matched observations, if individuals k and l are in the same matched set, then the probability that individual k is in the treated group and individual l is in the control group is

$$\frac{\pi_k}{\pi_k + \pi_l}$$

and the following equation can be used for sensitivity analysis:

$$\frac{1}{1+\Gamma} \leq \frac{\pi_k}{\pi_k + \pi_l} \leq \frac{\Gamma}{1+\Gamma}$$

If $\Gamma = 1$, then $\pi_k = \pi_l$.

For example, let y_{jt} and y_{jc} be the responses for the treated and control units in the jth matched set. The response is the improvement after treatment, and a positive value indicates a beneficial effect. Let

$$d_j = y_{jt} - y_{jc}$$

be the difference in responses between the treated and control units.

Suppose that a signed rank test is used in the outcome analysis. The signed rank statistic is

$$S = \sum_{j:d_j>0} d_j^+$$

where d_j^+ is the rank of $|d_j|$.

Assume that Γ=1. Then under the hypothesis of no treatment effect, S has mean

$$\mu_0 = \frac{n_t(n_t+1)}{4}$$

where n_t is the number of matched sets and the variance V (assuming that all d_j is distinct) is

$$V_0 = \frac{n_t(n_t+1)(2n_t+1)}{24}$$

When $n_t > 20$, the significance of

$$\frac{S - \mu_0}{\sqrt{V_0}}$$

can be computed from the Student's t distribution with $n_t - 1$ degrees of freedom.

For $\Gamma = \gamma$, S has mean

$$\mu_1 = \frac{\gamma}{1+\gamma}\frac{n_t(n_t+1)}{2} = \frac{2\gamma}{1+\gamma}\mu_0$$

and variance

$$V_1 = \frac{\gamma}{(1+\gamma)^2}\frac{n_t(n_t+1)(2n_t+1)}{6} = \frac{4\gamma}{(1+\gamma)^2}V_0$$

If a signed rank test shows a significantly better benefit in the treated group with $\Gamma = 1$, the sensitivity analysis searches for a tipping point that negates the study conclusion. A study conclusion is robust to hidden bias from the unobserved confounder if an extreme value of Γ is needed to alter the study conclusion.

Example 101.9 illustrates a sensitivity analysis on a set of one-to-one matched observations.

Entropy Balancing

In addition to performing propensity score analyses, the PSMATCH procedure also performs entropy balancing (Hainmueller 2012). Entropy balancing is a method of reweighting observations that does not require propensity scores and instead directly targets balancing the moments of covariates between the treated and control conditions. The procedure obtains the entropy balancing weights by solving a constrained optimization problem. The optimization problem is defined by a set of reference weights, constraints on the weighted moments for a set of covariates, and a measure of divergence.

To describe the entropy balancing weights and how they are computed, the following definitions are used:

- t denotes the level of the treatment T for which entropy balancing weights are to be computed.
- n_t denotes the number of observations in which $T = t$.
- $\boldsymbol{q}$ denotes a set of positive reference weights q_i, $i = 1, \ldots, n_t$, that have been normalized to satisfy $\sum_{i=1}^{n_t} q_i = 1$.
- $\boldsymbol{x}$ denotes a vector of covariates.

As described in Hainmueller (2012), the entropy balance optimization problem imposes the constraints $c_r(\boldsymbol{x})$, $r = 1, \ldots, R$, on the weighted moments of the covariates. In particular, the PSMATCH procedure supports constraints on the first and possibly second moments of continuous variables. For categorical variables, the procedure uses the GLM parameterization and imposes balance constraints on the level proportions. To define the constraints for a continuous variable x_j, let μ_j be an estimate of the variable mean, let m_r denote the target value of the weighted central moment, and assume that the weights w_i are nonnegative and sum to 1. A constraint on the first moment of x_j is then of the form

$$c_r(\boldsymbol{x}) = \sum_{i=1}^{n_t} c_{ri}(\boldsymbol{x}) = \sum_{i=1}^{n_t} w_i \left(x_{ij} - \mu_j\right) - m_r = 0$$

where m_r is equal to 0. A constraint on the second moment of x_j is of the form

$$c_r(\boldsymbol{x}) = \sum_{i=1}^{n_t} c_{ri}(\boldsymbol{x}) = \sum_{i=1}^{n_t} w_i \left(x_{ij} - \mu_j\right)^2 - m_r = 0$$

For a categorical predictor, let x_l denote a column that is created for a level of the variable by using the GLM parameterization, and let m_r denote the target proportion for that level. A constraint on the level proportion is of the form

$$c_r(\boldsymbol{x}) = \sum_{i=1}^{n_t} c_{ri}(\boldsymbol{x}) = \sum_{i=1}^{n_t} w_i x_l - m_r = 0$$

The objective function that is used for the entropy balance optimization problem is

$$d\,(\boldsymbol{w}) = \sum_{i=1}^{n_t} w_i \log\left(\frac{w_i}{q_i}\right)$$

where w_i, $i = 1, \ldots, n_t$, are nonnegative weights that are additionally constrained to sum to 1. This objective function corresponds to the Kullback-Leibler divergence, or relative entropy, between probability the distributions that are defined by the reference weights $\boldsymbol{q}$ and the weights $\boldsymbol{w}$. The entropy balance weights for the treatment group in which $T = t$ are determined by the optimal solution $\boldsymbol{w}^*$ to the optimization problem

$$\begin{aligned}
&\min_{\boldsymbol{w}} d(\boldsymbol{w}) \\
&c_r(\boldsymbol{x}) = 0 \qquad \text{for } r = 1, \ldots, R \\
&w_i \geq 0 \qquad \text{for } i = 1, \ldots, n_t \\
&\sum_{i=1}^{n_t} w_i = 1
\end{aligned}$$

As described in Hainmueller (2012), by introducing Lagrange multipliers for the constraints, you can obtain the entropy balance weights by solving a more tractable dual formulation of the problem. In particular, let $\boldsymbol{\lambda}$ denote a vector of dimension R, and consider the unconstrained dual problem of

$$\min_{\boldsymbol{\lambda}} \log\left(\sum_{i=1}^{n_t} q_i \exp\left(-\sum_{r=1}^{R} \lambda_r c_{ri}(\boldsymbol{x})\right)\right) + \sum_{r=1}^{R} \lambda_r m_r$$

Note that the maximum absolute gradient element of the dual problem corresponds to the maximum absolute violation of the balance constraints for the primal problem.

Given the solution $\boldsymbol{\lambda}^*$ to the dual problem, you can obtain the entropy balance weights by evaluating the expression

$$w_i^* = \frac{q_i \exp\left(-\sum_{r=1}^{R} \lambda_r^* c_{ri}(\boldsymbol{x})\right)}{\sum_{i=1}^{n_t} q_i \exp\left(-\sum_{r=1}^{R} \lambda_r^* c_{ri}(\boldsymbol{x})\right)}$$

You can rescale the weights $\boldsymbol{w}^*$ to sum to any positive target value.

The PSMATCH procedure formulates and solves the entropy balance optimization problem as follows. When WEIGHT=CONTROL, one optimization problem is solved to obtain the entropy balance weights for the control units, and the values of μ_j and the target moments m_r are determined by the central moments of the

treated units. When WEIGHT=CONTROL, the units in the treatment are either assigned reference weight values that are determined by the REFWEIGHT= option or set to 1 if no reference weights are specified. When WEIGHT=ALLOBS, two optimization problems are solved—one for the control units and a second one for the treated units—and the values of μ_j and the target moments m_r are determined by the central moments of all observations.

By default, the reference weights $\boldsymbol{q}$ are from a uniform distribution where $q_i = \frac{1}{n_t}$. Alternatively, you can use the REFWEIGHT= option to specify a variable in the input data set that contains the reference weights. Note that the PSMATCH procedure internally rescales the input reference weights so that they sum to 1. When WEIGHT=CONTROL, the unscaled values are used as weights for the treated units in the balance diagnostics and OUT= data set. When you specify reference weights, you can use the WEIGHTMOMENTS= suboption to specify whether or not the target moments for the entropy balance constraints should be weighted by the reference weights. By default, WEIGHTMOMENTS=YES.

Balance constraints are imposed on the variables that you specify in the BALANCEVARS= option. By default, balance constraints are added for both the first and second moments of continuous variables. You can use the MOMENTS=1 suboption to add balance constraints only for the first moments of continuous variables. If PROC PSMATCH detects linearly dependent balance constraints, a warning message is printed to the log. The procedure attempts to solve the entropy balance problem for the linearly independent constraints, and it returns an error if the resulting weights do not also satisfy the linearly dependent constraints.

PROC PSMATCH attempts to solve the dual formulation of the entropy balancing problem by using a Newton-Raphson optimization that combines a line-search algorithm with ridging. The procedure evaluates an absolute gradient convergence criterion for the dual problem that you can specify by using the ABSGTOL= option. In addition to the absolute gradient convergence criterion, you can specify a second, weaker value for evaluating the balance constraints by using the BALTOL= option. If the entropy balance optimization problem fails to converge, the weights from the final iteration are accepted if the maximum absolute violation of the balance constraints is less than or equal to the balance constraint tolerance value. By default, the values of the ABSGTOL= and BALTOL= options are the same.

When a set of entropy balance weights are obtained for a treatment condition, the weights are rescaled to sum to the value of the TOTALWEIGHT= option. If you omit the TOTALWEIGHT= option value and if WEIGHT=CONTROL, the weights for the control units are rescaled to equal the total frequency of the treated units. When WEIGHT=CONTROL, the weights for the treated units that are used in the balance assessments and OUT= data set are not rescaled to equal the TOTALWEIGHT= value. If you omit the TOTALWEIGHT= value and if WEIGHT=ALLOBS, the weights for the control units and the treated units are separately rescaled so that the total weight for each treatment condition equals the total frequency of the input observations.

Table Output

By default, the PSMATCH procedure displays the “Data Information” table. If you use a propensity score–based method, the procedure also displays the “Propensity Score Information” table. If you specify a MATCH statement, the procedure also displays the “Matching Information” table. If you specify a STRATA statement, the procedure also displays the “Strata Information” table. If you specify an EWEIGHT statement, the procedure also displays the “Entropy Balance Weight Information” table.

If you specify the NLARGESTWGT= option in the EWEIGHT or PSWEIGHT statement, the “Observations with Largest Weights” table is displayed. Similarly, if you specify the NMATCHMOST= option in the MATCH statement, the “Observations with Most Matches” table is also displayed.

If you specify the ASSESS statement, the “Standardized Mean Differences” table is displayed. In addition, if you specify a STRATA statement, the “Standardized Mean Differences within Strata” table is also displayed.

If you specify the VARINFO option in the ASSESS statement, the “Variable Information” table is displayed. In addition, if you specify a STRATA statement, the “Strata Variable Information” table is also displayed.

Data Information

The “Data Information” table displays the names of the input and output data sets, the numbers of observations in the treated group and the control group, and the numbers of observations in the support region that are in the treated group and the control group. The minimum and maximum propensity scores for observations in the support region are also displayed.

Entropy Balance Weight Information

The “Entropy Weight Information” table displays the treatment conditions for which weights are computed, the reference weight source, the sum of the estimated weights, the number of moments balanced for continuous variables, the number of balance constraints, and the balance tolerance value.

Matching Information

The “Matching Information” table displays the matching metric, the matching method, and the caliper width. The table also displays the number of matched sets of observations, the numbers of matched observations in the treated and control groups, and the total absolute difference across all matches.

Observations with Largest Weights

The “Observations with Largest Weights” table displays observations that have the largest weights in the treated and control groups. The table is produced only if you specify WEIGHT=ATTWGT or WEIGHT=ATEWGT in the PSWEIGHT statement. The table displays the observation numbers and their weights. If you also specify an ID statement, the table displays values of the ID variables.

Observations with Most Matches

The "Observations with Most Matches" table displays observations that have the greatest numbers of matches in the treated and control groups. The table is produced only if you specify a MATCH statement and omit the METHOD=GREEDY and METHOD=OPTIMAL options. The table displays the observation numbers and their numbers of matches. If you also specify an ID statement, the table displays the values of the ID variables.

Propensity Score Information

The "Propensity Score Information" table displays descriptive statistics (the number of observations, mean, standard deviation, minimum, and maximum) for the propensity scores of observations in the treated group and the control group. These statistics are computed using all observations, observations in the support region, and matched observations (if you specify a MATCH statement).

Standardized Mean Differences

The "Standardized Mean Differences" table displays statistics that summarize the differences in the variables and the logit propensity score (LPS) between the treated and control groups. These statistics are computed using all observations, observations in the support region, and matched observations (if you specify a MATCH statement).

The statistics include the following:

- the mean difference between observations in the treated and control groups
- the standard deviation that is used to compute the standardized mean difference
- the standardized mean difference, which is the mean difference divided by the standard deviation
- the percentage reduction of the standardized mean difference for observations in the support region, compared with the standardized mean difference of all observations (this statistic is also computed for matched observations if you specify a MATCH statement)
- the treated-to-control ratio of variances between observations in the treated and control groups

For more information about these statistics, see the sections "Standardized Mean Differences for All Observations" on page 8418, "Standardized Mean Differences for Observations in the Support Region" on page 8419, "Pooled Standardized Mean Differences across the Strata" on page 8420, and "Standardized Mean Differences for Matched Observations" on page 8420.

Strata Information

The "Strata Information" table displays descriptive statistics that include the propensity score range, the numbers of observations in the treated group and the control group, and the total number of observations in each stratum.

Strata Standardized Mean Differences

The "Standardized Mean Differences within Strata" table displays the variable difference statistics between the treated and control groups in each stratum.

For each variable, the statistics include the following:

- the mean difference between observations in the treated and control groups
- the standardized mean difference, which is the mean difference divided by the standard deviation that is displayed in the "Standardized Mean Differences" table
- the percentage reduction of the absolute standardized mean difference for observations in the stratum, compared with the absolute standardized mean difference for all observations
- the treated-to-control ratio of variances between observations in the treated and control groups in each stratum

Strata Variable Information

The "Strata Variable Information" table displays descriptive statistics that include the number of observations, variable mean, and standard deviation of the observations in each of the treatment and control groups within each stratum. For continuous variables, the statistics also include the minimum and maximum.

Variable Information

For variables that you specify in the ASSESS statement, the "Variable Information" table displays descriptive statistics that are computed using all observations and observations in each of the treatment and control groups in the support region.

These statistics include the sample size, mean, and standard deviation. For continuous variables, the statistics also include the minimum and maximum. If you specify a MATCH statement, the table also displays descriptive statistics for the matched observations in the treatment and control groups.

ODS Table Names

PROC PSMATCH assigns a name to each table that it creates. You must use these names to refer to tables when you use the Output Delivery System (ODS). These names are listed in Table 101.9. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 101.9 ODS Tables Produced by PROC PSMATCH

ODS Table Name	Description	Statements	Option
DataInfo	Data information		
EWeightInfo	Entropy balance weight information	EWEIGHT	
LargestWgtObs	Observations with the largest weights	PSWEIGHT, EWEIGHT	NLARGESTWGT=
MatchInfo	Matching information	MATCH	

Table 101.9 *continued*

ODS Table Name	Description	Statements	Option
MatchMostObs	Observations with the most matches	MATCH	NMATCHMOST=
PSInfo	Propensity score information		
StdDiff	Standardized mean differences between the treated group and the control group	ASSESS	
StrataInfo	Strata information	STRATA	
StrataStdDiff	Standardized mean differences within strata	ASSESS, STRATA	
StrataVarInfo	Strata variable information	ASSESS, STRATA	VARINFO
VarInfo	Variable information	ASSESS	VARINFO

Graphics Output

This section describes the use of ODS for creating graphics with the PSMATCH procedure. To request these graphs, ODS Graphics must be enabled and you must specify the ASSESS option. In addition, except for the standardized mean differences plot (which is the default), you must use the PLOTS= option in the ASSESS statement to specify the plots. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Bar Chart

The PLOTS=BARCHART option displays bar charts for binary classification variables in the treated and control groups for all observations and for observations in the support region. If you specify the MATCH statement, bar charts are also created for matched observations.

Box Plot

The PLOTS=BOXPLOT option displays box plots for continuous variables in the treated and control groups for all observations and for observations in the support region. If you specify the MATCH statement, box plots are also created for matched observations.

CDF Plot

The PLOTS=CDFPLOT option displays cumulative distribution function (CDF) plots for continuous variables in the treated and control groups for all observations and for observations in the support region. If you specify the MATCH statement, CDF plots are also created for matched observations.

Cloud Plot

The PLOTS=CLOUDPLOT option displays cloud plots for continuous variables in the treated and control groups for all observations and for observations in the support region. If you specify the MATCH statement, cloud plots are also created for matched observations. Here the term cloud plot refers to a scatter plot in which the points are jittered by adding random noise to data in the plot in order to prevent overplotting. For example, with a continuous variable and the default ORIENT=HORIZONTAL option, the variable values are displayed horizontally and the treated and control groups are displayed vertically. The exact variable values are displayed along the horizontal axis, but the points are jittered in the vertical direction.

Standardized Mean Differences Plot

The PLOTS=STDDIFFPLOT option displays a plot of the standardized mean differences for continuous and binary classification variables for all observations and for observations in the support region. If you specify the MATCH statement, plots are also created for matched observations.

Strata Bar Chart

If you specify a STRATA statement, the PLOTS=BARCHART option displays bar charts for binary classification variables in the treated and control groups for the observations in each stratum.

Strata Box Plot

If you specify a STRATA statement, the PLOTS=BOXPLOT option displays box plots for continuous variables in the treated and control groups for the observations in each stratum.

Strata CDF Plot

If you specify a STRATA statement, the PLOTS=CDFPLOT option displays cumulative distribution function (CDF) plots for continuous variables in the treated and control groups for the observations in each stratum.

Strata Cloud Plot

If you specify a STRATA statement, the PLOTS=CLOUDPLOT option displays cloud plots for continuous variables in the treated and control groups for the observations in each stratum. The cloud plot is a scatter plot that is jittered by adding random noise in order to prevent overplotting.

Strata Standardized Mean Differences Plot

If you specify a STRATA statement, the PLOTS=STDDIFFPLOT option displays standardized mean differences plots for continuous and binary classification variables for the observations in each stratum.

Weight Cloud Plot

The PLOTS=WGTCLOUDPLOT option displays cloud plots for weights in the treated and control groups for all observations and for observations in the support region. The cloud plot is also called the jittered scatter plot: it adds random noise to data in the plot in order to prevent overplotting. The option is applicable if you specify a PSWEIGHT statement.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC PSMATCH assigns a name to each graph it creates. You can use these names to refer to the graphs when you use ODS. To request the graph, ODS Graphics must be enabled and you must specify the ASSESS option. In addition, except for the standardized mean differences plot (which is the default), you must use the PLOTS= option in the ASSESS statement to specify the plots, as indicated in Table 101.10.

Table 101.10 Graphs Produced by PROC PSMATCH

ODS Graph Name	Plot Description	Statements	PLOTS=
BarChart	Binary variable bar chart	ASSESS	BARCHART
BoxPlot	Continuous variable box plot	ASSESS	BOXPLOT
CDFPlot	Continuous variable CDF plot	ASSESS	CDFPLOT
CloudPlot	Continuous variable cloud plot	ASSESS	CLOUDPLOT
StdDiffPlot	Standardized mean differences plot	ASSESS	STDDIFFPLOT
StrataBarChart	Strata binary variable bar chart	ASSESS, STRATA	BARCHART
StrataBoxPlot	Strata continuous variable box plot	ASSESS, STRATA	BOXPLOT
StrataCDFPlot	Strata continuous variable CDF plot	ASSESS, STRATA	CDFPLOT
StrataCloudPlot	Strata continuous variable cloud plot	ASSESS, STRATA	CLOUDPLOT
StrataStdDiffPlot	Standardized mean differences plot within strata	ASSESS, STRATA	STDDIFFPLOT
WgtCloudPlot	Weight cloud plot	ASSESS, PSWEIGHT	WGTCLOUDPLOT

Examples: PSMATCH Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

The examples in this section illustrate some of the methods of propensity analysis that are available in the PSMATCH procedure:

- Propensity score weighting is illustrated in Example 101.1.
- Stratification is illustrated in Example 101.2.
- Optimal variable ratio matching is illustrated in Example 101.3.
- Optimal one-to-one matching is illustrated in Example 101.7, Example 101.8, and Example 101.9. Example 101.7 uses Mahalanobis distance as the matching metric, and Example 101.8 and Example 101.9 use the logit of the propensity score as the matching metric.
- Greedy nearest neighbor matching is illustrated in Example 101.4 and Example 101.5.
- Matching with replacement is illustrated in Example 101.6.

With the exception of Example 101.8, all the examples use propensity scores that are obtained from a binary logistic regression model that is fitted by using the PSMATCH procedure. Example 101.8 illustrates the use of precomputed propensity scores.

The PSMATCH procedure provides a variety of statistical and graphical methods that you can use to assess covariate balance. Because this assessment is an essential aspect of propensity score analysis, the examples emphasize the use of the ASSESS statement.

Although the PSMATCH procedure does not provide outcome analysis, Example 101.5 illustrates an outcome analysis that is carried out after a propensity score analysis. Likewise, Example 101.9 illustrates a sensitivity analysis that accompanies an outcome analysis.

With the exception of Example 101.4, Example 101.5, and Example 101.6, all the examples illustrate situations in which the outcome data are not available at the time that the propensity score analysis is done. In such situations, you might not need to retain the covariate data for all individuals in the study in order to carry out the outcome analysis. For example, if you use the matching method for propensity score analysis, only the matched units are needed for follow-up. Retaining only the matched units reduces the cost of the study (Stuart 2010, p. 2).

Example 101.4, Example 101.5, and Example 101.6 illustrate situations in which the outcome data happen to be available at the time of the propensity score analysis. In such situations, the outcome data should not be used in the analysis (Stuart 2010, p. 2).

For simplicity, the examples in this section involve only a few variables. In practice, propensity score analysis often involves many more variables.

Example 101.1: Propensity Score Weighting

(View the complete code for this example (psmchex1.sas) in the example repository.)

This example illustrates how you can create observation weights that are appropriate for estimating the average treatment effect (ATE) in a subsequent outcome analysis (the outcome analysis itself is not shown here).

The data for this example are observations on patients in a nonrandomized clinical trial. The trial and the Drugs data set that contains the patient information are described in the section "Getting Started: PSMATCH Procedure" on page 8372.

The following statements specify a logistic regression model for obtaining propensity scores, compute observation weights from the propensity scores, request statistics and plots for balance assessment, and save the weights in an output data set:

```
ods graphics on;
proc psmatch data=drugs region=allobs;
   class Drug Gender;
   psmodel Drug(Treated='Drug_X')= Gender Age BMI;
   psweight weight=atewgt nlargestwgt=6;
   assess lps var=(Gender Age BMI)
          / varinfo plots=(barchart boxplot(display=(lps BMI)) wgtcloud);
   id BMI;
   output out(obs=all)=OutEx1 weight=_ATEWgt_;
run;
```

The PSMODEL statement specifies the logistic regression model that creates the propensity score for each observation, which is the probability that the patient receives Drug_X. The CLASS statement specifies the classification variables in the model. The Drug variable is the binary treatment indicator variable, and TREATED='Drug_X' identifies Drug_X as the treated group. The Gender, Age, and BMI variables are included in the model because they are believed to be related to the assignment. The REGION=ALLOBS option specifies that the support region contains all observations. Weights are computed for all observations, regardless of the REGION= option.

The "Data Information" table in Output 101.1.1 displays the numbers of observations in the treated and control groups, the lower and upper limits of the propensity scores for observations in the support region, and the numbers of observations in the treated and control groups that fall within the support region. Because REGION=ALLOBS is specified, the lower and upper limits for of the propensity scores for observations in the support region are the minimum and maximum of the propensity scores for all observations. Consequently, all 373 observations in the control group fall within the support region, and all 133 observations in the treated group fall within the support region.

Output 101.1.1 Data Information

The PSMATCH Procedure

Data Information	
Data Set	WORK.DRUGS
Output Data Set	WORK.OUTEX1
Treatment Variable	Drug
Treated Group	Drug_X
All Obs (Treated)	113
All Obs (Control)	373
Support Region	All Obs
Lower PS Support	0.020157
Upper PS Support	0.685757
Support Region Obs (Treated)	113
Support Region Obs (Control)	373

The "Propensity Score Information" table in Output 101.1.2 displays summary statistics by treatment group for all observations (labeled "All"), for observations in the support region (labeled "Region"), and for weighted observations in the support region (labeled "Weighted"). Because the support region consists of all observations, the first two rows in the table are identical. The WEIGHT=ATEWGT option in the PSWEIGHT statement displays summary statistics for ATE weighted observations.

Output 101.1.2 Propensity Score Information

Propensity Score Information													
	Treated (Drug = Drug_X)						Control (Drug = Drug_A)						
Observations	**N**	**Weight**	**Mean**	**Standard Deviation**	**Minimum**	**Maximum**	**N**	**Weight**	**Mean**	**Standard Deviation**	**Minimum**	**Maximum**
All	113		0.3108	0.1325	0.0602	0.6411	373		0.2088	0.1320	0.0202	0.6858
Region	113		0.3108	0.1325	0.0602	0.6411	373		0.2088	0.1320	0.0202	0.6858
Weighted	113	460.45	0.2454	0.1268	0.0602	0.6411	373	489.59	0.2381	0.1496	0.0202	0.6858

Propensity Score Information	
Observations	**Treated - Control Mean Difference**
All	0.1020
Region	0.1020
Weighted	0.0073

The ASSESS statement produces tables and plots, shown in Output 101.1.3 through Output 101.1.5 and in Output 101.1.7 through Output 101.1.10, that summarize differences in the distributions of specified variables between treated and control groups. As requested by the LPS and VAR= options, these variables are the logit of the propensity score and the data variables Gender, Age, and BMI. Differences are summarized for all observations and for observations in the support region. Again, these two sets of differences are identical because REGION=ALLOBS is specified. The WEIGHT=ATEWGT option in the PSWEIGHT statement requests that differences in the variables also be summarized for the weighted observations. By comparing the differences for weighted observations to the differences for observations in the support region, you can

assess how well weighting improves the balance for each variable.

The VARINFO option requests the "Variable Information" table, shown in Output 101.1.3, which displays variable summary statistics and differences between the treated and control groups for all observations (labeled "All"), for observations in the support region (labeled "Region"), and for weighted observations (labeled "Weighted"). For the binary classification variable (Gender), the difference is in the proportion of the first ordered level (Female).

Output 101.1.3 Variable Information

The PSMATCH Procedure

Variable Information							
				Treated (Drug = Drug_X)			
Variable	Observations	N	Weight	Mean	Standard Deviation	Minimum	Maximum
Logit Prop Score	All	113		-0.88062	0.681761	-2.74745	0.58035
	Region	113		-0.88062	0.681761	-2.74745	0.58035
	Weighted	113	460.45	-1.25406	0.741386	-2.74745	0.58035
Age	All	113		36.30973	5.534114	26.00000	49.00000
	Region	113		36.30973	5.534114	26.00000	49.00000
	Weighted	113	460.45	38.59813	5.773228	26.00000	49.00000
BMI	All	113		24.49257	1.863797	20.33000	28.34000
	Region	113		24.49257	1.863797	20.33000	28.34000
	Weighted	113	460.45	24.03522	1.896607	20.33000	28.34000
Gender	All	113		0.43363	0.495575		
	Region	113		0.43363	0.495575		
	Weighted	113	460.45	0.47335	0.499289		

Variable Information								
				Control (Drug = Drug_A)				Treated - Control
Variable	Observations	N	Weight	Mean	Standard Deviation	Minimum	Maximum	Mean Difference
Logit Prop Score	All	373		-1.52059	0.844486	-3.88386	0.78036	0.63997
	Region	373		-1.52059	0.844486	-3.88386	0.78036	0.63997
	Weighted	373	489.59	-1.35103	0.894234	-3.88386	0.78036	0.09698
Age	All	373		40.40483	6.579103	25.00000	57.00000	-4.09509
	Region	373		40.40483	6.579103	25.00000	57.00000	-4.09509
	Weighted	373	489.59	39.32670	6.771606	25.00000	57.00000	-0.72857
BMI	All	373		23.75327	1.980778	19.22000	28.61000	0.73930
	Region	373		23.75327	1.980778	19.22000	28.61000	0.73930
	Weighted	373	489.59	23.95492	2.004019	19.22000	28.61000	0.08030
Gender	All	373		0.45845	0.498270			-0.02482
	Region	373		0.45845	0.498270			-0.02482
	Weighted	373	489.59	0.45479	0.497952			0.01856

The statistics in Output 101.1.3 are identical for all observations and for observations in the support region because REGION=ALLOBS is specified.

As indicated in the column labeled Weight, the total weight of the treated units is 460.45 and the total weight of the control units is 489.59, which are close to 486, the total number of units. The weights are ATE weights

because WEIGHT=ATEWGT is specified in the PSWEIGHT statement. For information about ATE weights, see the section "Inverse Probability of Treatment Weighting" on page 8408.

Note that in comparison to the unweighted means, the weighted means for the control units are closer in absolute value to the corresponding weighted means for the treated units.

The "Standardized Mean Differences" table, shown in Output 101.1.4, displays standardized mean differences in the variables between the treated and control groups, based on all observations, on observations in the support region, and on weighted observations.

Output 101.1.4 Standardized Mean Differences

Standardized Mean Differences (Treated - Control)						
Variable	Observations	Mean Difference	Standard Deviation	Standardized Difference	Percent Reduction	Variance Ratio
Logit Prop Score	All	0.63997	0.767449	0.83389		0.6517
	Region	0.63997		0.83389	0.00	0.6517
	Weighted	0.09698		0.12636	84.85	0.6874
Age	All	-4.09509	6.079104	-0.67363		0.7076
	Region	-4.09509		-0.67363	0.00	0.7076
	Weighted	-0.72857		-0.11985	82.21	0.7269
BMI	All	0.73930	1.923178	0.38441		0.8854
	Region	0.73930		0.38441	0.00	0.8854
	Weighted	0.08030		0.04175	89.14	0.8957
Gender	All	-0.02482	0.496925	-0.04994		0.9892
	Region	-0.02482		-0.04994	0.00	0.9892
	Weighted	0.01856		0.03735	25.21	1.0054
Standard deviation of All observations used to compute standardized differences						

The standardized mean differences based on weighted observations are significantly reduced; the largest of these differences is 0.12637 in absolute value, which is less than the upper limit of 0.25 that is recommended by Rubin (2001, p. 174) and Stuart (2010, p. 11). The treated-to-control variance ratios between the two groups are within the recommended range of 0.5 to 2. The percentage of reduction in variable mean difference is 0 for observations in the support region because REGION=ALLOBS is specified.

The PSMATCH procedure displays a standardized mean differences plot, shown in Output 101.1.5, for the variables that are specified in the ASSESS statement.

Output 101.1.5 Standardized Mean Differences Plot

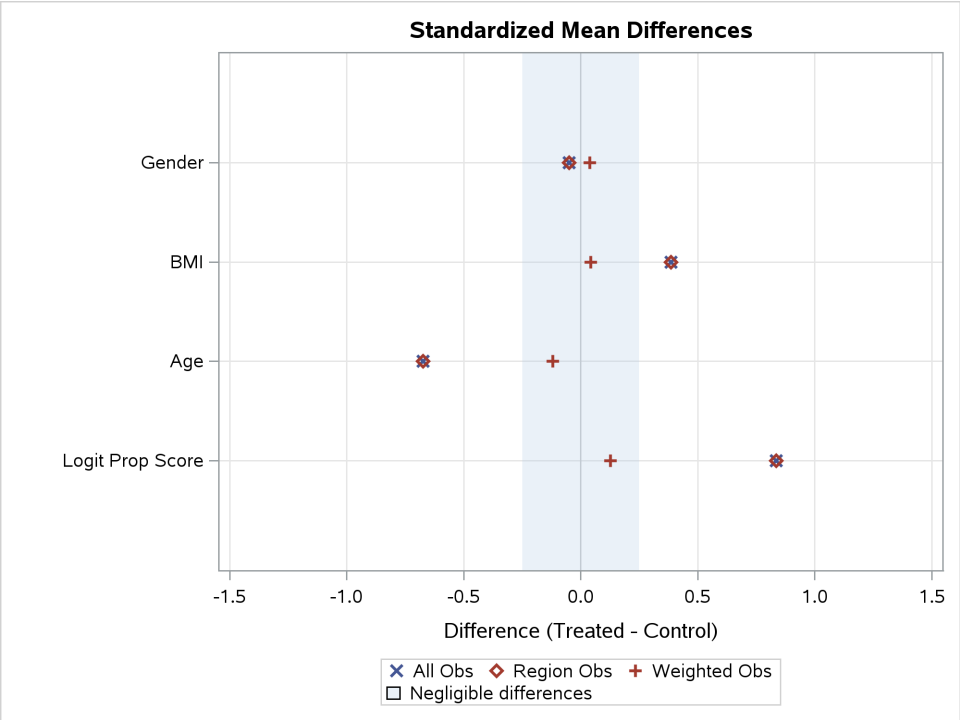

The "Standardized Mean Differences Plot" displays the differences that are shown in the "Standardized Mean Differences" table in Output 101.1.4. All differences for the weighted observations are within the recommended limits of –0.25 and 0.25, which are indicated by the shaded area.

The NLARGESTWGT=6 option displays the "Observations with Largest Weights" table, shown in Output 101.1.6, which lists the observations that have the six largest weights in the treated and control groups.

Output 101.1.6 Observations with Largest Weights

Observations with Largest IPTW-ATE Weights							
Treated (Drug = Drug_X)				Control (Drug = Drug_A)			
Expected Weight = 4.3009				Expected Weight = 1.3029			
Observation	BMI	Weight	Scaled Weight	Observation	BMI	Weight	Scaled Weight
202	20.75	16.60	3.86	**317**	28.61	3.18	2.44
479	22.22	14.79	3.44	**134**	28.07	3.15	2.42
250	23.96	11.40	2.65	**437**	25.76	2.74	2.10
227	21.11	11.23	2.61	**417**	26.81	2.62	2.01
274	24.17	9.69	2.25	**446**	27.75	2.62	2.01
174	23.56	9.02	2.10	**81**	27.20	2.40	1.84

In the table, the scaled weights (which are the weights divided by their expected weights) are also displayed for ease of comparison. For more information about the expected weights in the treated and control group, see the section "Propensity Score Weighting" on page 8407.

The PLOTS=WGTCLOUD option displays a cloud plot for the stabilized weights, which is shown in Output 101.1.7. This plot is called a cloud plot because the points are jittered in the vertical direction in order

to avoid overplotting.

Output 101.1.7 Weight Cloud Plot

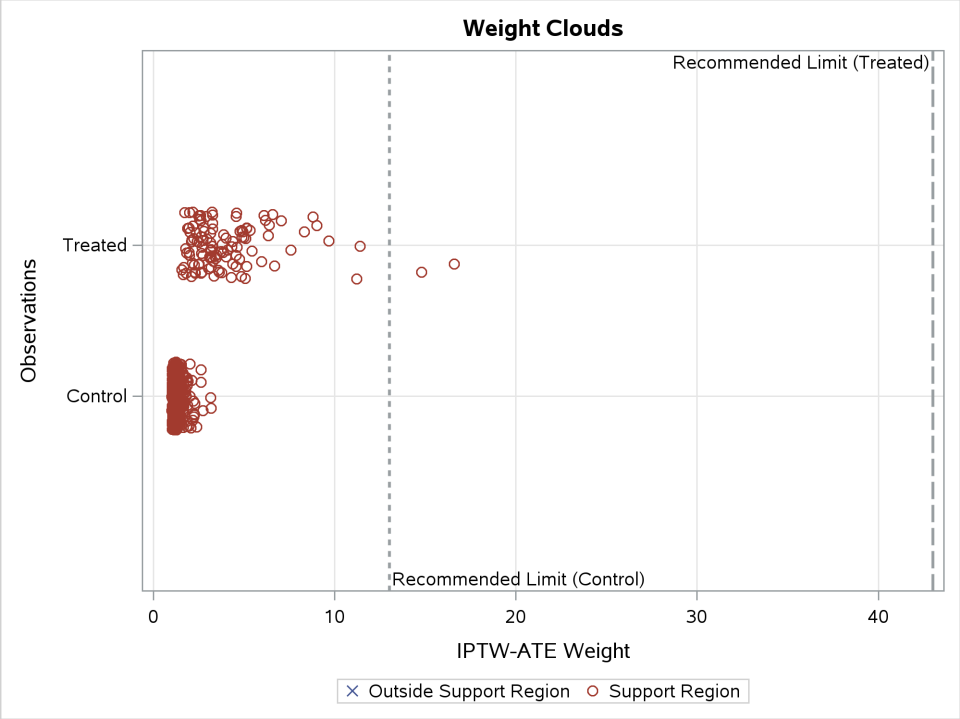

By default, the plot displays reference lines that represent 10 times the expected ATE weights in the treated and control groups. For information about these average weights, see the section "Inverse Probability of Treatment Weighting" on page 8408.

The PLOTS=BARCHART option displays a bar chart for each classification variable that is specified in the ASSESS statement. As shown in Output 101.1.8, the bar chart shows the distributions of Gender based on all observations, on observations in the support region, and on weighted observations. By default, the bar chart displays the proportions of levels of Gender. Weighting the observations makes a slight improvement in the balance between males and females.

Output 101.1.8 Gender Bar Chart

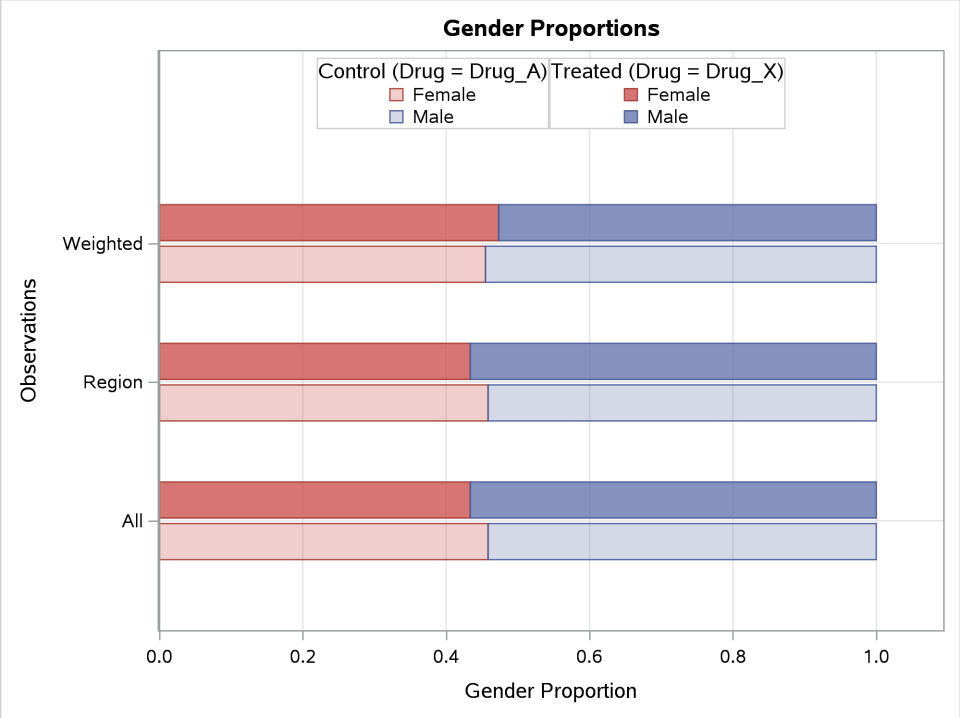

The PLOTS=BOXPLOT(DISPLAY=(LPS BMI)) option displays box plots for LPS and BMI, as shown in Output 101.1.9 and Output 101.1.10, respectively. These plots compare the distributions of the variables for the treated and control groups. Weighting the observations makes a good improvement in the balance between males and females.

Output 101.1.9 LPS Box Plot

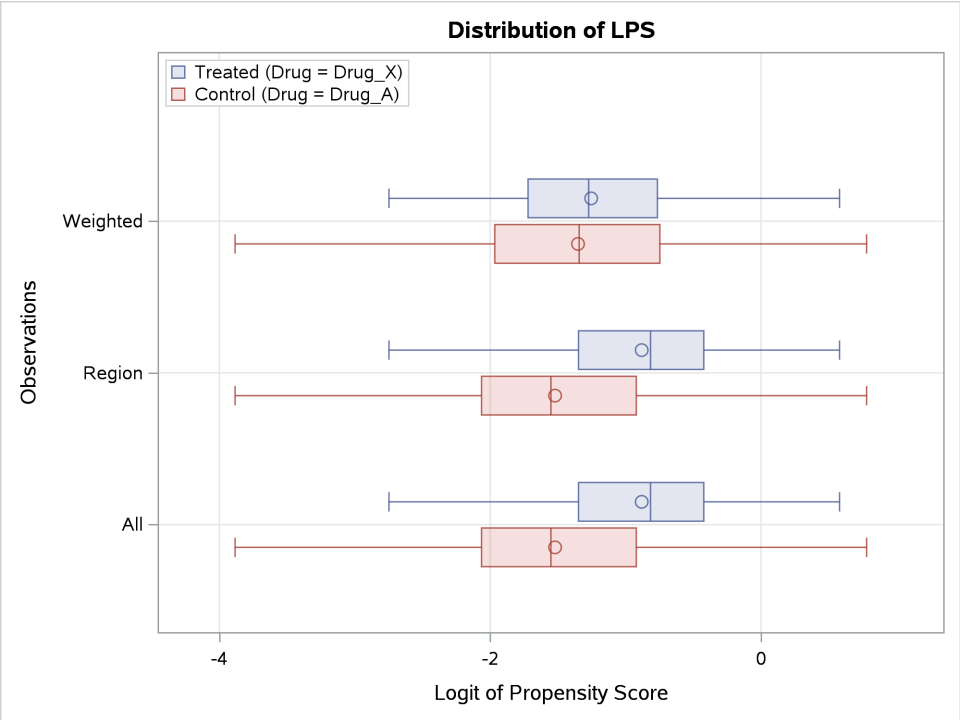

Output 101.1.10 BMI Box Plot

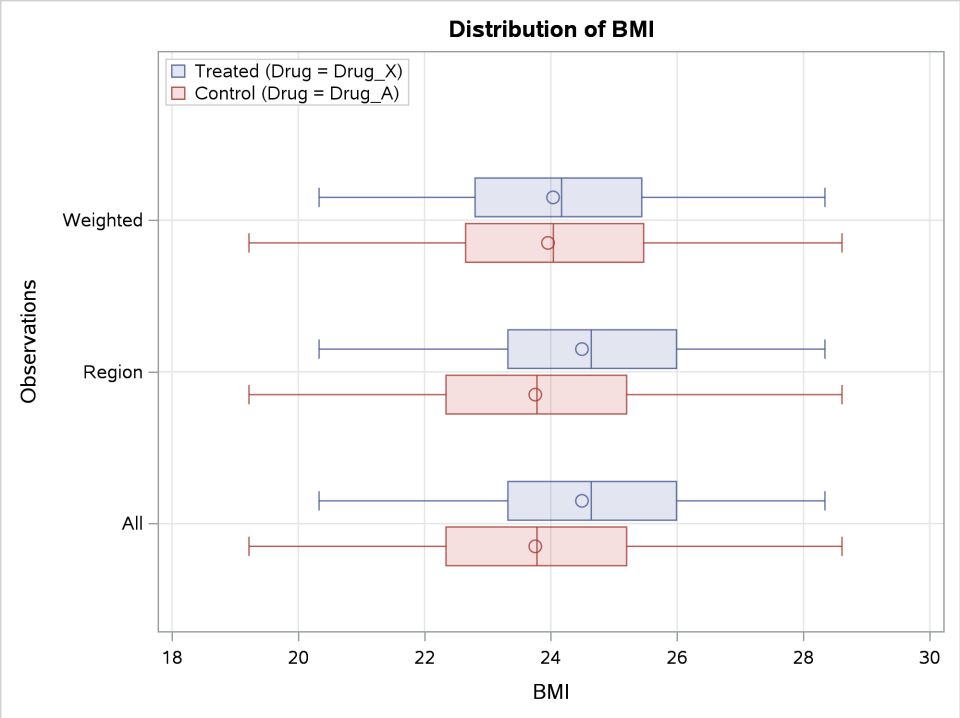

Because there is good balance in the weighted distributions of the variables Gender, Age, and BMI, the

observations and their weights can be saved in an output data set for use in a subsequent outcome analysis.

In situations where you are not satisfied with the variable balance, you can do one or more of the following to improve the balance: you can select another set of variables to fit the propensity score model, you can modify the specification of the propensity score model by using nonlinear terms for the continuous variables or by adding interactions (Rosenbaum and Rubin 1984), or you can choose another propensity score method (such as matching).

The OUT(OBS=ALL)=OutEx1 option in the OUTPUT statement creates an output data set, OutEx1, that contains all available observations. The following statements list the first 10 observations in OutEx1, as shown in Output 101.1.11.

```
proc print data=OutEx1(obs=10);
   var PatientID Drug Gender Age BMI _ps_ _AteWgt_;
run;
```

Output 101.1.11 Output Data Set with ATE Weights

Obs	PatientID	Drug	Gender	Age	BMI	_PS_	_ATEWgt_
1	284	Drug_X	Male	29	22.02	0.36444	2.74397
2	201	Drug_A	Male	45	26.68	0.22296	1.28694
3	147	Drug_A	Male	42	21.84	0.11323	1.12768
4	307	Drug_X	Male	38	22.71	0.19733	5.06767
5	433	Drug_A	Male	31	22.76	0.35311	1.54586
6	435	Drug_A	Male	43	26.86	0.27263	1.37482
7	159	Drug_A	Female	45	25.47	0.14911	1.17523
8	368	Drug_A	Female	49	24.28	0.07780	1.08437
9	286	Drug_A	Male	31	23.31	0.38341	1.62182
10	163	Drug_X	Female	39	25.34	0.24995	4.00073

By default, the output data set includes the variable _PS_, which provides the propensity score. The weight for each treated unit is computed as $1 / p$ and the weight for each control unit is computed as $1 / (1 - p)$, where p is the propensity score.

After the responses for the trial are observed, they can be added to the data set OutEx1 as the starting point for an outcome analysis. Assuming that no other confounding variables are associated with both the response variable and the treatment group indicator Drug, you can estimate the ATE by performing a weighted version of the outcome analysis that you would have used to estimate the treatment effect if the original data set had resulted from a randomized trial.

Example 101.2: Propensity Score Stratification

(View the complete code for this example (psmchex2.sas) in the example repository.)

This example illustrates how you can stratify observations based on their propensity scores, so that the stratified observations can be used to estimate the treatment effect in a subsequent outcome analysis (the outcome analysis is not shown here).

The data for this example are observations on patients in a nonrandomized clinical trial. The trial and the Drugs data set that contains the patient information are described in the section "Getting Started: PSMATCH Procedure" on page 8372.

The following statements create five strata that are based on propensity scores:

```
ods graphics on;
proc psmatch data=drugs region=allobs;
   class Drug Gender;
   psmodel Drug(Treated='Drug_X')= Gender Age BMI;
   strata nstrata=5 key=treated stratumwgt=total;
   assess ps var=(Gender BMI)
          / varinfo plots=(barchart cdfplot);
   output out(obs=all)=OutEx2;
run;
```

The PSMODEL statement specifies the logistic regression model that creates the propensity score for each observation, which is the probability that the patient receives Drug_X. The CLASS statement specifies the classification variables in the model. The Drug variable is the binary treatment indicator variable, and TREATED='Drug_X' identifies Drug_X as the treated group. The Gender, Age, and BMI variables are included in the model because they are believed to be related to the assignment.

The PSMATCH procedure stratifies the observations whose propensity scores lie in the support region that is specified in the REGION= option. The REGION=ALLOBS option requests that all observations be stratified.

The STRATA statement creates strata of observations based on their propensity scores. The NSTRATA=5 option (which is the default) stratifies the observations into five strata and the KEY=TREATED option (which is the default) requests that each stratum contain approximately the same number of treated observations.

The "Data Information" table in Output 101.2.1 displays the numbers of observations in the treated and control groups, the lower and upper limits of the propensity scores for observations in the support region, and the numbers of observations in the treated and control groups that fall within the support region. Because REGION=ALLOBS is specified, the lower and upper limits of the propensity scores for observations in the support region are simply the minimum and maximum of the propensity scores for all observations. Likewise, all 373 observations in the control group fall within the support region.

Output 101.2.1 Data Information

The PSMATCH Procedure

Data Information	
Data Set	WORK.DRUGS
Output Data Set	WORK.OUTEX2
Treatment Variable	Drug
Treated Group	Drug_X
All Obs (Treated)	113
All Obs (Control)	373
Support Region	All Obs
Lower PS Support	0.020157
Upper PS Support	0.685757
Support Region Obs (Treated)	113
Support Region Obs (Control)	373
Number of Strata	5

The "Propensity Score Information" table in Output 101.2.2 displays summary statistics for the treated and control groups. Statistics are computed for all observations, for observations in the support region, and for strata. The first two rows of statistics are identical because REGION=ALLOBS is specified.

Output 101.2.2 Propensity Score Information

Propensity Score Information											
	Treated (Drug = Drug_X)					Control (Drug = Drug_A)					Treated - Control
Observations	**N**	**Mean**	**Standard Deviation**	**Minimum**	**Maximum**	**N**	**Mean**	**Standard Deviation**	**Minimum**	**Maximum**	**Mean Difference**
All	113	0.3108	0.1325	0.0602	0.6411	373	0.2088	0.1320	0.0202	0.6858	0.1020
Region	113	0.3108	0.1325	0.0602	0.6411	373	0.2088	0.1320	0.0202	0.6858	0.1020
Strata	5	0.2426	0.0212	0.1404	0.5121	5	0.2318	0.0227	0.1159	0.5312	0.0107

Note that for strata, the displayed statistics refer to the strata rather than to the observational units. These statistics are the number of strata, the weighted mean of the stratum means, the standard deviation of this weighted mean, and the minimum and maximum of these stratum means. The weights for computing the weighted mean are the stratum weights (as shown in the last column of Output 101.2.3), which are proportions of total units (treated and control units combined) in strata by the default STRATUMWGT=TOTAL option.

When you specify a STRATA statement, the "Strata Information" table, which is shown in Output 101.2.3, displays the following information for each stratum: the minimum and maximum propensity scores, the number of observations in the treatment group, the number of observations in the control group, the total number of observations, and the stratum weight.

Output 101.2.3 Strata Information

Strata Information						
			Frequencies			
Stratum Index	Propensity Score Range		Treated	Control	Total	Stratum Weight
1	0.0202	0.1944	22	209	231	0.475
2	0.1967	0.2613	23	59	82	0.169
3	0.2619	0.3223	23	38	61	0.126
4	0.3259	0.4342	23	41	64	0.132
5	0.4379	0.6858	22	26	48	0.099

The table shows that each stratum contains approximately the same number of observations for treated units, as requested by the KEY=TREATED option. In addition, there are enough control units in each stratum to ensure a reliable estimate of the treatment effect for this stratum, even though the propensity score distributions in the treated and control groups are different.

The ASSESS statement produces tables and plots, shown in Output 101.2.4 through Output 101.2.14, that summarize differences in the distributions of specified variables between treated and control groups. As requested by the PS and VAR= options, these variables are the propensity score and the data variables Gender and BMI. By default, differences are summarized for all observations and for observations in the support region. Again, these two sets of differences are identical because REGION=ALLOBS is specified. When you specify a STRATA statement, differences after stratification are also displayed.

The VARINFO option requests the "Variable Information" table, shown in Output 101.2.4, which displays variable summary statistics and mean differences between the treated and control groups for all observations (labeled "All"), for observations in the support region (labeled "Region"), and for the stratified observations (labeled "Strata"). The first two sets of statistics and mean differences are identical because REGION=ALLOBS is specified. For the binary classification variable (Gender), the difference is in the proportion of the first ordered level (Female).

Output 101.2.4 Variable Information

The PSMATCH Procedure

Variable Information											
		Treated (Drug = Drug_X)					Control (Drug = Drug_A)				
Variable	Observations	N	Mean	Standard Deviation	Minimum	Maximum	N	Mean	Standard Deviation	Minimum	Maximum
Prop Score	All	113	0.31077	0.132467	0.06023	0.64115	373	0.20880	0.131969	0.02016	0.68576
	Region	113	0.31077	0.132467	0.06023	0.64115	373	0.20880	0.131969	0.02016	0.68576
	Strata	5	0.24256	0.021157	0.14040	0.51209	5	0.23182	0.022710	0.11589	0.53118
BMI	All	113	24.49257	1.863797	20.33000	28.34000	373	23.75327	1.980778	19.22000	28.61000
	Region	113	24.49257	1.863797	20.33000	28.34000	373	23.75327	1.980778	19.22000	28.61000
	Strata	5	24.08101	0.954962	23.50091	25.69500	5	23.89531	1.019772	23.17541	25.73077
Gender	All	113	0.43363	0.495575			373	0.45845	0.498270		
	Region	113	0.43363	0.495575			373	0.45845	0.498270		
	Strata	5	0.44921	0.270442			5	0.45232	0.270450		

Variable Information		
Variable	Observations	Treated - Control Mean Difference
Prop Score	All	0.10197
	Region	0.10197
	Strata	0.01074
BMI	All	0.73930
	Region	0.73930
	Strata	0.18570
Gender	All	-0.02482
	Region	-0.02482
	Strata	-0.00311

For each variable, the row labeled "Strata" displays the number of strata, the weighted mean of the stratum means, and the standard deviation of this weighted mean, where the weights are computed as proportions of total units (treated and control units combined) in strata by the default STRATUMWGT=TOTAL option. The row also displays the minimum and maximum of the variable averages within the strata.

For each variable, the "Standardized Mean Differences" table in Output 101.2.5 displays the standardized mean differences between the treated and control groups for all observations, for observations in the support region, and for the stratified observations. The sections "Weighting after Stratification" on page 8411 and "Pooled Standardized Mean Differences across the Strata" on page 8420 explain how the statistics are computed for the stratified observations.

Output 101.2.5 Standardized Mean Differences

Standardized Mean Differences (Treated - Control)						
Variable	Observations	Mean Difference	Standard Deviation	Standardized Difference	Percent Reduction	Variance Ratio
Prop Score	All	0.10197	0.132218	0.77124		1.0076
	Region	0.10197		0.77124	0.00	1.0076
	Strata	0.01074		0.08121	89.47	0.8678
BMI	All	0.73930	1.923178	0.38441		0.8854
	Region	0.73930		0.38441	0.00	0.8854
	Strata	0.18570		0.09656	74.88	0.8769
Gender	All	-0.02482	0.496925	-0.04994		0.9892
	Region	-0.02482		-0.04994	0.00	0.9892
	Strata	-0.00311		-0.00627	87.45	0.9999
Standard deviation of All observations used to compute standardized differences						

When you specify a STRATA statement, the ASSESS statement also produces stratum-specific versions of tables and plots that summarize differences in the distributions of the specified variables between treated and control groups.

In addition to the "Variable Information" table shown in Output 101.2.4, the VARINFO option in the ASSESS statement produces the "Strata Variable Information" table, shown in Output 101.2.6, which displays variable summary statistics and mean differences between the treated and control groups for the observations in each stratum.

Output 101.2.6 Strata Variable Information

The PSMATCH Procedure

Strata Variable Information												
		Treated (Drug = Drug_X)					Control (Drug = Drug_A)					Treated - Control
Variable	Stratum Index	N	Mean	Standard Deviation	Minimum	Maximum	N	Mean	Standard Deviation	Minimum	Maximum	Mean Difference
Prop Score	1	22	0.14040	0.041360	0.06023	0.19436	209	0.11589	0.043859	0.02016	0.19413	0.02451
	2	23	0.22199	0.019418	0.19674	0.25936	59	0.22821	0.018395	0.19734	0.26130	-0.00622
	3	23	0.29986	0.018811	0.26350	0.32230	38	0.29457	0.017541	0.26186	0.32156	0.00529
	4	23	0.38087	0.026077	0.32668	0.43418	41	0.37055	0.030646	0.32594	0.43421	0.01032
	5	22	0.51209	0.058200	0.43793	0.64115	26	0.53118	0.071893	0.44120	0.68576	-0.01910
BMI	1	22	23.50091	1.751203	20.33000	26.11000	209	23.17541	1.917237	19.24000	27.85000	0.32550
	2	23	23.65304	1.794401	20.43000	26.66000	59	23.87847	1.951062	19.22000	27.68000	-0.22543
	3	23	24.70783	1.764444	20.85000	27.56000	38	24.10816	1.698325	20.24000	27.60000	0.59967
	4	23	24.91522	1.950177	20.98000	28.34000	41	24.93585	1.484916	22.37000	28.29000	-0.02064
	5	22	25.69500	1.130338	23.32000	28.06000	26	25.73077	1.337953	23.41000	28.61000	-0.03577
Gender	1	22	0.45455	0.497930			209	0.50718	0.499948			-0.05263
	2	23	0.56522	0.495728			59	0.32203	0.467256			0.24318
	3	23	0.39130	0.488042			38	0.44737	0.497222			-0.05606
	4	23	0.43478	0.495728			41	0.39024	0.487805			0.04454
	5	22	0.31818	0.465770			26	0.50000	0.500000			-0.18182

The "Standardized Mean Differences within Strata" table in Output 101.2.7 is a stratum-specific version of the

"Standardized Mean Differences" table in Output 101.2.5; it displays the variable mean differences, standardized mean differences, percentage reductions, ratios of variances for the observations, and stratum weights in each stratum. In Output 101.2.7, the standardized mean difference is the variable mean difference divided by the standard deviation shown in the "Standardized Mean Differences" table; the percentage reduction compares the standardized mean difference with the standardized mean difference of all observations.

The stratum weight is the number of treated units in each stratum divided by the combined number of treated units, as specified by the STRATUMWGT=TREATED option.

Output 101.2.7 Standardized Mean Differences within Strata

Standardized Mean Differences (Treated - Control) within Strata						
Variable	Stratum Index	Mean Difference	Standardized Difference	Percent Reduction	Variance Ratio	Stratum Weight
Prop Score	1	0.02451	0.18537	75.96	0.8893	0.475
	2	-0.00622	-0.04703	93.90	1.1143	0.169
	3	0.00529	0.04003	94.81	1.1500	0.126
	4	0.01032	0.07803	89.88	0.7241	0.132
	5	-0.01910	-0.14443	81.27	0.6553	0.099
BMI	1	0.32550	0.16925	55.97	0.8343	0.475
	2	-0.22543	-0.11722	69.51	0.8459	0.169
	3	0.59967	0.31181	18.89	1.0794	0.126
	4	-0.02064	-0.01073	97.21	1.7248	0.132
	5	-0.03577	-0.01860	95.16	0.7137	0.099
Gender	1	-0.05263	-0.10591	0.00	0.9919	0.475
	2	0.24318	0.48938	0.00	1.1256	0.169
	3	-0.05606	-0.11282	0.00	0.9634	0.126
	4	0.04454	0.08963	0.00	1.0328	0.132
	5	-0.18182	-0.36589	0.00	0.8678	0.099

Note that a zero percentage reduction is displayed for Gender in each stratum because its standardized mean difference in the stratum (in absolute value) is larger than the standardized mean difference of all observations (0.04994 in absolute value).

Output 101.2.8 displays a standardized mean differences plot for the variables that are specified in the ASSESS statement.

Output 101.2.8 Standardized Mean Differences Plot

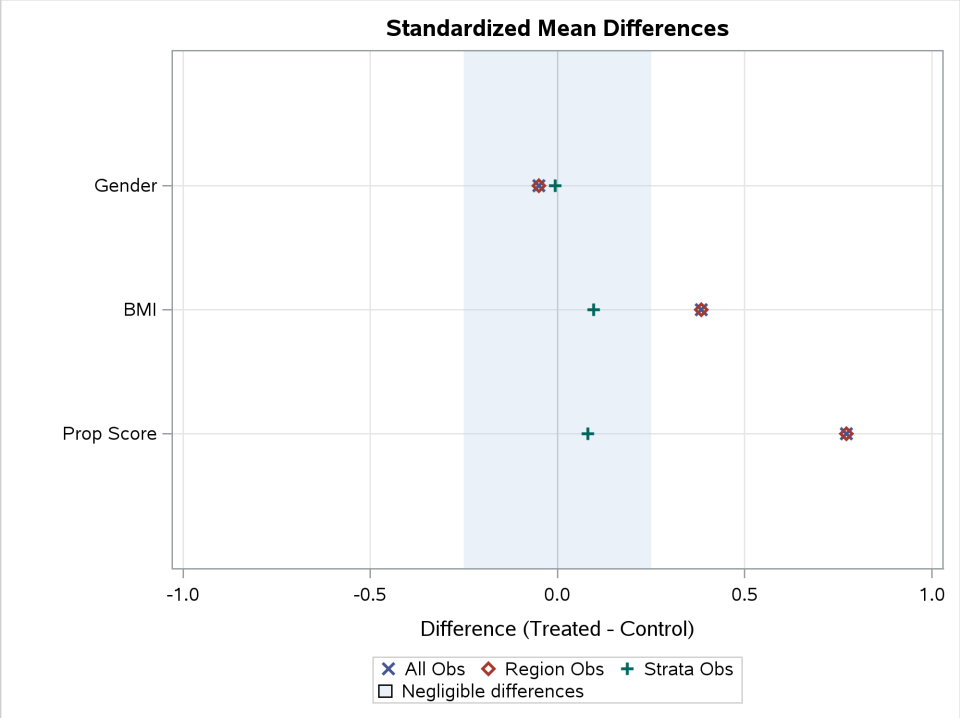

In addition to differences based on all observations and on observations in the support region (which are identical), this plot displays differences based on combining estimates across strata, which are much smaller. For more information about these differences, see the sections "Weighting after Stratification" on page 8411 and "Pooled Standardized Mean Differences across the Strata" on page 8420.

Output 101.2.9 displays a plot of the standardized mean differences for each of the five strata.

Output 101.2.9 Standardized Mean Differences within Strata Plot

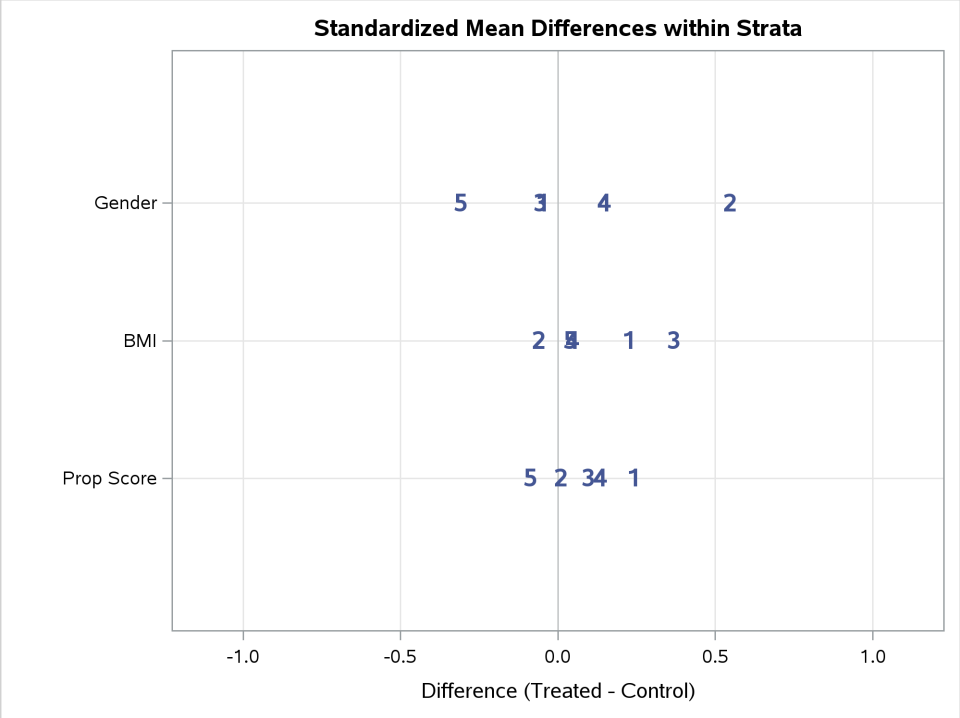

Note that recommended ranges for stratum-specific standardized mean differences are currently not available in the literature.

The "Standardized Mean Differences within Strata" plot corresponds to the "Standardized Mean Differences within Strata" table in Output 101.2.9. The plot reveals larger differences in Stratum 2 and Stratum 5 for Gender.

The PLOTS=BARCHART option displays stratum-specific bar charts for the distributions of classification variables in the treated and control groups, as shown in Output 101.2.10 for Gender. Here the largest differences in the distributions occur in Stratum 2 and Stratum 5.

Output 101.2.10 Gender Strata Bar Chart

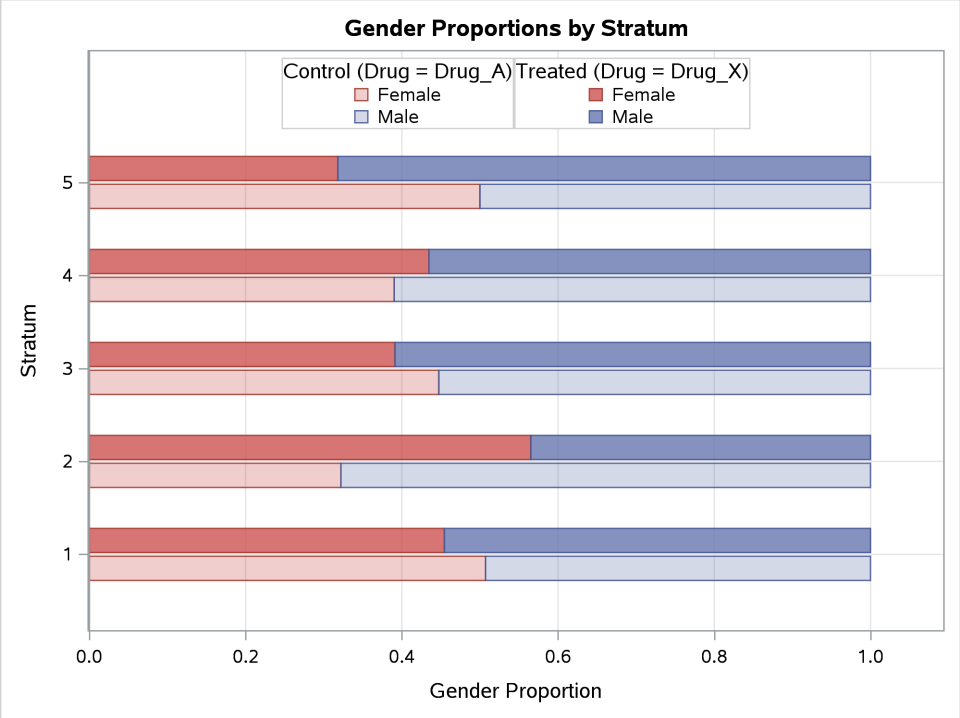

The PLOTS=CDFPLOT option displays stratum-specific CDF plots for the continuous variables in the treated and control groups, as shown in Output 101.2.11 and Output 101.2.12 for PS and in Output 101.2.13 and Output 101.2.14 for BMI.

Output 101.2.11 PS Strata CDF Plot

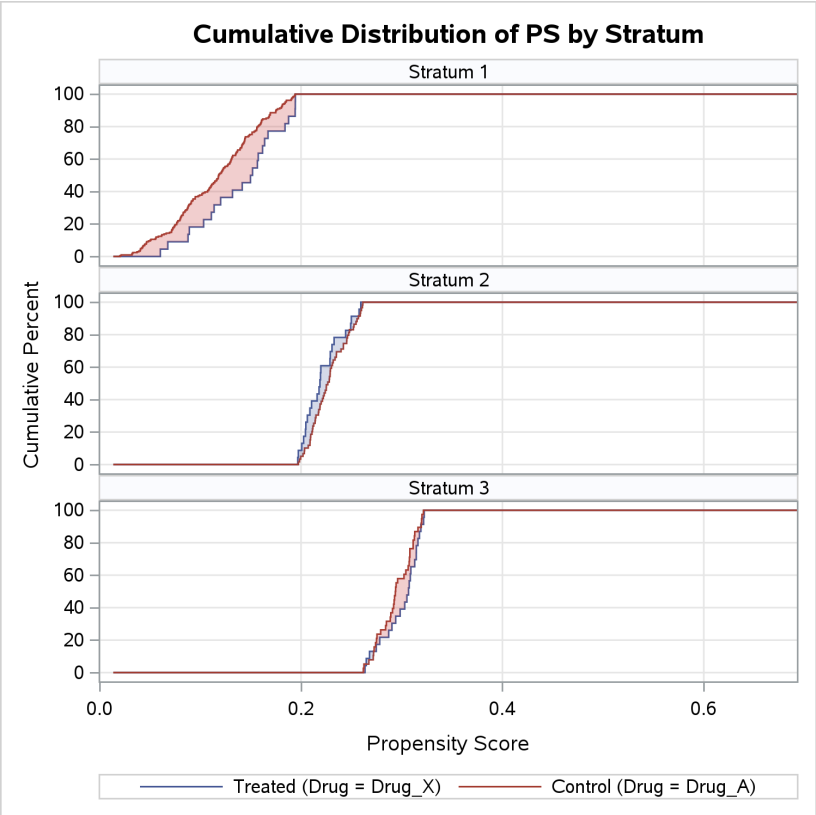

Output 101.2.12 PS Strata CDF Plot

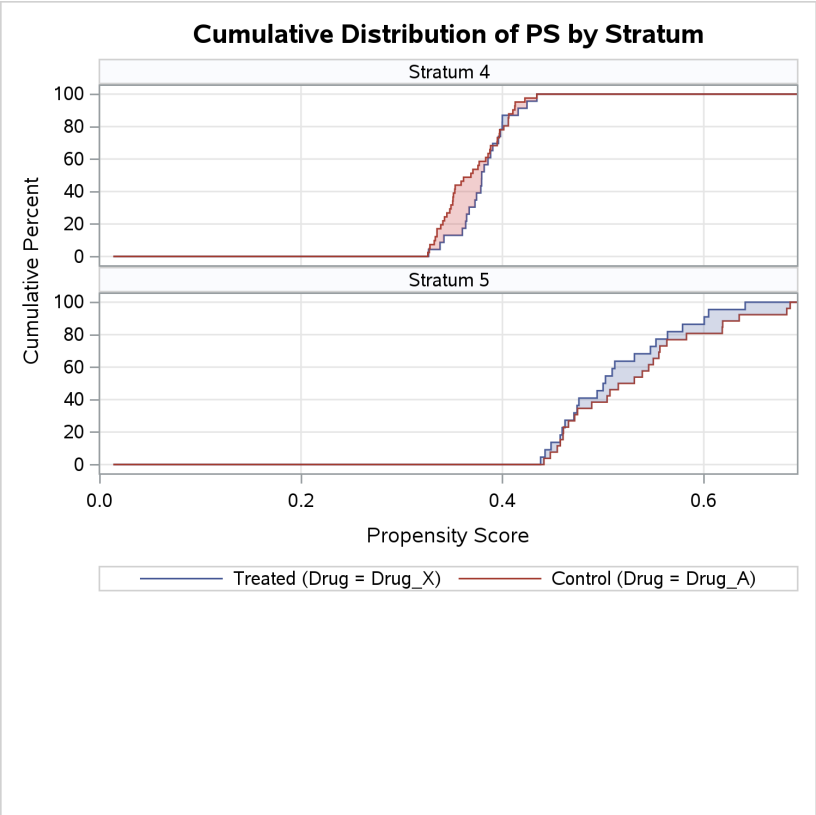

Output 101.2.13 BMI Strata CDF Plot

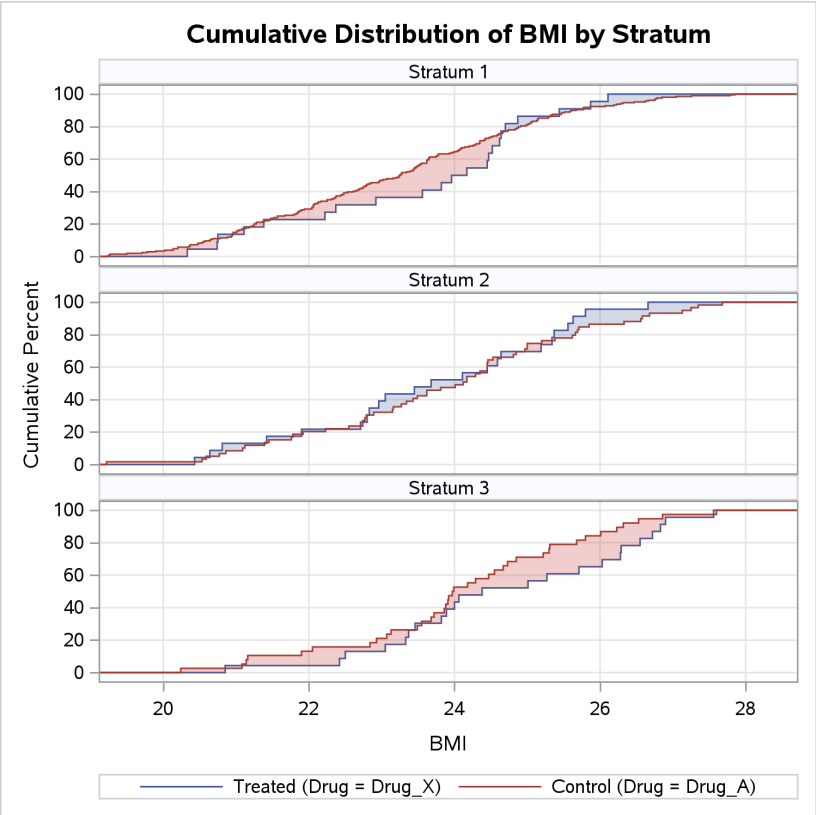

Output 101.2.14 BMI Strata CDF Plot

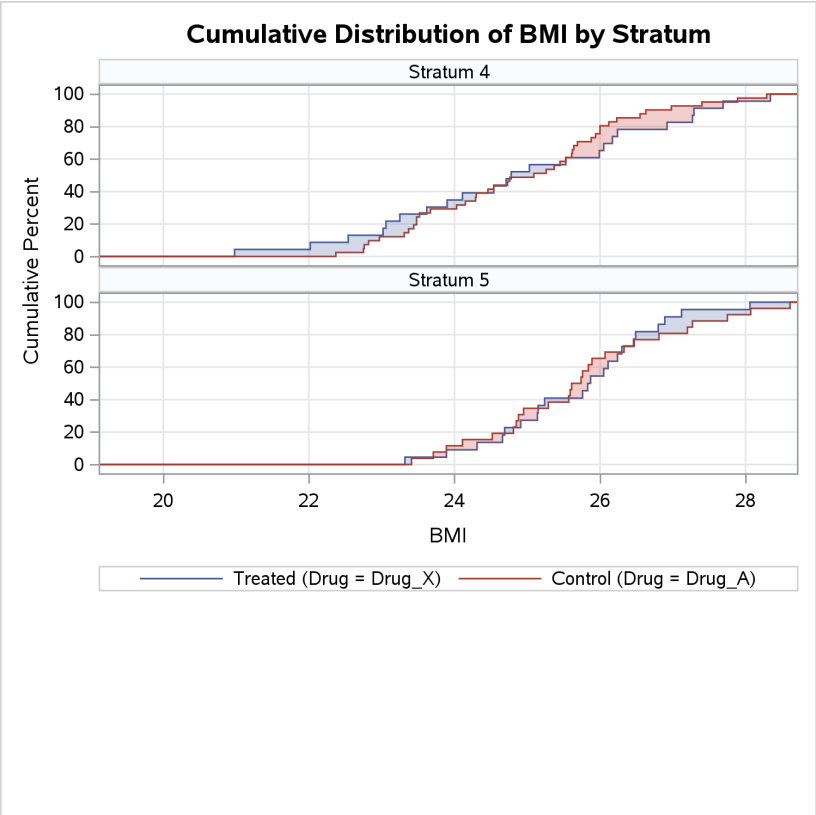

The plots show the differences in the distributions in strata. Here, the largest differences in the distributions of propensity score occur in Stratum 1 (lower values in the control group) and in Stratum 5 (higher values in the treated group)

Because stratification results in good balance for the variables in this example, as shown in Output 101.2.5 and Output 101.2.8, the stratified observations can be saved in an output data set for use in a subsequent outcome analysis.

In situations where you are not satisfied with the variable balance, you can do one or more of the following to improve the balance: you can select another set of variables to fit the propensity score model, you can modify the specification of the propensity score model (for instance, by using nonlinear terms for the continuous variables or by adding interactions), you can increase the number of strata, or you can choose another propensity score method (such as matching).

The OUT(OBS=ALL)=OutEx2 option in the OUTPUT statement creates an output data set named OutEx2 that contains all observations. The following statements list the first 10 observations in OutEx2, which are shown in Output 101.2.15:

```
proc print data=OutEx2(obs=10);
   var PatientID Drug Gender Age BMI _ps_ _Strata_;
run;
```

Output 101.2.15 Output Data Set with Strata

Obs	PatientID	Drug	Gender	Age	BMI	_PS_	_STRATA_
1	284	Drug_X	Male	29	22.02	0.36444	4
2	201	Drug_A	Male	45	26.68	0.22296	2
3	147	Drug_A	Male	42	21.84	0.11323	1
4	307	Drug_X	Male	38	22.71	0.19733	2
5	433	Drug_A	Male	31	22.76	0.35311	4
6	435	Drug_A	Male	43	26.86	0.27263	3
7	159	Drug_A	Female	45	25.47	0.14911	1
8	368	Drug_A	Female	49	24.28	0.07780	1
9	286	Drug_A	Male	31	23.31	0.38341	4
10	163	Drug_X	Female	39	25.34	0.24995	2

By default, the output data set includes the variable _PS_, which provides the propensity score, and the variable _STRATA_, which identifies the stratum.

After the responses for the trial are observed, they can be added to the data set OutEx2 as the starting point for an outcome analysis. Assuming that no other confounding variables are associated with both the response variable and the treatment group indicator Drug, you can estimate the treatment effect within each stratum and combine these estimates across strata to estimate the overall treatment effect (Stuart 2010, pp. 13–14). Note that the same stratum weights, as specified in the STRATUMWGT= option in the assessment, should be used in the outcome analysis.

Example 101.3: Optimal Variable Ratio Matching

(View the complete code for this example (psmchex3.sas) in the example repository.)

This example illustrates how you can perform optimal variable ratio matching of observations in a control group with observations in a treatment group, so that the matched observations can be used to estimate the treatment effect in a subsequent outcome analysis. The outcome analysis itself is not shown here.

The data for this example are observations on patients in a nonrandomized clinical trial. The trial and the Drugs data set that contains the patient information are described in the section "Getting Started: PSMATCH Procedure" on page 8372.

The following statements request optimal variable ratio matching to match each observation for patients in the treatment group with a variable number of observations for patients in the control group:

```
ods graphics on;
proc psmatch data=drugs region=treated(extend(distance=ps mult=one)=0.025);
   class Drug Gender;
   psmodel Drug(Treated='Drug_X')= Gender Age BMI;
   match distance=ps method=varratio(kmin=1 kmax=4) exact=(Gender) caliper=.
         nmatchmost=5;
   assess ps var=(Gender Age BMI)
          / stddev=pooled(allobs=no) plots(orient=vertical nodetails);
   id BMI;
   output out(obs=match)=OutEx3 matchid=_MatchID;
run;
```

The PSMODEL statement specifies the logistic regression model that creates the propensity score for each observation, which is the probability that the patient receives Drug_X. The CLASS statement specifies the classification variables in the model. The Drug variable is the binary treatment indicator variable, and TREATED='Drug_X' identifies Drug_X as the treated group. The Gender, Age, and BMI variables are included in the model because they are believed to be related to the assignment.

The PSMATCH procedure matches only those observations whose propensity scores lie in the support region that you specify with the REGION= option. Here the option REGION=TREATED requests that only those observations whose propensity scores lie in the region defined by the treated observations be used in matching. The suboption EXTEND(DISTANCE=PS MULT=ONE)=0.025 requests that this region be extended by 0.025 in propensity score.

The MATCH statement specifies the criteria for matching. The DISTANCE=PS option requests that the propensity score be used in computing differences between pairs of observations. The METHOD=VARRATIO(KMIN=1 KMAX=4) option requests optimal variable ratio matching of one to four control units to each unit in the treated group in order to minimize the total absolute difference in propensity scores across all matches.

The default average number of control units that are matched to each treated unit is computed as the mean of the KMIN= and KMAX= values, so an average of two control units are matched to each treated unit. The EXACT=GENDER option requests that the treated unit and its matched control unit have the same value of Gender. The CALIPER=. option ignores the caliper requirement for matching.

The "Data Information" table in Output 101.3.1 displays the numbers of observations in the treated and control groups, the lower and upper limits of the propensity scores for observations in the support region, and the numbers of observations in the treated and control groups that fall within the support region. Of the 373 observations in the control group, 366 fall within the support region. By definition, all 113 of the observations in the treated group fall within the support region.

Output 101.3.1 Data Information

The PSMATCH Procedure

Data Information	
Data Set	WORK.DRUGS
Output Data Set	WORK.OUTEX3
Treatment Variable	Drug
Treated Group	Drug_X
All Obs (Treated)	113
All Obs (Control)	373
Support Region	Extended Treated Group
Lower PS Support	0.035231
Upper PS Support	0.666148
Support Region Obs (Treated)	113
Support Region Obs (Control)	366

The "Propensity Score Information" table in Output 101.3.2 displays summary statistics by treatment group for all observations, for observations in the support region, and for matched observations. The three sets of summary statistics for the treated group are identical because REGION=TREATED is specified. Because the default WEIGHT=ATTWGT is used in the MATCH statement, the table also displays summary statistics of weighted matched observations by treatment group.

Output 101.3.2 Propensity Score Information

Propensity Score Information												
	Treated (Drug = Drug_X)						Control (Drug = Drug_A)					
Observations	N	Weight	Mean	Standard Deviation	Minimum	Maximum	N	Weight	Mean	Standard Deviation	Minimum	Maximum
All	113		0.3108	0.1325	0.0602	0.6411	373		0.2088	0.1320	0.0202	0.6858
Region	113		0.3108	0.1325	0.0602	0.6411	366		0.2087	0.1267	0.0371	0.6351
Matched	113		0.3108	0.1325	0.0602	0.6411	283		0.2450	0.1214	0.0510	0.6351
Weighted Matched	113	113.00	0.3108	0.1325	0.0602	0.6411	283	113.00	0.3006	0.1323	0.0510	0.6351

Propensity Score Information	
Observations	Treated - Control Mean Difference
All	0.1020
Region	0.1021
Matched	0.0658
Weighted Matched	0.0102

By default (or if you specify WEIGHT=ATTWGT in the MATCH statement), each treated unit receives a weight of 1 and each control unit receives a weight that is computed as the number of treated units divided by the number of control units in the matched set. For example, if three control units are matched to a treated unit in a matched set, then each control unit receives a weight of 1/3. Thus, the total weights of the weighted matched observations in the two treatment groups are the same. These weights are used to compute the standardized mean differences. For more information about these weights, see the section "Weighting after Matching" on page 8416.

The "Matching Information" table in Output 101.3.3 displays the matching criteria, the number of matched sets, the numbers of matched observations in the treated and control groups, and the total absolute difference in the propensity scores for all matches. Note that with an average of two and a half control units to each treated unit, 283 control units are matched.

Output 101.3.3 Matching Information

Matching Information	
Distance Metric	Propensity Score
Method	Optimal Variable Ratio Matching
Min Control/Treated Ratio	1
Max Control/Treated Ratio	4
Matched Sets	113
Matched Obs (Treated)	113
Matched Obs (Control)	283
Total Absolute Difference	3.644643

The ASSESS statement produces a table and a plot that summarize differences in the distributions of specified variables between treated and control groups for all observations, for observations in the support region, and for matched observations.

The "Standardized Mean Differences" table, shown in Output 101.3.4, displays standardized mean differences

in the variables between the treated and control groups for all observations, for observations in the support region, and for matched observations. As requested by the PS and VAR= options, the variables that are listed in the table are the propensity score and the variables Gender, Age, and BMI. For the binary classification variable (Gender), the difference is in the proportion of the first ordered level (Female).

Output 101.3.4 Standardized Mean Differences

The PSMATCH Procedure

Standardized Mean Differences (Treated - Control)						
Variable	Observations	Mean Difference	Standard Deviation	Standardized Difference	Percent Reduction	Variance Ratio
Prop Score	All	0.10197	0.132218	0.77124		1.0076
	Region	0.10210	0.129634	0.78757	0.00	1.0924
	Matched	0.06579	0.127033	0.51787	32.85	1.1915
	Weighted Matched	0.01019	0.132397	0.07696	90.02	1.0021
Age	All	-4.09509	6.079104	-0.67363		0.7076
	Region	-3.98262	6.002096	-0.66354	1.50	0.7393
	Matched	-2.19910	5.693934	-0.38622	42.67	0.8951
	Weighted Matched	-0.29720	5.612599	-0.05295	92.14	0.9460
BMI	All	0.73930	1.923178	0.38441		0.8854
	Region	0.72871	1.902421	0.38305	0.36	0.9227
	Matched	0.41500	1.879869	0.22076	42.57	0.9665
	Weighted Matched	0.09697	1.822647	0.05320	86.16	1.0957
Gender	All	-0.02482	0.496925	-0.04994		0.9892
	Region	-0.02266	0.496832	-0.04560	8.69	0.9899
	Matched	-0.02574	0.496963	-0.05179	0.00	0.9889
	Weighted Matched	0.00000	0.495575	0.00000	100.00	1.0000

Note that a zero percentage reduction is displayed for Gender in the matched observation because its standardized mean difference (0.05179 in absolute value) is larger than the standardized mean difference of all observations (0.04994 in absolute value).

The standardized mean differences are significantly reduced in the matched observations, the standardized mean differences are less than the recommended upper limit of 0.25, and the variance ratios between the two groups are within the recommended range of 0.5 to 2.

The “Standardized Mean Differences” plot is shown in Output 101.3.5.

Output 101.3.5 Standardized Mean Differences Plot

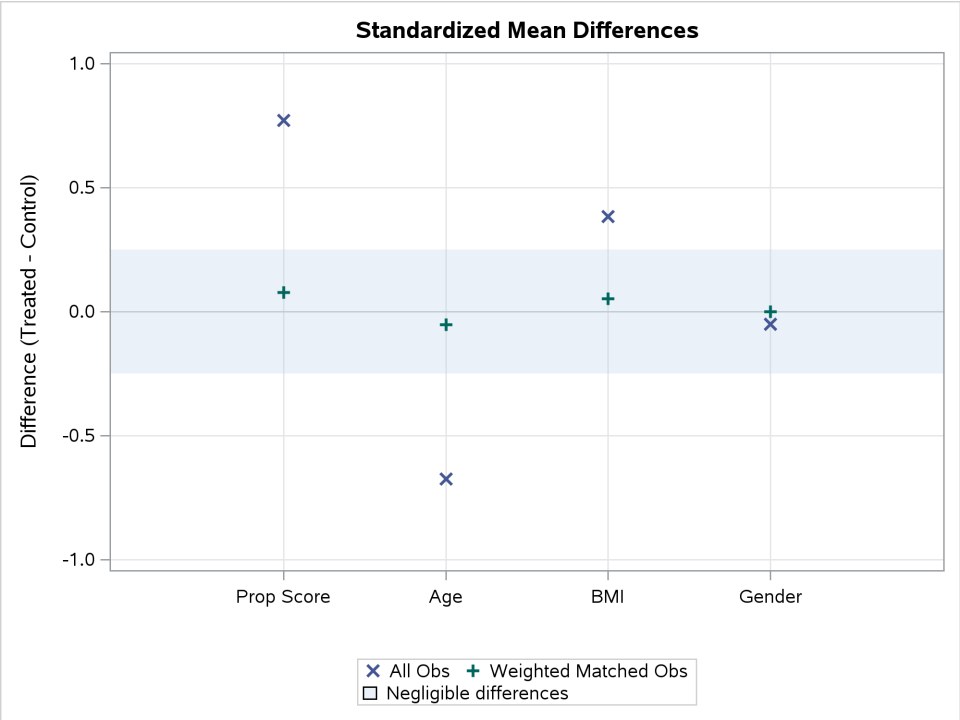

The "Standardized Mean Differences" plot displays the standardized mean differences that are listed in the "Standardized Mean Differences" table in Output 101.3.4. When you specify the ORIENT=VERTICAL option, the standardized mean differences are plotted on the vertical axis. All differences for the matched observations are within the recommended limits of –0.25 and 0.25, which are indicated by the shaded area.

The NMATCHMOST=5 option requests the "Observations with Most Matches" table, which is shown in Output 101.3.6, and displays the five observations that have the most matches in the treated and control groups.

Output 101.3.6 Observations with the Most Matches

Observations with Most Matches		
Treated (Drug = Drug_X)		
Observation	BMI	Matched Control
202	20.75	4
479	22.22	4
250	23.96	4
227	21.11	4
274	24.17	4

Because matching results in good balance for the variables in this example, the matched observations can be saved in an output data set for use in a subsequent outcome analysis.

In situations where you are not satisfied with the variable balance, you can do one or more of the following to improve the balance: you can select another set of variables to fit the propensity score model, you can modify

the matching criteria, or you can choose another matching method.

The OUT(OBS=MATCH)=OutEx3 option in the OUTPUT statement creates an output data set, OutEx3, that contains the matched observations. The following statements list the observations in the first two matched sets, as shown in Output 101.3.7:

```
proc sort data=OutEx3 out=OutEx3a;
   by _MatchID;
run;

proc print data=OutEx3a(obs=10);
   var PatientID Drug Gender Age BMI _PS_ _MATCHWGT_ _MatchID;
run;
```

Output 101.3.7 Output Data Set with Optimal Variable Ratio Matches

Obs	PatientID	Drug	Gender	Age	BMI	_PS_	_MATCHWGT_	_MatchID
1	141	Drug_A	Female	43	20.55	0.064010	0.25	1
2	213	Drug_A	Female	49	23.24	0.061866	0.25	1
3	89	Drug_X	Female	44	20.75	0.060231	1.00	1
4	311	Drug_A	Female	49	22.80	0.056086	0.25	1
5	104	Drug_A	Female	46	20.95	0.050951	0.25	1
6	137	Drug_A	Female	45	22.04	0.072149	0.25	2
7	158	Drug_A	Female	48	23.64	0.075028	0.25	2
8	245	Drug_A	Female	52	25.32	0.071559	0.25	2
9	40	Drug_A	Female	42	20.65	0.072654	0.25	2
10	323	Drug_X	Female	46	22.22	0.067625	1.00	2

By default, the output data set includes the variable _PS_ (which provides the propensity score) and the variable _MATCHWGT_ (which provides matched observation weights). The weight for each treated unit is 1. Because METHOD=VARRATIO(KMIN=1 KMAX=4) is specified in the MATCH statement, one, two, three, or four control units are matched to each treated unit; so the weight for each matched control unit is 1, 1/2, 1/3, or 1/4. The MATCHID=_MatchID option creates a variable named _MatchID that identifies the matched sets of observations.

After the responses for the trial are observed, they can be added to the data set OutEx3 as the starting point for an outcome analysis. Assuming that no other confounding variables are associated with both the response variable and the treatment group indicator Drug, you can estimate the treatment effect from the matched observations by performing a weighted version of the outcome analysis that you would have used to estimate the treatment effect if the original data set had resulted from a randomized trial.

Example 101.4: Greedy Nearest Neighbor Matching

(View the complete code for this example (psmchex4.sas) in the example repository.)

This example illustrates how you can perform greedy matching of observations in a control group with observations in a treatment group, so that the matched observations can be used to estimate the treatment effect in a subsequent outcome analysis. An outcome analysis is not shown here but is discussed in Example 101.5.

At the completion of a school year, a school administrator asks whether taking a music class caused an improvement in the grade point averages (GPAs) of students. The reasoning behind this question is that learning to read and perform music might improve general reading ability, concentration, and memory.

The data set School contains information about students that is available at the end of the school year. StudentID is the student identification number, Music indicates whether the student took a music class, Gender provides the gender of the student, and Absence is the percentage of absences. Output 101.4.1 lists the first 10 observations.

Output 101.4.1 Input School Data Set

Obs	StudentID	Music	Gender	Absence
1	18	No	Female	3.71
2	61	No	Male	2.08
3	95	No	Female	2.54
4	41	No	Male	3.01
5	19	Yes	Female	0.08
6	51	No	Female	1.20
7	110	No	Male	2.21
8	87	No	Female	2.30
9	103	No	Female	3.08
10	175	No	Female	1.12

In this example, the outcome variable GPA for the students was available at the end of the year. However, following recommended practice (Stuart 2010, p. 2), the values of GPA are not used in the propensity score analysis that is described in this example. Instead, the variable GPA is reserved for the outcome analysis, which is carried out on the output data set that is created by the PSMATCH procedure after it has been augmented with the values of GPA. See Example 101.5 for an illustration of an outcome analysis.

The following statements request greedy nearest neighbor matching to sequentially match each observation for students in the treatment group (those who took music) with one observation for students in the control group (those who did not take music):

```
ods graphics on;
proc psmatch data=School region=treated;
   class Music Gender;
   psmodel Music(Treated='Yes')= Gender Absence;
   match distance=lps method=greedy(k=1) exact=Gender caliper=0.5
         weight=none;
   assess lps var=(Gender Absence)
          / stddev=pooled(allobs=no) stdbinvar=no plots(nodetails)=all;
   output out(obs=match)=OutEx4 matchid=_MatchID;
run;
```

The PSMODEL statement specifies the logistic regression model that creates the propensity score for each student, which is the probability that the student enrolled in the music class. The Music variable is the binary treatment indicator variable and TREATED='Yes' identifies Yes as the treated group. The Gender and Absence variables are included in the model because they are believed to be related to enrolling in the music class. The CLASS statement specifies the classification variables.

The PSMATCH procedure matches only those observations whose propensity scores lie in the support region that you specify in the REGION= option. Here the REGION=TREATED option requests that only those observations whose propensity scores lie in the region defined by the treated observations be used in matching. By default, the region is extended by 0.25 times the pooled estimate of the common standard deviation of the logits of the propensity scores.

The MATCH statement requests matching and specifies the criteria for matching. The DISTANCE=LPS option (which is the default) requests that the logit of the propensity score be used in computing differences between pairs of observations. The METHOD=GREEDY(K=1) option requests greedy nearest neighbor matching in which one control unit is matched with each unit in the treated group; this produces the smallest within-pair difference among all available pairs with this treated unit. The EXACT=GENDER option requests that the treated unit and its matched control unit have the same value of the Gender variable. The CALIPER=0.5 option specifies the caliper requirement for matching. Units are matched only if the difference in the logits of the propensity scores for pairs of units from the two groups is less than or equal to 0.5 times the pooled estimate of the standard deviation.

The "Data Information" table, shown in Output 101.4.2, displays the numbers of observations in the treated and control groups, the lower and upper limits of the propensity scores for observations in the support region, and the numbers of observations in the treated and control groups that fall within the support region. Of the 140 observations in the control group, 132 fall within the support region.

Output 101.4.2 Data Information

The PSMATCH Procedure

Data Information	
Data Set	WORK.SCHOOL
Output Data Set	WORK.OUTEX4
Treatment Variable	Music
Treated Group	Yes
All Obs (Treated)	60
All Obs (Control)	140
Support Region	Extended Treated Group
Lower PS Support	0.079911
Upper PS Support	0.530945
Support Region Obs (Treated)	60
Support Region Obs (Control)	132

The "Propensity Score Information" table, shown in Output 101.4.3, displays summary statistics for the treatment and control groups. These statistics are computed for all observations, for observations in the support region, and for matched observations. The three sets of statistics are identical for the treated group because REGION=TREATED is specified.

Output 101.4.3 Propensity Score Information

Propensity Score Information											
	Treated (Music = Yes)					Control (Music = No)					Treated - Control
Observations	N	Mean	Standard Deviation	Minimum	Maximum	N	Mean	Standard Deviation	Minimum	Maximum	Mean Difference
All	60	0.3472	0.0963	0.0928	0.4902	140	0.2798	0.1251	0.0263	0.4887	0.0675
Region	60	0.3472	0.0963	0.0928	0.4902	132	0.2940	0.1141	0.0832	0.4887	0.0533
Matched	60	0.3472	0.0963	0.0928	0.4902	60	0.3402	0.0986	0.0928	0.4887	0.0070

The "Matching Information" table in Output 101.4.4 displays the matching criteria, the number of matched sets, the numbers of matched observations in the treated and control groups, and the total absolute difference in the logits of the propensity scores for all matches.

Output 101.4.4 Matching Information

Matching Information	
Distance Metric	Logit of Propensity Score
Method	Greedy Matching
Control/Treated Ratio	1
Order	Descending
Caliper (Logit PS)	0.326669
Matched Sets	60
Matched Obs (Treated)	60
Matched Obs (Control)	60
Total Absolute Difference	2.946274

The ASSESS statement produces tables and plots that summarize differences in the distributions of specified variables between treated and control groups for all observations, for observations in the support region, and for matched observations. You can use these results to assess how well matching achieves a balance in the distributions of these variables. As requested by the LPS and VAR= options, the variables are the logit of the propensity score and the covariates Gender and Absence. The WEIGHT=NONE option suppresses the display of differences for weighted matched observations. When PROC PSMATCH matches one control unit to each treated unit, it assigns a weight of 1 for all matched treated and control units, so the results are identical for weighted matched observations and matched observations.

The "Standardized Mean Differences" table, shown in Output 101.4.5, displays standardized mean differences in the variables between the treated and control groups, which are computed for all observations, for observations in the support region, and for matched observations. For the binary classification variable (Gender), the computed difference is in the proportion of the first ordered level (Female).

Output 101.4.5 Standardized Mean Differences

The PSMATCH Procedure

Standardized Mean Differences (Treated - Control)						
Variable	Observations	Mean Difference	Standard Deviation	Standardized Difference	Percent Reduction	Variance Ratio
Logit Prop Score	All	0.40769	0.653338	0.62402		0.3810
	Region	0.28487	0.556440	0.51195	17.96	0.6138
	Matched	0.03305	0.489082	0.06758	89.17	0.9698
Absence	All	-0.69807	1.136767	-0.61409		0.3550
	Region	-0.48623	0.973273	-0.49958	18.65	0.5561
	Matched	-0.05833	0.836446	-0.06974	88.64	0.9374
Gender	All	-0.04524				
	Region	-0.03485			22.97	
	Matched	0.00000			100.00	

The standardized mean differences are significantly reduced in the matched observations, and the largest of these differences is 0.07015 in absolute value, which is less than the recommended upper limit of 0.25 (Rubin 2001, p. 174; Stuart 2010, p. 11). The treated-to-control variance ratios are 0.9380 and 0.9701 in the matched observations, which are within the recommended range of 0.5 to 2. Because EXACT=GENDER is specified in the MATCH statement, the standardized mean difference for Gender is 0 in the matched observations.

When you specify PLOTS=ALL, the PSMATCH procedure creates all applicable plots. Output 101.4.6 displays a plot of the standardized mean differences in Gender, Absence, and the logit of the propensity score for all observations and matched observations. Because the NODETAILS option is specified, the comparison of observations in the support region is not displayed except for the cloud plots.

Output 101.4.6 Standardized Mean Differences Plot

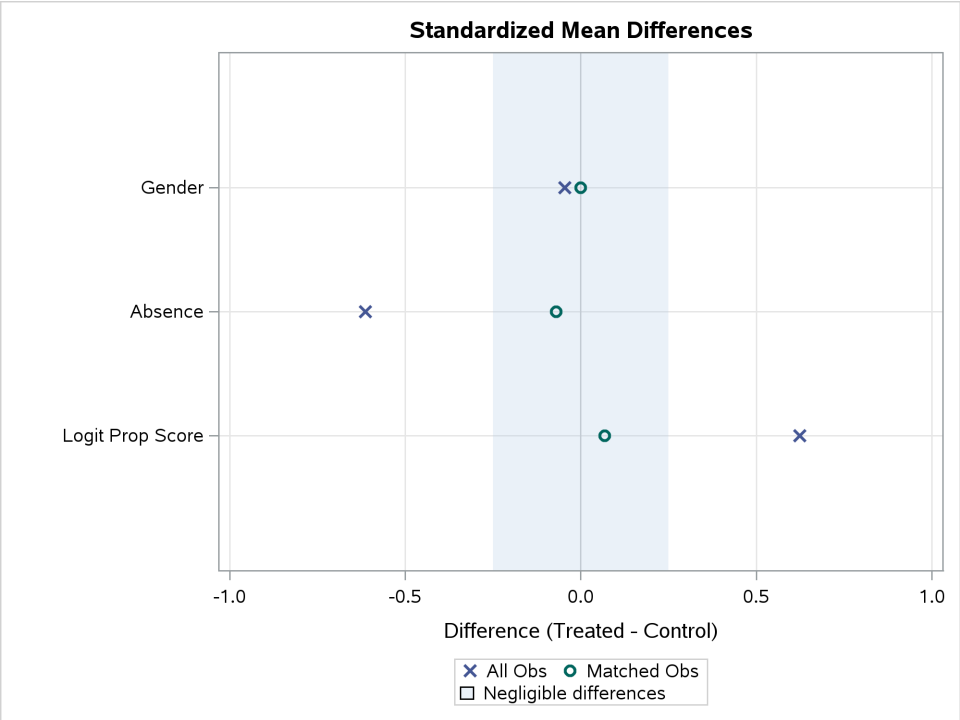

Output 101.4.7 displays box plots that compare the distributions of the logit propensity score for units in the treated and control groups, based on all observations, on observations in the support region, and on matched observations. Note that the two distributions are well balanced for matched observations.

Output 101.4.7 LPS Box Plot

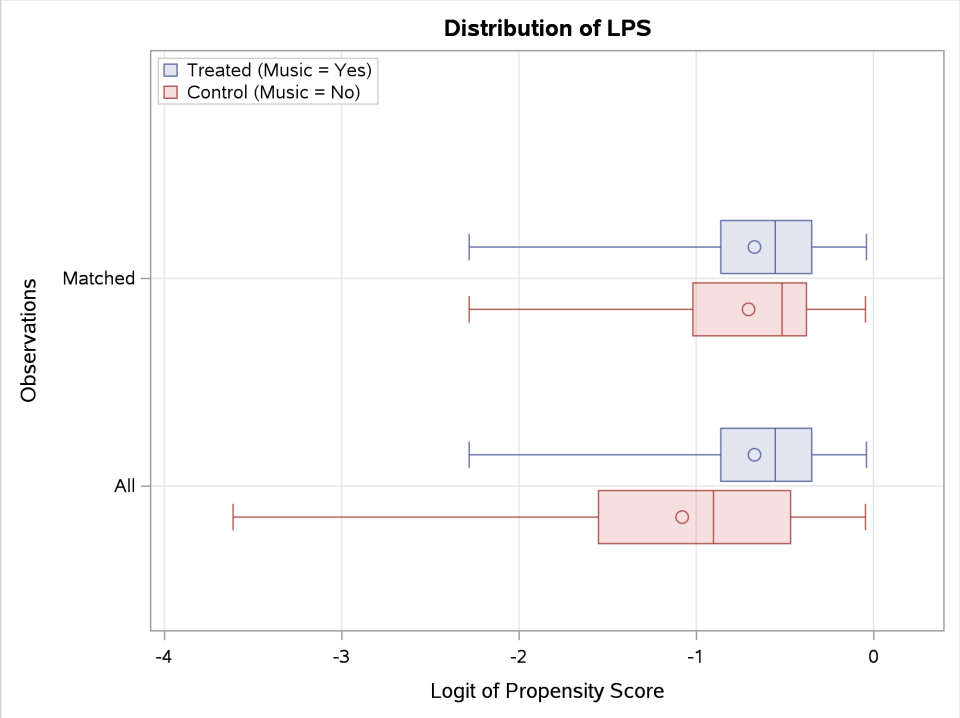

Output 101.4.8 displays box plots that compare the distributions of Absence for units in the treated and control groups, based on all observations and on matched observations. Again, note that the two distributions are well balanced for matched observations.

Output 101.4.8 Absence Box Plot

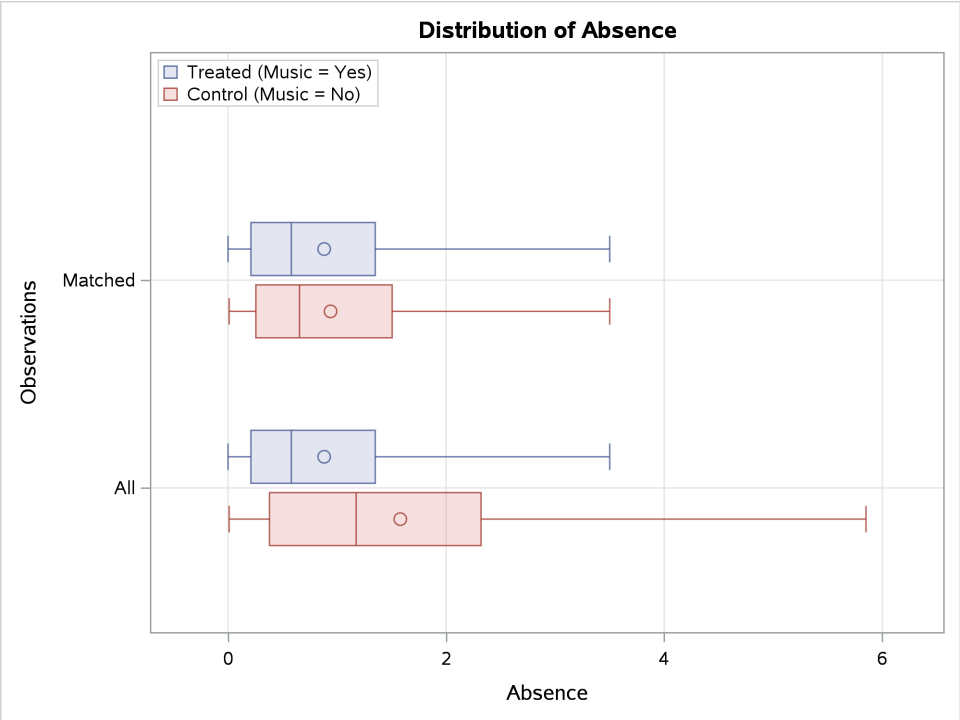

Output 101.4.9 displays bar charts that compare the distributions of Gender for units in the treated and control groups, based on all observations and on matched observations. Again, note that the two distributions are well balanced for matched observations.

Output 101.4.9 Gender Bar Chart

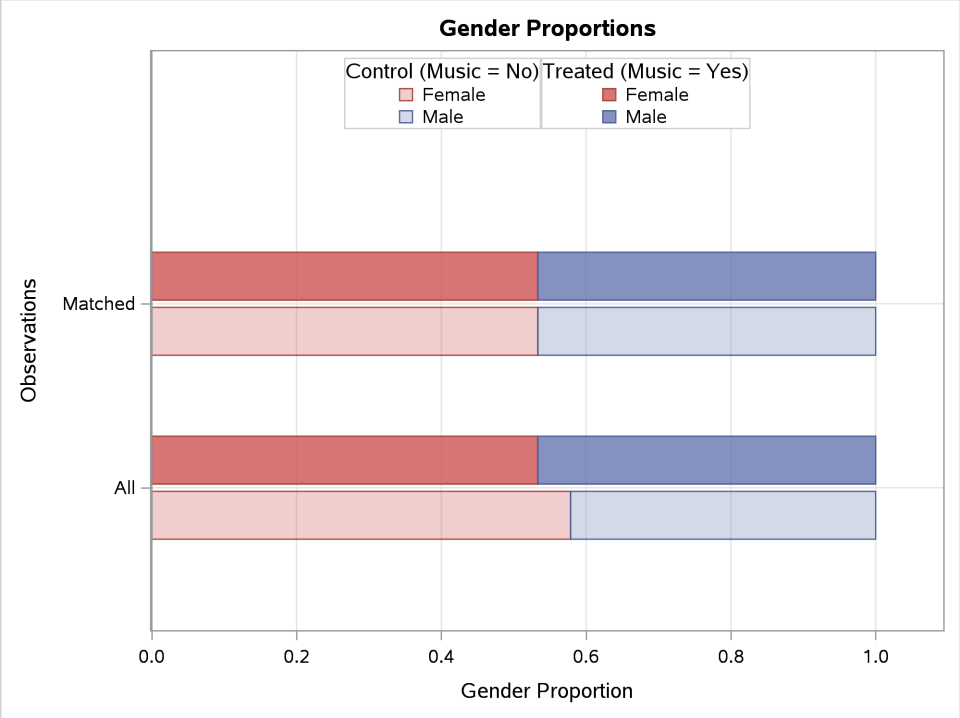

Output 101.4.10 displays a cumulative distribution function (CDF) plot that compares the CDFs of the logit of the propensity score (LPS) for observations in the treated and control groups, based on all observations and on matched observations.

Output 101.4.10 CDF Plot for Logit Propensity Score

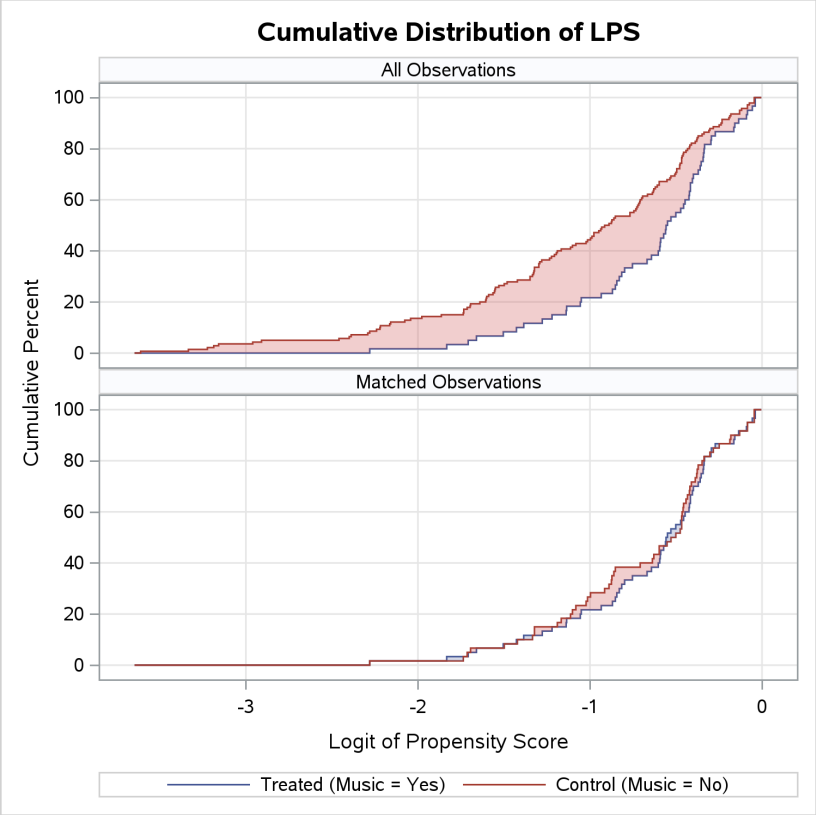

Output 101.4.11 displays a plot that compares the CDFs of Absence for observations in the treated and control groups.

Output 101.4.11 CDF Plot for Absence

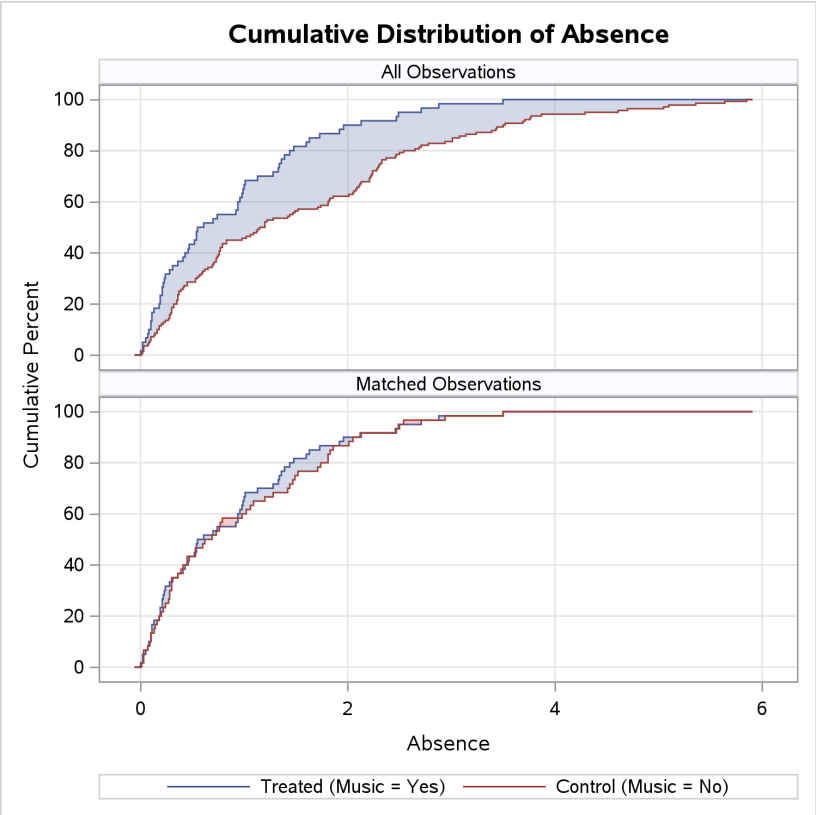

Output 101.4.12 displays a cloud plot that compares the values of the logit of the propensity score (LPS) for observations in the treated and control groups, based on all observations and on matched observations. The points are jittered in the vertical direction to avoid overplotting.

Output 101.4.12 LPS Cloud Plot

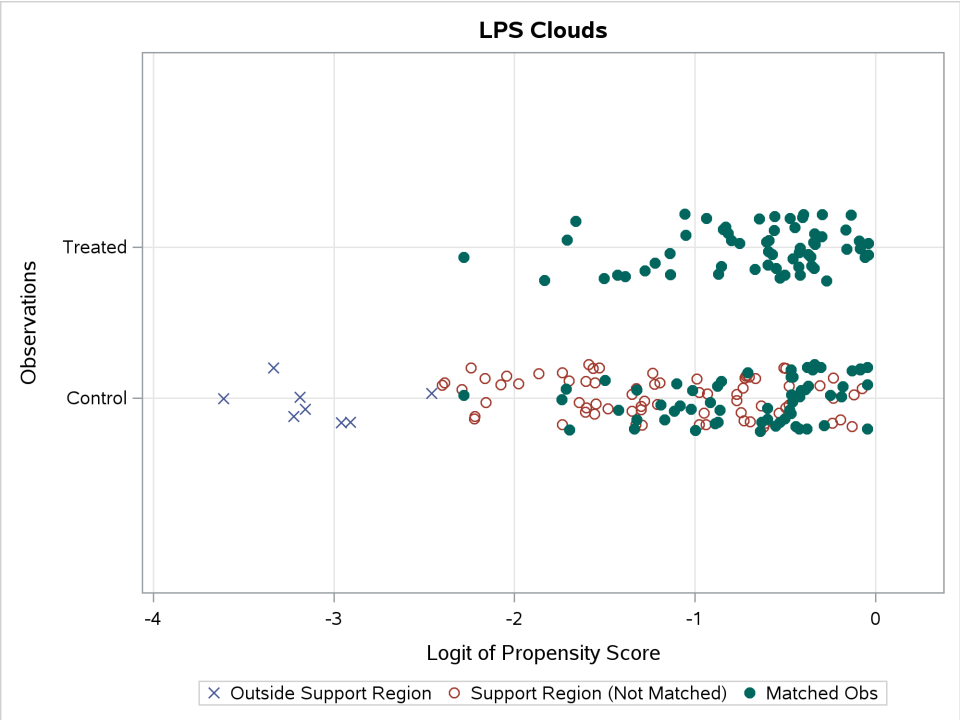

Output 101.4.13 displays a cloud plot that compares the values of Absence for observations in the treated and control groups.

Output 101.4.13 Absence Cloud Plot

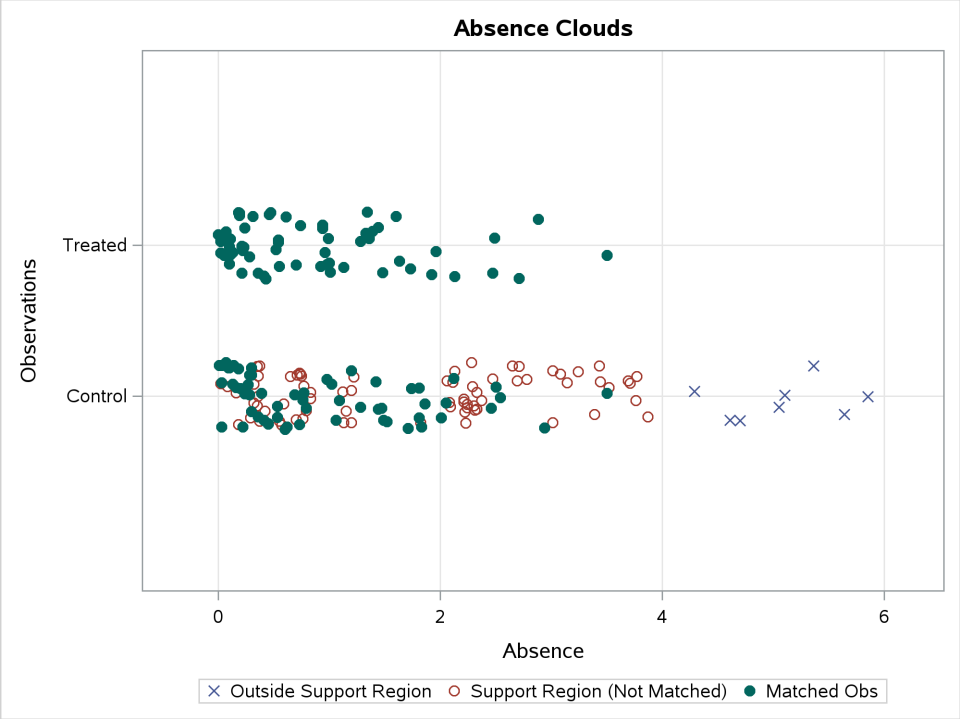

Note that the NODETAILS option does not apply to the cloud plots.

Because matching results in good balance for the variables in this example, the matched observations can be saved in an output data set for use in a subsequent outcome analysis.

In situations where you are not satisfied with the variable balance, you can do one or more of the following to improve the balance: you can select another set of variables to fit the propensity score model, you can modify the matching criteria, or you can choose another matching method.

The OUT(OBS=MATCH)=OutEx4 option in the OUTPUT statement creates an output data set, OutEx4, that contains the matched observations. The following statements list the observations in the first five matched sets, as shown in Output 101.4.14:

```
proc sort data=OutEx4 out=OutEx4a;
   by _MatchID;
run;

proc print data=OutEx4a(obs=10);
   var StudentID Music Gender Absence _PS_ _MATCHWGT_ _MatchID;
run;
```

Output 101.4.14 Output Data Set with Matching Numbers

Obs	StudentID	Music	Gender	Absence	_PS_	_MATCHWGT_	_MatchID
1	156	Yes	Male	0.02	0.49015	1	1
2	173	No	Male	0.03	0.48874	1	1
3	142	Yes	Male	0.02	0.49015	1	2
4	105	No	Male	0.03	0.48874	1	2
5	64	No	Male	0.03	0.48874	1	3
6	98	Yes	Male	0.05	0.48590	1	3
7	89	No	Male	0.10	0.47883	1	4
8	182	Yes	Male	0.10	0.47883	1	4
9	30	No	Male	0.10	0.47883	1	5
10	115	Yes	Male	0.11	0.47742	1	5

By default, the output data set includes the variable _PS_ (which provides the propensity score) and the variable _MATCHWGT_ (which provides matched observation weights). The MATCHID=_MatchID option creates a variable named _MatchID that identifies the matched sets of observations.

If you assume that no other confounding variables are associated with both the GPA and the music class indicator Music, you can add the GPAs for the students to the data set OutEx4 and perform an outcome analysis on that data set to estimate the effect of music class.

Example 101.5: Outcome Analysis after Matching

(View the complete code for this example (psmchex5.sas) in the example repository.)

This example illustrates how you can carry out an outcome analysis of observations that have been matched as the result of a propensity score analysis.

The data set School, described in Example 101.4, contains information about students that is available at the end of the school year.

The data set Grades contains information about the student grades. StudentID is the student identification number, and GPA is the GPA for the student.

The following statements combine the two data sets and list the 10 observations in the combined School-Grades data set, as shown in Output 101.4.14:

```
proc sort data=School out=School1;
   by StudentID;
run;

proc sort data=Grades out=Grades1;
   by StudentID;
run;

data SchoolGrades;
   merge School1 Grades1;
   by StudentID;
run;

proc print data=SchoolGrades(obs=10);
   var StudentID Music Gender Absence GPA;
run;
```

Output 101.5.1 SchoolGrades Data Set

Obs	StudentID	Music	Gender	Absence	GPA
1	1	Yes	Male	1.39	3.99
2	2	No	Female	0.71	3.94
3	3	No	Male	4.29	3.32
4	4	Yes	Female	2.49	3.78
5	5	No	Female	0.02	4.10
6	6	No	Female	0.32	4.12
7	7	No	Female	0.20	4.28
8	8	Yes	Female	0.21	4.40
9	9	No	Female	0.53	3.96
10	10	No	Male	2.78	3.14

For comparison with the outcome analyses that are performed on matched observations later in this example, the following steps perform a t test for the effect of music class on GPA using the original (unmatched) data:

```
proc ttest data=SchoolGrades;
   class Music;
   var GPA;
run;
```

The table in Output 101.5.2 shows that the effect of music class is significantly different from 0.

Output 101.5.2 *t* Test for Difference

The TTEST Procedure

Variable: GPA

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	198	-3.43	0.0007
Satterthwaite	Unequal	148.02	-3.85	0.0002

Although the *t* test shows a significant effect, this effect might be related to the student's gender or absence record. The following regression analysis controls for these effects:

```
proc glm data=SchoolGrades;
   class music(ref='No') gender;
   model GPA= music gender absence / solution;
run;
```

The parameter estimates table in Output 101.5.3 shows that the effect of music class has a *p*-value of 0.1089, which is larger than the *p*-values in Output 101.5.2.

Output 101.5.3 Music Class Effect Estimate

The GLM Procedure

Dependent Variable: GPA

Parameter		Estimate		Standard Error	t Value	Pr > |t|
Intercept		3.903163559	B	0.03913163	99.74	<.0001
Music	**Yes**	0.066558065	B	0.04132470	1.61	0.1089
Music	**No**	0.000000000	B	.	.	.
Gender	**Female**	0.059468770	B	0.03699850	1.61	0.1096
Gender	**Male**	0.000000000	B	.	.	.
Absence		-0.152888978		0.01483823	-10.30	<.0001

However, the regression adjustment requires a sufficient overlap between the covariate distributions for students who took music and students who did not take music. You can ensure a sufficient covariate overlap by performing a propensity score analysis that uses greedy nearest neighbor matching.

The following statements request this analysis:

```
proc psmatch data=School region=treated;
   class Music Gender;
   psmodel Music(Treated='Yes')= Gender Absence;
   match distance=lps method=greedy(k=1) exact=Gender caliper=0.5
```

```
        weight=none;
    output out(obs=match)=OutEx4 matchid=_MatchID;
run;
```

These statements are identical to the PROC PSMATCH statements in Example 101.4, except that the ASSESS statement is not used here.

The OUT(OBS=MATCH)=OutEx4 option creates an output data set, OutEx4, that contains the matched observations. Output 101.5.4 displays the observations in the first four matched sets, as shown in Output 101.5.4:

Output 101.5.4 Output Data Set with Matching Numbers

Obs	StudentID	Music	Gender	Absence	_PS_	_MATCHWGT_	_MatchID
1	156	Yes	Male	0.02	0.49015	1	1
2	173	No	Male	0.03	0.48874	1	1
3	142	Yes	Male	0.02	0.49015	1	2
4	105	No	Male	0.03	0.48874	1	2
5	64	No	Male	0.03	0.48874	1	3
6	98	Yes	Male	0.05	0.48590	1	3
7	89	No	Male	0.10	0.47883	1	4
8	182	Yes	Male	0.10	0.47883	1	4

By default, the output data set includes the variable _PS_ (which provides the propensity score) and the variable _MATCHWGT_ (which provides matched observation weights). The MATCHID=_MatchID option creates a variable named _MatchID that identifies the matched sets of observations. Because the data set OutEx4 contains only the matched observations for a one-to-one matching, all the matched observation weights are 1 and can therefore be omitted from the outcome analysis. For more information about the matched observation weights, see the section "Weighting after Matching" on page 8416.

If you assume that no other confounding variables are associated with both GPA and Music, you can add the GPAs for the students to the data set OutEx4 and perform an outcome analysis of GPA on that data set to estimate the effect of the music class. The following statements combine the two data sets:

```
proc sort data=OutEx4 out=OutEx4b;
   by StudentID;
run;

data OutEx4Grades;
   merge OutEx4b Grades1;
   by StudentID;
run;
```

The following statements use a *t* test to estimate the effect of music class from the matched observations:

```
proc ttest data=OutEx4Grades;
   class Music;
   var GPA;
run;
```

The *t* test in Output 101.5.5 has a a large *p*-value of 0.4974, which shows that the effect of the music class is not significant.

Output 101.5.5 *t* Test for Difference

The TTEST Procedure

Variable: GPA

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	118	-0.68	0.4974
Satterthwaite	Unequal	117.43	-0.68	0.4974

The following regression analysis of the matched observations controls for the effects of gender and absence:

```
proc glm data=OutEx4Grades;
   class music(ref='No') gender;
   model GPA= music gender absence / solution;
run;
```

The "Parameter Estimates" table in Output 101.5.6 shows that the effect of music class has a large *p*-value of 0.5647.

Output 101.5.6 Music Class Effect Estimate

The GLM Procedure

Dependent Variable: GPA

Parameter		Estimate		Standard Error	t Value	Pr > |t|
Intercept		3.919862471	B	0.04782202	81.97	<.0001
Music	**Yes**	0.026056453	B	0.04511875	0.58	0.5647
Music	**No**	0.000000000	B	.	.	.
Gender	**Female**	0.052862308	B	0.04521048	1.17	0.2447
Gender	**Male**	0.000000000	B	.	.	.
Absence		-0.121889379		0.02719279	-4.48	<.0001

In summary, the outcome analyses that are based on matched observations reach a different conclusion than the outcome analyses that are based on the original data.

Example 101.6: Matching with Replacement

(View the complete code for this example (psmchex6.sas) in the example repository.)

This example illustrates how you can perform matching with replacement of observations in a control group with observations in a treatment group, so that the matched observations can be used to estimate the treatment effect in a subsequent outcome analysis.

The data for this example are observations on students in a school. The School data set, which contains information about the students, is described in Example 101.4. The following statements request matching with replacement to match observations for students in the treatment group with observations for students in the control group:

```
ods graphics on;
proc psmatch data=School region=allobs(psmin=0.05);
   class Music Gender;
   psmodel Music(Treated='Yes')= Gender Absence;
   match method=replace(k=1) distance=ps exact=Gender caliper=.
         nmatchmost=6;
   assess ps var=(Gender Absence);
   output out(obs=match)=outex6 matchid=_MatchID;
run;
```

The PSMODEL statement specifies the logistic regression model that creates the propensity score for each student, which is the probability that the student enrolled in a music class. The CLASS statement specifies the classification variables. The Music variable is the binary treatment indicator variable, and TREATED='Yes' identifies Yes as the treated group. The Gender and Absence variables are included in the model because they are believed to be related to enrolling in the music class.

The PSMATCH procedure matches only those observations whose propensity scores lie in the support region that you specify with the REGION= option. Here the REGION=ALLOBS(PSMIN=0.05) option requests that all available observations whose propensity scores are greater than or equal to 0.05 be used for matching.

The MATCH statement requests matching and specifies the criteria for matching. The DISTANCE=PS option requests that the propensity score be used to compute differences between pairs of observations. The METHOD=REPLACE(K=1) option requests matching with replacement in which each treated unit is matched to the closest control unit.

The EXACT=GENDER option requests that the treated unit and its matched control unit have the same value of the Gender variable. The CALIPER=. option ignores the caliper requirement for matching.

The "Data Information" table in Output 101.6.1 displays the numbers of observations in the treated and control groups, the lower and upper limits of the propensity scores for observations in the support region, and the numbers of observations in the treated and control groups that fall within the support region. Of the 140 observations in the control group, 134 fall within the support region.

Output 101.6.1 Data Information

The PSMATCH Procedure

Data Information	
Data Set	WORK.SCHOOL
Output Data Set	WORK.OUTEX6
Treatment Variable	Music
Treated Group	Yes
All Obs (Treated)	60
All Obs (Control)	140
Support Region	PS Bounded Obs
Lower PS Support	0.05
Upper PS Support	0.490152
Support Region Obs (Treated)	60
Support Region Obs (Control)	134

The "Propensity Score Information" table in Output 101.6.2 displays summary statistics by treatment group for all observations, for observations in the support region, and for matched observations. Because the default

WEIGHT=ATTWGT is specified in the PSWEIGHT statement, the table also displays summary statistics for weighted matched observations.

Output 101.6.2 Propensity Score Information

Propensity Score Information												
	Treated (Music = Yes)						Control (Music = No)					
Observations	N	Weight	Mean	Standard Deviation	Minimum	Maximum	N	Weight	Mean	Standard Deviation	Minimum	Maximum
All	60		0.3472	0.0963	0.0928	0.4902	140		0.2798	0.1251	0.0263	0.4887
Region	60		0.3472	0.0963	0.0928	0.4902	134		0.2906	0.1166	0.0517	0.4887
Matched	60		0.3472	0.0963	0.0928	0.4902	41		0.3350	0.1036	0.0928	0.4887
Weighted Matched	60	60.00	0.3472	0.0963	0.0928	0.4902	41	60.00	0.3458	0.0959	0.0928	0.4887

Propensity Score Information	
Observations	Treated - Control Mean Difference
All	0.0675
Region	0.0567
Matched	0.0122
Weighted Matched	0.0014

Note that the number of matched control units (41) is less than the number of matched treated units (60). When matching is done with replacement, a control unit can be matched with more than one treated unit.

The "Matching Information" table in Output 101.6.3 displays the matching criteria, the number of matched sets, the numbers of matched observations in the treated and control groups, and the total absolute difference in the propensity scores for all matches. In this example, 41 control units are matched to 60 treated units.

Output 101.6.3 Matching Information

Matching Information	
Distance Metric	Propensity Score
Method	Replacement Matching
Control/Treated Ratio	1
Matched Sets	41
Matched Obs (Treated)	60
Matched Obs (Control)	41
Total Absolute Difference	0.263828

The ASSESS statement produces tables and plots that summarize differences in the distributions of the specified variables between treated and control groups for all observations, for observations in the support region, and for matched observations. You can use these results to assess how well the matching achieves a balance in the distributions of these variables. As requested by the PS and VAR= options, the variables are the propensity score and the covariates Gender and Absence.

The "Standardized Mean Differences" table displays standardized mean differences in the variables between the treated and control groups, as shown in Output 101.6.4.

Output 101.6.4 Standardized Mean Differences

The PSMATCH Procedure

Standardized Mean Differences (Treated - Control)						
Variable	**Observations**	**Mean Difference**	**Standard Deviation**	**Standardized Difference**	**Percent Reduction**	**Variance Ratio**
Prop Score	**All**	0.06749	0.111638	0.60451		0.5930
	Region	0.05667		0.50759	16.03	0.6824
	Matched	0.01220		0.10926	81.93	0.8652
	Weighted Matched	0.00141		0.01261	97.91	1.0083
Absence	**All**	-0.69807	1.136767	-0.61409		0.3550
	Region	-0.53223		-0.46820	23.76	0.5052
	Matched	-0.08899		-0.07828	87.25	0.8202
	Weighted Matched	-0.01033		-0.00909	98.52	1.0049
Gender	**All**	-0.04524	0.496344	-0.09114		1.0208
	Region	-0.03383		-0.06816	25.22	1.0138
	Matched	-0.05203		-0.10483	0.00	1.0254
	Weighted Matched	0.00000		0.00000	100.00	1.0000
Standard deviation of All observations used to compute standardized differences						

Note that a zero percentage reduction is displayed for Gender in the matched observation because its standardized mean difference (0.10483, in absolute value) is larger than the standardized mean difference of all observations (0.09114 in absolute value).

The PSMATCH procedure displays a standardized differences plot, shown in Output 101.6.5, for the variables that are specified in the ASSESS statement.

Output 101.6.5 Standardized Mean Differences Plot

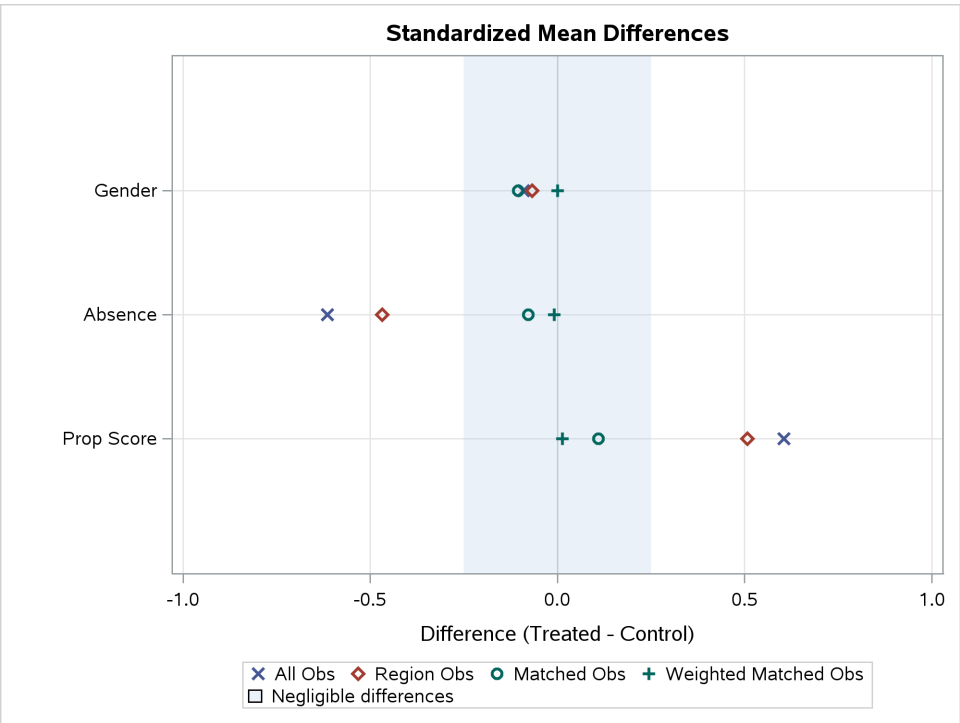

All differences for the matched observations are within the recommended limits of –0.25 and 0.25, which are indicated by the shaded area.

The NMATCHMOST=6 option requests a table of the six observations in the control group that have the most matches, which is shown in Output 101.6.6. The table does not display the observations that have the most matches in the treated group because each treated unit is matched only once (K=1).

Output 101.6.6 Observations with the Most Matches

Observations with Most Matches	
Control (Music = No)	
Observation	Matched Treated
124	4
99	3
123	3
113	2
140	2
101	2

Because matching results in good balance for the variables in this example, the matched observations can be saved in an output data set for use in a subsequent outcome analysis.

In situations where you are not satisfied with the variable balance, you can do one or more of the following to improve the balance: you can select another set of variables to fit the propensity score model, you can modify the matching criteria, or you can choose another matching method.

The OUT(OBS=MATCH)=OutEx6 option in the OUTPUT statement creates an output data set, OutEx6, that contains the matched observations. The following statements list the observations in the last four matched sets, which are shown in Output 101.6.7.

```
proc sort data=OutEx6 out=OutEx6a;
   by descending _MatchID;
run;

proc print data=OutEx6a(obs=10);
   var StudentID Music Gender Absence _PS_ _MATCHWGT_ _MatchID;
run;
```

Output 101.6.7 Output Data Set of Matched Observations with Replacement

Obs	StudentID	Music	Gender	Absence	_PS_	_MATCHWGT_	_MatchID
1	156	Yes	Male	0.02	0.49015	1	41
2	142	Yes	Male	0.02	0.49015	1	41
3	173	No	Male	0.03	0.48874	2	41
4	64	No	Male	0.03	0.48874	1	40
5	98	Yes	Male	0.05	0.48590	1	40
6	89	No	Male	0.10	0.47883	2	39
7	115	Yes	Male	0.11	0.47742	1	39
8	182	Yes	Male	0.10	0.47883	1	39
9	130	No	Male	0.18	0.46753	1	38
10	104	Yes	Male	0.19	0.46612	1	38

By default, the output data set includes the variable _PS_ (which provides the propensity score) and the variable _MATCHWGT_ (which provides matched observation weights). The MATCHID=_MatchID option creates a variable named _MatchID that identifies the matched sets of observations.

If you assume that no other confounding variables are associated with both the GPA and the music class indicator Music, you can add the GPAs for the students to the data set OutEx6 and perform an outcome analysis of GPA on this data set to estimate the music class effect.

Example 101.7: Mahalanobis Distance Matching

(View the complete code for this example (psmchex7.sas) in the example repository.)

This example illustrates how you can perform Mahalanobis distance matching of observations in a control group with observations in a treatment group, so that the matched observations can be used to estimate the treatment effect in a subsequent outcome analysis. The outcome analysis itself is not shown here.

The data for this example are observations on patients in a nonrandomized clinical trial. The trial and the Drugs data set that contains the patient information are described in the section "Getting Started: PSMATCH Procedure" on page 8372.

The following statements request optimal matching based on Mahalanobis distances between patients in the treatment group and patients in the control group:

```
ods graphics on;
proc psmatch data=drugs region=cs;
   class Drug Gender;
   psmodel Drug(Treated='Drug_X')= Gender Age BMI;
   match method=optimal(k=1) exact=Gender
         distance=mah(lps var=(Age BMI)) caliper=.
         weight=none;
   assess lps var=(Gender Age BMI);
   output out(obs=match)=OutEx7 matchid=_MatchID;
run;
```

The PSMODEL statement specifies the logistic regression model that creates the propensity score for each observation, which is the probability that the patient receives Drug_X. The CLASS statement specifies the classification variables in the model. The Drug variable is the binary treatment indicator variable, and TREATED='Drug_X' identifies Drug_X as the treated group. The Gender, Age, and BMI variables are included in the model because they are believed to be related to the assignment.

The PSMATCH procedure matches only those observations whose propensity scores lie in the support region that you specify with the REGION= option. Here the REGION=CS option requests that only those observations whose propensity scores (or equivalently, logits of propensity scores) lie in the common support region be used for matching. The common support region is the largest interval that contains propensity scores (or equivalently, logits of propensity scores) for both treated and control observations. By default, the region is extended by 0.25 times the pooled estimate of the common standard deviation of the logits of the propensity scores.

The MATCH statement specifies the criteria for matching. The DISTANCE=MAH(LPS VAR=(AGE BMI)) option requests that Mahalanobis distance be used in computing differences between pairs of observations, and that this distance be derived from the logit of the propensity score and the Age and BMI variables. The METHOD=OPTIMAL(K=1) option (which is the default) requests optimal matching of one control unit to

each unit in the treated group in order to minimize the total within-pair difference. The EXACT=GENDER option requests that the treated unit and its matched control unit have the same value of Gender. The CALIPER=. option ignores the caliper requirement for matching.

The "Data Information" table in Output 101.7.1 displays the numbers of observations in the treated and control groups, the lower and upper limits of propensity scores for observations in the support region, and the numbers of observations in the treated and control groups that fall within the support region. Of the 373 observations in the control group, 351 fall within the support region.

Output 101.7.1 Data Information

The PSMATCH Procedure

Data Information	
Data Set	WORK.DRUGS
Output Data Set	WORK.OUTEX7
Treatment Variable	Drug
Treated Group	Drug_X
All Obs (Treated)	113
All Obs (Control)	373
Support Region	Extended Common Support
Lower PS Support	0.050244
Upper PS Support	0.683999
Support Region Obs (Treated)	113
Support Region Obs (Control)	351

The "Propensity Score Information" table in Output 101.7.2 displays summary statistics by treatment group for all observations, for observations in the support region, and for matched observations.

Output 101.7.2 Propensity Score Information

Propensity Score Information											
	Treated (Drug = Drug_X)					Control (Drug = Drug_A)					Treated - Control
Observations	**N**	**Mean**	**Standard Deviation**	**Minimum**	**Maximum**	**N**	**Mean**	**Standard Deviation**	**Minimum**	**Maximum**	**Mean Difference**
All	113	0.3108	0.1325	0.0602	0.6411	373	0.2088	0.1320	0.0202	0.6858	0.1020
Region	113	0.3108	0.1325	0.0602	0.6411	351	0.2176	0.1267	0.0510	0.6824	0.0932
Matched	113	0.3108	0.1325	0.0602	0.6411	113	0.3053	0.1337	0.0640	0.6824	0.0055

The "Matching Information" table in Output 101.7.3 displays the matching criteria, the number of matched sets, the numbers of matched observations in the treated and control groups, and the total Mahalanobis difference for all matches.

Output 101.7.3 Matching Information

Matching Information	
Distance Metric	Mahalanobis Distance
Mahalanobis Covariance	Control Group
Method	Optimal Fixed Ratio Matching
Control/Treated Ratio	1
Matched Sets	113
Matched Obs (Treated)	113
Matched Obs (Control)	113
Total Absolute Difference	20.05243

The ASSESS statement produces tables and plots that summarize differences in the distributions of the specified variables between treated and control groups for all observations, for observations in the support region, and for matched observations. As requested by the LPS and VAR= options, the variables that are listed in the table are the logit of the propensity score and the variables Gender, Age, and BMI. The WEIGHT=NONE option suppresses the display of differences for the weighted matched observations. Note that when one control unit is matched with each treated unit, the weights are all 1 for matched treated and control units, and the results are identical for the weighted matched observations and the matched observations.

The "Standardized Mean Differences" table, shown in Output 101.7.4, displays standardized mean differences in the variables between the treated and control groups for all observations, for observations in the support region, and for matched observations. For a binary classification variable (Gender), the difference is in the proportion of the first ordered level (Female).

Output 101.7.4 Standardized Mean Differences

The PSMATCH Procedure

Standardized Mean Differences (Treated - Control)						
Variable	**Observations**	**Mean Difference**	**Standard Deviation**	**Standardized Difference**	**Percent Reduction**	**Variance Ratio**
Logit Prop Score	**All**	0.63997	0.767449	0.83389		0.6517
	Region	0.54546		0.71074	14.77	0.8314
	Matched	0.02757		0.03592	95.69	0.9813
Age	**All**	-4.09509	6.079104	-0.67363		0.7076
	Region	-3.49368		-0.57470	14.69	0.8000
	Matched	-0.11504		-0.01892	97.19	1.0007
BMI	**All**	0.73930	1.923178	0.38441		0.8854
	Region	0.63257		0.32892	14.44	0.9288
	Matched	0.06186		0.03216	91.63	1.0176
Gender	**All**	-0.02482	0.496925	-0.04994		0.9892
	Region	-0.01651		-0.03323	33.46	0.9922
	Matched	0.00000		0.00000	100.00	1.0000
Standard deviation of All observations used to compute standardized differences						

The standardized mean differences are significantly reduced in the matched observations. The largest of these differences is 0.03592 in absolute value, which is less than the recommended upper limit of 0.25. The treated-to-control variance ratios between the two groups are also in the recommended range of 0.5 to 2.

Because EXACT=GENDER is specified in the MATCH statement, the standardized mean difference for Gender is 0 in the matched observations.

The PSMATCH procedure displays a standardized mean differences plot, shown in Output 101.7.5, for the variables that are specified in the ASSESS statement.

Output 101.7.5 Standardized Mean Differences Plot

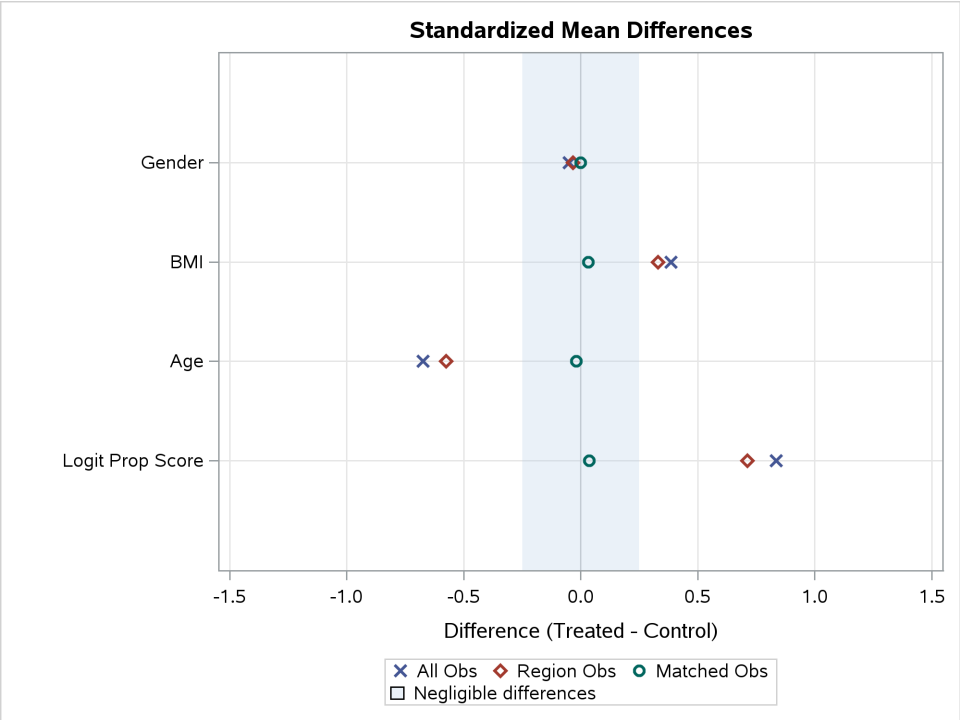

The plot displays the standardized mean differences that are listed in the "Standardized Mean Differences" table in Output 101.7.4. All differences for the matched observations are within the recommended limits of –0.25 and 0.25, which are indicated by the shaded area.

Because matching results in good balance for the variables in this example, the matched observations can be saved in an output data set for use in a subsequent outcome analysis.

In situations where you are not satisfied with the variable balance, you can do one or more of the following to improve the balance: you can select another set of variables to fit the propensity score model, you can modify the matching criteria, or you can choose another matching method.

The OUT(OBS=MATCH)=OutEx7 option in the OUTPUT statement creates an output data set, OutEx7, that contains the matched observations. The following statements list the observations in the first four matched sets, which are shown in Output 101.7.6:

```
proc sort data=OutEx7 out=OutEx7a;
   by _MatchID;
run;

proc print data=OutEx7a(obs=8);
    var PatientID Drug Gender Age BMI _ps_ _MatchWgt_ _MatchID;
run;
```

Output 101.7.6 Output Data Set with Mahalanobis Distance Matches

Obs	PatientID	Drug	Gender	Age	BMI	_PS_	_MATCHWGT_	_MatchID
1	141	Drug_A	Female	43	20.55	0.064010	1	1
2	89	Drug_X	Female	44	20.75	0.060231	1	1
3	137	Drug_A	Female	45	22.04	0.072149	1	2
4	323	Drug_X	Female	46	22.22	0.067625	1	2
5	429	Drug_A	Male	49	24.00	0.088477	1	3
6	217	Drug_X	Male	49	23.96	0.087716	1	3
7	111	Drug_A	Female	41	21.01	0.087140	1	4
8	234	Drug_X	Female	41	21.11	0.089042	1	4

By default, the output data set includes the variable _PS_ (which provides the propensity score) and the variable _MATCHWGT_ (which provides matched observation weights). The weight for each treated unit is 1. Because K=1 is specified in the METHOD= option in the MATCH statement, one control unit is matched to each treated unit, and so the weight for each matched control unit is also 1. The MATCHID=_MatchID option creates a variable named _MatchID that identifies the matched sets of observations.

After the responses for the trial are observed, they can be added to the data set OutEx7 as the starting point for an outcome analysis. Assuming that no other confounding variables are associated with both the response variable and the treatment group indicator Drug, you can estimate the treatment effect from the matched observations by performing an outcome analysis that you would have used to estimate the treatment effect if the original data set had resulted from a randomized trial.

Example 101.8: Matching with Precomputed Propensity Scores

(View the complete code for this example (psmchex8.sas) in the example repository.)

The PSMATCH procedure provides the capability for fitting a binary logistic regression model that is used to compute propensity scores for matching. However, there might be situations in which you have already computed the propensity scores—for example, by using other procedures in SAS/STAT software that perform logistic regression. This example illustrates optimal matching with precomputed propensity scores that are provided in the input data set for PROC PSMATCH.

The data for this example are observations on patients in a nonrandomized clinical trial. The trial and the Drugs data set that contains the patient information are described in the section "Getting Started: PSMATCH Procedure" on page 8372.

The following statements use the LOGISTIC procedure to derive propensity scores:

```
ods select none;
proc logistic data=drugs;
   class Drug Gender;
   model Drug(Event='Drug_X')= Gender Age BMI / link=cloglog;
   output out=drug1 p=pscore;
run;
ods select all;
```

The LINK=CLOGLOG option fits the complementary log-log model and derives propensity scores that are used in the PSMATCH procedure. The option is used just to demonstrate that, other than the logit link that is

provided in the PSMATCH procedure, you can use a different model to derive propensity scores and then input these propensity scores in the PSMATCH procedure.

The output data set Drug1 is constructed from the data set Drugs and contains the PScore variable for propensity scores.

Output 101.8.1 lists the first 10 observations.

Output 101.8.1 Data Set with Propensity Scores

Obs	PatientID	Drug	Gender	Age	BMI	pscore
1	284	Drug_X	Male	29	22.02	0.35498
2	201	Drug_A	Male	45	26.68	0.21794
3	147	Drug_A	Male	42	21.84	0.12261
4	307	Drug_X	Male	38	22.71	0.19821
5	433	Drug_A	Male	31	22.76	0.34298
6	435	Drug_A	Male	43	26.86	0.26261
7	159	Drug_A	Female	45	25.47	0.15077
8	368	Drug_A	Female	49	24.28	0.08713
9	286	Drug_A	Male	31	23.31	0.37211
10	163	Drug_X	Female	39	25.34	0.24005

The following statements request optimal matching to match patients in the treatment group to patients in the control group:

```
ods graphics on;
proc psmatch data=Drug1 region=cs;
   class Drug Gender;
   psdata treatvar=Drug(Treated='Drug_X') ps=pscore;
   match method=optimal(k=1) exact=Gender distance=lps caliper=0.5
         weight=none;
   assess lps var=(Gender Age BMI);
   output out(obs=match)=OutEx8 lps=_Lps matchid=_MatchID;
run;
```

The PSMODEL statement is not used in this example because the propensity scores are provided in Drug1. Instead, the PSDATA statement is used to identify the binary treatment variable and the propensity score variable in Drug1. The CLASS statement specifies the classification variables. The PS= option specifies pscore as the propensity score variable. The TREATVAR=DRUG option specifies Drug as the binary treatment indicator variable, and TREATED='Drug_X' identifies Drug_X as the treated group.

The PSMATCH procedure matches only those observations whose propensity scores lie in the support region that you specify with the REGION= option. Here the REGION=CS option requests that only those observations whose propensity scores (or equivalently, logits of propensity scores) lie in the common support region be used for matching. The common support region is the largest interval that contains propensity scores (or equivalently, logits of propensity scores) for both treated and control observations. By default, the region is extended by 0.25 times the pooled estimate of the common standard deviation of the logits of the propensity scores.

The MATCH statement specifies the criteria for matching. The DISTANCE=LPS option (which is the default) requests that the logit of the propensity score be used in computing differences between pairs of observations. The METHOD=OPTIMAL(K=1) option (which is the default) requests optimal matching of one control unit

to each unit in the treated group in order to minimize the total within-pair difference. The EXACT=GENDER option requests that the treated unit and its matched control unit have the same value of the Gender variable. The CALIPER=0.5 option requests that a match be made only if the difference in the logits of the propensity scores for pairs of individuals is less than or equal to 0.5 times the pooled estimate of the common standard deviation of the logits of the propensity scores.

The "Data Information" table in Output 101.8.2 displays the numbers of observations in the treated and control groups, the lower and upper limits for the propensity scores of observations in the support region, and the numbers of observations in the treated and control groups that fall within the support region. Of the 373 observations in the control group, 352 fall within the support region.

Output 101.8.2 Data Information

The PSMATCH Procedure

Data Information	
Data Set	WORK.DRUG1
Output Data Set	WORK.OUTEX8
Treatment Variable	Drug
Treated Group	Drug_X
All Obs (Treated)	113
All Obs (Control)	373
Support Region	Extended Common Support
Lower PS Support	0.060563
Upper PS Support	0.698199
Support Region Obs (Treated)	113
Support Region Obs (Control)	352

The "Propensity Score Information" table in Output 101.8.3 displays summary statistics by treatment group for all observations, for observations in the support region, and for matched observations.

Output 101.8.3 Propensity Score Information

Propensity Score Information											
	Treated (Drug = Drug_X)					Control (Drug = Drug_A)					Treated - Control
Observations	**N**	**Mean**	**Standard Deviation**	**Minimum**	**Maximum**	**N**	**Mean**	**Standard Deviation**	**Minimum**	**Maximum**	**Mean Difference**
All	113	0.3040	0.1287	0.0715	0.6594	373	0.2089	0.1255	0.0295	0.7135	0.0952
Region	113	0.3040	0.1287	0.0715	0.6594	352	0.2146	0.1177	0.0606	0.6519	0.0894
Matched	113	0.3040	0.1287	0.0715	0.6594	113	0.2984	0.1215	0.0723	0.6519	0.0056

The "Matching Information" table in Output 101.8.4 displays the matching criteria, the number of matched sets, the numbers of matched observations in the treated and control groups, and the total absolute difference in the logits of the propensity scores for all matches.

Output 101.8.4 Matching Information

Matching Information	
Distance Metric	Logit of Propensity Score
Method	Optimal Fixed Ratio Matching
Control/Treated Ratio	1
Caliper (Logit PS)	0.356051
Matched Sets	113
Matched Obs (Treated)	113
Matched Obs (Control)	113
Total Absolute Difference	3.616259

The ASSESS statement produces tables and plots that summarize differences in the distributions of the specified variables between treated and control groups for all observations, for observations in the support region, and for matched observations. As requested by the LPS and VAR= options, the variables that are listed in the table are the logit of the propensity score and the variables Gender, Age, and BMI. The WEIGHT=NONE option suppresses the display of differences for the weighted matched observations. When one control unit is matched to each treated unit, the weights are all 1 for matched treated and control units, so the results for weighted matched observations and matched observations are identical.

The "Standardized Mean Differences" table displays standardized mean differences in the variables between the treated and control groups. For a binary classification variable (Gender), the difference is in the proportion of the first ordered level (Female).

Output 101.8.5 Standardized Mean Differences

The PSMATCH Procedure

Standardized Mean Differences (Treated - Control)						
Variable	**Observations**	**Mean Difference**	**Standard Deviation**	**Standardized Difference**	**Percent Reduction**	**Variance Ratio**
Logit Prop Score	**All**	0.58239	0.712102	0.81785		0.7177
	Region	0.51613		0.72480	11.38	0.9052
	Matched	0.02268		0.03184	96.11	1.0929
Age	**All**	-4.09509	6.079104	-0.67363		0.7076
	Region	-3.58515		-0.58975	12.45	0.7928
	Matched	0.11504		0.01892	97.19	1.0143
BMI	**All**	0.73930	1.923178	0.38441		0.8854
	Region	0.65089		0.33845	11.96	0.9394
	Matched	0.14619		0.07602	80.23	1.3509
Gender	**All**	-0.02482	0.496925	-0.04994		0.9892
	Region	-0.01808		-0.03638	27.16	0.9916
	Matched	0.00000		0.00000	100.00	1.0000
Standard deviation of All observations used to compute standardized differences						

The standardized mean differences are significantly reduced in the matched observations, and the largest of these differences is 0.076 in absolute value, which is less than the recommended upper limit of 0.25. The treated-to-control variance ratios between the two groups are between 1 and 1.3509 for all variables in the matched observations, which is within the recommended range of 0.5 to 2. Because both EXACT=GENDER and METHOD=OPTIMAL are specified in the MATCH statement, the standardized mean difference for

Gender is 0 in the matched observations.

The PSMATCH procedure displays a standardized mean differences plot, as shown in Output 101.8.6, for the variables that are specified in the ASSESS statement.

Output 101.8.6 Standardized Mean Differences Plot

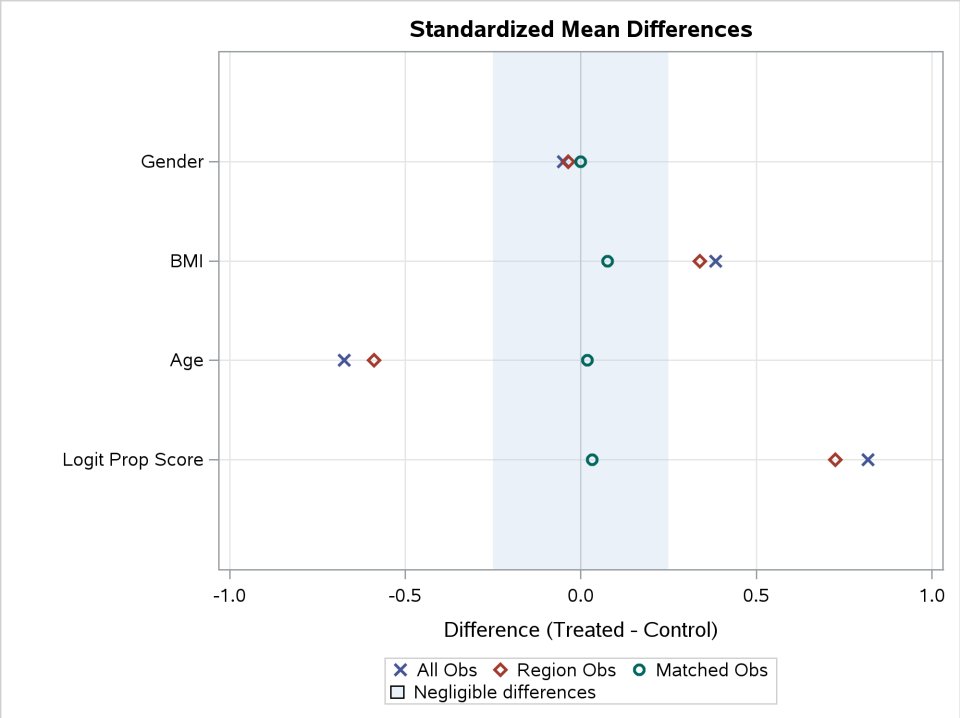

The "Standardized Mean Differences Plot" displays the standardized mean differences that are listed in the "Standardized Mean Differences" table in Output 101.8.5. All differences for the matched observations are within the recommended limits of –0.25 and 0.25, which are indicated by the shaded area.

Because matching results in good balance for the variables in this example, the matched observations can be saved in an output data set for use in a subsequent outcome analysis.

In situations where you are not satisfied with the variable balance, you can do one or more of the following to improve the balance: you can select another set of variables to fit the propensity score model, you can modify the matching criteria, or you can choose another matching method.

The OUT(OBS=MATCH)=OutEx8 option in the OUTPUT statement creates an output data set, OutEx8, that contains the matched observations. The following statements list the observations in the first five matched sets, as shown in Output 101.8.7:

```
proc sort data=OutEx8 out=OutEx8a;
   by _MatchID;
run;

proc print data=OutEx8a(obs=10);
   var PatientID Drug Gender Age BMI pscore _LPS _MatchWgt_ _MatchID;
run;
```

Output 101.8.7 Output Data Set With Optimal Matches

Obs	PatientID	Drug	Gender	Age	BMI	pscore	_Lps	_MATCHWGT_	_MatchID
1	213	Drug_A	Female	49	23.24	0.07234	-2.55123	1	1
2	89	Drug_X	Female	44	20.75	0.07152	-2.56356	1	1
3	245	Drug_A	Female	52	25.32	0.08090	-2.43015	1	2
4	323	Drug_X	Female	46	22.22	0.07822	-2.46677	1	2
5	429	Drug_A	Male	49	24.00	0.09865	-2.21228	1	3
6	217	Drug_X	Male	49	23.96	0.09796	-2.22013	1	3
7	234	Drug_X	Female	41	21.11	0.09887	-2.20987	1	4
8	66	Drug_A	Female	48	24.53	0.09927	-2.20531	1	4
9	183	Drug_A	Female	45	23.62	0.10931	-2.09786	1	5
10	320	Drug_X	Female	46	24.17	0.11056	-2.08507	1	5

By default, the output data set includes the variable _PS_ (which provides the propensity score) and the variable _MATCHWGT_ (which provides matched observation weights). The weight for each treated unit is 1. Because K=1 is specified in the METHOD=OPTIMAL option in the MATCH statement, one control unit is matched to each treated unit, so the weight for each matched control unit is also 1. The LPS=_LPS option creates a variable named _LPS (which provides the logit of the propensity score) and the MATCHID=_MatchID option creates a variable named _MatchID (which identifies the matched sets of observations).

After the responses for the trial are observed, they can be added to the data set OutEx8 as the starting point for an outcome analysis. Assuming that no other confounding variables are associated with both the response variable and the treatment group indicator Drug, you can estimate the treatment effect from the matched observations by performing an outcome analysis that you would have used to estimate the treatment effect if the original data set had resulted from a randomized trial.

Example 101.9: Sensitivity Analysis after One-to-One Matching

(View the complete code for this example (psmchex9.sas) in the example repository.)

This example illustrates how you can analyze sensitivity to the assumption of no unobserved confounders after performing one-to-one matching with the PSMATCH procedure. For a detailed description of this analysis, see the section "Sensitivity Analysis" on page 8421.

A pharmaceutical company conducts a nonrandomized clinical trial to demonstrate the efficacy of a new treatment (Drug_X) to decrease the low-density lipoprotein (LDL) by comparing it to an existing treatment (Drug_A). The data set Drugs, which is described in "Getting Started: PSMATCH Procedure" on page 8372, contains baseline variable measurements for individuals from the treated and control groups.

Output 101.9.1 lists the first eight observations.

Output 101.9.1 Input Drugs Data Set

Obs	PatientID	Drug	Gender	Age	BMI
1	1	Drug_X	Male	29	22.02
2	2	Drug_A	Male	45	26.68
3	3	Drug_A	Male	42	21.84
4	4	Drug_X	Male	38	22.71
5	5	Drug_A	Male	31	22.76
6	6	Drug_A	Male	43	26.86
7	7	Drug_A	Female	45	25.47
8	8	Drug_A	Female	49	24.28

The possibility of treatment selection bias is a concern in the analysis of the results. Patients in the trial can choose the treatment that they prefer; otherwise, physicians assign each patient to a treatment. This could lead to systematic differences in the distributions of the baseline variables in the two groups, resulting in a biased estimate of the treatment effect. Propensity score analysis that is based on matching offers an alternative that addresses this problem by balancing the distributions of the variables.

The following statements request optimal matching of observations for patients in the treatment group with observations for patients in the control group:

```
proc psmatch data=drugs region=cs;
   class Drug Gender;
   psmodel Drug(Treated='Drug_X')= Gender Age BMI;
   match method=optimal(k=1) exact=Gender distance=lps caliper=0.25
         weight=none;
   output out(obs=match)=Outgs lps=_Lps matchid=_MatchID;
run;
```

The statements are identical to those in "Getting Started: PSMATCH Procedure" on page 8372, except that the ASSESS statement is not used here. The MATCH statement requests optimal matching of one control unit to each unit in the treated group in order to minimize the total within-pair difference.

The OUT(OBS=MATCH)=Outgs option in the OUTPUT statement creates an output data set, Outgs, that contains the matched observations.

After the trial, the data set Cholesterol contains the LDL information for the matched observations. PatientID is the patient identification number, and the response variable LDL is the decrease in LDL, measured in milligrams per deciliter of blood (mg/dl).

The following statements combine the two data sets and list the eight observations in the combined Cholesterol data set, which are shown in Output 101.9.2:

```
proc sort data=Outgs out=Outgs1;
   by PatientID;
run;

proc sort data=Cholesterol out=Cholesterol1;
   by PatientID;
run;

data OutEx9a;
   merge Outgs1 Cholesterol1;
```

```
   by PatientID;
run;

proc print data=OutEx9a(obs=8);
   var PatientID Drug Gender Age BMI LDL _MatchID;
run;
```

Output 101.9.2 Output Data Set with LDL Decreases

Obs	PatientID	Drug	Gender	Age	BMI	LDL	_MatchID
1	1	Drug_X	Male	29	22.02	6.54	74
2	3	Drug_A	Male	42	21.84	-5.66	7
3	4	Drug_X	Male	38	22.71	5.52	24
4	5	Drug_A	Male	31	22.76	7.26	76
5	9	Drug_A	Male	31	23.31	2.64	82
6	10	Drug_X	Female	39	25.34	4.77	43
7	13	Drug_X	Female	32	24.78	4.25	84
8	18	Drug_X	Male	34	26.30	0.68	99

The following statements compute the differences in LDL between the treated and control units in each matched set:

```
proc sort data=OutEx9a out=OutEx9b;
   by _MatchID Drug;
run;

proc transpose data=OutEx9b out=OutEx9c;
   by _MatchID;
   var LDL;
run;

data OutEx9c;
   set OutEx9c;
   Diff= Col2 - Col1;
   drop Col1 COl2;
run;
```

Output 101.9.3 lists the differences in LDL decrease in the first four matched sets.

Output 101.9.3 LDL Differences in Matched sets

Obs	_MatchID	_NAME_	Diff
1	1	LDL	3.25
2	2	LDL	2.44
3	3	LDL	6.34
4	4	LDL	-1.51

The following statements perform a signed rank test, and the results are shown in Output 101.9.4.

```
ods select TestsForLocation;
proc univariate data=OutEx9c;
   var Diff;
```

```
   ods output TestsForLocation=LocTest;
run;
```

Output 101.9.4 Tests for Location

The UNIVARIATE Procedure
Variable: Diff

Tests for Location: Mu0=0				
Test	**Statistic**		**p Value**	
Student's t	**t**	2.690243	**Pr > |t|**	0.0082
Sign	**M**	11.5	**Pr >= |M|**	0.0380
Signed Rank	**S**	885.5	**Pr >= |S|**	0.0106

The "Tests for Location" table shows that there is a significant decrease in LDL at the 0.025 level for patients in the treated group.

Propensity score analysis assumes that all confounders (variables that affect both the outcome and the treatment assignment) have been measured. However, this assumption cannot be verified. When there are unobserved covariates, individuals that have the same observed covariates might not have the same probability of being assigned to the treated group. If you assume that all confounders have been measured, you should examine the sensitivity of inferences to departures from the assumption.

Based on the approach described in the section "Sensitivity Analysis on Matched Observations" on page 8422, the signed rank statistic is

$$S = \sum_{j:d_j>0} d_j^+$$

Note that this statistic is not centered, unlike the signed rank statistic that is computed by PROC UNIVARIATE and is shown in Output 101.9.4:

$$\sum_{j:d_j>0} d_j^+ - \frac{n_t(n_t+1)}{4}$$

The following statements compute the signed rank statistic:

```
data SgnRank;
   set LocTest;
   nPairs=113;
   if (Test='Signed Rank');
   SgnRank= Stat + nPairs*(nPairs+1)/4;
   keep nPairs SgnRank;
run;
```

Output 101.9.5 displays the signed rank statistic.

Output 101.9.5 Signed Rank Statistic

Obs	nPairs	SgnRank
1	113	4106

Using this statistic, the following statements compute and display p-values for signed rank tests that correspond to Γ values that range from 1 to 1.5.

```
data Test1;
   set SgnRank;
   mean0     = nPairs*(nPairs+1)/2;
   variance0 = mean0*(2*nPairs+1)/3;

   do Gamma=1 to 1.5 by 0.05;
      mean     = Gamma/(1+Gamma) * mean0;
      variance = Gamma/(1+Gamma)**2 * variance0;
      tTest    = (SgnRank - mean) / sqrt(variance);
      pValue   = 1 - probt(tTest, nPairs-1);
      output;
   end;
 run;

 proc print data=Test1;
 run;
```

Output 101.9.6 p-Values for Γ Values from 1 to 1.5

Obs	nPairs	SgnRank	mean0	variance0	Gamma	mean	variance	tTest	pValue
1	113	4106	6441	487369	1.00	3220.50	121842.25	2.53682	0.00628
2	113	4106	6441	487369	1.05	3299.05	121769.77	2.31248	0.01129
3	113	4106	6441	487369	1.10	3373.86	121565.96	2.09986	0.01899
4	113	4106	6441	487369	1.15	3445.19	121249.18	1.89775	0.03015
5	113	4106	6441	487369	1.20	3513.27	120835.29	1.70513	0.04547
6	113	4106	6441	487369	1.25	3578.33	120338.02	1.52110	0.06553
7	113	4106	6441	487369	1.30	3640.57	119769.32	1.34489	0.09069
8	113	4106	6441	487369	1.35	3700.15	119139.55	1.17581	0.12108
9	113	4106	6441	487369	1.40	3757.25	118457.74	1.01329	0.15655
10	113	4106	6441	487369	1.45	3812.02	117731.79	0.85678	0.19670
11	113	4106	6441	487369	1.50	3864.60	116968.56	0.70583	0.24088

Output 101.9.6 shows that at the tipping point Γ=1.15, the p-value is 0.0355, which is larger than the Type I error level of 0.025. Thus the study conclusion is reversed if for two individuals k and l in the same matched set, the probability that individual k is in the treated group and l is in the control group is

$$\frac{\pi_k}{\pi_k + \pi_l} = \frac{\Gamma}{1+\Gamma} = 0.535$$

If Γ=1.15 represents only a small departure from random treatment assignment (Γ=1), the study conclusion is not robust to hidden bias from an unobserved confounder.

Example 101.10: Entropy Balance Weighting

(View the complete code for this example (psmex10.sas) in the example repository.)

This example illustrates how you can perform entropy balance weighting. The data for this example are observations about patients in a nonrandomized clinical trial. The trial and the Drugs data set that contains the patient information are described in the section "Getting Started: PSMATCH Procedure" on page 8372. The outcome analysis is not shown here.

The following statements compute entropy balance weights and save the weights in an output data set:

```
proc psmatch data=drugs;
   class Drug Gender;
   eweight treatvar=Drug(Treated='Drug_X') balancevars=(Gender Age BMI)
           nlargestwgt=5;
   assess var=(Age BMI)/ varinfo;
   output out=OutEx10 weight=eweight;
run;
```

The EWEIGHT statement computes entropy balance weights. Because entropy balance weighting does not require propensity score values, no PSMODEL or PSDATA statement is specified.

The "Entropy Weight Information" table, shown in Output 101.10.1, displays information about the formulation of the entropy balance weight optimization problem. By default, uniform reference weights are used to formulate the entropy balance optimization problem, and entropy balance weights are computed only for observations in the control condition. Balance constraints are defined for the first two moments of continuous variables, resulting in five constraints for this example: two for each of the continuous variables Age and BMI, and one for the binary variable Gender. The entropy balance weights are accepted only if the maximum absolute violation of a balance constraint is less than the balance tolerance value of 0.0001. If accepted, the estimated weights are rescaled so that the sum of the estimated weights equals the number of treated units. For more information about the formulation of the entropy balance optimization problem, see the section "Entropy Balancing" on page 8423.

Output 101.10.1 Entropy Weight Information

The PSMATCH Procedure

Entropy Balance Weight Information	
Weighted Observations	Control Group
Reference Weights	Uniform
Sum of Weights	113
Moments Balanced	2
Balance Constraints	5
Balance Tolerance	0.0001

The "Standardized Mean Differences" table, shown in Output 101.10.2, displays standardized mean differences and variance ratios in the variables between the treated and control groups. As expected, given the formulation of the entropy balance optimization problem, the weighted means and variances of the variables Age and BMI are comparable between the two treatment conditions.

Output 101.10.2 Standardized Mean Differences

Standardized Mean Differences (Treated - Control)						
Variable	Observations	Mean Difference	Standard Deviation	Standardized Difference	Percent Reduction	Variance Ratio
Age	All	-4.09509	6.079104	-0.67363		0.7076
	Weighted	0.00000		0.00000	100.00	1.0000
BMI	All	0.73930	1.923178	0.38441		0.8854
	Weighted	0.00000		0.00000	100.00	1.0000
Standard deviation of All observations used to compute standardized differences						

The "Observations with Largest Weights" table, shown in Output 101.10.3, lists the observations in the control condition that have the five largest entropy balance weights. If you specified the WEIGHT=ALLOBS option in the EWEIGHT statement, entropy balance weights would be computed for each treatment condition, and the table would include information about the largest weights for the treatment condition. For this example, there do not appear to be any excessively large weights.

Output 101.10.3 Largest Entropy Balance Weights

Observations with Largest Entropy Balance Weights	
Control (Drug = Drug_A)	
Observation	Weight
317	1.43
134	1.40
81	1.19
446	1.09
437	1.04

References

Austin, P. C. (2007). "The Performance of Different Propensity Score Methods for Estimating Marginal Odds Ratios." *Statistics in Medicine* 26:3078–3094.

Austin, P. C. (2009). "Balance Diagnostics for Comparing the Distribution of Baseline Covariates between Treatment Groups in Propensity-Score Matched Samples." *Statistics in Medicine* 28:3083–3107.

Austin, P. C. (2011a). "An Introduction to Propensity Score Methods for Reducing the Effects of Confounding in Observational Studies." *Multivariate Behavioral Research* 46:399–424.

Austin, P. C. (2011b). "Optimal Caliper Widths for Propensity-Score Matching When Estimating Differences in Means and Differences in Proportions in Observational Studies." *Pharmaceutical Statistics* 10:150–161.

Austin, P. C. (2014). "A Comparison of 12 Algorithms for Matching on the Propensity Score." *Statistics in Medicine* 33:1057–1069.

Austin, P. C., Grootendorst, P., and Anderson, G. M. (2007). "A Comparison of the Ability of Different Propensity Score Models to Balance Measures Variables between Treated and Untreated Subjects: A Monte Carlo Study." *Statistics in Medicine* 26:734–753.

Austin, P. C., and Stuart, E. A. (2015a). "Moving towards Best Practice When Using Inverse Probability of Treatment Weighting (IPTW) Using the Propensity Score to Estimate Causal Treatment Effects in Observational Studies." *Statistics in Medicine* 34:3661–3679.

Austin, P. C., and Stuart, E. A. (2015b). "The Performance of Inverse Probability of Treatment Weighting and Full Matching on the Propensity Score in the Presence of Model Misspecification When Estimating the Effect of Treatment on Survival Outcomes." *Statistical Methods in Medical Research* 26:1654–1670.

Cole, S. R., and Hernán, M. A. (2008). "Constructing Inverse Probability Weights for Marginal Structural Models." *American Journal of Epidemiology* 168:656–664.

Crump, R. K., Hotz, V. J., Imbens, G. W., and Mitnik, O. A. (2009). "Dealing with Limited Overlap in Estimation of Average Treatment Effects." *Biometrika* 96:187–199.

Faries, D. E., Leon, A. C., Haro, J. M., and Obenchain, R. L., eds. (2010). *Analysis of Observational Health Care Data Using SAS*. Cary, NC: SAS Institute Inc.

Guo, S., and Fraser, M. W. (2015). *Propensity Score Analysis: Statistical Methods and Applications*. 2nd ed. Thousand Oaks, CA: Sage Publications.

Hainmueller, J. (2012). "Entropy Balancing for Causal Effects: A Multivariate Reweighting Method to Produce Balanced Samples in Observational Studies." *Political Analysis* 20:25–46. https://doi.org/10.1093/pan/mpr025.

Hansen, B. B. (2004). "Full Matching in an Observational Study of Coaching for the SAT." *Journal of the American Statistical Association* 99:609–618.

Hill, J., and Reiter, J. P. (2006). "Interval Estimation for Treatment Effects Using Propensity Score Matching." *Statistics in Medicine* 25:2230–2256.

Ho, D., Imai, K., King, G., and Stuart, E. A. (2007). "Matching as Nonparametric Preprocessing for Reducing Model Dependence in Parametric Causal Inference." *Political Analysis* 15:199–236.

Imbens, G. W., and Rubin, D. B. (2015). *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. New York: Cambridge University Press.

Liu, W., Kuramoto, S. J., and Stuart, E. A. (2013). "An Introduction to Sensitivity Analysis for Unobserved Confounding in Non-experimental Prevention Research." *Prevention Science* 14:570–580.

Lunceford, J. K., and Davidian, M. (2004). "Stratification and Weighting via the Propensity Score in Estimation of Causal Treatment Effects: A Comparative Study." *Statistics in Medicine* 23:2937–2960.

Mamdani, M., Sykora, K., Li, P., Normand, S. L., Streiner, D. L., Austin, P. C., Rochon, P. A., and Anderson, G. M. (2005). "Reader's Guide to Critical Appraisal of Cohort Studies: 2. Assessing Potential for Confounding." *BMJ* 330:960–962.

Normand, S.-L. T., Landrum, M. B., Guadagnoli, E., Ayanian, J. Z., Ryan, T. J., Cleary, P. D., and McNeil, B. J. (2001). "Validating Recommendations for Coronary Angiography Following Acute Myocardial Infarction in the Elderly: A Matched Analysis Using Propensity Scores." *Journal of Clinical Epidemiology* 54:387–398.

Pan, W., and Bai, H., eds. (2015). *Propensity Score Analysis: Fundamentals and Developments*. New York: Guilford Press.

Robins, J. M., Hernan, M. A., and Brumback, B. (2000). "Marginal Structural Models and Causal Inference in Epidemiology." *Epidemiology* 11:550–560.

Rosenbaum, P. R. (2010). *Design of Observational Studies*. New York: Springer-Verlag.

Rosenbaum, P. R., and Rubin, D. B. (1983). "The Central Role of the Propensity Score in Observational Studies for Causal Effects." *Biometrika* 70:41–55.

Rosenbaum, P. R., and Rubin, D. B. (1984). "Reducing Bias in Observational Studies Using Subclassification on the Propensity Score." *Journal of the American Statistical Association* 79:516–524.

Rosenbaum, P. R., and Rubin, D. B. (1985). "Constructing a Control Group Using Multivariate Matched Sampling Methods That Incorporate the Propensity Score." *American Statistician* 39:33–38.

Rubin, D. B. (1974). "Estimating Causal Effects of Treatments in Randomized and Nonrandomized Studies." *Journal of Educational Psychology* 66:688–701.

Rubin, D. B. (1980a). "Bias Reduction Using Mahalanobis-Metric Matching." *Biometrics* 36:293–298.

Rubin, D. B. (1980b). "Comment on D. Basu, 'Randomization Analysis of Experimental Data: The Fisher Randomization Test'." *Journal of the American Statistical Association* 75:591–593.

Rubin, D. B. (1990). "Comment: Neyman (1923) and Causal Inference in Experiments and Observational Studies." *Statistical Science* 5:472–480.

Rubin, D. B. (2001). "Using Propensity Scores to Help Design Observational Studies: Application to the Tobacco Litigation." *Health Services and Outcomes Research Methodology* 2:169–188.

Rubin, D. B. (2005). "Causal Inference Using Potential Outcomes: Design, Modeling, Decisions." *Journal of the American Statistical Association* 100:322–331.

Stuart, E. A. (2007). "Estimating Causal Effects Using School-Level Data Sets." *Educational Researcher* 36:187–198.

Stuart, E. A. (2010). "Matching Methods for Causal Inference: A Review and a Look Forward." *Statistical Science* 25:1–21.

Stuart, E. A., and Ialongo, N. S. (2010). "Matching Methods for Selection of Subjects for Follow-Up." *Multivariate Behavioral Research* 45:746–765.

Stuart, E. A., Marcus, S. M., Horvitz-Lennon, M. V., Gibbons, R. D., and Normand, S.-L. T. (2009). "Using Non-experimental Data to Estimate Treatment Effects." *Psychiatric Annals* 39:719–728.

Stürmer, T., Wyss, R., Glynn, R. J., and Brookhart, M. A. (2014). "Propensity Scores for Confounder Adjustment When Assessing the Effects of Medical Interventions Using Nonexperimental Study Designs." *Journal of Internal Medicine* 275:570–580.

Yue, L. Q., Lu, N., and Xu, Y. (2014). "Designing Premarket Observational Comparative Studies Using Existing Data as Controls: Challenges and Opportunities." *Journal of Biopharmaceutical Statistics* 24:994–1010.

Chapter 102
The QUANTLIFE Procedure

Contents

Overview: QUANTLIFE Procedure **8498**
- Features 8498
- Quantile Regression 8499

Getting Started: QUANTLIFE Procedure **8500**

Syntax: QUANTLIFE Procedure **8503**
- PROC QUANTLIFE Statement 8504
- BASELINE Statement 8507
- BY Statement 8507
- CLASS Statement 8508
- EFFECT Statement 8508
- MODEL Statement 8510
- OUTPUT Statement 8510
- TEST Statement 8511
- WEIGHT Statement 8511

Details: QUANTLIFE Procedure **8512**
- Notation for Censored Quantile Regression 8512
- Kaplan-Meier-Type Estimator for Censored Quantile Regression 8513
- Nelson-Aalen-Type Estimator for Censored Quantile Regression 8513
- Relationship of Quantile Function and Survival Function 8514
- Confidence Interval 8515
- Output Data Sets 8516
- ODS Table Names 8517
- ODS Graphics 8517

Examples: QUANTLIFE Procedure **8518**
- Example 102.1: Primary Biliary Cirrhosis Study 8518
- Example 102.2: Drug Abuse Study 8523

References **8527**

Overview: QUANTLIFE Procedure

The QUANTLIFE procedure performs quantile regression analysis for survival data, in which observations are not always directly observed.

Quantile regression analysis (Koenker and Bassett 1978) is a type of regression analysis that explores how the conditional quantile of a response variable depends on its covariates. Recall that a student's score on a test is at the 0.85th quantile if his or her score is higher than that of 85% of the students who took the test. The score is also said to be at the 85th percentile. By estimating a set of conditional quantiles, you can gain more insight into the conditional distribution of the response, given its covariates.

Quantile regression provides a flexible way to capture heterogeneous effects in the sense that the tails and the central location of the conditional distributions can vary differently with the covariates. Thus, quantile regression offers a powerful tool in survival analysis, where the lifetimes are skewed and extreme survival times can be of special interest (Koenker and Geling 2001; Huang 2010).

When the observations are fully observed, you can use the QUANTREG procedure to fit a standard quantile regression model. For an introduction to the basic concepts of quantile regression analysis, see Chapter 103, "The QUANTREG Procedure."

However, lifetime data often contain incomplete observations because of censoring (Klein and Moeschberger 2003; Hosmer, Lemeshow, and May 2008). When censoring occurs, the usual standard quantile regression approach can lead to biased estimates. Thus, special approaches have been developed that account for censoring and provide valid estimates. Portnoy (2003) proposed a method to estimate conditional quantile functions by generalizing the idea of the Kaplan-Meier estimator of the survival function. Peng and Huang (2008) developed a different quantile regression approach that is motivated by the Nelson-Aalen estimator of the cumulative hazard function. Both methods can be implemented using linear programming algorithms, and both are available in the QUANTLIFE procedure. Like the standard quantile regression method for uncensored data, these two methods are distribution-free and apply to heteroscedastic data.

Features

The QUANTLIFE procedure provides the following features:

- quantile regression methods for censored data that are based on generalizations of the Kaplan-Meier and the Nelson-Aalen estimator
- the interior point algorithm for parameter estimation, which uses parallel computing when multiple processors are available
- hypothesis tests for the regression parameter
- semiparametric quantile regression that uses spline effects
- survival plots, conditional quantile plots, and quantile process plots

Quantile Regression

Suppose that Y is a dependent variable of interest (such as the survival time or some monotone transformation of the survival time) and x is a $p \times 1$ vector of covariates. The observed data consist of a random sample $\{(Y_i, x_i), i = 1, \ldots, n\}$ from (Y, x).

You can use regression analysis to explore the relationship between the response Y and its predictor x. Classical linear regression estimates the conditional mean function $E(Y|x)$ by using a linear predictor $x'\beta$; a linear quantile regression estimates the τth conditional quantile function $Q_Y(\tau|x)$ by using a different linear predictor $x'\beta(\tau)$, where the quantile level τ ranges between 0 and 1. For example, $x'\beta(0.95)$ is the linear predictor for the 0.95th quantile (commonly referred to as the 95th percentile).

The quantile regression coefficient $\beta(\tau)$ can be estimated by minimizing the following objective function over b:

$$r(b) = \sum_{i=1}^{n} \rho_\tau(Y_i - x_i' b)$$

The loss function $\rho_\tau(u)$ is defined as $u(\tau - I(u < 0))$, in contrast to the square loss function for classical linear regression.

When $\tau = 0.5$, the coefficient $\boldsymbol{\beta}(0.5)$ minimizes the sum of absolute residuals, which corresponds to median regression (or L_1 regression).

The following set of regression quantiles is referred to as the *quantile process*, and it completely describes the conditional distribution of Y, given the predictor x:

$$\{\beta(\tau) : \tau \in (0, 1)\}$$

When all the observations are observed, you can use the QUANTREG procedure to estimate the quantile function $Q_Y(\tau|x)$ and draw statistical inference about the regression parameters $\beta(\tau)$. For more information, see Chapter 103, "The QUANTREG Procedure."

However, when the observations are incomplete, as is the case with censored data in survival analysis, the classical quantile regression method is not appropriate. The QUANTLIFE procedure implements appropriate quantile regression methods to model the relationship between the response Y and the predictor x.

Getting Started: QUANTLIFE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/qlifegs1.sas.)

This example uses the human immunodeficiency virus (HIV) study data from Hosmer and Lemeshow (1999) to illustrate the basic features of PROC QUANTLIFE.

In this study, subjects were followed after a confirmed diagnosis of HIV. The primary goal was to evaluate the effect of various factors on the survival time. Two covariates for each subject were collected: age and history of prior intravenous drug use.

The following DATA step creates the data set HIV, which contains the variable Time (the follow-up time in days), the variable Status (with value 0 if Time was censored and 1 otherwise), the variable Drug (with value 1 for prior intravenous drug use and 0 otherwise), and the variable Age (the patient's age in years at the beginning of the follow-up).

```
data HIV;
   input Time Age Drug Status;
   datalines;
    5      46      0        1
    6      35      1        0
    8      30      1        1
    3      30      1        1
   22      36      0        1
    1      32      1        0

   ... more lines ...

    1      34      1        1
;
```

You can use PROC QUANTLIFE to explore the relationship between the survival time and the two covariates at different quantiles.

Suppose you are interested in the median survivors and in the longer and shorter survivors. The following statements fit a linear model for the 25th, 50th, and 75th percentiles:

```
ods graphics on;
proc quantlife data=hiv log plots=quantplot seed=1268;
   class Drug;
   model Time*Status(0) = Drug Age / quantile=(0.25 0.5 0.75);
   Drug_Effect: test Drug;
run;
```

The LOG option fits a quantile regression model for the log of Time, as is done by an accelerated failure time (AFT) model in standard survival analysis. The SEED= option maintains reproducibility of the resampling method that is used for statistical inference.

The MODEL statement specifies the response variable, Time, and the censoring variable, Censor. The value that indicates censoring is enclosed in parentheses. The values of Time are considered to be censored if the value of Censor is 0; otherwise, they are considered to be event times. The QUANTILE= option requests a fit of the conditional quantile function $Q_{\log(Y)}(\tau|x)$ at the quantile levels 0.25, 0.5, and 0.75.

The TEST statement requests a test for the hypothesis that there is no drug effect at each of the quantile levels.

Figure 102.1 displays basic model information. For example, you can see from Figure 102.1 that the response is log(Time) and the censoring rate is 20%.

Figure 102.1 Model Fitting Information

The QUANTLIFE Procedure

Model Information	
Data Set	WORK.HIV
Dependent Variable	Log(Time)
Censoring Variable	Status
Censoring Value(s)	0
Number of Observations	100
Method	Kaplan-Meier
Replications	200
Seed for Random Number Generator	1268

Class Level Information		
Name	**Levels**	**Values**
Drug	2	0 1

Summary of the Number of Event and Censored Values			
Total	**Event**	**Censored**	**Percent Censored**
100	80	20	20.00

Figure 102.2 displays the parameter estimates, which are computed using the default Kaplan-Meier-type estimator. For more information, see the section "Kaplan-Meier-Type Estimator for Censored Quantile Regression" on page 8513. In addition, Figure 102.2 displays standard errors, 95% confidence limits, t values, and p-values that are computed by the default resampling method, exponentially weighted resampling. For more information, see the section "Exponentially Weighted Method" on page 8515.

A different quantile regression model is fitted for each quantile, and the first column (Quantile) in Figure 102.2 identifies the model for the parameter estimates. Age has a negative effect on survival time. You can use the parameter estimates to predict the survival time at the quantiles of interests. For example, the 75th percentile survival time for a person who has no previous intravenous drug use at age 46 is

$$\exp(5.3351 + 1.1451 - 0.0941 \times 46) = 8.6 \text{ years}$$

Figure 102.2 Parameter Estimates

Parameter Estimates								
Quantile	Parameter		DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
0.2500	**Intercept**		1	3.0373	1.1608	0.7622	5.3125	2.62	0.0103
	Drug	0	1	0.9516	0.4376	0.0940	1.8093	2.17	0.0321
	Drug	1	0	0	0	0	0	.	.
	Age		1	-0.0646	0.0259	-0.1154	-0.0138	-2.49	0.0144
0.5000	**Intercept**		1	5.3351	0.6656	4.0305	6.6397	8.02	<.0001
	Drug	0	1	0.8681	0.2796	0.3200	1.4161	3.10	0.0025
	Drug	1	0	0	0	0	0	.	.
	Age		1	-0.1059	0.0195	-0.1442	-0.0676	-5.42	<.0001
0.7500	**Intercept**		1	5.3351	0.9092	3.5531	7.1171	5.87	<.0001
	Drug	0	1	1.1451	0.2622	0.6311	1.6591	4.37	<.0001
	Drug	1	0	0	0	0	0	.	.
	Age		1	-0.0941	0.0223	-0.1378	-0.0505	-4.23	<.0001

The PLOTS=QUANTPLOT option in the PROC QUANTLIFE statement requests the quantile process plots, which are shown in Figure 102.3. The quantile process plot is a scatter plot of an estimated regression parameter against the quantile level. You can use these plots to compare quantile-specific covariate effects. A curve that is not constant can indicate heterogeneity in the data. For example, Figure 102.3 shows that the drug effect does not change much across quantiles.

Figure 102.3 Estimated Parameters

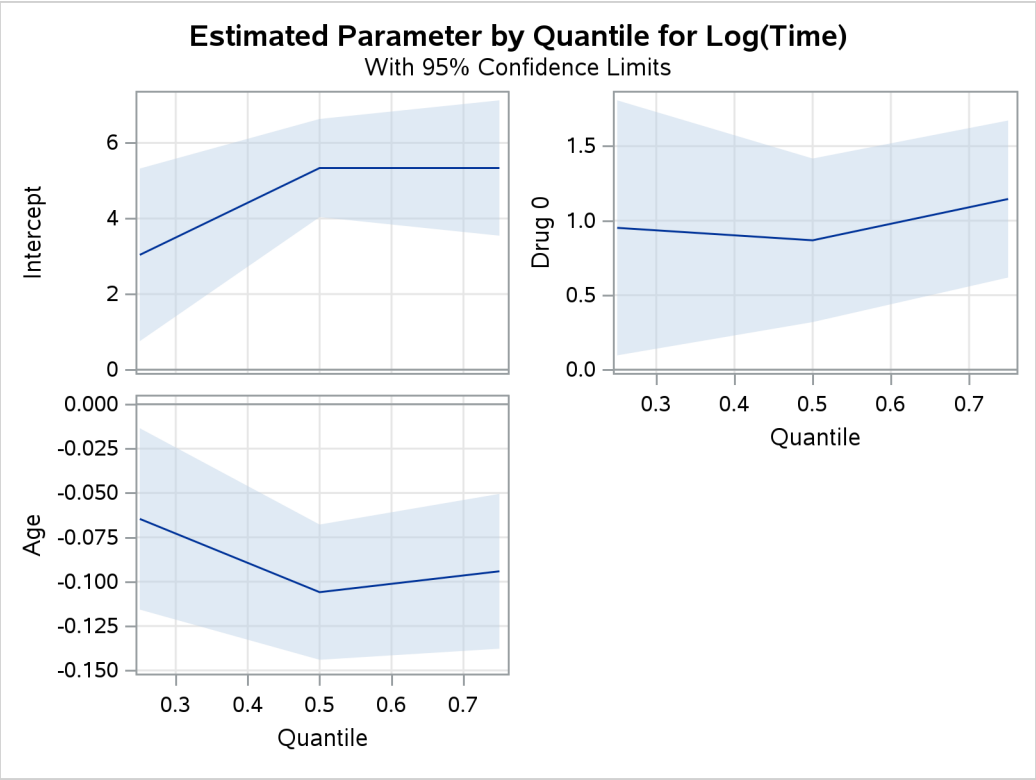

The tests that are requested by the TEST statement are shown in Figure 102.4.

Figure 102.4 Tests of Significance

Test Drug_Effect Results			
Quantile	**DF**	**Chi-Square**	**Pr > ChiSq**
0.2500	1	4.73	0.0296
0.5000	1	9.64	0.0019
0.7500	1	19.07	<.0001

The tests indicate that the coefficient of Drug is significantly different from 0 at the 25th, 50th, and 75th percentiles.

Syntax: QUANTLIFE Procedure

The following statements are available in the QUANTLIFE procedure:

```
PROC QUANTLIFE <options> ;
    BASELINE <options> ;
    BY variables ;
    CLASS variables ;
    EFFECT name = effect-type ( variables </ options> ) ;
    MODEL response <* censor(list)> = <effects> </ options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=name . . . keyword=name> ;
    TEST effects </ options> ;
    WEIGHT variable ;
```

The PROC QUANTLIFE and MODEL statements are required. The PROC QUANTLIFE statement invokes the procedure. The CLASS statement specifies which explanatory variables are treated as categorical. The MODEL statement specifies the variables to be used in the regression. You can specify main effects and interaction terms in the MODEL statement, as you can in the GLM procedure (Chapter 53, "The GLM Procedure"). The OUTPUT statement creates an output data set to contain predicted values, residuals, and estimated standard errors. The TEST statement requests linear tests for the model parameters. The WEIGHT statement identifies a variable in the input data set whose values are used to weight the observations. In one invocation of PROC QUANTLIFE, multiple OUTPUT and TEST statements are allowed.

The rest of this section provides detailed syntax information for each statement, beginning with the PROC QUANTLIFE statement. The remaining statements are covered in alphabetical order.

PROC QUANTLIFE Statement

PROC QUANTLIFE < *options* > **;**

The PROC QUANTLIFE statement invokes the QUANTLIFE procedure. Table 102.1 summarizes the options available in this statement.

Table 102.1 Options Available in the PROC QUANTLIFE Statement

Option	Description
Data Set Options	
DATA=	Specifies the input SAS data set
OUTBOOTEST=	Creates an output SAS data set for parameter estimates from resampled data sets
Basic Options	
ALPHA=	Specifies the confidence level
CI=	Specifies a resampling method for computing confidence interval and test statistics
LOG	Requests log transformation of the response
METHOD=	Specifies a method to fit quantile regression
NAMELEN=	Specifies the length of effect names
NREP=	Specifies the number of replications
SEED=	Specifies the seed for the random number generator
PLOTS=	Specifies the plots to be produced by using ODS Graphics
Computational Options	
GRIDSIZE=	Specifies a step size for the grid for computing regression quantiles
INITTAU=	Specifies the first quantile level for computing regression quantiles
KAPPA=	Specifies the step-length parameter for the interior point algorithm
MAXIT=	Specifies the maximum number of iterations for the interior point algorithm
NTHREADS=	Specifies the number of threads for the computation
TOLERANCE=	Specifies the convergence criterion of the interior point algorithm

You can specify the following *options* in the PROC QUANTLIFE statement.

ALPHA=*value*

specifies the confidence level for the regression parameters. The *value* must be between 0 and 1. By default, ALPHA=0.05, which corresponds to a 95% confidence interval.

CI=EW | PW | NONE

requests the method for computing confidence intervals for regression parameters. In addition to confidence intervals, the QUANTLIFE procedure also computes standard errors, t values, and p-values for regression parameters. You can suppress these computations by specifying CI=NONE. The QUANTLIFE procedure provides two resampling methods for computing confidence intervals, the exponentially weighted (EW) method and the pairwise (PW) resampling method. For more information, see the section "Confidence Interval" on page 8515. By default, CI=EW, which requests the exponentially weighted method.

DATA=*SAS-data-set*
specifies the *SAS-data-set* that the QUANTLIFE procedure uses. By default, the most recently created *SAS-data-set* is used.

GRIDSIZE=*value*
specifies the step size for computing regression quantiles. The *value* must be between 0 and 1. For more information, see the section "Details: QUANTLIFE Procedure" on page 8512.

INITTAU=*value*
specifies the first quantile level for computing regression quantiles. The *value* must be between 0 and 1. For more information, see the section "Details: QUANTLIFE Procedure" on page 8512.

KAPPA=*value*
specifies the step-length parameter for the interior point algorithm. The *value* must be between 0 and 1. The interior point method that the QUANTLIFE procedure uses is identical to the interior point method that the QUANTREG procedure uses. For more information, see the section "Interior Point Algorithm" on page 8575 in Chapter 103, "The QUANTREG Procedure." By default, KAPPA=0.99995.

LOG
requests that a log transformation of the response variable be performed before the model is fitted.

MAXIT=*n*
specifies the maximum number of iterations for the interior point algorithm. By default, MAXIT= 1000.

METHOD=KM | NA
requests the method to use to estimate the regression parameters. METHOD=KM specifies the Kaplan-Meier-type method (see the section "Kaplan-Meier-Type Estimator for Censored Quantile Regression" on page 8513) and METHOD=NA specifies the Nelson-Aalen-type method (see the section "Nelson-Aalen-Type Estimator for Censored Quantile Regression" on page 8513). By default, METHOD=KM.

NAMELEN=*n*
specifies the length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 200. By default, NAMELEN=20.

NREP=*n*
specifies the number of replications to draw in the resampling method. By default, NREP=200.

NTHREADS=*n*
specifies the number of threads for analytic computations and overrides the SAS system option THREADS | NOTHREADS. If you do not specify the NTHREADS= option or if you specify NTHREADS=0, the number of threads is determined based on the data size and the number of CPUs on the host where the analytic computations execute.

OUTBOOTEST=*SAS-data-set*
creates a data set to contain the parameter estimates from the resampled data sets. For a detailed description of the contents of the OUTBOOTEST= data set, see the section "OUTBOOTEST= Output Data Set" on page 8516.

PLOTS =(*plot-request* **< ...** *plot-request* **>)**

requests various plots.

When you specify one *plot-request*, you can omit the parentheses around it.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc quantlife plots=survival;
   model y=x1;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify one or more of the following *plot-requests*:

ALL
creates all appropriate plots.

NONE
suppresses all the plots in the procedure. Specifying this option is equivalent to disabling ODS Graphics for the entire procedure.

QUANTILE
plots the estimated quantile function for each combination of covariate values in the COVARIATES= data set that is specified in the BASELINE statement. If the COVARIATES= data set is not specified, then the estimated quantile function is plotted for the reference set of covariate values, which consists of reference levels for the CLASS variables and average values for the continuous variables. When the estimated quantile function is not monotonic, the quantile function (Chernozhukov, Fernandez-Val, and Galichon 2009) is rearranged to make it monotonic and then plotted.

QUANTPLOT < / UNPACK >
plots the regression quantile process. The estimated coefficient of each specified covariate effect is plotted as a function of the quantile level. You can use the UNPACK option to create individual process plots.

SURVIVAL
plots the estimated survival function for each combination of covariate values in the COVARIATES= data set that is specified in the BASELINE statement. If the COVARIATES= data set is not specified, then the estimated survival function is plotted for the reference set of covariate values, which consists of reference levels for the CLASS variables and average values for the continuous variables.

SEED=*number*
specifies a positive integer to start the pseudorandom number generator. The default is a value that is generated from reading the time of day from the computer's clock. However, to duplicate the results under identical situations, you must specify the same seed in subsequent runs of the QUANTLIFE procedure. The seed information is displayed in the "Model Information" table.

TOLERANCE=*value*
specifies the tolerance for the convergence criterion of the interior point algorithm. Both the QUANTLIFE procedure and the QUANTREG procedure use the duality gap as the convergence criterion. For more information, see Chapter 103.20, "Interior Point Algorithm." By default, TOLERANCE=1E–8.

BASELINE Statement

BASELINE <**OUT=***SAS-data-set*> <**COVARIATES=***SAS-data-set*> **;**

The BASELINE statement creates an output data set to contain the survival function estimates or the conditional quantile function estimates for every set of covariates ($\mathbf{x}$) in the COVARIATES= data set. If the COVARIATES= data set is not specified, PROC QUANTLIFE uses a reference set of covariates that consists of the reference levels for the CLASS variables and the average values for the continuous variables.

You can specify the following *options*:

COVARIATES=*SAS-data-set*
names the *SAS-data-set* that contains the sets of explanatory variable values for which the quantities of interest are estimated. All variables in the COVARIATES= data set are copied to the OUT= data set. Thus, the variables in the COVARIATES= data set can be used to identify the covariate sets in the OUT= data set.

OUT=*SAS-data-set*
specifies an output *SAS-data-set* that contains the estimated survival function and the estimated quantile function. If you omit the OUT= option, the data set is created and given a default name by using the DATA*n* convention. For more information, see the section "OUT= Output Data Set in the BASELINE Statement" on page 8516.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC QUANTLIFE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the QUANTLIFE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / **TRUNCATE** > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) **;**

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 102.2 summarizes the *options* available in the EFFECT statement.

Table 102.2 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are

computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

MODEL *response* < ∗ *censor*(*list*) > **=** < *effects* > < / *options* > **;**

The MODEL statement identifies the response variable, the optional censoring variable, and the explanatory effects, including covariates, main effects, interactions, and nested effects; for more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure." In the MODEL statement, the response variable precedes the equal sign. This name can optionally be followed by an asterisk, the name of the censoring variable, and a list of censoring values (separated by blanks or commas if you list more than one value) enclosed in parentheses. If the censoring variable takes on one of these values, the corresponding failure time is considered to be censored. Following the equal sign are the explanatory effects (sometimes called independent variables or covariates) for the model.

The censoring variable must be numeric.

Options

You can specify the following *options* after a slash (/):

NOINT
specifies no intercept regression.

QUANTILE=*number-list* | **PROCESS**
specifies the quantile levels of interest for quantile regression analysis. You can specify any number of quantile levels in the interval $(0, 1)$. You can also compute the entire quantile process by specifying the PROCESS option.

If you do not specify the QUANTILE= option, the QUANTLIFE procedure fits a median regression, which corresponds to QUANTILE=0.5.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > *keyword***=***name* < . . . *keyword***=***name* > **;**

The OUTPUT statement creates a *SAS-data-set* to contain statistics that are calculated after fitting models for all quantiles specified by the QUANTILE= option in the MODEL statement. At least one specification of the form *keyword***=***name* is required.

All variables in the original data set are included in the new data set, along with the variables that are created. These new variables contain fitted values and estimated quantiles.

You can use the following specifications in the OUTPUT statement:

OUT=*SAS-data-set*
specifies the new data set. By default, the QUANTLIFE procedure uses the DATA*n* convention to name the new data set. For more information, see the section "OUT= Output Data Set in the OUTPUT Statement" on page 8516.

keyword=*name* specifies the statistics to include in the output data set and gives names to the new variables. Specify a *keyword* for each desired statistic (see the following list of *keywords*), an equal sign, and the variable to contain the statistic.

You can specify the following *keywords*, which represent the indicated statistics:

PREDICTED | P specifies a variable to contain the predicted response.

RESIDUAL | RES specifies a variable to contain the residuals, $y_i - x_i'\hat{\beta}(\tau)$.

SAMPLEWEIGHT | SW specifies variables for sample weights from the bootstrap samples. For the *i*th sample, a column is added that contains the weights that are used for that sample. The name of this column is formed by appending an index *i* to the name that you specify. If you do not specify a name, then the default prefix is sw.

STDP specifies a variable to contain the estimates of the standard errors of the estimated response.

TEST Statement

< *label:* > **TEST** *effects* < / *options* > **;**

In quantile regression analysis, you might be interested in testing whether a covariate effect is statistically significant for a given quantile. You can use the TEST statement to obtain a test for the canonical linear hypothesis about the parameters of the tested *effects*,

$$\beta_j = 0, \quad j = i_1, \ldots, i_q$$

where q is the total number of parameters of the tested *effects*. The tested *effects* can be any set of effects in the MODEL statement.

You can include multiple TEST statements, provided that they appear after the MODEL statement. The optional *label*, which must be a valid SAS name, identifies output from the corresponding TEST statement. For more information about these tests, see the section "Testing Effects of Covariates" on page 8515.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement specifies a weight variable in the input data set.

To request weighted quantile regression, place the weights in a variable and specify the name in the WEIGHT statement. The values of the WEIGHT variable can be nonintegral and are not truncated. Observations that have nonpositive or missing values for the weight variable do not contribute to the fit of the model. For more information about weighted quantile regression, see the section "Details: QUANTREG Procedure" on page 8572 in Chapter 103, "The QUANTREG Procedure."

Details: QUANTLIFE Procedure

Notation for Censored Quantile Regression

Let T be a dependent variable, such as a survival time, and let x be a $p \times 1$ covariate vector. Quantile regression methods focus on modeling the conditional quantile function, $Q_T(\tau|x)$, which is defined as

$$Q_T(\tau|x) = \inf\{t : P(T \leq t|x) = \tau\},\ 0 < \tau < 1$$

For example, $Q_T(0.5|x)$ is the conditional median quantile, and $Q_T(0.95|x)$ is the conditional quantile function that corresponds to the 95th percentile.

A linear quantile regression model for $Q_T(\tau|x)$ has the form $x'\boldsymbol{\beta}(\tau)$. One of the advantages of quantile regression analysis is that the covariate effect $\boldsymbol{\beta}(\tau)$ can change with τ. Unlike ordinary least squares regression, which estimates the conditional expectation function $E(T|x)$, quantile regression offers the flexibility to model the entire conditional distribution.

Given observations $\{(T_i, x_i), i = 1, \ldots, n\}$, standard quantile regression estimates the regression coefficients $\beta(\tau)$ by minimizing the following objective function over b:

$$r(b) = \sum_{i=1}^{n} \rho_\tau(T_i - x_i' b)$$

where $\rho_\tau(u) = u(\tau - I(u < 0))$.

However, in many applications, the responses T_i are subject to censoring. For example, in a biomedical study, censoring occurs when patients withdraw from the study or die from a cause that is unrelated to the disease being studied.

Let C_i denote the censoring variable. In the case of right-censoring, the triples (x_i, Y_i, Δ_i) are observed, where $Y_i = \min(T_i, C_i)$ and $\Delta_i = I(T_i \leq C_i)$ are the observed response variable and the censoring indicator, respectively. Standard quantile regression can lead to a biased estimator of the regression parameters $\beta(\tau)$ when censoring occurs.

The following sections describe two methods for estimating the quantile coefficient $\beta(\tau)$ in the presence of right-censoring.

Kaplan-Meier-Type Estimator for Censored Quantile Regression

Portnoy (2003) proposes the use of weighted quantile regression to sequentially estimate $\beta(\tau_k)$ along the equally spaced grid $0 < \tau_1 < \cdots < \tau_M < 1$. You can request this method by specifying the METHOD=KM option in the PROC QUANTLIFE statement. The grid points $0 < \tau_1 < \cdots < \tau_M < 1$ are equally spaced, with τ_1 specified by the INITTAU= option and the step between adjacent grid points specified by the GRIDSIZE=option.

This method uses a weight function $w_i(\tau)$ for each censored observation. The weight function is constructed as follows: Let $\hat{\tau}_i$ be the first grid point at which $x_i'\hat{\beta}(\tau_i) \geq C_i$ and $x_i'\hat{\beta}(\tau_{i+1}) < C_i$; otherwise let $\hat{\tau}_i = 1$. When computing the τth quantile, assign weight $w_i(\tau) = \frac{\tau-\hat{\tau}_i}{1-\hat{\tau}_i}$ to the censored observation Y_i if $\tau > \hat{\tau}_i$; otherwise assign $w_i(\tau) = 1$. The algorithm for computing $\hat{\beta}(\tau_k), k = 1, \ldots, M$, is as follows:

1. Compute $\hat{\beta}(\tau_1)$ by using the standard quantile regression method.
2. For $k = 2, \ldots, M$, obtain $\hat{\beta}(\tau_k)$ sequentially by minimizing the following weighted quantile regression objective function:

$$\begin{aligned} r_w(b) &= \textstyle\sum_{\Delta_i=1} \rho_{\tau_k}(Y_i - x_i'b) \\ &\quad + \textstyle\sum_{\Delta_i=0} \left\{ w_i(\tau_k)\, \rho_{\tau_k}(Y_i - x_i'b) + (1 - w_i(\tau_k))\rho_{\tau_k}(Y^* - x_i'b) \right\} \end{aligned}$$

   where $w_i(\tau_k)$ is the weight for the right-censored observation Y_i at computing $\hat{\beta}(\tau_k)$, and the complementary weight $1 - w_i(\tau_k)$ is for Y^*, a large constant that is greater than all $x_i'\hat{\beta}(\tau)$.

The weighted quantile regression method is similar to Efron's redistribution-of-mass idea (Efron 1967) for the Kaplan-Meier estimator.

Note that if all observations are uncensored, $\hat{\beta}(\tau_k)$ is the same as the standard quantile regression estimator.

Nelson-Aalen-Type Estimator for Censored Quantile Regression

Peng and Huang (2008) propose a method of censored quantile regression that is based on the Nelson-Aalen estimator of the cumulative hazard function. Let $F_i(t|x) = P(T_i \leq t|x_i)$, $\Lambda_i(t|x) = -\log(1-F_i(t|x))$, and $N_i(t) = I\{\{T_i \leq t\}$ and $\{\Delta_i = 1\}\}$. Then the following equation is a martingale process that is associated with the counting process $N_i(t)$ (Fleming and Harrington 1991):

$$M_i(t) = N_i(t) - \Lambda_i(t \wedge Y_i|x)$$

Based on the martingale process, Peng and Huang (2008) derive the following estimating equation:

$$n^{-1/2} \sum_{i=1}^{n} x_i [N_i(\exp(x_i'\beta(\tau))) - \int_0^{\tau} I(Y_i \geq \exp(x_i'\beta(\tau)))dH(u)] = 0$$

where $H(u) = -\log(1-u)$ and $u \in [0, 1)$. By approximating the integral in the estimating equation on a grid $0 = \tau_0 < \tau_1 < \cdots < \tau_M < 1$, the regression quantiles $\beta(\tau_k)$, $k = 1, \ldots, M$, can be estimated sequentially by solving the following linear programming problem:

$$\min_b \{\alpha(\tau_k)'u + (\Delta - \alpha(\tau_k))'v \mid z = Xb + u - v, u \geq 0, v \geq 0\}$$

where

$$\alpha(\tau_k) = \sum_{j=1}^{k-1} I(Y_i \geq \exp(x_i'\hat{\beta}(\tau_j)))H((u_{j+1}) - H(u_j))$$

and X is the known matrix of x_i's. For more information, see Koenker (2008).

You can request this method by specifying the METHOD=NA option. The grid points $0 = \tau_0 < \tau_1 < \cdots < \tau_M < 1$ are equally spaced, with τ_1 specified by the INITTAU=option and the grid step between two adjacent grid points specified by the GRIDSIZE=option.

Relationship of Quantile Function and Survival Function

Both quantile function and survival function are useful in characterizing a lifetime distribution.

By the definition of the quantile function $Q_T(\tau|x)$,

$$F(Q_T(\tau|x)) = P(T \leq Q_T(\tau|x)) = \tau$$

In other words, the cumulative distribution function $F_T(t|x)$ maps $Q_T(\tau|x)$ to τ, and thus the corresponding survival function $S_T(t|x)$ maps $Q_T(\tau|x)$ to $1 - \tau$.

When you specify the LOG option, the QUANTLIFE procedure fits a linear quantile regression model for a log transformation of the lifetime as

$$Q_{\log(T)}(\tau|x) = x'\boldsymbol{\beta}(\tau)$$

where $Q_{\log(T)}(\tau|x)$ is the τth quantile of $\log(T)$ at x. The estimated quantile function for T given x is $\hat{Q}_T(\tau|x) = e^{x'\hat{\boldsymbol{\beta}}(\tau)}$, because the quantile function is invariant under a monotone transformation.

You can specify the covariates x in the COVARIATES= data set of the BASELINE statement and the PLOTS=(QUANTILE SURVIVAL) option in the PROC statement. Then the conditional quantile function at x is plotted as $\hat{Q}_T(\tau|x)$ against τ, and the conditional survival function at x is plotted as $1 - \tau$ against $\hat{Q}_T(\tau|x)$.

Confidence Interval

Direct computation of the covariance of the parameter estimators involves a complicated density estimation. Instead, the QUANTLIFE procedure computes confidence intervals for the quantile regression parameters $\beta(\tau)$ by using resampling methods. The QUANTLIFE procedure implements two different methods, the exponentially weighted method and the pairwise resampling method.

Exponentially Weighted Method

This method samples weights $w_i, i = 1, \ldots, n$, from a standard exponential distribution that has mean 1 and variance 1. Then it computes the censored quantile regression estimators $\hat{\beta}(\tau)$ based on the observed data (x_i, Y_i, Δ_i) with the weights w_i. These steps are repeated B times (where B is the value of the NREP= option in the PROC QUANTLIFE statement). The confidence intervals can be obtained from these B estimates. You can specify this method by using the CI=EW option in the PROC QUANTLIFE statement.

Pairwise Method

This method samples (x_i, Y_i, Δ_i) with replacement and computes the quantile regression estimators $\hat{\beta}(\tau)$ based on the resampled data. These steps are repeated B times (where B is the value of the NREP= option in the PROC QUANTLIFE statement). The confidence intervals can be obtained from these B estimates. You can specify this method by using the CI=PW option in the PROC QUANTLIFE statement.

Testing Effects of Covariates

Consider the linear model

$$y_i = \mathbf{x}_{1i}'\boldsymbol{\beta}_1 + \mathbf{x}_{2i}'\boldsymbol{\beta}_2 + \epsilon_i$$

where $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are p-dimensional and q-dimensional parameters, respectively, and ϵ_i, $i = 1, \ldots, n$, are errors. Denote $\mathbf{x}_i' = (\mathbf{x}_{1i}', \mathbf{x}_{2i}')$, and let $\hat{\boldsymbol{\beta}}_1(\tau)$ and $\hat{\boldsymbol{\beta}}_2(\tau)$ be the parameter estimates for $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$, respectively, at the τth quantile.

The QUANTLIFE procedure implements the Wald test for the null hypothesis:

$$H_0 : \beta_2(\tau) = 0$$

The Wald test statistic, which is based on the estimated coefficients $\hat{\beta}_2$ from the unrestricted fitted model, is given by

$$T_W(\tau) = \hat{\beta}_2'(\tau)\hat{\Sigma}(\tau)^{-1}\hat{\beta}_2(\tau)$$

where $\hat{\Sigma}(\tau)$ is an estimator of the covariance of $\hat{\beta}_2(\tau)$, which is obtained by using resampling methods.

Output Data Sets

OUTBOOTEST= Output Data Set

The OUTBOOTEST= data set contains parameter estimates for the specified model from resampled data sets. A set of observations is created for each quantile level and for each resampled data set.

If the QUANTLIFE procedure does not produce valid solutions, the parameter estimates are set to missing in the OUTBOOTEST= data set.

If created, this data set contains all variables that are specified in the MODEL statement. Each observation contains parameter estimates for a specified quantile level.

The following variables are also included in the data set:

- any specified BY variables
- _STATUS_, a character variable of length 12 that contains the status of the model fit: either NORMAL, NOUNIQUE, or NOVALID
- Intercept, a numeric variable that contains the intercept parameter estimates
- _TAU_, a numeric variable that contains the specified quantile levels from the MODEL statement

For continuous explanatory variables, the names of the parameters are the same as they are for the corresponding variables. For CLASS variables, the parameter names are obtained by concatenating the corresponding CLASS variable name to the CLASS category. For interaction and nested effects, the parameter names are created by concatenating the names of each component effect.

OUT= Output Data Set in the OUTPUT Statement

The OUT= data set that is specified in the OUTPUT statement contains all the variables in the input data set, along with statistics that you request by specifying *keyword=name* options. The additional variables are calculated for each observation in the input data set.

OUT= Output Data Set in the BASELINE Statement

The OUT= data set that is specified in the BASELINE statement contains all the variables in the COVARIATES= data set.

The following variables are also added to the data set:

QUANTILE	a numeric variable that contains the estimated quantile function
SURVIVAL	a numeric variable that contains the estimated survival function
TAU	a numeric variable that contains the quantile levels that are the complement of the survival function

ODS Table Names

Table 102.3 lists the names that the QUANTLIFE procedure assigns to each table that it creates. You can specify these names when you use the Output Delivery System (ODS) to select tables and create output data sets.

Table 102.3 ODS Tables Produced by PROC QUANTLIFE

ODS Table Name	Description	Statement	Option
ClassLevels	Classification variable levels	CLASS	Default
ModelInfo	Model information	MODEL	Default
NObs	Number of observations	PROC QUANTLIFE	Default
ParameterEstimates	Parameter estimates	MODEL	Default
CensoredSummary	Summary of event and censored observations	PROC QUANTLIFE	Default
Tests	Results of tests	TEST	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

The QUANTLIFE procedure assigns a name to each graph that it creates. You can use these names to refer to the graphs when you use ODS. The names along with the required statements and options are listed in Table 102.4.

Table 102.4 Graphs Produced by PROC QUANTLIFE

ODS Graph Name	Plot Description	PLOTS= Option
QuantilePlot	Quantile function plot	QUANTILE
QuantPanel	Panel of regression parameter estimates against quantile level with confidence limits	QUANTPLOT
ParamEstPlot	Scatter plot for regression parameter estimates against quantile level with confidence limits	QUANTPLOT / UNPACK
SurvivalPlot	Survival function plot	SURVIVAL

Examples: QUANTLIFE Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 102.1: Primary Biliary Cirrhosis Study

(View the complete code for this example (qlifeex1.sas) in the example repository.)

This example illustrates how to use quantile regression analysis to detect varying covariate effects on survival time. Consider a study of primary biliary cirrhosis, a rare but fatal chronic liver disease, discussed by Fleming and Harrington (1991). Researchers followed 418 patients who had this disease, 161 of whom died during the study.

The data set contains the following variables:

- Time, follow-up time, in years
- Status, event indicator, with value 1 for death time and value 0 for censored time
- Age, age from birth to study registration, in years
- Albumin, serum albumin level, in g/dl
- Bilirubin, serum bilirubin level, in mg/dl
- Edema, edema presence
- Protime, prothrombin time, in seconds

The following statements create the data set PBC, which is used in this example:

```
data pbc;
   input Time Status Age Albumin Bilirubin Edema Protime @@;
   label Time="Follow-Up Time in Days";
   logAlbumin   = log(Albumin);
   logBilirubin = log(Bilirubin);
   logProtime   = log(Protime);
   datalines;
  400 1 58.7652 2.60 14.5 1.0 12.2 4500 0 56.4463 4.14  1.1 0.0 10.6
 1012 1 70.0726 3.48  1.4 0.5 12.0 1925 1 54.7406 2.54  1.8 0.5 10.3
 1504 0 38.1054 3.53  3.4 0.0 10.9 2503 1 66.2587 3.98  0.8 0.0 11.0
 1832 0 55.5346 4.09  1.0 0.0  9.7 2466 1 53.0568 4.00  0.3 0.0 11.0
 2400 1 42.5079 3.08  3.2 0.0 11.0   51 1 70.5599 2.74 12.6 1.0 11.5
 3762 1 53.7139 4.16  1.4 0.0 12.0  304 1 59.1376 3.52  3.6 0.0 13.6

   ... more lines ...

  989 0 35.0000 3.23  0.7 0.0 10.8  681 1 67.0000 2.96  1.2 0.0 10.9
 1103 0 39.0000 3.83  0.9 0.0 11.2 1055 0 57.0000 3.42  1.6 0.0  9.9
```

```
  691 0 58.0000 3.75  0.8 0.0 10.4  976 0 53.0000 3.29  0.7 0.0 10.6
;
```

The next statements fit a linear model for the log of survival time of the PBC patients with the covariates logBilirubin, logProtime, logAlbumin, Age, and Edema:

```
ods graphics on;
proc quantlife data=pbc log method=na plot=(quantplot survival) seed=1268;
   model Time*Status(0)=logBilirubin logProtime logAlbumin Age Edema
                        /quantile=(.1 .2 .3 .4 .5 .6 .75);
run;
```

You use the QUANTILE= option to specify a set of quantiles of interest for comparing quantile-specific covariate effects. The METHOD= option specifies the Nelson-Aalen method for estimating the regression parameters.

The QUANTLIFE procedure provides resampling methods for computing confidence limits for the parameters; for more information, see the section "Confidence Interval" on page 8515. By default, the repetition number is 200. You can request a different number of repetitions by specifying the NREP= option. You can also use the SEED= option to specify the seed for generating random numbers so that you can later reproduce the results.

Figure 102.1.1 displays model information and information about censoring in the data. Out of 418 observations, 257 are censored.

Output 102.1.1 Model Information

The QUANTLIFE Procedure

Model Information	
Data Set	WORK.PBC
Dependent Variable	Log(Time)
Censoring Variable	Status
Censoring Value(s)	0
Number of Observations	418
Method	Nelson-Aalen
Replications	200
Seed for Random Number Generator	1268

Summary of the Number of Event and Censored Values			
Total	**Event**	**Censored**	**Percent Censored**
418	161	257	61.48

Figure 102.1.2 provides the parameter estimates. Each quantile level has a set of parameter estimates and confidence limits.

Output 102.1.2 Parameter Estimates at Different Quantiles

Parameter Estimates								
Quantile	Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
0.1000	Intercept	1	14.8030	4.0967	6.7736	22.8325	3.61	0.0003
	logBilirubin	1	-0.4488	0.1485	-0.7398	-0.1578	-3.02	0.0027
	logProtime	1	-3.6378	1.4560	-6.4915	-0.7841	-2.50	0.0129
	logAlbumin	1	1.9286	0.9756	0.0165	3.8408	1.98	0.0487
	Age	1	-0.0244	0.0107	-0.0455	-0.00334	-2.27	0.0237
	Edema	1	-1.0712	0.6688	-2.3820	0.2396	-1.60	0.1100
0.2000	Intercept	1	15.1800	2.6664	9.9540	20.4060	5.69	<.0001
	logBilirubin	1	-0.6532	0.0886	-0.8268	-0.4796	-7.37	<.0001
	logProtime	1	-3.3273	0.9401	-5.1699	-1.4847	-3.54	0.0004
	logAlbumin	1	1.6842	0.6888	0.3343	3.0342	2.45	0.0149
	Age	1	-0.0291	0.00687	-0.0425	-0.0156	-4.23	<.0001
	Edema	1	-0.7265	0.3179	-1.3497	-0.1034	-2.29	0.0228
0.3000	Intercept	1	13.2382	2.5296	8.2804	18.1961	5.23	<.0001
	logBilirubin	1	-0.6013	0.0762	-0.7506	-0.4521	-7.90	<.0001
	logProtime	1	-2.5816	0.8907	-4.3273	-0.8359	-2.90	0.0039
	logAlbumin	1	1.7246	0.7142	0.3248	3.1245	2.41	0.0162
	Age	1	-0.0244	0.00716	-0.0385	-0.0104	-3.41	0.0007
	Edema	1	-0.8577	0.2763	-1.3992	-0.3163	-3.10	0.0020
0.4000	Intercept	1	13.4716	3.0874	7.4204	19.5228	4.36	<.0001
	logBilirubin	1	-0.6047	0.0846	-0.7705	-0.4389	-7.15	<.0001
	logProtime	1	-2.1632	1.1726	-4.4615	0.1351	-1.84	0.0658
	logAlbumin	1	0.9819	0.7191	-0.4274	2.3912	1.37	0.1728
	Age	1	-0.0255	0.00681	-0.0389	-0.0122	-3.74	0.0002
	Edema	1	-1.0589	0.3104	-1.6672	-0.4506	-3.41	0.0007
0.5000	Intercept	1	10.9205	2.8047	5.4235	16.4175	3.89	0.0001
	logBilirubin	1	-0.5315	0.0904	-0.7087	-0.3543	-5.88	<.0001
	logProtime	1	-1.2222	1.2142	-3.6020	1.1577	-1.01	0.3148
	logAlbumin	1	1.5700	0.6284	0.3383	2.8016	2.50	0.0129
	Age	1	-0.0318	0.00883	-0.0491	-0.0145	-3.60	0.0004
	Edema	1	-0.7316	0.3743	-1.4653	0.00202	-1.95	0.0513
0.6000	Intercept	1	11.2381	2.6294	6.0846	16.3917	4.27	<.0001
	logBilirubin	1	-0.5701	0.0852	-0.7370	-0.4031	-6.69	<.0001
	logProtime	1	-1.3508	1.1402	-3.5856	0.8840	-1.18	0.2368
	logAlbumin	1	1.3704	0.5091	0.3726	2.3682	2.69	0.0074
	Age	1	-0.0226	0.0109	-0.0440	-0.00111	-2.06	0.0399
	Edema	1	-0.5141	0.3088	-1.1193	0.0912	-1.66	0.0968
0.7500	Intercept	1	10.0954	3.1893	3.8445	16.3463	3.17	0.0017
	logBilirubin	1	-0.6366	0.1071	-0.8466	-0.4267	-5.94	<.0001
	logProtime	1	-0.9670	1.2343	-3.3862	1.4521	-0.78	0.4338
	logAlbumin	1	1.8148	0.5883	0.6618	2.9678	3.08	0.0022
	Age	1	-0.0203	0.0156	-0.0509	0.0102	-1.30	0.1931
	Edema	1	-0.3529	0.3120	-0.9644	0.2586	-1.13	0.2587

For comparison, the following statements use the LIFEREG procedure to fit a Weibull distribution to the

data. The LIFEREG procedure fits an accelerated failure time model, which assumes that the independent variables have a multiplicative effect on the event time.

```
proc lifereg data=pbc;
   model Time*Status(0)=logBilirubin logProtime logAlbumin Age Edema;
run;
```

Figure 102.1.3 provides the parameter estimates that are computed by PROC LIFEREG.

Output 102.1.3 Parameter Estimates from PROC LIFEREG

The LIFEREG Procedure

Analysis of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	12.2155	1.4539	9.3658	15.0651	70.59	<.0001
logBilirubin	1	-0.5770	0.0556	-0.6861	-0.4680	107.55	<.0001
logProtime	1	-1.7565	0.5248	-2.7850	-0.7280	11.20	0.0008
logAlbumin	1	1.6694	0.4276	0.8313	2.5074	15.24	<.0001
Age	1	-0.0265	0.0053	-0.0368	-0.0162	25.35	<.0001
Edema	1	-0.6303	0.1805	-0.9842	-0.2764	12.19	0.0005
Scale	1	0.6807	0.0430	0.6014	0.7704		
Weibull Shape	1	1.4691	0.0928	1.2980	1.6628		

The p-value for logProtime is very small. For this same variable, the p-values that result from the quantile regression analysis are 0.3148 for the 0.5th quantile and 0.4338 for the 0.75th quantile, and the p-values are much smaller for the lower quantiles. Apparently, the effect of this covariate depends on which side of the response distribution is being modeled.

The PLOT=QUANTPLOT option in the PROC QUANTLIFE statement requests the quantile process plots in Figure 102.1.4, which plot the estimated regression parameter against the quantile level. You can use these plots to compare quantile-specific covariate effects. If the curve is not constant, it can indicate heterogeneity in the data. The interpretation of the regression coefficients at a given quantile is similar to that of classical regression analysis. That is, the coefficient from a given covariate indicates the effect on log(Time) of a unit change in that covariate, assuming that the other covariates are fixed.

In Figure 102.1.4, you can see that the effect of logProtime has a negative effect over the lower quantiles, which diminishes in magnitude at the median and upper quantiles. This insight would be missed if you were using the accelerated failure model.

Output 102.1.4 Quantile Processes with 95% Confidence Bands

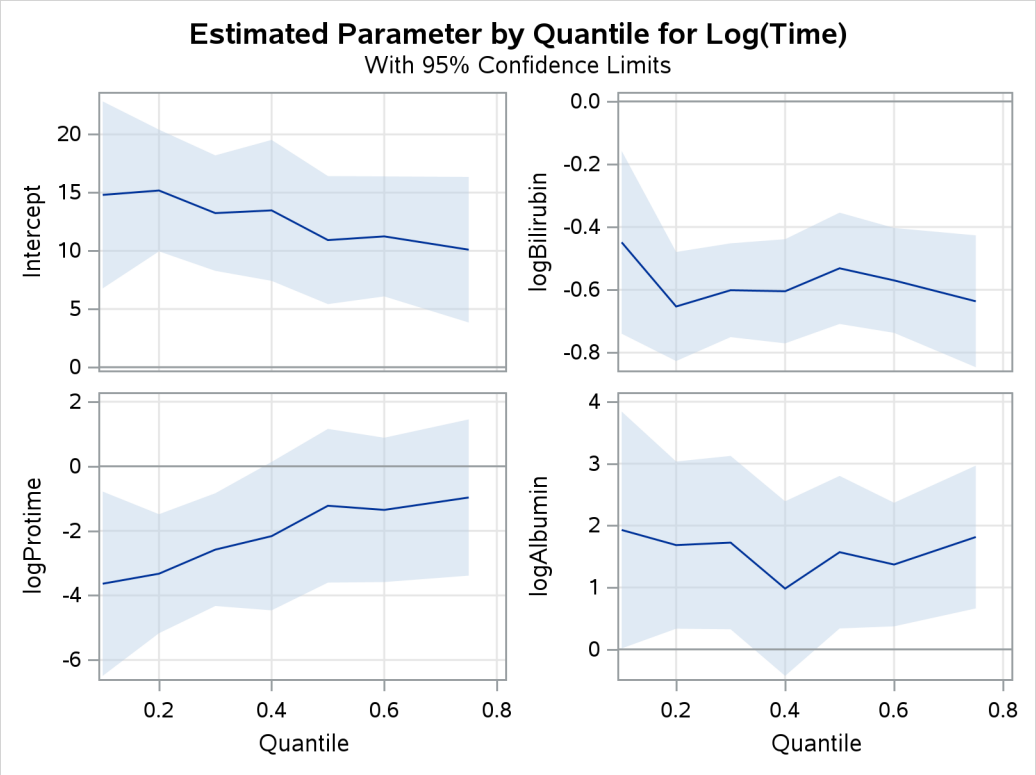

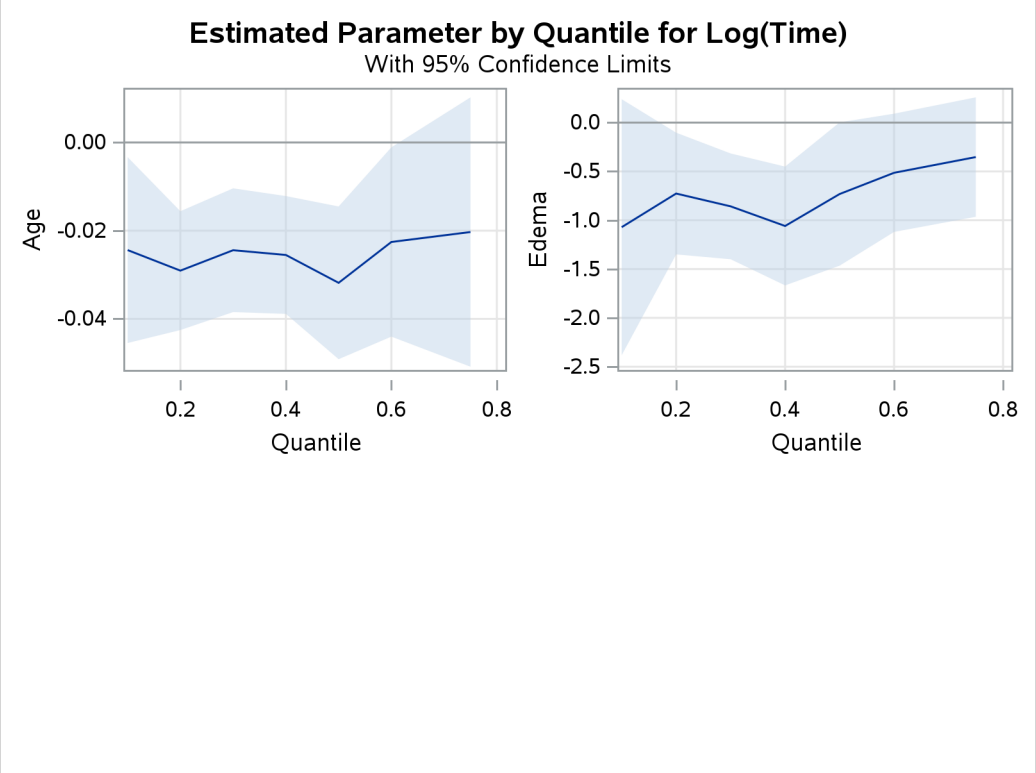

Example 102.2: Drug Abuse Study

(View the complete code for this example (qlifeex2.sas) in the example repository.)

This example reproduces analysis done by Portnoy (2003), which demonstrates how to use quantile regression to analyze survival times. The example uses drug abuse data from Hosmer and Lemeshow (1999). The goal of this study is to compare treatment effects on reducing drug abuse.

The data set contains the following variables:

- Time, time to return to drug use, in days
- Status, event indicator, with value 1 for return to drug use and value 0 for censored time
- Age, age at enrollment, in years
- Treatment, with value 1 for six-month treatment and value 0 for three-month treatment
- Beck, Beck Depression Inventory score at admission to the program
- IV3, indicator of recent intravenous drug use
- NDT, number of prior drug treatments
- Race, race indicator, with value 1 for white and value 0 for nonwhite
- Site, treatment sites (A and B)
- LOT, length of treatment, in days

The following statements create the data set UIS:

```
data uis;
   input  ID Age Becktota Hercoc Ivhx Ndrugtx Race Treat Site Lot Time
   Censor;
   Iv3 = (Ivhx = 3);
   Nd1 = 1/((Ndrugtx+1)/10);
   Nd2 = (1/((Ndrugtx+1)/10))*log((Ndrugtx+1)/10);
   if (Treat =1 ) then Frac = Lot/180;
   else  Frac = Lot/90;
   datalines;
     1  39   9.0000    4     3      1     0    1     0  123  188    1
     2  33  34.0000    4     2      8     0    1     0   25   26    1
     3  33  10.0000    2     3      3     0    1     0    7  207    1
     4  32  20.0000    4     3      1     0    0     0   66  144    1

   ... more lines ...

   626  28   10.0      4     2      3     0    1     1   21   35    1
   627  35   17.0      1     3      2     0    0     1  184  379    1
   628  46   31.5      1     3     15     1    1     1    9  377    1
;
```

The following statements replicate the analysis of Portnoy (2003):

```
ods graphics on;
proc quantlife data=uis log seed=999 plots=(quantplot survival);
   class Race Site Treat;
   model Time*Censor(0)=Nd1  Nd2  Iv3 Becktota
                        Treat  Frac  Race  Age|Site
                      / quantile=0.05 to 0.85 by 0.05 ;
   baseline out=Predsurvf;

run;
```

Figure 102.2.1 displays the model information. Out of 628 subjects, 53 contain missing values and are not included in the analysis. The censoring rate is 20.87%.

Output 102.2.1 Model Information

The QUANTLIFE Procedure

Model Information	
Data Set	WORK.UIS
Dependent Variable	Log(Time)
Censoring Variable	Censor
Censoring Value(s)	0
Number of Observations	575
Method	Kaplan-Meier
Replications	200
Seed for Random Number Generator	999

Class Level Information		
Name	Levels	Values
Race	2	0 1
Site	2	0 1
Treat	2	0 1

Summary of the Number of Event and Censored Values			
Total	Event	Censored	Percent Censored
575	464	111	19.30

Figure 102.2.2 displays regression quantile process plots for each covariate.

Output 102.2.2 Quantile Processes with 95% Confidence Bands

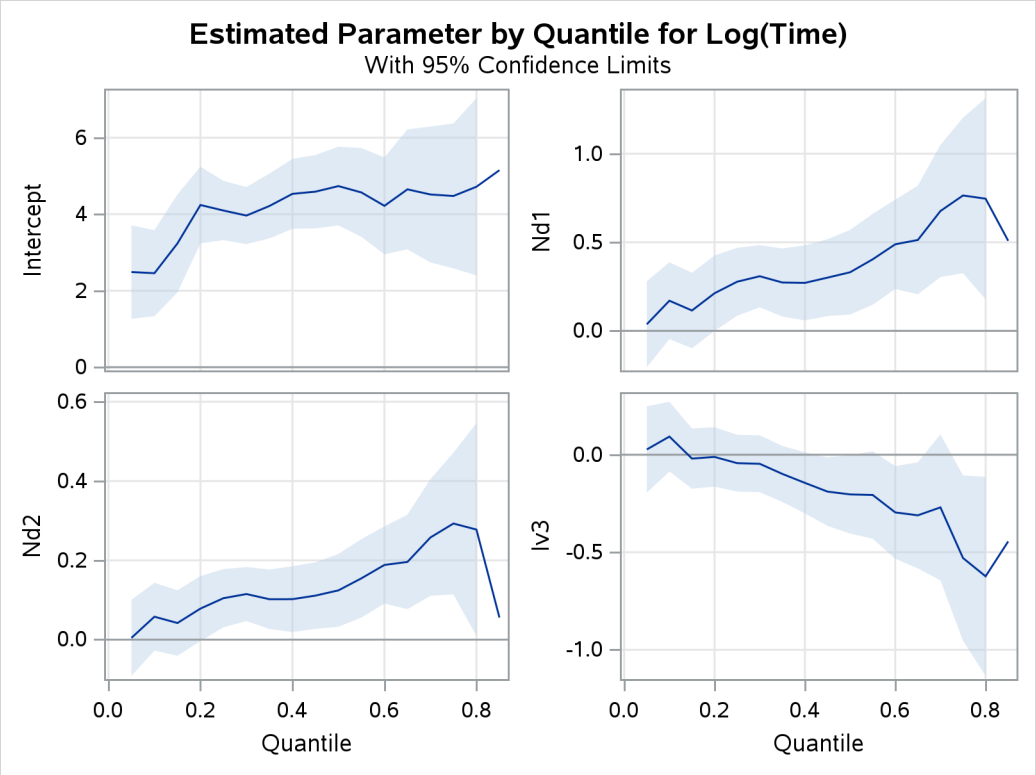

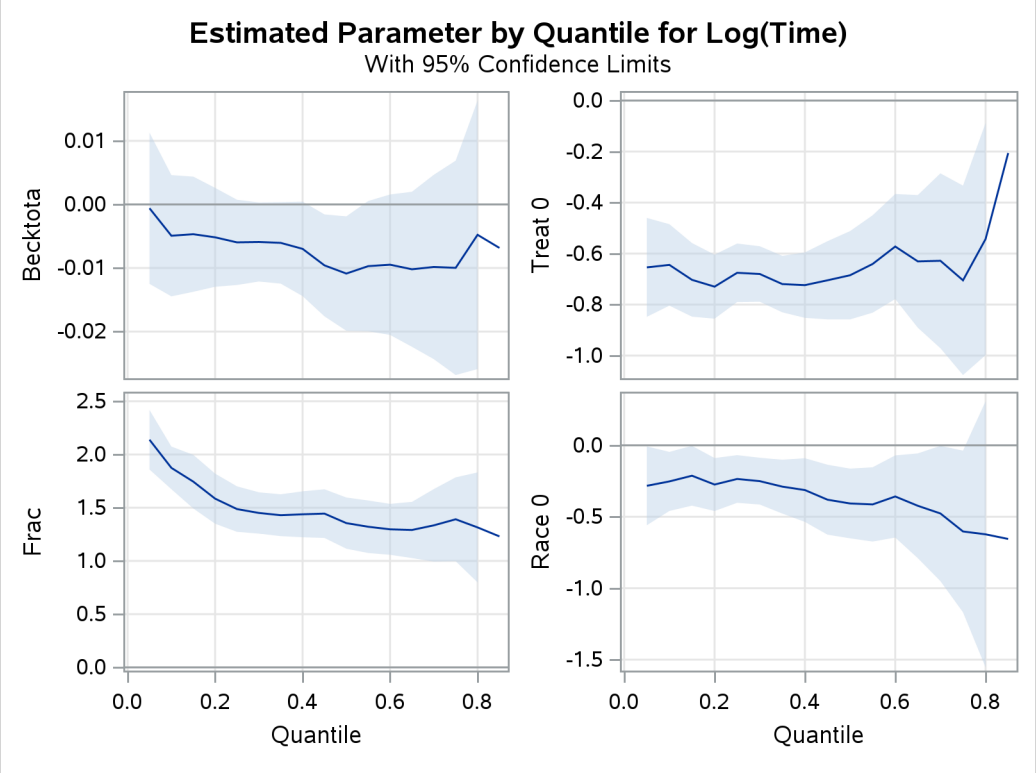

Output 102.2.2 *continued*

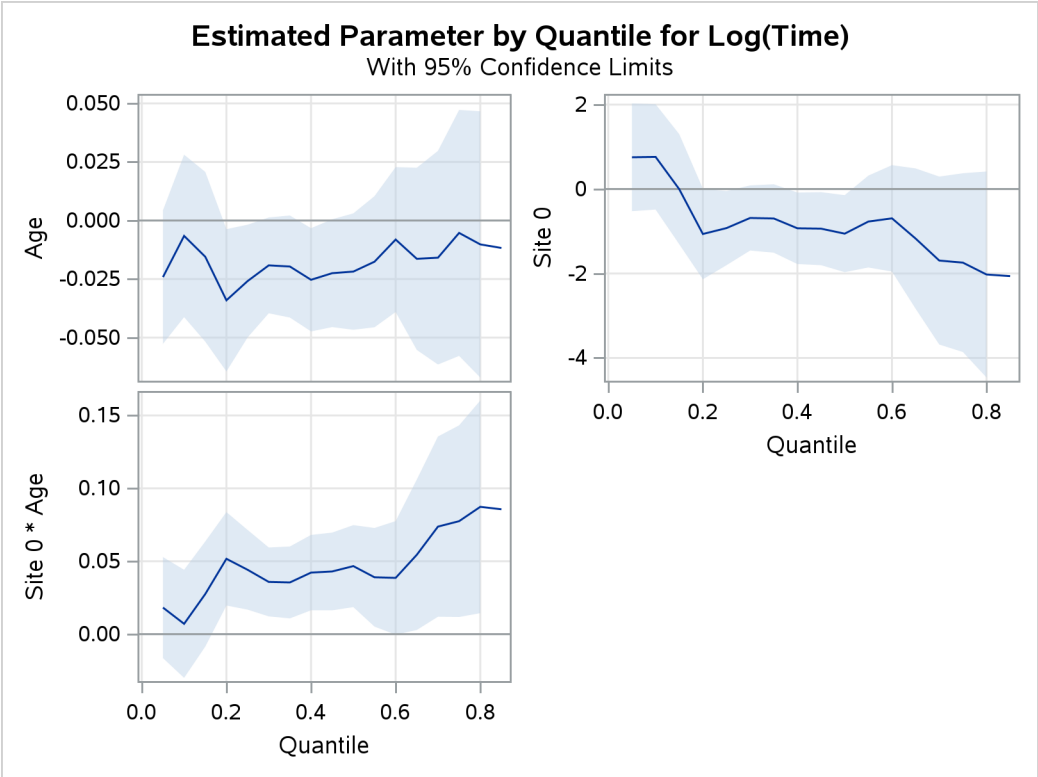

You can see the varying effects for Nd and Frac, whereas the treatment effect is fairly constant. For more information about the covariate effects that can be discovered using quantile regression, see Portnoy (2003).

In survival analysis, a plot of the estimated survival function is often of interest. There is a one-to-one relationship between the quantile function and the survival function. When you specify the PLOTS= SURVIVAL option, the QUANTLIFE procedure estimates the survival function by fitting a quantile regression model for a grid of equally spaced quantile levels. You can specify the grid points by using the INITTAU=option and the step between adjacent grid points by using the GRIDSIZE=option. For more information, see the section "Kaplan-Meier-Type Estimator for Censored Quantile Regression" on page 8513.

Figure 102.4 shows the estimated survival function at the reference set of covariate values that consist of reference levels for the CLASS variables and average values for the continuous variables. You can output the predicted survival function by specifying the SURVIVAL= option in the BASELINE statement.

Output 102.2.3 Survival Function

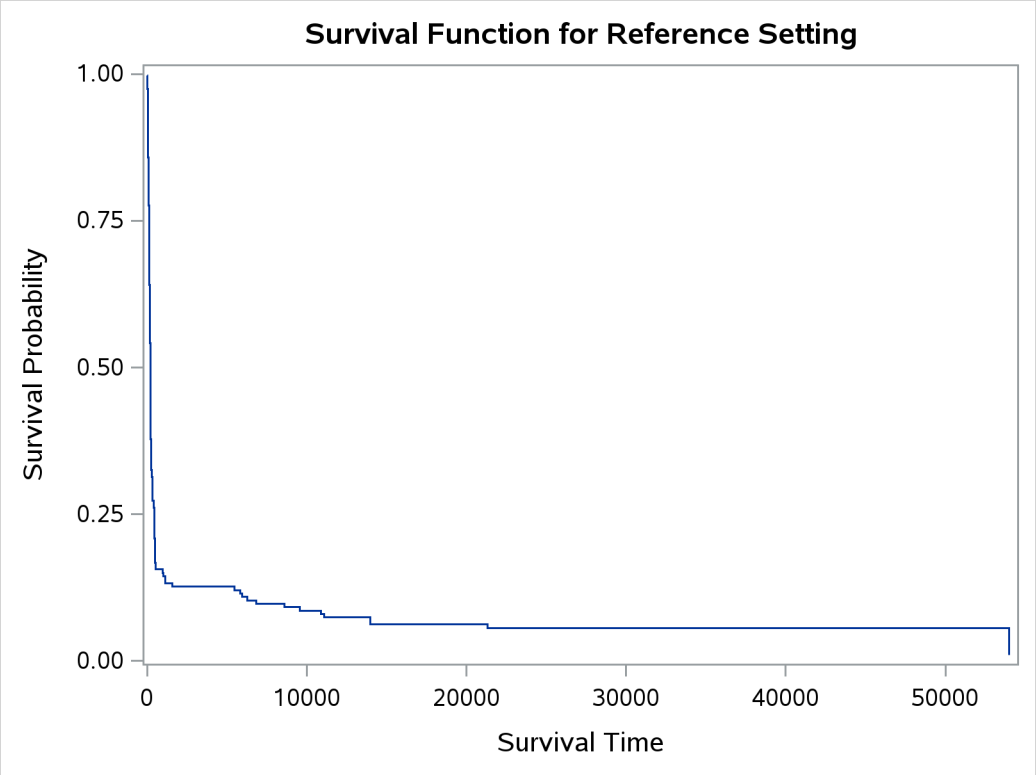

References

Chernozhukov, V., Fernandez-Val, I., and Galichon, A. (2009). "Improving Point and Interval Estimators of Monotone Functions by Rearrangement." *Biometrika* 96:559–575.

Efron, B. (1967). "The Two Sample Problem with Censored Data." In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 4, 831–853. Berkeley: University of California Press.

Fleming, T. R., and Harrington, D. P. (1991). *Counting Processes and Survival Analysis*. New York: John Wiley & Sons.

Hosmer, D. W., Jr., and Lemeshow, S. (1999). *Applied Survival Analysis: Regression Modeling of Time-to-Event Data*. New York: John Wiley & Sons.

Hosmer, D. W., Jr., Lemeshow, S., and May, S. (2008). *Applied Survival Analysis: Regression Modeling of Time-to-Event Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Huang, Y. (2010). "Quantile Calculus and Censored Regression." *Annals of Statistics* 38:1607–1637.

Klein, J. P., and Moeschberger, M. L. (2003). *Survival Analysis: Techniques for Censored and Truncated Data*. 2nd ed. New York: Springer-Verlag.

Koenker, R. (2008). "Censored Quantile Regression Redux." *Journal of Statistical Software* 27:1–24.

Koenker, R., and Bassett, G. W. (1978). "Regression Quantiles." *Econometrica* 46:33–50.

Koenker, R., and Geling, O. (2001). "Reappraising Medfly Longevity: A Quantile Regression Survival Analysis." *Journal of the American Statistical Association* 96:458–468.

Peng, L., and Huang, Y. (2008). "Survival Analysis with Quantile Regression Models." *Journal of the American Statistical Association* 103:637–649.

Portnoy, S. (2003). "Censored Regression Quantiles." *Journal of the American Statistical Association* 98:1001–1012.

Chapter 103
The QUANTREG Procedure

Contents

Overview: QUANTREG Procedure . . . **8530**
- Features . . . 8532
- Quantile Regression . . . 8533

Getting Started: QUANTREG Procedure . . . **8535**
- Analysis of Fish-Habitat Relationships . . . 8535
- Growth Charts for Body Mass Index . . . 8543

Syntax: QUANTREG Procedure . . . **8547**
- PROC QUANTREG Statement . . . 8548
- BY Statement . . . 8554
- CLASS Statement . . . 8554
- CONDDIST Statement . . . 8555
- EFFECT Statement . . . 8562
- ESTIMATE Statement . . . 8564
- ID Statement . . . 8565
- MODEL Statement . . . 8565
- OUTPUT Statement . . . 8569
- PERFORMANCE Statement . . . 8570
- TEST Statement . . . 8571
- WEIGHT Statement . . . 8572

Details: QUANTREG Procedure . . . **8572**
- Quantile Regression as an Optimization Problem . . . 8572
- Optimization Algorithms . . . 8573
- Confidence Interval . . . 8581
- Covariance-Correlation . . . 8585
- Linear Test . . . 8585
- Estimating Probability Functions by Using the CONDDIST Statement . . . 8587
- Leverage Point and Outlier Detection . . . 8594
- INEST= Data Set . . . 8595
- OUTEST= Data Set . . . 8596
- Computational Resources . . . 8596
- ODS Table Names . . . 8597
- ODS Graphics . . . 8598

Examples: QUANTREG Procedure . . . **8604**
- Example 103.1: Comparison of Algorithms . . . 8604
- Example 103.2: Quantile Regression for Econometric Growth Data . . . 8608
- Example 103.3: Quantile Regression Analysis of Birth-Weight Data . . . 8616

Example 103.4: Nonparametric Quantile Regression for Ozone Levels 8621
Example 103.5: Quantile Polynomial Regression for Salary Data 8623
Example 103.6: Counterfactual Analysis of Smoking-Weight Data 8627
References **8634**

Overview: QUANTREG Procedure

The QUANTREG procedure uses quantile regression to model the effects of covariates on the conditional quantiles of a response variable.

Quantile regression was introduced by Koenker and Bassett (1978) as an extension of ordinary least squares (OLS) regression, which models the relationship between one or more covariates X and the *conditional mean* of the response variable Y given $X = x$. Quantile regression extends the OLS regression to model the *conditional quantiles* of the response variable, such as the median or the 90th percentile. Quantile regression is particularly useful when the rate of change in the conditional quantile, expressed by the regression coefficients, depends on the quantile.

Figure 103.1 Trout Density in Streams

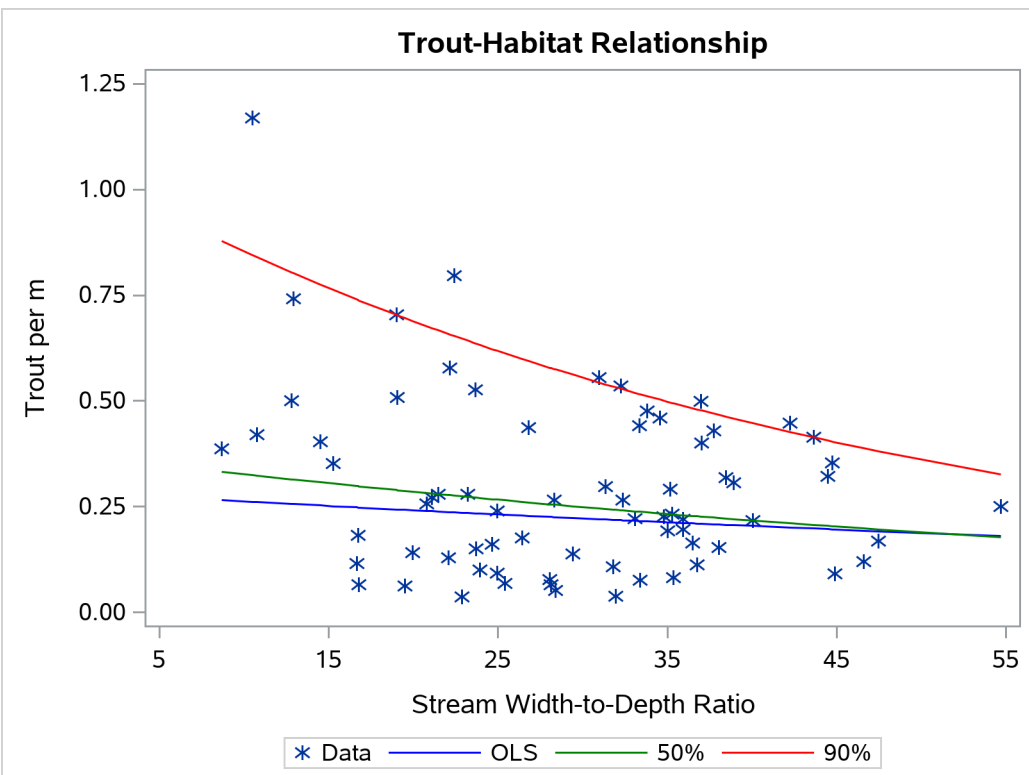

Figure 103.1 illustrates an ecological study in which modeling upper conditional quantiles reveals additional information. The points represent measurements of trout density and stream width-to-depth ratio that were taken at 13 streams over seven years.

As analyzed by Dunham, Cade, and Terrell (2002), both the ratio and the trout density depend on a number of unmeasured limiting factors that are related to the integrity of stream habitat. The interaction of these factors results in unequal variances for the conditional distributions of density given the ratio. When the ratio is the "active" limiting effect, changes in the upper conditional percentiles of density provide a better estimate of this effect than changes in the conditional mean.

The red and green curves represent the conditional 90th and 50th percentiles of density as determined by the QUANTREG procedure. The analysis was done by using a simple linear regression model for the logarithm of density. (The curves in Figure 103.1 were obtained by transforming the fitted lines back to the original scale. For more information, see the section "Analysis of Fish-Habitat Relationships" on page 8535.) The slope parameter for the 90th percentile has an estimated value of –0.0215 and is significant with a p-value less than 0.01. On the other hand, the slope parameter for the 50th percentile is not significantly different from 0. Similarly, the slope parameter for the mean, which is obtained with OLS regression, is not significantly different from 0.

Figure 103.2 Percentiles for Body Mass Index

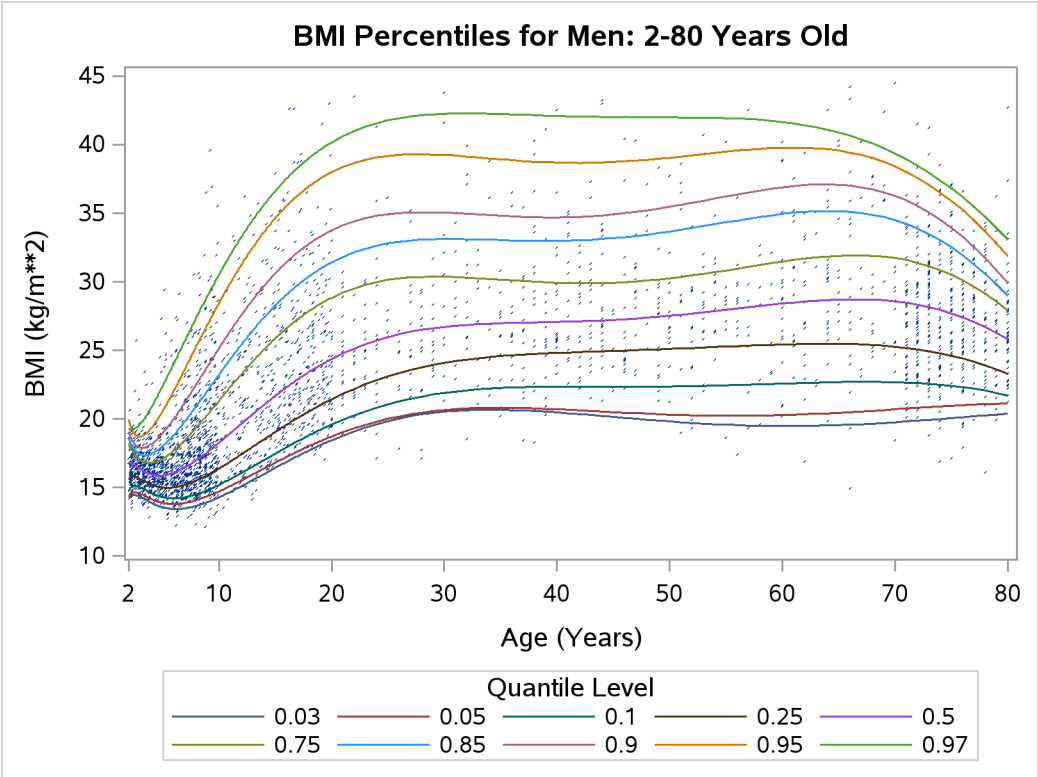

Quantile regression is especially useful when the data are heterogeneous in the sense that the tails and the central location of the conditional distributions vary differently with the covariates. An even more pronounced example of heterogeneity is shown in Figure 103.2, which plots the body mass index of 8,250 men versus their age.

Here, both upper (overweight) and lower (underweight) conditional quantiles are important because they provide the basis for developing growth charts and establishing health standards. The curves in Figure 103.2 were determined by using the QUANTREG procedure to perform polynomial quantile regression. For more information, see the section "Growth Charts for Body Mass Index" on page 8543. Clearly, the rate of change

with age (as expressed by the regression coefficients), particularly for ages less than 20, is different for each conditional quantile.

Heterogeneous data occur in many fields, including biomedicine, econometrics, survival analysis, and ecology. Quantile regression, which includes median regression as a special case, provides a complete picture of the covariate effect when a set of percentiles is modeled. So it can capture important features of the data that might be missed by models that average over the conditional distribution.

Because it makes no distributional assumption about the error term in the model, quantile regression offers considerable model robustness. The assumption of normality, which is often made with OLS regression in order to compute conditional quantiles as offsets from the mean, forces a common set of regression coefficients for all the quantiles. Obviously, quantiles with common slopes would be inappropriate in the preceding examples.

Quantile regression is also flexible because it does not involve a link function that relates the variance and the mean of the response variable. Generalized linear models, which you can fit with the GENMOD procedure, require both a link function and a distributional assumption such as the normal or Poisson distribution. The goal of generalized linear models is inference about the regression parameters in the linear predictor for the mean of the population. In contrast, the goal of quantile regression is inference about regression coefficients for the conditional quantiles of a response variable that is usually assumed to be continuous.

Quantile regression also offers a degree of data robustness. Unlike OLS regression, quantile regression is robust to extreme points in the response direction (outliers). However, it is not robust to extreme points in the covariate space (leverage points). When both types of robustness are of concern, consider using the ROBUSTREG procedure (Chapter 107, "The ROBUSTREG Procedure.")

Unlike OLS regression, quantile regression is equivariant to monotone transformations of the response variable. For example, as illustrated in the trout example, the logarithm of the 90th conditional percentile of trout density is the 90th conditional percentile of the logarithm of density.

Quantile regression cannot be carried out simply by segmenting the unconditional distribution of the response variable and then obtaining least squares fits for the subsets. This approach leads to disastrous results when, for example, the data include outliers. In contrast, quantile regression uses *all* of the data for fitting quantiles, even the extreme quantiles.

Features

The main features of the QUANTREG procedure are as follows:

- supports single-level quantile regression analysis and quantile process regression analysis
- offers simplex, interior point, and smoothing algorithms for estimation
- supports the parametric simplex algorithm for quantile process regression that estimates the optimal quantile-level grid
- supports the interior point and smoothing algorithms for fast quantile process regression that approximate the process by using a prespecified equally spaced quantile-level grid
- provides sparsity, rank, and resampling methods for confidence intervals

- provides asymptotic and bootstrap methods for covariance and correlation matrices of the estimated parameters
- provides the Wald, likelihood ratio, and rank tests for the regression parameter estimates and the Wald test for heteroscedasticity
- provides outlier and leverage-point diagnostics
- enables parallel computing when multiple processors are available
- provides rowwise or columnwise output data sets with multiple quantiles
- provides regression quantile spline fits
- produces fit plots, diagnostic plots, and quantile process plots by using ODS Graphics
- supports conditional and marginal distribution estimation of the response random variable

The next section provides notation and a formal definition for quantile regression.

Quantile Regression

Quantile regression generalizes the concept of a univariate quantile to a conditional quantile given one or more covariates. Recall that a student's score on a test is at the τ quantile if his or her score is better than that of $100\tau\%$ of the students who took the test. The score is also said to be at the 100τth percentile.

For a random variable Y with probability distribution function

$$F(y) = \text{ Prob } (Y \leq y)$$

the τ quantile of Y is defined as the inverse function

$$Q(\tau) = \text{ inf } \{y : F(y) \geq \tau\}$$

where the quantile level τ ranges between 0 and 1. In particular, the median is $Q(1/2)$.

For a random sample $\{y_1, \ldots, y_n\}$ of Y, it is well known that the sample median minimizes the sum of absolute deviations:

$$\text{median} = \arg\min_{\xi \in \mathbf{R}} \sum_{i=1}^{n} |y_i - \xi|$$

Likewise, the general τ sample quantile $\xi(\tau)$, which is the analog of $Q(\tau)$, is formulated as the minimizer

$$\xi(\tau) = \arg\min_{\xi \in \mathbf{R}} \sum_{i=1}^{n} \rho_\tau(y_i - \xi)$$

where $\rho_\tau(z) = z(\tau - I(z < 0))$, $0 < \tau < 1$, and where $I(\cdot)$ denotes the indicator function. The loss function ρ_τ assigns a weight of τ to positive residuals $y_i - \xi$ and a weight of $1 - \tau$ to negative residuals.

Using this loss function, the linear conditional quantile function extends the τ sample quantile $\xi(\tau)$ to the regression setting in the same way that the linear conditional mean function extends the sample mean. Recall that OLS regression estimates the linear conditional mean function $E(Y|X = \mathbf{x}) = \mathbf{x}'\boldsymbol{\beta}$ by solving for

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta} \in \mathbf{R}^p} \sum_{i=1}^{n} (y_i - \mathbf{x}_i'\boldsymbol{\beta})^2$$

The estimated parameter $\hat{\boldsymbol{\beta}}$ minimizes the sum of squared residuals in the same way that the sample mean $\hat{\mu}$ minimizes the sum of squares:

$$\hat{\mu} = \arg\min_{\mu \in \mathbf{R}} \sum_{i=1}^{n} (y_i - \mu)^2$$

Likewise, quantile regression estimates the linear conditional quantile function, $Q_\tau(Y|X = \mathbf{x}) = Q_{Y|\mathbf{x}}(\tau) = \mathbf{x}'\boldsymbol{\beta}(\tau)$, by solving the following equation for $\tau \in (0, 1)$:

$$\hat{\boldsymbol{\beta}}(\tau) = \arg\min_{\boldsymbol{\beta} \in \mathbf{R}^p} \sum_{i=1}^{n} \rho_\tau(y_i - \mathbf{x}_i'\boldsymbol{\beta})$$

The quantity $\hat{\boldsymbol{\beta}}(\tau)$ is called the τ *regression quantile*. The case $\tau = 0.5$ (which minimizes the sum of absolute residuals) corresponds to median regression (which is also known as L_1 regression).

The following set of regression quantiles is referred to as the *quantile process*:

$$\{\boldsymbol{\beta}(\tau) : \tau \in (0, 1)\}$$

The QUANTREG procedure computes the quantile function $Q_{Y|\mathbf{x}}(\tau)$ and conducts statistical inference on the estimated parameters $\hat{\boldsymbol{\beta}}(\tau)$.

Getting Started: QUANTREG Procedure

The following examples demonstrate how you can use the QUANTREG procedure to fit linear models for selected quantiles or for the entire quantile process. The first example explains the use of the procedure in a fish-habitat example, and the second example explains the use of the procedure to construct growth charts for body mass index.

Analysis of Fish-Habitat Relationships

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/qreggs1.sas.)

Quantile regression is used extensively in ecological studies (Cade and Noon 2003). Recently, Dunham, Cade, and Terrell (2002) applied quantile regression to analyze fish-habitat relationships for Lahontan cutthroat trout in 13 streams of the eastern Lahontan basin, which covers most of northern Nevada and parts of southern Oregon. The density of trout (number of trout per meter) was measured by sampling stream sites from 1993 to 1999. The width-to-depth ratio of the stream site was determined as a measure of stream habitat.

The goal of this study was to explore the relationship between the conditional quantiles of trout density and the width-to-depth ratio. The scatter plot of the data in Figure 103.1 indicates a nonlinear relationship, so it is reasonable to fit regression models for the conditional quantiles of the log of density. Because regression quantiles are equivariant under any monotonic (linear or nonlinear) transformation (Koenker and Hallock 2001), the exponential transformation converts the conditional quantiles to the original density scale.

The data set trout, which follows, includes the average numbers of Lahontan cutthroat trout per meter of stream (Density), the logarithm of Density (LnDensity), and the width-to-depth ratios (WDRatio) for 71 samples:

```
data trout;
   input Density WDRatio LnDensity @@;
   datalines;
 0.38732     8.6819    -0.94850    1.16956    10.5102     0.15662
 0.42025    10.7636    -0.86690    0.50059    12.7884    -0.69197
 0.74235    12.9266    -0.29793    0.40385    14.4884    -0.90672
 0.35245    15.2476    -1.04284    0.11499    16.6495    -2.16289
 0.18290    16.7188    -1.69881    0.06619    16.7859    -2.71523
 0.70330    19.0141    -0.35197    0.50845    19.0548    -0.67639

   ... more lines ...

 0.25125    54.6916    -1.38129
;
```

The following statements use the QUANTREG procedure to fit a simple linear model for the 50th and 90th percentiles of LnDensity:

```
ods graphics on;

proc quantreg data=trout alpha=0.1 ci=resampling;
   model LnDensity = WDRatio / quantile=0.5 0.9
```

```
                                CovB seed=1268;
   test WDRatio / wald lr;
run;
```

The MODEL statement specifies a simple linear regression model with LnDensity as the response variable Y and WDRatio as the covariate X. The QUANTILE= option requests that the regression quantile function $Q_{Y|\mathbf{x}}(\tau) = \mathbf{x}'\boldsymbol{\beta}(\tau)$ be estimated by solving the following equation, where $\tau = (0.5, 0.9)$:

$$\hat{\boldsymbol{\beta}}(\tau) = \arg\min_{\boldsymbol{\beta}\in\mathbf{R}^2} \sum_{i=1}^{n} \rho_\tau(y_i - \mathbf{x}_i'\boldsymbol{\beta})$$

By default, the regression coefficients $\hat{\boldsymbol{\beta}}(\tau)$ are estimated by using the simplex algorithm, which is explained in the section "Simplex Algorithm" on page 8574. The ALPHA= option requests 90% confidence limits for the regression parameters, and the option CI=RESAMPLING specifies that the intervals be computed by using the Markov chain marginal bootstrap (MCMB) resampling method of He and Hu (2002). When you specify the CI=RESAMPLING option, the QUANTREG procedure also computes standard errors, t values, and p-values of regression parameters by using the MCMB resampling method. The SEED= option specifies a seed for the resampling method. The COVB option requests covariance matrices for the estimated regression coefficients, and the TEST statement requests tests for the hypothesis that the slope parameter (the coefficient of WDRatio) is 0.

Figure 103.3 displays model information and summary statistics for the variables in the model. The summary statistics include the median and the standardized median absolute deviation (MAD), which are robust measures of univariate location and scale, respectively. For more information about the standardized MAD, see Huber (1981, p. 108).

Figure 103.3 Model Fitting Information and Summary Statistics

The QUANTREG Procedure

Model Information	
Data Set	WORK.TROUT
Dependent Variable	LnDensity
Number of Independent Variables	1
Number of Observations	71
Optimization Algorithm	Simplex
Method for Confidence Limits	Resampling
Quantile Levels	2

Summary Statistics						
Variable	Q1	Median	Q3	Mean	Standard Deviation	MAD
WDRatio	22.0917	29.4083	35.9382	29.1752	9.9859	10.4970
LnDensity	-2.0511	-1.3813	-0.8669	-1.4973	0.7682	0.8214

Figure 103.4 and Figure 103.5 display the parameter estimates, standard errors, 95% confidence limits, t values, and p-values that are computed by the resampling method.

Figure 103.4 Parameter Estimates at QUANTILE=0.5

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	90% Confidence Limits		t Value	Pr > |t|
Intercept	1	-0.9811	0.3952	-1.6400	-0.3222	-2.48	0.0155
WDRatio	1	-0.0136	0.0123	-0.0341	0.0068	-1.11	0.2705

Figure 103.5 Parameter Estimates at QUANTILE=0.9

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	90% Confidence Limits		t Value	Pr > |t|
Intercept	1	0.0576	0.2606	-0.3769	0.4921	0.22	0.8257
WDRatio	1	-0.0215	0.0075	-0.0340	-0.0091	-2.88	0.0053

The 90th percentile of trout density can be predicted from the width-to-depth ratio as follows:

$$\hat{y}_{90} = \exp(0.0576 - 0.0215x)$$

This is the upper dashed curve that is plotted in Figure 103.1. The lower dashed curve for the median can be obtained in a similar fashion.

The covariance matrices for the estimated parameters are shown in Figure 103.6. The resampling method that is used for the confidence intervals is also used to compute these matrices.

Figure 103.6 Covariance Matrices of the Estimated Parameters

The QUANTREG Procedure
Quantile Level = 0.5

Estimated Covariance Matrix for Quantile Level = 0.5		
	Intercept	WDRatio
Intercept	0.156191	-.004653
WDRatio	-.004653	0.000151

The QUANTREG Procedure
Quantile Level = 0.9

Estimated Covariance Matrix for Quantile Level = 0.9		
	Intercept	WDRatio
Intercept	0.067914	-.001877
WDRatio	-.001877	0.000056

The tests requested by the TEST statement are shown in Figure 103.7. Both the Wald test and the likelihood ratio test indicate that the coefficient of width-to-depth ratio is significantly different from 0 at the 90th percentile, but the difference is not significant at the median.

Figure 103.7 Tests of Significance

Test Results					
Quantile Level	Test	Test Statistic	DF	Chi-Square	Pr > ChiSq
0.5	**Wald**	1.2339	1	1.23	0.2666
0.5	**Likelihood Ratio**	1.1467	1	1.15	0.2842
0.9	**Wald**	8.3031	1	8.30	0.0040
0.9	**Likelihood Ratio**	9.0529	1	9.05	0.0026

In many quantile regression problems it is useful to examine how the estimated regression parameters for each covariate change as a function of τ in the interval $(0, 1)$. The following statements use the QUANTREG procedure to request the estimated quantile processes $\hat{\boldsymbol{\beta}}(\tau)$ for the slope and intercept parameters:

```
proc quantreg data=trout alpha=0.1 ci=resampling;
   model LnDensity = WDRatio / quantile=process seed=1268
                               plot=quantplot;
run;
```

The QUANTILE=PROCESS option requests an estimate of the quantile process for each regression parameter. The options ALPHA=0.1 and CI=RESAMPLING specify that 90% confidence bands for the quantile processes be computed by using the resampling method.

Figure 103.8 displays a portion of the objective function table for the quantile process model. The objective function is evaluated at 77 values of τ in the interval $(0, 1)$. The table also provides predicted values of the conditional quantile function $Q(\tau)$ at the mean for WDRatio, which can be used to estimate the conditional density function.

Figure 103.8 Objective Function Values for Quantile Process

Objective Function for Quantile Process			
Label	Quantile Level	Objective Function	Predicted at Mean
t0	0.005634	0.7044	-3.2582
t1	0.020260	2.5331	-3.0331
t2	0.031348	3.7421	-2.9376
t3	0.046131	5.2538	-2.7013
.	.	.	.
.	.	.	.
.	.	.	.
t73	0.945705	4.1433	-0.4361
t74	0.966377	2.5858	-0.4287
t75	0.976060	1.8512	-0.4082
t76	0.994366	0.4356	-0.4082

Figure 103.9 displays a portion of the table of the quantile processes for the estimated parameters and confidence limits.

Figure 103.9 Parameter Estimates for Quantile Process

Parameter Estimates for Quantile Process			
Label	**Quantile Level**	**Intercept**	**WDRatio**
.	.	.	.
.	.	.	.
.	.	.	.
t57	0.765705	-0.42205	-0.01335
lower90	0.765705	-0.91952	-0.02682
upper90	0.765705	0.07541	0.00012
t58	0.786206	-0.32688	-0.01592
lower90	0.786206	-0.80883	-0.02895
upper90	0.786206	0.15507	-0.00289
.	.	.	.
.	.	.	.
.	.	.	.

When ODS Graphics is enabled, the PLOT=QUANTPLOT option in the MODEL statement requests a plot of the estimated quantile processes.

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The left side of Figure 103.10 displays the process for the intercept, and the right side displays the process for the coefficient of WDRatio.

The process plot for WDRatio shows that the slope parameter changes from positive to negative as the quantile increases and that it changes sign with a sharp drop at the 40th percentile. The 90% confidence bands show that the relationship between LnDensity and WDRatio (expressed by the slope) is not significant below the 78th percentile. This situation can also be seen in Figure 103.9, which shows that 0 falls between the lower and upper confidence limits of the slope parameter for quantiles below 0.78. Since the confidence intervals for the extreme quantiles are not stable because of insufficient data, the confidence band is not displayed outside the interval (0.05, 0.95).

Figure 103.10 Quantile Processes for Intercept and Slope

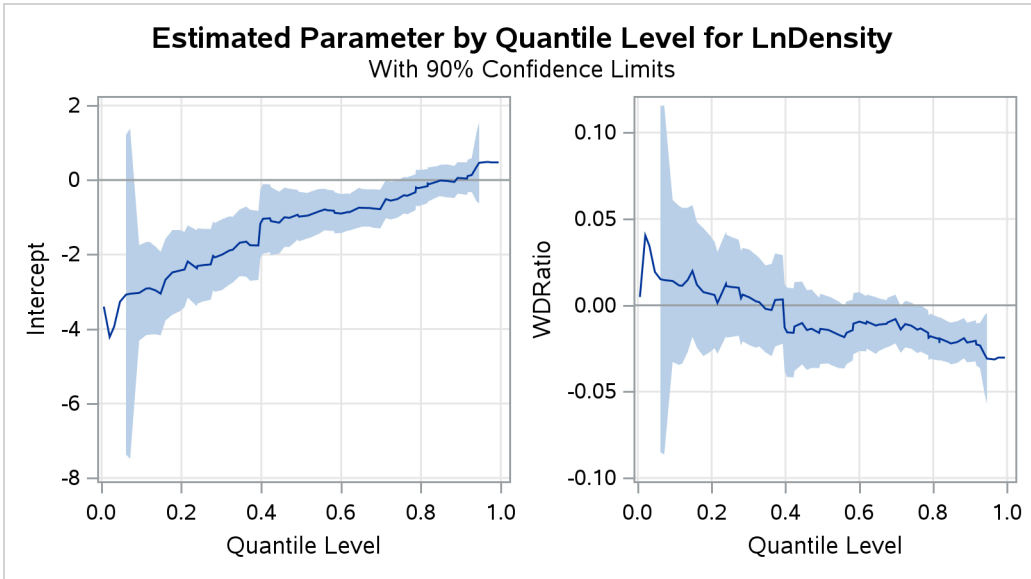

The QUANTILE=FQPR(N=q) option performs parameter estimation for the quantile process model in the equally spaced quantile-level grid $\left\{\tau_0 = \frac{0.5}{q}, \tau_1 = \frac{1.5}{q}, \ldots, \tau_{q-1} = \frac{q-0.5}{q}\right\}$ by using the fast quantile process regression algorithm. The following statements estimate the quantile process model, $Q_{\mathbf{Y}|\mathbf{x}}(\tau) = \mathbf{x}'\boldsymbol{\beta}(\tau)$, in a size-100, equally spaced quantile-level grid:

```
ods output ProcessEst=oProEst;
proc quantreg data=trout ci=none;
   model LnDensity = WDRatio / quantile=fqpr(n=100);
run;
```

Figure 103.11 displays a portion of the table that shows the parameter estimates of the quantile process model.

Figure 103.11 Parameter Estimates for Quantile Process

Parameter Estimates for Quantile Process			
Label	Quantile Level	Intercept	WDRatio
t0	0.005000	-3.39929	0.00483
t1	0.015000	-3.39929	0.00483
t2	0.025000	-4.21355	0.04046
t3	0.035000	-3.94081	0.03439
t4	0.045000	-3.94081	0.03439
t5	.	.	.
t99	0.995000	0.47468	-0.03026

Figure 103.12 displays the information about the quantile-level grid, the average objective-function value, and the average prediction at the mean value of WDRatio.

Figure 103.12 Average Objective Function

The QUANTREG Procedure

Quantile Levels and Average Objective Function	
Number of Quantile Levels	100
Minimum Quantile Level	0.0050
Maximum Quantile Level	0.9950
Average Objective Function	15.0539
Average Predicted Value at Mean	-1.4989

Figure 103.13 displays the average parameter estimates that model the conditional mean of the response variable LnDensity and are comparable to the parameter estimates from its counterpart linear regression model.

Figure 103.13 Average Parameter Estimates

Average Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	-1.3307
WDRatio	1	-0.0058

You can perform observationwise conditional distribution analysis by using the quantile process model. The following statements sort all the observations in ascending order of the width-to-depth ratios (WDRatio), and they use the CONDDIST statement of the QUANTREG procedure to perform the conditional distribution analysis for the second, 15th, 37th, and 63rd observations:

```
proc sort data=trout;
   by WDRatio;
run;

proc quantreg data=trout ci=none;
   model LnDensity = WDRatio / quantile=fqpr(n=100);
   conddist hr obs= 2 15 37 63 plots=all;
run;
```

Figure 103.14 displays the four selected training observations under conditional distribution analysis.

Figure 103.14 Observations for Conditional Distribution Analysis

Observations for Conditional Distribution Analysis			
Label	**Logarithm of Density**	**Density**	**WDRatio**
2	0.15662	1.16956	10.5102
15	-1.35772	0.25725	20.7852
37	-0.58719	0.55589	30.9569
63	-1.52685	0.21722	40.0388

Figure 103.15 shows the conditional distribution estimates for the selected observations. In Figure 103.15, the

regression quantile levels measure the quantile levels (which are equivalent to percentages) of the response values conditional on their respective WDRatio values; the sample quantile levels measure the percentages of the same response values in the pool of all the training response values without adjustment on their respective WDRatio values.

Figure 103.15 Average Objective Function

The QUANTREG Procedure
Conditional Distribution Analysis 1

Conditional Distribution Estimates

				Quantile Level		
Data	Type	Label	Response Value	Regression	Sample	Regression Density
Training	Fit for Obs	2	0.156620	0.9873	0.9930	0.167156
Training	Fit for Obs	15	-1.35772	0.4325	0.5141	0.381070
Training	Fit for Obs	37	-0.58719	0.9350	0.9225	0.355702
Training	Fit for Obs	63	-1.52685	0.5345	0.4155	0.544205

Figure 103.16 and Figure 103.17, respectively, show the cumulative distribution functions (CDFs) and the probability density functions (PDFs) of the response random variables for the four selected observations conditional on their respective WDRatio values. The dots represent the observed response values of the selected observations on the X axis, the relevant regression quantile levels on the Y axis (in Figure 103.16), and the relevant probability density values on the Y axis (in Figure 103.17).

Figure 103.16 Conditional CDFs for the Selected Observations

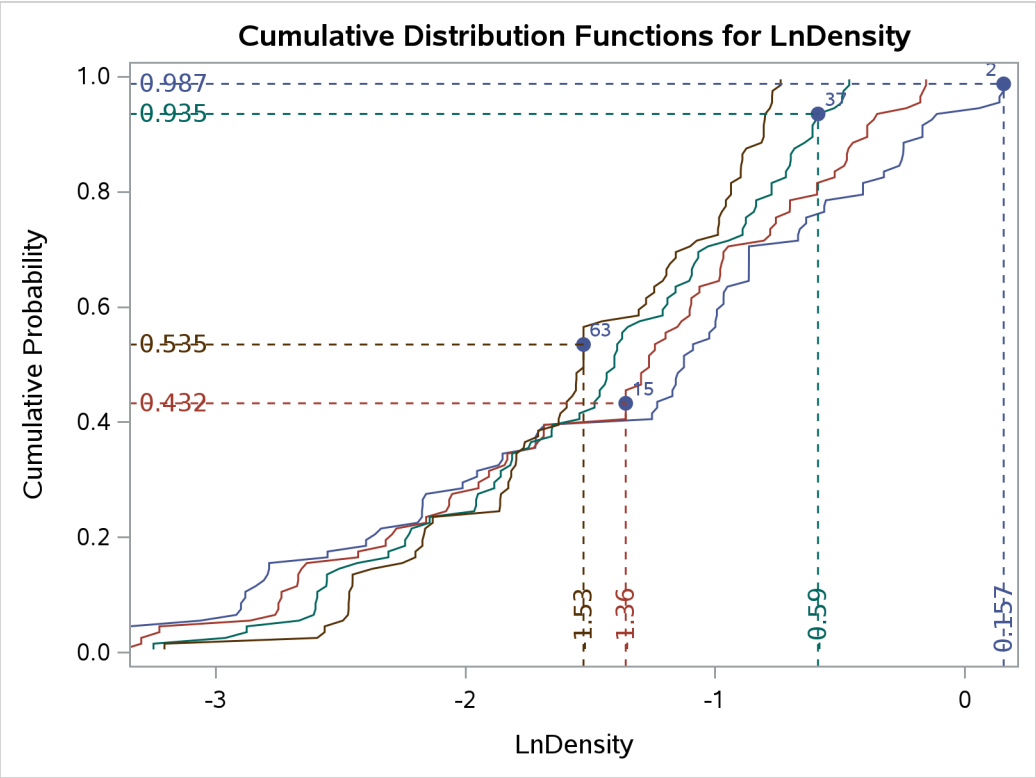

Figure 103.17 Conditional PDFs for the Selected Observations

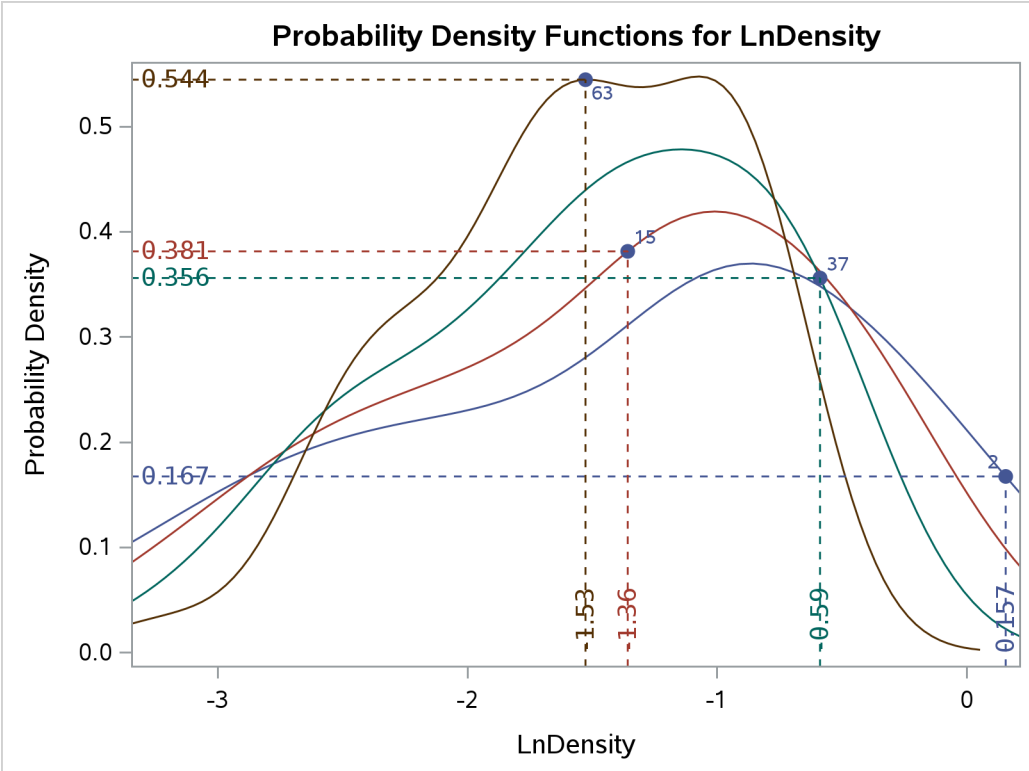

Growth Charts for Body Mass Index

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/qreggs2.sas`.)

Body mass index (BMI) is defined as the ratio of weight (kg) to squared height (m^2) and is a widely used measure for categorizing individuals as overweight or underweight. The percentiles of BMI for specified ages are of particular interest. As age increases, these percentiles provide growth patterns of BMI not only for the majority of the population, but also for underweight or overweight extremes of the population. In addition, the percentiles of BMI for a specified age provide a reference for individuals at that age with respect to the population.

Smooth quantile curves have been widely used for reference charts in medical diagnosis to identify unusual subjects, whose measurements lie in the tails of the reference distribution. This example explains how to use the QUANTREG procedure to create growth charts for BMI.

A SAS data set named bmimen was created by merging and cleaning the 1999–2000 and 2001–2002 survey results for men that is published by the National Center for Health Statistics. This data set contains the variables Weight (kg), Height (m), BMI (kg/m^2), Age (year), and SeQN (respondent sequence number) for 8,250 men (Chen 2005).

The data set that is used in this example is a subset of the original data set of Chen (2005). It contains the two variables BMI and Age with 3,264 observations.

```
data bmimen;
   input BMI Age @@;
   SqrtAge = sqrt(Age);
   InveAge = 1/Age;
   LogBMI  = log(BMI);
   datalines;
18.6  2.0 17.1  2.0 19.0  2.0 16.8  2.0 19.0  2.1  15.5   2.1
16.7  2.1 16.1  2.1 18.0  2.1 17.8  2.1 18.3  2.1  16.9   2.1
15.9  2.1 20.6  2.1 16.7  2.1 15.4  2.1 15.9  2.1  17.7   2.1

   ... more lines ...

29.0 80.0 24.1 80.0 26.6 80.0 24.2 80.0 22.7 80.0  28.4  80.0
26.3 80.0 25.6 80.0 24.8 80.0 28.6 80.0 25.7 80.0  25.8  80.0
22.5 80.0 25.1 80.0 27.0 80.0 27.9 80.0 28.5 80.0  21.7  80.0
33.5 80.0 26.1 80.0 28.4 80.0 22.7 80.0 28.0 80.0  42.7  80.0
;
```

The logarithm of BMI is used as the response. (Although this does not improve the quantile regression fit, it helps with statistical inference.) A preliminary median regression is fitted with a parametric model, which involves six powers of Age.

The following statements invoke the QUANTREG procedure:

```
proc quantreg data=bmimen algorithm=interior(tolerance=1e-5) ci=resampling;
   model logbmi = inveage sqrtage age sqrtage*age
                  age*age age*age*age
                  / diagnostics cutoff=4.5 quantile=.5 seed=1268;
   id age bmi;
   test_age_cubic: test age*age*age / wald lr rankscore(tau);
run;
```

The MODEL statement provides the model, and the option QUANTILE=0.5 requests median regression. The ALGORITHM= option requests that the interior point algorithm be used to compute $\hat{\boldsymbol{\beta}}(\frac{1}{2})$. For more information about this algorithm, see the section "Interior Point Algorithm" on page 8575.

Figure 103.18 displays the estimated parameters, standard errors, 95% confidence intervals, t values, and p-values that are computed by the resampling method, which is requested by the CI= option. All of the parameters are considered significant because the p-values are smaller than 0.001.

Figure 103.18 Parameter Estimates with Median Regression: Men

The QUANTREG Procedure

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	7.8909	0.8200	6.2832	9.4987	9.62	<.0001
InveAge	1	-1.8354	0.4367	-2.6915	-0.9792	-4.20	<.0001
SqrtAge	1	-5.1247	0.7166	-6.5298	-3.7196	-7.15	<.0001
Age	1	1.9759	0.2549	1.4761	2.4756	7.75	<.0001
SqrtAge*Age	1	-0.3347	0.0427	-0.4183	-0.2511	-7.85	<.0001
Age*Age	1	0.0227	0.0029	0.0170	0.0285	7.73	<.0001
Age*Age*Age	1	-0.0000	0.0000	-0.0001	-0.0000	-7.36	<.0001

The TEST statement requests Wald, likelihood ratio, and rank tests for the significance of the cubic term in Age. The test results, shown in Figure 103.19, indicate that this term is significant. Higher-order terms are not significant.

Figure 103.19 Test of Significance for Cubic Term

Test test_age_cubic Results				
Test	Test Statistic	DF	Chi-Square	Pr > ChiSq
Wald	54.1317	1	54.13	<.0001
Likelihood Ratio	56.9473	1	56.95	<.0001
Rank_Tau	42.5731	1	42.57	<.0001

Median regression and, more generally, quantile regression are robust to extremes of the response variable. The DIAGNOSTICS option in the MODEL statement requests a diagnostic table of outliers, shown in Figure 103.20, which uses a cutoff value that is specified in the CUTOFF= option. The variables that are specified in the ID statement are included in the table.

With CUTOFF=4.5, 14 men are identified as outliers. All of these men have large positive standardized residuals, which indicates that they are overweight for their age. The cutoff value 4.5 is ad hoc. It corresponds to a probability less than 0.5E–5 if normality is assumed, but the standardized residuals for median regression usually do not meet this assumption.

In order to construct the chart shown in Figure 103.2, the same model that is used for median regression is used for other quantiles. The QUANTREG procedure can compute fitted values for multiple quantiles.

Figure 103.20 Diagnostics with Median Regression

Diagnostics				
Obs	Age	BMI	Standardized Residual	Outlier
1337	8.900000	36.500000	5.3575	*
1376	9.200000	39.600000	5.8723	*
1428	9.400000	36.900000	5.3036	*
1505	9.900000	35.500000	4.8862	*
1764	14.900000	46.800000	5.6403	*
1838	16.200000	50.400000	5.9138	*
1845	16.300000	42.600000	4.6683	*
1870	16.700000	42.600000	4.5930	*
1957	18.100000	49.900000	5.5053	*
2002	18.700000	52.700000	5.8106	*
2016	18.900000	48.400000	5.1603	*
2264	32.000000	55.600000	5.3085	*
2291	35.000000	60.900000	5.9406	*
2732	66.000000	14.900000	-4.7849	*

The following statements request fitted values for 10 quantile levels that range from 0.03 to 0.97:

```
proc quantreg data=bmimen algorithm=interior(tolerance=1e-5) ci=none;
   model logbmi = inveage sqrtage age sqrtage*age
                  age*age age*age*age
                  / quantile=0.03,0.05,0.1,0.25,0.5,0.75,
                            0.85,0.90,0.95,0.97;
   output out=outp pred=p/columnwise;
run;

data outbmi;
   set outp;
   pbmi = exp(p);
run;

proc sgplot data=outbmi;
   title 'BMI Percentiles for Men: 2-80 Years Old';
   yaxis label='BMI (kg/m**2)' min=10 max=45 values=(10 15 20 25 30 35 40 45);
   xaxis label='Age (Years)' min=2 max=80 values=(2 10 20 30 40 50 60 70 80);

   scatter x=age y=bmi /markerattrs=(size=1);
   series  x=age y=pbmi/group=QUANTILE;
run;
```

The fitted values are stored in the OUTPUT data set outp. The COLUMNWISE option arranges these fitted values for all quantiles in the single variable p by groups of the quantiles. After the exponential transformation, both the fitted BMI values and the original BMI values are plotted against age to create the display shown in Figure 103.2.

The fitted quantile curves reveal important information. During the quick growth period (ages 2 to 20), the dispersion of BMI increases dramatically. It becomes stable during middle age, and then it contracts after age 60. This pattern suggests that effective population weight control should start in childhood.

Compared to the 97th percentile in reference growth charts that were published by the Centers for Disease Control and Prevention (CDC) in 2000 (Kuczmarski, Ogden, and Guo 2002), the 97th percentile for 10-year-old boys in Figure 103.2 is 6.4 BMI units higher (an increase of 27%). This can be interpreted as a warning of overweight or obesity. See Chen (2005) for a detailed analysis.

Syntax: QUANTREG Procedure

The following statements are available in the QUANTREG procedure:

```
PROC QUANTREG <options> ;
    BY variables ;
    CLASS variables </ option> ;
    CONDDIST <options> ;
    EFFECT name = effect-type (variables </ options>) ;
    ESTIMATE <'label'> estimate-specification </ options> ;
    ID variables ;
    MODEL response = <effects> </ options> ;
    OUTPUT <OUT= SAS-data-set> <options> ;
    PERFORMANCE <options> ;
    TEST effects </ options> ;
    WEIGHT variable ;
```

The PROC QUANTREG statement invokes the QUANTREG procedure. The CLASS statement specifies which explanatory variables are treated as categorical. The ID statement names variables to identify observations in the outlier diagnostics tables. The MODEL statement is required and specifies the variables used in the regression. Main effects and interaction terms can be specified in the MODEL statement, as in the GLM procedure (Chapter 53, "The GLM Procedure.") The OUTPUT statement creates an output data set that contains predicted values, residuals, and estimated standard errors. The PERFORMANCE statement tunes the performance of PROC QUANTREG by using single or multiple processors available in the hardware. The TEST statement requests linear tests for the model parameters. The WEIGHT statement identifies a variable in the input data set whose values are used to weight the observations. Multiple OUTPUT and TEST statements are allowed in one invocation of PROC QUANTREG.

The EFFECT and ESTIMATE statements are also available in other procedures. Summary descriptions of functionality and syntax for these statements are provided in this chapter, and you can find full documentation about them in Chapter 20, "Shared Concepts and Topics."

PROC QUANTREG Statement

PROC QUANTREG < *options* > **;**

The PROC QUANTREG statement invokes the QUANTREG procedure. Table 103.1 summarizes the *options* available in the PROC QUANTREG statement.

Table 103.1 PROC QUANTREG Statement Options

Option	Description
ALGORITHM=	Specifies an algorithm to estimate the regression parameters
ALPHA=	Specifies the level of significance
CI=	Specifies a method to compute confidence intervals
DATA=	Specifies the input SAS data set
INEST=	Specifies an input SAS data set that contains initial estimates
NAMELEN=	Specifies the length of effect names
ORDER=	Specifies the order in which to sort classification variables
OUTEST=	Specifies an output SAS data set containing the parameter estimates
PLOT	Specifies options that control details of the plots

You can specify the following *options* in the PROC QUANTREG statement.

ALGORITHM=*algorithm* **< (** *suboptions* **) >**
specifies an algorithm for estimating the regression parameters.

You can specify one of the following four *algorithms*:

SIMPLEX < *suboption* >
uses the simplex algorithm to estimate the regression parameters. You can specify the following *suboption*:

MAXSTATIONARY=*m* requests that the algorithm terminate if the objective function has not improved for *m* consecutive iterations. By default, *m* = 1000.

INTERIOR < (*suboptions* **) >**
uses the interior point algorithm to estimate the regression parameters. You can specify the following *suboptions*:

KAPPA=*value* specifies the step-length parameter for the interior point algorithm. The *value* should be between 0 and 1. The larger the *value*, the faster the algorithm. However, numeric instability can occur as the *value* approaches 1. By default, KAPPA=0.99995. For more information, see the section "Interior Point Algorithm" on page 8575.

MAXIT=*m* sets the maximum number of iterations for the interior point algorithm. By default, MAXIT=1000.

TOLERANCE=*value* specifies the tolerance for the convergence criterion of the interior point algorithm. By default, TOLERANCE=1E–8. The QUANTREG

procedure uses the duality gap as the convergence criterion. For more information, see the section "Interior Point Algorithm" on page 8575.

You can also use the PERFORMANCE statement to enable parallel computing when multiple processors are available in the hardware.

IPM < (*suboptions*) >
uses the efficient interior point algorithm to estimate the regression parameters. You can specify the following *suboptions*:

MAXIT=*m* sets the maximum number of iterations for the efficient interior point algorithm. By default, MAXIT=1000.

TOLERANCE=*value* specifies the tolerance for the convergence criterion of the efficient interior point algorithm. The QUANTREG procedure uses the complementarity value as the convergence criterion. By default, TOLERANCE=1E–8.

SMOOTH < *suboption* >
uses the smoothing algorithm to estimate the regression parameters. You can specify the following *suboption*:

RRATIO=*value* specifies the reduction ratio for the smoothing algorithm. This ratio is used to reduce the threshold of the smoothing algorithm. The *value* should be between 0 and 1. In theory, the smaller the *value*, the faster the smoothing algorithm. When the QUANTILE=FQPR option in the MODEL statement is omitted, the default RRATIO= value is dependent on the training data. For more information, see the section "Smoothing Algorithm" on page 8578. When the QUANTILE=FQPR option in the MODEL statement is specified, then by default, RRATIO=0.25. For more information about the FQPR smoothing algorithm, see the section "Fast Quantile Process Regression" on page 8580.

The default algorithm depends on the number of observations (n) and the number of covariates (p) in the model estimation. See Table 103.2 for the relevant defaults.

Table 103.2 The Default Estimation Algorithm

	$p \leq 100$	$p > 100$
$n \leq 5000$	SIMPLEX	SMOOTH
$n > 5000$	INTERIOR	SMOOTH

ALPHA=*value*

specifies the level of significance α for $100(1-\alpha)\%$ confidence intervals for regression parameters. The *value* must be between 0 and 1. The default is ALPHA=0.05, which corresponds to a 0.95 confidence interval.

CI=NONE | RANK | SPARSITY< (BF | HS) >< /IID > | RESAMPLING< (NREP=*n***) >**

specifies a method for computing confidence intervals for regression parameters. When you specify CI=SPARSITY or CI=RESAMPLING, the QUANTREG procedure also computes standard errors, *t* values, and *p*-values for regression parameters.

Table 103.3 summarizes these methods.

Table 103.3 Options for Confidence Intervals

Value of CI=	Method	Additional Options
NONE	No confidence intervals computed	
RANK	By inverting rank-score tests	
RESAMPLING	By resampling	NREP
SPARSITY	By estimating sparsity function	HS, BF, and IID

By default, when there are fewer than 5,000 observations, fewer than 20 variables in the data set, and the algorithm is simplex, the QUANTREG procedure computes confidence intervals by using the inverted rank-score test method. Otherwise, the resampling method is used.

By default, confidence intervals are not computed for the quantile process, which is estimated when you specify the QUANTILE=PROCESS option in the MODEL statement. Confidence intervals for the quantile process are computed by using the sparsity or resampling methods when you specify CI=SPARSITY or CI=RESAMPLING, respectively. The rank method for confidence intervals is not available for quantile processes because it is computationally prohibitive.

When you specify the SPARSITY option, you have two suboptions for estimating the sparsity function. If you specify the IID suboption, the sparsity function is estimated by assuming that the errors in the linear model are independent and identically distributed (iid). By default, the sparsity function is estimated by assuming that the conditional quantile function is locally linear. For more information, see the section "Sparsity" on page 8582. For both methods, two bandwidth selection methods are available: You can specify the BF suboption for the Bofinger method or the HS suboption for the Hall-Sheather method. By default, the Hall-Sheather method is used.

When you specify the RESAMPLING option, you can specify the NREP=*n* suboption for the number of repetitions. By default, NREP=200. The value of *n* must be greater than 50.

DATA=*SAS-data-set*

specifies the input SAS data set that contains the training observations to be used by the QUANTREG procedure. By default, the most recently created SAS data set is used.

INEST=*SAS-data-set*

specifies an input SAS data set that contains initial estimates for all the parameters in the model. The interior point algorithm and the smoothing algorithm use these estimates as a start. For a detailed description of the contents of the INEST= data set, see the section "INEST= Data Set" on page 8595.

NAMELEN=*n*

restricts the length of effect names in tables and output data sets to *n* characters, where *n* is a value between 20 and 200. By default, NAMELEN=20.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUTEST=*SAS-data-set*

specifies an output SAS data set to contain the parameter estimates for all quantiles. See the section "OUTEST= Data Set" on page 8596 for a detailed description of the contents of the OUTEST= data set.

PLOT | PLOTS< (*global-plot-options*) **>** **<** **=***plot-request* **>**
PLOT | PLOTS< (*global-plot-options*) **>** **<** **=(***plot-request* **<** **...** *plot-request* **>** **)** **>**

specifies options that control details of the plots. These plots fall into two categories: diagnostic plots and fit plots. You can also use the PLOT= option in the MODEL statement to request the quantile process plot for any effects that are specified in the model. If you do not specify the PLOTS= option, PROC QUANTREG produces the quantile fit plot by default when a single continuous variable is specified in the model.

When you specify only one *plot-request*, you can omit the parentheses around the plot request.

Here are some examples:

```
plots=ddplot
plots=(ddplot rdplot)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc quantreg plots=fitplot;
   model y=x1;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *global-plot-options*, which apply to all plots that PROC QUANTREG generates:

MAXPOINTS=NONE | *number*
suppresses plots that have elements that require processing more than *number* points. The default is MAXPOINTS=5000. This cutoff is ignored if you specify MAXPOINTS=NONE.

ONLY
suppresses the default quantile fit plot. Only plots specifically requested are displayed.

You can specify the following *plot-requests*:

ALL
creates all appropriate plots.

DDPLOT< (LABEL=ALL | LEVERAGE | NONE | OUTLIER) >
creates a plot of robust distance against Mahalanobis distance. For more information about robust distance, see the section "Leverage Point and Outlier Detection" on page 8594. The LABEL= option specifies how the points on this plot are to be labeled, as summarized by Table 103.4.

Table 103.4 Options for Label

Value of LABEL=	Label Method
ALL	Label all points
LEVERAGE	Label leverage points
NONE	No labels
OUTLIERS	Label outliers

By default, the QUANTREG procedure labels both outliers and leverage points.

If you specify ID variables in the ID statement, the values of the first ID variable are used as labels; otherwise, observation numbers are used as labels.

FITPLOT< (NOLIMITS | SHOWLIMITS | NODATA) >

creates a plot of fitted conditional quantiles against the single continuous variable that is specified in the model. This plot is produced only when the response is modeled as a function of a single continuous variable. Multiple lines or curves are drawn on this plot if you specify several quantiles with the QUANTILE= option in the MODEL statement. By default, confidence limits are added to the plot when a single quantile is requested, and the confidence limits are not shown on the plot when multiple quantiles are requested. The NOLIMITS option suppresses the display of the confidence limits. The SHOWLIMITS option adds the confidence limits when multiple quantiles are requested. The NODATA option suppresses the display of the observed data, which are superimposed on the plot by default.

HISTOGRAM

creates a histogram (based on the quantile regression estimates) for the standardized residuals. The histogram is superimposed with a normal density curve and a kernel density curve.

NONE

suppresses all plots.

QQPLOT

creates the normal quantile-quantile plot (based on the quantile regression estimates) for the standardized residuals.

RDPLOT< (LABEL=ALL | LEVERAGE | NONE | OUTLIER) >

creates the plot of standardized residual against robust distance. For more information about robust distance, see the section "Leverage Point and Outlier Detection" on page 8594.

The LABEL= option specifies a label method for points on this plot. These label methods are described in Table 103.4.

By default, the QUANTREG procedure labels both outliers and leverage points.

If you specify ID variables in the ID statement, the values of the first ID variable are used as labels; otherwise, observation numbers are used as labels.

PP

requests preprocessing to speed up the interior point algorithm or the smoothing algorithm. The preprocessing uses a subsampling algorithm (which assumes that the data set is evenly distributed) to iteratively reduce the original problem to a smaller one. Preprocessing should be used only for very large data sets, such as data sets with more than 100,000 observations. For more information, see Portnoy and Koenker (1997).

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC QUANTREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the QUANTREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / **TRUNCATE** > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC QUANTREG statement.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

CONDDIST Statement

< *label***: > CONDDIST** *options* **;**

The CONDDIST statement estimates and displays the conditional and marginal probability functions for the response random variable. These functions include the cumulative distribution functions (CDFs) and the probability density functions (PDFs). For more information about the probability functions, see the section "Estimating Probability Functions by Using the CONDDIST Statement" on page 8587.

You can specify multiple CONDDIST statements. You can use the optional *label*, which must be a valid SAS name, to identify the output group for its corresponding CONDDIST statement.

Table 103.5 summarizes the *options* available in the CONDDIST statement.

Table 103.5 CONDDIST Statement Options

Option	Description
HIDERAW	Hides the observed marginal distribution of the training response for the analysis
MCDF	Performs the marginal distributions analysis
MWU	Performs the Mann-Whitney U test against the observed marginal CDF sample of the training response variable
OBS=	Specifies a subset of the training observations for the analysis by using the indices of these observations in the DATA= data set of the PROC QUANTREG statement
PDF=KDE	Specifies the kernel density estimator for estimating the probability density functions
PLOTS=	Specifies options for graphically displaying the probability functions
SHOWAVG	Requests the distribution of the training response that is conditional on the average explanatory covariates for the analysis
TESTDATA=	Specifies an input SAS data set that contains test observations for estimating the probability functions

You can specify the following *options*:

HIDERAW
HR

hides the observed marginal CDF sample of the training response for the analysis. This CDF sample is estimated from the sample of all the response values y_i in the training data without using the quantile regression model. The CONDDIST statement assigns the type "Observed" and the label "TrainObs" to this distribution.

MCDF(*mcdf-option***)**

performs the marginal distributions analysis by using the weighted bootstrap resampling method. For more information, see the section "Marginal Distribution Analysis Using the Bootstrap Resampling Method" on page 8591. You can specify the following *mcdf-options*:

QUANTILERANGE=BOTH | TRAIN | NONE
QRNG=BOTH | TRAIN | NONE

specifies the type of limits for the quantile predictions. You can specify the following limits:

BOTH specifies the range of the observed response values for combined training and testing data sets.

TRAIN specifies the range of the observed response values for only the training data set.

NONE specifies no forced range.

If a quantile prediction is smaller than the lower limit of the specified response range, the quantile prediction is reset to the lower-limit value for postprocessing. If a quantile prediction is larger than the upper limit of the specified response range, the quantile prediction is reset to the upper-limit value for postprocessing. By default, QRNG=BOTH. For more information, see Step 5b in the section "Marginal Distribution Analysis Using the Bootstrap Resampling Method" on page 8591.

NREP=*n*

specifies the number of repetitions for the bootstrap resampling process.

The MCDF option repeatedly fits a quantile process regression model for each reweighted bootstrap sample of the training observations in the DATA= data set in the PROC QUANTREG statement. If you also specify the TESTDATA= testing data set option in the CONDDIST statement, then the MCDF option additionally generates a reweighted bootstrap sample for the testing observations in each repetition; estimates counterfactual distribution for the testing sample by using the relevant training-sample quantile process model; and outputs the marginal distribution comparisons table to compare the averaged-across-all-samples training, testing, and counterfactual testing marginal distributions. You can output all the fitted training-sample quantile process regression models by using the following ODS OUTPUT statement:

```
ods output BootProcessEst=BPE;
```

MWU(*mwu-option***)**
WILCOXON(*mwu-option***)**
RANKSUM(*mwu-option***)**

performs the Mann-Whitney U test (also called the Wilcoxon rank-sum test and denoted as the MWU test) for each CDF sample of all specified observations against the observed marginal CDF sample of the training response variable for the DATA= data set in the PROC QUANTREG statement.

You can specify the following *mwu-option*:

SAMPLESIZE=NOBS | NQ
SAMPSIZE=NOBS | NQ

specifies the sizes for the relevant CDF samples.

You can specify one of the following sample-size values:

NOBS requests that the size of the CDF samples, except the observed marginal CDF sample for the TESTDATA= data set in the CONDDIST statement (labeled as "TestObs"), be equal to the number of training observations in the DATA= data set in the PROC

QUANTREG statement. The SAMPSIZE=NOBS option requests that the size of the TestObs CDF sample be equal to the number of testing observations in the TESTDATA= data set in the CONDDIST statement. The SAMPSIZE=NOBS option is appropriate for the MWU tests when the size of the quantile-level grid is larger than the number of training observations in the DATA= data set in the PROC QUANTREG statement.

NQ requests that the size of the CDF samples be equal to the size of the quantile-level grid, so that you can indirectly specify the size of the CDF samples by using the QUANTLEV=FQPR(N=) option in the MODEL statement. The SAMPSIZE=NQ option is appropriate for the MWU tests when the size of the quantile-level grid is smaller than both the number of training observations in the DATA= data set in the PROC QUANTREG statement and the number of testing observations in the TESTDATA= data set in the CONDDIST statement.

By default, SAMPSIZE=NQ. The TESTDATA(MWU(SAMPSIZE=)) option overrides this MWU(SAMPSIZE=) option. For more information about the MWU tests and the CDF samples, see the section "Mann-Whitney U Test" on page 8589.

OBS=*number-list*

specifies the observation indices to use for the conditional distribution analysis. Each observation index identifies a training observation in the DATA= data set that you specify in the PROC QUANTREG statement. The CONDDIST statement types the distributions of these observations as "Fit for Obs" and labels the distributions by using the observation ID values (if available) or the observation indices.

PDF=KDE(*kde-options***)**

specifies the kernel density estimator for estimating the probability density functions. For more information about the kernel density estimates, see the section "Probability Density Functions" on page 8593.

You can specify the following *kde-options*:

C=*value*

specifies the standardized bandwidth parameter c for the kernel density estimator.

You can specify one of the following *values*:

number specifies a positive number.

MISE minimizes the approximate mean integrated square error (MISE).

SJPI computes the bandwidth parameter by using the Sheather-Jones plug-in method.

By default, C=MISE.

K=NORMAL | TRIANGULAR | QUADRATIC

specifies the type of kernel function to use for the kernel density estimator. NORMAL species the normal kernel. TRIANGULAR species the triangular kernel. QUADRATIC species the quadratic kernel. By default, K=NORMAL.

LOWER=*value*

L=*value*

specifies the lower bound value for the kernel density curves that suppresses the lower tails of the PDF estimates. If *value* is smaller than the minimum quantile-grid value of all the PDF estimates, the CONDDIST statement ignores this suboption and outputs the minimum quantile-grid value as the lower bound value in the density estimation table.

LUADJUST=TRIM | SCALE | REFLECT

LUA=TRIM | SCALE | REFLECT

specifies the adjustment type for kernel density estimation when you specify the LOWER=*value* or UPPER=*value* option (or both).

Let $\{\hat{f}_1, \ldots, \hat{f}_l\}$ denote the lower tail of the PDF estimates, $\{\hat{f}_{l+1}, \ldots, \hat{f}_{l+m}\}$ denote the remaining PDF estimates, and $\{\hat{f}_{l+m+1}, \ldots, \hat{f}_{l+m+u}\}$ denote the upper tail of the PDF estimates. For simplicity of notation, assume that $\hat{f}_j = 0$ for all $j \leq 0$ and $j > (l + m + u)$. You can specify one of the following types:

TRIM trims the tails of the PDF estimates without adjusting the remaining PDF estimates, so that $\hat{f}_j = 0$ for all $j \leq l$ and $j > (l + m)$.

SCALE suppresses the tails of the PDF estimates and scales the remaining PDF estimates by P/P_r, where $P = \sum_{j=1}^{l+m+u} \hat{f}_j$ is the sum of the PDF estimates and $P_r = \sum_{j=l+1}^{l+m} \hat{f}_j$ is the sum of the remaining PDF estimates.

REFLECT suppresses the tails of the PDF estimates and adjusts the remaining PDF estimates by using the reflections of the suppressed tails. The adjusted $\hat{f}_{l+j}$ equals

$$\hat{f}_{l+j} + \sum_{k=1}^{K} \left(\hat{f}_{l+(1-k)m+1-j} + \hat{f}_{l+(1+k)m+1-j} \right) \text{ for } j = 1, \ldots, m$$

where K is the smallest integer that satisfies $Km \geq l$ and $Km \geq u$.

By default, LUA=TRIM.

UPPER=*value*

U=*value*

specifies the upper bound value for the kernel density curves that suppresses the upper tails of the PDF estimates. If *value* is larger than the maximum quantile-grid value of all the PDF estimates, the CONDDIST statement ignores this suboption and outputs the maximum quantile-grid value as the upper bound value in the density estimation table.

When you specify the PLOT= option to create the CDF plot and the PDF plot, the LOWER= and UPPER= suboptions of the PDF=KDE option also set limits for the horizontal range of the plots.

PLOT | PLOTS< *global-plot-options* >< =*plot-request* >

PLOT | PLOTS< *global-plot-options* >< =(*plot-request* < **...** *plot-request* >) >

specifies graphical options for displaying the probability functions.

You can specify the following *global-plot-options*, which apply to all plots that the CONDDIST statement generates:

HIDEDROPLINES

HDL

suppresses the drop lines for the responses.

HIDEDROPNUMBERS

HDN

suppresses the drop numbers for the responses.

HIDEOBSDOTS

HOD

suppresses the response dots.

HIDEOBSLABELS

HOL

suppresses the response labels.

SHOWGRIDS

SG

displays the grid lines.

You can specify the following *plot-requests*:

ALL

creates all appropriate plots.

CDFPLOT< (*plot-options*) >

plots the cumulative distribution functions (CDFs).

You can use any of the *global-plot-options* for the PLOTS option as the *plot-options* for the CDFPLOT option.

MCDFPLOT< (*plot-options*) >

plots the bootstrap-averaged marginal CDF (MCDF) samples and their confidence bands. You must also specify the MCDF option in the CONDDIST statement to request the MCDF plot.

You can specify the following *plot-options*:

HIDEFIT

HF

suppresses the counterfactual marginal CDF sample of the fitted test response and its confidence limits.

HIDERAWTEST

HRTST

suppresses the observed marginal CDF sample of the test response and its confidence limits.

HIDERAWTRAIN

HRTRN

suppresses the observed marginal CDF sample of the training response and its confidence limits.

NOLIMITS

HCL

suppresses all the confidence limits.

You can use the SHOWGRIDS *global-plot-option* as the *plot-option* to display the grid lines.

PDFPLOT< (*plot-options*) >

plots the probability density functions (PDFs).

You can use any of the *global-plot-options* for the PLOTS option as the *plot-options* for the PDFPLOT option.

PPPLOT< (*plot-options*) >

creates the scatter plot of the regression quantile levels versus the sample quantile levels for the relevant response values. This plot is referred to as the probability-probability plot or PP plot in short.

You can use any of the *global-plot-options* of the PLOTS option, except the HOD suboption, as the *plot-options* for the PPPLOT option.

When you specify the PLOTS option, the following options in the CONDDIST statement control the visualization of their relevant probability functions:

- the HIDERAW and SHOWAVG options
- the HIDEFIT, HIDERAW, SHOWAVG, and SHOWOBS suboptions of the TESTDATA option

SHOWAVG

SA

requests the conditional distribution of the training response at average, $\mathbf{Y}|\bar{\mathbf{x}}$, for the analysis, where $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$ is the average explanatory covariates vector for all the training observations. The CONDDIST statement assigns the type “Fit at Average” and the label “TrainAvg” to this distribution.

TESTDATA(*options*)=*SAS-data-set*

specifies the test data set for the conditional distribution analysis. The TESTDATA= data set must contain all the explanatory variables that you specify in the MODEL statement.

You can specify the following *options*:

HIDEFIT
HF

hides the counterfactual marginal CDF sample of the fitted test response for the analysis. This marginal CDF sample for **Y** integrates out the quantile regression model by pooling together all the quantile process predictions of all the test observations in the TESTDATA= data set. The CONDDIST statement assigns the type "Fit and Pooled" and the label "TestFit" to this distribution.

HIDERAW
HR

hides the observed marginal CDF sample of the test response for the analysis. This CDF sample is estimated from the sample of all the response values y_i in the TESTDATA= data set without using the quantile regression model. The CONDDIST statement assigns the type "Observed" and the label "TestObs" to this distribution.

MWU(*mwu-option***)**
WILCOXON(*mwu-option***)**
RANKSUM(*mwu-option***)**

performs the Mann-Whitney U test for each CDF sample of all specified observations against the observed marginal CDF sample of the testing response variable for the TESTDATA= data set in the CONDDIST statement.

You can specify the following *mwu-option*:

SAMPLESIZE=NOBS | NQ
SAMPSIZE=NOBS | NQ

specifies the sizes of the relevant CDF samples.

You can specify one of the following sample-size values:

NOBS requests that the size of the CDF samples, except the observed marginal CDF sample for the TESTDATA= data set in the CONDDIST statement (labeled as "TestObs"), be equal to the number of training observations in the DATA= data set in the PROC QUANTREG statement. The SAMPSIZE=NOBS option requests that the size of the TestObs CDF sample be equal to the number of testing observations in the TESTDATA= data set in the CONDDIST statement. The SAMPSIZE=NOBS option is appropriate for the MWU tests when the size of the quantile-level grid is larger than the number of training observations in the DATA= data set in the PROC QUANTREG statement.

NQ requests that the size of the CDF samples be equal to the size of the quantile-level grid, so that you can indirectly specify the size of the CDF samples by using the QUANTLEV=FQPR(N=) option in the MODEL statement. The SAMPSIZE=NQ option is appropriate for the MWU tests when the size of the quantile-level grid is smaller than both the number of training observations in the DATA= data set in the PROC QUANTREG statement and the number of testing observations in the TESTDATA= data set in the CONDDIST statement.

By default, SAMPSIZE=NQ. This TESTDATA(MWU(SAMPSIZE=)) option overrides the MWU(SAMPSIZE=) option in the CONDDIST statement. For more information about the MWU tests and the CDF samples, see the section "Mann-Whitney U Test" on page 8589.

SHOWAVG

SA

requests the conditional CDF sample of the test response at average, $\mathbf{Y}|\bar{\mathbf{x}}$, for the analysis, where $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ is the average explanatory covariates vector for the TESTDATA= data set. The CONDDIST statement assigns the type "Fit at Average" and the label "TestAvg" to this CDF sample.

SHOWOBS

SO

requests the conditional CDF samples of all the test observations in the analysis. The CONDDIST statement assigns the type "Fit for Obs" to the CDF samples of these observations and labels these CDF samples by using the observation ID values (if available) or the observation indices. By default, the TESTDATA= option ignores these CDF samples.

By default, if the TESTDATA= data set contains fewer than 16 observations, the CONDDIST statement ignores the observed marginal CDF sample of $\mathbf{Y}$. Otherwise, the CONDDIST statement estimates the observed marginal CDF sample of $\mathbf{Y}$.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 103.6 summarizes the *options* available in the EFFECT statement.

Table 103.6 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, ... <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 103.7 summarizes the *options* available in the ESTIMATE statement.

Table 103.7 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the $\mathbf{L}$ matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement"

on page 475 in Chapter 20, "Shared Concepts and Topics."

ID Statement

ID *variables* **;**

When the diagnostics table is requested by the DIAGNOSTICS option in the MODEL statement, the variables listed in the ID statement are displayed in addition to the observation number. These values are useful for identifying observations. If the ID statement is omitted, only the observation number is displayed.

MODEL Statement

< *label*: > **MODEL** *response* **=** < *effects* > < / *options* > **;**

You can specify main effects and interaction terms in the MODEL statement, as you can in the GLM procedure (Chapter 53, "The GLM Procedure.") Classification variables in the MODEL statement must also be specified in the CLASS statement.

The optional *label*, which must be a valid SAS name, is used to label output from the matching MODEL statement.

Options

Table 103.8 summarizes the *options* available in the MODEL statement.

Table 103.8 MODEL Statement Options

Option	Description
CORRB	Produces the estimated correlation matrix
COVB	Produces the estimated covariance matrix
CUTOFF=	Specifies the multiplier of the cutoff value for outlier detection
DIAGNOSTICS	Requests the outlier diagnostics
ITPRINT	Displays the iteration history
LEVERAGE	Requests an analysis of leverage points
NODIAG	Suppresses the computation for outlier diagnostics
NOINT	Specifies no-intercept regression
NOSUMMARY	Suppresses the computation for summary statistics
PLOT=	Requests plots
QUANTILE=	Specifies the quantile levels
SCALE=	Specifies the scale value used to compute the standardized residuals
SEED=	Specifies the seed for the random number generator
SINGULAR=	Specifies the tolerance for testing singularity

You can specify the following *options* for the model fit.

CORRB

produces the estimated correlation matrix of the parameter estimates. When the resampling method is used to compute the confidence intervals, the QUANTREG procedure computes the bootstrap correlation. When the sparsity method is used to compute the confidence intervals, PROC QUANTREG bases its computation of the asymptotic correlation on an estimator of the sparsity function. The rank method for confidence intervals does not provide a correlation estimate.

COVB

produces the estimated covariance matrix of the parameter estimates. When the resampling method is used to compute the confidence intervals, the QUANTREG procedure computes the bootstrap covariance. When the sparsity method is used to compute the confidence intervals, PROC QUANTREG bases its computation of the asymptotic covariance on an estimator of the sparsity function. The rank method for confidence intervals does not provide a covariance estimate.

CUTOFF=*value*

specifies the multiplier of the cutoff value for outlier detection. By default, CUTOFF=3.

DIAGNOSTICS< (ALL) >

requests the outlier diagnostics. By default, only observations that are identified as outliers or leverage points are displayed. To request that all observations be displayed, specify the ALL option.

ITPRINT

displays the iteration history of the interior point algorithm or the smoothing algorithm.

LEVERAGE< (CUTOFF=*value* **| CUTOFFALPHA=***value* **| H=***n***) >**

requests an analysis of leverage points for the continuous covariates. The results are added to the diagnostics table, which you can request with the DIAGNOSTICS option in the MODEL statement. You can specify the cutoff value for leverage-point detection with the CUTOFF= option. The default cutoff value is $\sqrt{\chi^2_{p;1-\alpha}}$, where α can be specified with the CUTOFFALPHA= option. By default, $\alpha = 0.025$. You can use the H= option to specify the number of points to be minimized for the MCD algorithm used for the leverage-point analysis. By default, H=$[(3n + p + 1)/4]$, where n is the number of observations and p is the number of independent variables. The LEVERAGE option is ignored if the model includes classification variables as covariates.

NODIAG

suppresses the computation for outlier diagnostics. If you specify the NODIAG option, the diagnostics summary table is not provided.

NOINT

specifies no intercept regression.

NOSUMMARY

suppresses the computation of summary statistics. If you specify the NOSUMMARY option, the summary statistics table is not provided.

PLOT=*plot-option*

PLOTS=(*plot-option***)**

You can use the PLOTS= option in the MODEL statement together with ODS Graphics to request the quantile process plot in addition to all that plots that you request in the PLOT= option in the PROC QUANTREG statement.

You can specify the following *plot-option* in the MODEL statement:

QUANTPLOT< (EFFECTS) < / < NOLIMITS > < EXTENDCI > < UNPACK > < OLS > > >
plots the regression quantile process. The estimated coefficient of each specified covariate effect is plotted as a function of the quantile. If you do not specify a covariate effect, quantile processes are plotted for all covariate effects in the MODEL statement. You can use the NOLIMITS option to suppress confidence bands for the quantile processes. By default, confidence bands are plotted, and process plots are displayed in panels, each of which can hold up to four plots. By default, the confidence limits are plotted for quantiles in the range between 0.05 and 0.95. You can use the EXTENDCI option to plot the confidence limits even for quantiles outside this range. You can use the UNPACK option to create individual process plots. For an individual process plot, you can superimpose the ordinary least squares estimate by specifying the OLS option.

ODS Graphics must be enabled before you request plots.

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

QUANTILE=*number-list* | **PROCESS** | **FQPR< (***suboption***) >**

QUANTLEV=*number-list* | **PROCESS** | **FQPR< (***suboption***) >**
specifies the quantile levels for the quantile regression. A valid quantile level must be a number in the range of (0,1). You can specify the following values for the QUANTILE= option:

number-list
computes quantile regression for quantile levels that are specified in the *number-list*.

PROCESS
computes the entire quantile process regression. If you specify QUANTILE=PROCESS, the value of the ALGORITHM= option in the PROC QUANTREG statement must be SIMPLEX either by default or by specifying it.

The QUANTILE=PROCESS option produces the quantile process estimates table and the quantile process objective function table. The size of these two tables are proportional to the number of the training observations and can be large for a large training data set. You can suppress displaying these two tables by specifying the following ODS EXCLUDE statement:

```
ods exclude ProcessEst ProcessObj;
```

In addition, you can output these tables to data sets for further processing by specifying the following ODS OUTPUT statement:

```
ods output ProcessEst=PE ProcessObj=PO;
```

FQPR< (*suboption***) >**
uses a fast quantile process regression method to approximate quantile process regression on a grid of n equally spaced quantile levels. For more information about the fast quantile process regression method, see the section "Fast Quantile Process Regression" on page 8580. You can specify the following *suboptions*:

L=*value*
specifies the starting quantile level of the quantile-level grid. By default, **L**=$1/2n$ if **U** is not specified, otherwise **L**=**U**$/(2n-1)$.

N=*n*
specifies the number *n* of equally spaced quantile levels at which to fit the quantile process regression.

OBSRATIO=*value*
OR=*value*
specifies the number of equally spaced quantile levels as its ratio to the total number of training observations. For example, if the number of training observations is 1,000 and you specify the OR=0.2 suboption, a quantile process regression model is fit for $n = 0.2 \times 1{,}000 = 200$ equally spaced quantile levels. The FQPR option ignores the OR= suboption if a valid N= suboption is specified.

U=*value*
specifies the ending quantile level of the quantile-level grid. **U**=$(2n-1)/2n$ if **L** is not specified, otherwise **U**=(**L**$+2n-2)/(2n-1)$.

USEALLOBS
UAO
requests that the algorithm use all the observations to fit each of the quantile regression models in the quantile process. When you omit this option, the FQPR algorithm performs a quick-sort-style process for fitting the quantile process model.

If you specify neither the N=*n* nor the OR=*value* suboption, the FQPR option determines the number of quantile levels as the lesser of 100 and half the number of the training observations.

You can use the ALGORITHM= option in the PROC QUANTREG statement to specify the parameter estimation algorithm for the FQPR option. By default, the FQPR option uses the ALGORITHM=IPM option, which specifies the efficient interior point method. The FQPR option also supports the ALGORITHM=SMOOTH option, which specifies the fast smoothing algorithm. The FQPR option does not support the ALGORITHM=SIMPLEX option or the ALGORITHM=INTERIOR option.

Unlike the QUANTILE=PROCESS option, the QUANTILE=FQPR option does not display the quantile process estimates table and the quantile process objective function table. Instead the QUANTILE=FQPR option produces the average parameter estimates table and the average objective function table for the specified quantile-level grid. You can output the quantile process estimates table and the quantile process objective function table as SAS data sets by specifying the following ODS OUTPUT statement:

```
ods output ProcessEst=PE ProcessObj=PO;
```

By default, QUANTILE=0.5, which fits a median regression.

SCALE=*number*

specifies the scale value to use to compute the standardized residuals. By default, the scale is computed as the corrected median of absolute residuals. For more information, see the section "Leverage Point and Outlier Detection" on page 8594.

SEED=*number*

specifies the seed for the random number generator used to compute the MCMB confidence intervals. This seed is also used to randomly select the subgroups for preprocessing when you specify the PP option in the PROC QUANTREG statement. If you do not specify a seed, or if you specify a value less than or equal to 0, the seed is generated by reading the time of day from the computer's clock.

By default or if you specify SEED=0, the QUANTREG procedure generates a seed between one and one billion.

SINGULAR=*value*

sets the tolerance for testing singularity of the information matrix and the crossproducts matrix for the initial least squares estimates. Approximately, the test requires that a pivot be at least this value times the original diagonal value. By default, SINGULAR=1E–12.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > *keyword*=*name* < . . . *keyword*=*name* > < / **COLUMNWISE** > **;**

The OUTPUT statement creates a SAS data set to contain statistics that are calculated after PROC QUANTREG fits models for all specified quantiles that are specified in the QUANTILE= option in the MODEL statement. At least one specification of the form *keyword=name* is required.

All variables in the original data set are included in the new data set, along with the variables that are created from options in the OUTPUT statement. These new variables contain fitted values and estimated quantiles. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

If you specify multiple quantiles in the MODEL statement, the COLUMNWISE option arranges the created OUTPUT data set in columnwise form. This arrangement repeats the input data for each quantile. By default, the OUTPUT data set is created in rowwise form. For each appropriate keyword specified in the OUTPUT statement, one variable for each specified quantile is generated. These variables appear in the sorted order of the specified quantiles.

The following specifications can appear in the OUTPUT statement:

OUT=*SAS-data-set* specifies the new data set. By default, PROC QUANTREG uses the DATA*n* convention to name the new data set.

keyword=name specifies the statistics to include in the output data set and gives names to the new variables. For each desired statistic, specify a *keyword* from the following list of *keywords*, an equal sign, and the name of a variable to contain the statistic.

You can specify the following *keywords*:

LEVERAGE specifies a variable to indicate leverage points. To include this variable in the OUTPUT data set, you must specify the LEVERAGE option in the MODEL statement. See the section "Leverage Point and Outlier Detection" on page 8594 for how to define LEVERAGE.

MAHADIST | MD names a variable to contain the Mahalanobis distance. To include this variable in the OUTPUT data set, you must specify the LEVERAGE option in the MODEL statement.

OUTLIER specifies a variable to indicate outliers. See the section "Leverage Point and Outlier Detection" on page 8594 for how to define OUTLIER.

PREDICTED | P names a variable to contain the estimated response.

QUANTILE | Q names a variable to contain the quantile for which the quantile regression is fitted. If you specify the COLUMNWISE option, this variable is created by default. If multiple quantiles are specified in the MODEL statement and the COLUMNWISE option is not specified, this variable is not created.

RESIDUAL | RES names a variable to contain the residuals (unstandardized):

$$y_i - \mathbf{x}_i' \hat{\boldsymbol{\beta}}$$

ROBDIST | RD names a variable to contain the robust MCD distance. To include this variable in the OUTPUT data set, you must specify the LEVERAGE option in the MODEL statement.

SPLINE | SP names a variable to contain the estimated spline effect, which includes all spline effects in the model and their interactions.

SRESIDUAL | SR names a variable to contain the standardized residuals:

$$\frac{y_i - \mathbf{x}_i' \hat{\boldsymbol{\beta}}}{\hat{\sigma}}$$

See the section "Leverage Point and Outlier Detection" on page 8594 for how to compute σ.

STDP names a variable to contain the estimates of the standard errors of the estimated response.

PERFORMANCE Statement

You can use the PERFORMANCE statement to change default options that affect the performance of PROC QUANTREG and to request tables that show the performance options in effect and the timing details.

PERFORMANCE < *options* > **;**

You can specify the following *options*:

CPUCOUNT=*number* **| ACTUAL**

specifies the number of processors to use in the computation of the interior point algorithm. CPUCOUNT=ACTUAL sets CPUCOUNT to be the number of physical processors available, which this can be less than the physical number of CPUs if the SAS process has been restricted by system administration tools. You can specify any integer from 1 to 1024 for *number*. Setting CPUCOUNT= to a *number* greater than the actual number of available CPUs might result in reduced performance. If CPUCOUNT=1, then NOTHREADS is in effect, and PROC QUANTREG uses singly threaded code. This option overrides the SAS system option CPUCOUNT=.

DETAILS
creates the PerfSettings table that shows the performance settings in effect and the "Timing" table that provides a broad timing breakdown of the PROC QUANTREG step.

THREADS
enables multithreaded computation for the interior point algorithm. If you do not specify the ALGORITHM=INTERIOR option in the PROC QUANTREG statement, then PROC QUANTREG ignores this option and uses singly threaded code. This option overrides the SAS system option THREADS | NOTHREADS.

NOTHREADS
disables multithreaded computation for the interior point algorithm. This option overrides the SAS system option THREADS | NOTHREADS.

TEST Statement

< *label***: > TEST** *effects* **<** / *options* **> ;**

In quantile regression analysis, you might be interested in testing whether a covariate effect is statistically significant for a given quantile. In other situations, you might be interested in testing whether the coefficients of a covariate are the same across a set of quantiles. You can use the TEST Statement to perform these tests.

You can submit multiple TEST statements, provided that they appear after the MODEL statement. The optional *label*, which must be a valid SAS name, is used to identify output from the corresponding TEST statement. For more information about these tests, see the section "Linear Test" on page 8585.

Testing Effects of Covariates

You can use TEST statement to obtain a test for the canonical linear hypothesis concerning the parameters of the tested effects,

$$\beta_j = 0, \quad j = i_1, \ldots, i_q$$

where q is the total number of parameters of the tested effects. The tested *effects* can be any set of effects in the MODEL statement. You can specify three types of tests (Wald, likelihood ratio, and rank methods) for testing effects of covariates by using the following *options* in the TEST statement after a slash (/):

WALD
requests Wald tests.

LR
requests likelihood ratio tests.

RANKSCORE < (NORMAL | WILCOXON | SIGN | TAU) >
requests rank tests. The NORMAL, WILCOXON, and SIGN functions are implemented and suitable for iid error models, and the TAU score function is implemented and appropriate for non-iid error models. By default, the TAU score function is used. See Koenker (2005) for more information about the score functions.

Testing for Heteroscedasticity

You can test whether there is any difference among the estimated coefficients across quantiles if several quantiles are specified in the MODEL statement. The test for such heteroscedasticity can be requested by the option QINTERACT after a slash (/) in the TEST statement. See Example 103.5.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement specifies a weight *variable* in the input data set.

To request weighted quantile regression, place the weights in a *variable*. The values of the WEIGHT *variable* can be nonintegral and are not truncated. Observations with nonpositive or missing values for the weight variable do not contribute to the fit of the model. For more information about weighted quantile regression, see the section "Details: QUANTREG Procedure" on page 8572.

Details: QUANTREG Procedure

Quantile Regression as an Optimization Problem

The generic model for linear quantile regression is

$$Q_\tau(Y|X = \mathbf{x}) = Q_{Y|\mathbf{x}}(\tau) = \mathbf{x}'\boldsymbol{\beta}(\tau)$$

where Y is the response random variable, $\mathbf{x}$ is the explanatory covariates vector, $\boldsymbol{\beta}(\tau) = (\beta_1(\tau), \ldots, \beta_p(\tau))'$ is the $(p \times 1)$ vector of the functional model parameters at the quantile level τ, and $Q_{Y|\mathbf{x}}(\cdot)$ is the quantile function for Y conditional on $X = \mathbf{x}$.

This generic model is compatible with the following linear model:

$$y_i = \mathbf{x}_i'\boldsymbol{\beta}(\tau) + \epsilon_i(\tau) \text{ for } i = 1, \ldots, n$$

where y_i is the response value, $\mathbf{x}_i$ is the explanatory covariates vector, and $\epsilon_i(\tau) = y_i - Q_{Y|\mathbf{x}_i}(\tau)$ is an unknown error.

L_1 regression, also known as median regression, is a natural extension of the sample median when the response is conditioned on the covariates. In L_1 regression, the least absolute residuals estimate $\hat{\boldsymbol{\beta}}_{LAR}$, referred to as the L_1-*norm estimate*, is obtained as the solution of the following minimization problem:

$$\min_{\boldsymbol{\beta}\in\mathbf{R}^p} \sum_{i=1}^{n} |y_i - \mathbf{x}_i'\boldsymbol{\beta}|$$

More generally, for quantile regression Koenker and Bassett (1978) defined the τ *regression quantile*, $0 < \tau < 1$, as any solution to the following minimization problem:

$$\min_{\boldsymbol{\beta}\in\mathbf{R}^p} \left[\sum_{i\in\{i:y_i\geq\mathbf{x}_i'\boldsymbol{\beta}\}} \tau|y_i - \mathbf{x}_i'\boldsymbol{\beta}| + \sum_{i\in\{i:y_i<\mathbf{x}_i'\boldsymbol{\beta}\}} (1-\tau)|y_i - \mathbf{x}_i'\boldsymbol{\beta}| \right]$$

The solution is denoted as $\hat{\boldsymbol{\beta}}(\tau)$, and the L_1-norm estimate corresponds to $\hat{\boldsymbol{\beta}}(1/2)$. The τ regression quantile is an extension of the τ sample quantile $\hat{\xi}(\tau)$, which can be formulated as the solution of

$$\min_{\xi \in \mathbf{R}} \left[\sum_{i \in \{i : y_i \geq \xi\}} \tau |y_i - \xi| + \sum_{i \in \{i : y_i < \xi\}} (1-\tau)|y_i - \xi| \right]$$

If you specify weights $w_i, i = 1, \ldots, n$, with the WEIGHT statement, weighted quantile regression is carried out by solving

$$\min_{\boldsymbol{\beta}_w \in \mathbf{R}^p} \left[\sum_{i \in \{i : y_i \geq \mathbf{x}_i' \boldsymbol{\beta}_w\}} w_i \tau |y_i - \mathbf{x}_i' \boldsymbol{\beta}_w| + \sum_{i \in \{i : y_i < \mathbf{x}_i' \boldsymbol{\beta}_w\}} w_i (1-\tau)|y_i - \mathbf{x}_i' \boldsymbol{\beta}_w| \right]$$

Weighted regression quantiles $\boldsymbol{\beta}_w$ can be used for L-estimation (Koenker and Zhao 1994).

Optimization Algorithms

The optimization problem for median regression has been formulated and solved as a linear programming (LP) problem since the 1950s. Variations of the simplex algorithm, especially the method of Barrodale and Roberts (1973), have been widely used to solve this problem. The simplex algorithm is computationally demanding in large statistical applications, and in theory the number of iterations can increase exponentially with the sample size. This algorithm is often useful with data that contain no more than tens of thousands of observations.

Several alternatives have been developed to handle L_1 regression for larger data sets. The interior point approach of Karmarkar (1984) solves a sequence of quadratic problems in which the relevant interior of the constraint set is approximated by an ellipsoid. The worst-case performance of the interior point algorithm has been proved to be better than the worst-case performance of the simplex algorithm. More important, experience has shown that the interior point algorithm is advantageous for larger problems.

Like L_1 regression, general quantile regression fits nicely into the standard primal-dual formulations of linear programming.

In addition to the interior point method, various heuristic approaches are available for computing L_1-type solutions. Among these, the finite smoothing algorithm of Madsen and Nielsen (1993) is the most useful. It approximates the L_1-type objective function with a smoothing function, so that the Newton-Raphson algorithm can be used iteratively to obtain a solution after a finite number of iterations. The smoothing algorithm extends naturally to general quantile regression.

The QUANTREG procedure implements the simplex, interior point, and smoothing algorithms. The remainder of this section describes these algorithms in more detail.

Simplex Algorithm

Let $\boldsymbol{\mu} = [\mathbf{y} - \mathbf{A}'\boldsymbol{\beta}]_+$, $\boldsymbol{\nu} = [\mathbf{A}'\boldsymbol{\beta} - \mathbf{y}]_+$, $\boldsymbol{\phi} = [\boldsymbol{\beta}]_+$, and $\boldsymbol{\varphi} = [-\boldsymbol{\beta}]_+$, where $\mathbf{y} = (y_1, \ldots, y_n)'$ is the response vector, $\mathbf{A}' = (\mathbf{x}_1, \ldots, \mathbf{x}_n)'$ is the $(n \times p)$ regressor matrix, and $[z]_+$ is the nonnegative part of z.

Let $D_{LAR}(\boldsymbol{\beta}) = \sum_{i=1}^{n} |y_i - \mathbf{x}_i'\boldsymbol{\beta}|$. For the L_1 problem, the simplex approach solves $\min_{\boldsymbol{\beta}} D_{LAR}(\boldsymbol{\beta})$ by reformulating it as the constrained minimization problem

$$\min_{\boldsymbol{\beta}}\{\mathbf{e}'\boldsymbol{\mu} + \mathbf{e}'\boldsymbol{\nu} | \mathbf{y} = \mathbf{A}'\boldsymbol{\beta} + \boldsymbol{\mu} - \boldsymbol{\nu}, \{\boldsymbol{\mu}, \boldsymbol{\nu}\} \in \mathbf{R}_+^n\}$$

where $\mathbf{e}$ denotes an $(n \times 1)$ vector of ones.

Let $\mathbf{B} = [\mathbf{A}' \ \ -\mathbf{A}' \ \ \mathbf{I} \ \ -\mathbf{I}]$, $\boldsymbol{\theta} = (\boldsymbol{\phi}' \ \boldsymbol{\varphi}' \ \boldsymbol{\mu}' \ \boldsymbol{\nu}')'$, and $\mathbf{d} = (\mathbf{0}' \ \mathbf{0}' \ \mathbf{e}' \ \mathbf{e}')'$, where $\mathbf{0}' = (0\,0\,\ldots\,0)_p$. The reformulation presents a standard LP problem:

$$(P) \qquad \min_{\boldsymbol{\theta}} \mathbf{d}'\boldsymbol{\theta}; \text{subject to } \mathbf{B}\boldsymbol{\theta} = \mathbf{y}, \boldsymbol{\theta} \geq \mathbf{0}$$

This problem has the following dual formulation:

$$(D) \qquad \max_{z} \mathbf{y}'\mathbf{z}; \text{subject to } \mathbf{B}'\mathbf{z} \leq \mathbf{d}$$

This formulation can be simplified as

$$\max_{z} \mathbf{y}'\mathbf{z}; \quad \text{subject to } \mathbf{A}\mathbf{z} = \mathbf{0}, \mathbf{z} \in [-1, 1]^n$$

By setting $\boldsymbol{\eta} = \frac{1}{2} + \frac{1}{2}\mathbf{e}, \mathbf{b} = \frac{1}{2}\mathbf{A}\mathbf{e}$, the problem becomes

$$\max_{\boldsymbol{\eta}} \mathbf{y}'\boldsymbol{\eta}; \quad \text{subject to } \mathbf{A}\boldsymbol{\eta} = \mathbf{b}, \boldsymbol{\eta} \in [0, 1]^n$$

For quantile regression, the minimization problem is $\min_{\boldsymbol{\beta}} \sum \rho_\tau(y_i - \mathbf{x}_i'\boldsymbol{\beta})$, and a similar set of steps leads to the dual formulation

$$\max_{z} \mathbf{y}'\mathbf{z}; \quad \text{subject to } \mathbf{A}\mathbf{z} = (1 - \tau)\mathbf{A}\mathbf{e}, \mathbf{z} \in [0, 1]^n$$

The QUANTREG procedure solves this LP problem by using the simplex algorithm of Barrodale and Roberts (1973). This algorithm exploits the special structure of the coefficient matrix **B** by solving the primary LP problem (P) in two stages: The first stage chooses the columns in $\mathbf{A}'$ or $-\mathbf{A}'$ as pivotal columns. The second stage interchanges the columns in **I** or –**I** as basis or nonbasis columns, respectively. The algorithm obtains an optimal solution by executing these two stages interactively. Moreover, because of the special structure of **B**, only the main data matrix **A** is stored in the current memory.

Although this special version of the simplex algorithm was introduced for median regression, it extends naturally to quantile regression for any given quantile and even to the entire quantile process (Koenker and d'Orey 1994). It greatly reduces the computing time that is required by the general simplex algorithm, and it is suitable for data sets with fewer than 5,000 observations and 50 variables.

Interior Point Algorithm

The ALGORITHM=INTERIOR option implements an interior point algorithm. This algorithm uses the primal-dual predictor-corrector method that is proposed by Lustig, Marsten, and Shanno (1992). Roos, Terlaky, and Vial (1997) provide more information about this particular algorithm. The following brief introduction of this algorithm uses the notation in the first reference.

To be consistent with the conventional linear programming setting, let $\mathbf{c} = -\mathbf{y}$, let $\mathbf{b} = (1-\tau)\mathbf{A}\mathbf{e}$, and let u be the general upper bound. The dual form of quantile regression solves the following linear programming primal problem:

$$\min\{\mathbf{c}'\mathbf{z}\}; \text{ subject to } \mathbf{A}\mathbf{z} = \mathbf{b}, \mathbf{0} \le \mathbf{z} \le \mathbf{u}$$

This primal problem has n variables. The index i denotes a variable number, and k denotes an iteration number. If k is used as a subscript or superscript, it denotes "of iteration k."

Let $\mathbf{v}$ be the primal slack so that $\mathbf{z} + \mathbf{v} = \mathbf{u}$. Associate dual variables w with these constraints. The interior point algorithm solves the system of equations to satisfy the Karush-Kuhn-Tucker (KKT) conditions for optimality:

$$\begin{aligned}
\mathbf{b} &= \mathbf{A}\mathbf{z} \\
\mathbf{u} &= \mathbf{z} + \mathbf{v} \\
\mathbf{c} &= \mathbf{A}'\mathbf{t} + \mathbf{s} - \mathbf{w} \\
\mathbf{0} &= \mathbf{Z}\mathbf{S}\mathbf{e} \\
\mathbf{0} &= \mathbf{V}\mathbf{W}\mathbf{e} \\
\mathbf{z}, \mathbf{s}, \mathbf{v}, \mathbf{w} &\ge \mathbf{0}
\end{aligned}$$

where $\mathbf{W} = \mathrm{diag}(\mathbf{w})$ $\left(\text{that is, } W_{i,j} = \begin{cases} w_i & \text{for } i = j \\ 0 & \text{otherwise} \end{cases}\right)$, $\mathbf{V} = \mathrm{diag}(\mathbf{v})$, $\mathbf{Z} = \mathrm{diag}(\mathbf{z})$, $\mathbf{S} = \mathrm{diag}(\mathbf{s})$.

These are the conditions for feasibility, with the addition of *complementarity conditions* $\mathbf{Z}\mathbf{S}\mathbf{e} = \mathbf{0}$ and $\mathbf{V}\mathbf{W}\mathbf{e} = \mathbf{0}$. The equality $\mathbf{c}'\mathbf{z} = \mathbf{b}'\mathbf{t} - \mathbf{u}'\mathbf{w}$ must occur at the optimum. Complementarity forces the optimal objectives of the primal and dual to be equal, $\mathbf{c}'\mathbf{z}_{opt} = \mathbf{b}'\mathbf{t}_{opt} - \mathbf{u}'\mathbf{w}_{opt}$, because

$$\begin{aligned}
0 &= \mathbf{v}'_{opt}\mathbf{w}_{opt} = (\mathbf{u} - \mathbf{z}_{opt})'\mathbf{w}_{opt} = \mathbf{u}'\mathbf{w}_{opt} - \mathbf{z}'_{opt}\mathbf{w}_{opt} \\
0 &= \mathbf{z}'_{opt}\mathbf{s}_{opt} = \mathbf{s}'_{opt}\mathbf{z}_{opt} = (\mathbf{c} - \mathbf{A}'\mathbf{t}_{opt} + \mathbf{w}_{opt})'\mathbf{z}_{opt} \\
&= \mathbf{c}'\mathbf{z}_{opt} - \mathbf{t}'_{opt}(\mathbf{A}\mathbf{z}_{opt}) + \mathbf{w}'_{opt}\mathbf{z}_{opt} = \mathbf{c}'\mathbf{z}_{opt} - \mathbf{b}'\mathbf{t}_{opt} + \mathbf{u}'\mathbf{w}_{opt}
\end{aligned}$$

Therefore

$$0 = \mathbf{c}'\mathbf{z}_{opt} - \mathbf{b}'\mathbf{t}_{opt} + \mathbf{u}'\mathbf{w}_{opt}$$

The *duality gap*, $\mathbf{c}'\mathbf{z} - \mathbf{b}'\mathbf{t} + \mathbf{u}'\mathbf{w}$, measures the convergence of the algorithm. You can specify a tolerance for this convergence criterion in the TOLERANCE= option in the PROC QUANTREG statement.

Before the optimum is reached, it is possible for a solution $(\mathbf{z}, \mathbf{t}, \mathbf{s}, \mathbf{v}, \mathbf{w})$ to violate the KKT conditions in one of several ways:

- Primal bound constraints can be broken: $\boldsymbol{\delta}_b = \mathbf{u} - \mathbf{z} - \mathbf{v} \neq \mathbf{0}$.
- Primal constraints can be broken: $\boldsymbol{\delta}_c = \mathbf{b} - \mathbf{A}\mathbf{z} \neq \mathbf{0}$.
- Dual constraints can be broken: $\boldsymbol{\delta}_d s = \mathbf{c} - \mathbf{A}'\mathbf{t} - \mathbf{s} + \mathbf{w} \neq \mathbf{0}$.
- Complementarity conditions are unsatisfied: $\mathbf{z}'\mathbf{s} \neq 0$ and $\mathbf{v}'\mathbf{w} \neq 0$.

The interior point algorithm works by using Newton's method to find a direction $(\boldsymbol{\Delta} z^k, \boldsymbol{\Delta} t^k, \boldsymbol{\Delta} s^k, \boldsymbol{\Delta} v^k, \boldsymbol{\Delta} w^k)$ to move from the current solution $(\mathbf{z}^k, \mathbf{t}^k, \mathbf{s}^k, \mathbf{v}^k, \mathbf{w}^k)$ toward a better solution:

$$(\mathbf{z}^{k+1}, \mathbf{t}^{k+1}, \mathbf{s}^{k+1}, \mathbf{v}^{k+1}, \mathbf{w}^{k+1}) = (\mathbf{z}^k, \mathbf{t}^k, \mathbf{s}^k, \mathbf{v}^k, \mathbf{w}^k) + \kappa(\boldsymbol{\Delta} z^k, \boldsymbol{\Delta} t^k, \boldsymbol{\Delta} s^k, \boldsymbol{\Delta} v^k, \boldsymbol{\Delta} w^k)$$

κ is the *step length* and is assigned a value as large as possible, but not so large that a $\mathbf{z}_i^{k+1}$ or $\mathbf{s}_i^{k+1}$ is "too close" to 0. You can control the step length in the KAPPA= option in the PROC QUANTREG statement.

The QUANTREG procedure implements a predictor-corrector variant of the primal-dual interior point algorithm. First, Newton's method is used to find a direction $(\boldsymbol{\Delta} z_{aff}^k, \boldsymbol{\Delta} t_{aff}^k, \boldsymbol{\Delta} s_{aff}^k, \boldsymbol{\Delta} v_{aff}^k, \boldsymbol{\Delta} w_{aff}^k)$ in which to move. This is known as the *affine* step.

In iteration k, the affine step system that must be solved is

$$\begin{aligned}
\boldsymbol{\delta}_b &= \boldsymbol{\Delta} z_{aff} + \boldsymbol{\Delta} v_{aff} \\
\boldsymbol{\delta}_c &= \mathbf{A}\boldsymbol{\Delta} z_{aff} \\
\boldsymbol{\delta}_d &= \mathbf{A}'\boldsymbol{\Delta} t_{aff} + \boldsymbol{\Delta} s_{aff} - \boldsymbol{\Delta} w_{aff} = \boldsymbol{\delta}_d \\
-\mathbf{ZSe} &= \mathbf{S}\boldsymbol{\Delta} z_{aff} + \mathbf{Z}\boldsymbol{\Delta} s_{aff} \\
-\mathbf{VWe} &= \mathbf{V}\boldsymbol{\Delta} w_{aff} + \mathbf{W}\boldsymbol{\Delta} z_{aff}
\end{aligned}$$

Therefore, the following computations are involved in solving the affine step, where κ is the *step length* as before:

$$\begin{aligned}
\boldsymbol{\Theta} &= \mathbf{S}\mathbf{Z}^{-1} + \mathbf{W}\mathbf{V}^{-1} \\
\boldsymbol{\rho} &= \boldsymbol{\Theta}^{-1}(\boldsymbol{\delta}_d + (\mathbf{S} - \mathbf{W})\mathbf{e} - \mathbf{V}^{-1}\mathbf{W}\boldsymbol{\delta}_b) \\
\boldsymbol{\Delta} t_{aff} &= (\mathbf{A}\boldsymbol{\Theta}^{-1}\mathbf{A}')^{-1}(\boldsymbol{\delta}_c + \mathbf{A}\boldsymbol{\rho}) \\
\boldsymbol{\Delta} z_{aff} &= \boldsymbol{\Theta}^{-1}\mathbf{A}'\boldsymbol{\Delta} t_{aff} - \boldsymbol{\rho} \\
\boldsymbol{\Delta} v_{aff} &= \boldsymbol{\delta}_b - \boldsymbol{\Delta} z_{aff} \\
\boldsymbol{\Delta} w_{aff} &= -\mathbf{We} - \mathbf{V}^{-1}\mathbf{W}\boldsymbol{\Delta} z_{aff} \\
\boldsymbol{\Delta} s_{aff} &= -\mathbf{Se} - \mathbf{Z}^{-1}\mathbf{S}\boldsymbol{\Delta} z_{aff} \\
(\mathbf{z}_{aff}, \mathbf{t}_{aff}, \mathbf{s}_{aff}, \mathbf{v}_{aff}, \mathbf{w}_{aff}) &= (\mathbf{z}, \mathbf{t}, \mathbf{s}, \mathbf{v}, \mathbf{w}) + \kappa(\boldsymbol{\Delta} z_{aff}, \boldsymbol{\Delta} t_{aff}, \boldsymbol{\Delta} s_{aff}, \boldsymbol{\Delta} v_{aff}, \boldsymbol{\Delta} w_{aff})
\end{aligned}$$

The success of the affine step is gauged by calculating the complementarity of $\mathbf{z}'\mathbf{s}$ and $\mathbf{v}'\mathbf{w}$ at $(\mathbf{z}_{aff}^k, \mathbf{t}_{aff}^k, \mathbf{s}_{aff}^k, \mathbf{v}_{aff}^k, \mathbf{w}_{aff}^k)$ and comparing it with the complementarity at the starting point $(\mathbf{z}^k, \mathbf{t}^k, \mathbf{s}^k, \mathbf{v}^k, \mathbf{w}^k)$.

If the affine step was successful in reducing the complementarity by a substantial amount, the need for centering is not great. Therefore, a value close to 0 is assigned to σ in the following second linear system, which is used to determine a centering vector.

The following linear system is solved to determine a centering vector $(\mathbf{\Delta} z_c, \mathbf{\Delta} t_c, \mathbf{\Delta} s_c, \mathbf{\Delta} v_c, \mathbf{\Delta} w_c)$ from $(\mathbf{z}_{aff}, \mathbf{t}_{aff}, \mathbf{s}_{aff}, \mathbf{v}_{aff}, \mathbf{w}_{aff})$:

$$\begin{aligned}
\mathbf{\Delta} z_c + \mathbf{\Delta} v_c &= \mathbf{0} \\
\mathbf{A}\mathbf{\Delta} z_c &= \mathbf{0} \\
\mathbf{A}'\mathbf{\Delta} t_c + \mathbf{\Delta} s_c - \mathbf{\Delta} w_c &= \mathbf{0} \\
\mathbf{S}\mathbf{\Delta} z_c + \mathbf{Z}\mathbf{\Delta} s_c &= -\mathbf{Z}_{aff}\mathbf{S}_{aff}\mathbf{e} + \sigma\boldsymbol{\mu}\mathbf{e} \\
\mathbf{V}\mathbf{\Delta} w_c + \mathbf{W}\mathbf{\Delta} v_c &= -\mathbf{V}_{aff}\mathbf{W}_{aff}\mathbf{e} + \sigma\boldsymbol{\mu}\mathbf{e}
\end{aligned}$$

where $\zeta_{start} = \mathbf{z}'\mathbf{s} + \mathbf{v}'\mathbf{w}$, complementarity at the start of the iteration

$\zeta_{aff} = \mathbf{z}'_{aff}\mathbf{s}_{aff} + \mathbf{v}'_{aff}\mathbf{w}_{aff}$, the affine complementarity

$\mu = \zeta_{aff}/2n$, the average complementarity

$$\sigma = (\zeta_{aff}/\zeta_{start})^3$$

However, if the affine step was unsuccessful, then centering is deemed beneficial, and a value close to 1.0 is assigned to σ. In other words, the value of σ is adaptively altered depending on the progress made toward the optimum.

Therefore, the following computations are involved in solving the centering step:

$$\begin{aligned}
\boldsymbol{\rho} &= \mathbf{\Theta}^{-1}(\sigma\mu(\mathbf{Z}^{-1} - \mathbf{V}^{-1})\mathbf{e} - \mathbf{Z}^{-1}\mathbf{Z}_{aff}\mathbf{S}_{aff}\mathbf{e} + \mathbf{V}^{-1}\mathbf{V}_{aff}\mathbf{W}_{aff}\mathbf{e}) \\
\mathbf{\Delta} t_c &= (\mathbf{A}\mathbf{\Theta}^{-1}\mathbf{A}')^{-1}\mathbf{A}\boldsymbol{\rho} \\
\mathbf{\Delta} z_c &= \mathbf{\Theta}^{-1}\mathbf{A}'\mathbf{\Delta} t_c - \boldsymbol{\rho} \\
\mathbf{\Delta} v_c &= -\mathbf{\Delta} z_c \\
\mathbf{\Delta} w_c &= \sigma\mu\mathbf{V}^{-1}\mathbf{e} - \mathbf{V}^{-1}\mathbf{V}_{aff}\mathbf{W}_{aff}\mathbf{e} - \mathbf{V}^{-1}\mathbf{W}_{aff}\mathbf{\Delta} v_c \\
\mathbf{\Delta} s_c &= \sigma\mu\mathbf{Z}^{-1}\mathbf{e} - \mathbf{Z}^{-1}\mathbf{Z}_{aff}\mathbf{S}_{aff}\mathbf{e} - \mathbf{Z}^{-1}\mathbf{S}_{aff}\mathbf{\Delta} z_c
\end{aligned}$$

Then

$$\begin{aligned}
(\mathbf{\Delta} z, \mathbf{\Delta} t, \mathbf{\Delta} s, \mathbf{\Delta} v, \mathbf{\Delta} w) &= (\mathbf{\Delta} z_{aff}, \mathbf{\Delta} t_{aff}, \mathbf{\Delta} s_{aff}, \mathbf{\Delta} v_{aff}, \mathbf{\Delta} w_{aff}) + (\mathbf{\Delta} z_c, \mathbf{\Delta} t_c, \mathbf{\Delta} s_c, \mathbf{\Delta} v_c, \mathbf{\Delta} w_c) \\
(\mathbf{z}^{k+1}, \mathbf{t}^{k+1}, \mathbf{s}^{k+1}, \mathbf{v}^{k+1}, \mathbf{w}^{k+1}) &= (\mathbf{z}^k, \mathbf{t}^k, \mathbf{s}^k, \mathbf{v}^k, \mathbf{w}^k) + \kappa(\mathbf{\Delta} z, \mathbf{\Delta} t, \mathbf{\Delta} s, \mathbf{\Delta} v, \mathbf{\Delta} w)
\end{aligned}$$

where, as before, κ is the step length, which is assigned a value as large as possible but not so large that a $\mathbf{z}_i^{k+1}$, $\mathbf{s}_i^{k+1}$, $\mathbf{v}_i^{k+1}$, or $\mathbf{w}_i^{k+1}$ is "too close" to 0.

Although the predictor-corrector variant entails solving two linear systems instead of one, fewer iterations are usually required to reach the optimum. The additional overhead of the second linear system is small because the matrix $(\mathbf{A}\mathbf{\Theta}^{-1}\mathbf{A}')$ has already been factorized in order to solve the first linear system.

You can specify the starting point in the INEST= option in the PROC QUANTREG statement. By default, the starting point is set to be the least squares estimate.

Efficient Interior Point Algorithm

The ALGORITHM=IPM option implements a more efficient interior point algorithm than the one that is used when ALGORITHM=INTERIOR. The computing strategy of the ALGORITHM=IPM option is the same as the strategy for the ALGORITHM=INTERIOR option, but the ALGORITHM=IPM option implements the algorithm by using more efficient matrix functions. The ALGORITHM=IPM option uses the complementarity value to measure the convergence of the algorithm, which is different from the dual gap value that is used when ALGORITHM=INTERIOR. The complementarity value is defined as $(\mathbf{z}'\mathbf{s} + \mathbf{v}'\mathbf{w})$. You can specify a tolerance for this complementarity convergence criterion by using the TOLERANCE= option in the PROC QUANTREG statement. Unlike the ALGORITHM=INTERIOR option, the ALGORITHM=IPM option does not support the KAPPA= option.

Smoothing Algorithm

To minimize the sum of the absolute residuals $D_{LAR}(\boldsymbol{\beta})$, the smoothing algorithm approximates the nondifferentiable function D_{LAR} by the following smooth function(which is referred to as the Huber function),

$$D_\gamma(\boldsymbol{\beta}) = \sum_{i=1}^{n} H_\gamma(r_i(\boldsymbol{\beta}))$$

where

$$H_\gamma(t) = \begin{cases} t^2/(2\gamma) & \text{if } |t| \le \gamma \\ |t| - \gamma/2 & \text{if } |t| > \gamma \end{cases}$$

Here $r_i(\boldsymbol{\beta}) = y_i - \mathbf{x}_i'\boldsymbol{\beta}$, and the threshold γ is a positive real number. The function D_γ is continuously differentiable, and a minimizer β_γ of D_γ is close to a minimizer $\hat{\boldsymbol{\beta}}_{LAR}$ of $D_{LAR}(\boldsymbol{\beta})$ when γ is close to 0.

The advantage of the smoothing algorithm as described in Madsen and Nielsen (1993) is that the L_1 solution $\hat{\boldsymbol{\beta}}_{LAR}$ can be detected when $\gamma > 0$ is small. In other words, it is not necessary to let γ converge to 0 in order to find a minimizer of $D_{LAR}(\boldsymbol{\beta})$. The algorithm terminates before going through the entire sequence of values of γ that are generated by the algorithm. Convergence is indicated by no change of the status of residuals $r_i(\boldsymbol{\beta})$ as γ goes through this sequence.

The smoothing algorithm extends naturally from L_1 regression to general quantile regression (Chen 2007). The function

$$D_{\rho_\tau}(\boldsymbol{\beta}) = \sum_{i=1}^{n} \rho_\tau(y_i - \mathbf{x}_i'\boldsymbol{\beta})$$

can be approximated by the smooth function

$$D_{\gamma,\tau}(\boldsymbol{\beta}) = \sum_{i=1}^{n} H_{\gamma,\tau}(r_i(\boldsymbol{\beta}))$$

where

$$H_{\gamma,\tau}(t) = \begin{cases} t(\tau-1) - \frac{1}{2}(\tau-1)^2\gamma & \text{if } t \le (\tau-1)\gamma \\ \frac{t^2}{2\gamma} & \text{if } (\tau-1)\gamma \le t \le \tau\gamma \\ t\tau - \frac{1}{2}\tau^2\gamma & \text{if } t \ge \tau\gamma \end{cases}$$

The function $H_{\gamma,\tau}$ is determined by whether $r_i(\boldsymbol{\beta}) \leq (\tau-1)\gamma$, $r_i(\boldsymbol{\beta}) \geq \tau\gamma$, or $(\tau-1)\gamma \leq r_i(\boldsymbol{\beta}) \leq \tau\gamma$. These inequalities divide $\mathbf{R}^p$ into subregions that are separated by the parallel hyperplanes $r_i(\boldsymbol{\beta}) = (\tau-1)\gamma$ and $r_i(\boldsymbol{\beta}) = \tau\gamma$. The set of all such hyperplanes is denoted by $B_{\gamma,\tau}$:

$$B_{\gamma,\tau} = \{\boldsymbol{\beta} \in \mathbf{R}^p | \exists i : r_i(\boldsymbol{\beta}) = (\tau-1)\gamma \text{ or } r_i(\boldsymbol{\beta}) = \tau\gamma\}$$

Define the sign vector $\mathrm{s}_\gamma(\boldsymbol{\beta}) = (s_1(\boldsymbol{\beta}), \ldots, s_n(\boldsymbol{\beta}))'$ as

$$s_i = s_i(\boldsymbol{\beta}) = \begin{cases} -1 & \text{if } r_i(\boldsymbol{\beta}) \leq (\tau-1)\gamma \\ 0 & \text{if } (\tau-1)\gamma \leq r_i(\boldsymbol{\beta}) \leq \tau\gamma \\ 1 & \text{if } r_i(\boldsymbol{\beta}) \geq \tau\gamma \end{cases}$$

and introduce

$$w_i = w_i(\boldsymbol{\beta}) = 1 - s_i^2(\boldsymbol{\beta})$$

Therefore,

$$\begin{aligned} H_{\gamma,\tau}(r_i(\boldsymbol{\beta})) &= \frac{1}{2\gamma} w_i r_i^2(\boldsymbol{\beta}) \\ &+ s_i[\frac{1}{2} r_i(\boldsymbol{\beta}) + \frac{1}{4}(1-2\tau)\gamma + s_i(r_i(\boldsymbol{\beta})(\tau - \frac{1}{2}) - \frac{1}{4}(1-2\tau+2\tau^2)\gamma)] \end{aligned}$$

This equation yields

$$D_{\gamma,\tau}(\boldsymbol{\beta}) = \frac{1}{2\gamma}\mathbf{r}'\mathbf{W}_{\gamma,\tau}\mathbf{r} + \mathbf{v}'(s)\mathbf{r} + c(s)$$

where $\mathbf{W}_{\gamma,\tau}$ is the diagonal $n \times n$ matrix with diagonal elements $w_i(\boldsymbol{\beta})$, $\mathbf{v}'(s) = (s_1((2\tau-1)s_1+1)/2, \ldots, s_n((2\tau-1)s_n+1)/2)$, $c(s) = \sum[\frac{1}{4}(1-2\tau)\gamma s_i - \frac{1}{4}s_i^2(1-2\tau+2\tau^2)\gamma]$, and $r(\boldsymbol{\beta}) = (r_1(\boldsymbol{\beta}), \ldots, r_n(\boldsymbol{\beta}))'$.

The gradient of $D_{\gamma,\tau}$ is given by

$$D^{(1)}_{\gamma,\tau}(\boldsymbol{\beta}) = -\mathbf{A}[\frac{1}{\gamma}\mathbf{W}_{\gamma,\tau}(\boldsymbol{\beta})r(\boldsymbol{\beta}) + v(s)]$$

For $\boldsymbol{\beta} \in \mathbf{R}^p \backslash B_{\gamma,\tau}$ the Hessian exists and is given by

$$D^{(2)}_{\gamma,\tau}(\boldsymbol{\beta}) = \frac{1}{\gamma}\mathbf{A}\mathbf{W}_{\gamma,\tau}(\boldsymbol{\beta})\mathbf{A}'$$

The gradient is a continuous function in $\mathbf{R}^p$, whereas the Hessian is piecewise constant.

Following Madsen and Nielsen (1993), the vector s is referred to as a *γ-feasible sign vector* if there exists $\boldsymbol{\beta} \in \mathbf{R}^p \backslash B_{\gamma,\tau}$ with $\mathrm{s}_\gamma(\boldsymbol{\beta}) = \mathrm{s}$. If s is γ-feasible, then Q_s is defined as the quadratic function $Q_s(\boldsymbol{\alpha})$ that is derived from $D_{\gamma,\tau}(\boldsymbol{\beta})$ by substituting s for s_γ. Thus, for any $\boldsymbol{\beta}$ with $\mathrm{s}_\gamma = \mathrm{s}$,

$$Q_s(\boldsymbol{\alpha}) = \frac{1}{2}(\boldsymbol{\alpha}-\boldsymbol{\beta})' D^{(2)}_{\gamma,\tau}(\boldsymbol{\beta})(\boldsymbol{\alpha}-\boldsymbol{\beta}) + D^{(1)}_{\gamma,\tau}(\boldsymbol{\beta})(\boldsymbol{\alpha}-\boldsymbol{\beta}) + D_{\gamma,\tau}(\boldsymbol{\beta})$$

In the domain $C_s = \{\alpha | \mathrm{s}_\gamma(\alpha) = \mathrm{s}\}$,

$$D_{\gamma,\tau}(\alpha) = Q_s(\boldsymbol{\alpha})$$

For each $\gamma > 0$ and $\theta \in \mathbf{R}^p$, there can be one or several corresponding quadratics, Q_s. If $\theta \notin B_{\gamma,\tau}$, then Q_s is characterized by θ and γ. However, for $\theta \in B_{\gamma,\tau}$, the quadratic is not unique. Therefore, the following *reference* determines the quadratic:

$$(\gamma, \theta, \mathrm{s})$$

Again following Madsen and Nielsen (1993), let $(\gamma, \theta, \mathrm{s})$ be a *feasible reference* if s is a γ-feasible sign vector, where $\theta \in C_s$, and let $(\gamma, \theta, \mathrm{s})$ be a *solution reference* if s is feasible and θ minimizes $D_{\gamma,\tau}$.

The smoothing algorithm for minimizing D_{ρ_τ} is based on minimizing $D_{\gamma,\tau}$ for a set of decreasing γ. For each new value of γ, information from the previous solution is used. Finally, when γ is small enough, a solution can be found by the following modified Newton-Raphson algorithm as stated by Madsen and Nielsen (1993):

1. Find an initial solution reference $(\gamma, \boldsymbol{\beta}_\gamma, \mathrm{s})$.
2. Repeat the following substeps until $\gamma = 0$.
   a) Decrease γ.
   b) Find a solution reference $(\gamma, \boldsymbol{\beta}_\gamma, \mathrm{s})$.

   $\boldsymbol{\beta}_0$ is the solution.

By default, the initial solution reference is found by letting $\boldsymbol{\beta}_\gamma$ be the least squares solution. Alternatively, you can specify the initial solution reference with the INEST= option in the PROC QUANTREG statement. Then γ and s are chosen according to these initial values.

There are several approaches for determining a decreasing sequence of values of γ. The QUANTREG procedure uses a strategy by Madsen and Nielsen (1993). The computation that is uses is not significant compared to the Newton-Raphson step. You can control the ratio of consecutive decreasing values of γ by specifying the RRATIO= suboption in the ALGORITHM= option in the PROC QUANTREG statement. By default,

$$\mathrm{RRATIO} = \begin{cases} 0.1 & \text{if } n \geq 10{,}000 \text{ and } p \leq 20 \\ 0.9 & \text{if } \frac{p}{n} \geq 0.1 \text{ or } \{n \leq 5{,}000 \text{ and } p \geq 300\} \\ 0.5 & \text{otherwise} \end{cases}$$

For the L_1 and quantile regression, it turns out that the smoothing algorithm is very efficient and competitive, especially for a *fat* data set—namely, when $\frac{p}{n} > 0.05$ and $\mathbf{AA}'$ is dense. See Chen (2007) for a complete smoothing algorithm and details.

Fast Quantile Process Regression

The QUANTILE=FQPR option in the MODEL statement implements a fast quantile process regression (FQPR) method. This method can efficiently fit multiple quantile regression models by using the divide-and-conquer strategy proposed by Yao (2017).

The FQPR method begins by fitting a quantile regression model for a selected quantile level in a specified quantile-level grid of q-nodes . The quantile level, τ, is selected as the closest to 0.5 among all the quantile levels in the grid. Using this fit, FQPR defines two subsets of the data based on whether observed values y

are above or below their linear predictors $\mathbf{x}'\boldsymbol{\beta}(\tau)$ in this regression fit. Then FQPR proceeds to recursively perform separate quantile process regressions on the two subsets.

The successive quantile regression steps of FQPR are thus fit to smaller and smaller data sets. It is this sequence of reductions in problem size that provides the very significant reduction in computational cost that FQPR can achieve. In particular, FQPR can fit a quantile process regression model for q equally spaced quantiles in the time that it would approximately take to fit just $\log(q)$ quantile regression models to all the data.

By default, the QUANTILE=FQPR option uses the ALGORITHM=IPM option in the PROC QUANTREG statement to specify the efficient interior point algorithm for fitting the single-level models in the quantile process model. For more information about the ALGORITHM=IPM option, see the section "Efficient Interior Point Algorithm" on page 8578. You can also use the ALGORITHM=SMOOTH option in the PROC QUANTREG statement to specify a fast smoothing algorithm for fitting the single-level models in the quantile process model. The fast smoothing algorithm uses a smooth function different from the smoothing algorithm that is described in the section "Smoothing Algorithm" on page 8578. The fast algorithm smooth function is defined as

$$D_{\gamma,\tau}(\boldsymbol{\beta}) = \sum_{i=1}^{n} H_{\gamma,\tau}(r_i(\boldsymbol{\beta}))$$

where

$$H_{\gamma,\tau}(t) = \begin{cases} t(\tau - 1) & \text{if } t \le -\gamma \\ t(\tau - 0.5) + \frac{t^2}{4\gamma} + \frac{\gamma}{4} & \text{if } -\gamma \le t \le \gamma \\ t\tau & \text{if } t \ge \gamma \end{cases}$$

The QUANTILE=FQPR(USEALLOBS) option requests that the FQPR algorithm use all the observations for fitting each of the single-level models in the quantile process model. Because of possible crossing, the USEALLOBS suboption can output a quantile-process model slightly different from the fitted FQPR model that does not use the USEALLOBS suboption.

Confidence Interval

The QUANTREG procedure provides three methods to compute confidence intervals for the regression quantile parameter $\boldsymbol{\beta}(\tau)$: sparsity, rank, and resampling. The sparsity method is the most direct and the fastest, but it involves estimation of the sparsity function, which is not robust for data that are not independently and identically distributed. To deal with this problem, the QUANTREG procedure uses a local estimate of the sparsity function to compute a Huber sandwich estimate. The rank method, which computes confidence intervals by inverting the rank score test, does not suffer from this problem. However, the rank method uses the simplex algorithm and is computationally expensive with large data sets. The resampling method, which uses the bootstrap approach, addresses these problems, but at a computation cost.

Based on these properties, the QUANTREG uses a combination of the resampling and rank methods as the default. For data sets that have more than either 5,000 observations or more than 20 variables, the QUANTREG procedure uses the MCMB resampling method; otherwise it uses the rank method. You can request a particular method by using the CI= option in the PROC QUANTREG statement.

Sparsity

Consider the linear model

$$y_i = \mathbf{x}_i'\boldsymbol{\beta} + \epsilon_i$$

Assume that $\{\epsilon_i\}$, $i = 1, \ldots, n$, are iid with a distribution F and a density $f = F'$, where $f(F^{-1}(\tau)) > 0$ in a neighborhood of τ. Under some mild conditions,

$$\sqrt{n}(\hat{\boldsymbol{\beta}}(\tau) - \boldsymbol{\beta}(\tau)) \rightarrow N(0, \omega^2(\tau, F)\boldsymbol{\Omega}^{-1})$$

where $\omega^2(\tau, F) = \tau(1-\tau)/f^2(F^{-1}(\tau))$ and $\boldsymbol{\Omega} = \lim_{n\rightarrow\infty} n^{-1} \sum \mathbf{x}_i \mathbf{x}_i'$ (Koenker and Bassett 1982b).

This asymptotic distribution for the regression quantile $\hat{\boldsymbol{\beta}}(\tau)$ can be used to construct confidence intervals. However, the reciprocal of the density function,

$$s(\tau) = [f(F^{-1}(\tau))]^{-1}$$

which is called the *sparsity function*, must first be estimated.

Because

$$s(t) = \frac{d}{dt} F^{-1}(t)$$

$s(t)$ can be estimated by the difference quotient of the empirical quantile function—that is,

$$\hat{s}_n(t) = [\hat{F}_n^{-1}(t + h_n) - \hat{F}_n^{-1}(t - h_n)]/2h_n$$

where $\hat{F}_n$ is an estimate of F^{-1} and h_n is a bandwidth that tends to 0 as $n \rightarrow \infty$.

The QUANTREG procedure provides two bandwidth methods. The Bofinger bandwidth

$$h_n = n^{-1/5}(\frac{4.5s^2(t)}{(s^{(2)}(t))^2})^{1/5}$$

is an optimizer of mean squared error for standard density estimation. The Hall-Sheather bandwidth

$$h_n = n^{-1/3} z_\alpha^{2/3} (\frac{1.5s(t)}{s^{(2)}(t)})^{1/3}$$

is based on Edgeworth expansions for studentized quantiles, where $s^{(2)}(t)$ is the second derivative of $s(t)$ and z_α satisfies $\Phi(z_\alpha) = 1 - \alpha/2$ for the construction of $1 - \alpha$ confidence intervals. The following quantity is not sensitive to f and can be estimated by assuming f is Gaussian:

$$\frac{s(t)}{s^{(2)}(t)} = \frac{f^2}{2(f^{(1)}/f)^2 + [(f^{(1)}/f)^2 - f^{(2)}/f]}$$

F^{-1} can be estimated in either of the following ways:

- by the empirical quantile function of the residuals from the quantile regression fit,

  $$\hat{F}^{-1}(t) = r_{(i)}, \quad \text{for } t \in [(i-1)/n, i/n),$$

- by the empirical quantile function of regression proposed by Bassett and Koenker (1982),

$$\hat{F}^{-1}(t) = \bar{\mathbf{x}}'\hat{\boldsymbol{\beta}}(t)$$

The QUANTREG procedure interpolates the first empirical quantile function and produces the piecewise linear version:

$$\hat{F}^{-1}(t) = \begin{cases} r_{(1)} & \text{if } t \in [0, 0.5/n) \\ \lambda r_{(i+1)} + (1-\lambda) r_{(i)} & \text{if } t \in [(i-0.5)/n, (i+0.5)/n) \\ r_{(n)} & \text{if } t \in [(n-0.5)/n, 1] \end{cases}$$

$\hat{F}^{-1}$ is set to a constant if $t \pm h_n$ falls outside $[0, 1]$.

This estimator of the sparsity function is sensitive to the iid assumption. Alternately, Koenker and Machado (1999) consider the non-iid case. By assuming local linearity of the conditional quantile function $Q_{Y|\mathbf{x}}(\tau)$ in $\mathbf{x}$, they propose a local estimator of the density function by using the difference quotient. A Huber sandwich estimate of the covariance and standard error is computed and used to construct the confidence intervals. One difficulty with this method is the selection of the bandwidth when using the difference quotient. With a small sample size, either the Bofinger or the Hall-Sheather bandwidth tends to be too large to assure local linearity of the conditional quantile function. The QUANTREG procedure uses a heuristic bandwidth selection in these cases.

By default, the QUANTREG procedure computes non-iid confidence intervals. You can request iid confidence intervals by specifying the IID option in the PROC QUANTREG statement.

Inversion of Rank Tests

The classical theory of rank tests can be extended to test the hypothesis H_0: $\beta_2 = \eta$ in the linear regression model $\mathbf{y} = \mathbf{X}_1\beta_1 + \mathbf{X}_2\beta_2 + \boldsymbol{\epsilon}$. Here, $(\mathbf{X}_1, \mathbf{X}_2) = \mathbf{A}'$, where $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)'$ is the unknown error vector.

See Gutenbrunner and Jureckova (1992) for more details. By inverting this test, confidence intervals can be computed for the regression quantiles that correspond to β_2.

The rank score function $\hat{a}_n(t) = (\hat{a}_{n1}(t), \ldots, \hat{a}_{nn}(t))$ can be obtained by solving the dual problem:

$$\max_a \{(\mathbf{y} - \mathbf{X}_2\boldsymbol{\eta})'a | \mathbf{X}_1'a = (1-t)\mathbf{X}_1'\mathbf{e},\ a \in [0,1]^n\}$$

For a fixed quantile τ, integrating $\hat{a}_{ni}(t)$ with respect to the τ-quantile score function

$$\varphi_\tau(t) = \tau - I(t < \tau)$$

yields the τ-quantile scores

$$\hat{b}_{ni} = -\int_0^1 \varphi_\tau(t) d\hat{a}_{ni}(t) = \hat{a}_{ni}(\tau) - (1-\tau)$$

Under the null hypothesis H_0: $\beta_2 = \eta$,

$$S_n(\eta) = n^{-1/2}\mathbf{X}_2'\hat{b}_n(\eta) \to N(0, \tau(1-\tau)\boldsymbol{\Omega}_n)$$

for large n, where $\boldsymbol{\Omega}_n = n^{-1}\mathbf{X}_2'(\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1')\mathbf{X}_2$.

Let

$$T_n(\eta) = \frac{1}{\sqrt{\tau(1-\tau)}} S_n(\eta) \mathbf{\Omega}_n^{-1/2}$$

Then $T_n(\hat{\beta}_2(\tau)) = 0$ from the constraint $\mathbf{A}\hat{a} = (1-\tau)\mathbf{Ae}$ in the full model. In order to obtain confidence intervals for β_2, a critical value can be specified for T_n. The dual vector $\hat{a}_n(\eta)$ is a piecewise constant in η, and η can be altered without compromising the optimality of $\hat{a}_n(\eta)$ as long as the signs of the residuals in the primal quantile regression problem do not change. When η gets to such a boundary, the solution does change. But it can be restored by taking one simplex pivot. The process can continue in this way until $T_n(\eta)$ exceeds the specified critical value. Because $T_n(\eta)$ is piecewise constant, interpolation can be used to obtain the desired level of confidence interval (Koenker and d'Orey 1994).

Resampling

The bootstrap can be implemented to compute confidence intervals for regression quantile estimates. As in other regression applications, both the residual bootstrap and the *xy*-pair bootstrap can be used. The former assumes iid random errors and resamples from the residuals, whereas the latter resamples *xy* pairs and accommodates some forms of heteroscedasticity. Koenker (1994) considered a more interesting resampling mechanism, resampling directly from the full regression quantile process, which he called the Heqf bootstrap.

In contrast with these bootstrap methods, Parzen, Wei, and Ying (1994) observed that the following estimating equation for the τ regression quantile is a pivotal quantity for the τ quantile regression parameter β_τ:

$$S(\boldsymbol{\beta}) = n^{-1/2} \sum_{i=1}^{n} \mathbf{x}_i (\tau - I(y_i \le \mathbf{x}_i' \boldsymbol{\beta}))$$

In other words, the distribution of $\mathbf{S}(\boldsymbol{\beta})$ can be generated exactly by a random vector $\mathbf{U}$, which is a weighted sum of independent, re-centered Bernoulli variables. They further showed that for large n, the distribution of $\hat{\boldsymbol{\beta}}(\tau) - \beta_\tau$ can be approximated by the conditional distribution of $\hat{\boldsymbol{\beta}}_U - \hat{\boldsymbol{\beta}}_n(\tau)$, where $\hat{\boldsymbol{\beta}}_U$ solves an augmented quantile regression problem by using $n+1$ observations that have $\mathbf{x}_{n+1} = -n^{-1/2}u/\tau$ and y_{n+1} sufficiently large for a given realization of u. By exploiting the asymptotically pivotal role of the quantile regression "gradient condition," this approach also achieves some robustness to certain heteroscedasticity.

Although the bootstrap method by Parzen, Wei, and Ying (1994) is much simpler, it is too time-consuming for relatively large data sets, especially for high-dimensional data sets. The QUANTREG procedure implements a new, general resampling method developed by He and Hu (2002), which is called the Markov chain marginal bootstrap (MCMB). For quantile regression, the MCMB method has the advantage that it solves p one-dimensional equations instead of solving p-dimensional equations, as the previous bootstrap methods do. This greatly improves the feasibility of the resampling method in computing confidence intervals for regression quantiles.

Covariance-Correlation

You can specify the COVB and CORRB options in the MODEL statement to request covariance and correlation matrices for the estimated parameters.

The QUANTREG procedure provides two methods for computing the covariance and correlation matrices of the estimated parameters: an asymptotic method and a bootstrap method. Bootstrap covariance and correlation matrices are computed when resampling confidence intervals are computed. Asymptotic covariance and correlation matrices are computed when asymptotic confidence intervals are computed. The rank method for confidence intervals does not provide a covariance-correlation estimate.

Asymptotic Covariance-Correlation

This method corresponds to the sparsity method for the confidence intervals. For the sparsity function in the computation of the asymptotic covariance and correlation, the QUANTREG procedure provides both iid and non-iid estimates. By default, the QUANTREG procedure computes non-iid estimates.

Bootstrap Covariance-Correlation

This method corresponds to the resampling method for the confidence intervals. The Markov chain marginal bootstrap (MCMB) method is used.

Linear Test

Consider the linear model

$$y_i = \mathbf{x}_{1i}'\boldsymbol{\beta}_1 + \mathbf{x}_{2i}'\boldsymbol{\beta}_2 + \epsilon_i$$

where $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are p- and q-dimensional unknown parameters and $\{\epsilon_i\}$, $i = 1, \ldots, n$, are errors with unknown density function f_i. Let $\mathbf{x}_i' = (\mathbf{x}_{1i}', \mathbf{x}_{2i}')$, and let $\hat{\boldsymbol{\beta}}_1(\tau)$ and $\hat{\boldsymbol{\beta}}_2(\tau)$ be the parameter estimates for $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$, respectively at the τ quantile. The covariance matrix $\boldsymbol{\Omega}$ for the parameter estimates is partitioned correspondingly as $\boldsymbol{\Omega}_{ij}$ with $i = 1, 2$; $j = 1, 2$; and $\boldsymbol{\Omega}^{22} = (\boldsymbol{\Omega}_{22} - \boldsymbol{\Omega}_{21}\boldsymbol{\Omega}_{11}^{-1}\boldsymbol{\Omega}_{12})^{-1}$.

Testing Effects of Covariates

Three tests are available in the QUANTREG procedure for the linear null hypothesis $H_0 : \beta_2 = 0$ at the τ quantile:

- The Wald test statistic, which is based on the estimated coefficients for the unrestricted model, is given by

  $$T_W(\tau) = \hat{\boldsymbol{\beta}}_2'(\tau)\hat{\boldsymbol{\Sigma}}(\tau)^{-1}\hat{\boldsymbol{\beta}}_2(\tau)$$

  where $\hat{\boldsymbol{\Sigma}}(\tau)$ is an estimator of the covariance of $\hat{\boldsymbol{\beta}}_2(\tau)$. The QUANTREG procedure provides two estimators for the covariance, as described in the previous section. The estimator that is based on the asymptotic covariance is

$$\hat{\mathbf{\Sigma}}(\tau) = \frac{1}{n}\hat{\omega}(\tau)^2\mathbf{\Omega}^{22}$$

where $\hat{\omega}(\tau) = \sqrt{\tau(1-\tau)}\hat{s}(\tau)$ and $\hat{s}(\tau)$ is the estimated sparsity function. The estimator that is based on the bootstrap covariance is the empirical covariance of the MCMB samples.

- The likelihood ratio test is based on the difference between the objective function values in the restricted and unrestricted models. Let $D_0(\tau) = \sum \rho_\tau(y_i - \mathbf{x}_i'\hat{\boldsymbol{\beta}}(\tau))$, and let $D_1(\tau) = \sum \rho_\tau(y_i - \mathbf{x}_{1i}'\hat{\boldsymbol{\beta}}_1(\tau))$. Set

$$T_{\mathrm{LR}}(\tau) = 2(\tau(1-\tau)\hat{s}(\tau))^{-1}(D_1(\tau) - D_0(\tau))$$

where $\hat{s}(\tau)$ is the estimated sparsity function.

- The rank test statistic is given by

$$T_R(\tau) = \mathbf{S}_n'\mathbf{M}_n^{-1}\mathbf{S}_n/A^2(\varphi)$$

where

$$\mathbf{S}_n = n^{-1/2}(\mathbf{X}_2 - \hat{\mathbf{X}}_2)'\hat{\mathbf{b}}_n$$

$$\mathbf{\Psi} = \mathrm{diag}(f_i(Q_{y_i|\mathbf{x}_{1i},\mathbf{x}_{2i}}(\tau)))$$

$$\hat{\mathbf{X}}_2 = \mathbf{X}_1(\mathbf{X}_1'\mathbf{\Psi}\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2$$

$$\mathbf{M}_n = (\mathbf{X}_2 - \hat{\mathbf{X}}_2)(\mathbf{X}_2 - \hat{\mathbf{X}}_2)'/n$$

$$\hat{\mathbf{b}}_{ni} = \int_0^1 \hat{\mathbf{a}}_{ni}(t)d\varphi(t)$$

$$\hat{\mathbf{a}}(t) = \max_a\{\mathbf{y}'\mathbf{a}|\mathbf{X}_1'\mathbf{a} = (1-t)\mathbf{X}_1'\mathbf{e},\ \mathbf{a} \in [0,1]^n\}$$

$$A^2(\varphi) = \int_0^1 (\varphi(t) - \bar{\varphi}(t))^2 dt$$

$$\bar{\varphi}(t) = \int_0^1 \varphi(t)dt$$

and $\varphi(t)$ is one of the following score functions:

  - Wilcoxon scores: $\phi(t) = t - 1/2$
  - normal scores: $\phi(t) = \Phi^{-1}(t)$, where Φ is the normal distribution function
  - sign scores: $\phi(t) = 1/2\mathrm{sign}(t - 1/2)$
  - tau scores: $\phi_\tau(t) = \tau - I(t < \tau)$.

The rank test statistic $T_R(\tau)$, unlike Wald tests or likelihood ratio tests, requires no estimation of the nuisance parameter f_i under iid error models (Gutenbrunner et al. 1993).

Koenker and Machado (1999) prove that the three test statistics ($T_W(\tau)$, $T_{\mathrm{LR}}(\tau)$, and $T_R(\tau)$) are asymptotically equivalent and that their distributions converge to χ^2_q under the null hypothesis, where q is the dimension of β_2.

Testing for Heteroscedasticity

After you obtain the parameter estimates for several quantiles specified in the MODEL statement, you can test whether there are significant differences for the estimates for the same covariates across the quantiles. For example, if you want to test whether the parameters β_2 are the same across quantiles, the null hypothesis H_0 can be written as $\beta_2(\tau_1) = \ldots = \beta_2(\tau_k)$, where $\tau_j, j = 1, \ldots, k$, are the quantiles specified in the MODEL statement. See Koenker and Bassett (1982a) for details.

Estimating Probability Functions by Using the CONDDIST Statement

Because the cumulative distribution function (CDF) is the inverse of the quantile function, you can estimate the CDF and other CDF-based statistics by inverting the relevant quantile function estimates. The CONDDIST statement in the QUANTREG procedure performs this type of analysis by estimating the conditional and marginal probability functions for the response random variable $\mathbf{Y}$. These probability functions include the following:

- $F_{\mathbf{Y}|\mathbf{x}}(t)$ for the conditional CDF and $P_{\mathbf{Y}|\mathbf{x}}(t)$ for the conditional probability density function (PDF) that can be estimated using the model
- $F_{\mathbf{Y}}(t)$ for observable marginal CDF and $P_{\mathbf{Y}}(t)$ for the observable marginal PDF that can be estimated from the observed response values without using the model
- $F_{\mathbf{P}}(t)$ for the predictable marginal CDF and $P_{\mathbf{P}}(t)$ for the counterfactual marginal PDF that can be estimated from the quantile process predictions of the response variable and with the covariates $\mathbf{x}$ integrated out

The QUANTREG procedure estimates the conditional quantile function, $Q_{\mathbf{Y}|\mathbf{x}}(\tau) = \mathbf{x}'\boldsymbol{\beta}(\tau)$, by using the fast quantile process regression (FQPR) algorithm. If you specify the QUANTILE= FQPR option in the MODEL statement, the CONDDIST statement uses the FQPR parameter estimates $\hat{\boldsymbol{\beta}}(\tau)$ to estimate $\boldsymbol{\beta}(\tau)$. For more information about the FQPR option, see the section "Fast Quantile Process Regression" on page 8580. If the QUANTILE=FQPR option is not used in the MODEL statement, the first CONDDIST statement uses the FQPR algorithm to compute $\hat{\boldsymbol{\beta}}(\tau)$ for all the CONDDIST statements. This $\hat{\boldsymbol{\beta}}(\tau)$ is equal to the $\hat{\boldsymbol{\beta}}(\tau)$ that is output from the QUANTILE=FQPR option when you use the default FQPR suboption values.

For purposes of conceptual and computational simplicity, the CONDDIST statement estimates the quantile functions $Q(\tau)$ on a quantile-level grid $\{\tau_j = \tau_1 + (j-1)\tau_s : j = 1, \ldots, q\}$, where τ_1 is the lower end of the grid and τ_s is the step length of the grid. You can specify τ_1 and q by respectively using the L=*value* and N=*n* suboptions of the QUANTILE= FQPR option in the MODEL statement. You can also specify the upper end of the grid τ_q by using the U=*value* suboption of the QUANTILE=FQPR option in the MODEL statement, such that

$$\tau_s = \frac{\tau_q - \tau_1}{q - 1}$$

By default, the size of the grid q is the smaller number between 100 and half the number of training observations in the DATA= data set. And by default, $\tau_1 = 0.5/q$, $\tau_q = 1 - 0.5/q$, and $\tau_s = 1/q$.

The estimated quantile function $\left\{\widehat{Q}(\tau_1), \ldots, \widehat{Q}(\tau_q)\right\}$ is also called a CDF sample for $F(t)$.

Counterfactual Distributions for the TESTDATA= Data Set

The QUANTREG procedure computes the quantile function estimates $\widehat{Q}_{\mathbf{Y}|\mathbf{x}}(\tau) = \mathbf{x}'\hat{\boldsymbol{\beta}}(\tau)$ on the DATA= data set in the PROC QUANTREG statement. To test this $\widehat{Q}_{\mathbf{Y}|\mathbf{x}}(\tau)$ for more observations, you can specify a separate data set by using the TESTDATA= data set option in the CONDDIST statement. Without assuming that the TESTDATA= data set and the DATA= data set share the same conditional distribution for $\mathbf{Y}|\mathbf{X}$, the CONDDIST statement estimates counterfactual probability functions for the response variable $\mathbf{Y}$ of the TESTDATA= data set that impose the quantile function $Q_{\mathbf{Y}|\mathbf{x}}(\tau)$ of the DATA= data set on the TESTDATA= data set.

Conditional Cumulative Distribution Functions

The CONDDIST statement estimates $F_{\mathbf{Y}|\mathbf{x}}(t)$ by inverting the estimated quantile function on the quantile-level grid $\left\{\widehat{Q}_{\mathbf{Y}|\mathbf{x}}(\tau_1), \ldots, \widehat{Q}_{\mathbf{Y}|\mathbf{x}}(\tau_q)\right\}$, such that the estimated $F_{\mathbf{Y}|\mathbf{x}}(t)$ satisfies

$$\widehat{F}_{\mathbf{Y}|\mathbf{x}}\left(\widehat{Q}_{\mathbf{Y}|\mathbf{x}}(\tau_j)\right) = \tau_i \text{ for } j = 1, \ldots, q$$

By definition, a quantile function $Q(\tau)$ must be nondecreasing, so that $Q(\tau_1) \leq Q(\tau_2)$ for any $0 \leq \tau_1 < \tau_2 \leq 1$. However, quantile regression estimates could result in crossed quantile predictions. In other words, it is possible that $\mathbf{x}'\hat{\boldsymbol{\beta}}(\tau_1) > \mathbf{x}'\hat{\boldsymbol{\beta}}(\tau_2)$ for some $\mathbf{x}$ and some $0 \leq \tau_1 < \tau_2 \leq 1$. This predicament is called crossing. To avoid crossing, the CONDDIST statement defines $\widehat{Q}_{\mathbf{Y}|\mathbf{x}}(\tau_j)$ as the jth-smallest prediction among $\left\{\mathbf{x}'\hat{\boldsymbol{\beta}}(\tau_1), \ldots, \mathbf{x}'\hat{\boldsymbol{\beta}}(\tau_q)\right\}$. Therefore, it is possible that $\widehat{Q}_{\mathbf{Y}|\mathbf{x}}(\tau_j) \neq \mathbf{x}'\hat{\boldsymbol{\beta}}(\tau_j)$ for some $\mathbf{x}$ and τ_j.

For an individual observation, the CONDDIST statement assigns the type "Fit for Obs" to this conditional CDF sample and labels this sample by using the observation ID value (if available) or the observation index.

Conditional Cumulative Distribution Functions at Average

You can also request the conditional CDF sample at average $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ for $F_{\mathbf{Y}|\bar{\mathbf{x}}}(t)$ by using the SHOWAVG option for the training average or the TESTDATA(SHOWAVG)= option for the test average (or both).

The CONDDIST statement assigns the type "Fit at Average" to these conditional CDF samples. For training data, the CONDDIST statement labels this CDF sample as "TrainAvg" and assigns the average of the training response values $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ as its response value. For test data, the CONDDIST statement labels this CDF sample as "TestAvg" and assigns the average of the test response values as its response value.

Observed Marginal Cumulative Distribution Functions

The CONDDIST statement estimates the observable marginal $F_{\mathbf{Y}}(t)$ by inverting the empirical quantile function $\widehat{Q}_{\mathbf{Y}}(\tau)$ of the response variable.

Let $y_{(1)} \leq y_{(2)} \leq \cdots \leq y_{(n)}$ denote the sorted response values for either the training data that you can specify by using the DATA= data set in the PROC QUANTREG statement or the test data that you can specify by using the TESTDATA= data set in the CONDDIST statement. If you assign the quantile level $(i - 0.5)/n$ to the response value $y_{(i)}$, then $\widehat{Q}_{\mathbf{Y}}(\tau)$ is defined as

$$\widehat{Q}_{\mathbf{Y}}(\tau) = \begin{cases} y_{(1)} & \text{if } \tau \in [0, 0.5/n) \\ \lambda y_{(i+1)} + (1-\lambda) y_{(i)} & \text{if } \tau \in [(i-0.5)/n, (i+0.5)/n) \text{ for } \lambda = n\tau - (i - 0.5) \\ y_{(n)} & \text{if } \tau \in [(n-0.5)/n, 1] \end{cases}$$

To be consistent with the quantile-level grid that is used for estimating the conditional probability functions, the CONDDIST statement estimates $Q_{\mathbf{Y}}(\tau)$ by using the CDF sample $\left\{\widehat{Q}_Y(\tau_1), \ldots, \widehat{Q}_Y(\tau_q)\right\}$.

The CONDDIST statement assigns the type "Observed" to these observed marginal CDF samples. For training data, the CONDDIST statement labels this CDF sample as "TrainObs" and assigns the average of the training response values as its response value. For test data, the CONDDIST statement labels this CDF sample as "testObs" and assigns the average of the test response values as its response value.

Fitted Marginal Cumulative Distribution Functions

Let $F_{\mathbf{Y}|\mathbf{X}}$ denote the distribution of $\mathbf{Y}|\mathbf{X}$ for the DATA= data set, and let $F^*_{\mathbf{X}}$ denote the marginal distribution of the explanatory covariates vector $\mathbf{X}$ for the TESTDATA= data set. Then the counterfactual marginal distribution of $\mathbf{Y}$ for the TESTDATA= data set is defined as

$$F^*_{\mathbf{P}}(t) = \int F_{\mathbf{Y}|\mathbf{X}=\mathbf{x}}(t) dF^*_{\mathbf{X}}(\mathbf{x})$$

The CONDDIST statement estimates $F^*_{\mathbf{P}}(t)$ by inverting the quantile function of the quantile process predictions for the response variable.

Let $\left\{p_{ij} = \mathbf{x}_i'\hat{\boldsymbol{\beta}}(\tau_j) : i = 1, \ldots, n \text{ and } j = 1, \ldots, q\right\}$ denote the quantile predictions for all the observations of the TESTDATA= data set. And let $p_{(k)}$ denote the kth-smallest value in $\left\{p_{ij}\right\}$. The CONDDIST statement estimates $Q^*_{\mathbf{P}}(\tau)$ by defining

$$\widehat{Q}^*_{\mathbf{P}}(\tau_j) = \begin{cases} p_{(jn-0.5n+0.5)} & \text{if } n \text{ is an odd number} \\ 0.5p_{(jn-0.5n)} + 0.5p_{(jn-0.5n+1)} & \text{if } n \text{ is an even number} \end{cases}$$

where $\tau_j = \tau_1 + (j-1)\tau_s$ for $j = 1, \ldots, q$.

The CONDDIST statement assigns the type "Fit and Pooled" and the label "TestFit" to this fitted marginal CDF sample, and assigns the average of the test response values as its response value.

Recall that the marginal CDF is also available for the TESTDATA= data set. For clarity, let $F^*_{\mathbf{Y}}(t)$ denote the observable marginal CDF for the TESTDATA= data set. Then, by comparing the fitted marginal CDF sample $\widehat{F}^*_{\mathbf{P}}(t)$ with the observed marginal CDF sample $\widehat{F}^*_{\mathbf{Y}}(t)$, you can test whether the TESTDATA= data set follows the same model that is built on the DATA= data set. The CONDDIST statement supports the Mann-Whitney U test for this purpose when you use the MWU option.

Mann-Whitney U Test

The Mann-Whitney U test (also called the Wilcoxon rank-sum test) is a nonparametric two-sample test. Let n_1 denote the size of the first sample and n_2 denote the size of the second sample. By merging the two samples into one ordered set $\{a_j : j = 1, \ldots, n = n_1 + n_2\}$, the statistic of the Mann-Whitney U test is defined as

$$U = \sum_{j=1}^{n} c_j R_j \text{ with } c_j = \begin{cases} 1 & \text{if } a_j \text{ belongs to the first sample} \\ 0 & \text{if } a_j \text{ belongs to the second sample} \end{cases}$$

where R_j is the rank of a_j. Under some regularity conditions, U asymptotically follows a normal distribution whose expectation equals

$$\mathrm{E}(U) = \frac{n_1}{n} \sum_{j=1}^{n} R_j = \frac{n_1(n+1)}{2}$$

and whose variance equals

$$\mathrm{Var}(U) = \frac{n_1 n_2 (n+1)}{12}$$

You can perform the Mann-Whitney U test against the observed marginal CDF sample of the response variable for the following data sets:

- the DATA= data set in the PROC QUANTREG statement (by using the MWU option in the CONDDIST statement)
- the TESTDATA= data set in the CONDDIST statement (by using the TESTDATA(MWU)= data set option in the CONDDIST statement)

The null and alternative hypotheses of the Mann-Whitney U test for the CONDDIST statement are, respectively,

$$\begin{aligned} H_0 :&\quad P(\mathbf{Y}1 > \mathbf{Y}2) = P(\mathbf{Y}1 < \mathbf{Y}2) \\ H_1 :&\quad P(\mathbf{Y}1 > \mathbf{Y}2) > P(\mathbf{Y}1 < \mathbf{Y}2) \text{ or } P(\mathbf{Y}1 > \mathbf{Y}2) < P(\mathbf{Y}1 < \mathbf{Y}2) \end{aligned}$$

where $\mathbf{Y}1$ denotes the response random variable for a CDF under the hypothesis test such as $F_{\mathbf{Y}}$ or $F_{\mathbf{Y}|\mathbf{x}}$, and $\mathbf{Y}2$ denotes the random variable for the observable marginal CDF for either the DATA= data set or the TESTDATA= data set.

You can specify the size of the CDF samples by using the MWU(SAMPSIZE=NOBS | NQ) option or the TESTDATA(MWU(SAMPSIZE=NOBS | NQ)) option in the CONDDIST statement. The TESTDATA(MWU(SAMPSIZE=NOBS | NQ)) option overrides the MWU(SAMPSIZE=NOBS | NQ) option.

The SAMPSIZE=NOBS suboption requests that the size of the TrainObs CDF sample, the TestFit CDF sample, and all the predicted observationwise CDF samples be equal to the number of training observations in the DATA= data set in the PROC QUANTREG statement; and it requests that the size of the TestObs CDF sample be equal to the number of testing observations in the TESTDATA= data set in the CONDDIST statement. The SAMPSIZE=NOBS suboption is appropriate if the size of the quantile-level grid is larger than the number of training observations in the DATA= data set in the PROC QUANTREG statement. Otherwise, the Mann-Whitney U test could output an imprecise p-value because the fitted CDF samples might not sufficiently represent the CDF information that the original training data provide.

The SAMPSIZE=NQ suboption requests that the size of the CDF samples for the MWU tests be equal to the size of the quantile-level grid. The SAMPSIZE=NQ suboption is appropriate if the size of the quantile-level grid is smaller than both the number of training observations in the DATA= data set in the PROC QUANTREG statement and the number of testing observations in the TESTDATA= data set in the CONDDIST statement. Otherwise, the Mann-Whitney U test could output a smaller p-value and incorrectly reject H_0.

Marginal Distribution Analysis Using the Bootstrap Resampling Method

You can specify the MCDF option in the CONDDIST statement to perform the marginal distributions analysis by using a weighted bootstrap resampling method (Praestgaard and Wellner 1993). The bootstrap method iterates the following steps:

1. Assign a probability weight p_i to each of the training observations $(y_i, \mathbf{x}_i)$ in the DATA= data set in the QUANTREG statement. If you specify weights w_i by using the WEIGHT statement, p_i is generated from the Gamma$(w_i, 1)$ distribution and normalized by using the factor $\frac{\sum w_i}{\sum p_i}$, such that the finally assigned p_i satisfies $\sum p_i = \sum w_i$. Otherwise, p_i is generated from the Exponential(1) distribution and normalized by using $\frac{n}{\sum p_i}$, such that $\sum p_i = n$, where n is the total number of training observations.

2. Compute the following statistics by using the p_i-weighted training observations:

   a) Quantile process regression model: $\hat{\boldsymbol{\beta}}(\tau)$ by using the fast quantile process regression algorithm. You can use the FQPR(N=*n*) option in the MODEL statement to specify the quantile-level grid for your quantile process model.

   b) The TrainObs marginal CDF sample for the p_i-weighted response variable. Let $y_{(1)} \le y_{(2)} \le \cdots \le y_{(n)}$ denote the sorted response values for the training data; let $p_{(1)}, p_{(2)}, \ldots, p_{(n)}$ denote their associated normalized bootstrap weights; and define $p_{(0)} = 0$. Then the following formula computes a size-m TrainObs marginal CDF sample:

   $$\left\{ \widehat{Q}_{\mathbf{Y}}(\tau) = y_{(j)} \text{ if } \sum_{k=0}^{j-1} p_{(k)} < n\tau \le \sum_{k=0}^{j} p_{(k)} : \tau = \frac{i - 0.5}{m} \text{ for } i = 1, \ldots, m \right\}$$

   For the estimates in the marginal distribution comparison table, the size of the TrainObs marginal CDF sample equals the number of training observations. For the marginal CDF plot, the size of the TrainObs marginal CDF sample equals the number of quantile levels in the fast quantile process regression algorithm.

If you specify the TESTDATA= data set in the CONDDIST statement, the weighted bootstrap method also iterates the following steps:

3. Assign a probability weight p_j to each of the testing observations $(y_j, \mathbf{x}_j)$ by using the same method that is described in Step 1 for the TESTDATA= data set.

4. Compute quantile predictions $\mathbf{x}_j' \hat{\boldsymbol{\beta}}(\tau)$ for all the testing observations by using the quantile process regression model fitted in Step 2a.

5. Compute the following statistics:

   a) The TestObs marginal CDF sample for the p_j-weighted response variable. This marginal CDF sample is computed by using the same formula as in Step 2b but on the sorted testing response values.

b) The TestFit marginal CDF sample for the pooled quantile predictions from Step 5. This marginal CDF sample is computed by using the same formula as in Step 2b but on the sorted pooled quantile predictions, where the weight of each quantile prediction equals the weight of its associated testing observation. You can specify the QRNG= suboption in the MCDF option to specify the limits of the quantile predictions. By default, specifying QRNG=BOTH limits the quantile predictions to be within the range of observed response values that combines the training data and the testing data. If a quantile prediction is smaller than the minimum value of all the training and testing response values, then the quantile prediction would be reset to the minimum value. Similarly, if a quantile prediction is larger than the maximum value of all the training and testing response values, then the quantile prediction is reset to the maximum value.

c) The CDF comparison scores, including the MWU test, mean difference, and median difference.

For the estimates in the marginal distribution comparison table, the size of the TestObs and TestFit marginal CDF samples equals the number of testing observations. For the marginal CDF plot, the size of the TestObs and TestFit marginal CDF samples equals the number of quantile levels in the fast quantile process regression algorithm.

The NREP=*n* suboption in the MCDF option specifies the total number of bootstrap iterations. Let $\hat{\theta}_s$ denote the statistic that is computed in the *s*th bootstrap iteration for a parameter θ. This $\hat{\theta}_s$ could be a point in an empirical quantile function $\hat{Q}_Y(t)$, a MWU test score, or any other statistics from Step 2 or Step 6. Accordingly, $\sum \hat{\theta}_s$ and $\sum \hat{\theta}_s^2$, respectively, represent the total cumulative sum and the total cumulative sum of squares of the relevant statistic for all the bootstrap iterations that are cumulated in Step 3 and Step 7. Then the formulas in Table 103.9 calculate the final bootstrap estimate and its standard error, Z score, p-value, and confidence limits for each of the relevant statistics.

Table 103.9 Summarization Formulas

Statistic	Formula
Bootstrap estimate:	$\hat{\theta} = \sum \hat{\theta}_s / n$
Standard error:	$\mathrm{SE} = \sqrt{\left(\sum \hat{\theta}_s^2 / n\right) - \hat{\theta}^2}$
Z score:	$\hat{\theta}/\mathrm{SE}$
Two-sided p-value:	$\begin{cases} 2\Phi(Z) & \text{if } Z < 0 \\ 2\Phi(-Z) & \text{if } Z \geq 0 \end{cases}$
Confidence limits:	$\left(\hat{\theta} + \mathrm{SE}\Phi^{-1}(\alpha/2), \hat{\theta} - \mathrm{SE}\Phi^{-1}(\alpha/2)\right)$

In the marginal CDF plot, the confidence limits of a CDF sample are the inverse functions of the confidence limits of its associated empirical quantile function.

Regression Quantile Level and Sample Quantile Level

Given a response value y and a CDF sample for $F(t)$: $\left\{\widehat{Q}(\tau_1), \ldots, \widehat{Q}(\tau_q)\right\}$, the CONDDIST statement estimates the quantile level of y on $F(t)$ by using the following linear interpolation method:

$$\hat{\tau}_y \begin{cases} < \tau_1 & \text{if } y < \widehat{Q}(\tau_1) \\ = \lambda\tau_{i+1} + (1-\lambda)\tau_i & \text{if } \widehat{Q}(\tau_i) \le y \le \widehat{Q}(\tau_{i+1}) \text{ for } \lambda = \left(y - \widehat{Q}(\tau_i)\right) \Big/ \left(\widehat{Q}(\tau_{i+1}) - \widehat{Q}(\tau_i)\right) \\ > \tau_q & \text{if } y > \widehat{Q}(\tau_q) \end{cases}$$

Here $\hat{\tau}_y$ is defined as a regression quantile level if the CDF sample is conditional for $F_{\mathbf{Y}|\mathbf{x}}(t)$, or a sample quantile level if the CDF sample is observed and marginal for $F_{\mathbf{Y}}(t)$.

The PLOT=PPPLOT option in the CONDDIST statement creates the scatter plot for the regression quantile levels versus the sample quantile levels.

Probability Density Functions

The CONDDIST statement estimates the PDF by applying the kernel density estimator to the estimated CDF in the quantile-level grid. An estimated CDF for the CONDDIST statement is in the form of $\left\{\widehat{F}\left(\widehat{Q}(\tau_j)\right) = \tau_j : \tau_j = \tau_1 + (j-1)\tau_s, j = 1, \ldots, q\right\}$. The lower end of the estimated PDF is limited by the quantile level $\max(0, \tau_1 - 0.5\tau_s)$, and the upper end of the estimated PDF is limited by the quantile level $\min(\tau_q + 0.5\tau_s, 1)$.

The general form of the kernel density estimator is

$$\hat{f}_\lambda(t) = \frac{1}{q\lambda l} \sum_{j=1}^{q} K_0\left(\frac{t - t_j}{\lambda}\right)$$

where

$K_0(\cdot)$ is the kernel function

λ is the bandwidth

q is the sample size

t_j is the ith value $\widehat{F}\left(\widehat{Q}(\tau_j)\right)$

$l = q\tau_s$ is the length of the range of the quantile-level grid

The KDE option provides three kernel functions (K_0): normal, quadratic, and triangular. You can specify the function by using the K= *kernel-option* in parentheses after the KDE option. The values of the K= option are NORMAL, QUADRATIC, and TRIANGULAR. By default, a normal kernel is used. The formulas for the kernel functions are as follows:

Normal	$K_0(t) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}t^2\right)$	for $-\infty < t < \infty$
Quadratic	$K_0(t) = \frac{3}{4}\left(1 - t^2\right)$	for $\lvert t\rvert \le 1$
Triangular	$K_0(t) = 1 - \lvert t\rvert$	for $\lvert t\rvert \le 1$

The value of λ, referred to as the bandwidth parameter, determines the degree of smoothness in the estimated probability density function. You specify λ indirectly when you specify a standardized bandwidth c by using the C=*value* suboption. Let Q denote the interquartile range and q denote the sample size; then c is related to λ by the formula

$$\lambda = cQq^{-\frac{1}{5}}$$

For a specific kernel function, the discrepancy between the density estimator $\hat{f}_\lambda(x)$ and the true density $f(x)$ is measured by the mean integrated square error (MISE):

$$\mathrm{MISE}(\lambda) = \int_x \left\{\mathrm{E}\left(\hat{f}_\lambda(x)\right) - f(x)\right\}^2 dx + \int_x \mathrm{Var}\left(\hat{f}_\lambda(x)\right) dx$$

MISE is the sum of the integrated squared bias and the variance. An approximate mean integrated square error (AMISE) is defined as

$$\mathrm{AMISE}(\lambda) = \frac{1}{4}\lambda^4 \left(\int_t t^2 K(t)dt\right)^2 \int_x \left(f''(x)\right)^2 dx + \frac{1}{q\lambda}\int_t K(t)^2 dt$$

You can derive a bandwidth that minimizes AMISE by treating $f(x)$ as the normal density whose parameters μ and σ are estimated by the sample mean and standard deviation, respectively. If you do not specify a bandwidth parameter or if you specify C=MISE, the bandwidth that minimizes AMISE is used. The value of AMISE can be used to compare different density estimates. You can also specify C=SJPI to select the bandwidth by using the Sheather-Jones plug-in method (Jones, Marron, and Sheather 1996).

The general kernel density estimates assume that the domain of the density to be estimated can take all values on a real line. However, sometimes the domain of a density is an interval that is bounded on one or both sides. For example, if a variable Y is a measurement of only positive values, then the kernel density curve should be bounded so that it is zero for negative Y values. You can use the LOWER= and UPPER= *kde-options* in the PDF= option in the CONDDIST statement to specify the bounds.

Leverage Point and Outlier Detection

The QUANTREG procedure uses robust multivariate location and scale estimates for leverage-point detection.

Mahalanobis distance is defined as

$$\mathrm{MD}(\mathbf{x}_i) = [(\mathbf{x}_i - \bar{\mathbf{x}})'\bar{\mathbf{C}}(\mathbf{A})^{-1}(\mathbf{x}_i - \bar{\mathbf{x}})]^{1/2}$$

where $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$ and $\bar{\mathbf{C}}(\mathbf{A}) = \frac{1}{n-1}\sum_{i=1}^n (\mathbf{x}_i - \bar{\mathbf{x}})'(\mathbf{x}_i - \bar{\mathbf{x}})$ are the empirical multivariate location and scale, respectively. Here, $\mathbf{x}_i = (\mathbf{x}_{i1}, \ldots, \mathbf{x}_{i(p-1)})'$ does not include the intercept variable. The relationship between the Mahalanobis distance $\mathrm{MD}(\mathbf{x}_i)$ and the matrix $\mathbf{H} = (h_{ij}) = \mathbf{A}'(\mathbf{A}\mathbf{A}')^{-1}\mathbf{A}$ is

$$h_{ii} = \frac{1}{n-1}\mathrm{MD}_i^2 + \frac{1}{n}$$

Robust distance is defined as

$$\mathrm{RD}(\mathbf{x}_i) = [(\mathbf{x}_i - \mathbf{T}(\mathbf{A}))'\mathbf{C}(\mathbf{A})^{-1}(\mathbf{x}_i - \mathbf{T}(\mathbf{A}))]^{1/2}$$

where $\mathbf{T}(\mathbf{A})$ and $\mathbf{C}(\mathbf{A})$ are robust multivariate location and scale estimates that are computed according to the minimum covariance determinant (MCD) method of Rousseeuw and Van Driessen (1999).

These distances are used to detect leverage points. You can use the LEVERAGE and DIAGNOSTICS options in the MODEL statement to request leverage-point and outlier diagnostics, respectively. Two new variables, Leverage and Outlier, respectively, are created and saved in an output data set that is specified in the OUTPUT statement.

Let $C(p) = \sqrt{\chi^2_{p;1-\alpha}}$ be the cutoff value. The variable LEVERAGE is defined as

$$\text{LEVERAGE} = \begin{cases} 0 & \text{if } \mathrm{RD}(\mathbf{x}_i) \le C(p) \\ 1 & \text{otherwise} \end{cases}$$

You can specify a cutoff value in the LEVERAGE option in the MODEL statement.

Residuals $r_i, i = 1, \ldots, n$, that are based on quantile regression estimates are used to detect vertical outliers. The variable OUTLIER is defined as

$$\text{OUTLIER} = \begin{cases} 0 & \text{if } |r_i| \le k\sigma \\ 1 & \text{otherwise} \end{cases}$$

You can specify the multiplier k of the cutoff value in the CUTOFF= option in the MODEL statement. You can specify the scale σ in the SCALE= option in the MODEL statement. By default, $k = 3$ and the scale σ is computed as the corrected median of the absolute residuals:

$$\sigma = \text{median}\{|r_i|/\beta_0, i = 1, \ldots, n\}$$

where $\beta_0 = \Phi^{-1}(0.75)$ is an adjustment constant for consistency when the normal distribution is used.

An ODS table called DIAGNOSTICS contains the Leverage and Outlier variables.

INEST= Data Set

The INEST= data set specifies initial estimates for all the parameters in the model. The INEST= data set must contain the intercept variable (named Intercept) and all independent variables in the MODEL statement.

If BY processing is used, the INEST= data set should also include the BY variables, and there must be at least one observation for each BY group. If there is more than one observation in one BY group, the first one read is used for that BY group.

If the INEST= data set also contains the _TYPE_ variable, only observations with the _TYPE_ value `'PARMS'` are used as starting values.

You can specify starting values for the interior point algorithm or the smoothing algorithm in the INEST= data set. The INEST= data set has the same structure as the OUTEST= data set, but it is not required to have all the variables or observations that appear in the OUTEST= data set. One simple use of the INEST= option is passing the previous OUTEST= data set directly to the next model as an INEST= data set, assuming that the two models have the same parameterization. If you specify more than one quantile in the MODEL statement, the same initial values are used for all quantiles.

OUTEST= Data Set

The OUTEST= data set contains parameter estimates for the specified model with all quantiles. A set of observations is created for each quantile specified. You can also specify a label in the MODEL statement to distinguish between the estimates for different models that are used by the QUANTREG procedure.

If the QUANTREG procedure does not produce valid solutions, the parameter estimates are set to missing in the OUTEST data set.

If this data set is created, it contains all the variables that are specified in the MODEL statement and the BY statement. Each observation consists of parameter values for a specified quantile, and the dependent variable has the value –1.

The following variables are also added to the data set:

MODEL	a character variable of length 8 that contains the label of the MODEL statement, if present. Otherwise, the variable's value is blank.
ALGORITHM	a character variable of length 8 that contains the name of the algorithm that is used for computing the parameter estimates, either SIMPLEX, INTERIOR, or SMOOTH.
TYPE	a character variable of length 8 that contains the type of the observation. This variable is fixed as PARMS to indicate that the observation includes parameter estimates.
STATUS	a character variable of length 12 that contains the status of model fitting (either NORMAL, NOUNIQUE, or NOVALID).
Intercept	a numeric variable that contains the intercept parameter estimates.
QUANTILE	a numeric variable that contains the specified quantile levels.

Any specified BY variables are also added to the OUTEST= data set.

Computational Resources

The various algorithms need different amounts of memory for working space. Let p be the number of parameters that are estimated, n be the number of observations that are used in the model estimation, and s be the size (in bytes) of the double data type.

For the simplex algorithm, the minimum working space (in bytes) that is needed is

$$(2np + 6n + 10p)s$$

For the interior point algorithm, the minimum working space (in bytes) that is needed is

$$(np + p^2 + 13n + 4p)s$$

For the smoothing algorithm, the minimum working space (in bytes) that is needed is

$$(np + p^2 + 6n + 4p)s$$

For the last two algorithms, if you want to use preprocessing, the following additional amount of working space (in bytes) is needed:

$$(np + 6n + 2p)s$$

If sufficient space is available, the input data set is kept in memory. Otherwise, the input data set is reread as necessary, and the execution time of the procedure increases substantially.

ODS Table Names

The QUANTREG procedure assigns a name to each table that it creates. You can specify these names when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the Table 103.10.

Table 103.10 ODS Tables Produced in PROC QUANTREG

ODS Table Name	Description	Statement	Option
AvgParameterEst	Average parameter estimates	MODEL	FQPR
AvgObjFunction	Average objective function	MODEL	FQPR
BootProcessEst	Samplewise bootstrap quantile process estimates	CONDDIST	MCDF
ClassLevels	Classification variable levels	CLASS	Default
CmpMarginalDist	Comparisons of marginal distributions	CONDDIST	MCDF
CmpTestObs	Distribution comparisons to the observed marginal distribution of the test response	CONDDIST	TESTDATA(MWU)
CmpTrainObs	Distribution comparisons to the observed marginal distribution of the training response	CONDDIST	MWU
CondDistEst	Conditional distribution estimates	CONDDIST	Default
CorrB	Parameter estimate correlation matrix	MODEL	CORRB
CovB	Parameter estimate covariance matrix	MODEL	COVB
DensityEstInfo	Density estimation information	CONDDIST	PDF=KDE
Diagnostics	Outlier diagnostics	MODEL	DIAGNOSTICS
DiagSummary	Summary of the outlier diagnostics	MODEL	DIAGNOSTICS
IPIterHistory	Iteration history (interior point)	MODEL	ITPRINT
ModelInfo	Model information	MODEL	Default
NObs	Number of observations	PROC QUANTREG	Default
ObjFunction	Objective function	MODEL	Default
ParameterEstimates	Parameter estimates	MODEL	Default
ParmInfo	Parameter indices	MODEL	Default
PerfSettings	Performance settings	PERFORMANCE	DETAILS
ProcessEst	Quantile process estimates	MODEL	QUANTILE=
ProcessObj	Objective function for quantile process	MODEL	QUANTILE=
SMIterHistory	Iteration history (smoothing)	MODEL	ITPRINT
SummaryStatistics	Summary statistics for model variables	MODEL	Default
TestAcrossQuantiles	Results for across-quantiles test	TEST	QINTERACT
Tests	Results for tests	TEST	Default
ScalableTiming	Timing details	PERFORMANCE	DETAILS

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

For a single quantile, two plots are particularly useful in revealing outliers and leverage points: a scatter plot of the standardized residuals for the specified quantile against the robust distances and a scatter plot of the robust distances against the classical Mahalanobis distances. You can request these two plots by using the PLOT=RDPLOT and PLOT=DDPLOT options, respectively.

You can also request a normal quantile-quantile plot and a histogram of the standardized residuals for the specified quantile by using the PLOT=QQPLOT and PLOT=HISTOGRAM options, respectively.

You can request a plot of fitted conditional quantiles by the single continuous variable that is specified in the model by using the PLOT=FITPLOT option.

All these plots can be requested by specifying corresponding plot options in either the PROC QUANTREG statement or the MODEL statement. If you specify same plot options in both statements, options in the PROC QUANTREG statement override options in the MODEL statement.

In addition, you can specify the PLOT=QUANTPLOT option in the MODEL statement to request a quantile process plot with confidence bands.

For more information about these plots, see the PLOT= option in the PROC QUANTREG statement and the PLOT= option in the MODEL statement.

Besides the PROC QUANTREG and the MODEL statements, the CONDDIST statement can also output plots for conditional distribution analysis.

You can request the probability-probability plot by using the PLOT=PPPLOT option in the CONDDIST statement. This plot illustrates the power of the estimated conditional distributions on the response values by plotting the points for the regression quantile levels versus the sample quantile levels.

You can request the CDF plot by using the PLOT=CDFPLOT option in the CONDDIST statement. This plot illustrates the CDF estimates for the specified observations.

You can request the marginal CDF plot by using the PLOT=MCDFPLOT option in the CONDDIST statement. This plot illustrates the marginal CDF estimates and their confidence limit bands for the TrainObs, TestObs, and TestFit marginal CDFs.

You can request the PDF plot by using the PLOT=PDFPLOT option in the CONDDIST statement. This plot illustrates the PDF estimates for the specified observations.

For more information about the plots for the CONDDIST statement, see the PLOT= option in the CONDDIST statement.

All the plot options are summarized in Table 103.11.

Table 103.11 Options for Plots

Keyword	Plot
ALL	All appropriate plots
CDFPLOT	Series plot for CDF samples
DDPLOT	Robust distance versus Mahalanobis distance
FITPLOT	Conditional quantile fit versus independent variable
HISTOGRAM	Histogram of standardized robust residuals
MCDFPLOT	Series and CI bands plot for the bootstrap marginal CDFs
NONE	No plot
PDFPLOT	Series plot for PDF estimates
PPPLOT	Regression quantile level versus sample quantile level
QUANTPLOT	Scatter plot of regression quantile
QQPLOT	Q-Q plot of standardized robust residuals
RDPLOT	Standardized robust residual versus robust distance

The following subsections provide information about these graphs.

ODS Graph Names

The QUANTREG procedure assigns a name to each graph that it creates. You can use these names to refer to the graphs when you use ODS. The names along with the required statements and options are listed in Table 103.12.

Table 103.12 Graphs Produced by PROC QUANTREG

ODS Graph Name	Plot Description	Statements	Option
CDFPlot	Cumulative distribution functions	CONDDIST	CDFPLOT
DDPlot	Robust distance versus Mahalanobis distance	PROC QUANTREG, MODEL	DDPLOT
FitPlot	Quantile fit versus independent variable	PROC QUANTREG, MODEL	FITPLOT
Histogram	Histogram of standardized robust residuals	PROC QUANTREG, MODEL	HISTOGRAM
MCDFPlot	Marginal Cumulative distributions functions	CONDDIST	MCDFPLOT
PDFPlot	Probability density functions	CONDDIST	PDFPLOT
PPPlot	Regression quantile level versus sample quantile level	CONDDIST	PPPLOT
QQPlot	Q-Q plot of standardized robust residuals	PROC QUANTREG, MODEL	QQPLOT
QuantPanel	Panel of quantile plots with confidence limits	MODEL	QUANTPLOT
QuantPlot	Scatter plot for regression quantiles with confidence limits	MODEL	QUANTPLOT UNPACK
RDPlot	Standardized robust residual versus robust distance	PROC QUANTREG, MODEL	RDPLOT

Fit Plot

When the model has a single independent continuous variable (with or without the intercept), the QUANTREG procedure automatically creates a plot of fitted conditional quantiles against this independent variable for one or more quantiles that are specified in the MODEL statement.

The following example reuses the trout data set in the section "Analysis of Fish-Habitat Relationships" on page 8535 to show the fit plot for one or several quantiles:

```
ods graphics on;

proc quantreg data=trout ci=resampling;
   model LnDensity = WDRatio / quantile=0.9 seed=1268;
run;

proc quantreg data=trout ci=resampling;
   model LnDensity = WDRatio / quantile=0.5 0.75 0.9 seed=1268;
run;
```

For a single quantile, the confidence limits for the fitted conditional quantiles are also plotted if you specify the CI=RESAMPLING or CI=SPARSITY option. (See Figure 103.21.) For multiple quantiles, confidence limits are not plotted by default. (See Figure 103.22.) You can add the confidence limits on the plot by specifying the option PLOT=FITPLOT(SHOWLIMITS).

The QUANTREG procedure also provides fit plots for quantile regression splines and polynomials if they are based on a single continuous variable. (See Example 103.4 and Example 103.5 for some examples.)

Figure 103.21 Fit Plot with Confidence Limits

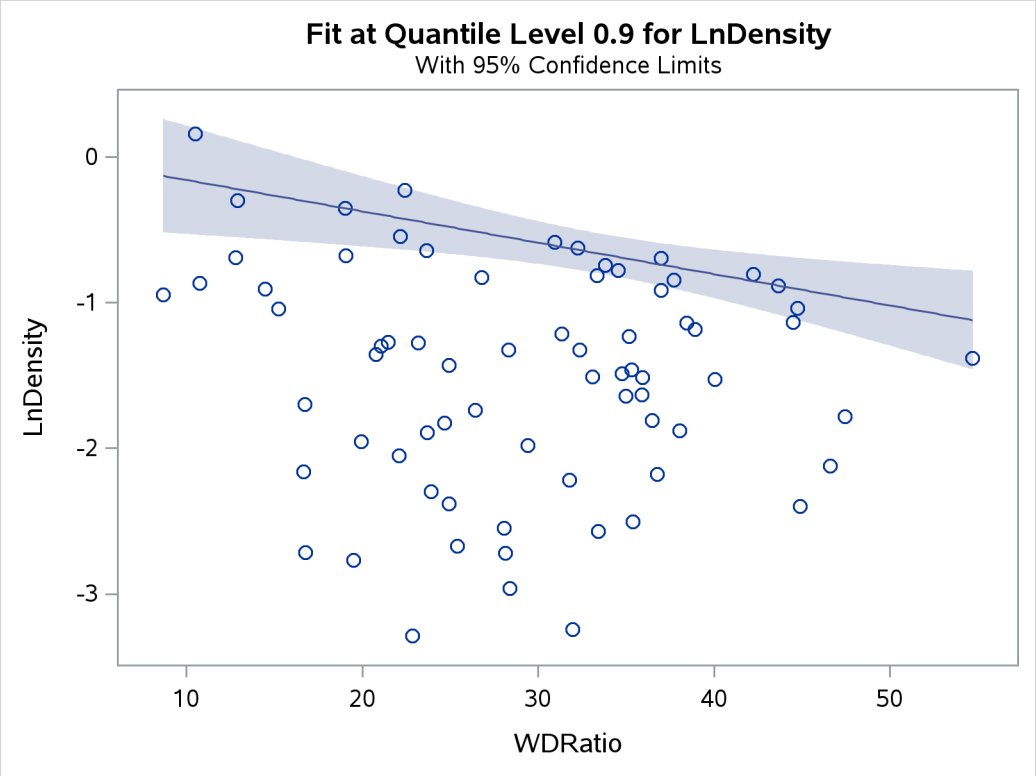

Figure 103.22 Fit Plot for Multiple Quantiles

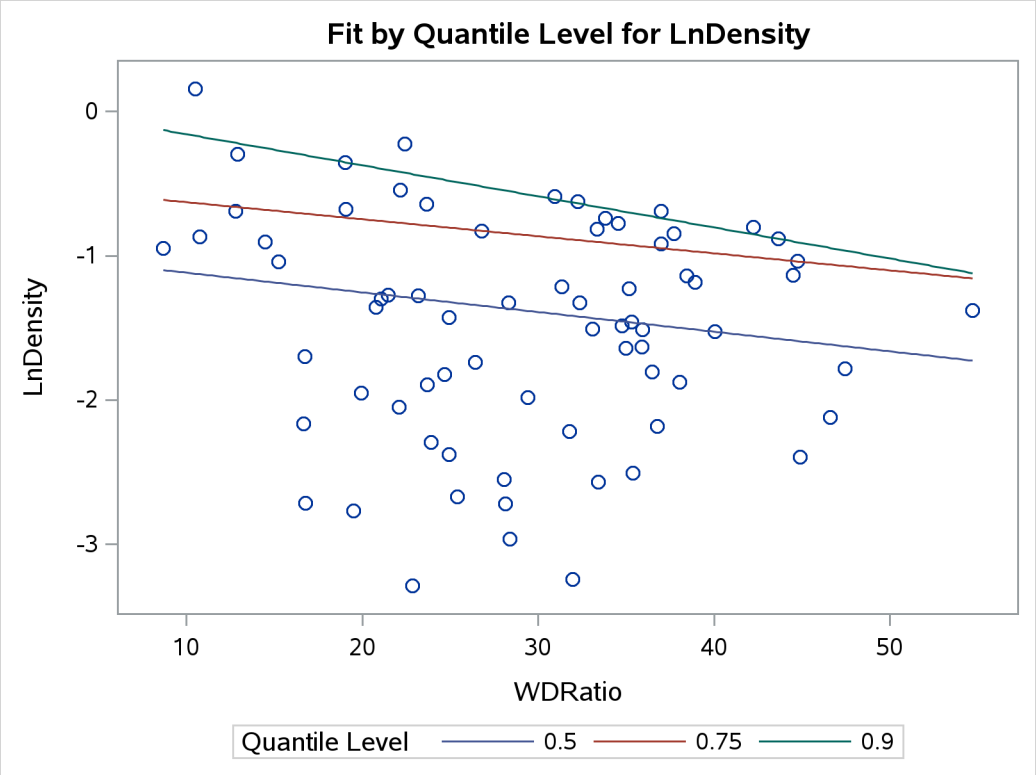

Quantile Process Plot

A quantile process plot is a scatter plot of an estimated regression parameter against a quantile. You can request this plot by specifying the PLOT=QUANTPLOT option in the MODEL statement when multiple regression quantiles are computed or when the entire quantile process is computed. Quantile process plots are often used to check model variations at different quantiles, which is usually called model heterogeneity.

By default, panels are used to hold multiple process plots (up to four in each panel). You can use the UNPACK option to request individual process plots. Figure 103.10 in the section "Analysis of Fish-Habitat Relationships" on page 8535 shows a panel that includes two quantile process plots. Output 103.2.9 in Example 103.2 shows a single quantile process plot. Example 103.3 demonstrates more quantile process plots and their usage.

Distance-Distance Plot

The distance-distance plot (DDPLOT) is mainly used for leverage-point diagnostics. It is a scatter plot of the robust distances against the classical Mahalanobis distances for the continuous independent variables. For more information about the robust distance, see the section "Leverage Point and Outlier Detection" on page 8594. If is a classification variable is specified in the model, this plot is not created.

You can use the PLOT=DDPLOT option to request this plot. The following statements use the growth data set in Example 103.2 to create a single plot, which is shown in Output 103.2.4 in Example 103.2:

```
proc quantreg data=growth ci=resampling plot=ddplot;
   model GDP = lgdp2 mse2 fse2 fhe2 mhe2 lexp2
               lintr2 gedy2 Iy2 gcony2 lblakp2 pol2 ttrad2
               / quantile=.5 diagnostics leverage(cutoff=8) seed=1268;
   id Country;
run;
```

The reference lines represent the cutoff values. The diagonal line is also drawn to show the distribution of the distances. By default, all outliers and leverage points are labeled with observation numbers. To change the default, you can use the LABEL= option as described in Table 103.4.

Residual-Distance Plot

The residual-distance plot (RDPLOT) is used for both outlier and leverage-point diagnostics. It is a scatter plot of the standardized residuals against the robust distances. For more information about the robust distance, see the section "Leverage Point and Outlier Detection" on page 8594. If a classification variable is specified in the model, this plot is not created.

You can use the PLOT=RDPLOT option to request this plot. The following statements use the growth data set in Example 103.2 to create a single plot, which is shown in Output 103.2.3 in Example 103.2:

```
proc quantreg data=growth ci=resampling plot=rdplot;
   model GDP = lgdp2 mse2 fse2 fhe2 mhe2 lexp2
               lintr2 gedy2 Iy2 gcony2 lblakp2 pol2 ttrad2
               / quantile=.5 diagnostics leverage(cutoff=8) seed=1268;
   id Country;
run;
```

The reference lines represent the cutoff values. By default, all outliers and leverage points are labeled with observation numbers. To change the default, you can use the LABEL= option as described in Table 103.4.

If you specify ID variables instead of observation numbers in the ID statement, the values of the first ID variable are used as labels.

Histogram and Q-Q Plot

PROC QUANTREG produces a histogram and a Q-Q plot for the standardized residuals. The histogram is superimposed with a normal density curve and a kernel density curve. Using the growth data set in Example 103.2, the following statements create the plot that is shown in Output 103.2.5 in Example 103.2:

```
proc quantreg data=growth ci=resampling plot=histogram;
   model GDP = lgdp2 mse2 fse2 fhe2 mhe2 lexp2
               lintr2 gedy2 Iy2 gcony2 lblakp2 pol2 ttrad2
               / quantile=.5 diagnostics leverage(cutoff=8) seed=1268;
   id Country;
run;
```

Examples: QUANTREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 103.1: Comparison of Algorithms

(View the complete code for this example (qregex1.sas) in the example repository.)

This example illustrates and compares the three algorithms for regression estimation available in the QUANTREG procedure. The simplex algorithm is the default because of its stability. Although this algorithm is slower than the interior point and smoothing algorithms for large data sets, the difference is not as significant for data sets with fewer than 5,000 observations and 50 variables. The simplex algorithm can also compute the entire quantile process, which is shown in Example 103.2.

The following statements generate 1,000 random observations. The first 950 observations are from a linear model, and the last 50 observations are significantly biased in the y-direction. In other words, 5% of the observations are contaminated with outliers.

```
data a (drop=i);
   do i=1 to 1000;
      x1=rannor(1234);
      x2=rannor(1234);
      e=rannor(1234);
      if i > 950 then y=100 + 10*e;
      else y=10 + 5*x1 + 3*x2 + 0.5 * e;
      output;
   end;
run;
```

The following statements invoke the QUANTREG procedure to fit a median regression model with the default simplex algorithm. They produce the results that are shown in Output 103.1.1 through Output 103.1.3.

```
proc quantreg data=a;
   model y = x1 x2;
run;
```

Output 103.1.1 displays model information and summary statistics for variables in the model. It indicates that the simplex algorithm is used to compute the optimal solution and that the rank method is used to compute confidence intervals of the parameters.

By default, the QUANTREG procedure fits a median regression model. This is indicated by the quantile value 0.5 in Output 103.1.2, which also displays the objective function value and the predicted value of the response at the means of the covariates.

Output 103.1.3 displays parameter estimates and confidence limits. These estimates are reasonable, which indicates that median regression is robust to the 50 outliers.

Output 103.1.1 Model Fit Information and Summary Statistics from the Simplex Algorithm

The QUANTREG Procedure

Model Information	
Data Set	WORK.A
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Optimization Algorithm	Simplex
Method for Confidence Limits	Inv_Rank
Quantile Levels	1

Summary Statistics

Variable	Q1	Median	Q3	Mean	Standard Deviation	MAD
x1	-0.6546	0.0230	0.7099	0.0222	0.9933	1.0085
x2	-0.7891	-0.0747	0.6839	-0.0401	1.0394	1.0857
y	6.1045	10.6936	14.9569	14.4864	20.4087	6.5696

Output 103.1.2 Quantile and Objective Function from the Simplex Algorithm

Quantile Level and Objective Function	
Quantile Level	0.5
Objective Function	2441.1927
Predicted Value at Mean	10.0259

Output 103.1.3 Parameter Estimates from the Simplex Algorithm

Parameter Estimates

Parameter	DF	Estimate	95% Confidence Limits	
Intercept	1	10.0364	9.9959	10.0756
x1	1	5.0106	4.9602	5.0388
x2	1	3.0294	2.9944	3.0630

The following statements refit the model by using the interior point algorithm:

```
proc quantreg algorithm=interior(tolerance=1e-6)
              ci=none data=a;
   model y = x1 x2 / itprint nosummary;
run;
```

The TOLERANCE= option specifies the stopping criterion for convergence of the interior point algorithm, which is controlled by the duality gap. Although the default criterion is 1E–8, the value 1E–6 is often sufficient. The ITPRINT option requests the iteration history for the algorithm. The option CI=NONE suppresses the computation of confidence limits, and the option NOSUMMARY suppresses the table of summary statistics.

Output 103.1.4 displays model fit information.

Output 103.1.4 Model Fit Information from the Interior Point Algorithm

The QUANTREG Procedure

Model Information	
Data Set	WORK.A
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Optimization Algorithm	Interior
Quantile Levels	1

Output 103.1.5 displays the iteration history of the interior point algorithm. Note that the duality gap is less than 1E–6 in the final iteration. The table also provides the number of iterations, the number of corrections, the primal step length, the dual step length, and the objective function value at each iteration.

Output 103.1.5 Iteration History for the Interior Point Algorithm

The QUANTREG Procedure
Quantile Level = 0.5

Iteration History of Interior Point Algorithm				
Iter	**Duality Gap**	**Primal Step**	**Dual Step**	**Objective Function**
1	2622.675	0.3113	0.4910	3303.469
2	3214.640	0.0427	1.0000	2461.377
3	1126.899	0.9882	0.3653	2451.134
4	760.887	0.3381	1.0000	2442.810
5	77.102902	1.0000	0.8916	2441.263
6	8.436664	0.9370	0.8381	2441.208
7	1.828685	0.8375	0.7674	2441.199
8	0.405843	0.6980	0.8636	2441.195
9	0.095499	0.9438	0.5955	2441.193
10	0.0066528	0.9818	0.9304	2441.193
11	0.00022482	0.9179	0.9994	2441.193
12	5.44651E-8	1.0000	1.0000	2441.193

Output 103.1.6 displays the parameter estimates that are obtained by using the interior point algorithm. These estimates are identical to those obtained by using the simplex algorithm.

Output 103.1.6 Parameter Estimates from the Interior Point Algorithm

Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	10.0364
x1	1	5.0106
x2	1	3.0294

The following statements refit the model by using the smoothing algorithm. They produce the results that are shown in Output 103.1.7 through Output 103.1.9.

```
proc quantreg algorithm=smooth(rratio=.5) ci=none data=a;
   model y = x1 x2 / itprint nosummary;
run;
```

The RRATIO= option controls the reduction speed of the threshold. Output 103.1.7 displays the model fit information.

Output 103.1.7 Model Fit Information from the Smoothing Algorithm

The QUANTREG Procedure

Model Information	
Data Set	WORK.A
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Optimization Algorithm	Smooth
Quantile Levels	1

Output 103.1.8 displays the iteration history of the smoothing algorithm. The threshold controls the convergence. Note that the thresholds decrease by a factor of at least 0.5, which is the value specified in the RRATIO= option. The table also provides the number of iterations, the number of factorizations, the number of full updates, the number of partial updates, and the objective function value in each iteration. For details concerning the smoothing algorithm, see Chen (2007).

Output 103.1.8 Iteration History for the Smoothing Algorithm

The QUANTREG Procedure
Quantile Level = 0.5

Iteration History of Smoothing Algorithm					
Iter	**Threshold**	**Refactor-ization**	**Full Update**	**Partial Update**	**Objective Function**
1	227.24557	1	1000	0	4267.0988
15	116.94090	4	1480	2420	3631.9653
17	1.44064	4	1480	2583	2441.4719
20	0.72032	5	1980	2598	2441.3315
22	0.36016	6	2248	2607	2441.2369
24	0.18008	7	2376	2608	2441.2056
26	0.09004	8	2446	2613	2441.1997
28	0.04502	9	2481	2617	2441.1971
30	0.02251	10	2497	2618	2441.1956
32	0.01126	11	2505	2620	2441.1946
34	0.00563	12	2510	2621	2441.1933
35	0.00281	13	2514	2621	2441.1930
36	0.0000846	14	2517	2621	2441.1927
37	1E-12	14	2517	2621	2441.1927

Output 103.1.9 displays the parameter estimates that are obtained by using the smoothing algorithm. These

estimates are identical to those obtained by using the simplex and interior point algorithms. All three algorithms should have the same parameter estimates unless the problem does not have a unique solution.

Output 103.1.9 Parameter Estimates from the Smoothing Algorithm

Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	10.0364
x1	1	5.0106
x2	1	3.0294

The interior point algorithm and the smoothing algorithm offer better performance than the simplex algorithm for large data sets. For more information about choosing an appropriate algorithm on the basis of data set size, see Chen (2004). All three algorithms should have the same parameter estimates, unless the optimization problem has multiple solutions.

Example 103.2: Quantile Regression for Econometric Growth Data

(View the complete code for this example (qregex2.sas) in the example repository.)

This example uses a SAS data set named Growth, which contains economic growth rates for countries during two time periods: 1965–1975 and 1975–1985. The data come from a study by Barro and Lee (1994) and have also been analyzed by Koenker and Machado (1999).

There are 161 observations and 15 variables in the data set. The variables, which are listed in the following table, include the national growth rates (GDP) for the two periods, 13 covariates, and a name variable (Country) for identifying the countries in one of the two periods.

Variable	Description
Country	Country's name and period
GDP	Annual change per capita in gross domestic product (GDP)
lgdp2	Initial per capita GDP
mse2	Male secondary education
fse2	Female secondary education
fhe2	Female higher education
mhe2	Male higher education
lexp2	Life expectancy
lintr2	Human capital
gedy2	Education/GDP
ly2	Investment/GDP
gcony2	Public consumption/GDP
lblakp2	Black market premium
pol2	Political instability
ttrad2	Growth rate terms trade

The goal is to study the effect of the covariates on GDP. The following statements request median regression for a preliminary exploration. They produce the results that are in Output 103.2.1 through Output 103.2.6.

```
data growth;
   length Country$ 22;
   input Country GDP lgdp2 mse2 fse2 fhe2 mhe2 lexp2 lintr2 gedy2
         Iy2 gcony2 lblakp2 pol2 ttrad2 @@;
   datalines;
Algeria75              .0415 7.330 .1320 .0670 .0050 .0220 3.880 .1138 .0382
                       .1898 .0601 .3823 .0833 .1001
Algeria85              .0244 7.745 .2760 .0740 .0070 .0370 3.978 -.107 .0437
                       .3057 .0850 .9386 .0000 .0657
Argentina75            .0187 8.220 .7850 .6200 .0740 .1660 4.181 .4060 .0221
                       .1505 .0596 .1924 .3575 -.011
Argentina85           -.014 8.407 .9360 .9020 .1320 .2030 4.211 .1914 .0243
                       .1467 .0314 .3085 .7010 -.052
Australia75            .0259 9.101 2.541 2.353 .0880 .2070 4.263 6.937 .0348
                       .3272 .0257 .0000 .0080 -.016

   ... more lines ...

Zambia75               .0120 6.989 .3760 .1190 .0130 .0420 3.757 .4388 .0339
                       .3688 .2513 .3945 .0000 -.032
Zambia85              -.046 7.109 .4200 .2740 .0110 .0270 3.854 .8812 .0477
                       .1632 .2637 .6467 .0000 -.033
Zimbabwe75             .0320 6.860 .1450 .0170 .0080 .0450 3.833 .7156 .0337
                       .2276 .0246 .1997 .0000 -.040
Zimbabwe85            -.011 7.180 .2200 .0650 .0060 .0400 3.944 .9296 .0520
                       .1559 .0518 .7862 .7161 -.024
;

ods graphics on;

proc quantreg data=growth ci=resampling
                plots=(rdplot ddplot reshistogram);
   model GDP = lgdp2 mse2 fse2 fhe2 mhe2 lexp2
               lintr2 gedy2 Iy2 gcony2 lblakp2 pol2 ttrad2
               / quantile=.5 diagnostics leverage(cutoff=8) seed=1268;
   id Country;
   test_lgdp2: test lgdp2 / lr wald;
run;
```

The QUANTREG procedure uses the default simplex algorithm to estimate the parameters and uses the MCMB resampling method to compute confidence limits.

Output 103.2.1 displays model information and summary statistics for the variables in the model. Six summary statistics are computed, including the median and the median absolute deviation (MAD), which are robust measures of univariate location and scale, respectively. For the variable lintr2 (human capital), both the mean and standard deviation are much larger than the corresponding robust measures (median and MAD), indicating that this variable might have outliers.

Output 103.2.1 Model Information and Summary Statistics

The QUANTREG Procedure

Model Information	
Data Set	WORK.GROWTH
Dependent Variable	GDP
Number of Independent Variables	13
Number of Observations	161
Optimization Algorithm	Simplex
Method for Confidence Limits	Resampling
Quantile Levels	1

Summary Statistics

Variable	Q1	Median	Q3	Mean	Standard Deviation	MAD
lgdp2	6.9890	7.7450	8.6080	7.7905	0.9543	1.1579
mse2	0.3160	0.7230	1.2675	0.9666	0.8574	0.6835
fse2	0.1270	0.4230	0.9835	0.7117	0.8331	0.5011
fhe2	0.0110	0.0350	0.0890	0.0792	0.1216	0.0400
mhe2	0.0400	0.1060	0.2060	0.1584	0.1752	0.1127
lexp2	3.8670	4.0640	4.2430	4.0440	0.2028	0.2728
lintr2	0.00160	0.5604	1.8805	1.4625	2.5491	1.0058
gedy2	0.0248	0.0343	0.0466	0.0360	0.0141	0.0151
ly2	0.1396	0.1955	0.2671	0.2010	0.0877	0.0981
gcony2	0.0480	0.0767	0.1276	0.0914	0.0617	0.0566
lblakp2	0	0.0696	0.2407	0.1916	0.3070	0.1032
pol2	0	0.0500	0.2429	0.1683	0.2409	0.0741
ttrad2	-0.0240	-0.0100	0.00730	-0.00570	0.0375	0.0239
GDP	0.00290	0.0196	0.0351	0.0191	0.0248	0.0237

Output 103.2.2 displays the parameter estimates and 95% confidence limits that are computed with the rank method.

Output 103.2.2 Parameter Estimates

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	-0.0488	0.0733	-0.1937	0.0961	-0.67	0.5065
lgdp2	1	-0.0269	0.0041	-0.0350	-0.0188	-6.58	<.0001
mse2	1	0.0110	0.0080	-0.0048	0.0269	1.38	0.1710
fse2	1	-0.0011	0.0088	-0.0185	0.0162	-0.13	0.8960
fhe2	1	0.0148	0.0321	-0.0485	0.0782	0.46	0.6441
mhe2	1	0.0043	0.0268	-0.0487	0.0573	0.16	0.8735
lexp2	1	0.0683	0.0229	0.0232	0.1135	2.99	0.0033
lintr2	1	-0.0022	0.0015	-0.0052	0.0008	-1.44	0.1513
gedy2	1	-0.0508	0.1654	-0.3777	0.2760	-0.31	0.7589
ly2	1	0.0723	0.0248	0.0233	0.1213	2.92	0.0041
gcony2	1	-0.0935	0.0382	-0.1690	-0.0181	-2.45	0.0154
lblakp2	1	-0.0269	0.0084	-0.0435	-0.0104	-3.22	0.0016
pol2	1	-0.0301	0.0093	-0.0485	-0.0117	-3.23	0.0015
ttrad2	1	0.1613	0.0740	0.0149	0.3076	2.18	0.0310

Diagnostics for the median regression fit, which are requested in the PLOTS= option, are displayed in Output 103.2.3 and Output 103.2.4. Output 103.2.3 plots the standardized residuals from median regression against the robust MCD distance. This display is used to diagnose both vertical outliers and horizontal leverage points. Output 103.2.4 plots the robust MCD distance against the Mahalanobis distance. This display is used to diagnose leverage points.

The cutoff value 8, which is specified in the LEVERAGE option, is close to the maximum of the Mahalanobis distance. Eighteen points are diagnosed as high leverage points, and almost all are countries with high human capital, which is the major contributor to the high leverage as observed from the summary statistics. Four points are diagnosed as outliers by using the default cutoff value of 3. However, these are not extreme outliers.

A histogram of the standardized residuals and two fitted density curves are displayed in Output 103.2.5. This output shows that median regression fits the data well.

Output 103.2.3 Plot of Residual versus Robust Distance

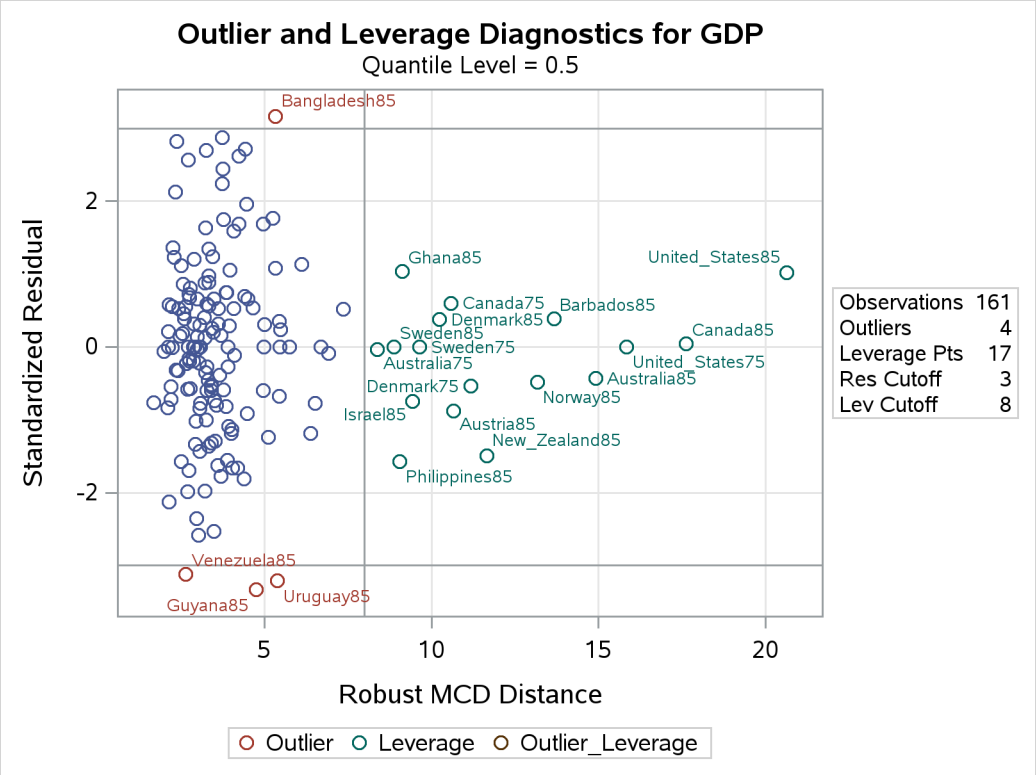

Output 103.2.4 Plot of Robust Distance versus Mahalanobis Distance

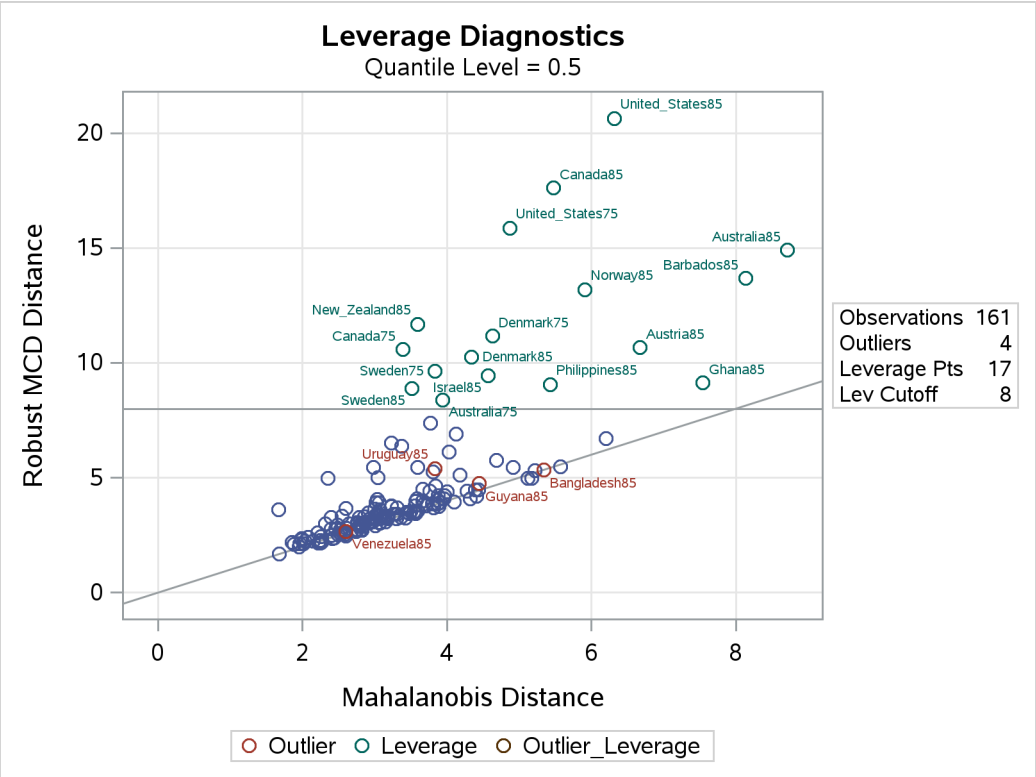

Output 103.2.5 Histogram for Residuals

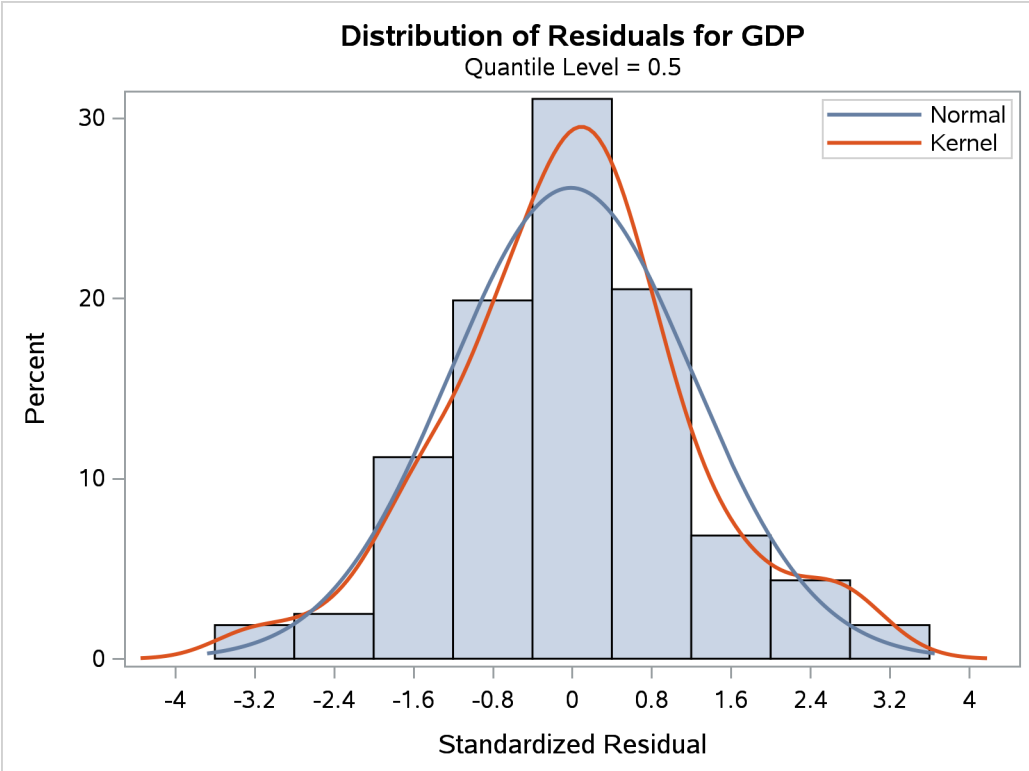

Tests of significance for the initial per-capita GDP (LGDP2) are shown in Output 103.2.6.

Output 103.2.6 Tests for Regression Coefficient

Test test_lgdp2 Results				
Test	Test Statistic	DF	Chi-Square	Pr > ChiSq
Wald	43.2684	1	43.27	<.0001
Likelihood Ratio	36.3047	1	36.30	<.0001

The QUANTREG procedure computes entire quantile processes for covariates when you specify QUANTILE=PROCESS in the MODEL statement, as follows:

```
proc quantreg data=growth ci=resampling;
   model GDP = lgdp2 mse2 fse2 fhe2 mhe2 lexp2 lintr2
               gedy2 Iy2 gcony2 lblakp2 pol2 ttrad2
               / quantile=process plot=quantplot seed=1268;
run;
```

Confidence limits for quantile processes can be computed by using the sparsity or resampling methods. But they cannot be computed by using the rank method, because the computation would be prohibitively expensive.

A total of 14 quantile process plots are produced. Output 103.2.7 and Output 103.2.8 display two panels of eight selected process plots. The 95% confidence bands are shaded.

Output 103.2.7 Quantile Processes with 95% Confidence Bands

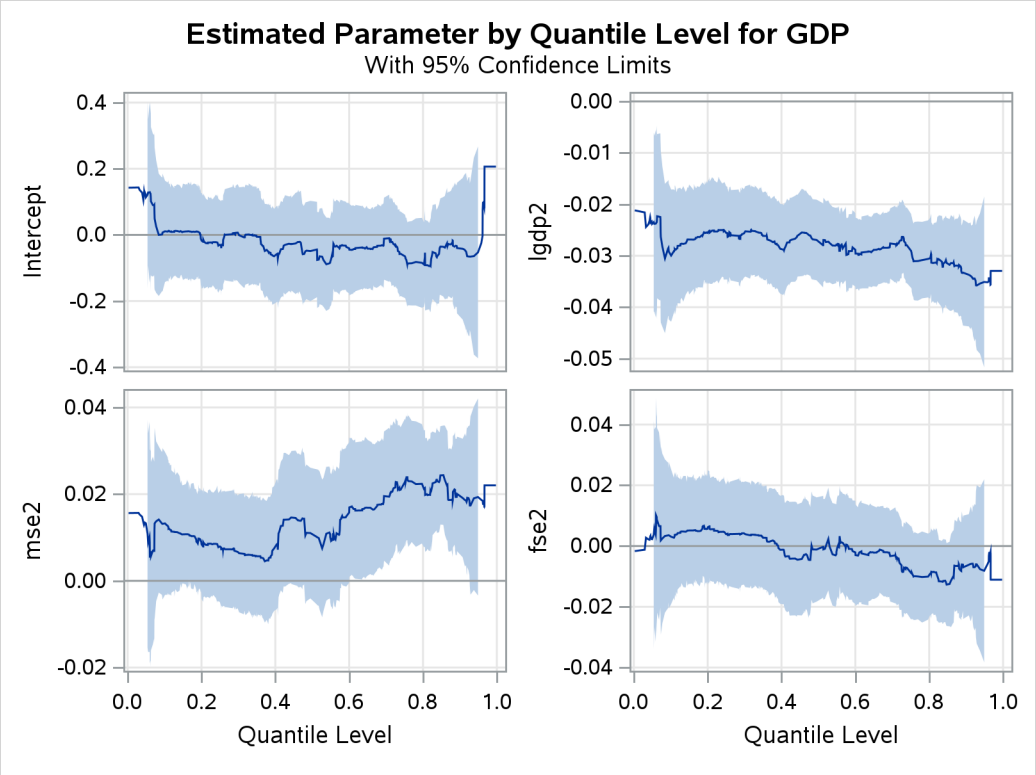

Output 103.2.8 Quantile Processes with 95% Confidence Bands

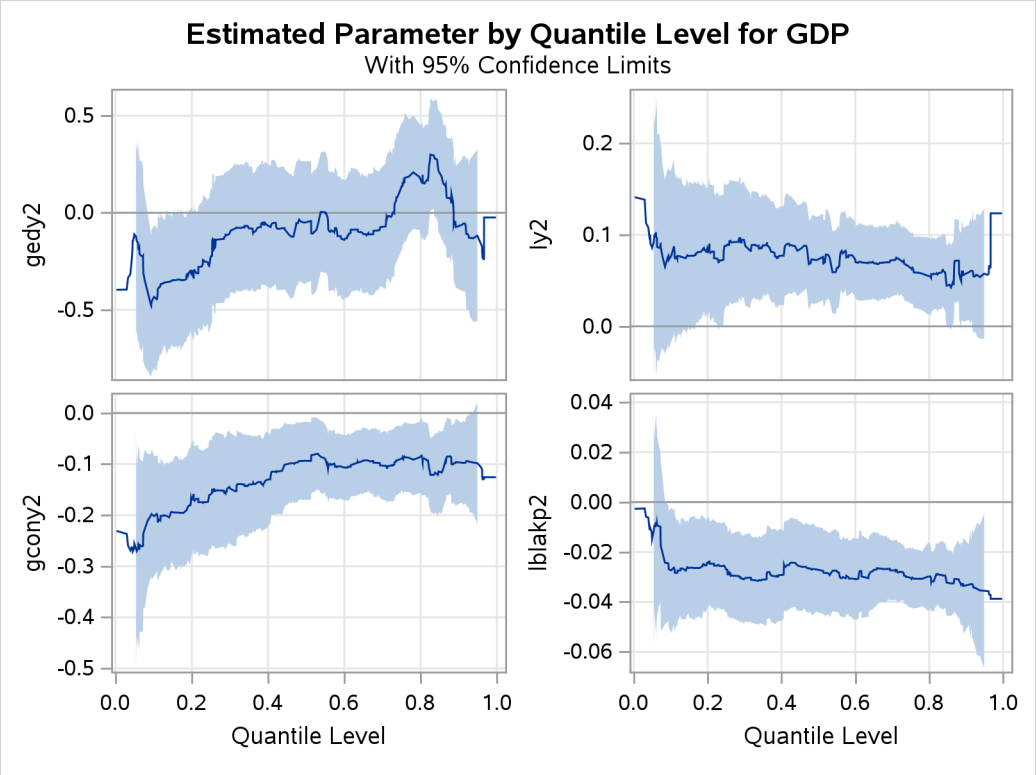

As pointed out by Koenker and Machado (1999), previous studies of the Barro growth data have focused on the effect of the initial per-capita GDP on the growth of this variable (annual change in per-capita GDP). The following statements request a single process plot for this effect:

```
proc quantreg data=growth ci=resampling;
   model GDP = lgdp2 mse2 fse2 fhe2 mhe2 lexp2 lintr2
               gedy2 Iy2 gcony2 lblakp2 pol2 ttrad2
               / quantile=process plot=quantplot(lgdp2) seed=1268;
run;
```

The plot is shown in Output 103.2.9.

Output 103.2.9 Quantile Process Plot for LGDP2

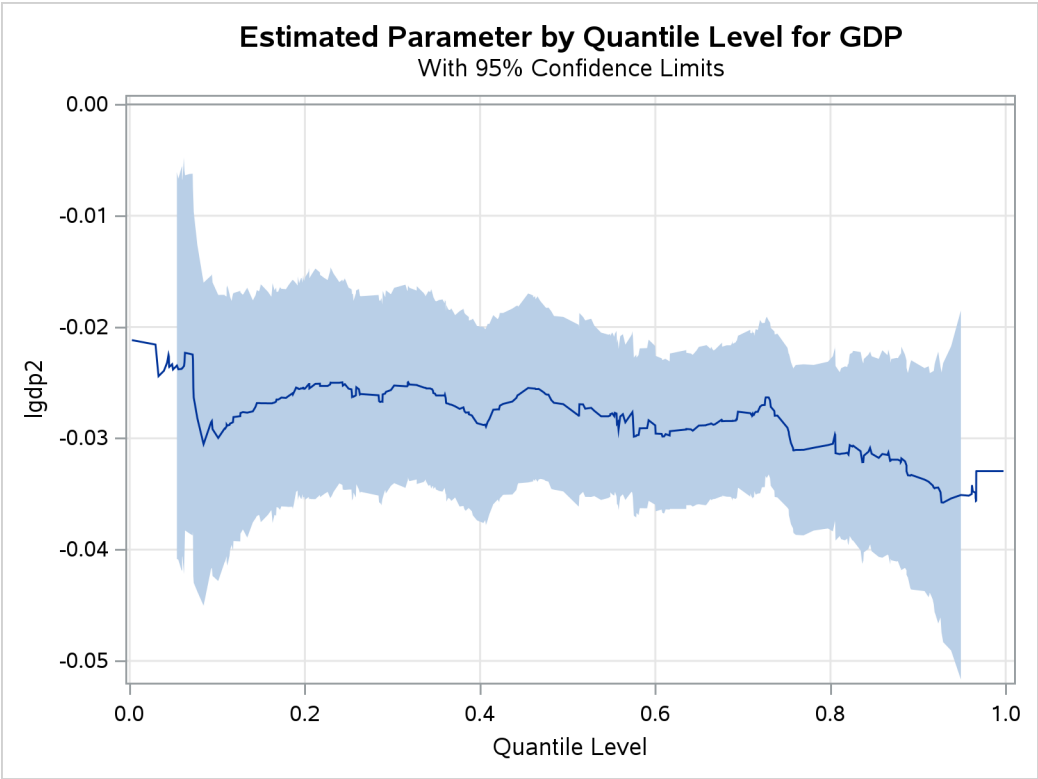

The confidence bands here are computed by using the MCMB resampling method. In contrast, Koenker and Machado (1999) used the rank method to compute confidence limits for a few selected points. Output 103.2.9 suggests that the effect of the initial level of GDP is relatively constant over the entire distribution, with a slightly stronger effect in the upper tail.

The effects of other covariates are quite varied. An interesting covariate is public consumption divided by GDP (gcony2) (first plot in second panel), which has a constant effect over the upper half of the distribution and a larger effect in the lower tail. For an analysis of the effects of the other covariates, see Koenker and Machado (1999).

Example 103.3: Quantile Regression Analysis of Birth-Weight Data

(View the complete code for this example (qregex3.sas) in the example repository.)

This example is patterned after a quantile regression analysis of covariates associated with birth weight that was carried out by Koenker and Hallock (2001). Their study uses a subset of the June 1997 Detailed Natality Data, which was published by the National Center for Health Statistics. The study demonstrates that conditional quantile functions provide more complete information about the covariate effects than ordinary least squares regression provides.

This example is based on Koenker and Hallock (2001); Abreveya (2001); it uses data for live, singleton births to mothers in the United States who were recorded as black or white, and who were between the ages of 18 and 45. For convenience, this example uses 50,000 observations, which are randomly selected from the qualified observations. Observations that have missing data for any of the variables are deleted. The data are available in the data set Sashelp.BWeight. The following step displays in Output 103.3.1 the variables in the data set:

```
proc contents varnum data=sashelp.bweight;
   ods select position;
run;
```

Output 103.3.1 Sashelp.BWeight Data Set

The CONTENTS Procedure

Variables in Creation Order				
#	Variable	Type	Len	Label
1	Weight	Num	8	Infant Birth Weight
2	Black	Num	8	Black Mother
3	Married	Num	8	Married Mother
4	Boy	Num	8	Baby Boy
5	MomAge	Num	8	Mother's Age
6	MomSmoke	Num	8	Smoking Mother
7	CigsPerDay	Num	8	Cigarettes Per Day
8	MomWtGain	Num	8	Mother's Pregnancy Weight Gain
9	Visit	Num	8	Prenatal Visit
10	MomEdLevel	Num	8	Mother's Education Level

The following step creates descriptive labels for the values of the classification variables Visit and MomEdLevel:

```
proc format;
   value vfmt 0 = 'No Visit'       1 = 'Second Trimester'
              2 = 'Last Trimester' 3 = 'First Trimester';
   value efmt 0 = 'High School'    1 = 'Some College'
              2 = 'College'        3 = 'Less Than High School';
run;
```

There are four levels of maternal education. When you specify the ORDER=INTERNAL option, PROC QUANTREG treats the highest unformatted value (3, which represents that the mother's education level is less than high school) as a reference level. The regression coefficients of other levels measure the effect

relative to this level. Likewise, there are four levels of prenatal medical care of the mother, and a first visit in the first trimester serves as the reference level.

The following statements fit a regression model for 19 quantiles of birth weight, which are evenly spaced in the interval (0, 1). The model includes linear and quadratic effects for the age of the mother and for weight gain during pregnancy.

```
ods graphics on;

proc quantreg ci=sparsity/iid algorithm=interior(tolerance=5.e-4)
   data=sashelp.bweight order=internal;
   class Visit MomEdLevel;
   model Weight = Black Married Boy Visit MomEdLevel MomSmoke
   CigsPerDay MomAge MomAge*MomAge
   MomWtGain MomWtGain*MomWtGain /
   quantile= 0.05 to 0.95 by 0.05
   plot=quantplot;
   format Visit vfmt. MomEdLevel efmt.;
run;
```

Output 103.3.2 displays the model information and summary statistics for the variables in the model.

Output 103.3.2 Model Information and Summary Statistics

The QUANTREG Procedure

Model Information		
Data Set	SASHELP.BWEIGHT	Infant Birth Weight
Dependent Variable	Weight	Infant Birth Weight
Number of Independent Variables	9	
Number of Continuous Independent Variables	7	
Number of Class Independent Variables	2	
Number of Observations	50000	
Optimization Algorithm	Interior	
Method for Confidence Limits	Sparsity	
Quantile Levels	19	

Summary Statistics						
Variable	**Q1**	**Median**	**Q3**	**Mean**	**Standard Deviation**	**MAD**
Black	0	0	0	0.1628	0.3692	0
Married	0	1.0000	1.0000	0.7126	0.4525	0
Boy	0	1.0000	1.0000	0.5158	0.4998	0
MomSmoke	0	0	0	0.1307	0.3370	0
CigsPerDay	0	0	0	1.4766	4.6541	0
MomAge	-4.0000	0	5.0000	0.4161	5.7285	5.9304
MomAge*MomAge	4.0000	16.0000	49.0000	32.9877	39.2861	22.2390
MomWtGain	-8.0000	0	9.0000	0.7092	12.8761	11.8608
MomWtGain*MomWtGain	16.0000	64.0000	196.0	166.3	298.8	88.9561
Weight	3062.0	3402.0	3720.0	3370.8	566.4	504.1

Among the 11 independent variables, Black, Married, Boy, and MomSmoke are binary variables. For these variables, the mean represents the proportion in the category. The two continuous variables, MomAge and

MomWtGain, are centered at their medians, which are 27 and 30, respectively.

The quantile plots for the intercept and the other 15 factors with nonzero degrees of freedom are shown in the following four panels. In each plot, the regression coefficient at a given quantile indicates the effect on birth weight of a unit change in that factor, assuming that the other factors are fixed. The bands represent 95% confidence intervals.

Although the data set used here is a subset of the Natality data set, the results are quite similar to those of Koenker and Hallock (2001) for the full data set.

In Output 103.3.3, the first plot is for the intercept. As explained by Koenker and Hallock (2001), the intercept "may be interpreted as the estimated conditional quantile function of the birth-weight distribution of a girl born to an unmarried, white mother with less than a high school education, who is 27 years old and had a weight gain of 30 pounds, didn't smoke, and had her first prenatal visit in the first trimester of the pregnancy."

The second plot shows that infants born to black mothers weigh less than infants born to white mothers, especially in the lower tail of the birth-weight distribution. The third plot shows that marital status has a large positive effect on birth weight, especially in the lower tail. The fourth plot shows that boys weigh more than girls for any chosen quantile; this difference is smaller in the lower quantiles of the distribution.

In Output 103.3.4, the first three plots deal with prenatal care. Compared with babies born to mothers who had a prenatal visit in the first trimester, babies born to mothers who received no prenatal care weigh less, especially in the lower quantiles of the birth-weight distributions. As noted by Koenker and Hallock (2001), "babies born to mothers who delayed prenatal visits until the second or third trimester have substantially *higher* birthweights in the lower tail than mothers who had a prenatal visit in the first trimester. This might be interpreted as the self-selection effect of mothers confident about favorable outcomes."

The fourth plot in Output 103.3.4 and the first two plots in Output 103.3.5 are for variables that are related to education. Education beyond high school is associated with a positive effect on birth weight. The effect of high school education is uniformly around 15 grams across the entire birth-weight distribution (this is a pure location shift effect), whereas the effect of some college and college education is more positive in the lower quantiles than the upper quantiles.

The remaining two plots in Output 103.3.5 show that smoking is associated with a large negative effect on birth weight.

The linear and quadratic effects for the two continuous variables are shown in Output 103.3.6. Both of these variables are centered at their median. At the lower quantiles, the quadratic effect of the mother's age is more concave. The optimal age at the first quantile is about 33, and the optimal age at the third quantile is about 38. The effect of the mother's weight gain is clearly positive, as indicated by the narrow confidence bands for both linear and quadratic coefficients.

For more information about the covariate effects that are discovered by using quantile regression, see Koenker and Hallock (2001).

Output 103.3.3 Quantile Processes with 95% Confidence Bands

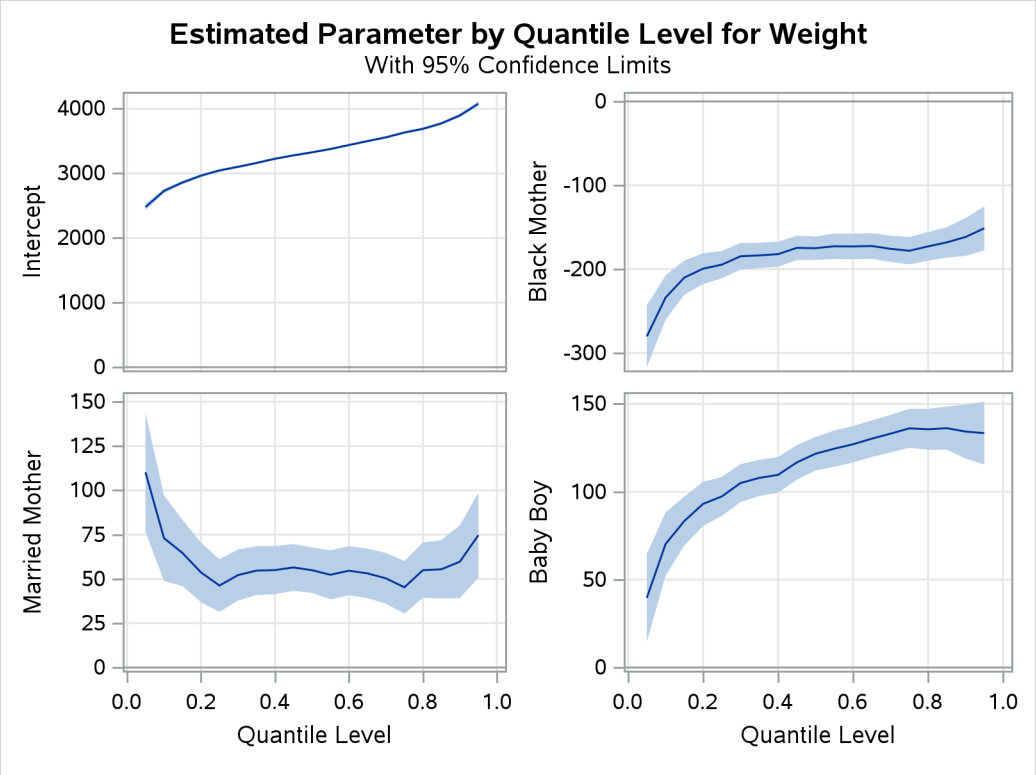

Output 103.3.4 Quantile Processes with 95% Confidence Bands

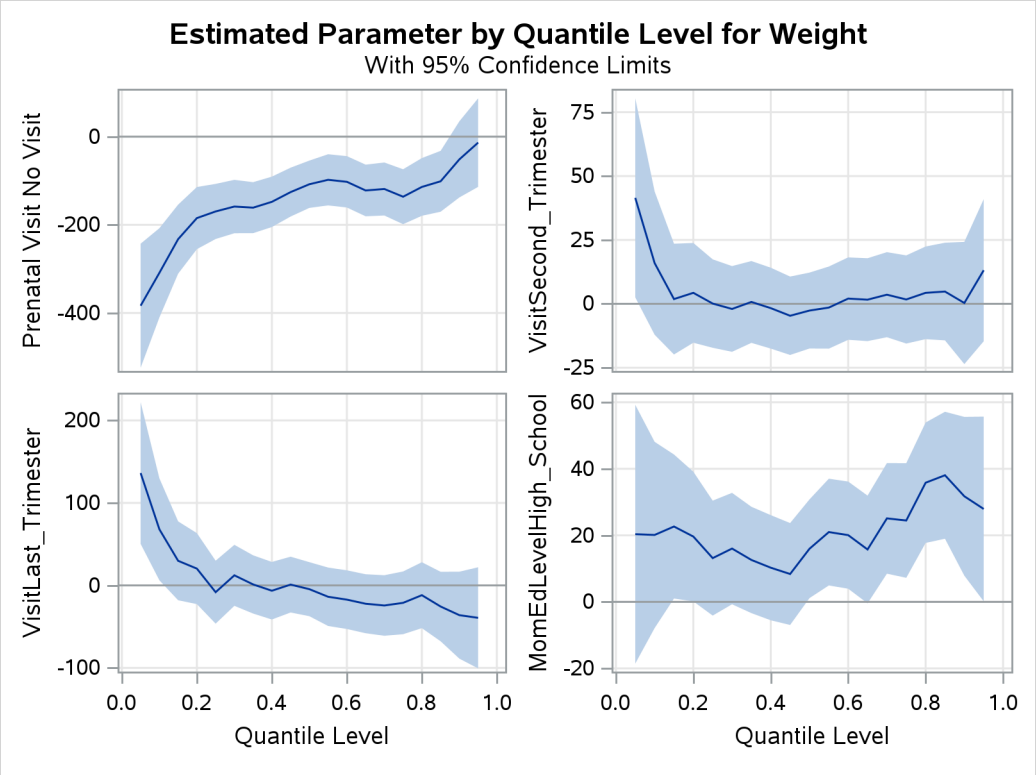

Output 103.3.5 Quantile Processes with 95% Confidence Bands

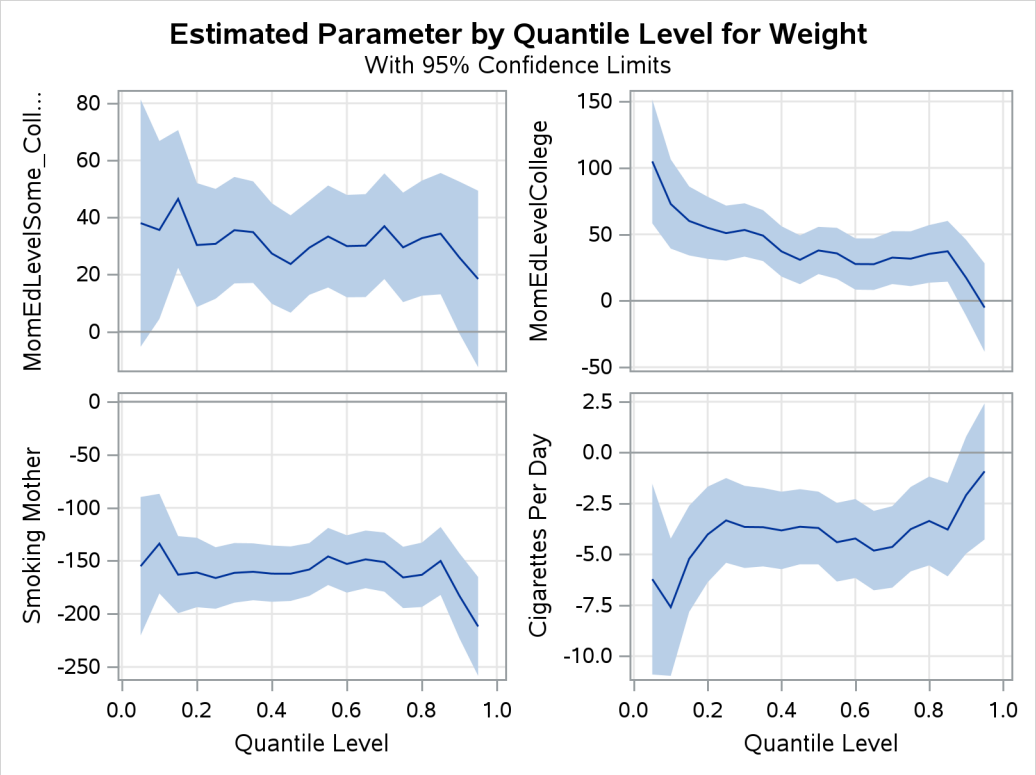

Output 103.3.6 Quantile Processes with 95% Confidence Bands

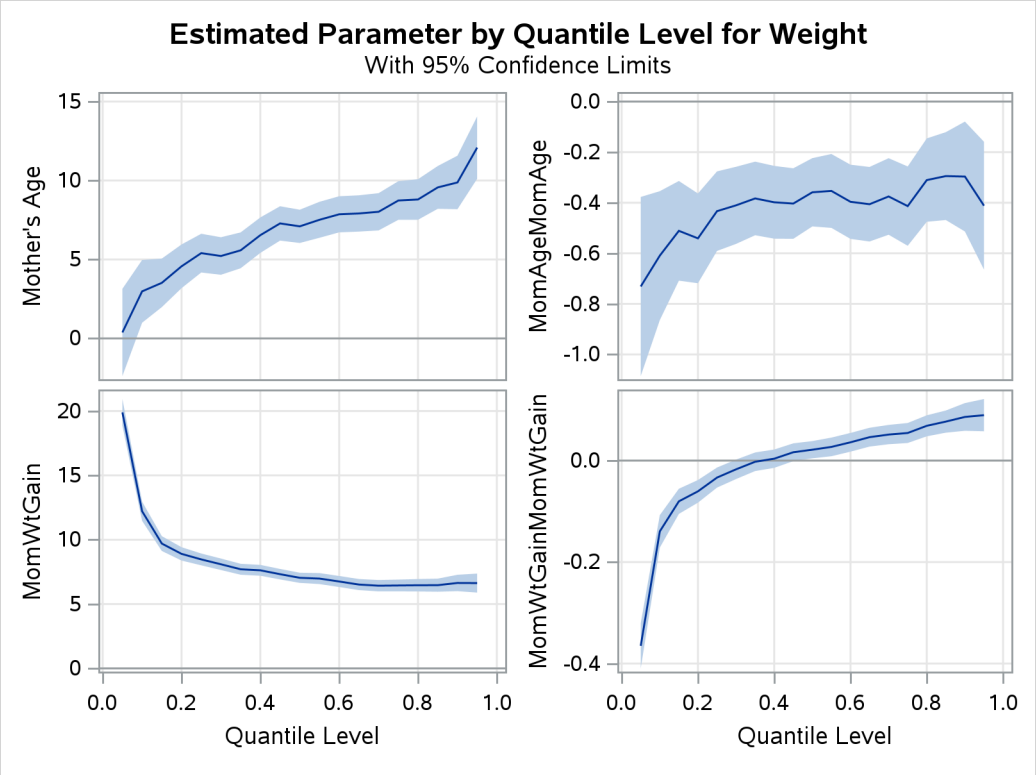

Example 103.4: Nonparametric Quantile Regression for Ozone Levels

(View the complete code for this example (qregex4.sas) in the example repository.)

Tracing seasonal trends in the level of tropospheric ozone is essential for predicting high-level periods, observing long-term trends, and discovering potential changes in pollution. Traditional methods for modeling seasonal effects are based on the conditional mean of ozone concentration. However, the upper conditional quantiles are more critical from a public-health perspective. In this example, the QUANTREG procedure fits conditional quantile curves for seasonal effects by using nonparametric quantile regression with cubic B-splines.

The data used here are from Chock, Winkler, and Chen (2000), who studied the association between daily mortality and ambient air pollutant concentrations in Pittsburgh, Pennsylvania. The data set ozone contains the following two variables: Ozone, which represents the daily maximum one-hour ozone concentration (ppm) and Days, which is an index of 1,095 days (3 years).

```
data ozone;
   days = _n_;
   input ozone @@;
   datalines;
0.0060 0.0060 0.0320 0.0320 0.0320 0.0150 0.0150 0.0150 0.0200 0.0200
0.0160 0.0070 0.0270 0.0160 0.0150 0.0240 0.0220 0.0220 0.0220 0.0185
0.0150 0.0150 0.0110 0.0070 0.0070 0.0240 0.0380 0.0240 0.0265 0.0290
0.0310 0.0460 0.0360 0.0260 0.0300 0.0250 0.0280 0.0310 0.0370 0.0325

   ... more lines ...

0.0220 0.0210 0.0210 0.0130 0.0130 0.0130 0.0330 0.0330 0.0330 0.0325
0.0320 0.0320 0.0320 0.0120 0.0200 0.0200 0.0200 0.0320 0.0320 0.0250
0.0180 0.0180 0.0270 0.0270 0.0290
;
```

Output 103.4.1, which displays the time series plot of ozone concentration for the three years, shows a clear seasonal pattern.

In this example, cubic B-splines are used to fit the seasonal effect. These splines are generated with 11 knots, which split the 3 years into 12 seasons. The following statements create the spline basis and fit multiple quantile regression spline curves:

```
ods graphics on;

proc quantreg data=ozone algorithm=smooth ci=none plot=fitplot(nodata);
   effect sp = spline( days / knotmethod = list
          (90 182 272 365 455 547 637 730 820 912 1002) );
   model ozone = sp / quantile = 0.5 0.75 0.90 0.95 seed=1268;
run;
```

Output 103.4.1 Time Series of Ozone Levels in Pittsburgh, Pennsylvania

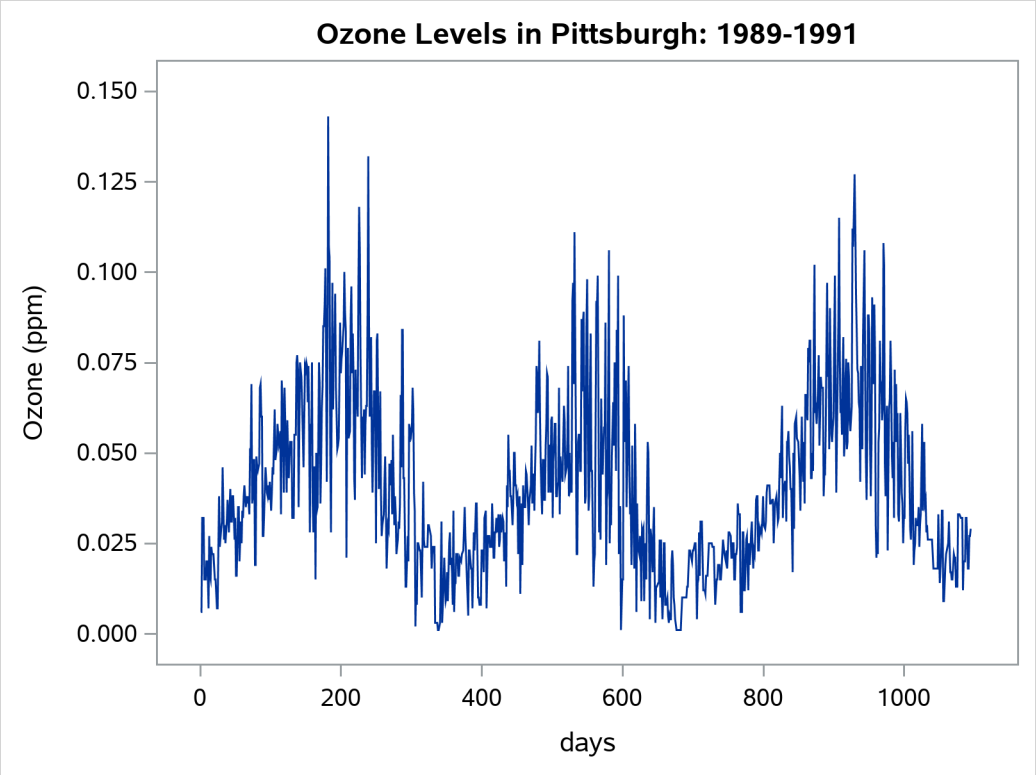

The EFFECT statement creates spline bases for the variable Days. The KNOTMETHOD=LIST option provides all internal knots for these bases. Cubic spline bases are generated by default. These bases are treated as components of the spline effect *sp*, which is specified in the MODEL statement. Spline fits for four quantiles are requested in the QUANTILE= option.

When ODS Graphics is enabled, the QUANTREG procedure automatically generates a fit plot, which includes all fitted curves.

Output 103.4.2 displays these curves. The curves show that peak ozone levels occur in the summer. For the three years 1989–1991, the median curve (labeled 50%) does not cross the 0.08 ppm line, which is the 1997 EPA eight-hour standard. The median curve and the 75% curve show a drop for the ozone concentration levels in 1990. However, for the 90% and 95% curves, peak ozone levels tend to increase. This indicates that there might have been more days with low ozone concentration in 1990, but the top 10% and 5% tend to have higher ozone concentration levels.

Output 103.4.2 Quantiles of Ozone Levels in Pittsburgh, Pennsylvania

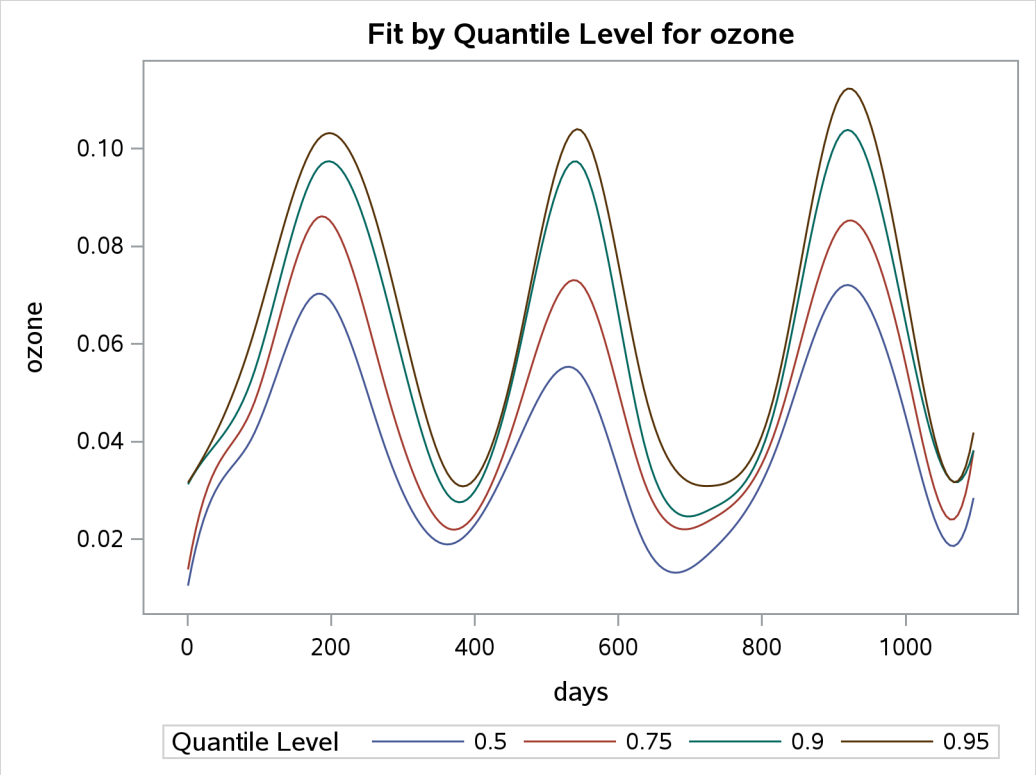

The quantile curves also show that high ozone concentration in 1989 had a longer duration than in 1990 and 1991. This is indicated by the wider spread of the quantile curves in 1989.

Example 103.5: Quantile Polynomial Regression for Salary Data

(View the complete code for this example (qregex5.sas) in the example repository.)

This example uses the data set from a university union survey of salaries of professors in 1991. The survey covered departments in US colleges and universities that list programs in statistics. The goal of this example is to examine the relationship between faculty salaries and years of service.

The data include salaries and years of service for 459 professors. The scatter plot in Output 103.5.1 shows that the relationship is not linear and that a quadratic or cubic regression curve is appropriate. Output 103.5.1 shows a cubic curve.

The curve in Output 103.5.1 does not adequately describe the conditional salary distributions and how they change with length of service. Output 103.5.2 shows the 25th, 50th, and 75th percentiles for each number of years, which gives a better picture of the conditional distributions.

```
data salary;
   input Salaries Years @@;
   label Salaries='Salaries (1000s of dollars)';
   datalines;
54.94   2  58.24   2  58.11   2  52.23   2  52.98    2  57.62    2
```

```
44.48   2  57.22   2  54.24   2  54.79   2  56.42    2  61.90    2
63.90   2  64.10   2  47.77   2  54.86   2  49.31    2  53.37    2
51.69   2  53.66   2  58.77   2  56.77   2  53.06    2  54.86    2
50.96   2  56.46   2  51.67   2  49.37   2  56.86    2  49.85    2

   ... more lines ...

85.72  25  64.87  25  51.76  25  51.11  25  51.31   25  78.28   25
57.91  25  86.78  25  58.27  25  56.56  25  76.33   25  61.83   25
69.13  25  63.15  25  66.13  25
;
```

Output 103.5.1 Salary and Years as Professor: Cubic Fit

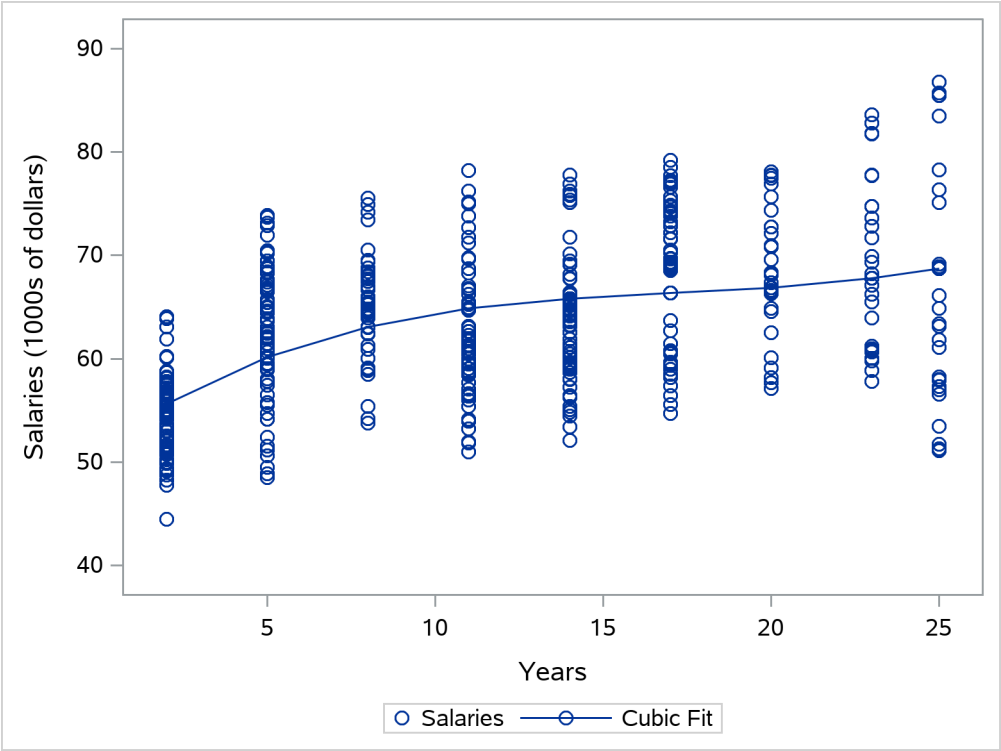

Output 103.5.2 Salary and Years as Professor: Sample Quantiles

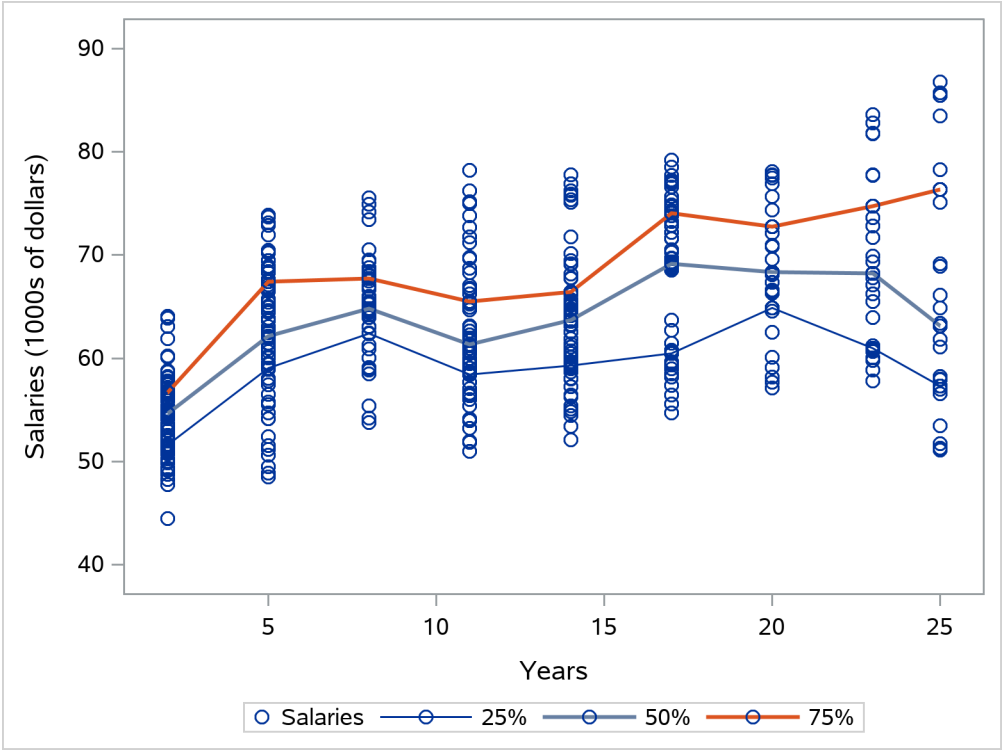

These descriptive percentiles do not clearly show trends with length of service. The following statements use polynomial quantile regression to obtain a smooth version.

```
ods graphics on;

proc quantreg data=salary ci=sparsity;
   model salaries = years years*years years*years*years
                    /quantile=0.25 0.5 0.75
                    plot=fitplot(showlimits);

   test  years/QINTERACT;

run;
```

The results are shown in Output 103.5.3 and Output 103.5.5. Output 103.5.3 displays the regression coefficients for the three quantiles, from which you can see a difference among the estimated parameters of the variable *years* across the three quantiles. To test whether the difference is significant, you can specify the option QINTERACT in the TEST statement. Output 103.5.4 indicates that the difference is not significant (the p-value is greater than 0.05).

Output 103.5.3 Regression Coefficients

The QUANTREG Procedure
Quantile Level = 0.25

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	48.2509	1.3484	45.6011	50.9007	35.78	<.0001
Years	1	2.2234	0.5455	1.1514	3.2953	4.08	<.0001
Years*Years	1	-0.1292	0.0500	-0.2275	-0.0308	-2.58	0.0101
Years*Years*Years	1	0.0024	0.0013	-0.0001	0.0049	1.86	0.0634

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	50.2512	1.2812	47.7334	52.7690	39.22	<.0001
Years	1	2.7173	0.5947	1.5485	3.8860	4.57	<.0001
Years*Years	1	-0.1632	0.0632	-0.2873	-0.0390	-2.58	0.0101
Years*Years*Years	1	0.0034	0.0018	-0.0002	0.0070	1.85	0.0647

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	51.0298	1.5886	47.9078	54.1517	32.12	<.0001
Years	1	3.6513	0.7594	2.1590	5.1436	4.81	<.0001
Years*Years	1	-0.2390	0.0764	-0.3892	-0.0888	-3.13	0.0019
Years*Years*Years	1	0.0055	0.0021	0.0013	0.0096	2.60	0.0098

Output 103.5.4 Tests for Heteroscedasticity

Test Results Equal Coefficients Across Quantiles

Chi-Square	DF	Pr > ChiSq
3.4026	2	0.1825

The three fitted quantile curves and their 95% confidence limits in the Output 103.5.5 clearly show that salary dispersion increases gradually with length of service. After 15 years, a salary more than $70,000 is relatively high, whereas a salary less than $60,000 is relatively low. Percentile curves of this type are useful in medical science as reference curves (Yu, Lu, and Stander 2003).

Output 103.5.5 Salary and Years as Professor: Regression Quantiles

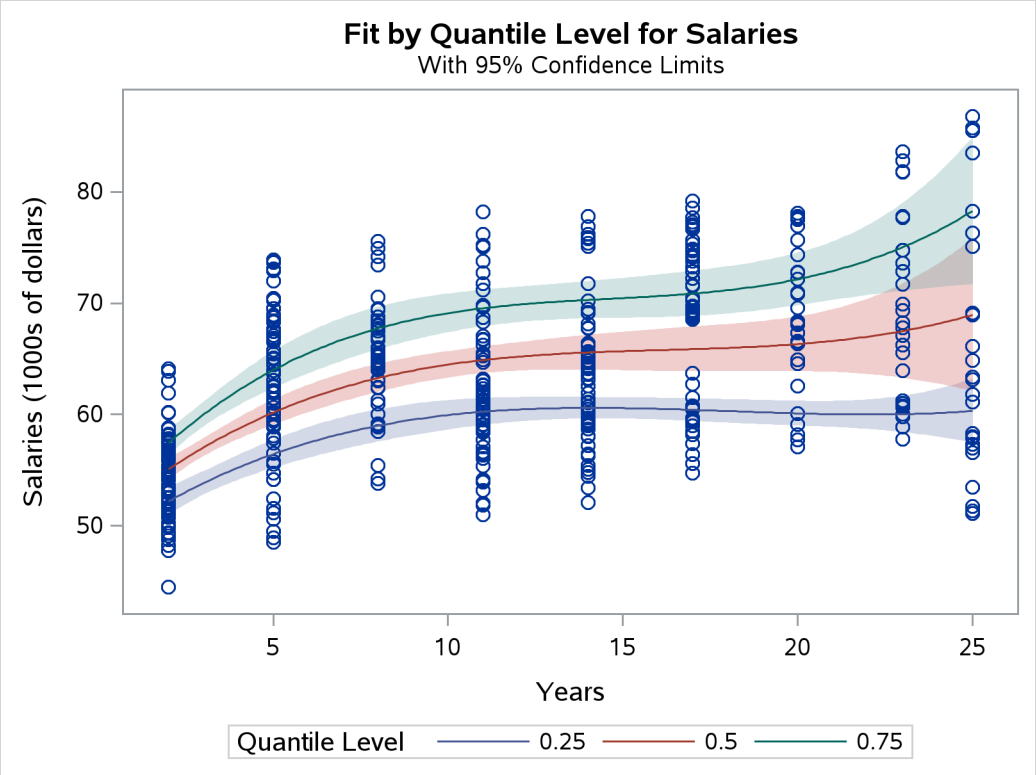

Example 103.6: Counterfactual Analysis of Smoking-Weight Data

(View the complete code for this example (qregex7.sas) in the example repository.)

This example demonstrates how you can use the CONDDIST statement in the QUANTREG procedure to perform counterfactual analysis. The data are from the NHANES I Epidemiologic Follow-Up Study (NHEFS) in Hernán and Robins (2018). NHEFS collected medical and behavioral information during an initial physical examination, and again at follow-up interviews approximately one decade later. Chapter 39, "The CAUSALTRT Procedure," uses these same data to analyze the average treatment effect.

The following DATA step creates the input data set SmokingWeight:

```
data SmokingWeight;
   input
   Sex Age Race Education Exercise BaseWeight Weight Change Activity
   YearsSmoke PerDay Quit;
   label
      Activity   = 'Level of daily activity with values in {0,1,2}'
      Age        = 'Age in 1971'
      BaseWeight = 'Weight in kilograms at the 1971 interview'
      Change     = 'Body Weight Change'
      Education  = 'Level of education with values in {0,1,2,3,4}'
      Exercise   = 'Level of recreational exercise with values in {0,1,2}'
      Perday     = 'Number of cigarettes smoked per day in 1971'
```

```
      Quit       = '1 for quitting smoking; 0 otherwise'
      Race       = '0 for white; 1 otherwise'
      Sex        = '0 for male; 1 for female'
      Weight     = 'Weight in kilograms at the follow-up interview'
      YearsSmoke = 'Number of years an individual has smoked';
   datalines;
   0 42 1 1 2 79.04 68.95 -10.09 0 29 30 0
   0 36 0 2 0 58.63 61.23 2.60 0 24 20 0
   1 56 1 2 2 56.81 66.22 9.41 0 26 20 0

   ... more lines ...

   1 51 0 3 0 62.71 . . 0 30 40 0
   0 68 0 1 1 52.39 57.15 4.76 1 46 15 0
   0 26 0 . 0 86.75 87.54 0.79 0 9 20 0
   0 29 0 2 1 90.83 106.59 15.76 1 14 30 1
;
```

The following statements split the SmokingWeight data set into two data subsets:

```
data QuitSmoking KeepSmoking;
   set SmokingWeight;
   if Quit=0 then output KeepSmoking;
   else if
      Quit=1 then output QuitSmoking;
run;
```

The data set KeepSmoking contains all the keep-smoking observations, and the data set QuitSmoking contains all the quit-smoking observations.

The goal of this study is to estimate the effect of quitting smoking on people's changes in body weight. Because NHEFS did not assign subjects to the quit-smoking group or the keep-smoking group, these data are observational but not experimental. To estimate the effect of quitting smoking, this study needs to estimate the counterfactual distribution of body weight changes for the quit-smoking group by assuming that the members of the quit-smoking group were continuing to smoke between their two interviews. With the further assumption that this counterfactual distribution follows the same conditional distribution model of body weight changes for the observed keep-smoking group in the KeepSmoking data set, quantile regression can estimate this counterfactual distribution by taking the following steps:

1. Build a quantile process regression model by using only the keep-smoking data.
2. Predict the quantile process for each observation in the quit-smoking data, which is a sequence of quantile predictions on a prespecified quantile-level grid.
3. Create a univariate sample by pooling together all the quantile predictions for all the observations in the quit-smoking data.
4. Estimate the counterfactual distribution of body weight changes on the univariate sample.
5. Repeat Steps 1–4 by using a bootstrap resampling method to perform statistical inferences for the relevant estimates.

Because these steps do not rely on the observed body weight changes of the quit-smoking data, given the explanatory covariates of the quit-smoking data, the fitted counterfactual marginal distribution is also

independent of the observed marginal distribution of the body weight changes for the quit-smoking group. For more information about this estimation method, see the sections "Fitted Marginal Cumulative Distribution Functions" on page 8589 and "Marginal Distribution Analysis Using the Bootstrap Resampling Method" on page 8591.

By using the fitted counterfactual distribution, this study can evaluate the effect of quitting smoking on the body weight changes by comparing the following three distributions:

TrainObs: the observed marginal distribution for the keep-smoking group

TestObs: the observed marginal distribution for the quit-smoking group

TestFit: the fitted counterfactual marginal distribution for the quit-smoking group that is fitted by using the model built on the keep-smoking observations

For more information about the observed and fitted marginal distributions, see the section "Estimating Probability Functions by Using the CONDDIST Statement" on page 8587.

The following statements invoke the PROC QUANTREG and CONDDIST statements:

```
ods graphics on;
ods select CondDistEst CmpTrainObs CmpTestObs
           DensityEstInfo CmpMarginalDist
           cdfplot pdfplot ppplot mcdfplot;

proc quantreg data=KeepSmoking ci=none;
   class sex Race Education Activity Exercise;
   model Change = Sex Age Education Activity Exercise YearsSmoke
                  PerDay / quantlev=fqpr(n=255 uao) seed=123;
   conddist mwu pdf=kde(l=-20 u=30) mcdf(nrep=100) plot(sg)=all
            testdata(mwu)=QuitSmoking;
run;
```

The DATA=KeepSmoking option in the PROC QUANTREG statement specifies the data set KeepSmoking as the training data set. The MODEL statement specifies the variable Change as the response variable and specifies a set of other variables as the explanatory variables. The QUANTLEV=FQPR(N=255) option in the MODEL statement specifies the fast quantile process regression method to build the quantile process regression model on a grid of 255 quantile levels, $\{0.5/255, 1.5/255, \ldots, 254.5/255\}$. For more information about the fast quantile process regression method, see the section "Fast Quantile Process Regression" on page 8580.

The MWU option in the CONDDIST statement performs the Mann-Whitney U test on the CDF sample of the TrainObs distribution. The PDF=KDE(L=–20 U=30) option specifies the kernel density estimation method to estimate probability density functions for the three distributions within the quantile range $(-20, 30)$. The PLOT(SG)=ALL option produces the CDF plot, the marginal CDF plot, the PDF plot, and the PP plot, where the SG suboption adds the light gray grid lines to the plots. The TESTDATA(MWU)=QuitSmoking option specifies the data set QuitSmoking as the test data set and performs the Mann-Whitney U test on the CDF sample of the TestObs distribution. The MCDF(NREP=100) option performs the marginal distributions analysis by using the bootstrap method, where the NREP=100 suboption specifies the number of iterations for the bootstrap process.

Output 103.6.1 shows the conditional distribution estimates.

Output 103.6.1 Conditional Distribution Estimates

The QUANTREG Procedure
Conditional Distribution Analysis 1

Conditional Distribution Estimates						
				Quantile Level		
Data	Type	Label	Response Value	Regression	Sample	Regression Density
Training	Observed	TrainObs	1.984359	0.4902	0.4892	0.065704
Test	Observed	TestObs	4.524913	0.5270	0.6573	0.053543
Test	Fit and Pooled	TestFit	4.524913	0.6927	0.6573	0.056120

Each row of this table corresponds to the pair of a predicted CDF sample and an assigned or observed response value. The Label column displays the labels of the CDF samples. The Data column indicates whether the CDF samples are associated with the keep-smoking data valued as "Training" or with the quit-smoking data valued as "Test." The Type column shows the types of CDF samples, where the "Observed" type is for the observed marginal distributions and the "Fit and Pooled" type is for the fitted counterfactual marginal distribution. The response values for these three CDF samples are all assigned averages. The Response Value column shows that the average weight change for the keep-smoking data is about 1.984 kilograms and the average weight change for the quit-smoking data is about 4.525 kilograms. If you specify the OBS= or TESTDATA(SHOWOBS)= option (or both) in the CONDDIST statement, the Response Value column shows the observed body weight changes for the relevant observations. The two Quantile Level columns, Regression and Sample, show the regression quantile levels and sample quantile levels for the response values. For more information about these estimated quantile levels, see the section "Regression Quantile Level and Sample Quantile Level" on page 8593.

Output 103.6.2 shows distribution comparisons to the TrainObs distribution from the TestObs distribution in the first row and from the TestFit distribution in the second row by using the Mann-Whitney U test.

Output 103.6.2 Distribution Comparisons to Observed Marginal Distribution of the Training Data Response

Distribution Comparisons to TrainObs				
Data	Type	Label	z Value	Pr > |z|
Test	Observed	TestObs	3.56	0.0004
Test	Fit and Pooled	TestFit	-1.05	0.2936

In the first row, the p-value 0.0004 rejects the hypothesis that the observed marginal distribution of the body weight changes for the quit-smoking group (labeled TestObs) is the same as that for the keep-smoking group (labeled TrainObs); and the z-value $3.571 > 0$ implies that the quit-smoking group gains significantly more body weight than the keep-smoking group. In the second row, the p-value 0.2905 does not have enough evidence to reject the hypothesis that the fitted counterfactual marginal distribution of the body weight changes for the quit-smoking group (labeled TestFit) is the same as that for the keep-smoking group (or the TrainObs distribution). The z-value $-1.0571 < 0$ implies that the quit-smoking group might even gain less body weight than the keep-smoking group if the quit-smoking group were not quitting smoking, although not enough evidence is available for this implication because the p-value 0.2905 is large.

Output 103.6.3 shows distribution comparisons to the TestObs distribution from the TestFit distribution by using the Mann-Whitney U test.

Output 103.6.3 Distribution Comparisons to Observed Marginal Distribution of the Test Data Response

Distribution Comparisons to TestObs				
Data	**Type**	**Label**	**z Value**	**Pr > |z|**
Test	**Fit and Pooled**	**TestFit**	-4.47	<.0001

The p-value 0 strongly rejects the hypothesis that quitting smoking has no impact on the distribution of body weight changes for the quit-smoking group. The z-value $-4.4813 < 0$ implies that the quit-smoking group gains significantly less body weight, if they were not actually quitting smoking, in comparison to this group in reality.

Output 103.6.4 shows the CDF samples for the TrainObs, TestObs, and TestFit marginal distributions.

Output 103.6.4 Cumulative Distribution Functions

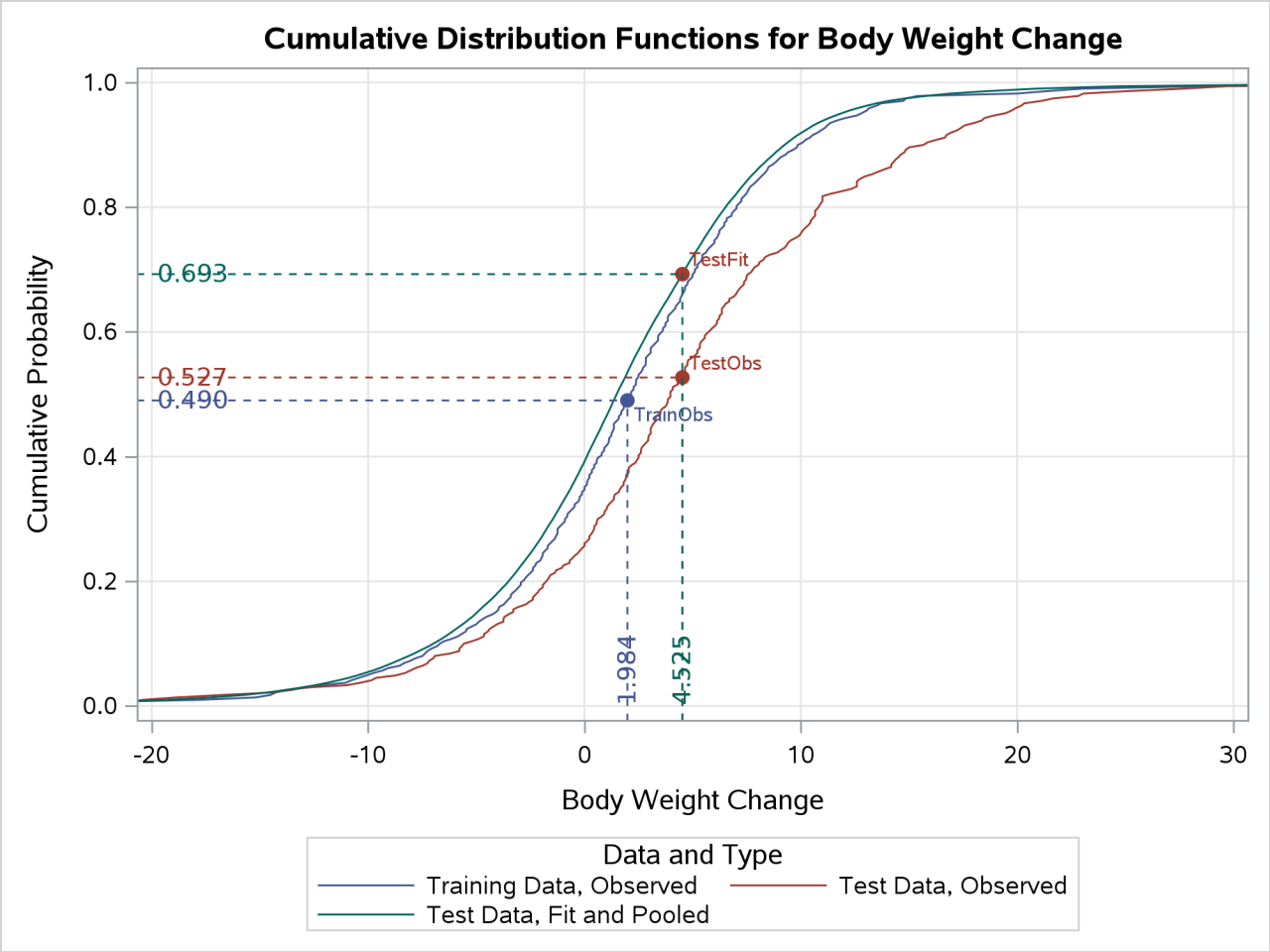

You can see that the CDF for the counterfactual quit-smoking group (green TestFit curve) is stochastically smaller than the observed marginal distributions for both the keep-smoking group (blue TrainObs curve) and the real quit-smoking group (red TestObs curve). The average weight change for the real quit-smoking group 4.525 has a percentage value of 52.7% according to the TestObs distribution but 69.4% according to the counterfactual TestFit distribution.

Output 103.6.5 shows that the CONDDIST statement uses the kernel density estimation (KDE) method to estimate PDFs.

Output 103.6.5 Density Estimation Information

Density Estimates Information	
Estimation Method	Kernel
Kernel Type	Normal
Grid Size	256
Bandwidth Parameter	0.7852
Bounds Adjustment Type	Trim
Lower Bound	-20.0000
Upper Bound	30.0000

Each PDF is built on a quantile grid of 255 equally spaced points. Because of the PDF=KDE(L=-20 U=30) option in the SAS program, the KDE method sets the PDF values to zero for the quantile points that are either smaller than L=–20 or greater than U=30. For more information about the KDE method, see the section "Probability Density Functions" on page 8593.

Output 103.6.6 shows the PDFs for the TrainObs, TestObs, and TestFit distributions.

Output 103.6.6 Probability Density Functions

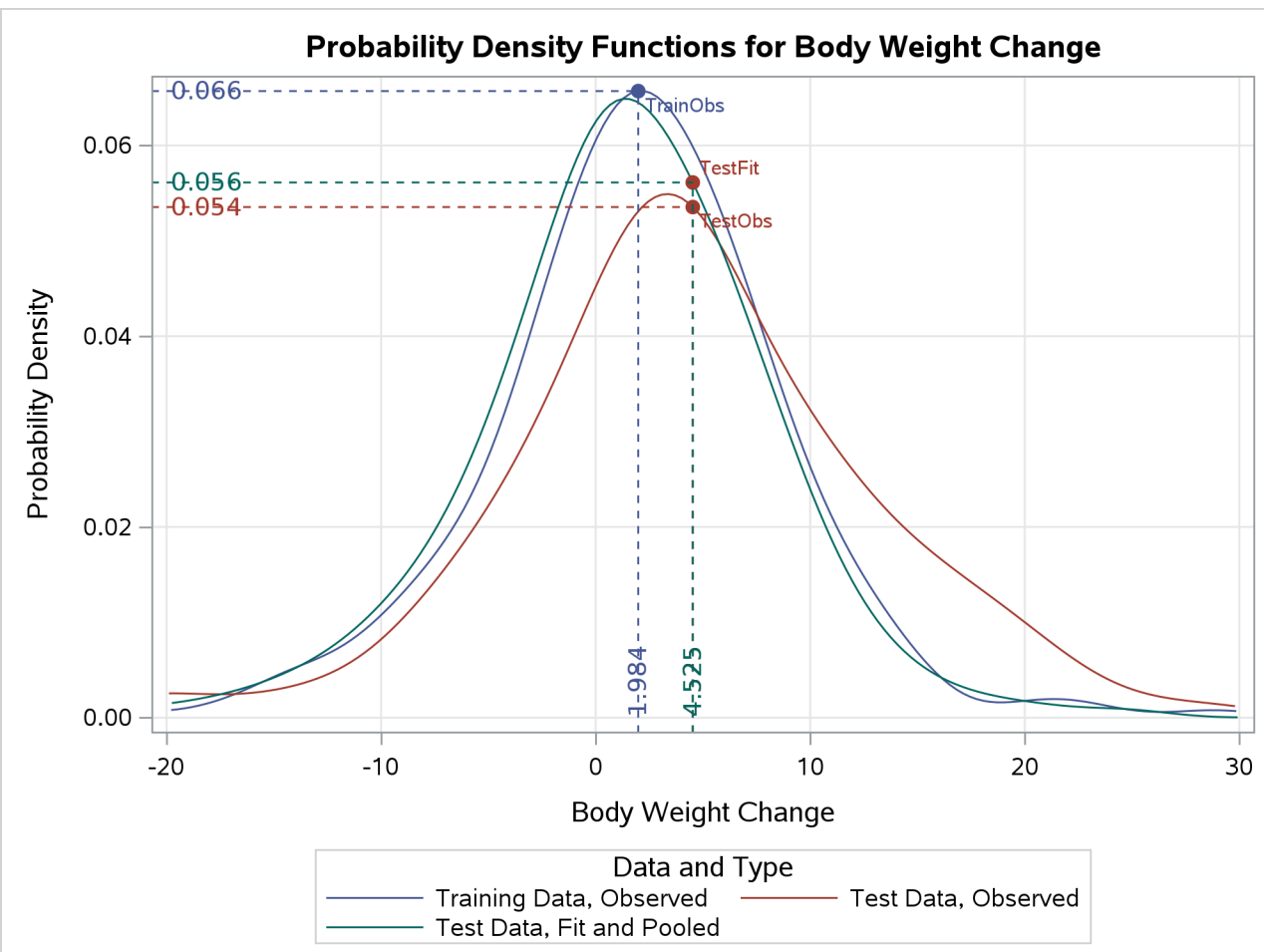

You can see that the TrainObs and TestFit PDFs are likely to be following the normal distribution. However, the TestObs PDF is right-skewed. The skewness might occur because the quit-smoking time, which is not available in the data set, introduces extra deviance to the right arm of the TestObs distribution.

Output 103.6.7 is similar to Output 103.6.4 and also shows the CDFs for the TrainObs, TestObs, and TestFit distributions. However, Output 103.6.7 uses the weighted bootstrap method to estimate the CDFs and their confidence bands. For more information, see the section "Marginal Distribution Analysis Using the Bootstrap Resampling Method" on page 8591.

Output 103.6.7 Marginal Cumulative Distribution Functions

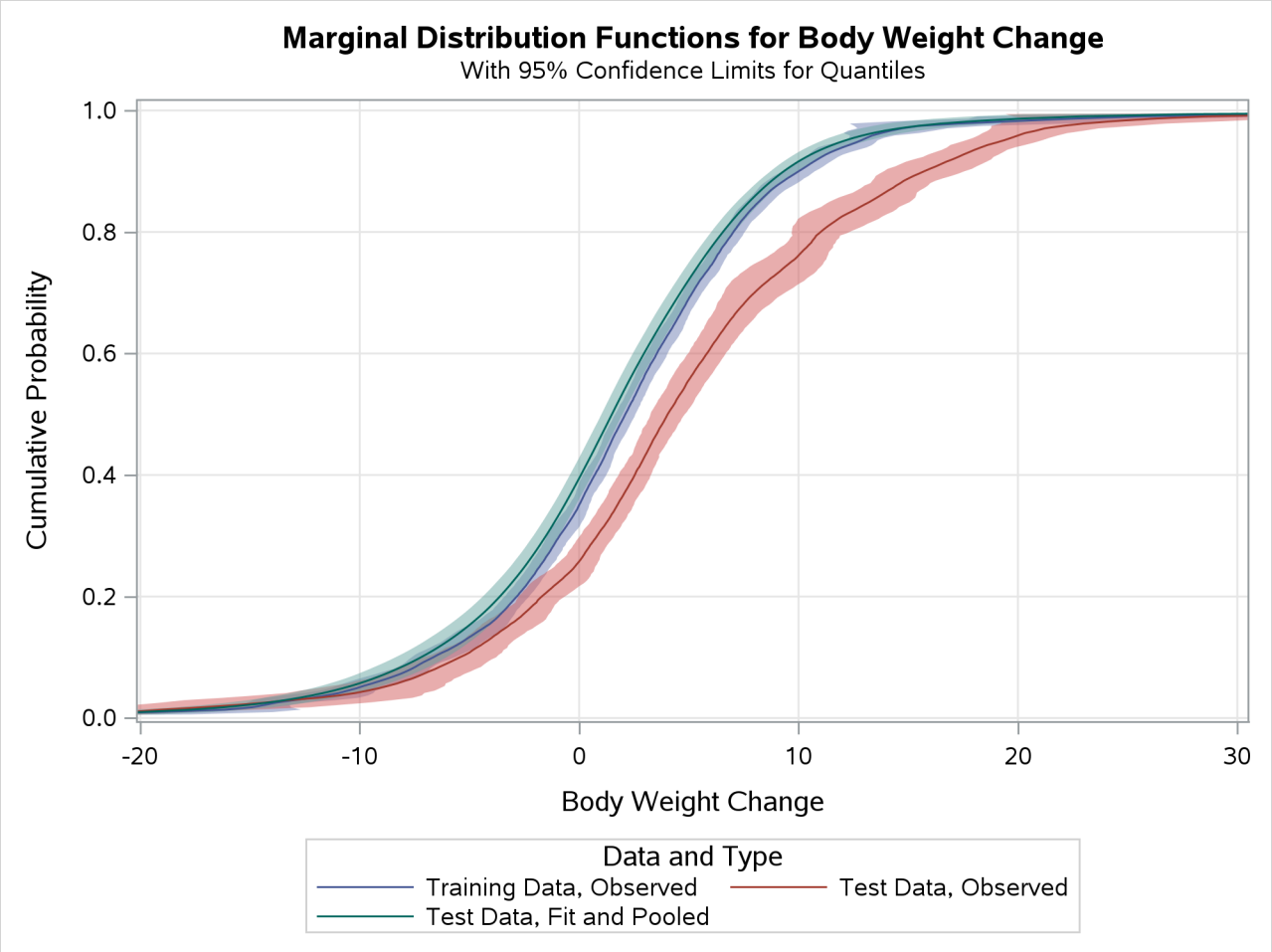

Output 103.6.8 shows the marginal distribution comparisons among the TrainObs, TestObs, and TestFit CDFs. Compared with the Z-values in Output 103.6.2 and Output 103.6.3, the Z-values of the Mann-Whitney U estimates in Output 103.6.8 are more significant, because Output 103.6.2 and Output 103.6.3 limit the number of quantile levels and display conservative estimates. The overall mean difference, TestObs–TrainObs, shows that, in terms of mean difference, the quit-smoking group gains 2.6239 kilograms more body weight than the keep-smoking group, where the smoking effect, –(TestFit–TestObs), contributes 3.2304 kilograms and the subject selection bias, TestFit–TrainObs, contributes –0.6066 kilograms. The overall median difference, TestObs–TrainObs, shows that, in terms of median difference, the quit-smoking group gains 1.9620 kilograms more body weight than the keep-smoking group, where the smoking effect, –(TestFit–TestObs), contributes 2.5676 kilograms and the subject selection bias, TestFit–TrainObs, contributes –0.6056 kilograms.

Output 103.6.8 Marginal Distribution Comparisons

Comparisons of Marginal Distributions						
CDF						
Statistic	Comparison	Base	Estimate	Standard Error	z Value	Pr > |z|
Mann-Whitney U	TestObs	TrainObs	5.5902	1.0798	5.18	<.0001
	TestFit	TrainObs	-1.6701	0.3664	-4.56	<.0001
	TestFit	TestObs	-5.7545	0.8479	-6.79	<.0001
Mean Difference	TestObs	TrainObs	2.5834	0.5120	5.05	<.0001
	TestFit	TrainObs	-0.6444	0.1457	-4.42	<.0001
	TestFit	TestObs	-3.2278	0.5070	-6.37	<.0001
Median Difference	TestObs	TrainObs	1.9319	0.4981	3.88	0.0001
	TestFit	TrainObs	-0.6084	0.1728	-3.52	0.0004
	TestFit	TestObs	-2.5403	0.5004	-5.08	<.0001

References

Abreveya, J. (2001). "The Effects of Demographics and Maternal Behavior on the Distribution of Birth Outcomes." *Journal of Economics* 26:247–257.

Barro, R., and Lee, J.-W. (1994). "Data Set for a Panel of 138 Countries." Discussion paper, National Bureau of Econometric Research. http://admin.nber.org/pub/barro.lee/readme.txt.

Barrodale, I., and Roberts, F. D. K. (1973). "An Improved Algorithm for Discrete l_1 Linear Approximation." *SIAM Journal on Numerical Analysis* 10:839–848.

Bassett, G. W., and Koenker, R. (1982). "An Empirical Quantile Function for Linear Models with iid Errors." *Journal of the American Statistical Association* 77:401–415.

Cade, B. S., and Noon, B. R. (2003). "A Gentle Introduction to Quantile Regression for Ecologists." *Frontiers in Ecology and the Environment* 1:412–420.

Chen, C. (2004). "An Adaptive Algorithm for Quantile Regression." In *Theory and Applications of Recent Robust Methods*, edited by M. Hubert, G. Pison, A. Struyf, and S. V. Aels, 39–48. Basel: Birkhäuser.

Chen, C. (2005). "Growth Charts of Body Mass Index (BMI) with Quantile Regression." In *Proceedings of 2005 International Conference on Algorithmic Mathematics and Computer Science*, edited by H. R. Arabnia and I. A. Ajwa, 114–120. Bogart, GA: CSREA Press.

Chen, C. (2007). "A Finite Smoothing Algorithm for Quantile Regression." *Journal of Computational and Graphical Statistics* 16:136–164.

Chock, D. P., Winkler, S. L., and Chen, C. (2000). "A Study of the Association between Daily Mortality and Ambient Air Pollutant Concentrations in Pittsburgh, Pennsylvania." *Journal of the Air and Waste Management Association* 50:1481–1500.

Dunham, J. B., Cade, B. S., and Terrell, J. W. (2002). "Influences of Spatial and Temporal Variation on Fish-Habitat Relationships Defined by Regression Quantiles." *Transactions of the American Fisheries Society* 131:86–98.

Gutenbrunner, C., and Jureckova, J. (1992). "Regression Rank Scores and Regression Quantiles." *Annals of Statistics* 20:305–330.

Gutenbrunner, C., Jureckova, J., Koenker, R., and Portnoy, S. (1993). "Tests of Linear Hypotheses Based on Regression Rank Scores." *Journal of Nonparametric Statistics* 2:307–331.

He, X., and Hu, F. (2002). "Markov Chain Marginal Bootstrap." *Journal of the American Statistical Association* 97:783–795.

Hernán, M. A., and Robins, J. M. (2018). *Causal Inference*. Boca Raton, FL: Chapman & Hall/CRC. Forthcoming.

Huber, P. J. (1981). *Robust Statistics*. New York: John Wiley & Sons.

Jones, M. C., Marron, J. S., and Sheather, S. J. (1996). "A Brief Survey of Bandwidth Selection for Density Estimation." *Journal of the American Statistical Association* 91:401–407.

Karmarkar, N. (1984). "A New Polynomial-Time Algorithm for Linear Programming." *Combinatorica* 4:373–395.

Koenker, R. (1994). "Confidence Intervals for Regression Quantiles." In *Asymptotic Statistics*, edited by P. Mandl and M. Huskova, 349–359. New York: Springer-Verlag.

Koenker, R. (2005). *Quantile Regression*. New York: Cambridge University Press.

Koenker, R., and Bassett, G. W. (1978). "Regression Quantiles." *Econometrica* 46:33–50.

Koenker, R., and Bassett, G. W. (1982a). "Robust Tests for Heteroscedasticity Based on Regression Quantiles." *Econometrica* 50:43–61.

Koenker, R., and Bassett, G. W. (1982b). "Tests of Linear Hypotheses and l_1 Estimation." *Econometrica* 50:1577–1583.

Koenker, R., and d'Orey, V. (1994). "Remark AS R92: A Remark on Algorithm AS 229: Computing Dual Regression Quantiles and Regression Rank Scores." *Journal of the Royal Statistical Society, Series C* 43:410–414.

Koenker, R., and Hallock, K. (2001). "Quantile Regression." *Journal of Economic Perspectives* 15:143–156.

Koenker, R., and Machado, A. F. (1999). "Goodness of Fit and Related Inference Processes for Quantile Regression." *Journal of the American Statistical Association* 94:1296–1310.

Koenker, R., and Zhao, Q. (1994). "L-Estimation for Linear Heteroscedastic Models." *Journal of Nonparametric Statistics* 3:223–235.

Kuczmarski, R. J., Ogden, C. L., and Guo, S. S. (2002). "2000 CDC Growth Charts for the United States: Methods and Development." *Vital and Health Statistics* 11:1–190.

Lustig, I. J., Marsten, R. E., and Shanno, D. F. (1992). "On Implementing Mehrotra's Predictor-Corrector Interior-Point Method for Linear Programming." *SIAM Journal on Optimization* 2:435–449.

Madsen, K., and Nielsen, H. B. (1993). "A Finite Smoothing Algorithm for Linear L_1 Estimation." *SIAM Journal on Optimization* 3:223–235.

Parzen, M. I., Wei, L. J., and Ying, Z. (1994). "A Resampling Method Based on Pivotal Estimating Functions." *Biometrika* 81:341–350.

Portnoy, S., and Koenker, R. (1997). "The Gaussian Hare and the Laplacian Tortoise: Computation of Squared-Error vs. Absolute-Error Estimators." *Statistical Science* 12:279–300.

Praestgaard, J., and Wellner, J. A. (1993). "Exchangeably Weighted Bootstraps of the General Empirical Process." *Annals of Probability* 21:2053 – 2086. https://doi.org/10.1214/aop/1176989011.

Roos, C., Terlaky, T., and Vial, J. (1997). *Theory and Algorithms for Linear Optimization*. Chichester, UK: John Wiley & Sons.

Rousseeuw, P. J., and Van Driessen, K. (1999). "A Fast Algorithm for the Minimum Covariance Determinant Estimator." *Technometrics* 41:212–223.

Yao, Y. (2017). "Fast Quantile Process Regression." Paper presented at the 2017 International Conference on Robust Statistics, Wollongong, NSW, Australia.

Yu, K., Lu, Z., and Stander, J. (2003). "Quantile Regression: Application and Current Research Areas." *Journal of the Royal Statistical Society, Series D* 52:331–350.

Chapter 104
The QUANTSELECT Procedure

Contents

Overview: QUANTSELECT Procedure . . . **8638**
- Features . . . 8638

Getting Started: QUANTSELECT Procedure . . . **8639**

Syntax: QUANTSELECT Procedure . . . **8648**
- PROC QUANTSELECT Statement . . . 8649
- BY Statement . . . 8658
- CLASS Statement . . . 8658
- CODE Statement . . . 8662
- EFFECT Statement . . . 8662
- MODEL Statement . . . 8664
- OUTPUT Statement . . . 8672
- PARTITION Statement . . . 8673
- WEIGHT Statement . . . 8674

Details: QUANTSELECT Procedure . . . **8674**
- Quantile Regression . . . 8674
  - Quasi-Likelihood Information Criteria . . . 8674
  - Quasi-Likelihood Ratio Tests . . . 8675
  - Quantile Process Regression . . . 8677
  - Observation Quantile Level . . . 8678
  - Quantile Regression for Extremal Quantile Levels . . . 8678
- Effect Selection Methods . . . 8679
  - Full Model Fitted (NONE) . . . 8679
  - Forward Selection (FORWARD) . . . 8679
  - Backward Elimination (BACKWARD) . . . 8679
  - Stepwise Selection (STEPWISE) . . . 8679
  - LASSO Method (LASSO) . . . 8680
- Criteria Used in Model Selection Methods . . . 8680
- Macro Variables That Contain Selected Models . . . 8683
- Using Validation and Test Data . . . 8684
- Displayed Output . . . 8687
- ODS Table Names . . . 8691
- ODS Graphics . . . 8692

Example: QUANTSELECT Procedure . . . **8693**
- Example 104.1: Simulation Study . . . 8693
- Example 104.2: Econometric Growth Data . . . 8696
- Example 104.3: Pollution and Mortality . . . 8703

Example 104.4: Surface Fitting with Many Noisy Variables 8710
Example 104.5: Quantile Process Regression 8713
References **8719**

Overview: QUANTSELECT Procedure

Quantile regression, which was introduced by Koenker and Bassett (1978), is a modern method that models the effects of covariates on the conditional quantiles of a response variable. The QUANTSELECT procedure performs effect selection in the framework of quantile regression. A variety of effect selection methods are available, including greedy methods and penalty methods. The QUANTSELECT procedure offers extensive capabilities for customizing the effect selection processes with a variety of candidate selecting, effect-selection stopping, and final-model choosing criteria. PROC QUANTSELECT also provides graphical summaries for the effect selection processes.

The QUANTSELECT procedure compares most closely to the GLMSELECT and QUANTREG procedures. PROC GLMSELECT performs effect selection in the framework of general linear models. PROC QUANTREG supports a variety of estimation and inference methods for quantile regression but does not directly provide effect selection facilities. The QUANTSELECT procedure, as a counterpart of PROC GLMSELECT for quantile regression, fills this gap.

The QUANTSELECT procedure focuses on linear quantile models for univariate responses and offers great flexibility for and insight into the effect selection algorithm. The QUANTSELECT procedure inherits most of its syntax from PROC GLMSELECT and PROC QUANTREG. The QUANTSELECT procedure provides results that are similar to those of PROC GLMSELECT and PROC QUANTREG. These results (displayed tables, output data sets, and macro variables) make it easy to explore the selected models in PROC QUANTREG.

Features

The main features of the QUANTSELECT procedure are as follows:

- supports the following model specifications:
  - interaction (crossed) effects and nested effects
  - constructed effects such as regression splines
  - hierarchy among effects
  - partitioning of data into training, validation, and testing roles
- provides the following selection controls:
  - multiple methods for effect selection
  - selection for quantile process and single quantile levels
  - selection of individual or grouped effects
  - selection based on a variety of selection criteria
  - stopping rules based on a variety of model evaluation criteria
- produces the following display and output:

  - graphical representation of the selection process
  - output data sets that contain predicted values and residuals
  - an output data set that contains the design matrix
  - macro variables that contain selected effects

The QUANTSELECT procedure supports the following effect selection methods. These methods are explained in detail in the section "Effect Selection Methods" on page 8679.

- Forward selection starts with no effects or with forced-in effects in the model and adds more effects.
- Backward elimination starts with all effects in the model and deletes effects.
- Stepwise regression is similar to the forward selection method except that effects already in the model do not necessarily stay there.
- LASSO regression adds and deletes effects based on a constrained version of estimated check risk where the L1-norm of regression coefficients is penalized (Tibshirani 1996; Belloni and Chernozhukov 2011). Adaptive LASSO (Zou 2006; Wu and Liu 2009) is implemented as a special case of LASSO methods where the L1-norm of certain weighted regression coefficients is penalized. See the discussion in the section "LASSO Method (LASSO)" on page 8680 for additional details. The QUANTSELECT procedure uses LASSO methods only to determine the adding and dropping covariate effects at a step; a post-penalized model that is associated with the step is refitted without penalty, and the selection criteria and the parameter estimates are from the post-penalized model.

The QUANTSELECT procedure is intended primarily as an effect selection procedure and does not include regression diagnostics and hypothesis testing. The intention is that you use the QUANTSELECT procedure to select a model or a set of models, where each model contains a set of selected effects, and then you can further investigate these models by using PROC QUANTREG or other analytic tools.

Getting Started: QUANTSELECT Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/o-q/qrsgs.sas`.)

This example demonstrates how you can use the QUANTSELECT procedure to select covariate effects for quantile regression. The Sashelp.Baseball data set contains salary and performance information for Major League Baseball (MLB) players, excluding pitchers, who played at least one game in both the 1986 and 1987 seasons. The salaries (Time Inc. 1987) are for the 1987 season, and the performance measures are from 1986 (Reichler 1987).

The following step displays in Figure 104.1 the variables in the data set:

```
proc contents varnum data=sashelp.baseball;
   ods select position;
run;
```

Figure 104.1 Sashelp.Baseball Data Set

The CONTENTS Procedure

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	Name	Char	18	Player's Name
2	Team	Char	14	Team at the End of 1986
3	nAtBat	Num	8	Times at Bat in 1986
4	nHits	Num	8	Hits in 1986
5	nHome	Num	8	Home Runs in 1986
6	nRuns	Num	8	Runs in 1986
7	nRBI	Num	8	RBIs in 1986
8	nBB	Num	8	Walks in 1986
9	YrMajor	Num	8	Years in the Major Leagues
10	CrAtBat	Num	8	Career Times at Bat
11	CrHits	Num	8	Career Hits
12	CrHome	Num	8	Career Home Runs
13	CrRuns	Num	8	Career Runs
14	CrRbi	Num	8	Career RBIs
15	CrBB	Num	8	Career Walks
16	League	Char	8	League at the End of 1986
17	Division	Char	8	Division at the End of 1986
18	Position	Char	8	Position(s) in 1986
19	nOuts	Num	8	Put Outs in 1986
20	nAssts	Num	8	Assists in 1986
21	nError	Num	8	Errors in 1986
22	Salary	Num	8	1987 Salary in $ Thousands
23	Div	Char	16	League and Division
24	logSalary	Num	8	Log Salary

Suppose you want to investigate how the MLB players' salaries for the 1987 season depend on performance measures for the players' previous season and MLB careers. As a starting point for such a analysis, you can use the following statements to obtain a parsimonious conditional median model at $\tau = 0.5$:

```
proc quantselect data=sashelp.baseball;
   class Div;
   model Salary = nAtBat nHits nHome nRuns nRBI nBB yrMajor crAtBat
                  crHits crHome crRuns crRbi crBB nAssts nError nOuts
                  Div
         / selection=lasso(adaptive stop=aic choose=sbc sh=7);
run;
```

The SELECTION=LASSO(ADAPTIVE) option in the MODEL statement specifies the adaptive LASSO method (Zou 2006), which controls the effect selection process. The STOP=AIC option specifies that Akaike's information criterion (AIC) be used to determine the stopping condition. The CHOOSE=SBC option specifies that the Schwarz Bayesian information criterion (SBC) be used to determine the final selected model. The SH= option specifies the number of stop horizons, which requests that the selection process be stopped whenever the STOP= criterion values at step $s+1, \ldots, s+\mathrm{SH}$ are worse than those for step s for some $s \in \{0, 1, \ldots\}$.

Figure 104.2 shows the "Model Information" table, which indicates the effect selection settings. You can see that the default quantile type is single level, so this effect selection is effective only for $\tau = 0.5$.

Figure 104.2 Model Information

The QUANTSELECT Procedure

Model Information	
Data Set	SASHELP.BASEBALL
Dependent Variable	Salary
Selection Method	Adaptive LASSO
Quantile Type	Single Level
Stop Criterion	AIC
Choose Criterion	SBC

Figure 104.3 summarizes the effect selection process, which starts with an intercept-only model at step 0. At step 1, the effect that corresponds to the career runs is added to the model that reduced the AIC value from 2691.6511 to 2510.7297. You can see that step 10 has the minimum AIC and that step 7 has the minimum SBC. Common sense also tells you that the SBC favors a smaller model than the AIC.

Figure 104.3 Selection Summary

The QUANTSELECT Procedure
Quantile Level = 0.5

Selection Summary					
Step	Effect Entered	Effect Removed	Number Effects In	AIC	SBC
0	Intercept		1	2691.6511	2695.2232
1	CrRuns		2	2510.7297	2517.8740
2	nHits		3	2470.4807	2481.1971
3	CrHome		4	2463.5953	2477.8839
4	nBB		5	2463.7806	2481.6414
5	nOuts		6	2455.6212	2477.0541
6	Div AW		7	2451.4609	2476.4660
7	nAtBat		8	2445.0446	2473.6218*
8	CrBB		9	2445.5432	2477.6926
9	nHome		10	2443.4818	2479.2033
10	nRuns		11	2442.6036*	2481.8973
11	Div NE		12	2444.2409	2487.1067
12	CrAtBat		13	2444.5049	2490.9429
13		Div NE	12	2442.8387	2485.7046
14	YrMajor		13	2443.5374	2489.9754
15	nError		14	2445.2085	2495.2187
16	Div NE		15	2446.4042	2499.9865
* Optimal Value Of Criterion					

Figure 104.4 shows that the selection process stopped at a local minimum of the STOP= criterion, which is step 10. According to the SH=7 option, the effect selection process is stopped at step 10 because all the AIC values for step 11 through step 17 are no less than the AIC at step 10. Step 17 is ignored in the selection

summary table because it is the last step.

Figure 104.4 Stop Reason

Selection stopped at a local minimum of the AIC criterion.

Figure 104.5 shows how the final selected model is determined. CHOOSE=SBC is specified in this example, so the model at step 7 is chosen as the final selected model.

Figure 104.5 Selection Reason

The model at step 7 is selected where SBC is 2473.622.

Figure 104.6 shows the final selected effects and Figure 104.7 shows the parameter estimates for the final selected model.

Figure 104.6 Selected Effects

Selected Effects:	Intercept nAtBat nHits nBB CrHome CrRuns nOuts Div AW

Figure 104.7 Parameter Estimates

Parameter Estimates			
Parameter	**DF**	**Estimate**	**Standardized Estimate**
Intercept	1	-18.187539	0
nAtBat	1	-1.582714	-0.500417
nHits	1	7.044354	0.686968
nBB	1	2.053726	0.097911
CrHome	1	1.429926	0.272726
CrRuns	1	0.425955	0.316167
nOuts	1	0.282803	0.175489
Div AW	1	-57.671778	-0.056862

Quantile regression can fit a conditional quantile model at any quantile level $\tau \in (0, 1)$, so it can describe the entire distribution of a response variable conditional on covariate effects. To further investigate the effects that might affect the MLB players' salaries, you can also conduct effect selection at $\tau = 0.1$ and $\tau = 0.9$, which correspond to low-end salaries and high-end salaries respectively. The following statements use the same selection settings that are used in the previous program:

```
proc quantselect data=sashelp.baseball;
   class Div;
   model Salary = nAtBat nHits nHome nRuns nRBI nBB yrMajor crAtBat
                  crHits crHome crRuns crRbi crBB nAssts nError nOuts
                  Div
         / quantiles=0.1 0.9 selection=lasso(adaptive stop=aic choose=sbc sh=7);
run;
```

Figure 104.8 shows the effect selection summary with $\tau = 0.1$.

Figure 104.8 Selection Summary: $\tau = 0.1$

The QUANTSELECT Procedure
Quantile Level = 0.1

Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	AIC	SBC
0	Intercept		1	2008.3489	2011.9211
1	CrRuns		2	1918.7675	1925.9118
2	nHits		3	1897.2425	1907.9590*
3	YrMajor		4	1897.2476	1911.5362
4	CrBB		5	1896.1765	1914.0373
5	nBB		6	1894.1257	1915.5587
6	CrHome		7	1895.6765	1920.6816
7	nAtBat		8	1890.4051	1918.9824
8	nHome		9	1891.3527	1923.5020
9	Div NE		10	1891.7566	1927.4781
10	nRBI		11	1893.7319	1933.0256
11	CrAtBat		12	1893.9432	1936.8090
12		nRBI	11	1891.9716	1931.2653
13	nAssts		12	1888.6870	1931.5529
14	nRBI		13	1890.5300	1936.9680
15	Div AE		14	1889.4234	1939.4336
16		nRBI	13	1887.6644*	1934.1024
17	CrRbi		14	1888.0966	1938.1068
18	Div AW		15	1890.0322	1943.6145
19	nError		16	1891.7949	1948.9494
20	nRuns		17	1893.2801	1954.0067
21	nRBI		18	1894.7805	1959.0793
22	CrHits		19	1896.6868	1964.5578

* Optimal Value Of Criterion

Figure 104.9 shows the parameter estimates for the final selected model with $\tau = 0.1$. You can see from Figure 104.9 that low-end salaries for MLB players depend mainly on career runs and hits in 1986.

Figure 104.9 Parameter Estimates: $\tau = 0.1$

The QUANTSELECT Procedure
Quantile Level = 0.1

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate
Intercept	1	-4.397043	0
nHits	1	0.878564	0.085678
CrRuns	1	0.327350	0.242977

Figure 104.10 shows the effect selection summary with $\tau = 0.9$.

Figure 104.10 Selection Summary: $\tau = 0.9$

The QUANTSELECT Procedure
Quantile Level = 0.9

Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	AIC	SBC
0	Intercept		1	2436.7289	2440.3011
1	CrHits		2	2197.4349	2204.5792
2	CrRbi		3	2183.6148	2194.3313
3	nHits		4	2113.2757	2127.5643
4		CrRbi	3	2127.8632	2138.5797
5	CrRbi		4	2113.2757	2127.5643
6		CrRbi	3	2127.8632	2138.5797
7	CrRbi		4	2113.2757	2127.5643
8	CrHome		5	2099.2203	2117.0811
9		CrRbi	4	2099.3891	2113.6777
10	CrRbi		5	2099.2203	2117.0811
11		CrRbi	4	2099.3891	2113.6777
12	nOuts		5	2067.1926	2085.0533
13	Div AW		6	2048.2393	2069.6723
14	CrRuns		7	2028.8040	2053.8090
15	nAtBat		8	2012.8195	2041.3968
16		CrHits	7	2017.0290	2042.0341
17	CrRbi		8	2009.3551	2037.9324
18	CrAtBat		9	2011.2415	2043.3908
19		CrRbi	8	2011.4053	2039.9825
20	CrRbi		9	2011.2415	2043.3908
21		CrAtBat	8	2009.3551	2037.9324
22	CrAtBat		9	2011.2415	2043.3908
23	nBB		10	2004.5033	2040.2249
24		CrAtBat	9	2003.1023	2035.2517
25	CrAtBat		10	2004.5033	2040.2249
26		CrAtBat	9	2003.1023	2035.2517*
27	CrAtBat		10	2004.5033	2040.2249
28	nError		11	2004.2230	2043.5167
29	CrHits		12	2003.0544	2045.9203
30	Div NE		13	2001.9603	2048.3983
31	Div AE		14	2001.8349*	2051.8451
32	nRuns		15	2003.5961	2057.1784
33	nHome		16	2004.2721	2061.4266
34	nRBI		17	2006.0023	2066.7289
35	YrMajor		18	2007.9975	2072.2963
36	CrBB		19	2009.9514	2077.8223
37	nAssts		20	2011.9095	2083.3525

* Optimal Value Of Criterion

Figure 104.11 shows the parameter estimates for the final selected model with $\tau = 0.9$.

Figure 104.11 Parameter Estimates: $\tau = 0.9$

Parameter Estimates			
Parameter	DF	Estimate	Standardized Estimate
Intercept	1	92.893875	0
nAtBat	1	-1.858170	-0.587509
nHits	1	8.155573	0.795335
nBB	1	3.392794	0.161751
CrHome	1	3.191472	0.608700
CrRuns	1	1.394317	1.034939
CrRbi	1	-0.913371	-0.664951
nOuts	1	0.437241	0.271323
Div AW	1	-167.110005	-0.164764

To visually illustrate how the model evolves through the selection process, the QUANTSELECT procedure provides the coefficient plot, the average check loss plot, and several criterion plots in either packed or unpacked forms. You can request these plots by using the PLOTS= option. The following statements request all the plots for the baseball data at $\tau = 0.1$; they also use the STOP=AIC criterion, the CHOOSE=SBC criterion, and the SH=7 option:

```
ods graphics on;
proc quantselect data=sashelp.baseball plots=all;
   class Div;
   model Salary = nAtBat nHits nHome nRuns nRBI nBB yrMajor crAtBat
                  crHits crHome crRuns crRbi crBB nAssts nError nOuts
                  Div
         / quantiles=0.1 selection=lasso(adaptive stop=aic choose=sbc sh=7);
run;
```

Figure 104.12 shows the progression of the parameter estimates as the selection process proceeds.

Figure 104.12 Coefficient Panel: $\tau = 0.1$

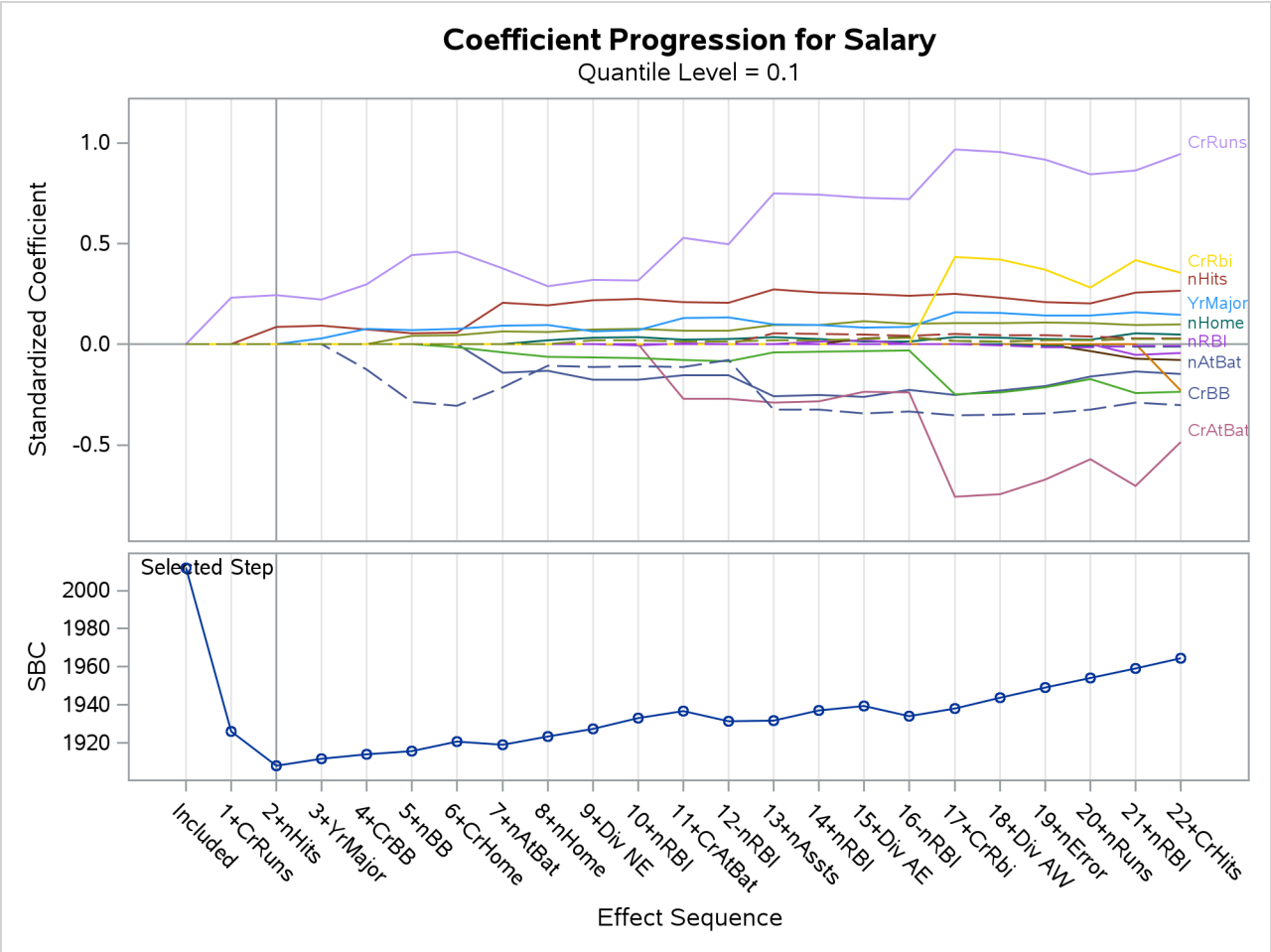

Figure 104.13 shows the progression of the average check losses as the selection process proceeds.

Figure 104.13 Average Check Loss Plot: $\tau = 0.1$

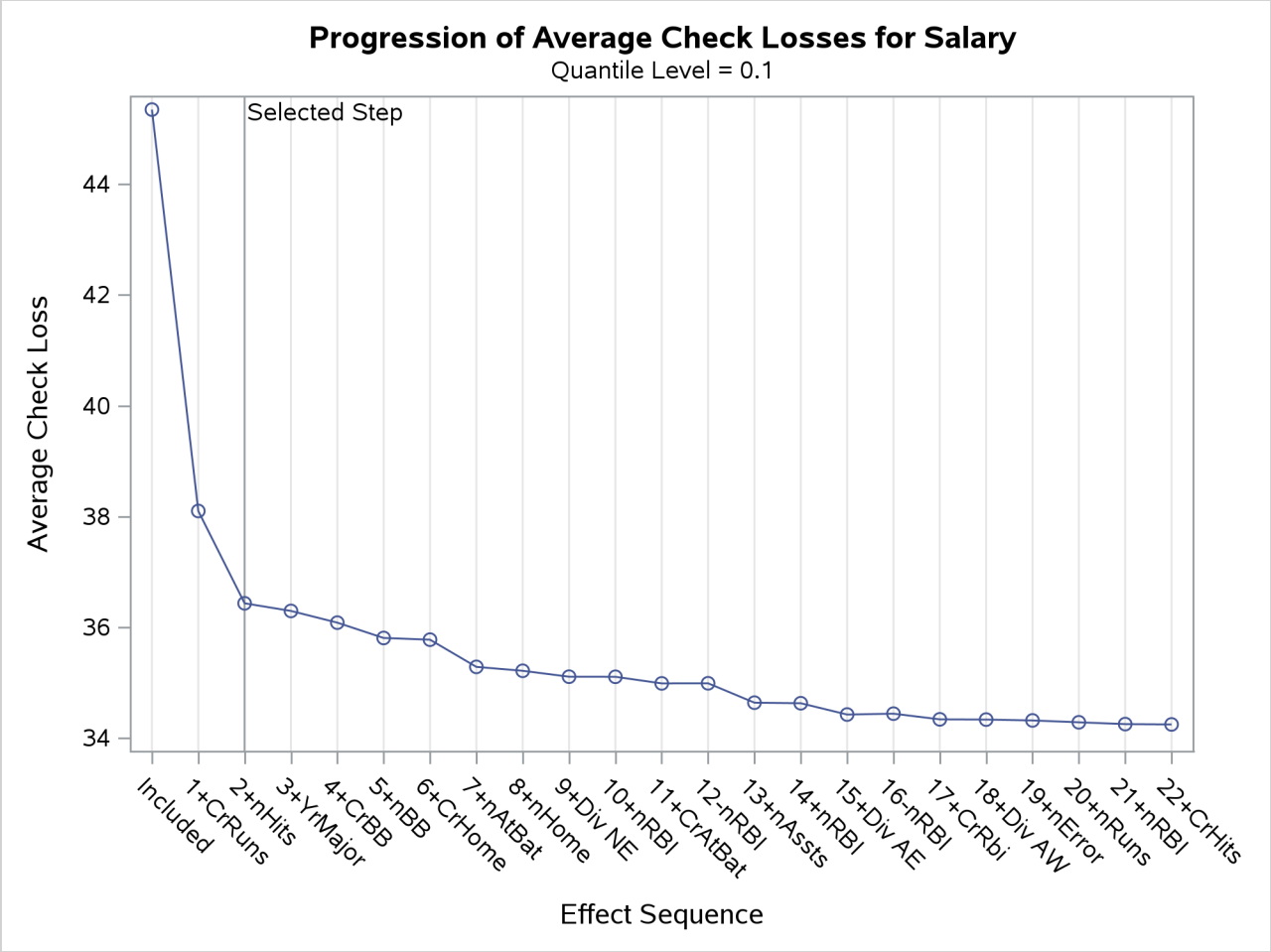

Figure 104.14 shows the progression of four effect selection criteria as the selection process proceeds.

Figure 104.14 Criterion Panel: $\tau = 0.1$

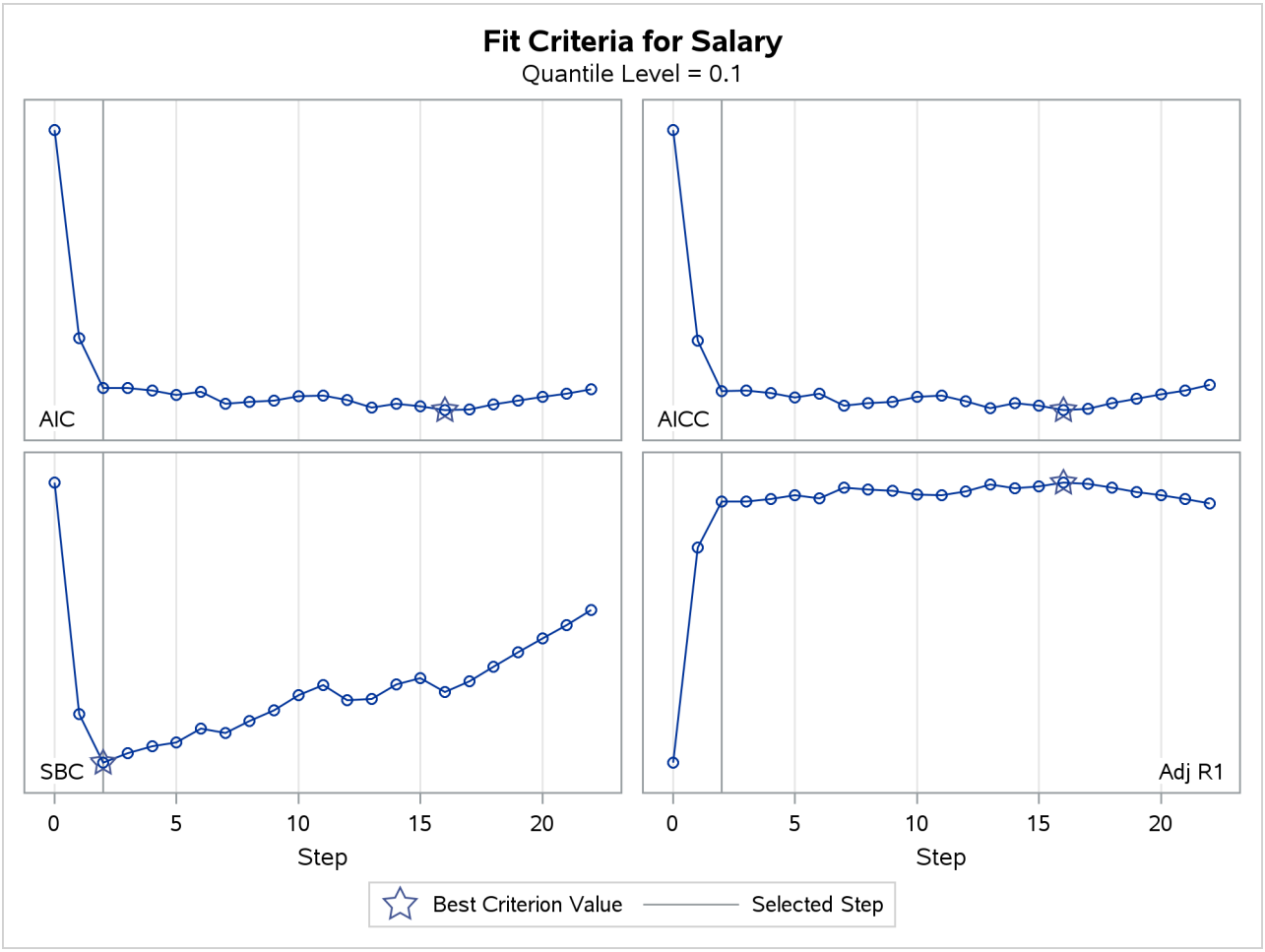

Syntax: QUANTSELECT Procedure

The following statements are available in PROC QUANTSELECT:

PROC QUANTSELECT < *options* > **;**
  BY *variables* **;**
  CLASS *variable* < **(***v-options***)** > < *variable* < **(***v-options . . .* **)** > > < / *v-options* >< *options* > **;**
  CODE < *options* > **;**
  EFFECT *name* **=** *effect-type* **(***variables* < / *options* > **) ;**
  MODEL *variable* **=** < *effects* > < / *options* > **;**
  OUTPUT < **OUT=***SAS-data-set* > < *keyword* < **=***name* > > < . . . *keyword*< **=***name* > > **;**
  PARTITION < *options* > **;**
  WEIGHT *variable* **;**

The PROC QUANTSELECT statement invokes the procedure. All statements other than the MODEL statement are optional. CLASS and EFFECT statements, if present, must precede the MODEL statement.

PROC QUANTSELECT Statement

PROC QUANTSELECT < *options* > **;**

Table 104.1 lists the *options* available in the PROC QUANTSELECT statement.

Table 104.1 PROC QUANTSELECT Statement Options

option	**Description**
Data Set Options	
DATA=	Names a data set to use for the regression
MAXMACRO=	Sets the maximum number of macro variables to produce
TESTDATA=	Names a data set that contains test data
VALDATA=	Names a data set that contains validation data
ODS Graphics Options	
PLOTS=	Produces ODS Graphics displays
Other Options	
ALGORITHM=	Specifies an algorithm for estimating the regression parameters
NAMELEN=	Specifies the maximum length of effect names in tables and output data sets
NOPRINT	Suppresses displayed output (including plots)
OUTDESIGN=	Names a data set that contains the design matrix
PARMLABELSTYLE=	Sets the style of parameter names and labels for nested and crossed effects
SEED=	Sets the seed used for pseudorandom number generation

You can specify the following *options* (shown in alphabetical order) in the PROC QUANTSELECT statement.

ALGORITHM=SIMPLEX | SMOOTH

specifies either the simplex algorithm (ALGORITHM=SIMPLEX) or the smoothing algorithm (ALGORITHM=SMOOTH) for estimating the regression parameters. The smoothing algorithm is computationally much more efficient than the simplex algorithm for fitting models on large data sets. You might consider specifying the ALGORITHM=SMOOTH if your DATA= data set contains more than 5,000 observations and more than 50 regressors. The smoothing algorithm does not support quantile process effect selection or the LASSO selection method. By default, ALGORITHM=SIMPLEX.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC QUANTSELECT. If the DATA= option is not specified, PROC QUANTSELECT uses the most recently created SAS data set. If the data set contains a variable named _ROLE_, then this variable is used to assign observations for training, validation, and testing roles. See the section "Using Validation and Test Data" on page 8684 for more information about using the _ROLE_ variable.

MAXMACRO=*n*

specifies the maximum number of macro variables with selected effects to create. By default, MAXMACRO=100.

PROC QUANTSELECT saves the list of selected effects in a macro variable, &_QRSIND. For example, suppose your input effect list consists of x1–x10. Then &_QRSIND would be set to x1 x3 x4 x10 if the first, third, fourth, and tenth effects were selected for the model. This list can be used in the MODEL statement of a subsequent procedure.

If you specify the OUTDESIGN= option in the PROC QUANTSELECT statement, then PROC QUANTSELECT saves the list of columns in the design matrix in a macro variable named &_QRSMOD.

With multiple quantile levels and BY processing, one macro variable is created for each combination of quantile level and BY group, and the macro variables are indexed by the BY-group number and the quantile-level index. You can use the MAXMACRO= option to either limit or increase the number of these macro variables when you are processing data sets with many combinations of quantile level and BY group.

With a single quantile level and no BY-group processing, PROC QUANTSELECT creates the macro variables shown in Table 104.2.

Table 104.2 Macro Variables Created for a Single Quantile Level and No BY Processing

Macro Variable Name	Contains
_QRSIND	Selected effects
_QRSIND1	Selected effects
_QRSINDT1	Selected effects
_QRSIND1T1	Selected effects
_QRSMOD	Selected design matrix columns
_QRSMOD1	Selected design matrix columns
_QRSMODT1	Selected design matrix columns
_QRSMOD1T1	Selected design matrix columns

With multiple quantile levels and BY-group processing, PROC QUANTSELECT creates the macro variables shown in Table 104.3.

Table 104.3 Macro Variables Created for a Multiple Quantile Levels and BY-Group Processing

Macro Variable Name	Contains
_QRSIND	Selected effects for quantile 1 and BY group 1
_QRSINDT1	Selected effects for quantile 1 and BY group 1
_QRSINDT2	Selected effects for quantile 2 and BY group 1
.	
.	
.	
_QRSIND1	Selected effects for quantile 1 and BY group 1
_QRSIND1T1	Selected effects for quantile 1 and BY group 1
_QRSIND1T2	Selected effects for quantile 2 and BY group 1
.	
.	
.	
_QRSIND2	Selected effects for quantile 1 and BY group 2
_QRSIND2T1	Selected effects for quantile 1 and BY group 2
_QRSIND2T2	Selected effects for quantile 2 and BY group 2
.	
.	
.	
_QRSIND*m*T*n*	Selected effects for quantile *n* and BY group *m*

If you specify the OUTDESIGN= option, PROC QUANTSELECT also creates the macro variables shown in Table 104.4.

Table 104.4 Macro Variables Created When the OUTDESIGN= Option Is Specified

Macro Variable Name	Contains
_QRSMOD	Selected design matrix columns for BY group 1
_QRSMOD1	Selected design matrix columns for BY group 1
_QRSMOD2	Selected design matrix columns for BY group 2
.	
.	
.	
_QRSMOD*m*T*n*	Selected design matrix columns for quantile *n* and BY group *m*

The macro variables in Table 104.5 show the number of quantiles and BY groups.

Table 104.5 Macro Variables Showing the Number of Quantiles and BY Groups

Macro Variable Name	Contains
_QRSNUMBYS	The number of BY groups
_QRSNUMTAUS	The number of quantiles
_QRSBY1NUMTAUS	The number of _QRSIND1T*j* macro variables actually made
_QRSBY2NUMTAUS	The number of _QRSIND2T*j* macro variables actually made
.	
.	
.	
_QRSNUMBYTAUS	The number of _QRSIND*i*T*j* macro variables actually made. This value can be less than _QRSNUMBYS × _QRSNUMTAUS, and it is less than or equal to MAXMACRO=*n*.

See the section "Macro Variables That Contain Selected Models" on page 8683 for more information.

NAMELEN=*number*

specifies the maximum length of effect names. By default, NAMELEN=20. If you specify a value less than 20, the default is used.

NOPRINT

suppresses all displayed output (including plots).

OUTDESIGN<(*options***)><=***SAS-data-set***>**

creates a data set that contains the design matrix. By default, the QUANTSELECT procedure includes in the OUTDESIGN data set the **X** matrix that corresponds to all the effects in the selected models. Two schemes for naming the columns of the design matrix are available:

- In the first scheme, names of the parameters are constructed from the parameter labels that appear in the parameter estimates table. This naming scheme is the default when you do not request BY processing, or when you specify the FULLMODEL option with BY processing.
- In the second scheme, the design matrix column names consist of a prefix followed by an index. The default name prefix is _X. This scheme is used when you specify the PREFIX= option, or when you specify a BY statement without using the FULLMODEL option; otherwise the first scheme is used.

If you have partitioned the input data (either by using the PARTITION statement or by including the _ROLE_ variable in the data set that is specified in the DATA= option in the PROC QUANTSELECT statement), then a character variable _ROLE_ is also included in the OUTDESIGN= data set. The following table shows the value of _ROLE_ for each observation:

ROLE Value	Observation Role
TEST	Testing
TRAIN	Training
VALIDATE	Validation
MISSING	Invalid for missing values or negative weight

You can specify the following *options* in parentheses to control the contents of the OUTDESIGN= data set:

ADDINPUTVARS
includes all the input data set variables in the OUTDESIGN= data set.

ADDVALDATA
includes the VALDATA= data set observations in the OUTDESIGN= data set. This option is ignored if the VALDATA= data set is not specified.

ADDTESTDATA
includes the TESTDATA= data set observations in the OUTDESIGN= data set. This option is ignored if TESTDATA= data set is not specified.

FULLMODEL
includes in the OUTDESIGN= data set parameters that correspond to all effects that are specified in the MODEL statement. By default, only parameters that correspond to the selected model are included.

NAMES
produces a table that associates columns in the OUTDESIGN= data set with the labels of the parameters they represent.

PREFIX< =*prefix* >
creates the design matrix column names from a prefix followed by an index. The default *prefix* is _X.

PARMLABELSTYLE=*options*
specifies how parameter names and labels are constructed for nested and crossed effects.

The following *options* are available:

INTERLACED < (SEPARATOR=*quoted string*) >
forms parameter names and labels by positioning levels of classification variables and constructed effects adjacent to the associated variable or constructed effect name and using " * " as the delimiter for both crossed and nested effects. This style of naming parameters and labels is used in the TRANSREG procedure. You can request truncation of the classification variable names used in forming the parameter names and labels by using the CPREFIX= and LPREFIX= options in the CLASS statement. You can use the SEPARATOR= suboption to change the delimiter between the crossed variables in the effect. PARMLABELSTYLE=INTERLACED is not supported if you specify the SPLIT option in an EFFECT statement or a CLASS statement. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender male * City Beijing
City London * Age
```

SEPARATE

specifies that in forming parameter names and labels, the effect name appears before the levels associated with the classification variables and constructed effects in the effect. You can control the length of the effect name by using the NAMELEN= option in the PROC GLMSELECT statement. In forming parameter labels, the first level that is displayed is positioned so that it starts at the same offset in every parameter label—this enables you to easily distinguish the effect name from the levels when the parameter labels are displayed in a column in the "Parameter Estimates" table. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender*City male Beijing
Age*City    London
```

SEPARATECOMPACT

requests the same parameter naming and labeling scheme as PARMLABELSTYLE=SEPARATE except that the first level in the parameter label is separated from the effect name by a single blank. This style of labeling is used in the PLS procedure and is the default if you do not specify the PARMLABELSTYLE option. The following are examples of the parameter labels in this style (Age is a continuous variable, Gender and City are classification variables):

```
Age
Gender*City male Beijing
Age*City London
```

PLOTS | PLOT < (*global-plot-options*) > < =*plot-request* < (*options*) > >

PLOTS | PLOT < (*global-plot-options*) > < =(*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the plots that are produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around it. Here are some examples:

```
plots=all
plots=coefficients(unpack)
plots(unpack)=(coef acl crit)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc quantselect plots=all;
   class temp sex / split;
   model depVar = sex sex*temp;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *global-plot-options*, which apply to all plots generated by the QUANTSELECT procedure, unless they are altered by specific plot *options*.

ENDSTEP=*n*
specifies that the step ranges shown on the horizontal axes of plots terminate at the specified step. By default, the step range shown terminates at the final step of the selection process. If you specify the ENDSTEP= option as both a *global-plot-option* and as an *option* for a specific *plot-request*, then PROC QUANTSELECT uses the ENDSTEP=*n option* for the specific *plot-request*.

LOGP | LOGPVALUE
displays the natural logarithm of the entry and removal significance levels when the SELECT=SL option is specified in the MODEL statement.

MAXSTEPLABEL=*n*
specifies the maximum number of characters beyond which labels of effects on plots are truncated. The default is MAXSTEPLABEL=256.

MAXPARMLABEL=*n*
specifies the maximum number of characters beyond which parameter labels on plots are truncated. The default is MAXPARMLABEL=256.

STARTSTEP=*n*
specifies that the step ranges shown on the horizontal axes of plots start at the specified step. By default, the step range shown starts at the initial step of the selection process. If you specify the STATSTEP= option as both a *global-plot-option* and as an *option* for a specific *plot-request*, then PROC QUANTSELECT uses the STARTSTEP=*n option* for the specific *plot-request*. The default is STARTSTEP=0.

STEPAXIS=EFFECT | NORMB | NUMBER
specifies the method for labeling the horizontal plot axis. This axis represents the sequence of entering or departing effects. The default is STEPAXIS=EFFECT.

STEPAXIS=EFFECT
labels each step by a prefix followed by the name of the effect that enters or leaves at that step. The prefix consists of the step number followed by a "+" sign or a "–" sign, depending on whether the effect enters or leaves at that step.

STEPAXIS=NORMB
labels the horizontal axis value at step *i* with the penalty on the parameter estimates at step *i*, normalized by the penalty on the parameter estimates at the final step. This option is valid only with regularization selection methods.

STEPAXIS=NUMBER
labels each step with the step number.

UNPACK
displays each graph separately. (By default, some graphs can appear together in a single panel.) You can also specify UNPACK as a suboption with CRITERIA and COEFFICIENTS options for specific *plot-requests*.

The following list describes the specific *plot-requests* and their *options*.

ALL
displays all appropriate graphs.

ACL | ACLPLOT < (*aclplot-option*) >
plots the progression of the average check losses on the training data, and on the test and validation data when these data are provided with the TESTDATA= or VALDATA= options or are produced by using the PARTITION statement. When the PROC QUANTSELECT procedure is applied on multiple quantile levels, the ACL option and its suboptions apply to the ACL plots for each of the quantile levels.

You can specify the following *aclplot-option*:

STEPAXIS=EFFECT | NORMB | NUMBER
specifies the method for labeling the horizontal plot axis. See the STEPAXIS= option in the *global-plot-options* for more information.

COEF | COEFFICIENTS | COEFFICIENTPANEL < (*coefficient-panel-options*) >
displays a panel of two plots for each quantile level. The upper plot shows the progression of the parameter values as the selection process proceeds. The lower plot shows the progression of the CHOOSE= criterion. If no CHOOSE= criterion is in effect, then the AICC criterion is displayed. You can specify the following *coefficient-panel-options*:

LABELGAP=*percentage*
specifies the percentage of the vertical axis range that forms the minimum gap between successive parameter labels at the final step of the coefficient progression plot. If the values of more than one parameter at the final step are closer than this gap, then the labels on all but one of these parameters are suppressed. The default is LABELGAP=5.

LOGP | LOGPVALUE
displays the natural logarithm of the entry and removal significance levels when the SELECT=SL option is specified in the MODEL statement.

STEPAXIS=EFFECT | NORMB | NUMBER
specifies the horizontal axis to be used. See the STEPAXIS= option in the *global-options* for more information.

UNPACK | UNPACKPANEL
displays the coefficient progression and the CHOOSE= criterion progression in separate plots.

CRIT | CRITERIA | CRITERIONPANEL < (*criterion-panel-options*) >
plots a panel of model fit criteria. If multiple quantile levels apply, the CRITERIA option plots a panel of model fit criteria for each quantile level. The criteria that are displayed are AIC, AICC, and SBC, in addition to any other criteria that are named in the CHOOSE=, SELECT=, STOP=, and STATS= options in the MODEL statement. You can specify the following *criterion-panel-options*:

STEPAXIS=EFFECT | NORMB | NUMBER
specifies the horizontal axis to be used. See the STEPAXIS= option in the *global-options* for more information.

UNPACK | UNPACKPANEL
displays each criterion progression on a separate plot.

NONE
suppresses all plots.

SEED=*number*
specifies an integer that is used to start the pseudorandom number generator for random partitioning of data for training, testing, and validation. If you do not specify a seed or if you specify a value less than or equal to 0, the seed is generated by reading the time of day from the computer's clock.

TESTDATA=*SAS-data-set*
names a SAS data set that contains test data. This data set must contain all the effects that are specified in the MODEL statement. Furthermore, when you also specify a BY statement and the TESTDATA= data set contains any of the BY variables, then the TESTDATA= data set must also contain all the BY variables sorted in the order of the BY variables. In this case, only the test data for a specific BY group are used with the corresponding BY group in the analysis data. If the TESTDATA= data set contains none of the BY variables, then the entire TESTDATA= data set is used with each BY group of the analysis data.

If you specify both a TESTDATA= data set and the PARTITION statement, then the testing observations from the DATA= data set are merged with the TESTDATA= data set for testing purposes.

VALDATA=*SAS-data-set*
names a SAS data set that contains validation data. This data set must contain all the effects that are specified in the MODEL statement. Furthermore, when a BY statement is used and the VALDATA= data set contains any of the BY variables, then the VALDATA= data set must also contain all the BY variables sorted in the order of the BY variables. In this case, only the validation data for a specific BY group are used with the corresponding BY group in the analysis data. If the VALDATA= data set contains none of the BY variables, then the entire VALDATA= data set is used with each BY group of the analysis data.

If you specify both a VALDATA= data set and the PARTITION statement, then the validation observations from the DATA= data set are merged with the VALDATA= data set for validation purposes.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC QUANTSELECT to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the QUANTSELECT procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < *(v-options)* > < *variable* < *(v-options . . .)* > > < / *v-options* > < *options* > **;**

The CLASS statement names the classification variables to be used in the analysis. The CLASS statement must precede the MODEL statement.

Table 104.6 summarizes the *options* and *v-options* available in the CLASS statement.

Table 104.6 CLASS Statement Options

option **or** *v-option*	**Description**
DELIMITER=	Specifies the delimiter
DESCENDING	Reverses the sort order
MISSING	Allows for missing values
ORDER=	Specifies the sort order
PARAM=	Specifies the parameterization method
REF=	Specifies the reference level
SHOW	Requests a table for each CLASS variable
SPLIT	Splits CLASS variables into independent effects

You can specify the following *options* after a slash (/):

DELIMITER=*'c'*

specifies the delimiter character, '*c*', to be used between levels of classification variables when parameter names and lists of class level values are built. The default delimiter is a space. This option is useful if the levels of a classification variable contain embedded blanks.

SHOW | SHOWCODING

requests a table that shows the coding used for each classification variable.

You can specify various *v-options* for each variable by enclosing them in parentheses after the variable name; these are called individual *v-options*. You can also specify global *v-options* by placing them after a slash (/) at the end of the CLASS statement. Global *v-options* are applied to all the variables specified in the CLASS statement. If you specify more than one CLASS statement, the global *v-options* specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *v-options* override the global *v-options* except for the PARAM=GLM option. The global PARAM=GLM option overrides all individual PARAM= options.

You can specify the following *v-options*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable. This option applies only when you specify the PARMLABELSTYLE=INTERLACED option in the PROC QUANTSELECT statement.

DESCENDING

DESC

reverses the sort order of the classification variable.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable. This option applies only when you specify the PARMLABELSTYLE=INTERLACED option in the PROC QUANTSELECT statement.

MISSING

allows missing values, such as '.' for a numeric variable or a blank for a character variable, as valid values for the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. If ORDER=FORMATTED for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. The following table shows how PROC QUANTSELECT interprets values of the ORDER= option.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For FORMATTED and INTERNAL, the sort order is machine dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. Design matrix columns are created from CLASS variables according to the following coding schemes. If the PARAM= option is not specified with any individual CLASS variable, by default, PARAM=GLM. Otherwise, the default is PARAM=EFFECT. If PARAM=ORTHPOLY or PARAM=POLY, and the CLASS levels are numeric, then the ORDER= option in the CLASS statement is ignored, and the internal, unformatted values are used. See the section "CLASS Variable Parameterization and the SPLIT Option" on page 4471 in Chapter 56, "The GLMSELECT Procedure," for more information.

EFFECT	specifies effect coding.
GLM	specifies less-than-full-rank coding. This option can be used only as a global *v-option* (after the slash in the CLASS statement).
ORDINAL | THERMOMETER	specifies the cumulative parameterization for an ordinal CLASS variable.
POLYNOMIAL | POLY	specifies polynomial coding.
REFERENCE | REF	specifies reference-cell coding.
ORTHEFFECT	orthogonalizes PARAM=EFFECT.
ORTHORDINAL | ORTHOTHERM	orthogonalizes PARAM=ORDINAL.
ORTHPOLY	orthogonalizes PARAM=POLYNOMIAL.
ORTHREF	orthogonalizes PARAM=REFERENCE.

The EFFECT, POLYNOMIAL, REFERENCE, and ORDINAL coding schemes and their orthogonal parameterizations are full rank. The REF= option in the CLASS statement determines the reference level for the EFFECT and REFERENCE schemes and their orthogonal parameterizations.

REF=*'level'* | **FIRST** | **LAST**

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For an individual (but not a global) REF= *v-option*, you can specify the *level* of the variable to use as the reference level. For a global or individual REF= *v-option*, you can specify REF=FIRST (which designates the first-ordered level as reference) or REF=LAST (which designates the last-ordered level as reference). The default is REF=LAST.

SPLIT

enables the columns of the design matrix that correspond to any effect that contains a split classification variable to be selected to enter or leave a model independently of the other design columns of that effect. For example, suppose a variable named temp has three levels with values `'hot'`, `'warm'`, and `'cold'`, and a variable named sex has two levels with values `'M'` and `'F'`. The following statements include SPLIT as a global *v-option*:

```
proc quantselect;
   class temp sex / split;
   model depVar = sex sex*temp;
run;
```

Because both the classification variables are split, the two effects named in the MODEL statement are split into eight effects. The effect `'sex'` is split into two effects labeled `'sex_M'` and `'sex_F'`. The effect `'sex*temp'` is split into six effects labeled `'sex_M*temp_hot'`, `'sex_F*temp_hot'`, `'sex_M*temp_warm'`, `'sex_F*temp_warm'`, `'sex_M*temp_cold'`, and `'sex_F*temp_cold'`. The previous PROC QUANTSELECT statements are equivalent to the following statements for the split version of the DATA= data set:

```
proc quantselect;
   model depVar =  sex_M sex_F sex_M*temp_hot  sex_F*temp_hot
                               sex_M*temp_warm sex_F*temp_warm
                               sex_M*temp_cold sex_F*temp_cold;
run;
```

You can specify the SPLIT option for individual classification variables. For example, consider the following PROC QUANTSELECT statements:

```
proc quantselect;
   class temp(split) sex;
   model depVar = sex sex*temp;
run;
```

In this case, the effect `'sex'` is not split, and the effect `'sex*temp'` is split into three effects labeled `'sex*temp_hot'`, `'sex*temp_warm'`, and `'sex*temp_cold'`. Furthermore each of these three split effects now has two parameters that correspond to the two levels of `'sex,'` and the previous PROC QUANTSELECT statements are equivalent to the following statements for the split version of the DATA= data set:

```
proc quantselect;
   class sex;
   model depVar = sex sex*temp_hot sex*temp_warm sex*temp_cold;
run;
```

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 104.7 summarizes the *options* available in the CODE statement.

Table 104.7 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) **;**

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 104.8 summarizes the *options* available in the EFFECT statement.

Table 104.8 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

MODEL *dependent* **=** < *effects* > / < *options* > **;**

The MODEL statement names the dependent variable and the covariate effects, including covariates, main effects, constructed effects, interactions, and nested effects; see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure," for more information. If you omit the explanatory effects, PROC QUANTSELECT fits an intercept-only model.

After the keyword MODEL, specify the dependent (response) variable, followed by an equal sign, followed by the explanatory effects.

Table 104.9 summarizes the *options* available in the MODEL statement.

Table 104.9 MODEL Statement Options

Option	Description
DETAILS=	Specifies the level of effect selection detail to display
HIERARCHY=	Specifies hierarchy of effects to impose
NOINT	Specifies models without an explicit intercept
QUANTILE=	Specifies quantile levels to be applied
SELECTION=	Specifies effect selection method
STATS=	Specifies additional statistics to be displayed
TEST=	Specifies the test type for computing significance levels

The following list provides details about the *options* that you can specify in the MODEL statement after a slash (/):

DETAILS=*level* | **STEPS** < **(***step options***)** >

specifies the level of effect selection detail that is displayed, where *level* can be ALL, STEPS, or SUMMARY. The default if the DETAILS= option is omitted is DETAILS=SUMMARY that produces only the selection summary table. The DETAILS=ALL option produces the following:

- entry and removal statistics for each variable that is selected in the model building process
- fit statistics and parameter estimates
- entry and removal statistics for the top five candidates for inclusion or exclusion at each step
- a selection summary table

The option DETAILS=STEPS < (*step options*) > provides the step information and the selection summary table. The following suboptions can be specified within parentheses after the DETAILS=STEPS option:

FITSTATISTICS | FITSTATS | FIT
requests fit statistics at each selection step.

PARAMETERESTIMATES | PARMEST
requests parameter estimates at each selection step.

CANDIDATES < (ALL | *n*) >
requests entry or removal statistics for the best *n* candidate effects for inclusion or exclusion at each step. If you specify the CANDIDATES(ALL) option, then all candidates are shown. If the CANDIDATES(*n*) is not specified. then the best 10 candidates are shown. The entry or removal statistic is the statistic named in the SELECT= option that is specified in the MODEL statement SELECTION= option.

HIERARCHY=*keyword*
HIER=*keyword*
specifies whether and how the model hierarchy requirement is applied. This option also controls whether a single effect or multiple effects are allowed to enter or leave the model in one step. You can specify that only CLASS effects, or both CLASS and continuous effects, be subject to the hierarchy requirement. This option is ignored unless you also specify one of the following options: SELECTION=FORWARD, SELECTION=BACKWARD, or SELECTION=STEPWISE.

Model hierarchy refers to the requirement that for any term to be in the model, all model effects contained in the term must be present in the model. For example, in order for the interaction A*B to enter the model, the main effects A and B must be in the model. Likewise, neither effect A nor effect B can leave the model while the interaction A*B is in the model.

You can specify the following *keywords*:

NONE
specifies that model hierarchy not be maintained. Any single effect can enter or leave the model at any given step of the selection process.

SINGLE
specifies that only one effect enter or leave the model at one time, subject to the model hierarchy requirement. For example, suppose that the model contains the main effects A and B and the interaction A*B. In the first step of the selection process, either A or B can enter the model. In the second step, the other main effect can enter the model. The interaction effect can enter the model only when both main effects have already entered. Also, before A or B can be removed from the model, the A*B interaction must first be removed. All effects (CLASS and interval) are subject to the hierarchy requirement.

SINGLECLASS
is the same as HIERARCHY=SINGLE except that only CLASS effects are subject to the hierarchy requirement.

The default is HIERARCHY=NONE.

NOINT

suppresses the intercept term that is otherwise included in the model.

QUANTILE=*number-list* | **PROCESS<** (*suboption*) **>** | **FQPR<** (*suboption*) **>**

QUANTLEV=*number-list* | **PROCESS<** (*suboption*) **>** | **FQPR<** (*suboption*) **>**

specifies the quantile levels for the quantile regression. A valid quantile level must be a number in the range (0,1). You can specify the following values for the QUANTILE= option:

number-list

performs effect selection for quantile regression at the quantile levels that are specified in the *number-list*. You can specify QUANTILE=0 or QUANTILE=1 for the ALGORITHM=SIMPLEX algorithm. For more information about extremal quantile levels, see the section "Quantile Regression for Extremal Quantile Levels" on page 8678.

PROCESS< (*suboption*) **>**

performs effect selection for quantile process regression. For more information about quantile process regression, see the section "Quantile Process Regression" on page 8677. If you specify QUANTILE=PROCESS, the value of the ALGORITHM= option in the PROC QUANTREG statement must be SIMPLEX either by default or by specifying it. The QUANTILE=PROCESS option cannot be used with LASSO selection methods. You can specify the following *suboption* in parentheses after QUANTILE=PROCESS.

N=*n* | **ALL**

NTAU=*n* | **ALL**

specifies how many quantile levels you expect to cover for the quantile process. You can specify one of the following values:

ALL	performs effect selection for accurate quantile process regression.
n	performs effect selection for approximate quantile process regression. The approximate quantile process is computed at *n* equally spaced quantile levels, $\left\{\frac{1}{n+1}, \ldots, \frac{n}{n+1}\right\}$, in addition to three control quantile levels $\{0, 0.5, 1\}$.

If the number of observations for training is more than 1,000, by default N=500. Otherwise, by default N=ALL.

The QUANTILE=PROCESS option produces the mean parameter estimates table and the fit statistics table for the specified quantile process. You can output the quantile process parameter estimates table to a data set by specifying the following ODS OUTPUT statement:

```
ods output ProcessEst=<data set>;
```

FQPR< (*suboption*) **>**

uses a fast quantile process regression method to approximate quantile process regression on a grid of *n* equally spaced quantile levels. The QUANTILE=FQPR option uses an efficient interior point algorithm to fit quantile models. You can specify the following *suboptions*:

N=*n*
specifies the number *n* of equally spaced quantile levels at which to fit the quantile process regression, where *n* is an integer.

OBSRATIO=*value*
OR=*value*
specifies the number of equally spaced quantile levels as its ratio to the total number of training observations. For example, if the number of training observations is 1,000 and you specify the OR=0.2 suboption, a quantile process regression model is fit for $n = 0.2 \times 1{,}000 = 200$ equally spaced quantile levels. The FQPR option ignores the OR= suboption if a valid N= suboption is specified.

L=*value*
specifies the starting quantile level of the quantile-level grid. By default, $\mathrm{L} = 1/2n$ if U is not specified; otherwise, $\mathrm{L} = \mathrm{U}/(2n-1)$.

U=*value*
specifies the ending quantile level of the quantile-level grid. By default, $\mathrm{U} = (2n-1)/2n$ if L is not specified; otherwise, $\mathrm{U} = (\mathrm{L} + 2n - 2)/(2n-1)$.

If you specify neither the N=*n* nor the OR=*value* suboption, the FQPR option determines the number of quantile levels as the lesser of 100 and half the number of the training observations.

The QUANTILE=FQPR option produces the average parameter estimates table, the fit statistics table, and the quantile level information table for the specified quantile-level grid. You can output the FQPR parameter estimates tables to a data set by specifying the following ODS OUTPUT statement:

```
ods output ProcessEst=<data set>;
```

By default, QUANTILE=0.5, which fits a median regression.

SELECTION=*method* **<(***method-options***)>**
specifies the *method* used to select the model, optionally followed by parentheses that enclose *method-options* that apply to the specified method. The default is SELECTION=STEPWISE.

You can specify the following *methods*, which are explained in detail in the section "Effect Selection Methods" on page 8679.

NONE specifies full model fitting without effect selection.

FORWARD specifies forward selection. This method starts with no effects in the model and adds effects.

BACKWARD specifies backward elimination. This method starts with all effects in the model and deletes effects.

STEPWISE specifies stepwise regression. This is similar to the FORWARD method except that effects already in the model do not necessarily stay there.

LASSO specifies a method that adds and deletes parameters based on a version of estimated check risk where the weighted L1-norm of certain weighted regression coefficients is penalized. For more information, see the section "LASSO Method (LASSO)" on

page 8680. If the model contains CLASS variables or constructed effects, these CLASS variables or constructed effects are split into separate covariates.

Table 104.10 lists the applicable *method-options* for each *method*.

Table 104.10 Applicable *method-options* for Each *method*

method-option	FORWARD	BACKWARD	STEPWISE	LASSO
ADAPTIVE				x
CHOOSE=	x	x	x	x
INCLUDE=	x	x	x	x
MAXSTEP=	x	x	x	x
SELECT=	x	x	x	
SLENTRY=	x		x	
SLSTAY=		x	x	
STOP=	x	x	x	x
STOPHORIZON=	x	x	x	x

You can specify the following *method-option* in parentheses after the *method*. As described in Table 104.10, not all *method-options* apply to every SELECTION= method.

ADAPTIVE
ADAPT

specifies the adaptive LASSO selection method. The ADAPTIVE option can be used only with the SELECTION=LASSO option.

CHOOSE=*criterion*

chooses from the list of models (with one model at each step of the selection process) the model that yields the best value of the specified *criterion* as the final selected model. If the optimal value of the specified *criterion* occurs for more than one model, then the model with the smallest number of parameters is chosen. If you do not specify the CHOOSE= option, then the model selected is the model at the final step in the selection process for the SELECT=SL criterion, or the STOP= option is applied as the CHOOSE= option for all the other cases.

You can specify the following values for *criterion* in the CHOOSE= option. See the section "Criteria Used in Model Selection Methods" on page 8680 for more information about these criteria.

ADJR1 chooses the model with the largest adjusted quantile regression R statistic.

AIC chooses the model with the smallest Akaike's information criterion.

AICC chooses the model with the smallest corrected Akaike's information criterion.

SBC chooses the model with the smallest Schwarz Bayesian information criterion.

VALIDATE chooses the model with the smallest average check loss for the validation data. You can specify CHOOSE=VALIDATE only if you have specified a VALDATA= data set in the PROC QUANTSELECT statement or if you have reserved part of the input data for validation by using either a PARTITION statement or a _ROLE_ variable in the input data.

INCLUDE=*n*

forces the first *n* effects listed in the MODEL statement to be included in all models. The selection methods are performed on the other effects in the MODEL statement.

MAXSTEP=*n*

specifies the maximum number of selection steps. The default value of *n* is the number of effects in the MODEL statement when SELECTION=FORWARD or SELECTION=BACKWARD and is three times the number of effects when SELECTION=STEPWISE or SELECTION=LASSO.

SELECT=*criterion*

specifies the criterion that PROC QUANTSELECT uses to determine the order in which effects enter or leave at each step of the specified selection method. This option is not valid when SELECTION=LASSO. You can specify the following values for *criterion*: ADJR1, AIC, AICC, SBC, SL, and VALIDATE. See the section "Criteria Used in Model Selection Methods" on page 8680 for more information about these criteria.

When SELECT=SL, the effect selection depends on the selection method and is described in the relevant subsection of the section "Effect Selection Methods" on page 8679. Otherwise, the effect that is selected to enter or leave at a step of the selection process is the effect whose addition to or removal from the current model produces the maximum improvement in the specified criterion.

If validation data exist, the default is SELECT=VALIDATE; otherwise, the default is SELECT=SBC.

SLENTRY=*value*

SLE=*value*

specifies the significance level for entry, used when the SELECT=SL option is in effect. The defaults are 0.50 when SELECTION=FORWARD and 0.15 when SELECTION=STEPWISE.

SLSTAY=*value*

SLS=*value*

specifies the significance level for staying in the model, used when the SELECT=SL option is in effect. The defaults are 0.10 when SELECTION=BACKWARD and 0.15 when SELECTION=STEPWISE.

STOP=*criterion*

specifies the *criterion* for stopping the selection process. If the maximum number of steps is specified in the MAXSTEP= option and the *criterion* does not stop the selection process before the maximum number of steps for the selection method, then the selection process terminates at the maximum number of steps.

You can specify the following values for *criterion*. See the section "Criteria Used in Model Selection Methods" on page 8680 for more detailed descriptions of these criteria.

NONE enables the model selection process to go through all possible steps.

ADJR1 stops selection at the step where the next SH= steps (or all remaining steps) would yield models with smaller values of the adjusted quantile regression R (ADJR1) statistic.

AIC stops selection at the step where the next SH= steps (or all remaining steps) would yield models with larger values of Akaike's information criterion.

AICC stops selection at the step where the next SH= steps (or all remaining steps) would yield models with larger values of the corrected Akaike's information criterion.

SBC stops selection at the step where the next SH= steps (or all remaining steps) would yield models with larger values of the Schwarz Bayesian information criterion.

VALIDATE stops selection at the step where the next SH= steps (or all remaining steps) would yield models with larger values of the average check loss for the validation data. You can specify STOP=VALIDATE only if you have specified a VALDATA= data set in the PROC QUANTSELECT statement or if you have reserved part of the input data for validation by using either a PARTITION statement or a _ROLE_ variable in the input data.

The default *criterion* depends on other factors as follows:

- If validation data exist, STOP=VALIDATE by default.
- If validation data do not exist and you specify SELECTION=LASSO, STOP=SBC by default. The SELECTION=LASSO option does not support the SELECT=*method-option*.
- If validation data do not exist and you specify SELECTION= STEPWISE, FORWARD, or BACKWARD, the default is one of the following:
  - When you specify SELECT=SL, the entry and stay significance levels terminate the effect selection process.
  - When you do not specify SELECT=SL, the default is the criterion that is specified in the SELECT= option.

If you specify both the STOP= option and SELECT=SL, the following rules apply:

- When you specify SELECTION=STEPWISE, the entry and stay significance levels can terminate the effect selection process when no candidate effect is available to be deleted from or added to the model. This extra check can result in the selection terminating before a local minimum of the STOP= criterion is found.
- When you specify SELECTION=FORWARD, the effect selection process ignores the entry significance level even if you use the SLE= option to specify the entry significance level.
- When you specify SELECTION=BACKWARD, the effect selection process ignores the stay significance level even if you use the SLS= option to specify the stay significance level.

STOPHORIZON=*n*
SH=*n*

looks ahead for the specified number of steps to decide whether an extremum of the stop criterion is achieved. This option applies only to the STOP= criterion. The default is STOPHORIZON=1.

For example, suppose that the stop criterion values at steps 1 through 5 are 4, 3, 5, 6, and 2, respectively. If you specify STOPHORIZON=1, then the selection process terminates after looking at the model at step 3, and the final selected model is the model at step 2. If you specify STOPHORIZON=2, the selection process stops after looking at the model at step 4, and the final selected model is the model at step 2. However, if you specify STOPHORIZON=3 or higher, then the local minimum in the stop value sequence at step 2 cannot stop the selection process because a lower value is achieved at step 5, which is within 3 steps beyond this local minimum step.

STAT=*name* | (*names*)

STATS=*name* | (*names*)

specifies which model fit statistics to display in the selection summary table. To specify multiple model fit statistics, specify a list of *names* in parentheses. If you omit this option, the default set of statistics that are displayed in these tables includes all the criteria that are specified in any of the CHOOSE=, SELECT=, and STOP= *method-options*.

You can specify the following values for *name*:

ADJR1 displays the adjusted quantile regression R statistic.

AIC displays the Akaike's information criterion.

AICC displays the corrected Akaike's information criterion.

ACL displays the average check losses for the training, test, and validation data. The ACL statistics for the test and validation data are reported only if you have specified the TESTDATA= option or the VALDATA= option in the PROC QUANTSELECT statement or if you have reserved part of the input data for testing or validation by using either a PARTITION statement or a _ROLE_ variable in the input data.

R1 displays the quantile regression R statistic.

SBC displays the Schwarz Bayesian information criterion.

The statistics ADJR1, AIC, AICC, and SBC can be computed with little computation cost. However, computing ACL for test and validation data when these are not used in any of the CHOOSE=, SELECT=, and STOP= *method-options* can hurt performance.

TEST=*name*

specifies the test type for computing significance levels.

You can specify the following values for *name*:

LR1 specifies the likelihood ratio test Type I. The LR1 test score is

$$\frac{2(D_1(\tau) - D_2(\tau))}{\tau(1-\tau)\hat{s}}$$

where $D_1(\tau) = \sum \rho_\tau \left(y_i - \mathbf{x}_i \hat{\boldsymbol{\beta}}_1(\tau)\right)$ is the sum of check losses for the reduced model, $D_2(\tau) = \sum \rho_\tau \left(y_i - \mathbf{x}_i \hat{\boldsymbol{\beta}}(\tau)\right)$ is the sum of check losses for the extended model, and $\hat{s}$ is the estimated sparsity function. See the section "Quasi-Likelihood Ratio Tests" on page 8675 for more information.

LR2 specifies the likelihood ratio test Type II. The LR2 test score is

$$\frac{2D_2(\tau)\left(\log(D_1(\tau)) - \log(D_2(\tau))\right)}{\tau(1-\tau)\hat{s}}.$$

See the section "Quasi-Likelihood Ratio Tests" on page 8675 for more information.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *keyword* < **=***name* > > . . . < *keyword* < **=***name* > > **;**

The OUTPUT statement creates a new SAS data set that saves diagnostic measures that are calculated for the selected model. If you do not specify a *keyword*, then the only diagnostic included is the predicted response.

All the variables in the original data set are included in the new data set, along with variables that are created by the *keyword* options in the OUTPUT statement. These new variables contain the values of a variety of statistics and diagnostic measures that are calculated for each observation in the data set.

The OUTPUT data set is created in row-wise form, and the variable _QUANTILE_ is optional. For each appropriate *keyword* specified in the OUTPUT statement, one variable for each specified quantile level is generated. These variables appear in the sorted order of the specified quantile levels.

If you specify a BY statement, then a variable _BY_ that indexes the BY groups is included. For each observation, the value of _BY_ is the index of the BY group to which this observation belongs. This variable is useful for matching BY groups with macro variables that PROC QUANTSELECT creates. See the section "Macro Variables That Contain Selected Models" on page 8683 for more information.

If you have partitioned the input data by either using the PARTITION statement or including the _ROLE_ variable in the DATA= data set of the PROC QUANTSELECT statement, then a character variable _ROLE_ is included in the output data set. The following table shows the value of _ROLE_ for each observation:

ROLE Value	Observation Role
TEST	Testing
TRAIN	Training
VALIDATE	Validation

If you want to create a permanent SAS data set, you must specify a two-level name. For more information about permanent SAS data sets, see the discussion in *SAS Programmers Guide: Essentials*.

You can specify the following arguments in the OUTPUT statement:

keyword < **=***name* >

specifies the statistics to include in the output data set and optionally names the new variables that contain the statistics. Specify one of the following *keywords* for each desired statistic, followed optionally by an equal sign, and the *name* of a variable to contain the statistic. If you specify *keyword=name*, the new variable that contains the requested statistic has the specified name. If you omit the optional *=name* after a *keyword*, then the new variable name is formed by using a prefix of one or more characters that identify the statistic, followed by an underscore (_), followed by the dependent variable name.

PREDICTED | PRED | P includes predicted values in the output data set. The prefix for the default name is p.

QUANTLEVEL | QL includes observation quantile levels in the output data set. The prefix for the default name is ql. The QL= option is available only when you specify QUANTILE=PROCESS in the MODEL statement. For more information about observation quantile level, see the section "Observation Quantile Level" on page 8678.

RESIDUAL | RESID | R includes residuals, calculated as ACTUAL – PREDICTED, in the output data set. The prefix for the default name is r.

OUT=*SAS-data-set*
names the output data set. By default, PROC QUANTSELECT uses the DATA*n* convention to name the new data set.

PARTITION Statement

PARTITION < *option* > **;**

The PARTITION statement specifies how observations in the input data set are logically partitioned into disjoint subsets for model training, validation, and testing. Either you can designate a variable in the input data set and a set of formatted values of that variable to determine the role of each observation, or you can specify proportions to use for random assignment of observations for each role.

An alternative to using a PARTITION statement is to provide a variable named _ROLE_ in the input data set to define roles of observations in the input data. For more information about the _ROLE_ variable, see the description of the OUTDESIGN= option and see the section "OUTPUT Statement" on page 8672. The QUANTSELECT procedure ignores all observations whose _ROLE_ values are not equal to TRAIN, TEST, or VALIDATE. If you specify the PARTITION statement, then the _ROLE_ variable in the input data set (if it exists) is ignored. If you do not use a PARTITION statement and the input data do not contain a variable named _ROLE_, then all observations in the input data set are assigned to model training.

You can specify either (but not both) of the following *options*:

ROLEVAR=*variable* **(< TEST=***value* **> < TRAIN=***value* **> < VALIDATE=***value* **>)**
ROLE=*variable* **(< TEST=***value* **> < TRAIN=***value* **> < VALIDATE=***value* **>)**
names the variable in the input data set whose values are used to assign roles to each observation. The TEST=, TRAIN=, and VALIDATE= suboptions specify the formatted values of this variable that are used to assign observations roles. If you do not specify the TRAIN= suboption, then all observations whose role is not determined by the TEST= or VALIDATE= suboptions are assigned to training.

FRACTION(< TEST=*fraction* **> < VALIDATE=***fraction* **>)**
requests that specified proportions of the observations in the input data set be randomly assigned training and validation roles. You specify the proportions for testing and validation by using the TEST= and VALIDATE= suboptions. If you specify both the TEST= and the VALIDATE= suboptions, then the sum of the specified fractions must be less than 1 and the remaining fraction of the observations are assigned to the training role.

WEIGHT Statement

WEIGHT *variable* **;**

A WEIGHT statement names a variable in the input data set with values that are relative weights for a weighted quantile regression fit.

Values of the weight variable must be nonnegative. If an observation's weight is 0, the observation is deleted from the analysis. If a weight is negative or missing, it is set to 0, and the observation is excluded from the analysis.

Details: QUANTSELECT Procedure

Quantile Regression

This section describes the basic concepts and notations for quantile regression and quantile regression model selection.

Let $\{(y_i, \mathbf{x}_i) : i = 1, \ldots, n\}$ denote a data set of observations, where y_i are responses, and $\mathbf{x}_i$ are regressors. Koenker and Bassett (1978) defined the *regression quantile* at quantile level $\tau \in (0, 1)$ as any solution that minimizes the following objective function in $\boldsymbol{\beta}$:

$$\sum_{i=1}^{n} \rho_\tau \left(y_i - \mathbf{x}_i' \boldsymbol{\beta}\right)$$

where $\rho_\tau(r) = \tau r^+ + (1 - \tau) r^-$ is a check loss function in which $r^+ = \max(r, 0)$ and $r^- = \max(-r, 0)$.

If you specify weights $w_i, i = 1, \ldots, n$, in the WEIGHT statement, weighted quantile regression is carried out by solving

$$\min_{\boldsymbol{\beta} \in \mathbf{R}^p} \sum_{i=1}^{n} \rho_\tau \left(w_i (y_i - \mathbf{x}_i' \boldsymbol{\beta})\right)$$

Quasi-Likelihood Information Criteria

Given quantile level τ, assume that the distribution of Y_i conditional on $\mathbf{x}_i$ follows the linear model

$$Y_i = \mathbf{x}_i' \boldsymbol{\beta} + \epsilon_i$$

where ϵ_i for $i = 1, \ldots, n$ are iid in distribution F. Further assume that F is an asymmetric Laplace distribution whose density function is

$$f_\tau(r) = \frac{\tau(1 - \tau)}{\sigma} \exp\left(-\frac{\rho_\tau(r)}{\sigma}\right)$$

where σ is the scale parameter. Then, the associated -log likelihood function is

$$l_\tau(\boldsymbol{\beta}, \sigma) = n \log(\sigma) + \sigma^{-1} \sum_{i=1}^{n} \rho_\tau(y_i - \mathbf{x}_i' \boldsymbol{\beta}) - n \log(\tau(1-\tau))$$

Under these settings, the maximum likelihood estimate (MLE) of $\boldsymbol{\beta}$ is the same as the relevant level τ quantile regression solution

$$\hat{\boldsymbol{\beta}}(\tau) = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{n} \rho_\tau \left(y_i - \mathbf{x}_i' \boldsymbol{\beta}\right)$$

The MLE for σ is

$$\hat{\sigma}(\tau) = n^{-1} \sum_{i=1}^{n} \rho_\tau \left(y_i - \mathbf{x}_i' \hat{\boldsymbol{\beta}}(\tau)\right)$$

where $\hat{\sigma}(\tau)$ equals the level τ average check loss, ACL(τ), for the quantile regression solution.

According to the general form of Akaike's information criterion, AIC $= (-2l + 2p)$, the quasi-likelihood AIC for quantile regression is

$$\mathrm{AIC}(\tau) = 2n \ln\left(\mathrm{ACL}(\tau)\right) + 2p$$

where p is the degrees of freedom for the fitted model.

Similarly, the quasi-likelihood corrected AIC and Schwarz Bayesian information criterion can be formulated respectively as follows:

$$\mathrm{AICC}(\tau) = 2n \ln\left(\mathrm{ACL}(\tau)\right) + \frac{2pn}{n-p-1}$$

$$\mathrm{SBC}(\tau) = 2n \ln\left(\mathrm{ACL}(\tau)\right) + p \ln(n)$$

In fact, the quasi-likelihood AIC, AICC, and SBC are fairly robust, and they can be used to select effects for data sets without the iid assumption in asymmetric Laplace distribution. See "Example 104.1: Simulation Study" on page 8693 for a simulation study that applies SBC for effect selection on a data set that is generated from a naive instrumental model (Chernozhukov and Hansen 2008).

Quasi-Likelihood Ratio Tests

Under the iid assumption, Koenker and Machado (1999) proposed two types of quasi-likelihood ratio tests for quantile regression, where the error distribution is flexible but not limited to the asymmetric Laplace distribution. The Type I test score, LR1, is defined as

$$\frac{2(D_1(\tau) - D_2(\tau))}{\tau(1-\tau)\hat{s}}$$

where $\hat{s}$ is the estimated sparsity function, $D_1(\tau) = \sum \rho_\tau \left(y_i - \mathbf{x}_i \hat{\boldsymbol{\beta}}_1(\tau)\right)$ is the sum of check losses for the reduced model, and $D_2(\tau) = \sum \rho_\tau \left(y_i - \mathbf{x}_i \hat{\boldsymbol{\beta}}(\tau)\right)$ is the sum of check losses for the extended model. The Type II test score, LR2, is defined as

$$\frac{2D_2(\tau)\left(\log(D_1(\tau)) - \log(D_2(\tau))\right)}{\tau(1-\tau)\hat{s}}$$

Under the null hypothesis that the reduced model is the true model, both LR1 and LR2 follow a χ^2 distribution with $df = df_2 - df_1$ degrees of freedom, where df_1 and df_2 are the degrees of freedom for the reduced model and the extended model, respectively.

If you specify the TEST=LR1 option in the MODEL statement, the QUANTSELECT procedure uses LR1 score to compute the significance level. Or you can use the substitutable TEST=LR2 option for computing the significance level on Type II quasi-likelihood ratio test.

Under the iid assumption, the sparsity function is defined as $s(\tau) = 1/f(F^{-1}(\tau))$. Here the distribution of errors F is flexible but not limited to the asymmetric Laplace distribution. The algorithm for estimating $s(\tau)$ is as follows:

1. Fit a quantile regression model and compute the residuals. Each residual $r_i = y_i - \mathbf{x}_i'\hat{\boldsymbol{\beta}}(\tau)$ can be viewed as an estimated realization of the corresponding error ϵ_i. Then $\hat{s}$ is computed on the reduced model for testing the entry effect and on the extended model for testing the removal effect.

2. Compute quantile-level bandwidth h_n. The QUANTSELECT procedure computes the Bofinger bandwidth, which is an optimizer of mean squared error for standard density estimation:

   $$h_n = n^{-1/5}(4.5v^2(\tau))^{1/5}$$

   The quantity

   $$v(\tau) = \frac{s(\tau)}{s^{(2)}(\tau)} = \frac{f^2}{2(f^{(1)}/f)^2 + [(f^{(1)}/f)^2 - f^{(2)}/f]}$$

   is not sensitive to f and can be estimated by assuming f is Gaussian as

   $$\hat{v}(\tau) = \frac{\exp(-q^2)}{2\pi(q^2+1)} \text{ with } q = \Phi^{-1}(\tau)$$

3. Compute residual quantiles $\hat{F}^{-1}(\tau_0)$ and $\hat{F}^{-1}(\tau_1)$ as follows:

   a) Set $\tau_0 = \max(0, \tau - h_n)$ and $\tau_1 = \min(1, \tau + h_n)$.

   b) Use the equation

   $$\hat{F}^{-1}(t) = \begin{cases} r_{(1)} & \text{if } t \in [0, 1/2n) \\ \lambda r_{(i+1)} + (1-\lambda) r_{(i)} & \text{if } t \in [(i-0.5)/n, (i+0.5)/n) \\ r_{(n)} & \text{if } t \in [(2n-1), 1] \end{cases}$$

   where $r_{(i)}$ is the ith smallest residual and $\lambda = t - (i - 0.5)/n$.

   c) If $\hat{F}^{-1}(\tau_0) = \hat{F}^{-1}(\tau_1)$, find i that satisfies $r_{(i)} < \hat{F}^{-1}(\tau_0)$ and $r_{(i+1)} \geq \hat{F}^{-1}(\tau_0)$. If such an i exists, reset $\tau_0 = (i - 0.5)/n$ so that $\hat{F}^{-1}(\tau_0) = r_{(i)}$. Also find j that satisfies $r_{(j)} > \hat{F}^{-1}(\tau_1)$ and $r_{(j-1)} \leq \hat{F}^{-1}(\tau_1)$. If such a j exists, reset $\tau_1 = (j - 0.5)/n$ so that $\hat{F}^{-1}(\tau_1) = r_{(j)}$.

4. Estimate the sparsity function $s(\tau)$ as

   $$\hat{s}(\tau) = \frac{\hat{F}^{-1}(\tau_1) - \hat{F}^{-1}(\tau_0)}{\tau_1 - \tau_0}$$

Because a real data set might not follow the null hypothesis and the iid assumptions, the LR1 and LR2 scores that are used for quantile regression effect selection often do not follow a χ^2 distribution. Hence, the SLENTRY and SLSTAY values cannot reliably be viewed as probabilities. One way to address this difficulty is to treat the SLENTRY and SLSTAY values only as criteria for comparing importance levels of effect candidates at each selection step, and not to explain these values as probabilities.

Quantile Process Regression

You can specify QUANTILE=PROCESS in the MODEL statement to perform quantile process regression. Quantile process regression fits quantile regression models for the entire range of quantile levels from 0 to 1. Because a quantile function is the inverse of its cumulative distribution function, quantile process regression can estimate the entire distribution of a response variable conditional on its covariates.

Because of the piecewise linearity of the check loss function, the optimal quantile regression solution $\hat{\boldsymbol{\beta}}(\tau)$ is a step function in $\tau \in [0, 1]$. In other words, given any optimal solution $\hat{\boldsymbol{\beta}}(\tau^*)$, there exists an optimal quantile-level range $[\tau_1, \tau_2]$ such that $\hat{\boldsymbol{\beta}}(\tau) = \hat{\boldsymbol{\beta}}(\tau^*)$ is optimal for any $\tau \in [\tau_1, \tau_2]$. This step-function property can simplify integration computation in quantile process regression. For example, to estimate conditional mean by using quantile process regression, you can substitute integration by using the summation

$$E(Y|\mathbf{X} = \mathbf{x}) = \int_0^1 \mathbf{x}\hat{\boldsymbol{\beta}}(\tau)d\tau = \mathbf{x}\sum_{i=1}^{s}(\tau_{i+1} - \tau_i)\hat{\boldsymbol{\beta}}_i$$

where $\tau_1 = 0$, $\tau_{s+1} = 1$, and $\hat{\boldsymbol{\beta}}_i$ is the optimal solution for quantile range $[\tau_i, \tau_{i+1}]$.

If you specify the N=ALL suboption in the QUANTILE=PROCESS option, PROC QUANTSELECT outputs $\hat{\boldsymbol{\beta}} = \sum_{i=1}^{s}(\tau_{i+1} - \tau_i)\hat{\boldsymbol{\beta}}_i$ as the mean parameter estimates in the parameter estimates table. If you request the "Parameter Estimates for Quantile Process" table, PROC QUANTSELECT outputs parameter estimates in the following quantile-level grid:

$$\left\{0, \frac{\tau_1 + \tau_2}{2}, \frac{\tau_2 + \tau_3}{2}, \ldots, \frac{\tau_s + \tau_{s+1}}{2}, 1\right\}$$

For more information about the "Parameter Estimates for Quantile Process" table, see "Parameter Estimates for Quantile Process" on page 8690. PROC QUANTSELECT also uses this grid to estimate observation quantile levels.

If you specify N=*n*, PROC QUANTSELECT approximates the quantile process regression in the following quantile-level grid:

$$\left\{0, \frac{1}{n+1}, \frac{2}{n+1}, \ldots, 0.5, \ldots, \frac{n}{n+1}, 1\right\}$$

When N=*n*, PROC QUANTSELECT also approximates integrations by using the linear interpolation method, which defines

$$\hat{\boldsymbol{\beta}}(\tau) = \frac{\tau - \tau_1}{\tau_2 - \tau_1}\hat{\boldsymbol{\beta}}(\tau_2) + \frac{\tau_2 - \tau}{\tau_2 - \tau_1}\hat{\boldsymbol{\beta}}(\tau_1)$$

Here,τ_1 and τ_2 denote two consecutive quantile levels in the quantile-level grid that satisfy $\tau \in [\tau_1, \tau_2]$.

Observation Quantile Level

The observation quantile level of a valid observation, $(y, \mathbf{x})$, is defined as $\tau_{(y,\mathbf{x})} = F_{Y|\mathbf{x}}(y)$, where $F_{Y|\mathbf{x}}(\cdot)$ denotes the cumulative distribution function (CDF) for the y's underlying distribution conditional on $\mathbf{x}$. For the CDF that is continuous at y, the equation $y = Q_{Y|\mathbf{x}}\left(\tau_{(y,\mathbf{x})}\right)$ holds because the quantile function is inversely related to the CDF. Ideally, if $y = \mathbf{x}\hat{\boldsymbol{\beta}}(\tau^*)$ for a unique $\tau^* \in [0, 1]$ and some quantile-regression optimal solution $\hat{\boldsymbol{\beta}}(\tau^*)$, then τ^* is a reasonable estimation for $\tau_{(y,\mathbf{x})}$, written as $\hat{\tau}_{(y,\mathbf{x})} = \tau^*$. However, such a τ^* might not exist or is nonunique in practice. The following steps show how the QUANTSELECT procedure estimates the observation quantile level $\tau_{(y,\mathbf{x})}$ via quantile process regression:

1. Fit the quantile process regression model and label its quantile-level grid as follows:
$$\left\{0 = \tau_{(0)} \le \tau_{(1)} \le \cdots \le \tau_{(s)} \le \tau_{(s+1)} = 1\right\}$$
2. Compute quantile predictions conditional on $\mathbf{x}$ in the quantile-level grid: $\left\{q_i = \mathbf{x}\hat{\boldsymbol{\beta}}_i : i = 0, \ldots, s+1\right\}$.
3. Sort q_i's to avoid crossing, such that $q_{(0)} \le q_{(1)} \le \cdots \le q_{(s+1)}$.
4. $\hat{\tau}_{(y,\mathbf{x})} = 0$ if $y < q_{(0)}$, or $\hat{\tau}_{(y,\mathbf{x})} = 1$ if $y > q_{(s+1)}$.
5. Otherwise, search index j such that $q_{(j)} < y < q_{(j+1)}$. If such a j exists,
$$\hat{\tau}_{(y,\mathbf{x})} = \left(\frac{y - q_{(j)}}{q_{(j+1)} - q_{(j)}}\right)\tau_{(j+1)} + \left(\frac{q_{(j+1)} - y}{q_{(j+1)} - q_{(j)}}\right)\tau_{(j)}$$
6. Otherwise, search j and k such that $q_{(j-1)} < y = q_{(j)} = \cdots = q_{(j+k)} < q_{(j+k+1)}$, and set $\hat{\tau}_{(y,\mathbf{x})} = \dfrac{\tau_{(j)} + \tau_{(j+k)}}{2}$. Here, define $q_{(-1)} = -\infty$ and $q_{(s+2)} = \infty$.

Quantile Regression for Extremal Quantile Levels

A quantile level τ is extremal if τ is equal to or approaching 0 or 1. The solution for an extremal quantile-level quantile regression problem can be nonunique because the parameter estimate of the intercept effect can be arbitrarily small or large. In a quantile process regression toward the direction of the specified extremal quantile level, the tightest solution refers to the first solution whose quantile-level range includes the specified extremal quantile level. Among all the valid solutions for an extremal quantile-level quantile regression problem, the tightest solution can generalize the terminology of sample minimum and sample maximum.

The QUANTSELECT procedure computes the tightest solution for an extremal quantile-level quantile regression problem by using the ALGORITHM=SIMPLEX algorithm. If $\tau \in \left[\dfrac{1}{4n}, 1 - \dfrac{1}{4n}\right]$, τ is not extremal. Otherwise, follow these steps:

1. Set $\tau_0 = \dfrac{1}{4n}\left(\text{ or } \tau_0 = \left(1 - \dfrac{1}{4n}\right)\right)$.
2. Compute $\hat{\boldsymbol{\beta}}(\tau_0) = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \displaystyle\sum_{i=1}^{n} \rho_{\tau_0}\left(y_i - \mathbf{x}_i'\boldsymbol{\beta}\right)$.
3. Find the quantile-level lower limit (or upper limit), τ_1, such that $\hat{\boldsymbol{\beta}}(\tau_0)$ is still optimal at τ_1.
4. If $\tau_1 \le \tau$ (or $\tau_1 \ge \tau$), return $\hat{\boldsymbol{\beta}}(\tau_0)$. Otherwise, update $\tau_0 = \tau_1 - c$ (or $\tau_0 = \tau_1 + c$) for a small tolerance $c > 0$, and go to step 2.

Effect Selection Methods

The effect selection methods implemented in PROC QUANTSELECT are specified with the SELECTION= option in the MODEL statement.

Full Model Fitted (NONE)

The complete model specified in the MODEL statement is used to fit the model, and no effect selection is done. You request this method by specifying SELECTION=NONE in the MODEL statement.

Forward Selection (FORWARD)

The forward selection technique begins with just the forced-in covariates and then sequentially adds the effect that most improves the fit. The process terminates when no significant improvement can be obtained by adding any effect. You request this method by specifying SELECTION=FORWARD in the MODEL statement.

If you specify the SELECT=SL *method-option*, you can use the TEST= *method-option* to specify a test statistic for gauging improvement in fit. For example, if TEST=LR1, at each step the effect that yields the most significant likelihood ratio statistic is added and the process continues until all effects that are not in the model have LR1 statistics that are not significant at the entry significance level (which is specified in the SLE= option). Because effects can contribute different degrees of freedom to the model, it is necessary to compare the p-values that correspond to these statistics.

Backward Elimination (BACKWARD)

The backward elimination technique starts from the full model, which includes all independent effects. Then effects are deleted one by one until a stopping condition is satisfied. At each step, the effect that shows the smallest contribution to the model is deleted. You request this method by specifying SELECTION=BACKWARD in the MODEL statement.

Suppose you specify the SELECT=SL *method-option* and the TEST=LR1 *method-option* to gauge improvement in quantile regression fit. At any step, the predictor that produces the least significant LR1 statistic is dropped and the process continues until all effects that remain in the model have LR1 statistics that are significant at the stay significance level (which is specified in the SLS= option).

Stepwise Selection (STEPWISE)

The stepwise method is a modification of the forward selection technique in which effects already in the model do not necessarily stay there. You request this method by specifying SELECTION=STEPWISE in the MODEL statement.

In the implementation of the stepwise selection method, the same entry and removal approaches for the forward selection and backward elimination methods are used to assess contributions of effects as they are added to or removed from a model. Suppose you specify SELECT=SL. If, at a step of the stepwise method, any effect in the model is not significant at the level specified by the SLSTAY= *method-option*, then the least significant of these effects is removed from the model and the algorithm proceeds to the next step. This ensures that no effect can be added to a model while some effect currently in the model is not deemed significant. Only after all necessary deletions have been accomplished can another effect be added to the model. In this case, the effect whose addition yields the most significant statistic value is added to the model

and the algorithm proceeds to the next step. The stepwise process ends when none of the effects outside the model is significant at the level specified by the SLENTRY= *method-option* and every effect in the model is significant at the level specified by the SLSTAY= *method-option*. In some cases, neither of these two conditions for stopping is met and the sequence of models cycles. In this case, the stepwise method terminates at the end of the second cycle.

Just as with forward selection and backward elimination, you can use the SELECT= *method-option* to change the criterion used to assess effect contributions. You can also use the STOP= *method-option* to specify a stopping criterion and use the CHOOSE= *method-option* to specify a criterion used to select among the sequence of models produced.

LASSO Method (LASSO)

The standard LASSO method uses a standardized design matrix that orthogonalizes selectable covariates against forced-in covariates, and then scales the orthogonalized selectable covariates so that they all have the same sum of squares. See the information about the standard parameter estimate in the section "Parameter Estimates" on page 8690 for more information about design matrix orthogonalization. The LASSO method initializes all the selectable coefficients into 0 at step 0. The predictor that reduces the average check loss the fastest relative to the L1-norm of the selectable coefficient increment is determined, and a step is taken in the direction of this predictor.

The difference between adaptive LASSO and standard LASSO methods is in the prescaling of the selectable coefficients. After orthogonalization against forced-in covariates, the adaptive LASSO method first fits a full model without penalty, and then scales the orthogonalized selectable covariates with the corresponding coefficients from the full model. This adaptive scaling can be equivalently substituted by using a weighted L1-norm penalty, where the weights are the reciprocals of the corresponding coefficients from the full model.

The length of this step determines the coefficient of this predictor and is chosen when some residual changes its sign or some predictor that is not used in the model can reduce the average check loss more efficiently. This process continues until all predictors are in the model.

As with other selection methods, the issue of when to stop the selection process is crucial. You can use the CHOOSE= *method-option* to specify a criterion for choosing among the models at each step. You can also use the STOP= *method-option* to specify a stopping criterion. See the section "Criteria Used in Model Selection Methods" on page 8680 for more information and Table 104.11 for the formulas for evaluating these criteria.

Criteria Used in Model Selection Methods

PROC QUANTSELECT supports a variety of fit statistics that you can specify as criteria for the CHOOSE=, SELECT=, and STOP= *method-options* in the MODEL statement.

Single Quantile Effect Selection

The following fit statistics are available for single quantile effect selection:

AIC applies the Akaike's information criterion (Akaike 1981; Darlington 1968; Judge et al. 1985).

AICC applies the corrected Akaike's information criterion (Hurvich and Tsai 1989).

SBC applies the Schwarz Bayesian information criterion (Schwarz 1978; Judge et al. 1985).

SL<(LR1 | LR2)> specifies the significance level of a statistic used to assess an effect's contribution to the fit when it is added to or removed from a model. LR1 specifies likelihood ratio Type I, and LR2 specifies the likelihood ratio Type II. By default, the LR1 statistic is applied.

ADJR1 applies the adjusted quantile regression R statistic.

VALIDATE applies the average check loss for the validation data.

Table 104.11 provides formulas and definitions for these fit statistics.

Table 104.11 Formulas and Definitions for Model Fit Summary Statistics for Single Quantile Effect Selection

Statistic	Definition or Formula
n	Number of observations
p	Number of parameters including the intercept
$r_i(\tau)$	Residual for the ith observation; $r_i(\tau) = y_i - \mathbf{x}_i\boldsymbol{\beta}(\tau)$
$D(\tau)$	Total sum of check losses; $D(\tau) = \sum_{i=1}^{n} \rho_\tau(r_i)$
$D_0(\tau)$	Total sum of check losses for intercept-only model if intercept is a forced-in effect, otherwise for empty-model
$\mathrm{ACL}(\tau)$	Average check loss; $\mathrm{ACL}(\tau) = \frac{D(\tau)}{n}$
$\mathrm{R1}(\tau)$	Counterpart of linear regression R-square for quantile regression; $1 - \frac{D(\tau)}{D_0(\tau)}$
$\mathrm{ADJR1}(\tau)$	Adjusted R1; $\mathrm{ADJR1}(\tau) = 1 - \frac{(n-1)D(\tau)}{(n-p)D_0(\tau)}$
$\mathrm{AIC}(\tau)$	Akaike's information criterion; $\mathrm{AIC}(\tau) = 2n\ln(\mathrm{ACL}(\tau)) + 2p$
$\mathrm{AICC}(\tau)$	Corrected Akaike's information criterion; $\mathrm{AICC}(\tau) = 2n\ln(\mathrm{ACL}(\tau)) + \frac{2pn}{n-p-1}$
$\mathrm{SBC}(\tau)$	Schwarz Bayesian information criterion; $\mathrm{SBC}(\tau) = 2n\ln(\mathrm{ACL}(\tau)) + p\ln(n)$

Quantile Process Effect Selection

The following statistics are available for quantile process effect selection:

AIC specifies Akaike's information criterion (Akaike 1981; Darlington 1968; Judge et al. 1985).

AICC specifies the corrected Akaike's information criterion (Hurvich and Tsai 1989).

SBC specifies Schwarz Bayesian information criterion (Schwarz 1978; Judge et al. 1985).

ADJR1 specifies the adjusted quantile regression R statistic.

VALIDATE specifies average check loss for the validation data.

Table 104.12 provides formulas and definitions for the fit statistics.

Table 104.12 Formulas and Definitions for Model Fit Summary Statistics for Quantile Process Effect Selection

Statistic	Definition or Formula
D	Integral of total sum of check losses; $D = \int_0^1 D(\tau)d\tau$
D_0	Integral of total sum of check losses for intercept-only model or empty-model if the NOINT option is used; $D_0 = \int_0^1 D_0(\tau)d\tau$
ACL	Integral of average check loss; $\text{ACL} = \frac{D}{n}$
R1	Counterpart of linear regression R-square for quantile process regression; $\text{R1} = 1 - \frac{D}{D_0}$
ADJR1	Adjusted R1; $\text{ADJR1} = 1 - \frac{(n-1)D}{(n-p)D_0}$
AIC	Akaike's information criterion; $\text{AIC} = \int_0^1 \text{AIC}(\tau)d\tau$
AICC	Corrected Akaike's information criterion; $\text{AICC} = \int_0^1 \text{AICC}(\tau)d\tau$
SBC	Schwarz Bayesian information criterion; $\text{SBC} = \int_0^1 \text{SBC}(\tau)d\tau$

FQPR Effect Selection

If you use the QUANTILE=FQPR option to perform the fast quantile process regression, the following statistics are available for FQPR effect selection:

AIC specifies Akaike's information criterion (Akaike 1981; Darlington 1968; Judge et al. 1985).

AICC specifies the corrected Akaike's information criterion (Hurvich and Tsai 1989).

SBC specifies Schwarz Bayesian information criterion (Schwarz 1978; Judge et al. 1985).

ADJR1 specifies the adjusted quantile regression R statistic.

VALIDATE specifies average check loss for the validation data.

Table 104.13 provides formulas and definitions for the fit statistics.

Table 104.13 Formulas and Definitions for Model Fit Summary Statistics for FQPR Effect Selection

Statistic	Definition or Formula
q	Number of quantile levels for the FQPR quantile-level grid
τ_i	The ith quantile level of the FQPR quantile-level grid
D	Average of total sum of check losses; $D = \frac{1}{q}\sum_{i=1}^{q} D(\tau_i)$
D_0	Average of total sum of check losses for intercept-only model or empty-model if the NOINT option is used; $D_0 = \frac{1}{q}\sum_{i=1}^{q} D_0(\tau_i)$
ACL	Average of average check loss; $\mathrm{ACL} = \frac{D}{n}$
R1	Counterpart of linear regression R-square for FQPR; $\mathrm{R1} = 1 - \frac{D}{D_0}$
ADJR1	Adjusted R1; $\mathrm{ADJR1} = 1 - \frac{(n-1)D}{(n-p)D_0}$
AIC	Akaike's information criterion; $\mathrm{AIC} = \frac{1}{q}\sum_{i=1}^{q} \mathrm{AIC}(\tau_i)$
AICC	Corrected Akaike's information criterion; $\mathrm{AICC} = \frac{1}{q}\sum_{i=1}^{q} \mathrm{AICC}(\tau_i)$
SBC	Schwarz Bayesian information criterion; $\mathrm{SBC} = \frac{1}{q}\sum_{i=1}^{q} \mathrm{SBC}(\tau_i)$

Macro Variables That Contain Selected Models

PROC QUANTSELECT saves the list of selected effects in a macro variable so that you can use other SAS procedures to perform post-selection analyses. This list does not explicitly include the intercept so that you can use it in the MODEL statement of other SAS/STAT regression procedures.

Table 104.14 describes the macro variables that PROC QUANTSELECT creates. When multiple quantile levels or BY processing are used, one macro variable, indexed by the quantile-level order and the BY group number, is created for each quantile level and BY group combination.

Table 104.14 Macro Variables Created for Subsequent Processing

Macro Variable	Description
Single Quantile Level and No BY processing	
_QRSIND	Selected model
Multiple Quantile Levels and No BY Processing	
_QRSNUMTAUS	Number of quantile levels
_QRSINDT1	Selected model for the first quantile level
_QRSINDT2	Selected model for the second quantile level
...	
Single Quantile Level and BY Processing	
_QRSNUMBYS	Number of BY groups
_QRSIND1	Selected model for BY group 1
_QRSIND2	Selected model for BY group 2
...	
Multiple Quantile Levels and BY Processing	
_QRSNUMTAUS	Number of quantile levels
_QRSNUMBYS	Number of BY groups
_QRSIND1T1	Selected model for the first quantile level and BY group 1
_QRSIND1T2	Selected model for the second quantile level and BY group 1
...	
_QRSIND2T1	Selected model for the first quantile level and BY group 2
_QRSIND2T2	Selected model for the second quantile level and BY group 2
...	

The macro variables _QRSIND, _QRSINDT1, _QRSIND1, and _QRSIND1T1 are all synonyms. If you do not specify multiple quantile levels or BY processing, the macro variables _QRSNUMTAUS and _QRSNUMBYS are both set to 1.

PROC QUANTSELECT creates two output data set variables, _BY_ and _QUANTILE_, to aid in associating macro variables with output data set observations when multiple quantile levels or BY processing are used. The values of these two variables are integers that match the *i*,*j* components of the macro variable names _QRSIND*i*T*j*.

Using Validation and Test Data

When you have sufficient data, you can subdivide your data into three parts: training, validation, and test data. During the selection process, models are fit on the training data, and the prediction error for the models so obtained is found by using the validation data. This prediction error on the validation data can be used to decide when to terminate the selection process or to decide which effects to include as the selection process proceeds. Finally, after a selected model has been obtained, the test set can be used to assess how the selected model generalizes on data that played no role in selecting the model.

In some cases you might want to use only training and test data. For example, you might decide to use an information criterion to decide which effects to include and when to terminate the selection process. In this case no validation data are required, but test data can still be useful in assessing the predictive performance

of the selected model. In other cases you might decide to use validation data during the selection process but forgo assessing the selected model on test data. Hastie, Tibshirani, and Friedman (2001) note that it is difficult to give a general rule for how many observations you should assign to each role. They note that a typical split might be 50% for training and 25% each for validation and testing.

PROC QUANTSELECT provides several methods for partitioning data into training, validation, and test data. You can provide data for each role in separate data sets that you specify with the DATA=, TESTDATA=, and VALDATA= options in the PROC QUANTSELECT procedure. An alternative method is to use a PARTITION statement to logically subdivide the DATA= data set into separate roles. You can name the fractions of the data that you want to reserve as test data and validation data. The following statements randomly subdivide the inData data set to use 25% of the data for validation and 25% for testing, leaving 50% of the data for training:

```
proc quantselect data=inData;
   partition fraction(test=0.25 validate=0.25);
   ...
run;
```

If you need to exercise more control over the partitioning of the input data set, you can name a variable in the input data set and a formatted value of that variable to correspond to each role. The following statements assign roles to observations in the inData data set based on the value of the variable named group in that data set:

```
proc quantselect data=inData;
   partition roleVar=group(test='group 1' train='group 2')
   ...
run;
```

Observations whose value of the variable group is `'group 1'` are assigned for testing, and those whose value is `'group 2'` are assigned to training. All other observations are ignored.

You can also combine the use of the PARTITION statement with named data sets for specifying data roles. For example, the following statements reserve 40% of the inData data set for validation, leaving the remaining 60% for training:

```
proc quantselect data=inData testData=inTest;
   partition fraction(validate=0.4);
   ...
run;
```

Data for testing are supplied in the inTest data set. Because a TESTDATA= data set is specified, additional observations for testing cannot be reserved by specifying a PARTITION statement.

When you use a PARTITION statement, the output data set that is created by an OUTPUT statement contains a character variable _ROLE_ whose values TRAIN, TEST, and VALIDATE indicate the role of each observation. _ROLE_ is blank for observations that were not assigned to any of these three roles. When the input data set specified in the DATA= option in the PROC QUANTSELECT statement contains an _ROLE_ variable, no PARTITION statement is used, and the TESTDATA= and VALDATA= options are not specified, then the _ROLE_ variable is used to define the roles of each observation. This is useful when you want to rerun PROC QUANTSELECT but use the same data partitioning as you used in a previous PROC QUANTSELECT step. For example, the following statements use the same data for testing and training in both PROC QUANTSELECT steps:

```
proc quantselect data=inData;
   partition fraction(test=0.5);
   model y=x1-x10 / selection=forward;
   output out=outDataForward;
run;

proc quantselect data=outDataForward;
   model y=x1-x10 / selection=backward;
run;
```

When you have reserved observations for training, validation, and testing, a model that is fit on the training data is scored on the validation and test data, and the average check loss, denoted by ACL, is computed separately for each of these subsets. The ACL for each data role is the sum of check losses for observations in that role divided by the number of observations in that role.

Using the Validation ACL as the STOP= Criterion

If you have provided observations for validation, then you can use the STOP=VALIDATE *method-option* to specify the validation ACL as the STOP= criterion in the SELECTION= option in the MODEL statement. At step k of the selection process, the best candidate effect to enter or leave the current model is determined. The "best candidate" means the effect that gives the best value of the SELECT= criterion that does not need to be based on the validation data. The validation ACL for the model with this candidate effect added is computed. If this validation ACL is greater than the validation ACL for the model at step k, then the selection process terminates at step k.

Using the Validation ACL as the CHOOSE= Criterion

When you specify the CHOOSE=VALIDATE *method-option* in the SELECTION= option in the MODEL statement, the validation ACL is computed for the models at each step of the selection process. The model that yields the smallest validation ACL and contains the fewest effects is selected.

Using the Validation ACL as the SELECT= Criterion

You request the validation ACL as the selection criterion by specifying the SELECT=VALIDATE *method-option* in the SELECTION= option in the MODEL statement. At step k of the selection process, the validation ACL is computed for each model where a candidate for entry is added or candidate for removal is dropped. The selected candidate for entry or removal is the one that yields a model with the minimal validation ACL.

Displayed Output

The following sections describe the output that is displayed by PROC QUANTSELECT. The output is organized into various tables, which are discussed in the order of appearance. The contents of a table might change depending on the options you specify.

Model Information

The "Model Information" table displays basic information about the data sets and the settings used to control effect selection. These settings include the following:

- the selection method
- the criteria used to select effects, stop the selection, and choose the selected model
- the effect hierarchy enforced

The ODS name of the "Model Information" table is ModelInfo.

Number of Observations

The "Number of Observations" table displays the number of observations read from the input data set and the number of observations used in the analysis. If you use a PARTITION statement, the table also displays the number of observations used for each data role. If you specify TESTDATA= or VALDATA= data sets in the PROC QUANTSELECT statement, then "Number of Observations" tables are also produced for these data sets. The ODS name of the "Number of Observations" table is NObs.

Class Level Information

The "Class Level Information" table lists the levels of every variable specified in the CLASS statement. The ODS name of the "Class Level Information" table is ClassLevelInfo.

Class Level Coding

The "Class Level Coding" table shows the coding used for every variable specified in the CLASS statement. The ODS name of the "Class Level Coding" table is ClassLevelCoding.

Dimensions

The "Dimensions" table displays information about the number of effects and the number of parameters from which the selected model is chosen. If you use split classification variables, then this table also includes the number of effects after splitting is taken into account. The ODS name of the "Dimensions" table is Dimensions.

Candidates

The "Candidates" table displays the effect name and value of the criterion used to select entering or departing effects at each step of the selection process. The effects are displayed in sorted order from best to worst of the selection criterion. You request this table with the DETAILS= option in the MODEL statement. The ODS name of the "Candidates" table is either EntryCandidates for addition candidates or RemovalCandidates for removal candidates.

Selection Summary

The "Selection Summary" table displays details about the sequence of steps of the selection process. For each step, the effect that entered or dropped out is displayed along with the statistics used to select the effect, stop the selection, and choose the selected model. You can request that additional statistics be displayed with the STATS= option in the MODEL statement. For all criteria that you can use for effect selection, the steps at which the optimal values of these criteria occur are also indicated. The ODS name of the "Selection Summary" table is SelectionSummary.

Stop Reason

The "Stop Reason" table displays the reason why the selection stopped. To facilitate programmatic use of this table, an integer code is assigned to each reason and is included if you output this table by using an ODS OUTPUT statement. The reasons and their associated codes follow:

Code	Stop Reason
1	All eligible effects are in the model.
2	All eligible effects have been removed.
3	Specified maximum number of steps done.
4	The model contains the specified maximum number of effects.
5	The model contains the specified minimum number of effects (for backward selection).
6	The stopping criterion is at a local optimum.
7	No suitable add or drop candidate could be found.
8	Adding or dropping any effect does not improve the selection criterion.
9	No candidate meets the appropriate SLE or SLS significance level.
10	Stepwise selection is cycling.
11	The model is an exact fit.
12	Dropping an effect would result in an empty model.

The ODS name of the "Stop Reason" table is StopReason.

Selection Reason

The "Selection Reason" table displays how the final selected model is determined. Table 104.15 shows the possible selection reasons:

Table 104.15 Selection Reasons

Selection Reason	Description
1	The last valid model that occurs in the selection process is the final model.
2	The first model with the minimum CHOOSE= criterion value in the selection process is the final model.

The ODS name of the "Selection Reason" table is SelectionReason.

Selected Effects

The "Selected Effects" table displays a string that contains the list of effects in the selected model. The ODS name of the "Selected Effects" table is SelectedEffects.

Fit Statistics

The "Fit Statistics" table displays fit statistics for the selected model. The statistics displayed include the following:

- OBJ, the sum of check losses. It is calculated as the minimized objective function value for the fit.
- R1, a measure between 0 and 1 that indicates the portion of the (corrected) total variation attributed to the fit rather than left to residual error. It is calculated as one minus OBJ(Model) divided by OBJ(Total).
- Adj R1, the adjusted $R1$, a version of $R1$ that has been adjusted for degrees of freedom. It is calculated as

  $$\bar{R}1 = 1 - \frac{(n-i)(1-R1)}{n-p}$$

  where i is equal to 1 if there is an intercept and 0 otherwise, n is the number of observations used to fit the model, and p is the number of parameters in the model.
- fit criteria AIC, AICC, and SBC.
- the average check losses (ACL) on the training, validation, and test data. See the section "Using Validation and Test Data" on page 8684 for details.

You can request "Fit Statistics" tables for the models at each step of the selection process with the DETAILS= option in the MODEL statement. The ODS name of the "Fit Statistics" table is FitStatistics.

Parameter Estimates

The "Parameter Estimates" table displays the parameters in the selected model and their estimates. The following information is displayed for each parameter in the selected model:

- the parameter label that includes the effect name and level information for effects that contain classification variables
- the degrees of freedom (DF) for the parameter. There is one degree of freedom unless the model is not full rank.
- the parameter estimate
- the standard parameter estimate, which is computed on a standardized design matrix. Let $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2)$ denote the original design matrix, where $\mathbf{X}_1$ is the submatrix for all the forced-in effects, and $\mathbf{X}_2$ is the submatrix for the rest of the effects that are subject to selection. Let

$$\mathbf{X}_2^* = \left[\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1{}'\mathbf{X}_1)^{-1}\mathbf{X}_1{}'\right]\mathbf{X}_2 \text{ and } \mathbf{X}_2^{**} = s_Y \mathbf{X}_2^* \left[\frac{\mathrm{diag}(\mathbf{X}_2^{*\prime}\mathbf{X}_2^*)}{n - p_1}\right]^{-\frac{1}{2}}$$

  where p_1 is the rank of $\mathbf{X}_1$ and $s_Y = \sqrt{\frac{\mathbf{Y}^{*\prime}\mathbf{Y}^*}{n-p_1}}$ with $\mathbf{Y}^* = \left[\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1{}'\mathbf{X}_1)^{-1}\mathbf{X}_1{}'\right]\mathbf{Y}$.

  Then standard parameter estimates are defined as $(\mathbf{0}, \boldsymbol{\beta}_2^{**})$, where $(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2^{**})$ are the parameter estimates computed on the standardized design matrix $(\mathbf{X}_1, \mathbf{X}_2^{**})$.

You can also use the DETAILS= option in the MODEL statement to request "Parameter Estimates" tables for the models at each step of the selection process. The ODS name of the "Parameter Estimates" table is ParameterEstimates.

Parameter Estimates for Quantile Process

The "Parameter Estimates for Quantile Process" table contains the parameter estimates for the quantile process of the final selected model. The following statements show how you can request the output data set of this table by using the ODS OUTPUT statement:

```
proc quantselect data=Data;
   ods output ProcessEst=outProcessEst;
   model y=x1-x10 / selection=forward quantile=process;
run;
proc print data=outProcessEst;
run;
```

The output data set contains the following variables:

- QuantileLabel, the label of quantile levels
- QuantileLevel, the quantile levels
- variables for parameter estimates

Given the quantile-level grid for the quantile process,

$$\{0 = \tau_{(0)} \leq \tau_{(1)} \leq \cdots \leq \tau_{(s)} \leq \tau_{(s+1)} = 1\}$$

The ith observation in the "Parameter Estimates for Quantile Process" table corresponds to the optimal solution of the ith quantile level in the quantile-level grid. The ith QuantileLabel value is in the form of ti, and the ith QuantileLevel value is equal to $\tau_{(i)}$. For more information about the quantile-level grid, see the section "Quantile Process Regression" on page 8677.

ODS Table Names

PROC QUANTSELECT assigns a name to each table it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 104.16.

For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 104.16 ODS Tables Produced by PROC QUANTSELECT

ODS Table Name	Description	Statement	Option
BSplineDetails	B-spline basis details	EFFECT	DETAILS
Dimensions	Number of effects and parameters	MODEL	Default
EntryCandidates	Entry effect ranking	MODEL	DETAILS=
FitStatistics	Selected model fit statistics	MODEL	Default
RemovalCandidates	Removal effect ranking	MODEL	DETAILS=
ClassLevelCoding	Classification variable coding	CLASS	SHOWCODING
ClassLevelInfo	Classification variable levels	CLASS	Default
CollectionLevelInfo	Levels of collection effects	EFFECT	DETAILS
MMLevelInfo	Levels of multimember effects	EFFECT	DETAILS
ModelInfo	Model information	MODEL	Default
NObs	Number of observations	MODEL	Default
ParameterNames	Labels for column names in the design matrix	PROC QUANTSELECT	OUTDESIGN(names)
ParameterEstimates	Selected model parameter estimates	MODEL	Default
PolynomialDetails	Polynomial details	EFFECT	DETAILS
PolynomialScaling	Polynomial scaling	EFFECT	DETAILS
ProcessEst	Parameter estimates for quantile process	MODEL	QUANTILE
SelectedEffects	List of selected effects	MODEL	Default
SelectionSummary	Selection summary	MODEL	Default
StopReason	Reason why selection stopped	MODEL	Default
TPFSplineDetails	Thin-plate spline basis details	EFFECT	DETAILS

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC QUANTSELECT assigns a name to each graph it creates using ODS. You can use these names to refer to the graphs when using ODS. The names are listed in Table 104.17.

Table 104.17 ODS Graphics Produced by PROC QUANTSELECT

ODS Graph Name	Plot Description	PLOTS Option
ACLPlot	Average check loss by step	ACL
AICCPlot	Corrected Akaike's information criterion by step	CRITERIA(UNPACK)
AICPlot	Akaike's information criterion by step	CRITERIA(UNPACK)
AdjR1Plot	Adjusted quantile regression R by step	CRITERIA(UNPACK)
ChooseCriterionPlot	CHOOSE= criterion by step	COEFFICIENTS(UNPACK)
CoefficientPanel	Coefficients and CHOOSE= criterion by step	COEFFICIENTS
CoefficientPlot	Coefficients by step	COEFFICIENTS(UNPACK)
CriterionPanel	Fit criteria by step	CRITERIA
SBCPlot	Schwarz Bayesian information criterion by step	CRITERIA(UNPACK)
ValidateACLPlot	Average square error on validation data by step	CRITERIA(UNPACK)

Example: QUANTSELECT Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 104.1: Simulation Study

(View the complete code for this example (qrsex1.sas) in the example repository.)

This simulation study exemplifies the unity of motive and effect for the PROC QUANTSELECT procedure. The following statements generate a data set that is based on a naive instrumental model (Chernozhukov and Hansen 2008):

```
%let seed=321;
%let p=20;
%let n=3000;

data analysisData;
   array x{&p} x1-x&p;
   do i=1 to &n;
      U  = ranuni(&seed);
      x1 = ranuni(&seed);
      x2 = ranexp(&seed);
      x3 = abs(rannor(&seed));
      y  = x1*(U-0.1) + x2*(U*U-0.25) + x3*(exp(U)-exp(0.9));
      do j=4 to &p;
         x{j} = ranuni(&seed);
      end;
      output;
   end;
run;
```

Variable U of the data set indicates the true quantile level of the response y conditional on $\mathbf{x} = (x_1, \ldots, x_p)$.

Let $Q_y(\tau|\mathbf{x}) = \mathbf{x}\boldsymbol{\beta}(\tau)$ denote the underlying quantile regression model, where $\boldsymbol{\beta}(\tau) = (\beta_1(\tau), \ldots, \beta_p(\tau))'$. Then, the true parameter functions are

$$\begin{aligned}
\beta_1(\tau) &= \tau - 0.1 \\
\beta_2(\tau) &= \tau^2 - 0.25 \\
\beta_3(\tau) &= \exp(\tau) - \exp(0.9) \\
\beta_4(\tau) &= \ldots = \beta_p(\tau) = 0
\end{aligned}$$

It is easy to see that, at $\tau = 0.1$, only $\beta_2(0.1) = -0.24$ and $\beta_3(0.1) = \exp(0.1) - \exp(0.9) \approx -1.354432$ are nonzero parameters. Therefore, an effective effect selection method should select x_2 and x_3 and drop all the other effects in this data set at $\tau = 0.1$. By the same rationale, x_1 and x_3 should be selected at $\tau = 0.5$ with $\beta_1(0.5) = 0.4$ and $\beta_3(0.5) \approx -0.810882$, and x_1 and x_2 should be selected at $\tau = 0.9$ with $\beta_1(0.9) = 0.8$ and $\beta_2(0.9) = 0.56$.

The following statements use PROC QUANTSELECT with the adaptive LASSO method:

```
proc quantselect data=analysisData;
   model y= x1-x&p / quantile=0.1 0.5 0.9
         selection=lasso(adaptive);
   output out=out p=pred;
run;
```

Output 104.1.1 shows that, by default, the CHOOSE= and STOP= options are both set to SBC.

Output 104.1.1 Model Information

The QUANTSELECT Procedure

Model Information	
Data Set	WORK.ANALYSISDATA
Dependent Variable	y
Selection Method	Adaptive LASSO
Quantile Type	Single Level
Stop Criterion	SBC
Choose Criterion	SBC

The selected effects and the relevant estimates are shown in Output 104.1.2 for $\tau = 0.1$, Output 104.1.3 for $\tau = 0.5$, and Output 104.1.4 for $\tau = 0.9$. You can see that the adaptive LASSO method correctly selects active effects for all three quantile levels.

Output 104.1.2 Parameter Estimates at $\tau = 0.1$

Selected Effects:	Intercept x2 x3

Parameter Estimates			
Parameter	**DF**	**Estimate**	**Standardized Estimate**
Intercept	1	0.011793	0
x2	1	-0.228709	-0.218287
x3	1	-1.379907	-0.784520

Output 104.1.3 Parameter Estimates at $\tau = 0.5$

Selected Effects:	Intercept x1 x3

Parameter Estimates			
Parameter	**DF**	**Estimate**	**Standardized Estimate**
Intercept	1	0.011778	0
x1	1	0.425843	0.118792
x3	1	-0.863316	-0.490822

Output 104.1.4 Parameter Estimates at $\tau = 0.9$

Selected Effects:	Intercept x1 x2

Output 104.1.4 *continued*

Parameter Estimates			
Parameter	DF	Estimate	Standardized Estimate
Intercept	1	-0.007738	0
x1	1	0.782942	0.218407
x2	1	0.576445	0.550177

The QUANTSELECT procedure can perform effect selection not only at a single quantile level but also for the entire quantile process. You can specify the QUANTILE=PROCESS option to do effect selection for the entire quantile process. With the QUANTILE=PROCESS option specified, the ParameterEstimates table produced by the QUANTSELECT procedure actually shows the mean prediction model of y conditional on $\mathbf{x}$. In this simulation study, the true mean model is

$$\mathrm{E}(y|\mathbf{x}) = \mathbf{x}\boldsymbol{\beta}$$

where

$$\begin{aligned}
\beta_1 &= \mathrm{E}(U) - 0.1 = 0.4 \\
\beta_2 &= \mathrm{E}(U^2) - 0.25 \approx 0.083333 \\
\beta_3 &= \mathrm{E}(\exp(U)) - \exp(0.9) \approx -0.741321 \\
\beta_4 &= \ldots = \beta_p = 0
\end{aligned}$$

The following statements perform effect selection for the quantile process with the forward selection method.

```
proc quantselect data=analysisData;
   model y= x1-x&p / quantile=process(n=all)
         selection=forward;
run;
```

Output 104.1.5 shows that, by default, the SELECT= and STOP= options are both set to SBC. The selected effects and the relevant estimates for the conditional mean model are shown in Output 104.1.6.

Output 104.1.5 Model Information

The QUANTSELECT Procedure

Model Information	
Data Set	WORK.ANALYSISDATA
Dependent Variable	y
Selection Method	Forward
Quantile Type	Process
Select Criterion	SBC
Stop Criterion	SBC
Choose Criterion	SBC

Output 104.1.6 Parameter Estimates

Parameter Estimates			
Parameter	DF	Estimate	Standardized Estimate
Intercept	1	0.007833	0
x1	1	0.418825	0.116834
x2	1	0.094791	0.090472
x3	1	-0.785686	-0.446687

Linear regression is the most popular method for estimating conditional means. The following statements show how to select effects with the GLMSELECT procedure, and Output 104.1.7 shows the resulting selected effects and their estimates. You can see that the mean estimates from the QUANTSELECT procedure are similar to those from the GLMSELECT procedure. However, quantile regression can provide detailed distribution information, which is not available from linear regression.

```
proc glmselect data=analysisData;
   model y= x1-x3 / selection=forward(select=sbc stop=sbc choose=sbc);
run;
```

Output 104.1.7 Parameter Estimates

The GLMSELECT Procedure
Selected Model

Parameter Estimates				
Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	-0.010143	0.043129	-0.24
x1	1	0.434553	0.057385	7.57
x2	1	0.114183	0.016771	6.81
x3	1	-0.797194	0.028156	-28.31

Example 104.2: Econometric Growth Data

(View the complete code for this example (qrsex2.sas) in the example repository.)

This example shows how you can use the QUANTREG procedure to further analyze the final selected models from the QUANTSELECT procedure, and how you can find the set of observations for a specified range of quantile levels. The data under investigation contain economic growth rate records for countries during two time periods: 1965–1975 and 1975–1985. This data set comes from a study by Barro and Lee (1994) and is also analyzed in the section "Example 103.2: Quantile Regression for Econometric Growth Data" on page 8608 of Chapter 103, "The QUANTREG Procedure."

The data set contains 161 observations and 16 variables. The variables, which are listed in Table 104.18, include the national GDP growth rates (GDPR), 14 covariates, and a name variable (Country) that identifies the countries in one of the two periods.

Table 104.18 Variables for Econometric Growth Data

Variable	Description
Country	Country's name and time period
GDPR	Annual change of per capita GDP
lgdp2	Initial per capita GDP
mse2	Male secondary education
fse2	Female secondary education
fhe2	Female higher education
mhe2	Male higher education
lexp2	Life expectancy
lintr2	Human capital
gedy2	Education/GDP
ly2	Investment/GDP
gcony2	Public consumption/GDP
lblakp2	Black market premium
pol2	Political instability
ttrad2	Growth rate terms trade
period	Time period

The goal is to compare the effect of the covariates on GDPR at different quantile levels. The following statements perform effect selection at three quantile levels (τ): 0.1, 0.5, and 0.9.

```
data growth;
   length Country$ 22;
   input Country GDPR lgdp2 mse2 fse2 fhe2 mhe2 lexp2 lintr2 gedy2
         Iy2 gcony2 lblakp2 pol2 ttrad2 @@;
   if(index(country,'75')) then period='65-75';
   if(index(country,'85')) then period='75-85';
   datalines;
Algeria75              .0415 7.330 .1320 .0670 .0050 .0220 3.880 .1138 .0382
                       .1898 .0601 .3823 .0833 .1001
Algeria85              .0244 7.745 .2760 .0740 .0070 .0370 3.978 -.107 .0437
                       .3057 .0850 .9386 .0000 .0657
Argentina75            .0187 8.220 .7850 .6200 .0740 .1660 4.181 .4060 .0221
                       .1505 .0596 .1924 .3575 -.011
Argentina85            -.014 8.407 .9360 .9020 .1320 .2030 4.211 .1914 .0243

   ... more lines ...

                       .0654 .1224 .9393 .7022 -.007
Zambia75               .0120 6.989 .3760 .1190 .0130 .0420 3.757 .4388 .0339
                       .3688 .2513 .3945 .0000 -.032
Zambia85               -.046 7.109 .4200 .2740 .0110 .0270 3.854 .8812 .0477
                       .1632 .2637 .6467 .0000 -.033
Zimbabwe75             .0320 6.860 .1450 .0170 .0080 .0450 3.833 .7156 .0337
                       .2276 .0246 .1997 .0000 -.040
Zimbabwe85             -.011 7.180 .2200 .0650 .0060 .0400 3.944 .9296 .0520
                       .1559 .0518 .7862 .7161 -.024
;
```

```
proc quantselect data=growth;
   class period;
   model GDPR = period lgdp2 mse2 fse2 fhe2 mhe2 lexp2
                lintr2 gedy2 Iy2 gcony2 lblakp2 pol2 ttrad2
                / quantile=0.1 0.5 0.9 selection=backward(choose=sbc sh=5);
run;
```

The SELECTION=BACKWARD option specifies the BACKWARD method as the effect selection method, and the CHOOSE=SBC option specifies the Schwarz Bayesian information criterion for choosing the final selected effects. The estimates for the final selected effects are shown in Output 104.2.1 for $\tau = 0.1$, Output 104.2.2 for $\tau = 0.5$, and Output 104.2.3 for $\tau = 0.9$.

Output 104.2.1 Parameter Estimates at $\tau = 0.1$

The QUANTSELECT Procedure
Quantile Level = 0.1

Selected Effects: Intercept period lgdp2 mse2 lexp2 lintr2 ly2 gcony2 lblakp2 pol2 ttrad2

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate
Intercept	1	0.048847	0
period 65-75	1	0.011861	0.238272
lgdp2	1	-0.024613	-0.947421
mse2	1	0.016031	0.554367
lexp2	1	0.033898	0.277298
lintr2	1	-0.001877	-0.192986
ly2	1	0.067877	0.240002
gcony2	1	-0.176072	-0.438350
lblakp2	1	-0.026364	-0.326506
pol2	1	-0.022975	-0.223264
ttrad2	1	0.096604	0.146071

Output 104.2.2 Parameter Estimates at $\tau = 0.5$

Selected Effects: Intercept period lgdp2 mse2 lexp2 lintr2 ly2 gcony2 lblakp2 pol2 ttrad2

Output 104.2.2 *continued*

Parameter Estimates			
Parameter	DF	Estimate	Standardized Estimate
Intercept	1	-0.040264	0
period 65-75	1	0.008913	0.179063
lgdp2	1	-0.025823	-0.993996
mse2	1	0.014161	0.489697
lexp2	1	0.062163	0.508527
lintr2	1	-0.002688	-0.276345
ly2	1	0.068294	0.241476
gcony2	1	-0.096543	-0.240354
lblakp2	1	-0.025265	-0.312892
pol2	1	-0.019387	-0.188396
ttrad2	1	0.150668	0.227819

Output 104.2.3 Parameter Estimates at $\tau = 0.9$

Selected Effects: Intercept lgdp2 mse2 lexp2 lintr2 ly2 gcony2 lblakp2 ttrad2

Parameter Estimates			
Parameter	DF	Estimate	Standardized Estimate
Intercept	1	-0.011162	0
lgdp2	1	-0.032753	-1.260735
mse2	1	0.016583	0.573447
lexp2	1	0.073326	0.599845
lintr2	1	-0.003334	-0.342843
ly2	1	0.063929	0.226041
gcony2	1	-0.089998	-0.224060
lblakp2	1	-0.032253	-0.399439
ttrad2	1	0.213457	0.322760

Comparing the three quantile models, you can see that the final selected models for $\tau = 0.1$ and $\tau = 0.5$ have the same set of selected effects, but the final selected model for $\tau = 0.9$ excludes the effects for time period and political instability. In other words, if a country's annual change in per capita GDP represents the 90% quantile conditional on the explanatory effects, then its GDP growth rate seems consistent for both the 1965–1975 and 1975–1985 periods and resistant to political instability. In addition, if a country's GDP growth rate represents the 50% or less quantile conditional on the explanatory effects, then the country's 1975–1985 GDP growth rate seems lower than its 1965–1975 GDP growth rate, and the effect for political instability has a negative impact on its GDP growth rate.

To further investigate the impact of time period and political instability on GDP growth rate, you can use the QUANTREG procedure to test the final selected effects. In the previous statements, PROC QUANTSELECT creates a macro variable for the final selected model at each of the three quantile levels. For the current example, the macro variable _QRSINDT1 contains the final model at $\tau = 0.1$; _QRSINDT2 contains the final model at $\tau = 0.5$; and _QRSINDT3 contains the final model at $\tau = 0.9$. The following statements show how to use _QRSINDT2 to specify the model for the QUANTREG procedure at $\tau = 0.5$:

```
proc quantreg data=growth;
   class period;
   model GDPR =  &_qrsindt2 / quantile=0.5;
   Time_Period: test  period;
   Political_Instability: test  pol2;
run;
```

Output 104.2.4 shows more information for the final selected model at $\tau = 0.5$. Output 104.2.5 and Output 104.2.6 show the test results for the effects of time period and political instability on GDP growth rate. You can see that both time period and political instability are significant for the $\tau = 0.5$ model.

Output 104.2.4 Parameter Estimates at $\tau = 0.5$

The QUANTREG Procedure

Parameter Estimates

Parameter		DF	Estimate	95% Confidence Limits	
Intercept		1	-0.0403	-0.1529	0.0453
period	65-75	1	0.0089	0.0060	0.0139
period	75-85	0	0.0000	0.0000	0.0000
lgdp2		1	-0.0258	-0.0324	-0.0212
mse2		1	0.0142	0.0068	0.0182
lexp2		1	0.0622	0.0400	0.1199
lintr2		1	-0.0027	-0.0045	-0.0011
ly2		1	0.0683	0.0143	0.1077
gcony2		1	-0.0965	-0.1526	-0.0576
lblakp2		1	-0.0253	-0.0537	-0.0174
pol2		1	-0.0194	-0.0377	-0.0116
ttrad2		1	0.1507	0.0190	0.2436

Output 104.2.5 Test Results at $\tau = 0.5$

Test Time_Period Results

Test	Test Statistic	DF	Chi-Square	Pr > ChiSq
Wald	13.9838	1	13.98	0.0002

Output 104.2.6 Test Results at $\tau = 0.5$

Test Political_Instability Results

Test	Test Statistic	DF	Chi-Square	Pr > ChiSq
Wald	11.0589	1	11.06	0.0009

As mentioned earlier, _QRSINDT1 and _QRSINDT2 are identical, and _QRSINDT3 excludes two effects from _QRSINDT2: time period and political instability. The following statements retest time period and political instability for the final selected model at $\tau = 0.9$:

```
proc quantreg data=growth;
   class period;
   model GDPR =  &_qrsindt2 / quantile=0.9;
   Time_Period:           test period;
   Political_Instability: test pol2;
   Period_and_Political:  test period pol2;
run;
```

Output 104.2.7, Output 104.2.8, and Output 104.2.9 show the test results for the effects of time period and political instability on GDP growth rate at $\tau = 0.9$. You can see that time period, political instability, and their combination are all insignificant for the $\tau = 0.9$ model.

Output 104.2.7 Test Results at $\tau = 0.9$

The QUANTREG Procedure

Test Time_Period Results

Test	Test Statistic	DF	Chi-Square	Pr > ChiSq
Wald	0.0001	1	0.00	0.9941

Output 104.2.8 Test Results at $\tau = 0.9$

Test Political_Instability Results

Test	Test Statistic	DF	Chi-Square	Pr > ChiSq
Wald	0.1238	1	0.12	0.7250

Output 104.2.9 Test Results at $\tau = 0.9$

Test Period_and_Political Results

Test	Test Statistic	DF	Chi-Square	Pr > ChiSq
Wald	0.1367	2	0.14	0.9339

Another interesting question for quantile regression is to find the observations for a certain range of quantile levels. For example, you might want to know which countries are winners in terms of conditional GDP growth rate at the $\tau = 0.9$ level. The following statements compute the $\tau = 0.9$ quantile predictions and then search, sort, and print the list of winner countries:

```
proc quantselect data=growth;
   class period;
   model GDPR = period lgdp2 mse2 fse2 fhe2 mhe2 lexp2
               lintr2 gedy2 Iy2 gcony2 lblakp2 pol2 ttrad2
               / quantile=0.9 selection=backward(choose=sbc sh=5);
   output out=growth90Out p=Pred;
run;

data growth90;
   set growth90Out;
```

```
   drop lgdp2  mse2  fse2  fhe2  mhe2 lexp2   lintr2 gedy2   Iy2
        gcony2 lblakp2  ttrad2;
   where  GDPR-Pred >= -1E-4;
   GdpDiff = GDPR-Pred;
run;

proc sort data=growth90;
   by GdpDiff;
run;
proc print data=growth90;
run;
```

Output 104.2.10 lists the countries whose conditional GDP growth rates are equal to or higher than their $\tau = 0.9$ quantile predictions.

Output 104.2.10 Countries with High Conditional GDP Growth Rates at $\tau = 0.9$ Level

Obs	Country	GDPR	pol2	period	Pred	GdpDiff
1	Canada75	0.0346	0.0047	65-75	0.034600	0.000000
2	Canada85	0.0240	0.0000	75-85	0.024000	0.000000
3	Congo75	0.0464	0.3385	65-75	0.046400	0.000000
4	Cyprus85	0.0709	0.6000	75-85	0.070900	0.000000
5	Finland75	0.0391	0.0000	65-75	0.039100	0.000000
6	Germany_West85	0.0214	0.0000	75-85	0.021400	0.000000
7	Ghana85	-0.0150	0.0500	75-85	-0.015000	0.000000
8	United_States75	0.0155	0.0015	65-75	0.015500	0.000000
9	Yemen85	0.0305	0.0730	75-85	0.030500	0.000000
10	Denmark85	0.0234	0.0000	75-85	0.023010	0.000390
11	Japan75	0.0636	0.0005	65-75	0.062519	0.001081
12	Jordan85	0.0593	0.5000	75-85	0.058201	0.001099
13	Sudan85	0.0007	0.7000	75-85	-0.000919	0.001619
14	Iran75	0.0538	0.0072	65-75	0.051880	0.001920
15	Spain75	0.0457	0.0014	65-75	0.043241	0.002459
16	Egypt85	0.0427	0.5500	75-85	0.038409	0.004291
17	Hong_Kong85	0.0649	0.0000	75-85	0.059040	0.005860
18	Bangladesh85	0.0133	0.6507	75-85	0.006816	0.006484
19	Rwanda75	0.0590	0.0500	65-75	0.050266	0.008734
20	Brazil75	0.0637	0.0011	65-75	0.050749	0.012951
21	Syria75	0.0601	0.2500	65-75	0.046072	0.014028
22	Botswana85	0.0512	0.0000	75-85	0.030626	0.020574

Example 104.3: Pollution and Mortality

(View the complete code for this example (qrsex3.sas) in the example repository.)

This example shows how you can use the PARTITION statement and other options to control the effect selection process. The data for this example come from a study by McDonald and Schwing (1973). The data set contains 60 observations, 15 covariates, and one response variable. The response variable is the total age-adjusted mortality rate for Standard Metropolitan Statistical Areas in 1959–1961.

The following statements fit a median model for mortality rate conditional on a set of climate, demographic, and pollution covariates by using the forward selection method. Because linear terms alone might not be sufficient to fit this model, quadratic terms are also added in the MODEL statement. The FRACTION option of the PARTITION statement requests that 30% of the observations be used for validation and the remaining 70% of the observations for training. The HIER=SINGLE option in the MODEL statement forces the effect selection process to ignore quadratic effect candidates if their corresponding main effects are not in the model. The OUTPUT statement creates a SAS data set named OutData, which contains the variable _ROLE_. This variable shows the role of each observation that the PARTITION statement assigns.

```
data mortality;
   input index aap ajant ajult size65 nph nsch25 nfek ppsm snwp nowk nin3k
   hpi nopi sdpi datm DeathRate;
   label index="the index"
      aap="Average Annual Precipitation"
      ajant="Average January Temperature"
      ajult="Average July Temperature"
      size65="Size of Population older than 65"
      nph="Number of Members per Household"
      nsch25="Number of Years of Schooling for Persons over 25"
      nfek="Number of Households with fully Equipped Kitchens"
      ppsm="Population per Square Mile"
      snwp="Size of the Nonwhite Population"
      nowk="Number of Office Workers"
      nin3k="Number of Families with an Income less than $3000"
      hpi="Hydrocarbon Pollution Index"
      nopi="Nitric Oxide Pollution Index"
      sdpi="Sulfur Dioxide Pollution Index"
      datm="Degree of Atmospheric Moisture"
      DeathRate="Age-Adjusted Death Rate: Deaths per 100,000 Population";
   datalines;
 1  36  27  71   8.1  3.34  11.4  81.5  3243   8.8  42.6  11.7   21
    15   59  59   921.870
 2  35  23  72  11.1  3.14  11.0  78.8  4281   3.6  50.7  14.4    8
    10   39  57   997.875
 3  44  29  74  10.4  3.21   9.8  81.6  4260   0.8  39.4  12.4    6
     6   33  54   962.354
 4  47  45  79   6.5  3.41  11.1  77.5  3125  27.1  50.2  20.6   18
     8   24  56   982.291
 5  43  35  77   7.6  3.44   9.6  84.6  6441  24.4  43.7  14.3   43

   ... more lines ...

    11   42  56  1003.502
```

```
58  45  24  70  11.8  3.25  11.1  79.8  3678   1.0  44.8  14.0    7
     3    8  56   895.696
59  42  83  76   9.7  3.22   9.0  76.2  9699   4.8  42.2  14.5    8
     8   49  54   911.817
60  38  28  72   8.9  3.48  10.7  79.8  3451  11.7  37.5  13.0   14
    13   39  58   954.442
;

ods graphics on;
proc quantselect data=Mortality seed=800 plots=all;
   partition fraction(validate=0.3);
   model DeathRate = aap aap*aap ajant ajant*ajant ajult
      ajult*ajult size65 size65*size65 nph nph*nph nsch25
      nsch25*nsch25 nfek nfek*nfek  ppsm ppsm*ppsm snwp snwp*snwp
      nowk nowk*nowk nin3k nin3k*nin3k hpi hpi*hpi nopi
      nopi*nopi sdpi sdpi*sdpi datm datm*datm
      / quantile=0.5 selection=forward(choose=val sh=8) hier=single;
   output out=OutData p=Pred;
run;

proc print data=OutData(obs=10); run;
```

Output 104.3.1 shows the selection summary. You can see that the best model is at step 13 for validation ACL, step 5 for the SBC, and step 14 for the AIC.

Output 104.3.1 Selection Summary

The QUANTSELECT Procedure
Quantile Level = 0.5

Selection Summary

Step	Effect Entered	Number Effects In	AIC	AICC	SBC	Validation ACL	Adjusted R1
0	Intercept	1	276.5053	276.6005	278.2895	31.7900	0.0000
1	snwp	2	251.6460	251.9387	255.2144	23.9139	0.2455
2	sdpi	3	240.3445	240.9445	245.6971	20.3977	0.3355
3	nopi	4	238.3223	239.3480	245.4591	16.9704	0.3493
4	ppsm	5	239.3875	240.9664	248.3084	16.3677	0.3397
5	aap	6	226.5892	228.8595*	237.2943*	15.7333	0.4272
6	aap*aap	7	227.6860	230.7971	240.1754	14.8892	0.4177
7	ajult	8	228.5136	232.6279	242.7871	14.5477	0.4095
8	nin3k	9	229.4258	234.7199	245.4835	14.3532	0.4001
9	ajant	10	224.7397	231.4063	242.5816	13.5693	0.4276*
10	ppsm*ppsm	11	226.5785	234.8285	246.2046	12.4032	0.4114
11	hpi	12	228.5050	238.5696	249.9153	11.6356	0.3935
12	ajant*ajant	13	229.9796	242.1129	253.1740	11.1214	0.3776
13	nfek	14	231.9208	246.4035	256.8994	10.9947*	0.3573
14	snwp*snwp	15	221.3905*	238.5334	248.1533	13.3735	0.4234
15	ajult*ajult	16	223.3153	243.4635	251.8624	13.0557	0.4033
16	sdpi*sdpi	17	224.8099	248.3483	255.1411	14.2347	0.3847
17	size65	18	226.7756	254.1356	258.8910	14.3067	0.3613
18	nin3k*nin3k	19	223.8621	255.5288	257.7618	14.5143	0.3719
19	nfek*nfek	20	224.0342	260.5559	259.7180	14.6314	0.3591
20	datm	21	222.7062	264.7062	260.1742	15.4604	0.3561

* Optimal Value Of Criterion

Output 104.3.2 shows the selected effects and the relevant estimates.

Output 104.3.2 Parameter Estimates

Selected Effects:	Intercept aap aap*aap ajant ajant*ajant ajult nfek ppsm ppsm*ppsm snwp nin3k hpi nopi sdpi

Output 104.3.2 *continued*

Parameter Estimates			
Parameter	DF	Estimate	Standardized Estimate
Intercept	1	909.689797	0
aap	1	4.634741	0.750747
aap*aap	1	-0.047789	-0.533679
ajant	1	0.009723	0.001962
ajant*ajant	1	-0.020447	-0.389447
ajult	1	-1.672607	-0.146182
nfek	1	-0.323920	-0.030436
ppsm	1	-0.007194	-0.194141
ppsm*ppsm	1	0.000001906	0.534144
snwp	1	3.483423	0.574703
nin3k	1	3.228388	0.252681
hpi	1	-0.401693	-0.351016
nopi	1	0.795823	0.389110
sdpi	1	0.151049	0.152444

Output 104.3.3 shows the progression of the standardized parameter estimates as the selection process proceeds.

Output 104.3.3 Coefficient Panel

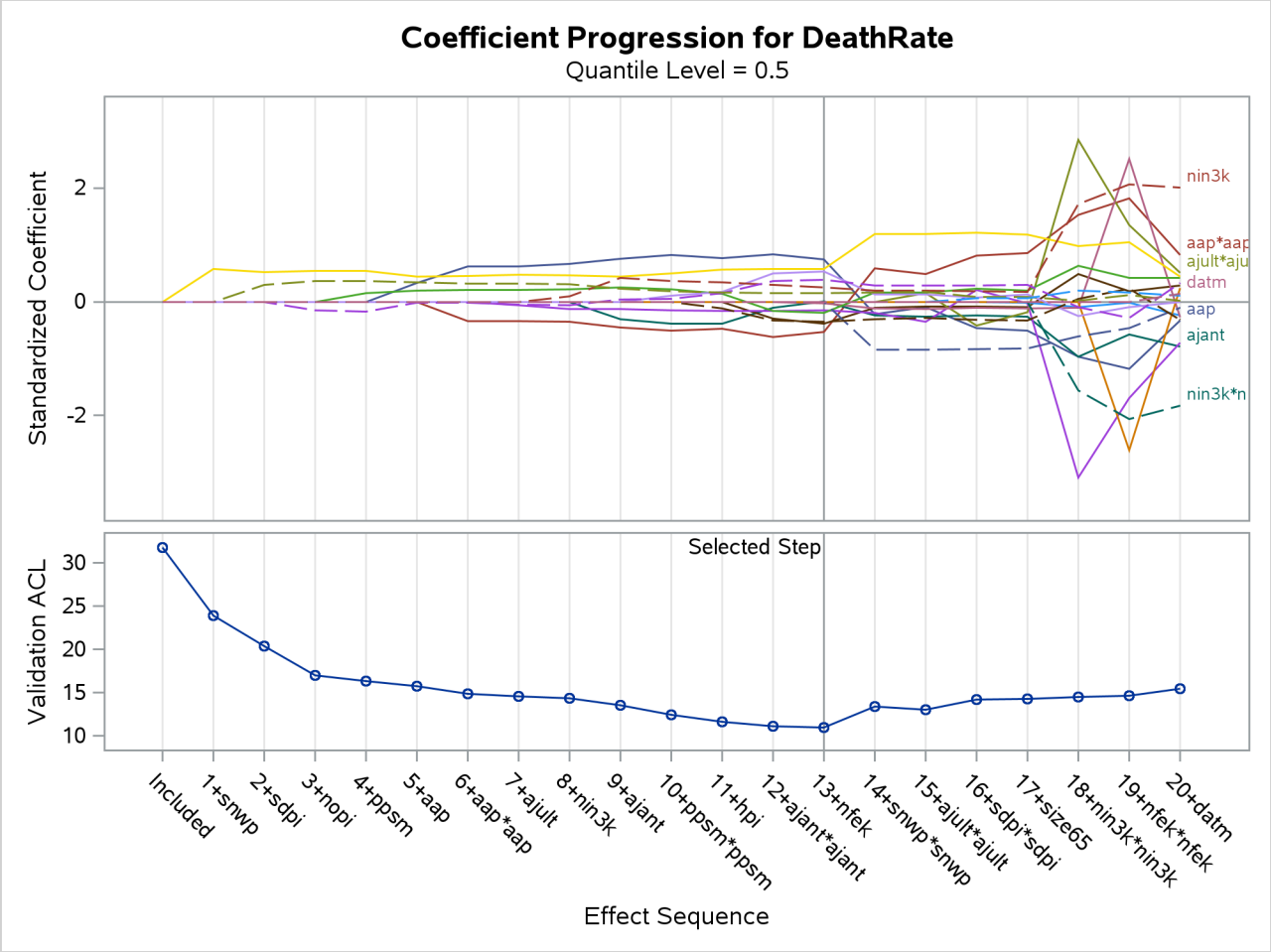

Output 104.3.4 shows the progression of the average check losses for training data and validation data as the selection process proceeds.

Output 104.3.4 Average Check Loss Plot

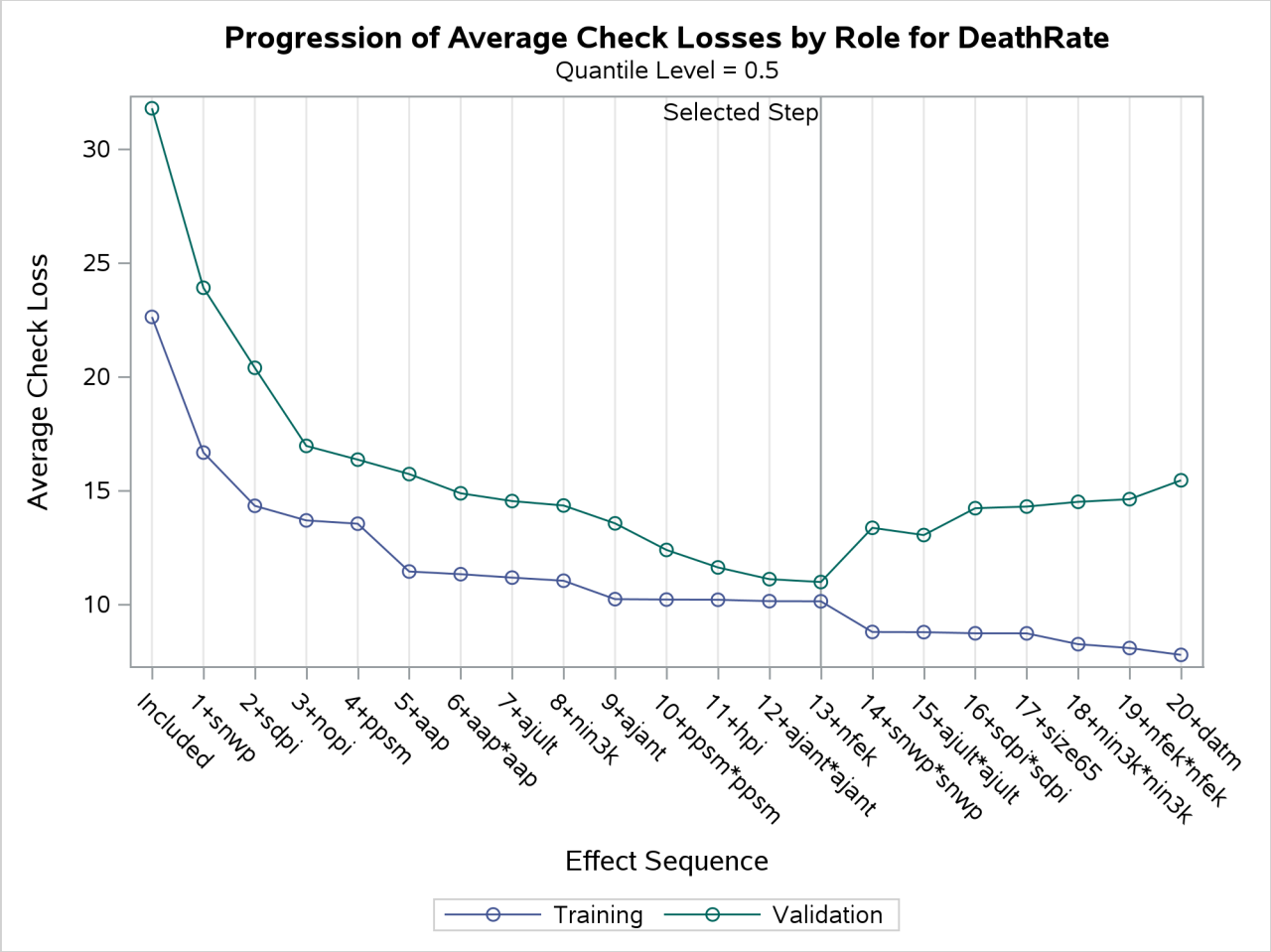

Output 104.3.5 shows the progression of five effect selection criteria as the selection process proceeds.

Output 104.3.5 Criterion Panel

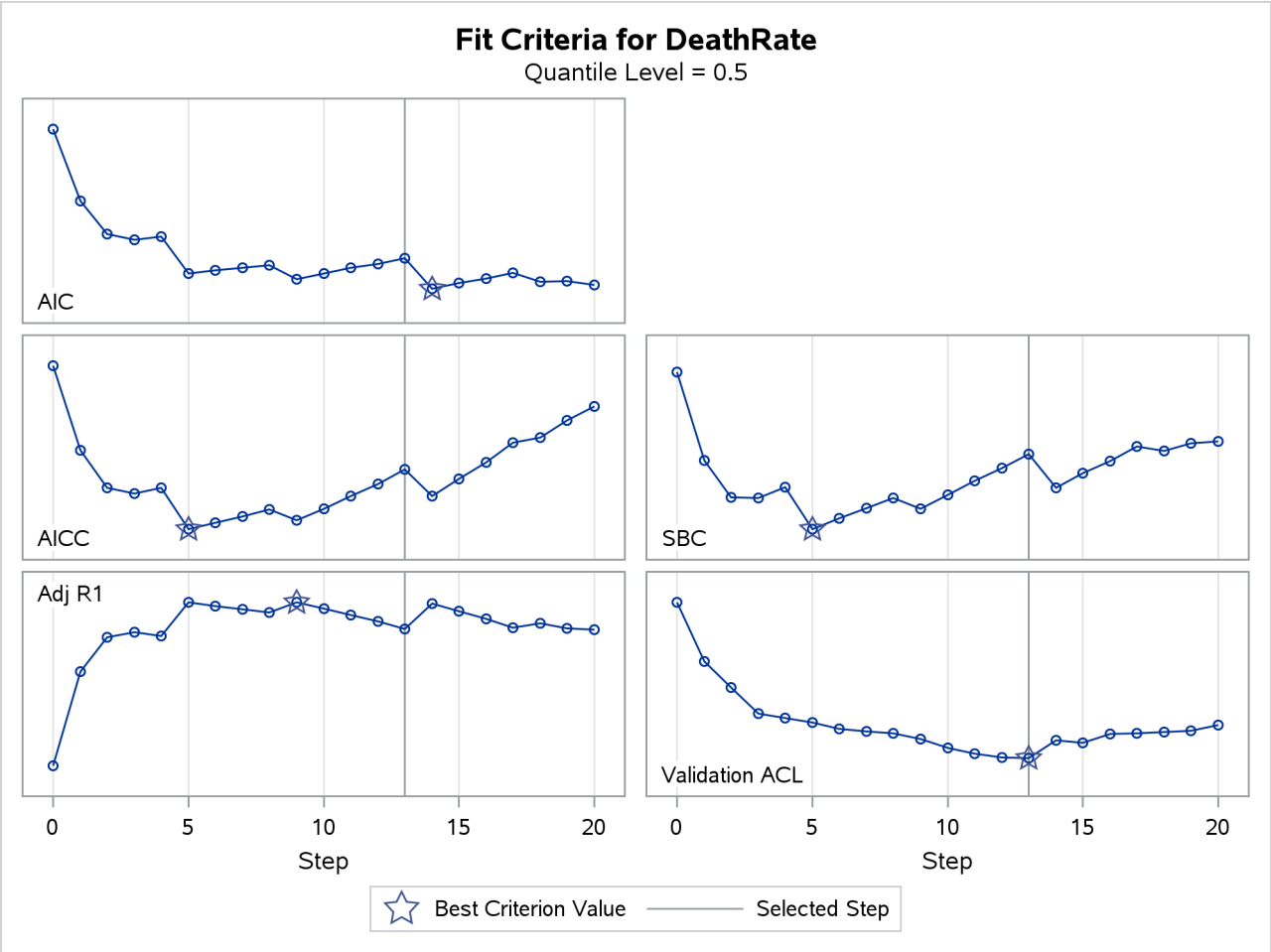

Output 104.3.6 shows the first 10 observations of the OUTPUT data set.

Output 104.3.6 OUTPUT Data Set

Obs	index	aap	ajant	ajult	size65	nph	nsch25	nfek	ppsm	snwp	nowk	nin3k	hpi	nopi	sdpi	datm	DeathRate	Pred	_ROLE_
1	1	36	27	71	8.1	3.34	11.4	81.5	3243	8.8	42.6	11.7	21	15	59	59	921.87	932.36	TRAIN
2	2	35	23	72	11.1	3.14	11.0	78.8	4281	3.6	50.7	14.4	8	10	39	57	997.88	930.62	VALIDATE
3	3	44	29	74	10.4	3.21	9.8	81.6	4260	0.8	39.4	12.4	6	6	33	54	962.35	908.09	TRAIN
4	4	47	45	79	6.5	3.41	11.1	77.5	3125	27.1	50.2	20.6	18	8	24	56	982.29	983.55	TRAIN
5	5	43	35	77	7.6	3.44	9.6	84.6	6441	24.4	43.7	14.3	43	38	206	55	1071.29	1047.71	VALIDATE
6	6	53	45	80	7.7	3.45	10.2	66.8	3325	38.5	43.1	25.5	30	32	72	54	1030.38	1062.56	TRAIN
7	7	43	30	74	10.9	3.23	12.1	83.9	4679	3.5	49.2	11.3	21	32	62	56	934.70	934.70	TRAIN
8	8	45	30	73	9.3	3.29	10.6	86.0	2140	5.3	40.4	10.5	6	4	4	56	899.53	900.48	TRAIN
9	9	36	24	70	9.0	3.31	10.5	83.2	6582	8.1	42.5	12.6	18	12	37	61	1001.90	971.06	TRAIN
10	10	36	27	72	9.5	3.36	10.7	79.3	4213	6.7	41.0	13.2	12	7	20	59	912.35	927.10	VALIDATE

Example 104.4: Surface Fitting with Many Noisy Variables

(View the complete code for this example (qrsex4.sas) in the example repository.)

This example is based on "Example 28.1: Surface Fitting with Many Noisy Variables" on page 1096 in Chapter 28, "The ADAPTIVEREG Procedure." This example shows how you can use the EFFECT statement to select a nonlinear surface model for a data set that contains many nuisance variables.

Consider a simulated data set that contains a response variable and 10 continuous predictors. Each continuous predictor is sampled independently from the uniform distribution $U(0,1)$. The true model of the artificial data set depends nonlinearly on two variables x1 and x2:

$$y = \frac{40\exp\left(8\left((x_1-0.5)^2+(x_2-0.5)^2\right)\right)}{\exp\left(8\left((x_1-0.2)^2+(x_2-0.7)^2\right)\right)+\exp\left(8\left((x_1-0.7)^2+(x_2-0.2)^2\right)\right)}$$

The values of the response variable are generated by adding errors from the standard normal distribution $N(0,1)$ to the true model. The generating mechanism is adapted from Gu et al. (1990). The following statements create an artificial data set that contains 400 observations for the purpose of effect selection and 10,201 observations of missing response values for the purpose of prediction:

```
%let p=10;
data artificial;
   drop i;
   array x{&p};
   do i=1 to 400;
      do j=1 to &p;
         x{j} = ranuni(1);
      end;
      yTrain = 40*exp(8*((x1-0.5)**2+(x2-0.5)**2))/
           (exp(8*((x1-0.2)**2+(x2-0.7)**2))+
           exp(8*((x1-0.7)**2+(x2-0.2)**2)))+rannor(1);
      output;
   end;

   yTrain = .;
   do x1=0 to 1 by 0.01;
      do x2 = 0 to 1 by 0.01;
         y = 40*exp(8*((x1-0.5)**2+(x2-0.5)**2))/
              (exp(8*((x1-0.2)**2+(x2-0.7)**2))+
              exp(8*((x1-0.7)**2+(x2-0.2)**2)));
         output;
      end;
   end;
run;
```

The variables x3 through x10 are nuisance variables that can cause overfitting in your analysis. The following statements invoke the QUANTSELECT procedure to select effects, fit a model on the selected effects, and output the model predictions to an output data set Out:

```
%macro art;
   proc quantselect data=artificial algorithm=smooth;
      %do i=1 %to &p;
         effect sp&i = spline(x&i);
```

```
      %end;
      model yTrain =
         sp1 %do i=2 %to &p; |sp&i %end; @2/details=all;
      output out=Out p=pred;
   run;
%mend;

%art;
```

You can use the EFFECT statement to generate nonlinear effects and model a nonlinear surface. This example uses spline effects on variables and includes all the two-way interactions among these spline effects.

The ALGORITHM=SMOOTH option specifies the smoothing algorithm for model fitting. It takes approximately 2.8 seconds to select the model on a PC that has an Intel i7-2600 quad-core CPU and 64-bit Windows 7 Enterprise operation system. If you use the ALGORITHM=SIMPLEX option, which is default, it takes approximately 8.7 seconds for the same computation settings.

Output 104.4.1 shows the model information. By default, the effect selection method is the stepwise method, and the selection criterion is SBC for the SELECT=, CHOOSE=, and STOP= options. The default quantile level is 0.5 for median regression.

Output 104.4.1 Model Information

The QUANTSELECT Procedure

Model Information	
Data Set	WORK.ARTIFICIAL
Dependent Variable	yTrain
Selection Method	Stepwise
Quantile Type	Single Level
Select Criterion	SBC
Stop Criterion	SBC
Choose Criterion	SBC

Output 104.4.2 shows the best 10 entry candidates at the selection step. You can see that sp1*sp2 is the most important effect, followed by sp1 and sp2.

Output 104.4.2 Best 10 Entry Candidates at Step 1

Best 10 Entry Candidates		
Rank	**Effect**	**SBC**
1	**sp1*sp2**	-496.6752
2	**sp1**	165.9104
3	**sp2**	178.2126
4	**sp3**	213.4593
5	**sp6**	220.8471
6	**sp7**	222.0916
7	**sp9**	224.3185
8	**sp4**	224.7100
9	**sp8**	226.8373
10	**sp5**	227.2176

Output 104.4.3 shows the selection summary.

Output 104.4.3 Selection Summary

The QUANTSELECT Procedure
Quantile Level = 0.5

Selection Summary

Step	Effect Entered	Number Effects In	Number Parms In	SBC
0	Intercept	1	1	195.8108
1	sp1*sp2	2	49	-496.6752*

* Optimal Value Of Criterion

The following statements produce a graph that shows both the true model and the fitted model:

```
ods graphics on;
data pred;
   set out;
   where yTrain=.;
run;

%let off0 = offsetmin=0 offsetmax=0;
%let off0 = xaxisopts=(&off0) yaxisopts=(&off0);
%let eopt = location=outside valign=top textattrs=graphlabeltext;
proc template;
   define statgraph surfaces;
      begingraph / designheight=360px;
         layout lattice/columns=2;
            layout overlay / &off0;
               entry "True Model" / &eopt;
               contourplotparm z=y y=x2 x=x1;
            endlayout;
            layout overlay / &off0;
               entry "Fitted Model" / &eopt;
               contourplotparm z=pred y=x2 x=x1;
            endlayout;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=pred template=surfaces;
run;
```

Output 104.4.4 displays surfaces for both the true model and the fitted model. You can see that the fitted model nicely approximates the underlying true model.

Output 104.4.4 True Model and Fitted Model

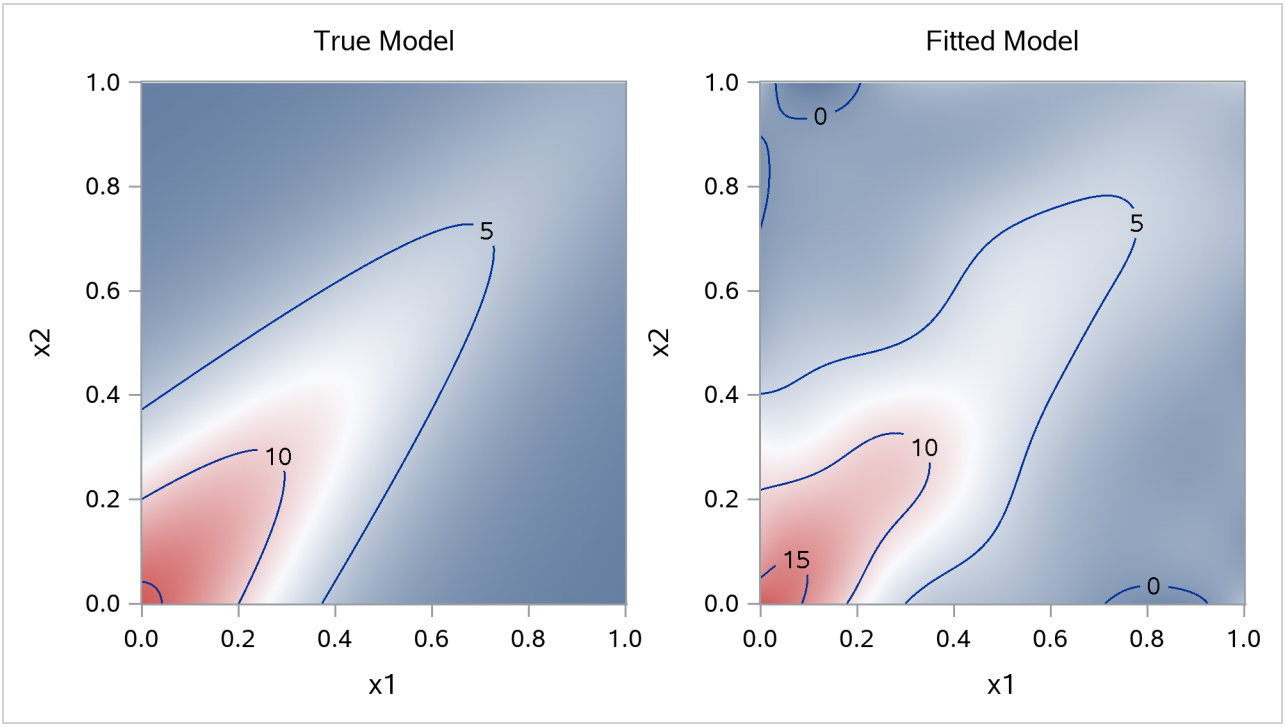

Example 104.5: Quantile Process Regression

(View the complete code for this example (qrsex5.sas) in the example repository.)

Quantile process regression fits quantile regression models for the entire range of quantile levels from 0 to 1, which can estimate the entire probability distribution of a response variable conditional on its covariates. This example demonstrates how you can conduct quantile process regression analysis by using the QUANTSELECT procedure.

Parameter Estimates for Quantile Process Regression

The following statements simulate the data set analysisData, solve a quantile process regression problem on the data set, and store the quantile process parameter estimates in the data set quantProcessEst. The data set analysisData contains one response y and two covariates, x1 and x2. The distribution of y conditional on x1 and x2 gradually changes from a normal distribution to an exponential distribution.

```
%let seed=123;
%let n=6001;
%let model=x1 x2;
data analysisData;
   do i=1 to &n;
      x1=(i-1)/(&n-1);
      x2=(1-x1)*(1-x1);
      y =x1*ranexp(&seed)+x2*(rannor(&seed)-3);
      output;
```

```
   end;
run;

proc quantselect data=analysisData;
   ods output ProcessEst=quantProcessEst;
   model y = &model / quantile=process(n=all) selection=none;
run;

proc print data=quantProcessEst(obs=10);
run;
```

Output 104.5.1 shows the first 10 parameter estimates of the quantile process. For more information about parameter estimates for the quantile process, see the section "Parameter Estimates for Quantile Process" on page 8690.

Output 104.5.1 Quantile Process Parameter Estimates

Obs	QuantileLabel	QuantileLevel	Intercept	x1	x2
1	t0	0.000000	-1.3799	1.3982	-4.3156
2	t1	0.000121	-1.3799	1.3982	-4.3156
3	t2	0.000248	-0.8210	0.8386	-5.0420
4	t3	0.000283	0.2030	-0.2006	-6.3677
5	t4	0.000322	0.3065	-0.3098	-6.5001
6	t5	0.000436	0.4713	-0.4854	-6.7102
7	t6	0.000541	0.2788	-0.2812	-6.2666
8	t7	0.000592	0.1957	-0.1939	-6.0724
9	t8	0.000760	0.1940	-0.1915	-6.0704
10	t9	0.000880	0.1987	-0.1963	-6.0765

To reduce the computation complexity, you can also approximate the quantile process regression by using the N=*n* suboption of the QUANTILE=PROCESS option. The following statements approximate the quantile process by using N=10:

```
proc quantselect data=analysisData;
   ods output ProcessEst=quantApproxProcessEst;
   model y = &model / quantile=process(n=10) selection=none;
run;

proc print data=quantApproxProcessEst;
run;
```

Output 104.5.2 shows the approximate quantile process parameter estimates. If you specify the N=*n* option, the approximate quantile process is computed at *n* equally spaced quantile levels: $\left\{\frac{1}{n+1}, \ldots, \frac{n}{n+1}\right\}$ besides three control quantile levels $\{0, 0.5, 1\}$.

Output 104.5.2 Approximate Quantile Process Parameter Estimates

Obs	QuantileLabel	QuantileLevel	Intercept	x1	x2
1	t0	0.000000	-1.3799	1.3982	-4.3156
2	t1	0.090909	0.6051	-0.5449	-4.8257
3	t2	0.181818	0.5273	-0.3595	-4.3719
4	t3	0.272727	0.3871	-0.0764	-3.9561
5	t4	0.363636	0.3381	0.0980	-3.6935
6	t5	0.454545	0.1747	0.4300	-3.2961
7	t6	0.500000	0.0197	0.6928	-3.0160
8	t7	0.545455	-0.1102	0.9365	-2.7518
9	t8	0.636364	-0.3543	1.4418	-2.2749
10	t9	0.727273	-0.6169	2.0165	-1.7714
11	t10	0.818182	-1.0307	2.8540	-1.0614
12	t11	0.909091	-1.4056	3.8941	-0.3141
13	t12	1.000000	-1.9936	9.1022	1.2334

Observation Quantile Levels

Quantile process regression can estimate observation quantile levels for any valid observations. For more information, see the section "Observation Quantile Level" on page 8678. You can convert an observation quantile level to the percentage of the response value conditional on its covariate values. In the following statements, the OUTPUT statement outputs observation quantile levels:

```
proc quantselect data=analysisData;
   model y = &model / quantile=process selection=none;
   output out=outQuantLev p ql;
run;

proc print data=outQuantLev(obs=10);
run;
```

Output 104.5.3 shows the first 10 observations of the OUT=outQuantLev data set, in which the variable ql_y contains the observation quantile levels.

Output 104.5.3 Observation Quantile Levels

Obs	i	x1	x2	y	p_y	ql_y
1	1	0	1.00000	-2.34428	-2.98693	0.74850
2	2	.000166667	0.99967	-2.74004	-2.98580	0.59681
3	3	.000333333	0.99933	-2.03681	-2.98467	0.83420
4	4	.000500000	0.99900	-3.65071	-2.98354	0.24282
5	5	.000666667	0.99867	-3.36232	-2.98240	0.35729
6	6	.000833333	0.99833	-2.27148	-2.98127	0.77046
7	7	.001000000	0.99800	-3.30503	-2.98014	0.38186
8	8	.001166667	0.99767	-3.39110	-2.97901	0.34331
9	9	.001333333	0.99734	-2.63652	-2.97788	0.63473
10	10	.001500000	0.99700	-3.62622	-2.97675	0.24533

Observationwise Distribution Estimation

Quantile process regression can estimate the entire distribution of a response variable conditional on its covariates. The following statements use the IML procedure to create macro variables for observation indices, observation quantile levels, and observation mean predictions and create a data set, distData, that contains all quantile levels and quantile predictions for the specified observations:

```
proc iml;
   Obs  = {1 3001 6001};
   nObs = ncol(Obs);
   call symputx("nObs",nObs);

   use analysisData;
   read all var {&model} into x;
   read all var {"y"} into y;
   close analysisData;

   use outQuantLev;
   read all var {"ql_y"} into ql;
   read all var {"p_y"} into  qm;
   close outQuantLev;

   use quantProcessEst;
   read all var {"Intercept"} into beta0;
   read all var {&model} into beta;
   read all var {"QuantileLevel"} into qLev;
   close quantProcessEst;

   nTau = nrow(qLev);
   qPrcs = j((nTau*nObs),3);
   obsInfo = j(nObs,6);
   obsIndex ="_Obs1":"_Obs&nObs";
   levNames="_qLev1":"_qLev&nObs";
   qtNames ="_qt1":"_qt&nObs";
   qmNames ="_qMean1":"_qMean&nObs";

   do j=1 to nObs;
      iObs = Obs[j];
      call symputx(obsIndex[j],iObs);
      call symputx(qtNames[j], (y[iObs]));
      call symputx(levNames[j],(ql[iObs]));
      call symputx(qmNames[j], (qm[iObs]));
      obsInfo[j,1]=iObs;
      obsInfo[j,2]=y[iObs];
      obsInfo[j,3]=x[iObs,1];
      obsInfo[j,4]=x[iObs,2];
      obsInfo[j,5]=ql[iObs];
      obsInfo[j,6]=qm[iObs];
      Quantiles = beta0 + beta*t(x[iObs,]);
      call sort(Quantiles,1);
      qPrcs[((j-1)*nTau+1):(j*nTau),1]=iObs;
      qPrcs[((j-1)*nTau+1):(j*nTau),2]=qLev;
      qPrcs[((j-1)*nTau+1):(j*nTau),3]=Quantiles;
   end;
```

```
   obsInfoColName = {"Index" "Response Value" "x1" "x2"
                     "Quantile Level" "Mean Prediction"};
   obsInfoLabel = {"Information for Specified Observations"};
   print obsInfo[colname=obsInfoColName label=obsInfoLabel];

   create distData from qPrcs[colname={"iObs" "qLev" "Quantiles"}];
   append from qPrcs;
   close distData;
quit;
```

Output 104.5.4 shows the observation information table for the specified observations.

Output 104.5.4 Information for Observations 1, 3001, and 6001

Information for Specified Observations						
	Index	Response Value	x1	x2	Quantile Level	Mean Prediction
ROW1	1	-2.34428	0	1	0.748503	-2.986932
ROW2	3001	-0.557309	0.5	0.25	0.3433134	-0.280479
ROW3	6001	0.9299095	1	0	0.5788423	1.044977

The following statements plot the conditional cumulative distribution functions (CDFs) for the specified observations:

```
data distData;
   set distData;
   label iObs  = "Observation Index"
         qLev  = "Cumulative Probability"
         Quantiles = "Quantile";
run;

%macro plotCDF;
   proc sgplot data=distData;
      series y=qLev x=Quantiles/group=iObs;
      %do j=1 %to &nObs;
      refline &&_qLev&j/label=("Obs &&_Obs&j")
                        axis=y labelloc=inside;
      refline &&_qt&j/ label=("Obs &&_Obs&j")
                        axis=x;
      %end;
   run;
%mend;

%plotCDF;
```

Figure 104.5.5 displays the conditional CDFs for the three specified observations. Each observation has a vertical reference line for its observed response value and a horizontal reference line for its quantile level.

Output 104.5.5 Conditional Cumulative Distribution Function for Observation 1

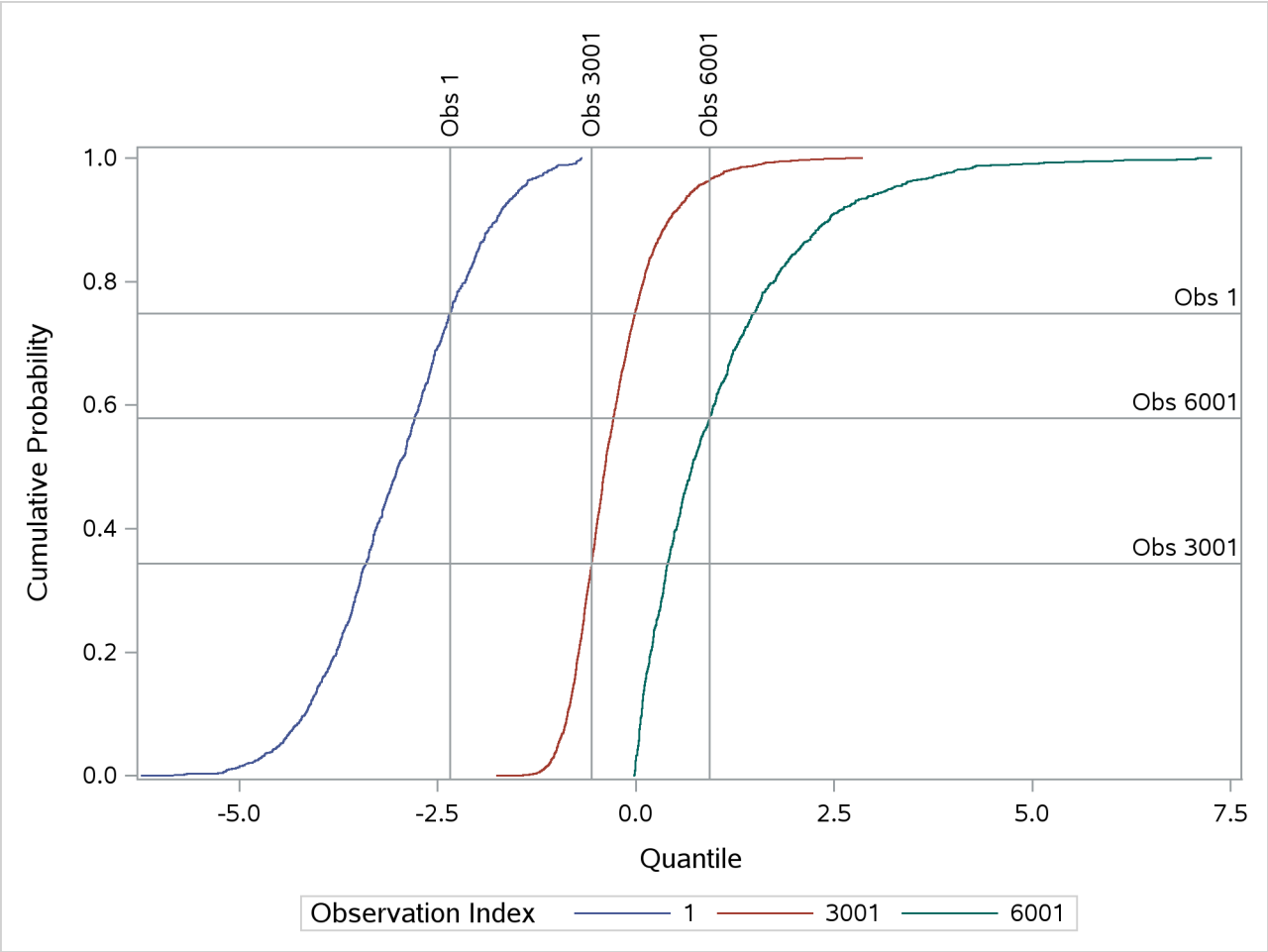

You can also estimate the conditional probability density function (PDF) for the specified observations by using the KDE procedure. The following statements compute the probability estimates for each predicted quantile of the specified observations and plot their PDFs:

```
proc iml;
   use distData;
   read all var {"iObs"} into iObs;
   read all var {"qLev"} into qLev;
   read all var {"Quantiles"} into Y;
   close distData;

   nObs = &nObs;
   nTau = nrow(qLev)/nObs;
   pProb = j(nTau,3+nObs);

   pProb[,1]            = t(1:nTau);
   pProb[,2]            = qLev[1:nTau];
   pProb[1,3]           = (qLev[1]+qLev[2])/2;
   pProb[nTau,3]        = 1-(qLev[nTau-1]+qLev[nTau])/2;
   pProb[2:(nTau-1),3]  = (qLev[3:nTau]-qLev[1:(nTau-2)])/2;
```

```
   do j=1 to nObs;
      jump = (j-1)*nTau;
      pProb[,(3+j)] = Y[(jump+1):(jump+nTau)];
   end;

   create probData from pProb[colname={"obs" "quantLev" "pProb"
                                       "Obs1" "Obs3001" "Obs6001"}];
   append from pProb;
   close probData;
quit;


proc kde data=probData;
   weight pProb;
   univar Obs1 Obs3001 Obs6001/ plots=densityoverlay;
run;
```

Figure 104.15 displays the density estimation plots for the specified observations. You can see that the PDF for Observation 1 is a normal PDF, the PDF for Observation 3001 is slightly right-skewed, and the PDF for Observation 6001 is an exponential PDF.

Figure 104.15 Density Estimates

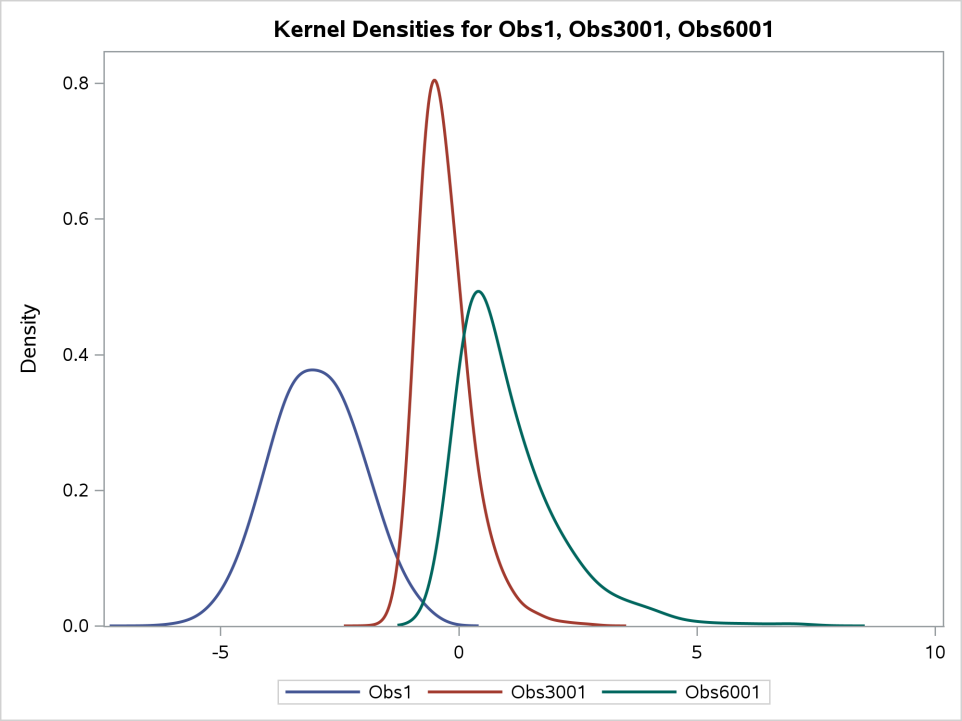

References

Akaike, H. (1981). "Likelihood of a Model and Information Criteria." *Journal of Econometrics* 16:3–14.

Barro, R., and Lee, J.-W. (1994). "Data Set for a Panel of 138 Countries." Discussion paper, National Bureau of Econometric Research. http://admin.nber.org/pub/barro.lee/readme.txt.

Belloni, A., and Chernozhukov, V. (2011). "L1-Penalized Quantile Regression in High-Dimensional Sparse Models." *Annals of Statistics* 39:82–130.

Chernozhukov, V., and Hansen, C. (2008). "Instrumental Variable Quantile Regression: A Robust Inference Approach." *Journal of Econometrics* 142:379–398.

Darlington, R. B. (1968). "Multiple Regression in Psychological Research and Practice." *Psychological Bulletin* 69:161–182.

Gu, C., Bates, D. M., Chen, Z., and Wahba, G. (1990). "The Computation of GCV Function through Householder Tridiagonalization with Application to the Fitting of Interaction Splines Models." *SIAM Journal on Matrix Analysis and Applications* 10:457–480.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Judge, G. G., Griffiths, W. E., Hill, R. C., Lütkepohl, H., and Lee, T.-C. (1985). *The Theory and Practice of Econometrics*. 2nd ed. New York: John Wiley & Sons.

Koenker, R., and Bassett, G. W. (1978). "Regression Quantiles." *Econometrica* 46:33–50.

Koenker, R., and Machado, A. F. (1999). "Goodness of Fit and Related Inference Processes for Quantile Regression." *Journal of the American Statistical Association* 94:1296–1310.

McDonald, G. C., and Schwing, R. C. (1973). "Instabilities of Regression Estimates Relating Air Pollution to Mortality." *Technometrics* 15:463–481.

Reichler, J. L., ed. (1987). *The 1987 Baseball Encyclopedia Update*. New York: Macmillan.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B* 58:267–288.

Time Inc. (1987). "What They Make." *Sports Illustrated* (April 20): 54–81.

Wu, Y., and Liu, Y. (2009). "Variable Selection in Quantile Regression." *Statistica Sinica* 19:801–817.

Zou, H. (2006). "The Adaptive Lasso and Its Oracle Properties." *Journal of the American Statistical Association* 101:1418–1429.

Chapter 105
The REG Procedure

Contents

Overview: REG Procedure . . . **8722**
Getting Started: REG Procedure . . . **8724**
- Simple Linear Regression . . . 8724
- Polynomial Regression . . . 8728
- Using PROC REG Interactively . . . 8739

Syntax: REG Procedure . . . **8740**
- PROC REG Statement . . . 8742
- ADD Statement . . . 8756
- BY Statement . . . 8756
- CODE Statement . . . 8757
- DELETE Statement . . . 8758
- FREQ Statement . . . 8758
- ID Statement . . . 8758
- MODEL Statement . . . 8759
- MTEST Statement . . . 8770
- OUTPUT Statement . . . 8772
- PRINT Statement . . . 8774
- REFIT Statement . . . 8774
- RESTRICT Statement . . . 8775
- REWEIGHT Statement . . . 8777
- STORE Statement . . . 8780
- TEST Statement . . . 8780
- VAR Statement . . . 8781
- WEIGHT Statement . . . 8782

Details: REG Procedure . . . **8782**
- Missing Values . . . 8782
- Input Data Sets . . . 8782
- Output Data Sets . . . 8785
- Interactive Analysis . . . 8792
- Model-Selection Methods . . . 8795
- Criteria Used in Model-Selection Methods . . . 8798
- Limitations in Model-Selection Methods . . . 8799
- Parameter Estimates and Associated Statistics . . . 8799
- Predicted and Residual Values . . . 8801
- Models of Less Than Full Rank . . . 8805
- Collinearity Diagnostics . . . 8806

Model Fit and Diagnostic Statistics 8808
Influence Statistics 8810
Reweighting Observations in an Analysis 8818
Testing for Heteroscedasticity 8824
Testing for Lack of Fit 8825
Multivariate Tests 8826
Autocorrelation in Time Series Data 8829
Computations for Ridge Regression and IPC Analysis 8830
Construction of Q-Q and P-P Plots 8831
Computational Methods 8831
Computer Resources in Regression Analysis 8832
Displayed Output 8832
Plot Options Superseded by ODS Graphics 8834
ODS Table Names 8850
ODS Graphics 8852
Examples: REG Procedure **8858**
Example 105.1: Modeling Salaries of Major League Baseball Players 8858
Example 105.2: Aerobic Fitness Prediction 8875
Example 105.3: Predicting Weight by Height and Age 8891
Example 105.4: Regression with Quantitative and Qualitative Variables 8896
Example 105.5: Ridge Regression for Acetylene Data 8901
Example 105.6: Chemical Reaction Response 8905
References **8907**

Overview: REG Procedure

The REG procedure is one of many regression procedures in the SAS System. It is a general-purpose procedure for regression, while other SAS regression procedures provide more specialized applications.

Other SAS/STAT procedures that perform at least one type of regression analysis are the CATMOD, GENMOD, GLM, LOGISTIC, MIXED, NLIN, ORTHOREG, PROBIT, RSREG, and TRANSREG procedures. SAS/ETS procedures are specialized for applications in time series or simultaneous systems. These other SAS/STAT regression procedures are summarized in Chapter 4, "Introduction to Regression Procedures," which also contains an overview of regression techniques and defines many of the statistics computed by PROC REG and other regression procedures.

PROC REG provides the following capabilities:

- multiple MODEL statements
- nine model-selection methods
- interactive changes both in the model and the data used to fit the model
- linear equality restrictions on parameters

- tests of linear hypotheses and multivariate hypotheses
- collinearity diagnostics
- predicted values, residuals, studentized residuals, confidence limits, and influence statistics
- correlation or crossproduct input
- requested statistics available for output through output data sets
- ODS Graphics. For more information, see the section "ODS Graphics" on page 8852.

Nine model-selection methods are available in PROC REG. In the simplest method, PROC REG fits the complete model that you specify. The other eight methods involve various ways of including or excluding variables from the model. You specify these methods with the SELECTION= option in the MODEL statement.

The methods are identified in the following list and are explained in detail in the section "Model-Selection Methods" on page 8795.

NONE	no model selection. This is the default. The complete model specified in the MODEL statement is fit to the data.
FORWARD	forward selection. This method starts with no variables in the model and adds variables.
BACKWARD	backward elimination. This method starts with all variables in the model and deletes variables.
STEPWISE	stepwise regression. This is similar to the FORWARD method except that variables already in the model do not necessarily stay there.
MAXR	forward selection to fit the best one-variable model, the best two-variable model, and so on. Variables are switched so that R square is maximized.
MINR	similar to the MAXR method, except that variables are switched so that the increase in R square from adding a variable to the model is minimized.
RSQUARE	finds a specified number of models with the highest R square in a range of model sizes.
ADJRSQ	finds a specified number of models with the highest adjusted R square in a range of model sizes.
CP	finds a specified number of models with the lowest C_p in a range of model sizes.

Getting Started: REG Procedure

Simple Linear Regression

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/reggs.sas.)

Suppose that a response variable Y can be predicted by a linear function of a regressor variable X. You can estimate β_0, the intercept, and β_1, the slope, in

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

for the observations $i = 1, 2, \ldots, n$. Fitting this model with the REG procedure requires only the following MODEL statement, where y is the outcome variable and x is the regressor variable.

```
proc reg;
   model y=x;
run;
```

For example, you might use regression analysis to find out how well you can predict a child's weight if you know that child's height. The Class data set used in this example is available in the Sashelp library.

The equation of interest is

$$\text{Weight} = \beta_0 + \beta_1 \text{Height} + \epsilon$$

The variable Weight is the response or dependent variable in this equation, and β_0 and β_1 are the unknown parameters to be estimated. The variable Height is the regressor or independent variable, and ϵ is the unknown error. The following commands invoke the REG procedure and fit this model to the data.

```
ods graphics on;

proc reg data=sashelp.class;
   model Weight = Height;
run;
```

Figure 105.1 includes some information concerning model fit.

The F statistic for the overall model is highly significant ($F = 57.076$, $p < 0.0001$), indicating that the model explains a significant portion of the variation in the data.

The degrees of freedom can be used in checking accuracy of the data and model. The model degrees of freedom are one less than the number of parameters to be estimated. This model estimates two parameters, β_0 and β_1; thus, the degrees of freedom should be 2 – 1 = 1. The corrected total degrees of freedom are always one less than the total number of observations in the data set, in this case 19 – 1 = 18.

Several simple statistics follow the ANOVA table. The Root MSE is an estimate of the standard deviation of the error term. The coefficient of variation, or Coeff Var, is a unitless expression of the variation in the data.

The R-square and Adj R-square are two statistics used in assessing the fit of the model; values close to 1 indicate a better fit. The R-square of 0.77 indicates that Height accounts for 77% of the variation in Weight.

Figure 105.1 ANOVA Table

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	7193.24912	7193.24912	57.08	<.0001
Error	17	2142.48772	126.02869		
Corrected Total	18	9335.73684			

Root MSE	11.22625	**R-Square**	0.7705
Dependent Mean	100.02632	**Adj R-Sq**	0.7570
Coeff Var	11.22330		

The "Parameter Estimates" table in Figure 105.2 contains the estimates of β_0 and β_1. The table also contains the t statistics and the corresponding p-values for testing whether each parameter is significantly different from zero. The p-values ($t = -4.43$, $p = 0.0004$ and $t = 7.55$, $p < 0.0001$) indicate that the intercept and Height parameter estimates, respectively, are highly significant.

From the parameter estimates, the fitted model is

$$\text{Weight} = -143.0 + 3.9 \times \text{Height}$$

Figure 105.2 Parameter Estimates

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

If ODS Graphics is enabled, then PROC REG produces a variety of plots. Figure 105.3 shows a plot of the residuals versus the regressor and Figure 105.4 shows a panel of diagnostic plots.

Figure 105.3 Residuals vs. Regressor

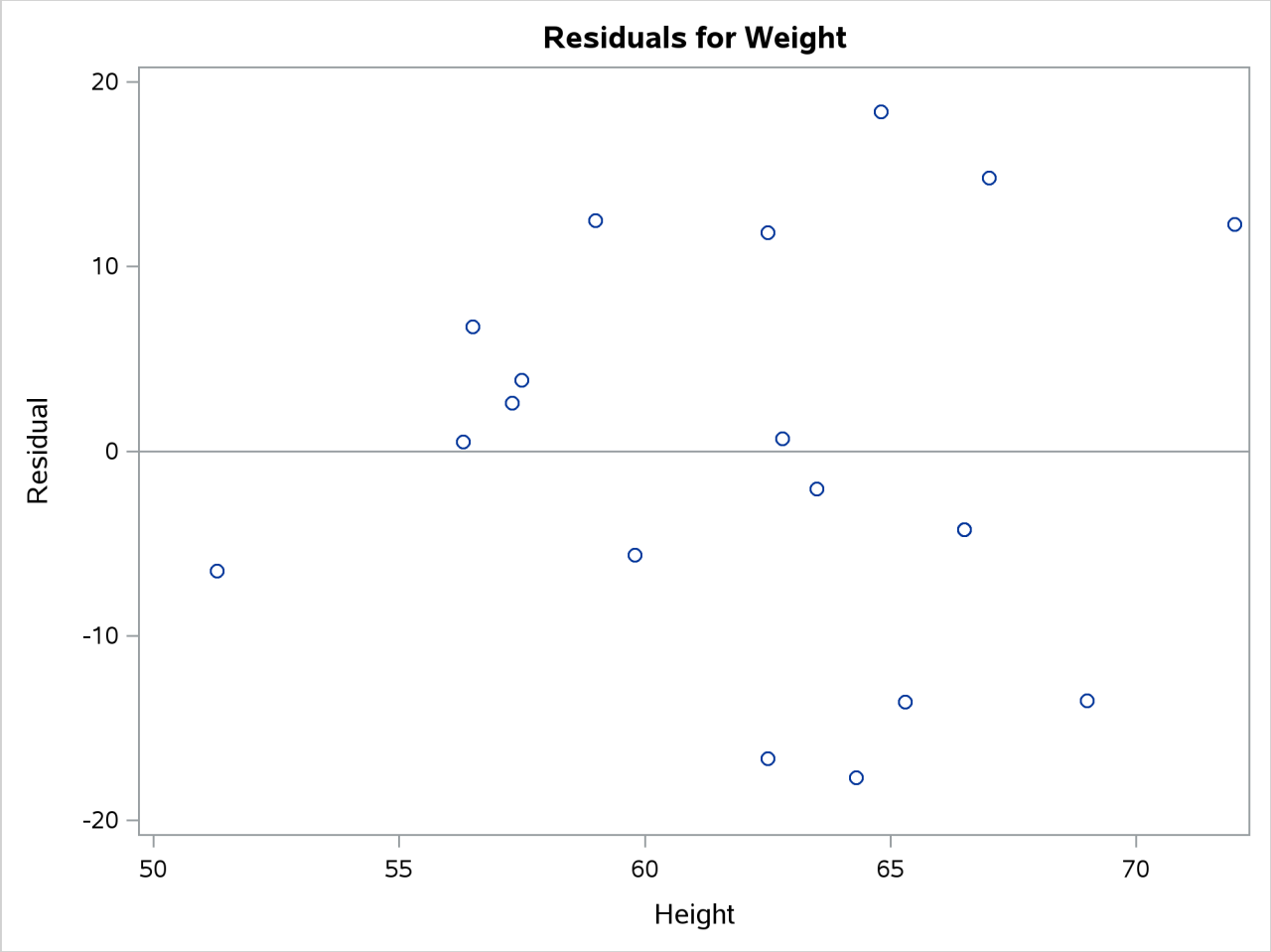

Figure 105.4 Fit Diagnostics

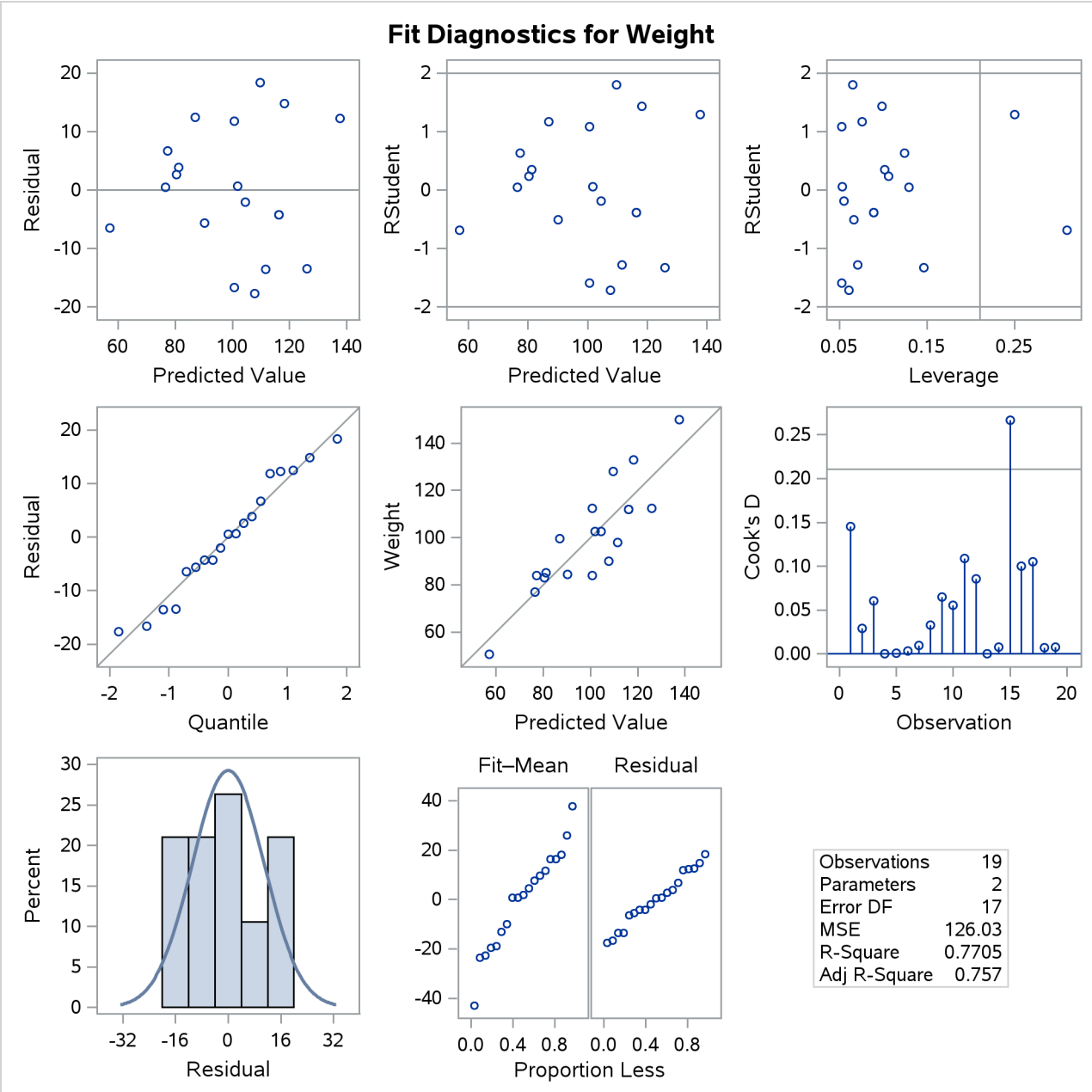

A trend in the residuals would indicate nonconstant variance in the data. The plot of residuals by predicted values in the upper-left corner of the diagnostics panel in Figure 105.4 might indicate a slight trend in the residuals; they appear to increase slightly as the predicted values increase. A fan-shaped trend might indicate the need for a variance-stabilizing transformation. A curved trend (such as a semicircle) might indicate the need for a quadratic term in the model. Since these residuals have no apparent trend, the analysis is considered to be acceptable.

Polynomial Regression

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/reggs.sas.)

Consider a response variable Y that can be predicted by a polynomial function of a regressor variable X. You can estimate β_0, the intercept; β_1, the slope due to X; and β_2, the slope due to X^2, in

$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \epsilon_i$$

for the observations $i = 1, 2, \ldots, n$.

Consider the following example on population growth trends. The population of the United States from 1790 to 2000 is fit to linear and quadratic functions of time. Note that the quadratic term, YearSq, is created in the DATA step; this is done since polynomial effects such as Year*Year cannot be specified in the MODEL statement in PROC REG. The data are as follows:

```
data USPopulation;
   input Population @@;
   retain Year 1780;
   Year       = Year+10;
   YearSq     = Year*Year;
   Population = Population/1000;
   datalines;
3929 5308 7239 9638 12866 17069 23191 31443 39818 50155
62947 75994 91972 105710 122775 131669 151325 179323 203211
226542 248710 281422
;

proc reg data=USPopulation plots=ResidualByPredicted;
   var YearSq;
   model Population=Year / r clm cli;
run;
```

The DATA option ensures that the procedure uses the intended data set. Any variable that you might add to the model but that is not included in the first MODEL statement must appear in the VAR statement.

The “Analysis of Variance” and “Parameter Estimates” tables are displayed in Figure 105.5.

Figure 105.5 ANOVA Table and Parameter Estimates

The REG Procedure
Model: MODEL1
Dependent Variable: Population

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	146869	146869	228.92	<.0001
Error	20	12832	641.58160		
Corrected Total	21	159700			

Root MSE	25.32946	**R-Square**	0.9197
Dependent Mean	94.64800	**Adj R-Sq**	0.9156
Coeff Var	26.76175		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-2345.85498	161.39279	-14.54	<.0001
Year	1	1.28786	0.08512	15.13	<.0001

The Model F statistic is significant ($F = 228.92$, $p < 0.0001$), indicating that the model accounts for a significant portion of variation in the data. The R-square indicates that the model accounts for 92% of the variation in population growth. The fitted equation for this model is

$$\text{Population} = -2345.85 + 1.29 \times \text{Year}$$

In the MODEL statement, three options are specified: R requests a residual analysis to be performed, CLI requests 95% confidence limits for an individual value, and CLM requests these limits for the expected value of the dependent variable. You can request specific $100(1-\alpha)\%$ limits with the ALPHA= option in the PROC REG or MODEL statement.

Figure 105.6 shows the "Output Statistics" table. The residual, its standard error, the studentized residual, and Cook's D are displayed for each observation. The studentized residual is the residual divided by its standard error. Cook's D is a measure of the change in the predicted values upon deletion of that observation from the data set; hence, it measures the influence of the observation on the estimated regression coefficients. The studentized residuals and Cook's D statistics are displayed in Figure 105.7.

Studentized residuals follow a beta distribution and can be used to identify outlying or extreme observations:

- Having a few large absolute studentized residuals is no cause for alarm.
- Many observations having absolute studentized residuals greater than two might indicate an inadequate model.
- Absolute studentized residuals greater than three are rarer; they should be regarded as extreme, and the data should be reviewed carefully.

Figure 105.6 Output Statistics

The REG Procedure
Model: MODEL1
Dependent Variable: Population

Output Statistics											
Obs	Dependent Variable	Predicted Value	Std Error Mean Predict	95% CL Mean		95% CL Predict		Residual	Std Error Residual	Student Residual	Cook's D
1	3.93	-40.5778	10.4424	-62.3602	-18.7953	-97.7280	16.5725	44.5068	23.077	1.929	0.381
2	5.31	-27.6991	9.7238	-47.9826	-7.4156	-84.2950	28.8968	33.0071	23.389	1.411	0.172
3	7.24	-14.8205	9.0283	-33.6533	4.0123	-70.9128	41.2719	22.0595	23.666	0.932	0.063
4	9.64	-1.9418	8.3617	-19.3841	15.5004	-57.5827	53.6991	11.5798	23.909	0.484	0.014
5	12.87	10.9368	7.7314	-5.1906	27.0643	-44.3060	66.1797	1.9292	24.121	0.080	0.000
6	17.07	23.8155	7.1470	8.9070	38.7239	-31.0839	78.7148	-6.7465	24.300	-0.278	0.003
7	23.19	36.6941	6.6208	22.8834	50.5048	-17.9174	91.3056	-13.5031	24.449	-0.552	0.011
8	31.44	49.5727	6.1675	36.7075	62.4380	-4.8073	103.9528	-18.1297	24.567	-0.738	0.017
9	39.82	62.4514	5.8044	50.3436	74.5592	8.2455	116.6573	-22.6334	24.655	-0.918	0.023
10	50.16	75.3300	5.5491	63.7547	86.9053	21.2406	129.4195	-25.1750	24.714	-1.019	0.026
11	62.95	88.2087	5.4170	76.9090	99.5084	34.1776	142.2398	-25.2617	24.743	-1.021	0.025
12	75.99	101.0873	5.4170	89.7876	112.3870	47.0562	155.1184	-25.0933	24.743	-1.014	0.025
13	91.97	113.9660	5.5491	102.3907	125.5413	59.8765	168.0554	-21.9940	24.714	-0.890	0.020
14	105.71	126.8446	5.8044	114.7368	138.9524	72.6387	181.0505	-21.1346	24.655	-0.857	0.020
15	122.78	139.7233	6.1675	126.8580	152.5885	85.3432	194.1033	-16.9483	24.567	-0.690	0.015
16	131.67	152.6019	6.6208	138.7912	166.4126	97.9904	207.2134	-20.9329	24.449	-0.856	0.027
17	151.33	165.4805	7.1470	150.5721	180.3890	110.5812	220.3799	-14.1555	24.300	-0.583	0.015
18	179.32	178.3592	7.7314	162.2317	194.4866	123.1163	233.6020	0.9638	24.121	0.040	0.000
19	203.21	191.2378	8.3617	173.7956	208.6801	135.5969	246.8787	11.9732	23.909	0.501	0.015
20	226.54	204.1165	9.0283	185.2837	222.9493	148.0241	260.2088	22.4255	23.666	0.948	0.065
21	248.71	216.9951	9.7238	196.7116	237.2786	160.3992	273.5910	31.7149	23.389	1.356	0.159
22	281.42	229.8738	10.4424	208.0913	251.6562	172.7235	287.0240	51.5482	23.077	2.234	0.511

Figure 105.7 Studentized Residuals

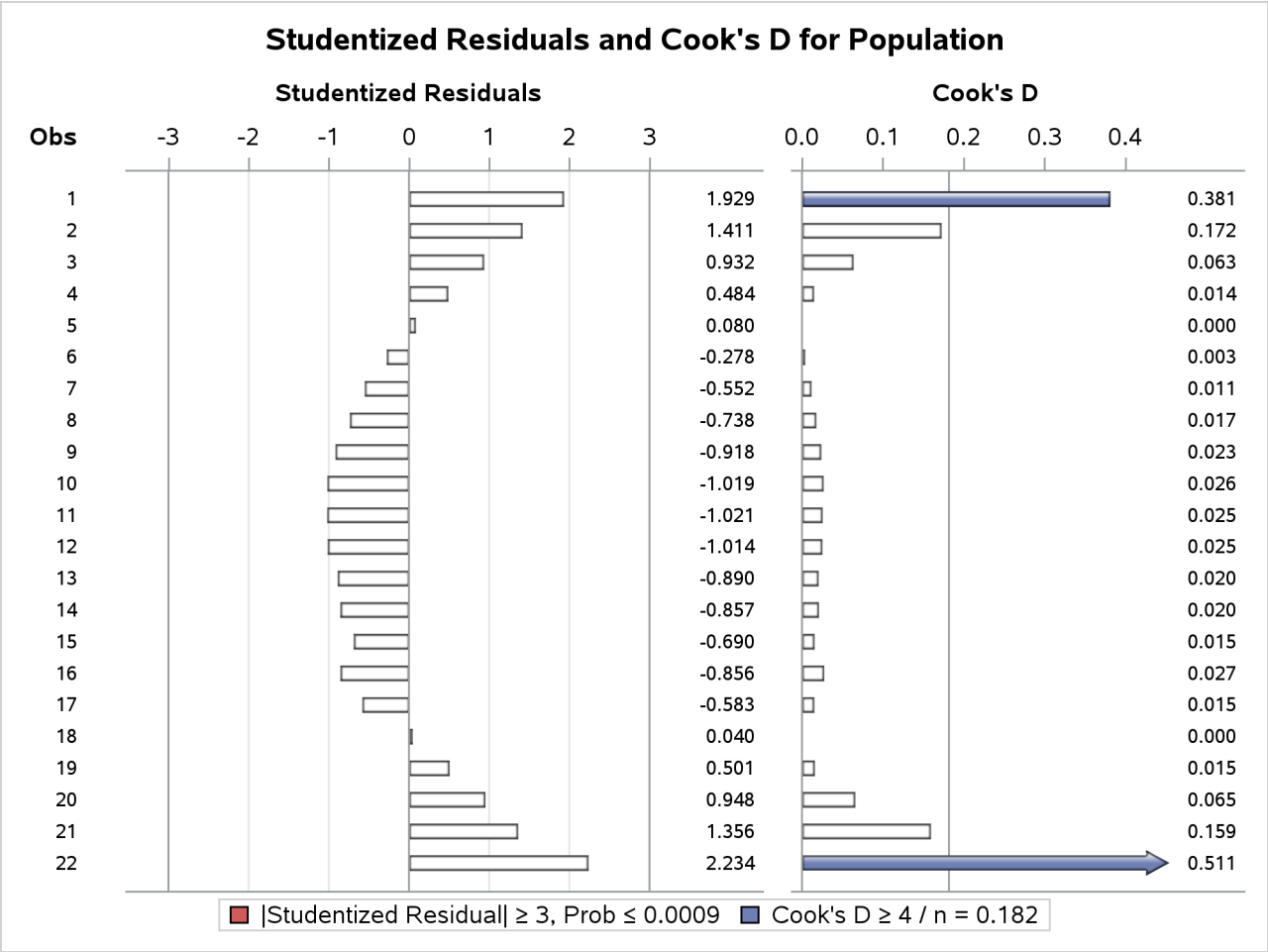

Figure 105.8 shows the residual statistics table. A fairly close agreement between the PRESS statistic (see Table 105.7) and the sum of squared residuals indicates that the MSE is a reasonable measure of the predictive accuracy of the fitted model (Neter, Wasserman, and Kutner 1990).

Figure 105.8 Residual Statistics

Sum of Residuals	0
Sum of Squared Residuals	12832
Predicted Residual SS (PRESS)	16662

Graphical representations are very helpful in interpreting the information in the "Output Statistics" table. When ODS Graphics is enabled, the REG procedure produces a default set of diagnostic plots that are appropriate for the requested analysis.

Figure 105.9 displays a panel of diagnostics plots. These diagnostics indicate an inadequate model:

- The plots of residual and studentized residual versus predicted value show a clear quadratic pattern.
- The plot of studentized residual versus leverage seems to indicate that there are two outlying data points. However, the plot of Cook's *D* distance versus observation number reveals that these two points

are just the data points for the endpoint years 1790 and 2000. These points show up as apparent outliers because the departure of the linear model from the underlying quadratic behavior in the data shows up most strongly at these endpoints.

- The normal quantile plot of the residuals and the residual histogram are not consistent with the assumption of Gaussian errors. This occurs as the residuals themselves still contain the quadratic behavior that is not captured by the linear model.
- The plot of the dependent variable versus the predicted value exhibits a quadratic form around the 45-degree line that represents a perfect fit.
- The "Residual-Fit" (or RF) plot consisting of side-by-side quantile plots of the centered fit and the residuals shows that the spread in the residuals is no greater than the spread in the centered fit. For inappropriate models, the spread of the residuals in such a plot is often greater than the spread of the centered fit. In this case, the RF plot shows that the linear model does indeed capture the increasing trend in the data, and hence accounts for much of the variation in the response.

Figure 105.9 Diagnostics Panel

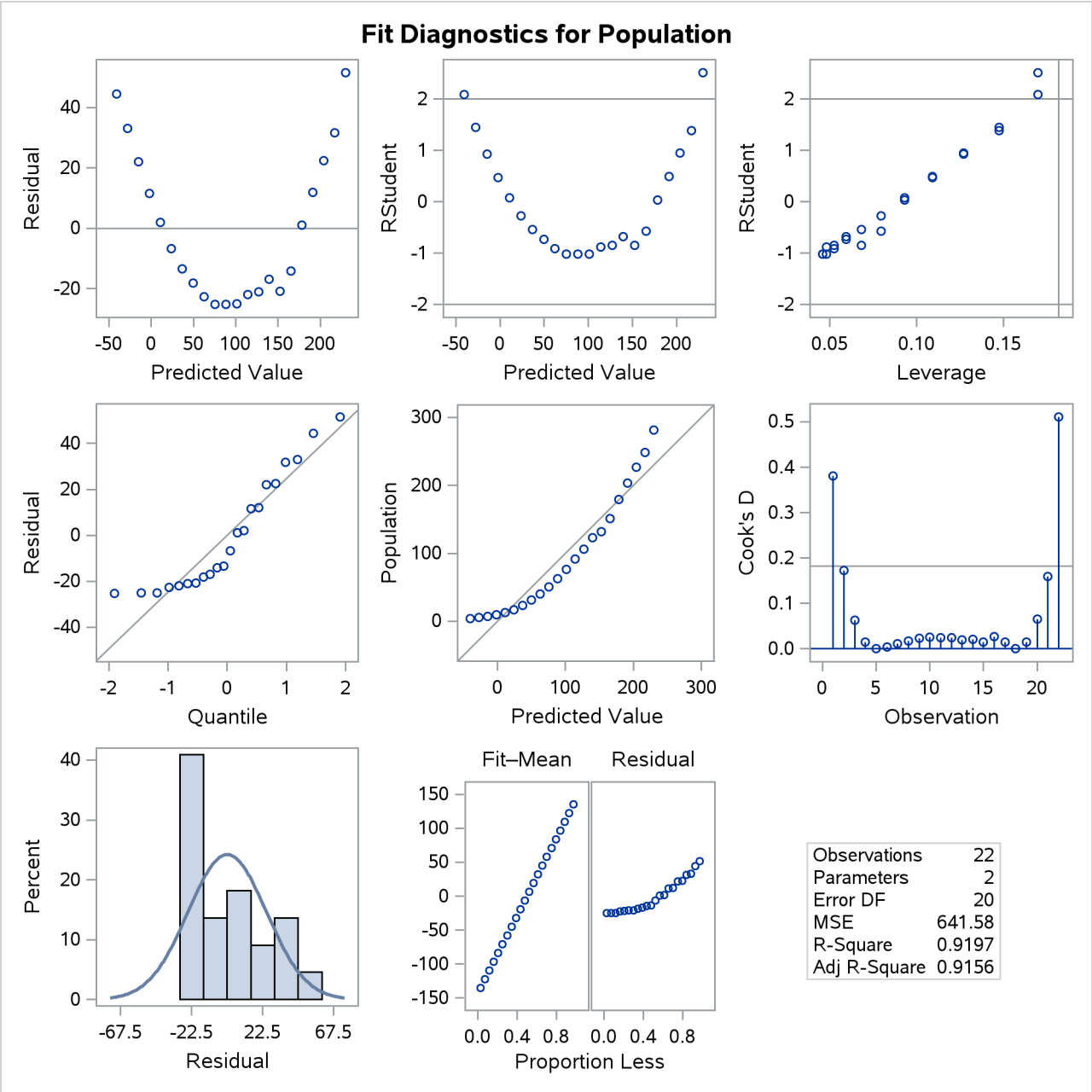

Figure 105.10 shows a plot of residuals versus Year. Again you can see the quadratic pattern that strongly indicates that a quadratic term should be added to the model.

Figure 105.10 Residual Plot

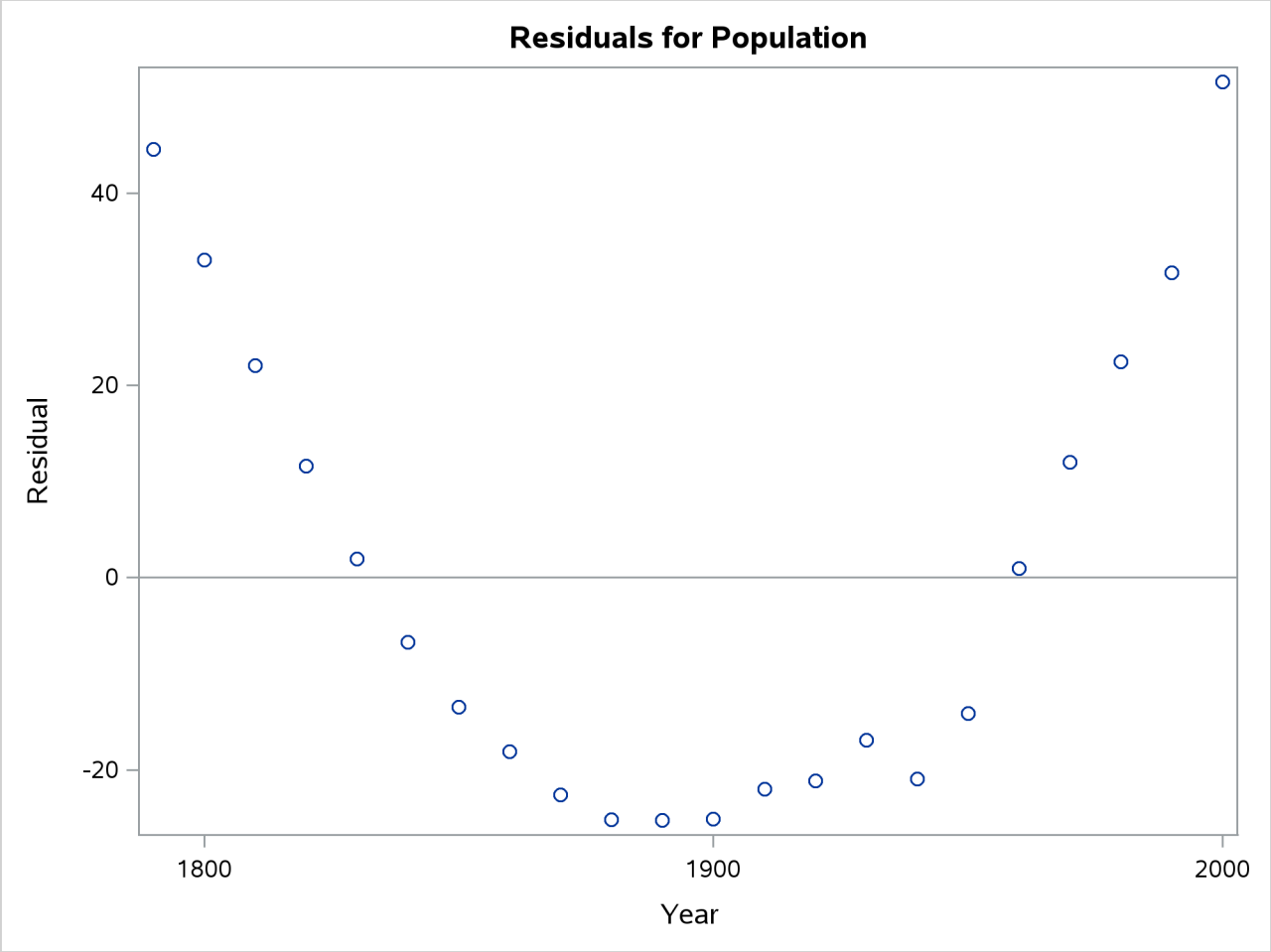

Figure 105.11 shows the FitPlot consisting of a scatter plot of the data overlaid with the regression line, and 95% confidence and prediction limits. Note that this plot also indicates that the model fails to capture the quadratic nature of the data. This plot is produced for models containing a single regressor. You can use the ALPHA= option in the model statement to change the significance level of the confidence band and prediction limits.

Figure 105.11 Fit Plot

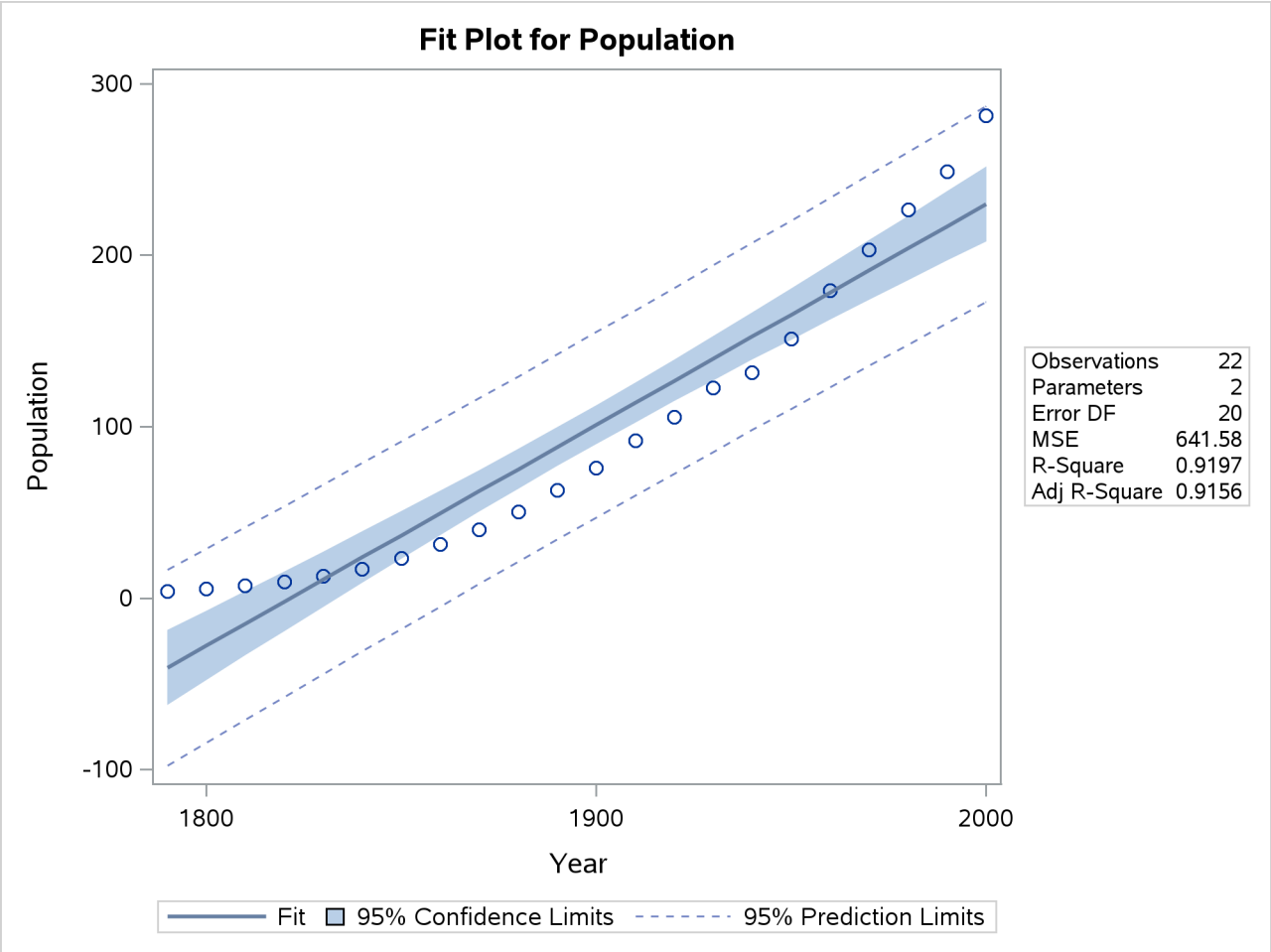

These default plots provide strong evidence that the Yearsq needs to be added to the model. You can use the interactive feature of PROC REG to do this by specifying the following statements:

```
    add YearSq;
    print;
run;
```

The ADD statement requests that YearSq be added to the model, and the PRINT command causes the model to be refit and displays the ANOVA and parameter estimates for the new model. The print statement also produces updated ODS graphical displays.

Figure 105.12 displays the ANOVA table and parameter estimates for the new model.

Figure 105.12 ANOVA Table and Parameter Estimates

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	159529	79765	8864.19	<.0001
Error	19	170.97193	8.99852		
Corrected Total	21	159700			

Root MSE	2.99975	R-Square	0.9989
Dependent Mean	94.64800	Adj R-Sq	0.9988
Coeff Var	3.16938		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	21631	639.50181	33.82	<.0001
Year	1	-24.04581	0.67547	-35.60	<.0001
YearSq	1	0.00668	0.00017820	37.51	<.0001

The overall F statistic is still significant ($F = 8864.19$, $p < 0.0001$). The R-square has increased from 0.9197 to 0.9989, indicating that the model now accounts for 99.9% of the variation in Population. All effects are significant with $p < 0.0001$ for each effect in the model.

The fitted equation is now

$$\text{Population} = 21631 - 24.046 \times \text{Year} + 0.0067 \times \text{Yearsq}$$

Figure 105.13 show the panel of diagnostics for this quadratic polynomial model. These diagnostics indicate that this model is considerably more successful than the corresponding linear model:

- The plots of residuals and studentized residuals versus predicted values exhibit no obvious patterns.
- The points on the plot of the dependent variable versus the predicted values lie along a 45-degree line, indicating that the model successfully predicts the behavior of the dependent variable.
- The plot of studentized residual versus leverage shows that the years 1790 and 2000 are leverage points with 2000 showing up as an outlier. This is confirmed by the plot of Cook's D distance versus observation number. This suggests that while the quadratic model fits the current data well, the model might not be quite so successful over a wider range of data. You might want to investigate whether the population trend over the last couple of decades is growing slightly faster than quadratically.

Figure 105.13 Diagnostics Panel

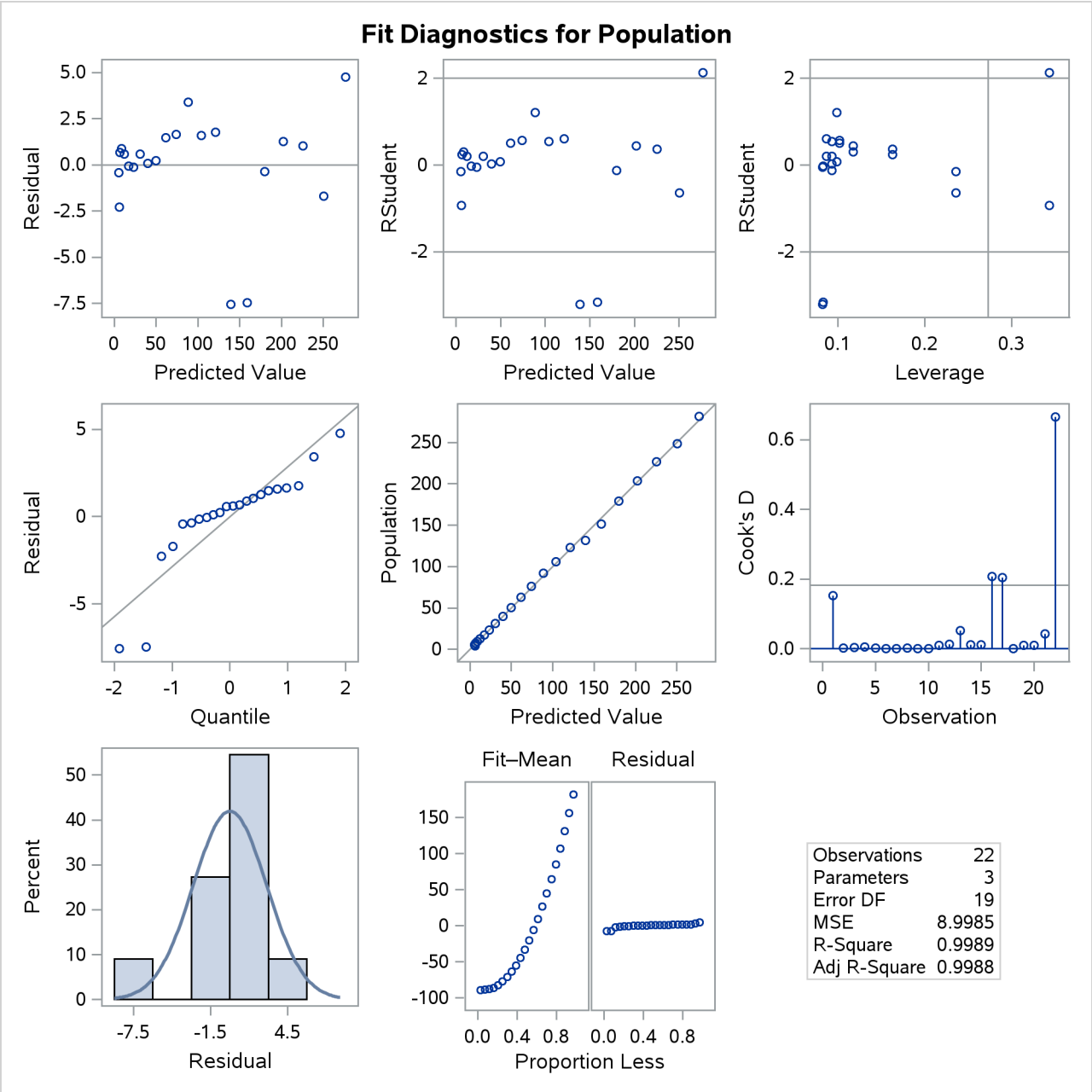

When a model contains more than one regressor, PROC REG does not produce a fit plot. However, when all the regressors in the model are functions of a single variable, it is appropriate to plot predictions and residuals as a function of that variable. You request such plots by using the PLOTS=PREDICTIONS option in the PROC REG statement, as the following code illustrates:

```
proc reg data=USPopulation plots=predictions(X=Year);
   model Population=Year Yearsq;
quit;
```

```
ods graphics off;
```

Figure 105.14 shows the data, predictions, and residuals by Year. These plots confirm that the quadratic polynomial model successfully model the growth in U.S. population between the years 1780 and 2000.

Figure 105.14 Predictions and Residuals by Year

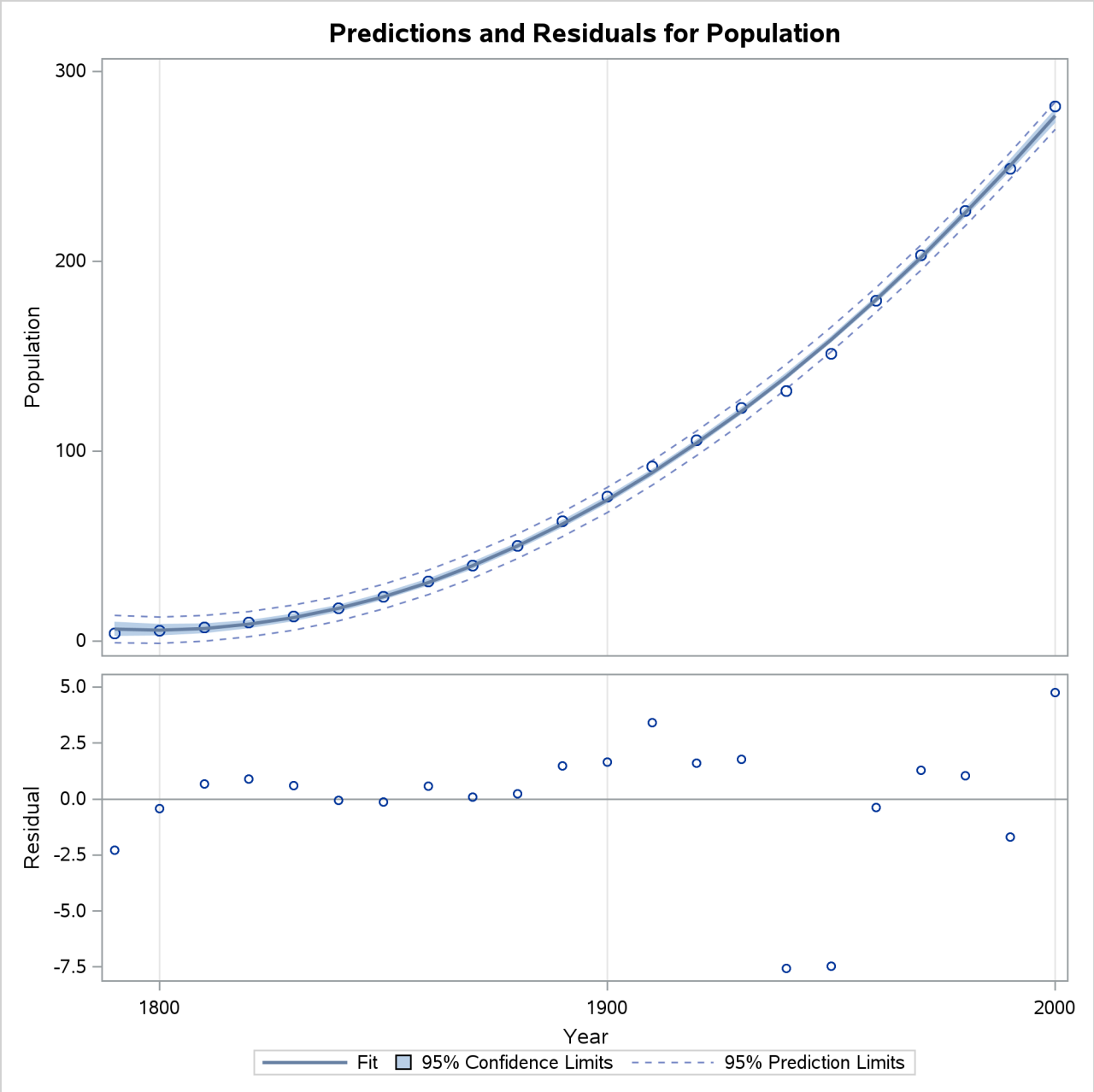

To complete an analysis of these data, you might want to examine influence statistics and, since the data are essentially time series data, examine the Durbin-Watson statistic.

Using PROC REG Interactively

The REG procedure can be used interactively. After you specify a model with a MODEL statement and run PROC REG with a RUN statement, a variety of statements can be executed without reinvoking PROC REG.

The section "Interactive Analysis" on page 8792 describes which statements can be used interactively. These interactive statements can be executed singly or in groups by following the single statement or group of statements with a RUN statement. Note that the MODEL statement can be repeated. This is an important difference from the GLM procedure, which supports only one MODEL statement.

If you use PROC REG interactively, you can end the REG procedure with a DATA step, another PROC step, an ENDSAS statement, or a QUIT statement. The syntax of the QUIT statement is

```
quit;
```

When you are using PROC REG interactively, additional RUN statements do not end PROC REG but tell the procedure to execute additional statements.

When a BY statement is used with PROC REG, interactive processing is not possible; that is, once the first RUN statement is encountered, processing proceeds for each BY group in the data set, and no further statements are accepted by the procedure.

When you use PROC REG interactively, you can fit a model, perform diagnostics, and then refit the model and perform diagnostics on the refitted model. Most of the interactive statements implicitly refit the model; for example, if you use the ADD statement to add a variable to the model, the regression equation is automatically recomputed. The two exceptions to this automatic recomputing are the PAINT and REWEIGHT statements. These two statements do not cause the model to be refitted. To refit the model, you can follow these statements either with a REFIT statement, which causes the model to be explicitly recomputed, or with another interactive statement that causes the model to be implicitly recomputed.

Syntax: REG Procedure

The following statements are available in the REG procedure:

```
PROC REG <options> ;
    <label:> MODEL dependents = <regressors> </options> ;
    BY variables ;
    FREQ variable ;
    ID variables ;
    VAR variables ;
    WEIGHT variable ;
    ADD variables ;
    CODE <options> ;
    DELETE variables ;
    <label:> MTEST <equation, ..., equation> </options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=names> <... keyword=names> ;
    PAINT <condition | ALLOBS> </options> | <STATUS | UNDO> ;
    PLOT <yvariable* xvariable> <=symbol> <... yvariable* xvariable> <=symbol> </options> ;
    PRINT <options> <ANOVA> <MODELDATA> ;
    REFIT ;
    RESTRICT equation, ..., equation ;
    REWEIGHT <condition | ALLOBS> </options> | <STATUS | UNDO> ;
    STORE <options> ;
    <label:> TEST equation, <, ..., equation> </option> ;
```

Although there are numerous statements and options available in PROC REG, many analyses use only a few of them. Often you can find the features you need by looking at an example or by scanning this section.

In the preceding list, brackets denote optional specifications, and vertical bars denote a choice of one of the specifications separated by the vertical bars. In all cases, *label* is optional.

The PROC REG statement is required. To fit a model to the data, you must specify the MODEL statement. If you want to use only the options available in the PROC REG statement, you do not need a MODEL statement, but you must use a VAR statement. (See the example in the section "OUTSSCP= Data Sets" on page 8791.) Several MODEL statements can be used. In addition, several MTEST, OUTPUT, PAINT, PLOT, PRINT, RESTRICT, and TEST statements can follow each MODEL statement.

The ADD, DELETE, and REWEIGHT statements are used interactively to change the regression model and the data used in fitting the model. The ADD, DELETE, MTEST, OUTPUT, PLOT, PRINT, RESTRICT, and TEST statements implicitly refit the model; changes made to the model are reflected in the results from these statements. The REFIT statement is used to refit the model explicitly and is most helpful when it follows PAINT and REWEIGHT statements, which do not refit the model.

The BY, FREQ, ID, VAR, and WEIGHT statements are optionally specified once for the entire PROC step, and they must appear before the first RUN statement.

When a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set is used as an input data set to PROC REG, statements and options that require the original data are not available. Specifically, the OUTPUT, PAINT, PLOT, and REWEIGHT statements and the MODEL and PRINT statement options P, R, CLM, CLI, DW, DWPROB, INFLUENCE, PARTIAL, and PARTIALDATA are disabled.

You can specify the following statements with the REG procedure in addition to the PROC REG statement:

ADD	adds independent variables to the regression model.
BY	specifies variables to define subgroups for the analysis.
CODE	requests that the procedure write SAS DATA step code to a file or catalog entry for computing predicted values according to the fitted model.
DELETE	deletes independent variables from the regression model.
FREQ	specifies a frequency variable.
ID	names a variable to identify observations in the tables.
MODEL	specifies the dependent and independent variables in the regression model, requests a model selection method, displays predicted values, and provides details on the estimates (according to which options are selected).
MTEST	performs multivariate tests across multiple dependent variables.
OUTPUT	creates an output data set and names the variables to contain predicted values, residuals, and other diagnostic statistics.
PAINT	paints points in scatter plots.
PLOT	generates scatter plots.
PRINT	displays information about the model and can reset options.
REFIT	refits the model.
RESTRICT	places linear equality restrictions on the parameter estimates.
REWEIGHT	excludes specific observations from analysis or changes the weights of observations used.
STORE	requests that the procedure save the estimated parameters of the fitted model.
TEST	performs an F test on linear functions of the parameters.
VAR	lists variables for which crossproducts are to be computed, variables that can be interactively added to the model, or variables to be used in scatter plots.
WEIGHT	declares a variable to weight observations.

The CODE and STORE statements are also used by many other procedures. A summary description of functionality and syntax for these statements is also shown after the PROC REG statement in alphabetical order, but you can find full documentation about them in the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

PROC REG Statement

PROC REG < *options* > **;**

The PROC REG statement invokes the REG procedure. The PROC REG statement is required. If you want to fit a model to the data, you must also use a MODEL statement. If you want to use only the PROC REG options, you do not need a MODEL statement, but you must use a VAR statement. If you do not use a MODEL statement, then the COVOUT and OUTEST= options are not available.

Table 105.1 summarizes the *options* available in the PROC REG statement. Note that any *option* specified in the PROC REG statement applies to all MODEL statements.

Table 105.1 PROC REG Statement Options

Option	Description
Data Set Options	
DATA=	Names a data set to use for the regression
OUTEST=	Outputs a data set that contains parameter estimates and other model fit summary statistics
OUTSSCP=	Outputs a data set that contains sums of squares and crossproducts
COVOUT	Outputs the covariance matrix for parameter estimates to the OUTEST= data set
EDF	Outputs the number of regressors, the error degrees of freedom, and the model R square to the OUTEST= data set
OUTSEB	Outputs standard errors of the parameter estimates to the OUTEST= data set
OUTSTB	Outputs standardized parameter estimates to the OUTEST= data set. Use only with the RIDGE= or PCOMIT= option.
OUTVIF	Outputs the variance inflation factors to the OUTEST= data set. Use only with the RIDGE= or PCOMIT= option.
PCOMIT=	Performs incomplete principal component analysis and outputs estimates to the OUTEST= data set
PRESS	Outputs the PRESS statistic to the OUTEST= data set
RIDGE=	Performs ridge regression analysis and outputs estimates to the OUTEST= data set
RSQUARE	Same effect as the EDF option
TABLEOUT	Outputs standard errors, confidence limits, and associated test statistics of the parameter estimates to the OUTEST= data set
ODS Graphics Options	
PLOTS=	Produces ODS graphical displays
Display Options	
CORR	Displays correlation matrix for variables listed in MODEL and VAR statements
SIMPLE	Displays simple statistics for each variable listed in MODEL and VAR statements
USSCP	Displays uncorrected sums of squares and crossproducts matrix

Table 105.1 *continued*

Option	Description
ALL	Displays all statistics (CORR, SIMPLE, and USSCP)
NOPRINT	Suppresses output
Other Options	
ALPHA=	Sets significance value for confidence and prediction intervals and tests
SINGULAR=	Sets criterion for checking for singularity

Following are explanations of the *options* that you can specify in the PROC REG statement (in alphabetical order).

Note that any *option* specified in the PROC REG statement applies to all MODEL statements.

ALL

requests the display of many tables. Using the ALL option in the PROC REG statement is equivalent to specifying ALL in every MODEL statement. The ALL option also implies the CORR, SIMPLE, and USSCP options.

ALPHA=*number*

sets the significance level used for the construction of confidence intervals. The value must be between 0 and 1; the default value of 0.05 results in 95% intervals. This option affects the PROC REG option TABLEOUT; the MODEL options CLB, CLI, and CLM; the OUTPUT statement keywords LCL, LCLM, UCL, and UCLM; the PLOT statement keywords LCL., LCLM., UCL., and UCLM.; and the PLOT statement options CONF and PRED.

CORR

displays the correlation matrix for all variables listed in the MODEL or VAR statement.

COVOUT

outputs the covariance matrices for the parameter estimates to the OUTEST= data set. This option is valid only if the OUTEST= option is also specified. See the section "OUTEST= Data Set" on page 8785.

DATA=*SAS-data-set*

names the SAS data set to be used by PROC REG. The data set can be an ordinary SAS data set or a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set. If one of these special TYPE= data sets is used, the OUTPUT, PAINT, PLOT, and REWEIGHT statements, ODS Graphics, and some options in the MODEL and PRINT statements are not available. See Appendix A, "Special SAS Data Sets," for more information about TYPE= data sets. If the DATA= option is not specified, PROC REG uses the most recently created SAS data set.

EDF

outputs the number of regressors in the model excluding and including the intercept, the error degrees of freedom, and the model R square to the OUTEST= data set.

NOPRINT

suppresses the normal display of results. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

OUTEST=*SAS-data-set*

requests that parameter estimates and optional model fit summary statistics be output to this data set. See the section "OUTEST= Data Set" on page 8785 for details. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTSEB

outputs the standard errors of the parameter estimates to the OUTEST= data set. The value SEB for the variable _TYPE_ identifies the standard errors. If the RIDGE= or PCOMIT= option is specified, additional observations are included and identified by the values RIDGESEB and IPCSEB, respectively, for the variable _TYPE_. The standard errors for ridge regression estimates and IPC estimates are limited in their usefulness because these estimates are biased. This option is available for all model selection methods except RSQUARE, ADJRSQ, and CP.

OUTSSCP=*SAS-data-set*

requests that the sums of squares and crossproducts matrix be output to this TYPE=SSCP data set. See the section "OUTSSCP= Data Sets" on page 8791 for details. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

OUTSTB

outputs the standardized parameter estimates as well as the usual estimates to the OUTEST= data set when the RIDGE= or PCOMIT= option is specified. The values RIDGESTB and ICPSTB for the variable _TYPE_ identify ridge regression estimates and IPC estimates, respectively.

OUTVIF

outputs the variance inflation factors (VIF) to the OUTEST= data set when the RIDGE= or PCOMIT= option is specified. The factors are the diagonal elements of the inverse of the correlation matrix of regressors as adjusted by ridge regression or IPC analysis. These observations are identified in the output data set by the values RIDGEVIF and IPCVIF for the variable _TYPE_.

PCOMIT=*list*

requests an incomplete principal component (IPC) analysis for each value *m* in the list. The procedure computes parameter estimates by using all but the last *m* principal components. Each value of *m* produces a set of IPC estimates, which are output to the OUTEST= data set. The values of *m* are saved by the variable _PCOMIT_, and the value of the variable _TYPE_ is set to IPC to identify the estimates. Only nonnegative integers can be specified with the PCOMIT= option.

If you specify the PCOMIT= option, RESTRICT statements are ignored.

PLOTS <(*global-plot-options***)> <=** *plot-request***<(***options***)>>**

PLOTS <(*global-plot-options***)> <=(***plot-request***<(***options***)> <...** *plot-request***<(***options***)>>)>**

controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots        = none
plots        = diagnostics(unpack)
plots        = (all fit(stats=none))
plots(label) = (rstudentbyleverage cooksd)
plots(only)  = (diagnostics(stats=all) fit(nocli stats=(aic sbc)))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc reg;
   model y = x1-x10;
run;

proc reg plots=diagnostics(stats=(default aic sbc));
   model y = x1-x10;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, then PROC REG produces a default set of plots. Table 105.2 lists the default set of plots produced.

Table 105.2 Default Graphs Produced

Plot	Conditional On
DiagnosticsPanel	Unconditional
ResidualPlot	Unconditional
FitPlot	Model with one regressor (excluding intercept)
PartialPlot	PARTIAL option specified in MODEL statement
RidgePanel	RIDGE= option specified in PROC REG or MODEL statement

For models with multiple dependent variables, separate plots are produced for each dependent variable. For jobs with more than one MODEL statement, plots are produced for each model statement.

The *global-options* apply to all plots generated by the REG procedure, unless it is altered by a *specific-plot-option*. The following *global-plot-options* are available:

LABEL

specifies that the LABEL option be applied to each plot that supports a LABEL option. See the descriptions of the specific plots for details.

MAXPOINTS=NONE | *max* < *heat-max* >
suppresses most plots that require processing more than *max* points. When the number of points exceeds *max* but does not exceed *heat-max* divided by the number of independent variables, heat maps are displayed instead of scatter plots for the fit and residual plots. All other plots are suppressed when the number of points exceeds *max*. The default is MAXPOINTS=5000 150000. These cutoffs are ignored if you specify MAXPOINTS=NONE.

MODELLABEL
requests that the model label be displayed in the upper-left corner of all plots. This option is useful when you use more than one MODEL statement.

ONLY
suppress the default plots. Only plots specifically requested are displayed.

STATS=ALL | DEFAULT | NONE | (*plot-statistics***)**
requests statistics that are included on the fit plot and diagnostics panel. Table 105.3 lists the statistics that you can request. STATS=ALL requests all these statistics; STATS=NONE suppresses them.

Table 105.3 Statistics Available on Plots

Keyword	Default	Description
ADJRSQ	x	Adjusted R-square
AIC		Akaike's information criterion
BIC		Sawa's Bayesian information criterion
CP		Mallows' C_p statistic
COEFFVAR		Coefficient of variation
DEPMEAN		Mean of dependent
DEFAULT		All default statistics
EDF	x	Error degrees of freedom
GMSEP		Estimated MSE of prediction, assuming multivariate normality
JP		Final prediction error
MSE	x	Mean squared error
NOBS	x	Number of observations used
NPARM	x	Number of parameters in the model (including the intercept)
PC		Amemiya's prediction criterion
RSQUARE	x	R-square
SBC		SBC statistic
SP		SP statistic
SSE		Error sum of squares

You request statistics in addition to the default set by including the keyword DEFAULT in the *plot-statistics* list.

UNPACK
suppresses paneling.

USEALL
specifies that predicted values at data points with missing dependent variable(s) be included on appropriate plots. By default, only points used in constructing the SSCP matrix appear on plots.

The following specific plots are available:

ADJRSQ < (*adjrsq-options*) >
displays the adjusted R-square values for the models examined when you request variable selection with the SELECTION= option in the MODEL statement.

The following *adjrsq-options* are available for models where you request the RSQUARE, ADJRSQ, or CP selection method:

LABEL
requests that the model number corresponding to the one displayed in the "Subset Selection Summary" table be used to label the model with the largest adjusted R-square statistic at each value of the number of parameters.

LABELVARS
requests that the list (excluding the intercept) of the regressors in the relevant model be used to label the model with the largest adjusted R-square statistic at each value of the number of parameters.

AIC < (*aic-options*) >
displays Akaike's information criterion (AIC) for the models examined when you request variable selection with the SELECTION= option in the MODEL statement.

The following *aic-options* are available for models where you request the RSQUARE, ADJRSQ, or CP selection method:

LABEL
requests that the model number corresponding to the one displayed in the "Subset Selection Summary" table be used to label the model with the smallest AIC statistic at each value of the number of parameters.

LABELVARS
requests that the list (excluding the intercept) of the regressors in the relevant model be used to label the model with the smallest AIC statistic at each value of the number of parameters.

ALL
produces all appropriate plots.

BIC < (*bic-options*) >
displays Sawa's Bayesian information criterion (BIC) for the models examined when you request variable selection with the SELECTION= option in the MODEL statement.

The following *bic-options* are available for models where you request the RSQUARE, ADJRSQ, or CP selection method:

LABEL
requests that the model number corresponding to the one displayed in the "Subset Selection Summary" table be used to label the model with the smallest BIC statistic at each value of the number of parameters.

LABELVARS
requests that the list (excluding the intercept) of the regressors in the relevant model be used to label the model with the smallest BIC statistic at each value of the number of parameters.

COOKSD < (LABEL) >
plots Cook's D statistic by observation number. Observations whose Cook's D statistic lies above the horizontal reference line at value $4/n$, where n is the number of observations used, are deemed to be influential (Rawlings, Pantula, and Dickey 1998). If you specify the LABEL option, then points deemed as influential are labeled. If you do not specify an ID variable, the observation number within the current BY group is used as the label. If you specify one or more ID variables in one or more ID statements, then the first ID variable you specify is used for the labeling.

CP < (*cp-options*) >
displays Mallows' C_p statistic for the models examined when you request variable selection with the SELECTION= option in the MODEL statement. For models where you request the RSQUARE, ADJRSQ, or CP selection, reference lines corresponding to the equations $C_p = p$ and $C_p = 2p - p_{full}$, where p_{full} is the number of parameters in the full model (excluding the intercept) and p is the number of parameters in the subset model (including the intercept), are displayed on the plot of C_p versus p. For the purpose of parameter estimation, Hocking (1976) suggests selecting a model where $C_p \leq 2p - p_{full}$. For the purpose of prediction, Hocking suggests the criterion $C_p \leq p$. Mallows (1973) suggests that all subset models with C_p small and near p be considered for further study.

The following *cp-options* are available for models where you request the RSQUARE, ADJRSQ, or CP selection method:

LABEL
requests that the model number corresponding to the one displayed in the "Subset Selection Summary" table be used to label the model with the smallest C_p statistic at each value of the number of parameters.

LABELVARS
requests that the list (excluding the intercept) of the regressors in the relevant model be used to label the model with the smallest C_p statistic at each value of the number of parameters.

CRITERIA | CRITERIONPANEL < (*criteria-options*) >
produces a panel of fit criteria for the models examined when you request variable selection with the SELECTION= option in the MODEL statement. The fit criteria displayed are R-square, adjusted R-square, Mallows' C_p, Akaike's information criterion (AIC), Sawa's Bayesian information criterion (BIC), and Schwarz's Bayesian information criterion (SBC). For SELECTION=RSQUARE, SELECTION=ADJRSQ, or SELECTION=CP, scatter plots of these statistics versus the number of parameters (including the intercept) are displayed. For other selection methods, line plots of these statistics as function of the selection step number are displayed.

The following *criteria-options* are available:

LABEL

requests that the model number corresponding to the one displayed in the "Subset Selection Summary" table be used to label the best model at each value of the number of parameters. This option applies only to the RSQUARE, ADJRSQ, and CP selection methods.

LABELVARS

requests that the list (excluding the intercept) of the regressors in the relevant model be used to label the best model at each value of the number of parameters. Since these labels are typically long, LABELVARS is supported only when the panel is unpacked. This option applies only to the RSQUARE, ADJRSQ, and CP selection methods.

UNPACK

suppresses paneling. Separate plots are produced for each of the six fit statistics. For models where you request the RSQUARE, ADJRSQ, or CP selection, two reference lines corresponding to the equations $C_p = p$ and $C_p = 2p - p_{full}$, where p_{full} is the number of parameters in the full model (excluding the intercept) and p is the number of parameters in the subset model (including the intercept), are displayed on the plot of C_p versus p. For the purpose of parameter estimation, Hocking (1976) suggests selecting a model where $C_p \leq 2p - p_{full}$. For the purpose of prediction, Hocking suggests the criterion $C_p \leq p$. Mallows (1973) suggests that all subset models with C_p small and near p be considered for further study.

DFBETAS < (*DFBETAS-options*) >

produces panels of DFBETAS by observation number for the regressors in the model. Note that each panel contains at most six plots, and multiple panels are used in the case where there are more than six regressors (including the intercept) in the model. Observations whose DFBETAS' statistics for a regressor are greater in magnitude than $2/\sqrt{n}$, where n is the number of observations used, are deemed to be influential for that regressor (Rawlings, Pantula, and Dickey 1998).

The following *DFBETAS-options* are available:

COMMONAXES

specifies that the same DFBETAS axis be used in all panels when multiple panels are needed. By default, the DFBETAS axis is chosen independently for each panel. If you also specify the UNPACK option, then the same DFBETAS axis is used for each regressor.

LABEL

specifies that observations whose magnitude are greater than $2/\sqrt{n}$ be labeled. If you do not specify an ID variable, the observation number within the current BY group is used as the label. If you specify one or more ID variables on one or more ID statements, then the first ID variable you specify is used for the labeling.

UNPACK

suppresses paneling. The DFBETAS statistics for each regressor are displayed on separate plots.

DFFITS < (LABEL) >

plots the DFFITS statistic by observation number. Observations whose DFFITS' statistic is greater in magnitude than $2\sqrt{p/n}$, where n is the number of observations used and p is the number of regressors, are deemed to be influential (Rawlings, Pantula, and Dickey 1998). If you specify the LABEL option, then these influential observations are labeled. If you do not specify an ID variable, the observation number within the current BY group is used as the label. If you specify one or more ID variables in one or more ID statements, then the first ID variable you specify is used for the labeling.

DIAGNOSTICS < (*diagnostics-options*) >

produces a summary panel of fit diagnostics:

- residuals versus the predicted values
- studentized residuals versus the predicted values
- studentized residuals versus the leverage
- normal quantile plot of the residuals
- dependent variable values versus the predicted values
- Cook's D versus observation number
- histogram of the residuals
- "Residual-Fit" (or RF) plot consisting of side-by-side quantile plots of the centered fit and the residuals
- box plot of the residuals if you specify the STATS=NONE suboption

You can specify the following *diagnostics-options*:

STATS=*stats-options*

determines which model fit statistics are included in the panel. See the global STATS= suboption for details. The PLOTS= suboption of the DIAGNOSTICSPANEL option overrides the global PLOTS= suboption.

UNPACK

produces the eight plots in the panel as individual plots. Note that you can also request individual plots in the panel by name without having to unpack the panel.

FITPLOT | FIT < (*fit-options*) >

produces a scatter plot of the data overlaid with the regression line, confidence band, and prediction band for models that depend on at most one regressor excluding the intercept. When the number of points exceeds the MAXPOINTS=*max* value, a heat map is displayed instead of a scatter plot. By default, heat maps are not displayed if the number of observations times the number of independent variables is greater than 150,000. See the MAXPOINTS= option.

You can specify the following *fit-options*:

NOCLI

suppresses the prediction limits.

NOCLM
suppresses the confidence limits.

NOLIMITS
suppresses the confidence and prediction limits.

STATS=*stats-options*
determines which model fit statistics are included in the panel. See the global STATS= suboption for details. The PLOTS= suboption of the FITPLOT option overrides the global PLOTS= suboption.

OBSERVEDBYPREDICTED < (LABEL) >
plots dependent variable values by the predicted values. If you specify the LABEL option, then points deemed as outliers or influential (see the RSTUDENTBYLEVERAGE option for details) are labeled.

NONE
suppresses all plots.

PARTIAL < (UNPACK) >
produces panels of partial regression plots for each regressor with at most six regressors per panel. If you specify the UNPACK option, then all partial plot panels are unpacked.

PREDICTIONS (X=*numeric-variable* **<** *prediction-options* **>)**
produces a panel of two plots whose horizontal axis is the variable you specify in the required X= suboption. The upper plot in the panel is a scatter plot of the residuals. The lower plot shows the data overlaid with the regression line, confidence band, and prediction band. This plot is appropriate for models where all regressors are known to be functions of the single variable that you specify in the X= suboption.

You can specify the following *prediction-options*:

NOCLI
suppresses the prediction limits.

NOCLM
suppresses the confidence limits

NOLIMITS
suppresses the confidence and prediction limits

SMOOTH
requests a nonparametric smoothing of the residuals as a function of the variable you specify in the X= suboption. This nonparametric fit is a loess fit that uses local linear polynomials, linear interpolation, and a smoothing parameter that is selected to yield a local minimum of the corrected Akaike's information criterion (AICC). See Chapter 78, "The LOESS Procedure," for details. The SMOOTH option is not supported when a FREQ statement is used.

UNPACK
suppresses paneling.

QQPLOT | QQ
produces a normal quantile plot of the residuals.

RESIDUALBOXPLOT | BOXPLOT < (LABEL) >
produces a box plot consisting of the residuals. If you specify label option, points deemed far-outliers are labeled. If you do not specify an ID variable, the observation number within the current BY group is used as the label. If you specify one or more ID variables in one or more ID statements, then the first ID variable you specify is used for the labeling.

RESIDUALBYPREDICTED < (LABEL) >
plots residuals by predicted values. If you specify the LABEL option, then points deemed as outliers or influential (see the RSTUDENTBYLEVERAGE option for details) are labeled.

RESIDUALCHART < (*residual-chart-options*) >

RC < (*residual-chart-options*) >
produces the residual chart and enables you to specify *residual-chart-options*. This chart displays studentized residuals and Cook's *D* in side-by-side bar charts. This chart is also displayed when you specify the R option in the MODEL statement.

Unlike most graphs, the height of this chart can vary as a function of the number of observations that appear in the chart. You can specify the following *residual-chart-options* to control the height and other aspects of the chart:

COMPUTEHEIGHT=*a b < max >*

CH=*a b < max >*
specifies the constants for computing the height of the chart. For n observations, intercept a, slope b, and maximum height *max*, the height is $\min(a + b\,(n + 1), max)$. By default, COMPUTEHEIGHT=150 15 1650. Thus, the default height in pixels is $\min(150 + 15(n + 1), 1650)$. The default unit is pixels, and you can use the UNIT= *residual-chart-option* to change the unit to inches or centimeters.

MAX=*max*
species the maximum number of points to display in each chart. When the number of points exceeds *max*, charts of up to *max* observations are displayed until all observations are displayed.

SETHEIGHT=*height*

SH=*height*
specifies the height of the chart. By default, the height is based on the COMPUTEHEIGHT= option. The default unit is pixels, and you can use the UNIT= *residual-chart-option* to change the unit to inches or centimeters.

UNIT=PX | IN | CM
specifies the unit (pixels, inches, or centimeters) for the SETHEIGHT= and COMPUTEHEIGHT= *residual-chart-options*. Inches equals pixels divided by 96, and centimeters equals inches times 2.54. By default, UNIT=PX.

UNPACK
suppresses paneling. The studentized residuals and Cook's *D* are displayed in separate charts. When you specify the UNPACK *residual-chart-option*, residuals, standard errors, and other values that go into the computations are added to each chart.

RESIDUALS < (*residual-options*) >
produces panels of the residuals versus the regressors in the model. Each panel contains at most six plots, and multiple panels are used when the model contains more than six regressors (including the intercept). When the number of points exceeds the MAXPOINTS=*max* value, a heat map is displayed instead of a scatter plot. By default, heat maps are not displayed if the number of observations times the number of independent variables is greater than 150,000. See the MAXPOINTS= option. You can specify the following *residual-options*:

SMOOTH
requests a nonparametric smoothing of the residuals for each regressor. Each nonparametric fit is a loess fit that uses local linear polynomials, linear interpolation, and a smoothing parameter that is selected to yield a local minimum of the corrected Akaike's information criterion (AICC). See Chapter 78, "The LOESS Procedure," for details. The SMOOTH option is not supported when a FREQ statement is used.

UNPACK
suppresses paneling.

RESIDUALHISTOGRAM
produces a histogram of the residuals.

RFPLOT | RF
produces a "Residual-Fit" (or RF) plot consisting of side-by-side quantile plots of the centered fit and the residuals. This plot "shows how much variation in the data is explained by the fit and how much remains in the residuals" (Cleveland 1993).

RIDGE | RIDGEPANEL | RIDGEPLOT < (*ridge-options*) >
creates panels of VIF values and standardized ridge estimates by ridge values for each coefficient. The VIF values for each coefficient are connected by lines and are displayed in the upper plot in each panel. The points corresponding to the standardized estimates of each coefficient are connected by lines and are displayed in the lower plot in each panel. By default, at most 10 coefficients are represented in a panel and multiple panels are produced for models with more than 10 regressors. For ridge estimates to be computed and plotted, the OUTEST= option must be specified in the PROC REG statement, and the RIDGE= list must be specified in either the PROC REG or the MODEL statement. (See Example 105.5.)

The following *ridge-options* are available:

COMMONAXES
specifies that the same VIF axis and the same standardized estimate axis are used in all panels when multiple panels are needed. By default, these axes are chosen independently for the regressors shown in each panel.

RIDGEAXIS=LINEAR | LOG
specifies the axis type used to display the ridge parameters. The default is RIDGEAXIS=LINEAR. Note that the point with the ridge parameter equal to zero is not displayed if you specify RIDGEAXIS=LOG.

UNPACK
suppresses paneling. The traces of the VIF statistics and standardized estimates are shown in separate plots.

VARSPERPLOT=ALL
VARSPERPLOT=*number*
specifies the maximum number of regressors displayed in each panel or in each plot if you additionally specify the UNPACK option. If you specify VARSPERPLOT=ALL, then the VIF values and ridge traces for all regressors are displayed in a single panel.

VIFAXIS=LINEAR | LOG
specifies the axis type used to display the VIF statistics. The default is VIFAXIS=LINEAR.

RSQUARE < (*rsquare-options*) >
displays the R-square values for the models examined when you request variable selection with the SELECTION= option in the MODEL statement.

The following *rsquare-options* are available for models where you request the RSQUARE, ADJRSQ, or CP selection method:

LABEL
requests that the model number corresponding to the one displayed in the "Subset Selection Summary" table be used to label the model with the largest R-square statistic at each value of the number of parameters.

LABELVARS
requests that the list (excluding the intercept) of the regressors in the relevant model be used to label the model with the largest R-square statistic at each value of the number of parameters.

RSTUDENTBYLEVERAGE < (LABEL) >
plots studentized residuals by leverage. Observations whose studentized residuals lie outside the band between the reference lines RSTUDENT $= \pm 2$ are deemed outliers. Observations whose leverage values are greater than the vertical reference LEVERAGE $= 2p/n$, where p is the number of parameters including the intercept and n is the number of observations used, are deemed influential (Rawlings, Pantula, and Dickey 1998). If you specify the LABEL option, then points deemed as outliers or influential are labeled. If you do not specify an ID variable, the observation number within the current BY group is used as the label. If you specify one or more ID variables in one or more ID statements, then the first ID variable you specify is used for the labeling.

RSTUDENTBYPREDICTED < (LABEL) >
plots studentized residuals by predicted values. If you specify the LABEL option, then points deemed as outliers or influential (see the RSTUDENTBYLEVERAGE option for details) are labeled.

SBC < (*sbc-options*) >
displays Schwarz's Bayesian information criterion (SBC) for the models examined when you request variable selection with the SELECTION= option in the MODEL statement.

The following *sbc-options* are available for models where you request the RSQUARE, ADJRSQ, or CP selection method:

LABEL
requests that the model number corresponding to the one displayed in the "Subset Selection Summary" table be used to label the model with the smallest SBC statistic at each value of the number of parameters.

LABELVARS
requests that the list (excluding the intercept) of the regressors in the relevant model be used to label the model with the smallest SBC statistic at each value of the number of parameters.

PRESS
outputs the PRESS statistic to the OUTEST= data set. The values of this statistic are saved in the variable _PRESS_. This option is available for all model selection methods except RSQUARE, ADJRSQ, and CP.

RIDGE=*list*
requests a ridge regression analysis and specifies the values of the ridge constant k (see the section "Computations for Ridge Regression and IPC Analysis" on page 8830). Each value of k produces a set of ridge regression estimates that are placed in the OUTEST= data set. The values of k are saved by the variable _RIDGE_, and the value of the variable _TYPE_ is set to RIDGE to identify the estimates.

Only nonnegative numbers can be specified with the RIDGE= option. Example 105.5 illustrates this option.

If ODS Graphics is enabled (see the section "ODS Graphics" on page 8852), then ridge regression plots are automatically produced. These plots consist of panels containing ridge traces for the regressors, with at most eight ridge traces per panel.

If you specify the RIDGE= option, RESTRICT statements are ignored.

RSQUARE
has the same effect as the EDF option.

SIMPLE
displays the sum, mean, variance, standard deviation, and uncorrected sum of squares for each variable used in PROC REG.

SINGULAR=*n*

tunes the mechanism used to check for singularities. The default value is machine dependent but is approximately 1E–9 on most machines. This option is rarely needed.

Singularity checking is described in the section "Computational Methods" on page 8831.

TABLEOUT

outputs the standard errors and $100(1-\alpha)\%$ confidence limits for the parameter estimates, the t statistics for testing if the estimates are zero, and the associated p-values to the OUTEST= data set. The _TYPE_ variable values STDERR, LnB, UnB, T, and PVALUE, where $n = 100(1-\alpha)$, identify these rows in the OUTEST= data set. The α level can be set with the ALPHA= option in the PROC REG or MODEL statement. The OUTEST= option must be specified in the PROC REG statement for this option to take effect.

USSCP

displays the uncorrected sums-of-squares and crossproducts matrix for all variables used in the procedure.

ADD Statement

ADD *variables* **;**

The ADD statement adds independent variables to the regression model. Only variables used in the VAR statement or used in MODEL statements before the first RUN statement can be added to the model. You can use the ADD statement interactively to add variables to the model or to include a variable that was previously deleted with a DELETE statement. Each use of the ADD statement modifies the MODEL label.

See the section "Interactive Analysis" on page 8792 for an example.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC REG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the REG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

When a BY statement is used with PROC REG, interactive processing is not possible; that is, once the first RUN statement is encountered, processing proceeds for each BY group in the data set, and no further statements are accepted by the procedure. A BY statement that appears after the first RUN statement is ignored.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CODE Statement

CODE < *options* > **;**

The CODE statement writes SAS DATA step code for computing predicted values of the fitted model either to a file or to a catalog entry. This code can then be included in a DATA step to score new data.

Table 105.4 summarizes the *options* available in the CODE statement.

Table 105.4 CODE Statement Options

Option	Description
CATALOG=	Names the catalog entry where the generated code is saved
DUMMIES	Retains the dummy variables in the data set
ERROR	Computes the error function
FILE=	Names the file where the generated code is saved
FORMAT=	Specifies the numeric format for the regression coefficients
GROUP=	Specifies the group identifier for array names and statement labels
IMPUTE	Imputes predicted values for observations with missing or invalid covariates
LINESIZE=	Specifies the line size of the generated code
LOOKUP=	Specifies the algorithm for looking up CLASS levels
RESIDUAL	Computes residuals

For details about the syntax of the CODE statement, see the section "CODE Statement" on page 424 in Chapter 20, "Shared Concepts and Topics."

DELETE Statement

DELETE *variables* **;**

The DELETE statement deletes independent variables from the regression model. The DELETE statement performs the opposite function of the ADD statement and is used in a similar manner. Each use of the DELETE statement modifies the MODEL label.

For an example of how the ADD statement is used (and how the DELETE statement can be used), see the section "Interactive Analysis" on page 8792.

FREQ Statement

FREQ *variable* **;**

When a FREQ statement appears, each observation in the input data set is assumed to represent *n* observations, where *n* is the value of the FREQ variable. The analysis produced when you use a FREQ statement is the same as an analysis produced by using a data set that contains *n* observations in place of each observation in the input data set. When the procedure determines degrees of freedom for significance tests, the total number of observations is considered to be equal to the sum of the values of the FREQ variable.

If the value of the FREQ variable is missing or is less than 1, the observation is not used in the analysis. If the value is not an integer, only the integer portion is used.

The FREQ statement must appear before the first RUN statement, or it is ignored.

ID Statement

ID *variables* **;**

When one of the MODEL statement options CLI, CLM, P, R, and INFLUENCE is requested, the variables listed in the ID statement are displayed beside each observation. These variables can be used to identify each observation. If the ID statement is omitted, the observation number is used to identify the observations.

Although there are no restrictions on the length of ID variables, PROC REG might truncate ID values to 16 characters for display purposes.

MODEL Statement

< *label*: > **MODEL** *dependents* **=** < *regressors* > < / *options* > **;**

After the keyword MODEL, the dependent (response) variables are specified, followed by an equal sign and the regressor variables. Variables specified in the MODEL statement must be numeric variables in the data set being analyzed. For example, if you want to specify a quadratic term for variable X1 in the model, you cannot use X1*X1 in the MODEL statement but must create a new variable (for example, X1SQUARE=X1*X1) in a DATA step and use this new variable in the MODEL statement. The label in the MODEL statement is optional.

Table 105.5 summarizes the *options* available in the MODEL statement. Equations for the statistics available are given in the section "Model Fit and Diagnostic Statistics" on page 8808.

Table 105.5 MODEL Statement Options

Option	Description
Model Selection and Details of Selection	
SELECTION=	Specifies model selection method
BEST=	Specifies maximum number of subset models displayed or output to the OUTEST= data set
DETAILS	Produces summary statistics at each step
DETAILS=	Specifies the display details for FORWARD, BACKWARD, and STEPWISE methods
GROUPNAMES=	Provides names for groups of variables
INCLUDE=	Includes first n variables in the model
MAXSTEP=	Specifies maximum number of steps that might be performed
NOINT	Fits a model without the intercept term
PCOMIT=	Performs incomplete principal component analysis and outputs estimates to the OUTEST= data set
RIDGE=	Performs ridge regression analysis and outputs estimates to the OUTEST= data set
SLE=	Sets criterion for entry into model
SLS=	Sets criterion for staying in model
START=	Specifies number of variables in model to begin the comparing and switching process
STOP=	Stops selection criterion
Statistics	
ADJRSQ	Computes adjusted R square
AIC	Computes Akaike's information criterion
B	Computes parameter estimates for each model
BIC	Computes Sawa's Bayesian information criterion
CP	Computes Mallows' C_p statistic
GMSEP	Computes estimated MSE of prediction assuming multivariate normality
JP	Computes J_p, the final prediction error
MSE	Computes MSE for each model
PC	Computes Amemiya's prediction criterion

Table 105.5 *continued*

Option	Description
RMSE	Displays root MSE for each model
SBC	Computes the SBC statistic
SP	Computes S_p statistic for each model
SSE	Computes error sum of squares for each model
Data Set Options	
EDF	Outputs the number of regressors, the error degrees of freedom, and the model R square to the OUTEST= data set
OUTSEB	Outputs standard errors of the parameter estimates to the OUTEST= data set
OUTSTB	Outputs standardized parameter estimates to the OUTEST= data set. Use only with the RIDGE= or PCOMIT= option.
OUTVIF	Outputs the variance inflation factors to the OUTEST= data set. Use only with the RIDGE= or PCOMIT= option.
PRESS	Outputs the PRESS statistic to the OUTEST= data set
RSQUARE	Has same effect as the EDF option
Regression Calculations	
I	Displays inverse of sums of squares and crossproducts
XPX	Displays sums-of-squares and crossproducts matrix
Details on Estimates	
ACOV	Displays heteroscedasticity- consistent covariance matrix of estimates and heteroscedasticity-consistent standard errors
ACOVMETHOD=	Specifies method for computing the asymptotic heteroscedasticity-consistent covariance matrix
COLLIN	Produces collinearity analysis
COLLINOINT	Produces collinearity analysis with intercept adjusted out
CORRB	Displays correlation matrix of estimates
COVB	Displays covariance matrix of estimates
HCC	Displays heteroscedasticity-consistent standard errors
HCCMETHOD=	Specifies method for computing the asymptotic heteroscedasticity-consistent covariance matrix
LACKFIT	Performs lack-of-fit test
PARTIALR2	Displays squared semipartial correlation coefficients computed using Type I sums of squares
PCORR1	Displays squared partial correlation coefficients computed using Type I sums of squares
PCORR2	Displays squared partial correlation coefficients computed using Type II sums of squares
SCORR1	Displays squared semipartial correlation coefficients computed using Type I sums of squares
SCORR2	Displays squared semipartial correlation coefficients computed using Type II sums of squares
SEQB	Displays a sequence of parameter estimates during selection process
SPEC	Tests that first and second moments of model are correctly specified

Table 105.5 *continued*

Option	Description
SRT	Displays the studentized residuals by using a series of asterisks
SS1	Displays the sequential sums of squares
SS2	Displays the partial sums of squares
STB	Displays standardized parameter estimates
TOL	Displays tolerance values for parameter estimates
WHITE	Displays heteroscedasticity-consistent standard errors
VIF	Computes variance-inflation factors
Predicted and Residual Values	
CLB	Computes $100(1-\alpha)\%$ confidence limits for the parameter estimates
CLI	Computes $100(1-\alpha)\%$ confidence limits for an individual predicted value
CLM	Computes $100(1-\alpha)\%$ confidence limits for the expected value of the dependent variable
DW	Computes a Durbin-Watson statistic
DWPROB	Computes a Durbin-Watson statistic and *p*-value
INFLUENCE	Computes influence statistics
P	Computes predicted values
PARTIAL	Displays partial regression plots for each regressor
PARTIALDATA	Displays partial regression data
R	Produces analysis of residuals
Display Options and Other Options	
ALL	Requests the following options: ACOV, CLB, CLI, CLM, CORRB, COVB, HCC, I, P, PCORR1, PCORR2, R, SCORR1, SCORR2, SEQB, SPEC, SS1, SS2, STB, TOL, VIF, XPX
ALPHA=	Sets significance value for confidence and prediction intervals and tests
NOPRINT	Suppresses display of results
SIGMA=	Specifies the true standard deviation of error term for computing CP and BIC
SINGULAR=	Sets criterion for checking for singularity

You can specify the following *options* in the MODEL statement after a slash (/).

ACOV

displays the estimated asymptotic covariance matrix of the estimates under the hypothesis of heteroscedasticity and heteroscedasticity-consistent standard errors of parameter estimates. See the HCCMETHOD= option and the HCC option and the section "Testing for Heteroscedasticity" on page 8824 for more information.

ACOVMETHOD=0,1,2, or 3

See the HCCMETHOD= option.

ADJRSQ

computes R square adjusted for degrees of freedom for each model selected (Darlington 1968; Judge et al. 1980).

AIC

outputs Akaike's information criterion for each model selected (Akaike 1969; Judge et al. 1980) to the OUTEST= data set. If SELECTION=ADJRSQ, SELECTION=RSQUARE, or SELECTION=CP is specified, then the AIC statistic is also added to the SubsetSelSummary table.

ALL

requests all these *options*: ACOV, CLB, CLI, CLM, CORRB, COVB, HCC, I, P, PCORR1, PCORR2, R, SCORR1, SCORR2, SEQB, SPEC, SS1, SS2, STB, TOL, VIF, and XPX.

ALPHA=*number*

sets the significance level used for the construction of confidence intervals for the current MODEL statement. The value must be between 0 and 1; the default value of 0.05 results in 95% intervals. This option affects the MODEL options CLB, CLI, and CLM; the OUTPUT statement keywords LCL, LCLM, UCL, and UCLM; the PLOT statement keywords LCL., LCLM., UCL., and UCLM.; and the PLOT statement options CONF and PRED. If you specify this option in the MODEL statement, it takes precedence over the ALPHA= option in the PROC REG statement.

B

is used with the RSQUARE, ADJRSQ, and CP model-selection methods to compute estimated regression coefficients for each model selected.

BEST=*n*

is used with the RSQUARE, ADJRSQ, and CP model-selection methods. If SELECTION=CP or SELECTION=ADJRSQ is specified, the BEST= option specifies the maximum number of subset models to be displayed or output to the OUTEST= data set. For SELECTION=RSQUARE, the BEST= option requests the maximum number of subset models for each size.

If the BEST= option is used without the B option (displaying estimated regression coefficients), the variables in each MODEL are listed in order of inclusion instead of the order in which they appear in the MODEL statement.

If the BEST= option is omitted and the number of regressors is less than 11, all possible subsets are evaluated. If the BEST= option is omitted and the number of regressors is greater than 10, the number of subsets selected is, at most, equal to the number of regressors. A small value of the BEST= option greatly reduces the CPU time required for large problems.

BIC

outputs Sawa's Bayesian information criterion for each model selected (Sawa 1978; Judge et al. 1980) to the OUTEST= data set. If SELECTION=ADJRSQ, SELECTION=RSQUARE, or SELECTION=CP is specified, then the BIC statistic is also added to the SubsetSelSummary table.

CLB

requests the $100(1-\alpha)\%$ upper and lower confidence limits for the parameter estimates. By default, the 95% limits are computed; the ALPHA= option in the PROC REG or MODEL statement can be used to change the α level. If any of the MODEL statement options ACOV, HCC, or WHITE are in effect, then the CLB option also produces heteroscedasticity-consistent $100(1-\alpha)\%$ upper and lower confidence limits for the parameter estimates.

CLI

requests the $100(1-\alpha)\%$ upper and lower confidence limits for an individual predicted value. By default, the 95% limits are computed; the ALPHA= option in the PROC REG or MODEL statement can be used to change the α level. The confidence limits reflect variation in the error, as well as variation in the parameter estimates. See the section "Predicted and Residual Values" on page 8801 and Chapter 4, "Introduction to Regression Procedures," for more information.

CLM

displays the $100(1-\alpha)\%$ upper and lower confidence limits for the expected value of the dependent variable (mean) for each observation. By default, the 95% limits are computed; the ALPHA= in the PROC REG or MODEL statement can be used to change the α level. This is not a prediction interval (see the CLI option) because it takes into account only the variation in the parameter estimates, not the variation in the error term. See the section "Predicted and Residual Values" on page 8801 and Chapter 4, "Introduction to Regression Procedures," for more information.

COLLIN

requests a detailed analysis of collinearity among the regressors. This includes eigenvalues, condition indices, and decomposition of the variances of the estimates with respect to each eigenvalue. See the section "Collinearity Diagnostics" on page 8806.

COLLINOINT

requests the same analysis as the COLLIN option with the intercept variable adjusted out rather than included in the diagnostics. See the section "Collinearity Diagnostics" on page 8806.

CORRB

displays the correlation matrix of the estimates. This is the $(\mathbf{X}'\mathbf{X})^{-1}$ matrix scaled to unit diagonals.

COVB

displays the estimated covariance matrix of the estimates. This matrix is $(\mathbf{X}'\mathbf{X})^{-1}s^2$, where s^2 is the estimated mean squared error.

CP

outputs Mallows' C_p statistic for each model selected (Mallows 1973; Hocking 1976) to the OUTEST= data set. See the section "Criteria Used in Model-Selection Methods" on page 8798 for a discussion of the use of C_p. If SELECTION=ADJRSQ, SELECTION=RSQUARE, or SELECTION=CP is specified, then the C_p statistic is also added to the SubsetSelSummary table.

DETAILS

DETAILS=*name*

specifies the level of detail produced when the BACKWARD, FORWARD, or STEPWISE method is used, where *name* can be ALL, STEPS, or SUMMARY. The DETAILS or DETAILS=ALL option produces entry and removal statistics for each variable in the model building process, ANOVA and parameter estimates at each step, and a selection summary table. The option DETAILS=STEPS provides the step information and summary table. The option DETAILS=SUMMARY produces only the summary table. The default if the DETAILS option is omitted is DETAILS=STEPS.

DW

calculates a Durbin-Watson statistic to test whether or not the errors have first-order autocorrelation. (This test is appropriate only for time series data.) Note that your data should be sorted by the date/time ID variable before you use this option. The sample autocorrelation of the residuals is also produced. See the section "Autocorrelation in Time Series Data" on page 8829.

DWPROB

calculates a Durbin-Watson statistic and a p-value to test whether or not the errors have first-order autocorrelation. Note that it is not necessary to specify the DW option if the DWPROB option is specified. (This test is appropriate only for time series data.) Note that your data should be sorted by the date/time ID variable before you use this option. The sample autocorrelation of the residuals is also produced. See the section "Autocorrelation in Time Series Data" on page 8829.

EDF

outputs the number of regressors in the model excluding and including the intercept, the error degrees of freedom, and the model R square to the OUTEST= data set.

GMSEP

outputs the estimated mean square error of prediction assuming that both independent and dependent variables are multivariate normal (Stein 1960; Darlington 1968) to the OUTEST= data set. (Note that Hocking's formula (1976, eq. 4.20) contains a misprint: "$n - 1$" should read "$n - 2$.") If SELECTION=ADJRSQ, SELECTION=RSQUARE, or SELECTION=CP is specified, then the GMSEP statistic is also added to the SubsetSelSummary table.

GROUPNAMES=*'name1' 'name2' . . .*

provides names for variable groups. This option is available only in the BACKWARD, FORWARD, and STEPWISE methods. The group name can be up to 32 characters. Subsets of independent variables listed in the MODEL statement can be designated as variable groups. This is done by enclosing the appropriate variables in braces. Variables in the same group are entered into or removed from the regression model at the same time. However, if the tolerance of any variable (see the TOL option on page 8770) in a group is less than the setting of the SINGULAR= option, then the variable is not entered into the model with the rest of its group. If the GROUPNAMES= option is not used, then the names GROUP1, GROUP2, . . . , GROUPn are assigned to groups encountered in the MODEL statement. Variables not enclosed by braces are used as groups of a single variable.

For example:

```
model y={x1 x2} x3 / selection=stepwise
  groupnames='x1 x2' 'x3';
```

Another example:

```
model y={ht wgt age} bodyfat / selection=forward
  groupnames='htwgtage' 'bodyfat';
```

HCC

requests heteroscedasticity-consistent standard errors of the parameter estimates. You can use the HCCMETHOD= option to specify the method used to compute the heteroscedasticity-consistent covariance matrix.

HCCMETHOD=0,1,2, or 3

specifies the method used to obtain a heteroscedasticity-consistent covariance matrix for use with the ACOV, HCC, or WHITE option in the MODEL statement and for heteroscedasticity-consistent tests with the TEST statement. The default is HCCMETHOD=0. See the section "Testing for Heteroscedasticity" on page 8824 for details.

I

displays the $(\mathbf{X}'\mathbf{X})^{-1}$ matrix. The inverse of the crossproducts matrix is bordered by the parameter estimates and SSE matrices.

INCLUDE=*n*

forces the first *n* independent variables listed in the MODEL statement to be included in all models. The selection methods are performed on the other variables in the MODEL statement. The INCLUDE= option is not available with SELECTION=NONE.

INFLUENCE

requests a detailed analysis of the influence of each observation on the estimates and the predicted values. See the section "Influence Statistics" on page 8810 for details.

JP

outputs J_p, the estimated mean square error of prediction for each model selected assuming that the values of the regressors are fixed and that the model is correct to the OUTEST= data set. The J_p statistic is also called the final prediction error (FPE) by Akaike (Nicholson 1948; Lord 1950; Mallows 1967; Darlington 1968; Rothman 1968; Akaike 1969; Hocking 1976; Judge et al. 1980) If SELECTION=ADJRSQ, SELECTION=RSQUARE, or SELECTION=CP is specified, then the J_p statistic is also added to the SubsetSelSummary table.

LACKFIT

performs a lack-of-fit test. See the section "Testing for Lack of Fit" on page 8825 for more information. See Draper and Smith (1981) for a discussion of lack-of-fit tests.

MSE

computes the mean square error for each model selected (Darlington 1968).

MAXSTEP=*n*

specifies the maximum number of steps that are done when SELECTION=FORWARD, SELECTION=BACKWARD, or SELECTION=STEPWISE is used. The default value is the number of independent variables in the model for the FORWARD and BACKWARD methods and three times this number for the stepwise method.

NOINT

suppresses the intercept term that is otherwise included in the model.

NOPRINT

suppresses the normal display of regression results. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

OUTSEB

outputs the standard errors of the parameter estimates to the OUTEST= data set. The value SEB for the variable _TYPE_ identifies the standard errors. If the RIDGE= or PCOMIT= option is specified, additional observations are included and identified by the values RIDGESEB and IPCSEB, respectively, for the variable _TYPE_. The standard errors for ridge regression estimates and incomplete principal components (IPC) estimates are limited in their usefulness because these estimates are biased. This option is available for all model-selection methods except RSQUARE, ADJRSQ, and CP.

OUTSTB

outputs the standardized parameter estimates as well as the usual estimates to the OUTEST= data set when the RIDGE= or PCOMIT= option is specified. The values RIDGESTB and IPCSTB for the variable _TYPE_ identify ridge regression estimates and IPC estimates, respectively.

OUTVIF

outputs the variance inflation factors (VIF) to the OUTEST= data set when the RIDGE= or PCOMIT= option is specified. The factors are the diagonal elements of the inverse of the correlation matrix of regressors as adjusted by ridge regression or IPC analysis. These observations are identified in the output data set by the values RIDGEVIF and IPCVIF for the variable _TYPE_.

P

calculates predicted values from the input data and the estimated model. The display includes the observation number, the ID variable (if one is specified), the actual and predicted values, and the residual. If the CLI, CLM, or R option is specified, the P option is unnecessary. See the section "Predicted and Residual Values" on page 8801 for more information.

PARTIAL

requests partial regression leverage plots for each regressor. You can use the PARTIALDATA option to obtain a tabular display of the partial regression leverage data. If ODS Graphics is enabled (see the section "ODS Graphics" on page 8852), then these partial plots are produced in panels with up to six plots per panel. See the section "Influence Statistics" on page 8810 for more information.

PARTIALDATA

requests partial regression leverage data for each regressor. You can request partial regression leverage plots of these data with the PARTIAL option. See the section "Influence Statistics" on page 8810 for more information.

PARTIALR2 < (< TESTS > < SEQTESTS >) >

See the SCORR1 option.

PC

outputs Amemiya's prediction criterion for each model selected (Amemiya 1976; Judge et al. 1980) to the OUTEST= data set. If SELECTION=ADJRSQ, SELECTION=RSQUARE, or SELECTION=CP is specified, then the PC statistic is also added to the SubsetSelSummary table.

PCOMIT=*list*

requests an IPC analysis for each value *m* in the list. The procedure computes parameter estimates by using all but the last *m* principal components. Each value of *m* produces a set of IPC estimates, which is output to the OUTEST= data set. The values of *m* are saved by the variable _PCOMIT_, and the value of the variable _TYPE_ is set to IPC to identify the estimates. Only nonnegative integers can be specified with the PCOMIT= option.

If you specify the PCOMIT= option, RESTRICT statements are ignored. The PCOMIT= option is ignored if you use the SELECTION= option in the MODEL statement.

PCORR1

displays the squared partial correlation coefficients computed using Type I sum of squares (SS). This is calculated as SS/(SS+SSE), where SSE is the error sum of squares.

PCORR2

displays the squared partial correlation coefficients computed using Type II sums of squares. These are calculated the same way as with the PCORR1 option, except that Type II SS are used instead of Type I SS.

PRESS

outputs the PRESS statistic to the OUTEST= data set. The values of this statistic are saved in the variable _PRESS_. This option is available for all model-selection methods except RSQUARE, ADJRSQ, and CP.

R

requests an analysis of the residuals. The results include everything requested by the P option plus the standard errors of the mean predicted and residual values, the studentized residual, and Cook's *D* statistic to measure the influence of each observation on the parameter estimates. See the section "Predicted and Residual Values" on page 8801 for more information.

RIDGE=*list*

requests a ridge regression analysis and specifies the values of the ridge constant *k* (see the section "Computations for Ridge Regression and IPC Analysis" on page 8830). Each value of *k* produces a set of ridge regression estimates that are placed in the OUTEST= data set. The values of *k* are saved by the variable _RIDGE_, and the value of the variable _TYPE_ is set to RIDGE to identify the estimates.

Only nonnegative numbers can be specified with the RIDGE= option. Example 105.5 illustrates this option.

If you specify the RIDGE= option, RESTRICT statements are ignored. The RIDGE= option is ignored if you use the SELECTION= option in the MODEL statement.

RMSE

displays the root mean square error for each model selected.

RSQUARE

has the same effect as the EDF option.

SBC

outputs the SBC statistic for each model selected (Schwarz 1978; Judge et al. 1980) to the OUTEST= data set. If SELECTION=ADJRSQ, SELECTION=RSQUARE, or SELECTION=CP is specified, then the SBC statistic is also added to the SubsetSelSummary table.

SCORR1 < (< TESTS > < SEQTESTS >) >

displays the squared semipartial correlation coefficients computed using Type I sums of squares. This is calculated as SS/SST, where SST is the corrected total SS. If the NOINT option is used, the uncorrected total SS is used in the denominator. The optional arguments TESTS and SEQTESTS request are sequentially added to a model. The F-test values are computed as the Type I sum of squares for the variable in question divided by a mean square error. If you specify the TESTS option, the denominator MSE is the residual mean square for the full model specified in the MODEL statement. If you specify the SEQTESTS option, the denominator MSE is the residual mean square for the model containing all the independent variables that have been added to the model up to and including the variable in question. The TESTS and SEQTESTS options are not supported if you specify model selection methods or the RIDGE or PCOMIT options. Note that the PARTIALR2 option is a synonym for the SCORR1 option.

SCORR2 < (TESTS) >

displays the squared semipartial correlation coefficients computed using Type II sums of squares. These are calculated the same way as with the SCORR1 option, except that Type II SS are used instead of Type I SS. The optional TEST argument requests F tests and p-values as variables are sequentially added to a model. The F-test values are computed as the Type II sum of squares for the variable in question divided by the residual mean square for the full model specified in the MODEL statement. The TESTS option is not supported if you specify model selection methods or the RIDGE or PCOMIT options.

SELECTION=*name*

specifies the method used to select the model, where *name* can be FORWARD (or F), BACKWARD (or B), STEPWISE, MAXR, MINR, RSQUARE, ADJRSQ, CP, or NONE (use the full model). The default method is NONE. See the section "Model-Selection Methods" on page 8795 for a description of each method.

SEQB

produces a sequence of parameter estimates as each variable is entered into the model. This is displayed as a matrix where each row is a set of parameter estimates.

SIGMA=*n*

specifies the true standard deviation of the error term to be used in computing the CP and BIC statistics. If the SIGMA= option is not specified, an estimate from the full model is used. This option is available in the RSQUARE, ADJRSQ, and CP model-selection methods only.

SINGULAR=*n*

tunes the mechanism used to check for singularities. If you specify this option in the MODEL statement, it takes precedence over the SINGULAR= option in the PROC REG statement. The default value is machine dependent but is approximately 1E–9 on most machines. This option is rarely needed. Singularity checking is described in the section "Computational Methods" on page 8831.

SLENTRY=*value*

SLE=*value*

specifies the significance level for entry into the model used in the FORWARD and STEPWISE methods. The defaults are 0.50 for FORWARD and 0.15 for STEPWISE.

SLSTAY=*value*

SLS=*value*

specifies the significance level for staying in the model for the BACKWARD and STEPWISE methods. The defaults are 0.10 for BACKWARD and 0.15 for STEPWISE.

SP

outputs the S_p statistic for each model selected (Hocking 1976) to the OUTEST= data set. If SELECTION=ADJRSQ, SELECTION=RSQUARE, or SELECTION=CP is specified, then the SP statistic is also added to the SubsetSelSummary table.

SPEC

performs a test that the first and second moments of the model are correctly specified. See the section "Testing for Heteroscedasticity" on page 8824 for more information.

SRT

displays a column in the "Output Statistics" table that shows the magnitude of studentized residuals by using a series of asterisks. This table is produced by the R option in the MODEL statement. When ODS Graphics is not enabled, the column is displayed in the table and the graph is not produced. When ODS Graphics is enabled, a graphical chart of the studentized residuals and Cook's D is produced, and the column is omitted from the table unless you specify the SRT option.

SS1

displays the sequential sums of squares (Type I SS) along with the parameter estimates for each term in the model. See Chapter 16, "The Four Types of Estimable Functions," for more information about the different types of sums of squares.

SS2

displays the partial sums of squares (Type II SS) along with the parameter estimates for each term in the model. See the SS1 option also.

SSE

computes the error sum of squares for each model selected.

START=*s*

specifies the number of regressors to be added or reported in a model.

When SELECTION= MAXR, MINR, or STEPWISE, START=*s* requests that the first *s* regressors be added to the model before beginning the comparing-and-switching process. Thus, START=*s* behaves like INCLUDE=*s* in that it forces selection to begin with the first *s* regressors included in the model, but it differs in that START=*s* allows these variables to be subsequently removed from the model in the course of variable selection. For these methods, the default is START=0.

When SELECTION= RSQUARE, ADJRSQ, or CP, START=*s* specifies the smallest number of regressors to be reported in a subset model. For these methods, the default is START=1.

The START= option cannot be used with model-selection methods other than the six described here.

STB

produces standardized regression coefficients. A standardized regression coefficient is computed by dividing a parameter estimate by the ratio of the sample standard deviation of the dependent variable to the sample standard deviation of the regressor.

STOP=*s*

causes PROC REG to stop when it has found the "best" *s*-variable model, where *s* is the STOP value. For the RSQUARE, ADJRSQ, and CP methods, STOP=*s* specifies the largest number of regressors to be reported in a subset model. For the MAXR and MINR methods, STOP=*s* specifies the largest number of regressors to be included in the model.

The default setting for the STOP= option is the number of variables in the MODEL statement. This option can be used only with the MAXR, MINR, RSQUARE, ADJRSQ, and CP methods.

TOL

produces tolerance values for the estimates. Tolerance for a variable is defined as $1 - R^2$, where R square is obtained from the regression of the variable on all other regressors in the model. See the section "Collinearity Diagnostics" on page 8806 for more details.

VIF

produces variance inflation factors with the parameter estimates. Variance inflation is the reciprocal of tolerance. See the section "Collinearity Diagnostics" on page 8806 for more detail.

WHITE

See the HCC option.

XPX

displays the $\mathbf{X}'\mathbf{X}$ crossproducts matrix for the model. The crossproducts matrix is bordered by the $\mathbf{X}'\mathbf{Y}$ and $\mathbf{Y}'\mathbf{Y}$ matrices.

MTEST Statement

< *label*: > **MTEST** < *equation* < , . . . , *equation* > > < / *options* > ;

where each *equation* is a linear function composed of coefficients and variable names. The *label* is optional.

The MTEST statement is used to test hypotheses in multivariate regression models where there are several dependent variables fit to the same regressors. If no equations or options are specified, the MTEST statement tests the hypothesis that all estimated parameters except the intercept are zero.

The hypotheses that can be tested with the MTEST statement are of the form

$$(\mathbf{L}\boldsymbol{\beta} - \mathbf{cj})\mathbf{M} = 0$$

where $\mathbf{L}$ is a linear function on the regressor side, $\boldsymbol{\beta}$ is a matrix of parameters, $\mathbf{c}$ is a column vector of constants, $\mathbf{j}$ is a row vector of ones, and $\mathbf{M}$ is a linear function on the dependent side. The special case where the constants are zero is

$$\mathbf{L}\boldsymbol{\beta}\mathbf{M} = \mathbf{0}$$

See the section "Multivariate Tests" on page 8826 for more details.

Each linear function extends across either the regressor variables or the dependent variables. If the equation is across the dependent variables, then the constant term, if specified, must be zero. The equations for the regressor variables form the **L** matrix and **c** vector in the preceding formula; the equations for dependent variables form the **M** matrix. If no equations for the dependent variables are given, PROC REG uses an identity matrix for **M**, testing the same hypothesis across all dependent variables. If no equations for the regressor variables are given, PROC REG forms a linear function corresponding to a test that all the nonintercept parameters are zero.

As an example, consider the following statements:

```
model y1 y2 y3=x1 x2 x3;
mtest x1,x2;
mtest y1-y2, y2 -y3, x1;
mtest y1-y2;
```

The first MTEST statement tests the hypothesis that the X1 and X2 parameters are zero for Y1, Y2, and Y3. In addition, the second MTEST statement tests the hypothesis that the X1 parameter is the same for all three dependent variables. For the same model, the third MTEST statement tests the hypothesis that all parameters except the intercept are the same for dependent variables Y1 and Y2.

You can specify the following *options* in the MTEST statement:

CANPRINT

displays the canonical correlations for the hypothesis combinations and the dependent variable combinations. If you specify

```
mtest / canprint;
```

the canonical correlations between the regressors and the dependent variables are displayed.

DETAILS

displays the **M** matrix and various intermediate calculations.

MSTAT=FAPPROX | EXACT

specifies the method of evaluating the multivariate test statistics. The default is MSTAT=FAPPROX, which specifies that the multivariate tests are evaluated by using the usual approximations based on the F distribution, as discussed in the "Multivariate Tests" section in Chapter 4, "Introduction to Regression Procedures." Alternatively, you can specify MSTAT=EXACT to compute exact p-values for three of the four tests (Wilks' lambda, the Hotelling-Lawley trace, and Roy's greatest root) and an improved F approximation for the fourth (Pillai's trace). While MSTAT=EXACT provides better control of the significance probability for the tests, especially for Roy's greatest root, computations for the exact p-values can be appreciably more demanding, and are in fact infeasible for large problems (many dependent variables). Thus, although MSTAT=EXACT is more accurate for most data, it is not the default method.

PRINT
displays the **H** and **E** matrices.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set*> < *keyword*=*names*> < . . . *keyword*=*names*> **;**

The OUTPUT statement creates a new SAS data set that saves diagnostic measures calculated after fitting the model. The OUTPUT statement refers to the most recent MODEL statement. At least one *keyword=names* specification is required.

All the variables in the original data set are included in the new data set, along with variables created in the OUTPUT statement. These new variables contain the values of a variety of statistics and diagnostic measures that are calculated for each observation in the data set. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

The OUTPUT statement cannot be used when a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set is used as the input data set for PROC REG. See the section "Input Data Sets" on page 8782 for more details.

The statistics created in the OUTPUT statement are described in this section. More details are given in the section "Predicted and Residual Values" on page 8801 and the section "Influence Statistics" on page 8810. Also see Chapter 4, "Introduction to Regression Procedures," for definitions of the statistics available from the REG procedure.

You can specify the following *options* in the OUTPUT statement:

OUT=*SAS data set*
gives the name of the new data set. By default, the procedure uses the DATA*n* convention to name the new data set.

keyword=names
specifies the statistics to include in the output data set and names the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), an equal sign, and the variable or variables to contain the statistic.

In the output data set, the first variable listed after a *keyword* in the OUTPUT statement contains that statistic for the first dependent variable listed in the MODEL statement; the second variable contains the statistic for the second dependent variable in the MODEL statement, and so on. The list of variables following the equal sign can be shorter than the list of dependent variables in the MODEL statement. In this case, the procedure creates the new names in order of the dependent variables in the MODEL statement.

For example, the following SAS statements create an output data set named b:

```
proc reg data=a;
   model y z=x1 x2;
   output out=b
      p=yhat zhat
      r=yresid zresid;
run;
```

In addition to the variables in the input data set, b contains the following variables:

- yhat, with values that are predicted values of the dependent variable y
- zhat, with values that are predicted values of the dependent variable z
- yresid, with values that are the residual values of y
- zresid, with values that are the residual values of z

You can specify the following *keywords* in the OUTPUT statement. See the section "Model Fit and Diagnostic Statistics" on page 8808 for computational formulas.

Table 105.6 Keywords for OUTPUT Statement

Keyword	Description
COOKD=*names*	Cook's D influence statistic
COVRATIO=*names*	Standard influence of observation on covariance of betas, as discussed in the section "Influence Statistics" on page 8810
DFFITS=*names*	Standard influence of observation on predicted value
H=*names*	Leverage, $\mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_i'$
LCL=*names*	Lower bound of a $100(1-\alpha)\%$ confidence interval for an individual prediction. This includes the variance of the error, as well as the variance of the parameter estimates.
LCLM=*names*	Lower bound of a $100(1-\alpha)\%$ confidence interval for the expected value (mean) of the dependent variable
PREDICTED | **P=***names*	Predicted values
PRESS=*names*	ith residual divided by $(1-h)$, where h is the leverage, and where the model has been refit without the ith observation
RESIDUAL | **R=***names*	Residuals, calculated as ACTUAL minus PREDICTED
RSTUDENT=*names*	A studentized residual with the current observation deleted
STDI=*names*	Standard error of the individual predicted value
STDP=*names*	Standard error of the mean predicted value
STDR=*names*	Standard error of the residual
STUDENT=*names*	Studentized residuals, which are the residuals divided by their standard errors
UCL=*names*	Upper bound of a $100(1-\alpha)\%$ confidence interval for an individual prediction
UCLM=*names*	Upper bound of a $100(1-\alpha)\%$ confidence interval for the expected value (mean) of the dependent variable

PRINT Statement

PRINT < *options* > < **ANOVA** > < **MODELDATA** > **;**

The PRINT statement enables you to interactively display the results of MODEL statement options, produce an ANOVA table, display the data for variables used in the current model, or redisplay the options specified in a MODEL or a previous PRINT statement. In addition, like most other interactive statements in PROC REG, the PRINT statement implicitly refits the model; thus, effects of REWEIGHT statements are seen in the resulting tables. If ODS Graphics is enabled (see the section "ODS Graphics" on page 8852), the PRINT statement also requests the use of the ODS graphical displays associated with the current model.

The following specifications can appear in the PRINT statement:

options
: interactively displays the results of MODEL statement options, where *options* is one or more of the following: ACOV, ALL, CLI, CLM, COLLIN, COLLINOINT, CORRB, COVB, DW, I, INFLUENCE, P, PARTIAL, PCORR1, PCORR2, R, SCORR1, SCORR2, SEQB, SPEC, SS1, SS2, STB, TOL, VIF, or XPX. See the section "MODEL Statement" on page 8759 for a description of these *options*.

ANOVA
: produces the ANOVA table associated with the current model. This is either the model specified in the last MODEL statement or the model that incorporates changes made by ADD, DELETE, or REWEIGHT statements after the last MODEL statement.

MODELDATA
: displays the data for variables used in the current model.

Use the statement

```
print;
```

to reprint options in the most recently specified PRINT or MODEL statement.

Options that require original data values, such as R or INFLUENCE, cannot be used when a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set is used as the input data set to PROC REG. See the section "Input Data Sets" on page 8782 for more detail.

REFIT Statement

REFIT ;

The REFIT statement causes the current model and corresponding statistics to be recomputed immediately. No output is generated by this statement. The REFIT statement is needed after one or more REWEIGHT statements to cause them to take effect before subsequent PAINT or REWEIGHT statements. This is sometimes necessary when you are using statistical conditions in REWEIGHT statements. For example, consider the following statements:

```
paint student.>2;
plot student.*p.;
reweight student.>2;
refit;
paint student.>2;
plot student.*p.;
```

The second PAINT statement paints any additional observations that meet the condition after deleting observations and refitting the model. The REFIT statement is used because the REWEIGHT statement does not cause the model to be recomputed. In this particular example, the same effect could be achieved by replacing the REFIT statement with a PLOT statement.

Most interactive statements can be used to implicitly refit the model; any plots or statistics produced by these statements reflect changes made to the model and changes made to the data used to compute the model. The two exceptions are the PAINT and REWEIGHT statements, which do not cause the model to be recomputed.

RESTRICT Statement

RESTRICT *equation* <**,** . . . **,** *equation*> **;**

A RESTRICT statement is used to place restrictions on the parameter estimates in the MODEL preceding it. More than one RESTRICT statement can follow each MODEL statement. Each RESTRICT statement replaces any previous RESTRICT statement. To lift all restrictions on a model, submit a new MODEL statement. If there are several restrictions, separate them with commas. The statement

```
restrict equation1=equation2=equation3;
```

is equivalent to imposing the two restrictions

```
restrict equation1=equation2;
restrict equation2=equation3;
```

Each restriction is written as a linear equation and can be written as

equation

or

equation = *equation*

The form of each *equation* is

$$c_1 \times variable_1 \pm c_2 \times variable_2 \pm \cdots \pm c_n \times variable_n$$

where the c_j's are constants and the $variable_j$'s are any regressor variables.

When no equal sign appears, the linear combination is set equal to zero. Each variable name mentioned must be a variable in the MODEL statement to which the RESTRICT statement refers. The keyword INTERCEPT can also be used as a variable name, and it refers to the intercept parameter in the regression model.

Note that the parameters associated with the variables are restricted, not the variables themselves. Restrictions should be consistent and not redundant.

Examples of valid RESTRICT statements include the following:

```
restrict x1;
restrict a+b=1;
restrict a=b=c;
restrict a=b, b=c;
restrict 2*f=g+h, intercept+f=0;
restrict f=g=h=intercept;
```

The third and fourth statements in this list produce identical restrictions. You cannot specify

```
restrict f-g=0,
         f-intercept=0,
         g-intercept=1;
```

because the three restrictions are not consistent. If these restrictions are included in a RESTRICT statement, one of the restrict parameters is set to zero and has zero degrees of freedom, indicating that PROC REG is unable to apply a restriction.

The restrictions usually operate even if the model is not of full rank. Check to ensure that DF = –1 for each restriction. In addition, the model DF should decrease by 1 for each restriction.

The parameter estimates are those that minimize the quadratic criterion (SSE) subject to the restrictions. If a restriction cannot be applied, its parameter value and degrees of freedom are listed as zero.

The method used for restricting the parameter estimates is to introduce a Lagrangian parameter for each restriction (Pringle and Rayner 1971). The estimates of these parameters are displayed with test statistics. Note that the t statistic reported for the Lagrangian parameters does not follow a Student's t distribution, but its square follows a beta distribution (LaMotte 1994). The p-value for these parameters is computed using the beta distribution.

The Lagrangian parameter γ measures the sensitivity of the SSE to the restriction constant. If the restriction constant is changed by a small amount ϵ, the SSE is changed by $2\gamma\epsilon$. The t ratio tests the significance of the restrictions. If γ is zero, the restricted estimates are the same as the unrestricted estimates, and a change in the restriction constant in either direction increases the SSE.

RESTRICT statements are ignored if the PCOMIT= or RIDGE= option is specified in the PROC REG statement.

REWEIGHT Statement

REWEIGHT < *condition* | **ALLOBS** > < / *options* > **;**

REWEIGHT < **STATUS** | **UNDO** > **;**

The REWEIGHT statement interactively changes the weights of observations that are used in computing the regression equation. The REWEIGHT statement can change observation weights, or set them to zero, which causes selected observations to be excluded from the analysis. When a REWEIGHT statement sets observation weights to zero, the observations are not deleted from the data set. More than one REWEIGHT statement can be used. The requests from all REWEIGHT statements are applied to the subsequent statements. Each use of the REWEIGHT statement modifies the MODEL label.

The model and corresponding statistics are not recomputed after a REWEIGHT statement. For example, consider the following statements:

```
reweight r.>0;
reweight r.>0;
```

The second REWEIGHT statement does not exclude any additional observations since the model is not recomputed after the first REWEIGHT statement. Either use a REFIT statement to explicitly refit the model, or implicitly refit the model by following the REWEIGHT statement with any other interactive statement except a PAINT statement or another REWEIGHT statement.

The REWEIGHT statement cannot be used if a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set is used as an input data set to PROC REG. Note that the syntax used in the REWEIGHT statement is the same as the syntax in the PAINT statement.

The syntax of the REWEIGHT statement is described in the following sections.

For detailed examples of using this statement, see the section "Reweighting Observations in an Analysis" on page 8818.

Specifying Condition

Condition is used to find observations to be reweighted. The syntax of *condition* is

variable compare value

or

variable compare value *logical* *variable compare value*

where

variable is one of the following:

- a variable name in the input data set
- OBS**.**, which is the observation number
- *keyword***.**, where *keyword* is a keyword for a statistic requested in the OUTPUT statement. The *keyword* specification is applied to all dependent variables in the model.

compare is an operator that compares *variable* to *value*. *Compare* can be any one of the following: <, <=, >, >=, =, ˆ =. The operators LT, LE, GT, GE, EQ, and NE, respectively, can be used instead of the preceding symbols. See the "Expressions" chapter in *SAS Programmers Guide: Essentials* for more information about comparison operators.

value gives an unformatted value of *variable*. Observations are selected to be reweighted if they satisfy the condition created by *variable compare value*. *Value* can be a number or a character string. If *value* is a character string, it must be eight characters or less and must be enclosed in quotes. In addition, *value* is case-sensitive. In other words, the following two statements are not the same:

```
reweight name='steve';

reweight name='Steve';
```

logical is one of two logical operators. Either AND or OR can be used. To specify AND, use AND or the symbol &. To specify OR, use OR or the symbol |.

Here are some examples of the *variable compare value* form:

```
reweight obs. le 10;
reweight temp=55;
reweight type='new';
```

Here are some examples of the *variable compare value logical variable compare value* form:

```
reweight obs.<=10 and residual.<2;
reweight student.<-2 or student.>2;
reweight name='Mary' | name='Susan';
```

Using ALLOBS

Instead of specifying *condition*, you can use the ALLOBS option to select all observations. This is most useful when you want to restore the original weights of all observations. For example,

```
reweight allobs / reset;
```

resets weights for all observations and uses all observations in the subsequent analysis. Note that

```
reweight allobs;
```

specifies that all observations be excluded from analysis. Consequently, using ALLOBS is useful only if you also use one of the options discussed in the following section.

Options in the REWEIGHT Statement

The following *options* can be used when either a condition, ALLOBS, or nothing is specified before the slash. If only an *option* is listed, the *option* applies to the observations selected in the previous REWEIGHT statement, not to the observations selected by reapplying the condition from the previous REWEIGHT statement. For example, consider the following statements:

```
reweight r.>0 / weight=0.1;
refit;
reweight;
```

The second REWEIGHT statement excludes from the analysis only those observations selected in the first REWEIGHT statement. No additional observations are excluded even if there are new observations that meet the condition in the first REWEIGHT statement.

NOTE: *Options* are not available when either the UNDO or STATUS option is used.

NOLIST

suppresses the display of the selected observation numbers. If you omit the NOLIST option, a list of observations selected is written to the log.

RESET

resets the observation weights to their original values as defined by the WEIGHT statement or to WEIGHT=1 if no WEIGHT statement is specified. For example,

```
reweight / reset;
```

resets observation weights to the original weights in the data set. If previous REWEIGHT statements have been submitted, this REWEIGHT statement applies only to the observations selected by the previous REWEIGHT statement. Note that, although the RESET option does reset observation weights to their original values, it does not cause the model and corresponding statistics to be recomputed.

WEIGHT=*value*

changes observation weights to the specified nonnegative real number. If you omit the WEIGHT= option, the observation weights are set to zero, and observations are excluded from the analysis. For example:

```
reweight name='Alan';
...other interactive statements
reweight / weight=0.5;
```

The first REWEIGHT statement changes weights to zero for all observations with name='Alan', effectively deleting these observations. The subsequent analysis does not include these observations. The second REWEIGHT statement applies only to those observations selected by the previous REWEIGHT statement, and it changes the weights to 0.5 for all the observations with NAME='Alan'. Thus, the next analysis includes all original observations; however, those observations with NAME='Alan' have their weights set to 0.5.

STATUS and UNDO

If you omit *condition* and the ALLOBS options, you can specify one of the following *options*.

STATUS

writes to the log the observation's number and the weight of all reweighted observations. If an observation's weight has been set to zero, it is reported as deleted. However, the observation is not deleted from the data set, only from the analysis.

UNDO

undoes the changes made by the most recent REWEIGHT statement. Weights might be, but are not necessarily, reset. For example, consider the following statements:

```
reweight student.>2 / weight=0.1;
reweight;
reweight undo;
```

The first REWEIGHT statement sets the weights of observations that satisfy the condition to 0.1. The second REWEIGHT statement sets the weights of the same observations to zero. The third REWEIGHT statement undoes the second, changing the weights back to 0.1.

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement requests that the procedure save the estimated parameters of the fitted model. The resulting item store is a binary file format that cannot be modified. The contents of the item store can be processed with the PLM procedure.

For details about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

NOTE: The information stored by the STORE statement in PROC REG is a subset of what is usually stored by other procedures that implement this statement. In particular, PROC REG stores only the estimated parameters of the model, so that you can later use the CODE statement in PROC PLM to write SAS DATA step code for prediction to a file or catalog entry. With only this subset of information, many other postprocessing features of PROC PLM are not available for item stores that are created by PROC REG.

TEST Statement

< *label*: > **TEST** *equation*, < , . . . , *equation* > < / *option* > **;**

The TEST statement tests hypotheses about the parameters estimated in the preceding MODEL statement. It has the same syntax as the RESTRICT statement except that it supports an *option*. Each *equation* specifies a linear hypothesis to be tested. The rows of the hypothesis are separated by commas.

Variable names must correspond to regressors, and each variable name represents the coefficient of the corresponding variable in the model. An optional label is useful to identify each test with a name. The keyword INTERCEPT can be used instead of a variable name to refer to the model's intercept.

The REG procedure performs an F test for the joint hypotheses specified in a single TEST statement. More than one TEST statement can accompany a MODEL statement. The numerator is the usual quadratic form of the estimates; the denominator is the mean squared error. If hypotheses can be represented by

$$\mathbf{L}\boldsymbol{\beta} = \mathbf{c}$$

then the numerator of the F test is

$$\mathbf{Q} = (\mathbf{Lb} - \mathbf{c})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{Lb} - \mathbf{c})$$

divided by degrees of freedom, where $\mathbf{b}$ is the estimate of $\boldsymbol{\beta}$. For example:

```
model y=a1 a2 b1 b2;
aplus: test a1+a2=1;
b1:    test b1=0, b2=0;
b2:    test b1, b2;
```

The last two statements are equivalent; since no constant is specified, zero is assumed.

Note that, when the ACOV, HCC, or WHITE option is specified in the MODEL statement, tests are recomputed using the heteroscedasticity-consistent covariance matrix specified with the HCCMETHOD= option in the MODEL statement (see the section "Testing for Heteroscedasticity" on page 8824).

One *option* can be specified in the TEST statement after a slash (/):

PRINT

displays intermediate calculations. This includes $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}'$ bordered by $\mathbf{Lb} - \mathbf{c}$, and $(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}$ bordered by $(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{Lb} - \mathbf{c})$.

VAR Statement

VAR *variables* **;**

The VAR statement is used to include numeric variables in the crossproducts matrix that are not specified in the first MODEL statement.

Variables not listed in MODEL statements before the first RUN statement must be listed in the VAR statement if you want the ability to add them interactively to the model with an ADD statement, to include them in a new MODEL statement, or to plot them in a scatter plot with the PLOT statement.

In addition, if you want to use options in the PROC REG statement and do not want to fit a model to the data (with a MODEL statement), you must use a VAR statement.

WEIGHT Statement

WEIGHT *variable* **;**

A WEIGHT statement names a *variable* in the input data set with values that are relative weights for a weighted least squares fit. If the weight value is proportional to the reciprocal of the variance for each observation, then the weighted estimates are the best linear unbiased estimates (BLUE).

Values of the weight *variable* must be nonnegative. If an observation's weight is zero, the observation is deleted from the analysis. If a weight is negative or missing, it is set to zero, and the observation is excluded from the analysis. A more complete description of the WEIGHT statement can be found in Chapter 53, "The GLM Procedure."

Observation weights can be changed interactively with the REWEIGHT statement.

Details: REG Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/regdt.sas`.)

Missing Values

PROC REG constructs only one crossproducts matrix for the variables in all regressions. If any variable needed for any regression is missing, the observation is excluded from all estimates. If you include variables with missing values in the VAR statement, the corresponding observations are excluded from all analyses, even if you never include the variables in a model. PROC REG assumes that you might want to include these variables after the first RUN statement and deletes observations with missing values.

Input Data Sets

PROC REG does not compute new regressors. For example, if you want a quadratic term in your model, you should create a new variable when you prepare the input data. For example, the statement

```
model y=x1 x1*x1;
```

is not valid. Note that this MODEL statement is valid in the GLM procedure.

The input data set for most applications of PROC REG contains standard rectangular data, but special TYPE=CORR, TYPE=COV, and TYPE=SSCP data sets can also be used. TYPE=CORR and TYPE=COV data sets created by the CORR procedure contain means and standard deviations. In addition, TYPE=CORR data sets contain correlations and TYPE=COV data sets contain covariances. TYPE=SSCP data sets created in previous runs of PROC REG that used the OUTSSCP= option contain the sums of squares and crossproducts of the variables.

See Appendix A, "Special SAS Data Sets," and the "SAS Files" section in *SAS Programmers Guide: Essentials* for more information about special SAS data sets.

These summary files save CPU time. It takes nk^2 operations (where n = number of observations and k = number of variables) to calculate crossproducts; the regressions are of the order k^3. When n is in the thousands and k is less than 10, you can save 99% of the CPU time by reusing the SSCP matrix rather than recomputing it.

When you want to use a special SAS data set as input, PROC REG must determine the TYPE for the data set. PROC CORR and PROC REG automatically set the type for their output data sets. However, if you create the data set by some other means (such as a DATA step), you must specify its type with the TYPE= data set option. If the TYPE for the data set is not specified when the data set is created, you can specify TYPE= as a data set option in the DATA= option in the PROC REG statement. For example:

```
proc reg data=a(type=corr);
```

When a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set is used with PROC REG, statements and options that require the original data values have no effect. The OUTPUT, PAINT, PLOT, and REWEIGHT statements and the MODEL and PRINT statement options P, R, CLM, CLI, DW, INFLUENCE, and PARTIAL are disabled since the original observations needed to calculate predicted and residual values are not present.

Example Using TYPE=CORR Data Set

The following statements use PROC CORR to produce an input data set for PROC REG. The fitness data for this analysis can be found in Example 105.2.

```
proc corr data=fitness outp=r noprint;
   var Oxygen RunTime Age Weight RunPulse MaxPulse RestPulse;
run;
proc print data=r;
run;
proc reg data=r;
   model Oxygen=RunTime Age Weight;
run;
```

Since the OUTP= data set from PROC CORR is automatically set to TYPE=CORR, the TYPE= data set option is not required in this example. The data set containing the correlation matrix is displayed by the PRINT procedure as shown in Figure 105.15. Figure 105.16 shows results from the regression that uses the TYPE=CORR data as an input data set.

Figure 105.15 TYPE=CORR Data Set Created by PROC CORR

Obs	_TYPE_	_NAME_	Oxygen	RunTime	Age	Weight	RunPulse	MaxPulse	RestPulse
1	MEAN		47.3758	10.5861	47.6774	77.4445	169.645	173.774	53.4516
2	STD		5.3272	1.3874	5.2114	8.3286	10.252	9.164	7.6194
3	N		31.0000	31.0000	31.0000	31.0000	31.000	31.000	31.0000
4	CORR	Oxygen	1.0000	-0.8622	-0.3046	-0.1628	-0.398	-0.237	-0.3994
5	CORR	RunTime	-0.8622	1.0000	0.1887	0.1435	0.314	0.226	0.4504
6	CORR	Age	-0.3046	0.1887	1.0000	-0.2335	-0.338	-0.433	-0.1641
7	CORR	Weight	-0.1628	0.1435	-0.2335	1.0000	0.182	0.249	0.0440
8	CORR	RunPulse	-0.3980	0.3136	-0.3379	0.1815	1.000	0.930	0.3525
9	CORR	MaxPulse	-0.2367	0.2261	-0.4329	0.2494	0.930	1.000	0.3051
10	CORR	RestPulse	-0.3994	0.4504	-0.1641	0.0440	0.352	0.305	1.0000

Figure 105.16 Regression on TYPE=CORR Data Set

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	656.27095	218.75698	30.27	<.0001
Error	27	195.11060	7.22632		
Corrected Total	30	851.38154			

Root MSE	2.68818	R-Square	0.7708
Dependent Mean	47.37581	Adj R-Sq	0.7454
Coeff Var	5.67416		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	93.12615	7.55916	12.32	<.0001
RunTime	1	-3.14039	0.36738	-8.55	<.0001
Age	1	-0.17388	0.09955	-1.75	0.0921
Weight	1	-0.05444	0.06181	-0.88	0.3862

The following example uses the saved crossproducts matrix:

```
proc reg data=fitness outsscp=sscp noprint;
   model Oxygen=RunTime Age Weight RunPulse MaxPulse RestPulse;
run;
proc print data=sscp;
run;
proc reg data=sscp;
   model Oxygen=RunTime Age Weight;
run;
```

First, all variables are used to fit the data and create the SSCP data set. Figure 105.17 shows the PROC PRINT display of the SSCP data set. The SSCP data set is then used as the input data set for PROC REG, and a reduced model is fit to the data.

Figure 105.17 TYPE=SSCP Data Set Created by PROC REG

Obs	_TYPE_	_NAME_	Intercept	RunTime	Age	Weight	RunPulse	MaxPulse	RestPulse	Oxygen
1	SSCP	Intercept	31.00	328.17	1478.00	2400.78	5259.00	5387.00	1657.00	1468.65
2	SSCP	RunTime	328.17	3531.80	15687.24	25464.71	55806.29	57113.72	17684.05	15356.14
3	SSCP	Age	1478.00	15687.24	71282.00	114158.90	250194.00	256218.00	78806.00	69767.75
4	SSCP	Weight	2400.78	25464.71	114158.90	188008.20	407745.67	417764.62	128409.28	113522.26
5	SSCP	RunPulse	5259.00	55806.29	250194.00	407745.67	895317.00	916499.00	281928.00	248497.31
6	SSCP	MaxPulse	5387.00	57113.72	256218.00	417764.62	916499.00	938641.00	288583.00	254866.75
7	SSCP	RestPulse	1657.00	17684.05	78806.00	128409.28	281928.00	288583.00	90311.00	78015.41
8	SSCP	Oxygen	1468.65	15356.14	69767.75	113522.26	248497.31	254866.75	78015.41	70429.86
9	N		31.00	31.00	31.00	31.00	31.00	31.00	31.00	31.00

Figure 105.18 also shows the PROC REG results for the reduced model. (For the PROC REG results for the full model, see Figure 105.30.)

Figure 105.18 Regression on TYPE=SSCP Data Set

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	656.27095	218.75698	30.27	<.0001
Error	27	195.11060	7.22632		
Corrected Total	30	851.38154			

Root MSE	2.68818	**R-Square**	0.7708
Dependent Mean	47.37581	**Adj R-Sq**	0.7454
Coeff Var	5.67416		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	93.12615	7.55916	12.32	<.0001
RunTime	1	-3.14039	0.36738	-8.55	<.0001
Age	1	-0.17388	0.09955	-1.75	0.0921
Weight	1	-0.05444	0.06181	-0.88	0.3862

In the preceding example, the TYPE= data set option is not required since PROC REG sets the OUTSSCP= data set to TYPE=SSCP.

Output Data Sets

OUTEST= Data Set

The OUTEST= specification produces a TYPE=EST output SAS data set containing estimates and optional statistics from the regression models. For each BY group on each dependent variable occurring in each MODEL statement, PROC REG outputs an observation to the OUTEST= data set. The variables output to the data set are as follows:

- the BY variables, if any
- _MODEL_, a character variable containing the label of the corresponding MODEL statement, or MODEL*n* if no label is specified, where *n* is 1 for the first MODEL statement, 2 for the second model statement, and so on
- _TYPE_, a character variable with the value 'PARMS' for every observation
- _DEPVAR_, the name of the dependent variable

- _RMSE_, the root mean squared error or the estimate of the standard deviation of the error term
- Intercept, the estimated intercept, unless the NOINT option is specified
- all the variables listed in any MODEL or VAR statement. Values of these variables are the estimated regression coefficients for the model. A variable that does not appear in the model corresponding to a given observation has a missing value in that observation. The dependent variable in each model is given a value of –1.

If you specify the COVOUT option, the covariance matrix of the estimates is output after the estimates; the _TYPE_ variable is set to the value 'COV' and the names of the rows are identified by the character variable, _NAME_.

If you specify the TABLEOUT option, the following statistics listed by _TYPE_ are added after the estimates:

- STDERR, the standard error of the estimate
- T, the t statistic for testing if the estimate is zero
- PVALUE, the associated p-value
- LnB, the $100(1-\alpha)$ lower confidence limit for the estimate, where n is the nearest integer to $100(1-\alpha)$ and α defaults to 0.05 or is set by using the ALPHA= option in the PROC REG or MODEL statement
- UnB, the $100(1-\alpha)$ upper confidence limit for the estimate

Specifying the option ADJRSQ, AIC, BIC, CP, EDF, GMSEP, JP, MSE, PC, RSQUARE, SBC, SP, or SSE in the PROC REG or MODEL statement automatically outputs these statistics and the model R^2 for each model selected, regardless of the model selection method. Additional variables, in order of occurrence, are as follows:

- _IN_, the number of regressors in the model not including the intercept
- _P_, the number of parameters in the model including the intercept, if any
- _EDF_, the error degrees of freedom
- _SSE_, the error sum of squares, if the SSE option is specified
- _MSE_, the mean squared error, if the MSE option is specified
- _RSQ_, the R square statistic
- _ADJRSQ_, the adjusted R square, if the ADJRSQ option is specified
- _CP_, the C_p statistic, if the CP option is specified
- _SP_, the S_p statistic, if the SP option is specified
- _JP_, the J_p statistic, if the JP option is specified
- _PC_, the PC statistic, if the PC option is specified

- _GMSEP_, the GMSEP statistic, if the GMSEP option is specified
- _AIC_, the AIC statistic, if the AIC option is specified
- _BIC_, the BIC statistic, if the BIC option is specified
- _SBC_, the SBC statistic, if the SBC option is specified

The following statements produce and display the OUTEST= data set. This example uses the population data given in the section "Polynomial Regression" on page 8728. Figure 105.19 through Figure 105.21 show the regression equations and the resulting OUTEST= data set.

```
proc reg data=USPopulation outest=est;
   m1: model Population=Year;
   m2: model Population=Year YearSq;
run;
proc print data=est;
run;
```

Figure 105.19 Regression Output for Model M1

The REG Procedure
Model: m1
Dependent Variable: Population

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	146869	146869	228.92	<.0001
Error	20	12832	641.58160		
Corrected Total	21	159700			

Root MSE	25.32946	**R-Square**	0.9197
Dependent Mean	94.64800	**Adj R-Sq**	0.9156
Coeff Var	26.76175		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-2345.85498	161.39279	-14.54	<.0001
Year	1	1.28786	0.08512	15.13	<.0001

Figure 105.20 Regression Output for Model M2

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	159529	79765	8864.19	<.0001
Error	19	170.97193	8.99852		
Corrected Total	21	159700			

Root MSE	2.99975	R-Square	0.9989
Dependent Mean	94.64800	Adj R-Sq	0.9988
Coeff Var	3.16938		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	21631	639.50181	33.82	<.0001
Year	1	-24.04581	0.67547	-35.60	<.0001
YearSq	1	0.00668	0.00017820	37.51	<.0001

Figure 105.21 OUTEST= Data Set

Obs	_MODEL_	_TYPE_	_DEPVAR_	_RMSE_	Intercept	Year	Population	YearSq
1	m1	PARMS	Population	25.3295	-2345.85	1.2879	-1	.
2	m2	PARMS	Population	2.9998	21630.89	-24.0458	-1	.006684346

The following modification of the previous example uses the TABLEOUT and ALPHA= options to obtain additional information in the OUTEST= data set:

```
proc reg data=USPopulation outest=est tableout alpha=0.1;
   m1: model Population=Year/noprint;
   m2: model Population=Year YearSq/noprint;
run;
proc print data=est;
run;
```

Notice that the TABLEOUT option causes standard errors, t statistics, p-values, and confidence limits for the estimates to be added to the OUTEST= data set. Also note that the ALPHA= option is used to set the confidence level at 90%. The OUTEST= data set is shown in Figure 105.22.

Figure 105.22 The OUTEST= Data Set When TABLEOUT Is Specified

Obs	_MODEL_	_TYPE_	_DEPVAR_	_RMSE_	Intercept	Year	Population	YearSq
1	m1	PARMS	Population	25.3295	-2345.85	1.2879	-1	.
2	m1	STDERR	Population	25.3295	161.39	0.0851	.	.
3	m1	T	Population	25.3295	-14.54	15.1300	.	.
4	m1	PVALUE	Population	25.3295	0.00	0.0000	.	.
5	m1	L90B	Population	25.3295	-2624.21	1.1411	.	.
6	m1	U90B	Population	25.3295	-2067.50	1.4347	.	.
7	m2	PARMS	Population	2.9998	21630.89	-24.0458	-1	0.0067
8	m2	STDERR	Population	2.9998	639.50	0.6755	.	0.0002
9	m2	T	Population	2.9998	33.82	-35.5988	.	37.5096
10	m2	PVALUE	Population	2.9998	0.00	0.0000	.	0.0000
11	m2	L90B	Population	2.9998	20525.11	-25.2138	.	0.0064
12	m2	U90B	Population	2.9998	22736.68	-22.8778	.	0.0070

A slightly different OUTEST= data set is created when you use the RSQUARE selection method. The following statements request only the "best" model for each subset size but ask for a variety of model selection statistics, as well as the estimated regression coefficients. An OUTEST= data set is created and displayed. See Figure 105.23 and Figure 105.24 for the results.

```
proc reg data=fitness outest=est;
   model Oxygen=Age Weight RunTime RunPulse RestPulse MaxPulse
         / selection=rsquare mse jp gmsep cp aic bic sbc b best=1;
run;
proc print data=est;
run;
```

Figure 105.23 PROC REG Output for Physical Fitness Data: Best Models

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

R-Square Selection Method

Number in Model	R-Square	C(p)	AIC	BIC	Estimated MSE of Prediction	J(p)	MSE	SBC
1	0.7434	13.6988	64.5341	65.4673	8.0546	8.0199	7.53384	67.40210
2	0.7642	12.3894	63.9050	64.8212	7.9478	7.8621	7.16842	68.20695
3	0.8111	6.9596	59.0373	61.3127	6.8583	6.7253	5.95669	64.77326
4	0.8368	4.8800	56.4995	60.3996	6.3984	6.2053	5.34346	63.66941
5	0.8480	5.1063	56.2986	61.5667	6.4565	6.1782	5.17634	64.90250
6	0.8487	7.0000	58.1616	64.0748	6.9870	6.5804	5.36825	68.19952

		Parameter Estimates						
Number in Model	R-Square	Intercept	Age	Weight	RunTime	RunPulse	RestPulse	MaxPulse
1	0.7434	82.42177	.	.	-3.31056	.	.	.
2	0.7642	88.46229	-0.15037	.	-3.20395	.	.	.
3	0.8111	111.71806	-0.25640	.	-2.82538	-0.13091	.	.
4	0.8368	98.14789	-0.19773	.	-2.76758	-0.34811	.	0.27051
5	0.8480	102.20428	-0.21962	-0.07230	-2.68252	-0.37340	.	0.30491
6	0.8487	102.93448	-0.22697	-0.07418	-2.62865	-0.36963	-0.02153	0.30322

Figure 105.24 PROC PRINT Output for Physical Fitness Data: OUTEST= Data Set

Obs	_MODEL_	_TYPE_	_DEPVAR_	_RMSE_	Intercept	Age	Weight	RunTime	RunPulse	RestPulse
1	MODEL1	PARMS	Oxygen	2.74478	82.422	.	.	-3.31056	.	.
2	MODEL1	PARMS	Oxygen	2.67739	88.462	-0.15037	.	-3.20395	.	.
3	MODEL1	PARMS	Oxygen	2.44063	111.718	-0.25640	.	-2.82538	-0.13091	.
4	MODEL1	PARMS	Oxygen	2.31159	98.148	-0.19773	.	-2.76758	-0.34811	.
5	MODEL1	PARMS	Oxygen	2.27516	102.204	-0.21962	-0.072302	-2.68252	-0.37340	.
6	MODEL1	PARMS	Oxygen	2.31695	102.934	-0.22697	-0.074177	-2.62865	-0.36963	-0.021534

Obs	MaxPulse	Oxygen	_IN_	_P_	_EDF_	_MSE_	_RSQ_	_CP_	_JP_	_GMSEP_	_AIC_	_BIC_	_SBC_
1	.	-1	1	2	29	7.53384	0.74338	13.6988	8.01990	8.05462	64.5341	65.4673	67.4021
2	.	-1	2	3	28	7.16842	0.76425	12.3894	7.86214	7.94778	63.9050	64.8212	68.2069
3	.	-1	3	4	27	5.95669	0.81109	6.9596	6.72530	6.85833	59.0373	61.3127	64.7733
4	0.27051	-1	4	5	26	5.34346	0.83682	4.8800	6.20531	6.39837	56.4995	60.3996	63.6694
5	0.30491	-1	5	6	25	5.17634	0.84800	5.1063	6.17821	6.45651	56.2986	61.5667	64.9025
6	0.30322	-1	6	7	24	5.36825	0.84867	7.0000	6.58043	6.98700	58.1616	64.0748	68.1995

OUTSSCP= Data Sets

The OUTSSCP= option produces a TYPE=SSCP output SAS data set containing sums of squares and crossproducts. A special row (observation) and column (variable) of the matrix called Intercept contain the number of observations and sums. Observations are identified by the character variable _NAME_. The data set contains all variables used in MODEL statements. You can specify additional variables that you want included in the crossproducts matrix with a VAR statement.

The SSCP data set is used when a large number of observations are explored in many different runs. The SSCP data set can be saved and used for subsequent runs, which are much less expensive since PROC REG never reads the original data again. If you run PROC REG once to create only a SSCP data set, you should list all the variables that you might need in a VAR statement or include all the variables that you might need in a MODEL statement.

The following statements use the fitness data from Example 105.2 to produce an output data set with the OUTSSCP= option. The resulting output is shown in Figure 105.25.

```
proc reg data=fitness outsscp=sscp;
   var Oxygen RunTime Age Weight RestPulse RunPulse MaxPulse;
run;
proc print data=sscp;
run;
```

Since a model is not fit to the data and since the only request is to create the SSCP data set, a MODEL statement is not required in this example. However, since the MODEL statement is not used, the VAR statement is required.

Figure 105.25 SSCP Data Set Created with OUTSSCP= Option: REG Procedure

Obs	_TYPE_	_NAME_	Intercept	Oxygen	RunTime	Age	Weight	RestPulse	RunPulse	MaxPulse
1	SSCP	Intercept	31.00	1468.65	328.17	1478.00	2400.78	1657.00	5259.00	5387.00
2	SSCP	Oxygen	1468.65	70429.86	15356.14	69767.75	113522.26	78015.41	248497.31	254866.75
3	SSCP	RunTime	328.17	15356.14	3531.80	15687.24	25464.71	17684.05	55806.29	57113.72
4	SSCP	Age	1478.00	69767.75	15687.24	71282.00	114158.90	78806.00	250194.00	256218.00
5	SSCP	Weight	2400.78	113522.26	25464.71	114158.90	188008.20	128409.28	407745.67	417764.62
6	SSCP	RestPulse	1657.00	78015.41	17684.05	78806.00	128409.28	90311.00	281928.00	288583.00
7	SSCP	RunPulse	5259.00	248497.31	55806.29	250194.00	407745.67	281928.00	895317.00	916499.00
8	SSCP	MaxPulse	5387.00	254866.75	57113.72	256218.00	417764.62	288583.00	916499.00	938641.00
9	N		31.00	31.00	31.00	31.00	31.00	31.00	31.00	31.00

Interactive Analysis

PROC REG enables you to change interactively both the model and the data used to compute the model, and to produce and highlight scatter plots. See the section "Using PROC REG Interactively" on page 8739 for an overview of interactive analysis that uses PROC REG. The following statements can be used interactively (without reinvoking PROC REG): ADD, DELETE, MODEL, MTEST, OUTPUT, PAINT, PLOT, PRINT, REFIT, RESTRICT, REWEIGHT, and TEST. All interactive features are disabled if there is a BY statement.

The ADD, DELETE, and REWEIGHT statements can be used to modify the current MODEL. Every use of an ADD, DELETE, or REWEIGHT statement causes the model label to be modified by attaching an additional number to it. This number is the cumulative total of the number of ADD, DELETE, or REWEIGHT statements following the current MODEL statement.

A more detailed explanation of changing the data used to compute the model is given in the section "Reweighting Observations in an Analysis" on page 8818.

The following statements illustrate the usefulness of the interactive features. First, the full regression model is fit to the Sashelp.Class data, and Figure 105.26 is produced.

```
ods graphics on;

proc reg data=sashelp.Class plots(modelLabel only)=ResidualByPredicted;
   model Weight=Age Height;
run;
```

Figure 105.26 Interactive Analysis: Full Model

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	7215.63710	3607.81855	27.23	<.0001
Error	16	2120.09974	132.50623		
Corrected Total	18	9335.73684			

Root MSE	11.51114	R-Square	0.7729
Dependent Mean	100.02632	Adj R-Sq	0.7445
Coeff Var	11.50811		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-141.22376	33.38309	-4.23	0.0006
Age	1	1.27839	3.11010	0.41	0.6865
Height	1	3.59703	0.90546	3.97	0.0011

Next, the regression model is reduced by the following statements, and Figure 105.27 is produced.

```
   delete age;
   print;
run;
```

Figure 105.27 Interactive Analysis: Reduced Model

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	7193.24912	7193.24912	57.08	<.0001
Error	17	2142.48772	126.02869		
Corrected Total	18	9335.73684			

Root MSE	11.22625	**R-Square**	0.7705
Dependent Mean	100.02632	**Adj R-Sq**	0.7570
Coeff Var	11.22330		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-143.02692	32.27459	-4.43	0.0004
Height	1	3.89903	0.51609	7.55	<.0001

Note that the MODEL label has been changed from MODEL1 to MODEL1.1, since the original MODEL has been changed by the delete statement.

When ODS Graphics is enabled, updated plots are produced whenever a PRINT statement is used. The option

```
plots(modelLabel only)=ResidualByPredicted
```

in the PROC REG statement specifies that the only plot produced is a scatter plot of residuals by predicted values. The MODELLABEL option specifies that the current model label is added to the plot.

The following statements generate a scatter plot of the residuals against the predicted values from the full model. Figure 105.28 is produced, and the scatter plot shows a possible outlier.

```
   add age;
   print;
run;
```

Figure 105.28 Interactive Analysis: Scatter Plot

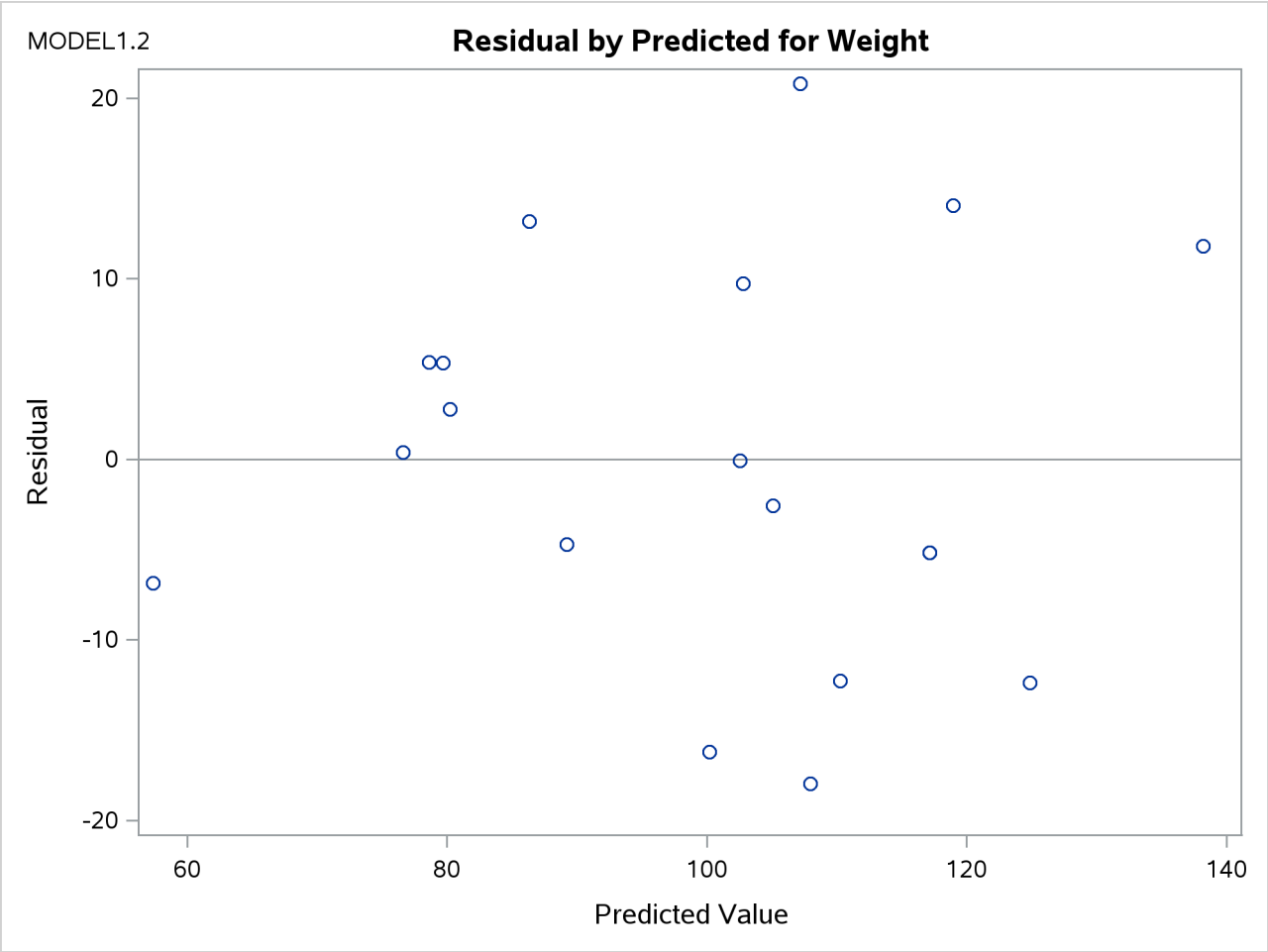

The following statements delete the observation with the largest residual, refit the regression model, and produce a scatter plot of residuals against predicted values for the refitted model. Figure 105.29 shows the new scatter plot.

```
    reweight r.>20;
    print;
run;
```

Figure 105.29 Interactive Analysis: Scatter Plot

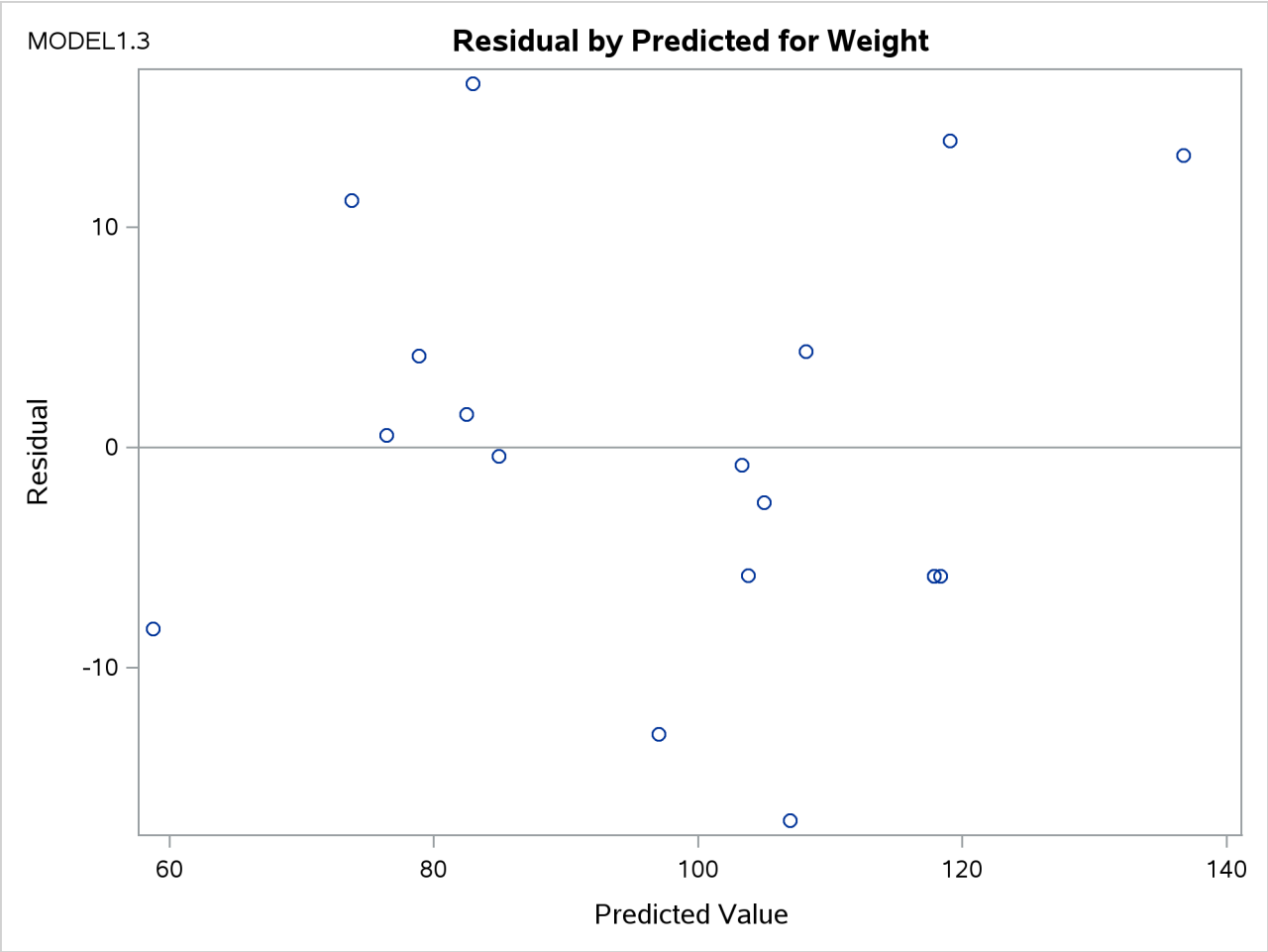

Model-Selection Methods

The nine methods of model selection implemented in PROC REG are specified with the SELECTION= option in the MODEL statement. Each method is discussed in this section.

Full Model Fitted (NONE)

This method is the default and provides no model selection capability. The complete model specified in the MODEL statement is used to fit the model. For many regression analyses, this might be the only method you need.

Forward Selection (FORWARD)

The forward-selection technique begins with no variables in the model. For each of the independent variables, the FORWARD method calculates F statistics that reflect the variable's contribution to the model if it is included. The p-values for these F statistics are compared to the SLENTRY= value that is specified in the MODEL statement (or to 0.50 if the SLENTRY= option is omitted). If no F statistic has a significance level greater than the SLENTRY= value, the FORWARD selection stops. Otherwise, the FORWARD method adds the variable that has the largest F statistic to the model. The FORWARD method then calculates F statistics again for the variables still remaining outside the model, and the evaluation process is repeated. Thus, variables are added one by one to the model until no remaining variable produces a significant F statistic. Once a variable is in the model, it stays.

Backward Elimination (BACKWARD)

The backward elimination technique begins by calculating F statistics for a model which includes all of the independent variables. Then the variables are deleted from the model one by one until all the variables remaining in the model produce F statistics significant at the SLSTAY= level specified in the MODEL statement (or at the 0.10 level if the SLSTAY= option is omitted). At each step, the variable showing the smallest contribution to the model is deleted.

Stepwise (STEPWISE)

The stepwise method is a modification of the forward-selection technique and differs in that variables already in the model do not necessarily stay there. As in the forward-selection method, variables are added one by one to the model, and the F statistic for a variable to be added must be significant at the SLENTRY= level. After a variable is added, however, the stepwise method looks at all the variables already included in the model and deletes any variable that does not produce an F statistic significant at the SLSTAY= level. Only after this check is made and the necessary deletions are accomplished can another variable be added to the model. The stepwise process ends when none of the variables outside the model has an F statistic significant at the SLENTRY= level and every variable in the model is significant at the SLSTAY= level, or when the variable to be added to the model is the one just deleted from it.

Maximum R^2 Improvement (MAXR)

The maximum R square improvement technique does not settle on a single model. Instead, it tries to find the "best" one-variable model, the "best" two-variable model, and so forth, although it is not guaranteed to find the model with the largest R square for each size.

The MAXR method begins by finding the one-variable model producing the highest R square. Then another variable, the one that yields the greatest increase in R square, is added. Once the two-variable model is obtained, each of the variables in the model is compared to each variable not in the model. For each comparison, the MAXR method determines if removing one variable and replacing it with the other variable increases R square. After comparing all possible switches, the MAXR method makes the switch that produces the largest increase in R square. Comparisons begin again, and the process continues until the MAXR method finds that no switch could increase R square. Thus, the two-variable model achieved is considered the "best" two-variable model the technique can find. Another variable is then added to the model, and the comparing-and-switching process is repeated to find the "best" three-variable model, and so forth.

The difference between the STEPWISE method and the MAXR method is that all switches are evaluated before any switch is made in the MAXR method. In the STEPWISE method, the "worst" variable might

be removed without considering what adding the "best" remaining variable might accomplish. The MAXR method might require much more computer time than the STEPWISE method.

Minimum R^2 (MINR) Improvement

The MINR method closely resembles the MAXR method, but the switch chosen is the one that produces the smallest increase in R square. For a given number of variables in the model, the MAXR and MINR methods usually produce the same "best" model, but the MINR method considers more models of each size.

R^2 Selection (RSQUARE)

The RSQUARE method finds subsets of independent variables that best predict a dependent variable by linear regression in the given sample. You can specify the largest and smallest number of independent variables to appear in a subset and the number of subsets of each size to be selected. The RSQUARE method can efficiently perform all possible subset regressions and display the models in decreasing order of R square magnitude within each subset size. Other statistics are available for comparing subsets of different sizes. These statistics, as well as estimated regression coefficients, can be displayed or output to a SAS data set.

The subset models selected by the RSQUARE method are optimal in terms of R square for the given sample, but they are not necessarily optimal for the population from which the sample is drawn or for any other sample for which you might want to make predictions. If a subset model is selected on the basis of a large R square value or any other criterion commonly used for model selection, then all regression statistics computed for that model under the assumption that the model is given a priori, including all statistics computed by PROC REG, are biased.

While the RSQUARE method is a useful tool for exploratory model building, no statistical method can be relied on to identify the "true" model. Effective model building requires substantive theory to suggest relevant predictors and plausible functional forms for the model.

The RSQUARE method differs from the other selection methods in that RSQUARE always identifies the model with the largest R square for each number of variables considered. The other selection methods are not guaranteed to find the model with the largest R square. The RSQUARE method requires much more computer time than the other selection methods, so a different selection method such as the STEPWISE method is a good choice when there are many independent variables to consider.

Adjusted R^2 Selection (ADJRSQ)

This method is similar to the RSQUARE method, except that the adjusted R square statistic is used as the criterion for selecting models, and the method finds the models with the highest adjusted R square within the range of sizes.

Mallows' C_p Selection (CP)

This method is similar to the ADJRSQ method, except that Mallows' C_p statistic is used as the criterion for model selection. Models are listed in ascending order of C_p.

Additional Information about Model-Selection Methods

Reviews of model-selection methods by Hocking (1976) and Judge et al. (1980) describe these and other variable-selection methods.

Criteria Used in Model-Selection Methods

When many significance tests are performed, each at a level of, for example, 5%, the overall probability of rejecting at least one true null hypothesis is much larger than 5%. If you want to guard against including any variables that do not contribute to the predictive power of the model in the population, you should specify a very small SLE= significance level for the FORWARD and STEPWISE methods and a very small SLS= significance level for the BACKWARD and STEPWISE methods.

In most applications, many of the variables considered have some predictive power, however small. If you want to choose the model that provides the best prediction computed using the sample estimates, you need only to guard against estimating more parameters than can be reliably estimated with the given sample size, so you should use a moderate significance level, perhaps in the range of 10% to 25%.

In addition to R square, the C_p statistic is displayed for each model generated in the model-selection methods. The C_p statistic is proposed by Mallows (1973) as a criterion for selecting a model. It is a measure of total squared error defined as

$$C_p = \frac{\mathrm{SSE}_p}{s^2} - (N - 2p)$$

where s^2 is the MSE for the full model, and SSE_p is the sum-of-squares error for a model with p parameters including the intercept, if any. If C_p is plotted against p, Mallows recommends the model where C_p first approaches p. When the right model is chosen, the parameter estimates are unbiased, and this is reflected in C_p near p. For further discussion, see Daniel and Wood (1980).

The adjusted R square statistic is an alternative to R square that is adjusted for the number of parameters in the model. The adjusted R square statistic is calculated as

$$\mathrm{ADJRSQ} = 1 - \frac{(n - i)(1 - R^2)}{n - p}$$

where n is the number of observations used in fitting the model, and i is an indicator variable that is 1 if the model includes an intercept, and 0 otherwise.

Limitations in Model-Selection Methods

The use of model-selection methods can be time-consuming in some cases because there is no built-in limit on the number of independent variables, and the calculations for a large number of independent variables can be lengthy. The recommended limit on the number of independent variables for the MINR method is $20 + i$, where i is the value of the INCLUDE= option.

For the RSQUARE, ADJRSQ, or CP method, with a large value of the BEST= option, adding one more variable to the list from which regressors are selected might significantly increase the CPU time. Also, the time required for the analysis is highly dependent on the data and on the values of the BEST=, START=, and STOP= options.

Parameter Estimates and Associated Statistics

The following example uses the fitness data from Example 105.2. Figure 105.31 shows the parameter estimates and the tables from the SS1, SS2, STB, CLB, COVB, and CORRB options:

```
proc reg data=fitness;
   model Oxygen=RunTime Age Weight RunPulse MaxPulse RestPulse
         / ss1 ss2 stb clb covb corrb;
run;
```

The procedure first displays an analysis of variance table (Figure 105.30). The F statistic for the overall model is significant, indicating that the model explains a significant portion of the variation in the data.

Figure 105.30 ANOVA Table

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	722.54361	120.42393	22.43	<.0001
Error	24	128.83794	5.36825		
Corrected Total	30	851.38154			

Root MSE	2.31695	R-Square	0.8487
Dependent Mean	47.37581	Adj R-Sq	0.8108
Coeff Var	4.89057		

The procedure next displays parameter estimates and some associated statistics (Figure 105.31). First, the estimates are shown, followed by their standard errors. The next two columns of the table contain the t statistics and the corresponding probabilities for testing the null hypothesis that the parameter is not significantly different from zero. These probabilities are usually referred to as p-values. For example, the Intercept term in the model is estimated to be 102.9 and is significantly different from zero. The next two columns of the table are the result of requesting the SS1 and SS2 options, and they show sequential and partial sums of squares (SS) associated with each variable. The standardized estimates (produced by the

STB option) are the parameter estimates that result when all variables are standardized to a mean of 0 and a variance of 1. These estimates are computed by multiplying the original estimates by the standard deviation of the regressor (independent) variable and then dividing by the standard deviation of the dependent variable. The CLB option adds the upper and lower 95% confidence limits for the parameter estimates; the α level can be changed by specifying the ALPHA= option in the PROC REG or MODEL statement.

Figure 105.31 SS1, SS2, STB, CLB, COVB, and CORRB Options: Parameter Estimates

Parameter Estimates										
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	Type I SS	Type II SS	Standardized Estimate	95% Confidence Limits	
Intercept	1	102.93448	12.40326	8.30	<.0001	69578	369.72831	0	77.33541	128.53355
RunTime	1	-2.62865	0.38456	-6.84	<.0001	632.90010	250.82210	-0.68460	-3.42235	-1.83496
Age	1	-0.22697	0.09984	-2.27	0.0322	17.76563	27.74577	-0.22204	-0.43303	-0.02092
Weight	1	-0.07418	0.05459	-1.36	0.1869	5.60522	9.91059	-0.11597	-0.18685	0.03850
RunPulse	1	-0.36963	0.11985	-3.08	0.0051	38.87574	51.05806	-0.71133	-0.61699	-0.12226
MaxPulse	1	0.30322	0.13650	2.22	0.0360	26.82640	26.49142	0.52161	0.02150	0.58493
RestPulse	1	-0.02153	0.06605	-0.33	0.7473	0.57051	0.57051	-0.03080	-0.15786	0.11480

The final two tables are produced as a result of requesting the COVB and CORRB options (Figure 105.32). These tables show the estimated covariance matrix of the parameter estimates, and the estimated correlation matrix of the estimates.

Figure 105.32 SS1, SS2, STB, CLB, COVB, and CORRB Options: Covariances and Correlations

Covariance of Estimates							
Variable	Intercept	RunTime	Age	Weight	RunPulse	MaxPulse	RestPulse
Intercept	153.84081152	0.7678373769	-0.902049478	-0.178237818	0.280796516	-0.832761667	-0.147954715
RunTime	0.7678373769	0.1478880839	-0.014191688	-0.004417672	-0.009047784	0.0046249498	-0.010915224
Age	-0.902049478	-0.014191688	0.009967521	0.0010219105	-0.001203914	0.0035823843	0.0014897532
Weight	-0.178237818	-0.004417672	0.0010219105	0.0029804131	0.0009644683	-0.001372241	0.0003799295
RunPulse	0.280796516	-0.009047784	-0.001203914	0.0009644683	0.0143647273	-0.014952457	-0.000764507
MaxPulse	-0.832761667	0.0046249498	0.0035823843	-0.001372241	-0.014952457	0.0186309364	0.0003425724
RestPulse	-0.147954715	-0.010915224	0.0014897532	0.0003799295	-0.000764507	0.0003425724	0.0043631674

Correlation of Estimates							
Variable	Intercept	RunTime	Age	Weight	RunPulse	MaxPulse	RestPulse
Intercept	1.0000	0.1610	-0.7285	-0.2632	0.1889	-0.4919	-0.1806
RunTime	0.1610	1.0000	-0.3696	-0.2104	-0.1963	0.0881	-0.4297
Age	-0.7285	-0.3696	1.0000	0.1875	-0.1006	0.2629	0.2259
Weight	-0.2632	-0.2104	0.1875	1.0000	0.1474	-0.1842	0.1054
RunPulse	0.1889	-0.1963	-0.1006	0.1474	1.0000	-0.9140	-0.0966
MaxPulse	-0.4919	0.0881	0.2629	-0.1842	-0.9140	1.0000	0.0380
RestPulse	-0.1806	-0.4297	0.2259	0.1054	-0.0966	0.0380	1.0000

For further discussion of the parameters and statistics, see the section "Displayed Output" on page 8832, and Chapter 4, "Introduction to Regression Procedures."

Predicted and Residual Values

The display of the predicted values and residuals is controlled by the P, R, CLM, and CLI options in the MODEL statement. The P option causes PROC REG to display the observation number, the ID value (if an ID statement is used), the actual value, the predicted value, and the residual. The R, CLI, and CLM options also produce the items under the P option. Thus, P is unnecessary if you use one of the other options.

The R option requests more detail, especially about the residuals. The standard errors of the mean predicted value and the residual are displayed. The studentized residual, which is the residual divided by its standard error, is both displayed and plotted. A measure of influence, Cook's D, is displayed and plotted. Cook's D measures the change to the estimates that results from deleting each observation (Cook 1977, 1979). This statistic is very similar to DFFITS.

The CLM option requests that PROC REG display the $100(1-\alpha)\%$ lower and upper confidence limits for the mean predicted values. This accounts for the variation due to estimating the parameters only. If you want a $100(1-\alpha)\%$ confidence interval for observed values, then you can use the CLI option, which adds in the variability of the error term. The α level can be specified with the ALPHA= option in the PROC REG or MODEL statement.

You can use these statistics in PLOT and PAINT statements. This is useful in performing a variety of regression diagnostics. For definitions of the statistics produced by these *options*, see Chapter 4, "Introduction to Regression Procedures."

The following statements use the U.S. population data found in the section "Polynomial Regression" on page 8728. The results are shown in Figure 105.33 and Figure 105.34.

```
ods graphics on;

data USPop2;
   input Year @@;
   YearSq=Year*Year;
   datalines;
2010 2020 2030
;

data USPop2;
   set USPopulation USPop2;
run;

proc reg data=USPop2;
   id Year;
   model Population=Year YearSq / r cli clm;
run;
```

Figure 105.33 Regression Using the R, CLI, and CLM Options

The REG Procedure
Model: MODEL1
Dependent Variable: Population

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	159529	79765	8864.19	<.0001
Error	19	170.97193	8.99852		
Corrected Total	21	159700			

Root MSE	2.99975	R-Square	0.9989
Dependent Mean	94.64800	Adj R-Sq	0.9988
Coeff Var	3.16938		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	21631	639.50181	33.82	<.0001
Year	1	-24.04581	0.67547	-35.60	<.0001
YearSq	1	0.00668	0.00017820	37.51	<.0001

Figure 105.34 Regression Using the R, CLI, and CLM Options

The REG Procedure
Model: MODEL1
Dependent Variable: Population

			Output Statistics									
Obs	Year	Dependent Variable	Predicted Value	Std Error Mean Predict	95% CL Mean		95% CL Predict		Residual	Std Error Residual	Student Residual	Cook's D
1	1790	3.93	6.2127	1.7565	2.5362	9.8892	-1.0631	13.4884	-2.2837	2.432	-0.939	0.153
2	1800	5.31	5.7226	1.4560	2.6751	8.7701	-1.2565	12.7017	-0.4146	2.623	-0.158	0.003
3	1810	7.24	6.5694	1.2118	4.0331	9.1057	-0.2021	13.3409	0.6696	2.744	0.244	0.004
4	1820	9.64	8.7531	1.0305	6.5963	10.9100	2.1144	15.3918	0.8849	2.817	0.314	0.004
5	1830	12.87	12.2737	0.9163	10.3558	14.1916	5.7087	18.8386	0.5923	2.856	0.207	0.001
6	1840	17.07	17.1311	0.8650	15.3207	18.9415	10.5968	23.6655	-0.0621	2.872	-0.022	0.000
7	1850	23.19	23.3254	0.8613	21.5227	25.1281	16.7932	29.8576	-0.1344	2.873	-0.047	0.000
8	1860	31.44	30.8566	0.8846	29.0051	32.7080	24.3107	37.4024	0.5864	2.866	0.205	0.001
9	1870	39.82	39.7246	0.9163	37.8067	41.6425	33.1597	46.2896	0.0934	2.856	0.033	0.000
10	1880	50.16	49.9295	0.9436	47.9545	51.9046	43.3476	56.5114	0.2255	2.847	0.079	0.000
11	1890	62.95	61.4713	0.9590	59.4641	63.4785	54.8797	68.0629	1.4757	2.842	0.519	0.010
12	1900	75.99	74.3499	0.9590	72.3427	76.3571	67.7583	80.9415	1.6441	2.842	0.578	0.013
13	1910	91.97	88.5655	0.9436	86.5904	90.5405	81.9836	95.1473	3.4065	2.847	1.196	0.052
14	1920	105.71	104.1178	0.9163	102.2000	106.0357	97.5529	110.6828	1.5922	2.856	0.557	0.011
15	1930	122.78	121.0071	0.8846	119.1556	122.8585	114.4612	127.5529	1.7679	2.866	0.617	0.012
16	1940	131.67	139.2332	0.8613	137.4305	141.0359	132.7010	145.7654	-7.5642	2.873	-2.632	0.208
17	1950	151.33	158.7962	0.8650	156.9858	160.6066	152.2618	165.3306	-7.4712	2.872	-2.601	0.205
18	1960	179.32	179.6961	0.9163	177.7782	181.6139	173.1311	186.2610	-0.3731	2.856	-0.131	0.001
19	1970	203.21	201.9328	1.0305	199.7759	204.0896	195.2941	208.5715	1.2782	2.817	0.454	0.009
20	1980	226.54	225.5064	1.2118	222.9701	228.0427	218.7349	232.2779	1.0356	2.744	0.377	0.009
21	1990	248.71	250.4168	1.4560	247.3693	253.4644	243.4378	257.3959	-1.7068	2.623	-0.651	0.044
22	2000	281.42	276.6642	1.7565	272.9877	280.3407	269.3884	283.9400	4.7578	2.432	1.957	0.666
23	2010	.	304.2484	2.1073	299.8377	308.6591	296.5754	311.9214	.	.	.	.
24	2020	.	333.1695	2.5040	327.9285	338.4104	324.9910	341.3479	.	.	.	.
25	2030	.	363.4274	2.9435	357.2665	369.5883	354.6310	372.2238	.	.	.	.

Figure 105.35 Studentized Residuals and Cook's D

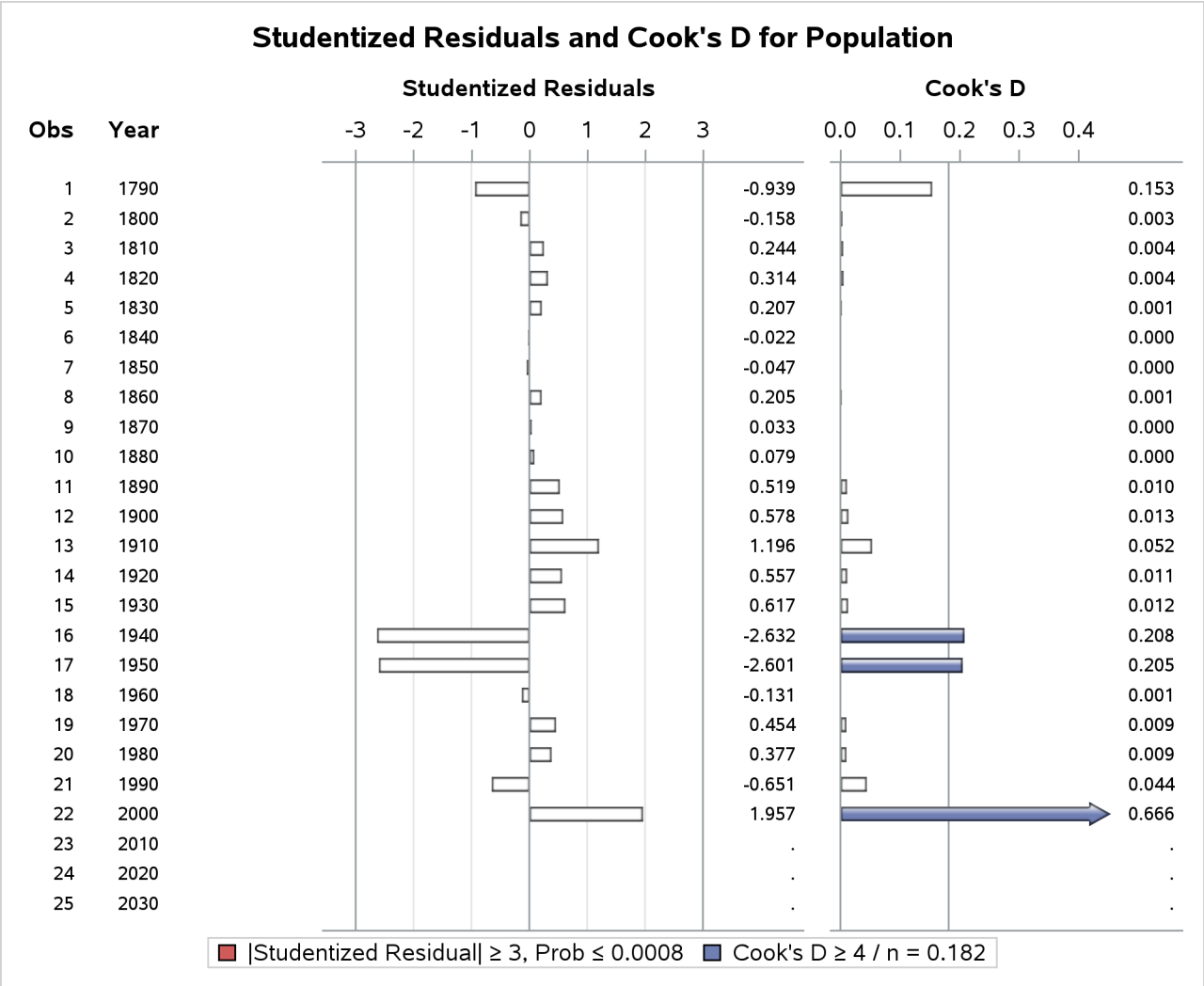

After producing the usual analysis of variance and parameter estimates tables (Figure 105.33), the procedure displays the results of requesting the options for predicted and residual values (Figure 105.34). For each observation, the requested information is shown. Note that the ID variable is used to identify each observation. Also note that, for observations with missing dependent variables, the predicted value, standard error of the predicted value, and confidence intervals for the predicted value are still available.

The studentized residuals and Cook's D statistics in Figure 105.34 and Figure 105.35 are displayed as a result of specifying the R option. The large absolute studentized residuals for 1940 and 1950 (best seen in Figure 105.35) indicate that the overall model is inadequate for explaining the population in these two years. You can use ODS Graphics to obtain plots of studentized residuals by predicted values or leverage; see Example 105.1 for a similar example.

Models of Less Than Full Rank

If the model is not full rank, there are an infinite number of least squares solutions for the estimates. PROC REG chooses a nonzero solution for all variables that are linearly independent of previous variables and a zero solution for other variables. This solution corresponds to using a generalized inverse in the normal equations, and the expected values of the estimates are the Hermite normal form of $\mathbf{X}$ multiplied by the true parameters:

$$E(\mathbf{b}) = (\mathbf{X}'\mathbf{X})^{-}(\mathbf{X}'\mathbf{X})\boldsymbol{\beta}$$

Degrees of freedom for the zeroed estimates are reported as zero. The hypotheses that are not testable have t tests reported as missing. The message that the model is not full rank includes a display of the relations that exist in the matrix.

The following statements use the fitness data from Example 105.2. The variable Dif=RunPulse–RestPulse is created. When this variable is included in the model along with RunPulse and RestPulse, there is a linear dependency (or exact collinearity) between the independent variables. Figure 105.36 shows how this problem is diagnosed.

```
data fit2;
   set fitness;
   Dif=RunPulse-RestPulse;
run;
proc reg data=fit2;
   model Oxygen=RunTime Age Weight RunPulse MaxPulse RestPulse Dif;
run;
```

Figure 105.36 Model That Is Not Full Rank: REG Procedure

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	722.54361	120.42393	22.43	<.0001
Error	24	128.83794	5.36825		
Corrected Total	30	851.38154			

Root MSE	2.31695	R-Square	0.8487
Dependent Mean	47.37581	Adj R-Sq	0.8108
Coeff Var	4.89057		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	102.93448	12.40326	8.30	<.0001
RunTime	1	-2.62865	0.38456	-6.84	<.0001
Age	1	-0.22697	0.09984	-2.27	0.0322
Weight	1	-0.07418	0.05459	-1.36	0.1869
RunPulse	B	-0.36963	0.11985	-3.08	0.0051
MaxPulse	1	0.30322	0.13650	2.22	0.0360
RestPulse	B	-0.02153	0.06605	-0.33	0.7473
Dif	0	0	.	.	.

PROC REG produces a message informing you that the model is less than full rank. Parameters with DF=0 are not estimated, and parameters with DF=B are biased. In addition, the form of the linear dependency among the regressors is displayed.

Collinearity Diagnostics

When a regressor is nearly a linear combination of other regressors in the model, the affected estimates are unstable and have high standard errors. This problem is called *collinearity* or *multicollinearity*. It is a good idea to find out which variables are nearly collinear with which other variables. The approach in PROC REG follows that of Belsley, Kuh, and Welsch (1980). PROC REG provides several methods for detecting collinearity with the COLLIN, COLLINOINT, TOL, and VIF options.

The COLLIN option in the MODEL statement requests that a collinearity analysis be performed. First, $\mathbf{X}'\mathbf{X}$ is scaled to have 1s on the diagonal. If you specify the COLLINOINT option, the intercept variable is adjusted out first. Then the eigenvalues and eigenvectors are extracted. The analysis in PROC REG is reported with eigenvalues of $\mathbf{X}'\mathbf{X}$ rather than singular values of $\mathbf{X}$. The eigenvalues of $\mathbf{X}'\mathbf{X}$ are the squares of the singular values of $\mathbf{X}$.

The condition indices are the square roots of the ratio of the largest eigenvalue to each individual eigenvalue. The largest condition index is the condition number of the scaled $\mathbf{X}$ matrix. Belsley, Kuh, and Welsch (1980)

suggest that, when this number is around 10, weak dependencies might be starting to affect the regression estimates. When this number is larger than 100, the estimates might have a fair amount of numerical error (although the statistical standard error almost always is much greater than the numerical error).

For each variable, PROC REG produces the proportion of the variance of the estimate accounted for by each principal component. A collinearity problem occurs when a component associated with a high condition index contributes strongly (variance proportion greater than about 0.5) to the variance of two or more variables.

The VIF option in the MODEL statement provides the variance inflation factors (VIF). These factors measure the inflation in the variances of the parameter estimates due to collinearities that exist among the regressor (independent) variables. There are no formal criteria for deciding if a VIF is large enough to affect the predicted values.

The TOL option requests the tolerance values for the parameter estimates. The tolerance is defined as 1 / VIF.

For a complete discussion of the preceding methods, see Belsley, Kuh, and Welsch (1980). For a more detailed explanation of using the methods with PROC REG, see Freund and Littell (1986).

This example uses the COLLIN option on the fitness data found in Example 105.2. The following statements produce Figure 105.37.

```
proc reg data=fitness;
   model Oxygen=RunTime Age Weight RunPulse MaxPulse RestPulse
         / tol vif collin;
run;
```

Figure 105.37 Regression Using the TOL, VIF, and COLLIN Options

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	722.54361	120.42393	22.43	<.0001
Error	24	128.83794	5.36825		
Corrected Total	30	851.38154			

Root MSE	2.31695	R-Square	0.8487
Dependent Mean	47.37581	Adj R-Sq	0.8108
Coeff Var	4.89057		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	Tolerance	Variance Inflation
Intercept	1	102.93448	12.40326	8.30	<.0001	.	0
RunTime	1	-2.62865	0.38456	-6.84	<.0001	0.62859	1.59087
Age	1	-0.22697	0.09984	-2.27	0.0322	0.66101	1.51284
Weight	1	-0.07418	0.05459	-1.36	0.1869	0.86555	1.15533
RunPulse	1	-0.36963	0.11985	-3.08	0.0051	0.11852	8.43727
MaxPulse	1	0.30322	0.13650	2.22	0.0360	0.11437	8.74385
RestPulse	1	-0.02153	0.06605	-0.33	0.7473	0.70642	1.41559

Figure 105.37 *continued*

Collinearity Diagnostics									
			Proportion of Variation						
Number	Eigenvalue	Condition Index	Intercept	RunTime	Age	Weight	RunPulse	MaxPulse	RestPulse
1	6.94991	1.00000	0.00002326	0.00021086	0.00015451	0.00019651	0.00000862	0.00000634	0.00027850
2	0.01868	19.29087	0.00218	0.02522	0.14632	0.01042	0.00000244	0.00000743	0.39064
3	0.01503	21.50072	0.00061541	0.12858	0.15013	0.23571	0.00119	0.00125	0.02809
4	0.00911	27.62115	0.00638	0.60897	0.03186	0.18313	0.00149	0.00123	0.19030
5	0.00607	33.82918	0.00133	0.12501	0.11284	0.44442	0.01506	0.00833	0.36475
6	0.00102	82.63757	0.79966	0.09746	0.49660	0.10330	0.06948	0.00561	0.02026
7	0.00017947	196.78560	0.18981	0.01455	0.06210	0.02283	0.91277	0.98357	0.00568

Model Fit and Diagnostic Statistics

This section gathers the formulas for the statistics available in the MODEL, PLOT, and OUTPUT statements. The model to be fit is $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, and the parameter estimate is denoted by $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{Y}$. The subscript i denotes values for the ith observation, the parenthetical subscript (i) means that the statistic is computed by using all observations except the ith observation, and the subscript jj indicates the jth diagonal matrix entry. The ALPHA= option in the PROC REG or MODEL statement is used to set the α value for the t statistics.

Table 105.7 contains the summary statistics for assessing the fit of the model.

Table 105.7 Formulas and Definitions for Model Fit Summary Statistics

Model Option or Statistic	Definition or Formula
n	The number of observations
p	The number of parameters including the intercept
i	1 if there is an intercept, 0 otherwise
$\hat{\sigma}^2$	The estimate of pure error variance from the SIGMA= option or from fitting the full model
SST_0	The uncorrected total sum of squares for the dependent variable
SST_1	The total sum of squares corrected for the mean for the dependent variable
SSE	The error sum of squares
MSE	$\dfrac{\mathrm{SSE}}{n-p}$
R^2	$1 - \dfrac{\mathrm{SSE}}{\mathrm{SST}_i}$
ADJRSQ	$1 - \dfrac{(n-i)(1-R^2)}{n-p}$
AIC	$n \ln\left(\dfrac{\mathrm{SSE}}{n}\right) + 2p$

Table 105.7 *continued*

Model Option or Statistic	Definition or Formula
BIC	$n \ln\left(\frac{\text{SSE}}{n}\right) + 2(p+2)q - 2q^2$ where $q = \frac{n\hat{\sigma}^2}{\text{SSE}}$
CP (C_p)	$\frac{\text{SSE}}{\hat{\sigma}^2} + 2p - n$
GMSEP	$\frac{\text{MSE}(n+1)(n-2)}{n(n-p-1)} = \frac{1}{n} S_p(n+1)(n-2)$
JP (J_p)	$\frac{n+p}{n}\text{MSE}$
PC	$\frac{n+p}{n-p}(1-R^2) = J_p\left(\frac{n}{\text{SST}_i}\right)$
PRESS	The sum of squares of predr_i (see Table 105.8)
RMSE	$\sqrt{\text{MSE}}$
SBC	$n \ln\left(\frac{\text{SSE}}{n}\right) + p \ln(n)$
SP (S_p)	$\frac{\text{MSE}}{n-p-1}$

Table 105.8 contains the diagnostic statistics and their formulas; these formulas and further information can be found in Chapter 4, "Introduction to Regression Procedures," and in the section "Influence Statistics" on page 8810. Each statistic is computed for each observation.

Table 105.8 Formulas and Definitions for Diagnostic Statistics

MODEL Option or Statistic	Formula
PRED ($\widehat{\mathbf{Y}}_i$)	$\mathbf{X}_i\mathbf{b}$
RES (r_i)	$\mathbf{Y}_i - \widehat{\mathbf{Y}}_i$
H (h_i)	$\mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-}\mathbf{x}_i'$
STDP	$\sqrt{h_i\widehat{\sigma}^2}$
STDI	$\sqrt{(1+h_i)\widehat{\sigma}^2}$
STDR	$\sqrt{(1-h_i)\widehat{\sigma}^2}$
LCL	$\widehat{Y}_i - t_{\frac{\alpha}{2}}\text{STDI}$
LCLM	$\widehat{Y}_i - t_{\frac{\alpha}{2}}\text{STDP}$
UCL	$\widehat{Y}_i + t_{\frac{\alpha}{2}}\text{STDI}$
UCLM	$\widehat{Y}_i + t_{\frac{\alpha}{2}}\text{STDP}$
STUDENT	$\frac{r_i}{\text{STDR}_i}$
RSTUDENT	$\frac{r_i}{\hat{\sigma}_{(i)}\sqrt{1-h_i}}$

Table 105.8 *continued*

MODEL Option or Statistic	Formula
COOKD	$\frac{1}{p}\text{STUDENT}^2\frac{\text{STDP}^2}{\text{STDR}^2}$
COVRATIO	$\frac{\det(\hat{\sigma}^2_{(i)}(\mathbf{x}'_{(i)}\mathbf{x}_{(i)})^{-1}}{\det(\hat{\sigma}^2(\mathbf{X}'\mathbf{X})^{-1})}$
DFFITS	$\frac{(\widehat{\mathbf{Y}}_i - \widehat{\mathbf{Y}}_{(i)})}{(\hat{\sigma}_{(i)}\sqrt{h_i})}$
DFBETAS$_j$	$\frac{\mathbf{b}_j - \mathbf{b}_{(i)j}}{\hat{\sigma}_{(i)}\sqrt{(\mathbf{X}'\mathbf{X})_{jj}}}$
PRESS(predr$_i$)	$\frac{r_i}{1-h_i}$

Influence Statistics

This section discusses the INFLUENCE option, which produces several influence statistics, and the PARTIAL option, which produces partial regression leverage plots.

The INFLUENCE Option

The INFLUENCE option (in the MODEL statement) requests the statistics proposed by Belsley, Kuh, and Welsch (1980) to measure the influence of each observation on the estimates. Influential observations are those that, according to various criteria, appear to have a large influence on the parameter estimates.

Let $\mathbf{b}(i)$ be the parameter estimates after deleting the ith observation; let $s(i)^2$ be the variance estimate after deleting the ith observation; let $\mathbf{X}(i)$ be the $\mathbf{X}$ matrix without the ith observation; let $\hat{y}(i)$ be the ith value predicted without using the ith observation; let $r_i = y_i - \hat{y}_i$ be the ith residual; and let h_i be the ith diagonal of the projection matrix for the predictor space, also called the *hat matrix*:

$$h_i = \mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}'_i$$

Belsley, Kuh, and Welsch (1980) propose a cutoff of $2p/n$, where n is the number of observations used to fit the model and p is the number of parameters in the model. Observations with h_i values above this cutoff should be investigated.

For each observation, PROC REG first displays the residual, the studentized residual (RSTUDENT), and the h_i. The studentized residual RSTUDENT differs slightly from STUDENT since the error variance is estimated by $s^2_{(i)}$ without the ith observation, not by s^2. For example,

$$\text{RSTUDENT} = \frac{r_i}{s_{(i)}\sqrt{(1-h_i)}}$$

Observations with RSTUDENT larger than 2 in absolute value might need some attention.

The COVRATIO statistic measures the change in the determinant of the covariance matrix of the estimates by deleting the ith observation:

$$\text{COVRATIO} = \frac{\det\left(s^2(i)(\mathbf{X}'_{(i)}\mathbf{X}_{(i)})^{-1}\right)}{\det\left(s^2(\mathbf{X}'\mathbf{X})^{-1}\right)}$$

Belsley, Kuh, and Welsch (1980) suggest that observations with

$$|\text{COVRATIO} - 1| \geq \frac{3p}{n}$$

where p is the number of parameters in the model and n is the number of observations used to fit the model, are worth investigation.

The DFFITS statistic is a scaled measure of the change in the predicted value for the ith observation and is calculated by deleting the ith observation. A large value indicates that the observation is very influential in its neighborhood of the $\mathbf{X}$ space.

$$\text{DFFITS} = \frac{\hat{y}_i - \hat{y}_{(i)}}{s_{(i)}\sqrt{h_{(i)}}}$$

Large values of DFFITS indicate influential observations. A general cutoff to consider is 2; a size-adjusted cutoff recommended by Belsley, Kuh, and Welsch (1980) is $2\sqrt{p/n}$, where n and p are as defined previously.

The DFFITS statistic is very similar to Cook's D, defined in the section "Predicted and Residual Values" on page 8801.

The DFBETAS statistics are the scaled measures of the change in each parameter estimate and are calculated by deleting the ith observation:

$$\text{DFBETAS}_j = \frac{b_j - b_{(i)j}}{s_{(i)}\sqrt{(\mathbf{X}'\mathbf{X})_{jj}}}$$

where $(\mathbf{X}'\mathbf{X})_{jj}$ is the (j, j) element of $(\mathbf{X}'\mathbf{X})^{-1}$.

In general, large values of DFBETAS indicate observations that are influential in estimating a given parameter. Belsley, Kuh, and Welsch (1980) recommend 2 as a general cutoff value to indicate influential observations and $2/\sqrt{n}$ as a size-adjusted cutoff.

The following statements use a subset of the data in the population example in the section "Polynomial Regression" on page 8728. The INFLUENCE option produces the tables shown in Figure 105.38 and Figure 105.39.

```
proc reg data=USPopulation;
   where Year <= 1970;
   model Population=Year YearSq / influence;
run;
```

Figure 105.38 Regression Using the INFLUENCE Option

The REG Procedure
Model: MODEL1
Dependent Variable: Population

Output Statistics								
						DFBETAS		
Obs	Residual	RStudent	Hat Diag H	Cov Ratio	DFFITS	Intercept	Year	YearSq
1	-1.1094	-0.4972	0.3865	1.8834	-0.3946	-0.2842	0.2810	-0.2779
2	0.2691	0.1082	0.2501	1.6147	0.0625	0.0376	-0.0370	0.0365
3	0.9305	0.3561	0.1652	1.4176	0.1584	0.0666	-0.0651	0.0636
4	0.7908	0.2941	0.1184	1.3531	0.1078	0.0182	-0.0172	0.0161
5	0.2110	0.0774	0.0983	1.3444	0.0256	-0.0030	0.0033	-0.0035
6	-0.6629	-0.2431	0.0951	1.3255	-0.0788	0.0296	-0.0302	0.0307
7	-0.8869	-0.3268	0.1009	1.3214	-0.1095	0.0609	-0.0616	0.0621
8	-0.2501	-0.0923	0.1095	1.3605	-0.0324	0.0216	-0.0217	0.0218
9	-0.7593	-0.2820	0.1164	1.3519	-0.1023	0.0743	-0.0745	0.0747
10	-0.5757	-0.2139	0.1190	1.3650	-0.0786	0.0586	-0.0587	0.0587
11	0.7938	0.2949	0.1164	1.3499	0.1070	-0.0784	0.0783	-0.0781
12	1.1492	0.4265	0.1095	1.3144	0.1496	-0.1018	0.1014	-0.1009
13	3.1664	1.2189	0.1009	1.0168	0.4084	-0.2357	0.2338	-0.2318
14	1.6746	0.6207	0.0951	1.2430	0.2013	-0.0811	0.0798	-0.0784
15	2.2406	0.8407	0.0983	1.1724	0.2776	-0.0427	0.0404	-0.0380
16	-6.6335	-3.1845	0.1184	0.2924	-1.1673	-0.1531	0.1636	-0.1747
17	-6.0147	-2.8433	0.1652	0.3989	-1.2649	-0.4843	0.4958	-0.5076
18	1.6770	0.6847	0.2501	1.4757	0.3954	0.2240	-0.2274	0.2308
19	3.9895	1.9947	0.3865	0.9766	1.5831	1.0902	-1.1025	1.1151

Figure 105.39 Residual Statistics

Sum of Residuals	8.85264E-11
Sum of Squared Residuals	123.74557
Predicted Residual SS (PRESS)	188.54924

In Figure 105.38, observations 16, 17, and 19 exceed or are near the cutoff value of 2 for RSTUDENT. None of the observations exceeds the general cutoff of 2 for DFFITS or the DFBETAS, but observations 16, 17, and 19 exceed at least one of the size-adjusted cutoffs for these statistics. Observations 1 and 19 exceed the cutoff for the hat diagonals, and observations 1, 2, 16, 17, and 18 exceed the cutoffs for COVRATIO. Taken together, these statistics indicate that you should look first at observations 16, 17, and 19 and then perhaps investigate the other observations that exceeded a cutoff.

When ODS Graphics is enabled, you can request influence diagnostic plots by using the PLOTS= option in the PROC REG statement as shown in the following statements:

```
ods graphics on;

proc reg data=USPopulation
      plots(label)=(CooksD RStudentByLeverage DFFITS DFBETAS);
   where Year <= 1970;
   id Year;
   model Population=Year YearSq;
run;
```

The LABEL suboption specified in the PLOTS(LABEL)= option requests that observations that exceed the relevant cutoffs for the statistics being plotted are labeled. Since Year has been named in an ID statement, the value of Year is used for the labels. The requested plots are shown in Figure 105.40.

Figure 105.40 Influence Diagnostics

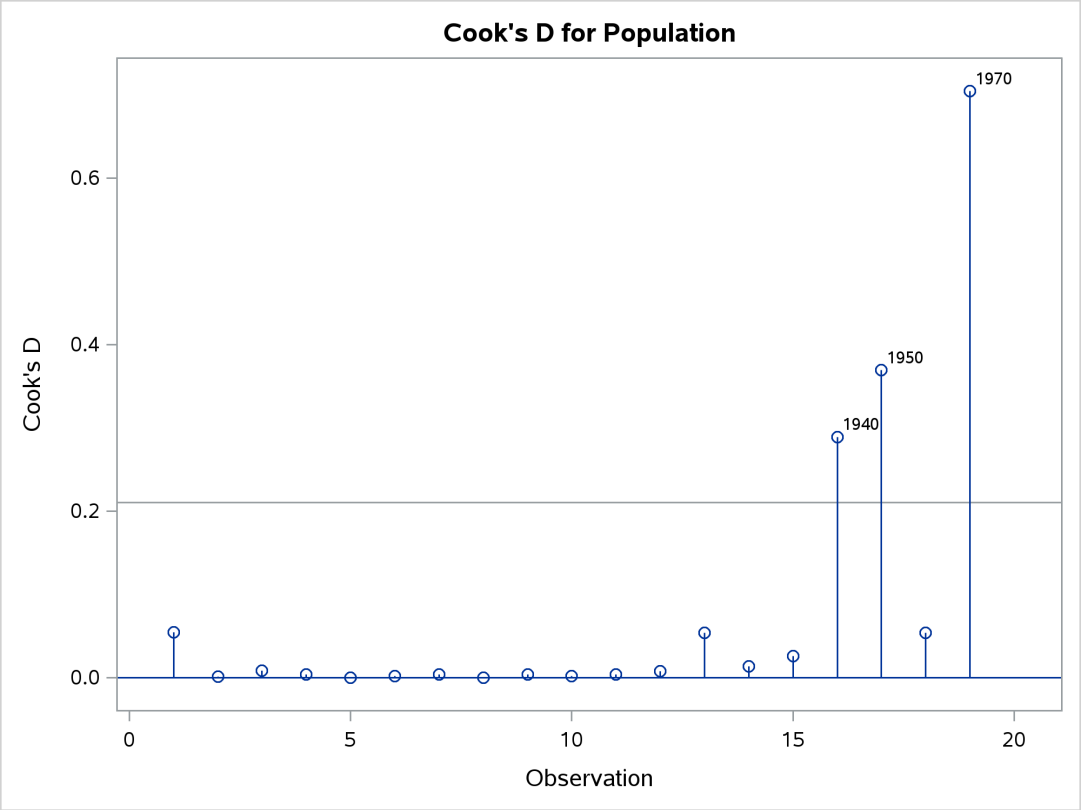

Figure 105.40 *continued*

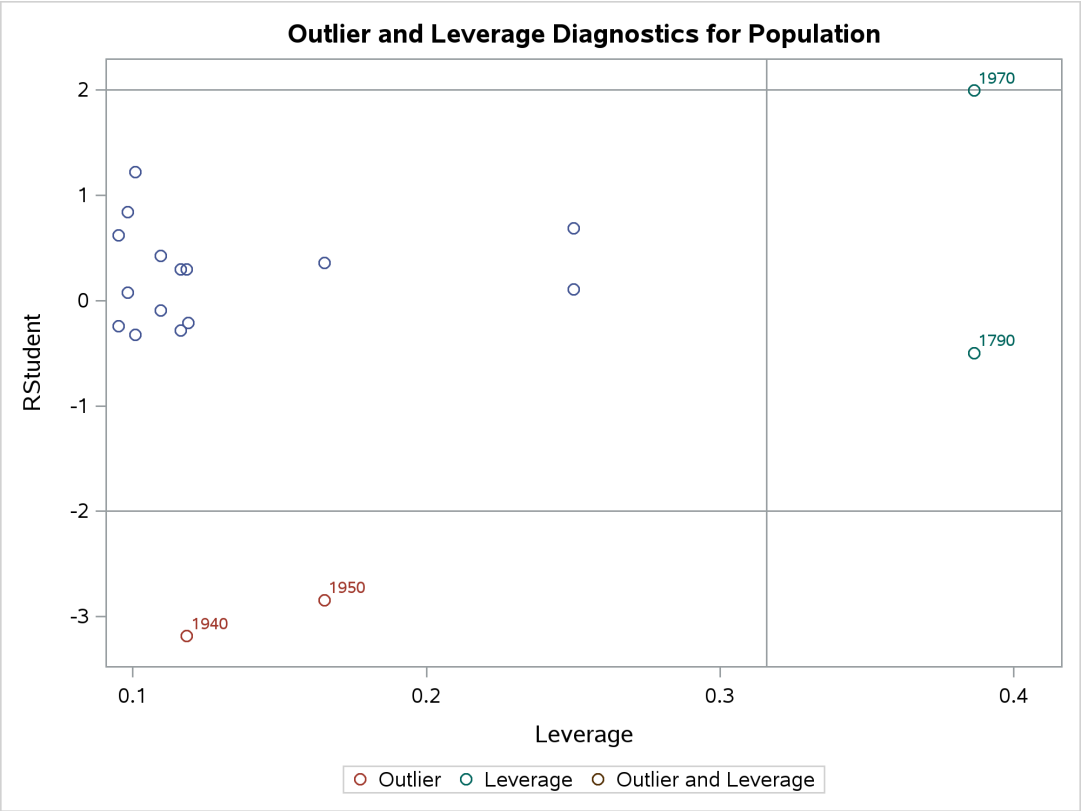

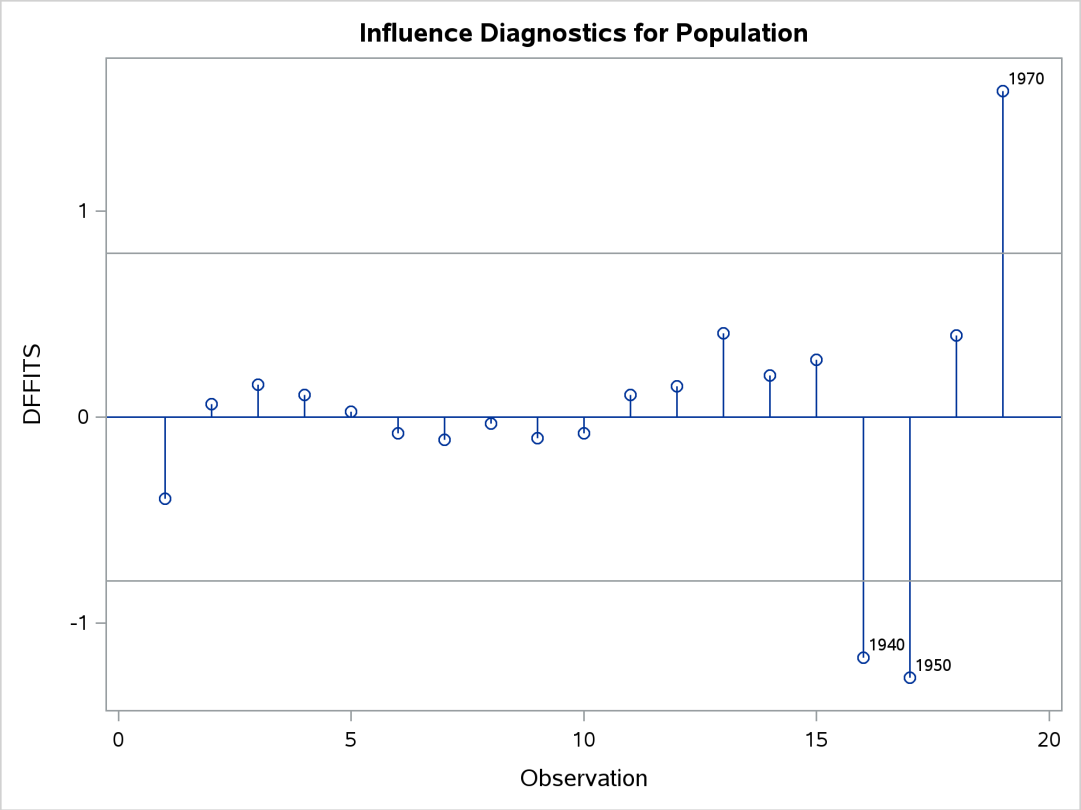

Figure 105.40 *continued*

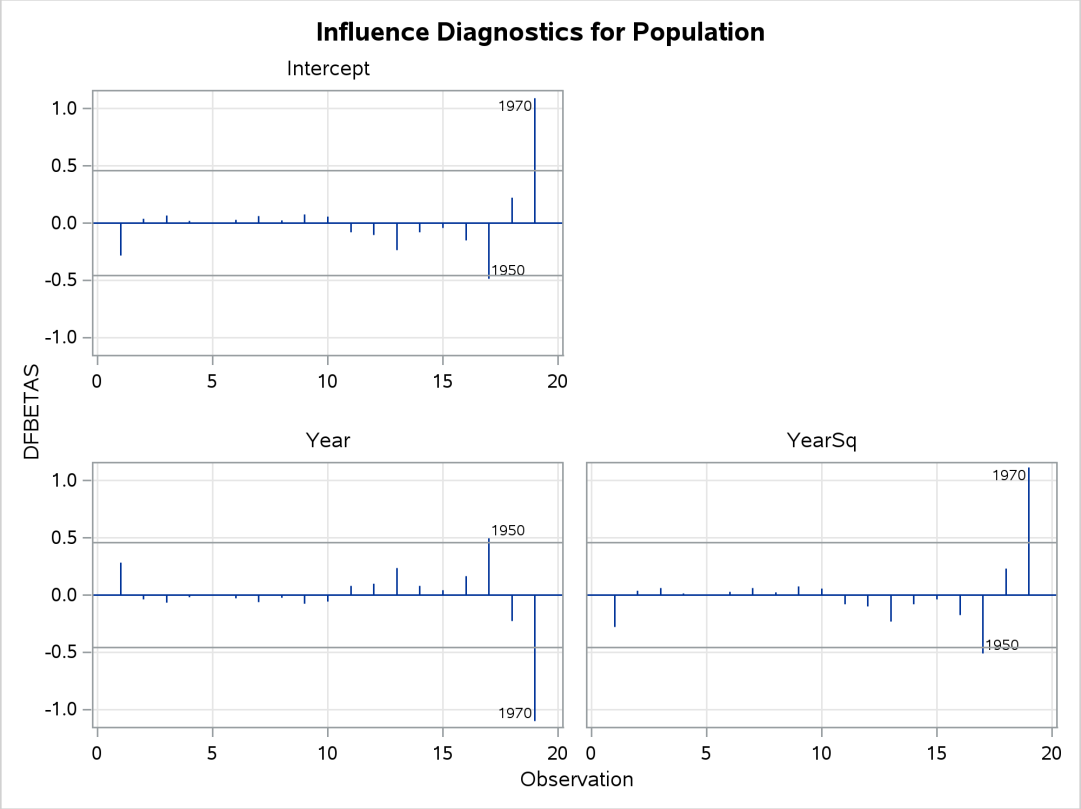

When ODS Graphics is enabled, you can request a chart of the studentized residuals and Cook's *D* by using the PLOTS=RESIDUALCHART option as shown in the following statements:

```
proc reg data=USPopulation plots(only)=residualchart;
   where Year <= 1970;
   id Year;
   model Population=Year YearSq;
run;
```

The chart is shown in Figure 105.41.

Figure 105.41 Studentized Residuals and Cook's D Chart

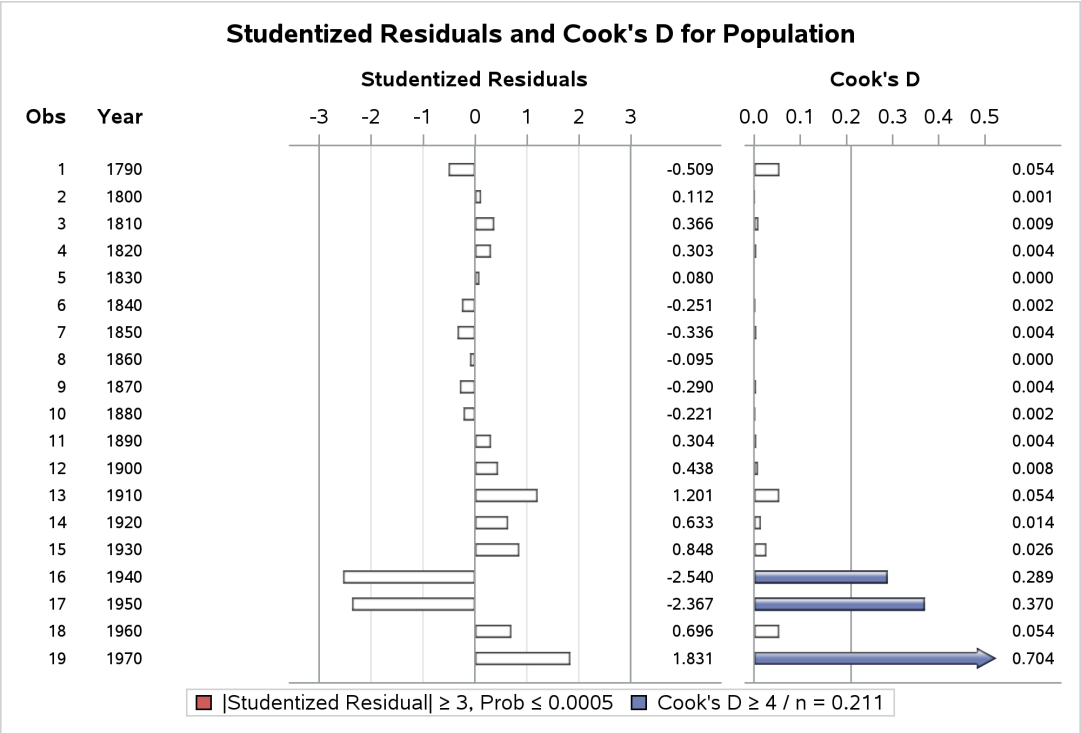

The PARTIAL and PARTIALDATA Options

The PARTIAL option in the MODEL statement produces partial regression leverage plots. If ODS Graphics is not enabled, this option requires the use of the LINEPRINTER option in the PROC REG statement. One plot is created for each regressor in the current full model. For example, plots are produced for regressors included by using ADD statements; plots are not produced for interim models in the various model-selection methods but only for the full model. If you use a model-selection method and the final model contains only a subset of the original regressors, the PARTIAL option still produces plots for all regressors in the full model. If ODS Graphics is enabled, these plots are produced as high-resolution graphics, in panels with a maximum of six partial regression leverage plots per panel. Multiple panels are displayed for models with more than six regressors.

For a given regressor, the partial regression leverage plot is the plot of the dependent variable and the regressor after they have been made orthogonal to the other regressors in the model. These can be obtained by plotting the residuals for the dependent variable against the residuals for the selected regressor, where the residuals for the dependent variable are calculated with the selected regressor omitted, and the residuals for the selected regressor are calculated from a model where the selected regressor is regressed on the remaining regressors. A line fit to the points has a slope equal to the parameter estimate in the full model.

When ODS Graphics is not enabled, points in the plot are marked by the number of replicates appearing at one position. The symbol '*' is used if there are 10 or more replicates. If an ID statement is specified, the leftmost nonblank character in the value of the ID variable is used as the plotting symbol.

The PARTIALDATA option in the MODEL statement produces a table that contains the partial regression data that are displayed in the partial regression leverage plots. You can request partial regression data even if you do not requests plots with the PARTIAL option.

The following statements use the fitness data in Example 105.2 with the PARTIAL option and ODS Graphics to produce the partial regression leverage plots. The plots are shown in Figure 105.42.

```
ods graphics on;

proc reg data=fitness;
   model Oxygen=RunTime Weight Age / partial;
run;

ods graphics off;
```

Figure 105.42 Partial Regression Leverage Plots

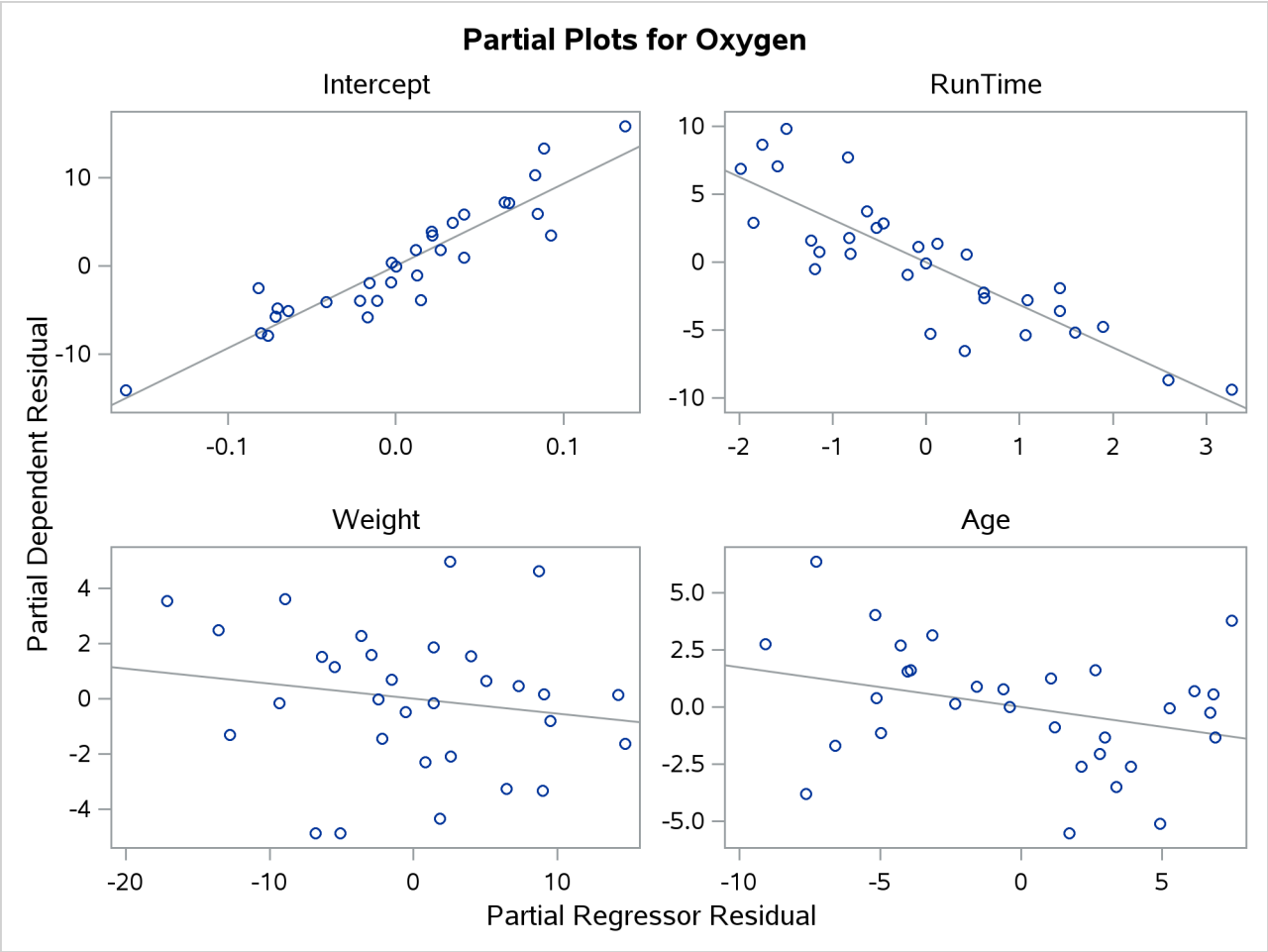

Reweighting Observations in an Analysis

Reweighting observations is an interactive feature of PROC REG that enables you to change the weights of observations used in computing the regression equation. Observations can also be deleted from the analysis (not from the data set) by changing their weights to zero. In the following statements, the Sashelp.Class data are used to illustrate some of the features of the REWEIGHT statement. First, the full model is fit, and the residuals are displayed in Figure 105.43.

```
proc reg data=sashelp.Class;
   model Weight=Age Height / p;
   id Name;
run;
```

Figure 105.43 Full Model for Sashelp.Class Data, Residuals Shown

The REG Procedure
Model: MODEL1
Dependent Variable: Weight

Output Statistics				
Obs	Name	Dependent Variable	Predicted Value	Residual
1	Alfred	112.5	124.8686	-12.3686
2	Alice	84.0	78.6273	5.3727
3	Barbara	98.0	110.2812	-12.2812
4	Carol	102.5	102.5670	-0.0670
5	Henry	102.5	105.0849	-2.5849
6	James	83.0	80.2266	2.7734
7	Jane	84.5	89.2191	-4.7191
8	Janet	112.5	102.7663	9.7337
9	Jeffrey	84.0	100.2095	-16.2095
10	John	99.5	86.3415	13.1585
11	Joyce	50.5	57.3660	-6.8660
12	Judy	90.0	107.9625	-17.9625
13	Louise	77.0	76.6295	0.3705
14	Mary	112.0	117.1544	-5.1544
15	Philip	150.0	138.2164	11.7836
16	Robert	128.0	107.2043	20.7957
17	Ronald	133.0	118.9529	14.0471
18	Thomas	85.0	79.6676	5.3324
19	William	112.0	117.1544	-5.1544

Sum of Residuals	0
Sum of Squared Residuals	2120.09974
Predicted Residual SS (PRESS)	3272.72186

Upon examining the data and residuals, you realize that observation 17 (Ronald) was mistakenly included in the analysis. Also, you would like to examine the effect of reweighting to 0.5 those observations with residuals that have absolute values greater than or equal to 17. The following statements show how you request this reweighting:

```
reweight obs.=17;
reweight r. le -17 or r. ge 17 / weight=0.5;
print p;
run;
```

At this point, a message appears (in the log) that tells you which observations have been reweighted and what the new weights are. Figure 105.44 is produced.

Figure 105.44 Model with Reweighted Observations

The REG Procedure
Model: MODEL1.2
Dependent Variable: Weight

Output Statistics					
Obs	Name	Weight	Dependent Variable	Predicted Value	Residual
1	Alfred	1.0	112.5	121.6250	-9.1250
2	Alice	1.0	84.0	79.9296	4.0704
3	Barbara	1.0	98.0	107.5484	-9.5484
4	Carol	1.0	102.5	102.1663	0.3337
5	Henry	1.0	102.5	104.3632	-1.8632
6	James	1.0	83.0	79.9762	3.0238
7	Jane	1.0	84.5	87.8225	-3.3225
8	Janet	1.0	112.5	103.6889	8.8111
9	Jeffrey	1.0	84.0	98.7606	-14.7606
10	John	1.0	99.5	85.3117	14.1883
11	Joyce	1.0	50.5	58.6811	-8.1811
12	Judy	0.5	90.0	106.8740	-16.8740
13	Louise	1.0	77.0	76.8377	0.1623
14	Mary	1.0	112.0	116.2429	-4.2429
15	Philip	1.0	150.0	135.9688	14.0312
16	Robert	0.5	128.0	103.5150	24.4850
17	Ronald	0.0	133.0	117.8121	15.1879
18	Thomas	1.0	85.0	78.1398	6.8602
19	William	1.0	112.0	116.2429	-4.2429

Sum of Residuals	0
Sum of Squared Residuals	1500.61194
Predicted Residual SS (PRESS)	2287.57621

The first REWEIGHT statement excludes observation 17, and the second REWEIGHT statement reweights observations 12 and 16 to 0.5. An important feature to note from this example is that the model is not refit until after the PRINT statement. REWEIGHT statements do not cause the model to be refit. This is so that multiple REWEIGHT statements can be applied to a subsequent model.

In this example, since the intent is to reweight observations with large residuals, the observation that was mistakenly included in the analysis should be deleted; then the model should be fit for those remaining observations, and the observations with large residuals should be reweighted. To accomplish this, use the REFIT statement. Note that the model label has been changed from MODEL1 to MODEL1.2 since two

REWEIGHT statements have been used. The following statements produce Figure 105.45:

```
reweight allobs / weight=1.0;
reweight obs.=17;
refit;
reweight r. le -17 or r. ge 17 / weight=.5;
print;
run;
```

Figure 105.45 Observations Excluded from Analysis, Model Refitted, and Observations Reweighted

The REG Procedure
Model: MODEL1.5
Dependent Variable: Weight

Output Statistics					
Obs	Name	Weight	Dependent Variable	Predicted Value	Residual
1	Alfred	1.0	112.5	120.9716	-8.4716
2	Alice	1.0	84.0	79.5342	4.4658
3	Barbara	1.0	98.0	107.0746	-9.0746
4	Carol	1.0	102.5	101.5681	0.9319
5	Henry	1.0	102.5	103.7588	-1.2588
6	James	1.0	83.0	79.7204	3.2796
7	Jane	1.0	84.5	87.5443	-3.0443
8	Janet	1.0	112.5	102.9467	9.5533
9	Jeffrey	1.0	84.0	98.3117	-14.3117
10	John	1.0	99.5	85.0407	14.4593
11	Joyce	1.0	50.5	58.6253	-8.1253
12	Judy	1.0	90.0	106.2625	-16.2625
13	Louise	1.0	77.0	76.5908	0.4092
14	Mary	1.0	112.0	115.4651	-3.4651
15	Philip	1.0	150.0	134.9953	15.0047
16	Robert	0.5	128.0	103.1923	24.8077
17	Ronald	0.0	133.0	117.0299	15.9701
18	Thomas	1.0	85.0	78.0288	6.9712
19	William	1.0	112.0	115.4651	-3.4651

Sum of Residuals	0
Sum of Squared Residuals	1637.81879
Predicted Residual SS (PRESS)	2473.87984

Notice that this results in a slightly different model than the previous set of statements: only observation 16 is reweighted to 0.5. Also note that the model label is now MODEL1.5 since five REWEIGHT statements have been used for this model.

Another important feature of the REWEIGHT statement is the ability to nullify the effect of a previous or all REWEIGHT statements. First, assume that you have several REWEIGHT statements in effect and you want to restore the original weights of all the observations. The following REWEIGHT statement accomplishes this and produces Figure 105.46:

```
reweight allobs / reset;
print;
run;
```

Figure 105.46 Restoring Weights of All Observations

The REG Procedure
Model: MODEL1.6
Dependent Variable: Weight

		Output Statistics		
Obs	Name	Dependent Variable	Predicted Value	Residual
1	Alfred	112.5	124.8686	-12.3686
2	Alice	84.0	78.6273	5.3727
3	Barbara	98.0	110.2812	-12.2812
4	Carol	102.5	102.5670	-0.0670
5	Henry	102.5	105.0849	-2.5849
6	James	83.0	80.2266	2.7734
7	Jane	84.5	89.2191	-4.7191
8	Janet	112.5	102.7663	9.7337
9	Jeffrey	84.0	100.2095	-16.2095
10	John	99.5	86.3415	13.1585
11	Joyce	50.5	57.3660	-6.8660
12	Judy	90.0	107.9625	-17.9625
13	Louise	77.0	76.6295	0.3705
14	Mary	112.0	117.1544	-5.1544
15	Philip	150.0	138.2164	11.7836
16	Robert	128.0	107.2043	20.7957
17	Ronald	133.0	118.9529	14.0471
18	Thomas	85.0	79.6676	5.3324
19	William	112.0	117.1544	-5.1544

Sum of Residuals	0
Sum of Squared Residuals	2120.09974
Predicted Residual SS (PRESS)	3272.72186

The resulting model is identical to the original model specified at the beginning of this section. Notice that the model label is now MODEL1.6. Note that the Weight column does not appear, since all observations have been reweighted to have weight=1.

Now suppose you want only to undo the changes made by the most recent REWEIGHT statement. Use REWEIGHT UNDO for this. The following statements produce Figure 105.47:

```
reweight r. le -12 or r. ge 12 / weight=.75;
reweight r. le -17 or r. ge 17 / weight=.5;
reweight undo;
print;
run;
```

Figure 105.47 Example of UNDO in REWEIGHT Statement

The REG Procedure
Model: MODEL1.9
Dependent Variable: Weight

Output Statistics					
Obs	Name	Weight	Dependent Variable	Predicted Value	Residual
1	Alfred	0.75	112.5	125.1152	-12.6152
2	Alice	1.00	84.0	78.7691	5.2309
3	Barbara	0.75	98.0	110.3236	-12.3236
4	Carol	1.00	102.5	102.8836	-0.3836
5	Henry	1.00	102.5	105.3936	-2.8936
6	James	1.00	83.0	80.1133	2.8867
7	Jane	1.00	84.5	89.0776	-4.5776
8	Janet	1.00	112.5	103.3322	9.1678
9	Jeffrey	0.75	84.0	100.2835	-16.2835
10	John	0.75	99.5	86.2090	13.2910
11	Joyce	1.00	50.5	57.0745	-6.5745
12	Judy	0.75	90.0	108.2622	-18.2622
13	Louise	1.00	77.0	76.5275	0.4725
14	Mary	1.00	112.0	117.6752	-5.6752
15	Philip	1.00	150.0	138.9211	11.0789
16	Robert	0.75	128.0	107.0063	20.9937
17	Ronald	0.75	133.0	119.4681	13.5319
18	Thomas	1.00	85.0	79.3061	5.6939
19	William	1.00	112.0	117.6752	-5.6752

Sum of Residuals	0
Sum of Squared Residuals	1694.87114
Predicted Residual SS (PRESS)	2547.22751

The resulting model reflects changes made only by the first REWEIGHT statement since the third REWEIGHT statement negates the effect of the second REWEIGHT statement. Observations 1, 3, 9, 10, 12, 16, and 17 have their weights changed to 0.75. Note that the label MODEL1.9 reflects the use of nine REWEIGHT statements for the current model.

Now suppose you want to reset the observations selected by the most recent REWEIGHT statement to their original weights. Use the REWEIGHT statement with the RESET option to do this. The following statements produce Figure 105.48:

```
reweight r. le -12 or r. ge 12 / weight=.75;
reweight r. le -17 or r. ge 17 / weight=.5;
reweight / reset;
print;
run;
```

Figure 105.48 REWEIGHT Statement with RESET option

The REG Procedure
Model: MODEL1.12
Dependent Variable: Weight

Output Statistics					
Obs	Name	Weight	Dependent Variable	Predicted Value	Residual
1	Alfred	0.75	112.5	126.0076	-13.5076
2	Alice	1.00	84.0	77.8727	6.1273
3	Barbara	0.75	98.0	111.2805	-13.2805
4	Carol	1.00	102.5	102.4703	0.0297
5	Henry	1.00	102.5	105.1278	-2.6278
6	James	1.00	83.0	80.2290	2.7710
7	Jane	1.00	84.5	89.7199	-5.2199
8	Janet	1.00	112.5	102.0122	10.4878
9	Jeffrey	0.75	84.0	100.6507	-16.6507
10	John	0.75	99.5	86.6828	12.8172
11	Joyce	1.00	50.5	56.7703	-6.2703
12	Judy	1.00	90.0	108.1649	-18.1649
13	Louise	1.00	77.0	76.4327	0.5673
14	Mary	1.00	112.0	117.1975	-5.1975
15	Philip	1.00	150.0	138.7581	11.2419
16	Robert	1.00	128.0	108.7016	19.2984
17	Ronald	0.75	133.0	119.0957	13.9043
18	Thomas	1.00	85.0	80.3076	4.6924
19	William	1.00	112.0	117.1975	-5.1975

Sum of Residuals	0
Sum of Squared Residuals	1879.08980
Predicted Residual SS (PRESS)	2959.57279

Note that observations that meet the condition of the second REWEIGHT statement (residuals with an absolute value greater than or equal to 17) now have weights reset to their original value of 1. Observations 1, 3, 9, 10, and 17 have weights of 0.75, but observations 12 and 16 (which meet the condition of the second REWEIGHT statement) have their weights reset to 1.

Notice how the last three examples show three ways to change weights back to a previous value. In the first example, ALLOBS and the RESET option are used to change weights for all observations back to their original values. In the second example, the UNDO option is used to negate the effect of a previous REWEIGHT statement, thus changing weights for observations selected in the previous REWEIGHT statement to the weights specified in still another REWEIGHT statement. In the third example, the RESET option is used to change weights for observations selected in a previous REWEIGHT statement back to their original values. Finally, note that the label MODEL1.12 indicates that 12 REWEIGHT statements have been applied to the original model.

Testing for Heteroscedasticity

The regression model is specified as $y_i = \mathbf{x}_i\boldsymbol{\beta} + \epsilon_i$, where the ϵ_i's are identically and independently distributed: $E(\boldsymbol{\epsilon}) = 0$ and $E(\boldsymbol{\epsilon}'\boldsymbol{\epsilon}) = \sigma^2\mathbf{I}$. If the ϵ_i's are not independent or their variances are not constant, the parameter estimates are unbiased, but the estimate of the covariance matrix is inconsistent.

In the case of heteroscedasticity, if the regression data are from a simple random sample, then White (1980), showed that matrix

$$\mathrm{HC}_0 = (\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathrm{diag}(e_i^2)\mathbf{X})(\mathbf{X}'\mathbf{X})^{-1}$$

where

$$e_i = y_i - \mathbf{x}_i\mathbf{b}$$

is an asymptotically consistent estimate of the covariance matrix. MacKinnon and White (1985) introduced three alternative heteroscedasticity-consistent covariance matrix estimators that are all asymptotically equivalent to the estimator HC_0 but that typically have better small sample behavior. These estimators labeled HC_1, HC_2, and HC_3 are defined as follows:

$$\mathrm{HC}_1 = \frac{n}{n-p}\mathrm{HC}_0$$

where n is the number of observations and p is the number of regressors including the intercept.

$$\mathrm{HC}_2 = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{diag}(\frac{e_i^2}{1-h_{ii}})\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}$$

where

$$h_{ii} = \mathbf{x}_i(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_\mathbf{i}'$$

is the leverage of the ith observation.

$$\mathrm{HC}_3 = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathrm{diag}(\frac{e_i^2}{(1-h_{ii})^2})\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}$$

Long and Ervin (2000) studied the performance of these estimators and recommend using the HC_3 estimator if the sample size is less than 250.

You can use the HCCMETHOD=0,1,2, or 3 in the MODEL statement to select a heteroscedasticity-consistent covariance matrix estimator, with HC_0 being the default. The ACOV option in the MODEL statement displays the heteroscedasticity-consistent covariance matrix estimator in effect and adds heteroscedasticity-consistent standard errors, also known as White standard errors, to the parameter estimates table. If you

specify the HCC or WHITE option in the MODEL statement, but do not also specify the ACOV option, then the heteroscedasticity-consistent standard errors are added to the parameter estimates table but the heteroscedasticity- consistent covariance matrix is not displayed.

The SPEC option performs a model specification test. The null hypothesis for this test maintains that the errors are homoscedastic and independent of the regressors and that several technical assumptions about the model specification are valid. For details, see theorem 2 and assumptions 1–7 of White (1980). When the model is correctly specified and the errors are independent of the regressors, the rejection of this null hypothesis is evidence of heteroscedasticity. In implementing this test, an estimator of the average covariance matrix (White 1980, p. 822) is constructed and inverted. The nonsingularity of this matrix is one of the assumptions in the null hypothesis about the model specification. When PROC REG determines this matrix to be numerically singular, a generalized inverse is used and a note to this effect is written to the log. In such cases, care should be taken in interpreting the results of this test.

When you specify the SPEC, ACOV, HCC, or WHITE option in the MODEL statement, tests listed in the TEST statement are performed with both the usual covariance matrix and the heteroscedasticity-consistent covariance matrix requested with the HCCMETHOD= option. Tests performed with the consistent covariance matrix are asymptotic. For more information, see White (1980).

Both the ACOV and SPEC options can be specified in a MODEL or PRINT statement.

Testing for Lack of Fit

The test for lack of fit compares the variation around the model with "pure" variation within replicated observations. This measures the adequacy of the specified model. In particular, if there are n_i replicated observations $Y_{i1}, \ldots, Y_{in_i}$ of the response all at the same values $\mathbf{x}_i$ of the regressors, then you can predict the true response at $\mathbf{x}_i$ either by using the predicted value $\hat{Y}_i$ based on the model or by using the mean $\bar{Y}_i$ of the replicated values. The test for lack of fit decomposes the residual error into a component due to the variation of the replications around their mean value (the "pure" error) and a component due to the variation of the mean values around the model prediction (the "bias" error):

$$\sum_i \sum_{j=1}^{n_i} \left(Y_{ij} - \hat{Y}_i\right)^2 = \sum_i \sum_{j=1}^{n_i} \left(Y_{ij} - \bar{Y}_i\right)^2 + \sum_i n_i \left(\bar{Y}_i - \hat{Y}_i\right)^2$$

If the model is adequate, then both components estimate the nominal level of error; however, if the bias component of error is much larger than the pure error, then this constitutes evidence that there is significant lack of fit.

If some observations in your design are replicated, you can test for lack of fit by specifying the LACKFIT option in the MODEL statement (see Example 105.6). Note that, since all other tests use total error rather than pure error, you might want to hand-calculate the tests with respect to pure error if the lack of fit is significant. On the other hand, significant lack of fit indicates that the specified model is inadequate, so if this is a problem you can also try to refine the model.

Multivariate Tests

The MTEST statement described in the section "MTEST Statement" on page 8770 can test hypotheses involving several dependent variables in the form

$$(\mathbf{L}\boldsymbol{\beta} - \mathbf{cj})\mathbf{M} = 0$$

where $\mathbf{L}$ is a linear function on the regressor side, $\boldsymbol{\beta}$ is a matrix of parameters, $\mathbf{c}$ is a column vector of constants, $\mathbf{j}$ is a row vector of ones, and $\mathbf{M}$ is a linear function on the dependent side. The special case where the constants are zero is

$$\mathbf{L}\boldsymbol{\beta}\mathbf{M} = 0$$

To test this hypothesis, PROC REG constructs two matrices called $\mathbf{H}$ and $\mathbf{E}$ that correspond to the numerator and denominator of a univariate F test:

$$\begin{aligned}
\mathbf{H} &= \mathbf{M}'(\mathbf{LB} - \mathbf{cj})'(\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}\mathbf{L}')^{-1}(\mathbf{LB} - \mathbf{cj})\mathbf{M} \\
\mathbf{E} &= \mathbf{M}'(\mathbf{Y}'\mathbf{Y} - \mathbf{B}'(\mathbf{X}'\mathbf{X})\mathbf{B})\mathbf{M}
\end{aligned}$$

These matrices are displayed for each MTEST statement if the PRINT option is specified.

Four test statistics based on the eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$ or $(\mathbf{E} + \mathbf{H})^{-1}\mathbf{H}$ are formed. These are Wilks' lambda, Pillai's trace, the Hotelling-Lawley trace, and Roy's greatest root. These test statistics are discussed in Chapter 4, "Introduction to Regression Procedures."

The following code creates MANOVA data from Morrison (1976):

```
* Manova Data from Morrison (1976, 190);
data a;
   input sex $ drug $ @;
   do rep=1 to 4;
      input y1 y2 @;
      sexcode=(sex='m')-(sex='f');
      drug1=(drug='a')-(drug='c');
      drug2=(drug='b')-(drug='c');
      sexdrug1=sexcode*drug1;
      sexdrug2=sexcode*drug2;
      output;
   end;
   datalines;
m a  5  6  5  4  9  9  7  6
m b  7  6  7  7  9 12  6  8
m c 21 15 14 11 17 12 12 10
```

```
f a  7 10  6  6  9  7  8 10
f b 10 13  8  7  7  6  6  9
f c 16 12 14  9 14  8 10  5
;
```

The following statements perform a multivariate analysis of variance and produce Figure 105.49 through Figure 105.52:

```
proc reg;
   model y1 y2=sexcode drug1 drug2 sexdrug1 sexdrug2;
   y1y2drug: mtest y1=y2, drug1,drug2;
   drugshow: mtest drug1, drug2 / print canprint;
run;
```

Figure 105.49 Multivariate Analysis of Variance: REG Procedure

The REG Procedure
Model: MODEL1
Dependent Variable: y1

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	316.00000	63.20000	12.04	<.0001
Error	18	94.50000	5.25000		
Corrected Total	23	410.50000			

Root MSE	2.29129	**R-Square**	0.7698
Dependent Mean	9.75000	**Adj R-Sq**	0.7058
Coeff Var	23.50039		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	9.75000	0.46771	20.85	<.0001
sexcode	1	0.16667	0.46771	0.36	0.7257
drug1	1	-2.75000	0.66144	-4.16	0.0006
drug2	1	-2.25000	0.66144	-3.40	0.0032
sexdrug1	1	-0.66667	0.66144	-1.01	0.3269
sexdrug2	1	-0.41667	0.66144	-0.63	0.5366

Figure 105.50 Multivariate Analysis of Variance: REG Procedure

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	69.33333	13.86667	2.19	0.1008
Error	18	114.00000	6.33333		
Corrected Total	23	183.33333			

Root MSE	2.51661	**R-Square**	0.3782
Dependent Mean	8.66667	**Adj R-Sq**	0.2055
Coeff Var	29.03782		

Figure 105.50 *continued*

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	8.66667	0.51370	16.87	<.0001
sexcode	1	0.16667	0.51370	0.32	0.7493
drug1	1	-1.41667	0.72648	-1.95	0.0669
drug2	1	-0.16667	0.72648	-0.23	0.8211
sexdrug1	1	-1.16667	0.72648	-1.61	0.1257
sexdrug2	1	-0.41667	0.72648	-0.57	0.5734

Figure 105.51 Multivariate Analysis of Variance: First Test

The REG Procedure
Model: MODEL1
Multivariate Test: y1y2drug

Multivariate Statistics and Exact F Statistics					
	S=1 M=0 N=8				
Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.28053917	23.08	2	18	<.0001
Pillai's Trace	0.71946083	23.08	2	18	<.0001
Hotelling-Lawley Trace	2.56456456	23.08	2	18	<.0001
Roy's Greatest Root	2.56456456	23.08	2	18	<.0001

The four multivariate test statistics are all highly significant, giving strong evidence that the coefficients of drug1 and drug2 are not the same across dependent variables y1 and y2.

Figure 105.52 Multivariate Analysis of Variance: Second Test

The REG Procedure
Model: MODEL1
Multivariate Test: drugshow

Error Matrix (E)	
94.5	76.5
76.5	114

Hypothesis Matrix (H)	
301	97.5
97.5	36.333333333

Figure 105.52 *continued*

	Canonical Correlation	Adjusted Canonical Correlation	Approximate Standard Error	Squared Canonical Correlation	Eigenvalues of Inv(E)*H = CanRsq/(1-CanRsq) Eigenvalue	Difference	Proportion	Cumulative
1	0.905903	0.899927	0.040101	0.820661	4.5760	4.5125	0.9863	0.9863
2	0.244371	.	0.210254	0.059717	0.0635		0.0137	1.0000

Test of H0: The canonical correlations in the current row and all that follow are zero					
	Likelihood Ratio	Approximate F Value	Num DF	Den DF	Pr > F
1	0.16862952	12.20	4	34	<.0001
2	0.94028273	1.14	1	18	0.2991

Multivariate Statistics and F Approximations					
	S=2 M=-0.5 N=7.5				
Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.16862952	12.20	4	34	<.0001
Pillai's Trace	0.88037810	7.08	4	36	0.0003
Hotelling-Lawley Trace	4.63953666	19.40	4	19.407	<.0001
Roy's Greatest Root	4.57602675	41.18	2	18	<.0001
NOTE: F Statistic for Roy's Greatest Root is an upper bound.					
NOTE: F Statistic for Wilks' Lambda is exact.					

The four multivariate test statistics are all highly significant, giving strong evidence that the coefficients of drug1 and drug2 are not zero for both dependent variables.

Autocorrelation in Time Series Data

When regression is performed on time series data, the errors might not be independent. Often errors are autocorrelated; that is, each error is correlated with the error immediately before it. Autocorrelation is also a symptom of systematic lack of fit. The DW option provides the Durbin-Watson d statistic to test that the autocorrelation is zero:

$$d = \frac{\sum_{i=2}^{n}(e_i - e_{i-1})^2}{\sum_{i=1}^{n} e_i^2}$$

The value of d is close to 2 if the errors are uncorrelated. The distribution of d is reported by Durbin and Watson (1951). Tables of the distribution are found in most econometrics textbooks, such as Johnston (1972) and Pindyck and Rubinfeld (1981).

The sample autocorrelation estimate is displayed after the Durbin-Watson statistic. The sample is computed as

$$r = \frac{\sum_{i=2}^{n} e_i e_{i-1}}{\sum_{i=1}^{n} e_i^2}$$

This autocorrelation of the residuals might not be a very good estimate of the autocorrelation of the true errors, especially if there are few observations and the independent variables have certain patterns. If there are missing observations in the regression, these measures are computed as though the missing observations did not exist.

Positive autocorrelation of the errors generally tends to make the estimate of the error variance too small, so confidence intervals are too narrow and true null hypotheses are rejected with a higher probability than the stated significance level. Negative autocorrelation of the errors generally tends to make the estimate of the error variance too large, so confidence intervals are too wide and the power of significance tests is reduced. With either positive or negative autocorrelation, least squares parameter estimates are usually not as efficient as generalized least squares parameter estimates. For more details, see Judge et al. (1985, Chapter 8) and the *SAS/ETS User's Guide*.

The following SAS statements request the DWPROB option for the U.S. population data (see Figure 105.53). If you use the DW option instead of the DWPROB option, then p-values are not produced.

```
proc reg data=USPopulation;
   model Population=Year YearSq / dwProb;
run;
```

Figure 105.53 Regression Using DW Option

The REG Procedure
Model: MODEL1
Dependent Variable: Population

Durbin-Watson D	1.191
Pr < DW	0.0050
Pr > DW	0.9950
Number of Observations	22
1st Order Autocorrelation	0.323

Computations for Ridge Regression and IPC Analysis

In ridge regression analysis, the crossproduct matrix for the independent variables is centered (the NOINT option is ignored if it is specified) and scaled to one on the diagonal elements. The ridge constant k (specified with the RIDGE= option) is then added to each diagonal element of the crossproduct matrix. The ridge regression estimates are the least squares estimates obtained by using the new crossproduct matrix.

Let $\mathbf{X}$ be an $n \times p$ matrix of the independent variables after centering the data, and let $\mathbf{Y}$ be an $n \times 1$ vector corresponding to the dependent variable. Let $\mathbf{D}$ be a $p \times p$ diagonal matrix with diagonal elements as in $\mathbf{X}'\mathbf{X}$. The ridge regression estimate corresponding to the ridge constant k can be computed as

$$\mathbf{D}^{-\frac{1}{2}}(\mathbf{Z}'\mathbf{Z} + k\mathbf{I}_p)^{-1}\mathbf{Z}'\mathbf{Y}$$

where $\mathbf{Z} = \mathbf{X}\mathbf{D}^{-\frac{1}{2}}$ and $\mathbf{I}_p$ is a $p \times p$ identity matrix.

For IPC analysis, the smallest m eigenvalues of $\mathbf{Z}'\mathbf{Z}$ (where m is specified with the PCOMIT= option) are omitted to form the estimates.

For information about ridge regression and IPC standardized parameter estimates, parameter estimate standard errors, and variance inflation factors, see Rawlings, Pantula, and Dickey (1998); Neter, Wasserman, and

Kutner (1990); Marquardt and Snee (1975). Unlike Rawlings, Pantula, and Dickey (1998), the REG procedure uses the mean squared errors of the submodels instead of the full model MSE to compute the standard errors of the parameter estimates.

Construction of Q-Q and P-P Plots

If a normal probability-probability or quantile-quantile plot for the variable x is requested, the n nonmissing values of x are first ordered from smallest to largest:

$$x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(n)}$$

If a Q-Q plot is requested (with a PLOT statement of the form PLOT *yvariable**NQQ.), the ith-ordered value $x_{(i)}$ is represented by a point with y-coordinate $x_{(i)}$ and x-coordinate $\Phi^{-1}\left(\frac{i-0.375}{n+0.25}\right)$, where $\Phi(\cdot)$ is the standard normal distribution.

If a P-P plot is requested (with a PLOT statement of the form PLOT *yvariable**NPP.), the ith-ordered value $x_{(i)}$ is represented by a point with y-coordinate $\frac{i}{n}$ and x-coordinate $\Phi\left(\frac{x_{(i)}-\mu}{\sigma}\right)$, where μ is the mean of the nonmissing x-values and σ is the standard deviation. If an x-value has multiplicity k (that is, $x_{(i)} = \cdots = x_{(i+k-1)}$), then only the point $\left(\Phi\left(\frac{x_{(i)}-\mu}{\sigma}\right), \frac{i+k-1}{n}\right)$ is displayed.

Computational Methods

The REG procedure first composes a crossproducts matrix. The matrix can be calculated from input data, reformed from an input correlation matrix, or read in from an SSCP data set. For each model, the procedure selects the appropriate crossproducts from the main matrix. The normal equations formed from the crossproducts are solved by using a sweep algorithm (Goodnight 1979). The method is accurate for data that are reasonably scaled and not too collinear.

The mechanism that PROC REG uses to check for singularity involves the diagonal (pivot) elements of $\mathbf{X}'\mathbf{X}$ as it is being swept. If a pivot is less than SINGULAR*CSS, then a singularity is declared and the pivot is not swept (where CSS is the corrected sum of squares for the regressor and SINGULAR is machine dependent but is approximately 1E–9 on most machines or reset in the PROC REG statement).

The sweep algorithm is also used in many places in the model-selection methods. The RSQUARE method uses the leaps-and-bounds algorithm by Furnival and Wilson (1974).

Computer Resources in Regression Analysis

The REG procedure is efficient for ordinary regression; however, requests for optional features can greatly increase the amount of time required.

The major computational expense in the regression analysis is the collection of the crossproducts matrix. For p variables and n observations, the time required is proportional to np^2. For each model run, PROC REG needs time roughly proportional to k^3, where k is the number of regressors in the model. Include an additional nk^2 for the R, CLM, or CLI option and another nk^2 for the INFLUENCE option.

Most of the memory that PROC REG needs to solve large problems is used for crossproducts matrices. PROC REG requires $4p^2$ bytes for the main crossproducts matrix plus $4k^2$ bytes for the largest model. If several output data sets are requested, memory is also needed for buffers.

See the section "Input Data Sets" on page 8782 for information about how to use TYPE=SSCP data sets to reduce computing time.

Displayed Output

Many of the more specialized tables are described in detail in previous sections. Most of the formulas for the statistics are in Chapter 4, "Introduction to Regression Procedures," while other formulas can be found in the section "Model Fit and Diagnostic Statistics" on page 8808 and the section "Influence Statistics" on page 8810.

The analysis-of-variance table includes the following:

- the Source of the variation, Model for the fitted regression, Error for the residual error, and C Total for the total variation after correcting for the mean. The Uncorrected Total Variation is produced when the NOINT option is used.
- the degrees of freedom (DF) associated with the source
- the Sum of Squares for the term
- the Mean Square, the sum of squares divided by the degrees of freedom
- the F Value for testing the hypothesis that all parameters are zero except for the intercept. This is formed by dividing the mean square for Model by the mean square for Error.
- the Prob>F, the probability of getting a greater F statistic than that observed if the hypothesis is true. This is the significance probability.

Other statistics displayed include the following:

- Root MSE is an estimate of the standard deviation of the error term. It is calculated as the square root of the mean square error.
- Dep Mean is the sample mean of the dependent variable.

- C.V. is the coefficient of variation, computed as 100 times Root MSE divided by Dep Mean. This expresses the variation in unitless values.
- R-square is a measure between 0 and 1 that indicates the portion of the (corrected) total variation that is attributed to the fit rather than left to residual error. It is calculated as SS(Model) divided by SS(Total). It is also called the *coefficient of determination*. It is the square of the multiple correlation—in other words, the square of the correlation between the dependent variable and the predicted values.
- Adj R-square, the adjusted R square, is a version of R square that has been adjusted for degrees of freedom. It is calculated as

  $$\bar{R}^2 = 1 - \frac{(n-i)(1-R^2)}{n-p}$$

  where i is equal to 1 if there is an intercept and 0 otherwise, n is the number of observations used to fit the model, and p is the number of parameters in the model.

The parameter estimates and associated statistics are then displayed, and they include the following:

- the Variable used as the regressor, including the name Intercept to represent the estimate of the intercept parameter
- the degrees of freedom (DF) for the variable. There is one degree of freedom unless the model is not full rank.
- the Parameter Estimate
- the Standard Error, the estimate of the standard deviation of the parameter estimate
- T for H0: Parameter=0, the t test that the parameter is zero. This is computed as the Parameter Estimate divided by the Standard Error.
- the Prob > |T|, the probability that a t statistic would obtain a greater absolute value than that observed given that the true parameter is zero. This is the two-tailed significance probability.

If model-selection methods other than NONE, RSQUARE, ADJRSQ, and CP are used, the analysis-of-variance table and the parameter estimates with associated statistics are produced at each step. Also displayed are the following:

- C(p), which is Mallows' C_p statistic
- bounds on the condition number of the correlation matrix for the variables in the model (Berk 1977)

After statistics for the final model are produced, the following is displayed when the method chosen is FORWARD, BACKWARD, or STEPWISE:

- a Summary table listing Step number, Variable Entered or Removed, Partial and Model R-square, and C(p) and F statistics

The RSQUARE method displays its results beginning with the model containing the fewest independent variables and producing the largest R square. Results for other models with the same number of variables are then shown in order of decreasing R square, and so on, for models with larger numbers of variables. The ADJRSQ and CP methods group models of all sizes together and display results beginning with the model having the optimal value of adjusted R square and C_p, respectively.

For each model considered, the RSQUARE, ADJRSQ, and CP methods display the following:

- Number in Model or IN, the number of independent variables used in each model
- R-square or RSQ, the squared multiple correlation coefficient

If the B option is specified, the RSQUARE, ADJRSQ, and CP methods produce the following:

- Parameter Estimates, the estimated regression coefficients

If the B option is not specified, the RSQUARE, ADJRSQ, and CP methods display the following:

- Variables in Model, the names of the independent variables included in the model

Plot Options Superseded by ODS Graphics

You can select one of the following three types of graphics in PROC REG: ODS, traditional, and line printer. ODS Graphics is the preferred method of creating graphs, superseding the other two. This section describes the options that are available on the PROC REG, PAINT, and PLOT statements for traditional and line printer graphics.

When ODS Graphics is enabled, you can use the PLOTS= option in the PROC REG statement to create plots by using ODS Graphics. For more information about ODS Graphics options see the PLOTS= option in the section "Syntax: REG Procedure" on page 8740.

If ODS Graphics is not enabled and you specify the LINEPRINTER option, line printer plots are produced; otherwise traditional graphics are produced.

Table 105.9 summarizes the *options* available in the PROC REG statement for line printer and traditional graphics.

Table 105.9 PROC REG Statement Traditional Graphics and Line Printer Options

Option	Description
ANNOTATE=	Specifies an annotation data set
GOUT=	Specifies the graphics catalog in which graphics output is saved
LINEPRINTER	Creates printer plots

The following *options* are used to produce line printer and traditional graphics:

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an input data set containing annotate variables, as described in *SAS/GRAPH: Reference*. You can use this data set to add features to the traditional graphics that you request with the PLOT statement. Features provided in this data set are applied to all plots produced in the current run of PROC REG. To add features to individual plots, use the ANNOTATE= option in the PLOT statement. This option cannot be used if the LINEPRINTER option is specified.

GOUT=*graphics-catalog*

specifies the graphics catalog in which traditional graphics output is saved. The default *graphics-catalog* is WORK.GSEG. The GOUT= option cannot be used if the LINEPRINTER option is specified.

LINEPRINTER | LP

creates printer plots. If you do not specify this option, requested plots are created on a high-resolution graphics device. See the PLOTS= option for information about using ODS graphics to create modern statistical graphics.

PAINT Statement

PAINT < *condition* | **ALLOBS** > < / *options* > **;**

PAINT < **STATUS** | **UNDO** > **;**

The PAINT statement is used with line printer plots. See the PLOTS= option for information about using ODS graphics to create modern statistical graphics.

The PAINT statement selects observations to be *painted* or highlighted in a scatter plot on line printer output; the PAINT statement is ignored if the LINEPRINTER option is not specified in the PROC REG statement.

All observations that satisfy *condition* are painted using some specific symbol. The PAINT statement does not generate a scatter plot and must be followed by a PLOT statement, which does generate a scatter plot. Several PAINT statements can be used before a PLOT statement, and all prior PAINT statement requests are applied to all later PLOT statements.

The PAINT statement lists the observation numbers of the observations selected, the total number of observations selected, and the plotting symbol used to paint the points.

On a plot, paint symbols take precedence over all other symbols. If any position contains more than one painted point, the paint symbol for the observation plotted last is used.

The PAINT statement cannot be used when a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set is used as the input data set for PROC REG. Also, the PAINT statement cannot be used for models with more than one dependent variable. Note that the syntax for the PAINT statement is the same as the syntax for the REWEIGHT statement.

Specifying Condition

Condition is used to select observations to be painted. The syntax of *condition* is

variable compare value

or

variable compare value *logical* *variable compare value*

where

variable is one of the following:

- a variable name in the input data set
- OBS**.**, which is the observation number
- *keyword***.**, where *keyword* is a keyword for a statistic requested in the OUTPUT statement

compare is an operator that compares *variable* to *value*. *Compare* can be any one of the following: <, <=, >, >=, =, ˆ=. The operators LT, LE, GT, GE, EQ, and NE, respectively, can be used instead of the preceding symbols. See the "Expressions" section in *SAS Programmers Guide: Essentials* for more information about comparison operators.

value gives an unformatted value of *variable*. Observations are selected to be painted if they satisfy the condition created by *variable compare value*. *Value* can be a number or a character string. If *value* is a character string, it must be eight characters or less and must be enclosed in quotes. In addition, *value* is case-sensitive. In other words, the statements

```
paint name='henry';
```

and

```
paint name='Henry';
```

are not the same.

logical is one of two logical operators. Either AND or OR can be used. To specify AND, use AND or the symbol &. To specify OR, use OR or the symbol |.

Here are some examples of the *variable compare value* form:

```
paint name='Henry';
paint residual.>=20;
paint obs.=99;
```

Here are some examples of the *variable compare value* *logical* *variable compare value* form:

```
paint name='Henry'|name='Mary';
paint residual.>=20 or residual.<=10;
paint obs.>=11 and residual.<=20;
```

Using ALLOBS

Instead of specifying *condition*, the ALLOBS option can be used to select all observations. This is most useful when you want to unpaint all observations. For example,

```
paint allobs / reset;
```

resets the symbols for all observations.

Options in the PAINT Statement

The following *options* can be used when either a condition is specified, the ALLOBS option is specified, or nothing is specified before the slash. If only an *option* is listed, the *option* applies to the observations selected in the previous PAINT statement, *not* to the observations selected by reapplying the condition from the previous PAINT statement. For example, in the statements

```
paint r.>0 / symbol='a';
reweight r.>0;
refit;
paint / symbol='b';
```

the second PAINT statement paints only those observations selected in the first PAINT statement. No additional observations are painted even if, after refitting the model, there are new observations that meet the condition in the first PAINT statement.

NOTE: Options are not available when either the UNDO or STATUS option is used.

You can specify the following *options* after a slash (/).

NOLIST

suppresses the display of the selected observation numbers. If the NOLIST option is not specified, a list of observations selected is written to the log. The list includes the observation numbers and painting symbol used to paint the points. The total number of observations selected to be painted is also shown.

RESET

changes the painting symbol to the current default symbol, effectively unpainting the observations selected. If you set the default symbol by using the SYMBOL= option in the PLOT statement, the RESET option in the PAINT statement changes the painting symbol to the symbol you specified. Otherwise, the default symbol of '1' is used.

SYMBOL='character'

specifies a painting symbol. If the SYMBOL= option is omitted, the painting symbol is either the one used in the most recent PAINT statement or, if there are no previous PAINT statements, the symbol '@'. For example,

```
paint / symbol='#';
```

changes the painting symbol for the observations selected by the most recent PAINT statement to '#'. As another example,

```
paint temp lt 22 / symbol='c';
```

changes the painting symbol to 'c' for all observations with TEMP<22. In general, the numbers 1, 2, . . . , 9 and the asterisk are not recommended as painting symbols. These symbols are used as default symbols in the PLOT statement, where they represent the number of replicates at a point. If

SYMBOL=‘’ is used, no painting is done in the current plot. If SYMBOL=‘ ’ is used, observations are painted with a blank and are no longer seen on the plot.

STATUS and UNDO

Instead of specifying *condition* or the ALLOBS option, you can use the STATUS or UNDO option as follows:

STATUS

lists (in the log) the observation number and plotting symbol of all currently painted observations.

UNDO

undoes changes made by the most recent PAINT statement. Observations might be, but are not necessarily, unpainted. For example:

```
paint obs. <=10 / symbol='a';
...other interactive statements
paint obs.=1 / symbol='b';
...other interactive statements
paint undo;
```

The last PAINT statement changes the plotting symbol used for observation 1 back to ‘a’. If the statement

```
paint / reset;
```

is used instead, observation 1 is unpainted.

PLOT Statement

PLOT < *yvariable* ∗ *xvariable* > < **=***symbol* > < . . . *yvariable* ∗ *xvariable* > < **=***symbol* > < / *options* > **;**

The PLOT statement is used with line printer and traditional graphics. See the PLOTS= option for information about using ODS graphics to create modern statistical graphics.

The PLOT statement in PROC REG displays scatter plots with *yvariable* on the vertical axis and *xvariable* on the horizontal axis. Line printer plots are generated if the LINEPRINTER option is specified in the PROC REG statement; otherwise, the traditional graphics are created. Points in line printer plots can be marked with *symbols*, while global graphics statements such as GOPTIONS and SYMBOL are used to enhance the traditional graphics. Note that the plots you request by using the PLOT statement are independent of the ODS graphical displays (see the section “ODS Graphics” on page 8852) that are available in PROC REG.

As with most other interactive statements, the PLOT statement implicitly refits the model. For example, if a PLOT statement is preceded by a REWEIGHT statement, the model is recomputed, and the plot reflects the new model.

If there are multiple MODEL statements preceding a PLOT statement, then the PLOT statement refers to the latest MODEL statement.

The PLOT statement cannot be used when a TYPE=CORR, TYPE=COV, or TYPE=SSCP data set is used as input to PROC REG.

You can specify several PLOT statements for each MODEL statement, and you can specify more than one plot in each PLOT statement.

Specifying Yvariables, Xvariables, and Symbol

More than one *yvariable**xvariable* pair can be specified to request multiple plots. The *yvariables* and *xvariables* can be as follows:

- any variables specified in the VAR or MODEL statement before the first RUN statement
- *keyword*., where *keyword* is a regression diagnostic statistic available in the OUTPUT statement (see Table 105.10). For example,

  ```
  plot predicted.*residual.;
  ```

  generates one plot of the predicted values by the residuals for each dependent variable in the MODEL statement. These statistics can also be plotted against any of the variables in the VAR or MODEL statements.
- the keyword OBS. (the observation number), which can be plotted against any of the preceding variables
- the keyword NPP. or NQQ., which can be used with any of the preceding variables to construct normal P-P or Q-Q plots, respectively (see the section "Construction of Q-Q and P-P Plots" on page 8831 for more information)
- *keywords* for model fit summary statistics available in the OUTEST= data set with _TYPE_= PARMS (see Table 105.10). A SELECTION= method (other than NONE) must be requested in the MODEL statement for these variables to be plotted. If one member of a *yvariable**xvariable* pair is from the OUTEST= data set, the other member must also be from the OUTEST= data set.

The OUTPUT statement and the OUTEST= option are not required when their *keywords* are specified in the PLOT statement.

The *yvariable* and *xvariable* specifications can be replaced by a set of variables and statistics enclosed in parentheses. When this occurs, all possible combinations of *yvariable* and *xvariable* are generated. For example, the following two statements are equivalent:

```
plot (y1 y2)*(x1 x2);
plot y1*x1 y1*x2 y2*x1 y2*x2;
```

The statement

```
plot;
```

is equivalent to respecifying the most recent PLOT statement without any options. However, the line printer options COLLECT, HPLOTS=, SYMBOL=, and VPLOTS=, described in the section "Line Printer Plots" on page 8847, apply across PLOT statements and remain in effect if they have been previously specified.

Options used for the traditional graphics are described in the following section; see "Line Printer Plots" on page 8847 for more information.

Traditional Graphics

The display of traditional graphics is described in the following paragraphs, the options are summarized in Table 105.10 and described in the section "Dictionary of PLOT Statement Options" on page 8843.

Several line printer statements and options are not supported for the traditional graphics. In particular the PAINT statement is disabled, as are the PLOT statement options CLEAR, COLLECT, HPLOTS=, NOCOLLECT, SYMBOL=, and VPLOTS=. To display more than one plot per page or to collect plots from multiple PLOT statements, use the PROC GREPLAY statement (see *SAS/GRAPH: Reference*). Also note that traditional graphics options are not recognized for line printer plots.

The fitted model equation and a label are displayed in the top margin of the plot; this display can be suppressed with the NOMODEL option. If the label is requested but cannot fit on one line, it is not displayed. The equation and label are displayed on one line when possible; if more lines are required, the label is displayed in the first line with the model equation in successive lines. If displaying the entire equation causes the plot to be unacceptably small, the equation is truncated. Table 105.11 lists options to control the display of the equation.

Four statistics are displayed by default in the right margin: the number of observations, R square, the adjusted R square, and the root mean square error. The display of these statistics can be suppressed with the NOSTAT option. You can specify other options to request the display of various statistics in the right margin; see Table 105.11.

A default reference line at zero is displayed if residuals are plotted. If the dependent variable is plotted against the independent variable in a simple linear regression model, the fitted regression line is displayed by default. Default reference lines can be suppressed with the NOLINE option; the lines are not displayed if the OVERLAY option is specified.

Specialized plots are requested with special options. For each coefficient, the RIDGEPLOT option plots the ridge estimates against the ridge values k; see the description of the RIDGEPLOT option in the section "Dictionary of PLOT Statement Options" on page 8843 for more details. The CONF option plots $100(1-\alpha)\%$ confidence intervals for the mean while the PRED option plots $100(1-\alpha)\%$ prediction intervals; see the description of these *options* in the section "Dictionary of PLOT Statement Options" on page 8843 for more details.

If a SELECTION= method is requested, the fitted model equation and the statistics displayed in the margin correspond to the selected model. For the ADJRSQ and CP methods, the selected model is treated as a submodel of the full model. If a CP.*NP. plot is requested, the CHOCKING= and CMALLOWS= options display model selection reference lines; see the descriptions of these *options* in the section "Dictionary of PLOT Statement Options" on page 8843 for more details.

PLOT Statement *variable* Keywords The following table lists the *keywords* available as PLOT statement *xvariables* and *yvariables*. All *keywords* have a trailing dot; for example, "*COOKD*." requests Cook's D statistic. Neither the OUTPUT statement nor the OUTEST= option needs to be specified.

Table 105.10 Keywords for PLOT Statement *xvariables*

Keyword	Description
Diagnostic Statistics	
COOKD.	Cook's D influence statistics
COVRATIO.	Standard influence of observation on covariance of betas

Table 105.10 *continued*

Keyword	Description
DFFITS.	Standard influence of observation on predicted value
H.	Leverage
LCL.	Lower bound of $100(1-\alpha)\%$ confidence interval for individual prediction
LCLM.	Lower bound of $100(1-\alpha)\%$ confidence interval for the mean of the dependent variable
PREDICTED. | PRED. | P.	Predicted values
PRESS.	Residuals from refitting the model with current observation deleted
RESIDUAL. | R.	Residuals
RSTUDENT.	Studentized residuals with the current observation deleted
STDI.	Standard error of the individual predicted value
STDP.	Standard error of the mean predicted value
STDR.	Standard error of the residual
STUDENT.	Residuals divided by their standard errors
UCL.	Upper bound of $100(1-\alpha)\%$ confidence interval for individual prediction
UCLM.	Upper bound of $100(1-\alpha)\%$ confidence interval for the mean of the dependent variables
Other Keywords Used with Diagnostic Statistics	
NPP.	Normal probability-probability plot
NQQ.	Normal quantile-quantile plot
OBS.	Observation number (cannot plot against OUTEST= statistics)
Model Fit Summary Statistics	
ADJRSQ.	Adjusted R-square
AIC.	Akaike's information criterion
BIC.	Sawa's Bayesian information criterion
CP.	Mallows' C_p statistic
EDF.	Error degrees of freedom
GMSEP.	Estimated MSE of prediction, assuming multivariate normality
IN.	Number of regressors in the model not including the intercept
JP.	Final prediction error
MSE.	Mean squared error
NP.	Number of parameters in the model (including the intercept)
PC.	Amemiya's prediction criterion
RMSE.	Root MSE
RSQ.	R-square
SBC.	SBC statistic
SP.	SP statistic
SSE.	Error sum of squares

Summary of PLOT Statement Graphics Options Table 105.11 summarizes the *options* available in the PLOT statement. These *options* are available unless the LINEPRINTER option is specified in the PROC

REG statement. For complete descriptions, see the section "Dictionary of PLOT Statement Options" on page 8843.

Table 105.11 Traditional Graphics Options

Option	Description
General Graphics Options	
ANNOTATE= *SAS-data-set*	Specifies the annotate data set
CHOCKING=*color*	Requests a reference line for C_p model selection criteria
CMALLOWS=*color*	Requests a reference line for the C_p model selection criterion
CONF	Requests plots of $100(1-\alpha)\%$ confidence intervals for the mean
DESCRIPTION= '*string*'	Specifies a description for graphics catalog member
NAME='*string*'	Names the plot in the graphics catalog
OVERLAY	Overlays plots from the same model
PRED	Requests plots of $100(1-\alpha)\%$ prediction intervals for individual responses
RIDGEPLOT	Requests the ridge trace for ridge regression
Axis and Legend Options	
LEGEND=*LEGENDn*	Specifies LEGEND statement to be used
NOLEGEND	Suppresses display of the legend
HAXIS=*values*	Specifies tick mark values for horizontal axis
VAXIS=*values*	Specifies tick mark values for vertical axis
Reference Line Options	
HREF=*values*	Specifies reference lines perpendicular to horizontal axis
LHREF=*linetype*	Specifies line style for HREF= lines
LLINE=*linetype*	Specifies line style for lines displayed by default
LVREF=*linetype*	Specifies line style for VREF= lines
NOLINE	Suppresses display of any default reference line
VREF=*values*	Specifies reference lines perpendicular to vertical axis
Color Options	
CAXIS=*color*	Specifies color for axis line and tick marks
CFRAME=*color*	Specifies color for frame
CHREF=*color*	Specifies color for HREF= lines
CLINE=*color*	Specifies color for lines displayed by default
CTEXT=*color*	Specifies color for text
CVREF=*color*	Specifies color for VREF= lines
Options for Displaying the Fitted Model Equation	
MODELFONT=*font*	Specifies font of model equation and model label
MODELHT=*value*	Specifies text height of model equation and model label
MODELLAB='*label*'	Specifies model label
NOMODEL	Suppresses display of the fitted model and the label
Options for Displaying Statistics in the Plot Margin	
AIC	Displays Akaike's information criterion
BIC	Displays Sawa's Bayesian information criterion

Table 105.11 *continued*

Option	Description
CP	Displays Mallows' C_p statistic
EDF	Displays the error degrees of freedom
GMSEP	Displays the estimated MSE of prediction assuming multivariate normality
IN	Displays the number of regressors in the model not including the intercept
JP	Displays the J_p statistic
MSE	Displays the mean squared error
NOSTAT	Suppresses display of the default statistics: the number of observations, R-square, adjusted R-square, and root mean square error
NP	Displays the number of parameters in the model including the intercept, if any
PC	Displays the PC statistic
SBC	Displays the SBC statistic
SP	Displays the S_p statistic
SSE	Displays the error sum of squares
STATFONT=*font*	Specifies font of text displayed in the margin
STATHT=*value*	Specifies height of text displayed in the margin

Dictionary of PLOT Statement Options The following entries describe the PLOT statement *options* in detail. Note that these *options* are available unless you specify the LINEPRINTER option in the PROC REG statement.

AIC

displays Akaike's information criterion in the plot margin.

ANNOTATE=*SAS-data-set*

ANNO=*SAS-data-set*

specifies an input data set that contains appropriate variables for annotation. This applies only to displays created with the current PLOT statement. See *SAS/GRAPH: Reference* for more information.

BIC

displays Sawa's Bayesian information criterion in the plot margin.

CAXIS=*color*

CAXES=*color*

CA=*color*

specifies the color for the axes, frame, and tick marks.

CFRAME=*color*

CFR=*color*

specifies the color for filling the area enclosed by the axes and the frame.

CHOCKING=*color*

requests reference lines corresponding to the equations $C_p = p$ and $C_p = 2p - p_{full}$, where p_{full} is the number of parameters in the full model (excluding the intercept) and p is the number of parameters in the subset model (including the intercept). The *color* must be specified; the $C_p = p$ line is solid and the $C_p = 2p - p_{full}$ line is dashed. Only PLOT statements of the form PLOT CP.*NP. produce these lines.

For the purpose of parameter estimation, Hocking (1976) suggests selecting a model where $C_p \leq 2p - p_{full}$. For the purpose of prediction, Hocking suggests the criterion $C_p \leq p$. You can request the single reference line $C_p = p$ with the CMALLOWS= option. If, for example, you specify both CHOCKING=RED and CMALLOWS=BLUE, then the $C_p = 2p - p_{full}$ line is red and the $C_p = p$ line is blue.

CHREF=*color*

CH=*color*

specifies the color for lines requested with the HREF= option.

CLINE=*color*

CL=*color*

specifies the color for lines displayed by default. See the NOLINE option for details.

CMALLOWS=*color*

requests a $C_p = p$ reference line, where p is the number of parameters (including the intercept) in the subset model. The *color* must be specified; the line is solid. Only PLOT statements of the form PLOT CP.*NP. produce this line.

Mallows (1973) suggests that all subset models with C_p small and near p be considered for further study. See the CHOCKING= option for related model-selection criteria.

CONF

is a *keyword* used as a shorthand option to request plots that include $(100 - \alpha)\%$ confidence intervals for the mean response. The ALPHA= option in the PROC REG or MODEL statement selects the significance level α, which is 0.05 by default. The CONF option is valid for simple regression models only, and is ignored for plots where confidence intervals are inappropriate. The CONF option replaces the CONF95 option; however, the CONF95 option is still supported when the ALPHA= option is not specified. The OVERLAY option is ignored when the CONF option is specified.

CP

displays Mallows' C_p statistic in the plot margin.

CTEXT=*color*

CT=*color*

specifies the color for text including tick mark labels, axis labels, the fitted model label and equation, the statistics displayed in the margin, and legends.

CVREF=*color*

CV=*color*

specifies the color for lines requested with the VREF= option.

DESCRIPTION='*string* '

DESC='*string* '

specifies a descriptive string, up to 40 characters, that appears in the description field of the PROC GREPLAY master menu.

EDF

displays the error degrees of freedom in the plot margin.

GMSEP

displays the estimated mean square error of prediction in the plot margin. Note that the estimate is calculated under the assumption that both independent and dependent variables have a multivariate normal distribution.

HAXIS=*values*

HA=*values*

specifies tick mark values for the horizontal axis.

HREF=*values*

specifies where reference lines perpendicular to the horizontal axis are to appear.

IN

displays the number of regressors in the model (not including the intercept) in the plot margin.

JP

displays the J_p statistic in the plot margin.

LEGEND=LEGEND*n*

specifies the LEGEND*n* statement to be used. The LEGEND*n* statement is a global graphics statement; see *SAS/GRAPH: Reference* for more information.

LHREF=*linetype*

LH=*linetype*

specifies the line style for lines requested with the HREF= option. The default *linetype* is 2. Note that LHREF=1 requests a solid line. See *SAS/GRAPH: Reference* for a table of available line types.

LLINE=*linetype*

LL=*linetype*

specifies the line style for reference lines displayed by default; see the NOLINE option for details. The default *linetype* is 2. Note that LLINE=1 requests a solid line.

LVREF=*linetype*

LV=*linetype*

specifies the line style for lines requested with the VREF= option. The default *linetype* is 2. Note that LVREF=1 requests a solid line.

MODELFONT=*font*
specifies the font used for displaying the fitted model label and the fitted model equation. See *SAS/GRAPH: Reference* for tables of software fonts.

MODELHT=*height*
specifies the text height for the fitted model label and the fitted model equation.

MODELLAB='*label* **'**
specifies the label to be displayed with the fitted model equation. By default, no label is displayed. If the label does not fit on one line, it is not displayed. See the section "Traditional Graphics" on page 8840 for more information.

MSE
displays the mean squared error in the plot margin.

NAME='*string* **'**
specifies a descriptive string, up to eight characters, that appears in the name field of the PROC GREPLAY master menu. The default *string* is REG.

NOLEGEND
suppresses the display of the legend.

NOLINE
suppresses the display of default reference lines. A default reference line at zero is displayed if residuals are plotted. If the dependent variable is plotted against the independent variable in a simple regression model, then the fitted regression line is displayed by default. Default reference lines are not displayed if the OVERLAY option is specified.

NOMODEL
suppresses the display of the fitted model equation.

NOSTAT
suppresses the display of statistics in the plot margin. By default, the number of observations, R-square, adjusted R-square, and root MSE are displayed.

NP
displays the number of regressors in the model including the intercept, if any, in the plot margin.

OVERLAY
overlays all plots specified in the PLOT statement from the same model on one set of axes. The variables for the first plot label the axes. The procedure automatically scales the axes to fit all of the variables unless the HAXIS= or VAXIS= option is used. Default reference lines are not displayed. A default legend is produced; the LEGEND= option can be used to customize the legend.

PC
displays the PC statistic in the plot margin.

PRED

is a *keyword* used as a shorthand option to request plots that include $(100 - \alpha)\%$ prediction intervals for individual responses. The ALPHA= option in the PROC REG or MODEL statement selects the significance level α, which is 0.05 by default. The PRED option is valid for simple regression models only, and is ignored for plots where prediction intervals are inappropriate. The PRED option replaces the PRED95 option; however, the PRED95 option is still supported when the ALPHA= option is not specified. The OVERLAY option is ignored when the PRED option is specified.

RIDGEPLOT

creates overlaid plots of ridge estimates against ridge values for each coefficient. The points corresponding to the estimates of each coefficient in the plot are connected by lines. For ridge estimates to be computed and plotted, the OUTEST= option must be specified in the PROC REG statement, and the RIDGE=list must be specified in either the PROC REG or MODEL statement.

SBC

displays the SBC statistic in the plot margin.

SP

displays the S_p statistic in the plot margin.

SSE

displays the error sum of squares in the plot margin.

STATFONT=*font*

specifies the font used for displaying the statistics that appear in the plot margin. See *SAS/GRAPH: Reference* for tables of software fonts.

STATHT=*height*

specifies the text height of the statistics that appear in the plot margin.

USEALL

specifies that predicted values at data points with missing dependent variable(s) be included on appropriate plots. By default, only points used in constructing the SSCP matrix appear on plots.

VAXIS=*values*

VA=*values*

specifies tick mark values for the vertical axis.

VREF=*values*

specifies where reference lines perpendicular to the vertical axis are to appear.

Line Printer Plots

Line printer plots are requested with the LINEPRINTER option in the PROC REG statement. Points in line printer plots can be marked with *symbols*, which can be specified as a single character enclosed in quotes or the name of any variable in the input data set.

If a character variable is used for the symbol, the first (leftmost) nonblank character in the formatted value of the variable is used as the plotting symbol. If a character in quotes is specified, that character becomes the plotting symbol. If a character is used as the plotting symbol, and if there are different plotting symbols needed at the same point, the symbol '?' is used at that point.

If an unformatted numeric variable is used for the symbol, the symbols '1', '2', . . . , '9' are used for variable values 1, 2, . . . , 9. For noninteger values, only the integer portion is used as the plotting symbol. For values of 10 or greater, the symbol '*' is used. For negative values, a '?' is used. If a numeric variable is used, and if there is more than one plotting symbol needed at the same point, the sum of the variable values is used at that point. If the sum exceeds 9, the symbol '*' is used.

If a symbol is not specified, the number of replicates at the point is displayed. The symbol '*' is used if there are 10 or more replicates.

If the LINEPRINTER option is used, you can specify the following options in the PLOT statement after a slash (/):

CLEAR

clears any collected scatter plots before plotting begins but does not turn off the COLLECT option. Use this option when you want to begin a new collection with the plots in the current PLOT statement. For more information about collecting plots, see the COLLECT and NOCOLLECT options in this section.

COLLECT

specifies that plots begin to be collected from one PLOT statement to the next and that subsequent plots show an overlay of all collected plots. This option enables you to overlay plots before and after changes to the model or to the data used to fit the model. Plots collected before changes are unaffected by the changes and can be overlaid on later plots. You can request more than one plot with this option, and you do not need to request the same number of plots in subsequent PLOT statements. If you specify an unequal number of plots, plots in corresponding positions are overlaid. For example, the statements

```
plot residual.*predicted. y*x / collect;
run;
```

produce two plots. If these statements are then followed by

```
plot residual.*x;
run;
```

two plots are again produced. The first plot shows residual against X values overlaid on residual against predicted values. The second plot is the same as that produced by the first PLOT statement.

Axes are scaled for the first plot or plots collected. The axes are not rescaled as more plots are collected.

Once specified, the COLLECT option remains in effect until the NOCOLLECT option is specified.

HPLOTS=*number*

sets the number of scatter plots that can be displayed across the page. The procedure begins with one plot per page. The value of the HPLOTS= option remains in effect until you change it in a later PLOT statement. See the VPLOTS= option for an example.

NOCOLLECT

specifies that the collection of scatter plots ends after adding the plots in the current PLOT statement. PROC REG starts with the NOCOLLECT option in effect. After you specify the NOCOLLECT option, any following PLOT statement produces a new plot that contains only the plots requested by that PLOT statement.

For more information, see the COLLECT option.

OVERLAY

enables requested scatter plots to be superimposed. The axes are scaled so that points on all plots are shown. If the HPLOTS= or VPLOTS= option is set to more than one, the overlaid plot occupies the first position on the page. The OVERLAY option is similar to the COLLECT option in that both options produce superimposed plots. However, OVERLAY superimposes only the plots in the associated PLOT statement; COLLECT superimposes plots across PLOT statements. The OVERLAY option can be used when the COLLECT option is in effect.

SYMBOL='*character***'**

changes the default plotting symbol used for all scatter plots produced in the current and in subsequent PLOT statements. Both SYMBOL='' and SYMBOL=' ' are allowed.

If the SYMBOL= option has not been specified, the default symbol is '1' for positions with one observation, '2' for positions with two observations, and so on. For positions with more than 9 observations, '*' is used. The SYMBOL= option (or a plotting symbol) is needed to avoid any confusion caused by this default convention. Specifying a particular symbol is especially important when either the OVERLAY or COLLECT option is being used.

If you specify the SYMBOL= option and use a number for *character*, that number is used for all points in the plot. For example, the statement

```
plot y*x / symbol='1';
```

produces a plot with the symbol '1' used for all points.

If you specify a plotting symbol and the SYMBOL= option, the plotting symbol overrides the SYMBOL= option. For example, in the statements

```
plot y*x y*v='.' / symbol='*';
```

the symbol used for the plot of Y against X is '*', and a '.' is used for the plot of Y against V.

If a paint symbol is defined with a PAINT statement, the paint symbol takes precedence over both the SYMBOL= option and the default plotting symbol for the PLOT statement.

VPLOTS=*number*

sets the number of scatter plots that can be displayed down the page. The procedure begins with one plot per page. The value of the VPLOTS= option remains in effect until you change it in a later PLOT statement.

For example, to specify a total of six plots per page, with two rows of three plots, use the HPLOTS= and VPLOTS= options as follows:

```
plot y1*x1 y1*x2 y1*x3 y2*x1 y2*x2 y2*x3 /
     hplots=3 vplots=2;
run;
```

ODS Table Names

PROC REG assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in the following table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 105.12 ODS Tables Produced by PROC REG

ODS Table Name	Description	Statement	Option
ACovEst	Consistent covariance of estimates matrix	MODEL	ALL, ACOV
ACovTestANOVA	Test ANOVA using ACOV estimates	TEST	ACOV (MODEL statement)
ANOVA	Model ANOVA table	MODEL	Default
CanCorr	Canonical correlations for hypothesis combinations	MTEST	CANPRINT
CollinDiag	Collinearity Diagnostics table	MODEL	COLLIN
CollinDiagNoInt	Collinearity Diagnostics for no intercept model	MODEL	COLLINOINT
ConditionBounds	Bounds on condition number	MODEL	(SELECTION=BACKWARD | FORWARD | STEPWISE | MAXR | MINR) and DETAILS
Corr	Correlation matrix for analysis variables	PROC REG	ALL, CORR
CorrB	Correlation of estimates	MODEL	CORRB
CovB	Covariance of estimates	MODEL	COVB
CrossProducts	Bordered model **X'X** matrix	MODEL	ALL, XPX
DWStatistic	Durbin-Watson statistic	MODEL	ALL, DW
DependenceEquations	Linear dependence equations	MODEL	Default if needed
Eigenvalues	MTest eigenvalues	MTEST	CANPRINT
Eigenvectors	MTest eigenvectors	MTEST	CANPRINT
EntryStatistics	Entry statistics for selection methods	MODEL	(SELECTION=BACKWARD | FORWARD | STEPWISE | MAXR | MINR) and DETAILS
ErrorPlusHypothesis	MTest error plus hypothesis matrix **H**+**E**	MTEST	PRINT
ErrorSSCP	MTest error matrix **E**	MTEST	PRINT

Table 105.12 *continued*

ODS Table Name	Description	Statement	Option
FitStatistics	Model fit statistics	MODEL	Default
HypothesisSSCP	MTest hypothesis matrix	MTEST	PRINT
InvMTestCov	Inv(**L** Ginv(**X**'**X**) **L**') and Inv(**Lb-c**)	MTEST	DETAILS
InvTestCov	Inv(**L** Ginv(**X**'**X**) **L**') and Inv(**Lb-c**)	TEST	PRINT
InvXPX	Bordered **X**'**X** inverse matrix	MODEL	I
MTestCov	**L** Ginv(**X**'**X**) **L**' and **Lb-c**	MTEST	DETAILS
MTransform	MTest matrix **M**, across dependents	MTEST	DETAILS
MultStat	Multivariate test statistics	MTEST	Default
NObs	Number of observations		Default
OutputStatistics	Output statistics table	MODEL	ALL, CLI, CLM, INFLUENCE, P, R
PartialPlotData	Partial regression leverage data	MODEL	PARTIALDATA
ParameterEstimates	Model parameter estimates	MODEL	Default if SELECTION= is not specified
RemovalStatistics	Removal statistics for selection methods	MODEL	(SELECTION=BACKWARD | STEPWISE | MAXR | MINR) and DETAILS
ResidualStatistics	Residual statistics and PRESS statistic	MODEL	ALL, CLI, CLM, INFLUENCE, P, R
SelParmEst	Parameter estimates for selection methods	MODEL	SELECTION=BACKWARD | FORWARD | STEPWISE | MAXR | MINR
SelectionSummary	Selection summary for FORWARD, BACKWARD, and STEPWISE methods	MODEL	SELECTION=BACKWARD | FORWARD | STEPWISE
SeqParmEst	Sequential parameter estimates	MODEL	SEQB
SimpleStatistics	Simple statistics for analysis variables	PROC REG	ALL, SIMPLE
SpecTest	White's heteroscedasticity test	MODEL	ALL, SPEC
SubsetSelSummary	Selection summary for RSQUARE, ADJRSQ, and CP methods	MODEL	SELECTION=RSQUARE | ADJRSQ | CP
TestANOVA	Test ANOVA table	TEST	Default
TestCov	**L** Ginv(**X**'**X**) **L**' and **Lb-c**	TEST	PRINT
USSCP	Uncorrected SSCP matrix for analysis variables	PROC REG	ALL, USSCP

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

The following sections describe the ODS graphical displays produced by PROC REG.

Diagnostics Panel

The "Diagnostics Panel" provides a display that you can use to get an overall assessment of your model. See Figure 105.9 for an example.

The panel contains the following plots:

- residuals versus the predicted values
- externally studentized residuals (RSTUDENT) versus the predicted values
- externally studentized residuals versus the leverage
- normal quantile-quantile plot (Q-Q plot) of the residuals
- dependent variable values versus the predicted values
- Cook's D versus observation number
- histogram of the residuals
- "Residual-Fit" (or RF) plot consisting of side-by-side quantile plots of the centered fit and the residuals
- box plot of the residuals if you specify the STATS=NONE suboption

Patterns in the plots of residuals or studentized residuals versus the predicted values, or spread of the residuals being greater than the spread of the centered fit in the RF plot, are indications of an inadequate model. Patterns in the spread about the 45-degree reference line in the plot of the dependent variable values versus the predicted values are also indications of an inadequate model.

The Q-Q plot, residual histogram, and box plot of the residuals are useful for diagnosing violations of the normality and homoscedasticity assumptions. If the data in a Q-Q plot come from a normal distribution, the points will cluster tightly around the reference line. A normal density is overlaid on the residual histogram to help in detecting departures form normality.

Following Rawlings, Pantula, and Dickey (1998), reference lines are shown on the relevant plots to identify observations deemed outliers or influential. Observations whose externally studentized residual magnitudes exceed 2 are deemed outliers. Observations whose leverage value exceeds $2p/n$ or whose Cook's D value

exceeds $4/n$ are deemed influential (p is the number of regressors including the intercept, and n is the number of observations used in the analysis). If you specify the LABEL suboption of the PLOTS=DIAGNOSTICS option, then the points deemed outliers or influential are labeled on the appropriate plots.

Fit statistics are shown in the lower right of the plot and can be customized or suppressed by using the STATS= suboption of the PLOTS=DIAGNOSTICS option.

Residuals by Regressor Plots

Panels of plots of the residuals versus each of the regressors in the model are produced by default. Patterns in these plots are indications of an inadequate model. To help in detecting patterns, you can use the SMOOTH= suboption of the PLOTS=RESIDUALS option to add loess fits to these residual plots. See Figure 105.1.7 for an example.

Fit and Prediction Plots

A fit plot consisting of a scatter plot of the data overlaid with the regression line, as well as confidence and prediction limits, is produced for models depending on a single regressor. Fit statistics are shown to the right of the plot and can be customized or suppressed by using the STATS= suboption of the PLOTS=FIT option.

When a model contains more than one regressor, a fit plot is not appropriate. However, if all the regressors in the model are transformations of a single variable in the input data set, then you can request a scatter plot of the dependent variable overlaid with a fit line and confidence and prediction limits versus this variable. You can also plot residuals versus this variable. You request these plots, shown in a panel, with the PLOTS=PREDICTION option. See Figure 105.14 for an example.

Influence Plots

In addition to the "Cook's *D* Plot" and the "RStudent By Leverage Plot," you can request plots of the DFBETAS and DFFITS statistics versus observation number by using the PLOTS=DFBETAS and PLOTS=DFFITS options. You can also obtain partial regression leverage plots by using the PLOTS=PARTIAL option. See the section "Influence Statistics" on page 8810 for examples of these plots and details about their interpretation.

Ridge and VIF Plots

When you use ridge regression, you can request plots of the variance inflation factor (VIF) values and standardized ridge estimates by ridge values for each coefficient with the PLOTS=RIDGE option. See Example 105.5 for examples.

Variable Selection Plots

When you request variable selection by using the SELECTION= option in the MODEL statement, you can request plots of fit criteria for the models examined by using the PLOTS=CRITERIA option. The fit criteria are displayed versus the step number for the FORWARD, BACKWARD, and STEPWISE selection methods and the step at which the optimal value of each criterion is obtained is indicated using a "Star" marker. For the all-subset-based selection methods (SELECTION=RSQUARE | ADJRSQ | CP), the fit criteria are displayed versus the number of parameters in the model.

The criteria are shown in a panel, but you can use the UNPACK suboption of the PLOTS=CRITERIA option to obtain separate plots for each criterion. You can also use the LABEL suboption of the PLOTS=CRITERIA option to request that optimal models be labeled on the plots. Example 105.2 provides several examples.

Heat Maps

PROC REG can produce either fit and residual scatter plots for smaller data sets or heat maps for larger data sets. The global plot option MAXPOINTS=*max heat-max* controls which of these are produced. When the number of points exceeds the value of *max* but does not exceed the value of *heat-max* divided by the number of independent variables, heat maps are displayed instead of scatter plots for the fit and residual plots. All other graphs are suppressed when the number of points exceeds *max*. The default is MAXPOINTS=5000 150000. These cutoffs are ignored if you specify MAXPOINTS=NONE. The following statements create both scatter plots and heat maps with artificial data:

```
data x;
   do i = 1 to 25000;
      x = 2 * normal(104);
      y = x + sin(x * 2) + 3 * normal(104);
      output;
   end;
run;

ods graphics on;

proc reg data=x plots(maxpoints=30000);
   model y = x;
quit;

proc reg data=x;
   model y = x;
quit;
```

Scatter plots are displayed in Output 105.54, and heat maps are displayed in Output 105.55.

Figure 105.54 Scatter Plot

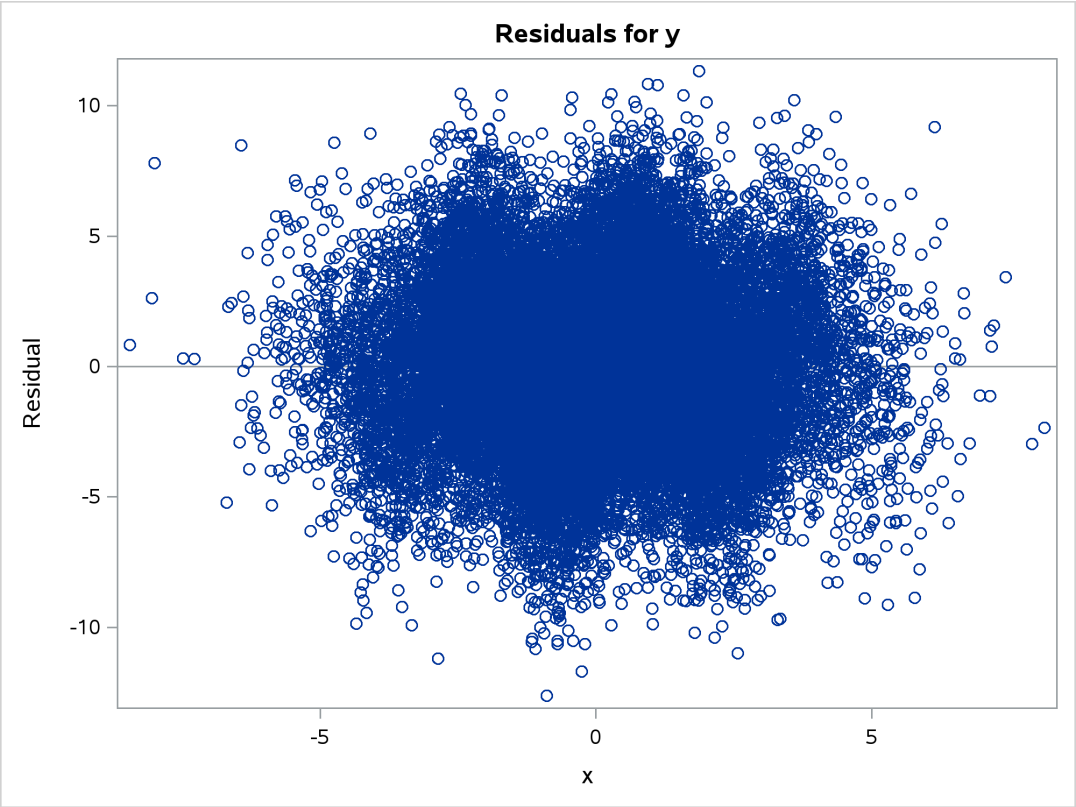

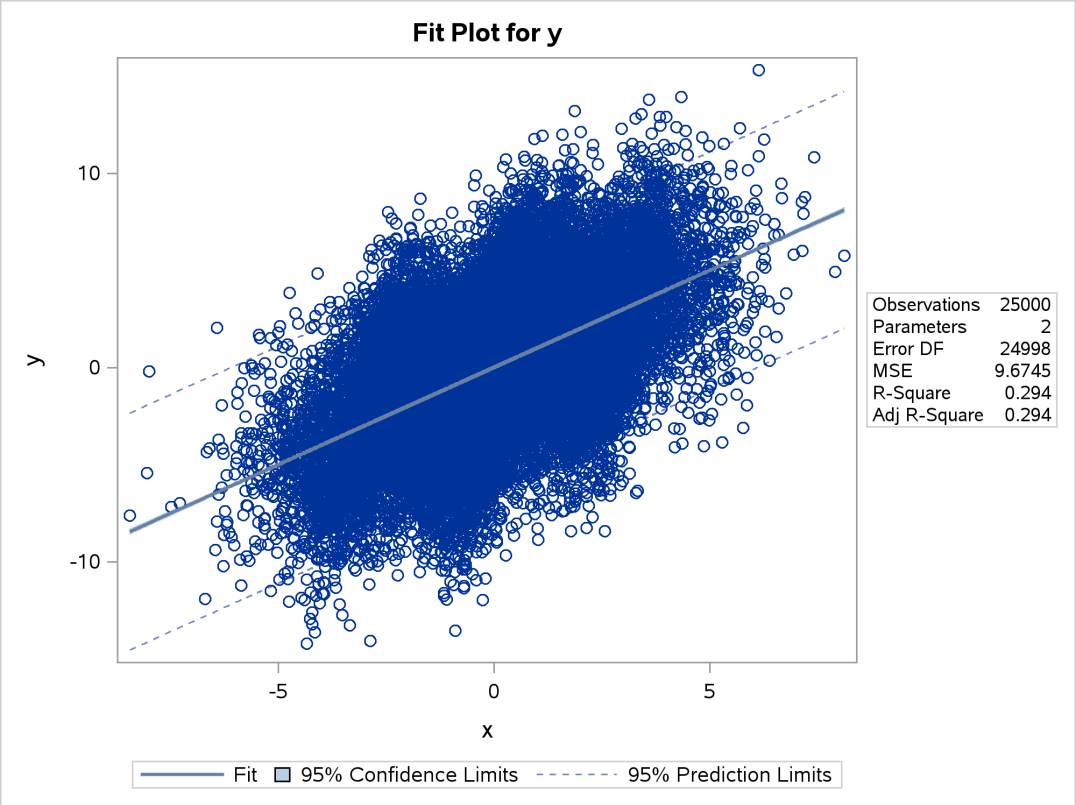

Figure 105.55 Heat Map

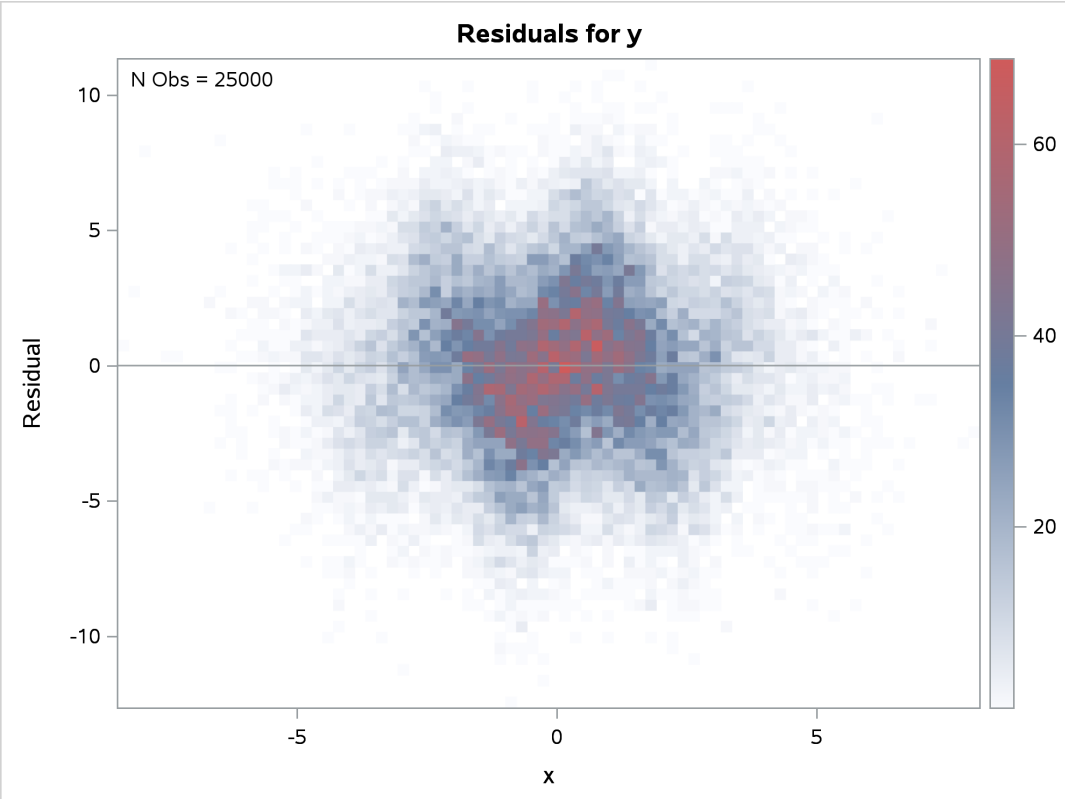

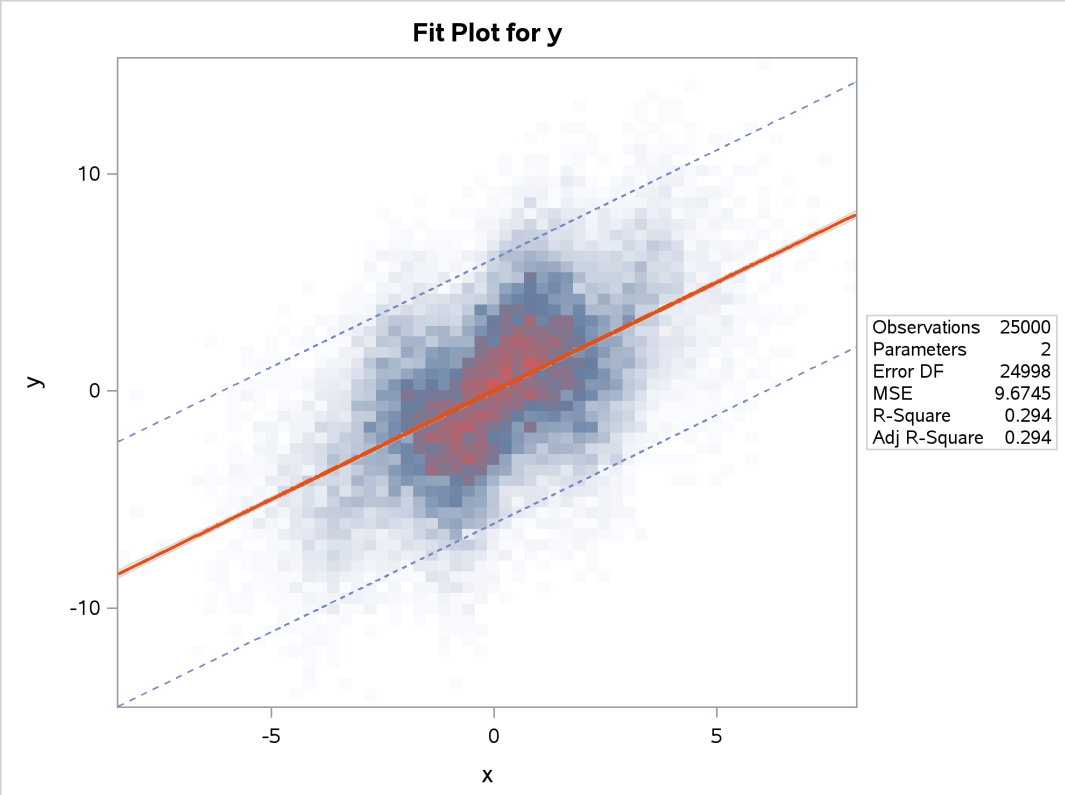

The heat maps show more clearly that the sine function is not fit well by the linear fit function.

ODS Graph Names

PROC REG assigns a name to each graph it creates using ODS. You can use these names to reference the graphs when using ODS. The names are listed in Table 105.13.

Table 105.13 ODS Graphical Displays Produced by PROC REG

ODS Graph Name	Plot Description	PLOTS Option
AdjrsqPlot	Adjusted R-square statistic for models examined doing variable selection	ADJRSQ
AICPlot	AIC statistic for models examined doing variable selection	AIC
BICPlot	BIC statistic for models examined doing variable selection	BIC
CooksDChart	Cook's *D* chart	PLOTS=RESIDUALCHART(UNPACK)
CooksDPlot	Cook's *D* statistic versus observation number	COOKSD
CPPlot	C_p statistic for models examined doing variable selection	CP
DFFITSPlot	DFFITS statistics versus observation number	DFFITS
DFBETASPanel	Panel of DFBETAS statistics versus observation number	DFBETAS
DFBETASPlot	DFBETAS statistics versus observation number	DFBETAS(UNPACK)
DiagnosticsPanel	Panel of fit diagnostics	DIAGNOSTICS
FitPlot	Regression line, confidence limits, and prediction limits overlaid on a scatter plot of the data	FIT, MAXPOINTS=*max* not exceeded
FitPlot	Regression line overlaid on a heat map of the data	FIT, MAXPOINTS=*max* exceeded
ObservedByPredicted	Dependent variable versus predicted values	OBSERVEDBYPREDICTED
PartialPlot	Partial regression plot	PARTIAL
PredictionPanel	Panel of residuals and fit versus specified variable	PREDICTIONS
PredictionPlot	Regression line, confidence limits, and prediction limits versus specified variable	PREDICTIONS(UNPACK)
PredictionResidualPlot	Residuals versus specified variable	PREDICTIONS(UNPACK)
QQPlot	Normal quantile plot of residuals	QQ
ResidualBoxPlot	Box plot of residuals	BOXPLOT
ResidualByPredicted	Residuals versus predicted values	RESIDUALBYPREDICTED
ResidualHistogram	Histogram of fit residuals	RESIDUALHISTOGRAM

Table 105.13 *continued*

ODS Graph Name	Plot Description	PLOTS Option
ResidualPlot	Scatter plot of residuals versus regressor	RESIDUALS, MAXPOINTS=*max* not exceeded
ResidualPlot	Heat map of residuals versus regressor	RESIDUALS, MAXPOINTS=*max* exceeded
RFPlot	Side-by-side plots of quantiles of centered fit and residuals	RF
RidgePanel	Plot of VIF and ridge traces	RIDGE
RidgePlot	Plot of ridge traces	RIDGE(UNPACK)
RSquarePlot	R-square statistic for models examined doing variable selection	RSQUARE
RStudentByLeverage	Studentized residuals versus leverage	RSTUDENTBYLEVERAGE
RStudentByPredicted	Studentized residuals versus predicted values	RSTUDENTBYPREDICTED
SBCPlot	SBC statistic for models examined doing variable selection	SBC
SelectionCriterionPanel	Panel of fit statistics for models examined doing variable selection	CRITERIA
StudResCooksDChart	Studentized residual and Cook's *D* chart	PLOTS=RESIDUALCHART
StudentResChart	Studentized residual chart	PLOTS=RESIDUALCHART(UNPACK)
VIFPlot	Plot of VIF traces	RIDGE(UNPACK)

Examples: REG Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 105.1: Modeling Salaries of Major League Baseball Players

(View the complete code for this example (regex1.sas) in the example repository.)

This example features the use of ODS Graphics in the process of building models by using the REG procedure and highlights the use of fit and influence diagnostics.

The Sashelp.Baseball data set contains salary and performance information for Major League Baseball players who played at least one game in both the 1986 and 1987 seasons, excluding pitchers. The salaries (*Sports Illustrated,* April 20, 1987) are for the 1987 season and the performance measures are from 1986 (Collier Books, *The 1987 Baseball Encyclopedia Update*). The following step displays in Output 105.1.1 the variables in the data set:

```
proc contents varnum data=sashelp.baseball;
   ods select position;
run;
```

Output 105.1.1 Sashelp.Baseball Data Set

The CONTENTS Procedure

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	Name	Char	18	Player's Name
2	Team	Char	14	Team at the End of 1986
3	nAtBat	Num	8	Times at Bat in 1986
4	nHits	Num	8	Hits in 1986
5	nHome	Num	8	Home Runs in 1986
6	nRuns	Num	8	Runs in 1986
7	nRBI	Num	8	RBIs in 1986
8	nBB	Num	8	Walks in 1986
9	YrMajor	Num	8	Years in the Major Leagues
10	CrAtBat	Num	8	Career Times at Bat
11	CrHits	Num	8	Career Hits
12	CrHome	Num	8	Career Home Runs
13	CrRuns	Num	8	Career Runs
14	CrRbi	Num	8	Career RBIs
15	CrBB	Num	8	Career Walks
16	League	Char	8	League at the End of 1986
17	Division	Char	8	Division at the End of 1986
18	Position	Char	8	Position(s) in 1986
19	nOuts	Num	8	Put Outs in 1986
20	nAssts	Num	8	Assists in 1986
21	nError	Num	8	Errors in 1986
22	Salary	Num	8	1987 Salary in $ Thousands
23	Div	Char	16	League and Division
24	logSalary	Num	8	Log Salary

Suppose you want to investigate whether you can model the players' salaries for the 1987 season based on batting statistics for the previous season and lifetime batting performance. Since the variation in salaries is much greater for higher salaries, it is appropriate to apply a log transformation for this analysis. The following statements begin the analysis:

```
ods graphics on;

proc reg data=sashelp.baseball;
   id name team league;
   model logSalary = nhits nruns nrbi nbb yrmajor crhits;
run;
```

Output 105.1.2 shows the default output produced by PROC REG. The number of observations table shows that 59 observations are excluded because they have missing values for at least one of the variables used in the analysis. The analysis of variance and parameter estimates tables provide details about the fitted model.

Output 105.1.2 Default Output from PROC REG

The REG Procedure
Model: MODEL1
Dependent Variable: logSalary Log Salary

Number of Observations Read	322
Number of Observations Used	263
Number of Observations with Missing Values	59

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	121.53052	20.25509	60.56	<.0001
Error	256	85.62322	0.33447		
Corrected Total	262	207.15373			

Root MSE	0.57833	**R-Square**	0.5867
Dependent Mean	5.92722	**Adj R-Sq**	0.5770
Coeff Var	9.75719		

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	Intercept	1	4.14614	0.13612	30.46	<.0001
nHits	Hits in 1986	1	0.00663	0.00210	3.15	0.0018
nRuns	Runs in 1986	1	0.00019890	0.00398	0.05	0.9602
nRBI	RBIs in 1986	1	0.00125	0.00235	0.53	0.5947
nBB	Walks in 1986	1	0.00672	0.00239	2.81	0.0054
YrMajor	Years in the Major Leagues	1	0.07108	0.01925	3.69	0.0003
CrHits	Career Hits	1	0.00023910	0.00014571	1.64	0.1020

Before you accept a regression model, it is important to examine influence and fit diagnostics to see whether the model might be unduly influenced by a few observations and whether the data support the assumptions that underlie the linear regression. To facilitate such investigations, you can obtain diagnostic plots by enabling ODS Graphics.

Output 105.1.3 Fit Diagnostics

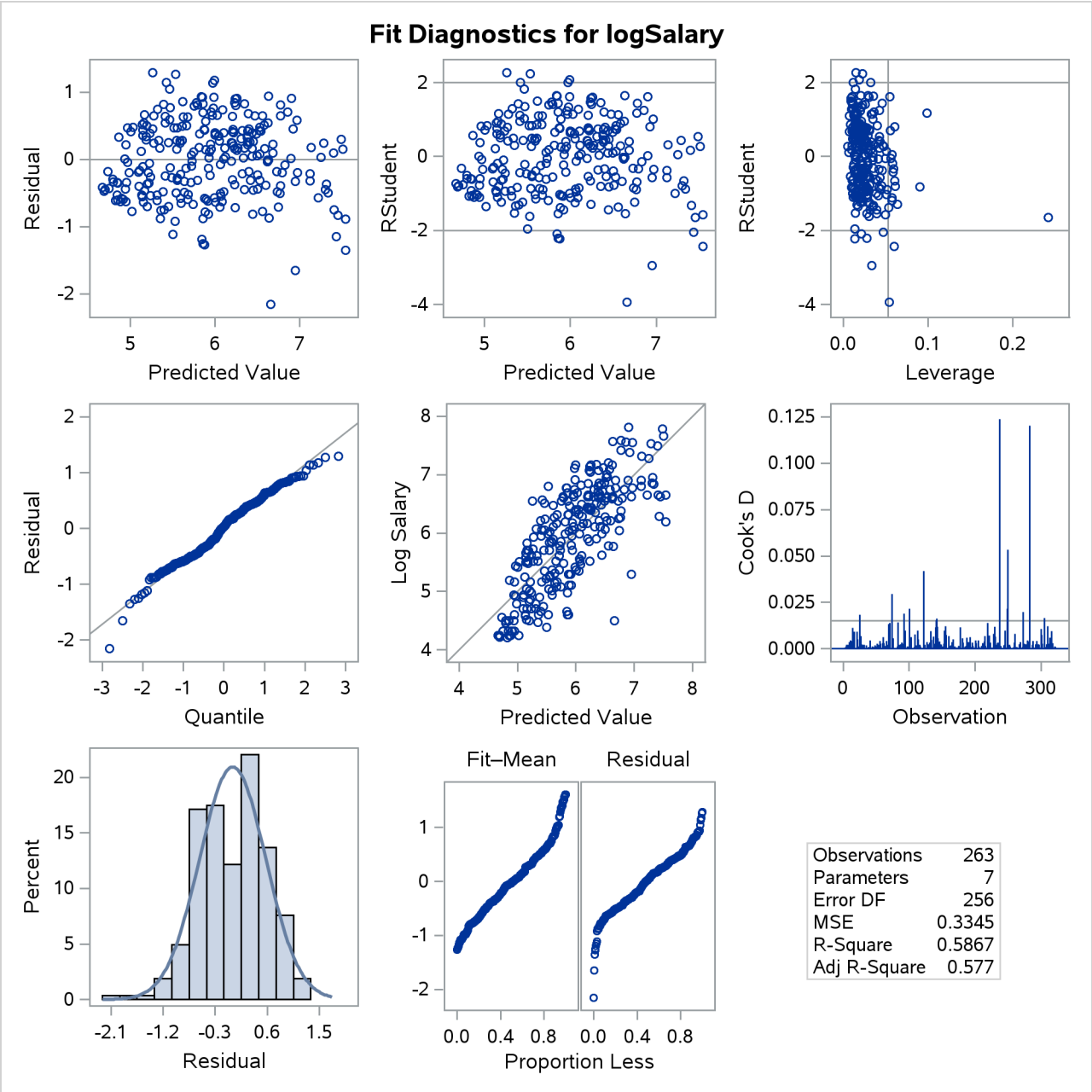

Output 105.1.3 shows a panel of diagnostic plots. The plot of externally studentized residuals (RStudent) by leverage values reveals that there is one observation with very high leverage that might be overly influencing the fit produced. The plot of Cook's *D* by observation also indicates two highly influential observations. To investigate further, you can use the PLOTS= option in the PROC REG statement as follows to produce labeled versions of these plots:

```
proc reg data=sashelp.baseball
         plots(only label)=(RStudentByLeverage CooksD);
   id name team league;
   model logSalary = nhits nruns nrbi nbb yrmajor crhits;
```

```
run;
```

Output 105.1.4 and Output 105.1.5 reveal that Pete Rose is the highly influential observation. You might obtain a better fit to the remaining data if you omit his statistics when building the model.

Output 105.1.4 Outlier and Leverage Diagnostics

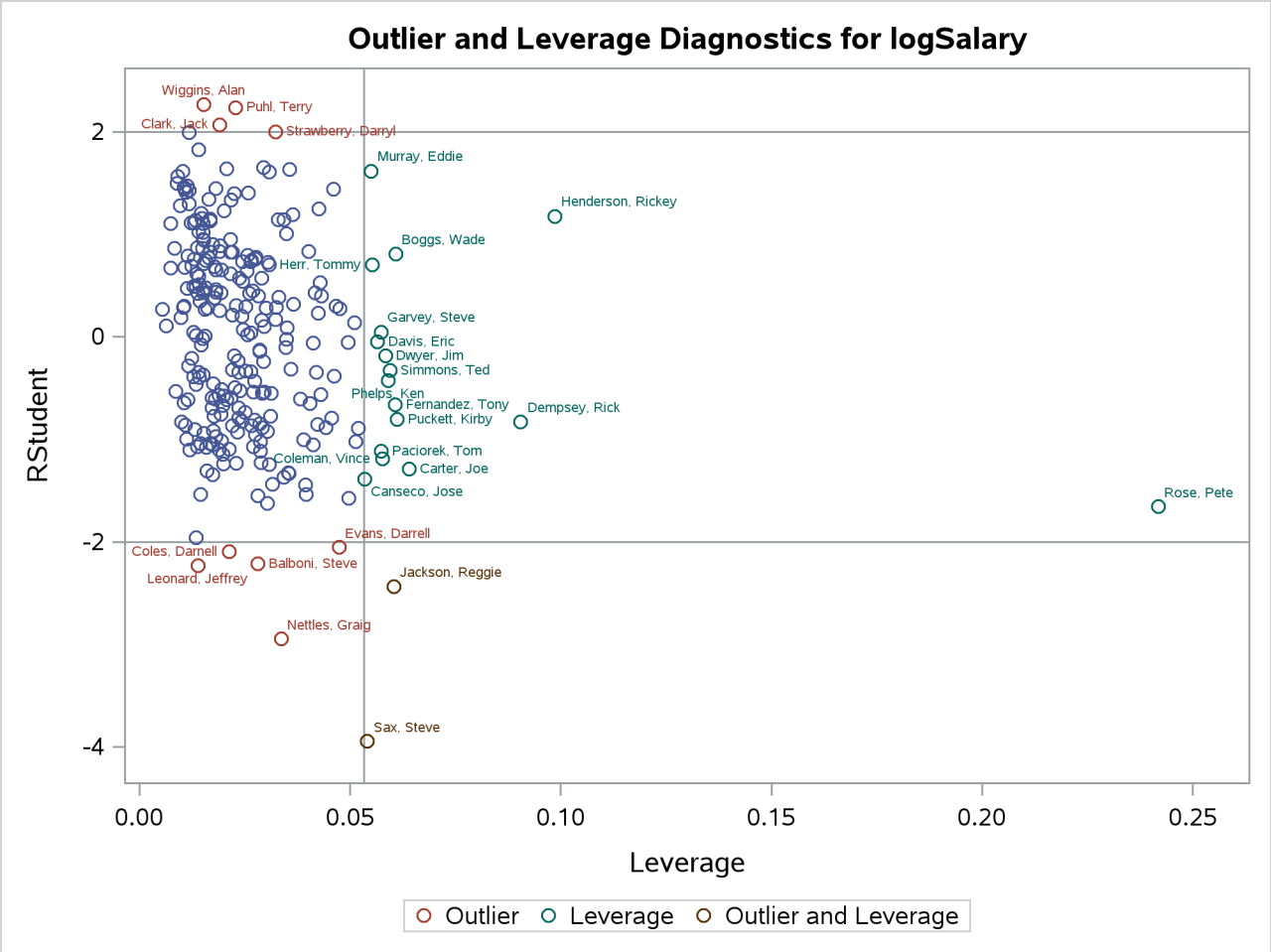

Output 105.1.5 Cook's D

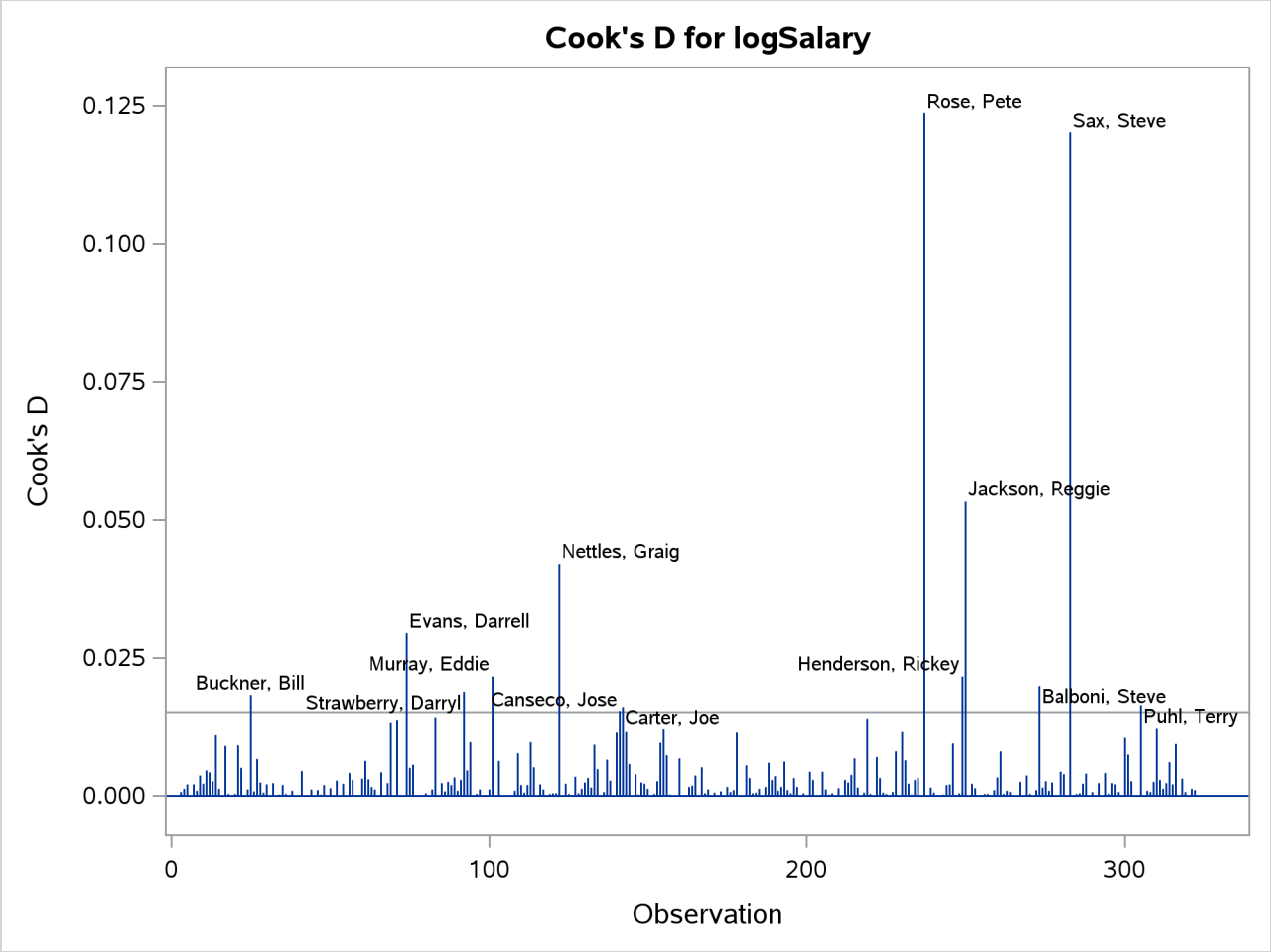

The following statements use a WHERE statement to omit Pete Rose's statistics when building the model. An alternative way to do this within PROC REG is to use a REWEIGHT statement. See "Reweighting Observations in an Analysis" on page 8818 for details about reweighting.

```
proc reg data=sashelp.baseball
         plots=(RStudentByLeverage(label) residuals(smooth));
   where name^="Rose, Pete";
   id name team league;
   model logSalary = nhits nruns nrbi nbb yrmajor crhits;
run;
```

Output 105.1.6 shows the new fit diagnostics panel. You can see that there are still several influential and outlying observations. One possible reason for observing outliers is that the linear model specified is not appropriate to capture the variation in this data. You can often see evidence of an inappropriate model by observing patterns in plots of residuals.

Output 105.1.6 Fit Diagnostics

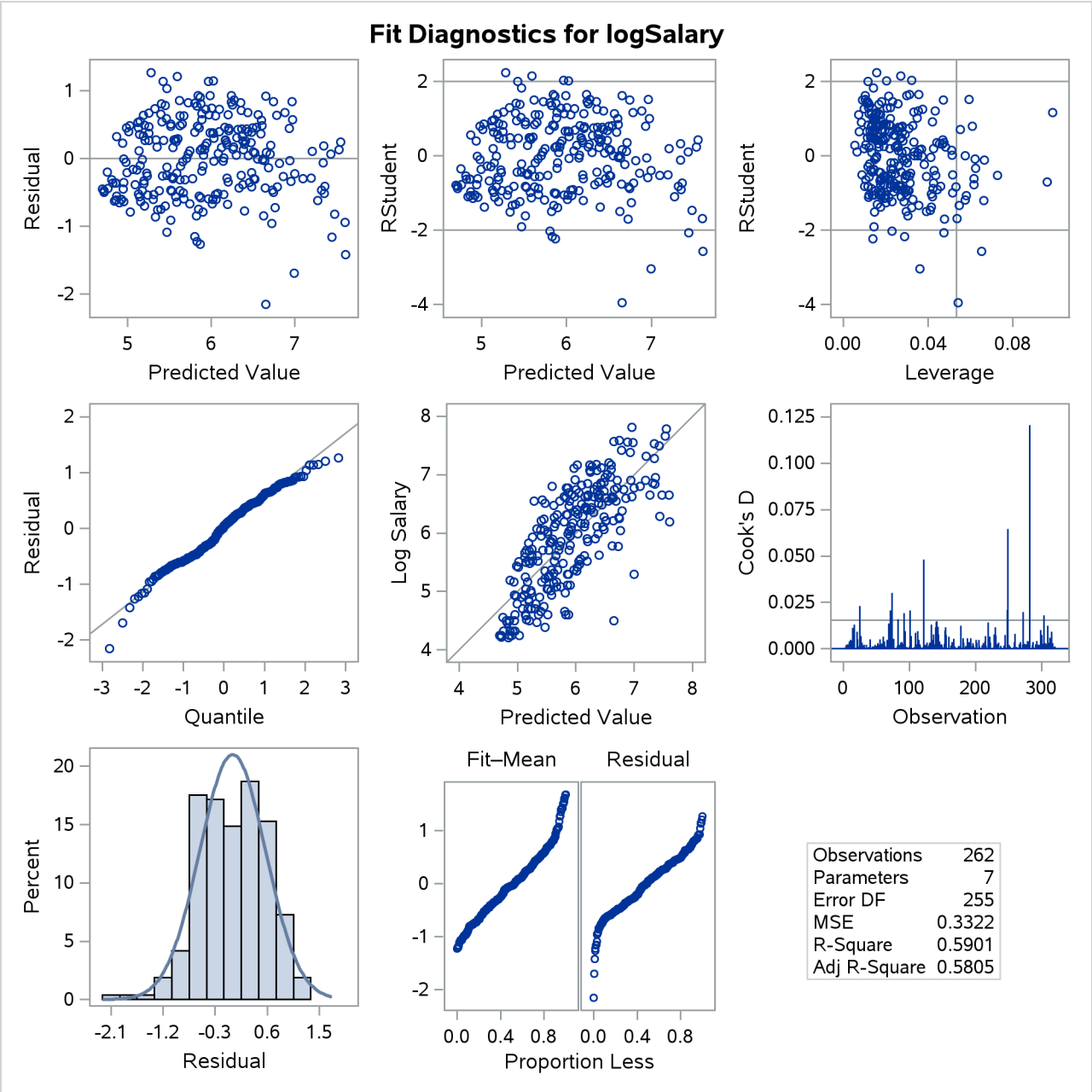

Output 105.1.7 shows plots of the residuals by the regressors in the model. When you specify the RESIDUALS(SMOOTH) suboption of the PLOTS option in the PROC REG statement, a loess fit is overlaid on each of these plots. You can see the same clear pattern in the residual plots for YrMajor and CrHits. Players near the start of their careers and players near the end of their careers get paid less than the model predicts.

Output 105.1.7 Residuals by Regressors

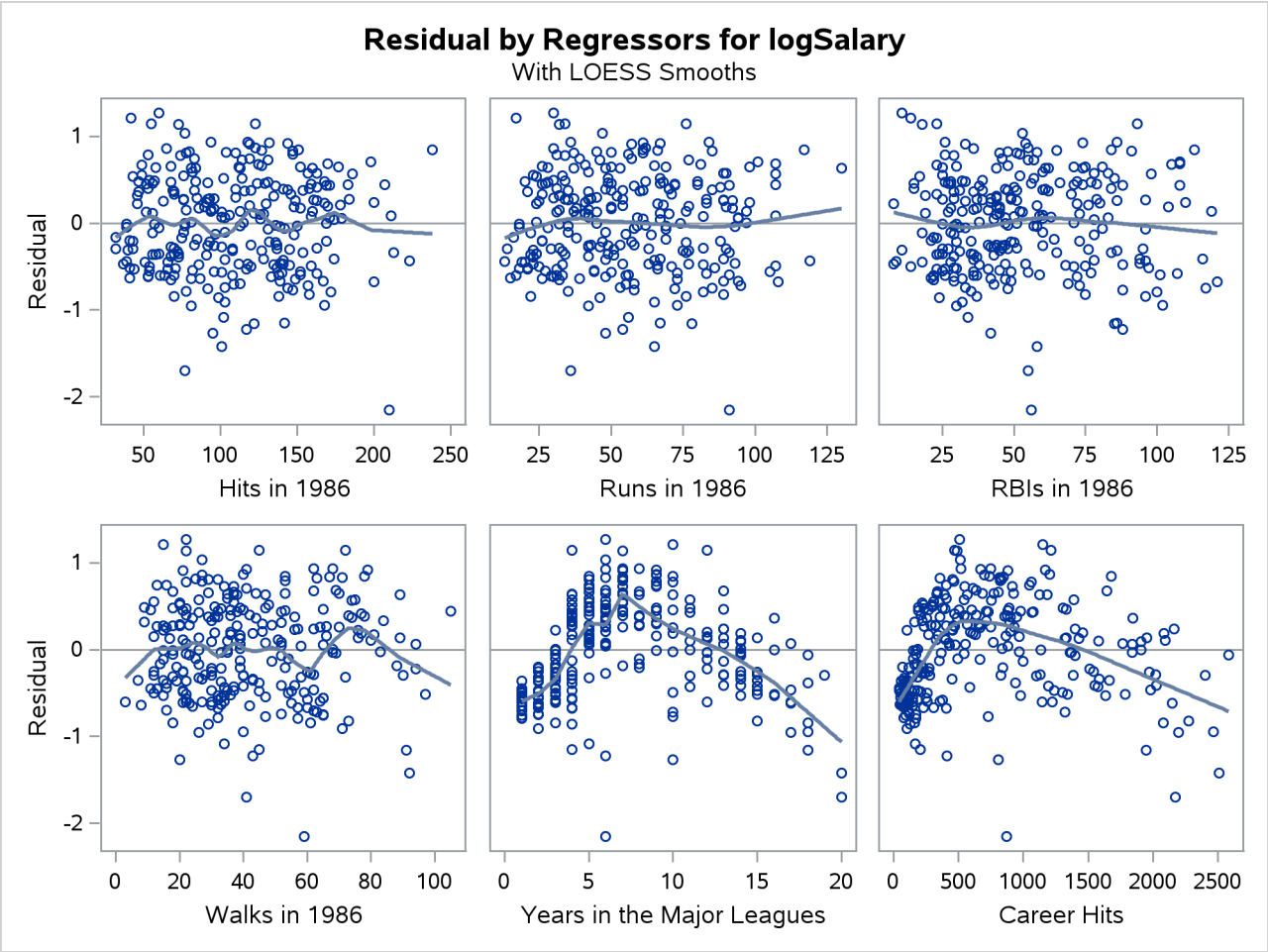

You can address this lack of fit by using polynomials of degree 2 for these two variables as shown in the following statements:

```
data baseball;
   set sashelp.baseball(where=(name^="Rose, Pete"));
   YrMajor2 = yrmajor*yrmajor;
   CrHits2  = crhits*crhits;
run;

proc reg data=baseball
      plots=(diagnostics(stats=none) RStudentByLeverage(label)
             CooksD(label) Residuals(smooth)
             DFFITS(label) DFBETAS ObservedByPredicted(label));
   id name team league;
   model logSalary = nhits nruns nrbi nbb yrmajor crhits
                     yrmajor2 crhits2;
run;
```

Output 105.1.8 shows the analysis of variance and parameter estimates for this model. Note that the R-square value of 0.787 for this model is considerably larger than the R-square value of 0.587 for the initial model shown in Output 105.1.2.

Output 105.1.8 Output from PROC REG

The REG Procedure
Model: MODEL1
Dependent Variable: logSalary Log Salary

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	8	162.73473	20.34184	117.13	<.0001
Error	253	43.93712	0.17366		
Corrected Total	261	206.67186			

Root MSE	0.41673	**R-Square**	0.7874
Dependent Mean	5.92458	**Adj R-Sq**	0.7807
Coeff Var	7.03393		

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	Intercept	1	3.78922	0.11581	32.72	<.0001
nHits	Hits in 1986	1	-0.00012753	0.00159	-0.08	0.9363
nRuns	Runs in 1986	1	0.00215	0.00288	0.75	0.4549
nRBI	RBIs in 1986	1	0.00431	0.00172	2.51	0.0127
nBB	Walks in 1986	1	0.00501	0.00173	2.90	0.0040
YrMajor	Years in the Major Leagues	1	0.23908	0.03443	6.94	<.0001
CrHits	Career Hits	1	0.00170	0.00027562	6.18	<.0001
YrMajor2		1	-0.01440	0.00165	-8.73	<.0001
CrHits2		1	-3.31739E-7	1.001272E-7	-3.31	0.0011

The plots of residuals by regressors in Output 105.1.9 and Output 105.1.10 show that the strong pattern in the plots for CrMajors and CrHits has been reduced, although there is still some indication of a pattern remaining in these residuals. This suggests that a quadratic function might be insufficient to capture dependence of salary on these regressors.

Output 105.1.9 Residuals by Regressors

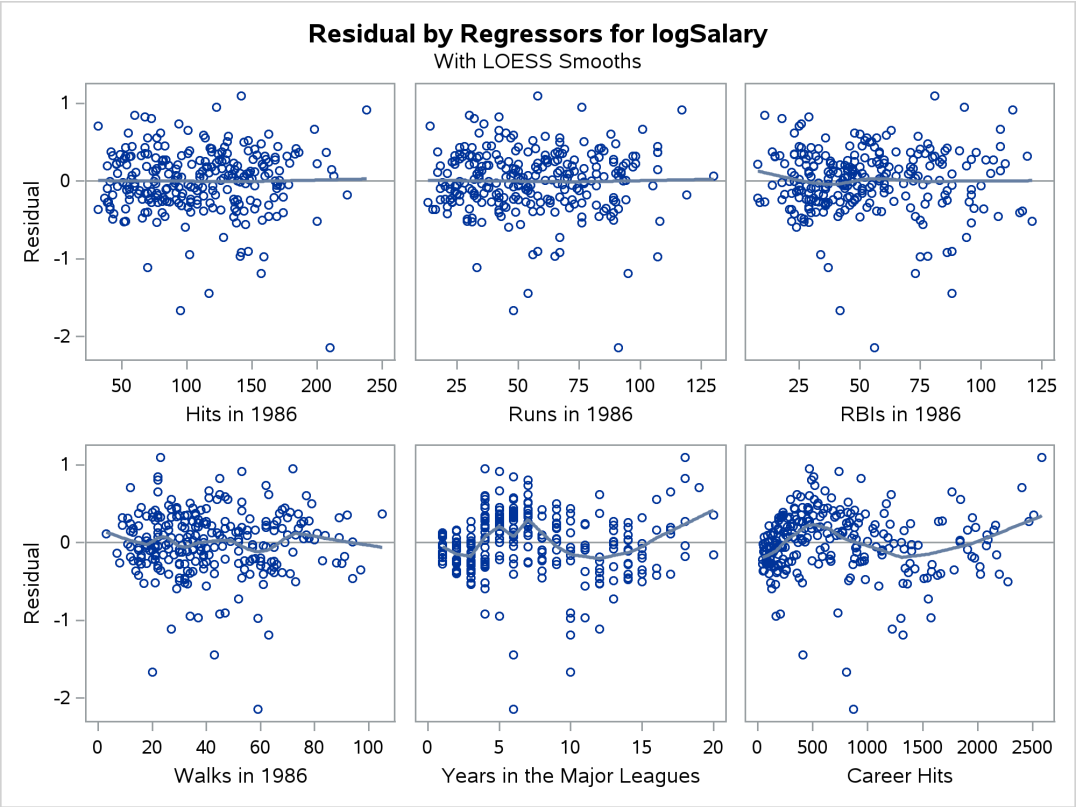

Output 105.1.10 Residuals by Regressors

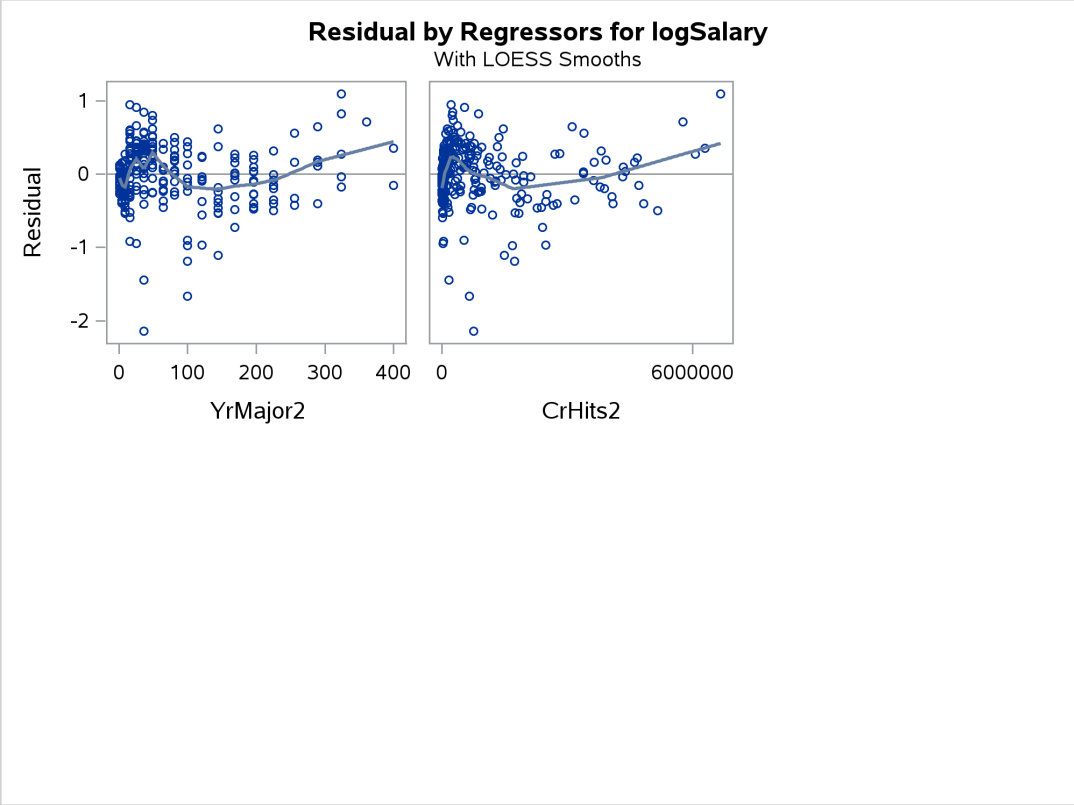

Output 105.1.11 show the diagnostics plots; three of the plots, with points of interest labeled, are shown individually in Output 105.1.12, Output 105.1.13, and Output 105.1.14. The STATS=NONE suboption specified in the PLOTS=DIAGNOSTICS option replaces the inset of statistics with a box plot of the residuals in the fit diagnostics panel. The observed by predicted value plot reveals a reasonably successful model for explaining the variation in salary for most of the players. However, the model tends to overpredict the salaries of several players near the lower end of the salary range. This bias can also be seen in the distribution of the residuals that you can see in the histogram, Q-Q plot, and box plot in Output 105.1.11.

Output 105.1.11 Fit Diagnostics

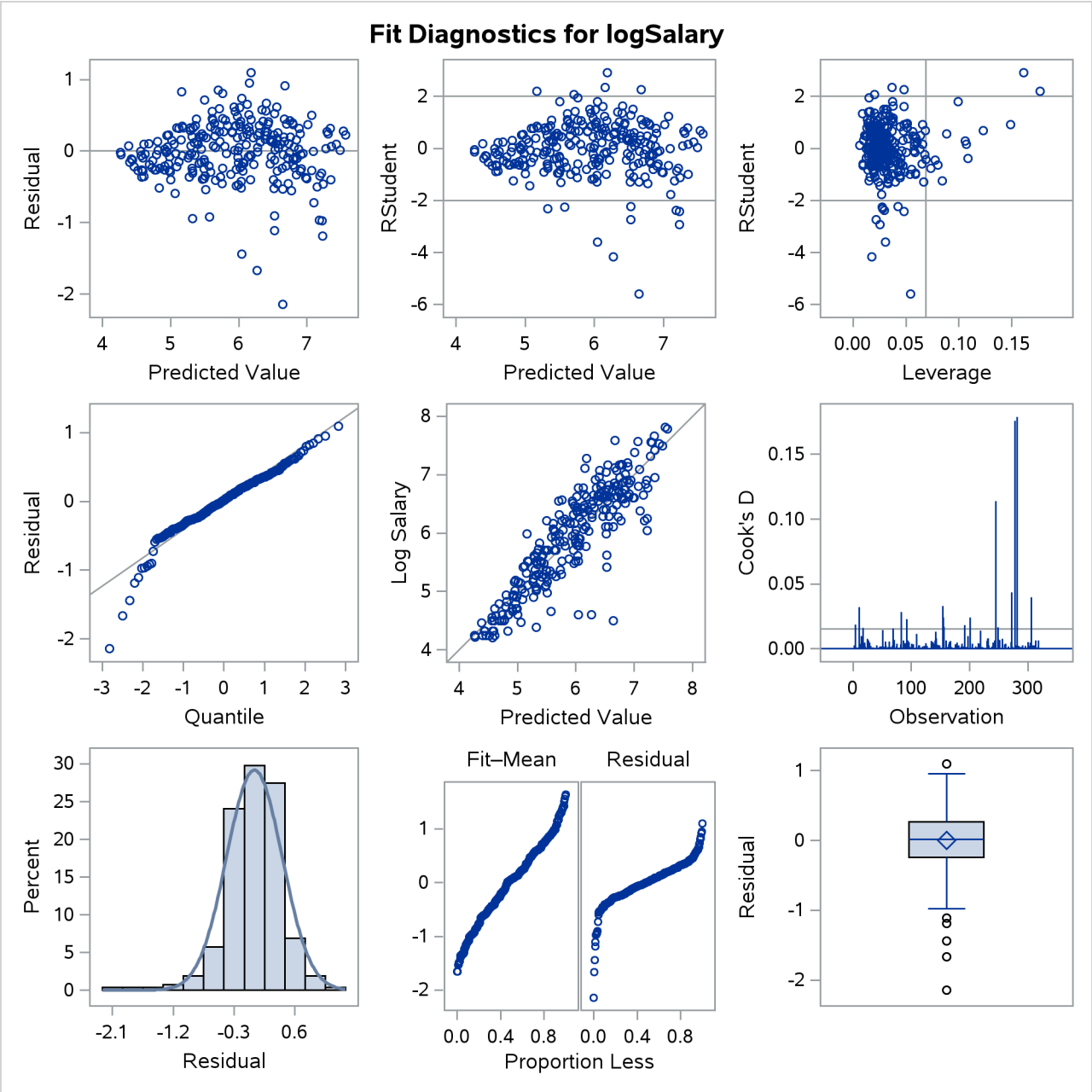

Output 105.1.12 Outlier and Leverage Diagnostics

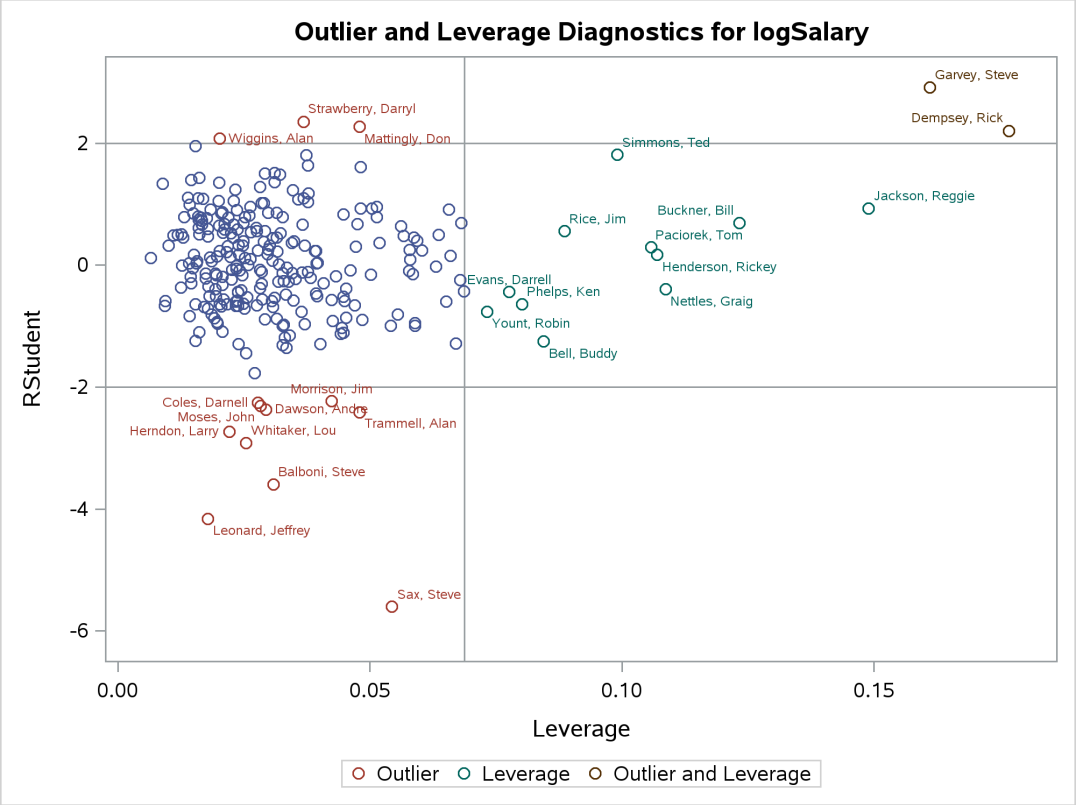

Output 105.1.13 Observed by Predicted Values

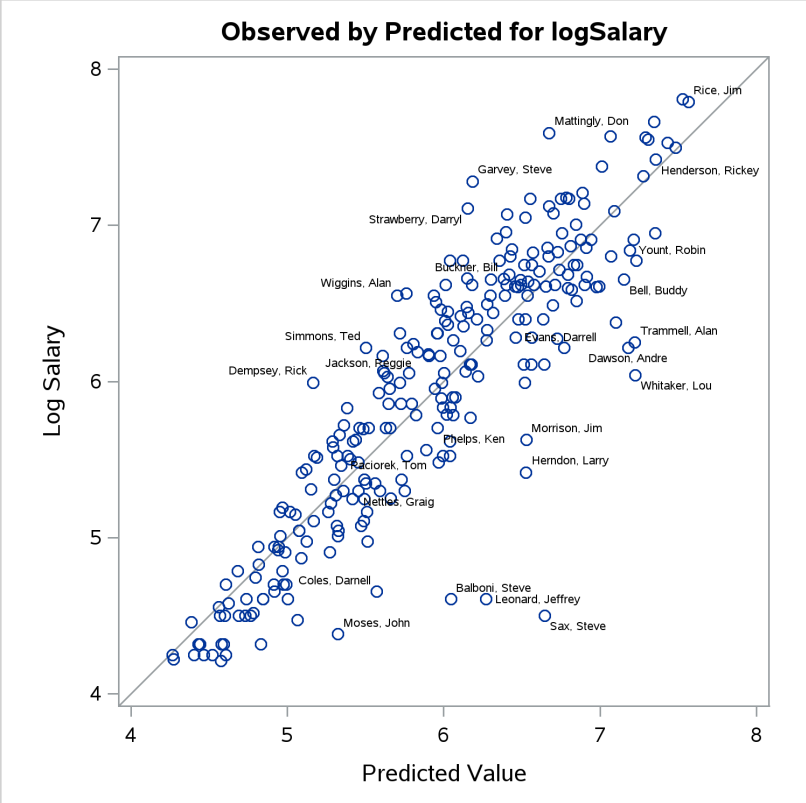

Output 105.1.14 Cook's D

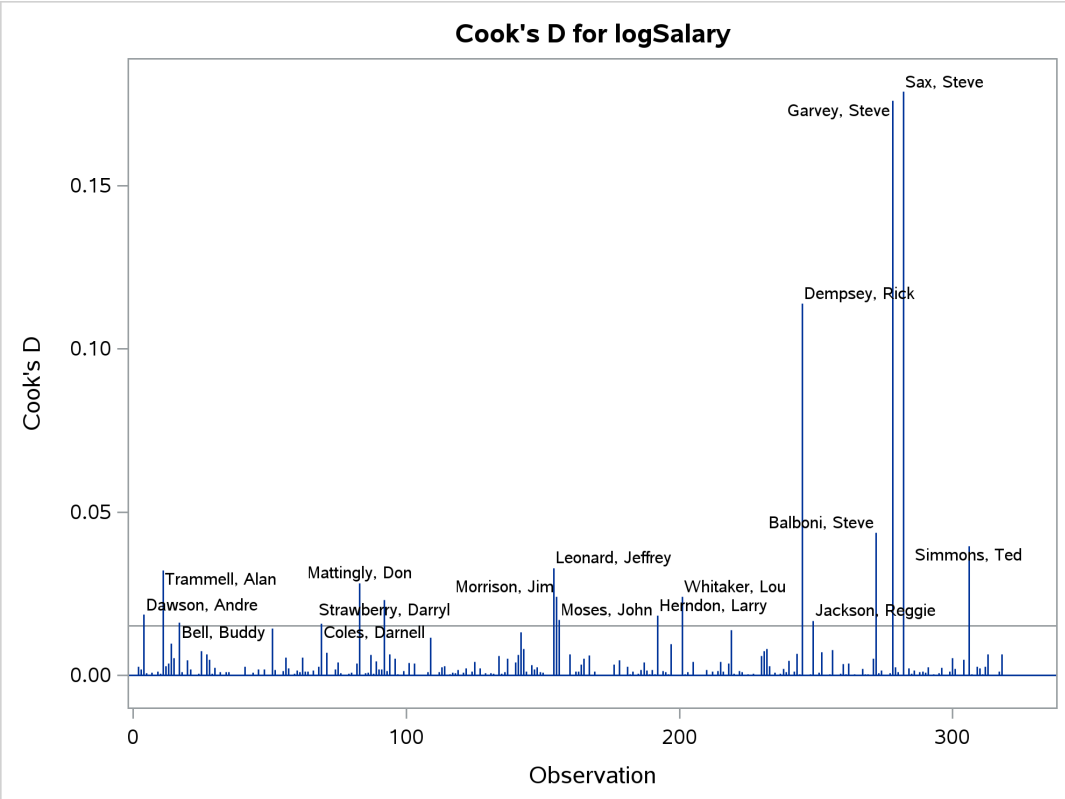

The RStudent by leverage plot in Output 105.1.12 and the Cook's D plot in Output 105.1.14 show that there are still a number of influential observations. By specifying the DFFITS and DFBETAS suboptions of the PLOTS= option, you obtain the additional influence diagnostics plots shown in Output 105.1.15 and Output 105.1.16. For more information about the interpretation of DFFITS and DFBETAS statistics, see the section "Influence Statistics" on page 8810.

You can continue this analysis by investigating how the influential observations that are identified in the various influence plots affect the fit. You can also use PROC ROBUSTREG to obtain a fit that is resistant to the presence of high-leverage points and outliers.

You can create marginal fit diagnostic plots outside PROC REG by using the %Marginal autocall macro. Marginal model plots (Cook and Weisberg 1997) display the marginal relationship between the response and each predictor. Each plot contains a scatter plot of the two variables overlaid with two curves: a smooth fit function for the variables in the plot (labeled "Data") and the overall model predicted values (labeled "Model"). One additional plot displays predicted values on the horizontal axis rather than an independent variable. (See Output 105.1.17.) When the two curves are similar in each of the graphs, there is evidence that the model fits well. When they differ in at least one of the graphs, there is evidence that the model does not fit well.

Output 105.1.15 DFFITS

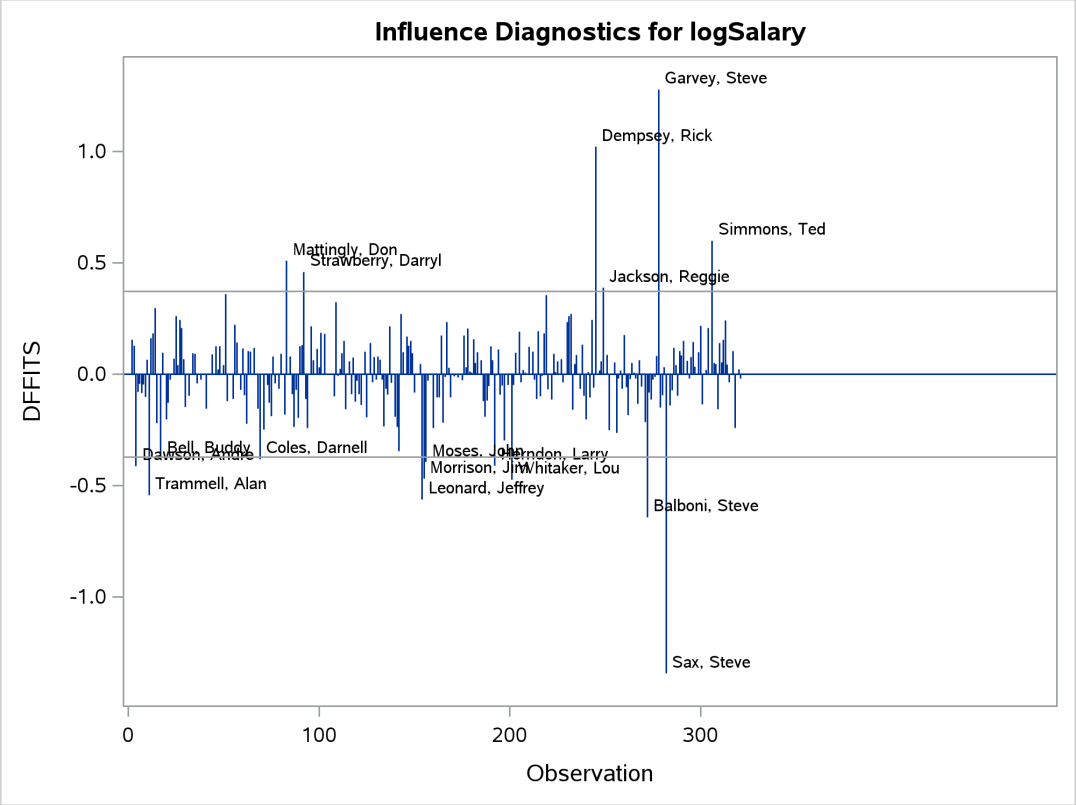

Output 105.1.16 DFBETAS

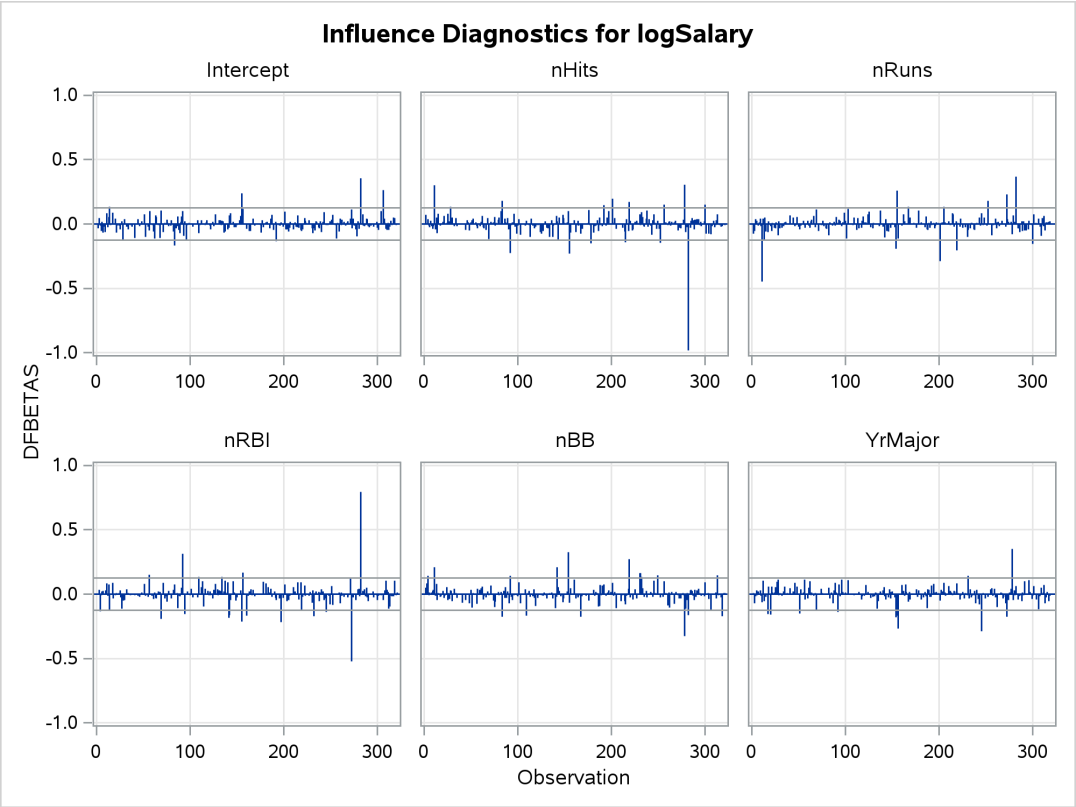

The following statements refit the last model, create a data set that contains predicted values, and use the %Marginal autocall macro to display the marginal model plots:

```
%let ind = nhits nruns nrbi nbb yrmajor crhits yrmajor2 crhits2;
proc reg data=baseball plots=none;
   model logSalary = &ind;
   output out=pred p=p;
run;

%marginal(dependent=logSalary, predicted=p, independents=&ind)
```

The independent variables are specified in a macro variable for convenience to avoid listing them twice; this is not required. Output 105.1.17 displays the results and shows that the model fits less well for the players who spent the most time in the major leagues.

The macro requires you to specify the dependent variable, predicted values variable, and independent variables. The macro input data set is the output data set from PROC REG. Additional options are available. For more information about the %Marginal macro, see the section "Example 25.7: Marginal Model Plots" on page 908 in Chapter 25, "ODS Graphics Template Modification."

Output 105.1.17 Marginal Model Plots

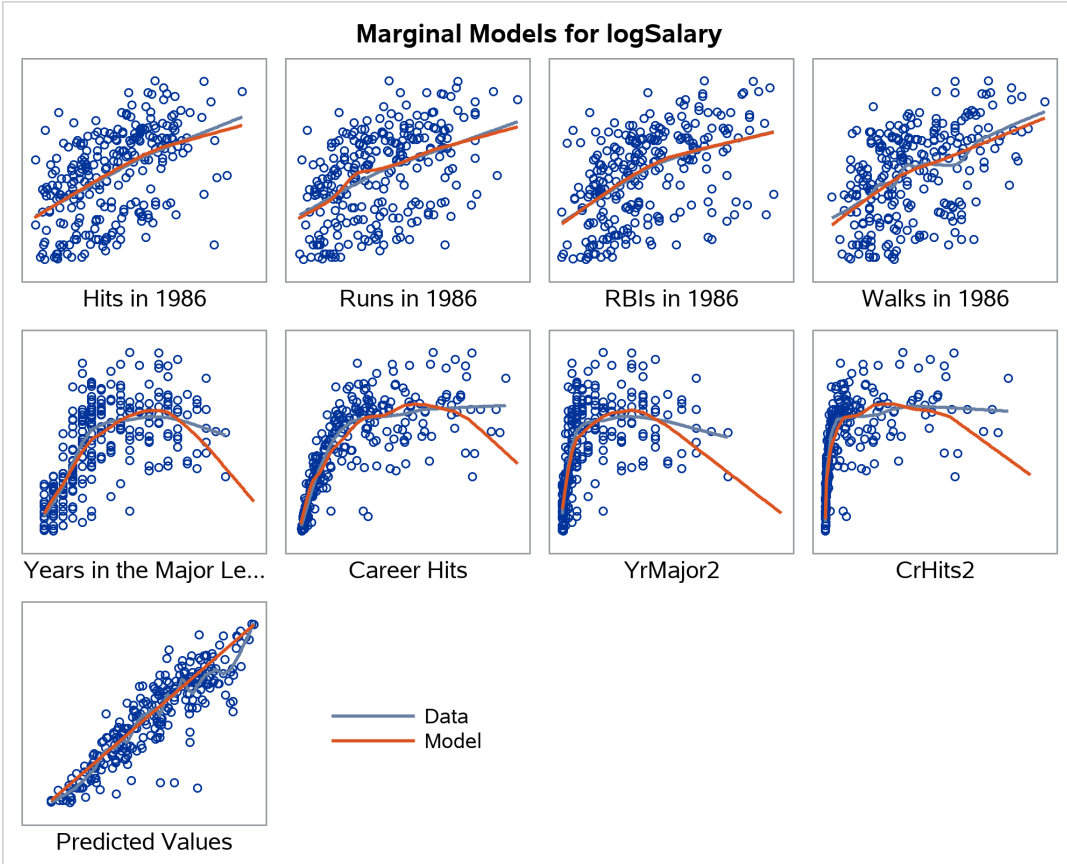

Example 105.2: Aerobic Fitness Prediction

(View the complete code for this example (regex2.sas) in the example repository.)

Aerobic fitness (measured by the ability to consume oxygen) is fit to some simple exercise tests. The goal is to develop an equation to predict fitness based on the exercise tests rather than on expensive and cumbersome oxygen consumption measurements. Three model-selection methods are used: forward selection, backward selection, and MAXR selection. Here are the data:

```
*-------------------Data on Physical Fitness-------------------*
| These measurements were made on men involved in a physical   |
| fitness course at N.C.State Univ. The variables are Age      |
| (years), Weight (kg), Oxygen intake rate (ml per kg body     |
| weight per minute), time to run 1.5 miles (minutes), heart   |
| rate while resting, heart rate while running (same time      |
| Oxygen rate measured), and maximum heart rate recorded while |
| running.                                                     |
| ***Certain values of MaxPulse were changed for this analysis.|
*-------------------------------------------------------------*;
data fitness;
   input Age Weight Oxygen RunTime RestPulse RunPulse MaxPulse @@;
```

```
   datalines;
44 89.47 44.609 11.37 62 178 182   40 75.07 45.313 10.07 62 185 185
44 85.84 54.297  8.65 45 156 168   42 68.15 59.571  8.17 40 166 172
38 89.02 49.874  9.22 55 178 180   47 77.45 44.811 11.63 58 176 176
40 75.98 45.681 11.95 70 176 180   43 81.19 49.091 10.85 64 162 170
44 81.42 39.442 13.08 63 174 176   38 81.87 60.055  8.63 48 170 186
44 73.03 50.541 10.13 45 168 168   45 87.66 37.388 14.03 56 186 192
45 66.45 44.754 11.12 51 176 176   47 79.15 47.273 10.60 47 162 164
54 83.12 51.855 10.33 50 166 170   49 81.42 49.156  8.95 44 180 185
51 69.63 40.836 10.95 57 168 172   51 77.91 46.672 10.00 48 162 168
48 91.63 46.774 10.25 48 162 164   49 73.37 50.388 10.08 67 168 168
57 73.37 39.407 12.63 58 174 176   54 79.38 46.080 11.17 62 156 165
52 76.32 45.441  9.63 48 164 166   50 70.87 54.625  8.92 48 146 155
51 67.25 45.118 11.08 48 172 172   54 91.63 39.203 12.88 44 168 172
51 73.71 45.790 10.47 59 186 188   57 59.08 50.545  9.93 49 148 155
49 76.32 48.673  9.40 56 186 188   48 61.24 47.920 11.50 52 170 176
52 82.78 47.467 10.50 53 170 172
;
```

The following statements demonstrate the FORWARD, BACKWARD, and MAXR model selection methods:

```
proc reg data=fitness;
   model Oxygen=Age Weight RunTime RunPulse RestPulse MaxPulse
         / selection=forward;
   model Oxygen=Age Weight RunTime RunPulse RestPulse MaxPulse
         / selection=backward;
   model Oxygen=Age Weight RunTime RunPulse RestPulse MaxPulse
         / selection=maxr;
run;
```

Output 105.2.1 shows the sequence of models produced by the FORWARD model-selection method.

Output 105.2.1 Forward Selection Method: PROC REG

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

Forward Selection: Step 1

Variable RunTime Entered: R-Square = 0.7434 and C(p) = 13.6988

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	632.90010	632.90010	84.01	<.0001
Error	29	218.48144	7.53384		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	82.42177	3.85530	3443.36654	457.05	<.0001
RunTime	-3.31056	0.36119	632.90010	84.01	<.0001

Output 105.2.1 *continued*

Bounds on condition number: 1, 1

Forward Selection: Step 2

Variable Age Entered: R-Square = 0.7642 and C(p) = 12.3894

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	650.66573	325.33287	45.38	<.0001
Error	28	200.71581	7.16842		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	88.46229	5.37264	1943.41071	271.11	<.0001
Age	-0.15037	0.09551	17.76563	2.48	0.1267
RunTime	-3.20395	0.35877	571.67751	79.75	<.0001

Bounds on condition number: 1.0369, 4.1478

Forward Selection: Step 3

Variable RunPulse Entered: R-Square = 0.8111 and C(p) = 6.9596

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	690.55086	230.18362	38.64	<.0001
Error	27	160.83069	5.95669		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	111.71806	10.23509	709.69014	119.14	<.0001
Age	-0.25640	0.09623	42.28867	7.10	0.0129
RunTime	-2.82538	0.35828	370.43529	62.19	<.0001
RunPulse	-0.13091	0.05059	39.88512	6.70	0.0154

Bounds on condition number: 1.3548, 11.597

Forward Selection: Step 4

Variable MaxPulse Entered: R-Square = 0.8368 and C(p) = 4.8800

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	712.45153	178.11288	33.33	<.0001
Error	26	138.93002	5.34346		
Corrected Total	30	851.38154			

Output 105.2.1 *continued*

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	98.14789	11.78569	370.57373	69.35	<.0001
Age	-0.19773	0.09564	22.84231	4.27	0.0488
RunTime	-2.76758	0.34054	352.93570	66.05	<.0001
RunPulse	-0.34811	0.11750	46.90089	8.78	0.0064
MaxPulse	0.27051	0.13362	21.90067	4.10	0.0533

Bounds on condition number: 8.4182, 76.851

Forward Selection: Step 5

Variable Weight Entered: R-Square = 0.8480 and C(p) = 5.1063

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	721.97309	144.39462	27.90	<.0001
Error	25	129.40845	5.17634		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	102.20428	11.97929	376.78935	72.79	<.0001
Age	-0.21962	0.09550	27.37429	5.29	0.0301
Weight	-0.07230	0.05331	9.52157	1.84	0.1871
RunTime	-2.68252	0.34099	320.35968	61.89	<.0001
RunPulse	-0.37340	0.11714	52.59624	10.16	0.0038
MaxPulse	0.30491	0.13394	26.82640	5.18	0.0316

Bounds on condition number: 8.7312, 104.83

The final variable available to add to the model, RestPulse, is not added since it does not meet the 50% (the default value of the SLE option is 0.5 for FORWARD selection) significance-level criterion for entry into the model.

The BACKWARD model-selection method begins with the full model. Output 105.2.2 shows the steps of the BACKWARD method. RestPulse is the first variable deleted, followed by Weight. No other variables are deleted from the model since the variables remaining (Age, RunTime, RunPulse, and MaxPulse) are all significant at the 10% (the default value of the SLS option is 0.1 for the BACKWARD elimination method) significance level.

Output 105.2.2 Backward Selection Method: PROC REG

Backward Elimination: Step 0

All Variables Entered: R-Square = 0.8487 and C(p) = 7.0000

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	722.54361	120.42393	22.43	<.0001
Error	24	128.83794	5.36825		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	102.93448	12.40326	369.72831	68.87	<.0001
Age	-0.22697	0.09984	27.74577	5.17	0.0322
Weight	-0.07418	0.05459	9.91059	1.85	0.1869
RunTime	-2.62865	0.38456	250.82210	46.72	<.0001
RunPulse	-0.36963	0.11985	51.05806	9.51	0.0051
RestPulse	-0.02153	0.06605	0.57051	0.11	0.7473
MaxPulse	0.30322	0.13650	26.49142	4.93	0.0360

Bounds on condition number: 8.7438, 137.13

Backward Elimination: Step 1

Variable RestPulse Removed: R-Square = 0.8480 and C(p) = 5.1063

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	721.97309	144.39462	27.90	<.0001
Error	25	129.40845	5.17634		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	102.20428	11.97929	376.78935	72.79	<.0001
Age	-0.21962	0.09550	27.37429	5.29	0.0301
Weight	-0.07230	0.05331	9.52157	1.84	0.1871
RunTime	-2.68252	0.34099	320.35968	61.89	<.0001
RunPulse	-0.37340	0.11714	52.59624	10.16	0.0038
MaxPulse	0.30491	0.13394	26.82640	5.18	0.0316

Output 105.2.2 *continued*

Bounds on condition number: 8.7312, 104.83

Backward Elimination: Step 2

Variable Weight Removed: R-Square = 0.8368 and C(p) = 4.8800

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	712.45153	178.11288	33.33	<.0001
Error	26	138.93002	5.34346		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	98.14789	11.78569	370.57373	69.35	<.0001
Age	-0.19773	0.09564	22.84231	4.27	0.0488
RunTime	-2.76758	0.34054	352.93570	66.05	<.0001
RunPulse	-0.34811	0.11750	46.90089	8.78	0.0064
MaxPulse	0.27051	0.13362	21.90067	4.10	0.0533

Bounds on condition number: 8.4182, 76.851

The MAXR method tries to find the "best" one-variable model, the "best" two-variable model, and so on. Output 105.2.3 shows that the one-variable model contains RunTime; the two-variable model contains RunTime and Age; the three-variable model contains RunTime, Age, and RunPulse; the four-variable model contains Age, RunTime, RunPulse, and MaxPulse; the five-variable model contains Age, Weight, RunTime, RunPulse, and MaxPulse; and finally, the six-variable model contains all the variables in the MODEL statement.

Output 105.2.3 Maximum R-Square Improvement Selection Method: PROC REG

Maximum R-Square Improvement: Step 1

Variable RunTime Entered: R-Square = 0.7434 and C(p) = 13.6988

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	632.90010	632.90010	84.01	<.0001
Error	29	218.48144	7.53384		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	82.42177	3.85530	3443.36654	457.05	<.0001
RunTime	-3.31056	0.36119	632.90010	84.01	<.0001

Output 105.2.3 *continued*

Bounds on condition number: 1, 1

The above model is the best 1-variable model found.

Maximum R-Square Improvement: Step 2

Variable Age Entered: R-Square = 0.7642 and C(p) = 12.3894

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	650.66573	325.33287	45.38	<.0001
Error	28	200.71581	7.16842		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	88.46229	5.37264	1943.41071	271.11	<.0001
Age	-0.15037	0.09551	17.76563	2.48	0.1267
RunTime	-3.20395	0.35877	571.67751	79.75	<.0001

Bounds on condition number: 1.0369, 4.1478

The above model is the best 2-variable model found.

Maximum R-Square Improvement: Step 3

Variable RunPulse Entered: R-Square = 0.8111 and C(p) = 6.9596

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	690.55086	230.18362	38.64	<.0001
Error	27	160.83069	5.95669		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	111.71806	10.23509	709.69014	119.14	<.0001
Age	-0.25640	0.09623	42.28867	7.10	0.0129
RunTime	-2.82538	0.35828	370.43529	62.19	<.0001
RunPulse	-0.13091	0.05059	39.88512	6.70	0.0154

Output 105.2.3 *continued*

Bounds on condition number: 1.3548, 11.597

The above model is the best 3-variable model found.

Maximum R-Square Improvement: Step 4

Variable MaxPulse Entered: R-Square = 0.8368 and C(p) = 4.8800

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	712.45153	178.11288	33.33	<.0001
Error	26	138.93002	5.34346		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	98.14789	11.78569	370.57373	69.35	<.0001
Age	-0.19773	0.09564	22.84231	4.27	0.0488
RunTime	-2.76758	0.34054	352.93570	66.05	<.0001
RunPulse	-0.34811	0.11750	46.90089	8.78	0.0064
MaxPulse	0.27051	0.13362	21.90067	4.10	0.0533

Bounds on condition number: 8.4182, 76.851

The above model is the best 4-variable model found.

Maximum R-Square Improvement: Step 5

Variable Weight Entered: R-Square = 0.8480 and C(p) = 5.1063

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	721.97309	144.39462	27.90	<.0001
Error	25	129.40845	5.17634		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	102.20428	11.97929	376.78935	72.79	<.0001
Age	-0.21962	0.09550	27.37429	5.29	0.0301
Weight	-0.07230	0.05331	9.52157	1.84	0.1871
RunTime	-2.68252	0.34099	320.35968	61.89	<.0001
RunPulse	-0.37340	0.11714	52.59624	10.16	0.0038
MaxPulse	0.30491	0.13394	26.82640	5.18	0.0316

Output 105.2.3 *continued*

Bounds on condition number: 8.7312, 104.83

The above model is the best 5-variable model found.

Maximum R-Square Improvement: Step 6

Variable RestPulse Entered: R-Square = 0.8487 and C(p) = 7.0000

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	722.54361	120.42393	22.43	<.0001
Error	24	128.83794	5.36825		
Corrected Total	30	851.38154			

Variable	Parameter Estimate	Standard Error	Type II SS	F Value	Pr > F
Intercept	102.93448	12.40326	369.72831	68.87	<.0001
Age	-0.22697	0.09984	27.74577	5.17	0.0322
Weight	-0.07418	0.05459	9.91059	1.85	0.1869
RunTime	-2.62865	0.38456	250.82210	46.72	<.0001
RunPulse	-0.36963	0.11985	51.05806	9.51	0.0051
RestPulse	-0.02153	0.06605	0.57051	0.11	0.7473
MaxPulse	0.30322	0.13650	26.49142	4.93	0.0360

Bounds on condition number: 8.7438, 137.13

Note that for all three of these methods, RestPulse contributes least to the model. In the case of forward selection, it is not added to the model. In the case of backward selection, it is the first variable to be removed from the model. In the case of MAXR selection, RestPulse is included only for the full model.

For the STEPWISE, BACKWARD, and FORWARD selection methods, you can control the amount of detail displayed by using the DETAILS option, and you can use ODS Graphics to produce plots that show how selection criteria progress as the selection proceeds. For example, the following statements display only the selection summary table for the FORWARD selection method (Output 105.2.4) and produce the plots shown in Output 105.2.5 and Output 105.2.6.

```
ods graphics on;

proc reg data=fitness plots=(criteria sbc);
   model Oxygen=Age Weight RunTime RunPulse RestPulse MaxPulse
         / selection=forward details=summary;
run;
```

Output 105.2.4 Forward Selection Summary

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

Summary of Forward Selection

Step	Variable Entered	Number Vars In	Partial R-Square	Model R-Square	C(p)	F Value	Pr > F
1	RunTime	1	0.7434	0.7434	13.6988	84.01	<.0001
2	Age	2	0.0209	0.7642	12.3894	2.48	0.1267
3	RunPulse	3	0.0468	0.8111	6.9596	6.70	0.0154
4	MaxPulse	4	0.0257	0.8368	4.8800	4.10	0.0533
5	Weight	5	0.0112	0.8480	5.1063	1.84	0.1871

Output 105.2.5 show how six fit criteria progress as the forward selection proceeds. The step at which each criterion achieves its best value is indicated. For example, the BIC criterion achieves its minimum value for the model at step 4. Note that this does not mean that the model at step 4 achieves the smallest BIC criterion among all possible models that use a subset of the regressors; the model at step 4 yields the smallest BIC statistic among the models at each step of the forward selection. Output 105.2.6 show the progression of the SBC statistic in its own plot. If you want to see six of the selection criteria in individual plots, you can specify the UNPACK suboption of the PLOTS=CRITERIA option in the PROC REG statement.

Output 105.2.5 Fit Criteria

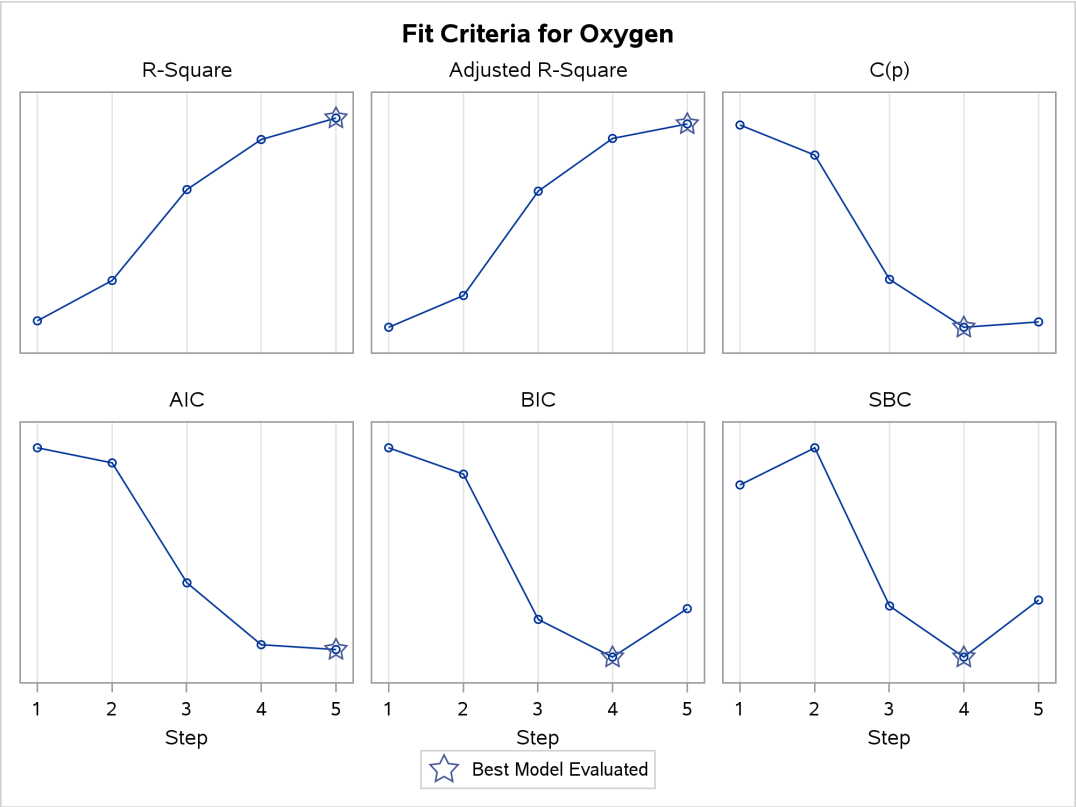

Output 105.2.6 SBC Criterion

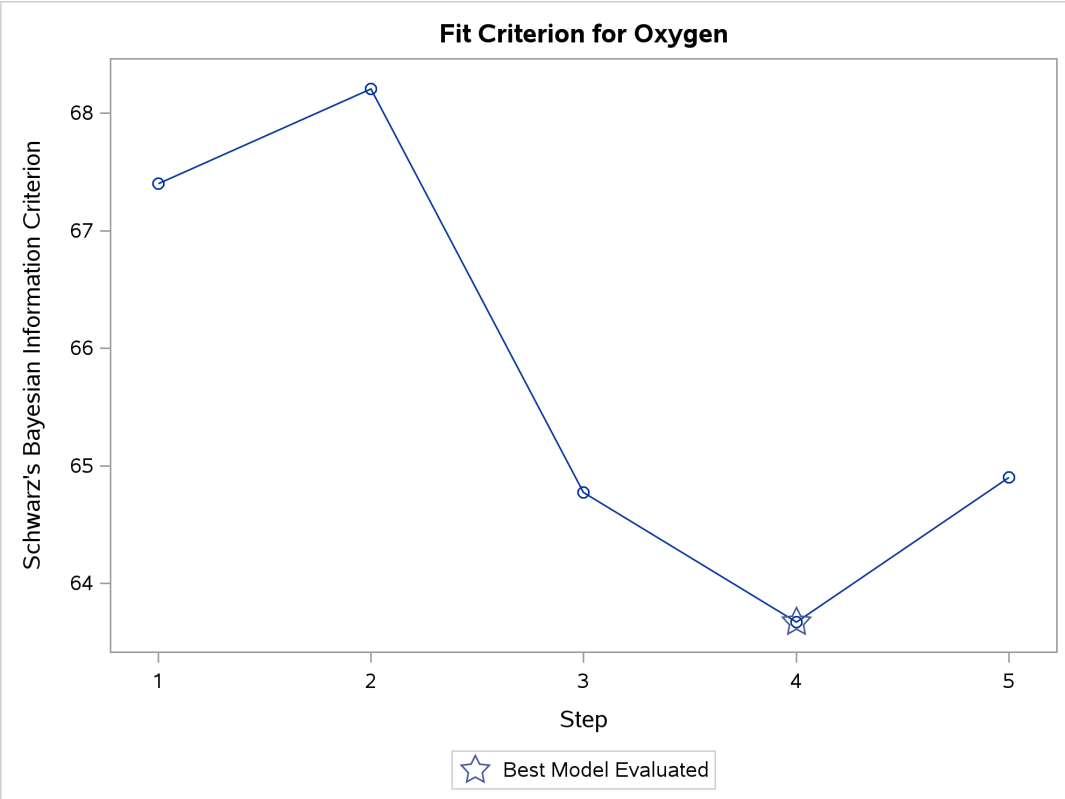

Next, the RSQUARE model-selection method is used to request R square and C_p statistics for all possible combinations of the six independent variables. The following statements produce Output 105.2.7:

```
proc reg data=fitness plots=(criteria(label) cp);
   model Oxygen=Age Weight RunTime RunPulse RestPulse MaxPulse
         / selection=rsquare cp;
   title 'Physical fitness data: all models';
run;
```

Output 105.2.7 All Models by the RSQUARE Method: PROC REG

Physical fitness data: all models

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

R-Square Selection Method

Model Index	Number in Model	R-Square	C(p)	Variables in Model
1	1	0.7434	13.6988	RunTime
2	1	0.1595	106.3021	RestPulse
3	1	0.1584	106.4769	RunPulse
4	1	0.0928	116.8818	Age
5	1	0.0560	122.7072	MaxPulse
6	1	0.0265	127.3948	Weight
7	2	0.7642	12.3894	Age RunTime
8	2	0.7614	12.8372	RunTime RunPulse
9	2	0.7452	15.4069	RunTime MaxPulse
10	2	0.7449	15.4523	Weight RunTime
11	2	0.7435	15.6746	RunTime RestPulse
12	2	0.3760	73.9645	Age RunPulse
13	2	0.3003	85.9742	Age RestPulse
14	2	0.2894	87.6951	RunPulse MaxPulse
15	2	0.2600	92.3638	Age MaxPulse
16	2	0.2350	96.3209	RunPulse RestPulse
17	2	0.1806	104.9523	Weight RestPulse
18	2	0.1740	105.9939	RestPulse MaxPulse
19	2	0.1669	107.1332	Weight RunPulse
20	2	0.1506	109.7057	Age Weight
21	2	0.0675	122.8881	Weight MaxPulse
22	3	0.8111	6.9596	Age RunTime RunPulse
23	3	0.8100	7.1350	RunTime RunPulse MaxPulse
24	3	0.7817	11.6167	Age RunTime MaxPulse
25	3	0.7708	13.3453	Age Weight RunTime
26	3	0.7673	13.8974	Age RunTime RestPulse
27	3	0.7619	14.7619	RunTime RunPulse RestPulse
28	3	0.7618	14.7729	Weight RunTime RunPulse
29	3	0.7462	17.2588	Weight RunTime MaxPulse
30	3	0.7452	17.4060	RunTime RestPulse MaxPulse
31	3	0.7451	17.4243	Weight RunTime RestPulse
32	3	0.4666	61.5873	Age RunPulse RestPulse
33	3	0.4223	68.6250	Age RunPulse MaxPulse
34	3	0.4091	70.7102	Age Weight RunPulse
35	3	0.3900	73.7424	Age RestPulse MaxPulse
36	3	0.3568	79.0013	Age Weight RestPulse
37	3	0.3538	79.4891	RunPulse RestPulse MaxPulse
38	3	0.3208	84.7216	Weight RunPulse MaxPulse
39	3	0.2902	89.5693	Age Weight MaxPulse
40	3	0.2447	96.7952	Weight RunPulse RestPulse
41	3	0.1882	105.7430	Weight RestPulse MaxPulse

Output 105.2.7 *continued*

Physical fitness data: all models

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

R-Square Selection Method

Model Index	Number in Model	R-Square	C(p)	Variables in Model
42	**4**	0.8368	4.8800	Age RunTime RunPulse MaxPulse
43	**4**	0.8165	8.1035	Age Weight RunTime RunPulse
44	**4**	0.8158	8.2056	Weight RunTime RunPulse MaxPulse
45	**4**	0.8117	8.8683	Age RunTime RunPulse RestPulse
46	**4**	0.8104	9.0697	RunTime RunPulse RestPulse MaxPulse
47	**4**	0.7862	12.9039	Age Weight RunTime MaxPulse
48	**4**	0.7834	13.3468	Age RunTime RestPulse MaxPulse
49	**4**	0.7750	14.6788	Age Weight RunTime RestPulse
50	**4**	0.7623	16.7058	Weight RunTime RunPulse RestPulse
51	**4**	0.7462	19.2550	Weight RunTime RestPulse MaxPulse
52	**4**	0.5034	57.7590	Age Weight RunPulse RestPulse
53	**4**	0.5025	57.9092	Age RunPulse RestPulse MaxPulse
54	**4**	0.4717	62.7830	Age Weight RunPulse MaxPulse
55	**4**	0.4256	70.0963	Age Weight RestPulse MaxPulse
56	**4**	0.3858	76.4100	Weight RunPulse RestPulse MaxPulse
57	**5**	0.8480	5.1063	Age Weight RunTime RunPulse MaxPulse
58	**5**	0.8370	6.8461	Age RunTime RunPulse RestPulse MaxPulse
59	**5**	0.8176	9.9348	Age Weight RunTime RunPulse RestPulse
60	**5**	0.8161	10.1685	Weight RunTime RunPulse RestPulse MaxPulse
61	**5**	0.7887	14.5111	Age Weight RunTime RestPulse MaxPulse
62	**5**	0.5541	51.7233	Age Weight RunPulse RestPulse MaxPulse
63	**6**	0.8487	7.0000	Age Weight RunTime RunPulse RestPulse MaxPulse

The models in Output 105.2.7 are arranged first by the number of variables in the model and then by the magnitude of R square for the model.

Output 105.2.8 shows the panel of fit criteria for the RSQUARE selection method. The best models (based on the R-square statistic) for each subset size are indicated on the plots. The LABEL suboption specifies that these models are labeled by the model number that appears in the summary table shown in Output 105.2.7.

Output 105.2.8 Fit Criteria

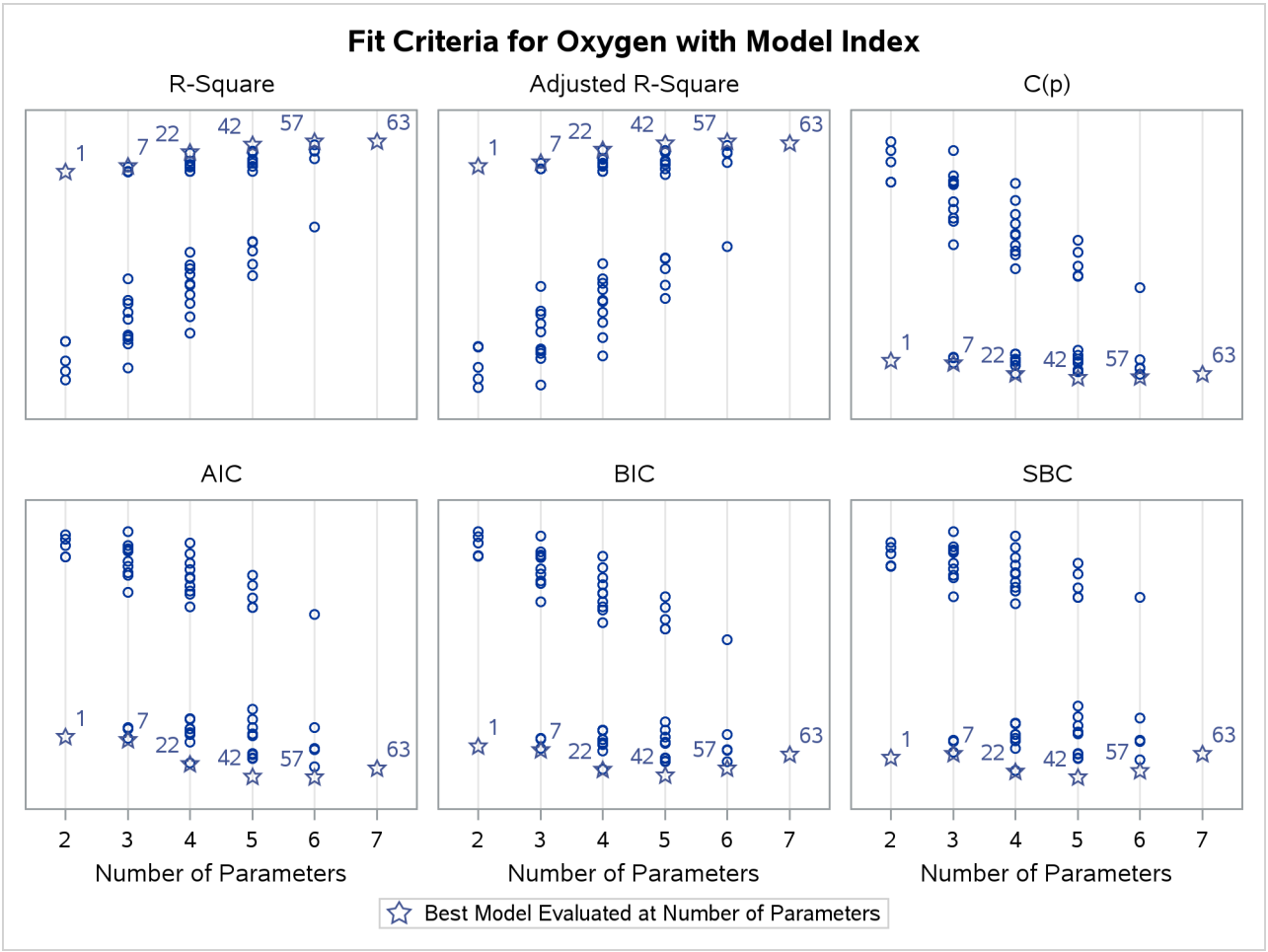

Output 105.2.9 shows the plot of the C_p criterion by number of regressors in the model. Useful reference lines suggested by Mallows (1973) and Hocking (1976) are included on the plot. However, because all possible subset models are included on this plot, the better models are all compressed near the bottom of the plot.

Output 105.2.9 C_p Criterion

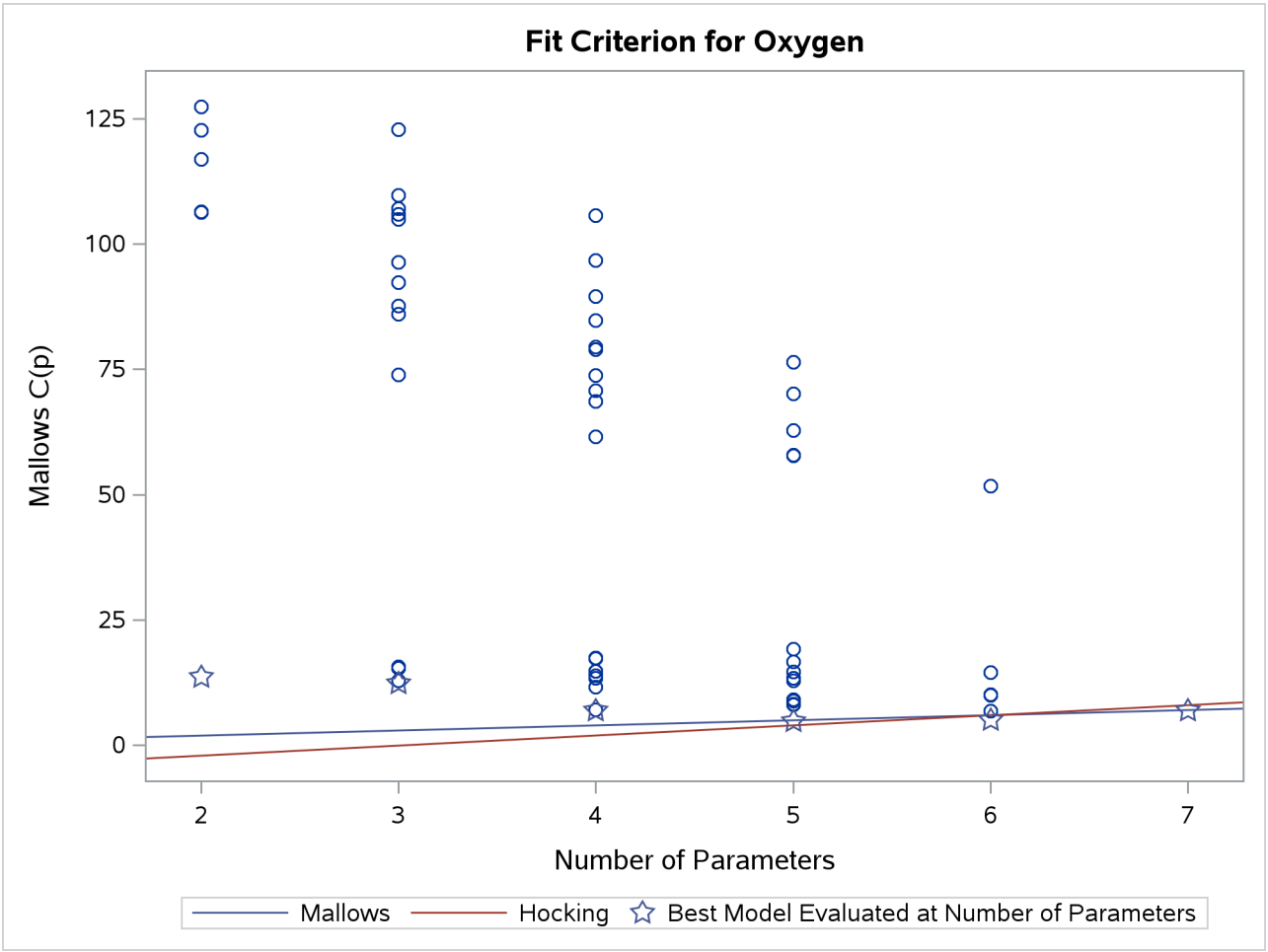

The following statements use the BEST=20 option in the model statement and SELECTION=CP to restrict attention to the models that yield the 20 smallest values of the C_p statistic:

```
proc reg data=fitness plots(only)=cp(label);
   model Oxygen=Age Weight RunTime RunPulse RestPulse MaxPulse
         / selection=cp best=20;
run;

ods graphics off;
```

Output 105.2.10 shows the summary table listing the regressors in the 20 models that yield the smallest C_p values, and Output 105.2.11 presents the results graphically. Reference lines $C_p = 2p - p_{full}$ and $C_p = p$ are shown on this plot. See the PLOTS=CP option on page 8748 for interpretations of these lines. For the Fitness data, these lines indicate that a six-variable model is a reasonable choice for doing parameter estimation, while a five-variable model might be suitable for doing prediction.

Output 105.2.10 C_p Selection Summary: PROC REG

The REG Procedure
Model: MODEL1
Dependent Variable: Oxygen

C(p) Selection Method

Model Index	Number in Model	C(p)	R-Square	Variables in Model
1	4	4.8800	0.8368	Age RunTime RunPulse MaxPulse
2	5	5.1063	0.8480	Age Weight RunTime RunPulse MaxPulse
3	5	6.8461	0.8370	Age RunTime RunPulse RestPulse MaxPulse
4	3	6.9596	0.8111	Age RunTime RunPulse
5	6	7.0000	0.8487	Age Weight RunTime RunPulse RestPulse MaxPulse
6	3	7.1350	0.8100	RunTime RunPulse MaxPulse
7	4	8.1035	0.8165	Age Weight RunTime RunPulse
8	4	8.2056	0.8158	Weight RunTime RunPulse MaxPulse
9	4	8.8683	0.8117	Age RunTime RunPulse RestPulse
10	4	9.0697	0.8104	RunTime RunPulse RestPulse MaxPulse
11	5	9.9348	0.8176	Age Weight RunTime RunPulse RestPulse
12	5	10.1685	0.8161	Weight RunTime RunPulse RestPulse MaxPulse
13	3	11.6167	0.7817	Age RunTime MaxPulse
14	2	12.3894	0.7642	Age RunTime
15	2	12.8372	0.7614	RunTime RunPulse
16	4	12.9039	0.7862	Age Weight RunTime MaxPulse
17	3	13.3453	0.7708	Age Weight RunTime
18	4	13.3468	0.7834	Age RunTime RestPulse MaxPulse
19	1	13.6988	0.7434	RunTime
20	3	13.8974	0.7673	Age RunTime RestPulse

Output 105.2.11 C_p Criterion

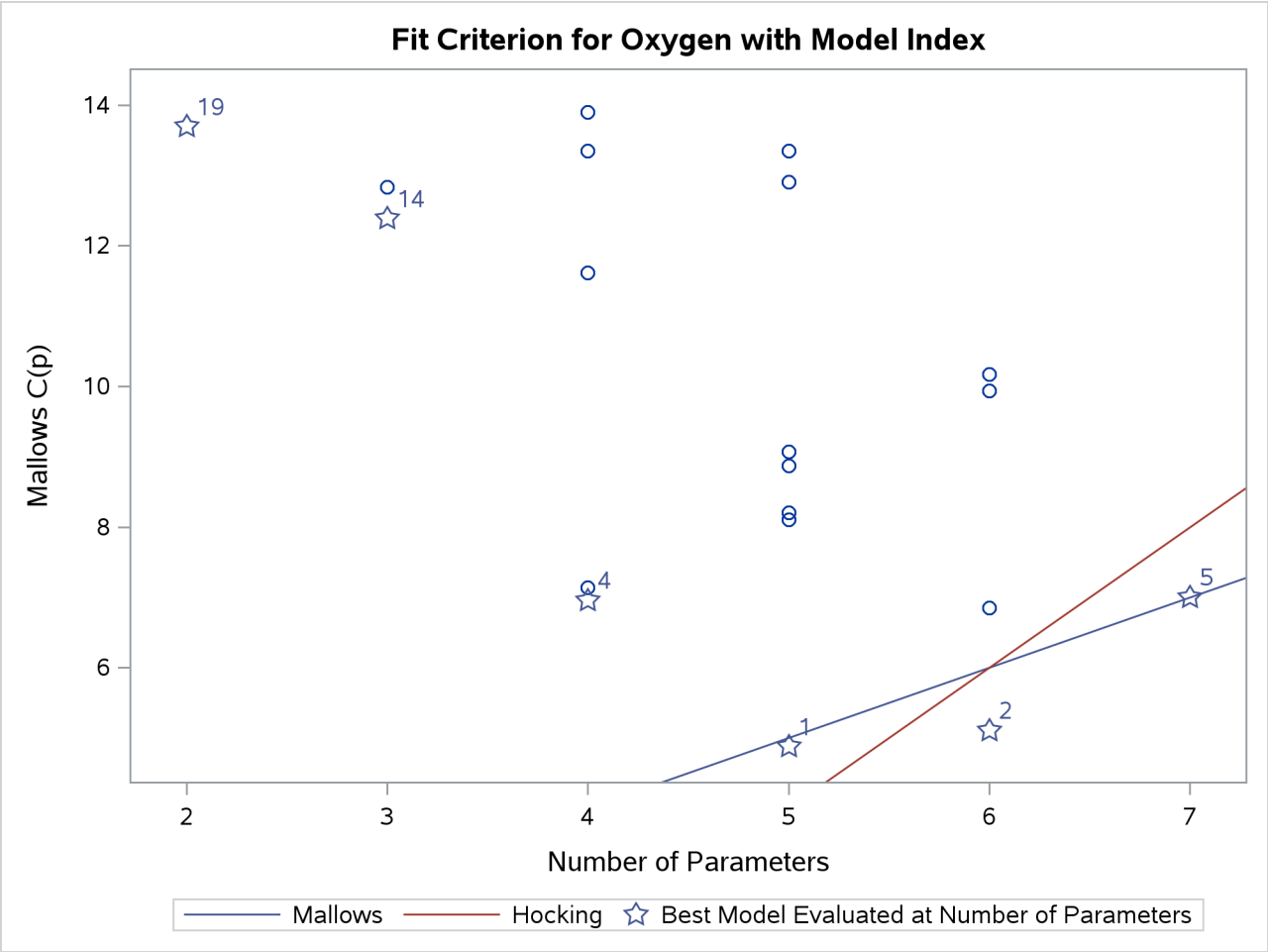

Before making a final decision about which model to use, you would want to perform collinearity diagnostics. Note that, since many different models have been fit and the choice of a final model is based on R square, the statistics are biased and the p-values for the parameter estimates are not valid.

Example 105.3: Predicting Weight by Height and Age

(View the complete code for this example (regex3.sas) in the example repository.)

In this example, the weights of schoolchildren are modeled as a function of their heights and ages. The example shows the use of a BY statement with PROC REG, multiple MODEL statements, and the OUTEST= and OUTSSCP= options, which create data sets. Here are the data:

```
*------------Data on Age, Weight, and Height of Children-------*
| Age (months), height (inches), and weight (pounds) were      |
| recorded for a group of school children.                     |
| From Lewis and Taylor (1967).                                |
*--------------------------------------------------------------*;
```

```
data htwt;
   input sex $ age :3.1 height weight @@;
   datalines;
f 143 56.3  85.0 f 155 62.3 105.0 f 153 63.3 108.0 f 161 59.0  92.0
f 191 62.5 112.5 f 171 62.5 112.0 f 185 59.0 104.0 f 142 56.5  69.0
f 160 62.0  94.5 f 140 53.8  68.5 f 139 61.5 104.0 f 178 61.5 103.5
f 157 64.5 123.5 f 149 58.3  93.0 f 143 51.3  50.5 f 145 58.8  89.0
f 191 65.3 107.0 f 150 59.5  78.5 f 147 61.3 115.0 f 180 63.3 114.0

   ... more lines ...

m 164 66.5 112.0 m 189 65.0 114.0 m 164 61.5 140.0 m 167 62.0 107.5
m 151 59.3  87.0
;
```

Modeling is performed separately for boys and girls. Since the BY statement is used, interactive processing is not possible in this example; no statements can appear after the first RUN statement.

The following statements produce Output 105.3.1 through Output 105.3.4:

```
proc reg outest=est1 outsscp=sscp1 rsquare;
   by sex;
   eq1: model  weight=height;
   eq2: model  weight=height age;
run;

proc print data=sscp1;
   title2 'SSCP type data set';
run;

proc print data=est1;
   title2 'EST type data set';
run;
```

Output 105.3.1 Height and Weight Data: Submodel for Female Children

The REG Procedure
Model: eq1
Dependent Variable: weight

sex=f

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	21507	21507	141.09	<.0001
Error	109	16615	152.42739		
Corrected Total	110	38121			

Root MSE	12.34615	**R-Square**	0.5642
Dependent Mean	98.87838	**Adj R-Sq**	0.5602
Coeff Var	12.48620		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-153.12891	21.24814	-7.21	<.0001
height	1	4.16361	0.35052	11.88	<.0001

Output 105.3.2 Height and Weight Data: Full Model for Female Children

The REG Procedure
Model: eq2
Dependent Variable: weight

sex=f

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	22432	11216	77.21	<.0001
Error	108	15689	145.26700		
Corrected Total	110	38121			

Root MSE	12.05268	**R-Square**	0.5884
Dependent Mean	98.87838	**Adj R-Sq**	0.5808
Coeff Var	12.18939		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-150.59698	20.76730	-7.25	<.0001
height	1	3.60378	0.40777	8.84	<.0001
age	1	1.90703	0.75543	2.52	0.0130

Output 105.3.3 Height and Weight Data: Submodel for Male Children

The REG Procedure
Model: eq1
Dependent Variable: weight

sex=m

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	31126	31126	206.24	<.0001
Error	124	18714	150.92222		
Corrected Total	125	49840			

Root MSE	12.28504	R-Square	0.6245
Dependent Mean	103.44841	Adj R-Sq	0.6215
Coeff Var	11.87552		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-125.69807	15.99362	-7.86	<.0001
height	1	3.68977	0.25693	14.36	<.0001

Output 105.3.4 Height and Weight Data: Full Model for Male Children

The REG Procedure
Model: eq2
Dependent Variable: weight

sex=m

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	32975	16487	120.24	<.0001
Error	123	16866	137.11922		
Corrected Total	125	49840			

Root MSE	11.70979	R-Square	0.6616
Dependent Mean	103.44841	Adj R-Sq	0.6561
Coeff Var	11.31945		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-113.71346	15.59021	-7.29	<.0001
height	1	2.68075	0.36809	7.28	<.0001
age	1	3.08167	0.83927	3.67	0.0004

For both female and male children, the overall F statistics for both models are significant, indicating that the model explains a significant portion of the variation in the data. For females, the full model is

$$\text{weight} = -150.57 + 3.60 \times \text{height} + 1.91 \times \text{age}$$

and for males, the full model is

$$\text{weight} = -113.71 + 2.68 \times \text{height} + 3.08 \times \text{age}$$

The OUTSSCP= data set is shown in Output 105.3.5. Note how the BY groups are separated. Observations with _TYPE_='N' contain the number of observations in the associated BY group. Observations with _TYPE_='SSCP' contain the rows of the uncorrected sums of squares and crossproducts matrix. The observations with _NAME_='Intercept' contain crossproducts for the intercept.

Output 105.3.5 SSCP Matrix

SSCP type data set

Obs	sex	_TYPE_	_NAME_	Intercept	height	weight	age
1	f	SSCP	Intercept	111.0	6718.40	10975.50	1824.90
2	f	SSCP	height	6718.4	407879.32	669469.85	110818.32
3	f	SSCP	weight	10975.5	669469.85	1123360.75	182444.95
4	f	SSCP	age	1824.9	110818.32	182444.95	30363.81
5	f	N		111.0	111.00	111.00	111.00
6	m	SSCP	Intercept	126.0	7825.00	13034.50	2072.10
7	m	SSCP	height	7825.0	488243.60	817919.60	129432.57
8	m	SSCP	weight	13034.5	817919.60	1398238.75	217717.45
9	m	SSCP	age	2072.1	129432.57	217717.45	34515.95
10	m	N		126.0	126.00	126.00	126.00

The OUTEST= data set is displayed in Output 105.3.6; again, the BY groups are separated. The _MODEL_ column contains the labels for models from the MODEL statements. If no labels are specified, the defaults MODEL1 and MODEL2 would appear as values for _MODEL_. Note that _TYPE_='PARMS' for all observations, indicating that all observations contain parameter estimates. The _DEPVAR_ column displays the dependent variable, and the _RMSE_ column gives the root mean square error for the associated model. The Intercept column gives the estimate for the intercept for the associated model, and variables with the same name as variables in the original data set (height, age) give parameter estimates for those variables. The dependent variable, weight, is shown with a value of –1. The _IN_ column contains the number of regressors in the model not including the intercept; _P_ contains the number of parameters in the model; _EDF_ contains the error degrees of freedom; and _RSQ_ contains the R square statistic. Finally, note that the _IN_, _P_, _EDF_, and _RSQ_ columns appear in the OUTEST= data set since the RSQUARE option is specified in the PROC REG statement.

Output 105.3.6 OUTEST Data Set

EST type data set

Obs	sex	_MODEL_	_TYPE_	_DEPVAR_	_RMSE_	Intercept	height	weight	age	_IN_	_P_	_EDF_	_RSQ_
1	f	eq1	PARMS	weight	12.3461	-153.129	4.16361	-1	.	1	2	109	0.56416
2	f	eq2	PARMS	weight	12.0527	-150.597	3.60378	-1	1.90703	2	3	108	0.58845
3	m	eq1	PARMS	weight	12.2850	-125.698	3.68977	-1	.	1	2	124	0.62451
4	m	eq2	PARMS	weight	11.7098	-113.713	2.68075	-1	3.08167	2	3	123	0.66161

Example 105.4: Regression with Quantitative and Qualitative Variables

(View the complete code for this example (regex4.sas) in the example repository.)

At times it is desirable to have independent variables in the model that are qualitative rather than quantitative. This is easily handled in a regression framework. Regression uses qualitative variables to distinguish between populations. There are two main advantages of fitting both populations in one model. You gain the ability to test for different slopes or intercepts in the populations, and more degrees of freedom are available for the analysis.

Regression with qualitative variables is different from analysis of variance and analysis of covariance. Analysis of variance uses qualitative independent variables only. Analysis of covariance uses quantitative variables in addition to the qualitative variables in order to account for correlation in the data and reduce MSE; however, the quantitative variables are not of primary interest and merely improve the precision of the analysis.

Consider the case where Y_i is the dependent variable, $\mathbf{X1}_i$ is a quantitative variable, $\mathbf{X2}_i$ is a qualitative variable taking on values 0 or 1, and $\mathbf{X1}_i\mathbf{X2}_i$ is the interaction. The variable $\mathbf{X2}_i$ is called a dummy, binary, or indicator variable. With values 0 or 1, it distinguishes between two populations. The model is of the form

$$\mathbf{Y}_i = \beta_0 + \beta_1\mathbf{X1}_i + \beta_2\mathbf{X2}_i + \beta_3\mathbf{X1}_i\mathbf{X2}_i + \epsilon_i$$

for the observations $i = 1, 2, \ldots, n$. The parameters to be estimated are β_0, β_1, β_2, and β_3. The number of dummy variables used is one less than the number of qualitative levels. This yields a nonsingular $\mathbf{X}'\mathbf{X}$ matrix. See Chapter 10 of Neter, Wasserman, and Kutner (1990) for more details.

An example from Neter, Wasserman, and Kutner (1990) follows. An economist is investigating the relationship between the size of an insurance firm and the speed at which it implements new insurance innovations. He believes that the type of firm might affect this relationship and suspects that there might be some interaction between the size and type of firm. The dummy variable in the model enables the two firms to have different intercepts. The interaction term enables the firms to have different slopes as well.

In this study, $\mathbf{Y}_i$ is the number of months from the time the first firm implemented the innovation to the time it was implemented by the ith firm. The variable $\mathbf{X1}_i$ is the size of the firm, measured in total assets of the firm. The variable $\mathbf{X2}_i$ denotes the firm type; it is 0 if the firm is a mutual fund company and 1 if the firm is a stock company. The dummy variable enables each firm type to have a different intercept and slope.

The previous model can be broken down into a model for each firm type by plugging in the values for $\mathbf{X2}_i$. If $\mathbf{X2}_i = 0$, the model is

$$\mathbf{Y}_i = \beta_0 + \beta_1 \mathbf{X1}_i + \boldsymbol{\epsilon}_i$$

This is the model for a mutual company. If $\mathbf{X2}_i = 1$, the model for a stock firm is

$$\mathbf{Y}_i = (\beta_0 + \beta_2) + (\beta_1 + \beta_3)\mathbf{X1}_i + \boldsymbol{\epsilon}_i$$

This model has intercept $\beta_0 + \beta_2$ and slope $\beta_1 + \beta_3$.

The data[1] follow. Note that the interaction term is created in the DATA step since polynomial effects such as size*type are not allowed in the MODEL statement in the REG procedure.

```
title 'Regression with Quantitative and Qualitative Variables';
data insurance;
   input time size type @@;
   sizetype=size*type;
   datalines;
17 151 0   26  92 0   21 175 0   30  31 0   22 104 0   0  277 0   12 210 0
19 120 0    4 290 0   16 238 0   28 164 1   15 272 1   11 295 1   38  68 1
31  85 1   21 224 1   20 166 1   13 305 1   30 124 1   14 246 1
;
```

The following statements begin the analysis and produce the ANOVA table in Output 105.4.1:

```
proc reg data=insurance;
   model time = size type sizetype;
run;
```

Output 105.4.1 ANOVA Table and Parameter Estimates

Regression with Quantitative and Qualitative Variables

The REG Procedure
Model: MODEL1
Dependent Variable: time

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	1504.41904	501.47301	45.49	<.0001
Error	16	176.38096	11.02381		
Corrected Total	19	1680.80000			

Root MSE	3.32021	R-Square	0.8951
Dependent Mean	19.40000	Adj R-Sq	0.8754
Coeff Var	17.11450		

[1]From Neter, J., et al., *Applied Linear Statistical Models*, Third Edition, Copyright (c) 1990, Richard D. Irwin. Reprinted with permission of The McGraw-Hill Companies.

Output 105.4.1 *continued*

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	33.83837	2.44065	13.86	<.0001
size	1	-0.10153	0.01305	-7.78	<.0001
type	1	8.13125	3.65405	2.23	0.0408
sizetype	1	-0.00041714	0.01833	-0.02	0.9821

The overall F statistic is significant ($F = 45.490$, $p < 0.0001$). The interaction term is not significant ($t = -0.023$, $p = 0.9821$). Hence, this term should be removed and the model refitted, as shown in the following statements:

```
    delete sizetype;
    print;
run;
```

The DELETE statement removes the interaction term (sizetype) from the model. The new ANOVA and parameter estimates tables are shown in Output 105.4.2.

Output 105.4.2 ANOVA Table and Parameter Estimates

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	1504.41333	752.20667	72.50	<.0001
Error	17	176.38667	10.37569		
Corrected Total	19	1680.80000			

Root MSE	3.22113	R-Square	0.8951
Dependent Mean	19.40000	Adj R-Sq	0.8827
Coeff Var	16.60377		

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	33.87407	1.81386	18.68	<.0001
size	1	-0.10174	0.00889	-11.44	<.0001
type	1	8.05547	1.45911	5.52	<.0001

The overall F statistic is still significant ($F = 72.497$, $p < 0.0001$). The intercept and the coefficients associated with size and type are significantly different from zero ($t = 18.675$, $p < 0.0001$; $t = -11.443$, $p < 0.0001$; $t = 5.521$, $p < 0.0001$, respectively). Notice that the R square did not change with the omission of the interaction term.

The fitted model is

$$\text{time} = 33.87 - 0.102 \times \text{size} + 8.055 \times \text{type}$$

The fitted model for a mutual fund company ($X2_i = 0$) is

$$\text{time} = 33.87 - 0.102 \times \text{size}$$

and the fitted model for a stock company ($X2_i = 1$) is

$$\text{time} = (33.87 + 8.055) - 0.102 \times \text{size}$$

So the two models have different intercepts but the same slope.

The following statements first use an OUTPUT statement to save the residuals and predicted values from the new model in the OUT= data set. Next PROC SGPLOT is used to produce Output 105.4.3, which plots residuals versus predicted values. The firm type is used as the plot symbol; this can be useful in determining if the firm types have different residual patterns.

```
   output out=out r=r p=p;
run;

proc sgplot data=out;
   scatter x=p y=r / markerchar=type group=type;
run;
```

Output 105.4.3 Plot of Residual vs. Predicted Values

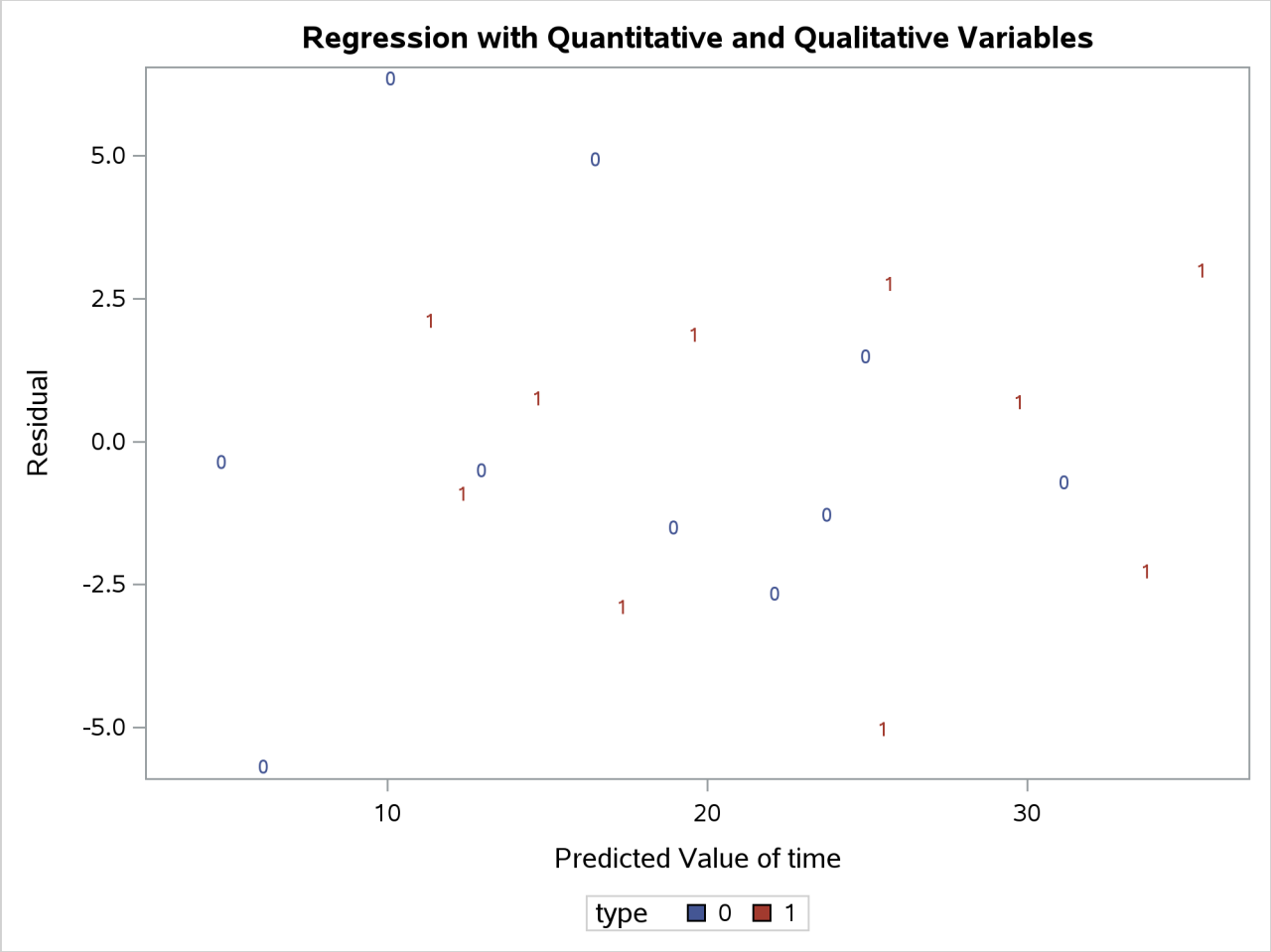

The residuals show no major trend. Neither firm type by itself shows a trend either. This indicates that the model is satisfactory.

The following statements produce the plot of the predicted values versus size that appears in Output 105.4.4, where the firm type is again used as the plotting symbol:

```
proc sgplot data=out;
   scatter x=size y=p / markerchar=type group=type;
run;
```

Output 105.4.4 Plot of Predicted vs. Size

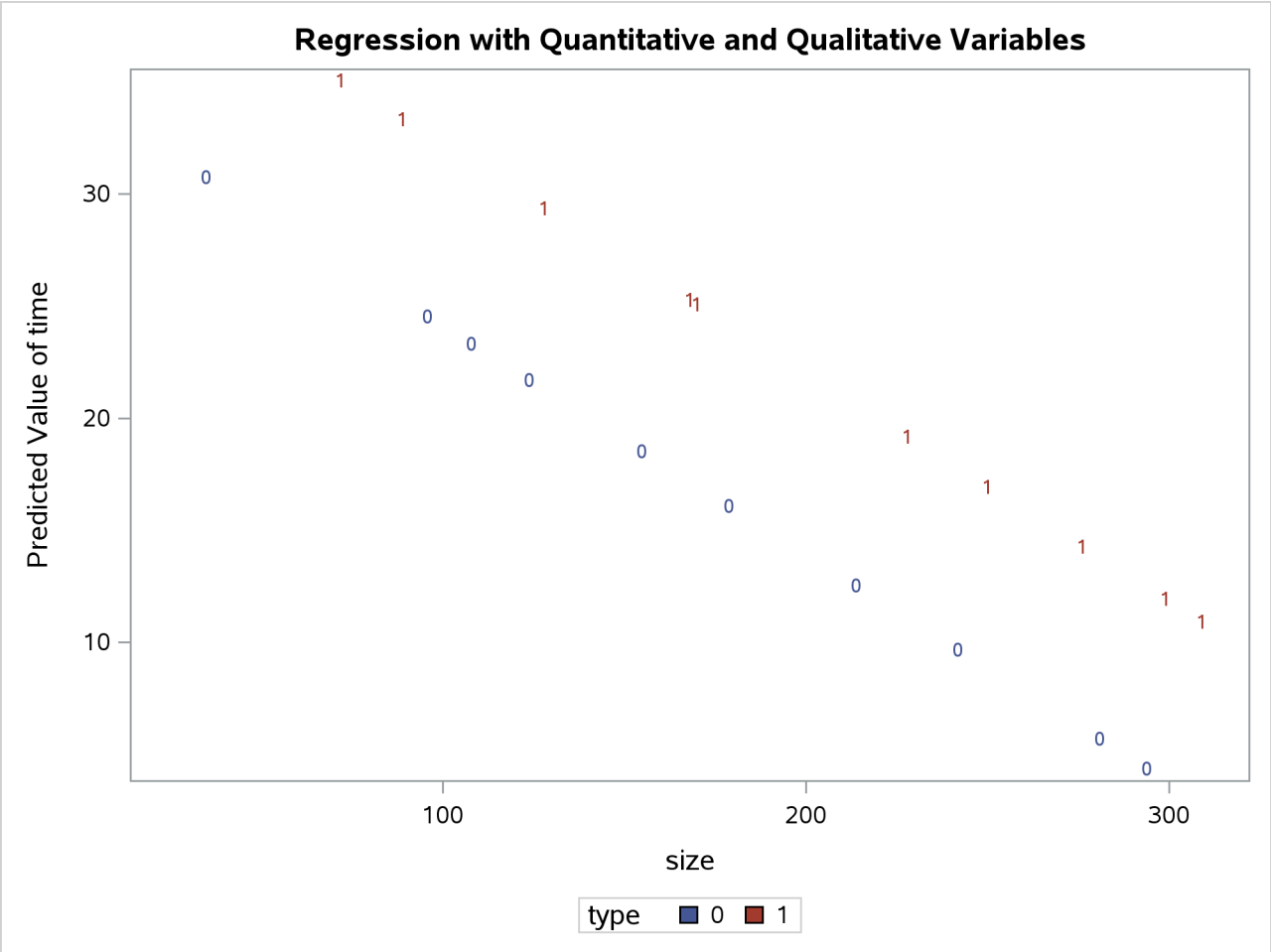

The different intercepts are very evident in this plot.

Example 105.5: Ridge Regression for Acetylene Data

(View the complete code for this example (regex5.sas) in the example repository.)

This example uses the acetylene data in Marquardt and Snee (1975) to illustrate the RIDGEPLOT and OUTVIF options. Here are the data:

```
data acetyl;
   input x1-x4 @@;
   x1x2 = x1 * x2;
   x1x1 = x1 * x1;
   label x1   = 'reactor temperature(celsius)'
         x2   = 'h2 to n-heptone ratio'
         x3   = 'contact time(sec)'
         x4   = 'conversion percentage'
         x1x2= 'temperature-ratio interaction'
         x1x1= 'squared temperature';
```

```
   datalines;
1300  7.5 .012 49   1300  9   .012  50.2 1300 11 .0115 50.5
1300 13.5 .013 48.5 1300 17   .0135 47.5 1300 23 .012  44.5
1200  5.3 .04  28   1200  7.5 .038  31.5 1200 11 .032  34.5
1200 13.5 .026 35   1200 17   .034  38   1200 23 .041  38.5
1100  5.3 .084 15   1100  7.5 .098  17   1100 11 .092  20.5
1100 17   .086 29.5
;

ods graphics on;

proc reg data=acetyl outvif
         outest=b ridge=0 to 0.02 by .002;
   model x4=x1 x2 x3 x1x2 x1x1;
run;
proc print data=b;
run;
```

When ODS Graphics is enabled and you request ridge regression by using the RIDGE= option in the PROC REG statement, PROC REG produces a panel showing variance inflation factors (VIF) in the upper plot in the panel and ridge traces in the lower plot. This panel is shown in Output 105.5.1.

Output 105.5.1 Ridge Regression and VIF Traces

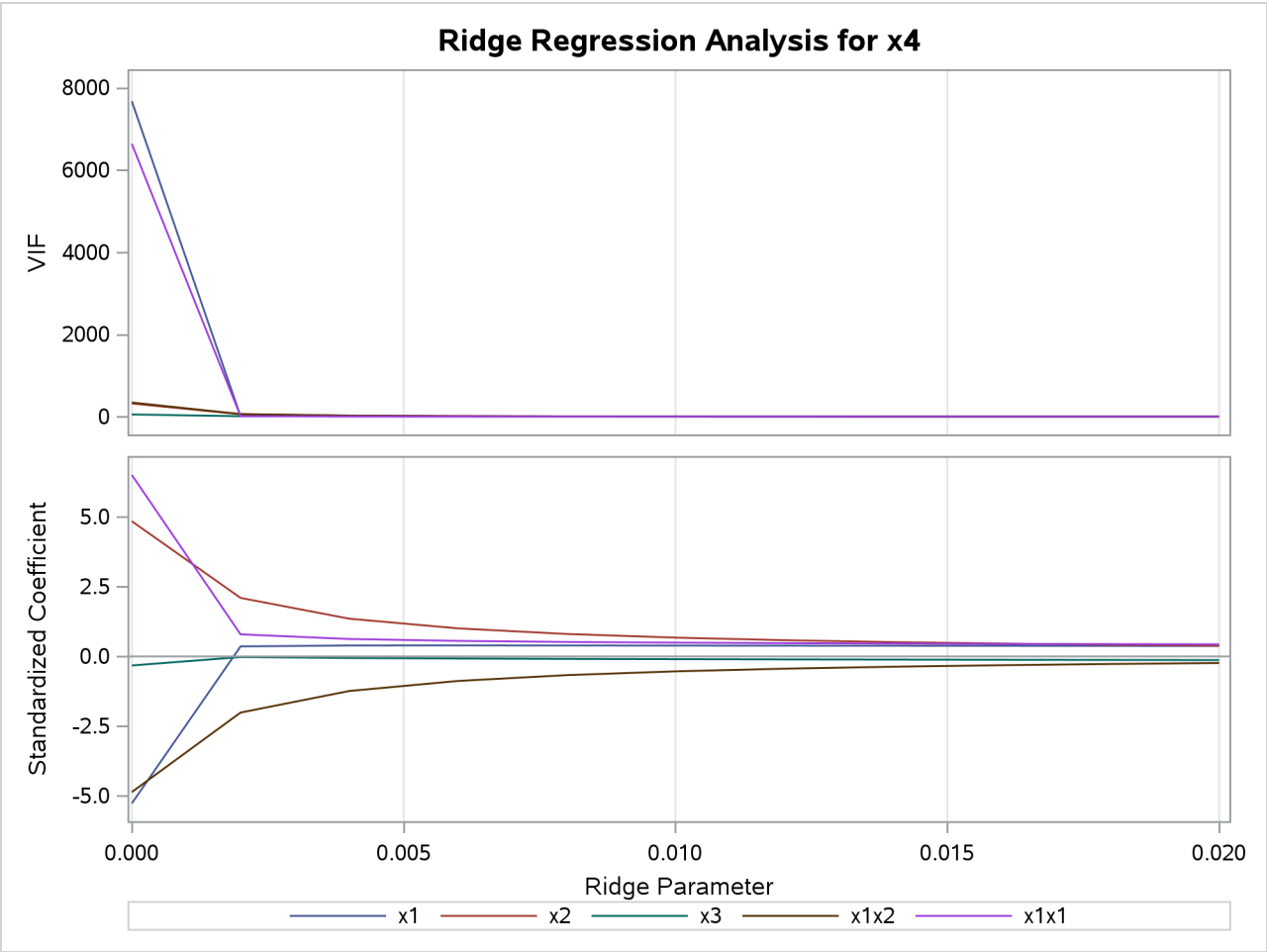

The OUTVIF option outputs the variance inflation factors to the OUTEST= data set that is shown in Output 105.5.2.

Output 105.5.2 OUTEST Data Set Showing VIF Values

Obs	_MODEL_	_TYPE_	_DEPVAR_	_RIDGE_	_PCOMIT_	_RMSE_	Intercept	x1	x2	x3	x1x2	x1x1	x4
1	MODEL1	PARMS	x4	.	.	1.15596	390.538	-0.78	10.174	-121.626	-0.008	0.00	-1
2	MODEL1	RIDGEVIF	x4	0.000	.	.	.	7682.37	320.022	53.525	344.545	6643.32	-1
3	MODEL1	RIDGE	x4	0.000	.	1.15596	390.538	-0.78	10.174	-121.626	-0.008	0.00	-1
4	MODEL1	RIDGEVIF	x4	0.002	.	.	.	11.18	58.731	10.744	63.208	11.22	-1
5	MODEL1	RIDGE	x4	0.002	.	2.69721	-103.388	0.05	4.404	-9.065	-0.003	0.00	-1
6	MODEL1	RIDGEVIF	x4	0.004	.	.	.	4.36	23.939	9.996	25.744	5.15	-1
7	MODEL1	RIDGE	x4	0.004	.	3.22340	-93.797	0.06	2.839	-21.338	-0.002	0.00	-1
8	MODEL1	RIDGEVIF	x4	0.006	.	.	.	2.93	13.011	9.383	13.976	3.81	-1
9	MODEL1	RIDGE	x4	0.006	.	3.47752	-87.687	0.06	2.110	-28.447	-0.001	0.00	-1
10	MODEL1	RIDGEVIF	x4	0.008	.	.	.	2.36	8.224	8.838	8.821	3.23	-1
11	MODEL1	RIDGE	x4	0.008	.	3.62677	-83.593	0.06	1.689	-33.377	-0.001	0.00	-1
12	MODEL1	RIDGEVIF	x4	0.010	.	.	.	2.04	5.709	8.343	6.112	2.89	-1
13	MODEL1	RIDGE	x4	0.010	.	3.72505	-80.603	0.06	1.414	-37.177	-0.001	0.00	-1
14	MODEL1	RIDGEVIF	x4	0.012	.	.	.	1.84	4.226	7.891	4.514	2.65	-1
15	MODEL1	RIDGE	x4	0.012	.	3.79477	-78.276	0.06	1.221	-40.297	-0.001	0.00	-1
16	MODEL1	RIDGEVIF	x4	0.014	.	.	.	1.69	3.279	7.476	3.493	2.46	-1
17	MODEL1	RIDGE	x4	0.014	.	3.84693	-76.381	0.06	1.078	-42.965	-0.001	0.00	-1
18	MODEL1	RIDGEVIF	x4	0.016	.	.	.	1.57	2.637	7.094	2.801	2.31	-1
19	MODEL1	RIDGE	x4	0.016	.	3.88750	-74.785	0.06	0.968	-45.309	-0.001	0.00	-1
20	MODEL1	RIDGEVIF	x4	0.018	.	.	.	1.47	2.182	6.741	2.310	2.18	-1
21	MODEL1	RIDGE	x4	0.018	.	3.92004	-73.407	0.06	0.880	-47.407	-0.000	0.00	-1
22	MODEL1	RIDGEVIF	x4	0.020	.	.	.	1.39	1.847	6.415	1.949	2.06	-1
23	MODEL1	RIDGE	x4	0.020	.	3.94679	-72.193	0.06	0.809	-49.310	-0.000	0.00	-1

If you want to obtain separate plots containing the ridge traces and VIF traces, you can specify the UNPACK suboption in the PLOTS=RIDGE option. You can also request that one or both of the VIF axis and ridge parameter axis be displayed on a logarithmic scale. You can see in Output 105.5.1 that the VIF traces for several of the parameters are nearly indistinguishable when displayed on a linear scale. The following code illustrates how you obtain separate VIF and ridge traces with the VIF values displayed on a logarithmic scale. Note that you can obtain plots of VIF values even though you do not specify the OUTVIF option in the PROC REG statement.

```
proc reg data=acetyl plots(only)=ridge(unpack VIFaxis=log)
         outest=b ridge=0 to 0.02 by .002;
   model x4=x1 x2 x3 x1x2 x1x1;
run;

ods graphics off;
```

The requested plots are shown in Output 105.5.3 and Output 105.5.4.

Output 105.5.3 VIF Traces

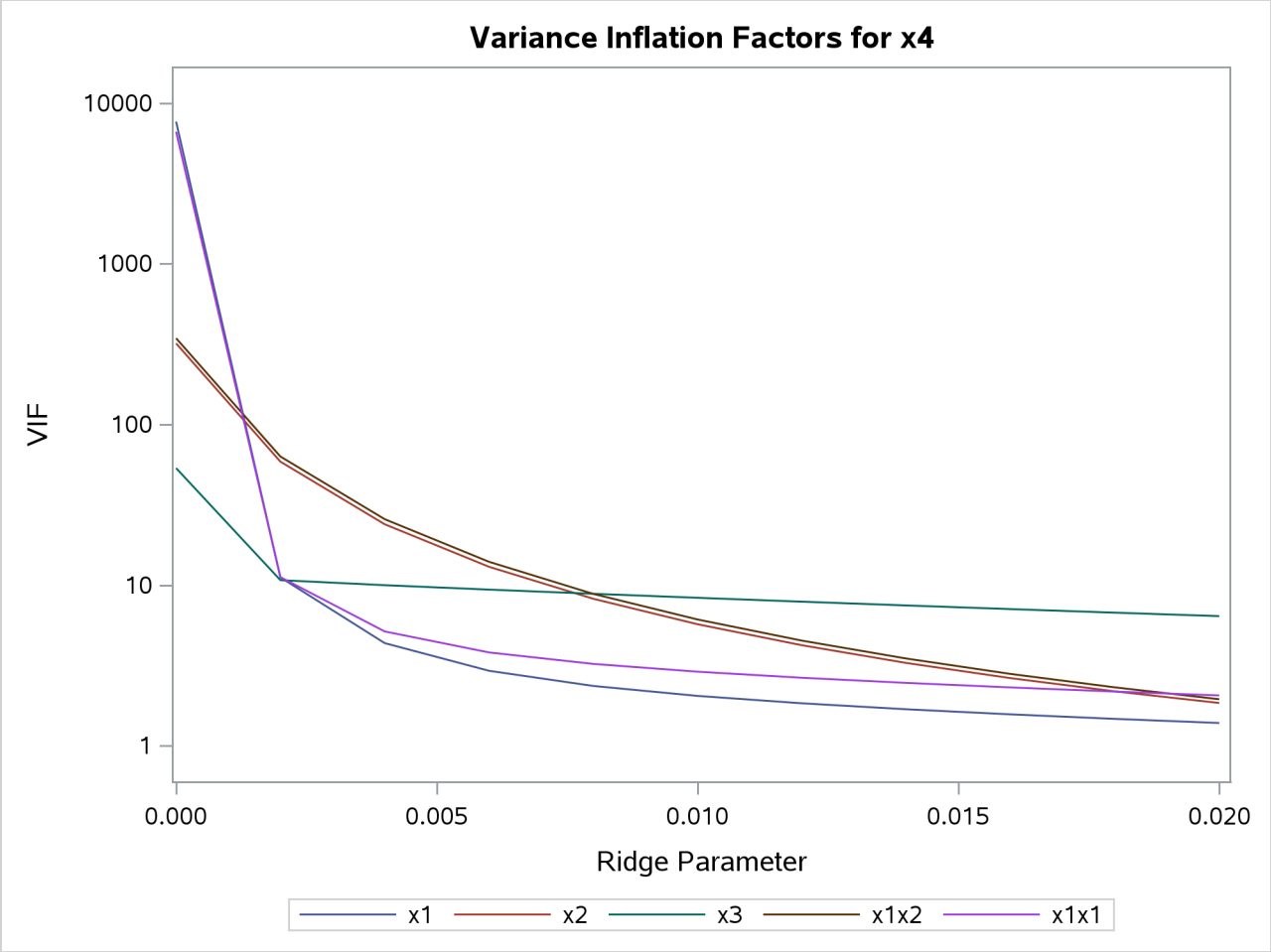

Output 105.5.4 Ridge Traces

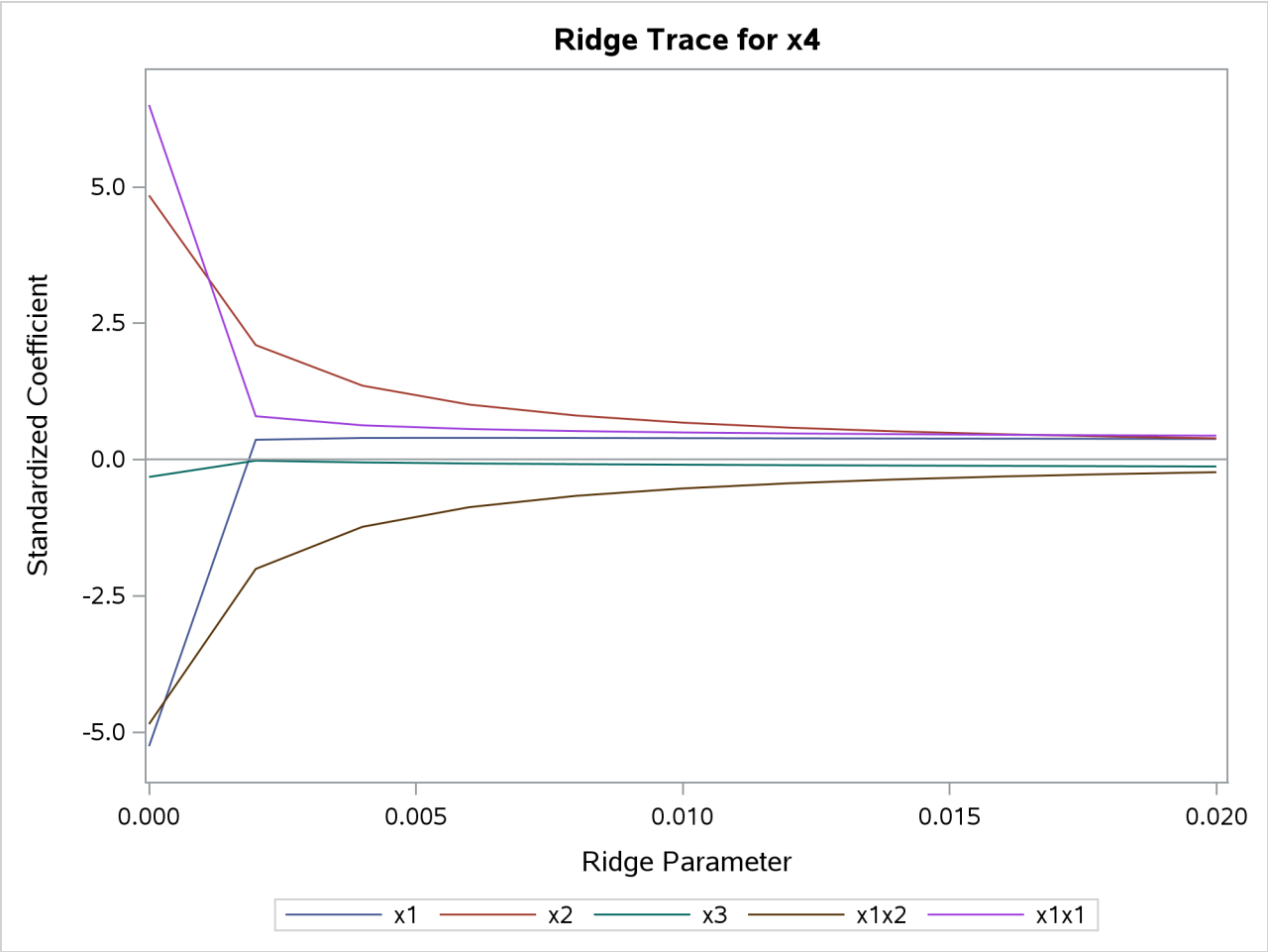

Example 105.6: Chemical Reaction Response

(View the complete code for this example (regex6.sas) in the example repository.)

This example shows how you can use lack-of-fit tests with the REG procedure. See the section "Testing for Lack of Fit" on page 8825 for details about lack-of-fit tests.

In a study of the percentage of raw material that responds in a reaction, researchers identified the following five factors:

- the feed rate of the chemicals (FeedRate), ranging from 10 to 15 liters per minute
- the percentage of the catalyst (Catalyst), ranging from 1% to 2%
- the agitation rate of the reactor (AgitRate), ranging from 100 to 120 revolutions per minute
- the temperature (Temperature), ranging from 140 to 180 degrees Celsius
- the concentration (Concentration), ranging from 3% to 6%

The following data set contains the results of an experiment designed to estimate main effects for all factors:

```
data reaction;
   input FeedRate Catalyst AgitRate Temperature
         Concentration ReactionPercentage;
   datalines;
10.0   1.0  100   140  6.0   37.5
10.0   1.0  120   180  3.0   28.5
10.0   2.0  100   180  3.0   40.4
10.0   2.0  120   140  6.0   48.2
15.0   1.0  100   180  6.0   50.7
15.0   1.0  120   140  3.0   28.9
15.0   2.0  100   140  3.0   43.5
15.0   2.0  120   180  6.0   64.5
12.5   1.5  110   160  4.5   39.0
12.5   1.5  110   160  4.5   40.3
12.5   1.5  110   160  4.5   38.7
12.5   1.5  110   160  4.5   39.7
;
```

The first eight runs of this experiment enable orthogonal estimation of the main effects for all factors. The last four comprise four replicates of the centerpoint.

The following statements fit a linear model. Because this experiment includes replications, you can test for lack of fit by using the LACKFIT option in the MODEL statement.

```
proc reg data=reaction;
   model  ReactionPercentage=FeedRate Catalyst AgitRate
                             Temperature Concentration / lackfit;
run;
```

Output 105.6.1 shows that the lack of fit for the linear model is significant, indicating that a more complex model is required. Models that include interactions should be investigated. In this case, this will require additional experimentation to obtain appropriate data for estimating the effects.

Output 105.6.1 Analysis of Variance

The REG Procedure
Model: MODEL1
Dependent Variable: ReactionPercentage

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	990.27000	198.05400	33.29	0.0003
Error	6	35.69917	5.94986		
Lack of Fit	3	34.15167	11.38389	22.07	0.0151
Pure Error	3	1.54750	0.51583		
Corrected Total	11	1025.96917			

Root MSE	2.43923	R-Square	0.9652
Dependent Mean	41.65833	Adj R-Sq	0.9362
Coeff Var	5.85533		

References

Akaike, H. (1969). "Fitting Autoregressive Models for Prediction." *Annals of the Institute of Statistical Mathematics* 21:243–247.

Allen, D. M. (1971). "Mean Square Error of Prediction as a Criterion for Selecting Variables." *Technometrics* 13:469–475.

Allen, D. M., and Cady, F. B. (1982). *Analyzing Experimental Data by Regression*. Belmont, CA: Lifetime Learning Publications.

Amemiya, T. (1976). *Selection of Regressors*. Technical Report 225, Stanford University, Stanford, CA.

Belsley, D. A., Kuh, E., and Welsch, R. E. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. New York: John Wiley & Sons.

Berk, K. N. (1977). "Tolerance and Condition in Regression Computations." *Journal of the American Statistical Association* 72:863–866.

Bock, R. D. (1975). *Multivariate Statistical Methods in Behavioral Research*. New York: McGraw-Hill.

Box, G. E. P. (1966). "The Use and Abuse of Regression." *Technometrics* 8:625–629.

Cleveland, W. S. (1993). *Visualizing Data*. Summit, NJ: Hobart Press.

Cook, R. D. (1977). "Detection of Influential Observations in Linear Regression." *Technometrics* 19:15–18.

Cook, R. D. (1979). "Influential Observations in Linear Regression." *Journal of the American Statistical Association* 74:169–174.

Cook, R. D., and Weisberg, S. (1997). "Graphics for Assessing the Adequacy of Regression Models." *Journal of the American Statistical Association* 92:490–499.

Daniel, C., and Wood, F. (1980). *Fitting Equations to Data*. Rev. ed. New York: John Wiley & Sons.

Darlington, R. B. (1968). "Multiple Regression in Psychological Research and Practice." *Psychological Bulletin* 69:161–182.

Draper, N. R., and Smith, H. (1981). *Applied Regression Analysis*. 2nd ed. New York: John Wiley & Sons.

Durbin, J., and Watson, G. S. (1951). "Testing for Serial Correlation in Least Squares Regression." *Biometrika* 37:409–428.

Freund, R. J., and Littell, R. C. (1986). *SAS System for Regression*. 1986 ed. Cary, NC: SAS Institute Inc.

Furnival, G. M., and Wilson, R. W. (1974). "Regression by Leaps and Bounds." *Technometrics* 16:499–511.

Goodnight, J. H. (1979). "A Tutorial on the Sweep Operator." *American Statistician* 33:149–158.

Hocking, R. R. (1976). "The Analysis and Selection of Variables in a Linear Regression." *Biometrics* 32:1–50.

Johnston, J. (1972). *Econometric Methods*. 2nd ed. New York: McGraw-Hill.

Judge, G. G., Griffiths, W. E., Hill, R. C., and Lee, T.-C. (1980). *The Theory and Practice of Econometrics*. New York: John Wiley & Sons.

Judge, G. G., Griffiths, W. E., Hill, R. C., Lütkepohl, H., and Lee, T.-C. (1985). *The Theory and Practice of Econometrics*. 2nd ed. New York: John Wiley & Sons.

Kennedy, W. J., Jr., and Gentle, J. E. (1980). *Statistical Computing*. New York: Marcel Dekker.

LaMotte, L. R. (1994). "A Note on the Role of Independence in *t* Statistics Constructed from Linear Statistics in Regression Models." *American Statistician* 48:238–240.

Lewis, T., and Taylor, L. R. (1967). *Introduction to Experimental Ecology*. New York: Academic Press.

Long, J. S., and Ervin, L. H. (2000). "Using Heteroscedasticity Consistent Standard Errors in the Linear Regression Model." *American Statistician* 54:217–224.

Lord, F. M. (1950). *Efficiency of Prediction When a Progression Equation from One Sample Is Used in a New Sample*. Research bulletin, Educational Testing Service, Princeton, NJ.

MacKinnon, J. G., and White, H. (1985). "Some Heteroskedasticity-Consistent Covariance Matrix Estimators with Improved Finite Sample Properties." *Journal of Econometrics* 29:305–325.

Mallows, C. L. (1967). "Choosing a Subset Regression." Bell Telephone Laboratories.

Mallows, C. L. (1973). "Some Comments on C_P." *Technometrics* 15:661–675.

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis*. London: Academic Press.

Marquardt, D. W., and Snee, R. D. (1975). "Ridge Regression in Practice." *American Statistician* 29:3–20.

Morrison, D. F. (1976). *Multivariate Statistical Methods*. 2nd ed. New York: McGraw-Hill.

Mosteller, F., and Tukey, J. W. (1977). *Data Analysis and Regression*. Reading, MA: Addison-Wesley.

Neter, J., Wasserman, W., and Kutner, M. H. (1990). *Applied Linear Statistical Models*. 3rd ed. Homewood, IL: Irwin.

Nicholson, G. E., Jr. (1948). "The Application of a Regression Equation to a New Sample." Ph.D. diss., University of North Carolina at Chapel Hill.

Pillai, K. C. S. (1960). *Statistical Table for Tests of Multivariate Hypotheses*. Manila: University of Philippines Statistical Center.

Pindyck, R. S., and Rubinfeld, D. L. (1981). *Econometric Models and Econometric Forecasts*. 2nd ed. New York: McGraw-Hill.

Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.

Rao, C. R. (1973). *Linear Statistical Inference and Its Applications*. 2nd ed. New York: John Wiley & Sons.

Rawlings, J. O., Pantula, S. G., and Dickey, D. A. (1998). *Applied Regression Analysis: A Research Tool*. 2nd ed. New York: Springer-Verlag.

Reichler, J. L., ed. (1987). *The 1987 Baseball Encyclopedia Update*. New York: Macmillan.

Rothman, D. (1968). "Letter to the Editor." *Technometrics* 10:432.

Sall, J. P. (1981). *SAS Regression Applications*. Technical Report A-102, SAS Institute Inc., Cary, NC.

Sawa, T. (1978). "Information Criteria for Discriminating among Alternative Regression Models." *Econometrica* 46:1273–1282.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Stein, C. (1960). "Multiple Regression." In *Contributions to Probability and Statistics: Essays in Honor of Harold Hotelling*, edited by I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow, and H. B. Mann, 264–305. Stanford, CA: Stanford University Press.

Time Inc. (1987). "What They Make." *Sports Illustrated* (April 20): 54–81.

Timm, N. H. (1975). *Multivariate Analysis with Applications in Education and Psychology*. Monterey, CA: Brooks/Cole.

Weisberg, S. (1980). *Applied Linear Regression*. New York: John Wiley & Sons.

White, H. (1980). "A Heteroskedasticity-Consistent Covariance Matrix Estimator and a Direct Test for Heteroskedasticity." *Econometrica* 48:817–838.

Chapter 106
The RMSTREG Procedure

Contents

Overview: RMSTREG Procedure . . . **8912**
- Comparison with Other Procedures . . . 8912

Getting Started: RMSTREG Procedure . . . **8913**

Syntax: RMSTREG Procedure . . . **8917**
- PROC RMSTREG Statement . . . 8917
- BY Statement . . . 8919
- CLASS Statement . . . 8919
- ESTIMATE Statement . . . 8923
- FREQ Statement . . . 8924
- LSMEANS Statement . . . 8924
- LSMESTIMATE Statement . . . 8925
- MODEL Statement . . . 8927
- SLICE Statement . . . 8929
- STORE Statement . . . 8929
- TEST Statement . . . 8930

Details: RMSTREG Procedure . . . **8930**
- Concepts and Basic Estimators . . . 8930
- Supported Models . . . 8931
- Specification of Effects . . . 8932
- Parameterization Used in PROC RMSTREG . . . 8933
- Type 3 Tests and Joint Tests . . . 8933
- Confidence Intervals and Predicted Values . . . 8934
- Pseudovalue Regression . . . 8935
- Inverse Probability Censoring Weighting Estimation . . . 8936
- Fitting Algorithm . . . 8938
- Missing Values . . . 8939
- Displayed Output . . . 8939
- ODS Table Names . . . 8941
- ODS Graphics . . . 8941

Examples: RMSTREG Procedure . . . **8942**
- Example 106.1: Comparison of PROC RMSTREG with Other Procedures . . . 8942
- Example 106.2: Comparing the RMSTs of Two Groups . . . 8945
- Example 106.3: Making Model-Based Inferences . . . 8949

References . . . **8952**

Overview: RMSTREG Procedure

The RMSTREG procedure analyzes time-to-event data by using regression with respect to the restricted mean survival time (RMST), using specialized methods such as those developed by Andersen, Hansen, and Klein (2004) and Tian, Zhao, and Wei (2014). The RMST is defined as the expected value of the time-to-event variable up to a prespecified time point τ.

For a prospective study in which the time to an event (or survival) is of interest, it is common practice to summarize the outcome in terms of the mean or median survival rate. However, if the last observation is censored, then the mean cannot be reliably estimated; and when not enough events occur, the median can be inestimable. The survival rate at a specified time is another common summary statistic, and although it can be less problematic than the mean or median, it does not provide an overall summary of the time-to-event outcome. The restricted mean survival time, sometimes called the restricted mean event time, is an alternative measure that is more often reliably estimable than the mean and median survival time in certain situations. Also, it provides a summary of the whole survival curve up to a time horizon, in contrast to the survival rate at a specified time (Royston and Parmar 2013; Uno et al. 2014; Trinquart et al. 2016).

Although it is straightforward to estimate the RMST in the absence of covariates by using the Kaplan-Meier curve up to time τ, estimating RMST when covariates are present is more complicated. It is possible to estimate RMST from a classical survival model, such as the Cox proportional hazards model (Zucker 1998), but the process is complex and difficult to extend. You can avoid this difficulty by using generalized linear modeling techniques to directly model the RMST. This has the double advantage of making inferences on the results straightforward and providing all the machinery of generalized linear model comparisons for studying RMST effects. This is the technique that PROC RMSTREG employs.

The RMSTREG procedure supports classification and interaction effects, along with certain generalized linear models links (Nelder and Wedderburn 1972), such as linear and log-linear functions. Models are fitted using methods based on estimating equations, such as pseudovalue regression (Andersen, Hansen, and Klein 2004) or inverse probability censoring weighting (IPCW) (Tian, Zhao, and Wei 2014). Compared to Cox regression and other classical methods, these methods enable you to model the RMST directly, facilitating model-based inference and prediction.

The RMSTREG procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Comparison with Other Procedures

The RMSTREG procedure compares most closely to the PHREG and LIFEREG procedures in SAS/STAT software. The three procedures fit different models to time-to-event data. The PHREG procedure fits the Cox proportional hazards model and various extensions. The LIFEREG procedure fits accelerated failure time models. The RMSTREG procedure fits generalized linear models to the RMST. Table 106.1 summarizes the key features of these procedures.

Table 106.1 Survival Modeling Procedures

Procedure	Focus	Model Type	Estimation Method
PROC LIFEREG	Time-to-event	Accelerated failure time models	Likelihood
PROC PHREG	Hazard function	Proportional hazards models	Partial likelihood
PROC RMSTREG	Restricted mean survival time	Generalized linear models	Estimating equations

Getting Started: RMSTREG Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/rmstgs.sas.)

This section presents an example that illustrates some of the basic features of the RMSTREG procedure by analyzing liver disease data from Lin, Wei, and Ying (1993).

The data represent 418 patients who have primary biliary cirrhosis (PBC), among whom 161 had died as of the date of data listing. A subset of the variables is saved in the SAS data set Liver. The data set contains the following variables:

- Time, follow-up time, in years
- Status, event indicator with the value 1 for death time and 0 for censored time
- Age, age in years, from birth to study registration
- Albumin, serum albumin level, in g/dl
- Bilirubin, serum bilirubin level, in mg/dl
- Edema, edema presence
- Protime, prothrombin time, in seconds

The following statements create the data set Liver:

```
data Liver;
   input Time Status Age Albumin Bilirubin Edema Protime @@;
   label Time="Follow-Up Time in Years";
   Time= Time / 365.25;
   datalines;
  400 1 58.7652 2.60 14.5 1.0 12.2 4500 0 56.4463 4.14  1.1 0.0 10.6
 1012 1 70.0726 3.48  1.4 0.5 12.0 1925 1 54.7406 2.54  1.8 0.5 10.3
 1504 0 38.1054 3.53  3.4 0.0 10.9 2503 1 66.2587 3.98  0.8 0.0 11.0
 1832 0 55.5346 4.09  1.0 0.0  9.7 2466 1 53.0568 4.00  0.3 0.0 11.0
 2400 1 42.5079 3.08  3.2 0.0 11.0   51 1 70.5599 2.74 12.6 1.0 11.5
 3762 1 53.7139 4.16  1.4 0.0 12.0  304 1 59.1376 3.52  3.6 0.0 13.6
 3577 0 45.6893 3.85  0.7 0.0 10.6 1217 1 56.2218 2.27  0.8 1.0 11.0
 3584 1 64.6461 3.87  0.8 0.0 11.0 3672 0 40.4435 3.66  0.7 0.0 10.8
```

```
   ... more lines ...

  989 0 35.0000 3.23  0.7 0.0 10.8  681 1 67.0000 2.96  1.2 0.0 10.9
 1103 0 39.0000 3.83  0.9 0.0 11.2 1055 0 57.0000 3.42  1.6 0.0  9.9
  691 0 58.0000 3.75  0.8 0.0 10.4  976 0 53.0000 3.29  0.7 0.0 10.6
;
```

The following statements fit a linear model for the RMST with the covariates Bilirubin, Age, and Edema:

```
proc rmstreg data=liver tau=10;
   class Edema;
   model Time*Status(0) = Age Bilirubin Edema / link=linear method=pv;
run;
```

The TAU= option in the PROC RMSTREG statement specifies the time limit that defines the RMST for this analysis. If you omit this option, the largest event time from the input data is used. Because the variable Edema is specified as a CLASS variable, it contributes one dummy variable to the regression model for each of its values.

In the MODEL statement, the response consists of the observed variable Time and an indicator variable Status, which specifies whether or not the Time value is censored. The values of Time are considered to be censored if the value of Status is 0; otherwise, they are considered to be event times.

An intercept term is included by default. Thus, the model matrix $\mathbf{X}$ consists of a column of 1s that represent the intercept term, three columns of 0s and 1s that correspond to the levels of the Edema variable, and two additional columns for the values of the variables Age and Bilirubin.

That is, the model matrix is

$$\mathbf{X} = \left[\begin{array}{c|ccc|cc} 1 & 1 & 0 & 0 & \text{Age} & \text{Bilirubin} \\ 1 & 0 & 1 & 0 & \text{Age} & \text{Bilirubin} \\ 1 & 0 & 0 & 1 & \text{Age} & \text{Bilirubin} \end{array}\right]$$

The LINK=LINEAR option fits a linear model. That is, the RMST at $\tau = 10$ for a specific subject (denoted by μ_i) is related to the linear predictor by

$$\log(\mu_i) = \mathbf{x}_i'\boldsymbol{\beta}$$

The METHOD=PV option specifies pseudovalue regression to fit the model.

Figure 106.1 shows the "Model Information" table, which provides information about the specified linear model of the RMST and the input data set.

Figure 106.1 Model Information

The RMSTREG Procedure

Model Information	
Data Set	WORK.LIVER
Time Variable	Time
Censoring Variable	Status
Censoring Value(s)	0
Link Function	Linear
Estimation Method	Pseudo Value
Tau Value	10

Figure 106.2 displays a summary of the number of event and censored observations in the data set.

Figure 106.2 Event and Censoring Summary

Summary of the Number of Event and Censored Values		
Total	**Event**	**Censored**
418	161	257

Figure 106.3 shows how the Edema variable is coded in the model matrix.

Figure 106.3 CLASS Variable Level Information

Class Level Information				
Class	**Value**	**Design Variables**		
Edema	**0**	1	0	0
	0.5	0	1	0
	1	0	0	1

For each parameter in the model, PROC RMSTREG displays a table (Figure 106.4) that contains columns of the parameter name, the degrees of freedom associated with the parameter, the estimated parameter value, the standard error of the parameter estimate, the confidence intervals, and the Wald chi-square statistic and associated p-value for testing the significance of the parameter to the model. If a column of the model matrix that corresponds to a parameter is found to be linearly dependent, or *aliased*, with columns that correspond to parameters preceding it in the model, PROC RMSTREG assigns it zero degrees of freedom and displays a value of 0 for both the parameter estimate and its standard error.

Figure 106.4 Analysis of Parameter Estimates

Analysis of Parameter Estimates								
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	9.0588	1.1201	6.8635	11.2540	65.41	<.0001
Age		1	-0.0686	0.0142	-0.0964	-0.0408	23.41	<.0001
Bilirubin		1	-0.3614	0.0383	-0.4366	-0.2862	88.82	<.0001
Edema	0	1	3.0259	0.7353	1.5849	4.4670	16.94	<.0001
Edema	0.5	1	1.7778	0.8658	0.0807	3.4748	4.22	0.0401
Edema	1	0	0.0000					

You generally assess the importance of the main effects in the model. By default, PROC RMSTREG performs a Type 3 analysis, the results of which are shown in Figure 106.5.

Figure 106.5 Type 3 Analysis of Effects

Type 3 Analysis of Effects			
Effect	DF	Chi-Square	Pr > ChiSq
Age	1	23.4052	<.0001
Bilirubin	1	88.8223	<.0001
Edema	2	19.4651	<.0001

The results indicate that all three variables are strong predictors of the RMST at $\tau = 10$. The Type 3 chi-square value for the Age variable, for example, is computed by comparing the model with Intercept, Age, Bilirubin, and Edema included and the model with Age excluded. The hypothesis that is tested in this case is that Age adds no predictability to the model, over and above Edema and Bilirubin. This hypothesis is strongly rejected.

If the RMST at $\tau = 5$ is of interest, then the following statements fit this model by using pseudovalue regression:

```
proc rmstreg data=liver tau=5;
   class Edema;
   model Time*Status(0) = Age Bilirubin Edema / link=linear method=pv;
run;
```

Figure 106.6 displays the parameter estimates and the Type 3 tests from the fitted model. The new results appear to be similar to the previous ones. Using values 5 and 10 for τ, respectively, makes little difference.

Figure 106.6 Parameter Estimates and Type 3 Tests for RMST at $\tau = 5$

The RMSTREG Procedure

Type 3 Analysis of Effects			
Effect	DF	Chi-Square	Pr > ChiSq
Age	1	18.9013	<.0001
Bilirubin	1	62.3617	<.0001
Edema	2	30.8671	<.0001

Figure 106.6 *continued*

Analysis of Parameter Estimates								
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	4.0394	0.4848	3.0892	4.9896	69.42	<.0001
Age		1	-0.0220	0.0051	-0.0319	-0.0121	18.90	<.0001
Bilirubin		1	-0.1378	0.0174	-0.1720	-0.1036	62.36	<.0001
Edema	0	1	1.8901	0.3621	1.1805	2.5997	27.25	<.0001
Edema	0.5	1	1.2573	0.4205	0.4331	2.0815	8.94	0.0028
Edema	1	0	0.0000					

Syntax: RMSTREG Procedure

The following statements are available in the RMSTREG procedure. Items within < > are optional.

```
PROC RMSTREG <options> ;
    BY variables ;
    CLASS variable <(options)> ... <variable <(options)>> </options> ;
    ESTIMATE <'label'> estimate-specification </options> ;
    FREQ | FREQUENCY variable ;
    LSMEANS <model-effects> </options> ;
    LSMESTIMATE model-effect <'label'> values <divisor=n> <, ... <'label'> values <divisor=n>>
                </options> ;
    MODEL response <* censor(list)> = <effects > </options> ;
    SLICE model-effect </options> ;
    STORE <OUT=>item-store-name </ LABEL='label'> ;
    TEST <model-effects> </options> ;
```

The PROC RMSTREG and MODEL statements are required. The CLASS statement, if present, must precede the MODEL statement. The following sections describe the PROC RMSTREG statement and then describe the other statements in alphabetical order.

PROC RMSTREG Statement

```
PROC RMSTREG <options> ;
```

The PROC RMSTREG statement invokes the RMSTREG procedure. Table 106.2 summarizes the *options* available in the PROC RMSTREG statement.

Table 106.2 PROC RMSTREG Statement Options

Option	Description
ALPHA=	Specifies the alpha level
DATA=	Specifies the input data set

Table 106.2 *continued*

Option	Description
ITHISTORY	Prints the iteration history of the model parameters
NAMELEN=	Specifies the length of effect names
NOPRINT	Suppresses all displayed output
OUTPV=	Outputs the pseudovalues used in pseudovalue regression to a SAS data set
SINGULAR=	Specifies the tolerance for testing the singularity
TAU=	Specifies the upper time limit that defines the RMST

You can specify the following *options*.

ALPHA=*number*

specifies the α level for $100(1-\alpha)\%$ confidence limits. The *number* must be between 0 and 1; the default value is 0.05, which results in 95% intervals. This value is used as the default level for confidence limits that are computed by the MODEL statement. You can override this default by specifying the ALPHA= option in the MODEL statement.

DATA=*SAS-data-set*

specifies the SAS data set that contains the data to be analyzed. If you omit this option, PROC RMSTREG uses the most recently created SAS data set.

ITHISTORY

displays the iteration history of computing estimates on the basis of the estimating equations and the final evaluation of the gradient.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters long, where *n* is a value between 20 and 200 characters. By default, NAMELEN=20.

NOPRINT

suppresses all displayed output. This option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OUTPV=*SAS-data-set*

names the output data set that contains the pseudovalues that are used in the pseudovalue regression analysis. To use this option, you must also specify METHOD=PV in the MODEL statement.

SINGULAR=*number*

EPSILON=*number*

specifies the tolerance for testing the singularity of the $\mathbf{X}'\mathbf{X}$ matrix that is formed from the design matrix $\mathbf{X}$ and for testing the singularity of the Hessian matrix upon convergence of the optimization algorithm. Appropriately, the test requires that a pivot be at least *number* times the original diagonal value. By default, *number* is 10^7 times the machine epsilon. On most machines, the default *number* is approximately 10^{-9}.

TAU=*value*
specifies the upper time limit of the RMST. The default *value* is the largest event time, where *value* must be positive.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC RMSTREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the RMSTREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(***options***)** > . . . < *variable* < **(***options***)** > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis. Response variables do not need to be specified in the CLASS statement.

The CLASS statement must precede the MODEL statement. Most options can be specified either as individual variable *options* or as *global-options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify *global-options* for the CLASS statement by placing them after a slash (/). *Global-options* are applied to all the variables that are specified in the CLASS statement. If you specify more than one CLASS statement, the *global-options* that are specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the *global-options*. You can specify the following values for either an *option* or a *global-option*:

CPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding design variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING

DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC RMSTREG orders the categories according to the ORDER= option and then reverses that order.

LPREFIX=*n*

specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding design variables. The default is $256 - \min(256, \max(2, f))$, where f is the formatted length of the CLASS variable.

MISSING

treats missing values (., ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values of the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so this option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC RMSTREG interprets values of the ORDER= option:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. You can specify any of the *keywords* shown in the following table. The default is PARAM=GLM. Design matrix columns are created from CLASS variables according to the corresponding coding schemes.

Value of PARAM=	Coding
EFFECT	Effect coding
GLM	Less-than-full-rank reference cell coding (this *keyword* can be used only in a global option)
ORDINAL **THERMOMETER**	Cumulative parameterization for an ordinal CLASS variable
POLYNOMIAL **POLY**	Polynomial coding
REFERENCE **REF**	Reference cell coding
ORTHEFFECT	Orthogonalizes PARAM=EFFECT coding
ORTHORDINAL **ORTHOTHERM**	Orthogonalizes PARAM=ORDINAL coding
ORTHPOLY	Orthogonalizes PARAM=POLYNOMIAL coding
ORTHREF	Orthogonalizes PARAM=REFERENCE coding

All parameterizations are full rank, except for the GLM parameterization. The REF= option in the CLASS statement determines the reference level for EFFECT and REFERENCE coding and for their orthogonal parameterizations. It also indirectly determines the reference level for a singular GLM parameterization through the order of levels.

If a PARAM= option is specified as a variable option for some variables, then any variables for which PARAM= is not specified use either the EFFECT parameterization if the global PARAM= option is not specified, or the full-rank parameterization indicated in the global PARAM= option if specified. If the global PARAM=GLM option is specified and PARAM= is also specified for some variables, GLM parameterization is used for all variables.

If PARAM=ORTHPOLY or PARAM=POLY and the classification variable is numeric, then the ORDER= option in the CLASS statement is ignored, and the internal unformatted values are used. For more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

REF=*'level'* | *keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*:

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n*>

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

Class Variable Default Parameterization

If the PARAM= option is not specified together with any individual CLASS variable, then by default, PARAM=GLM. Otherwise, the default is PARAM=EFFECT.

Class Variable Naming Convention

Parameter names for a CLASS predictor variable are constructed by concatenating the CLASS variable name with the CLASS levels. However, for the POLYNOMIAL and orthogonal parameterizations, parameter names are formed by concatenating the CLASS variable name and keywords that reflect the parameterization. For examples and more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

Class Variable Parameterization with Unbalanced Designs

PROC RMSTREG initially parameterizes the CLASS variables by looking at the levels of the variables across the complete data set. If you have an *unbalanced* replication of levels across variables or BY groups, then the design matrix and the parameter interpretation might be different from what you expect. For example, suppose you have a model that has one CLASS variable A with three levels (1, 2, and 3) and another CLASS variable B with two levels (1 and 2). If the third level of A occurs only with the first level of B, if you use the EFFECT parameterization, and if your model contains the effect A(B) and an intercept, then the design for A within the second level of B is not a differential effect. In particular, the design looks like the following:

		Design Matrix			
		A(B=1)		A(B=2)	
B	A	A1	A2	A1	A2
1	1	1	0	0	0
1	2	0	1	0	0
1	3	–1	–1	0	0
2	1	0	0	1	0
2	2	0	0	0	1

PROC RMSTREG detects linear dependency among the last two design variables and sets the parameter for A2(B=2) to zero, resulting in an interpretation of these parameters as if they were reference- or dummy-coded. The REFERENCE or GLM parameterization might be more appropriate for such problems.

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 106.3 summarizes the *options* available in the ESTIMATE statement.

Table 106.3 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the $\mathbf{L}$ matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement"

on page 475 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

FREQUENCY *variable* **;**

The *variable* in the FREQ statement identifies a variable in the input data set that contains the frequency of occurrence of each observation. PROC RMSTREG treats each observation as if it appeared n times, where n is the value of the FREQ variable for the observation. If the frequency value is not an integer, it is truncated to an integer. If it is less than 1 or missing, the observation is not used. The frequencies must be the same for all observations within each subject.

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 106.4 summarizes the *options* available in the LSMEANS statement.

Table 106.4 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion

Table 106.4 *continued*

Option	Description
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 106.5 summarizes the *options* available in the LSMESTIMATE statement.

Table 106.5 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

MODEL *response* < ∗ *censor* (*list*) > **=** < *effects* > < / *options* > **;**

The MODEL statement specifies the response variables and the effects (explanatory variables). If you omit the explanatory variables, PROC RMSTREG fits an intercept-only model. An intercept term is included in the model by default. You can omit the intercept by specifying the NOINT option.

The name of the failure time variable precedes the equal sign. This name can optionally be followed by an asterisk, the name of the censoring variable, and a list of censoring values (separated by blanks or commas if there is more than one) enclosed in parentheses. If the censoring variable takes one of these values, the corresponding failure time is considered to be censored. Following the equal sign are the explanatory effects (sometimes called independent variables or covariates) for the model.

The censoring variable must be numeric, and the failure time variables must contain nonnegative values. Any observation that has a negative failure time is excluded from the analysis, as is any observation that has a missing value for any of the variables listed in the MODEL statement. Failure time variables in a SAS date format are not recommended, because the dates might be translated into negative numbers and consequently the corresponding observation would be discarded.

The *effects* in the MODEL statement consist of an explanatory variable or combination of variables. Explanatory variables can be continuous or classification variables. Classification variables can be character or numeric. Explanatory variables that represent nominal (classification) data must be declared in a CLASS statement. Interactions between variables can also be included as effects. Columns of the design matrix are automatically generated for classification variables and interactions. The syntax for specifying effects is the same as in the GLM procedure. For more information, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

Table 106.6 summarizes the *options* available in the MODEL statement.

Table 106.6 MODEL Statement Options

Option	Description
ALPHA=	Sets the confidence coefficient
CONVERGE=	Sets the convergence criterion
CONVH=	Sets the relative Hessian convergence criterion
CORRB	Displays the parameter estimate correlation matrix
COVB	Displays the parameter estimate covariance matrix
LINK=	Specifies the link function
MAXITER=	Sets the maximum allowable number of iterations for all iterative computation processes
METHOD=	Specifies the estimation method for model fitting
NOFIT	Suppresses model fitting
NOINT	Excludes the intercept term from the model

You can specify the following *options* after a slash (/).

ALPHA=*number*

sets the confidence coefficient for parameter confidence intervals to 1–*number*. The value of *number* must be between 0 and 1. The default value of *number* is 0.05.

CONVERGE=*number*

sets the convergence criterion for parameter estimation. If the maximum absolute difference between regression parameter estimates is less than the value of *number* in two successive iterations, convergence is declared. If the absolute value of a regression parameter estimate is greater than 0.08, then the absolute difference normalized by the regression parameter value is used instead of the absolute difference. The default value of *number* is 0.0001.

CONVH=*number*

sets the relative Hessian convergence criterion. The value of *number* must be between 0 and 1. After convergence is determined by using the change in parameter criterion that is specified by the CONVERGE= option, the quantity $tc = \mathbf{g}'\mathbf{H}^{-1}\mathbf{g}$ is computed and compared to *number*, where $\mathbf{g}$ is the gradient vector and $\mathbf{H}$ is the Hessian matrix for the model parameters. If tc is greater than *number*, a warning is issued that the relative Hessian convergence criterion has been exceeded. This criterion detects the occasional case where the change in parameter convergence criterion is satisfied but a global solution has not been attained. By default, CONVH=1E–4.

CORRB

displays the parameter estimate correlation matrix.

COVB

displays the parameter estimate covariance matrix.

LINK=*keyword*

specifies the link function in the model. You can specify the *keywords* shown in Table 106.7.

Table 106.7 Built-In Link Functions of the RMSTREG Procedure

LINK=	Link Function	$g(\mu) =$
IDENTITY | **ID** | **LINEAR**	Identity	μ
LOG	Log	$\log(\mu)$

By default, LINK=LOG.

MAXITER=*number*

MAXIT=*number*

sets the maximum allowable number of iterations for all iterative computation processes in PROC RMSTREG. By default, MAXITER=50.

METHOD=*method***<(STRATA=***variable***)>**

specifies the estimation method to fit the specified model. You can specify the *methods* shown in Table 106.8.

Table 106.8 Estimation Methods of the RMSTREG Procedure

METHOD=	Estimation Method
PV	Pseudovalue regression (Andersen, Hansen, and Klein 2004)
IPCW	Inverse probability censoring weighting estimation (Tian, Zhao, and Wei 2014)

By default, METHOD=PV. For METHOD=IPCW, you can compute stratified weights and use them in the iterative estimation process by specifying the STRATA=*variable* option. For more information about pseudovalue regression, see the section "Pseudovalue Regression" on page 8935. For more information about the inverse probability censoring weighting techniques, see the section "Inverse Probability Censoring Weighting Estimation" on page 8936.

NOFIT

suppresses model fitting. If this option is specified along with other MODEL statement options, NOFIT takes precedence, and all other options are ignored.

NOINT

excludes the intercept term from the model. An intercept is included unless you specify this option.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

TEST Statement

TEST < *model-effects* > < / *options* > **;**

The TEST statement enables you to perform Wald tests for model effects that test Type I, Type II, or Type III hypotheses. For more information about constructing Type I, II, and III estimable functions, see Chapter 16, "The Four Types of Estimable Functions."

Table 106.9 summarizes the *options* that you can specify in the TEST statement.

Table 106.9 TEST Statement Options

Option	Description
E	Requests Type I, Type II, and Type III coefficients
E1	Requests Type I coefficients
E2	Requests Type II coefficients
E3	Requests Type III coefficients
HTYPE=	Indicates the type of hypothesis test to perform
INTERCEPT	Adds a row that corresponds to the overall intercept

For more information about the syntax of the TEST statement, see the section "TEST Statement" on page 556 in Chapter 20, "Shared Concepts and Topics."

Details: RMSTREG Procedure

Concepts and Basic Estimators

Let T be a nonnegative random variable that represents the failure time of an individual from a homogeneous population. The survival function (also known as the survivor function) of T is defined as

$$S(t) = \Pr(T \geq t)$$

Assume that τ is a prespecified time point of interest. Let R be the minimal of T and τ,

$$R = T \wedge \tau = \min(T, \tau)$$

The restricted mean survival time (RMST) is defined as the expected value of R:

$$\mathrm{RMST}(\tau) = E(R) = E[\min(T, \tau)]$$

It can be evaluated by the area under the survival function over $[0, \tau]$ as

$$\mathrm{RMST}(\tau) = \int_0^\tau S(u)du$$

Let $t_1 < t_2 < \cdots < t_D$ represent the distinct event times. For each $i = 1, \ldots, D$, let Y_i be the number of surviving units (the size of the risk set) just prior to t_i, and let d_i be the number of units that fail at t_i.

The Kaplan-Meier (product-limit) estimate of the survival function at t_i is the cumulative product

$$\hat{S}(t_i) = \prod_{j=1}^{i} \left(1 - \frac{d_j}{Y_j}\right)$$

The estimated mean survival time is

$$\hat{\mu} = \sum_{i=1}^{D} \hat{S}(t_{i-1})(t_i - t_{i-1})$$

where t_0 is defined to be zero.

The $\mathrm{RMST}(\tau)$ is estimated by

$$\widehat{\mathrm{RMST}}(\tau) = \sum_{i=1}^{N^*} \hat{S}(t_{i-1})(t_i - t_{i-1}) + \hat{S}(t_{N^*})(\tau - t_{N^*})$$

where N^* is the number of t_i values that are less than τ.

When the largest observed time is censored, the mean survival time is underestimated because the Kaplan-Meier estimate does not drop to zero (Klein and Moeschberger 2003). The RMST, on the other hand, can still be estimated unbiasedly as long as $\tau \leq t_D$.

Supported Models

Let D_i be the response variable for the ith observation. The quantity $\mathbf{x}_i$ is a column vector of covariates, or explanatory variables, for observation i that is known from the experimental setting and is considered to be fixed, or nonrandom.

The expected value of D_i, denoted by μ_i, is

$$\mu_i = \mathbf{x}_i' \boldsymbol{\beta}$$

where $\boldsymbol{\beta}$ is an unknown parameter vector and another column can be added to $\mathbf{x}_i$ for an intercept effect.

Under the specification of generalized linear models (Nelder and Wedderburn 1972), μ_i is related to a linear predictor through a monotone and differentiable link function g:

$$g(\mu_i) = \mathbf{x}_i' \boldsymbol{\beta}$$

Assume that τ is a prespecified time point of interest. Let T_i be the time-to-event variable for the ith subject. The subject-specific RMST at τ is defined by $\mathrm{RMST}_i(\tau) = E[\min(T_i, \tau)]$ and can be conveniently modeled via a generalized linear model as

$$g[\mathrm{RMST}_i(\tau)] = \mathbf{x}_i' \boldsymbol{\beta}$$

Under the natural logarithm link $g(\cdot) = \log(\cdot)$, the model is

$$\log[\mathrm{RMST}_i(\tau)] = \mathbf{x}_i'\boldsymbol{\beta}$$

Under the identity or linear link, the model is

$$\mathrm{RMST}_i(\tau) = \mathbf{x}_i'\boldsymbol{\beta}$$

Specification of Effects

Each term in a model is called an effect. You specify effects in the MODEL statement by using a special notation that uses variable names and operators. There are two types of variables: *classification* (CLASS) variables and *continuous* variables. There are two primary types of operators: *crossing* and *nesting*. A third type, the *bar* operator, is used to simplify effect specification.

Variables that identify classification levels are called CLASS variables in SAS and are specified in a CLASS statement. These might also be called *categorical, qualitative, discrete,* or *nominal* variables. CLASS variables can be either character or numeric. The values of CLASS variables are called *levels*. For example, the CLASS variable Sex could have the levels "male" and "female."

In a model, an explanatory variable that is not declared in a CLASS statement is assumed to be continuous. Continuous variables must be numeric. For example, the heights and weights of subjects in an experiment are continuous variables.

The following list shows types of effects that are often useful in practice, where A, B, and C are classification variables and X1 and X2 are continuous variables:

- Regressor effects are specified by writing continuous variables by themselves: X1, X2.
- Polynomial effects are specified by using asterisks to join two or more continuous variables: X1*X2.
- Main effects are specified by writing classification variables by themselves: A, B, C.
- Crossed effects (interactions) are specified by using asterisks to join two or more classification variables: A*B, B*C, A*B*C.
- Nested effects are specified by following a main effect or crossed effect with a classification variable or list of classification variables that are enclosed in parentheses: B(A), C(B A), A*B(C). In this example, B(A) is "B nested within A."
- Combinations of continuous and classification variables can be specified in the same way by using the crossing and nesting operators.

The bar operator uses a vertical bar (|) to join two effects. The bar operator is shorthand notation for including the left-hand side, the right-hand side, and the cross between them as effects in the model. For example, the expression A | B is equivalent to A B A*B. The effects that are joined by the bar operator can be classification variables, continuous variables, or combinations of effects that are defined by using operators. Multiple bars are permitted. For example, A | B | C means A B C A*B A*C B*C A*B*C.

You can specify the maximum number of variables in any effect that results from bar evaluation by specifying the maximum number, preceded by an @ sign. For example, A | B | C@2 results in effects that involve no more than two variables: A B C A*B A*C B*C.

Parameterization Used in PROC RMSTREG

Design Matrix

The linear predictor part of a generalized linear model is

$$\boldsymbol{\eta} = \mathbf{X}'\boldsymbol{\beta}$$

where $\boldsymbol{\beta}$ is an unknown parameter vector and $\mathbf{X}$ is a known design matrix. By default, all models automatically contain an intercept term; that is, the first column of $\mathbf{X}$ contains all 1s. Additional columns of $\mathbf{X}$ are generated for classification variables, regression variables, and any interaction terms included in the model. It is important to understand the ordering of classification variable parameters when you use the ESTIMATE statement. The ordering of these parameters is displayed in the "CLASS Level Information" table and in tables that display the parameter estimates of the fitted model.

When you specify an overparameterized model by using the PARAM=GLM option in the CLASS statement, some columns of $\mathbf{X}$ can be linearly dependent on other columns. For example, when you specify a model that consists of an intercept term and a classification variable, the column that corresponds to any one of the levels of the classification variable is linearly dependent on the other columns of $\mathbf{X}$. The columns of $\mathbf{X}'\mathbf{X}$ are checked in the order in which the model is specified for dependence on preceding columns. If a dependency is found, the parameter that corresponds to the dependent column is set to 0 along with its standard error to indicate that it is not estimated. The order in which the levels of a classification variable are checked for dependencies can be set by the ORDER= option in the CLASS statement. For full-rank parameterizations, the columns of the $\mathbf{X}$ matrix are designed to be linearly independent.

You can exclude the intercept term from the model by specifying the NOINT option in the MODEL statement.

Missing Level Combinations

All levels of interaction terms that involves classification variables might not be represented in the data. In that case, PROC RMSTREG does not include parameters in the model for the missing levels.

Type 3 Tests and Joint Tests

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, performing the Type 3 test of a main effect corresponds to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions." Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are zero. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), performing the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses reference parameterization (as specified by the PARAM=REF option in the CLASS statement), performing the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects.

For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).

If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimable function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.

Confidence Intervals and Predicted Values

Wald Confidence Intervals

You can use PROC RMSTREG to produce Wald confidence intervals for the parameters. The $(1-\alpha)100\%$ Wald confidence interval for a parameter β is defined as

$$\hat{\beta} \pm z_{1-\alpha/2}\hat{\sigma}$$

where z_p is the $100p$th percentile of the standard normal distribution, $\hat{\beta}$ is the parameter estimate, and $\hat{\sigma}$ is the estimate of its standard error.

Predicted Values

A predicted value, or fitted value, of the mean μ_i that corresponds to the vector of covariates $\mathbf{x}_i$ is given by

$$\hat{\mu_i} = g^{-1}(\mathbf{x}_i'\hat{\boldsymbol{\beta}})$$

where g is the link function, regardless of whether $\mathbf{x}_i$ corresponds to an observation. That is, even if the response variable is missing, the predicted value is still computed for valid $\mathbf{x}_i$. In the case where $\mathbf{x}_i$ does not correspond to a valid observation, $\mathbf{x}_i$ is not checked for estimability. You should check the estimability of $\mathbf{x}_i$ in this case in order to ensure the uniqueness of the predicted value of the mean.

Confidence Intervals on Predicted Values

Approximate confidence intervals for predicted values of the mean can be computed as follows. The variance of the linear predictor $\eta_i = \mathbf{x}_i'\hat{\boldsymbol{\beta}}$ is estimated by

$$\sigma_x^2 = \mathbf{x}_i'\boldsymbol{\Sigma}\mathbf{x}_i$$

where $\boldsymbol{\Sigma}$ is the estimated covariance of $\hat{\boldsymbol{\beta}}$.

Approximate $100(1-\alpha)\%$ confidence intervals are computed as

$$g^{-1}\left(\mathbf{x}_i'\hat{\beta} \pm z_{1-\alpha/2}\sigma_x\right)$$

where z_p is the $100p$th percentile of the standard normal distribution and g is the link function. If either endpoint in the argument is outside the valid range of arguments for the inverse link function, the corresponding confidence interval endpoint is set to missing.

Pseudovalue Regression

Pseudovalue regression is a generic method of fitting generalized linear models to time-to-event data (Andersen, Klein, and Rosthøj 2003). This section describes how the method works and how you can apply it to analyze models of the RMST.

Let $\mathbf{D}_1, \ldots, \mathbf{D}_n$ be independent and identically distributed quantities that might be random variables or vectors of variables. Let $\boldsymbol{\theta} = E[f(\mathbf{D}_i)]$ for some function $f(\cdot)$. Suppose $\hat{\boldsymbol{\theta}}$ is an unbiased estimator of $\boldsymbol{\theta}$.

Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be independent and identically distributed samples of covariates, and define the conditional expectation of $f(\mathbf{D}_i)$ given by $\mathbf{x}_i$ as

$$\boldsymbol{\theta}_i = E[f(\mathbf{D}_i)|\mathbf{x}_i]$$

The ith pseudo-observation of $\boldsymbol{\theta}$ is computed as

$$\hat{\boldsymbol{\theta}}_i = n\hat{\boldsymbol{\theta}} - (n-1)\hat{\boldsymbol{\theta}}^{-i}$$

where $\hat{\boldsymbol{\theta}}^{-i}$ is the jackknife leave-one-out estimator for $\boldsymbol{\theta}$ based on $\{\mathbf{D}_j : j \neq i\}$.

The generalized linear model (Nelder and Wedderburn 1972) for $\boldsymbol{\theta}$ assumes

$$g(\boldsymbol{\theta}_i) = \mathbf{x}_i'\boldsymbol{\beta}$$

where $g(\cdot)$ is a suitable link function. Note that an additional column can be added to X_i for an intercept effect.

Using pseudo-observations, you can estimate the regression parameters $\boldsymbol{\beta}$ by solving the following estimating equations

$$\mathbf{U}(\boldsymbol{\beta}) = \sum_{i=1}^{n} \mathbf{U}_i(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left(\frac{\partial \boldsymbol{\theta}_i}{\partial \boldsymbol{\beta}}\right)' \mathbf{V}_i^{-1} \left(\hat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i\right) = \mathbf{0}$$

where $\mathbf{V}_i$ is a working covariance matrix.

Let $\hat{\boldsymbol{\beta}}$ be a solution of the estimating equations. You can use a sandwich estimator to estimate the variance of $\hat{\boldsymbol{\beta}}$. It takes the form

$$\boldsymbol{\Sigma}_e = \mathbf{I}_0^{-1}\mathbf{I}_1\mathbf{I}_0^{-1}$$

$\mathbf{I}_0^{-1}$ is the model-based estimator of $\mathrm{Cov}(\hat{\boldsymbol{\beta}})$ and is given by

$$\mathbf{I}_0 = \sum_{i=1}^{n} \frac{\partial \boldsymbol{\theta}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} \frac{\partial \boldsymbol{\theta}_i}{\partial \boldsymbol{\beta}}$$

$\mathbf{I}_1^{-1}$ is the empirical estimator of $\mathrm{Cov}(\hat{\boldsymbol{\beta}})$ and is computed as

$$\mathbf{I}_1 = \sum_{i=1}^{n} \mathbf{U}_i(\hat{\boldsymbol{\beta}})'\mathbf{U}_i(\hat{\boldsymbol{\beta}})$$

Andersen, Hansen, and Klein (2004) proposed using pseudovalue regression to analyze the RMST models. Assume τ is a prespecified time point of interest. Let T_i be the time-to-event variable for the ith subject. The RMST models can be fitted using pseudovalue regression by letting

$$\boldsymbol{\theta}_i = \mathrm{RMST}_i(\tau) = E(T_i \wedge \tau | \mathbf{x}_i)$$

$$\mathbf{V}_i = \begin{cases} \theta_i = \mathrm{RMST}_i(\tau) & g(u) = \log(u) \\ 1 & g(u) = u \end{cases}$$

Because the nonparametric estimator $\widehat{\mathrm{RMST}}(\tau)$ is unbiased, it can be used in place of $\hat{\boldsymbol{\theta}}$ in the estimation process.

Inverse Probability Censoring Weighting Estimation

Suppose you can observe the quantities $(T, C, \mathbf{x})$, where T is the event time, C is the censoring time, and $\mathbf{x}$ is a p-dimensional vector of covariates. For the ith subject, $i = 1, \ldots, n$, let $U_i = T_i \wedge C_i$, $\Delta_i = I(T_i \le C_i)$, and $\mathbf{x}_i$ be the observed time, event indicator, and covariate vector, respectively.

Assume that τ is a prespecified time point of interest and $P(T > \tau) > 0$. Let

$$\begin{aligned} R_i &= T_i \wedge \tau \\ \mathrm{RMST}_i(\tau) &= E(R_i | \mathbf{x}_i) \\ \tilde{\Delta}_i &= I(R_i \le C_i) \\ w_i &= \frac{\tilde{\Delta}_i}{\hat{G}(R_i)} \end{aligned}$$

where $\hat{G}(t)$ is the Kaplan-Meier estimate (alternatively, the Breslow estimate) of the survival function of the censoring variable, which is calculated using $\{(U_i, 1 - \Delta_i) : i = 1, 2, \ldots, n\}$.

Suppose that the following relationship holds for the RMST,

$$g[\mathrm{RMST}_i(\tau)] = \mathbf{x}_i'\boldsymbol{\beta}$$

where $g(\cdot)$ is a smooth and strictly increasing function. Note that another column can be added to X_i for an intercept effect.

Under suitable regularity conditions, the regression coefficients $\boldsymbol{\beta}$ are estimated by solving the following score function (Tian, Zhao, and Wei 2014):

$$\mathbf{U}(\boldsymbol{\beta}) = \sum_{i=1}^{n} w_i \left(R_i - g^{-1}(\mathbf{x}_i'\boldsymbol{\beta}) \right) \mathbf{x}_i = \mathbf{0}$$

Let

$$\hat{\boldsymbol{\Omega}} = \sum_{i=1}^{n} \mathbf{x}_i^{\otimes 2}\left(g^{-1}(\mathbf{x}_i'\hat{\boldsymbol{\beta}})\right)$$

The sandwich variance estimate of $\hat{\boldsymbol{\beta}}$ is

$$\widehat{\mathrm{Var}}(\hat{\boldsymbol{\beta}}) = \hat{\boldsymbol{\Omega}}^{-1}\hat{\boldsymbol{\Sigma}}\hat{\boldsymbol{\Omega}}^{-1}$$

where $\hat{\boldsymbol{\Sigma}}$ is the empirical variance-covariance matrix of $\mathbf{U}(\hat{\boldsymbol{\beta}})$ that is given by

$$\hat{\boldsymbol{\Sigma}} = \sum_{i=1}^{n}(\hat{\boldsymbol{\eta}}_i + \hat{\boldsymbol{\psi}}_i)^{\otimes 2}$$

where

$$\hat{\boldsymbol{\eta}}_i = w_i\left(R_i - g^{-1}(\mathbf{x}_i'\boldsymbol{\beta})\right)\mathbf{x}_i$$

$$\hat{\boldsymbol{\psi}}_i = \int_0^{\infty} \frac{\hat{\mathbf{q}}(u)}{\pi(u)} d\hat{M}_i^c(u)$$

$$\hat{\mathbf{q}}(u) = \sum_{i=1}^{n} w_i\left(R_i - g^{-1}(\mathbf{x}_i'\boldsymbol{\beta})\right)\mathbf{x}_i I(U_i \geq u)$$

$$\pi(u) = \sum_j I(U_j \geq u)$$

$$\hat{M}_i^c(t) = I(U_i \leq t, \Delta_i = 0) - \int_0^t I(U_i \geq u) d\hat{\Lambda}^c(u)$$

$$\hat{\Lambda}^c(t) = \int_0^t \frac{dN^c(u)}{\pi(u)}$$

$$N^c(u) = \sum_j I(U_j \leq u, \Delta_j = 0)$$

Estimation with Stratified Weights

Assuming that you have K strata, within each stratum the censoring distribution is homogeneous. For the ith subject, let $B_i \in (1, \ldots, K)$ be the stratum indicator. It is more appropriate to use stratum-specific weights in the estimation. For the kth stratum, you compute the Kaplan-Meier estimate $\hat{G}_k(t)$ for the censoring variable by using $\{(U_i, 1 - \Delta_i) : B_i = k, i = 1, 2, \ldots, n\}$.

For the ith subject, the weight is computed as

$$w_i = \frac{\tilde{\Delta}_i}{\hat{G}_{k=B_i}(R_i)}$$

The following quantities are also adjusted accordingly in the estimation:

$$\hat{\mathbf{q}}_k(u) = \sum_{i=1}^{n} w_i\left(R_i - g^{-1}(\mathbf{x}_i'\boldsymbol{\beta})\right)\mathbf{x}_i I(U_i \geq u, B_j = k)$$

$$\pi_k(u) = \sum_j I(U_j \geq u, B_j = k)$$

$$\hat{M}_i^c(t) = I(U_i \leq t, \Delta_i = 0) - \int_0^t I(U_i \geq u) d\,\hat{\Lambda}_{k=B_i}^c(u)$$

$$\hat{\Lambda}_k^c(t) = \int_0^t \frac{dN_k^c(u)}{\pi_k(u)}$$

$$N_k^c(u) = \sum_j I(U_j \leq u, \Delta_j = 0, B_j = k)$$

Fitting Algorithm

The following is an algorithm for solving generalized estimating equations (GEEs) (Liang and Zeger 1986). The algorithm is generic and can be conveniently adapted to solve the estimating equations from a pseudovalue regression or the inverse probability censoring weighting (IPCW) method.

Typically, the estimating equations of a GEE have the form

$$\mathbf{U}(\boldsymbol{\beta}) = \sum_{i=1}^{n} \mathbf{U}_i(\boldsymbol{\beta}) = \sum_{i=1}^{n} f_i \left(\frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}\right)' \mathbf{V}_i^{-1} \left(\mathbf{Y}_i - \boldsymbol{\mu}_i\right) = \mathbf{0}$$

where $\mathbf{Y}_i = (y_{i1}, \ldots, y_{in_i})'$ is the outcome vector for the ith subject, $\boldsymbol{\mu}_i = (\mu_{i1}, \ldots, \mu_{in_i})'$ is the model-based means of $\mathbf{Y}_i$, $\mathbf{V}_i$ is the covariance matrix of $\mathbf{Y}_i$, and f_i is the subject-specific weight.

Let $\mathbf{R}_i(\boldsymbol{\alpha})$ be an $n_i \times n_i$ "working" correlation matrix that is fully specified by the vector of parameters $\boldsymbol{\alpha}$. The covariance matrix of $\mathbf{Y}_i$ is modeled as

$$\mathbf{V}_i = \phi \mathbf{A}_i^{\frac{1}{2}} \mathbf{W}_i^{-\frac{1}{2}} \mathbf{R}(\boldsymbol{\alpha}) \mathbf{W}_i^{-\frac{1}{2}} \mathbf{A}_i^{\frac{1}{2}}$$

where $\mathbf{A}_i$ is an $n_i \times n_i$ diagonal matrix with $v(\mu_{ij})$ as the jth diagonal element and $\mathbf{W}_i$ is an $n_i \times n_i$ diagonal matrix with w_{ij} as the jth diagonal, where w_{ij} is a weight that is specified a priori. If $\mathbf{R}_i(\boldsymbol{\alpha})$ is the true correlation matrix of $\mathbf{Y}_i$, then $\mathbf{V}_i$ is the true covariance matrix of $\mathbf{Y}_i$.

The working correlation matrix is usually unknown and must be estimated. It is estimated in the iterative fitting process by using the current value of the parameter vector $\boldsymbol{\beta}$ to compute appropriate functions of the Pearson residual

$$e_{ij} = \frac{y_{ij} - \mu_{ij}}{\sqrt{v(\mu_{ij})/w_{ij}}}$$

If you specify the working correlation as $\mathbf{R_0} = \mathbf{I}$, which is the identity matrix, the GEE reduces to the independence estimating equation.

The dispersion parameter ϕ is estimated by

$$\hat{\phi} = \frac{1}{N-p} \sum_{i=1}^{K} f_i \sum_{j=1}^{n_i} e_{ij}^2$$

where $N = \sum_{i=1}^{K} f_i n_i$ is the total number of measurements and p is the number of regression parameters.

To solve the estimating equations, the algorithm proceeds as follows:

1. Computes an initial estimate of $\boldsymbol{\beta}$ by using an ordinary generalized linear model that assumes independence.

2. Computes the working correlations $\mathbf{R}$ on the basis of the standardized residuals, the current $\boldsymbol{\beta}$, and the assumed structure of $\mathbf{R}$.

3. Computes an estimate of the covariance:

$$\mathbf{V}_i = \phi \mathbf{A}_i^{\frac{1}{2}} \mathbf{W}_i^{-\frac{1}{2}} \hat{\mathbf{R}}(\boldsymbol{\alpha}) \mathbf{W}_i^{-\frac{1}{2}} \mathbf{A}_i^{\frac{1}{2}}$$

4. Updates $\boldsymbol{\beta}$:

$$\boldsymbol{\beta}_{r+1} = \boldsymbol{\beta}_r + \left[\sum_{i=1}^{K} f_i \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}} \right]^{-1} \left[\sum_{i=1}^{K} f_i \frac{\partial \boldsymbol{\mu}_i}{\partial \boldsymbol{\beta}}' \mathbf{V}_i^{-1} (\mathbf{Y}_i - \boldsymbol{\mu}_i) \right]$$

5. Repeats steps 2–4 until convergence.

Missing Values

Observations that contain a missing or negative value in the response are not used in the analysis. If a FREQ variable value is missing or 0, the observation is not used. If a STRATA variable is missing, the observation is not used. If any explanatory variable that you specify in the MODEL statement has a missing value, that observation is not used in the model fitting.

Displayed Output

PROC RMSTREG displays the following information as results of the model fitting.

Model Information

The “Model Information” table displays the two-level name of the input data set, the name and label of the failure time variable, the name and label of the censoring variable, the values that indicate censored times, the link function, the estimation method and inference method used, the name and label of the FREQ variable, and the name and label of the STRATA variable.

Number of Observations

The “Number of Observations” table displays the number of observations that were read and used in the analysis.

CLASS Level Information

If you specify a CLASS statement, PROC RMSTREG outputs the "Class Level Information" table. This table displays the design information for the classification variables when the parameterization is full rank.

Summary of the Number of Event and Censored Values

The "Summary of the Number of Event and Censored Values" table displays a breakdown of the number of events and censored values.

Analysis of Parameter Estimates

The "Analysis of Parameter Estimates" table displays the parameter name, the degrees of freedom for each parameter, the estimate of each parameter, the estimated standard error of the parameter estimator, confidence limits for each parameter, a chi-square statistic for testing whether the parameter is 0, and the associated p-value for the statistic.

Iteration History for Parameter Estimates

If you specify the ITHISTORY option in the PROC RMSTREG statement, the procedure outputs a table that contains the following for each iteration in the iterative process of model fitting: the iteration number, the maximum gradient, and the values of all parameters in the model.

Last Evaluation of the Gradient

If you specify the ITHISTORY option in the PROC RMSTREG statement, the procedure displays the last evaluation of the gradient vector.

Convergence Status

The "Convergence Status" table displays the convergence status of the iterative estimation routine.

Estimated Covariance Matrix

If you specify the COVB option in the MODEL statement, the procedure displays the estimated covariance matrix. This matrix is defined as the inverse of the information matrix at the final iteration and is based on the Hessian matrix that is used at the final iteration.

Estimated Correlation Matrix

If you specify the CORRB option in the MODEL statement, the procedure displays the estimated correlation matrix, which is based on the Hessian matrix that is used at the final iteration.

Type 3 Tests

The "Type 3 Tests" table is displayed if the model contains a CLASS variable or if you specify the TYPE3 option in the MODEL statement. The table displays, for each specified statistic, the Type 3 chi-square, the degrees of freedom, and the p-value for each effect in the model.

ODS Table Names

PROC RMSTREG assigns a name to each table that it creates. You can use these names to refer to the tables when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 106.10. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 106.10 ODS Tables Produced by PROC RMSTREG

ODS Table Name	Description	Statement	Option
ClassLevelInfo	Design information for CLASS variables	CLASS	
Coef	Coefficients for LS-means	LSMEANS	E
ConvergenceStatus	Convergence status	MODEL	
CorrB	Parameter estimate correlation matrix	MODEL	CORRB
CovB	Parameter estimate covariance matrix	MODEL	COVB
Diffs	Differences of LS-means	LSMEANS	DIFF
IterHistory	Iteration history	PROC RMSTREG	ITHISTORY
LastGradient	Last evaluation of the gradient	PROC RMSTREG	ITHISTORY
LSMeans	LS-means	LSMEANS	Default
LSMLines	Lines display for LS-means	LSMEANS	LINES
ModelInfo	Model and data information	MODEL	
NObs	Number of observations	MODEL	
ParameterEstimates	Parameter estimates	MODEL	
Type3Test	Type 3 tests	MODEL	

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, the RMSTREG procedure produces plots through the LSMEANS statement.

Examples: RMSTREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The following examples illustrate some of the capabilities of the RMSTREG procedure. These examples are not intended to represent definitive analyses of the data sets that are presented here.

Example 106.1: Comparison of PROC RMSTREG with Other Procedures

(View the complete code for this example (rmstex1.sas) in the example repository.)

This example compares the RMSTREG procedure with the PHREG and LIFEREG procedures.

Krall, Uthoff, and Harley (1975) analyzed data from a study on multiple myeloma in which researchers treated 65 patients by using alkylating agents. Of those patients, 48 died during the study and 17 survived. The following DATA step creates the data set Myeloma:

```
data Myeloma;
   input Time VStatus LogBUN HGB Platelet Age LogWBC Frac
         LogPBM Protein SCalc;
   label Time='Survival Time'
         VStatus='0=Alive 1=Dead';
   datalines;
 1.25  1  2.2175   9.4  1  67  3.6628  1  1.9542  12  10
 1.25  1  1.9395  12.0  1  38  3.9868  1  1.9542  20  18
 2.00  1  1.5185   9.8  1  81  3.8751  1  2.0000   2  15
 2.00  1  1.7482  11.3  0  75  3.8062  1  1.2553   0  12
 2.00  1  1.3010   5.1  0  57  3.7243  1  2.0000   3   9
 3.00  1  1.5441   6.7  1  46  4.4757  0  1.9345  12  10
 5.00  1  2.2355  10.1  1  50  4.9542  1  1.6628   4   9

   ... more lines ...

53.00  0  1.1139  12.0  1  66  3.6128  1  2.0000   1  11
57.00  0  1.2553  12.5  1  66  3.9685  0  1.9542   0  11
77.00  0  1.0792  14.0  1  60  3.6812  0  0.9542   0  12
;
```

The variable Time represents the survival time in months from diagnosis. The variable VStatus indicates whether the patient survived the study. If the value of VStatus is 0, then the patient survived and thus the corresponding value of Time is censored. The variables that are thought to be related to survival are LogBUN (log(BUN) at diagnosis), HGB (hemoglobin at diagnosis, in g/dl), Platelet (platelets at diagnosis: 0=abnormal, 1=normal), Age (age at diagnosis, in years), LogWBC (log(WBC) at diagnosis), Frac (fractures at diagnosis: 0=none, 1=present), LogPBM (log percentage of plasma cells in bone marrow), Protein (proteinuria at diagnosis, in mg/dl), and SCalc (serum calcium at diagnosis, in mg/dl). The goal of the study is to identify important prognostic factors from these nine explanatory variables.

Suppose that T_i represents the survival time of patient i and μ_i represents the RMST at the specified time τ

for the patient. A log-linear model for the RMST assumes that

$$\log(\mu_i) = \mathbf{x}_i{}'\boldsymbol{\beta}$$

The following statements fit the log-linear model for the RMST at $\tau = 50$ with covariates LogBUN and HGB:

```
proc rmstreg data=Myeloma tau=50;
   model Time*VStatus(0)=LogBUN HGB / method=ipcw link=log;
run;
```

Output 106.1.1 displays the parameter estimates for the fitted log-linear regression model.

Output 106.1.1 Parameter Estimates

The RMSTREG Procedure

Analysis of Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	4.2187	0.6091	3.0249	5.4126	47.97	<.0001
LogBUN	1	-1.0925	0.3053	-1.6908	-0.4942	12.81	0.0003
HGB	1	0.0438	0.0416	-0.0378	0.1254	1.11	0.2923

The following statements fit the linear model with covariates LogBUN and HGB:

```
proc rmstreg data=Myeloma tau=50;
   model Time*VStatus(0)=LogBUN HGB / method=ipcw link=linear;
   store rr;
run;
```

Output 106.1.2 displays the resulting parameter estimates for the fitted linear regression model.

Output 106.1.2 Parameter Estimates

The RMSTREG Procedure

Analysis of Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	45.2318	14.0717	17.6518	72.8118	10.33	0.0013
LogBUN	1	-23.1680	5.5614	-34.0681	-12.2679	17.35	<.0001
HGB	1	1.1493	1.0259	-0.8615	3.1601	1.25	0.2626

For comparison, a Cox proportional hazards (PH) model is fitted to the data, using the same set of covariates. The PH model assumes that

$$\lambda(t|\mathbf{x}_i) = \lambda_0(t)\exp(\mathbf{x}_i{}'\boldsymbol{\beta}_{\mathrm{PH}})$$

where $\lambda_0(t)$ is the baseline hazard function and $\boldsymbol{\beta}_{\mathrm{PH}}$ is a vector of unknown regression parameters.

The following PROC PHREG statements fit a Cox PH model with covariates LogBUN and HGB:

```
proc phreg data=Myeloma;
   model Time*VStatus(0)=LogBUN HGB;
   store ph;
run;
```

Output 106.1.3 displays the parameter estimates for the fitted Cox regression model.

Output 106.1.3 Parameter Estimates from Cox Regression

The PHREG Procedure

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
LogBUN	1	1.67440	0.61209	7.4833	0.0062	5.336
HGB	1	-0.11899	0.05751	4.2811	0.0385	0.888

If you compare the results from PROC RMSTREG with results from PROC PHREG, you see that the estimated effects have opposite signs. This is because Cox regression models the hazard function, and RMST regression models the restricted mean at a certain time via a log link. Thus, you interpret the estimated effects from PROC PHREG as ratios of the hazard functions; you interpret the estimated effects from PROC RMSTREG as log ratios of the RMST. An increase in the hazard rate would correspond roughly to a decrease in the restricted mean, although the exact relationship is difficult to quantify (Karrison 1987).

Also, note that the effect of HGB is significant at the 5% level in the PROC PHREG results but not significant in the PROC RMSTREG results. This might be due to the particular time τ that is specified for the RMST analysis, and it might also be due to the different characteristics of PROC PHREG's likelihood-based approach versus PROC RMSTREG's approach, which uses estimating equations.

The RMST regression model can also be compared to the accelerated failure time (AFT) model fitted by the LIFEREG procedure. The AFT model assumes that

$$\log(T_i) = \mathbf{x}_i \boldsymbol{\beta}_{\mathrm{AFT}} + \sigma \epsilon_i$$

where $\boldsymbol{\beta}_{\mathrm{AFT}}$ is a vector of unknown regression parameters, ϵ_i is the error term that is sampled from a known distribution, and σ is an unknown scale parameter.

The following PROC LIFEREG statements fit the AFT model with a Weibull distribution, using the same set of covariates as in the previous examples:

```
proc lifereg data=Myeloma;
   model Time*VStatus(0)=LogBUN HGB;
   store aft;
run;
```

Output 106.1.4 displays the parameter estimates for the fitted AFT regression model.

Output 106.1.4 Parameter Estimates from AFT Model

The LIFEREG Procedure

Analysis of Maximum Likelihood Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	4.5458	0.8939	2.7937	6.2979	25.86	<.0001
LogBUN	1	-1.5304	0.5031	-2.5165	-0.5444	9.25	0.0024
HGB	1	0.0975	0.0492	0.0011	0.1939	3.93	0.0474
Scale	1	0.8770	0.0945	0.7100	1.0833		
Weibull Shape	1	1.1403	0.1229	0.9231	1.4085		

The estimated effects from the AFT model exhibit the same pattern as the estimated effects from the log-linear model of PROC RMSTREG. However, the interpretations of the two models are quite different. Instead of modeling a specific measure, as PROC RMSTREG does, the AFT model assumes that the effect of the independent variables on the event time is additive on the log scale. As with the Cox regression model fit by PROC PHREG, it is difficult to translate the estimated effects from PROC LIFEREG to the scale of the RMST model.

Example 106.2: Comparing the RMSTs of Two Groups

(View the complete code for this example (rmstex2.sas) in the example repository.)

This example illustrates how to use PROC RMSTREG to compare the RMSTs of two groups while adjusting for other factors.

In a study of the human immunodeficiency virus (HIV), subjects were followed after a confirmed HIV-positive diagnosis (Hosmer and Lemeshow 1999). The primary goal is to evaluate the effect of two different covariates on mortality: the subject's age and history of prior intravenous drug use.

The following DATA step creates the data set HIV, which contains the variables Time (the follow-up time in days), Status (with a value of 0 if Time was censored and 1 otherwise), Drug (with a value of 1 for prior intravenous drug use and 0 otherwise), and Age (the patient's age in years at the beginning of the follow-up):

```
data HIV;
   input Time Age Drug Status;
   datalines;
    5      46      0        1
    6      35      1        0
    8      30      1        1
    3      30      1        1
   22      36      0        1
    1      32      1        0

   ... more lines ...

    1      34      1        1
;
```

The following statements fit a linear model for the RMST at $\tau = 48$:

```
proc rmstreg data=hiv tau=48;
   class Drug;
   model Time*Status(0) = Drug Age / link=linear;
   lsmeans Drug;
run;
```

Output 106.2.1 shows the "Model Information" table, which provides information about the specified linear model of the RMST and the input data set.

Output 106.2.1 Model Information

The RMSTREG Procedure

Model Information	
Data Set	WORK.HIV
Time Variable	Time
Censoring Variable	Status
Censoring Value(s)	0
Link Function	Linear
Estimation Method	Pseudo Value
Tau Value	48

Output 106.2.2 displays the parameter estimates for the fitted linear model:

Output 106.2.2 Parameter Estimates

Analysis of Parameter Estimates

Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	47.3681	9.0332	29.6633	65.0728	27.50	<.0001
Drug	0	1	10.1327	2.8592	4.5288	15.7366	12.56	0.0004
Drug	1	0	0.0000					
Age		1	-1.0778	0.2342	-1.5367	-0.6188	21.18	<.0001

The results show both age and drug use history are strongly associated with mortality.

The following statements fit the linear model for the RMST at $\tau = 48$ by using the method of inverse probability censoring weighting (IPCW) to illustrate that estimation method.

```
proc rmstreg data=hiv tau=48;
   class Drug;
   model Time*Status(0) = Drug Age / link=linear method=ipcw;
   lsmeans Drug;
run;
```

Output 106.2.3 displays the parameter estimates for the fitted linear model:

Output 106.2.3 Parameter Estimates

The RMSTREG Procedure

Analysis of Parameter Estimates

Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	44.6305	8.8479	27.2889	61.9720	25.44	<.0001
Drug	0	1	12.5483	2.6029	7.4468	17.6499	23.24	<.0001
Drug	1	0	0.0000					
Age		1	-1.0748	0.2289	-1.5234	-0.6262	22.05	<.0001

This method assumes homogeneity of the censoring mechanism. Such an assumption can be evaluated. The following statements use PROC LIFETEST to estimate the survival functions of censoring for the two levels of Drug. Note that the censoring value in the TIME statement uses 1 instead of 0.

```
ods graphics on;
proc lifetest data=hiv plot=s(test);
   strata Drug;
   time Time*Status(1);
run;
```

Output 106.2.4 displays the Kaplan-Meier curves for the two Drug groups.

Output 106.2.4 Plot of Estimated Survival Functions

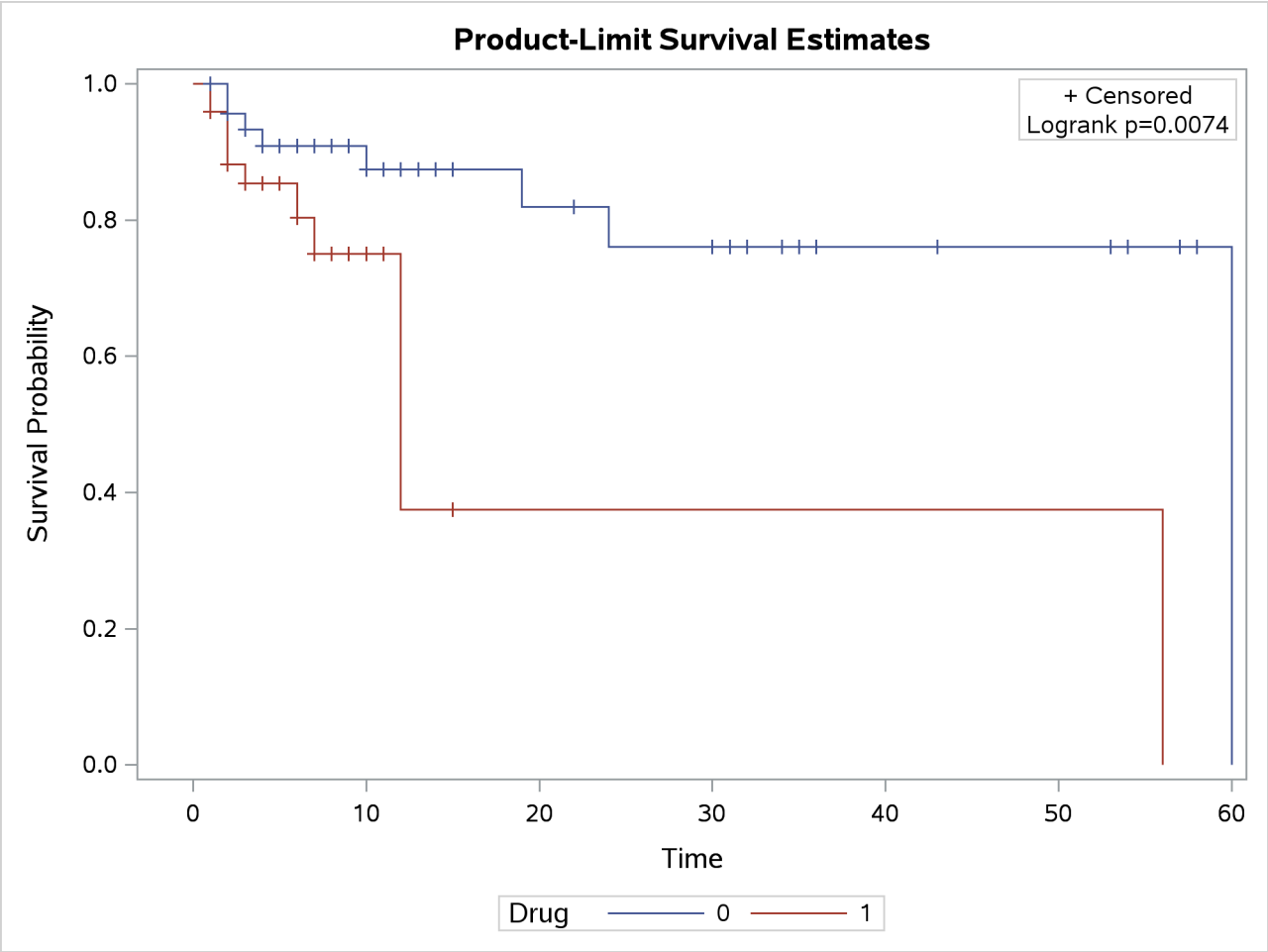

The curves show substantial separation over time. The small p-value from the log-rank test also suggests that the censoring patterns of the two groups are quite different.

By default, the IPCW method uses the Kaplan-Meier technique to obtain the weights, and this approach implicitly assumes that the censoring mechanism is homogeneous among all subjects. Sometimes this is not a reasonable assumption—for example, if there are distinct groups, such as treatment arms in randomized clinical trials. Under such circumstances, it is more appropriate to use group-specific weights by applying the Kaplan-Meier technique to different groups.

The following statements fit the linear model for the RMST at $\tau = 48$ by using the IPCW technique, with weights estimated separately for the two Drug groups:

```
proc rmstreg data=hiv tau=48;
   class Drug;
   model Time*Status(0) = Drug Age / link=linear method=ipcw(strata=Drug);
   lsmeans Drug;
run;
```

Output 106.2.5 displays the parameter estimates for the fitted linear model of the RMST.

Output 106.2.5 Parameter Estimates

The RMSTREG Procedure

Analysis of Parameter Estimates								
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	47.5159	10.5939	26.7522	68.2795	20.12	<.0001
Drug	0	1	11.4049	2.5667	6.3743	16.4354	19.74	<.0001
Drug	1	0	0.0000					
Age		1	-1.1380	0.2654	-1.6582	-0.6179	18.39	<.0001

The parameter estimates are similar to those displayed in Output 106.2.3.

Example 106.3: Making Model-Based Inferences

(View the complete code for this example (rmstex3.sas) in the example repository.)

This example reconsiders the Liver data set introduced in the section "Getting Started: RMSTREG Procedure" on page 8913. In this case, you are interested in studying RMST differences over different levels of edema, at $\tau = 10$.

The following statements fit a log-linear model for the RMST at $\tau = 10$ with covariates Bilirubin, Age, and Edema:

```
proc rmstreg data=liver tau=10;
   class Edema;
   model Time*Status(0) = Age Bilirubin Edema / link=log method=pv;
   estimate '0 vs 0.5' Edema 1 -1  0 / exp cl;
   estimate '0.5 vs 1' Edema 0  1 -1 / exp cl;
   estimate '0 vs 1'   Edema 1  0 -1 / exp cl;
run;
```

The METHOD=PV option in the MODEL statement specifies pseudovalue regression to fit the model. The ESTIMATE statements compute differences in predicted RMST on the log scale for different pairs of levels of the variable Edema. The EXP option in the ESTIMATE statements exponentiates these differences to form ratio estimates of the RMST. Standard errors and confidence intervals are also computed.

Figure 106.3.1 displays the parameter estimates from the fitted model.

Output 106.3.1 Parameter Estimates

The RMSTREG Procedure

Analysis of Parameter Estimates								
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	1.7815	0.3241	1.1462	2.4168	30.21	<.0001
Age		1	-0.0098	0.0020	-0.0138	-0.0058	23.27	<.0001
Bilirubin		1	-0.0848	0.0115	-0.1074	-0.0621	53.99	<.0001
Edema	0	1	0.9362	0.2952	0.3576	1.5148	10.06	0.0015
Edema	0.5	1	0.7356	0.3083	0.1314	1.3398	5.69	0.0170
Edema	1	0	0.0000					

Output 106.3.2 displays the results of the ESTIMATE statement.

Output 106.3.2 Ratio Estimates

	Estimate									
Label	Estimate	Standard Error	z Value	Pr > |z|	Alpha	Lower	Upper	Exponentiated	Exponentiated Lower	Exponentiated Upper
0 vs 0.5	0.2006	0.09844	2.04	0.0416	0.05	0.007668	0.3936	1.2222	1.0077	1.4822

	Estimate									
Label	Estimate	Standard Error	z Value	Pr > |z|	Alpha	Lower	Upper	Exponentiated	Exponentiated Lower	Exponentiated Upper
0.5 vs 1	0.7356	0.3083	2.39	0.0170	0.05	0.1314	1.3398	2.0867	1.1404	3.8182

	Estimate									
Label	Estimate	Standard Error	z Value	Pr > |z|	Alpha	Lower	Upper	Exponentiated	Exponentiated Lower	Exponentiated Upper
0 vs 1	0.9362	0.2952	3.17	0.0015	0.05	0.3576	1.5148	2.5503	1.4299	4.5487

The results indicate that the different levels of Edema have significantly different predicted RMST values.

An analysis that uses LS-means can provide more detail for these Edema differences in predicted RMST. In the following statements, the LSMEANS statement is specified with several options: the E option displays the coefficients that are used to compute the LS-means for each Treatment level; the DIFF option takes all pairwise differences of the LS-means for the levels of the Treatment variable; the ILINK option exponentiates these differences to form ratio estimates; the CL option produces confidence intervals for the differences and ratios; and the ADJUST=BON option performs a very conservative adjustment of the p-values and confidence intervals.

```
proc rmstreg data=liver tau=10;
   class Edema;
   model Time*Status(0) = Age Bilirubin Edema / link=log method=pv;
   lsmeans Edema / e diff ilink cl adjust=bon;
run;
```

The results from the LSMEANS statement are displayed in Output 106.3.3 through Output 106.3.5.

The LS-means are computed by constructing each of the $\boldsymbol{l}$ coefficient vectors shown in Output 106.3.3

and then computing $\boldsymbol{l}'\boldsymbol{\beta}$. The LS-means are not estimates of the RMST; they are estimates of the linear predictors on the logit scale and therefore are estimated log differences. To obtain the RMST, you need to apply the inverse-link transformation by specifying the ILINK option in the LSMEANS statement. For more information about the construction of LS-means, see the section "Construction of Least Squares Means" on page 4244 in Chapter 53, "The GLM Procedure."

Output 106.3.3 Edema LS-Means Coefficients

The RMSTREG Procedure

Coefficients for Edema Least Squares Means				
Parameter	**Edema**	**Row1**	**Row2**	**Row3**
Intercept		1	1	1
Age		50.741	50.741	50.741
Bilirubin		3.2208	3.2208	3.2208
Edema 0	0	1		
Edema 0.5	0.5		1	
Edema 1	1			1

The Edema LS-means shown in Output 106.3.4 are all significantly nonzero at the 0.05 level. These LS-means are *predicted population margins* of the RMST; that is, they estimate the marginal means over a balanced population, and they are effectively the within-Edema means appropriately adjusted for the other effects in the model.

Output 106.3.4 Edema LS-Means

Edema Least Squares Means											
Edema	**Estimate**	**Standard Error**	**z Value**	**Pr > |z|**	**Alpha**	**Lower**	**Upper**	**Mean**	**Standard Error of Mean**	**Lower Mean**	**Upper Mean**
0	1.9480	0.02857	68.19	<.0001	0.05	1.8920	2.0040	7.0147	0.2004	6.6327	7.4187
0.5	1.7474	0.09663	18.08	<.0001	0.05	1.5580	1.9368	5.7396	0.5546	4.7494	6.9364
1	1.0118	0.2914	3.47	0.0005	0.05	0.4406	1.5830	2.7505	0.8016	1.5536	4.8696

Pairwise differences between the Edema LS-means, requested using the DIFF option, are displayed in Output 106.3.5. The LS-mean for the level that is displayed in the _Edema column is subtracted from the LS-mean for the level in the Edema column, so the first row displays the LS-mean for Edema level 0 minus the LS-mean for Edema level 0.5. The difference (0.2006) is the estimated difference in log scale, or equivalently the log ratio of the two Edema levels. The Pr > |z| column indicates that the 0 and 0.5 levels are not significantly different; however, both levels are significantly different from level 1. If the inverse-link transformation is specified by the ILINK option, then these differences do not transform back to differences in the RMST.

Because multiple tests are performed, you can protect yourself from falsely significant results by adjusting your *p*-values for multiplicity. The ADJUST=BON option performs the conservative Bonferroni adjustment and adds the columns labeled with 'Adj' to Output 106.3.5. By comparing the Pr > |z| column to the Adj P column, you can see that the *p*-values are adjusted upward and two of the three values are now greater than the alpha level. The confidence intervals are also adjusted for multiplicity—all adjusted intervals are wider than the unadjusted intervals, but the results are unchanged. You can specify other adjustment methods by using the ADJUST= option.

Output 106.3.5 Differences and RMST Ratios for the Edema LS-Means

Differences of Edema Least Squares Means
Adjustment for Multiple Comparisons: Bonferroni

Edema	_Edema	Estimate	Standard Error	z Value	Pr > |z|	Adj P	Alpha	Lower	Upper	Adj Lower	Adj Upper
0	0.5	0.2006	0.09844	2.04	0.0416	0.1247	0.05	0.007668	0.3936	-0.03506	0.4363
0	1	0.9362	0.2952	3.17	0.0015	0.0046	0.05	0.3576	1.5148	0.2295	1.6430
0.5	1	0.7356	0.3083	2.39	0.0170	0.0511	0.05	0.1314	1.3398	-0.00237	1.4736

If you want to jointly test whether the three Edema levels are different, you can specify a custom hypothesis test among LS-means by using the LSMESTIMATE statement. In the following statements, the LS-means for the two Edema levels are contrasted with the LS-mean for the third level, and the JOINT option performs a joint test that the three Edema levels are the same:

```
proc rmstreg data=liver tau=10;
   class Edema;
   model Time*Status(0) = Age Bilirubin Edema / link=log method=pv;
   lsmestimate Edema 1 0 -1, 0 1 -1 / joint;
run;
```

Output 106.3.6 displays the results from the LSMESTIMATE statement. The "Least Squares Means Estimates" table displays the requested differences, and the results are identical to the second and third rows of Output 106.3.5. The "Chi-Square Test for Least Squares Means Estimates" table displays the joint test. In all these tests, you reject the null hypothesis that the three Edema levels are the same.

Output 106.3.6 Custom LS-Mean Tests

The RMSTREG Procedure

Least Squares Means Estimates

Effect	Label	Estimate	Standard Error	z Value	Pr > |z|
Edema	Row 1	0.9362	0.2952	3.17	0.0015
Edema	Row 2	0.7356	0.3083	2.39	0.0170

Chi-Square Test for Least Squares Means Estimates

Effect	Num DF	Chi-Square	Pr > ChiSq
Edema	2	13.82	0.0010

References

Andersen, P. K., Hansen, M. G., and Klein, J. P. (2004). "Regression Analysis of Restricted Mean Survival Time Based on Pseudo-observations." *Lifetime Data Analysis* 10:335–350.

Andersen, P. K., Klein, J. P., and Rosthøj, S. (2003). "Generalised Linear Models for Correlated Pseudo-observations, with Applications to Multi-state Models." *Biometrika* 90:15–27.

Hosmer, D. W., Jr., and Lemeshow, S. (1999). *Applied Survival Analysis: Regression Modeling of Time-to-Event Data*. New York: John Wiley & Sons.

Karrison, T. (1987). "Restricted Mean Life with Adjustment for Covariates." *Journal of the American Statistical Association* 82:1169–1176.

Klein, J. P., and Moeschberger, M. L. (2003). *Survival Analysis: Techniques for Censored and Truncated Data*. 2nd ed. New York: Springer-Verlag.

Krall, J. M., Uthoff, V. A., and Harley, J. B. (1975). "A Step-Up Procedure for Selecting Variables Associated with Survival." *Biometrics* 31:49–57.

Liang, K.-Y., and Zeger, S. L. (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika* 73:13–22.

Lin, D. Y., Wei, L. J., and Ying, Z. (1993). "Checking the Cox Model with Cumulative Sums of Martingale-Based Residuals." *Biometrika* 80:557–572.

Littell, R. C., Freund, R. J., and Spector, P. C. (1991). *SAS System for Linear Models*. 3rd ed. Cary, NC: SAS Institute Inc.

Muller, K. E., and Fetterman, B. A. (2002). *Regression and ANOVA: An Integrated Approach Using SAS Software*. Cary, NC: SAS Institute Inc.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Royston, P., and Parmar, M. K. B. (2013). "Restricted Mean Survival Time: An Alternative to the Hazard Ratio for the Design and Analysis of Randomized Trials with a Time-to-Event Outcome." *BMC Medical Research Methodology* 13:152–166.

Tian, L., Zhao, L., and Wei, L. J. (2014). "Predicting the Restricted Mean Event Time with the Subject's Baseline Covariates in Survival Analysis." *Biostatistics* 15:222–233.

Trinquart, L., Jacot, J., Conner, S. C., and Porcher, R. (2016). "Comparison of Treatment Effects Measured by the Hazard Ratio and by the Ratio of Restricted Mean Survival Times in Oncology Randomized Controlled Trials." *Journal of Clinical Oncology* 34:1813–1819.

Uno, H., Claggett, B., Tian, L., Inoue, E., Gallo, P., Miyata, T., Schrag, D., et al. (2014). "Moving Beyond the Hazard Ratio in Quantifying the Between-Group Difference in Survival Analysis." *Journal of Clinical Oncology* 32:2380–2385.

Zucker, D. M. (1998). "Restricted Mean Life with Covariates: Modification and Extension of a Useful Survival Analysis Method." *Journal of the American Statistical Association* 93:702–709.

Chapter 107
The ROBUSTREG Procedure

Contents

Overview: ROBUSTREG Procedure . . . **8956**
- Features . . . 8956

Getting Started: ROBUSTREG Procedure . . . **8957**
- M Estimation . . . 8957
- LTS Estimation . . . 8964

Syntax: ROBUSTREG Procedure . . . **8967**
- PROC ROBUSTREG Statement . . . 8968
- BY Statement . . . 8976
- CLASS Statement . . . 8976
- EFFECT Statement . . . 8977
- ID Statement . . . 8978
- MODEL Statement . . . 8979
- OUTPUT Statement . . . 8981
- PERFORMANCE Statement . . . 8983
- TEST Statement . . . 8984
- WEIGHT Statement . . . 8984

Details: ROBUSTREG Procedure . . . **8984**
- M Estimation . . . 8985
- High Breakdown Value Estimation . . . 8991
- MM Estimation . . . 8996
- Robust Distance . . . 9000
- Leverage-Point and Outlier Detection . . . 9006
- Implementation of the WEIGHT Statement . . . 9007
- INEST= Data Set . . . 9008
- OUTEST= Data Set . . . 9009
- Computational Resources . . . 9010
- ODS Table Names . . . 9010
- ODS Graphics . . . 9012

Examples: ROBUSTREG Procedure . . . **9015**
- Example 107.1: Comparison of Robust Estimates . . . 9015
- Example 107.2: Robust ANOVA . . . 9022
- Example 107.3: Growth Study of De Long and Summers . . . 9025
- Example 107.4: Constructed Effects . . . 9032
- Example 107.5: Robust Diagnostics . . . 9039

References . . . **9046**

Overview: ROBUSTREG Procedure

The main purpose of robust regression is to detect outliers and provide resistant (stable) results in the presence of outliers. In order to achieve this stability, robust regression limits the influence of outliers. Historically, robust regression techniques have addressed three classes of problems:

- problems with outliers in the Y direction (response direction)
- problems with multivariate outliers in the X space (that is, outliers in the covariate space, which are also referred to as leverage points)
- problems with outliers in both the Y direction and the X space

Many methods have been developed in response to these problems. However, in statistical applications of outlier detection and robust regression, the methods that are most commonly used today are Huber M estimation, high breakdown value estimation, and combinations of these two methods. The ROBUSTREG procedure provides four such methods: M estimation, LTS estimation, S estimation, and MM estimation.

- M estimation, introduced by Huber (1973), is the simplest approach both computationally and theoretically. Although it is not robust with respect to leverage points, it is still used extensively in data analysis when contamination can be assumed to be mainly in the response direction.
- Least trimmed squares (LTS) estimation is a high breakdown value method that was introduced by Rousseeuw (1984). The breakdown value is a measure of the proportion of contamination that an estimation method can withstand and still maintain its robustness. The performance of this method was improved by the FAST-LTS algorithm of Rousseeuw and Van Driessen (2000).
- S estimation is a high breakdown value method that was introduced by Rousseeuw and Yohai (1984). Given the same breakdown value, S estimation has a higher statistical efficiency than LTS estimation.
- MM estimation, introduced by Yohai (1987), combines high breakdown value estimation and M estimation. It has the same high breakdown property as S estimation but a higher statistical efficiency.

Features

The ROBUSTREG procedure has the following main features:

- offers four estimation methods: M, LTS, S, and MM
- provides 10 weight functions for M estimation
- provides robust R square and deviance for all estimates
- provides asymptotic covariance and confidence intervals for regression parameters by using the M, S, and MM methods

- provides robust Wald and F tests for regression parameters by using the M and MM methods
- provides Mahalanobis distance and robust Mahalanobis distance by using the generalized minimum covariance determinant (MCD) algorithm
- provides outlier and leverage-point diagnostics
- supports parallel computing for S and LTS estimates
- supports constructed effects, including spline and multimember effects
- produces fit plots and diagnostic plots by using ODS Graphics

Getting Started: ROBUSTREG Procedure

The following examples demonstrate how you can use the ROBUSTREG procedure to fit a linear regression model and obtain outlier and leverage-point diagnostics.

M Estimation

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/rreggs1.sas.)

This example shows how you can use the ROBUSTREG procedure to do M estimation, which is a commonly used method for outlier detection and robust regression when contamination is mainly in the response direction.

The following data set, Stack, is the well-known stack loss data set presented by Brownlee (1965). The data describe the operation of a plant for the oxidation of ammonia to nitric acid and consist of 21 four-dimensional observations. The explanatory variables for the response stack loss (y) are the rate of operation (x1), the cooling water inlet temperature (x2), and the acid concentration (x3).

```
data Stack;
   input  x1 x2 x3 y exp $ @@;
   datalines;
80  27   89  42   e1   80  27   88  37   e2
75  25   90  37   e3   62  24   87  28   e4
62  22   87  18   e5   62  23   87  18   e6
62  24   93  19   e7   62  24   93  20   e8
58  23   87  15   e9   58  18   80  14  e10
58  18   89  14  e11   58  17   88  13  e12
58  18   82  11  e13   58  19   93  12  e14
50  18   89   8  e15   50  18   86   7  e16
50  19   72   8  e17   50  19   79   8  e18
50  20   80   9  e19   56  20   82  15  e20
70  20   91  15  e21
;
```

The following PROC ROBUSTREG statements analyze the data:

```
proc robustreg data=stack;
   model y = x1 x2 x3 / diagnostics leverage;
   id    exp;
   test  x3;
run;
```

By default, the ROBUSTREG procedure uses the bisquare weight function to do M estimation, and it uses the median method to estimate the scale parameter. The MODEL statement specifies the covariate effects. The DIAGNOSTICS option requests a table for outlier diagnostics, and the LEVERAGE option adds leverage-point diagnostic results to this table for continuous covariate effects. The ID statement specifies that the variable exp be used to identify each observation (experiment) in this table. If the ID statement is omitted, the observation number is used to identify the observations. The TEST statement requests a test of significance for the covariate effects that are specified. The results of this analysis are displayed in Figures 107.1 through 107.3.

Figure 107.1 displays the model-fitting information and summary statistics for the response variable and the continuous covariates. The Q1, Median, and Q3 columns provide the lower quantile, median, and upper quantile, respectively. The MAD column provides a robust estimate of the univariate scale, which is computed as the standardized median absolute deviation (MAD). For more information about the standardized MAD, see Huber (1981, p. 108). The Mean and Standard Deviation columns provide the usual mean and standard deviation. A large difference between the standard deviation and the MAD for a variable indicates some extreme values for that variable. In the stack loss data, the stack loss (response y) has the biggest difference between the standard deviation and the MAD.

Figure 107.1 Model-Fitting Information and Summary Statistics

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.STACK
Dependent Variable	y
Number of Independent Variables	3
Number of Observations	21
Method	M Estimation

Summary Statistics

Variable	Q1	Median	Q3	Mean	Standard Deviation	MAD
x1	53.0000	58.0000	62.0000	60.4286	9.1683	5.9304
x2	18.0000	20.0000	24.0000	21.0952	3.1608	2.9652
x3	82.0000	87.0000	89.5000	86.2857	5.3586	4.4478
y	10.0000	15.0000	19.5000	17.5238	10.1716	5.9304

Figure 107.2 displays the table of robust parameter estimates, standard errors, and confidence limits. The Scale row provides a point estimate of the scale parameter in the linear regression model, which is obtained by the median method. For more information about scale estimation methods, see the section "M Estimation" on page 8985. For the stack loss data, M estimation yields the fitted linear model:

$$\hat{y} = -42.2845 + 0.9276\text{x1} + 0.6507\text{x2} - 0.1123\text{x3}$$

Figure 107.2 Model Parameter Estimates

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-42.2854	9.5045	-60.9138	-23.6569	19.79	<.0001
x1	1	0.9276	0.1077	0.7164	1.1387	74.11	<.0001
x2	1	0.6507	0.2940	0.0744	1.2270	4.90	0.0269
x3	1	-0.1123	0.1249	-0.3571	0.1324	0.81	0.3683
Scale	1	2.2819					

Figure 107.3 displays outlier and leverage-point diagnostics. Standardized robust residuals are computed based on the estimated parameters. Both the Mahalanobis distance and the robust MCD distance are displayed. Outliers and leverage points, identified by asterisks, are defined by the standardized robust residuals and robust MCD distances that exceed the corresponding cutoff values displayed in the diagnostics summary. Observations 4 and 21 are outliers because their standardized robust residuals exceed the cutoff value in absolute value. The procedure detects four observations that have high leverage values. Leverage points (points with high leverage values) that have smaller standardized robust residuals than the cutoff value in absolute value are called good leverage points; others are called bad leverage points. Observation 21 is a bad leverage point.

Figure 107.3 Diagnostics

Diagnostics						
Obs	exp	Mahalanobis Distance	Robust MCD Distance	Leverage	Standardized Robust Residual	Outlier
1	e1	2.2536	5.5284	*	1.0995	
2	e2	2.3247	5.6374	*	-1.1409	
3	e3	1.5937	4.1972	*	1.5604	
4	e4	1.2719	1.5887		3.0381	*
21	e21	2.1768	3.6573	*	-4.5733	*

Two particularly useful plots for revealing outliers and leverage points are a scatter plot of the standardized robust residuals against the robust distances (RD plot) and a scatter plot of the robust distances against the classical Mahalanobis distances (DD plot).

For the stack loss data, the following statements produce the RD plot in Figure 107.4 and the DD plot in Figure 107.5. The histogram and the normal quantile-quantile plots (shown in Figure 107.6 and Figure 107.7, respectively) for the standardized robust residuals are also created using the HISTOGRAM and QQPLOT suboptions of the PLOTS= option.

```
ods graphics on;

proc robustreg data=stack plots=(rdplot ddplot histogram qqplot);
   model y = x1 x2 x3;
run;

ods graphics off;
```

These plots are helpful in identifying outliers in addition to good and bad high-leverage points.

These plots are requested when ODS Graphics is enabled by specifying the PLOTS= option in the PROC ROBUSTREG statement. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the ROBUSTREG procedure, see the section "ODS Graphics" on page 9012.

Figure 107.4 RD Plot for Stack Loss Data

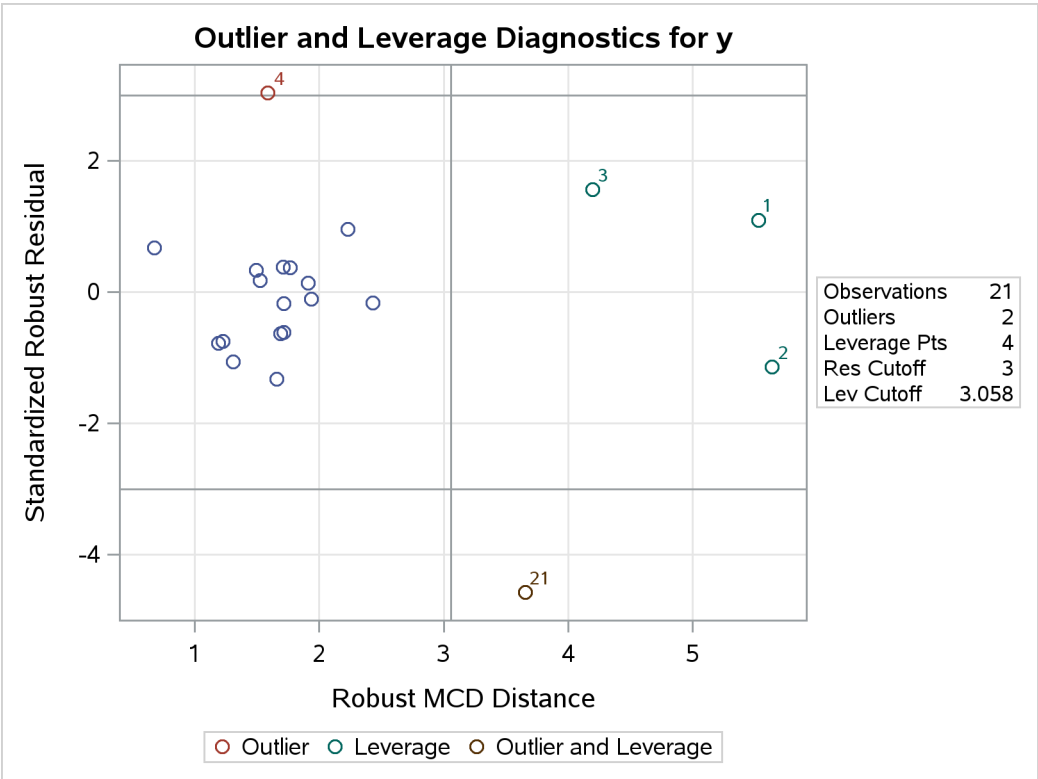

Figure 107.5 DD Plot for Stack Loss Data

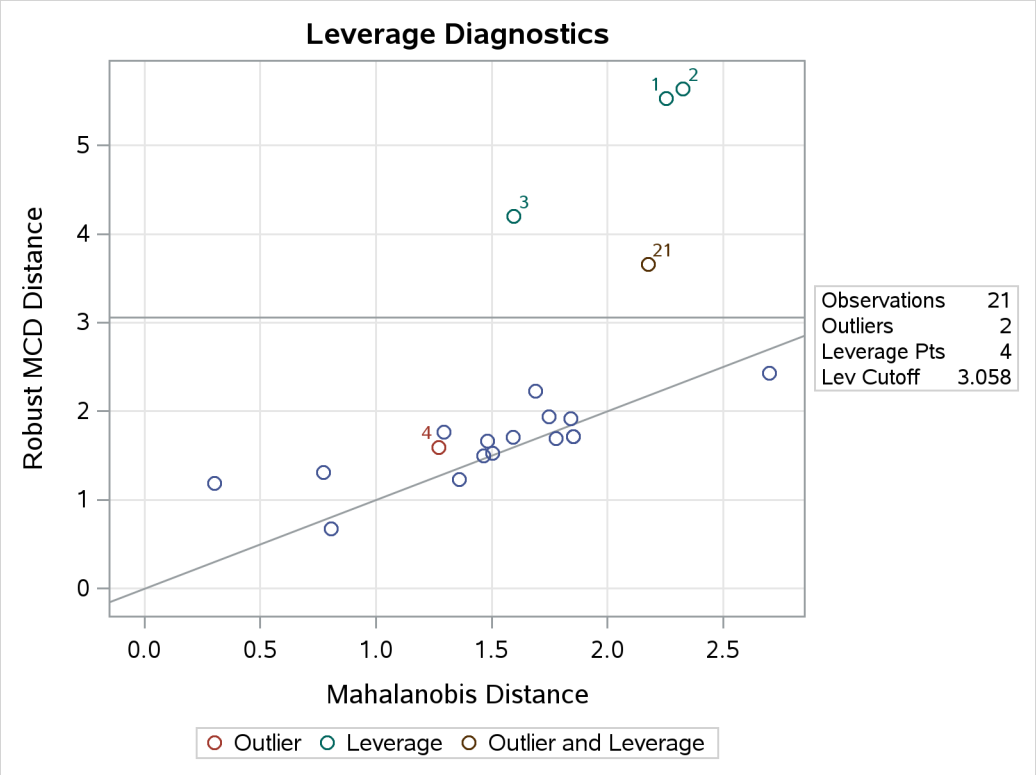

Figure 107.6 Histogram

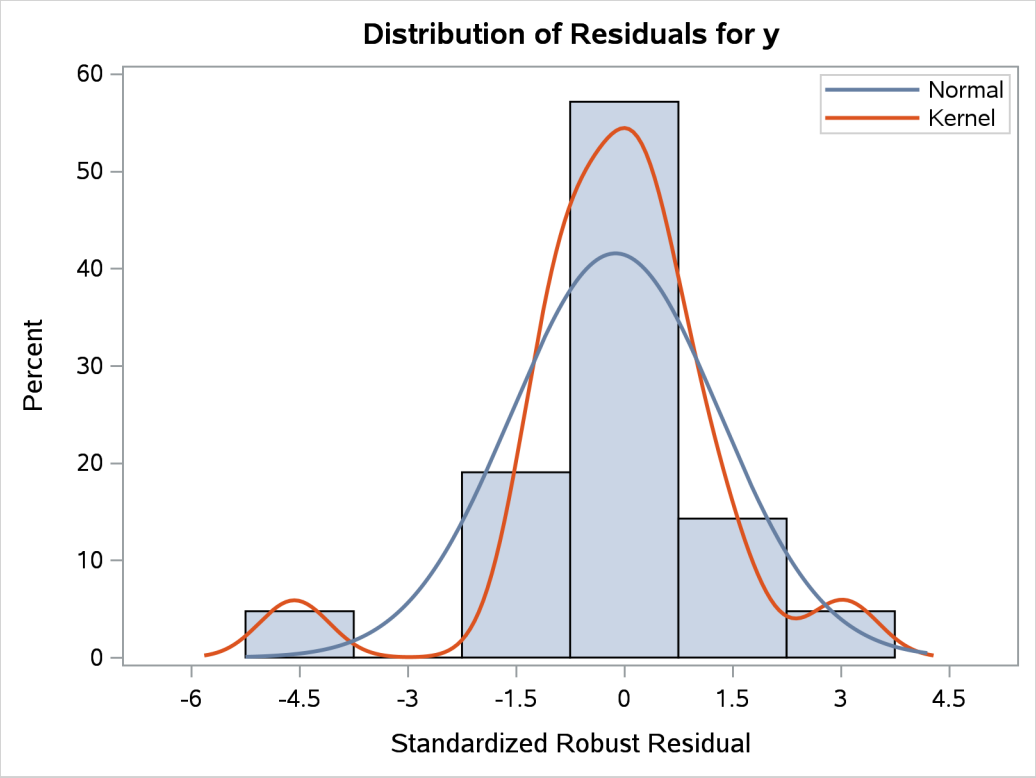

Figure 107.7 Q-Q Plot

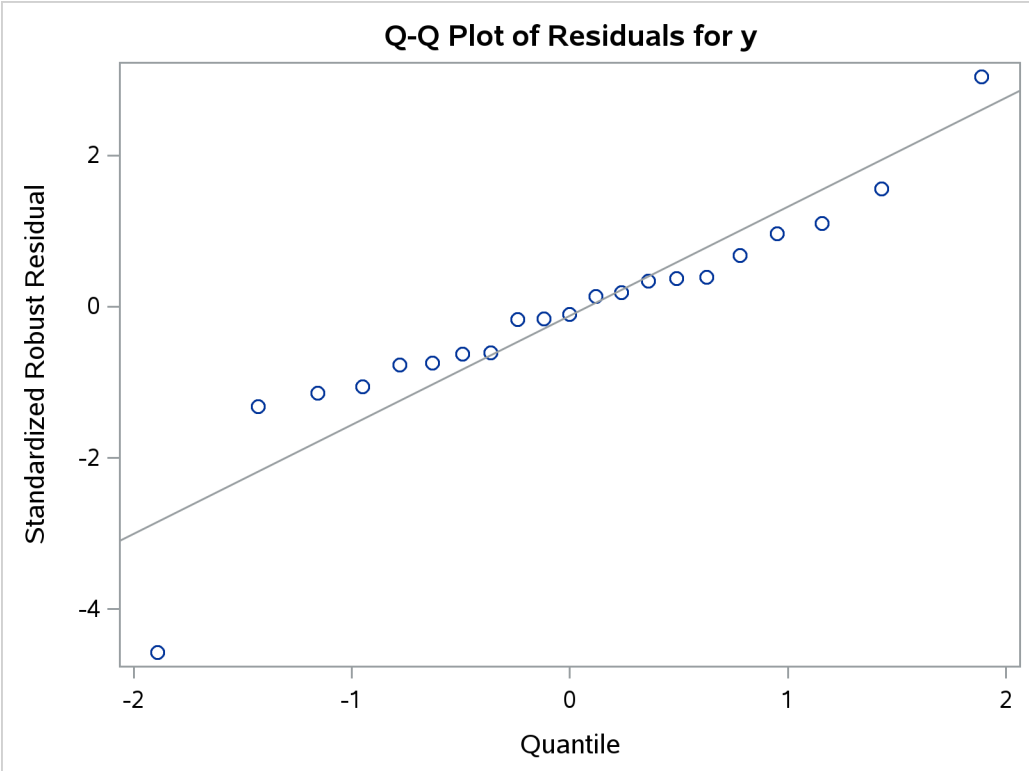

Figure 107.8 displays robust versions of goodness-of-fit statistics for the model. You can use the robust information criteria, AICR and BICR, for model selection and comparison. For both AICR and BICR, the lower the value, the more desirable the model.

Figure 107.8 Goodness-of-Fit Statistics

Goodness-of-Fit	
Statistic	**Value**
R-Square	0.6659
AICR	29.5231
BICR	36.3361
Deviance	125.7905

Figure 107.9 displays the test results that are requested by the TEST statement. The ROBUSTREG procedure conducts two robust linear tests, the ρ test and the R_n^2 test. For information about how the ROBUSTREG procedure computes test statistics and the correction factor lambda, see the section "Linear Tests" on page 8990. Because of the large p-values for both tests, you can conclude that the effect x3 is not significant at the 5% level.

Figure 107.9 Test of Significance

Robust Linear Test					
Test	Test Statistic	Lambda	DF	Chi-Square	Pr > ChiSq
Rho	0.9378	0.7977	1	1.18	0.2782
Rn2	0.8092		1	0.81	0.3683

For the bisquare weight function, the default tuning constant, $c = 4.685$, is chosen to yield a 95% asymptotic efficiency of the M estimates with the Gaussian distribution. For more information, see the section "M Estimation" on page 8985. The smaller the constant c, the lower the asymptotic efficiency but the sharper the M estimate as an outlier detector. For the stack loss data set, you could consider using a sharper outlier detector.

The following PROC ROBUSTREG step uses a smaller constant, $c = 3.5$. This tuning constant corresponds to an efficiency close to 85%. For the relationship between the tuning constant and asymptotic efficiency of M estimates, see Chen and Yin (2002).

```
proc robustreg method=m(wf=bisquare(c=3.5)) data=stack;
   model y = x1 x2 x3 / diagnostics leverage;
   id    exp;
   test  x3;
run;
```

Figure 107.10 displays the table of robust parameter estimates, standard errors, and confidence limits when PROC ROBUSTREG uses the constant $c = 3.5$.

The refitted linear model is

$$\hat{y} = -37.1076 + 0.8191x1 + 0.5173x2 - 0.0728x3$$

Figure 107.10 Model Parameter Estimates

The ROBUSTREG Procedure

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-37.1076	5.4731	-47.8346	-26.3805	45.97	<.0001
x1	1	0.8191	0.0620	0.6975	0.9407	174.28	<.0001
x2	1	0.5173	0.1693	0.1855	0.8492	9.33	0.0022
x3	1	-0.0728	0.0719	-0.2138	0.0681	1.03	0.3111
Scale	1	1.4265					

Figure 107.11 displays outlier and leverage-point diagnostics when the constant $c = 3.5$ is used. Besides observations 4 and 21, observations 1 and 3 are also detected as outliers.

Figure 107.11 Diagnostics

Diagnostics						
Obs	exp	Mahalanobis Distance	Robust MCD Distance	Leverage	Standardized Robust Residual	Outlier
1	e1	2.2536	5.5284	*	4.2719	*
2	e2	2.3247	5.6374	*	0.7158	
3	e3	1.5937	4.1972	*	4.4142	*
4	e4	1.2719	1.5887		5.7792	*
21	e21	2.1768	3.6573	*	-6.2727	*

LTS Estimation

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/rreggs2.sas.)

If the data are contaminated in the X space, M estimation might yield improper results. It is better to use the high breakdown value method. This example shows how you can use LTS estimation to deal with X-space contaminated data. The following data set, hbk, is an artificial data set that was generated by Hawkins, Bradu, and Kass (1984).

```
data hbk;
   input index $ x1 x2 x3 y @@;
   datalines;
 1  10.1  19.6  28.3   9.7          2   9.5  20.5  28.9  10.1
 3  10.7  20.2  31.0  10.3          4   9.9  21.5  31.7   9.5
 5  10.3  21.1  31.1  10.0          6  10.8  20.4  29.2  10.0
 7  10.5  20.9  29.1  10.8          8   9.9  19.6  28.8  10.3
 9   9.7  20.7  31.0   9.6         10   9.3  19.7  30.3   9.9
11  11.0  24.0  35.0  -0.2         12  12.0  23.0  37.0  -0.4
13  12.0  26.0  34.0   0.7         14  11.0  34.0  34.0   0.1
15   3.4   2.9   2.1  -0.4         16   3.1   2.2   0.3   0.6
17   0.0   1.6   0.2  -0.2         18   2.3   1.6   2.0   0.0
19   0.8   2.9   1.6   0.1         20   3.1   3.4   2.2   0.4
21   2.6   2.2   1.9   0.9         22   0.4   3.2   1.9   0.3
23   2.0   2.3   0.8  -0.8         24   1.3   2.3   0.5   0.7
25   1.0   0.0   0.4  -0.3         26   0.9   3.3   2.5  -0.8
27   3.3   2.5   2.9  -0.7         28   1.8   0.8   2.0   0.3
29   1.2   0.9   0.8   0.3         30   1.2   0.7   3.4  -0.3
31   3.1   1.4   1.0   0.0         32   0.5   2.4   0.3  -0.4
33   1.5   3.1   1.5  -0.6         34   0.4   0.0   0.7  -0.7
35   3.1   2.4   3.0   0.3         36   1.1   2.2   2.7  -1.0
37   0.1   3.0   2.6  -0.6         38   1.5   1.2   0.2   0.9
39   2.1   0.0   1.2  -0.7         40   0.5   2.0   1.2  -0.5
41   3.4   1.6   2.9  -0.1         42   0.3   1.0   2.7  -0.7
43   0.1   3.3   0.9   0.6         44   1.8   0.5   3.2  -0.7
45   1.9   0.1   0.6  -0.5         46   1.8   0.5   3.0  -0.4
47   3.0   0.1   0.8  -0.9         48   3.1   1.6   3.0   0.1
49   3.1   2.5   1.9   0.9         50   2.1   2.8   2.9  -0.4
51   2.3   1.5   0.4   0.7         52   3.3   0.6   1.2  -0.5
```

```
53   0.3   0.4   3.3   0.7        54   1.1   3.0   0.3   0.7
55   0.5   2.4   0.9   0.0        56   1.8   3.2   0.9   0.1
57   1.8   0.7   0.7   0.7        58   2.4   3.4   1.5  -0.1
59   1.6   2.1   3.0  -0.3        60   0.3   1.5   3.3  -0.9
61   0.4   3.4   3.0  -0.3        62   0.9   0.1   0.3   0.6
63   1.1   2.7   0.2  -0.3        64   2.8   3.0   2.9  -0.5
65   2.0   0.7   2.7   0.6        66   0.2   1.8   0.8  -0.9
67   1.6   2.0   1.2  -0.7        68   0.1   0.0   1.1   0.6
69   2.0   0.6   0.3   0.2        70   1.0   2.2   2.9   0.7
71   2.2   2.5   2.3   0.2        72   0.6   2.0   1.5  -0.2
73   0.3   1.7   2.2   0.4        74   0.0   2.2   1.6  -0.9
75   0.3   0.4   2.6   0.2
;
```

Both ordinary least squares (OLS) estimation and M estimation (not shown here) suggest that observations 11 to 14 are outliers. However, these four observations were generated from the underlying model, whereas observations 1 to 10 were contaminated. The reason that OLS estimation and M estimation do not pick up the contaminated observations is that they cannot distinguish good leverage points (observations 11 to 14) from bad leverage points (observations 1 to 10). In such cases, the LTS method identifies the true outliers.

The following statements invoke the ROBUSTREG procedure and use the LTS estimation method:

```
proc robustreg data=hbk fwls method=lts;
   model y = x1 x2 x3 / diagnostics leverage;
   id index;
run;
```

Figure 107.12 displays the model-fitting information and summary statistics for the response variable and independent covariates.

Figure 107.12 Model-Fitting Information and Summary Statistics

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.HBK
Dependent Variable	y
Number of Independent Variables	3
Number of Observations	75
Method	LTS Estimation

Summary Statistics

Variable	Q1	Median	Q3	Mean	Standard Deviation	MAD
x1	0.8000	1.8000	3.1000	3.2067	3.6526	1.9274
x2	1.0000	2.2000	3.3000	5.5973	8.2391	1.6309
x3	0.9000	2.1000	3.0000	7.2307	11.7403	1.7791
y	-0.5000	0.1000	0.7000	1.2787	3.4928	0.8896

Figure 107.13 displays information about the LTS fit, which includes the breakdown value of the LTS estimate. The breakdown value is a measure of the proportion of contamination that an estimation method can withstand and still maintain its robustness. In this example the LTS estimate minimizes the sum of 57 smallest squares of residuals. It can still estimate the true underlying model if the remaining 18 observations

are contaminated. This corresponds to a breakdown value around 0.25, which is set as the default.

Figure 107.13 LTS Profile

LTS Profile	
Total Number of Observations	75
Number of Squares Minimized	57
Number of Coefficients	4
Highest Possible Breakdown Value	0.2533

Figure 107.14 displays parameter estimates for covariates and scale. Two robust estimates of the scale parameter are displayed. For information about computing these estimates, see the section "Final Weighted Scale Estimator" on page 8993. The weighted scale estimator (Wscale) is a more efficient estimator of the scale parameter.

Figure 107.14 LTS Parameter Estimates

LTS Parameter Estimates		
Parameter	DF	Estimate
Intercept	1	-0.3431
x1	1	0.0901
x2	1	0.0703
x3	1	-0.0731
Scale (sLTS)	0	0.7451
Scale (Wscale)	0	0.5749

Figure 107.15 displays outlier and leverage-point diagnostics. The ID variable index is used to identify the observations. If you do not specify this ID variable, the observation number is used to identify the observations. However, the observation number depends on how the data are read. The first 10 observations are identified as outliers, and observations 11 to 14 are identified as good leverage points.

Figure 107.15 Diagnostics

Diagnostics						
Obs	index	Mahalanobis Distance	Robust MCD Distance	Leverage	Standardized Robust Residual	Outlier
1	1	1.9168	29.4424	*	17.0868	*
2	2	1.8558	30.2054	*	17.8428	*
3	3	2.3137	31.8909	*	18.3063	*
4	4	2.2297	32.8621	*	16.9702	*
5	5	2.1001	32.2778	*	17.7498	*
6	6	2.1462	30.5892	*	17.5155	*
7	7	2.0105	30.6807	*	18.8801	*
8	8	1.9193	29.7994	*	18.2253	*
9	9	2.2212	31.9537	*	17.1843	*
10	10	2.3335	30.9429	*	17.8021	*
11	11	2.4465	36.6384	*	0.0406	
12	12	3.1083	37.9552	*	-0.0874	
13	13	2.6624	36.9175	*	1.0776	
14	14	6.3816	41.0914	*	-0.7875	

Figure 107.16 displays the final weighted least squares estimates. These estimates are least squares estimates that are computed after the detected outliers are deleted.

Figure 107.16 Final Weighted LS Estimates

Parameter Estimates for Final Weighted Least Squares Fit							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-0.1805	0.1044	-0.3852	0.0242	2.99	0.0840
x1	1	0.0814	0.0667	-0.0493	0.2120	1.49	0.2222
x2	1	0.0399	0.0405	-0.0394	0.1192	0.97	0.3242
x3	1	-0.0517	0.0354	-0.1210	0.0177	2.13	0.1441
Scale	0	0.5572					

Syntax: ROBUSTREG Procedure

The following statements are available in the ROBUSTREG procedure:

```
PROC ROBUSTREG <options> ;
    BY variables ;
    CLASS variables ;
    EFFECT name=effect-type(variables < / options>) ;
    ID variables ;
    MODEL response = < effects> < / options> ;
    OUTPUT <OUT=SAS-data-set> keyword=name < . . . keyword=name> ;
    PERFORMANCE < options> ;
    TEST  effects  ;
    WEIGHT variable ;
```

The PROC ROBUSTREG statement invokes the procedure. The METHOD= option in the PROC ROBUSTREG statement selects one of the four estimation methods, M, LTS, S, and MM. By default, Huber M estimation is used. The MODEL statement is required and specifies the variables to be used in the regression. You can specify main effects and interaction terms in the MODEL statement, as in the GLM procedure. (See Chapter 53, "The GLM Procedure.") The CLASS statement specifies which explanatory variables to treat as categorical. The ID statement names variables to identify observations in the outlier diagnostics tables. The WEIGHT statement identifies a variable in the input data set whose values are used to weight the observations. The OUTPUT statement creates an output data set that contains final weights, predicted values, and residuals. The TEST statement requests robust linear tests for the model parameters. The PERFORMANCE statement tunes the performance of the procedure by using single or multiple processors available on the hardware. Multiple OUTPUT and TEST statements are permitted.

PROC ROBUSTREG Statement

PROC ROBUSTREG < *options* > **;**

The PROC ROBUSTREG statement invokes the ROBUSTREG procedure. Table 107.1 summarizes the *options* available in the PROC ROBUSTREG statement.

Table 107.1 PROC ROBUSTREG Statement Options

Option	Description
COVOUT	Saves the estimated covariance matrix
DATA=	Specifies the input SAS data set
FWLS	Computes the final weighted least squares estimates
INEST=	Specifies an input SAS data set that contains initial estimates
ITPRINT	Displays the iteration history of the iteratively reweighted least squares algorithm
METHOD=	Specifies the estimation method
NAMELEN=	Specifies the length of effect names
ORDER=	Specifies the order in which to sort classification variables
OUTEST=	Specifies an output SAS data set that contains the parameter estimates
PLOT	Specifies options that control details of the plots
SEED=	Specifies the seed for the random number generator

You can specify the following *options* in the PROC ROBUSTREG statement.

COVOUT
saves the estimated covariance matrix in the OUTEST= data set. This option is not supported for LTS estimation.

DATA=*SAS-data-set*
specifies the input SAS data set to be used by PROC ROBUSTREG. By default, the most recently created SAS data set is used.

FWLS
computes the final weighted least squares estimates. These estimates are equivalent to the least squares estimates after the detected outliers are deleted.

INEST=*SAS-data-set*
specifies an input SAS data set that contains initial estimates for all the parameters in the model. For a detailed description of the contents of the INEST= data set, see the section "INEST= Data Set" on page 9008.

ITPRINT
displays the iteration history of the iteratively reweighted least squares algorithm that is used in M and MM estimation. You can also use this option in the MODEL statement.

METHOD=*method-type* **<(***options***)>**

specifies the estimation method and some additional *options* for the estimation method. PROC ROBUSTREG provides four estimation methods: M estimation, LTS estimation, S estimation, and MM estimation. The default method is M estimation.

NOTE: Because the LTS and S methods use subsampling algorithms, these methods are not suitable in an analysis that uses variables that have only a few unequal values or a few unequal values within one BY group. For example, indicator variables that correspond to a classification variable often fall into this category. The same issue also applies to the initial LTS and S estimates in the MM method. For a model that includes classification independent variables or continuous independent variables with a few unequal values, the M method is recommended.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 200. The default length is 20 characters.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

OUTEST=*SAS-data-set*

specifies an output SAS data set that contains the parameter estimates and, if the COVOUT option is specified, the estimated covariance matrix. For a detailed description of the contents of the OUTEST= data set, see the section "OUTEST= Data Set" on page 9009.

PLOT | PLOTS < (*global-plot-options*) > < =*plot-request* >

PLOT | PLOTS < (*global-plot-options*) > < =(*plot-request* < . . . *plot-request* >) >

specifies options that control details of the plots. If ODS Graphics is enabled but you do not specify the PLOTS= option, then PROC ROBUSTREG produces the robust fit plot by default when the model includes a single continuous independent variable.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc robustreg data=stack plots=all;
   model y = x1 x2 x3;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The *global-plot-options* apply to all plots that are generated by the ROBUSTREG procedure. The following *global-plot-option* is available:

ONLY

suppresses the default robust fit plot. Only plots that are specifically requested are displayed.

You can specify more than one *plot-request* within the parentheses after PLOTS=. For a single *plot-request*, you can omit the parentheses. The following *plot-requests* are available:

ALL

creates all appropriate plots.

DDPLOT < (**LABEL=ALL | LEVERAGE | NONE | OUTLIER**) >

creates a plot of robust distance against Mahalanobis distance. For more information about robust distance, see the section "Leverage-Point and Outlier Detection" on page 9006. The LABEL= option specifies how the points in this plot are to be labeled, as summarized in Table 107.2.

Table 107.2 Options for Label

Value of LABEL=	Label Method
ALL	Label all points
LEVERAGE	Label leverage points
NONE	No labels
OUTLIERS	Label outliers

By default, the ROBUSTREG procedure labels both outliers and leverage points.

If you specify ID variables in the ID statement, the values of the first ID variable are used as labels; otherwise, observation numbers are used as labels.

FITPLOT < (NOLIMITS) >
creates a plot of robust fit against the single independent continuous variable that is specified in the model. You can request this plot when only a single independent continuous variable is specified in the model. Confidence limits are added to the plot by default. The NOLIMITS option suppresses these limits.

HISTOGRAM
creates a histogram for the standardized robust residuals. The histogram is superimposed with a normal density curve and a kernel density curve.

NONE
suppresses all plots.

QQPLOT
creates the normal quantile-quantile plot for the standardized robust residuals.

RDPLOT < (LABEL=ALL | LEVERAGE | NONE | OUTLIER) >
creates the plot of standardized robust residual against robust distance. For more information about robust distance, see the section "Leverage-Point and Outlier Detection" on page 9006. The LABEL= option specifies a label method for points in this plot. These label methods are described in Table 107.2.

If you specify ID variables in the ID statement, the values of the first ID variable are used as labels; otherwise, observation numbers are used as labels.

SEED=*number*
specifies the seed for the random number generator that is used to randomly select the subgroups and subsets for LTS and S estimation. By default, or if you specify 0, the ROBUSTREG procedure generates a random seed. In addition to both LTS and S estimation, if you initialize MM estimation by using either the LTS estimate or the S estimate, you might observe random variation in the results if you do not specify a fixed random number seed.

Options with METHOD=M

When you specify METHOD=M < (*options*) >, you can specify the following *options*:

ASYMPCOV=H1 | H2 | H3
specifies the type of asymptotic covariance that is computed for the M estimate. The three types are described in the section "Asymptotic Covariance and Confidence Intervals" on page 8989. By default, ASYMPCOV=H1.

CONVERGENCE=*criterion* **< (EPS=***value***) >**
specifies a convergence criterion for the M estimate. Table 107.3 lists the three criteria that are available.

Table 107.3 Options to Specify Convergence Criteria

Type	Option
Coefficient	**CONVERGENCE=COEF**
Residual	**CONVERGENCE=RESID**
Weight	**CONVERGENCE=WEIGHT**

By default, CONVERGENCE=COEF. You can specify the precision of the convergence criterion by using the EPS= option; by default, EPS=1E–8.

MAXITER=*n*

sets the maximum number of iterations during the parameter estimation. By default, MAXITER=1000.

SCALE=*scale-type* | *value*

specifies the scale parameter or a method of estimating the scale parameter. These methods and options are summarized in Table 107.4.

Table 107.4 Options to Specify Scale

Scale	Option	Default *d*
Fixed constant	**SCALE=***value*	
Huber estimate	**SCALE=HUBER** <(D=*d*)>	2.5
Median estimate	**SCALE=MED**	
Tukey estimate	**SCALE=TUKEY** <(D=*d*)>	2.5

By default, SCALE=MED.

WEIGHTFUNCTION=*function-type*

WF=*function-type*

specifies the weight function that is used for the M estimate. The ROBUSTREG procedure provides 10 weight functions, which are listed in Table 107.5. You can specify the parameters in these functions by using the A=, B=, and C= options. These functions are described in the section "M Estimation" on page 8985. The default weight function is bisquare.

Table 107.5 Options to Specify Weight Functions

Weight Function	Option	Default *a*, *b*, *c*
Andrews	**WF=ANDREWS** <(C=*c*)>	1.339
Bisquare	**WF=BISQUARE** <(C=*c*)>	4.685
Cauchy	**WF=CAUCHY** <(C=*c*)>	2.385
Fair	**WF=FAIR** <(C=*c*)>	1.4
Hampel	**WF=HAMPEL** <(<A=*a*><B=*b*><C=*c*>)>	2, 4, 8
Huber	**WF=HUBER** <(C=*c*)>	1.345
Logistic	**WF=LOGISTIC** <(C=*c*)>	1.205
Median	**WF=MEDIAN** <(C=*c*)>	0.01
Talworth	**WF=TALWORTH** <(C=*c*)>	2.795
Welsch	**WF=WELSCH** <(C=*c*)>	2.985

Options with METHOD=LTS

When you specify METHOD=LTS <(*options*)>, you can specify the following *options*:

CSTEP=*n*
specifies the number of concentration steps (C-steps) for the LTS estimate. For information about how the default value is determined, see the section "LTS Estimate" on page 8991.

H=*n*
specifies the quantile for the LTS estimate. For information about how the default value is determined, see the section "LTS Estimate" on page 8991

IADJUST=ALL | NONE
requests (IADJUST=ALL) or suppresses (IADJUST=NONE) the intercept adjustment for all estimates in the LTS algorithm. By default, the intercept adjustment is used for data sets that contain fewer than 10,000 observations. For more information, see the section "Algorithm" on page 8992.

NBEST=*n*
specifies the number of best solutions that are kept for each subgroup during the computation of the LTS estimate. The default number is 10, which is the maximum number allowed.

NREP=*n*
specifies the number of times to repeat least squares fit in subgroups during the computation of the LTS estimate. For information about how the default number is determined, see the section "LTS Estimate" on page 8991.

SUBANALYSIS
requests a display of the subgrouping information and parameter estimates within subgroups. This option generates the ODS tables that are listed in Table 107.6.

Table 107.6 ODS Tables Available with SUBANALYSIS Option

ODS Table Name	Description
BestEstimates	Best final estimates for LTS
BestSubEstimates	Best estimates for each subgroup
CStep	C-step information for LTS
Groups	Grouping information for LTS

SUBGROUPSIZE=*n*
specifies the data set size of the subgroups in the computation of the LTS estimate. The default number is 300.

Options with METHOD=S

When you specify METHOD=S <(*options*)>, you can specify the following *options*:

ASYMPCOV=H1 | H2 | H3 | H4
specifies the type of asymptotic covariance that is computed for the S estimate. The four types are described in the section "Asymptotic Covariance and Confidence Intervals" on page 8996. By default, ASYMPCOV=H4.

CHIF= TUKEY | YOHAI
specifies the χ function for the S estimate. PROC ROBUSTREG provides two χ functions, Tukey's bisquare function and Yohai's optimal function, which you can request by specifying CHIF=TUKEY and CHIF=YOHAI, respectively. The default is Tukey's bisquare function.

EFF=*value*
specifies the efficiency (as a fraction) of the S estimate. The parameter k_0 in the χ function is determined by this efficiency. The default efficiency is determined such that the consistent S estimate has a breakdown value of 25%. This option is overwritten by the K0= option if both options are used.

K0=*value*
specifies the k_0 parameter in the χ function of the S estimate. If you specify CHIF=TUKEY, the default is 1.548. If you specify CHIF=YOHAI, the default is 0.66. These default values correspond to a 50% breakdown value of the consistent S estimate.

MAXITER=*n*
sets the maximum number of iterations for computing the scale parameter of the S estimate. By default, MAXITER=1000.

NREP=*n*
specifies the number of repeats of subsampling in the computation of the S estimate. For information about how the default number of repeats is determined, see the section "Algorithm" on page 8994.

NOREFINE
suppresses the refinement of the S estimate. For more information, see the section "Algorithm" on page 8994.

SUBSETSIZE=*n*
specifies the size of the subset for the S estimate. For information about how the default value is determined, see the section "Algorithm" on page 8994.

TOLERANCE=*value*
specifies the tolerance for the S estimate of the scale. The default value is 0.001.

Options with METHOD=MM

When you specify METHOD=MM <(*options*)>, you can specify the following *options*:

ASYMPCOV=H1 | H2 | H3 | H4

specifies the type of asymptotic covariance that is computed for the MM estimate. The four types are described in the section "Details: ROBUSTREG Procedure" on page 8984. By default, ASYMPCOV=H4.

BIASTEST < (ALPHA=*number*) >

requests the bias test for the final MM estimate. For more information about this test, see the section "Bias Test" on page 8998.

CHIF=TUKEY | YOHAI

selects the χ function for the MM estimate. PROC ROBUSTREG provides two χ functions, Tukey's bisquare function and Yohai's optimal function, which you can request by specifying CHIF=TUKEY and CHIF=YOHAI, respectively. The default is Tukey's bisquare function. This χ function is also used by the initial S estimate if you specify the INITEST=S option.

CONVERGENCE=*criterion* < (EPS=*number*) >

specifies a convergence criterion for the MM estimate. Table 107.7 lists the three criteria that are available.

Table 107.7 Options to Specify Convergence Criteria

Type	Option
Coefficient	**CONVERGENCE=COEF**
Residual	**CONVERGENCE=RESID**
Weight	**CONVERGENCE=WEIGHT**

By default, CONVERGENCE=COEF. You can specify the precision of the convergence criterion by using the EPS= option; by default, EPS=1E–8.

EFF=*value*

specifies the efficiency (as a fraction) of the MM estimate. The parameter k_1 in the χ function is determined by this efficiency. The default efficiency is set to 0.85, which corresponds to $k_1 = 3.440$ if you specify CHIF=TUKEY or $k_1 = 0.868$ if you specify CHIF=YOHAI.

INITEST=LTS | S

specifies the initial estimator for the MM estimator. By default, the LTS estimator with its default settings is used as the initial estimator for the MM estimator.

INITH=*h*

specifies the integer *h* for the initial LTS estimate that is used by the MM estimator. For information about how to specify *h* and how the default is determined, see the section "Algorithm" on page 8998.

K0=*number*

specifies the parameter k_0 in the χ function for the MM estimate. If you specify CHIF=TUKEY, the default is $k_0 = 2.9366$. If you specify CHIF=YOHAI, the default is $k_0 = 0.7405$. These default values correspond to the 25% breakdown value of the MM estimator.

MAXITER=*n*
sets the maximum number of iterations during the parameter estimation. By default, MAXITER=1000.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC ROBUSTREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the ROBUSTREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* < / **TRUNCATE** > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC ROBUSTREG statement.

You can specify the following *option* in the CLASS statement after a slash (/):

TRUNCATE
specifies that class levels should be determined by using only up to the first 16 characters of the formatted values of CLASS variables.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 107.8 summarizes the *options* available in the EFFECT statement.

Table 107.8 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.

Table 107.8 *continued*

Option	Description
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

ID Statement

ID *variables* **;**

When you request the diagnostics table by specifying the DIAGNOSTICS option in the MODEL statement, the variables that are listed in the ID statement are displayed in addition to the observation number. You can use these variables to identify each observation. If the ID statement is omitted, the observation number is used to identify the observations.

MODEL Statement

< *label*: > **MODEL** *response* **=** < *effects* > < / *options* > **;**

You can specify main effects and interaction terms in the MODEL statement, as in the GLM procedure (see Chapter 53, "The GLM Procedure").

The optional *label*, which must be a valid SAS name, is used to label the model in the OUTEST data set.

Table 107.9 summarizes the *options* available in the MODEL statement.

Table 107.9 MODEL Statement Options

Option	Description
ALPHA=	Specifies the significance level
CORRB	Produces the estimated correlation matrix
COVB	Produces the estimated covariance matrix
CUTOFF=	Specifies the multiplier of the cutoff value for outlier detection
DIAGNOSTICS	Requests the outlier diagnostics
FAILRATIO=	Specifies the failure-ratio threshold
ITPRINT	Displays the iteration history
LEVERAGE	Requests an analysis of leverage points
NOGOODFIT	Suppresses the computation of goodness-of-fit statistics
NOINT	Specifies no-intercept regression
SINGULAR=	Specifies the tolerance for testing singularity

You can specify the following *options* for the model fit.

ALPHA=*value*
specifies the significance level for the confidence intervals for regression parameters. The *value* must be between 0 and 1. By default, ALPHA=0.05.

CORRB
produces the estimated correlation matrix of the parameter estimates.

COVB
produces the estimated covariance matrix of the parameter estimates.

CUTOFF=*value*
specifies the multiplier of the cutoff value for outlier detection. By default, CUTOFF=3.

DIAGNOSTICS < (ALL) >
requests the outlier diagnostics. By default, only observations that are identified as outliers or leverage points are displayed. Specify the ALL option to display all observations.

FAILRATIO=*value*

specifies the failure-ratio threshold for the subsampling algorithm of an LTS or S estimate. It also applies to the initial LTS or S step in an MM estimate. The threshold must be between 0 and 1. Its default value is 0.99. For more information, see the section "LTS Estimate" on page 8991 or "S Estimate" on page 8994.

ITPRINT

displays the iteration history of the iteratively reweighted least squares algorithm that is used by M and MM estimation. You can also use this option in the PROC ROBUSTREG statement.

LEVERAGE < (*leverage-options***) >**

requests an analysis of leverage points for the covariates. The results are added to the diagnostics table, which you can request by specifying the DIAGNOSTICS option in the MODEL statement.

You can use the following *leverage-options*:

CUTOFF=*value*

specifies the leverage cutoff value for leverage-point detection. For more information, see the section "Leverage-Point and Outlier Detection" on page 9006. You can also specify the cutoff value by using the CUTOFFALPHA= option.

CUTOFFALPHA=*alpha-value*

specifies the leverage cutoff α value for leverage-point detection. The respective leverage cutoff value equals $\sqrt{\chi^2_{p;1-\alpha}}$ (or $\sqrt{\chi^2_{q;1-\alpha}}$ if projection is applied in the generalized MCD algorithm). By default, $\alpha = 0.025$.

H=*n*

QUANTILE=*n*

specifies the quantile to be minimized for the MCD algorithm that is used for the leverage-point analysis. By default, H=$[(3n + p + 1)/4]$, where n is the number of observations and p is the number of independent variables, excluding the intercept.

MCDALPHA=*alpha-value*

specifies the MCD cutoff α value for the final MCD reweighting step. The respective MCD cutoff value equals $\sqrt{\chi^2_{p;1-\alpha}}$ (or $\sqrt{\chi^2_{q;1-\alpha}}$ if projection is applied in the generalized MCD algorithm). By default, $\alpha = 0.025$.

MCDCUTOFF=*value*

MCDCUT=*value*

specifies the MCD cutoff value for the final MCD reweighting step. For more information, see the section "Mahalanobis Distance versus Robust Distance" on page 9000 and Rousseeuw and Van Driessen (1999). You can also specify the cutoff value by using the MCDALPHA= option.

MCDINFO

requests that detailed information about the MCD covariance estimate be displayed, including the low-dimensional structure, the breakdown value, the MCD center, and the MCD covariance itself. The option outputs the ODS tables of the MCD profile, MCD center, MCD covariance, and MCD correlation.

OPC | OFFPLANECOEF
requests the ODS table of the coefficients for MCD-dropped components, when projection is applied in the generalized MCD algorithm. The OFFPLANECOEF option is ignored for the regular MCD algorithm.

PROJECTIONALPHA=*alpha-value*
PALPHA=*alpha-value*
specifies the projection cutoff α value to be used to judge whether an observation is on or off the low-dimensional hyperplane that is identified by the generalized MCD algorithm. The respective projection cutoff value equals $\sqrt{\chi^2_{1;1-\alpha}}$. By default, $\alpha = 0.001$.

PROJECTIONCUTOFF=*value*
PCUTOFF=*value*
specifies the projection cutoff value to be used to judge whether an observation is on or off the low-dimensional hyperplane identified by the projected MCD algorithm. For more information, see the section "Mahalanobis Distance versus Robust Distance" on page 9000 and Rousseeuw and Van Driessen (1999). You can also specify the projection cutoff value by using the PALPHA= option.

PROJECTIONTOLERANCE=*value*
PTOL=*value*
specifies the projection tolerance value for the low-dimensional structure detection. For more information, see the section "Leverage-Point and Outlier Detection" on page 9006.

NOGOODFIT
suppresses the computation of goodness-of-fit statistics.

NOINT
specifies no-intercept regression.

SINGULAR=*value*
specifies the tolerance for testing singularity of the information matrix and the crossproducts matrix for the initial least squares estimates. By default, SINGULAR=1E–12.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > *keyword*=*name* < . . . *keyword*=*name* > **;**

The OUTPUT statement creates an output SAS data set that contains statistics calculated after the model is fitted. At least one specification of the form *keyword*=*name* is required.

All variables in the original data set are included in the new data set, along with the variables that are created by using *keyword*= options in the OUTPUT statement. These new variables contain fitted values and estimated quantiles. To create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

You can use the following specifications in the OUTPUT statement:

OUT=*SAS-data-set* specifies the new data set. By default, the procedure uses the DATA*n* convention to name the new data set.

*keyword***=***name* specifies the statistics to include in the output data set and gives names to the new variables. Specify a *keyword* for each desired statistic (see the following list), an equal sign, and the variable to contain the statistic.

The *keywords* that are allowed and the statistics that they represent are as follows:

LEVERAGE specifies a variable to indicate leverage points. To include this variable in the OUT= data set, you must specify the LEVERAGE option in the MODEL statement. For information about how to define the LEVERAGE keyword, see the section "Leverage-Point and Outlier Detection" on page 9006.

MD specifies a variable to contain the Mahalanobis distances. For the definition of Mahalanobis distance, see the section "Robust Distance" on page 9000.

OUTLIER specifies a variable to indicate outliers. For information about how to define the OUTLIER keyword, see the section "Leverage-Point and Outlier Detection" on page 9006.

PMD specifies a variable to contain the projected Mahalanobis distances. For the definition of projected Mahalanobis distance, see the section "Robust Distance" on page 9000.

POD specifies a variable to contain the projected off-plane distances. For the definition of off-plane distance, see the section "Robust Distance" on page 9000.

PRD specifies a variable to contain the projected robust MCD Mahalanobis distances. For the definition of projected robust distance, see the section "Robust Distance" on page 9000.

PREDICTED | P specifies a variable to contain the estimated responses

$$\hat{y}_i = \mathbf{x}_i' \hat{\boldsymbol{\theta}}$$

RD specifies a variable to contain the robust MCD Mahalanobis distances. For the definition of robust distance, see the section "Robust Distance" on page 9000.

RESIDUAL | R specifies a variable to contain the unstandardized residuals

$$y_i - \hat{y}_i \text{ or } y_i - \mathbf{x}_i' \hat{\boldsymbol{\theta}}$$

SRESIDUAL | SR specifies a variable to contain the standardized residuals

$$\frac{y_i - \hat{y}_i}{\hat{\sigma}} \text{ or } \frac{y_i - \mathbf{x}_i' \hat{\boldsymbol{\theta}}}{\hat{\sigma}}.$$

By default, the LTS method uses Wscale as $\hat{\sigma}$ for computing the standardized residuals.

STDP specifies a variable to contain the estimates of the standard errors of the estimated mean responses

$$\sqrt{\mathbf{x}_i' \boldsymbol{\Sigma} \mathbf{x}_i}$$

where $\boldsymbol{\Sigma}$ denotes the covariance matrix of the parameter estimates. You can request the ODS table of this covariance matrix by specifying the COVB option in the MODEL statement. The STDP= option is applied to M, S, and MM estimation, but not to LTS estimation.

STDI specifies a variable to contain the estimates of the standard errors of the individual predicted values

$$\sqrt{\mathbf{x}_i' \boldsymbol{\Sigma} \mathbf{x}_i + \hat{\sigma}^2}.$$

The STDI= option is applied to M, S, and MM estimation, but not to LTS estimation.

WEIGHT specifies a variable to contain the computed final weights.

PERFORMANCE Statement

The PERFORMANCE statement is used to specify options that affect the performance of PROC ROBUSTREG and to request tables that show the performance options in effect and timing details. See Chen (2002) for some empirical results.

PERFORMANCE < *options* > **;**

You can specify the following *options*:

CPUCOUNT=*n* | **ACTUAL**
specifies the number of processors to use in forming crossproduct matrices. You can specify any integer in the range 1–1024 for *n*. CPUCOUNT=ACTUAL sets CPUCOUNT to the number of physical processors available. This can be less than the number of physical CPUs if the SAS process has been restricted by system administration tools. Setting CPUCOUNT= to a number greater than the actual number of available CPUs might result in reduced performance. This option overrides the SAS system option CPUCOUNT=. If CPUCOUNT=1, then NOTHREADS is in effect, and PROC ROBUSTREG uses singly threaded code.

DETAILS
requests the PerfSettings table that shows the performance settings in effect and the "Timing" table that provides a broad timing breakdown of the PROC ROBUSTREG step.

NOTHREADS
disables multithreaded computation. This option overrides the SAS system option THREADS.

THREADS
enables multithreaded computation. This option overrides the SAS system option NOTHREADS.

TEST Statement

< *label*: > TEST *effects* ;

When you use M estimation and MM estimation, the TEST statement provides a means of obtaining a test for the canonical linear hypothesis about the parameters of the tested effects

$$\theta_j = 0, \quad j = i_1, \ldots, i_q$$

where q is the total number of parameters of the tested effects.

PROC ROBUSTREG provides two kinds of robust tests: the ρ test and the R_n^2 test. They are described in the section "Details: ROBUSTREG Procedure" on page 8984. No test is available for LTS and S estimation.

The optional *label*, which must be a valid SAS name, is used to label output from the corresponding TEST statement.

WEIGHT Statement

WEIGHT *variable* ;

The WEIGHT statement specifies a weight variable in the input data set.

If you want to use fixed weights for each observation in the input data set, place the weights in a variable in the data set and specify the name in a WEIGHT statement. The values of the WEIGHT variable can be nonintegral and are not truncated. Observations that have nonpositive or missing values for the weight variable do not contribute to the fit of the model.

Details: ROBUSTREG Procedure

This section describes the statistical and computational aspects of the ROBUSTREG procedure. The following notation is used throughout the section.

Let $\mathbf{X} = (x_{ij})$ denote an $n \times p$ matrix, let $\mathbf{y} = (y_1, \ldots, y_n)'$ denote a given n-vector of responses, and let $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)'$ denote an unknown p-vector of parameters or coefficients whose components are to be estimated. The matrix $\mathbf{X}$ is called the design matrix. Consider the usual linear model,

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} + \mathbf{e}$$

where $\mathbf{e} = (e_1, \ldots, e_n)'$ is an n-vector of unknown errors. It is assumed that (for a given $\mathbf{X}$) the components e_i of $\mathbf{e}$ are independent and identically distributed according to a distribution $L(\cdot/\sigma)$, where σ is a scale parameter (usually unknown). Often $L(\cdot) \approx \Phi(\cdot)$, the standard normal distribution function. The vector of residuals for a given value of $\hat{\boldsymbol{\theta}}$ is denoted by $\mathbf{r} = (r_1, \ldots, r_n)'$, and the ith row of the matrix $\mathbf{X}$ is denoted by $\mathbf{x}_i'$.

M Estimation

M estimation in the context of regression was first introduced by Huber (1973) as a result of making the least squares approach robust. Although M estimators are not robust with respect to leverage points, they are popular in applications where leverage points are not an issue.

Instead of minimizing a sum of squares of the residuals, a Huber-type M estimator $\hat{\boldsymbol{\theta}}_M$ of $\boldsymbol{\theta}$ minimizes a sum of less rapidly increasing functions of the residuals:

$$Q(\boldsymbol{\theta}) = \sum_{i=1}^{n} \rho\left(\frac{r_i}{\sigma}\right)$$

where $\mathbf{r} = \mathbf{y} - \mathbf{X}\boldsymbol{\theta}$. For the ordinary least squares estimation, ρ is the square function, $\rho(z) = z^2$.

If σ is known, then when derivatives are taken with respect to $\boldsymbol{\theta}$, $\hat{\boldsymbol{\theta}}_M$ is also a solution of the system of p equations:

$$\sum_{i=1}^{n} \psi\left(\frac{r_i}{\sigma}\right) x_{ij} = 0,\ j = 1, \ldots, p$$

where $\psi = \frac{\partial \rho}{\partial z}$. If ρ is convex, $\hat{\boldsymbol{\theta}}_M$ is the unique solution.

The ROBUSTREG procedure solves this system by using iteratively reweighted least squares (IRLS). The weight function $w(x)$ is defined as

$$w(z) = \frac{\psi(z)}{z}$$

The ROBUSTREG procedure provides 10 kinds of weight functions through the WEIGHTFUNCTION= option in the MODEL statement. Each weight function corresponds to a ρ function. For a complete discussion, see the section "Weight Functions" on page 8987. You can specify the scale parameter σ by using the SCALE= option in the PROC ROBUSTREG statement.

If σ is unknown, both $\boldsymbol{\theta}$ and σ are estimated by minimizing the function

$$Q(\boldsymbol{\theta}, \sigma) = \sum_{i=1}^{n} \left[\rho\left(\frac{r_i}{\sigma}\right) + a\right]\sigma,\ a > 0$$

The algorithm proceeds by alternately improving $\hat{\boldsymbol{\theta}}$ in a location step and $\hat{\sigma}$ in a scale step.

For the scale step, the following three methods are available to estimate σ, which you can select by specifying the SCALE= option:

1. (SCALE=HUBER <(D=*d*)>) Compute $\hat{\sigma}$ by the iteration

$$\left(\hat{\sigma}^{(m+1)}\right)^2 = \frac{1}{nh}\sum_{i=1}^{n}\chi_d\left(\frac{r_i}{\hat{\sigma}^{(m)}}\right)\left(\hat{\sigma}^{(m)}\right)^2$$

where

$$\chi_d(x) = \begin{cases} x^2/2 & \text{if } |x| < d \\ d^2/2 & \text{otherwise} \end{cases}$$

is the Huber function and $h = \frac{n-p}{n}\left(d^2 + (1-d^2)\Phi(d) - 0.5 - \frac{d}{\sqrt{2\pi}}e^{-\frac{1}{2}d^2}\right)$ is the Huber constant (Huber 1981, p. 179). You can use the D=*d* option to specify *d*. By default, D=2.5.

2. (SCALE=TUKEY <(D=*d*)>) Compute $\hat{\sigma}$ by the iteration

$$\left(\hat{\sigma}^{(m+1)}\right)^2 = \frac{1}{(n-p)\beta}\sum_{i=1}^{n}\chi_d\left(\frac{r_i}{\hat{\sigma}^{m}}\right)\left(\hat{\sigma}^{(m)}\right)^2$$

where

$$\chi_d(x) = \begin{cases} \frac{3x^2}{d^2} - \frac{3x^4}{d^4} + \frac{x^6}{d^6} & \text{if } |x| < d \\ 1 & \text{otherwise} \end{cases}$$

is the Tukey bisquare function and $\beta = \int \chi_d(s)d\Phi(s)$ is the constant such that the solution $\hat{\sigma}$ is asymptotically consistent when $L(\cdot/\sigma) = \Phi(\cdot)$ (Hampel et al. 1986, p. 149). You can use the D=*d* option to specify *d*. By default, D=2.5.

3. (SCALE=MED) Compute $\hat{\sigma}$ by the iteration

$$\hat{\sigma}^{(m+1)} = \text{median}\left\{|y_i - \mathbf{x}_i'\hat{\boldsymbol{\theta}}^{(m)}|/\beta_0, i = 1, \ldots, n\right\}$$

where $\beta_0 = \Phi^{-1}(0.75)$ is the constant such that the solution $\hat{\sigma}$ is asymptotically consistent when $L(\cdot/\sigma) = \Phi(\cdot)$ (Hampel et al. 1986, p. 312).

SCALE=MED is the default.

Algorithm

The basic algorithm for computing M estimates for regression is iteratively reweighted least squares (IRLS). As the name suggests, a weighted least squares fit is carried out inside an iteration loop. For each iteration, a set of weights for the observations is used in the least squares fit. The weights are constructed by applying a weight function to the current residuals. Initial weights are based on residuals from an initial fit. The ROBUSTREG procedure uses the unweighted least squares fit as a default initial fit. The iteration terminates when a convergence criterion is satisfied. The maximum number of iterations is set to 1,000. You can specify both the weight function and the convergence criteria.

Weight Functions

You can specify the weight function for M estimation by using the WEIGHTFUNCTION= option. The ROBUSTREG procedure provides 10 weight functions. By default, the procedure uses the bisquare weight function. In most cases, M estimates are more sensitive to the parameters of these weight functions than to the type of weight function. The median weight function is not stable and is seldom recommended in data analysis; it is included in the ROBUSTREG procedure for completeness. You can specify the parameters for these weight functions. Except for the Hampel and median weight functions, default values for these parameters are defined such that the corresponding M estimates have 95% asymptotic efficiency in the location model with the Gaussian distribution (Holland and Welsch 1977).

The following list shows the 10 weight functions available. See Table 107.5 for the default values of the constants in these weight functions.

Andrews $$W(x,c) = \begin{cases} \frac{\sin(x/c)}{x/c} & \text{if } |x| \le \pi c \\ 0 & \text{otherwise} \end{cases}$$

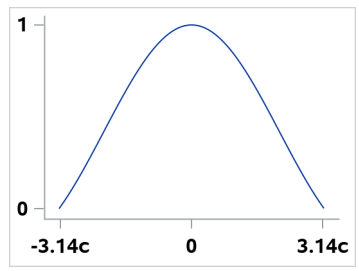

Bisquare $$W(x,c) = \begin{cases} \left(1-(x/c)^2\right)^2 & \text{if } |x| < c \\ 0 & \text{otherwise} \end{cases}$$

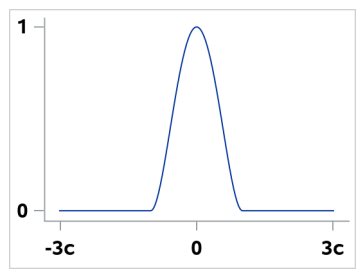

Cauchy $$W(x,c) = \frac{1}{1+(|x|/c)^2}$$

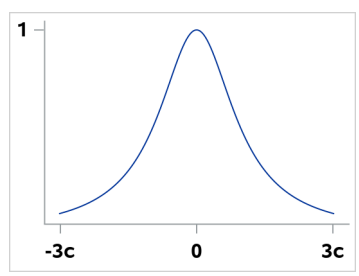

Fair $$W(x,c) = \frac{1}{(1+|x|/c)}$$

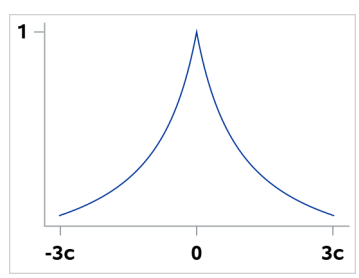

Hampel $$W(x,a,b,c) = \begin{cases} 1 & |x| < a \\ \frac{a}{|x|} & a < |x| \le b \\ \frac{a}{|x|}\frac{c-|x|}{c-b} & b < |x| \le c \\ 0 & \text{otherwise} \end{cases}$$

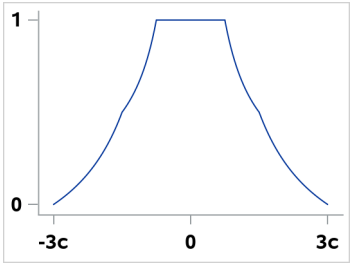

Huber $$W(x,c) = \begin{cases} 1 & \text{if } |x| < c \\ \frac{c}{|x|} & \text{otherwise} \end{cases}$$

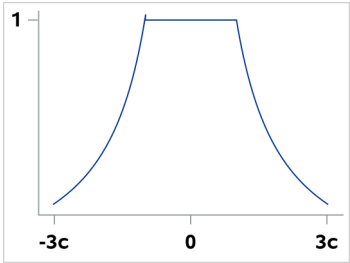

Logistic $$W(x,c) = \frac{\tanh(x/c)}{x/c}$$

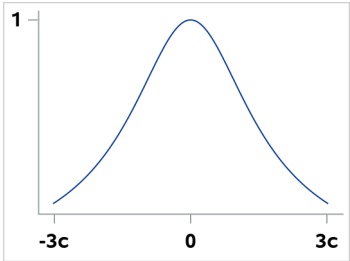

Median $$W(x,c) = \begin{cases} \frac{1}{c} & \text{if } x = 0 \\ \frac{1}{|x|} & \text{otherwise} \end{cases}$$

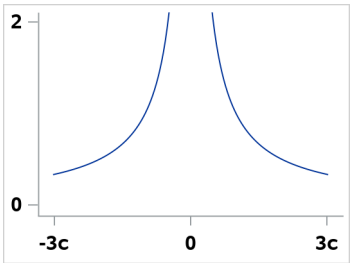

Talworth $$W(x,c) = \begin{cases} 1 & \text{if } |x| < c \\ 0 & \text{otherwise} \end{cases}$$

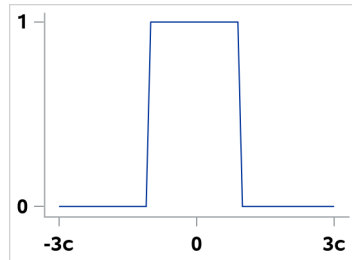

Welsch $$W(x,c) = \exp\left(-\frac{(x/c)^2}{2}\right)$$

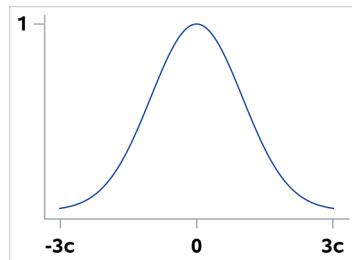

Convergence Criteria

The following convergence criteria are available in PROC ROBUSTREG:

- relative change in the coefficients (CONVERGENCE=COEF)
- relative change in the scaled residuals (CONVERGENCE=RESID)
- relative change in weights (CONVERGENCE=WEIGHT)

You can specify the criteria by using the CONVERGENCE= option in the PROC ROBUSTREG statement. The default is CONVERGENCE=COEF.

You can specify the precision of the convergence criterion by using the EPS= suboption. The default is EPS=1E–8.

In addition to these convergence criteria, a convergence criterion that is based on a scale-independent measure of the gradient is always checked. For more information, see Coleman et al. (1980). A warning is issued if this additional criterion is not satisfied.

Asymptotic Covariance and Confidence Intervals

The following three estimators of the asymptotic covariance of the robust estimator are available in PROC ROBUSTREG:

H1: $$K^2 \frac{[1/(n-p)]\sum(\psi(r_i))^2}{[(1/n)\sum(\psi'(r_i))]^2}(\mathbf{X}'\mathbf{X})^{-1}$$

H2: $$K \frac{[1/(n-p)]\sum(\psi(r_i))^2}{[(1/n)\sum(\psi'(r_i))]}\mathbf{W}^{-1}$$

H3: $$K^{-1}\frac{1}{(n-p)}\sum(\psi(r_i))^2\mathbf{W}^{-1}(\mathbf{X}'\mathbf{X})\mathbf{W}^{-1}$$

where $K = 1 + \frac{p}{n}\frac{\mathrm{Var}(\psi')}{(E\psi')^2}$ is a correction factor and $W_{jk} = \sum \psi'(r_i)x_{ij}x_{ik}$. For more information, see Huber (1981, p. 173).

You can specify the asymptotic covariance estimate by using the ASYMPCOV= option. The ROBUSTREG procedure uses H1 as the default because of its simplicity and stability. Confidence intervals are computed from the diagonal elements of the estimated asymptotic covariance matrix.

R Square and Deviance

The robust version of R square is defined as

$$R^2 = \frac{\sum \rho\left(\frac{y_i - \hat{\mu}}{\hat{s}}\right) - \sum \rho\left(\frac{y_i - \mathbf{x}_i'\hat{\boldsymbol{\theta}}}{\hat{s}}\right)}{\sum \rho\left(\frac{y_i - \hat{\mu}}{\hat{s}}\right)}$$

The robust deviance is defined as the optimal value of the objective function on the σ^2 scale,

$$D = 2\hat{s}^2 \sum \rho\left(\frac{y_i - \mathbf{x}_i'\hat{\boldsymbol{\theta}}}{\hat{s}}\right)$$

where $\rho' = \psi$, $\hat{\boldsymbol{\theta}}$ is the M estimator of $\boldsymbol{\theta}$, $\hat{\mu}$ is the M estimator of location, and $\hat{s}$ is the M estimator of the scale parameter in the full model.

Linear Tests

Two tests are available in PROC ROBUSTREG for the canonical linear hypothesis

$$H_0: \quad \theta_j = 0, \quad j = i_1, \ldots, i_q$$

where q is the total number of parameters of the tested effects. The first test is a robust version of the F test, which is referred to as the ρ test. Denote the M estimators in the full and reduced models as $\hat{\boldsymbol{\theta}}(0) \in \Omega_0$ and $\hat{\boldsymbol{\theta}}(1) \in \Omega_1$, respectively. Let

$$\begin{aligned} Q_0 &= Q(\hat{\boldsymbol{\theta}}(0)) = \min\{Q(\boldsymbol{\theta}) | \boldsymbol{\theta} \in \Omega_0\} \\ Q_1 &= Q(\hat{\boldsymbol{\theta}}(1)) = \min\{Q(\boldsymbol{\theta}) | \boldsymbol{\theta} \in \Omega_1\} \end{aligned}$$

with

$$Q = \sum_{i=1}^{n} \rho\left(\frac{r_i}{\sigma}\right)$$

The robust F test is based on the test statistic

$$S_n^2 = \frac{2}{q}[Q_1 - Q_0]$$

Asymptotically $S_n^2 \sim \lambda\chi_q^2$ under H_0, where the standardization factor is $\lambda = \int \psi^2(s)d\Phi(s) / \int \psi'(s)\, d\Phi(s)$ and Φ is the cumulative distribution function of the standard normal distribution. Large values of S_n^2 are significant. This test is a special case of the general τ test of Hampel et al. (1986, Section 7.2).

The second test is a robust version of the Wald test, which is referred to as the R_n^2 test. This test uses a test statistic

$$R_n^2 = n(\hat{\theta}_{i_1}, \ldots, \hat{\theta}_{i_q})\mathbf{H}_{22}^{-1}(\hat{\theta}_{i_1}, \ldots, \hat{\theta}_{i_q})'$$

where $\frac{1}{n}\mathbf{H}_{22}$ is the $q \times q$ block (corresponding to $\theta_{i_1}, \ldots, \theta_{i_q}$) of the asymptotic covariance matrix of the M estimate $\hat{\boldsymbol{\theta}}_M$ of $\boldsymbol{\theta}$ in a p-parameter linear model.

Under H_0, the statistic R_n^2 has an asymptotic χ^2 distribution with q degrees of freedom. Large values of R_n^2 are significant. For more information, see Hampel et al. (1986, Chapter 7).

Model Selection

When M estimation is used, two criteria are available in PROC ROBUSTREG for model selection. The first criterion is a counterpart of Akaike's (1974) information criterion for robust regression (AICR); it is defined as

$$\text{AICR} = 2\sum_{i=1}^{n} \rho(r_{i:p}) + \alpha p$$

where $r_{i:p} = (y_i - \mathbf{x}_i'\hat{\boldsymbol{\theta}})/\hat{\sigma}$, $\hat{\sigma}$ is a robust estimate of σ and $\hat{\boldsymbol{\theta}}$ is the M estimator with the p-dimensional design matrix.

As with AIC, α is the weight of the penalty for dimensions. The ROBUSTREG procedure uses $\alpha = 2E\psi^2/E\psi'$ (Ronchetti 1985) and estimates it by using the final robust residuals.

The second criterion is a robust version of the Schwarz information criteria (BICR); it is defined as

$$\text{BICR} = 2\sum_{i=1}^{n} \rho(r_{i:p}) + p\log(n)$$

High Breakdown Value Estimation

The *breakdown value* of an estimator is the smallest contamination fraction of the data that can cause the estimates on the entire data to be arbitrarily far from the estimates on only the uncontaminated data. The breakdown value of an estimator can be used to measure the robustness of the estimator. Rousseeuw and Leroy (1987) and others introduced the following high breakdown value estimators for linear regression.

LTS Estimate

The least trimmed squares (LTS) estimate that was proposed by Rousseeuw (1984) is defined as the p-vector

$$\hat{\boldsymbol{\theta}}_{LTS} = \arg\min_{\boldsymbol{\theta}} Q_{LTS}(\boldsymbol{\theta}) \text{ with } Q_{LTS}(\boldsymbol{\theta}) = \sum_{i=1}^{h} r_{(i)}^2$$

where $r_{(1)}^2 \le r_{(2)}^2 \le \ldots \le r_{(n)}^2$ are the ordered squared residuals $r_i^2 = (y_i - \mathbf{x}_i'\boldsymbol{\theta})^2$, $i = 1, \ldots, n$, and h is defined in the range $\frac{n}{2} + 1 \le h \le \frac{3n+p+1}{4}$.

You can specify the parameter h by using the H= option in the PROC ROBUSTREG statement. By default, $h = [\frac{3n+p+1}{4}]$. The breakdown value is $\frac{n-h}{n}$ for the LTS estimate.

The ROBUSTREG procedure computes LTS estimates by using the FAST-LTS algorithm of Rousseeuw and Van Driessen (2000). The estimates are often used to detect outliers in the data, which are then downweighted in the resulting weighted LS regression.

Algorithm

Least trimmed squares (LTS) regression is based on the subset of h observations (out of a total of n observations) whose least squares fit possesses the smallest sum of squared residuals. The coverage h can be set between $\frac{n}{2}$ and n. The LTS method was proposed by Rousseeuw (1984, p. 876) as a highly robust regression estimator with breakdown value $\frac{n-h}{n}$. The ROBUSTREG procedure uses the FAST-LTS algorithm that was proposed by Rousseeuw and Van Driessen (2000). The intercept adjustment technique is also used in this implementation. However, because this adjustment is expensive to compute, it is optional. You can use the IADJUST= option in the PROC ROBUSTREG statement to request or suppress the intercept adjustment. By default, PROC ROBUSTREG does intercept adjustment for data sets that contain fewer than 10,000 observations. The steps of the algorithm are described briefly as follows. For more information, see Rousseeuw and Van Driessen (2000).

1. The default h is $[\frac{3n+p+1}{4}]$, where p is the number of independent variables. You can specify any integer h with $[\frac{n}{2}] + 1 \leq h \leq [\frac{3n+p+1}{4}]$ by using the H= option in the MODEL statement. The breakdown value for LTS, $\frac{n-h}{n}$, is reported. The default h is a good compromise between breakdown value and statistical efficiency.

2. If $p = 1$ (single regressor), the procedure uses the exact algorithm of Rousseeuw and Leroy (1987, p. 172).

3. If $p \geq 2$, PROC ROBUSTREG uses the following algorithm. If $n < 2$ *ssubs*, where *ssubs* is the size of the subgroups (you can specify *ssubs* by using the SUBGROUPSIZE= option in the PROC ROBUSTREG statement; by default, *ssubs* = 300), PROC ROBUSTREG draws a random p-subset and computes the regression coefficients by using these p points (if the regression is degenerate, another p-subset is drawn). The absolute residuals for all observations in the data set are computed, and the first h points that have the smallest absolute residuals are selected. From this selected h-subset, PROC ROBUSTREG carries out *nsteps* C-steps (concentration steps; for more information, see Rousseeuw and Van Driessen 2000). You can specify *nsteps* by using the CSTEP= option in the PROC ROBUSTREG statement; by default, *nsteps* = 2. PROC ROBUSTREG redraws p-subsets and repeats the preceding computation *nrep* times, and then finds the *nbsol* (at most) solutions that have the lowest sums of h squared residuals. You can specify *nrep* by using the NREP= option in the PROC ROBUSTREG statement; by default, NREP=$\min\{500, \binom{n}{p}\}$. For small n and p, all $\binom{n}{p}$ subsets are used and the NREP= option is ignored (Rousseeuw and Hubert 1996). You can specify *nbsol* by using the NBEST= option in the PROC ROBUSTREG statement; by default, NBEST=10. For each of these *nbsol* best solutions, C-steps are taken until convergence and the best final solution is found.

4. If $n \geq 5ssubs$, construct five disjoint random subgroups with size *ssubs*. If $2ssubs < n < 5ssubs$, the data are split into at most four subgroups with *ssubs* or more observations in each subgroup, so that each observation belongs to a subgroup and the subgroups have roughly the same size. Let *nsubs* denote the number of subgroups. Inside each subgroup, PROC ROBUSTREG repeats the step 3 algorithm *nrep* / *nsubs* times, keeps the *nbsol* best solutions, and pools the subgroups, yielding the merged set of size n_{merged}. In the merged set, for each of the $nsubs \times nbsol$ best solutions, *nsteps* C-steps are carried out by using n_{merged} and $h_{merged} = [n_{merged}\frac{h}{n}]$ and the *nbsol* best solutions are

kept. In the full data set, for each of these *nbsol* best solutions, C-steps are taken by using n and h until convergence and the best final solution is found.

NOTE: At step 3 in the algorithm, a randomly selected p-subset might be degenerate (that is, its design matrix might be singular). If the total number of p-subsets from any subgroup is greater than 4,000 and the ratio of degenerate p-subsets is higher than the threshold that is specified in the FAILRATIO= option, the algorithm terminates with an error message.

R Square

For models with the intercept term, the robust version of R square for the LTS estimate is defined as

$$R^2_{LTS} = 1 - \frac{s^2_{LTS}(\mathbf{X}, \mathbf{y})}{s^2_{LTS}(\mathbf{1}, \mathbf{y})}$$

For models without the intercept term, it is defined as

$$R^2_{LTS} = 1 - \frac{s^2_{LTS}(\mathbf{X}, \mathbf{y})}{s^2_{LTS}(\mathbf{0}, \mathbf{y})}$$

For both models,

$$s_{LTS}(\mathbf{X}, \mathbf{y}) = d_{h,n} \sqrt{\frac{1}{h} \sum_{i=1}^{h} r^2_{(i)}}$$

Note that s_{LTS} is a preliminary estimate of the parameter σ in the distribution function $L(\cdot/\sigma)$.

Here $d_{h,n}$ is chosen to make s_{LTS} consistent, assuming a Gaussian model. Specifically,

$$\begin{aligned} d_{h,n} &= 1/\sqrt{1 - \frac{2n}{h c_{h,n}} \phi(1/c_{h,n})} \\ c_{h,n} &= 1/\Phi^{-1}\left(\frac{h+n}{2n}\right) \end{aligned}$$

where Φ and ϕ are the distribution function and the density function of the standard normal distribution, respectively.

Final Weighted Scale Estimator

The ROBUSTREG procedure displays two scale estimators, s_{LTS} and Wscale. The estimator Wscale is a more efficient scale estimator based on the preliminary estimate s_{LTS}; it is defined as

$$\text{Wscale} = \sqrt{\frac{\sum_i w_i {r_i}^2}{\sum_i w_i - p}}$$

where

$$w_i = \begin{cases} 0 & \text{if } |r_i|/s_{LTS} > k \\ 1 & \text{otherwise} \end{cases}$$

You can specify k by using the CUTOFF= option in the MODEL statement. By default, $k = 3$.

S Estimate

The S estimate that was proposed by Rousseeuw and Yohai (1984) is defined as the p-vector

$$\hat{\boldsymbol{\theta}}_S = \arg\min_{\theta} S(\boldsymbol{\theta})$$

where the dispersion $S(\boldsymbol{\theta})$ is the solution of

$$\frac{1}{n-p}\sum_{i=1}^{n}\chi\left(\frac{y_i - \mathbf{x}_i'\boldsymbol{\theta}}{S}\right) = \beta$$

Here β is set to $\int \chi(s)d\Phi(s)$ such that $\hat{\boldsymbol{\theta}}_S$ and $S(\hat{\boldsymbol{\theta}}_S)$ are asymptotically consistent estimates of $\boldsymbol{\theta}$ and σ for the Gaussian regression model. The breakdown value of the S estimate is

$$\frac{\beta}{\max_s \chi(s)}$$

The ROBUSTREG procedure provides two choices for χ: Tukey's bisquare function and Yohai's optimal function.

Tukey's bisquare function, which you can specify by using the option CHIF=TUKEY, is

$$\chi_{k_0}(s) = \begin{cases} 3(\frac{s}{k_0})^2 - 3(\frac{s}{k_0})^4 + (\frac{s}{k_0})^6, & \text{if } |s| \le k_0 \\ 1 & \text{otherwise} \end{cases}$$

The constant k_0 controls the breakdown value and efficiency of the S estimate. If you use the EFF= option to specify the efficiency, you can determine the corresponding k_0. The default k_0 is 2.9366, such that the breakdown value of the S estimate is 0.25, with a corresponding asymptotic efficiency for the Gaussian model of 75.9%.

The Yohai function, which you can specify by using the option CHIF=YOHAI, is

$$\chi_{k_0}(s) = \begin{cases} \frac{s^2}{2} & \text{if } |s| \le 2k_0 \\ k_0^2[b_0 + b_1(\frac{s}{k_0})^2 + b_2(\frac{s}{k_0})^4 & \\ +b_3(\frac{s}{k_0})^6 + b_4(\frac{s}{k_0})^8] & \text{if } 2k_0 < |s| \le 3k_0 \\ 3.25k_0^2 & \text{if } |s| > 3k_0 \end{cases}$$

where $b_0 = 1.792$, $b_1 = -0.972$, $b_2 = 0.432$, $b_3 = -0.052$, and $b_4 = 0.002$. If you use the EFF= option to specify the efficiency, you can determine the corresponding k_0. By default, k_0 is set to 0.7405, such that the breakdown value of the S estimate is 0.25, with a corresponding asymptotic efficiency for the Gaussian model of 72.7%.

Algorithm

The ROBUSTREG procedure implements the algorithm that was proposed by Marazzi (1993) for the S estimate, which is a refined version of the algorithm that was proposed by Ruppert (1992). The refined algorithm is briefly described as follows.

Initialize *iter* = 1.

1. Draw a random q-subset of the total n observations, and compute the regression coefficients by using these q observations (if the regression is degenerate, draw another q-subset), where $q \geq p$ can be specified by using the SUBSIZE= option. By default, $q = p$.

2. Compute the residuals: $r_i = y_i - \sum_{j=1}^{p} x_{ij}\theta_j$ for $i = 1, \ldots, n$. For the first iteration, where *iter* = 1, take the following substeps:

   a) If all $|r_i| = 0$ for $i = 1, \ldots, n$, which means y_i exactly equals $\sum_{j=1}^{p} x_{ij}\theta_j$ for all $i = 1, \ldots, n$, this algorithm terminates with a message for exact fit.

   b) Otherwise, set $s^* = 2\text{median}\{|r_i|, i = 1, \ldots, n\}$.

   c) If $s^* = 0$, update $s^* = \min\{|r_i| > 0, i = 1, \ldots, n\}$.

   d) If $\sum_{i=1}^{n} \chi(r_i/s^*) > (n-p)\beta$, set $s^* = 1.5s^*$; go to step 3.

   If *iter* > 1 and $\sum_{i=1}^{n} \chi(r_i/s^*) <= (n-p)\beta$, go to step 3; otherwise, go to step 5.

3. Solve the following equation for s by using an iterative algorithm:

$$\frac{1}{n-p}\sum_{i=1}^{n} \chi(r_i/s) = \beta$$

4. If *iter* > 1 and $s > s^*$, go to step 5. Otherwise, set $s^* = s$ and $\boldsymbol{\theta}^* = \boldsymbol{\theta}$. If $s^* <$ TOLS, return s^* and $\boldsymbol{\theta}^*$; otherwise, go to step 5.

5. If *iter* < NREP, set $iter = iter + 1$ and return to step 1; otherwise, return s^* and $\boldsymbol{\theta}^*$.

The ROBUSTREG procedure performs the following refinement step by default. You can request that this refinement not be performed by specifying the NOREFINE option in the PROC ROBUSTREG statement.

6. Let $\psi = \chi'$. Using the values s^* and $\boldsymbol{\theta}^*$ from the previous steps, compute the M estimates $\boldsymbol{\theta}_M$ and σ_M of $\boldsymbol{\theta}$ and σ with the setup for M estimation that is described in the section "M Estimation" on page 8985. If $\sigma_M > s^*$, give a warning and return s^* and $\boldsymbol{\theta}^*$; otherwise, return σ_M and $\boldsymbol{\theta}_M$.

You can specify TOLS by using the TOLERANCE= option; by default, TOLERANCE=0.001. Alternately, you can specify NREP by using the NREP= option. You can also use the option NREP=NREP0 or NREP=NREP1 to determine NREP according to Table 107.11. NREP=NREP0 is set as the default.

Table 107.11 Default NREP

P	NREP0	NREP1
1	150	500
2	300	1000
3	400	1500
4	500	2000
5	600	2500
6	700	3000
7	850	3000
8	1250	3000
9	1500	3000
>9	1500	3000

NOTE: At step 1 in the algorithm, a randomly selected q-subset might be degenerate. If the total number of q-subsets from any subgroup is greater than 4,000 and the ratio of degenerate q-subsets is higher than the threshold specified in the FAILRATIO= option, the algorithm terminates with an error message.

R Square and Deviance

For the model with the intercept term, the robust version of R square for the S estimate is defined as

$$R_S^2 = 1 - \frac{(n-p)S_p^2}{(n-1)S_\mu^2}$$

For the model without the intercept term, it is defined as

$$R_S^2 = 1 - \frac{(n-p)S_p^2}{nS_0^2}$$

In both cases, S_p is the S estimate of the scale in the full model, S_μ is the S estimate of the scale in the regression model with only the intercept term, and S_0 is the S estimate of the scale without any regressor. The deviance D is defined as the optimal value of the objective function on the σ^2 scale:

$$D = S_p^2$$

Asymptotic Covariance and Confidence Intervals

Because the S estimate satisfies the first-order necessary conditions as the M estimate, it has the same asymptotic covariance as the M estimate. All three estimators of the asymptotic covariance for the M estimate in the section "Asymptotic Covariance and Confidence Intervals" on page 8989 can be used for the S estimate. Besides, the weighted covariance estimator H4 that is described in the section "Asymptotic Covariance and Confidence Intervals" on page 8999 is also available and is set as the default. Confidence intervals for estimated parameters are computed from the diagonal elements of the estimated asymptotic covariance matrix.

MM Estimation

MM estimation is a combination of high breakdown value estimation and efficient estimation that was introduced by Yohai (1987). It has the following three steps:

1. Compute an initial (consistent) high breakdown value estimate $\hat{\boldsymbol{\theta}}'$. The ROBUSTREG procedure provides two kinds of estimates as the initial estimate: the LTS estimate and the S estimate. By default, the LTS estimate is used because of its speed and high breakdown value. The breakdown value of the final MM estimate is decided by the breakdown value of the initial LTS estimate and the constant k_0 in the χ function. To use the S estimate as the initial estimate, specify the INITEST=S option in the PROC ROBUSTREG statement. In this case, the breakdown value of the final MM estimate is decided only by the constant k_0. Instead of computing the LTS estimate or the S estimate as the initial estimate, you can also specify the initial estimate explicitly by using the INEST= option in the PROC ROBUSTREG statement. For more information, see the section "INEST= Data Set" on page 9008.

2. Find $\hat{\sigma}'$ such that

$$\frac{1}{n-p}\sum_{i=1}^{n}\chi\left(\frac{y_i - \mathbf{x}_i'\hat{\boldsymbol{\theta}}'}{\hat{\sigma}'}\right) = \beta$$

where $\beta = \int \chi(s)d\Phi(s)$.

The ROBUSTREG procedure provides two choices for χ: Tukey's bisquare function and Yohai's optimal function.

Tukey's bisquare function, which you can specify by using the option CHIF=TUKEY, is

$$\chi_{k_0}(s) = \begin{cases} 3(\frac{s}{k_0})^2 - 3(\frac{s}{k_0})^4 + (\frac{s}{k_0})^6 & \text{if } |s| \le k_0 \\ 1 & \text{otherwise} \end{cases}$$

where k_0 can be specified by using the K0= option. The default k_0 is 2.9366, such that the asymptotically consistent scale estimate $\hat{\sigma}'$ has a breakdown value of 25%.

Yohai's optimal function, which you can specify by using the option CHIF=YOHAI, is

$$\chi_{k_0}(s) = \begin{cases} \frac{s^2}{2} & \text{if } |s| \le 2k_0 \\ k_0^2[b_0 + b_1(\frac{s}{k_0})^2 + b_2(\frac{s}{k_0})^4 & \\ +b_3(\frac{s}{k_0})^6 + b_4(\frac{s}{k_0})^8] & \text{if } 2k_0 < |s| \le 3k_0 \\ 3.25k_0^2 & \text{if } |s| > 3k_0 \end{cases}$$

where $b_0 = 1.792$, $b_1 = -0.972$, $b_2 = 0.432$, $b_3 = -0.052$, and $b_4 = 0.002$. You can use the K0= option to specify k_0. The default k_0 is 0.7405, such that the asymptotically consistent scale estimate $\hat{\sigma}'$ has a breakdown value of 25%.

3. Find a local minimum $\hat{\boldsymbol{\theta}}_{MM}$ of

$$Q_{MM} = \sum_{i=1}^{n}\rho\left(\frac{y_i - \mathbf{x}_i'\boldsymbol{\theta}}{\hat{\sigma}'}\right)$$

such that $Q_{MM}(\hat{\boldsymbol{\theta}}_{MM}) \le Q_{MM}(\hat{\boldsymbol{\theta}}')$. The algorithm for M estimation is used here.

The ROBUSTREG procedure provides two choices for ρ: Tukey's bisquare function and Yohai's optimal function.

Tukey's bisquare function, which you can specify by using the option CHIF=TUKEY, is

$$\rho(s) = \chi_{k_1}(s) = \begin{cases} 3(\frac{s}{k_1})^2 - 3(\frac{s}{k_1})^4 + (\frac{s}{k_1})^6 & \text{if } |s| \le k_1 \\ 1 & \text{otherwise} \end{cases}$$

where k_1 can be indirectly specified by using the EFF= option. The default $k_1 = 3.440$ for the default EFF=0.85 option such that the MM estimate has 85% asymptotic efficiency with the Gaussian distribution.

Yohai's optimal function, which you can specify by using the option CHIF=YOHAI, is

$$\rho(s) = \chi_{k_1}(s) = \begin{cases} \frac{s^2}{2} & \text{if } |s| \le 2k_1 \\ k_1^2[b_0 + b_1(\frac{s}{k_1})^2 + b_2(\frac{s}{k_1})^4 & \\ +b_3(\frac{s}{k_1})^6 + b_4(\frac{s}{k_1})^8] & \text{if } 2k_1 < |s| \le 3k_1 \\ 3.25k_1^2 & \text{if } |s| > 3k_1 \end{cases}$$

where k_1 can be indirectly specified by using the EFF= option. The default $k_1 = 0.868$ for the default EFF=0.85 option such that the MM estimate has 85% asymptotic efficiency with the Gaussian distribution.

Algorithm

The initial LTS estimate is computed using the algorithm described in the section "LTS Estimate" on page 8991. You can control the quantile of the LTS estimate by specifying the option INITH=*h*, where *h* is an integer between $[\frac{n}{2}] + 1$ and $[\frac{3n+p+1}{4}]$. By default, $h = [\frac{3n+p+1}{4}]$, which corresponds to a breakdown value of around 25%.

The initial S estimate is computed using the algorithm described in the section "S Estimate" on page 8994. You can control the breakdown value and efficiency of this initial S estimate by the constant k_0, which you can specify by using the K0= option.

The scale parameter σ is solved by an iterative algorithm

$$(\sigma^{(m+1)})^2 = \frac{1}{(n-p)\beta} \sum_{i=1}^{n} \chi_{k_0}\left(\frac{r_i}{\sigma^{(m)}}\right)\left(\sigma^{(m)}\right)^2$$

where $\beta = \int \chi_{k_0}(s)d\Phi(s)$.

After the scale parameter is computed, the iteratively reweighted least squares (IRLS) algorithm with fixed scale parameter is used to compute the final MM estimate.

Convergence Criteria

In the iterative algorithm for the scale parameter, the relative change of the scale parameter controls the convergence.

In the iteratively reweighted least squares algorithm, the same convergence criteria for the M estimate that are used before are used here.

Bias Test

Although the final MM estimate inherits the high breakdown value property, its bias from the distortion of the outliers can be high. Yohai, Stahel, and Zamar (1991) introduced a bias test. The ROBUSTREG procedure implements this test when you specify the BIASTEST= option in the PROC ROBUSTREG statement. This test is based on the initial scale estimate $\hat{\sigma}'$ and the final scale estimate $\hat{\sigma}'_1$, which is the solution of

$$\frac{1}{n-p}\sum_{i=1}^{n} \chi\left(\frac{y_i - \mathbf{x}_i'\hat{\boldsymbol{\theta}}_{MM}}{\hat{\sigma}'_1}\right) = \beta$$

Let $\psi_{k_0}(z) = \frac{\partial \chi_{k_0}(z)}{\partial z}$ and $\psi_{k_1}(z) = \frac{\partial \chi_{k_1}(z)}{\partial z}$. Compute

$$\begin{aligned} \tilde{r}_i &= (y_i - \mathbf{x}_i'\hat{\boldsymbol{\theta}}')/\hat{\sigma}' \quad \text{for } i = 1, \ldots, n \\ v_0 &= \frac{(1/n)\sum \psi'_{k_0}(\tilde{r}_i)}{(\hat{\sigma}'_1/n)\sum \psi_{k_0}(\tilde{r}_i)\tilde{r}_i} \end{aligned}$$

$$
\begin{aligned}
p_i^{(0)} &= \frac{\psi_{k_0}(\tilde{r}_i)}{(1/n)\sum \psi'_{k_0}(\tilde{r}_i)} \quad \text{for } i = 1, \ldots, n \\
p_i^{(1)} &= \frac{\psi_{k_1}(\tilde{r}_i)}{(1/n)\sum \psi'_{k_1}(\tilde{r}_i)} \quad \text{for } i = 1, \ldots, n \\
d^2 &= \frac{1}{n}\sum (p_i^{(1)} - p_i^{(0)})^2
\end{aligned}
$$

Let

$$
T = \frac{2n(\hat{\sigma}'_1 - \hat{\sigma}')}{v_0 d^2 (\hat{\sigma}')^2}
$$

Standard asymptotic theory shows that T approximately follows a χ^2 distribution with p degrees of freedom. If T exceeds the α quantile χ^2_α of the χ^2 distribution with p degrees of freedom, then the ROBUSTREG procedure gives a warning and recommends that you use other methods. Otherwise, the final MM estimate and the initial scale estimate are reported. You can specify α by using the ALPHA= option after the BIASTEST= option. By default, ALPHA=0.99.

Asymptotic Covariance and Confidence Intervals

Because the MM estimate is computed as an M estimate with a known scale in the last step, the asymptotic covariance for the M estimate can be used here for the asymptotic covariance of the MM estimate. Besides the three estimators H1, H2, and H3 as described in the section "Asymptotic Covariance and Confidence Intervals" on page 8989, a weighted covariance estimator H4 is available. H4 is calculated as

$$
K^2 \frac{[1/(n-p)]\sum(\psi(r_i))^2}{[(1/n)\sum(\psi'(r_i))]^2}\mathbf{W}^{-1}
$$

where $K = 1 + \frac{p}{n}\frac{\mathrm{Var}(\psi')}{(E\psi')^2}$ is the correction factor and $W_{jk} = \frac{1}{\bar{w}}\sum w_i x_{ij} x_{ik}$, $\bar{w} = \frac{1}{n}\sum w_i$.

You can specify these estimators by using the option ASYMPCOV=[H1 | H2 | H3 | H4]. The ROBUSTREG procedure uses H4 as the default. Confidence intervals for estimated parameters are computed from the diagonal elements of the estimated asymptotic covariance matrix.

R Square and Deviance

The robust version of R square for the MM estimate is defined as

$$
R^2 = \frac{\sum \rho\left(\frac{y_i - \hat{\mu}}{\hat{s}}\right) - \sum \rho\left(\frac{y_i - \mathbf{x}'_i\hat{\boldsymbol{\theta}}}{\hat{s}}\right)}{\sum \rho\left(\frac{y_i - \hat{\mu}}{\hat{s}}\right)}
$$

and the robust deviance is defined as the optimal value of the objective function on the σ^2 scale,

$$
D = 2\hat{s}^2 \sum \rho\left(\frac{y_i - \mathbf{x}'_i\hat{\boldsymbol{\theta}}}{\hat{s}}\right)
$$

where $\rho' = \psi$, $\hat{\boldsymbol{\theta}}$ is the MM estimator of $\boldsymbol{\theta}$, $\hat{\mu}$ is the MM estimator of location, and $\hat{s}$ is the MM estimator of the scale parameter in the full model.

Linear Tests

For MM estimation, you can use the same ρ test and R_n^2 test that used for M estimation. For more information, see the section "Linear Tests" on page 8990.

Model Selection

For MM estimation, you can use the same two model selection methods that are used for M estimation. For more information, see the section "Model Selection" on page 8991.

Robust Distance

The ROBUSTREG procedure uses the robust multivariate location and scatter estimates for leverage-point detection. The procedure computes a robust version of the Mahalanobis distance by using a generalized minimum covariance determinant (MCD) method. The original MCD method was proposed by Rousseeuw (1984).

Algorithm

PROC ROBUSTREG implements a generalized MCD algorithm that is based on the fast-MCD algorithm formulated by Rousseeuw and Van Driessen (1999), which is similar to the algorithm for least trimmed squares (LTS).

Mahalanobis Distance versus Robust Distance

The canonical Mahalanobis distance is defined as

$$\mathrm{MD}(\mathbf{x}_i) = [(\mathbf{x}_i - \bar{\mathbf{x}})'\bar{\mathbf{C}}(\mathbf{X})^{-1}(\mathbf{x}_i - \bar{\mathbf{x}})]^{1/2}$$

where $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$ and $\bar{\mathbf{C}}(\mathbf{X}) = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})'(\mathbf{x}_i - \bar{\mathbf{x}})$ are the empirical multivariate location and scatter, respectively. Here $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})'$ excludes the intercept. The relationship between the Mahalanobis distance $\mathrm{MD}(\mathbf{x}_i)$ and the hat matrix $\mathbf{H} = (h_{ij}) = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$ is

$$h_{ii} = \frac{1}{n-1}\mathrm{MD}_i^2 + \frac{1}{n}$$

The canonical robust distance is defined as

$$\mathrm{RD}(\mathbf{x}_i) = [(\mathbf{x}_i - \mathbf{T}(\mathbf{X}))'\mathbf{C}(\mathbf{X})^{-1}(\mathbf{x}_i - \mathbf{T}(\mathbf{X}))]^{1/2}$$

where $\mathbf{T}(\mathbf{X})$ and $\mathbf{C}(\mathbf{X})$ are the robust multivariate location and scatter, respectively, that are obtained by the MCD method.

To achieve robustness, the MCD algorithm estimates the covariance of a multivariate data set mainly through an MCD h-point subset of the data set. This subset has the smallest sample-covariance determinant among all possible h-subsets. Accordingly, the breakdown value for the MCD algorithm equals $\frac{(n-h)}{n}$. This means the MCD estimate is reliable, even if up to $\frac{100(n-h)}{n}\%$ observations in the data set are contaminated.

Low-Dimensional Structure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/rregmcd.sas.)

It is possible that the original data are in p-dimensional space but the h-point subset that yields the minimum covariance determinant lies in a lower-dimensional hyperplane. Applying the canonical MCD algorithm to such a data set would result in a singular covariance problem (called exact fit in Rousseeuw and Van Driessen 1999), such that the relevant robust distances cannot be computed. To deal with the singularity problem and provide further leverage-point analysis, PROC ROBUSTREG implements a generalized MCD algorithm. (For more information, see the section "Generalized MCD Algorithm" on page 9004.) The algorithm distinguishes in-(hyper)plane points from off-(hyper)plane points, and performs MCD leverage-point analysis in the dimension-reduced space by projecting all points onto the hyperplane.

Low-dimensional structure is often induced by classification covariates. Suppose that, in a study that has 25 female subjects and 5 male subjects, *gender* is the only classification effect. If the breakdown setting is greater than $\frac{5}{(25+5)}$, the canonical MCD algorithm fails, and so does the relevant leverage-point analysis. In this case, the MCD h-subset would contain only female observations, and the constant gender in the h-subset would cause the relevant MCD estimate to be singular. The generalized MCD algorithm solves that problem by identifying all male observations as off-plane leverage points, and then carries out the leverage-point analysis, with all the other covariates being centered separately for female and male groups against their group means.

In general, low-dimensional structure is not necessarily due to classification covariates. Imagine that 80 children are supposed to play on a straight trail (denoted by $y = x$) but that some adventurous children go off the trail. The following statements generate the Children data set and the relevant scatter plot:

```
data Children;
   do i=1 to 80;
      off_trail=ranuni(321)>.9;
      x=rannor(111)*ranuni(321);
      trail_x=(i-40)/80*3;
      trail_y=trail_x;
      if off_trail=1 then y=x-1+rannor(321);
      else y=x;
      output;
   end;
run;

proc sgplot data=Children;
   series   x=trail_x y=trail_y/lineattrs=(color="red" pattern=4);
   scatter  x=x y=y/group=off_trail;
   ellipse  x=x y=y/alpha=.05 lineattrs=(color="green" pattern=34);
run;
```

Figure 107.17 shows the positions of all 80 children, the trail (as a red dashed line), and a contour curve of regular Mahalanobis distance that is centered at the mean position (as a green dotted ellipse). In terms of regular Mahalanobis distance, the associated covariance estimate is not singular, but its relevant leverage-point analysis completely ignores the trail (which is the entity of the low-dimensional structure). The children outside the ellipse are defined as leverage points, but the children off the trail would not be viewed as leverage points unless they had large Mahalanobis distances. As mentioned in Rousseeuw and Van Driessen (1999), the canonical MCD method can find the low-dimensional structure, but it does not provide further robust covariance estimation because the MCD covariance estimate is singular. As an improved version of the

canonical MCD method, the generalized MCD method can find the trail, identify the children off the trail as off-plane leverage points, and further execute in-plane leverage analysis. The following statements apply the generalized MCD algorithm to the Children data set:

```
ods graphics on;
proc robustreg data=children plots=ddplot(label=none);
   model i = x y/leverage(mcdinfo opc);
run;
ods graphics off;
```

Figure 107.17 Scatter Plot for Children Data

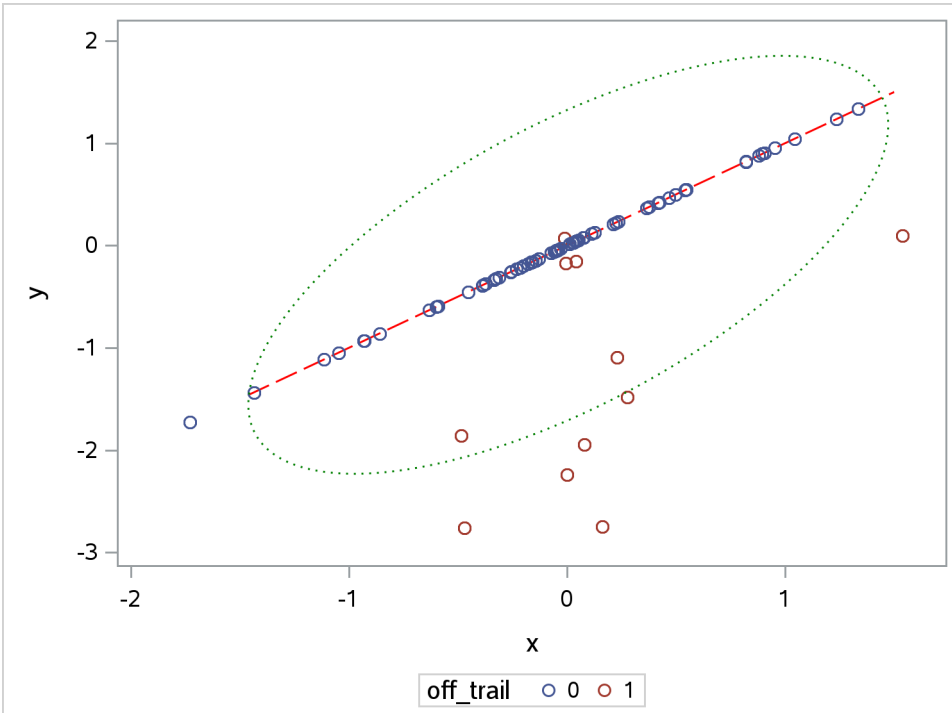

Figure 107.18 exactly identifies the equation that underlies the trail. The analysis projects off-plane points onto the trail and computes their projected robust distances and projected Mahalanobis distances the same way as it does for the in-plane points.

Figure 107.18 Robust Dependence Equations

The ROBUSTREG Procedure

Note: The following robust dependence equations simultaneously hold for 86.25% of the observations in the data set. The breakdown setting for the MCD algorithm is 25.00 %.

y = x

Figure 107.19 shows the relevant distance-distance plot. Robust distance is typically greater than Mahalanobis distance because the sample covariance can be strongly influenced by unusual points that cause the sample covariance to be larger than the MCD covariance.

Figure 107.19 DD Plot for Children Data

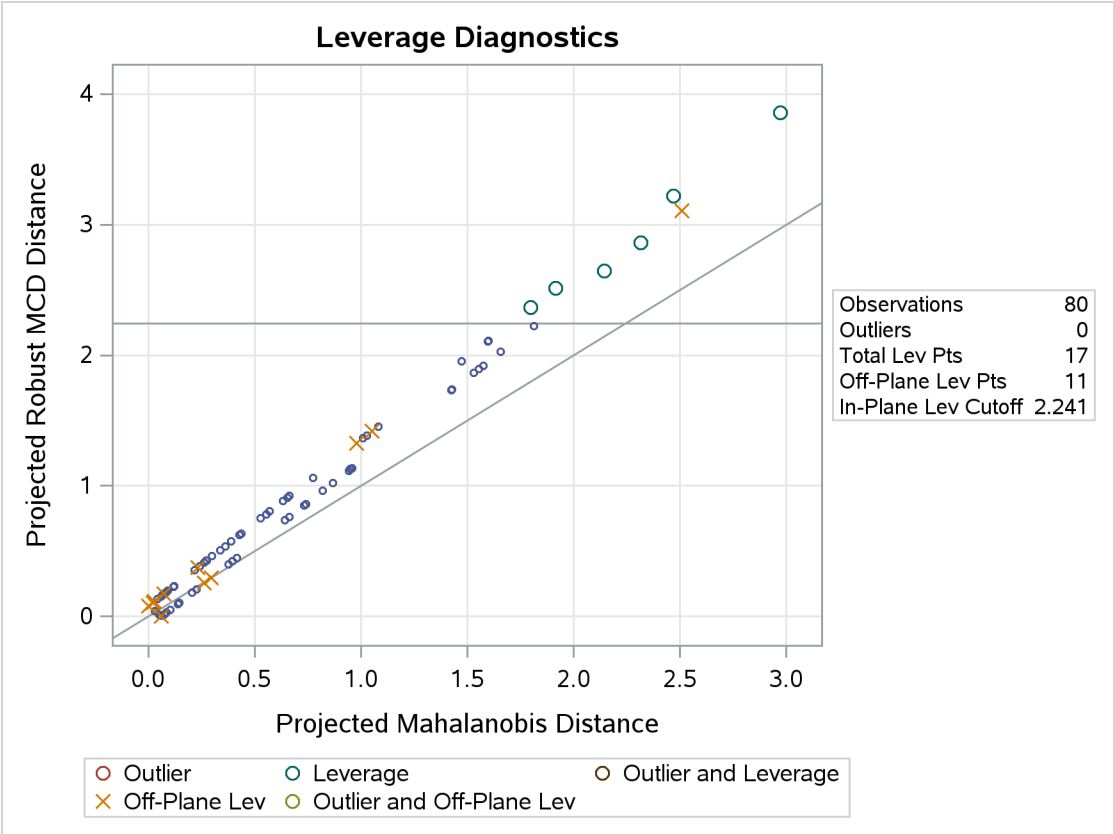

NOTE: The PROC ROBUSTREG step in this example is used to obtain the leverage diagnostics; the response is not relevant for this analysis.

Through the off-plane and in-plane symbols and the horizontal cutoff line in Figure 107.19, you can separate the children into four groups:

- on the trail and close to the MCD center
- on the trail but far away from the MCD center
- off the trail but close to the MCD center
- off the trail and far away from the MCD center

The children in the latter three groups are defined as leverage points in PROC ROBUSTREG.

Generalized MCD Algorithm

The generalized MCD algorithm follows the same resampling strategy as the canonical MCD algorithm by Rousseeuw and Van Driessen (1999) but with the following modifications:

1. Data are orthonormalized before further processing. The orthonormalized covariates, $\mathbf{x}_i^*$, are defined by $\mathbf{x}_i^* = (\mathbf{x}_i - \bar{\mathbf{x}})\mathbf{P}\mathbf{\Lambda}^{-1/2}$, where $\mathbf{P}$ and $\mathbf{\Lambda}$ are the eigenvector and eigenvalue matrices, respectively, of $\bar{\mathbf{C}}(\mathbf{X})$ (that is, $\bar{\mathbf{C}}(\mathbf{X}) = \mathbf{P}\mathbf{\Lambda}\mathbf{P}'$).

2. Let

$$S_h(\mathbf{X}^*) = \frac{1}{h-1}\sum_{j=1}^{h}(\mathbf{x}_{i_j}^* - \bar{\mathbf{x}}^*)'(\mathbf{x}_{i_j}^* - \bar{\mathbf{x}}^*) = \sum_{j=1}^{p-1}\lambda_j \mathbf{p}_j \mathbf{p}_j{}'$$

denote the covariance and eigendecomposition of a low-dimensional h-subset $\{\mathbf{x}_{i_1}^*, \ldots, \mathbf{x}_{i_h}^*\}$, where $\bar{\mathbf{x}}^* = \frac{1}{h}\sum_{j=1}^{h}\mathbf{x}_{i_j}^*$ and the eigenvalues satisfy

$$\lambda_1 \geq \cdots \geq \lambda_q > 0 = \lambda_{q+1} = \cdots = \lambda_p$$

Then, the rank of $S_h(\mathbf{X}^*)$ equals q, and the pseudo-determinant of $S_h(\mathbf{X}^*)$ is defined as $\prod_{j=1}^{q}\lambda_j$. In finite-precision arithmetic, q is defined as the number of λ's where $\frac{\lambda_i}{\lambda_1}$ is greater than a certain tolerance value. You can specify this tolerance by using the PTOL= suboption of the LEVERAGE option.

3. If $S_h(\mathbf{X}^*)$ and $\bar{\mathbf{x}}^*$ are the covariance and center estimates, respectively, then the projected Mahalanobis distance for $\mathbf{x}_i$ is defined as

$$\left[\sum_{j=1}^{q}\frac{\left((\mathbf{x}_i^* - \bar{\mathbf{x}}^*)\mathbf{p}_j\right)^2}{\lambda_j}\right]^{1/2}$$

The generalized algorithm also computes off-plane distance for each $\mathbf{x}_i$ as

$$\left[\sum_{j=q+1}^{p}\left((\mathbf{x}_i^* - \bar{\mathbf{x}}^*)\mathbf{p}_j\right)^2\right]^{1/2}$$

In finite-precision arithmetic, $\left((\mathbf{x}_i^* - \bar{\mathbf{x}}^*)\mathbf{p}_j\right)^2$ in the previous off-plane formula are truncated to zero if they satisfy

$$\frac{\left((\mathbf{x}_i^* - \bar{\mathbf{x}}^*)\mathbf{p}_j\right)^2}{\lambda_j} \leq \text{cutoff}$$

You can tune this cutoff by using either the PCUTOFF= or PALPHA= suboption of the LEVERAGE option. The points with zero off-plane distances are called in-plane points; otherwise, they are called off-plane points. Analogous to ordering all points in terms of their canonical Mahalanobis distances, for the generalized MCD algorithm the points are first sorted by their off-plane distances, and the points with the same off-plane distance values are further sorted by their projected Mahalanobis distances.

4. Instead of comparing the determinants of h-subset covariance matrices, the generalized algorithm compares both the ranks and pseudo-determinants of the h-subset covariance matrices. If the ranks of two matrices are different, the matrix that has the smaller rank is treated as if its determinant were smaller. If two matrices are of the same rank, they are compared in terms of their pseudo-determinants.

5. Suppose that the $S_h(\mathbf{X}^*)$ of the minimum determinant is singular. Then the relevant low-dimensional structure or hyperplane can be identified by using the eigendecomposition of $S_h(\mathbf{X}^*)$. The eigenvectors that correspond to the nonzero eigenvalues form a basis for the low-dimensional hyperplane. The projected off-plane distance (POD) for $\mathbf{x}_i$ is defined as the off-plane distance that is associated with the $S_h(\mathbf{X}^*)$. To provide further leverage analysis on the low-dimensional hyperplane, every $\mathbf{x}_i^*$ is transformed into $(\mathbf{x}_i^*\mathbf{p}_1, \ldots, \mathbf{x}_i^*\mathbf{p}_q)$, where $\mathbf{p}_j$ are the eigenvectors of the $S_h(\mathbf{X}^*)$. The projected robust distance (PRD) is then computed as the reweighted Mahalanobis distance on all the transformed in-plane points. The off-plane points are assigned zero weights at the reweighting stage, because they are leverage points by definition. The in-plane points are classified into two groups, the normal group and the in-plane leverage group. This classification is made by comparing their projected robust distances with a leverage cutoff value. (For more information, see the section "Leverage-Point and Outlier Detection" on page 9006.) This reweighting process mirrors the one that was proposed by Rousseeuw and Van Driessen (1999). However, the degrees of freedom p for the reweighting critical χ^2 value are replaced by q. You can control the χ^2 critical value by specifying the MCDCUTOFF= or MCDALPHA= option.

If the data set under investigation has a low-dimensional structure, you can use two ODS objects, DependenceEquations and MCDDependenceEquations, to identify the regressors that are linear combinations of other regressors plus certain constants. The equations in DependenceEquations hold for the entire data set, whereas the equations in MCDDependenceEquations apply only to the majority of the observations.

By using the OPC suboption of the LEVERAGE option, you can request an ODS table called DroppedComponents. Figure 107.20 shows the DroppedComponents table for the Children data set example. This table contains a set of coefficient vectors for regressors, which form a basis of the complementary space for the relevant low-dimensional structure.

Figure 107.20 MCD-Dropped Components

Coefficients for MCD-Dropped Components	
Parameter	**RobustDrop1**
x	1.0000
y	-1.000

By using the MCDINFO suboption of the LEVERAGE option, you can request that detailed information about the MCD covariance estimate be displayed in four ODS tables: MCDProfile, MCDCenter, MCDCov, and MCDCorr. Figure 107.21 shows an example of the MCD information tables for the Children data set. The number of dimensions in the table MCDProfile equals the number of nonintercept regressors minus the number of design-dropped components. The specified value of H is the same as h for the h-subset that you can specify by using the QUANTILE= suboption of the LEVERAGE option in the MODEL statement. The reweighted H is the number of observations that are actually used to compute the MCD center and MCD covariance after the reweighting step of the MCD algorithm.

Figure 107.21 MCD Information

MCD Profile	
Number of Dimensions	2
Number of Robust Dropped Components	1
Number of Observations	80
Number of Off-Plane Observations	11
Specified Value of H	60
Reweighted Value of H	63
Breakdown Value	0.2500

MCD Center		
Parameter Name	Parameter	Center
x	x	0.0307
y	y	0.0307

MCD Covariance		
	x	y
x	0.207713	0.207713
y	0.207713	0.207713

MCD Correlation		
	x	y
x	1	1
y	1	1

Leverage-Point and Outlier Detection

The regular variable LEVERAGE is defined as

$$\text{LEVERAGE} = \begin{cases} 0 & \text{if } \mathrm{RD}(\mathbf{x}_i) \le C(p) \\ 1 & \text{otherwise} \end{cases}$$

where $C(p) = \sqrt{\chi^2_{p;1-\alpha}}$ is the cutoff value. $C(p)$ can be set by using the leverage CUTOFF= option, and α can be set by using the leverage CUTOFFALPHA= option.

If projected robust distances are computed for a data set that has a low-dimensional structure, the default cutoff value is $C(q) = \sqrt{\chi^2_{q;1-\alpha}}$, where q is the dimensionality of the low-dimensional space. LEVERAGE is then defined as

$$\text{LEVERAGE} = \begin{cases} 0 & \text{if } \mathrm{POD}(\mathbf{x}_i) = 0 \text{ and } \mathrm{PRD}(\mathbf{x}_i) \le C(q) \\ 1 & \text{if } \mathrm{POD}(\mathbf{x}_i) = 0 \text{ and } \mathrm{PRD}(\mathbf{x}_i) > C(q) \text{ (called in-plane leverage)} \\ 1 & \text{if } \mathrm{POD}(\mathbf{x}_i) > 0 \text{ (called off-plane leverage)} \end{cases}$$

where POD is the projected off-plane distance and PRD denotes the projected robust distance. You can specify a cutoff value by using the CUTOFF= or CUTOFFALPHA= suboption of the LEVERAGE option in the MODEL statement.

Residuals $r_i, i = 1, \ldots, n$, based on robust regression estimates are used to detect vertical outliers. The variable OUTLIER is defined as

$$\text{OUTLIER} = \begin{cases} 0 & \text{if } |r_i| \le k\hat{\sigma} \\ 1 & \text{otherwise} \end{cases}$$

where $\hat{\sigma}$ is the estimated scale in the model and the multiplier k of the cutoff value is specified by the CUTOFF= option in the MODEL statement. By default, $k = 3$.

An ODS table called Diagnostics contains the LEVERAGE and OUTLIER variables.

Implementation of the WEIGHT Statement

You can use the WEIGHT statement to specify a weight variable in the input data set. (For more information, see the section "WEIGHT Statement" on page 8984.) This section describes how PROC ROBUSTREG implements the WEIGHT statement for each of the estimation methods and for leverage detection.

M Estimation

If you use M estimation with a known scale, then instead of minimizing $Q(\boldsymbol{\theta}) = \sum_{i=1}^{n} \rho(\frac{r_i}{\sigma})$, the weighted M estimation minimizes the weighted Huber-type objective function

$$Q(\boldsymbol{\theta}) = \sum_{i=1}^{n} v_i \rho\left(\frac{r_i}{\sigma}\right)$$

where v_i is the weight variable that is specified by the WEIGHT statement. If you use M estimation with an unknown scale, the weight variable is used in the location steps but not in the scale steps. (For more information, see the section "M Estimation" on page 8985 and the SCALE= option.) For estimating the covariance of the weighted M estimation, $\psi(r_i)$ and $\psi'(r_i)$ are obtained from the final iteration of the weighted M estimation, and $\mathbf{X}'\mathbf{X}$ and $\mathbf{W}$ are replaced, respectively, by $\mathbf{X}'\mathbf{V}\mathbf{X}$ and $W_{jk} = \sum v_i \psi'(r_i) x_{ij} x_{ik}$, where $\mathbf{V}$ is a diagonal matrix whose diagonal elements are v_i. (For more information, see the section "Asymptotic Covariance and Confidence Intervals" on page 8989.) The weight variable does not affect the model degrees of freedom p and the error degrees of freedom $n - p$.

LTS Estimation

LTS estimation ignores the weight variable.

S Estimation

S estimation applies the weight variable only in its M-refinement step. Except for the initial estimates, the M-refinement step of S estimation is the same as the weighted M estimation with unknown scale. If you use the NOREFINE suboption, S estimation ignores the weight variable along with the M-refinement step.

MM Estimation

By default, the initial step of MM estimation is the initial LTS estimation. Unlike the regular LTS estimation, the initial LTS estimation is applied to the weighted data $(y_i{}^*, \mathbf{x}_i{}^*)$'s, where $y_i{}^* = \sqrt{v_i}\, y_i$ and $\mathbf{x}_i{}^* = \sqrt{v_i}\mathbf{x}_i$. After the initial LTS estimation, the weight variable is ignored for the subsequent scale adjustment.

You can use INITEST=S to specify the initial S estimation as the initial step of the MM estimation. As with the regular S estimation, the weight variable is used only in the M-refinement step of the initial S estimation. There is no subsequent scale adjustment step if the initial S estimation is applied.

Except for the initial estimates, the final M estimation of the MM estimation is the same as the weighted M estimation with known scale.

Final Weighted Least Squares Estimation

Final weighted least squares estimation is always applied to the weighted data $(y_i{}^*, \mathbf{x}_i{}^*)$, no matter how the weight variable is applied in the preceding estimation. For example, if the option METHOD=LTS is specified along with the FWLS option, although the outliers that are identified by LTS estimation do not depend on the weight variable, final weighted least squares estimation applies the weight variable to all the points that are not outliers.

Robust Distances and Leverage Detection

Robust distance computation ignores the weight variable. Because leverage detection depends on robust distance, it also ignores the weight variable.

INEST= Data Set

When you use M or MM estimation, you can use the INEST= data set to specify initial estimates for all the parameters in the model. The INEST= option is ignored if you specify LTS or S estimation by using the METHOD=LTS or METHOD=S option or if you specify the INITEST= option after the METHOD=MM option in the PROC ROBUSTREG statement. The INEST= data set must contain the intercept variable (named Intercept) and all independent variables in the MODEL statement.

If BY processing is used, the INEST= data set should also include the BY variables, and there must be at least one observation in each BY group. If there is more than one observation in a BY group, the first one that is read is used for that BY group.

If the INEST= data set also contains the _TYPE_ variable, only observations that have the _TYPE_ value "PARMS" are used as starting values.

You can specify starting values for the iteratively reweighted least squares algorithm in the INEST= data set. The INEST= data set has the same structure as the OUTEST= data set but is not required to contain all the variables or observations that appear in the OUTEST= data set. One simple use of the INEST= option is to pass the previous OUTEST= data set directly to the next model as an INEST= data set, assuming that the two models have the same parameterization.

OUTEST= Data Set

The OUTEST= data set contains parameter estimates for the model. You can specify a label in the MODEL statement to distinguish between the estimates for different models that the ROBUSTREG procedure uses. If the COVOUT option is specified, the OUTEST= data set also contains the estimated covariance matrix of the parameter estimates. If the ROBUSTREG procedure does not converge, the parameter estimates are set to missing in the OUTEST data set.

The OUTEST= data set contains all variables that are specified in the MODEL statement and the BY statement. For each BY group, the OUTEST= data set has the following:

- one observation that consists of the model's parameter estimates, where the value of the dependent variable is set to –1
- if the COVOUT option is specified, p observations that contain the rows of the estimated covariance matrix. For these observations, the dependent variable contains the parameter estimates for the corresponding row variables.

The following variables are also added to the data set in their display order:

MODEL	is a character variable that contains the label of the MODEL statement, if present. Otherwise, the variable's value is blank.
NAME	is a character variable that contains the name of the dependent variable for the parameter estimates or the name of the row for the covariance matrix estimates.
TYPE	is a character variable that contains the type of the observation: either PARMS for parameter estimates or COV for covariance estimates.
METHOD	is a character variable that contains the type of estimation method: either M estimation, LTS estimation, S estimation, or MM estimation.
STATUS	is a character variable that contains the status of model fitting: either Converged, Warning, or Failed.
INTERCEPT	is a numeric variable that contains the intercept parameter estimates and covariances.
SCALE	is a numeric variable that contains the scale parameter estimates.

Any BY variables that are specified are also added to the OUTEST= data set.

Computational Resources

The algorithms for the various estimation methods need a different amount of memory for working space. Let p be the number of parameters that are estimated, and let n be the number of observations that are used in the model estimation.

For M estimation, the minimum required working space (in bytes) is

$$3n + 2p^2 + 30p$$

If sufficient space is available, the input data set is also kept in memory; otherwise, the input data set is read again to compute the iteratively reweighted least squares estimates, and the execution time of the procedure increases substantially. For each of the reweighted least squares, $O(np^2 + p^3)$ multiplications and additions are required for computing the crossproduct matrix and its inverse. The $O(v)$ notation means that, for large values of the argument, v, $O(v)$ is approximately a constant times v.

Because the iteratively reweighted least squares algorithm converges very quickly (usually within fewer than 20 iterations), the computation of M estimates is fast.

LTS estimation is more expensive in computation. The minimum required working space (in bytes) is

$$np + 12n + 4p^2 + 60p$$

The memory is mainly used to store the current data that the LTS algorithm uses for modeling. The LTS algorithm uses subsampling and spends much of its computing time on resampling and computing estimates for subsamples. Because it resamples if singularity is detected, the LTS algorithm might take more time if the data set has serious singularities.

The MCD algorithm for leverage-point diagnostics is similar to the LTS algorithm.

ODS Table Names

The ROBUSTREG procedure assigns a name to each table that it creates. You can specify these names when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 107.12.

Table 107.12 ODS Tables Produced by PROC ROBUSTREG

ODS Table Name	Description	Statement	Option
BestEstimates	Best final estimates for LTS	PROC ROBUSTREG	SUBANALYSIS
BestHalf	Best half set for LTS	PROC ROBUSTREG	SUBANALYSIS
BestSubEstimates	Best estimates for each subgroup	PROC ROBUSTREG	SUBANALYSIS
BiasTest	Bias test for MM estimation	PROC ROBUSTREG	BIASTEST
ClassLevels	Classification variable levels	CLASS	Default
CorrB	Parameter estimate correlation matrix	MODEL	CORRB
CovB	Parameter estimate covariance matrix	MODEL	COVB

Table 107.12 *continued*

ODS Table Name	Description	Statement	Option
CStep	C-step for LTS fitting	PROC ROBUSTREG	SUBANALYSIS
DependenceEquations	Design dependence equations	MODEL	LEVERAGE
Diagnostics	Outlier diagnostics	MODEL	DIAGNOSTICS
DiagSummary	Summary of the outlier diagnostics	MODEL	Default
DroppedComponents	Coefficients for MCD-dropped components	MODEL	LEVERAGE (OPC)
GoodFit	R square, deviance, AIC, and BIC	PROC ROBUSTREG	METHOD=
InitLTSProfile	Profile for initial LTS estimate	PROC ROBUSTREG	METHOD=
InitSProfile	Profile for initial S estimate	PROC ROBUSTREG	METHOD=
IterHistory	Iteration history	PROC ROBUSTREG	ITPRINT
LTSEstimates	LTS parameter estimates	PROC ROBUSTREG	METHOD=
LTSLocationScale	Location and scale for LTS	PROC ROBUSTREG	METHOD=
LTSProfile	Profile for LTS estimator	PROC ROBUSTREG	METHOD=
LTSRsquare	R square for LTS estimate	PROC ROBUSTREG	METHOD=
MCDDependenceEquations	Robust dependence equations	MODEL	LEVERAGE
MCDProfile	MCD profile	MODEL	LEVERAGE (MCDINFO)
MCDCenter	MCD center estimate	MODEL	LEVERAGE (MCDINFO)
MCDCov	MCD covariance estimate	MODEL	LEVERAGE (MCDINFO)
MCDCorr	MCD correlation estimate	MODEL	LEVERAGE (MCDINFO)
MMProfile	Profile for MM estimator	PROC ROBUSTREG	METHOD=
ModelInfo	Model information	MODEL	Default
NObs	Observations summary	PROC ROBUSTREG	Default
ParameterEstimates	Parameter estimates	MODEL	Default
ParameterEstimatesF	Final weighted LS estimates	PROC ROBUSTREG	FWLS
ParameterEstimatesR	Reduced parameter estimates	TEST	Default
ParmInfo	Parameter indices	MODEL	Default
SProfile	Profile for S estimator	PROC ROBUSTREG	METHOD=
Groups	Groups for LTS fitting	PROC ROBUSTREG	SUBANALYSIS
SummaryStatistics	Summary statistics for model variables	MODEL	Default
Tests	Results for tests	TEST	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

If the model includes a single continuous independent variable, a plot of robust fit against this variable (fit plot) is provided by default. Two plots are particularly useful in revealing outliers and leverage points. The first is a scatter plot of the standardized robust residuals against the robust distances (RD plot). The second is a scatter plot of the robust distances against the classical Mahalanobis distances (DD plot). In addition to these two plots, a histogram and a quantile-quantile plot of the standardized robust residuals are also helpful.

PROC ROBUSTREG assigns a name to each graph that it creates using ODS Graphics. You can use these names to refer to the graphs when using ODS. The graph names and corresponding PLOTS= options are listed in Table 107.13.

Table 107.13 Graphs Produced by PROC ROBUSTREG

ODS Graph Name	Plot Description	Statement	PLOTS= Option
DDPlot	Robust distance versus Mahalanobis distance (or projected robust distance versus projected Mahalanobis distance)	PROC ROBUSTREG	DDPLOT
FitPlot	Robust fit versus independent variable	PROC ROBUSTREG	FITPLOT
Histogram	Histogram of standardized robust residuals	PROC ROBUSTREG	HISTOGRAM
QQPlot	Quantile-quantile plot of standardized robust residuals	PROC ROBUSTREG	QQPLOT
RDPlot	Standardized robust residual versus robust distance (or projected robust distance)	PROC ROBUSTREG	RDPLOT

Fit Plot

When the model has a single independent continuous variable (with or without the intercept), the ROBUSTREG procedure automatically creates a plot of robust fit against this independent variable.

The following simple example shows the fit plot. The data, from Rousseeuw and Leroy (1987, Table 3), include the logarithm of surface temperature and the logarithm of light intensity for 47 stars in the direction of the constellation Cygnus.

```
data star;
   input index x y @@;
   label x = 'Log Temperature'
         y = 'Log Light Intensity';
```

```
   datalines;
 1  4.37  5.23    2  4.56  5.74    3  4.26  4.93    4  4.56  5.74
 5  4.30  5.19    6  4.46  5.46    7  3.84  4.65    8  4.57  5.27
 9  4.26  5.57   10  4.37  5.12   11  3.49  5.73   12  4.43  5.45
13  4.48  5.42   14  4.01  4.05   15  4.29  4.26   16  4.42  4.58
17  4.23  3.94   18  4.42  4.18   19  4.23  4.18   20  3.49  5.89
21  4.29  4.38   22  4.29  4.22   23  4.42  4.42   24  4.49  4.85
25  4.38  5.02   26  4.42  4.66   27  4.29  4.66   28  4.38  4.90
29  4.22  4.39   30  3.48  6.05   31  4.38  4.42   32  4.56  5.10
33  4.45  5.22   34  3.49  6.29   35  4.23  4.34   36  4.62  5.62
37  4.53  5.10   38  4.45  5.22   39  4.53  5.18   40  4.43  5.57
41  4.38  4.62   42  4.45  5.06   43  4.50  5.34   44  4.45  5.34
45  4.55  5.54   46  4.45  4.98   47  4.42  4.50
;
```

The following statements use the MM method to plot the robust fit of the logarithm of light intensity against the logarithm of the surface temperature:

```
ods graphics on;
proc robustreg data=star method=mm;
   model y = x;
run;
```

Figure 107.22 shows the fit plot. Confidence limits are added to the plot by default.

Figure 107.22 Robust Fit

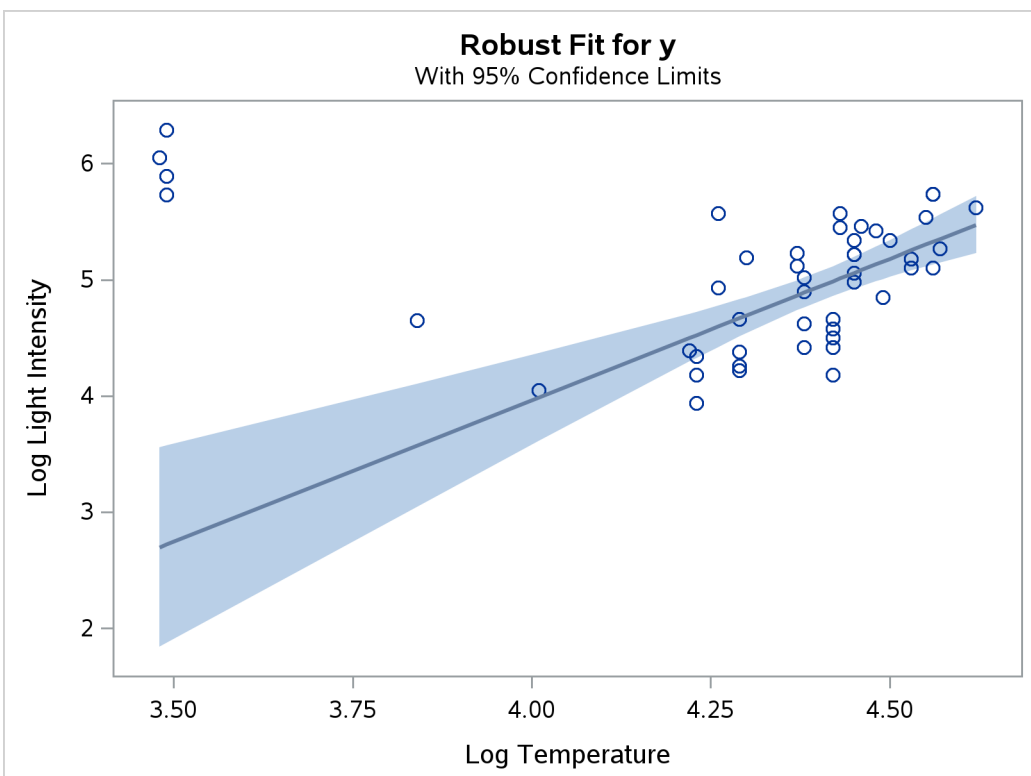

You can suppress the confidence limits by specifying the NOLIMITS option, as shown in the following statements:

```
proc robustreg data=star method=mm plot=fitplot(nolimits);
   model y = x;
run;
```

Distance-Distance Plot

The distance-distance (DD) plot is mainly used for leverage-point diagnostics. It is a scatter plot of the robust distances (or projected robust distances) against the classical Mahalanobis distances (or projected classical Mahalanobis distances) for the independent variables. For more information about the robust distance, see the section "Leverage-Point and Outlier Detection" on page 9006.

You can use the PLOT=DDPLOT option to request this plot. The following statements use the Stack data set in the section "M Estimation" on page 8957 to create the single plot shown in Figure 107.5:

```
proc robustreg data=Stack plot=ddplot;
   model y = x1 x2 x3;
run;
```

The reference lines represent the cutoff values. The diagonal line is also drawn to show the distribution of the distances. By default, all outliers and leverage points are labeled with observation numbers. To change the default, you can use the LABEL= option as described in Table 107.2.

If you specify ID variables in the ID statement, the values of the first ID variable instead of observation numbers are used as labels.

Residual-Distance Plot

The residual-distance (RD) plot is used for both outlier and leverage-point diagnostics. It is a scatter plot of the standardized robust residuals against the robust distances. For more information about the robust distance, see the section "Leverage-Point and Outlier Detection" on page 9006.

You can use the PLOT=RDPLOT option to request the RD plot. The following statements use the Stack data set in the section "M Estimation" on page 8957 to create the plot shown in Figure 107.4:

```
proc robustreg data=Stack plot=rdplot;
   model y = x1 x2 x3;
run;
```

The reference lines represent the cutoff values. By default, all outliers and leverage points are labeled with observation numbers. To change the default, you can use the LABEL= option as described in Table 107.2.

If you specify ID variables in the ID statement, the values of the first ID variable instead of observation numbers are used as labels.

Histogram and Q-Q Plot

PROC ROBUSTREG produces a histogram and a Q-Q plot for the standardized robust residuals. The histogram is superimposed with a normal density curve and a kernel density curve. The following statements use the Stack data set from the section "M Estimation" on page 8957 to create the plots in Figure 107.6 and Figure 107.7:

```
proc robustreg data=Stack plots=(histogram qqplot);
   model y = x1 x2 x3;
run;
```

Examples: ROBUSTREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 107.1: Comparison of Robust Estimates

(View the complete code for this example (rregex1.sas) in the example repository.)

This example contrasts several of the robust methods available in the ROBUSTREG procedure.

The following statements generate 1,000 random observations. The first 900 observations are from a linear model, and the last 100 observations are significantly biased in the Y direction. In other words, 10% of the observations are contaminated with outliers.

```
data a (drop=i);
   do i=1 to 1000;
      x1=rannor(1234);
      x2=rannor(1234);
      e=rannor(1234);
      if i > 900 then y=100 + e;
      else y=10 + 5*x1 + 3*x2 + .5 * e;
      output;
   end;
run;
```

The following statements invoke PROC REG and PROC ROBUSTREG with the data set a:

```
proc reg data=a;
   model y = x1 x2;
run;

proc robustreg data=a method=m;
   model y = x1 x2;
run;

proc robustreg data=a method=mm seed=100;
   model y = x1 x2;
run;

proc robustreg data=a method=s seed=100;
   model y = x1 x2;
run;
```

```
proc robustreg data=a method=lts seed=100;
   model y = x1 x2;
run;
```

The tables of parameter estimates that are generated by using M estimation, MM estimation, S estimation, and LTS estimation in the ROBUSTREG procedure are shown in Output 107.1.2, Output 107.1.3, Output 107.1.4, and Output 107.1.5, respectively. For comparison, the ordinary least squares (OLS) estimates that are produced by the REG procedure (see Chapter 105, "The REG Procedure") are shown in Output 107.1.1. The four robust methods, M, MM, S, and LTS, correctly estimate the regression coefficients for the underlying model (10, 5, and 3), but the OLS estimate does not.

Output 107.1.1 OLS Estimates for Data with 10% Contamination

The REG Procedure
Model: MODEL1
Dependent Variable: y

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	19.06712	0.86322	22.09	<.0001
x1	1	3.55485	0.86892	4.09	<.0001
x2	1	2.12341	0.83039	2.56	0.0107

Output 107.1.2 M Estimates for Data with 10% Contamination

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.A
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	M Estimation

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	10.0024	0.0174	9.9683	10.0364	331908	<.0001
x1	1	5.0077	0.0175	4.9735	5.0420	82106.9	<.0001
x2	1	3.0161	0.0167	2.9834	3.0488	32612.5	<.0001
Scale	1	0.5780					

Output 107.1.3 MM Estimates for Data with 10% Contamination

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.A
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	MM Estimation

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	10.0035	0.0176	9.9690	10.0379	323947	<.0001
x1	1	5.0085	0.0178	4.9737	5.0433	79600.6	<.0001
x2	1	3.0181	0.0168	2.9851	3.0511	32165.0	<.0001
Scale	0	0.6733					

Output 107.1.4 S Estimates for Data with 10% Contamination

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.A
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	S Estimation

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	10.0055	0.0180	9.9703	10.0408	309917	<.0001
x1	1	5.0096	0.0182	4.9740	5.0452	76045.2	<.0001
x2	1	3.0210	0.0172	2.9873	3.0547	30841.3	<.0001
Scale	0	0.6721					

Output 107.1.5 LTS Estimates for Data with 10% Contamination

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.A
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	LTS Estimation

Output 107.1.5 *continued*

LTS Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	10.0083
x1	1	5.0316
x2	1	3.0396
Scale (sLTS)	0	0.5880
Scale (Wscale)	0	0.5113

The next statements demonstrate that if the percentage of contamination is increased to 40%, the M method and the MM method with default options fail to estimate the underlying model. Output 107.1.6 and Output 107.1.7 display these estimates. However, by tuning the constant *c* for the M method and the constants INITH and K0 for the MM method, you can increase the breakdown values of the estimates and capture the right model. Output 107.1.8 and Output 107.1.9 display these estimates. Similarly, you can tune the constant EFF for the S method and the constant H for the LTS method and correctly estimate the underlying model by using these methods. Results are not presented.

```
data b (drop=i);
   do i=1 to 1000;
      x1=rannor(1234);
      x2=rannor(1234);
      e=rannor(1234);
      if i > 600 then y=100 + e;
      else y=10 + 5*x1 +  3*x2 + .5 * e;
      output;
   end;
run;

proc robustreg data=b method=m;
   model y = x1 x2;
run;

proc robustreg data=b method=mm;
   model y = x1 x2;
run;

proc robustreg data=b method=m(wf=bisquare(c=2));
   model y = x1 x2;
run;

proc robustreg data=b method=mm(inith=502 k0=1.8);
   model y = x1 x2;
run;
```

Output 107.1.6 M Estimates (Default Setting) for Data with 40% Contamination

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.B
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	M Estimation

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	44.8991	1.5609	41.8399	47.9584	827.46	<.0001
x1	1	2.4309	1.5712	-0.6485	5.5104	2.39	0.1218
x2	1	1.3742	1.5015	-1.5687	4.3171	0.84	0.3601
Scale	1	56.6342					

Output 107.1.7 MM Estimates (Default Setting) for Data with 40% Contamination

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.B
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	MM Estimation

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	43.0607	1.7978	39.5370	46.5844	573.67	<.0001
x1	1	2.7369	1.8140	-0.8185	6.2924	2.28	0.1314
x2	1	1.5211	1.7265	-1.8628	4.9049	0.78	0.3783
Scale	0	52.8496					

Output 107.1.8 M Estimates (Tuned) for Data with 40% Contamination

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.B
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	M Estimation

Output 107.1.8 *continued*

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	10.0137	0.0219	9.9708	10.0565	209688	<.0001
x1	1	4.9905	0.0220	4.9473	5.0336	51399.1	<.0001
x2	1	3.0399	0.0210	2.9987	3.0811	20882.4	<.0001
Scale	1	1.0531					

Output 107.1.9 MM Estimates (Tuned) for Data with 40% Contamination

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.B
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	MM Estimation

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	10.0103	0.0213	9.9686	10.0520	221639	<.0001
x1	1	4.9890	0.0218	4.9463	5.0316	52535.9	<.0001
x2	1	3.0363	0.0201	2.9970	3.0756	22895.5	<.0001
Scale	0	1.8992					

When there are bad leverage points, the M method fails to estimate the underlying model no matter what constant c you use. In this case, other methods (LTS, S, and MM) in PROC ROBUSTREG, which are robust to bad leverage points, correctly estimate the underlying model.

The following statements generate and analyze 1,000 observations, 1% of which are bad high-leverage points.

```
data c (drop=i);
   do i=1 to 1000;
      x1=rannor(1234);
      x2=rannor(1234);
      e=rannor(1234);
      if i > 600 then y=100 + e;
      else y=10 + 5*x1 + 3*x2 + .5 * e;
      if i < 11 then x1=200 * rannor(1234);
      if i < 11 then x2=200 * rannor(1234);
      if i < 11 then y= 100*e;
      output;
   end;
run;
```

```
proc robustreg data=c method=mm(inith=502 k0=1.8) seed=100;
   model y = x1 x2;
run;

proc robustreg data=c method=s(k0=1.8) seed=100;
   model y = x1 x2;
run;

proc robustreg data=c method=lts(h=502) seed=100;
   model y = x1 x2;
run;
```

Output 107.1.10 displays the MM estimates with initial LTS estimates, Output 107.1.11 displays the S estimates, and Output 107.1.12 displays the LTS estimates.

Output 107.1.10 MM Estimates for Data with 1% Leverage Points

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.C
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	MM Estimation

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	9.9820	0.0215	9.9398	10.0241	215369	<.0001
x1	1	5.0303	0.0206	4.9898	5.0707	59469.1	<.0001
x2	1	3.0222	0.0221	2.9789	3.0655	18744.9	<.0001
Scale	0	2.2134					

Output 107.1.11 S Estimates for Data with 1% Leverage Points

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.C
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	S Estimation

Output 107.1.11 *continued*

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	9.9808	0.0216	9.9383	10.0232	212532	<.0001
x1	1	5.0303	0.0208	4.9896	5.0710	58656.3	<.0001
x2	1	3.0217	0.0222	2.9782	3.0652	18555.7	<.0001
Scale	0	2.2094					

Output 107.1.12 LTS Estimates for Data with 1% Leverage Points

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.C
Dependent Variable	y
Number of Independent Variables	2
Number of Observations	1000
Method	LTS Estimation

LTS Parameter Estimates		
Parameter	DF	Estimate
Intercept	1	9.9742
x1	1	5.0010
x2	1	3.0219
Scale (sLTS)	0	0.9952
Scale (Wscale)	0	0.5216

Example 107.2: Robust ANOVA

(View the complete code for this example (rregex2.sas) in the example repository.)

The classical analysis of variance (ANOVA) technique that is based on least squares assumes that the underlying experimental errors are normally distributed. However, data often contain outliers as a result of recording or other errors. In other cases, extreme responses occur when control variables in the experiments are set to extremes. It is important to distinguish among these extreme points and determine whether they are outliers or important extreme cases. You can use the ROBUSTREG procedure for robust analysis of variance based on M estimation. Usually there are no high-leverage points in a well-designed experiment, so M estimation is appropriate.

This example shows how to use the ROBUSTREG procedure for robust ANOVA.

An experiment studied the effects of two successive treatments (T1, T2) on the recovery time of mice that had certain diseases. Sixteen mice were randomly assigned to four groups for the four different combinations of the treatments. The recovery times (time) were recorded (in hours) as shown in the following data set:

```
data recover;
   input  T1 $ T2 $ time @@;
   datalines;
0 0 20.2  0 0 23.9  0 0 21.9  0 0 42.4
1 0 27.2  1 0 34.0  1 0 27.4  1 0 28.5
0 1 25.9  0 1 34.5  0 1 25.1  0 1 34.2
1 1 35.0  1 1 33.9  1 1 38.3  1 1 39.9
;
```

The following statements invoke the GLM procedure (see Chapter 53, "The GLM Procedure") for a standard ANOVA:

```
proc glm data=recover;
   class T1 T2;
   model time = T1 T2 T1*T2;
run;
```

Output 107.2.1 indicates that the overall model effect is not significant at the 10% level, and Output 107.2.2 indicates that neither treatment is significant at the 10% level.

Output 107.2.1 Overall ANOVA

The GLM Procedure

Dependent Variable: time

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	209.9118750	69.9706250	1.86	0.1905
Error	12	451.9225000	37.6602083		
Corrected Total	15	661.8343750			

R-Square	Coeff Var	Root MSE	time Mean
0.317167	19.94488	6.136791	30.76875

Output 107.2.2 Model ANOVA

Source	DF	Type I SS	Mean Square	F Value	Pr > F
T1	1	81.4506250	81.4506250	2.16	0.1671
T2	1	106.6056250	106.6056250	2.83	0.1183
T1*T2	1	21.8556250	21.8556250	0.58	0.4609

The following statements invoke the ROBUSTREG procedure and use the same model:

```
proc robustreg data=recover;
   class T1 T2;
   model time = T1 T2 T1*T2 / diagnostics;
   T1_T2: test T1*T2;
   output out=robout r=resid sr=stdres;
run;
```

Output 107.2.3 shows some basic information about the model and the response variable time.

Output 107.2.3 Model-Fitting Information and Summary Statistics

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.RECOVER
Dependent Variable	time
Number of Independent Variables	2
Number of Continuous Independent Variables	0
Number of CLASS Independent Variables	2
Number of Observations	16
Method	M Estimation

Summary Statistics

Variable	Q1	Median	Q3	Mean	Standard Deviation	MAD
time	25.5000	31.2000	34.7500	30.7688	6.6425	6.8941

The "Parameter Estimates" table in Output 107.2.4 indicates that the main effects of both treatments are significant at the 5% level.

Output 107.2.4 Model Parameter Estimates

Parameter Estimates

Parameter			DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept			1	36.7655	2.0489	32.7497	40.7814	321.98	<.0001
T1	**0**		1	-6.8307	2.8976	-12.5100	-1.1514	5.56	0.0184
T1	**1**		0	0.0000	.	.	.	.	.
T2	**0**		1	-7.6755	2.8976	-13.3548	-1.9962	7.02	0.0081
T2	**1**		0	0.0000	.	.	.	.	.
T1*T2	**0**	**0**	1	-0.2619	4.0979	-8.2936	7.7698	0.00	0.9490
T1*T2	**0**	**1**	0	0.0000	.	.	.	.	.
T1*T2	**1**	**0**	0	0.0000	.	.	.	.	.
T1*T2	**1**	**1**	0	0.0000	.	.	.	.	.
Scale			1	3.5346					

The reason for the difference between the traditional ANOVA and the robust ANOVA is explained by Output 107.2.5, which shows that the fourth observation is an outlier. Further investigation shows that the original value of 24.4 for the fourth observation was recorded incorrectly.

Output 107.2.5 Diagnostics

Diagnostics

Obs	Standardized Robust Residual	Outlier
4	5.7722	*

Output 107.2.6 displays the robust test results. The interaction between the two treatments is not significant.

Output 107.2.6 Test of Significance

Robust Linear Test T1_T2					
Test	Test Statistic	Lambda	DF	Chi-Square	Pr > ChiSq
Rho	0.0041	0.7977	1	0.01	0.9431
Rn2	0.0041		1	0.00	0.9490

Output 107.2.7 displays the robust residuals and standardized robust residuals.

Output 107.2.7 PROC ROBUSTREG Output

Obs	T1	T2	time	resid	stdres
1	0	0	20.2	-1.7974	-0.50851
2	0	0	23.9	1.9026	0.53827
3	0	0	21.9	-0.0974	-0.02756
4	0	0	42.4	20.4026	5.77222
5	1	0	27.2	-1.8900	-0.53472
6	1	0	34.0	4.9100	1.38911
7	1	0	27.4	-1.6900	-0.47813
8	1	0	28.5	-0.5900	-0.16693
9	0	1	25.9	-4.0348	-1.14152
10	0	1	34.5	4.5652	1.29156
11	0	1	25.1	-4.8348	-1.36785
12	0	1	34.2	4.2652	1.20668
13	1	1	35.0	-1.7655	-0.49950
14	1	1	33.9	-2.8655	-0.81070
15	1	1	38.3	1.5345	0.43413
16	1	1	39.9	3.1345	0.88679

Example 107.3: Growth Study of De Long and Summers

(View the complete code for this example (rregex3.sas) in the example repository.)

Robust regression and outlier detection techniques have considerable applications to econometrics. This example, from Zaman, Rousseeuw, and Orhan (2001), shows how these techniques substantially improve the ordinary least squares (OLS) results for the growth study of De Long and Summers.

De Long and Summers (1991) studied the national growth of 61 countries from 1960 to 1985 by applying OLS to the following data set:

```
data growth;
   input country $ GDP LFG EQP NEQ GAP @@;
   datalines;
Argentin  0.0089 0.0118 0.0214 0.2286 0.6079
Austria   0.0332 0.0014 0.0991 0.1349 0.5809
Belgium   0.0256 0.0061 0.0684 0.1653 0.4109
Bolivia   0.0124 0.0209 0.0167 0.1133 0.8634
Botswana  0.0676 0.0239 0.1310 0.1490 0.9474
```

```
   ... more lines ...

Venezuel  0.0120 0.0378 0.0340 0.0760 0.4974
Zambia   -0.0110 0.0275 0.0702 0.2012 0.8695
Zimbabwe  0.0110 0.0309 0.0843 0.1257 0.8875
;
```

The regression equation that they used is

$$\mathrm{GDP} = \beta_0 + \beta_1 \mathrm{LFG} + \beta_2 \mathrm{GAP} + \beta_3 \mathrm{EQP} + \beta_4 \mathrm{NEQ} + \epsilon$$

where the response variable is the growth in gross domestic product per worker (GDP) and the regressors are labor force growth (LFG), relative GDP gap (GAP), equipment investment (EQP), and nonequipment investment (NEQ).

The following statements invoke the REG procedure (see Chapter 105, "The REG Procedure") for the OLS analysis:

```
proc reg data=growth;
   model GDP  = LFG GAP EQP NEQ;
run;
```

The OLS analysis that is shown in Output 107.3.1 indicates that GAP and EQP have a significant influence on GDP at the 5% level.

Output 107.3.1 OLS Estimates

The REG Procedure
Model: MODEL1
Dependent Variable: GDP

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	-0.01430	0.01028	-1.39	0.1697
LFG	1	-0.02981	0.19838	-0.15	0.8811
GAP	1	0.02026	0.00917	2.21	0.0313
EQP	1	0.26538	0.06529	4.06	0.0002
NEQ	1	0.06236	0.03482	1.79	0.0787

The following statements invoke the ROBUSTREG procedure and use the default M estimation:

```
ods graphics on;

proc robustreg data=growth plots=all;
   model GDP  = LFG GAP EQP NEQ / diagnostics leverage;
   id country;
run;

ods graphics off;
```

Output 107.3.2 displays model information and summary statistics for variables in the model.

Output 107.3.2 Model-Fitting Information and Summary Statistics

The ROBUSTREG Procedure

Model Information	
Data Set	WORK.GROWTH
Dependent Variable	GDP
Number of Independent Variables	4
Number of Observations	61
Method	M Estimation

Summary Statistics

Variable	Q1	Median	Q3	Mean	Standard Deviation	MAD
LFG	0.0118	0.0239	0.0281	0.0211	0.00979	0.00949
GAP	0.5796	0.8015	0.8863	0.7258	0.2181	0.1778
EQP	0.0265	0.0433	0.0720	0.0523	0.0296	0.0325
NEQ	0.0956	0.1356	0.1812	0.1399	0.0570	0.0624
GDP	0.0121	0.0231	0.0310	0.0224	0.0155	0.0150

Output 107.3.3 displays the M estimates. Besides GAP and EQP, the robust analysis also indicates that NEQ is significant. This new finding is explained by Output 107.3.4, which shows that Zambia, the 60th country in the data, is an outlier. Output 107.3.4 also identifies leverage points that are based on the robust MCD distances; however, there are no serious high-leverage points in this data set.

Output 107.3.3 M Estimates

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-0.0247	0.0097	-0.0437	-0.0058	6.53	0.0106
LFG	1	0.1040	0.1867	-0.2619	0.4699	0.31	0.5775
GAP	1	0.0250	0.0086	0.0080	0.0419	8.36	0.0038
EQP	1	0.2968	0.0614	0.1764	0.4172	23.33	<.0001
NEQ	1	0.0885	0.0328	0.0242	0.1527	7.29	0.0069
Scale	1	0.0099					

Output 107.3.4 Diagnostics

Diagnostics						
Obs	country	Mahalanobis Distance	Robust MCD Distance	Leverage	Standardized Robust Residual	Outlier
1	Argentin	2.6083	4.0639	*	-0.9424	
5	Botswana	3.4351	6.7391	*	1.4200	
8	Canada	3.1876	4.6843	*	-0.1972	
9	Chile	3.6752	5.0599	*	-1.8784	
17	Finland	2.6024	3.8186	*	-1.7971	
23	HongKong	2.1225	3.8238	*	1.7161	
27	Israel	2.6461	5.0336	*	0.0909	
31	Japan	2.9179	4.7140	*	0.0216	
53	Tanzania	2.2600	4.3193	*	-1.8082	
57	U.S.	3.8701	5.4874	*	0.1448	
58	Uruguay	2.5953	3.9671	*	-0.0978	
59	Venezuel	2.9239	4.1663	*	0.3573	
60	Zambia	1.8562	2.7135		-4.9798	*
61	Zimbabwe	1.9634	3.9128	*	-2.5959	

Output 107.3.5 displays robust versions of goodness-of-fit statistics for the model.

Output 107.3.5 Goodness-of-Fit Statistics

Goodness-of-Fit	
Statistic	Value
R-Square	0.3178
AICR	80.2134
BICR	91.5095
Deviance	0.0070

The PLOTS=ALL option generates four diagnostic plots. Output 107.3.6 and Output 107.3.7 are for outlier and leverage-point diagnostics. Output 107.3.8 and Output 107.3.9 are a histogram and a Q-Q plot, respectively, of the standardized robust residuals.

Output 107.3.6 RD Plot for Growth Data

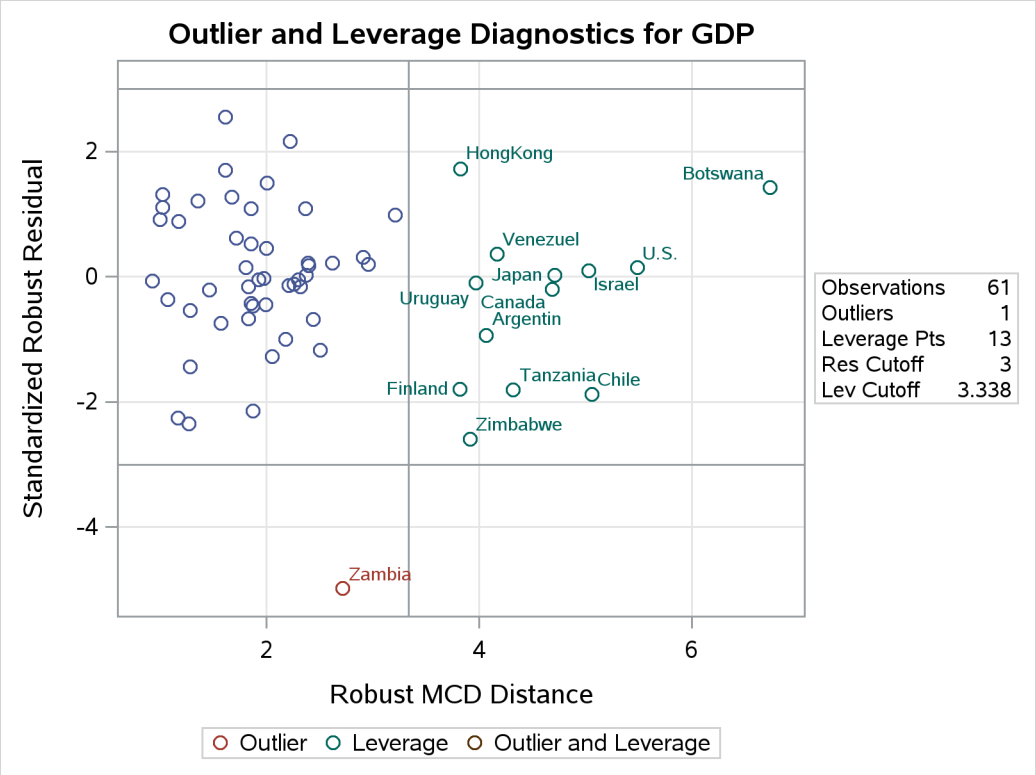

Output 107.3.7 DD Plot for Growth Data

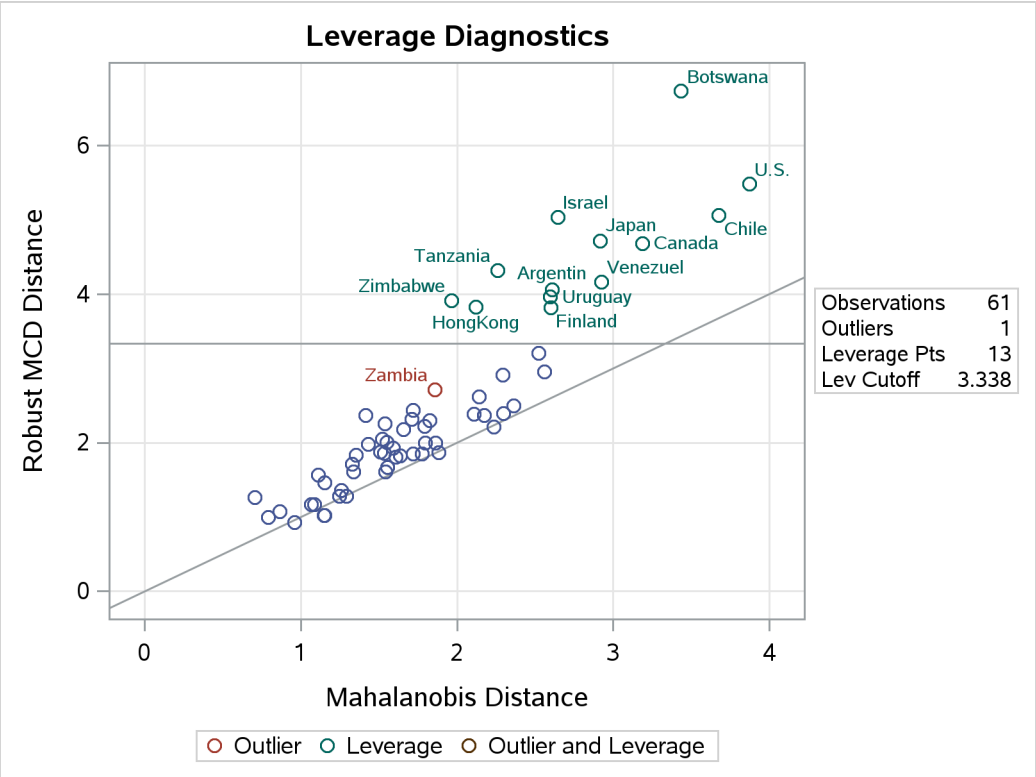

Output 107.3.8 Histogram

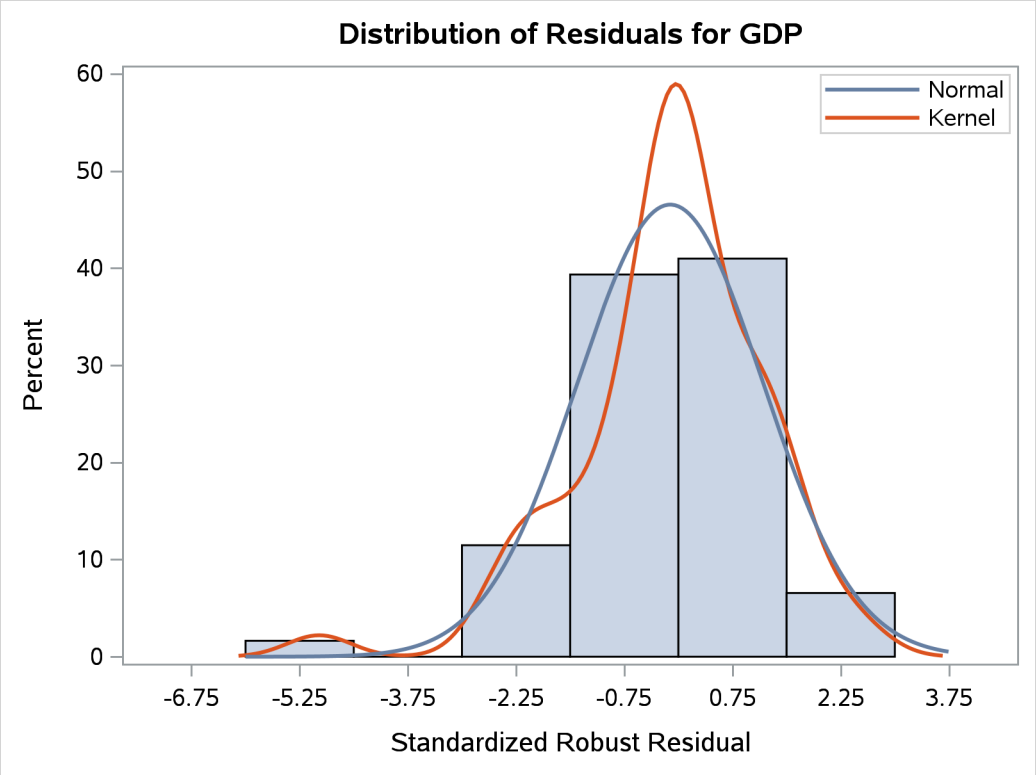

Output 107.3.9 Q-Q Plot

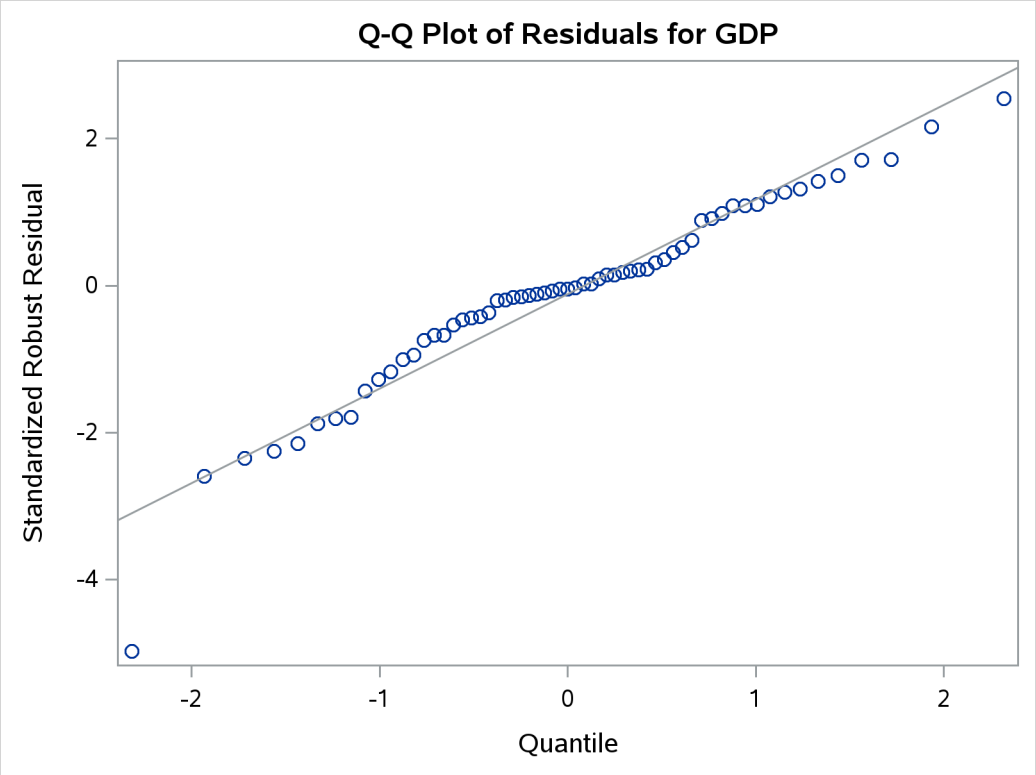

The following statements invoke the ROBUSTREG procedure and use LTS estimation, which was used by Zaman, Rousseeuw, and Orhan (2001). The results are consistent with those of M estimation.

```
proc robustreg method=lts(h=33) fwls data=growth seed=100;
   model GDP  = LFG GAP EQP NEQ / diagnostics leverage;
   id country;
run;
```

Output 107.3.10 displays the LTS estimates and the LTS R square.

Output 107.3.10 LTS Estimates and LTS R Square

The ROBUSTREG Procedure

LTS Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	-0.0249
LFG	1	0.1123
GAP	1	0.0214
EQP	1	0.2669
NEQ	1	0.1110
Scale (sLTS)	0	0.0076
Scale (Wscale)	0	0.0109

R-Square for LTS Estimation	
R-Square	0.7418

Output 107.3.11 displays outlier and leverage-point diagnostics that are based on the LTS estimates and the robust MCD distances.

Output 107.3.11 Diagnostics

Diagnostics						
Obs	**country**	**Mahalanobis Distance**	**Robust MCD Distance**	**Leverage**	**Standardized Robust Residual**	**Outlier**
1	Argentin	2.6083	4.0639	*	-1.0715	
5	Botswana	3.4351	6.7391	*	1.6574	
8	Canada	3.1876	4.6843	*	-0.2324	
9	Chile	3.6752	5.0599	*	-2.0896	
17	Finland	2.6024	3.8186	*	-1.6367	
23	HongKong	2.1225	3.8238	*	1.7570	
27	Israel	2.6461	5.0336	*	0.2334	
31	Japan	2.9179	4.7140	*	0.0971	
53	Tanzania	2.2600	4.3193	*	-1.2978	
57	U.S.	3.8701	5.4874	*	0.0605	
58	Uruguay	2.5953	3.9671	*	-0.0857	
59	Venezuel	2.9239	4.1663	*	0.4113	
60	Zambia	1.8562	2.7135		-4.4984	*
61	Zimbabwe	1.9634	3.9128	*	-2.1201	

Output 107.3.12 displays the final weighted least squares estimates, which are identical to those that are reported in Zaman, Rousseeuw, and Orhan (2001).

Output 107.3.12 Final Weighted LS Estimates

Parameter Estimates for Final Weighted Least Squares Fit							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-0.0222	0.0093	-0.0405	-0.0039	5.65	0.0175
LFG	1	0.0446	0.1771	-0.3026	0.3917	0.06	0.8013
GAP	1	0.0245	0.0082	0.0084	0.0406	8.89	0.0029
EQP	1	0.2824	0.0581	0.1685	0.3964	23.60	<.0001
NEQ	1	0.0849	0.0314	0.0233	0.1465	7.30	0.0069
Scale	0	0.0116					

Example 107.4: Constructed Effects

(View the complete code for this example (rregex4.sas) in the example repository.)

The algorithms of PROC ROBUSTREG assume that a response variable is linearly dependent on the regressors. However, in practice, a response often depends on some factors in a nonlinear manner. This example demonstrates how a nonlinear response-factor relationship can be modeled by using constructed effects. (For more information, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics.")

The following data set contains 526 female observations and 474 male observations that are sampled from the 2003 National Health and Nutrition Examination Survey (NHANES). Each observation is composed of three values, which correspond to the variables BMI (body mass index), Age, and Gender, measured for subjects between ages 20 and 60.

```
data BMI;
   input BMI Age Gender $ @@;
   datalines;
46.16 30.33 F 20.67 31.83 F 30.98 51.33 F 30.71 31.42 F
29.81 30.50 M 19.94 25.08 F 29.97 41.67 F 24.48 26.92 F
34.34 51.25 F 20.24 53.67 F 27.72 60.25 F 32.85 41.67 M
22.75 47.50 F 32.78 22.42 F 43.07 29.50 F 38.34 58.50 F
40.03 39.92 F 21.78 56.42 M 28.77 39.83 F 28.77 28.75 F
29.73 54.25 M 33.75 35.67 M 28.48 35.83 M 22.12 29.58 F

   ... more lines ...

26.98 42.50 F 29.44 39.75 M 25.60 52.67 F 19.30 22.00 F
26.53 27.92 F 23.77 29.00 F 29.86 60.58 M 25.41 44.08 M
26.53 24.83 M 33.33 42.08 F 30.52 32.50 F 31.89 38.17 F
32.20 35.92 F 21.73 26.67 M 32.10 39.33 M 25.13 51.75 M
;
```

The goal of this analysis is to evaluate whether the BMI by Age curves for women and men are different at a 5% significance level. In order to provide sufficient flexibility to model the effect of Age on BMI, you can use

regression splines that you define by using an EFFECT statement. In this example, a regression spline of degree 2 that has three knots is used for the variable Age. The knots are placed at the 25th, 50th, and 75th percentiles of Age. This analysis assume that there is no interaction between Gender and Age, so that the BMI by Age curves for women and men are the same up to a constant. The following statements produce the BMI by Age scatter plot shown in Output 107.4.1:

```
proc sort data=bmi;
   by age;
run;

proc sgplot data=bmi;
   scatter x=age y=bmi/group=gender;
run;
```

Output 107.4.1 Scatter Plot for BMI Data

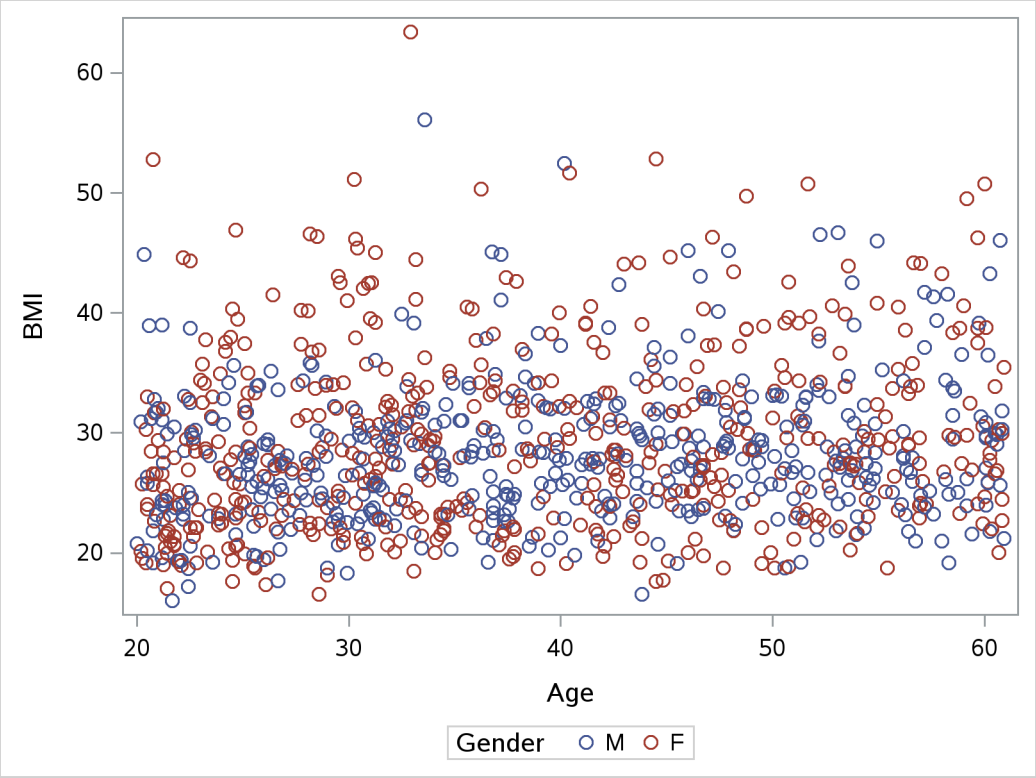

The observations that have large BMI values (for example, BMI > 40) are outliers that can substantially influence an ordinary least squares (OLS) analysis. Output 107.4.1 shows that the distributions of BMI conditional on Age are skewed toward the side of large BMI, and that there are more observations that have large BMI values (outliers) in the female group. Hence you can expect a significant Gender difference in the BMI by Age OLS regression analysis. This expectation is confirmed by the OLS Gender p-value 0.0059 in Output 107.4.2, which is produced by the following statements:

```
proc glmselect data=bmi;
   class gender;
   effect Age_Sp=spl(age/degree=2 knotmethod=percentiles(3));
   model bmi = gender age_sp /selection=none showpvalues;
```

```
   output out=out_ols P=pred R=res;
run;
```

Output 107.4.2 OLS Estimates

The GLMSELECT Procedure
Least Squares Model (No Selection)

Parameter Estimates

Parameter		DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept		1	29.890089	1.022825	29.22	<.0001
Gender	F	1	1.167332	0.422565	2.76	0.0058
Gender	M	0	0	.	.	.
Age_Sp	1	1	-4.404487	1.473761	-2.99	0.0029
Age_Sp	2	1	-3.329537	1.374096	-2.42	0.0156
Age_Sp	3	1	-0.966875	1.314964	-0.74	0.4623
Age_Sp	4	1	-1.611621	1.123854	-1.43	0.1519
Age_Sp	5	1	-0.484787	1.701281	-0.28	0.7757
Age_Sp	6	0	0	.	.	.

A robust regression method can reduce the outlier influence by automatically assigning smaller or even zero weights to outliers. For the BMI data, a robust regression method is likely to assign less weight to observations that have large BMI, so more female observations than male observations would receive smaller weights. The following statements invoke PROC ROBUSTREG and the BMI data set:

```
proc robustreg data=bmi method=s seed=100;
   class gender;
   effect Age_Sp=spl(age/degree=2 knotmethod=percentiles(3));
   model bmi = gender age_sp;
   output out=out_s P=pred R=res;
run;
```

Output 107.4.3 shows the parameter estimates and the diagnostics summary that are produced by PROC ROBUSTREG and the S method. In contrast to OLS, the robust p-value 0.5581 of the Gender coefficient indicates that the Gender effect is not significant. The outlier diagnostics that are based on the S estimates find 19 outliers that are assigned lower weights by the S method than by the OLS method.

Output 107.4.3 S Estimates and S Diagnostics Summary

The ROBUSTREG Procedure

Parameter Estimates

Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	28.2858	1.0081	26.3100	30.2616	787.33	<.0001
Gender	**F**	1	0.2409	0.4114	-0.5654	1.0473	0.34	0.5581
Gender	**M**	0	0.0000	.	.	.	.	.
Age_Sp	**1**	1	-3.8956	1.4376	-6.7133	-1.0779	7.34	0.0067
Age_Sp	**2**	1	-1.8692	1.3430	-4.5014	0.7630	1.94	0.1640
Age_Sp	**3**	1	-0.8336	1.2877	-3.3574	1.6903	0.42	0.5174
Age_Sp	**4**	1	-0.2329	1.1055	-2.3997	1.9338	0.04	0.8331
Age_Sp	**5**	1	0.0055	1.6632	-3.2543	3.2652	0.00	0.9974
Age_Sp	**6**	0	0.0000	.	.	.	.	.
Scale		0	6.1715					

Diagnostics Summary

Observation Type	Proportion	Cutoff
Outlier	0.0190	3.0000

To further compare the OLS and S outputs, the following statements plot the BMI predictions in the variable Age for both methods in the same graph, which is shown in Output 107.4.4:

```
data out2_s;
   set out_s;
   if gender="F" then label="F, S  ";
   if gender="M" then label="M, S  ";
run;

data out2_ols;
   merge bmi out_ols;
   if gender='F' then label='F, OLS';
   if gender='M' then label='M, OLS';
   keep pred bmi gender age label;
run;

data out2;
   set out2_ols out2_s;
run;

proc sgplot data=out2;
   series x=age y=pred/group=label;
run;
```

Output 107.4.4 OLS and S Predictions

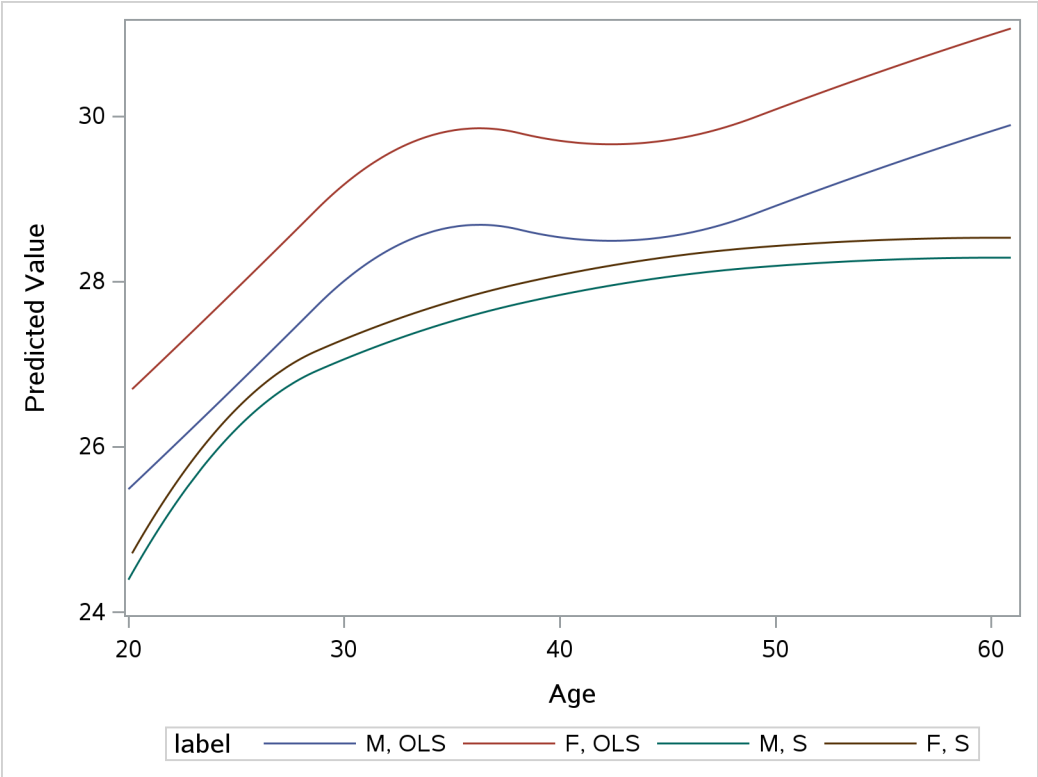

You can observe the following differences between the OLS and S predictions:

- The OLS prediction is larger.
- The OLS curves have a local maximum near Age = 35.

One question remains: is the significance of the Gender effect for the OLS regression due solely to the outlying observations? To tentatively answer this question, the following statements omit the observations that have the top 10% of BMI values from the original data set and reapply OLS and S methods to the reduced data set:

```
data three;
   set bmi;
   where bmi<38.315;
run;

proc robustreg data=three method=s seed=100;
   class gender;
   effect Age_Sp=spl(age/degree=2 knotmethod=percentiles(3));
   model bmi = gender age_sp;
   output out=out_s P=pred R=res;
run;
```

```
data out2_s;
   set out_s;
   if gender="F" then label="F, S  ";
   if gender="M" then label="M, S  ";
run;

proc glmselect data=three outdesign=four;
   class gender;
   effect Age_Sp=spl(age/degree=2 knotmethod=percentiles(3));
   model bmi = gender age_sp /selection=none showpvalues;
   output out=out_ols P=pred R=res;
run;

data out2_ols;
   merge three out_ols;
   if gender='F' then label='F, OLS';
   if gender='M' then label='M, OLS';
   keep pred bmi gender age label;
run;

data out2;
   set out2_ols out2_s;
run;

proc sgplot data=out2;
   series x=age y=pred/group=label;
run;
ods graphics off;
```

In the reduced data set, 71 female observations and 29 male observations are dropped. Output 107.4.5 and Output 107.4.6 show the refitted S and OLS parameter estimates, respectively.

Output 107.4.5 S Estimates

The ROBUSTREG Procedure

Parameter Estimates								
Parameter		DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept		1	27.6427	0.9741	25.7334	29.5520	805.23	<.0001
Gender	F	1	-0.2650	0.4023	-1.0535	0.5234	0.43	0.5100
Gender	M	0	0.0000	.	.	.	.	.
Age_Sp	1	1	-3.1859	1.4032	-5.9361	-0.4356	5.15	0.0232
Age_Sp	2	1	-1.5354	1.3051	-4.0934	1.0226	1.38	0.2394
Age_Sp	3	1	-0.3776	1.2499	-2.8273	2.0721	0.09	0.7626
Age_Sp	4	1	0.3299	1.0668	-1.7610	2.4208	0.10	0.7572
Age_Sp	5	1	0.0949	1.6221	-3.0845	3.2742	0.00	0.9534
Age_Sp	6	0	0.0000	.	.	.	.	.
Scale		0	4.9440					

Output 107.4.6 OLS Estimates

The GLMSELECT Procedure
Least Squares Model (No Selection)

Parameter Estimates

Parameter		DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept		1	27.841568	0.780817	35.66	<.0001
Gender	F	1	-0.040924	0.317749	-0.13	0.8976
Gender	M	0	0	.	.	.
Age_Sp	1	1	-3.253964	1.121292	-2.90	0.0038
Age_Sp	2	1	-0.975273	1.034172	-0.94	0.3459
Age_Sp	3	1	-0.508979	0.999609	-0.51	0.6108
Age_Sp	4	1	0.089393	0.852774	0.10	0.9165
Age_Sp	5	1	-0.113706	1.298157	-0.09	0.9302
Age_Sp	6	0	0	.	.	.

Output 107.4.7 displays the fitted curves on the reduced data set. You can see that Gender is no longer significant for the OLS model, and the OLS turning pattern has also disappeared, but the new S curves do not change much from the previous ones. The OLS BMI by Age curves in Output 107.4.7 are closer to the S curves than to the OLS curves in Output 107.4.4. This suggests that the difference between the OLS and S estimate results is indeed due solely to the influence of the outlying observations.

Output 107.4.7 OLS and S Predictions for the Reduced Data Set

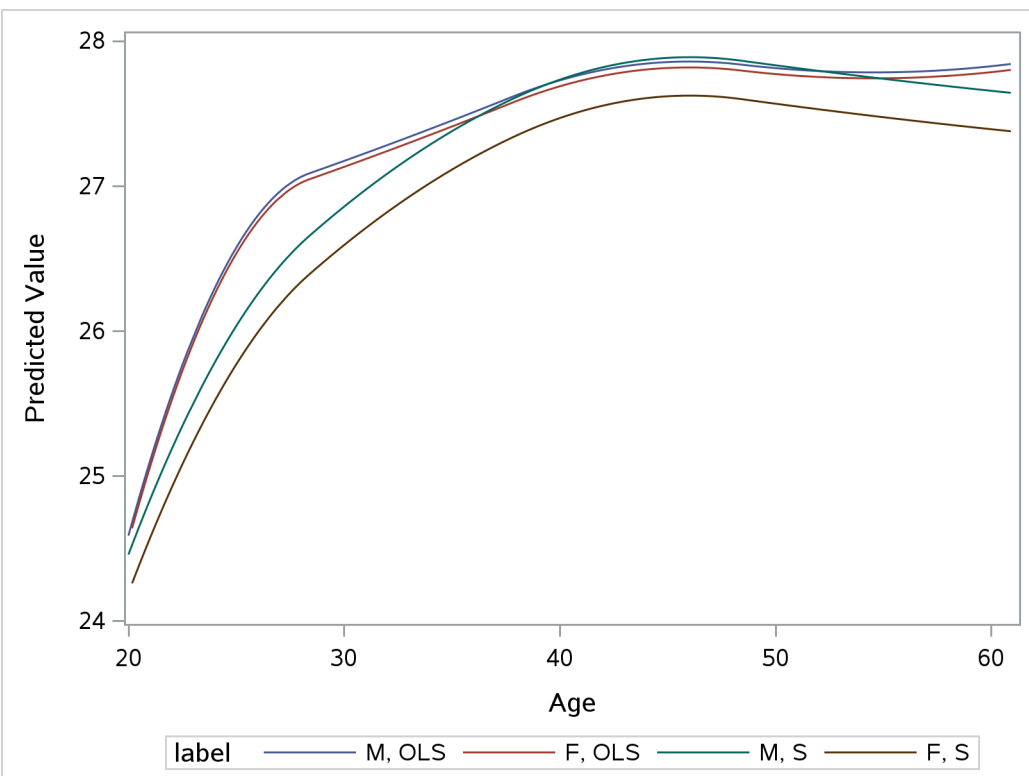

Example 107.5: Robust Diagnostics

(View the complete code for this example (rregex5.sas) in the example repository.)

This example models the selling price of a house as a function of several covariates. One of these covariates is a classification variable that indicates whether a house is located on a corner lot (called a corner house in this example). Because corner houses are relatively rare, the inclusion of this classification effect in the model introduces a low-dimensional structure (that is, the majority of the observations are located in a lower-dimensional hyperplane that is defined as containing non-corner houses) into the design matrix. As discussed in the section "Robust Distance" on page 9000, the presence of this low-dimensional structure causes difficulties in the traditional computation of robust distances. This example illustrates how you can use the projected robust distance to address those difficulties and to obtain meaningful leverage diagnostics. It also shows how you can use the RDPLOT= and DDPLOT= options to illustrate the outlier-leverage relationship.

The following house price data set contains 66 home resale records on seven variables from February 15 to April 30, 1993 (Data and Story Library 2005). The records are randomly selected from a database that is maintained by the Albuquerque Board of Realtors.

```
data house;
   input price sqft age feats ne cor tax @@;
   label price = "Selling price"
         sqft  = "Square feet of living space"
         age   = "Age of home in year"
         feats = "Number out of 11 features (dishwasher, refrigerator,
                  microwave, disposer, washer, intercom, skylight(s),
                  compactor, dryer, handicap fit, cable TV access)"
         ne    = "Located in northeast sector of city (1) or not (0)"
         cor   = "Corner location (1) or not (0)"
         tax   = "Annual taxes";
   sum = sqft+age+feats+ne+cor+tax;
   id  = _N_;
   datalines;
2050 2650 13 7 1 0 1639
2150 2664  6 5 1 0 1193
2150 2921  3 6 1 0 1635
1999 2580  4 4 1 0 1732

   ... more lines ...

 870 1273  4 4 0 0  638
 869 1165  7 4 0 0  694
 766 1200  7 4 0 1  634
 739  970  4 4 0 1  541
;
```

To illustrate the dependence detection ability of the generalized MCD algorithm, an extra variable called sum is created such that all the observations satisfy sum = sqft + age + feats + ne + cor + tax. Adding the variable sum does not change the rank of the original design matrix; sum is expected to be ignored in the model and also in the diagnostics. The following statements apply the MM method and the generalized MCD algorithm to the house price data:

```
ods graphics on;
proc robustreg data=house method=MM plots=all;
   model price = sqft age feats ne cor tax sum /
         leverage(opc mcdinfo) diagnostics;
run;
```

As shown in Output 107.5.1 and Output 107.5.2, PROC ROBUSTREG finds the design dependence equation and forces the parameter estimate of variable sum to be 0.

Output 107.5.1 MM Estimates

The ROBUSTREG Procedure

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	46.4062	79.1714	-108.767	201.5792	0.34	0.5578
sqft	1	0.3809	0.0756	0.2327	0.5291	25.37	<.0001
age	1	-2.6067	1.7610	-6.0582	0.8449	2.19	0.1388
feats	1	8.3627	14.7107	-20.4697	37.1951	0.32	0.5697
ne	1	65.0081	40.1329	-13.6508	143.6671	2.62	0.1053
cor	1	-19.2997	38.1907	-94.1520	55.5526	0.26	0.6133
tax	1	0.4699	0.1260	0.2229	0.7170	13.90	0.0002
sum	0	0.0000	.	.	.	.	.
Scale	0	157.5593					

Output 107.5.2 Design Dependence Equations

Note: The following variables have been ignored in the MCD computation because of linear dependence.

sum = sqft + age + feats + ne + cor + tax

Moreover, PROC ROBUSTREG also identifies a robust dependence equation on cor in Output 107.5.3, which holds for 77.27% of the observations but not for the entire data set.

Output 107.5.3 Robust Dependence Equations

Note: The following robust dependence equations simultaneously hold for 77.27% of the observations in the data set. The breakdown setting for the MCD algorithm is 22.73 %.

cor = 0

Another way to represent the low-dimensional structure is to specify the coefficients of the MCD-dropped components on the data (see Output 107.5.4), which form a basis of the complementary space to the relevant low-dimensional hyperplane.

Output 107.5.4 Coefficients of MCD-Dropped Components

Coefficients for MCD-Dropped Components		
Parameter	**DesignDrop0**	**RobustDrop1**
sqft	0	0
age	0	0
feats	0	0
ne	0	0
cor	0	1.0000
tax	0	0
sum	1.0000	0

By the definitions of projected robust distance and leverage point, an observation is called an off-plane leverage point if at least one of the robust or design dependence equations does not apply to the observation. In this example, the observations in which cor = 1 are all off-plane leverage points. Output 107.5.5 lists the leverage points and outliers along with the relevant distance measurements and standardized residuals.

Output 107.5.5 Diagnostics

Diagnostics						
	Projected Distance					
Obs	**Mahalanobis**	**Robust**	**Off-Plane**	**Leverage**	**Standardized Robust Residual**	**Outlier**
1	3.5567	4.0211	0.0000	*	0.8522	
13	4.0034	5.2310	0.0000	*	0.1411	
15	1.3221	1.5219	2.3681	*	0.0226	
16	1.0839	1.0905	2.3681	*	0.4148	
18	1.9452	2.4655	2.3681	*	-0.2789	
20	3.6006	4.0771	2.3681	*	-0.0150	
22	3.0210	3.4307	2.3681	*	1.1664	
23	1.5920	1.8197	2.3681	*	0.2422	
24	3.4967	4.5154	0.0000	*	0.6464	
26	3.0420	3.6975	0.0000	*	-1.7068	
29	2.3264	2.9925	2.3681	*	-2.4980	
30	1.2587	1.2714	2.3681	*	-1.2558	
38	2.4064	2.7249	2.3681	*	-1.0620	
42	1.4722	1.4645	2.3681	*	0.2584	
44	2.8491	3.0019	0.0000		4.5665	*
46	3.9725	5.2271	0.0000	*	3.5835	*
47	2.9431	3.3728	2.3681	*	0.1365	
55	2.2325	2.9590	2.3681	*	0.3217	
56	1.7999	1.8119	2.3681	*	0.1715	
65	1.8831	2.1822	2.3681	*	-0.1990	
66	2.2483	2.5673	2.3681	*	0.4134	

From Output 107.5.6 and Output 107.5.7, you can see that there is no apparent corner-related difference for the houses in terms of standardized robust residual and projected MD versus projected RD, although all the corner houses are defined as off-plane leverage points.

Output 107.5.6 Projected RD Plot

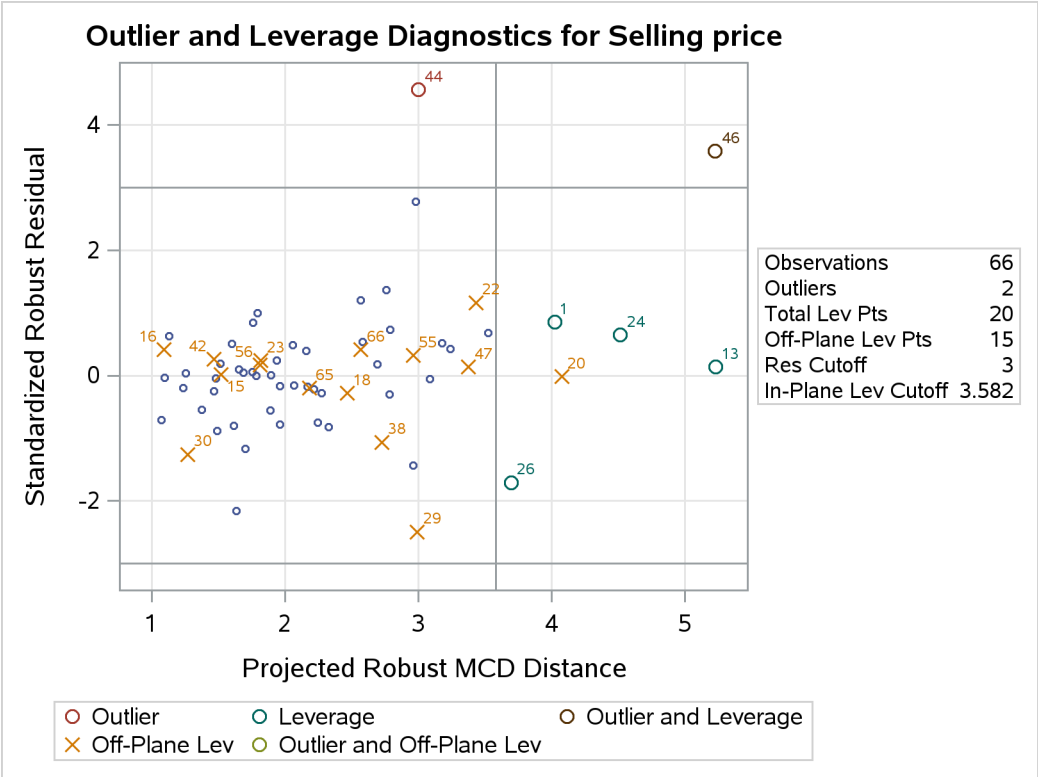

Output 107.5.7 Projected DD Plot

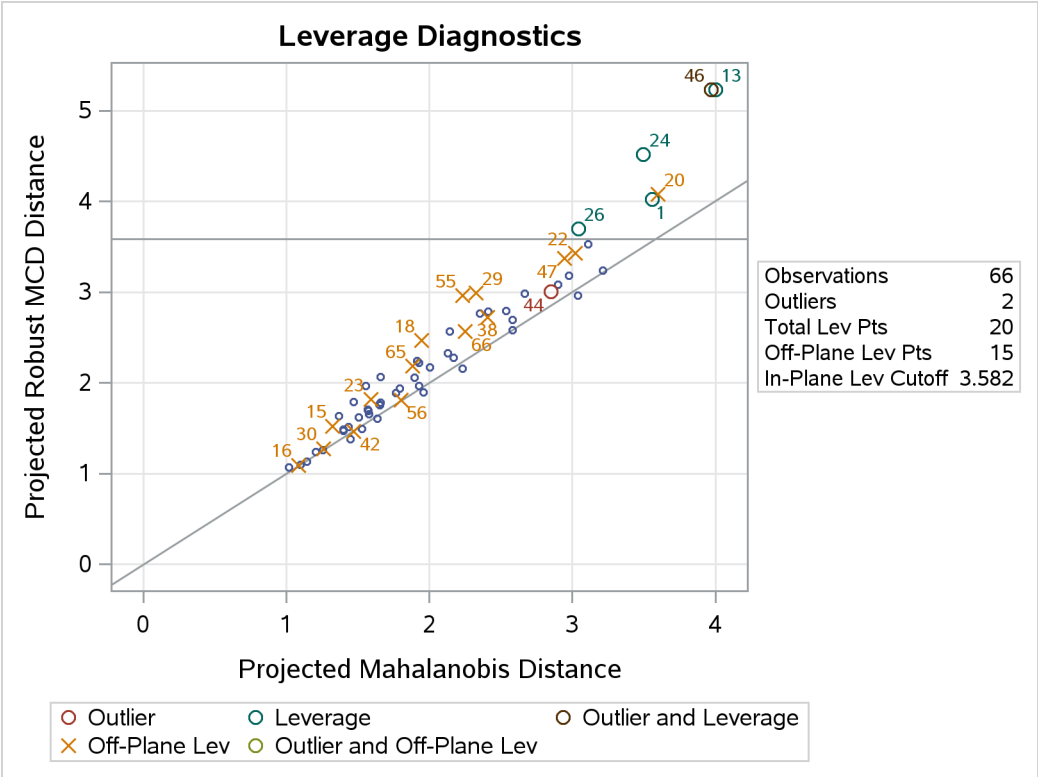

Output 107.5.8 shows more details of the robust diagnostics. The number of dimensions indicates that six regressors are used in the MCD analysis. Because sum is excluded in model fitting, it is ignored in the MCD analysis. The number of robust dropped components equals 1 because cor = 1. The number of off-plane points implies the 15 corner-house observations. The reweighted value of H is the number of observations that are finally used to estimate the MCD covariance.

Output 107.5.8 MCD Information

MCD Profile	
Number of Dimensions	6
Number of Robust Dropped Components	1
Number of Observations	66
Number of Off-Plane Observations	15
Specified Value of H	51
Reweighted Value of H	47
Breakdown Value	0.2273

MCD Center		
Parameter Name	**Parameter**	**Center**
sqft	**sqft**	1752.7
age	**age**	12.809
feats	**feats**	4.0426
ne	**ne**	0.6170
cor	**cor**	-2E-16
tax	**tax**	895.40
sum	**sum**	2665.6

MCD Covariance							
	sqft	**age**	**feats**	**ne**	**cor**	**tax**	**sum**
sqft	248870.3	-853.232	147.0347	88.60083	0	148494.5	396747.3
age	-853.232	126.2886	-1.18733	1.229417	0	-1251.44	-1978.34
feats	147.0347	-1.18733	0.99815	0.234043	0	87.0259	361.5814
ne	88.60083	1.229417	0.234043	0.241443	0	45.76688	134.42
cor	0	0	0	0	0	0	0
tax	148494.5	-1251.44	87.0259	45.76688	0	106652.5	255147
sum	396747.3	-1978.34	361.5814	134.42	0	255147	650413.7

MCD Correlation							
	sqft	**age**	**feats**	**ne**	**cor**	**tax**	**sum**
sqft	1	-0.15219	0.295009	0.361446	0	0.911462	0.986126
age	-0.15219	1	-0.10575	0.222643	0	-0.34099	-0.21829
feats	0.295009	-0.10575	1	0.476749	0	0.266726	0.448759
ne	0.361446	0.222643	0.476749	1	0	0.285206	0.339204
cor	0	0	0	0	0	0	0
tax	0.911462	-0.34099	0.266726	0.285206	0	1	0.968747
sum	0.986126	-0.21829	0.448759	0.339204	0	0.968747	1

You might speculate that the projected MD and projected RD are equal to the regular MD and RD on the same data set without the variable cor. In fact, this is not true. (See Output 107.5.9 and Output 107.5.10 for the RD

plot and DD plot of the data set without cor.) When cor is included in the MODEL statement, it is omitted from the distance calculation, but it is still used for the initial orthonormalization step and the h-subset searching. In this example, inclusion of cor causes all the other covariates to be centered separately for corner houses and non-corner houses. However, without cor, the centering process does not distinguish corner houses from non-corner houses, and therefore the MCD algorithm can still be influenced by cor through the correlation between cor and other covariates. The following statements drop the variable cor and produce the RD plot and DD plot for the reduced model, which are shown in Output 107.5.9 and Output 107.5.10, respectively:

```
proc robustreg data=house method=MM plots=all;
   model price = sqft age feats ne tax/leverage(mcdinfo) diagnostics;
run;
ods graphics off;
```

Output 107.5.9 RD Plot for the Reduced Model

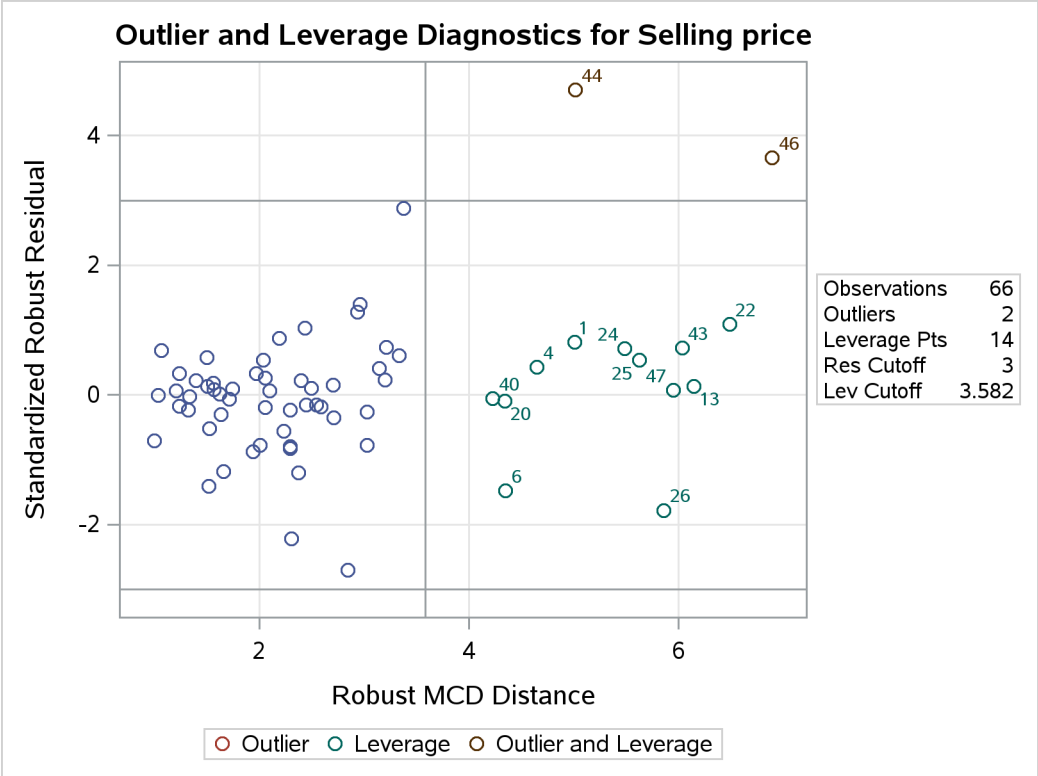

Output 107.5.10 DD Plot for the Reduced Model

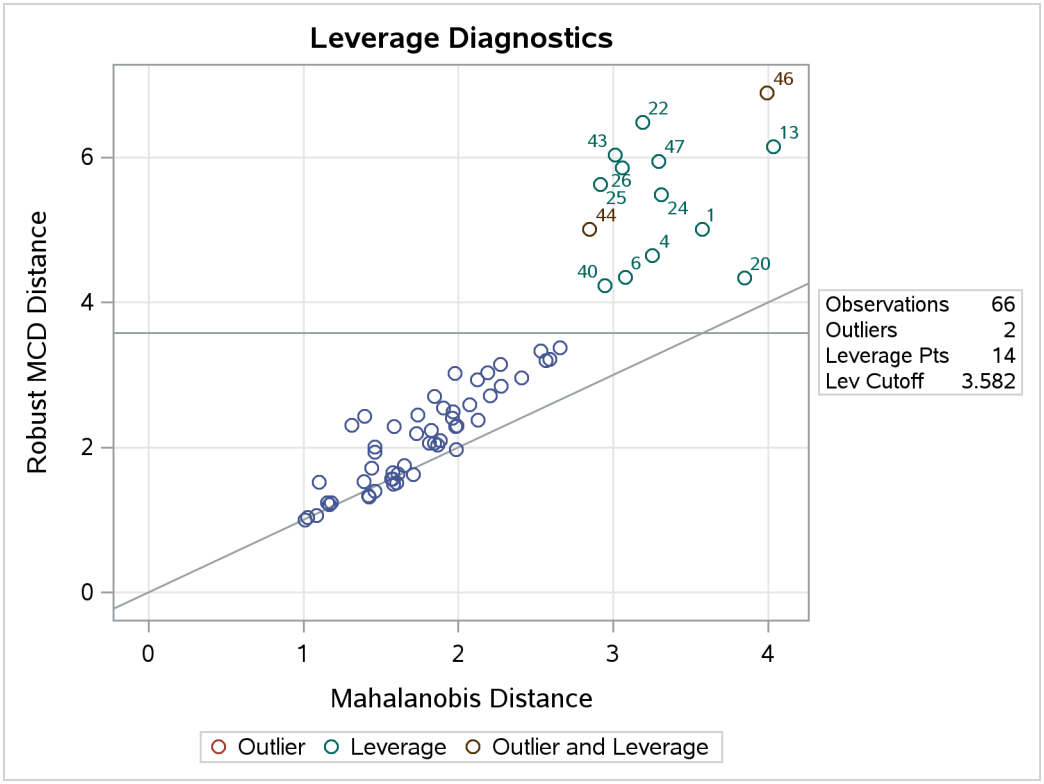

Compared with Output 107.5.8, Output 107.5.11 shows the changes of the MCD information by removing cor from the model. You can see that the corner houses are no longer identified as off-plane points and that the reweighted value of H is increased from 47 to 52. The breakdown value is intact because it depends only on the specified value of H and the total number of observations.

Output 107.5.11 MCD Information for the Reduced Model

MCD Profile	
Number of Dimensions	5
Number of Robust Dropped Components	0
Number of Observations	66
Number of Off-Plane Observations	0
Specified Value of H	51
Reweighted Value of H	52
Breakdown Value	0.2273

MCD Center		
Parameter Name	**Parameter**	**Center**
sqft	**sqft**	1710.9
age	**age**	11.173
feats	**feats**	3.9423
ne	**ne**	0.5962
tax	**tax**	858.10

Output 107.5.11 *continued*

MCD Covariance					
	sqft	age	feats	ne	tax
sqft	216974.7	681.2327	199.2492	103.0388	107503.1
age	681.2327	64.49887	-0.9506	1.855581	-187.135
feats	199.2492	-0.9506	0.878959	0.152715	114.9076
ne	103.0388	1.855581	0.152715	0.245475	49.98077
tax	107503.1	-187.135	114.9076	49.98077	66558.68

MCD Correlation					
	sqft	age	feats	ne	tax
sqft	1	0.182102	0.456255	0.44647	0.89457
age	0.182102	1	-0.12625	0.466337	-0.09032
feats	0.456255	-0.12625	1	0.328771	0.475075
ne	0.44647	0.466337	0.328771	1	0.391018
tax	0.89457	-0.09032	0.475075	0.391018	1

References

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Brownlee, K. A. (1965). *Statistical Theory and Methodology in Science and Engineering*. New York: John Wiley & Sons.

Chen, C. (2002). "Robust Regression and Outlier Detection with the ROBUSTREG Procedure." In *Proceedings of the Twenty-Seventh Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi27/p265-27.pdf.

Chen, C., and Yin, G. (2002). "Computing the Efficiency and Tuning Constants for M-Estimation." In *Proceedings of the 2002 Joint Statistical Meetings*, 478–482. Alexandria, VA: American Statistical Association.

Coleman, D. E., Holland, P. W., Kaden, N., Klema, V., and Peters, S. C. (1980). "A System of Subroutines for Iteratively Reweighted Least Squares Computations." *ACM Transactions on Mathematical Software* 6:327–336.

Data and Story Library (2005). "Home Prices." Carnegie Mellon University, Department of Statistics. Accessed July 22, 2011.

De Long, J. B., and Summers, L. H. (1991). "Equipment Investment and Economic Growth." *Quarterly Journal of Economics* 106:445–501.

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J., and Stahel, W. A. (1986). *Robust Statistics: The Approach Based on Influence Functions*. New York: John Wiley & Sons.

Hawkins, D. M., Bradu, D., and Kass, G. V. (1984). "Location of Several Outliers in Multiple Regression Data Using Elemental Sets." *Technometrics* 26:197–208.

Holland, P. W., and Welsch, R. E. (1977). "Robust Regression Using Iteratively Reweighted Least-Squares." *Communications in Statistics—Theory and Methods* 6:813–827.

Huber, P. J. (1973). "Robust Regression: Asymptotics, Conjectures, and Monte Carlo." *Annals of Statistics* 1:799–821.

Huber, P. J. (1981). *Robust Statistics*. New York: John Wiley & Sons.

Marazzi, A. (1993). *Algorithm, Routines, and S Functions for Robust Statistics*. Pacific Grove, CA: Wadsworth & Brooks/Cole.

Ronchetti, E. M. (1985). "Robust Model Selection in Regression." *Statistics and Probability Letters* 3:21–23.

Rousseeuw, P. J. (1984). "Least Median of Squares Regression." *Journal of the American Statistical Association* 79:871–880.

Rousseeuw, P. J., and Hubert, M. (1996). "Recent Development in PROGRESS." *Computational Statistics and Data Analysis* 21:67–85.

Rousseeuw, P. J., and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*. New York: John Wiley & Sons.

Rousseeuw, P. J., and Van Driessen, K. (1999). "A Fast Algorithm for the Minimum Covariance Determinant Estimator." *Technometrics* 41:212–223.

Rousseeuw, P. J., and Van Driessen, K. (2000). "An Algorithm for Positive-Breakdown Regression Based on Concentration Steps." In *Data Analysis: Scientific Modeling and Practical Application*, edited by W. Gaul, O. Opitz, and M. Schader, 335–346. New York: Springer-Verlag.

Rousseeuw, P. J., and Yohai, V. (1984). "Robust Regression by Means of S-Estimators." In *Robust and Nonlinear Time Series Analysis*, edited by J. Franke, W. Härdle, and R. D. Martin, 256–274. Vol. 26 of Lecture Notes in Statistics. Berlin: Springer-Verlag.

Ruppert, D. (1992). "Computing S Estimators for Regression and Multivariate Location/Dispersion." *Journal of Computational and Graphical Statistics* 1:253–270.

Yohai, V. J. (1987). "High Breakdown Point and High Efficiency Robust Estimates for Regression." *Annals of Statistics* 15:642–656.

Yohai, V. J., Stahel, W. A., and Zamar, R. H. (1991). "A Procedure for Robust Estimation and Inference in Linear Regression." In *Directions in Robust Statistics and Diagnostics, Part 2*, edited by W. A. Stahel and S. W. Weisberg, 365–374. New York: Springer-Verlag.

Yohai, V. J., and Zamar, R. H. (1997). "Optimal Locally Robust M-Estimates of Regression." *Journal of Statistical Planning and Inference* 64:309–323.

Zaman, A., Rousseeuw, P. J., and Orhan, M. (2001). "Econometric Applications of High-Breakdown Robust Regression Techniques." *Econometrics Letters* 71:1–8.

Chapter 108
The RSREG Procedure

Contents

Overview: RSREG Procedure **9050**
- Comparison to Other SAS Software 9050
- Terminology 9051

Getting Started: RSREG Procedure **9051**
- A Response Surface with a Simple Optimum 9051

Syntax: RSREG Procedure **9056**
- PROC RSREG Statement 9056
- BY Statement 9060
- ID Statement 9061
- MODEL Statement 9061
- RIDGE Statement 9064
- WEIGHT Statement 9065

Details: RSREG Procedure **9066**
- Introduction to Response Surface Experiments 9066
- Coding the Factor Variables 9068
- Missing Values 9068
- Plotting the Surface 9069
- Searching for Multiple Response Conditions 9069
- Handling Covariates 9071
- Computational Method 9072
- Output Data Sets 9073
- Displayed Output 9074
- ODS Table Names 9076
- ODS Graphics 9077

Examples: RSREG Procedure **9078**
- Example 108.1: A Saddle Surface Response Using Ridge Analysis 9078
- Example 108.2: Response Surface Analysis with Covariates 9082

References **9087**

Overview: RSREG Procedure

The RSREG procedure uses the method of least squares to fit quadratic response surface regression models. Response surface models are a kind of general linear model in which attention focuses on characteristics of the fit response function and in particular, where optimum estimated response values occur.

In addition to fitting a quadratic function, you can use the RSREG procedure to do the following:

- test for lack of fit
- test for the significance of individual factors
- analyze the canonical structure of the estimated response surface
- compute the ridge of optimum response
- predict new values of the response

The RSREG procedure uses ODS Graphics to display the response surfaces, residuals, fit diagnostics, and ridges of optimum response. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Comparison to Other SAS Software

Other SAS/STAT procedures can be used to fit the response surface, but the RSREG procedure is more specialized. PROC RSREG uses a much more compact model syntax than other procedures; for example, the following statements model a three-factor response surface in the REG, GLM, and RSREG procedures:

```
proc reg;
   model y=x1 x1*x1
           x2 x1*x2 x2*x2
           x3 x1*x3 x2*x3 x3*x3;
run;

proc glm;
   model y=x1|x2|x3@2 x1*x1 x2*x2 x3*x3;
run;

proc rsreg;
   model y=x1 x2 x3;
run;
```

Additionally, PROC RSREG includes specialized methodology for analyzing the fitted response surface, such as canonical analysis and optimum response ridges.

Terminology

Variables are referred to according to the following conventions:

factor variables — independent variables used to construct the quadratic response surface. To estimate the necessary parameters, each variable must have at least three distinct values in the data. Independent variables must be numeric.

response variables — the dependent variables to which the quadratic response surfaces are fit. Dependent variables must be numeric.

covariates — additional independent variables for use in the regression but not in the formation of the quadratic response surface. Covariates must be numeric.

WEIGHT variable — a variable for weighting the observations in the regression. The WEIGHT variable must be numeric.

ID variables — variables not previously described that are transferred to an output data set containing statistics for each observation in the input data set. This data set is created by using the OUT= option in the PROC RSREG statement. ID variables can be either character or numeric.

BY variables — variables for grouping observations. Separate analyses are obtained for each BY group. BY variables can be either character or numeric.

Getting Started: RSREG Procedure

A Response Surface with a Simple Optimum

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/rsrgs.sas.)

This example uses the three-factor quadratic model discussed in John (1971). Settings of the temperature, gas–liquid ratio, and packing height are controlled factors in the production of a certain chemical; Schneider and Stockett (1963) performed an experiment in order to determine the values of these three factors that minimize the unpleasant odor of the chemical. The following statements input the SAS data set smell; the variable Odor is the response, while the variables T, R, and H are the independent factors.

```
title 'Response Surface with a Simple Optimum';
data smell;
   input Odor T R H @@;
   label
      T = "Temperature"
      R = "Gas-Liquid Ratio"
      H = "Packing Height";
   datalines;
 66 40 .3 4     39 120 .3 4     43 40 .7 4     49 120 .7  4
 58 40 .5 2     17 120 .5 2     -5 40 .5 6    -40 120 .5  6
```

```
 65 80 .3 2      7  80 .7 2     43 80 .3 6    -22  80 .7  6
-31 80 .5 4    -35  80 .5 4    -26 80 .5 4
;
```

The following statements invoke PROC RSREG on the data set smell. Figure 108.1 through Figure 108.3 display the results of the analysis, including a lack-of-fit test requested with the LACKFIT option.

```
proc rsreg data=smell;
   model Odor = T R H / lackfit;
run;
```

Figure 108.1 displays the coding coefficients for the transformation of the independent variables to lie between –1 and 1, simple statistics for the response variable, hypothesis tests for linear, quadratic, and crossproduct terms, and the lack-of-fit test. The hypothesis tests can be used to gain a rough idea of importance of the effects; here the crossproduct terms are not significant. However, the lack of fit for the model is significant, so more complicated modeling or further experimentation with additional variables should be performed before firm conclusions are made concerning the underlying process.

Figure 108.1 Summary Statistics and Analysis of Variance

Response Surface with a Simple Optimum

The RSREG Procedure

Coding Coefficients for the Independent Variables		
Factor	**Subtracted off**	**Divided by**
T	80.000000	40.000000
R	0.500000	0.200000
H	4.000000	2.000000

Response Surface for Variable Odor	
Response Mean	15.200000
Root MSE	22.478508
R-Square	0.8820
Coefficient of Variation	147.8849

Regression	DF	Type I Sum of Squares	R-Square	F Value	Pr > F
Linear	3	7143.250000	0.3337	4.71	0.0641
Quadratic	3	11445	0.5346	7.55	0.0264
Crossproduct	3	293.500000	0.0137	0.19	0.8965
Total Model	9	18882	0.8820	4.15	0.0657

Residual	DF	Sum of Squares	Mean Square	F Value	Pr > F
Lack of Fit	3	2485.750000	828.583333	40.75	0.0240
Pure Error	2	40.666667	20.333333		
Total Error	5	2526.416667	505.283333		

Parameter estimates and the factor ANOVA are shown in Figure 108.2. Looking at the parameter estimates, you can see that the crossproduct terms are not significantly different from zero, as noted previously. The

Estimate column contains estimates based on the raw data, and the Parameter Estimate from Coded Data column contains estimates based on the coded data. The factor ANOVA table displays tests for all four parameters corresponding to each factor—the parameters corresponding to the linear effect, the quadratic effect, and the effects of the crossproducts with each of the other two factors. The only factor with a significant overall effect is R, indicating that the level of noise left unexplained by the model is still too high to estimate the effects of T and H accurately. This might be due to the lack of fit.

Figure 108.2 Parameter Estimates and Hypothesis Tests

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Parameter Estimate from Coded Data
Intercept	1	568.958333	134.609816	4.23	0.0083	-30.666667
T	1	-4.102083	1.489024	-2.75	0.0401	-12.125000
R	1	-1345.833333	335.220685	-4.01	0.0102	-17.000000
H	1	-22.166667	29.780489	-0.74	0.4902	-21.375000
T*T	1	0.020052	0.007311	2.74	0.0407	32.083333
R*T	1	1.031250	1.404907	0.73	0.4959	8.250000
R*R	1	1195.833333	292.454665	4.09	0.0095	47.833333
H*T	1	0.018750	0.140491	0.13	0.8990	1.500000
H*R	1	-4.375000	28.098135	-0.16	0.8824	-1.750000
H*H	1	1.520833	2.924547	0.52	0.6252	6.083333

Factor	DF	Sum of Squares	Mean Square	F Value	Pr > F	Label
T	4	5258.016026	1314.504006	2.60	0.1613	Temperature
R	4	11045	2761.150641	5.46	0.0454	Gas-Liquid Ratio
H	4	3813.016026	953.254006	1.89	0.2510	Packing Height

Figure 108.3 displays the canonical analysis and eigenvectors. The canonical analysis indicates that the directions of principal orientation for the predicted response surface are along the axes associated with the three factors, confirming the small interaction effect in the regression ANOVA (Figure 108.1). The largest eigenvalue (48.8588) corresponds to the eigenvector (0.238091, 0.971116, -0.015690), the largest component of which (0.971116) is associated with R; similarly, the second-largest eigenvalue (31.1035) is associated with T. The third eigenvalue (6.0377), associated with H, is quite a bit smaller than the other two, indicating that the response surface is relatively insensitive to changes in this factor. The coded form of the canonical analysis indicates that the estimated response surface is at a minimum when T and R are both near the middle of their respective ranges (that is, the coded critical values for T and R are both near 0) and H is relatively high; in uncoded terms, the model predicts that the unpleasant odor is minimized when T = 84.876502, R = 0.539915, and H = 7.541050.

Figure 108.3 Canonical Analysis and Eigenvectors

	Critical Value		
Factor	**Coded**	**Uncoded**	**Label**
T	0.121913	84.876502	Temperature
R	0.199575	0.539915	Gas-Liquid Ratio
H	1.770525	7.541050	Packing Height
Predicted value at stationary point: -52.024631			

	Eigenvectors		
Eigenvalues	**T**	**R**	**H**
48.858807	0.238091	0.971116	-0.015690
31.103461	0.970696	-0.237384	0.037399
6.037732	-0.032594	0.024135	0.999177
Stationary point is a minimum.			

To plot the response surface with respect to two of the factor variables, fix H, the least significant factor variable, at its estimated optimum value. The following statements use ODS Graphics to display the surface:

```
ods graphics on;
proc rsreg data=smell
           plots(unpack)=surface(3d at(H=7.541050));
   model Odor = T R H;
   ods select 'T * R = Pred';
run;
ods graphics off;
```

Note that the ODS SELECT statement is specified to select the plot of interest.

Figure 108.4 The Response Surface at the Optimum H

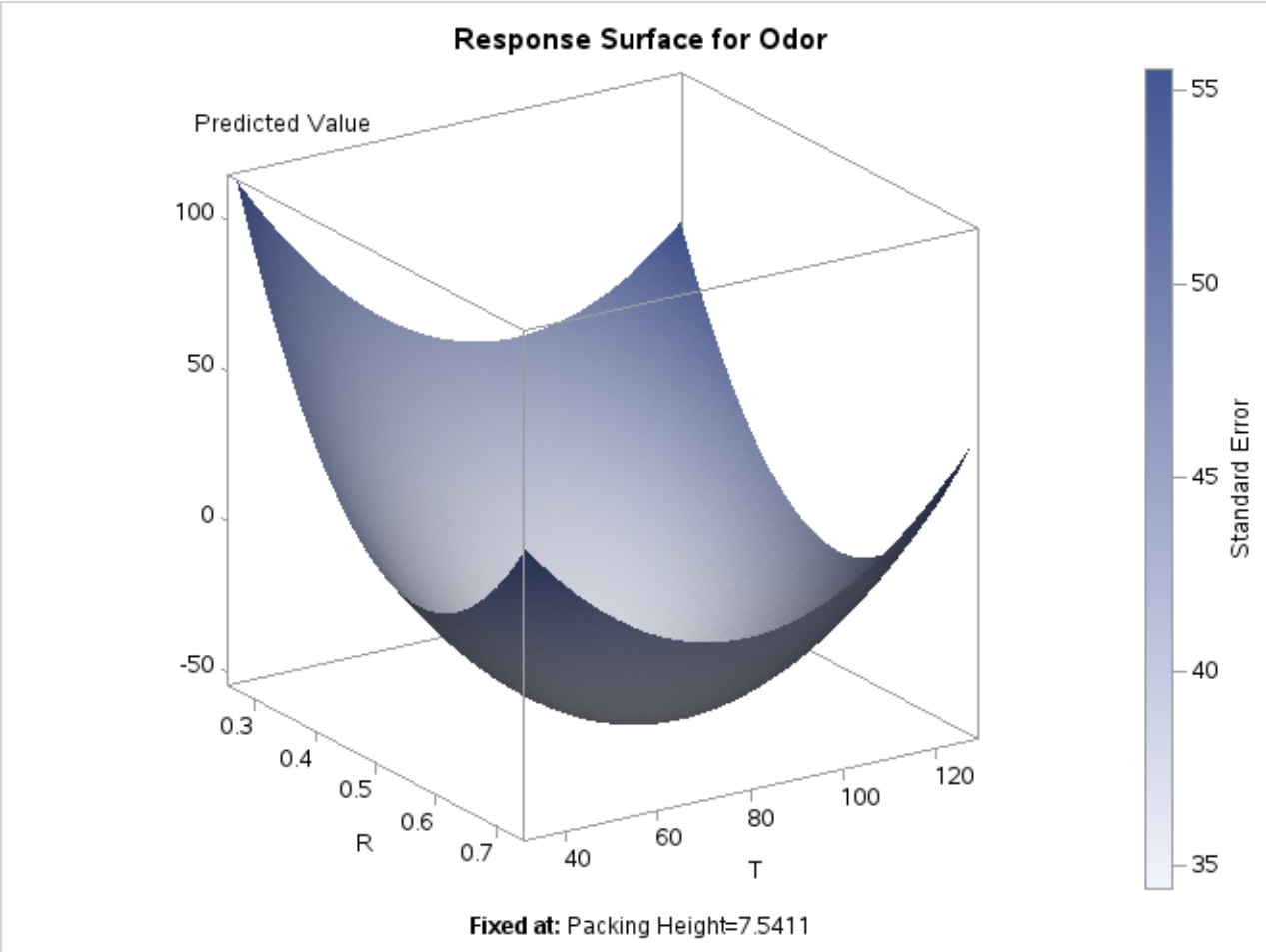

Alternatively, the following statements produce an output data set containing the surface information, which you can then use for plotting surfaces or searching for optima. The first DATA step fixes H, the least significant factor variable, at its estimated optimum value (7.541), and generates a grid of points for T and R. To ensure that the grid data do not affect parameter estimates, the response variable (Odor) is set to missing. (See the section "Missing Values" on page 9068.) The second DATA step concatenates these grid points to the original data. Then PROC RSREG computes predictions for the combined data. The last DATA step subsets the predicted values over just the grid points, which excludes the predictions at the original data.

```
data grid;
   do;
      Odor =  .  ;
      H    = 7.541;
      do T = 20 to 140 by 5;
         do R = .1 to .9 by .05;
            output;
         end;
      end;
   end;
run;
data grid;
```

```
   set smell grid;
run;

proc rsreg data=grid out=predict noprint;
   model Odor = T R H / predict;
run;

data grid;
   set predict;
   if H = 7.541;
run;
```

Syntax: RSREG Procedure

The following statements are available in the RSREG procedure.

PROC RSREG < *options* > **;**
  MODEL *responses* **=** *independents* < / *options* > **;**
  RIDGE < *options* > **;**
  WEIGHT *variable* **;**
  ID *variables* **;**
  BY *variables* **;**

The PROC RSREG and MODEL statements are required.

The BY, ID, MODEL, RIDGE, and WEIGHT statements are described after the PROC RSREG statement, and they can appear in any order.

PROC RSREG Statement

PROC RSREG < *options* > **;**

The PROC RSREG statement invokes the RSREG procedure. Table 108.1 summarizes the *options* available in the PROC RSREG statement.

Table 108.1 PROC RSREG Statement Options

Option	Description
DATA=	Names the input SAS data set
NOPRINT	Suppresses the normal display of results
OUT=	Creates the output SAS data set
PLOTS	Controls the plots produced through ODS Graphics

The following list describes these *options*.

DATA=*SAS-data-set*

specifies the input SAS data set that contains the data to be analyzed. By default, PROC RSREG uses the most recently created SAS data set.

NOPRINT

suppresses the normal display of results when only the output data set is required.

For more information, see the description of the NOPRINT option in the MODEL and RIDGE statements.

Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

OUT=*SAS-data-set*

creates an output SAS data set that contains statistics for each observation in the input data set. In particular, this data set contains the BY variables, the ID variables, the WEIGHT variable, the variables in the MODEL statement, and the output options requested in the MODEL statement. You must specify output statistic options in the MODEL statement; otherwise, the output data set is created but contains no observations. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For more details, see the section "OUT=SAS-data-set" on page 9073.

PLOTS<(*global-plot-option***)>=** *plot-request***<(***options***)>**

PLOTS<(*global-plot-option***)>=(***plot-request* **<(***options***)><** . . . *plot-request***<(***options***)>>)**

controls the plots produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses from around the *plot-request*. For example:

```
plots = all
plots = (diagnostics ridge surface(unpack))
plots(unpack) = surface(overlaypairs)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc rsreg plots=all;
   model y=x;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

By default, no graphs are created; you must specify the PLOTS= option to make graphs. See Figure 108.4, Output 108.1.5, Output 108.2.3, and Output 108.2.4 for examples of the ODS graphical displays.

The following *global-plot-option* is available.

UNPACKPANELS | UNPACK

suppresses paneling. By default, multiple plots can appear in some output *panels*. Specify the UNPACK option to display each plot separately.

The following *plot-requests* are available.

ALL

produces all appropriate plots. You can specify other options with ALL; for example, to display all plots and unpack the SURFACE contours you can specify `plots=(all surface(unpack))`.

DIAGNOSTICS < (LABEL | UNPACK) >

displays a panel of summary fit diagnostic plots. The plots produced and their usage are discussed in Table 108.2.

Table 108.2 Diagnostic Plots

Diagnostic Plot	Usage
Cook's *D* statistic versus observation number	Evaluate influence of an observation on the entire parameter estimate vector
Dependent variable values versus predicted values	Evaluate adequacy of fit and detect influential observations
Externally studentized residuals (RStudent) versus leverage	Detect outliers and influential (high-leverage) observations
Externally studentized residuals versus predicted values	Evaluate adequacy of fit and detect outliers
Histogram of residuals	Confirm normality of error terms
Normal quantile plot of residuals	Confirm normality and homogeneity of error terms, and detect outliers
Residuals versus predicted values	Evaluate adequacy of fit and detect outliers
Residual-fit (RF) spread plot	side-by-side quantile plots of the centered fit and the residuals show "how much variation in the data is explained by the fit and how much remains in the residuals" (Cleveland 1993)

Observations satisfying RStudent > 2 or RStudent < –2 are called *outliers*, and observations with leverage > $2p/n$ are called *influential*, where n is the number of observations used in fitting the model and p is the number of parameters used in the model (Rawlings, Pantula, and Dickey 1998). Specifying the LABEL option labels the influential and outlying observations—the label is the first ID variable if the ID statement is specified; otherwise, it is the observation number. Note in the Cook's *D* plot that only observations with *D* exceeding $4/n$ are labeled; these are also called influential observations. The UNPACK option displays each diagnostic plot separately. See Output 108.2.3 for an example of the diagnostics panel.

FIT < (GRIDSIZE=*number*) >
plots the predicted values against a single predictor when you have only one factor or only one covariate in the model. The GRIDSIZE= option specifies the number of points at which the fitted values are computed; by default, GRIDSIZE=200.

NONE
suppresses all plots.

RESIDUALS < (UNPACK | SMOOTH) >
displays plots of residuals against each factor and covariate. The UNPACK option displays each residual plot separately. The SMOOTH option overlays a loess smooth on each residual plot; see Chapter 78, "The LOESS Procedure," for more information. See Output 108.2.4 for an example of this plot.

RIDGE < (UNPACK) >
displays the maximum and/or minimum ridge plots. This option is available only when a MAXIMUM or MINIMUM option is specified in the RIDGE statement. The UNPACK option displays the estimated response and factor level ridge plots separately. See Output 108.1.5 for an example of this plot.

SURFACE < (*surface-options*) >
displays the response surface for each response variable and each pair of factors with all other factors and covariates fixed at their means. By default a panel of contour plots is produced; see Output 108.1.5 for an example of this plot. The following *surface-options* can be specified:

3D
displays three-dimensional surface plots instead of contour plots. See Figure 108.4 for an example of this plot.

AT < *keyword* >< (*variable=value-list* | *keyword* < . . . *variable=value-list* | *keyword* >) >
specifies fixed values for factors and covariates. You can specify one or more numbers in the *value-list* or one of the following *keywords*:

MIN	sets the variable to its minimum value.
MEAN	sets the variable to its mean value.
MIDRANGE	sets the variable to the middle value: $\frac{\max + \min}{2}$.
MAX	sets the variable to its maximum value.

Specifying a *keyword* immediately after AT sets the default value of all variables; for example, **`AT MIN`** sets all variables not displayed on an axis to their minimum values. By default, continuous variables are set to their means (**`AT MEAN`**) when they are not used on an axis. For example, if your model contains variables X1, X2, and X3, then specifying **`AT(X1=7 9)`** produces a contour plot of X2 versus X3 fixing X1 = 7 and then another contour plot with X1 = 9, along with contour plots of X1 versus X2 fixing X3 at its mean, and X1 versus X3 fixing X2 at its mean.

EXTEND=*value*
extends the surface *value*-times the range of each factor in each direction, which enables you to see more of the fitted surface. For example, if factor A has range [0, 10], then specifying `EXTEND=0.1` will compute and display the surface for A in [-1, 11]. You can specify *value* ≥ 0; by default, *value* = 0.1.

FILL=PRED | SE | NONE
produces a filled contour plot for either the predicted values or the standard errors. FILL=SE is the default. If the 3D option is also specified, then the contour plot is projected onto the surface.

GRIDSIZE=*n*
creates an $n \times n$ grid of points at which the estimated values for the surface and standard errors are computed, for $n \geq 1$. By default, $n = 50$.

LINE< =PRED | SE | NONE >
produces a contour line plot for either the predicted values or the standard errors. LINE=PRED is the default. If the 3D option is also specified, then specifying LINE displays a grid on the surface, and the other LINE= specifications are ignored.

NODESIGN
suppresses the display of the design points on the contour surface plots and the overlaid contour-line plots.

OVERLAYPAIRS
produces overlaid contour line plots for all pairs of response variables in addition to the contour surface plots. See Figure 108.6 for an example of this plot.

ROTATE=*angle*
rotates the 3-D surface plots *angle* degrees, $-180 < angle < 180$. By default, *angle* = 57.

TILT=*angle*
tilts the 3-D surface plots *angle* degrees, $-180 < angle < 180$. By default, *angle* = 20.

UNPACKPANELS | UNPACK
suppresses paneling, and displays each surface plot separately.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC RSREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the RSREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

ID Statement

ID *variables* **;**

The ID statement names variables that are to be transferred to the data set created by the OUT= option in the PROC RSREG statement.

MODEL Statement

MODEL *responses* **=** *independents* < / *options* > **;**

In the MODEL statement, you specify the response (dependent) variables followed by an equal sign and then the independent variables, some of which can be covariates.

Table 108.3 summarizes the *options* available in the MODEL statement. The output statistics specify which statistics are output to the OUT= data set. If none of the output statistics are specified, the data set is created but contains no observations. The keywords for the output statistics become values of the special variable _TYPE_ in the output data set.

Table 108.3 MODEL Statement Options

Option	Description
BYOUT	Uses only the first BY group to estimate the model
COVAR=	Declares variables to be simple linear regressors
LACKFIT	Performs a lack-of-fit test
NOCODE	Performs the canonical and ridge analyses using original values
PRESS	Displays the predicted residual sum of squares (PRESS) statistic
Suppress Displayed Output	
NOANOVA	Suppresses the analysis of variance and parameter estimates
NOOPTIMAL	Suppresses the canonical analysis
NOPRINT	Suppresses both the analysis of variance and the canonical analysis

Table 108.3 *continued*

Option	Description
Output Statistics	
ACTUAL	Includes observed response values
PREDICT	Includes values predicted by the model
RESIDUAL	Includes the residuals
L95	Includes the lower bound of a 95% confidence interval for an individual predicted value
U95	Includes the upper bound of a 95% confidence interval for an individual predicted value
L95M	Includes the lower bound of a 95% confidence interval for the expected value of the dependent variable
U95M	Includes the upper bound of a 95% confidence interval for the expected value of the dependent variable
D	Includes Cook's *D* influence statistic

The following list describes these *options* in alphabetical order.

ACTUAL

specifies that the observed response values from the input data set be written to the output data set.

BYOUT

uses only the first BY group to estimate the model. Subsequent BY groups have scoring statistics computed in the output data set only. The BYOUT option is used only when a BY statement is specified.

COVAR=*n*

declares that the first *n* variables on the right side of the model are simple linear regressors (covariates) and not factors in the quadratic response surface. By default, PROC RSREG forms quadratic and crossproduct effects for all regressor variables in the MODEL statement.

See the section "Handling Covariates" on page 9071 for more details and Example 108.2 for an example that uses covariates.

D

specifies that Cook's *D* influence statistic be written to the output data set.

See Chapter 4, "Introduction to Regression Procedures," for details and formulas.

LACKFIT

performs a lack-of-fit test.

See Draper and Smith (1981) for a discussion of lack-of-fit tests.

L95

specifies that the lower bound of a 95% confidence interval for an individual predicted value be written to the output data set. The variance used in calculating this bound is a function of both the mean square error and the variance of the parameter estimates.

See Chapter 4, "Introduction to Regression Procedures," for details and formulas.

L95M

specifies that the lower bound of a 95% confidence interval for the expected value of the dependent variable be written to the output data set. The variance used in calculating this bound is a function of the variance of the parameter estimates.

See Chapter 4, "Introduction to Regression Procedures," for details and formulas.

NOANOVA

NOAOV

suppresses the display of the analysis of variance and parameter estimates from the model fit.

NOCODE

performs the canonical and ridge analyses with the parameter estimates derived from fitting the response to the original values of the factor variables, rather than their coded values (see the section "Coding the Factor Variables" on page 9068 for more details). Use this option if the data are already stored in a coded form.

NOOPTIMAL

NOOPT

suppresses the display of the canonical analysis for the quadratic response surface.

NOPRINT

suppresses the display of both the analysis of variance and the canonical analysis.

PREDICT

specifies that the values predicted by the model be written to the output data set.

PRESS

computes and displays the predicted residual sum of squares (PRESS) statistic for each dependent variable in the model. The PRESS statistic is added to the summary information at the beginning of the analysis of variance, so if the NOANOVA or NOPRINT option is specified, then the PRESS option has no effect.

See Chapter 4, "Introduction to Regression Procedures," for details and formulas.

RESIDUAL

specifies that the residuals, calculated as ACTUAL – PREDICTED, be written to the output data set.

U95

specifies that the upper bound of a 95% confidence interval for an individual predicted value be written to the output data set. The variance used in calculating this bound is a function of both the mean square error and the variance of the parameter estimates.

See Chapter 4, "Introduction to Regression Procedures," for details and formulas.

U95M

specifies that the upper bound of a 95% confidence interval for the expected value of the dependent variable be written to the output data set. The variance used in calculating this bound is a function of the variance of the parameter estimates.

See Chapter 4, "Introduction to Regression Procedures," for details and formulas.

RIDGE Statement

RIDGE < *options* > **;**

A RIDGE statement computes the ridge of optimum response. The ridge starts at a given point $\mathbf{x}_0$, and the point on the ridge at radius r from $\mathbf{x}_0$ is the collection of factor settings that optimizes the predicted response at this radius. You can think of the ridge as climbing or falling as fast as possible on the surface of predicted response. Thus, the ridge analysis can be used as a tool to help interpret an existing response surface or to indicate the direction in which further experimentation should be performed.

The default starting point, $\mathbf{x}_0$, has each coordinate equal to the point midway between the highest and lowest values of the factor in the design. The default radii at which the ridge is computed are 0, 0.1, . . ., 0.9, 1. If the ridge analysis is based on the response surface fit to coded values for the factor variables (see the section "Coding the Factor Variables" on page 9068 for details), then this results in a ridge that starts at the point with a coded zero value for each coordinate and extends toward, but not beyond, the edge of the range of experimentation. Alternatively, both the center point of the ridge and the radii at which it is to be computed can be specified.

You can specify the following *options* in the RIDGE statement:

CENTER=*uncoded-factor-values*

gives the coordinates of the point $\mathbf{x}_0$ from which to begin the ridge. The coordinates should be given in the original (uncoded) factor variable values and should be separated by commas. There must be as many coordinates specified as there are factors in the model, and the order of the coordinates must be the same as that used in the MODEL statement. This starting point should be well inside the range of experimentation. The default sets each coordinate equal to the value midway between the highest and lowest values for the associated factor.

MAXIMUM

MAX

computes the ridge of maximum response. Both the MIN and MAX options can be specified; at least one must be specified.

MINIMUM

MIN

computes the ridge of minimum response. Both the MIN and MAX options can be specified; at least one must be specified.

NOPRINT
suppresses the display of the ridge analysis when only an output data set is required.

OUTR=*SAS-data-set*
creates an output SAS data set containing the computed optimum ridge.

For details, see the section "OUTR=SAS-data-set" on page 9073.

RADIUS=*coded-radii*
gives the distances from the ridge starting point at which to compute the optima. The values in the list represent distances between coded points. The list can take any of the following forms or can be composed of mixtures of them:

$m_1, m_2, \ldots, m_n$	specifies several values.
m TO n	specifies a sequence where m equals the starting value, n equals the ending value, and the increment equals 1.
m TO n BY i	specifies a sequence where m equals the starting value, n equals the ending value, and i equals the increment.

Mixtures of the preceding forms should be separated by commas. The default list runs from 0 to 1 by increments of 0.1. The following are examples of valid lists.

```
radius=0 to 5 by .5;
radius=0, .2, .25, .3, .5 to 1.0 by .1;
```

WEIGHT Statement

WEIGHT *variable* **;**

When a WEIGHT statement is specified, a weighted residual sum of squares

$$\sum_i w_i (y_i - \hat{y}_i)^2$$

is minimized, where w_i is the value of the variable specified in the WEIGHT statement, y_i is the observed value of the response variable, and $\hat{y}_i$ is the predicted value of the response variable.

The observation is used in the analysis only if the value of the WEIGHT statement variable is greater than zero. The WEIGHT statement has no effect on degrees of freedom or number of observations. If the weights for the observations are proportional to the reciprocals of the error variances, then the weighted least squares estimates are best linear unbiased estimators (BLUE).

Details: RSREG Procedure

Introduction to Response Surface Experiments

Many industrial experiments are conducted to discover which values of given factor variables optimize a response. If each factor is measured at three or more values, a quadratic response surface can be estimated by least squares regression. The predicted optimal value can be found from the estimated surface if the surface is shaped like a simple hill or valley. If the estimated surface is more complicated, or if the predicted optimum is far from the region of experimentation, then the shape of the surface can be analyzed to indicate the directions in which new experiments should be performed.

Suppose that a response variable y is measured at combinations of values of two factor variables, x_1 and x_2. The quadratic response surface model for this variable is written as

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1^2 + \beta_4 x_2^2 + \beta_5 x_1 x_2 + \epsilon$$

The steps in the analysis for such data are as follows:

1. model fitting and analysis of variance, including lack-of-fit testing, to estimate parameters
2. canonical analysis to investigate the shape of the predicted response surface
3. ridge analysis to search for the region of optimum response

Model Fitting and Analysis of Variance

The first task in analyzing the response surface is to estimate the parameters of the model by least squares regression and to obtain information about the fit in the form of an analysis of variance. The estimated surface is typically curved: a *hill* with the peak occurring at the unique estimated point of maximum response, a *valley,* or a *saddle surface* with no unique minimum or maximum. Use the results of this phase of the analysis to answer the following questions:

- What is the contribution of each type of effect—linear, quadratic, and crossproduct—to the statistical fit? The ANOVA table with sources labeled “Regression” addresses this question.
- What part of the residual error is due to lack of fit? Does the quadratic response model adequately represent the true response surface? If you specify the LACKFIT option in the MODEL statement, then the ANOVA table with sources labeled “Residual” addresses this question. See the section “Lack-of-Fit Test” on page 9067 for details.
- What is the contribution of each factor variable to the statistical fit? Can the response be predicted accurately if the variable is removed? The ANOVA table with sources labeled “Factor” addresses this question.
- What are the predicted responses for a grid of factor values? (See the section “Plotting the Surface” on page 9069 and the section “Searching for Multiple Response Conditions” on page 9069.)

Lack-of-Fit Test

The lack-of-fit test compares the variation around the model with *pure* variation within replicated observations. This measures the adequacy of the quadratic response surface model. In particular, if there are n_i replicated observations $Y_{i1}, \ldots, Y_{in_i}$ of the response all at the same values $\mathbf{x}_i$ of the factors, then you can predict the true response at $\mathbf{x}_i$ either by using the predicted value $\hat{Y}_i$ based on the model or by using the mean $\bar{Y}_i$ of the replicated values. The lack-of-fit test decomposes the residual error into a component due to the variation of the replications around their mean value (the *pure* error) and a component due to the variation of the mean values around the model prediction (the *bias* error):

$$\sum_i \sum_{j=1}^{n_i} \left(Y_{ij} - \hat{Y}_i\right)^2 = \sum_i \sum_{j=1}^{n_i} \left(Y_{ij} - \bar{Y}_i\right)^2 + \sum_i n_i \left(\bar{Y}_i - \hat{Y}_i\right)^2$$

If the model is adequate, then both components estimate the nominal level of error; however, if the bias component of error is much larger than the pure error, then this constitutes evidence that there is significant lack of fit.

If some observations in your design are replicated, you can test for lack of fit by specifying the LACKFIT option in the MODEL statement. Note that, since all other tests use total error rather than pure error, you might want to hand-calculate the tests with respect to pure error if the lack of fit is significant. On the other hand, significant lack of fit indicates that the quadratic model is inadequate, so if this is a problem you can also try to refine the model, possibly by using PROC GLM for general polynomial modeling; see Chapter 53, "The GLM Procedure," for more information. Example 108.1 illustrates the use of the LACKFIT option.

Canonical Analysis

The second task in analyzing the response surface is to examine the overall shape of the curve and determine whether the estimated stationary point is a maximum, a minimum, or a saddle point. The canonical analysis can be used to answer the following questions:

- Is the surface shaped like a hill, a valley, or a saddle, or is it flat?
- If there is a unique optimum combination of factor values, where is it?
- To which factor or factors are the predicted responses most sensitive?

The eigenvalues and eigenvectors in the matrix of second-order parameters characterize the shape of the response surface. The eigenvectors point in the directions of principal orientation for the surface, and the signs and magnitudes of the associated eigenvalues give the shape of the surface in these directions. Positive eigenvalues indicate directions of upward curvature, and negative eigenvalues indicate directions of downward curvature. The larger an eigenvalue is in absolute value, the more pronounced is the curvature of the response surface in the associated direction. Often, all the coefficients of an eigenvector except for one are relatively small, indicating that the vector points roughly along the axis associated with the factor corresponding to the single large coefficient. In this case, the canonical analysis can be used to determine the relative sensitivity of the predicted response surface to variations in that factor. (See the section "Getting Started: RSREG Procedure" on page 9051 for an example.)

Ridge Analysis

If the estimated surface is found to have a simple optimum well within the range of experimentation, the analysis performed by the preceding two steps might be sufficient. In more complicated situations, further search for the region of optimum response is required. The method of ridge analysis computes the estimated ridge of optimum response for increasing radii from the center of the original design. The ridge analysis answers the following question:

- If there is not a unique optimum of the response surface within the range of experimentation, in which direction should further searching be done in order to locate the optimum?

You can use the RIDGE statement to compute the ridge of maximum or minimum response.

Coding the Factor Variables

For the results of the canonical and ridge analyses to be interpretable, the values of different factor variables should be comparable. This is because the canonical and ridge analyses of the response surface are not invariant with respect to differences in scale and location of the factor variables. The analysis of variance is not affected by these changes. Although the actual predicted surface does not change, its parameterization does. The usual solution to this problem is to code each factor variable so that its minimum in the experiment is –1 and its maximum is 1 and to carry through the analysis with the coded values instead of the original ones. This practice has the added benefit of making 1 a reasonable boundary radius for the ridge analysis since 1 represents approximately the edge of the experimental region. By default, PROC RSREG computes the linear transformation to perform this coding as the data are initially read in, and the canonical and ridge analyses are performed on the model fit to the coded data. The actual form of the coding operation for each value of a variable is

$$coded\ value = (original\ value - M)/S$$

where M is the average of the highest and lowest values for the variable in the design and S is half their difference.

Missing Values

If an observation has missing data for any of the variables used by the procedure, then that observation is not used in the estimation process. If one or more response variables are missing, but no factor or covariate variables are missing, then predicted values and confidence limits are computed for the output data set, but the residual and Cook's D statistic are missing.

Plotting the Surface

Specifying the PLOTS=SURFACE option in the PROC RSREG statement displays contour plots for all pairs of factors in the model (see Example 108.1), while specifying the PLOTS=SURFACE(3D) option displays a three-dimensional surface as shown in Figure 108.4.

You can also generate predicted values for a grid of points with the PREDICT option (see the section "Getting Started: RSREG Procedure" on page 9051 for an example) and then use these values to create a contour plot or a three-dimensional plot of the response surface over a two-dimensional grid. Any two factor variables can be chosen to form the grid for the plot. Several plots can be generated by using different pairs of factor variables.

Searching for Multiple Response Conditions

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/rsrdt.sas.)

Suppose you have the following data with two factors and three responses, and you want to find the factor setting that produces responses in a certain region:

```
data a;
   input x1 x2 y1 y2 y3;
   datalines;
-1      -1         1.8 1.940  3.6398
-1       1         2.6 1.843  4.9123
 1      -1         5.4 1.063  6.0128
 1       1         0.7 1.639  2.3629
 0       0         8.5 0.134  9.0910
 0       0         3.0 0.545  3.7349
 0       0         9.8 0.453 10.4412
 0       0         4.1 1.117  5.0042
 0       0         4.8 1.690  6.6245
 0       0         5.9 1.165  6.9420
 0       0         7.3 1.013  8.7442
 0       0         9.3 1.179 10.2762
 1.4142  0         3.9 0.945  5.0245
-1.4142  0         1.7 0.333  2.4041
 0       1.4142    3.0 1.869  5.2695
 0      -1.4142    5.7 0.099  5.4346
;
```

You want to find the values of x1 and x2 that maximize y1 subject to y2<2 and y3<y2+y1. The exact answer is not easy to obtain analytically, but you can obtain a practically feasible solution by checking conditions across a grid of values in the range of interest. First, append a grid of factor values to the observed data, with missing values for the responses:

```
data b;
   set a end=eof;
   output;
   if eof then do;
```

```
      y1=.;
      y2=.;
      y3=.;
      do x1=-2 to 2 by .1;
         do x2=-2 to 2 by .1;
            output;
         end;
      end;
   end;
run;
```

Next, use PROC RSREG to fit a response surface model to the data and to compute predicted values for both the observed data and the grid, putting the predicted values in a data set c:

```
proc rsreg data=b out=c;
   model y1 y2 y3=x1 x2 / predict;
run;
```

Finally, find the subset of predicted values that satisfy the constraints, sort by the unconstrained variable, and display the top five predictions:

```
data d;
   set c;
   if y2<2;
   if y3<y2+y1;

proc sort data=d;
   by descending y1;
run;

data d; set d;
   if (_n_ <= 5);
proc print;
run;
```

The results are displayed in Figure 108.5. They indicate that optimal values of the factors are around 0.3 for x1 and around –0.5 for x2.

Figure 108.5 Top Five Predictions

Obs	x1	x2	_TYPE_	y1	y2	y3
1	0.3	-0.5	PREDICT	6.92570	0.75784	7.60471
2	0.3	-0.6	PREDICT	6.91424	0.74174	7.54194
3	0.3	-0.4	PREDICT	6.91003	0.77870	7.64341
4	0.4	-0.6	PREDICT	6.90769	0.73357	7.51836
5	0.4	-0.5	PREDICT	6.90540	0.75135	7.56883

If you are also interested in simultaneously optimizing y1 and y2, you can specify the following statements to make a visual comparison of the two response surfaces by overlaying their contour plots:

```
ods graphics on;
proc rsreg data=a plots=surface(overlaypairs);
   model y1 y2=x1 x2;
run;
```

```
ods graphics off;
```

Figure 108.6 shows that you have to make some compromises in any attempt to maximize both y1 and y2; however, you might be able to maximize y1 while minimizing y2.

Figure 108.6 Overlaid Line Contours of Predicted Responses

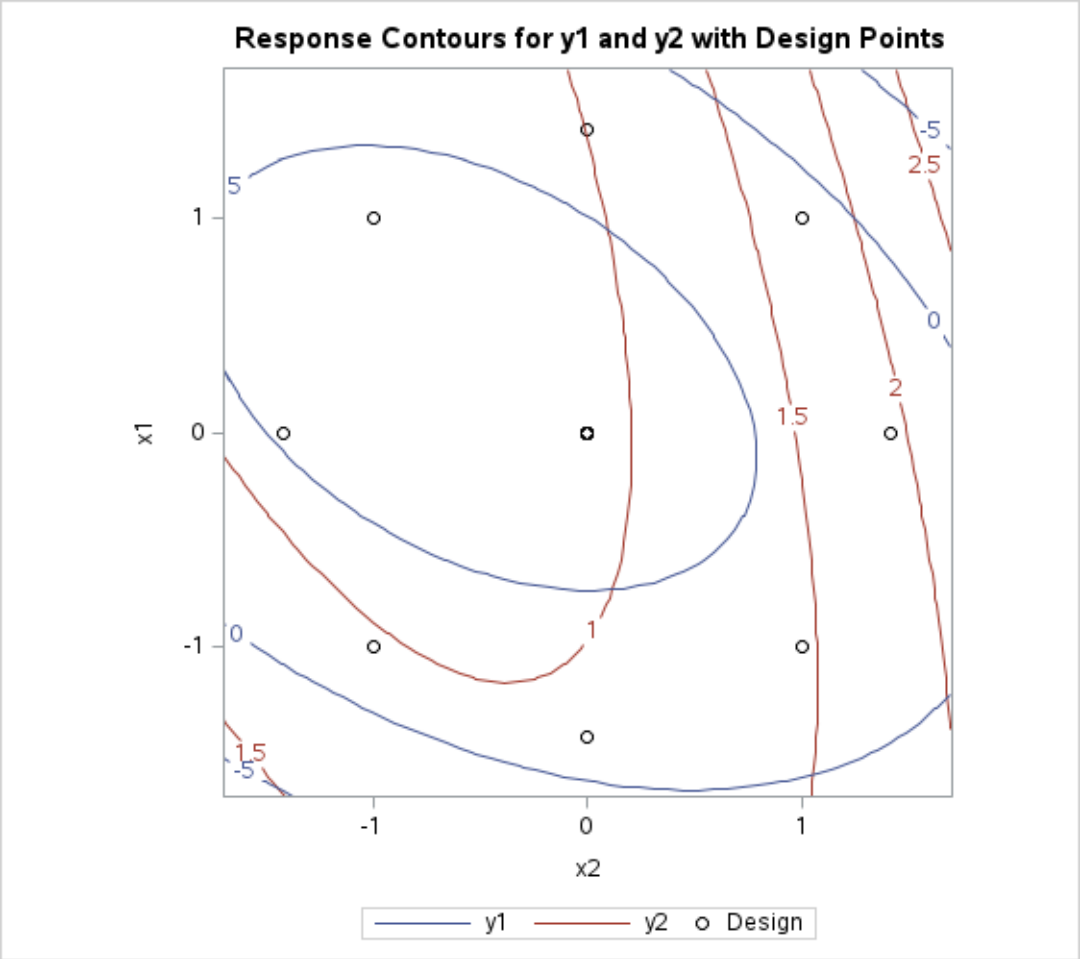

Handling Covariates

Covariate regressors are added to a response surface model because they are believed to account for a sizable yet relatively uninteresting portion of the variation in the data. What the experimenter is really interested in is the response corrected for the effect of the covariates. A common example is the block effect in a block design. In the canonical and ridge analyses of a response surface, which estimate responses at hypothetical levels of the factor variables, the actual value of the predicted response is computed by using the average values of the covariates. The estimated response values do optimize the estimated surface of the response corrected for covariates, but true prediction of the response requires actual values for the covariates. You can use the COVAR= option in the MODEL statement to include covariates in the response surface model. Example 108.2 illustrates the use of this option.

Computational Method

Canonical Analysis

For each response variable, the model can be written in the form

$$y_i = \mathbf{x}_i'\mathbf{A}\mathbf{x}_i + \mathbf{b}'\mathbf{x}_i + \mathbf{c}'\mathbf{z}_i + \epsilon_i$$

where

y_i is the ith observation of the response variable.

$\mathbf{x}_i$ $= (x_{i1}, x_{i2}, \ldots, x_{ik})'$ are the k factor variables for the ith observation.

$\mathbf{z}_i$ $= (z_{i1}, z_{i2}, \ldots, z_{iL})'$ are the L covariates, including the intercept term.

$\mathbf{A}$ is the $k \times k$ symmetrized matrix of quadratic parameters, with diagonal elements equal to the coefficients of the pure quadratic terms in the model and off-diagonal elements equal to half the coefficient of the corresponding crossproduct.

$\mathbf{b}$ is the $k \times 1$ vector of linear parameters.

$\mathbf{c}$ is the $L \times 1$ vector of covariate parameters, one of which is the intercept.

ϵ_i is the error associated with the ith observation. Tests performed by PROC RSREG assume that errors are independently and normally distributed with mean zero and variance σ^2.

The parameters in $\mathbf{A}$, $\mathbf{b}$, and $\mathbf{c}$ are estimated by least squares. To optimize $\mathbf{y}$ with respect to $\mathbf{x}$, take partial derivatives, set them to zero, and solve:

$$\frac{\partial y}{\partial \mathbf{x}} = 2\mathbf{x}'\mathbf{A} + \mathbf{b}' = \mathbf{0} \quad \Longrightarrow \quad \mathbf{x} = -\frac{1}{2}\mathbf{A}^{-1}\mathbf{b}$$

You can determine if the solution is a maximum or minimum by looking at the eigenvalues of $\mathbf{A}$:

If the eigenvalues...	then the solution is...
are all negative	a maximum
are all positive	a minimum
have mixed signs	a saddle point
contain zeros	in a flat area

Ridge Analysis

If the largest eigenvalue is positive, its eigenvector gives the direction of steepest ascent from the stationary point; if the largest eigenvalue is negative, its eigenvector gives the direction of steepest descent. The eigenvectors corresponding to small or zero eigenvalues point in directions of relative flatness.

The point on the optimum response ridge at a given radius R from the ridge origin is found by optimizing

$$(\mathbf{x}_0 + \mathbf{d})'\mathbf{A}(\mathbf{x}_0 + \mathbf{d}) + \mathbf{b}'(\mathbf{x}_0 + \mathbf{d})$$

over $\mathbf{d}$ satisfying $\mathbf{d}'\mathbf{d} = R^2$, where $\mathbf{x}_0$ is the $k \times 1$ vector containing the ridge origin and $\mathbf{A}$ and $\mathbf{b}$ are as previously discussed. By the method of Lagrange multipliers, the optimal $\mathbf{d}$ has the form

$$\mathbf{d} = -(\mathbf{A} - \mu\mathbf{I})^{-1}(\mathbf{A}\mathbf{x}_0 + 0.5\mathbf{b})$$

where $\mathbf{I}$ is the $k \times k$ identity matrix and μ is chosen so that $\mathbf{d}'\mathbf{d} = R^2$. There can be several values of μ that satisfy this constraint; the correct one depends on which sort of response ridge is of interest. If you are searching for the ridge of maximum response, then the appropriate μ is the unique one that satisfies the constraint and is greater than all the eigenvalues of $\mathbf{A}$. Similarly, the appropriate μ for the ridge of minimum response satisfies the constraint and is less than all the eigenvalues of $\mathbf{A}$. (See Myers and Montgomery (1995) for details.)

Output Data Sets

OUT=SAS-data-set

An output data set containing statistics requested with options in the MODEL statement for each observation in the input data set is created whenever the OUT= option is specified in the PROC RSREG statement. The data set contains the following variables:

- the BY variables
- the ID variables
- the WEIGHT variable
- the independent variables in the MODEL statement
- the variable _TYPE_, which identifies the observation type in the output data set. _TYPE_ is a character variable with a length of eight, and it takes on the values 'ACTUAL', 'PREDICT', 'RESIDUAL', 'U95M', 'L95M', 'U95', 'L95', and 'D', corresponding to the options specified.
- the response variables containing special output values identified by the _TYPE_ variable

All confidence limits use the two-tailed Student's t value.

OUTR=SAS-data-set

An output data set containing the optimum response ridge is created when the OUTR= option is specified in the RIDGE statement. The data set contains the following variables:

- the current values of the BY variables
- a character variable _DEPVAR_ containing the name of the dependent variable
- a character variable _TYPE_ identifying the type of ridge being computed, MINIMUM or MAXIMUM. If both MAXIMUM and MINIMUM are specified, the data set contains observations for the minimum ridge followed by observations for the maximum ridge.
- a numeric variable _RADIUS_ giving the distance from the ridge starting point
- the values of the model factors at the estimated optimum point at distance _RADIUS_ from the ridge starting point

- a numeric variable _PRED_, which is the estimated expected value of the dependent variable at the optimum
- a numeric variable _STDERR_, which is the standard error of the estimated expected value

Displayed Output

All estimates and hypothesis tests assume that the model is correctly specified and the errors are distributed according to classical statistical assumptions.

The output displayed by PROC RSREG includes the following.

Estimation and Analysis of Variance

- The actual form of the coding operation for each value of a variable is

$$coded\ value = \frac{1}{S}(original\ value - M)$$

  where M is the average of the highest and lowest values for the variable in the design and S is half their difference. The Subtracted off column contains the M values for this formula for each factor variable, and S is found in the Divided by column.

- The summary table for the response variable contains the following information.
  - "Response Mean" is the mean of the response variable in the sample. When a WEIGHT statement is specified, the mean $\bar{y}$ is calculated by

$$\bar{y} = \frac{\sum_i w_i y_i}{\sum_i w_i}$$

  - "Root MSE" estimates the standard deviation of the response variable and is calculated as the square root of the "Total Error" mean square.
  - The "R-Square" value is R^2, or the coefficient of determination. R^2 measures the proportion of the variation in the response that is attributed to the model rather than to random error.
  - The "Coefficient of Variation" is 100 times the ratio of the "Root MSE" to the "Response Mean."

- A table analyzing the significance of the terms of the regression is displayed. Terms are brought into the regression in four steps: (1) the "Intercept" and any covariates in the model, (2) "Linear" terms like X1 and X2, (3) pure "Quadratic" terms like X1*X1 or X2*X2, and (4) "Crossproduct" terms like X1*X2. The table displays the following information:
  - the degrees of freedom in the DF column, which should be the same as the number of corresponding parameters unless one or more of the parameters are not estimable
  - Type I Sum of Squares, also called the sequential sums of squares, which measures the reduction in the error sum of squares as sets of terms (Linear, Quadratic, and so forth) are added to the model

  - R-Square, which measures the portion of total R^2 contributed as each set of terms (Linear, Quadratic, and so forth) is added to the model
  - F Value, which tests the null hypothesis that all parameters in the term are zero by using the Total Error mean square as the denominator. This is a test of a Type I hypothesis, containing the usual F test numerator, conditional on the effects of subsequent variables not being in the model.
  - Pr > F, which is the significance value or probability of obtaining at least as great an F ratio given that the null hypothesis is true.

- The Sum of Squares column partitions the "Total Error" into "Lack of Fit" and "Pure Error." When "Lack of Fit" is significant, there is variation around the model other than random error (such as cubic effects of the factor variables).
  - The "Total Error" Mean Square estimates σ^2, the variance.
  - F Value tests the null hypothesis that the variation is adequately described by random error.

- A table containing the parameter estimates from the model is displayed.
  - The Estimate column contains the parameter estimates based on the *uncoded* values of the factor variables. If an effect is a linear combination of previous effects, the parameter for the effect is not estimable. When this happens, the degrees of freedom are zero, the parameter estimate is set to zero, and estimates and tests on other parameters are conditional on this parameter being zero.
  - The Standard Error column contains the estimated standard deviations of the parameter estimates based on *uncoded* data.
  - The t Value column contains t values of a test of the null hypothesis that the true parameter is zero when the *uncoded* values of the factor variables are used.
  - The Pr > |T| column gives the significance value or probability of a greater absolute t ratio given that the true parameter is zero.
  - The Parameter Estimate from Coded Data column contains the parameter estimates based on the *coded* values of the factor variables. These are the estimates used in the subsequent canonical and ridge analyses.

- The sum of squares are partitioned by the factors in the model, and an analysis table is displayed. The test on a factor is a joint test on all the parameters involving that factor. For example, the test for the factor X1 tests the null hypothesis that the true parameters for X1, X1*X1, and X1*X2 are all zero.

Canonical Analysis

- The Critical Value columns contain the values of the factor variables that correspond to the stationary point of the fitted response surface. The critical values can be at a minimum, maximum, or saddle point.

- The eigenvalues and eigenvectors are from the matrix of quadratic parameter estimates based on the coded data. They characterize the shape of the response surface.

Ridge Analysis

- The Coded Radius column contains the distance from the coded version of the associated point to the coded version of the origin of the ridge. The origin is given by the point at radius zero.
- The Estimated Response column contains the estimated value of the response variable at the associated point. The standard error of this estimate is also given. This quantity is useful for assessing the relative credibility of the prediction at a given radius. Typically, this standard error increases rapidly as the ridge moves up to and beyond the design perimeter, reflecting the inherent difficulty of making predictions beyond the range of experimentation.
- The Uncoded Factor Values columns contain the values of the uncoded factor variables that give the optimum response at this radius from the ridge origin.

ODS Table Names

PROC RSREG assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 108.4. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 108.4 ODS Tables Produced by PROC RSREG

ODS Table Name	Description	Statement
Coding	Coding coefficients for the independent variables	default
ErrorANOVA	Error analysis of variance	default
FactorANOVA	Factor analysis of variance	default
FitStatistics	Overall statistics for fit	default
ModelANOVA	Model analysis of variance	default
ParameterEstimates	Estimated linear parameters	default
Ridge	Ridge analysis for optimum response	RIDGE
Spectral	Spectral analysis	default
StationaryPoint	Stationary point of response surface	default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC RSREG assigns a name to each graph it creates using ODS. The names are listed in Table 108.5. You can use these names to reference the graphs when using ODS. You must also specify the PLOTS= option and any other options indicated in Table 108.5.

Table 108.5 Graphs Produced by PROC RSREG

ODS Graph Name	Plot Description	PLOTS= Option
FitPlot	Fit plot for 1 predictor	FIT
DiagnosticsPanel	Panel of fit diagnostics	DIAGNOSTICS
CooksDPlot	Cook's *D* plot	DIAGNOSTICS(UNPACK)
ObservedByPredicted	Observed by predicted	DIAGNOSTICS(UNPACK)
QQPlot	Residual Q-Q plot	DIAGNOSTICS(UNPACK)
ResidualByPredicted	Residual by predicted values	DIAGNOSTICS(UNPACK)
ResidualHistogram	Residual histogram	DIAGNOSTICS(UNPACK)
RFPlot	RF plot	DIAGNOSTICS(UNPACK)
RStudentByPredicted	Studentized residuals by predicted	DIAGNOSTICS(UNPACK)
RStudentByLeverage	RStudent by hat diagonals	DIAGNOSTICS(UNPACK)
ResidualPlots	Panel of residuals by predictors	RESIDUALS
	Residuals by predictors	RESIDUALS(UNPACK)
RidgePlots	Panel of ridge plot and factors	RIDGE
		(with RIDGE MAX or MIN)
	Ridge plot	RIDGE(UNPACK)
		(with RIDGE MAX or MIN)
	Ridge factors	RIDGE(UNPACK)
		(with RIDGE MAX or MIN)
Contour	Panel of contour plots	SURFACE
	Contour plots	SURFACE(UNPACK)
Surface	Panel of 3-D surface plots	SURFACE(3D)
	3-D surface plots	SURFACE(3D UNPACK)
ContourOverlay	Panel of overlaid line-contour plots	SURFACE(OVERLAYPAIRS)
	Overlaid line-contour plots	SURFACE(OVERLAYPAIRS UNPACK)

Examples: RSREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 108.1: A Saddle Surface Response Using Ridge Analysis

(View the complete code for this example (rsrex1.sas) in the example repository.)

Myers (1976) analyzes an experiment reported by Frankel (1961) aimed at maximizing the yield of mercaptobenzothiazole (MBT) by varying processing time and temperature. Myers (1976) uses a two-factor model in which the estimated surface does not have a unique optimum. A ridge analysis is used to determine the region in which the optimum lies. The objective is to find the settings of time and temperature in the processing of a chemical that maximize the yield. The following statements produce Output 108.1.1 through Output 108.1.5:

```
data d;
   input Time Temp MBT;
   label Time = "Reaction Time (Hours)"
         Temp = "Temperature (Degrees Centigrade)"
         MBT  = "Percent Yield Mercaptobenzothiazole";
   datalines;
 4.0   250   83.8
20.0   250   81.7
12.0   250   82.4
12.0   250   82.9
12.0   220   84.7
12.0   280   57.9
12.0   250   81.2
 6.3   229   81.3
 6.3   271   83.1
17.7   229   85.3
17.7   271   72.7
 4.0   250   82.0
;

ods graphics on;
proc rsreg data=d plots=(ridge surface);
   model MBT=Time Temp / lackfit;
   ridge max;
run;
ods graphics off;
```

Output 108.1.1 displays the coding coefficients for the transformation of the independent variables to lie between –1 and 1 and some simple statistics for the response variable.

Output 108.1.1 Coding and Response Variable Information

The RSREG Procedure

Coding Coefficients for the Independent Variables		
Factor	**Subtracted off**	**Divided by**
Time	12.000000	8.000000
Temp	250.000000	30.000000

Response Surface for Variable MBT: Percent Yield Mercaptobenzothiazole	
Response Mean	79.916667
Root MSE	4.615964
R-Square	0.8003
Coefficient of Variation	5.7760

Output 108.1.2 shows that the lack of fit for the model is highly significant. Since the quadratic model does not fit the data very well, firm statements about the underlying process should not be based only on the current analysis. Note from the analysis of variance for the model that the test for the time factor is not significant. If further experimentation is undertaken, it might be best to fix Time at a moderate to high value and to concentrate on the effect of temperature. In the actual experiment discussed here, extra runs were made that confirmed the results of the following analysis.

Output 108.1.2 Analyses of Variance

Regression	DF	Type I Sum of Squares	R-Square	F Value	Pr > F
Linear	2	313.585803	0.4899	7.36	0.0243
Quadratic	2	146.768144	0.2293	3.44	0.1009
Crossproduct	1	51.840000	0.0810	2.43	0.1698
Total Model	5	512.193947	0.8003	4.81	0.0410

Residual	DF	Sum of Squares	Mean Square	F Value	Pr > F
Lack of Fit	3	124.696053	41.565351	39.63	0.0065
Pure Error	3	3.146667	1.048889		
Total Error	6	127.842720	21.307120		

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Parameter Estimate from Coded Data
Intercept	1	-545.867976	277.145373	-1.97	0.0964	82.173110
Time	1	6.872863	5.004928	1.37	0.2188	-1.014287
Temp	1	4.989743	2.165839	2.30	0.0608	-8.676768
Time*Time	1	0.021631	0.056784	0.38	0.7164	1.384394
Temp*Time	1	-0.030075	0.019281	-1.56	0.1698	-7.218045
Temp*Temp	1	-0.009836	0.004304	-2.29	0.0623	-8.852519

Output 108.1.2 *continued*

Factor	DF	Sum of Squares	Mean Square	F Value	Pr > F	Label
Time	3	61.290957	20.430319	0.96	0.4704	Reaction Time (Hours)
Temp	3	461.250925	153.750308	7.22	0.0205	Temperature (Degrees Centigrade)

The canonical analysis (Output 108.1.3) indicates that the predicted response surface is shaped like a saddle. The eigenvalue of 2.5 shows that the valley orientation of the saddle is less curved than the hill orientation, with an eigenvalue of –9.99. The coefficients of the associated eigenvectors show that the valley is more aligned with Time and the hill with Temp. Because the canonical analysis resulted in a saddle point, the estimated surface does not have a unique optimum.

Output 108.1.3 Canonical Analysis

	Critical Value		
Factor	**Coded**	**Uncoded**	**Label**
Time	-0.441758	8.465935	Reaction Time (Hours)
Temp	-0.309976	240.700718	Temperature (Degrees Centigrade)
Predicted value at stationary point: 83.741940			

	Eigenvectors	
Eigenvalues	**Time**	**Temp**
2.528816	0.953223	-0.302267
-9.996940	0.302267	0.953223
Stationary point is a saddle point.		

However, the ridge analysis in Output 108.1.4 and the ridge plot in Output 108.1.5 indicate that maximum yields result from relatively high reaction times and low temperatures. A contour plot of the predicted response surface, shown in Output 108.1.5, confirms this conclusion.

Output 108.1.4 Ridge Analysis

Estimated Ridge of Maximum Response for Variable MBT: Percent Yield Mercaptobenzothiazole

			Uncoded Factor Values	
Coded Radius	**Estimated Response**	**Standard Error**	**Time**	**Temp**
0.0	82.173110	2.665023	12.000000	250.000000
0.1	82.952909	2.648671	11.964493	247.002956
0.2	83.558260	2.602270	12.142790	244.023941
0.3	84.037098	2.533296	12.704153	241.396084
0.4	84.470454	2.457836	13.517555	239.435227
0.5	84.914099	2.404616	14.370977	237.919138
0.6	85.390012	2.410981	15.212247	236.624811
0.7	85.906767	2.516619	16.037822	235.449230
0.8	86.468277	2.752355	16.850813	234.344204
0.9	87.076587	3.130961	17.654321	233.284652
1.0	87.732874	3.648568	18.450682	232.256238

Output 108.1.5 Ridge and Contour Plot of Predicted Response Surface

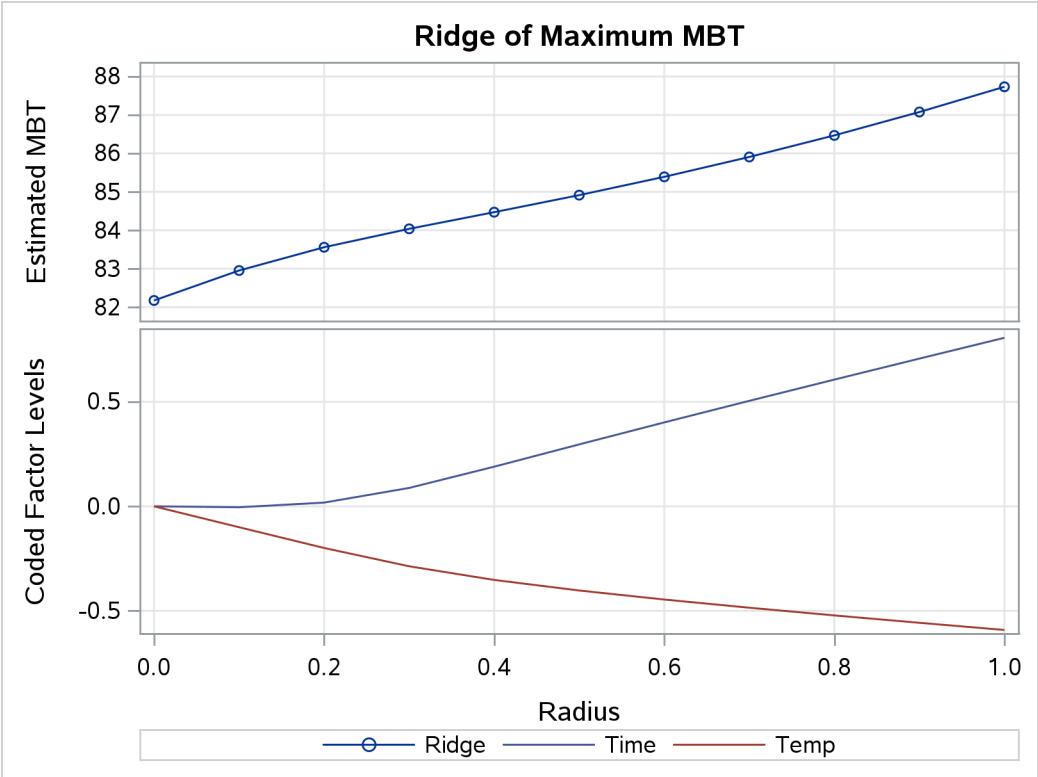

Output 108.1.5 *continued*

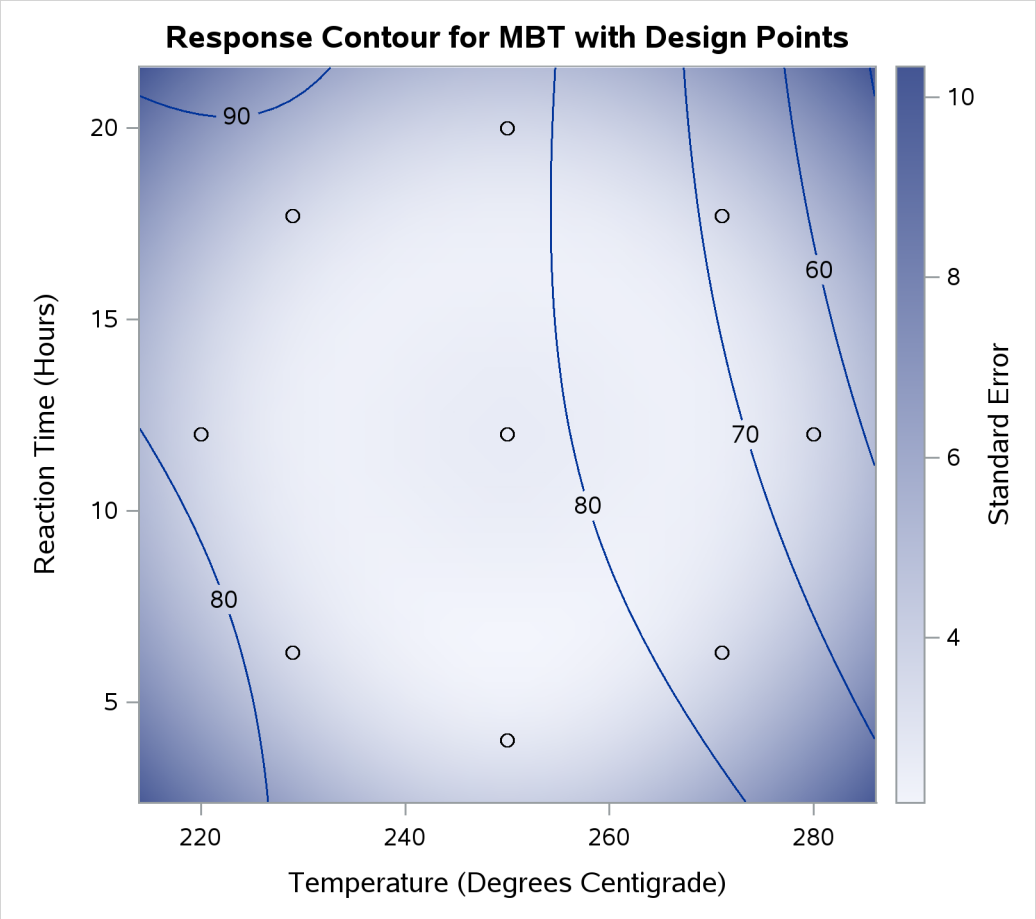

Example 108.2: Response Surface Analysis with Covariates

(View the complete code for this example (rsrex2.sas) in the example repository.)

One way of viewing covariates is as extra sources of variation in the dependent variable that can mask the variation due to primary factors. This example demonstrates the use of the COVAR= option in PROC RSREG to fit a response surface model to the dependent variables corrected for the covariates.

You have a chemical process with a yield that you hypothesize to be dependent on three factors: reaction time, reaction temperature, and reaction pressure. You perform an experiment to measure this dependence. You are willing to include up to 20 runs in your experiment, but you can perform no more than 8 runs on the same day, so the design for the experiment is composed of three blocks. Additionally, you know that the grade of raw material for the reaction has a significant impact on the yield. You have no control over this, but you keep track of it. The following statements create a SAS data set containing the results of the experiment:

```
data Experiment;
   input Day Grade Time Temp Pressure Yield;
   datalines;
1 67      -1      -1      -1          32.98
1 68      -1       1       1          47.04
```

```
1 70       1      -1       1          67.11
1 66       1       1      -1          26.94
1 74       0       0       0         103.22
1 68       0       0       0          42.94
2 75      -1      -1       1         122.93
2 69      -1       1      -1          62.97
2 70       1      -1      -1          72.96
2 71       1       1       1          94.93
2 72       0       0       0          93.11
2 74       0       0       0         112.97
3 69       1.633   0       0          78.88
3 67      -1.633   0       0          52.53
3 68       0       1.633   0          68.96
3 71       0      -1.633   0          92.56
3 70       0       0       1.633      88.99
3 72       0       0      -1.633     102.50
3 70       0       0       0          82.84
3 72       0       0       0         103.12
;
```

Your first analysis neglects to take the covariates into account. The following statements use PROC RSREG to fit a response surface to the observed yield, but note that Day and Grade are omitted:

```
proc rsreg data=Experiment;
   model Yield = Time Temp Pressure;
run;
```

The ANOVA results shown in Output 108.2.1 indicate that *no* process variable effects are significantly larger than the background noise.

Output 108.2.1 Analysis of Variance Ignoring Covariates

The RSREG Procedure

Regression	DF	Type I Sum of Squares	R-Square	F Value	Pr > F
Linear	3	1880.842426	0.1353	0.67	0.5915
Quadratic	3	2370.438681	0.1706	0.84	0.5023
Crossproduct	3	241.873250	0.0174	0.09	0.9663
Total Model	9	4493.154356	0.3233	0.53	0.8226

Residual	DF	Sum of Squares	Mean Square
Total Error	10	9405.129724	940.512972

However, when the yields are adjusted for covariate effects of day and grade of raw material, very strong process variable effects are revealed. The following statements produce the ANOVA results in Output 108.2.2. Note that in order to include the effects of the classification factor Day as covariates, you need to create dummy variables indicating each day separately.

```
data Experiment;
   set Experiment;
   d1 = (Day = 1);
   d2 = (Day = 2);
```

```
   d3 = (Day = 3);

ods graphics on;
proc rsreg data=Experiment plots=all;
   model Yield = d1-d3 Grade Time Temp Pressure / covar=4;
run;
ods graphics off;
```

The results show very strong effects due to both the covariates and the process variables.

Output 108.2.2 Analysis of Variance Including Covariates

The RSREG Procedure

Regression	DF	Type I Sum of Squares	R-Square	F Value	Pr > F
Covariates	3	13695	0.9854	316957	<.0001
Linear	3	156.524497	0.0113	3622.53	<.0001
Quadratic	3	22.989775	0.0017	532.06	<.0001
Crossproduct	3	23.403614	0.0017	541.64	<.0001
Total Model	12	13898	1.0000	80413.2	<.0001

Residual	DF	Sum of Squares	Mean Square
Total Error	7	0.100820	0.014403

The number of observations in the data set might be too small for the diagnostic plots in Output 108.2.3 to dependably identify problems; however, some outliers are indicated. The residual plots in Output 108.2.4 do not display any obvious structure.

Output 108.2.3 Fit Diagnostics

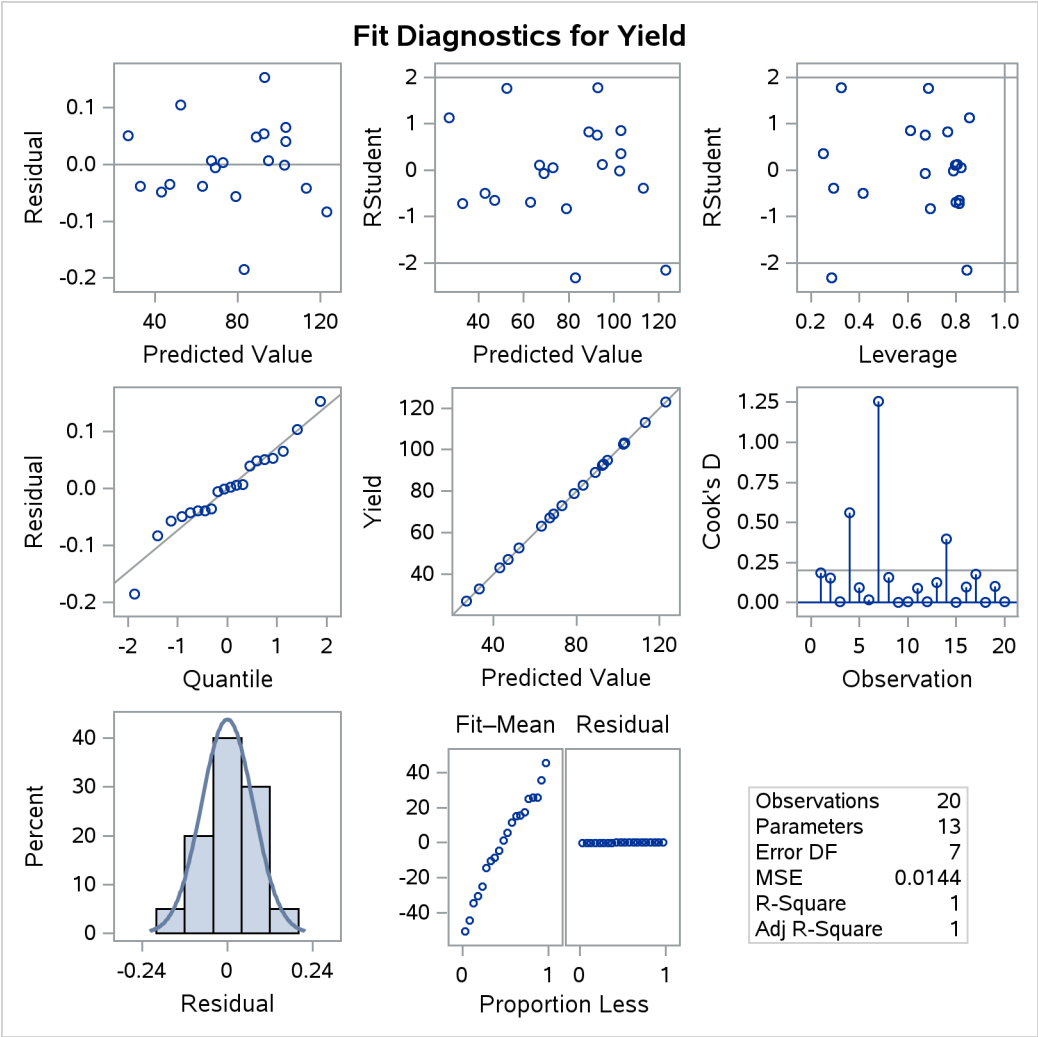

Output 108.2.4 Residual Plots

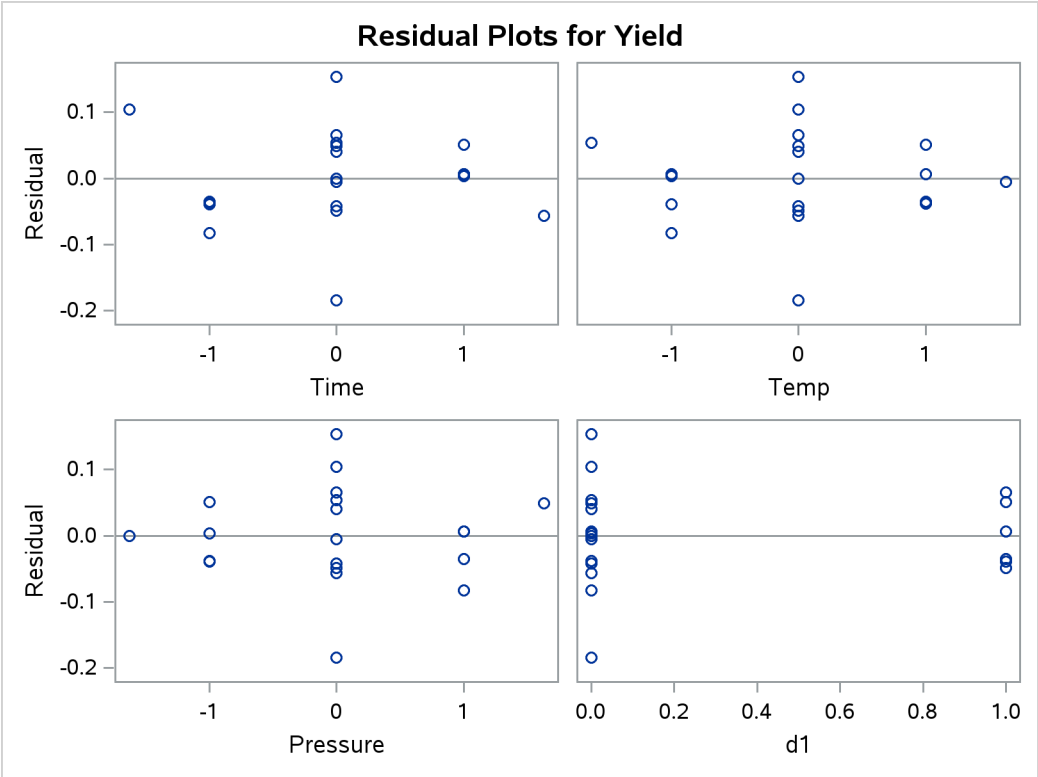

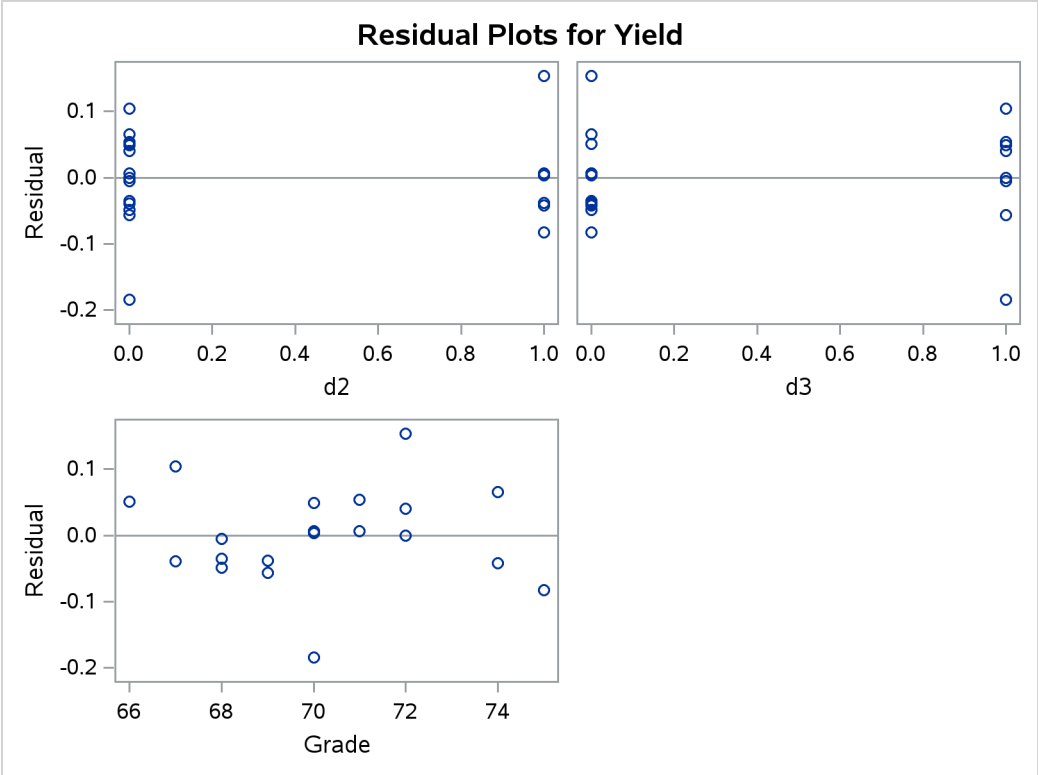

References

Box, G. E. P. (1954). "The Exploration and Exploitation of Response Surfaces: Some General Considerations." *Biometrics* 10:16.

Box, G. E. P., and Draper, N. R. (1982). "Measures of Lack of Fit for Response Surface Designs and Predictor Variable Transformations." *Technometrics* 24:1–8.

Box, G. E. P., and Draper, N. R. (1987). *Empirical Model Building and Response Surfaces*. New York: John Wiley & Sons.

Box, G. E. P., and Hunter, J. S. (1957). "Multifactor Experimental Designs for Exploring Response Surfaces." *Annals of Mathematical Statistics* 28:195–242.

Box, G. E. P., Hunter, W. G., and Hunter, J. S. (1978). *Statistics for Experimenters*. New York: John Wiley & Sons.

Box, G. E. P., and Wilson, K. B. (1951). "On the Experimental Attainment of Optimum Conditions." *Journal of the Royal Statistical Society, Series B* 13:1–45.

Cleveland, W. S. (1993). *Visualizing Data*. Summit, NJ: Hobart Press.

Cochran, W. G., and Cox, G. M. (1957). *Experimental Designs*. 2nd ed. New York: John Wiley & Sons.

Draper, N. R. (1963). "Ridge Analysis of Response Surfaces." *Technometrics* 5:469–479.

Draper, N. R., and John, J. A. (1988). "Response Surface Designs for Quantitative and Qualitative Variables." *Technometrics* 30:423–428.

Draper, N. R., and Smith, H. (1981). *Applied Regression Analysis*. 2nd ed. New York: John Wiley & Sons.

Frankel, S. A. (1961). "Statistical Design of Experiments for Process Development of MBT." *Rubber Age* 89:453–458.

John, P. W. M. (1971). *Statistical Design and Analysis of Experiments*. New York: Macmillan.

Mead, R., and Pike, D. J. (1975). "A Review of Response Surface Methodology from a Biometric Viewpoint." *Biometrics* 31:803–851.

Meyer, D. L. (1963). "Response Surface Methodology in Education and Psychology." *Journal of Experimental Education* 31:329–336.

Myers, R. H. (1976). *Response Surface Methodology*. Blacksburg: Virginia Polytechnic Institute and State University.

Myers, R. H., and Montgomery, D. C. (1995). *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. New York: John Wiley & Sons.

Rawlings, J. O., Pantula, S. G., and Dickey, D. A. (1998). *Applied Regression Analysis: A Research Tool*. 2nd ed. New York: Springer-Verlag.

Schneider, A. M., and Stockett, A. L. (1963). "An Experiment to Select Optimum Operating Conditions on the Basis of Arbitrary Preference Ratings." *Chemical Engineering Progress Symposium Series* 59:34–38.

Chapter 109
The SCORE Procedure

Contents

Overview: SCORE Procedure **9090**
- Raw Data Set 9090
- Scoring Coefficients Data Set 9090
- Standardization of Raw Data 9091

Getting Started: SCORE Procedure **9092**

Syntax: SCORE Procedure **9095**
- PROC SCORE Statement 9096
- BY Statement 9097
- ID Statement 9098
- VAR Statement 9098

Details: SCORE Procedure **9099**
- Missing Values 9099
- Regression Parameter Estimates from PROC REG 9099
- Output Data Set 9099
- Computational Resources 9100

Examples: SCORE Procedure **9100**
- Example 109.1: Factor Scoring Coefficients 9100
- Example 109.2: Regression Parameter Estimates 9105
- Example 109.3: Custom Scoring Coefficients 9109

References **9110**

Overview: SCORE Procedure

The SCORE procedure multiplies values from two SAS data sets, one containing coefficients (for example, factor-scoring coefficients or regression coefficients) and the other containing raw data to be scored using the coefficients from the first data set. The result of this multiplication is a SAS data set containing linear combinations of the coefficients and the raw data values.

Many statistical procedures output coefficients that PROC SCORE can apply to raw data to produce scores. The new score variable is formed as a linear combination of raw data and scoring coefficients. For each observation in the raw data set, PROC SCORE multiplies the value of a variable in the raw data set by the matching scoring coefficient from the data set of scoring coefficients. This multiplication process is repeated for each variable in the VAR statement. The resulting products are then summed to produce the value of the new score variable. This entire process is repeated for each observation in the raw data set. In other words, PROC SCORE cross multiplies part of one data set with another.

Raw Data Set

The raw data set can contain the original data used to calculate the scoring coefficients, or it can contain an entirely different data set. The raw data set must contain all the variables needed to produce scores. In addition, the scoring coefficients and the variables in the raw data set that are used in scoring must have the same names. See the section "Getting Started: SCORE Procedure" on page 9092 for more information.

Scoring Coefficients Data Set

The data set containing scoring coefficients must contain two special variables: the _TYPE_ variable and the _NAME_ or _MODEL_ variable.

- The _TYPE_ variable identifies the observations that contain scoring coefficients.
- The _NAME_ or _MODEL_ variable provides a SAS name for the new score variable.

PROC SCORE first looks for a _NAME_ variable in the SCORE= input data set. If there is such a variable, the variable's value is what SCORE uses to name the new score variable. If the SCORE= data set does not have a _NAME_ variable, then PROC SCORE looks for a _MODEL_ variable.

For example, PROC FACTOR produces an output data set that contains factor-scoring coefficients. In this output data set, the scoring coefficients are identified by _TYPE_='SCORE'. For _TYPE_='SCORE', the _NAME_ variable has values of 'Factor1', 'Factor2', and so forth. PROC SCORE gives the new score variables the names Factor1, Factor2, and so forth.

As another example, the REG procedure produces an output data set that contains parameter estimates. In this output data set, the parameter estimates are identified by _TYPE_='PARMS'. The _MODEL_ variable contains the label used in the MODEL statement in PROC REG, or it uses MODEL*n* if no label is specified. This label is the name PROC SCORE gives to the new score variable.

Standardization of Raw Data

PROC SCORE automatically standardizes or centers the DATA= variables for you, based on information from the original variables and analysis from the SCORE= data set.

If the SCORE= scoring coefficients data set contains observations with _TYPE_='MEAN' and _TYPE_='STD', then PROC SCORE standardizes the raw data before scoring. For example, this type of SCORE= data set can come from PROC PRINCOMP without the COV option.

If the SCORE= scoring coefficients data set contains observations with _TYPE_='MEAN' but _TYPE_='STD' is absent, then PROC SCORE centers the raw data (the means are subtracted) before scoring. For example, this type of SCORE= data set can come from PROC PRINCOMP with the COV option.

If the SCORE= scoring coefficients data set does not contain observations with _TYPE_='MEAN' and _TYPE_='STD', or if you use the NOSTD option, then PROC SCORE does not center or standardize the raw data.

If the SCORE= scoring coefficients are obtained from observations with _TYPE_='USCORE', then PROC SCORE "standardizes" the raw data by using the uncorrected standard deviations identified by _TYPE_='USTD', and the means are not subtracted from the raw data. For example, this type of SCORE= data set can come from PROC PRINCOMP with the NOINT option. For more information about _TYPE_='USCORE' scoring coefficients in TYPE=UCORR or TYPE=UCOV output data sets, see Appendix A, "Special SAS Data Sets."

You can use PROC SCORE to score the data that were also used to generate the scoring coefficients, although more typically, scoring results are directly obtained from the OUT= data set in a procedure that computes scoring coefficients. When scoring new data, it is important to realize that PROC SCORE assumes that the new data have approximately the same scales as the original data. For example, if you specify the COV option with PROC PRINCOMP for the original analysis, the scoring coefficients in the PROC PRINCOMP OUTSTAT= data set are not appropriate for standardized data. With the COV option, PROC PRINCOMP will not output _TYPE_='STD' observations to the OUTSTAT= data set, and PROC SCORE will only subtract the means of the original (not new) variables from the new variables before multiplying. Without the COV option in PROC PRINCOMP, both the original variable means and standard deviations will be in the OUTSTAT= data set, and PROC SCORE will subtract the original variable means from the new variables and divide them by the original variable standard deviations before multiplying.

In general, procedures that output scoring coefficients in their OUTSTAT= data sets provide the necessary information for PROC SCORE to determine the appropriate standardization. However, if you use PROC SCORE with a scoring coefficients data set that you constructed without _TYPE_='MEAN' and _TYPE_='STD' observations, you might have to do the relevant centering or standardization of the new data first. If you do this, you must use the means and standard deviations of the original variables—that is, the variables that were used to generate the coefficients—not the means and standard deviations of the variables to be scored.

See the section "Getting Started: SCORE Procedure" on page 9092 for further illustration.

Getting Started: SCORE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/scoreex.sas.)

The SCORE procedure multiplies the values from two SAS data sets and creates a new data set to contain the results of the multiplication. The variables in the new data set are linear combinations of the variables in the two input data sets. Typically, one of these data sets contains raw data that you want to score, and the other data set contains scoring coefficients.

The following example demonstrates how to use the SCORE procedure to multiply values from two SAS data sets, one containing factor-scoring coefficients and the other containing raw data to be scored using the scoring coefficients.

Suppose you are interested in the performance of three different types of schools: private schools, state-run urban schools, and state-run rural schools. You want to compare the schools' performances as measured by student grades on standard tests in English, mathematics, and biology. You administer these tests and record the scores for each of the three types of schools.

The following DATA step creates the SAS data set Schools. The data are provided by Chaseling (1996).

```
data Schools;
   input Type $ English Math Biology @@;
   datalines;
p  52  55  45  p  42  49  40  p  63  64  54
p  47  50  51  p  64  69  47  p  63  67  54
p  59  63  42  p  56  61  41  p  41  44  72
p  39  42  45  p  56  63  44  p  63  73  42

   ... more lines ...

r  50  47  49  r  55  48  46  r  38  36  51
;
```

The data set Schools contains the character variable Type, which represents the type of school. Valid values are p (private schools), r (state-run rural schools), and u (state-run urban schools).

The three numeric variables in the data set are English, Math, and Biology, which represent the student scores for English, mathematics, and biology, respectively. The double trailing at sign (@@) in the INPUT statement specifies that observations are input from each line until all values are read.

The following statements invoke the FACTOR procedure to compute the data set of factor scoring coefficients. The statements perform a principal components factor analysis that uses all three numeric variables in the SAS data set Schools. The OUTSTAT= option requests that PROC FACTOR output the factor scores to the data set Scores. The NOPRINT option suppresses display of the output.

```
proc factor data=Schools score outstat=Scores noprint;
   var english math biology;
run;

proc score data=schools score=Scores out=New;
   var english math biology;
   id type;
```

```
run;
```

The SCORE procedure is then invoked using Schools as the raw data set to be scored and Scores as the scoring data set. The OUT= option creates the SAS data set New to contain the linear combinations.

The VAR statement specifies that the variables English, Math, and Biology are used in computing scores. The ID statement copies the variable Type from the Schools data set to the output data set New.

The following statements print the SAS output data set Scores, the first two observations from the original data set Schools, and the first two observations of the resulting data set New.

```
title 'OUTSTAT= Data Set from PROC FACTOR';
proc print data=Scores;
run;

title 'First Two Observations of the DATA= Data Set from PROC SCORE';
proc print data=Schools(obs=2);
run;

title 'First Two Observations of the OUT= Data Set from PROC SCORE';
proc print data=New(obs=2);
run;
```

Figure 109.1 displays the output data set Scores produced by the FACTOR procedure. The last observation (number 11) contains the scoring coefficients (_TYPE_='SCORE'). Only one factor has been retained.

Figure 109.1 Listing of the Data Set Created by PROC FACTOR

OUTSTAT= Data Set from PROC FACTOR

Obs	_TYPE_	_NAME_	English	Math	Biology
1	MEAN		55.525	52.325	50.350
2	STD		12.949	12.356	12.239
3	N		120.000	120.000	120.000
4	CORR	English	1.000	0.833	0.672
5	CORR	Math	0.833	1.000	0.594
6	CORR	Biology	0.672	0.594	1.000
7	COMMUNAL		0.881	0.827	0.696
8	PRIORS		1.000	1.000	1.000
9	EIGENVAL		2.405	0.437	0.159
10	PATTERN	Factor1	0.939	0.910	0.834
11	SCORE	Factor1	0.390	0.378	0.347

Figure 109.2 lists the first two observations of the original SAS data set (Schools).

Figure 109.2 First Two Observations of the Schools Data Set

First Two Observations of the DATA= Data Set from PROC SCORE

Obs	Type	English	Math	Biology
1	p	52	55	45
2	p	42	49	40

Figure 109.3 lists the first two observations of the output data set New created by PROC SCORE.

Figure 109.3 Listing of the New Data Set

First Two Observations of the OUT= Data Set from PROC SCORE

Obs	Type	Factor1
1	p	-0.17604
2	p	-0.80294

The score variable Factor1 in the New data set is named according to the value of the _NAME_ variable in the Scores data set. The values of the variable Factor1 are computed as follows: the DATA= data set variables are standardized using the same means and standard deviations that PROC FACTOR used when extracting the factors because the Scores data set contains observations with _TYPE_='MEAN' and _TYPE_='STD'.

Note that in order to correctly use standardized scoring coefficients created by other procedures such as PROC FACTOR in this example, the data to be scored must be standardized in the same way that the data were standardized when the scoring coefficients were computed. Otherwise, the resulting scores might be incorrect. PROC SCORE does this automatically if the SCORE= data set is the original OUTSTAT= data set output from the procedure creating the scoring coefficients.

These standardized variables are then multiplied by their respective standardized scoring coefficients from the data set Scores. These products are summed over all three variables, and the sum is the value of the new variable Factor1. The first two values of the scored variable Factor1 are obtained as follows:

$$\left(\frac{(52-55.525)}{12.949}\times 0.390\right)+\left(\frac{(55-52.325)}{12.356}\times 0.378\right)+\left(\frac{(45-50.350)}{12.239}\times 0.347\right)=-0.17604$$

$$\left(\frac{(42-55.525)}{12.949}\times 0.390\right)+\left(\frac{(49-52.325)}{12.356}\times 0.378\right)+\left(\frac{(40-50.350)}{12.239}\times 0.347\right)=-0.80294$$

The following statements request that the SGPLOT procedure produce a horizontal bar chart of the variable Type. The length of each bar represents the mean of the variable Factor1.

```
title 'Mean Score of Variable Factor1 by Each Type of Schools';
proc sgplot data=New;
   hbar type / stat = mean response=Factor1;
run;
```

Figure 109.4 displays the mean score of the variable Factor1 for each of the three school types. For private schools (Type=p), the average value of the variable Factor1 is 0.384, while for state-run schools the average values are much lower. The state-run urban schools (Type=u) have the lowest mean value of –0.202, and the state-run rural schools (Type=r) have a mean value of –0.183.

Figure 109.4 Bar Chart of School Type

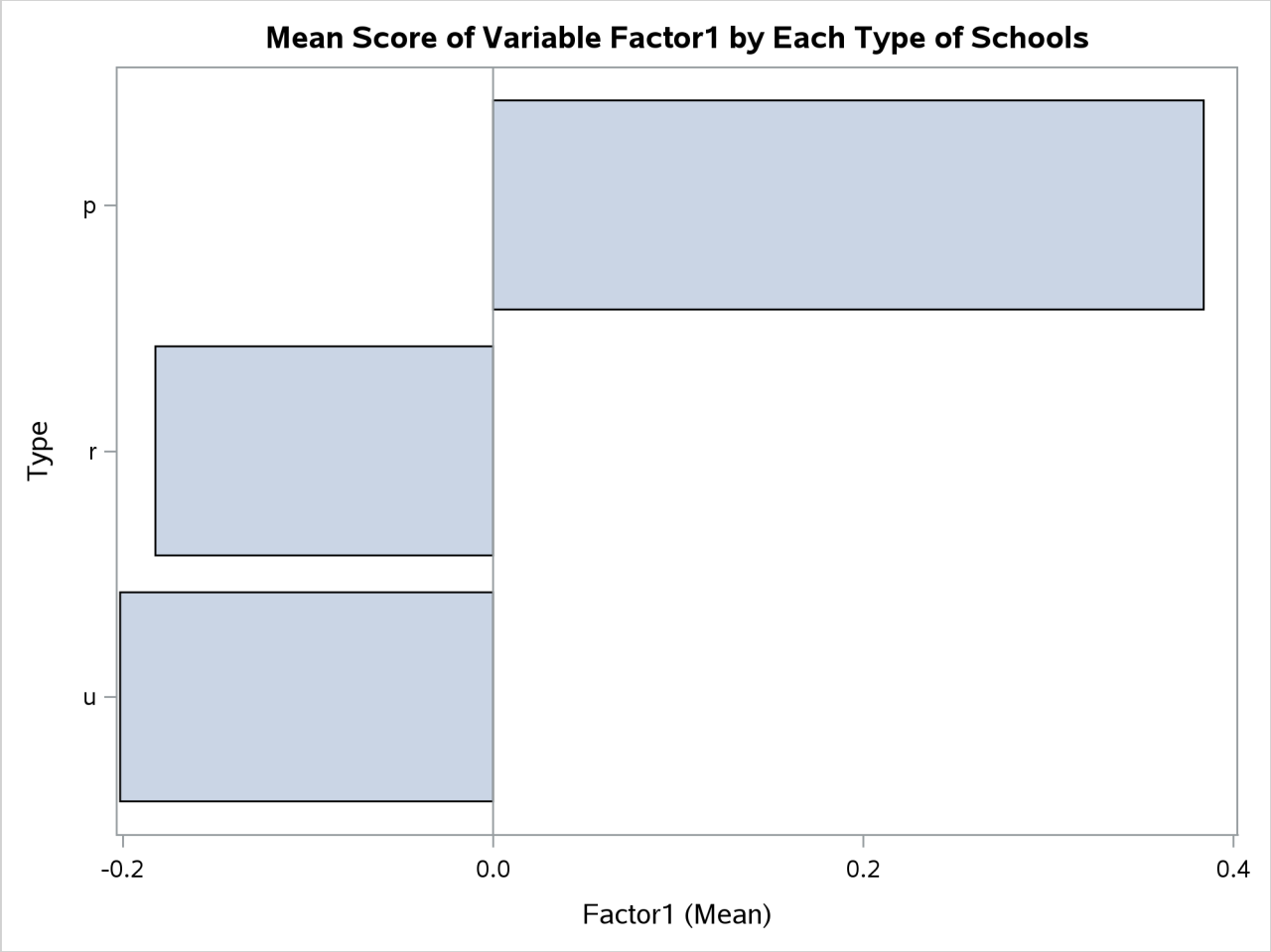

Syntax: SCORE Procedure

The following statements are available in the SCORE procedure:

PROC SCORE DATA=*SAS-data-set* < *options* > **;**
  BY *variables* **;**
  ID *variables* **;**
  VAR *variables* **;**

The only required statement is the PROC SCORE statement. The BY, ID, and VAR statements are described following the PROC SCORE statement.

PROC SCORE Statement

PROC SCORE DATA=*SAS-data-set* < *options* > **;**

The PROC SCORE statement invokes the SCORE procedure. Table 109.1 summarizes the options available in the PROC SCORE statement.

Table 109.1 PROC SCORE Statement Options

Option	Description
DATA=	Names the input SAS data set containing the raw data to score
NOSTD	Suppresses centering and scaling of the raw data
OUT=	Specifies the name of the SAS data set
PREDICT	Treats coefficients of –1 as 0
RESIDUAL	Reverses the sign of each score
SCORE=	Names the data set containing the scoring coefficients
TYPE=	Specifies the observations that contain scoring coefficients

You can specify the following options in the PROC SCORE statement.

DATA=*SAS-data-set*
names the input SAS data set containing the raw data to score. This option is required.

NOSTD
suppresses centering and scaling of the raw data. Ordinarily, if PROC SCORE finds _TYPE_='MEAN', _TYPE_= 'USCORE', _TYPE_='USTD', or _TYPE_='STD' observations in the SCORE= data set, the procedure uses these to standardize the raw data before scoring.

OUT=*SAS-data-set*
specifies the name of the SAS data set created by PROC SCORE. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. If the OUT= option is omitted, PROC SCORE still creates an output data set and automatically names it according to the DATA*n* convention, as if you omitted a data set name in a DATA statement.

PREDICT
specifies that PROC SCORE should treat coefficients of –1 in the SCORE= data set as 0. In regression applications, the dependent variable is coded with a coefficient of –1. Applied directly to regression results, PROC SCORE produces negative residuals (see the description of the RESIDUAL option, which follows); the PREDICT option produces predicted values instead.

RESIDUAL
reverses the sign of each score. Applied directly to regression results, PROC SCORE produces negative residuals (PREDICT–ACTUAL); the RESIDUAL option produces positive residuals (ACTUAL–PREDICT) instead.

SCORE=*SAS-data-set*

names the data set containing the scoring coefficients. If you omit the SCORE= option, the most recently created SAS data set is used. This data set must have two special variables: _TYPE_ and either _NAME_ or _MODEL_.

TYPE=*name* **or** *'string'*

specifies the observations in the SCORE= data set that contain scoring coefficients. The TYPE= procedure option is unrelated to the data set option that has the same name. PROC SCORE examines the values of the special variable _TYPE_ in the SCORE= data set. When the value of _TYPE_ matches TYPE=*name*, the observation in the SCORE= data set is used to score the raw data in the DATA= data set.

If you omit the TYPE= option, scoring coefficients are read from observations with either _TYPE_='SCORE' or _TYPE_='USCORE'. Because the default for PROC SCORE is TYPE=SCORE, you need not specify the TYPE= option for factor scoring or for computing scores from OUTSTAT= data sets from the CANCORR, CANDISC, PRINCOMP, or VARCLUS procedure. When you use regression coefficients from PROC REG, specify TYPE=PARMS.

The maximum length of the argument specified in the TYPE= option depends on the length defined by the VALIDVARNAME= SAS system option. For additional information, see *SAS System Options: Reference*.

Note that the TYPE= option setting is not case sensitive. For example, the two option settings TYPE='MyScore' and TYPE='myscore' are equivalent.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SCORE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SCORE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

You can specify a BY statement to apply separate groups of scoring coefficients to the entire DATA= data set.

If the DATA= data set does not contain any of the BY variables, the entire DATA= data set is scored by each BY group of scoring coefficients in the SCORE= data set.

If the DATA= data set contains some but not all of the BY variables, or if some BY variables do not have the same type or length in the DATA= data set as in the SCORE= data set, then PROC SCORE prints an error message and stops.

If all the BY variables appear in the DATA= data set with the same type and length as in the SCORE= data set, then each BY group in the DATA= data set is scored using scoring coefficients from the corresponding BY group in the SCORE= data set. The BY groups in the DATA= data set must be in the same order as in the SCORE= data set. All BY groups in the DATA= data set must also appear in the SCORE= data set. If you do not specify the NOTSORTED option, some BY groups can appear in the SCORE= data set but not in the DATA= data set; such BY groups are not used in computing scores.

ID Statement

ID *variables* **;**

The ID statement identifies variables from the DATA= data set to be included in the OUT= data set. If there is no ID statement, all variables from the DATA= data set are included in the OUT= data set. The ID variables can be character or numeric.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the variables to be used in computing scores. These variables must be in both the DATA= and SCORE= input data sets and must be numeric. If you do not specify a VAR statement, the procedure uses all numeric variables in the SCORE= data set. You should almost always specify a VAR statement with PROC SCORE because you would rarely use all the numeric variables in your data set to compute scores.

Details: SCORE Procedure

Missing Values

If one of the scoring variables in the DATA= data set has a missing value for an observation, all the scores have missing values for that observation. The exception to this criterion is that if the PREDICT option is specified, the variable with a coefficient of –1 can tolerate a missing value and still produce a prediction score. Also, a variable with a coefficient of 0 can tolerate a missing value.

If a scoring coefficient in the SCORE= data set has a missing value for an observation, the coefficient is not used in creating the new score variable for the observation. In other words, missing values of scoring coefficients are treated as zeros. This treatment affects only the observation in which the missing value occurs.

Regression Parameter Estimates from PROC REG

If the SCORE= data set is an OUTEST= data set produced by PROC REG and if you specify TYPE=PARMS, the interpretation of the new score variables depends on the PROC SCORE options chosen and the variables listed in the VAR statement. If the VAR statement contains only the independent variables used in a model in PROC REG, the new score variables give the predicted values. If the VAR statement contains the dependent variables and the independent variables used in a model in PROC REG, the interpretation of the new score variables depends on the PROC SCORE options chosen. If you omit both the PREDICT and the RESIDUAL options, the new score variables give negative residuals (PREDICT–ACTUAL). If you specify the RESIDUAL option, the new score variables give positive residuals (ACTUAL–PREDICT). If you specify the PREDICT option, the new score variables give predicted values.

Unless you specify the NOINT option for PROC REG, the OUTEST= data set contains the variable Intercept. The SCORE procedure uses the intercept value in computing the scores.

Output Data Set

PROC SCORE produces an output data set but displays no output. The output OUT= data set contains the following variables:

- the ID variables, if any
- all variables from the DATA= data set, if no ID variables are specified
- the BY variables, if any
- the new score variables, named from the _NAME_ or _MODEL_ values in the SCORE= data set

Computational Resources

Let

v = number of variables used in computing scores
s = number of new score variables
b = maximum number of new score variables in a BY group
n = original input value

Memory

The array storage required is approximately $8(4v + (3+v)b + s)$ bytes. When you do not use BY processing, the array storage required is approximately $8(4v + (4+v)s)$ bytes.

Time

The time required to construct the scoring matrix is roughly proportional to vs, and the time needed to compute the scores is roughly proportional to nvs.

Examples: SCORE Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The following three examples use a subset of the Fitness data set. The complete data set is given in Chapter 105, "The REG Procedure."

Example 109.1: Factor Scoring Coefficients

(View the complete code for this example (scoreex1.sas) in the example repository.)

This example shows how to use PROC SCORE with factor scoring coefficients. First, the FACTOR procedure produces an output data set containing scoring coefficients in observations identified by _TYPE_='SCORE'. These data, together with the original data set Fitness, are supplied to PROC SCORE, resulting in a data set containing scores Factor1 and Factor2. The following statements produce Output 109.1.1 through Output 109.1.3:

```
   /* This data set contains only the first 12 observations   */
   /* from the full data set used in the chapter on PROC REG. */
data Fitness;
   input Age Weight Oxygen RunTime RestPulse RunPulse @@;
   datalines;
44 89.47  44.609 11.37 62 178     40 75.07  45.313 10.07 62 185
44 85.84  54.297  8.65 45 156     42 68.15  59.571  8.17 40 166
38 89.02  49.874  9.22 55 178     47 77.45  44.811 11.63 58 176
40 75.98  45.681 11.95 70 176     43 81.19  49.091 10.85 64 162
44 81.42  39.442 13.08 63 174     38 81.87  60.055  8.63 48 170
```

```
44 73.03  50.541 10.13 45 168     45 87.66  37.388 14.03 56 186
;

proc factor data=Fitness outstat=FactOut
            method=prin rotate=varimax score;
   var Age Weight RunTime RunPulse RestPulse;
   title 'Factor Scoring Example';
run;

proc print data=FactOut;
   title2 'Data Set from PROC FACTOR';
run;

proc score data=Fitness score=FactOut out=FScore;
   var Age Weight RunTime RunPulse RestPulse;
run;

proc print data=FScore;
   title2 'Data Set from PROC SCORE';
run;
```

Output 109.1.1 shows the PROC FACTOR output. The scoring coefficients for the two factors are shown at the end of the PROC FACTOR output.

Output 109.1.1 Creating an OUTSTAT= Data Set with PROC FACTOR

Factor Scoring Example

The FACTOR Procedure

Input Data Type	Raw Data
Number of Records Read	12
Number of Records Used	12
N for Significance Tests	12

Factor Scoring Example

The FACTOR Procedure
Initial Factor Method: Principal Components

Prior Communality Estimates: ONE

Eigenvalues of the Correlation Matrix:
Total = 5 Average = 1

	Eigenvalue	Difference	Proportion	Cumulative
1	2.30930638	1.11710686	0.4619	0.4619
2	1.19219952	0.30997249	0.2384	0.7003
3	0.88222702	0.37965990	0.1764	0.8767
4	0.50256713	0.38886717	0.1005	0.9773
5	0.11369996		0.0227	1.0000

Output 109.1.1 *continued*

2 factors will be retained by the MINEIGEN criterion.

Factor Pattern		
	Factor1	Factor2
Age	0.29795	0.93675
Weight	0.43282	-0.17750
RunTime	0.91983	0.28782
RunPulse	0.72671	-0.38191
RestPulse	0.81179	-0.23344

Variance Explained by Each Factor	
Factor1	Factor2
2.3093064	1.1921995

Final Communality Estimates: Total = 3.501506				
Age	Weight	RunTime	RunPulse	RestPulse
0.96628351	0.21883401	0.92893333	0.67396207	0.71349297

Factor Scoring Example

The FACTOR Procedure
Rotation Method: Varimax

Orthogonal Transformation Matrix		
	1	2
1	0.92536	0.37908
2	-0.37908	0.92536

Rotated Factor Pattern		
	Factor1	Factor2
Age	-0.07939	0.97979
Weight	0.46780	-0.00018
RunTime	0.74207	0.61503
RunPulse	0.81725	-0.07792
RestPulse	0.83969	0.09172

Variance Explained by Each Factor	
Factor1	Factor2
2.1487753	1.3527306

Final Communality Estimates: Total = 3.501506				
Age	Weight	RunTime	RunPulse	RestPulse
0.96628351	0.21883401	0.92893333	0.67396207	0.71349297

Output 109.1.1 *continued*

Factor Scoring Example

The FACTOR Procedure
Rotation Method: Varimax

Scoring Coefficients Estimated by Regression

Squared Multiple Correlations of the Variables with Each Factor	
Factor1	**Factor2**
1.0000000	1.0000000

Standardized Scoring Coefficients		
	Factor1	**Factor2**
Age	-0.17846	0.77600
Weight	0.22987	-0.06672
RunTime	0.27707	0.37440
RunPulse	0.41263	-0.17714
RestPulse	0.39952	-0.04793

Output 109.1.2 lists the OUTSTAT= data set from PROC FACTOR. Note that observations 18 and 19 have _TYPE_='SCORE'. Observations 1 and 2 have _TYPE_='MEAN' and _TYPE_='STD', respectively. These four observations are used by PROC SCORE.

Output 109.1.2 OUTSTAT= Data Set from PROC FACTOR Reproduced with PROC PRINT

Factor Scoring Example
Data Set from PROC FACTOR

Obs	_TYPE_	_NAME_	Age	Weight	RunTime	RunPulse	RestPulse
1	MEAN		42.4167	80.5125	10.6483	172.917	55.6667
2	STD		2.8431	6.7660	1.8444	8.918	9.2769
3	N		12.0000	12.0000	12.0000	12.000	12.0000
4	CORR	Age	1.0000	0.0128	0.5005	-0.095	-0.0080
5	CORR	Weight	0.0128	1.0000	0.2637	0.173	0.2396
6	CORR	RunTime	0.5005	0.2637	1.0000	0.556	0.6620
7	CORR	RunPulse	-0.0953	0.1731	0.5555	1.000	0.4853
8	CORR	RestPulse	-0.0080	0.2396	0.6620	0.485	1.0000
9	COMMUNAL		0.9663	0.2188	0.9289	0.674	0.7135
10	PRIORS		1.0000	1.0000	1.0000	1.000	1.0000
11	EIGENVAL		2.3093	1.1922	0.8822	0.503	0.1137
12	UNROTATE	Factor1	0.2980	0.4328	0.9198	0.727	0.8118
13	UNROTATE	Factor2	0.9368	-0.1775	0.2878	-0.382	-0.2334
14	TRANSFOR	Factor1	0.9254	-0.3791	.	.	.
15	TRANSFOR	Factor2	0.3791	0.9254	.	.	.
16	PATTERN	Factor1	-0.0794	0.4678	0.7421	0.817	0.8397
17	PATTERN	Factor2	0.9798	-0.0002	0.6150	-0.078	0.0917
18	SCORE	Factor1	-0.1785	0.2299	0.2771	0.413	0.3995
19	SCORE	Factor2	0.7760	-0.0667	0.3744	-0.177	-0.0479

Since the PROC SCORE statement does not contain the NOSTD option, the data in the Fitness data set are standardized before scoring. For each variable specified in the VAR statement, the mean and standard deviation are obtained from the FactOut data set. For each observation in the Fitness data set, the variables are then standardized. For example, for observation 1 in the Fitness data set, the variable Age is standardized to $0.5569 = [(44 - 42.4167)/2.8431]$.

After the data in the Fitness data set are standardized, the standardized values of the variables in the VAR statement are multiplied by the matching coefficients in the FactOut data set, and the resulting products are summed. This sum is output as a value of the new score variable.

Output 109.1.3 displays the FScore data set produced by PROC SCORE. This data set contains the variables Age, Weight, Oxygen, RunTime, RestPulse, and RunPulse from the Fitness data set. It also contains Factor1 and Factor2, the two new score variables.

Output 109.1.3 OUT= Data Set from PROC SCORE Reproduced with PROC PRINT

Factor Scoring Example
Data Set from PROC SCORE

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	Factor1	Factor2
1	44	89.47	44.609	11.37	62	178	0.82129	0.35663
2	40	75.07	45.313	10.07	62	185	0.71173	-0.99605
3	44	85.84	54.297	8.65	45	156	-1.46064	0.36508
4	42	68.15	59.571	8.17	40	166	-1.76087	-0.27657
5	38	89.02	49.874	9.22	55	178	0.55819	-1.67684
6	47	77.45	44.811	11.63	58	176	-0.00113	1.40715
7	40	75.98	45.681	11.95	70	176	0.95318	-0.48598
8	43	81.19	49.091	10.85	64	162	-0.12951	0.36724
9	44	81.42	39.442	13.08	63	174	0.66267	0.85740
10	38	81.87	60.055	8.63	48	170	-0.44496	-1.53103
11	44	73.03	50.541	10.13	45	168	-1.11832	0.55349
12	45	87.66	37.388	14.03	56	186	1.20836	1.05948

Example 109.2: Regression Parameter Estimates

(View the complete code for this example (scoreex2.sas) in the example repository.)

In this example, PROC REG computes regression parameter estimates for the Fitness data. (See Example 109.1 to for more information about how to create the Fitness data set.) The parameter estimates are output to a data set and used as scoring coefficients. For the first part of this example, PROC SCORE is used to score the Fitness data, which are the same data used in the regression.

In the second part of this example, PROC SCORE is used to score a new data set, Fitness2. For PROC SCORE, the TYPE= specification is PARMS, and the names of the score variables are found in the variable _MODEL_, which gets its values from the model label. The following code produces Output 109.2.1 through Output 109.2.3:

```
proc reg data=Fitness outest=RegOut;
   OxyHat: model Oxygen=Age Weight RunTime RunPulse RestPulse;
   title 'Regression Scoring Example';
run;

proc print data=RegOut;
   title2 'OUTEST= Data Set from PROC REG';
run;

proc score data=Fitness score=RegOut out=RScoreP type=parms;
   var Age Weight RunTime RunPulse RestPulse;
run;

proc print data=RScoreP;
   title2 'Predicted Scores for Regression';
run;
```

```
proc score data=Fitness score=RegOut out=RScoreR type=parms;
   var Oxygen Age Weight RunTime RunPulse RestPulse;
run;

proc print data=RScoreR;
   title2 'Negative Residual Scores for Regression';
run;
```

Output 109.2.1 shows the PROC REG output. The column labeled "Parameter Estimates" lists the parameter estimates. These estimates are output to the RegOut data set.

Output 109.2.1 Creating an OUTEST= Data Set with PROC REG

Regression Scoring Example

The REG Procedure
Model: OxyHat
Dependent Variable: Oxygen

Number of Observations Read	12
Number of Observations Used	12

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	509.62201	101.92440	15.80	0.0021
Error	6	38.70060	6.45010		
Corrected Total	11	548.32261			

Root MSE	2.53970	**R-Square**	0.9294
Dependent Mean	48.38942	**Adj R-Sq**	0.8706
Coeff Var	5.24847		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	151.91550	31.04738	4.89	0.0027
Age	1	-0.63045	0.42503	-1.48	0.1885
Weight	1	-0.10586	0.11869	-0.89	0.4068
RunTime	1	-1.75698	0.93844	-1.87	0.1103
RunPulse	1	-0.22891	0.12169	-1.88	0.1090
RestPulse	1	-0.17910	0.13005	-1.38	0.2176

Output 109.2.2 lists the RegOut data set. Note that _TYPE_='PARMS' and _MODEL_='OXYHAT', which are from the label in the MODEL statement in PROC REG.

Output 109.2.2 OUTEST= Data Set from PROC REG Reproduced with PROC PRINT

Regression Scoring Example
OUTEST= Data Set from PROC REG

Obs	_MODEL_	_TYPE_	_DEPVAR_	_RMSE_	Intercept	Age	Weight	RunTime	RunPulse	RestPulse	Oxygen
1	OxyHat	PARMS	Oxygen	2.53970	151.916	-0.63045	-0.10586	-1.75698	-0.22891	-0.17910	-1

Output 109.2.3 lists the data sets created by PROC SCORE. Since the SCORE= data set does not contain observations with _TYPE_='MEAN' or _TYPE_='STD', the data in the Fitness data set are not standardized before scoring. The SCORE= data set contains the variable Intercept, so this intercept value is used in computing the score. To produce the RScoreP data set, the VAR statement in PROC SCORE includes only the independent variables from the model in PROC REG. As a result, the OxyHat variable contains predicted values. To produce the RScoreR data set, the VAR statement in PROC SCORE includes both the dependent variables and the independent variables from the model in PROC REG. As a result, the OxyHat variable contains negative residuals (PREDICT–ACTUAL) as shown in Output 109.2.4. If the RESIDUAL option is specified, the variable OxyHat contains positive residuals (ACTUAL–PREDICT). If the PREDICT option is specified, the OxyHat variable contains predicted values.

Output 109.2.3 Predicted Scores from the OUT= Data Set Created by PROC SCORE

Regression Scoring Example
Predicted Scores for Regression

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	OxyHat
1	44	89.47	44.609	11.37	62	178	42.8771
2	40	75.07	45.313	10.07	62	185	47.6050
3	44	85.84	54.297	8.65	45	156	56.1211
4	42	68.15	59.571	8.17	40	166	58.7044
5	38	89.02	49.874	9.22	55	178	51.7386
6	47	77.45	44.811	11.63	58	176	42.9756
7	40	75.98	45.681	11.95	70	176	44.8329
8	43	81.19	49.091	10.85	64	162	48.6020
9	44	81.42	39.442	13.08	63	174	41.4613
10	38	81.87	60.055	8.63	48	170	56.6171
11	44	73.03	50.541	10.13	45	168	52.1299
12	45	87.66	37.388	14.03	56	186	37.0080

Output 109.2.4 Residual Scores from the OUT= Data Set Created by PROC SCORE

Regression Scoring Example
Negative Residual Scores for Regression

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	OxyHat
1	44	89.47	44.609	11.37	62	178	-1.73195
2	40	75.07	45.313	10.07	62	185	2.29197
3	44	85.84	54.297	8.65	45	156	1.82407
4	42	68.15	59.571	8.17	40	166	-0.86657
5	38	89.02	49.874	9.22	55	178	1.86460
6	47	77.45	44.811	11.63	58	176	-1.83542
7	40	75.98	45.681	11.95	70	176	-0.84811
8	43	81.19	49.091	10.85	64	162	-0.48897
9	44	81.42	39.442	13.08	63	174	2.01935
10	38	81.87	60.055	8.63	48	170	-3.43787
11	44	73.03	50.541	10.13	45	168	1.58892
12	45	87.66	37.388	14.03	56	186	-0.38002

The second part of this example uses the parameter estimates to score a new data set. The following statements produce Output 109.2.5 and Output 109.2.6:

```
   /* The FITNESS2 data set contains observations 13-16 from */
   /* the FITNESS data set used in EXAMPLE 2 in the PROC REG */
   /* chapter.                                               */
data Fitness2;
   input Age Weight Oxygen RunTime RestPulse RunPulse;
   datalines;
45  66.45  44.754  11.12  51  176
47  79.15  47.273  10.60  47  162
54  83.12  51.855  10.33  50  166
49  81.42  49.156   8.95  44  180
;

proc print data=Fitness2;
   title 'Regression Scoring Example';
   title2 'New Raw Data Set to be Scored';
run;

proc score data=Fitness2 score=RegOut out=NewPred type=parms
           nostd predict;
   var Oxygen Age Weight RunTime RunPulse RestPulse;
run;

proc print data=NewPred;
   title2 'Predicted Scores for Regression';
   title3 'for Additional Data from FITNESS2';
run;
```

Output 109.2.5 lists the Fitness2 data set.

Output 109.2.5 Listing of the Fitness2 Data Set

Regression Scoring Example
New Raw Data Set to be Scored

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse
1	45	66.45	44.754	11.12	51	176
2	47	79.15	47.273	10.60	47	162
3	54	83.12	51.855	10.33	50	166
4	49	81.42	49.156	8.95	44	180

PROC SCORE scores the Fitness2 data set by using the parameter estimates in the RegOut data set. These parameter estimates result from fitting a regression equation to the Fitness data set. The NOSTD option is specified, so the raw data are not standardized before scoring. (However, the NOSTD option is not necessary here. The SCORE= data set does not contain observations with _TYPE_='MEAN' or _TYPE_='STD', so standardization is not performed.) The VAR statement contains the dependent variables and the independent variables used in PROC REG. In addition, the PREDICT option is specified. This combination gives predicted values for the new score variable. The name of the new score variable is OxyHat, from the value of the _MODEL_ variable in the SCORE= data set. Output 109.2.6 shows the data set produced by PROC SCORE.

Output 109.2.6 Predicted Scores from the OUT= Data Set Created by PROC SCORE and Reproduced Using PROC PRINT

Regression Scoring Example
Predicted Scores for Regression
for Additional Data from FITNESS2

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	OxyHat
1	45	66.45	44.754	11.12	51	176	47.5507
2	47	79.15	47.273	10.60	47	162	49.7802
3	54	83.12	51.855	10.33	50	166	43.9682
4	49	81.42	49.156	8.95	44	180	47.5949

Example 109.3: Custom Scoring Coefficients

(View the complete code for this example (scoreex3.sas) in the example repository.)

This example uses a specially created custom scoring data set and produces Output 109.3.1 and Output 109.3.2. The first scoring coefficient creates a variable that is Age–Weight; the second scoring coefficient evaluates the variable RunPulse–RstPulse; and the third scoring coefficient totals all six variables. Since the scoring coefficients data set (data set A) does not contain any observations with _TYPE_='MEAN' or _TYPE_='STD', the data in the Fitness data set (see Example 109.1) are not standardized before scoring.

The following statements produce Output 109.3.1 and Output 109.3.2:

```
data A;
   input _type_ $ _name_ $
         Age Weight RunTime RunPulse RestPulse;
   datalines;
SCORE  AGE_WGT  1 -1  0  0  0
SCORE  RUN_RST  0  0  0  1 -1
SCORE  TOTAL    1  1  1  1  1
;

proc print data=A;
   title 'Constructed Scoring Example';
   title2 'Scoring Coefficients';
run;

proc score data=Fitness score=A out=B;
   var Age Weight RunTime RunPulse RestPulse;
run;

proc print data=B;
   title2 'Scored Data';
run;
```

Output 109.3.1 Custom Scoring Data Set and Scored Fitness Data: PROC PRINT

Constructed Scoring Example
Scoring Coefficients

Obs	_type_	_name_	Age	Weight	RunTime	RunPulse	RestPulse
1	SCORE	AGE_WGT	1	-1	0	0	0
2	SCORE	RUN_RST	0	0	0	1	-1
3	SCORE	TOTAL	1	1	1	1	1

Output 109.3.2 Custom Scored Fitness Data: PROC PRINT

Constructed Scoring Example
Scored Data

Obs	Age	Weight	Oxygen	RunTime	RestPulse	RunPulse	AGE_WGT	RUN_RST	TOTAL
1	44	89.47	44.609	11.37	62	178	-45.47	116	384.84
2	40	75.07	45.313	10.07	62	185	-35.07	123	372.14
3	44	85.84	54.297	8.65	45	156	-41.84	111	339.49
4	42	68.15	59.571	8.17	40	166	-26.15	126	324.32
5	38	89.02	49.874	9.22	55	178	-51.02	123	369.24
6	47	77.45	44.811	11.63	58	176	-30.45	118	370.08
7	40	75.98	45.681	11.95	70	176	-35.98	106	373.93
8	43	81.19	49.091	10.85	64	162	-38.19	98	361.04
9	44	81.42	39.442	13.08	63	174	-37.42	111	375.50
10	38	81.87	60.055	8.63	48	170	-43.87	122	346.50
11	44	73.03	50.541	10.13	45	168	-29.03	123	340.16
12	45	87.66	37.388	14.03	56	186	-42.66	130	388.69

References

Chaseling, J. (1996). “Standard Test Results of Students at Three Types of Schools.” Sample data, Faculty of Environmental Sciences, Griffith University, Queensland, Australia.

Chapter 110
The SEQDESIGN Procedure

Contents

Overview: SEQDESIGN Procedure **9112**
- Features of the SEQDESIGN Procedure 9115
- Output from the SEQDESIGN Procedure 9116
- Boundaries for Group Sequential Designs 9117
- Group Sequential Methods 9118

Getting Started: SEQDESIGN Procedure **9120**

Syntax: SEQDESIGN Procedure **9127**
- PROC SEQDESIGN Statement 9127
- DESIGN Statement 9131
- SAMPLESIZE Statement 9137

Details: SEQDESIGN Procedure **9147**
- Fixed-Sample Clinical Trials 9147
- One-Sided Fixed-Sample Tests in Clinical Trials 9151
- Two-Sided Fixed-Sample Tests in Clinical Trials 9153
- Group Sequential Methods 9156
- Statistical Assumptions for Group Sequential Designs 9159
- Boundary Scales 9161
- Boundary Variables 9163
- Type I and Type II Errors 9166
- Unified Family Methods 9168
- Haybittle-Peto Method 9173
- Whitehead Methods 9173
- Error Spending Methods 9177
- Acceptance ($\boldsymbol{\beta}$) Boundary 9179
- Boundary Adjustments for Overlapping Lower and Upper $\boldsymbol{\beta}$ Boundaries 9182
- Specified and Derived Parameters 9183
- Applicable Boundary Keys 9184
- Sample Size Computation 9185
- Applicable One-Sample Tests and Sample Size Computation 9193
- Applicable Two-Sample Tests and Sample Size Computation 9195
- Applicable Regression Parameter Tests and Sample Size Computation 9203
- Aspects of Group Sequential Designs 9206
- Summary of Methods in Group Sequential Designs 9209
- Table Output 9211
- ODS Table Names 9215
- Graphics Output 9215

ODS Graphics 9217
Examples: SEQDESIGN Procedure **9217**
Example 110.1: Creating Fixed-Sample Designs 9218
Example 110.2: Creating a One-Sided O'Brien-Fleming Design 9224
Example 110.3: Creating Two-Sided Pocock and O'Brien-Fleming Designs 9230
Example 110.4: Generating Graphics Display for Sequential Designs 9238
Example 110.5: Creating Designs Using Haybittle-Peto Methods 9244
Example 110.6: Creating Designs with Various Stopping Criteria 9252
Example 110.7: Creating Whitehead's Triangular Designs 9262
Example 110.8: Creating a One-Sided Error Spending Design 9270
Example 110.9: Creating Designs with Various Number of Stages 9277
Example 110.10: Creating Two-Sided Error Spending Designs with and without Overlapping Lower and Upper $\boldsymbol{\beta}$ Boundaries 9284
Example 110.11: Creating a Two-Sided Asymmetric Error Spending Design with Early Stopping to Reject H_0 9290
Example 110.12: Creating a Two-Sided Asymmetric Error Spending Design with Early Stopping to Reject or Accept H_0 9298
Example 110.13: Creating a Design with a Nonbinding Beta Boundary 9305
Example 110.14: Computing Sample Size for Survival Data That Have Uniform Accrual 9310
Example 110.15: Computing Sample Size for Survival Data with Truncated Exponential Accrual 9315
References **9320**

Overview: SEQDESIGN Procedure

The purpose of the SEQDESIGN procedure is to design interim analyses for clinical trials. Clinical trials are experiments on human subjects to demonstrate the efficacy and safety of new drugs or treatments. A simple example is a trial to test the effectiveness of a new drug in humans by comparing the outcomes in a group of patients who receive the new drug with the outcomes in a comparable group of patients who receive a placebo.

A clinical trial is conducted according to a plan called a *protocol*. A protocol details the objectives of the trial, the data collection process, and the analysis. The protocol specifies the null hypothesis and an alternative hypothesis, a test statistic, the probability α of a Type I error, the probability β of a Type II error, the sample size needed to attain a specified power of $1 - \beta$ at an alternative reference, and critical values that are associated with the test statistic.

In a fixed-sample trial, data about all individuals are first collected and then examined at the end of the study. Most major trials have committees that periodically monitor safety and efficacy data during the trial and recommend that a trial be stopped for safety concerns such as an unacceptable toxicity level. In certain situations, the committee might recommend that a trial be stopped for efficacy. In contrast to a fixed-sample trial, a group sequential trial provides for interim analyses before the completion of the trial while maintaining the specified overall Type I and Type II error probabilities.

A group sequential trial is most useful in situations where it is important to monitor the trial to prevent unnecessary exposure of patients to an unsafe new drug, or alternatively to a placebo treatment if the new

drug shows significant improvement. In most cases, if a group sequential trial stops early for safety concerns, fewer patients are exposed to the new treatment than in the fixed-sample trial. If a trial stops early for efficacy reasons, the new treatment is available sooner than it would be in a fixed-sample trial. Early stopping can also save time and resources.

A group sequential design provides detailed specifications for a group sequential trial. In addition to the usual specification for a fixed-sample design, it provides the total number of stages (the number of interim stages plus a final stage) and a stopping criterion to reject, to accept, or to either reject or accept the null hypothesis at each interim stage. It also provides critical values and the sample size at each stage for the trial.

At each interim stage, the data collected at the current stage in addition to the data collected at previous stages are analyzed, and statistics such as a maximum likelihood test statistic and its associated standard error are computed. The test statistic is then compared with critical values that are generated from the sequential design, and the trial is stopped or continued. If a trial continues to the final stage, the null hypothesis is either rejected or accepted. The critical values for each stage are chosen in such a way that the overall α and β are maintained at the specified levels.

Figure 110.1 shows a two-sided symmetric group sequential trial that stops early to reject the null hypothesis that the parameter Trt is zero.

Figure 110.1 Sequential Plot for Two-Sided Test

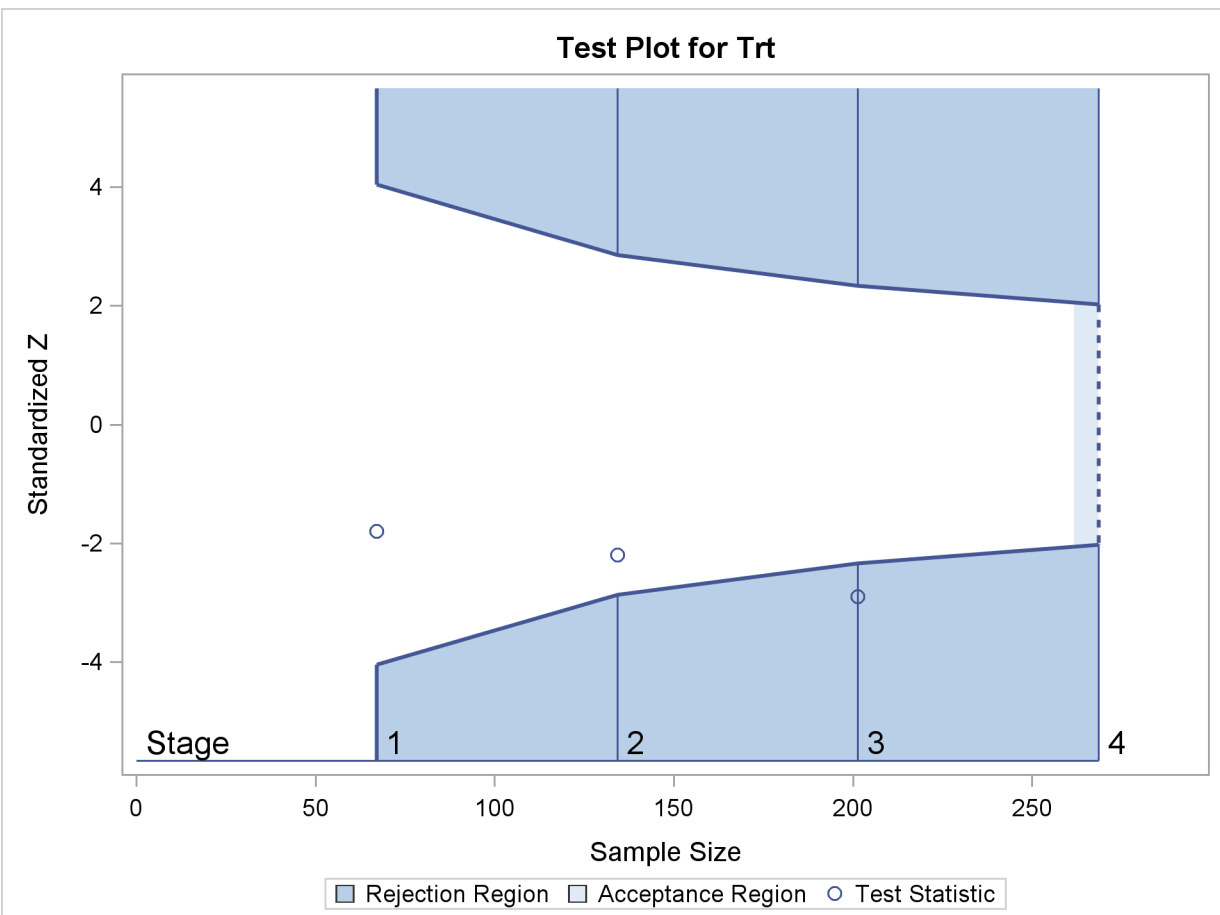

The trial has four stages, which are indicated by the vertical lines labeled 1, 2, 3, and 4. With early stopping

to reject the null hypothesis, the lower rejection boundary is constructed by connecting the lower critical values for the stages. Similarly, the upper rejection boundary is constructed by connecting the upper critical values for the stages. The horizontal axis indicates the sample size for the group sequential trial, and the vertical axis indicates the values of the test statistic on the standardized Z scale.

At each interim stage, if the test statistic falls into a rejection region (the darker shaded areas in Figure 110.1), the trial stops and the null hypothesis is rejected. Otherwise, the trial continues to the next stage. At the final stage (stage 4), the null hypothesis is rejected if Z falls into a rejection region. Otherwise, the null hypothesis is not rejected. In Figure 110.1, the test statistic does not fall into the rejection regions for stages 1 and 2, and so the trial continues to stage 3. At stage 3, the test statistic falls into the rejection region, and the null hypothesis is rejected.

A group sequential trial usually involves six steps:

1. You specify the statistical details of the design, including the null and alternative hypotheses, a test statistic for the hypothesis test, the Type I and II error probabilities, a stopping criterion, the total number of stages, and the relative information level at each stage.

2. You compute the boundary values for the trial based on the specifications in Step 1. You also compute the sample size required at each stage for the specified hypothesis test.

3. At each stage, you collect additional data with the required sample sizes. The data available at each stage include the data collected at previous stages in addition to the data collected at the current stage.

4. At each stage, you analyze the available data with a procedure such as the REG procedure, and you compute the test statistic.

5. At each stage, you compare the test statistic with the corresponding boundary values. You stop the trial to reject or accept the hypothesis, or you continue the trial to the next stage. If you continue the trial to the final stage, you either accept or reject the hypothesis.

6. After the trial stops, you compute parameter estimates, confidence limits for the parameter, and a p-value for the hypothesis test.

You use the SEQDESIGN procedure at Step 2 to compute the initial boundary values and required sample sizes for the trial. You use the companion SEQTEST procedure at Step 5 to compare the test statistic with its boundary values. At stage 1, the boundary values are derived by using the boundary information tables created by the SEQDESIGN procedure. These boundary information tables are structured for input to the SEQTEST procedure. At each subsequent stage, the boundary values are derived by using the test information tables are created by the SEQTEST procedure at the previous stage. These test information tables are also structured for input to the SEQTEST procedure. You also use the SEQTEST procedure at Step 6 to compute parameter estimates, confidence limits, and p-values after the trial stops.

The flowchart in Figure 110.2 summarizes the steps in a typical group sequential trial and the relevant SAS procedures.

Figure 110.2 Group Sequential Trial

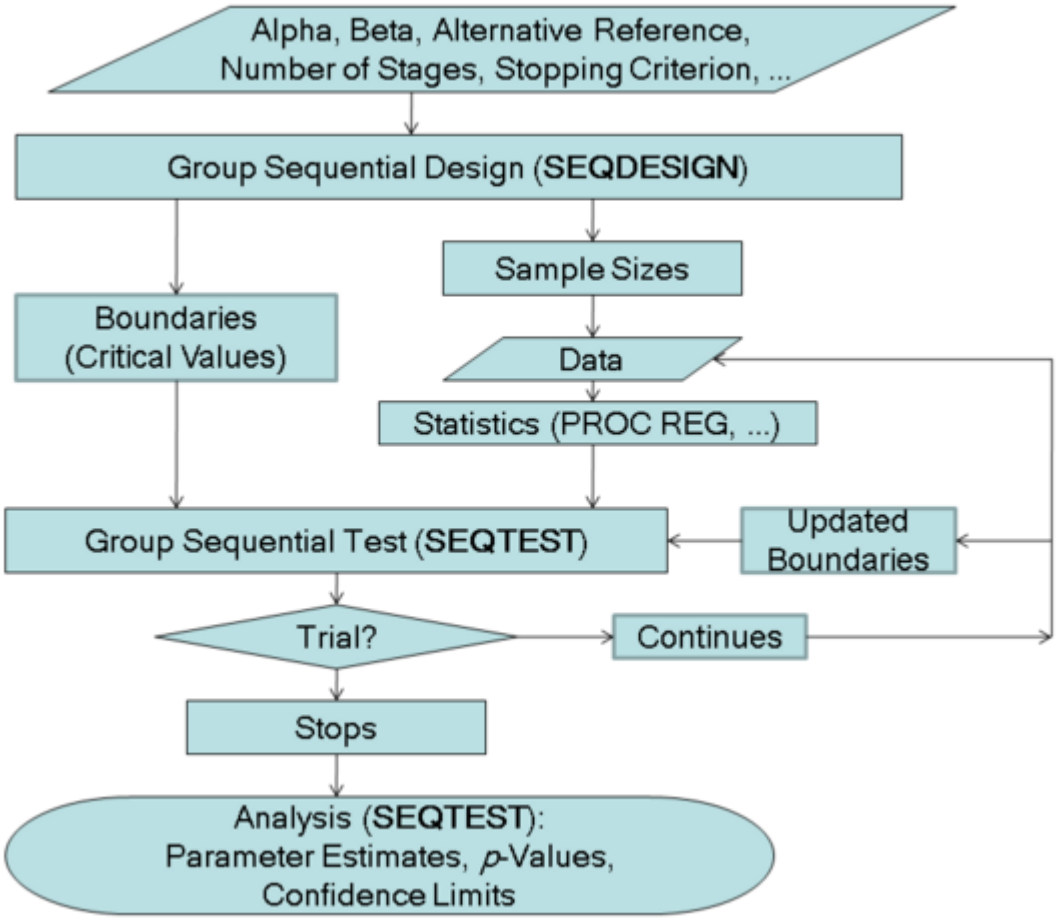

Features of the SEQDESIGN Procedure

The SEQDESIGN procedure assumes that the standardized Z test statistics for the null hypothesis $H_0 : \theta = 0$ at the stages have the joint canonical distribution with the information levels at the stages for the parameter θ. This implies that these test statistics are normally distributed. If the test statistic is not normally distributed, then it is assumed that the test statistic is computed from a large sample such that the statistic has an approximately normal distribution. See the section "Statistical Assumptions for Group Sequential Designs" on page 9159 for a detailed description of the joint canonical distribution.

You can use the SEQDESIGN procedure to compute required sample sizes for commonly used hypothesis tests. Note that for a fixed-sample design, you should use the POWER and GLMPOWER procedures to compute sample sizes.

The applicable tests include tests for binomial proportions and the log-rank test for two survival distributions. See the section "Applicable One-Sample Tests and Sample Size Computation" on page 9193, the section "Applicable Two-Sample Tests and Sample Size Computation" on page 9195, and the section "Applicable Regression Parameter Tests and Sample Size Computation" on page 9203 for examples of applicable tests in group sequential trials.

At each stage, the data are analyzed with a statistical procedure such as the REG procedure, and a test statistic and its associated information level are computed. The information level is the amount of information available about the unknown parameter. For a maximum likelihood statistic, the information level is the inverse of its variance.

At each stage, you use the SEQTEST procedure to derive the boundary values that correspond to the information level associated with the test statistic. You then use the SEQTEST procedure to compare the test statistic with these boundary values. When a trial is stopped at an interim stage or at the final stage, the SEQTEST procedure also derives parameter estimates, confidence limits for the parameter, and a *p*-value for hypothesis testing.

Output from the SEQDESIGN Procedure

In addition to computing the boundary values for a group sequential design, the SEQDESIGN procedure computes the following quantities:

- maximum sample size (as a percentage of the corresponding fixed-sample size) if the trial does not stop at an interim stage
- average sample numbers (as percentages of the corresponding fixed-sample sizes for nonsurvival data or fixed-sample numbers of events for survival data) under various hypothetical references, including the null and alternative references
- stopping probabilities at each stage under various hypothetical references to indicate how likely it is that the trial will stop at that stage

If you specify a SAMPLESIZE statement, the SEQDESIGN procedure also computes the following quantities:

- sample sizes that are required at each stage for the specified hypothesis test for nonsurvival data, in both fractional and integer numbers
- the number of events, time, and sample size that are required at each stage for the specified hypothesis test for survival data, in both fractional and integer-valued times or in both fractional and integer-valued sample sizes

Furthermore, the CEILADJDESIGN=INCLUDE option in the SAMPLESIZE statement creates an additional group sequential design that corresponds to the integer-valued sample sizes at the stages for the nonsurvival data, and to the integer-valued times or sample sizes at the stages for the survival data. When CEILADJDESIGN=INCLUDE, the SEQDESIGN procedure also displays the ceiling-adjusted design information in the “Design Information” table and the ceiling-adjusted design boundary information in the new “Ceiling-Adjusted Design Boundary Information” table.

You can create more than one design by specifying multiple DESIGN statements, and then you can choose the design that has the most desirable features. The next two subsections introduce some basic aspects of group sequential designs that are useful for getting started with the SEQDESIGN procedure.

Boundaries for Group Sequential Designs

A one-sided test is a test of a hypothesis with either a lower alternative ($H_1 : \theta < 0$) or an upper alternative ($H_1 : \theta > 0$), and a two-sided test is a test with a two-sided alternative ($H_1 : \theta \neq 0$). The number of critical values for a test depends on whether the alternative is one-sided or two-sided, and it also depends on whether the trial is conducted with a fixed-sample design or a group sequential design.

For a fixed-sample trial, a one-sided test has one critical value and a two-sided test has two critical values. These critical values are computed with the specified Type I error probability α. In contrast, at each interim stage of a group sequential trial, a one-sided test has up to two critical values and a two-sided group sequential test has up to four critical values. Thus, there are two or four possible boundaries for a group sequential design, and each boundary is a set of critical values, one from each stage.

Figure 110.3 illustrates the boundaries for a one-sided test with an upper alternative that allows for early stopping to either reject or accept the null hypothesis.

Figure 110.3 Boundary Plot for One-Sided Design

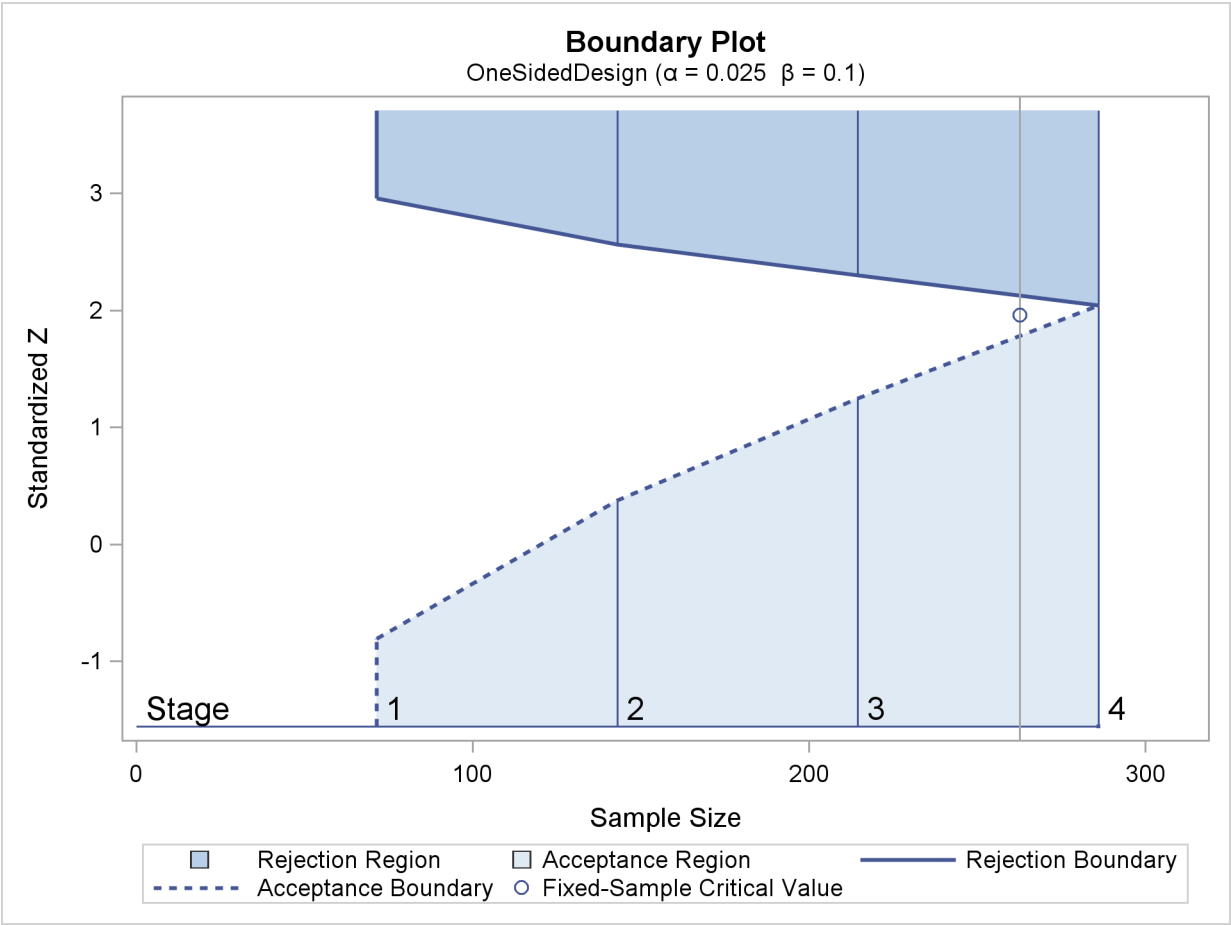

With an upper alternative, as in this example, the design has the following two boundaries: an upper α (rejection) boundary for the rejection region that consists of upper rejection critical values and an upper β (acceptance) boundary for the acceptance region that consists of upper acceptance critical values. The stages are indicated by vertical lines with accompanying stage numbers. In Figure 110.3, the horizontal axis

indicates the cumulative sample size for the group sequential trial. The vertical axis indicates the critical values at each stage on the standardized Z scale. Other scales can be used for the vertical axis, including the MLE scale, score statistic scale, and p-value scale.

At each interim stage, if the test statistic is in the rejection region (darker shaded area in Figure 110.3), the trial stops and the null hypothesis is rejected. If the test statistic is in the acceptance region (lightly shaded area in Figure 110.3), the trial stops and the hypothesis is accepted. Otherwise, the trial continues to the next stage. If the trial proceeds to the final stage (stage 4), the upper α and upper β critical values are identical, and the trial stops to either reject or accept the null hypothesis.

If you do not want to stop a trial when a test statistic falls in the acceptance region at an interim stage, you can derive a design with a nonbinding acceptance boundary to allow the trial to continue while maintaining the Type I error level. You need to increase the sample size for a design with this nonbinding feature. See the section "Acceptance ($\boldsymbol{\beta}$) Boundary" on page 9179 for a description of the nonbinding acceptance boundary.

Group Sequential Methods

For a group sequential design, there are two possible boundaries for a one-sided test and four possible boundaries for a two-sided test. Each boundary consists of one boundary value (critical value) for each stage. The SEQDESIGN procedure provides various methods for computing the boundary values.

The boundary value for a fixed-sample test cannot be applied to each stage of sequential design because, as shown by Armitage, McPherson, and Rowe (1969), repeated significance tests at a fixed level on accumulating data increase the probability of a Type I error. For example, with a fixed-sample two-sided test, the critical values ± 1.96 for a standardized Z statistic produce a Type I error probability level $\alpha = 0.05$. But for a two-sided group sequential test with two equally spaced stages, if the same critical values ± 1.96 are used to reject the null hypothesis at these two stages, the Type I error probability level is $\alpha = 0.083$, larger than the fixed-sample α.

Numerous methods are available for deriving the critical values for each boundary in a sequential design. Pocock (1977) applies repeated significance tests to group sequential trials with equal-size groups and derives a constant critical value on the standardized normal Z scale across all stages that maintains the specified Type I error probability level. O'Brien and Fleming (1979) propose a sequential procedure that has boundary values (in absolute value) decrease over the stages on the standardized normal Z scale.

The SEQDESIGN procedure provides the following three types of methods:

- fixed boundary shape methods, which derive boundaries with specified boundary shapes
- Whitehead methods, which adjust boundaries derived for continuous monitoring so that they apply to discrete monitoring
- error spending methods

Each type of methods uses a distinct approach to derive the boundary values for a group sequential trial. Whitehead methods require much less computation with resulting Type I error probability and power that are close but differ slightly from the specified values due to the approximations used in deriving the tests (Jennison and Turnbull 2000, p. 106). Fixed boundary shape methods derive boundary values by estimating a fixed number of parameters and require more computation. Error spending methods derive boundary values at

each stage sequentially and require much more computation than other types of methods for group sequential trials with a large number of stages.

Within each type of methods, you can choose methods that creating boundary values range from conservative stopping boundary values at early stages to liberal stopping boundary values at very early stages.

You can use the SEQDESIGN procedure to specify methods from the same type for each design. A different method can be specified for each boundary separately, but all methods in a design must be of the same type.

Fixed Boundary Shape Methods

The fixed boundary shape methods include the unified family methods and the Haybittle-Peto method. The unified family methods (Kittelson and Emerson 1999) derive boundaries from specified boundary shapes. These methods include Pocock's method (Pocock 1977) and the O'Brien-Fleming method (O'Brien and Fleming 1979) as special cases.

The Haybittle-Peto method (Haybittle 1971; Peto et al. 1976) uses a value of 3 for the critical values in interim stages, so the critical value at the final stage is close to the critical value for the fixed-sample design. In the SEQDESIGN procedure, the Haybittle-Peto method has been generalized to allow for different boundary values at interim stages.

Whitehead Methods

Whitehead and Stratton (1983) and Whitehead (1997, 2001) develop triangular and straight-line boundaries by adapting tests constructed for continuous monitoring to discrete monitoring of group sequential tests. With continuous monitoring, the values for each boundary fall in a straight line when plotted on the score statistic scale. The discrete boundary is derived by subtracting the expected overshoot from the continuous boundary to obtain the desired Type I and Type II error probabilities. For a design with early stopping to either reject or accept the null hypothesis, the boundaries form a triangle when plotted on the score statistic scale. See the section "Score Statistic" on page 9149 for a detailed description of the score statistic.

Error Spending Methods

For every sequential design, the α and β errors at each stage can be computed from the boundary values. On the other hand, you can derive the boundary values from specified α and β errors for each stage. The error spending function approach (Lan and DeMets 1983) uses an error spending function to specify the errors at each stage for each boundary and then derives the boundary values.

Getting Started: SEQDESIGN Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/seqdgs.sas.)

This section illustrates a clinical study design that uses a two-sided O'Brien-Fleming design (O'Brien and Fleming 1979) to stop the trial early for ethical concerns about possible harm or for unexpectedly strong efficacy of the new drug.

Suppose that a pharmaceutical company is conducting a clinical trial to test the efficacy of a new cholesterol-lowering drug. The primary focus is low-density lipoprotein (LDL), the so-called bad cholesterol, which is a risk factor for coronary heart disease. LDL is measured in mg/dl, milligrams per deciliter of blood.

The trial consists of two groups of equally allocated patients with elevated LDL levels: an experimental group given the new drug and a placebo control group. Suppose the changes in LDL level after the treatment for individuals in the experimental and control groups are normally distributed with means μ_e and μ_c, respectively, and have a common variance σ^2. Then the null hypothesis of no effect for the new drug is $H_0 : \theta = 0$, where $\theta = \mu_e - \mu_c$.

For a fixed-sample design with a total sample size N, the MLE for θ is computed as $\hat{\theta} = \hat{\mu}_e - \hat{\mu}_c$, where $\hat{\mu}_e$ and $\hat{\mu}_c$ are the sample means of the decreases in LDL level in the experimental and control groups, respectively.

Following the derivation in the section "Test for the Difference between Two Normal Means" on page 9196, the statistic $\hat{\theta}$ has a normal distribution

$$\hat{\theta} \sim N\left(\theta, \frac{4\sigma^2}{N}\right)$$

Thus, under the null hypothesis $H_0 : \theta = 0$, the standardized statistic

$$Z = \frac{\hat{\theta}}{\sqrt{\frac{4\sigma^2}{N}}} \sim N\,(0,\,1)$$

The Z statistic can be used to test the null hypothesis H_0. If the variance σ^2 is unknown, the sample variance can be used to compute the test statistic if it is assumed that the sample variance is computed from a large sample such that the Z statistic has an approximately standard normal distribution.

With a Type I error probability $\alpha = 0.05$, the critical values for the Z statistic are given by $\Phi^{-1}(\alpha/2) = -1.96$ and $\Phi^{-1}(1-\alpha/2) = 1.96$, where Φ is the cumulative standard normal distribution function. At the end of study, if $Z \geq 1.96$, the null hypothesis is rejected for harmful drug effect, and if $Z \leq -1.96$, the null hypothesis is rejected for efficacy of the new drug. Otherwise, the null hypothesis is not rejected and the drug effect is not significant.

Also suppose that for the trial, the alternative reference $\theta = -10$ is the clinically meaningful difference that the trial should detect with a high probability (power). Further suppose that a good estimate of the standard deviation for the changes in LDL level is $\hat{\sigma} = 20$. The following statements invoke the SEQDESIGN procedure and request a four-stage O'Brien-Fleming design for standardized normal test statistics:

```
ods graphics on;
proc seqdesign altref=-10
               plots=boundary(hscale=samplesize)
               ;
   TwoSidedOBrienFleming: design nstages=4
                          method=obf
                          ;
   samplesize model=twosamplemean(stddev=20);
ods output Boundary=Bnd_LDL;
run;
```

The ALTREF= option specifies the alternative reference, and the actual maximum information is derived in the SEQDESIGN procedure. With ODS Graphics enabled, the PLOTS=BOUNDARY option displays a boundary plot with the rejection and acceptance regions.

In the DESIGN statement, the label **TwoSidedOBrienFleming** identifies the design in the output tables. By default (or equivalently if you specify ALT=TWOSIDED and STOP=REJECT in the DESIGN statement), the design has a two-sided alternative hypothesis in which early stopping in the interim stages occurs to reject the null hypothesis. That is, at each interim stage, the trial either is stopped to reject the null hypothesis or continues to the next stage.

The NSTAGES=4 option in the DESIGN statement specifies the total number of stages in the group sequential trial, including three interim stages and a final stage. In the SEQDESIGN procedure, the null hypothesis for the design is $H_0 : \theta = 0$. By default (or equivalently if you specify ALPHA=0.05 and BETA=0.10 in the DESIGN statement), the design has a Type I error probability $\alpha = 0.05$, and a Type II error probability $\beta = 0.10$; the latter corresponds to a power of $1 - \beta = 0.90$ at the alternative reference $H_1 : \theta = -10$.

For a two-sided design with early stopping to reject the null hypothesis, there are two boundaries for the design: an upper α boundary that consists of upper rejection critical values and a lower α boundary that consists of lower rejection critical values. Each boundary is a set of critical values, one from each stage. With the METHOD=OBF option in the DESIGN statement, the O'Brien-Fleming method is used for the two boundaries for the design; see Figure 110.7.

A property of the boundaries constructed with the O'Brien-Fleming design is that the null hypothesis is more difficult to reject in the early stages than in the later stages. That is, the trial is rejected in the early stages only with overwhelming evidence, because in these stages there might not be a sufficient number of responses for a reliable estimate of the treatment effect.

The SAMPLESIZE statement with the MODEL=TWOSAMPLEMEAN option uses the derived maximum information to compute required sample sizes for a two-sample test for mean difference. The ODS OUTPUT statement with the BOUNDARY=BND_LDL option creates an output data set named BND_LDL which contains the resulting boundary information.

In a clinical trial, the amount of information about an unknown parameter available from the data can be measured by the Fisher information. For a maximum likelihood statistic, the information level is the inverse of its variance. See the section "Maximum Likelihood Estimator" on page 9148 for a detailed description of Fisher information. At each stage of the trial, data are collected and analyzed with a statistical procedure, and a test statistic and its corresponding information level are computed.

In this example, you can use the REG procedure to compute the maximum likelihood estimate $\hat{\theta}$ for the drug effect and the corresponding standard error for $\hat{\theta}$. At stage 1, you can use the SEQTEST procedure to compare the test statistic with adjusted boundaries derived from the boundary information stored in the

BOUND_LDL data set. At each subsequent stage, you can use the SEQTEST procedure to compare the test statistic with adjusted boundaries derived from the boundary information stored in the test information table created by the SEQTEST procedure at the previous stage. The test information tables are structured for input to the SEQTEST procedure.

At each interim stage, the trial will either be stopped to reject the null hypothesis or continue to the next stage. At the final stage, the null hypothesis is either rejected or accepted.

By default (or equivalently if you specify INFO=EQUAL in the DESIGN statement), the SEQDESIGN procedure derives boundary values with equally spaced information levels for all stages—that is, the same information increment between successive stages. The "Design Information," "Method Information," and "Boundary Information" tables are displayed by default, as shown in Figure 110.4, Figure 110.5, and Figure 110.6, respectively.

The "Design Information" table in Figure 110.4 displays design specifications and four derived statistics: the actual maximum information, the maximum information, the average sample number under the null hypothesis (Null Ref ASN), and the average sample number under the alternative hypothesis (Alt Ref ASN). Except for the actual maximum information, each statistic is expressed as a percentage of the identical statistic for the corresponding fixed-sample information. The average sample number is the expected sample size (for nonsurvival data) or expected number of events (for survival data). Note that for a symmetric two-sided design, the ALTREF=–10 option implies a lower alternative reference of –10 and an upper alternative reference of 10.

Figure 110.4 O'Brien-Fleming Design Information

The SEQDESIGN Procedure
Design: TwoSidedOBrienFleming

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	O'Brien-Fleming
Boundary Key	Both
Alternative Reference	-10
Number of Stages	4
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	102.2163
Max Information	0.107403
Null Ref ASN (Percent of Fixed Sample)	101.5728
Alt Ref ASN (Percent of Fixed Sample)	76.7397

The maximum information is the information level at the final stage of the group sequential trial. The Max Information (Percent Fixed-Sample) is the maximum information for the sequential design expressed as a percentage of the information for the corresponding fixed-sample design. In Figure 110.4, the Max Information (Percent Fixed-Sample) is 102.22%, which means that the information needed for the group sequential trial is 2.22% more than that of the corresponding fixed-sample design if the trial does not stop at any interim stage.

The Null Ref ASN (Percent Fixed-Sample) is the average sample number (expected sample size) required under the null hypothesis for the group sequential design expressed as a percentage of the sample size for the corresponding fixed-sample design. In Figure 110.4, the Null Ref ASN is 101.57%, which means that the expected sample size for the group sequential trial is 1.57% greater than the corresponding fixed-sample size.

Similarly, the Alt Ref ASN (Percent Fixed-Sample) is the average sample number (expected sample size) required under the alternative hypothesis for the group sequential design expressed as a percentage of the sample size for the corresponding fixed-sample design. In Figure 110.4, the Alt Ref ASN is 76.74%, which means that the expected sample size for the group sequential trial is 76.74% of the corresponding fixed-sample size. That is, if the alternative hypothesis is true, then on average, only 76.74% of the fixed-sample size is needed for the group sequential trial.

In this example, the O'Brien-Fleming design requires only a slight increase in sample size if the trial proceeds to the final stage. On the other hand, if the alternative hypothesis is correct, this design provides a substantial saving in sample size on average.

The "Method Information" table in Figure 110.5 displays the computed Type I and Type II error probabilities α and β, and the derived drift parameter for the design. For a two-sided test with early stopping to reject the null hypothesis, both lower and upper α boundaries are created. With the specified ALTREF= option, the alternative references are also included.

With the zero null reference, the drift parameter is the standardized alternative reference at the final stage $\theta_1\sqrt{I_X}$, where θ_1 is the alternative reference and I_X is the maximum information. See the section "Specified and Derived Parameters" on page 9183 for a detailed description of the drift parameter. The drift parameters for the design are derived in the SEQDESIGN procedure even if the alternative reference is not specified or derived in the procedure.

Figure 110.5 Method Information

Method Information

				Unified Family				
Boundary	**Method**	**Alpha**	**Beta**	**Rho**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	O'Brien-Fleming	0.02500	0.10000	0.5	0	2.02429	10	3.277238
Lower Alpha	O'Brien-Fleming	0.02500	0.10000	0.5	0	2.02429	-10	-3.27724

The O'Brien-Fleming method belongs to the unified family of designs, which is parameterized by two parameters, ρ and τ, as implemented in the SEQDESIGN procedure. See Table 110.3 for parameter values of commonly used methods in the unified family. The "Method Information" table in Figure 110.5 displays the values of $\rho = 0.5$ and $\tau = 0$, which are the parameters for the O'Brien-Fleming method. The table also displays the derived parameter $C_\alpha = 2.0243$, which is used in the construction of symmetric lower and upper α boundaries; see the section "Unified Family Methods" on page 9168.

The "Boundary Information" table in Figure 110.6 displays the information level, including the proportion, actual level, and corresponding sample size (N) at each stage. The table also displays the lower and upper alternative references, and the lower and upper boundary values at each stage.

Figure 110.6 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0							
	Information Level			Alternative Reference		Boundary Values	
						Lower	Upper
Stage	Proportion	Actual	N	Lower	Upper	Alpha	Alpha
1	0.2500	0.026851	42.96116	-1.63862	1.63862	-4.04859	4.04859
2	0.5000	0.053701	85.92233	-2.31736	2.31736	-2.86278	2.86278
3	0.7500	0.080552	128.8835	-2.83817	2.83817	-2.33745	2.33745
4	1.0000	0.107403	171.8447	-3.27724	3.27724	-2.02429	2.02429

The information proportion is the proportion of maximum information available at each stage and N is the corresponding sample size. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the procedure displays boundary values with the standardized Z scale in the boundary information table and the boundary plot. The alternative reference on the standardized Z scale at stage k is given by $\theta_1\sqrt{I_k}$, where θ_1 is the alternative reference and I_k is the information available at stage k, $k = 1, 2, 3, 4$. These standardized alternative references for the design are derived in the SEQDESIGN procedure even if the alternative reference is not specified or derived in the procedure.

In this example, a standardized Z statistic is computed by standardizing the parameter estimate of the effect in LDL level. A lower Z test statistic indicates a beneficial effect. Consequently, at each interim stage, if the standardized Z test statistic is less than or equal to the corresponding lower α boundary value, the hypothesis $H_0 : \theta = 0$ is rejected for efficacy. If the test statistic is greater than or equal to the corresponding upper α boundary value, the hypothesis H_0 is rejected for harmful effect. Otherwise, the process continues to the next stage. At the final stage (stage 4), the hypothesis H_0 is rejected for efficacy if the Z statistic is less than or equal to the corresponding lower α boundary value –2.0243, and the hypothesis H_0 is rejected for harmful effect if the Z statistic is greater than or equal to the corresponding upper α boundary value 2.0243. Otherwise, the hypothesis of no significant difference is accepted.

Note that in a typical trial, the actual information levels do not match the information levels specified in the design. The SEQTEST procedure modifies the boundary values stored in the BOUND_LDL data set to adjust for these new information levels.

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Figure 110.7. This plot displays the boundary values in the "Boundary Information" table in Figure 110.6. The stages are indicated by vertical lines with accompanying stage numbers. The horizontal axis indicates the sample sizes for the stages. Note that comparing with a fixed-sample design, only a small increase in sample size is needed for the O'Brien-Fleming design, as shown in Figure 110.7.

If a test statistic at an interim stage is in the rejection region (shaded area), the trial stops and the null hypothesis is rejected. If the statistic is not in any rejection region, the trial continues to the next stage.

Figure 110.7 Boundary Plot

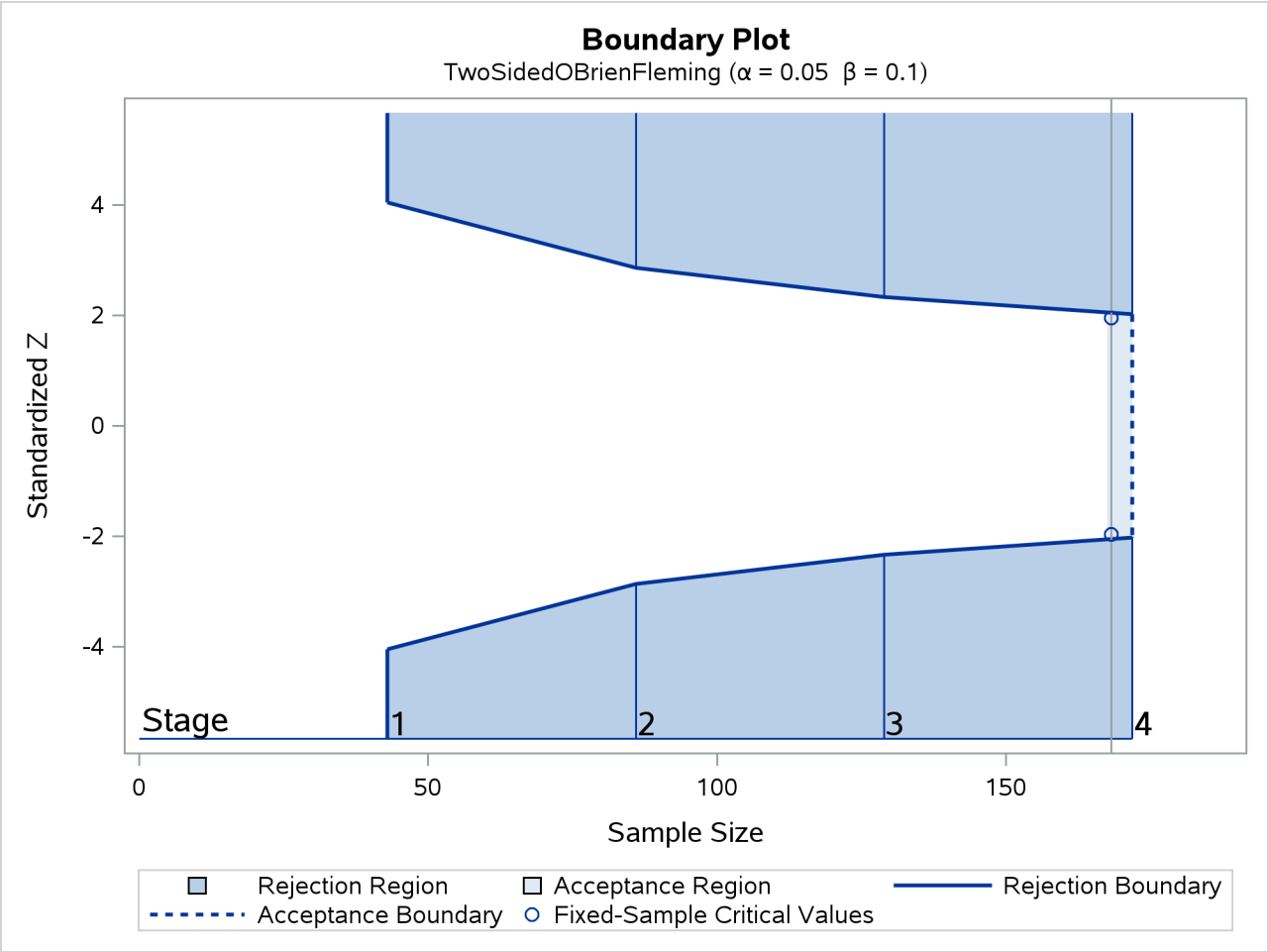

The boundary plot also displays critical values for the corresponding fixed-sample design. The symbol "∘" identifies the fixed-sample critical values of –1.96 and 1.96, and the accompanying vertical line indicates the required sample size for the fixed-sample design at the horizontal axis. Note that the boundary values ±2.0243 at the final stage are close to the fixed-sample critical values ±1.96.

When you specify the SAMPLESIZE statement, the maximum information (either explicitly specified or derived in the SEQDESIGN procedure) is used to compute the required sample sizes for the study. The MODEL=TWOSAMPLEMEAN(STDDEV=20) option specifies the test for the difference between two normal means. See the section "Test for the Difference between Two Normal Means" on page 9196 for a detailed derivation of these required sample sizes.

The "Sample Size Summary" table in Figure 110.8 displays the parameters for the sample size computation and the resulting maximum and expected sample sizes.

Figure 110.8 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Means
Mean Difference	-10
Standard Deviation	20
Max Sample Size	171.8447
Expected Sample Size (Null Ref)	170.7627
Expected Sample Size (Alt Ref)	129.0137

The "Sample Sizes (N)" table in Figure 110.9 displays the required sample sizes at each stage for the trial, in both fractional and integer numbers. The derived fractional sample sizes are displayed under the heading "Fractional N." These sample sizes are rounded up to integers under the heading "Ceiling N." By default (or equivalently if you specify WEIGHT=1 in the MODEL=TWOSAMPLEMEAN option), the sample sizes for the two groups are equal for the two-sample test.

Figure 110.9 Derived Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Mean Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	42.96	21.48	21.48	0.0269	44	22	22	0.0275
2	85.92	42.96	42.96	0.0537	86	43	43	0.0538
3	128.88	64.44	64.44	0.0806	130	65	65	0.0812
4	171.84	85.92	85.92	0.1074	172	86	86	0.1075

In practice, integer sample sizes are used in the trial, and the resulting information levels increase slightly. Thus, 22, 43, 65, and 86 individuals are needed in each of the two groups for the four stages, respectively.

You can also create an adjusted design that corresponds to these integer-valued sample sizes at the stages by specifying the CEILADJDESIGN=INCLUDE option in the SAMPLESIZE statement.

Syntax: SEQDESIGN Procedure

The following statements are available in the SEQDESIGN procedure:

PROC SEQDESIGN < *options* > **;**
< *label*: > **DESIGN** *options* **;**
SAMPLESIZE < **MODEL=** *option* > **;**

The PROC SEQDESIGN statement and the DESIGN statement are required for the SEQDESIGN procedure. Each DESIGN statement requests a new group sequential design, and multiple DESIGN statements can be used to create more than one design for comparison of features. The label, which must be a valid SAS name, is used to identify the design in the output tables and graphics. The SAMPLESIZE statement computes the required sample sizes for the design specified in each DESIGN statement. With a selected design, the SAMPLESIZE statement computes the required sample sizes for the trial.

PROC SEQDESIGN Statement

PROC SEQDESIGN < *options* > **;**

The PROC SEQDESIGN statement invokes the SEQDESIGN procedure. Table 110.1 summarizes the *options* available in the PROC SEQDESIGN statement.

Table 110.1 Summary of PROC SEQDESIGN Options

Option	Description
Design Parameters	
ALTREF=	Specifies the alternative reference
BOUNDARYSCALE=	Specifies the statistic scale for the boundary
MAXINFO=	Specifies the maximum information level
Table Output	
ERRSPEND	Displays the cumulative error spending at each stage
PSS	Displays powers and expected sample sizes
STOPPROB	Displays expected cumulative stopping probabilities
Graphics Output	
PLOTS=ASN	Displays the expected sample numbers plot
PLOTS=BOUNDARY	Displays the detailed boundary plot
PLOTS=COMBINEDBOUNDARY	Displays the combined boundary plot
PLOTS=ERRSPEND	Displays the error spending plot
PLOTS=POWER	Displays the powers plot

By default, the SEQDESIGN procedure displays tables of design information, method information, and boundary information for each specified design. If ODS Graphics is enabled, it also displays a detailed boundary plot.

In addition, you can use output options to display output tables such as expected cumulative stopping

probability at each stage under various hypothetical references. If ODS Graphics is enabled, you can also use output options to display plots such as powers and expected sample sizes under various hypothetical references.

The following *options* can be used in the PROC SEQDESIGN statement to derive boundary values for all sequential designs in the procedure. They are listed in alphabetical order.

ALTREF=θ_1 **< (< LOWER=**θ_{1l} **> < UPPER=**θ_{1u} **>) >**

specifies the alternative reference—that is, the hypothetical reference under the alternative hypothesis at which the power is computed. The LOWER= and UPPER= options are applicable only for a two-sided design with different lower and upper alternative references.

For a one-sided design, $\theta_{1l} = -|\theta_1|$ is the lower alternative reference and $\theta_{1u} = |\theta_1|$ is the upper alternative reference. For a two-sided design, the specified θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. If the LOWER= option is not specified, $\theta_{1l} = -|\theta_1|$, and if the UPPER= option is not specified, $\theta_{1u} = |\theta_1|$.

The specification of the ALTREF= option depends on the hypothesis used in the clinical trial. For example, suppose the null hypothesis $H_0 : \theta = 0$ with an alternative hypothesis $H_1 : \theta = \theta_1$ is used to compare two binomial populations, $p_a = p_b$. Then θ_1 is the proportion difference under H_1 if $\theta = p_a - p_b$, and θ_1 is the log odds ratio under H_1 if $\theta = \log\left(\frac{p_a(1-p_b)}{p_b(1-p_a)}\right)$.

If the ALTREF= option is not specified, the alternative reference θ_1 can also be specified or derived in the SAMPLESIZE statement. If θ_1 is specified or derived in the SAMPLESIZE statement, $\theta_{1l} = -|\theta_1|$ and $\theta_{1u} = |\theta_1|$ are the lower and upper alternative references, respectively.

Note that if the SAMPLESIZE statement is specified with a two-sided design, the sample sizes derived by using the lower and upper alternatives might be different. If θ_1 is specified or derived in the SAMPLESIZE statement, it is used to compute the sample sizes. Otherwise, the θ_1 specified in the ALTREF= option is used.

BOUNDARYSCALE=MLE | SCORE | STDZ | PVALUE

BSCALE=MLE | SCORE | STDZ | PVALUE

specifies the scale for the statistic that is displayed in the boundary table and boundary plots. The keywords MLE, SCORE, STDZ, and PVALUE correspond to the boundary with the maximum likelihood estimate scale, the score statistic scale, the standardized normal Z scale, and the p-value scale, respectively. The default is BOUNDARYSCALE=STDZ.

With the BOUNDARYSCALE=MLE or BOUNDARYSCALE=SCORE option, the maximum information must be either explicitly specified with the MAXINFO= option or derived in the SEQDESIGN procedure to provide the necessary information level at each stage to compute the boundary values. See the section "Boundary Scales" on page 9161 for a detailed description of the statistic scale for the boundary values.

Note that for a two-sided design, the p-value scale displays the one-sided fixed-sample p-value under the null hypothesis with a lower alternative hypothesis.

MAXINFO=*number*

specifies the maximum information level for the design. If the MAXINFO=option is specified and the alternative reference is either specified explicitly with the ALTREF= option or derived from the SAMPLESIZE statement, then the Type I and Type II error probability levels cannot be met simultaneously. In this case, the ALPHA= option in the DESIGN statement is applicable only with the BOUNDARYKEY=ALPHA option (which is the default) in the DESIGN statement, and the Type II error probability β is derived. The BETA= option in the DESIGN statement is applicable only with the BOUNDARYKEY=BETA option in the DESIGN statement, and the Type I error probability α is derived.

Table Output Options

The following *options* can be used in the PROC SEQDESIGN statement to display addition table output. They are listed in alphabetical order.

ERRSPEND

displays the error spending at each stage for each boundary in the design.

PSS < (CREF= $c_1 < c_2 \ldots >$ **) >**

displays powers and expected sample sizes under various hypothetical references, where the numbers $c_i \geq 0$.

For a one-sided design, the power and expected sample sizes under hypotheses $\theta = c_i\, \theta_1$ are displayed, where θ_1 is the alternative reference and c_i are the values specified in the CREF= option.

For a two-sided design, the power and expected sample sizes under hypotheses $\theta = c_i \theta_{1l}$ and $\theta = c_i \theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. The default is CREF= 0 0.5 1.0 1.5.

Note that for a symmetric two-sided design, only the power and expected sample sizes under hypotheses $\theta = c_i\, \theta_{1u}$ are derived. See the section "Type I and Type II Errors" on page 9166 for a detailed description of the power computation. See the section "Powers and Expected Sample Sizes" on page 9213 for a detailed description of the expected sample size computation.

STOPPROB < (CREF= $c_1 < c_2 \ldots >$ **) >**

displays expected cumulative stopping probabilities under various hypothetical references, where the numbers $c_i \geq 0$.

For a one-sided design, expected cumulative stopping probabilities at each stage under hypotheses $\theta = c_i\, \theta_1$ are displayed, where θ_1 is the alternative reference and c_i are the values specified in the CREF= option.

For a two-sided design, expected cumulative stopping probabilities at each stage under hypotheses $\theta = c_i \theta_{1l}$ and $\theta = c_i \theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. Note that for a symmetric two-sided design, only expected cumulative stopping probabilities under hypotheses $\theta = c_i\, \theta_{1u}$ are derived. The default is CREF= 0 0.5 1.0 1.5.

Graphics Output Options

This section describes the *options* for using ODS Graphics with the SEQDESIGN procedure to create plots.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc seqdesign altref=-10 plots=boundary(hscale=samplesize);
   TwoSidedOBrienFleming: design nstages=4 method=obf;
   samplesize model=twosamplemean(stddev=20);
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The following *options* can be used in the PROC SEQDESIGN statement to display graphs with ODS Graphics. They are listed in alphabetical order.

PLOTS < (ONLY) > < = *plot-request* >

PLOTS < (ONLY) > < = (*plot-request* < . . . *plot-request* >) >

specifies options that control the details of the plots. The default is PLOTS=BOUNDARY. The global plot option ONLY suppresses the default plots and displays only plots specifically requested.

The plot request options are as follows.

ALL

produces all appropriate plots.

ASN < (CREF= $c_1 < c_2 \ldots >$ **) >**

displays a plot of the average sample numbers (expected sample sizes for nonsurvival data or expected numbers of events for survival data) under various hypothetical references, where the numbers $c_i \geq 0$. These average sample numbers are displayed as percentages of the average sample numbers for the corresponding fixed-sample design.

For a one-sided design, expected sample numbers under hypotheses $\theta = c_i\,\theta_1$ are displayed, where θ_1 is the alternative reference and c_i are the values specified in the CREF= option.

For a two-sided design, expected sample numbers under hypotheses $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. Note that for a symmetric two-sided design, only the average sample numbers under hypotheses $\theta = c_i\,\theta_{1u}$ are derived. The default is CREF= 0 to 1.5 by 0.01.

BOUNDARY < (HSCALE=INFO | SAMPLESIZE) >

displays a plot of the resulting sequential boundaries with the acceptance and rejection regions for each design. Either the information level (HSCALE=INFO) or the sample size (HSCALE=SAMPLESIZE) is displayed on the horizontal axis. If the maximum information is not available for the design, the information in percentage of its corresponding fixed-sample design are used in the plot. The stage number for each stage is displayed inside the plot. The default is HSCALE=INFO.

If the HSCALE=SAMPLESIZE option is specified. the SAMPLESIZE statement must also be specified. The options MODEL=INPUTNEVENTS, MODEL=TWOSAMPLESURVIVAL, and MODEL=PHREG in the SAMPLESIZE statement indicate survival data. For a sample that does not contain survival data, the sample size at each stage is displayed on the horizontal axis. For

survival data, the number of events is displayed on the horizontal axis at each stage. The critical values for the corresponding fixed-sample design are also displayed in the plot.

COMBINEDBOUNDARY < (HSCALE=INFO | SAMPLESIZE | STAGE) >
displays a plot of the resulting sequential boundaries for all designs simultaneously. You can display the information level (HSCALE=INFO), the sample size (HSCALE=SAMPLESIZE), or the stage number (HSCALE=STAGE) on the horizontal axis. The default is HSCALE=INFO. With HSCALE=INFO, if the maximum information is not available for the design, then the information in percentage of its corresponding fixed-sample design is used in the plot.

If the HSCALE=SAMPLESIZE option is specified, the SAMPLESIZE statement must also be specified. The options MODEL=INPUTNEVENTS, MODEL=TWOSAMPLESURVIVAL, and MODEL=PHREG in the SAMPLESIZE statement indicate survival data. For a sample that does not contain survival data, the sample size at each stage is displayed on the horizontal axis. For survival data, the number of events is displayed on the horizontal axis at each stage.

ERRSPEND < (HSCALE=INFO | STAGE) >
displays a plot of the error spending for all sequential boundaries in the designs simultaneously. You can display the information level (HSCALE=INFO) or the stage number (HSCALE=STAGE) on the horizontal axis. With HSCALE=INFO, the information fractions are used in the plot. The default is HSCALE=STAGE.

NONE
suppresses all plots.

POWER < (CREF= $c_1 < c_2 \ldots >$ **) >**
displays a plot of the power curves under various hypothetical references, where the numbers $c_i \geq 0$.

For a one-sided design, powers under hypotheses $\theta = c_i\,\theta_1$ are displayed, where θ_1 is the alternative reference and c_i are the values specified in the CREF= option.

For a two-sided design, powers under hypotheses $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. Note that for a symmetric two-sided design, only powers under hypotheses $\theta = c_i\,\theta_{1u}$ are derived. The default is CREF= 0 to 1.5 by 0.01.

DESIGN Statement

< *label*: > DESIGN < *options* > **;**

The DESIGN statement requests a new group sequential design. You can use multiple DESIGN statements, and each DESIGN statement corresponds to a separate group sequential design.

Table 110.2 summarizes the *options* available in the DESIGN statement.

Table 110.2 Design Statement Options

Option	Description
Design Parameters	
ALPHA=	Specifies the Type I error probability level α
ALT=	Specifies the type of alternative hypothesis
BETA=	Specifies the Type II error probability level β
BETAOVERLAP=	Checks for overlapping of the lower and upper β boundaries in a two-sided design with error spending methods
BOUNDARYKEY=	Specifies the type of error probability to maintain
INFO=	Specifies the information levels
NSTAGES=	Specifies the number of stages
STOP=	Specifies the condition for early stopping
Boundary Methods	
METHOD=	Specifies methods for boundary values

The required NSTAGES= option specifies the number of stages. The METHOD= option is required if the number of stages specified in the NSTAGES= option is greater than one. The following *options* can be used in the DESIGN statement. They are listed in alphabetical order.

ALPHA=α <(<LOWER=α_l> <UPPER=α_u>)>

specifies the Type I error probability α. The default is $\alpha = 0.05$. The LOWER= and UPPER= options are applicable only for the two-sided design. The LOWER= option specifies the lower Type I error probability α_l, and the upper Type I error probability is computed as $\alpha_u = \alpha - \alpha_l$. The UPPER= option specifies the upper Type I error probability α_u, and the lower Type I error probability is computed as $\alpha_l = \alpha - \alpha_u$. If both LOWER= and UPPER= options are not specified, $\alpha_l = \alpha_u = \alpha/2$.

If both the MAXINFO= and ALTREF= options are specified, then the Type I and Type II error probability levels cannot be met simultaneously. In this case, the ALPHA= option is applicable only with the BOUNDARYKEY=ALPHA option (which is the default), and the Type II error probability β is derived.

ALT=LOWER | UPPER | TWOSIDED

specifies the type of alternative hypothesis in the design. For a test of $H_0 : \theta = 0$, the keywords LOWER, UPPER, and TWOSIDED correspond to the alternatives of $\theta < 0$, $\theta > 0$, and $\theta \neq 0$, respectively. The default is ALT=TWOSIDED.

BETA=β <(<LOWER=β_l> <UPPER=β_u>)>

specifies the Type II error probability level β. The default is $\beta = 0.10$. The LOWER= and UPPER= options are applicable only for the two-sided design. The LOWER= option specifies the lower Type II error probability level β_l, and the UPPER= option specifies the upper Type II error probability level β_u. If the LOWER= or UPPER= option is not specified, β is used.

If both the MAXINFO= and ALTREF= options are specified, then the Type I and Type II error probability levels cannot be met simultaneously. In this case, the BETA= option is applicable only with the BOUNDARYKEY=BETA option, and the Type I error probability α is derived.

BETAOVERLAP=ADJUST | NOADJUST

OVERLAP=ADJUST | NOADJUST

specifies whether to check for overlapping of the lower and upper β boundaries for the two corresponding one-sided tests. This option applies to two-sided designs with STOP=ACCEPT or STOP=BOTH that are constructed with error spending methods, and this type of overlapping might result from a small β spending at an interim stage. When you specify BETAOVERLAP=ADJUST, the procedure checks for this type of overlapping. If such overlapping is found, the β boundaries for the two-sided design at that stage are set to missing, and the β spending values at subsequent stages are adjusted, as described in the section "Boundary Adjustments for Overlapping Lower and Upper $\boldsymbol{\beta}$ Boundaries" on page 9182".

You can specify BETAOVERLAP=NOADJUST to request that no adjustment be made. The default is BETAOVERLAP=ADJUST.

BOUNDARYKEY=ALPHA | BETA | BOTH | NONE

specifies types of errors to be maintained in the resulting boundary. The default is BOUNDARYKEY=ALPHA if both ALTREF= and MAXINFO= options are specified. Otherwise, the default is BOUNDARYKEY=NONE for Whitehead methods with the STOP=BOTH option, and it is BOUNDARYKEY=BOTH for others.

See the section "Applicable Boundary Keys" on page 9184 for a detailed description of applicable boundary keys.

INFO=EQUAL | CUM(*numbers* **)**

specifies relative information levels for all stages in the design. The INFO=EQUAL option specifies equally spaced information levels, and the INFO=CUM option specifies cumulative relative information levels. The default is INFO=EQUAL.

If the number of information levels specified in the INFO=CUM option is less than the number of stages specified in the NSTAGES= option, the last available information increment is used as the information increment for each subsequent stage.

METHOD=WHITEHEAD < (TAU=τ **< (< LOWER=**τ_l **> < UPPER=**τ_u **>) >) >**

METHOD=*method*

METHOD(*boundary***) =** *method*

specifies the methods for the boundaries in the design, where $0 \leq \tau < 0.5$.

For a one-sided design, an α boundary is created with the STOP=REJECT or STOP=BOTH option, and a β boundary is created with the STOP=ACCEPT or STOP=BOTH option. For a two-sided design, lower and upper α boundaries are created with the STOP=REJECT or STOP=BOTH option, and lower and upper β boundaries are created with the STOP=ACCEPT or STOP=BOTH option.

There are three types of methods available in the SEQDESIGN procedure. The unified family methods and Haybittle-Peto methods derive boundary values with fixed boundary shape; the Whitehead methods derive boundary values by adjusting the boundary values generated from continuous monitoring; and the error spending methods derive the boundary values from the specified errors used at each stage. You can specify different methods for the same design, but all methods must be from the same group.

For a design with early stopping to reject or accept the null hypothesis, the METHOD=WHITEHEAD option uses Whitehead's triangular design and double-triangular design for a one-sided design and two-sided design, respectively (Whitehead and Stratton 1983; Whitehead 1997, 2001). For a design

with early stopping only to reject the null hypothesis or only to accept the null hypothesis, you can specify the slope of the boundary line in the score statistic scale with the TAU= τ option. The default is TAU=0.25. See the section "Whitehead Methods" on page 9173 for a detailed description of the Whitehead methods.

The following *options* specify available error spending methods for the boundary. Each of these methods can be specified with the METHOD= option for all boundaries, or with the METHOD(*boundary*) = option for an individual boundary. See the section "Error Spending Methods" on page 9177 for a detailed description of these error spending methods.

ERRFUNCGAMMA < (GAMMA= γ **) >**
specifies a gamma cumulative error spending function for the boundary (Hwang, Shih, and DeCani 1990). The GAMMA= option specifies the gamma parameter γ in the function, where $\gamma \leq 3$. The boundaries created with $\gamma = 1$ are similar to the boundaries from the Pocock method, and the boundaries created with $\gamma = -4$ or $\gamma = -5$ are similar to the boundaries from the O'Brien-Fleming method. The default is GAMMA=–2, which is the average of $\gamma = 1$ and $\gamma = -5$.

ERRFUNCOBF
specifies the O'Brien-Fleming-type cumulative error spending function for the boundary (Lan and DeMets 1983).

ERRFUNCPOC
specifies the Pocock-type cumulative error spending function for the boundary (Lan and DeMets 1983).

ERRFUNCPOW < (RHO=ρ **) >**
specifies a power cumulative error spending function for the boundary (Jennison and Turnbull 2000, p. 148). The RHO= option specifies the power parameter ρ in the function, where $\rho \geq 0.25$. The boundaries created with $\rho = 1$ are similar to the boundaries from the Pocock method, and the boundaries created with $\rho = 3$ are similar to the boundaries from the O'Brien-Fleming method. The default is RHO=2, which is the average of $\rho = 1$ and $\rho = 3$.

ERRSPEND (*numbers* **)**
specifies the relative cumulative error spending at each stage.

With a fixed boundary shape, you can use the following available Haybittle-Peto methods and unified family methods to derive the boundary. You can specify each of these methods in the METHOD= option for all boundaries, or in the METHOD(*boundary*) = option for an individual boundary. See the section "Haybittle-Peto Method" on page 9173 for a detailed description of the Haybittle-Peto methods, and see the section "Unified Family Methods" on page 9168 for a detailed description of unified family methods.

HP | HAYBITTLE | PETO < (Z= *numbers* **| PVALUE=** *numbers* **) >**
specifies the Haybittle-Peto method (Haybittle 1971; Peto et al. 1976). The values specified are used to create the boundary values. The boundary value at the final stage can be derived in the procedure to maintain the Type I and Type II error probability levels. The default is Z=3.

OBF | OBRIENFLEMING

specifies the O'Brien-Fleming method (O'Brien and Fleming 1979). The O'Brien-Fleming method is equivalent to a power family method with RHO=0.5.

POC | POCOCK

specifies the Pocock method (Pocock 1977). The Pocock method is equivalent to a power family method with RHO=0.

POW | POWER < (RHO=ρ) >

specifies a power family method (Wang and Tsiatis 1987; Emerson and Fleming 1989; Pampallona and Tsiatis 1994). The RHO= option specifies the power parameter ρ in the power family method, where $\rho \geq 0$. The power family method with $\rho = 0$ corresponds to the Pocock method, and the power family method with $\rho = 0.5$ corresponds to the O'Brien-Fleming method. The default is RHO=0.25, a value halfway between the Pocock and O'Brien-Fleming methods. A power family method is equivalent to a unified family method with RHO=ρ and TAU=0.

TRI | TRIANGULAR < (TAU=τ) >

specifies a unified family triangular method (Kittelson and Emerson 1999), where $0 \leq \tau \leq 1$. The default is TAU=1.0. The triangular method is identical to the unified family method with RHO=0.5 and TAU=τ. Note that this unified family triangular method is different from Whitehead's triangular method.

UNI | UNIFIED < (< TAU=τ > < RHO=ρ >) >

specifies a unified family method (Kittelson and Emerson 1999). The TAU= and RHO= options specify the τ and ρ parameters in a unified family method, respectively, where $\rho \geq 0$ and $0 \leq \tau \leq 2\rho$. The defaults are TAU=0 and RHO=0.25. See the section "Unified Family Methods" on page 9168 for a detailed description of the unified family methods.

The O'Brien-Fleming, Pocock, power family, and triangular methods are all special cases of the unified family methods. Table 110.3 summarizes the corresponding parameters in the unified family for these methods.

Table 110.3 Parameters in the Unified Family for Various Methods

		Unified Family	
Method	**Option**	**Rho**	**Tau**
Pocock	**POC**	0	0
O'Brien-Fleming	**OBF**	0.5	0
Power family	**POW** (**RHO=**ρ)	ρ	0
Triangular	**TRI** (**TAU=**τ)	0.5	τ

Note that the power parameter $\rho = 1/2 - \Delta = \rho^* - 1/2$, where Δ is the power parameter used in Jennison and Turnbull (2000) and Wang and Tsiatis (1987) and ρ^* is the power parameter used in Kittelson and Emerson (1999).

If a method with specified parameters is used for all boundaries in the design, you can use the METHOD= option to specify the method. Otherwise, you can use the following METHOD(*boundary*)= options to specify different methods from the same group for the boundaries.

METHOD(ALPHA)=*method*

METHOD(REJECT)=*method*

specifies the method for the α boundary of a one-sided design or the lower and upper α boundaries for a two-sided design.

METHOD(LOWERALPHA)=*method*

METHOD(LOWERREJECT)=*method*

specifies the method for the lower α boundary of a two-sided design.

METHOD(UPPERALPHA)=*method*

METHOD(UPPERREJECT)=*method*

specifies the method for the upper α boundary of a two-sided design.

METHOD(BETA)=*method*

METHOD(ACCEPT)=*method*

specifies the method for the β boundary of a one-sided design or the lower and upper β boundaries for a two-sided design.

METHOD(LOWERBETA)=*method*

METHOD(LOWERACCEPT)=*method*

specifies the method for the lower β boundary of a two-sided design.

METHOD(UPPERBETA)=*method*

METHOD(UPPERACCEPT)=*method*

specifies the method for the upper β boundary of a two-sided design.

NSTAGES=*number*

specifies the number of stages for the design. This option is required in the DESIGN statement, and the maximum allowed number of stages is 25.

STOP=ACCEPT

STOP=REJECT

STOP=BOTH < (BETABOUNDARY=BINDING | NONBINDING) >

specifies the early stopping criterion for the design. The keywords ACCEPT, REJECT, and BOTH correspond to early stopping only to accept, only to reject, and either to accept or reject the null hypothesis H_0, respectively. The default is STOP=REJECT. With STOP=BOTH, the BETABOUNDARY=BINDING suboption computes the Type I error probability with the acceptance boundary, and the BETABOUNDARY=NONBINDING suboption computes the Type I error probability without the acceptance boundary. The default is BETABOUNDARY=BINDING. The BETABOUNDARY=NONBINDING option is not applicable with METHOD=WHITEHEAD.

SAMPLESIZE Statement

SAMPLESIZE < *option* > **;**

If each observation in the data set provides one unit of information in a hypothesis testing (such as a one-sample test for the mean), the SAMPLESIZE statement computes the required sample sizes for each sequential design that is specified in a DESIGN statement. However, for a survival analysis, an individual in the survival time data might provide only partial information because of censoring. For this hypothesis, the SAMPLESIZE statement computes the required numbers of events. With additional accrual information in a survival analysis, the sample sizes can also be computed.

Only one SAMPLESIZE statement can be specified. For each specified group sequential design, the SAMPLESIZE statement computes the required sample sizes or numbers of events. The SAMPLESIZE statement is not required if the SEQDESIGN procedure is used only to compare features among different designs.

You can specify the following *option*:

MODEL < (CEILADJDESIGN=INCLUDE | EXCLUDE) > = *model-request*

specifies the input sample size or number of events from a fixed-sample study or specifies a statistical model to compute the required sample size. Table 110.4 summarizes the types of *model-request* that you can specify, and the subsections that follow Table 110.4 provide more information about the *model-request* for each type of model.

You can also specify one of the following values for the optional CEILADJDESIGN= suboption:

EXCLUDE does not create a ceiling-adjusted design.

INCLUDE creates a ceiling-adjusted design that corresponds to integer-valued sample sizes at the stages for nonsurvival data, and to integer-valued times or sample sizes at the stages for survival data.

Because the information levels in the adjusted design are different from the levels in the original design, the Type I and Type II error levels cannot be maintained simultaneously. If BOUNDARYKEY=BOTH or BOUNDARYKEY=NONE in the DESIGN statement, then the Type I error level is maintained. Otherwise, the BOUNDARYKEY= option selects the type of error level to be maintained.

The CEILADJDESIGN=INCLUDE option adds the ceiling-adjusted design information in the design information table, and creates a ceiling-adjusted boundary information table. See Example 110.2 for an illustration of the CEILADJDESIGN=INCLUDE option for nonsurvival data and Example 110.14 for an illustration of the option for survival data.

Other tables that are associated with the adjusted design, such as error-spending information, are not displayed in the SEQDESIGN procedure, but you can easily generate them from this adjusted-boundary information table in the companion SEQTEST procedure. See Example 110.2 for an illustration of this usage.

By default, CEILADJDESIGN=EXCLUDE.

Table 110.4 summarizes the types of *model-request* that you can specify.

Table 110.4 MODEL= Option

Option	Description
Fixed-Sample Models	
INPUTNOBS	Specifies the sample size for a fixed-sample design
INPUTNEVENTS	Specifies the number of events for a fixed-sample design
One-Sample Models	
ONESAMPLEMEAN	Specifies the one-sample Z test for the mean
ONESAMPLEFREQ	Specifies the one-sample test for the binomial proportion
Two-Sample Models	
TWOSAMPLEMEAN	Specifies the two-sample Z test for the mean difference
TWOSAMPLEFREQ	Specifies the two-sample test for binomial proportions
TWOSAMPLESURVIVAL	Specifies the log-rank test for two survival distributions
Regression Models	
REG	Specifies the test for a regression parameter
LOGISTIC	Specifies the test for a logistic regression parameter
PHREG	Specifies the test for a proportional hazards regression parameter

The MODEL=INPUTNOBS option specifies the input sample size from a fixed-sample study of nonsurvival data, and the MODEL=INPUTNEVENTS option specifies the number of events from a fixed-sample study of survival data. The remaining MODEL= options specify the statistical models that are used to compute the required sample size. The default is MODEL=TWOSAMPLEMEAN, the two-sample Z test for the mean difference.

When MODEL=INPUTNOBS or MODEL=INPUTNEVENTS, the required sample size or number of events for the group sequential trial is computed by multiplying the input sample size or number of events by the ratio of the design information level to its corresponding fixed-sample information level. This ratio can be obtained by dividing the Max Information (Percent Fixed-Sample) in the "Design Information" table by 100. For a description of the "Design Information" table, see the section "Design Information" on page 9211.

Fixed-Sample Models

The following two options compute the required sample size or number of events for a group sequential trial by using the sample size or number of events for the fixed-sample design:

MODEL=INPUTNOBS (*options* **)**

specifies the input sample size from a fixed-sample study of nonsurvival data. The available *options* are as follows:

- **N=***n*
- **SAMPLE=ONE | TWO**
- **WEIGHT=**w_a $< w_b >$
- **MATCHNOBS=YES | NO**

The required N=*n* option specifies the sample size *n* for the fixed-sample design. The SAMPLE=ONE option specifies a one-sample test, and the SAMPLE=TWO option specifies a two-sample test. The default is SAMPLE=ONE.

With a two-sample test, the WEIGHT= option specifies the sample size allocation weights for the two groups. If w_b is not specified, $w_b = 1$ is used. The default is WEIGHT=1, which results in equal allocation for the two groups. The derived fractional sample sizes are rounded up to integers, and the MATCHNOBS=YES option requests these integer sample sizes to match the sample size allocation.

For more information about the input sample size for the fixed-sample design in sample size computation, see the section "Input Sample Size for Fixed-Sample Design" on page 9187.

MODEL=INPUTNEVENTS (D=*d* < *options* >)

specifies the number of events from a fixed-sample study of survival data. The required D=*d* option specifies the fixed-sample number of events *d*. To derive the sample size, you need to specify additional *options* from the following list:

- **ACCNOBS=**n_a
- **ACCRATE=**r_a
- **ACCRUAL=UNIFORM | EXP(PARM=**γ_0**)**
- **ACCTIME=**T_a
- **CEILING=TIME | N**
- **FOLTIME=**T_f
- **HAZARD=**h_a < h_b >
- **LOSS=NONE | EXP(HAZARD=**τ **| MEDLOSSTIME=**t_τ **)**
- **MEDSURVTIME=**t_a < t_b >
- **SAMPLE=ONE | TWO**
- **TOTALTIME=***T*
- **WEIGHT=**w_a < w_b >

The SAMPLE=ONE option specifies a one-sample test, and the SAMPLE=TWO option specifies a two-sample test. The default is SAMPLE=ONE. With a two-sample test, the WEIGHT= option specifies the sample size allocation weights for the two groups. If w_b is not specified, w_b=1 is used. The default is WEIGHT=1.

For a one-sample test, the HAZARD= h_a option specifies the hazard rate h_a explicitly, and the MEDSURVTIME=t_a option specifies the hazard rate implicitly through the median survival time t_a. Similarly, for a two-sample test, the HAZARD= $h_a h_b$ option specifies the hazard rates h_a and h_b for groups A and B explicitly, and the MEDSURVTIME=$t_a t_b$ option specifies hazard rates for groups A and B implicitly through the median survival times t_a and t_b. Also, for a two-sample test, $h_b = h_a$ if h_b is not specified and $t_b = t_a$ if t_b is not specified.

The ACCRUAL= option specifies the method for individual accrual. The ACCRUAL=UNIFORM option (which is the default) specifies that the individual accrual is uniform in the accrual time T_a with a constant accrual rate r_a, and the ACCRUAL=EXP(PARM=γ_0) option specifies that the individual accrual is truncated exponential with a scaled power parameter γ_0, where $\gamma_0 \geq -10$ and $\gamma_0 \neq 0$. With a scaled parameter γ_0, the power parameter for the truncated exponential with the accrual time T_a is given by $\gamma = \gamma_0 / T_a$.

The ACCTIME= and FOLTIME= options specify the accrual time T_a and follow-up time T_f, respectively. The TOTALTIME= option specifies the total study time, $T = T_a + T_f$.

The ACCRATE= option specifies the constant accrual rate r_a. If you specify the ACCRATE= option, then you must also specify one of the ACCTIME=, FOLTIME=, and TOTALTIME= options for the sample size computation. Otherwise, you must specify two of the ACCTIME=, FOLTIME=, and TOTALTIME= options to compute the accrual rate and sample size.

The ACCNOBS= option specifies the total number of individuals in the study. If you specify the ACCNOBS= option, then you must also specify one of the ACCTIME=, FOLTIME=, and TOTALTIME= options for the sample size computation. Otherwise, you must specify two of the ACCTIME=, FOLTIME=, and TOTALTIME= options to compute the accrual rate and sample size.

The CEILING= option specifies the additional sample size information to be displayed in the "Number of Events (D) and Sample Sizes (N)" table. The CEILING=TIME option (which is the default) displays additional information that includes ceiling times at the stages, and the CEILING=N option displays additional information that includes ceiling sample sizes at the stages. When you specify the SAMPLE=TWO option along with the CEILING=N option, the ceiling sample sizes at each stage are derived to match the sample size allocation weights for the two groups.

The LOSS= option specifies the individual loss to follow-up in the sample size computation. The LOSS=NONE option (which is the default) specifies no loss to follow-up, and the LOSS=EXP option specifies exponential loss function. The HAZARD=τ suboption specifies the exponential loss hazard rate τ, and the MEDLOSSTIME=t_τ suboption specifies the exponential loss hazard rate through the median loss times t_τ.

For a detailed description of the input number of events for the fixed-sample design in sample size computation, see the section "Input Number of Events for Fixed-Sample Design" on page 9187.

One-Sample Models

The following two *options* compute the required sample size for a one-sample group sequential test:

MODEL=ONESAMPLEMEAN < (*options*) >
specifies the one-sample Z test for mean. The available *options* are as follows:

- **MEAN=** μ_1
- **STDDEV=** σ

The MEAN= option specifies the alternative reference μ_1 and is required if the alternative reference is not specified or derived in the procedure. If the MEAN=option is not specified, the specified or derived alternative reference is used.

The STDDEV= option specifies the standard deviation σ. The default is STDDEV=1. See the section "Test for a Normal Mean" on page 9193 for a detailed description of the one-sample Z test for mean.

Note that the one-sample Z test for mean also includes the paired difference in two-treatment comparison (Jennison and Turnbull 2000, pp. 51–52), where μ_1 is the mean of differences within pairs under the alternative hypothesis and σ is the standard deviation for the mean of differences within pairs.

MODEL=ONESAMPLEFREQ < (*options*) >

specifies the one-sample test for binomial proportion with the null hypothesis $H_0 : \theta = 0$ and the alternative hypothesis $H_1 : \theta = \theta_1$, where $\theta = p - p_0$ and $\theta_1 = p_1 - p_0$. The available *options* are as follows:

- **NULLPROP=** p_0
- **PROP=** p_1
- **REF= NULLPROP | PROP**

The NULLPROP= and PROP= options specify the proportions under the null and alternative hypotheses, respectively. The default for the null reference is NULLPROP=0.5. The PROP= option is required if the alternative reference is not specified or derived in the procedure. If the PROP= option is not specified, the specified or derived alternative reference θ_1 is used to compute the alternative reference $p_1 = p_0 + \theta_1$.

The REF= option specifies the hypothesis under which the proportion is used in the sample size computation. The REF=NULLPROP option uses the null hypothesis, and the REF=PROP option uses the alternative hypothesis to compute the sample size. The default is REF=PROP. See the section "Test for a Binomial Proportion" on page 9194 for a detailed description of the one-sample tests for proportion.

Two-Sample Models

The following three *options* compute the required sample size or number of events for a two-sample group sequential trial.

MODEL=TWOSAMPLEMEAN < (*options*) >

specifies the two-sample Z test for mean difference. The available *options* are as follows:

- **MEANDIFF=** θ_1
- **STDDEV=** σ_a $< \sigma_b >$
- **WEIGHT=** w_a $< w_b >$
- **MATCHNOBS= YES | NO**

The MEANDIFF= option specifies the alternative reference θ_1 and is required if the alternative reference is not specified or derived in the procedure. If the MEANDIFF= option is not specified, the specified or derived alternative reference is used.

The STDDEV= option specifies the standard deviations σ_a and σ_b. If σ_b is not specified, $\sigma_b = \sigma_a$. The default is STDDEV=1.

The WEIGHT= option specifies the sample size allocation weights for the two groups. If w_b is not specified, $w_b = 1$ is used. The default is WEIGHT=1, equal sample size for the two groups. The derived fractional sample sizes are rounded up to integers, and the MATCHNOBS=YES option requests these integer sample sizes to match the sample size allocation. The default is MATCHNOBS=NO.

See the section "Test for the Difference between Two Normal Means" on page 9196 for a detailed description of the two-sample Z test for mean difference.

MODEL=TWOSAMPLEFREQ < (*options*) >

specifies the two-sample test for binomial proportions. The available *options* are as follows:

- **NULLPROP=** $p_{0a} < p_{0b} >$
- **PROP=** p_{1a}
- **TEST= PROP | LOGOR | LOGRR**
- **REF= NULLPROP | PROP | AVGNULLPROP | AVGPROP**
- **WEIGHT=** $w_a < w_b >$
- **MATCHNOBS= YES | NO**

The NULLPROP= option specifies proportions $p_a = p_{0a}$ and $p_b = p_{0b}$ in groups A and B, respectively, under the null hypothesis. If p_{0b} is not specified, $p_{0b} = p_{0a}$. The default is NULLPROP=0.5.

The PROP= option specifies proportion $p_a = p_{1a}$ in group A under the alternative hypothesis. The proportion p_{1b} in group B under the alternative hypothesis is given by $p_{1b} = p_{0b}$. The PROP= option is required if the alternative reference is not specified or derived in the procedure. If the PROP= option is not specified, the specified or derived alternative reference is used to compute p_{1a}, the proportion in group A under the alternative hypothesis.

The TEST= option specified the null hypothesis $H_0 : \theta = 0$ in the test. The TEST=PROP option uses the difference in proportions $\theta = (p_a - p_b) - (p_{0a} - p_{0b})$, the TEST=LOGOR option uses the log odds-ratio test $\theta = \delta - \delta_0$, where

$$\delta = \log\left(\frac{p_a(1-p_b)}{p_b(1-p_a)}\right) \qquad \delta_0 = \log\left(\frac{p_{0a}(1-p_{0b})}{p_{0b}(1-p_{0a})}\right)$$

and the TEST=LOGRR option uses the log relative risk test with $\theta = \delta - \delta_0$, where

$$\delta = \log\left(\frac{p_a}{p_b}\right) \qquad \delta_0 = \log\left(\frac{p_{0a}}{p_{0b}}\right)$$

The default is TEST=LOGOR.

The REF= option specifies the hypothesis under which the proportions are used in the sample size computation. The REF=NULLPROP option uses the null proportions p_{0a} and p_{0b}, the REF=PROP option uses the alternative proportions p_{1a} and p_{1b}, the REF=AVGNULLPROP option uses the average null proportion, and the REF=AVGPROP option uses the average alternative proportion. The default is REF=PROP.

The WEIGHT= option specifies the sample size allocation weights for the two groups. If w_b is not specified, $w_b = 1$ is used. The default is WEIGHT=1, equal sample size for the two groups. The derived fractional sample sizes are also rounded up to integers, and the MATCHNOBS=YES option requests that these integer sample sizes match the sample size allocation.

See the section "Test for the Difference between Two Binomial Proportions" on page 9197, the section "Test for Two Binomial Proportions with a Log Odds Ratio Statistic" on page 9199, and the section "Test for Two Binomial Proportions with a Log Relative Risk Statistic" on page 9200 for a detailed description of the two-sample tests for proportions.

MODEL=TWOSAMPLESURVIVAL < (*options*) >

MODEL=TWOSAMPLESURV < (*options*) >

specifies the log-rank test for two survival distributions with the null hypothesis $H_0 : \theta = \delta - \delta_0 = 0$, where the parameter $\delta = -\log(h_a/h_b)$, δ_0 is the value of δ under the null hypothesis and the values h_a and h_b are the constant hazard rates for groups A and B, respectively.

The available *options* for the number of events are as follows:

- **NULLHAZARD=** $h_{0a} < h_{0b} >$
- **NULLMEDSURVTIME=** $t_{0a} < t_{0b} >$
- **HAZARD=** h_{1a}
- **HAZARDRATIO=** λ_1
- **MEDSURVTIME=** t_{1a}

The NULLHAZARD= option specifies hazard rates $h_a = h_{0a}$ and $h_b = h_{0b}$ for groups A and B, respectively, under the null hypothesis. If h_{0b} is not specified, $h_{0b} = h_{0a}$. The NULLMEDSURVTIME= option specifies the median survival times $t_a = t_{0a}$ and $t_b = t_{0b}$ under the null hypothesis. If t_{0b} is not specified, $t_{0b} = t_{0a}$. If both NULLHAZARD= and NULLMEDSURVTIME= option are not specified, NULLHAZARD=0.06931, which corresponds to NULLMEDSURVTIME=10, is used.

The hazard rate for group B under the alternative hypothesis, h_{1b}, is identical to the hazard rate under the null hypothesis, h_{0b}. The HAZARD=, MEDSURVTIME=, and HAZARDRATIO= options specify the group A hazard rate h_{1a}, the group A median survival time t_{1a}, and the hazard ratio $\lambda_1 = h_{1a}/h_{1b}$, respectively, under the alternative hypothesis. The HAZARD=, MEDSURVTIME=, or HAZARDRATIO= option is required if the alternative reference is not specified or derived in the procedure. If these three options are not specified, the specified or derived alternative reference θ_1 is used to compute h_{1a} from the equation:

$$\theta_1 = -\log\left(\frac{h_{1a}}{h_{1b}}\right) - \left(-\log\left(\frac{h_{0a}}{h_{0b}}\right)\right) = -\log\left(\frac{h_{1a}}{h_{0a}}\right)$$

To derive the sample size, you need to specify additional *options* from the following list:

- **ACCNOBS=**n_a
- **ACCRATE=**r_a
- **ACCRUAL=UNIFORM | EXP(PARM=**γ_0**)**
- **ACCTIME=**T_a
- **CEILING=TIME | N**
- **FOLTIME=**T_f
- **LOSS=NONE | EXP(HAZARD=**τ **| MEDLOSSTIME=**t_τ**)**
- **REF=NULLHAZARD | HAZARD**
- **TOTALTIME=***T*
- **WEIGHT=**w_a $< w_b >$

The REF= option specifies the hypothesis under which the hazard is used in the sample size computation. The REF=NULLHAZARD option uses the null hypothesis, and the REF=HAZARD option uses the alternative hypothesis. The default is REF=HAZARD.

The WEIGHT= option specifies the sample size allocation weights for the two groups. If w_b is not specified, $w_b = 1$ is used. The default is WEIGHT=1, equal sample size for the two groups.

The ACCRUAL= option specifies the method for individual accrual. The ACCRUAL=UNIFORM option (which is the default) specifies that the individual accrual is uniform in the accrual time T_a with a constant accrual rate r_a, and the ACCRUAL=EXP(PARM=γ_0) option specifies that the individual accrual is truncated exponential with a scaled power parameter γ_0, where $\gamma_0 \geq -10$ and $\gamma_0 \neq 0$. With a scaled parameter γ_0, the power parameter for the truncated exponential with the accrual time T_a is given by $\gamma = \gamma_0 / T_a$.

The ACCTIME= and FOLTIME= options specify the accrual time T_a and follow-up time T_f, respectively. The TOTALTIME= option specifies the total study time, $T = T_a + T_f$.

The ACCRATE= option specifies the constant accrual rate r_a. If you specify the ACCRATE= option, then you must also specify one of the ACCTIME=, FOLTIME=, and TOTALTIME= options for the sample size computation. Otherwise, you must specify two of the ACCTIME=, FOLTIME=, and TOTALTIME= options to compute the accrual rate and sample size.

The ACCNOBS= option specifies the total number of individuals in the study. If you specify the ACCNOBS= option, then you must also specify one of the ACCTIME=, FOLTIME=, and TOTALTIME= options for the sample size computation. Otherwise, you must specify two of the ACCTIME=, FOLTIME=, and TOTALTIME= options to compute the accrual rate and sample size.

The CEILING= option specifies the additional sample size information to be displayed in the "Number of Events (D) and Sample Sizes (N)" table. The CEILING=TIME option (which is the default) displays additional information that includes ceiling times at the stages, and the CEILING=N option displays additional information that includes ceiling sample sizes at the stages. When you specify CEILING=N, the ceiling sample sizes at each stage are derived to match the sample size allocation weights for the two groups.

The LOSS= option specifies the individual loss to follow-up in the sample size computation. The LOSS=NONE option (which is the default) specifies no loss to follow-up, and the LOSS=EXP option specifies exponential loss function. The HAZARD=τ suboption specifies the exponential loss hazard rate τ, and the MEDLOSSTIME=t_τ suboption specifies the exponential loss hazard rate through the median loss times t_τ.

See the section "Test for Two Survival Distributions with a Log-Rank Test" on page 9201 for a detailed description of the two-sample log-rank test for survival data.

Regression Models

The following three options compute the required sample size or number of events for group sequential tests on a regression parameter.

MODEL=REG < (*options*) >

specifies the Z test for a normal regression parameter. The available *options* are as follows:

- **BETA=** β_1
- **VARIANCE=** σ_y^2
- **XVARIANCE=** σ_x^2
- **XRSQUARE=** r_x^2

The BETA= option specifies the alternative reference β_1 and is required if the alternative reference is not specified or derived in the procedure. If the BETA= option is not specified, $\beta_1 = \theta_1$, the specified or derived alternative reference.

The VARIANCE= and XVARIANCE= options specify the variances for the response variable Y and covariate X, respectively. The defaults are VARIANCE=1 and XVARIANCE=1. For a model with more than one covariate, the XRSQUARE= option can be used to derive the variance of X after adjusting for other covariates. The default is XRSQUARE=0.

See the section "Test for a Parameter in the Regression Model" on page 9204 for a detailed description of the Z test for the regression parameter.

MODEL=LOGISTIC < (*options*) >

specifies the Z test for a logistic regression parameter. The available *options* are as follows:

- **BETA=** β_1
- **PROP=** p
- **XVARIANCE=** σ_x^2
- **XRSQUARE=** r_x^2

The BETA= option specifies the alternative reference β_1 and is required if the alternative reference is not specified or derived in the procedure. If the BETA= option is not specified, $\beta_1 = \theta_1$, the specified or derived alternative reference.

The PROP= option specifies the proportion of the binary response variable Y. The default is PROP=0.5. The XVARIANCE= option specifies the variance of the covariate X. The default is XVARIANCE=1. For a model with more than one covariate, the XRSQUARE= option can be used to derive the variance of X after adjusting for other covariates. The default is XRSQUARE=0.

See the section "Test for a Parameter in the Logistic Regression Model" on page 9205 for a detailed description of the Z test for the logistic regression parameter.

MODEL=PHREG < (*options*) >

specifies the Z test for a proportional hazards regression parameter. The available *options* for the number of events are as follows:

- **BETA=** β_1
- **XVARIANCE=** σ_x^2
- **XRSQUARE=** r_x^2

The BETA= option specifies the alternative reference β_1 and is required if the alternative reference is not specified or derived in the procedure. If the BETA= option is not specified, $\beta_1 = \theta_1$, the specified or derived alternative reference.

The XVARIANCE= option specifies the variance of the covariate X. The default is XVARIANCE=1. For a model with more than one covariate, the XRSQUARE= option can be used to derive the variance of X after adjusting for other covariates. The default is XRSQUARE=0.

To derive the sample size, you need to specify additional *options* from the following list:

- **ACCNOBS=**n_a
- **ACCRATE=**r_a
- **ACCRUAL=UNIFORM | EXP(PARM=**γ_0**)**
- **ACCTIME=**T_a
- **CEILING=TIME | N**
- **FOLTIME=**T_f
- **HAZARD=**h_a
- **LOSS=NONE | EXP(HAZARD=**τ **| MEDLOSSTIME=**t_τ**)**
- **MEDSURVTIME=**t_a
- **TOTALTIME=***T*

The hazard rate is required for the sample size computation. The HAZARD= h_a option specifies the hazard rate h_a explicitly, and the MEDSURVTIME=t_a option specifies the hazard rate implicitly through the median survival time t_a.

The ACCRUAL= option specifies the method for individual accrual. The ACCRUAL=UNIFORM option (which is the default) specifies that the individual accrual is uniform in the accrual time T_a with a constant accrual rate r_a, and the ACCRUAL=EXP(PARM=γ_0) option specifies that the individual accrual is truncated exponential with a scaled power parameter γ_0, where $\gamma_0 \geq -10$ and $\gamma_0 \neq 0$. With a scaled parameter γ_0, the power parameter for the truncated exponential with the accrual time T_a is given by $\gamma = \gamma_0 / T_a$.

The ACCTIME= and FOLTIME= options specify the accrual time T_a and follow-up time T_f, respectively. The TOTALTIME= option specifies the total study time, $T = T_a + T_f$.

The ACCRATE= option specifies the constant accrual rate r_a. If you specify the ACCRATE= option, then you must also specify one of the ACCTIME=, FOLTIME=, and TOTALTIME= options for the sample size computation. Otherwise, you must specify two of the ACCTIME=, FOLTIME=, and TOTALTIME= options to compute the accrual rate and sample size.

The ACCNOBS= option specifies the total number of individuals in the study. If you specify the ACCNOBS= option, then you must also specify one of the ACCTIME=, FOLTIME=, and TOTALTIME= options for the sample size computation. Otherwise, you must specify two of the ACCTIME=, FOLTIME=, and TOTALTIME= options to compute the accrual rate and sample size.

The CEILING= option specifies the additional sample size information to be displayed in the "Number of Events (D) and Sample Sizes (N)" table. The CEILING=TIME option (which is the default) displays additional information that includes ceiling times at the stages, and the CEILING=N option displays additional information that includes ceiling sample sizes at the stages.

The LOSS= option specifies the individual loss to follow-up in the sample size computation. The LOSS=NONE option (which is the default) specifies no loss to follow-up, and the LOSS=EXP option specifies exponential loss function. The HAZARD=τ suboption specifies the exponential loss hazard rate τ, and the MEDLOSSTIME=t_τ suboption specifies the exponential loss hazard rate through the median loss times t_τ.

For a detailed description of the Z test for the proportional hazards regression parameter, see the section "Test for a Parameter in the Proportional Hazards Regression Model" on page 9206.

Details: SEQDESIGN Procedure

Fixed-Sample Clinical Trials

A clinical trial is a research study in consenting human beings to answer specific health questions. One type of trial is a treatment trial, which tests the effectiveness of an experimental treatment. An example is a planned experiment designed to assess the efficacy of a treatment in humans by comparing the outcomes in a group of patients who receive the test treatment with the outcomes in a comparable group of patients who receive a placebo control treatment, where patients in both groups are enrolled, treated, and followed over the same time period.

A clinical trial is conducted according to a plan called a protocol. The protocol provides detailed description of the study. For a fixed-sample trial, the study protocol contains detailed information such as the null hypothesis, the one-sided or two-sided test, and the Type I and II error probability levels. It also includes the test statistic and its associated critical values in the hypothesis testing.

Generally, the efficacy of a new treatment is demonstrated by testing a hypothesis $H_0 : \theta = 0$ in a clinical trial, where θ is the parameter of interest. For example, to test whether a population mean μ is greater than a specified value μ_0, $\theta = \mu - \mu_0$ can be used with an alternative $\theta > 0$.

A one-sided test is a test of the hypothesis with either an upper (greater) or a lower (lesser) alternative, and a two-sided test is a test of the hypothesis with a two-sided alternative. The drug industry often prefers to use a one-sided test to demonstrate clinical superiority based on the argument that a study should not be run if the test drug would be worse (Chow, Shao, and Wang 2003, p. 28). But in practice, two-sided tests are commonly performed in drug development (Senn 1997, p. 161). For a fixed Type I error probability α, the sample sizes required by one-sided and two-sided tests are different. See Senn (1997, pp. 161–167) for a detailed description of issues involving one-sided and two-sided tests.

For independent and identically distributed observations $y_1, y_2, \ldots, y_n$ of a random variable, the likelihood function for θ is

$$L(\theta) = \prod_{j=1}^{n} L_i(\theta)$$

where θ is the population parameter and $L_i(\theta)$ is the probability or probability density of y_i. Using the likelihood function, two statistics can be derived that are useful for inference: the maximum likelihood estimator and the score statistic.

Maximum Likelihood Estimator

The maximum likelihood estimate (MLE) of θ is the value $\hat{\theta}$ that maximizes the likelihood function for θ. Under mild regularity conditions, $\hat{\theta}$ is an asymptotically unbiased estimate of θ with variance $1/E_\theta(I(\theta))$, where $I(\theta)$ is the Fisher information

$$I(\theta) = -\frac{\partial^2 \log(L(\theta))}{\partial \theta^2}$$

and $E_\theta(I(\theta))$ is the expected Fisher information (Diggle et al. 2002, p. 340)

$$E_\theta(I(\theta)) = -E_\theta\left(\frac{\partial^2 \log(L(\theta))}{\partial \theta^2}\right)$$

The score function for θ is defined as

$$S(\theta) = \frac{\partial \log(L(\theta))}{\partial \theta}$$

and usually, the MLE can be derived by solving the likelihood equation $S(\theta) = 0$. Asymptotically, the MLE is normally distributed (Lindgren 1976, p. 272):

$$\hat{\theta} \sim N\left(\theta,\ \frac{1}{E_\theta(I(\theta))}\right)$$

If the Fisher information $I(\theta)$ does not depend on θ, then $I(\theta)$ is known. Otherwise, either the expected information evaluated at the MLE $\hat{\theta}$ ($E_{\theta=\hat{\theta}}(I(\theta))$) or the observed information $I(\hat{\theta})$ can be used for the Fisher information (Cox and Hinkley 1974, p. 302; Efron and Hinkley 1978, p. 458), where the observed Fisher information

$$I(\hat{\theta}) = -\left(\frac{\partial^2 \log(L(\theta))}{\partial \theta^2} \mid \theta = \hat{\theta}\right)$$

If the Fisher information $I(\theta)$ does depend on θ, the observed Fisher information is recommended for the variance of the maximum likelihood estimator (Efron and Hinkley 1978, p. 457).

Thus, asymptotically, for large n,

$$\hat{\theta} \sim N\left(\theta,\ \frac{1}{I}\right)$$

where I is the information, either the expected Fisher information $E_{\theta=0}(I(\theta))$ or the observed Fisher information $I\hat{\theta}$).

So to test $H_0 : \theta = 0$ versus $H_1 : \theta \neq 0$, you can use the standardized Z test statistic

$$Z = \frac{\hat{\theta}}{\sqrt{\mathrm{Var}(\hat{\theta})}} = \hat{\theta}\sqrt{I} \sim N\,(0,\ 1)$$

and the two-sided p-value is given by

$$\mathrm{Prob}(|Z| > |z_0|) = 1 - 2\Phi(|z_0|)$$

where Φ is the cumulative standard normal distribution function and z_0 is the observed Z statistic.

If the BOUNDARYSCALE=SCORE is specified in the SEQDESIGN procedure, the boundary values for the test statistic are displayed in the score statistic scale. With the standardized Z statistic, the score statistic $S = Z\sqrt{I} = \hat{\theta} I$ and

$$S \sim N\,(0,\ I)$$

Score Statistic

The score statistic is based on the score function for θ,

$$S(\theta) = \frac{\partial \log(L(\theta))}{\partial \theta}$$

Under the null hypothesis $H_0 : \theta = 0$, the score statistic $S(0)$ is the first derivative of the log likelihood evaluated at the null reference 0:

$$S(0) = \frac{\partial \log(L(\theta))}{\partial \theta} \mid \theta = 0$$

Under regularity conditions, $S(0)$ is asymptotically normally distributed with mean zero and variance $E_{\theta=0}(I(\theta))$, the expected Fisher information evaluated at the null hypothesis $\theta = 0$ (Kalbfleisch and Prentice 1980, p. 45), where $I(\theta)$ is the Fisher information

$$I(\theta) = -E\left(\frac{\partial^2 \log(L(\theta))}{\partial \theta^2}\right)$$

That is, for large n,

$$S(0) \sim N\,(0,\ E_{\theta=0}(I(\theta)))$$

Asymptotically, the variance of the score statistic $S(0)$, $E_{\theta=0}(I(\theta))$, can also be replaced by the expected Fisher information evaluated at the MLE $\theta = \hat{\theta}$ ($E_{\theta=\hat{\theta}}(I(\theta))$), the observed Fisher information evaluated at the null hypothesis $\theta = 0$ ($I(0)$), or the observed Fisher information evaluated at the MLE $\theta = \hat{\theta}$ ($I(\hat{\theta})$) (Kalbfleisch and Prentice 1980, p. 46), where

$$I(0) = -\left(\frac{\partial^2 \log(L(\theta))}{\partial \theta^2} \mid \theta = 0\right)$$

$$I(\hat{\theta}) = -\left(\frac{\partial^2 \log(L(\theta))}{\partial \theta^2} \mid \theta = \hat{\theta}\right)$$

Thus, asymptotically, for large n,

$$S(0) \sim N\,(0,\ I)$$

where I is the information, either an expected Fisher information ($E_{\theta=0}(I(\theta))$ or $E_{\theta=\hat{\theta}}(I(\theta))$) or a observed Fisher information ($I(0)$ or $I(\hat{\theta})$).

So to test $H_0 : \theta = 0$ versus $H_1 : \theta \neq 0$, you can use the standardized Z test statistic

$$Z = \frac{S(0)}{\sqrt{I}}$$

If the BOUNDARYSCALE=MLE is specified in the SEQDESIGN procedure, the boundary values for the test statistic are displayed in the MLE scale. With the standardized Z statistic, the MLE statistic $\hat{\theta} = Z/\sqrt{I} = U(0)/I$ and

$$\hat{\theta} \sim N\left(0, \frac{1}{I}\right)$$

One-Sample Test for Mean

The following one-sample test for mean is used to demonstrate fixed-sample clinical trials in the section "One-Sided Fixed-Sample Tests in Clinical Trials" on page 9151 and the section "Two-Sided Fixed-Sample Tests in Clinical Trials" on page 9153.

Suppose $y_1, y_2, \ldots, y_n$ are n observations of a response variable Y from a normal distribution

$$y_i \sim N\left(\theta, \sigma^2\right)$$

where θ is the unknown mean and σ^2 is the known variance.

Then the log likelihood function for θ is

$$\log(L(\theta)) = \sum_{j=1}^{n} -\frac{1}{2}\frac{(y_j - \theta)^2}{\sigma^2} + c$$

where c is a constant. The first derivative is

$$\frac{\partial \log(L(\theta))}{\partial \theta} = \frac{1}{\sigma^2}\sum_{j=1}^{n}(y_j - \theta) = \frac{n}{\sigma^2}(\overline{y} - \theta)$$

where $\overline{y}$ is the sample mean.

Setting the first derivative to zero, the MLE of θ is $\hat{\theta} = \overline{y}$, the sample mean. The variance for $\hat{\theta}$ can be derived from the Fisher information

$$I(\theta) = -\frac{\partial^2 \log(L(\theta))}{\partial \theta^2} = \frac{n}{\sigma^2}$$

Since the Fisher information $I_0 = I(\theta)$ does not depend on θ in this case, $1/I_0$ is used as the variance for $\hat{\theta}$. Thus the sample mean $\overline{y}$ has a normal distribution with mean θ and variance σ^2/n:

$$\hat{\theta} = \overline{y} \sim N\left(\theta, \frac{1}{I_0}\right) = N\left(\theta, \frac{\sigma^2}{n}\right)$$

Under the null hypothesis $H_0 : \theta = 0$, the score statistic

$$S(0) = \frac{\partial \log(L(\theta))}{\partial \theta}|\theta = 0 = \frac{n}{\sigma^2}\overline{y}$$

has a mean zero and variance

$$I(\theta) = -\frac{\partial^2 \log(L(\theta))}{\partial \theta^2} = \frac{n}{\sigma^2}$$

With the MLE $\hat{\theta}$, the corresponding standardized statistic is computed as $Z = \hat{\theta}\sqrt{I_0} = \overline{y}/(\sigma/\sqrt{n})$, which has a normal distribution with variance 1:

$$Z \sim N\left(\theta\sqrt{I_0},\ 1\right) = N\left(\frac{\theta}{\sigma/\sqrt{n}},\ 1\right)$$

Also, the corresponding score statistic is computed as $S = \hat{\theta} I_0 = n\overline{y}/\sigma^2$ and

$$S \sim N\left(\theta I_0,\ I_0\right) = N\left(\frac{n\theta}{\sigma^2},\ \frac{n}{\sigma^2}\right)$$

which is identical to $S(0)$ computed under the null hypothesis $H_0 : \theta = 0$.

Note that if the variable Y does not have a normal distribution, then it is assumed that the sample size n is large such that the sample mean has an approximately normal distribution.

One-Sided Fixed-Sample Tests in Clinical Trials

A one-sided test has either an upper (greater) or a lower (lesser) alternative. This section describes one-sided tests with upper alternatives only. Corresponding results for one-sided tests with lower alternatives can be derived similarly.

For a one-sided test of $H_0 : \delta \le \delta_0$ with an upper alternative $H_1 : \delta > \delta_0$, an equivalent null hypothesis is $H_0 : \theta \le 0$ with an upper alternative $H_1 : \theta > 0$, where $\theta = \delta - \delta_0$. A fixed-sample test rejects H_0 if the standardized test statistic $Z_0 = \hat{\theta}\sqrt{I_0} \ge C_\alpha$, where $\hat{\theta}$ is the sample estimate of θ and $C_\alpha = \Phi^{-1}(1-\alpha)$ is the critical value.

The p-value of the test is given by $1 - \Phi(Z_0)$, and the hypothesis H_0 is rejected if the p-value is less than α. An upper $(1-\alpha)$ confidence interval has the lower limit

$$\theta_l = \hat{\theta} - \frac{\Phi^{-1}(1-\alpha)}{\sqrt{I_0}} = \frac{Z_0 - \Phi^{-1}(1-\alpha)}{\sqrt{I_0}}$$

The hypothesis H_0 is rejected if the confidence interval for the parameter θ does not contain zero—that is, if the lower limit θ_l is greater than 0.

With an alternative reference $\theta = \theta_1$, $\theta_1 > 0$, a Type II error probability is defined as

$$\beta = P_{\theta=\theta_1}(Z_0 < C_\alpha)$$

which is equivalent to

$$\beta = P_{\theta=\theta_1}\left(Z_0 - \theta_1\sqrt{I_0}\ <\ C_\alpha - \theta_1\sqrt{I_0}\right) = \Phi\left(C_\alpha - \theta_1\sqrt{I_0}\right)$$

Thus, $\Phi^{-1}(\beta) = C_\alpha - \theta_1\sqrt{I_0}$. Then, with $C_\alpha = \Phi^{-1}(1-\alpha)$,

$$\theta_1\sqrt{I_0} = \Phi^{-1}(1-\alpha) + \Phi^{-1}(1-\beta)$$

The drift parameter $\theta_1\sqrt{I_0}$ can be computed for specified α and β and the maximum information is given by

$$I_0 = \left(\frac{\Phi^{-1}(1-\alpha) + \Phi^{-1}(1-\beta)}{\theta_1}\right)^2$$

If the maximum information is available, then the required sample size can be derived. For example, in a one-sample test for the mean with a specific standard deviation σ, the sample size n required for the test is

$$n = \sigma^2 I_0 = \sigma^2\left(\frac{\Phi^{-1}(1-\alpha) + \Phi^{-1}(1-\beta)}{\theta_1}\right)^2$$

On the other hand, if the alternative reference θ_1, standard deviation σ, and sample size n are all specified, then α can be derived for a given β and, similarly, β can be derived for a given α.

With an alternative reference $\theta = \theta_1$, $\theta_1 > 0$, the power $1-\beta$ is the probability of correctly rejecting the null hypothesis H_0 at θ_1:

$$1-\beta = 1 - P_{\theta=\theta_1}(Z_0 < C_\alpha) = \Phi\left(\theta_1\sqrt{I_0} - C_\alpha\right)$$

Superiority Trials

A superiority trial that tests the response to a new drug is clinically superior to a comparative placebo control or active control therapy. If a positive value indicates a beneficial effect, a test for superiority has

$$H_0 : \theta \le 0 \qquad H_1 : \theta > 0$$

where H_0 is the hypothesis of nonsuperiority and H_1 is the alternative hypothesis of superiority.

The superiority test rejects the hypothesis H_0 and declares superiority if the standardized statistic $Z_0 = \hat{\theta}\sqrt{I_0} \ge C_\alpha$, where the critical value $C_\alpha = \Phi^{-1}(1-\alpha)$.

For example, if θ is the response difference between the treatment and placebo control groups, then a superiority trial can be

$$H_0 : \theta \le 0 \qquad H_1 : \theta = 6$$

with a Type I error probability level $\alpha = 0.025$ and a power $1-\beta = 0.90$ at $\theta_1 = 6$.

Noninferiority Trials

A noninferiority trial does not compare the response to a new treatment with the response to a placebo. Instead, it demonstrates the effectiveness of a new treatment compared with that of a nonexisting placebo by showing that the response of a new treatment is not clinically inferior to the response of a standard therapy with an established effect. That is, this type of trial attempts to demonstrate that the new treatment effect is not worse than the standard therapy effect by an acceptable margin. These trials are often performed when there is an existing effective therapy for a serious disease, and therefore a placebo control group cannot be ethically included.

It can be difficult to specify an appropriate noninferiority margin. One practice is to choose with reference to the effect of the active control in historical placebo-controlled trials (Snapinn 2000, p. 20). With this practice, there is some basis to imply that the new treatment is better than the placebo for a positive noninferiority trial.

If a positive value indicates a beneficial effect, a test for noninferiority has a null hypothesis $\delta \leq -\delta_0$ and an alternative hypothesis $\delta = \delta_1 > -\delta_0$, where $\delta_0 > 0$ is the specified noninferiority margin.

An equivalent test has

$$H_0 : \theta \leq 0 \qquad H_1 : \theta = \theta_1 > 0$$

where the parameter $\theta = \delta + \delta_0$, H_0 is the null hypothesis of inferiority, and H_1 is the alternative hypothesis of noninferiority,

The noninferiority test rejects the hypothesis H_0 and declares noninferiority if the standardized statistic $Z_0 = \hat{\theta}\sqrt{I_0} = (\hat{\delta} + \delta_0)\sqrt{I_0} \geq C_\alpha$, where the critical value $C_\alpha = \Phi^{-1}(1-\alpha)$.

For example, if δ is the response difference between the treatment and active control groups and $\delta_0 = 2$ is the noninferiority margin, then a noninferiority trial with a power $1 - \beta = 0.90$ at $\delta_1 = 1$ might be

$$H_0 : \theta \leq 0 \qquad H_1 : \theta = 3$$

where $\theta = \delta + \delta_0 = \delta + 2$.

Two-Sided Fixed-Sample Tests in Clinical Trials

A two-sided test is a test of a hypothesis with a two-sided alternative. Two-sided tests include simple symmetric tests and more complicated asymmetric tests that might have distinct lower and upper alternative references.

Symmetric Two-Sided Tests for Equality

For a symmetric two-sided test with the null hypothesis $\delta = \delta_0$ against the alternative $\delta \neq \delta_0$, an equivalent null hypothesis is $H_0 : \theta = 0$ with a two-sided alternative $H_1 : \theta \neq 0$, where $\theta = \delta - \delta_0$. A fixed-sample test rejects H_0 if $|\hat{\theta}\sqrt{I_0}| \geq C_{\alpha/2}$, where $\hat{\theta}$ is a sample estimate of θ and $C_{\alpha/2} = \Phi^{-1}(1-\alpha/2)$ is the critical value.

A common two-sided test is the test for the response difference between a treatment group and a control group. The null and alternative hypotheses are $H_0 : \theta = 0$ and $H_1 : \theta \neq 0$, respectively, where θ is the response difference between the two groups. If a greater value indicates a beneficial effect, then there are three possible results:

- The test rejects the hypothesis H_0 of equality and indicates that the treatment is significantly better if the standardized statistic $Z_0 = \hat{\theta}\sqrt{I_0} \geq C_{\alpha/2}$.
- The test rejects the hypothesis H_0 and indicates the treatment is significantly worse if the standardized statistic $Z_0 = \hat{\theta}\sqrt{I_0} \leq -C_{\alpha/2}$.
- The test indicates no significant difference between the two responses if $-C_{\alpha/2} < \hat{\theta}\sqrt{I_0} < C_{\alpha/2}$.

The p-value of the test is $2(1 - \Phi(Z_0))$ if $Z_0 > 0$ and $2\Phi(Z_0)$ if $Z_0 \leq 0$. The hypothesis H_0 is rejected if the p-value of the test is less than α—that is, if $1 - \Phi(Z_0) < \alpha/2$ or $\Phi(Z_0) < \alpha/2$. A symmetric $(1 - \alpha)$ confidence interval for θ has lower and upper limits

$$\left(\hat{\theta} - \frac{C_{\alpha/2}}{\sqrt{I_0}}\,,\ \ \hat{\theta} + \frac{C_{\alpha/2}}{\sqrt{I_0}}\right)$$

which is

$$\left(\frac{1}{\sqrt{I_0}}\left(Z_0 - C_{\alpha/2}\right)\ ,\ \ \frac{1}{\sqrt{I_0}}\left(Z_0 + C_{\alpha/2}\right)\right)$$

The hypothesis H_0 is rejected if the confidence interval for the parameter θ does not contain zero. That is, the lower limit is greater than zero or the upper limit is less than zero.

With an alternative reference $\theta = \theta_1 > 0$, a Type II error probability is defined as

$$\beta = P_{\theta=\theta_1}(-C_{\alpha/2}\ <\ Z_0\ <\ C_{\alpha/2})$$

which is

$$\beta = P_{\theta=\theta_1}\left((-C_{\alpha/2} - \theta_1\sqrt{I_0})\ <\ (Z_0 - \theta_1\sqrt{I_0})\ <\ (C_{\alpha/2} - \theta_1\sqrt{I_0})\right)$$

Thus

$$\beta = \Phi\left(C_{\alpha/2} - \theta_1\sqrt{I_0}\right) - \Phi\left(-C_{\alpha/2} - \theta_1\sqrt{I_0}\right)$$

The resulting power $1 - \beta$ is the probability of correctly rejecting the null hypothesis, which includes the probability for the lower alternative and the probability for the upper alternative. The SEQDESIGN procedure uses only the probability of correctly rejecting the null hypothesis for the correct alternative in the power computation.

Thus, under the upper alternative hypothesis, the power in the SEQDESIGN procedure is computed as the probability of rejecting the null hypothesis for the upper alternative, $1. - \Phi\left(C_{\alpha/2} - \theta_1\sqrt{I_0}\right) = \Phi\left(\theta_1\sqrt{I_0} - C_{\alpha/2}\right)$, and a very small probability of rejecting the null hypothesis for the lower alternative, $\Phi\left(-C_{\alpha/2} - \theta_1\sqrt{I_0}\right)$, is ignored. This power computation is more rational than the power based on the probability of correctly rejecting the null hypothesis (Whitehead 1997, p. 75).

That is,

$$\beta = P_{\theta=\theta_1}\left((Z_0 - \theta_1\sqrt{I_0})\ <\ (C_{\alpha/2} - \theta_1\sqrt{I_0})\right) = \Phi\left(C_{\alpha/2} - \theta_1\sqrt{I_0}\right)$$

Then with $\Phi^{-1}(\beta) = C_{\alpha/2} - \theta_1\sqrt{I_0}$,

$$\theta_1\sqrt{I_0} = C_{\alpha/2} - \Phi^{-1}(\beta) = \Phi^{-1}(1-\alpha/2) + \Phi^{-1}(1-\beta)$$

The drift parameter $\theta_1\sqrt{I_0}$ can be derived for specified α and β, and the maximum information is given by

$$I_0 = \left(\frac{\Phi^{-1}(1-\alpha/2) + \Phi^{-1}(1-\beta)}{\theta_1}\right)^2$$

If the maximum information is available, then the required sample size can be derived. For example, in a one-sample test for mean, if the standard deviation σ is known, the sample size n required for the test is

$$n = \sigma^2 I_0 = \sigma^2\left(\frac{\Phi^{-1}(1-\alpha/2) + \Phi^{-1}(1-\beta)}{\theta_1}\right)^2$$

On the other hand, if the alternative reference θ_1, standard deviation σ, and sample size n are all known, then α can be derived with a given β and, similarly, β can be derived with a given α.

Generalized Two-Sided Tests for Equality

For a generalized two-sided test with the null hypothesis $\delta = \delta_0$ against the alternative $\delta \neq \delta_0$, an equivalent null hypothesis is $H_0 : \theta \leq 0$ with a two-sided alternative $H_1 : \theta \neq 0$, where $\theta = \delta - \delta_0$. A fixed-sample test rejects H_0 if the standardized statistic $Z_0 = \hat{\theta}\sqrt{I_0} < -C_{\alpha_l}$ or $Z_0 = \hat{\theta}\sqrt{I_0} > C_{\alpha_u}$, where the critical values $C_{\alpha_l} = \Phi^{-1}(1-\alpha_l)$ and $C_{\alpha_u} = \Phi^{-1}(1-\alpha_u)$.

With the lower alternative reference $\theta_{1l} < 0$, a lower Type II error probability is defined as

$$\beta_l = P_{\theta=\theta_{1l}}\left(-C_{\alpha_l} \leq Z_{0l}\sqrt{I_0}\right) = P_{\theta=\theta_{1l}}\left(-C_{\alpha_l} - \theta_{1l}\sqrt{I_0} \leq Z_{0l}\sqrt{I_0} - \theta_{1l}\sqrt{I_0}\right)$$

This implies

$$\beta_l = 1 - \Phi(-C_{\alpha_l} - \theta_{1l}\sqrt{I_0})$$

and the power is the probability of correctly rejecting the null hypothesis for the lower alternative,

$$1 - \beta_l = \Phi(-C_{\alpha_l} - \theta_{1l}\sqrt{I_0})$$

The lower drift parameter is derived as

$$\theta_{1l}\sqrt{I_0} = -\left(\Phi^{-1}(1-\alpha_l) + \Phi^{-1}(1-\beta_l)\right)$$

Then, with specified α_l and β_l, if the maximum information is known, the lower alternative reference θ_{1l} can be derived. If the maximum information is unknown, then with the specified lower alternative reference θ_{1l}, the maximum information required is

$$I_{0l} = \left(\frac{\Phi^{-1}(1-\alpha_l) + \Phi^{-1}(1-\beta_l)}{-\theta_{1l}}\right)^2$$

Similarly, the upper drift parameter is derived as

$$\theta_{1u}\sqrt{I_0} = \Phi^{-1}(1-\alpha_u) + \Phi^{-1}(1-\beta_u)$$

For a given α_u, β_u, and the upper alternative reference θ_{1u}, the maximum information required is

$$I_{0u} = \left(\frac{\Phi^{-1}(1-\alpha_u) + \Phi^{-1}(1-\beta_u)}{\theta_{1u}}\right)^2$$

Thus, the maximum information required for the design is given by

$$I_0 = \max(I_{0l},\ I_{0u})$$

Note that with the maximum information level I_0, if $I_{0l} < I_0$, then the derived power from the lower alternative is larger than the specified $1-\beta_l$. Similarly, if $I_{0u} < I_0$, then the derived power from the upper alternative is larger than the specified $1-\beta_u$.

If maximum information is available, the required sample size can be derived. For example, in a one-sample test for mean, if the standard deviation σ is known, the sample size n required for the test is $n = \sigma^2 I_0$.

On the other hand, if the alternative references, Type I error probabilities α_l and α_u, standard deviation σ, and sample size n are all specified, then the Type II error probabilities β_l and β_u and the corresponding powers can be derived.

Group Sequential Methods

A group sequential design provides interim analyses before the formal completion of a trial. The monitoring process provides possible early stopping for either positive or negative results and thus reduces the time to complete the trial. With a specified number of stages, the design creates critical values such that at each interim analysis, a hypothesis can be rejected, accepted, or continued to the next time point. At the final stage, a hypothesis is either rejected or accepted. Usually, the critical values are derived such that the specified overall Type I and Type II error probability levels are maintained in the design.

For example, to test a null hypothesis H_0 with an upper alternative in a fixed-sample design, a critical value c_α is created. The null hypothesis H_0 is rejected if the test statistic is greater than or equal to the critical value c_α. Otherwise, H_0 is accepted. But, for a group sequential design with early stopping to reject or accept the null hypothesis H_0, there are two critical values created at each interim analysis: an α critical value $c_{\alpha k}$ to reject the null hypothesis and a β critical value $c_{\beta k}$ to accept the null hypothesis. The null hypothesis H_0 is rejected if the test statistic is greater than or equal to the α critical value $c_{\alpha k}$, and H_0 is accepted if the test statistic is less than the β critical value $c_{\beta k}$. If the test statistic is between these two critical values, the process continues to the next stage. At the final stage, the two critical values are equal, and the hypothesis is either rejected or accepted.

Armitage, McPherson, and Rowe (1969) showed that repeated significance tests at a fixed level on accumulating data increase the probability of obtaining a significant result under the null hypothesis. For example, with a significance level 0.05 in a two-sided fixed-sample test, the critical value is 1.96. If this value is used in a five-stage group sequential trial with early stopping to reject the null hypothesis, then the probability of rejecting the null hypothesis at or before the fifth stage is 0.14169, much larger than the nominal value 0.05 (Armitage, McPherson, and Rowe 1969, p. 239).

Pocock (1977) applied these repeated significance tests to group sequential trials with equally spaced information levels and derives a constant critical value on the standardized normal Z scale across all stages that maintains the Type I error probability level. For example, with a significance level 0.05 in a two-sided test, the derived critical value at each stage is 2.413 on the standardized normal Z scale, larger than the fixed-sample critical value 1.96. The corresponding nominal p-value is 0.0158, which is smaller than the fixed-sample p-value 0.025 (Pocock 1977, p. 193).

O'Brien and Fleming (1979) proposed a sequential procedure that has boundary values decrease over the stages on the standardized normal Z scale to make the early stop less likely. The procedure has conservative stopping boundary values at very early stages, and boundary values at the final stage are close to the fixed-sample design. For example, with a significance level 0.05 in a two-sided test, the derived critical values at these five stages on the standardized normal Z scale are 4.562, 3.226, 2.634, 2.281, and 2.040.

The following references: Wang and Tsiatis (1987); Emerson and Fleming (1989); Pampallona and Tsiatis (1994) generalized the Pocock and O'Brien-Fleming methods to the power family, where a power parameter is used to allow a continuous set of designs between the Pocock and O'Brien-Fleming methods.

Kittelson and Emerson (1999) extended the methods in the power family even further to the unified family, which also includes the exact triangular method. The shape and location of each of the four boundaries can be independently specified in the unified family methods.

Whitehead and Stratton (1983) and Whitehead (1997, 2001) developed triangular methods by adapting tests for continuous monitoring to discrete monitoring. With early stopping to reject or accept the null hypothesis in a one-sided test, the derived continuation region has a triangular shape for the score-scaled boundaries. Only elementary calculations are needed to derive the boundary values for Whitehead's triangular methods.

For a sequential design, you can derive the α and β error probabilities at each stage from the boundaries. On the other hand, you can derive the boundaries from specified α and β error probabilities at each stage. The error spending function approach (Lan and DeMets 1983) uses the error spending function to specify the error probabilities at each stage and then uses these probabilities to derive the boundaries. You can specify α and β explicitly or implicitly with an error spending function for the cumulative probabilities.

See Jennison and Turnbull (2000, pp. 5–11) for a more detailed history of group sequential methods.

The following three types of methods are available in the SEQDESIGN procedure to derive boundaries in a sequential design:

- fixed boundary shape methods, which derive boundaries with specified boundary shapes. These include the unified family method and Haybittle-Peto method.
- Whitehead methods, which adjust the boundaries from continuous monitoring for discrete monitoring
- error spending methods

You can use the SEQDESIGN procedure to specify methods from the same group for each design. A different method can be specified for each boundary separately, but all methods in a design must be from the same group.

Fixed Boundary Shape Methods

The fixed boundary shape methods include the unified family method (Kittelson and Emerson 1999) and the Haybittle-Peto method (Haybittle 1971; Peto et al. 1976). The unified family methods derive the boundary values with the specified boundary shape. The unified family methods include the Pocock method (Pocock 1977), the O'Brien-Fleming method (O'Brien and Fleming 1979), the power family method (Wang and Tsiatis 1987; Emerson and Fleming 1989; Pampallona and Tsiatis 1994), and the triangular method (Kittelson and Emerson 1999). See the section "Unified Family Methods" on page 9168 for a detailed description of the methods that use the unified family approach.

The Haybittle-Peto method uses a value of 3 for the critical values in interim stages, so that the critical value at the final stage is close to the original design without interim monitoring. In the SEQDESIGN procedure, the Haybittle-Peto method has been generalized to allow for different boundary values at different stages. See the section "Haybittle-Peto Method" on page 9173 for a detailed description of the Haybittle-Peto method.

Whitehead Methods

The Whitehead methods (Whitehead and Stratton 1983; Whitehead 1997, 2001) derive the boundary values by adapting the continuous monitoring tests to the discrete monitoring of group sequential tests. The Type I error probability and power corresponding to the resulting boundaries are extremely close but differ slightly from the specified values because of the approximations used in deriving the tests (Jennison and Turnbull 2000, p. 106). The SEQDESIGN procedure provides the BOUNDARYKEY= option to adjust the boundary value at the final stage for the exact Type I or Type II error probability level. See the section "Whitehead Methods" on page 9173 for a detailed description of Whitehead's methods.

Error Spending Methods

An error spending method (Lan and DeMets 1983) uses the error spending function to specify the error spending at each stage and then uses these error probabilities to derive the boundary values. You can specify these errors explicitly or with an error spending function for these cumulative errors. See the section "Error Spending Methods" on page 9177 for a detailed description of the error spending methods.

Error spending methods derive boundary values at each stage sequentially and require much more computation than other types of methods for group sequential trials with a large number of stages, especially for a two-sided asymmetric design with early stopping to accept H_0, or to reject or accept H_0.

The sample size requirement for some applicable tests can also be computed in the procedure. After the actual data from a clinical trial are collected, you can then use the boundary information created in the SEQDESIGN procedure to perform a group sequential test in the SEQTEST procedure.

Statistical Assumptions for Group Sequential Designs

The SEQDESIGN procedure assumes that with a total number of stages K, the sequence of the standardized test statistics $\{Z_1, Z_2, \ldots, Z_K\}$ has the canonical joint distribution with information levels $\{I_1, I_2, \ldots, I_K\}$ for the parameter θ (Jennison and Turnbull 2000, p. 49):

- $(Z_1, Z_2, \ldots, Z_K)$ is multivariate normal
- $Z_k \sim N\left(\theta\sqrt{I_k},\ 1\right),\ k = 1, 2, \ldots, K$
- $\mathrm{Cov}(Z_{k_1}, Z_{k_2}) = \sqrt{(I_{k_1}/I_{k_2})},\quad 1 \le k_1 \le k_2 \le K$

In terms of the maximum likelihood estimator, $\hat{\theta}_k = Z_k/\sqrt{I_k}$, $k = 1, 2, \ldots, K$, the canonical joint distribution can be expressed as follows:

- $(\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K)$ is multivariate normal
- $\hat{\theta}_k \sim N\left(\theta,\ 1/I_k\right),\ k = 1, 2, \ldots, K$
- $\mathrm{Cov}(\hat{\theta}_{k_1}, \hat{\theta}_{k_2}) = 1/I_{k_2},\quad 1 \le k_1 \le k_2 \le K$

Furthermore, in terms of the score statistics $S_k = Z_k\sqrt{I_k}$, $k = 1, 2, \ldots, K$, the canonical joint distribution can be expressed as follows:

- $(S_1, S_2, \ldots, S_K)$ is multivariate normal
- $S_k \sim N\left(\theta\, I_k,\ I_k\right),\ k = 1, 2, \ldots, K$
- $\mathrm{Cov}(S_{k_1}, S_{k_2}) = \mathrm{Var}(S_{k_1}) = I_{k_1},\quad 1 \le k_1 \le k_2 \le K$

That is, the increments S_1, $S_2 - S_1, \ldots,$ and $S_K - S_{(K-1)}$ are independently distributed.

If the test statistic is computed from the data that are not from a normal distribution, such as a binomial distribution, then it is assumed that the test statistic is computed from a large sample such that the statistic has an approximately normal distribution.

If the increments S_1, $S_2 - S_1, \ldots,$ and $S_K - S_{(K-1)}$ are not independently distributed, then it is inappropriate to use group sequential methods in the SEQDESIGN procedure. One such example is the Gehan statistic, which is a weighted log-rank statistic for censored data. See Jennison and Turnbull (2000, pp. 232–233, 276–277) and Proschan, Lan, and Wittes (2006, pp. 150–151) for a description of statistics with nonindependent increments.

If a trial stops at an early interim stage with only a small number of responses observed, it can lead to a distrust of the statistical findings, which rely on the assumption that the sample is large (Whitehead 1997, p. 167). A group sequential design can be specified such that at the first interim analysis, there are a sufficient number of responses to ensure that the analysis to be conducted is both reliable and persuasive (Whitehead 1997, p. 167).

Alternatively, a method such as the O'Brien-Fleming method can be used to derive conservative stopping boundary values at very early stages to make the early stop less likely. That is, the trial is stopped in early stages only with overwhelming evidence.

A simple example of the group sequential tests is the test for a normal mean, $\mu = \mu_0$. Suppose $y_1, y_2, \ldots, y_n$ are n observations of a response variable Y in a data set from a normal distribution with an unknown mean μ and a known variance σ^2. Then the maximum likelihood estimate of μ is the sample mean

$$\overline{y} = \frac{1}{n}\sum_{j=1}^{n} y_j$$

The sample mean has a normal distribution with mean μ and variance σ^2/n:

$$\overline{y} \sim N\left(\mu, \frac{\sigma^2}{n}\right)$$

An equivalent hypothesis for $\mu = \mu_0$ is $H_0 : \theta = 0$, where $\theta = \mu - \mu_0$. The MLE statistic for θ,

$$\hat{\theta} = \overline{y} - \mu_0 \sim N\left(\theta, {I_0}^{-1}\right)$$

where the information $I_0 = n/\sigma^2$.

For a group sequential test with K stages, there are $N_1, N_2, \ldots, N_K$ observations available at these stages. At stage k, the sample mean is computed as

$$\overline{y}_k = \frac{1}{N_k}\sum_{j=1}^{N_k} y_{kj}$$

where y_{kj} is the value of the jth observation available at the kth stage and N_k is the cumulative sample size at stage k, which includes the N_{k-1} observations collected at previous stages and the $N_k - N_{k-1}$ observations collected at the current stage.

The maximum likelihood estimate

$$\hat{\theta}_k = \overline{y}_k - \mu_0 \sim N\left(\theta, {I_k}^{-1}\right)$$

where the information

$$I_k = \frac{1}{\mathrm{Var}(\overline{y}_k)} = \frac{N_k}{\sigma^2}$$

is the inverse of the variance.

Thus, the standardized statistic

$$Z_k = \hat{\theta}_k\sqrt{I_k} = (\overline{y}_k - \mu_0)\sqrt{I_k} \sim N\left(\theta\sqrt{I_k}, 1\right)$$

The covariance of Z_{k_1} and Z_{k_2}, $1 \le k_1 \le k_2 \le K$ can be expressed as

$$\mathrm{Cov}(Z_{k_1}, Z_{k_2}) = \frac{1}{\sqrt{(I_{k_1} I_{k_2})}}\,\mathrm{Cov}(S_{k_1}, S_{k_2})$$

where $S_{k_1} = Z_{k_1}\sqrt{I_{k_1}}$ and $S_{k_2} = Z_{k_2}\sqrt{I_{k_2}}$.

Since $S_{k_2} - S_{k_1}$ is independent of S_{k_1}, $\mathrm{Cov}(S_{k_1}, S_{k_2}) = \mathrm{Var}(S_{k_1}) = I_{k_1}$ and

$$\mathrm{Cov}(Z_{k_1}, Z_{k_2}) = \frac{1}{\sqrt{(I_{k_1} I_{k_2})}} I_{k_1} = \sqrt{I_{k_1}/I_{k_2}}$$

Thus the statistics $\{Z_1, Z_2, \ldots, Z_K\}$ has the canonical joint distribution with information levels $\{I_1, I_2, \ldots, I_K\}$ for the parameter μ. See the section "Applicable One-Sample Tests and Sample Size Computation" on page 9193, the section "Applicable Two-Sample Tests and Sample Size Computation" on page 9195, and the section "Applicable Regression Parameter Tests and Sample Size Computation" on page 9203 for more examples of applicable tests in group sequential trials.

Boundary Scales

The boundaries computed by the SEQDESIGN procedure are applied to test statistics computed during the analysis, and so generally, the scale you select for the boundaries is determined by the scale of the statistics that you will be using.

The following scales are available in the SEQDESIGN procedure:

- MLE, maximum likelihood estimate
- standardized Z
- score statistic S
- p-value

These scales are all equivalent for a given set of boundary values—that is, there exists a unique transformation between any two of these scales. If you know the boundary values in terms of statistics from one scale, you can uniquely derive the boundary values of statistics for other scales. You can specify the scale with the BOUNDARYSCALE= option; the default is BOUNDARYSCALE=STDZ, the standardized Z scale.

You can also select the boundary scale to better examine the features of an individual group sequential design or to compare features among multiple designs. For example, with the standardized Z scale, the boundary values for the Pocock design are identical across all stages, and the O'Brien-Fleming design has boundary values (in absolute value) that decrease over the stages.

The remaining section demonstrates the transformations from one scale to the other scales. If the maximum likelihood estimate $\hat{\theta}$ is computed by the analysis, then

$$\hat{\theta} \sim N\left(\theta,\ \frac{1}{I}\right)$$

where I is the Fisher information if it does not depend on θ. Otherwise, I is either the expected Fisher information evaluated at $\hat{\theta}$ or the observed Fisher information. See the section "Maximum Likelihood Estimator" on page 9148 for a detailed description of these statistics.

With the MLE statistic $\hat{\theta}$, the corresponding standardized Z statistic is computed as

$$Z = \hat{\theta}\sqrt{I} \sim N\left(\theta\sqrt{I},\ 1\right)$$

and the corresponding score statistic is computed as

$$S = \hat{\theta}\, I \sim N\,(\theta\, I,\ I)$$

Similarly, if a score statistic S is computed by the analysis, then with

$$S \sim N\,(\theta\, I,\ I)$$

where I is the information, either an expected Fisher information ($E_{\theta=0}(I(\theta))$ or $E_{\theta=\hat{\theta}}(I(\theta))$) or an observed Fisher information ($I(0)$ or $I(\hat{\theta})$).

The corresponding standardized Z statistic is computed as

$$Z = \frac{S}{\sqrt{I}} \sim N\left(\theta\sqrt{I},\ 1\right)$$

and the corresponding MLE-scaled statistic is computed as

$$\hat{\theta} = \frac{S}{I} \sim N\left(\theta,\ \frac{1}{I}\right)$$

With a standardized normal Z statistic, the corresponding fixed-sample nominal p-value depends on the type of alternative hypothesis. With an upper alternative, the nominal p-value is defined as the one-sided p-value under the null hypothesis $H_0 : \theta = 0$ with an upper alternative:

$$p_k = 1 - \Phi(Z)$$

With a lower alternative or a two-sided alternative, the nominal p-value is defined as the one-sided p-value under the null hypothesis $H_0 : \theta = 0$ with a lower alternative:

$$p_k = \Phi(Z)$$

which is an increasing function of the standardized Z statistic (Emerson, Kittelson, and Gillen 2005, p. 12).

The BOUNDARYSCALE= MLE, STDZ, SCORE, and PVALUE options display the boundary values in the MLE, standardize Z, score, and p-value scales, respectively. For example, suppose $y_{k1}, y_{k2}, \ldots, y_{kn_k}$ are n_k observations of a response variable Y in a data set from a normal distribution with an unknown mean μ and a known variance σ^2. Then

$$y_{kj} \sim N\left(\mu,\ \sigma^2\right)$$

for $k = 1, 2, \ldots, K$, where K is the number of groups and n_k is the number of observations at group k.

If N_k is the cumulative number of observations for the first k groups, then the sample mean from these N_k observations

$$\overline{y}_k = \frac{1}{N_k}\sum_{j=1}^{N_k} y_{kj}$$

has a normal distribution with mean θ and variance σ^2/N_k:

$$\overline{y}_k \sim N\left(\theta,\ \frac{\sigma^2}{N_k}\right)$$

To test the null hypothesis $\mu = \mu_0$, $H_0 : \theta = 0$, where $\theta = \mu - \mu_0$ can be used. The MLE of θ is $\hat{\theta}_k = \overline{y}_k - \mu_0$ and

$$\hat{\theta}_k \sim N\left(\theta, \frac{1}{I_k}\right)$$

where the information is the inverse of the variance of $\overline{y}_k$,

$$I_k = \frac{N_k}{\sigma^2}$$

The corresponding standardized Z statistic is

$$Z_k = \hat{\theta}_k \, I_k^{\frac{1}{2}} \sim N\left(\theta \, I_k^{\frac{1}{2}}, 1\right)$$

The score statistic in the SEQDESIGN procedure is then given by

$$S_k = \hat{\theta}_k \, I_k = Z_k \, I_k^{\frac{1}{2}} \sim N\left(\theta \, I_k, I_k\right)$$

For a null hypothesis $H_0 : \theta = 0$ with an upper alternative, the nominal p-value of the standardized Z statistic is $p_k = 1 - \Phi(Z_k)$. For a null hypothesis $H_0 : \theta = 0$ with a lower alternative or a two-sided alternative, the nominal p-value of the standardized Z statistic is $p_k = \Phi(Z_k)$.

Boundary Variables

The boundaries created in group sequential trials depend on the type of the alternative hypothesis and the early stopping criterion. Table 110.5 shows the boundaries created with various design specifications.

Table 110.5 Boundary Variables

Specifications		Boundary Variables			
Alternative		**Lower**		**Upper**	
Hypothesis	**Early Stopping**	**Alpha**	**Beta**	**Beta**	**Alpha**
Lower	Accept H_0		X		
	Reject H_0	X			
	Accept/Reject H_0	X	X		
Upper	Accept H_0			X	
	Reject H_0				X
	Accept/Reject H_0			X	X
Two-sided	Accept H_0		X	X	
	Reject H_0	X			X
	Accept/Reject H_0	X	X	X	X

Up to four boundaries can be generated in a group sequential design:

- the upper α boundary, to reject the null hypothesis for the upper alternative
- the upper β boundary, to accept the null hypothesis with an upper alternative
- the lower β boundary, to accept the null hypothesis with a lower alternative
- the lower α boundary, to reject the null hypothesis for the lower alternative

For a two-sided design, the null hypothesis is accepted only if both the hypothesis is accepted with an upper alternative and the hypothesis is accepted with a lower alternative.

For a one-sided design with a lower alternative, only the lower boundaries are created. Similarly, for a one-sided design with an upper alternative, only the upper boundaries are created. For example, Figure 110.10 shows the boundary plot for a one-sided test with an upper alternative.

Figure 110.10 Boundary Plot for One-Sided Test

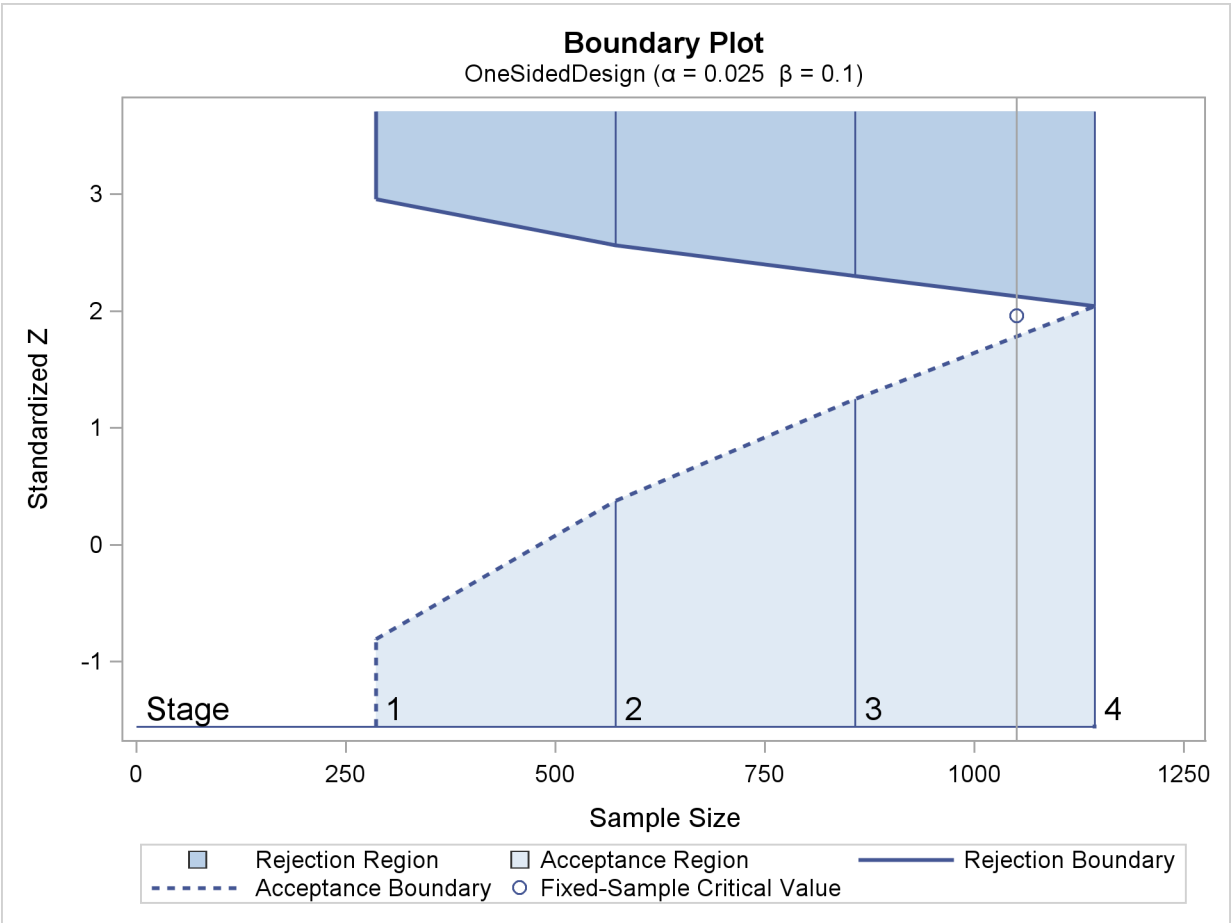

Figure 110.10 corresponds to a one-sided sequential design with early stopping to reject or accept the null hypothesis. For a sequential test with early stopping only to reject the null hypothesis, there are no acceptance boundary values at interim stages. The acceptance boundary value and its associated acceptance region are displayed only at the final stage. Similarly, for a sequential test with early stopping only to accept the null hypothesis, there are no rejection boundary values at interim stages. The rejection boundary value and its associated rejection region are displayed only at the final stage.

For a two-sided design, both the lower and upper boundaries are created. For a design with early stopping to reject the null hypothesis, α boundaries are created. Similarly, for a design with early stopping to accept the null hypothesis, β boundaries are created. For a design with early stopping to accept or reject the null hypothesis, both the α and β boundaries are created.

For example, Figure 110.11 shows the boundary plot for a two-sided test.

Figure 110.11 Boundary Plot for Two-Sided Test

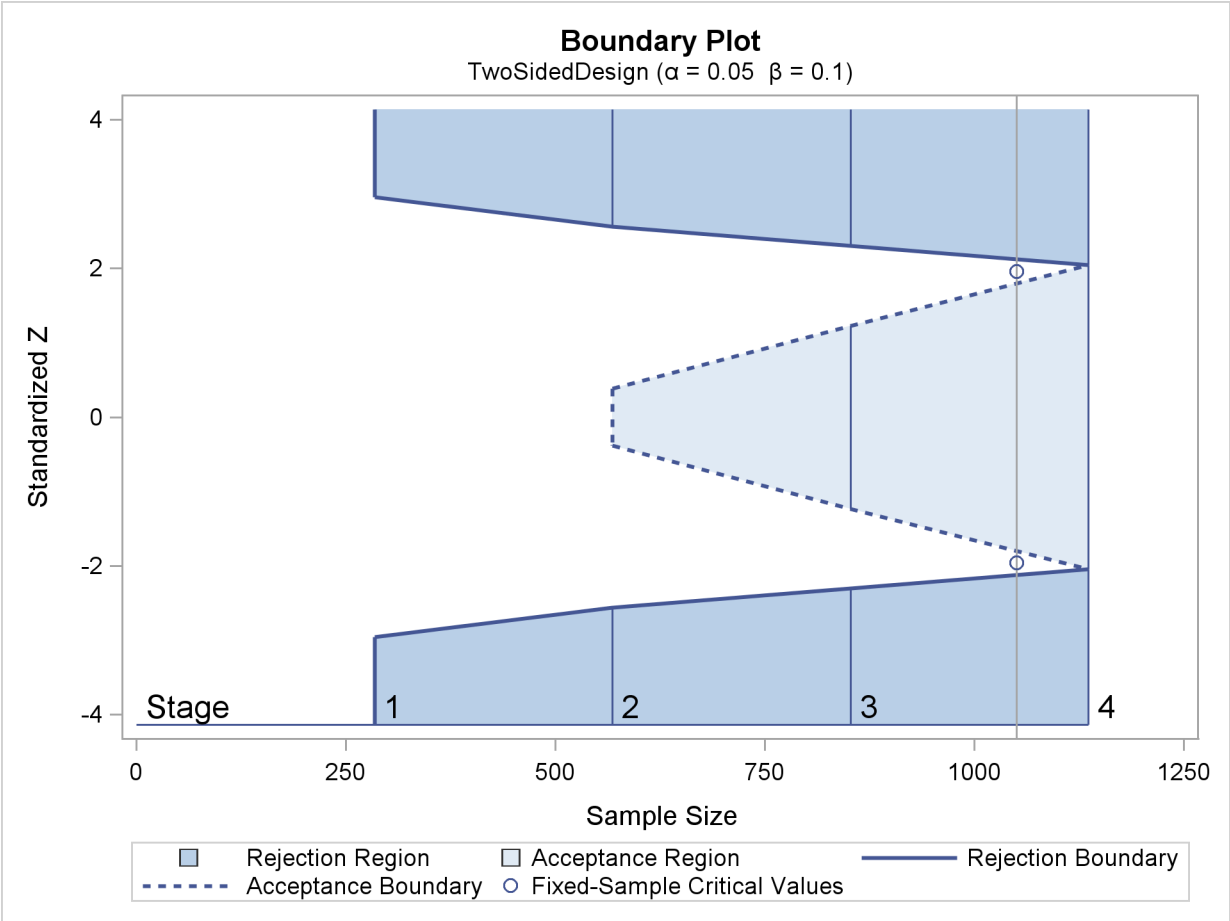

Figure 110.11 corresponds to a two-sided sequential design with early stopping to reject or accept the null hypothesis. For a sequential test with early stopping only to reject the null hypothesis, there are no acceptance boundary values at interim stages. The acceptance boundary value and its associated acceptance region are displayed only at the final stage. Similarly, for a sequential test with early stopping only to accept the null hypothesis, there are no rejection boundary values at interim stages. The rejection boundary value and its associated rejection region are displayed only at the final stage.

Type I and Type II Errors

The Type I error is the error of rejecting the null hypothesis when the null hypothesis is correct, and the Type II error is the error of not rejecting the null hypothesis when the null hypothesis is incorrect. The level of significance α is the probability of making a Type I error. The Type II error depends on the hypothetical reference of the alternative hypothesis, and the Type II error probability β is defined as the probability of not rejecting the null hypothesis when a specific alternative reference is true. The power $1 - \beta$ is then defined as the probability of rejecting the null hypothesis at the alternative reference.

In a sequential design, if the maximum information and alternative reference are not both specified, the critical values are created such that both the specified Type I and the specified Type II error probability levels are maintained in the design. Otherwise, the critical values are created such that either the specified Type I error probability or the specified Type II error probability is maintained.

One-Sided Tests

For a K-stage group sequential design with an upper alternative hypothesis $H_1 : \theta = \theta_1$ and early stopping to reject or accept the null hypothesis $H_0 : \theta = 0$, the boundaries contain the upper α critical values a_k and upper β critical values b_k, $k = 1, 2, \ldots, K$. At each interim stage, $b_k < a_k$, the null hypothesis H_0 is rejected if the observed statistic $z_k \geq a_k$, H_0 is accepted if $z_k < b_k$, or the process is continued to the next stage if $b_k \leq z_k < a_k$. At the final stage $b_K = a_K$, the hypothesis is either rejected or accepted.

The overall Type I error probability α is given by

$$\alpha = \sum_{k=1}^{K} \alpha_k$$

where α_k is the α spending at stage k. That is, at stage 1,

$$\alpha_1 = P_{\theta=0}(z_1 \geq a_1)$$

At a subsequent stage k,

$$\alpha_k = P_{\theta=0}(b_j \leq z_j < a_j, j = 1, 2, \ldots, k-1, z_k \geq a_k)$$

Similarly, the Type II error probability

$$\beta = \sum_{k=1}^{K} \beta_k$$

where β_k is the β spending at stage k. That is, at stage 1,

$$\beta_1 = P_{\theta=\theta_1}(z_1 < b_1)$$

At a subsequent stage k,

$$\beta_k = P_{\theta=\theta_1}(b_j \leq z_j < a_j, j = 1, 2, \ldots, k-1, z_k < b_k)$$

With an upper alternative hypothesis $H_1 : \theta = \theta_1 > 0$, the power $1 - \beta$ is the probability of rejecting the null hypothesis for the upper alternative.

$$1 - \beta = 1 - \sum_{k=1}^{K} \beta_k = \sum_{k=1}^{K} P_{\theta=\theta_1}(\, b_j \le z_j < a_j, j = 1, 2, \ldots, k-1,\ z_k \ge a_k)$$

For a design with early stopping to reject H_0 only, the interim upper β critical values are set to $-\infty$, $b_k = -\infty, k = 1, 2, \ldots, K-1$, and $\beta = \beta_K$. For a design with early stopping to accept H_0 only, the interim upper α critical values are set to ∞, $a_k = \infty, k = 1, 2, \ldots, K-1$, and $\alpha = \alpha_K$.

Similarly, the Type I and Type II error probabilities for a K-stage design with a lower alternative hypothesis $H_0 : \theta = -\theta_1$ can also be derived.

Two-Sided Tests

For a K-stage group sequential design with two-sided alternative hypotheses $H_{1u} : \theta = \theta_{1u}$ and $H_{1l} : \theta = \theta_{1l}$, and early stopping to reject or accept the null hypothesis $H_0 : \theta = 0$, the boundaries contain the upper α critical values a_k, upper β critical values b_k, lower β critical values $_b_k$, and lower α critical values $_a_k$, $k = 1, 2, \ldots, K$. At each interim stage, $_a_k < _b_k \le b_k < a_k$, the null hypothesis H_0 is rejected if the observed statistic $z_k \ge a_k$ or $z_k \le _a_k$, H_0 is accepted if $_b_k < z_k < b_k$, or the process is continued to the next stage if $b_k \le z_k < a_k$ or $_a_k < z_k \le _b_k$. At the final stage $b_K = a_K$ and $_b_K = _a_K$, the hypothesis is either rejected or accepted.

The overall upper Type I error probability α_u is given by

$$\alpha_u = \sum_{k=1}^{K} \alpha_{uk}$$

where α_{uk} is the α spending at stage k for the upper alternative. That is, at stage 1,

$$\alpha_{u1} = P_{\theta=0}(\, z_1 \ge a_1 \,)$$

At a subsequent stage k,

$$\alpha_{uk} = P_{\theta=0}(\, _a_j < z_j \le _b_j \text{ or } b_j \le z_j < a_j, j = 1, 2, \ldots, k-1,\ z_k \ge a_k \,)$$

Similarly, the overall lower Type I error probability α_l can also be derived, and the overall Type I error probability $\alpha = \alpha_l + \alpha_u$.

The overall upper Type II error probability β_u is given by

$$\beta_u = \sum_{k=1}^{K} \beta_{uk}$$

where β_{uk} is the upper β spending at stage k. That is, at stage 1,

$$\beta_{u1} = P_{\theta=\theta_{1u}}(\, z_1 < _a_1 \text{ or } _b_1 < z_1 < b_1 \,)$$

At a subsequent stage k,

$$\beta_{uk} = P_{\theta=\theta_{1u}}(_a_j < z_j \le _b_j \text{ or } b_j \le z_j < a_j,\ j = 1, 2, \ldots, k-1,\ z_k < _a_k \text{ or } _b_k < z_k < b_k)$$

With an upper alternative hypothesis $H_1 : \theta = \theta_{1u} > 0$, the power $1 - \beta_u$ is the probability of rejecting the null hypothesis for the upper alternative:

$$1 - \beta_u = 1 - \sum_{k=1}^{K} \beta_{uk}$$

which is

$$P_{\theta=\theta_{1u}}(_a_j < z_j \le _b_j \text{ or } b_j \le z_j < a_j,\ j = 1, 2, \ldots, k-1,\ z_k \ge a_k)$$

The overall lower Type II error probability β_l and power $1 - \beta_l$ can be similarly derived.

For a design with early stopping only to reject H_0, both the interim lower and upper β critical values are set to missing, $k = 1, 2, \ldots, K-1$, and $\beta_{lK} = \beta_l$, $\beta_{uK} = \beta_u$. For a design with early stopping only to accept H_0, the interim upper α critical values are set to ∞, $a_{uk} = \infty$, and the interim lower α critical values are set to $-\infty$, $a_{lk} = -\infty$, $k = 1, 2, \ldots, K-1$, and $\alpha_{uK} = \alpha_u$, $\alpha_{lK} = \alpha_l$.

Unified Family Methods

Unified family methods (Kittelson and Emerson 1999) derive boundary values with a specified boundary shape. For example, Pocock's method (Pocock 1977) derives equal boundary values for all stages in the standardized Z scale. In addition to Pocock's method, the unified family methods include the O'Brien-Fleming, power family, and unified family triangular methods.

The boundary values at each stage depend on the information fractions

$$\Pi_k = \frac{I_k}{I_X}$$

where I_k is the information available at stage k and I_X is the maximum information, the information available at the end of the trial if the trial does not stop early.

Boundary Values in Standardized Z Scale

With the unified family method, the boundary values for the upper α boundary $Z_{\alpha u}$, upper β boundary $Z_{\beta u}$, lower β boundary $Z_{\beta l}$, and lower α boundary $Z_{\alpha l}$, using the standardized normal scale, are given by the following:

- $Z_{\alpha u}(\Pi_k) = f_{\alpha u}(\Pi_k)\, C_{\alpha u}$
- $Z_{\beta u}(\Pi_k) = \theta_{1u}\, I_k^{\frac{1}{2}} - f_{\beta u}(\Pi_k)\, C_{\beta u}$
- $Z_{\beta l}(\Pi_k) = \theta_{1l}\, I_k^{\frac{1}{2}} + f_{\beta l}(\Pi_k)\, C_{\beta l}$

- $Z_{\alpha l}(\Pi_k) = -f_{\alpha l}(\Pi_k)\, C_{\alpha l}$

where $\theta_{1l}(< 0)$ and $\theta_{1u}(> 0)$ are the lower and upper alternative references, $f_{\alpha l}(\Pi_k)$, $f_{\beta l}(\Pi_k)$, $f_{\beta u}(\Pi_k)$, and $f_{\alpha u}(\Pi_k)$ are the specified shape functions, and $C_{\alpha l}$, $C_{\beta l}$, $C_{\beta u}$, and $C_{\alpha u}$ are the critical values derived to achieve the specified α and β levels.

If a derived lower β boundary value $Z_{\beta l}(\Pi_k)$ is greater than its corresponding upper β boundary value $Z_{\beta u}(\Pi_k)$, then both values are set to missing.

Note that the drift parameters $d_l = \theta_{1l}\sqrt{I_X}$ and $d_u = \theta_{1u}\sqrt{I_X}$ are derived in the SEQDESIGN procedure. The boundary values in standardized Z scale can be derived without specifying the maximum information and alternative reference.

Shape Parameters

The shape function in the SEQDESIGN procedure is given by

$$f(\Pi_k) = f(\Pi_k; \tau, \rho) = \tau\, \Pi_k^{\frac{1}{2}} + \Pi_k^{-\rho} = \Pi_k^{\frac{1}{2}}\, (\tau + \Pi_k^{-(\rho+\frac{1}{2})})$$

where the parameters $\rho \geq 0$ and $0 \leq \tau \leq 2\rho$. can be specified for each boundary separately.

The parameters τ and ρ determine the shape of the boundaries. Special cases of the unified family methods also include power family methods and triangular methods. Table 110.6 summarizes the corresponding parameter values in the unified family for these methods.

Table 110.6 Parameters in the Unified Family for Various Methods

		Unified Family	
Method	**Option**	**Rho**	**Tau**
Pocock	POC	0	0
O'Brien-Fleming	OBF	0.5	0
Power family	POW (RHO=ρ)	ρ	0
Triangular	TRI (TAU=τ)	0.5	τ

Note that the power parameter $\rho = 1/2 - \Delta = \rho^* - 1/2$, where Δ is the power parameter used in Jennison and Turnbull (2000) and Wang and Tsiatis (1987) and ρ^* is the power parameter used in Kittelson and Emerson (1999).

Also note that instead of the three parameters used in the unified family methods by Kittelson and Emerson (1999), only two parameters are used in the SEQDESIGN procedure. The other parameter is fixed at zero.

Boundary Values in MLE Scale

If the maximum information is available, the boundary values derived from a unified family method can also be displayed in the MLE scale:

- $\theta_{\alpha u}(\Pi_k) = I_k^{-\frac{1}{2}} f_{\alpha u}(\Pi_k) C_{\alpha u}$
- $\theta_{\beta u}(\Pi_k) = \theta_{1u} - I_k^{-\frac{1}{2}} f_{\beta u}(\Pi_k) C_{\beta u}$
- $\theta_{\beta l}(\Pi_k) = \theta_{1l} + I_k^{-\frac{1}{2}} f_{\beta l}(\Pi_k) C_{\beta l}$
- $\theta_{\alpha l}(\Pi_k) = -I_k^{-\frac{1}{2}} f_{\alpha l}(\Pi_k) C_{\alpha l}$

These MLE scale boundary values are computed by multiplying $I_k^{-\frac{1}{2}}$ by the standardized Z scale boundary values at stage k.

Boundary Values in Score Scale

If the maximum information is available, the boundary values derived from a unified family method can also be displayed in the score scale:

- $S_{\alpha u}(\Pi_k) = I_k^{\frac{1}{2}} f_{\alpha u}(\Pi_k) C_{\alpha u}$
- $S_{\beta u}(\Pi_k) = \theta_{1u} I_k - I_k^{\frac{1}{2}} f_{\beta u}(\Pi_k) C_{\beta u}$
- $S_{\beta l}(\Pi_k) = \theta_{1l} I_k + I_k^{\frac{1}{2}} f_{\beta l}(\Pi_k) C_{\beta l}$
- $S_{\alpha l}(\Pi_k) = -I_k^{\frac{1}{2}} f_{\alpha l}(\Pi_k) C_{\alpha l}$

These MLE scale boundary values are computed by multiplying $I_k^{\frac{1}{2}}$ by the standardized Z scale boundary values at stage k.

Boundary Values in *p*-Value Scale

For a design with a lower alternative or a two-sided alternative, the p-value scale boundary values are the cumulative normal distribution function values of the standardized Z boundary values:

- $P_{\alpha u}(\Pi_k) = \Phi(Z_{\alpha u}(\Pi_k))$
- $P_{\beta u}(\Pi_k) = \Phi(Z_{\beta u}(\Pi_k))$
- $P_{\beta l}(\Pi_k) = \Phi(Z_{\beta l}(\Pi_k))$
- $P_{\alpha l}(\Pi_k) = \Phi(Z_{\alpha l}(\Pi_k))$

These nominal p-values are the one-sided fixed-sample p-values under the null hypothesis with a lower alternative.

For a one-sided design with an upper alternative, the p-value scale boundary values are the one-sided fixed-sample p-values under the null hypothesis with an upper alternative:

- $P_{\alpha u}(\Pi_k) = 1 - \Phi(Z_{\alpha u}(\Pi_k))$
- $P_{\beta u}(\Pi_k) = 1 - \Phi(Z_{\beta u}(\Pi_k))$

Pocock's Method

The shape function for Pocock's method (Pocock 1977) is given by

$$f(\Pi_k) = 1$$

The resulting boundary values for a two-sided design with an early stopping to reject the null hypothesis $H_0 : \theta = 0$ are as follows:

- $Z_{\alpha u}(\Pi_k) = C_{\alpha u}$
- $Z_{\alpha l}(\Pi_k) = -C_{\alpha l}$

That is, the rejection boundary values are constant over all stages of different information levels in the standardized Z scale.

Note that compared with other designs, Pocock's design tends to stop the trials early with a larger p-value. For a new treatment, Pocock's design to stop a trial early with a large p-value might not be persuasive enough to make a new treatment widely accepted (Pocock and White 1999). A Pocock design is illustrated in Example 110.3.

O'Brien-Fleming Method

The shape function for the O'Brien-Fleming method (O'Brien and Fleming 1979) is given by

$$f(\Pi_k) = \Pi_k^{-\frac{1}{2}}$$

The resulting boundary values for a two-sided design with early stopping to reject the null hypothesis $H_0 : \theta = 0$ are as follows:

- $Z_{\alpha u}(\Pi_k) = \Pi_k^{-\frac{1}{2}} C_{\alpha u}$
- $Z_{\alpha l}(\Pi_k) = -\Pi_k^{-\frac{1}{2}} C_{\alpha l}$

That is, the rejection boundaries are inversely proportional to the square root of the information levels in the standardized Z scale.

In the score scale, these boundaries can be displayed as follows:

- $S_{\alpha u}(\Pi_k) = C_{\alpha u}\, I_X^{\frac{1}{2}}$
- $S_{\alpha l}(\Pi_k) = -C_{\alpha l}\, I_X^{\frac{1}{2}}$

which are constants over all stages in the score scale. An O'Brien-Fleming design is illustrated in Example 110.2.

Power Family Method

The shape function for a power family method (Wang and Tsiatis 1987; Emerson and Fleming 1989; Pampallona and Tsiatis 1994) is given by

$$f(\Pi_k) = \Pi_k^{-\rho}$$

The resulting boundary values for a two-sided design with early stopping to reject the null hypothesis $H_0 : \theta = 0$ are as follows:

- $Z_{\alpha u}(\Pi_k) = \Pi_k^{-\rho}\, C_{\alpha u}$
- $Z_{\alpha l}(\Pi_k) = -\Pi_k^{-\rho}\, C_{\alpha l}$

The rejection boundaries depend on the power parameter ρ. The power family includes the Pocock and O'Brien-Fleming methods, and the power parameter is used to allow continuous movement between these two methods.

Triangular Method

The shape function for a triangular method (Kittelson and Emerson 1999) in the unified family is given by

$$f(\Pi_k) = \Pi_k^{-\frac{1}{2}} + \tau\, \Pi_k^{\frac{1}{2}}$$

The resulting boundary values for a two-sided design with early stopping to reject the null hypothesis $H_0 : \theta = 0$ are as follows:

- $Z_{\alpha u}(\Pi_k) = (\Pi_k^{-\frac{1}{2}} + \tau\, \Pi_k^{\frac{1}{2}})\, C_{\alpha u} = C_{\alpha u}\, \Pi_k^{-\frac{1}{2}}\, (1 + \tau\, \Pi_k)$
- $Z_{\alpha l}(\Pi_k) = -(\Pi_k^{-\frac{1}{2}} + \tau\, \Pi_k^{\frac{1}{2}})\, C_{\alpha l} = -C_{\alpha l}\, \Pi_k^{-\frac{1}{2}}\, (1 + \tau\, \Pi_k)$

In the score scale, these boundaries are as follows:

- $S_{\alpha u}(\Pi_k) = C_{\alpha u}\, I_X^{\frac{1}{2}}\, (1 + \tau\, \Pi_k) = C_{\alpha u}\, I_X^{\frac{1}{2}} + C_{\alpha u} \tau I_X^{-\frac{1}{2}}\, I_k$
- $S_{\alpha l}(\Pi_k) = -C_{\alpha l}\, I_X^{\frac{1}{2}}\, (1 + \tau\, \Pi_k) = -C_{\alpha l}\, I_X^{\frac{1}{2}} - C_{\alpha l} \tau I_X^{-\frac{1}{2}}\, I_k$

Thus, in the score scale, the boundary function is a linear function of the information I_k. With these straight-line boundaries, a triangular method for a one-sided trial with early stopping to reject or accept the null hypothesis produces a triangular continuation region. Similarly, for a two-sided design, the continuation region is a union of two separate triangular regions. A triangular method is illustrated in Example 110.6.

Haybittle-Peto Method

The Haybittle-Peto method (Haybittle 1971; Peto et al. 1976) uses a value of 3 for the critical values in interim stages, so that the critical value at the final stage is close to the original design without interim monitoring.

In the SEQDESIGN procedure, the Haybittle-Peto method has been generalized to allow for different boundary values at different stages. That is, with the standardized normal scale, the boundary values are given by the following:

- $Z_{\alpha u}(\Pi_k) = z_{\alpha uk}$
- $Z_{\beta u}(\Pi_k) = \theta_{1u}\, I_k^{\frac{1}{2}} - z_{\beta uk}$
- $Z_{\beta l}(\Pi_k) = \theta_{1l}\, I_k^{\frac{1}{2}} + z_{\beta lk}$
- $Z_{\alpha l}(\Pi_k) = -z_{\alpha lk}$

where θ_{1l} and θ_{1u} are the lower and upper alternative references and the boundary values $z_{\alpha uk}$, $z_{\beta uk}$, $z_{\beta lk}$, and $z_{\alpha lk}$ are specified either explicitly with the HP(Z= *numbers*) option or implicitly with the HP(PVALUE= *numbers*) option. The HP(PVALUE= *numbers*) option specifies the nominal p-values p_k for the corresponding boundary values z_k:

$$z_k = \Phi^{-1}(1 - p_k)$$

The Haybittle-Peto method is illustrated in Example 110.5.

Whitehead Methods

The Whitehead methods (Whitehead and Stratton 1983; Whitehead 1997, 2001) derive boundary values by adjusting the boundary values generated from continuous monitoring. With continuous monitoring, the boundary values are on a straight line in the score scale for each boundary. For a group sequential design, the boundary values at an interim stage k depend on the information fractions

$$\Pi_k = \frac{I_k}{I_X}$$

where I_k is the information available at stage k and I_X is the maximum information, the information available at the end of the trial if the trial does not stop early.

One-Sided Symmetric Designs

A one-sided symmetric design is a one-sided design with identical Type I and Type II error probabilities. For a one-sided symmetric design with an upper alternative, $\alpha_u = \beta_u$, the boundary values in the score scale from continuous monitoring are as follows:

- $S_{\alpha u}(\Pi_k) = C_u \theta_u^{-1} + \tau_u \theta_u I_k$
- $S_{\beta u}(\Pi_k) = \theta_u\, I_k - (C_u \theta_u^{-1} - \tau_u \theta_u I_k)$

where θ_u is the upper alternative reference, τ_u is a specified constant for the slope, $0 \le \tau_u < \frac{1}{2}$, and C_u is a constant, fixed for STOP=BOTH and derived for STOP=ACCEPT and STOP=REJECT.

The upper β boundary value can also be expressed as

- $S_{\beta u}(\Pi_k) = -C_u \theta_u^{-1} + (1 - \tau_u)\theta_u I_k$

Thus, these straight-line boundaries form a triangle in the score statistic scale.

To adjust for the nature of discrete monitoring, the group sequential boundary values are given by the following:

- $S_{\alpha u}(\Pi_k) = C_u \theta_u^{-1} + \tau_u \theta_u I_k - g_k$
- $S_{\beta u}(\Pi_k) = -C_u \theta_u^{-1} + (1 - \tau_u)\theta_u I_k + g_k$

where $g_1 = 0.583\sqrt{I_1}$ and $g_k = 0.583\sqrt{I_k - I_{(k-1)}}$, $k > 1$ are the adjustments.

Note that with the adjustment g_k, the resulting boundaries form a Christmas tree shape within the original triangle and are referred to as the Christmas tree boundaries (Whitehead 1997, p. 73).

One-Sided Asymmetric Designs

For a one-sided asymmetric design with an upper alternative, $\alpha_u \neq \beta_u$, the boundary values computed using the score scale, are given by the following:

- $S_{\alpha u}(\Pi_k) = C_u \tilde{\theta}_u^{-1} + \tau_u \tilde{\theta}_u I_k - g_k$
- $S_{\beta u}(\Pi_k) = -C_u \tilde{\theta}_u^{-1} + (1 - \tau_u)\tilde{\theta}_u I_k + g_k$

where $\tilde{\theta}_u$ is the modified alternative reference

$$\tilde{\theta}_u = \frac{2\Phi^{-1}(1 - \alpha_u)}{\Phi^{-1}(1 - \alpha_u) + \Phi^{-1}(1 - \beta_u)}\,\theta_u$$

The modified alternative reference $\tilde{\theta}_u = \theta_u$ if $\alpha_u = \beta_u$.

For a design with early stopping to reject or accept the null hypothesis, $S_{\alpha u}(1) = S_{\beta u}(1)$, the boundary values at the final stage are equal. The modified drift parameter $\tilde{d}_u$ is given by

$$\tilde{d}_u = \tilde{\theta}_u \sqrt{I_X} = \frac{1}{1-2\tau_u}\left(\sqrt{{h_K}^2 + 2C_u(1-2\tau_u)} - h_K\right)$$

where $h_K = g_K\, I_X^{-\frac{1}{2}} = 0.583\sqrt{1-\Pi_{(K-1)}}$.

A one-sided Whitehead design with early stopping to reject or accept the null hypothesis is illustrated in Example 110.7.

Two-Sided Designs

The boundary values for a two-sided design are generated by combining boundary values from two one-sided designs. With the STOP=BOTH option, this produces a double triangular design (Whitehead 1997, p. 98).

The boundary values for a two-sided design, using the score scale, are then given by the following:

- $S_{\alpha u}(\Pi_k) = C_u\tilde{\theta}_u^{-1} + \tau_u\tilde{\theta}_u I_k - g_k$
- $S_{\beta u}(\Pi_k) = -C_u\tilde{\theta}_u^{-1} + (1-\tau_u)\tilde{\theta}_u I_k + g_k$
- $S_{\beta l}(\Pi_k) = -C_l\tilde{\theta}_l^{-1} + (1-\tau_l)\tilde{\theta}_l I_k - g_k$
- $S_{\alpha l}(\Pi_k) = C_l\tilde{\theta}_l^{-1} + \tau_l\tilde{\theta}_l I_k + g_k$

where the modified alternative references are

$$\tilde{\theta}_u = \frac{2\Phi^{-1}(1-\alpha_u)}{\Phi^{-1}(1-\alpha_u) + \Phi^{-1}(1-\beta_u)}\,\theta_u$$

$$\tilde{\theta}_l = \frac{2\Phi^{-1}(1-\alpha_l)}{\Phi^{-1}(1-\alpha_l) + \Phi^{-1}(1-\beta_l)}\,\theta_l$$

The modified alternative reference $\tilde{\theta}_u = \theta_u$ if $\alpha_u = \beta_u$ and $\tilde{\theta}_l = \theta_l$ if $\alpha_l = \beta_l$.

For a design with early stopping to reject or accept the null hypothesis, the two upper boundary values at the final stage are identical and the two lower boundary values at the final stage are identical. That is, $S_{\alpha l}(1) = S_{\beta l}(1)$ and $S_{\alpha u}(1) = S_{\beta u}(1)$. These modified drift parameters are then given by

$$\tilde{d}_l = \tilde{\theta}_l \sqrt{I_X} = \frac{1}{1-2\tau_l}\left(\sqrt{{h_K}^2 + 2C_l(1-2\tau_l)} - h_K\right)$$

$$\tilde{d}_u = \tilde{\theta}_u \sqrt{I_X} = \frac{1}{1-2\tau_u}\left(\sqrt{{h_K}^2 + 2C_u(1-2\tau_u)} - h_K\right)$$

where $h_K = g_K I_X^{-\frac{1}{2}} = 0.583\sqrt{1-\Pi_{(K-1)}}$.

For a design with early stopping to reject the null hypothesis, or a design with early stopping to accept the null hypothesis, you can specify the slope parameters τ_u and τ_l in the TAU= option, and then the intercept parameters C_u and C_l, and the resulting boundary values are derived. If both the maximum information

and alternative references are specified, the procedure derives C_u and C_l by maintaining either the overall α levels (BOUNDARYKEY=ALPHA) or the overall β levels (BOUNDARYKEY=BETA). If the maximum information and alternative reference are not both specified, the procedure derives the boundary values C_u and C_l by maintaining both the overall α and overall β levels.

For a design with early stopping to reject or accept the null hypothesis (STOP=BOTH), Whitehead's triangular test uses $\tau_u = \tau_l = 0.25$ and compute $C_u = -2\log(2\alpha_u)$ and $C_l = -2\log(2\alpha_l)$ for the boundary values. If the maximum information and alternative reference are both specified, the BOUNDARYKEY=ALPHA option uses the specified α values to compute the β values and boundary values. The final-stage boundary values are modified to maintain the overall α levels if they exist. Similarly, the BOUNDARYKEY=BETA option uses the specified β values to compute the α values and boundary values. The final-stage boundary values are modified to maintain the overall β levels if they exist.

If the maximum information and alternative reference are not both specified, the specified α and β values are used to derive boundary values. The BOUNDARYKEY=NONE option uses these boundary values without adjustment. The BOUNDARYKEY=ALPHA option modifies the final-stage boundary values to maintain the overall α levels if they exist. Similarly, the BOUNDARYKEY=BETA option modifies the final-stage boundary values to maintain the overall β levels if they exist.

Applicable Boundary Keys

Table 110.7 lists applicable boundary keys for a design that uses Whitehead methods.

Table 110.7 Applicable Boundary Keys for Whitehead Methods

	Specified Parameters		Boundary Keys			
Early Stopping	**(Alt Ref – Max Info)**	**Tau**	**Alpha**	**Beta**	**None**	**Both**
Reject H_0	X	X	X	X		
Accept H_0	X	X	X	X		
Reject/Accept H_0	X	0.25	X	X		
Reject H_0		X				X
Accept H_0		X				X
Reject/Accept H_0		0.25	X	X	X	

Note that the symbol "X" under "(Alt Ref – Max Info)" indicates that both alternative reference and maximum information are specified.

For a design with early stopping to reject the null hypothesis, or a design with early stopping to accept the null hypothesis, you can specify the slope parameter τ_u in the TAU= option, and then the intercept parameter C_u and the resulting boundary values are derived. If both the maximum information and alternative reference are specified, the procedure derives C_u by maintaining either the overall α levels (BOUNDARYKEY=ALPHA) or the overall β levels (BOUNDARYKEY=BETA). If the maximum information and alternative reference are not both specified, the procedure derives the boundary values and C_u by maintaining both the overall α and overall β levels.

For a design with early stopping to reject or accept the null hypothesis (STOP=BOTH), Whitehead's triangular test uses $\tau_u = 0.25$ and solves $C_u = 2\log(\frac{1}{2\alpha_u})$ for the boundary values. If the maximum information and

alternative reference are both specified, the BOUNDARYKEY=ALPHA option uses the specified α value to compute the β value and boundary values. The final-stage boundary value is modified to maintain the overall α level if it exists. Similarly, the BOUNDARYKEY=BETA option uses the specified β value to compute the α value and boundary values. The final-stage boundary value is modified to maintain the overall β level if it exists.

If the maximum information and alternative reference are not both specified, the specified α and β values are used to derive boundary values. The BOUNDARYKEY=NONE option uses these boundary values without adjustment. The BOUNDARYKEY=ALPHA option modifies the final-stage boundary value to maintain the overall α level if it exists. Similarly, the BOUNDARYKEY=BETA option modifies the final-stage boundary value to maintain the overall β level if it exists.

Error Spending Methods

For each sequential design, the α and β errors spent at each stage can be computed from the boundary values. For example, for a K-stage design with an upper alternative hypothesis $H_1 : \theta = \theta_1$ and early stopping to reject the null hypothesis $H_0 : \theta = 0$, the boundary values in a standardized Z scale are the upper α critical values $a_k, k = 1, 2, \ldots, K$. At each interim stage, the null hypothesis H_0 is rejected if the observed standardized Z statistic $z_k \geq a_k$. Otherwise, the process continues to the next stage. At the final stage, the hypothesis is rejected if $z_K \geq a_K$. Otherwise, the null hypothesis is accepted.

The boundary values a_k are derived such that the overall Type I error probability

$$\alpha = \sum_{k=1}^{K} \alpha_k$$

where α_k is the α spending at stage k. That is, at stage 1,

$$\alpha_1 = P_{\theta=0}\,(\,z_1 \geq a_1\,)$$

At a subsequent stage k,

$$\alpha_k = P_{\theta=0}\,(\,z_j < a_j, j = 1, 2, \ldots, k-1,\ z_k \geq a_k\,)$$

Since each design can be uniquely identified by the α and β errors spent at each stage, a design can then be derived by specifying the α and β errors to be used at each stage. The error spending method (Lan and DeMets 1983) distributes the error to be used at each stage and then derives the boundary values. Numerous forms of the error spending function are available. Kim and DeMets (1987) examine the functions $f(t) = t$, $f(t) = t^{\frac{3}{2}}$, and $f(t) = t^2$, where t is the information fraction. Jennison and Turnbull (2000, p. 148) generalize these functions to the power functions $f(t; \rho) = t^{\rho}, \rho > 0$.

The ERRFUNCPOC option uses the cumulative error spending function (Lan and DeMets 1983)

$$E(t) = \begin{cases} 1 & \text{if } t \geq 1 \\ \log(\,1 + (e-1)t) & \text{if } 0 < t < 1 \\ 0 & \text{otherwise} \end{cases}$$

With a specified error of α or β, the cumulative error spending at stage k is $\alpha\ E(\Pi_k)$ or $\beta\ E(\Pi_k)$, where $\Pi_k = I_k/I_X$ is the information fraction at stage k. The method produces boundaries similar to those produced with Pocock's method.

The ERRFUNCOBF option uses the cumulative error spending function (Lan and DeMets 1983)

$$E(t;a) = \begin{cases} 1 & \text{if } t \geq 1 \\ \frac{1}{a}\,2\,(1-\Phi(\frac{z_{(1-a/2)}}{\sqrt{t}})) & \text{if } 0 < t < 1 \\ 0 & \text{otherwise} \end{cases}$$

where a is either α for the α spending function or β for the β spending function. That is, with a specified error of α or β, the cumulative error spending at stage k is $\alpha\ E(\Pi_k;\alpha)$ or $\beta\ E(\Pi_k;\beta)$. The method produces boundaries similar to those produced with the O'Brien-Fleming method.

The ERRFUNCGAMMA option uses the gamma cumulative error spending function (Hwang, Shih, and DeCani 1990)

$$E(t;\gamma) = \begin{cases} 1 & \text{if } t \geq 1 \\ \frac{1-e^{-\gamma t}}{1-e^{-\gamma}} & \text{if } 0 < t < 1, \gamma \neq 0 \\ t & \text{if } 0 < t < 1, \gamma = 0 \\ 0 & \text{otherwise} \end{cases}$$

where γ is the parameter γ specified in the GAMMA= option. That is, with a specified error of α or β, the cumulative error spending at stage k is $\alpha\ E(\Pi_k;\gamma)$ or $\beta\ E(\Pi_k;\gamma)$.

The ERRFUNCPOW option uses the cumulative error spending function (Jennison and Turnbull 2000, p. 148)

$$E(t;\rho) = \begin{cases} 1 & \text{if } t \geq 1 \\ t^{\rho} & \text{if } 0 < t < 1 \\ 0 & \text{otherwise} \end{cases}$$

where ρ is the power parameter specified in the RHO= option. That is, with a specified error of α or β, the cumulative error spending at stage k is $\alpha\ E(\Pi_k;\rho)$ or $\beta\ E(\Pi_k;\rho)$.

Error spending methods derive boundary values at each stage sequentially and require much more computation than other types of methods for group sequential trials with a large number of stages, especially for a two-sided asymmetric design with early stopping to accept H_0, or to reject or accept H_0.

Note that for a two-sided design with the STOP=BOTH or STOP=ACCEPT option, at each interim stage, the SEQDESIGN procedure first produces the lower and upper β boundary values based on the one-sided β spending. If the lower β boundary value is greater than or equal to its corresponding upper β boundary value, there is no early stopping to accept the null hypothesis at this stage, and the corresponding β spending is distributed proportionally to the remaining stages.

For the error spending functions not available in the SEQDESIGN procedure, you can first compute the corresponding error spending at each stage explicitly, then use the SEQDESIGN procedure with the ERRSPEND= option to specify these errors directly.

For example, if the information levels are equally spaced in a five-stage design, the option ERRFUNCPOW (RHO=2) produces relative cumulative errors of $(1/5)^2$, $(2/5)^2$, $(3/5)^2$, $(4/5)^2$, and 1. This is equivalent to using the option ERRSPEND (1 4 9 16 25).

A one-sided error spending design is illustrated in Example 110.8 and a two-sided asymmetric error spending design is illustrated in Example 110.11.

Acceptance (β) Boundary

In a group sequential trial, the rejection boundary is derived under the null hypothesis H_0 and is used to stop the trial early to reject H_0. Similarly, the acceptance boundary is derived under the alternative hypothesis and is used to stop the trial early to accept H_0. But, for a trial with early stopping either to reject or to accept the null hypothesis, dependency exists between these two boundaries. This section describes the effects of the acceptance boundary on the derivation of the rejection boundary in a group sequential trial.

The following statements create a one-sided four-stage group sequential design with early stopping either to reject or to accept H_0:

```
ods graphics on;
proc seqdesign altref=10;
   ErrSpendPower_2: design nstages=4
                    method=errfuncpow(rho=2)
                    alt=upper   stop=both
                    alpha=0.025 beta=0.10;
run;
```

The ALTREF=10 option specifies the alternative reference 10. The METHOD=ERRFUNCPOW(RHO=2) option uses a $\rho = 2$ power family error spending method to generate the rejection boundary. The ALPHA=0.025 and BETA=0.10 options specify the Type I error level 0.025 and Type II error level 0.10, respectively.

The power parameter $\rho = 2$ used in the design lies between $\rho = 1$ and $\rho = 3$, where the boundaries created with the $\rho = 1$ power family error spending method are similar to the boundaries created from the Pocock method, and the boundaries created with $\rho = 3$ are similar to the boundaries created from the O'Brien-Fleming method.

The "Boundary Information" table in Figure 110.12 shows the rejection and acceptance boundary values at the stages. With an error spending function method, the boundary values are derived sequentially. In particular, the rejection boundary value at a stage is derived conditionally on both rejection and acceptance boundary values at the previous stage. See the section "Error Spending Methods" on page 9177 for a detailed description of the error spending methods.

Figure 110.12 Boundary Information

The SEQDESIGN Procedure
Design: ErrSpendPower_2

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference	Boundary Values	
				Upper	
Stage	Proportion	Actual	Upper	Beta	Alpha
1	0.2500	0.028605	1.69130	-0.80640	2.95517
2	0.5000	0.05721	2.39186	0.37356	2.55934
3	0.7500	0.085815	2.92942	1.24940	2.29904
4	1.0000	0.11442	3.38261	2.04182	2.04182

With ODS Graphics enabled, the "Boundary Plot" is displayed by default, as shown in Figure 110.13. The continuation region is affected by the acceptance boundary, and the rejection boundary is thus adjusted for

this acceptance boundary to maintain the Type I error level.

Figure 110.13 Boundary Plot

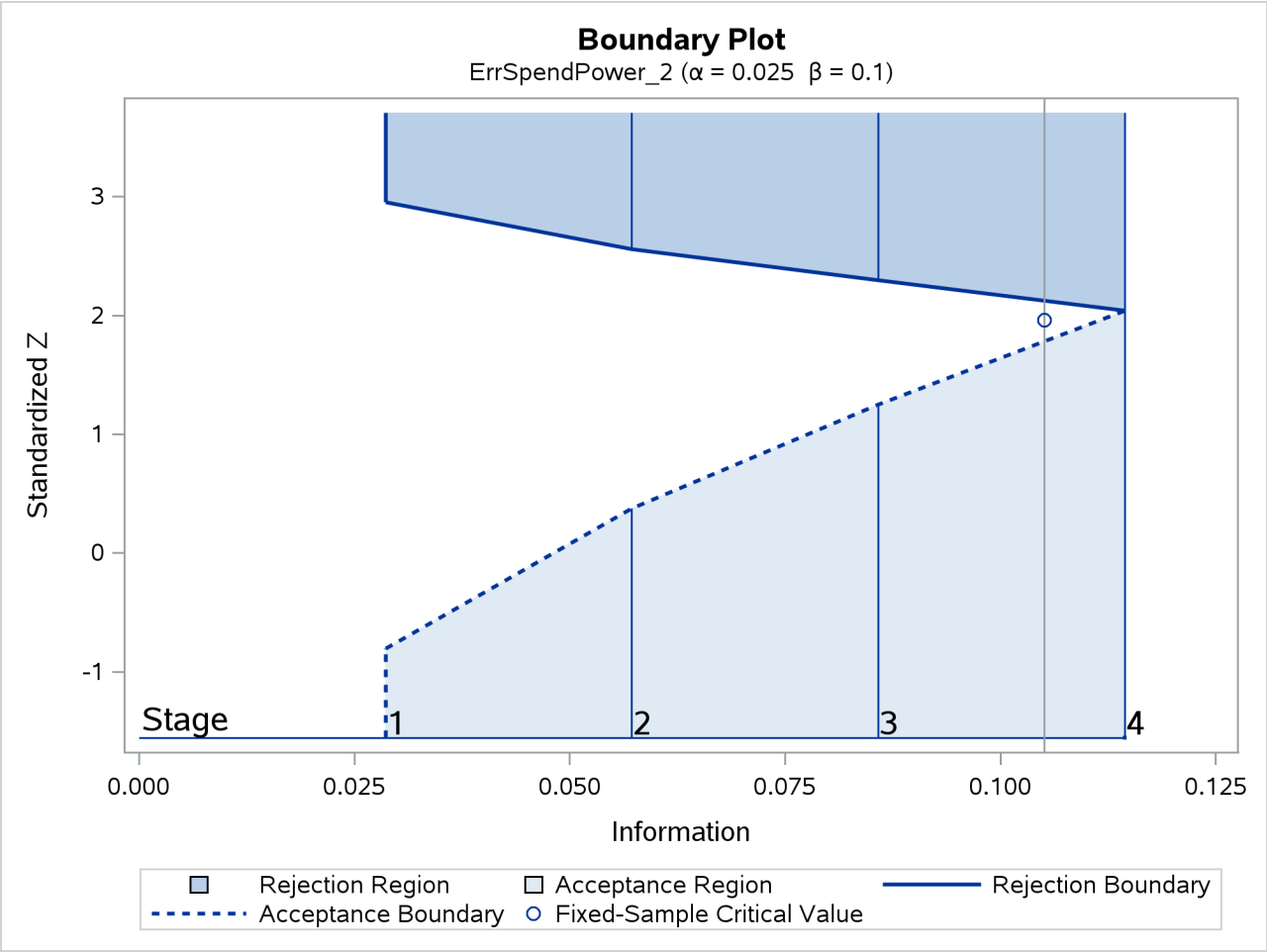

For a design with early stopping either to accept or to reject H_0, the shapes of boundaries affect the critical value at the final stage. For the acceptance boundary, a liberal method at early stages, such as a $\rho = 1$ power family error spending method, lowers the critical value at the final stage. For the rejection boundary, a conservative method at early stages, such as a $\rho = 3$ power family error spending method, also lowers the critical value at the final stage. In addition, a larger Type II error level also lowers the critical value at the final stage. The resulting critical value at the final stage might even be less than the critical value for the corresponding fixed-sample design.

To illustrates how this can occur, the following statements use a $\rho = 1$ power family error spending method for the acceptance boundary and a $\rho = 3$ power family error spending method for the rejection boundary to create a group sequential design:

```
proc seqdesign altref=10;
   ErrSpendPower_3_1: design nstages=4
                      method(alpha)=errfuncpow(rho=3)
                      method(beta)= errfuncpow(rho=1)
                      alt=upper   stop=both
                      alpha=0.025 beta=0.10;
```

```
run;
```

The resulting "Boundary Information" table in Figure 110.14 shows that the boundary value at the final stage 1.9267 is less than 1.96, the critical value of the corresponding fixed-sample design.

Figure 110.14 Boundary Information

The SEQDESIGN Procedure
Design: ErrSpendPower_3_1

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference	Boundary Values	
Stage	Proportion	Actual	Upper	Upper Beta	Alpha
1	0.2500	0.029725	1.72410	-0.23587	3.35935
2	0.5000	0.05945	2.43824	0.63117	2.76024
3	0.7500	0.089175	2.98622	1.31554	2.35119
4	1.0000	0.1189	3.44819	1.92672	1.92672

That is, H_0 can be rejected even if the test statistic at the final stage is less than the critical value for the corresponding fixed-sample design, which is not desirable. Therefore, for a design with an acceptance boundary, the design should be used with care.

Another reason why the rejection boundary should not be affected by the acceptance boundary is that a Data and Safety Monitoring Board (DSMB) might not strictly adhere to this acceptance boundary, and thus the Type I error level might not be maintained. Therefore, it might not be desirable for the rejection boundary to be affected by the acceptance boundary (Lan and DeMets 2009, p. 103).

To use a design whose rejection boundary is not affected by the acceptance boundary and that can still stop the trial early to accept H_0, you can use one of the following strategies:

- You can use the conditional power approach for a design that stops early only to reject H_0. For this design, the conditional power at an interim stage k can be defined as the total probability of rejecting H_0 at any future stages under a specified hypothetical reference, given the observed statistic at stage k. A small conditional power indicates a small probability of success (rejecting H_0) given the current data, and the trial can be stopped early to accept H_0 (Lan, Simon, and Halperin 1982; Lan and DeMets 2009, p. 101). For a detailed description of conditional power, see the section "Stochastic Curtailment" in Chapter 111, "The SEQTEST Procedure."

- You can use a design with a nonbinding acceptance boundary (Zhu, Ni, and Yao 2011, pp. 132–133). For this design, the rejection boundary is created by ignoring the acceptance boundary. Thus, the Type I error level is maintained even when a trial is allowed to continue for a test statistic that falls in the acceptance region, and the rejection boundary values (with the standardized Z statistic scale) are identical to the boundary values that are derived in the corresponding design that stops early only to reject H_0.

Boundary Adjustments for Overlapping Lower and Upper β Boundaries

For the fixed boundary shape methods and Whitehead methods, the boundary values for all stages are derived simultaneously for each boundary. For a two-sided design with STOP=ACCEPT or STOP=BOTH, simultaneous derivation might result in overlapping of the lower and upper β boundaries. That is, at an interim stage k, the lower β boundary value might be greater than its corresponding upper β boundary value. In this case, these two β boundary values are set to missing and the design does not stop at stage k to accept the null hypothesis (Jennison and Turnbull 2000, p. 113).

For the error spending methods, the boundary values are derived sequentially for the stages. For a two-sided design with STOP=ACCEPT or STOP=BOTH, a small β spending at an interim stage might result in overlapping of the lower and upper β boundaries for the two corresponding one-sided tests. Specifically, this form of overlapping occurs at an interim stage k if the upper β boundary value derived from the one-sided test for the upper alternative is less than the lower β boundary value derived from the one-sided test for the lower alternative (Kittelson and Emerson 1999, pp. 881–882; Rudser and Emerson 2007, p. 6). You can use the BETAOVERLAP= option to specify how this type of overlapping is to be handled.

If BETAOVERLAP=ADJUST (which is the default) is specified, the procedure derives the boundary values for the two-sided design and then checks for overlapping of the two one-sided β boundaries at interim stages. If overlapping occurs at a particular stage, the β boundary values for the two-sided design are set to missing (so the trial does not stop to accept the null hypothesis at this stage), and the β spending values at subsequent stages are adjusted proportionally as follows.

If the β boundary values are set to missing at stage k in a K-stage trial, the adjusted β spending value at stage k, e'_k, is updated for these missing β boundary values, and then the β spending values at subsequent stages are adjusted proportionally by

$$e'_j = e'_k + \frac{e_j - e_k}{e_K - e_k}\,(e_K - e'_k)$$

for $j = k + 1, \ldots, K$, where e_j and e'_j are the cumulative β spending values st stage j before and after the adjustment, respectively.

After all these adjusted β spending values are computed, the boundary values are then further modified for these adjusted β spending values.

If you specify BETAOVERLAP=NOADJUST, no adjustment is made when overlapping of one-sided β boundaries occurs. The BETAOVERLAP= option is illustrated in Example 110.10.

Specified and Derived Parameters

In the SEQDESIGN procedure, the type of alternative hypothesis (ALT= option) and the condition for early stopping (STOP= option) must be specified for each sequential design. The drift parameters are derived for each design specified. Other parameters, such as Type I error probability α, Type II error probability β, the alternative reference θ_1, and maximum information are either specified or derived in the SEQDESIGN procedure.

Table 110.8 summarizes the available combinations for the specified and derived parameters in the SEQDESIGN procedure.

Table 110.8 Specified and Derived Parameters in the SEQDESIGN Procedure

Specified Parameters				Derived Parameters				
Alt Ref	**Max Info**	**Alpha**	**Beta**	**Alt Ref**	**Max Info**	**Alpha**	**Beta**	**Drift**
Z	X	X					X	X
Z	X		X			X		X
Z	X					X	X	X
Z		X	X		X			X
	X	X	X	X				X
		X	X					X

The symbol "X" indicates that the parameter is either specified or derived in the design and the symbol "Z" indicates that the alternative reference is either specified explicitly with the ALTREF= option or derived from the SAMPLESIZE statement. The drift parameter is always derived in the SEQDESIGN procedure.

For example, if the ALTREF= option is specified without the MAXINFO= option being specified, then the maximum information is derived in the SEQDESIGN procedure with the specified α and β, as illustrated in Example 110.5.

The drift parameter is the standardized reference difference at the final stage. For a design, the drift parameter is

$$d_l = \theta_{1l}\sqrt{I_X}$$

if it has a lower alternative, and

$$d_u = \theta_{1u}\sqrt{I_X}$$

if it has an upper alternative, where I_X is the maximum information and θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively.

If the alternative reference and the maximum information are not both specified, then the specified α and β are used to derive the drift parameter. Then if either the alternative reference or the maximum information is specified, the other is derived from the drift parameter.

If both the alternative reference and the maximum information are specified, then either the α error or the β error is derived in the procedure. However, for a Haybittle-Peto method with the BOUNDARYKEY=NONE option, both α and β errors are derived from the completely specified boundary values.

For a nonsymmetric two-sided design with different lower and upper specifications (such as different lower and upper α errors, β errors, or alternative references in absolute values), the derived lower and upper boundaries are not symmetric. If the alternative references θ_{1l} and θ_{1u} are not both specified, then the SEQDESIGN procedure assumes symmetric alternative references, $\theta_1 = \theta_{1u} = -\theta_{1l}$, for the computation of the boundary values.

Applicable Boundary Keys

In the SEQDESIGN procedure, the BOUNDARYKEY= option in the DESIGN statement specifies the types of errors to be maintained for the design. Table 110.9 lists applicable boundary keys for designs that use unified family and Haybittle-Peto methods, designs that use error spending methods, and designs that use the Haybittle-Peto method only.

Table 110.9 Applicable Boundary Keys for Designs without Whitehead Methods

Method	Specified Parameters (Alt Ref – Max Info)	Boundary Keys Alpha	Beta	None	Both
Unified	X	X	X		
Unified/Haybittle-Peto	X	X	X		
Error spending	X	X	X		
Haybittle-Peto	X	X	X	X	
Unified/Haybittle-Peto					X
Error spending					X
Haybittle-Peto					X

Note that the symbol "X" under "(Alt Ref – Max Info)" indicates that both the alternative reference and maximum information are specified, and the method "Unified/Haybittle-Peto" indicates that both the unified method and the Haybittle-Peto method are used in the same design.

If the ALTREF= and MAXINFO= options are both specified, then Type I and Type II error probability levels cannot be met simultaneously if both error probabilities are specified. The BOUNDARYKEY=ALPHA option maintains the Type I error probability level α and derives Type II error probability β. The BOUNDARYKEY=BETA option maintains the Type II error probability level β and derives Type I error probability α.

If the Haybittle-Peto method is used for all boundaries, the BOUNDARYKEY=NONE option uses the specified α boundary value (STOP=REJECT or STOP=BOTH) and the specified β boundary value (STOP=ACCEPT) at the final stage for the design.

If the ALTREF= and MAXINFO= options are not both specified, the BOUNDARYKEY=BOTH option derives boundary values that maintain both Type I and Type II error probability levels.

Table 110.10 lists applicable boundary keys for a design that uses Whitehead methods.

Table 110.10 Applicable Boundary Keys for Whitehead Methods

Early Stopping	Specified Parameters (Alt Ref – Max Info)	Tau	Boundary Keys Alpha	Beta	None	Both
Reject H_0	X	X	X	X		
Accept H_0	X	X	X	X		
Reject/Accept H_0	X	0.25	X	X		
Reject H_0		X				X
Accept H_0		X				X
Reject/Accept H_0		0.25	X	X	X	

Note that the symbol "X" under "(Alt Ref – Max Info)" indicates that both alternative reference and maximum information are specified.

If the ALTREF= and MAXINFO= options are both specified, then Type I and Type II error probability levels cannot be achieved simultaneously if both are specified. the BOUNDARYKEY=ALPHA option maintains the Type I error probability level α and derives Type II error probability β. The BOUNDARYKEY=BETA option maintains the Type II error probability level β and derives the Type I error probability α.

If the ALTREF= and MAXINFO= options are not both specified, then for a design with the STOP=REJECT or STOP=ACCEPT option, the BOUNDARYKEY=BOTH option derives boundary values that maintain both Type I and Type II error probability levels. If the STOP=BOTH option is specified, Whitehead's triangular method produces boundaries with approximate Type I and Type II error probabilities. The BOUNDARYKEY=NONE option specifies no adjustment to these boundaries. The BOUNDARYKEY=ALPHA and BOUNDARYKEY=BETA options maintain the Type I error probability level α and Type II error probability level β, respectively, by adjusting boundary values at the final stage.

Sample Size Computation

The SEQDESIGN procedure assumes that the data are from a multivariate normal distribution and the sequence of the standardized test statistics $\{Z_1, Z_2, \ldots, Z_K\}$ has the following canonical joint distribution:

- $(Z_1, Z_2, \ldots, Z_K)$ is multivariate normal
- $Z_k \sim N\left(\theta\sqrt{I_k}, 1\right)$
- $\mathrm{Cov}(Z_{k_1}, Z_{k_2}) = \sqrt{I_{k_1}/I_{k_2}} \quad , \quad 1 \le k_1 \le k_2 \le K$

where K is the total number of stages and I_k is the information available at stage k.

If the test statistic is computed from the data that are not from a normal distribution, such as a binomial distribution, then it is assumed that the test statistic is computed from a large sample such that the statistic has an approximately normal distribution.

In a typical clinical trial, the sample size required depends on the Type I error probability level α, alternative reference θ_1, power $1-\beta$, and variance of the response variable. Given a one-sided null hypothesis $H_0 : \theta = 0$ with an upper alternative hypothesis $H_1 : \theta = \theta_1$, the information required for a fixed-sample test is given by

$$I_0 = \frac{(\Phi^{-1}(1-\alpha) + \Phi^{-1}(1-\beta))^2}{\theta_1^2}$$

The parameter θ and the subsequent alternative reference θ_1 depend on the test specified in the clinical trial. For example, suppose you are comparing two binomial populations $p_a = p_b$; then $\theta = p_a - p_b$ is the difference between two proportions if the proportion difference statistic is used, and $\theta = \log\left(\frac{p_a(1-p_b)}{p_b(1-p_a)}\right)$, the log odds ratio for the two proportions if the log odds ratio statistic is used.

If the maximum likelihood estimate $\hat{\theta}$ from the likelihood function can be derived, then the asymptotic variance for $\hat{\theta}$ is $\mathrm{Var}(\hat{\theta}) = 1/I$, where I is Fisher information for θ. The resulting statistic $\hat{\theta}$ corresponds to the MLE statistic scale as specified in the BOUNDARYSCALE=MLE option in the PROC SEQDESIGN statement, $\hat{\theta}\sqrt{I}$ corresponds to the standardized Z scale (BOUNDARYSCALE=STDZ), and $\hat{\theta}\ I$ corresponds to the score statistic scale (BOUNDARYSCALE=SCORE).

Alternatively, if the score statistic S is derived in a statistical procedure, it can be used as the test statistic and its asymptotic variance is given by Fisher information, I. In this case, $S/\sqrt{I}$ corresponds to the standardized Z scale and S/I corresponds to the MLE statistic scale.

For a group sequential trial, the maximum information I_X is derived in the SEQDESIGN procedure with the specified α, β, and θ_1. With the maximum information

$$I_X = \frac{1}{\mathrm{Var}(\hat{\theta})}$$

the sample size required for a specified test statistic in the trial can be evaluated or estimated from the known or estimated variance of the response variable. Note that different designs might produce different maximum information levels for the same hypothesis, and this in turn might require a different number of observations for the trial.

If each observation in the data set provides one unit of information in a hypothesis testing, such as a one-sample test for the mean, the required sample size for the sequential design can be derived from the maximum information. However, for a survival analysis, an individual in the survival time data might provide only partial information because of censoring. In this case, the required number of events can be derived from the maximum information. With addition accrual information, the sample size can also be computed.

The SEQDESIGN procedure provides sample size computation for some one-sample and two-sample tests in the SAMPLESIZE statement. It also provides sample size computation for tests of a parameter in regression models such as normal regression, logistic regression, and proportional hazards regression. In addition, the procedure can also compute the required sample size or number of events from the corresponding number in the fixed-sample design.

Table 110.11 lists the *options* available in the SAMPLESIZE statement.

Table 110.11 SAMPLESIZE Statement Options

Option	Description
Fixed-Sample Models	
INPUTNOBS	Specifies sample size for fixed-sample design
INPUTNEVENTS	Specifies number of events for fixed-sample design
One-Sample Models	
ONESAMPLEMEAN	Specifies one-sample Z test for mean
ONESAMPLEFREQ	Specifies one-sample test for binomial proportion
Two-Sample Models	
TWOSAMPLEMEAN	Specifies two-sample Z test for mean difference
TWOSAMPLEFREQ	Specifies two-sample test for binomial proportions
TWOSAMPLESURVIVAL	Specifies log-rank test for two survival distributions
Regression Models	
REG	Specifies test for a regression parameter
LOGISTIC	Specifies test for a logistic regression parameter
PHREG	Specifies test for a proportional hazards regression parameter

The MODEL=INPUTNOBS and MODEL=INPUTNEVENTS options are described next, and the remaining options are described in the next three sections.

Input Sample Size for Fixed-Sample Design

The MODEL=INPUTNOBS option derives the sample size required for a group sequential trial from the sample size n_0 for the corresponding fixed-sample design. With the N= n_0 option specifying the sample size n_0 for a fixed-sample design, the sample size required for a group sequential trial is then computed as

$$N_X = \frac{I_X}{I_0}\, n_0$$

where I_X is the maximum information for the group sequential design and I_0 is the information for the corresponding fixed-sample design. The information ratio between I_X and I_0 is derived in the SEQDESIGN procedure.

The SAMPLE=ONE option specifies a one-sample test, and the SAMPLE=TWO option specifies a two-sample test. For a two-sample test, the WEIGHT= option specifies the sample size allocation weights for the two groups.

Input Number of Events for Fixed-Sample Design

The MODEL=INPUTNEVENTS option derives the number of events required for a group sequential trial from the number of events d_0 for the corresponding fixed-sample design. With the D= d_0 option specifies the number of events d_0 for a fixed-sample survival analysis, the number of events required for a group sequential trial is then computed as

$$d_X = \frac{I_X}{I_0}\, d_0$$

where I_X is the maximum information for the group sequential design and I_0 is the information for the corresponding fixed-sample design. The information ratio between I_X and I_0 is derived in the SEQDESIGN procedure.

The SAMPLE=ONE option specifies a one-sample test, and the SAMPLE=TWO option specifies a two-sample test. For a two-sample test, the WEIGHT= option specifies the sample size allocation weights for the two groups.

The ACCRUAL= option specifies the method for individual accrual. The ACCRUAL=UNIFORM option (which is the default) specifies that the individual accrual is uniform in the accrual time T_a with a constant accrual rate r_a, and the ACCRUAL=EXP(PARM=γ_0) option specifies that the individual accrual is truncated exponential with a scaled power parameter γ_0, where $\gamma_0 \geq -10$ and $\gamma_0 \neq 0$. With a scaled parameter γ_0, the power parameter for the truncated exponential with the accrual time T_a is given by $\gamma = \gamma_0/T_a$.

The LOSS= option specifies the individual loss to follow up in the sample size computation. The LOSS=NONE option (which is the default) specifies no loss to follow up, and the EXP(POWER=τ) option specifies exponential loss function with a power parameter τ.

With the computed number of events d_X for a group sequential survival design, the required total sample size and sample size at each stage can be derived from specifications of hazard rates, accrual information, and losses to follow-up information. For each study group, the hazard rate h is constant (which corresponds to an exponential survival distribution) in the sample size computation.

The next four subsections describe required sample sizes for uniform accrual (with and without losses to follow up) and for truncated exponential accrual (with and without losses to follow up).

Uniform Accrual without Losses to Follow Up

For a study group with a constant hazard rate h, if the individual accrual is uniform in the accrual time T_a with a constant accrual rate r_a, Kim and Tsiatis (1990, pp. 83–84) show that the expected number of events by time t is given by

$$D_h(t) = \begin{cases} r_a\left(t - \frac{1-e^{-ht}}{h}\right) & \text{if } t \leq T_a \\ r_a\left(T_a - \frac{e^{-ht}}{h}(e^{hT_a} - 1)\right) & \text{if } t > T_a \end{cases}$$

For a one-sample design (such as a proportional hazards regression), the expected number of events by time t is $E(t) = D_h(t)$, where h is the hazard rate for the group. For a two-sample design (such as a log-rank test for two survival distributions), the expected number of events by time t is

$$E(t) = \frac{R}{R+1} D_{h_a}(t) + \frac{1}{R+1} D_{h_b}(t)$$

where h_a and h_b are hazard rates in groups A and B, respectively, and R is the ratio of the sample size allocation weights w_a/w_b.

If the accrual rate r_a is specified with one of the three time parameters—the accrual time, follow-up time, and total study time—then PROC SEQDESIGN derives the other two time parameters by solving the equation for the expected number of events. Similarly, if the accrual rate r_a is not specified but two of the three time parameters are specified, then PROC SEQDESIGN derives the accrual rate.

If the accrual rate r_a is specified without the accrual time T_a, follow-up time T_f, and total study time $T = T_a + T_f$, the minimum and maximum accrual times can be computed from the following equation, as

described in Kim and Tsiatis (1990, p. 85):

$$\frac{d_X}{r_a} \le T_a \le E^{-1}(d_X)$$

With the accrual rate r_a and the accrual time T_a, the total sample size is

$$N_X = r_a\ T_a$$

At each stage k, the number of events is given by

$$d_k = \frac{I_k}{I_X}\ d_X$$

The corresponding time T_k can be derived from the equation for the expected number of events, $E(t) = d_k$, and the resulting sample size is computed as

$$N_k = \begin{cases} r_a\ T_k & \text{if } T_k \le T_a \\ r_a\ T_a & \text{if } T_k > T_a \end{cases}$$

Uniform Accrual with Losses to Follow Up

With the LOSS=EXP(HAZARD=τ) option, the individual loss to follow up has an exponential loss distribution function

$$H(t) = 1 - e^{-\tau t}$$

where $\tau \ge 0$ is the loss hazard rate. The loss hazard rate can also be specified implicitly with the MEDTIME=t_τ suboption through the median loss time t_τ.

For a study group with a constant hazard rate h, if the individual accrual is uniform in the accrual time T_a with a constant accrual rate r_a and the individual loss to follow up has an exponential loss distribution function $H(t)$, Lachin and Foulkes (1986, p. 511) derive the expected number of events by time t (where $t > T_a$) as

$$D_h(t, T_a) = r_a\ T_a\ \frac{h}{h+\tau}\left(1 - \frac{1}{(h+\tau)T_a} e^{-(h+\tau)t}(e^{h+\tau)T_a} - 1)\right)$$

For $t \le T_a$, the SEQDESIGN procedure estimates the expected number of events by time t as

$$D_h(t, T_a) = r_a\, t\ \frac{h}{h+\tau}\left(1 - \frac{1}{(h+\tau)\,t}(1 - e^{-(h+\tau)t})\right)$$

For a one-sample design (such as a proportional hazards regression), the expected number of events by time t is $E(t, T_a) = D_h(t, T_a)$, where h is the hazard rate for the group. For a two-sample design (such as a log-rank test for two survival distributions), the expected number of events by time t is

$$E(t, T_a) = \frac{R}{R+1} D_{h_a}(t, T_a) + \frac{1}{R+1} D_{h_b}(t, T_a)$$

where h_a and h_b are hazard rates in groups A and B, respectively, and R is the ratio of the sample size allocation weights w_a/w_b.

If the accrual rate r_a is specified with one of the three time parameters—the accrual time, follow-up time, and total study time—then PROC SEQDESIGN derives the other two time parameters by solving the equation for the expected number of events. Similarly, if the accrual rate r_a is not specified, but two of the three time parameters are specified, then PROC SEQDESIGN derives the accrual rate.

If the accrual rate r_a is specified without the accrual time T_a, follow-up time T_f, and total study time $T = T_a + T_f$, the SEQDESIGN procedure computes the minimum accrual time T_a by solving the equation

$$E(\infty, T_a) = d_X$$

A closed-form solution is then given by

$$\frac{d_X}{r_a}\left(\frac{R}{R+1}\,\frac{h_a}{h_a+\tau} + \frac{1}{R+1}\,\frac{h_b}{h_b+\tau}\right)^{-1}$$

Similarly, the SEQDESIGN procedure derives the maximum accrual time T_a by solving the equation

$$E(T_a, T_a) = d_X$$

The maximum accrual time T_a is then obtained by an iterative process.

With the accrual rate r_a and the accrual time T_a, the total sample size is

$$N_X = r_a\, T_a$$

At each stage k, the number of events is given by

$$d_k = \frac{I_k}{I_X}\, d_X$$

The corresponding time T_k can be derived from the equation for the expected number of events, $E(t) = d_k$, and the resulting sample size is computed as

$$N_k = \begin{cases} r_a\, T_k & \text{if } T_k \le T_a \\ r_a\, T_a & \text{if } T_k > T_a \end{cases}$$

Truncated Exponential Accrual without Losses to Follow Up

For a study group with a constant hazard rate h, if the individual accrual is truncated exponential with parameter γ over the accrual period from 0 to T_a with distribution

$$F(t) = \frac{1 - e^{-\gamma t}}{1 - e^{-\gamma T_a}} \quad \text{for } 0 \le t \le T_a$$

Lachin and Foulkes (1986, p. 510) derive the expected number of events by time t (where $t > T_a$) as

$$D_h(t, N) = N\left(1 + \frac{\gamma}{h-\gamma}\,\frac{e^{-ht} - e^{-h(t-T_a)}e^{-\gamma T_a}}{1 - e^{-\gamma T_a}}\right)$$

where N is the total sample size.

For $t \leq T_a$, the SEQDESIGN procedure estimates the expected number of events by time t as

$$D_h(t, N) = N\ F(t) \left(1 + \frac{\gamma}{h - \gamma}\ \frac{e^{-ht} - e^{-\gamma t}}{1 - e^{-\gamma t}} \right)$$

For the truncated exponential accrual function with a parameter γ over the accrual period from 0 to T_a, you specify the scaled parameter $\gamma_0 = \gamma T_a$ in the ACCRUAL=EXP(PARM=γ_0) option, where $\gamma_0 \geq -10$ and $\gamma_0 \neq 0$.

For a one-sample design (such as a proportional hazards regression), the expected number of events by time t is $E(t, N) = D_h(t, N)$, where h is the hazard rate for the group. For a two-sample design (such as a log-rank test for two survival distributions), the expected number of events by time t is

$$E(t, N) = \frac{R}{R+1} D_{h_a}(t, N) + \frac{1}{R+1} D_{h_b}(t, N)$$

where h_a and h_b are hazard rates in groups A and B, respectively, and R is the ratio of the sample size allocation weights w_a / w_b.

If the total sample size N is specified, then at least one of the three time parameters—the accrual time, follow-up time, and total study time—must be specified, and then PROC SEQDESIGN derives the other two time parameters by solving the equation for the expected number of events. Similarly, if the total sample size N is not specified, then at least two of the three time parameters must be specified, and PROC SEQDESIGN derives the sample size.

If the accrual sample size N is not specified, the SEQDESIGN procedure computes the minimum accrual sample size by solving the equation

$$E(\infty, N) = d_X$$

That is, $N = d_X$.

Similarly, if the total sample size N is not specified but the accrual time T_a is specified, the SEQDESIGN procedure derives the maximum accrual sample size N by solving the equation

$$E(T_a, N) = d_X$$

At each stage k, the number of events is given by

$$d_k = \frac{I_k}{I_X}\ d_X$$

The corresponding time T_k can be derived from the equation for the expected number of events, $E(t) = d_k$, and the resulting sample size is computed as

$$N_k = \begin{cases} N\ F(T_k) & \text{if } T_k \leq T_a \\ N & \text{if } T_k > T_a \end{cases}$$

Truncated Exponential Accrual with Losses to Follow Up

With the LOSS=EXP(HAZARD=τ) option, the individual loss to follow up has an exponential loss distribution function

$$H(t) = 1 - e^{-\tau t}$$

where $\tau \geq 0$ is the loss hazard rate. The loss hazard rate can also be specified implicitly with the MEDTIME=t_τ suboption through the median loss time t_τ.

For a study group with a constant hazard rate h, if the individual accrual is truncated exponential with parameter γ over the accrual period from 0 to T_a with distribution

$$F(t) = \frac{1 - e^{-\gamma t}}{1 - e^{-\gamma T_a}} \quad \text{for } 0 \leq t \leq T_a$$

and the individual loss to follow up has an exponential loss distribution function $H(t)$, Lachin and Foulkes (1986, p. 513) derive the expected number of events by time t (where $t > T_a$) as

$$D_h(t, N) = N \frac{h}{h+\tau} \left(1 + \frac{\gamma}{h+\tau-\gamma} \frac{e^{-(h+\tau)t} - e^{-(h+\tau)(t-T_a)} e^{-\gamma T_a}}{1 - e^{-\gamma T_a}} \right)$$

For $t \leq T_a$, the SEQDESIGN procedure estimates the expected number of events by time t as

$$D_h(t, N) = N\, F(t) \frac{h}{h+\tau} \left(1 + \frac{\gamma}{h+\tau-\gamma} \frac{e^{-(h+\tau)t} - e^{-\gamma t}}{1 - e^{-\gamma t}} \right)$$

For the truncated exponential accrual function with a parameter γ over the accrual period from 0 to T_a, you specify the scaled parameter $\gamma_0 = \gamma T_a$ in the ACCRUAL=EXP(PARM=γ_0) option, where $\gamma_0 \geq -10$ and $\gamma_0 \neq 0$.

For a one-sample design (such as a proportional hazards regression), the expected number of events by time t is $E(t, N) = D_h(t, N)$, where h is the hazard rate for the group. For a two-sample design (such as a log-rank test for two survival distributions), the expected number of events by time t is

$$E(t, N) = \frac{R}{R+1} D_{h_a}(t, N) + \frac{1}{R+1} D_{h_b}(t, N)$$

where h_a and h_b are hazard rates in groups A and B, respectively, and R is the ratio of the sample size allocation weights w_a/w_b.

If the total sample size N is specified, then at least one of the three time parameters—the accrual time, follow-up time, and total study time—must be specified, and then PROC SEQDESIGN derives the other two time parameters by solving the equation for the expected number of events. Similarly, if the total sample size N is not specified, then at least two of the three time parameters must be specified, and PROC SEQDESIGN derives the sample size.

If the accrual sample size is not specified, the SEQDESIGN procedure computes the minimum sample size N by solving the equation

$$E(\infty, N) = d_X$$

A closed-form solution is then given by

$$d_X \left(\frac{R}{R+1} \frac{h_a}{h_a+\tau} + \frac{1}{R+1} \frac{h_b}{h_b+\tau} \right)^{-1}$$

Similarly, if the accrual sample size N is not specified but the accrual time T_a is specified, the SEQDESIGN procedure derives the maximum accrual sample size N by solving the equation

$$E(T_a, N) = d_X$$

At each stage k, the number of events is given by

$$d_k = \frac{I_k}{I_X} d_X$$

The corresponding time T_k can be derived from the equation for the expected number of events, $E(t) = d_k$, and the resulting sample size is computed as

$$N_k = \begin{cases} N \ F(T_k) & \text{if } T_k \le T_a \\ N & \text{if } T_k > T_a \end{cases}$$

The following three sections describe examples of test statistics with their resulting information levels, which can then be used to derive the required sample size. The maximum likelihood estimators are used for all tests except to compare two survival distributions with a log-rank test, where a score statistic is used.

Applicable One-Sample Tests and Sample Size Computation

The SEQDESIGN procedure provides sample size computation for two one-sample tests: normal mean and binomial proportion. The required sample size depends on the variance of the response variable—that is, the sample proportion for a binomial proportion test.

In a typical clinical trial, a hypothesis is designed to reject, not accept, the null hypothesis to show the evidence for the alternative hypothesis. Thus, in most cases, the proportion under the alternative hypothesis is used to derive the required sample size. For a test of the binomial proportion, the REF=NULLPROP and REF=PROP options use proportions under the null and alternative hypotheses, respectively.

Test for a Normal Mean

The MODEL=ONESAMPLEMEAN option in the SAMPLESIZE statement derives the sample size required to test a normal mean by using the sample mean statistic for the null hypothesis $\mu = \mu_0$. At stage k, the sample mean is computed as

$$\overline{y}_k = \frac{1}{N_k} \sum_{j=1}^{N_k} y_{kj}$$

where y_{kj} is the value of the jth observation available in the kth stage and N_k is the cumulative sample size at stage k.

An equivalent hypothesis is $H_0 : \theta = 0$, where $\theta = \mu - \mu_0$.

The MLE statistic for θ,

$$\hat{\theta}_k = \overline{y}_k - \mu_0 \sim N\left(\theta,\ {I_k}^{-1}\right)$$

where the information

$$I_k = \frac{1}{\mathrm{Var}(\hat{\theta})} = \frac{1}{\mathrm{Var}(\overline{y}_k)} = \frac{N_k}{\sigma^2}$$

is the inverse of the variance.

That is, the standardized statistic

$$Z_k = \hat{\theta}_k \sqrt{I_k} = (\overline{y}_k - \mu_0)\sqrt{I_k} \sim N\left(\theta\sqrt{I_k},\ 1\right)$$

Thus, to test the hypothesis $H_0 : \theta = 0$ against a two-sided alternative $H_1 : \theta = \theta_1$, H_0 is rejected at stage k if the statistic Z_k is less than or equal to the lower α boundary value or if Z_k is greater than or equal to the upper α boundary value at stage k.

If the variance σ^2 is unknown, the sample variance can be used if it is assumed that the sample variance is computed from a large sample such that the test statistic has an approximately normal distribution.

The maximum information is needed to derive the required sample size. If the maximum information is not specified or derived with the ALTREF= option in the procedure, the MEAN=θ_1 option in the SAMPLESIZE statement is used to specify the alternative reference and thus to derive the maximum information.

In the SEQDESIGN procedure, the computed total sample size

$$N_K = \sigma^2\ I_X$$

where I_X is the maximum information and σ is the specified standard deviation. With an available maximum information, you can specify the MODEL=ONESAMPLEMEAN(STDDEV= σ) option in the SAMPLESIZE statement to compute the required total sample size and individual sample size at each stage. A procedure such as PROC MEANS can be used to derive a one-sample Z test for a normal mean.

Test for a Binomial Proportion

The MODEL=ONESAMPLEFREQ option in the SAMPLESIZE statement derives the sample size required to test a binomial proportion by using the null hypothesis $p = p_0$, where p is the proportion of a binomial population. At stage k, the MLE for p is computed as

$$\hat{p}_k = \frac{1}{N_k}\sum_{j=1}^{N_k} y_{kj}$$

where y_{kj} is the value of the jth observation available in the kth stage and N_k is the cumulative sample size at stage k.

An equivalent hypothesis is $H_0 : \theta = 0$, where $\theta = p - p_0$. If p_0 is not close to 0 or 1, then for a large sample, $\hat{\theta}_k = \hat{p}_k - p_0$ has an approximately normal distribution

$$\hat{\theta}_k \sim N\left(\theta,\ I_k^{-1}\right)$$

where the information $I_k = (p\,(1-p)\,/\,N_k)^{-1}$ is the inverse of the variance $\mathrm{Var}(\hat{\theta})$.

Then the standardized statistic

$$Z_k = \hat{\theta}_k \sqrt{I_k} \sim N\left(\theta\sqrt{I_k},\ 1\right)$$

In practice, the estimated sample proportion $\hat{p}$ at stage k can be used to derive the information I_k and test statistic Z_k. Thus, to test the hypothesis H_0 against an upper alternative $H_1 : \theta = \theta_1 > 0$, H_0 is rejected at stage k if the statistic Z_k is greater than or equal to the upper α boundary at stage k.

The maximum information I_X is needed to derive the required sample size. If the maximum information is not specified or derived with the ALTREF= option in the procedure, the PROP= option in the SAMPLESIZE statement is used to specify the alternative reference and to derive the maximum information for the sample size calculation.

It is assumed that the sample size is sufficiently large such that the test statistic has an approximately normal distribution. With the hypotheses $H_0 : p = p_0$ and $H_1 : p = p_1$, the SEQDESIGN procedure derives the total sample size

$$N_X = p^*\,(1-p^*)\,I_X$$

where $p^* = p_0$ if REF=NULLPROP is specified. Otherwise, $p^* = p_1$.

If the PROP= option in the SAMPLESIZE statement is not specified, then the alternative reference θ_1 derived in the SEQDESIGN procedure is used to compute $p_1 = p_0 + \theta_1$.

The ALTREF= option in the PROC statement can be used to specify θ_1. Otherwise, the PROP= option in the SAMPLESIZE statement must be specified.

For example, with $H_0 : p = 0.5$, $H_1 : p = 0.6$, and REF=PROP (which is the default),

$$N_K = p^*(1-p^*)\,I_X = (0.6 \times 0.4)\,I_X = 0.24\,I_X$$

You can specify the MODEL=ONESAMPLEFREQ option in the SAMPLESIZE statement to compute the required total sample size and individual sample size at each stage. A procedure such as PROC GENMOD with the default DIST=NORMAL option in the MODEL statement can be used to derive the Z test for a binomial proportion.

Applicable Two-Sample Tests and Sample Size Computation

The SEQDESIGN procedure provides sample size computation for two-sample tests: the test for the difference between two normal means, tests for binomial proportions, and the log-rank test for two survival distributions. These tests for binomial proportions include the test for the difference between two binomial proportions, the log odds ratio test for binomial proportions, and the log relative risk test for binomial proportions,

For a test of difference between two sample means, the required sample size depends on the assumed sample variances. Similarly, for a test of two-sample proportions, the required sample size depends on the assumed sample proportions. For a log-rank test of two survival distributions, the required sample size depends on the assumed sample hazard rates, accrual rate, and accrual time.

If the REF=NULLPROP or REF=NULLHAZARD option is specified, the proportions or hazard rates under the null hypothesis are used to derive the required sample size or number of events. Otherwise, the REF=PROP option (which is the default in the MODEL=TWOSAMPLEFREQ option) or the REF=HAZARD option (which is the default in the MODEL=TWOSAMPLESURVIVAL option) uses proportions or hazard rates under the alternative hypothesis to derive the required sample size or number of events.

Test for the Difference between Two Normal Means

The MODEL=TWOSAMPLEMEAN option in the SAMPLESIZE statement derives the sample size required to test the difference between the means of two normal populations μ_a and μ_b by using the null hypothesis $H_0 : \theta = 0$, where $\theta = \mu_a - \mu_b$.

At stage k, the MLE for θ is computed as

$$\hat{\theta}_k = \overline{y}_{ak} - \overline{y}_{bk} = \frac{1}{N_{ak}} \sum_{j=1}^{N_{ak}} y_{akj} - \frac{1}{N_{bk}} \sum_{j=1}^{N_{bk}} y_{bkj}$$

where y_{akj} and y_{bkj} are the values of the jth observation available in the kth stage groups A and B, respectively, and N_{ak} and N_{bk} are the cumulative sample sizes at stage k for these two groups.

The statistic $\hat{\theta}_k$ has a normal distribution

$$\hat{\theta}_k \sim N\left(\theta,\ {I_k}^{-1}\right)$$

where the information I_k is the inverse of the variance $\mathrm{Var}(\hat{\theta}_k) = \sigma_a^2/N_{ak} + \sigma_b^2/N_{bk}$.

Then the standardized statistic

$$Z_k = \hat{\theta}_k \sqrt{I_k} \sim N\left(\theta\sqrt{I_k},\ 1\right)$$

Thus, to test the hypothesis $H_0 : \theta = 0$ against an upper alternative $H_1 : \theta = \theta_1, \theta_1 > 0$, H_0 is rejected at stage k if the statistic $Z_k \geq a_k$, the upper α boundary for the standardized Z statistic at stage k.

If the variances σ_a^2 and σ_b^2 are unknown, the sample variances can be used to derive the information I_k if it is assumed that each sample variance is computed from a large sample such that the test statistic has an approximately normal distribution.

The maximum information is needed to derive the required sample size. If the maximum information is not specified or derived in the procedure, the alternative reference θ_1^* specified in the MEANDIFF option is used to derive the maximum information.

Note that in order to derive the sample sizes N_{ak} and N_{bk} uniquely from the information, $N_{ak} = R\, N_{bk}$ is assumed for $k = 1, 2, \ldots, K$, where $R = w_a/w_b$ is the constant allocation ratio computed from the WEIGHT=w_a w_b option in the SAMPLESIZE statement.

In PROC SEQDESIGN, the computed total sample sizes for the two groups are

$$N_{aK} = (\sigma_a^2 + R\,\sigma_b^2)\, I_X = R\,(\frac{\sigma_a^2}{R} + \sigma_b^2)\, I_X$$

$$N_{bK} = (\frac{\sigma_a^2}{R} + \sigma_b^2)\, I_X$$

where I_X is the maximum information derived in the SEQDESIGN procedure, R is the constant allocation ratio, and σ_a and σ_b are the specified standard deviations.

For $R = 1$, the two sample sizes are equal, then

$$N_{aK} = N_{bK} = \frac{N_K}{2} = (\sigma_a^2 + \sigma_b^2)\, I_X$$

If the variances from the two groups are equal, $\sigma_a^2 = \sigma_b^2 = \sigma^2$, then the total sample sizes for the two groups are

$$N_{aK} = (1 + R)\, \sigma^2\, I_X$$

$$N_{bK} = (1 + \frac{1}{R})\, \sigma^2\, I_X$$

and the total sample size is

$$N_X = N_{aK} + N_{bK} = \frac{(R+1)^2}{R}\, \sigma^2\, I_X$$

Furthermore, for $R = 1$, the two sample sizes are equal, then

$$N_{aK} = N_{bK} = \frac{N_X}{2} = 2\, \sigma^2\, I_X$$

With an available maximum information, you can specify the MODEL=TWOSAMPLEMEAN(WEIGHT= R STDDEV= σ_a σ_b) option in the SAMPLESIZE statement to compute the required total sample size and individual sample size at each stage. A procedure such as PROC GLM can be used to derive the two-sample Z test for the mean difference.

Test for the Difference between Two Binomial Proportions

The MODEL=TWOSAMPLEFREQ(TEST=PROP) option in the SAMPLESIZE statement derives the sample size required to test the difference between two binomial populations with $H_0 : \theta = 0$, where $\theta = p_a - p_b$. At stage k, the MLE for θ is

$$\hat{\theta}_k = \hat{p}_{ak} - \hat{p}_{bk} = \frac{1}{N_{ak}} \sum_{j=1}^{N_{ak}} y_{akj} - \frac{1}{N_{bk}} \sum_{j=1}^{N_{bk}} y_{bkj}$$

where y_{akj} and y_{bkj} are the values of the jth observation available in the kth stage for groups A and B, respectively, and N_{ak} and N_{bk} are the cumulative sample sizes at stage k for these two groups.

For sufficiently large sample sizes N_{ak} and N_{bk}, the statistic $\hat{\theta}_k$ has an approximate normal distribution

$$\hat{\theta}_k \sim N\left(\theta,\, {I_k}^{-1}\right)$$

where the information is the inverse of the variance

$$\mathrm{Var}(\hat{\theta}_k) = \frac{p_a\,(1 - p_a)}{N_{ak}} + \frac{p_b\,(1 - p_b)}{N_{bk}}$$

Thus, the standardized statistic

$$Z_k = \hat{\theta}_k \sqrt{I_k} \sim N\left(\theta\sqrt{I_k},\, 1\right)$$

In practice, $p_a = \hat{p}_a$ and $p_b = \hat{p}_b$, the estimated sample proportions for groups A and B, respectively, at stage k, can be used to derive the information I_k and the test statistic Z_k. Thus, to test the hypothesis H_0 against an upper alternative $H_1 : \theta > 0$, H_0 is rejected at stage k if the statistic $Z_k \geq a_k$, the upper α boundary for the standardized Z statistic at stage k.

The maximum information I_X is needed to derive the required sample size. If the maximum information is not specified or derived with the ALTREF= option in the procedure, the PROP= option in the SAMPLESIZE statement is used to provide proportions under the alternative hypothesis for the alternative reference and then to derive the maximum information.

The proportions in the two groups are needed to derive the sample size. Also, in order to derive the sample sizes N_{ak} and N_{bk} uniquely from the information, $N_{ak} = R\, N_{bk}$ is assumed for $k = 1, 2, \ldots, K$, where $R = w_a/w_b$ is the constant allocation ratio computed from the WEIGHT=$w_a\ w_b$ option in the SAMPLESIZE statement. Then

$$I_X = \left(\frac{p_a\,(1-p_a)}{N_{aK}} + \frac{p_b\,(1-p_b)}{N_{bK}}\right)^{-1} = \frac{N_{aK}}{p_a(1-p_a) + R\,p_b(1-p_b)}$$

In PROC SEQDESIGN, the total sample sizes in the two groups are computed as

$$N_{aK} = \left(p_a^*\,(1-p_a^*) + R\,p_b^*\,(1-p_b^*)\right)\, I_X$$

$$N_{bK} = \frac{1}{R}\, N_{aK}$$

where $R = w_a/w_b$ is the constant allocation ratio, and p_a^* and p_b^* are proportions specified with the REF= option:

- REF=NULLPROP uses proportions under H_0: $p_a^* = p_{0a}$, $p_b^* = p_{0b}$
- REF=AVGNULLPROP uses the average proportion under H_0: $p_a^* = p_b^* = (Rp_{0a} + p_{0b})/(R+1)$
- REF=PROP uses proportions under H_1: $p_a^* = p_{1a}$, $p_b^* = p_{1b}$
- REF=AVGPROP uses the average proportion under $H_1 : p_a^* = p_b^* = (Rp_{1a} + p_{1b})/(R+1)$

The total sample size is given by

$$N_X = N_{aK} + N_{bK} = (R+1)\left(\frac{1}{R}\, p_a^*\,(1-p_a^*) + p_b^*\,(1-p_b^*)\right)\, I_X$$

For $R = 1$, the two sample sizes are equal,

$$N_{aK} = N_{bK} = \frac{N_X}{2} = \left(p_a^*\,(1-p_a^*) + p_b^*\,(1-p_b^*)\right)\, I_X$$

You can specify the MODEL=TWOSAMPLEFREQ(TEST=PROP WEIGHT=R) option in the SAMPLESIZE statement to compute the required total sample size and individual sample size at each stage. A procedure such as PROC GENMOD with the default DIST=NORMAL option in the MODEL statement can be used to derive the two-sample Z test for proportion difference.

Test for Two Binomial Proportions with a Log Odds Ratio Statistic

The MODEL=TWOSAMPLEFREQ(TEST=LOGOR) option in the SAMPLESIZE statement derives the sample size required to test two binomial proportions by using a log odds ratio statistic. The odds ratio is the ratio of the odds in one group to the odds in the other group, and the log odds ratio is the logarithm of the odds ratio

$$\theta = \log\left(\frac{p_a/(1-p_a)}{p_b/(1-p_b)}\right) = \log\left(\frac{p_a(1-p_b)}{p_b(1-p_a)}\right)$$

The hypothesis of no difference between two proportions, $p_a = p_b$, can be tested through the null hypothesis $H_0 : \theta = 0$, where θ is the log odds ratio. For example, with $H_0 : p_a = p_b = 0.6$ and $H_1 : p_a = 0.8,\ p_b = 0.6$, it corresponds to the equivalent hypothesis $H_0 : \theta = 0$ and $H_1 : \theta = \log\left(\frac{0.8(1-0.6)}{0.6(1-0.8)}\right) = \log(8/3) = 0.98083$.

The maximum likelihood estimate of θ is given by

$$\hat{\theta} = \log\left(\frac{\hat{p}_a(1-\hat{p}_b)}{\hat{p}_b(1-\hat{p}_a)}\right)$$

with an asymptotic variance

$$\mathrm{Var}(\hat{\theta}) = I^{-1} = \frac{1}{N_a p_a(1-p_a)} + \frac{1}{N_b p_b(1-p_b)}$$

where I is the information (Diggle et al. 2002, pp. 341–342). That is, the standardized statistic

$$Z_k = \hat{\theta}_k \sqrt{I_k} \sim N\left(\theta\sqrt{I_k},\ 1\right)$$

In practice, $p_a = \hat{p}_a$ and $p_b = \hat{p}_b$, the estimated sample proportions for groups A and B, respectively, at stage k, can be used to derive the information I_k and the test statistic $Z_k = \hat{\theta}_k\sqrt{I_k}$ if the two sample sizes N_a and N_b are sufficiently large such that the test statistic has an approximately normal distribution.

The maximum information I_X is needed to derive the required sample size. If the maximum information is not specified or derived with the ALTREF= option in the procedure, the PROP= option in the SAMPLESIZE statement is used to provide proportions under the alternative hypothesis for the alternative reference and then to derive the maximum information.

In order to derive the sample sizes N_{ak} and N_{bk} uniquely from the information, $N_{ak} = R\,N_{bk}$ is assumed for $k = 1, 2, \ldots, K$, where $R = w_a/w_b$ is the constant allocation ratio computed from the WEIGHT=w_a w_b option in the SAMPLESIZE statement. Then with

$$I_X = N_{bK}\left(\frac{1}{R\,p_a(1-p_a)} + \frac{1}{p_b(1-p_b)}\right)^{-1}$$

the sample size can be computed.

In PROC SEQDESIGN, the total sample sizes in the two groups are computed as

$$N_{bK} = I_X\left(\frac{1}{R\,p_a^*(1-p_a^*)} + \frac{1}{p_b^*(1-p_b^*)}\right)$$

$$N_{aK} = R\, N_{bK}$$

where $R = w_a/w_b$ is the constant allocation ratio, and p_a^* and p_b^* are proportions specified with the REF= option:

- REF=NULLPROP uses proportions under H_0: $p_a^* = p_{0a}$, $p_b^* = p_{0b}$
- REF=AVGNULLPROP uses the average proportion under H_0: $p_a^* = p_b^* = (Rp_{0a} + p_{0b})/(R+1)$
- REF=PROP uses proportions under H_1: $p_a^* = p_{1a}$, $p_b^* = p_{1b}$
- REF=AVGPROP uses the average proportion under H_1 : $p_a^* = p_b^* = (Rp_{1a} + p_{1b})/(R+1)$

You can specify the MODEL=TWOSAMPLEFREQ(TEST=LOGOR WEIGHT=*R*) option in the SAMPLESIZE statement to compute the required total sample size and individual sample size at each stage. A procedure such as PROC LOGISTIC can be used to derive the log odds ratio statistic.

Test for Two Binomial Proportions with a Log Relative Risk Statistic

The MODEL=TWOSAMPLEFREQ(TEST=LOGRR) option in the SAMPLESIZE statement derives the sample size required to test two binomial proportions by using a log relative risk statistic. The relative risk is the ratio of the proportion in one group to the proportion in the other group. The log relative risk statistic is the logarithm of the relative risk

$$\theta = \log\left(\frac{p_a}{p_b}\right)$$

The hypothesis of no difference between two proportions, $p_a = p_b$, can be tested through the null hypothesis $H_0 : \theta = 0$. For example, with $H_0 : p_a = p_b = 0.6$ and $H_1 : p_a = 0.8,\ p_b = 0.6$, it corresponds to the equivalent hypothesis $H_0 : \theta = 0$ and $H_1 : \theta = \log\left(\frac{0.8}{0.6}\right) = \log(4/3) = 0.28768$.

The maximum likelihood estimate of θ is given by

$$\hat{\theta} = \log\left(\frac{\hat{p}_a}{\hat{p}_b}\right)$$

with an asymptotic variance

$$I^{-1} = \frac{1-p_a}{N_a\, p_a} + \frac{1-p_b}{N_b\, p_b}$$

where I is the information (Chow and Liu 1998, p. 329).

In practice, $p_a = \hat{p}_a$ and $p_b = \hat{p}_b$, the estimated sample proportions for groups A and B, respectively, at stage k, are used to derive the information I_k and the test statistic $Z_k = \hat{\theta}_k\sqrt{I_k}$.

The maximum information I_X and proportions p_a and p_b are needed to derive the required sample size. If the maximum information is not specified or derived with the ALTREF= option in the procedure, the PROP= option in the SAMPLESIZE statement is used to provide proportions under the alternative hypothesis for the alternative reference and then to derive the maximum information.

Note that in order to derive the sample sizes N_{ak} and N_{bk} uniquely from the information, $N_{ak} = R\,N_{bk}$ is assumed for $k = 1, 2, \ldots, K$, where $R = w_a/w_b$ is the constant allocation ratio computed from the WEIGHT=$w_a\ w_b$ option in the SAMPLESIZE statement. Then the sample size can be computed from

$$I_X = N_{bK}\left(\frac{1-p_a}{R\,p_a} + \frac{1-p_b}{p_b}\right)^{-1}$$

In PROC SEQDESIGN, the computed sample sizes in the two groups are

$$N_{bK} = I_X\left(\frac{1-p_a^*}{R\,p_a^*} + \frac{1-p_b^*}{p_b^*}\right)$$

$$N_{aK} = R\,N_{bK}$$

where $R = w_a/w_b$ is the constant allocation ratio, and p_a^* and p_b^* are proportions specified with the REF= option:

- REF=NULLPROP uses proportions under H_0: $p_a^* = p_{0a}$, $p_b^* = p_{0b}$
- REF=AVGNULLPROP uses the average proportion under H_0: $p_a^* = p_b^* = (Rp_{0a} + p_{0b})/(R+1)$
- REF=PROP uses proportions under H_1: $p_a^* = p_{1a}$, $p_b^* = p_{1b}$
- REF=AVGPROP uses the average proportion under H_1 : $p_a^* = p_b^* = (Rp_{1a} + p_{1b})/(R+1)$

You can specify the MODEL=TWOSAMPLEFREQ(TEST=LOGRR WEIGHT=*R*) option in the SAMPLESIZE statement to compute the required total sample size and individual sample size at each stage. A procedure such as PROC LOGISTIC can be used to derive the log relative risk statistic.

Test for Two Survival Distributions with a Log-Rank Test

The MODEL=TWOSAMPLESURV option in the SAMPLESIZE statement derives the number of events required for a log-rank test of two survival distributions. The analysis of survival data involves the survival times for both censored and uncensored data. A noncensored survival time is the time from treatment to an event such as remission or relapse for an individual. A censored survival time is the time from treatment to the time of analysis for an individual surviving at that time, and the status is unknown beyond that time.

PROC SEQDESIGN computes the number of events by using the approximations described in Schoenfeld (1981). It differs from PROC POWER, which uses an approach similar to that of Freedman (1982). Thus, even in the case of a one-stage design, the designs that PROC SEQDESIGN and PROC POWER produce do not match exactly. For a comparison of these approximation methods, see Hsieh (1992).

Let T be the random variable of the survival time. Then the survival function

$$S(t) = \Pr(T > t)$$

is the probability that an individual from the population has a survival time that exceeds t. And the hazard function is given by

$$h(t) = \frac{f(t)}{S(t)}$$

where $f(t)$ is the density function of T.

The hazard functions can be used to test the equality of two survival distributions $S_a(t) = S_b(t)$ with the null hypothesis $H_0 : h_a(t) = h_b(t), t > 0$, where $S_a(t)$ and $S_b(t)$ are survival functions for groups A and B, respectively, and $h_a(t)$ and $h_b(t)$ are the corresponding hazard functions.

If the two hazards are proportional, $h_a(t) = \lambda\, h_b(t)$, where λ is a constant, then an equivalent null hypothesis is

$$H_0 : \lambda = \frac{h_a(t)}{h_b(t)} = 1$$

Alternatively, another equivalent null hypothesis is given by

$$H_0 : \theta = -\log(\lambda) = 0$$

Suppose that the hazard rate h is a constant. Then with a specified median survival time T_m, the hazard rate can be derived from the equation

$$e^{-h\,T_m} = \frac{1}{2}$$

Denote the distinct event times at stage k as τ_{kj}, $j = 1, 2, \ldots, t_k$, where t_k is the total number of distinct event times. Then the score statistic is the log-rank statistic (Jennison and Turnbull 2000, pp. 259–261; Whitehead 1997, pp. 36–39)

$$S_k = \sum_{j=1}^{t_k} (d_{akj} - e_{akj})$$

where d_{akj} is the number of events from group A and e_{akj} is the number of expected events from A. The number of expected events from A is computed as

$$e_{akj} = d_{kj} \frac{r_{akj}}{r_{kj}}$$

where d_{kj} is the number of events from both groups, r_{akj} is the number of individuals from the treatment group who survived up to time τ_{kj}, and r_{kj} is the number of individuals from both groups who survived up to time τ_{kj}.

If the number of events d_{kj} is small relative to r_{kj}, the number of individuals survived up to time τ_{kj}, then with a sufficiently large sample size, S_k has an approximately normal distribution

$$S_k \sim N(\theta\, I_k,\ I_k)$$

where the variance of S_k is the estimated information

$$I_k = \sum_{j=1}^{t_k} \frac{r_{akj}\, r_{bkj}\, d_{kj}}{r_{kj}^2}$$

In order to derive the number of events from the information I_k, $N_{ak} = R\, N_{bk}$ is assumed for $k = 1, 2, \ldots, K$, where $R = w_a/w_b$ is the constant allocation ratio computed from the WEIGHT=$w_a\ w_b$ option in the SAMPLESIZE statement.

The maximum information I_X is needed to derive the required sample size. If the maximum information is specified or derived with the ALTREF= option in the procedure, the HAZARD=, MEDSURVTIME=, and HAZARDRATIO= options are not applicable. Otherwise, the HAZARD=, MEDSURVTIME=, or HAZARDRATIO= option is used to compute the alternative reference and then to derive the maximum information for the sample size calculation.

With $N_{aK} = R\,N_{bK}$, if the number of events is few relative to the number of individuals who survived, then $r_{aKj} \approx R\,r_{bKj}$, and

$$I_X \approx \sum_{j=1}^{t_K} \frac{R}{(R+1)^2} d_{Kj} = \frac{R}{(R+1)^2}\,D_X$$

where D_X is the total number of events.

Thus, the required total number of events

$$D_X = \frac{(R+1)^2}{R}\,I_X$$

For a study group, if the hazard rate is constant, corresponding to an exponential survival distribution, and the individual accrual is uniform in the accrual time T_a with a constant accrual rate r_a, then the required total sample size and sample size at each stage can be derived. See the section "Input Number of Events for Fixed-Sample Design" on page 9187 for a detailed description of the sample size computation that uses hazard rates, accrual rate, and accrual time.

You can specify the MODEL=TWOSAMPLESURVIVAL option in the SAMPLESIZE statement to compute the required total number of events and individual number of events at each stage. With the specifications of hazard rates, accrual rate, and accrual time, the required total sample size and individual sample size at each stage can also be derived. If the REF=NULLHAZARD option is specified, the hazard rates under the null hypothesis, h_{0a} and h_{0b}, are used in the sample size computation. Otherwise, the hazard rates under the alternative hypothesis, h_{1a} and h_{1b}, are used. A procedure such as PROC LIFETEST can be used to derive the log-rank statistic.

Applicable Regression Parameter Tests and Sample Size Computation

The SEQDESIGN procedure provides sample size computation for tests of a regression parameter in three regression models: normal regression, logistic regression, and proportional hazards regression.

To test a parameter β_1 in a regression model, the variance of the parameter estimate $\hat{\beta}_1$ is needed for the sample size computation. In a simple regression model with one covariate X1, the variance of $\hat{\beta}_1$ is inversely related to the variance of X1, σ_x^2. That is,

$$\mathrm{Var}(\hat{\beta}_1) \propto \frac{1}{N\,\sigma_x^2}$$

for the normal regression and logistic regression models, where N is the sample size, and

$$\mathrm{Var}(\hat{\beta}_1) \propto \frac{1}{D\,\sigma_x^2}$$

for the proportional hazards regression model, where D is the number of events.

For a regression model with more than one covariate, the variance of $\hat{\beta}_1$ for the normal regression and logistic regression models is inversely related to the variance of X1 after adjusting for other covariates. That is,

$$\mathrm{Var}(\hat{\beta}_1) \propto \frac{1}{N\,(1-r_x^2)\,\sigma_x^2}$$

where $\hat{\beta}_1$ is the estimate of the parameter β_1 in the model and r_x^2 is the R square from the regression of X1 on other covariates—that is, the proportion of the variance σ_x^2 explained by these covariates.

Similarly, for a proportional hazards regression model,

$$\mathrm{Var}(\hat{\beta}_1) \propto \frac{1}{D\,(1-r_x^2)\,\sigma_x^2}$$

Thus, with the derived maximum information, the required sample size or number of events can also be computed for the testing of a parameter in a regression model with covariates.

Test for a Parameter in the Regression Model

The MODEL=REG option in the SAMPLESIZE statement derives the sample size required for a Z test of a normal regression. For a normal linear regression model, the response variable is normally distributed with the mean equal to a linear function of the explanatory variables and the constant variance σ^2.

The normal linear model is

$$\mathbf{y} \sim N\left(\mathbf{X}\boldsymbol{\beta},\, \sigma_y^2\,\mathbf{I}_{(N)}\right)$$

where $\mathbf{Y}_{(N\times 1)}$ is the vector of the N observed responses, $\mathbf{X}_{(N\times p)}$ is the design matrix for these N observations, $\boldsymbol{\beta}_{(p\times 1)}$ is the parameter vector, and $\mathbf{I}_{(N)}$ is the $(N \times N)$ identity matrix.

The least squares estimate is

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}$$

and is normally distributed with mean $\boldsymbol{\beta}$ and variance

$$\mathrm{Var}(\hat{\boldsymbol{\beta}}) = \sigma_y^2(\mathbf{X}'\mathbf{X})^{-1}$$

For a model with only one covariate X1,

$$\hat{\beta}_1 \sim N\left(\beta_1,\, \mathrm{Var}(\hat{\beta}_1)\right)$$

where the variance

$$\mathrm{Var}(\hat{\beta}_1) = I_{\beta_1}^{-1} = \sigma_y^2\,\frac{1}{N\,\sigma_x^2}$$

Thus, with the derived maximum information $I_X = I_{\beta_1}$, the required sample size is given by

$$N = I_X\,\frac{\sigma_y^2}{\sigma_x^2}$$

For a normal linear model with more than one covariate, the variance of a single parameter β_1 is

$$\mathrm{Var}(\hat{\beta}_1) = \sigma_y^2\,(\mathbf{X}'\mathbf{X})^{-1}_{(11)} = \sigma_y^2\,\frac{1}{N\,\sigma_x^2\,(1-r_x^2)}$$

where $(\mathbf{X}'\mathbf{X})^{-1}_{(11)}$ is the diagonal element of the $(\mathbf{X}'\mathbf{X})^{-1}$ matrix corresponding to the parameter β_1, σ_x^2 is the variance of the variable X1, and r_x^2 is the proportion of variance of X1 explained by other covariates. The value $\sigma_x^2\,(1-r_x^2)$ represents the variance of X1 after adjusting for all other covariates.

Thus, with the derived maximum information I_X, the required sample size is

$$N = I_X\,\frac{\sigma_y^2}{(1-r_x^2)\,\sigma_x^2}$$

In the SEQDESIGN procedure, you can specify the MODEL=REG(VARIANCE=σ_y^2 XVARIANCE=σ_x^2 XRSQUARE=r_x^2) option in the SAMPLESIZE statement to compute the required total sample size and individual sample size at each stage. A SAS procedure such as PROC REG can be used to compute the parameter estimate and its standard error at each stage.

Test for a Parameter in the Logistic Regression Model

The MODEL=LOGISTIC option in the SAMPLESIZE statement derives the sample size required for a Z test of a logistic regression parameter. The linear logistic model has the form

$$\mathrm{logit}(p) = \log\left(\frac{p}{1-p}\right) = \mathbf{x}\boldsymbol{\beta}$$

where p is the response probability to be modeled and $\boldsymbol{\beta}$ is a vector of parameters.

Following the derivation in the section "Test for a Parameter in the Regression Model" on page 9204, the required sample size for testing a parameter in $\boldsymbol{\beta}$ is given by

$$N = I_X\,\frac{\sigma_y^2}{(1-r_x^2)\,\sigma_x^2}$$

With the variance of the logit response, $\sigma_y^2 = 1/(p(1-p))$,

$$N = I_X\,\frac{1}{p(1-p)}\,\frac{1}{(1-r_x^2)\,\sigma_x^2}$$

where σ_x^2 is the variance of X and r_x^2 is the proportion of variance explained by other covariates.

In the SEQDESIGN procedure, you can specify the MODEL=LOGISTIC(PROP=p XVARIANCE=σ_x^2 XRSQUARE=r_x^2) option in the SAMPLESIZE statement to compute the required total sample size and individual sample size at each stage.

A SAS procedure such as PROC LOGISTIC can be used to compute the parameter estimate and its standard error at each stage.

Test for a Parameter in the Proportional Hazards Regression Model

The MODEL=PHREG option in the SAMPLESIZE statement derives the number of events required for a Z test of a proportional hazards regression parameter. For analyses of survival data, Cox's semiparametric model is often used to examine the effect of explanatory variables on hazard rates. The survival time of each observation in the population is assumed to follow its own hazard function, $h_i(t)$, expressed as

$$h_i(t) = h(t; \mathbf{X}_i) = h_0(t) \exp(\mathbf{X}_i' \boldsymbol{\beta})$$

where $h_0(t)$ is an arbitrary and unspecified baseline hazard function, $\mathbf{x}_i$ is the vector of explanatory variables for the ith individual, and $\boldsymbol{\beta}$ is the vector of regression parameters associated with the explanatory variables.

Hsieh and Lavori (2000, p. 553) show that the required number of events for testing a parameter in $\boldsymbol{\beta}$, β_1, associated with the variable X1 is given by

$$D_X = I_X \frac{1}{(1 - r_x^2)\ \sigma_x^2}$$

where σ_x^2 is the variance of X1 and r_x^2 is the proportion of variance of X1 explained by other covariates.

In the SEQDESIGN procedure, you can specify the MODEL=PHREG(XVARIANCE=σ_x^2 XRSQUARE=r_x^2) option in the SAMPLESIZE statement to compute the required number of events and individual number of events at each stage.

A SAS procedure such as PROC PHREG can be used to compute the parameter estimate and its standard error at each stage.

Note that for a two-sample test, X1 is an indicator variable and is the only covariate in the model. Thus, if the two sample sizes are equal, then the variance $\sigma_x^2 = 1/4$ and the required number of events for testing the parameter β_1 is given by

$$D_X = I_X \frac{1}{\sigma_x^2} = 4\, I_X$$

See the section "Input Number of Events for Fixed-Sample Design" on page 9187 for a detailed description of the sample size computation that uses hazard rates, accrual rate, and accrual time.

Aspects of Group Sequential Designs

This section summarizes various aspects of group sequential designs that are encountered in applications of the SEQDESIGN procedure. Features are illustrated through two-sided designs with $\alpha = 0.05$ and $\beta = 0.10$. The null hypothesis $H_0 : \theta = 0$ and an alternative reference $\theta_1 = \pm 0.25$ are used for the designs with early stopping only to reject the null hypothesis.

Canonical Joint Distribution

The SEQDESIGN procedure assumes that with a total number of stages K, the sequence of the standardized test statistics $\{Z_1, Z_2, \ldots, Z_K\}$ has the canonical joint distribution with information levels $\{I_1, I_2, \ldots, I_K\}$ for the parameter θ (Jennison and Turnbull 2000, p. 49):

- $(Z_1, Z_2, \ldots, Z_K)$ is multivariate normal
- $Z_k \sim N\left(\theta\sqrt{I_k}, 1\right), \; k = 1, 2, \ldots, K$
- $\mathrm{Cov}(Z_{k_1}, Z_{k_2}) = \sqrt{(I_{k_1}/I_{k_2})}, \quad 1 \le k_1 \le k_2 \le K$

Normality Assumption

The SEQDESIGN procedure derives the boundary values by assuming that the sequence of the standardized test statistics $\{Z_1, Z_2, \ldots, Z_K\}$ has the canonical joint distribution with information levels $\{I_1, I_2, \ldots, I_K\}$ for the parameter θ. If the test statistic Z_k does not have a normal distribution, it is assumed that the test statistic is computed from a large sample, so that the resulting statistic has an approximately normal distribution.

Number of Stages

For group sequential trials with fixed significance level α, power $1 - \beta$, and alternative reference, if the number of stages is increased, the required maximum information is also increased, but the average sample number under the alternative hypothesis is likely to decrease.

For example, for two-sided designs with early stopping only to reject the null hypothesis $H_0 : \theta = 0$, $\alpha = 0.05$, and $\beta = 0.10$ at the alternative reference $\theta_1 = \pm 0.25$, the O'Brien-Fleming method increases the maximum information from 168.12 for a fixed-sample design to 169.32 for a two-stage design, 172.57 for a five-stage design, and then 174.42 for a ten-stage design. In the mean time, the average sample number (as a percentage of fixed-sample) under the alternative hypothesis decreases from 100 for a fixed-sample design to 85.11 for a two-stage design, 75.03 for a five-stage design, and then 71.80 for a ten-stage design. The reduction in average sample number decreases as the number of stages increases. Thus there seems to be little to gain from choosing a design with more than five stages (Pocock 1982, p. 155).

Alternative Reference

The alternative reference θ_1 is the hypothetical reference under the alternative hypothesis at which the power is computed. It is a treatment value that the investigators would hope to detect with high probability (Jennison and Turnbull 2000, p. 21).

For a group sequential design with specified parameters such as α and β errors, the drift parameter $\theta_1\sqrt{I_X}$ is always derived in the SEQDESIGN procedure. Thus, with a smaller alternative reference θ_1, a larger maximum information level I_X is needed. That is, in order to detect a smaller difference with the same high power, a larger sample size is required.

Maximum Information

In a clinical trial, the amount of information about an unknown parameter available from the data can be measured by the Fisher information, the variance of the score statistic. The maximum information is the information level needed at the final stage of the group sequential trial if the trial does not stop at an interim stage. For a group sequential design, the maximum information can be derived with the specified alternative reference.

The maximum information is proportional to the sample size or number of events required for the design. Thus, it can also be used to compare different designs. Generally, a design with a larger probability to stop the trial early tends to have a larger maximum information. For example, for two-sided four-stage designs with early stopping only to reject H_0, $\alpha = 0.05$, and $\beta = 0.10$ at the alternative reference $\theta_1 = \pm 0.25$, the Pocock method has a maximum information of 198.91 and the O'Brien-Fleming method has a maximum information of 171.84, indicating a much larger information level required for the Pocock method.

Drift Parameter

The drift parameter $\theta_1 \sqrt{I_X}$ is derived for each design in the SEQDESIGN procedure, where θ_1 is the alternative reference. It is proportional to the square root of maximum information required for the design and can be used to compare maximum information for different designs with the same alternative reference. For example, for two-sided four-stage designs with early stopping only to reject H_0, $\alpha = 0.05$, and $\beta = 0.10$, the Pocock method has a drift parameter 3.526 and the O'Brien-Fleming method has a drift parameter 3.277, indicating that a larger maximum information level is required for the Pocock method than for the O'Brien-Fleming method.

Average Sample Number

The average sample number is the expected sample size (for nonsurvival data) or expected number of events (for survival data) of the design under a specific hypothetical reference. The percent average sample numbers with respect to the corresponding fixed-sample design are displayed in the SEQDESIGN procedure.

The design that requires a larger maximum information level tends to have a smaller average sample number under the alternative hypothesis. For example, for two-sided four-stage designs with early stopping only to reject H_0, $\alpha = 0.05$, and $\beta = 0.10$ at the alternative reference $\theta_1 = \pm 0.25$, the Pocock design has a maximum information of 198.91 and an average sample number (in percentage of fixed-sample design) of 69.75 under the alternative hypothesis, and the O'Brien-Fleming design has a maximum information of 171.84 and an average sample number of 76.74.

Sample Size

The maximum information for the sequential design expressed as a percentage of its corresponding fixed-sample information is derived in the SEQDESIGN procedure. The sample size or number of events needed for a group sequential trial is computed by multiplying the sample size or number of events for the corresponding fixed-sample design by the derived percentage.

If the sample size or number of events for the fixed-sample design is available, you can use the MODEL=INPUTNOBS or MODEL=INPUTNEVENTS option in the SAMPLESIZE statement to derive the sample size or number of events needed at each stage. Otherwise, with the specified or derived maximum information, you can use the MODEL= option in the SAMPLESIZE statement to specify a hypothesis test and then to derive the sample size or number of events needed at each stage. See the section "Sample Size Computation" on page 9185 for the sample size computation for commonly used tests.

Summary of Methods in Group Sequential Designs

There are three different types of methods available in the SEQDESIGN procedure: fixed boundary shape methods for specified boundary shape, Whitehead methods for boundaries from continuous monitoring, and error spending methods for specified error spending at each stage.

The fixed boundary shape methods include unified family methods and Haybittle-Peto methods. The unified family methods include Pocock, O'Brien-Fleming, power family, and triangular methods.

Pocock Method

Pocock derives the constant boundary on the standardized Z scale to demonstrate the sequential design while maintaining the overall α and β levels (Pocock 1977). The resulting boundary tends to stop the trials early with a larger p-value. This boundary is commonly called a Pocock boundary, but Pocock himself does not advocate these boundary values for stopping a trial early to reject the null hypothesis, because large p-values might not be persuasive enough (Pocock and White 1999). Also, the nominal p-value at the final stage is much smaller than the overall p-value of the design. That is, the trial might stop at the final stage with a small nominal p-value, but the test is not rejected, which might not be easy to justify.

O'Brien-Fleming Method

O'Brien-Fleming boundary values are inversely proportional to the square root of information levels on the standardized Z scale (O'Brien and Fleming 1979). The O'Brien-Fleming boundary is conservative in the early stages and tends to stop the trials early only with a small p-value. But the nominal value at the final stage is close to the overall p-value of the design.

Power Family Method

The power family method (Wang and Tsiatis 1987; Emerson and Fleming 1989; Pampallona and Tsiatis 1994) generalizes the Pocock and O'Brien-Fleming methods with a power parameter to allow continuous movement between the Pocock and O'Brien-Fleming methods. The power parameter is $\rho = 0$ for the Pocock method and $\rho = 0.5$ for the O'Brien-Fleming method.

Triangular Method

The unified family triangular method (Kittelson and Emerson 1999) contains straight-line boundaries on the score scale. For a one-sided trial with early stopping either to reject and to accept the null hypothesis, the method produces a triangular continuation region. The boundary shape is specified with the slope parameter τ.

Unified Family Method

The unified family method (Kittelson and Emerson 1999) extends power family methods to incorporate the triangular method, which contains straight-line boundaries on the score scale.

Haybittle-Peto Method

The Haybittle-Peto method (Haybittle 1971; Peto et al. 1976) uses a Z value of 3 for the critical values in interim stages and derives the critical value at the final stage. With this method, the final-stage critical value is close to the original design without interim monitoring. The SEQDESIGN procedure extends this method further to allow for different Z or nominal p-values for the boundaries.

Whitehead Method

Whitehead methods (Whitehead and Stratton 1983; Whitehead 1997, 2001) derive the boundary values by adapting the continuous monitoring tests to the discrete monitoring of group sequential tests. With early stopping to reject or accept the null hypothesis in a one-sided test, the derived continuation region has a triangular shape on the score-scaled boundaries. Only elementary calculations are needed to derive the boundary values in Whitehead's triangular methods. The resulting Type I error probability and power are extremely close but differ slightly from the specified values due to the approximations used in deriving the tests (Jennison and Turnbull 2000, p. 106). The SEQDESIGN procedure provides the BOUNDARYKEY= option to adjust the boundary value at the final stage for the exact Type I or Type II error probability levels.

Error Spending Method

The error spending method uses the specified α and β errors to be used at each stage of the design to derive the boundary values.

Error Spending Function Method

The error spending function method uses the error spending function to compute the α and β errors to be used at each stage of the design and then to derive the boundary values for these errors. The following four error spending functions are available in the SEQDESIGN procedure:

- The Pocock-type error spending function (Lan and DeMets 1983) produces boundaries similar to those produced with Pocock's method.
- The O'Brien-Fleming-type error spending function (Lan and DeMets 1983) produces boundaries similar to those produced with the O'Brien-Fleming method.
- The gamma error spending function (Hwang, Shih, and DeCani 1990) specifies a gamma cumulative error spending function indexed by the gamma parameter γ. The boundaries created with $\gamma = 1$ are similar to the boundaries from the Pocock method, and the boundaries created with $\gamma = -4$ or $\gamma = -5$ are similar to the boundaries from the O'Brien-Fleming method.
- The power error spending function (Jennison and Turnbull 2000, p. 148) specifies a power cumulative error spending function indexed by the power parameter ρ. The boundaries created with $\rho = 1$ are similar to the boundaries from the Pocock method, and the boundaries created with $\rho = 3$ are similar to the boundaries from the O'Brien-Fleming method.

Table Output

For each design, the SEQDESIGN procedure displays the "Design Information," "Method Information," and "Boundary Information" tables by default.

Boundary Information and Ceiling-Adjusted Design Boundary Information

The "Boundary Information" and "Ceiling-Adjusted Design Boundary Information" tables display the following information at each stage:

- Information Level Proportion, the information proportion
- Actual Information Level, if the maximum information is either specified or derived
- N, the required sample size for nonsurvival data, if the SAMPLESIZE statement is specified
- Events, the required number of events for survival data, if the SAMPLESIZE statement is specified
- Alternative References, alternative references that are displayed using the specified statistic scale (if a p-value scale is specified, the standardized Z scale is used)
- Boundary Values, boundary values that are displayed using the specified statistic scale to reject or accept the null hypothesis

The "Boundary Information" table displays the boundary information of the specified design, and the "Ceiling-Adjusted Design Boundary Information" table displays the boundary information for the design that corresponds to the integer-valued sample sizes at the stages for the nonsurvival data, and the integer-valued times or sample sizes at the stages for the survival data.

Implicitly, the boundary information table also contains the variables for the boundary scale, the stopping criterion, and the type of alternative hypothesis. That is, if an ODS statement is used to save the table, the data set also contains the variables _Scale_ for the boundary scale, _Stop_ for the stopping criterion, and _ALT_ for the type of alternative hypothesis.

Design Information

The "Design Information" table displays the following design information:

- Method, the method used to generate boundary values
- Boundary Key, the type of error level to maintain
- Number of Stages, the total number of stages
- Alpha, the Type I error level
- Beta, the Type II error level
- Power, the power of the design

- Max Information (Percent Fixed-Sample), the maximum information for the sequential design, as a percentage of the corresponding fixed-sample information
- Null Ref ASN (Percent Fixed-Sample), the average sample number (expected sample size for nonsurvival data or expected number of events for survival data) that is required under the null hypothesis for the group sequential design, as a percentage of the corresponding fixed-sample design
- Alt Ref ASN (Percent Fixed-Sample), the average sample number that is required under the alternative reference for the group sequential design, as a percentage of the corresponding fixed-sample design

If both the maximum information (MAXINFO= option) and the alternative reference θ_1 (ALTREF= option) are specified in the PROC SEQDESIGN statement, then either the ALPHA= option is used to derive the Type II error probability β (BOUNDARYKEY=ALPHA) or the BETA= option is used to derive the Type I error probability α (BOUNDARYKEY=BETA).

If the CEILADJDESIGN=INCLUDE option is specified in the PROC SEQDESIGN statement, the "Design Information" table also displays Alpha, Beta, Power, Max Information, Null Ref ASN, and Alt Ref ASN for the adjusted design that corresponds to integer sample sizes at the stages for the nonsurvival data, and to integer times or sample sizes at the stages for the survival data.

Error Spending Information

The "Error Spending Information" table displays the following information at each stage:

- proportion of information
- actual information level, if the maximum information is either specified or derived
- cumulative error spending for each boundary

Method Information

The "Method Information" table displays detailed method information for the design. For each boundary, it displays the following:

- the group sequential method used
- the α or β errors
- the specified parameter ρ, if an error spending function is used
- the specified parameters ρ and τ with the derived critical value C, if a unified family method is used
- the alternative reference θ_1, if either the ALTREF= or the MAXINFO= option is specified
- the derived drift parameter, $\theta_1\sqrt{I_X}$, where I_X is the maximum information and θ_1 is the alternative reference

Note that the alternative references are displayed with the MLE scale in the "Method Information" table. In contrast, the alternative references in the "Boundary Information" table are displayed with the specified statistic scale (if the p-value scale is not specified) or the standardized Z scale (if the p-value scale is specified).

Powers and Expected Sample Sizes

The "Powers and Expected Sample Sizes" table displays the following information under each of the specified hypothetical references $\theta = c_i \theta_1$, where θ_1 is the alternative reference and c_i are values specified in the CREF= option.

- coefficient c_i for the hypothetical references. The value $c_i = 0$ corresponds to the null hypothesis and $c_i = 1$ corresponds to the alternative hypothesis
- power
- expected sample size, as percentage of fixed-sample size

For a one-sided design, the power and expected sample sizes under the hypothetical references $\theta = c_i \theta_1$ are displayed.

For a two-sided symmetric design, the power and expected sample sizes under each of the hypothetical references $\theta = c_i \theta_{1u}$ are displayed, where θ_{1u} is the upper alternative reference.

For a two-sided asymmetric design, the power and expected sample sizes under each of the hypothetical references $\theta = c_i \theta_{1l}$ and $\theta = c_i \theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively.

For a two-sided design, the power is the probability of correctly rejecting the null hypothesis for the correct alternative. Thus, under the null hypothesis, the displayed power corresponds to a one-sided Type I error probability level—that is, the lower α level or the upper α level.

The expected sample size as a percentage of the corresponding fixed-sample design is

$$100 \times \frac{\sum_{k=1}^{K} p_k I_k}{I_0}$$

where p_k is the stopping probability at stage k, $\sum_{k=1}^{K} p_k I_k$ is the expected information level, and I_0 is the information level for the fixed-sample design.

Sample Size Summary

When you use the SAMPLESIZE statement with the SEQDESIGN procedure, the "Sample Size Summary" table displays parameters for the sample size computation. It also displays the expected sample sizes or numbers of events for the model under both the null and alternative hypotheses.

The expected sample size is the average sample size

$$\frac{\sum_{k=1}^{K} p_k I_k}{I_0} N_0$$

where p_k is the stopping probability at stage k, $\sum_{k=1}^{K} p_k I_k$ is the expected information level, I_0 is the information level for the fixed-sample design, and N_0 is the sample size for the fixed-sample design.

The expected number of events is the average number of events

$$\frac{\sum_{k=1}^{K} p_k I_k}{I_0} D_0$$

where D_0 is the fixed-sample number of events for the model.

Sample Size Information

The "Sample Sizes (N)" table displays the required sample sizes and information levels at each stage, in both fractional and integer numbers. The derived fractional sample sizes are under the heading "Fractional N." These sample sizes are rounded up to integers under the heading "Ceiling N." The matched integer sample sizes are also displayed for two-sample tests.

The "Required Number of Events (D)" table displays the required number of events required and information level at each stage.

The "Number of Events (D) and Sample Sizes (N)" table displays the number of events and sample size required at each stage with the study time. The derived times under the heading "Fractional Time" are not integers. These times are rounded up to integers under the heading "Ceiling Time."

Stopping Probabilities

The "Expected Cumulative Stopping Probabilities" table displays the following information under each of the specified hypothetical references $\theta = c_i \theta_1$, where c_i are values specified in the CREF= option, and θ_1 is the alternative reference:

- coefficient c_i for the hypothetical references. The value $c_i = 0$ corresponds to the null hypothesis, and $c_i = 1$ corresponds to the alternative hypothesis
- expected stopping stage
- source of the stopping probability: reject H_0 (with STOP=REJECT or STOP=BOTH), accept H_0 (with STOP=ACCEPT or STOP=BOTH), or either reject or accept H_0 (with STOP=BOTH)
- expected cumulative stopping probabilities at each stage

For a one-sided design, the expected cumulative stopping probabilities under the hypothetical references $\theta = c_i \theta_1$ are displayed.

For a two-sided design, the expected cumulative stopping probabilities under each of the hypothetical references $\theta = c_i \theta_{1l}$ and $\theta = c_i \theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively.

Note that for a symmetric two-sided design, only the expected cumulative stopping probabilities under the hypothetical references $\theta = c_i \theta_{1u}$ are derived.

The expected stopping stage is given by $k_0 + d$, where the integer k_0 and the fraction d $(0 \le d < 1)$ are derived from the expected information level equation

$$\sum_{k=1}^{K} p_k \, I_k = I_{k_0} + d \, (I_{(k_0+1)} - I_{k_0})$$

where p_k is the stopping probability at stage k.

For equally spaced information levels, the expected stopping stage is reduced to the weighted average

$$\sum_{k=1}^{K} p_k \, k$$

ODS Table Names

PROC SEQDESIGN assigns a name to each table it creates. You must use these names to reference tables when using the Output Delivery System (ODS). These names are listed in Table 110.12. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 110.12 ODS Tables Produced by PROC SEQDESIGN

ODS Table Name	Description	Statement	Option
AdjustedBoundary	Ceiling-adjusted design boundary values	SAMPLESIZE	MODEL (CEILADJDESIGN=INCLUDE)
Boundary	Boundary values		
Design	Design information		
ErrSpend	Error spending		ERRSPEND
Method	Method information		
PowerSampleSize	Power and expected sample size		PSS
SampleSize	Derived sample sizes	SAMPLESIZE	
SampleSizeSummary	Sample size summary	SAMPLESIZE	
StopProb	Stopping probabilities		STOPPROB

Graphics Output

This section describes the use of ODS for creating graphics with the SEQDESIGN procedure. To request these graphs, ODS Graphics must be enabled and you must specify the associated graphics options in the PROC SEQDESIGN statement. Except for the PLOTS=BOUNDARY option, where a detailed boundary plot is generated for each design separately, each option produces a plot for all designs together. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Sequential ASN Plot

The PLOTS=ASN option displays the average sample numbers (expected sample sizes for nonsurvival data or expected numbers of events for survival data) under various hypothetical references. The average sample numbers are connected for each design, and these connected curves for all designs are displayed in the "Sequential ASN Plot" graph.

For a one-sided design, average sample numbers under the hypothetical references $\theta = c_i\theta_1$ are displayed, where c_i are the values specified in the CREF= option and θ_1 is the alternative reference. The horizontal axis displays the c_i values of these hypothetical references.

For a two-sided design, average sample numbers under each of the hypothetical references $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. The horizontal axis displays $-c_i$ values for lower hypothetical references $\theta = c_i\theta_{1l}$ and c_i values for upper hypothetical references $\theta = c_i\theta_{1u}$.

Note that for a symmetric two-sided design, only average sample numbers under the hypothetical references $\theta = c_i\theta_{1u}$ are derived.

Sequential Boundary Plot

The PLOTS=BOUNDARY option displays boundary values and the acceptance and rejection regions at each stage for each design separately in the "Detailed Boundary Information" graph. The BOUNDARYSCALE= option is used to specify the scale of the boundaries on the vertical axis. The keywords MLE, SCORE, STDZ, and PVALUE in the BOUNDARYSCALE= option correspond to the boundary with the MLE scale, score statistic scale, standardized normal Z scale, and p-value scale, respectively.

The stage numbers are displayed on the horizontal axis. In addition, the HSCALE= option in the PLOTS=BOUNDARY option can be used to specify the scale on the horizontal axis. The keywords INFO and SAMPLESIZE in the HSCALE= option correspond to the information levels and sample sizes, respectively.

Combined Sequential Boundary Plot

The PLOTS=COMBINEDBOUNDARY option displays boundary values. The boundary values are connected for each boundary in each design, and these connected curves for all designs are displayed in the "Sequential Boundary Information" graph. The BOUNDARYSCALE= option is used to specify the scale of the boundaries on the vertical axis. The keywords MLE, SCORE, STDZ, and PVALUE in the BOUNDARYSCALE= option correspond to the boundary with the MLE scale, score statistic scale, standardized normal Z scale, and p-value scale, respectively.

The HSCALE= option in the PLOTS=COMBINEDBOUNDARY option can be used to specify the scale on the horizontal axis. The keywords INFO, SAMPLESIZE, and STAGE in the HSCALE= option correspond to the information levels, sample sizes, and stage numbers, respectively.

Sequential Error Spending Plot

The PLOTS=ERRSPEND option displays the cumulative error spending at each stage on each boundary in the "Sequential Error Spending Plot" graph. A legend table uses the design labels to identify the curves for the corresponding design in the plot. Another legend table uses symbols to identify boundaries in the plot.

Sequential Power Plot

The PLOTS=POWER option displays the powers under various hypothetical references. The powers are connected for each design, and these connected curves for all designs are displayed in the "Sequential Power Plot" graph.

For a one-sided design, powers under hypothetical references $\theta = c_i\theta_1$ are displayed, where c_i are the values specified in the CREF= option and θ_1 is the alternative reference. The horizontal axis displays the c_i values of these hypothetical references.

For a two-sided design, powers under hypothetical references $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. The horizontal axis displays $-c_i$ values for lower hypothetical references $\theta = c_i\theta_{1l}$ and c_i values for upper hypothetical references $\theta = c_i\theta_{1u}$.

Note that for a symmetric two-sided design, only powers under hypothetical references $\theta = c_i\theta_{1u}$ are derived.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC SEQDESIGN assigns a name to each graph it creates. You can use these names to reference the graphs when using ODS. To request these graphs, ODS Graphics must be enabled and you must specify the *options* indicated in Table 110.13.

Table 110.13 Graphs Produced by PROC SEQDESIGN

ODS Graph Name	Plot Description	Option
ASNPlot	Average sample numbers	PLOTS=ASN
BoundaryPlot	Detailed boundary values	PLOTS=BOUNDARY
CombinedBoundaryPlot	Boundary values	PLOTS=COMBINEDBOUNDARY
ErrSpendPlot	Error spending	PLOTS=ERRSPEND
PowerPlot	Power curves	PLOTS=POWER

Examples: SEQDESIGN Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

The following examples demonstrate the usage of group sequential methods. Example 110.1 uses the NSTAGES=1 option to derive boundaries of critical values for a fixed-sample design. The remaining examples use different methods to create boundaries for various group sequential designs.

Example 110.1: Creating Fixed-Sample Designs

(View the complete code for this example (seqdex1.sas) in the example repository.)

This example demonstrates a one-sided fixed-sample design and a two-sided fixed-sample design. The following statements request a fixed-sample design with an upper alternative:

```
ods graphics on;
proc seqdesign pss
                ;
   OneSidedFixedSample: design nstages=1
                               alt=upper
                               alpha=0.025 beta=0.10
                               ;
   samplesize model=onesamplemean(mean=0.25);
run;
```

In the DESIGN statement, the label **OneSidedFixedSample** identifies the design in the output tables. The NSTAGES=1 option specifies that the design has only one stage; this corresponds to a fixed-sample design. In the SEQDESIGN procedure, the null hypothesis for the design is $H_0 : \theta = 0$ and the ALT=UPPER option specifies an upper alternative hypothesis $H_1 : \theta = \theta_1 > 0$. The MEAN=0.25 option in the SAMPLESIZE statement specifies the upper alternative reference $\theta_1 = 0.25$.

The options ALPHA=0.025 and BETA=0.10 specify the Type I error probability level $\alpha = 0.025$ and the Type II error probability level $\beta = 0.10$. That is, the design has a power $1 - \beta = 0.90$ at $\theta_1 = 0.25$.

The "Design Information" table in Output 110.1.1 displays design specifications and the derived statistics such as power. As expected, the derived statistics such as maximum information and average sample number (in percentage of its corresponding fixed-sample information) are 100 for the fixed-sample design (NSTAGES=1). Also, for a fixed-sample design, the STOP= and METHOD= options in the DESIGN statement are not applicable.

Output 110.1.1 One-Sided Fixed-Sample Design Information

The SEQDESIGN Procedure
Design: OneSidedFixedSample

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Alternative Reference	0.25
Number of Stages	1
Alpha	0.025
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	100
Max Information	168.1188
Null Ref ASN (Percent of Fixed Sample)	100
Alt Ref ASN (Percent of Fixed Sample)	100

The "Method Information" table in Output 110.1.2 displays the α and β error levels. It also displays the

derived drift parameter, which is the standardized reference improvement, $\theta_1\sqrt{I_0}$, where θ_1 is the alternative reference and I_0 is the maximum information for the design. If either θ_1 or I_0 is specified, the other statistic is derived in the SEQDESIGN procedure. For a fixed-sample design,

$$\theta_1\sqrt{I_0} = \Phi^{-1}(1-\alpha) + \Phi^{-1}(1-\beta) = \Phi^{-1}(0.975) + \Phi^{-1}(0.90) = 3.2415$$

Output 110.1.2 Method Information

Method Information				
Boundary	Alpha	Beta	Alternative Reference	Drift
Upper Alpha	0.02500	0.10000	0.25	3.241516

The "Boundary Information" table in Output 110.1.3 displays information level, alternative reference, and boundary value at each stage. The information proportion indicates the proportion of maximum information available at the stage. With only one stage for a fixed-sample design, the proportion is 1. With the SAMPLESIZE statement, the required sample size N is also displayed under the heading "Information Level."

Output 110.1.3 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0					
	Information Level			Alternative Reference	Boundary Values
Stage	Proportion	Actual	N	Upper	Upper Alpha
1	1.0000	168.1188	168.1188	3.24152	1.95996

By default (or equivalently if you specify BOUNDARYSCALE=STDZ), output alternative references and boundaries are displayed with the standardized normal Z scale. The alternative reference on the standardized Z scale at stage 1 is given by $\theta_1\sqrt{I_1}$, where I_1 is the information level at stage 1. With a boundary value 1.96, the hypothesis of $\theta = 0$ is rejected if the standardized normal statistic $Z \geq 1.96$.

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.1.4. The boundary values in the "Boundary Information" table in Figure 110.1.3 are displayed in the plot.

Output 110.1.4 Boundary Plot

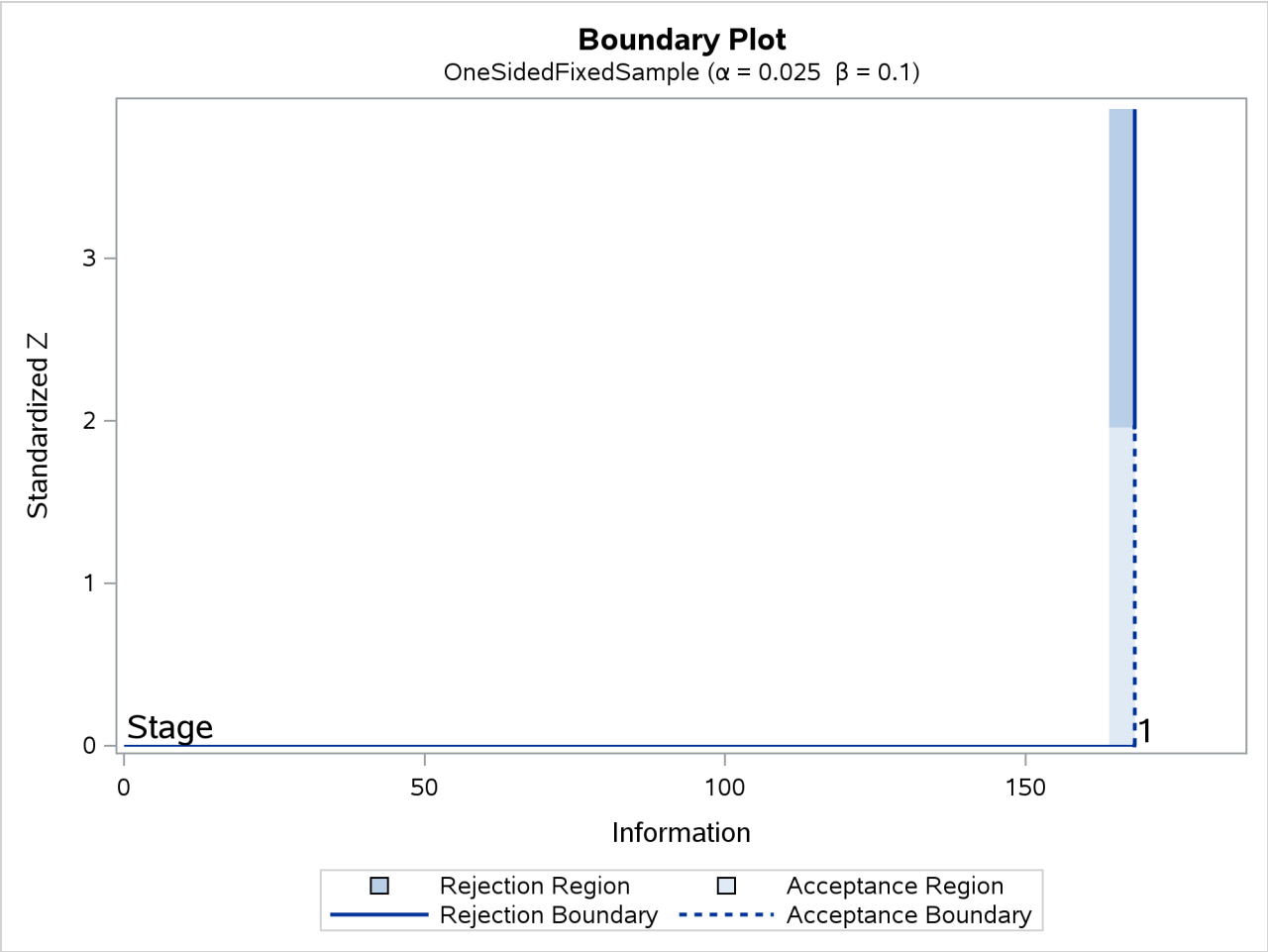

The "Sample Size Summary" table in Output 110.1.5 displays parameters for the sample size computation of the test for a normal mean.

Output 110.1.5 Sample Size Summary

Sample Size Summary	
Test	One-Sample Mean
Mean	0.25
Standard Deviation	1
Max Sample Size	168.1188
Expected Sample Size (Null Ref)	168.1188
Expected Sample Size (Alt Ref)	168.1188

The "Sample Sizes (N)" table in Output 110.1.6 displays the derived sample sizes, in both fractional and integer numbers. With the resulting integer sample sizes, the corresponding information level is slightly larger than the level from the design. This can increase the power slightly if the integer sample size is used in the trial.

Output 110.1.6 Derived Sample Sizes

Sample Sizes (N) One-Sample Z Test for Mean				
	Fractional N		Ceiling N	
Stage	N	Information	N	Information
1	168.12	168.1	169	169.0

The following statements request a two-sided fixed-sample design with a specified alternative reference:

```
ods graphics on;
proc seqdesign altref=1.2
               pss
               ;
   TwoSidedFixedSample: design nstages=1
                               alt=twosided
                               alpha=0.05 beta=0.10
                               ;
   samplesize model=twosamplemean(stddev=2 weight=2);
run;
```

In the SEQDESIGN procedure, the null hypothesis for the design is $H_0 : \theta = 0$. The ALT=TWOSIDED option specifies a two-sided alternative hypothesis $H_1 : \theta = \theta_1 \neq 0$. The ALTREF=1.2 option in the PROC SEQDESIGN statement specifies the alternative reference $\theta_1 = \pm 1.2$.

The ALPHA=0.05 option (which is the default) specifies the two-sided Type I error probability level $\alpha = 0.05$. That is, the lower and upper Type I error probabilities $\alpha_l = \alpha_u = 0.025$. The BETA=0.10 option (which is the default) specifies the Type II error probability level $\beta = 0.10$, and the design has a power $1 - \beta = 0.90$ at the alternative reference $\theta_1 = \pm 1.2$.

The "Design Information" table in Output 110.1.7 displays design specifications and the derived power. With a specified alternative reference, the maximum information is derived.

Output 110.1.7 Two-Sided Fixed-Sample Design Information

The SEQDESIGN Procedure
Design: TwoSidedFixedSample

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Alternative Reference	1.2
Number of Stages	1
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	100
Max Information	7.296822
Null Ref ASN (Percent of Fixed Sample)	100
Alt Ref ASN (Percent of Fixed Sample)	100

The "Method Information" table in Output 110.1.8 displays the α and β errors, alternative references, and drift parameter. For a fixed-sample design, the derived drift parameter

$$\theta_1 \sqrt{I_0} = \Phi^{-1}(1 - \frac{\alpha}{2}) + \Phi^{-1}(1 - \beta) = \Phi^{-1}(0.975) + \Phi^{-1}(0.90) = 3.2415$$

Output 110.1.8 Method Information

Method Information

Boundary	Alpha	Beta	Alternative Reference	Drift
Upper Alpha	0.02500	0.10000	1.2	3.241516
Lower Alpha	0.02500	0.10000	-1.2	-3.24152

With a specified alternative reference $\theta_1 = 1.2$, the maximum information

$$I_0 = \left(\frac{3.2415}{1.2}\right)^2 = 7.2968$$

The default "Boundary Information" table in Output 110.1.9 displays information level, alternative reference, and boundary values. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the alternative reference and boundary values are displayed with the standardized normal Z scale. Thus, the standardized alternative references $\pm\theta_1 \sqrt{I_0}$ are displayed.

Output 110.1.9 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference		Boundary Values	
Stage	Proportion	Actual	N	Lower	Upper	Lower Alpha	Upper Alpha
1	1.0000	7.296822	131.3428	-3.24152	3.24152	-1.95996	1.95996

With boundary values of –1.96 and 1.96, the hypothesis of $\theta = 0$ is rejected if the standardized normal statistic $Z \geq 1.96$ or $Z \leq -1.96$.

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.1.10 . The boundary values in the "Boundary Information" table in Figure 110.1.9 are displayed in the plot.

Output 110.1.10 Boundary Plot

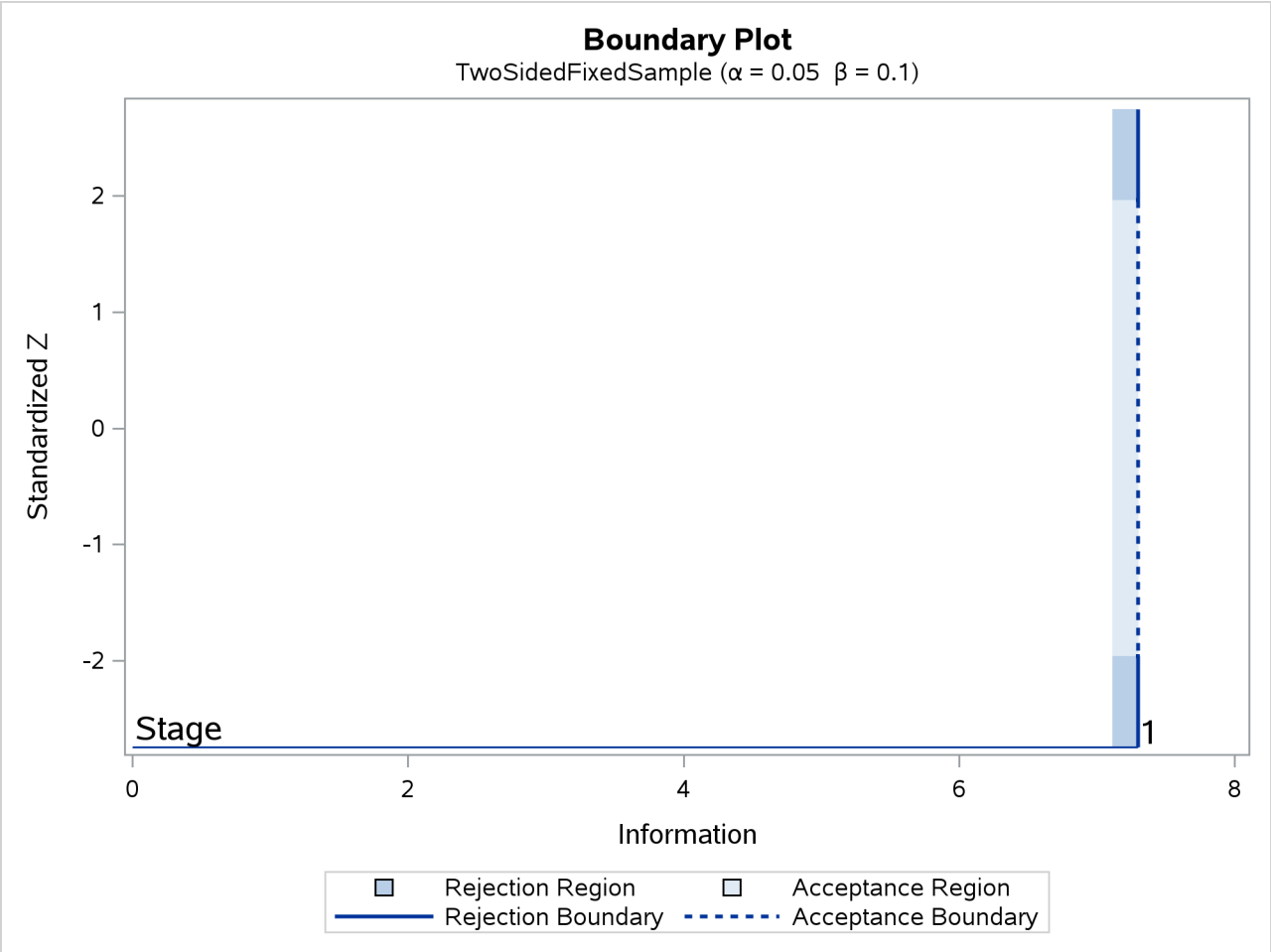

The "Sample Size Summary" table in Output 110.1.11 displays parameters for the sample size computation of the test for a normal mean.

Output 110.1.11 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Means
Mean Difference	1.2
Standard Deviation	2
Max Sample Size	131.3428
Expected Sample Size (Null Ref)	131.3428
Expected Sample Size (Alt Ref)	131.3428
Weight (Group A)	2
Weight (Group B)	1

The "Sample Sizes (N)" table in Output 110.1.12 displays the derived sample sizes, in both fractional and integer numbers. With the WEIGHT=2 option, the allocation ratio is 2 for the first group and 1 for the second group. With the resulting integer sample sizes, the corresponding information level is slightly larger than the level from the design. This can increase the power slightly if the integer sample size is used in the trial.

Output 110.1.12 Derived Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Mean Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	131.34	87.56	43.78	7.2968	132	88	44	7.3333

Example 110.2: Creating a One-Sided O'Brien-Fleming Design

(View the complete code for this example (seqdex2.sas) in the example repository.)

This example demonstrates a group sequential design for a clinical study. A clinic is conducting a study on the effect of vitamin C supplements in treating flu symptoms. The study groups consist of patients in the clinic with their first sign of flu symptoms within the last 24 hours. These individuals are randomly assigned to either the control group, which receives the placebo pills, or the treatment group, which receives large doses of vitamin C supplements. At the end of a five-day period, the flu symptoms of each individual are recorded.

Suppose that from past experience, 60% of individuals experiencing flu symptoms have the symptoms disappeared within five days. The clinic wants to detect a 75% symptoms disappearance with a high probability in the trial. A test that compares the proportions directly is to specify a null hypothesis $H_0 : \theta = p_a - p_b = 0$ with a Type I error probability level $\alpha = 0.025$, where p_a and p_b are the proportions of symptoms' disappearance in the treatment group and control group, respectively. A one-sided alternative $H_1 : \theta > 0$ is also specified with a power of $1 - \beta = 0.80$ at $H_1 : \theta = 0.15$.

For a one-sided fixed-sample design, the critical value for the standardized Z test statistic is given by $C_\alpha = \Phi^{-1}(1-\alpha) = 1.96$. That is, at the end of study, if the test statistic $z \geq C_\alpha$, then the null hypothesis is rejected and the efficacy of vitamin C supplements is declared. Otherwise, the null hypothesis is not rejected and the effect of vitamin C supplements is not significant.

To achieve a $1 - \beta = 0.80$ power at $H_1 : \theta = 0.15$ for a fixed-sample design, the information required is given by

$$I_0 = \frac{(\Phi^{-1}(1-\alpha) + \Phi^{-1}(1-\beta))^2}{0.15^2} = \frac{(1.96 + 0.8416)^2}{0.0225} = 348.84$$

With an equal sample size on the treatment and control groups, $N_a = N_b$, the sample size required for each group under H_1 is computed from the information I_0:

$$N_a = N_b = (p_{1a}\,(1 - p_{1a}) + p_{1b}\,(1 - p_{1b}))\ I_0$$

where $p_{1a} = 0.75$ and $p_{1b} = 0.60$ are proportions in the treatment and control groups under H_1. That is,

$$N_a = N_b = (0.75 \times 0.25 + 0.6 \times 0.4) \times 348.84 = 149.13$$

Thus, 150 individuals are required for each group in the fixed-sample study. See the section "Test for the Difference between Two Binomial Proportions" on page 9197 for a detailed derivation of these required sample sizes.

Instead of a fixed-sample design for the trial, a group sequential design is used to stop the trial early for ethical concerns of possible harm or an unexpected strong efficacy outcome of the new drug. It can also save time and resources in the process. The following statements invoke the SEQDESIGN procedure and request a four-stage group sequential design that uses an O'Brien-Fleming method for normally distributed statistics. The design uses a one-sided alternative hypothesis H_1 with early stopping to reject or accept H_0.

```
ods graphics on;
proc seqdesign altref=0.15
               ;
   OneSidedOBrienFleming: design nstages=4
                          method=obf
                          alt=upper   stop=both
                          alpha=0.025 beta=0.20
                          ;
   samplesize model(ceiladjdesign=include)
              =twosamplefreq(nullprop=0.6 test=prop);
ods output AdjustedBoundary=Bnd_Prop;
run;
```

At each interim stage in a sequential design, a hypothesis can be rejected, accepted, or continued to the next time point. The STOP=BOTH option specifies early stopping to reject or accept the null hypothesis. The CEILADJDESIGN=INCLUDE option creates an additional design that corresponds to the integer-valued sample sizes at the stages. The option adds this ceiling-adjusted design information to the design information table and displays the ceiling-adjusted design boundary information in a separate boundary information table.

The ADJUSTEDBOUNDARY=BND_PROP option in the ODS OUTPUT statement creates an output data set that contains the resulting ceiling-adjusted boundary information. After the actual data from the clinical trial are collected and analyzed at each stage by a procedure such as PROC GENMOD, you can use the SEQTEST procedure to test the resulting statistics at stage 1 by using the boundary information that is stored in the BOUND_PROP data set.

The "Design Information," "Method Information," and "Boundary Information" tables are displayed by default. The "Design Information" table in Output 110.2.1 displays design specifications and derived statistics such as power and maximum information. If you specify an alternative reference (for example, ALTREF=0.15), the maximum information I_X is derived.

Output 110.2.1 Design Information

The SEQDESIGN Procedure
Design: OneSidedOBrienFleming

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Method	O'Brien-Fleming
Boundary Key	Both
Alternative Reference	0.15
Number of Stages	4
Alpha	0.025
Beta	0.2
Power	0.8
Max Information (Percent of Fixed Sample)	111.5566
Max Information	389.1522
Null Ref ASN (Percent of Fixed Sample)	55.96565
Alt Ref ASN (Percent of Fixed Sample)	79.72258
Adj Design Alpha	0.025
Adj Design Beta	0.19835
Adj Design Power	0.80165
Adj Design Max Information (Percent of Fixed Sample)	111.5115
Adj Design Max Information	390.6433
Adj Design Null Ref ASN (Percent of Fixed Sample)	56.04399
Adj Design Alt Ref ASN (Percent of Fixed Sample)	79.62143

The Max Information (Percent Fixed-Sample) is the ratio in percentage between the maximum information for the group sequential design and the information required for a corresponding fixed-sample design:

$$100 \times \frac{I_X}{I_0} = 100 \times \frac{389.15}{348.84} = 111.56$$

That is, if the group sequential trial does not stop at any interim stages, the information needed is 11.56% more than is needed for the corresponding fixed-sample design. For a two-sample test for binomial proportions, the information is proportional to the sample size. Thus, 11.56% more observations are needed for the group sequential trial.

The Null Ref ASN (Percent Fixed-Sample) is the ratio in percentage between the expected sample size required under the null hypothesis for the group sequential design and the sample size required for the corresponding fixed-sample design. With a ratio of 56%, the expected sample size for the group sequential trial under the null hypothesis is 56% of the sample size in the corresponding fixed-sample design.

Similarly, the Alt Ref ASN (Percent Fixed-Sample) is the ratio in percentage between the expected sample size required under the alternative hypothesis for the group sequential design and the sample size required for the corresponding fixed-sample design. With a ratio of 79.7%, the expected sample size for the group sequential trial under the alternative hypothesis is 79.7% of the sample size in the corresponding fixed-sample design.

The CEILADJDESIGN=INCLUDE option creates an adjusted design that has ceiling sample sizes at the stages. The "Design Information" table also includes design information for this adjusted design. Only the Type I error level is maintained for the adjusted design when BOUNDARYKEY=BOTH (the default) in the DESIGN statement.

For a one-sided design that has an upper alternative and early stopping to reject or accept the null hypothesis, upper α and β boundaries are created. The "Method Information" table in Output 110.2.2 displays the Type I error probability α, the Type II error probability β, and the derived drift parameter. The drift parameter is the standardized reference improvement between the alternative and null hypotheses at the final stage. It is also the standardized alternative reference at the final stage if the null reference is zero.

Output 110.2.2 Method Information

Method Information

				Unified Family				
Boundary	**Method**	**Alpha**	**Beta**	**Rho**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	O'Brien-Fleming	0.02500	.	0.5	0	1.94947	0.15	2.959041
Upper Beta	O'Brien-Fleming	.	0.20000	0.5	0	1.00957	0.15	2.959041

With the METHOD=OBF option, the O'Brien-Fleming method is used for each boundary. The O'Brien-Fleming method is one of the unified family methods, and the "Method Information" table displays the corresponding parameter ρ in the unified family method. The table also displays the critical values $C_\alpha = 1.9495$ for the α boundary and $C_\beta = 1.0096$ for the β boundary. These critical values are used to create the boundary values.

The "Boundary Information" table in Output 110.2.3 displays information level, alternative reference, and boundary values at each stage. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the alternative references and boundary values are displayed with the standardized Z statistic scale. The resulting standardized alternative reference at stage k is given by $\theta_1\sqrt{I_k}$, where θ_1 is the alternative reference and I_k is the information level at stage k, $k = 1, 2, 3, 4$.

Output 110.2.3 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference	Boundary Values	
					Upper	
Stage	**Proportion**	**Actual**	**N**	**Upper**	**Beta**	**Alpha**
1	0.2500	97.28805	83.18128	1.47952	-0.53963	3.89893
2	0.5000	194.5761	166.3626	2.09236	0.66460	2.75696
3	0.7500	291.8641	249.5438	2.56260	1.39685	2.25105
4	1.0000	389.1522	332.7251	2.95904	1.94947	1.94947

By default (or equivalently if you specify INFO=EQUAL), equally spaced information levels are used. An information proportion is the proportion of maximum information available at each stage. With the derived maximum information, the actual information level at each stage is also displayed. With the SAMPLESIZE statement, the required sample size N is also displayed under the heading "Information Level."

At each interim stage, if the standardized Z test statistic is larger than or equal to the corresponding upper α

boundary, then the hypothesis $H_0 : \theta = 0$ is rejected. If the test statistic is less than the corresponding upper β boundary, then the trial is stopped and the hypothesis H_0 is accepted. Otherwise, the process continues to the next stage. At the final stage, stage 4, the trial stops and the hypothesis H_0 is rejected if the standardized Z statistic $Z_4 \geq 1.94947$. Otherwise, the trial is accepted.

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.2.4.

Output 110.2.4 Boundary Plot

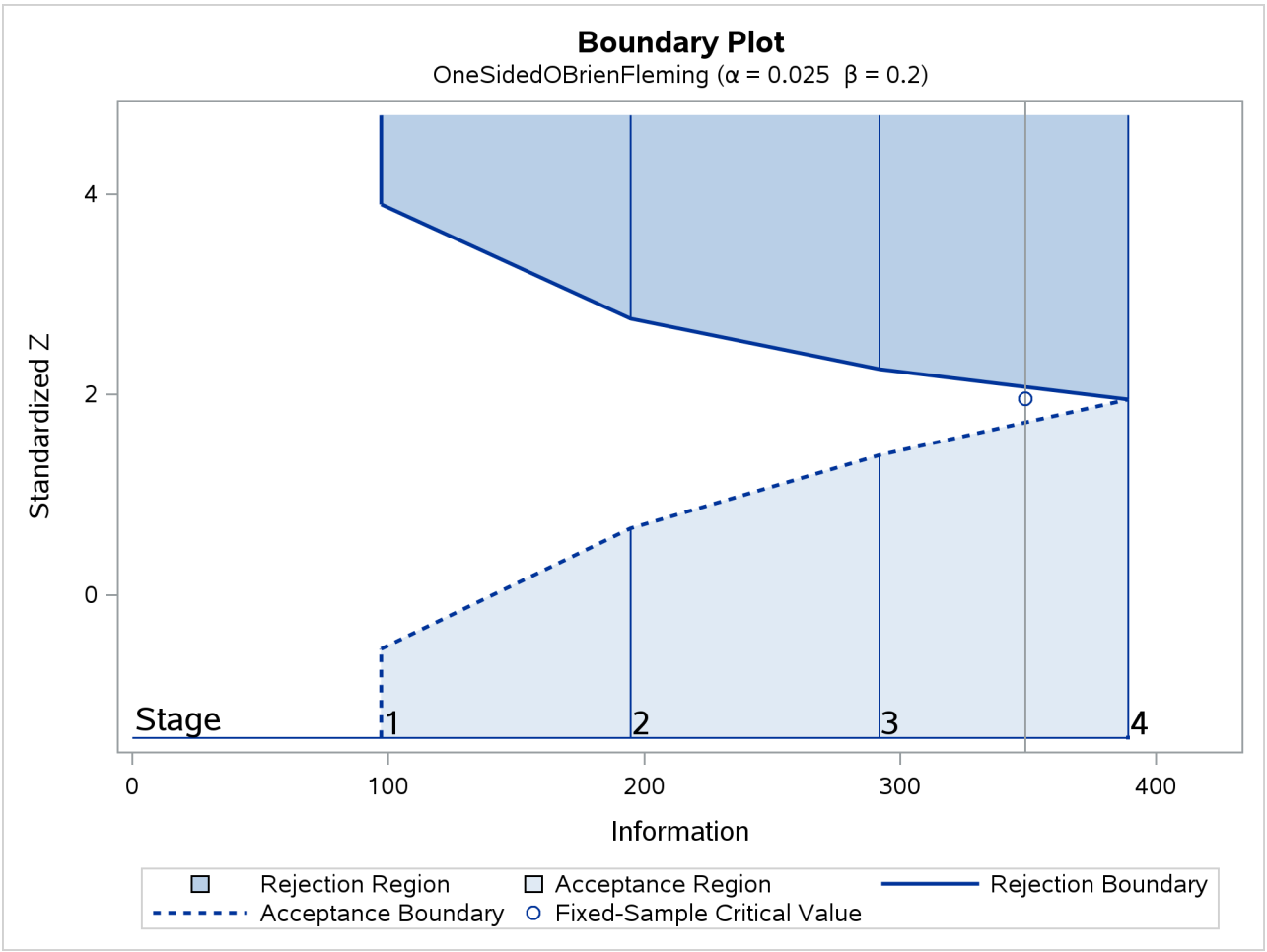

The horizontal axis indicates the information levels for the design. The stages are indicated by vertical lines with accompanying stage numbers. If at any stage a test statistic is in a rejection region, the trial stops and the hypothesis is rejected. If a test statistic is in an acceptance region, then the trial also stops and the hypothesis is accepted. If the statistic is not in a rejection region or an acceptance region, the trial continues to the next stage. The boundary plot also displays the information level and the critical value for the corresponding fixed-sample design.

The SEQDESIGN procedure derives the drift parameter $\theta_1 \sqrt{I_X}$, where θ_1 is the alternative reference and I_X is the maximum information. If either θ_1 or I_X is specified, the other can be derived. With the SAMPLESIZE statement, the maximum information is used to compute the required sample size for the study.

The "Sample Size Summary" table in Output 110.2.5 displays parameters for the sample size computation.

With the MODEL=TWOSAMPLEFREQ(NULLPROP=0.6 TEST=PROP) option in the SAMPLESIZE statement, the total sample size in each group for testing the difference between two proportions is computed. By default (or equivalently if you specify REF=PROP in the MODEL=TWOSAMPLEFREQ option), the required sample sizes are computed under the alternative hypothesis. That is,

$$N_a = N_b = (\, p_{1a}\,(1 - p_{1a}) + p_{1b}\,(1 - p_{1b})\,)\; I_X$$

where $p_{1b} = 0.60$ and $p_{1a} = p_{1b} + \theta_1 = 0.75$ are the proportions in the control and treatment groups, respectively, under the alternative hypothesis H_1. See the section "Test for the Difference between Two Binomial Proportions" on page 9197 for a detailed description of these parameters.

Output 110.2.5 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Proportions
Null Proportion	0.6
Proportion (Group A)	0.75
Test Statistic	Z for Proportion
Reference Proportions	Alt Ref
Max Sample Size	332.7251
Expected Sample Size (Null Ref)	166.9213
Expected Sample Size (Alt Ref)	237.7779

The "Sample Sizes (N)" table in Output 110.2.6 displays the required sample sizes at each stage, in both fractional and integer numbers. The derived fractional sample sizes are under the heading "Fractional N." These sample sizes are rounded up to integers under the heading "Ceiling N." In practice, integer sample sizes are used, and the resulting information levels increase slightly. Thus, 42, 84, 125, and 167 individuals are needed in each group for the four stages, respectively.

Output 110.2.6 Derived Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Proportion Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	83.18	41.59	41.59	97.2880	84	42	42	98.2456
2	166.36	83.18	83.18	194.6	168	84	84	196.5
3	249.54	124.77	124.77	291.9	250	125	125	292.4
4	332.73	166.36	166.36	389.2	334	167	167	390.6

When CEILADJDESIGN=INCLUDE in the SAMPLESIZE statement, the "Ceiling-Adjusted Design Boundary Information" table in Figure 110.2.7 displays boundary information, similar to Figure 110.2.3 but with ceiling sample sizes at the stages.

Output 110.2.7 Adjusted O'Brien-Fleming Boundary Information

Ceiling-Adjusted Design Boundary Information (Standardized Z Scale) Null Reference = 0						
				Alternative	Boundary Values	
	Information Level			Reference	Upper	
Stage	Proportion	Actual	N	Upper	Beta	Alpha
1	0.2515	98.24561	84	1.48678	-0.53660	3.88835
2	0.5030	196.4912	168	2.10263	0.67188	2.74948
3	0.7485	292.3977	250	2.56495	1.39208	2.25390
4	1.0000	390.6433	334	2.96470	1.94999	1.94999

Because the sample sizes have integer values, the information levels at the stages are not equally spaced in this example, but the design is still an O'Brien-Fleming design.

The CEILADJDESIGN=INCLUDE option displays the adjusted-design information in the "Design Information" table and the adjusted boundary information in the "Ceiling-Adjusted Design Boundary Information" table. You can use the SEQTEST procedure to create other tables, such as error spending tables. For example, when you specify ADJUSTEDBOUNDARY=BND_PROP in the ODS OUTPUT statement, the following statements create the error-spending information table for the adjusted design:

```
proc seqtest Boundary=Bnd_Prop
             errspend
             ;
run;
```

Example 110.3: Creating Two-Sided Pocock and O'Brien-Fleming Designs

(View the complete code for this example (seqdex3.sas) in the example repository.)

This example requests two 4-stage group sequential designs for normally distributed statistics with equally spaced information levels at all stages. One design uses Pocock's method and the other uses the O'Brien-Fleming method. The following statements invoke the SEQDESIGN procedure and request these two designs:

```
proc seqdesign altref=0.4
               pss
               stopprob
               errspend
               ;
   TwoSidedPocock:        design nstages=4 method=poc;
   TwoSidedOBrienFleming: design nstages=4 method=obf;
   samplesize model=twosamplemean(stddev=0.8 weight=2);
run;
```

By default (or equivalently if you specify ALT=TWOSIDED and STOP=REJECT in the DESIGN statement), each design has a null hypothesis H_0 with a two-sided alternative with early stopping to reject H_0.

The "Design Information" table in Output 110.3.1 displays design specifications and derived statistics for the Pocock's design. With the specified ALTREF= option, the maximum information $I_X = 77.6984$ is also derived.

Output 110.3.1 Pocock Design Information

The SEQDESIGN Procedure
Design: TwoSidedPocock

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	Pocock
Boundary Key	Both
Alternative Reference	0.4
Number of Stages	4
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	118.3143
Max Information	77.69844
Null Ref ASN (Percent of Fixed Sample)	115.6074
Alt Ref ASN (Percent of Fixed Sample)	69.74805

With the corresponding fixed-sample information

$$I_0 = \frac{(\Phi^{-1}(1-\alpha/2) + \Phi^{-1}(1-\beta))^2}{0.4^2} = \frac{(1.96 + 1.28155)^2}{0.16} = 65.6728$$

the fixed-sample information ratio is $77.6984/65.6728 = 1.1831$.

For a two-sided design with early stopping to reject the null hypothesis, lower and upper α boundaries are created. The "Method Information" table in Output 110.3.2 displays the α and β errors, alternative references, and derived drift parameters, which are the standardized alternative references at the final stage.

Output 110.3.2 Method Information

Method Information								
				Unified Family				
Boundary	**Method**	**Alpha**	**Beta**	**Rho**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	Pocock	0.02500	0.10000	0	0	2.36129	0.4	3.525869
Lower Alpha	Pocock	0.02500	0.10000	0	0	2.36129	-0.4	-3.52587

With the METHOD=POC option, the Pocock method is used for each boundary. The Pocock method is one of the unified family methods, and the table also displays its corresponding parameters $\rho = 0$ as a unified family method and the derived parameters $C = 2.3613$ for the boundary values.

With the PSS option, the "Power and Expected Sample Sizes" table in Output 110.3.3 displays powers and expected sample sizes under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and c_i are values specified in the CREF= option. By default, $c_i = 0, 0.5, 1.0, 1.5$.

Output 110.3.3 Power and Expected Sample Size Information

Powers and Expected Sample Sizes Reference = CRef * (Alt Reference)		
		Sample Size
CRef	Power	Percent Fixed-Sample
0.0000	0.02500	115.6074
0.5000	0.34252	104.0615
1.0000	0.90000	69.7480
1.5000	0.99869	43.6600

Note that at $c_i = 0$, the null reference $\theta = 0$, and the power 0.025 corresponds to the one-sided Type I error probability 0.025. At $c_i = 1$, $\theta = \theta_1$, the power 0.9 is the power of the design. The expected sample sizes are displayed in a percentage scale to its corresponding fixed-sample size design. With the specified SAMPLESIZE statement, the expected sample sizes for the specified model in the SAMPLESIZE statement are also displayed.

With the STOPPROB option, the "Expected Cumulative Stopping Probabilities" table in Output 110.3.4 displays the expected cumulative stopping stage and cumulative stopping probability to reject the null hypothesis H_0 at each stage under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and c_i are values specified in the CREF= option. By default, $c_i = 0, 0.5, 1.0, 1.5$.

Output 110.3.4 Stopping Probabilities

Expected Cumulative Stopping Probabilities Reference = CRef * (Alt Reference)						
			Stopping Probabilities			
CRef	Expected Stopping Stage	Source	Stage_1	Stage_2	Stage_3	Stage_4
0.0000	3.908	Reject Null	0.01821	0.03155	0.04176	0.05000
0.5000	3.518	Reject Null	0.07005	0.15939	0.25242	0.34327
1.0000	2.358	Reject Null	0.27482	0.58074	0.78638	0.90002
1.5000	1.476	Reject Null	0.61145	0.92348	0.98900	0.99869

Note that at $c_i = 0$, the cumulative stopping probability to reject H_0 at the final stage is the overall Type I error probability 0.05. At $c_i = 1$, the alternative hypothesis $H_1 : \theta = \theta_1$, the cumulative stopping probability to reject H_0 includes both the probability in the lower rejection region and the probability in the upper rejection region. This stopping probability to reject H_0 at the final stage, 0.90002, is slightly greater than the power $1 - \beta = 0.90$, which corresponds to the cumulative stopping probability in the upper rejection region only. See the section "Type I and Type II Errors" on page 9166 for a detailed description of the Type II error probability β.

The "Boundary Information" table in Output 110.3.5 displays the information level, alternative references, and boundary values at each stage. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the standardized Z scale is used to display the alternative references and boundary values. The resulting standardized alternative reference at stage k is given by $\theta_1\sqrt{I_k}$, where θ_1 is the alternative reference and I_k is the information level at stage k, $k = 1, 2, 3, 4$.

Output 110.3.5 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0							
	Information Level			Alternative Reference		Boundary Values	
						Lower	Upper
Stage	Proportion	Actual	N	Lower	Upper	Alpha	Alpha
1	0.2500	19.42461	55.94288	-1.76293	1.76293	-2.36129	2.36129
2	0.5000	38.84922	111.8858	-2.49317	2.49317	-2.36129	2.36129
3	0.7500	58.27383	167.8286	-3.05349	3.05349	-2.36129	2.36129
4	1.0000	77.69844	223.7715	-3.52587	3.52587	-2.36129	2.36129

By default (or equivalently if you specify INFO=EQUAL in the DESIGN statement), equally spaced information levels are used. With the SAMPLESIZE statement, the required sample size N is also displayed under the heading "Information Level." With the Pocock method, the standardized Z boundary values are identical at all stages for each α boundary.

At each interim stage, the hypothesis of $H_0 : \theta = 0$ is rejected if the standardized normal test statistic $z \leq -2.36129$, the lower α boundary, or $z \geq 2.36129$, the upper α boundary. Otherwise, the trial continues to the next stage. At the final stage, stage 4, the trial stops and the hypothesis is rejected if the test statistic $|z_4| \geq 2.36129$. Otherwise, the hypothesis is accepted.

The "Error Spending Information" in Output 110.3.6 displays cumulative error spending at each stage for each boundary. It shows that more α errors are used in early stages than in later stages.

Output 110.3.6 Error Spending Information

Error Spending Information					
		Cumulative Error Spending			
	Information Level	Lower		Upper	
Stage	Proportion	Alpha	Beta	Beta	Alpha
1	0.2500	0.00911	0.00002	0.00002	0.00911
2	0.5000	0.01577	0.00002	0.00002	0.01577
3	0.7500	0.02088	0.00002	0.00002	0.02088
4	1.0000	0.02500	0.10000	0.10000	0.02500

The "Sample Size Summary" table in Output 110.3.7 displays the specified parameters for the sample size computation of the two-sample test for mean difference.

Output 110.3.7 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Means
Mean Difference	0.4
Standard Deviation	0.8
Max Sample Size	223.7715
Expected Sample Size (Null Ref)	218.652
Expected Sample Size (Alt Ref)	131.9167
Weight (Group A)	2
Weight (Group B)	1

The "Sample Sizes (N)" table in Output 110.3.8 displays the derived sample sizes at each stage, in both fractional and integer numbers. With the WEIGHT=2 option, the allocation ratio is 2 for the first group and 1 for the second group. See the section "Test for the Difference between Two Normal Means" on page 9196 for the derivation of these sample sizes. With the fixed-sample information ratio 1.1831, the derived sample sizes in fractional numbers are derived by multiplying 1.1831 by the corresponding sample sizes in the fixed-sample design.

Output 110.3.8 Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Mean Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	55.94	37.30	18.65	19.4246	57	38	19	19.7917
2	111.89	74.59	37.30	38.8492	113	75	38	39.4082
3	167.83	111.89	55.94	58.2738	168	112	56	58.3333
4	223.77	149.18	74.59	77.6984	225	150	75	78.1250

These fractional sample sizes are rounded up to integers under the heading "Ceiling N." When the resulting integer sample sizes are used, the corresponding information levels are slightly larger than the levels from the design. This can increase the power slightly if a trial uses these integer sample sizes.

Note that compared with other designs, a Pocock design can stop the trial early with a larger p-value. However, this might not be persuasive enough to make a new treatment widely accepted (Pocock and White 1999).

The "Design Information" table in Output 110.3.9 displays design specifications and the derived statistics for the O'Brien-Fleming design. With the specified ALTREF= option, the maximum information $I_X = 67.1268$ is derived.

Output 110.3.9 O'Brien-Fleming Design Information

The SEQDESIGN Procedure
Design: TwoSidedOBrienFleming

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	O'Brien-Fleming
Boundary Key	Both
Alternative Reference	0.4
Number of Stages	4
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	102.2163
Max Information	67.12682
Null Ref ASN (Percent of Fixed Sample)	101.5728
Alt Ref ASN (Percent of Fixed Sample)	76.7397

With the corresponding fixed-sample information

$$I_0 = \frac{(\Phi^{-1}(1-\alpha/2) + \Phi^{-1}(1-\beta))^2}{0.4^2} = \frac{(1.96 + 1.28155)^2}{0.16} = 65.6728$$

the fixed-sample information ratio is $67.1268/65.6728 = 1.022$. That is, the maximum information for the O'Brien-Fleming design is only 2.2% more than for the corresponding fixed-sample design.

The "Method Information" table in Output 110.3.10 displays the Type I α level and Type II β level. It also displays the derived drift parameter $\theta_1\sqrt{I_X}$, which is the standardized alternative reference at the final stage.

Output 110.3.10 Method Information

Method Information

				Unified Family				
Boundary	**Method**	**Alpha**	**Beta**	**Rho**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	O'Brien-Fleming	0.02500	0.10000	0.5	0	2.02429	0.4	3.277238
Lower Alpha	O'Brien-Fleming	0.02500	0.10000	0.5	0	2.02429	-0.4	-3.27724

With the METHOD=OBF option, the O'Brien-Fleming method is used for each boundary. The O'Brien-Fleming method is one of the unified family methods, and the table also displays its corresponding parameters $\rho = 0.5$ as a unified family method and the derived parameter $C = 2.0243$ for the boundary values.

With the PSS option, the "Power and Expected Sample Sizes" table in Output 110.3.11 displays powers and expected sample sizes under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and c_i are values specified in the CREF= option.

Output 110.3.11 Power and Expected Sample Size Information

Powers and Expected Sample Sizes Reference = CRef * (Alt Reference)		
		Sample Size
CRef	Power	Percent Fixed-Sample
0.0000	0.02500	101.5728
0.5000	0.36495	96.3684
1.0000	0.90000	76.7397
1.5000	0.99821	57.2590

Compared with the corresponding Pocock design, the O'Brien-Fleming design has a smaller maximum sample size, and smaller expected sample sizes under hypothetical references $\theta = 0$ and $\theta = 0.5\,\theta_1$, but larger expected sample sizes under hypothetical references $\theta = \theta_1$ and $\theta = 1.5\,\theta_1$.

With the STOPPROB option, the "Expected Cumulative Stopping Probabilities" table in Output 110.3.12 displays the expected stopping stage and cumulative stopping probability to reject the null hypothesis at each stage under various hypothetical references $\theta = c_i\,\theta_1$, where θ_1 is the alternative reference and c_i are values specified in the CREF= option.

Output 110.3.12 Stopping Probabilities

Expected Cumulative Stopping Probabilities Reference = CRef * (Alt Reference)						
			Stopping Probabilities			
CRef	Expected Stopping Stage	Source	Stage_1	Stage_2	Stage_3	Stage_4
0.0000	3.975	Reject Null	0.00005	0.00422	0.02091	0.05000
0.5000	3.771	Reject Null	0.00062	0.04430	0.18392	0.36515
1.0000	3.003	Reject Null	0.00798	0.29296	0.69603	0.90000
1.5000	2.241	Reject Null	0.05584	0.73031	0.97315	0.99821

Compared with the corresponding Pocock design, the O'Brien-Fleming design has smaller stopping probabilities in early stages under each hypothetical reference.

The "Boundary Information" table in Output 110.3.13 displays the boundary values for the design that uses the O'Brien-Fleming method. Compared with the Pocock method, the standardized statistics α boundary values derived from the O'Brien-Fleming method in absolute values are larger in early stages and smaller in later stages. This makes the O'Brien-Fleming design less likely to reject the null hypothesis in early stages than the Pocock design. With the derived parameter $C = 2.0243$ for the α boundary, the α boundaries at stage j are computed as $C\sqrt{4/j}$, $j = 1, \ldots, 4$.

Output 110.3.13 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0							
	Information Level			Alternative Reference		Boundary Values	
						Lower	Upper
Stage	Proportion	Actual	N	Lower	Upper	Alpha	Alpha
1	0.2500	16.7817	48.33131	-1.63862	1.63862	-4.04859	4.04859
2	0.5000	33.56341	96.66262	-2.31736	2.31736	-2.86278	2.86278
3	0.7500	50.34511	144.9939	-2.83817	2.83817	-2.33745	2.33745
4	1.0000	67.12682	193.3252	-3.27724	3.27724	-2.02429	2.02429

The "Error Spending Information" in Output 110.3.14 displays cumulative error spending at each stage for each boundary. With smaller α spending in early stages for the O'Brien-Fleming method, it also indicates that the O'Brien-Fleming design is less likely to reject the null hypothesis in early stages than the Pocock design.

Output 110.3.14 Error Spending Information

Error Spending Information					
		Cumulative Error Spending			
	Information Level	Lower		Upper	
Stage	Proportion	Alpha	Beta	Beta	Alpha
1	0.2500	0.00003	0.00000	0.00000	0.00003
2	0.5000	0.00211	0.00000	0.00000	0.00211
3	0.7500	0.01046	0.00000	0.00000	0.01046
4	1.0000	0.02500	0.10000	0.10000	0.02500

The "Sample Size Summary" table in Output 110.3.15 displays the specified parameters for the sample size computation of the two-sample test for mean difference.

Output 110.3.15 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Means
Mean Difference	0.4
Standard Deviation	0.8
Max Sample Size	193.3252
Expected Sample Size (Null Ref)	192.1081
Expected Sample Size (Alt Ref)	145.1404
Weight (Group A)	2
Weight (Group B)	1

The "Sample Sizes (N)" table in Output 110.3.16 displays the derived sample sizes at each stage, in both fractional and integer numbers. With the fixed-sample information ratio 1.0222, the required sample sizes in fractional numbers are derived by multiplying 1.0222 by the corresponding sample sizes in the fixed-sample design.

Output 110.3.16 Derived Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Mean Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	48.33	32.22	16.11	16.7817	50	33	17	17.5313
2	96.66	64.44	32.22	33.5634	98	65	33	34.1996
3	144.99	96.66	48.33	50.3451	146	97	49	50.8669
4	193.33	128.88	64.44	67.1268	194	129	65	67.5338

Example 110.4: Generating Graphics Display for Sequential Designs

(View the complete code for this example (seqdex4.sas) in the example repository.)

This example creates the same group sequential design as in Example 110.3 and creates graphics by using ODS Graphics. The following statements request all available graphs in the SEQDESIGN procedure:

```
ods graphics on;
proc seqdesign altref=0.4
               plots=all
               ;
   TwoSidedPocock:        design nstages=4 method=poc;
   TwoSidedOBrienFleming: design nstages=4 method=obf;
run;
```

With the PLOTS=ALL option, a detailed boundary plot with the rejection region and acceptance region is displayed for the Pocock design, as shown in Output 110.4.1. By default (or equivalently if you specify STOP=REJECT), the rejection boundaries are also generated at interim stages.

Output 110.4.1 Pocock Boundary Plot

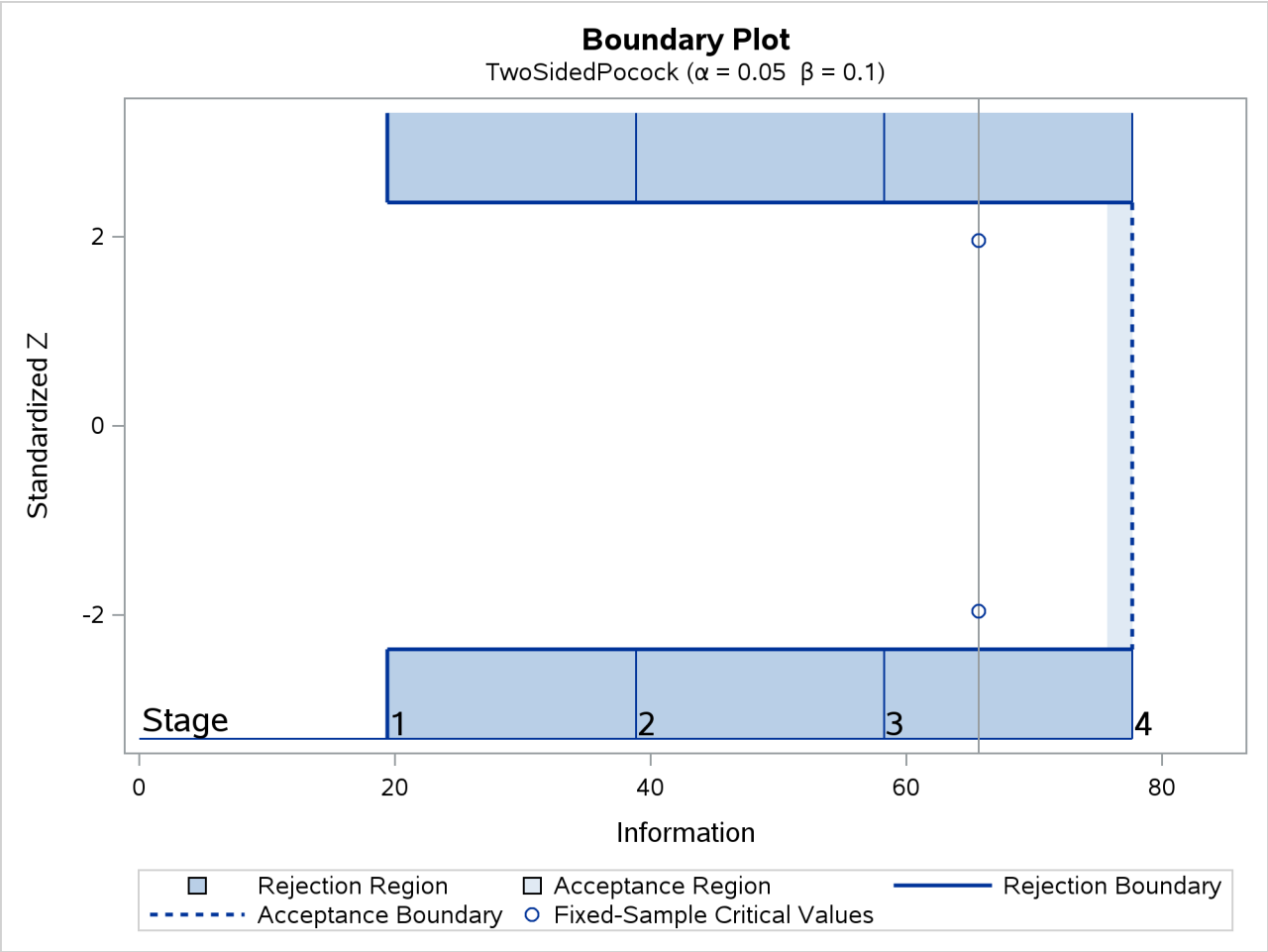

The plot shows identical boundary values in each boundary in the standardized Z scale for the Pocock design. The information level and the critical value for the corresponding fixed-sample design are also displayed.

With the PLOTS=ALL option, a detailed boundary plot with the rejection region and acceptance region is also displayed for the O'Brien-Fleming design, as shown in Output 110.4.2. The plot shows that the rejection boundary values are decreasing as the trial advances in the standardized Z scale.

Output 110.4.2 O'Brien-Fleming Boundary Plot

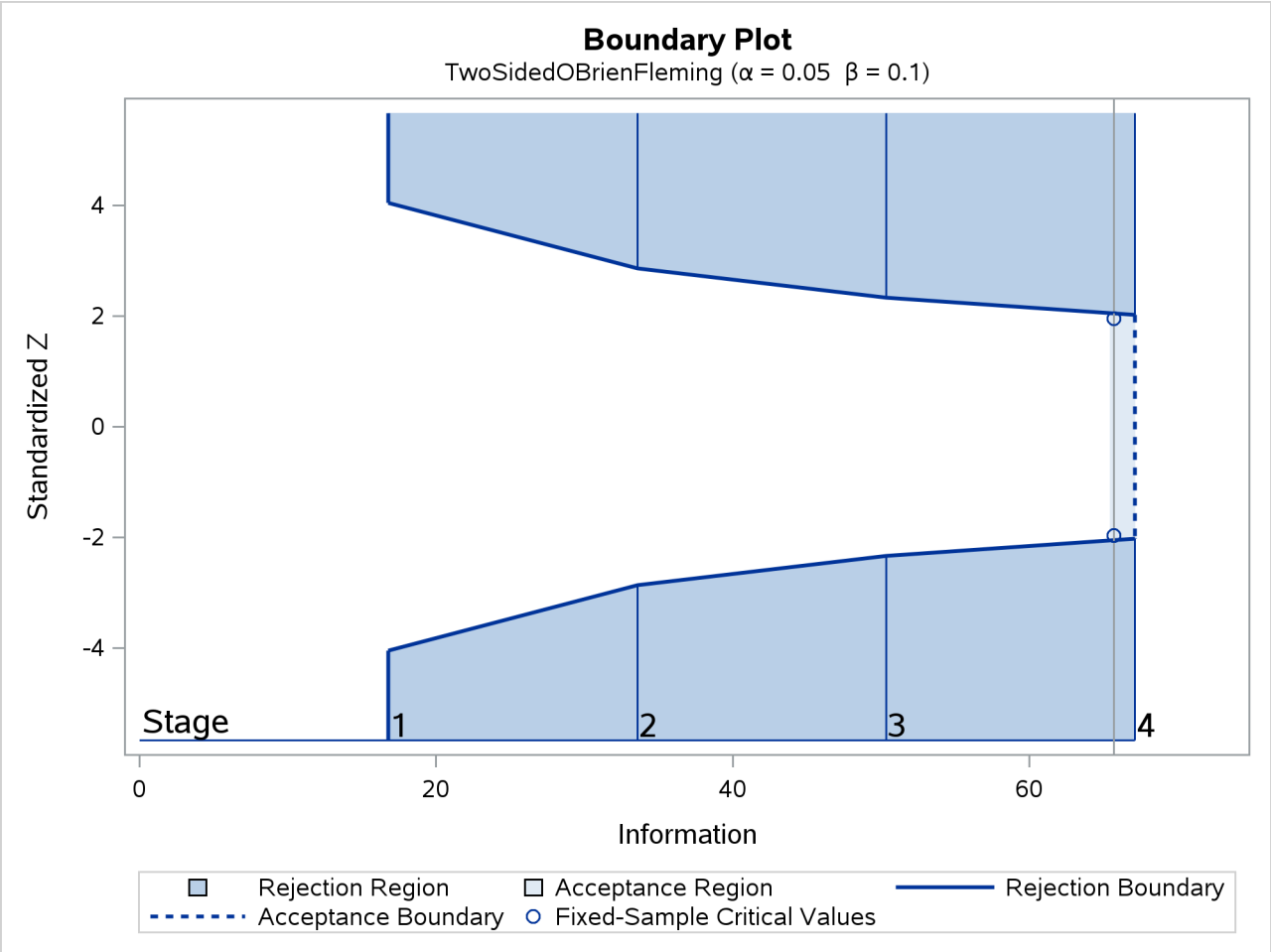

With the PLOTS=ALL option, the procedure displays a plot of average sample numbers (expected sample sizes for nonsurvival data or expected numbers of events for survival data) under various hypothetical references for all designs simultaneously, as shown in Output 110.4.3. By default, the option CREF= $0, 0.01, 0.02, \ldots, 1.50$ and expected sample sizes under the hypothetical references $\theta = c_i\,\theta_1$ are displayed, where c_i are values specified in the CREF= option. These CREF= values are displayed on the horizontal axis.

Output 110.4.3 ASN Plot

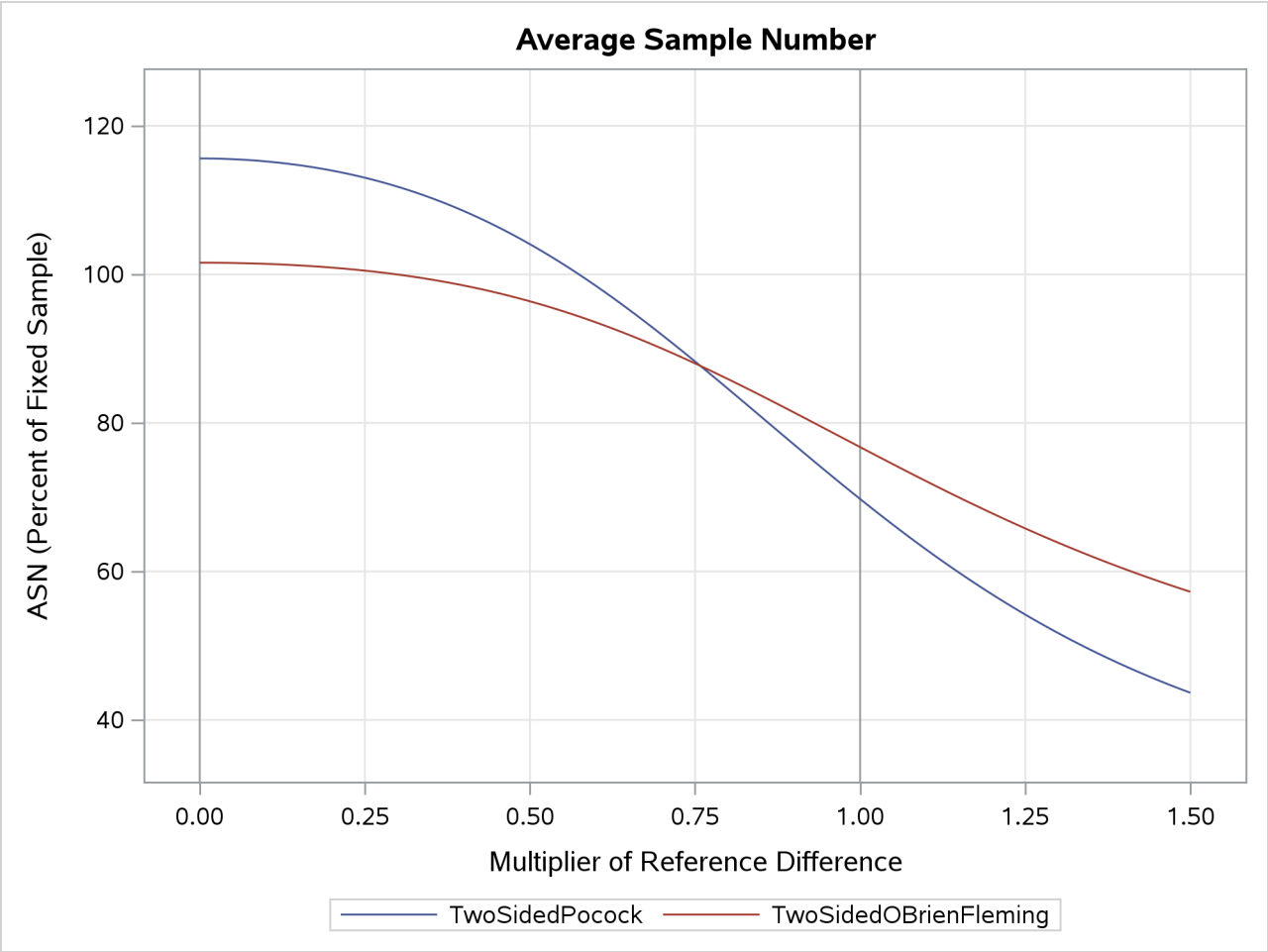

The plot shows that the Pocock design has a larger expected sample size than the O'Brien-Fleming design under the null hypothesis ($c_i = 0$) and has a smaller expected sample size under the alternative hypothesis ($c_i = 1$).

With the PLOTS=ALL option, the procedure displays a plot of the power curves under various hypothetical references for all designs simultaneously, as shown in Output 110.4.4. By default, the option CREF= $0, 0.01, 0.02, \ldots, 1.50$ and powers under hypothetical references $\theta = c_i\,\theta_1$ are displayed, where c_i are values specified in the CREF= option. These CREF= values are displayed on the horizontal axis.

Output 110.4.4 Power Plot

Under the null hypothesis, $c_i = 0$, the power is 0.025, the upper Type I error probability. Under the alternative hypothesis, $c_i = 1$, the power is 0.9, one minus the Type II error probability. The plot shows only minor difference between the two designs.

With the PLOTS=ALL option, the procedure displays a plot of sequential boundaries for all designs simultaneously, as shown in Output 110.4.5. By default (or equivalently if you specify HSCALE=INFO in the COMBINEDBOUNDARY option), the information levels are used on the horizontal axis.

Output 110.4.5 Combined Boundary Plot

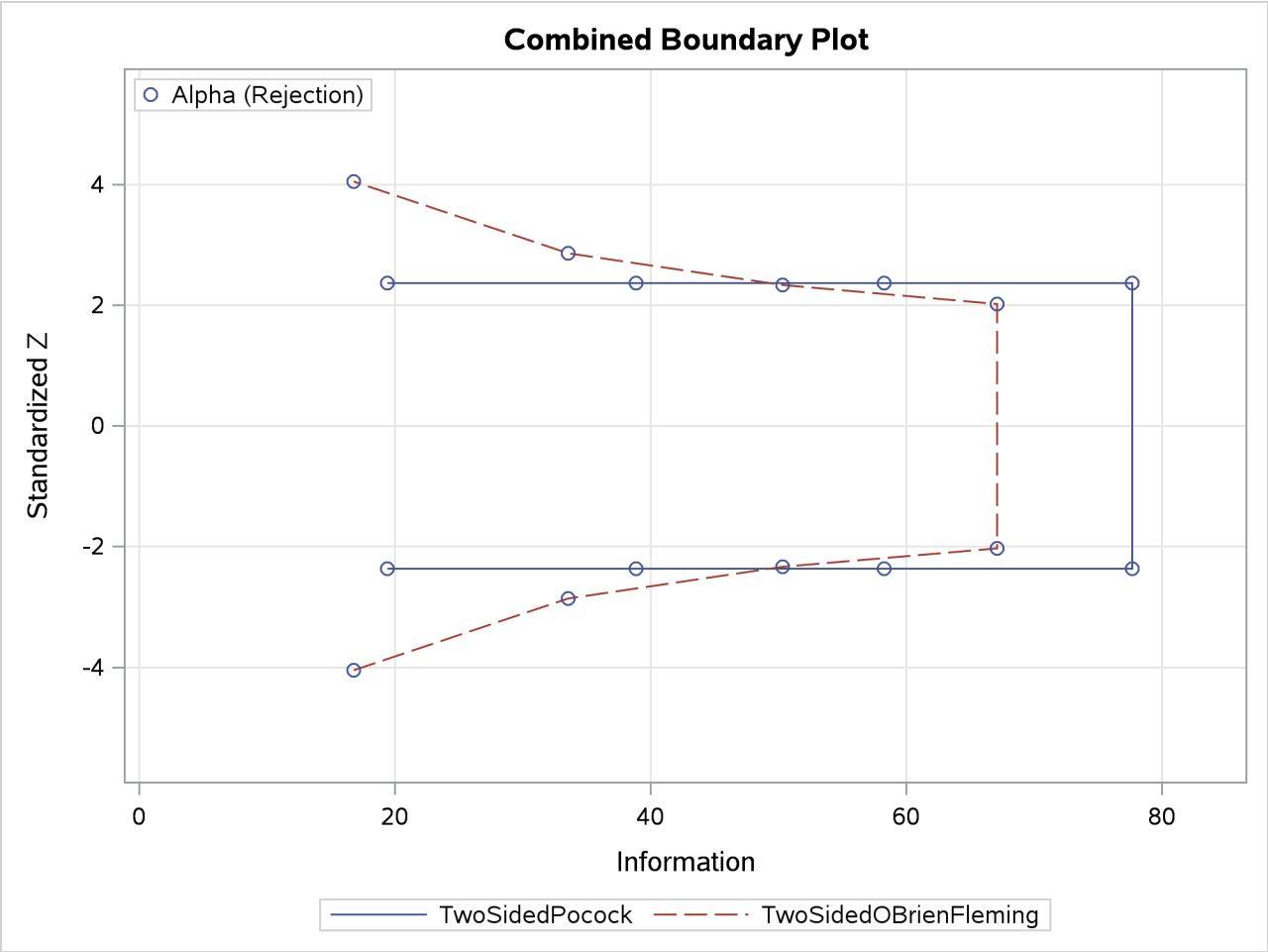

The plot shows that the α boundary values (in absolute value) created from the O'Brien-Fleming method are greater in early stages and smaller in later stages than the boundary values from the Pocock method. The plot also shows that the information level in the Pocock design is larger than the corresponding level in the O'Brien-Fleming design at each stage.

With the PLOTS=ALL option, the procedure displays a plot of cumulative error spends for all boundaries in the designs simultaneously, as shown in Output 110.4.6. With a symmetric two-sided design, cumulative error spending is displayed only for the upper α boundary. The plot shows that for the upper α boundary, the O'Brien-Fleming method spends fewer errors in early stages and more errors in later stages than the corresponding Pocock method.

Output 110.4.6 Error Spending Plot

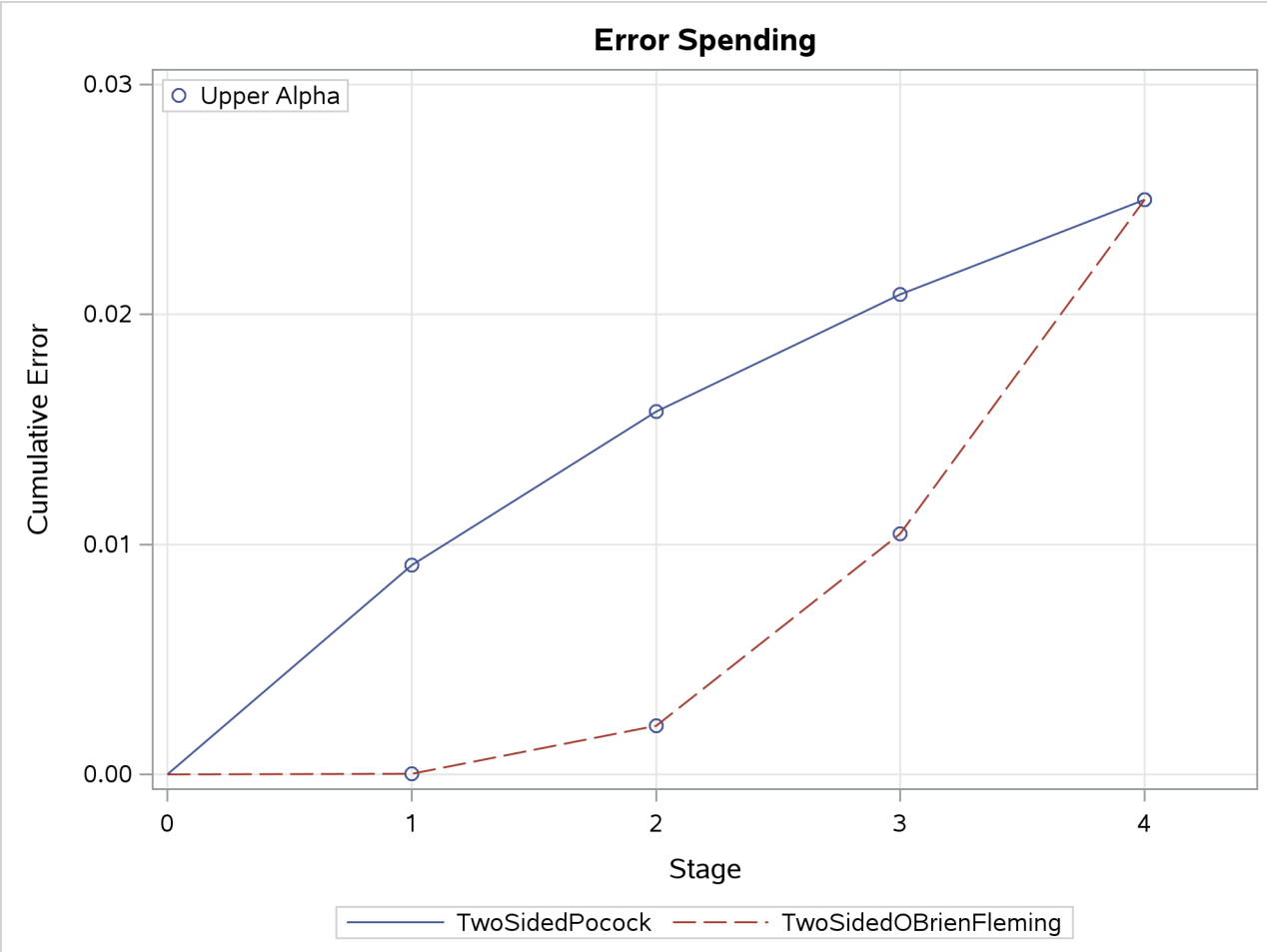

Example 110.5: Creating Designs Using Haybittle-Peto Methods

(View the complete code for this example (seqdex5.sas) in the example repository.)

This example requests two 3-stage group sequential designs for normally distributed statistics. Each design uses a Haybittle-Peto method with a two-sided alternative and early stopping to reject the hypothesis. One design uses the specified interim boundary Z values and derives the final-stage boundary value for the specified α and β errors. The other design uses the specified boundary Z values and derives the overall α and β errors.

The following statements specify the interim boundary Z values and derive the final-stage boundary value for the specified $\alpha = 0.05$ and $\beta = 0.10$:

```
ods graphics on;
proc seqdesign altref=0.25
               errspend
               stopprob
               plots=errspend
               ;
```

```
    OneSidedPeto: design nstages=3
                  method=peto( z=3)
                  alpha=0.05  beta=0.10;
run;
```

The "Design Information" table in Output 110.5.1 displays design specifications and maximum information in percentage of its corresponding fixed-sample design.

Output 110.5.1 Haybittle-Peto Design Information

The SEQDESIGN Procedure
Design: OneSidedPeto

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Reject Null
Method	Haybittle-Peto
Boundary Key	Both
Alternative Reference	0.25
Number of Stages	3
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	100.2466
Max Information	137.3592
Null Ref ASN (Percent of Fixed Sample)	100.1192
Alt Ref ASN (Percent of Fixed Sample)	87.35

The "Method Information" table in Output 110.5.2 displays the α and β errors and the derived drift parameter, which is the standardized alternative reference at the final stage.

Output 110.5.2 Method Information

Method Information					
Boundary	**Method**	**Alpha**	**Beta**	**Alternative Reference**	**Drift**
Upper Alpha	Haybittle-Peto	0.05000	0.10000	0.25	2.930009

With the STOPPROB option, the "Expected Cumulative Stopping Probabilities" table in Output 110.5.3 displays the expected stopping stage and cumulative stopping probability to reject the null hypothesis at each stage under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and $c_i = 0, 0.5, 1, 1.5$ are the default values in the CREF= option.

Output 110.5.3 Stopping Probabilities

Expected Cumulative Stopping Probabilities Reference = CRef * (Alt Reference)					
			Stopping Probabilities		
CRef	Expected Stopping Stage	Source	Stage_1	Stage_2	Stage_3
0.0000	2.996	Reject Null	0.00135	0.00246	0.05000
0.5000	2.941	Reject Null	0.01561	0.04372	0.42762
1.0000	2.614	Reject Null	0.09538	0.29057	0.90000
1.5000	1.944	Reject Null	0.32185	0.73442	0.99698

The "Boundary Information" table in Output 110.5.4 displays information level, alternative references, and boundary values. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the standardized Z scale is used to display the alternative references and boundary values. The resulting standardized alternative reference at stage k is given by $\theta_1\sqrt{I_k}$, where θ_1 is the alternative reference and I_k is the information level at stage k, $k = 1, 2, 3$.

Output 110.5.4 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0				
			Alternative	Boundary Values
	Information Level		Reference	Upper
Stage	Proportion	Actual	Upper	Alpha
1	0.3333	45.7864	1.69164	3.00000
2	0.6667	91.57281	2.39234	3.00000
3	1.0000	137.3592	2.93001	1.65042

At each interim stage, if the standardized statistic $z \geq 3$, the trial is stopped and the null hypothesis is rejected. If the statistic $z < 3$, the trial continues to the next stage. At the final stage, the null hypothesis is rejected if the statistic $z_3 > 1.65$. Otherwise, the hypothesis is accepted. Note that the boundary values at the final stage, 1.65, are close to the critical values 1.645 in the corresponding fixed-sample design.

The "Error Spending Information" in Output 110.5.5 displays cumulative error spending at each stage for each boundary. The stage 1 α spending 0.00135 corresponds to the one-sided p-value for a standardized Z statistic, $Z > 3$.

Output 110.5.5 Error Spending Information

Error Spending Information			
		Cumulative Error Spending	
	Information Level	Upper	
Stage	Proportion	Beta	Alpha
1	0.3333	0.00000	0.00135
2	0.6667	0.00000	0.00246
3	1.0000	0.10000	0.05000

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.5.6. With the STOP=REJECT option, the interim rejection boundaries are displayed.

Output 110.5.6 Boundary Plot

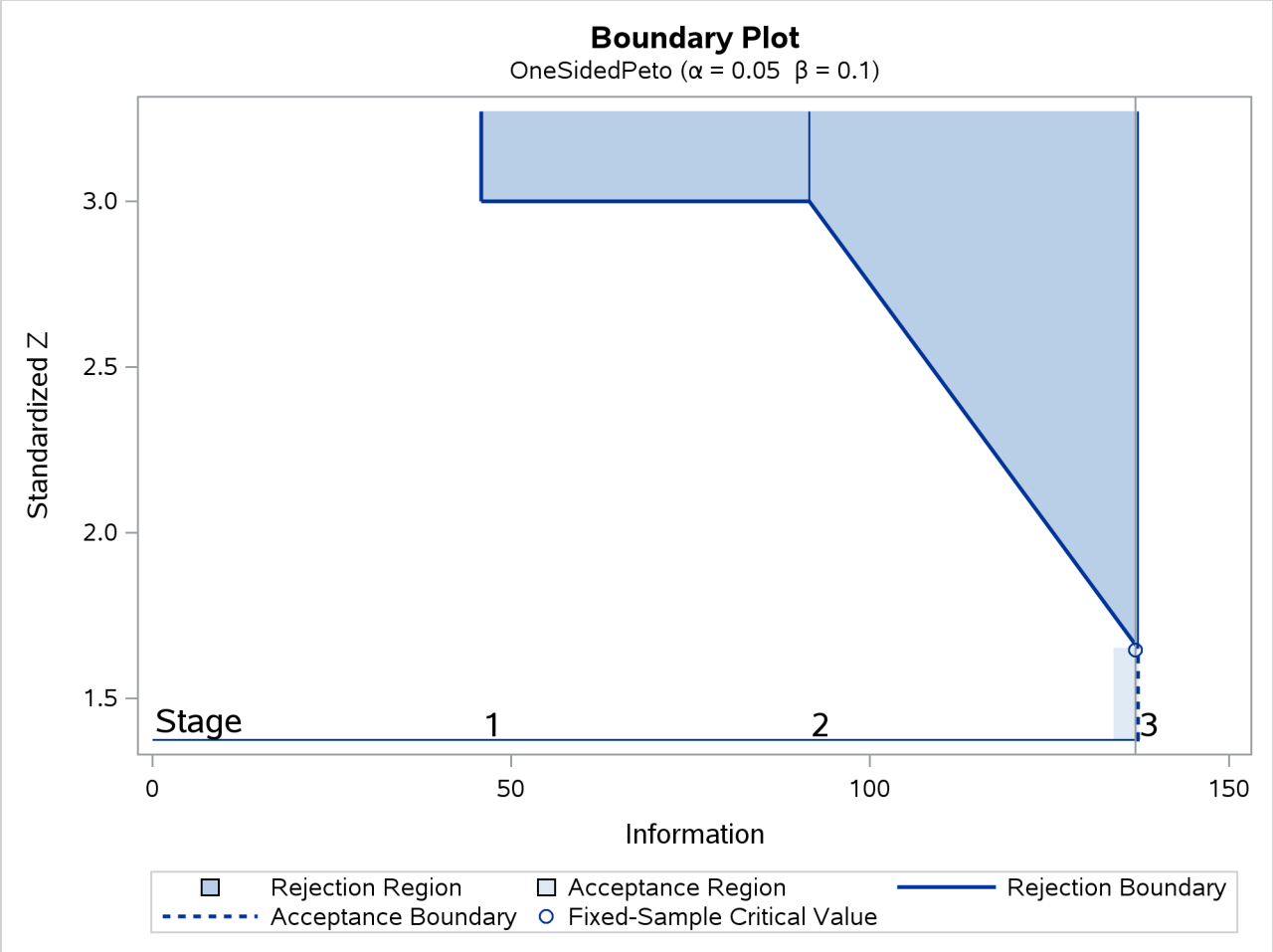

With the PLOTS=ERRSPEND option, the procedure displays a plot of error spending for each boundary, as shown in Output 110.5.7. The error spending values in the "Error Spending Information" in Output 110.5.4 are displayed in the plot. As expected, the error spending at each of the first two stages is small, with the standardized Z boundary value 3.

Output 110.5.7 Error Spending Plot

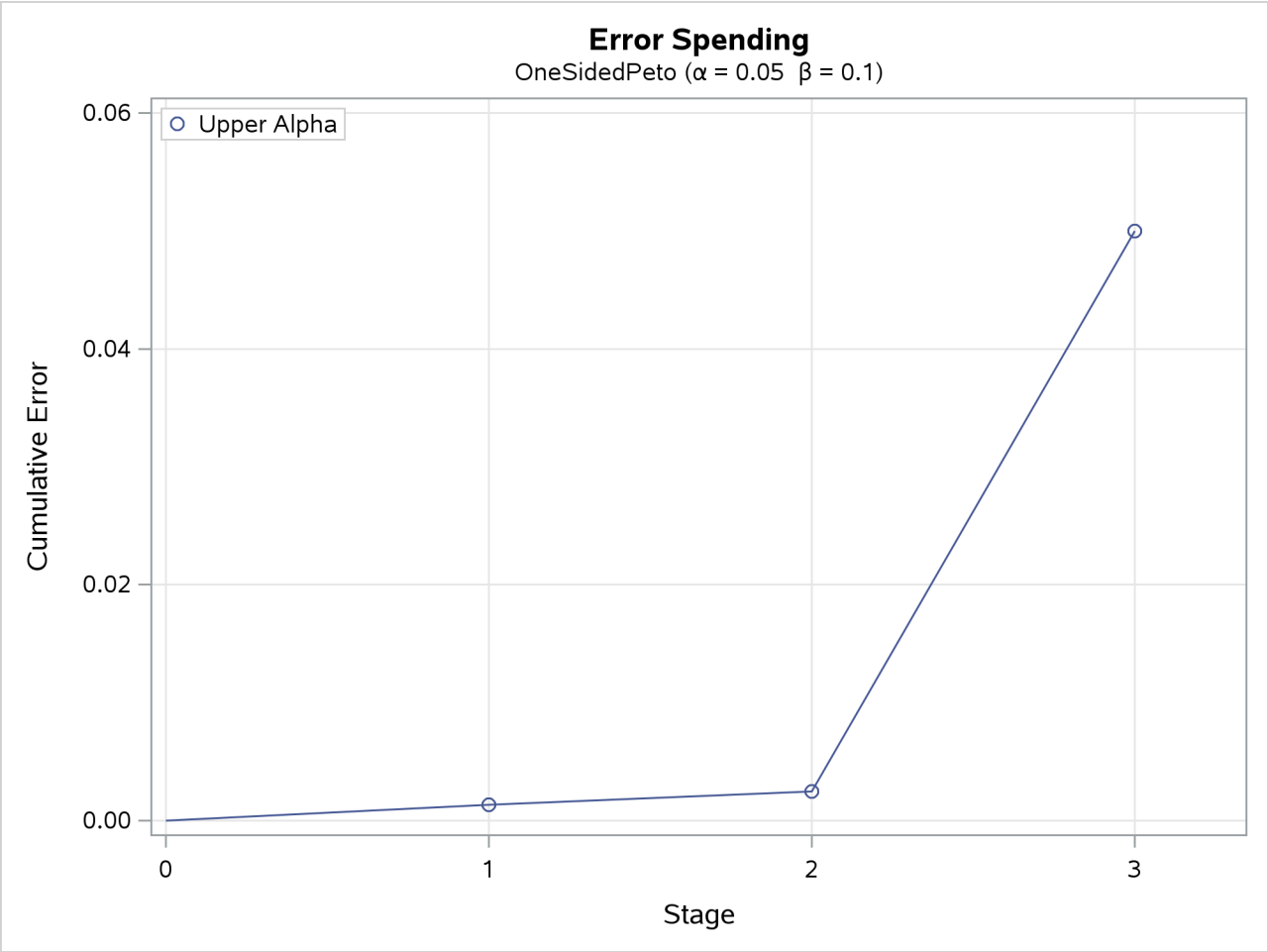

The following statements specify the boundary Z values and derive the α and β errors from these completely specified boundary values:

```
ods graphics on;
proc seqdesign altref=0.25
               maxinfo=200
               errspend
               stopprob
               plots=errspend
               ;
   OneSidedPeto: design nstages=3
                 method=peto(z=3 2.5 2)
                 alt=upper  stop=reject
                 boundarykey=none
                 ;
run;
```

The “Design Information” table in Output 110.5.8 displays design specifications and derived α and β error levels.

Output 110.5.8 Design Information

The SEQDESIGN Procedure
Design: OneSidedPeto

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Reject Null
Method	Haybittle-Peto
Boundary Key	None
Alternative Reference	0.25
Number of Stages	3
Alpha	0.02532
Beta	0.06035
Power	0.93965
Max Information (Percent of Fixed Sample)	101.6769
Max Information	200
Null Ref ASN (Percent of Fixed Sample)	101.3933
Alt Ref ASN (Percent of Fixed Sample)	73.74031

The "Method Information" table in Output 110.5.9 displays the α and β errors and the derived drift parameter for each boundary.

Output 110.5.9 Method Information

Method Information

Boundary	Method	Alpha	Beta	Alternative Reference	Drift
Upper Alpha	Haybittle-Peto	0.02532	0.06035	0.25	3.535534

With the STOPPROB option, the "Expected Cumulative Stopping Probabilities" table in Output 110.5.10 displays the expected stopping stage and cumulative stopping probability to reject the null hypothesis at each stage under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and $c_i = 0, 0.5, 1, 1.5$ are the default values in the CREF= option.

Output 110.5.10 Stopping Probabilities

Expected Cumulative Stopping Probabilities
Reference = CRef * (Alt Reference)

			Stopping Probabilities		
CRef	Expected Stopping Stage	Source	Stage_1	Stage_2	Stage_3
0.0000	2.992	Reject Null	0.00135	0.00702	0.02532
0.5000	2.826	Reject Null	0.02389	0.15030	0.41775
1.0000	2.176	Reject Null	0.16884	0.65544	0.93965
1.5000	1.508	Reject Null	0.52466	0.96708	0.99954

The "Boundary Information" table in Output 110.5.11 displays information level, alternative references, and boundary values.

Output 110.5.11 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0				
			Alternative	Boundary Values
	Information Level		Reference	Upper
Stage	Proportion	Actual	Upper	Alpha
1	0.3333	66.66667	2.04124	3.00000
2	0.6667	133.3333	2.88675	2.50000
3	1.0000	200	3.53553	2.00000

The "Error Spending Information" in Output 110.5.12 displays cumulative error spending at each stage for each boundary. The first-stage α spending 0.00135 corresponds to the one-sided p-value for a standardized Z statistic, $Z > 3$.

Output 110.5.12 Error Spending Information

Error Spending Information			
		Cumulative Error Spending	
	Information Level	Upper	
Stage	Proportion	Beta	Alpha
1	0.3333	0.00000	0.00135
2	0.6667	0.00000	0.00702
3	1.0000	0.06035	0.02532

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.5.13. With the STOP=REJECT option, the interim rejection boundaries are displayed.

Output 110.5.13 Boundary Plot

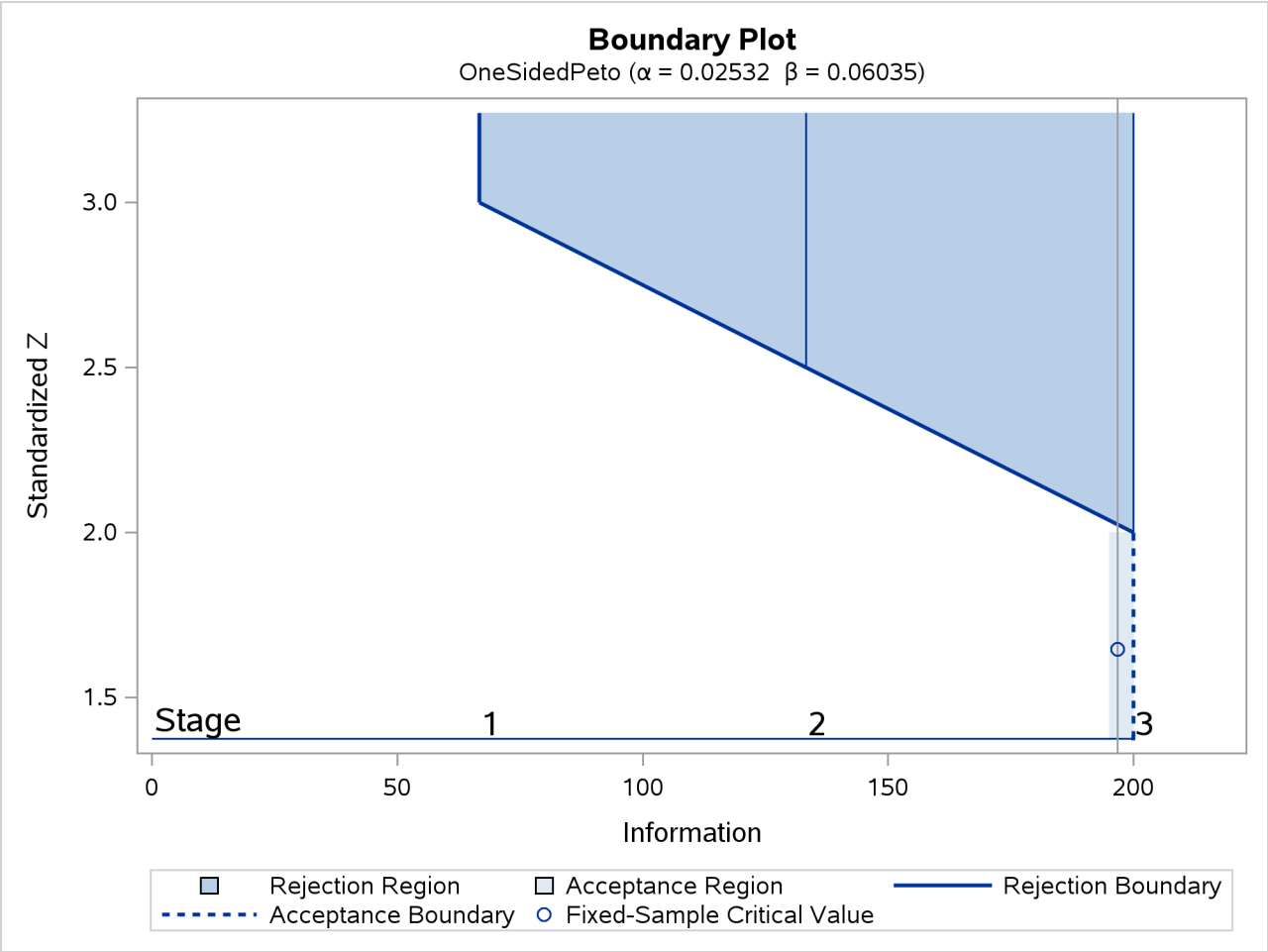

With the PLOTS=ERRSPEND option, the procedure displays a plot of error spending for each boundary, as shown in Output 110.5.14. The error spending values in the "Error Spending Information" table in Output 110.5.10 are displayed in the plot.

Output 110.5.14 Error Spending Plot

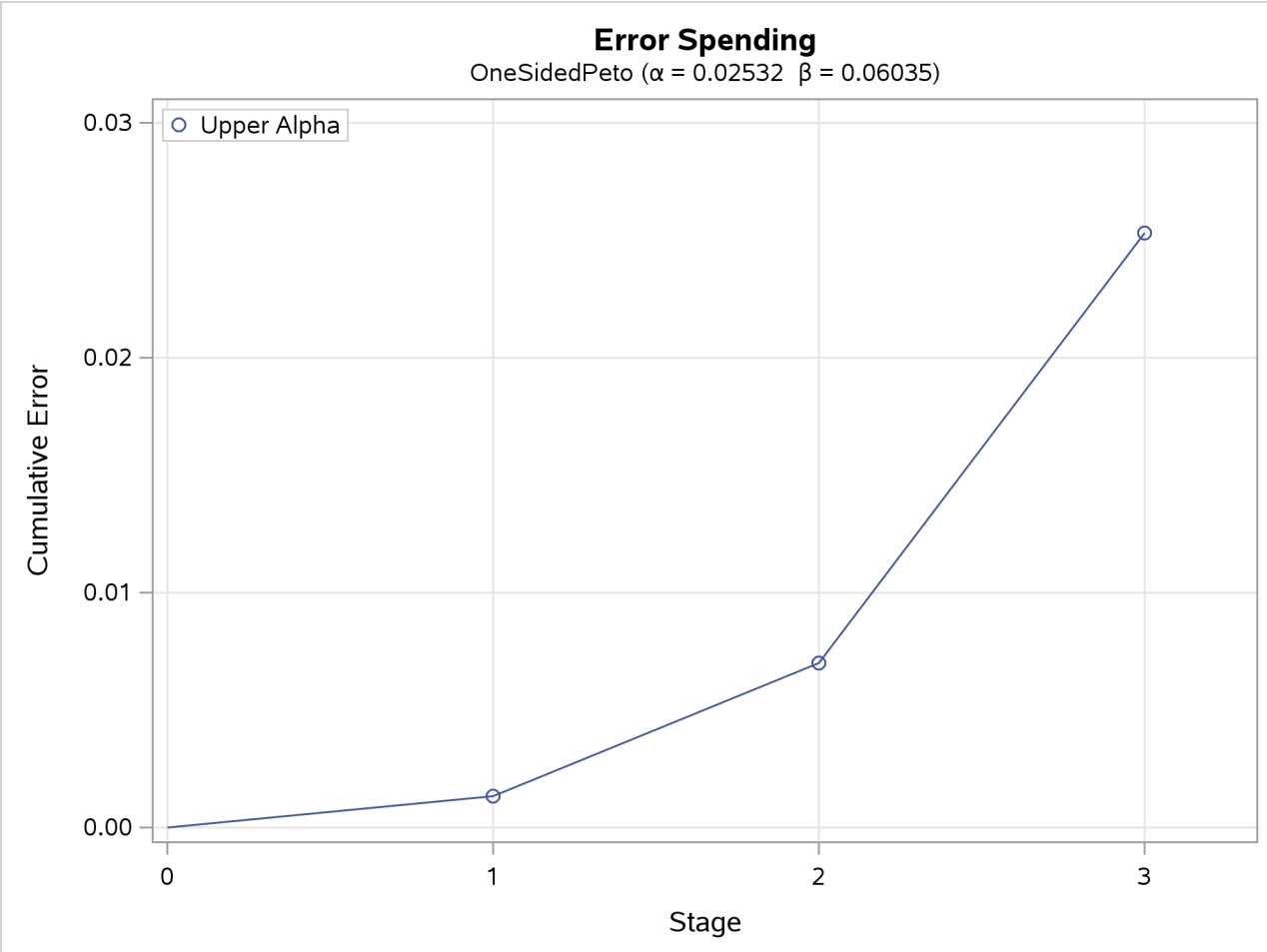

Example 110.6: Creating Designs with Various Stopping Criteria

(View the complete code for this example (seqdex6.sas) in the example repository.)

This example requests three 5-stage group sequential designs for normally distributed statistics. Each design uses a triangular method with the specified one-sided upper alternative reference $\theta_1 = 0.2$. The resulting boundary values are displayed with the score scale. Note that these unified family triangular designs are different from Whitehead's triangular designs.

The following statements request three designs with different stopping criterion:

```
ods graphics on;
proc seqdesign altref=0.2
               bscale=score
               errspend
               plots=(combinedboundary errspend(hscale=info))
               ;
   StopToRejectAccept: design nstages=5 method=tri alt=upper stop=both;
   StopToReject:       design nstages=5 method=tri alt=upper stop=reject;
```

```
   StopToAccept:       design nstages=5 method=tri alt=upper stop=accept;
run;
```

The first design has early stopping to reject or accept the null hypothesis H_0.

The "Design Information" table in Output 110.6.1 displays design specifications and derived statistics. With the specified alternative reference, the maximum information is derived.

Output 110.6.1 Triangular Design Information

The SEQDESIGN Procedure
Design: StopToRejectAccept

Design Information	
Statistic Distribution	Normal
Boundary Scale	Score
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Method	Triangular
Boundary Key	Both
Alternative Reference	0.2
Number of Stages	5
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	140.0293
Max Information	299.797
Null Ref ASN (Percent of Fixed Sample)	59.11973
Alt Ref ASN (Percent of Fixed Sample)	66.94909

The "Method Information" table in Output 110.6.2 displays the α and β errors and the derived drift parameter, which is the standardized alternative reference at the final stage. The table also shows the corresponding parameters for a triangular method as a unified family method.

Output 110.6.2 Method Information

Method Information								
				Unified Family				
Boundary	**Method**	**Alpha**	**Beta**	**Rho**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	Triangular	0.05000	.	0.5	1	0.94394	0.2	3.46293
Upper Beta	Triangular	.	0.10000	0.5	1	0.78753	0.2	3.46293

The "Boundary Information" table in Output 110.6.3 displays information level, alternative reference, and boundary values. With the specified BOUNDARYSCALE=SCORE option, the alternative reference and boundary values are displayed in the score statistic scale. With a score scale, the alternative reference is $\theta_1 I_k$, where θ_1 is the specified alternative reference and I_k is the information level at stage k, $k = 1, 2, \ldots, 5$.

Output 110.6.3 Boundary Information

Boundary Information (Score Scale) Null Reference = 0					
	Information Level		Alternative Reference	Boundary Values	
				Upper	
Stage	Proportion	Actual	Upper	Beta	Alpha
1	0.2000	59.9594	11.99188	-4.37102	19.61274
2	0.4000	119.9188	23.98376	4.89371	22.88154
3	0.6000	179.8782	35.97564	14.15845	26.15033
4	0.8000	239.8376	47.96752	23.42318	29.41912
5	1.0000	299.797	59.95940	32.68791	32.68791

The "Error Spending Information" table in Output 110.6.4 displays cumulative error spending at each stage for each boundary.

Output 110.6.4 Error Spending Information

Error Spending Information			
		Cumulative Error Spending	
	Information Level		
		Upper	
Stage	Proportion	Beta	Alpha
1	0.2000	0.01729	0.00566
2	0.4000	0.04927	0.02138
3	0.6000	0.07611	0.03643
4	0.8000	0.09357	0.04641
5	1.0000	0.10000	0.05000

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.6.5. With the STOP=BOTH option, both the acceptance and rejection boundaries at interim stages are displayed. With the score scale, the acceptance and rejection boundaries are straight lines and form a triangular-shape continuation region.

Output 110.6.5 Boundary Plot with Score Statistics

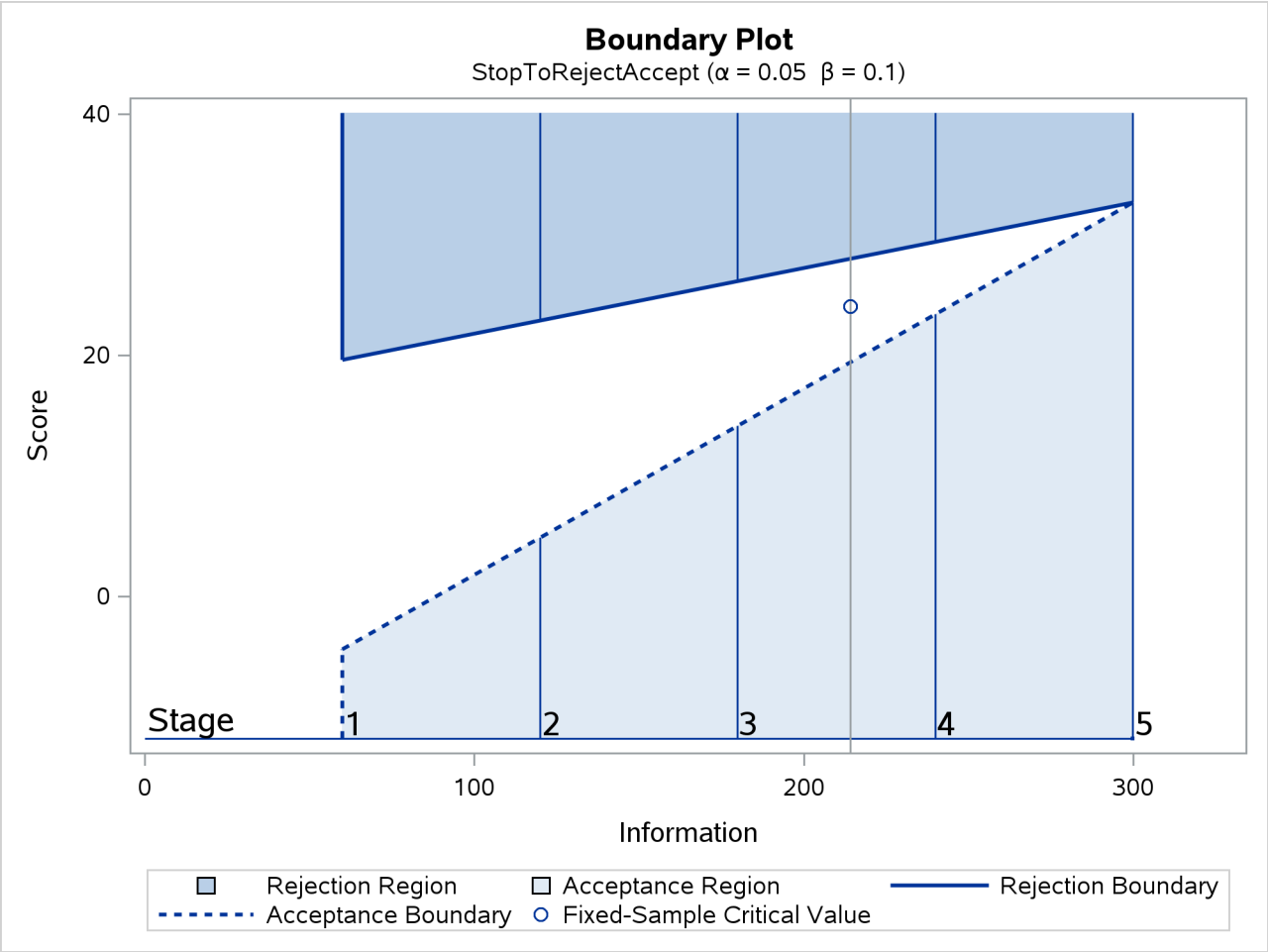

The second design has early stopping only to reject the null hypothesis H_0.

The "Design Information" table in Output 110.6.6 displays design specifications and derived statistics. With the specified alternative reference, the maximum information is derived.

Output 110.6.6 Triangular Design Information

The SEQDESIGN Procedure
Design: StopToReject

Design Information	
Statistic Distribution	Normal
Boundary Scale	Score
Alternative Hypothesis	Upper
Early Stop	Reject Null
Method	Triangular
Boundary Key	Both
Alternative Reference	0.2
Number of Stages	5
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	113.4443
Max Information	242.8799
Null Ref ASN (Percent of Fixed Sample)	111.3399
Alt Ref ASN (Percent of Fixed Sample)	67.41968

The "Method Information" table in Output 110.6.7 displays the α and β errors and the derived drift parameter. The table also shows the corresponding parameters for a triangular method as a unified family method.

Output 110.6.7 Method Information

Method Information

				Unified Family				
Boundary	**Method**	**Alpha**	**Beta**	**Rho**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	Triangular	0.05000	0.10000	0.5	1	0.9833	0.2	3.116921

The "Boundary Information" table in Output 110.6.8 displays information level, alternative reference, and boundary values. With the specified BOUNDARYSCALE=SCORE option, the alternative reference and boundary values are displayed in the score statistic scale.

Output 110.6.8 Boundary Information

Boundary Information (Score Scale)
Null Reference = 0

	Information Level		Alternative Reference	Boundary Values
Stage	**Proportion**	**Actual**	**Upper**	**Upper Alpha**
1	0.2000	48.57597	9.71519	18.38919
2	0.4000	97.15194	19.43039	21.45405
3	0.6000	145.7279	29.14558	24.51891
4	0.8000	194.3039	38.86078	27.58378
5	1.0000	242.8799	48.57597	30.64864

The "Error Spending Information" table in Output 110.6.9 displays cumulative error spending at each stage for each boundary.

Output 110.6.9 Error Spending Information

Error Spending Information			
		Cumulative Error Spending	
	Information Level	Upper	
Stage	Proportion	Beta	Alpha
1	0.2000	0.00000	0.00416
2	0.4000	0.00000	0.01705
3	0.6000	0.00000	0.03027
4	0.8000	0.00000	0.04127
5	1.0000	0.10000	0.05000

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.6.10. For a triangular design, these rejection boundaries form a straight line with the score scale.

Output 110.6.10 Boundary Plot with Score Statistics

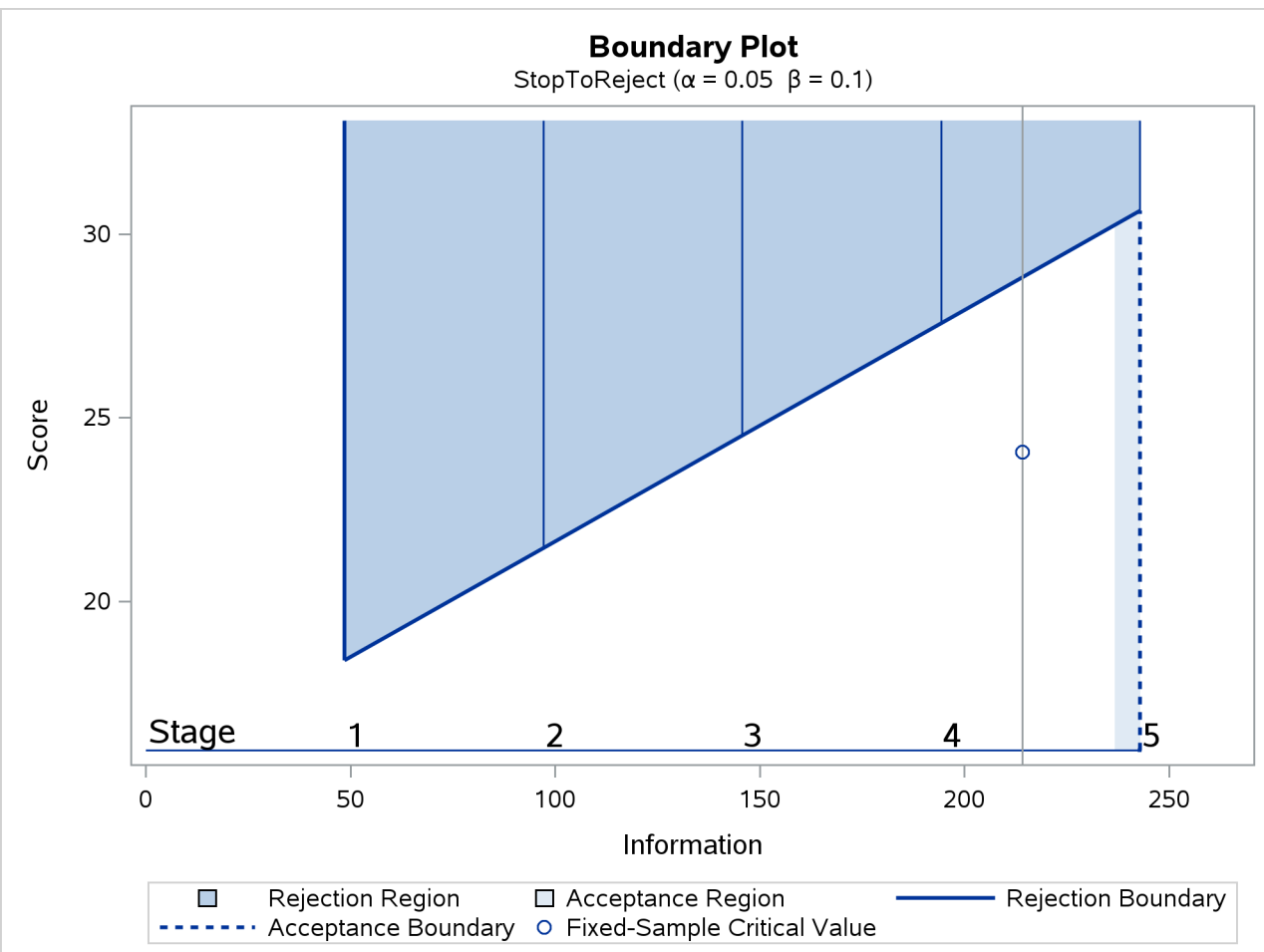

The third design has early stopping to accept the null hypothesis H_0.

The "Design Information" table in Output 110.6.11 displays design specifications and derived statistics. With the specified alternative reference, the maximum information is derived.

Output 110.6.11 Triangular Design Information

The SEQDESIGN Procedure
Design: StopToAccept

Design Information	
Statistic Distribution	Normal
Boundary Scale	Score
Alternative Hypothesis	Upper
Early Stop	Accept Null
Method	Triangular
Boundary Key	Both
Alternative Reference	0.2
Number of Stages	5
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	114.9925
Max Information	246.1945
Null Ref ASN (Percent of Fixed Sample)	57.83208
Alt Ref ASN (Percent of Fixed Sample)	110.2477

The "Method Information" table in Output 110.6.12 displays the α and β errors and the derived drift parameter. The table also shows the corresponding parameters for a triangular method as a unified family method.

Output 110.6.12 Method Information

Method Information								
				Unified Family				
Boundary	**Method**	**Alpha**	**Beta**	**Rho**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Beta	Triangular	0.05000	0.10000	0.5	1	0.82154	0.2	3.138117

The "Boundary Information" table in Output 110.6.13 displays information level, alternative reference, and boundary values. With the specified BOUNDARYSCALE=SCORE option, the alternative reference and boundary values are displayed in the score statistic scale.

Output 110.6.13 Boundary Information

Boundary Information (Score Scale) Null Reference = 0				
			Alternative	Boundary Values
	Information Level		Reference	Upper
Stage	Proportion	Actual	Upper	Beta
1	0.2000	49.2389	9.84778	-5.62074
2	0.4000	98.4778	19.69556	1.64895
3	0.6000	147.7167	29.54334	8.91865
4	0.8000	196.9556	39.39112	16.18834
5	1.0000	246.1945	49.23890	23.45803

The "Error Spending Information" table in Output 110.6.14 displays cumulative error spending at each stage for each boundary.

Output 110.6.14 Error Spending Information

Error Spending Information			
		Cumulative Error Spending	
	Information Level	Upper	
Stage	Proportion	Beta	Alpha
1	0.2000	0.01375	0.00000
2	0.4000	0.04149	0.00000
3	0.6000	0.06594	0.00000
4	0.8000	0.08513	0.00000
5	1.0000	0.10000	0.05000

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.6.15. For a triangular design, these rejection boundaries form a straight line with the score scale.

Output 110.6.15 Boundary Plot with Score Scale

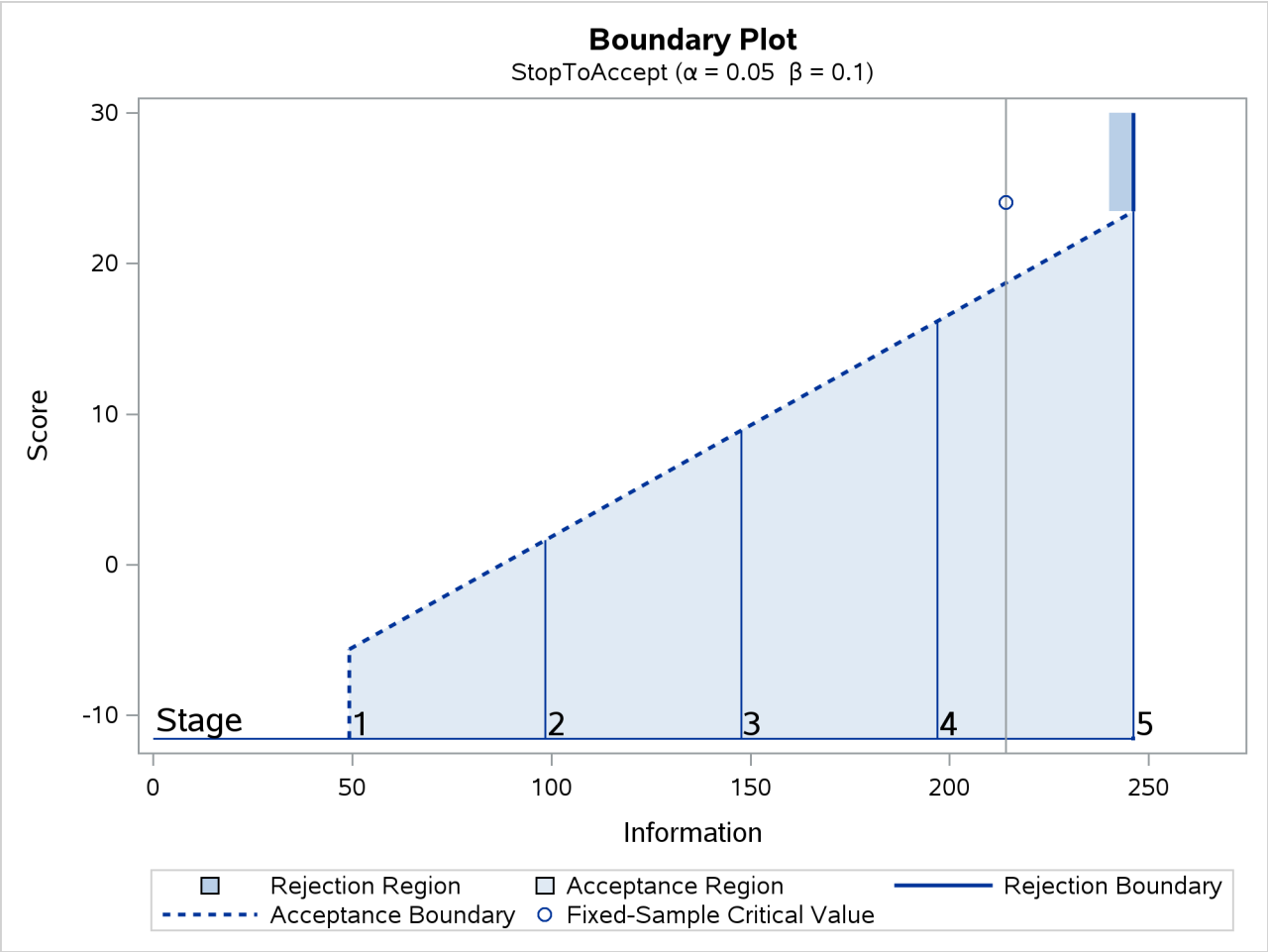

With the PLOTS=COMBINEDBOUNDARY option, a plot of the resulting sequential boundaries for all designs is displayed, as shown in Output 110.6.16. The plot shows that the design with early stopping to reject and to accept H_0 has larger maximum information than the other two designs.

Output 110.6.16 Combined Boundary Plot with Score Scale

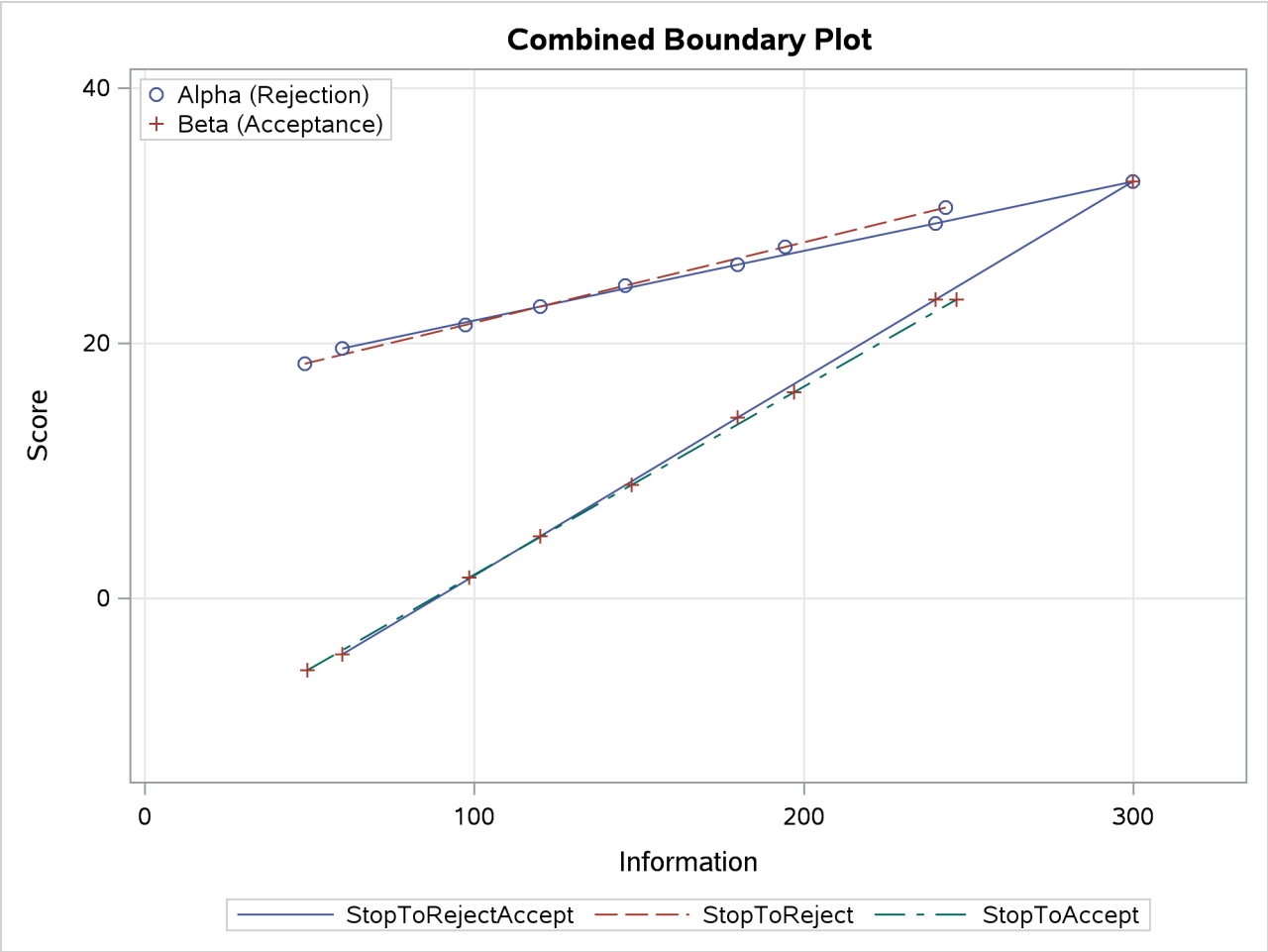

With the PLOTS=ERRSPEND(HSCALE=INFO) option, the error spending plot is displayed with the information level on the horizontal axis, as shown in Output 110.6.17. The design with early stopping to reject or accept the null hypothesis H_0 has larger α spending and larger β spending in early stages than the other two designs.

Output 110.6.17 Error Spending Plot

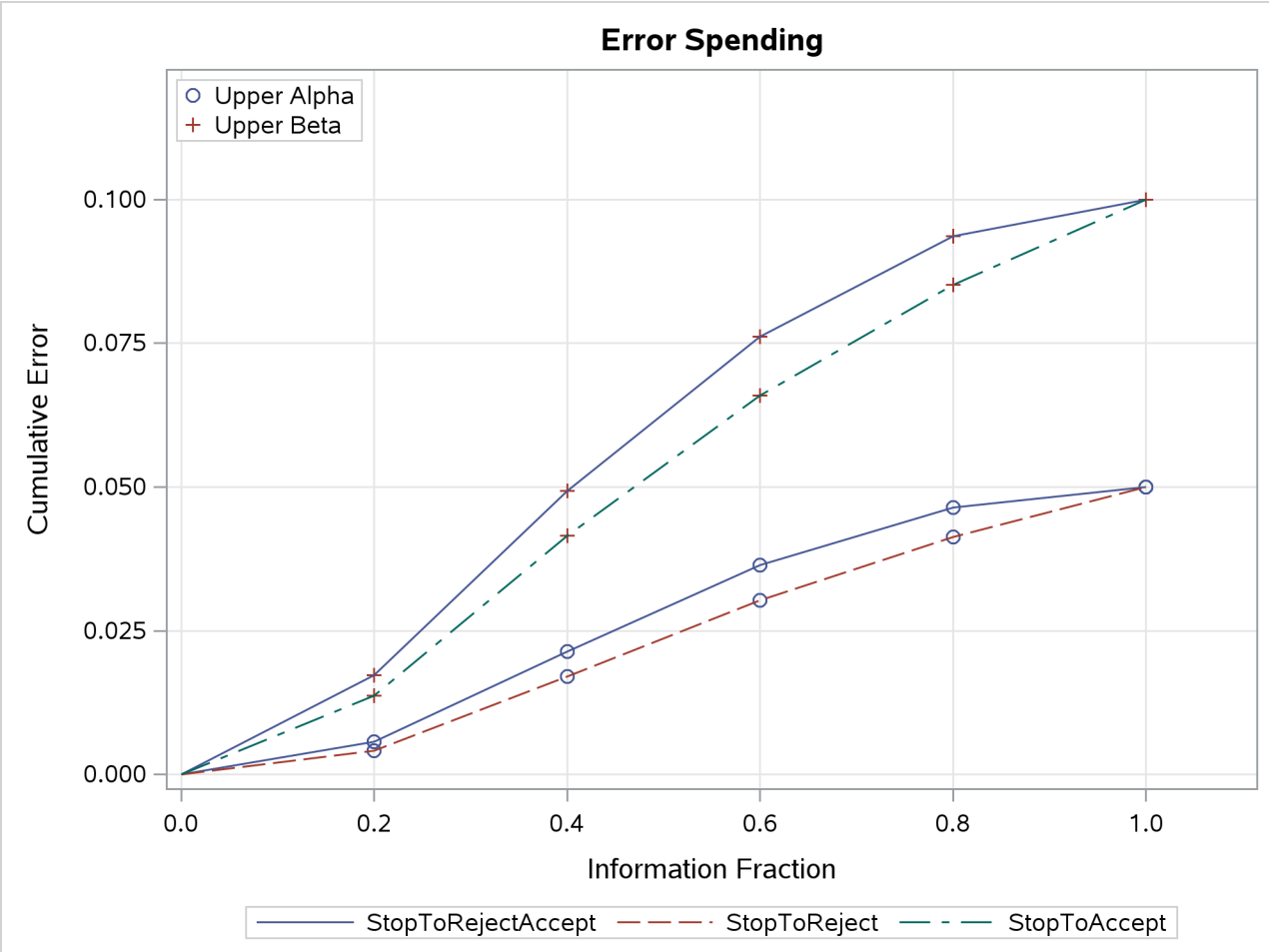

Example 110.7: Creating Whitehead's Triangular Designs

(View the complete code for this example (seqdex7.sas) in the example repository.)

This example requests three 4-stage Whitehead's triangular designs for normally distributed statistics. Each design has a one-sided alternative hypothesis $H_1 : \theta_1 = 0.693147$ with early stopping to reject or accept the null hypothesis H_0. Note that Whitehead's triangular designs are different from unified family triangular designs.

The following statements invoke the SEQDESIGN procedure and specify three Whitehead's triangular designs:

```
ods graphics on;
proc seqdesign altref=0.693147
               bscale=score
               plots=combinedboundary
               ;
   BoundaryKeyNone:  design nstages=4
                            method=whitehead
```

```
                              boundarykey=none
                              alt=upper   stop=both
                              alpha=0.05  beta=0.20
                              ;
    BoundaryKeyAlpha: design nstages=4
                              method=whitehead
                              boundarykey=alpha
                              alt=upper   stop=both
                              alpha=0.05  beta=0.20
                              ;

    BoundaryKeyBeta:  design nstages=4
                              method=whitehead
                              boundarykey=beta
                              alt=upper   stop=both
                              alpha=0.05  beta=0.20
                              ;
  run;
```

Whitehead methods with early stopping to reject or accept the null hypothesis create boundaries that approximately satisfy the Type I and Type II error probability specification. The BOUNDARYKEY=NONE option specifies no adjustment to the boundary value at the final stage to maintain either a Type I or a Type II error probability level.

The "Design Information" table in Output 110.7.1 displays design specifications and maximum information. Note that with the BOUNDARYKEY=NONE option, the derived errors $\alpha = 0.05071$ and $\beta = 0.19771$ are not the same as the specified errors $\alpha = 0.05$ and $\beta = 0.20$.

Output 110.7.1 Whitehead Design Information

The SEQDESIGN Procedure
Design: BoundaryKeyNone

Design Information	
Statistic Distribution	Normal
Boundary Scale	Score
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Method	Whitehead
Boundary Key	None
Alternative Reference	0.693147
Number of Stages	4
Alpha	0.05071
Beta	0.19771
Power	0.80229
Max Information (Percent of Fixed Sample)	129.6815
Max Information	16.70639
Null Ref ASN (Percent of Fixed Sample)	62.48184
Alt Ref ASN (Percent of Fixed Sample)	73.82535

The "Method Information" table in Output 110.7.2 displays the derived α and β errors and the derived drift parameter. The derived errors $\alpha = 0.05071$ and $\beta = 0.19771$ are not exactly the same as the specified errors

$\alpha = 0.05$ and $\beta = 0.20$ with the BOUNDARYKEY=NONE option.

Output 110.7.2 Method Information

Method Information							
				Whitehead			
Boundary	Method	Alpha	Beta	Tau	C	Alternative Reference	Drift
Upper Alpha	Whitehead	0.05071	.	0.25	4.60517	0.693147	2.833131
Upper Beta	Whitehead	.	0.19771	0.25	4.60517	0.693147	2.833131

The "Boundary Information" table in Output 110.7.3 displays information level, alternative reference, and boundary values. With the specified BOUNDARYSCALE=SCORE option, the alternative reference and boundary values are displayed with the score statistics scale.

Output 110.7.3 Boundary Information

Boundary Information (Score Scale) Null Reference = 0					
	Information Level		Alternative Reference	Boundary Values	
				Upper	
Stage	Proportion	Actual	Upper	Beta	Alpha
1	0.2500	4.176597	2.89500	-0.95755	4.78775
2	0.5000	8.353195	5.78999	1.91510	5.74530
3	0.7500	12.52979	8.68499	4.78775	6.70285
4	1.0000	16.70639	11.57998	7.66039	7.66039

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.7.4.

Output 110.7.4 Boundary Plot

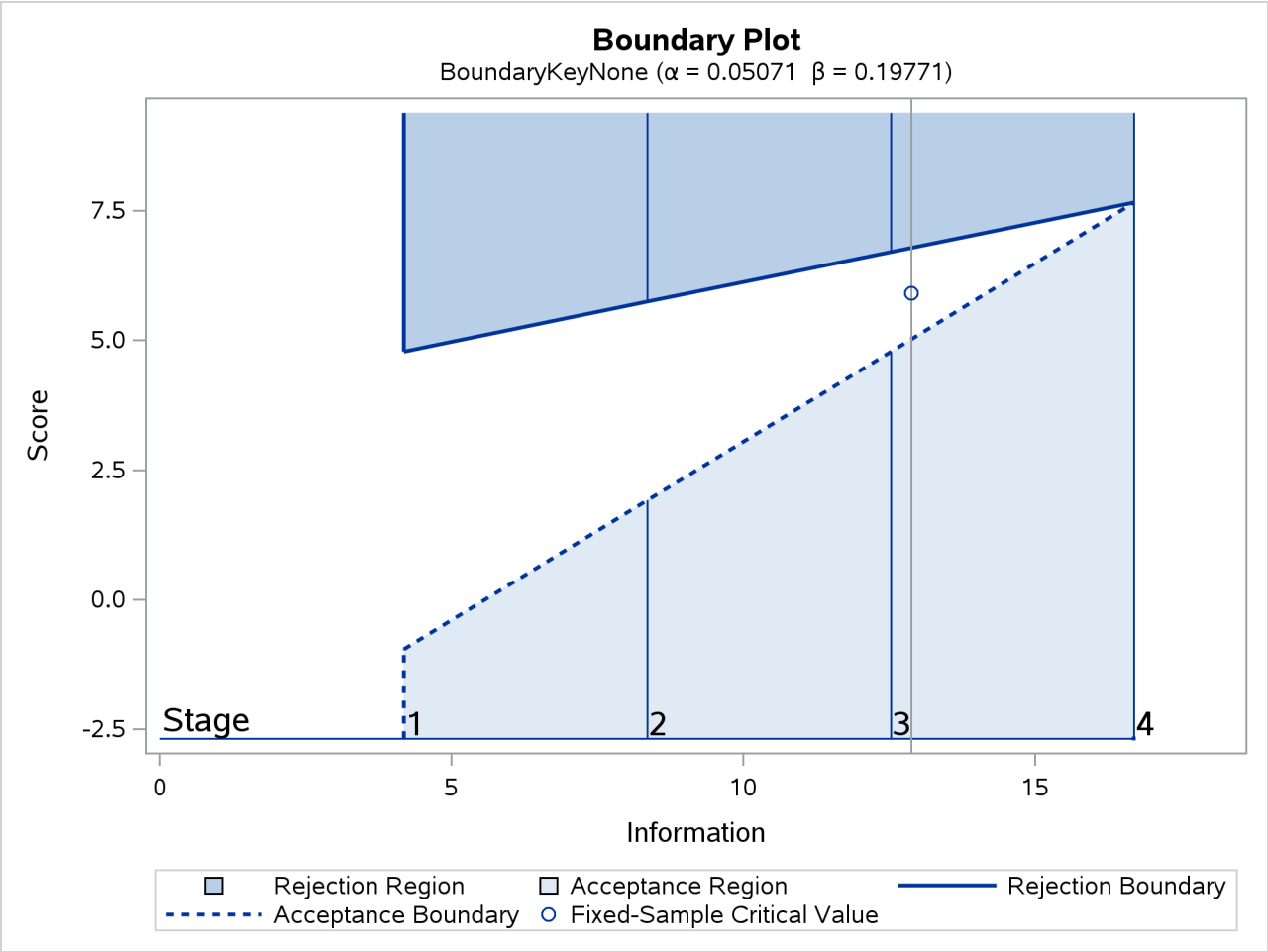

The second design uses the BOUNDARYKEY=ALPHA option to adjust the boundary value at the final stage to maintain the Type I error probability level.

The "Design Information" table in Output 110.7.5 displays design specifications and the derived maximum information. Note that with the BOUNDARYKEY=ALPHA option, the specified Type I error probability $\alpha = 0.05$ is maintained.

Output 110.7.5 Whitehead Design Information

The SEQDESIGN Procedure
Design: BoundaryKeyAlpha

Design Information	
Statistic Distribution	Normal
Boundary Scale	Score
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Method	Whitehead
Boundary Key	Alpha
Alternative Reference	0.693147
Number of Stages	4
Alpha	0.05
Beta	0.20044
Power	0.79956
Max Information (Percent of Fixed Sample)	129.9894
Max Information	16.70639
Null Ref ASN (Percent of Fixed Sample)	62.6302
Alt Ref ASN (Percent of Fixed Sample)	74.00064

The "Method Information" table in Output 110.7.6 displays the specified and derived α and β errors and the derived drift parameter. The derived Type I error probability is the same as the specified $\alpha = 0.05$ and the derived Type II error probability $\beta = 0.20044$ is not the same as the specified $\beta = 0.20$ with the BOUNDARYKEY=ALPHA option.

Output 110.7.6 Method Information

Method Information

				Whitehead			
Boundary	**Method**	**Alpha**	**Beta**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	Whitehead	0.05000	.	0.25	4.60517	0.693147	2.833131
Upper Beta	Whitehead	.	0.20044	0.25	4.60517	0.693147	2.833131

The "Boundary Information" table in Output 110.7.7 displays information level, alternative reference, and boundary values.

Output 110.7.7 Boundary Information

Boundary Information (Score Scale)
Null Reference = 0

	Information Level		Alternative Reference	Boundary Values Upper	
Stage	**Proportion**	**Actual**	**Upper**	**Beta**	**Alpha**
1	0.2500	4.176597	2.89500	-0.95755	4.78775
2	0.5000	8.353195	5.78999	1.91510	5.74530
3	0.7500	12.52979	8.68499	4.78775	6.70285
4	1.0000	16.70639	11.57998	7.81300	7.81300

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.7.8.

Output 110.7.8 Boundary Plot

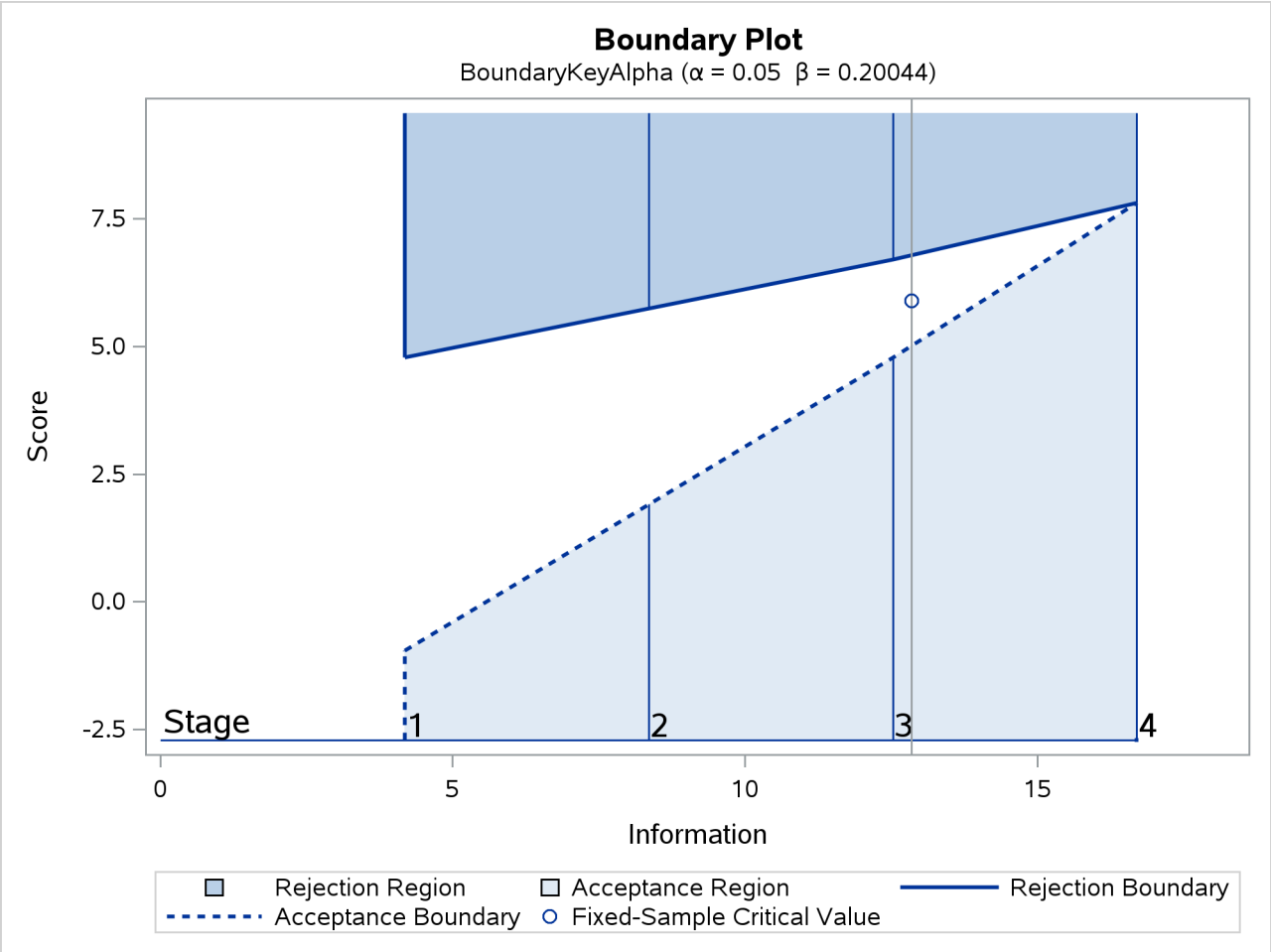

The third design specifies the BOUNDARYKEY=BETA option to derive the boundary values to maintain the Type II error probability level β.

The "Design Information" table in Output 110.7.9 displays design specifications and the derived maximum information. Note that with the BOUNDARYKEY=BETA option, the specified Type II error probability $\beta = 0.20$ is maintained.

Output 110.7.9 Whitehead Design Information

The SEQDESIGN Procedure
Design: BoundaryKeyBeta

Design Information	
Statistic Distribution	Normal
Boundary Scale	Score
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Method	Whitehead
Boundary Key	Beta
Alternative Reference	0.693147
Number of Stages	4
Alpha	0.05011
Beta	0.2
Power	0.8
Max Information (Percent of Fixed Sample)	129.9364
Max Information	16.70639
Null Ref ASN (Percent of Fixed Sample)	62.60462
Alt Ref ASN (Percent of Fixed Sample)	73.97042

The "Method Information" table in Output 110.7.10 displays the α and β errors and the derived drift parameter. The derived Type II error probability is the same as the specified $\beta = 0.20$ and the derived Type I error probability $\alpha = 0.05011$ is not the same as the specified $\alpha = 0.05$ with the BOUNDARYKEY=BETA option.

Output 110.7.10 Method Information

Method Information							
				Whitehead			
Boundary	**Method**	**Alpha**	**Beta**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	Whitehead	0.05011	.	0.25	4.60517	0.693147	2.833131
Upper Beta	Whitehead	.	0.20000	0.25	4.60517	0.693147	2.833131

The "Boundary Information" table in Output 110.7.11 displays information level, alternative reference, and boundary values.

Output 110.7.11 Boundary Information

Boundary Information (Score Scale) Null Reference = 0					
	Information Level		Alternative Reference	Boundary Values Upper	
Stage	**Proportion**	**Actual**	**Upper**	**Beta**	**Alpha**
1	0.2500	4.176597	2.89500	-0.95755	4.78775
2	0.5000	8.353195	5.78999	1.91510	5.74530
3	0.7500	12.52979	8.68499	4.78775	6.70285
4	1.0000	16.70639	11.57998	7.78899	7.78899

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.7.12.

Output 110.7.12 Boundary Plot

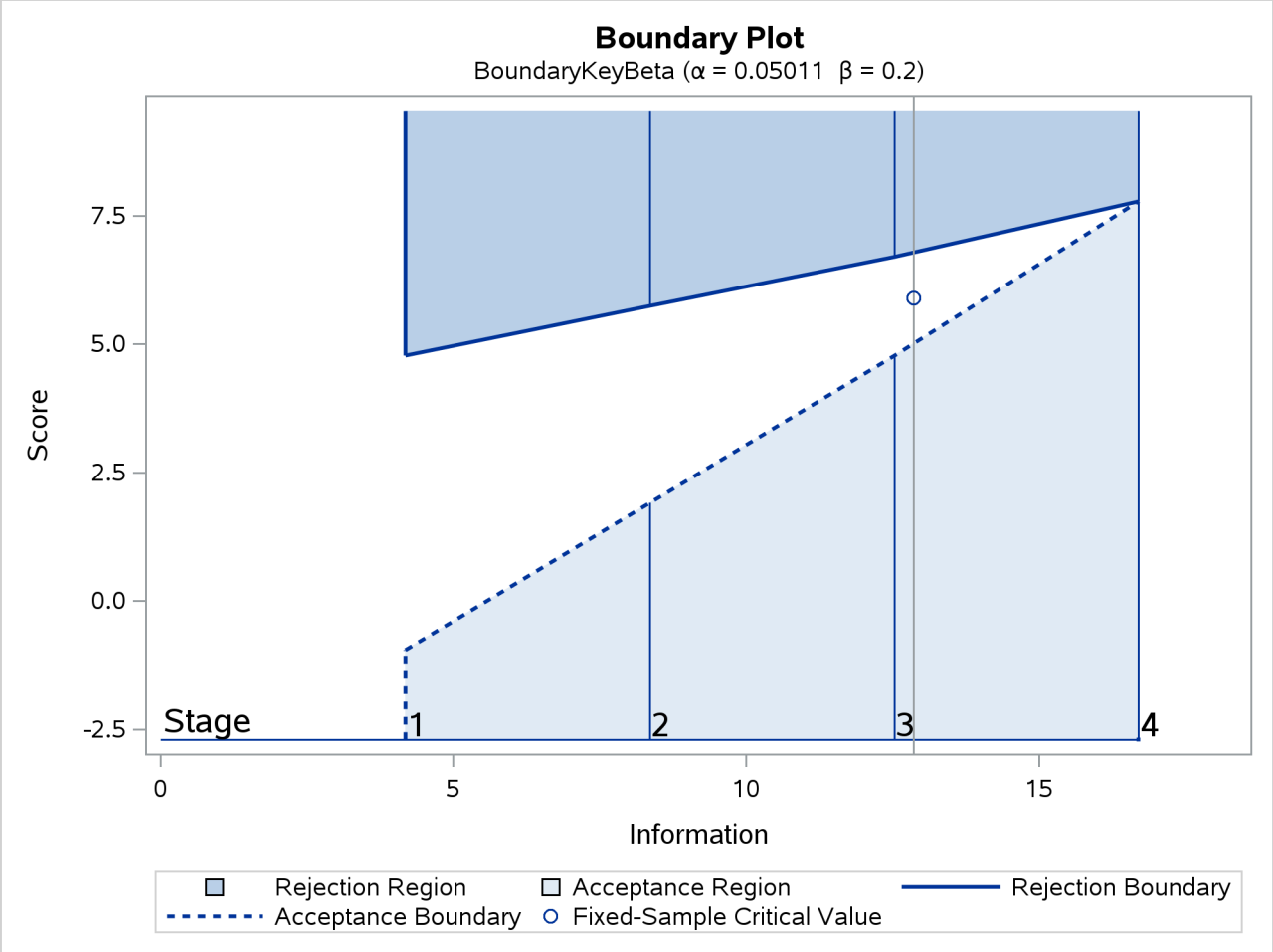

With the PLOTS=COMBINEDBOUNDARY option, a combined plot of group sequential boundaries for all designs is displayed, as shown in Output 110.7.13. It shows that three designs are similar, with a slightly smaller boundary value at the final stage for the design with the BOUNDARYKEY=NONE option.

Output 110.7.13 Combined Boundary Plot

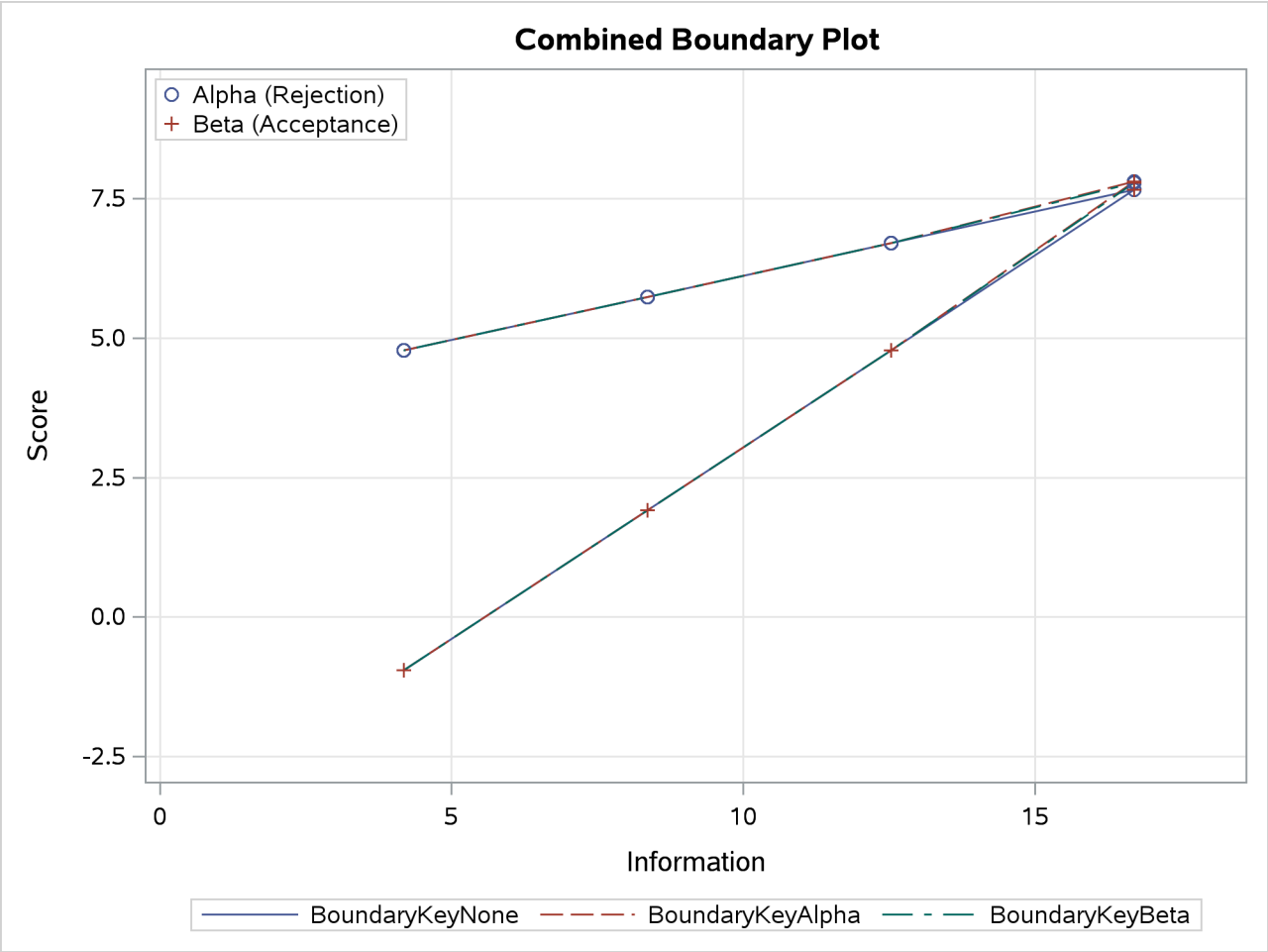

Example 110.8: Creating a One-Sided Error Spending Design

(View the complete code for this example (seqdex8.sas) in the example repository.)

This example requests a five-stage, one-sided group sequential design for normally distributed statistics. The design uses an O'Brien-Fleming-type error spending function for the α boundary and a Pocock-type error spending function for the β boundary. The following statements request a one-sided design by using different α and β spending functions:

```
ods graphics on;
proc seqdesign altref=0.2  errspend
               pss(cref=0 0.5 1)
               stopprob(cref=0 0.5 1)
               plots=(asn power errspend)
               ;
   OneSidedErrorSpending: design nstages=5
                          method(alpha)=errfuncobf
                          method(beta)=errfuncpoc
                          alt=upper  stop=both
```

```
                                alpha=0.025
                                ;
run;
```

The "Design Information" table in Output 110.8.1 displays design specifications and the derived statistics. With the specified alternative reference, the maximum information is derived.

Output 110.8.1 Error Spending Method Design Information

The SEQDESIGN Procedure
Design: OneSidedErrorSpending

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.2
Number of Stages	5
Alpha	0.025
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	119.4278
Max Information	313.7196
Null Ref ASN (Percent of Fixed Sample)	50.35408
Alt Ref ASN (Percent of Fixed Sample)	78.77223

The "Method Information" table in Output 110.8.2 displays the α and β errors, alternative reference, and derived drift parameter, which is the standardized alternative reference at the final stage.

Output 110.8.2 Method Information

Method Information						
				Error Spending		
Boundary	**Method**	**Alpha**	**Beta**	**Function**	**Alternative Reference**	**Drift**
Upper Alpha	Error Spending	0.02500	.	Approx O'Brien-Fleming	0.2	3.542426
Upper Beta	Error Spending	.	0.10000	Approx Pocock	0.2	3.542426

With the STOPPROB option, the "Expected Cumulative Stopping Probabilities" table in Output 110.8.3 displays the expected stopping stage and cumulative stopping probability to reject the null hypothesis at each stage under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and c_i are values specified in the CREF= option.

Output 110.8.3 Stopping Probabilities

Expected Cumulative Stopping Probabilities Reference = CRef * (Alt Reference)							
			Stopping Probabilities				
CRef	Expected Stopping Stage	Source	Stage_1	Stage_2	Stage_3	Stage_4	Stage_5
0.0000	2.108	Reject Null	0.00000	0.00039	0.00381	0.01221	0.02500
0.0000	2.108	Accept Null	0.38080	0.69133	0.86162	0.94170	0.97500
0.0000	2.108	Total	0.38080	0.69173	0.86543	0.95391	1.00000
0.5000	3.296	Reject Null	0.00002	0.01265	0.09650	0.24465	0.38724
0.5000	3.296	Accept Null	0.13665	0.28063	0.41080	0.52230	0.61276
0.5000	3.296	Total	0.13667	0.29328	0.50730	0.76695	1.00000
1.0000	3.298	Reject Null	0.00050	0.13209	0.52642	0.80390	0.90000
1.0000	3.298	Accept Null	0.02954	0.05231	0.07085	0.08648	0.10000
1.0000	3.298	Total	0.03004	0.18440	0.59728	0.89039	1.00000

With the PSS option, the "Power and Expected Sample Sizes" table in Output 110.8.4 displays powers and expected sample sizes under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and $c_i = 0, 0.5, 1, 1.5$ are the default values in the CREF= option.

Output 110.8.4 Power and Expected Sample Size Information

Powers and Expected Sample Sizes Reference = CRef * (Alt Reference)		
		Sample Size
CRef	Power	Percent Fixed-Sample
0.0000	0.02500	50.3541
0.5000	0.38724	78.7219
1.0000	0.90000	78.7722

With the PLOTS=ASN option, the procedure displays a plot of expected sample sizes under various hypothetical references, as shown in Output 110.8.5. By default, expected sample sizes under the hypotheses $\theta = c_i\,\theta_1$, $c_i = 0, 0.01, 0.02, \ldots, 1.50$, are displayed, where θ_1 is the alternative reference.

Output 110.8.5 ASN Plot

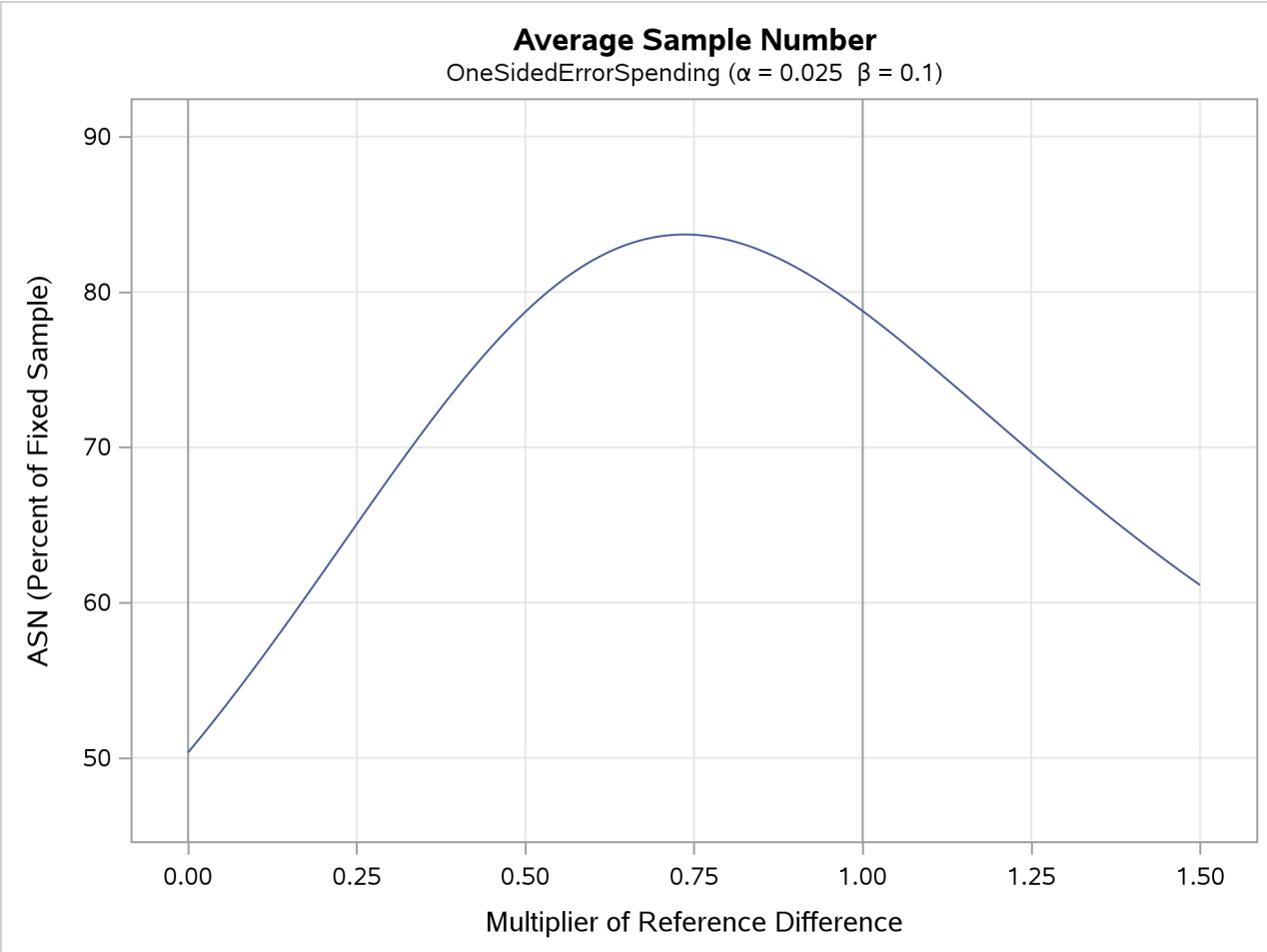

With the PLOTS=POWER option, the procedure displays a plot of the power curves under various hypothetical references for all designs simultaneously, as shown in Output 110.8.6. By default, the option CREF= $0, 0.01, 0.02, \ldots, 1.50$ and powers under hypothetical references $\theta = c_i\,\theta_1$ are displayed, where c_i are values specified in the CREF= option. These CREF= values are displayed on the horizontal axis.

Under the null hypothesis, $c_i = 0$, the power is 0.025, the upper Type I error probability. Under the alternative hypothesis, $c_i = 1$, the power is 0.9, one minus the Type II error probability. The plot shows only minor difference between the two designs.

Output 110.8.6 Power Plot

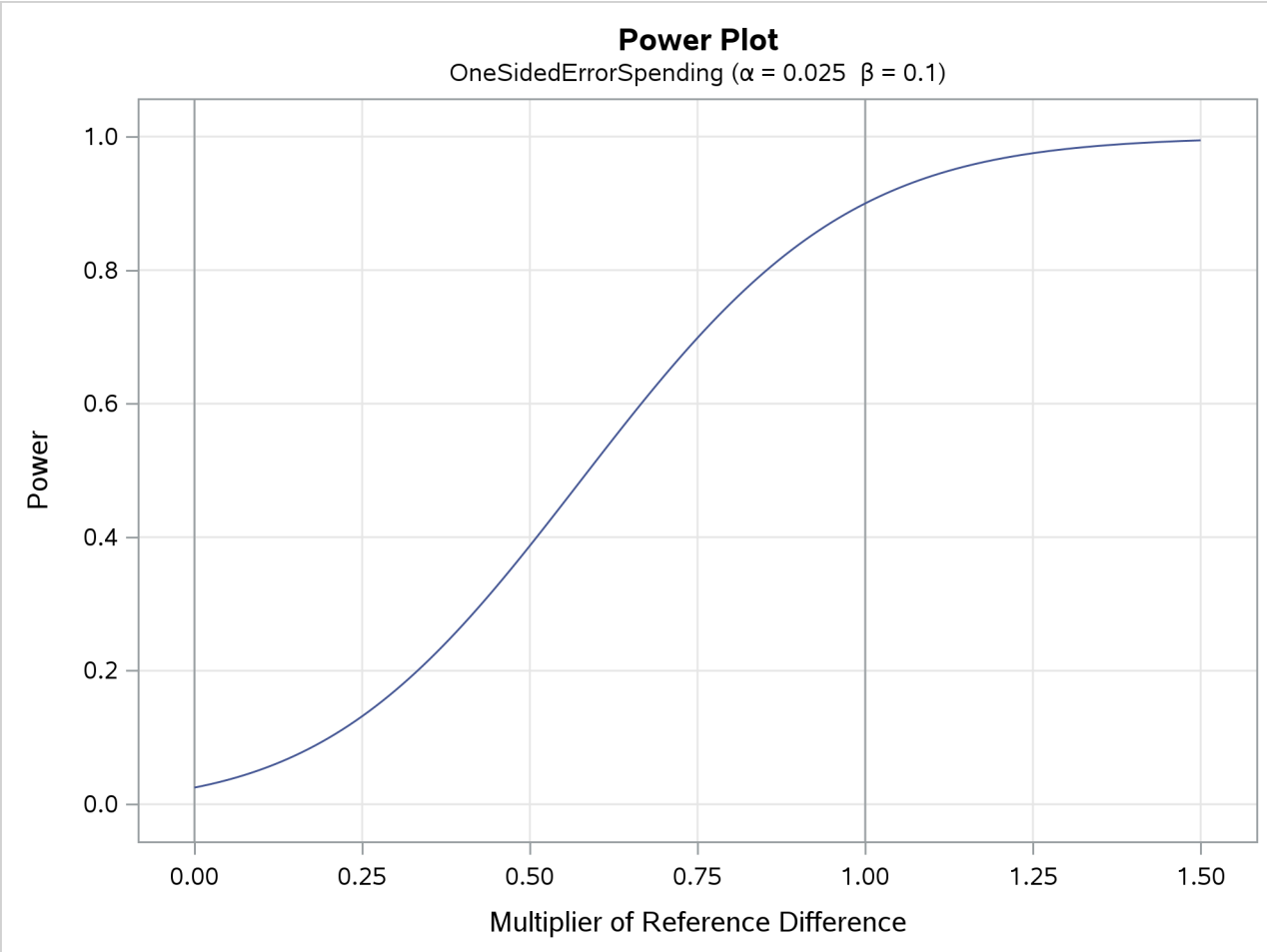

The "Boundary Information" table in Output 110.8.7 displays information level, alternative reference, and boundary values. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the alternative reference and boundary values are displayed with the standardized Z scale. That is, the resulting standardized alternative reference at stage k is given by $\theta_1\sqrt{I_k}$, where θ_1 is the specified alternative reference and I_k is the information level at stage k, $k = 1, 2, \ldots, 5$.

Output 110.8.7 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference	Boundary Values	
				Upper	
Stage	Proportion	Actual	Upper	Beta	Alpha
1	0.2000	62.74393	1.58422	-0.30338	4.87688
2	0.4000	125.4879	2.24043	0.41667	3.35706
3	0.6000	188.2318	2.74395	0.97165	2.67766
4	0.8000	250.9757	3.16844	1.43627	2.26535
5	1.0000	313.7196	3.54243	1.87522	1.87522

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.8.8. This plot displays the boundary values in the "Boundary Information" table in Output 110.8.7.

Output 110.8.8 Boundary Plot

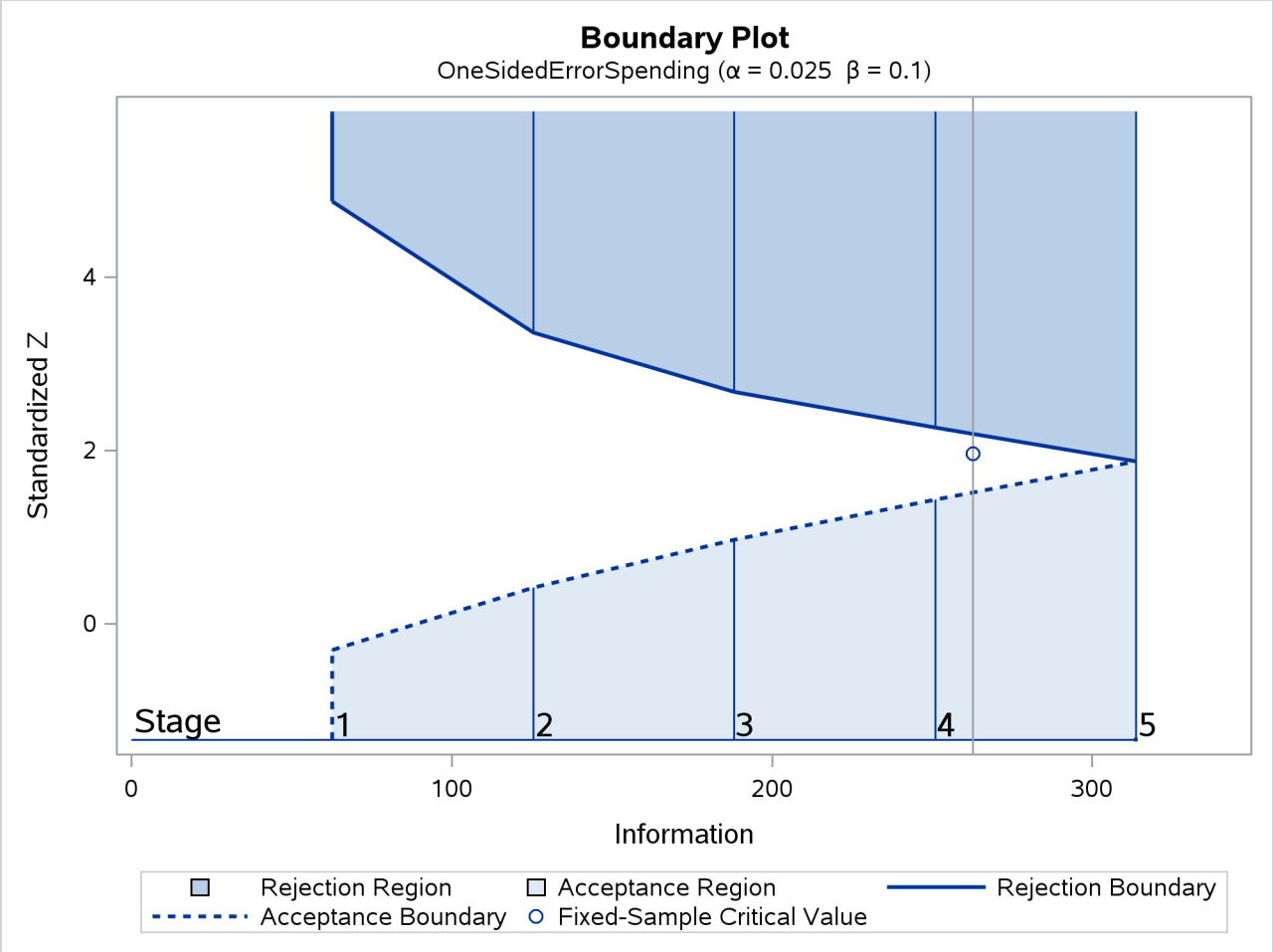

The "Error Spending Information" table in Output 110.8.9 displays cumulative error spending at each stage for each boundary.

Output 110.8.9 Error Spending Information

Error Spending Information			
		Cumulative Error Spending	
	Information Level	Upper	
Stage	Proportion	Beta	Alpha
1	0.2000	0.02954	0.00000
2	0.4000	0.05231	0.00039
3	0.6000	0.07085	0.00381
4	0.8000	0.08648	0.01221
5	1.0000	0.10000	0.02500

With the PLOTS=ERRSPEND option, the procedure displays a plot of error spending for each boundary, as shown in Output 110.8.10. This plot displays the cumulative error spending at each stage in the "Error Spending Information" table in Output 110.8.9. The O'Brien-Fleming-type α spending function is conservative in early stages because it uses much less at early stages than in the later stages. In contrast, the Pocock-type β spending function uses more at early stages than in the later stages.

Output 110.8.10 Error Spending Plot

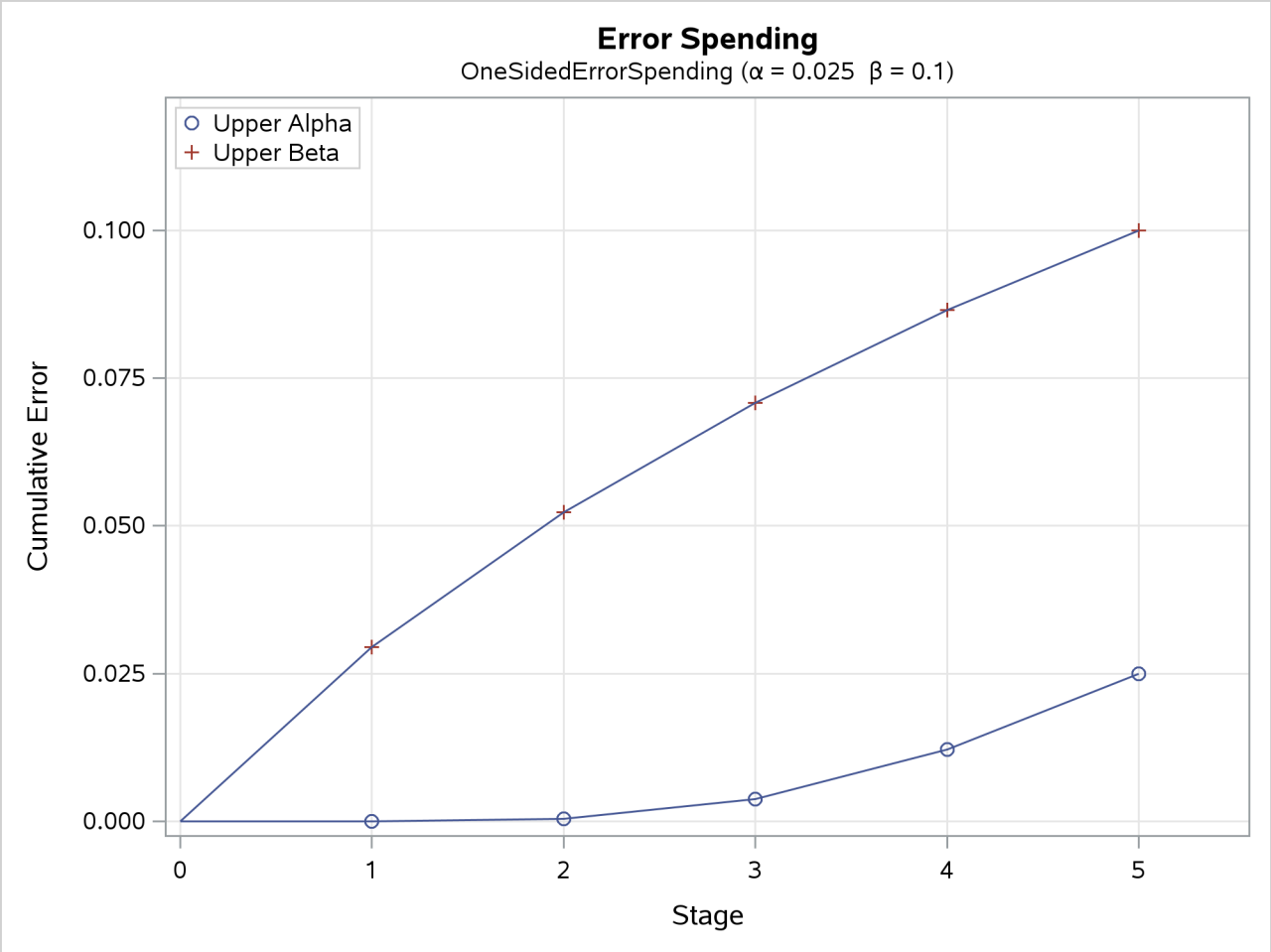

Example 110.9: Creating Designs with Various Number of Stages

(View the complete code for this example (seqdex9.sas) in the example repository.)

This example requests three group sequential designs for normally distributed statistics. Each design uses the power family error spending function with the default power parameter $\rho = 2$. The specified error spending method is between the approximated Pocock method ($\rho = 1$) and the approximated O'Brien-Fleming method ($\rho = 3$) (Jennison and Turnbull 2000, p. 148). The three designs are identical except for the specified number of stages. The following statements request these three group sequential designs:

```
ods graphics on;
proc seqdesign plots=( asn
                       power
                       combinedboundary
                       errspend(hscale=info)
                       )
                 ;
   TwoStageDesign:  design nstages=2
                    method=errfuncpow
                    alt=upper  stop=reject
                    ;
   FiveStageDesign: design nstages=5
                    method=errfuncpow
                    alt=upper  stop=reject
                    ;
   TenStageDesign:  design nstages=10
                    method=errfuncpow
                    alt=upper  stop=reject
                    ;
run;
```

The "Design Information" table in Output 110.9.1 displays design information for the two-stage design.

Output 110.9.1 Design Information

The SEQDESIGN Procedure
Design: TwoStageDesign

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Reject Null
Method	Error Spending
Boundary Key	Both
Number of Stages	2
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	102.4167
Null Ref ASN (Percent of Fixed Sample)	101.7766
Alt Ref ASN (Percent of Fixed Sample)	79.81021

The "Boundary Information" table in Output 110.9.2 displays the information level, alternative reference, and boundary values. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the alternative reference and boundary values are displayed with the standardized normal Z scale. The resulting standardized alternative reference at stage k is given by $\theta_1\sqrt{I_k}$, where θ_1 is the alternative reference and I_k is the information level at stage k, $k = 1, 2$.

Output 110.9.2 Boundary Information in Z Scale

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level	Alternative Reference	Boundary Values
Stage	Proportion	Upper	Upper Alpha
1	0.5000	2.09414	2.24140
2	1.0000	2.96156	1.69970

The "Design Information" table in Output 110.9.3 displays design information for the five-stage design. Compared with the two-stage design in Output 110.9.1, the maximum information increases from 102.42 to 105.62, and the average sample number under the alternative reference (Alt Ref ASN) decreases from 79.81 to 69.64.

Output 110.9.3 Design Information

The SEQDESIGN Procedure
Design: FiveStageDesign

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Reject Null
Method	Error Spending
Boundary Key	Both
Number of Stages	5
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	105.6235
Null Ref ASN (Percent of Fixed Sample)	104.356
Alt Ref ASN (Percent of Fixed Sample)	69.64322

The "Boundary Information" table in Output 110.9.4 displays the information level, alternative reference, and boundary values with the default standardized normal Z scale.

Output 110.9.4 Boundary Information in Z Scale

Boundary Information (Standardized Z Scale) Null Reference = 0			
		Alternative	Boundary Values
	Information Level	Reference	Upper
Stage	Proportion	Upper	Alpha
1	0.2000	1.34502	2.87816
2	0.4000	1.90215	2.47023
3	0.6000	2.32965	2.20095
4	0.8000	2.69005	1.98182
5	1.0000	3.00756	1.79024

The "Design Information" table in Output 110.9.5 displays design information for the ten-stage design. Compared with the five-stage design in Output 110.9.3, the maximum information increases further from 105.62 to 107.26 and under the alternative reference, the average sample number decreases further from 69.64 to 66.36.

Output 110.9.5 Design Information

The SEQDESIGN Procedure
Design: TenStageDesign

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Reject Null
Method	Error Spending
Boundary Key	Both
Number of Stages	10
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	107.256
Null Ref ASN (Percent of Fixed Sample)	105.7276
Alt Ref ASN (Percent of Fixed Sample)	66.35565

The "Boundary Information" table in Output 110.9.6 displays the information level, alternative reference, and boundary values with the default standardized normal Z scale.

Output 110.9.6 Boundary Information in Z Scale

Boundary Information (Standardized Z Scale) Null Reference = 0			
		Alternative	Boundary Values
	Information Level	Reference	Upper
Stage	Proportion	Upper	Alpha
1	0.1000	0.95840	3.29053
2	0.2000	1.35538	2.94037
3	0.3000	1.65999	2.72115
4	0.4000	1.91679	2.54808
5	0.5000	2.14304	2.40114
6	0.6000	2.34758	2.27127
7	0.7000	2.53568	2.15359
8	0.8000	2.71076	2.04503
9	0.9000	2.87519	1.94355
10	1.0000	3.03072	1.84765

With the PLOTS=ASN option, the procedure displays a plot of average sample numbers under various hypothetical references for all designs simultaneously, as shown in Output 110.9.7. By default, the option CREF= $0, 0.01, 0.02, \ldots, 1.50$ and expected sample sizes under the hypothetical references $\theta = c_i\,\theta_1$ are displayed, where c_i are values specified in the CREF= option. These CREF= values are displayed on the horizontal axis.

Output 110.9.7 ASN Plot

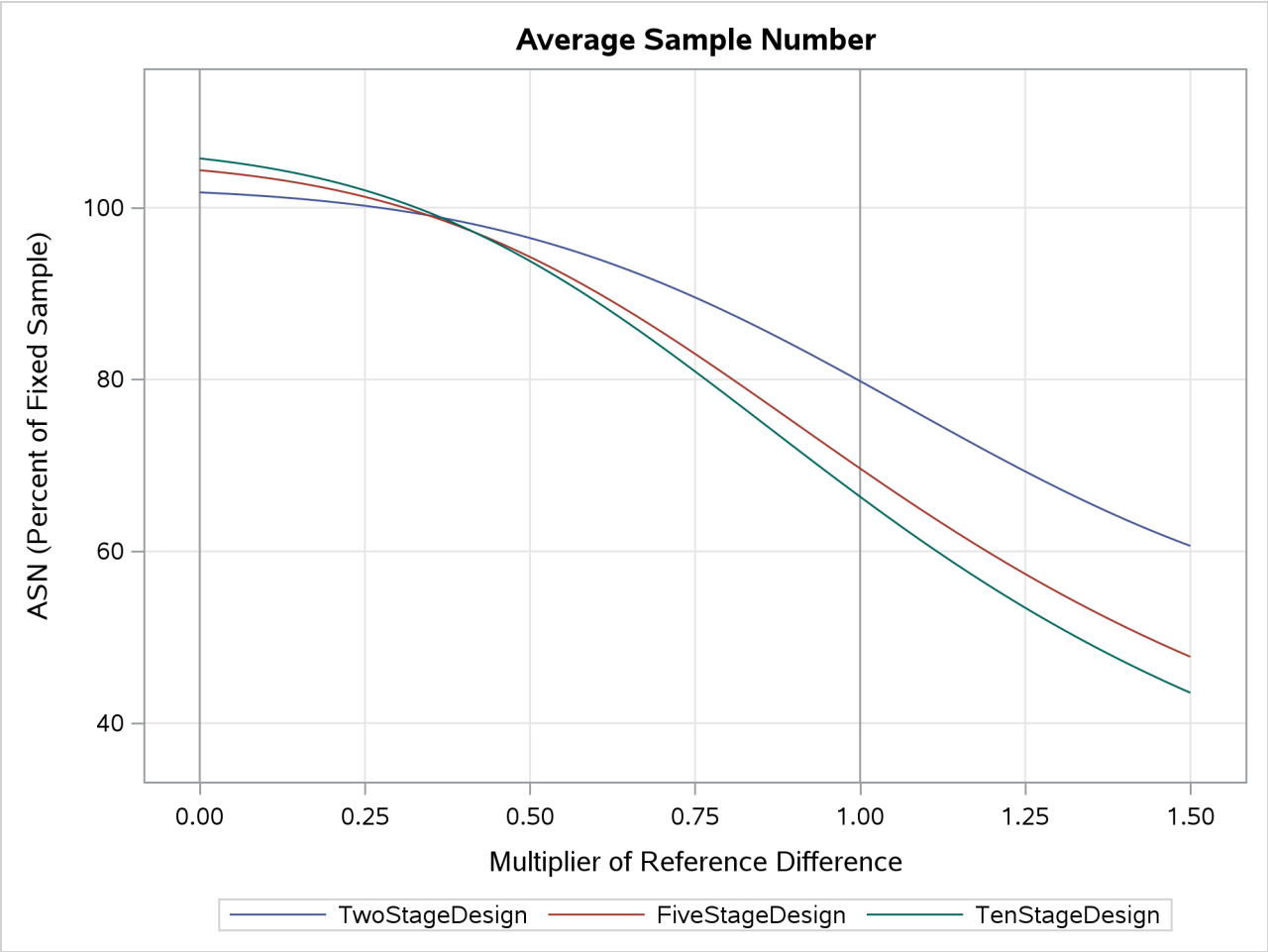

The plot shows that as the number of stages increases, the average sample number as a percentage of the fixed-sample design increases under the null hypothesis ($c_i = 0$) but decreases under the alternative hypothesis ($c_i = 1$).

With the PLOTS=POWER option, the procedure displays a plot of the power curves under various hypothetical references for all designs simultaneously, as shown in Output 110.9.8. By default, the option CREF= $0, 0.01, 0.02, \ldots, 1.50$ and powers under hypothetical references $\theta = c_i\,\theta_1$ are displayed, where c_i are values specified in the CREF= option. These CREF= values are displayed on the horizontal axis.

Output 110.9.8 Power Plot

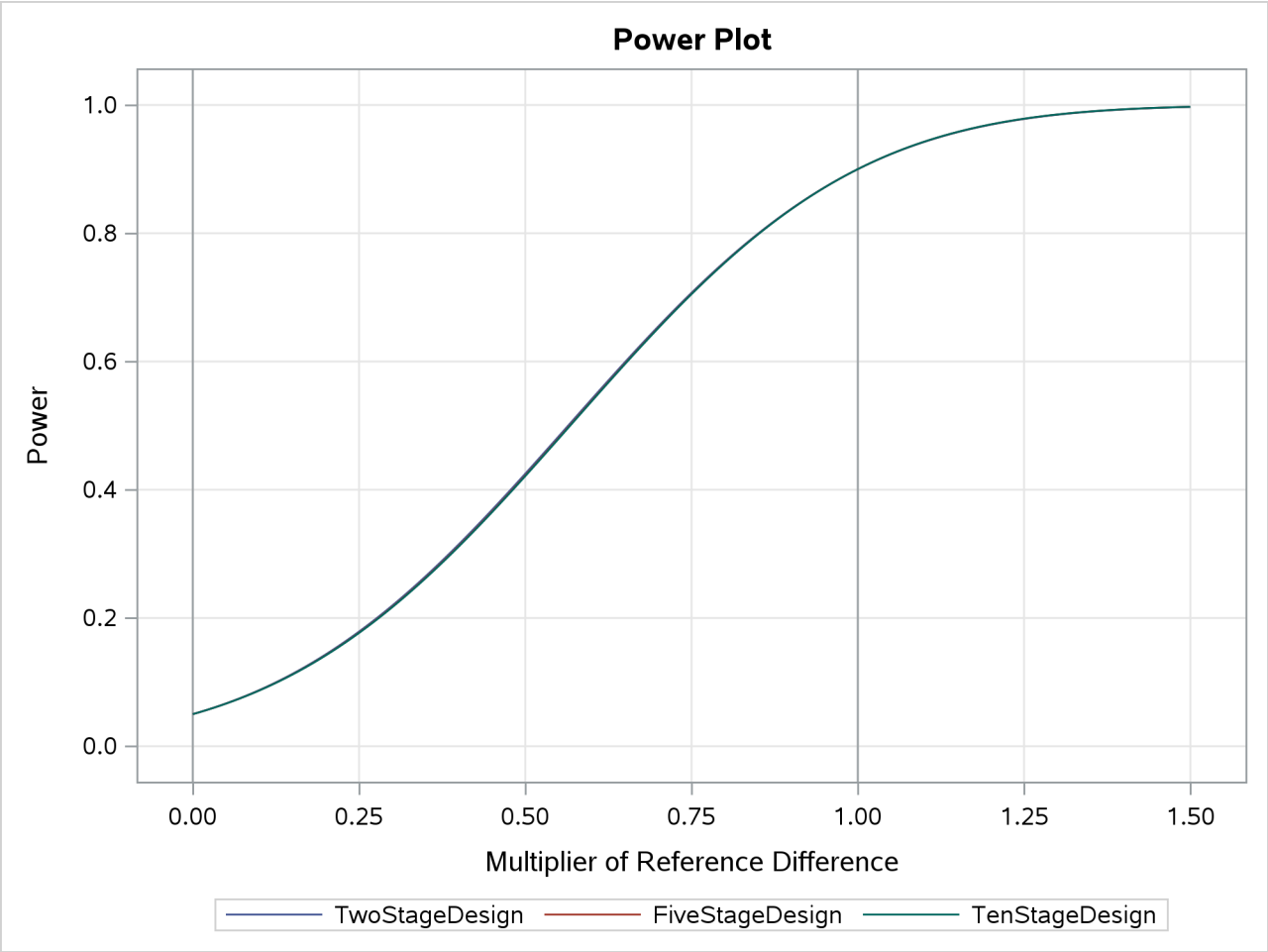

Under the null hypothesis, $c_i = 0$, the power is 0.05, the upper Type I error probability. Under the alternative hypothesis, $c_i = 1$, the power is 0.9, one minus the Type II error probability. The plot shows only minor difference among the three designs.

With the PLOTS=COMBINEDBOUNDARY option, the procedure displays a plot of sequential boundaries for all designs simultaneously, as shown in Output 110.9.9. By default (or equivalently if you specify COMBINEDBOUNDARY(HSCALE=INFO)), the information levels are used on the horizontal axis. Since the maximum information is not available for the design, the percent information ratios with respect to the corresponding fixed-sample design are displayed in the plot.

Output 110.9.9 Combined Boundary Plot

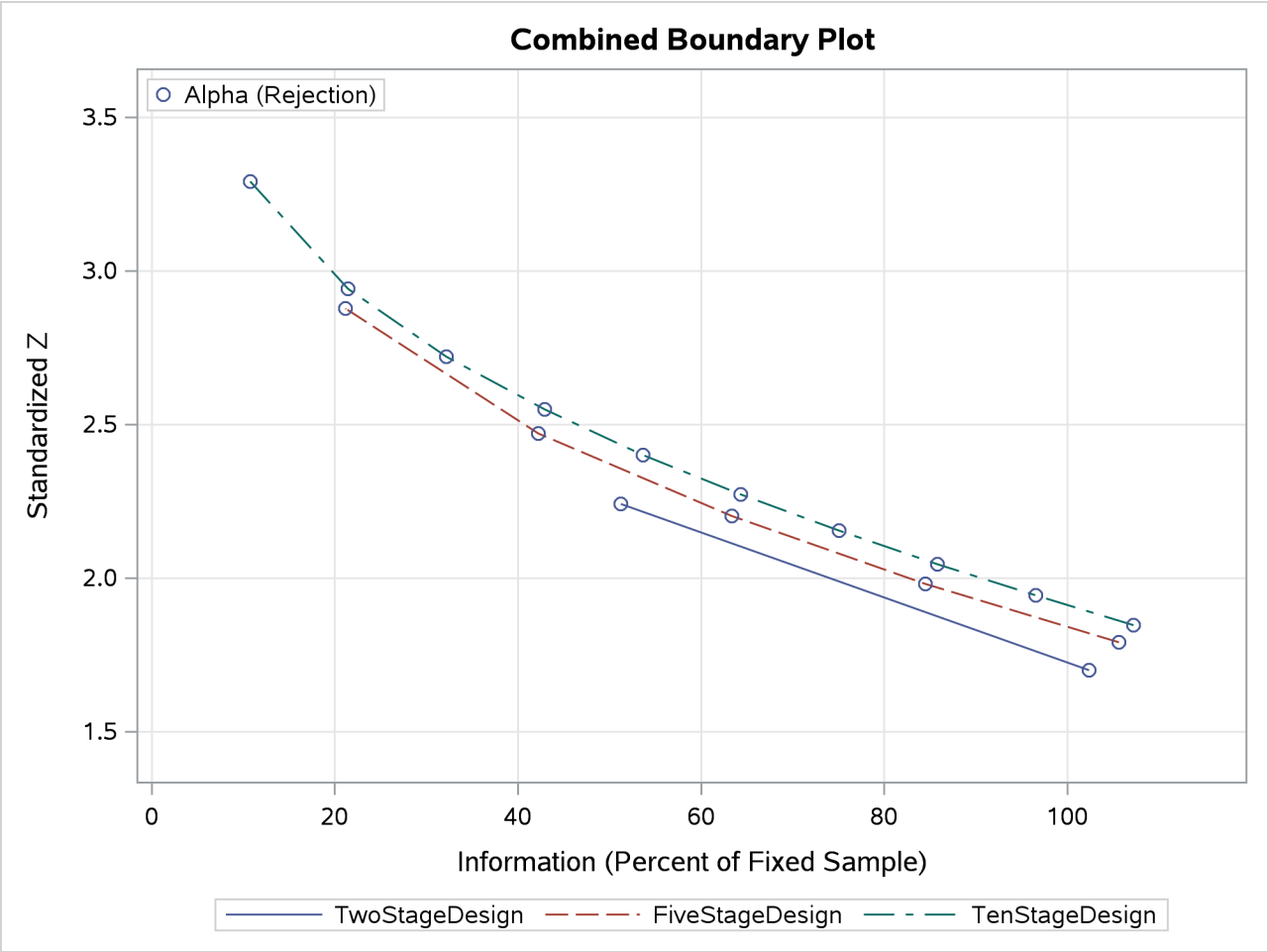

The plot shows that as the number of stages increases, the maximum information increases and the α boundary values also increase.

With the PLOTS=ERRSPEND(HSCALE=INFO) option, the procedure displays a plot of cumulative error spends for all boundaries in the designs simultaneously, as shown in Output 110.9.10.

Output 110.9.10 Error Spending Plot

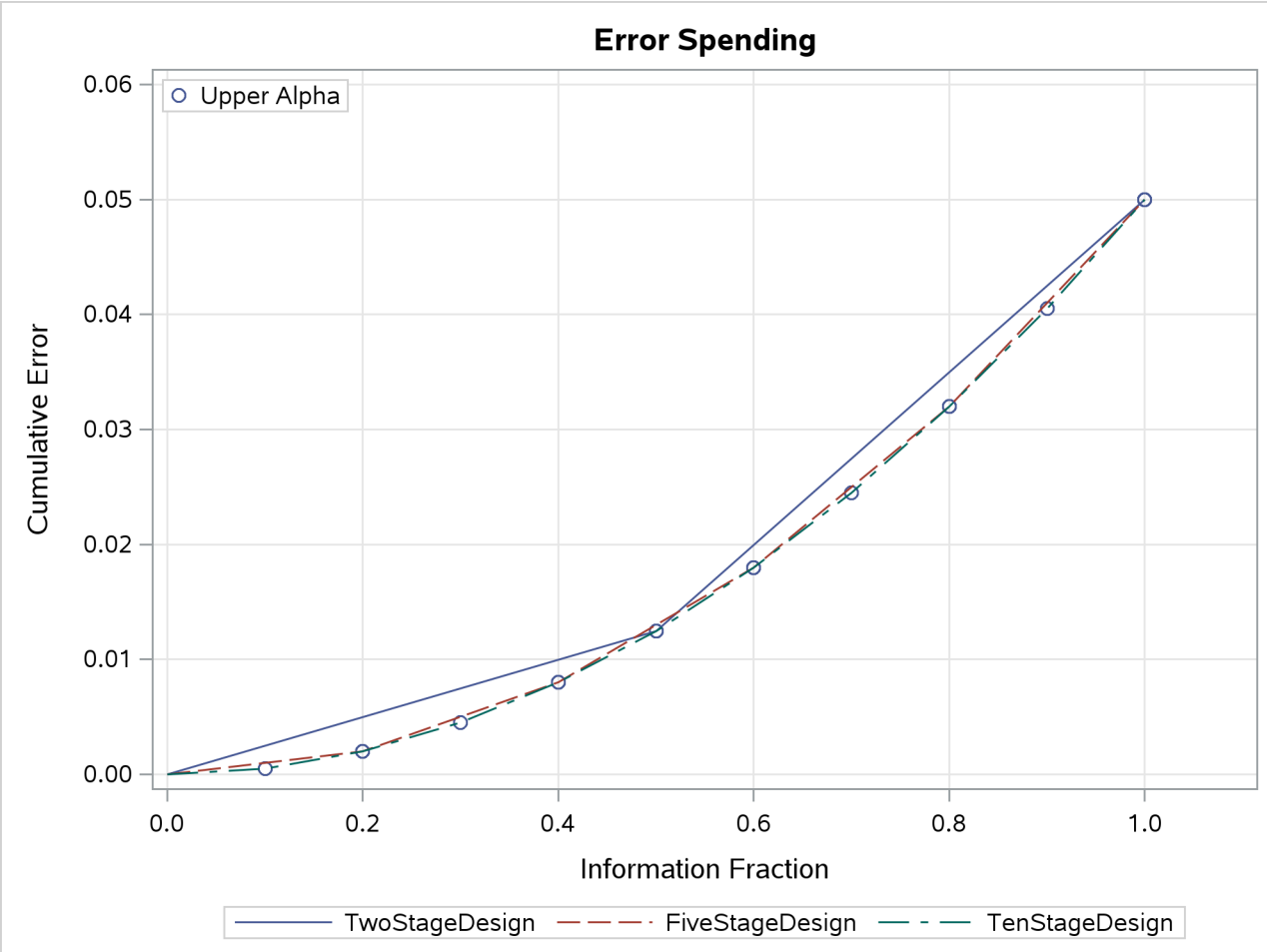

The plot shows similar error spending for these three designs since all three designs are generated from the same power family error spending function.

Example 110.10: Creating Two-Sided Error Spending Designs with and without Overlapping Lower and Upper β Boundaries

(View the complete code for this example (seqdx10.sas) in the example repository.)

This example requests two three-stage group sequential designs for normally distributed statistics. Each design uses a power family error spending function with a specified two-sided alternative hypothesis $H_1 : \theta_1 = \pm 0.2$ and early stopping only to accept the null hypothesis H_0.

The first design uses the BETAOVERLAP=NOADJUST option to derive acceptance boundary values without adjusting for the possible overlapping of the lower and upper β boundaries computed from the two corresponding one-sided tests. The second design uses the BETAOVERLAP=ADJUST option to test the overlapping of the β boundaries at each interim stage based on the two corresponding one-sided tests and then to set the β boundary values at the stage to missing if overlapping occurs at that stage.

The following statements request a two-sided design with the BETAOVERLAP=NOADJUST option:

```
ods graphics on;
proc seqdesign altref=0.2 errspend;
   design nstages=3
          method=errfuncpow
          alt=twosided  stop=accept
          betaoverlap=noadjust
          beta=0.09
          ;
run;
```

The "Design Information" table in Output 110.10.1 displays design specifications and the derived statistics for the first design. With the specified alternative reference $\theta_1 = 0.2$, the maximum information is derived.

Output 110.10.1 Design Information

The SEQDESIGN Procedure
Design: Design_1

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Accept Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.2
Number of Stages	3
Alpha	0.05
Beta	0.09
Power	0.91
Max Information (Percent of Fixed Sample)	103.8789
Max Information	282.9328
Null Ref ASN (Percent of Fixed Sample)	79.20197
Alt Ref ASN (Percent of Fixed Sample)	102.1476

The "Boundary Information" table in Output 110.10.2 displays the information level, alternative reference, and boundary values. With a specified alternative reference θ_1, the maximum information is derived from the procedure, and the actual information level at each stage is displayed in the table. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the alternative reference and boundary values are displayed with the standardized Z scale. The alternative reference at stage k is given by $\theta_1 \sqrt{I_k}$, where θ_1 is the specified alternative reference and I_k is the information level at stage k, $k = 1, 2, 3$.

Output 110.10.2 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0						
	Information Level		Alternative Reference		Boundary Values	
					Lower	Upper
Stage	Proportion	Actual	Lower	Upper	Beta	Beta
1	0.3333	94.31094	-1.94228	1.94228	-0.08239	0.08239
2	0.6667	188.6219	-2.74679	2.74679	-0.90351	0.90351
3	1.0000	282.9328	-3.36412	3.36412	-1.92519	1.92519

The "Error Spending Information" table in Output 110.10.3 displays the cumulative error spending at each stage for each boundary.

Output 110.10.3 Error Spending Information

Error Spending Information					
		Cumulative Error Spending			
	Information Level	Lower		Upper	
Stage	Proportion	Alpha	Beta	Beta	Alpha
1	0.3333	0.00000	0.01000	0.01000	0.00000
2	0.6667	0.00000	0.04000	0.04000	0.00000
3	1.0000	0.02500	0.09000	0.09000	0.02500

With the STOP=ACCEPT option, the design does not stop at interim stages to reject H_0, and the α spending at each interim stage is zero. For the power family error spending function with the default parameter $\rho = 2$, the beta spending at stage 1 is $(1/3)^\rho \beta = (1/3)^2 0.09 = 0.01$, and the cumulative beta spending at stage 2 is $(2/3)^\rho \beta = (2/3)^2 0.09 = 0.04$.

With ODS Graphics enabled, a detailed boundary plot with the acceptance and rejection regions is displayed, as shown in Output 110.10.4.

Output 110.10.4 Boundary Plot

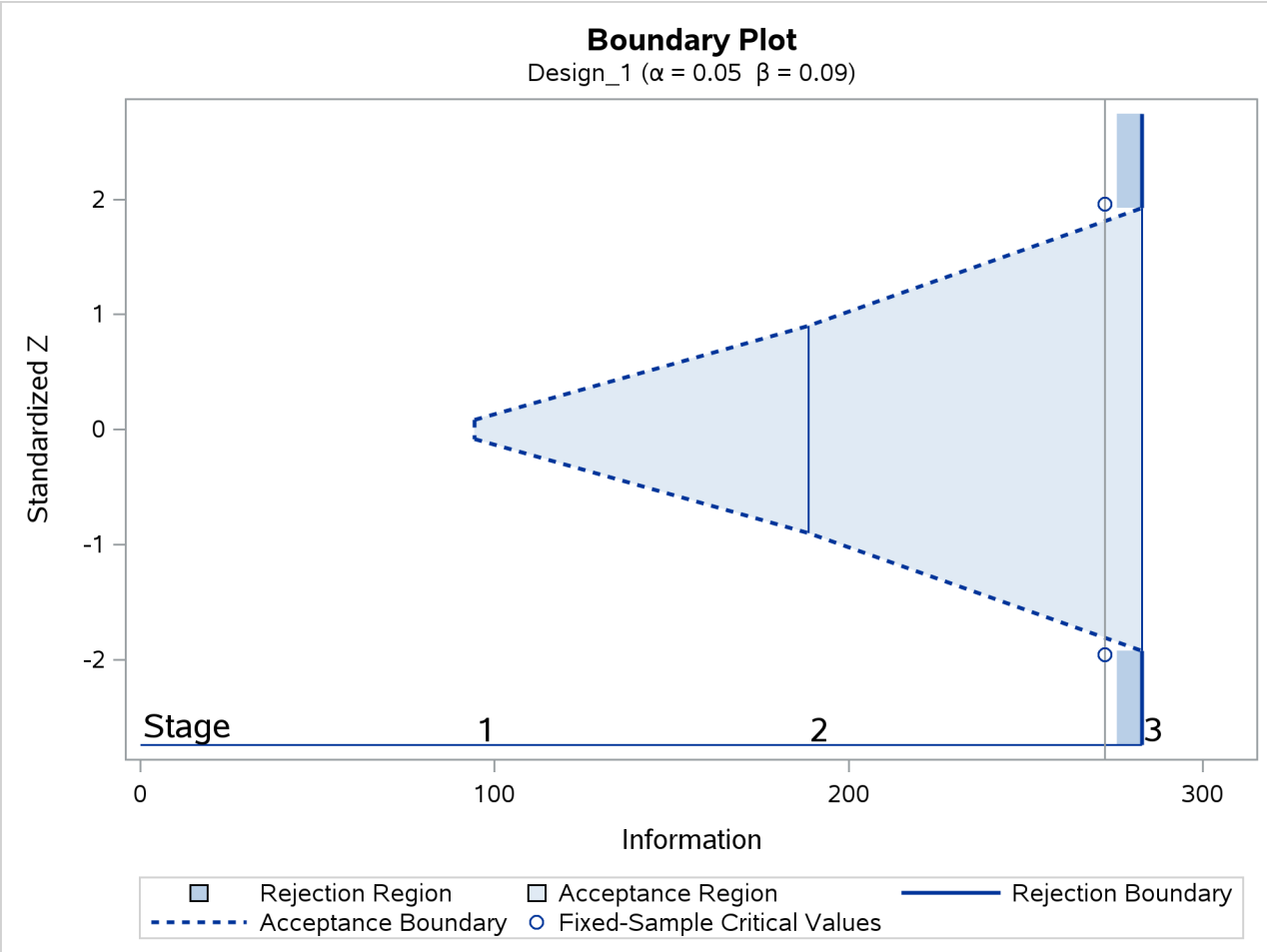

The following statements request a two-sided design with the BETAOVERLAP=ADJUST option, which is the default:

```
ods graphics on;
proc seqdesign altref=0.2 errspend;
   design nstages=3
          method=errfuncpow
          alt=twosided
          stop=accept
          betaoverlap=adjust
          beta=0.09
          ;
run;
```

With the BETAOVERLAP=ADJUST option, the procedure first derives the usual β boundary values for the two-sided design and then checks for overlapping of the β boundaries for the two corresponding one-sided tests at each stage. If this type of overlapping occurs at a particular stage, the β boundary values for that stage are set to missing, the β spending values at that stage are reset to zero, and the β spending values at subsequent stages are adjusted proportionally.

The boundary values without adjusting for the possible overlapping of the two one-sided β boundaries are

identical to the boundary values derived in the first design (with the BETAOVERLAP=NOADJUST option, as shown in Output 110.10.2). At stage 1, the upper β boundary value for the corresponding one-sided test is

$$\theta_1\sqrt{I_1} - \Phi^{-1}(1-\beta_1) = 0.2\sqrt{94.31094} - \Phi^{-1}(0.99) = 1.94228 - 2.32635 = -0.38407$$

where $\theta_1 = 0.2$ is the upper alternative reference, $I_1 = 94.31094$ is the information level at stage 1, and $\beta_1 = 0.01$ is the β spending at stage 1 (as shown in Output 110.10.3).

Similarly, the lower β boundary value for the corresponding one-sided test is computed as 0.38407. Since the upper β boundary value is less than the lower β boundary at stage 1, overlapping occurs, and so the β boundary values for the two-sided design are set to missing at stage 1.

With the β boundary values set to missing at stage 1 and the β spending $\beta'_1 = 0$ the β spending values at subsequent interim stages are adjusted proportionally. In this example, the adjusted β spending at stage 2 is computed as

$$\beta'_2 = \beta'_1 + \frac{\beta_2 - \beta_1}{\beta_3 - \beta_1}(\beta_3 - \beta'_1) = 0 + \frac{0.04 - 0.01}{0.09 - 0.01}\,0.09 = 0.03375$$

where β_k is the cumulative β spending at stage k before the adjustment, $k = 1, 2, 3$.

The "Design Information" table in Output 110.10.5 displays design specifications and derived statistics for the design.

Output 110.10.5 Design Information

The SEQDESIGN Procedure
Design: Design_1

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Accept Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.2
Number of Stages	3
Alpha	0.05
Beta	0.09
Power	0.91
Max Information (Percent of Fixed Sample)	101.9388
Max Information	277.649
Null Ref ASN (Percent of Fixed Sample)	80.56408
Alt Ref ASN (Percent of Fixed Sample)	100.792

The "Boundary Information" table in Output 110.10.6 displays the information levels, alternative references, and boundary values.

Output 110.10.6 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0						
	Information Level		Alternative Reference		Boundary Values	
					Lower	Upper
Stage	Proportion	Actual	Lower	Upper	Beta	Beta
1	0.3333	92.54967	-1.92405	1.92405	.	.
2	0.6667	185.0993	-2.72102	2.72102	-0.89469	0.89469
3	1.0000	277.649	-3.33256	3.33256	-1.93494	1.93494

The "Error Spending Information" table in Output 110.10.7 displays the cumulative error spending at each stage for each boundary.

Output 110.10.7 Error Spending Information

Error Spending Information					
		Cumulative Error Spending			
	Information Level	Lower		Upper	
Stage	Proportion	Alpha	Beta	Beta	Alpha
1	0.3333	0.00000	0.00000	0.00000	0.00000
2	0.6667	0.00000	0.03375	0.03375	0.00000
3	1.0000	0.02500	0.09000	0.09000	0.02500

With ODS Graphics enabled, a detailed boundary plot with the acceptance and rejection regions is displayed, as shown in Output 110.10.8.

Output 110.10.8 Boundary Plot

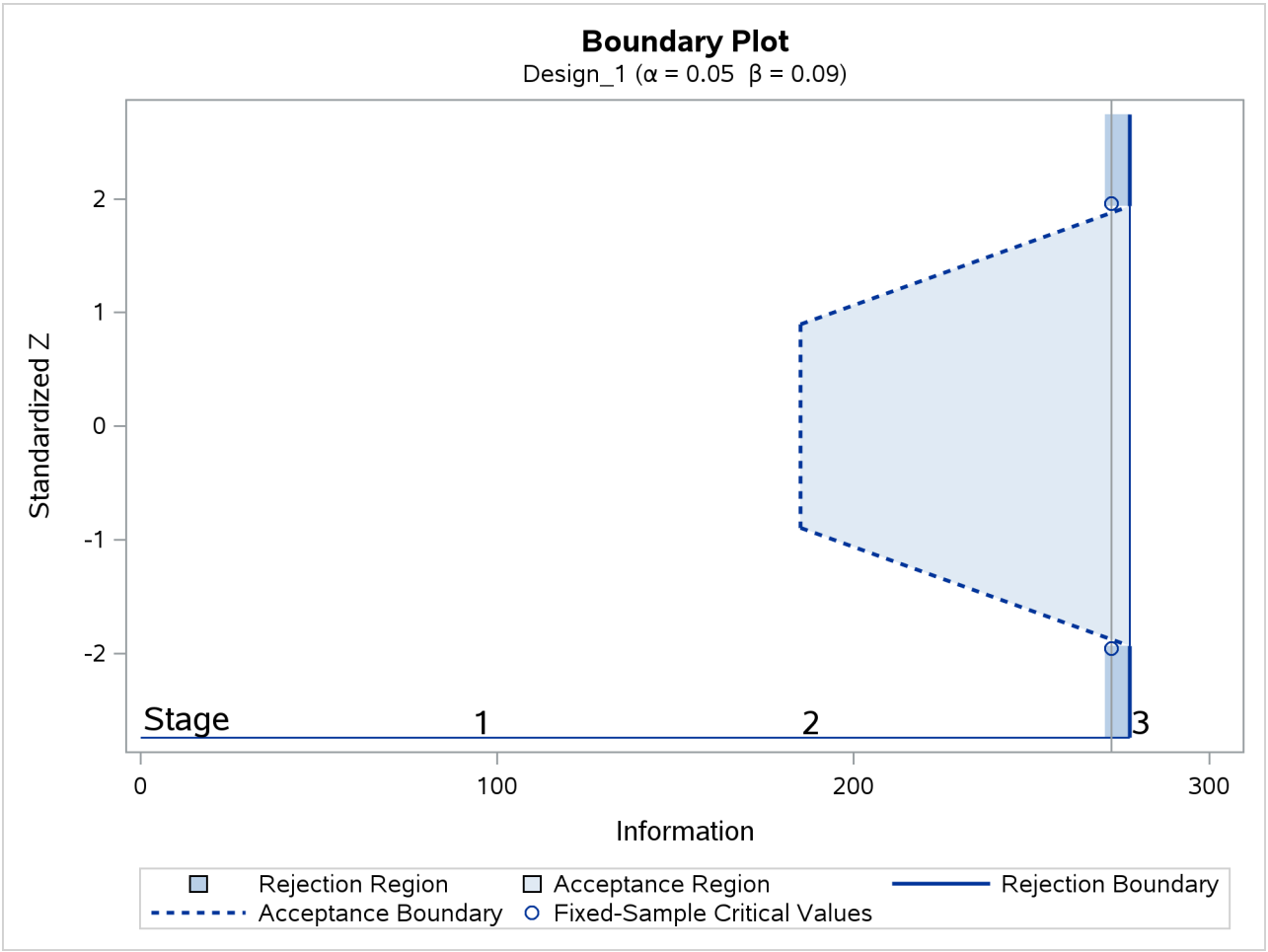

Example 110.11: Creating a Two-Sided Asymmetric Error Spending Design with Early Stopping to Reject H_0

(View the complete code for this example (seqdx11.sas) in the example repository.)

This example requests a three-stage two-sided asymmetric group sequential design for normally distributed statistics.

The O'Brien-Fleming boundary can be approximated using a power family error spending function with parameter $\rho = 3$, and the Pocock boundary can be approximated using a power family error spending function with parameter $\rho = 1$ (Jennison and Turnbull 2000, p. 148). The following statements use the power family error spending function to creates a two-sided asymmetric design with early stopping to reject the null hypothesis H_0:

```
ods graphics on;
proc seqdesign altref=1.0
               pss(cref=0 0.5 1)
               stopprob(cref=0 0.5 1)
```

```
                errspend
                plots=(asn power errspend)
                ;

   TwoSidedErrorSpending: design nstages=3
                          method(upperalpha)=errfuncpow(rho=3)
                          method(loweralpha)=errfuncpow(rho=1)
                          info=cum(2 3 4)
                          alt=twosided
                          stop=reject
                          alpha=0.075(upper=0.025)
                          ;
run;
```

The design uses power family error spending functions with $\rho = 1$ for the lower α boundary and $\rho = 3$ for the upper α boundary. Thus, the design is conservative in the early stages and tends to stop the trials early only with a small p-value for the upper α boundary. The upper α level 0.025 is specified explicitly, and the lower α level is computed as 0.075 – 0.025 = 0.05.

The "Design Information" table in Output 110.11.1 displays design specifications and the derived maximum information. Note that in order to attain the same information level for the asymmetric lower and upper boundaries, the derived power at the lower alternative 0.92963 is larger than the default 0.90.

Output 110.11.1 Design Information

The SEQDESIGN Procedure
Design: TwoSidedErrorSpending

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	1
Number of Stages	3
Alpha	0.075
Alpha (Lower)	0.05
Alpha (Upper)	0.025
Beta (Lower)	0.07037
Beta (Upper)	0.1
Power (Lower)	0.92963
Power (Upper)	0.9
Max Information (Percent of Fixed Sample)	102.4384
Max Information	10.76365
Null Ref ASN (Percent of Fixed Sample)	100.4877
Lower Alt Ref ASN (Percent of Fixed Sample)	64.8288
Upper Alt Ref ASN (Percent of Fixed Sample)	75.98778

The "Method Information" table in Output 110.11.2 displays the specified α and β error levels and the derived drift parameter. With the same information level used for the asymmetric lower and upper boundaries,

only one of the β levels is maintained, and the other is derived to have the level less than or equal to the default level.

Output 110.11.2 Method Information

Method Information						
				Error Spending		
Boundary	Method	Alpha	Beta	Function	Alternative Reference	Drift
Upper Alpha	Error Spending	0.02500	0.10000	Power (Rho=3)	1	3.280801
Lower Alpha	Error Spending	0.05000	0.07037	Power (Rho=1)	-1	-3.2808

With the STOPPROB(CREF=0 0.5 1) option, the "Expected Cumulative Stopping Probabilities" table in Output 110.11.3 displays the expected stopping stage and cumulative stopping probability to reject the null hypothesis H_0 at each stage under hypothetical references $\theta = 0$ (null hypothesis H_0), $\theta = 0.5\,\theta_1$, and $\theta = \theta_1$ (alternative hypothesis H_1), where θ_1 is the alternative reference.

Output 110.11.3 Stopping Probabilities

Expected Cumulative Stopping Probabilities Reference = CRef * (Alt Reference)						
				Stopping Probabilities		
CRef	Ref	Expected Stopping Stage	Source	Stage_1	Stage_2	Stage_3
0.0000	Lower Alt	2.924	Rej Null (Lower Alt)	0.02500	0.03750	0.05000
0.0000	Lower Alt	2.924	Rej Null (Upper Alt)	0.00313	0.01055	0.02500
0.0000	Lower Alt	2.924	Reject Null	0.02813	0.04805	0.07500
0.5000	Lower Alt	2.456	Rej Null (Lower Alt)	0.21185	0.33190	0.45370
0.5000	Lower Alt	2.456	Rej Null (Upper Alt)	0.00005	0.00012	0.00021
0.5000	Lower Alt	2.456	Reject Null	0.21190	0.33202	0.45391
1.0000	Lower Alt	1.531	Rej Null (Lower Alt)	0.64054	0.82803	0.92963
1.0000	Lower Alt	1.531	Rej Null (Upper Alt)	0.00000	0.00000	0.00000
1.0000	Lower Alt	1.531	Reject Null	0.64054	0.82803	0.92963
0.0000	Upper Alt	2.924	Rej Null (Lower Alt)	0.02500	0.03750	0.05000
0.0000	Upper Alt	2.924	Rej Null (Upper Alt)	0.00313	0.01055	0.02500
0.0000	Upper Alt	2.924	Reject Null	0.02813	0.04805	0.07500
0.5000	Upper Alt	2.758	Rej Null (Lower Alt)	0.00090	0.00110	0.00120
0.5000	Upper Alt	2.758	Rej Null (Upper Alt)	0.05769	0.18269	0.36458
0.5000	Upper Alt	2.758	Reject Null	0.05860	0.18379	0.36578
1.0000	Upper Alt	1.967	Rej Null (Lower Alt)	0.00001	0.00001	0.00001
1.0000	Upper Alt	1.967	Rej Null (Upper Alt)	0.33926	0.69356	0.90000
1.0000	Upper Alt	1.967	Reject Null	0.33927	0.69357	0.90001

"Rej Null (Lower Alt)" and "Rej Null (Upper Alt)" under the heading "Source" indicate the probabilities of rejecting the null hypothesis for the lower alternative and for the upper alternative, respectively. "Reject Null" indicates the probability of rejecting the null hypothesis for either the lower or upper alternative.

Note that with the STOP=REJECT option, the cumulative stopping probability of accepting the null hypothesis H_0 at each interim stage is zero and is not displayed.

With the PSS(CREF=0 0.5 1.0) option, the "Power and Expected Sample Sizes" table in Output 110.11.4

displays powers and expected sample sizes under hypothetical references $\theta = 0$ (null hypothesis H_0), $\theta = 0.5\,\theta_1$, and $\theta = \theta_1$ (alternative hypothesis H_1), where θ_1 is the alternative reference. The expected sample sizes are displayed in a percentage scale relative to the corresponding fixed-sample size design.

Output 110.11.4 Power and Expected Sample Size Information

Powers and Expected Sample Sizes Reference = CRef * (Alt Reference)			
			Sample Size
CRef	Ref	Power	Percent Fixed-Sample
0.0000	Lower Alt	0.05000	100.4877
0.5000	Lower Alt	0.45370	88.5090
1.0000	Lower Alt	0.92963	64.8288
0.0000	Upper Alt	0.02500	100.4877
0.5000	Upper Alt	0.36458	96.2309
1.0000	Upper Alt	0.90000	75.9878

Note that at $c_i = 0$, the null reference $\theta = 0$, the power with the lower alternative is the lower α error 0.05, and the power with the upper alternative is the upper α error 0.025. At $c_i = 1$, the alternative reference $\theta = \theta_1$, the power with the upper alternative is the specified power 0.90, and the power with the lower alternative 0.92963 is greater than the specified power 0.90 because the same information level is used for these two asymmetric boundaries.

With the PLOTS=POWER option, the procedure displays a plot of the power curves under various hypothetical references, as shown in Output 110.11.5. By default, powers under the lower hypotheses $\theta = c_i\,\theta_{1l}$ and under the upper hypotheses $\theta = c_i\,\theta_{1u}$ are displayed for a two-sided asymmetric design, where $c_i = 0, 0.01, 0.02, \ldots, 1.50$ and $\theta_{1l} = -1$ and $\theta_{1u} = 1$ are the lower and upper alternative references, respectively.

Output 110.11.5 Power Plot

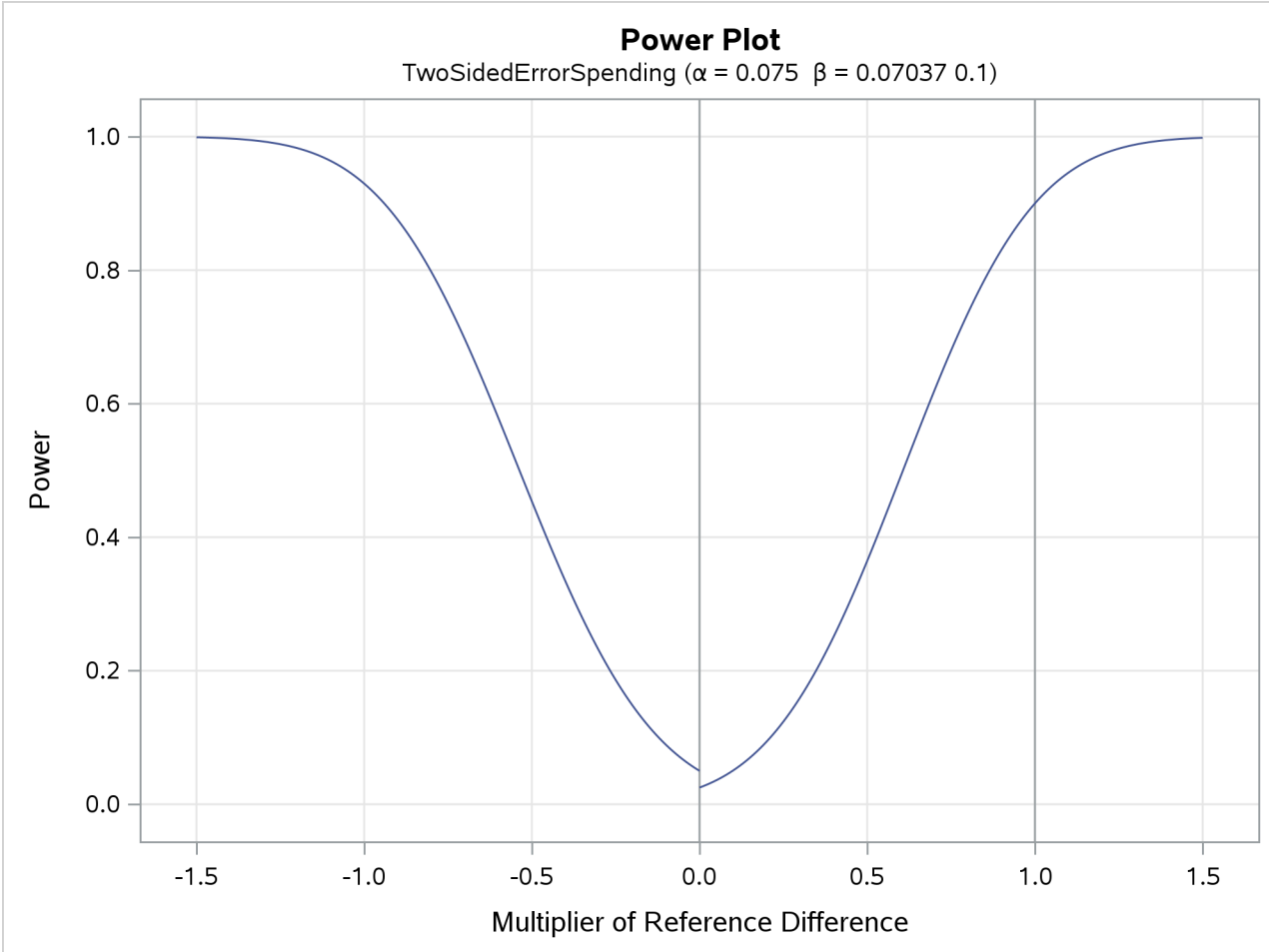

The horizontal axis displays the multiplier of the reference difference. A positive multiplier corresponds to c_i for the upper alternative hypothesis, and a negative multiplier corresponds to $-c_i$ for the lower alternative hypothesis. For lower reference hypotheses, the power is the lower α error 0.05 under the null hypothesis ($c_i = 0$) and is 0.92963 under the alternative hypothesis ($c_i = 1$). For upper reference hypotheses, the power is the upper α error 0.025 under the null hypothesis ($c_i = 0$) and is 0.90 under the alternative hypothesis ($c_i = 1$).

With the PLOTS=ASN option, the procedure displays a plot of expected sample sizes under various hypothetical references, as shown in Output 110.11.6. By default, expected sample sizes under the lower hypotheses $\theta = c_i\,\theta_{1l}$ and under the upper hypotheses $\theta = c_i\,\theta_{1u}$, $c_i = 0, 0.01, 0.02, \ldots, 1.50$, are displayed for a two-sided asymmetric design, where $\theta_{1l} = -1$ and $\theta_{1u} = 1$ are the lower and upper alternative references, respectively.

Output 110.11.6 ASN Plot

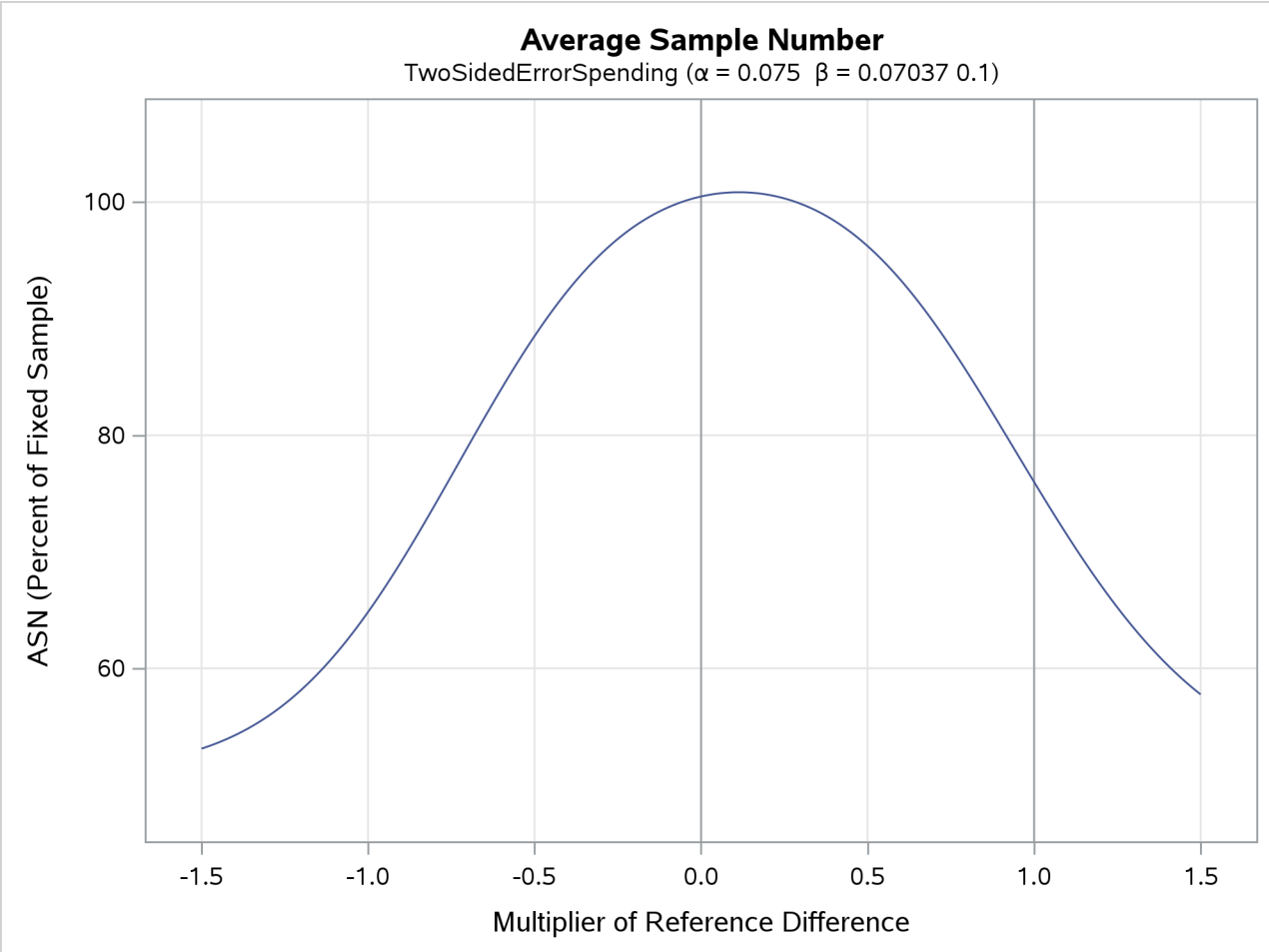

The horizontal axis displays the multiplier of the reference difference. A positive multiplier corresponds to c_i for the upper alternative hypothesis and a negative multiplier corresponds to $-c_i$ for the lower alternative hypothesis.

The "Boundary Information" table in Output 110.11.7 displays the information levels, alternative references, and boundary values. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the standardized Z scale is used to display the alternative references and boundary values. The resulting standardized alternative references at stage k are given by $\pm\theta_1\sqrt{I_k}$, where θ_1 is the specified alternative reference and I_k is the information level at stage k, $k = 1, 2, 3$.

Output 110.11.7 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0						
	Information Level		Alternative Reference		Boundary Values	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Upper Alpha
1	0.5000	5.381827	-2.31988	2.31988	-1.95996	2.73437
2	0.7500	8.07274	-2.84126	2.84126	-1.98394	2.35681
3	1.0000	10.76365	-3.28080	3.28080	-1.90855	2.02853

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.11.8.

Output 110.11.8 Boundary Plot

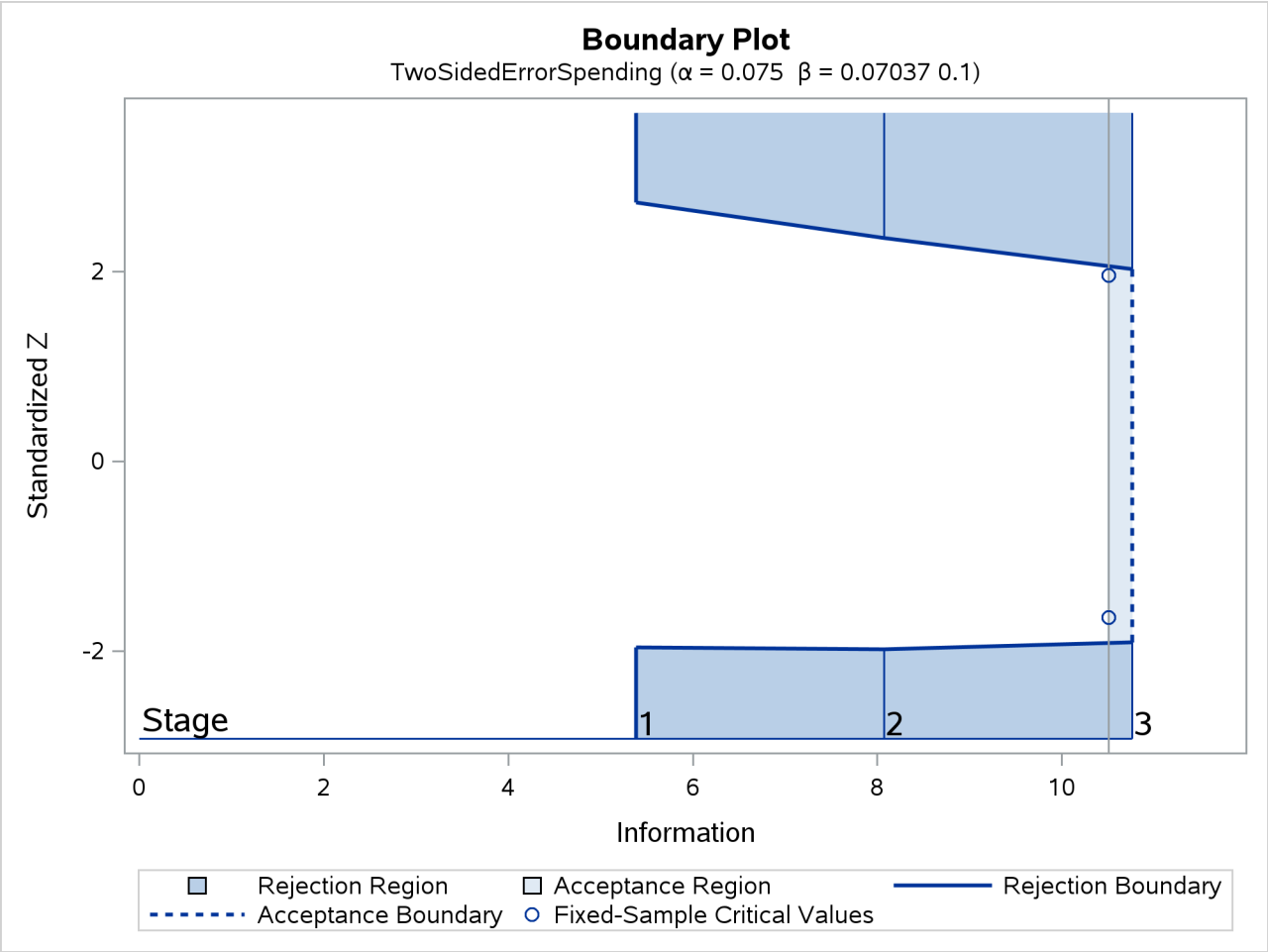

The "Error Spending Information" table in Output 110.11.9 displays the cumulative error spending at each stage for each boundary.

Output 110.11.9 Error Spending Information

Error Spending Information					
		Cumulative Error Spending			
	Information Level	Lower		Upper	
Stage	Proportion	Alpha	Beta	Beta	Alpha
1	0.5000	0.02500	0.00000	0.00001	0.00313
2	0.7500	0.03750	0.00000	0.00001	0.01055
3	1.0000	0.05000	0.07037	0.10000	0.02500

With the STOP=REJECT option, there is no early stopping to accept H_0, and the corresponding β spending at an interim stage is computed from the rejection region. For example, the upper β spending at stage 1 (0.00001) is the probability of rejecting H_0 for the lower alternative under the upper alternative reference.

With the PLOTS=ERRSPEND option, the procedure displays a plot of the cumulative error spending on each boundary at each stage, as shown in Output 110.11.10.

Output 110.11.10 Error Spending Plot

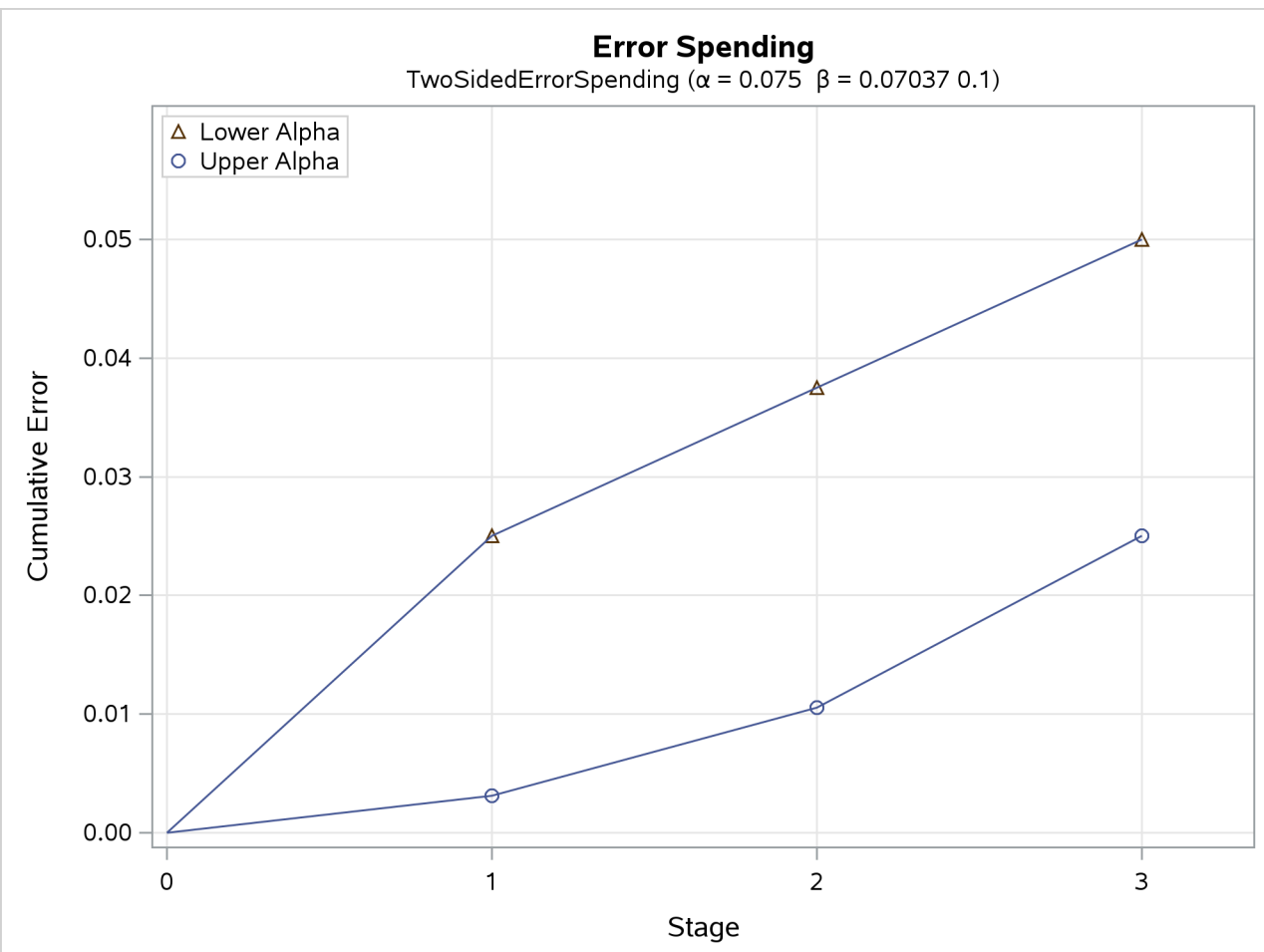

Example 110.12: Creating a Two-Sided Asymmetric Error Spending Design with Early Stopping to Reject or Accept H_0

(View the complete code for this example (seqdx12.sas) in the example repository.)

This example requests a four-stage two-sided asymmetric group sequential design for normally distributed statistics. The O'Brien-Fleming boundary can be approximated by a gamma family error spending function with parameter $\gamma = -4$ or –5, and the Pocock boundary can be approximated with parameter $\gamma = 1$ (Hwang, Shih, and DeCani 1990, p. 1440). The following statements use the gamma error spending function with early stopping to reject or accept the null hypothesis H_0:

```
ods graphics on;
proc seqdesign altref=2
               pss(cref=0 0.5 1)
               stopprob(cref=0 1)
               errspend
               plots=(asn power errspend)
               ;
TwoSidedAsymmetric: design nstages=4
                    method=errfuncgamma(gamma=1)
                    method(beta)=errfuncgamma(gamma=-2)
                    method(upperalpha)=errfuncgamma(gamma=-5)
                    alt=twosided
                    stop=both
                    beta=0.1
                    ;
run;
```

The design uses gamma family error spending functions with $\gamma = -5$ for the upper α boundary, $\gamma = 1$ for the lower α boundary, and $\gamma = -2$ for the lower and upper β boundaries.

The "Design Information" table in Output 110.12.1 displays design specifications and the derived maximum information. Note that in order to attain the same information level for the asymmetric lower and upper boundaries, the derived power at the upper alternative 0.93655 is larger than the specified $1 - \beta = 0.90$.

Output 110.12.1 Design Information

The SEQDESIGN Procedure
Design: TwoSidedAsymmetric

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Accept/Reject Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	2
Number of Stages	4
Alpha	0.05
Beta (Lower)	0.1
Beta (Upper)	0.06345
Power (Lower)	0.9
Power (Upper)	0.93655
Max Information (Percent of Fixed Sample)	104.0688
Max Information	3.162386
Null Ref ASN (Percent of Fixed Sample)	74.16654
Lower Alt Ref ASN (Percent of Fixed Sample)	59.10271
Upper Alt Ref ASN (Percent of Fixed Sample)	73.78797

The "Method Information" table in Output 110.11.2 displays the specified α and β error levels and the derived drift parameter. With the same information level used for the asymmetric lower and upper boundaries, only one of the β levels is maintained and the other is derived to have the level less than or equal to the specified level.

Output 110.12.2 Method Information

Method Information						
				Error Spending		
Boundary	**Method**	**Alpha**	**Beta**	**Function**	**Alternative Reference**	**Drift**
Upper Alpha	Error Spending	0.02500	.	Gamma (Gamma=-5)	2	3.55662
Upper Beta	Error Spending	.	0.06345	Gamma (Gamma=-2)	2	3.55662
Lower Beta	Error Spending	.	0.10000	Gamma (Gamma=-2)	-2	-3.55662
Lower Alpha	Error Spending	0.02500	.	Gamma (Gamma=1)	-2	-3.55662

With the STOPPROB(CREF=0 1) option, the "Expected Cumulative Stopping Probabilities" table in Output 110.12.3 displays the expected stopping stage and cumulative stopping probabilities at each stage under the null reference $\theta = 0$ and under the alternative reference $\theta = \theta_1$.

Output 110.12.3 Stopping Probabilities

Expected Cumulative Stopping Probabilities Reference = CRef * (Alt Reference)							
				Stopping Probabilities			
CRef	Ref	Expected Stopping Stage	Source	Stage_1	Stage_2	Stage_3	Stage_4
0.0000	Lower Alt	2.851	Rej Null (Lower Alt)	0.00875	0.01556	0.02087	0.02500
0.0000	Lower Alt	2.851	Rej Null (Upper Alt)	0.00042	0.00190	0.00704	0.02500
0.0000	Lower Alt	2.851	Reject Null	0.00917	0.01746	0.02791	0.05000
0.0000	Lower Alt	2.851	Accept Null	0.00000	0.30125	0.79354	0.95000
0.0000	Lower Alt	2.851	Total	0.00917	0.31870	0.82145	1.00000
1.0000	Lower Alt	2.272	Rej Null (Lower Alt)	0.27499	0.58934	0.79601	0.90000
1.0000	Lower Alt	2.272	Rej Null (Upper Alt)	0.00000	0.00000	0.00000	0.00000
1.0000	Lower Alt	2.272	Reject Null	0.27499	0.58934	0.79601	0.90000
1.0000	Lower Alt	2.272	Accept Null	0.00000	0.01863	0.04935	0.10000
1.0000	Lower Alt	2.272	Total	0.27499	0.60797	0.84536	1.00000
0.0000	Upper Alt	2.851	Rej Null (Lower Alt)	0.00875	0.01556	0.02087	0.02500
0.0000	Upper Alt	2.851	Rej Null (Upper Alt)	0.00042	0.00190	0.00704	0.02500
0.0000	Upper Alt	2.851	Reject Null	0.00917	0.01746	0.02791	0.05000
0.0000	Upper Alt	2.851	Accept Null	0.00000	0.30125	0.79354	0.95000
0.0000	Upper Alt	2.851	Total	0.00917	0.31870	0.82145	1.00000
1.0000	Upper Alt	2.836	Rej Null (Lower Alt)	0.00002	0.00002	0.00002	0.00002
1.0000	Upper Alt	2.836	Rej Null (Upper Alt)	0.05945	0.33802	0.72323	0.93655
1.0000	Upper Alt	2.836	Reject Null	0.05947	0.33804	0.72325	0.93657
1.0000	Upper Alt	2.836	Accept Null	0.00000	0.01182	0.03131	0.06343
1.0000	Upper Alt	2.836	Total	0.05947	0.34986	0.75456	1.00000

"Rej Null (Lower Alt)" and "Rej Null (Upper Alt)" under the heading "Source" indicate the probabilities of rejecting the null hypothesis for the lower alternative and for the upper alternative, respectively. "Reject Null" indicates the probability of rejecting the null hypothesis for either the lower or upper alternative, "Accept Null" indicates the probability of accepting the null hypothesis, and "Total" indicates the total probability of stopping the trial.

With the PSS(CREF=0 0.5 1.0) option, the "Power and Expected Sample Sizes" table in Output 110.12.4 displays powers and expected sample sizes under hypothetical references $\theta = 0$ (null hypothesis H_0), $\theta = 0.5\,\theta_1$, and $\theta = \theta_1$ (alternative hypothesis H_1), where θ_1 is the alternative reference. The expected sample sizes are displayed in a scale that indicates a percentage of its corresponding fixed-sample size design.

Output 110.12.4 Power and Expected Sample Size Information

Powers and Expected Sample Sizes Reference = CRef * (Alt Reference)			
			Sample Size
CRef	Ref	Power	Percent Fixed-Sample
0.0000	Lower Alt	0.02500	74.1665
0.5000	Lower Alt	0.34601	75.8425
1.0000	Lower Alt	0.90000	59.1027
0.0000	Upper Alt	0.02500	74.1665
0.5000	Upper Alt	0.41647	85.3976
1.0000	Upper Alt	0.93655	73.7880

Note that at $c_i = 0$, the null reference $\theta = 0$, the power with the lower alternative is the lower α error 0.025, and the power with the upper alternative is the upper α error 0.025. At $c_i = 1$, the alternative reference $\theta = \theta_1$, the power with the lower alternative is the specified power 0.90, and the power with the upper alternative 0.93655 is greater than the specified power 0.90 because the same information level is used for these two asymmetric boundaries.

With the PLOTS=POWER option, the procedure displays a plot of the power curves under various hypothetical references, as shown in Output 110.12.5. By default, powers under the lower hypotheses $\theta = c_i\,\theta_{1l}$ and under the upper hypotheses $\theta = c_i\,\theta_{1u}$, are displayed for a two-sided asymmetric design, where $c_i = 0, 0.01, 0.02, \ldots, 1.50$ and $\theta_{1l} = -1$ and $\theta_{1u} = 1$ are the lower and upper alternative references, respectively.

Output 110.12.5 Power Plot

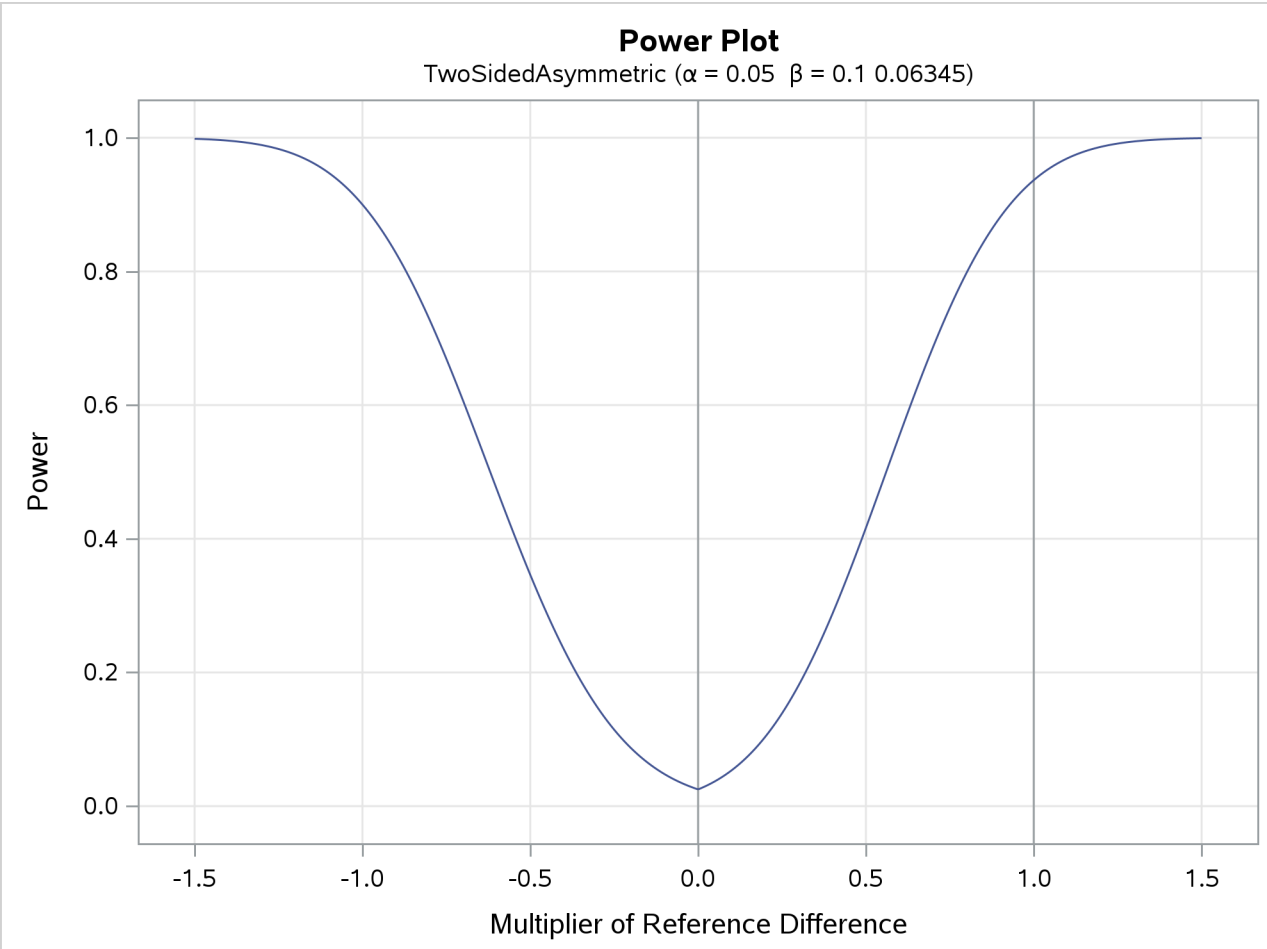

The horizontal axis displays the multiplier of the reference difference. A positive multiplier corresponds to c_i for the upper alternative hypothesis, and a negative multiplier corresponds to $-c_i$ for the lower alternative hypothesis. For lower reference hypotheses, the power is the lower α error 0.025 under the null hypothesis ($c_i = 0$) and is 0.90 under the alternative hypothesis ($c_i = 1$). For upper reference hypotheses, the power is the upper α error 0.025 under the null hypothesis ($c_i = 0$) and is 0.93655 under the alternative hypothesis ($c_i = 1$).

With the PLOTS=ASN option, the procedure displays a plot of expected sample sizes under various hypothet-

ical references, as shown in Output 110.12.6. By default, expected sample sizes under the lower hypotheses $\theta = c_i\,\theta_{1l}$ and under the upper hypotheses $\theta = c_i\,\theta_{1u}$ are displayed for a two-sided asymmetric design, where $c_i = 0, 0.01, 0.02, \ldots, 1.50$ and $\theta_{1l} = -1$ and $\theta_{1u} = 1$ are the lower and upper alternative references, respectively.

Output 110.12.6 ASN Plot

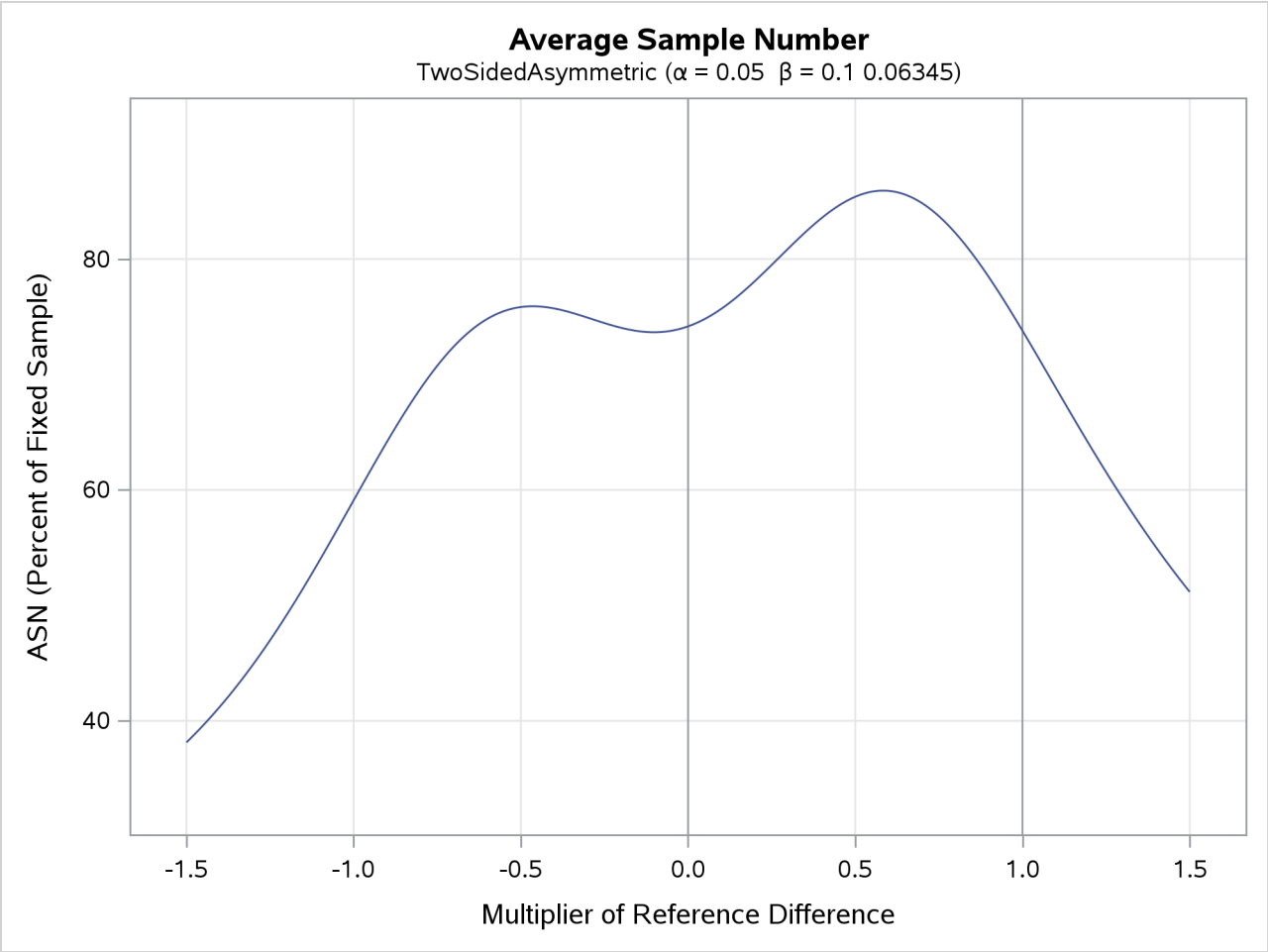

The horizontal axis displays the multiplier of the reference difference. A positive multiplier corresponds to c_i for the upper alternative hypothesis, and a negative multiplier corresponds to $-c_i$ for the lower alternative hypothesis.

By default (or equivalently if you specify BETAOVERLAP=ADJUST), the SEQDESIGN procedure first derives boundary values without adjusting for the possible overlapping of the two one-sided β boundaries based on two corresponding one-sided tests. Then the procedure checks for overlapping of the β boundaries at the interim stages. Since the two β boundaries overlap at stage 1, the β boundary values for stage 1 are set to missing, the β spending values at stage 1 are set to zero, and the β spending values at subsequent stages are adjusted proportionally.

The "Boundary Information" table in Output 110.12.7 displays the information levels, alternative references, and boundary values. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the standardized Z scale is used to display the alternative references and boundary values. The resulting standardized

alternative references at stage k is given by $\pm\theta_1\sqrt{I_k}$, where θ_1 is the specified alternative reference and I_k is the information level at stage k, $k = 1, 2, 3, 4$.

Output 110.12.7 Boundary Information

Boundary Information (Standardized Z Scale) Null Reference = 0								
	Information Level		Alternative Reference		Boundary Values			
					Lower		Upper	
Stage	Proportion	Actual	Lower	Upper	Alpha	Beta	Beta	Alpha
1	0.2500	0.790597	-1.77831	1.77831	-2.37610	.	.	3.33772
2	0.5000	1.581193	-2.51491	2.51491	-2.35714	-0.48408	0.29400	2.94871
3	0.7500	2.37179	-3.08012	3.08012	-2.34861	-1.36183	1.13898	2.50473
4	1.0000	3.162386	-3.55662	3.55662	-2.32105	-2.32105	1.95675	1.95675

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.12.8.

Output 110.12.8 Boundary Plot

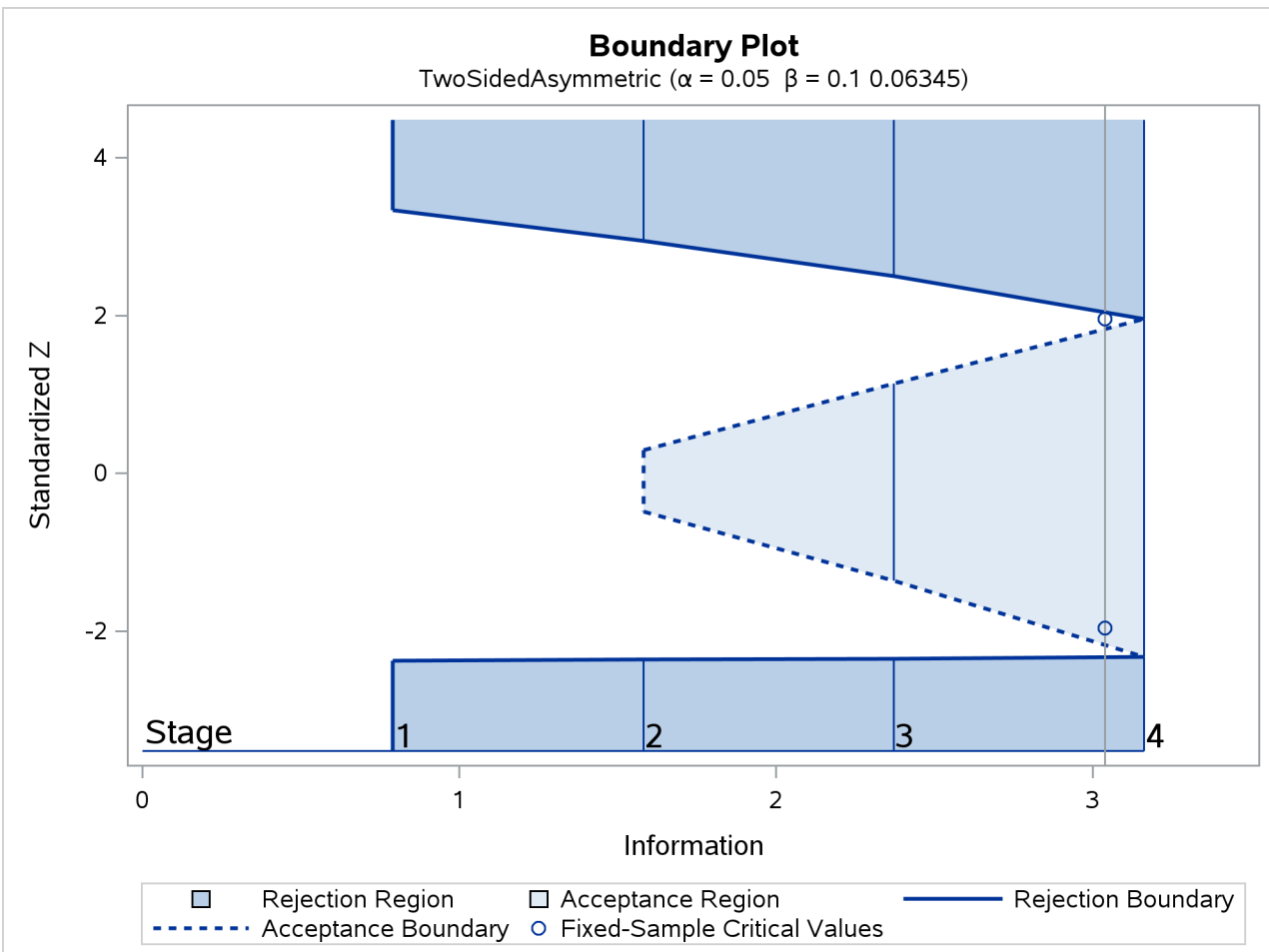

The "Error Spending Information" in Output 110.12.9 displays the cumulative error spending at each stage

for each boundary.

Output 110.12.9 Error Spending Information

Error Spending Information					
		Cumulative Error Spending			
	Information Level	Lower		Upper	
Stage	Proportion	Alpha	Beta	Beta	Alpha
1	0.2500	0.00875	0.00000	0.00002	0.00042
2	0.5000	0.01556	0.01863	0.01184	0.00190
3	0.7500	0.02087	0.04935	0.03132	0.00704
4	1.0000	0.02500	0.10000	0.06345	0.02500

With the β boundary values missing at stage 1, there is no early stopping to accept H_0 at stage 1, and the corresponding β spending at stage 1 is computed from the rejection region. For example, the upper β spending at stage 1 (0.00002) is the probability of rejecting H_0 for the lower alternative under the upper alternative reference.

With the PLOTS=ERRSPEND option, the procedure displays a plot of the cumulative error spending on each boundary at each stage, as shown in Output 110.12.10.

Output 110.12.10 Error Spending Plot

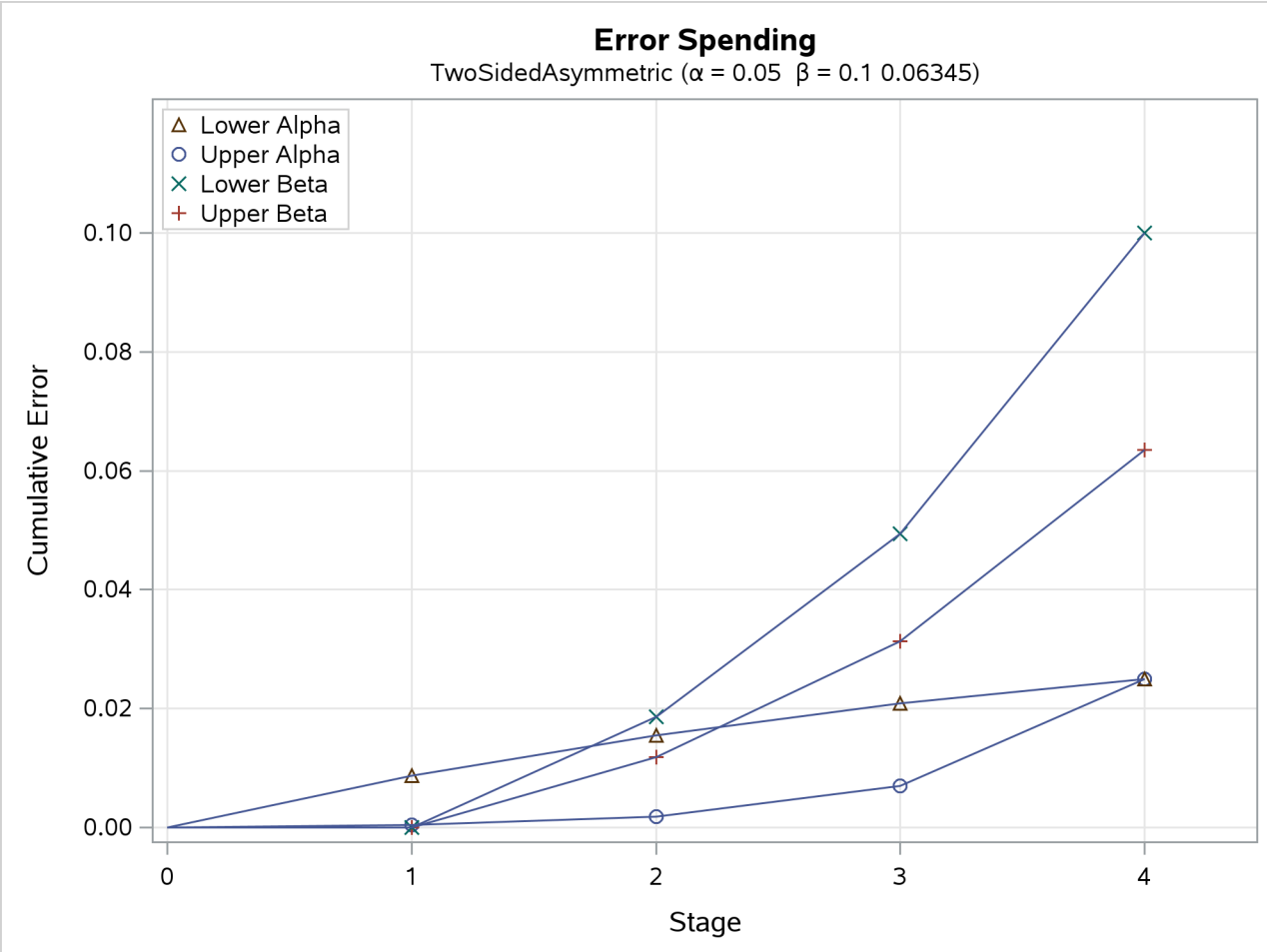

Example 110.13: Creating a Design with a Nonbinding Beta Boundary

(View the complete code for this example (seqdx13.sas) in the example repository.)

This example requests a two-sided design similar to Example 110.2, but with a nonbinding beta boundary. With a nonbinding acceptance boundary, the trial does not need to stop to accept the null hypothesis when a test statistic falls in the acceptance region at interim stages. The design maintains the specified Type I error level, but at the cost of increased sample size.

* The following statements invoke the SEQDESIGN procedure and request a four-stage group sequential design with early stopping to reject or accept H_0:

```
ods graphics on;
proc seqdesign altref=0.15 errspend
               ;
   NonbindingDesign: design nstages=4
                     method=obf
                     alt=upper
                     stop=both(betaboundary=nonbinding)
```

```
                        alpha=0.025 beta=0.10
                        ;
    samplesize model=twosamplefreq(nullprop=0.6 test=prop);
  run;
```

The STOP=BOTH option specifies early stopping to reject or accept the null hypothesis, and the BETABOUNDARY=NONBINDING suboption requests the nonbinding beta boundary.

The "Design Information," "Method Information," and "Boundary Information" tables are displayed by default. The "Design Information" table in Output 110.13.1 displays design specifications and derived statistics such as power and maximum information. With a specified alternative reference, ALTREF=0.15, the maximum information I_X is derived.

Output 110.13.1 Design Information

The SEQDESIGN Procedure
Design: NonbindingDesign

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Accept(Nonbinding)/Reject Null
Method	O'Brien-Fleming
Boundary Key	Both
Alternative Reference	0.15
Number of Stages	4
Alpha (Binding Beta Boundary)	0.02228
Alpha (Nonbinding Beta Boundary)	0.025
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	110.7138
Max Information	517.0296
Null Ref ASN (Percent of Fixed Sample)	62.29796
Alt Ref ASN (Percent of Fixed Sample)	78.5392

With the BETABOUNDARY=NONBINDING option, there are two ways to compute the Type I error level: with or without the beta boundary. Alpha (Binding Beta Boundary) displays the Type I error level with the beta boundary, and Alpha (Nonbinding Beta Boundary) displays the Type I error level without the beta boundary.

The maximum information is the information level at the final stage of the group sequential trial. The Max Information (Percent Fixed-Sample) row displays the maximum information for the sequential design expressed as a percentage of the information for the corresponding fixed-sample design (which has a Type I error level 0.025). In Output 110.13.1, Max Information (Percent Fixed-Sample) is 110.71%, which means that the information needed for the group sequential trial is 10.71% more than the information needed for the corresponding fixed-sample design if the trial does not stop at any interim stage.

Compared to the corresponding design with a binding beta boundary, which has a Max Information (Percent Fixed-Sample) of 107.67% as displayed in Output 110.2.1, the nonbinding design requires a larger sample size if the trial proceeds to the final stage.

The "Method Information" table in Output 110.13.2 displays the values of $\rho = 0.5$ and $\tau = 0$, which are the parameters for the O'Brien-Fleming method. The table also displays the drift parameter 3.4107, which is greater than the drift parameter 3.3636 in the corresponding binding boundary design, as displayed in Output 110.2.2.

Output 110.13.2 Method Information

Method Information

				Unified Family				
Boundary	**Method**	**Alpha**	**Beta**	**Rho**	**Tau**	**C**	**Alternative Reference**	**Drift**
Upper Alpha	O'Brien-Fleming	0.02500	.	0.5	0	2.0243	0.15	3.410743
Upper Beta	O'Brien-Fleming	.	0.10000	0.5	0	1.38645	0.15	3.410743

The "Boundary Information" table in Output 110.13.3 displays information level, alternative reference, and boundary values at each stage. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the alternative reference and boundary values are displayed with the standardized Z statistic scale. The resulting standardized alternative reference at stage k is given by $\theta_1\sqrt{I_k}$, where θ_1 is the alternative reference and I_k is the information level at stage k, for k= 1, 2, 3, 4.

Output 110.13.3 Boundary Information

Boundary Information (Standardized Z Scale)
Nonbinding Beta Boundary, Null Reference = 0

	Information Level			Alternative Reference	Boundary Values	
Stage	**Proportion**	**Actual**	**N**	**Upper**	**Upper Beta**	**Upper Alpha**
1	0.2500	129.2574	110.5151	1.70537	-1.06752	4.04859
2	0.5000	258.5148	221.0302	2.41176	0.45103	2.86279
3	0.7500	387.7722	331.5452	2.95379	1.35286	2.33746
4	1.0000	517.0296	442.0603	3.41074	2.02430	2.02430

By default (or equivalently if you specify INFO=EQUAL), equally spaced information levels are used. With the derived maximum information, the actual information level at each stage is also displayed. With the SAMPLESIZE statement, the required sample size N is also displayed under the heading "Information Level."

Note that with the standardized Z statistic scale, the rejection boundary values are identical to the boundary values that are derived in the corresponding design with early stopping to reject H_0.

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 110.13.4.

Output 110.13.4 Boundary Plot

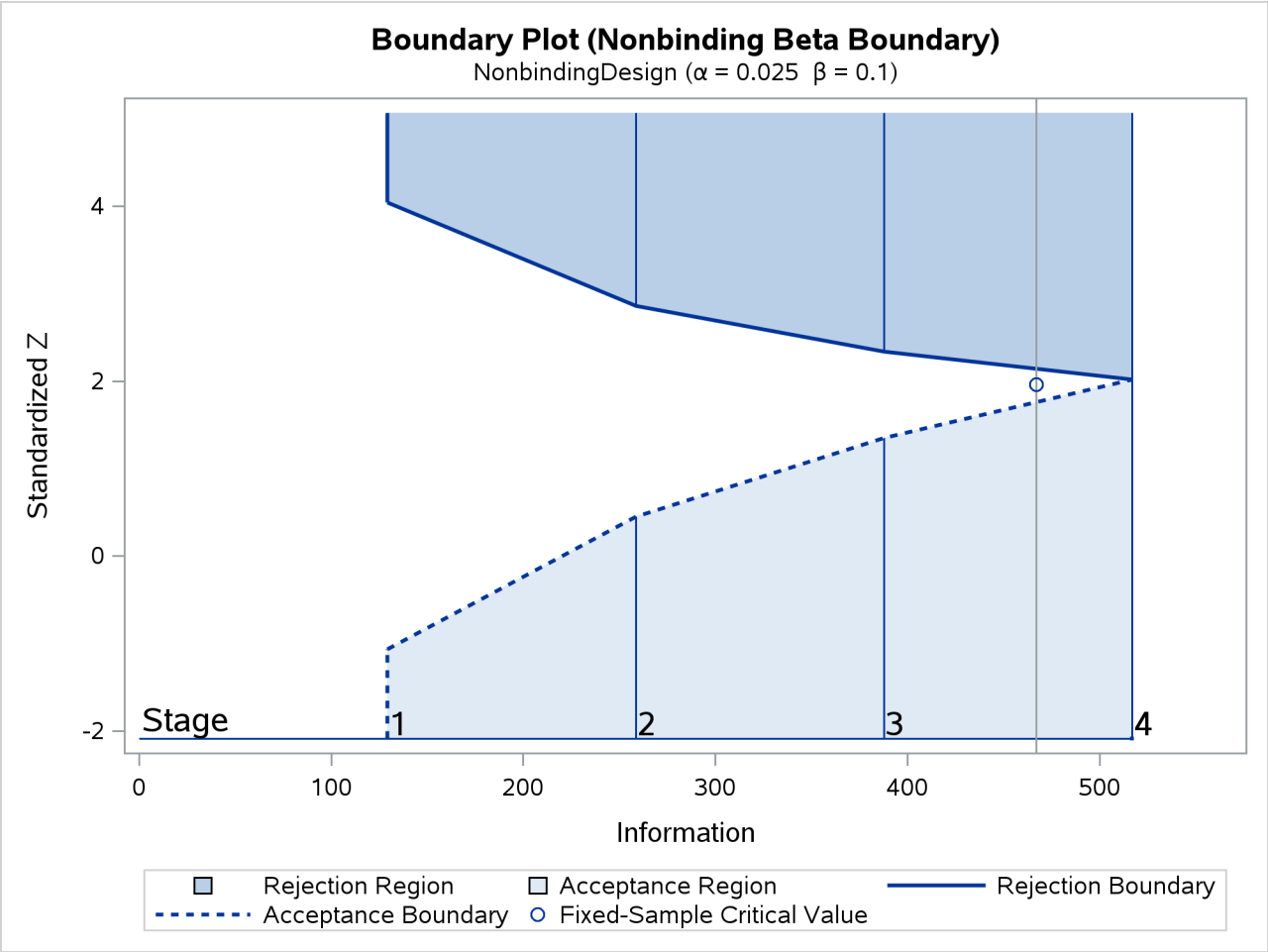

The horizontal axis indicates the information levels for the design. The stages are indicated by vertical lines with accompanying stage numbers. The boundary plot also displays the information level and the critical value for the corresponding fixed-sample design.

The "Error Spending Information (Nonbinding Beta Boundary)" table in Output 110.13.5 displays cumulative error spending at each stage for each boundary. With a nonbinding beta boundary, the α spending at each stage is computed by using the α boundary only.

Output 110.13.5 Error Spending Information

Error Spending Information (Nonbinding Beta Boundary)			
		Cumulative Error Spending	
	Information Level	Upper	
Stage	Proportion	Beta	Alpha
1	0.2500	0.00278	0.00003
2	0.5000	0.02603	0.00211
3	0.7500	0.06343	0.01046
4	1.0000	0.10000	0.02500

The SEQDESIGN procedure derives the drift parameter $\theta_1\sqrt{I_X}$, where θ_1 is the alternative reference and I_X is the maximum information. With the SAMPLESIZE statement, the maximum information is used to compute the required sample size for the study.

The "Sample Size Summary" table in Output 110.13.6 displays parameters for the sample size computation. With the MODEL=TWOSAMPLEFREQ(NULLPROP=0.6 TEST=PROP) option in the SAMPLESIZE statement, the total sample size and expected sample sizes under the null and alternative hypotheses for testing the difference between two proportions are displayed.

Output 110.13.6 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Proportions
Null Proportion	0.6
Proportion (Group A)	0.75
Test Statistic	Z for Proportion
Reference Proportions	Alt Ref
Max Sample Size	442.0603
Expected Sample Size (Null Ref)	248.7446
Expected Sample Size (Alt Ref)	313.5929

The "Sample Sizes (N)" table in Output 110.13.7 displays the required sample sizes at each stage, in both fractional and integer numbers. The derived fractional sample sizes are under the heading "Fractional N." These sample sizes are rounded up to integers under the heading "Ceiling N." As expected, the sample sizes in the table are larger than the corresponding sample sizes in the corresponding design with binding beta boundary.

Output 110.13.7 Derived Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Proportion Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	110.52	55.26	55.26	129.3	112	56	56	131.0
2	221.03	110.52	110.52	258.5	222	111	111	259.6
3	331.55	165.77	165.77	387.8	332	166	166	388.3
4	442.06	221.03	221.03	517.0	444	222	222	519.3

Example 110.14: Computing Sample Size for Survival Data That Have Uniform Accrual

(View the complete code for this example (seqdx14.sas) in the example repository.)

This example illustrates sample size computation for survival data when the accrual is uniform and when the data do not contain individual loss to follow-up.

Suppose that a clinic is conducting a study of the effect of a new cancer treatment. The study consists of exposing mice to a carcinogen and randomly assigning them to either the control group or the treatment group. The event of interest is death from cancer induced by the carcinogen, and the response is the time from randomization to death.

Following the derivations in the section "Test for Two Survival Distributions with a Log-Rank Test" on page 9201, the hypothesis $H_0 : \theta = -\log(\lambda) = 0$ with the alternative hypothesis $H_1 : \theta = \theta_1 > 0$ is used, where λ is the hazard ratio between the treatment group and the control group.

Also suppose that from past experience, the median survival time for the control group is $t_0 = 20$ weeks, and the study wants to detect a median survival time, $t_1 = 40$ weeks, with an 80% power in the trial. Assuming exponential survival functions for the two groups, the hazard rates can be computed from

$$S_j(t_j) = e^{-h_j t_j} = \frac{1}{2}$$

where $j = 0, 1$.

Thus, the hazard rates are $h_0 = 0.03466$ and $h_1 = 0.01733$ under the null and alternative hypotheses, respectively.

The following statements invoke the SEQDESIGN procedure and specify the SAMPLESIZE statement to derive required sample sizes for a log-rank test that compares two survival distributions for the treatment effect (Jennison and Turnbull 2000, pp. 77–79; Whitehead 1997, pp. 36–39):

```
proc seqdesign;
   ErrorSpend: design nstages=4 method=errfuncobf
               ;
   samplesize model=twosamplesurvival
                   ( nullhazard=0.03466 hazard=0.01733
                     accrual=uniform accrate=15);
run;
```

In the SAMPLESIZE statement, the MODEL=TWOSAMPLESURVIVAL option specifies a log-rank test to compare two survival distributions for the treatment effect. The NULLHAZARD=0.03466 option specifies null hazard rates for the two groups under the null hypothesis, and the HAZARD=0.01733 option specifies the hazard rate for the first group under the alternative hypothesis.

The ACCRUAL= option specifies the method for individual accrual. The ACCRUAL=UNIFORM option (which is the default) specifies that the individual accrual is uniform with a constant accrual rate, and the ACCRATE= option specifies the accrual rate.

You do not have to specify the alternative reference explicitly for the sample size computation in the SEQDESIGN procedure. For the specified null and alternative hazards, $h_0 = 0.03466$ and $h_1 = 0.01733$,

the hazard ratio $\lambda_1 = h_1/h_0 = 1/2$, and the alternative reference is

$$\theta_1 = -\log(\lambda_1) = -\log\left(\frac{1}{2}\right) = 0.6931$$

For a detailed derivation of these required sample sizes, see the section "Test for Two Survival Distributions with a Log-Rank Test" on page 9201.

The "Design Information" table in Output 110.14.1 displays design specifications and four derived statistics: the actual maximum information, the maximum information, the average sample number under the null hypothesis (Null Ref ASN), and the average sample number under the alternative hypothesis (Alt Ref ASN).

Output 110.14.1 Error Spending Design Information

The SEQDESIGN Procedure
Design: ErrorSpend

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.693147
Number of Stages	4
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	101.8279
Max Information	22.26962
Null Ref ASN (Percent of Fixed Sample)	101.2586
Alt Ref ASN (Percent of Fixed Sample)	77.73131

The "Boundary Information" table in Output 110.14.2 displays the information level, including the proportion, actual level, and corresponding number of events at each stage. The table also displays the lower and upper alternative references, and the lower and upper boundary values at each stage.

Output 110.14.2 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference		Boundary Values	
Stage	Proportion	Actual	Events	Lower	Upper	Lower Alpha	Upper Alpha
1	0.2500	5.567405	22.26962	-1.63550	1.63550	-4.33263	4.33263
2	0.5000	11.13481	44.53924	-2.31295	2.31295	-2.96333	2.96333
3	0.7500	16.70221	66.80886	-2.83278	2.83278	-2.35902	2.35902
4	1.0000	22.26962	89.07847	-3.27101	3.27101	-2.01409	2.01409

The "Sample Size Summary" table in Output 110.14.3 displays parameters for the sample size computation. Because the ACCTIME= option is not specified along with the ACCRATE= option, the minimum and maximum accrual times are derived.

Output 110.14.3 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Survival
Null Hazard Rate	0.03466
Hazard Rate (Group A)	0.01733
Hazard Rate (Group B)	0.03466
Hazard Ratio	0.5
log(Hazard Ratio)	-0.69315
Reference Hazards	Alt Ref
Accrual	Uniform
Accrual Rate	15
Min Accrual Time	5.938565
Min Sample Size	89.07847
Max Accrual Time	23.78469
Max Sample Size	356.7704
Max Number of Events	89.07847

With the minimum and maximum accrual times of 5.9386 and 23.7847, respectively, the ACCTIME=18 option specifies an accrual time of 18 for the trial.

The following statements invoke the SEQDESIGN procedure and specify the ACCTIME=18 option in the SAMPLESIZE statement to derive required sample sizes:

```
proc seqdesign;
   ErrorSpend: design nstages=4 method=errfuncobf
               ;
   samplesize model(ceiladjdesign=include)=twosamplesurvival
                   ( nullhazard=0.03466 hazard=0.01733
                     accrual=uniform accrate=15 acctime=18
                     ceiling=time);
run;
```

When CEILADJDESIGN=INCLUDE in the SAMPLESIZE statement, the "Design Information" table in Output 110.14.4 also displays the information for the adjusted design with ceiling times at the stages.

Output 110.14.4 Error Spending Design Information

The SEQDESIGN Procedure
Design: ErrorSpend

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.693147
Number of Stages	4
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	101.8279
Max Information	22.26962
Null Ref ASN (Percent of Fixed Sample)	101.2586
Alt Ref ASN (Percent of Fixed Sample)	77.73131
Adj Design Alpha	0.05
Adj Design Beta	0.08832
Adj Design Power	0.91168
Adj Design Max Information (Percent of Fixed Sample)	101.8064
Adj Design Max Information	23.23194
Adj Design Null Ref ASN (Percent of Fixed Sample)	101.2299
Adj Design Alt Ref ASN (Percent of Fixed Sample)	76.32612

Because of the ceiling sample sizes in the adjusted design, Type I and Type II error levels cannot be maintained simultaneously. When BOUNDARYKEY=BOTH (the default) in the DESIGN statement, only the Type I error level is maintained for the adjusted design. The adjusted design has a power of 0.91168, and it reflects the change of maximum information from 22.2696 to 23.2319.

The “Sample Size Summary” table in Output 110.14.5 displays the follow-up time and maximum sample size with the specified accrual time. When you specify the CEILING=TIME option (which is the default), the required times at the stages are rounded up to integers for additional statistics, and the table also displays the follow-up time and total time that correspond to these ceiling times at the stages.

Output 110.14.5 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Survival
Null Hazard Rate	0.03466
Hazard Rate (Group A)	0.01733
Hazard Rate (Group B)	0.03466
Hazard Ratio	0.5
log(Hazard Ratio)	-0.69315
Reference Hazards	Alt Ref
Accrual	Uniform
Accrual Rate	15
Accrual Time	18
Follow-up Time	7.133226
Total Time	25.13323
Max Number of Events	89.07847
Max Sample Size	270
Expected Sample Size (Null Ref)	269.9206
Expected Sample Size (Alt Ref)	263.1141
Follow-up Time (Ceiling Time)	8
Total Time (Ceiling Time)	26

The "Number of Events (D) and Sample Sizes (N)" table in Output 110.14.6 displays the required time at each stage, in both fractional and integer numbers. The derived times under the heading "Fractional Time" are not integers. These times are rounded up to integers under the heading "Ceiling Time." The table also displays the numbers of events and sample sizes at each stage.

Output 110.14.6 Number of Events and Sample Sizes

Numbers of Events (D) and Sample Sizes (N)
Two-Sample Log-Rank Test

				Fractional Time				
Stage	D	D(Grp 1)	D(Grp 2)	Time	N	N(Grp 1)	N(Grp 2)	Information
1	22.27	7.73	14.54	11.2631	168.95	84.47	84.47	5.5674
2	44.54	15.73	28.81	16.2875	244.31	122.16	122.16	11.1348
3	66.81	23.93	42.88	20.4926	270.00	135.00	135.00	16.7022
4	89.08	32.51	56.57	25.1332	270.00	135.00	135.00	22.2696

Numbers of Events (D) and Sample Sizes (N)
Two-Sample Log-Rank Test

				Ceiling Time				
Stage	D	D(Grp 1)	D(Grp 2)	Time	N	N(Grp 1)	N(Grp 2)	Information
1	25.11	8.74	16.37	12	180.00	90.00	90.00	6.2781
2	48.22	17.07	31.16	17	255.00	127.50	127.50	12.0552
3	69.39	24.90	44.48	21	270.00	135.00	135.00	17.3468
4	92.93	34.04	58.89	26	270.00	135.00	135.00	23.2319

The "Ceiling-Adjusted Design Boundary Information" table in Figure 110.14.7 displays boundary information, similar to Figure 110.14.2 but with ceiling times at the stages.

Output 110.14.7 Adjusted O'Brien-Fleming Boundary Information

Ceiling-Adjusted Design Boundary Information (Standardized Z Scale) Null Reference = 0							
	Information Level			Alternative Reference		Boundary Values	
						Lower	Upper
Stage	Proportion	Actual	Events	Lower	Upper	Alpha	Alpha
1	0.2702	6.278064	25.11225	-1.73675	1.73675	-4.15591	4.15591
2	0.5189	12.05517	48.22068	-2.40665	2.40665	-2.90189	2.90189
3	0.7467	17.34678	69.38712	-2.88692	2.88692	-2.36973	2.36973
4	1.0000	23.23194	92.92776	-3.34094	3.34094	-2.01362	2.01362

Because the times have integer values, the information levels at the stages are not equally spaced in this example, but the design is still an O'Brien-Fleming error spending design.

Alternatively, you can specify the CEILING=N option in the SAMPLESIZE statement as follows to derive additional sample size information that includes ceiling sample sizes at each stage:

```
proc seqdesign;
   ErrorSpend: design nstages=4 method=errfuncobf
               ;
   samplesize model(ceiladjdesign=include)=twosamplesurvival
                   ( nullhazard=0.03466 hazard=0.01733
                     accrual=uniform accrate=15 acctime=18
                     ceiling=n);
run;
```

When CEILING=N, the required sample sizes at the stages are rounded up to integers, and the ceiling-adjusted design has ceiling sample sizes at the stages.

Example 110.15: Computing Sample Size for Survival Data with Truncated Exponential Accrual

(View the complete code for this example (seqdx15.sas) in the example repository.)

This example illustrates sample size computation for survival data when the accrual is not uniform and when the data contain individual loss to follow-up.

Following the derivations in Example 110.14, the following statements invoke the SEQDESIGN procedure for sample size computation with exponential accrual and individual loss to follow-up:

```
proc seqdesign;
   OBrienFleming: design nstages=4 method=obf
                  ;
   samplesize model=twosamplesurvival
                   ( nullhazard=0.03466
                     hazard=0.01733
                     accrual=exp(parm=-0.1)
                     loss=exp(hazard=0.05)
                     acctime=20);
run;
```

The SAMPLESIZE statement derives required sample sizes for the test. The MODEL=TWOSAMPLESURVIVAL option specifies a log-rank test to compare two survival distributions for the treatment effect. The NULL-HAZARD=0.03466 option specifies null hazard rates for the two groups under the null hypothesis, and the HAZARD=0.01733 option specifies the hazard rate for the first group under the alternative hypothesis.

The ACCTIME= option specifies the accrual time T_a, and the ACCRUAL=EXP(POWER=–0.1) option specifies that the individual accrual be truncated exponential with a scaled power parameter $\gamma_0 = -0.1$. That is, the truncated exponential accrual has a power parameter $\gamma = \gamma_0 / T_a = -0.1 / 20 = -0.005$ in the accrual time T_a. The LOSS=EXP(HAZARD=0.05) option specifies an exponential loss function with a hazard rate $\tau = 0.05$. For a detailed derivation of these required sample sizes, see the section "Test for Two Survival Distributions with a Log-Rank Test" on page 9201.

The "Design Information" table in Output 110.15.1 displays design specifications and the derived maximum information.

Output 110.15.1 O'Brien-Fleming Design Information

The SEQDESIGN Procedure
Design: OBrienFleming

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	O'Brien-Fleming
Boundary Key	Both
Alternative Reference	0.693147
Number of Stages	4
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	102.2163
Max Information	22.35451
Null Ref ASN (Percent of Fixed Sample)	101.5728
Alt Ref ASN (Percent of Fixed Sample)	76.7397

The "Boundary Information" table in Output 110.15.2 displays information level, alternative reference, and boundary values.

Output 110.15.2 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference		Boundary Values	
Stage	Proportion	Actual	Events	Lower	Upper	Lower Alpha	Upper Alpha
1	0.2500	5.588627	22.35451	-1.63862	1.63862	-4.04859	4.04859
2	0.5000	11.17725	44.70902	-2.31736	2.31736	-2.86278	2.86278
3	0.7500	16.76588	67.06352	-2.83817	2.83817	-2.33745	2.33745
4	1.0000	22.35451	89.41803	-3.27724	3.27724	-2.02429	2.02429

The "Sample Size Summary" table in Output 110.15.3 displays parameters for the sample size computation. Because the ACCNOBS= option is not specified along with the ACCTIME= option, the minimum and maximum sample sizes are derived.

Output 110.15.3 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Survival
Null Hazard Rate	0.03466
Hazard Rate (Group A)	0.01733
Hazard Rate (Group B)	0.03466
Hazard Ratio	0.5
log(Hazard Ratio)	-0.69315
Reference Hazards	Alt Ref
Accrual	Truncated Exponential with Losses
Scaled Exponential Parameter	-0.1
Loss Hazard Rate	0.05
Accrual Time	20
Min Accrual Sample Size	269
Max Accrual Sample Size	553
Max Number of Events	89.41803

With the minimum and maximum sample sizes of 269 and 553, respectively, the ACCNOBS=360 option specifies an accrual sample size of 360 for the trial.

```
proc seqdesign;
   OBrienFleming: design nstages=4
                  method=obf
                  ;
   samplesize model=twosamplesurvival
                    ( nullhazard=0.03466
                      hazard=0.01733
                      accrual=exp(parm=-0.1)
                      loss=exp(hazard=0.05)
                      acctime=20 accnobs=360 );
run;
```

With the specified accrual sample size and accrual time, the "Sample Size Summary" table in Output 110.15.4 displays the follow-up time and maximum sample size with the specified accrual time. When you specify the CEILING=TIME option (which is the default), the required times at the stages are rounded up to integers for additional statistics, and the table also displays the follow-up time and total time that correspond to these ceiling times at the stages.

Output 110.15.4 Sample Size Summary

The SEQDESIGN Procedure
Design: OBrienFleming

Sample Size Summary	
Test	Two-Sample Survival
Null Hazard Rate	0.03466
Hazard Rate (Group A)	0.01733
Hazard Rate (Group B)	0.03466
Hazard Ratio	0.5
log(Hazard Ratio)	-0.69315
Reference Hazards	Alt Ref
Accrual	Truncated Exponential with Losses
Scaled Exponential Parameter	-0.1
Loss Hazard Rate	0.05
Accrual Time	20
Follow-up Time	9.100306
Total Time	29.10031
Max Number of Events	89.41803
Max Sample Size	360
Expected Sample Size (Null Ref)	359.7605
Expected Sample Size (Alt Ref)	342.9204
Follow-up Time (Ceiling Time)	10
Total Time (Ceiling Time)	30

The "Number of Events (D) and Sample Sizes (N)" table in Output 110.15.5 displays the required time at each stage, in both fractional and integer numbers. The derived times under the heading "Fractional Time" are not integers. These times are rounded up to integers under the heading "Ceiling Time." The table also displays the numbers of events and sample sizes at each stage.

Output 110.15.5 Number of Events and Sample Sizes

Numbers of Events (D) and Sample Sizes (N)
Two-Sample Log-Rank Test

				Fractional Time				
Stage	D	D(Grp 1)	D(Grp 2)	Time	N	N(Grp 1)	N(Grp 2)	Information
1	22.35	7.74	14.62	11.4005	200.79	100.39	100.39	5.5886
2	44.71	15.69	29.01	17.0454	304.52	152.26	152.26	11.1773
3	67.06	23.81	43.25	21.9812	360.00	180.00	180.00	16.7659
4	89.42	32.39	57.03	29.1003	360.00	180.00	180.00	22.3545

Numbers of Events (D) and Sample Sizes (N)
Two-Sample Log-Rank Test

				Ceiling Time				
Stage	D	D(Grp 1)	D(Grp 2)	Time	N	N(Grp 1)	N(Grp 2)	Information
1	24.46	8.48	15.98	12	211.67	105.83	105.83	6.1162
2	48.97	17.23	31.75	18	322.36	161.18	161.18	12.2436
3	67.14	23.84	43.30	22	360.00	180.00	180.00	16.7851
4	91.46	33.21	58.25	30	360.00	180.00	180.00	22.8649

Alternatively, you can specify the CEILING=N option in the SAMPLESIZE statement as follows to derive additional sample size information with ceiling sample sizes at each stage:

```
proc seqdesign;
   OBrienFleming: design nstages=4
                  method=obf
                  ;
   samplesize model=twosamplesurvival
                    ( nullhazard=0.03466
                      hazard=0.01733
                      accrual=exp(parm=-0.1)
                      loss=exp(hazard=0.05)
                      acctime=20 accnobs=360
                      ceiling=n);
run;
```

When you specify CEILING=N, the required sample sizes at the stages are rounded up to integers for additional statistics, and the "Sample Size Summary" table in Output 110.15.6 also displays the accrual time, follow-up time, and maximum sample size that correspond to these ceiling sample sizes.

Output 110.15.6 Sample Size Summary

The SEQDESIGN Procedure
Design: OBrienFleming

Sample Size Summary	
Test	Two-Sample Survival
Null Hazard Rate	0.03466
Hazard Rate (Group A)	0.01733
Hazard Rate (Group B)	0.03466
Hazard Ratio	0.5
log(Hazard Ratio)	-0.69315
Reference Hazards	Alt Ref
Accrual	Truncated Exponential with Losses
Scaled Exponential Parameter	-0.1
Loss Hazard Rate	0.05
Accrual Time	20
Follow-up Time	9.100306
Total Time	29.10031
Max Number of Events	89.41803
Max Sample Size	360
Expected Sample Size (Null Ref)	359.7605
Expected Sample Size (Alt Ref)	342.9204
Accrual Time (Ceiling N)	20
Follow-up Time (Ceiling N)	9.100306
Max Sample Size (Ceiling N)	360

The "Number of Events (D) and Sample Sizes (N)" table in Output 110.15.7 displays the required sample sizes at each stage, in both fractional and integer numbers. The derived numbers under the heading "Fractional N" are not integers. These numbers are rounded up to integers to match the sample size allocation under the heading "Ceiling N (Matched)." The table also displays the numbers of events and time at each stage.

Output 110.15.7 Number of Events and Sample Sizes

Numbers of Events (D) and Sample Sizes (N) Two-Sample Log-Rank Test								
				Fractional N				
Stage	D	D(Grp 1)	D(Grp 2)	Time	N	N(Grp 1)	N(Grp 2)	Information
1	22.35	7.74	14.62	11.4005	200.79	100.39	100.39	5.5886
2	44.71	15.69	29.01	17.0454	304.52	152.26	152.26	11.1773
3	67.06	23.81	43.25	21.9812	360.00	180.00	180.00	16.7659
4	89.42	32.39	57.03	29.1003	360.00	180.00	180.00	22.3545

Numbers of Events (D) and Sample Sizes (N) Two-Sample Log-Rank Test								
				Ceiling N (Matched)				
Stage	D	D(Grp 1)	D(Grp 2)	Time	N	N(Grp 1)	N(Grp 2)	Information
1	22.59	7.82	14.77	11.4674	202	101	101	5.6466
2	45.06	15.82	29.24	17.1246	306	153	153	11.2645
3	67.06	23.81	43.25	21.9812	360	180	180	16.7659
4	89.42	32.39	57.03	29.1003	360	180	180	22.3545

References

Armitage, P., McPherson, C. K., and Rowe, B. C. (1969). "Repeated Significance Test on Accumulating Data." *Journal of the Royal Statistical Society, Series A* 132:235–244.

Chow, S.-C., and Liu, J.-P. (1998). *Design and Analysis of Clinical Trials: Concept and Methodologies*. New York: John Wiley & Sons.

Chow, S.-C., Shao, J., and Wang, H. (2003). *Sample Size Calculations in Clinical Research*. Boca Raton, FL: CRC Press.

Cox, D. R., and Hinkley, D. V. (1974). *Theoretical Statistics*. London: Chapman & Hall.

DeMets, D. L., Furberg, C. D., and Friedman, L. M. (2006). *Data Monitoring in Clinical Trials*. New York: Springer.

Diggle, P. J., Heagerty, P., Liang, K.-Y., and Zeger, S. L. (2002). *Analysis of Longitudinal Data*. 2nd ed. New York: Oxford University Press.

Dmitrienko, A., Molenberghs, G., Chuang-Stein, C., and Offen, W. (2005). *Analysis of Clinical Trials Using SAS: A Practical Guide*. Cary, NC: SAS Institute Inc.

Efron, B., and Hinkley, D. V. (1978). "Assessing the Accuracy of the Maximum Likelihood Estimator: Observed versus Expected Fisher Information." *Biometrika* 65:457–483.

Ellenberg, S. S., Fleming, T. R., and DeMets, D. L. (2003). *Data Monitoring Committees in Clinical Trials*. New York: John Wiley & Sons.

Emerson, S. S. (1996). "Statistical Packages for Group Sequential Methods." *American Statistician* 50:183–192.

Emerson, S. S., and Fleming, T. R. (1989). "Symmetric Group Sequential Designs." *Biometrics* 45:905–923.

Emerson, S. S., Kittelson, J. M., and Gillen, D. L. (2005). "On the Use of Stochastic Curtailment in Group Sequential Clinical Trials." UW Biostatistics Working Paper Series, No. 243. http://biostats.bepress.com/uwbiostat/paper243/.

Food and Drug Administration (1998). "E9: Statistical Principles for Clinical Trials." *Federal Register* 63:49583–49598.

Freedman, L. S. (1982). "Tables of the Number of Patients Required in Clinical Trials Using the Logrank Test." *Statistics in Medicine* 1:121–129.

Haybittle, J. L. (1971). "Repeated Assessment of Results in Clinical Trials of Cancer Treatment." *British Journal of Radiology* 44:793–797.

Hsieh, F. Y. (1992). "Comparing Sample Size Formulae for Trials with Unbalanced Allocation Using the Logrank Test." *Statistics in Medicine* 11:1091–1098.

Hsieh, F. Y., and Lavori, P. W. (2000). "Sample-Size Calculations for the Cox Proportional Hazards Regression Model with Nonbinary Covariates." *Controlled Clinical Trials* 21:552–560.

Hwang, I. K., Shih, W. J., and DeCani, J. S. (1990). "Group Sequential Designs Using a Family of Type I Error Probability Spending Functions." *Statistics in Medicine* 9:1439–1445.

Jennison, C., and Turnbull, B. W. (1990). "Statistical Approaches to Interim Monitoring of Medical Trials: A Review and Commentary." *Statistical Science* 5:299–317.

Jennison, C., and Turnbull, B. W. (2000). *Group Sequential Methods with Applications to Clinical Trials*. New York: Chapman & Hall.

Kalbfleisch, J. D., and Prentice, R. L. (1980). *The Statistical Analysis of Failure Time Data*. New York: John Wiley & Sons.

Kim, K., and DeMets, D. L. (1987). "Design and Analysis of Group Sequential Tests Based on the Type I Error Spending Rate Function." *Biometrika* 74:149–154.

Kim, K., and Tsiatis, A. A. (1990). "Study Duration for Clinical Trials with Survival Response and Early Stopping Rule." *Biometrics* 46:81–92.

Kittelson, J. M., and Emerson, S. S. (1999). "A Unifying Family of Group Sequential Test Designs." *Biometrics* 55:874–882.

Lachin, J. M., and Foulkes, M. A. (1986). "Evaluation of Sample Size and Power for Analyses of Survival with Allowance for Nonuniform Patient Entry, Losses to Follow-Up, Noncompliance, and Stratification." *Biometrics* 42:507–519.

Lan, K. K. G., and DeMets, D. L. (1983). "Discrete Sequential Boundaries for Clinical Trials." *Biometrika* 70:659–663.

Lan, K. K. G., and DeMets, D. L. (2009). "Further Comments on the Alpha-Spending Function." *Statistics in Biosciences* 1:95–111.

Lan, K. K. G., Lachin, J. M., and Bautista, O. (2003). "Over-ruling a Group Sequential Boundary: A Stopping Rule versus a Guideline." *Statistics in Medicine* 22:3347–3355.

Lan, K. K. G., Simon, R., and Halperin, M. (1982). "Stochastically Curtailed Tests in Long-Term Clinical Trials." *Sequential Analysis* 1:207–219.

Lindgren, B. W. (1976). *Statistical Theory*. 3rd ed. New York: Macmillan.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Mehta, C. R., and Tsiatis, A. A. (2001). "Flexible Sample Size Considerations under Information Based Interim Monitoring." *Drug Information Journal* 35:1095–1112.

O'Brien, P. C., and Fleming, T. R. (1979). "A Multiple Testing Procedure for Clinical Trials." *Biometrics* 35:549–556.

O'Neill, R. T. (1994). "Interim Analysis: A Regulatory Perspective on Data Monitoring and Interim Analysis." In *Statistics in the Pharmaceutical Industry*, edited by C. R. Buncher and J.-Y. Tsay, 285–290. New York: Marcel Dekker.

Pampallona, S., and Tsiatis, A. A. (1994). "Group Sequential Designs for One-Sided and Two-Sided Hypothesis Testing with Provision for Early Stopping in Favor of the Null Hypothesis." *Journal of Statistical Planning and Inference* 42:19–35.

Peto, R., Pike, M. C., Armitage, P., Breslow, N. E., Cox, D. R., Howard, S. V., Mantel, N., McPherson, K., Peto, J., and Smith, P. G. (1976). "Design and Analysis of Randomized Clinical Trials Requiring Prolonged Observation of Each Patient: I. Introduction and Design." *British Journal of Cancer* 34:585–612.

Pocock, S. J. (1977). "Group Sequential Methods in the Design and Analysis of Clinical Trials." *Biometrika* 64:191–199.

Pocock, S. J. (1982). "Interim Analyses for Randomized Clinical Trials: The Group Sequential Approach." *Biometrics* 38:153–162.

Pocock, S. J., and White, I. (1999). "Trials Stopped Early: Too Good to Be True?" *Lancet* 353:943–944.

Proschan, M. A., Lan, K. K. G., and Wittes, J. T. (2006). *Statistical Monitoring of Clinical Trials*. New York: Springer.

Rudser, K. D., and Emerson, S. S. (2007). "Implementing Type I and Type II Error Spending for Two-Sided Group Sequential Designs." *Contemporary Clinical Trials* 29:351–358.

Schoenfeld, D. A. (1981). "The Asymptotic Properties of Nonparametric Tests for Comparing Survival Distributions." *Biometrika* 68:316–319.

Schoenfeld, D. A. (1983). "Sample-Size Formula for the Proportional-Hazards Regression Model." *Biometrics* 39:499–503.

Senn, S. (1997). *Statistical Issues in Drug Development*. New York: John Wiley & Sons.

Snapinn, S. M. (2000). "Noninferiority Trials." *Current Controlled Trials in Cardiovascular Medicine* 1:19–21.

Wang, S. K., and Tsiatis, A. A. (1987). "Approximately Optimal One Parameter Boundaries for Group Sequential Trials." *Biometrics* 43:193–199.

Ware, J. H., Muller, J. E., and Braunwald, E. (1985). "The Futility Index: An Approach to the Cost-Effective Termination of Randomized Clinical Trials." *American Journal of Medicine* 78:635–643.

Whitehead, J. (1997). *The Design and Analysis of Sequential Clinical Trials*. Rev. 2nd ed. Chichester, UK: John Wiley & Sons.

Whitehead, J. (2001). "Use of the Triangular Test in Sequential Clinical Trials." In *Handbook of Statistics in Clinical Oncology*, edited by J. Crowley, 211–228. New York: Marcel Dekker.

Whitehead, J., and Jones, D. R. (1979). "The Analysis of Sequential Clinical Trials." *Biometrika* 66:443–452.

Whitehead, J., and Stratton, I. (1983). "Group Sequential Clinical Trials with Triangular Continuation Regions." *Biometrics* 39:227–236.

Zhu, L., Ni, L., and Yao, B. (2011). "Group Sequential Methods and Software Applications." *American Statistician* 65:127–135.

Chapter 111
The SEQTEST Procedure

Contents

Overview: SEQTEST Procedure . . . **9326**
- Features of the SEQTEST Procedure . . . 9329
- Output from the SEQTEST Procedure . . . 9330

Getting Started: SEQTEST Procedure . . . **9331**

Syntax: SEQTEST Procedure . . . **9344**
- PROC SEQTEST Statement . . . 9344

Details: SEQTEST Procedure . . . **9355**
- Input Data Sets . . . 9355
- Boundary Variables . . . 9357
- Information Level Adjustments at Future Stages . . . 9360
- Boundary Adjustments for Information Levels . . . 9361
- Boundary Adjustments for Minimum Error Spending . . . 9363
- Boundary Adjustments for Overlapping Lower and Upper β Boundaries . . . 9364
- Stochastic Curtailment . . . 9364
- Repeated Confidence Intervals . . . 9367
- Analysis after a Sequential Test . . . 9368
- Available Sample Space Orderings in a Sequential Test . . . 9369
- Applicable Tests and Sample Size Computation . . . 9371
- Table Output . . . 9372
- ODS Table Names . . . 9376
- Graphics Output . . . 9376
- ODS Graphics . . . 9378

Examples: SEQTEST Procedure . . . **9379**
- Example 111.1: Testing the Difference between Two Proportions . . . 9379
- Example 111.2: Testing an Effect in a Regression Model . . . 9390
- Example 111.3: Testing an Effect with Early Stopping to Accept H_0 . . . 9404
- Example 111.4: Testing a Binomial Proportion . . . 9416
- Example 111.5: Comparing Two Proportions with a Log Odds Ratio Test . . . 9428
- Example 111.6: Comparing Two Survival Distributions with a Log-Rank Test . . . 9439
- Example 111.7: Testing an Effect in a Proportional Hazards Regression Model . . . 9452
- Example 111.8: Testing an Effect in a Logistic Regression Model . . . 9464
- Example 111.9: Conducting a Trial with a Nonbinding Acceptance Boundary . . . 9476

References . . . **9489**

Overview: SEQTEST Procedure

The purpose of the SEQTEST procedure is to perform interim analyses for clinical trials. Clinical trials are experiments on human beings to demonstrate the efficacy and safety of new drugs or treatments. A simple example is a trial to test the effectiveness of a new drug in humans by comparing the outcomes in a group of patients who receive the new drug with the outcomes in a comparable group of patients who receive a placebo.

A clinical trial is conducted according to a plan called a *protocol*. A protocol details the objectives of the trial, the data collection process, and the analyses of the data. The protocol contains information such as a null hypothesis and an alternative hypothesis, a test statistic, the probability α of a Type I error (incorrectly rejecting the null hypothesis), the probability β of a Type II error (incorrectly accepting the null hypothesis), the sample size needed to attain a specified power (probability of correctly rejecting the null hypothesis) of $1 - \beta$ at an alternative reference, and critical values that are associated with the test statistic for hypothesis testing.

In a fixed-sample trial, data about all individuals are first collected and then examined at the end of the study. Most major trials have data safety monitoring boards or data monitoring committees that periodically monitor safety and efficacy data during the trial and recommend that a trial be stopped for safety concerns such as an unacceptable toxicity level. In certain rare situations, the board or committee might even recommend that a trial be stopped for efficacy. In contrast to a fixed-sample trial, a group sequential trial provides for interim analyses before the formal completion of the trial while maintaining the specified overall Type I and Type II error probability levels.

A group sequential trial is most useful in situations where it is important to monitor the trial to prevent unnecessary exposure of patients to an unsafe new drug, or alternatively to a placebo treatment if the new drug shows significant improvement. If a group sequential trial stops early, then it usually requires fewer participants than a corresponding fixed-sample trial.

Thus, in most cases, if a group sequential trial stops early for safety of the new treatment, fewer patients will be exposed to the new treatment than in the fixed-sample trial. Also, if a trial stops early for efficacy of the new treatment, the new treatment will be available sooner than it would be in a fixed-sample trial. Furthermore, if a trial stops early, this can also save time and resources.

A group sequential design provides detailed specifications for a group sequential trial. In addition to the usual specification for a fixed-sample design, it provides the total number of stages (the number of interim stages plus a final stage) and a stopping criterion to reject, to accept, or to either reject or accept the null hypothesis at each interim stage. It also provides critical values and the sample size at each stage for the trial.

At each interim stage, the data collected at the current stage in addition to the data collected at previous stages are analyzed, and statistics such as a maximum likelihood test statistic and its associated standard error are computed. The test statistic is then compared with its corresponding critical values at the stage, and the trial is stopped or continued. If a trial continues to the final stage, the null hypothesis is either rejected or accepted. The critical values for each stage are chosen in such a way to maintain the overall α level, the overall β level, or both the overall α and β levels.

Figure 111.1 shows a two-sided symmetric group sequential trial that stops early to reject the null hypothesis that the parameter Trt is zero.

Figure 111.1 Sequential Plot for Two-Sided Test

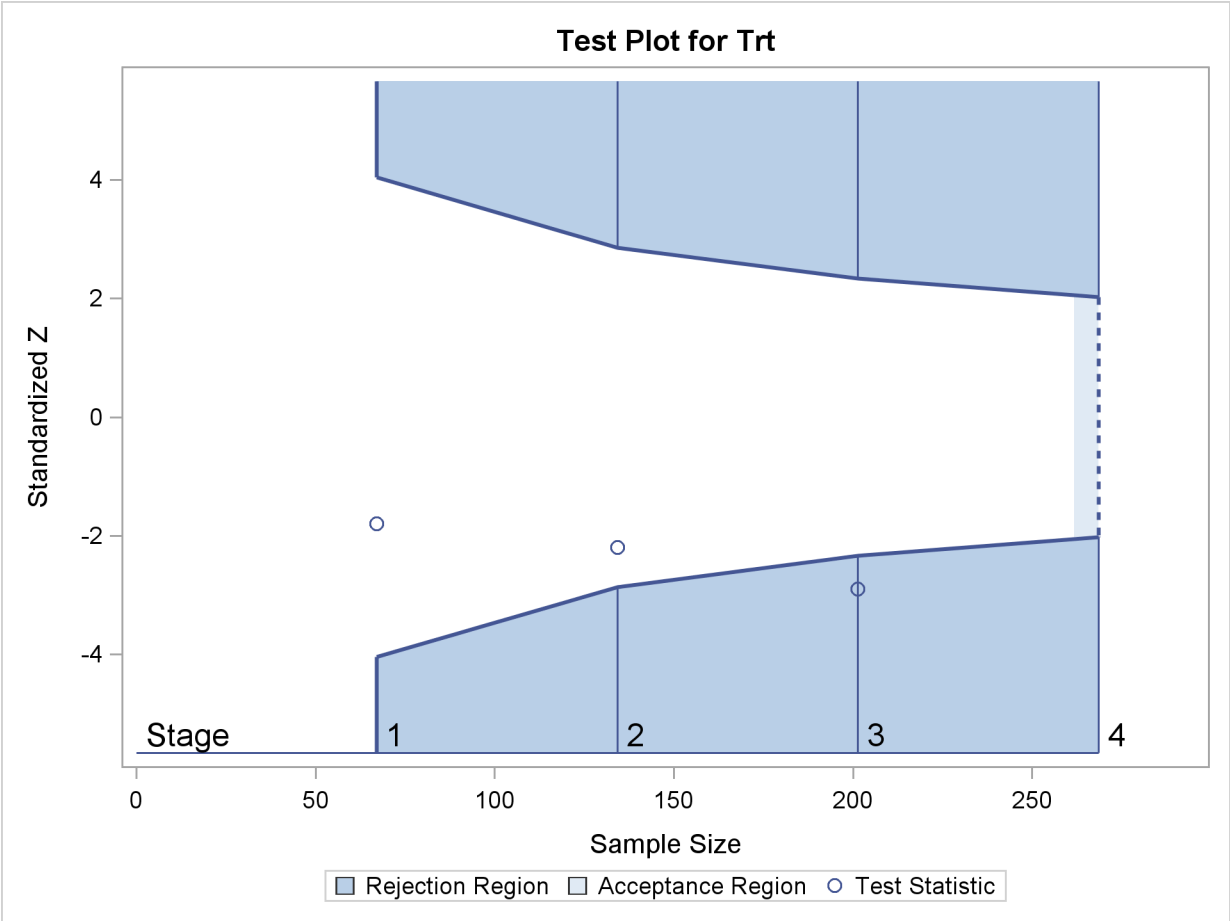

The trial has four stages, which are indicated by vertical lines with accompanying stage numbers. With early stopping to reject the null hypothesis, the lower rejection boundary is constructed by connecting the lower critical values (boundary values) for the stages. Similarly, the upper rejection boundary is constructed by connecting the upper critical values for the stages. The horizontal axis indicates the sample size for the group sequential trial, and the vertical axis indicates the boundary values and test statistics on the standardized Z scale.

At each interim stage, if the standardized Z test statistic falls into a rejection region (the darker shaded areas in Figure 111.1), the trial stops and the null hypothesis is rejected. Otherwise, the trial continues to the next stage. At the final stage (stage 4), the trial is rejected if Z falls into a rejection region. Otherwise, the trial is accepted. In Figure 111.1, the test statistic does not fall into the rejection regions for stages 1 and 2, and so the trial continues to stage 3. At stage 3, the test statistic falls into the rejection region, and the null hypothesis is rejected.

A group sequential trial usually involves six steps:

1. You specify the statistical details of the design, including the null and alternative hypotheses, a test statistic for the hypothesis test, the Type I and II error probabilities, a stopping criterion, the total number of stages, and the relative information level at each stage.

2. You compute the boundary values for the trial based on the specifications in Step 1. You also compute the sample size required at each stage for the specified hypothesis test.

3. At each stage, you collect additional data with the required sample sizes. The data available at each stage include the data collected at previous stages in addition to the data collected at the current stage.

4. At each stage, you analyze the available data with a procedure such as the REG procedure, and you compute the test statistic.

5. At each stage, you compare the test statistic with the corresponding boundary values. You stop the trial to reject or accept the hypothesis, or you continue the trial to the next stage. If you continue the trial to the final stage, you either accept or reject the hypothesis.

6. After the trial stops, you compute parameter estimates, confidence limits for the parameter, and a *p*-value for the hypothesis test.

You use the companion SEQDESIGN procedure at Step 2 to compute the boundary values and required sample sizes for the trial. You use the SEQTEST procedure at Step 5 to compare the test statistic with its boundary values. At stage 1, the boundary values are derived by using the boundary information tables created by the SEQDESIGN procedure. These boundary information tables are structured for input to the SEQTEST procedure. At each subsequent stage, the boundary values are derived by using the test information tables created by the SEQTEST procedure at the previous stage. These test information tables are also structured for input to the SEQTEST procedure. You also use the SEQTEST procedure at Step 6 to compute parameter estimates, confidence limits, and *p*-values after the trial stops.

Note that for some clinical trials, the information levels are derived from statistics based on individuals specified in the design plan and might not reach the target maximum information level. For example, if an estimate of the variance is used to compute the required sample size for a group sequential trial, the computed variance at each stage might not be the same as the estimated variance. Thus, instead of specifying the number of individuals in the protocol, the information level can be specified. You can then adjust the sample sizes with the updated variance estimates at interim stages to achieve the target maximum information level for the trial (Jennison and Turnbull 2000, p. 295).

The flowchart in Figure 111.2 summarizes the steps in a typical group sequential trial and the relevant SAS procedures.

Figure 111.2 Group Sequential Trial

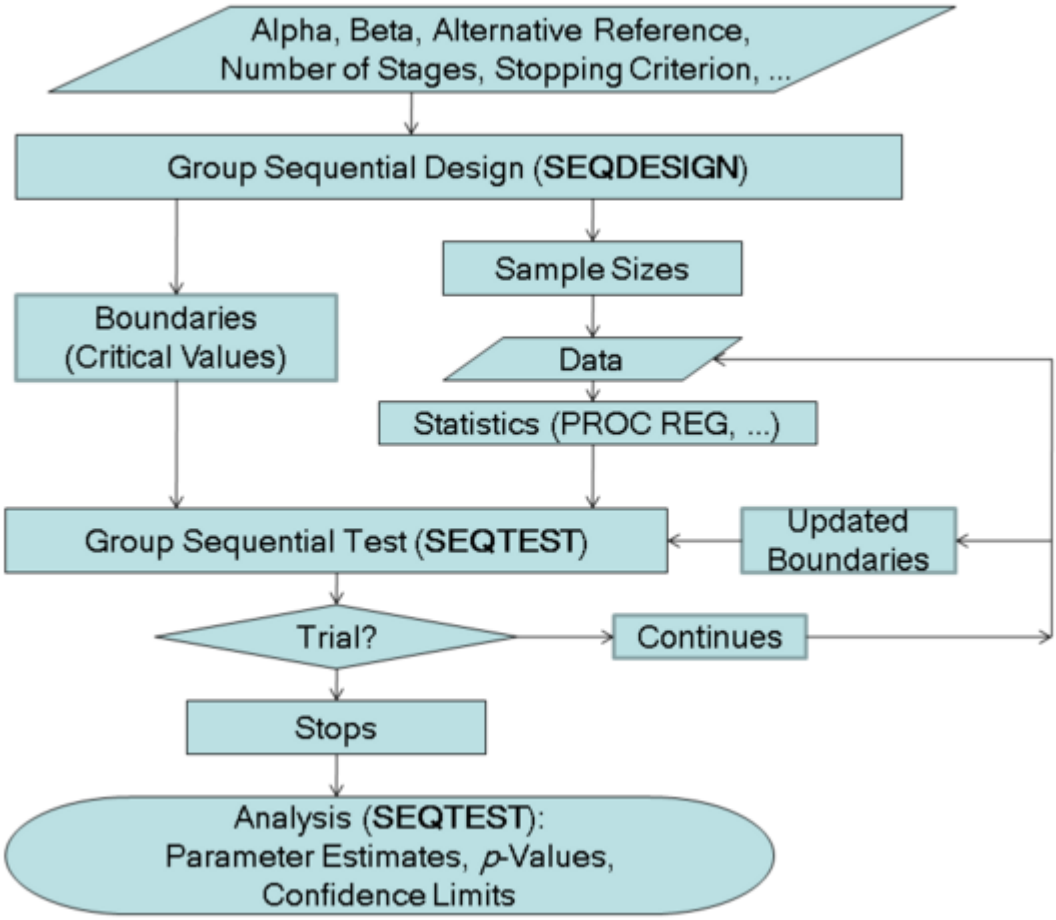

Features of the SEQTEST Procedure

At each stage, the data are analyzed with a statistical procedure such as the REG procedure, and a test statistic and its associated information level are computed. The information level is the amount of information available about the unknown parameter. For a maximum likelihood statistic, the information level is the inverse of its variance.

At each stage, you use the SEQTEST procedure to compare the test statistic with its boundary values. At stage 1, the boundary values are derived by using the boundary information tables created by the SEQDESIGN procedure. At each subsequent stage, the boundary values are derived by using the test information tables created by the SEQTEST procedure at the previous stage.

If the observed information level does not match the corresponding information level in the BOUNDARY= data set, the SEQTEST procedure modifies the boundary values to adjust for new information levels at the current and subsequent stages. See the section "Boundary Adjustments for Information Levels" on page 9361 for a detailed description of these boundary adjustments.

Either you can specify the test statistic and its information level in the DATA= input data set, or you can specify the test statistic and its associated standard error in the PARMS= input data set. With the PARMS=

input data set, the information level for the test statistic is computed from its standard error. See the section "Input Data Sets" on page 9355 for a detailed description of these input data sets.

At the end of a trial, the parameter estimate is computed. The median unbiased estimate, confidence limits, and p-value depend on the specified sample space ordering. A sample space ordering specifies the ordering for test statistics that result in the stopping of a trial. That is, for all the statistics in the rejection region and in acceptance region, the SEQTEST procedure provides three different sample space orderings: the stagewise ordering uses counterclockwise ordering around the continuation region, the LR ordering uses the distance between the observed Z statistic z and its hypothetical value, and the MLE ordering uses the observed maximum likelihood estimate. See the section "Available Sample Space Orderings in a Sequential Test" on page 9369 for a detailed description of these orderings.

Output from the SEQTEST Procedure

In addition to the adjusted boundary values and test results for the group sequential trial, the SEQTEST procedure also computes the following quantities:

- average sample numbers (as percentages of the corresponding fixed-sample sizes for nonsurvival data or fixed-sample numbers of events for survival data) under various hypothetical references, including the null and alternative references
- stopping probabilities at each stage under various hypothetical references to indicate how likely it is that the trial will stop at that stage
- conditional power given the most recently observed statistic under specified hypothetical references
- predictive power given the most recently observed statistic
- repeated confidence intervals for the parameter from the observed statistic at each stage
- parameter estimate, p-value for hypothesis testing, and median and confidence limits for the parameter at the conclusion of a sequential trial

Getting Started: SEQTEST Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/seqtgs.sas.)

The following example illustrates a clinical study that uses a two-sided O'Brien-Fleming design (O'Brien and Fleming 1979) to stop the trial early for ethical concerns about possible harm or for unexpectedly strong efficacy of the new drug.

Suppose that a pharmaceutical company is conducting a clinical trial to test the efficacy of a new cholesterol-lowering drug. The primary focus is low-density lipoprotein (LDL), the so-called bad cholesterol, which is a risk factor for coronary heart disease. LDL is measured in mg/dl, milligrams per deciliter of blood.

The trial consists of two groups of equally allocated patients with elevated LDL level: an experimental group given the new drug and a placebo control group. Suppose the changes in LDL level after the treatment for patients in the experimental and control groups are normally distributed with means μ_e and μ_c, respectively, and have a common variance σ^2. Then the null hypothesis of no effect for the new drug is $H_0 : \theta = \mu_e - \mu_c = 0$. Also suppose that the alternative reference $\theta = -10$ is the clinically meaningful difference that the trial should detect with a high probability (power), and that a good estimate of the standard deviation for the changes in LDL level is $\hat{\sigma} = 20$.

The following statements invoke the SEQDESIGN procedure and request a four-stage O'Brien-Fleming design for standardized normal test statistics:

```
ods graphics on;
proc seqdesign altref=-10.0;
   TwoSidedOBrienFleming: design nstages=4
                          method=obf
                          ;
   samplesize model=twosamplemean(stddev=20);
   ods output Boundary=Bnd_LDL;
run;
```

The ALTREF= option specifies the alternative reference, and the actual maximum information is derived in the SEQDESIGN procedure.

In the DESIGN statement, the label **TwoSidedOBrienFleming** identifies the design in the output tables. By default (or equivalently if you specify ALT=TWOSIDED and STOP=REJECT), the design has a two-sided alternative hypothesis with early stopping in the interim stages only to reject the null hypothesis. That is, at each interim stage, the trial will either be stopped to reject the null hypothesis or continue to the next stage.

The NSTAGES=4 option in the DESIGN statement specifies the total number of stages in the group sequential trial, including three interim stages and a final stage. In the SEQDESIGN procedure, the null hypothesis for the design is $H_0 : \theta = 0$. By default (or equivalently if you specify ALPHA=0.05 and BETA=0.10), the design has a Type I error probability $\alpha = 0.05$, and a Type II error probability $\beta = 0.10$, which corresponds to a power of $1 - \beta = 0.90$ at the alternative reference $H_1 : \theta = -10$.

For a two-sided design with early stopping to reject the null hypothesis, there are two boundaries for the design: an upper α (rejection) boundary that consists of upper rejection critical values and a lower α boundary that consists of lower rejection critical values. Each boundary is a set of critical values, one from each stage. With the METHOD=OBF option in the DESIGN statement, the O'Brien-Fleming method is used for the two boundaries for the design; see Figure 111.5.

The SAMPLESIZE statement with the MODEL=TWOSAMPLEMEAN option uses the derived maximum information to compute required sample sizes for a two-sample test for mean difference.

The ODS OUTPUT statement with the BOUNDARY=BND_LDL option creates an output data set named BND_LDL which contains the resulting boundary information. At each stage of the trial, data are collected and analyzed with a statistical procedure, and a test statistic and its corresponding information level are computed.

In this example, you can use the REG procedure to compute the maximum likelihood estimate $\hat{\theta}$ for the drug effect and the corresponding standard error for $\hat{\theta}$. At stage 1, you can use the SEQTEST procedure to compare the test statistic with adjusted boundaries that are derived from the boundary information stored in the BOUND_LDL data set. At each subsequent stage, you can use the SEQTEST procedure to compare the test statistic with adjusted boundaries that are derived from the boundary information stored in the test information table that was created by the SEQTEST procedure at the previous stage. The test information tables are structured for input to the SEQTEST procedure.

At each interim stage, the trial will either be stopped to reject the null hypothesis or continue to the next stage. At the final stage, the null hypothesis is either rejected or accepted.

By default (or equivalently if you specify INFO=EQUAL), the SEQDESIGN procedure derives boundary values with equally spaced information levels for all stages—that is, the same information increment between successive stages.

The "Design Information" table in Figure 111.3 displays design specifications and three derived statistics: the maximum information, the average sample number under the null hypothesis (Null Ref ASN), and the average sample number under the alternative hypothesis (Alt Ref ASN). Each statistic is expressed as a percentage of the identical statistic for the corresponding fixed-sample information. The average sample number is the expected sample size (for nonsurvival data) or expected number of events (for survival data). When you specify an alternative reference (in this case, ALTREF=–10), the actual maximum information 0.1074 is also computed. Note that for a symmetric two-sided design, the ALTREF=–10 option implies a lower alternative reference of –10 and an upper alternative reference of 10.

Figure 111.3 O'Brien-Fleming Design Information

The SEQDESIGN Procedure
Design: TwoSidedOBrienFleming

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	O'Brien-Fleming
Boundary Key	Both
Alternative Reference	-10
Number of Stages	4
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	102.2163
Max Information	0.107403
Null Ref ASN (Percent of Fixed Sample)	101.5728
Alt Ref ASN (Percent of Fixed Sample)	76.7397

The "Boundary Information" table in Figure 111.4 displays the information level, the lower and upper alternative references, and the lower and upper boundary values at each stage. By default (or equivalently if you specify INFO=EQUAL), the SEQDESIGN procedure uses equally spaced information levels for all stages.

Figure 111.4 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference		Boundary Values	
Stage	Proportion	Actual	N	Lower	Upper	Lower Alpha	Upper Alpha
1	0.2500	0.026851	42.96116	-1.63862	1.63862	-4.04859	4.04859
2	0.5000	0.053701	85.92233	-2.31736	2.31736	-2.86278	2.86278
3	0.7500	0.080552	128.8835	-2.83817	2.83817	-2.33745	2.33745
4	1.0000	0.107403	171.8447	-3.27724	3.27724	-2.02429	2.02429

The information proportion is the proportion of maximum information available at each stage. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the alternative references and boundary values are displayed with the standardized Z statistic scale. The alternative reference in the standardized Z scale at stage k is given by $\theta_1\sqrt{I_k}$, where θ_1 is the alternative reference and I_k is the information available at stage k, k = 1, 2, 3, 4.

In this example, a standardized Z statistic is computed by standardizing the parameter estimate of the effect in LDL level. A lower Z test statistic indicates a beneficial effect. Consequently, at each interim stage, if the standardized Z test statistic is less than or equal to the corresponding lower α boundary value, the hypothesis $H_0 : \theta = 0$ is rejected for efficacy. If the test statistic is greater than or equal to the corresponding upper α boundary value, the hypothesis H_0 is rejected for harmful effect. Otherwise, the process continues to the

next stage. At the final stage (stage 4), the hypothesis H_0 is rejected for efficacy if the Z statistic is less than or equal to the corresponding lower α boundary value –2.0243, and the hypothesis H_0 is rejected for harmful effect if the Z statistic is greater than or equal to the corresponding upper α boundary value 2.0243. Otherwise, the hypothesis of no significant difference is accepted.

Note that in a typical trial, the actual information levels do not match the information levels specified in the design. Consequently, the SEQTEST procedure modifies the boundary values stored in the BND_LDL data set to adjust for these new information levels.

If ODS Graphics is enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Figure 111.5.

Figure 111.5 O'Brien-Fleming Boundary Plot

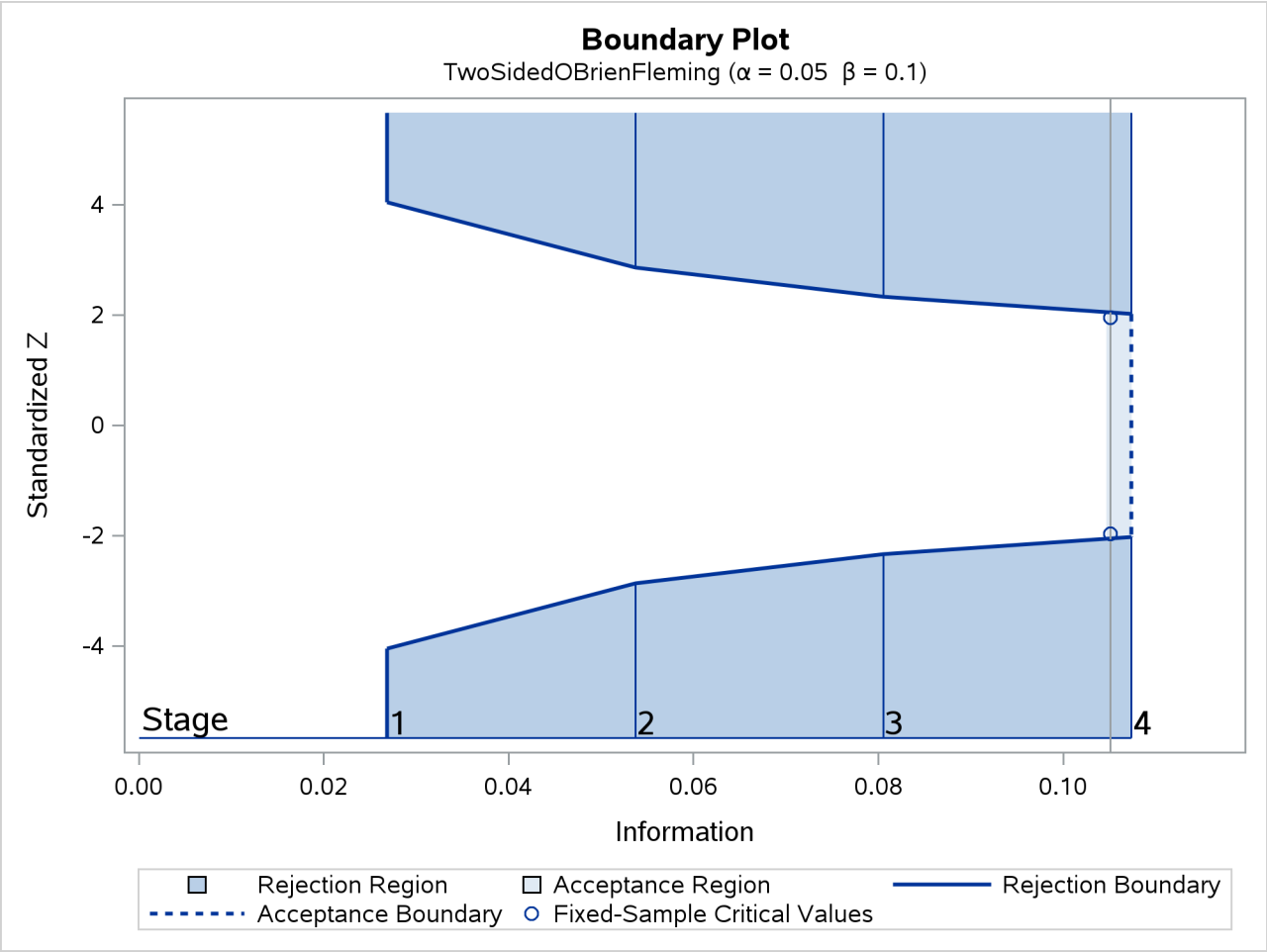

This boundary plot displays the boundary values in the "Boundary Information" table in Figure 111.4. The stages are indicated by vertical lines with accompanying stage numbers. The horizontal axis indicates the information levels for the stages. If a test statistic at an interim stage is in the rejection region (shaded area), the trial stops and the null hypothesis is rejected. If the statistic is not in any rejection region, the trial continues to the next stage. The plot also displays critical values for the corresponding fixed-sample design. The symbol "○" identifies the fixed-sample critical values of –1.96 and 1.96.

When you specify the SAMPLESIZE statement, the maximum information (either explicitly specified or

derived in the SEQDESIGN procedure) is used to compute the required sample sizes for the study. The MODEL=TWOSAMPLEMEAN(STDDEV=20) option specifies the test for the difference between two normal means. See the section "Test for the Difference between Two Normal Means" on page 9196 in Chapter 110, "The SEQDESIGN Procedure," for more information about how these required sample sizes are calculated.

The "Sample Size Summary" table in Figure 111.6 displays the parameters for the sample size computation and the resulting maximum and expected sample sizes.

Figure 111.6 Required Sample Size Summary

Sample Size Summary	
Test	Two-Sample Means
Mean Difference	-10
Standard Deviation	20
Max Sample Size	171.8447
Expected Sample Size (Null Ref)	170.7627
Expected Sample Size (Alt Ref)	129.0137

With the derived maximum information 0.1074 and the specified MODEL=TWOSAMPLEMEAN (STDDEV=20) option in the SAMPLESIZE statement, the total sample size in each group is

$$N_a = N_b = 2\,\sigma^2\, I_X = 2 \times 20^2 \times 0.1074 = 85.92$$

The "Sample Sizes (N)" table in Figure 111.7 displays the required sample sizes at each stage for the trial, in both fractional and integer numbers. The derived fractional sample sizes are displayed under the heading "Fractional N." These sample sizes are rounded up to integers under the heading "Ceiling N." By default (or equivalently if you specify WEIGHT=1 in the MODEL=TWOSAMPLEMEAN option), the sample sizes for the two groups are equal for the two-sample test.

Figure 111.7 Required Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Mean Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	42.96	21.48	21.48	0.0269	44	22	22	0.0275
2	85.92	42.96	42.96	0.0537	86	43	43	0.0538
3	128.88	64.44	64.44	0.0806	130	65	65	0.0812
4	171.84	85.92	85.92	0.1074	172	86	86	0.1075

In practice, integer sample sizes are used in the trial, and the resulting information levels increase slightly. Thus, each of the two groups needs 22, 43, 65, and 86 patients for the four stages, respectively.

Suppose that 22 patients are available in each group at stage 1 and that their measurements for LDL are saved in the data set LDL_1. Figure 111.8 lists the first 10 observations in the data set LDL_1.

Figure 111.8 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Trt	Ldl
1	0	33.33
2	1	-14.89
3	0	15.30
4	1	4.71
5	0	26.89
6	1	-48.74
7	0	-39.35
8	1	-8.13
9	0	-8.22
10	1	12.35

The variable Trt is an indicator variable with value 1 for patients in the treatment group and value 0 for patients in the placebo control group. The variable Ldl is the LDL level of these patients.

The following statements use the REG procedure to estimate the mean treatment difference and its associated standard error at stage 1:

```
proc reg data=LDL_1;
   model Ldl=Trt;
   ods output ParameterEstimates=Parms_LDL1;
run;
```

The following statements create the data set for the mean treatment difference and its associated standard error as a PARMS= data set, which will subsequently serve as an input data set for PROC SEQTEST. Note that all of the variables are required for a PARMS= data set, as described in the section "PARMS= Data Set" on page 9356.

```
data Parms_LDL1;
   set Parms_LDL1;
   if Variable='Trt';
   _Scale_='MLE';
   _Stage_= 1;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;

proc print data=Parms_LDL1;
   title 'Statistics Computed at Stage 1';
run;
```

Figure 111.9 displays the statistics computed at stage 1.

Figure 111.9 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	Trt	-2.52591	5.68572	MLE	1

Since the sample sizes derived are based on the estimated variance at the designing phase, the information level that corresponds to the test statistic at stage 1 is estimated by

$$I_1 = \frac{1}{s_1^2} = \frac{1}{5.686^2} = 0.0309$$

where s_1 is the standard error of the treatment estimate.

The following statements invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_LDL
             Parms(Testvar=Trt)=Parms_LDL1
             infoadj=prop
             ;
   ods output Test=Test_LDL1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The PARMS=PARMS_LDL1 option specifies the input data set PARMS_LDL1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=TRT option identifies the test variable TRT in the data set.

By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the maximum information and the Type I error level are maintained. Furthermore, with the INFOADJ=PROP option (which is the default), the information levels at future interim stages (2 and 3) are adjusted proportionally from the levels provided in the BOUNDARY= data set.

The ODS OUTPUT statement with the TEST=TEST_LDL1 option creates an output data set named TEST_LDL1 which contains the updated boundary information for the test at stage 1, and the boundary information that is needed for the group sequential test at the next stage. See the section "Boundary Adjustments for Information Levels" on page 9361 for details.

The "Design Information" table in Figure 111.10 displays design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary values are adjusted for the updated information levels to maintain the Type I α level, and the maximum information remains the same as in the BOUNDARY= data set. But the derived Type II error probability β and power $1 - \beta$ are slightly different with new information levels. With the updated power $1 - \beta$, the corresponding fixed-sample design is also updated.

Figure 111.10 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_LDL
Data Set	WORK.PARMS_LDL1
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Number of Stages	4
Alpha	0.05
Beta	0.10074
Power	0.89926
Max Information (Percent of Fixed Sample)	102.4815
Max Information	0.10740291
Null Ref ASN (Percent of Fixed Sample)	101.7765
Alt Ref ASN (Percent of Fixed Sample)	75.4928

The "Test Information" table in Figure 111.11 displays the boundary values for the test statistic. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), these statistics are displayed with the standardized Z scale. With the INFOADJ=PROP option (which is the default), information levels at future interim stages are derived proportionally from the corresponding levels provided in the BOUNDARY= data set.

Figure 111.11 Sequential Tests

Test Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values		Test Trt	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Upper Alpha	Estimate	Action
1	0.2880	0.030934	-1.75879	1.75879	-3.39532	3.39532	-0.44426	Continue
2	0.5253	0.056423	-2.37536	2.37536	-2.77374	2.77374	.	
3	0.7627	0.081913	-2.86205	2.86205	-2.32412	2.32412	.	
4	1.0000	0.107403	-3.27724	3.27724	-2.03147	2.03147	.	

At stage 1, the standardized Z statistic –0.44426 is between the lower and upper α boundary values, and so the trial continues to the next stage. With the observed information level at stage 1, $I_1 = 0.0309$ (which is not substantially different from the target information level at stage 1), the trial continues to the next stage without adjustment of the sample size according to the study plan.

If an observed information level is different from its target level at an interim stage, the sample sizes at future stages can be adjusted to achieve the target maximum information level according to the study plan. That is, a study plan might modify the final sample size to achieve the target maximum information level if the observed information level is different from its target level by a specified amount at the interim stage. For example, if the variance estimate is used to compute the required sample size of a two-sample Z test for mean difference, the study plan might use the current variance estimate to update the required sample size for the trial (Jennison and Turnbull 2000, p. 295).

See the section "Test for the Difference between Two Normal Means" on page 9196 in Chapter 110, "The SEQDESIGN Procedure," for more information about how to compute the sample size from the variance estimate.

If ODS Graphics is enabled, a detailed test plot with the rejection and acceptance regions is displayed, as shown in Figure 111.12. This plot displays the boundary values in the "Test Information" table in Figure 111.11. The stages are indicated by vertical lines with accompanying stage numbers. The horizontal axis indicates the information levels for the stages. As expected, the test statistic is in the continuation region between the lower and upper α boundaries.

Figure 111.12 Sequential Test Plot

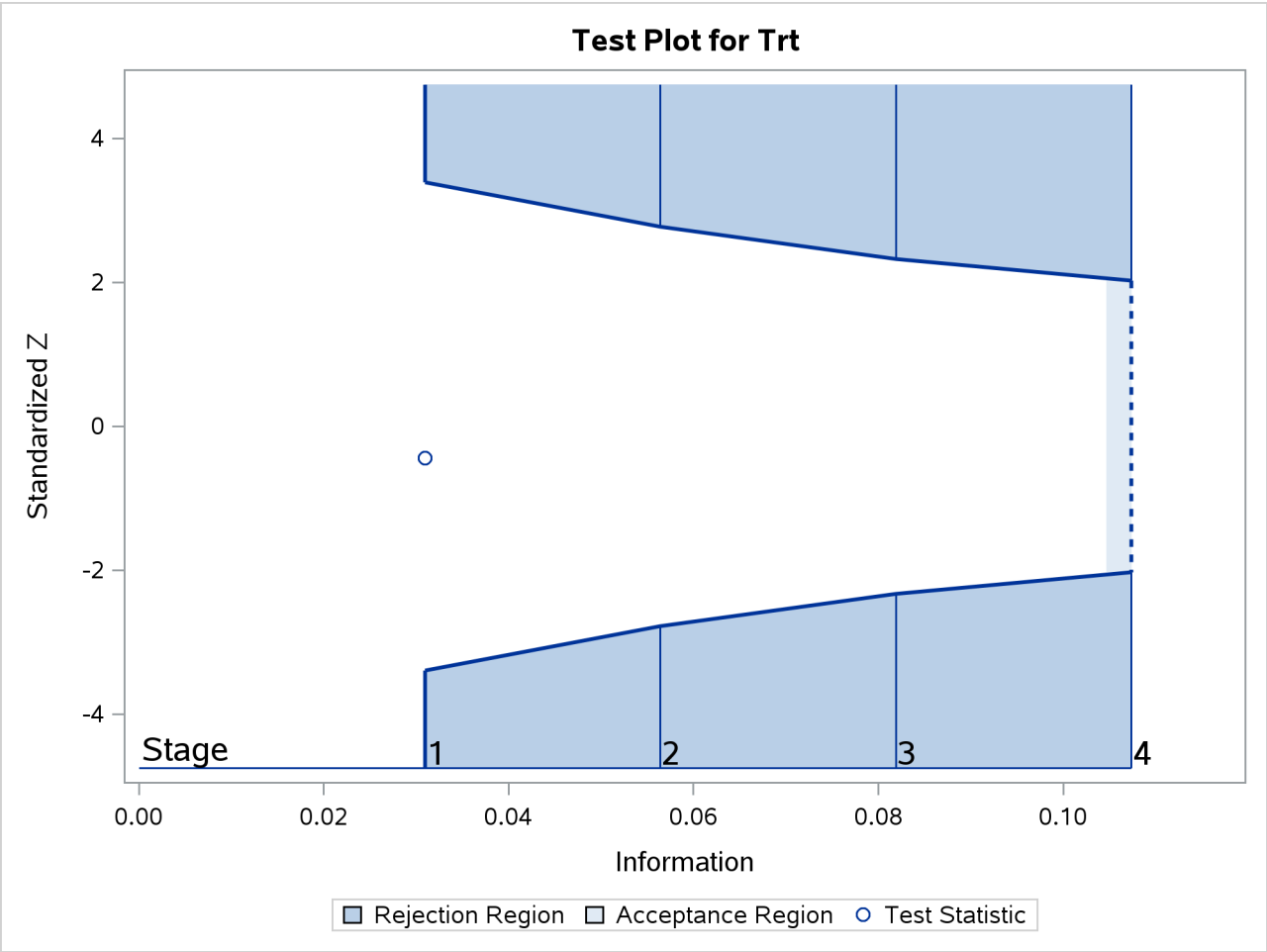

The following statements use the REG procedure with the data available at the first two stages to estimate the mean treatment difference and its associated standard error at stage 2:

```
proc reg data=LDL_2;
   model Ldl=Trt;
   ods output ParameterEstimates=Parms_LDL2;
run;
```

The following statements create and display (in Figure 111.13) the data set for the mean treatment difference and its associated standard error:

```
data Parms_LDL2;
   set Parms_LDL2;
   if Variable='Trt';
   _Scale_='MLE';
   _Stage_= 2;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;

proc print data=Parms_LDL2;
   title 'Statistics Computed at Stage 2';
run;
```

Figure 111.13 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	Trt	-8.37628	4.24405	MLE	2

Using the standard error for the treatment estimate available at stage 2, the information level that corresponds to the test statistic at stage 2 is estimated by

$$I_2 = \frac{1}{s_2^2} = \frac{1}{4.244^2} = 0.0555$$

where s_2 is the standard error of the treatment estimate at stage 2.

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2:

```
proc seqtest Boundary=Test_LDL1
             Parms(Testvar=Trt)=Parms_LDL2
             infoadj=prop
             ;
   ods output Test=Test_LDL2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=TEST_LDL2 option creates an output data set named TEST_LDL2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Test Information" table in Figure 111.14 displays the boundary values for the test statistic with the default standardized Z scale

Figure 111.14 Sequential Tests

The SEQTEST Procedure

Test Information (Standardized Z Scale) Null Reference = 0								
	Information Level		Alternative Reference		Boundary Values		Test Trt	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Upper Alpha	Estimate	Action
1	0.2880	0.030934	-1.75879	1.75879	-3.39532	3.39532	-0.44426	Continue
2	0.5169	0.055519	-2.35624	2.35624	-2.78456	2.78456	-1.97365	Continue
3	0.7585	0.081461	-2.85413	2.85413	-2.32908	2.32908	.	
4	1.0000	0.107403	-3.27724	3.27724	-2.03097	2.03097	.	

At stage 2, the standardized test statistic, $z = -8.37628/4.24405 = -1.97365$, is between its corresponding lower and upper α boundary values. Therefore, the trial continues to the next stage.

The following statements use the REG procedure with the data available at the first three stages to estimate the mean treatment difference and its associated standard error at stage 3:

```
proc reg data=LDL_3;
   model Ldl=Trt;
   ods output ParameterEstimates=Parms_LDL3;
run;
```

The following statements create and display (in Figure 111.15) the data set for the mean treatment difference and its associated standard error:

```
data Parms_LDL3;
   set Parms_LDL3;
   if Variable='Trt';
   _Scale_='MLE';
   _Stage_= 3;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;

proc print data=Parms_LDL3;
   title 'Statistics Computed at Stage 3';
run;
```

Figure 111.15 Statistics Computed at Stage 3

Statistics Computed at Stage 3

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	Trt	-9.21369	3.42149	MLE	3

The following statements invoke the SEQTEST procedure to test for early stopping at stage 3:

```
ods graphics on;
proc seqtest Boundary=Test_LDL2
             Parms(Testvar=Trt)=Parms_LDL3
             infoadj=prop
```

```
                ;
    ods output Test=Test_LDL3;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 3, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 3, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=TEST_LDL3 option creates an output data set named TEST_LDL3 which contains the updated boundary information for the test at stage 3. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Test Information" table in Figure 111.16 displays the boundary values for the test statistic with the default standardized Z scale.

Figure 111.16 Sequential Tests

The SEQTEST Procedure

Test Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values		Test	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Upper Alpha	Trt Estimate	Action
1	0.2880	0.030934	-1.75879	1.75879	-3.39532	3.39532	-0.44426	Continue
2	0.5169	0.055519	-2.35624	2.35624	-2.78456	2.78456	-1.97365	Continue
3	0.7953	0.085422	-2.92271	2.92271	-2.25480	2.25480	-2.69289	Reject Null
4	1.0000	0.107403	-3.27724	3.27724	-2.04573	2.04573	.	

The sequential test stops at stage 3 to reject the null hypothesis for the lower alternative because the test statistic –2.69289 is less than the corresponding upper α boundary –2.25480. That is, the test demonstrates significant beneficial effect for the new drug.

The "Test Plot" displays boundary values for the design and the test statistic at the first three stages, as shown in Figure 111.17. It shows that the test statistic is in the "Rejection Region" below the lower α boundary at stage 3.

Figure 111.17 Sequential Test Plot

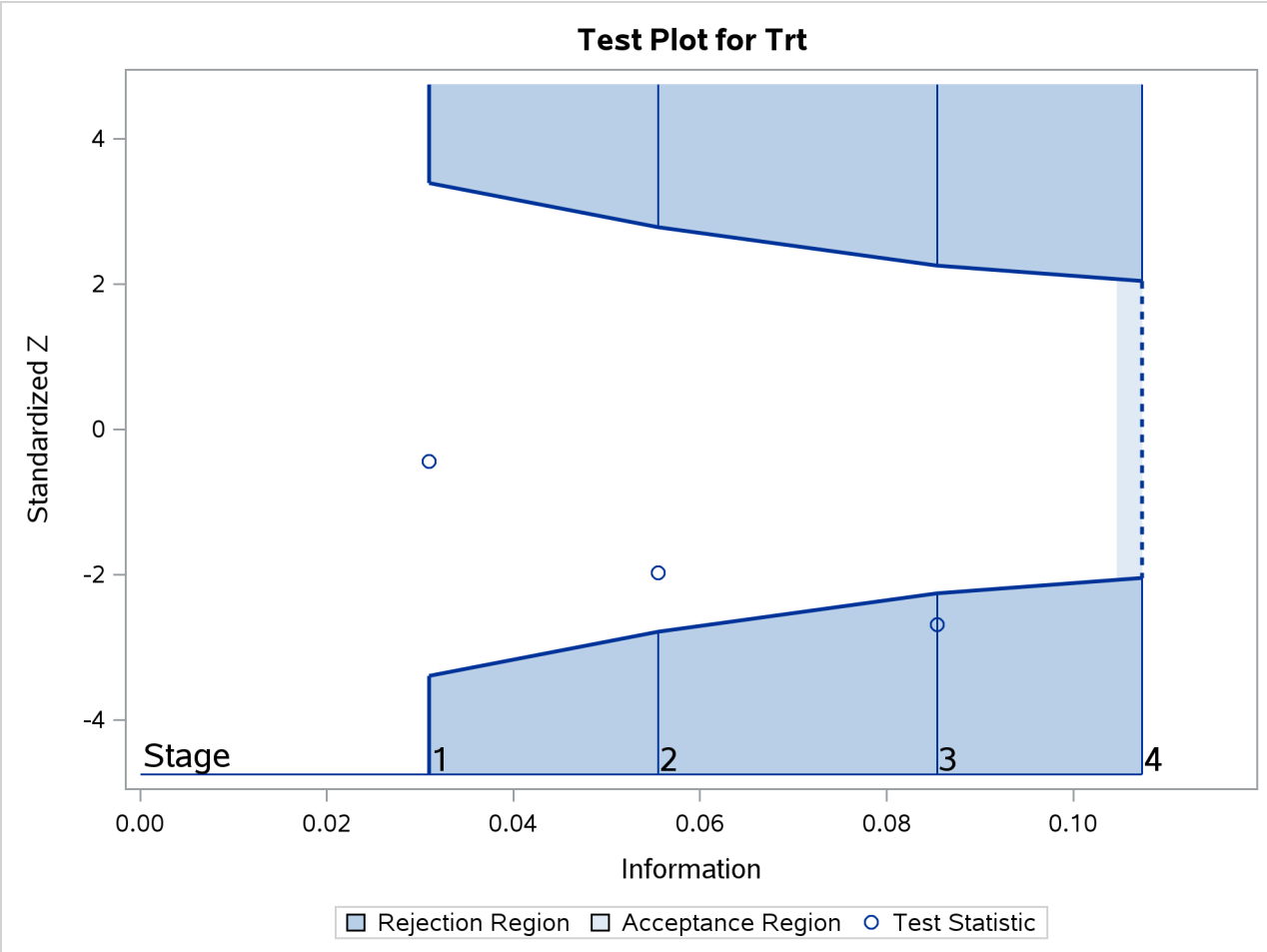

When a trial stops, the "Parameter Estimates" table in Figure 111.18 displays the stopping stage, parameter estimate, unbiased median estimate, confidence limits, and p-value under the null hypothesis $H_0 : \theta = 0$. As expected, the p-value 0.0108 is significant at the two-sided α level, $\alpha = 0.05$, and the confidence interval does not contain the value zero. The p-value, unbiased median estimate, and confidence limits depend on the ordering of the sample space (k, z), where k is the stage number and z is the standardized Z statistic. See the section "Analysis after a Sequential Test" on page 9368 for a detailed description of these statistics.

Figure 111.18 Parameter Estimates

Parameter Estimates
Stagewise Ordering

Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	95% Confidence Limits	
Trt	3	-9.213692	0.0108	-9.022891	-15.79845	-2.13138

Syntax: SEQTEST Procedure

The following statement is available in the SEQTEST procedure:

PROC SEQTEST < *options* > **;**

PROC SEQTEST Statement

The PROC SEQTEST statement invokes the SEQTEST procedure. Table 111.1 summarizes the *options* available in the PROC SEQTEST statement.

Table 111.1 Summary of PROC SEQTEST Options

Option	Description
Input Data Sets	
BOUNDARY=	Specifies the data set for boundary information
DATA=	Specifies the data set for parameter estimates and information levels
PARMS=	Specifies the data set for parameter estimates and standard errors
Boundaries	
BETABOUNDARY=	Specifies whether the β boundary is used in the computation of the Type I error level α
BETAOVERLAP=	Checks for overlapping of the lower and upper β boundaries at the current and subsequent interim stages in a two-sided design
BOUNDARYKEY=	Specifies the boundary key to maintain Type I and II error probability levels
BOUNDARYSCALE=	Specifies the boundary scale
ERRSPENDADJ=	Specifies error spending methods for boundary adjustments
ERRSPENDMIN=	Specifies minimum error spending values for the boundaries
INFOADJ=	Specifies whether information levels at future interim stages should be adjusted
NSTAGES=	Specifies the number of stages
Test Statistics	
DATA(INFOVAR=)	Specifies the information variable in the DATA= data set
DATA(TESTVAR=)	Specifies the test variable in the DATA= data set
PARMS(INFOVAR=)	Specifies the information variable in the PARMS= data set
PARMS(TESTVAR=)	Specifies the test variable in the PARMS= data set
***p*-Values and Confidence Intervals**	
CIALPHA=	Specifies the significance levels for the confidence interval
CITYPE=	Specifies the types of confidence interval
ORDER=	Specifies the ordering of the sample space used to derive the *p*-values and confidence limits
Table Output	
CONDPOWER	Displays conditional powers
ERRSPEND	Displays the cumulative error spending at each stage

Table 111.1 *continued*

Option	Description
PREDPOWER	Displays the predictive powers
PSS	Displays the powers and expected sample sizes
RCI	Displays the repeated confidence intervals
STOPPROB	Displays the expected cumulative stopping probabilities
Graphics Output	
PLOTS=ASN	Displays the expected sample numbers plot
PLOTS=CONDPOWER	Displays the conditional powers plot
PLOTS=ERRSPEND	Displays the error spending plot
PLOTS=POWER	Displays the powers plot
PLOTS=RCI	Displays the repeated confidence intervals plot
PLOTS=TEST	Displays the boundary plot with test statistics

The BOUNDARY= option provides the information for the design and is required in the PROC SEQTEST statement. By default, the SEQTEST procedure displays tables of design information and test information. If ODS Graphics is enabled, the procedure also displays a sequential test plot.

The following *options* can be used in the PROC SEQTEST statement. They are listed in alphabetical order.

BETABOUNDARY=BINDING | NONBINDING

specifies whether the β boundary is used in the computation of the Type I error level α. The BETABOUNDARY=BINDING option computes the Type I error probability with the β (acceptance) boundary, and the BETABOUNDARY=NONBINDING suboption computes the Type I error probability without the β boundary (Zhu, Ni, and Yao 2011, pp. 132–133). For a detailed description of nonbinding acceptance boundary, see the section "Acceptance ($\boldsymbol{\beta}$) Boundary" on page 9179 in Chapter 110, "The SEQDESIGN Procedure." The default is BETABOUNDARY=BINDING.

BETAOVERLAP=ADJUST | NOADJUST

OVERLAP=ADJUST | NOADJUST

specifies whether to check for overlapping of the lower and upper β boundaries for the two corresponding one-sided tests at the current and subsequent interim stages. This option applies to two-sided designs with early stopping to accept H_0, or to either accept or reject H_0. This type of overlapping might result from a small β spending at an interim stage. When you specify BETAOVERLAP=ADJUST, the procedure checks for this type of overlapping at the current and subsequent interim stages. If such overlapping is found, the β boundaries for the two-sided design at that stage are set to missing, and the β spending values at subsequent stages are adjusted, as described in the section "Boundary Adjustments for Overlapping Lower and Upper $\boldsymbol{\beta}$ Boundaries" on page 9364.

You can specify BETAOVERLAP=NOADJUST to request that no adjustment be made. The default is BETAOVERLAP=ADJUST.

BOUNDARY=*SAS-data-set*

names the required SAS data set that contains the design boundary information. At stage 1, the data set is usually created from the "Boundary Information" table created by the SEQDESIGN procedure. At each subsequent stage, the data set is usually created from the "Test Information" table created by the SEQTEST procedure at the previous stage. The data set includes the variables _Scale_ for the boundary scale, _Stop_ for the stopping criterion, and _ALT_ for the type of alternative hypothesis. It also includes _Stage_ for the stage number, Info_Prop for the information proportion, and a set of the boundary variables from Bound_LA, Bound_LB, Bound_UB, and Bound_UA for boundary values at each stage.

The data set might also include _Info_ for the actual information level, NObs for the number of observation, and Events for the number of events required at each stage.

BOUNDARYKEY=ALPHA | BETA | BOTH

specifies the boundary key to be maintained in the boundary adjustments. The BOUNDARYKEY=ALPHA option maintains the Type I α level and derives the Type II error probability, and the BOUNDARYKEY=BETA option maintains the Type II β level and derives the Type I error probability. The BOUNDARYKEY=BOTH option maintains both α and β levels simultaneously by deriving a new maximum information. The default is BOUNDARYKEY=ALPHA.

BOUNDARYSCALE=MLE | SCORE | STDZ | PVALUE

BSCALE=MLE | SCORE | STDZ | PVALUE

specifies the boundary scale to be displayed in the output boundary table and plot. The BOUNDARYSCALE=MLE, BOUNDARYSCALE=SCORE, BOUNDARYSCALE=STDZ, and BOUNDARYSCALE=PVALUE options correspond to the boundary with the maximum likelihood estimator scale, score statistic scale, standardized normal Z scale, and p-value scale, respectively. The default is BOUNDARYSCALE=STDZ.

With the BOUNDARYSCALE=MLE or BOUNDARYSCALE=SCORE option, either the MAXINFO= option must be specified or the _Info_ variable must be in the BOUNDARY= data set to provide the necessary information level at each stage to derive the boundary values. Usually, these values are obtained from analysis output in SAS procedures.

Note that for a two-sided design, the p-value scale displays the one-sided fixed-sample p-value under the null hypothesis with a lower alternative hypothesis.

CIALPHA=α < (< LOWER=α_l > < UPPER=α_u >) >

specifies the significance levels for the confidence interval, where $0 < \alpha < 1$, $0 < \alpha_l < 0.5$, and $0 < \alpha_u < 0.5$. The default is CIALPHA= 0.05.

For a lower confidence interval (CITYPE=LOWER), the CIALPHA=α option produces a $(1 - \alpha)$ lower confidence interval. For an upper confidence interval (CITYPE=UPPER), the CIALPHA=α option produces a $(1 - \alpha)$ upper confidence interval. The LOWER= and UPPER= suboptions are applicable only for a two-sided confidence interval (CITYPE=TWOSIDED). The LOWER= suboption specifies the lower significance level α_l and the upper significance level $\alpha_u = 1 - \alpha_l$. The UPPER= suboption specifies the upper significance level α_u and the lower significance level $\alpha_l = 1 - \alpha_u$. If both LOWER= and UPPER= suboptions are not specified, $\alpha_l = \alpha_u = \alpha/2$. The significance levels α_l and α_u are then used for the $(1 - \alpha_l)$ lower confidence limit and $(1 - \alpha_u)$ upper confidence limit, respectively.

CITYPE=LOWER | UPPER | TWOSIDED

specifies the type of confidence interval. The CITYPE=LOWER, CITYPE=UPPER, and CITYPE=TWOSIDED options correspond to the lower confidence interval, upper confidence interval, and two-sided confidence interval, respectively. The default is CITYPE=LOWER for the design with an upper alternative, CITYPE=UPPER for the design with a lower alternative, and CITYPE=TWOSIDED for the design with a two-sided alternative.

DATA < (TESTVAR=*variable* **| INFOVAR=***variable* **) >=***SAS-data-set*

names the SAS data set that contains the test statistic and its associated information level for the stage. The *SAS-data-set* includes the stage variable _Stage_. You can also specify the following options within parentheses after the DATA keyword:

TESTVAR=*variable*

identifies the test variable in the *SAS-data-set*. If you specify this option, the data set also includes the scale variable _Scale_ for the test statistic. Usually, these test variable values are obtained from analysis output in SAS procedures.

INFOVAR=*variable*

identifies the information variable in the *SAS-data-set* to either identify or derive the information levels. If you specify this option, the following information variables apply:

- _Info_ uses _Info_ for the information levels.
- Events uses Events for the numbers of events to derive the information levels.
- NObs uses NObs for the numbers of observations to derive the information levels.
- None uses the information levels in the BOUNDARY= data set.

If you do not specify the INFOVAR= option, PROC SEQTEST uses _Info_ (if it exists) in the *SAS-data-set* for the information levels. Otherwise, the procedure uses NObs to derive information levels from the numbers of observation or uses Events to derive information levels from the numbers of events. If these variables are not in the *SAS-data-set*, the information levels in the BOUNDARY= data set are used. See Example 111.4 for an illustration of the INFOVAR=NObs option.

ERRSPENDADJ=*method*

ERRSPENDADJ(*boundary***)=***method*

BOUNDARYADJ=*method*

BOUNDARYADJ(*boundary***)=***method*

specifies methods to compute the error spending values at the current and future interim stages for the boundaries. This option is applicable only if the observed information level at the current stage does not match the value provided in the BOUNDARY= data set. These error spending values are then used to derive the updated boundary values. The default is ERRSPENDADJ=ERRLINE. Note that the information levels at future interim stages are determined by the INFOADJ= option.

The following options specify available error spending methods for boundary adjustment:

NONE
specifies that the cumulative error spending at each interim stage not be changed, even if the corresponding information level has been changed.

ERRLINE
specifies the linear interpolation method for the adjustment.

ERRFUNCGAMMA < (GAMMA= γ **) >**
specifies the gamma function method for the adjustment. The GAMMA= suboption specifies the γ parameter in the function, where $\gamma \leq 3$. The default is GAMMA=–2.

ERRFUNCOBF
specifies the approximate O'Brien-Fleming cumulative error spending function for the adjustment.

ERRFUNCPOC
specifies the approximate Pocock cumulative error spending function for the adjustment.

ERRFUNCPOW < (RHO= ρ **) >**
specifies the power function method for the adjustment. The RHO= suboption specifies the power parameter ρ in the function, where $\rho \geq 0.25$. The default is RHO=2.

See the section "Boundary Adjustments for Information Levels" on page 9361 for a detailed description of the available error spending methods for boundary adjustment in the SEQTEST procedure.

If an error spending method for boundary adjustments is used for all boundaries in a group sequential test, you can use the ERRSPENDADJ=*method* option to specify the method. Otherwise, you can use the following ERRSPENDADJ(*boundary*)=*method* options to specify different methods for the boundaries.

ERRSPENDADJ(ALPHA)=*method*
ERRSPENDADJ(REJECT)=*method*
BOUNDARYADJ(ALPHA)=*method*
BOUNDARYADJ(REJECT)=*method*
specifies the adjustment method for the α (rejection) boundary of a one-sided design or the lower and upper α boundaries of a two-sided design.

ERRSPENDADJ(LOWERALPHA)=*method*
ERRSPENDADJ(LOWERREJECT)=*method*
BOUNDARYADJ(LOWERALPHA)=*method*
BOUNDARYADJ(LOWERREJECT)=*method*
specifies the adjustment method for the lower α boundary of a two-sided design.

ERRSPENDADJ(UPPERALPHA)=*method*
ERRSPENDADJ(UPPERREJECT)=*method*
BOUNDARYADJ(UPPERALPHA)=*method*
BOUNDARYADJ(UPPERREJECT)=*method*
specifies the adjustment method for the upper α boundary of a two-sided design.

ERRSPENDADJ(BETA)=*method*

ERRSPENDADJ(ACCEPT)=*method*

BOUNDARYADJ(BETA)=*method*

BOUNDARYADJ(ACCEPT)=*method*

specifies the adjustment method for the β (acceptance) boundary of a one-sided design or the lower and upper β boundaries of a two-sided design.

ERRSPENDADJ(LOWERBETA)=*method*

ERRSPENDADJ(LOWERACCEPT)=*method*

BOUNDARYADJ(LOWERBETA)=*method*

BOUNDARYADJ(LOWERACCEPT)=*method*

specifies the adjustment method for the lower β boundary of a two-sided design.

ERRSPENDADJ(UPPERBETA)=*method*

ERRSPENDADJ(UPPERACCEPT)=*method*

BOUNDARYADJ(UPPERBETA)=*method*

BOUNDARYADJ(UPPERACCEPT)=*method*

specifies the adjustment method for the upper β boundary of a two-sided design.

ERRSPENDMIN=*numbers*

ERRSPENDMIN(*boundary***)=***numbers*

specifies the minimum error spending values at the current observed and future interim stages for the boundaries specified in the BOUNDARYKEY= option. The default is ERRSPENDMIN=0.

If a set of numbers is used for each boundary in the design, you can use the ERRSPENDMIN=*numbers* option. Otherwise, you can use the following ERRSPENDMIN(*boundary*)=*numbers* options to specify different sets of minimum error spending values for the boundaries. For a boundary, the error spending value at stage 1 is identical to its nominal p-value.

ERRSPENDMIN(ALPHA)=*numbers*

ERRSPENDMIN(REJECT)=*numbers*

specifies the minimum error spending values for the α boundary of a one-sided design or the lower and upper α boundaries of a two-sided design.

ERRSPENDMIN(LOWERALPHA)=*numbers*

ERRSPENDMIN(LOWERREJECT)=*numbers*

specifies the minimum error spending values for the lower α boundary of a two-sided design.

ERRSPENDMIN(UPPERALPHA)=*numbers*

ERRSPENDMIN(UPPERREJECT)=*numbers*

specifies the minimum error spending values for the upper α boundary of a two-sided design.

ERRSPENDMIN(BETA)=*numbers*

ERRSPENDMIN(ACCEPT)=*numbers*

specifies the minimum error spending values for the β boundary of a one-sided design or the lower and upper β boundaries of a two-sided design.

ERRSPENDMIN(LOWERBETA)=*numbers*

ERRSPENDMIN(LOWERACCEPT)=*numbers*

specifies the minimum error spending values for the lower β boundary of a two-sided design.

ERRSPENDMIN(UPPERBETA)=*numbers*

ERRSPENDMIN(UPPERACCEPT)=*numbers*

specifies the minimum error spending values for the upper β boundary of a two-sided design.

INFOADJ=NONE | PROP

specifies whether information levels at future interim stages are to be adjusted. If you specify INFOADJ=NONE, no adjustment is made, and the information levels are preserved at the levels provided in the BOUNDARY= data set. If you specify INFOADJ=PROP (which is the default), the information levels are adjusted proportionally from the levels provided in the BOUNDARY= data set. The section "Information Level Adjustments at Future Stages" on page 9360 describes how the adjustments are computed.

Note that if you specify BOUNDARYKEY=BOTH, the INFOADJ=NONE option is not applicable, and the INFOADJ=PROP option is used to adjusted the information levels at future stages proportionally from the levels provided in the BOUNDARY= data set to maintain both α and β levels.

NSTAGES=*number*

specifies the number of stages for the clinical trial. The default is the number derived from the BOUNDARY= data set.

The specified NSTAGES= number might or might not be the same as the number derived in the BOUNDARY= data set. You can use the NSTAGES= option to set the next stage as the final stage to compute the conditional power, as described in the section "Conditional Power Approach" on page 9365.

ORDER=LR | MLE | STAGEWISE

specifies the ordering of the sample space (k, z), where k is the stage number and z is the observed standardized Z statistic. The ordering is used to derive the p-values for the observed (k, z) statistic and to create unbiased median estimate and confidence limits from the statistic. The ORDER=LR option specifies the LR ordering that compares the distances between observed standardized Z statistics and their corresponding hypothetical values, the ORDER=MLE option specifies the MLE ordering that compares values in the MLE scale, and the ORDER=STAGEWISE specifies the stagewise ordering that uses counterclockwise ordering around the continuation region. The default is ORDER=STAGEWISE. See the section "Available Sample Space Orderings in a Sequential Test" on page 9369 for a detailed description of these sample space orderings.

PARMS < (TESTVAR=*variable* **| INFOVAR=***variable* **) >=***SAS-data-set*

names the SAS data set that contains the parameter estimate and its associated standard error for the stage. The *SAS-data-set* includes the stage variable _Stage_. You can also specify the following *options* within parentheses after the PARMS keyword:

TESTVAR=*variable*
identifies the test variable in the *SAS-data-set*. If you specify this option, the *SAS-data-set* also includes the Parameter, Effect, Variable, or Parm variable that contains the *variable*. In addition, the *SAS-data-set* also includes the test statistic Estimate, the standard error of the estimate StdErr, and the test statistic scale variable _Scale_. Usually, these test variable values are obtained from analysis output in SAS procedures.

INFOVAR=*variable*
identifies the information variable in the *SAS-data-set* to either identify or derive the information levels. If you specify this option, the following information variables apply:

- Events uses Events for the numbers of events to derive the information levels.
- NObs uses NObs for the numbers of observations to derive the information levels.
- None uses the information levels in the BOUNDARY= data set.
- StdErr uses StdErr to derive the information levels.

If you do not specify the INFOVAR= option, PROC SEQTEST uses StdErr (if it exists) in the *SAS-data-set* to derive the information levels. Otherwise, the procedure uses NObs to derive information levels from the numbers of observation or uses Events to derive information levels from the numbers of events. If these variables are not in the *SAS-data-set*, the information levels in the BOUNDARY= data set are used.

Table Output Options

You can specify the following *options* in the PROC SEQTEST statement to display additional table output. They are listed in alphabetical order.

CONDPOWER < (*options*) >
displays conditional powers given the most recently observed statistic under specified hypothetical references.

You can specify the following *options*:

CREF= $c_1 < c_2 \ldots >$
specifies the hypothetical references, where $c_i \geq 0$. For a one-sided test, the powers are derived under the hypothetical references $\theta = \hat{\theta}$ and $\theta = c_i\theta_1$, where $\hat{\theta}$ is the observed statistic, θ_1 is the alternative reference, and c_i are the values specified in the CREF= option. For a two-sided test, the powers are derived under hypothetical references $\theta = \hat{\theta}$, $\theta = c_i\theta_{1l}$, and $\theta = c_i\theta_{1u}$, where θ_{1l} is the lower alternative reference and θ_{1u} is the upper alternative reference. The default is CREF= 0 0.5 1.0 1.5.

TYPE=ALLSTAGES | FINALSTAGE
specifies the probability to be computed for the conditional power. Two types of conditional power can be computed: TYPE=ALLSTAGES computes the conditional power as the total probability of rejecting the null hypothesis at all future stages given the observed statistic, and TYPE=FINALSTAGE computes the conditional power as the probability that the test statistic at the final stage would exceed the rejection critical value given the observed statistic. The default is TYPE=ALLSTAGES.

If interim stages exist between the current stage and the final stage, the conditional power that is computed when TYPE=FINALSTAGE is not the conditional probability to reject the null

hypothesis H_0. In this case, you can set the next stage as the final stage, and the resulting conditional power is the conditional probability to reject H_0.

ERRSPEND

displays the error spending at each stage for each sequential boundary.

PREDPOWER

displays predictive powers given the most recently observed statistic. The predictive power is the posterior probability that the test statistic at the final stage would exceed the rejection critical value given the observed statistic and a prior distribution of the hypothetical reference. A noninformative prior is used in the procedure.

PSS < (CREF= $c_1 < c_2 \ldots >$ **) >**

displays powers and expected sample sizes under various hypothetical references, where $c_i \geq 0$.

For a one-sided design with the null reference $\theta_0 = 0$, the power and expected sample sizes under hypotheses $\theta = c_i\, \theta_1$ are displayed, where θ_1 is the alternative reference and c_i are the values specified in the CREF= option.

For a two-sided design, the power and expected sample sizes under hypotheses $\theta = c_i \theta_{1l}$ and $\theta = c_i \theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. The default is CREF= 0 0.5 1.0 1.5.

Note that for a symmetric two-sided design, only the power and expected sample sizes under hypotheses $\theta = c_i\, \theta_{1u}$ are derived.

RCI

displays repeated confidence intervals for the parameter from the observed statistic at each stage. Repeated confidence intervals include both rejection and acceptance confidence intervals.

With the STOP=REJECT or STOP=BOTH option, rejection confidence limits can be derived, and the null hypothesis $H_0 : \theta = 0$ is rejected if the lower rejection confidence limit is greater than 0 or the upper rejection confidence limit is less than 0.

With the STOP=ACCEPT or STOP=BOTH option, acceptance confidence limits can be derived, and the null hypothesis is accepted with alternative hypotheses $H_{1l} : \theta = \theta_{1l}$ and $H_{1u} : \theta = \theta_{1u}$ if the upper acceptance confidence limit is less than θ_{1u} and the lower acceptance confidence limit is greater than θ_{1l}.

STOPPROB < (CREF= $c_1 < c_2 \ldots >$ **) >**

displays expected cumulative stopping probabilities under various hypothetical references, where $c_i \geq 0$.

For a one-sided design, expected cumulative stopping probabilities at each stage under hypotheses $\theta = c_i\, \theta_1$ are displayed, where θ_1 is the alternative reference and c_i are the values specified in the CREF= option.

For a two-sided design, expected cumulative stopping probabilities at each stage under hypotheses $\theta = c_i \theta_{1l}$ and $\theta = c_i \theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. Note that for a symmetric two-sided design, only expected cumulative stopping probabilities under hypotheses $\theta = c_i\, \theta_{1u}$ are derived. The default is CREF= 0 0.5 1.0 1.5.

Graphics Output Options

The following *options* can be used in the PROC SEQTEST statement to display plots with ODS Graphics. They are listed in alphabetical order.

PLOTS < (ONLY) > < = *plot-request* >

PLOTS < (ONLY) > < = (*plot-request* < . . . *plot-request* >) >

specifies options that control the details of the plots. The default is PLOTS=TEST. The global plot option ONLY suppresses the default plots and displays only plots specifically requested.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc seqtest Boundary=Bnd_LDL
             Parms(Testvar=Trt)=Parms_LDL1
             Plots=(test errspend);
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The *plot-request-options* include the following.

ALL

produces all appropriate plots.

ASN < (CREF= $c_1 < c_2 \ldots >$ **) >**

displays a plot of the average sample numbers (expected sample sizes for nonsurvival data or expected number of events for survival data) under various hypothetical references, where $c_i \geq 0$.

For a one-sided design, expected sample numbers under hypotheses $\theta = c_i\,\theta_1$ are displayed, where θ_1 is the alternative reference and c_i are the values specified in the CREF= option.

For a two-sided design, expected sample numbers under hypotheses $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. Note that for a symmetric two-sided design, only the average sample numbers under hypotheses $\theta = c_i\,\theta_{1u}$ are derived. The default is CREF= 0 to 1.5 by 0.01.

CONDPOWER < (*options*) >

displays a plot of conditional powers given the most recently observed statistic under specified hypothetical references.

You can specify the following *options*:

CREF= $c_1 < c_2 \ldots >$

specifies the hypothetical references, where $c_i \geq 0$. For a one-sided test, the powers are derived under hypothetical references $\theta = \hat{\theta}$ and $\theta = c_i\theta_1$, where $\hat{\theta}$ is the observed statistic, θ_1 is the alternative reference, and c_i are the values specified in the CREF= option. For a two-sided test, the powers are derived under hypothetical references $\theta = \hat{\theta}$, $\theta = c_i\theta_{1l}$, and $\theta = c_i\theta_{1u}$, where θ_{1l} is the lower alternative reference and θ_{1u} is the upper alternative reference. The default is CREF= 0 to 1.5 by 0.01.

TYPE=ALLSTAGES | FINALSTAGE

specifies the probability to be computed for the conditional power. Two types of conditional power can be computed: TYPE=ALLSTAGES computes the conditional power as the total probability of rejecting the null hypothesis at all future stages given the observed statistic, and TYPE=FINALSTAGE computes the conditional power as the probability that the test statistic at the final stage would exceed the rejection critical value given the observed statistic. The default is TYPE=ALLSTAGES.

ERRSPEND < (HSCALE=INFO | STAGE) >

displays a plot of the error spending for all sequential boundaries in the designs simultaneously. You can display the information level (HSCALE=INFO) or the stage number (HSCALE=STAGE) on the horizontal axis. With HSCALE=INFO, the information fractions are used in the plot. The default is HSCALE=STAGE.

NONE

suppresses all plots.

POWER < (CREF= $c_1 < c_2 \ldots >$ **) >**

displays a plot of the power curves under various hypothetical references, where $c_i \geq 0$.

For a one-sided design, powers under hypotheses $\theta = c_i\,\theta_1$ are displayed, where θ_1 is the alternative reference and c_i are the values specified in the CREF= option.

For a two-sided design, powers under hypotheses $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. Note that for a symmetric two-sided design, only powers under hypotheses $\theta = c_i\,\theta_{1u}$ are derived. The default is CREF= 0 to 1.5 by 0.01.

RCI

displays a plot of repeated confidence intervals. Repeated confidence intervals include both rejection and acceptance confidence intervals.

With the STOP=REJECT or STOP=BOTH option, rejection confidence limits can be derived and the null hypothesis $H_0 : \theta = 0$ is rejected if the lower rejection confidence limit is greater than 0 or the upper rejection confidence limit is less than 0.

With the STOP=ACCEPT or STOP=BOTH option, acceptance confidence limits can be derived and the null hypothesis is accepted with alternative hypotheses $H_{1l} : \theta = \theta_{1l}$ and $H_{1u} : \theta = \theta_{1u}$ if the upper acceptance confidence limit is less than θ_{1u} and the lower acceptance confidence limit is greater than θ_{1l}.

TEST < (HSCALE=INFO | SAMPLESIZE) >

displays a plot of the sequential boundaries and test variables. Either the information level (HSCALE=INFO) or the sample size (HSCALE=SAMPLESIZE) is displayed on the horizontal axis. The HSCALE=SAMPLESIZE option is applicable only if the sample size information is available in both the input BOUNDARY= data set and input DATA= data set. The stage number for each stage is displayed inside the plot. The default is HSCALE=INFO.

Details: SEQTEST Procedure

Input Data Sets

The BOUNDARY= data set option is required, and if neither the DATA= nor the PARMS= data set option is specified, the procedure derives statistics such as Type I and Type II error probabilities from the BOUNDARY= data set. The resulting boundaries are displayed with the scale specified in the BOUNDARYSCALE= option.

BOUNDARY= Data Set

The BOUNDARY= data set provides the boundary information for the sequential test. At stage 1, the data set is usually created with an ODS OUTPUT statement from the "Boundary Information" table created by the SEQDESIGN procedure. At each subsequent stage, the data set is usually created with an ODS OUTPUT statement from the "Test Information" table that was created by the SEQTEST procedure at the previous stage. See the section "Getting Started: SEQTEST Procedure" on page 9331 for an illustration of the BOUNDARY= data set option.

The BOUNDARY= data set contains the following variables:

- _Scale_, the boundary scale, with the value MLE for the maximum likelihood estimate, STDZ for the standardized Z, SCORE for the score statistic, or PVALUE for the nominal p-value. Note that for a two-sided design, the nominal p-value is the one-sided fixed-sample p-value under the null hypothesis with a lower alternative hypothesis.
- _Stop_, the stopping criterion, with the value REJECT for rejecting the null hypothesis H_0, ACCEPT for accepting H_0, or BOTH for both rejecting and accepting H_0
- _ALT_, the type of alternative hypothesis, with the value UPPER for an upper alternative, LOWER for a lower alternative, or TWOSIDED for a two-sided alternative
- _Stage_, the stage number
- the boundary variables, a subset of Bound_LA for lower α boundary, Bound_LB for lower β boundary, Bound_UB for upper β boundary, and Bound_UA for upper α boundary
- AltRef_L, the lower alternative reference, if ALT=LOWER or ALT=TWOSIDED
- AltRef_U, the upper alternative reference, if ALT=UPPER or ALT=TWOSIDED
- _InfoProp_, the information proportion at each stage

Optionally, the BOUNDARY= data set also contains the following variables:

- _Info_, the information level at each stage
- NObs, the required number of observations for nonsurvival data at each stage
- Events, the required number of events for survival data at each stage

- Parameter, the variable specified in the DATA(TESTVAR=) or PARMS(TESTVAR=) option
- Estimate, the parameter estimate

If the BOUNDARY= data set contains the variable Parameter for the test variable that is specified in the TESTVAR= option, and the variable Estimate for the test statistics, then these test statistics are also displayed in the output test information table and output test plot.

DATA= Data Set

The DATA= data set provides the test variable information for the current stage of the trial. Such data sets are usually created with an ODS OUTPUT statement by using a procedure such as PROC MEANS. See "Example 111.4: Testing a Binomial Proportion" on page 9416 for an illustration of the DATA= data set option.

The DATA= data set includes the following variables:

- _Stage_, the stage number
- _Scale_, the scale for the test statistic, with the value MLE for the maximum likelihood estimate, STDZ for the standardized Z, SCORE for the score statistic, or PVALUE for the nominal p-value
- _Info_, the information level
- NObs, the number of observations for nonsurvival data at each stage
- Events, the number of events for survival data at each stage
- test variable, specified in the TESTVAR= option, contains the test variable value in the scale specified in the _Scale_ variable

With the specified DATA= data set, PROC SEQTEST derives boundary values from the information levels in the _Info_ variable. If the data set does not include the _Info_ variable, then the information levels are derived from the NObs or Events variable in the DATA= data set if that variable is also in the input BOUNDARY= data set. That is, the information level at stage k is computed as $I_k^* = I_k \times (n_k^*/n_k)$, where I_k and n_k are the information level and sample size, respectively, at stage k in the BOUNDARY= data set and n_k^* is the sample size at stage k in the DATA= data set. Otherwise, the information levels from the BOUNDARY= data set are used.

If the TESTVAR= option is specified, the DATA= data set must also include the test variable for the test statistic and _Scale_ variable for the corresponding scale. Note that for a two-sided design, the nominal p-value is the one-sided fixed-sample p-value under the null hypothesis with a lower alternative hypothesis.

PARMS= Data Set

The PARMS= data set provides a parameter estimate and associated standard error for the current stage of the trial. Such data sets are usually created with an ODS OUTPUT statement by using procedures such as the GENMOD, GLM, LOGISTIC, and REG procedures. See the section "Getting Started: SEQTEST Procedure" on page 9331 for an illustration of the PARMS= data set option.

The PARMS= data set includes the following variables:

- _Stage_, the stage number
- _Scale_, the scale for the test statistic, with the value MLE for the maximum likelihood estimate, STDZ for the standardized Z, SCORE for the score statistic, or PVALUE for the nominal p-value
- Parameter, Effect, Variable, or Parm, which contains the variable specified in the TESTVAR= option
- Estimate, the parameter estimate
- StdErr, standard error of the parameter estimate
- NObs, the number of observations for nonsurvival data at each stage
- Events, the number of events for survival data at each stage

With the specified PARMS= data set, the information level is derived from the StdErr variable. For a score statistic, the information level I_k is the variance of the statistic, $\hat{s}_k^2$, where $\hat{s}_k$ is the standard error in the StdErr variable. Otherwise, the information level is the inverse of the variance of the statistic, $\hat{s}_k^{-2}$. If the data set does not include the StdErr variable, the information levels derived from the BOUNDARY= data set are used.

If the data set does not include the StdErr variable, then the information levels are derived from the NObs or Events variable in the PARMS= data set if that variable is also in the input BOUNDARY= data set. That is, the information level at stage k is computed as $I_k^* = I_k \times (n_k^*/n_k)$, where I_k and n_k are the information level and sample size, respectively, at stage k in the BOUNDARY= data set and n_k^* is the sample size at stage k in the PARMS= data set. Otherwise, the information levels from the BOUNDARY= data set are used.

If the TESTVAR= option is specified, the PARMS= data set also includes the variable Parameter, Effect, Variable, or Parm for the test variable, Estimate for the test statistic, and _Scale_ variable for the corresponding scale. Note that for a two-sided design, the nominal p-value is the one-sided fixed-sample p-value under the null hypothesis with a lower alternative hypothesis.

Boundary Variables

The boundaries created in group sequential trials depend on the type of the alternative hypothesis and the early stopping criterion. Table 111.2 shows the boundaries created with various design specifications.

Table 111.2 Boundary Variables

Specifications		Boundary Variables			
		Lower		Upper	
Alternative Hypothesis	**Early Stopping**	**Alpha**	**Beta**	**Beta**	**Alpha**
Lower	Accept H_0		X		
	Reject H_0	X			
	Accept/Reject H_0	X	X		
Upper	Accept H_0			X	
	Reject H_0				X
	Accept/Reject H_0			X	X
Two-sided	Accept H_0		X	X	
	Reject H_0	X			X
	Accept/Reject H_0	X	X	X	X

Up to four different boundaries can be generated in a group sequential design:

- the upper α boundary, used to reject the null hypothesis in favor of an upper alternative hypothesis
- the upper β boundary, used to accept the null hypothesis with an upper alternative hypothesis
- the lower β boundary, used to accept the null hypothesis with a lower alternative hypothesis
- the lower α boundary, used to reject the null hypothesis in favor of a lower alternative hypothesis

For a two-sided design, the null hypothesis is accepted only if both the null hypothesis is accepted with an upper alternative hypothesis and the null hypothesis is accepted with a lower alternative hypothesis.

For a one-sided design with a lower alternative, only the lower boundaries are created. Similarly, for a one-sided design with an upper alternative, only the upper boundaries are created. For example, Figure 111.19 shows the boundary plot for a one-sided test with an upper alternative.

Figure 111.19 Boundary Plot for One-Sided Test

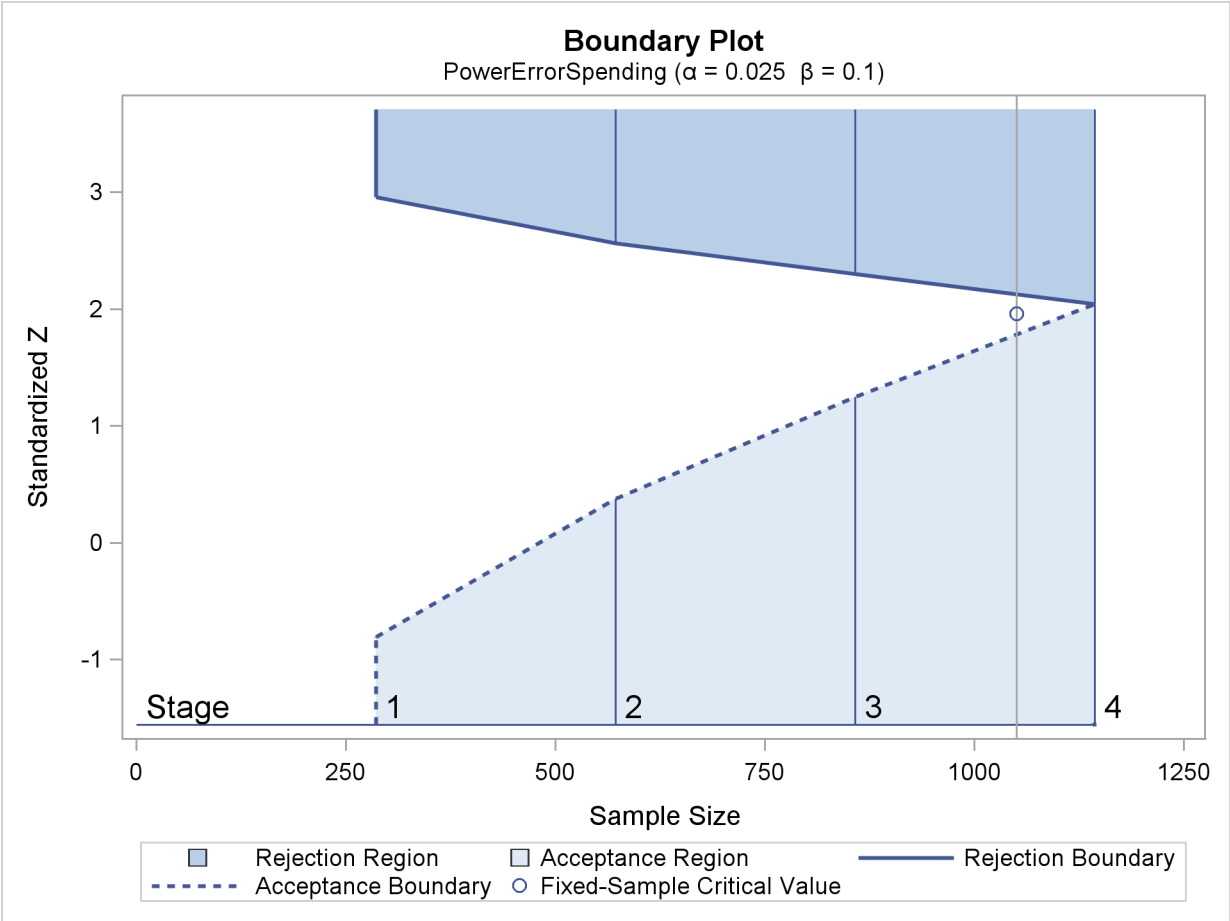

Figure 111.19 corresponds to a one-sided sequential design with early stopping to reject or accept the null hypothesis. For a sequential test with early stopping only to reject the null hypothesis, there are no acceptance boundary values at interim stages. The acceptance boundary value and its associated acceptance region are displayed only at the final stage. Similarly, for a sequential test with early stopping only to accept the null hypothesis, there are no rejection boundary values at interim stages. The rejection boundary value and its associated rejection region are displayed only at the final stage.

For a two-sided design, both the lower and upper boundaries are created. For a design with early stopping to reject the null hypothesis, α boundaries are created. Similarly, for a design with early stopping to accept the null hypothesis, β boundaries are created. For a design with early stopping to accept or reject the null hypothesis, both the α and β boundaries are created.

For example, Figure 111.20 shows the boundary plot for a two-sided test.

Figure 111.20 Boundary Plot for Two-Sided Test

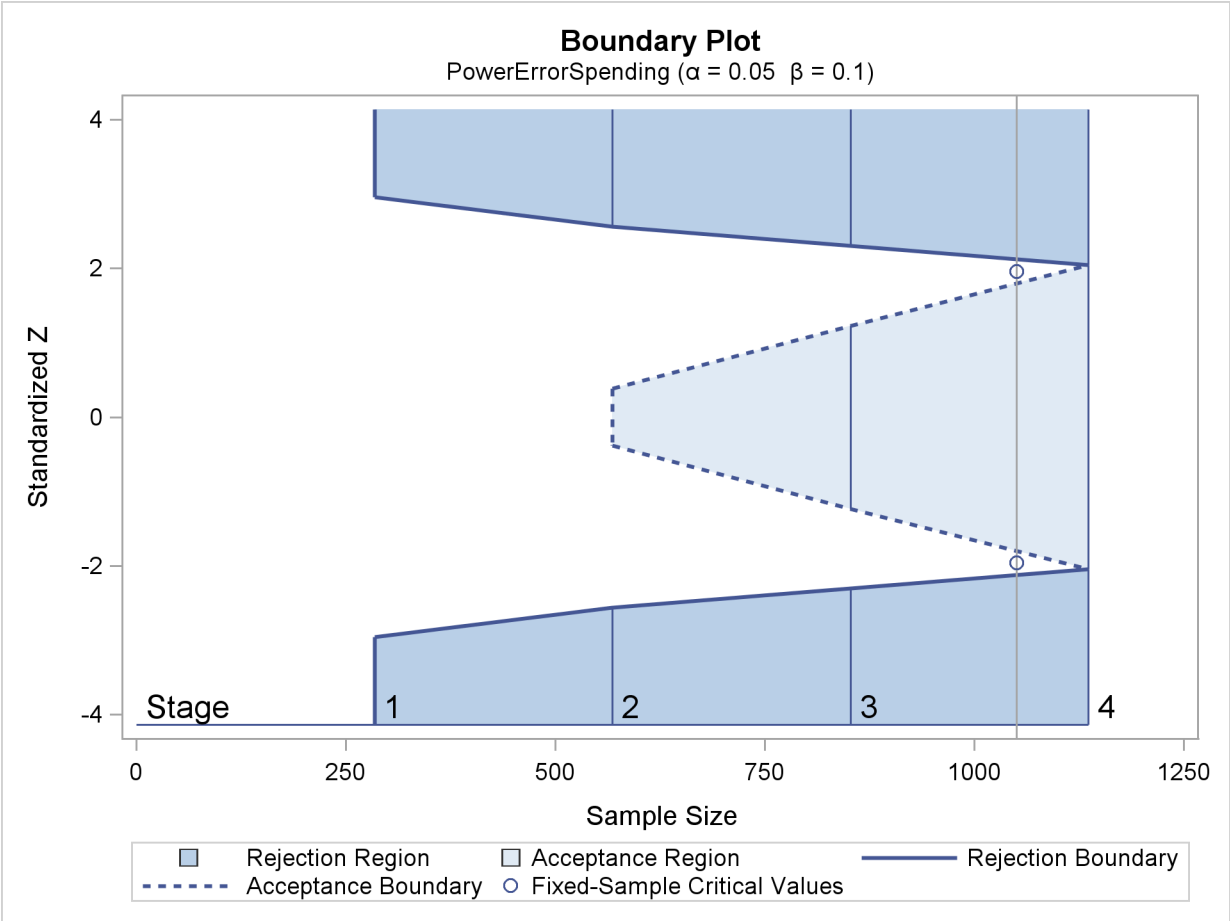

Figure 111.20 corresponds to a two-sided sequential design with early stopping to reject or accept the null hypothesis. For a sequential test with early stopping only to reject the null hypothesis, there are no acceptance boundary values at interim stages. The acceptance boundary value and its associated acceptance region are displayed only at the final stage. Similarly, for a sequential test with early stopping only to accept the null hypothesis, there are no rejection boundary values at interim stages. The rejection boundary value and its associated rejection region are displayed only at the final stage.

Information Level Adjustments at Future Stages

In a group sequential clinical trial, the information level for the observed test statistic at the current stage generally does not match the corresponding information level in the BOUNDARY= data set. By default (or equivalently if you specify INFOADJ=PROP), the SEQTEST procedure accommodates the observed information level by adjusting the information levels at future interim stages. The adjustment of information levels depends on the boundary key to be maintained in the boundary adjustments, which in turn is determined by the BOUNDARYKEY= option.

If you specify BOUNDARYKEY=ALPHA (which is the default) or BOUNDARYKEY=BETA, the maximum information level (the information level at the final stage) provided in the BOUNDARY= data set is maintained.

In this case, if an observed information level at the current stage is different from the level provided in the BOUNDARY= data set, you can use the INFOADJ= option to determine whether the information levels at subsequent interim stages are to be adjusted. Specifying INFOADJ=NONE preserves the levels provided in the BOUNDARY= data set without adjustment. Specifying INFOADJ=PROP proportionally adjusts the levels provided in the BOUNDARY= data set as follows.

Denote the information level at stage k for the K-stage design that is stored in the BOUNDARY= data set by I_k, $k = 1, 2, \ldots, K$. Also denote the information level that corresponds to the test statistic at an interim stage k_0 by I'_{k_0}, $1 \le k_0 \le (K-1)$. Then for the updated design, the information level at stage k, $k = k_0 + 1, \ldots, (K-1)$, is computed as

$$I'_k = I'_{k_0} + (I_K - I'_{k_0}) \frac{I_k - I_{k_0}}{I_K - I_{k_0}}$$

Note that if $I'_{k_0} \ge I_K$, the information level at stage k_0 reaches the maximum information level in the design, the trial stops at stage k_0, and no future information levels are derived.

If you specify BOUNDARYKEY=BOTH, the maximum information level for the trial is recomputed to maintain the α and β levels simultaneously. In this case, the INFOADJ=NONE option is not applicable, and the INFOADJ=PROP option is used to proportionally adjust the information levels at future interim stages with the updated maximum information I'_K. That is, with an updated I'_K, the information level at a future interim stage k is computed as

$$I'_k = I'_{k_0} + (I'_K - I'_{k_0}) \frac{I_k - I_{k_0}}{I_K - I_{k_0}}$$

Because PROC SEQTEST computes a new maximum information level when you specify BOUNDARYKEY=BOTH, you can use this option even if the original design is ceiling-adjusted (that is, if you specified the CEILADJDESIGN=INCLUDE suboption in the SAMPLESIZE statement in PROC SEQDESIGN). In this regard, the BOUNDARYKEY=BOTH option in PROC SEQTEST differs from the BOUNDARYKEY=BOTH option in PROC SEQDESIGN.

Boundary Adjustments for Information Levels

In a group sequential clinical trial, if the information level for the observed test statistic does not match the corresponding information level in the BOUNDARY= data set, the INFOADJ=PROP option (which is the default) can be used to modify information levels at future stages to accommodate this observed information level. With the adjusted information levels, the ERRSPENDADJ= option provides various methods to compute error spending values at the current and future interim stages. These error spending values are then used to derive boundary values in the SEQTEST procedure. See the section "Error Spending Methods" on page 9177 in Chapter 110, "The SEQDESIGN Procedure," for more information about how to use these error spending values to derive boundary values.

The ERRSPENDADJ=NONE option keeps the error spending the same at each stage. The ERRSPENDADJ=ERRLINE option uses a linear interpolation on the cumulative error spending in the design stored in the BOUNDARY= data set to derive the error spending for each unmatched information level (Kittelson and Emerson 1999, p. 882). That is, the cumulative error spending for an information level I

is computed as

$$e(I) = \begin{cases} e_1 \left(\frac{I}{I_1}\right) & \text{if } I < I_1 \\ e_j + (\alpha_{j+1} - \alpha_j) \left(\frac{I - I_j}{I_{j+1} - I_j}\right) & \text{if } I_j \le I < I_{j+1} \\ e_K & \text{if } I \ge I_K \end{cases}$$

where $e_1, e_2, \ldots, e_K$ are the cumulative errors at the K stages of the design that is stored in the BOUNDARY= data set.

The ERRSPENDADJ=ERRFUNCPOC option uses Pocock-type cumulative error spending function (Lan and DeMets 1983):

$$E(t) = \begin{cases} 1 & \text{if } t \ge 1 \\ \log(1 + (e-1)t) & \text{if } 0 < t < 1 \\ 0 & \text{otherwise} \end{cases}$$

With an error level of α or β, the cumulative error spending for an information level I is $e(I) = \alpha\, E(I/I_K)$ or $e(I) = \beta\, E(I/I_K)$.

The ERRSPENDADJ=ERRFUNCOBF option uses O'Brien-Fleming-type cumulative error spending function (Lan and DeMets 1983):

$$E(t; a) = \begin{cases} 1 & \text{if } t \ge 1 \\ \frac{1}{a}\, 2\,(1 - \Phi(\frac{z_{(1-a/2)}}{\sqrt{t}})) & \text{if } 0 < t < 1 \\ 0 & \text{otherwise} \end{cases}$$

where a is either α for the α spending function or β for the β spending function, and Φ is the cumulative distribution function of the standardized Z statistic. That is, with an error level of α or β, the cumulative error spending for an information level I is $e(I) = \alpha\, E(I/I_K; \alpha)$ or $e(I) = \beta\, E(I/I_K; \beta)$.

The ERRSPENDADJ=ERRFUNCGAMMA option uses gamma cumulative error spending function (Hwang, Shih, and DeCani 1990):

$$E(t; \gamma) = \begin{cases} 1 & \text{if } t \ge 1 \\ \frac{1-e^{-\gamma t}}{1-e^{-\gamma}} & \text{if } 0 < t < 1, \gamma \ne 0 \\ t & \text{if } 0 < t < 1, \gamma = 0 \\ 0 & \text{otherwise} \end{cases}$$

where γ is the parameter γ specified in the GAMMA= option. That is, with an error level of α or β, the cumulative error spending for an information level I is $e(I) = \alpha\, E(I/I_K; \gamma)$ or $e(I) = \beta\, E(I/I_K; \gamma)$.

The ERRSPENDADJ=ERRFUNCPOW option uses power cumulative error spending function (Jennison and Turnbull 2000, p. 148):

$$E(t; \rho) = \begin{cases} 1 & \text{if } t \ge 1 \\ t^{\rho} & \text{if } 0 < t < 1 \\ 0 & \text{otherwise} \end{cases}$$

where ρ is the power parameter specified in the RHO= suboption. With an error level of α or β, the cumulative error spending for an information level I is $e(I) = \alpha\, E(I/I_K; \rho)$ or $e(I) = \beta\, E(I/I_K; \rho)$.

If the BOUNDARYKEY=BOTH option is specified, the maximum information required for the trial might not be the same as the maximum information level stored in the BOUNDARY= data set. In this case, the information levels at future stages are adjusted proportionally, and the same error spending values that were computed based on the maximum information level stored in the BOUNDARY= data set are used to derive boundary values for the trial.

If an error spending function is used to create boundaries for the design in the SEQDESIGN procedure, then in order to better maintain the design features throughout the group sequential trial, the same error spending function to create boundaries for the design in the SEQDESIGN procedure should be used to modify boundaries in the SEQTEST procedure at each subsequent stage.

Boundary Adjustments for Minimum Error Spending

In a group sequential clinical trial, boundary values created from a design such as an O'Brien-Fleming design might be too conservative in early stages. Thus the trial is unlikely to stop in early stages. Lan and DeMets (1983, p. 662) suggest truncating boundary values to a number such as 3.5 for the trial to have a reasonable probability of stopping at early stages. Instead of truncating boundary values by a specified number, the ERRSPENDMIN= option provides individual minimum error spending at each interim stage to stop the trial early.

For a K-stage trial, denote the derived cumulative error spending at stage k after adjusting for information levels by $e_k, k = 1, 2, \ldots, K$. Also denote the specified minimum error spending at interim stage k by $\epsilon_k, k = 1, 2, \ldots, K-1$. Then the cumulative error spending at stage 1 is $e'_1 = \max(e_1,\ \epsilon_1)$. If $e_1 < e'_1$, the error spending values at subsequent interim stages are adjusted proportionally by

$$e'_j = e'_1 + \frac{e_j - e_1}{e_K - e_1}\,(e_K - e'_1)$$

for $j = 2, \ldots, K-1$.

The process is repeated at each subsequent interim stage. That is, at stage $k, k = 2, \ldots, K-1$, denote the updated cumulative β spending at stage j by $e_j,\ j = k, k+1, \ldots, K$. Then the cumulative error spending at stage k is $e'_k = \max(e_k,\ e'_{k-1} + \epsilon_k)$. If $e_k < e'_k$, the error spending values at subsequent interim stages are adjusted proportionally by

$$e'_j = e'_k + \frac{e_j - e_k}{e_K - e_k}\,(e_K - e'_k)$$

for $j = k+1, \ldots, K-1$.

Note that the ERRSPENDMIN= option is applicable only to the boundaries specified in the BOUNDARYKEY= option. That is, the ERRSPENDMIN= option is applicable to the α boundaries with BOUNDARYKEY=ALPHA or BOUNDARYKEY=BOTH, and it is applicable to the β boundaries with BOUNDARYKEY=BETA or BOUNDARYKEY=BOTH.

Boundary Adjustments for Overlapping Lower and Upper β Boundaries

In the SEQTEST procedure, the α and β spending values at the stages are used to derive the boundary values for the trial. For a two-sided design with early stopping to accept H_0, or to either reject or accept H_0, a zero β spending at an interim stage sets the β boundary values to missing. A small β spending at the current or subsequent interim stage might result in overlapping of the lower and upper β boundaries for the two corresponding one-sided tests. Specifically, this form of overlapping occurs at an interim stage k if the upper β boundary value that is derived from the one-sided test for the upper alternative is less than the lower β boundary value that is derived from the one-sided test for the lower alternative (Kittelson and Emerson 1999, pp. 881–882; Rudser and Emerson 2007, p. 6). You can use the BETAOVERLAP= option to specify how this type of overlapping is to be handled.

If BETAOVERLAP=ADJUST (which is the default) is specified, the procedure derives the boundary values for the two-sided design and then checks for overlapping of the two one-sided β boundaries at the current and subsequent interim stages. If overlapping occurs at a particular stage, the β boundary values for the two-sided design are set to missing (so the trial does not stop to accept the null hypothesis at this stage), and the β spending values at subsequent stages are adjusted proportionally as follows.

If the β boundary values are set to missing at stage k in a K-stage trial, the adjusted β spending value at stage k, e'_k, is updated for these missing β boundary values, and then the β spending values at subsequent stages are adjusted proportionally by

$$e'_j = e'_k + \frac{e_j - e_k}{e_K - e_k}\,(e_K - e'_k)$$

for $j = k+1, \ldots, K$, where e_j and e'_j are cumulative β spending values at stage j before and after the adjustment, respectively.

After all these adjusted β spending values are computed, the boundary values are then further modified for these adjusted β spending values.

If you specify BETAOVERLAP=NOADJUST, no adjustment is made when overlapping of one-sided β boundaries occurs.

Stochastic Curtailment

Lan, Simon, and Halperin (1982) introduce stochastic curtailment to stop a trial if, given current data, it is likely to predict the outcome of the trial with high probability. That is, a trial can be stopped to reject the null hypothesis H_0 if, given current data in the analyses, the conditional probability of rejecting H_0 under H_0 at the end of the trial is greater than γ, where the constant γ should be between 0.5 and 1 and values of 0.8 or 0.9 are recommended (Jennison and Turnbull 2000, p. 206). Similarly, a trial can be stopped to accept the null hypothesis H_0 if, given current data in the analyses, the conditional probability of rejecting H_0 under the alternative hypothesis H_1 at the end of the trial is less than γ.

The following two approaches for stochastic curtailment are available in the SEQTEST procedures: conditional power approach and predictive power approach. For each approach, the derived group sequential test is used as the reference test for rejection.

Conditional Power Approach

In the SEQTEST procedure, you can compute two types of conditional power as described in the following sections:

TYPE=ALLSTAGES

The default TYPE=ALLSTAGES suboption in the CONDPOWER and PLOT=CONDPOWER options computes the conditional power at an interim stage k as the total probability of rejecting the null hypothesis at all future stages given the observed statistic (Zhu, Ni, and Yao 2011, pp. 131–132).

For a one-sided test with an upper alternative, the conditional power at an interim stage k is given by

$$\begin{aligned} p_{ku}(\theta) &= P_\theta(z_{k+1} > a_{k+1} \mid z_k, \theta) \\ &+ P_\theta(b_{k+1} \le z_{k+1} < a_{k+1}, z_{k+2} > a_{k+2} \mid z_k, \theta) \\ &+ \ldots \\ &+ P_\theta(b_j \le z_j < a_j, j = k+1, \ldots, K-1, z_K > a_K \mid z_k, \theta) \end{aligned}$$

where z_k is the observed statistic and θ is the hypothetical reference. The conditional power for a one-sided test with a lower alternative is similarly derived.

For a two-sided test, the conditional power for the upper alternative is given by

$$\begin{aligned} p_{ku}(\theta) &= P_\theta(z_{k+1} > a_{k+1} \mid z_k, \theta) \\ &+ P_\theta(_a_{k+1} < z_{k+1} \le _b_{k+1} \text{ or } b_{k+1} \le z_{k+1} < a_{k+1}, z_{k+2} > a_{k+2} \mid z_k, \theta) \\ &+ \ldots \\ &+ {}_\theta(_a_j < z_j \le _b_j \text{ or } b_j \le z_j < a_j, j = k+1, \ldots, K-1, z_K > a_K \mid z_k, \theta) \end{aligned}$$

The conditional power for the lower alternative is similarly derived.

TYPE=FINALSTAGE

The TYPE=FINALSTAGE suboption in the CONDPOWER and PLOT=CONDPOWER options computes the conditional power at an interim stage k as the probability that the test statistic at the final stage (stage K) would exceed the rejection critical value given the observed statistic (Jennison and Turnbull 2000, p. 207).

The conditional distribution of Z_K given the observed statistic z_k at the kth stage and the hypothetical reference θ is

$$Z_K \mid (z_k, \theta) \sim N\left(z_k \ \Pi_k^{\frac{1}{2}} + \theta \ I_X^{\frac{1}{2}} (1 - \Pi_k) \, , \ 1 - \Pi_k \right)$$

where $\Pi_k = I_k / I_X$ is the fraction of information at the kth stage.

The power for the upper alternative, $\text{prob}(\, Z_K > a_K \mid z_k, \theta)$, is then given by

$$p_{ku}(\theta) = \Phi\left((1 - \Pi_k)^{-\frac{1}{2}} \, (z_k \, \Pi_k^{\frac{1}{2}} - a_K) \ + \ \theta \, I_X^{\frac{1}{2}} \, (1 - \Pi_k)^{\frac{1}{2}} \right)$$

where Φ is the cumulative distribution function of the standardized Z statistic and a_K is the upper critical value at the final stage.

Similarly, the power for the lower alternative, $\text{prob}(Z_K < a_K \mid z_k, \theta)$, is

$$p_{kl}(\theta) = 1 - \Phi\left((1-\Pi_k)^{-\frac{1}{2}}\,(z_k\,\Pi_k^{\frac{1}{2}} - a_K) \;+\; \theta\, I_X^{\frac{1}{2}}\,(1-\Pi_k)^{\frac{1}{2}}\right)$$

where a_K is the lower critical value at the final stage.

If $\theta = \hat{\theta}_k = z_k I_k^{-\frac{1}{2}}$, the maximum likelihood estimate at stage k, the powers for the upper and lower alternatives can be simplified:

$$p_{ku}(\theta) = \Phi\left((1-\Pi_k)^{-\frac{1}{2}}\;(z_k\,\Pi_k^{-\frac{1}{2}} - a_K)\right)$$

$$p_{kl}(\theta) = 1 - \Phi\left((1-\Pi_k)^{-\frac{1}{2}}\;(z_k\,\Pi_k^{-\frac{1}{2}} - a_K)\right)$$

If there exist interim stages between the kth stage and the final stage, $k < K-1$, the conditional power computed with TYPE=FINALSTAGE is not the conditional probability to reject the null hypothesis H_0. In this case, you can set the next stage as the final stage, and the conditional power is the conditional probability of rejecting H_0.

A special case of the conditional power is the futility index (Ware, Muller, and Braunwald 1985). It is 1 minus the conditional power under $H_1 : \theta = \theta_1$:

$$1 - p_{ku}(\theta_1) \;\text{ or }\; 1 - p_{kl}(\theta_1)$$

That is, it is the probability of accepting the null hypothesis under the alternative hypothesis given current data. A high futility index indicates a small probability of success (rejecting H_0) given the current data.

Predictive Power Approach

The conditional power depends on the specified reference θ, which might be supported by the current data (Jennison and Turnbull 2000, p. 210). An alternative is to use the predictive power (Herson 1979), which is a weighted average of the conditional power over values of θ. Without prior knowledge about θ, then with $\hat{\theta} = z_k/\sqrt{I_k}$, the maximum likelihood estimate at stage k, the posterior distribution for θ (Jennison and Turnbull 2000, p. 211) is

$$\theta \mid Z_K \sim N\left(\frac{z_k}{\sqrt{I_k}},\; \frac{1}{I_k}\right)$$

Thus, the predictive power at stage k for the upper and lower alternatives can be derived as

$$p_{ku} = 1 - \Phi\left((1-\Pi_k)^{-\frac{1}{2}}\;(a_K\,\Pi_k^{\frac{1}{2}} - z_k)\right)$$

$$p_{kl} = \Phi\left((1-\Pi_k)^{-\frac{1}{2}}\;(a_K\,\Pi_k^{\frac{1}{2}} - z_k)\right)$$

where a_K and a_K are the upper and lower critical values at the final stage.

Repeated Confidence Intervals

In a group sequential test, repeated confidence intervals for a parameter θ are defined as a sequence of intervals $(\hat{\theta}_{kl}, \hat{\theta}_{ku})$, $k = 1, 2, \ldots, K$, for which a simultaneous coverage probability is maintained (Jennison and Turnbull 2000, p. 189). That is, a $(1-\alpha)$ sequence of repeated confidence intervals has

$$\text{Prob}(\hat{\theta}_{kl} \le \theta \le \hat{\theta}_{ku}) = 1 - \alpha$$

These confidence limits $\hat{\theta}_{kl}$ and $\hat{\theta}_{ku}$ can be created from observed statistic and boundary values at each stage.

Two-Sided Repeated Confidence Intervals

Two sequences of repeated confidence intervals can be derived for a two-sided test. One is a $(1-\alpha_l-\alpha_u)$ rejection repeated confidence intervals $(\hat{\theta}_{kl}(\alpha), \hat{\theta}_{ku}(\alpha))$, $k = 1, 2, \ldots, K$, and the other is a $(1-\beta_l-\beta_u)$ acceptance repeated confidence intervals $(\hat{\theta}_{kl}(\beta), \hat{\theta}_{ku}(\beta))$, $k = 1, 2, \ldots, K$, where α_l and α_u are the lower and upper Type I error probabilities for the test and β_l and β_u are the lower and upper Type II error probabilities for the test (Jennison and Turnbull 2000, p. 196).

The rejection lower and upper repeated confidence limits at stage k are

$$\hat{\theta}_{kl}(\alpha) = \hat{\theta}_k - \frac{a_k}{\sqrt{I_k}} \qquad \hat{\theta}_{ku}(\alpha) = \hat{\theta}_k - \frac{a_k}{\sqrt{I_k}}$$

The hypothesis is rejected for upper alternative if the lower limit $\hat{\theta}_{kl}(\alpha) > \theta_{0u}$ and is rejected for lower alternative if the upper limit $\hat{\theta}_{ku}(\alpha) < \theta_{0l}$. That is, the hypothesis is rejected if both θ_{0l} and θ_{0u} are not in a rejection repeated confidence interval $(\hat{\theta}_{kl}(\alpha), \hat{\theta}_{ku}(\alpha))$.

The acceptance lower and upper repeated confidence limits at stage k are

$$\hat{\theta}_{kl}(\beta) = \hat{\theta}_k + \left(\theta_{1l} - \frac{b_k}{\sqrt{I_k}}\right) \qquad \hat{\theta}_{ku}(\beta) = \hat{\theta}_k + \left(\theta_{1u} - \frac{b_k}{\sqrt{I_k}}\right)$$

The hypothesis is accepted if the lower limit $\hat{\theta}_{kl}(\beta) > \theta_{1l}$ and the upper limit $\hat{\theta}_{ku}(\beta) < \theta_{1u}$. That is, a repeated confidence interval is contained in the interval $(\theta_{1l}, \theta_{1u})$.

One-Sided Repeated Confidence Intervals

Like the two-sided repeated confidence intervals, two sequences of repeated confidence intervals can be derived for a one-sided test. Suppose the one-sided test has an upper alternative θ_{1u}. Then one sequence of repeated confidence intervals is a $(1-\alpha_u)$ rejection repeated confidence intervals $(\hat{\theta}_{kl}(\alpha), \infty)$, $k = 1, 2, \ldots, K$, and the other is a $(1-\beta_u)$ acceptance repeated confidence intervals $(-\infty, \hat{\theta}_{ku}(\beta))$, $k = 1, 2, \ldots, K$, where α_u and β_u are the upper Type I and Type II error probabilities for the test. Thus, a sequence of repeated confidence intervals with confidence level greater than or equal to $(1-\alpha_u-\beta_u)$ is given by $(\hat{\theta}_{kl}(\alpha), \hat{\theta}_{ku}(\beta))$.

The rejection lower repeated confidence limit and the acceptance upper repeated confidence limit at stage k are

$$\hat{\theta}_{kl}(\alpha) = \hat{\theta}_k - \left(\frac{a_k}{\sqrt{I_k}} - \theta_{0u}\right) \qquad \hat{\theta}_{ku}(\beta) = \hat{\theta}_k + \left(\theta_{1u} - \frac{b_k}{\sqrt{I_k}}\right)$$

The hypothesis is rejected if the lower limit $\hat{\theta}_{kl}(\alpha) > \theta_{0u}$. and it is accepted if the upper limit $\hat{\theta}_{ku}(\beta) < \theta_{1u}$.

Analysis after a Sequential Test

At the end of a trial, the hypothesis is either rejected or accepted. But the p-value, median, and confidence limits depend on the ordering the sample space (k, z), where k is the stage number and z is the standardized Z statistic.

Following the notations used in Jennison and Turnbull (2000, pp. 179–180), $(k', z') \succ (k, z)$ if (k', z') has a higher order or more extreme than (k, z). Then for a given ordering, the p-value, median, and confidence limits associated with the observed statistics (k, z) can be derived.

p-value

With the observed pair of statistics (k_0, z_0) when the trial is stopped, a one-sided upper p-value is computed as

$$\mathrm{Prob}\{\, (k, z) \succeq (k_0, z_0) \,\}$$

A one-sided lower p-value is computed as

$$\mathrm{Prob}\{\, (k, z) \preceq (k_0, z_0) \,\}$$

A two-sided p-value is twice the smaller of the lower and upper p-values.

Median Unbiased Estimate

With the observed pair (k_0, z_0), a median unbiased estimate θ_m is computed from

$$\mathrm{Prob}\{\, (k, z) \succeq (k_0, z_0) \mid \theta_m \,\} = 0.50$$

Confidence Limits

With the observed pair (k_0, z_0), a lower $(1 - \alpha_l)$ confidence limit for θ, θ_l, is computed from

$$\mathrm{Prob}\{\, (k, z) \succeq (k_0, z_0) \mid \theta_l \,\} = \alpha_l$$

Similarly, an upper $(1 - \alpha_u)$ confidence limit for θ, θ_u, is computed from

$$\mathrm{Prob}\{\, (k, z) \preceq (k_0, z_0) \mid \theta_u \,\} = \alpha_u$$

Available Sample Space Orderings in a Sequential Test

At the end of a trial, the hypothesis is either rejected or accepted. Denote the stage number and the statistic at the end of a trial by a pair of statistics (k, z), where k is the stage number and z is the standardized Z statistic. Then an ordering on the sample space (k, z) is needed to derive the p-value, median, and confidence limits associated with the observed statistics (k^*, z^*).

The SEQTEST procedure provides the stagewise, LR, and MLE orderings. See Jennison and Turnbull (2000, pp. 179–187) for a detailed description and comparison of these orderings.

Stagewise Ordering

If the continuation regions of a design are intervals, the stagewise ordering (Fairbanks and Madsen 1982; Tsiatis, Rosner, and Mehta 1984; Jennison and Turnbull 2000, pp. 179–180) uses counter-clockwise ordering around the continuation region to compute the p-value, unbiased median estimate, and confidence limits. This ordering depends on the stopping region, stopping stage, and standardized statistic at the stopping stage. But it does not depend on information levels beyond the observed stage. For a one-sided design with an upper alternative, $(k', z') \succ (k, z)$ if one of the following criteria holds:

- $k' = k$ and $z' > z$
- $k' < k$ and $z' \geq a_{k'}$, the upper α boundary at stage k'
- $k' > k$ and $z < b_k$, the upper β boundary at stage k

Similar criteria can be derived for a one-sided design with a lower alternative.

For a two-sided design with early stopping to reject the null hypothesis, $(k', z') \succ (k, z)$ if one of the following criteria holds:

- $k' = k$ and $z' > z$
- $k' < k$ and $z' \geq a_{k'}$, the upper α boundary at stage k'
- $k' > k$ and $z \leq _a_k$, the lower α boundary at stage k

Note that the stagewise ordering is not applicable for two-sided designs with early stopping to accept H_0 or to either accept or reject H_0, which might have two disjoint continuous intervals at each interim stage.

For a two-sided design with early stopping either to reject or to accept the null hypothesis, $(k', z') \succ (k, z)$ if one of the following criteria holds:

- $z' \geq a_{k'}$ and $z < b_k$
- $z' > _b_{k'}$ and $z \leq _a_k$

That is, each value in the continuation region is less extreme than each value in the upper rejection region and more extreme than each value in the lower rejection region. Then, combining with the ordering defined for a two-sided design with early stopping to reject the null hypothesis, the p-value, median, and confidence limits can be derived for the observed statistics in the lower or upper rejection region.

Thus, if the stagewise ordering is specified in the SEQTEST procedure for a two-sided design with early stopping to either reject or accept the null hypothesis, the stagewise ordering is used to derive these statistics only if the observed statistics is in the lower or upper rejection region. Otherwise, the LR ordering is used.

LR Ordering

The LR ordering (Chang 1989) depends on the observed standardized Z statistic z, information levels, and a specified hypothetical reference. For the LR ordering under a given hypothesis $H : \theta = \theta_g$, $(k', z') \succ (k, z)$ if

$$(z' - \theta_g \sqrt{I_{k'}}) \ > \ (z - \theta_g \sqrt{I_k})$$

Under the null hypothesis $H_0 : \theta = 0$, it reduces to

$$z' > z$$

and can be used to derive statistics under H_0, such as p-values.

The LR ordering is applicable to all designs if all information levels are available. But depending on the boundary shape, some observed statistics (k, z) in the rejection region might be less extreme than the statistics in the acceptance region. That is, the p-value for observed statistics in the rejection region might be greater than the significance level.

MLE Ordering

The MLE ordering (Emerson and Fleming 1990) depends only on the observed maximum likelihood estimate. $(k', z') \succ (k, z)$ if

$$\frac{z'}{\sqrt{I_{k'}}} > \frac{z}{\sqrt{I_k}}$$

The MLE ordering is applicable to all designs if all information levels are available.

Applicable Tests and Sample Size Computation

The SEQDESIGN procedure assumes that the data are from a multivariate normal distribution and the sequence of the standardized test statistics $\{Z_1, Z_2, \ldots, Z_K\}$ have the following canonical joint distribution:

- $Z_k \sim N\left(\theta\sqrt{I_k}, 1\right)$
- $\mathrm{Cov}(Z_{k_1}, Z_{k_2}) = \sqrt{I_{k_1}/I_{k_2}} \quad , \quad 1 \le k_1 \le k_2 \le K$

where K is the total number of stages and I_k is the information available at stage k.

If the data are not from a normal distribution such as binomial distribution, then it is assumed that the test statistic is computed from a large sample such that the statistic has an approximately normal distribution.

In a clinical trial, the sample size required depends on the Type I error probability α, reference improvement θ_1, power $1-\beta$, and variance of the response variable. Given a null hypothesis $H_0 : \theta = 0$ with an upper alternative hypothesis $H_1 : \theta = \theta_1$, the information required for a fixed-sample test is given by

$$I_0 = \frac{(\Phi^{-1}(1-\alpha) + \Phi^{-1}(1-\beta))^2}{\theta_1^2}$$

where the parameter θ depends on the test specified in the clinical trial. For example, if you are comparing two binomial populations $H_0 : \theta = 0$, then $\theta = p_t - p_c$ is the difference between two proportions if the proportion difference statistic is used, and $\theta = \log\left(\frac{p_t(1-p_c)}{p_c(1-p_t)}\right)$, the log odds ratio for the two proportions if the log odds ratio statistic is used.

If the maximum likelihood estimate $\hat{\theta}$ from the likelihood function can be derived, then the asymptotic variance for $\hat{\theta}$ is $\mathrm{Var}(\hat{\theta}) = 1/I$, where I is Fisher's information for θ.

The resulting statistic $\hat{\theta}$ corresponds to the MLE scale as specified in the BOUNDARYSCALE=MLE option in the PROC SEQDESIGN statement, $\hat{\theta}\sqrt{I}$ corresponds to the standardized Z scale (BOUNDARYSCALE=STDZ), and $\hat{\theta}I$ corresponds to the score scale (BOUNDARYSCALE=SCORE).

Alternatively, if the score statistic is derived, it can also be used as the test statistic and its asymptotic variance is given by Fisher's information.

For a group sequential trial, the maximum information I_X is derived in the SEQDESIGN procedure by using the specified α, β, and θ_1. With the maximum information

$$I_X = \frac{1}{\mathrm{Var}(\hat{\theta})}$$

the sample size required for a specified test statistic in the trial can be evaluated or estimated from the known or estimated variance of the response variable. Note that different designs might produce different maximum information levels for the same hypothesis, and this in turn might require a different number of observations for the trial.

With a specified test statistic, the required sample sizes at the stages can be computed. These tests include commonly used tests for normal means, binomial proportions, and survival distributions. See the section "Sample Size Computation" on page 9185 in Chapter 110, "The SEQDESIGN Procedure," for more information about these tests.

Table Output

The SEQTEST procedure displays the "Design Information" and "Test Information" tables by default.

Conditional Power

The "Conditional Power Information" table displays the following information under a hypothetical reference:

- stopping stage
- MLE, observed maximum likelihood estimate
- conditional power under the hypothetical reference

For a one-sided test, the power are derived under hypothetical references $\theta = \hat{\theta}$ and $\theta = c_i\theta_1$, where $\hat{\theta}$ is the observed statistic, θ_1 is the alternative reference, and c_i are the values specified in the CREF= option. For a two-sided test, the power are derived under the hypothetical references $\theta = \hat{\theta}$, $\theta = c_i\theta_{1l}$, and $\theta = c_i\theta_{1u}$, where θ_{1l} is the lower alternative reference and θ_{1u} is the upper alternative reference. The default is CREF= 0 0.5 1.0 1.5.

Design Information

The "Design Information" table displays the design specifications and derived statistics. The derived Max Information (Percent Fixed-Sample) is the maximum information for the sequential design in percentage of the corresponding fixed-sample information.

The Null Ref ASN (Percent Fixed-Sample) is the average sample size required under the null hypothesis for the group sequential design in percentage of the corresponding fixed-sample design. Similarly, the Alt Ref ASN (Percent Fixed-Sample) is the average sample size required under the alternative reference for the group sequential design in percentage of the corresponding fixed-sample design.

Error Spending Information

The "Error Spending Information" table displays the following information at each stage:

- proportion of information
- actual information level, if the maximum information is either specified or derived
- cumulative error spending for each boundary

Parameter Estimates

The "Parameter Estimates" table displays the following information at the conclusion of a sequential trial:

- stopping stage
- parameter estimate
- median and confidence limits based on the specified ordering
- p-value for the hypothesis H_0 based on the specified ordering

Powers and Expected Sample Sizes

The "Powers and Expected Sample Sizes" table displays the following information under each of the specified hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and c_i are values specified in the CREF= option.

- coefficient c_i for the hypothetical references. The value $c_i = 0$ corresponds to the null hypothesis and $c_i = 1$ corresponds to the alternative hypothesis
- power
- expected sample size, as percentage of fixed-sample size

For a one-sided design, the power and expected sample sizes under the hypothetical references $\theta = c_i\theta_1$ are displayed.

For a two-sided symmetric design, the power and expected sample sizes under each of the hypothetical references $\theta = c_i\theta_{1u}$ are displayed, where θ_{1u} is the upper alternative reference.

For a two-sided asymmetric design, the power and expected sample sizes under each of the hypothetical references $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively.

For a two-sided design, the power is the probability of correctly rejecting the null hypothesis for the correct alternative. Thus, under the null hypothesis, the displayed power corresponds to a one-sided Type I error probability level—that is, the lower α level or the upper α level.

The expected sample size as a percentage of the corresponding fixed-sample design is

$$100 \times \frac{\sum_{k=1}^{K} p_k\, I_k}{I_0}$$

where p_k is the stopping probability at stage k, $\sum_{k=1}^{K} p_k\, I_k$ is the expected information level, and I_0 is the information level for the fixed-sample design.

Predictive Power

The "Predictive Power Information" table displays the following information:

- stopping stage
- MLE, observed maximum likelihood estimate
- predictive power

Repeated Confidence Intervals

The "Repeated Confidence Intervals" table displays the following information for the observed statistic at each stage:

- information level
- parameter estimate
- rejection confidence limits. The null hypothesis is rejected for the upper alternative if the lower rejection confidence limit is greater than the null parameter value. Similarly, the null hypothesis is rejected for the lower alternative if the upper rejection confidence limit is less than the null parameter value.
- acceptance confidence limits. The upper alternative hypothesis is rejected if the upper acceptance confidence limit is less than the upper alternative value. Similarly, the lower alternative hypothesis is rejected if the lower acceptance confidence limit is greater than the upper alternative value. For a two-sided design, if both upper and lower alternative hypothesis are rejected, the null hypothesis is accepted.

Stopping Probabilities

The "Expected Cumulative Stopping Probabilities" table displays the following information under each of the specified hypothetical references $\theta = c_i\theta_1$, where c_i are values specified in the CREF= option, and θ_1 is the alternative reference:

- coefficient c_i for the hypothetical references. The value $c_i = 0$ corresponds to the null hypothesis, and $c_i = 1$ corresponds to the alternative hypothesis
- expected stopping stage
- source of the stopping probability: reject H_0 (with STOP=REJECT or STOP=BOTH), accept H_0 (with STOP=ACCEPT or STOP=BOTH), or either reject or accept H_0 (with STOP=BOTH)
- expected cumulative stopping probabilities at each stage

For a one-sided design, the expected cumulative stopping probabilities under the hypothetical references $\theta = c_i \theta_1$ are displayed.

For a two-sided design, the expected cumulative stopping probabilities under each of the hypothetical references $\theta = c_i \theta_{1l}$ and $\theta = c_i \theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively.

Note that for a symmetric two-sided design, only the expected cumulative stopping probabilities under the hypothetical references $\theta = c_i \theta_{1u}$ are derived.

The expected stopping stage is given by $k_0 + d$ and is derived from the expected information level

$$\sum_{k=1}^{K} p_k \, I_k = I_{k_0} + d \, (I_{(k_0+1)} - I_{k_0})$$

where p_k is the stopping probability at stage k and $0 \leq d < 1$.

For equally spaced information levels, the expected stopping stage is reduced to the weighted average

$$\sum_{k=1}^{K} p_k \, k$$

Test Information

The "Test Information" table displays the following information at each stage:

- proportion of information
- actual information level, if the maximum information is available from the input BOUNDARY= data set
- alternative references with the specified statistic scale. If a p-value scale is specified, the standardized Z scale is used.
- boundary values with the specified statistic scale to reject or accept the null hypothesis

Note that implicitly, the test information table also contains variables for the boundary scale, stopping criterion, and type of alternative hypothesis. That is, if an ODS statement is used to save the table, the data set also contains the variables _Scale_ for the boundary scale, _Stop_ for the stopping criterion, and _ALT_ for the type of alternative hypothesis.

If the test variable is specified, the table also displays the following:

- test statistic
- resulting action of test statistic: continue to the next stage, accept the null hypothesis H_0, or reject H_0

ODS Table Names

PROC SEQTEST assigns a name to each table it creates. You must use these names to reference tables when using the Output Delivery System (ODS). These names are listed in Table 111.3. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 111.3 ODS Tables Produced by PROC SEQTEST

ODS Table Name	Description	Option
CondPower	Conditional power	CONDPOWER
Design	Design information	
ErrSpend	Error spending	ERRSPEND
ParameterEstimates	Parameter estimates	DATA(TESTVAR=) or PARMS(TESTVAR=)
PowerSampleSize	Power and expected sample sizes	PSS
PredPower	Predictive power	PREDPOWER
RepeatedCI	Repeated confidence intervals	RCI
StopProb	Stopping probabilities	STOPPROB
Test	Test statistics and boundary values	

Graphics Output

This section describes the use of ODS for creating graphics with the SEQTEST procedure. To request these graphs, ODS Graphics must be enabled and you must specify the associated graphics options in the PROC SEQTEST statement. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Sequential ASN Plot

The PLOTS=ASN option displays the average sample numbers (expected sample sizes for nonsurvival data or expected numbers of events for survival data) under various hypothetical references. The average sample numbers are connected for each design, and these connected curves for all designs are displayed in the "Sequential ASN Plot" graph.

For a one-sided design, average sample numbers under the hypothetical references $\theta = c_i\theta_1$ are displayed, where c_i are the values specified in the CREF= option and θ_1 is the alternative reference. The horizontal axis displays the c_i values of these hypothetical references.

For a two-sided design, average sample numbers under each of the hypothetical references $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. The horizontal axis displays $-c_i$ values for lower hypothetical references $\theta = c_i\theta_{1l}$ and c_i values for upper hypothetical references $\theta = c_i\theta_{1u}$. Note that for a symmetric two-sided design, only average sample numbers under the hypothetical references $\theta = c_i\theta_{1u}$ are derived.

If the trial stops after the sequential test, the hypothetical reference corresponding to the test statistic is also indicated in the plot.

Conditional Power Plot

The PLOTS=CONDPOWER option displays the conditional powers given the observed statistic under various hypothetical references. These powers are connected and are displayed in the "Conditional Power Plot" graph.

For a one-sided test, the power are derived under the hypothetical references $\theta = \hat{\theta}$ and $\theta = c_i\theta_1$, where $\hat{\theta}$ is the observed statistic, θ_1 is the alternative reference, and c_i are the values specified in the CREF= option. The horizontal axis displays these c_i values for hypothetical references.

For a two-sided test, the power are derived under hypothetical references $\theta = \hat{\theta}$, $\theta = c_i\theta_{1l}$, and $\theta = c_i\theta_{1u}$, where θ_{1l} is the lower alternative reference and θ_{1u} is the upper alternative reference. The horizontal axis displays $-c_i$ values for hypothetical references $\theta = c_i\theta_{1l}$ and c_i values for hypothetical references $\theta = c_i\theta_{1u}$.

If the trial stops after the sequential test, the hypothetical reference corresponding to the test statistic is also indicated in the plot.

Sequential Error Spending Plot

The PLOTS=ERRSPEND option displays the cumulative error spending at each stage on each boundary in the "Sequential Error Spending Plot" graph. A legend table uses the design labels to identify the curves for the corresponding design in the plot.

Sequential Power Plot

The PLOTS=POWER option displays the powers under various hypothetical references. The powers are connected for each design, and these connected curves for all designs are displayed in the "Sequential Power Plot" graph.

For a one-sided design, powers under hypothetical references $\theta = c_i\theta_1$ are displayed, where c_i are the values specified in the CREF= option and θ_1 is the alternative reference. The horizontal axis displays the c_i values of these hypothetical references.

For a two-sided design, powers under hypothetical references $\theta = c_i\theta_{1l}$ and $\theta = c_i\theta_{1u}$ are displayed, where θ_{1l} and θ_{1u} are the lower and upper alternative references, respectively. The horizontal axis displays $-c_i$ values for lower hypothetical references $\theta = c_i\theta_{1l}$ and c_i values for upper hypothetical references $\theta = c_i\theta_{1u}$. Note that for a symmetric two-sided design, only powers under hypothetical references $\theta = c_i\theta_{1u}$ are derived.

If the trial stops after the sequential test, the hypothetical reference corresponding to the test statistic is also indicated in the plot.

Repeated Confidence Intervals Plot

The PLOTS=RCI option displays repeated confidence intervals at each stage given the observed statistic at that stage. These repeated confidence intervals are displayed in the "Repeated Confidence Intervals Plot" graph.

Sequential Test Plot

The PLOTS=TEST option displays boundary values and test statistics in the "Test Plot" graph. The boundary values are connected for each boundary, and both the stage number and the information level at each stage are displayed. The legend table identifies the acceptance and rejection regions in the plot.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC SEQTEST assigns a name to each graph it creates. You can use these names to reference the graphs when using ODS. To request these graphs, ODS Graphics must be enabled and you must specify the *options* indicated in Table 111.4.

Table 111.4 Graphs Produced by PROC SEQTEST

ODS Graph Name	Plot Description	Option
AsnPlot	Average sample numbers	PLOTS=ASN
CondPowerPlot	Conditional power curves	PLOTS=CONDPOWER
ErrSpendPlot	Error spending	PLOTS=ERRSPEND
PowerPlot	Power curves	PLOTS=POWER
RepeatedCIPlot	Repeated confidence intervals	PLOTS=RCI
TestPlot	Boundary values and test statistics	PLOTS=TEST

Examples: SEQTEST Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The following examples perform group sequential tests with various designs and test statistics.

Four statistic scales are available for the input boundary values, the input test statistic, and the displayed test information in the SEQTEST procedure. These are the maximum likelihood estimator scale, score statistic scale, standardized normal Z scale, and p-value scale. There is a unique one-to-one transformation between any two of the scales, and you can use different scales for the input boundary values and input test statistic. These boundary values and test statistic are displayed with the scale specified in the BOUNDARYSCALE= option in the SEQTEST procedure.

Example 111.1: Testing the Difference between Two Proportions

(View the complete code for this example (seqtex1.sas) in the example repository.)

This example demonstrates group sequential tests that use an O'Brien-Fleming group sequential design. A clinic is studying the effect of vitamin C supplements in treating flu symptoms. The study consists of patients in the clinic who have exhibited the first sign of flu symptoms within the last 24 hours. These patients are randomly assigned to either the control group (which receives placebo pills) or the treatment group (which receives large doses of vitamin C supplements). At the end of a five-day period, the flu symptoms of each patient are recorded.

Suppose that you know from past experience that flu symptoms disappear in five days for 60% of patients who experience flu symptoms. The clinic would like to detect a 75% symptom disappearance with a high probability. A test that compares the proportions directly specifies the null hypothesis $H_0 : \theta = p_t - p_c = 0$ with a one-sided alternative $H_1 : \theta > 0$ and a power of 0.90 at $H_1 : \theta = 0.15$, where p_t and p_c are the proportions of symptom disappearance in the treatment group and control group, respectively.

The following statements invoke the SEQDESIGN procedure and request a four-stage group sequential design by using an O'Brien-Fleming method for normally distributed data. The design uses a one-sided alternative hypothesis with early stopping either to accept or reject the null hypothesis H_0. The BOUNDARYSCALE=MLE option uses the MLE scale to display statistics in the boundary table and boundary plots.

```
ods graphics on;
proc seqdesign altref=0.15
               boundaryscale=mle
               ;
   OBrienFleming: design method=obf
                         nstages=4
                         alt=upper
                         stop=both
                         alpha=0.025
                         ;
   samplesize model=twosamplefreq(nullprop=0.6 test=prop);
   ods output Boundary=Bnd_Count;
```

```
run;
```

The ODS OUTPUT statement with the BOUNDARY=BND_COUNT option creates an output data set named BND_COUNT which contains the resulting boundary information for the subsequent sequential tests.

The "Design Information" table in Output 111.1.1 displays design specifications. With the specified alternative hypothesis $H_1 : \theta = 0.15$, the maximum information is derived to achieve a power of 0.90 at H_1. The derived fixed-sample information ratio 1.0767 is the maximum information needed for a group sequential design relative to its corresponding fixed-sample design.

Output 111.1.1 O'Brien-Fleming Design Information

The SEQDESIGN Procedure
Design: OBrienFleming

Design Information	
Statistic Distribution	Normal
Boundary Scale	MLE
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Method	O'Brien-Fleming
Boundary Key	Both
Alternative Reference	0.15
Number of Stages	4
Alpha	0.025
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	107.6741
Max Information	502.8343
Null Ref ASN (Percent of Fixed Sample)	61.12891
Alt Ref ASN (Percent of Fixed Sample)	75.89782

The "Boundary Information" table in Output 111.1.2 displays the information level, alternative reference, and boundary values at each stage. With the BOUNDARYSCALE=MLE option, the SEQDESIGN procedure displays the output boundaries with the maximum likelihood estimator scale.

Output 111.1.2 O'Brien-Fleming Boundary Information

Boundary Information (MLE Scale)
Null Reference = 0

	Information Level			Alternative Reference	Boundary Values Upper	
Stage	Proportion	Actual	N	Upper	Beta	Alpha
1	0.2500	125.7086	107.4808	0.15000	-0.09709	0.35291
2	0.5000	251.4171	214.9617	0.15000	0.02645	0.17645
3	0.7500	377.1257	322.4425	0.15000	0.06764	0.11764
4	1.0000	502.8343	429.9233	0.15000	0.08823	0.08823

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.1.3. The horizontal axis indicates the information levels for the design. The stages

are indicated by vertical lines with accompanying stage numbers. If the test statistic at a stage is in a rejection region, the trial stops and the hypothesis is rejected. If the test statistic is in an acceptance region, then the trial also stops and the hypothesis is accepted. If the statistic is not in a rejection or an acceptance region, the trial continues to the next stage.

Output 111.1.3 O'Brien-Fleming Boundary Plot

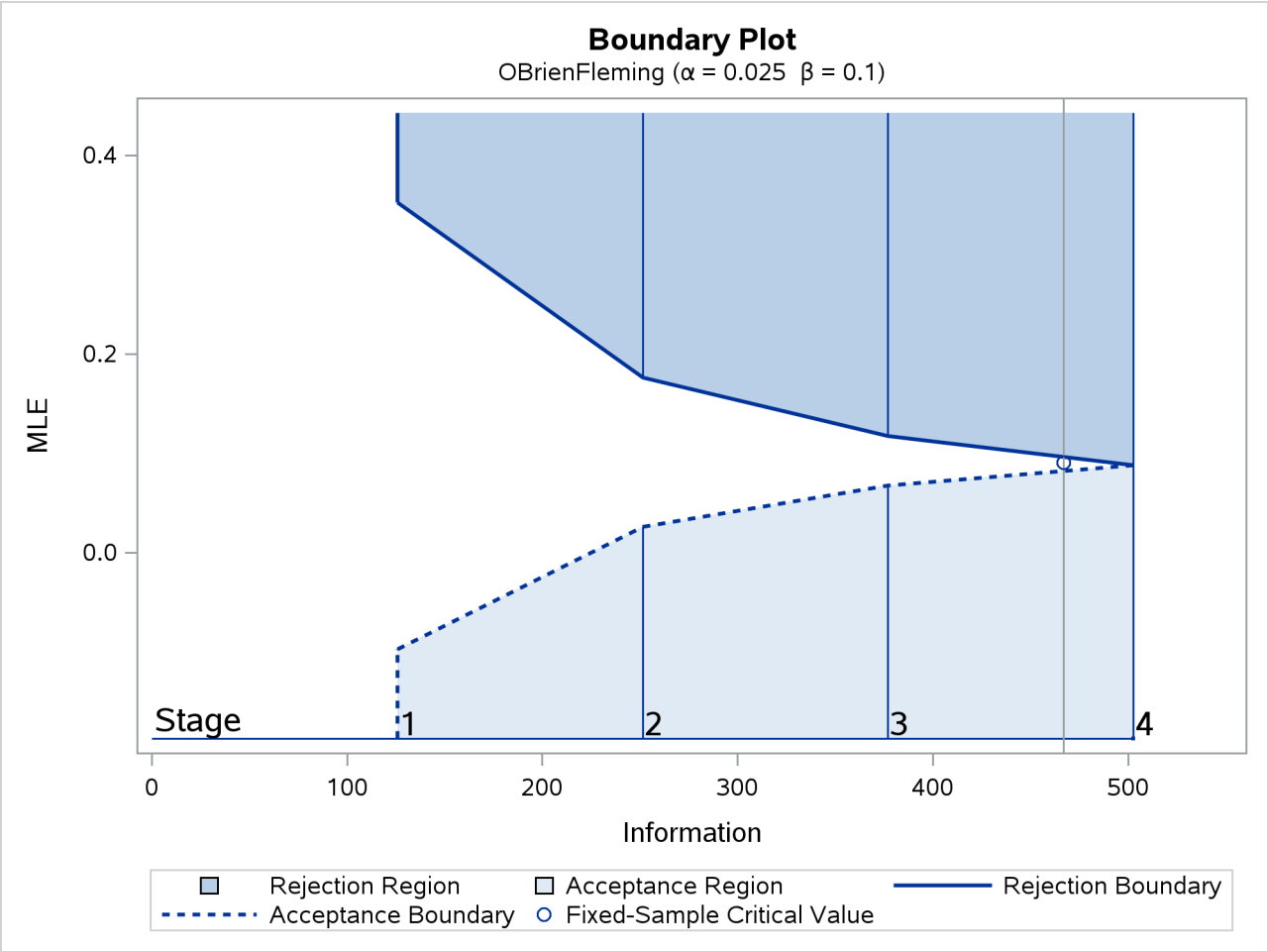

The boundary plot also displays the information level and critical value for the corresponding fixed-sample design. The solid and dashed lines at the fixed-sample information level correspond to the rejection and acceptance lines, respectively.

With the SAMPLESIZE statement, the maximum information is used to derive the required sample size for the study. The "Sample Size Summary" table in Output 111.1.4 displays parameters for the sample size computation.

Output 111.1.4 Required Sample Size Summary

Sample Size Summary	
Test	Two-Sample Proportions
Null Proportion	0.6
Proportion (Group A)	0.75
Test Statistic	Z for Proportion
Reference Proportions	Alt Ref
Max Sample Size	429.9233
Expected Sample Size (Null Ref)	244.0768
Expected Sample Size (Alt Ref)	303.0464

With the derived maximum information and the specified MODEL= option in the SAMPLESIZE statement, the total sample size in each group for testing the difference between two proportions under the alternative hypothesis is

$$N_1 = N_2 = (p_{1c}\,(1 - p_{1c}) + p_{1t}\,(1 - p_{1t}))\ I_X$$

where $p_{1c} = 0.6$ and $p_{1t} = p_{1c} + \theta_1 = 0.75$. By default (or equivalently if you specify REF=PROP in the MODEL=TWOSAMPLEFREQ option), the required sample sizes are computed under the alternative hypothesis. See the section "Test for the Difference between Two Binomial Proportions" on page 9197 in Chapter 110, "The SEQDESIGN Procedure," for more information about these parameters.

The "Sample Sizes (N)" table in Output 111.1.5 displays the required sample sizes at each stage, in both fractional and integer numbers. The derived sample sizes under the heading Fractional N which correspond to the design are not integers. These sample sizes are rounded up to integers under the heading Ceiling N. In practice, integer sample sizes are used, and the information levels increase slightly. Thus, 54, 108, 162, and 215 patients are needed in each group for the four stages, respectively.

Output 111.1.5 Required Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Proportion Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	107.48	53.74	53.74	125.7	108	54	54	126.3
2	214.96	107.48	107.48	251.4	216	108	108	252.6
3	322.44	161.22	161.22	377.1	324	162	162	378.9
4	429.92	214.96	214.96	502.8	430	215	215	502.9

Suppose the trial follows the study plan, and 54 patients are available in each group at stage 1. The data set count_1 contains these 108 patients. Output 111.1.6 lists the first 10 observations of the data set.

Output 111.1.6 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Trt	Resp
1	0	0
2	1	1
3	0	1
4	1	0
5	0	0
6	1	0
7	0	0
8	1	1
9	0	1
10	1	0

The Trt variable is a grouping variable with value 0 for a patient in the placebo control group and value 1 for a patient in the treatment group who is given vitamin C supplements. The Resp variable is an indicator variable with value 1 for a patient without flu symptoms after five days and value 0 for a patient with flu symptoms after five days.

The following statements use the GENMOD procedure to estimate the treatment effect at stage 1:

```
proc genmod data=count_1;
   model Resp= Trt;
   ods output ParameterEstimates=Parms_Count1;
run;
```

Output 111.1.7 displays the treatment effect at stage 1.

Output 111.1.7 Stage 1 Treatment Difference

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	0.6296	0.0627	0.5066	0.7526	100.68	<.0001
Trt	1	0.1111	0.0887	-0.0628	0.2850	1.57	0.2105
Scale	1	0.4611	0.0314	0.4035	0.5269		

Note: The scale parameter was estimated by maximum likelihood.

The test statistic is $\hat{\theta}_1 = \hat{p}_t - \hat{p}_c = 0.1111$, and its associated standard error is

$$\sqrt{\mathrm{Var}(\hat{\theta}_1)} = \sqrt{\frac{\hat{p}_c(1-\hat{p}_c)}{54} + \frac{\hat{p}_t(1-\hat{p}_t)}{54}} = 0.0887$$

The following statements create and display (in Output 111.1.8) the data set that contains the parameter estimate at stage 1, $\hat{\theta}_1 = 0.1111$, and its associated standard error $\sqrt{\mathrm{Var}(\hat{\theta}_1)} = 0.0887$ which are used in the SEQTEST procedure:

```
data Parms_Count1;
   set Parms_Count1;
   if Parameter='Trt';
   _Scale_='MLE';
   _Stage_= 1;
   keep _Scale_ _Stage_ Parameter Estimate StdErr;
run;

proc print data=Parms_Count1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.1.8 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Parameter	Estimate	StdErr	_Scale_	_Stage_
1	Trt	0.1111	0.0887	MLE	1

The initial required sample sizes are derived with the proportions $p_c = 0.6$ and $p_t = 0.75$. If the observed proportions are different from these assumed values, or if the number of available patients is different from the study plan in one of the stages, then the information level that corresponds to the test statistic is estimated from

$$I_k = \frac{1}{\mathrm{Var}(\hat{\theta}_k)}$$

The following statements invoke the SEQTEST procedure and test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Count
             Parms(Testvar=Trt)=Parms_Count1
             infoadj=none
             errspendmin=0.001
             boundaryscale=mle
             errspend
             plots=errspend
             ;
   ods output Test=Test_Count1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The PARMS=PARMS_COUNT1 option specifies the input data set PARMS_COUNT1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=TRT option identifies the test variable TRT in the data set. The INFOADJ=NONE option maintains the information levels at future interim stages (2 and 3) as provided in the BOUNDARY= data set. The BOUNDARYSCALE=MLE option displays the output boundaries in terms of the MLE scale.

The O'Brien-Fleming design is conservative in early stages and might not be desirable in a clinical trial. The ERRSPENDMIN=0.001 option specifies the minimum error spending at each stage to be 0.001, and it might increase the corresponding nominal p-value in early stages for the trial. The BOUNDARYSCALE=MLE option uses the MLE scale to display test statistics in the boundary table and boundary plots.

The ODS OUTPUT statement with the TEST=TEST_COUNT1 option creates an output data set named TEST_COUNT1 which contains the updated boundary information for the test at stage 1. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.1.9 displays design specifications. The derived statistics, such as the overall α and β levels, are derived from the specified maximum information and boundary values in the BOUNDARY= data set. Note that with a minor change in the information level at stage 1, the power also changes slightly from the design provided in the BOUNDARY= data set.

Output 111.1.9 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_COUNT
Data Set	WORK.PARMS_COUNT1
Statistic Distribution	Normal
Boundary Scale	MLE
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Number of Stages	4
Alpha	0.025
Beta	0.10147
Power	0.89853
Max Information (Percent of Fixed Sample)	108.2301
Max Information	502.834283
Null Ref ASN (Percent of Fixed Sample)	61.09917
Alt Ref ASN (Percent of Fixed Sample)	73.9745

With the ERRSPEND option, the "Error Spending Information" table in Output 111.1.10 displays cumulative error spending at each stage for each boundary. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the Type I error level $\alpha = 0.025$ is maintained. Furthermore, with the ERRSPENDMIN=0.001 option, the α spending at each stage is greater than or equal to 0.001.

Output 111.1.10 Error Spending Information

Error Spending Information				
			Cumulative Error Spending	
	Information Level		Upper	
Stage	Proportion	Actual	Beta	Alpha
1	0.2525	126.9871	0.00308	0.00100
2	0.5000	251.4171	0.02653	0.00343
3	0.7500	377.1257	0.06456	0.01254
4	1.0000	502.8343	0.10147	0.02500

With the PLOTS=ERRSPEND option, the procedure displays a plot of error spending for each boundary, as shown in Output 111.1.11. The error spending values in the "Error Spending Information" table in Output 111.1.10 are displayed in the plot.

Output 111.1.11 Error Spending Plot

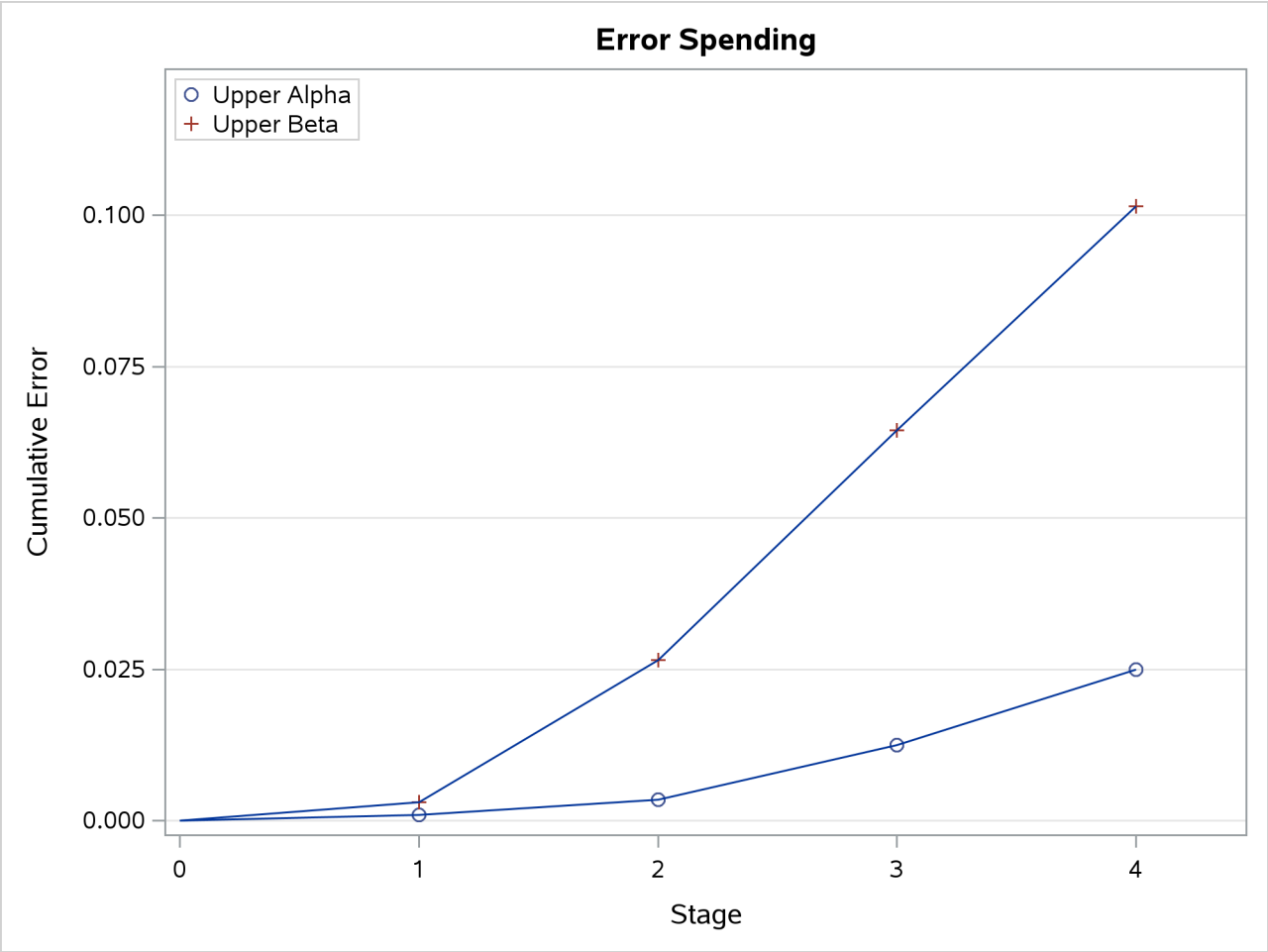

The "Test Information" table in Output 111.1.12 displays the boundary values for the design, test statistic, and resulting action at each stage. With the BOUNDARYSCALE=MLE option, the maximum likelihood estimator scale is used for the test statistic and boundary values. The table shows that the test statistic 0.1111 is between the upper α and β boundaries, so the trial continues to the next stage.

Output 111.1.12 Sequential Test

Test Information (MLE Scale)
Null Reference = 0

	Information Level		Alternative Reference	Boundary Values		Test	
				Upper		Trt	
Stage	Proportion	Actual	Upper	Beta	Alpha	Estimate	Action
1	0.2525	126.9871	0.15000	-0.09306	0.27423	0.11111	Continue
2	0.5000	251.4171	0.15000	0.02674	0.17527	.	
3	0.7500	377.1257	0.15000	0.06805	0.11792	.	
4	1.0000	502.8343	0.15000	0.08875	0.08875	.	

The information level at stage 1 is derived from the standard error,

$$I_1 = \frac{1}{\mathrm{Var}(\hat{\theta}_1)} = \frac{1}{(\mathrm{s.e.}(\hat{\theta}_1))^2} = 126.987$$

By default (or equivalently if you specify PLOTS=TEST), the "Test Plot" graph displays boundary values of the design and the test statistic at stage 1, as shown in Output 111.1.13. It also shows that the observed statistic is in the continuation region.

Output 111.1.13 Sequential Test Plot

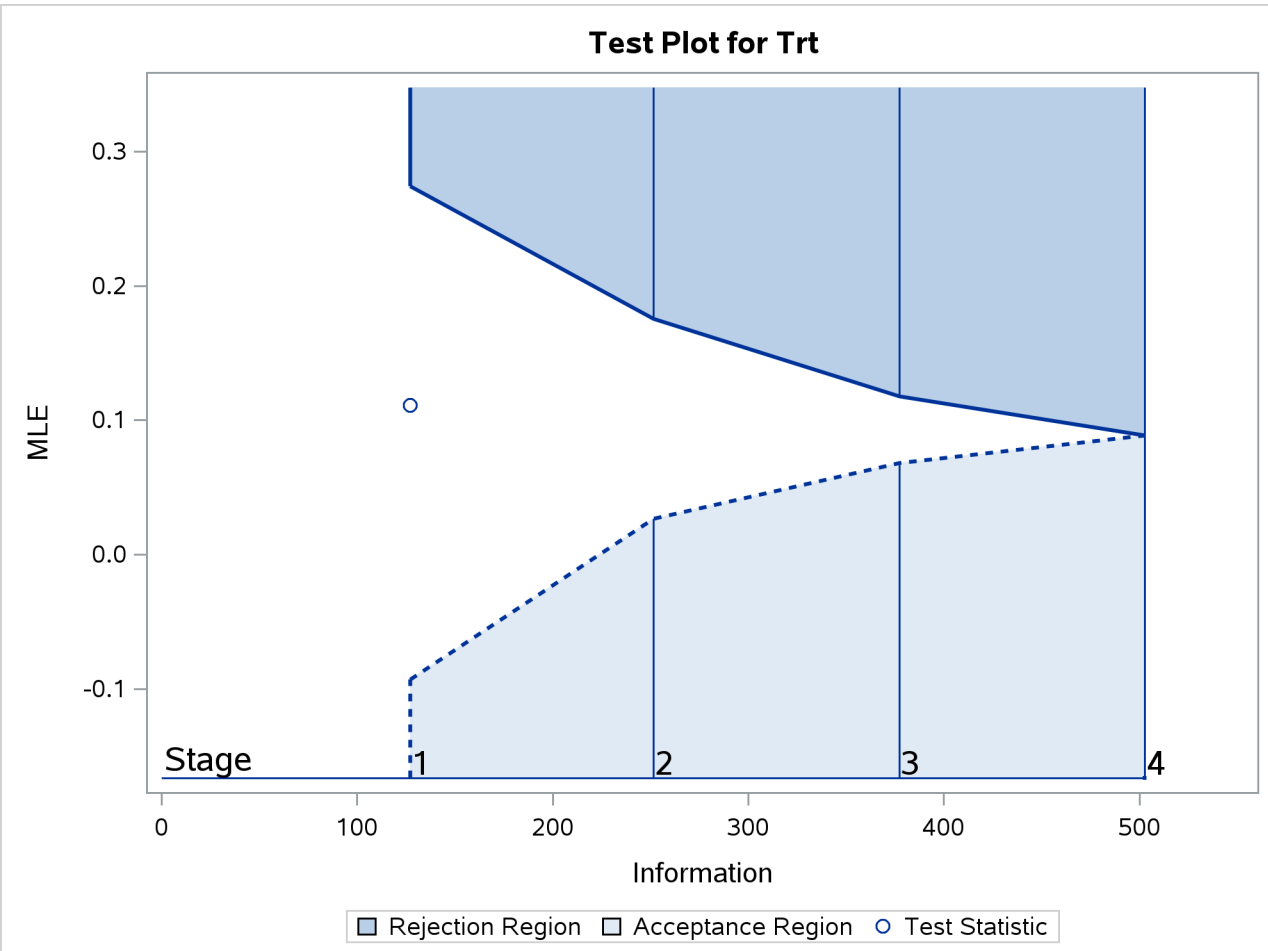

The observed information level at stage 1, $I_1 = 126.987$, is slightly larger than the target information level at the design. If an observed information level in the study is substantially different from its target level in the design, then the sample sizes should be adjusted in the subsequent stages to achieve the target information levels.

Suppose the trial continues to the next stage, and 108 patients are available in each group at stage 2. The data set COUNT_2 contains these 216 patients.

The following statements use the GENMOD procedure to estimate the treatment effect at stage 2:

```
proc genmod data=Count_2;
   model Resp= Trt;
   ods output ParameterEstimates=Parms_Count2;
run;
```

The following statements create the parameter estimate at stage 2, $\hat{\theta}_2 = \hat{p}_2 - \hat{p}_1 = 0.1759$, and its associated standard error $\sqrt{\mathrm{Var}(\hat{\theta}_2)} = 0.0623$ into a test data set:

```
data Parms_Count2;
   set Parms_Count2;
   if Parameter='Trt';
   _Scale_='MLE';
   _Stage_= 2;
   keep _Scale_ _Stage_ Parameter Estimate StdErr;
run;

proc print data=Parms_Count2;
   title 'Statistics Computed at Stage 2';
run;
```

Output 111.1.14 displays the test statistics at stage 2.

Output 111.1.14 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	Parameter	Estimate	StdErr	_Scale_	_Stage_
1	Trt	0.1759	0.0623	MLE	2

The following statements invoke the SEQTEST procedure and test for early stopping at stage 2:

```
ods graphics on;
proc seqtest Boundary=Test_Count1
             Parms(Testvar=Trt)=Parms_Count2
             infoadj=none
             boundaryscale=mle
             ;
   ods output Test=Test_Count2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=TEST_COUNT2 option creates an output data set named TEST_COUNT2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Test Information" table in Output 111.1.15 displays the boundary values, test statistic, and resulting action at each stage. The table shows that the test statistic 0.17593 is larger than the corresponding upper alpha boundary value, so the trial stops to reject the hypothesis.

Output 111.1.15 Sequential Test

The SEQTEST Procedure

Test Information (MLE Scale) Null Reference = 0							
	Information Level		Alternative Reference	Boundary Values		Test Trt	
Stage	Proportion	Actual	Upper	Upper Beta	Alpha	Estimate	Action
1	0.2525	126.9871	0.15000	-0.09306	0.27423	0.11111	Continue
2	0.5122	257.5571	0.15000	0.03019	0.17001	0.17593	Reject Null
3	0.7500	377.1257	0.15000	0.06783	0.11826	.	
4	1.0000	502.8343	0.15000	0.08878	0.08878	.	

With ODS Graphics enabled, the "Test Plot" is displayed, as shown in Output 111.1.16. The plot displays boundary values of the design and the test statistics at the first two stages. As expected, the test statistic at stage 2 is in the "Upper Rejection Region" above the upper alpha boundary.

Output 111.1.16 Sequential Test Plot

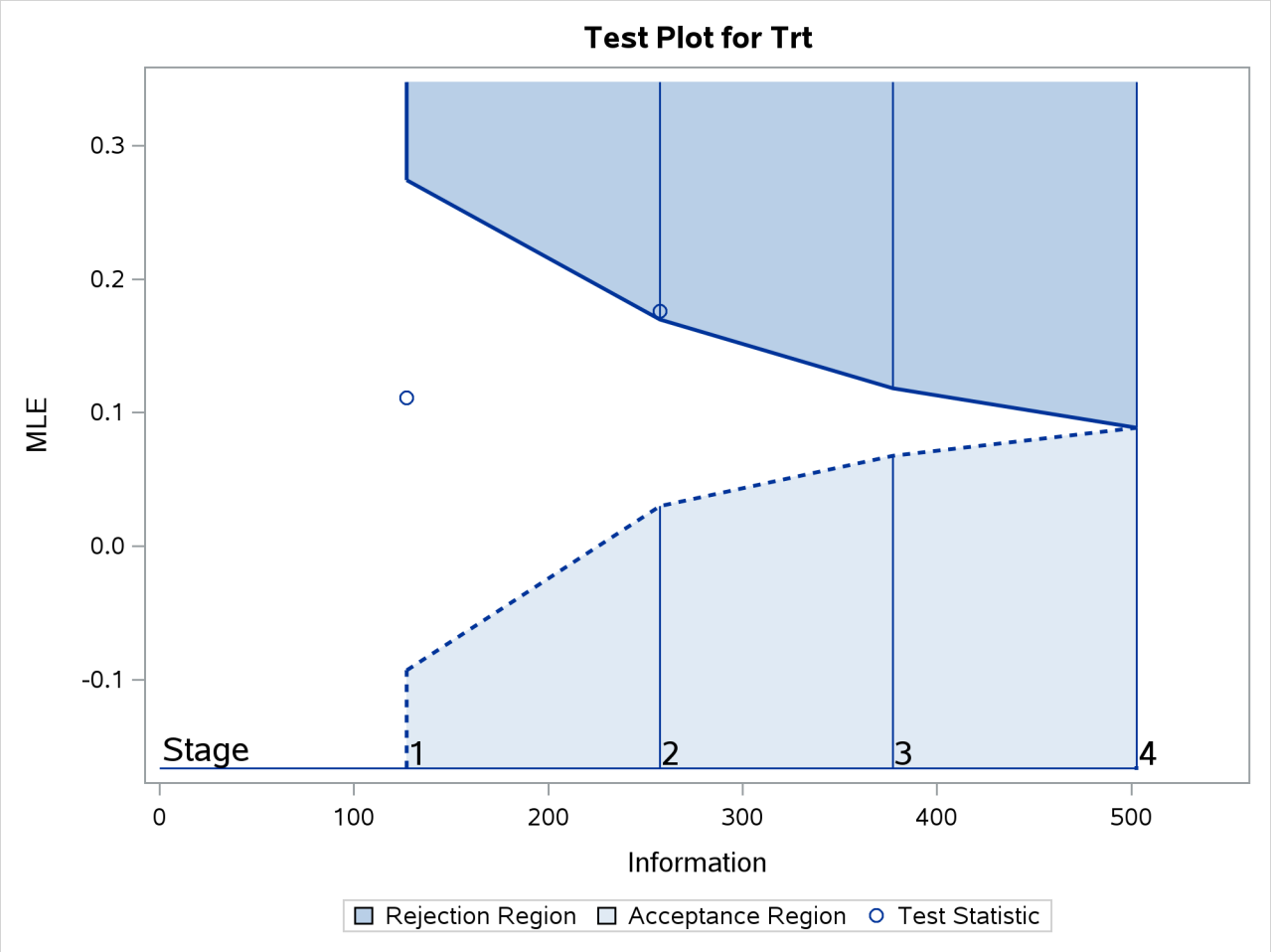

After a trial is stopped, the "Parameter Estimates" table in Output 111.1.17 displays the stopping stage and the

maximum likelihood estimate of the parameter. It also displays the p-value, median estimate, and confidence limits for the parameter that correspond to the observed statistic by using the specified sample space ordering.

Output 111.1.17 Parameter Estimates

Parameter Estimates Stagewise Ordering					
Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	Lower 95% CL
Trt	2	0.175926	0.0031	0.174462	0.07059

The MLE statistic at the stopping stage is the maximum likelihood estimate of the parameter and is biased. The computation of p-value, unbiased median estimate, and confidence limits depends on the ordering of the sample space (k, z), where k is the stage number and z is the observed standardized Z statistic. By default (or equivalently if you specify ORDER=STAGEWISE), the stagewise ordering that uses counterclockwise ordering around the continuation region is used to compute the p-value, unbiased median estimate, and confidence limits. As expected, the p-value is less than 0.025, and the confidence interval does not contain the null reference zero. With the stagewise ordering, the p-value is computed as

$$P_{\theta=0}\left(Z_1 > a_1\right) + P_{\theta=0}\left(Z_2 > z_2 \mid b_1 < Z_1 < a_1\right)$$

where z_2 is the observed standardized Z statistic at stage 2, Z_1 is the standardized normal variate at stage 1, Z_2 is the standardized normal variate at stage 2, and a_1 and b_1 are the stage 1 upper rejection and acceptance boundary values, respectively.

See the section "Available Sample Space Orderings in a Sequential Test" on page 9369 for a detailed description of the stagewise ordering.

Example 111.2: Testing an Effect in a Regression Model

(View the complete code for this example (seqtex2.sas) in the example repository.)

This example demonstrates a two-sided group sequential test that uses an error spending design with early stopping to reject the null hypothesis. A study is conducted to examine the effects of Age (years), Weight (kg), RunTime (time in minutes to run 1.5 miles), RunPulse (heart rate while running), and MaxPulse (maximum heart rate recorded while running) on Oxygen (oxygen intake rate, ml per kg body weight per minute). The primary interest is whether oxygen intake rate is associated with weight.

The hypothesis is tested using the following linear model:

$$\text{Oxygen} = \text{Age} + \text{Weight} + \text{RunTime} + \text{RunPulse} + \text{MaxPulse}$$

The null hypothesis is $H_0 : \beta_w = 0$, where β_w is the regression parameter for the variable Weight. Suppose that $\beta_w = 0.10$ is the reference improvement that should be detected at a 0.90 level. Then the maximum information I_X can be derived in the SEQDESIGN procedure.

Following the derivations in the section "Test for a Parameter in the Regression Model" on page 9204 in Chapter 110, "The SEQDESIGN Procedure," the required sample size can be derived from

$$N = I_X \, \frac{\sigma_y^2}{\left(1 - r_x^2\right) \sigma_x^2}$$

where σ_y^2 is the variance of the response variable in the regression model, r_x^2 is the proportion of variance of Weight explained by other covariates, and σ_x^2 is the variance of Weight.

Further suppose that from past experience, $\sigma_y^2 = 5$, $r_x^2 = 0.10$, and $\sigma_x^2 = 64$. Then the required sample size can be derived using the SAMPLESIZE statement in the SEQDESIGN procedure.

The following statements invoke the SEQDESIGN procedure and request a three-stage group sequential design for normally distributed data to test the null hypothesis of a regression parameter $H_0 : \beta_w = 0$ against the alternative $H_1 : \beta_w \neq 0$:

```
ods graphics on;
proc seqdesign altref=0.10;
   OBFErrorFunction: design method=errfuncobf
                            nstages=3
                            info=cum(2 3 4)
                            ;
   samplesize model=reg(variance=5 xvariance=64 xrsquare=0.10);
   ods output Boundary=Bnd_Fit;
run;
```

By default (or equivalently if you specify ALPHA=0.05 and BETA=0.10), the procedure uses a Type I error probability 0.05 and a Type II error probability 0.10. The ALTREF=0.10 option specifies a power of $1 - \beta = 0.90$ at the alternative hypothesis $H_1 : \beta_w = \pm 0.10$. The INFO=CUM(2 3 4) option specifies that the study perform the first interim analysis with information proportion $2/4 = 0.5$—that is, after half of the total observations are collected.

The ODS OUTPUT statement with the BOUNDARY=BND_FIT option creates an output data set named BND_FIT which contains the resulting boundary information for the subsequent sequential tests.

The "Design Information" table in Output 111.2.1 displays design specifications and derived statistics. Since the alternative reference is specified, the maximum information is derived.

Output 111.2.1 Design Information

The SEQDESIGN Procedure
Design: OBFErrorFunction

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.1
Number of Stages	3
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	101.8276
Max Information	1069.948
Null Ref ASN (Percent of Fixed Sample)	101.2587
Alt Ref ASN (Percent of Fixed Sample)	77.81586

The "Boundary Information" table in Output 111.2.2 displays information level, alternative reference, and boundary values at each stage.

Output 111.2.2 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference		Boundary Values	
Stage	Proportion	Actual	N	Lower	Upper	Lower Alpha	Upper Alpha
1	0.5000	534.9738	46.43869	-2.31295	2.31295	-2.96259	2.96259
2	0.7500	802.4606	69.65804	-2.83277	2.83277	-2.35902	2.35902
3	1.0000	1069.948	92.87739	-3.27101	3.27101	-2.01409	2.01409

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.2.3. The boundary plot also displays the information level and critical value for the corresponding fixed-sample design. This design has characteristics of an O'Brien-Fleming design; the probability for early stopping is low, and the maximum information and critical values at the final stage are similar to those of the corresponding fixed-sample design.

Output 111.2.3 Boundary Plot

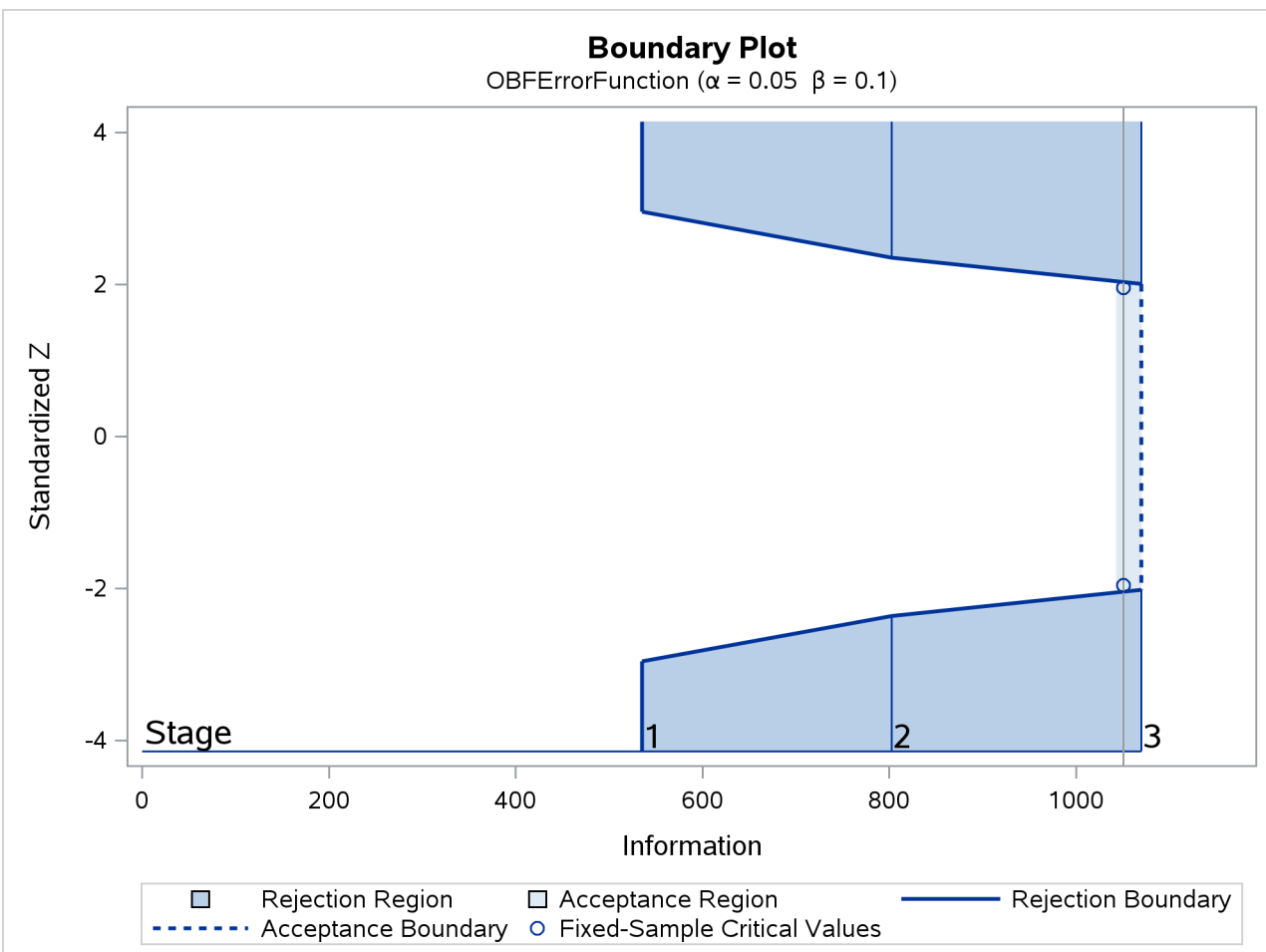

With the MODEL=REG option in the SAMPLESIZE statement, the "Sample Size Summary" table in Output 111.2.4 displays the parameters for the sample size computation.

Output 111.2.4 Required Sample Size Summary

Sample Size Summary	
Test	Reg Parameter
Parameter	0.1
Variance	5
X Variance	64
R Square (X)	0.1
Max Sample Size	92.87739
Expected Sample Size (Null Ref)	92.35845
Expected Sample Size (Alt Ref)	70.97617

The "Sample Sizes" table in Output 111.2.5 displays the required sample sizes for the group sequential clinical trial.

Output 111.2.5 Required Sample Sizes

Sample Sizes (N) Z Test for Regression Parameter				
	Fractional N		Ceiling N	
Stage	N	Information	N	Information
1	46.44	535.0	47	541.4
2	69.66	802.5	70	806.4
3	92.88	1069.9	93	1071.4

Thus, 47, 70, and 93 individuals are needed in stages 1, 2, and 3, respectively. Since the sample sizes are derived from estimated values of σ_y^2, r_x^2, and σ_x^2, the actual information levels might not achieve the target information levels. Thus, instead of specifying sample sizes in the protocol, you can specify the maximum information levels. Then if an actual information level is much less than the target level, you can increase the sample sizes for the remaining stages to achieve the desired information levels and power.

Suppose that 47 individuals are available at stage 1. Output 111.2.6 lists the first 10 observations of the trial data.

Output 111.2.6 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Oxygen	Age	Weight	RunTime	RunPulse	MaxPulse
1	54.5521	44	87.7676	11.6949	178.435	181.607
2	52.2821	40	75.4853	9.8872	184.433	183.667
3	62.1871	44	89.0638	8.7950	155.540	167.108
4	65.3269	42	67.7310	8.4577	162.926	173.877
5	59.9809	37	93.1902	9.3228	179.033	180.144
6	52.5588	47	75.9044	12.0385	177.753	175.033
7	51.7838	40	73.5422	11.6607	175.838	178.140
8	57.0024	43	81.2861	11.2219	160.963	171.770
9	48.0775	44	85.2290	13.1789	173.722	176.548
10	68.3357	38	80.2490	8.5066	171.824	184.011

The following statements use the REG procedure to estimate the slope β_w and its associated standard error at stage 1:

```
proc reg data=Fit_1;
   model Oxygen=Age Weight RunTime RunPulse MaxPulse;
   ods output ParameterEstimates=Parms_Fit1;
run;
```

The following statements create and display (in Output 111.2.7) the input data set that contains slope β_w and its associated standard error for the SEQTEST procedure:

```
data Parms_Fit1;
   set Parms_Fit1;
   if Variable='Weight';
   _Scale_='MLE';
   _Stage_= 1;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;

proc print data=Parms_Fit1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.2.7 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	Weight	0.03772	0.04345	MLE	1

The following statements invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Fit
             Parms(Testvar=Weight)=Parms_Fit1
             infoadj=prop
             errspendadj=errfuncobf
```

```
                order=lr
                stopprob
                ;
      ods output Test=Test_Fit1;
   run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1 (this data set was generated in the SEQDESIGN procedure). Recall that these boundary values were derived for the information levels specified with the INFO=CUM(2 3 4) option in the SEQDESIGN procedure. The PARMS=PARMS_FIT1 option specifies the input data set PARMS_FIT1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=WEIGHT option identifies the test variable WEIGHT in the data set.

If the computed information level for stage 1 is not the same as the value provided in the BOUNDARY= data set, the INFOADJ=PROP option (which is the default) proportionally adjusts the information levels at future interim stages from the levels provided in the BOUNDARY= data set. The ORDER=LR option uses the LR ordering to derive the p-value, the unbiased median estimate, and the confidence limits for the regression slope estimate. The ERRSPENDADJ=ERRFUNCOBF option adjusts the boundaries with the updated error spending values generated from an O'Brien-Fleming-type cumulative error spending function.

The ODS OUTPUT statement with the TEST=TEST_FIT1 option creates an output data set named TEST_FIT1 which contains the updated boundary information for the test at stage 1. The adjustment is needed because the observed information level is different from the information level in the BOUNDARY= data set. The data set TEST_FIT1 also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.2.8 displays the design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary values are modified for the new information levels to maintain the Type I α level. The maximum information remains the same as in the BOUNDARY= data set, but the derived Type II error probability β and power $1 - \beta$ are different because of the new information level.

Output 111.2.8 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_FIT
Data Set	WORK.PARMS_FIT1
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Number of Stages	3
Alpha	0.05
Beta	0.09994
Power	0.90006
Max Information (Percent of Fixed Sample)	101.8057
Max Information	1069.94751
Null Ref ASN (Percent of Fixed Sample)	101.2416
Alt Ref ASN (Percent of Fixed Sample)	77.87607

With the STOPPROB option, the "Expected Cumulative Stopping Probabilities" table in Output 111.2.9 displays the expected stopping stage and cumulative stopping probability to reject the null hypothesis at each stage under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and $c_i = 0, 0.5, 1, 1.5$ by default. You can specify other values for c_i with the CREF= option.

Output 111.2.9 Stopping Probabilities

Expected Cumulative Stopping Probabilities
Reference = CRef * (Alt Reference)

CRef	Expected Stopping Stage	Source	Stopping Probabilities Stage_1	Stage_2	Stage_3
0.0000	2.978	Reject Null	0.00289	0.01906	0.05000
0.5000	2.792	Reject Null	0.03373	0.17443	0.36566
1.0000	2.069	Reject Null	0.24884	0.68206	0.90006
1.5000	1.348	Reject Null	0.68172	0.97032	0.99820

The "Test Information" table in Output 111.2.10 displays the boundary values for the test statistic. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), these statistics are displayed with the standardized Z scale. The information level at stage 1 is derived from the standard error s_1 in the PARMS= data set,

$$I_1 = \frac{1}{s_1^2} = \frac{1}{0.043453^2} = 529.62$$

Output 111.2.10 Sequential Tests

Test Information (Standardized Z Scale)
Null Reference = 0

Stage	Information Level Proportion	Actual	Alternative Reference Lower	Upper	Boundary Values Lower Alpha	Upper Alpha	Test Weight Estimate	Action
1	0.4950	529.6232	-2.30135	2.30135	-2.97951	2.97951	0.86798	Continue
2	0.7475	799.7853	-2.82805	2.82805	-2.36291	2.36291	.	
3	1.0000	1069.948	-3.27101	3.27101	-2.01336	2.01336	.	

With the INFOADJ=PROP option (which is the default), the information level at stage 2 is derived proportionally from the observed information at stage 1 and the information levels in the BOUNDARY= data set. See the section "Boundary Adjustments for Information Levels" on page 9361 for details about how the adjusted information levels are computed.

At stage 1, the standardized Z statistic 0.86798 is between the lower α boundary –2.97951 and the upper α boundary 2.97951, so the trial continues to the next stage.

With ODS Graphics enabled, a boundary plot with test statistics is displayed, as shown in Output 111.2.11. As expected, the test statistic is in the continuation region between the lower and upper α boundaries.

Output 111.2.11 Sequential Test Plot

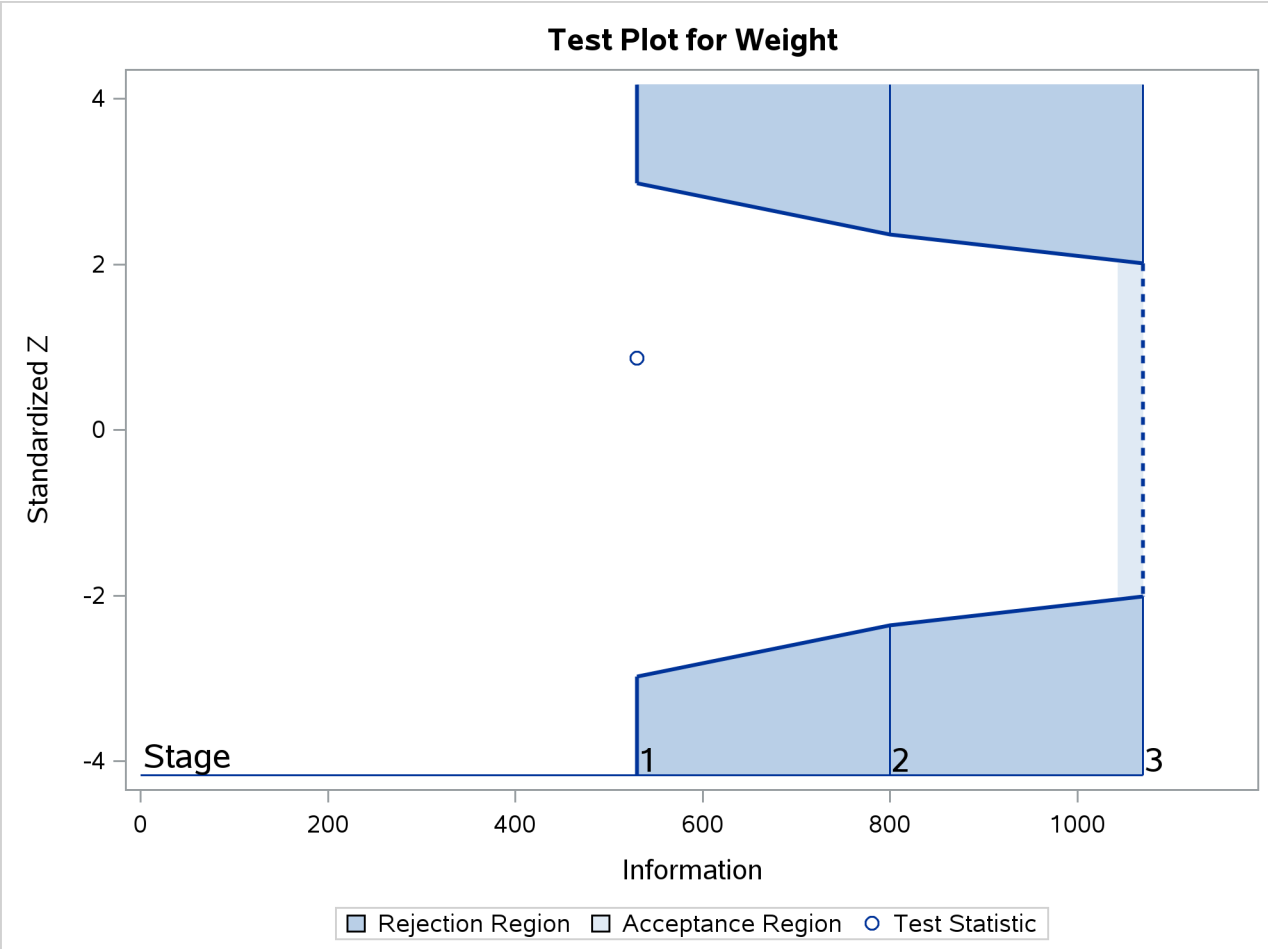

The following statements use the REG procedure to estimate the slope β_w and its associated standard error at stage 2:

```
proc reg data=Fit_2;
   model Oxygen=Age Weight RunTime RunPulse MaxPulse;
   ods output ParameterEstimates=Parms_Fit2;
run;
```

Note that the data set Fit_2 contains both the data from stage 1 and the data from stage 2.

The following statements create and display (in Output 111.2.12) the input data set that contains slope β_w and its associated standard error at stage 2 for the SEQTEST procedure:

```
data Parms_Fit2;
   set Parms_Fit2;
   if Variable='Weight';
   _Scale_='MLE';
   _Stage_= 2;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;
```

```
proc print data=Parms_Fit2;
   title 'Statistics Computed at Stage 2';
run;
```

Output 111.2.12 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	Weight	0.02932	0.03520	MLE	2

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2:

```
proc seqtest Boundary=Test_Fit1
             Parms(Testvar=Weight)=Parms_Fit2
             infoadj=prop
             errspendadj=errfuncobf
             order=lr
             ;
   ods output Test=Test_Fit2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=TEST_FIT2 option creates an output data set named TEST_FIT2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

Since the data set PARMS_FIT2 does not contain the test information at stage 1, the information level at stage 1 in the TEST_FIT1 data set is used to generate boundary values for the test at stage 2.

Following the process at stage 1, the slope estimate is also between its corresponding lower and upper α boundary values, so the trial continues to the next stage.

The following statements use the REG procedure to estimate the slope β_w and its associated standard error at the final stage:

```
proc reg data=Fit_3;
   model Oxygen=Age Weight RunTime RunPulse MaxPulse;
   ods output ParameterEstimates=Parms_Fit3;
run;
```

The following statements create the input data set that contains slope β_w and its associated standard error at stage 3 for the SEQTEST procedure:

```
data Parms_Fit3;
   set Parms_Fit3;
   if Variable='Weight';
   _Scale_='MLE';
   _Stage_= 3;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;
```

The following statements print (in Output 111.2.13) the test statistics at stage 3:

```
proc print data=Parms_Fit3;
   title 'Statistics Computed at Stage 3';
run;
```

Output 111.2.13 Statistics Computed at Stage 3

Statistics Computed at Stage 3

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	Weight	0.02189	0.03028	MLE	3

The following statements invoke the SEQTEST procedure to test the hypothesis:

```
ods graphics on;
proc seqtest Boundary=Test_Fit2
             Parms(testvar=Weight)=Parms_Fit3
             errspendadj=errfuncobf
             order=lr
             pss
             plots=(asn power)
             ;
   ods output Test=Test_Fit3;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 3, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 3, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=TEST_FIT3 option creates an output data set named TEST_FIT3 which contains the updated boundary information for the test at stage 3.

The "Design Information" table in Output 111.2.14 displays design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary values are modified for the new information levels to maintain the Type I α level.

Output 111.2.14 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.TEST_FIT2
Data Set	WORK.PARMS_FIT3
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Number of Stages	3
Alpha	0.05
Beta	0.09514
Power	0.90486
Max Information (Percent of Fixed Sample)	102.0102
Max Information	1090.63724
Null Ref ASN (Percent of Fixed Sample)	101.4122
Alt Ref ASN (Percent of Fixed Sample)	77.22139

The maximum information is derived from the standard error associated with the slope estimate at the final stage and is larger than the target level. The derived Type II error probability β and power $1 - \beta$ are different because of the new information levels.

With the PSS option, the "Power and Expected Sample Sizes" table in Output 111.2.15 displays powers and expected mean sample sizes under various hypothetical references $\theta = c_i \theta_1$, where θ_1 is the alternative reference and $c_i = 0, 0.5, 1, 1.5$ by default. You can specify the c_i values with the CREF= option.

Output 111.2.15 Power and Expected Sample Size Information

Powers and Expected Sample Sizes Reference = CRef * (Alt Reference)		
		Sample Size
CRef	**Power**	**Percent Fixed-Sample**
0.0000	0.02500	101.4122
0.5000	0.37046	96.3754
1.0000	0.90486	77.2214
1.5000	0.99844	58.5301

With the PLOTS=ASN option, the procedure displays a plot of expected sample sizes under various hypothetical references, as shown in Output 111.2.16. By default, expected sample sizes under the hypotheses $\theta = c_i\ \theta_1$, $c_i = 0, 0.01, 0.02, \ldots, 1.50$, are displayed, where θ_1 is the alternative reference.

Output 111.2.16 ASN Plot

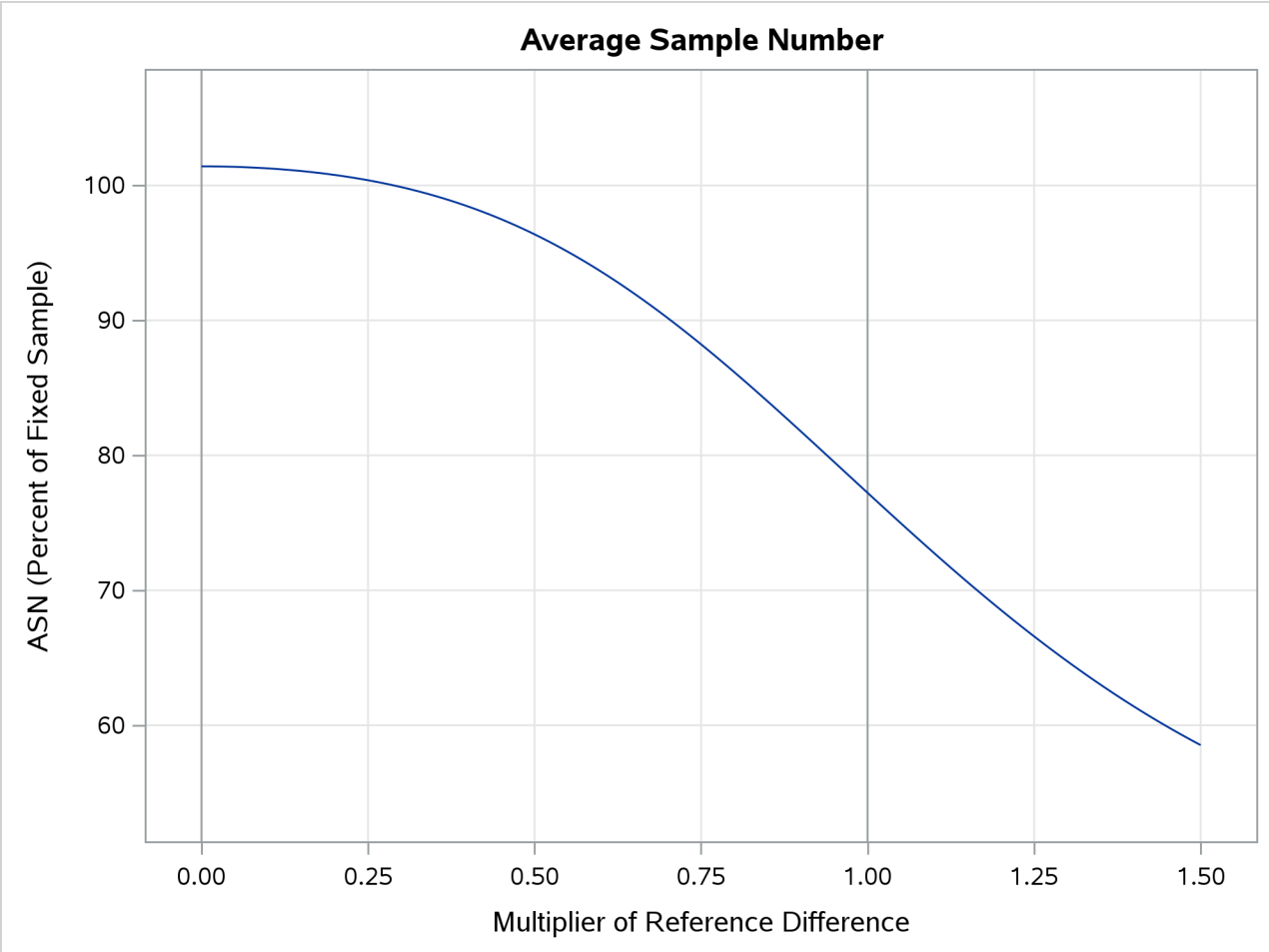

With the PLOTS=POWER option, the procedure displays a plot of powers under various hypothetical references, as shown in Output 111.2.17. By default, powers under hypothetical references $\theta = c_i\ \theta_1$ are displayed, where $c_i = 0, 0.01, 0.02, \ldots, 1.50$ by default. You can specify c_i values with the CREF= option. The c_i values are displayed on the horizontal axis.

Output 111.2.17 Power Plot

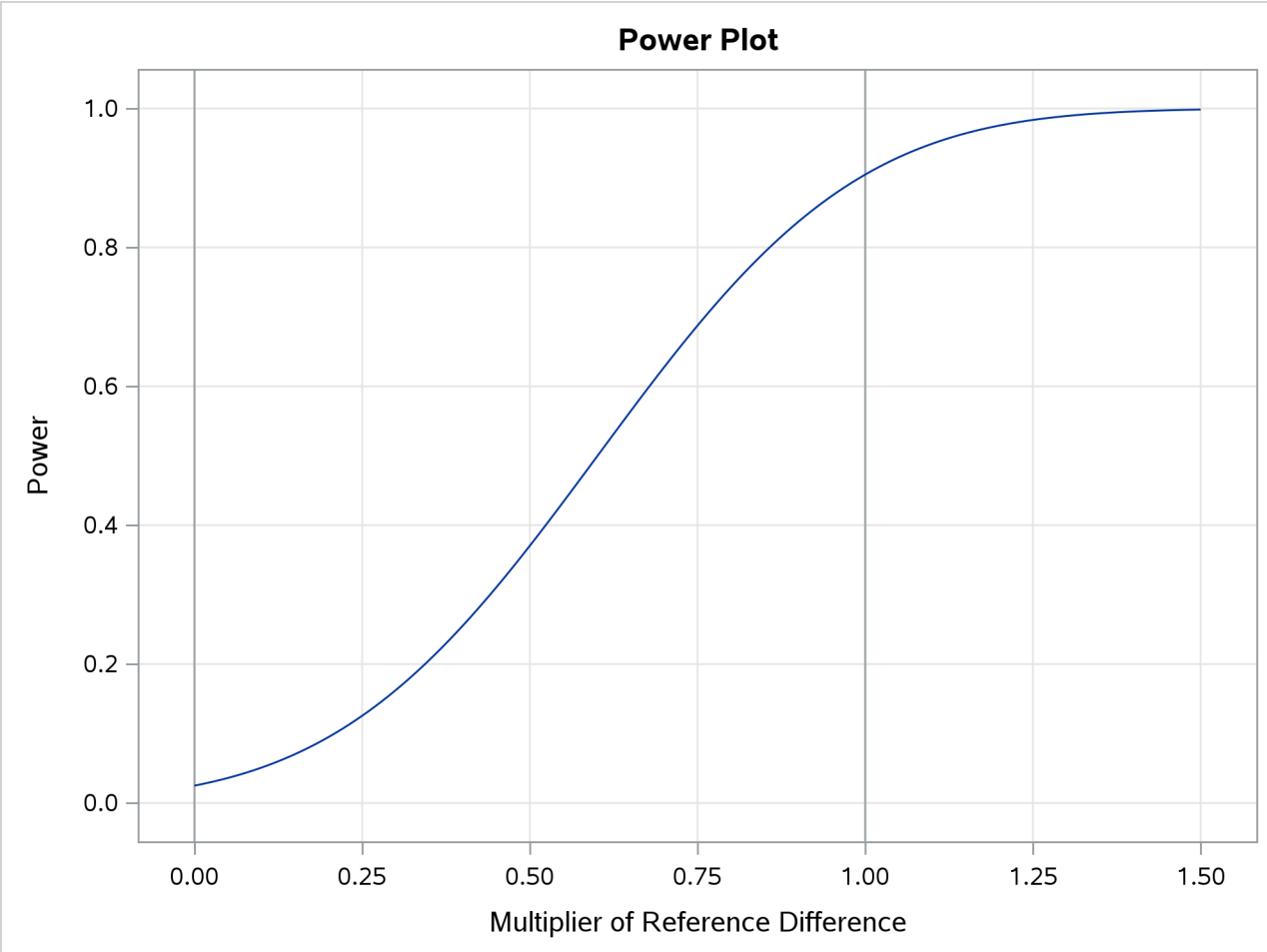

Under the null hypothesis, $c_i = 0$, the power is 0.025, which is the upper Type I error probability. Under the alternative hypothesis, $c_i = 1$, the power is 0.90486, which is one minus the Type II error probability.

The "Test Information" table in Output 111.2.18 displays the boundary values for the test statistic with the default standardized Z scale. The information level at the current stage is derived from the standard error for the current stage in the PARMS= data set. At stage 3, the standardized slope estimate 0.72284 is still between the lower and upper α boundary values. Since it is the final stage, the trial stops to accept the null hypothesis that the variable Weight has no effect on the oxygen intake rate after adjusting for other covariates.

Output 111.2.18 Sequential Tests

Test Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values		Test	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Upper Alpha	Weight Estimate	Action
1	0.4856	529.6232	-2.30135	2.30135	-2.97951	2.97951	0.86798	Continue
2	0.7401	807.1954	-2.84112	2.84112	-2.34945	2.34945	0.83305	Continue
3	1.0000	1090.637	-3.30248	3.30248	-2.01885	2.01885	0.72284	Accept Null

Since the data set FIT_3 contains the test information only at stage 3, the information levels at previous stages in the TEST_FIT2 data set are used to generate boundary values for the test.

With ODS Graphics enabled, a boundary plot with test statistics is displayed, as shown in Output 111.2.19. As expected, the test statistic is in the acceptance region between the lower and upper α boundaries at the final stage.

Output 111.2.19 Sequential Test Plot

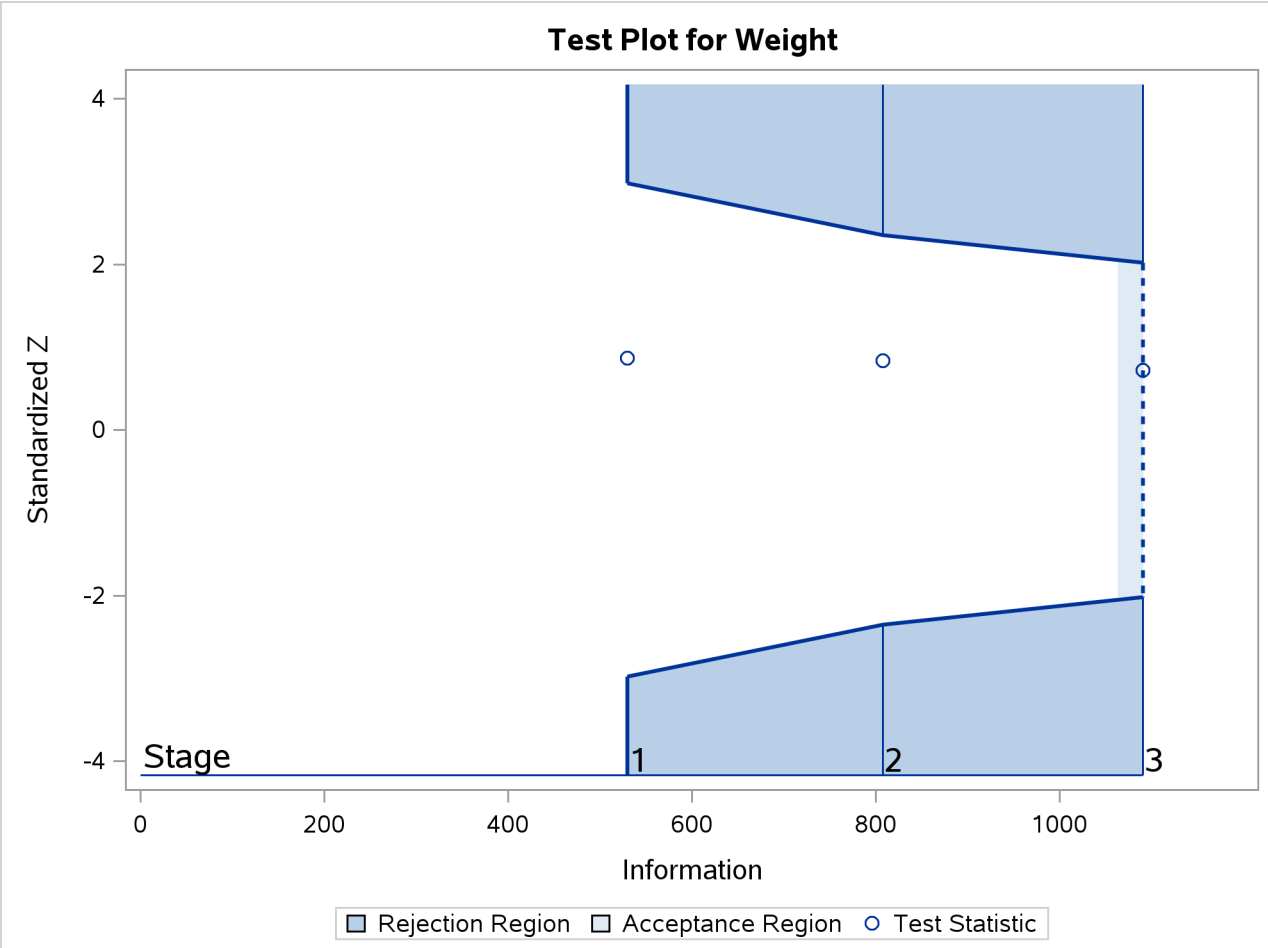

After a trial is stopped, the "Parameter Estimates" table in Output 111.2.20 displays the stopping stage, parameter estimate, unbiased median estimate, confidence limits, and the p-value under the null hypothesis $H_0 : \beta_w = 0$.

Output 111.2.20 Parameter Estimates

Parameter Estimates
LR Ordering

Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	95% Confidence Limits	
Weight	3	0.021888	0.4699	0.021884	-0.03747	0.08123

As expected, the p-value 0.4699 is not significant at the $\alpha = 0.05$ level, and the confidence interval does

contain the value zero. The p-value, unbiased median estimate, and confidence limits depend on the ordering of the sample space (k, z), where k is the stage number and z is the standardized Z statistic. With the specified LR ordering, the p-values are computed with the ordering $(k', z') \succ (k, z)$ if $z' > z$. See the section "Available Sample Space Orderings in a Sequential Test" on page 9369 for a detailed description of the LR ordering.

Example 111.3: Testing an Effect with Early Stopping to Accept H_0

(View the complete code for this example (seqtex3.sas) in the example repository.)

This example demonstrates a two-sided group sequential test that uses an error spending design with early stopping to accept the null hypothesis H_0. The example is similar to Example 111.2 but with early stopping to accept H_0.

A study is conducted to examine the effects of Age (years), Weight (kg), RunTime (time in minutes to run 1.5 miles), RunPulse (heart rate while running), and MaxPulse (maximum heart rate recorded while running) on Oxygen (oxygen intake rate, ml per kg body weight per minute). The primary interest is whether oxygen intake rate is associated with weight.

The hypothesis is tested using the following linear model:

$$\text{Oxygen} = \text{Age} + \text{Weight} + \text{RunTime} + \text{RunPulse} + \text{MaxPulse}$$

The null hypothesis is $H_0 : \beta_w = 0$, where β_w is the regression parameter for the variable Weight. Suppose that $\beta_w = 0.10$ is the reference improvement that should be detected at a 0.90 level. Then the maximum information I_X can be derived in the SEQDESIGN procedure.

Following the derivations in the section "Test for a Parameter in the Regression Model" on page 9204 in Chapter 110, "The SEQDESIGN Procedure," the required sample size can be derived from

$$N = I_X \, \frac{\sigma_y^2}{(1 - r_x^2)\, \sigma_x^2}$$

where σ_y^2 is the variance of the response variable in the regression model, r_x^2 is the proportion of variance of Weight explained by other covariates, and σ_x^2 is the variance of Weight.

Further suppose that from past experience, $\sigma_y^2 = 5$, $r_x^2 = 0.10$, and $\sigma_x^2 = 64$. Then the required sample size can be derived using the SAMPLESIZE statement in the SEQDESIGN procedure.

The following statements invoke the SEQDESIGN procedure and request a three-stage group sequential design for normally distributed data to test the null hypothesis of a regression parameter $H_0 : \beta_w = 0$ against the alternative $H_1 : \beta_w \neq 0$:

```
ods graphics on;
proc seqdesign altref=0.10;
   OBFErrorFunction: design method=errfuncgamma
                            stop=accept
                            nstages=3
                            info=cum(2 3 4);

   samplesize model=reg( variance=5 xvariance=64 xrsquare=0.10);
```

```
    ods output Boundary=Bnd_Fit;
  run;
```

By default (or equivalently if you specify ALPHA=0.05 and BETA=0.10), the procedure uses a Type I error probability 0.05 and a Type II error probability 0.10. The ALTREF=0.10 option specifies a power of $1 - \beta = 0.90$ at the alternative hypothesis $H_1 : \beta_w = \pm 0.10$. The INFO=CUM(2 3 4) option specifies that the study perform the first interim analysis with information proportion $2/4 = 0.5$—that is, after half of the total observations are collected.

The ODS OUTPUT statement with the BOUNDARY=BND_FIT option creates an output data set named BND_FIT which contains the resulting boundary information for the subsequent sequential tests.

The "Design Information" table in Output 111.3.1 displays design specifications and derived statistics. Since the alternative reference is specified, the maximum information is derived.

Output 111.3.1 Error Spending Design Information

The SEQDESIGN Procedure
Design: OBFErrorFunction

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Accept Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.1
Number of Stages	3
Alpha	0.05
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	103.9245
Max Information	1091.972
Null Ref ASN (Percent of Fixed Sample)	75.00521
Alt Ref ASN (Percent of Fixed Sample)	101.8099

The "Boundary Information" table in Output 111.3.2 displays information level, alternative reference, and boundary values at each stage.

Output 111.3.2 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference		Boundary Values	
						Lower	Upper
Stage	Proportion	Actual	N	Lower	Upper	Beta	Beta
1	0.5000	545.9862	47.39463	-2.33663	2.33663	-0.44937	0.44937
2	0.7500	818.9792	71.09195	-2.86178	2.86178	-1.13583	1.13583
3	1.0000	1091.972	94.78926	-3.30450	3.30450	-1.91428	1.91428

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.3.3. The boundary plot also displays the information level and critical value for the corresponding fixed-sample design.

Output 111.3.3 Boundary Plot

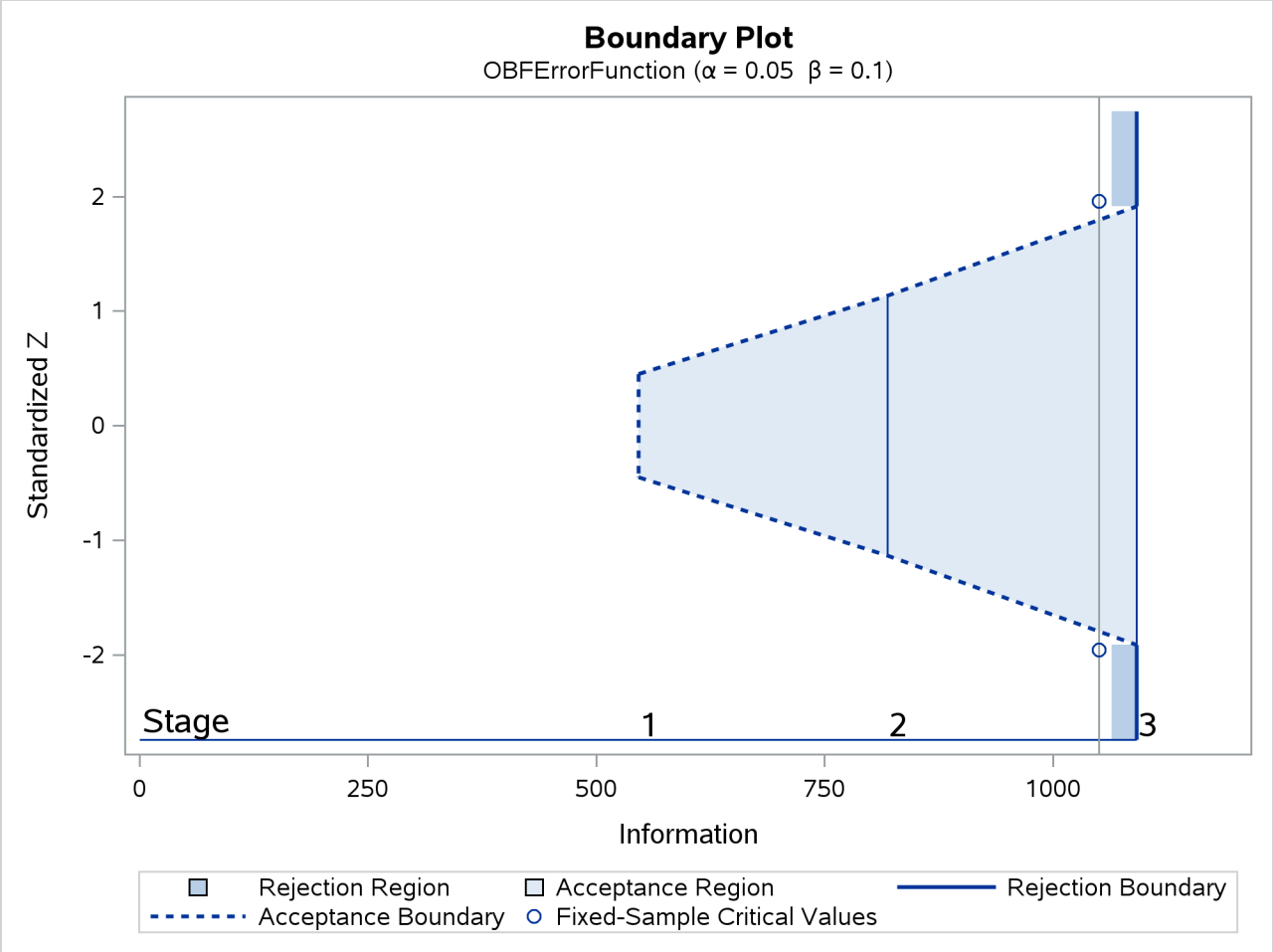

With the MODEL=REG option in the SAMPLESIZE statement, the "Sample Size Summary" table in Output 111.3.4 displays the parameters for the sample size computation.

Output 111.3.4 Required Sample Size Summary

Sample Size Summary	
Test	Reg Parameter
Parameter	0.1
Variance	5
X Variance	64
R Square (X)	0.1
Max Sample Size	94.78926
Expected Sample Size (Null Ref)	68.41207
Expected Sample Size (Alt Ref)	92.86057

The "Sample Sizes" table in Output 111.3.5 displays the required sample sizes for the group sequential clinical trial.

Output 111.3.5 Required Sample Sizes

Sample Sizes (N) Z Test for Regression Parameter				
	Fractional N		Ceiling N	
Stage	N	Information	N	Information
1	47.39	546.0	48	553.0
2	71.09	819.0	72	829.4
3	94.79	1092.0	95	1094.4

Thus, 48, 72, and 95 individuals are needed in stages 1, 2, and 3, respectively. Since the sample sizes are derived from estimated values of σ_y^2, r_x^2, and σ_x^2, the actual information levels might not achieve the target information levels. Thus, instead of specifying sample sizes in the protocol, you can specify the maximum information levels. Then if an actual information level is much less than the target level, you can increase the sample sizes for the remaining stages to achieve the desired information levels and power.

Suppose that 48 individuals are available at stage 1. Output 111.3.6 lists the first 10 observations of the trial data.

Output 111.3.6 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Oxygen	Age	Weight	RunTime	RunPulse	MaxPulse
1	54.5521	44	87.7676	11.6949	178.435	181.607
2	52.2821	40	75.4853	9.8872	184.433	183.667
3	62.1871	44	89.0638	8.7950	155.540	167.108
4	65.3269	42	67.7310	8.4577	162.926	173.877
5	59.9809	37	93.1902	9.3228	179.033	180.144
6	52.5588	47	75.9044	12.0385	177.753	175.033
7	51.7838	40	73.5422	11.6607	175.838	178.140
8	57.0024	43	81.2861	11.2219	160.963	171.770
9	48.0775	44	85.2290	13.1789	173.722	176.548
10	68.3357	38	80.2490	8.5066	171.824	184.011

The following statements use the REG procedure to estimate the slope β_w and its associated standard error at stage 1:

```
proc reg data=Fit_1;
   model Oxygen=Age Weight RunTime RunPulse MaxPulse;
   ods output ParameterEstimates=Parms_Fit1;
run;
```

The following statements create and display (in Output 111.3.7) the input data set that contains slope β_w and its associated standard error for the SEQTEST procedure:

```
data Parms_Fit1;
   set Parms_Fit1;
   if Variable='Weight';
   _Scale_='MLE';
   _Stage_= 1;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;

proc print data=Parms_Fit1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.3.7 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	Weight	0.04660	0.04308	MLE	1

The following statements invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Fit
             Parms(testvar=Weight)=Parms_Fit1
             infoadj=none
             errspendadj=errfuncgamma
             stopprob
             order=lr
             ;
   ods output Test=Test_Fit1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The PARMS=PARMS_FIT1 option specifies the input data set PARMS_FIT1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=WEIGHT option identifies the test variable WEIGHT in the data set. The INFOADJ=NONE option maintains the information level for stage 2 at the value provided in the BOUNDARY= data set.

The ORDER=LR option uses the LR ordering to derive the p-value, the unbiased median estimate, and the confidence limits for the regression slope estimate. The ERRSPENDADJ=ERRFUNCGAMMA option adjusts the boundaries with the updated error spending values generated from a gamma cumulative error spending function.

The ODS OUTPUT statement with the TEST=TEST_FIT1 option creates an output data set named TEST_FIT1 which contains the updated boundary information for the test at stage 1. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.3.8 displays the design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary values are modified for the new information levels to maintain the Type I α level. The maximum information remains the same as in the BOUNDARY= data set, but the derived Type II error probability β and power $1 - \beta$ are different because of the new information level.

Output 111.3.8 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_FIT
Data Set	WORK.PARMS_FIT1
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Accept Null
Number of Stages	3
Alpha	0.05
Beta	0.10007
Power	0.89993
Max Information (Percent of Fixed Sample)	103.9498
Max Information	1091.97232
Null Ref ASN (Percent of Fixed Sample)	75.15846
Alt Ref ASN (Percent of Fixed Sample)	101.8296

With the STOPPROB option, the "Expected Cumulative Stopping Probabilities" table in Output 111.3.9 displays the expected stopping stage and the cumulative stopping probability of accepting the null hypothesis at each stage under various hypothetical references $\theta = c_i\theta_1$, where θ_1 is the alternative reference and $c_i = 0, 0.5, 1, 1.5$ by default. You can specify other values for c_i with the CREF= option.

Output 111.3.9 Stopping Probabilities

Expected Cumulative Stopping Probabilities
Reference = CRef * (Alt Reference)

			Stopping Probabilities		
CRef	**Expected Stopping Stage**	**Source**	**Stage_1**	**Stage_2**	**Stage_3**
0.0000	1.895	Accept Null	0.33304	0.76607	0.95000
0.5000	2.409	Accept Null	0.17680	0.40947	0.62828
1.0000	2.918	Accept Null	0.02636	0.05453	0.10007
1.5000	2.997	Accept Null	0.00109	0.00166	0.00242

The "Test Information" table in Output 111.3.10 displays the boundary values for the test statistic. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), these statistics are displayed with the standardized Z scale. The information level at stage 1 is derived from the standard error s_1 in the PARMS= data set,

$$I_1 = \frac{1}{s_1^2} = \frac{1}{0.04308^2} = 538.8$$

Output 111.3.10 Sequential Tests

Test Information (Standardized Z Scale) Null Reference = 0								
	Information Level		Alternative Reference		Boundary Values		Test Weight	
Stage	Proportion	Actual	Lower	Upper	Lower Beta	Upper Beta	Estimate	Action
1	0.4934	538.7887	-2.32118	2.32118	-0.43033	0.43033	1.08174	Continue
2	0.7500	818.9792	-2.86178	2.86178	-1.13623	1.13623	.	
3	1.0000	1091.972	-3.30450	3.30450	-1.91431	1.91431	.	

At stage 1, the standardized Z statistic 1.08174 is greater than the upper β boundary 0.43033, so the trial continues to the next stage.

With ODS Graphics enabled, a boundary plot with test statistics is displayed, as shown in Output 111.3.11. As expected, the test statistic is in the continuation region.

Output 111.3.11 Sequential Test Plot

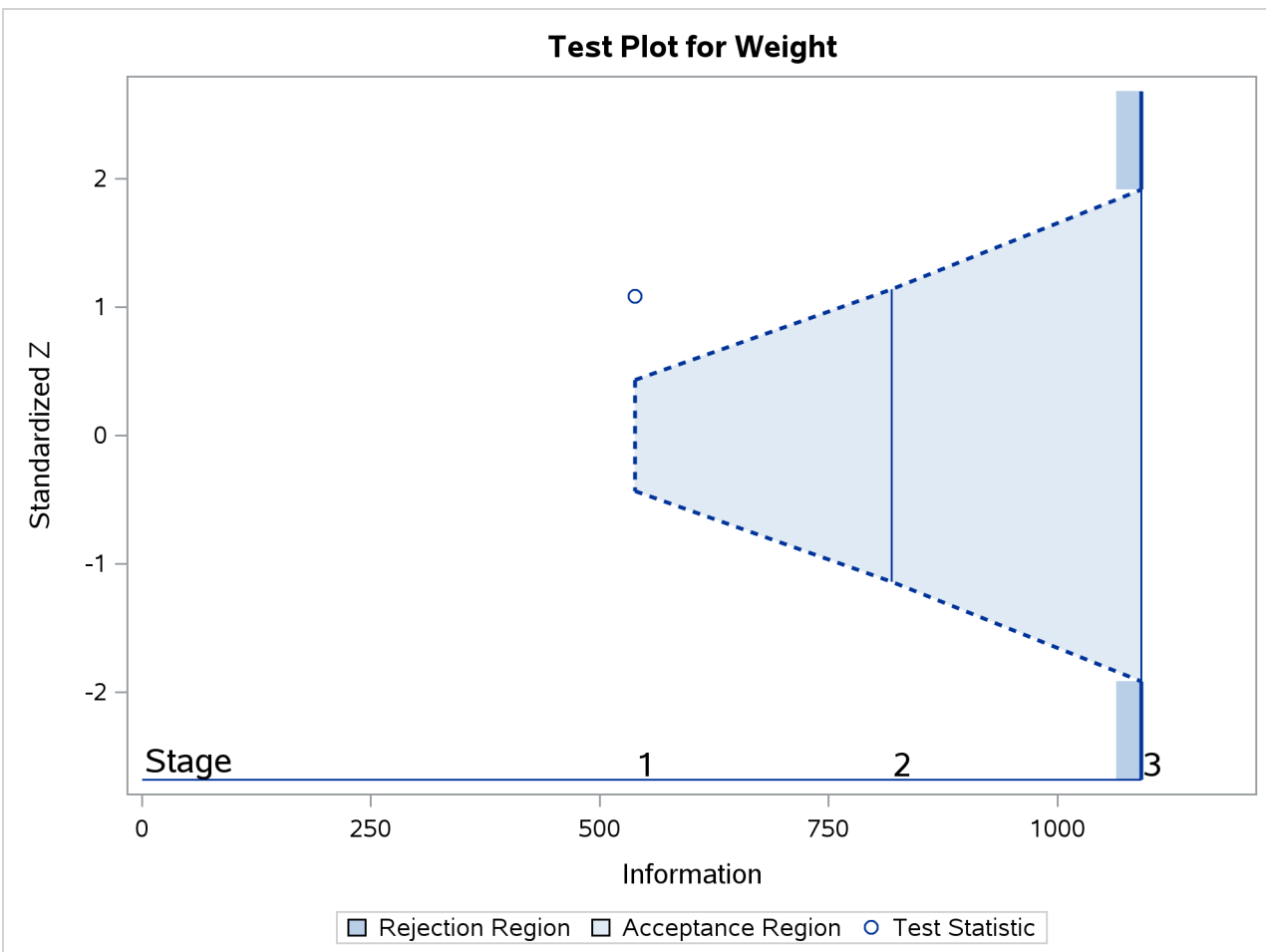

The following statements use the REG procedure to estimate the slope β_w and its associated standard error at stage 2:

```
proc reg data=Fit_2;
   model Oxygen=Age Weight RunTime RunPulse MaxPulse;
   ods output ParameterEstimates=Parms_Fit2;
run;
```

Note that the data set Fit_2 contains both the data from stage 1 and the data from stage 2,

The following statements create and display (in Output 111.3.12) the input data set that contains slope β_w and its associated standard error at stage 2 for the SEQTEST procedure:

```
data Parms_Fit2;
   set Parms_Fit2;
   if Variable='Weight';
   _Scale_='MLE';
   _Stage_= 2;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;


proc print data=Parms_Fit2;
   title 'Statistics Computed at Stage 2';
run;
```

Output 111.3.12 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	Weight	0.02925	0.03490	MLE	2

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2:

```
ods graphics on;
proc seqtest Boundary=Test_Fit1
             Parms(testvar=Weight)=Parms_Fit2
             errspendadj=errfuncgamma
             order=lr
             pss
             plots=(asn power)
             ;
   ods output Test=Test_Fit2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set.

Since the data set PARMS_FIT2 does not contain the test information at stage 1, the information level at stage 1 in the TEST_FIT1 data set is used to generate boundary values for the test.

The ORDER=LR option uses the LR ordering to derive the p-value, unbiased median estimate, and confidence limits for the regression slope estimate.

The ODS OUTPUT statement with the TEST=TEST_FIT2 option creates an output data set named TEST_FIT2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.3.13 displays design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary values are modified for the new information levels to maintain the Type I α level.

Output 111.3.13 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.TEST_FIT1
Data Set	WORK.PARMS_FIT2
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Accept Null
Number of Stages	3
Alpha	0.05
Beta	0.10009
Power	0.89991
Max Information (Percent of Fixed Sample)	103.9566
Max Information	1091.97232
Null Ref ASN (Percent of Fixed Sample)	75.18254
Alt Ref ASN (Percent of Fixed Sample)	101.8349

The derived Type II error probability β and power $1 - \beta$ are different because of the new information levels.

With the PSS option, the "Power and Expected Sample Sizes" table in Output 111.3.14 displays powers and expected mean sample sizes under various hypothetical references $\theta = c_i \theta_1$, where θ_1 is the alternative reference and $c_i = 0, 0.5, 1, 1.5$ are the default values in the CREF= option.

Output 111.3.14 Power and Expected Sample Size Information

Powers and Expected Sample Sizes Reference = CRef * (Alt Reference)		
		Sample Size
CRef	Power	Percent Fixed-Sample
0.0000	0.02500	75.1825
0.5000	0.37154	88.5975
1.0000	0.89991	101.8349
1.5000	0.99758	103.8843

With the PLOTS=ASN option, the procedure displays a plot of expected sample sizes under various hypothetical references, as shown in Output 111.3.15. By default, expected sample sizes under the hypotheses $\theta = c_i\, \theta_1$, $c_i = 0, 0.01, 0.02, \ldots, 1.50$, are displayed, where θ_1 is the alternative reference.

Output 111.3.15 ASN Plot

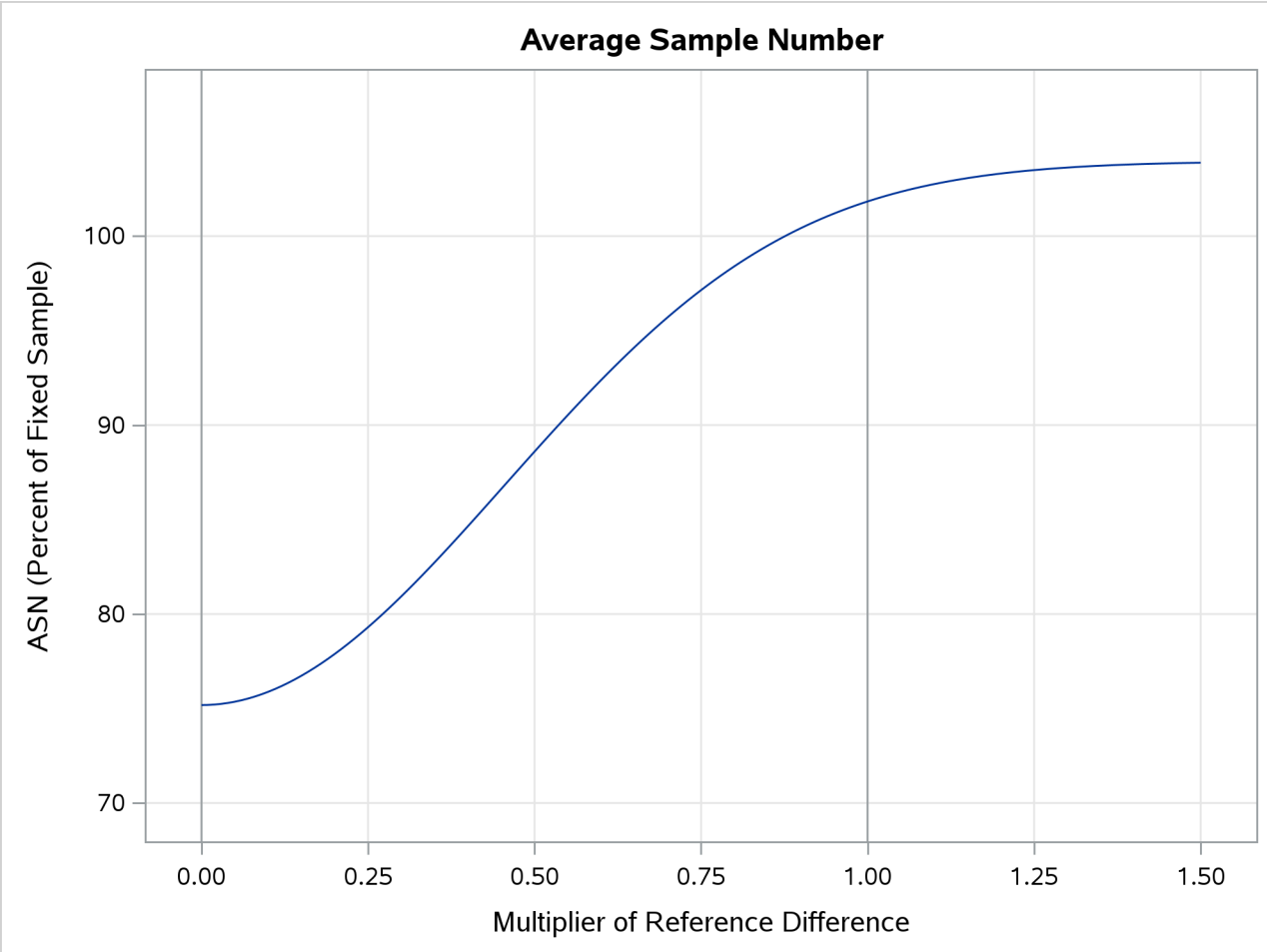

With the PLOTS=POWER option, the procedure displays a plot of the power curves under various hypothetical references for all designs simultaneously, as shown in Output 111.3.16. By default, powers under hypothetical references $\theta = c_i\ \theta_1$ are displayed, where $c_i = 0, 0.01, 0.02, \ldots, 1.50$ by default. You can specify c_i values with the CREF= option. The c_i values are displayed on the horizontal axis.

Output 111.3.16 Power Plot

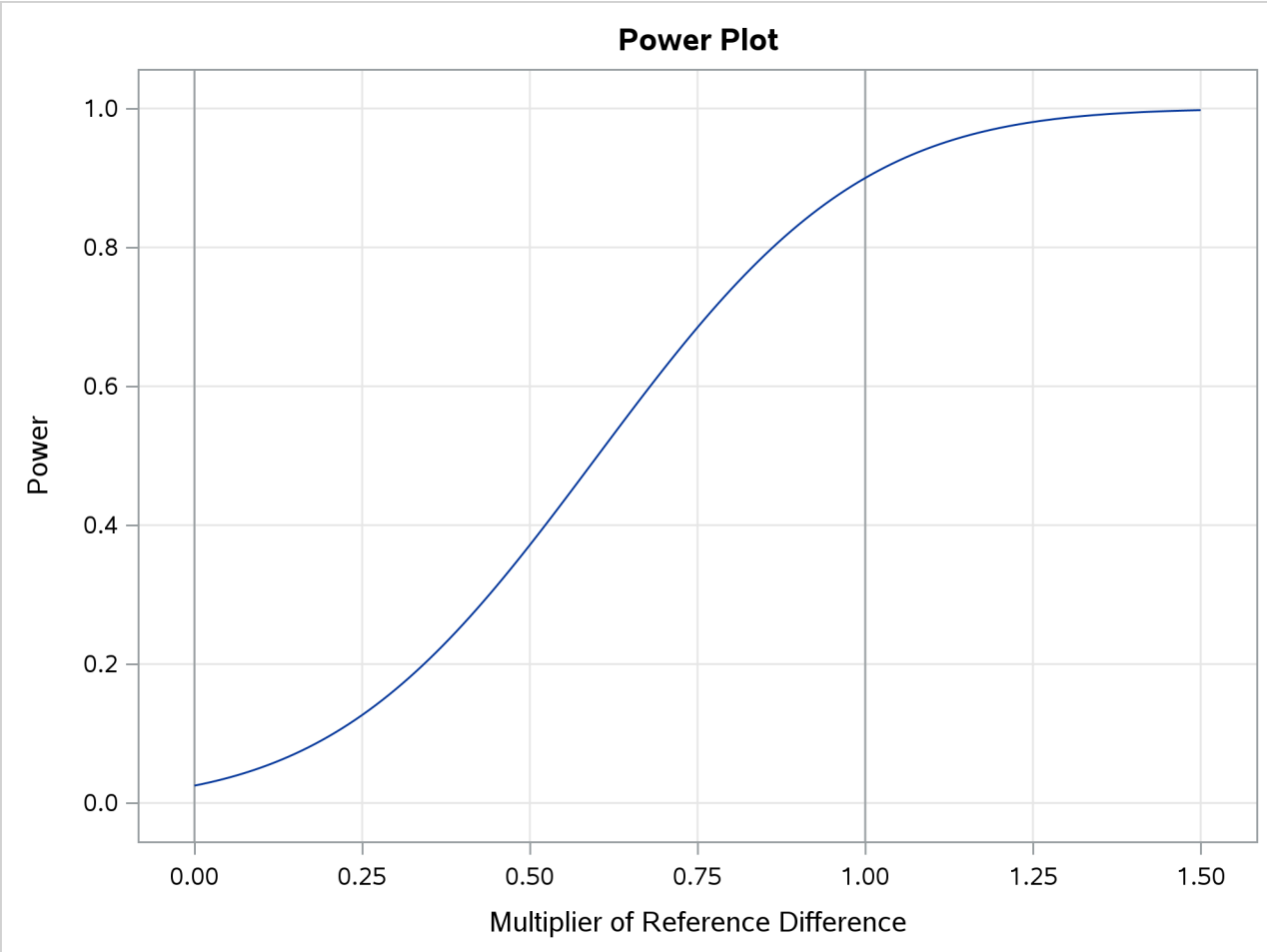

Under the null hypothesis, $c_i = 0$, the power is 0.025, which is the upper Type I error probability. Under the alternative hypothesis, $c_i = 1$, the power is 0.89991, which is one minus the Type II error probability, as displayed in the "Design Information" table in Output 111.3.13.

The "Test Information" table in Output 111.3.17 displays the boundary values for the test statistic with the default standardized Z scale. At stage 2, the standardized slope estimate 0.83805 is between the lower and upper β boundary values. The trial stops to accept the null hypothesis that the variable Weight has no effect on the oxygen intake rate after adjusting for other covariates.

Output 111.3.17 Sequential Tests

Test Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values		Test Weight	
Stage	Proportion	Actual	Lower	Upper	Lower Beta	Upper Beta	Estimate	Action
1	0.4934	538.7887	-2.32118	2.32118	-0.43033	0.43033	1.08174	Continue
2	0.7517	820.8509	-2.86505	2.86505	-1.14239	1.14239	0.83805	Accept Null
3	1.0000	1091.972	-3.30450	3.30450	-1.91408	1.91408	.	

Since the data set PARMS_FIT2 contains the test information only at stage 2, the information level at stage 1 in the TEST_FIT1 data set is used to generate boundary values for the test.

With ODS Graphics enabled, a boundary plot with test statistics is displayed, as shown in Output 111.3.18. As expected, the test statistic is in the acceptance region between the lower and upper α boundaries at the final stage.

Output 111.3.18 Sequential Test Plot

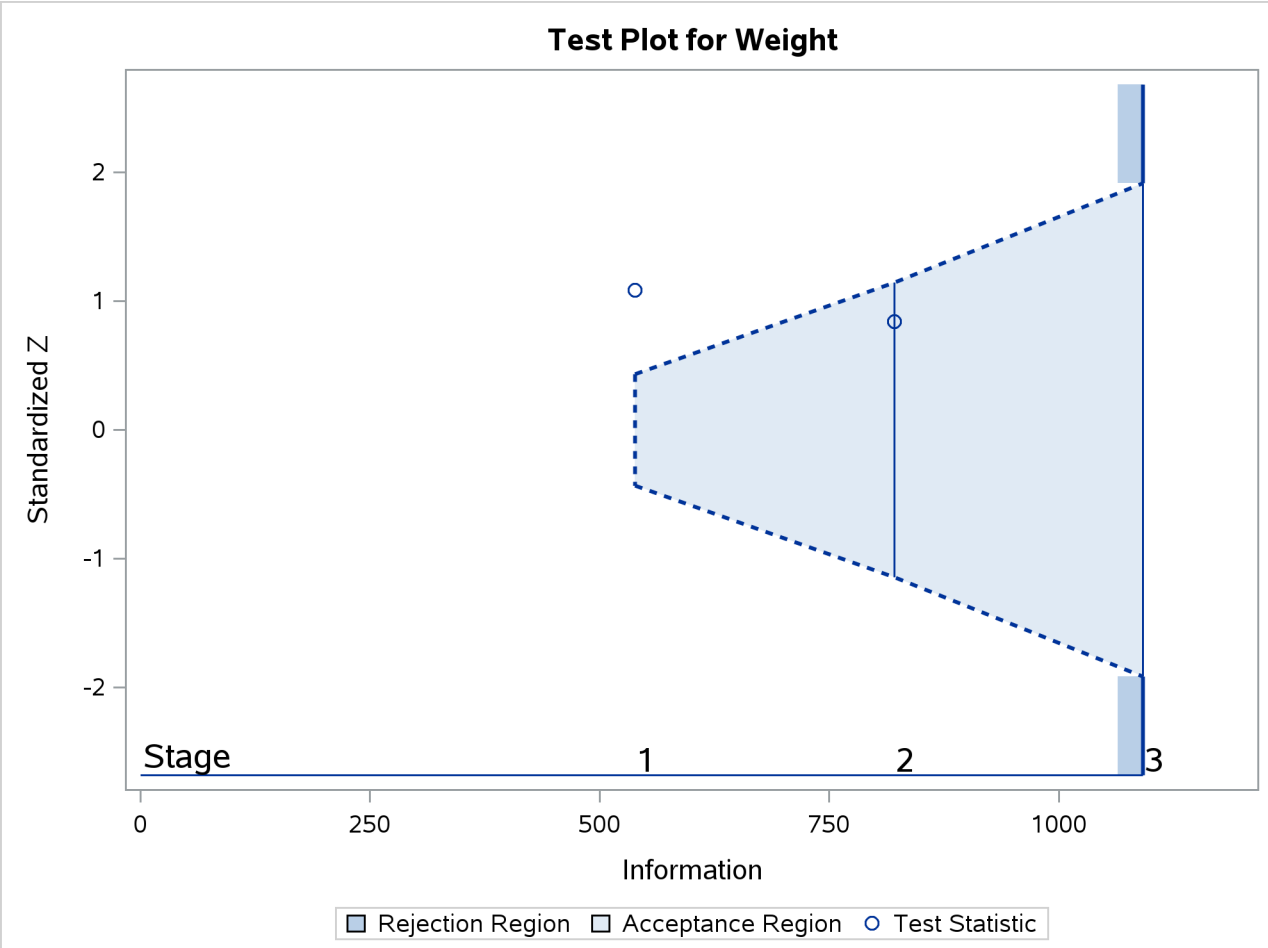

After a trial is stopped, the "Parameter Estimates" table in Output 111.3.19 displays the stopping stage, parameter estimate, unbiased median estimate, confidence limits, and the p-value under the null hypothesis $H_0 : \beta_w = 0$. As expected, the p-value 0.3056 is not significant at the $\alpha = 0.05$ level, and the confidence interval does contain the value zero. The p-value, unbiased median estimate, and confidence limits depend on the ordering of the sample space (k, z), where k is the stage number and z is the standardized Z statistic. With the specified LR ordering, the p-values are computed with the ordering $(k', z') \succ (k, z)$ if $z' > z$. See the section "Available Sample Space Orderings in a Sequential Test" on page 9369 for a detailed description of the LR ordering.

Output 111.3.19 Parameter Estimates

Parameter Estimates LR Ordering						
Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	95% Confidence Limits	
Weight	2	0.029251	0.3056	0.037080	-0.03368	0.10532

Example 111.4: Testing a Binomial Proportion

(View the complete code for this example (seqtex4.sas) in the example repository.)

This example tests a binomial proportion by using a four-stage group sequential design. Suppose a supermarket is developing a new store-brand coffee. From past studies, the positive response for the current store-brand coffee from customers is around 60%. The store is interested in whether the new brand has a better positive response than the current brand.

A power family method is used for the group sequential trial with the null hypothesis $H_0 : p = p_0 = 0.60$ and a one-sided upper alternative with a power of 0.80 at $H_1 : p = 0.70$. To accommodate the zero null reference that is assumed in the SEQDESIGN procedure, an equivalent hypothesis $H_0 : \theta = 0$ with $H_1 : \theta = 0.10$ is used, where $\theta = p - p_0$. The following statements request a power family method with early stopping to reject the null hypothesis:

```
ods graphics on;
proc seqdesign altref=0.10
               boundaryscale=mle
               ;
   PowerFamily: design method=pow
                       nstages=4
                       alt=upper
                       beta=0.20
                       ;
   samplesize model(ceiladjdesign=include)=onesamplefreq( nullprop=0.6);
   ods output AdjustedBoundary=Bnd_Prop;
run;
```

The NULLPROP= option in the SAMPLESIZE statement specifies $p_0 = 0.60$ for the sample size computation. When BOUNDARYSCALE=MLE in the PROC SEQTEST statement, the procedure displays the output boundaries in terms of the maximum likelihood estimates.

When CEILADJDESIGN=INCLUDE in the SAMPLESIZE statement, the table also displays the ceiling-adjusted design information. When you specify ADJUSTEDBOUNDARY=BND_PROP in the ODS OUTPUT statement, PROC SEQTEST creates an output data set named BND_PROP, which contains the resulting ceiling-adjusted design boundary information for the subsequent sequential tests.

The "Design Information" table in Output 111.4.1 displays design specifications and derived statistics. With the specified alternative reference $\theta_1 = p_1 - p_0 = 0.7 - 0.6 = 0.1$, the maximum information 670.38 is also derived.

Output 111.4.1 Design Information

The SEQDESIGN Procedure
Design: PowerFamily

Design Information	
Statistic Distribution	Normal
Boundary Scale	MLE
Alternative Hypothesis	Upper
Early Stop	Reject Null
Method	Power Family
Boundary Key	Both
Alternative Reference	0.1
Number of Stages	4
Alpha	0.05
Beta	0.2
Power	0.8
Max Information (Percent of Fixed Sample)	108.4306
Max Information	670.3782
Null Ref ASN (Percent of Fixed Sample)	106.9276
Alt Ref ASN (Percent of Fixed Sample)	78.51072
Adj Design Alpha	0.05
Adj Design Beta	0.19951
Adj Design Power	0.80049
Adj Design Max Information (Percent of Fixed Sample)	108.4461
Adj Design Max Information	671.4286
Adj Design Null Ref ASN (Percent of Fixed Sample)	106.9406
Adj Design Alt Ref ASN (Percent of Fixed Sample)	78.42926

The "Boundary Information" table in Output 111.4.2 displays the information level, alternative reference, and boundary values at each stage. With the STOP=REJECT option, only the rejection boundary values are displayed.

Output 111.4.2 Boundary Information

Boundary Information (MLE Scale)
Null Reference = 0

	Information Level			Alternative Reference	Boundary Values
Stage	Proportion	Actual	N	Upper	Upper Alpha
1	0.2500	167.5945	35.19485	0.10000	0.20018
2	0.5000	335.1891	70.38971	0.10000	0.11903
3	0.7500	502.7836	105.5846	0.10000	0.08782
4	1.0000	670.3782	140.7794	0.10000	0.07077

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.4.3.

Output 111.4.3 Boundary Plot

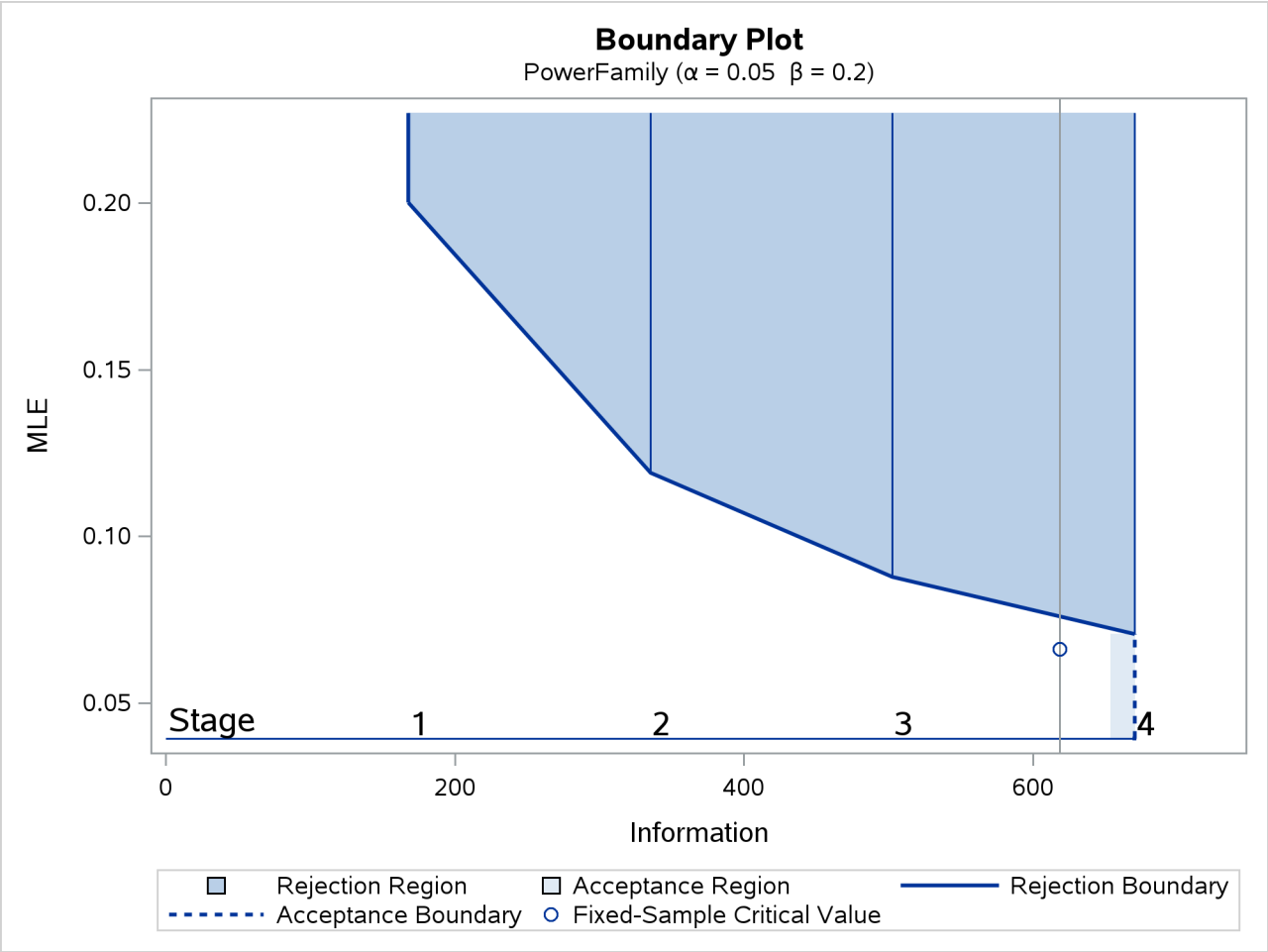

With the MODEL=ONESAMPLEFREQ option in the SAMPLESIZE statement, the "Sample Size Summary" table in Output 111.4.4 displays the parameters for the sample size computation.

Output 111.4.4 Required Sample Size Summary

Sample Size Summary	
Test	One-Sample Proportion
Null Proportion	0.6
Proportion	0.7
Test Statistic	Z for Proportion
Reference Proportion	Alt Ref
Max Sample Size	140.7794
Expected Sample Size (Null Ref)	138.828
Expected Sample Size (Alt Ref)	101.9333

The "Sample Sizes" table in Output 111.4.5 displays the required sample sizes for the group sequential clinical trial.

Output 111.4.5 Required Sample Sizes

Sample Sizes (N) One-Sample Z Test for Proportion				
	Fractional N		Ceiling N	
Stage	N	Information	N	Information
1	35.19	167.6	36	171.4
2	70.39	335.2	71	338.1
3	105.58	502.8	106	504.8
4	140.78	670.4	141	671.4

When you specify CEILADJDESIGN=INCLUDE in the SAMPLESIZE statement, the "Ceiling-Adjusted Design Boundary Information" table in Output 111.4.6 displays the boundary information for the ceiling-adjusted design.

Output 111.4.6 Ceiling-Adjusted Design Boundary Information

Ceiling-Adjusted Design Boundary Information (MLE Scale) Null Reference = 0					
				Alternative	Boundary Values
	Information Level			Reference	Upper
Stage	Proportion	Actual	N	Upper	Alpha
1	0.2553	171.4286	36	0.10000	0.19696
2	0.5035	338.0952	71	0.10000	0.11835
3	0.7518	504.7619	106	0.10000	0.08763
4	1.0000	671.4286	141	0.10000	0.07075

Thus, 36 customers are needed at stage 1, and 35 new customers are needed at each of the remaining stages. Suppose that 36 customers are available at stage 1. Output 111.4.6 lists the 10 observations in the data set count_1.

Output 111.4.7 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Resp
1	1
2	1
3	0
4	0
5	1
6	1
7	0
8	1
9	1
10	1

The Resp variable is an indicator variable with a value of 1 for a customer with a positive response and a value of 0 for a customer without a positive response.

The following statements use the MEANS procedure to compute the mean response at stage 1:

```
proc means data=Prop_1;
   var Resp;
   ods output Summary=Data_Prop1;
run;
```

The following statements create and display (in Output 111.4.8) the data set for the centered mean positive response, $\hat{p} - p_0$:

```
data Data_Prop1;
   set Data_Prop1;
   _Scale_='MLE';
   _Stage_= 1;
   NObs= Resp_N;
   PDiff= Resp_Mean - 0.6;
   keep _Scale_ _Stage_ NObs PDiff;
run;
proc print data=Data_Prop1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.4.8 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	_Scale_	_Stage_	NObs	PDiff
1	MLE	1	36	-0.016667

The following statements invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Prop
             Data(Testvar=PDiff Infovar=NObs)=Data_Prop1
             infoadj=prop
             boundarykey=both
             boundaryscale=mle
             ;
   ods output Test=Test_Prop1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The DATA=DATA_PROP1 option specifies the input data set DATA_PROP1, which contains the test statistic and its associated sample size at stage 1. The TESTVAR=PDIFF option identifies the test variable PDIFF, and the INFOVAR=NOBS option uses the variable NOBS for the number of observations to derive the information level.

If the computed information level for stage 1 is not the same as the value provided in the BOUNDARY= data set, the INFOADJ=PROP option (which is the default) proportionally adjusts the information levels at future interim stages from the levels provided in the BOUNDARY= data set. The BOUNDARYKEY=BOTH option maintains both the α and β levels for the design. Note that you can use the BOUNDARYKEY=BOTH option in PROC SEQTEST even though you used PROC SEQDESIGN to request a ceiling-adjusted design (that is, you specified the CEILADJDESIGN=INCLUDE suboption in the SAMPLESIZE statement in PROC SEQDESIGN). For more information about boundary adjustments in PROC SEQTEST, see the section

"Information Level Adjustments at Future Stages" on page 9360. The BOUNDARYSCALE=MLE option displays the output boundaries in terms of the MLE scale.

The ODS OUTPUT statement with the TEST=TEST_PROP1 option creates an output data set named TEST_PROP1 which contains the updated boundary information for the test at stage 1. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.4.9 displays design specifications. With the specified BOUNDARYKEY=BOTH option, the information levels and boundary values at future stages are modified to maintain both the α and β levels.

Output 111.4.9 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_PROP
Data Set	WORK.DATA_PROP1
Statistic Distribution	Normal
Boundary Scale	MLE
Alternative Hypothesis	Upper
Early Stop	Reject Null
Number of Stages	4
Alpha	0.05
Beta	0.19951
Power	0.80049
Max Information (Percent of Fixed Sample)	108.4461
Max Information	671.428589
Null Ref ASN (Percent of Fixed Sample)	106.9406
Alt Ref ASN (Percent of Fixed Sample)	78.42929

The "Test Information" table in Output 111.4.10 displays the boundary values for the test statistic with the specified MLE scale.

Output 111.4.10 Sequential Tests

Test Information (MLE Scale)
Null Reference = 0

	Information Level			Alternative Reference	Boundary Values Upper		Test PDiff	
Stage	Proportion	Actual	N	Upper	Alpha	Estimate	Action
1	0.2553	171.4286	36	0.10000	0.19696	-0.01667	Continue
2	0.5035	338.0952	71	0.10000	0.11835	.	
3	0.7518	504.7619	106	0.10000	0.08763	.	
4	1.0000	671.4286	141	0.10000	0.07075	.	

When you specify INFOVAR=NOBS in the DATA= option in the PROC SEQTEST statement, the information level at stage 1 is computed as $I_1^* = I_1 \times (n_1^*/n_1)$, where I_1 and n_1 are the information level and sample size, respectively, at stage 1 in the BOUNDARY= data set, and where $n_1^* = 36$ is the available sample size at stage 1. Because $n_1^* = n_1$, the information level at stage 1 is not changed.

With INFOADJ=PROP (which is the default) in the PROC SEQTEST statement, the information levels at interim stages 2 and 3 are derived proportionally from the information levels in the BOUNDARY= data set. At stage 1, the statistic $\hat{\theta} = \hat{p} - p_0 = 0.58333 - 0.6 = -0.01667$ is less than the upper α boundary value 0.19696, so the trial continues to the next stage.

When ODS Graphics is enabled, a boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.4.11. As expected, the test statistic is in the continuation region.

Output 111.4.11 Sequential Test Plot

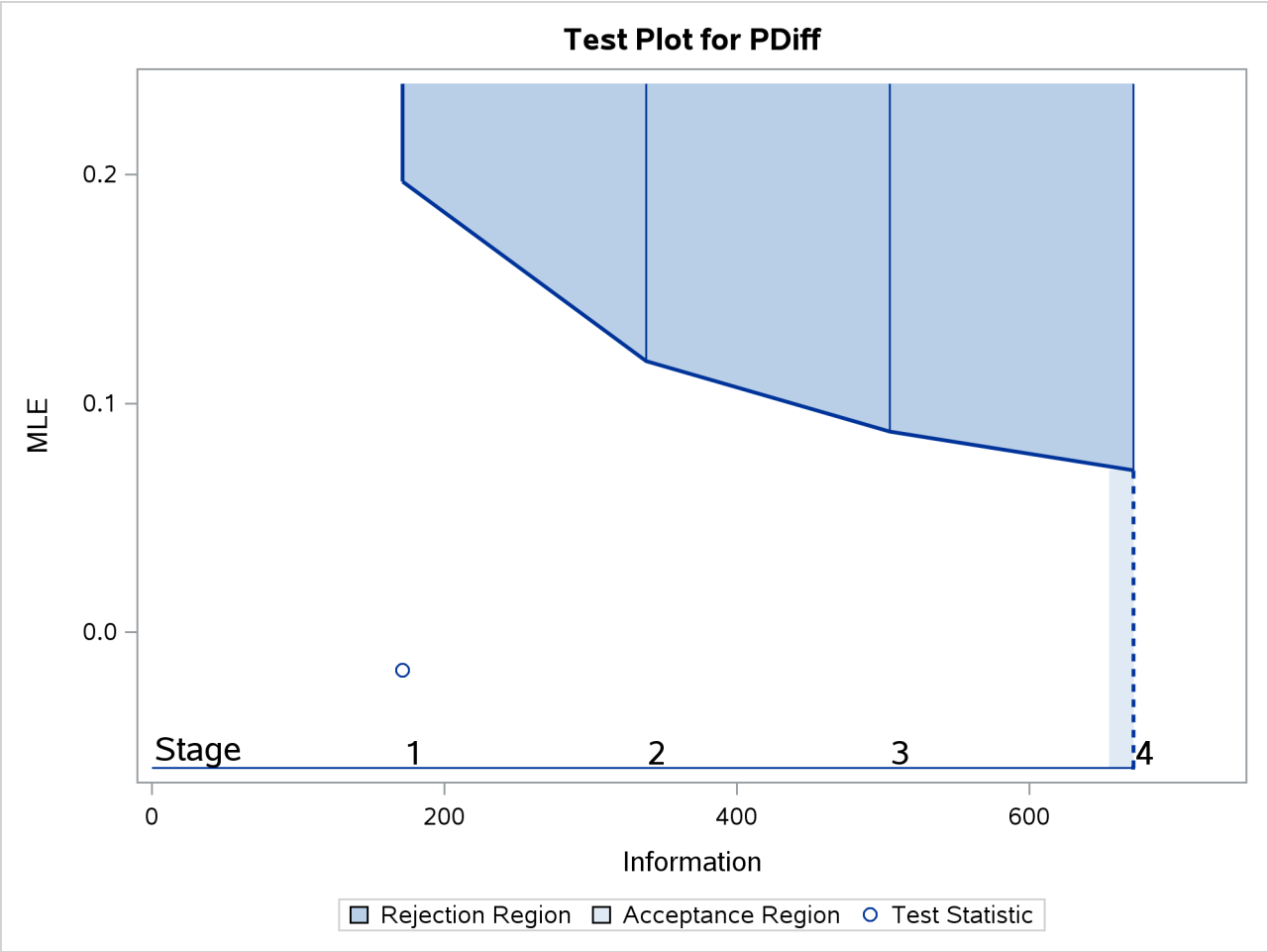

The following statements use the MEANS procedure to compute the mean response at stage 2:

```
proc means data=Prop_2;
   var Resp;
   ods output Summary=Data_Prop2;
run;
```

The following statements create and display (in Output 111.4.12) the data set for the centered mean positive response ($\hat{p} - p_0$) at stage 2:

```
data Data_Prop2;
   set Data_Prop2;
   _Scale_='MLE';
   _Stage_= 2;
   NObs= Resp_N;
   PDiff= Resp_Mean - 0.6;
   keep _Scale_ _Stage_ NObs PDiff;
run;

proc print data=Data_Prop2;
   title 'Statistics Computed at Stage 2';
run;
```

Output 111.4.12 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	_Scale_	_Stage_	NObs	PDiff
1	MLE	2	71	-0.064789

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2:

```
ods graphics on;
proc seqtest Boundary=Test_Prop1
             Data(Testvar=PDiff Infovar=NObs)=Data_Prop2
             infoadj=prop
             boundarykey=both
             boundaryscale=mle
             condpower(cref=1)
             predpower
             plots=condpower
             ;
   ods output test=Test_Prop2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The DATA= option specifies the input data set that contains the test statistic and its associated sample size at stage 2, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=TEST_PROP2 option creates an output data set named TEST_PROP2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The CONDPOWER(CREF=1) option requests the conditional power with the observed statistic under the alternative hypothesis, in addition to the conditional power under the hypothetical reference $\theta = \hat{\theta}$, the MLE estimate. The PREDPOWER option requests the noninformative predictive power with the observed statistic.

The "Test Information" table in Output 111.4.13 displays the boundary values for the test statistic with the specified MLE scale. The test statistic $\hat{\theta} = -0.06479$ is less than the corresponding upper α boundary 0.11835, so the sequential test does not stop at stage 2 to reject the null hypothesis.

Output 111.4.13 Sequential Tests

The SEQTEST Procedure

Test Information (MLE Scale) Null Reference = 0							
	Information Level			Alternative Reference	Boundary Values Upper	Test PDiff	
Stage	Proportion	Actual	N	Upper	Alpha	Estimate	Action
1	0.2553	171.4286	36	0.10000	0.19696	-0.01667	Continue
2	0.5035	338.0952	71	0.10000	0.11835	-0.06479	Continue
3	0.7518	504.7619	106	0.10000	0.08763	.	
4	1.0000	671.4285	141	0.10000	0.07075	.	

With ODS Graphics enabled, the "Test Plot" displays boundary values of the design and the test statistic, as shown in Output 111.4.14. It also shows that the test statistic is in the "Continuation Region" below the upper α boundary value at stage 2.

Output 111.4.14 Sequential Test Plot

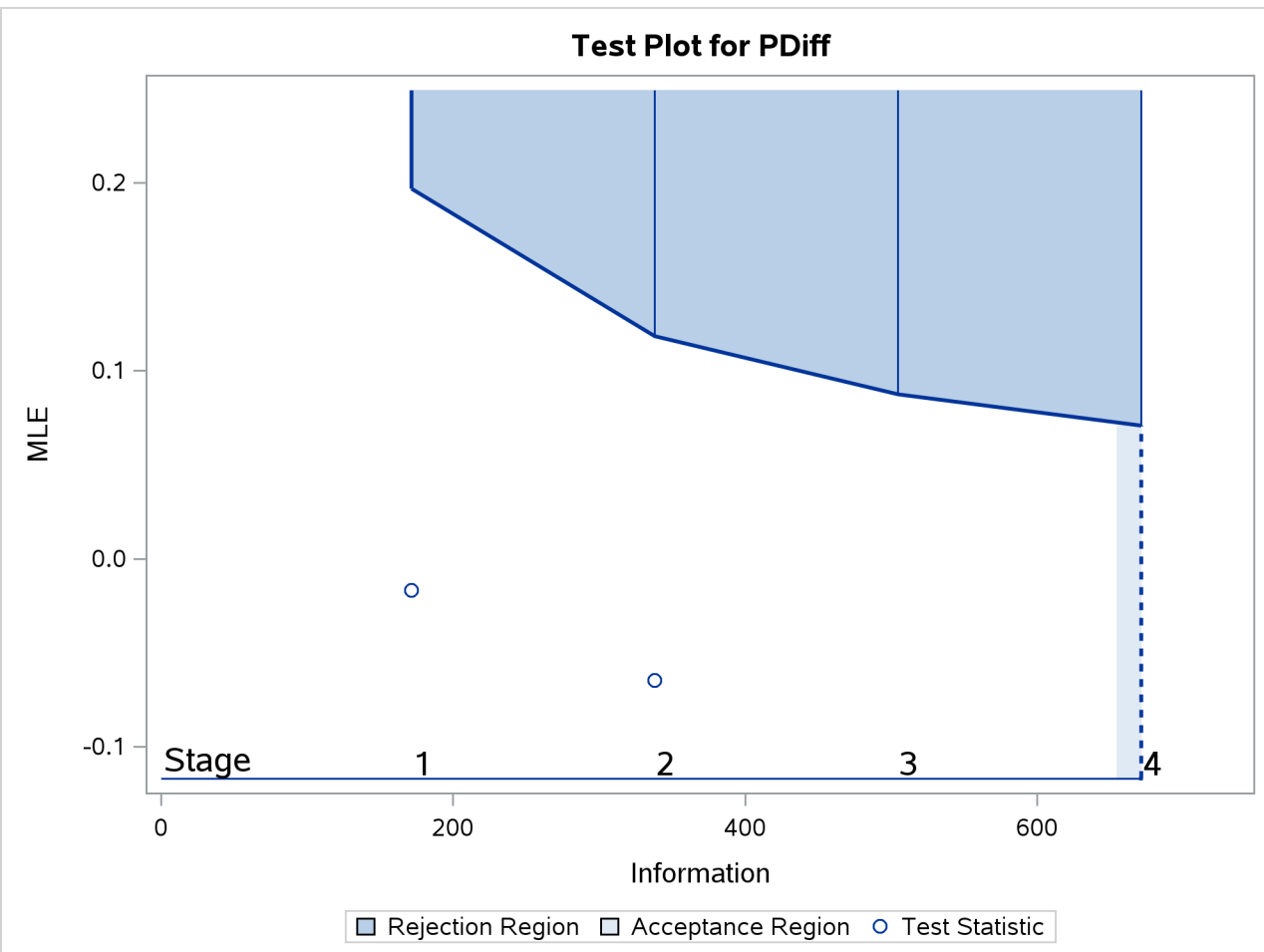

The "Conditional Power Information" table in Output 111.4.15 displays conditional powers given the observed

statistic under hypothetical references $\theta = \hat{\theta}$, the maximum likelihood estimate, and $\theta = \theta_1$. The constant c under CRef for the MLE is derived from $\hat{\theta} = c\theta_1$; that is, $c = \hat{\theta}/\theta_1 = -0.06479/0.1 = -0.6479$.

Output 111.4.15 Conditional Power

Conditional Power Information
Reference = CRef * (Alt Reference)

Stopping Stage	Reference MLE	Reference Ref	Reference CRef	Conditional Power
2	-0.06479	MLE	-0.6479	0.00000
2	-0.06479	Alternative	1.0000	0.02410

The conditional power is the probability of rejecting the null hypothesis under these hypothetical references given the observed statistic $\hat{\theta} = -0.06479$. The table in Output 111.4.14 shows a weak conditional power of 0.0241 under the alternative hypothesis.

With the default TYPE=ALLSTAGES suboption in the CONDPOWER option, the conditional power at the interim stage 2 is the probability that the test statistic would exceed the rejection critical value at all future stages given the observed statistic $\hat{\theta} = -0.06479$.

The "Conditional Power Plot" displays conditional powers given the observed statistic under various hypothetical references, as shown in Output 111.4.16. These references include $\theta = \hat{\theta}$, the maximum likelihood estimate, and $\theta = c_i\theta_1$, where θ_1 is the alternative reference and $c_i = 0, 0.01, \ldots, 1.50$ are constants that are specified in the CREF= option. Output 111.4.16 shows that the conditional power increases as c_i increases.

Output 111.4.16 Conditional Power Plot

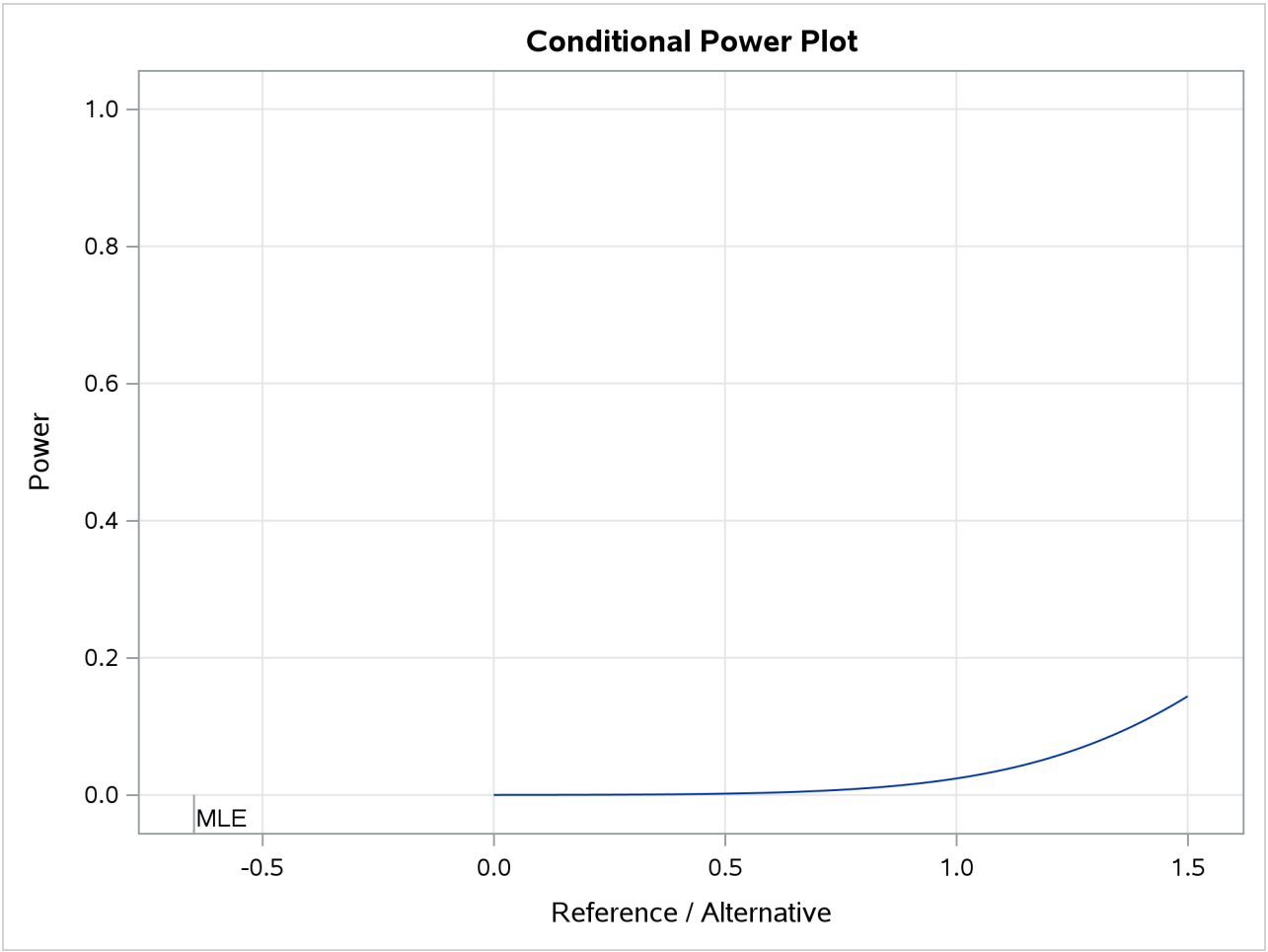

The predictive power is the probability to reject the null hypothesis under the posterior distribution with a noninformative prior given the observed statistic $\hat{\theta} = -0.06479$. The "Predictive Power Information" table in Output 111.4.17 indicates that the predictive power at $\hat{\theta} = -0.06479$ is 0.0002.

Output 111.4.17 Predictive Power

Predictive Power Information

Stopping Stage	MLE	Predictive Power
2	-0.06479	0.00020

With a predictive power 0.0002 and a conditional power of 0.0241 under H_1, the supermarket decides to stop the trial and accept the null hypothesis. That is, the positive response for the new store-brand coffee is not better than that for the current store-brand coffee.

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2. The NSTAGES=3 option sets the next stage as the final stage (stage 3), and the BOUNDARYKEY=BOTH option derives the information level at stage 3 that maintain both Type I and Type II error probability levels. The CONDPOWER(CREF=1) option requests the conditional power with the observed statistic under the

alternative hypothesis, in addition to the conditional power under the hypothetical reference $\theta = \hat{\theta}$, the MLE estimate.

```
proc seqtest Boundary=Test_Prop1
             Data(Testvar=PDiff Infovar=NObs)=Data_Prop2
             nstages=3
             boundarykey=both
             boundaryscale=mle
             condpower(cref=1)
             ;
run;
```

The "Test Information" table in Output 111.4.18 displays the boundary values for the test statistic with the specified MLE scale, assuming that the next stage is the final stage.

Output 111.4.18 Sequential Tests

The SEQTEST Procedure

Test Information (MLE Scale)
Null Reference = 0

	Information Level			Alternative Reference	Boundary Values Upper		Test PDiff	
Stage	Proportion	Actual	N	Upper	Alpha	Estimate	Action
1	0.2642	171.4286	36	0.10000	0.19696	-0.01667	Continue
2	0.5211	338.0952	71	0.10000	0.11835	-0.06479	Continue
3	1.0000	648.8122	136.2506	0.10000	0.06825	.	

The "Conditional Power Information" table in Output 111.4.19 displays conditional powers given the observed statistic, assuming that the next stage is the final stage.

Output 111.4.19 Conditional Power

Conditional Power Information
Reference = CRef * (Alt Reference)

Stopping Stage	Reference MLE	Ref	CRef	Conditional Power
2	-0.06479	MLE	-0.6479	0.00000
2	-0.06479	Alternative	1.0000	0.02318

The conditional power is the probability of rejecting the null hypothesis under these hypothetical references given the observed statistic $\hat{\theta} = -0.06479$. The table in Output 111.4.19 also shows a weak conditional power of 0.02318 under the alternative hypothesis.

Example 111.5: Comparing Two Proportions with a Log Odds Ratio Test

(View the complete code for this example (seqtex5.sas) in the example repository.)

This example compares two binomial proportions by using a log odds ratio statistic in a five-stage group sequential test. A clinic is studying the effect of vitamin C supplements in treating flu symptoms. The study consists of patients in the clinic who exhibit the first sign of flu symptoms within the last 24 hours. These patients are randomly assigned to either the control group (which receives placebo pills) or the treatment group (which receives large doses of vitamin C supplements). At the end of a five-day period, the flu symptoms of each patient are recorded.

Suppose that you know from past experience that flu symptoms disappear in five days for 60% of patients who experience flu symptoms. The clinic would like to detect a 70% symptom disappearance with a high probability. A test that compares the proportions directly specifies the null hypothesis $H_0 : \theta = p_t - p_c = 0$ with a one-sided alternative $H_1 : \theta > 0$ and a power of 0.90 at $H_1 : \theta = 0.10$, where p_t and p_c are the proportions of symptom disappearance in the treatment group and control group, respectively. An alternative trial tests an equivalent hypothesis by using the log odds ratio statistics:

$$\theta = \log\left(\frac{(\frac{p_t}{1-p_t})}{(\frac{p_c}{1-p_c})}\right)$$

Then the null hypothesis is $H_0 : \theta = \theta_0 = 0$ and the alternative hypothesis is

$$H_1 : \theta = \theta_1 = \log\left(\frac{(\frac{0.70}{0.30})}{(\frac{0.6}{0.4})}\right) = 0.441833$$

The following statements invoke the SEQDESIGN procedure and request a five-stage group sequential design by using an error spending function method for normally distributed statistics. The design uses a two-sided alternative hypothesis with early stopping to reject the null hypothesis H_0.

```
ods graphics on;
proc seqdesign altref=0.441833
               boundaryscale=mle
               ;
   OneSidedErrorSpending: design method=errfuncpow
                                 nstages=5
                                 alt=upper
                                 stop=accept
                                 alpha=0.025;
   samplesize model=twosamplefreq( nullprop=0.6 test=logor);
   ods output Boundary=Bnd_CSup;
run;
```

The ODS OUTPUT statement with the BOUNDARY=BND_CSUP option creates an output data set named BND_CSUP which contains the resulting boundary information for the subsequent sequential tests.

The “Design Information” table in Output 111.5.1 displays design specifications and derived statistics. With the specified alternative reference, the maximum information 56.30934 is derived.

Output 111.5.1 Design Information

The SEQDESIGN Procedure
Design: OneSidedErrorSpending

Design Information	
Statistic Distribution	Normal
Boundary Scale	MLE
Alternative Hypothesis	Upper
Early Stop	Accept Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.441833
Number of Stages	5
Alpha	0.025
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	104.6166
Max Information	56.30934
Null Ref ASN (Percent of Fixed Sample)	57.21399
Alt Ref ASN (Percent of Fixed Sample)	102.1058

The "Boundary Information" table in Output 111.5.2 displays information level, alternative reference, and boundary values at each stage. With the specified BOUNDARYSCALE=MLE option, the procedure displays the output boundaries in terms of the MLE scale.

Output 111.5.2 Boundary Information

Boundary Information (MLE Scale)
Null Reference = 0

	Information Level			Alternative Reference	Boundary Values
Stage	Proportion	Actual	N	Upper	Upper Beta
1	0.2000	11.26187	201.1048	0.44183	-0.34844
2	0.4000	22.52374	402.2096	0.44183	-0.02262
3	0.6000	33.7856	603.3144	0.44183	0.11527
4	0.8000	45.04747	804.4192	0.44183	0.19708
5	1.0000	56.30934	1005.524	0.44183	0.25345

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.5.3.

Output 111.5.3 Boundary Plot

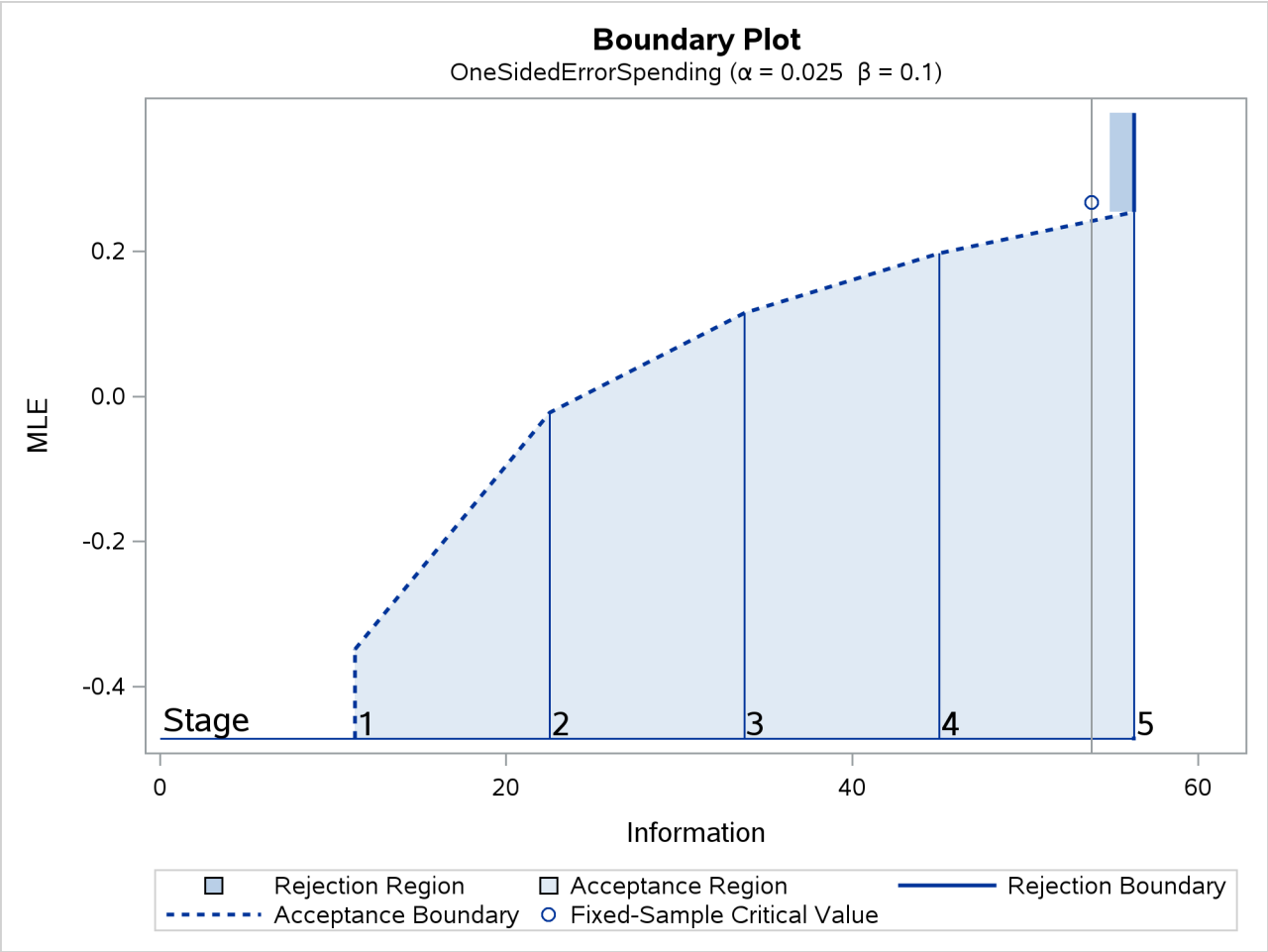

With the SAMPLESIZE statement, the "Sample Size Summary" table in Output 111.5.4 displays the parameters for the sample size computation.

Output 111.5.4 Sample Size Summary

Sample Size Summary	
Test	Two-Sample Proportions
Null Proportion	0.6
Proportion (Group A)	0.7
Test Statistic	Log Odds Ratio
Reference Proportions	Alt Ref
Max Sample Size	1005.524
Expected Sample Size (Null Ref)	549.9132
Expected Sample Size (Alt Ref)	981.3914

The "Sample Sizes" table in Output 111.5.5 displays the required sample sizes for the group sequential clinical trial.

Output 111.5.5 Required Sample Sizes

Sample Sizes (N) Two-Sample Log Odds Ratio Test for Proportion Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	201.10	100.55	100.55	11.2619	202	101	101	11.3120
2	402.21	201.10	201.10	22.5237	404	202	202	22.6240
3	603.31	301.66	301.66	33.7856	604	302	302	33.8240
4	804.42	402.21	402.21	45.0475	806	403	403	45.1360
5	1005.52	502.76	502.76	56.3093	1006	503	503	56.3360

Thus, 101 new patients are needed in each group at stages 1, 2, and 4, and 100 new patients are needed in each group at stages 3 and 5. Suppose that 101 patients are available in each group at stage 1. Output 111.5.6 lists the 10 observations in the data set count_1.

Output 111.5.6 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	TrtGrp	Resp
1	Control	1
2	C_Sup	0
3	Control	0
4	C_Sup	1
5	Control	1
6	C_Sup	1
7	Control	1
8	C_Sup	0
9	Control	0
10	C_Sup	1

The TrtGrp variable is a grouping variable with the value Control for a patient in the placebo control group and the value C_Sup for a patient in the treatment group who receives vitamin C supplements. The Resp variable is an indicator variable with the value 1 for a patient without flu symptoms after five days and the value 0 for a patient with flu symptoms after five days.

The following statements use the LOGISTIC procedure to compute the log odds ratio statistic and its associated standard error at stage 1:

```
proc logistic data=CSup_1 descending;
   class TrtGrp / param=ref;
   model Resp= TrtGrp;
   ods output ParameterEstimates=Parms_CSup1;
run;
```

The DESCENDING option is used to reverse the order for the response levels, so the LOGISTIC procedure is modeling the probability that Resp = 1.

The following statements create and display (in Output 111.5.7) the data set for the log odds ratio statistic and its associated standard error:

```
data Parms_CSup1;
   set Parms_CSup1;
   if Variable='TrtGrp' and ClassVal0='C_Sup';
   _Scale_='MLE';
   _Stage_= 1;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;

proc print data=Parms_CSup1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.5.7 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	TrtGrp	0.3247	0.2856	MLE	1

The following statements invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_CSup
             Parms(Testvar=TrtGrp)=Parms_CSup1
             infoadj=prop
             errspendadj=errfuncpow
             boundarykey=both
             boundaryscale=mle
             ;
   ods output test=Test_CSup1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The PARMS=PARMS_CSUP1 option specifies the input data set PARMS_CSUP1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=TRTGRP option identifies the test variable TRTGRP in the data set.

If the computed information level for stage 1 is not the same as the value provided in the BOUNDARY= data set, the INFOADJ=PROP option (which is the default) proportionally adjusts the information levels at future interim stages from the levels provided in the BOUNDARY= data set. The ERRSPENDADJ=ERRFUNCPOW option adjusts the boundaries with the updated error spending values generated from the power error spending function. The BOUNDARYKEY=BOTH option maintains both the α and β levels. The BOUNDARYSCALE=MLE option displays the output boundaries in terms of the MLE scale.

The ODS OUTPUT statement with the TEST=TEST_CSUP1 option creates an output data set named TEST_CSUP1 which contains the updated boundary information for the test at stage 1. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.5.8 displays design specifications. With the specified BOUNDARYKEY=BOTH option, the information levels and boundary values at future stages are modified to maintain both the α and β levels.

Output 111.5.8 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_CSUP
Data Set	WORK.PARMS_CSUP1
Statistic Distribution	Normal
Boundary Scale	MLE
Alternative Hypothesis	Upper
Early Stop	Accept Null
Number of Stages	5
Alpha	0.025
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	104.6673
Max Information	56.3361718
Null Ref ASN (Percent of Fixed Sample)	57.02894
Alt Ref ASN (Percent of Fixed Sample)	102.1369

The "Test Information" table in Output 111.5.9 displays the boundary values for the test statistic with the specified MLE scale. With the INFOADJ=PROP option (which is the default), the information levels at future interim stages are derived proportionally from the observed information at stage 1 and the information levels in the BOUNDARY= data set.

Since the information level at stage 1 is derived from the PARMS= data set and other information levels are not specified, equal increments are used at remaining stages. At stage 1, the MLE statistic 0.32474 is greater than the corresponding upper β boundary value –0.29906, so the sequential test continues to the next stage.

Output 111.5.9 Sequential Tests

Test Information (MLE Scale)
Null Reference = 0

	Information Level		Alternative Reference	Boundary Values Upper	Test TrtGrp	
Stage	Proportion	Actual	Upper	Beta	Estimate	Action
1	0.2176	12.26014	0.44183	-0.29906	0.32474	Continue
2	0.4132	23.27914	0.44183	-0.01067	.	
3	0.6088	34.29815	0.44183	0.11942	.	
4	0.8044	45.31716	0.44183	0.19829	.	
5	1.0000	56.33617	0.44183	0.25325	.	

With ODS Graphics enabled, a boundary plot with the boundary values and test statistics is displayed, as shown in Output 111.5.10. As expected, the test statistic is in the continuation region.

Output 111.5.10 Sequential Test Plot

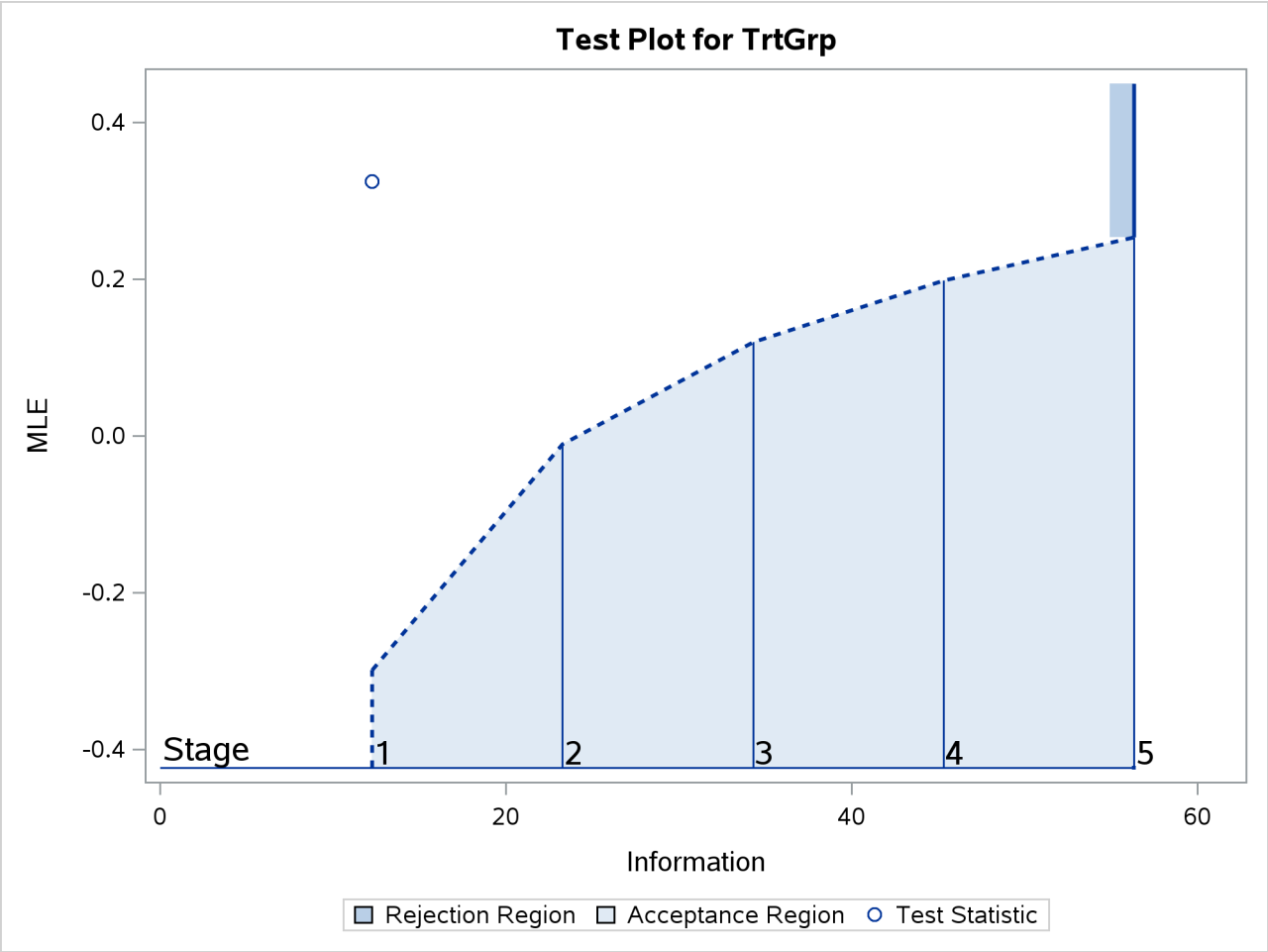

The following statements use the LOGISTIC procedure to compute the log odds ratio statistic and its associated standard error at stage 2:

```
proc logistic data=CSup_2 descending;
   class TrtGrp / param=ref;
   model Resp= TrtGrp;
   ods output ParameterEstimates=Parms_CSup2;
run;
```

The following statements create and display (in Output 111.5.11) the data set for the mean positive response and its associated standard error at stage 2:

```
data Parms_CSup2;
   set Parms_CSup2;
   if Variable='TrtGrp' and ClassVal0='C_Sup';
   _Scale_='MLE';
   _Stage_= 2;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;
proc print data=Parms_CSup2;
   title 'Statistics Computed at Stage 2';
```

```
run;
```

Output 111.5.11 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	TrtGrp	0.2356	0.2073	MLE	2

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2:

```
proc seqtest Boundary=Test_CSup1
             Parms( testvar=TrtGrp)=Parms_CSup2
             infoadj=prop
             errspendadj=errfuncpow
             boundarykey=both
             boundaryscale=mle
             ;
   ods output Test=Test_CSup2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=CSUP_LDL2 option creates an output data set named CSUP_LDL2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Test Information" table in Output 111.5.12 displays the boundary values for the test statistic with the specified MLE scale. The test statistic 0.2356 is greater than the corresponding upper β boundary value –0.01068, so the sequential test continues to the next stage.

Output 111.5.12 Sequential Tests

The SEQTEST Procedure

Test Information (MLE Scale) Null Reference = 0						
	Information Level		Alternative Reference	Boundary Values	Test TrtGrp	
Stage	Proportion	Actual	Upper	Upper Beta	Estimate	Action
1	0.2176	12.26014	0.44183	-0.29906	0.32474	Continue
2	0.4132	23.27916	0.44183	-0.01068	0.23560	Continue
3	0.6088	34.29799	0.44183	0.11942	.	
4	0.8044	45.31681	0.44183	0.19829	.	
5	1.0000	56.33563	0.44183	0.25325	.	

Similar results are found at stages 3 and stage 4, so the trial continues to the final stage. The following statements use the LOGISTIC procedure to compute the log odds ratio statistic and its associated standard error at stage 5:

```
proc logistic data=CSup_5 descending;
   class TrtGrp / param=ref;
   model Resp= TrtGrp;
   ods output ParameterEstimates=Parms_CSup5;
run;
```

The following statements create and display (in Output 111.5.13) the data set for the log odds ratio statistic and its associated standard error at stage 5:

```
data Parms_CSup5;
   set Parms_CSup5;
   if Variable='TrtGrp' and ClassVal0='C_Sup';
   _Scale_='MLE';
   _Stage_= 5;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;

proc print data=Parms_CSup5;
   title 'Statistics Computed at Stage 5';
run;
```

Output 111.5.13 Statistics Computed at Stage 5

Statistics Computed at Stage 5

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	TrtGrp	0.2043	0.1334	MLE	5

The following statements invoke the SEQTEST procedure to test for the hypothesis at stage 5:

```
ods graphics on;
proc seqtest Boundary=Test_CSup4
             Parms( testvar=TrtGrp)=Parms_CSup5
             errspendadj=errfuncpow
             boundaryscale=mle
             cialpha=.025
             rci
             plots=rci
             ;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 5, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 5, and the TESTVAR= option identifies the test variable in the data set. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary value at stage 5 is derived to maintain the α level.

The "Test Information" table in Output 111.5.14 displays the boundary values for the test statistic with the specified MLE scale. The test statistic 0.2043 is less than the corresponding upper β boundary 0.25375, so the sequential test stops to accept the null hypothesis. That is, there is no reduction in duration of symptoms for the group receiving vitamin C supplements.

Output 111.5.14 Sequential Tests

The SEQTEST Procedure

	Test Information (MLE Scale) Null Reference = 0					
	Information Level		Alternative Reference	Boundary Values Upper	Test TrtGrp	
Stage	Proportion	Actual	Upper	Beta	Estimate	Action
1	0.2183	12.26014	0.44183	-0.29906	0.32474	Continue
2	0.4145	23.27916	0.44183	-0.01068	0.23560	Continue
3	0.6141	34.48793	0.44183	0.12134	0.14482	Continue
4	0.8092	45.44685	0.44183	0.19899	0.20855	Continue
5	1.0000	56.16068	0.44183	0.25375	0.20430	Accept Null

The "Test Plot" displays boundary values of the design and the test statistics, as shown in Output 111.5.15. It also shows that the test statistic is in the "Acceptance Region" at the final stage.

Output 111.5.15 Sequential Test Plot

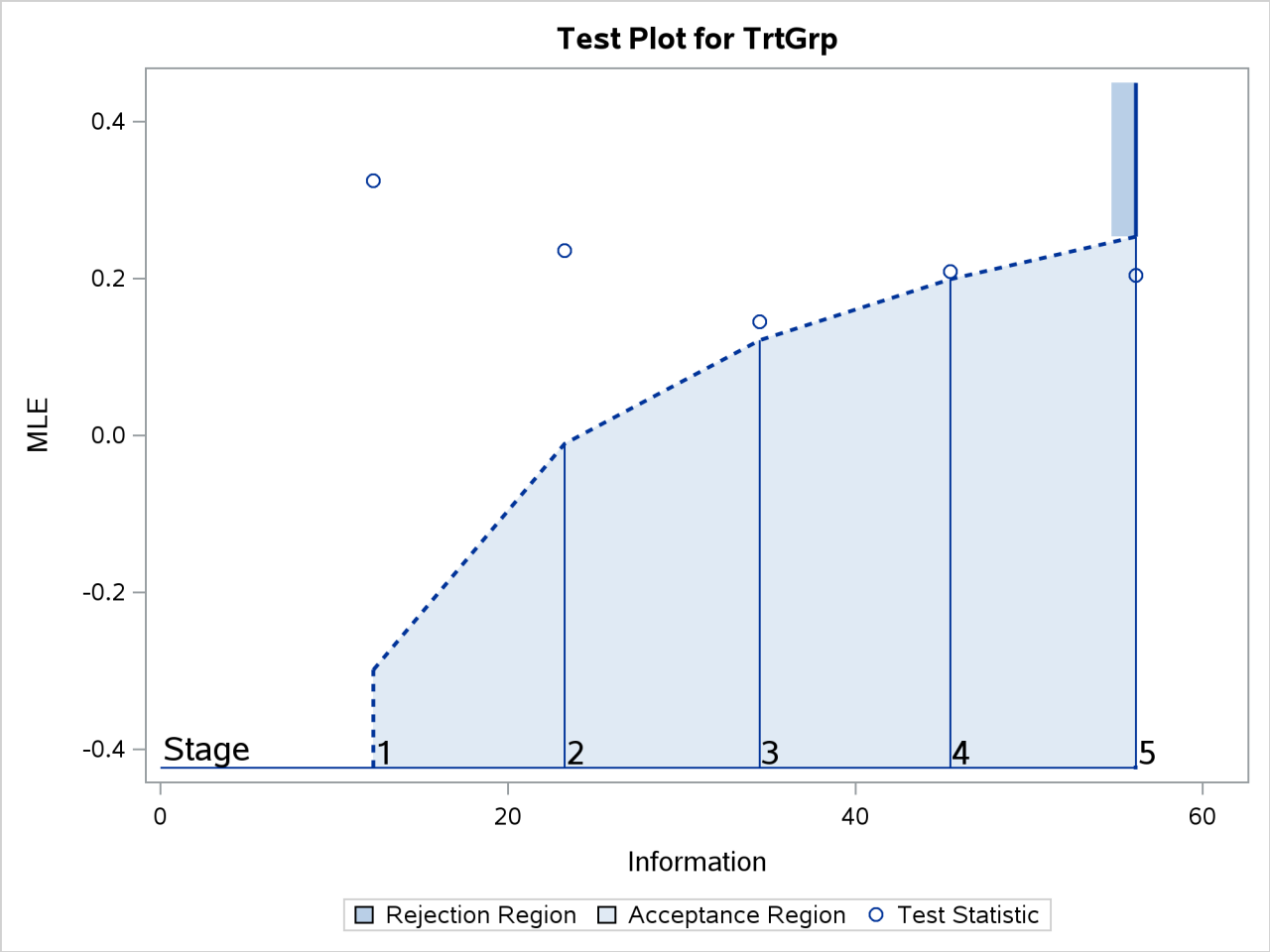

After a trial is stopped, the "Parameter Estimates" table in Output 111.5.16 displays the stopping stage,

parameter estimate, unbiased median estimate, confidence limits, and the p-value under the null hypothesis $H_0 : \theta = 0$. As expected, the p-value 0.0456 is not significant at $\alpha = 0.025$ level and the lower 97.5% confidence limit is less than the value $\theta_0 = 0$. The p-value, unbiased median estimate, and confidence limits depend on the ordering of the sample space (k, z), where k is the stage number and z is the standardized Z statistic.

Output 111.5.16 Parameter Estimates

Parameter Estimates Stagewise Ordering					
Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	Lower 97.5% CL
TrtGrp	5	0.204303	0.0456	0.234494	-0.03712

Since the test is accepted at stage 5, the p-value computed by using the default stagewise ordering can be expressed as

$$\alpha_u = P_{\theta=0}\left(z_5 < Z_5 \mid b_k < Z_k, k < 5\right)$$

where $z_5 = 1.53105$ is the test statistic at stage 5, Z_k is a standardized normal variate at stage k, and b_k is the upper β boundary value in the standardized Z scale at stage $k, k = 1, 2, \ldots, 5$.

With the RCI option, the "Repeated Confidence Intervals" table in Output 111.5.17 displays repeated confidence intervals for the parameter. For a one-sided test with an upper alternative hypothesis, since the upper acceptance repeated confidence limit 0.3924 at the final stage is less than the alternative reference 0.441833, the null hypothesis is accepted.

Output 111.5.17 Repeated Confidence Intervals

Repeated Confidence Intervals			
			Acceptance Boundary
Stage	Information Level	Parameter Estimate	Upper 89.94% CL
1	12.2601	0.32474	1.0656
2	23.2792	0.23560	0.6881
3	34.4879	0.14482	0.4653
4	45.4468	0.20855	0.4514
5	56.1607	0.20430	0.3924

With the PLOTS=RCI option, the "Repeated Confidence Intervals Plot" displays repeated confidence intervals for the parameter, as shown in Output 111.5.18. It shows that the upper acceptance repeated confidence limit at the final stage is less than the alternative reference 0.441833. This implies that the study accepts the null hypothesis at the final stage.

Output 111.5.18 Repeated Confidence Intervals Plot

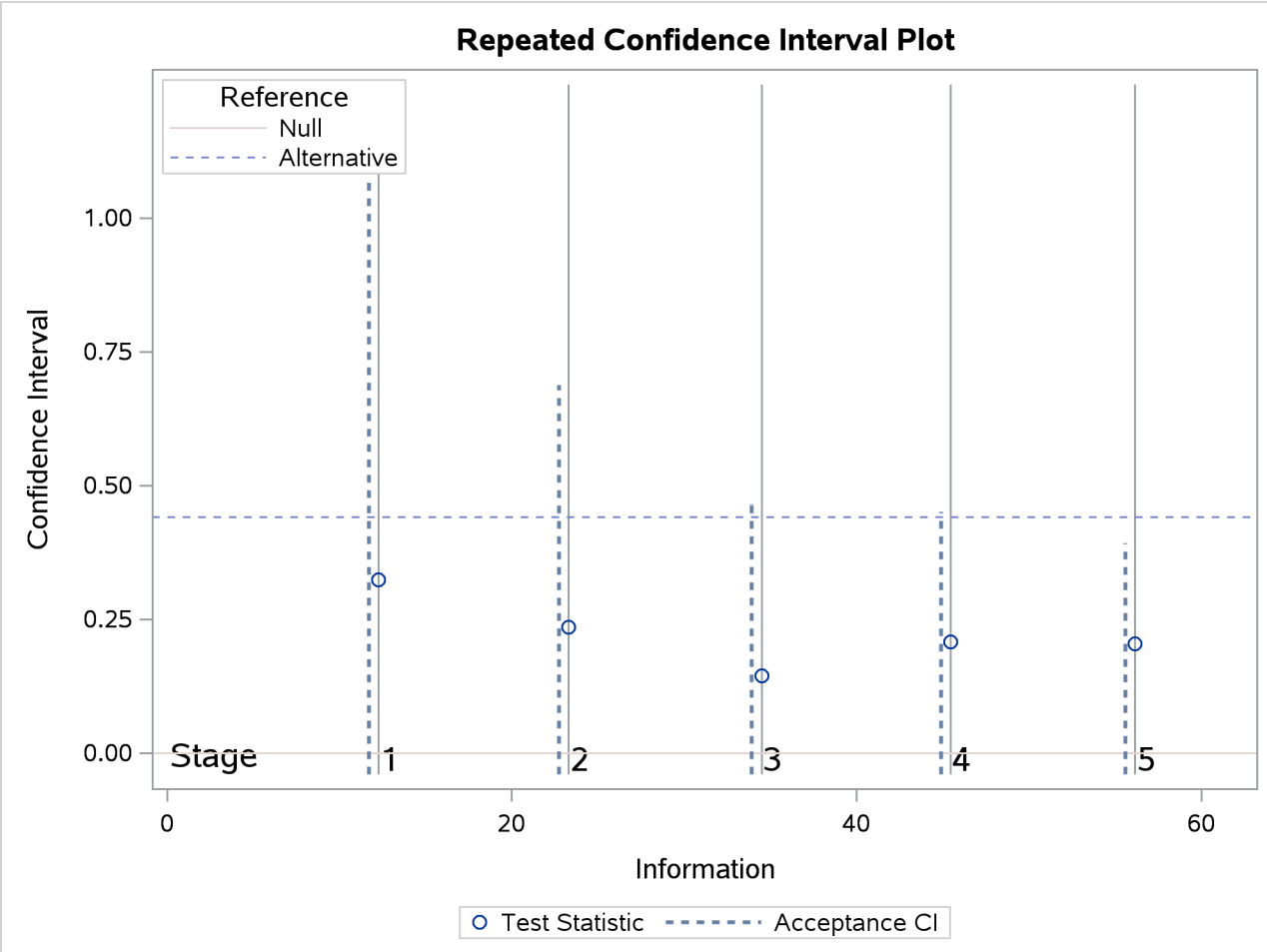

Example 111.6: Comparing Two Survival Distributions with a Log-Rank Test

(View the complete code for this example (seqtex6.sas) in the example repository.)

This example requests a log-rank test that compares two survival distributions for the treatment effect (Jennison and Turnbull 2000, pp. 77–79; Whitehead 1997, pp. 36–39).

A clinic is studying the effect of a new cancer treatment. The study consists of mice exposed to a carcinogen and randomized to either the control group or the treatment group. The event of interest is the death from cancer induced by the carcinogen, and the response is the time from randomization to death.

Following the derivations in the section "Test for Two Survival Distributions with a Log-Rank Test" on page 9201 in Chapter 110, "The SEQDESIGN Procedure," the hypothesis $H_0 : \theta = -\log(\lambda) = 0$ with an alternative hypothesis $H_1 : \theta = \theta_1 > 0$ can be used, where λ is the hazard ratio between the treatment group and control group.

Suppose that from past experience, the median survival time for the control group is $t_0 = 20$ weeks, and the study would like to detect a $t_1 = 40$ weeks median survival time with a 80% power in the trial. Assuming

exponential survival functions for the two groups, the hazard rates can be computed from

$$S_j(t_j) = e^{-h_j t_j} = \frac{1}{2}$$

where $j = 0, 1$.

Thus, with $h_0 = 0.03466$ and $h_1 = 0.01733$, the hazard ratio $\lambda_1 = h_1/h_0 = 1/2$ and the alternative hypothesis is

$$\theta_1 = -\log(\lambda_1) = -\log(\frac{1}{2}) = 0.69315$$

The following statements invoke the SEQDESIGN procedure and request a four-stage group sequential design for normally distributed data. The design uses a one-sided alternative hypothesis with early stopping to reject and to accept the null hypothesis H_0. Whitehead's triangular method is used to derive the boundaries.

```
ods graphics on;
proc seqdesign boundaryscale=score
               ;
   OneSidedWhitehead: design method=whitehead
                             nstages=4
                             boundarykey=alpha
                             alt=upper  stop=both
                             beta=0.20;
   samplesize model=twosamplesurvival
                     ( nullhazard=0.03466
                       hazard=0.01733
                       accrate=10);
run;
```

A Whitehead method creates boundaries that approximately satisfy the Type I and Type II error probability level specification. The BOUNDARYKEY=ALPHA option is used to adjust the boundary value at the last stage and to meet the specified Type I probability level.

The specified ACCRATE=10 option indicates that 10 mice will be accrued each week and the resulting minimum and maximum accrual times are displayed. With the BOUNDARYSCALE=SCORE option, the procedure displays the output boundaries with the score statistics.

The "Design Information" table in Output 111.6.1 displays design specifications and derived statistics.

Output 111.6.1 Design Information

The SEQDESIGN Procedure
Design: OneSidedWhitehead

Design Information	
Statistic Distribution	Normal
Boundary Scale	Score
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Method	Whitehead
Boundary Key	Alpha
Alternative Reference	0.693147
Number of Stages	4
Alpha	0.05
Beta	0.20044
Power	0.79956
Max Information (Percent of Fixed Sample)	129.9894
Max Information	16.70638
Null Ref ASN (Percent of Fixed Sample)	62.6302
Alt Ref ASN (Percent of Fixed Sample)	74.00064

The "Boundary Information" table in Output 111.6.2 displays the information level, alternative reference, and boundary values at each stage.

Output 111.6.2 Boundary Information

Boundary Information (Score Scale)
Null Reference = 0

	Information Level			Alternative Reference	Boundary Values Upper	
Stage	Proportion	Actual	Events	Upper	Beta	Alpha
1	0.2500	4.176595	16.70638	2.89500	-0.95755	4.78775
2	0.5000	8.35319	33.41276	5.78999	1.91510	5.74529
3	0.7500	12.52979	50.11914	8.68499	4.78775	6.70284
4	1.0000	16.70638	66.82552	11.57998	7.81300	7.81300

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.6.3.

Output 111.6.3 Boundary Plot

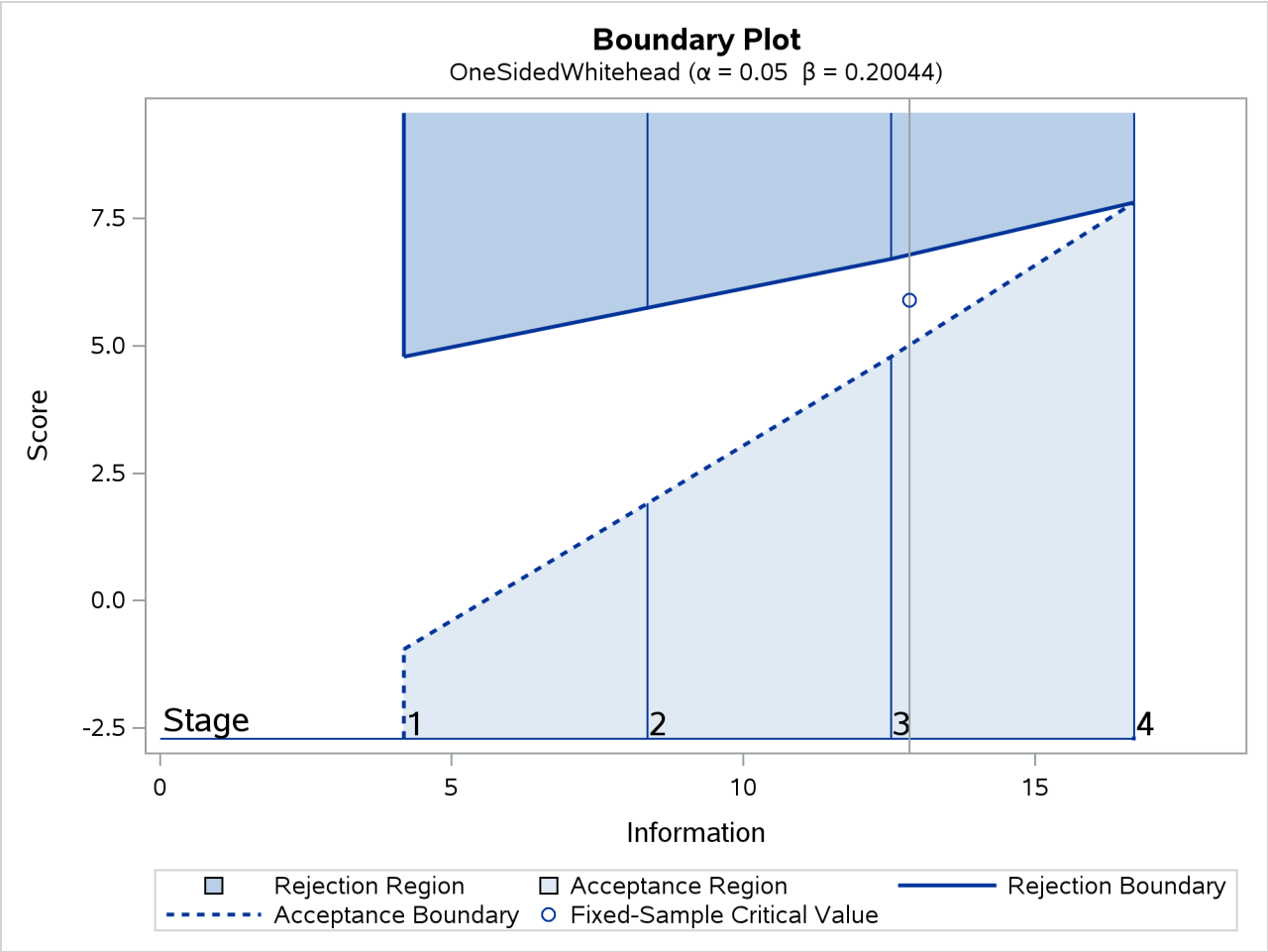

With the MODEL=TWOSAMPLESURVIVAL option in the SAMPLESIZE statement, the "Sample Size Summary" table in Output 111.6.4 displays the parameters for the sample size computation.

Output 111.6.4 Required Sample Size Summary

Sample Size Summary	
Test	Two-Sample Survival
Null Hazard Rate	0.03466
Hazard Rate (Group A)	0.01733
Hazard Rate (Group B)	0.03466
Hazard Ratio	0.5
log(Hazard Ratio)	-0.69315
Reference Hazards	Alt Ref
Accrual	Uniform
Accrual Rate	10
Min Accrual Time	6.682552
Min Sample Size	66.82552
Max Accrual Time	25.40111
Max Sample Size	254.0111
Max Number of Events	66.82552

With a minimum accrual time of 6.6826 weeks and a maximum accrual time of 25.401 weeks, an accrual time of 20 weeks is used in the study.

The "Numbers of Events" table in Output 111.6.5 displays the required number of events for the group sequential clinical trial.

Output 111.6.5 Required Numbers of Events

Numbers of Events (D) Two-Sample Log-Rank Test		
Stage	D	Information
1	16.71	4.1766
2	33.41	8.3532
3	50.12	12.5298
4	66.83	16.7064

The following statements invoke the SEQDESIGN procedure and provide more detailed sample size information:

```
proc seqdesign boundaryscale=score
               ;
   OneSidedWhitehead: design method=whitehead
                             nstages=4
                             boundarykey=alpha
                             alt=upper
                             stop=both
                             beta=0.20;
   samplesize model=twosamplesurvival
                     ( nullhazard=0.03466
                       hazard=0.01733
                       accrate=10 acctime=20);
   ods output Boundary=Bnd_Surv;
run;
```

The ODS OUTPUT statement with the BOUNDARY=BND_SURV option creates an output data set named BND_SURV which contains the resulting boundary information for the subsequent sequential tests.

With an accrual time of 20 weeks, the "Sample Size Summary" table in Output 111.6.6 displays the follow-up time for the trial.

Output 111.6.6 Required Sample Size Summary

The SEQDESIGN Procedure
Design: OneSidedWhitehead

Sample Size Summary	
Test	Two-Sample Survival
Null Hazard Rate	0.03466
Hazard Rate (Group A)	0.01733
Hazard Rate (Group B)	0.03466
Hazard Ratio	0.5
log(Hazard Ratio)	-0.69315
Reference Hazards	Alt Ref
Accrual	Uniform
Accrual Rate	10
Accrual Time	20
Follow-up Time	6.474366
Total Time	26.47437
Max Number of Events	66.82552
Max Sample Size	200
Expected Sample Size (Null Ref)	161.5941
Expected Sample Size (Alt Ref)	172.4692
Follow-up Time (Ceiling Time)	7
Total Time (Ceiling Time)	27

The "Numbers of Events and Sample Sizes" table in Output 111.6.7 displays the required sample sizes for the group sequential clinical trial.

Output 111.6.7 Numbers of Events and Sample Sizes

Numbers of Events (D) and Sample Sizes (N)
Two-Sample Log-Rank Test

				Fractional Time				
Stage	D	D(Grp 1)	D(Grp 2)	Time	N	N(Grp 1)	N(Grp 2)	Information
1	16.71	5.82	10.89	11.9867	119.87	59.93	59.93	4.1766
2	33.41	11.84	21.57	17.3585	173.58	86.79	86.79	8.3532
3	50.12	18.01	32.11	21.7480	200.00	100.00	100.00	12.5298
4	66.83	24.46	42.37	26.4744	200.00	100.00	100.00	16.7064

Numbers of Events (D) and Sample Sizes (N)
Two-Sample Log-Rank Test

				Ceiling Time				
Stage	D	D(Grp 1)	D(Grp 2)	Time	N	N(Grp 1)	N(Grp 2)	Information
1	16.74	5.83	10.91	12	120.00	60.00	60.00	4.1854
2	35.73	12.68	23.04	18	180.00	90.00	90.00	8.9322
3	51.07	18.37	32.70	22	200.00	100.00	100.00	12.7667
4	68.55	25.14	43.41	27	200.00	100.00	100.00	17.1378

Thus the study will perform three interim analyses after 12, 18, and 22 weeks and a final analysis after 27 weeks if the study does not stop at any of the interim analyses.

Note that the SEQDESIGN procedure does not compute numbers of events or sample sizes for all statistical models. If the number of events or sample size for a fixed-sample design is available, then the MODEL=INPUTNEVENTS or MODEL=INPUTNOBS option can be used to input fixed-sample information. For example, with a required fixed-sample number of events 51.41, the following SAMPLESIZE statement can be used to produce the same sample size results:

```
samplesize model=inputnevents
                 ( d=51.41 sample=two
                   hazard=0.03466 0.01733
                   accrate=10 acctime=20);
```

Suppose that 120 mice are available after week 12 for the first interim analysis. Output 111.6.8 lists the 10 observations in the data set weeks_1.

Output 111.6.8 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	TrtGp	Event	Weeks
1	0	0	11
2	1	0	11
3	0	0	11
4	1	0	11
5	0	1	6
6	1	0	11
7	0	0	11
8	1	0	11
9	0	1	9
10	1	0	11

The TrtGp variable is a grouping variable with the value 0 for a mouse in the placebo control group and the value 1 for a mouse in the treatment group. The Weeks variable is the survival time variable measured in weeks and the Event variable is the censoring variable with the value 0 indicating censoring. That is, the values of Weeks are considered censored if the corresponding values of Event are 0; otherwise, they are considered as event times.

The following statements use the LIFETEST procedure to estimate the log-rank statistic at stage 1:

```
proc lifetest data=Surv_1;
   time Weeks*Event(0);
   test TrtGp;
   ods output logunichisq=Parms_Surv1;
run;
```

The following statements create and display (in Output 111.6.9) the data set for the log-rank statistic and its associated standard error:

```
data Parms_Surv1;
   set Parms_Surv1(rename=(Statistic=Estimate));
   if Variable='TrtGp';
   _Scale_='Score';
   _Stage_= 1;
   keep Variable _Scale_ _Stage_ StdErr Estimate;
```

```
run;

proc print data=Parms_Surv1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.6.9 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	TrtGp	3.2004	1.9979	Score	1

The following statements invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Surv
             Parms(Testvar=TrtGp)=Parms_Surv1
             infoadj=none
             boundaryscale=score
             ;
   ods output Test=Test_Surv1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The PARMS=PARMS_SURV1 option specifies the input data set PARMS_SURV1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=TRTGP option identifies the test variable TRTGP in the data set. The INFOADJ=NONE option maintains the information levels for future interim stages (2 and 3) at the values provided in the BOUNDARY= data set.

The ODS OUTPUT statement with the TEST=TEST_SURV1 option creates an output data set named TEST_SURV1 which contains the updated boundary information for the test at stage 1. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.6.10 displays design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the maximum information and the Type I error level α are preserved. Since the computed information level at stage 1 is not the same as the value provided in the BOUNDARY= data set, the power has been modified.

Output 111.6.10 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_SURV
Data Set	WORK.PARMS_SURV1
Statistic Distribution	Normal
Boundary Scale	Score
Alternative Hypothesis	Upper
Early Stop	Accept/Reject Null
Number of Stages	4
Alpha	0.05
Beta	0.20055
Power	0.79945
Max Information (Percent of Fixed Sample)	130.0335
Max Information	16.7063807
Null Ref ASN (Percent of Fixed Sample)	62.80859
Alt Ref ASN (Percent of Fixed Sample)	74.19158

The "Test Information" table in Output 111.6.11 displays the boundary values for the test statistic with the SCORE statistic scale. Since only the information level at stage 1 is specified in the DATA= data set, the information levels at subsequent stages are derived proportionally from the corresponding information levels provided in the BOUNDARY= data set. At stage 1, the score statistic 3.2004 is between the upper β boundary value –1.0386 and the upper α boundary value 4.7142, so the trial continues to the next stage.

Output 111.6.11 Sequential Tests

Test Information (Score Scale) Null Reference = 0							
	Information Level		Alternative Reference	Boundary Values		Test TrtGp	
Stage	Proportion	Actual	Upper	Upper Beta	Upper Alpha	Estimate	Action
1	0.2389	3.991698	2.76683	-1.03862	4.71423	3.20040	Continue
2	0.5000	8.35319	5.78999	1.91799	5.73973	.	
3	0.7500	12.52979	8.68499	4.78804	6.70287	.	
4	1.0000	16.70638	11.57998	7.81349	7.81349	.	

Note that the observed information level 3.9917 corresponds to a proportion of 0.2389 in information level. If the observed information level is much smaller than the target proportion of 0.25, then you need to increase the accrual rate, accrual time, or follow-up time to achieve the target maximum information level for the trial. Scharfstein and Tsiatis (1998) use the simulation and bootstrap methods to modify the trial at interim stages to achieve the target maximum information level. These modifications should be specified in the study protocol or study plan before the study starts.

With ODS Graphics enabled, a boundary plot with test statistics is displayed, as shown in Output 111.6.12. As expected, the test statistic is in the continuation region between the upper β and α boundary values.

Output 111.6.12 Sequential Test Plot

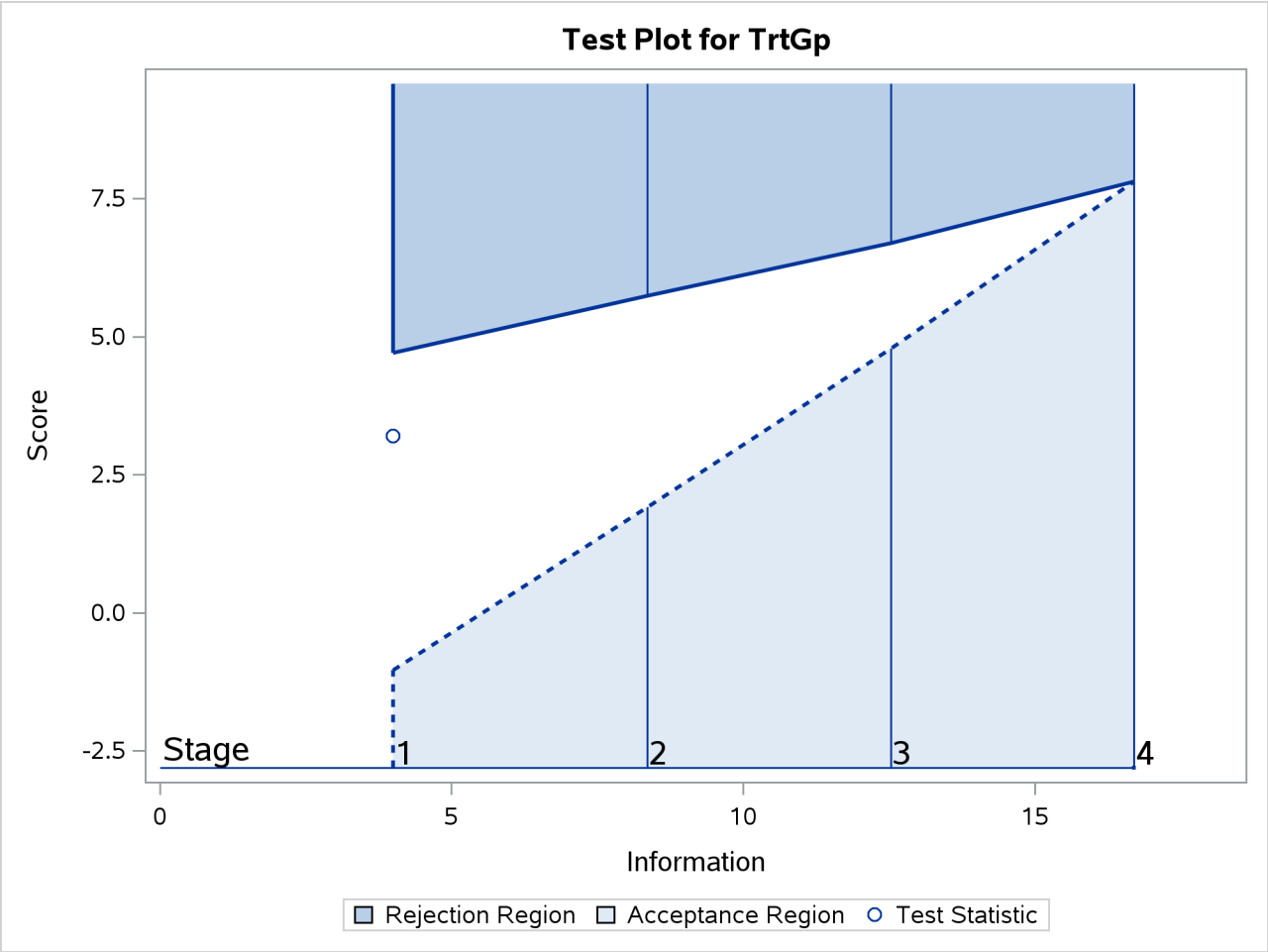

Note that the input DATA= option can also be used for the test statistics. For example, the following statements create and display (in Output 111.6.13) the data set for the log-rank statistic and its associated standard error after the LIFETEST procedure. Since the log-rank statistic is a score statistic, the corresponding information level is the variance of the statistic.

```
proc lifetest data=Surv_1;
   time Weeks*Event(0);
   test TrtGp;
   ods output logunichisq=Parms_Surv1a;
run;

data Parms_Surv1a;
   set Parms_Surv1a(rename=(Statistic=TrtGp));
   keep _Scale_ _Stage_ _Info_ TrtGp;
   _Scale_='Score';
   _Stage_= 1;
   _Info_= StdErr * StdErr;
   if Variable='TrtGp';
run;
```

```
proc print data=Parms_Surv1a;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.6.13 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	TrtGp	_Scale_	_Stage_	_Info_
1	3.2004	Score	1	3.99170

The following statements can be used to invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Surv
             Data(Testvar=TrtGp)=Parms_Surv1a
             infoadj=none
             boundaryscale=score
             ;
   ods output Test=Test_Surv1;
run;
```

The following statements use the LIFETEST procedure to compute the log-rank statistic and its associated standard error at stage 2:

```
proc lifetest data=Surv_2;
   time Weeks*Event(0);
   test TrtGp;
   ods output logunichisq=Parms_Surv2;
run;
```

The following statements create and display (in Output 111.6.14) the data set for the log-rank statistic and its associated standard error for each of the first two stages:

```
data Parms_Surv2;
   set Parms_Surv2 (rename=(Statistic=Estimate));
   if Variable='TrtGp';
   _Scale_='Score';
   _Stage_= 2;
   keep Variable _Scale_ _Stage_ StdErr Estimate;
run;

proc print data=Parms_Surv2;
   title 'Statistics Computed at Stage 2';
run;
```

Output 111.6.14 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	TrtGp	7.3136	2.9489	Score	2

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2:

```
ods graphics on;
proc seqtest Boundary=Test_Surv1
             Parms(Testvar=TrtGp)=Parms_Surv2
             infoadj=none
             boundaryscale=score
             citype=lower
             ;
   ods output Test=Test_Surv2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set. The INFOADJ=NONE option maintains the information level for stage 3 at the value provided in the BOUNDARY= data set.

The ODS OUTPUT statement with the TEST=TEST_SURV2 option creates an output data set named TEST_SURV2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage if the trial does not stop at the current stage.

The "Test Information" table in Output 111.6.15 displays the boundary values for the test statistic. The test statistic 7.31365 is larger than the corresponding upper α boundary 5.79334, so the study stops and rejects the null hypothesis. That is, there is evidence of reduction in hazard rate for the new treatment.

Output 111.6.15 Sequential Tests

The SEQTEST Procedure

Test Information (Score Scale)
Null Reference = 0

	Information Level		Alternative Reference	Boundary Values		Test TrtGp	
Stage	Proportion	Actual	Upper	Upper Beta	Upper Alpha	Estimate	Action
1	0.2389	3.991698	2.76683	-1.03862	4.71423	3.20040	Continue
2	0.5205	8.696125	6.02769	2.17041	5.79334	7.31365	Reject Null
3	0.7500	12.52979	8.68499	4.76308	6.72917	.	
4	1.0000	16.70638	11.57998	7.81290	7.81290	.	

With ODS Graphics enabled, the "Test Plot" displays boundary values of the design and the test statistic at the first two stages, as shown in Output 111.6.16. It also shows that the test statistic is in the "Rejection Region" above the upper α boundary value at stage 2.

Output 111.6.16 Sequential Test Plot

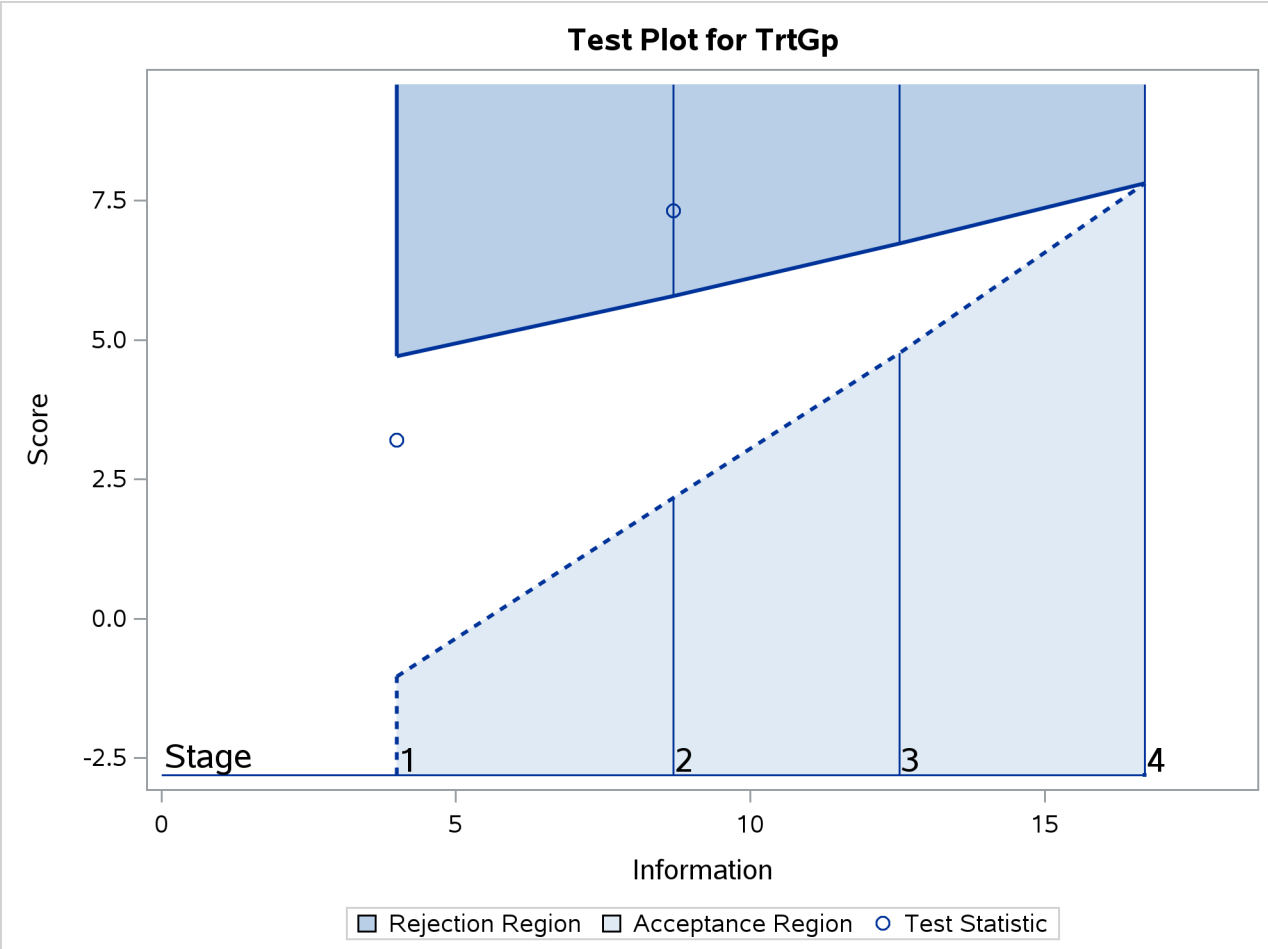

After the stopping of a trial, the "Parameter Estimates" table in Output 111.6.17 displays the stopping stage, parameter estimate, unbiased median estimate, confidence limits, and p-value under the null hypothesis $H_0 : \theta = 0$.

Output 111.6.17 Parameter Estimates

Parameter Estimates Stagewise Ordering					
Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	Lower 95% CL
TrtGp	2	0.841024	0.0139	0.810329	0.21615

As expected, the p-value 0.0139 is significant at the $\alpha = 0.05$ level and the lower 95% confidence limit is larger than $\theta_0 = 0$. The p-value, unbiased median estimate, and lower confidence limit depend on the ordering of the sample space (k, z), where k is the stage number and z is the standardized Z statistic. With the specified stagewise ordering, the p-value is $p_1 + p_2$, where p_1 is the α spending at stage 1,

$$p_1 = P_{\theta=0}(Z_1\sqrt{I_1} \geq 4.71423) = 0.00915$$

$$p_2 = P_{\theta=0}(Z_2\sqrt{I_2} \geq 7.31365 \mid -1.03862 < Z_1\sqrt{I_1} < 4.71423)$$

where Z_k is a standardized normal variate and I_k is the information level at stage k for $k = 1, 2$.

Example 111.7: Testing an Effect in a Proportional Hazards Regression Model

(View the complete code for this example (seqtex7.sas) in the example repository.)

This example compares two survival distributions for the treatment effect. The example uses a power family method to generate two-sided asymmetric boundaries and then uses a proportional hazards regression model to test the hypothesis with a covariate.

A clinic is conducting a clinical study for the effect of a new cancer treatment. The study consists of mice exposed to a carcinogen and randomized to either the control group or the treatment group. The event of interest is the death from cancer induced by the carcinogen, and the response is the time from randomization to death.

Consider the proportional hazards regression model

$$h(t;\ \mathrm{TrtGp},\ \mathrm{Wgt}) = h_0(t)\ \exp(\beta_g\ \mathrm{TrtGp}\ +\ \beta_w \mathrm{Wgt})$$

where $h_0(t)$ is an arbitrary and unspecified baseline hazard function, TrtGp is the grouping variable for the two groups, Wgt is the initial weight of the mice, and β_g and β_w are the regression parameters associated with the variables TrtGp and Wgt, respectively. The grouping variable has the value 0 for each mouse in the control group and the value 1 for each mouse in the treatment group.

The hypothesis $H_0 : \beta_g = 0$ with an alternative hypothesis $H_1 : \beta_g \neq 0$ is used for the study.

Suppose that from past experience, the median survival time for the control group is $t_0 = 20$ weeks. The study would like to detect a $t_1 = 40$ weeks median survival time with a 80% power in the trial. Assuming exponential survival functions for the two groups, the hazard rates can be computed from

$$S_j(t_j) = e^{-h_j t_j} = \frac{1}{2}$$

where $j = 0, 1$.

Thus, with the hazard rates $h_0 = 0.03466$ and $h_1 = 0.01733$, the hazard ratio $\exp(\beta_g) = h_1/h_0 = 1/2$ and the alternative hypothesis

$$\beta_{g1} = \log(\frac{1}{2}) = -0.69315$$

Following the derivations in the section "Test for a Parameter in the Proportional Hazards Regression Model" on page 9206 in Chapter 110, "The SEQDESIGN Procedure," the required number of events for testing a parameter in $\boldsymbol{\beta}$ is given by

$$D_X = I_X\ \frac{1}{(1 - r_x^2)\ \sigma_x^2}$$

where σ_x^2 is the variance of TrtGp and r_x^2 is the proportion of variance of TrtGp explained by the variable Wgt.

If the two groups have the same number of mice in the study, then the MLE of the variance is $\hat{\sigma}_x^2 = 0.25$. Further, if $r_x^2 = 0.10$, then you can specify the MODEL=PHREG(XVARIANCE=0.25 XRSQUARE=0.10)

option in the SAMPLESIZE statement in the SEQDESIGN procedure to compute the required number of events and the individual number of events at each stage.

The following statements invoke the SEQDESIGN procedure and request a four-stage group sequential design for normally distributed data. The design uses a two-sided alternative hypothesis with early stopping to reject the null hypothesis H_0. A power family method is used to derive the boundaries.

```
ods graphics on;
proc seqdesign altref=0.69315;
   TwoSidedPowerFamily: design method=pow
                               nstages=4
                               alpha=0.075(lower=0.025)
                               beta=0.20;
   samplesize model=phreg( xvariance=0.25 xrsquare=0.10
                           hazard=0.02451 accrate=10);
run;
```

The ALPHA=0.075(LOWER=0.025) option specifies a lower α level 0.025 for the lower rejection boundary and an upper α level $0.05 = 0.075 - 0.025$ for the upper rejection boundary. The geometric average hazard $\sqrt{h_0 \times h_1} = \sqrt{0.03466 \times 0.01733} = 0.02451$ is used in the HAZARD= option in the SAMPLESIZE statement to compute the required sample size. The specified ACCRATE=10 option indicates that 10 mice will be accrued each week and the resulting minimum and maximum accrual times will be displayed.

The "Design Information" table in Output 111.7.1 displays the design specifications and the derived statistics.

Output 111.7.1 Design Information

The SEQDESIGN Procedure
Design: TwoSidedPowerFamily

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Method	Power Family
Boundary Key	Both
Alternative Reference	0.69315
Number of Stages	4
Alpha	0.075
Alpha (Lower)	0.025
Alpha (Upper)	0.05
Beta (Lower)	0.2
Beta (Upper)	0.12764
Power (Lower)	0.8
Power (Upper)	0.87236
Max Information (Percent of Fixed Sample)	106.468
Max Information	17.39288
Null Ref ASN (Percent of Fixed Sample)	104.3691
Lower Alt Ref ASN (Number of Events)	58.04014
Upper Alt Ref ASN (Number of Events)	52.05395

The "Boundary Information" table in Output 111.7.2 displays the information level, alternative reference, and boundary values at each stage. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the procedure displays the output boundaries with the standardized Z statistic.

Output 111.7.2 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference		Boundary Values	
Stage	Proportion	Actual	Events	Lower	Upper	Lower Alpha	Upper Alpha
1	0.2500	4.348221	19.32543	-1.44538	1.44538	-2.98871	2.59149
2	0.5000	8.696441	38.65085	-2.04408	2.04408	-2.51320	2.17917
3	0.7500	13.04466	57.97628	-2.50348	2.50348	-2.27093	1.96910
4	1.0000	17.39288	77.3017	-2.89077	2.89077	-2.11334	1.83246

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.7.3.

Output 111.7.3 Boundary Plot

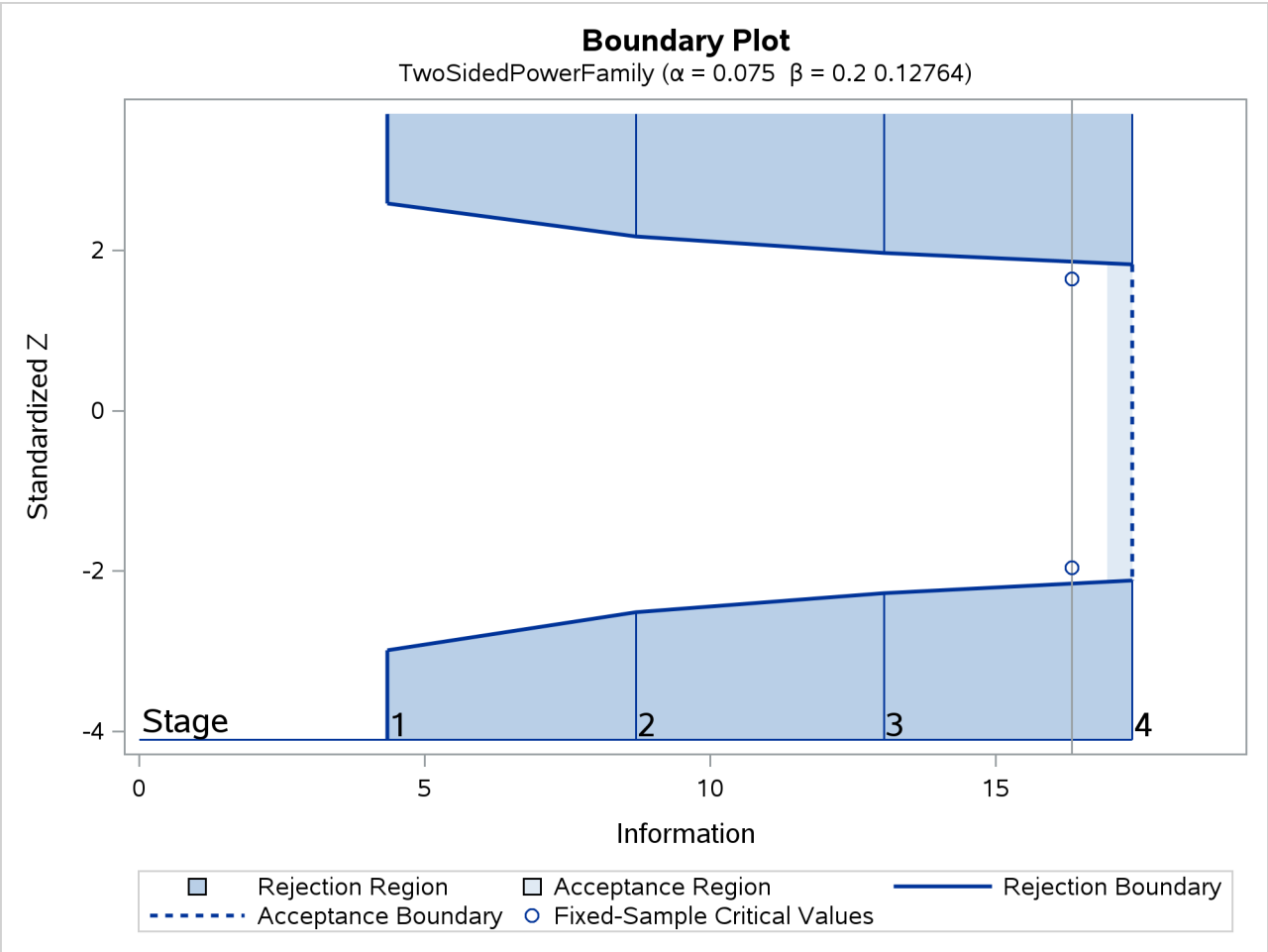

With the MODEL=PHREG option in the SAMPLESIZE statement, the "Sample Size Summary" table in Output 111.7.4 displays the parameters used in the sample size computation for the proportional hazards regression model.

Output 111.7.4 Required Sample Size Summary

Sample Size Summary	
Test	PH Reg Parameter
Parameter	0.69315
X Variance	0.25
R Square (X)	0.1
Hazard Rate	0.02451
Accrual	Uniform
Accrual Rate	10
Min Accrual Time	7.73017
Min Sample Size	77.3017
Max Accrual Time	27.97872
Max Sample Size	279.7872
Max Number of Events	77.3017

With a minimum accrual time of 7.73 weeks and maximum accrual time of 27.98 weeks, an accrual time of 20 weeks is used in the study. The "Numbers of Events" table in Output 111.7.5 displays the required numbers of events for the group sequential clinical trial.

Output 111.7.5 Required Sample Sizes

Numbers of Events (D) Z Test for PH Regression Parameter		
Stage	**D**	**Information**
1	19.33	4.3482
2	38.65	8.6964
3	57.98	13.0447
4	77.30	17.3929

The following statements invoke the SEQDESIGN procedure and provide more detailed sample size information with a 20-week accrual time:

```
proc seqdesign altref=0.69315;
   TwoSidedPowerFamily: design method=pow
                               nstages=4
                               alpha=0.075(lower=0.025)
                               beta=0.20;
   samplesize model=phreg( xvariance=0.25 xrsquare=0.10
                           hazard=0.02451
                           accrate=10 acctime=20);
   ods output Boundary=Bnd_Time;
run;
```

The ODS OUTPUT statement with the BOUNDARY=BND_TIME option creates an output data set named BND_TIME which contains the resulting boundary information for the subsequent sequential tests.

With an accrual time of 20 weeks, the "Sample Size Summary" table in Output 111.7.6 displays the follow-up time for the trial.

Output 111.7.6 Sample Size Summary

The SEQDESIGN Procedure
Design: TwoSidedPowerFamily

Sample Size Summary	
Test	PH Reg Parameter
Parameter	0.69315
X Variance	0.25
R Square (X)	0.1
Hazard Rate	0.02451
Accrual	Uniform
Accrual Rate	10
Accrual Time	20
Follow-up Time	10.34195
Total Time	30.34195
Max Number of Events	77.3017
Max Sample Size	200
Expected Sample Size (Null Ref)	199.4282
Expected Sample Size (Alt Ref)	188.6561
Follow-up Time (Ceiling Time)	11
Total Time (Ceiling Time)	31

The "Numbers of Events and Sample Sizes" table in Output 111.7.7 displays the required sample sizes for the group sequential clinical trial.

Output 111.7.7 Numbers of Events and Sample Sizes

Numbers of Events (D) and Sample Sizes (N)
Z Test for PH Regression Parameter

	Fractional Time				Ceiling Time			
Stage	D	Time	N	Information	D	Time	N	Information
1	19.33	13.2362	132.36	4.3482	21.49	14	140.00	4.8359
2	38.65	19.1466	191.47	8.6964	41.90	20	200.00	9.4281
3	57.98	24.3744	200.00	13.0447	60.14	25	200.00	13.5309
4	77.30	30.3420	200.00	17.3929	79.26	31	200.00	17.8346

Thus, the study will perform three interim analyses after 14, 20, and 25 weeks and a final analysis after 31 weeks if the study does not stop at any of the interim analyses.

Suppose 140 mice are available for the first interim analysis after week 14. Output 111.7.8 lists the first 10 observations in the data set weeks_1.

Output 111.7.8 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	TrtGp	Event	Wgt	Weeks
1	0	0	22.1659	12
2	1	0	28.4458	12
3	0	0	26.2857	12
4	1	0	25.0283	12
5	0	0	21.5114	12
6	1	0	23.2240	12
7	0	1	22.6845	6
8	1	0	27.9292	12
9	0	0	22.5514	12
10	1	1	27.3793	11

The TrtGp variable is a grouping variable with the value 0 for a mouse in the placebo control group and the value 1 for a mouse in the treatment group.

The Weeks variable is the survival time variable measured in weeks and the Event variable is the censoring variable with the value 0 indicating censoring. That is, the values of Weeks are considered censored if the corresponding values of Event are 0; otherwise, they are considered as event times.

The following statements use the PHREG procedure to estimate the treatment effect after adjusting for the Wgt variable at stage 1:

```
proc phreg data=Time_1;
   model Weeks*Event(0)= TrtGp Wgt;
   ods output parameterestimates=Parms_Time1;
run;
```

The following statements create and display (in Output 111.7.9) the data set for the treatment effect MLE statistic and its associated standard error. Note that for a MLE statistic, the inverse of the variance of the statistic is the information.

```
data Parms_Time1;
   set Parms_Time1;
   if Parameter='TrtGp';
   _Scale_='MLE';
   _Stage_= 1;
   keep _Scale_ _Stage_ Parameter Estimate StdErr;
run;

proc print data=Parms_Time1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.7.9 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Parameter	Estimate	StdErr	_Scale_	_Stage_
1	TrtGp	0.00836	0.46588	MLE	1

The following statements invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Time
             Parms(Testvar=TrtGp)=Parms_Time1
             infoadj=prop
             order=lr
             ;
   ods output Test=Test_Time1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The PARMS=PARMS_TIME1 option specifies the input data set PARMS_TIME1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=TRTGP option identifies the test variable TRTGP in the data set.

If the computed information level for stage 1 is not the same as the value provided in the BOUNDARY= data set, the INFOADJ=PROP option (which is the default) proportionally adjusts the information levels at future interim stages from the levels provided in the BOUNDARY= data set. The ORDER=LR option uses the LR ordering to derive the p-value, the unbiased median estimate, and the confidence limits for the regression slope estimate.

The ODS OUTPUT statement with the TEST=TEST_TIME1 option creates an output data set named TEST_TIME1 which contains the updated boundary information for the test at stage 1. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.7.10 displays design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary values are modified for the new information levels to maintain the Type I α level. Since the computed information level at stage 1 is not the same as the value provided in the BOUNDARY= data set, the power has been modified.

Output 111.7.10 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_TIME
Data Set	WORK.PARMS_TIME1
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Reject Null
Number of Stages	4
Alpha	0.075
Alpha (Lower)	0.025
Alpha (Upper)	0.05
Beta (Lower)	0.20048
Beta (Upper)	0.12795
Power (Lower)	0.79952
Power (Upper)	0.87205
Max Information (Percent of Fixed Sample)	106.5982
Max Information	17.3928828
Null Ref ASN (Percent of Fixed Sample)	104.4715
Lower Alt Ref ASN (Percent of Fixed Sample)	79.7886
Upper Alt Ref ASN (Percent of Fixed Sample)	71.53877

The "Test Information" table in Output 111.7.11 displays the boundary values for the test statistic with the MLE statistic scale.

Output 111.7.11 Sequential Tests

Test Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values		Test TrtGp	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Upper Alpha	Estimate	Action
1	0.2649	4.607347	-1.48783	1.48783	-2.92457	2.54086	0.01795	Continue
2	0.5099	8.869192	-2.06428	2.06428	-2.50505	2.17290	.	
3	0.7550	13.13104	-2.51175	2.51175	-2.27093	1.96941	.	
4	1.0000	17.39288	-2.89077	2.89077	-2.11635	1.83531	.	

With the INFOADJ=PROP option (which is the default), the information levels at interim stages 2 and 3 are derived proportionally from the information levels in the BOUNDARY= data set. At stage 1, the standardized Z statistic 0.01795 is between the lower and upper α boundary values of –2.92457 and 2.54086, so the trial continues to the next stage.

Note that the observed information level 4.6073 corresponds to a proportion of 0.2649 in the information level. If the observed information level is much larger than the target proportion of 0.25, then you can decrease the accrual rate, accrual time, or follow-up time to achieve target information levels for subsequent stages. These modifications should be specified in the study plan before the study begins.

With ODS Graphics enabled, a boundary plot with test statistics is displayed, as shown in Output 111.7.12. As expected, the test statistic is in the continuation region between the lower and upper α boundary values.

Output 111.7.12 Sequential Test Plot

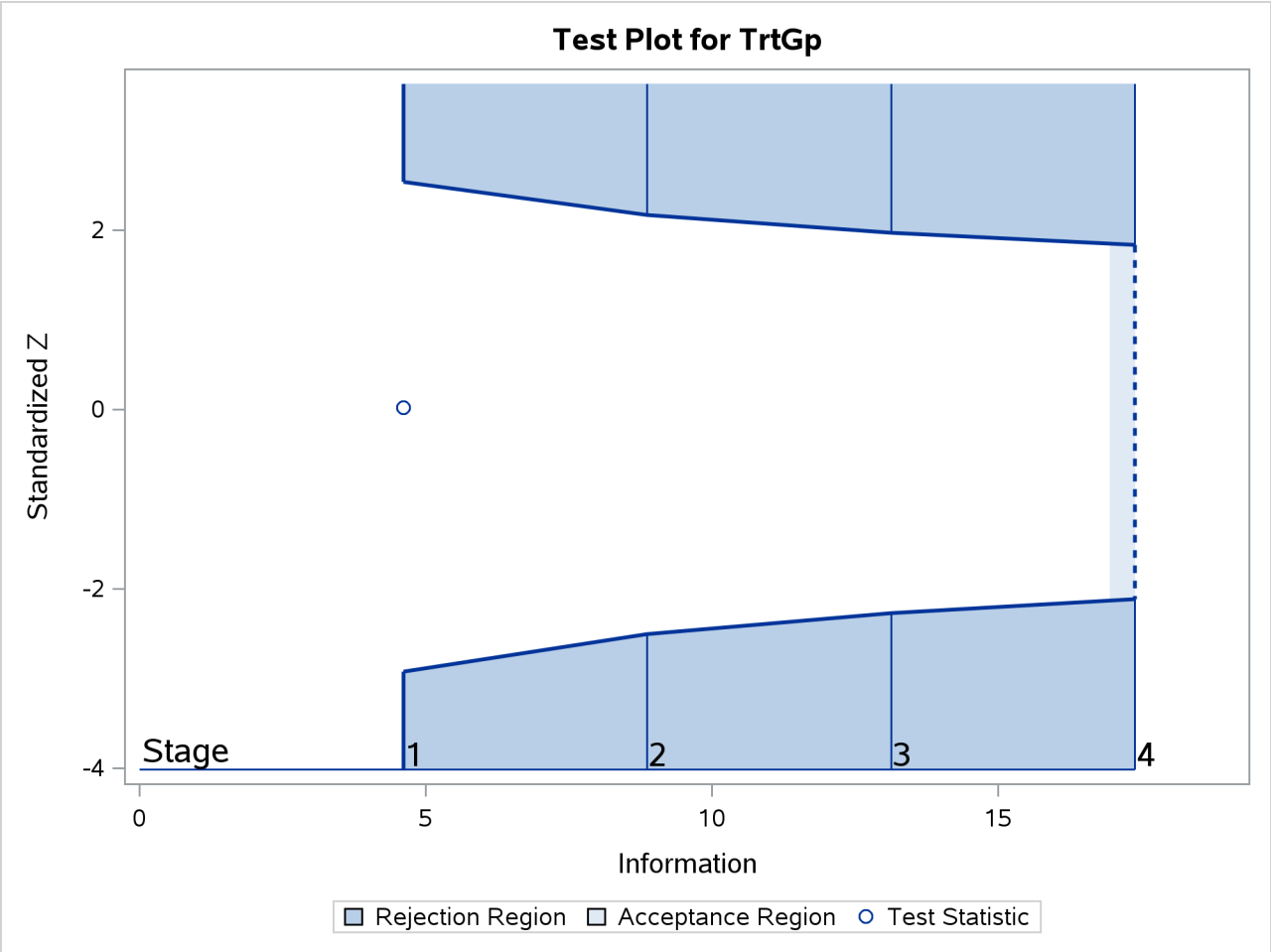

The following statements use the PHREG procedure to compute the MLE statistic and its associated standard error at stage 2:

```
proc phreg data=Time_2;
   model Weeks*Event(0)= TrtGp Wgt;
   ods output parameterestimates= Parms_Time2;
run;
```

The following statements create the data set for the MLE statistic and its associated standard error at stage 2:

```
data Parms_Time2;
   set Parms_Time2;
   if Parameter='TrtGp';
   _Scale_='MLE';
   _Stage_= 2;
   keep _Scale_ _Stage_ Parameter Estimate StdErr;
run;
```

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2:

```
proc seqtest Boundary=Test_Time1
             Parms(Testvar=TrtGp)=Parms_Time2
             infoadj=prop
             order=lr
             ;
   ods output Test=Test_Time2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=TEST_TIME2 option creates an output data set named TEST_TIME2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Test Information" table in Output 111.7.13 displays the boundary values for the test statistic with the MLE statistic scale. At stage 2, the standardized Z statistic –0.43552 is between the lower α and upper boundary values, –2.47689 and 2.14819, respectively, so the trial continues to the next stage.

Output 111.7.13 Sequential Tests

The SEQTEST Procedure

Test Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values		Test TrtGp	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Upper Alpha	Estimate	Action
1	0.2649	4.607347	-1.48783	1.48783	-2.92457	2.54086	0.01795	Continue
2	0.5251	9.132918	-2.09475	2.09475	-2.47689	2.14819	-0.43552	Continue
3	0.7625	13.2629	-2.52433	2.52433	-2.26878	1.96770	.	
4	1.0000	17.39288	-2.89077	2.89077	-2.12017	1.83880	.	

Since the data set PARMS_Time2 contains the test information only at stage 2, the information level at stage 1 in the TEST_Time1 data set is used to generate boundary values for the test.

Similarly, the test statistic at stage 3 is also between its corresponding lower and upper α boundary values. The trial continues to the next stage.

The following statements use the PHREG procedure to compute the MLE statistic and its associated standard error at the final stage:

```
proc phreg data=Time_4;
   model Weeks*Event(0)= TrtGp Wgt;
   ods output parameterestimates= Parms_Time4;
run;
```

The following statements create and display (in Output 111.7.14) the data set for the MLE statistic and its associated standard error at each stage of the study:

```
data Parms_Time4;
   set Parms_Time4;
   if Parameter='TrtGp';
   _Scale_='MLE';
   _Stage_= 4;
   keep _Scale_ _Stage_ Parameter Estimate StdErr;
run;

proc print data=Parms_Time4;
   title 'Statistics Computed at Stage 4';
run;
```

Output 111.7.14 Statistics Computed at Stage 4

Statistics Computed at Stage 4

Obs	Parameter	Estimate	StdErr	_Scale_	_Stage_
1	TrtGp	-0.04451	0.23971	MLE	4

The following statements invoke the SEQTEST procedure to test the hypothesis at stage 4:

```
ods graphics on;
proc seqtest Boundary=Test_Time3
             Parms(Testvar=TrtGp)=Parms_Time4
             order=lr
             ;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 4, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 4, and the TESTVAR= option identifies the test variable in the data set.

The "Test Information" table in Output 111.7.15 displays the boundary values for the test statistic. The standardized test statistic –0.1857 is between the lower and upper α boundary values of –2.10447 and 1.82112, respectively, so the study stops and accepts the null hypothesis. That is, there is no evidence of reduction in hazard rate for the new treatment.

Output 111.7.15 Sequential Tests

The SEQTEST Procedure

Test Information (Standardized Z Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values		Test TrtGp	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Upper Alpha	Estimate	Action
1	0.2647	4.607347	-1.48783	1.48783	-2.92457	2.54086	0.01795	Continue
2	0.5248	9.132918	-2.09475	2.09475	-2.47689	2.14819	-0.43552	Continue
3	0.7095	12.34753	-2.43566	2.43566	-2.32705	2.02634	0.34864	Continue
4	1.0000	17.40274	-2.89159	2.89159	-2.10447	1.82112	-0.18570	Accept Null

The "Test Plot" displays boundary values of the design and the test statistic at the first two stages, as shown in Output 111.7.16. It also shows that the test statistic is in the "Acceptance Region" between the lower and upper α boundary values at stage 4.

Output 111.7.16 Sequential Test Plot

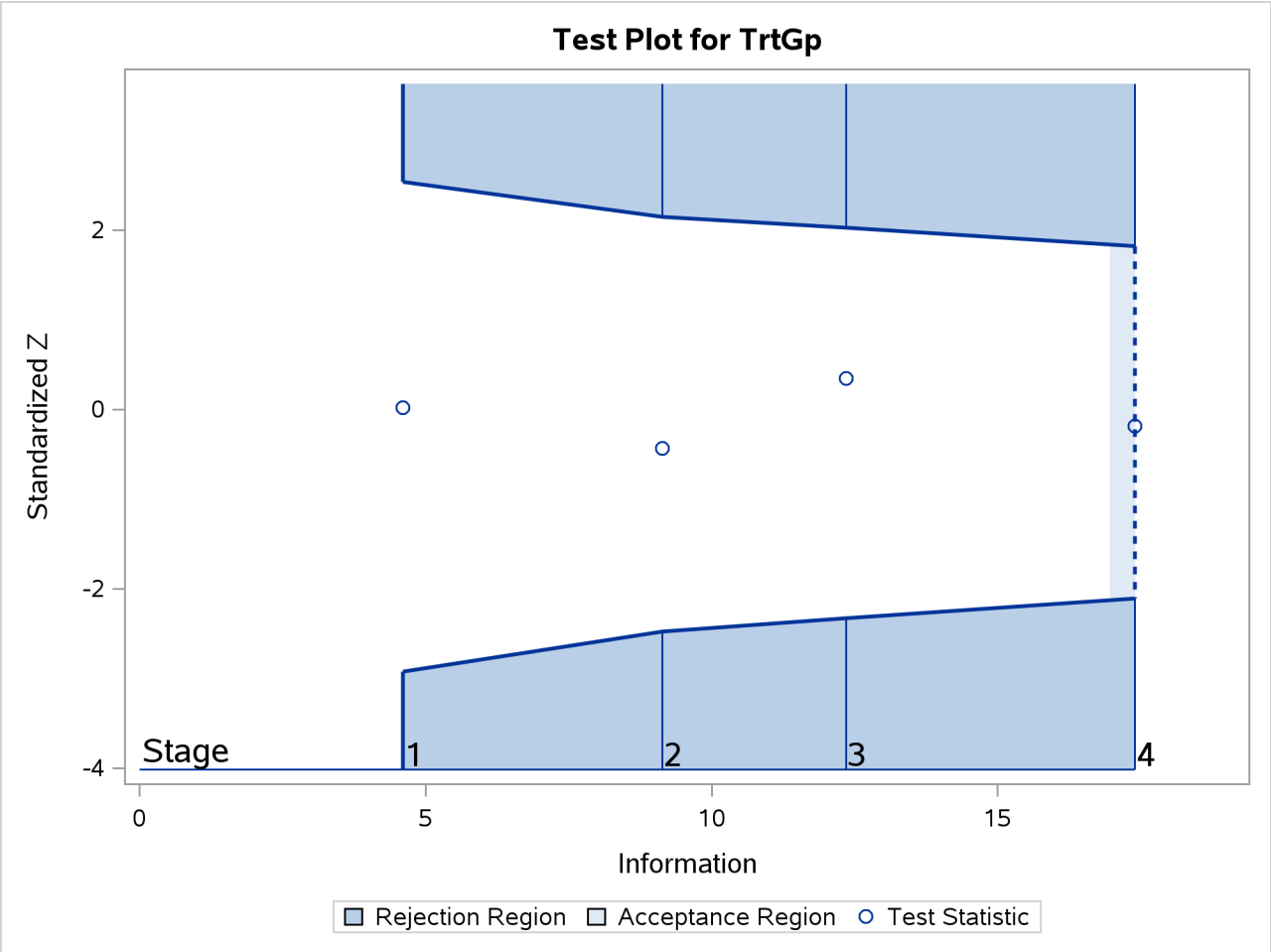

After the stopping of a trial, the "Parameter Estimates" table in Output 111.7.17 displays the stopping stage, parameter estimate, unbiased median estimate, confidence limits, and p-value under the null hypothesis $H_0 : \theta = 0$.

Output 111.7.17 Parameter Estimates

Parameter Estimates LR Ordering						
Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	95% Confidence Limits	
TrtGp	4	-0.044514	0.8525	-0.044577	-0.51461	0.42538

As expected, the two-sided p-value 0.8525 is not significant at the lower $\alpha = 0.025$ level and the upper $\alpha = 0.05$ level, and the two-sided 95% confidence interval contains the null value zero. The p-value, unbiased median estimate, and lower confidence limit depend on the ordering of the sample space (k, z), where k is

the stage number and z is the standardized Z statistic. With the specified LR ordering, the two-sided p-value is derived from the one-sided p-value

$$p_u = \sum_{k=1}^{4} P_{\theta=0}\left(Z_k \geq z_4 \mid _a_{k'} < Z_{k'} < a_{k'}, k' < k\right)$$

where $z_4 = -0.1857$ is the observed test statistic at stage 4, Z_k is a standardized normal variate at stage k, and $_a_{k'}$ and $a_{k'}$ are the stage k lower and upper rejection boundary values, respectively.

Thus,

$$p_u = \alpha_u + P_{\theta=0}\left(z_4 \leq Z_4 < a_4 \mid _a_{k'} < Z_{k'} < a_{k'}, k' < 4\right)$$

where $\alpha_u = 0.05$ is the upper α level and $a_4 = 1.82112$.

Since $P_{\theta=0}\left(z_4 \leq Z_4 \leq a_4 \mid _a_{k'} < Z_{k'} < a_{k'}, k' < 4\right) = 0.52374$, $p_u = 0.05 + 0.52374 = 0.57374$, which is greater than 0.50. Thus, the two-sided p-value is given by $2 \times (1.0 - p_u) = 0.8525$.

Example 111.8: Testing an Effect in a Logistic Regression Model

(View the complete code for this example (seqtex8.sas) in the example repository.)

This example requests a two-sided test for the dose effect in a dose-response model (Whitehead 1997, pp. 262–263). Consider the logistic regression model

$$\text{logit}(p) = \log(\frac{p}{1-p}) = \beta_0 + \beta_1 \text{ LDose}$$

where $p = \text{Prob}(\text{Resp} = 1 \mid \text{LDose})$ is the response probability to be modeled for the binary response Resp and LDose = log(Dose +1) is the covariate. The dose levels are 0 for the control group, and they are 1, 3, and 6 for the three treatment groups.

Following the derivations in the section "Test for a Parameter in the Logistic Regression Model" on page 9205 in Chapter 110, "The SEQDESIGN Procedure," the required sample size can be derived from

$$N = I_X \frac{\sigma_y^2}{(1 - r_x^2)\,\sigma_x^2}$$

where σ_y^2 is the variance of the response variable in the logistic regression model, r_x^2 is the proportion of variance of LDose explained by other covariates, and σ_x^2 is the variance of LDose.

Since LDose is the only covariate in the model, $r_x^2 = 0$. For a logistic model, the variance σ^2 can be estimated by

$$\sigma_y^2 = \frac{1}{\hat{p}(1-\hat{p})}$$

where $\hat{p}$ is the estimated probability of the response variable Resp. Thus, the sample size can be computed as

$$N = I_X \frac{1}{p(1-p)} \frac{1}{\sigma_x^2}$$

The null hypothesis $H_0 : \beta_1 = 0$ corresponds to no treatment effect. Suppose that the alternative hypothesis $H_1 : \beta_1 = 0.5$ is the reference improvement that should be detected at a 0.90 level.

Note that $\beta_1 = 0.5$ corresponds to an odds ratio of 2 between the treatment group with dose level 3 and the control group. The log odds ratio between the two groups is

$$\log\left(\frac{p_t(1-p_c)}{(1-p_t)p_0}\right) = \log\left(\frac{p_t}{1-p_t}\right) - \log\left(\frac{p_c}{1-p_c}\right)$$

which corresponds to

$$(\beta_0 + \beta_1 \log(3+1)) - (\beta_0 + \beta_1 \log(1)) = \beta_1 \log(4) = \log(2)$$

If the same number of patients are assigned in each of the four groups, then the MLE of the variance of LDose is $\hat{\sigma}_x^2 = 0.5345$. Further, if the response rate is 0.40, then the required sample size can be derived using the SAMPLESIZE statement in the SEQDESIGN procedure.

The following statements invoke the SEQDESIGN procedure and request a three-stage group sequential design for normally distributed data. The design has a null hypothesis of no treatment effect $H_0 : \beta_1 = 0$ with early stopping to reject the null hypothesis with a two-sided alternative hypothesis $H_1 : \beta_1 = \pm 0.5$.

```
ods graphics on;
proc seqdesign altref=0.5;
   TwoSidedErrorSpending: design method=errfuncpow
                                 method(loweralpha)=errfuncpow(rho=1)
                                 method(upperalpha)=errfuncpow(rho=3)
                                 nstages=3
                                 stop=both;
   samplesize model=logistic( prop=0.4 xvariance=0.5345);
   ods output Boundary=Bnd_Dose;
run;
```

The ODS OUTPUT statement with the BOUNDARY=BND_DOSE option creates an output data set named BND_DOSE which contains the resulting boundary information for the subsequent sequential tests.

The “Design Information” table in Output 111.8.1 displays design specifications and derived statistics. Since the alternative reference is specified, the maximum information 47.22445 is derived.

Output 111.8.1 Error Spending Design Information

The SEQDESIGN Procedure
Design: TwoSidedErrorSpending

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Two-Sided
Early Stop	Accept/Reject Null
Method	Error Spending
Boundary Key	Both
Alternative Reference	0.5
Number of Stages	3
Alpha	0.05
Beta (Lower)	0.1
Beta (Upper)	0.07871
Power (Lower)	0.9
Power (Upper)	0.92129
Max Information (Percent of Fixed Sample)	103.7223
Max Information	47.22445
Null Ref ASN (Percent of Fixed Sample)	79.47628
Lower Alt Ref ASN (Sample Size)	234.1646
Upper Alt Ref ASN (Sample Size)	270.4058

The "Boundary Information" table in Output 111.8.2 displays the information level, alternative reference, and boundary values at each stage. By default (or equivalently if you specify BOUNDARYSCALE=STDZ), the boundary values are displayed with the standardized Z statistic scale.

Output 111.8.2 Boundary Information

Boundary Information (Standardized Z Scale)
Null Reference = 0

	Information Level			Alternative Reference		Boundary Values			
						Lower		Upper	
Stage	Proportion	Actual	N	Lower	Upper	Alpha	Beta	Beta	Alpha
1	0.3333	15.74148	122.7119	-1.98378	1.98378	-2.39398	.	.	3.11302
2	0.6667	31.48297	245.4238	-2.80548	2.80548	-2.29380	-1.02812	0.91855	2.46193
3	1.0000	47.22445	368.1357	-3.43600	3.43600	-2.17479	-2.17479	1.98311	1.98311

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.8.3.

Output 111.8.3 Boundary Plot

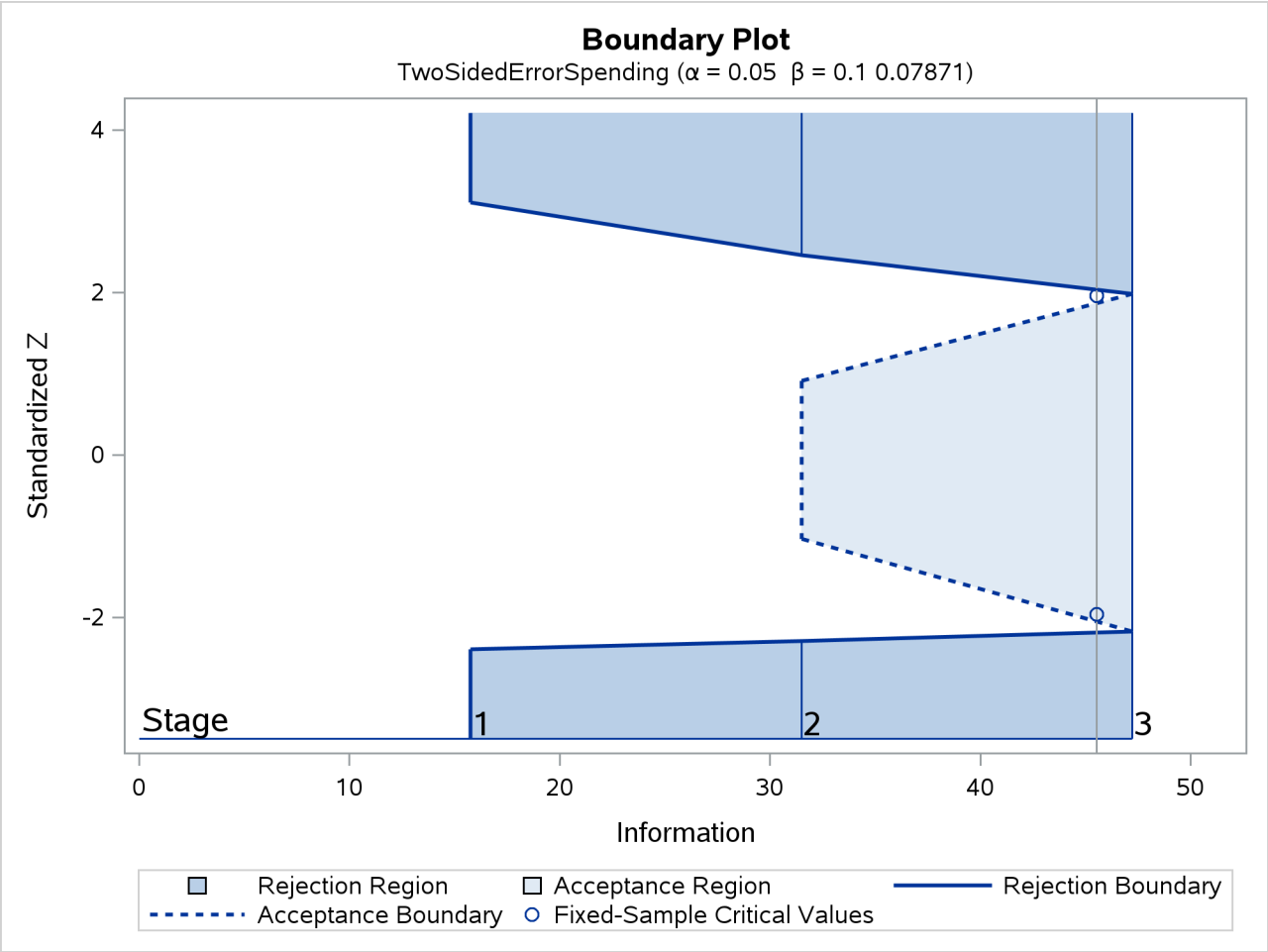

With the SAMPLESIZE statement, the "Sample Size Summary" table in Output 111.8.4 displays the parameters for the sample size computation.

Output 111.8.4 Required Sample Size Summary

Sample Size Summary	
Test	Logistic Reg Parameter
Parameter	0.5
Proportion	0.4
X Variance	0.5345
R Square (X)	0
Max Sample Size	368.1357
Expected Sample Size (Null Ref)	282.0807
Expected Sample Size (Alt Ref)	270.4058

The "Sample Sizes" table in Output 111.8.5 displays the required sample sizes for the group sequential clinical trial.

Output 111.8.5 Required Sample Sizes

Sample Sizes (N) Z Test for Logistic Regression Parameter				
	Fractional N		Ceiling N	
Stage	N	Information	N	Information
1	122.71	15.7415	123	15.7784
2	245.42	31.4830	246	31.5569
3	368.14	47.2245	369	47.3353

That is, 123 new patients are needed in each stage and the number is rounded up to 124 for each stage to have a multiple of four for the four dose levels in the trial. Note that since the sample sizes are derived from an estimated response probability and are rounded up, the actual information levels might not match the corresponding target information levels.

Output 111.8.6 lists the first 10 observations of the trial data.

Output 111.8.6 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Resp	Dose	LDose
1	0	0	0.00000
2	0	1	0.69315
3	1	3	1.38629
4	1	6	1.94591
5	1	0	0.00000
6	1	1	0.69315
7	1	3	1.38629
8	1	6	1.94591
9	0	0	0.00000
10	0	1	0.69315

The following statements use the LOGISTIC procedure to estimate the slope β_1 and its associated standard error at stage 1:

```
proc logistic data=Dose_1;
   model Resp(event='1')= LDose;
   ods output ParameterEstimates=Parms_Dose1;
run;
```

The following statements create and display (in Output 111.8.7) the input data set that contains slope β_1 and its associated standard error for the SEQTEST procedure:

```
data Parms_Dose1;
   set Parms_Dose1;
   if Variable='LDose';
   _Scale_='MLE';
   _Stage_= 1;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;
```

```
proc print data=Parms_Dose1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.8.7 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	LDose	0.5741	0.2544	MLE	1

The following statements invoke the SEQTEST procedure to test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Dose
             Parms(testvar=LDose)=Parms_Dose1
             infoadj=prop
             order=mle
             boundaryscale=mle
             ;
   ods output Test=Test_Dose1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The PARMS=PARMS_DOSE1 option specifies the input data set PARMS_DOSE1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=LDOSE option identifies the test variable LDOSE in the data set.

If the computed information level for stage 1 is not the same as the value provided in the BOUNDARY= data set, the INFOADJ=PROP option (which is the default) proportionally adjusts the information levels at future interim stages from the levels provided in the BOUNDARY= data set. The ORDER=MLE option uses the MLE ordering to derive the p-value, the unbiased median estimate, and the confidence limits for the regression slope estimate.

The ODS OUTPUT statement with the TEST=TEST_DOSE1 option creates an output data set named TEST_DOSE1 which contains the updated boundary information for the test at stage 1. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.8.8 displays design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary values are modified for the new information levels to maintain the Type I α level. The maximum information remains the same as the design stored in the BOUNDARY= data set, but the derived Type II error probability β and power $1 - \beta$ are different because of the new information levels.

Output 111.8.8 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_DOSE
Data Set	WORK.PARMS_DOSE1
Statistic Distribution	Normal
Boundary Scale	MLE
Alternative Hypothesis	Two-Sided
Early Stop	Accept/Reject Null
Number of Stages	3
Alpha	0.05
Beta (Lower)	0.09992
Beta (Upper)	0.07871
Power (Lower)	0.90008
Power (Upper)	0.92129
Max Information (Percent of Fixed Sample)	103.7231
Max Information	47.2244524
Null Ref ASN (Percent of Fixed Sample)	79.45049
Lower Alt Ref ASN (Percent of Fixed Sample)	66.05269
Upper Alt Ref ASN (Percent of Fixed Sample)	76.24189

The "Test Information" table in Output 111.8.9 displays the boundary values for the test statistic with the specified MLE scale.

Output 111.8.9 Sequential Tests

Test Information (MLE Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values Lower		Upper		Test LDose	
Stage	Proportion	Actual	Lower	Upper	Alpha	Beta	Beta	Alpha	Estimate	Action
1	0.3272	15.45062	-0.50000	0.50000	-0.61078	.	.	0.79337	0.57409	Continue
2	0.6636	31.33753	-0.50000	0.50000	-0.40974	-0.18169	0.16222	0.44033	.	
3	1.0000	47.22445	-0.50000	0.50000	-0.31633	-0.31633	0.28860	0.28860	.	

The information level at stage 1 is derived from the standard error s_1 in the PARMS= data set,

$$I_1 = \frac{1}{s_1^2} = \frac{1}{0.2544^2} = 15.45$$

With the INFOADJ=PROP option (which is the default), the information level at stage 2 is derived proportionally from the observed information at stage 1 and the information levels in the BOUNDARY= data set. At stage 1, the β boundary values are missing and there is no early stopping to accept H_0. The MLE statistic 0.57409 is between the lower and upper α boundaries, so the trial continues to the next stage.

With ODS Graphics enabled, a boundary plot with the boundary values and test statistics is displayed, as shown in Output 111.8.10. As expected, the test statistic is in the continuation region below the upper alpha boundary.

Output 111.8.10 Sequential Test Plot

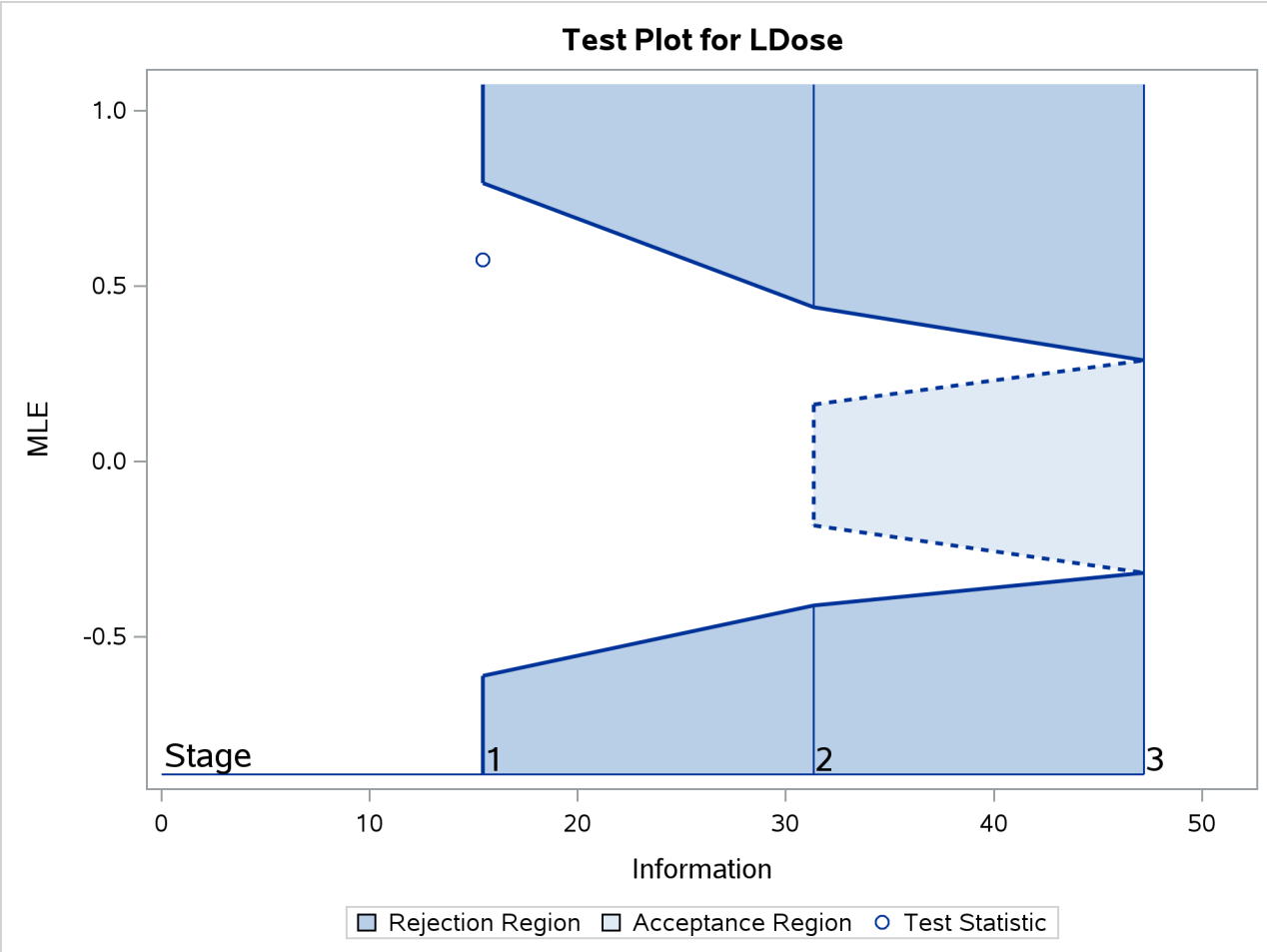

The following statements use the LOGISTIC procedure to estimate the slope β_1 and its associated standard error at stage 2:

```
proc logistic data=dose_2;
   model Resp(event='1')=LDose;
   ods output ParameterEstimates=Parms_Dose2;
run;
```

The following statements create and display (in Output 111.8.11) the input data set that contains slope β_1 and its associated standard error at stage 2 for the SEQTEST procedure:

```
data Parms_Dose2;
   set Parms_Dose2;
   if Variable='LDose';
   _Scale_='MLE';
   _Stage_= 2;
   keep _Scale_ _Stage_ Variable Estimate StdErr;
run;

proc print data=Parms_Dose2;
   title 'Statistics Computed at Stage 2';
```

```
run;
```

Output 111.8.11 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	Variable	Estimate	StdErr	_Scale_	_Stage_
1	LDose	0.5213	0.1788	MLE	2

The following statements invoke the SEQTEST procedure to test for early stopping at stage 2:

```
ods graphics on;
proc seqtest Boundary=Test_Dose1
             Parms(Testvar=LDose)=Parms_Dose2
             infoadj=prop
             order=mle
             boundaryscale=mle
             rci
             plots=rci
             ;
   ods output Test=Test_Dose2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set.

The ORDER=MLE option uses the MLE ordering to derive the p-value, unbiased median estimate, and confidence limits for the regression slope estimate.

The ODS OUTPUT statement with the TEST=TEST_DOSE2 option creates an output data set named TEST_DOSE2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.8.12 displays design specifications. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the boundary values are modified for the new information levels to maintain the Type I α level.

Output 111.8.12 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.TEST_DOSE1
Data Set	WORK.PARMS_DOSE2
Statistic Distribution	Normal
Boundary Scale	MLE
Alternative Hypothesis	Two-Sided
Early Stop	Accept/Reject Null
Number of Stages	3
Alpha	0.05
Beta (Lower)	0.0999
Beta (Upper)	0.07871
Power (Lower)	0.9001
Power (Upper)	0.92129
Max Information (Percent of Fixed Sample)	103.7227
Max Information	47.2244524
Null Ref ASN (Percent of Fixed Sample)	79.44086
Lower Alt Ref ASN (Percent of Fixed Sample)	66.04641
Upper Alt Ref ASN (Percent of Fixed Sample)	76.22739

The information is derived from the standard error associated with the slope estimate at the final stage and is larger than the target level. The derived Type II error probability β and power $1 - \beta$ are different because of the new information levels.

The "Test Information" table in Output 111.8.13 displays the boundary values for the test statistic with the specified MLE scale. The information levels are derived from the standard errors in the PARMS= data set. At stage 2, the slope estimate 0.52128 is larger than 0.44091, the upper α boundary value, the trial stops to reject the null hypothesis of no treatment effect.

Output 111.8.13 Sequential Tests

Test Information (MLE Scale)
Null Reference = 0

	Information Level		Alternative Reference		Boundary Values				Test LDose	
Stage	Proportion	Actual	Lower	Upper	Lower Alpha	Lower Beta	Upper Beta	Upper Alpha	Estimate	Action
1	0.3272	15.45062	-0.50000	0.50000	-0.61078	.	.	0.79337	0.57409	Continue
2	0.6624	31.28346	-0.50000	0.50000	-0.41028	-0.18112	0.16166	0.44091	0.52128	Reject Null
3	1.0000	47.22445	-0.50000	0.50000	-0.31628	-0.31628	0.28861	0.28861	.	

With ODS Graphics enabled, a boundary plot with the boundary values and test statistics is displayed, as shown in Output 111.8.14. As expected, the test statistic is above the upper α boundary in the upper rejection region at stage 2.

Output 111.8.14 Sequential Test Plot

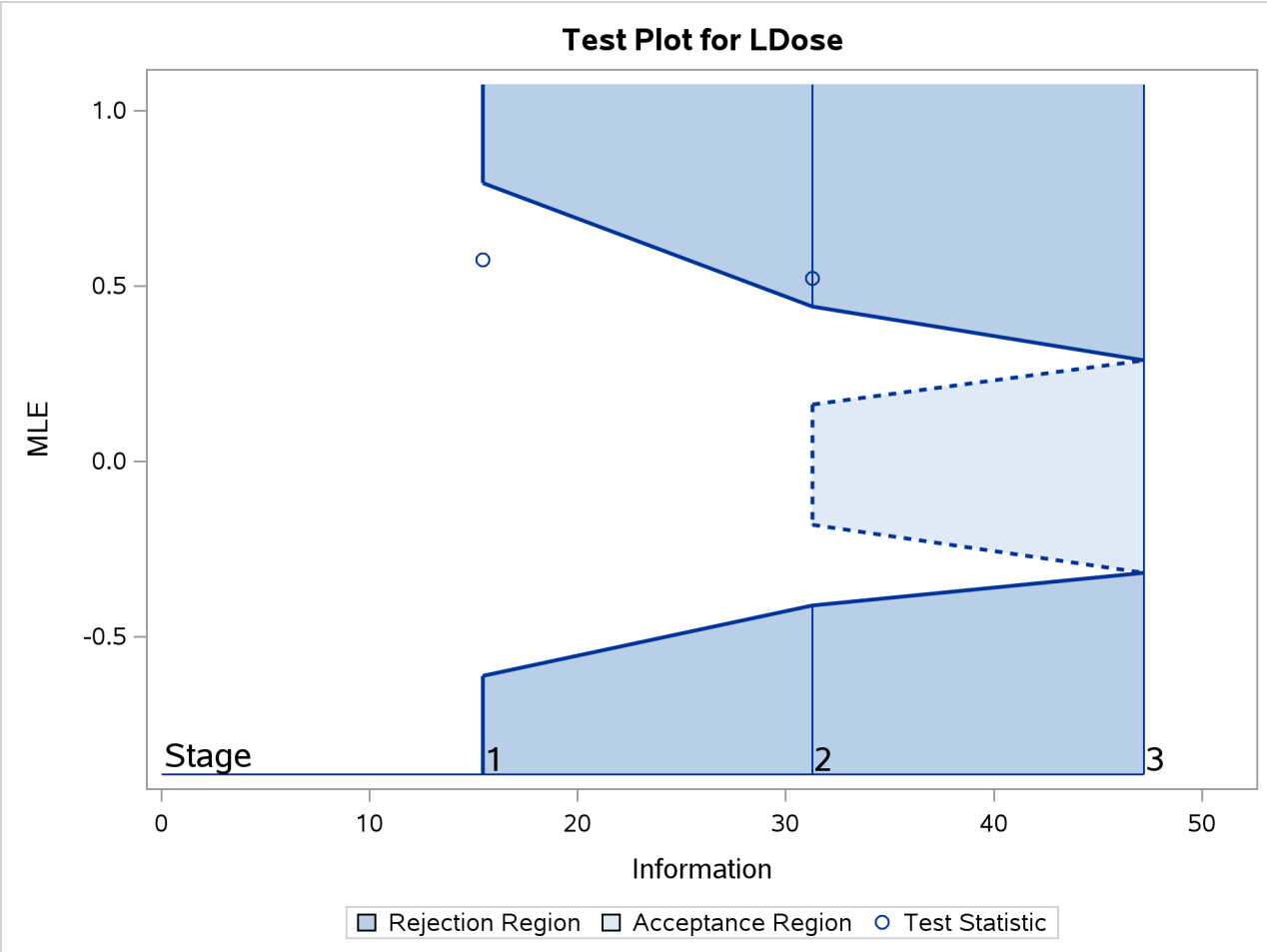

After a trial is stopped, the "Parameter Estimates" table in Output 111.8.15 displays the stopping stage, parameter estimate, unbiased median estimate, confidence limits, and the p-value under the null hypothesis $H_0 : \beta_1 = 0$.

Output 111.8.15 Parameter Estimates

Parameter Estimates MLE Ordering						
Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	95% Confidence Limits	
LDose	2	0.521275	0.0050	0.502647	0.15745	0.85154

With the ORDER=MLE option, the MLE ordering is used to compute the p-value, unbiased median estimate, and confidence limits. As expected, the p-value 0.005 is significant at the $\alpha = 0.05$ level and the confidence interval does not contain the null reference zero.

With the RCI option, the "Repeated Confidence Intervals" table in Output 111.8.16 displays repeated confidence intervals for the parameter. For a two-sided test, since the rejection lower repeated confidence limit 0.0804 is greater than the null reference zero, the trial is stopped to reject the hypothesis.

Output 111.8.16 Repeated Confidence Intervals

Repeated Confidence Intervals				
			Rejection Boundary	
Stage	Information Level	Parameter Estimate	Lower 97.5% Repeated CL	Upper 97.5% Repeated CL
1	15.4506	0.57409	-0.2193	1.1849
2	31.2835	0.52128	0.0804	0.9316

Repeated Confidence Intervals		
	Acceptance Boundary	
Stage	Lower 90.01% Repeated CL	Upper 92.13% Repeated CL
1	.	.
2	0.2024	0.8596

With the PLOTS=RCI option, the "Repeated Confidence Intervals Plot" displays repeated confidence intervals for the parameter, as shown in Output 111.8.17. It shows that the null reference zero is inside the rejection repeated confidence interval at stage 1 but outside the rejection repeated confidence interval at stage 2. This implies that the study stops at stage 2 to reject the hypothesis.

Output 111.8.17 Repeated Confidence Intervals Plot

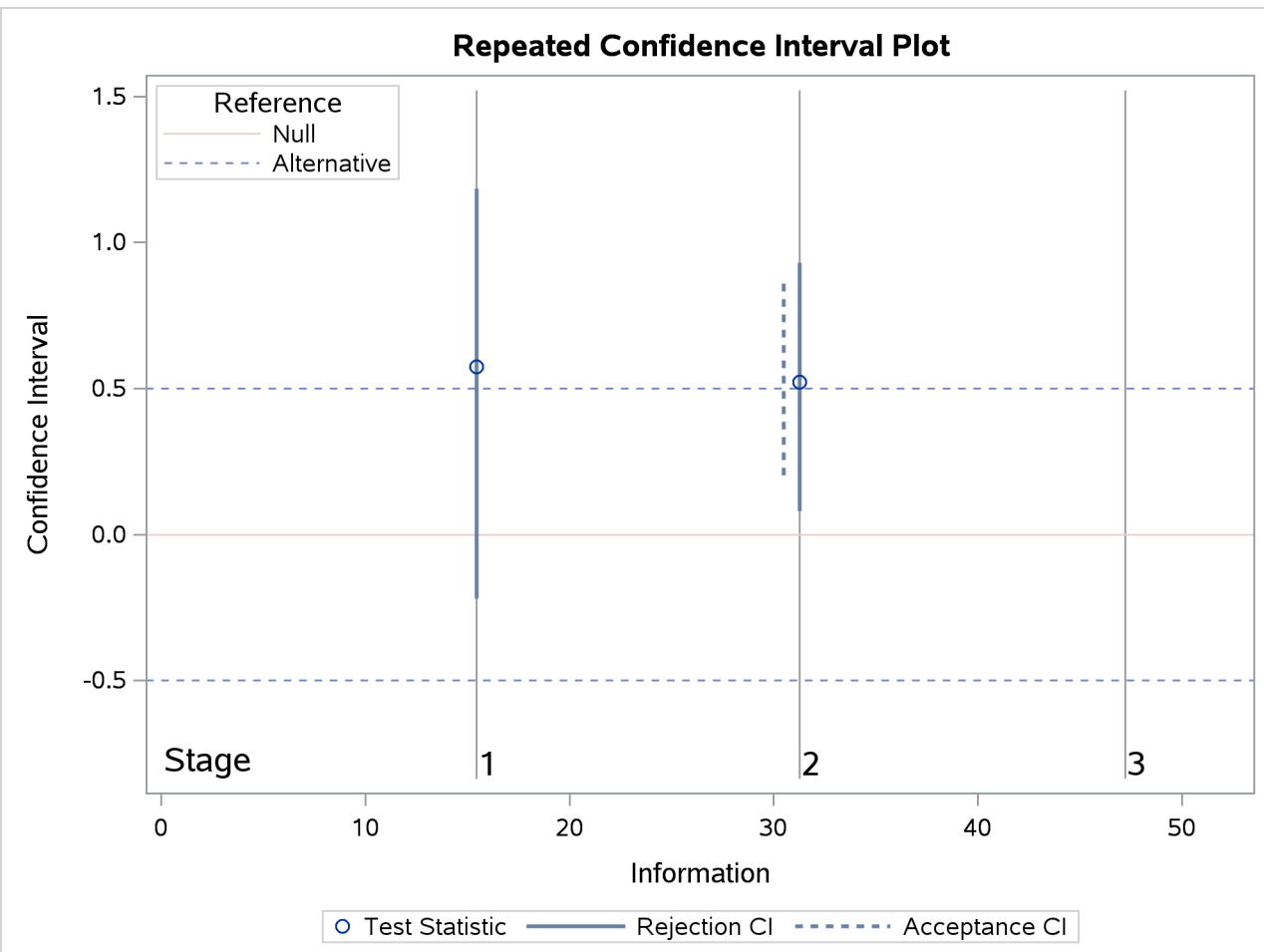

Note that the hypothesis is accepted if at any stage, the acceptance repeated confidence interval falls within the interval $(-0.5, 0.5)$ of the alternative references.

Example 111.9: Conducting a Trial with a Nonbinding Acceptance Boundary

(View the complete code for this example (seqtex9.sas) in the example repository.)

This example demonstrates a trail with a nonbinding acceptance boundary. The design is similar to the design in Example 111.1, but has a nonbinding beta boundary.

A clinic is studying the effect of vitamin C supplements in treating flu symptoms. The study consists of patients in the clinic who have exhibited the first sign of flu symptoms within the last 24 hours. These patients are randomly assigned to either the control group (which receives placebo pills) or the treatment group (which receives large doses of vitamin C supplements). At the end of a five-day period, the flu symptoms of each patient are recorded.

Suppose that you know from past experience that flu symptoms disappear in five days for 60% of patients who experience flu symptoms. The clinic would like to detect a 75% symptom disappearance with a high probability. A test that compares the proportions directly specifies the null hypothesis $H_0 : \theta = p_t - p_c = 0$ with a one-sided alternative $H_1 : \theta > 0$ and a power of 0.90 at $H_1 : \theta = 0.15$, where p_t and p_c are the proportions of symptom disappearance in the treatment group and control group, respectively.

The following statements invoke the SEQDESIGN procedure and request a four-stage group sequential design by using an O'Brien-Fleming method for normally distributed data.

```
ods graphics on;
proc seqdesign altref=0.15 errspend
               ;
   NonbindingDesign: design nstages=4
                     method=obf
                     alt=upper
                     stop=both(betaboundary=nonbinding)
                     alpha=0.025
                     ;
   samplesize model=twosamplefreq(nullprop=0.6 test=prop);
ods output Boundary=Bnd_Count;
run;
```

The ALT=UPPER option specifies a one-sided alternative hypothesis, and the STOP=BOTH option specifies early stopping to reject or accept the null hypothesis H_0, The BETABOUNDARY=NONBINDING suboption requests nonbinding beta boundary to have the flexibility to continue the trial when a test statistic falls in the acceptance region at interim stages (Zhu, Ni, and Yao 2011, pp. 132–133). The ODS OUTPUT statement with the BOUNDARY=BND_COUNT option creates an output data set named BND_COUNT which contains the resulting boundary information for the subsequent sequential tests.

The "Design Information" table in Output 111.9.1 displays design specifications. With the specified alternative hypothesis $H_1 : \theta = 0.15$, the maximum information is derived to achieve a power of 0.90 at H_1. The derived fixed-sample information ratio 1.107 is the maximum information needed for a group sequential design relative to its corresponding fixed-sample design. Compare to the default BETABOUNDARY=BINDING option, the BETABOUNDARY=NONBINDING option requires a larger sample size for the design.

Output 111.9.1 O'Brien-Fleming Design Information

The SEQDESIGN Procedure
Design: NonbindingDesign

Design Information	
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Accept(Nonbinding)/Reject Null
Method	O'Brien-Fleming
Boundary Key	Both
Alternative Reference	0.15
Number of Stages	4
Alpha (Binding Beta Boundary)	0.02228
Alpha (Nonbinding Beta Boundary)	0.025
Beta	0.1
Power	0.9
Max Information (Percent of Fixed Sample)	110.7138
Max Information	517.0296
Null Ref ASN (Percent of Fixed Sample)	62.29796
Alt Ref ASN (Percent of Fixed Sample)	78.5392

The "Boundary Information" table in Output 111.9.2 displays the information level, alternative reference, and boundary values at each stage. With the default BOUNDARYSCALE=STDZ option, the SEQDESIGN procedure displays the output boundaries with the standardized normal Z scale.

Output 111.9.2 O'Brien-Fleming Boundary Information

Boundary Information (Standardized Z Scale)
Nonbinding Beta Boundary, Null Reference = 0

	Information Level			Alternative Reference	Boundary Values	
					Upper	
Stage	Proportion	Actual	N	Upper	Beta	Alpha
1	0.2500	129.2574	110.5151	1.70537	-1.06752	4.04859
2	0.5000	258.5148	221.0302	2.41176	0.45103	2.86279
3	0.7500	387.7722	331.5452	2.95379	1.35286	2.33746
4	1.0000	517.0296	442.0603	3.41074	2.02430	2.02430

With ODS Graphics enabled, a detailed boundary plot with the rejection and acceptance regions is displayed, as shown in Output 111.9.3. The horizontal axis indicates the information levels for the design. The stages are indicated by vertical lines with accompanying stage numbers. If the test statistic at a stage is in a rejection region, the trial stops and the hypothesis is rejected. If the test statistic is in an acceptance region, then the trial either stops to accept the hypothesis or continues to the next stage. If the statistic is not in a rejection or an acceptance region, the trial continues to the next stage.

Output 111.9.3 O'Brien-Fleming Boundary Plot

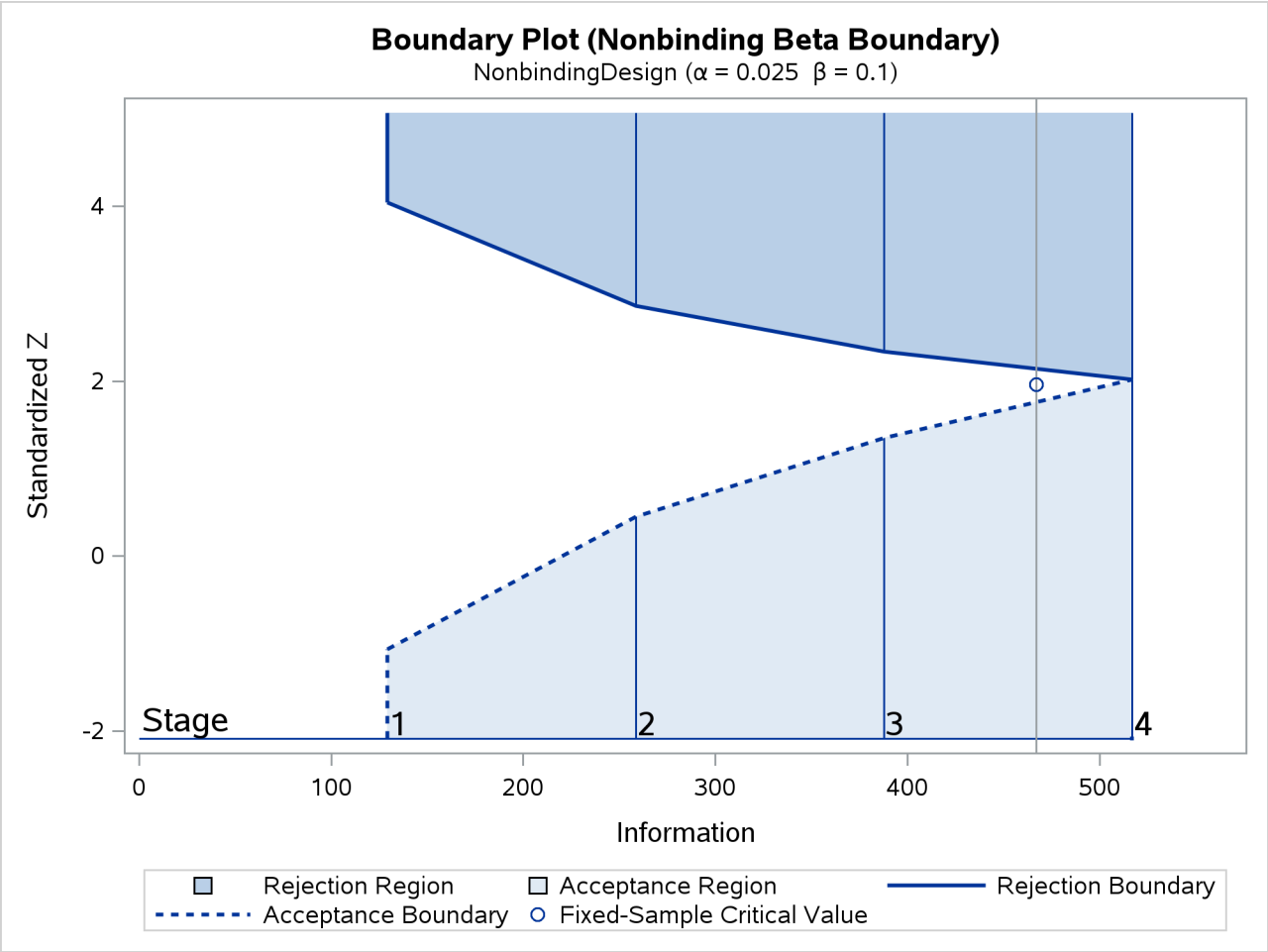

The boundary plot also displays the information level and critical value for the corresponding fixed-sample design. The solid and dashed lines at the fixed-sample information level correspond to the rejection and acceptance lines, respectively.

The "Error Spending Information (Nonbinding Beta Boundary)" in Output 111.9.4 displays cumulative error spending at each stage for each boundary. With a nonbinding beta boundary, the α spending at each stage is computed by using the α boundary only.

Output 111.9.4 Error Spending Information

Error Spending Information (Nonbinding Beta Boundary)

	Information Level	Cumulative Error Spending	
		Upper	
Stage	Proportion	Beta	Alpha
1	0.2500	0.00278	0.00003
2	0.5000	0.02603	0.00211
3	0.7500	0.06343	0.01046
4	1.0000	0.10000	0.02500

With the SAMPLESIZE statement, the maximum information is used to derive the required sample size for the study. The "Sample Size Summary" table in Output 111.9.5 displays parameters for the sample size computation.

Output 111.9.5 Required Sample Size Summary

Sample Size Summary	
Test	Two-Sample Proportions
Null Proportion	0.6
Proportion (Group A)	0.75
Test Statistic	Z for Proportion
Reference Proportions	Alt Ref
Max Sample Size	442.0603
Expected Sample Size (Null Ref)	248.7446
Expected Sample Size (Alt Ref)	313.5929

With the derived maximum information and the specified MODEL= option in the SAMPLESIZE statement, the total sample size in each group for testing the difference between two proportions under the alternative hypothesis is

$$N_1 = N_2 = (p_{1c}\,(1 - p_{1c}) + p_{1t}\,(1 - p_{1t}))\ I_X$$

where $p_{1c} = 0.6$ and $p_{1t} = p_{1c} + \theta_1 = 0.75$. By default (or equivalently if you specify REF=PROP in the MODEL=TWOSAMPLEFREQ option), the required sample sizes are computed under the alternative hypothesis. See the section "Test for the Difference between Two Binomial Proportions" on page 9197 in Chapter 110, "The SEQDESIGN Procedure," for more information about these parameters.

The "Sample Sizes (N)" table in Output 111.9.6 displays the required sample sizes at each stage, in both fractional and integer numbers. The derived sample sizes under the heading Fractional N which correspond to the design are not integers. These sample sizes are rounded up to integers under the heading Ceiling N. In practice, integer sample sizes are used, and the information levels increase slightly. Thus, 56, 111, 166, and 222 patients are needed in each group for the four stages, respectively.

Output 111.9.6 Required Sample Sizes

Sample Sizes (N) Two-Sample Z Test for Proportion Difference								
	Fractional N				Ceiling N			
Stage	N	N(Grp 1)	N(Grp 2)	Information	N	N(Grp 1)	N(Grp 2)	Information
1	110.52	55.26	55.26	129.3	112	56	56	131.0
2	221.03	110.52	110.52	258.5	222	111	111	259.6
3	331.55	165.77	165.77	387.8	332	166	166	388.3
4	442.06	221.03	221.03	517.0	444	222	222	519.3

Suppose the trial follows the study plan, and 56 patients are available in each group at stage 1. The data set count_1 contains these 108 patients. Output 111.9.7 lists the first 10 observations of the data set.

Output 111.9.7 Clinical Trial Data

First 10 Obs in the Trial Data

Obs	Trt	Resp
1	0	0
2	1	1
3	0	1
4	1	1
5	0	1
6	1	0
7	0	0
8	1	1
9	0	1
10	1	0

The Trt variable is a grouping variable with value 0 for a patient in the placebo control group and value 1 for a patient in the treatment group who is given vitamin C supplements. The Resp variable is an indicator variable with value 1 for a patient without flu symptoms after five days and value 0 for a patient with flu symptoms after five days.

The following statements use the GENMOD procedure to estimate the treatment effect at stage 1:

```
proc genmod data=count_1;
   model Resp= Trt;
ods output ParameterEstimates=Parms_Count1;
run;
```

Output 111.9.8 displays the treatment effect at stage 1.

Output 111.9.8 Stage 1 Treatment Difference

The GENMOD Procedure

Analysis Of Maximum Likelihood Parameter Estimates

Parameter	DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept	1	0.6786	0.0619	0.5572	0.7999	120.07	<.0001
Trt	1	0.0179	0.0876	-0.1538	0.1895	0.04	0.8384
Scale	1	0.4634	0.0310	0.4065	0.5283		

Note: The scale parameter was estimated by maximum likelihood.

The test statistic is $\hat{\theta}_1 = \hat{p}_t - \hat{p}_c = 0.0179$, and its associated standard error is

$$\sqrt{\mathrm{Var}(\hat{\theta}_1)} = \sqrt{\frac{\hat{p}_c(1-\hat{p}_c)}{56} + \frac{\hat{p}_t(1-\hat{p}_t)}{56}} = 0.0876$$

The following statements create and display (in Output 111.9.9) the data set that contains the parameter estimate at stage 1, $\hat{\theta}_1 = 0.0179$, and its associated standard error $\sqrt{\mathrm{Var}(\hat{\theta}_1)} = 0.0876$ which are used in the SEQTEST procedure:

```
data Parms_Count1;
   set Parms_Count1;
   if Parameter='Trt';
   _Scale_='MLE';
   _Stage_= 1;
   keep _Scale_ _Stage_ Parameter Estimate StdErr;
run;

proc print data=Parms_Count1;
   title 'Statistics Computed at Stage 1';
run;
```

Output 111.9.9 Statistics Computed at Stage 1

Statistics Computed at Stage 1

Obs	Parameter	Estimate	StdErr	_Scale_	_Stage_
1	Trt	0.0179	0.0876	MLE	1

The initial required sample sizes are derived with the proportions $p_c = 0.6$ and $p_t = 0.75$. If the observed proportions are different from these assumed values, or if the number of available patients is different from the study plan in one of the stages, then the information level that corresponds to the test statistic is estimated from

$$I_k = \frac{1}{\mathrm{Var}(\hat{\theta}_k)}$$

The following statements invoke the SEQTEST procedure and test for early stopping at stage 1:

```
ods graphics on;
proc seqtest Boundary=Bnd_Count
             Parms(Testvar=Trt)=Parms_Count1
             betaboundary=nonbinding
             infoadj=none
             errspendmin=0.001
             errspend
             ;
ods output Test=Test_Count1;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 1, which was generated in the SEQDESIGN procedure. The PARMS=PARMS_COUNT1 option specifies the input data set PARMS_COUNT1 that contains the test statistic and its associated standard error at stage 1, and the TESTVAR=TRT option identifies the test variable TRT in the data set.

The BETABOUNDARY=NONBINDING option requests nonbinding beta boundary to have the flexibility to continue the trial when the test statistic falls in the acceptance region at stage 1. The INFOADJ=NONE option maintains the information levels at future interim stages (2 and 3) as provided in the BOUNDARY= data set. The default BOUNDARYSCALE=STDZ option displays the output boundaries in terms of the standardized normal Z scale.

The O'Brien-Fleming design is conservative in early stages and might not be desirable in a clinical trial. The ERRSPENDMIN=0.001 option specifies the minimum error spending at each stage to be 0.001, and

it might increase the corresponding nominal p-value in early stages for the trial. The default BOUNDARYSCALE=STDZ option uses the standardized normal Z scale to display test statistics in the boundary table and boundary plots.

The ODS OUTPUT statement with the TEST=TEST_COUNT1 option creates an output data set named TEST_COUNT1 which contains the updated boundary information for the test at stage 1. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Design Information" table in Output 111.9.10 displays design specifications. The derived statistics, such as the overall α and β levels, are derived from the specified maximum information and boundary values in the BOUNDARY= data set. Note that with a minor change in the information level at stage 1, the power also changes slightly from the design provided in the BOUNDARY= data set.

Output 111.9.10 Design Information

The SEQTEST Procedure

Design Information	
BOUNDARY Data Set	WORK.BND_COUNT
Data Set	WORK.PARMS_COUNT1
Statistic Distribution	Normal
Boundary Scale	Standardized Z
Alternative Hypothesis	Upper
Early Stop	Accept(Nonbinding)/Reject Null
Number of Stages	4
Alpha (Binding Beta Boundary)	0.02237
Alpha (Nonbinding Beta Boundary)	0.025
Beta	0.10142
Power	0.89858
Max Information (Percent of Fixed Sample)	111.2654
Max Information	517.029617
Null Ref ASN (Percent of Fixed Sample)	62.27694
Alt Ref ASN (Percent of Fixed Sample)	76.31328

With the ERRSPEND option, the "Error Spending Information" table in Output 111.9.11 displays cumulative error spending at each stage for each boundary. By default (or equivalently if you specify BOUNDARYKEY=ALPHA), the Type I error level with nonbinding beta boundary $\alpha = 0.025$ is maintained. Furthermore, with the ERRSPENDMIN=0.001 option, the α spending at each stage is greater than or equal to 0.001.

Output 111.9.11 Error Spending Information

Error Spending Information (Nonbinding Beta Boundary)

	Information Level		Cumulative Error Spending	
			Upper	
Stage	Proportion	Actual	Beta	Alpha
1	0.2522	130.3756	0.00302	0.00100
2	0.5000	258.5148	0.02640	0.00299
3	0.7500	387.7722	0.06433	0.01101
4	1.0000	517.0296	0.10142	0.02500

The "Test Information" table in Output 111.9.12 displays the boundary values for the design, test statistic, and resulting action at each stage. With the default BOUNDARYSCALE=STDZ option, the standardized normal Z scale is used for the test statistic and boundary values. The table shows that the test statistic 0.2039 is between the upper α and β boundaries, so the trial continues to the next stage.

Output 111.9.12 Sequential Test

Test Information (Standardized Z Scale)
Nonbinding Beta Boundary, Null Reference = 0

	Information Level		Alternative Reference	Boundary Values		Test	
				Upper		Trt	
Stage	Proportion	Actual	Upper	Beta	Alpha	Estimate	Action
1	0.2522	130.3756	1.71273	-1.03271	3.09023	0.20390	Continue
2	0.5000	258.5148	2.41176	0.45564	2.84016	.	
3	0.7500	387.7722	2.95379	1.36069	2.34211	.	
4	1.0000	517.0296	3.41074	2.03599	2.03599	.	

The information level at stage 1 is derived from the standard error,

$$I_1 = \frac{1}{\mathrm{Var}(\hat{\theta}_1)} = \frac{1}{(\mathrm{s.e.}(\hat{\theta}_1))^2} = 130.376$$

By default (or equivalently if you specify PLOTS=TEST), the "Test Plot" graph displays boundary values of the design and the test statistic at stage 1, as shown in Output 111.9.13. It also shows that the observed statistic is in the continuation region.

Output 111.9.13 Sequential Test Plot

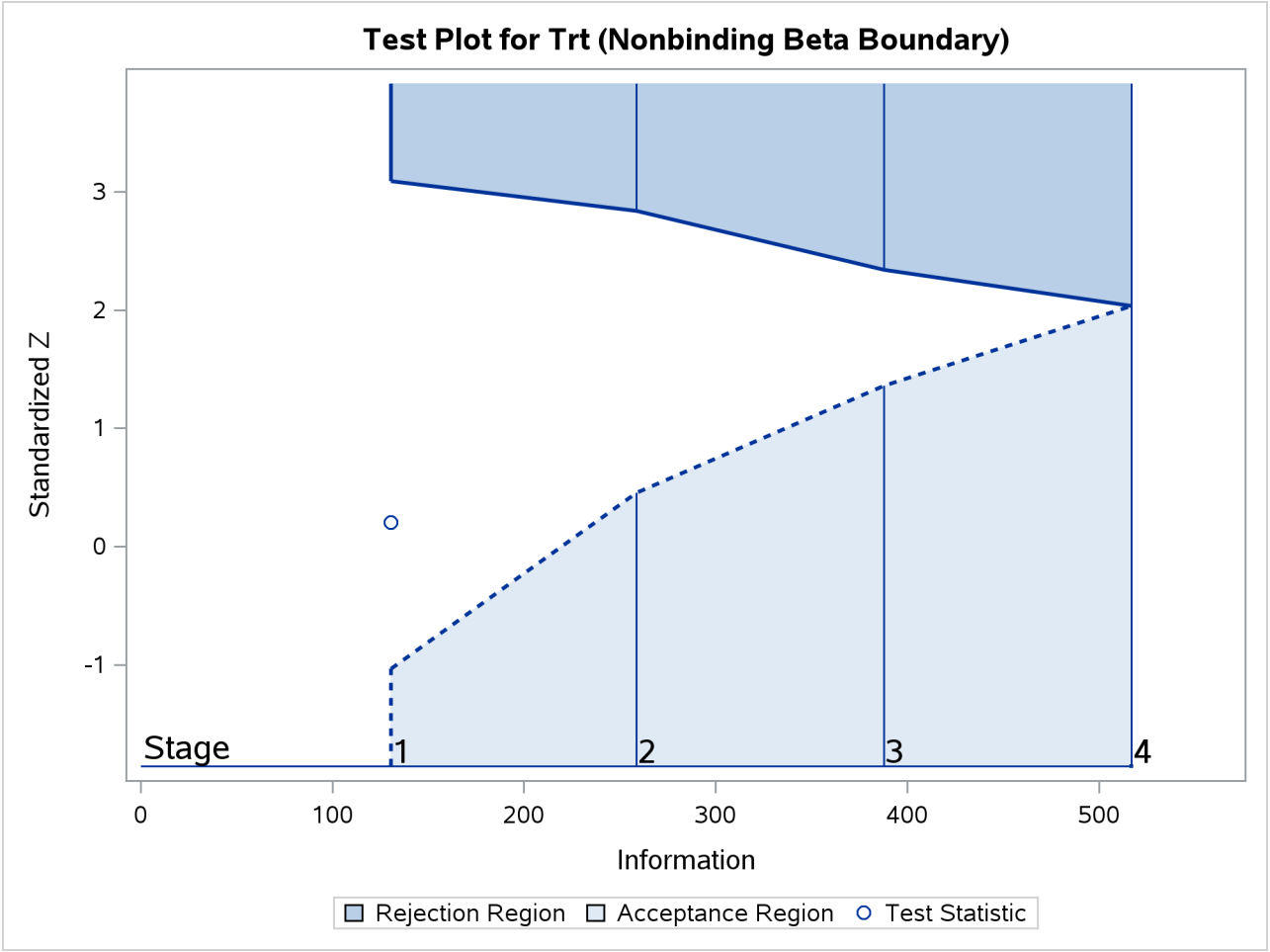

The observed information level at stage 1, $I_1 = 130.376$, is slightly larger than the target information level at the design. If an observed information level in the study is substantially different from its target level in the design, then the sample sizes should be adjusted in the subsequent stages to achieve the target information levels.

Suppose the trial continues to the next stage, and 111 patients are available in each group at stage 2. The data set COUNT_2 contains these 222 patients.

The following statements use the GENMOD procedure to estimate the treatment effect at stage 2:

```
proc genmod data=Count_2;
   model Resp= Trt;
ods output ParameterEstimates=Parms_Count2;
run;
```

The following statements create the parameter estimate at stage 2, $\hat{\theta}_2 = \hat{p}_2 - \hat{p}_1 = 0.027$, and its associated standard error $\sqrt{\mathrm{Var}(\hat{\theta}_2)} = 0.0621$ into a test data set:

```
data Parms_Count2;
   set Parms_Count2;
   if Parameter='Trt';
   _Scale_='MLE';
   _Stage_= 2;
   keep _Scale_ _Stage_ Parameter Estimate StdErr;
run;

proc print data=Parms_Count2;
   title 'Statistics Computed at Stage 2';
run;
```

Output 111.9.14 displays the test statistics at stage 2.

Output 111.9.14 Statistics Computed at Stage 2

Statistics Computed at Stage 2

Obs	Parameter	Estimate	StdErr	_Scale_	_Stage_
1	Trt	0.0270	0.0621	MLE	2

The following statements invoke the SEQTEST procedure and test for early stopping at stage 2:

```
ods graphics on;
proc seqtest Boundary=Test_Count1
             Parms(Testvar=Trt)=Parms_Count2
             betaboundary=nonbinding
             infoadj=none
             ;
ods output Test=Test_Count2;
run;
```

The BOUNDARY= option specifies the input data set that provides the boundary information for the trial at stage 2, which was generated by the SEQTEST procedure at the previous stage. The PARMS= option specifies the input data set that contains the test statistic and its associated standard error at stage 2, and the TESTVAR= option identifies the test variable in the data set.

The ODS OUTPUT statement with the TEST=TEST_COUNT2 option creates an output data set named TEST_COUNT2 which contains the updated boundary information for the test at stage 2. The data set also provides the boundary information that is needed for the group sequential test at the next stage.

The "Test Information" table in Output 111.9.15 displays the boundary values, test statistic, and resulting action at each stage. The table shows that the test statistic 0.43522 is less than the corresponding upper beta boundary value, so the trial can be stopped to accept the hypothesis. With the BETABOUNDARY=NONBINDING option, The clinic has the option to continue the trial while still maintaining the Type I error level.

Output 111.9.15 Sequential Test

The SEQTEST Procedure

Test Information (Standardized Z Scale) Nonbinding Beta Boundary, Null Reference = 0							
	Information Level		Alternative Reference	Boundary Values Upper			Test Trt
Stage	Proportion	Actual	Upper	Beta	Alpha	Estimate	Action
1	0.2522	130.3756	1.71273	-1.03271	3.09023	0.20390	Continue
2	0.5015	259.3157	2.41549	0.46349	2.83297	0.43522	Accept Null (Nonbinding)
3	0.7500	387.7722	2.95379	1.36018	2.34294	.	
4	1.0000	517.0296	3.41074	2.03610	2.03610	.	

With ODS Graphics enabled, the "Test Plot" is displayed, as shown in Output 111.9.16. The plot displays boundary values of the design and the test statistics at the first two stages. As expected, the test statistic at stage 2 is in the "Upper Acceptance Region" below the upper beta boundary.

Output 111.9.16 Sequential Test Plot

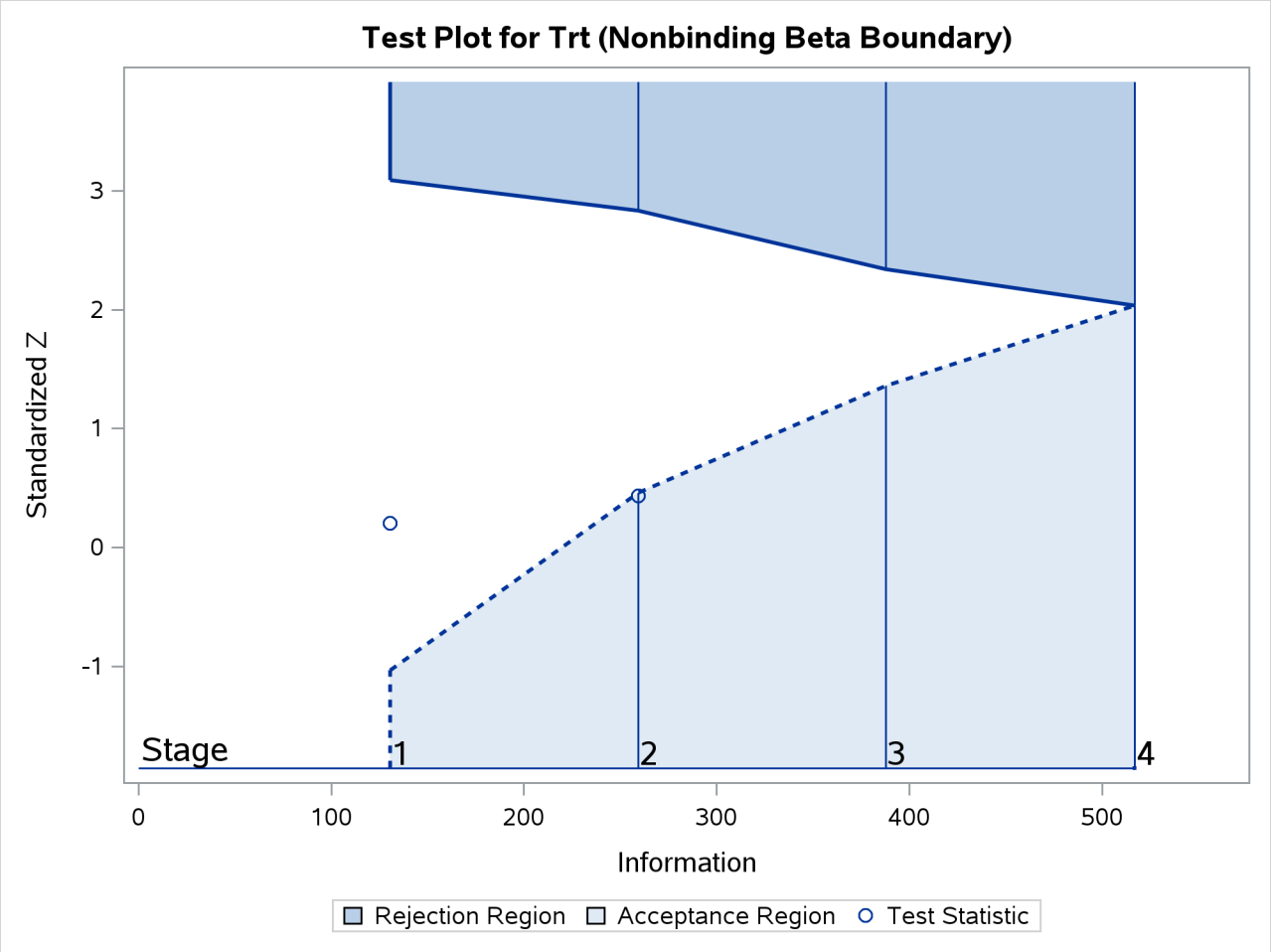

Suppose the clinic decides to continue the trial, and at the next stage, 108 patients are available in each group

at stage 3. The data set COUNT_3 contains these 332 patients.

The following statements use the GENMOD procedure to estimate the treatment effect at stage 3:

```
proc genmod data=Count_3;
   model Resp= Trt;
ods output ParameterEstimates=Parms_Count3;
run;
```

The following statements create the parameter estimate at stage 2, $\hat{\theta}_2 = \hat{p}_2 - \hat{p}_1 = 0.1205$, and its associated standard error $\sqrt{\mathrm{Var}(\hat{\theta}_2)} = 0.0507$ into a test data set:

```
data Parms_Count3;
   set Parms_Count3;
   if Parameter='Trt';
   _Scale_='MLE';
   _Stage_= 3;
   keep _Scale_ _Stage_ Parameter Estimate StdErr;
run;

proc print data=Parms_Count3;
   title 'Statistics Computed at Stage 3';
run;
```

Output 111.9.17 displays the test statistics at stage 3.

Output 111.9.17 Statistics Computed at Stage 3

Statistics Computed at Stage 3

Obs	Parameter	Estimate	StdErr	_Scale_	_Stage_
1	Trt	0.1205	0.0507	MLE	3

The following statements invoke the SEQTEST procedure and test for early stopping at stage 3:

```
ods graphics on;
proc seqtest Boundary=Test_Count2
             Parms(Testvar=Trt)=Parms_Count3
             betaboundary=nonbinding
             ;
ods output Test=Test_Count3;
run;
```

The “Test Information” table in Output 111.9.18 displays the boundary values, test statistic, and resulting action at each stage. The table shows that the test statistic 2.37437 is greater than the corresponding upper alpha boundary value, so the trial stops to reject the hypothesis.

Output 111.9.18 Sequential Test

The SEQTEST Procedure

Test Information (Standardized Z Scale) Nonbinding Beta Boundary, Null Reference = 0							
	Information Level		Alternative Reference	Boundary Values Upper			Test Trt
Stage	Proportion	Actual	Upper	Beta	Alpha	Estimate	Action
1	0.2522	130.3756	1.71273	-1.03271	3.09023	0.20390	Continue
2	0.5015	259.3157	2.41549	0.46349	2.83297	0.43522	Accept Null (Nonbinding)
3	0.7512	388.3763	2.95609	1.36434	2.34036	2.37437	Reject Null
4	1.0000	517.0296	3.41074	2.03658	2.03658	.	

With ODS Graphics enabled, the "Test Plot" is displayed, as shown in Output 111.9.19. The plot displays boundary values of the design and the test statistics at the first two stages. As expected, the test statistic at stage 3 is in the "Upper Rejection Region" above the upper alpha boundary.

Output 111.9.19 Sequential Test Plot

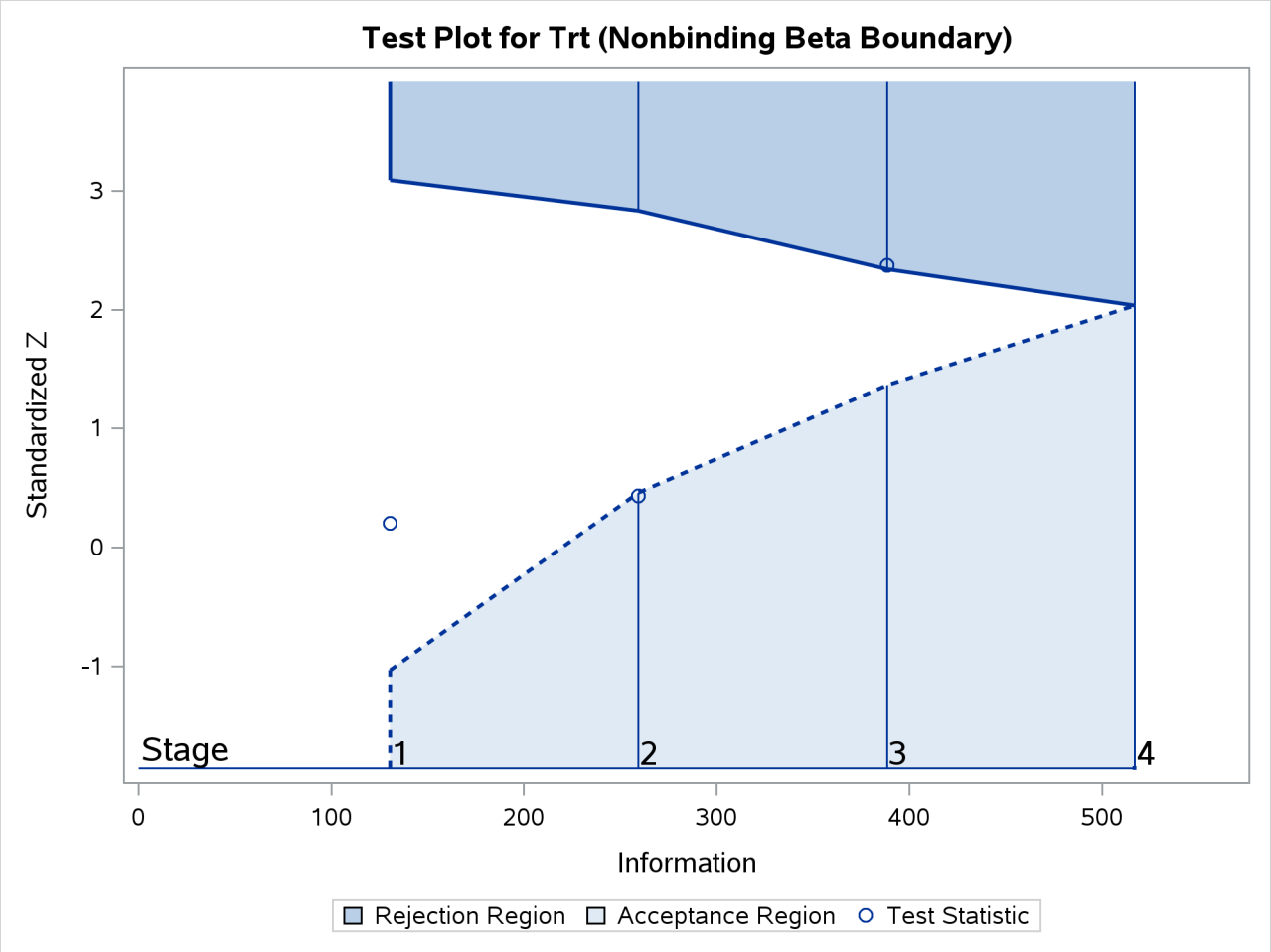

After a trial is stopped, the "Parameter Estimates" table in Output 111.9.20 displays the stopping stage and the

maximum likelihood estimate of the parameter. It also displays the p-value, median estimate, and confidence limits for the parameter that correspond to the observed statistic by using the specified sample space ordering.

Output 111.9.20 Parameter Estimates

Parameter Estimates Stagewise Ordering					
Parameter	Stopping Stage	MLE	p-Value for H0:Parm=0	Median Estimate	Lower 95% CL
Trt	3	0.120482	0.0103	0.119371	0.03494

The MLE statistic at the stopping stage is the maximum likelihood estimate of the parameter and is biased. The computation of p-value, unbiased median estimate, and confidence limits depends on the ordering of the sample space (k, z), where k is the stage number and z is the observed standardized Z statistic. By default (or equivalently if you specify ORDER=STAGEWISE), the stagewise ordering that uses counterclockwise ordering around the continuation region is used to compute the p-value, unbiased median estimate, and confidence limits. As expected, the p-value is less than 0.025, and the confidence interval does not contain the null reference zero.

See the section "Available Sample Space Orderings in a Sequential Test" on page 9369 for a detailed description of the stagewise ordering.

References

Chang, M. N. (1989). "Confidence Intervals for a Normal Mean Following a Group Sequential Test." *Biometrics* 45:247–254.

Chang, M. N., Gould, A. L., and Snapinn, S. M. (1995). "p-Values for Group Sequential Testing." *Biometrika* 82:650–654.

Chow, S.-C., and Liu, J.-P. (1998). *Design and Analysis of Clinical Trials: Concept and Methodologies*. New York: John Wiley & Sons.

Chow, S.-C., Shao, J., and Wang, H. (2003). *Sample Size Calculations in Clinical Research*. Boca Raton, FL: CRC Press.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220. With discussion.

DeMets, D. L., Furberg, C. D., and Friedman, L. M. (2006). *Data Monitoring in Clinical Trials*. New York: Springer.

Dmitrienko, A., Molenberghs, G., Chuang-Stein, C., and Offen, W. (2005). *Analysis of Clinical Trials Using SAS: A Practical Guide*. Cary, NC: SAS Institute Inc.

Ellenberg, S. S., Fleming, T. R., and DeMets, D. L. (2003). *Data Monitoring Committees in Clinical Trials*. New York: John Wiley & Sons.

Emerson, S. S. (1996). "Statistical Packages for Group Sequential Methods." *American Statistician* 50:183–192.

Emerson, S. S., and Fleming, T. R. (1989). "Symmetric Group Sequential Designs." *Biometrics* 45:905–923.

Emerson, S. S., and Fleming, T. R. (1990). "Parameter Estimation Following Group Sequential Hypothesis Testing." *Biometrika* 77:875–892.

Fairbanks, K., and Madsen, R. (1982). "*P*-Values for Tests Using a Repeated Significance Test Design." *Biometrika* 69:69–74.

Food and Drug Administration (1998). "E9: Statistical Principles for Clinical Trials." *Federal Register* 63:49583–49598.

Herson, J. (1979). "Predictive Probability Early Termination for Phase II Clinical Trials." *Biometrics* 35:775–783.

Hwang, I. K., Shih, W. J., and DeCani, J. S. (1990). "Group Sequential Designs Using a Family of Type I Error Probability Spending Functions." *Statistics in Medicine* 9:1439–1445.

Jennison, C., and Turnbull, B. W. (2000). *Group Sequential Methods with Applications to Clinical Trials*. New York: Chapman & Hall.

Kim, K., and Tsiatis, A. A. (1990). "Study Duration for Clinical Trials with Survival Response and Early Stopping Rule." *Biometrics* 46:81–92.

Kittelson, J. M., and Emerson, S. S. (1999). "A Unifying Family of Group Sequential Test Designs." *Biometrics* 55:874–882.

Lan, K. K. G., and DeMets, D. L. (1983). "Discrete Sequential Boundaries for Clinical Trials." *Biometrika* 70:659–663.

Lan, K. K. G., Lachin, J. M., and Bautista, O. (2003). "Over-ruling a Group Sequential Boundary: A Stopping Rule versus a Guideline." *Statistics in Medicine* 22:3347–3355.

Lan, K. K. G., Simon, R., and Halperin, M. (1982). "Stochastically Curtailed Tests in Long-Term Clinical Trials." *Sequential Analysis* 1:207–219.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

Mehta, C. R., and Tsiatis, A. A. (2001). "Flexible Sample Size Considerations under Information Based Interim Monitoring." *Drug Information Journal* 35:1095–1112.

O'Brien, P. C., and Fleming, T. R. (1979). "A Multiple Testing Procedure for Clinical Trials." *Biometrics* 35:549–556.

Proschan, M. A., Lan, K. K. G., and Wittes, J. T. (2006). *Statistical Monitoring of Clinical Trials*. New York: Springer.

Rosner, G. L., and Tsiatis, A. A. (1988). "Exact Confidence Intervals Following a Group Sequential Trial: A Comparison of Methods." *Biometrika* 75:723–729.

Rudser, K. D., and Emerson, S. S. (2007). "Implementing Type I and Type II Error Spending for Two-Sided Group Sequential Designs." *Contemporary Clinical Trials* 29:351–358.

Scharfstein, D. O., and Tsiatis, A. A. (1998). "The Use of Simulation and Bootstrap in Information-Based Group Sequential Studies." *Statistics in Medicine* 17:75–87.

Tsiatis, A. A., and Mehta, C. R. (2003). "On the Inefficiency of the Adaptive Design for Monitoring Clinical Trials." *Biometrika* 90:367–378.

Tsiatis, A. A., Rosner, G. L., and Mehta, C. R. (1984). "Exact Confidence Intervals Following a Group Sequential Test." *Biometrics* 40:797–803.

Ware, J. H., Muller, J. E., and Braunwald, E. (1985). "The Futility Index: An Approach to the Cost-Effective Termination of Randomized Clinical Trials." *American Journal of Medicine* 78:635–643.

Whitehead, J. (1997). *The Design and Analysis of Sequential Clinical Trials*. Rev. 2nd ed. Chichester, UK: John Wiley & Sons.

Zhu, L., Ni, L., and Yao, B. (2011). "Group Sequential Methods and Software Applications." *American Statistician* 65:127–135.

Chapter 112
The SIM2D Procedure

Contents

Overview: SIM2D Procedure . . . **9494**
- Introduction to Spatial Simulation . . . 9494

Getting Started: SIM2D Procedure . . . **9495**
- Preliminary Spatial Data Analysis . . . 9495
- Investigating Variability by Simulation . . . 9496

Syntax: SIM2D Procedure . . . **9502**
- PROC SIM2D Statement . . . 9504
- BY Statement . . . 9510
- COORDINATES Statement . . . 9511
- GRID Statement . . . 9511
- ID Statement . . . 9514
- RESTORE Statement . . . 9514
- SIMULATE Statement . . . 9516
- MEAN Statement . . . 9525

Details: SIM2D Procedure . . . **9526**
- Computational and Theoretical Details of Spatial Simulation . . . 9526
  - Introduction . . . 9526
  - Theoretical Development . . . 9526
  - Computational Details . . . 9529
- Output Data Set . . . 9530
- Displayed Output . . . 9531
- ODS Table Names . . . 9531
- ODS Graphics . . . 9532

Examples: SIM2D Procedure . . . **9533**
- Example 112.1: Simulation and Economic Feasibility . . . 9533
  - Simulating a Subregion for Economic Feasibility . . . 9533
  - Implementation Using PROC SIM2D . . . 9534
- Example 112.2: Variability at Selected Locations . . . 9538
- Example 112.3: Risk Analysis with Simulation . . . 9541

References . . . **9551**

Overview: SIM2D Procedure

The SIM2D procedure uses an LU decomposition technique to produce a spatial simulation for a Gaussian random field with a specified mean and covariance structure in two dimensions.

The simulation can be conditional or unconditional. If it is conditional, a set of coordinates and associated field values are read from a SAS data set. The resulting simulation honors these data values.

You can specify the mean structure as a quadratic function in the coordinates. Specify the semivariance by naming the form and supplying the associated parameters, or by using the contents of an item store file that was previously created by PROC VARIOGRAM.

PROC SIM2D can handle anisotropic and nested semivariogram models. Seven covariance models are supported: Gaussian, exponential, spherical, cubic, pentaspherical, sine hole effect, and Matérn. A single nugget effect is also supported.

You can specify the locations of simulation points in a GRID statement, or they can be read from a SAS data set. The grid specification is most suitable for a regular grid; the data set specification can handle any irregular pattern of points.

The SIM2D procedure writes the simulated values for each grid point to an output data set. The SIM2D procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For more information about the graphics available in PROC SIM2D, see the section "ODS Graphics" on page 9532.

Introduction to Spatial Simulation

The purpose of spatial simulation is to produce a set of partial realizations of a spatial random field (SRF) $Z(\boldsymbol{s}), \boldsymbol{s} \in D \subset \mathcal{R}^2$ in a way that preserves a specified mean $\mu(\boldsymbol{s}) = E\left[Z(\boldsymbol{s})\right]$ and covariance structure $C_z(\boldsymbol{s}_1 - \boldsymbol{s}_2) = \mathrm{Cov}\left(Z(\boldsymbol{s}_1), Z(\boldsymbol{s}_2)\right)$. The realizations are partial in the sense that they occur only at a finite set of locations $(\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n)$. These locations are typically on a regular grid, but they can be arbitrary locations in the plane.

PROC SIM2D produces simulations for continuous processes in two dimensions by using the lower-upper (LU) decomposition method. In these simulations the possible values of the measured quantity $Z(\boldsymbol{s}_0)$ at location $\boldsymbol{s}_0 = (x_0, y_0)$ can vary continuously over a certain range. An additional assumption, needed for computational purposes, is that the spatial random field $Z(\boldsymbol{s})$ is Gaussian. The section "Details: SIM2D Procedure" on page 9526 provides more information about different types of spatial simulation and associated computational methods.

Spatial simulation is different from spatial prediction, where the emphasis is on predicting a point value at a given grid location. In this sense, spatial prediction is local. In contrast, spatial simulation is global; the emphasis is on the entire realization $(Z(\boldsymbol{s}_1), Z(\boldsymbol{s}_2), \ldots, Z(\boldsymbol{s}_n))$.

Given the correct mean $\mu(\boldsymbol{s})$ and covariance structure $C_z(\boldsymbol{s}_1 - \boldsymbol{s}_2)$, SRF quantities that are difficult or impossible to calculate in a spatial prediction context can easily be approximated by functions of multiple simulations.

Getting Started: SIM2D Procedure

Spatial simulation, just like spatial prediction, requires a model of spatial dependence, usually in terms of the covariance $C_z(\boldsymbol{h})$. For a given set of spatial data $Z(\boldsymbol{s}_i), i = 1, \ldots, n$, the covariance structure (both the form and parameter values) can be found by the VARIOGRAM procedure. This example uses the coal seam thickness data that are also used in the section "Getting Started: VARIOGRAM Procedure" on page 10962 in Chapter 131, "The VARIOGRAM Procedure."

In this example, the data consist of coal seam thickness measurements (in feet) taken over an area of 100×100 (10^6 ft^2). The coordinates are offsets from a point in the southwest corner of the measurement area, with the north and east distances in units of thousands of feet.

Preliminary Spatial Data Analysis

A semivariance analysis of the coal seam thickness thick data set is performed in "Getting Started: VARIOGRAM Procedure" on page 10962 in Chapter 131, "The VARIOGRAM Procedure." The analysis considers the spatial random field (SRF) $Z(\boldsymbol{s})$ of the Thick variable to be free of surface trends. The expected value $E[Z(\boldsymbol{s})]$ is then a constant $\mu(\boldsymbol{s}) = \mu$, which suggests that you can work with the original thickness data rather than residuals from a trend surface fit. In fact, a reasonable approximation of the spatial process generating the coal seam data is given by

$$Z(\boldsymbol{s}) = \mu + \varepsilon(\boldsymbol{s})$$

where $\varepsilon(\boldsymbol{s})$ is a Gaussian SRF with Gaussian covariance structure

$$C_z(\boldsymbol{h}) = c_0 \exp\left(-\frac{h^2}{a_0^2}\right)$$

Of note, the term "Gaussian" is used in two ways in this description. For a set of locations $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n$, the random vector

$$\boldsymbol{Z}(\boldsymbol{s}) = \begin{bmatrix} Z(\boldsymbol{s}_1) \\ Z(\boldsymbol{s}_2) \\ \vdots \\ Z(\boldsymbol{s}_n) \end{bmatrix}$$

has a multivariate Gaussian or normal distribution $N_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. The (i,j) element of $\boldsymbol{\Sigma}$ is computed by $C_z(\boldsymbol{s}_i - \boldsymbol{s}_j)$, which happens to be a Gaussian functional form.

Any functional form for $C_z(\boldsymbol{h})$ that yields a valid covariance matrix $\boldsymbol{\Sigma}$ can be used. Both the functional form of $C_z(\boldsymbol{h})$ and the parameter values

$$\mu = 40.1173$$

$$c_0 = 7.4599$$

$$a_0 = 30.1111$$

are estimated by using PROC VARIOGRAM in section "Theoretical Semivariogram Model Fitting" on page 10971 in Chapter 131, "The VARIOGRAM Procedure." Specifically, the expected value μ is reported in the VARIOGRAM procedure OUTV output data set, and the parameters c_0 and a_0 are estimates derived from a weighted least squares fit.

The choice of a Gaussian functional form for $C_z(\boldsymbol{h})$ is simply based on the data, and it is not at all crucial to the simulation. However, it *is* crucial to the simulation method used in PROC SIM2D that $Z(\boldsymbol{s})$ be a Gaussian SRF. For details, see the section "Computational and Theoretical Details of Spatial Simulation" on page 9526.

Investigating Variability by Simulation

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/sim2dgs1.sas.)

The variability of $Z(\boldsymbol{s})$, as modeled by

$$Z(\boldsymbol{s}) = \mu + \varepsilon(\boldsymbol{s})$$

with the Gaussian covariance structure $C_z(\boldsymbol{h})$ found previously, is not obvious from the covariance model form and parameters. The variation around the mean of the surface is relatively small, making it difficult visually to pick up differences in surface plots of simulated realizations.

Instead, you can compute the mean for each location on a grid from a series of realizations in a simulation. Then, the standard deviation of all the simulated values at each grid location provides you with a measure of the variability of $Z(\boldsymbol{s})$ for the given covariance structure. You can also investigate variations at selected grid points in more detail, as shown in the "Example 112.2: Variability at Selected Locations" on page 9538.

The present example shows how to use ODS Graphics with PROC SIM2D to investigate the mean and standard deviation of simulated values. You use the thick data set which is available from the Sashelp library. In the data set, the Thick variable represents simulated observations of coal seam thickness. For your goal, you produce 5,000 realizations of a simulation with PROC SIM2D, where you specify the Gaussian model with the parameters found previously. You want the simulated data to pass through the simulated values, so first you define the data with the following data step:

```
title 'Using PROC SIM2D for Spatial Simulation';

data thick;
   input East North Thick @@;
   label Thick='Coal Seam Thickness';
   datalines;
    0.7  59.6  34.1   2.1  82.7  42.2   4.7  75.1  39.5
    4.8  52.8  34.3   5.9  67.1  37.0   6.0  35.7  35.9
    6.4  33.7  36.4   7.0  46.7  34.6   8.2  40.1  35.4
   13.3   0.6  44.7  13.3  68.2  37.8  13.4  31.3  37.8
   17.8   6.9  43.9  20.1  66.3  37.7  22.7  87.6  42.8
   23.0  93.9  43.6  24.3  73.0  39.3  24.8  15.1  42.3
   24.8  26.3  39.7  26.4  58.0  36.9  26.9  65.0  37.8
   27.7  83.3  41.8  27.9  90.8  43.3  29.1  47.9  36.7
```

```
    29.5  89.4  43.0  30.1   6.1  43.6  30.8  12.1  42.8
    32.7  40.2  37.5  34.8   8.1  43.3  35.3  32.0  38.8
    37.0  70.3  39.2  38.2  77.9  40.7  38.9  23.3  40.5
    39.4  82.5  41.4  43.0   4.7  43.3  43.7   7.6  43.1
    46.4  84.1  41.5  46.7  10.6  42.6  49.9  22.1  40.7
    51.0  88.8  42.0  52.8  68.9  39.3  52.9  32.7  39.2
    55.5  92.9  42.2  56.0   1.6  42.7  60.6  75.2  40.1
    62.1  26.6  40.1  63.0  12.7  41.8  69.0  75.6  40.1
    70.5  83.7  40.9  70.9  11.0  41.7  71.5  29.5  39.8
    78.1  45.5  38.7  78.2   9.1  41.7  78.4  20.0  40.8
    80.5  55.9  38.7  81.1  51.0  38.6  83.8   7.9  41.6
    84.5  11.0  41.5  85.2  67.3  39.4  85.5  73.0  39.8
    86.7  70.4  39.6  87.2  55.7  38.8  88.1   0.0  41.6
    88.4  12.1  41.3  88.4  99.6  41.2  88.8  82.9  40.5
    88.9   6.2  41.5  90.6   7.0  41.5  90.7  49.6  38.9
    91.5  55.4  39.0  92.9  46.8  39.1  93.4  70.9  39.7
    55.8  50.5  38.1  96.2  84.3  40.3  98.2  58.2  39.5
;
```

Since this is a conditional simulation, you can specify the OBSERV option in the PLOTS option in PROC SIM2D to see the locations and values of the measured points in the area where you want to perform spatial simulations.

Furthermore, the SIM suboption in the PLOTS option specifies that you want to create a plot that shows the means of the simulated values across the region. The SIM suboption with no other arguments produces a plot that shows the contours of the simulated means in the foreground and the gradient of the simulated standard deviations in the background.

You obtain these PROC SIM2D results at the nodes of an output grid that you specify according to your application needs. In the present analysis, a convenient area that encompasses all the Thick data points is a square with a side length of 100,000 feet. You define a regular grid for your simulation in this area. Assume a distance of 2,500 feet between grid nodes in both directions for a smooth contour plot. Based on this choice, your square grid has 41 nodes on each side. This means that PROC SIM2D computes the simulated values at a total of 1,681 grid points. You use the GRID statement of the PROC SIM2D to specify this grid.

The SIMULATE statement specifies the parameters of your simulation across the output grid. In particular, the VAR= option specifies the conditional simulation variable. The number of realizations in the simulation is specified with the NUMREAL= option. The SEED= option specifies the seed for the simulation random number generator.

The spatial correlation model for the simulation is also specified in the SIMULATE statement. You specify the model type by using the FORM= option. The options SCALE= and RANGE= specify the covariance structure sill c_0 and range a_0 parameters, respectively, as discussed in the previous section.

Although it is not included in the original spatial structure, note that a minimal nugget effect is specified with the NUGGET= option to avoid singularity issues. Singularity can appear in the present example as a result of the combined use of the Gaussian covariance model and relatively short distances between nodes, data, or nodes and data in the simulation area.

These steps are implemented using the following DATA step and statements:

```
ods graphics on;

proc sim2d data=thick outsim=sim plot=(observ sim);
   coordinates xc=East yc=North;
   simulate var=Thick numreal=5000 seed=79931
      scale=7.4599 range=30.1111 nugget=1e-8 form=gauss;
   mean 40.1173;
   grid x=0 to 100 by 2.5 y=0 to 100 by 2.5;
run;
```

The table in Figure 112.1 shows the number of observations read and used in the conditional simulation. This table can provide you with useful information in case you have missing values in the input data.

Figure 112.1 Number of Observations for the thick Data Set

Using PROC SIM2D for Spatial Simulation

The SIM2D Procedure

Simulation: SIM1, Dependent Variable: Thick

Number of Observations Read	75
Number of Observations Used	75

The sample locations are then plotted in Figure 112.2. The figure clearly shows some small-scale variation that is typical of spatial data.

Figure 112.2 Scatter Plot of the Observations Spatial Distribution

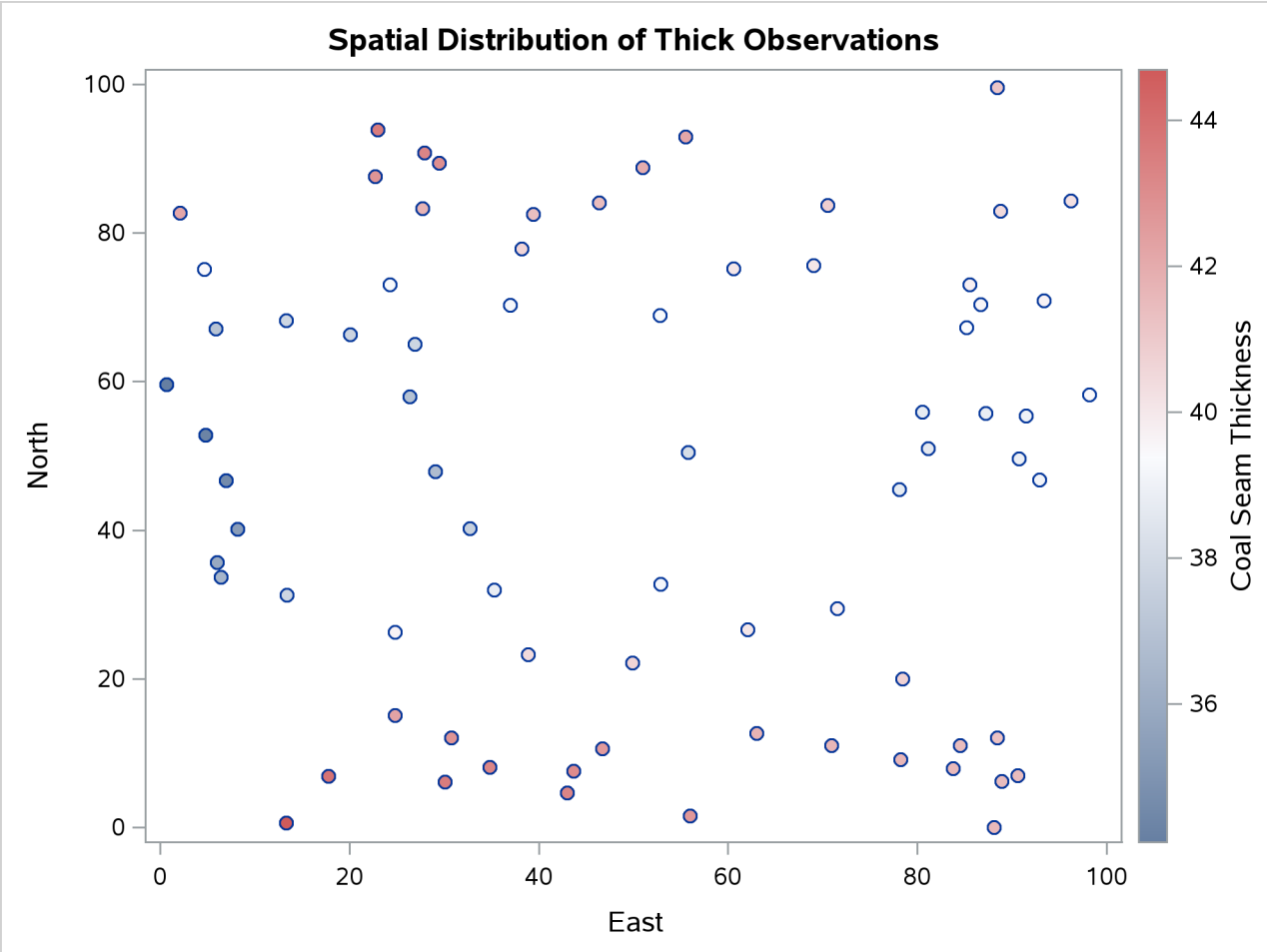

PROC SIM2D also produces the table shown in Figure 112.3, which contains information about the type of simulation you run and the number of realizations requested.

Figure 112.3 Simulation Analysis Information

Simulation Information	
Simulation Grid Points	1681
Type	Conditional
Number of Realizations	5000

The table in Figure 112.4 displays the spatial correlation model information that is used by PROC SIM2D for the current simulation. If applicable, the table also provides the effective range. This is the distance r_ϵ at which the covariance is 5% of its value at zero. Here you specified the Gaussian model, for which the effective range r_ϵ is $\sqrt{3}a_0$.

Figure 112.4 Simulation Covariance Model Information

Covariance Model Information	
Type	Gaussian
Sill	7.4599
Range	30.1111
Effective Range	52.153955
Nugget Effect	1E-8

Eventually, the SIM2D procedure produces the requested simulation plot shown in Figure 112.5. The contours of the mean of the simulated values show the average of the simulated realizations at each grid node; the average is based on the given spatial structure characteristics. In this case, these means are also conditioned by the Thick observations across the region.

Figure 112.5 Contour Plot of Conditionally Simulated Coal Seam Thickness

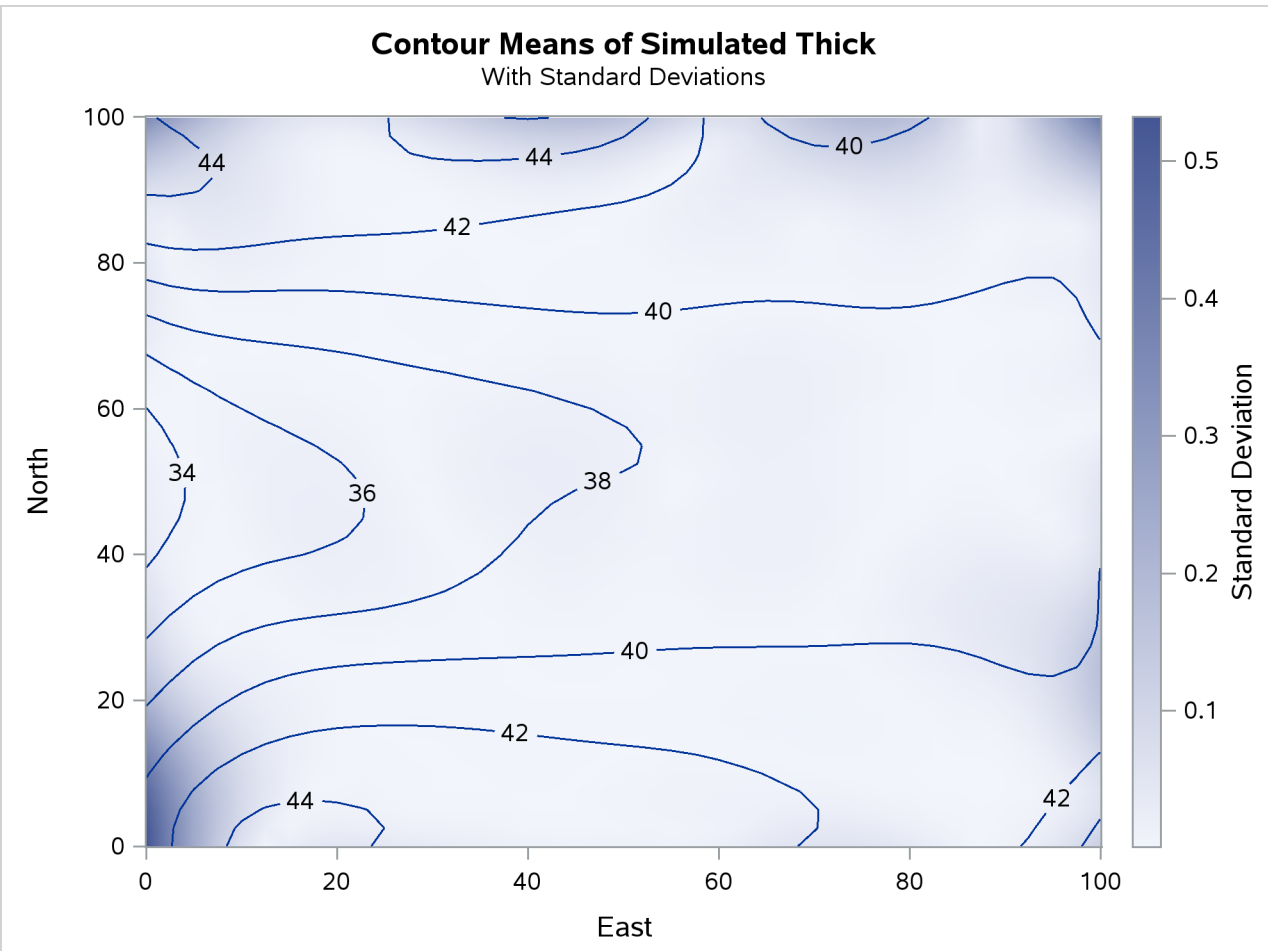

Observe also the gradient that shows the standard deviation of the simulated values at each grid node. This gradient appears to be generally small throughout the region. A few exceptions are evident close to the region borders. In these areas the simulated realizations depend on a limited amount of neighboring data. The simulation at these locations relies mainly on the underlying spatial structure.

In addition to the simulation analysis, you can use the PROC SIM2D output to obtain statistical information about the simulated values at selected locations. Assume that you would like some basic statistics for the extreme southwest point at (East=0, North=0) and the point (East=75, North=75) toward the northeast corner of the region. You use the following DATA step to select the realizations for these points from the OUTSIM= output data set:

```
data selected;
   set sim(where=((gxc=0 and gyc=0) or (gxc=75 and gyc=75)));
   label gxc = "X-coord";
   label gyc = "Y-coord";
run;
```

Then, you use PROC SORT to sort the selected data set entries and PROC MEANS to produce the simulation statistics for the selected points. The following statements yield the mean, standard deviation, and maximum values of the 5,000 realizations of the Thick values at each one of the selected locations:

```
proc sort data=selected;
   by gxc gyc;
run;
proc means data=selected Mean Std Max;
   class gxc gyc;
   ways 2;
   where (  ((gxc =  0) & (gyc =  0))
          | ((gxc = 75) & (gyc = 75)));
   var SValue;
run;

ods graphics off;
```

The requested statistics for the grid points (East=0, North=0) and (East=75,North=75) are shown in Figure 112.6.

Figure 112.6 Simulation Statistics at Grid Points (East=0, North=0) and (East=75, North=75)

Using PROC SIM2D for Spatial Simulation

The MEANS Procedure

Analysis Variable : SVALUE Simulated Value at Grid Point					
X-coord	Y-coord	N Obs	Mean	Std Dev	Maximum
0	0	5000	40.6968472	0.5328597	42.6616357
75	75	5000	40.1090845	0.0024556	40.1197239

"Example 112.2: Variability at Selected Locations" on page 9538 shows you how to perform a simulation at a set of selected locations rather than on a domain-wide grid, and how to obtain more detailed statistics from the simulation.

Syntax: SIM2D Procedure

The following statements are available in the SIM2D procedure:

PROC SIM2D *options* **;**
  BY *variables* **;**
  COORDINATES *coordinate-variables* **;**
  GRID *grid-options* **;**
  ID *variable* **;**
  RESTORE *store-options* **;**
  SIMULATE *simulate-options* **;**
  MEAN *mean-options* **;**

The SIMULATE and MEAN statements are hierarchical; you can specify any number of SIMULATE statements, but you must specify at least one. If you specify a MEAN statement, it refers to the preceding SIMULATE statement. If you omit the MEAN statement, a zero-mean model is simulated.

You must specify a single COORDINATES statement to identify the x and y coordinate variables in the input data set when you perform a conditional simulation. You must also specify a single GRID statement to specify the grid information.

Table 112.1 outlines the *options* available in PROC SIM2D classified by function.

Table 112.1 Options Available in the SIM2D Procedure

Task	Statement	Option
Data Set Options		
Specify an input data set	PROC SIM2D	DATA=
Specify a grid data set	GRID	GDATA=
Specify labels for individual grid points or in 1-D	GRID	LABEL
Specify correlation model and parameters	SIMULATE	MDATA=
Write simulated values	PROC SIM2D	OUTSIM=
Specify plot display and options	PROC SIM2D	PLOTS
Specify a quadratic form data set	MEAN	QDATA=
Specify plot display and options	PROC SIM2D	PLOTS
Declaring the Role of Variables		
Specify variables to define analysis subgroups	BY	
Specify a variable with observation labels	ID	
Specify the conditioning variable	SIMULATE	VAR=
Specify the x and y coordinate variables in the DATA= data set	COORDINATES	XC= YC=
Specify the x and y coordinate variables in the GDATA= data set	GRID	XC= YC=
Specify the constant coefficient variable in the QDATA= data set	MEAN	CONST=
Specify the linear x coefficient variable in the QDATA= data set	MEAN	CX=

Table 112.1 *continued*

Task	Statement	Option
Specify the linear y coefficient variable in the QDATA= data set	MEAN	CY=
Specify the quadratic x coefficient variable in the QDATA= data set	MEAN	CXX=
Specify the quadratic y coefficient variable in the QDATA= data set	MEAN	CYY=
Specify the quadratic xy coefficient variable in the QDATA= data set	MEAN	CXY=
Controlling the Simulation		
Specify the number of grid points in one-dimensional cases	GRID	NPTS=
Specify the number of realizations	SIMULATE	NUMREAL=
Specify the seed value for the random generator	SIMULATE	SEED=
Controlling the Mean Quadratic Surface		
Specify the CONST term	MEAN	CONST=
Specify the linear x term	MEAN	CX=
Specify the linear y term	MEAN	CY=
Specify the quadratic x term	MEAN	CXX=
Specify the quadratic y term	MEAN	CYY=
Specify the quadratic cross term	MEAN	CXY=
Controlling the Semivariogram Model		
Specify an angle for an anisotropic model	SIMULATE	ANGLE=
Specify nested angles	SIMULATE	ANGLE=$(a_1, \ldots, a_k)$
Specify a functional form	SIMULATE	FORM=
Specify nested functional forms	SIMULATE	FORM=$(f_1, \ldots, f_k)$
Specify a nugget effect	SIMULATE	NUGGET=
Specify a range parameter	SIMULATE	RANGE=
Specify nested range parameters	SIMULATE	RANGE=$(r_1, \ldots, r_k)$
Specify a minor-major axis ratio for an anisotropic model	SIMULATE	RATIO=
Specify nested minor-major axis ratios	SIMULATE	RATIO=$(ra_1, \ldots, ra_k)$
Specify a scale parameter	SIMULATE	SCALE=
Specify nested scale parameters	SIMULATE	SCALE=$(s_1, \ldots, s_k)$
Specify item store with correlation information	RESTORE	IN=
Specify model and parameters from an item store	SIMULATE	STORESELECT

PROC SIM2D Statement

PROC SIM2D *options* **;**

The PROC SIM2D statement invokes the SIM2D procedure. Table 112.2 summarizes the *options* available in the PROC SIM2D statement.

Table 112.2 PROC SIM2D Statement Options

Option	Description
DATA=	Specifies an input data set
IDGLOBAL	Uses ascending observation numbers across BY groups
IDNUM	Uses the observation number for the observation labels
NARROW	Restricts the variables included in the OUTSIM= data set
NOPRINT	Suppresses the normal display of results
OUTSIM=	Writes simulated values to a SAS data set
PLOTS	Specifies plot display and options

You can specify the following *options* with the PROC SIM2D statement.

DATA=*SAS-data-set*

specifies a SAS data set that contains the x and y coordinate variables and the VAR= variables that are used in the SIMULATE statements. This data set is required if you specify the BY statement or the COORDINATES statement or if any of the SIMULATE statements are conditional—that is, if you specify the VAR= option in any of those. Otherwise, you do not need the DATA= option, and this option is ignored if you specify it.

IDGLOBAL

specifies that ascending observation numbers be used across BY groups for the observation labels in the appropriate output data sets and the OBSERVATIONS plot, instead of resetting the observation number in the beginning of each BY group. The IDGLOBAL option is ignored if no BY variables are specified. Also, if you specify the ID statement, then the IDGLOBAL option is ignored unless you also specify the IDNUM option in the PROC SIM2D statement.

IDNUM

specifies that the observation number be used for the observation labels in the appropriate output data sets and the OBSERVATIONS plot. The IDNUM option takes effect when you specify the ID statement; otherwise, it is ignored.

NARROW

restricts the variables included in the OUTSIM= data set. When you specify the NARROW option, only four variables are included. This option is useful when a large number of simulations are produced. Including only four variables reduces the memory required for the OUTSIM= data set. For details about the variables that are excluded with the NARROW option, see the section "Output Data Set" on page 9530.

NOPRINT

suppresses the normal display of results. The NOPRINT option is useful when you want only to create one or more output data sets with the procedure. **NOTE:** This option temporarily disables the Output Delivery System (ODS); see the section "ODS Graphics" on page 9532 for more information.

OUTSIM=*SAS-data-set*

specifies a SAS data set in which to store the simulation values, iteration number, simulate statement label, variable name, and grid location. For details, see the section "Output Data Set" on page 9530.

PLOTS <(*global-plot-option***) > < = *plot-request*< (***options***) > >**

PLOTS <(*global-plot-option***) > < = (***plot-request*< (***options***) > < . . . *plot-request*< (***options***) > >) >**

controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=none
plots=observ
plots=(observ(outl) sim)
plots=(sim(fill=mean line=sd obs=grad) sim(fill=sd))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc sim2d data=thick outsim=sim;
   coordinates xc=East yc=North;
   simulate var=Thick numreal=5000 seed=79931
      scale=7.4599 range=30.1111 form=gauss;
   mean 40.1173;
   grid x=0 to 100 by 2.5 y=0 to 100 by 2.5;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

By default, no graphs are created; you must specify the PLOTS= option to make graphs.

The following *global-plot-option* is available:

ONLY

produces only plots that are specifically requested.

The following individual *plot-requests* and plot *options* are available:

ALL

produces all appropriate plots. You can specify other *options* with ALL. For example, to request all appropriate plots and an additional simulation plot, specify PLOTS=(ALL SIM).

EQUATE
specifies that all appropriate plots be produced in a way in which the axes coordinates have equal size units.

NONE
suppresses all plots.

OBSERVATIONS < (*observations-plot-options*) >
OBSERV < (*observations-plot-options*) >
OBS < (*observations-plot-options*) >
produces the observed data plot in conditional simulations. Only one observations plot is created if you specify the OBSERVATIONS option more than once within a PLOTS option.

The OBSERVATIONS option has the following suboptions:

GRADIENT
specifies that observations be displayed as circles colored by the observed measurement.

LABEL < (*label-option*) >
labels the observations. The label is the ID variable if the ID statement is specified; otherwise, it is the observation number. The *label-option* can be one of the following:

EQ=*number*
specifies that labels show for any observation whose value is equal to the specified *number*.

MAX=*number*
specifies that labels show for observations with values smaller than or equal to the specified *number*.

MIN=*number*
specifies that labels show for observations with values equal to or greater than the specified *number*.

If you specify multiple instances of the OBSERVATIONS option and you specify the LABEL suboption in any of those, then the resulting observations plot displays the observations labels. If more than one *label-option* is specified in multiple LABEL suboptions, then the prevailing *label-option* in the resulting OBSERVATIONS plot emerges by adhering to the choosing order: MIN, MAX, EQ.

OUTLINE
specifies that observations be displayed as circles with a border but with a completely transparent fill.

OUTLINEGRADIENT
is the same as OBSERVATIONS(GRADIENT) except that a border is shown around each observation.

SHOWMISSING

specifies that observations with missing values be displayed in addition to the observations with nonmissing values. By default, missing values locations are not shown on the plot. If you specify multiple instances of the OBSERVATIONS option and you specify the SHOWMISSING suboption in any of those, then the resulting observations plot displays the observations with missing values.

If you omit any of the GRADIENT, OUTLINE, and OUTLINEGRADIENT suboptions, the OUTLINEGRADIENT is the default suboption. If you specify multiple instances of the OBSERVATIONS option or multiple suboptions for OBSERVATIONS, then the resulting observations plot honors the last specified GRADIENT, OUTLINE, or OUTLINEGRADIENT suboption.

SIMULATION < (*sim-plot-options*) >

SIM < (*sim-plot-options*) >

specifies that simulation plots be produced. You can specify the SIM option multiple times in the same PLOTS option to request instances of plots with the following *sim-plot-options*:

ALPHA=*number*

specifies a parameter to obtain the confidence level for constructing confidence limits based on the simulation standard deviation. The value of *number* must be between 0 and 1, and the confidence level is $1-$*number*. The default is ALPHA=0.05; this corresponds to the confidence level of 95%. The ALPHA= suboption is used only for simulation plots in one dimension, and it is incompatible with the FILL and LINE suboptions.

CLONLY

specifies that only the confidence limits be shown in a simulation plot without the simulation mean. This suboption can be useful for identifying confidence limits when the simulation standard deviation is small at the simulation locations. CLONLY is used only for simulation band plots of simulations on a linear grid, and it is incompatible with the FILL and LINE suboptions.

CONNP

specifies that grid points that you provide as individual simulation locations be connected with a line on the area map. This suboption is ignored when you have a single grid point, a prediction grid in two dimensions, or when you also specify the NOMAP suboption. The CONNP suboption is incompatible with the FILL and LINE suboptions.

FILL=NONE | MEAN | SD

produces a surface plot for either the values of the means or the standard deviations. FILL=SD is the default. However, if you omit the FILL suboption the behavior depends on the LINE suboption as follows: If you specify LINE=NONE or entirely omit the LINE suboption, then the FILL suboption is set to its default value. If LINE=PRED or LINE=SE, then the FILL suboption is set to the same value as the LINE suboption.

LINE=NONE | MEAN | SD

produces a contour line plot for either the values of the means or the standard deviations. LINE=MEAN is the default. However, if you omit the LINE suboption the behavior depends on the FILL suboption as follows: If you specify FILL=NONE or entirely omit the FILL suboption, then the LINE suboption is set to its default value. If FILL=PRED or FILL=SE, then the LINE suboption is set to the same value as the FILL suboption.

NOMAP
specifies that the simulation plot be produced without a map of the domain where you have observations. The NOMAP suboption is used in the case of simulation in one dimension or at individual points. It is ignored in the case of unconditional simulation, and it is incompatible with the FILL and LINE suboptions.

OBS=*obs-options*
produces an overlaid scatter plot of the observations in addition to the specified contour plots. The following *obs-options* are available:

GRAD
specifies that observations be displayed as circles colored by the observed measurement. The same color gradient displays the means surface and the observations. The conditional simulation honors the observed values, so the means surface at the observation locations has the same color as the corresponding observations.

LINEGRAD
is the same as OBS=GRAD except that a border is shown around each observation. This option is useful for identifying the location of observations where the standard deviations are small, because at these points the color of the observations and the color of the surface are indistinguishable.

NONE
specifies that no observations be displayed.

OUTL
specifies that observations be displayed as circles with a border but with a completely transparent fill.

OBS=NONE is the default when you have a grid in two dimensions, and OBS=LINEGRAD is the default used in the area map when you specify a conditional simulation in one dimension.

SHOWD
specifies that the horizontal axis in scatter plots of linear simulation grids show the distance between grid points instead of the grid points' coordinates. When the area map is displayed, the simulation locations are also connected with a line. In all other grid configurations the SHOWD suboption is ignored, and it is incompatible with the FILL and LINE suboptions.

SHOWP
specifies that the grid points in band plots of linear simulation grids be shown as marks on the band plot. In all other grid configurations the SHOWP suboption is ignored, and it is incompatible with the FILL and LINE suboptions.

TYPE=BAND | BOX
requests a particular type of plot when you have a linear grid, regardless of the default SIM plot behavior in this case. The TYPE suboption is incompatible with the FILL and LINE suboptions.

If you specify multiple instances of the ALPHA, FILL, LINE, OBS, or TYPE suboptions in the same SIM option, then the resulting simulation plot honors the last value specified for any of the

suboptions. Any combination where you specify FILL=NONE and LINE=NONE is not available. When the simulation grid is in two dimensions, only the FILL, LINE, and OBS suboptions apply. If you specify incompatible suboptions in the same SIM plot, then the plot instance is skipped.

The SIM option produces a surface or contour line plot for grids in two dimensions and a band plot or box plot for grids in one dimension or individual points. In two dimensions the plot illustrates the means and standard deviations of the simulation realizations at each grid point. By default, when you specify a linear grid with fewer than 10 points, PROC SIM2D produces a SIM box plot that depicts the simulation distribution at each point. For 10 or more points in a linear grid, the SIM plot is a band plot of the simulation means and the confidence limits at the 95% confidence level. You can override the default behavior in linear grids with the TYPE suboption. Simulation at individual locations always produces a SIM box plot.

In cases of conditional simulation in one dimension or at individual points an area map is produced that shows the observations and the grid points. Band plots of linear grids display the grid points as a line on the map. When you specify individual simulation locations, the grid points are indicated with marks on the area map. The area map appears on the side of conditional simulation band plots or box plots, unless you specify the NOMAP suboption. You can also label individual grid points or the ends of linear grid segments with the LABEL option of the GRID statement.

SEMIVARIOGRAM < (*semivar-plot-option*) >

SEMIVAR < (*semivar-plot-option*) >

specifies that the semivariogram used for the simulation be produced. You can use the following *semivar-plot-option*:

MAXD=*number*

specifies a positive value for the upper limit of the semivariogram horizontal axis of distance. The SEMIVARIOGRAM plot extends by default to a distance that depends on the correlation model range. You can use the MAXD= option to adjust the default maximum distance value for the plot.

The SEMIVARIOGRAM option produces a plot for each correlation model that you specify for your simulation tasks. In an anisotropic case, the plot is not produced if you assign different anisotropy angles for different model components. The only exception is when you specify zonal components at right angles with the nonzonal model components. Also, the SEMIVARIOGRAM option is ignored for models that consist of purely zonal components.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SIM2D to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SIM2D procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

In PROC SIM2D it makes sense to use the BY statement only in conditional simulations where observations are involved. In particular, using the BY statement assumes that you have specified an input data set with the DATA= option in the PROC SIM2D statement. In PROC SIM2D if you omit the VAR= option in the SIMULATE statement, then this is a request for unconditional simulation even if you have specified the DATA= option in the PROC SIM2D statement. Therefore, it is possible to specify the BY statement and request mixed types of simulations by specifying multiple SIMULATE statements in the same PROC SIM2D step.

A special case occurs when you omit the DATA= option in the PROC SIM2D statement, your unconditional simulation correlation model input comes from an item store in the RESTORE statement, and this store has its own BY groups. The SIM2D procedure exhibits then a BY-like behavior, even though you specified no BY statement. This behavior enables you to distinguish the simulation tasks that depend on models in the different store BY groups.

For more information about BY-group processing, see the “Grouping Data” section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

COORDINATES Statement

COORDINATES *coordinate-variables* **;**

The two options in the COORDINATES statement give the name of the variables in the DATA= data set that contains the values of the *x* and *y* coordinates of the conditioning data. You must specify the COORDINATES statement when you specify an input data set with the DATA= option in the PROC SIM2D statement.

Only one COORDINATES statement is allowed, and it is applied to all SIMULATE statements that have the VAR= specification. In other words, it is assumed that all the VAR= variables in all SIMULATE statements have the same *x* and *y* coordinates.

You can abbreviate the COORDINATES statement as COORD.

XCOORD=(*variable-name***)**
XC=(*variable-name***)**
gives the name of the variable that contains the *x* coordinate of the data in the DATA= data set.

YCOORD=(*variable-name***)**
YC=(*variable-name***)**
gives the name of the variable that contains the *y* coordinate of the data locations in the DATA= data set.

GRID Statement

GRID *grid-options* < / *option* > **;**

The GRID statement specifies the grid of spatial locations at which to perform the simulations. A single GRID statement is required and is applied to all SIMULATE statements. Specify the grid in one of the following three ways:

- Specify the *x* and *y* coordinates explicitly for a grid in two dimensions.
- Specify the NPTS= option in addition to the *x* and *y* coordinates to define a grid of individual points or in one dimension.
- Specify the coordinates by using a SAS data set for a grid of individual points or in one dimension.

The GRID statement has the following *grid-options*:

NPTS=*number* | **ALL**
controls specification of a grid in one dimension or a grid of individual simulation locations.

When you specify the NPTS=*number* option and the coordinates of two points in the GRIDDATA= data set or in both the X= and Y= options, you request a linear simulation grid. Its direction is across the line defined by the specified points. The grid size is equal to the *number* of points that you specify in the NPTS= option, where *number* ≥ 2.

When you specify the NPTS=ALL option and the coordinates for any number of points in the GRIDDATA= data set or in each of the X= and Y= options, the SIM2D procedure performs simulation

only at the specified individual locations. Use the NPTS=ALL option to examine a set of individual points anywhere on the XY plane or to specify a custom grid in one dimension.

If the number of x coordinates and the number of y coordinates in the X= and Y= options, respectively, are different, then the NPTS= option is ignored; in that case, a two-dimensional grid is used according to the specified X= and Y= options.

If you specify a simulation grid with any number of points other than two in the GRIDDATA= data set, then the option NPTS=ALL has the same effect as omitting the NPTS= option.

X=*number*

X=$x_1, \ldots, x_m$

X=x_1 **TO** x_m

X=x_1 **TO** x_m **BY** δx

specifies the x coordinate of the grid locations.

Y=*number*

Y=$y_1, \ldots, y_m$

Y=y_1 **TO** y_m

Y=y_1 **TO** y_m **BY** δy

specifies the y coordinate of the grid locations.

Use the X= and Y= options of the GRID statement to specify a grid in one or two dimensions, or a grid of individual simulation locations.

For example, the following two GRID statements are equivalent:

```
GRID X=1,2,3,4,5  Y=0,2,4,6,8,10;

GRID X=1 TO 5     Y=0 to 10 by 2;
```

In the following example, the first GRID statement produces a grid in two dimensions. The second statement produces simulation output only for the four individual points at the locations (1,0), (2,5), (3,7), and (4,10) on the XY plane.

```
GRID X=1 TO 4 Y=0,5,7,10;

GRID X=1 TO 4 Y=0,5,7,10 NPTS=ALL;
```

In the next example, the first GRID statement specifies a 2-by-2 grid in two dimensions. The second GRID statement specifies a linear grid of eight points. The grid is in the direction of the line defined by the specified points (2,8) and (3,5) on the XY plane and it extends between these two points.

```
GRID X=2,3 Y=8,5;

GRID x=2,3 Y=8,5 NPTS=8;
```

The last example shows a GRID statement that specifies a linear grid made of seven points across the Y axis. In this case, the syntax is sufficient to fully define a linear grid without the NPTS= option.

```
GRID X=5 Y=3 TO 9;
```

To specify grid locations from a SAS data set, you must provide the name of the data set and the variables that contain the values of the *x* and *y* coordinates.

GRIDDATA=*SAS-data-set*

GDATA=*SAS-data-set*

specifies a SAS data set that contains the *x* and *y* grid coordinates. Use the GRIDDATA= option of the GRID statement to specify a grid in one dimension or a grid of individual simulation locations.

XCOORD=(*variable-name***)**

XC=(*variable-name***)**

gives the name of the variable that contains the *x* coordinate of the grid locations in the GRIDDATA= data set.

YCOORD=(*variable-name***)**

YC=(*variable-name***)**

gives the name of the variable that contains the *y* coordinate of the grid locations in the GRIDDATA= data set.

You can specify the following *option* in the GRID statement after a slash (/):

LABEL < (*suboption***) > = (***character-list***)**

specifies labels to tag grid points in simulation plots when you use grids in one dimension. You can specify one or more such labels as quoted strings in the *character-list*.

When the number of labels in the *character-list* exceeds the number of points in your grid, the labels in the list are used sequentially and any labels in excess are ignored. When the number of labels in the *character-list* is smaller than the number of points in your grid, the behavior is as follows:

- If an area map is included in the simulation plot, then blank labels are assigned to the remaining nonlabeled grid points on the map.
- For the simulation band and box plots, the coordinates of nonlabeled grid points are automatically assigned as their labels.

If the grid points are collinear and the horizontal axis displays distance, then two labels appear by default in the simulation plot. These are assigned to the first and the last points of the grid to help identify the ends of the linear grid segment on the plot map. This label pair is shown only when the plot includes an area map. Specifically, the two labels appear when you request simulation band plots, or simulation box plots for which you specify the SIM(SHOWD) suboption, if applicable. The two labels do not appear if you specify explicitly the NOMAP suboption in the PLOTS=SIM option.

The two labels have default values, unless you choose to specify your own labels with the LABEL= option. If you specify more than two labels in the *character-list* under these conditions, then only the first and last labels in the list are used; any additional labels in between are ignored.

The LABEL= option has the following *suboption*:

ALL

specifies that all individual points in the grid are assigned sequentially the labels you specify in the LABEL(ALL)= option when the SIM(SHOWD) suboption is applicable and specified in a simulation box plot. In all other cases, the ALL suboption is ignored.

The ALL suboption enables you to override the default behavior when the SIM(SHOWD) suboption is specified (the default behavior is to display labels only for the first and last grid points). As a result, you can use the ALL suboption to label grid points in both conditional and unconditional simulation tasks regardless of whether you specify the NOMAP suboption in the PLOTS=SIM option.

The LABEL= option is ignored when you produce simulation plots of grids in two dimensions.

ID Statement

ID *variable* **;**

The ID statement specifies which variable to include for identification of the observations in the labels and tool tips of the OBSERVATIONS plot and the tool tips of the SIM plot. The ID statement has an effect only when you perform conditional simulation.

In the SIM2D procedure you can specify only one ID variable in the ID statement. If no ID statement is given, then PROC SIM2D uses the observation number in the plots.

RESTORE Statement

RESTORE IN=*store-name* < / *option* > **;**

The RESTORE statement specifies an item store that provides spatial correlation model input for the PROC SIM2D simulation tasks. An item store is a binary file defined by the SAS System. You cannot modify the contents of an item store. The SIM2D procedure can use only item stores created by PROC VARIOGRAM.

Item stores enable you to use saved correlation models without having to repeat specification of these models in the SIMULATE statement. In principle, an item store contains the chosen model from a model fitting process in PROC VARIOGRAM. If more than one model form is fitted, then all successful fits are included in the item store. In this case, you can choose any of the available models to use for simulation with the STORESELECT(MODEL=) option in the SIMULATE statement. Successfully fitted models might include questionable fits, which are so flagged when you specify the INFO option to display model names.

The *store-name* is a usual one- or two-level SAS name, as for SAS data sets. If you specify a one-level name, then the item store resides in the WORK library and is deleted at the end of the SAS session. Since item stores are often used for postprocessing tasks, typical usage specifies a two-level name of the form *libname.membername*.

When you specify the RESTORE statement, the default output contains some general information about the input item store. This information includes the store name, label (if assigned), the data set that was used to create the store, BY group information, the procedure that created the store, and the creation date.

You can specify the following *option* in the RESTORE statement after a slash (/):

INFO < (*info-options*) >

specifies that additional information about the input item store be printed. This information is provided in two ODS tables. One table displays the variables in the item store, in addition to the mean and standard deviation for each of them. These statistics are based on the observations that were used to produce the store results. The second table shows the model on top of the list of all fitted models for each direction angle in the item store. The INFO option has the following *info-options*:

DETAILS

DET

specifies that more detailed information be displayed about the input item store. This option produces the full list of models for each direction angle in the item store, in addition to the model equivalence class. For more information about classes of equivalence, see the section "Classes of Equivalence" on page 11034 in Chapter 131, "The VARIOGRAM Procedure." The DETAILS option is ignored if the input item store contains information about a single fitted model.

ONLY

specifies that only information about the input item store without any simulation tasks be displayed.

Each variable in an item store has a mean value that is passed to PROC SIM2D and used in simulations. If the mean in the item store is a missing value, then a zero mean is used by default. Specify the "MEAN Statement" on page 9525 to override the mean information in the item store. For example, if you want to use only the correlation model in the item store and exclude the accompanying mean, then explicitly specify a zero mean in the "MEAN Statement" on page 9525.

When you specify an input item store with the RESTORE statement in PROC SIM2D, all the DATA= input data set variables must match input item store variables. If there are BY groups in the input DATA= set or in the input RESTORE variables, then PROC SIM2D handles the different cases as follows:

- If both PROC SIM2D has BY groups and the RESTORE statement has BY groups, then the analysis variables must match. This matching assumes implicitly that in each BY group of PROC SIM2D and the item store, the corresponding set of observations and correlation model comes from the same random field. This assumption is valid if you use the same data set, first in PROC VARIOGRAM to fit a model and save it in the item store, and then in PROC SIM2D to run simulations with the resulting correlation models.

- If PROC SIM2D has BY groups but the item store does not, then the item store is accepted only if the procedure and the item store analysis variables match. In this case, the same item store model choice iterates across the BY groups of the input data. You are advised to proceed with caution: each BY group in the input DATA= set corresponds to a different realization of a random field. Hence, by using the same correlation model for simulation purposes, you implicitly assume that all these different realizations are instances of the same random field.

- If PROC SIM2D has an input DATA= set and no BY groups but the item store has BY groups, then the item store is rejected

- If PROC SIM2D has no input DATA= set and the item store has BY groups, then PROC SIM2D runs unconditional simulations for the models in the store BY groups. See also the BY statement for more about the behavior in this case.

SIMULATE Statement

SIMULATE *simulate-options* **;**

The SIMULATE statement specifies details on the simulation and the covariance model used in the simulation. You can specify the following *simulate-options* with a SIMULATE statement, which can be abbreviated by SIM.

Table 112.3 summarizes the *options* available in the SIMULATE statement.

Table 112.3 SIMULATE Statement Options

Option	Description
Simulate-Options	
NUMREAL=	Specifies the number of realizations to produce
VAR=	Specifies the conditioning variable
Covariance Model Specification	
ANGLE=	Specifies the angle of the major axis
FORM=	Specifies the functional form
MDATA=	Specifies the input data set containing parameter values
NUGGET=	Specifies the nugget effect for the model
RANGE=	Specifies the range parameter
RATIO=	Specifies the ratio of the minor axis to the major axis
SCALE=	Specifies the scale (or *sill*) parameter
SEED=	Specifies the seed to use for the random number generator
SINGULAR=	Gives the singularity criterion
SMOOTH=	Specifies the smoothness parameter
STORESELECT	Uses information from an input item store for prediction

NUMREAL=*number*

NUMR=*number*

NR=*number*

specifies the number of realizations to produce for the spatial process specified by the covariance model. As a result, the number of observations in the OUTSIM= data set contributed by a given SIMULATE statement is the product of the NUMREAL= value and the number of grid points. This can cause the OUTSIM= data set to become large even for moderate values of the NUMREAL= option.

VAR=(*variable-name***)**

specifies the single numeric variable used as the conditioning variable in the simulation. In other words, the simulation is conditional on the values of the VAR= variable found in the DATA= data set. If you omit the VAR= option or if all observations of the VAR= variable are missing values, then the simulation is *unconditional*. Since multiple SIMULATE statements are allowed, you can perform both unconditional and conditional simulations with a single PROC SIM2D statement.

Covariance Model Specification

You can specify a semivariogram or covariance model in three ways:

- You specify the required parameters SCALE, RANGE, FORM, and SMOOTH (if you specify the MATERN form), and possibly the optional parameters NUGGET, ANGLE, and RATIO, explicitly in the SIMULATE statement.
- You specify an MDATA= data set. This data set contains variables that correspond to the required parameters SCALE, RANGE, FORM, and SMOOTH (if you specify the MATERN form), and, optionally, variables for the NUGGET, ANGLE, and RATIO parameters.
- You can specify an input item store in the RESTORE statement. The item store contains one or more correlation models for one or more direction angles. You can specify these models in the STORESELECT option of the SIMULATE statement to run a simulation.

The three methods are mutually exclusive: you specify all parameters explicitly, they are all are read from the MDATA= data set, or you select a model and its parameters from an input item store. The following *simulate-options* are related to model specification:

ANGLE=*angle* **|** **(***angle1, . . . , anglek***)**
specifies the angle of the major axis for anisotropic models, measured in degrees clockwise from the N–S axis. The default is ANGLE=0.

In the case of a nested semivariogram model with k nestings, you have the following two ways to specify the anisotropy major axis: you can specify only one *angle* which is then applied to all nested forms, or you can specify one angle for each of the k nestings.

NOTE: The syntax makes it possible to specify different angles for different forms of the nested model, but this practice is rarely used.

FORM=*form* **|** **(***form1, . . . , formk***)**
specifies the functional form (type) of the semivariogram model. Use the syntax with the single *form* to specify a non-nested model. Use the syntax with forms *formi*, $i = 1, \ldots, k$, to specify a nested model with k structures. Each of the forms can be any of the following:

CUBIC | EXPONENTIAL | GAUSSIAN | MATERN |
PENTASPHERICAL | SINEHOLEEFFECT | SPHERICAL
CUB | EXP | GAU | MAT | PEN | SHE | SPH
specify the functional form (type).

For example, the syntax

```
FORM=GAU
```

specifies a model with a single Gaussian structure. Also, the syntax

```
FORM=(EXP,SHE,MAT)
```

specifies a nested model with an exponential, a sine hole effect, and a Matérn structure. Finally

```
FORM=(EXP,EXP)
```

specifies a nested model with two structures both of which are exponential.

NOTE: In the documentation, models are named either by using their full names or by using the first three letters of their structures. Also, the names of different structures in a nested model are separated by a hyphen (-). According to this convention, the previous examples illustrate how to specify a GAU, an EXP-SHE-MAT, and an EXP-EXP model, respectively, with the FORM= option.

All the supported model forms have two parameters specified by the SCALE= and RANGE= options, except for the MATERN model which has a third parameter specified by the SMOOTH= option. A FORM= value is required, unless you specify the MDATA= option or the STORESELECT option.

Computation of the MATERN covariance is numerically demanding. As a result, simulations that use Matérn covariance structures can be time-consuming.

MDATA=*SAS-data-set*

specifies the input data set that contains parameter values for the covariance or semivariogram model. The MDATA= option cannot be combined with any of the FORM= or STORESELECT options.

The MDATA= data set must contain variables named SCALE, RANGE, and FORM, and it can optionally contain variables NUGGET, ANGLE, and RATIO. If you specify the MATERN form, then you must also include a variable named SMOOTH in the MDATA= data set.

The FORM variable must be a character variable, and it can assume only the values allowed in the explicit FORM= syntax described previously. The RANGE, SCALE and SMOOTH variables must be numeric. The optional variables ANGLE, RATIO, and NUGGET must also be numeric if present.

The number of observations present in the MDATA= data set corresponds to the level of nesting of the covariance or semivariogram model. For example, to specify a non-nested model that uses a spherical covariance, an MDATA= data set might contain the following statements:

```
data md1;
   input scale range form $;
   datalines;
   25 10 SPH
;
```

The PROC SIM2D statement to use the MDATA= specification is of the form shown in the following:

```
proc sim2d data=...;
   sim var=.... mdata=md1;
run;
```

This is equivalent to the following explicit specification of the covariance model parameters:

```
proc sim2d data=...;
   sim var=.... scale=25 range=10 form=sph;
run;
```

The following MDATA= data set is an example of an anisotropic nested model:

```
data md2;
   input scale range form $ nugget angle ratio smooth;
   datalines;
   20 8 SPH  5 35 .7   .
   12 3 MAT  5 0  .8 2.8
   4  1 GAU  5 45 .5   .
;


proc sim2d data=...;
   sim var=.... mdata=md2;
run;
```

This is equivalent to the following explicit specification of the covariance model parameters:

```
proc sim2d data=...;
   sim var=.... scale=(20,12,4) range=(8,3,1) form=(SPH,MAT,GAU)
            angle=(35,0,45) ratio=(.7,.8,.5) nugget=5 smooth=2.8;
run;
```

This example is somewhat artificial in that it is usually hard to detect different anisotropy directions and ratios for different nestings by using an experimental semivariogram. **NOTE:** The NUGGET variable value is the same for all nestings. This is always the case; the nugget effect is a single additive term for all models. For further details, see the section "The Nugget Effect" on page 5585 in Chapter 74, "The KRIGE2D Procedure."

The example also shows that if you specify a MATERN form in the nested model, then the SMOOTH variable must be specified for all nestings in the MDATA= data set. You simply specify the SMOOTH value as missing for nestings other than MATERN.

The SIMULATE statement can be given a label. This is useful for identification in the OUTSIM= data set when multiple SIMULATE statements are specified. For example:

```
proc sim2d data=...;
   gauss1: sim var=.... form=gau;
   mean ....;
   gauss2: sim var=.... form gau;
   mean ....;
   exp1: sim var=.... form=exp;
   mean ....;
   exp2: sim var=.... form=exp;
   mean ....;
run;
```

In the OUTSIM= data set, the values "GAUSS1," "GAUSS2," "EXP1," and "EXP2" for the LABEL variable help to identify the realizations that correspond to the four SIMULATE statements. If you do not provide a label for a SIMULATE statement, a default label of SIM*n* is given, where *n* is the number of unlabeled SIMULATE statements seen so far.

NUGGET=*number*

specifies the nugget effect for the model. This effect is due to a discontinuity in the semivariogram as determined by plotting the sample semivariogram (see the section "The Nugget Effect" on page 5585 in Chapter 74, "The KRIGE2D Procedure," for details). For models without any nugget effect, the NUGGET= option is left out. The default is NUGGET=0.

RANGE=*range* | **(***range1, . . . , rangek***)**

specifies the range parameter in the semivariogram models. In the case of a nested semivariogram model with *k* nestings, you must specify a range for each nesting.

The range parameter is the divisor in the exponent in all supported models. It has the units of distance or distance squared for these models, and it is related to the correlation scale for the underlying spatial process.

See the section "Theoretical Semivariogram Models" on page 5577 in Chapter 74, "The KRIGE2D Procedure," for details about how the RANGE= values are determined.

RATIO=*ratio* | **(***ratio1, . . . , ratiok***)**

specifies the ratio of the length of the minor axis to the length of the major axis for anisotropic models. The value of the RATIO= option must be between 0 and 1. In the case of a nested semivariogram model with *k* nestings, you can specify a ratio for each nesting. The default is RATIO=1.

SCALE=*scale* | **(***scale1, . . . , scalek***)**

specifies the scale (or *sill*) parameter in semivariogram models. In the case of a nested semivariogram model with *k* nestings, you must specify a scale for each nesting. The scale parameter is the multiplicative factor in all supported models; it has the same units as the variance of the VAR= variable.

See the section "Theoretical Semivariogram Models" on page 5577 in Chapter 74, "The KRIGE2D Procedure," for details about how the SCALE= values are determined.

SEED=*seed-value*

specifies the seed to use for the random number generator. The SEED= option *seed-value* has to be an integer.

SINGULAR=*number*

gives the singularity criterion for solving the set of linear equations involved in the computation of the mean and covariance of the conditional distribution associated with a given SIMULATE statement. The larger the value of the SINGULAR= option, the easier it is for the covariance matrix system to be declared singular. The default is SINGULAR=1E–8.

For more details about the use of the SINGULAR= option, see the section "Computational and Theoretical Details of Spatial Simulation" on page 9526.

SMOOTH=*smooth* | (*smooth1, . . . , smoothm*)

specifies the smoothness parameter $\nu > 0$ in the Matérn type of semivariance structures. The special case $\nu = 0.5$ is equivalent to the exponential model, whereas $\nu \to \infty$ gives the Gaussian model.

When you specify m different MATERN forms in the FORM= option, you must also provide m smoothness values in the SMOOTH option. If you must specify more than one smoothness value, the values are assigned sequentially to the MATERN nestings in the order the nestings are specified. If you specify more smoothness values than necessary, then values in excess are ignored.

STORESELECT(*ssel-options***)**

SSEL(*ssel-options***)**

specifies that information from an input item store be used for the prediction. You cannot combine the STORESELECT option with any of the FORM= or MDATA= options. The STORESELECT option has the following *ssel-options*:

TYPE=*field-type*

specifies whether to perform isotropic or anisotropic simulation. You can choose the *field-type* from one of the following:

ISO

specifies isotropic field for the simulation.

ANIGEO | GEO

specifies a field with geometric anisotropy for the simulation.

ANIZON(*zonal-form1, . . . , zonal-formn***)**

ZON(*zonal-form1, . . . , zonal-formn***)**

specifies a field with zonal anisotropy for the simulation. Each *zonal-formi*, $i = 1, \ldots, n$, can be any of the following:

CUB | EXP | GAU | MAT | PEN | SHE | SPH

specify a form for the simulation.

Each *zonal-formi*, $i = 1, \ldots, n$, is a structure in the purely zonal component of the correlation model in the direction angle of the minor anisotropy axis. For this reason, when you specify the TYPE=ANIZON suboption you must also specify the nonzonal component of the correlation model in the MODEL= suboption of the STORESELECT option. Assume the nonzonal component has k structures; these are common across all directions and each one has the same scale in all directions. In that sense, you use the TYPE=ANIZON suboption to specify only the n zonal anisotropy structures of an input store $(k + n)$-structure nested model in the direction angle of the minor anisotropy axis.

Given this specification, $k+n$ must be up to the maximum number of nested model structures that is supported by the item store. See also the MODEL= suboption of the STORESELECT option.

In conclusion, you can use an input item store for prediction with zonal anisotropy if you know that every structure in the nonzonal model component has the same scale across all directions. When this condition does not apply for the item store models, specify the model parameters explicitly in the SIMULATE statement.

Computation of the MATERN covariance is numerically demanding. As a result, predictions that use Matérn covariance structures can be time-consuming.

If you omit the TYPE= option, the default behavior is TYPE=ISO when the input item store contains information for only one angle or for the omnidirectional case. If you specify an item store with information for more than one direction, then the default behavior is TYPE=ANIGEO.

When you specify TYPE=ISO to request isotropic analysis in the presence of an item store with information for multiple directions, you must specify the ANGLEID= suboption of the STORESELECT option with one argument. This argument specifies which of the direction angles information to use for the isotropic analysis.

When you indicate the presence of anisotropy with the TYPE=ANIGEO or TYPE=ANIZON suboptions of the STORESELECT option, the following conditions apply:

- You must specify the ANGLEID= suboption of the STORESELECT option to designate the major and minor anisotropy axes. See the ANGLEID= suboption of the STORESELECT option for details.
- – For TYPE=ANIGEO, ensure that you have the same scale in all anisotropy directions.
  – For TYPE=ANIZON, ensure that the nonzonal component scale is the same in all anisotropy directions.

  If you import a nested model, these rules also apply to each one of the nested structures.
- Model ranges in the major anisotropy axis must be longer than ranges in the minor anisotropy axis.
- Any Matérn covariance structure must maintain its smoothness parameter value in all anisotropy directions.

ANGLEID=*angleid1* | (*angleid1, angleid2*)

specifies which direction angles in the input item store be used for simulation. The angles are identified by the corresponding number in the AngleID column of the "Store Models Information" table, or by the AngleID parameter in the table title when you specify the INFO(DETAILS) option in the RESTORE statement.

If you request isotropic prediction in the TYPE= suboption of the STORESELECT option and the item store has omnidirectional contents or information about only one angle, then the ANGLEID= option is ignored. The simulation input comes from the omnidirectional information. In the case of a single angle, you still perform isotropic simulation and the model parameters are provided by the model in the single direction angle in the item store. However, if the item store contains information for more than one angle, then you must specify one angle ID in *angleid1*. The model information from the corresponding angle is then used in your isotropic simulation.

When you specify an anisotropic simulation in the TYPE= option of the STORESELECT option, you need to have information about two perpendicular direction angles. One of them is the major and the other is the minor anisotropy axis. You must always specify the major anisotropy axis angle ID in *angleid1* and the minor anisotropy axis angle ID in *angleid2*. This means that the range parameters of the model forms in the angle designated by the *angleid1* need to be larger than the corresponding ranges of the forms in the angle designated by the *angleid2*. Conveniently, if the item store has only two angles, then you only need to specify the ID *angleid1* of the major anisotropy axis angle. If the item store has only one angle, then you cannot perform anisotropic simulation with input from the item store.

NOTE: You can perform geometric anisotropic analysis even if the item store does not contain information about a direction that is perpendicular to the one specified by *angleid1*. This is possible due to the geometry of the ellipse. In particular, when you specify the major axis with *angleid1* and an angle ID for a second direction with a corresponding smaller range, then PROC SIM2D automatically computes the minor anisotropy axis range and the necessary range ratio parameter.

Anisotropic analysis is not possible when you specify instances of the same angle in the input item store. It is possible that PROC VARIOGRAM produces an item store where two or more directions can be the same if their corresponding correlation models were obtained for different angle tolerances or bandwidths in the VARIOGRAM procedure. Consequently, you cannot specify anisotropic simulation if the input store contains only two angles that are the same or if you specify *angleid1* and *angleid2* that correspond to equal angles.

MODEL=*form* **|** **(***form1, . . . , formk***)**
specifies the theoretical semivariogram model selection to use for the simulation. Use any combination of one, two, or three forms to describe a model in the input item store because up to three nested structures are supported. Each *formi*, $i = 1, \ldots, k$, can be any of the following:

CUB | EXP | GAU | MAT | PEN | SHE | SPH
specify the selection model.

Computation of the MATERN covariance is numerically demanding. As a result, simulations that use Matérn covariance structures can be time-consuming.

All fitted models that are stored in the input item store contain information about their component parameters and also about the nugget effect if any. The SIM2D procedure retrieves this information when you make a model selection in the MODEL= option, and you do not need to individually specify a nugget effect or any other parameter of the model.

By default, the model that is ranked first among the models for a given angle in the item store is used for the simulation task. If more than one model is available in the item store, then you can specify the MODEL= option to use a different model for the simulation.

In an anisotropic simulation, the default selection is the model that is ranked first in the direction angle of the major anisotropy axis. If you specify the TYPE=ANIGEO option, then a model that consists of identical structures needs to be present in the selected minor anisotropy axis angle in the item store. If you specify the TYPE=ANIZON option, then a model with the exact same first k structures must be present in the selected minor anisotropy axis angle, and it must feature at least one more structure as a zonal component. The zonal component is specified separately in the TYPE=ANIZON suboption of the STORESELECT option. Consequently, remember that in zonal anisotropy the MODEL= suboption designates only the nonzonal component of the correlation model in the minor anisotropy axis direction. In all, if there are k common structures and n structures in the purely zonal component, then $k + n$ must be up to the maximum number of nested model structures that is supported by the item store.

SVAR=*store-var* | (*store-varlist*)

specifies one *store-var* item store variable or a list *store-varlist* of variables that are present in the item store. This option selects one or more item store variables whose correlation models you want to use in the current simulation task.

If you are performing a conditional simulation, then PROC SIM2D searches the input item store for the variable that is specified in the VAR= option of the SIMULATE statement. Then, the procedure selects the appropriate correlation model for the task. In this case, if you specify the SVAR= option, it is ignored. However, when you request an unconditional simulation and specify input from an item store, then you must also use SVAR= to specify a source for your correlation model.

In comparison to the other two ways of specifying a correlation model in PROC SIM2D, the STORESELECT option is quite different because you can avoid explicit specification of all parameter values of a model. When you specify the STORESELECT option, then the corresponding scale, range, nugget effect, and smoothness (if appropriate) parameter values are invoked as saved attributes of the model that you select from the item store.

In the case of anisotropy, you specify the angles indirectly with the ANGLEID= option of the STORESELECT option, and the ratios are computed implicitly by using the selected model ranges. Explore how to specify valid anisotropical models imported from an input item store with the two examples that follow.

In the first example, assume the input item store InStoreGeo contains exponential models in the angles $\theta_1 = 0°$, $\theta_2 = 45°$, and $\theta_3 = 90°$. You know in advance that all models have the same scale $c_1 = c_2 = c_3$ across these directions and that the respective ranges are $a_1 = 15$, $a_2 = 20$, and $a_3 = 25$ in distance units. Hence, you have a case of geometric anisotropy where the major anisotropy axis is in the direction of angle θ_3 and the minor anisotropy axis is in the direction of angle θ_1. The following statements in PROC SIM2D use the information in the item store InStoreGeo to perform simulation under the assumption of geometric anisotropy:

```
proc sim2d data=...;
   restore in=InStoreGeo;
   simulate storeselect(model=exp type=anigeo angleid=(3,1));
run;
```

For the second example, assume a case of zonal anisotropy. Consider the input item store InStoreZon, which contains models in the two angles, $\theta_1 = 30°$ and $\theta_2 = 120°$. Specifically, in θ_1 you have an exponential-spherical model: the exponential structure has scale $c_{1E} = 3$ and range $a_{1E} = 10$; the spherical structure has scale $c_{1S} = 1$ and range $a_{1S} = 6$. In direction θ_2 you have an exponential model with scale $c_{1E} = 3$ and range $a_{1E} = 12$. Hence, the zonal anisotropy major axis is in the direction of the lowest total variance, which is in angle θ_2; then, the minor axis is in the direction of angle θ_1. The following statements in PROC SIM2D use the information in the store InStoreZon to perform simulation under the assumption of zonal anisotropy:

```
proc sim2d data=...;
   restore in=InStoreZon;
   simulate storeselect(model=exp type=anizon(sph) angleid=(2,1));
run;
```

MEAN Statement

MEAN *spec1*, . . . , *spec6* **;**

MEAN QDATA=*SAS-data-set* **CONST=***var1* **CX=***var2* **CY=***var3*
CXX=*var4* **CYY=***var5* **CXY=***var6* **;**

MEAN QDATA=*SAS-data-set* **;**

A mean function $\mu(s)$ that is a quadratic in the coordinates can be written as

$$\mu(s) = \mu(x, y) = \beta_0 + \beta_1 x + \beta_2 y + \beta_3 x^2 + \beta_4 y^2 + \beta_5 xy$$

The MEAN statement specifies the quadratic surface to use as the mean function for the simulated SRF. There are two ways to specify the MEAN statement. The MEAN statement allows the specification of the coefficients $\beta_0, \ldots, \beta_5$ either explicitly or through a QDATA= data set.

An example of an explicit specification is the following:

```
mean 1.4 + 2.5*x + 3.6*y + 0.47*x*x + 0.58*y*y + 0.69*x*y;
```

In this example, all terms have a nonzero coefficient. Any term with a zero coefficient is simply left out of the specification. For example,

```
mean 1.4;
```

is a valid quadratic form with all terms having zero coefficients except the constant term.

An equivalent way of specifying the mean function is through the QDATA= data set. For example, the MEAN statement

```
mean 1.4 + 2.5*x + 3.6*y + 0.47*x*x + 0.58*y*y + 0.69*x*y;
```

can be alternatively specified by the following DATA step and MEAN statement:

```
data q1;
   input c1 c2 c3 c4 c5 c6;
   datalines;
   1.4 2.5 3.6 0.47 0.58 0.69
;

proc sim2d data=....;
   simulate ...;
   mean qdata=q1 const=c1 cx=c2 cy=c3 cxx=c4 cyy=c5 cxy=c6;
run;
```

The QDATA= data set specifies the data set containing the coefficients. The parameters CONST=, CX=, CY=, CXX=, CYY=, and CYX= specify the variables in the QDATA= data set that correspond to the constant, linear y, linear y, and so on. For any coefficient not specified in this list, the QDATA= data set is checked for the presence of variables with default names of CONST, CX, CY, CXX, CYY, and CXY. If these variables are present, their values are taken as the corresponding coefficients. Hence, you can rewrite the previous example as follows:

```
data q1;
   input const cx cy cxx cyy cxy;
   datalines;
   1.4 2.5 3.6 0.47 0.58 0.69
;

proc sim2d data=....;
   simulate ...;
   mean qdata=q1;
run;
```

If a given coefficient does not appear in the list or in the data set with the default name, a value of zero is assumed.

If you run a simulation task with input from a RESTORE statement, then by default the simulation uses the mean of the item store variable in the simulation. You can override this default behavior if you explicitly specify the MEAN statement with a different mean function.

Details: SIM2D Procedure

Computational and Theoretical Details of Spatial Simulation

Introduction

There are a number of approaches to simulating spatial random fields or, more generally, simulating sets of dependent random variables. These include sequential indicator methods, turning bands, and the Karhunen-Loeve expansion. See Christakos (1992, Chapter 8) and Deutsch and Journel (1992, Chapter V) for details.

A particularly simple method available for Gaussian spatial random fields is the LU decomposition method. This method is computationally efficient. For a given covariance matrix, the $\mathrm{LU} = \mathbf{LL}'$ decomposition is computed once, and the simulation proceeds by repeatedly generating a vector of independent $N(0,1)$ random variables and multiplying by the $\mathbf{L}$ matrix.

One problem with this technique is memory requirements; memory is required to hold the full data and grid covariance matrix in core. While this is especially limiting in the three-dimensional case, you can use PROC SIM2D, which handles only two-dimensional data, for moderately sized simulation problems.

Theoretical Development

It is a simple matter to produce an $N(0,1)$ random number, and by stacking k $N(0,1)$ random numbers in a column vector, you can obtain a vector with independent standard normal components $\mathbf{W} \sim N_k(\mathbf{0}, \mathbf{I})$. The meaning of the terms *independence* and *randomness* in the context of a deterministic algorithm required for the generation of these numbers is subtle; see Knuth (1981, Chapter 3) for details.

Rather than $\mathbf{W} \sim N_k(\mathbf{0}, \mathbf{I})$, what is required is the generation of a vector $\mathbf{Z} \sim N_k(\mathbf{0}, \mathbf{C})$—that is,

$$\mathbf{Z} = \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_k \end{bmatrix}$$

with covariance matrix

$$\mathbf{C} = \begin{pmatrix} C_{11} & C_{12} & \cdots & C_{1k} \\ C_{21} & C_{22} & \cdots & C_{2k} \\ & \ddots & & \\ C_{k1} & C_{k2} & \cdots & C_{kk} \end{pmatrix}$$

If the covariance matrix is symmetric and positive definite, it has a Cholesky root $\mathbf{L}$ such that $\mathbf{C}$ can be factored as

$$\mathbf{C} = \mathbf{L}\mathbf{L}'$$

where $\mathbf{L}$ is lower triangular. See Ralston and Rabinowitz (1978, Chapter 9, Section 3-3) for details. This vector $\mathbf{Z}$ can be generated by the transformation $\mathbf{Z} = \mathbf{L}\mathbf{W}$. Here is where the assumption of a Gaussian SRF is crucial. When $\mathbf{W} \sim N_k(\mathbf{0}, \mathbf{I})$, then $\mathbf{Z} = \mathbf{L}\mathbf{W}$ is also Gaussian. The mean of $\mathbf{Z}$ is

$$E(\mathbf{Z}) = \mathbf{L}(E(\mathbf{W})) = \mathbf{0}$$

and the variance is

$$\mathrm{Var}(\mathbf{Z}) = \mathrm{Var}(\mathbf{L}\mathbf{W}) = E(\mathbf{L}\mathbf{W}\mathbf{W}'\mathbf{L}') = \mathbf{L}E(\mathbf{W}\mathbf{W}')\mathbf{L}' = \mathbf{L}\mathbf{L}' = \mathbf{C}$$

Consider now an SRF $Z(\boldsymbol{s}), \boldsymbol{s} \in D \subset \mathcal{R}^2$, with spatial covariance function $C(\boldsymbol{h})$. Fix locations $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_k$, and let $\mathbf{Z}$ denote the random vector

$$\mathbf{Z} = \begin{bmatrix} Z(\boldsymbol{s}_1) \\ Z(\boldsymbol{s}_2) \\ \vdots \\ Z(\boldsymbol{s}_k) \end{bmatrix}$$

with corresponding covariance matrix

$$\mathbf{C}_z = \begin{pmatrix} C(\mathbf{0}) & C(\boldsymbol{s}_1 - \boldsymbol{s}_2) & \cdots & C(\boldsymbol{s}_1 - \boldsymbol{s}_k) \\ C(\boldsymbol{s}_2 - \boldsymbol{s}_1) & C(\mathbf{0}) & \cdots & C(\boldsymbol{s}_2 - \boldsymbol{s}_k) \\ & \ddots & & \\ C(\boldsymbol{s}_k - \boldsymbol{s}_1) & C(\boldsymbol{s}_k - \boldsymbol{s}_2) & \cdots & C(\mathbf{0}) \end{pmatrix}$$

Since this covariance matrix is symmetric and positive definite, it has a Cholesky root, and the $Z(\boldsymbol{s}_i), i = 1, \ldots, k$, can be simulated as described previously. This is how the SIM2D procedure implements unconditional simulation in the zero-mean case. More generally,

$$Z(\boldsymbol{s}) = \mu(\boldsymbol{s}) + \varepsilon(\boldsymbol{s})$$

where $\mu(\boldsymbol{s})$ is a quadratic form in the coordinates $\boldsymbol{s} = (x, y)$ and the $\varepsilon(\boldsymbol{s})$ is an SRF that has the same covariance matrix $\mathbf{C}_z$ as previously. In this case, the $\mu(\boldsymbol{s}_i), i = 1, \ldots, k$, is computed once and added to the simulated vector $\varepsilon(\boldsymbol{s}_i), i = 1, \ldots, k$, for each realization.

For a conditional simulation, this distribution of

$$\mathbf{Z} = \begin{bmatrix} Z(\boldsymbol{s}_1) \\ Z(\boldsymbol{s}_2) \\ \vdots \\ Z(\boldsymbol{s}_k) \end{bmatrix}$$

must be conditioned on the observed data. The relevant general result concerning conditional distributions of multivariate normal random variables is the following. Let $\mathbf{X} \sim N_m(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix}$$

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}$$

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}$$

The subvector $\mathbf{X}_1$ is $k \times 1$, $\mathbf{X}_2$ is $n \times 1$, $\boldsymbol{\Sigma}_{11}$ is $k \times k$, $\boldsymbol{\Sigma}_{22}$ is $n \times n$, and $\boldsymbol{\Sigma}_{12} = \boldsymbol{\Sigma}'_{21}$ is $k \times n$, with $k + n = m$. The full vector $\mathbf{X}$ is partitioned into two subvectors, $\mathbf{X}_1$ and $\mathbf{X}_2$, and $\boldsymbol{\Sigma}$ is similarly partitioned into covariances and cross covariances.

With this notation, the distribution of $\mathbf{X}_1$ conditioned on $\mathbf{X}_2 = \boldsymbol{x}_2$ is $N_k(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}})$, with

$$\tilde{\boldsymbol{\mu}} = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2)$$

and

$$\tilde{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}$$

See Searle (1971, pp. 46–47) for details. The correspondence with the conditional spatial simulation problem is as follows. Let the coordinates of the observed data points be denoted $\tilde{\boldsymbol{s}}_1, \tilde{\boldsymbol{s}}_2, \ldots, \tilde{\boldsymbol{s}}_n$, with values $\tilde{z}_1, \tilde{z}_2, \ldots, \tilde{z}_n$. Let $\tilde{\mathbf{Z}}$ denote the random vector

$$\tilde{\mathbf{Z}} = \begin{bmatrix} Z(\tilde{\boldsymbol{s}}_1) \\ Z(\tilde{\boldsymbol{s}}_2) \\ \vdots \\ Z(\tilde{\boldsymbol{s}}_n) \end{bmatrix}$$

The random vector $\tilde{\mathbf{Z}}$ corresponds to $\mathbf{X}_2$, while $\mathbf{Z}$ corresponds to $\mathbf{X}_1$. Then $\left(\mathbf{Z} \mid \tilde{\mathbf{Z}} = \tilde{\mathbf{z}}\right) \sim N_k(\tilde{\boldsymbol{\mu}}, \tilde{\mathbf{C}})$ as in the previous distribution. The matrix

$$\tilde{\mathbf{C}} = \mathbf{C}_{11} - \mathbf{C}_{12}\mathbf{C}_{22}^{-1}\mathbf{C}_{21}$$

is again positive definite, so a Cholesky factorization can be performed.

The dimension n for $\tilde{\mathbf{Z}}$ is simply the number of nonmissing observations for the VAR= variable; the values $\tilde{z}_1, \tilde{z}_2, \ldots, \tilde{z}_n$ are the values of this variable. The coordinates $\tilde{\boldsymbol{s}}_1, \tilde{\boldsymbol{s}}_2, \ldots, \tilde{\boldsymbol{s}}_n$ are also found in the DATA= data set, with the variables that correspond to the x and y coordinates identified in the COORDINATES statement. **NOTE:** All VAR= variables use the same set of conditioning coordinates; this fixes the matrix $\mathbf{C}_{22}$ for all simulations.

The dimension k for $\mathbf{Z}$ is the number of grid points specified in the GRID statement. Since there is a single GRID statement, this fixes the matrix $\mathbf{C}_{11}$ for all simulations. Similarly, $\mathbf{C}_{12}$ is fixed.

The Cholesky factorization $\tilde{\mathbf{C}} = \mathbf{L}\mathbf{L}'$ is computed once, as is the mean correction

$$\tilde{\boldsymbol{\mu}} = \boldsymbol{\mu}_1 + \mathbf{C}_{12}\mathbf{C}_{22}^{-1}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2)$$

The means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are computed using the grid coordinates $\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_k$, the data coordinates $\tilde{\boldsymbol{s}}_1, \tilde{\boldsymbol{s}}_2, \ldots, \tilde{\boldsymbol{s}}_n$, and the quadratic form specification from the MEAN statement. The simulation is now performed exactly as in the unconditional case. A $k \times 1$ vector of independent standard $N(0, 1)$ random variables is generated and multiplied by $\mathbf{L}$, and $\tilde{\boldsymbol{\mu}}$ is added to the transformed vector. This is repeated N times, where N is the value specified for the NR= option.

Computational Details

In the computation of $\tilde{\boldsymbol{\mu}}$ and $\boldsymbol{\Sigma}$ described in the previous section, the inverse $\boldsymbol{\Sigma}_{22}^{-1}$ is never actually computed; an equation of the form

$$\boldsymbol{\Sigma}_{22}\mathbf{A} = \mathbf{B}$$

is solved for $\mathbf{A}$ by using a modified Gaussian elimination algorithm that takes advantage of the fact that $\boldsymbol{\Sigma}_{22}$ is symmetric with constant diagonal $C_z(0)$ that is larger than all off-diagonal elements. The SINGULAR= option pertains to this algorithm. The value specified for the SINGULAR= option is scaled by $C_z(0)$ before comparison with the pivot element.

Memory Usage

For conditional simulations, the largest matrix held in core memory at any one time depends on the number of grid points and data points. Using the previous notation, the data-data covariance matrix $\mathbf{C}_{22}$ is $n \times n$, where n is the number of nonmissing observations for the VAR= variable in the DATA= data set. The grid-data cross covariance $\mathbf{C}_{12}$ is $n \times k$, where k is the number of grid points. The grid-grid covariance $\mathbf{C}_{11}$ is $k \times k$. The maximum memory required at any one time for storing these matrices is

$$\max\left(k(k+1), n(n+1) + 2(n \times k)\right) \times \text{sizeof(double)}$$

There are additional memory requirements that add to the total memory usage, but usually these matrix calculations dominate, especially when the number of grid points is large.

Output Data Set

The SIM2D procedure produces a single output data set: the OUTSIM=*SAS-data-set*. The OUTSIM= data set contains all the needed information to uniquely identify the simulated values.

The OUTSIM= data set contains the following variables:

- LABEL, which is the label for the current SIMULATE statement
- VARNAME, which is the name of the conditioning variable for the current SIMULATE statement
- MODSVAR, which is the name of the input item store variable associated with the current correlation model in an unconditional simulation
- _ITER_, which is the iteration number within the current SIMULATE statement
- GXC, which is the *x* coordinate for the current grid point
- GYC, which is the *y* coordinate for the current grid point
- SVALUE, which is the value of the simulated variable

If you specify the NARROW option in the PROC SIM2D statement, the LABEL and VARNAME variables are not included in the OUTSIM= data set. This option is useful in the case where the number of data points, grid points, and realizations are such that they generate a very large OUTSIM= data set. The size of the OUTSIM= data set is reduced when these variables are not included.

In unconditional simulation tasks where no input data set is specified, the VARNAME variable is excluded from the OUTSIM= data set. In unconditional simulation tasks where you have specified an input data set, the VARNAME variable is included but given a missing value. In the case of mixed conditional and unconditional simulations (that is, when multiple SIMULATE statements are specified, among which one or more contain a VAR= specification and one or more have no VAR= specification), the VARNAME variable is included but is given a missing value for those observations that correspond to an unconditional simulation.

The MODSVAR variable is included in the OUTSIM= data set only when you specify an input item store with the RESTORE statement, and it indicates the presence of that store. This variable helps you identify the output of different unconditional simulations when the model input comes from an item store; it is not suggesting that a simulation task is conditioned upon it.

Specifically, the MODSVAR variable has a missing value for conditional simulation tasks. The variable also has a missing value for unconditional simulations for which you specify a correlation model explicitly, either with the FORM= option or with the MDATA= data set in the SIMULATE statement. In all other cases, the MODSVAR variable indicates the input item store variable that is associated with the store model used for the current unconditional simulation task.

Displayed Output

In addition to the output data set, the SIM2D procedure produces output objects as well. The SIM2D procedure output objects are the following:

- a default "Number of Observations" table that displays the number of observations read from the input data set and the number of observations used in the analysis.
- a map that shows the spatial distribution of the observations of the current VAR variable in the SIMULATE statement, in the case of conditional simulations. The observations are displayed by default with circled markers whose color indicates the VAR value at the corresponding location.
- a default table for each SIMULATE statement that summarizes the simulation specifications.
- a default table for each SIMULATE statement that shows the covariance model parameters for the corresponding simulation.
- plots of simulation outcome at each point of the specified output grid or at specified individual locations. You can produce more than one of these plots for every SIMULATE statement with styles that you can specify by using the available suboptions of the PLOTS=SIM option.
- a "Store Info" table with basic information about the input item store. This table is produced by default when you specify the RESTORE statement.
- a "Store Variables Information" table that describes the analysis variables of an input item store. The table is produced by default when you specify an item store with the RESTORE statement.
- a "Store Models Information" table with detailed information about the models and direction angles that are contained in an input item store. The table is produced by default when you specify an item store with the RESTORE statement.

ODS Table Names

Each table created by PROC SIM2D has a name associated with it, and you must use this name to reference the table when using ODS Graphics. These names are listed in Table 112.4.

Table 112.4 ODS Tables Produced by PROC SIM2D

ODS Table Name	Description	Statement	Option
ModelInfo	Parameters of the covariance model used in current simulation	PROC	Default output
NObs	Number of observations read and used	PROC	Default output
SimuInfo	General information about the simulation	PROC	Default output
StoreInfo	Input item store identity information	RESTORE	Default output

Table 112.4 *continued*

ODS Table Name	Description	Statement	Option
StoreModelInfo	Input item store direction angles and models information	RESTORE	INFO
StoreVarInfo	Input item store variables and their statistics	RESTORE	INFO

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

For additional control of the graphics that are displayed, see the PLOTS option in the section "PROC SIM2D Statement" on page 9504.

ODS Graph Names

PROC SIM2D assigns a name to each graph it creates by using ODS Graphics. You can use these names to reference the graphs when using ODS Graphics. You must also specify the PLOTS= option indicated in Table 112.5.

Table 112.5 Graphs Produced by PROC SIM2D

ODS Graph Name	Plot Description	Statement	Option
SimulationPlot	Outlines of the observation locations, and either a contour plot of the simulated means and surface of the standard deviation in areal grids, or a band plot of the simulated means or box plot of the simulation distribution in linear grids or individual locations	PROC	PLOTS=SIM
ObservationsPlot	Scatter plot of observed data and colored markers indicating observed values	PROC	PLOTS=OBSERV
Semivariogram	Plots of the semivariogram models used for all simulation tasks	PROC	PLOTS=SEMIVAR

Examples: SIM2D Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 112.1: Simulation and Economic Feasibility

(View the complete code for this example (sim2dex1.sas) in the example repository.)

You can use simulations to investigate the expected behavior of a stochastic process. Simulations with PROC SIM2D can indicate spatial characteristics in your study that might be important for decision making or general assessment. The present example and the one in section "Example 112.3: Risk Analysis with Simulation" on page 9541 are two instances of this type of analysis in different fields.

Continuing with the coal seam thickness example from the section "Getting Started: SIM2D Procedure" on page 9495, this example asks a rather complicated question of economic nature. For illustration, an (approximate) answer is provided, which requires the use of simulation.

Simulating a Subregion for Economic Feasibility

The coal seam must be of a minimum thickness, called a *cutoff value*, for a mining operation to be profitable. Suppose that, for a subregion of the measured area, the cost of mining is higher than in the remaining areas due to the geology of the overburden. This higher cost results in a higher thickness cutoff value for the subregion. Suppose also that it is determined from a detailed cost analysis that at least 60% of the subregion must exceed a seam thickness of 39.7 feet for profitability.

How can you use the SRF model (μ and $C_z(\boldsymbol{s})$) and the measured seam thickness values $Z(\boldsymbol{s}_i), i = 1, \ldots, 75$, to determine, in some approximate way, whether at least 60% of the subregion exceeds this minimum?

Spatial prediction does not appear to be helpful in answering this question. Although it is easy to determine whether a predicted value at a location in the subregion is above the 39.7-feet cutoff value, it is not clear how to incorporate the standard error associated with the predicted value. The standard error is what characterizes the stochastic nature of the prediction (and the underlying SRF). It is clear that it must be included in any realistic approach to the problem.

A conditional simulation, on the other hand, seems to be a natural way of obtaining an approximate answer. By simulating the SRF on a sufficiently fine grid in the subregion, you can determine the proportion of grid points in which the mean value over realizations exceeds the 39.7-feet cutoff and compare it with the 60% value needed for profitability.

It is desirable in any simulation study that the quantity being estimated (in this case, the proportion that exceeds the 39.7-feet cutoff) not depend on the number of simulations performed. For example, suppose that the maximum seam thickness is simulated. It is likely that the maximum value increases as the number of simulations performed increases. Hence, a simulation is not useful for such an estimate. A simulation is useful for determining the *distribution* of the maximum, but there are general theoretical results for such distributions, making such a simulation unnecessary. See Leadbetter, Lindgren, and Rootzen (1983) for details.

In the case of simulating the proportion that exceeds the 39.7-feet cutoff, it is expected that this quantity will settle down to a fixed value as the number of realizations increases. At a fixed grid point, the quantity being compared with the cutoff value is the mean over all simulated realizations; this mean value settles down to a fixed number as the number of realizations increases. In the same manner, the proportion of the grid where the mean values exceed the cutoff also becomes constant. This can be tested using PROC SIM2D.

A crucial, nonprovable assumption in applying SRF theory to the coal seam thickness data is that the values $Z(\boldsymbol{s}_i), i = 1, \ldots, 75$, represent a *single* realization from the set of all possible realizations consistent with the SRF model (μ and $C_z(\boldsymbol{h})$). A conditional simulation repeatedly produces other possible simulated realizations consistent with the model and data. However, the only concern of the mining company is this single unique realization. It is not concerned about similar coal fields to be mined sometime in the future; it might never see another coal field remotely similar to this one, or it might not be in business in the future.

Hence the proportion found by generating repeated simulated realizations must somehow relate back to the unique realization that is the coal field (seam thickness). This is done by interpreting the proportion found from a simulation to the spatial mean proportion for the unique realization. The term "spatial mean" is simply an appropriate integral over the fixed (but unknown) spatial function $z(\boldsymbol{s})$. (The SRF is denoted $Z(\boldsymbol{s})$; a particular realization, a deterministic function of the spatial coordinates, is denoted $z(\boldsymbol{s})$.)

This interpretation requires an ergodic assumption, which is also needed in the original estimation of $C_z(\boldsymbol{s})$. See Cressie (1993, pp. 53–58) for a discussion of ergodicity and Gaussian SRFs.

Implementation Using PROC SIM2D

The subregion to be considered is the southeast corner of the field, which is a square region with a length of 40 distance units (in thousands of feet). First, you input the thickness data as the following DATA step shows:

```
title 'Simulating a Subregion for Economic Feasibility';

data thick;
   input East North Thick @@;
   label Thick='Coal Seam Thickness';
   datalines;
    0.7  59.6  34.1   2.1  82.7  42.2   4.7  75.1  39.5
    4.8  52.8  34.3   5.9  67.1  37.0   6.0  35.7  35.9
    6.4  33.7  36.4   7.0  46.7  34.6   8.2  40.1  35.4
   13.3   0.6  44.7  13.3  68.2  37.8  13.4  31.3  37.8
   17.8   6.9  43.9  20.1  66.3  37.7  22.7  87.6  42.8
   23.0  93.9  43.6  24.3  73.0  39.3  24.8  15.1  42.3
   24.8  26.3  39.7  26.4  58.0  36.9  26.9  65.0  37.8
   27.7  83.3  41.8  27.9  90.8  43.3  29.1  47.9  36.7
   29.5  89.4  43.0  30.1   6.1  43.6  30.8  12.1  42.8
   32.7  40.2  37.5  34.8   8.1  43.3  35.3  32.0  38.8
   37.0  70.3  39.2  38.2  77.9  40.7  38.9  23.3  40.5
   39.4  82.5  41.4  43.0   4.7  43.3  43.7   7.6  43.1
   46.4  84.1  41.5  46.7  10.6  42.6  49.9  22.1  40.7
   51.0  88.8  42.0  52.8  68.9  39.3  52.9  32.7  39.2
   55.5  92.9  42.2  56.0   1.6  42.7  60.6  75.2  40.1
   62.1  26.6  40.1  63.0  12.7  41.8  69.0  75.6  40.1
   70.5  83.7  40.9  70.9  11.0  41.7  71.5  29.5  39.8
   78.1  45.5  38.7  78.2   9.1  41.7  78.4  20.0  40.8
   80.5  55.9  38.7  81.1  51.0  38.6  83.8   7.9  41.6
   84.5  11.0  41.5  85.2  67.3  39.4  85.5  73.0  39.8
```

```
   86.7  70.4  39.6  87.2  55.7  38.8  88.1   0.0  41.6
   88.4  12.1  41.3  88.4  99.6  41.2  88.8  82.9  40.5
   88.9   6.2  41.5  90.6   7.0  41.5  90.7  49.6  38.9
   91.5  55.4  39.0  92.9  46.8  39.1  93.4  70.9  39.7
   55.8  50.5  38.1  96.2  84.3  40.3  98.2  58.2  39.5
;
```

PROC SIM2D is run on the entire data set for conditioning, while the simulation grid covers only this subregion. It is convenient to be able to vary the seed, the grid increment, and the number of simulations performed. The following macro implements the computation of the percent area that exceeds the cutoff value by using the seed, the grid increment, and the number of simulated realizations as macro arguments.

Within the macro, the data set produced by PROC SIM2D is transposed with PROC TRANSPOSE so that each grid location is a separate variable. The MEANS procedure averages then the simulated value at each grid point over all realizations. It is this average that is compared to the cutoff value. The last DATA step does the comparison, uses an appropriate loop to determine the percent of the grid locations that exceed this cutoff value, and writes the results to the listing file in the form of a report. This sequence is implemented with the following statements:

```
/* Construct macro for conditional simulation ------------------*/
%let cc0=7.4599;
%let aa0=30.1111;
%let ngt=1e-8;
%let form=gauss;
%let cut=39.7;

%macro area_sim(seed=,nr=,ginc=);
   %let ngrid=%eval(40/&ginc+1);
   %let tgrid=%eval(&ngrid*&ngrid);

   proc sim2d data=thick outsim=sim1;
      coordinates xc=east yc=north;
      simulate var=thick numreal=&nr seed=&seed
               scale=&cc0 range=&aa0 nugget=&ngt form=&form;
      mean 40.1173;
      grid x=60 to 100 by &ginc
           y= 0 to  40 by &ginc;
   run;

   proc transpose data=sim1 out=sim2 prefix=sims;
      by _iter_;
      var svalue;
   run;

   proc means data=sim2 noprint n mean;
      var sims1-sims&tgrid;
      output out=msim n=numsim mean=ms1-ms&tgrid;
   run;

   data _null_;
      file print;
      array simss ms1-ms&tgrid;
      set msim;
      cflag=0;
```

```
      do ss=1 to &tgrid;
         tempv=simss[ss];
         if simss[ss] > &cut then do;
            cflag + 1;
         end;
      end;

      area_per=100*(cflag/&tgrid);
      put // +5 'Conditional Simulation of Coal Seam'
                ' Thickness for Subregion';
      put / +5 'Subregion is South-East Corner 40 by 40 distance units';
      put / +5 "Seed:&seed" +2 "Grid Increment:&ginc";
      put / +5 "Total Number of Grid Points:&tgrid" +2
               "Number of Simulations:&nr";
      put / +5 "Percent of Subregion Exceeding Cutoff of %left(&cut) ft.:"
                +2 area_per 5.2;
   run;
%mend area_sim;
```

In the following statement, you invoke the macro three times. Each time, the macro is invoked with a different seed and combination of the grid increment and number of simulations. The macro is first invoked with a relatively coarse grid (grid increment of 10 distance units) and a small number of realizations (5). The output of this conditional simulation is shown in Output 112.1.1.

```
/* Execute macro for coarse grid -------------------------------*/
%area_sim(seed=12345,nr=5,ginc=10);
```

Output 112.1.1 Conditional Simulation of Coal Seam Thickness on a Coarse Grid

Simulating a Subregion for Economic Feasibility

```
Conditional Simulation of Coal Seam Thickness for Subregion

Subregion is South-East Corner 40 by 40 distance units

Seed:12345  Grid Increment:10

Total Number of Grid Points:25  Number of Simulations:5

Percent of Subregion Exceeding Cutoff of 39.7 ft.:  76.00
```

The next invocation, in the following statement, uses a finer grid and 50 realizations. The output of the second conditional simulation is shown in Output 112.1.2.

```
/* Execute macro for fine grid and fewer simulations -----------*/
%area_sim(seed=54321,nr=50,ginc=1);
```

Output 112.1.2 Conditional Simulation of Coal Seam Thickness on a Fine Grid

Simulating a Subregion for Economic Feasibility

```
Conditional Simulation of Coal Seam Thickness for Subregion

Subregion is South-East Corner 40 by 40 distance units

Seed:54321  Grid Increment:1

Total Number of Grid Points:1681  Number of Simulations:50

Percent of Subregion Exceeding Cutoff of 39.7 ft.:  76.09
```

The final invocation, in the following statement, uses the same grid increment and 500 realizations. The output of this conditional simulation is shown in Output 112.1.3.

```
/* Execute macro for fine grid and more simulations ------------*/
%area_sim(seed=655311,nr=500,ginc=1);
```

Output 112.1.3 Conditional Simulation of Coal Seam Thickness on a Fine Grid

Simulating a Subregion for Economic Feasibility

```
Conditional Simulation of Coal Seam Thickness for Subregion

Subregion is South-East Corner 40 by 40 distance units

Seed:655311  Grid Increment:1

Total Number of Grid Points:1681  Number of Simulations:500

Percent of Subregion Exceeding Cutoff of 39.7 ft.:  76.09
```

The results from the preceding simulations indicate that about 76% of the subregion exceeds the cutoff value.

NOTE: The number of grid points in the simulation increases with the square of the decrease in the grid increment, leading to long CPU processing times. Increasing the number of realizations results in a linear increase in processing times. Hence, using as coarse a grid as possible allows for more realizations and experimentation with different seeds.

Example 112.2: Variability at Selected Locations

(View the complete code for this example (sim2dex2.sas) in the example repository.)

This example exhibits a more detailed investigation of the variation of simulated Thick variable values. You use the same thick data set from the section "Getting Started: SIM2D Procedure" on page 9495, and you are interested in the simulated values statistics at two selected grid points.

Specifically, you perform a simulation asking for 5,000 realizations (iterations) at two points of the region defined in the section "Preliminary Spatial Data Analysis" on page 9495. These are the extreme southwest point and a point toward the northeast corner of the region. Since you want to avoid performing the simulation across the whole region, you need to produce a GDATA= data set to specify the coordinates of the selected points. These steps are implemented using the following DATA step and statements:

```
title 'Investigation of Random Field Variability';

data thick;
   input East North Thick @@;
   label Thick='Coal Seam Thickness';
   datalines;
    0.7  59.6  34.1   2.1  82.7  42.2   4.7  75.1  39.5
    4.8  52.8  34.3   5.9  67.1  37.0   6.0  35.7  35.9
    6.4  33.7  36.4   7.0  46.7  34.6   8.2  40.1  35.4
   13.3   0.6  44.7  13.3  68.2  37.8  13.4  31.3  37.8
   17.8   6.9  43.9  20.1  66.3  37.7  22.7  87.6  42.8
   23.0  93.9  43.6  24.3  73.0  39.3  24.8  15.1  42.3
   24.8  26.3  39.7  26.4  58.0  36.9  26.9  65.0  37.8
   27.7  83.3  41.8  27.9  90.8  43.3  29.1  47.9  36.7
   29.5  89.4  43.0  30.1   6.1  43.6  30.8  12.1  42.8
   32.7  40.2  37.5  34.8   8.1  43.3  35.3  32.0  38.8
   37.0  70.3  39.2  38.2  77.9  40.7  38.9  23.3  40.5
   39.4  82.5  41.4  43.0   4.7  43.3  43.7   7.6  43.1
   46.4  84.1  41.5  46.7  10.6  42.6  49.9  22.1  40.7
   51.0  88.8  42.0  52.8  68.9  39.3  52.9  32.7  39.2
   55.5  92.9  42.2  56.0   1.6  42.7  60.6  75.2  40.1
   62.1  26.6  40.1  63.0  12.7  41.8  69.0  75.6  40.1
   70.5  83.7  40.9  70.9  11.0  41.7  71.5  29.5  39.8
   78.1  45.5  38.7  78.2   9.1  41.7  78.4  20.0  40.8
   80.5  55.9  38.7  81.1  51.0  38.6  83.8   7.9  41.6
   84.5  11.0  41.5  85.2  67.3  39.4  85.5  73.0  39.8
   86.7  70.4  39.6  87.2  55.7  38.8  88.1   0.0  41.6
   88.4  12.1  41.3  88.4  99.6  41.2  88.8  82.9  40.5
   88.9   6.2  41.5  90.6   7.0  41.5  90.7  49.6  38.9
   91.5  55.4  39.0  92.9  46.8  39.1  93.4  70.9  39.7
   55.8  50.5  38.1  96.2  84.3  40.3  98.2  58.2  39.5
;

data grid;
   input xc yc;
   datalines;
   0   0
   75  75
;
```

Then, you run PROC SIM2D with the same parameters and characteristics as those shown in the section "Preliminary Spatial Data Analysis" on page 9495. This time, however, you ask for simulated values only at the two locations you specified in the previous DATA step. The following statements execute the requested simulation:

```
proc sim2d data=thick outsim=sim1;
   coordinates xc=East yc=North;
   simulate var=Thick numreal=5000 seed=79931
            scale=7.4599 range=30.1111 form=gauss;
   mean 40.1173;
   grid gdata=grid xc=xc yc=yc;
run;
```

After the simulation is performed, summary statistics are computed for each of the specified grid points. In particular, you use PROC UNIVARIATE and a BY statement to request the quantiles and the extreme observations at these locations, as the following statements show:

```
proc sort data=sim1;
   by gxc gyc;
run;

proc univariate data=sim1;
   var svalue;
   by gxc gyc;
   ods select Quantiles ExtremeObs;
   title 'Simulation Statistics at Selected Grid Points';
run;
```

The summary statistics for the first grid point (East=0, North=0) are presented in Output 112.2.1.

Output 112.2.1 Simulation Statistics at Grid Point (East=0, North=0)

Simulation Statistics at Selected Grid Points

The UNIVARIATE Procedure
Variable: SVALUE (Simulated Value at Grid Point)

X-coordinate of the grid point=0 Y-coordinate of the grid point=0

Quantiles (Definition 5)	
Level	**Quantile**
100% Max	42.4207
99%	41.8960
95%	41.5315
90%	41.3419
75% Q3	41.0324
50% Median	40.6701
25% Q1	40.2871
10%	39.9904
5%	39.7825
1%	39.4181
0% Min	38.6864

Output 112.2.1 *continued*

X-coordinate of the grid point=0 Y-coordinate of the grid point=0

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
38.6864	2691	42.2952	1149
38.8611	1817	42.3114	3612
38.9013	3026	42.3305	3757
38.9467	2275	42.4177	135
38.9823	3100	42.4207	4536

Finally, Output 112.2.2 displays the summary statistics for the second grid point (East=75, North=75).

Output 112.2.2 Simulation Statistics at Grid Point (East=75, North=75)

Simulation Statistics at Selected Grid Points

The UNIVARIATE Procedure
Variable: SVALUE (Simulated Value at Grid Point)

X-coordinate of the grid point=75 Y-coordinate of the grid point=75

Quantiles (Definition 5)	
Level	Quantile
100% Max	40.1171
99%	40.1147
95%	40.1131
90%	40.1122
75% Q3	40.1108
50% Median	40.1092
25% Q1	40.1075
10%	40.1062
5%	40.1053
1%	40.1035
0% Min	40.1001

X-coordinate of the grid point=75 Y-coordinate of the grid point=75

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
40.1001	7176	40.1167	8980
40.1007	6262	40.1167	9272
40.1011	7383	40.1169	5676
40.1016	7156	40.1170	6514
40.1017	5643	40.1171	5329

For each simulation location, a single realization might result in values that differ significantly from the random field mean at that location. However, the averages of progressively larger numbers of realizations tend to shorten this gap and reduce the simulation variability, as exhibited in the results for the two example locations in Output 112.2.1 and Output 112.2.2. At the limit of an infinite number of realizations, the

simulation mean recovers the mean and covariance structure of the random field.

Example 112.3: Risk Analysis with Simulation

(View the complete code for this example (sim2dex3.sas) in the example repository.)

This example is in the field of environmental risk assessment. A square region of size 500 km × 500 km has been sampled for arsenic in drinking water. The logAsData data set consists of 138 simulated arsenic logarithm concentration observations represented by the logAs variable. Section "Example 131.1: Aspects of Semivariogram Model Fitting" on page 11047 in Chapter 131, "The VARIOGRAM Procedure," treats these observations as actual data in order to determine the spatial continuity structure for illustration in the examples.

A preliminary analysis indicates that the population exposure to the pollutant is currently at a relatively low level, as shown in the section "Example 74.1: Spatial Prediction of Pollutant Concentration" on page 5602 in Chapter 74, "The KRIGE2D Procedure," procedure. In particular, spatial prediction with kriging suggests that less than 1% of the region is affected by arsenic concentration in water that exceeds the World Health Organization (WHO) standard of 10 μg/lt.

You want to simulate the random field of the arsenic logarithm concentration so that you can gain insight into the characteristics of this field. Your objective is to assess whether the occurrence of above-threshold arsenic concentrations is a localized phenomenon, and whether you might expect more such occurrences across the region. For the simulation you need the outcome of the spatial continuity analysis in section "Example 131.1: Aspects of Semivariogram Model Fitting" on page 11047 in Chapter 131, "The VARIOGRAM Procedure." These results are the fitted semivariance models that are stored in the SemivAsStore item store by PROC VARIOGRAM.

Based on the discussion in the section "Example 112.1: Simulation and Economic Feasibility" on page 9533, you simulate the arsenic logarithm concentration on the premise of the ergodicity assumption. In brief, this assumption relates the mean and covariances of each individual realization to the corresponding values of the random field; see also the section "Ergodicity" on page 11015 in Chapter 131, "The VARIOGRAM Procedure." In the present case you are interested in the average of a large size of realizations, rather than individual ones. A single realization could suggest possible locations where the WHO regulatory standard might be violated. The average of multiple realizations has a smoothing effect on individual excessive-concentration episodes in single realizations. The smoothing enables you to see general characteristics of the arsenic concentration field behavior on the basis of its spatial correlation description.

For illustration, assume a rectangular grid of nodes with an equal spacing of 10 km between neighboring nodes in the north and east directions. Then, simulated values are produced at a total of $51 \times 51 = 2601$ locations.

You begin by reading the logAsData data set with the following DATA step:

```
title 'Risk Assessment with Simulation';

data logAsData;
   input East North logAs @@;
   label logAs='log(As) Concentration';
   datalines;
   193.0 296.6 -0.68153   232.6 479.1  0.96279   268.7 312.5 -1.02908
    43.6   4.9  0.65010   152.6  54.9  1.87076   449.1 395.8  0.95932
```

```
   310.9 493.6 -1.66208   287.8 164.9 -0.01779   330.0   8.0  2.06837
   225.7 241.7  0.15899   452.3  83.4 -1.21217   156.5 462.5 -0.89031
    11.5  84.4 -0.24496   144.4 335.7  0.11950   149.0 431.8 -0.57251
   234.3 123.2 -1.33642    37.8 197.8 -0.27624   183.1 173.9 -2.14558
   149.3 426.7 -1.06506   434.4  67.5 -1.04657   439.6 237.0 -0.09074
    36.4 175.2 -1.21211   370.6 244.0  3.28091   452.0  96.5 -0.77081
   247.0  86.8  0.04720   413.6 373.2  1.78235   253.5 291.7  0.56132
   129.7 111.9  1.34000   352.7  42.1  0.23621   279.3  82.7  2.12350
   382.6 290.7  0.86756   188.2 222.8 -1.23308   382.8 154.5 -0.94094
   304.4 309.2 -1.95158   337.5 387.2 -1.31294   490.7 189.8  0.40206
   159.0 100.1 -0.22272   245.5 329.2 -0.26082   372.1 379.5 -1.89078
   417.8  84.1 -1.25176   173.9 407.6 -0.24240   121.5 107.7  1.54509
   453.5 313.6  0.65895   143.5 346.7 -0.87196   157.4 125.5 -1.96165
   371.8 353.2 -0.59464   358.9 338.2 -1.07133     8.6 437.8  1.44203
   395.9 394.2 -0.24144   149.5  58.9  1.17459   453.5 420.6 -0.63951
   182.3  85.0  1.00005    21.0 290.1  0.31016    11.1 352.2 -0.88418
   131.2 238.4 -0.57184   104.9   6.3  1.12054   247.3 256.0  0.14019
   428.4 383.7  0.92448   327.8 481.1 -2.72543   199.2  92.8 -0.05717
   453.9 230.1  0.16571   205.0 250.6  0.07581   459.5 271.6  0.93700
   229.5 262.8  1.83590   370.4 228.6  2.96611   330.2 281.9  1.79723
   354.8 388.3 -3.18262   406.2 222.7  2.41594   254.4 393.1  2.03221
    96.7  85.2 -0.47156   407.2 256.8  0.66747   498.5 273.8  1.03041
   417.2 471.4 -1.42766   368.8 424.3 -0.70506   303.0  59.1  1.43070
   403.1 264.1  1.64554    21.2 360.8  0.67094   148.2  78.1  2.15323
   305.5 310.7 -1.47985   228.5 180.3 -0.68386   161.1 143.3  1.07901
    70.5 155.1  0.54652   363.1 282.6 -0.43051    86.0 472.5 -1.18855
   175.9 105.3 -2.08112    96.8 426.3  1.56592   475.1 453.1 -1.53776
   125.7 485.4  1.40054   277.9 201.6 -0.54565   406.2 125.0 -1.38657
    60.0 275.5 -0.59966   431.3 494.6 -0.36860   399.9 399.0 -0.77265
    28.8 311.1  0.91693   166.1 348.2 -0.49056   266.6  83.5  0.67277
    54.7 356.3  0.49596   433.5 460.3 -1.61309   201.7 167.6 -1.40678
   158.1 203.6 -1.32499    67.6 230.4  1.14672    81.9 250.0  0.63378
   372.0  50.7  0.72445    26.4 264.6  1.00862   300.1  91.7 -0.74089
   303.0 447.4  1.74589   108.4 386.2  1.12847    55.6 191.7  0.95175
    36.3 273.2  1.78880    94.5 298.3 -2.43320   366.1 187.3 -0.80526
   130.7 389.2 -0.31513    37.2 324.2  0.24489   295.5 211.8  0.41899
    58.6 206.2  0.18495   346.3 142.8 -0.92038   484.2 215.9  0.08012
   451.4 415.7  0.02773    58.9  86.5  0.17652   212.6 363.9  0.17215
   378.7 407.6  0.51516   265.9 305.0 -0.30718   123.2 314.8 -0.90591
    26.9 471.7  1.70285    16.5   7.1  0.51736   255.1 472.6  2.02381
   111.5 148.4 -0.09658   440.4 375.0  1.23285   406.4  19.5  1.01181
   321.2  65.8 -0.02095   466.4 357.1 -0.49272     2.0 484.6  0.50994
   200.9 205.1  0.43543    30.3 337.0  1.60882   297.0  12.7  1.79824
   158.2 450.7  0.05295   122.8 105.3  1.53936   417.8 329.7 -2.08124
;
```

For this simulation you use the spatial correlation information that is saved in the SemivAsStore item store in the section "Example 131.1: Aspects of Semivariogram Model Fitting" on page 11047 in Chapter 131, "The VARIOGRAM Procedure," with the following statements:

```
ods graphics on;
```

```
proc variogram data=logAsData plots=none;
   store out=SemivAsStore / label='LogAs Concentration Models';
   compute lagd=5 maxlag=40;
   coord xc=East yc=North;
   model form=auto(mlist=(exp,gau,mat) nest=1 to 2);
   var logAs;
run;
```

In PROC SIM2D you specify the container item store with the IN= option of the RESTORE statement. You request correlation input from an item store by specifying the STORESELECT option in the SIMULATE statement. You request 5,000 realizations for this simulation by specifying the number in the NUMREAL= option of the SIMULATE statement.

Assume that you first want to review the saved models in the item store. Use the INFO option of the RESTORE statement to produce a table with information about the top-ranking fitted model in the item store. Use the DET and ONLY suboptions of the INFO option to request details about all fitted models included in the item store. The ONLY suboption suppresses the simulation tasks and produces only the tables about the item store. You specify the following statements:

```
proc sim2d data=logAsData outsim=Outsim plots=none;
   restore in=SemivAsStore / info(det only);
   coordinates xc=East yc=North;
   simulate var=logAs numreal=5000 storeselect seed=39841;
   grid x=0 to 500 by 10 y=0 to 500 by 10;
run;
```

PROC SIM2D produces a table with general information about the input item store identity, as shown in Output 112.3.1.

Output 112.3.1 PROC SIM2D and Input Item Store General Information

Risk Assessment with Simulation

The SIM2D Procedure

Correlation Model Item Store Information	
Input Item Store	WORK.SEMIVASSTORE
Item Store Label	LogAs Concentration Models
Data Set Created From	WORK.LOGASDATA
By-group Information	No By-groups Present
Created By	PROC VARIOGRAM
Date Created	22APR20:01:49:43

Output 112.3.2 displays the item store variables, in addition to the mean and standard deviation of their data set of origin. In this case, the logAs values come from the logAsData data set. By default, the SIM2D procedure uses the variable mean in the item store for the simulation, unless you explicitly specify the MEAN statement.

Output 112.3.2 Variables in the Input Item Store

Item Store Variables		
Variable	Mean	Std Deviation
logAs	0.084309	1.527707

The models in the SemivAsStore item store that have been fitted to the arsenic logarithm logAs empirical semivariance are shown in Output 112.3.3. The default item store model selection is the model on top of the list in Output 112.3.3.

Output 112.3.3 Angle and Models Information in the Input Item Store

Item Store Models For logAs	
Class	Model
1	Gau-Gau
	Gau-Mat
	Mat-Mat
2	Exp-Gau
3	Exp-Mat
4	Mat
5	Gau
6	Exp
	Exp-Exp
	Mat-Exp
	Gau-Exp

You run again the SIM2D procedure without the INFO option in the RESTORE statement. This action prompts PROC SIM2D to run the simulation tasks that you specify with the SIMULATE statement. You specify the STORESELECT option in the SIMULATE statement without any suboptions, so that the simulation uses the default model selection in the SemivAsStore item store. You save the simulation output in the Outsim output data set. You also specify the SIM and the SEMIVAR suboptions in the PLOTS option of the PROC SIM2D statement to obtain output plots. You use the following statements:

```
proc sim2d data=logAsData outsim=Outsim plots=(sim semivar);
   restore in=SemivAsStore;
   coordinates xc=East yc=North;
   simulate var=logAs numreal=5000 storeselect seed=89702;
   grid x=0 to 500 by 10 y=0 to 500 by 10;
run;
```

NOTE: This step can take several minutes to run, and it produces a data set with over 10 million observations. When you run these statements, PROC SIM2D again produces a table about the input item store identity. This output is followed by a number of observations table and information about the simulation task, as shown in Output 112.3.4.

Output 112.3.4 Number of Observations and Simulation Information Tables

Risk Assessment with Simulation

The SIM2D Procedure

Simulation: SIM1, Dependent Variable: logAs

Number of Observations Read	138
Number of Observations Used	138

Simulation Information	
Simulation Grid Points	2601
Type	Conditional
Number of Realizations	5000

The SIM2D procedure uses the selected fitted Gaussian-Gaussian model in the SemivAsStore item store. Output 112.3.5 shows the saved parameter values of the model that are used in the simulation.

Output 112.3.5 Information about the Gaussian-Gaussian Model

Covariance Model Information	
Nested Structure 1 Type	Gaussian
Nested Structure 1 Sill	0.3276621
Nested Structure 1 Range	62.313012
Nested Structure 1 Effective Range	107.9293
Nested Structure 2 Type	Gaussian
Nested Structure 2 Sill	1.2615484
Nested Structure 2 Range	21.459555
Nested Structure 2 Effective Range	37.16904
Nugget Effect	0.083075

Output 112.3.6 shows a plot of the correlation model used in the simulation. Its parameters come from the stored information in the SemivAsStore item store.

Output 112.3.6 Gaussian-Gaussian Semivariogram Model Used in Simulation

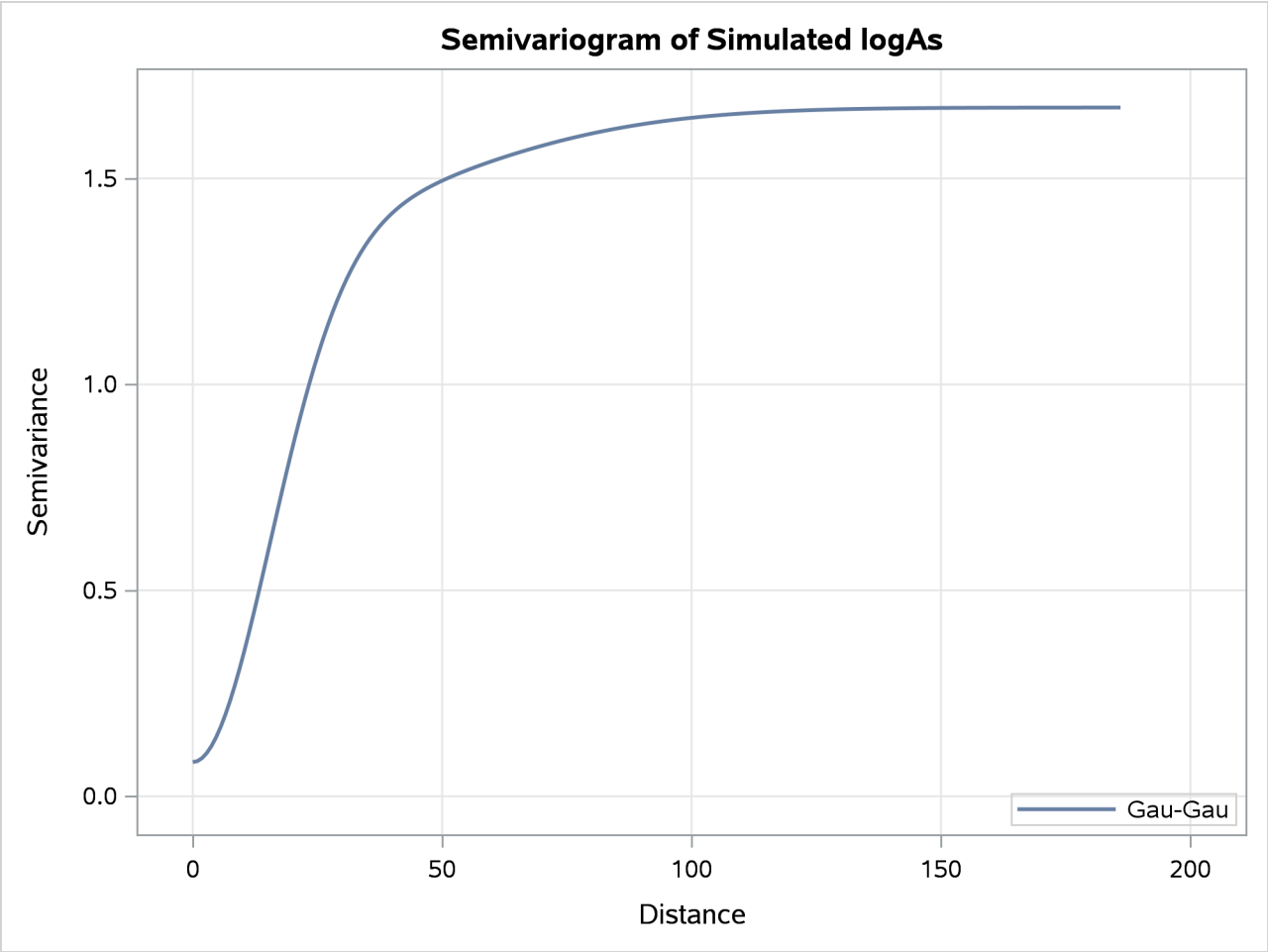

The PLOTS=SIM option produces contours of the mean value of the 5,000 realizations for each one of the output grid locations, and a surface of the associated standard deviation. The resulting plot is shown in Output 112.3.7. The mean value contours exhibit reduced variation compared to a SIM plot of one single realization. In fact, Output 112.3.7 is very similar to the prediction plot of the same Gaussian model in the section "Example 74.1: Spatial Prediction of Pollutant Concentration" on page 5602 in Chapter 74, "The KRIGE2D Procedure."

Clearly, there appears to be a small area of increased arsenic concentration values, which is located in the central-eastern part of the domain. The WHO threshold of 10 μg/lt for the maximum allowed arsenic concentration in water translates into about 2.3 in the log scale. The indicated area is the only one within the region where the simulated means exceed the value 3. In addition, there appear to be two smaller areas of simulated means values at or above 2 in the southern part of the region.

Output 112.3.7 Simulated Arsenic Logarithm Values with Gaussian-Gaussian Covariance

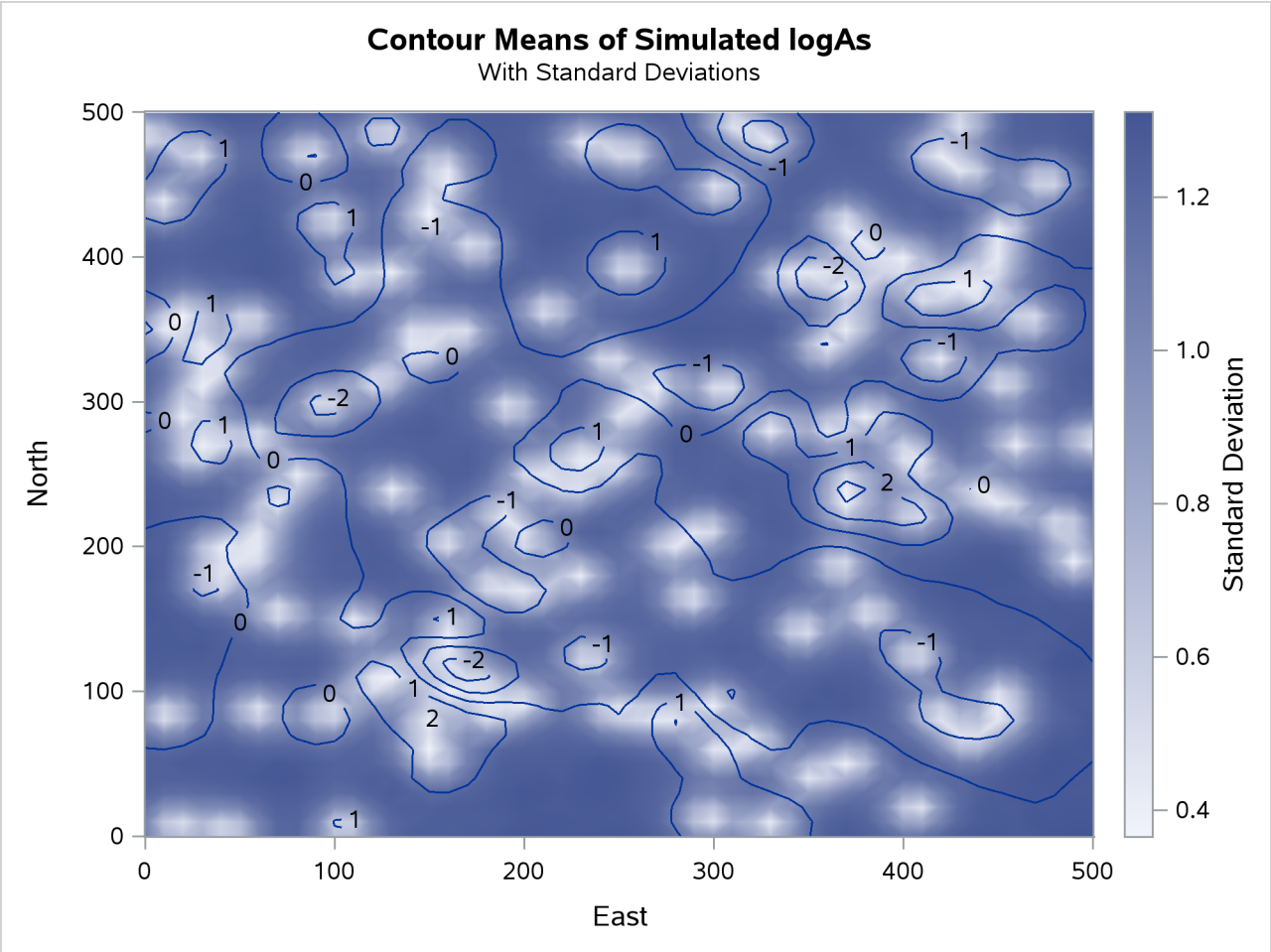

The simulation plot indicates that arsenic concentration values in excess of the WHO health standard is a rather localized phenomenon. If it were not so, then the plot would suggest that increased arsenic concentrations extend across larger areas. In turn, this means that individual realizations would tend to produce systematically higher arsenic concentrations at different neighboring locations, and their average would appear as higher logAs values in the SIM plot. Instead, you conclude that the Gaussian-Gaussian correlation that describes the spatial continuity for the arsenic logarithm observations leads to only localized occurrence of the WHO standard violation.

Now you want to find out whether additional localized occurrences might take place on the basis of the Gaussian-Gaussian spatial continuity behavior. The distribution of the simulation standard deviation values in Output 112.3.7 offers important feedback about this issue. Observe the areas in the plot where the means gradient indicates increasing values. The means contours create several pools where the values increase to simulated means higher than 1.

With the clear exceptions of the isolated area in the central-eastern part of the domain and the pool in the south-west that exhibit logAs values above 2, the simulation standard deviation seems to be around 1 or less in almost all other pools with logAs values above 1. Due to those relatively low standard deviations, this visual inspection suggests that even in individual realizations you might rarely expect the arsenic logarithm to exceed the threshold value of around 2.3.

However, in the few pools of logAs values above 1, there can be patches of increased standard deviation. In these cases, you suspect that arsenic concentration could exceed occasionally the WHO regulatory standard, even if the plot of the simulated means does not explicitly portray so.

Based on these remarks, you want to quantify the percentage of the region that could be affected by excessive arsenic concentration. You begin with a DATA step that takes as input the simulation Outsim output data set. The DATA step marks the simulated arsenic logarithm means values that are in excess of the WHO concentration threshold of 10 μg/lt, and saves the outcome into an indicator variable OverLimit. At the same time, the arsenic logarithm values are transformed back into arsenic concentration values so that they can be compared to the threshold value. The simulated means in the Outsim data set are stored in the svalue variable. You use the following statements:

```
data AsOverLimit;
   set Outsim;
   OverLimit = (exp(svalue) > 10) * 100;
run;
```

Then, you use the MEANS procedure to express the selected nodes population (where the WHO standard violation occurs) as a percentage of the entire domain area. You invoke PROC MEANS twice. The first time, you average over each realization in the PROC SIM2D output. For that purpose, you use the variable _ITER_ in the simulation sim1 output data set as a BY variable in PROC MEANS. You save the iteration-averaged indicator values in the PctOverLimit variable. The second time, PROC MEANS averages the PctOverLimit variable to obtain the percentage you want. You also specify the P5 and P95 options in the PROC MEANS statement to request the lower and upper confidence limits, respectively, in this computation. The interval between those limits expresses the 90% interval for the true area percentage based on the stochastic simulation. You use the following statements:

```
proc means data=AsOverLimit noprint;
   by _ITER_;
   var OverLimit;
   output out=OverLimitData mean=PctOverLimit;
run;
proc means data=OverLimitData mean p5 p95;
   var PctOverLimit;
   label PctOverLimit="Percent above threshold";
run;
```

The result is shown in Output 112.3.8. PROC MEANS accounts for all individual occurrences throughout the simulation where the WHO arsenic concentration threshold is exceeded. This happens at about 3.9% of the total area in 5,000 realizations. Compare this value to the less than 1% percentage in the PROC KRIGE2D prediction, as reported in the beginning of this section. On one hand, the prediction outcome tells you what goes on currently across the region within a degree of certainty given by the prediction error. On the other hand, the simulation provides you with an indicator that under the given spatial continuity estimate a relatively larger area is in potential danger of being affected by above-threshold arsenic concentration.

In fact, the 5% and 95% confidence limits in Output 112.3.8 show the area percentage limits within which you can expect excessive arsenic concentration. Based on the stochastic simulation, at the 90% confidence level the arsenic concentration in drinking water is expected to violate the WHO regulatory standard in anywhere from about 2.6% to 5.4% of the study region.

Output 112.3.8 Violation of Arsenic Concentration Threshold Using Gaussian-Gaussian Model

Risk Assessment with Simulation

The MEANS Procedure

Analysis Variable : PctOverLimit Percent above threshold		
Mean	5th Pctl	95th Pctl
3.9308804	2.6143791	5.4209919

You also want to compute the probability that individual areas in the region are expected to violate the WHO arsenic concentration standard. Start by identifying again the simulated arsenic logarithm means values in excess of the WHO concentration threshold of 10 μg/lt. The following DATA step saves the outcome into the variable LocalOverLimit:

```
data LocalAsOverLimit;
   set Outsim;
   LocalOverLimit = (exp(svalue) > 10);
run;
```

Then you use the SORT procedure to sort the information based on location coordinates. This intermediate step is necessary so that you can use the MEANS procedure with the LocalAsOverLimit data set. In the following statements, PROC MEANS computes the expected violations of the WHO standard for each location in the region across all realizations of the simulation:

```
proc sort data=LocalAsOverLimit;
   by gxc gyc;
run;
proc means data=LocalAsOverLimit noprint;
   by gxc gyc;
   var LocalOverLimit;
   output out=OverLimLoci mean=ProbOverLimit;
run;
```

The output is the probability that the WHO standard is violated at each one of the simulation locations across the region. You create a plot of this probability with the help of the TEMPLATE and the SGRENDER procedures. You use the following statements:

```
proc template;
   define statgraph surfacePlot;
      dynamic _VARX _VARY _VAR _TITLE _LEGENDLABEL;
      BeginGraph;
      entrytitle _TITLE;
      layout overlay /
         xaxisopts = (offsetmax=0)
         yaxisopts = (offsetmax=0);
         contourplotparm x=_VARX y=_VARY z=_VAR /
                         nhint=10 name='probplot';
         continuouslegend 'probplot' / title=_LEGENDLABEL;
      endlayout;
      EndGraph;
   end;
```

```
run;

proc sgrender data=OverLimLoci template=surfacePlot;
   dynamic _VARX ='gxc'
           _VARY ='gyc'
           _VAR  ='ProbOverLimit'
           _TITLE='Arsenic WHO Standard Violation'
           _LEGENDLABEL='Violation Probability';
   label gyc='North' gxc='East';
run;

ods graphics off;
```

Output 112.3.9 shows the map of probability that the arsenic concentration WHO standard is violated in the region for the selected spatial correlation model of the pollutant. Based on the preceding analysis, the probability is very close to 1 in the area where the simulation mean values are above the standard limit. A few more areas that were indicated earlier in the analysis suggest a violation probability between 20% and 40%. The remaining areas in the region exhibit very low probability, which is notably nonzero at a few scattered locations. These findings suggest that throughout the simulation there have been realizations where the arsenic WHO standard was exceeded at locations whose simulated mean is well below that standard.

From the environmental risk assessment perspective, the preceding analysis can trigger a more detailed investigation into areas in the region where the health standard might be violated. Although the original observations in the logAsData data set indicate no existing problem in some of these areas, the present example illustrates that spatial analysis and simulation can raise flags of caution. This type of analysis can help to focus scientific, management, and policy efforts on these particular areas of the region and to monitor closely the pollutant concentration for potential health risks.

Output 112.3.9 Map of Violation of the Arsenic WHO Standard Probability

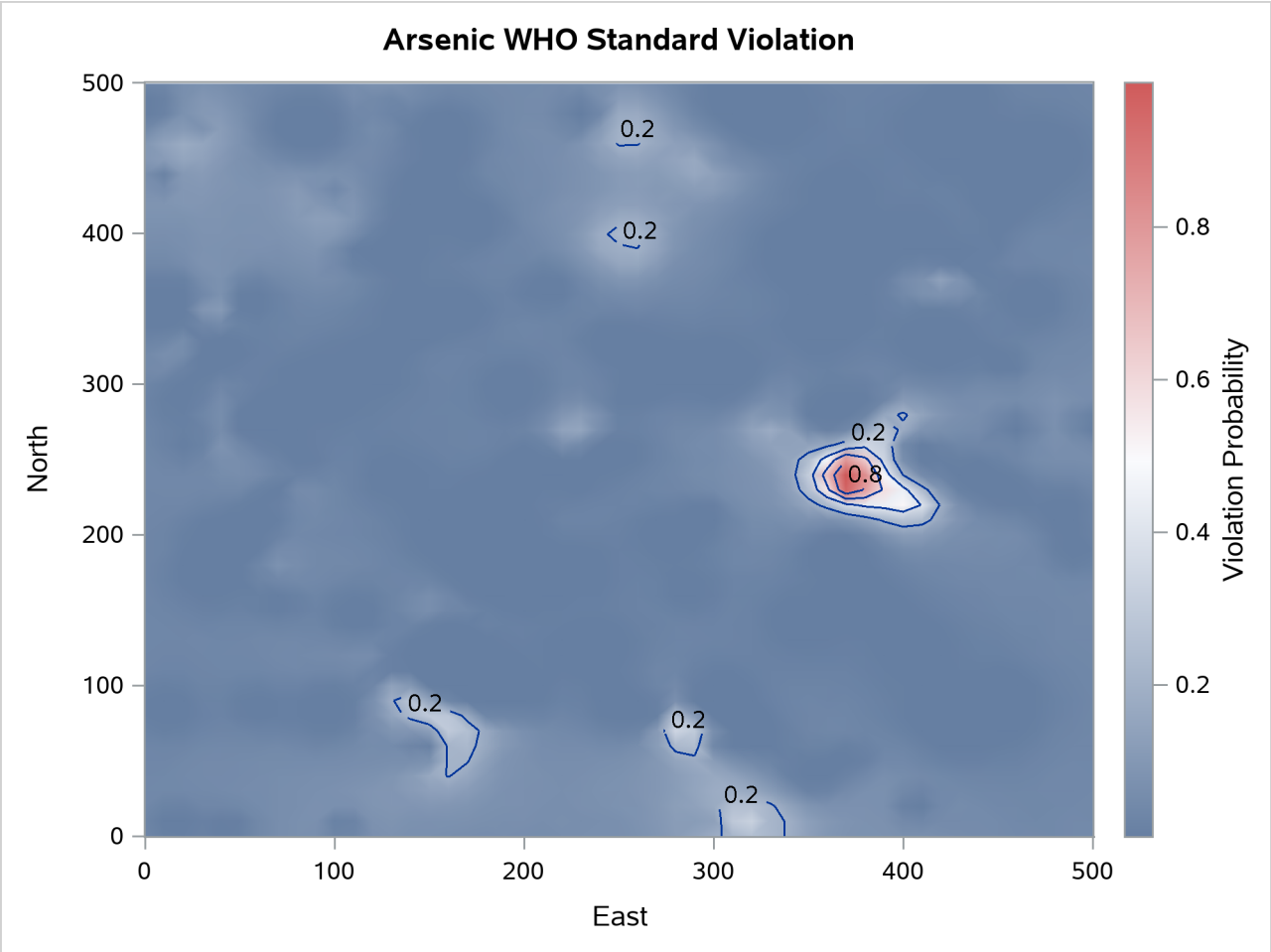

References

Christakos, G. (1992). *Random Field Models in Earth Sciences*. New York: Academic Press.

Cressie, N. (1993). *Statistics for Spatial Data*. Rev. ed. New York: John Wiley & Sons.

Deutsch, C. V., and Journel, A. G. (1992). *GSLIB: Geostatistical Software Library and User's Guide*. New York: Oxford University Press.

Knuth, D. E. (1981). *Seminumerical Algorithms*. Vol. 2 of The Art of Computer Programming. 2nd ed. Reading, MA: Addison-Wesley.

Leadbetter, M. R., Lindgren, G., and Rootzen, H. (1983). *Extremes and Related Properties of Random Sequences and Processes*. New York: Springer-Verlag.

Ralston, A., and Rabinowitz, P. (1978). *A First Course in Numerical Analysis*. New York: McGraw-Hill.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Chapter 113
The SIMNORMAL Procedure

Contents

Overview: SIMNORMAL Procedure **9553**
Getting Started: SIMNORMAL Procedure **9554**
Syntax: SIMNORMAL Procedure **9555**
PROC SIMNORMAL Statement 9555
BY Statement 9557
CONDITION Statement 9557
VAR Statement 9558
OUT= Output Data Set 9558
Details: SIMNORMAL Procedure **9558**
Introduction 9558
Unconditional Simulation 9559
Conditional Simulation 9560
Example: SIMNORMAL Procedure **9561**
Example 113.1: Simulating Correlated Normal Variables 9561
References **9568**

Overview: SIMNORMAL Procedure

The SIMNORMAL procedure can perform conditional and unconditional simulation for a set of correlated normal or Gaussian random variables.

The means, variances, and covariances (or correlations) are read from an input TYPE=CORR or TYPE=COV data set. This data set is typically produced by the CORR procedure. Conditional simulations are performed by appending a special observation, identified by the value of 'COND' for the _TYPE_ variable, which contains the conditioning value.

The output data set from PROC SIMNORMAL contains simulated values for each of the analysis variables. Optionally, the output data set also contains the seed stream and the values of the conditioning variables. PROC SIMNORMAL produces no printed output.

Getting Started: SIMNORMAL Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/simngs.sas.)

The following example illustrates the use of PROC SIMNORMAL to generate two normal random variates that have specified means and covariance.

In this example, the means and covariances are given; these might have come from previous experiments, observational studies, or other considerations.

First you create a _TYPE_=COV data set as the input data set, and then you run PROC SIMNORMAL with NUMREAL=5000, creating a sample that contains 5,000 observations. The simple statistics of this sample are checked using PROC CORR. The results are shown in Figure 113.1.

```
data scov(type=COV) ;
   input _TYPE_ $ 1-4 _NAME_ $ 9-10 S1 S2 ;
   datalines ;
COV     S1      1.915  0.3873
COV     S2      0.3873 4.321
MEAN            1.305  2.003
;

proc simnormal data=scov outsim=ssim
               numreal = 5000
               seed = 54321 ;
   var s1 s2 ;
run;

proc corr data=ssim cov ;
   var s1 s2 ;
   title "Statistics for PROC SIMNORMAL Sample Using NUMREAL=5000" ;
run;
```

Figure 113.1 Statistics for PROC SIMNORMAL Sample Using NUMREAL=5000

Statistics for PROC SIMNORMAL Sample Using NUMREAL=5000

The CORR Procedure

2 Variables:	S1	S2

Covariance Matrix, DF = 4999		
	S1	S2
S1	1.895805499	0.424837163
S2	0.424837163	4.132974275

Simple Statistics						
Variable	N	Mean	Std Dev	Sum	Minimum	Maximum
S1	5000	1.30254	1.37688	6513	-3.90682	6.49864
S2	5000	1.98790	2.03297	9940	-5.69812	9.42833

Syntax: SIMNORMAL Procedure

The following statements are available in the SIMNORMAL procedure:

PROC SIMNORMAL DATA=*SAS-data-set* < *options* > **;**
  VAR *variables* **;**
  BY *variables* **;**
  CONDITION *variables* **;**

Both the PROC SIMNORMAL and VAR statements are required. The following sections describe the PROC SIMNORMAL statement and then describe the other statements in alphabetical order.

PROC SIMNORMAL Statement

PROC SIMNORMAL DATA=*SAS-data-set* < *options* > **;**

The PROC SIMNORMAL statement invokes the SIMNORMAL procedure. Table 113.1 summarizes the options available in the PROC SIMNORMAL statement.

Table 113.1 Summary of PROC SIMNORMAL Statement Options

Option	Description
Specify Input and Output Data Sets	
DATA=	Specifies input data set (TYPE=CORR, COV, and so on)
OUT=	Creates output data set that contains simulated values
Seed Values	
SEED=	Specifies seed value (integer)
SEEDBY	Requests reinitialization of seed for each BY group
Control Contents of OUT= Data Set	
OUTSEED	Requests seed values written to OUT= data set
OUTCOND	Requests conditioning variable values written to OUT=data set
Control Number of Simulated Values	
NUMREAL=	Specifies the number of realizations for each BY group written to the OUT= data set
Singularity Criteria	
SINGULAR1=	Sets the singularity criterion for Cholesky decomposition
SINGULAR2=	Sets the singularity criterion for covariance matrix sweeping

The following options can be used with the PROC SIMNORMAL statement.

DATA=*SAS-data-set*

specifies the input data set that must be a specially structured TYPE=CORR, COV, UCORR, UCOV, or SSCP SAS data set. If the DATA= option is omitted, the most recently created SAS data set is used.

SEED=*seed-value*

specifies the seed to use for the random number generator. If the SEED= value is omitted, the system clock is used. If the system clock is used, a note is written to the log; the note gives the seed value based on the system clock. In addition, the random seed stream is copied to the OUT= data set if the OUTSEED option is specified.

SEEDBY

specifies that the seed stream be reinitialized for each BY group. By default, a single random stream is used over all BY groups. If you specify SEEDBY, the random stream starts again at the initial seed value. This initial value is from the SEED= value that you specify. If you do not specify a SEED=value, the system clock generates this initial seed.

For example, suppose you had a TYPE=CORR data set with BY groups, and the mean, variances, and covariance or correlation values were identical for each BY group. Then if you specified SEEDBY, the simulated values in each BY group in the OUT= data set would be identical.

OUT=*SAS-data-set*

specifies a SAS data set in which to store the simulated values for the VAR variables. If you omit the OUT=option, the output data set is created and given a default name by using the DATA*n* convention.

See the section "OUT= Output Data Set" on page 9558 for details.

NUMREAL=*n*

specifies the number of realizations to generate. A value of NUMREAL=500 generates 500 observations in the OUT=dataset, or 500 observations within each BY group if a BY statement is given.

NUMREAL can be abbreviated as NUMR or NR.

OUTSEED

requests that the seed values be included in the OUT= data set. The variable Seed is added to the OUT= data set. The first value of Seed is the SEED= value specified in the PROC SIMNORMAL statement (or obtained from the system clock); subsequent values are produced by the random number generator.

OUTCOND

requests that the values of the conditioning variables be included in the OUT= data set. These values are constant for the data set or within a BY group. Note that specifying OUTCOND can greatly increase the size of the OUT= data set. This increase depends on the number of conditioning variables.

SINGULAR1=*number*

specifies the first singularity criterion, which is applied to the Cholesky decomposition of the covariance matrix. The SINGULAR1= value must be in the range $(0, 1)$. The default value is 10^{-8}. SINGULAR1 can be abbreviated SING1.

SINGULAR2=*number*

specifies the second singularity criterion, which is applied to the sweeping of the covariance or correlation matrix to obtain the conditional covariance. The SINGULAR2=option is applicable only when a CONDITION statement is given. The SINGULAR2= value must be in the range $(0, 1)$. The default value is 10^{-8}. SINGULAR2 can be abbreviated SING2.

BY Statement

BY *variables* **;**

A BY statement can be used with the SIMNORMAL procedure to obtain separate simulations for each covariance structure defined by the BY variables. When a BY statement appears, the procedure expects the input DATA= data set to be sorted in the order of the BY variables. If a CONDITION statement is used along with a BY statement, there must be a _TYPE_='COND' observation within each BY group. Note that if a BY statement is specified, the number of realizations specified by the NUMREAL= option are produced for each BY group.

CONDITION Statement

CONDITION | COND *variables* **;**

A CONDITION statement specifies the conditioning variables. The presence of a CONDITION statement requests that a conditional simulation be performed.

The lack of a CONDITIONAL statement simply means that an unconditional simulation for the VAR variables is to be performed.

If a CONDITION statement is given, the variables listed must be numeric variables in the DATA= data set. This requires a conditioning value for each of the CONDITION variables. This value is supplied by adding a _TYPE_='COND' observation for each CONDITION variable. Such observations are added to the DATA= data set by a DATA step.

Note that a data set created by the CORR procedure is automatically given the TYPE=COV, UCOV, CORR, or UCORR attribute, so you do not have to specify the TYPE= option in the DATA= option in the PROC SIMNORMAL statement. However, when adding the conditioning values by using a DATA step with a SET statement, you must use the TYPE=COV, UCOV, CORR, or UCORR attribute in the new data set. See the section "Getting Started: SIMNORMAL Procedure" on page 9554 for an example in which the TYPE is set.

VAR Statement

VAR *variables* **;**

Use the VAR statement to specify the analysis variables. Only numeric variables can be specified. If a VAR statement is not given, all numeric variables in the DATA= data set that are not in the CONDITION or BY statement are used.

OUT= Output Data Set

The SIMNORMAL procedure produces a single output data set: the OUT=*SAS-data-set*.

The OUT= data set contains the following variables:

- all variables listed in the VAR statement
- all variables listed in the BY statement, if one is given
- Rnum, which is the realization number within the current BY group
- Seed, which is current seed value, if the OUTSEED option is specified
- all variables listed in the CONDITION statement, if a CONDITION statement is given and the OUTCOND option is specified

The number of observations is determined by the value of the NUMREAL= option. If there are no BY groups, the number of observations in the OUT= data set is equal to the value of the NUMREAL= option. If there are BY groups, there are number of observations equals the value of the NUMREAL= option for each BY group.

Details: SIMNORMAL Procedure

Introduction

There are a number of approaches to simulating a set of dependent random variables. In the context of spatial random fields, these include sequential indicator methods, turning bands, and the Karhunen-Loeve expansion. See Christakos (1992, Chapter 8) and Deutsch and Journel (1992, Chapter 5) for details.

In addition, there is the LU decomposition method, a particularly simple and computationally efficient for normal or Gaussian variates. For a given covariance matrix, the $\mathbf{LU} = \mathbf{LL}'$ decomposition is computed once, and the simulation proceeds by repeatedly generating a vector of independent $N(0, 1)$ random variables and multiplying by the $\mathbf{L}$ matrix.

One problem with this technique is that memory is required to hold the covariance matrix of all the analysis and conditioning variables in core.

Unconditional Simulation

It is a simple matter to produce an $N(0, 1)$ random number, and by stacking k such numbers in a column vector you obtain a vector with independent standard normal components $\mathbf{W} \sim N_k(\mathbf{0}, \mathbf{I})$. The meaning of the terms *independence* and *randomness* in the context of a deterministic algorithm required for the generation of these numbers is somewhat subtle; see Knuth (1973, Vol. 2, Chapter 3) for a discussion of these issues.

Rather than $\mathbf{W} \sim N_k(\mathbf{0}, \mathbf{I})$, what is required is the generation of a vector $\mathbf{Z} \sim N_k(\mathbf{0}, \mathbf{V})$—that is,

$$\mathbf{Z} = \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_k \end{bmatrix}$$

with covariance matrix

$$\mathbf{V} = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1k} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2k} \\ & \ddots & & \\ \sigma_{k1} & \sigma_{k2} & \cdots & \sigma_{kk} \end{pmatrix}$$

where

$$\sigma_{ij} = \mathrm{Cov}(Z_i, Z_j)$$

If the covariance matrix is symmetric and positive definite, it has a Cholesky root $\mathbf{L}$ such that $\mathbf{V}$ can be factored as

$$\mathbf{V} = \mathbf{L}\mathbf{L}'$$

where $\mathbf{L}$ is lower triangular. See Ralston and Rabinowitz (1978, Chapter 9, Section 3-3) for details. This vector $\mathbf{Z}$ can be generated by the transformation $\mathbf{Z} = \mathbf{L}\mathbf{W}$. Note that this is where the assumption of multivariate normality is crucial. If $\mathbf{W} \sim N_k(\mathbf{0}, \mathbf{I}_k)$, then $\mathbf{Z} = \mathbf{L}\mathbf{W}$ is also normal or Gaussian. The mean of Z is

$$\mathrm{E}(\mathbf{Z}) = \mathbf{L}(\mathrm{E}(\mathbf{W})) = 0$$

and the variance is

$$\mathrm{Var}(\mathbf{Z}) = \mathrm{Var}(\mathbf{L}\mathbf{W}) = \mathrm{E}(\mathbf{L}\mathbf{W}\mathbf{W}'\mathbf{L}') = \mathbf{L}\,\mathrm{E}(\mathbf{W}\mathbf{W}')\,\mathbf{L}' = \mathbf{L}\mathbf{L}' = \mathbf{V}$$

Finally, let $Y_k = Z_k + \mu_k$; that is, you add a mean term to each variable Z_k. The covariance structure of the $Y_k's$ remains the same. Unconditional simulation is done by simply repeatedly generating k $N(0, 1)$ random numbers, stacking them, and performing the transformation

$$\mathbf{W} \longmapsto \mathbf{Z} = \mathbf{L}\mathbf{W} \longmapsto \mathbf{Y} = \mathbf{Z} + \boldsymbol{\mu}$$

Conditional Simulation

For a conditional simulation, this distribution of

$$\mathbf{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_k \end{bmatrix}$$

must be conditioned on the values of the CONDITION variables. The relevant general result concerning conditional distributions of multivariate normal random variables is the following. Let $\mathbf{X} \sim N_m(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix}$$

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}$$

$$\Sigma = \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix}$$

and where bX_1 is $k \times 1$, X_2 is $n \times 1$, $\boldsymbol{\Sigma}_{11}$ is $k \times k$, $\boldsymbol{\Sigma}_{22}$ is $n \times n$, and $\boldsymbol{\Sigma}_{12} = \boldsymbol{\Sigma}'_{21}$ is $k \times n$, with $k + n = m$. The full vector $\mathbf{X}$ has simply been partitioned into two subvectors, $\mathbf{X}_1$ and $\mathbf{X}_2$, and $\boldsymbol{\Sigma}$ has been similarly partitioned into covariances and cross covariances.

With this notation, the distribution of $\mathbf{X}_1$ conditioned on $\mathbf{X}_2 = x_2$ is $N_k(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}})$, with

$$\tilde{\boldsymbol{\mu}} = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x}_2 - \boldsymbol{\mu}_2)$$

and

$$\tilde{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}$$

See Searle (1971, pp. 46–47) for details.

Using the SIMNORMAL procedure corresponds with the conditional simulation as follows. Let $Y_1, \ldots, Y_k$ be the VAR variables as before (k is the number of variables in the VAR list). Let the mean vector for $\mathbf{Y}$ be denoted by $\boldsymbol{\mu}_1 = \mathrm{E}(\mathbf{Y})$. Let the CONDITION variables be denoted by $C_1, \ldots, C_n$ (where n is the number of variables in the COND list). Let the mean vector for $\mathbf{C}$ be denoted by $\boldsymbol{\mu}_2 = \mathrm{E}(\mathbf{C})$ and the conditioning values be denoted by

$$\mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}$$

Then stacking

$$\mathbf{X} = \begin{bmatrix} \mathbf{Y} \\ \mathbf{C} \end{bmatrix}$$

the variance of $\mathbf{X}$ is

$$\mathbf{V} = \mathrm{Var}(\mathbf{X}) = \boldsymbol{\Sigma} = \begin{pmatrix} \mathbf{V}_{11} & \mathbf{V}_{12} \\ \mathbf{V}_{21} & \mathbf{V}_{22} \end{pmatrix}$$

where $\mathbf{V}_{11} = \mathrm{Var}(\mathbf{Y})$, $\mathbf{V}_{12} = \mathrm{Cov}(\mathbf{Y}, \mathbf{C})$, and $\mathbf{V}_{22} = \mathrm{Var}(\mathbf{C})$. By using the preceding general result, the relevant covariance matrix is

$$\tilde{\mathbf{V}} = \mathbf{V}_{11} - \mathbf{V}_{12}\mathbf{V}_{22}^{-1}\mathbf{V}_{21}$$

and the mean is

$$\tilde{\boldsymbol{\mu}} = \boldsymbol{\mu}_1 + \mathbf{V}_{12}\mathbf{V}_{22}^{-1}(\mathbf{c} - \boldsymbol{\mu}_2)$$

By using $\tilde{\mathbf{V}}$ and $\tilde{\boldsymbol{\mu}}$, simulating $(\mathbf{Y}|\mathbf{C} = \mathbf{c}) \sim N_k(\tilde{\boldsymbol{\mu}}, \tilde{\mathbf{V}})$ now proceeds as in the unconditional case.

Example: SIMNORMAL Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 113.1: Simulating Correlated Normal Variables

(View the complete code for this example (simnex1.sas) in the example repository.)

The following example illustrates the use of PROC SIMNORMAL to generate variable values conditioned on a set of related or correlated variables.

Suppose you are given a sample of size 50 from ten normally distributed, correlated random variables, $IN_{1,i}, \ldots, IN_{5,i}, OUT_{1,i}, \ldots, OUT_{5,i}, i = 1, \ldots, 50$. The first five variables represent input variables for a chemical manufacturing process, and the last five are output variables.

First, the data are input and the correlation structure is determined by using PROC CORR, as in the following statements. The results are shown in Figure 113.1.1.

```
data a ;
   input in1-in5 out1-out5 ;
   datalines ;
 9.3500    10.0964     7.3177    10.3617    10.3444    9.4612
10.7443     9.9026     9.0144    11.7968
 7.8599    10.4560    10.0075     8.5875    10.0014   10.3869

   ... more lines ...

 8.9174     9.9623     9.5742     9.9713
;
```

```
proc corr data=a cov nocorr outp=outcov ;
   var in1-in5 out1-out5 ;
run ;
```

Output 113.1.1 Correlation of Chemical Process Variables

The CORR Procedure

10 Variables:	in1	in2	in3	in4	in5	out1	out2	out3	out4	out5

Covariance Matrix, DF = 49

	in1	**in2**	**in3**	**in4**	**in5**	**out1**
in1	1.019198331	0.128086799	0.291646382	0.327014916	0.417546732	0.097650713
in2	0.128086799	1.056460818	0.143581799	0.095937707	0.104117743	0.056612934
in3	0.291646382	0.143581799	1.384051249	0.058853960	0.326107730	0.093498839
in4	0.327014916	0.095937707	0.058853960	1.023128678	0.347916864	0.022915645
in5	0.417546732	0.104117743	0.326107730	0.347916864	1.606858140	0.360270318
out1	0.097650713	0.056612934	0.093498839	0.022915645	0.360270318	0.807007554
out2	0.206698403	-0.121700731	0.078294087	0.125961491	0.297046593	0.217285879
out3	0.516271121	0.266581451	0.481576554	0.179627237	0.749212945	0.064816340
out4	0.118726106	0.092288067	0.057816322	0.075028230	0.220196337	-0.053931448
out5	0.261770905	-0.020971411	0.259053423	0.078147576	0.349618466	0.037758721

Covariance Matrix, DF = 49

	out2	**out3**	**out4**	**out5**
in1	0.206698403	0.516271121	0.118726106	0.261770905
in2	-0.121700731	0.266581451	0.092288067	-0.020971411
in3	0.078294087	0.481576554	0.057816322	0.259053423
in4	0.125961491	0.179627237	0.075028230	0.078147576
in5	0.297046593	0.749212945	0.220196337	0.349618466
out1	0.217285879	0.064816340	-0.053931448	0.037758721
out2	0.929455806	0.206825664	0.138551008	0.054039499
out3	0.206825664	1.837505268	0.292963975	0.165910481
out4	0.138551008	0.292963975	0.832831377	-0.067396486
out5	0.054039499	0.165910481	-0.067396486	0.697717191

Simple Statistics

Variable	**N**	**Mean**	**Std Dev**	**Sum**	**Minimum**	**Maximum**
in1	50	10.18988	1.00955	509.49400	7.63500	12.58860
in2	50	10.10673	1.02784	505.33640	8.12580	13.78310
in3	50	10.14888	1.17646	507.44420	7.31770	12.40080
in4	50	10.03884	1.01150	501.94200	7.40490	11.99060
in5	50	10.22587	1.26762	511.29340	7.23350	12.93360
out1	50	9.85347	0.89834	492.67340	8.01220	12.24660
out2	50	9.96857	0.96408	498.42840	7.76420	12.09450
out3	50	10.29588	1.35555	514.79410	7.29660	13.74200
out4	50	10.15856	0.91260	507.92780	8.43090	12.45230
out5	50	10.26023	0.83529	513.01130	7.86060	11.96000

After the mean and correlation structure are determined, any subset of these variables can be simulated.

Suppose you are interested in a particular function of the output variables for two sets of values of the input variables for the process. In particular, you are interested in the mean and variability of the following function over 500 runs of the process conditioned on each set of input values:

$$f(out_1, \ldots, out_5) = \frac{out_1 - out_3}{out_1 + out_2 + out_3 + out_4 + out_5}$$

Although the distribution of these quantities could be determined theoretically, it is simpler to perform a conditional simulation by using PROC SIMNORMAL.

To do this, you first append a _TYPE_='COND' observation to the covariance data set produced by PROC CORR for each group of input values:

```
data cond1 ;
   _TYPE_='COND' ;
   in1 = 8    ;
   in2 = 10.5 ;
   in3 = 12   ;
   in4 = 13.5 ;
   in5 = 14.4 ;
   output ;
run ;

data cond2 ;
   _TYPE_='COND' ;
   in1 = 15.4 ;
   in2 = 13.7 ;
   in3 = 11 ;
   in4 = 7.9 ;
   in5 = 5.5 ;
   output ;
run ;
```

Next, each of these conditioning observations is appended to a copy of the OUTP=OUTCOV data from the CORR procedure, as in the following statements. A new variable, INPUT, is added to distinguish the sets of input values. This variable is used as a BY variable in subsequent steps.

```
data outcov1 ;
   input=1 ;
   set outcov cond1 ;
run ;

data outcov2 ;
   input=2 ;
   set outcov cond2 ;
run ;
```

Finally, these two data sets are concatenated:

```
data outcov ;
   set outcov1 outcov2 ;
run ;
proc print data=outcov ;
```

```
   where (_type_ ne 'COV') ;
run ;
```

Figure 113.1.2 shows the added observations.

Output 113.1.2 OUTP= Data Set from PROC CORR with _TYPE_=COND Observations Appended

Obs	input	_TYPE_	_NAME_	in1	in2	in3	in4	in5	out1	out2	out3	out4	out5
11	1	MEAN		10.1899	10.1067	10.1489	10.0388	10.2259	9.8535	9.9686	10.2959	10.1586	10.2602
12	1	STD		1.0096	1.0278	1.1765	1.0115	1.2676	0.8983	0.9641	1.3555	0.9126	0.8353
13	1	N		50.0000	50.0000	50.0000	50.0000	50.0000	50.0000	50.0000	50.0000	50.0000	50.0000
14	1	COND		8.0000	10.5000	12.0000	13.5000	14.4000	.	.	.	.	.
25	2	MEAN		10.1899	10.1067	10.1489	10.0388	10.2259	9.8535	9.9686	10.2959	10.1586	10.2602
26	2	STD		1.0096	1.0278	1.1765	1.0115	1.2676	0.8983	0.9641	1.3555	0.9126	0.8353
27	2	N		50.0000	50.0000	50.0000	50.0000	50.0000	50.0000	50.0000	50.0000	50.0000	50.0000
28	2	COND		15.4000	13.7000	11.0000	7.9000	5.5000	.	.	.	.	.

You now run PROC SIMNORMAL, specifying the input data set and the VAR and COND variables. Note that you must specify a TYPE=COV or TYPE=CORR for the input data set. PROC CORR automatically assigns a TYPE=COV or TYPE=CORR attribute for the OUTP= data set. However, since the intermediate DATA steps that appended the _TYPE_='COND' observations turned off this attribute, an explicit TYPE=CORR in the DATA= option in the PROC SIMNORMAL statement is needed.

The specification of PROC SIMNORMAL now follows from the problem description. The condition variables are IN1–IN5, the analysis variables are OUT1–OUT5, and 500 realizations are required. A seed value can be chosen arbitrarily, or the system clock can be used. Note that in the following statements, the simulation is done for each of the values of the BY variable INPUT:

```
proc simnormal data=outcov(type=cov)
      out = osim
      numreal = 500
      seed = 33179
      ;
   by input ;
   var out1-out5 ;
   cond in1-in5 ;
run;

data b;
   set osim ;
   denom = sum(of out1-out5) ;
   if abs(denom) < 1e-8 then ff = . ;
   else ff = (out1-out3)/denom ;
run ;
```

The DATA step that follows the simulation computes the function $f(out_1, \ldots, out_5)$; in the following statements the UNIVARIATE procedure computes the simple statistics for this function for each set of conditioning input values. This is shown in Figure 113.1.3, and Figure 113.1.4 shows the distribution of the function values for each set of input values by using the SGPANEL procedure.

```
proc univariate data=b ;
   by input ;
   var ff ;
run ;
title ;
proc sgpanel data=b ;
   panelby input ;
   REFLINE 0 / axis= x ;
   density ff ;
run ;
```

Output 113.1.3 Simple Statistics for ff for Each Set of Input Values

The UNIVARIATE Procedure
Variable: ff

input=1

Moments			
N	500	**Sum Weights**	500
Mean	-0.0134833	**Sum Observations**	-6.7416303
Std Deviation	0.02830426	**Variance**	0.00080113
Skewness	0.56773239	**Kurtosis**	1.31522925
Uncorrected SS	0.49066351	**Corrected SS**	0.39976435
Coeff Variation	-209.92145	**Std Error Mean**	0.0012658

input=1

Basic Statistical Measures			
Location		**Variability**	
Mean	-0.01348	**Std Deviation**	0.02830
Median	-0.01565	**Variance**	0.0008011
Mode	.	**Range**	0.21127
		Interquartile Range	0.03618

input=1

Tests for Location: Mu0=0				
Test		**Statistic**	**p Value**	
Student's t	**t**	-10.6519	**Pr > |t|**	<.0001
Sign	**M**	-106	**Pr >= |M|**	<.0001
Signed Rank	**S**	-33682	**Pr >= |S|**	<.0001

Output 113.1.3 *continued*

input=1

Quantiles (Definition 5)	
Level	**Quantile**
100% Max	0.11268600
99%	0.07245656
95%	0.03270269
90%	0.02064338
75% Q3	0.00370322
50% Median	-0.01564850
25% Q1	-0.03247389
10%	-0.04716239
5%	-0.05572806
1%	-0.07201126
0% Min	-0.09858350

input=1

Extreme Observations			
Lowest		**Highest**	
Value	**Obs**	**Value**	**Obs**
-0.0985835	471	0.0750538	22
-0.0908179	472	0.0794747	245
-0.0802423	90	0.0840160	48
-0.0760645	249	0.1004812	222
-0.0756070	226	0.1126860	50

The UNIVARIATE Procedure
Variable: ff

input=2

Moments			
N	500	**Sum Weights**	500
Mean	-0.0405913	**Sum Observations**	-20.295631
Std Deviation	0.03027008	**Variance**	0.00091628
Skewness	0.1033062	**Kurtosis**	-0.1458848
Uncorrected SS	1.28104777	**Corrected SS**	0.4572225
Coeff Variation	-74.57289	**Std Error Mean**	0.00135372

input=2

Basic Statistical Measures			
Location		**Variability**	
Mean	-0.04059	**Std Deviation**	0.03027
Median	-0.04169	**Variance**	0.0009163
Mode	.	**Range**	0.18332
		Interquartile Range	0.04339

Output 113.1.3 *continued*

input=2

Tests for Location: Mu0=0				
Test	**Statistic**		**p Value**	
Student's t	**t**	-29.985	**Pr > |t|**	<.0001
Sign	**M**	-203	**Pr >= |M|**	<.0001
Signed Rank	**S**	-58745	**Pr >= |S|**	<.0001

input=2

Quantiles (Definition 5)	
Level	**Quantile**
100% Max	0.06101208
99%	0.02693796
95%	0.01008202
90%	-0.00111776
75% Q3	-0.01847726
50% Median	-0.04169199
25% Q1	-0.06187039
10%	-0.07798499
5%	-0.08606522
1%	-0.11026564
0% Min	-0.12231183

input=2

Extreme Observations			
Lowest		**Highest**	
Value	**Obs**	**Value**	**Obs**
-0.122312	937	0.0272906	688
-0.119884	980	0.0291769	652
-0.113512	920	0.0388217	670
-0.112345	523	0.0477261	845
-0.110497	897	0.0610121	632

Output 113.1.4 Frequency Plot for ff for Each Set of Input Values

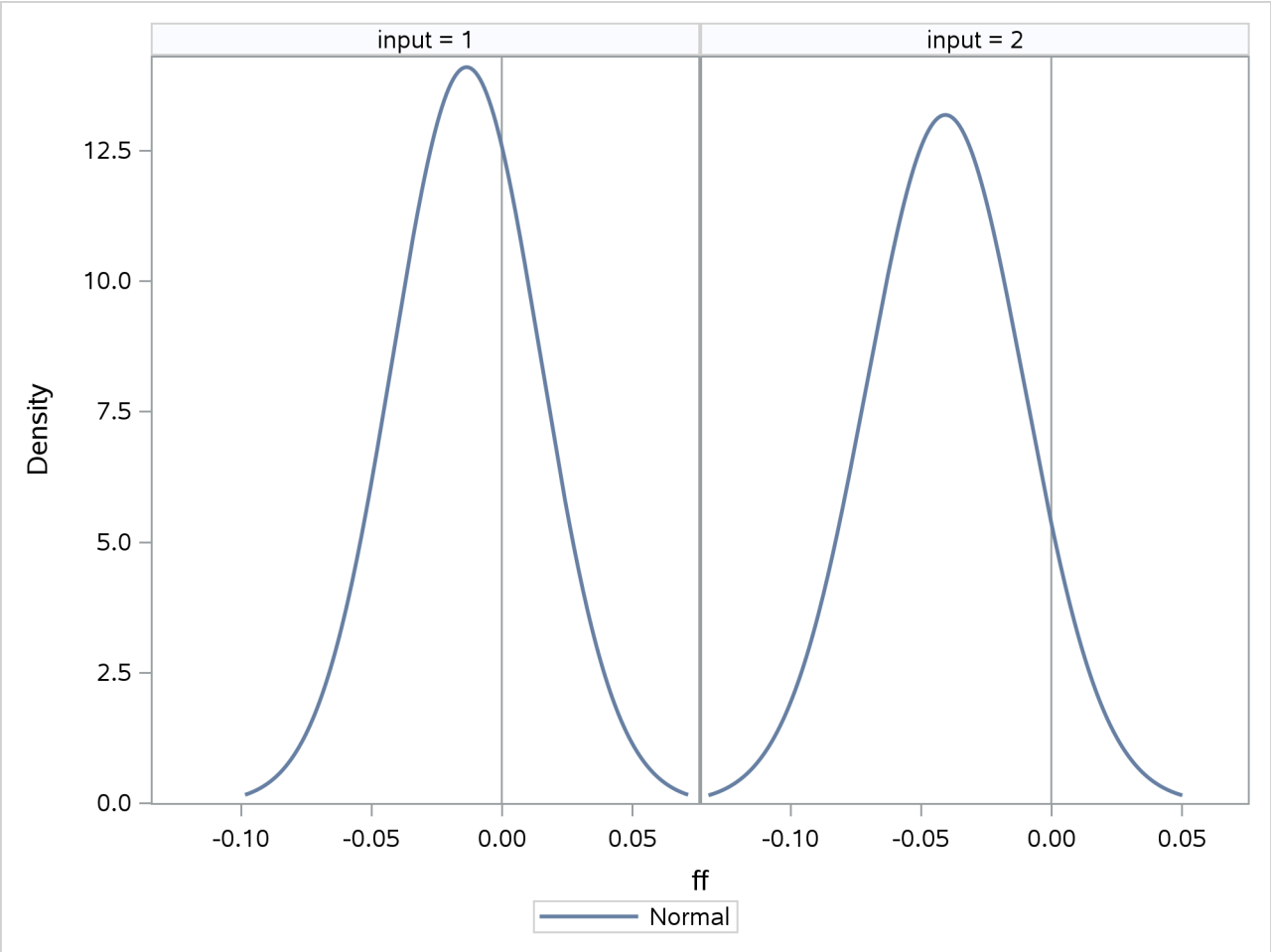

References

Christakos, G. (1992). *Random Field Models in Earth Sciences*. New York: Academic Press.

Deutsch, C. V., and Journel, A. G. (1992). *GSLIB: Geostatistical Software Library and User's Guide*. New York: Oxford University Press.

Knuth, D. E. (1973). *Seminumerical Algorithms.* Vol. 2 of The Art of Computer Programming. Reading, MA: Addison-Wesley.

Ralston, A., and Rabinowitz, P. (1978). *A First Course in Numerical Analysis*. New York: McGraw-Hill.

Searle, S. R. (1971). *Linear Models*. New York: John Wiley & Sons.

Chapter 114
The SPP Procedure

Contents

Overview: SPP Procedure . . . **9570**
- Classes of Spatial Data . . . 9570
- Introduction to Point Pattern Analysis . . . 9571

Getting Started: SPP Procedure . . . **9572**

Syntax: SPP Procedure . . . **9578**
- PROC SPP Statement . . . 9579
- BY Statement . . . 9586
- COVTEST Statement . . . 9586
- MODEL Statement . . . 9587
- NLOPTIONS Statement . . . 9589
- PARMS Statement . . . 9590
- PROCESS Statement . . . 9592
- TREND Statement . . . 9597

Details: SPP Procedure . . . **9598**
- Testing for Complete Spatial Randomness . . . 9598
  - Quadrat Count Test for CSR . . . 9598
- Exploring Interpoint Interaction . . . 9598
  - Nearest-Neighbor Distance Functions . . . 9599
  - Statistics Based on Second-Order Characteristics . . . 9600
- Distance Functions for Multitype Point Patterns . . . 9602
- Border Edge Correction for Distance Functions . . . 9603
- Confidence Intervals for Summary Statistics . . . 9604
- Ripley-Rasson Window Estimator . . . 9604
- Covariate Dependence Tests . . . 9605
  - EDF Goodness-of-Fit Tests . . . 9605
  - Testing Covariate Dependency with EDF Tests . . . 9606
- Nonparametric Intensity Estimation . . . 9606
- Inhomogeneous Poisson Process Model Fitting . . . 9608
  - Likelihood Methods for Model Fitting . . . 9608
  - Fit Statistics . . . 9609
  - Fitted Model Validation That Uses Goodness-of-Fit Tests . . . 9609
  - Fitted Model Validation That Uses Residuals . . . 9609
- Negative Binomial Modeling . . . 9611
- Output Data Sets . . . 9611
- Displayed Output . . . 9613
- ODS Table Names . . . 9615

ODS Graphics 9616
Examples: SPP Procedure **9619**
Example 114.1: Exploration of a Multitype Point Pattern 9619
Example 114.2: Testing Covariate Dependence of a Point Pattern 9623
Example 114.3: Intensity Model Validation Diagnostics 9625
References **9632**

Overview: SPP Procedure

The SPP procedure performs analysis for spatial point patterns in two dimensions. You can specify the point process rectangular window or rely on the input data set coordinates. Summary descriptions are available through the F, G, J, K functions, which compare the empirical function distributions to the theoretical homogeneous Poisson functions.

The SPP procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For more information about the graphics available in PROC SPP, see the section "ODS Graphics" on page 9616.

Classes of Spatial Data

There are three broad classes of spatial data:

- *Point-referenced data* are values that are sampled at specific locations within an area of a predefined size. An example is air temperatures that are measured where weather monitoring instruments are located. The stochastic nature of spatial processes can be described by using spatial random fields (SRFs). A set of point-referenced data can be seen as a realization of a continuous SRF that takes values over the entire study area. The values at unsampled locations are unknown but can be predicted by means of geostatistical analysis. You can analyze point-referenced data by using the SAS/STAT procedures VARIOGRAM, KRIGE2D, and SIM2D.

- *Areal (lattice) data* are values for a fixed number of areal units within a particular area. These data differ the point-referenced data in that one areal observation is assigned to a whole areal unit instead of to a specific location. An example is crime rates that are aggregated over counties within a state.

- *Point pattern data* are a collection of locations of single events of a spatial process. In this category, the study area can have a variable size and observations might have associated covariates, but the main interest is in their spatial patterns of occurrence. Examples include locations of tree growth, locations of petty crimes, and so on. A set of point-pattern data can be seen as a realization of a discrete SRF that has values only at the event locations (Illian et al. 2008, p. 44). A collection of this type of data is known as a spatial *point pattern*. Point pattern analysis usually does not refer to the SRF concept. The applied techniques in point patterns differ from the geostatistical approach, although both types of analysis share corresponding measures to describe correlation among the data. You can use the SPP procedure to analyze point pattern data.

Introduction to Point Pattern Analysis

In point pattern analysis, you want to describe characteristics of the *events* (observations) that compose the pattern. The events are manifestations of a phenomenon or process at random locations. Therefore, your analysis goal is to investigate underlying connections among these events that could explain the phenomenon.

In some cases, events might have additional attributes, known as *marks*. If a point pattern has a mark, then it is called a *marked point pattern*. There can be *continuous marks* or *categorical marks*, depending on whether the mark attribute takes continuous values or values from a list of discrete levels, respectively. A marked point pattern that has a categorical mark attribute is known as a *multitype point pattern*. A multitype point pattern is also called a *multivariate point pattern* because you can view it as a collection of point patterns, one for each type.

To study the events, you use the concepts of the *study region* (also called a *study window*) to represent the area where the point pattern is defined. The window selection can be a subjective choice, and it definitely can affect the analysis. When the window is a subregion of a larger region where the point process operates, you might need to account for *edge effects*. This term describes discrepancies that can appear in the analysis, depending on whether you consider that events close to the window edges have neighbors outside the window area.

Point pattern analysis often focuses on whether interaction exists among the observations in a spatial point pattern. That is, you test whether the points are spread evenly around the study region with no particular pattern, or alternatively whether there tends to be more or less clumping of points than you would expect purely from randomness. To this end, you usually test the hypothesis of complete spatial randomness (CSR) in the point pattern. According to CSR, the events follow a Poisson distribution with constant mean, and they have no interactions. A point pattern can follow CSR, in which case it is known as a *homogeneous Poisson process*. Alternatively, a point pattern can demonstrate event interaction or clustering.

You can test CSR by using heuristic approaches that use *sparse sampling methods* in exploratory and summary analysis. Two general approaches to this are as follows:

- *distance methods*, where you compare the empirical distribution function (EDF) of distance between events with an EDF that is based on the CSR assumption
- *quadrats*, where you partition the spatial framework into smaller subregions and study the number of events (also known as the *quadrat count*) in each subregion

The SPP procedure provides options for implementing both of these approaches. For more information, see the sections "Testing for Complete Spatial Randomness" on page 9598 and "Statistics Based on Second-Order Characteristics" on page 9600.

You can tell a lot about the behavior of a point pattern if you have an expression for the point pattern *intensity*, which shows the number of events per unit area. A simple way to estimate intensity from the point pattern events is to produce kernel density estimates. You can also model the intensity by maximizing suitable pseudolikelihood expressions for the logarithmic intensity. Intensity models can also incorporate information about covariate variables; together with distance methods, they enable you to examine whether a covariate plays a significant role in the underlying process.

A SAS/STAT procedure that compares to PROC SPP is the KDE procedure, which fits the special case of Gaussian bivariate kernels for the purpose of nonparametric density estimation. PROC SPP enables you to

perform much more extensive nonparametric intensity estimation by using different types of kernels, and it provides support for adaptive kernel estimation. In addition, PROC SPP enables you to fit parametric inhomogeneous poisson process models and use a variety of residual diagnostics to perform model validation.

Getting Started: SPP Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/sppgs.sas.)

This example uses forestry data, which are shown in Figure 114.4, to show how you can use PROC SPP to fit a model for the first-order intensity of a spatial point pattern. The Sashelp.BEI data set contains the locations of 3,604 trees in tropical rain forests. A study window of 1,000 × 500 square kilometers is appropriate. The data set also contains covariates that are represented by the variables Gradient and Elevation, which are collected at 20,301 locations on a regular grid across the study region. The variable Trees distinguishes the event observations in the data set. These data are a part of a much larger data set, which contains the positions of hundreds of thousands of trees that belong to thousands of species (Condit 1998; Hubbell and Foster 1983; Condit, Hubbell, and Foster 1996).[1] The Sashelp.BEI data set contains five variables:

- X and Y: the X and Y coordinates for locations of trees and for measurements of the height and slope of the study area
- Trees: a 0/1 variable that indicates which observation corresponds to locations of trees: 1 indicates the presence of a tree, and 0 indicates absence
- Elevation: which measures how far the study area is above sea level
- Gradient: which measures the slope of the study area

The following statements produce a plot of the event observations (which is shown in Figure 114.4) and plots of the covariates (which are shown in Figure 114.5 and Figure 114.6).

```
ods graphics on;
proc spp data=sashelp.bei plots(equate)=(trends observations);
   process trees = (x, y /area=(0,0,1000,500) Event=Trees);
   trend grad = field(x,y, gradient);
   trend elev = field(x,y, elevation);
run;
```

In addition, the preceding statements produce three tables, which are shown in Figure 114.1, Figure 114.2, and Figure 114.3. The number of observations in the combined data set is shown in Figure 114.1; it includes both the number of event observations and the number of covariate observations.

[1] This data set is used with kind permission from Professor S. Hubbell, with acknowledgment of the support of the Center for Tropical Forest Science of the Smithsonian Tropical Research Institute and the primary granting agencies that have supported the BCI plot. The BCI forest dynamics research project was made possible by National Science Foundation grants to Stephen P. Hubbell: DEB-0640386, DEB-0425651, DEB-0346488, DEB-0129874, DEB-00753102, DEB-9909347, DEB-9615226, DEB-9615226, DEB-9405933, DEB-9221033, DEB-9100058, DEB-8906869, DEB-8605042, DEB-8206992, DEB-7922197, support from the Center for Tropical Forest Science, the Smithsonian Tropical Research Institute, the John D. and Catherine T. MacArthur Foundation, the Mellon Foundation, the Small World Institute Fund, and numerous private individuals, and through the hard work of over 100 people from 10 countries over the past two decades. The plot project is part of the Center for Tropical Forest Science, a global network of large-scale demographic tree plots.

Figure 114.1 Number of Events and Number of Covariate Observations

The SPP Procedure

Observations Read	24205
Observations Used	23905
Event Observations Read	3604
Event Observations Used	3604
Gradient Observations Read	20301
Gradient Observations Used	20301
Elevation Observations Read	20301
Elevation Observations Used	20301

Figure 114.2 provides some summary information about the point pattern, including the average intensity or the number of events per unit area.

Figure 114.2 Exploratory Information about the Point Pattern

Summary of Point Pattern	
Data Type	Point Pattern
Pattern Name	trees
Region Type	User Defined Window
Region X Range	[0,1000] Units
Region Y Range	[0,500] Units
Region X Size	1000 Units
Region Y Size	500 Units
Region Area	500000 Square Units
Observations in Window	3604
Average Intensity	0.007208
Grid Nodes in X	50
Grid Nodes in Y	50
Grid Nodes in Window	2500
Quadrat Dimension in X	10
Quadrat Dimension in Y	10

Figure 114.3 provides the results of a default 10×10 quadrat-based Pearson chi-square test for CSR.

Figure 114.3 Pearson Chi-Square Test for CSR

Pearson Chi-Square Test for CSR				
Expected Frequency	**DF**	**Dispersion Index**	**Chi-Square**	**Pr > ChiSq**
36.04	99	33.222	3288.95	<.0001

Figure 114.4 Spatial Point Pattern of Tropical Rain Forest Trees

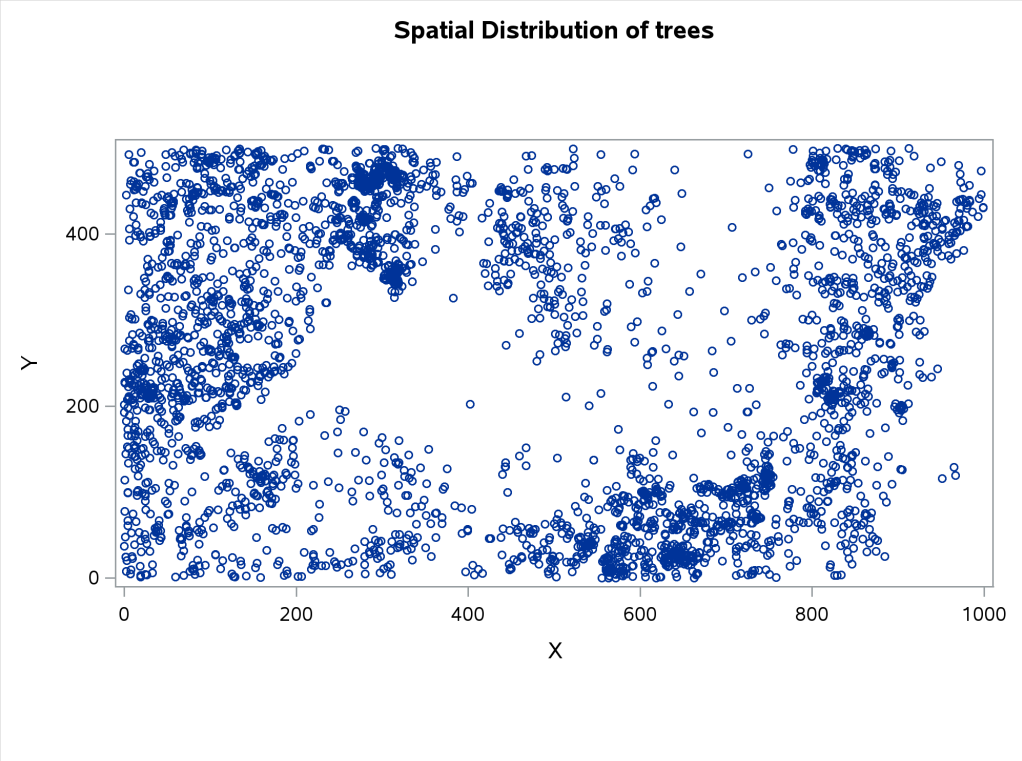

Figure 114.5 Spatial Covariate Gradient

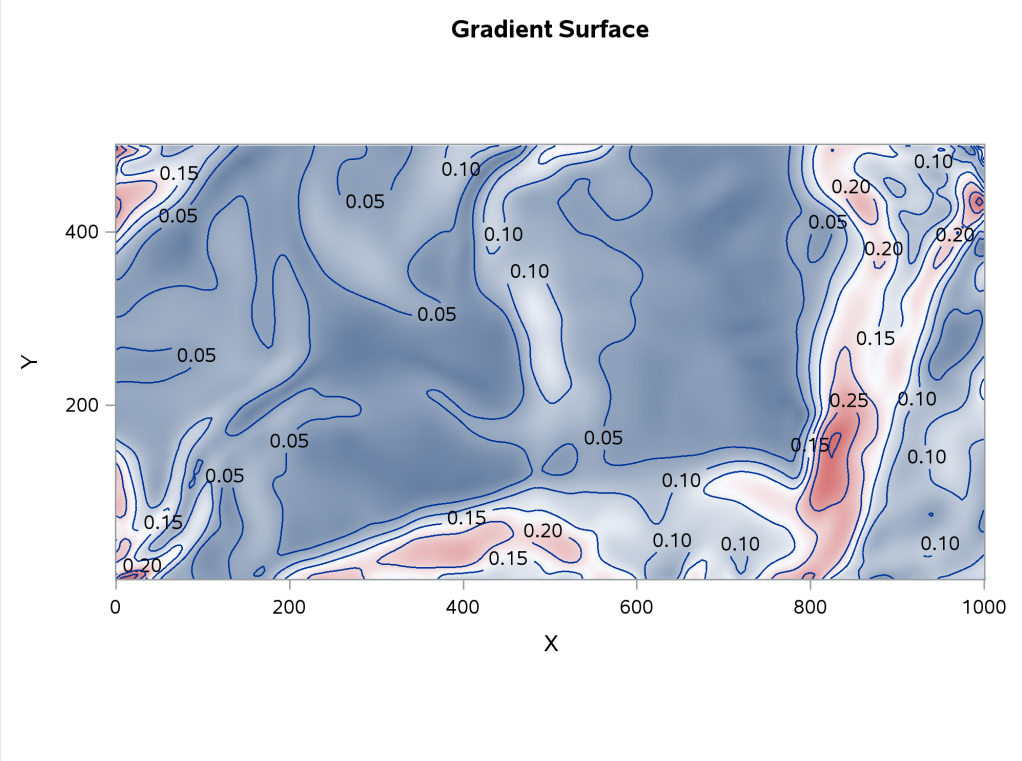

Figure 114.6 Spatial Covariate Elevation

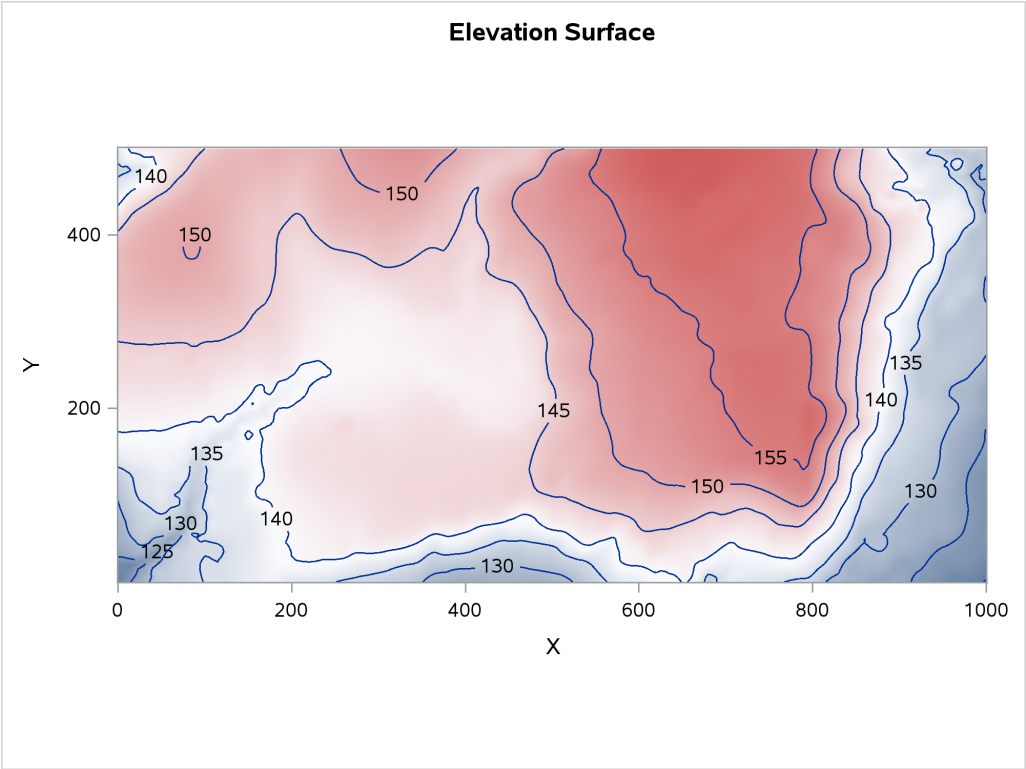

The variables Gradient and Elevation are both continuous functions, because any arbitrary point that is chosen in the study area has a value for both these variables. However, these variables are sampled at select points where measuring them is easy. In spatial analysis and geographic information systems (GISs), such variables are termed *field* variables and are associated with a spatial trend. You can include such variables in the SPP procedure by using the TREND statement.

The sashelp.bei data contains combined information for both the point pattern and the spatial covariates. However, the SPP procedure requires you to identify the point pattern event identifier separately. This is done by using the EVENT= option in the PROCESS statement to specify that the variable Trees identifies the event.

It is natural to suppose that tree growth is affected by the gradient and elevation of the surrounding land. Hence, you can use the gradient and elevation in a parametric model to model the intensity of tree growth in the study area. Such a model is an inhomogeneous Poisson process (Baddeley 2010, p. 354), whose first-order intensity, $\lambda(s)$, is log linear in the covariates. You can use the MODEL statement to compose models for a point pattern's intensity. In the MODEL statement, you specify the response pattern on the left side. The response pattern is a process that you define before you specify the MODEL statement. You can specify any covariates that are likely to influence the target point pattern on the right side of the MODEL statement syntax.

To obtain a plot of the model-based intensity estimate, you specify the PLOTS=INTENSITY option. In addition, if you want to request residual diagnostics, you can specify the PLOTS=RESIDUAL option. If you want to specify a response grid to obtain the intensity estimates, you can use the GRID option in the MODEL statement. The following statements explore the influence of the covariates Elevation and Gradient on the intensity of Tree presence:

```
proc spp data=sashelp.bei plots(equate)=(residual intensity);
   process trees = (x,y /area=(0,0,1000,500) event=Trees);
   trend elev = field(x,y,elevation);
   trend grad = field(x,y,gradient);
   model trees = elev grad / grid(64,64) residual(B=70) ;
run;
```

In addition to the tables shown in previous figures, these statements produce a table that contains the parameter estimates (Figure 114.7) and a fit summary table (Figure 114.8). The parameter estimates designate the intercept value and the values of the factors of the model terms. The relative values of the parameter estimates indicate how much each factor contributes to the model. In this case, Gradient is much more important in modeling where trees grow than Elevation, although both are highly significant.

Figure 114.7 Parameter Estimates Table

The SPP Procedure

Poisson Parameter Estimates				
Parameter	Estimate	Standard Error	z Value	Approx Pr > |z|
Intercept	-8.5672	0.3415	-25.08	<.0001
Elevation	0.02146	0.002291	9.37	<.0001
Gradient	5.8616	0.2567	22.83	<.0001

The fit summary table in Figure 114.8 shows the model fit statistics. You can use these values to compare multiple fits from different models and to select an optimal model in your study.

Figure 114.8 Fit Summary Table

Fit Statistics	
Criterion	Value
-2 Log Likelihood	42290.0
AIC (smaller is better)	42296.0
BIC (smaller is better)	42316.8

The corresponding fitted intensity is shown in Figure 114.9.

Figure 114.9 Intensity Estimates of Tree Presence in Study Area

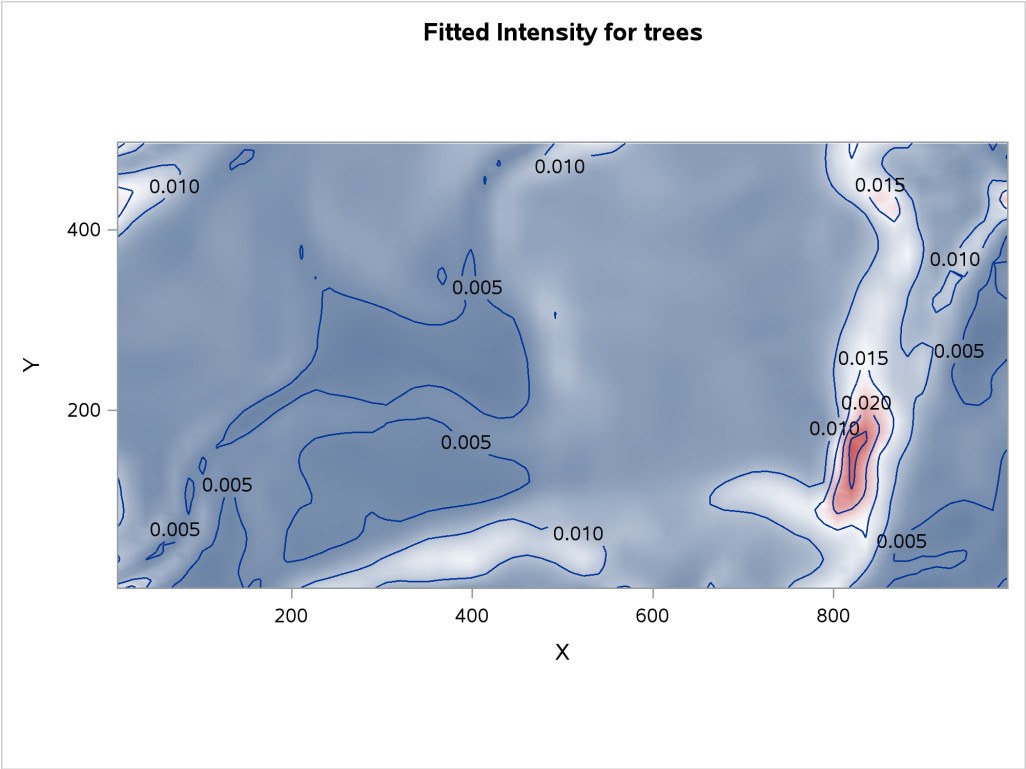

The resulting residual diagnostics are shown in Figure 114.10.

Figure 114.10 Residual Diagnostics for Fitted Log-Intensity Model

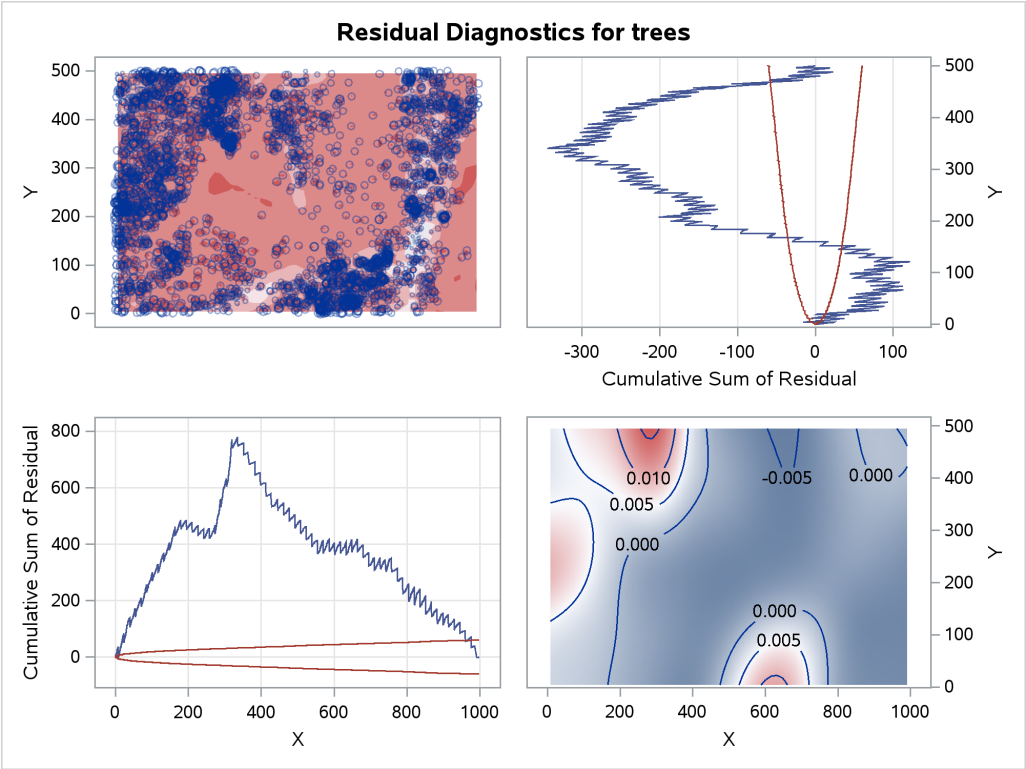

The residual diagnostics plot in Figure 114.10 provides an informal assessment of the fitted parametric model. In particular, the smoothed residual plot in the right bottom corner reveals a trend in the residual that is not accounted for by the model. In addition, the lurking variable plots with respect to the coordinate variables show significant deviation from the 2σ limits, indicating that the model does not account for a variation in intensity with respect to these variables.

Syntax: SPP Procedure

The following statements are available in PROC SPP:

```
PROC SPP options ;
    BY variables ;
    PROCESS name = (variables < /pattern-options >)< /process-options < distance-function-options > >
                ;
    TREND name = FIELD(field-definition ) ;
    COVTEST process-name = trend-name < trend-name, . . . >< /options > ;
    MODEL process-name = < trend-name, . . . >< /model-options > ;
    PARMS value-list < / PARMSDATA=SAS-data set > ;
    NLOPTIONS < options > ;
```

You must specify at least one PROCESS statement. The MODEL statement and the COVTEST statements must have one process variable on the left side and can have one or more processes or trends on the right

side. When you specify the PARMS and NLOPTIONS statements, they must be preceded by the MODEL statement.

The following sections describe the PROC SPP statement and then describe the other statements in alphabetical order.

PROC SPP Statement

PROC SPP *options* **;**

The PROC SPP statement invokes the SPP procedure. Table 114.1 summarizes the options available in the PROC SPP statement.

Table 114.1 Options Available in the PROC SPP Statement

Option	Description
DATA=	Specifies the input data set
EDGECORR=	Requests edge correction in the analysis
NODUP	Specifies inclusion or exclusion of collocated observations
NOPRINT	Suppresses normal display of results
PLOTS	Specifies the plot display and options
SEED=	Specifies the seed value for the random number generator

You can specify the following *options* in the PROC SPP statement.

DATA=*SAS-data-set*

specifies a SAS data set that contains the x and y coordinate variables of one or more point patterns, associated mark variables, and event identifiers. Mark variables and event identifiers are specified using MARK= and EVENT= options, respectively, in the PROCESS statement. If your analysis involves covariates, you must also include them in the DATA= data set. When you include covariates, you must identify individual point patterns by specifying the EVENT= option in the PROCESS statement. You must specify a DATA=*SAS-data-set*; there is no default.

EDGECORR=ON | OFF

specifies whether you want to correct edge effects in the distance function computations and kernel density estimation. Edge correction is not applicable for the J function. For more information about how SPP implements edge correction, see the section "Border Edge Correction for Distance Functions" on page 9603. By default, EDGECORR=ON.

NODUP=*nodup-option*

specifies whether to eliminate multiple records of data that have the same pairs of coordinates in the DATA= data set. When multiple such records exist among observations of the event, or among observations of the same covariate variable, they are known as duplicates. For example, if two or more event records feature the same coordinates, then your data contain duplicates. However, if your data contain a record of an event and a record of a covariate that happen to be sampled at the same coordinates, then they are not duplicates.

The analysis of a spatial point pattern usually requires that no two events can share the same location. If your data include such duplicates, this option enables you to deal with them in different ways. You can specify the following values:

TRUE < (*true-suboption*) >
removes duplicates from the analysis. You can also specify the following *true-suboption*:

KEEP=AVG | ONE
specifies how to treat removal of duplicate records. You can specify the following values:

AVG removes all but one record from a set of records that contain duplicate coordinates. In addition, if the duplicates are records of a numeric mark or covariate, then the average attribute value of all duplicate records is assigned to the single record that is retained. If any of the duplicate records has a missing value for the numeric mark or covariate, then it does not contribute to the average. Character variables ignore the KEEP=AVG suboption and retain only the last value in any series of collocated records.

ONE keeps only a single record out of multiple records that have the same duplicate coordinates. When you specify KEEP=ONE, PROC SPP retains the last record in any series of collocated records.

By default, KEEP=ONE.

FALSE
retains and uses all duplicates in the analysis.

If mark or covariate variables are included in the analysis, the NODUP= option specification applies the same mode of action to each individual variable. If PROC SPP finds duplicates, then it issues a note. By default, NODUP=TRUE(KEEP=ONE).

NOPRINT
suppresses the normal display of results. This option is useful when you want only to create one or more output data sets with the procedure.
NOTE: This option temporarily disables the Output Delivery System (ODS). For more information, see the section "ODS Graphics" on page 9616.

PLOTS < (*global-plot-options*) > < = *plot-request* < *(options)* > >
PLOTS < (*global-plot-options*) > < = (*plot-request* < *(options)* > < . . . *plot-request* < *(options)* > >) >
controls the plots that are produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around the plot request. Here are some examples:

```
plots=none
plots=observ
plots=(observ intensity)
plots(unpack)=observ
plots=(observ(attr=mark) observ(attr=event))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

You can specify the following *global-plot-options*:

EQUATE
produces all plots that have coordinates so that the axes coordinates have equal size units. This option is ignored for panel plots.

ONLY
suppresses the default plots. Only plots that are specifically requested are displayed.

UNPACKPANEL
UNPACK
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACKPANEL to get each plot in a separate panel. You can specify PLOTS(UNPACKPANEL) to unpack the default plots. You can also unpack individual panel plots by specifying the UNP suboption in the FFUN, GFUN, KFUN, LFUN, and OBSERVATIONS(LEVEL=(SPLIT)) plot options.

You can specify the following individual *plot-requests* and *options*:

ALL
produces all appropriate plots. You can also specify other *options* with ALL.

CSRKSTEST
produces a plot for the Kolmogorov-Smirnov weighted EDF test for complete spatial randomness in the presence of covariates. To request this plot, you must specify the COVTEST statement and include trends on the right side of the COVTEST statement.

EMPTYSPACE < (*emptyspace-plot-options*) >
produces a plot of the nearest-neighbor distance for every grid node in the window. You can specify the following *emptyspace-plot-options*:

FILL=ON | OFF
specifies whether to produce a surface plot of the nearest neighbor distances. By default, FILL=ON.

LINE=ON | OFF
specifies whether to produce a contour line plot of the nearest neighbor distances. By default, LINE=OFF.

OBS=ON | OFF
specifies whether to produce an overlaid scatter plot of the observations in addition to nearest neighbor distances. By default, OBS=OFF.

F < (UNPACK) >

requests that a panel of diagnostics for the empty space function F be produced. The F function is the empirical distribution of observed distances to the nearest observation from any location in the point pattern window. The panel contains four plots: an EDF plot that shows simulation envelopes for CSR, an EDF-CSR difference plot, a PP plot that compares the EDF of the summary statistic, and a confidence interval plot that shows envelopes for the confidence intervals of the summary statistic. If you specify the PLOTS=F option without requesting any distance function calculations in the PROCESS statement, then it is ignored. You can specify the following option:

UNPACK

suppresses paneling of the F function plots and produces each constituent plot in the panel separately.

The F plot is produced when you specify the F option in the PROCESS statement.

G < (UNPACK) >

produces a panel of diagnostics for the nearest-neighbor distance function G. The G function is the empirical distribution function of observed distances to the nearest observation from any other observation in the point pattern window. The panel contains four plots: an EDF plot that shows simulation envelopes for CSR, an EDF-CSR difference plot, a PP plot that compares the EDF of the summary statistic, and a confidence interval plot that shows envelopes for the confidence intervals of the summary statistic. If you specify PLOTS=G option without requesting any distance function calculations in the PROCESS statement, then it is ignored.

You can specify the following option:

UNPACK

suppresses paneling of the G function plots and produces each constituent plot in the panel separately.

The G plot is produced when you specify the G if you specify the G function option in the PROCESS statement.

INTENSITY < (*intensity-plot-options*) >

produces a plot of the estimated intensity function for every grid node in the window. You can specify the following *intensity-plot-options*:

EST=KERNEL | FIT

specifies the source to use for the intensity estimate. You can specify the following values:

KERNEL produces a plot of the intensity kernel density estimate. This suboption is incompatible with requests for standard error in the FILL= and LINE= intensity plot options. If you specify EST=KERNEL and either the FILL=SE suboption or the LINE=SE suboption, then intensity plot request is ignored.

FIT produces a plot of the estimated intensity on the basis of a model fit when you fit an intensity model by specifying the MODEL statement.

FILL=INTENSITY | NONE | SE
specifies which type of surface plot to produce. You can specify the following values:

INTENSITY produces an estimated intensity surface plot.

NONE produces no surface plot.

SE produces a standard errors surface plot.

The default behavior depends on the LINE suboption as follows: If you specify LINE=NONE or entirely omit the LINE suboption, then FILL=INTENSTIY. If you specify LINE=INTENSITY or LINE=SE, then the FILL= suboption is set to the same value as the LINE suboption.

LINE=INTENSITY | NONE | SE
specifies which type of plot to produce. You can specify the following values:

INTENSITY produces an estimated intensity contour line plot.

NONE produces no contour line plot.

SE produces a standard errors contour line plot.

If you omit the LINE suboption, the behavior depends on the FILL suboption as follows: If you specify FILL=NONE or entirely omit the FILL= suboption, then LINE=INTENSITY. If you specify FILL=INTENSITY or FILL=SE, then the LINE suboption is set to the same value as the FILL suboption.

OBS=ON | OFF
specifies whether to produce an overlaid scatter plot of the observations in addition to the intensity plot. By default, OBS=OFF.

You can specify multiple instances of the INTENSITY plot option to produce intensity plots that have different characteristics. If you specify multiple instances of any of the FILL=, LINE=, or OBS= suboptions in the same INTENSITY plot request, then one plot is produced that honors the last value specified for any of these suboptions. If you explicitly specify (or the suboptions imply) the combination FILL=NONE and LINE=NONE, then the intensity plot is not produced.

J < (UNPACK) >
produces a combined plot of the J function. The J function is the ratio of transformations of the F and G nearest-neighbor functions. The combined plot shows both the confidence intervals for the summary statistic and the simulation envelope for comparison with CSR. You can specify the following option:

UNPACK
produces each constituent J plot separately.

J plots are produced when you specify the J option in the PROCESS statement. If you specify PLOTS=J without specifying the J option in the PROCESS statement, then PLOTS=J is ignored.

K < (UNPACK) >

produces a panel of Ripley's K function. The K function is the ratio of the expected number of point pattern observations within distance *r* of any other observation divided by the average intensity value of the point pattern. The panel contains four plots: an EDF plot that shows simulation envelopes for CSR, an EDF-CSR difference plot, a PP plot that compares the EDF of the summary statistic, and a confidence interval plot that shows envelopes for the confidence intervals of the summary statistic.

The K plot is produced when you specify the K option in the PROCESS statement. If you specify PLOTS=K without specifying the K option in the PROCESS statement, then PLOTS=K is ignored. You can specify the following option:

UNPACK

suppresses paneling of the K function plots and produces each constituent plot separately.

L < (UNPACK) >

produces a panel of the L function, which is a transformation of the K function. The panel contains four plots: an EDF plot with simulation envelopes for CSR, an EDF-CSR difference plot, a PP plot that compares the EDF of the summary statistic, and a confidence interval plot that shows envelopes for the confidence intervals of the summary statistic.

The L plot is produced when you specify the L option in the PROCESS statement. If you specify the PLOTS=L option without requesting the L option in the PROCESS statement, then PLOTS=L is ignored. You can specify the following option:

UNPACK

suppresses paneling of the L function plots and produces each constituent plot separately.

LURKING < (*lurking-plot-options*) >

requests lurking variable plots, which show the cumulative raw residual with respect to the covariates or the coordinate variables or both. By default, PROC SPP computes lurking variable panel plots with respect to both covariates and coordinates. You can specify the following *lurking-plot-options*:

ALL

creates lurking variable plots of the model's covariates and of the coordinate variables that are specified in the PROCESS statement.

COORD

creates lurking variable plots only of the coordinate variables that are specified in the PROCESS statement.

COVAR

creates lurking variable plots only of the covariates and does not create plots with respect to the coordinate variables X and Y.

UNPACK
unpacks the lurking variable panel plots into individual lurking variable plots.

The default is LURKING(ALL).

NONE
suppresses all plots.

OBSERVATIONS < (*observations-plot-option*) >
OBSERV < (*observations-plot-option*) >
OBS < (*observations-plot-option*) >
produces the observed data plot. You can specify the following *observations-plot-options*:

ATTR=EVENT | MARK
specifies the observations attribute that you want to plot. You can specify the following values:

EVENT
specifies a plot of the locations of the point-pattern event observations.

MARK
specifies a plot of the locations and the mark values of the point-pattern event observations. If you do not specify OBS(MARK) or if the analysis skips the specified mark variable, then the observations plot request is ignored.

PCF < (UNPACK) >
produces a combined plot of the pair correlation function, g. The combined plot shows both the confidence intervals for the summary statistic and the simulation envelope for comparison with CSR.

The PCF plot is produced when you specify the PCF option in the PROCESS statement. If you specify the PLOTS=PCF option without specifying the PCF option in the PROCESS statement, then PLOTS=PCF is ignored. You can specify the following option:

UNPACK
suppresses the combination of different PCF plots into a single plot and produces each constituent plot separately.

RESIDUAL < (*residual-plot-options*) >
produces a plot of the residual diagnostics. By default, the SPP procedure produces a panel plot that contains smoothed raw residuals, raw residuals, and lurking variable plots with respect to the X and Y coordinates. In addition, you can specify the following *residual-plot-options*:

TYPE=CUM | RES
specifies the type of residual to be plotted in the lurking variable plots of the coordinate variables. You can specify the following values:

CUM	plots the cumulative residual
RES	plots a noncumulative residual as a scatter plot.

UNPACK
unpacks the panel plot, which contains smoothed raw residuals, raw residuals, and a lurking variable plot, into four separate plots.

SEED=*seed-value*
specifies the seed to use for the random number generator. The SEED= value has to be an integer.

TRENDS
produces a plot of all trend covariates. This option is ignored if no trend covariates are specified in the TREND statement.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SPP to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SPP procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

COVTEST Statement

COVTEST *process-name* **=** *trend-name* < *trend-name, . . .* >< */options* > **;**

You use the COVTEST statement to perform covariate dependency tests that are based on an empirical distribution function (EDF). The COVTEST statement contains two essential parts: a *process-name* (which must be declared in a PROCESS statement that precedes the COVTEST statement) that you specify on the left side and a list of *trend-names* (which must be defined in a preceding TREND statement within the same PROC SPP call) that you specify on the right side. The procedure performs separate EDF tests for every *trend-name* that is specified in the right side of the COVTEST statement. When you include a trend on the right side of the COVTEST statement, PROC SPP performs a weighted EDF test. Performing EDF tests involves computing EDF statistics, for which the SPP procedure calculates the Kolmogorov-Smirnov D

statistic by default. PROC SPP also produces an EDF plot for the Kolmogorov-Smirnov D statistic, which you can request by using the TEST=D option. If you are interested instead in the Cramér–von Mises W^2 statistic, you need to request it via the following *covtest-option*:

TEST=CM | D

requests the test statistics for any type of requested weighted tests. This option applies only if you have specified a trend on the right side of the COVTEST statement. The requested weighted test statistic is applied to every trend that is specified on the right side of the COVTEST statement. You can specify the following values:

D requests the Kolmogorov-Smirnov D statistic.

CM requests the Cramér–von Mises W^2 statistic.

By default, TEST=D.

MODEL Statement

MODEL *process-name* **=** < *trend-name, . . .* > < */model-options* > **;**

The MODEL statement enables you to fit an inhomogeneous Poisson process model. You must specify a *process-name* as the dependent variable. In addition, the MODEL statement enables you to specify multiple trends as covariates. If you do not specify any trends as covariates in the MODEL statement, PROC SPP fits a second-degree polynomial. The *process-name* must be defined in a preceding PROCESS statement, and each *trend-name* must be defined in a preceding TREND statement. Table 114.2 summarizes the *model-options* that you can specify.

Table 114.2 MODEL Statement Options

Option	Description
CENSCALE	Displays optimization centering and scaling information
CORRB	Requests the approximate correlation matrix
COVB	Requests the approximate covariance matrix
CL	Constructs a t-type confidence interval
GOF	Performs a chi-square-based goodness-of-fit test
GRID	Specifies the intensity response GRID size
ITHIST	Requests the optimization iteration history
MTYPE	Requests a specific type of model to be fit
OUTINTENSITY	Specifies an output data set to store the intensity estimates
OUTSIM	Specifies an output data set to store the simulations from an intensity model
POLYNOMIAL	Requests an additional polynomial component to be included in the model fitting process
RESIDUAL	Requests residual computations and specifies the bandwidth for smoothed residuals
SOLUTION	Requests display of raw results

You can specify the following *model-options*:

CENSCALE
lists the centering and scaling (standardization) information for each coordinate and covariate in the model.

CL< (*alpha-value*) >
requests a *t*-type confidence interval for the estimated parameters. You can also specify the significance level via the *alpha-value*. The default *alpha-value* is 0.05, which corresponds to the default confidence level of 95%.

CORRB
requests the estimated correlation matrix for the parameter estimates. To request the estimated correlation matrix for the model parameters with respect to the standardized covariates, specify both this *model-option* and the SOLUTION *model-option*.

COVB
requests the estimated covariance matrix for the parameter estimates. To request the estimated covariance matrix for the model parameters with respect to the standardized covariates, specify this *model-option* and the SOLUTION *model-option*.

GOF(*num-simulations*)
requests a goodness-of-fit test for the fitted intensity model. You can specify the number of Monte Carlo simulation runs as an integer in *num-simulations*. By default, the SPP procedure performs 100 simulations when you specify this option. It is recommended that you specify a QUADRAT option in the definition of the response/dependent point pattern in the PROCESS statement. If you do not specify such an option, the SPP procedure uses a default 10×10 quadrat.

GRID(*value-NX*,*value-NY*)
specifies the grid resolution for model fitting, where *value-NX* specifies the number of grids in the horizontal direction and *value-NY* specifies the number of grids in the vertical direction. By default, the SPP procedure fits the model on a 128×128 grid.

ITHIST< (PARM) >
requests an iteration history table for the model-fitting optimization. Specify this option to produce additional levels of output detail. You can specify the following value:

PARM
includes the fitting parameters in the iteration history table.

MTYPE=POISSON | NEGBINOMIAL
specifies the type of inhomogeneous intensity model to be fit by PROC SPP. You fit a negative binomial model only in order to diagnose overdispersion, so in this case no fitted intensity is produced, and likewise none of the goodness-of-fit tests or residual diagnostics that are based on the intensity are produced. You can specify the following values:

POISSON fits a Poisson process model.

NEGBINOMIAL fits a negative binomial model.

By default, MTYPE=POISSON.

OUTINTENSITY=*SAS-data-set*
specifies a *SAS-data-set* in which to store the output intensity estimate.

OUTSIM<(*iter-value***)>=***SAS-data-set*
specifies a *SAS-data-set* in which to store a simulated point pattern from a fitted intensity model. Specify the number of iterations in < *iter-value* > to generate multiple point pattern data sets. By default, the number of simulation iterations is set to 1.

POLYNOMIAL|POLY<(*degree* **)>**
specifies a polynomial trend in the coordinates. You can also specify the *degree* of the polynomial component. If you do not specify the degree, PROC SPP procedure uses a second-degree polynomial by default.

RESIDUAL(B=*value***)**
requests residual diagnostics for the inhomogeneous Poisson process model. If you specify this option, you must also specify the residual bandwidth for computing smoothed residuals via the B= suboption.

SOLUTION
displays the parameter estimates table in a location- and scale-standardized space. For optimization purposes, any polynomial coordinates and covariates in the model are centered and scaled. The parameters and the approximate covariance and the correlation matrices are displayed by default in the untransformed, unstandardized space. This option causes the output to be displayed on the basis of the actual fitted parameters in the transformed space. If you also specify the COVB or CORRB *model-option* (or both), then PROC SPP also displays the estimated covariance or correlation matrix, respectively (or both), in the transformed space.

You can specify additional options that are related to the nonlinear optimization aspects of the MODEL fitting process via the NLOPTIONS statement.

NLOPTIONS Statement

NLOPTIONS < *options* > **;**

The NLOPTIONS statement specifies details about the nonlinear optimization technique that PROC SPP uses to maximize the log-likelihood function for the first-order intensity model. By default, PROC SPP uses the Newton-Raphson with ridging optimization technique. For more information about the NLOPTIONS statement, see the section "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics."

PARMS Statement

PARMS *value-list* < / **PARMSDATA**=*SAS-data-set* > **;**

The PARMS statement specifies initial values for the parameters in the MODEL statement. Alternatively, the PARMS statement can request a grid search over several values of these parameters. The PARMS statement is optional and must follow the associated MODEL statement.

Table 114.3 PARMS Statement Options

Option	Description
Component Options	
PARMSDATA=	Specifies an input data set that contains initial values for the model parameters

Specification of parameter values in the PARMS statement is ordered, but the order is unrelated to the order in which you specify covariates in the MODEL statement. In particular, you must specify the initial parameter values by starting with the intercept parameter. Depending on the terms you specify in the model, you must continue sequentially by specifying the initial values for each of the monomials in a polynomial, and finally specify the coefficients that correspond to plain covariate terms in the model. If you have no initial value for one or more of the model parameters, then you can specify missing values as initial values. You can specify the *value-list* in any of following forms:

m	a single value
$m_1, m_2, \ldots, m_n$	several values
m to n	a sequence in which m equals the starting value, n equals the ending value, and the increment equals 1
m to n by i	a sequence in which m equals the starting value, n equals the ending value, and i equals the increment
m_1, m_2 to m_3	mixed values and sequences

For example, suppose you are fitting an intensity model that consists of a polynomial of first degree in each of the coordinates x and y and a term with the covariate variable Elevation. You want to specify an initial value of –3.5 for the intercept, and an initial value of –5 for the covariate Elevation. In the PARMS statement, you specify initial values for the Intercept parameter and the parameter of the Elevation variable, and no initial values for the parameters of the polynomial terms x, xy, and y. The following SAS statements implement these specifications:

```
proc spp data=sashelp.bei plots(equate) = intensity;
   process trees = (x,y /area=(0,0,1000,500) event=Trees);
   trend grad = field(x,y,gradient);
   trend elev = field(x,y,elevation);
   model trees = elev / grid(50,25) poly(1);
   parms (-3.5) (.) (-5);
run;
```

If you specify more than one set of initial values, a grid of initial values sets is created. PROC SPP searches among the specified sets for the set that yields the lowest objective function value. Then, the procedure uses the initial values in the selected set for the optimization.

The results from the PARMS statement are the values of the parameters on the specified grid.

You can specify the following option after a slash (/) in the PARMS statement:

PARMSDATA=*SAS-data-set*

PDATA=*SAS-data-set*

specifies the SAS data set from which to read model parameter values. The data set should contain the values in the sequence that is required by the PARMS statement in either of the following two ways:

- Specify one single column under the variable Estimate (Est) that contains all the parameter values.
- Use one column for each parameter, and place the *n* columns under the Parm1–Parm*n* variables.

For example, the following two data sets are equivalent ways to specify initial values for a model that requires four parameters:

```
data parData1;
   input Estimate @@;
   datalines;
0.5 -2 0.03 -3.4
;
```

```
 data parData2;
   input Parm1 Parm2 Parm3 Parm4 Parm5 Parm6 Parm7;
   datalines;
0.5 -2 0.3 . . 1 -5
0.5 -2 0.1 . . 0.1 -5
;
```

You can specify more than one set of initial values in the *SAS-data-set*. PROC SPP seeks among the specified sets for the one that gives the lowest objective function value. Then, the procedure uses the initial values in the selected set for the fitting optimization.

You can either explicitly specify initial parameter values in the PARMS statement or use the PDATA= option, but you cannot use both at the same time.

PROCESS Statement

PROCESS *name* **=** (*variables* < /*pattern-options* >)< /*process-options* < *distance-function-options* > > **;**

The PROCESS statement defines a point pattern for analysis. You must use a valid SAS variable *name* to define the process, and you can describe it by using *variables* that contain the x and y coordinates of the points within the point pattern. The *variables* must also be in the DATA= data set. You can specify only one PROCESS statement in PROC SPP.

The coordinates in spatial data can be spherical (represented as longitude and latitude) or projected (represented as Cartesian x and y coordinates). All the SAS/STAT procedures that analyze spatial data, including PROC SPP, assume that you are working with projected coordinates, for which Euclidean distance is appropriate. If your data consist of spherical coordinates, you are responsible for transforming the data to projected coordinates, such as by using PROC GPROJECT in SAS/GRAPH software. For more information about the spatial modeling issues that pertain to the use of geodetic versus simple Euclidean distance, see Banerjee (2005).

You can also specify *pattern-options* and *process-options*. The *pattern-options* are related to different attributes of the observed point pattern that is read from the DATA= data set. The *process-options* represent different analyses that are associated with a point pattern. These analyses are usually helpful in characterizing the underlying stochastic process that might have generated the point pattern. The PROCESS statement's *pattern-options* are listed in Table 114.4. The PROCESS statement's *process-options* are listed in Table 114.5.

Table 114.4 Point Pattern Definition Options

Option	Description
AREA=	Specifies a rectangular study window
EVENT=	Specifies an EVENT variable that identifies individual point pattern events
MARK=	Specifies the MARK variable for the point pattern

You can specify the following *pattern-options*, which enable you to describe various aspects of a point pattern data set:

AREA=(*xmin-number, ymin-number, xmax-number, ymax-number***)**
specifies parameters that define the study area bounds for the spatial point pattern. This option describes is a key attribute that governs the intensity estimates that are obtained by different methods in PROC SPP. When you specify this option, you must identify all the following area specifications:

- *xmin-number*, the lower left limit for the x coordinate
- *ymin-number*, the lower left limit for the y coordinate
- *xmax-number*, the upper right limit for the x coordinate
- *ymax-number*, the upper right limit for the y coordinate

If there are BY groups in the DATA= data set, then the explicit bounds remain the same across all BY groups. If you do not specify this option, then PROC SPP estimates a default area based on the Ripley-Rasson window estimator. For more information about the Ripley-Rasson window estimate, see the section "Ripley-Rasson Window Estimator" on page 9604.

EVENT=*variable-name*
specifies an event variable that is associated with instances (points) in this point pattern. If your DATA= data set also contains information about covariates, use this option to identify the events in the point pattern.

MARK=*variable-name*
specifies a character or quantitative variable from the DATA= data set as a mark variable. Character variable marks are used for requesting distance function summary statistics across different variable values.

Table 114.5 PROCESS Statement Options

Option	Description
F	Computes the empty-space F function
G	Computes the G function
J	Computes the J function
K	Computes the K function to test for complete spatial randomness (CSR)
KERNEL	Obtains a nonparametric intensity estimate of the point pattern
L	Computes the L function
OUTSIM	Specifies an output data set to store the simulated data sets in computation of distance functions
PCF	Computes the PCF function
QUADRAT	Performs a quadrat-based test for CSR

You can specify the following *process-options* to study the point pattern data set and the underlying spatial point process that is likely to have generated this pattern:

F< GRID(*value-NX***,***value-NY***) >**
performs a test for complete spatial randomness that is based on the empty-space F function. For more information about the F function and related functions see the section "Statistics Based on Second-Order Characteristics" on page 9600. You can specify the following suboption:

GRID(*value-NX***, *value-NY***)**
specifies a reference grid for computing the empty-space F function, where *value-NX* represents the number of horizontal divisions and *value-NY* represents the number of vertical divisions. By default, the SPP procedure uses a 50×50 grid.

G
performs a test for complete spatial randomness that is based on the nearest-neighbor G function.

J< GRID(*value-NX***, *value-NY***) >**
performs a test for complete spatial randomness that is based on the J function. You can specify the following suboption:

GRID(*value-NX*, *value-NY*)
specifies a reference grid for computing the J function, where *value-NX* represents the number of horizontal divisions and *value-NY* represents the number of vertical divisions. By default, the SPP procedure uses a 50×50 grid.

K
performs a test for complete spatial randomness that is based on the K function.

KERNEL< (*kernel-suboptions*) >
produces a nonparametric estimate of the first-order intensity, or a nonparametric smoothed estimate of a quantitative mark variable of the point pattern, depending on the *kernel-suboptions*. When you do not specify the *kernel-suboptions*, PROC SPP computes a nonparametric intensity estimate that is based on a default bandwidth and uses a Gaussian kernel. You can specify the following *kernel-suboptions*.

TYPE=EPANECHNIKOV | GAUSSIAN | QUARTIC | TRIANGULAR | UNIFORM
specifies the kernel type for obtaining the nonparametric estimate. For more information about the different kernel types that PROC SPP supports, see the section "Nonparametric Intensity Estimation" on page 9606. By default, TYPE=GAUSSIAN.

B=*value*
specifies the *value* for the kernel bandwidth parameter. The bandwidth is a nonnegative number. By default, the SPP procedure uses a bandwidth of $0.1/\sqrt{\lambda}$, where λ is the CSR average intensity of the point pattern (Illian et al. 2008, p. 236).

ADAPTIVE
performs adaptive kernel estimation. Adaptive kernel estimation requires an initial bandwidth value to compute bandwidth estimates for each data point. If you specify a bandwidth in the B= *kernel-suboption*, then the SPP procedure uses this value as the initial bandwidth. Otherwise, it uses a default bandwidth value that is based on the suggestion by Illian et al. (2008, p.236). For more information about adaptive kernel estimation, see the section "Nonparametric Intensity Estimation" on page 9606.

OUT=*SAS-data-set*
specifies the name of a *SAS-data-set* to contain the kernel based nonparametric estimates.

GRID(*value-NX*, *value-NY*)
specifies a reference grid for computing the kernel estimate, where *value-NX* represents the number of horizontal divisions and *value-NY* represents the number of vertical divisions. By default, the SPP procedure uses a 50×50 grid.

L
performs a test for complete spatial randomness that is based on the L function.

OUTSIM=*SAS-data-set*
specifies the name of a *SAS-data-set* to contain the results of simulations in distance functions. This option is ignored unless one of the distance functions is specified in the PROCESS statement.

PCF< B=*value* >

performs a test for complete spatial randomness that is based on the pair correlation function (PCF) function. The pair correlation function is calculated only when you specify EDGECORR=ON in the PROC SPP statement. You can specify the following suboption:

B=*value*

specifies the bandwidth value to use in the kernel density estimation inside the pair correlation function. The *value* must be a nonnegative real number. Otherwise, it is assigned a default value of $0.1/\sqrt{\lambda}$, where λ is the CSR average intensity of the point pattern or of the current categorical mark type (Illian et al. 2008, p. 236).

QUADRAT< (< *value-NX*,*value-NY* > < /DETAILS >) >

performs a test for complete spatial randomness. You can specify *value-NX* and *value-NY* to provide a quadrat specification that includes the number of horizontal and vertical divisions. If you do not specify the number of horizontal and vertical divisions, PROC SPP computes a default quadrat of 10×10. By default, the QUADRAT option displays only the Pearson chi-square test for CSR. If you also specify the DETAILS suboption, then PROC SPP displays the quadrat count in addition to the Pearson residual information.

When you specify an F, G, J, K, L, or PCF *process-option* (shown in Table 114.5), you can also specify the *distance-function-options* shown in Table 114.6.

Table 114.6 Distance Function Options

Option	Description
BYTYPE	Requests categorical mark typewise calculation of distance functions
CROSS	Requests cross-type distance function analysis that is based on the categorical mark that is specified in the MARK= option
MAXDIST=	Specifies the ending distance for distance functions
MINDIST=	Specifies the starting distance for distance functions
NDIST=	Specifies the number of distances to use for different distance functions
NSIM=	Specifies the number of simulations to compute the CSR envelope
BLOCKS	Specifies the block size for calculation of confidence intervals for distance functions

BYTYPE(ALL|*value-list*)

requests distance function calculation by values of the mark variable. This option produces individual distance function calculations for each mark type. You can specify the following options:

ALL

requests distance function calculation for all available character mark variable values in the DATA= data set.

value-list
requests distance function calculation for certain formatted mark variable values, which you specify as quoted strings in the *value-list*.

CROSS=TYPES(*value-list1***<,***value-list2***>)**
requests cross-type distance function analysis between different mark values. For cross-type analysis, you must specify a mark variable in the point pattern definition by using the MARK= *pattern-option*. The CROSS= option applies only to any requested distance functions K, L, G, J, or PCF. You must specify the TYPES suboption as follows:

TYPES(*value-list1***<,***value-list2***>)**
requests cross-type analysis only among types that are specified in *value-list1* and an optional *value-list2*. If you specify only *value-list1*, then PROC SPP performs cross-type analysis within all the types that are specified in *value-list1*. If you also specify the additional *value-list2*, PROC SPP performs cross-type analysis across both lists. For *value-list1* and *value-list2*, specify quoted strings that correspond to values of the variable that is specified in the MARK=*pattern-option*.

MAXDIST=*value* **| MAX | CUT**
specifies the option to be used for computing the maximum distance for different distance functions. You can specify the following options:

value
specifies a *value* for the maximum distance for performing distance function calculations. The *value* must be positive and larger than the value of the MINDIST=*value* option. You can specify any positive value for the maximum distance. However, values that are too large might produce artifacts that do not reflect the true underlying process.

MAX
uses the maximum possible distance, based on the suggestion by Baddeley and Turner (2013). The maximum possible distance is calculated as follows:

- For the K and L functions, the maximum possible distance is calculated as

  $$\min\{\min\{\mathrm{Range}(x), \mathrm{Range}(y)\}/4, \sqrt{1000/(\pi \times \lambda)}\}$$

  where λ is the intensity of the point pattern in the study area and the ranges of x and y are computed over the minimum bounding rectangular window of the study area.
- For the PCF functions, the maximum possible distance is calculated as in the case of K and L functions except that the ranges of x and y are computed over a block division of the study area and the λ corresponds to the intensity in a block division. The computed maximum distance for the PCF distance is the minimum of the maximum distance computed over all the block divisions in the study area.
- For the F and G functions, the maximum possible distance is calculated as

  $$\min\{\mathrm{Diameter}(W)/2, \sqrt{\log(100000)/(\pi \times \lambda)}\}$$

  where λ is the intensity of the point pattern in the study area and W is the minimum bounding rectangular window of the study area.
- For the J function, the maximum possible distance is calculated as $\mathrm{Diameter}(W)/2$.

CUT

uses the maximum distance at certain cutoff values that are recommended by Baddeley (2014). The cutoff values are as follows:

- for the F and G functions, the distance at which the F or G value reaches 0.9
- for the J function, the distance at which the F or G value in the calculation of the J function reaches 0.9
- for the PCF function, the distance that corresponds to the MAX option that is applied to individual subdivisions of the study area for computing the confidence interval of the PCF statistic
- for the K and L functions, the distance that corresponds to theMAX option that is applied to the entire study area

By default, PROC SPP uses the value of MAXDIST is CUT.

MINDIST=*value*

specifies a positive number for the minimum distance (or starting) distance for all distance function calculations. The *value* of this option cannot be more than the value of MAXDIST= option.

NDIST=*value*

specifies the number of distance bins with which to compute all the specified distance functions. This is a global option that applies to all specified distance functions. When you specify a *value* for this option, the SPP procedure uses this *value* instead of others for distance function calculations.

NSIM=*value*

specifies a positive integer for the number of simulations to be used to compute envelopes for the CSR tests in all distance functions. When you specify this option, it applies to all specified distance functions.

BLOCKS(*NX***,** *NY***)**

specifies the block size that is required for calculating the confidence intervals of distance functions, where *NX* specifies the number of horizontal blocks and *NY* specifies the number of vertical blocks. The block size should be neither too small nor too large for this option to behave reasonably. For more information about estimating the confidence intervals for distance functions see the section "Confidence Intervals for Summary Statistics" on page 9604. The default block size is 5×5.

TREND Statement

TREND *name* **= FIELD***(field-definition)* **;**

The TREND statement enables you to define a spatial trend covariate, where *name* is a standard SAS variable name that names the trend and the FIELD suboption describes the field as follows:

FIELD (*X-variable***,** *Y-variable***,** *field-variable* **)**

specifies a spatial field variable as a trend by using any spatial field covariates that are available in the DATA= data set, where *X-variable* specifies the X coordinate and *Y-variable* specifies the Y coordinate of the spatial field. The third argument is the *field-variable*, which is a numeric variable in the DATA= data set. The *X-variable* and *Y-variable* should be the same as the ones in the PROCESS statement. If

you specify a different *X-variable* and a different *Y-variable* from the ones specified in the PROCESS statement then, PROC SPP will produce an error.

Details: SPP Procedure

Testing for Complete Spatial Randomness

The homogeneous Poisson point process serves as a reference model for a completely spatially random (CSR) point pattern. A homogeneous Poisson point process that has intensity $\lambda > 0$ has the following properties:

- The number of points $N(X \cap W)$ that fall in any region W has a Poisson distribution whose mean is $\lambda \times |W|$, where $|W|$ denotes the area of W.
- If W_1 and W_2 are disjoint sets, then $N(X \cap W_1)$ and $N(X \cap W_2)$ are independent random variables.
- The $N(X \cap W)$ points within a study area W are independent and uniformly distributed.

Quadrat Count Test for CSR

The quadrat test is a test of complete spatial randomness (CSR) that uses the χ^2 statistic based on quadrat counts. In the quadrat test, the study area window W is divided into subregions called quadrats (W_1,W_2,..., W_m) of equal area. The test counts the number of points that fall in each quadrat $n_j = n(X \cap W_j)$ for $j = 1, ldots, m$. Under the null hypothesis of CSR, the n_j are iid Poisson random variables. The following Pearson χ^2 test statistic assesses whether there is a departure from the homogeneous poisson process:

$$\chi^2 = \frac{\sum_j (n_j - n/m)}{n/m}$$

A significant p-value indicates that the underlying point pattern is not CSR.

Exploring Interpoint Interaction

A common question that arises while exploring point pattern data sets is whether points are distributed independent of each other or whether there exists some kind of interaction between points. There are two broad categories of summary statistics, which are based on distances between points:

- nearest-neighbor statistics, such as the F, G, and J functions
- statistics that are based on second-order characteristics, such as the K, L, and g functions (Illian et al. 2008).

The following subsections discuss these statistics in detail.

Nearest-Neighbor Distance Functions

The SPP procedure implements the following nearest-neighbor distance functions:

- empty-space F function
- nearest-neighbor G function
- J function

A typical test that uses any nearest-neighbor function compares the empirical distribution function with the corresponding function for a homogeneous Poisson process that has first-order intensity λ. Usually, the first-order intensity is obtained as the number of observations per unit of area, $\hat{\lambda} = n(x)/|W|$.

The empty-space F function is defined as the empirical distribution function of the observed empty-space distances, $d(g, x)$, which is measured from a set of reference grid points g to the nearest point in the point pattern. The empty-space distance can be defined as

$$d(g, x) = \min\{||g - x_i||, \text{for } x_i \in x\}$$

In practice, the computation of the empty-space F function also involves an edge correction. The edge-corrected empty-space F function is defined as

$$\hat{F}(r) = \sum_j e(g_j, r)\mathbf{1}\{d(g_j, x) \leq r\}$$

where $e(g_j, r)$ is an edge correction. PROC SPP implements the border edge correction (Illian et al. 2008, p. 185–186) as described in the section "Border Edge Correction for Distance Functions" on page 9603.

For a homogeneous Poisson process that has first-order intensity λ, the F function is

$$F_P(r) = 1 - \exp(-\lambda\pi r^2)$$

You compare the empirical and Poisson empty-space F function by using the EDF and the P-P plot in the F function summary panel plot. Values of $\hat{F}(r) > F_P(r)$ suggest a regularly spaced pattern, and values of $\hat{F}(r) < F_P(r)$ suggest a clustered pattern (Baddeley and Turner 2005).

The nearest-neighbor G function is the empirical distribution of the observed nearest-neighbor distance of the points within the point pattern. In practice, the G function also involves an edge correction and is defined as

$$\hat{G}(r) = \sum_i e(x_i, r)\mathbf{1}\{d_i \leq r\}$$

where $e(x_i, r)$ is the border edge correction (Illian et al. 2008, p. 185-186) as described in the section "Border Edge Correction for Distance Functions" on page 9603 and d_i is the distance to the nearest neighbor for the ith point.

For a homogeneous Poisson process that has first-order intensity λ, the G function can be defined as

$$G_P(r) = 1 - \exp(-\lambda\pi r^2)$$

The interpretation of $\hat{G}(r)$ is opposite to the interpretation of $\hat{F}(r)$. That is, values of $\hat{G}(r) > G_P(r)$ imply a clustered pattern, and values of $\hat{G}(r) < G_P(r)$ suggest a regular pattern (Baddeley and Turner 2005).

The third type of nearest-neighbor distance function is the J function, which is defined as a combination of both the F and G functions (Baddeley et al. 2000). The J function is defined for all distances r such that $F(r) < 1$. The J function can be defined as

$$J(r) = \frac{1 - G(r)}{1 - F(r)}$$

For a homogeneous Poisson process, $J_P(r) = 1$. When $J(r)$ takes values greater than 1, regularity is indicated; when $J(r)$ takes values less than 1, the underlying process is more clustered than expected. As can be seen from the expression of $J(r)$, the estimate is an uncorrected estimate of the J-function and hence its computation does not require an edge correction (Baddeley et al. 2000).

Statistics Based on Second-Order Characteristics

Statistics that are based on second-order characteristics include Ripley's K function, Besag's L function, and the pair correlation function (also called the g function). To understand why these functions are based on second-order characteristics, see Illian et al. (2008, p. 223-243). These functions usually involve computation of pairwise distances between points.

The K function of a stationary point process is defined such that $\lambda K(r)$ is the expected number of points within a distance of r from an arbitrary point of the process. The empirical K function of a set of points is the weighted and renormalized empirical distribution function of the set of pairwise distances between points. The empirical K function can be written as

$$\hat{K}(r) = \frac{1}{\hat{\lambda}^2 |W|} \sum_i \sum_{j \neq i} \mathbf{1}\{||x_i - x_j|| \leq r\} e(x_i, x_j; r)$$

where $e(x_i, x_j; r)$ is the border edge correction that is described in the section "Border Edge Correction for Distance Functions" on page 9603.

For a homogeneous Poisson process, $K_P(r)$ can be written as

$$K_P(r) = \pi r^2$$

Exploratory analysis usually involves computing both the empirical K function, $\hat{K}(r)$, and the K function for a Poisson process, $K_P(r)$. A comparison of $\hat{K(r)}$ and $K_P(r)$ might indicate clustering or regularity depending on whether $\hat{K}(r) > K_P(r)$ or $\hat{K}(r) < K_P(r)$.

Besag's L function is a transformation of the K function and is defined as

$$L(r) = \sqrt{\frac{K(r)}{\pi}}$$

For a homogeneous Poisson process, $L_P(r) = r$.

The pair correlation function, *g(r)*, can also be expressed as a transformation of the K function:

$$g(r) = \frac{K'(r)}{2\pi r}$$

Illian et al. (2008), Stoyan (1987), and Fiksel (1988) suggest an alternative expression for $g(r)$:

$$g(r) = \rho(r)/\lambda^2$$

where $\rho(r)$ is the second-order product density function. Cressie and Collins (2001) provides an expression for $\rho(r)$ as

$$\rho(r) = \frac{\hat{\lambda}^2 K'(r)}{2\pi r}$$

where $\hat{\lambda}^2 K'(r)$ can be written as a kernel estimate,

$$\hat{\lambda}^2 K'(r) = \frac{1}{a} \sum_{i=1}^{n} \sum_{j \neq i} k_h(||x_i - x_j|| - r)$$

where a is the area, $k_h(u) = k(u/h)/h$, and $k(.)$ is a kernel such as the uniform kernel or the Epanechnikov kernel (Silverman 1986). PROC SPP uses the version that is based on the uniform kernel; for more information about the uniform kernel, see the section "Nonparametric Intensity Estimation" on page 9606. Based on the formula for the second-order product density $\rho(r)$ in terms of the kernel estimate, Stoyan (1987) gives an edge-corrected kernel estimate for $\rho(r)$ as

$$\rho(r) = \frac{1}{2\pi r} \sum_{i} \sum_{j \neq i} \frac{k_h(||x_i - x_j|| - r)}{a(W_{x_i} \cap W_{x_j})}$$

Dividing $\rho(r)$ by $\hat{\lambda}^2$ gives the pair correlation function $g(r)$ as

$$g(r) = \frac{1}{2\pi r \hat{\lambda}^2} \sum_{i} \sum_{j \neq i} \frac{k_h(||x_i - x_j|| - r)}{a(W_{x_i} \cap W_{x_j})}$$

where W_{x_i} indicates the translation of the study area window W by the distance x_i from its origin. The above expression for $g(r)$ was given by Stoyan and Stoyan (1994) using the translation edge correction.

A border-edge-corrected version of $g(r)$ can be written as

$$g(r) = \frac{1}{2\pi r \hat{\lambda}} \frac{\sum_i \sum_{j \neq i} k_h(||x_i - x_j|| - r)}{\sum_i \mathbf{1}\{b_i \geq r\}}$$

where x_i and x_j are points within the boundary at a distance greater than or equal to r; where b_i is the distance of x_i to the boundary of W, ∂W; and where $k_h(u) = k(u/h)/h$ for a kernel $k(.)$, such as the uniform kernel or the Epanechnikov kernel. For more information about the uniform kernel, see the section "Nonparametric Intensity Estimation" on page 9606. For a homogeneous Poisson process, $g(r) = 1$. For any point pattern, values of $g(r)$ greater than 1 indicate clustering or attraction at distance r, whereas values of $g(r)$ less than 1 indicate regularity.

Distance Functions for Multitype Point Patterns

Distance functions (such as G, J, K, L, and g) can also be defined for point patterns that are "marked" with a categorical mark variable, called a type. Usually you consider mark variables that have more than one type to define distance functions. When distance functions are defined between two types, they are called cross-type distance functions. For any pair of types i and j, the cross-distance functions G_{ij}, J_{ij}, K_{ij}, L_{ij}, and g_{ij} can be defined analogously to the single-type distance functions. The interpretation of cross-type distance functions is slightly different from the interpretation of single type functions. Suppose that X is the point pattern, X_j refers to the subpattern of points of type j; X_i refers to the subpattern of points of type i, and λ_j represents the intensity of the subpattern X_j. Then the interpretation is to treat X_j as a homogeneous Poisson process and independent of X_i. If the computed empirical cross-type function is identical to the function that corresponds to a homogeneous Poisson process, then X_i and X_j can be treated as independent of each other.

The empirical cross-G-function, G_{ij}, is defined as the distribution of the distance from a point of type i in X_i to the nearest point of type j in X_j. Formally, G_{ij} can be written as

$$G_{ij}(r) = \sum_i e(x_i, r)\mathbf{1}\{d_{ij} \le r\}$$

where $e(x_i, r)$ is an edge correction and d_{ij} is the distance from a point of type i to the nearest point of type j. If the two subpatterns X_i and X_j are independent of each other, then the theoretical cross-G-function is

$$G^*_{ij}(r) = 1 - \exp(\lambda_j \pi r^2)$$

The empirical cross-type J-function, J_{ij}, can be defined again in terms of the G_{ij} function and the empty-space F function for subpattern X_j as

$$J_{ij}(r) = J_{ij}(r) = \frac{1 - G_{ij}(r)}{1 - F_j(r)}$$

where $F_j(r)$ is the empty-space function for the subpattern X_j. If the two subpatterns X_i and X_j are independent of each other, then the theoretical cross-J-function is $J^*_{ij}(r) = 1$.

The empirical cross-type K function, K_{ij}, is $1/\lambda_j$ times the expected number of points of type j within a distance r of a typical point of type i. Formally, K_{ij} can be written as

$$K_{ij}(r) = K_{ij}(r) = \frac{1}{\lambda_j \lambda_i |W|} \sum_i \sum_j \mathbf{1}\{||x_i - x_j|| \le r\} e(x_i, x_j; r)$$

where $e(x_i, x_j; r)$ is an edge correction. If the two subpatterns X_i and X_j are independent of each other, then the theoretical cross-K-function is $K^*_{ij}(r) = \pi r^2$.

The empirical cross-type L function, L_{ij}, is a transformation of K_{ij}. Formally, L_{ij} can be written as

$$L_{ij}(r) = L_{ij}(r) = \sqrt{\frac{K_{ij}(r)}{2\pi r}}$$

If the two subpatterns X_i and X_j are independent of each other, then the theoretical cross-type L-function is $L^*_{ij}(r) = r$.

The empirical cross-type pair correlation function, g_{ij}, is a kernel estimate of the form

$$g_{ij}(r) = \frac{\rho(r)}{\hat{\lambda}^2} = \frac{1}{2\pi r \hat{\lambda}_i \hat{\lambda}_j} \sum_i \sum_j \frac{k_h(||x_i - x_j|| - r)}{|W \cap W_{i-j}|}$$

Based on the definition of Stoyan and Stoyan (1994), $g_{ij}(r)$ can be written as

$$g_{ij}(r) = \frac{\rho(r)}{\hat{\lambda}^2} = \frac{1}{2\pi r \hat{\lambda}_i \hat{\lambda}_j} \sum_i \sum_j \frac{k_h(||x_i - x_j|| - r)}{|W_{x_i} \cap W_{x_j}|}$$

A border-edge-corrected version of $g_{ij}(r)$ can be written as

$$g_{ij}(r) = \frac{1}{2\pi r \hat{\lambda_j}} \frac{\sum_i \sum_j k_h(||x_i - x_j|| - r)}{\sum_i \mathbf{1}\{b_i \geq r\}}$$

where b_i is the distance of x_i to the boundary of W, which is denoted as ∂W. If the two subpatterns X_i and X_j are independent of each other, then the theoretical cross-type pair correlation function is $g^*_{ij}(r) = 1$.

Border Edge Correction for Distance Functions

To compute the edge correction factors $e(x_i, r)$ that appear in the formulas of the distance functions, the SPP procedure implements border edge correction (Illian et al. 2008; Ripley 1988; Baddeley 2007). Border edge correction is necessary because the data are given for a bounded observation window W, but the pattern itself is assumed to extend beyond the observation window. However, because you can observe only what is within the window, a disc $b(x, r)$ of radius r around a point x that lies close to the boundary of W might extend outside W. Because the original process X is not observed outside W, the number of points of X in $b(x, r)$ is not observable (Baddeley 2007). Ignoring the fact that the observable quantity $n(X \cap W \cap b(x, r))$ is less than or equal to $n(X \cap b(x, r))$ leads to a bias that is caused by edge effects. The border edge corrector is a simple strategy to eliminate the bias that is caused by edge effects. Under the border method, the window W is replaced by a reduced window,

$$W_{\ominus r} = W \ominus b(0, r) = \{x \in W : ||x - \partial W|| \geq r\}$$

where $||x - \partial W||$ denotes the minimum distance from X to a point on the boundary. The reduced window contains all the points in W that are at least r units away from the boundary ∂W.

Based on the preceding definition, the border edge corrected F, K, and G functions are

$$\hat{F}(r) = \frac{1}{\lambda |W_{\ominus r}|} \sum_{g_j \in W_{\ominus r}} \mathbf{1}\{d(g_j, x) \leq r\}$$

$$\hat{K}(r) = \frac{\sum_{i=1}^{n} \sum_{j \neq i} \mathbf{1}\{||x_i - x_j|| \leq r\}}{\hat{\beta} n(x \cap W_{\ominus r})}$$

$$\hat{G}(r) = \frac{\sum_{x_i \in W_{\ominus r}} \mathbf{1}\{||x_i - X/\ x_i|| \leq r\}}{n(X \cap W_{\ominus r})}$$

$$\hat{G}(r) = \frac{\sum_i \mathbf{1}\{d_i \leq r, b_i \geq r\}}{\sum_i \mathbf{1}\{b_i \geq r\}}$$

where $\hat{\beta} = n(x)/\lambda|W|$; $||x_i - X/\ x_i||$ is the observed nearest-neighbor distance, d_i, for the ith point x_i; and b_i is the distance from x_i to the boundary ∂W. For more information about these border-edge-corrected functions, see Baddeley (2007).

Confidence Intervals for Summary Statistics

The SPP procedure computes confidence intervals for the true value of a summary statistic such as the K, L, F, G, J, or PCF function. The window that contains the point pattern is divided into a number of blocks. By default, PROC SPP divides the window into 5×5 blocks. The summary statistic is calculated in each block, and the pointwise sample mean, sample variance, and sample standard deviation of these summary statistics are computed. If any edge corrections are required, they are also applied in the calculation of the individual summary statistics within each block. If the summary statistic is a function such as the K function, the estimate for a particular block B is computed by counting pairs of points in which the first point lies in B and the second point lies elsewhere (Baddeley and Turner 2013).

The variance of the summary statistics is estimated by

$$\mathrm{var}(\tilde{K}(t)) \approx \frac{\sum_{i=1}^{m}\{k_i - \tilde{K}(t)\}}{\{m(m-1)\}}$$

where m is the number of blocks, k_i is the value of the summary statistic in individual blocks, and $\tilde{K}(t) = \sum_{i=1}^{m} k_i/m$ (Diggle 2003, pp. 52–53).

Ripley-Rasson Window Estimator

When the sampling window for a point pattern is unknown, it can be estimated from the data. A common error is to assume that W is the smallest rectangle that contains the data points, or is the convex hull that encloses the data points (Baddeley 2010). Either choice is an underestimate of the true region W and usually yields an overestimate of the point process intensity λ or summary statistics such as the K function. The Ripley-Rasson window estimator is an estimate of the spatial window from which the points were drawn (Ripley and Rasson 1977). For estimating a rectangular study region, the Ripley-Rasson estimate is the rescaled copy of the minimum bounding box of the collection of points, centered at the box's centroid and expanded using a scaling factor of $1/\sqrt{1 - \frac{4}{n}}$, where n is the number of data points.

Covariate Dependence Tests

For analyses that include covariates, the SPP procedure implements nonparametric goodness-of-fit tests that are based on the empirical distribution function (EDF). PROC SPP provides weighted EDF tests that depend primarily on the covariate values.

The next subsection reviews the EDF tests that are at the heart of covariate dependency testing, and the subsequent subsection describes the covariate dependency tests in more detail.

EDF Goodness-of-Fit Tests

You use goodness-of-fit tests to examine the fit of a parametric distribution. In the SPP procedure, this task emerges when you test your data for dependence on a covariate. You can examine the goodness of fit by using tests that are based on the EDF. These tests offer advantages over traditional chi-square goodness-of-fit tests, as discussed in D'Agostino and Stephens (1986). The empirical distribution function is defined for a set of n independent observations, $X_1, \ldots, X_n$, that have a common distribution function $F(x)$ as follows. Denote the observations ordered from smallest to largest as $X_{(1)}, \ldots, X_{(n)}$. Then the empirical distribution function, $F_n(x)$, is

$$F_n(x) = \begin{cases} 0, & x < X_{(1)} \\ \frac{i}{n}, & X_{(i)} \le x < X_{(i+1)} \quad i = 1, \ldots, n-1 \\ 1, & X_{(n)} \le x \end{cases}$$

$F_n(x)$ is a step function that takes a step of height $\frac{1}{n}$ at each observation. This function estimates the distribution function $F(x)$. At any value x, $F_n(x)$ is the proportion of observations that are less than or equal to x, whereas $F(x)$ is the probability of an observation being less than or equal to x. EDF statistics measure the discrepancy between $F_n(x)$ and $F(x)$.

The computational formulas for the EDF statistics make use of the probability integral transformation $Z = F(X)$. If $F(X)$ is the true distribution function of X, then the random variable Z is uniformly distributed between 0 and 1. For example, assume that you believe $X \sim N(\mu, \sigma^2)$. In this case, the probability integral transform $Z = F(X)$ for the normal $N(\mu, \sigma^2)$ is given by the EDF of the standardized value $(X - \mu) / \sigma$. To test the fit of your sample EDF $F_n(x)$ to the assumed exact $F(X)$, you can equivalently test the fit of $F_n(z)$ to the EDF $F(Z)$ of Z. As $Z \sim U(0, 1)$, $F(Z)$ is the cumulative density function (CDF) of the standard uniform $U(0, 1)$, which is simply $F(Z) = z$. This also means that your empirical $F_n(x) = F_n(z)$. Consequently, the probability integral transform translates the initial fit task into an easier comparison between $F_n(z)$ and $F(Z)$.

There are two main classes of EDF statistics: the supremum and the quadratic class. The supremum class is based on the largest vertical difference between $F(x)$ and $F_n(x)$. The quadratic class is based on the squared difference $(F_n(x) - F(x))^2$. Quadratic statistics have the following general form:

$$Q = n \int_{-\infty}^{+\infty} (F_n(x) - F(x))^2 \psi(x) dF(x)$$

The function $\psi(x)$ weights the squared difference $(F_n(x) - F(x))^2$.

As previously discussed, the SPP procedure considers the ordered observations $X_{(1)}, \ldots, X_{(n)}$ and computes the values $Z_{(i)} = F(X_{(i)})$ by applying the probability integral transform. PROC SPP examines the goodness of fit by computing the following two EDF statistics:

- Kolmogorov-Smirnov two-sided D from the supremum class
- Cramér-von Mises W^2 from the quadratic class

Within the different classes of EDF statistics, the quadratic class is known to have more powerful statistics than the supremum class. The details of the statistics used by PROC SPP are discussed in the following subsection.

After the EDF test statistics are computed, the SPP procedure computes the associated significance values. In the scope of the PROC SPP analysis, the true distribution function, $F(X)$, is a completely specified distribution. For computations in this scenario, PROC SPP applies slightly modified D and W^2 statistics, as described by D'Agostino and Stephens (1986).

Testing Covariate Dependency with EDF Tests

In a test for covariate dependency, the goal is to test the null hypothesis H_0 that the point process is independent of the covariate. PROC SPP tests H_0 by interpolating the covariate values at the event locations. The EDF is weighted by the intensity at the corresponding locations (Baddeley and Turner 2005). PROC SPP performs this weighted EDF test for covariates that are defined in a TREND statement.

Weighted EDF Tests

To test dependence on a trend covariate, PROC SPP initially computes the covariate EDF. The EDF is weighted by using intensity-based weights to account for the current intensity model. For example, under the CSR assumption the intensity λ is constant across the study area; hence, the weight for each of the M observations of a covariate is $\lambda_i / \sum_i M\lambda_i = \lambda / M\lambda = 1/M$. This weighted EDF is the predicted distribution that any other set of independent covariate observations should follow under the assumed intensity model.

Next, the covariate is interpolated at the n event locations $\boldsymbol{s}_i, i = 1, \ldots, n$, using ordinary kriging; kriging analysis assumes a linear semivariance correlation function and considers the four closest covariate observations for each event location. The outcome is a set of covariate values X_i. With the X_i in hand, PROC SPP assumes that the probability integral transform $Z = F(X)$ is the linear interpolation of the weighted EDF at the covariate values X_i, and it produces the transformed EDF Z in [0, 1]. If the intensity model assumption is correct, then Z follows a uniform distribution $U(0, 1)$. Finally, PROC SPP uses EDF tests to examine the fit of the EDF $F_n(x) = F_n(z)$ to a standard uniform EDF.

Nonparametric Intensity Estimation

The KERNEL option in the PROCESS statement enables you to perform nonparametric intensity estimation. You can use five different kernel types: Epanechnikov, Gaussian, uniform, triangular, and quartic (Silverman 1986), whose kernel functions are as follows, where $t = \sqrt{(s_x - x)^2 + (s_y - y)^2}/h$, s_x, s_y are the grid point coordinates, x and y are the point coordinates, and h is the bandwidth parameter:

- Epanechnikov

$$K(t) = \begin{cases} \frac{3}{4}(1 - \frac{t^2}{5})\frac{1}{\sqrt{5}} & |t| < \sqrt{5} \\ 0 & \text{otherwise} \end{cases}$$

- Gaussian

$$K(t) = \frac{e^{-\frac{t^2}{2}}}{\sqrt{2\pi}}$$

- uniform

$$K(t) = \begin{cases} \frac{1}{2} & |t| < 1 \\ 0 & \text{otherwise} \end{cases}$$

- triangular

$$K(t) = \begin{cases} 1 - |t| & |t| < 1 \\ 0 & \text{otherwise} \end{cases}$$

- quartic

$$K(t) = \begin{cases} \frac{15}{16}(1 - t^2)^2 & |t| < 1 \\ 0 & \text{otherwise} \end{cases}$$

Given the preceding kernel definitions, the nonparametric intensity estimate can be computed as

$$\lambda(s) = \sum_{i=1}^{n} h^{-2} \times K\left(\frac{s - s_i}{h}\right)$$

where h is the fixed bandwidth. In practice, nonparametric intensity estimation also involves an edge correction. By default, PROC SPP divides the nonparametric estimate $\lambda(s)$ by an edge correction factor

$$\rho(s) = \int_A h^{-2} \times K\left(\frac{s - s_i}{h}\right)$$

where A is the study area. The choice of the bandwidth parameter that nonparametric intensity estimation requires is more important than the choice of the kernel type itself (Silverman 1986). The bandwidth can be spatially fixed or spatial varying. If the bandwidth is spatially varying, it is called adaptive kernel estimation. For adaptive kernel estimation, the SPP procedure uses the technique suggested in Silverman (1986, p. 101) and Diggle, Rowlingson, and Su (2005, p. 426), which is computed in two steps:

1. Use an initial bandwidth h to compute pilot estimates of the first-order intensity as

   $$\lambda_0(s) = \sum_{i=1}^{n} h^{-2} \times K\left(\frac{s - s_i}{h}\right)$$

   where *K(.)* is a kernel.

2. Compute bandwidth factors as

   $$h_i = h \times \left(\frac{\lambda_0(s)}{\widehat{g}}\right)^{-0.5}$$

   where $\widehat{g}$ is the geometric mean of the pilot estimates $\lambda_0(s)$.

Based on the computed bandwidth estimates, h_i, the nonparametric intensity estimates are computed as

$$\lambda(s) = \sum_{i=1}^{n} h_i^{-2} \times K\left(\frac{s - s_i}{h_i}\right)$$

In PROC SPP, adaptive kernel estimation does not incorporate edge correction.

Inhomogeneous Poisson Process Model Fitting

An inhomogeneous Poisson process that has intensity function $\lambda(s)$ is a point process in which the number of points that fall in a spatial region W, $N(X \cap W)$ has the following expectation:

$$\mathbb{E}[N(X \cap W)] = \int_W \lambda(s)ds$$

Also, the $N(X \cap W)$ points are independent and identically distributed for disjoint subsets W with a probability density of

$$f(s) = \frac{\lambda(s)}{\int_W \lambda(s)ds}$$

.

Likelihood Methods for Model Fitting

The intensity function $\lambda_\theta(s)$ is assumed to be log linear in the parameters θ. So

$$\log \lambda_\theta(s) = \theta \cdot Z(s)$$

where $Z(s)$ is a real-valued or vector-valued function of location s. $Z(s)$ can include a polynomial function of coordinate variables or a spatial covariate. The log likelihood for the parameters θ is given by

$$\log L(\theta; x) = \sum_{i=1}^{n} \log \lambda_\theta(x_i) - \int_W \lambda_\theta(s)ds$$

The integral in the expression for the log likelihood can be approximated using quadrature as

$$\int_W \lambda_\theta(s; x)ds \approx \sum_{j=1}^{m} \lambda_\theta(s_j; x)w_j$$

for some quadrature weights w_j. Hence, the log likelihood can be rewritten as follows:

$$\log L(\theta; x) \approx \sum_{i=1}^{n(x)} \log \lambda_\theta(x_i; x) - \sum_{j=1}^{m} \lambda_\theta(s_j; x)w_j$$

Based on the observation by Berman and Turner (1992) and Baddeley and Turner (2000), the log likelihood can be approximated as

$$\log L(\theta; x) \approx \sum_{j=1}^{m} (y_j \log \lambda_j - \lambda_j)w_j$$

where $\lambda_j = \lambda_\theta(s_j)$. If the list of points $\{s_j, j = 1, \ldots, m\}$ also includes the collection of data points $\{x_i, i = 1, \ldots, n\}$, then $y_j = z_j/w_j$ and

$$z_j = \begin{cases} 1 & \text{if } s_j \text{ is a data point, } s_j \in x \\ 0 & \text{if } s_j \text{ is a dummy point, } s_j \notin x \end{cases}$$

The log pseudolikelihood can be maximized using standard optimization algorithms.

Fit Statistics

The SPP procedure displays three fit statistics for model selection. For a model that has p parameters, uses n event observations, and produces a maximum log likelihood Log L, these criteria are calculated as in Table 114.7.

Table 114.7 Fit Statistics

Option	Description
–2 log likelihood	$2\text{LL} = -2\ \text{Log L}$
Akaike's information criterion (AIC)	$\text{AIC} = -2\ \text{Log L} + 2p$
Schwarz criterion or Bayesian information criterion (BIC)	$\text{BIC} = -2\ \text{Log L} + p\log(n)$

The AIC and BIC statistics give two different ways of adjusting the –2 Log L statistic for the number of terms in the model and the number of observations used. These statistics can be used when different models for the same data are compared. Lower values of the statistics indicate a more desirable model.

Fitted Model Validation That Uses Goodness-of-Fit Tests

If you want to check how likely the data are to be generated by the fitted model, you can perform a goodness-of-fit test that is based on a chi-square statistic. The model goodness-of-fit test that is displayed by the GOF option in the MODEL statement uses quadrats to compute the observed and expected counts and subsequently to perform the chi-square test. The model goodness-of-fit test is a simulation-based test that uses the fitted model to generate different realizations of the point process. For each simulated realization, the SPP procedure calculates the expected count under the model and computes the mean of this expected count over all the realizations. The mean of this expected count over all realizations is used to compute a Pearson residual as

$$\text{Pearson residual} = \frac{O_c - E_c}{\sqrt{E_c}}$$

where O_c is the observed count in each quadrat, based on the data, and E_c is the expected count under the model. Based on these observed and expected counts, a chi-square statistic is computed and a Pearson chi-square test is performed. A small p-value indicates that the data are not likely to be generated by the model.

Fitted Model Validation That Uses Residuals

Residual diagnostics are tools for checking and examining the fitted model. Residual plots and influence diagnostics help you identify influential observations, assess model assumptions, and recognize departures from the model. Baddeley et al. (2005) define four types of residuals: raw residuals, inverse residuals, Pearson residuals, and score residuals. PROC SPP implements only raw residuals. Given a point pattern x and using a parameter estimate $\hat{\theta} = \hat{\theta}(x)$, the raw residuals can be defined as

$$R_{\hat{\theta}}(W) = n(x \cap W) - \int_W \hat{\lambda}(s, x)ds$$

In order to be able to compute the raw residual, Baddeley et al. (2005) suggest a discretization of this residual measure. According to Baddeley and Turner (2013), discretization of the raw residuals yields

$$r_j = z_j - w_j \lambda_j$$

at the quadrature points u_j, where z_j is an indicator equal to 1 if u_j is a data point or 0 if u_j is a dummy point, w_j is the quadrature weight that is attached to u_j, and $\lambda_j = \hat{\lambda}(u_j, x)$ is the conditional intensity of the fitted model at u_j.

Smoothed Residuals

The smoothed raw residuals are defined as

$$s(u) = \hat{\lambda}(u) - \tilde{\lambda}(u)$$

where $\hat{\lambda}(u)$ is a nonparametric kernel estimate of the intensity,

$$\hat{\lambda}(u) = e(u) \sum_{i=1}^{n(x)} k(u - x_i)$$

where $e(u)$ is an edge correction and $\tilde{\lambda}(u)$ is a smoothed version of the parametric estimate of the intensity according to the fitted model:

$$\tilde{\lambda}(u) = e(u) \int_W k(u - s) \lambda_{\hat{\theta}}(s) ds$$

If the fitted model is correct, the kernel estimate and the kernel smoothed estimate of the fitted intensity should be approximately equal. Positive values of $s(u)$ suggest that the model underestimates the intensity (Baddeley and Turner 2005).

Lurking Variable Plots

Lurking variable plots help detect dependence on an unobserved covariate. Any systematic pattern in these plots indicate a departure from the model (Baddeley and Turner 2005). For point process models, you can plot the residuals against a spatial covariate or one of the coordinates to investigate the presence of a spatial trend and to assess whether the true trend differs from the trend that is specified by the fitted model. For a spatial covariate $Z(u)$ that is defined at each location $u \in W$, the residual on each sublevel set,

$$W(z) = \{u \in W : Z(u) \leq z\}$$

yields a cumulative residual function for the raw residuals as follows:

$$A(z) = n(\{x \cap W(z)\}) - \int_{W(z)} \hat{\lambda}(u, x) du$$

In addition to plotting the cumulative residual function, the lurking variable plot also shows 2σ limits based on the variance of the innovations under an inhomogeneous Poisson process (Baddeley et al. 2005). The variance of the innovations under an inhomogeneous Poisson process is

$$\mathrm{var}\{A(z)\} = \mathrm{var}\{I\{W(z)\}\} = \int_{W(z)} \lambda(u)du$$

The 2σ limits can be interpreted as pointwise significance limits. A systematic violation of the limits suggests that the proposed model does not account for the dependence on the covariate under consideration (Baddeley et al. 2005).

Negative Binomial Modeling

Spatial data in many applications can be overdispersed—that is, the variance of the counts of spatial events might be inflated relative to what is expected under the assumption of a Poisson model. The negative binomial modeling in PROC SPP serves as a diagnostic to assess overdispersion in your data.

The log likelihood for the negative binomial model is

$$\log L(\theta; x) = \sum_{i=1}^{n} \left\{ y_i \cdot \log\left\{\frac{\phi \cdot \lambda_i}{w_i}\right\} - (y_i + \frac{w_i}{\phi}) \cdot \log\left\{1 + \frac{\phi \cdot \lambda_i}{w_i}\right\} + \log\left\{\frac{\Gamma(y_i + w_i/\phi)}{\Gamma(w_i/\phi)\Gamma(y_i + 1)}\right\}\right\}$$

where y_i is the response at a location, λ_i is the intensity associated with the location, ϕ is the negative binomial scale parameter, and w_i is the quadrature weight associated with the location. Overdispersion is indicated by the value of the scale parameter ϕ: for $\phi = 0$, the negative binomial distribution is identical to the Poisson distribution; therefore, large values of ϕ indicate overdispersion.

Because the negative binomial model in PROC SPP is intended only for diagnostic purposes, the only results are the estimated parameters themselves, including ϕ. No fitted intensity is produced, and likewise nothing that depends on the fitted intensity, such as the goodness-of-fit tests and the residual diagnostics, is produced.

Output Data Sets

The SPP procedure produces output data sets that are specified in the OUT= suboption of the KERNEL option in the PROCESS statement, and in the OUTINTESITY= and OUTSIM= options in the MODEL statement. These data sets are described in the following sections.

OUT= Suboption in the KERNEL Option in the PROCESS Statement

This suboption specifies the name of an output data set to store the kernel-based nonparametric estimates. This data set contains the following variables:

- KERNEL, the kernel type that is used for the corresponding intensity estimate
- BANDW, the kernel bandwidth that is used for the corresponding intensity estimate
- VARNAME, the label of the event variable
- GXC, the x coordinate of the grid point at which the intensity estimate is made
- GYC, the y coordinate of the grid point at which the intensity estimate is made
- ESTIMATE, the intensity estimate

OUTSIM= Option in the PROCESS Statement

This option specifies the name of an output data set to store the simulations of distance functions that are included as options in the PROCESS statement. This data set contains the following variables:

- GXC, X coordinate of current simulated event location
- GYC, Y coordinate of current simulated event location
- ITER, current iteration
- IXY, simulated event observation number in the current iteration
- MARK, label of current mark
- MTYPE, categorical mark level
- MVALUE, numeric mark simulated value
- NITER, number of iterations in simulation
- NXY, number of simulated events in current iteration
- VARNAME, label of the event the results refer to

OUTINTENSITY= Option in the MODEL Statement

This option specifies the name of an output data set to store the output intensity estimate. This data set contains the following variables:

- ESTIMATE, fitted intensity at current location
- GXC, X coordinate of current output grid location
- GYC, Y coordinate of current output grid location
- STDERR, standard error for the intensity estimate
- VARNAME, event label from the DATA= data sets that are associated with events and identified during the procedure call
- PROB, probability of occurrence based on a logit transformation of the fitted intensity estimate at the grid. This represents the probability that the number of events at the location is not zero.

OUTSIM= Option in the MODEL Statement

This option specifies the name of an output data set to store a simulated point pattern from a fitted intensity model. This data set contains the following variables:

- ESTIMATE, fitted intensity estimate at the simulated event location
- GXC, X coordinate of current simulated event location
- GYC, Y coordinate of current simulated event location
- ITER, current simulation iteration
- NITER, number of iterations in current simulation
- NXY, number of simulated events in current iteration
- VARNAME, event label from the DATA= data set that is associated with events and identified during the procedure call

Displayed Output

The SPP procedure produces the following output objects.

- By default, PROC SPP outputs a "Number of Observations" table, which displays the number of observations that are read from the input data set and the number of valid observations that are used. The actual number of observations that are used in the analysis can be equal to or smaller than the number of valid observations, depending on the specification of the study window and the existence and handling of duplicate observations. When you include a covariate variable in your analysis, this table contains more detailed information about the number of observations that are used in the study window for each variable.
- If you use the PROCESS statement to specify a point pattern, PROC SPP outputs a table is displayed by default that contains exploratory information about the point pattern, any mark variable that is present, and information about the point pattern domain window and grid.
- If you use the PROCESS statement to specify a point pattern, PROC SPP outputs a default plot of the event observations in the point pattern.
- If you specify a mark variable for the point pattern, PROC SPP outputs a table that contains information about the mark variable.
- If you do not specify any options for the PROCESS statement, PROC SPP performs the quadrat test by default and outputs a table that shows the Pearson chi-square test for CSR by default. If you specify the QUADRAT option with the DETAILS suboption, PROC SPP outputs a detailed quadrat counts table, a quadrat information table that contains Pearson residuals, and a table for the Pearson chi-square test for CSR.

- If you specify a KERNEL option in the PROCESS statement, PROC SPP outputs a kernel intensity information table. In addition, if you request the ADAPTIVE suboption in the KERNEL option, an adaptive kernel information table is displayed. In addition, if you specify the KERNEL option and ODS Graphics is enabled, a map of the kernel intensity estimate is also produced.

- If you specify one or more of the K, L, G, and F options in the PROCESS statement, PROC SPP outputs an information table for each of the specified distance functions. The information table contains basic information, such as the minimum analysis distance, maximum analysis distance, maximum difference between the empirical distribution function of the summary statistic and the CSR function, and the distance at which the maximum difference is observed. In addition, PROC SPP outputs a panel plot for each distance function that is included as an option in the PROCESS statement. Each panel plot contains four constituent plots: the empirical distribution function (EDF) plot, the EDF−CSR difference plot, a probability-probability plot that compares the EDF and CSR, and a confidence interval plot for the summary statistic.

- If you specify the J and PCF options in the PROCESS statement, PROC SPP outputs a combined plot that shows the EDF, the simulation intervals, and the confidence interval for the summary statistic.

- If you specify a COVTEST statement that has appropriate trend covariates on the right side, PROC SPP outputs a table for the Kolmogorov-Smirnov EDF test statistic and creates a plot of the empirical and transformed EDF by default for each covariate that you include in the COVTEST statement. The plot illustrates the Kolmogorov-Smirnov test analysis for testing for point pattern dependency on covariates. If you specify the Cramér–von Mises EDF test statistic in the TEST= option in the COVTEST statement, PROC SPP outputs the table for the Cramér–von Mises EDF test statistic.

- If you specify a MODEL statement to fit a model for the first-order intensity of the point pattern that is defined in a preceding PROCESS statement, PROC SPP produces the following results by default:

  - a “Model Information” table that lists the intercept, covariates, and polynomial terms that are included in the model, along with the initial values for the coefficients
  - an optimization information table that shows the optimization technique, the number of parameters in the optimization, and the number of fixed parameters and starting values
  - a table for the convergence status that shows the convergence criterion
  - a “Parameter Estimates” table that shows the estimate for each parameter, the standard error, the number of degrees of freedom, a t value, and a p-value
  - a “Fit Statistics” table that shows different fit statistics, such as the log likelihood, Akaike’s information criterion, and the Bayesian information criterion
  - a map that shows the fitted intensity estimate based on the model

- If you specify the MODEL statement and include the ITHIST option, PROC SPP outputs an iteration history table that shows the value of the objective function and the maximum value of the gradient over different iterations of the optimization algorithm.

- If you specify the MODEL statement and include the CORRB option, PROC SPP outputs the approximate correlation matrix.

- If you specify the MODEL statement and include the COVB option, PROC SPP outputs the approximate covariance matrix.

- If you specify the MODEL statement and include the GOF option, PROC SPP outputs a table that shows the Pearson chi-square test for goodness of fit. This table shows a *p*-value that indicates how likely it is for the data to be generated by the fitted model. In addition, if you also specify a QUADRAT option with the DETAILS suboption for the response process in a preceding PROCESS statement, PROC SPP also displays a quadrat information table that shows Pearson residuals that are based on the expected counts under the fitted model and observed counts from the point pattern data set that is defined for the response process.

The complete listing of the PROC SPP output follows in the sections "ODS Table Names" on page 9615 and "ODS Graph Names" on page 9617.

ODS Table Names

Each table that PROC SPP creates has a name that is associated with it, and you must use this name to refer to the table when you use ODS Graphics. Table 114.8 lists these names and shows the statement and options that you must specify to produce the table.

Table 114.8 ODS Tables Produced by PROC SPP

ODS Table Name	Description	Statement	Option
CenScale	Model parameter standardization information	MODEL	CENSCALE
CenScaleCorrB	Approximate correlation matrix of model-standardized parameter estimates	MODEL	CORRB, SOLUTION
CenScaleCovB	Approximate covariance matrix of model-standardized parameter estimates	MODEL	COVB, SOLUTION
CenScaleParms	Parameter estimates for standardized output	MODEL	SOLUTION
ConvergenceStatus	Status of optimization at conclusion	MODEL	Default output
CorrB	Approximate correlation matrix of model parameter estimates	MODEL	CORRB
CovariateInfo	Numerical covariate information	COVTEST	Default output
CovariateLevelInfo	Levels of categorical covariate	COVTEST	Default output
CovB	Approximate covariance matrix of model parameter estimates	MODEL	COVB
EdfCsrTest	EDF test for complete spatial randomness	COVTEST	Default output
ExploratoryInfo	General point pattern information	PROC	Default output
FitStatistics	Goodness-of-fit information	MODEL	Default output
IterHist	Iteration history	MODEL	Default output
KernIntensityInfo	Intensity function information from kernel density estimation	PROCESS	KERNEL

Table 114.8 *continued*

ODS Table Name	Description	Required Statement	Option
MarkInfo	Numerical mark information	PROC SPP	Default output
MarkLevelInfo	Levels of categorical mark	PROC SPP	Default output
ModelInfo	Model information	MODEL	Default output
ModelPearsonsChiSq	Chi-square goodness-of-fit test	MODEL	GOF
ModelQuadratInfo	Detailed fitted model quadrat information	MODEL	GOF
ModelSimulationInfo	Information table for simulating a point pattern from a fitted model	MODEL	OUTSIM
NObs	Number of observations read and used	PROC SPP	Default output
OptInfo	Optimization information	MODEL	Default output
ParameterEstimates	Model-fitting solution and statistics	MODEL	Default output
ParmSearch	Parameter search values	PARMS	Default output
PearsonsChiSq	Chi-square test for CSR	PROCESS	QUADRAT
QuadratCount	Counts of quadrats	PROCESS	QUADRAT
QuadratInfo	Detailed quadrat information	PROCESS	QUADRAT/DETAILS
FFuncInfo	F function information	PROCESS	F
GFuncInfo	G function information	PROCESS	G
KFuncInfo	K function information	PROCESS	K
LFuncInfo	L function information	PROCESS	L
SummaryInfo	Summary statistics information	PROCESS	F, G, K, L, PCF, J

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by using the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS." For additional control of the graphics that are displayed, see the PLOTS= option in the section "PROC SPP Statement" on page 9579.

ODS Graph Names

PROC SPP assigns a name to each graph that it creates by using ODS Graphics. You can use these names to refer to the graphs when you use ODS Graphics. Table 114.9 lists the names and shows the statement and option that you must specify to produce the graph.

Table 114.9 Graphs Produced by PROC SPP

ODS Graph Name	Plot Description	Statement	Option
CovariateEDFPlot	Plot of Kolmogorov-Smirnov test analysis for CSR	COVTEST	PLOTS=CSRKSTEST
EmptySpacePlot	Surface plot of nearest-neighbor distances from any window location	PROCESS	
		PROC SPP	PLOTS=EMPTYSPACE
FCIPlot	Confidence interval plot of empty-space function F	PROCESS	F
		PROC SPP	PLOTS=F(UNPACK)
FDiffPlot	Difference plot of empty-space function F and CSR	PROCESS	F
		PROC SPP	PLOTS=F(UNPACK)
FEdfPlot	EDF plot of empty-space function F	PROCESS	F
		PROC SPP	PLOTS=F(UNPACK)
FNppPlot	PP plot of empty-space function F and CSR	PROCESS	F
		PROC SPP	PLOTS=F(UNPACK)
FPanelPlot	Panel plot of empty-space function F	PROCESS	F
		PROC SPP	PLOTS=F(ALL)
GCIPlot	CI plot of nearest-neighbor function G	PROCESS	G
		PROC SPP	PLOTS=G(UNPACK)
GDiffPlot	Difference plot of nearest-neighbor function G and CSR	PROCESS	G
		PROC SPP	PLOTS=G(UNPACK)
GEdfPlot	EDF plot of nearest-neighbor function G	PROCESS	G
		PROC SPP	PLOTS=G(UNPACK)
GNppPlot	PP plot of nearest-neighbor function G and CSR	PROCESS	G
		PROC SPP	PLOTS=G(UNPACK)
GPanelPlot	Plot of nearest-neighbor function G	PROCESS	G
		PROC SPP	PLOTS=G(ALL)
IntensityPlot	Surface plot of estimated intensity	Default	Default
JCIPlot	CI plot of function J	PROCESS	J
		PROC SPP	PLOTS=J(UNPACK)

Table 114.9 *continued*

ODS Graph Name	Plot Description	Statement	Option
JCombinedPlot	Plot of function J	PROCESS	J
		PROC SPP	PLOTS=J(ALL)
JEdfPlot	EDF plot of function J	PROCESS	J
		PROC SPP	PLOTS=J(UNPACK)
KCIPlot	CI plot of Ripley's function K	PROCESS	K
		PROC SPP	PLOTS=K(UNPACK)
KDiffPlot	Difference plot of Ripley's function K and CSR	PROCESS	K
		PROC SPP	PLOTS=K(UNPACK)
KEdfPlot	EDF plot of Ripley's function K	PROCESS	K
		PROC SPP	PLOTS=K(UNPACK)
KNppPlot	PP plot of Ripley's function K and CSR	PROCESS	K
		PROC SPP	PLOTS=K(UNPACK)
KPanelPlot	Plot of Ripley's function K	PROCESS	K
		PROC SPP	PLOTS=K(ALL)
LCIPlot	CI plot of Besag's L function	PROCESS	L
		PROC SPP	PLOTS=L(UNPACK)
LDiffPlot	Difference plot of Besag's L function and CSR	PROCESS	L
		PROC SPP	PLOTS=L(UNPACK)
LEdfPlot	EDF plot of Besag's L function	PROCESS	L
		PROC SPP	PLOTS=L(UNPACK)
LNppPlot	PP plot of Besag's L function	PROCESS	L
		PROC SPP	PLOTS=L(UNPACK)
LPanelPlot	Plot of Besag's L function	PROCESS	L
		PROC SPP	PLOTS=L(ALL)
LurkingPanel	Cumulative residual and lurking variable panel plot	MODEL	
		PROC SPP	PLOTS=LURKING(ALL)
LurkingVariable	Cumulative residual and lurking variable plot	MODEL	
		PROC SPP	PLOTS=LURKING(UNPACK)
ObservationsPlot	Scatter plot of observed events, marked events, or covariate variables	PROC SPP	PLOTS=OBSERV
PCFCIPlot	CI plot of the pair correlation function, g	PROCESS	PCF
		PROC SPP	PLOTS=PCF(UNPACK)

Table 114.9 *continued*

ODS Graph Name	Plot Description	Statement	Option
PCFCombinedPlot	Plot of the pair correlation function, *g*	PROCESS	PCF
		PROC SPP	PLOTS=PCF(ALL)
PCFEdfPlot	EDF plot of the pair correlation function, *g*	PROCESS	PCF
		PROC SPP	PLOTS=PCF(UNPACK)
ProbabilityPlot	Surface plot of occurrence probability under a poisson model	Default	Default
RawResidual	Raw residual plot	MODEL	RESIDUAL
		PROC SPP	PLOTS=RESIDUAL(UNPACK)
ResidualPanel	Raw residual panel plot	MODEL	RESIDUAL
		PROC SPP	PLOTS=RESIDUAL
ResidualScatter	Cumulative residual plot	MODEL	RESIDUAL
		PROC SPP	PLOTS=RESIDUAL(UNPACK)
SmoothedResidual	Cumulative residual plot	MODEL	RESIDUAL
		PROC SPP	PLOTS=RESIDUAL(UNPACK)
TrendCovariatePlot	Trend covariate plot	TREND	PLOTS=CSRKSTEST

To request these graphs, you must specify the ODS GRAPHICS statement in addition to the statements indicated in Table 114.9. For more information about the ODS GRAPHICS statement, see Chapter 24, "Statistical Graphics Using ODS."

Examples: SPP Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 114.1: Exploration of a Multitype Point Pattern

(View the complete code for this example (sppex1.sas) in the example repository.)

This example demonstrates how you can use PROC SPP to explore a multitype point pattern. Consider the following data set, which consists of locations of retinal amacrine cells in a rabbit's eye. The data set contains three variables: X and Y are the coordinates of the cell locations, and Type is the type of each cell (which is based on whether it turns *on* or *off* when exposed to light). The data were originally analyzed by Diggle, Eglen, and Troy (2006) and Hughes (1985).

```
data amacrine;
   input X Y Type $ @@;
   label Type='Cell Type';
   datalines;
0.0224   0.0243   on
0.0243   0.1028   on
0.1626   0.1477   on
0.1215   0.0729   on
0.2411   0.0486   on
0.0766   0.1776   on
0.1047   0.2579   on

   ... more lines ...

1.5729   0.729    off
1.457    0.6822   off
1.4168   0.7374   off
;
```

Accounting for mark types enables you to study possible interaction across types. You can study such interactions by using the cross-type variants of distance functions. The following statements compute the cross-K-function between the types of amacrine cells:

```
proc spp data=amacrine edgecorr=on seed=1
         plots(equate)= (observations(attr=mark) K);
   process cells= (X,Y / area=(0,0,1.6,1) mark=Type)
                  / K cross=types('on' 'off');
run;
```

Output 114.1.1 shows two types of cells that are characterized by their state as *on* (in red) and *off* (in blue).

Output 114.1.1 Amacrine Cell Types

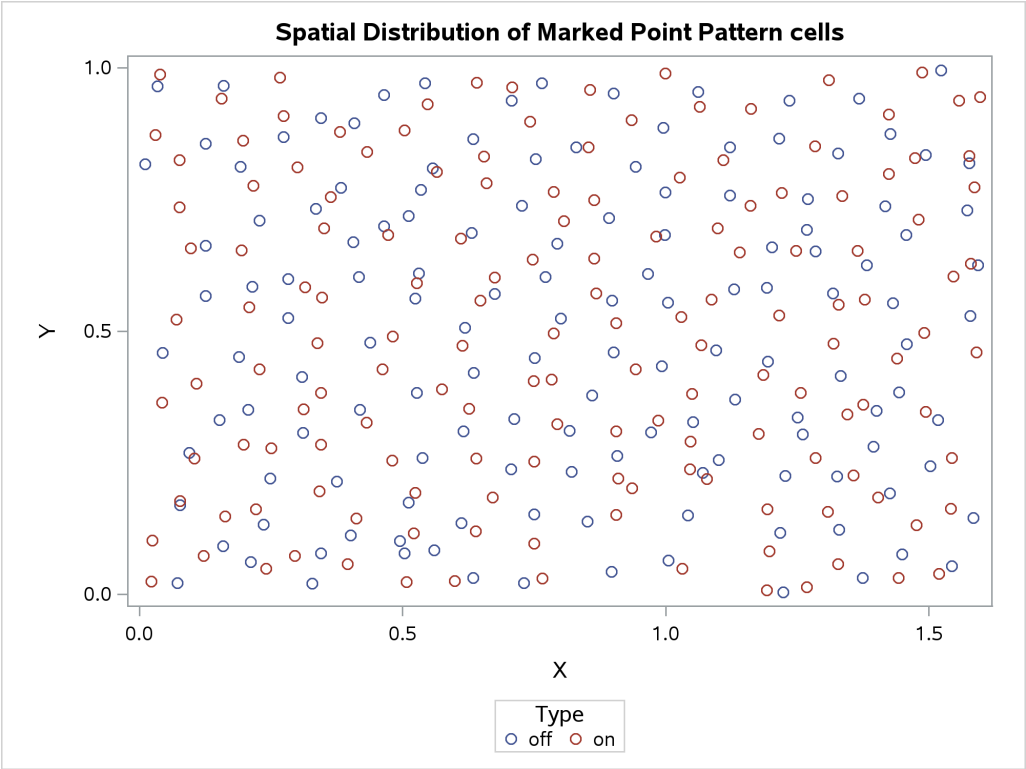

Output 114.1.2 lists exploratory information for the amacrine point pattern, and identifies it as a marked point pattern.

Output 114.1.2 Exploratory Information for the Amacrine Marked Point Pattern

Summary of Point Pattern	
Data Type	Marked Point Pattern
Pattern Name	cells
Region Type	User Defined Window
Region X Range	[0,1.6] Units
Region Y Range	[0,1] Units
Region X Size	1.6 Units
Region Y Size	1 Unit
Region Area	1.6 Square Units
Observations in Window	294
Average Intensity	183.75
Grid Nodes in X	50
Grid Nodes in Y	50
Grid Nodes in Window	2500
Mark	Type
Mark Type	Multitype

Output 114.1.3 shows the information for each mark type, including the frequency, percentage in the data set, and the first-order intensity, which measures the number of events of that particular mark type per unit of

area that is contained in the marked point pattern.

Output 114.1.3 Mark Information for the Amacrine Marked Point Pattern

Level Information for Type			
Value	**Frequency**	**Percentage**	**Intensity**
off	142	48.30%	88.7500
on	152	51.70%	95.0000

Output 114.1.4 shows the cross-K-function test to detect a clustering of points for which Type=*off* around points for which Type=*on* and vice versa. It is very clear from the plots in the top left corner and top right corner that there is no significant difference between the computed cross-K function and the theoretical cross-K function. This clearly indicates that there is no significant clustering of *on* amacrine cells around *off* amacrine cells.

Output 114.1.4 Cross-K Function Panel

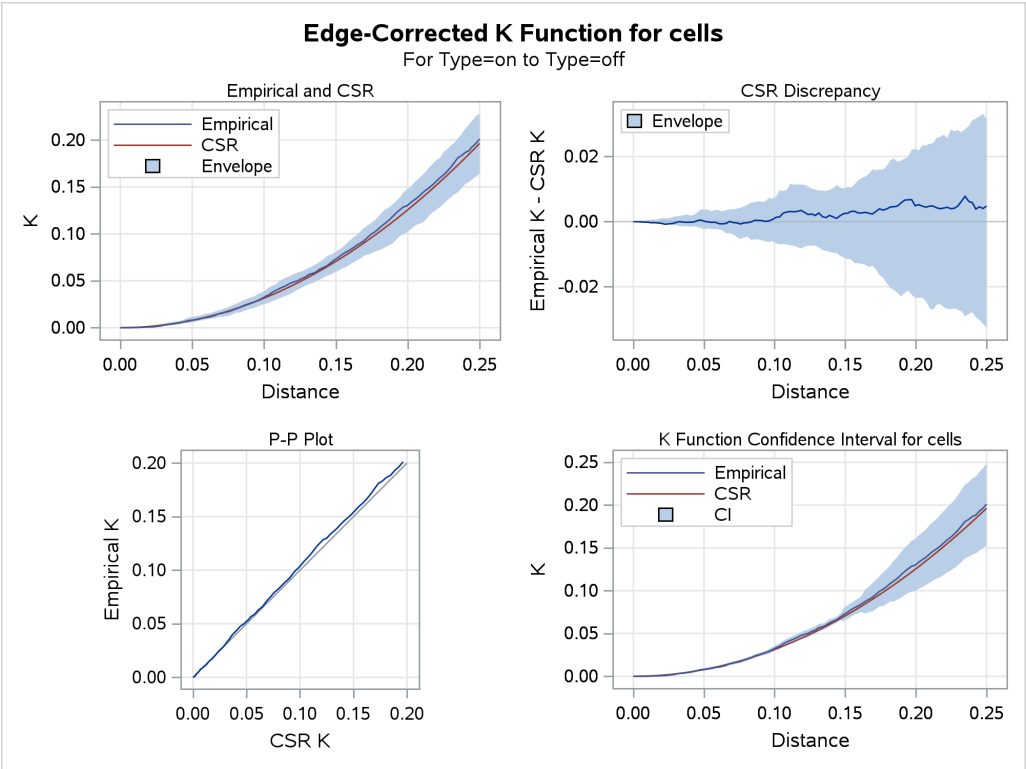

Example 114.2: Testing Covariate Dependence of a Point Pattern

(View the complete code for this example (sppex2.sas) in the example repository.)

In most spatial analysis applications, you are likely to have one or more covariates in addition to the point pattern data set. Hence, you can test for a possible dependency between the observed point pattern and the covariates by using covariate dependency tests that compute empirical distribution function (EDF) statistics. These tests are nonparametric, and the selected EDF statistic indicates whether the covariate values interpolated at the point locations are independent of the transformed covariate and a known model of point pattern intensity such as CSR. Covariate dependency testing serve multiple objectives:

- They tell you whether the empirical distribution of the covariates at the point locations and the empirical distribution function weighted and transformed according to the underlying intensity model (or the predicted distribution) are similar.
- In cases of dissimilarity between the empirical distribution and the predicted distribution, the interpretation is that the covariate gives evidence against the intensity model (CSR in this case).

To request a covariate dependency test that is based on an EDF statistic, you use the COVTEST statement, in which you specify the point process and the covariates that need to be tested. The following statements perform a covariate dependency test that is based on an EDF statistic:

```
proc spp data=sashelp.bei;
   process trees = (x,y /area=(0,0,1000,500) event=Trees);
   trend grad = field(x,y,Gradient);
   trend elev = field(x,y,Elevation);
   covtest trees = grad elev;
run;
```

When you do not request any specific EDF test statistic, by default the SPP procedure produces an EDF test that uses the Kolmogorov-Smirnov statistic for each covariate that is specified in the COVTEST statement, as shown in Output 114.2.1.

Output 114.2.1 Weighted EDF Test Statistic

The SPP Procedure

Kolmogorov-Smirnov Weighted EDF Test for Covariate Values		
Source	D Statistic	p Value
Gradient	0.194805	<0.0010
Elevation	0.106492	<0.0010

Output 114.2.1 reports the value of the Kolmogorov-Smirnov D test statistic and a p-value. The p-value indicates that the null hypothesis of independence on the covariate is rejected. In addition to the test statistic and the p-value, PROC SPP produces a plot of the empirical and transformed distribution function for each covariate. Output 114.2.2 and Output 114.2.3 show the plots of the Kolmogorov-Smirnov statistic for the covariates Gradient and Elevation, respectively. In each figure, the plot on the left includes the

empirical density function (EDF) of the variable (solid blue line) and the weighted EDF (transformed EDF or transformed line). The plot on the right is a PP plot that plots the Empirical probability (EDF) versus the Transformed probability(Transformed) against the reference standard uniform (red line). The same plot also shows the largest vertical difference between the normal and uniform lines, which is the Kolmogorov-Smirnov statistic D. From the right plots in Output 114.2.2 and Output 114.2.3, it is quite apparent that the EDF by Transformed line overlaps only at the ends for the Gradient covariate and crosses the uniform line once for the Elevation covariate. Thus, you can infer that the Gradient covariate (in addition to having a higher D statistic value) deviates considerably from the uniform line.

Output 114.2.2 Kolmogorov-Smirnov EDF Test Plot for Gradient

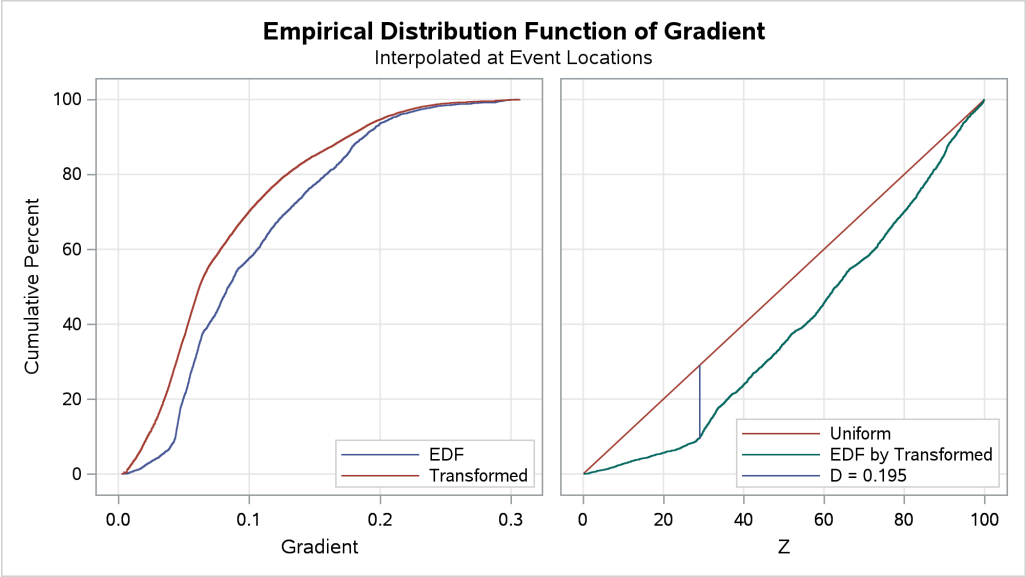

Output 114.2.3 Kolmogorov-Smirnov EDF Test Plot for Elevation

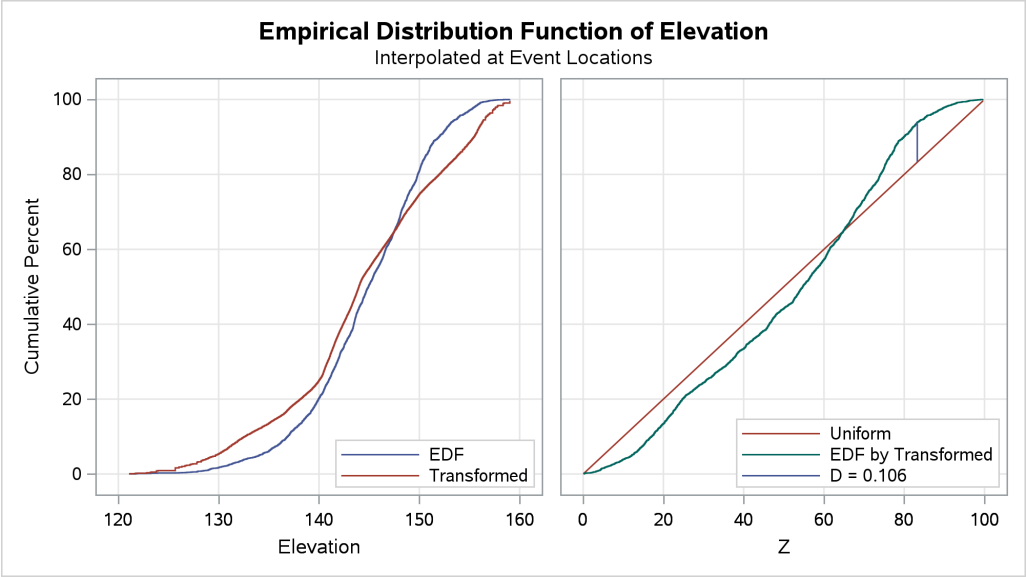

Example 114.3: Intensity Model Validation Diagnostics

(View the complete code for this example (sppex3.sas) in the example repository.)

Model validation diagnostics help you evaluate whether the fitted model that involves the covariates is appropriate for the specified point pattern. The SPP procedure provides two types of diagnostics:

- goodness-of-fit test
- residual diagnostics

This example demonstrates the usage of these diagnostics for model validation. It uses a simulated point pattern data set and also simulates two covariates over a 50×50 grid. The following statements define functions for simulating a spatial point pattern given an intensity function by the method of random thinning of Lewis and Shedler (1979), as discussed in Schabenberger and Gotway (2005) and Wicklin (2013). For more information about the method and the code, see Wicklin (2013). The functions are saved in a SAS/IML storage catalog to make them available for reuse.

```
ods graphics on;

proc iml;
   start Uniform2d(n, a, b);
      u = j(n, 2);
      call randgen(u, "Uniform");
      return( u # (a||b) );
   finish;

   start HomogPoissonProcess(lambda, a, b);
      n = 1;
      call randgen(n,"Poisson", lambda*2500);
      return( Uniform2d(n, a, b) );
   finish;

   start InhomogPoissonProcess(a, b) global(lambda0);
      u = HomogPoissonProcess(lambda0, a, b);
      lambda = Intensity(u[,1], u[,2]);
      r = shape(.,sum(lambda<=lambda0),1);
      call randgen(r,"Bernoulli", lambda[loc(lambda<=lambda0)]/lambda0);
      return( u[loc(r),] );
   finish;

   reset storage=sasuser.SPPThin;
   store module=Uniform2d
         module=HomogPoissonProcess
         module=InhomogPoissonProcess;
quit;
```

The following statements define a certain intensity function that is based on the elevation and slope of the land around particular hills, with hills characterized by a four-column matrix Hills, where the first two columns give the X and Y coordinates of each hill center and the last two columns give their height and radii. In the model, both elevation and slope are assumed to be positive, with a negative effect on intensity, so lambda0 (the maximum value of intensity) is the value at Elevation=Slope=0.

```
proc iml;
   %let xH = Hills[iHill,1];
   %let yH = Hills[iHill,2];
   %let hH = Hills[iHill,3];
   %let rH = Hills[iHill,4];

   start Elevation(x,y) global(Hills);
      Elevation = 0;
      do iHill = 1 to nrow(Hills);
         Height = &hH*exp(-((x - &xH)##2 + (y - &yH)##2)/&rH);
         Elevation = Elevation + Height;
      end;
      return(Elevation);
   finish;

   start Slope(x,y) global(Hills);
      xslope = 0;
      yslope = 0;
      do iHill = 1 to nrow(Hills);
         Height = &hH*exp(-((x - &xH)##2 + (y - &yH)##2)/&rH);
         dxHeight = -2*Height#(x - &xH)/&rH;
         dyHeight = -2*Height#(y - &yH)/&rH;
         xslope   = xslope + dxHeight;
         yslope   = yslope + dyHeight;
      end;
      Slope = sqrt(xslope##2 + yslope##2);
      return(Slope);
   finish;

   start Intensity(x,y) global(lambda0);
      lin = 0.5 - 2*Elevation(x,y) - 10*Slope(x,y);
      return(exp(lin));
   finish;

   lambda0 = exp(0.5);

   reset storage=sasuser.SPPFlowers;
   store lambda0 module=Elevation module=Slope module=Intensity;
quit;
```

Finally, the following statements use the simulation method of Wicklin (2013) and the previously defined intensity to simulate a spatial point pattern on 10 hills in an area of 50×50 units. The covariates, Elevation and Slope, are also computed over a grid of points in the region of interest.

```
proc iml;
   reset storage=sasuser.SPPThin;
   load module=Uniform2d
        module=HomogPoissonProcess
        module=InhomogPoissonProcess;

   reset storage=sasuser.SPPFlowers;
   load  module=Elevation module=Slope module=Intensity lambda0;
```

```
   a = 50;
   b = 50;

   Hills = { 9.2 48.5 0.2 13.0,
           46.1 48.5 0.3 26.6,
            2.5  3.3 0.7 26.2,
           42.7  3.4 0.9 14.9,
           13.6 34.5 1.0 11.3,
           34.4 20.6 0.3 14.4,
           23.8 42.2 0.4 29.5,
           29.1 18.9 0.5 25.3,
           46.6 46.5 0.3 14.9,
           19.6 23.6 0.5  8.4};

   call randseed(12345);

   free Cov;
   Cov = j((a+1) * (b+1), 5, 0);
   do x = 0 to a; do y = 0 to b;
      Cov[x#(a+1)+y+1,] =  (x || y || Elevation(x,y) || Slope(x,y)
                           || Intensity(x,y));
   end; end;

   create Covariates var {"x" "y" "Elevation" "Slope" "Intensity"};
   append from Cov;
   close Covariates;

   Hills = Hills // {25 5 2 15};
   z = InhomogPoissonProcess(a, b);

   create Events var {"x" "y"};
   append from z;
   close;

quit;

data simAll;
   set Events(in=e) Covariates;
   Flowers = e;
run;
```

The point pattern data set and the covariate data set are combined in the simAll data set and the event observations can be identified by using a variable Flowers.

```
proc spp data=simAll plots(equate)=(trends observations);
   process trees = (x, y /area=(0,0,50,50) Event=Flowers);
   trend grad = field(x,y, Elevation);
   trend elev = field(x,y, Slope);
run;
```

Output 114.3.1 shows the point pattern. The point pattern has been simulated to include a Gaussian bump at the center of the study region.

Output 114.3.1 Spatial Point Pattern of Simulated Flowers

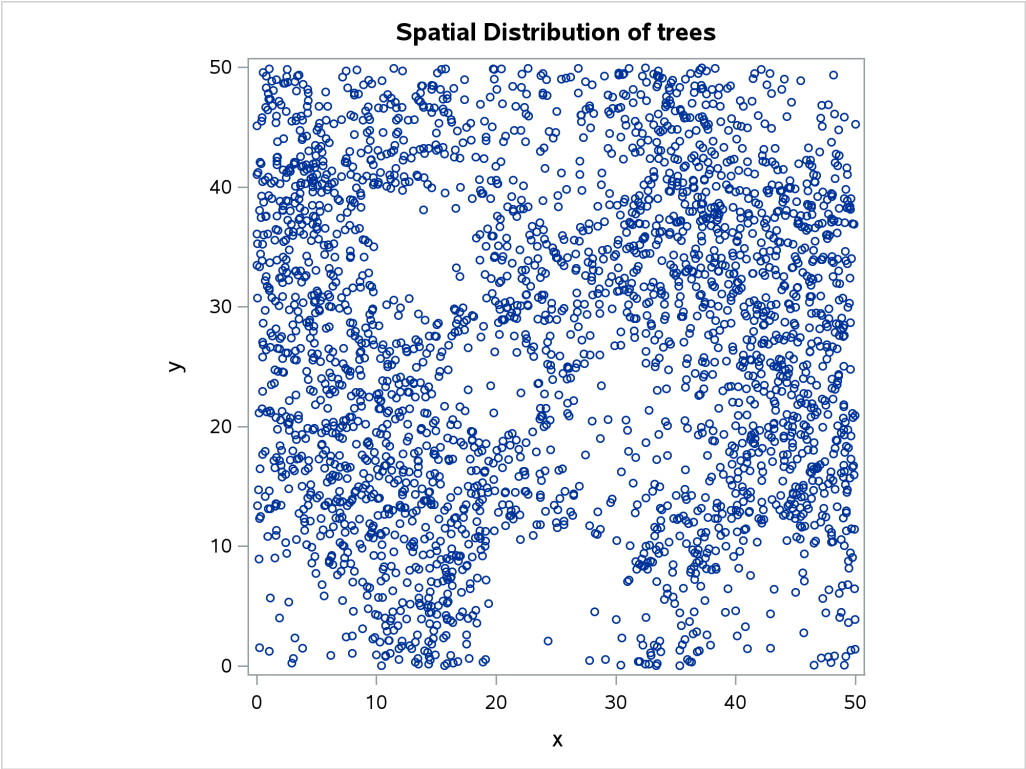

Output 114.3.2 shows the spatial covariate Elevation, and Output 114.3.3 shows the spatial covariate Slope. The covariates have been simulated to include several Gaussian hills, and they are continuous within the 50×50 study region (that is, every point in the region has a value for these covariates).

Output 114.3.2 Spatial Covariate Elevation

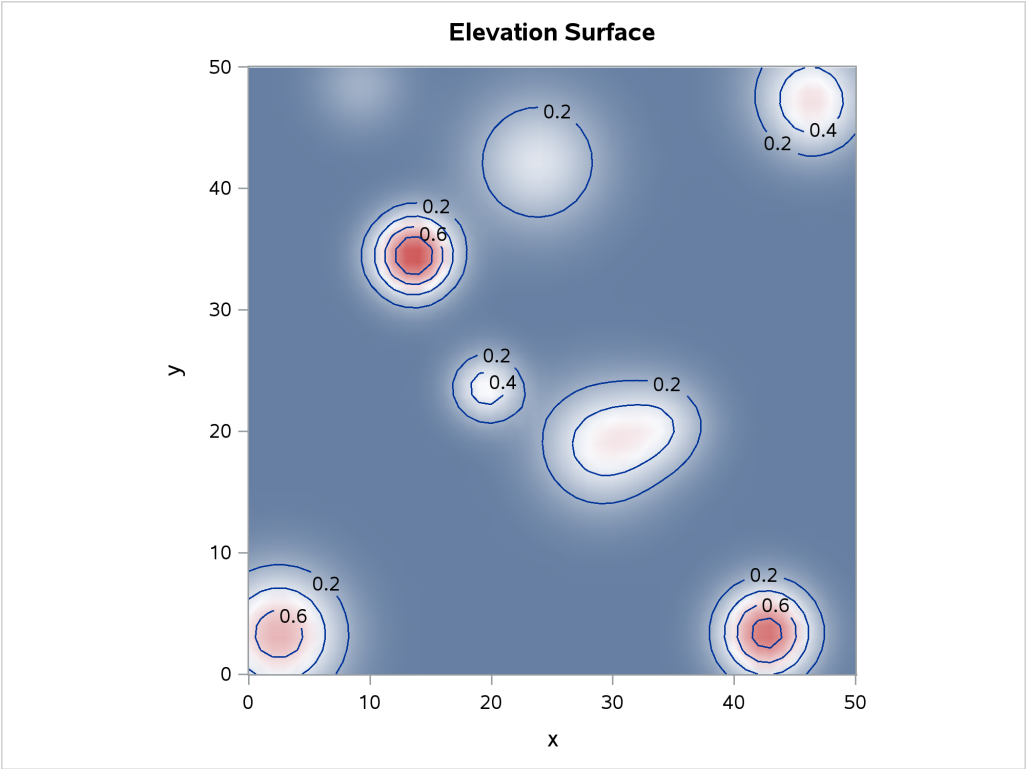

Output 114.3.3 Spatial Covariate Slope

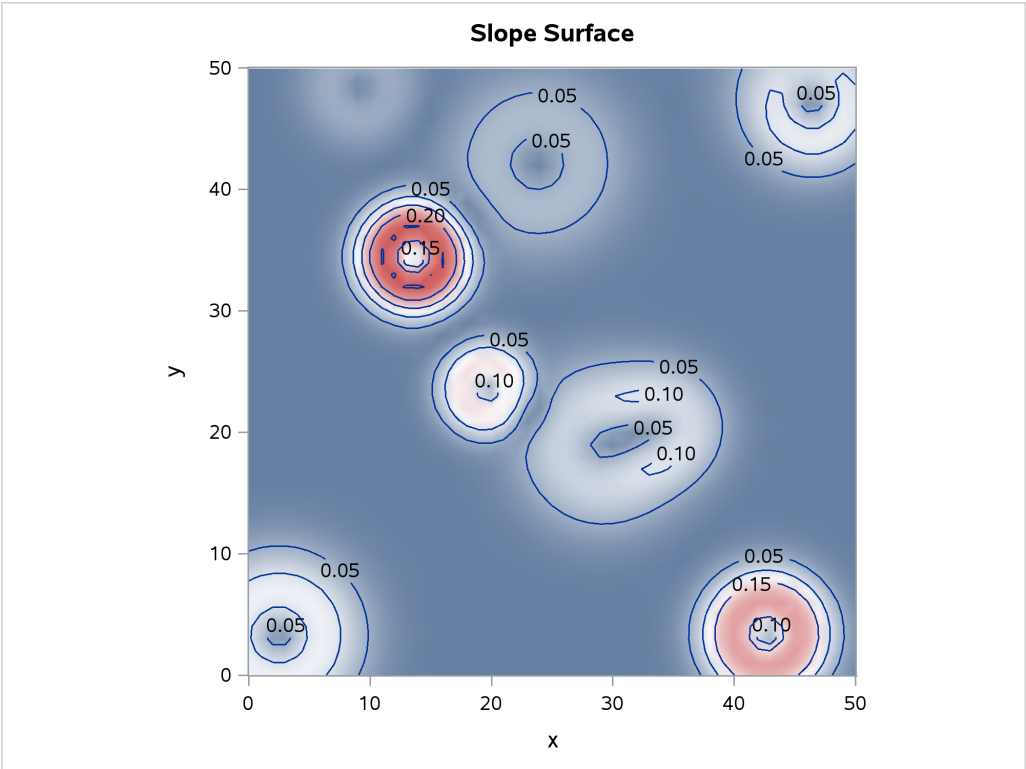

The following code fits an intensity model for the simulated point pattern that involves the simulated covariates Elevation and Slope. It also requests model validation diagnostics, including residuals and the goodness-of-fit test.

```
proc spp data=simAll seed=1  plots(equate)=(residual);
   process trees = (x,y /area=(0,0,50,50) Event=Flowers) /quadrat(4,2 /details);
   trend elev = field(x,y, Elevation);
   trend slope = field(x,y, Slope);
   model trees = elev slope/residual(b=5) gof;
run;
```

Output 114.3.4 shows the fitted intensity estimate that is based on the model that involves just the covariates Elevation and Slope.

Output 114.3.4 Fitted Intensity Estimate for the Simulated Point Pattern

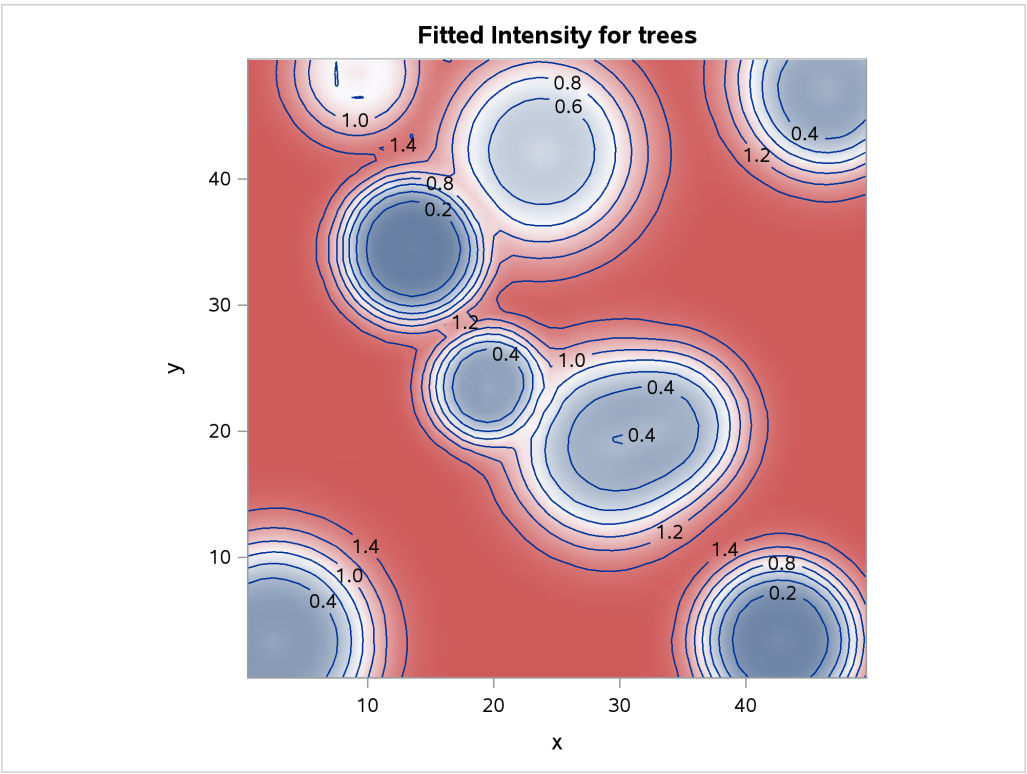

Output 114.3.5 shows the residual diagnostics for the model. It is clear from the smoothed residual plot at the bottom right corner of Output 114.3.5 that the model that involves just the covariates Elevation and Slope fails to account for the Gaussian bump in the middle of the study region. This is revealed by the trend at the center of the smoothed residual plot at the bottom right corner of Output 114.3.5.

Output 114.3.5 Residual Diagnostics for the Fitted Model

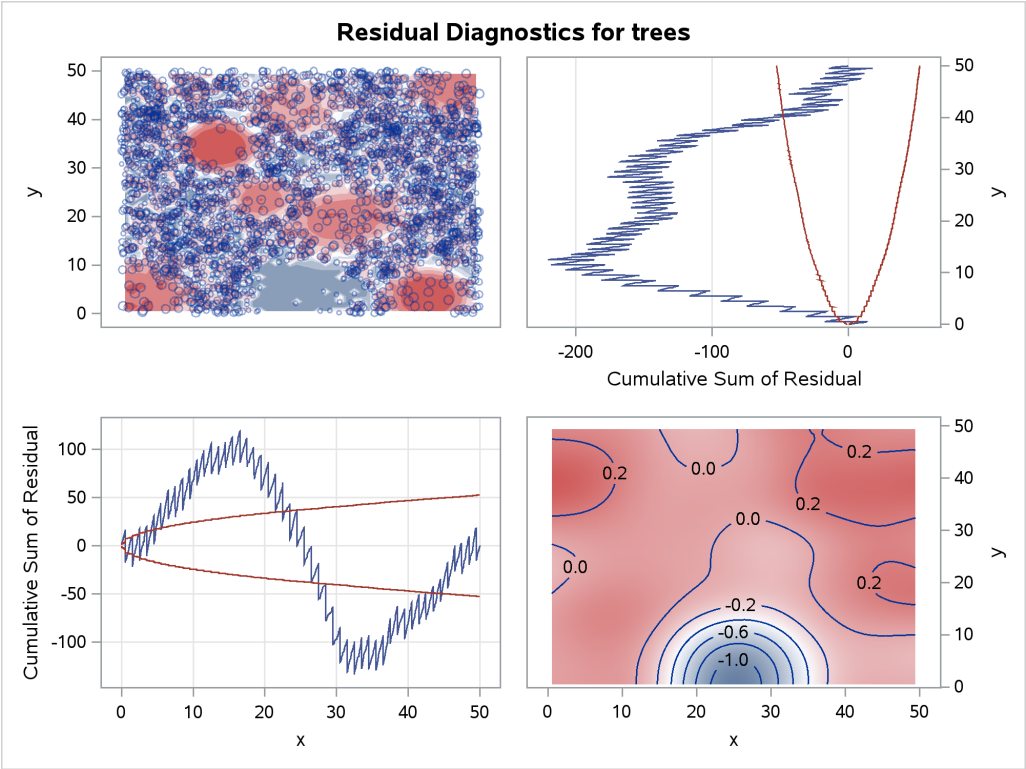

Consequently, the goodness-of-fit test rejects the hypothesis that the point pattern was generated by the fitted model. This is evident in the low p-value that is obtained for the Pearson chi-square test for goodness of fit, which is shown in Output 114.3.6.

Output 114.3.6 Pearson Chi-Square Test for Goodness of Fit

Pearson Chi-Square Test for Goodness-of-Fit			
DF	Dispersion Index	Chi-Square	Pr > ChiSq
7	12.742	89.19	<.0001

Output 114.3.7 shows the corresponding Pearson residuals for the goodness-of-fit test.

Output 114.3.7 Quadrat Information and Pearson Residuals for Goodness-of-Fit Test

Quadrat Information for Goodness-of-Fit Test					
ID	Quadrat	Expected Frequency	Count	Percentage	Pearson Residual
1	(1,1)	352	375	13.58%	1.20
2	(2,1)	398	310	11.22%	-4.41
3	(3,1)	298	184	6.66%	-6.60
4	(4,1)	323	349	12.64%	1.45
5	(1,2)	383	450	16.29%	3.43
6	(2,2)	261	264	9.56%	0.19
7	(3,2)	383	397	14.37%	0.70
8	(4,2)	371	433	15.68%	3.22

When the model involves only the covariates Elevation and Slope, the residual diagnostics and the goodness-of-fit test both reveal discrepancies in the model that do not fully account for the simulated point pattern. In particular, the model misses the Gaussian bump in the middle of the study region.

References

Baddeley, A. (2007). "Spatial Point Processes and Their Applications." In *Stochastic Geometry*, edited by W. Weil, 1–75. Vol. 1892 of Lecture Notes in Mathematics. Berlin: Springer.

Baddeley, A. (2010). "Modeling Strategies." In *Handbook of Spatial Statistics*, edited by A. E. Gelfand, P. J. Diggle, M. Fuentes, and P. Guttorp, 339–369. Boca Raton, FL: Chapman & Hall/CRC.

Baddeley, A. (2014). Personal communication.

Baddeley, A., and Turner, R. (2000). "Practical Maximum Pseudolikelihood for Spatial Point Patterns." *Australian and New Zealand Journal of Statistics* 42:283–322.

Baddeley, A., and Turner, R. (2005). "Spatstat: An R Package for Analyzing Spatial Point Patterns." *Journal of Statistical Software* 12:1–42.

Baddeley, A., and Turner, R. (2013). "Spatstat: An R Package for Spatial Statistics (ver. 1.31-3)." http://www.spatstat.org/.

Baddeley, A., Turner, R., Moller, J., and Hazelton, M. (2005). "Residual Analysis for Spatial Point Processes." *Journal of the Royal Statistical Society, Series B* 67:616–666.

Baddeley, A. J., Kerscher, M., Schladitz, K., and Scott, B. T. (2000). "Estimating the *J* Function without Edge Correction." *Statistica Neerlandica* 54:315–328.

Banerjee, S. (2005). "On Geodetic Distance Computations in Spatial Modeling." *Biometrics* 61:617–625.

Berman, M., and Turner, R. (1992). "Approximating Point Process Likelihoods with GLIM." *Journal of the Royal Statistical Society, Series C* 41:31–38.

Condit, R. (1998). *Tropical Forest Census Plots: Methods and Results from Barro Colorado Island, Panama, and a Comparison with Other Plots*. Berlin: Springer-Verlag.

Condit, R., Hubbell, S. P., and Foster, R. B. (1996). "Changes in Tree Species Abundance in a Neotropical Forest: Impact of Climate Change." *Journal of Tropical Ecology* 12:231–256.

Cressie, N., and Collins, L. B. (2001). "Analysis of Spatial Point Patterns Using Bundles of Product Density LISA Functions." *Journal of Agricultural, Biological, and Environmental Statistics* 6:118–135.

D'Agostino, R. B., and Stephens, M., eds. (1986). *Goodness-of-Fit Techniques*. New York: Marcel Dekker.

Diggle, P. J. (2003). *Statistical Analysis of Spatial Point Patterns*. New York: Oxford University Press.

Diggle, P. J., Eglen, S. R., and Troy, J. B. (2006). "Modelling the Bivariate Spatial Distribution of Amacrine Cells." In *Case Studies in Spatial Point Process Modeling*, edited by A. Baddeley, P. Gregori, J. Mateu, R. Stoica, and D. Stoyan, 215–233. New York: Springer.

Diggle, P. J., Rowlingson, B., and Su, T.-L. (2005). "Point Process Methodology for On-Line Spatio-temporal Disease Surveillance." *Environmetrics* 16:423–434.

Fiksel, T. (1988). "Edge-Corrected Density Estimators for Point Processes." *Statistics* 19:67–75.

Hubbell, S. P., and Foster, R. B. (1983). "Diversity of Canopy Trees in a Neotropical Forest and Implications for the Conservation of Tropical Trees." In *Tropical Rain Forest: Ecology and Management*, edited by S. J. Sutton, T. C. Whitmore, and A. C. Chadwick, 25–41. Oxford: Blackwell.

Hughes, A. (1985). "New Perspectives in Retinal Organisation." *Progress in Retinal Research* 4:243–314.

Illian, J., Penttinen, A., Stoyan, H., and Stoyan, D. (2008). *Statistical Analysis and Modelling of Spatial Point Patterns*. Hoboken, NJ: John Wiley & Sons.

Lewis, P. A. W., and Shedler, G. S. (1979). "Simulation of Nonhomogeneous Poisson Processes by Thinning." *Naval Research Logistics Quarterly* 26:403–413.

Ripley, B. D. (1988). *Statistical Inference for Spatial Processes*. Cambridge: Cambridge University Press.

Ripley, B. D., and Rasson, J.-P. (1977). "Finding the Edge of a Poisson Forest." *Journal of Applied Probability* 14:483–491.

Schabenberger, O., and Gotway, C. A. (2005). *Statistical Methods for Spatial Data Analysis*. Boca Raton, FL: Chapman & Hall/CRC.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. New York: Chapman & Hall.

Stoyan, D. (1987). "Statistical Analysis of Spatial Point Processes: A Soft-Core Model and Cross-Correlations of Marks." *Biometrical Journal* 29:971–980.

Stoyan, D., and Stoyan, H. (1994). *Fractals, Random Shapes, and Point Fields: Methods of Geometrical Statistics*. Chichester, UK: John Wiley & Sons.

Wicklin, R. (2013). *Simulating Data with SAS*. Cary, NC: SAS Institute Inc.

Chapter 115
The STDIZE Procedure

Contents

Overview: STDIZE Procedure **9636**
Getting Started: STDIZE Procedure **9636**
Syntax: STDIZE Procedure **9643**
- PROC STDIZE Statement 9643
- BY Statement 9649
- FREQ Statement 9650
- LOCATION Statement 9650
- SCALE Statement 9650
- VAR Statement 9650
- WEIGHT Statement 9651

Details: STDIZE Procedure **9652**
- Standardization Methods 9652
- Computation of the Statistics 9654
- Computing Quantiles 9655
- Constant Data 9656
- Missing Values 9657
- Output Data Sets 9657
- Displayed Output 9658
- ODS Table Names 9659

Example: STDIZE Procedure **9659**
- Example 115.1: Standardization of Variables in Cluster Analysis 9659

References **9669**

Overview: STDIZE Procedure

The STDIZE procedure standardizes one or more numeric variables in a SAS data set by subtracting a location measure and dividing by a scale measure. A variety of location and scale measures are provided, including estimates that are resistant to outliers and clustering. Some of the well-known standardization methods such as mean, median, standard deviation, range, Huber's estimate, Tukey's biweight estimate, and Andrew's wave estimate are available in the STDIZE procedure.

In addition, you can multiply each standardized value by a constant and add a constant. Thus, the final output value is

$$result = add + multiply \times \frac{original - location}{scale}$$

where

result	= final output value
add	= constant to add (ADD= option)
multiply	= constant to multiply by (MULT= option)
original	= original input value
location	= location measure
scale	= scale measure

PROC STDIZE can also find quantiles in one pass of the data, a capability that is especially useful for very large data sets. With such data sets, the UNIVARIATE procedure might have high or excessive memory or time requirements.

Getting Started: STDIZE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/stdizein.sas.)

The following example demonstrates how you can use the STDIZE procedure to obtain location and scale measures of your data.

In the following hypothetical data set, a random sample of grade twelve students is selected from a number of coeducational schools. Each school is classified as one of two types: Urban or Rural. There are 40 observations.

The variables are id (student identification), Type (type of school attended: 'urban'=urban area and 'rural'=rural area), and total (total assessment scores in History, Geometry, and Chemistry).

The following DATA step creates the SAS data set TotalScores.

```
data TotalScores;
   title 'High School Scores Data';
   input id Type $ total @@;
   datalines;
```

```
 1 rural 135   2 rural 125   3 rural 223   4 rural 224   5 rural 133
 6 rural 253   7 rural 144   8 rural 193   9 rural 152  10 rural 178
11 rural 120  12 rural 180  13 rural 154  14 rural 184  15 rural 187
16 rural 111  17 rural 190  18 rural 128  19 rural 110  20 rural 217
21 urban 192  22 urban 186  23 urban  64  24 urban 159  25 urban 133
26 urban 163  27 urban 130  28 urban 163  29 urban 189  30 urban 144
31 urban 154  32 urban 198  33 urban 150  34 urban 151  35 urban 152
36 urban 151  37 urban 127  38 urban 167  39 urban 170  40 urban 123
;
```

Suppose you now want to standardize the total scores in different types of schools prior to any further analysis. Before standardizing the total scores, you can use the box plot from PROC BOXPLOT to summarize the total scores for both types of schools.

```
ods graphics on;
proc boxplot data=TotalScores;
   plot total*Type / boxstyle=schematic noserifs;
run;
```

The PLOT statement in the PROC BOXPLOT statement creates the schematic plots (without the serifs) when you specify **boxstyle=schematic noserifs**. Figure 115.1 displays a box plot for each type of school.

Figure 115.1 Schematic Plots from PROC BOXPLOT

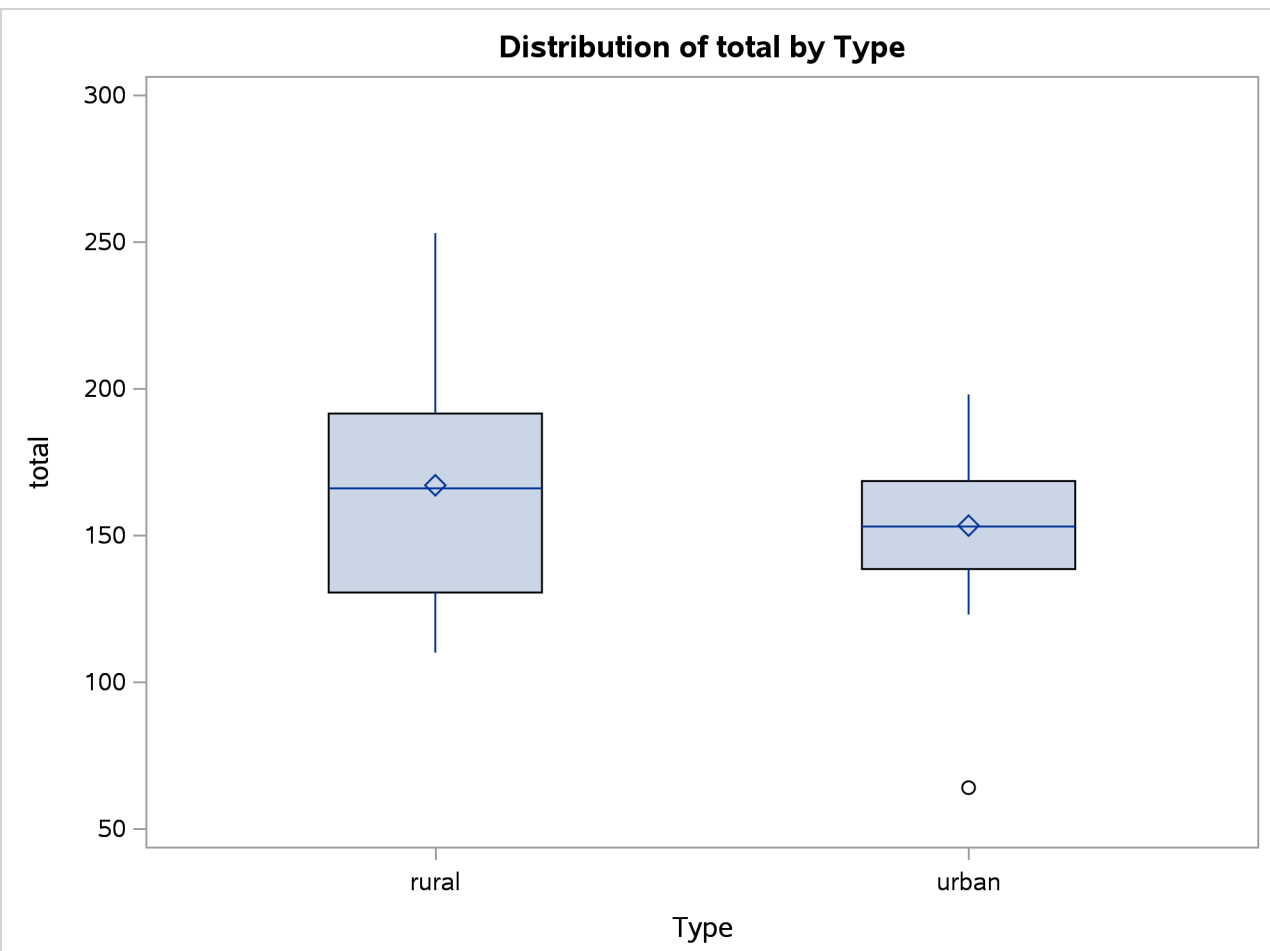

Inspection reveals that one urban score is a low outlier. Also, if you compare the lengths of two box plots, there seems to be twice as much dispersion for the rural scores as for the urban scores.

The following PROC UNIVARIATE statement reports the information about the extreme values of the Score variable for each type of school:

```
proc univariate data=TotalScores;
   var total;
   by Type;
run;
```

Figure 115.2 displays the table from PROC UNIVARIATE for the lowest and highest five total scores for urban schools. The outlier (Obs = 23), marked in Figure 115.2 by the symbol '0', has a score of 64.

Figure 115.2 Table for Extreme Observations When Type=urban

High School Scores Data

The UNIVARIATE Procedure
Variable: total

Type=urban

Extreme Observations			
Lowest		Highest	
Value	Obs	Value	Obs
64	23	170	39
123	40	186	22
127	37	189	29
130	27	192	21
133	25	198	32

The following PROC STDIZE procedure requests the METHOD=STD option for computing the location and scale measures:

```
proc stdize data=totalscores method=std pstat;
   title2 'METHOD=STD';
   var total;
   by Type;
run;
```

Figure 115.3 displays the table of location and scale measures from the PROC STDIZE statement. PROC STDIZE uses the sample mean as the location measure and the sample standard deviation as the scale measure for standardizing. The PSTAT option displays a table containing these two measures.

Figure 115.3 Location and Scale Measures Table When METHOD=STD

High School Scores Data
METHOD=STD

The STDIZE Procedure

Type=rural

Location and Scale Measures

Location = mean
Scale = standard deviation

Name	Location	Scale	N
total	167.050000	41.956713	20

High School Scores Data
METHOD=STD

The STDIZE Procedure

Type=urban

Location and Scale Measures

Location = mean
Scale = standard deviation

Name	Location	Scale	N
total	153.300000	30.066768	20

The ratio of the scale of rural scores to the scale of urban scores is approximately 1.4 (41.96/30.07). This ratio is smaller than the dispersion ratio observed in the previous schematic plots.

The STDIZE procedure provides several location and scale measures that are resistant to outliers. The following statements invoke three different standardization methods and display the tables for the location and scale measures:

```
proc stdize data=totalscores method=mad pstat;
   title2 'METHOD=MAD';
   var total;
   by Type;
run;

proc stdize data=totalscores method=iqr pstat;
   title2 'METHOD=IQR';
   var total;
   by Type;
run;

proc stdize data=totalscores method=abw(4) pstat;
   title2 'METHOD=ABW(4)';
   var total;
   by Type;
run;
```

Figure 115.4 displays the table of location and scale measures when the standardization method is median absolute deviation (MAD). The location measure is the median, and the scale measure is the median absolute

deviation from the median. The ratio of the scale of rural scores to the scale of urban scores is approximately 2.06 (32.0/15.5) and is close to the dispersion ratio observed in the previous schematic plots.

Figure 115.4 Location and Scale Measures Table When METHOD=MAD

High School Scores Data
METHOD=MAD

The STDIZE Procedure

Type=rural

Location and Scale Measures

Location = median
Scale = median abs dev from median

Name	Location	Scale	N
total	166.000000	32.000000	20

High School Scores Data
METHOD=MAD

The STDIZE Procedure

Type=urban

Location and Scale Measures

Location = median
Scale = median abs dev from median

Name	Location	Scale	N
total	153.000000	15.500000	20

Figure 115.5 displays the table of location and scale measures when the standardization method is IQR. The location measure is the median, and the scale measure is the interquartile range. The ratio of the scale of rural scores to the scale of urban scores is approximately 2.03 (61/30) and is, in fact, the dispersion ratio observed in the previous schematic plots.

Figure 115.5 Location and Scale Measures Table When METHOD=IQR

High School Scores Data
METHOD=IQR

The STDIZE Procedure

Type=rural

Location and Scale Measures

Location = median
Scale = interquartile range

Name	Location	Scale	N
total	166.000000	61.000000	20

Figure 115.5 *continued*

High School Scores Data
METHOD=IQR

The STDIZE Procedure

Type=urban

Location and Scale Measures

Location = median
Scale = interquartile range

Name	Location	Scale	N
total	153.000000	30.000000	20

Figure 115.6 displays the table of location and scale measures when the standardization method is ABW, for which the location measure is the biweight one-step M-estimate, and the scale measure is the biweight A-estimate. Note that the initial estimate for ABW is MAD. The following steps help to decide the value of the tuning constant:

1. For rural scores, the location estimate for MAD is 166.0, and the scale estimate for MAD is 32.0. The maximum of the rural scores is 253 (not shown), and the minimum is 110 (not shown). Thus, the tuning constant needs to be 3 so that it does not reject any observation that has a score between 110 to 253.

2. For urban scores, the location estimate for MAD is 153.0, and the scale estimate for MAD is 15.5. The maximum of the rural scores is 198, and the minimum (also an outlier) is 64. Thus, the tuning constant needs to be 4 so that it rejects the outlier (64) but includes the maximum (198) as an normal observation.

3. The maximum of the tuning constants, obtained in steps 1 and 2, is 4.

See Goodall (1983, Chapter 11) for details about the tuning constant. The ratio of the scale of rural scores to the scale of urban scores is approximately 2.06 (32.0/15.5). It is also close to the dispersion ratio observed in the previous schematic plots.

Figure 115.6 Location and Scale Measures Table When METHOD=ABW

High School Scores Data
METHOD=ABW(4)

The STDIZE Procedure

Type=rural

Location and Scale Measures

Location = biweight 1-step M-estimate Scale = biweight A-estimate

Name	Location	Scale	N
total	162.889603	56.662855	20

Figure 115.6 *continued*

High School Scores Data
METHOD=ABW(4)

The STDIZE Procedure

Type=urban

Location and Scale Measures

Location = biweight 1-step M-estimate A-estimate
Scale = biweight

Name	Location	Scale	N
total	156.014608	28.615980	20

The preceding analysis shows that METHOD=MAD, METHOD=IQR, and METHOD=ABW all provide better dispersion ratios than METHOD=STD does.

You can recompute the standard deviation after deleting the outlier from the original data set for comparison. The following statements create a data set NoOutlier that excludes the outlier from the TotalScores data set and invoke PROC STDIZE with METHOD=STD.

```
data NoOutlier;
   set totalscores;
   if (total = 64) then delete;
run;

proc stdize data=NoOutlier method=std pstat;
   title2 'After Removing Outlier, METHOD=STD';
   var total;
   by Type;
run;
```

Figure 115.7 displays the location and scale measures after deleting the outlier. The lack of resistance of the standard deviation to outliers is clearly illustrated: if you delete the outlier, the sample standard deviation of urban scores changes from 30.07 to 22.09. The new ratio of the scale of rural scores to the scale of urban scores is approximately 1.90 (41.96/22.09).

Figure 115.7 Location and Scale Measures Table When METHOD=STD without the Outlier

High School Scores Data
After Removing Outlier, METHOD=STD

The STDIZE Procedure

Type=rural

Location and Scale Measures

Location = mean
Scale = standard deviation

Name	Location	Scale	N
total	167.050000	41.956713	20

Figure 115.7 *continued*

High School Scores Data
After Removing Outlier, METHOD=STD

The STDIZE Procedure

Type=urban

Location and Scale Measures

Location = mean
Scale = standard deviation

Name	Location	Scale	N
total	158.000000	22.088207	19

Syntax: STDIZE Procedure

The following statements are available in the STDIZE procedure:

```
PROC STDIZE <options> ;
    BY variables ;
    FREQ variable ;
    LOCATION variables ;
    SCALE variables ;
    VAR variables ;
    WEIGHT variable ;
```

The PROC STDIZE statement is required. The BY, LOCATION, FREQ, VAR, SCALE, and WEIGHT statements are described in alphabetical order following the PROC STDIZE statement.

PROC STDIZE Statement

PROC STDIZE < *options* > **;**

The PROC STDIZE statement invokes the STDIZE procedure. You can specify the following options in the PROC STDIZE statement. Table 115.1 summarizes the options available in the PROC STDIZE statement.

Table 115.1 Summary of PROC STDIZE Statement Options

Option	Description
Specify standardization methods	
METHOD=	Specifies the name of the standardization method
INITIAL=	Specifies the method for computing initial estimates for the A estimates
Unstandardize variables	
UNSTD	Unstandardizes variables when you also specify the METHOD=IN option

Table 115.1 *continued*

Option	Description
Process missing values	
NOMISS	Omits observations with any missing values from computation
MISSING=	Specifies the method or a numeric value for replacing missing values
REPLACE	Replaces missing data with zero in the standardized data
REPONLY	Replaces missing data with the location measure (does not standardize the data)
Specify data set details	
DATA=	Specifies the input data set
KEEPLEN	Specifies that output variables inherit the length of the analysis variable
OUT=	Specifies the output data set
OPREFIX=	Specifies that original variables appear in the OUT= data set
SPREFIX=	Specifies a prefix for the standardized variable names
OUTSTAT=	Specifies the output statistic data set
Specify computational settings	
VARDEF=	Specifies the variances divisor
NMARKERS=	Specifies the number of markers when you also specify PCTLMTD=ONEPASS
MULT=	Specifies the constant to multiply each value by after standardizing
ADD=	Specifies the constant to add to each value after standardizing and multiplying by the value specified in the MULT= option
FUZZ=	Specifies the relative fuzz factor for writing the output
Specify percentiles	
PCTLDEF=	Specifies the definition of percentiles when you also specify the PCTLMTD=ORD_STAT option
PCTLMTD=	Specifies the method used to estimate percentiles
PCTLPTS=	Writes observations containing percentiles to the data set specified in the OUTSTAT= option
Normalize scale estimators	
NORM	Normalizes the scale estimator to be consistent for the standard deviation of a normal distribution
SNORM	Normalizes the scale estimator to have an expectation of approximately 1 for a standard normal distribution
Specify output	
PSTAT	Displays the location and scale measures

These options and their abbreviations are described (in alphabetical order) in the remainder of this section.

ADD=*c*

specifies a constant, *c*, to add to each value after standardizing and multiplying by the value you specify in the MULT= option. The default value is 0.

DATA=*SAS-data-set*

specifies the input data set to be standardized. If you omit the DATA= option, the most recently created data set is used.

FUZZ=*c*

specifies the relative fuzz factor. The default value is 1E–14. For the OUT= data set, the score is computed as follows:

$$\text{if } |result| < m \times c \text{ then } result = 0$$

where *m* is the constant specified in the MULT= option, or 1 if MULT= option is not specified.

For the OUTSTAT= data set and the location and scale table, the *scale* and *location* values are computed as follows:

$$\text{if } scale < |location| \times c \text{ then } scale = 0$$

Otherwise,

$$\text{if } |location| < m \times c \text{ then } location = 0$$

INITIAL=*method*

specifies the method for computing initial estimates for the A estimates (ABW, AWAVE, and AHUBER). You cannot specify the following methods for initial estimates: INITIAL=ABW, INITIAL=AHUBER, INITIAL=AWAVE, and INITIAL=IN. The default is INITIAL=MAD.

KEEPLEN

specifies that the standardized variables inherit the lengths of the analysis variables that PROC STDIZE uses to derive them. PROC STDIZE stores numbers in double-precision without this option.

Caution: The KEEPLEN option causes the standardized variables to permanently lose numeric precision by truncating or rounding the values. However, the precision of the output variables will match that of the input.

METHOD=*name*

specifies the name of the method for computing location and scale measures. Valid values for *name* are as follows: MEAN, MEDIAN, SUM, EUCLEN, USTD, STD, RANGE, MIDRANGE, MAXABS, IQR, MAD, ABW, AHUBER, AWAVE, AGK, SPACING, L, and IN.

For details about these methods, see the descriptions in the section "Standardization Methods" on page 9652. The default is METHOD=STD.

MISSING=*method* | *value*

specifies the method (or a numeric value) to use for replacing missing values. If you omit the MISSING= option, the REPLACE option replaces missing values with the location measure that you specify in the METHOD= option. Use the MISSING= option when you want to replace missing values with a different value.

If you specify a *method*, the corresponding location measure (see the section "Standardization Methods" on page 9652) replaces the missing values before standardizing. This *method*, which is used to compute the location measure that replaces the missing values, can be different from what you specify in the METHOD= option, which is used to compute the standardization. If you specify a numeric *value*, it replaces the missing values after standardizing.

Note that the standardization depends only on nonmissing data; the location and scale measures are computed from nonmissing data. Therefore any replacement of the missing data (regardless of whether the replacement takes place before or after the standardization) does not affect the standardization of nonmissing values.

You can also replace missing values first and then compute the standardization by using the specified *value*. For more information, see the section "Missing Values" on page 9657.

If you specify the UNSTD option, the MISSING= option is ignored.

MULT=*c*

specifies a constant, *c*, by which to multiply each value after standardizing. The default value is 1.

NMARKERS=*n*

specifies the number of markers used when you specify the one-pass algorithm (PCTLMTD=ONEPASS). The value *n* must be greater than or equal to 5. The default value is 105.

NOMISS

omits observations that have missing values for any of the analyzed variables from calculation of the location and scale measures. If you omit this option, all nonmissing values are used.

NORM

normalizes the scale estimator to be consistent for the standard deviation of a normal distribution when you specify the option METHOD=AGK, METHOD=IQR, METHOD=MAD, or METHOD=SPACING.

OPREFIX< =*o-prefix* **>**

specifies that the original variables should appear in the OUT= data set. You can optionally specify an equal sign and a prefix. For example, if OPREFIX=Original, then the names of the variables are OriginalVAR1, OriginalVAR2, and so on, where VAR1 and VAR2 are the original variable names. The value of OPREFIX= must be different from the value of SPREFIX=. If you specify OPREFIX, without an equal sign and a prefix, then the default prefix is null and you must specify SPREFIX=*s-prefix*.

OUT=*SAS-data-set*

specifies the name of the SAS data set that PROC STDIZE creates. By default, the output data set is a copy of the DATA= data set except that the analyzed variables have been standardized. Analyzed variables are those that you specify in the VAR statement or, if there is no VAR statement, all numeric variables that are not listed in any other statement. However, you can use the OPREFIX option to include both the original and standardized variables in the output data set. You can change variable

names by specifying prefixes with the OPREFIX= and SPREFIX= options. See the section "Output Data Sets" on page 9657 for more information.

If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

If you omit the OUT= option, PROC STDIZE creates an output data set that is named according to the DATA*n* convention.

OUTSTAT=*SAS-data-set*
specifies the name of the SAS data set that contains the location and scale measures and other computed statistics. See the section "Output Data Sets" on page 9657 for more information.

PCTLDEF=*percentiles*
specifies which of five definitions is used to calculate percentiles when you specify the option PCTLMTD=ORD_STAT. By default, PCTLDEF=5. Note that the option PCTLMTD=ONEPASS implies PCTLDEF=5. See the section "Computational Methods for the PCTLDEF= Option" on page 9655 for details about percentile definition.

You cannot use the PCTLDEF= option when you compute weighted quantiles.

PCTLMTD=ORD_STAT | ONEPASS | P2
specifies the method to be used to estimate percentiles. Specify the PCTLMTD=ORD_STAT option to compute the percentiles by the order statistics method.

The PCTLMTD=ONEPASS option modifies an algorithm invented by Jain and Chlamtac (1985). See the section "Computing Quantiles" on page 9655 for more details about this algorithm.

PCTLPTS=*n*
writes percentiles to the OUTSTAT= data set. Values of *n* can be any decimal number between 0 and 100, inclusive.

A requested percentile is identified by the _TYPE_ variable in the OUTSTAT= data set with a value of P*n*. For example, suppose that you specify the option PCTLPTS=10, 30. The corresponding observations in the OUTSTAT= data set that contain the 10th and 30th percentiles would then have the values _TYPE_=P10 and _TYPE_=P30, respectively.

PSTAT
displays the location and scale measures.

REPLACE
replaces missing data with the value 0 in the standardized data. This computation is equivalent to using the location measure (which depends only on nonmissing data) as the replacement value for the missing data before standardization: subtraction of the location measure by itself leads to zero. To replace missing data with other values, see the description of the MISSING= option. You cannot specify both the REPLACE and REPONLY options. If you specify the UNSTD option, the REPLACE option is ignored.

REPONLY

replaces missing data only; PROC STDIZE does not standardize the data. Missing values are replaced with the location measure unless you also specify the MISSING=*value* option, in which case missing values are replaced with *value*. You cannot specify both the REPONLY and REPLACE options.

SNORM

normalizes the scale estimator to have an expectation of approximately 1 for a standard normal distribution when you specify the METHOD=SPACING option.

SPREFIX< =*s-prefix* >

specifies a prefix for the standardized variables. For example, if SPREFIX=Std, then the names of the standardized variables are StdVAR1, StdVAR2, and so on, where VAR1 and VAR2 are the original variable names. The value of SPREFIX= must be different from the value of OPREFIX=. The default prefix is null. If you omit the SPREFIX option, the standardized variables still appear in the OUT= data set by default and the variable names remain the same. If you want to have the variable names changed, you need to specify a prefix with SPREFIX=*s-prefix*.

UNSTD

UNSTDIZE

unstandardizes variables when you specify the METHOD=IN(ds) option. The location and scale measures, along with constants for addition and multiplication that the unstandardization is based on, are identified by the _TYPE_ variable in the ds data set.

The ds data set must have a _TYPE_ variable and contain the following two observations: a _TYPE_= 'LOCATION' observation and a _TYPE_= 'SCALE' observation. The variable _TYPE_ can also contain the optional observations, 'ADD' and 'MULT'; if these observations are not found in the ds data set, the constants specified in the ADD= and MULT= options (or their default values) are used for unstandardization.

See the section "OUTSTAT= Data Set" on page 9657 for details about the statistics that each value of _TYPE_ represents. The formula used for unstandardization is as follows: If the final output value from the previous standardization is calculated as

$$result = add + multiply \times \frac{original - location}{scale}$$

The unstandardized variable is computed as

$$original = scale \times \frac{result - add}{multiply} + location$$

The UNSTD option is ignored if you specify the REPONLY option. If you specify the UNSTD option, the MISSING= and REPLACE options are ignored.

VARDEF=DF | N | WDF | WEIGHT | WGT

specifies the divisor to be used in the calculation of variances. By default, VARDEF=DF. The values and associated divisors are as follows.

Value	Divisor	Formula
DF	Degrees of freedom	$n - 1$
N	Number of observations	n
WDF	Sum of weights minus 1	$(\sum_i w_i) - 1$
WEIGHT | WGT	Sum of weights	$\sum_i w_i$

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC STDIZE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the STDIZE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

When you specify the option METHOD=IN(ds), the following rules are applied to BY-group processing:

- If the ds data set does not contain any of the BY variables, the entire DATA= data set is standardized by the location and scale measures (along with the constants for addition and multiplication) in the ds data set.
- If the ds data set contains some, but not all, of the BY variables or if some BY variables do not have the same type or length in the ds data set that they have in the DATA= data set, PROC STDIZE displays an error message and stops.
- If all of the BY variables appear in the ds data set with the same type and length as in the DATA= data set, each BY group in the DATA= data set is standardized using the location and scale measures (along with the constants for addition and multiplication) from the corresponding BY group in the ds data set. The BY groups in the ds data set must be in the same order in which they appear in the DATA= data set. All BY groups in the DATA= data set must also appear in the ds data set. If you do not specify the NOTSORTED option, some BY groups can appear in the ds data set but not in the DATA= data set; such BY groups are not used in standardizing data.

For more information about BY-group processing, see the “Grouping Data” section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If one variable in the input data set represents the frequency of occurrence for other values in the observation, specify the variable name in a FREQ statement. PROC STDIZE treats the data set as if each observation appeared *n* times, where *n* is the value of the FREQ variable for the observation. Nonintegral values of the FREQ variable are truncated to the largest integer less than the FREQ value. If the FREQ variable has a value that is less than 1 or is missing, the observation is not used in the analysis.

NOTRUNCATE

NOTRUNC

specifies that frequency values are not truncated to integers.

The nonintegral values of the FREQ variable can be used for the following standardization methods: AGK, ABW, AHUBER, AWAVE, EUCLEN, IQR, L, MAD, MEAN, MEDIAN, SPACING, STD, SUM, and USTD. The nonintegral frequency values are used for the MAD, MEDIAN, or IQR method only when PCTLMTD=ORD_STAT is specified. If PCTLMTD=ONEPASS is specified, the NOTRUNCATE option is ignored.

LOCATION Statement

LOCATION *variables* **;**

The LOCATION statement specifies a list of numeric variables that contain location measures in the input data set specified by the METHOD=IN option.

SCALE Statement

SCALE *variables* **;**

The SCALE statement specifies the list of numeric variables that contain scale measures in the input data set specified by the METHOD=IN option.

VAR Statement

VAR *variable* **;**

The VAR statement lists numeric variables to be standardized. If you omit the VAR statement, all numeric variables not listed in the BY, FREQ, and WEIGHT statements are used.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement specifies a numeric variable in the input data set with values that are used to weight each observation. Only one variable can be specified.

The WEIGHT variable values can be nonintegers. An observation is used in the analysis only if the value of the WEIGHT variable is greater than zero.

The WEIGHT variable applies only when you specify the following standardization methods: AGK, EUCLEN, IQR, L, MAD, MEAN, MEDIAN, STD, SUM, and USTD. Weights are used for the METHOD=MAD, MEDIAN, or IQR only when PCTLMTD=ORD_STAT is specified; if PCTLMTD=ONEPASS is specified, the WEIGHT statement is ignored.

PROC STDIZE uses the value of the WEIGHT variable to calculate the sample mean and sample variances:

$$\overline{x}_w = \sum_i w_i x_i / \sum_i w_i \qquad \text{(sample mean)}$$

$$us_w^2 = \sum_i w_i x_i^2 / d \qquad \text{(uncorrected sample variances)}$$

$$s_w^2 = \sum_i w_i (x_i - \overline{x}_w)^2 / d \qquad \text{(sample variances)}$$

where w_i is the weight value of the ith observation, x_i is the value of the ith observation, and d is the divisor controlled by the VARDEF= option (see the VARDEF= option for details).

The following weighted statistics are defined accordingly:

MEAN — the weighted mean, $\overline{x}_w$

SUM — the weighted sum, $\sum_i w_i x_i$

USTD — the weighted uncorrected standard deviation, $\sqrt{us_w^2}$

STD — the weighted standard deviation, $\sqrt{s_w^2}$

EUCLEN — the weighted Euclidean length, computed as the square root of the weighted uncorrected sum of squares:

$$\sqrt{\sum_i w_i x_i^2}$$

MEDIAN — the weighted median. See the section "Weighted Percentiles" on page 9656 for the formulas and descriptions.

MAD — the weighted median absolute deviation from the weighted median. See the section "Weighted Percentiles" on page 9656 for the formulas and descriptions.

IQR — the weighted median, 25th percentile, and the 75th percentile. See the section "Weighted Percentiles" on page 9656 for the formulas and descriptions.

AGK — the AGK estimate. This estimate is documented further in the ACECLUS procedure as the METHOD=COUNT option. See the discussion of the WEIGHT statement in Chapter 27, "The ACECLUS Procedure," for information about how the WEIGHT variable is applied to the AGK estimate.

L — the L_p estimate. This estimate is documented further in the FASTCLUS procedure as the LEAST= option. See the discussion of the WEIGHT statement in Chapter 45, "The FASTCLUS Procedure," for information about how the WEIGHT variable is used to compute weighted cluster means. The number of clusters is always 1.

Details: STDIZE Procedure

Standardization Methods

The following table lists standardization methods and their corresponding location and scale measures available with the METHOD= option.

Table 115.2 Available Standardization Methods

Method	Location	Scale
MEAN	Mean	1
MEDIAN	Median	1
SUM	0	Sum
EUCLEN	0	Euclidean length
USTD	0	Standard deviation about origin
STD	Mean	Standard deviation
RANGE	Minimum	Range
MIDRANGE	Midrange	Range/2
MAXABS	0	Maximum absolute value
IQR	Median	Interquartile range
MAD	Median	Median absolute deviation from median
ABW(c)	Biweight one-step M-estimate	Biweight A-estimate
AHUBER(c)	Huber one-step M-estimate	Huber A-estimate
AWAVE(c)	Wave one-step M-estimate	Wave A-estimate
AGK(p)	Mean	AGK estimate (ACECLUS)
SPACING(p)	Mid-minimum spacing	Minimum spacing
L(p)	L(p)	L(p)
IN(ds)	Read from data set	Read from data set

For METHOD=ABW(c), METHOD=AHUBER(c), or METHOD=AWAVE(c), c is a positive numeric tuning constant.

For METHOD=AGK(p), p is a numeric constant that gives the proportion of pairs to be included in the

estimation of the within-cluster variances.

For METHOD=SPACING(p), p is a numeric constant that gives the proportion of data to be contained in the spacing.

For METHOD=L(p), p is a numeric constant greater than or equal to 1 that specifies the power to which differences are to be raised in computing an L(p) or Minkowski metric.

For METHOD=IN(ds), ds is the name of a SAS data set that meets either of the following two conditions:

- The data set contains a _TYPE_ variable. The observation that contains the location measure corresponds to the value _TYPE_= 'LOCATION', and the observation that contains the scale measure corresponds to the value _TYPE_= 'SCALE'. You can also use a data set created by the OUTSTAT= option from another PROC STDIZE statement as the ds data set. See the section "Output Data Sets" on page 9657 for the contents of the OUTSTAT= data set.
- The data set contains the location and scale variables specified by the LOCATION and SCALE statements.

PROC STDIZE reads in the location and scale variables in the ds data set by first looking for the _TYPE_ variable in the ds data set. If it finds this variable, PROC STDIZE continues to search for all variables specified in the VAR statement. If it does not find the _TYPE_ variable, PROC STDIZE searches for the location variables specified in the LOCATION statement and the scale variables specified in the SCALE statement.

The variable _TYPE_ can also contain the optional observations, 'ADD' and 'MULT'. If these observations are found in the ds data set, the values in the observation of _TYPE_ = 'MULT' are the multiplication constants, and the values in the observation of _TYPE_ = 'ADD' are the addition constants; otherwise, the constants specified in the ADD= and MULT= options (or their default values) are used.

For robust estimators, see Goodall (1983) and Iglewicz (1983). The MAD method has the highest breakdown point (50%), but it is somewhat inefficient. The ABW, AHUBER, and AWAVE methods provide a good compromise between breakdown and efficiency. The L(p) location estimates are increasingly robust as p drops from 2 (which corresponds to least squares, or mean estimation) to 1 (which corresponds to least absolute value, or median estimation). However, the L(p) scale estimates are not robust.

The SPACING method is robust to both outliers and clustering (Janssen et al. 1995) and is, therefore, a good choice for cluster analysis or nonparametric density estimation. The mid-minimum spacing method estimates the mode for small p. The AGK method is also robust to clustering and more efficient than the SPACING method, but it is not as robust to outliers and takes longer to compute. If you expect g clusters, the argument to METHOD=SPACING or METHOD=AGK should be $\frac{1}{g}$ or less. The AGK method is less biased than the SPACING method for small samples. As a general guide, it is reasonable to use AGK for samples of size 100 or less and SPACING for samples of size 1,000 or more, with the treatment of intermediate sample sizes depending on the available computer resources.

Computation of the Statistics

Formulas for statistics of METHOD=MEAN, METHOD=MEDIAN, METHOD=SUM, METHOD=USTD, METHOD=STD, METHOD=RANGE, and METHOD=IQR are given in the chapter "Elementary Statistics Procedures" (*Base SAS Procedures Guide*).

Note that the computations of median and upper and lower quartiles depend on the PCTLMTD= option.

The other statistics listed in Table 115.2, except for METHOD=IN, are described as follows:

EUCLEN
: Euclidean length.
$\sqrt{\sum_{i=1}^{n} x_i^2}$, where x_i is the ith observation and n is the total number of observations in the sample.

L(p)
: Minkowski metric. This metric is documented as the LEAST=p option in the PROC FASTCLUS statement of the FASTCLUS procedure (see Chapter 45, "The FASTCLUS Procedure").

  If you specify METHOD=L(p) in the PROC STDIZE statement, your results are similar to those obtained from PROC FASTCLUS if you specify the LEAST=p option with MAXCLUS=1 (and use the default values of the MAXITER= option). The difference between the two types of calculations concerns the maximum number of iterations. In PROC STDIZE, it is a criterion for convergence on all variables; in PROC FASTCLUS, it is a criterion for convergence on a single variable.

  The location and scale measures for L(p) are output to the OUTSEED= data set in PROC FASTCLUS.

MIDRANGE
: $(maximum + minimum)/2$

ABW(c)
: Tukey's biweight. See Goodall (1983, pp. 376–378, p. 385) for the biweight one-step M-estimate. Also see Iglewicz (1983, pp. 416-418) for the biweight A-estimate.

AHUBER(c)
: Hubers. See Goodall (1983, pp. 371–374) for the Huber one-step M-estimate. Also see Iglewicz (1983, pp. 416-418) for the Huber A-estimate of scale.

AWAVE(c)
: Andrews' wave. See Goodall (1983, p. 376) for the Wave one-step M-estimate. Also see Iglewicz (1983, pp. 416-418) for the Wave A-estimate of scale.

AGK(p)
: The noniterative univariate form of the estimator described by Art, Gnanadesikan, and Kettenring (1982).

  The AGK estimate is documented in the section on the METHOD= option in the PROC ACECLUS statement of the ACECLUS procedure (also see the section "Background" on page 1028 in Chapter 27, "The ACECLUS Procedure"). Specifying METHOD=AGK(p) in the PROC STDIZE statement is the same as specifying METHOD=COUNT and P=p in the PROC ACECLUS statement.

SPACING(p)
: The absolute difference between two data values. The minimum spacing for a proportion p is the minimum absolute difference between two data values that contain a proportion p of the data between them. The mid-minimum spacing is the mean of these two data values.

Computing Quantiles

PROC STDIZE offers two methods for computing quantiles: the one-pass approach and the order-statistics approach (like that used in the UNIVARIATE procedure).

The one-pass approach used in PROC STDIZE modifies the P^2 algorithm for histograms proposed by Jain and Chlamtac (1985). The primary difference comes from the movement of markers. The one-pass method allows a marker to move to the right (or left) by more than one position (to the largest possible integer) as long as it does not result in two markers being in the same position. The modification is necessary in order to incorporate the FREQ variable.

You might obtain inaccurate results if you use the one-pass approach to estimate quantiles beyond the quartiles (that is, when you estimate quantiles < P25 or quantiles > P75). A large sample size (10,000 or more) is often required if the tail quantiles (quantiles $\leq$ P10 or quantiles $\geq$ P90) are requested. Note that, for variables with highly skewed or heavy-tailed distributions, tail quantile estimates might be inaccurate.

The order-statistics approach for estimating quantiles is faster than the one-pass method but requires that the entire data set be stored in memory. The accuracy in estimating the quantiles is comparable for both methods when the requested percentiles are between the lower and upper quartiles. The default is PCTLMTD=ORD_STAT if enough memory is available; otherwise, PCTLMTD=ONEPASS.

Computational Methods for the PCTLDEF= Option

You can specify one of five methods for computing quantile statistics when you use the order-statistics approach (PCTLMTD=ORD_STAT); otherwise, the PCTLDEF=5 method is used when you use the one-pass approach (PCTLMTD=ONEPASS).

Percentile Definitions Let n be the number of nonmissing values for a variable, and let $x_1, x_2, \ldots, x_n$ represent the ordered values of the variable. For the tth percentile, let $p = t/100$. In the following definitions numbered 1, 2, 3, and 5, let

$$np = j + g$$

where j is the integer part and g is the fractional part of np. For definition 4, let

$$(n+1)p = j + g$$

Given the preceding definitions, the tth percentile, y, is defined as follows:

PCTLDEF=1 weighted average at x_{np}

$$y = (1-g)x_j + gx_{j+1}$$

where x_0 is taken to be x_1

PCTLDEF=2 observation numbered closest to np

$$y = x_i$$

where i is the integer part of $np + 1/2$ if $g \neq 1/2$. If $g = 1/2$, then
$y = x_j$ if j is even, or
$y = x_{j+1}$ if j is odd

PCTLDEF=3 empirical distribution function

$$y = x_j \text{ if } g = 0$$

$$y = x_{j+1} \text{ if } g > 0$$

PCTLDEF=4 weighted average aimed at $x_{p(n+1)}$

$$y = (1 - g)x_j + gx_{j+1}$$

where x_{n+1} is taken to be x_n

PCTLDEF=5 empirical distribution function with averaging

$$y = (x_j + x_{j+1})/2 \text{ if } g = 0$$

$$y = x_{j+1} \text{ if } g > 0$$

Weighted Percentiles

When you specify a WEIGHT statement, or specify the NOTRUNCATE option in a FREQ statement, the percentiles are computed differently. The 100pth weighted percentile y is computed from the empirical distribution function with averaging

$$y = \begin{cases} \frac{1}{2}(x_i + x_{i+1}) & \text{if } \sum_{j=1}^{i} w_j = pW \\ x_{i+1} & \text{if } \sum_{j=1}^{i} w_j < pW < \sum_{j=1}^{i+1} w_j \end{cases}$$

where w_i is the weight associated with x_i, and where $W = \sum_{i=1}^{n} w_i$ is the sum of the weights.

For PCTLMTD= ORD_STAT, the PCTLDEF= option is not applicable when a WEIGHT statement is used, or when a NOTRUNCATE option is specified in a FREQ statement. However, in this case, if all the weights are identical, the weighted percentiles are the same as the percentiles that would be computed without a WEIGHT statement and with PCTLDEF=5.

For PCTLMTD= ONEPASS, the quantile computation currently does not use any weights.

Constant Data

Constant variables are not standardized. The scale value is set to missing when the data are constant.

Missing Values

By default, missing values are replaced by a specified constant after standardization. In some cases, such as when you specify the REPLACE option and the value 0 is used, this is computationally equivalent to replacing missing values with the location measure before standardization. In all cases, the replacement value does not affect the standardization of nonmissing values: the location and scale measures are computed only from nonmissing data.

You can suppress the standardization if you want only to replace missing values (see the REPONLY option).

You can also replace missing values first and then compute the standardization by using the specified value. This requires two steps. First, you specify both the MISSING=*value* option and the REPONLY option; this suppresses the standardization and creates a new, complete data set (use the OUT= option to specify the name of the new data set), where the missing values are replaced with *value*. Then you run PROC STDIZE again with the new data set. This way, the standardization calculation uses the replaced *value*.

If you specify the NOMISS option, PROC STDIZE omits observations that have missing values in the analyzed variables from computation of the location and scale measures.

Output Data Sets

OUT= Data Set

By default, the output data set is a copy of the DATA= data set except that the analyzed variables have been standardized. Analyzed variables are those that you specify in the VAR statement or, if there is no VAR statement, all numeric variables that are not listed in any other statement. However, you can use the OPREFIX option to include both the original and standardized variables in the output data set. You can change variable names by specifying prefixes with the OPREFIX=*o-prefix* and SPREFIX=*s-prefix* options, but keep in mind that the two prefixes must be different. See OPREFIX and SPREFIX for more information.

OUTSTAT= Data Set

The new data set contains the following variables:

- the BY variables, if any
- _TYPE_, a character variable
- the analyzed variables

Each observation in the new data set contains a type of statistic as indicated by the _TYPE_ variable. The values of the _TYPE_ variable are as follows:

LOCATION	location measure of each variable
SCALE	scale measure of each variable
ADD	constant specified in the ADD= option. This value is the same for each variable.
MULT	constant specified in the MULT= option. This value is the same for each variable.

N	total number of nonmissing positive frequencies of each variable
NORM	norm measure of each variable. This observation is produced only when you specify the NORM option with METHOD=AGK, METHOD=IQR, METHOD=MAD, or METHOD=SPACING or when you specify the SNORM option with METHOD=SPACING.
NObsRead	number of physical records read
NObsUsed	number of physical records used in the analysis
NObsMiss	number of physical records containing missing values
P*n*	percentiles of each variable, as specified by the PCTLPTS= option. The argument n is any real number such that $0 \leq n \leq 100$
SumFreqsRead	sum of the frequency variable (or the sum of NObsUsed ones when there is no frequency variable) for all observations read
SumFreqsUsed	sum of the frequency variable (or the sum of NObsUsed ones when there is no frequency variable) for all observations used in the analysis
SumWeightsRead	sum of the weight variable (or the sum of NObsUsed ones when there is no weight variable) for all observations read
SumWeightsUsed	sum of the weight variable (or the sum of NObsUsed ones when there is no weight variable) for all observations used in the analysis

Displayed Output

If you specify the PSTAT option, PROC STDIZE displays the following statistics for each variable:

- the name of the variable, Name
- the location estimate, Location
- the scale estimate, Scale
- the norm estimate, Norm (when you specify the NORM option with METHOD=AGK, METHOD=IQR, METHOD=MAD, or METHOD=SPACING or when you specify the SNORM option with METHOD=SPACING)
- sum of nonmissing positive frequencies, N
- sum of nonmissing positive weights if the WEIGHT statement is specified, Sum of Weights

ODS Table Names

PROC STDIZE assigns a name to the single table it creates. You can use this name to reference the table when using the Output Delivery System (ODS) to select a table or create an output data set. This name is listed in Table 115.3. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 115.3 ODS Table Produced by PROC STDIZE

ODS Table Name	Description	Statement	Option
Statistics	Location and Scale Measures	PROC	PSTAT

Example: STDIZE Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 115.1: Standardization of Variables in Cluster Analysis

(View the complete code for this example (stdizeex.sas) in the example repository.)

To illustrate the effect of standardization in cluster analysis, this example uses the Fish data set described in the "Getting Started" section of Chapter 45, "The FASTCLUS Procedure." The numbers are measurements taken on 159 fish caught from the same lake (Laengelmaevesi) near Tampere in Finland (Puranen 1917). The fish data set is available from the Sashelp library.

The species (bream, parkki, pike, perch, roach, smelt, and whitefish), weight, three different length measurements (measured from the nose of the fish to the beginning of its tail, the notch of its tail, and the end of its tail), height, and width of each fish are recorded.

A couple of new variables are created in the Fish data set: Weight3 and logLengthRatio. The weight of a fish indicates its size—a heavier pike tends to be larger than a lighter pike. To get a one-dimensional measure of the size of a fish, take the cubic root of the weight (Weight3). The variables Height, Width, Length1, Length2, and Length3 are rescaled in order to adjust for dimensionality. The logLengthRatio variable measures the tail length.

Because the new variables Weight3–logLengthRatio depend on the variable Weight, observations with missing values for Weight are not added to the data set. Consequently, there are 157 observations in the SAS data set Sashelp.Fish.

Before you perform a cluster analysis on coordinate data, it is necessary to consider scaling or transforming the variables since variables with large variances tend to have a larger effect on the resulting clusters than variables with small variances do.

This example uses three different approaches to standardize or transform the data prior to the cluster analysis. The first approach uses several standardization methods provided in the STDIZE procedure. However, since

standardization is not always appropriate prior to the clustering (see Milligan and Cooper (1987) for a Monte Carlo study on various methods of variable standardization), the second approach performs the cluster analysis with no standardization. The third approach invokes the ACECLUS procedure to transform the data into a within-cluster covariance matrix.

The clustering is performed by the FASTCLUS procedure to find seven clusters. Note that the variables Length2 and Length3 are eliminated from this analysis since they both are significantly and highly correlated with the variable Length1. The correlation coefficients are 0.9958 and 0.9604, respectively. An output data set is created, and the FREQ procedure is invoked to compare the clusters with the species classification.

The DATA step is as follows:

```
title 'Fish Measurement Data';

data Fish;
   set sashelp.fish;
   if Weight <= 0 or Weight = . then delete;
   Weight3 = Weight ** (1/3);
   Height = Height / Weight3;
   Width  = Width  / Weight3;
   Length1 = Length1 / Weight3;
   Length2 = Length2 / Weight3;
   Length3 = Length3 / Weight3;
   LogLengthRatio = log(Length3 / Length1);
run;
```

The following macro, Std, standardizes the Fish data. The macro reads a single argument, mtd, which selects the METHOD= specification to be used in PROC STDIZE.

```
/*--- macro for standardization ---*/

%macro Std(mtd);
   title2 "Data are Standardized by PROC STDIZE with METHOD= &mtd";
   proc stdize data=fish out=sdzout method=&mtd;
      var Length1 logLengthRatio Height Width Weight3;
   run;
%mend Std;
```

The following macro, FastFreq, includes a PROC FASTCLUS statement for performing cluster analysis and a PROC FREQ statement for crosstabulating species with the cluster membership information that is derived from the previous PROC FASTCLUS statement. The macro reads a single argument, ds, which selects the input data set to be used in PROC FASTCLUS.

```
/*--- macro for clustering and crosstabulating ---*/
/*--- cluster membership with species ---*/

%macro FastFreq(ds);
   proc fastclus data=&ds out=clust maxclusters=7 maxiter=100 noprint;
      var Length1 logLengthRatio Height Width Weight3;
   run;

   proc freq data=clust;
      tables species*cluster;
   run;
```

```
%mend FastFreq;
```

The following analysis (labeled 'Approach 1') includes 18 different methods of standardization followed by clustering. Since there is a large amount of output from this approach, only results from METHOD=STD, METHOD=RANGE, METHOD=AGK(0.14), and METHOD=SPACING(0.14) are shown. The following statements produce Output 115.1.1 through Output 115.1.4.

```
/*     Approach 1: data are standardized by PROC STDIZE   */

%Std(MEAN);
%FastFreq(sdzout);

%Std(MEDIAN);
%FastFreq(sdzout);

%Std(SUM);
%FastFreq(sdzout);

%Std(EUCLEN);
%FastFreq(sdzout);

%Std(USTD);
%FastFreq(sdzout);

%Std(STD);
%FastFreq(sdzout);

%Std(RANGE);
%FastFreq(sdzout);

%Std(MIDRANGE);
%FastFreq(sdzout);

%Std(MAXABS);
%FastFreq(sdzout);

%Std(IQR);
%FastFreq(sdzout);

%Std(MAD);
%FastFreq(sdzout);

%Std(AGK(.14));
%FastFreq(sdzout);

%Std(SPACING(.14));
%FastFreq(sdzout);

%Std(ABW(5));
%FastFreq(sdzout);

%Std(AWAVE(5));
%FastFreq(sdzout);
```

```
%Std(L(1));
%FastFreq(sdzout);

%Std(L(1.5));
%FastFreq(sdzout);

%Std(L(2));
%FastFreq(sdzout);
```

Output 115.1.1 Data Are Standardized by PROC STDIZE with METHOD=STD

Fish Measurement Data
Data are Standardized by PROC STDIZE with METHOD= STD

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Species by CLUSTER CLUSTER(Cluster)							
Species	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Total**
Bream	0	0	0	0	0	34	0	34
	0.00	0.00	0.00	0.00	0.00	21.66	0.00	21.66
	0.00	0.00	0.00	0.00	0.00	100.00	0.00	
	0.00	0.00	0.00	0.00	0.00	100.00	0.00	
Parkki	0	0	0	0	11	0	0	11
	0.00	0.00	0.00	0.00	7.01	0.00	0.00	7.01
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	
Perch	0	17	0	12	0	0	27	56
	0.00	10.83	0.00	7.64	0.00	0.00	17.20	35.67
	0.00	30.36	0.00	21.43	0.00	0.00	48.21	
	0.00	89.47	0.00	92.31	0.00	0.00	54.00	
Pike	17	0	0	0	0	0	0	17
	10.83	0.00	0.00	0.00	0.00	0.00	0.00	10.83
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
Roach	0	0	0	0	0	0	19	19
	0.00	0.00	0.00	0.00	0.00	0.00	12.10	12.10
	0.00	0.00	0.00	0.00	0.00	0.00	100.00	
	0.00	0.00	0.00	0.00	0.00	0.00	38.00	
Smelt	0	0	13	0	0	0	1	14
	0.00	0.00	8.28	0.00	0.00	0.00	0.64	8.92
	0.00	0.00	92.86	0.00	0.00	0.00	7.14	
	0.00	0.00	100.00	0.00	0.00	0.00	2.00	
Whitefish	0	2	0	1	0	0	3	6
	0.00	1.27	0.00	0.64	0.00	0.00	1.91	3.82
	0.00	33.33	0.00	16.67	0.00	0.00	50.00	
	0.00	10.53	0.00	7.69	0.00	0.00	6.00	
Total	17	19	13	13	11	34	50	157
	10.83	12.10	8.28	8.28	7.01	21.66	31.85	100.00

Output 115.1.2 Data Are Standardized by PROC STDIZE with METHOD=RANGE

Fish Measurement Data
Data are Standardized by PROC STDIZE with METHOD= RANGE

The FREQ Procedure

Table of Species by CLUSTER

Frequency Percent Row Pct Col Pct	CLUSTER(Cluster)							
Species	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Total**
Bream	0	0	34	0	0	0	0	34
	0.00	0.00	21.66	0.00	0.00	0.00	0.00	21.66
	0.00	0.00	100.00	0.00	0.00	0.00	0.00	
	0.00	0.00	100.00	0.00	0.00	0.00	0.00	
Parkki	0	0	0	0	0	11	0	11
	0.00	0.00	0.00	0.00	0.00	7.01	0.00	7.01
	0.00	0.00	0.00	0.00	0.00	100.00	0.00	
	0.00	0.00	0.00	0.00	0.00	100.00	0.00	
Perch	0	0	0	9	20	0	27	56
	0.00	0.00	0.00	5.73	12.74	0.00	17.20	35.67
	0.00	0.00	0.00	16.07	35.71	0.00	48.21	
	0.00	0.00	0.00	29.03	86.96	0.00	100.00	
Pike	17	0	0	0	0	0	0	17
	10.83	0.00	0.00	0.00	0.00	0.00	0.00	10.83
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
Roach	0	0	0	19	0	0	0	19
	0.00	0.00	0.00	12.10	0.00	0.00	0.00	12.10
	0.00	0.00	0.00	100.00	0.00	0.00	0.00	
	0.00	0.00	0.00	61.29	0.00	0.00	0.00	
Smelt	0	14	0	0	0	0	0	14
	0.00	8.92	0.00	0.00	0.00	0.00	0.00	8.92
	0.00	100.00	0.00	0.00	0.00	0.00	0.00	
	0.00	100.00	0.00	0.00	0.00	0.00	0.00	
Whitefish	0	0	0	3	3	0	0	6
	0.00	0.00	0.00	1.91	1.91	0.00	0.00	3.82
	0.00	0.00	0.00	50.00	50.00	0.00	0.00	
	0.00	0.00	0.00	9.68	13.04	0.00	0.00	
Total	17	14	34	31	23	11	27	157
	10.83	8.92	21.66	19.75	14.65	7.01	17.20	100.00

Output 115.1.3 Data Are Standardized by PROC STDIZE with METHOD=AGK(0.14)

Fish Measurement Data
Data are Standardized by PROC STDIZE with METHOD= AGK(.14)

The FREQ Procedure

Table of Species by CLUSTER

Frequency Percent Row Pct Col Pct	CLUSTER(Cluster)							
Species	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Total**
Bream	0	0	34	0	0	0	0	34
	0.00	0.00	21.66	0.00	0.00	0.00	0.00	21.66
	0.00	0.00	100.00	0.00	0.00	0.00	0.00	
	0.00	0.00	100.00	0.00	0.00	0.00	0.00	
Parkki	11	0	0	0	0	0	0	11
	7.01	0.00	0.00	0.00	0.00	0.00	0.00	7.01
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
Perch	0	0	0	3	0	20	33	56
	0.00	0.00	0.00	1.91	0.00	12.74	21.02	35.67
	0.00	0.00	0.00	5.36	0.00	35.71	58.93	
	0.00	0.00	0.00	13.04	0.00	86.96	94.29	
Pike	0	0	0	0	17	0	0	17
	0.00	0.00	0.00	0.00	10.83	0.00	0.00	10.83
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	
Roach	0	0	0	17	0	0	2	19
	0.00	0.00	0.00	10.83	0.00	0.00	1.27	12.10
	0.00	0.00	0.00	89.47	0.00	0.00	10.53	
	0.00	0.00	0.00	73.91	0.00	0.00	5.71	
Smelt	0	14	0	0	0	0	0	14
	0.00	8.92	0.00	0.00	0.00	0.00	0.00	8.92
	0.00	100.00	0.00	0.00	0.00	0.00	0.00	
	0.00	100.00	0.00	0.00	0.00	0.00	0.00	
Whitefish	0	0	0	3	0	3	0	6
	0.00	0.00	0.00	1.91	0.00	1.91	0.00	3.82
	0.00	0.00	0.00	50.00	0.00	50.00	0.00	
	0.00	0.00	0.00	13.04	0.00	13.04	0.00	
Total	11	14	34	23	17	23	35	157
	7.01	8.92	21.66	14.65	10.83	14.65	22.29	100.00

Output 115.1.4 Data Are Standardized by PROC STDIZE with METHOD=SPACING(0.14)

Fish Measurement Data
Data are Standardized by PROC STDIZE with METHOD= SPACING(.14)

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Species by CLUSTER CLUSTER(Cluster)							
Species	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Total**
Bream	0	0	0	0	0	0	34	34
	0.00	0.00	0.00	0.00	0.00	0.00	21.66	21.66
	0.00	0.00	0.00	0.00	0.00	0.00	100.00	
	0.00	0.00	0.00	0.00	0.00	0.00	100.00	
Parkki	0	0	11	0	0	0	0	11
	0.00	0.00	7.01	0.00	0.00	0.00	0.00	7.01
	0.00	0.00	100.00	0.00	0.00	0.00	0.00	
	0.00	0.00	100.00	0.00	0.00	0.00	0.00	
Perch	20	0	0	0	0	36	0	56
	12.74	0.00	0.00	0.00	0.00	22.93	0.00	35.67
	35.71	0.00	0.00	0.00	0.00	64.29	0.00	
	86.96	0.00	0.00	0.00	0.00	94.74	0.00	
Pike	0	17	0	0	0	0	0	17
	0.00	10.83	0.00	0.00	0.00	0.00	0.00	10.83
	0.00	100.00	0.00	0.00	0.00	0.00	0.00	
	0.00	100.00	0.00	0.00	0.00	0.00	0.00	
Roach	0	0	0	17	0	2	0	19
	0.00	0.00	0.00	10.83	0.00	1.27	0.00	12.10
	0.00	0.00	0.00	89.47	0.00	10.53	0.00	
	0.00	0.00	0.00	85.00	0.00	5.26	0.00	
Smelt	0	0	0	0	14	0	0	14
	0.00	0.00	0.00	0.00	8.92	0.00	0.00	8.92
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	
	0.00	0.00	0.00	0.00	100.00	0.00	0.00	
Whitefish	3	0	0	3	0	0	0	6
	1.91	0.00	0.00	1.91	0.00	0.00	0.00	3.82
	50.00	0.00	0.00	50.00	0.00	0.00	0.00	
	13.04	0.00	0.00	15.00	0.00	0.00	0.00	
Total	23	17	11	20	14	38	34	157
	14.65	10.83	7.01	12.74	8.92	24.20	21.66	100.00

The following analysis (labeled ‘Approach 2’) applies the cluster analysis directly to the original data. The following statements produce Output 115.1.5.

```
/*         Approach 2: data are untransformed             */

title2 'Data are Untransformed';
%FastFreq(fish);
```

Output 115.1.5 Untransformed Data

Fish Measurement Data
Data are Untransformed

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of Species by CLUSTER CLUSTER(Cluster)							
Species	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Total**
Bream	13	0	0	0	0	0	21	34
	8.28	0.00	0.00	0.00	0.00	0.00	13.38	21.66
	38.24	0.00	0.00	0.00	0.00	0.00	61.76	
	44.83	0.00	0.00	0.00	0.00	0.00	47.73	
Parkki	2	3	0	0	6	0	0	11
	1.27	1.91	0.00	0.00	3.82	0.00	0.00	7.01
	18.18	27.27	0.00	0.00	54.55	0.00	0.00	
	6.90	18.75	0.00	0.00	15.38	0.00	0.00	
Perch	8	9	0	1	20	0	18	56
	5.10	5.73	0.00	0.64	12.74	0.00	11.46	35.67
	14.29	16.07	0.00	1.79	35.71	0.00	32.14	
	27.59	56.25	0.00	6.67	51.28	0.00	40.91	
Pike	0	0	10	0	1	4	2	17
	0.00	0.00	6.37	0.00	0.64	2.55	1.27	10.83
	0.00	0.00	58.82	0.00	5.88	23.53	11.76	
	0.00	0.00	100.00	0.00	2.56	100.00	4.55	
Roach	3	4	0	0	12	0	0	19
	1.91	2.55	0.00	0.00	7.64	0.00	0.00	12.10
	15.79	21.05	0.00	0.00	63.16	0.00	0.00	
	10.34	25.00	0.00	0.00	30.77	0.00	0.00	
Smelt	0	0	0	14	0	0	0	14
	0.00	0.00	0.00	8.92	0.00	0.00	0.00	8.92
	0.00	0.00	0.00	100.00	0.00	0.00	0.00	
	0.00	0.00	0.00	93.33	0.00	0.00	0.00	
Whitefish	3	0	0	0	0	0	3	6
	1.91	0.00	0.00	0.00	0.00	0.00	1.91	3.82
	50.00	0.00	0.00	0.00	0.00	0.00	50.00	
	10.34	0.00	0.00	0.00	0.00	0.00	6.82	
Total	29	16	10	15	39	4	44	157
	18.47	10.19	6.37	9.55	24.84	2.55	28.03	100.00

The following analysis (labeled ‘Approach 3’) transforms the original data with the ACECLUS procedure and creates a TYPE=ACE output data set that is used as an input data set for the cluster analysis. The following statements produce Output 115.1.6.

```
/*    Approach 3: data are transformed by PROC ACECLUS    */

title2 'Data are Transformed by PROC ACECLUS';
proc aceclus data=fish out=ace p=.02 noprint;
   var Length1 logLengthRatio Height Width Weight3;
run;
%FastFreq(ace);
```

Output 115.1.6 Data Are Transformed by PROC ACECLUS

Fish Measurement Data
Data are Transformed by PROC ACECLUS

The FREQ Procedure

Table of Species by CLUSTER

Frequency Percent Row Pct Col Pct	CLUSTER(Cluster)							
Species	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**Total**
Bream	13	0	0	0	0	0	21	34
	8.28	0.00	0.00	0.00	0.00	0.00	13.38	21.66
	38.24	0.00	0.00	0.00	0.00	0.00	61.76	
	44.83	0.00	0.00	0.00	0.00	0.00	47.73	
Parkki	2	3	0	0	6	0	0	11
	1.27	1.91	0.00	0.00	3.82	0.00	0.00	7.01
	18.18	27.27	0.00	0.00	54.55	0.00	0.00	
	6.90	18.75	0.00	0.00	15.38	0.00	0.00	
Perch	8	9	0	1	20	0	18	56
	5.10	5.73	0.00	0.64	12.74	0.00	11.46	35.67
	14.29	16.07	0.00	1.79	35.71	0.00	32.14	
	27.59	56.25	0.00	6.67	51.28	0.00	40.91	
Pike	0	0	10	0	1	4	2	17
	0.00	0.00	6.37	0.00	0.64	2.55	1.27	10.83
	0.00	0.00	58.82	0.00	5.88	23.53	11.76	
	0.00	0.00	100.00	0.00	2.56	100.00	4.55	
Roach	3	4	0	0	12	0	0	19
	1.91	2.55	0.00	0.00	7.64	0.00	0.00	12.10
	15.79	21.05	0.00	0.00	63.16	0.00	0.00	
	10.34	25.00	0.00	0.00	30.77	0.00	0.00	
Smelt	0	0	0	14	0	0	0	14
	0.00	0.00	0.00	8.92	0.00	0.00	0.00	8.92
	0.00	0.00	0.00	100.00	0.00	0.00	0.00	
	0.00	0.00	0.00	93.33	0.00	0.00	0.00	
Whitefish	3	0	0	0	0	0	3	6
	1.91	0.00	0.00	0.00	0.00	0.00	1.91	3.82
	50.00	0.00	0.00	0.00	0.00	0.00	50.00	
	10.34	0.00	0.00	0.00	0.00	0.00	6.82	
Total	29	16	10	15	39	4	44	157
	18.47	10.19	6.37	9.55	24.84	2.55	28.03	100.00

Table 115.4 displays a table summarizing each classification results. In this table, the first column represents the standardization method, the second column represents the number of clusters that the seven species are classified into, and the third column represents the total number of observations that are misclassified.

Table 115.4 Summary of Clustering Results

Method of Standardization	Number of Clusters	Misclassification
MEAN	5	71
MEDIAN	5	71
SUM	6	51
EUCLEN	6	45
USTD	6	45
STD	5	33
RANGE	7	32
MIDRANGE	7	32
MAXABS	7	26
IQR	5	28
MAD	4	35
ABW(5)	6	34
AWAVE(5)	6	29
AGK(0.14)	7	28
SPACING(0.14)	7	25
L(1)	6	41
L(1.5)	5	33
L(2)	5	33
untransformed	5	71
PROC ACECLUS	5	71

Consider the results displayed in Output 115.1.1. In that analysis, the method of standardization is STD, and the number of clusters and the number of misclassifications are computed as shown in Table 115.5.

Table 115.5 Computations of Numbers of Clusters and Misclassification When Standardization Method Is STD

Species	Cluster Number	Misclassification in Each Species
Bream	6	0
Parkki	5	0
Perch	7	29
Pike	1	0
Roach	7	0
Smelt	3	1
Whitefish	7	3

In Output 115.1.1, the bream species is classified as cluster 6 since all 34 bream are categorized into cluster 6 with no misclassification. A similar pattern is seen with the roach, parkki, pike, and smelt species.

For the whitefish species, two fish are categorized into cluster 2, one fish is categorized into cluster 4, and three fish are categorized into cluster 7. Because the majority of this species is categorized into cluster 7, it is recorded in Table 115.5 as being classified as cluster 7 with 3 misclassifications. A similar pattern is seen with the perch species: it is classified as cluster 7 with 29 misclassifications.

In summary, when the standardization method is STD, seven species of fish are classified into only five clusters and the total number of misclassified observations is 33.

The result of this analysis demonstrates that when variables are standardized by the STDIZE procedure with methods including RANGE, MIDRANGE, MAXABS, AGK(0.14), and SPACING(0.14), the FASTCLUS procedure produces the correct number of clusters and less misclassification than it does when other standardization methods are used. The SPACING method attains the best result, probably because the variables Length1 and Height both exhibit marked groupings (bimodality) in their distributions.

References

Art, D., Gnanadesikan, R., and Kettenring, J. R. (1982). "Data-Based Metrics for Cluster Analysis." *Utilitas Mathematica* 75–99.

Goodall, C. (1983). "M-Estimators of Location: An Outline of Theory." In *Understanding Robust and Exploratory Data Analysis*, edited by D. C. Hoaglin, M. Mosteller, and J. W. Tukey, 339–403. New York: John Wiley & Sons.

Iglewicz, B. (1983). "Robust Scale Estimators and Confidence Intervals for Location." In *Understanding Robust and Exploratory Data Analysis*, edited by D. C. Hoaglin, M. Mosteller, and J. W. Tukey, 404–431. New York: John Wiley & Sons.

Jain, R., and Chlamtac, I. (1985). "The P^2 Algorithm for Dynamic Calculation of Quantiles and Histograms without Storing Observations." *Communications of the ACM* 28:1076–1085.

Janssen, P., Marron, J. S., Veraverbeke, N., and Sarle, W. S. (1995). "Scale Measures for Bandwidth Selection." *Journal of Nonparametric Statistics* 5:359–380.

Milligan, G. W., and Cooper, M. C. (1987). "A Study of Variable Standardization." College of Administrative Science Working Paper Series, No. 87-63, Ohio State University.

Puranen, J. (1917). "Fish Catch data set (1917)." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Chapter 116
The STDRATE Procedure

Contents

Overview: STDRATE Procedure **9672**
Getting Started: STDRATE Procedure **9674**
Syntax: STDRATE Procedure **9681**
- PROC STDRATE Statement 9681
- BY Statement 9685
- POPULATION Statement 9685
- REFERENCE Statement 9687
- STRATA Statement 9688

Details: STDRATE Procedure **9690**
- Rate 9690
- Risk 9693
- Direct Standardization 9695
- Mantel-Haenszel Effect Estimation 9697
- Indirect Standardization and Standardized Morbidity/Mortality Ratio 9699
- Attributable Fraction and Population Attributable Fraction 9702
- Applicable Data Sets and Required Variables for Method Specifications 9706
- Applicable Confidence Limits for Rate and Risk Statistics 9707
- Table Output 9708
- ODS Table Names 9713
- Graphics Output 9713
- ODS Graphics 9714

Examples: STDRATE Procedure **9715**
- Example 116.1: Comparing Directly Standardized Rates 9715
- Example 116.2: Computing Mantel-Haenszel Risk Estimation 9720
- Example 116.3: Computing Attributable Fraction Estimates 9726
- Example 116.4: Displaying SMR Results from BY Groups 9732

References **9740**

Overview: STDRATE Procedure

Epidemiology is the study of the occurrence and distribution of health-related states or events in specified populations. Epidemiology also includes the study of the determinants that influence these states, and the application of this knowledge to control health problems (Porta 2008). It is a discipline that describes, quantifies, and postulates causal mechanisms for health phenomena in populations (Friss and Sellers 2009). A common goal is to establish relationships between various factors (such as exposure to a specific chemical) and the event outcomes (such as incidence of disease). But the measure of an association between an exposure and an event outcome can be biased due to confounding. That is, the association of the exposure to some other variables, such as age, influences the occurrence of the event outcome. With confounding, the usual effect between an exposure and an event outcome can be biased because some of the effect might be accounted for by other variables. For example, with an event rate discrepancy among different age groups of a population, the overall crude rate might not provide a useful summary statistic to compare populations.

One strategy to control confounding is stratification. In stratification, a population is divided into several subpopulations according to specific criteria for the confounding variables, such as age and sex groups. The effect of the exposure on the event outcome is estimated within each stratum, and then these stratum-specific effect estimates are combined into an overall estimate.

Two commonly used event frequency measures are rate and risk:

- A rate is a measure of the frequency with which an event occurs in a defined population in a specified period of time. It measures the change in one quantity per unit of another quantity. For example, an event rate measures how fast the events are occurring. That is, an event rate of a population over a specified time period can be defined as the number of new events divided by population-time (Kleinbaum, Kupper, and Morgenstern 1982, p. 100) over the same time period.
- A risk is the probability that an event occurs in a specified time period. It is assumed that only one event can occur in the time period for each subject or item. The overall crude risk of a population over a specified time period is the number of new events in the time period divided by the population size at the beginning of the time period.

Standardized overall rate and risk estimates based on stratum-specific estimates can be derived with the effects of confounding variables removed. These estimates provide useful summary statistics and allow valid comparison of the populations. There are two types of standardization:

- Direct standardization computes the weighted average of stratum-specific estimates in the study population, using the weights from a standard or reference population. This standardization is applicable when the study population is large enough to provide stable stratum-specific estimates. The directly standardized estimate is the overall crude rate in the study population if it has the same strata distribution as the reference population. When standardized estimates for different populations are derived by using the same reference population, the resulting estimates can also be compared by using the estimated difference and estimated ratio statistics.
- Indirect standardization computes the weighted average of stratum-specific estimates in the reference population, using the weights from the study population. The ratio of the overall crude rate or risk in the study population and the corresponding weighted estimate in the reference population is the

standardized morbidity ratio (SMR). This ratio is also the standardized mortality ratio if the event is death. SMR is used to compare rates or risks in the study and reference populations. With SMR, the indirectly standardized estimate is then computed as the product of the SMR and the overall crude estimate for the reference population. SMR and indirect standardization are applicable even when the study population is so small that the resulting stratum-specific rates are not stable.

Assuming that an effect, such as the rate difference between two populations, is homogeneous across strata, each stratum provides an estimate of the same effect. A pooled estimate of the effect can then be derived from these stratum-specific effect estimates. One way to estimate a homogeneous effect is the Mantel-Haenszel method (Greenland and Rothman 2008, p. 271). For a homogeneous rate difference effect between two populations, the Mantel-Haenszel estimate is identical to the difference between two directly standardized rates, but with weights derived from the two populations instead of from an explicitly specified reference population. The Mantel-Haenszel method can also be applied to other homogeneous effects between populations, such as the rate ratio, risk difference, and risk ratio.

The STDRATE procedure computes directly standardized rates and risks for study populations. For two study populations with the same reference population, PROC STDRATE compares directly standardized rates or risks from these two populations. For homogeneous effects across strata, PROC STDRATE computes Mantel-Haenszel estimates. The STDRATE procedure also computes indirectly standardized rates and risks, including SMR.

The attributable fraction measures the excess event rate or risk fraction in the exposed population that can be attributed to the exposure. The rate or risk ratio statistic is required in the attributable fraction computation, and the STDRATE procedure estimates the ratio by using either SMR or the rate ratio statistic in the Mantel-Haenszel estimates.

Although the STDRATE procedure provides useful summary standardized statistics, standardization is not a substitute for individual comparisons of stratum-specific estimates. PROC STDRATE provides summary statistics, such as rate and risk estimates and their confidence limits, in each stratum. In addition, PROC STDRATE also displays these stratum-specific statistics by using ODS Graphics.

Note that the term standardization has different meanings in other statistical applications. For example, the STDIZE procedure standardizes numeric variables in a SAS data set by subtracting a location measure and dividing by a scale measure.

Getting Started: STDRATE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/sratgs.sas.)

This example illustrates indirect standardization and uses the standardized mortality ratio to compare the death rate from skin cancer between people who live in the state of Florida and people who live in the United States as a whole.

The Florida_C43 data set contains the stratum-specific mortality information for skin cancer in year 2000 for the state of Florida (Florida Department of Health 2000, 2020). The variable Age is a grouping variable that forms the strata in the standardization, and the variables Event and PYear identify the number of events and total person-years, respectively. The COMMA9. format is specified in the DATA step to input numerical values that contain commas in PYear.

```
data Florida_C43;
   input Age $1-5 Event PYear:comma9.;
   datalines;
00-04    0    953,785
05-14    0  1,997,935
15-24    4  1,885,014
25-34   14  1,957,573
35-44   43  2,356,649
45-54   72  2,088,000
55-64   70  1,548,371
65-74  126  1,447,432
75-84  136  1,087,524
85+     73    335,944
;
```

The US_C43 data set contains the corresponding stratum-specific mortality information for the United States in year 2000 (Miniño et al. 2002; US Bureau of the Census 2011). The variable Age is the grouping variable, and the variables Event and PYear identify the number of events and the total person-years, respectively.

```
data US_C43;
   input Age $1-5 Event:comma5. PYear:comma10.;
   datalines;
00-04      0  19,175,798
05-14      1  41,077,577
15-24     41  39,183,891
25-34    186  39,892,024
35-44    626  45,148,527
45-54  1,199  37,677,952
55-64  1,303  24,274,684
65-74  1,637  18,390,986
75-84  1,624  12,361,180
85+      803   4,239,587
;
```

The following statements invoke the STDRATE procedure and request indirect standardization to compare death rates between the state of Florida and the United States:

```
ods graphics on;
proc stdrate data=Florida_C43 refdata=US_C43
             method=indirect
             stat=rate(mult=100000)
             plots=all
             ;
   population event=Event total=PYear;
   reference  event=Event total=PYear;
   strata Age / stats smr;
run;
```

The DATA= and REFDATA= options name the study data set and reference data set, respectively. The METHOD=INDIRECT option requests indirect standardization. The STAT=RATE option specifies the rate as the frequency measure for standardization, and the MULT=100000 suboption (which is the default) displays the rates per 100,000 person-years in the table output and graphics output. The PLOTS=ALL option requests all appropriate plots with indirect standardization.

The POPULATION statement specifies the options that are related to the study population, and the EVENT= and TOTAL= options specify variables for the number of events and person-years in the study population, respectively.

The REFERENCE statement specifies the options related to the reference population, and the EVENT= and TOTAL= options specify variables for the number of events and person-years in the reference population, respectively.

The STRATA statement lists the variable Age that forms the strata. The STATS option requests a strata information table that contains stratum-specific statistics such as rates, and the SMR option requests a table of stratum-specific SMR estimates.

The "Standardization Information" table in Figure 116.1 displays the standardization information.

Figure 116.1 Standardization Information

The STDRATE Procedure

Standardization Information	
Data Set	WORK.FLORIDA_C43
Reference Data Set	WORK.US_C43
Method	Indirect Standardization
Statistic	Rate
Number of Strata	10
Rate Multiplier	100000

The STATS option in the STRATA statement requests that the "Indirectly Standardized Strata Statistics" table in Figure 116.2 display the strata information and expected number of events at each stratum. The MULT=100000 suboption in the STAT=RATE option requests that crude rates per 100,000 person-years be displayed. The Expected Events column displays the expected number of events when the stratum-specific rates in the reference data set are applied to the corresponding person-years in the study data set.

Figure 116.2 Strata Information (Indirect Standardization)

The STDRATE Procedure

Indirectly Standardized Strata Statistics Rate Multiplier = 100000											
			Study Population						Reference Population		
			Population-Time						Population-Time		
Stratum Index	Age	Observed Events	Value	Proportion	Crude Rate	Standard Error	95% Normal Confidence Limits		Value	Proportion	Crude Rate
1	00-04	0	953785	0.0609	0.0000	0.00000	0.0000	0.0000	19175798	0.0681	0.0000
2	05-14	0	1997935	0.1276	0.0000	0.00000	0.0000	0.0000	41077577	0.1460	0.0024
3	15-24	4	1885014	0.1204	0.2122	0.10610	0.0042	0.4202	39183891	0.1392	0.1046
4	25-34	14	1957573	0.1250	0.7152	0.19114	0.3405	1.0898	39892024	0.1418	0.4663
5	35-44	43	2356649	0.1505	1.8246	0.27825	1.2793	2.3700	45148527	0.1604	1.3865
6	45-54	72	2088000	0.1333	3.4483	0.40638	2.6518	4.2448	37677952	0.1339	3.1822
7	55-64	70	1548371	0.0989	4.5209	0.54035	3.4618	5.5799	24274684	0.0863	5.3677
8	65-74	126	1447432	0.0924	8.7051	0.77551	7.1851	10.2250	18390986	0.0654	8.9011
9	75-84	136	1087524	0.0695	12.5055	1.07234	10.4037	14.6072	12361180	0.0439	13.1379
10	85+	73	335944	0.0215	21.7298	2.54328	16.7451	26.7146	4239587	0.0151	18.9405

Indirectly Standardized Strata Statistics Rate Multiplier = 100000	
Stratum Index	Expected Events
1	0.000
2	0.049
3	1.972
4	9.127
5	32.676
6	66.445
7	83.112
8	128.837
9	142.878
10	63.630

With ODS Graphics enabled, the PLOTS=ALL option displays all appropriate plots. With indirect standardization and a rate statistic, these plots include the strata distribution plot, the strata rate plot, and the strata SMR plot. By default, strata levels are displayed on the vertical axis for these plots.

The strata distribution plot displays proportions for stratum-specific person-years in the study and reference populations, as shown in Figure 116.3.

Figure 116.3 Strata Distribution Plot

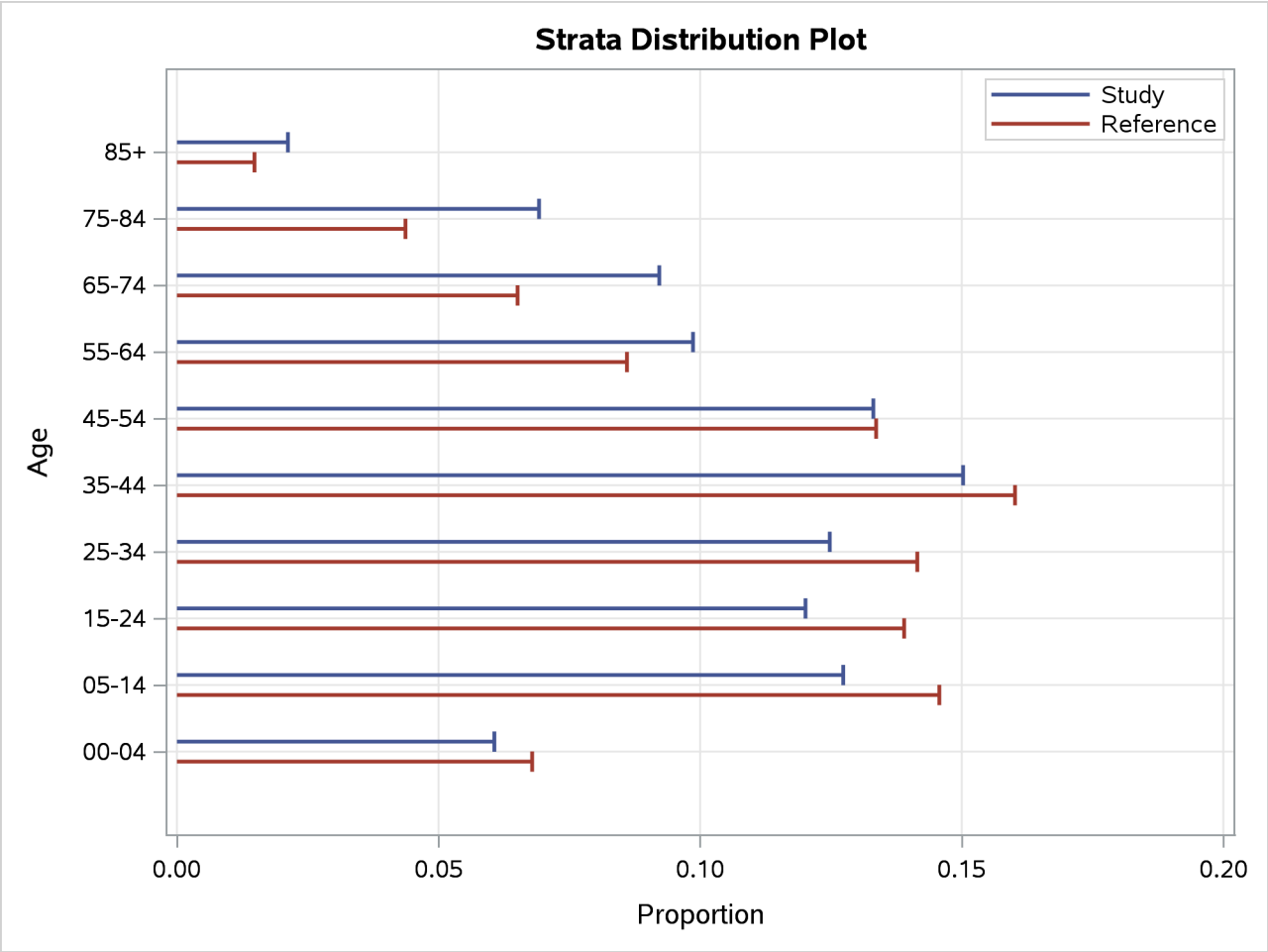

The strata distribution plot displays the proportions in the "Indirectly Standardized Strata Statistics" table in Figure 116.2. In the plot, the proportions of the study population are identified by the blue lines, and the proportions of the reference population are identified by the red lines. The plot shows that the study population has higher proportions in older age groups and lower proportions in younger age groups than the reference population.

The strata rate plot displays stratum-specific rate estimates in the study and reference populations, as shown in Figure 116.4. This plot displays the rate estimates in the "Indirectly Standardized Strata Statistics" table in Figure 116.2. In addition, the plot displays the confidence limits for the rate estimates in the study population and the overall crude rates for the two populations.

Figure 116.4 Strata Rate Plot

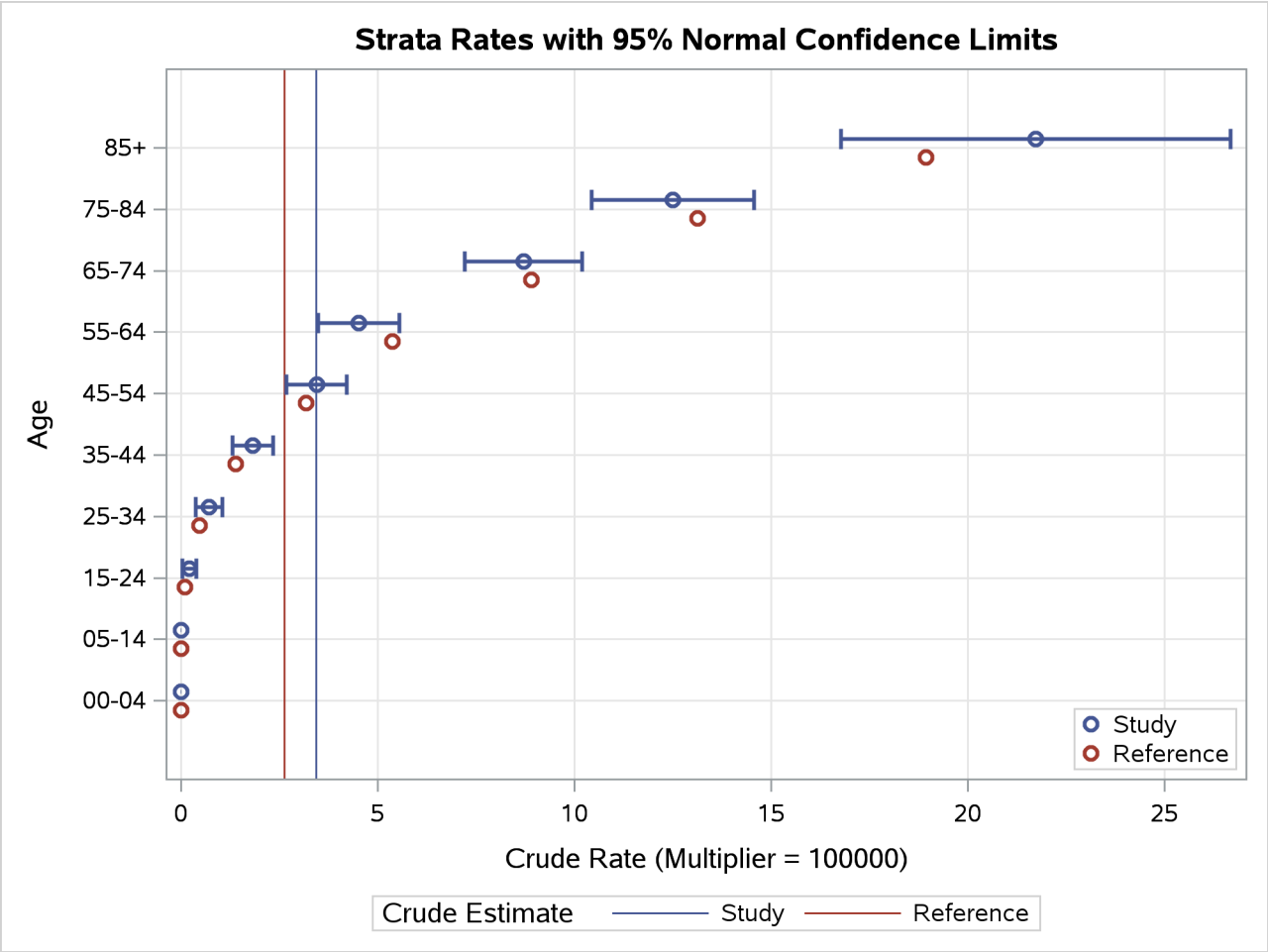

The SMR option in the STRATA statement requests that the "Strata SMR Estimates" table in Figure 116.5 display the strata SMR at each stratum. The MULT=100000 suboption in the STAT=RATE option requests that the reference rates per 100,000 person-years be displayed.

Figure 116.5 Strata SMR Information

Strata SMR Estimates
Rate Multiplier = 100000

		Study Population						95% Normal Confidence Limits	
Stratum Index	Age	Observed Events	Population-Time	Reference Crude Rate	Expected Events	SMR	Standard Error		
1	00-04	0	953785	0.0000	0.000	.	.	.	.
2	05-14	0	1997935	0.0024	0.049	0.0000	.	.	.
3	15-24	4	1885014	0.1046	1.972	2.0280	1.0140	0.0406	4.0154
4	25-34	14	1957573	0.4663	9.127	1.5339	0.4099	0.7304	2.3373
5	35-44	43	2356649	1.3865	32.676	1.3160	0.2007	0.9226	1.7093
6	45-54	72	2088000	3.1822	66.445	1.0836	0.1277	0.8333	1.3339
7	55-64	70	1548371	5.3677	83.112	0.8422	0.1007	0.6449	1.0395
8	65-74	126	1447432	8.9011	128.837	0.9780	0.0871	0.8072	1.1487
9	75-84	136	1087524	13.1379	142.878	0.9519	0.0816	0.7919	1.1118
10	85+	73	335944	18.9405	63.630	1.1473	0.1343	0.8841	1.4104

The "Strata SMR Estimates" table shows that although SMR is less than 1 only at three age strata (55–64, 65–74, and 75–84), these three strata contain about 60% of the total events.

The strata SMR plot displays stratum-specific SMR estimates with confidence limits, as shown in Figure 116.6. The plot displays the SMR estimates in the "Strata SMR Estimates" table in Figure 116.5.

Figure 116.6 Strata SMR Plot

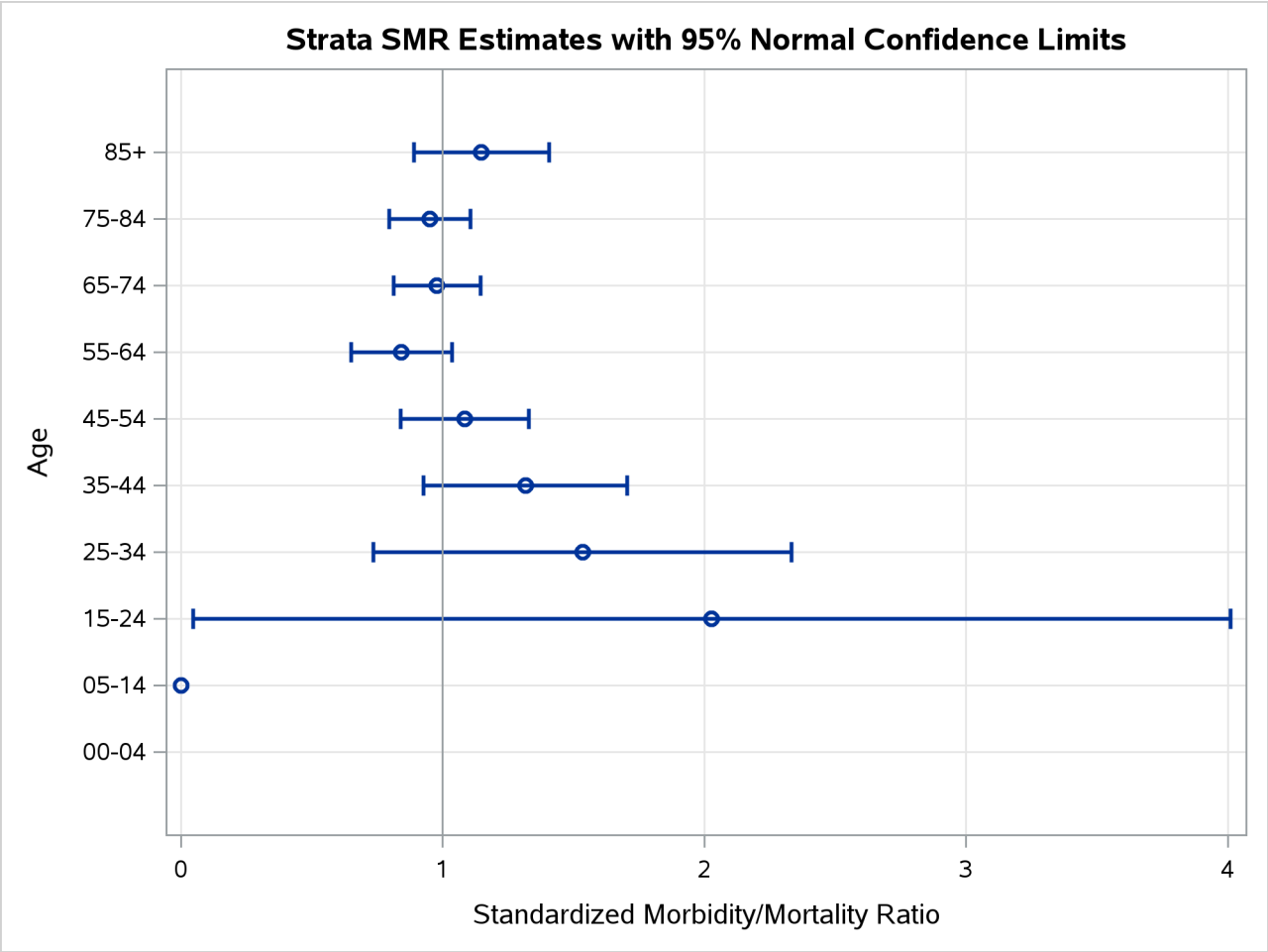

The METHOD=INDIRECT option requests that the "Standardized Morbidity/Mortality Ratio" table in Figure 116.7 be displayed. The table displays the SMR, its confidence limits, and the test for the null hypothesis H_0 : SMR $=$ 1. The default ALPHA=0.05 option requests that 95% confidence limits be constructed.

Figure 116.7 Standardized Morbidity/Mortality Ratio

Standardized Morbidity/Mortality Ratio							
Observed Events	Expected Events	SMR	Standard Error	95% Normal Confidence Limits		Z	Pr > |Z|
538	528.726	1.0175	0.0439	0.9316	1.1035	0.40	0.6893

The 95% normal confidence limits contain the null hypothesis value SMR $=$ 1, and the hypothesis of SMR $=$ 1 is not rejected at the $\alpha = 0.05$ level from the normal test.

The "Indirectly Standardized Rate Estimates" table in Figure 116.8 displays the indirectly standardized rate and related statistics.

Figure 116.8 Standardized Rate Estimates (Indirect Standardization)

Indirectly Standardized Rate Estimates Rate Multiplier = 100000									
Study Population						Standardized Rate			
Observed Events	Population-Time	Crude Rate	Reference Crude Rate	Expected Events	SMR	Estimate	Standard Error	95% Normal Confidence Limits	
538	15658227	3.4359	2.6366	528.726	1.0175	2.6829	0.1157	2.4562	2.9096

The indirectly standardized rate estimate is the product of the SMR and the crude rate estimate for the reference population. The table shows that although the crude rate in the state of Florida (3.4359) is 30% higher than the crude rate in the U.S. (2.6366), the resulting standardized rate (2.6829) is close to the crude rate in the U.S.

Syntax: STDRATE Procedure

The following statements are available in PROC STDRATE:

```
PROC STDRATE <options> ;
    BY variables ;
    POPULATION options ;
    REFERENCE options ;
    STRATA variables </ option> ;
```

The PROC STDRATE statement invokes the procedure, names the data sets, specifies the standardization method, and identifies the statistic for standardization. The BY statement requests separate analyses of groups defined by the BY variables. The required POPULATION statement specifies the rate or risk information in study populations, and the REFERENCE statement specifies the rate or risk information in the reference population. The STRATA statement lists the variables that form the strata.

The following sections describe the PROC STDRATE statement and then describe the other statements in alphabetical order.

PROC STDRATE Statement

```
PROC STDRATE <options> ;
```

Table 116.1 summarizes the *options* in the PROC STDRATE statement.

Table 116.1 Summary of PROC STDRATE Options

Option	Description
Input Data Sets	
DATA=	Names the SAS data set that contains the study populations
REFDATA=	Names the SAS data set that contains the reference population

Table 116.1 *continued*

Option	Description
Standardization Methods	
METHOD=	Specifies the method for standardization
STAT=	Specifies the statistic for standardization
EFFECT	Specifies the test to compare study populations for direct standardization and Mantel-Haenszel estimation
Displayed Output	
ALPHA=	Specifies the significance level for confidence intervals
CL=	Requests the confidence limits for the standardized estimates
PLOTS	Requests stratum-specific plots

You can specify the following *options* in the PROC STDRATE statement to compute standardized rates and risks in the procedure. They are listed in alphabetical order.

ALPHA=α

requests that confidence limits be constructed with confidence level $100(1-\alpha)\%$, where $0 < \alpha < 1$. The default is ALPHA=0.05. These confidence limits include confidence limits for the stratum-specific rates or risks, standardized rate and risk, standardized morbidity/mortality ratio, and population attributable rate and risk.

CL=GAMMA < (TYPE=AVERAGE | CONSERVATIVE) > | LOGNORMAL | NONE | NORMAL | POISSON

specifies the method to construct confidence limits for SMR and standardized rate and risk. You can specify the following values for this option:

GAMMA

requests confidence limits based on a gamma distribution for METHOD=DIRECT and METHOD=MH. This value applies only when STAT=RATE. You can specify the TYPE=CONSERVATIVE suboption to request conservative confidence limits that are based on a gamma distribution and were developed by Fay and Feuer (1997), or you can use the default TYPE=AVERAGE suboption to request modified confidence limits proposed by Tiwari, Clegg, and Zou (2006).

LOGNORMAL

requests confidence limits based on a lognormal distribution.

NONE

suppresses construction of confidence limits.

NORMAL

requests confidence limits based on a normal distribution.

POISSON
requests confidence limits based on a Poisson distribution. This value applies only when METHOD=INDIRECT.

The default is CL=NORMAL.

DATA=*SAS-data-set*
names the required SAS data set that contains the event information in the study populations.

EFFECT < =DIFF | RATIO >
displays a table of the effect estimate and associated confidence limits. This option applies only when METHOD=DIRECT with two study populations and when METHOD=MH, where two study populations are required.

EFFECT and EFFECT=RATIO display a test on the ratio effect of estimates between the study populations, and the EFFECT=DIFF option displays a test on the difference effect.

METHOD=DIRECT | INDIRECT < (AF) > | MH < (AF) >
M=DIRECT | INDIRECT < (AF) > | MH < (AF) >
specifies the required method for standardization. This option is required. The AF suboption (available only for METHOD=INDIRECT or METHOD=MH) requests the attributable fraction, which measures how much of the excess event rate or risk fraction in the exposed population is attributable to the exposure. This suboption also requests the population attributable fraction, which measures how much of the excess event rate or risk fraction in the total population is attributable to the exposure.

You can specify the following values:

DIRECT requests direct standardization.

INDIRECT requests indirect standardization. If you specify the AF suboption, the study population is treated as the exposed population and the reference population is treated as the unexposed population.

MH requests Mandel-Haenszel estimation. The order of the two study populations is indicated by the ORDER= suboption in the GROUP option in the POPULATION statement. If you specify the AF suboption, the exposed population is identified by the EXPOSED= suboption in the GROUP option in the POPULATION statement. If the EXPOSED= suboption is not specified, then the first study population is treated as the exposed population and the second study population is treated as the unexposed population.

PLOTS < (*global-options* **) > < =** *plot-request* **>**
PLOTS < (*global-options* **) > < = (** *plot-request* **< . . .** *plot-request* **>) >**
specifies options that control the details of the plots. The default is PLOTS=RATE for STAT=RATE and PLOTS=RISK for STAT=RISK.

You can specify the following *global-options*:

DISPLAY=INDEX | LEVEL
specifies tick mark values for the strata axis. DISPLAY=LEVEL displays strata levels on the strata axis, and DISPLAY=INDEX displays strata indices of sequential strata identification numbers on the strata axis. The default is DISPLAY=LEVEL.

ONLY
suppresses the default plots and displays only plots that are specifically requested.

STRATUM=HORIZONTAL | VERTICAL
controls the orientation of the plots. STRATUM=VERTICAL places the strata information on the vertical axis, and STRATUM=HORIZONTAL places the strata information on the horizontal axis. The default is STRATUM=VERTICAL.

You can specify the following *plot-requests*:

ALL
produces all appropriate plots.

DIST | DISTRIBUTION
displays a plot of the proportions for stratum-specific exposed time or sample size.

EFFECT
displays a plot of the stratum-specific effect estimates and associated confidence limits. This option applies only when METHOD=DIRECT with two study populations and when METHOD=MH, where two study populations are required. If the EFFECT=DIFF option is specified, the stratum-specific rate or risk difference effects are displayed. Otherwise, the stratum-specific rate or risk ratio effects are displayed.

NONE
suppresses all plots.

RATE
displays a plot of the stratum-specific rates and associated confidence limits. This option applies only when STAT=RATE. If a confidence limits method is specified in the STATS(CL=) option in the STRATA statement, that method is used to compute the confidence limits. Otherwise, the normal approximation is used.

RISK
displays a plot of the stratum-specific risks and associated confidence limits. This option applies only when STAT=RISK. If a confidence limits method is specified in the STATS(CL=) option in the STRATA statement, that method is used to compute the confidence limits. Otherwise, the normal approximation is used.

SMR
displays a plot of the stratum-specific SMR estimates and associated confidence limits. This option applies only when METHOD=INDIRECT. If a method is specified in the SMR(CL=) option in the STRATA statement, that method is used to compute the confidence limits. Otherwise, the normal approximation is used.

REFDATA=*SAS-data-set*
names the required SAS data set that contains the event information in the reference population.

STAT=RATE < (MULT =*c* **) >**
STAT=RISK
specifies the statistic for standardization. STAT=RATE computes standardized rates, and STAT=RISK computes standardized risks. The default is STAT=RATE.

The MULT= suboption in the STAT=RATE option specifies a power of 10 constant *c*, and requests that rates per *c* population-time units be displayed in the output tables and graphics. The default is MULT=100000, which specifies rates per 100,000 population-time units.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC STDRATE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the STDRATE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

POPULATION Statement

POPULATION *< options >* **;**

The required POPULATION statement specifies the information in the study data set. You can specify the following *options* in the POPULATION statement:

EVENT=*variable*
specifies the *variable* for the number of events in the study data set. The input values for the *variable* must be numeric and nonnegative.

GROUP < (*group-options*) > =*variable*
specifies the *variable* whose values identify the various populations. The GROUP= option is required when METHOD=MH and also applies when METHOD=DIRECT in the PROC STDRATE statement.

You can specify the following *group-options*:

EXPOSED='*group*'
identifies the exposed group in the derivation of the attributable fraction. This option applies only when you specify METHOD=MH(AF). If you do not specify the EXPOSED= option, the first study population, as indicated by the ORDER= option, is treated as the exposed population.

ORDER=DATA | FORMATTED | INTERNAL
specifies the order in which the values of the *variable* are to be displayed. You can specify the following values for the ORDER= suboption:

DATA	sorts by the order in which the values appear in the input data set.
FORMATTED	sorts by their external formatted values.
INTERNAL	sorts by the unformatted values, which yields the same order that the SORT procedure does.

By default, ORDER=INTERNAL. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

POPEVENT=*number*
specifies the total number of events in the study data set. This option applies only when METHOD=INDIRECT is specified in the PROC STDRATE statement and the total number of events is not available in the study data set.

RATE < (MULT=*c*) > = *variable*
specifies the *variable* for the observed rate in the study data set. The input values for the *variable* must be numeric and nonnegative. This option applies only when STAT=RATE is specified in the PROC STDRATE statement. The MULT=*c* suboption specifies a power of 10 constant *c* and requests that the rates per *c* population-time units be read from the data set. The default is the value of the MULT= suboption used in the STAT=RATE option in the PROC STDRATE statement.

RISK=*variable*
specifies the *variable* for the observed risk in the study data set. The input values for the *variable* must be numeric and between 0 and 1, inclusively. This option applies only when STAT=RISK is specified in the PROC STDRATE statement.

TOTAL=*variable*
specifies the *variable* for either the population-time (STAT=RATE) or the number of observations (STAT=RISK) in the study data set. The input values for the *variable* must be numeric and positive.

REFERENCE Statement

REFERENCE < *options* > **;**

The REFERENCE statement specifies the information in the reference data set. This statement is required when METHOD=DIRECT or METHOD=INDIRECT is specified in the PROC STDRATE statement.

You can specify the following *options* in the REFERENCE statement:

EVENT=*variable*
specifies the *variable* for the number of events in the reference data set. The input values for the *variable* must be numeric and nonnegative.

RATE < (MULT=*c* **) > =** *variable*
specifies the *variable* for the observed rate in the reference data set. The input values for the *variable* must be numeric and nonnegative. This option applies only when STAT=RATE is specified in the PROC STDRATE statement. The MULT=*c* suboption specifies a power of 10 constant *c* and requests that the rates per *c* population-time units be read from the data set. The default is the value of the MULT= suboption used in the STAT=RATE option in the PROC STDRATE statement.

RISK=*variable*
specifies the *variable* for the observed risk in the reference data set. The input values for the *variable* must be numeric and between 0 and 1, inclusively. This option applies only when STAT=RISK is specified in the PROC STDRATE statement.

TOTAL=*variable*
specifies the *variable* for either the population-time (STAT=RATE) or the number of observations (STAT=RISK) in the reference data set. The input values for the *variable* must be numeric and positive.

When METHOD=INDIRECT is specified in the PROC STDRATE statement, the overall reference population rate and risk are needed to compute indirect standardized rate and risk, respectively. If the information is not available in the reference data set, you can specify the following *options* for overall reference population rate and risk.

REFEVENT=*number*
specifies the total *number* of events in the reference data set.

REFRATE < (MULT=*c* **) > =** *number*
specifies the crude rate in the reference data set. This option applies only when STAT=RATE is specified in the PROC STDRATE statement. The MULT=*c* suboption specifies a power of 10 constant *c*, and the *number* is the crude rate per *c* population-time units in the data set. The default is the value of the MULT= suboption in the STAT=RATE option in the PROC STDRATE statement.

REFRISK=*number*
specifies the crude risk in the reference data set. This option applies only when STAT=RISK is specified in the PROC STDRATE statement.

REFTOTAL=*number*
specifies either the total population-time (STAT=RATE) or the total number of observations (STAT=RISK) in the reference data set.

When STAT=RATE, the REFRATE= option specifies the crude reference rate for the indirect standardized rate. If the REFRATE= option is not specified, the REFEVENT= and REFTOTAL options can be used to compute the crude reference rate. Similarly, when STAT=RISK, the REFRISK= option specifies the crude reference risk for the indirect standardized rate. If the REFRISK= option is not specified, the REFEVENT= and REFTOTAL options can be used to compute the crude reference risk.

STRATA Statement

STRATA *variables* < / *options* > **;**

The STRATA statement names *variables* that form the strata in the standardization. The combinations of categories of STRATA variables define the strata in the population.

The STRATA variables are one or more variables in all input data sets. These variables can be either character or numeric. The formatted values of the STRATA variables determine the levels. Thus, you can use formats to group values into levels. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference* for more information.

When the STRATA statement is not specified or the statement is specified without variables, all observations in a data set are treated as though they are from a single stratum.

You can specify the following *options* in the STRATA statement after a slash (/):

EFFECT
displays a table of the stratum-specific effect estimates and associated confidence limits. This option applies only when METHOD=DIRECT with two study populations and when METHOD=MH, where two study populations are required. If the EFFECT=DIFF option in the PROC STDRATE statement is specified, the stratum-specific rate or risk difference effects are displayed. Otherwise, the stratum-specific rate or risk ratio effects are displayed.

MISSING
treats missing values as a valid (nonmissing) category for all STRATA variables. When PROC STDRATE determines levels of a STRATA variable, an observation with missing values for that STRATA variable is excluded, unless the MISSING option is specified.

ORDER=DATA | FORMATTED | INTERNAL
specifies the order in which the values of the categorical variables are to be displayed. You can specify the following values for the ORDER= option:

DATA	sorts by the order in which the values appear in the input data set.
FORMATTED	sorts by their external formatted values.
INTERNAL	sorts by the unformatted values, which yields the same order that the SORT procedure does.

By default, ORDER=INTERNAL. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

STATS < (CL=LOGNORMAL | NONE | NORMAL | POISSON) >

displays tables for stratum-specific statistics such as stratum-specific rates and risks. You can specify the following values of the CL= suboption to request confidence limits for the rate or risk estimate in each stratum:

LOGNORMAL

requests confidence limits based on a lognormal approximation.

NONE

suppresses confidence limits.

NORMAL

requests confidence limits based on a normal approximation and also displays the standard error for the rate estimate in each stratum.

POISSON

requests confidence limits based on a Poisson distribution for stratum-specific rates. This values applies only when STAT=RATE in the PROC STDRATE statement.

The default is CL=NORMAL.

SMR < (CL=LOGNORMAL | NONE | NORMAL | POISSON) >

displays tables for stratum-specific SMR estimates. This option applies only when METHOD=INDIRECT is specified in the PROC STDRATE statement. You can specify the following values of the CL= suboption to request confidence limits for the SMR estimate in each stratum:

LOGNORMAL

requests confidence limits based on a lognormal approximation.

NONE

suppresses confidence limits.

NORMAL

requests confidence limits based on a normal approximation and also displays the standard error for the SMR estimate in each stratum.

POISSON

requests confidence limits based on a Poisson distribution for stratum-specific SMR estimates. This values applies only when STAT=RATE in the PROC STDRATE statement.

The default is CL=NORMAL.

Details: STDRATE Procedure

Rate

A major task in epidemiology is to compare event frequencies for groups of people. Both rate and risk are commonly used to measure event frequency in the comparison. Rate is a measure of change in one quantity per unit of another quantity. An event rate measures how fast the events are occurring. In contrast, an event risk is the probability that an event occurs over a specified follow-up time period.

An event rate of a population over a specified time period can be defined as the number of new events divided by the population-time of the population over the same time period,

$$\hat{\lambda} = \frac{d}{\mathcal{T}}$$

where d is the number of events and $\mathcal{T}$ is the population-time that is computed by adding up the time contributed by each subject in the population over the specified time period.

For a general population, the subsets (strata) might not be homogeneous enough to have a similar rate. Thus, the rate for each stratum should be computed separately to reflect this discrepancy. For a population that consists of K homogeneous strata (such as different age groups), the stratum-specific rate for the jth stratum in a population is computed as

$$\hat{\lambda}_j = \frac{d_j}{\mathcal{T}_j}$$

where d_j is the number of events and $\mathcal{T}_j$ is the population-time for subjects in the jth stratum of the population.

Assuming the number of events in the jth stratum, d_j, has a Poisson distribution, the variance of $\hat{\lambda}_j$ is

$$V(\hat{\lambda}_j) = V(\frac{d_j}{\mathcal{T}_j}) = \frac{1}{{\mathcal{T}_j}^2} V(d_j) = \frac{d_j}{\mathcal{T}_j^2} = \frac{\hat{\lambda}_j}{\mathcal{T}_j}$$

By using the method of statistical differentials (Elandt-Johnson and Johnson 1980, pp. 70–71), the variance of the logarithm of rate can be estimated by

$$V(\log(\hat{\lambda}_j)) = \frac{1}{\hat{\lambda}_j^2} V(\hat{\lambda}_j) = \frac{1}{\hat{\lambda}_j^2} \frac{\hat{\lambda}_j}{\mathcal{T}_j} = \frac{1}{\hat{\lambda}_j \mathcal{T}_j} = \frac{1}{d_j}$$

Because the rate value can be very small, especially for rare events, it is sometimes expressed in terms of the product of a multiplier and the rate itself. For example, a rate can be expressed as the number of events per 100,000 person-years.

Normal Distribution Confidence Interval for Rate

A $(1-\alpha)$ confidence interval for $\hat{\lambda}_j$ based on a normal distribution is given by

$$\left(\hat{\lambda}_j - z\sqrt{V(\hat{\lambda}_j)}\,,\ \ \hat{\lambda}_j + z\sqrt{V(\hat{\lambda}_j)} \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution.

Lognormal Distribution Confidence Interval for Rate

A $(1-\alpha)$ confidence interval for $\log(\hat{\lambda}_j)$ based on a normal distribution is given by

$$\left(\log(\hat{\lambda}_j) - z\sqrt{V(\log(\hat{\lambda}_j))}\,,\ \ \log(\hat{\lambda}_j) + z\sqrt{V(\log(\hat{\lambda}_j))} \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution and the variance $V(\log(\hat{\lambda}_j)) = 1/d_j$.

Thus, a $(1-\alpha)$ confidence interval for $\hat{\lambda}_j$ based on a lognormal distribution is given by

$$\left(\hat{\lambda}_j\, e^{-\frac{z}{\sqrt{d_j}}}\,,\ \ \hat{\lambda}_j\, e^{\frac{z}{\sqrt{d_j}}} \right)$$

Poisson Distribution Confidence Interval for Rate

Denote the $(\alpha/2)$ quantile for the χ^2 distribution with $2\,d_j$ degrees of freedom by

$$q_{lj} = \left(\chi^2_{2\,d_j}\right)^{-1}(\alpha/2)$$

Denote the $(1-\alpha/2)$ quantiles for the χ^2 distribution with $2(d_j+1)$ degrees of freedom by

$$q_{uj} = \left(\chi^2_{2\,(d_j+1)}\right)^{-1}(1-\alpha/2)$$

Then a $(1-\alpha)$ confidence interval for $\hat{\lambda}_j$ based on the χ^2 distribution is given by

$$\left(\frac{q_{lj}}{2\,\mathcal{T}_j}\,,\ \ \frac{q_{uj}}{2\,\mathcal{T}_j} \right)$$

Confidence Interval for Rate Difference Statistic

For rate estimates from two independent samples, $\hat{\lambda}_{1j}$ and $\hat{\lambda}_{2j}$, a $(1-\alpha)$ confidence interval for the rate difference $\hat{\lambda}_{dj} = \hat{\lambda}_{1j} - \hat{\lambda}_{2j}$ is

$$\left(\hat{\lambda}_{dj} - z\sqrt{V(\hat{\lambda}_{dj})}\,,\ \ \hat{\lambda}_{dj} + z\sqrt{V(\hat{\lambda}_{dj})} \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution and the variance

$$V(\hat{\lambda}_{dj}) = V(\hat{\lambda}_{1j}) + V(\hat{\lambda}_{2j})$$

Confidence Interval for Rate Ratio Statistic

For rate estimates from two independent samples, $\hat{\lambda}_{1j}$ and $\hat{\lambda}_{2j}$, a $(1-\alpha)$ confidence interval for the log rate ratio statistic $\log(\hat{\lambda}_{rj}) = \log(\hat{\lambda}_{1j}/\hat{\lambda}_{2j})$ is

$$\left(\log(\hat{\lambda}_{rj}) - z\sqrt{V(\log(\hat{\lambda}_{rj}))}\,,\ \ \log(\hat{\lambda}_{rj}) + z\sqrt{V(\log(\hat{\lambda}_{rj}))} \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution and the variance

$$V(\log(\hat{\lambda}_{rj})) = V(\log(\hat{\lambda}_{1j})) + V(\log(\hat{\lambda}_{2j}))$$

Thus, a $(1-\alpha)$ confidence interval for the rate ratio statistic $\hat{\lambda}_{rj}$ is given by

$$\left(\frac{\hat{\lambda}_{1j}}{\hat{\lambda}_{2j}}\, e^{-z\sqrt{V(\log(\hat{\lambda}_{rj}))}}\,,\ \ \frac{\hat{\lambda}_{1j}}{\hat{\lambda}_{2j}}\, e^{z\sqrt{V(\log(\hat{\lambda}_{rj}))}} \right)$$

Confidence Interval for Rate SMR

At stratum j, a stratum-specific standardized morbidity/mortality ratio is

$$\mathcal{R}_j = \frac{d_j}{\mathcal{E}_j}$$

where $\mathcal{E}_j$ is the expected number of events.

With the rate

$$\hat{\lambda}_j = \frac{d_j}{\mathcal{T}_j}$$

SMR can be expressed as

$$\mathcal{R}_j = \frac{\mathcal{T}_j}{\mathcal{E}_j}\,\hat{\lambda}_j$$

Thus, a $(1-\alpha)$ confidence interval for $\mathcal{R}_j$ is given by

$$\left(\frac{\mathcal{T}_j}{\mathcal{E}_j}\,\hat{\lambda}_{jl}\,,\ \ \frac{\mathcal{T}_j}{\mathcal{E}_j}\,\hat{\lambda}_{ju} \right)$$

where $(\hat{\lambda}_{jl}\,,\ \hat{\lambda}_{ju})$ is a $(1-\alpha)$ confidence interval for the rate $\hat{\lambda}_j$.

Risk

An event risk of a population over a specified time period can be defined as the number of new events in the follow-up time period divided by the event-free population size at the beginning of the time period,

$$\hat{\gamma} = \frac{d}{\mathcal{N}}$$

where $\mathcal{N}$ is the population size.

For a general population, the subsets (strata) might not be homogeneous enough to have a similar risk. Thus, the risk for each stratum should be computed separately to reflect this discrepancy. For a population that consists of K homogeneous strata (such as different age groups), the stratum-specific risk for the jth stratum in a population is computed as

$$\hat{\gamma}_j = \frac{d_j}{\mathcal{N}_j}$$

where $\mathcal{N}_j$ is the population size in the jth stratum of the population.

Assuming the number of events, d_j, has a binomial distribution, then a variance estimate of $\hat{\gamma}_j$ is

$$V(\hat{\gamma}_j) = \frac{\hat{\gamma}_j(1-\hat{\gamma}_j)}{\mathcal{N}_j}$$

By using the method of statistical differentials (Elandt-Johnson and Johnson 1980, pp. 70–71), the variance of the logarithm of risk can be estimated by

$$V(\log(\hat{\gamma}_j)) = \frac{1}{\hat{\gamma}_j^2}\, V(\hat{\gamma}_j) = \frac{1}{\hat{\gamma}_j^2}\,\frac{\hat{\gamma}_j\,(1-\hat{\gamma}_j)}{\mathcal{N}_j} = \frac{1-\hat{\gamma}_j}{\hat{\gamma}_j\,\mathcal{N}_j} = \frac{1}{d_j} - \frac{1}{\mathcal{N}_j}$$

Normal Distribution Confidence Interval for Risk

A $(1-\alpha)$ confidence interval for $\hat{\gamma}_j$ based on a normal distribution is given by

$$\left(\hat{\gamma}_j - z\sqrt{V(\hat{\gamma}_j)}\,,\ \ \hat{\gamma}_j + z\sqrt{V(\hat{\gamma}_j)}\right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution.

Lognormal Distribution Confidence Interval for Risk

A $(1-\alpha)$ confidence interval for $\log(\hat{\gamma}_j)$ based on a normal distribution is given by

$$\left(\log(\hat{\gamma}_j) - z\sqrt{V(\log(\hat{\gamma}_j))}\,,\ \ \log(\hat{\gamma}_j) + z\sqrt{V(\log(\hat{\gamma}_j))}\right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution and the variance $V(\log(\hat{\gamma}_j)) = 1/d_j - 1/\mathcal{N}_j$.

Thus, a $(1-\alpha)$ confidence interval for $\hat{\gamma}_j$ based on a lognormal distribution is given by

$$\left(\hat{\gamma}_j\, e^{-z\sqrt{\frac{1}{d_j}-\frac{1}{\mathcal{N}_j}}}\,,\ \ \hat{\gamma}_j\, e^{z\sqrt{\frac{1}{d_j}-\frac{1}{\mathcal{N}_j}}}\right)$$

Confidence Interval for Risk Difference Statistic

For rate estimates from two independent samples, $\hat{\gamma}_{1j}$ and $\hat{\gamma}_{2j}$, a $(1-\alpha)$ confidence interval for the risk difference $\hat{\gamma}_{dj} = \hat{\gamma}_{1j} - \hat{\gamma}_{2j}$ is

$$\left(\hat{\gamma}_{dj} - z\sqrt{V(\hat{\gamma}_{dj})}\,,\ \ \hat{\gamma}_{dj} + z\sqrt{V(\hat{\gamma}_{dj})} \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution and the variance

$$V(\hat{\gamma}_{dj}) = V(\hat{\gamma}_{1j}) + V(\hat{\gamma}_{2j})$$

Confidence Interval for Risk Ratio Statistic

For rate estimates from two independent samples, $\hat{\gamma}_{1j}$ and $\hat{\gamma}_{2j}$, a $(1-\alpha)$ confidence interval for the log risk ratio statistic $\log(\hat{\gamma}_{rj}) = \log(\hat{\gamma}_{1j}/\hat{\gamma}_{2j})$ is

$$\left(\log(\hat{\gamma}_{rj}) - z\sqrt{V(\log(\hat{\gamma}_{rj}))}\,,\ \ \log(\hat{\gamma}_{rj}) + z\sqrt{V(\log(\hat{\gamma}_{rj}))} \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution and the variance

$$V(\log(\hat{\gamma}_{rj}) = V(\log(\hat{\gamma}_{1j})) + V(\log(\hat{\gamma}_{2j}))$$

Thus, a $(1-\alpha)$ confidence interval for the risk ratio statistic $\hat{\gamma}_{rj}$ is given by

$$\left(\frac{\hat{\gamma}_{1j}}{\hat{\gamma}_{2j}}\, e^{-z\sqrt{V(\log(\hat{\gamma}_{rj}))}}\,,\ \ \frac{\hat{\gamma}_{1j}}{\hat{\gamma}_{2j}}\, e^{z\sqrt{V(\log(\hat{\gamma}_{rj}))}} \right)$$

Confidence Interval for Risk SMR

At stratum j, a stratum-specific standardized morbidity/mortality ratio is

$$\mathcal{R}_j = \frac{d_j}{\mathcal{E}_j}$$

where $\mathcal{E}_j$ is the expected number of events.

With the risk

$$\hat{\gamma}_j = \frac{d_j}{\mathcal{N}_j}$$

SMR can be expressed as

$$\mathcal{R}_j = \frac{\mathcal{N}_j}{\mathcal{E}_j}\,\hat{\gamma}_j$$

Thus, a $(1-\alpha)$ confidence interval for $\mathcal{R}_j$ is given by

$$\left(\frac{\mathcal{N}_j}{\mathcal{E}_j}\,\hat{\gamma}_{jl}\,,\ \ \frac{\mathcal{N}_j}{\mathcal{E}_j}\,\hat{\gamma}_{ju} \right)$$

where $(\hat{\gamma}_{jl}\,,\ \hat{\gamma}_{ju})$ is a $(1-\alpha)$ confidence interval for the risk $\hat{\gamma}_j$.

Direct Standardization

Direct standardization uses the weights from a reference population to compute the standardized rate of a study group as the weighted average of stratum-specific rates in the study population. The standardized rate is computed as

$$\hat{\lambda}_{ds} = \frac{\sum_j \mathcal{T}_{rj}\, \hat{\lambda}_{sj}}{\mathcal{T}_r}$$

where $\hat{\lambda}_{sj}$ is the rate in the *j*th stratum of the study population, $\mathcal{T}_{rj}$ is the population-time in the *j*th stratum of the reference population, and $\mathcal{T}_r = \sum_k \mathcal{T}_{rk}$ is the population-time in the reference population.

Similarly, direct standardization uses the weights from a reference population to compute the standardized risk of a study group as the weighted average of stratum-specific risks in the study population. The standardized risk is computed as

$$\hat{\gamma}_{ds} = \frac{\sum_j \mathcal{N}_{rj}\, \hat{\gamma}_{sj}}{\mathcal{N}_r}$$

where $\hat{\gamma}_{sj}$ is the risk in the *j*th stratum of the study population, $\mathcal{N}_{rj}$ is the number of observations in the *j*th stratum of the reference population, and $\mathcal{N}_r = \sum_k \mathcal{N}_{rk}$ is the total number of observations in the reference population.

That is, the directly standardized rate and risk of a study population are weighted averages of the stratum-specific rates and risks, respectively, where the weights are the corresponding strata population sizes in the reference population. The direct standardization can be used when the study population is large enough to provide stable stratum-specific rates or risks. When the same reference population is used for multiple study populations, directly standardized rates and risks provide valid comparisons between study populations.

The variances of the directly standardized rate and risk are

$$V(\hat{\lambda}_{ds}) = V\left(\frac{\sum_j \mathcal{T}_{rj}\, \hat{\lambda}_{sj}}{\mathcal{T}_r}\right) = \frac{\sum_j \mathcal{T}_{rj}^2\, V(\hat{\lambda}_{sj})}{\mathcal{T}_r^2}$$

$$V(\hat{\gamma}_{ds}) = V\left(\frac{\sum_j \mathcal{N}_{rj}\, \hat{\gamma}_{sj}}{\mathcal{N}_r}\right) = \frac{\sum_j \mathcal{N}_{rj}^2\, V(\hat{\gamma}_{sj})}{\mathcal{N}_r^2}$$

By using the method of statistical differentials (Elandt-Johnson and Johnson 1980, pp. 70–71), the variance of the logarithm of directly standardized rate and risk can be estimated by

$$V(\log(\hat{\lambda}_{ds})) = \frac{1}{\hat{\lambda}_{ds}^2}\, V(\hat{\lambda}_{ds})$$

$$V(\log(\hat{\gamma}_{ds})) = \frac{1}{\hat{\gamma}_{ds}^2}\, V(\hat{\gamma}_{ds})$$

The confidence intervals for $\hat{\lambda}_{ds}$ and $\hat{\gamma}_{ds}$ can be constructed based on normal and lognormal distributions. A gamma distribution confidence interval can also be constructed for $\hat{\lambda}_{ds}$.

In the next four subsections, $\beta = \lambda$ denotes the rate statistic and $\beta = \gamma$ denotes the risk statistic.

Normal Distribution Confidence Intervals for Standardized Rate and Risk

A $(1-\alpha)$ confidence interval for $\hat{\beta}_{ds}$ based on a normal distribution is then given by

$$\left(\hat{\beta}_{ds} - z\sqrt{V(\hat{\beta}_{ds})}\,,\ \ \hat{\beta}_{ds} + z\sqrt{V(\hat{\beta}_{ds})} \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution.

Lognormal Distribution Confidence Intervals for Standardized Rate and Risk

A $(1-\alpha)$ confidence interval for $\log(\hat{\beta}_{ds})$ based on a normal distribution is given by

$$\left(\log(\hat{\beta}_{ds}) - z\sqrt{V(\log(\hat{\beta}_{ds}))}\,,\ \ \log(\hat{\beta}_{ds}) + z\sqrt{V(\log(\hat{\beta}_{ds}))} \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution.

Thus, a $(1-\alpha)$ confidence interval for $\hat{\beta}_{ds}$ based on a lognormal distribution is given by

$$\left(\hat{\beta}_{ds}\, e^{-z\sqrt{V(\log(\hat{\beta}_{ds}))}}\,,\ \ \hat{\beta}_{ds}\, e^{z\sqrt{V(\log(\hat{\beta}_{ds}))}} \right)$$

Gamma Distribution Confidence Interval for Standardized Rate

Fay and Feuer (1997) use the relationship between the Poisson and gamma distributions to derive approximate confidence intervals for the standardized rate $\hat{\lambda}_{ds}$ based on the gamma distribution. As in the construction of the asymptotic normal confidence intervals, it is assumed that the number of events has a Poisson distribution, and the standardized rate is a weighted sum of independent Poisson random variables. A confidence interval for $\hat{\lambda}_{ds}$ is then given by

$$\left(\frac{v}{2\hat{\lambda}_{ds}}\,(\chi^2)^{-1}_{\frac{2\hat{\lambda}^2_{ds}}{v}}\left(\frac{\alpha}{2}\right)\,,\ \ \frac{v+w_x^2}{2(\hat{\lambda}_{ds}+w_x)}\,(\chi^2)^{-1}_{\frac{2(\hat{\lambda}_{ds}+w_x)^2}{v+w_x^2}}\left(1-\frac{\alpha}{2}\right) \right)$$

where

$$v = \sum_j w_j^2\,\hat{\lambda}_{sj}\,\mathcal{T}_{sj}$$

$$w_j = \frac{\mathcal{T}_{rj}}{\mathcal{T}_r}\,\frac{1}{\mathcal{T}_{sj}}$$

and w_x is the maximum w_j.

Tiwari, Clegg, and Zou (2006) propose a less conservative confidence interval for $\hat{\lambda}_{ds}$ with a different upper confidence limit,

$$\left(\frac{v}{2\hat{\lambda}_{ds}}\,(\chi^2)^{-1}_{\frac{2\hat{\lambda}^2_{ds}}{v}}\left(\frac{\alpha}{2}\right)\,,\ \ \frac{v+w_{2m}}{2(\hat{\lambda}_{ds}+w_m)}\,(\chi^2)^{-1}_{\frac{2(\hat{\lambda}_{ds}+w_m)^2}{v+w_{2m}}}\left(1-\frac{\alpha}{2}\right) \right)$$

where w_m is the average w_j and w_{2m} is the average w_j^2.

Comparing Standardized Rates and Comparing Standardized Risks

By using the same reference population, two directly standardized rates or risks from different populations can be compared. Both the difference and ratio statistics can be used in the comparison. Assume that $\hat{\beta}_1$ and $\hat{\beta}_2$ are directly standardized rates or risks for two populations with variances $V(\hat{\beta}_1)$ and $V(\hat{\beta}_2)$, respectively. The difference test assumes that the difference statistic

$$\hat{\beta}_1 - \hat{\beta}_2$$

has a normal distribution with mean 0 under the null hypothesis $H_0 : \beta_1 = \beta_2$. The variance is given by

$$V(\hat{\beta}_1 - \hat{\beta}_2) = V(\hat{\beta}_1) + V(\hat{\beta}_2)$$

The ratio test assumes that the log ratio statistic,

$$\log\left(\frac{\hat{\beta}_1}{\hat{\beta}_2}\right)$$

has a normal distribution with mean 0 under the null hypothesis $H_0 : \beta_1 = \beta_2$, or equivalently, $\log(\beta_1/\beta_2) = 0$. An estimated variance is given by

$$V\left(\log\left(\frac{\hat{\beta}_1}{\hat{\beta}_2}\right)\right) = V(\log(\hat{\beta}_1)) + V(\log(\hat{\beta}_2)) = \frac{1}{\hat{\beta}_1^2}\, V(\hat{\beta}_1) \;+\; \frac{1}{\hat{\beta}_2^2}\, V(\hat{\beta}_2)$$

Mantel-Haenszel Effect Estimation

In direct standardization, the derived standardized rates and risks in a study population are the weighted average of the stratum-specific rates and risks in the population, respectively, where the weights are given by the population-time for standardized rate and the number of observations for standardized risk in a reference population.

Assuming that an effect, such as rate difference, rate ratio, risk difference, and risk ratio between two populations, is homogeneous across strata, the Mantel-Haenszel estimates of this effect can be constructed from directly standardized rates or risks in the two populations, where the weights are constructed from the stratum-specific population-times for rate and number of observations for risk of the two populations.

That is, for population k, k=1 and 2, the standardized rate and risk are

$$\hat{\lambda}_k = \frac{\sum_j w_j\, \hat{\lambda}_{kj}}{\sum_j w_j} \qquad \text{and} \qquad \hat{\gamma}_k = \frac{\sum_j w_j\, \hat{\gamma}_{kj}}{\sum_j w_j}$$

where the weights are

$$w_j = \frac{\mathcal{T}_{1j}\, \mathcal{T}_{2j}}{\mathcal{T}_{1j} + \mathcal{T}_{2j}}$$

for standardized rate, and

$$w_j = \frac{\mathcal{N}_{1j}\, \mathcal{N}_{2j}}{\mathcal{N}_{1j} + \mathcal{N}_{2j}}$$

for standardized risk.

Rate and Risk Difference Statistics

Denote $\beta = \lambda$ for rate and $\beta = \gamma$ for risk. The variance is

$$V(\hat{\beta}_k) = V\left(\frac{\sum_j w_j \hat{\beta}_{kj}}{\sum_j w_j}\right) = \frac{1}{(\sum_j w_j)^2} \sum_j w_j^2 V(\hat{\beta}_{kj})$$

The Mantel-Haenszel difference statistic is

$$\hat{\beta}_1 - \hat{\beta}_2$$

with variance

$$V(\hat{\beta}_1 - \hat{\beta}_2) = V(\hat{\beta}_1) + V(\hat{\beta}_2)$$

Under the null hypothesis $H_0 : \beta_1 = \beta_2$, the difference statistic $\hat{\beta}_1 - \hat{\beta}_2$ has a normal distribution with mean 0.

Rate Ratio Statistic

The Mantel-Haenszel rate ratio statistic is $\hat{\lambda}_1/\hat{\lambda}_2$, and the log ratio statistic is

$$\log\left(\frac{\hat{\lambda}_1}{\hat{\lambda}_2}\right)$$

Under the null hypothesis $H_0 : \lambda_1 = \lambda_2$ (or equivalently, $\log(\lambda_1/\lambda_2) = 0$), the log ratio statistic has a normal distribution with mean 0 and variance

$$V\left(\log\left(\frac{\hat{\lambda}_1}{\hat{\lambda}_2}\right)\right) = \frac{\sum_j w_j \hat{\lambda}_{pj}}{(\sum_j w_j \hat{\lambda}_{1j})\,(\sum_j w_j \hat{\lambda}_{2j})}$$

where

$$\hat{\lambda}_{pj} = \frac{d_{1j} + d_{2j}}{\mathcal{T}_{1j} + \mathcal{T}_{2j}}$$

is the combined rate estimate in stratum j under the null hypothesis of equal rates (Greenland and Robins 1985; Greenland and Rothman 2008, p. 273).

Risk Ratio Statistic

The Mantel-Haenszel risk ratio statistic is $\hat{\gamma}_1/\hat{\gamma}_2$, and the log ratio statistic is

$$\log\left(\frac{\hat{\gamma}_1}{\hat{\gamma}_2}\right)$$

Under the null hypothesis $H_0 : \gamma_1 = \gamma_2$ (or equivalently, $\log(\gamma_1/\gamma_2) = 0$), the log ratio statistic has a normal distribution with mean 0 and variance

$$V\left(\log\left(\frac{\hat{\gamma}_1}{\hat{\gamma}_2}\right)\right) = \frac{\sum_j w_j (\hat{\gamma}_{pj} - \hat{\gamma}_{1j}\hat{\gamma}_{2j})}{(\sum_j w_j \hat{\gamma}_{1j})\,(\sum_j w_j \hat{\gamma}_{2j})}$$

where

$$\hat{\gamma}_{pj} = \frac{d_{1j} + d_{2j}}{\mathcal{N}_{1j} + \mathcal{N}_{2j}}$$

is the combined risk estimate in stratum j under the null hypothesis of equal risks (Greenland and Robins 1985; Greenland and Rothman 2008, p. 275).

Indirect Standardization and Standardized Morbidity/Mortality Ratio

Indirect standardization compares the rates of the study and reference populations by applying the stratum-specific rates in the reference population to the study population, where the stratum-specific rates might not be reliable.

The expected number of events in the study population is

$$\mathcal{E} = \sum_j \mathcal{T}_{sj}\ \hat{\lambda}_{rj}$$

where $\mathcal{T}_{sj}$ is the population-time in the jth stratum of the study population and $\hat{\lambda}_{rj}$ is the rate in the jth stratum of the reference population.

With the expected number of events, $\mathcal{E}$, the standardized morbidity ratio or standardized mortality ratio can be expressed as

$$\mathcal{R}_{sm} = \frac{\mathcal{D}}{\mathcal{E}}$$

where $\mathcal{D}$ is the observed number of events (Breslow and Day 1987, p. 65).

The ratio $\mathcal{R}_{sm} > 1$ indicates that the mortality rate or risk in the study population is larger than the estimate in the reference population, and $\mathcal{R}_{sm} < 1$ indicates that the mortality rate or risk in the study population is smaller than the estimate in the reference population.

With the ratio $\mathcal{R}_{sm}$, an indirectly standardized rate for the study population is computed as

$$\hat{\lambda}_{is} = \mathcal{R}_{sm}\ \hat{\lambda}_r$$

where $\hat{\lambda}_r$ is the overall crude rate in the reference population.

Similarly, to compare the risks of the study and reference populations, the stratum-specific risks in the reference population are used to compute the expected number of events in the study population

$$\mathcal{E} = \sum_j \mathcal{N}_{sj}\ \hat{\gamma}_{rj}$$

where $\mathcal{N}_{sj}$ is the number of observations in the jth stratum of the study population and $\hat{\gamma}_{rj}$ is the risk in the jth stratum of the reference population.

Also, with the standardized morbidity ratio $\mathcal{R}_{sm} = \mathcal{D}/\mathcal{E}$, an indirectly standardized risk for the study population is computed as

$$\hat{\gamma}_{is} = \mathcal{R}_{sm}\ \hat{\gamma}_r$$

where $\hat{\gamma}_r$ is the overall crude risk in the reference population.

The observed number of events in the study population is $\mathcal{D} = \sum_j d_{sj}$, where d_{sj} is the number of events in the jth stratum of the population. For the rate estimate, if d_{sj} has a Poisson distribution, then the variance of the standardized mortality ratio $\mathcal{R}_{sm} = \mathcal{D}/\mathcal{E}$ is

$$V(\mathcal{R}_{sm}) = \frac{1}{\mathcal{E}^2} \sum_j V(d_{sj}) = \frac{1}{\mathcal{E}^2} \sum_j d_{sj} = \frac{\mathcal{D}}{\mathcal{E}^2} = \frac{\mathcal{R}_{sm}}{\mathcal{E}}$$

For the risk estimate, if d_{sj} has a binomial distribution, then the variance of $\mathcal{R}_{sm} = \mathcal{D}/\mathcal{E}$ is

$$V(\mathcal{R}_{sm}) = V\left(\frac{1}{\mathcal{E}} \sum_j d_{sj}\right) = \frac{1}{\mathcal{E}^2} \sum_j V(d_{sj}) = \frac{1}{\mathcal{E}^2} \sum_j \mathcal{N}_{sj}^2 V(\hat{\gamma}_{sj})$$

where

$$V(\hat{\gamma}_{sj}) = \frac{\hat{\gamma}_{sj}(1-\hat{\gamma}_{sj})}{\mathcal{N}_{sj}}$$

By using the method of statistical differentials (Elandt-Johnson and Johnson 1980, pp. 70–71), the variance of the logarithm of $\mathcal{R}_{sm}$ can be estimated by

$$V(\log(\mathcal{R}_{sm})) = \frac{1}{\mathcal{R}_{sm}^2} V(\mathcal{R}_{sm})$$

For the rate estimate,

$$V(\log(\mathcal{R}_{sm})) = \frac{1}{\mathcal{R}_{sm}^2} V(\mathcal{R}_{sm}) = \frac{1}{\mathcal{R}_{sm}^2} \frac{\mathcal{R}_{sm}}{\mathcal{E}} = \frac{1}{\mathcal{R}_{sm}} \frac{1}{\mathcal{E}} = \frac{1}{\mathcal{D}}$$

The confidence intervals for $\mathcal{R}_{sm}$ can be constructed based on normal, lognormal, and Poisson distributions.

Normal Distribution Confidence Interval for SMR

A $(1-\alpha)$ confidence interval for $\mathcal{R}_{sm}$ based on a normal distribution is given by

$$(\mathcal{R}_l,\ \mathcal{R}_u) = \left(\ \mathcal{R}_{sm} - z\sqrt{V(\mathcal{R}_{sm})},\ \ \mathcal{R}_{sm} + z\sqrt{V(\mathcal{R}_{sm})}\ \right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution.

A test statistic for the null hypothesis H_0 : SMR $= 1$ is then given by

$$\frac{\mathcal{R}_{sm} - 1}{\sqrt{V(\mathcal{R}_{sm})}}$$

The test statistic has an approximate standard normal distribution under H_0.

Lognormal Distribution Confidence Interval for SMR

A $(1-\alpha)$ confidence interval for $\log(\mathcal{R}_{sm})$ based on a normal distribution is given by

$$\left(\log(\mathcal{R}_{sm}) - z\,\sqrt{V(\log(\mathcal{R}_{sm}))}\,,\ \ \log(\mathcal{R}_{sm}) + z\,\sqrt{V(\log(\mathcal{R}_{sm}))}\,\right)$$

where $z = \Phi^{-1}(1-\alpha/2)$ is the $(1-\alpha/2)$ quantile of the standard normal distribution.

Thus, a $(1-\alpha)$ confidence interval for $\mathcal{R}_{sm}$ based on a lognormal distribution is given by

$$\left(\mathcal{R}_{sm}\, e^{-z\sqrt{V(\log(\mathcal{R}_{sm}))}}\,,\ \ \mathcal{R}_{sm}\, e^{z\sqrt{V(\log(\mathcal{R}_{sm}))}}\right)$$

A test statistic for the null hypothesis $H_0 : \text{SMR} = 1$ is then given by

$$\frac{\log(\mathcal{R}_{sm})}{\sqrt{V(\log(\mathcal{R}_{sm}))}}$$

The test statistic has an approximate standard normal distribution under H_0.

Poisson Distribution Confidence Interval for SMR

Denote the $(\alpha/2)$ quantile for the χ^2 distribution with $2\mathcal{D}$ degrees of freedom by

$$q_l = \left(\chi^2_{2\mathcal{D}}\right)^{-1}(\alpha/2)$$

Denote the $(1-\alpha/2)$ quantiles for the χ^2 distribution with $2(\mathcal{D}+1)$ degrees of freedom by

$$q_u = \left(\chi^2_{2(\mathcal{D}+1)}\right)^{-1}(1-\alpha/2)$$

Then a $(1-\alpha)$ confidence interval for $\mathcal{R}_{sm}$ based on the χ^2 distribution is given by

$$(\mathcal{R}_l,\ \mathcal{R}_u) = \left(\frac{q_l}{2\,\mathcal{E}}\,,\ \ \frac{q_u}{2\,\mathcal{E}}\right)$$

A p-value for the test of the null hypothesis $H_0 : \text{SMR} = 1$ is given by

$$2\min\left(\sum_{k=0}^{\mathcal{D}} \frac{e^{-\mathcal{E}}\mathcal{E}^k}{k!},\ \sum_{k=\mathcal{D}}^{\infty} \frac{e^{-\mathcal{E}}\mathcal{E}^k}{k!} \right)$$

Indirectly Standardized Rate and Its Confidence Interval

With a rate-standardized mortality ratio $\mathcal{R}_{sm}$, an indirectly standardized rate for the study population is computed as

$$\hat{\lambda}_{is} = \mathcal{R}_{sm}\,\hat{\lambda}_r$$

where $\hat{\lambda}_r$ is the overall crude rate in the reference population.

The $(1-\alpha/2)$ confidence intervals for $\hat{\lambda}_{is}$ can be constructed as

$$(\mathcal{R}_l\,\hat{\lambda}_r,\ \mathcal{R}_u\,\hat{\lambda}_r)$$

where $(\mathcal{R}_l,\ \mathcal{R}_u)$ is the confidence interval for $\mathcal{R}_{sm}$.

Indirectly Standardized Risk and Its Confidence Interval

With a risk-standardized mortality ratio $\mathcal{R}_{sm}$, an indirectly standardized risk for the study population is computed as

$$\hat{\gamma}_{is} = \mathcal{R}_{sm}\ \hat{\gamma}_r$$

where $\hat{\gamma}_r$ is the overall crude risk in the reference population.

The $(1-\alpha/2)$ confidence intervals for $\hat{\gamma}_{is}$ can be constructed as

$$(\mathcal{R}_l\ \hat{\gamma}_r,\ \mathcal{R}_u\ \hat{\gamma}_r)$$

where $(\mathcal{R}_l,\ \mathcal{R}_u)$ is the confidence interval for $\mathcal{R}_{sm}$.

Attributable Fraction and Population Attributable Fraction

The attributable fraction measures the excess event rate or risk fraction in the exposed population that is attributable to the exposure. That is, it is the proportion of event rate or risk in the exposed population that would be reduced if the exposure were not present. In contrast, the population attributable fraction measures the excess event rate or risk fraction in the total population that is attributable to the exposure.

In the STDRATE procedure, you can compute the attributable fraction by using either indirect standardization or Mantel-Haenszel estimation.

Indirect Standardization

With indirect standardization, you specify a study population that consists of subjects who are exposed to a factor, such as smoking, and a reference population that consists of subjects who are not exposed to the factor. Denote the numbers of events in the study and reference populations by $\mathcal{D}_s$ and $\mathcal{D}_r$, respectively.

For the rate estimate, denote the population-times in the study and reference populations by $\mathcal{T}_s$ and $\mathcal{T}_r$, respectively. Then the event rates in the two populations can be expressed as the following equations, respectively:

$$\hat{\lambda}_s = \frac{\mathcal{D}_s}{\mathcal{T}_s} \quad \text{and} \quad \hat{\lambda}_r = \frac{\mathcal{D}_r}{\mathcal{T}_r}$$

Similarly, for the risk estimate, denote the numbers of observations in the study and reference populations by $\mathcal{N}_s$ and $\mathcal{N}_r$, respectively. Then the event risks in the two populations can be expressed as the following equations, respectively:

$$\hat{\gamma}_s = \frac{\mathcal{D}_s}{\mathcal{N}_s} \quad \text{and} \quad \hat{\gamma}_r = \frac{\mathcal{D}_r}{\mathcal{N}_r}$$

In the next two subsections, $\beta = \lambda$ denotes the rate statistic and $\beta = \gamma$ denotes the risk statistic.

Attributable Fraction with Indirect Standardization

The attributable fraction is the fraction of event rate or risk in the exposed population that is attributable to exposure:

$$\mathcal{R}_a = \frac{\hat{\beta}_s - \hat{\beta}_r}{\hat{\beta}_s}$$

With a standardized mortality ratio $\mathcal{R}_{sm}$, the attributable fraction is estimated by

$$\mathcal{R}_a = \frac{\mathcal{R}sm - 1}{\mathcal{R}_{sm}}$$

The confidence intervals for the attributable fraction can be computed using the confidence intervals for $\mathcal{R}_{sm}$. That is, with a confidence interval $(\mathcal{R}_l,\ \mathcal{R}_u)$ for $\mathcal{R}_{sm}$, the corresponding $\mathcal{R}_a$ confidence interval is given by

$$\left(\frac{\mathcal{R}_l - 1}{\mathcal{R}_l},\ \frac{\mathcal{R}_u - 1}{\mathcal{R}_u}\right)$$

Population Attributable Fraction with Indirect Standardization

The population attributable fraction for a population is the fraction of event rate or risk in a given time period that is attributable to exposure. The population attributable fraction is

$$\mathcal{R}_{pa} = \frac{\hat{\beta}_0 - \hat{\beta}_r}{\hat{\beta}_0}$$

where

$$\hat{\beta}_0 = \frac{\mathcal{D}_s + \mathcal{D}_r}{\mathcal{T}_s + \mathcal{T}_r}$$

is the combined rate in the total population for the rate statistic and where

$$\hat{\beta}_0 = \frac{\mathcal{D}_s + \mathcal{D}_r}{\mathcal{N}_s + \mathcal{N}_r}$$

is the combined risk in the total population for the risk statistic.

Denote $\rho = \mathcal{D}_s/(\mathcal{D}_s + \mathcal{D}_r)$, the proportion of exposure among events, then $\mathcal{R}_{pa}$ can also be expressed as

$$\mathcal{R}_{pa} = \rho\ \frac{\mathcal{R}_{sm} - 1}{\mathcal{R}_{sm}}$$

where $\mathcal{R}_{sm}$ is the standardized mortality ratio.

An approximate confidence interval for the population attributable rate $\mathcal{R}_{pa}$ can be derived by using the complementary log transformation (Greenland 2008, p. 296). That is, with

$$\mathcal{H} = \log(1 - \mathcal{R}_{pa})$$

a variance estimator for the estimated $\mathcal{H}$ is given by

$$\mathrm{Var}(\hat{\mathcal{H}}) = \frac{\mathcal{R}_{pa}^2}{(1 - \mathcal{R}_{pa})^2}\left(\frac{\hat{V}}{(\mathcal{R}_{sm} - 1)^2} + \frac{2}{\mathcal{D}_s\,(\mathcal{R}_{sm} - 1)} + \frac{\mathcal{D}_r}{\mathcal{D}_s\,(\mathcal{D}_s + \mathcal{D}_r)}\right)$$

where $\hat{V}$ is a variance estimate for $\log(\mathcal{R}_{sm})$.

Mantel-Haenszel Estimation

With Mantel-Haenszel estimation, you specify one study population that consists of subjects who are exposed to a factor and another study population that consists of subjects who are not exposed to the factor. Denote the numbers of events in the exposed and nonexposed study populations by $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively.

For the rate estimate, denote the population-times in the two populations by $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively. Then the event rates in the two populations can be expressed as the following equations, respectively:

$$\hat{\lambda}_1 = \frac{\mathcal{D}_1}{\mathcal{T}_1} \quad \text{and} \quad \hat{\lambda}_2 = \frac{\mathcal{D}_2}{\mathcal{T}_2}$$

Similarly, for the risk estimate, denote the numbers of observations in the two populations by $\mathcal{N}_1$ and $\mathcal{N}_2$, respectively. Then the event risks in the two populations can be expressed as the following equations, respectively:

$$\hat{\gamma}_1 = \frac{\mathcal{D}_1}{\mathcal{N}_1} \quad \text{and} \quad \hat{\gamma}_2 = \frac{\mathcal{D}_2}{\mathcal{N}_2}$$

In the next two subsections, $\beta = \lambda$ denotes the rate statistic and $\beta = \gamma$ denotes the risk statistic.

Attributable Fraction with Mantel-Haenszel Estimation

The attributable fraction is the fraction of event rate or risk in the exposed population that is attributable to exposure:

$$\mathcal{R}_a = \frac{\hat{\beta}_1 - \hat{\beta}_2}{\hat{\beta}_1}$$

Denote the rate or risk ratio by $\mathcal{R} = \hat{\beta}_1/\hat{\beta}_2$. The attributable fraction is given by

$$\mathcal{R}_a = \frac{\mathcal{R} - 1}{\mathcal{R}}$$

The confidence intervals for the attributable fraction can be computed using the confidence intervals for the rate or risk ratio $\mathcal{R}$. That is, with a confidence interval $(\mathcal{R}_l,\ \mathcal{R}_u)$ for $\mathcal{R}$, the corresponding $\mathcal{R}_a$ confidence interval is given by

$$\left(\frac{\mathcal{R}_l - 1}{\mathcal{R}_l},\ \frac{\mathcal{R}_u - 1}{\mathcal{R}_u}\right)$$

For Mantel-Haenszel estimation, you can use the Mantel-Haenszel rate or risk ratio to estimate $\mathcal{R}$.

Population Attributable Fraction with Mantel-Haenszel Estimation

The population attributable fraction for a population is the fraction of event rate or risk in a given time period that is attributable to exposure. The population attributable fraction is

$$\mathcal{R}_{pa} = \frac{\hat{\beta}_0 - \hat{\beta}_2}{\hat{\beta}_0}$$

where

$$\hat{\beta}_0 = \frac{\mathcal{D}_1 + \mathcal{D}_2}{\mathcal{T}_1 + \mathcal{T}_2}$$

is the combined rate in the total population for the rate statistic and where

$$\hat{\beta}_0 = \frac{\mathcal{D}_1 + \mathcal{D}_2}{\mathcal{N}_1 + \mathcal{N}_2}$$

is the combined risk in the total population for the risk statistic.

Denote the proportion of exposure among events as $\rho = \mathcal{D}_1/(\mathcal{D}_1 + \mathcal{D}_2)$. Then $\mathcal{R}_{pa}$ can also be expressed as

$$\mathcal{R}_{pa} = \rho \ \frac{\mathcal{R} - 1}{\mathcal{R}}$$

where $\mathcal{R} = \hat{\beta}_1/\hat{\beta}_2$ is the rate or risk ratio.

An approximate confidence interval for the population attributable rate $\mathcal{R}_{pa}$ can be derived by using the complementary log transformation (Greenland 2008, p. 296). That is, with

$$\mathcal{H} = \log(1 - \mathcal{R}_{pa})$$

a variance estimator for the estimated $\mathcal{H}$ is given by

$$\mathrm{Var}(\hat{\mathcal{H}}) = \frac{\mathcal{R}_{pa}^2}{(1 - \mathcal{R}_{pa})^2} \left(\frac{\hat{V}}{(\mathcal{R} - 1)^2} + \frac{2}{\mathcal{D}_1\,(\mathcal{R} - 1)} + \frac{\mathcal{D}_2}{\mathcal{D}_1\,(\mathcal{D}_1 + \mathcal{D}_2)} \right)$$

where $\hat{V}$ is a variance estimate for $\log(\mathcal{R})$.

For Mantel-Haenszel estimation, you can use the Mantel-Haenszel rate or risk ratio to estimate $\mathcal{R}$.

Applicable Data Sets and Required Variables for Method Specifications

The METHOD= and DATA= options are required in the STDRATE procedure. The METHOD= option specifies the standardization method, and the DATA= and REFDATA= options specify the study populations and reference population, respectively. You can use the GROUP= option in the POPULATION statement to identify various study populations. Table 116.2 lists applicable data sets for each method.

Table 116.2 Applicable Data Sets for Method Specifications

METHOD=	Number of Populations in DATA= Data Set	REFDATA= Data Set
DIRECT	1	X
	2	X
MH	2	
INDIRECT	1	X

Table 116.3 lists the required variables for each method.

Table 116.3 Required Variables for Method Specifications

		DATA= Data Set				REFDATA= Data Set			
METHOD=	STAT=	RATE	RISK	EVENT	TOTAL	RATE	RISK	EVENT	TOTAL
DIRECT	RATE	X		X	X				X
	RISK		X	X	X				X
MH	RATE	X		X	X				
	RISK		X	X	X				
INDIRECT	RATE	X		X	X	X		X	X
	RISK		X	X	X		X	X	X

The symbol “X” indicates that the variable is either explicitly specified or implicitly available from other variables. For example, when STAT=RATE, the variable RATE is available if the corresponding variables EVENT and TOTAL are specified. In other words, if you specify at least two of the three variables, the third variable can be derived.

Applicable Confidence Limits for Rate and Risk Statistics

In the STDRATE procedure, the METHOD= option specifies the standardization method, and the STAT= option specifies either rate or risk for standardization. Table 116.4 lists applicable confidence limits for different methods with standardized rate, rate SMR, standardized risk, and risk SMR.

Table 116.4 Applicable Confidence Limits for Standardized Rate and Risk Statistics

		Confidence Limits			
Statistic	**METHOD=**	**Normal**	**Lognormal**	**Gamma**	**Poisson**
Rate	DIRECT	X	X	X	
	MH	X	X	X	
	INDIRECT	X	X		X
Rate SMR	INDIRECT	X	X		X
Risk	DIRECT	X	X		
	MH	X	X		
	INDIRECT	X	X		X
Risk SMR	INDIRECT	X	X		X

Table 116.5 lists applicable confidence limits for stratum-specific rate, rate SMR, risk, and risk SMR.

Table 116.5 Applicable Confidence Limits for Strata Rate and Risk Statistics

	Confidence Limits		
Statistic	**Normal**	**Lognormal**	**Poisson**
Rate	X	X	X
Rate SMR	X	X	X
Risk	X	X	
Risk SMR	X	X	

Table Output

The STDRATE procedure displays the "Standardization Information" table by default. In addition, the procedure also displays the "Standardized Rate Estimates" table (with the default STAT=RATE option in the PROC STDRATE statement) and the "Standardized Risk Estimates" table (with the STAT=RISK option) by default. The rest of this section describes the output tables in alphabetical order.

Attributable Fraction Estimates

The "Attributable Fraction Estimates" table displays the following information:

- Parameter: attributable rate and population attributable rate for the rate statistic, and attributable risk and population attributable risk for the risk statistic
- Estimate: estimate of the parameter
- Method: method to construct confidence limits
- Lower and Upper: lower and upper confidence limits

Effect Estimates

The "Effect Estimates" table displays the following information:

- Standardized Rate: directly standardized rates for study populations
- Standardized Risk: directly standardized risks for study populations

When EFFECT=RATIO, the table displays the following:

- Estimate: the rate or risk ratio estimate
- Log Ratio: the logarithm of rate ratio or risk ratio estimate
- Standard Error: standard error of the logarithm of the ratio estimate
- Z: the standard Z statistic
- Pr > |Z|: the p-value for the test

When EFFECT=DIFF, the table displays the following:

- Estimate: the rate or risk difference estimate
- Standard Error: standard error of the difference estimate
- Z: the standard Z statistic
- Pr > |Z|: the p-value for the test

Standardization Information

The "Standardization Information" table displays the input data sets, type of statistic to be standardized, standardization method, and number of strata. The table also displays the variance divisor for the risk estimate, and the rate multiplier for the rate estimate. With a rate multiplier c, the rates per c population-time units are displayed in the output tables.

Standardized Morbidity/Mortality Ratio

The "Standardized Morbidity/Mortality Ratio" table displays the following information:

- SMR: standardized morbidity/mortality ratio
- Standard Error: standard error for SMR
- Lower and Upper: lower and upper confidence limits for SMR
- Test Statistic: SMR-1, for the test of SMR=1
- Estimate: value of test statistic
- Standard Error: standard error of the estimate
- Z: the standard Z statistic
- Pr > |Z|: the p-value for the test

Standardized Rate Estimates

The "Standardized Rate Estimates" table displays the following information:

- Population: study populations, and reference population for indirect standardization
- Number of Events: number of events in population
- Population-Time: total contributed time in population, for the rate statistic
- Crude Rate: event rate in the population
- Expected Number of Events
- SMR: standardized morbidity/mortality ratio, for indirect standardization
- Standardized Rate: for the rate statistic
- Standard Error: standard error of the standardized estimate of rate
- Confidence Limits: lower and upper confidence limits for standardized estimate

Standardized Risk Estimates

The "Standardized Risk Estimates" table displays the following information:

- Population: study populations, and reference population for indirect standardization
- Number of Events: number of events in population
- Number of Observations: number of observations in population
- Crude Risk: event risk in the population
- Expected Number of Events
- SMR: standardized morbidity/mortality ratio, for indirect standardization
- Standardized Risk
- Standard Error: standard error of the standardized estimate of risk
- Confidence Limits: lower and upper confidence limits for standardized estimate

Strata Effect Estimates

The "Strata Effect Estimates" table displays the following information for each stratum:

- Stratum Index: a sequential stratum identification number
- STRATA variables: the levels of STRATA variables
- Crude Rate: crude rates for the study populations, for the rate statistic
- Crude Risk: crude risks for the study populations, for the risk statistic

When EFFECT=DIFF, the table displays the following information for each stratum:

- Estimate: difference estimate of the crude rate or crude risk of the study populations
- Standard Error: the standard error of the difference estimate
- Confidence Limits: confidence limits for the difference estimate

When EFFECT=RATIO, the table displays the following information for each stratum:

- Estimate: ratio estimate of the crude rate or crude risk of the study populations
- Confidence Limits: confidence limits for the ratio estimate

Strata Statistics

For each POPULATION statement, the "Strata Information" table displays the following information for each stratum:

- Stratum Index: a sequential stratum identification number
- STRATA variables: the levels of STRATA variables

If the REFERENCE statement is specified, the table displays the following information for each stratum in the reference population:

- Population-Time Value: population-time for the crude rate statistic for direct standardization
- Population-Time Proportion: proportion for the population-time
- Number of Observations Value: number of observations for the crude risk statistic for direct standardization
- Number of Observations Proportion: proportion for the number of observations
- Crude Rate: crude rate statistic for indirect standardization
- Crude Risk: crude risk statistic for indirect standardization

For the rate statistic, the table displays the following information for each stratum in the specified study data set:

- Number of Events
- Population-Time Value
- Population-Time Proportion
- Crude Rate Estimate
- Standard Error: standard error for the crude rate estimate if the CL=NORMAL suboption is specified in the STATS option in the STRATA statement
- Confidence Limits: confidence limits for the crude rate estimate if the CL suboption is specified in the STATS option in the STRATA statement
- Expected Number of Events: expected number of events that use the reference population population-time for direct standardization, Mantel-Haenszel weight for Mantel-Haenszel estimation, or reference population rate for indirect standardization

For the risk statistic, the table displays the following information for each stratum in the specified study data set:

- Number of Events
- Number of Observations Value

- Number of Observations Proportion
- Crude Risk Estimate
- Standard Error: standard error for the crude risk estimate, if the CL=NORMAL suboption is specified in the STATS option in the STRATA statement
- Confidence Limits: confidence limits for the crude risk estimate, if the CL suboption is specified in the STATS option in the STRATA statement
- Expected Number of Events: expected number of events that uses the reference population number of observations for direct standardization, Mantel-Haenszel weight for Mantel-Haenszel estimation, or reference population risk for indirect standardization

Strata SMR Estimates

The "Strata SMR Estimates" table displays the following information for each stratum:

- Stratum Index: a sequential stratum identification number
- STRATA variables: the levels of STRATA variables
- Number of Events
- Expected Number of Observations
- SMR Estimate
- Standard Error: standard error for the SMR estimate, if the CL=NORMAL suboption is specified in the SMR option in the STRATA statement
- Confidence Limits: confidence limits for the SMR estimate, if the CL suboption is specified in the SMR option in the STRATA statement

For the rate statistic, the table also displays the following information for each stratum:

- Population-Time
- Reference Rate

For the risk statistic, the table also displays the following information for each stratum:

- Number of Observations
- Reference Risk

ODS Table Names

PROC STDRATE assigns a name to each table it creates. You must use these names to refer to tables when you use the Output Delivery System (ODS). These names are listed in Table 116.6. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 116.6 ODS Tables Produced by PROC STDRATE

ODS Table Name	Description	Statement	Option
AttrFraction	Attributable fraction	PROC STDRATE	METHOD=INDIRECT(AF)
		PROC STDRATE	METHOD=MH(AF)
Effect	Effect estimates	PROC STDRATE	EFFECT
SMR	Standardized morbidity/mortality ratio	PROC STDRATE	METHOD=INDIRECT
StdInfo	Standardization information	PROC STDRATE	
StdRate	Standardized rate estimates	PROC STDRATE	STAT=RATE
StdRisk	Standardized risk estimates	PROC STDRATE	STAT=RISK
StrataEffect	Strata effect estimates	STRATA	EFFECT
StrataStats	Strata statistics	STRATA	STATS
StrataSMR	Strata SMR estimates	STRATA	SMR

Graphics Output

This section describes the use of ODS for creating graphics with the STDRATE procedure. To request these graphs, ODS Graphics must be enabled and you must specify the associated graphics options in the PROC STDRATE statement. For more information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Strata Distribution Plot

The PLOTS=DIST option displays the proportion of exposed time or sample size for each stratum in the populations.

Strata Effect Plot

The PLOTS=EFFECT option displays the stratum-specific effect measure of rate difference, rate ratio, risk difference, or risk ratio. In addition, the crude effect measure and confidence limits of these stratum-specific effect estimates are also displayed.

Strata Rate Plot

The PLOTS=RATE option displays the stratum-specific rate estimates and their confidence limits of populations. In addition, the overall crude rates of populations are also displayed.

Strata Risk Plot

The PLOTS=RISK option displays the stratum-specific risk estimates and their confidence limits of populations. In addition, the overall crude risks of populations are also displayed.

Strata SMR Plot

The PLOTS=SMR option displays the SMR for each stratum in the populations.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

PROC STDRATE assigns a name to each graph it creates. You can use these names to refer to the graphs when you use ODS. To request the graph, ODS Graphics must be enabled and you must use the PLOTS option in the PROC STDRATE statement to specify the *plot-request* indicated in Table 116.7.

Table 116.7 Graphs Produced by PROC STDRATE

ODS Graph Name	Plot Description	*plot-request* in PLOTS Option
StrataDistPlot	Strata proportion of exposed time or sample size	DIST
StrataEffectPlot	Strata effect measure of rate difference, rate ratio, risk difference, or risk ratio	EFFECT
StrataRatePlot	Strata rate estimates	RATE
StrataRiskPlot	Strata risk estimates	RISK
StrataSMRPlot	Strata SMR of rates or risks	SMR

Examples: STDRATE Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 116.1: Comparing Directly Standardized Rates

(View the complete code for this example (sratex1.sas) in the example repository.)

This example computes directly standardized mortality rates for populations in the states of Alaska and Florida, and then compares these two standardized rates with a rate ratio statistic.

The Alaska data set contains the stratum-specific mortality information in a given period of time for the state of Alaska (Alaska Bureau of Vital Statistics 2000a, b). Variables Sex and Age are the grouping variables that form the strata in the standardization, and variables Death and PYear indicate the number of events and person-years, respectively. The COMMA7. format is specified in the DATA step to input numerical values that contain commas in PYear.

```
data Alaska;
   State='Alaska';
   input Sex $ Age $ Death PYear:comma7.;
   datalines;
Male    00-14   37   81,205
Male    15-34   68   93,662
Male    35-54  206  108,615
Male    55-74  369   35,139
Male    75+    556    5,491
Female  00-14   78   77,203
Female  15-34  181   85,412
Female  35-54  395  100,386
Female  55-74  555   32,118
Female  75+    479    7,701
;
```

The Florida data set contains the corresponding stratum-specific mortality information for the state of Florida (Florida Department of Health 2000, 2020). Variables Sex and Age are the grouping variables that form the strata in the standardization, and variables Death and PYear indicate the number of events and person-years, respectively.

```
data Florida;
   State='Florida';
   input Sex $ Age $ Death:comma6. PYear:comma9.;
   datalines;
Male    00-14   1,189  1,505,889
Male    15-34   2,962  1,972,157
Male    35-54  10,279  2,197,912
Male    55-74  26,354  1,383,533
Male    75+    42,443    554,632
Female  00-14     906  1,445,831
Female  15-34   1,234  1,870,430
```

```
Female  35-54   5,630  2,246,737
Female  55-74  18,309  1,612,270
Female  75+    53,489    868,838
;
```

The TwoStates data set contains the data sets Alaska and Florida:

```
data TwoStates;
   length State $ 7.;
   set Alaska Florida;
run;
```

The US data set contains the corresponding stratum-specific person-years information for the United States (US Bureau of the Census 2011). Variables Sex and Age are the grouping variables that form the strata in the standardization, and the variable PYear indicates the person-years.

```
data US;
   input Sex $ Age $ PYear:comma10.;
   datalines;
Male    00-14  30,854,207
Male    15-34  40,199,647
Male    35-54  40,945,028
Male    55-74  19,948,630
Male    75+     6,106,351
Female  00-14  29,399,168
Female  15-34  38,876,268
Female  35-54  41,881,451
Female  55-74  22,717,040
Female  75+    10,494,416
;
```

The following statements invoke the STDRATE procedure and compute the direct standardized rates for the states of Florida and Alaska by using the United States as the reference population. The DATA= option names the data set for the study populations, and the REFDATA= option names the data set for the reference population.

```
ods graphics on;
proc stdrate data=TwoStates
             refdata=US
             method=direct
             stat=rate(mult=1000)
             effect
             plots(only)=(dist effect)
             ;
   population group=State event=Death total=PYear;
   reference  total=PYear;
   strata Sex Age / effect;
run;
```

The METHOD=DIRECT option requests direct standardization, and the STAT=RATE option specifies the rate statistic for standardization. With the EFFECT option, the procedure computes the rate effect between the study populations with the default rate ratio statistics.

The “Standardization Information” table in Output 116.1.1 displays the standardization information.

Output 116.1.1 Standardization Information

The STDRATE Procedure

Standardization Information	
Data Set	WORK.TWOSTATES
Group Variable	State
Reference Data Set	WORK.US
Method	Direct Standardization
Statistic	Rate
Number of Strata	10
Rate Multiplier	1000

With ODS Graphics enabled, the PLOTS(ONLY)=(DIST EFFECT) option displays the strata distribution plot and the strata effect plot, but does not display the default strata rate plot.

The strata distribution plot displays proportions for stratum-specific person-years in the study populations and reference population, as shown in Output 116.1.2.

Output 116.1.2 Strata Distribution Plot

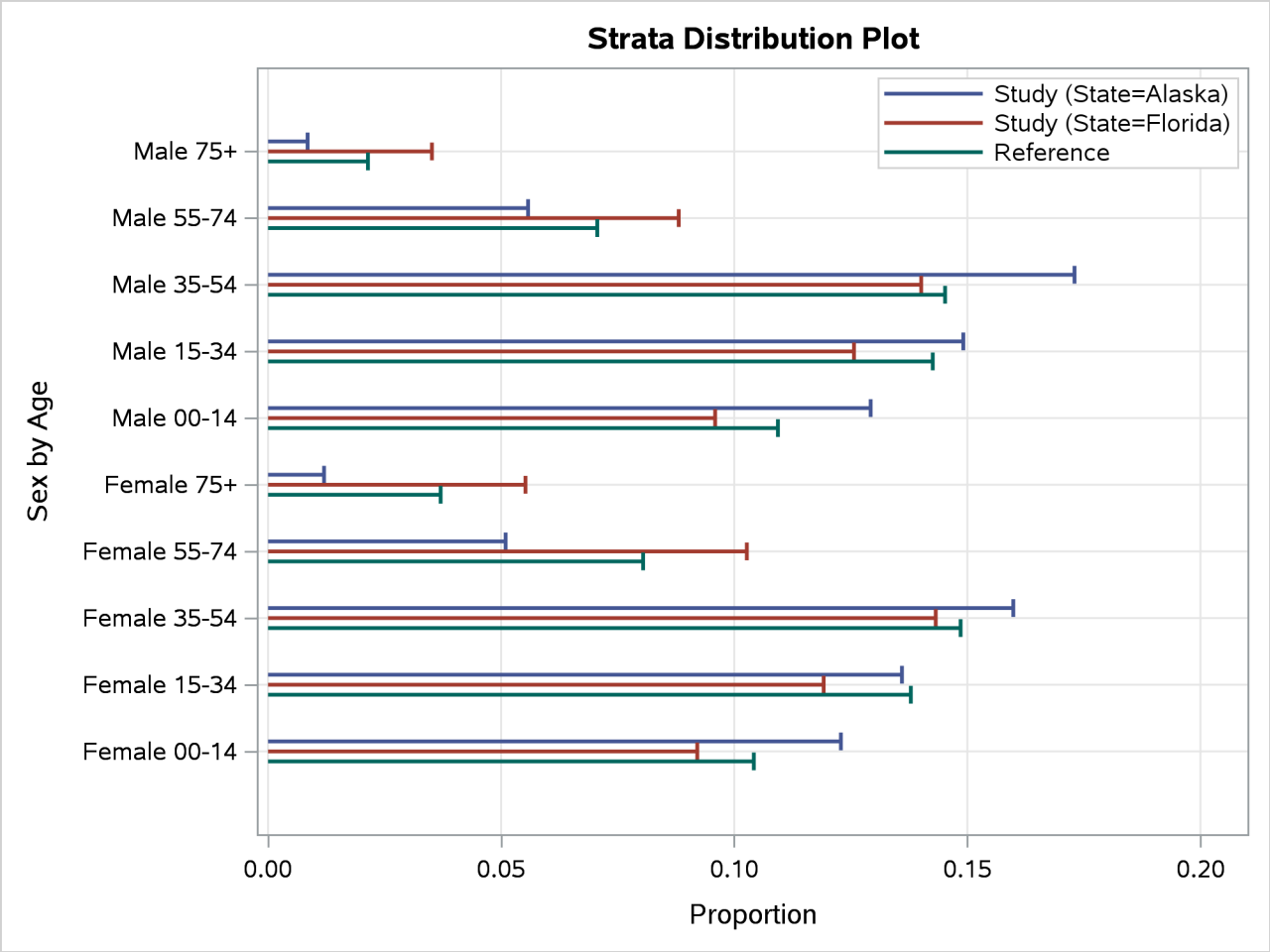

The EFFECT option in the STRATA statement and the STAT=RATE option request that the “Strata Rate

Effect Estimates" table in Output 116.1.3 display the stratum-specific rate effect statistics between the two study populations. The default EFFECT=RATIO in the PROC STDRATE statement requests that the stratum-specific rate ratio statistics be displayed.

Output 116.1.3 Strata Effect Estimates

Strata Rate Effect Estimates (Rate Multiplier = 1000)							
			State				
Stratum Index	Sex	Age	Alaska	Florida	Rate Ratio	95% Lognormal Confidence Limits	
1	Female	00-14	1.010	0.6266	1.61231	1.27940	2.03185
2	Female	15-34	2.119	0.6597	3.21208	2.74812	3.75437
3	Female	35-54	3.935	2.5059	1.57025	1.41795	1.73889
4	Female	55-74	17.280	11.3560	1.52166	1.39844	1.65574
5	Female	75+	62.200	61.5638	1.01033	0.92341	1.10542
6	Male	00-14	0.456	0.7896	0.57707	0.41604	0.80044
7	Male	15-34	0.726	1.5019	0.48339	0.38010	0.61476
8	Male	35-54	1.897	4.6767	0.40554	0.35330	0.46552
9	Male	55-74	10.501	19.0483	0.55129	0.49746	0.61094
10	Male	75+	101.257	76.5246	1.32319	1.21699	1.43866

The "Strata Rate Effect Estimates" table shows that except for the age group 75+, Alaska has lower mortality rates for male groups and higher mortality rates for female groups than Florida. For age group 75+, Alaska has higher mortality rates than Florida for both male and female groups.

With ODS Graphics enabled and two study populations, the PLOTS=EFFECT option displays the stratum-specific effect measures and their associated confidence limits, as shown in Output 116.1.4. The STAT=RATE option and the default EFFECT=RATIO option request that the strata rate ratios be displayed. By default, confidence limits are generated with 95% confidence level. This plot displays the stratum-specific rate ratios in the "Strata Rate Effect Estimates" table in Output 116.1.3.

Output 116.1.4 Strata Effect Measure Plot

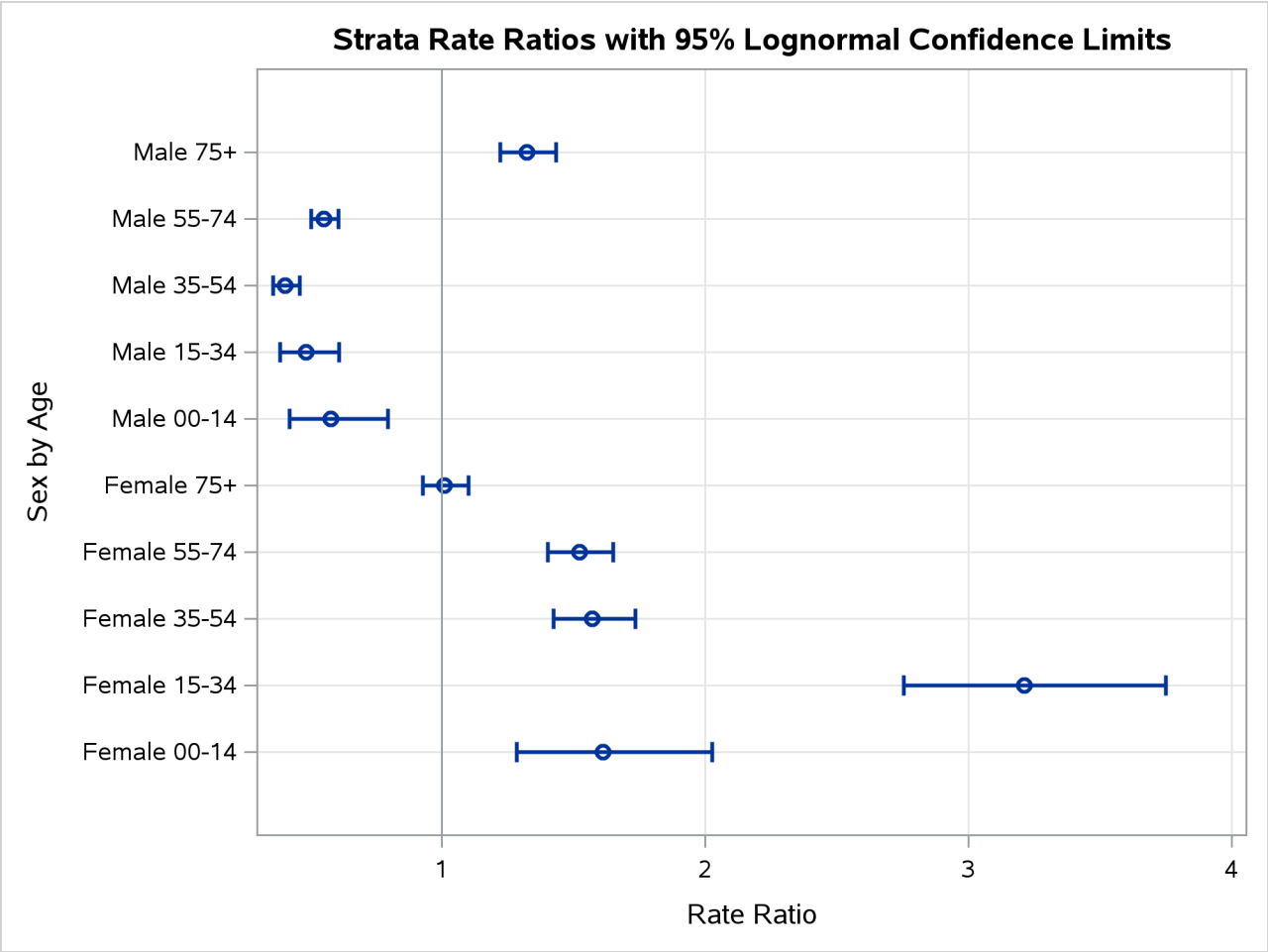

The "Directly Standardized Rate Estimates" table in Output 116.1.5 displays directly standardized rates and related statistics.

Output 116.1.5 Directly Standardized Rate Estimates

Directly Standardized Rate Estimates
Rate Multiplier = 1000

	Study Population			Reference Population		Standardized Rate			
State	Observed Events	Population-Time	Crude Rate	Expected Events	Population-Time	Estimate	Standard Error	95% Normal Confidence Limits	
Alaska	2924	626932	4.6640	2270876	281422206	8.0693	0.1643	7.7472	8.3913
Florida	162795	15658229	10.3968	2176572	281422206	7.7342	0.0195	7.6959	7.7725

The MULT=1000 suboption in the STAT=RATE option requests that rates per 1,000 person-years be displayed. The table shows that the although the crude rate in the Florida population (10.3968) is higher than the crude rate in the Alaska population (4.664), the resulting standardized rate in the Florida population (7.7342) is lower than the crude rate in the Alaska population (8.0693).

The EFFECT option requests that the "Rate Effect Estimates" table in Output 116.1.6 display the log rate ratio statistics of the two directly standardized rates by default.

Output 116.1.6 Effect Estimates

Rate Effect Estimates (Rate Multiplier = 1000)								
State								
Alaska	Florida	Rate Ratio	95% Lognormal Confidence Limits		Log Rate Ratio	Standard Error	Z	Pr > |Z|
8.0693	7.7342	1.0433	1.00220	1.08614	0.0424	0.0205	2.07	0.0387

The table shows that with a log rate ratio statistic 1.0433, the resulting p-value is 0.0387, which indicates that the death rate is significantly higher in Alaska than in Florida at the 5% significance level.

Example 116.2: Computing Mantel-Haenszel Risk Estimation

(View the complete code for this example (sratex2.sas) in the example repository.)

This example uses Mantel-Haenszel method to estimate the effect of household smoking on respiratory symptoms of school children, after adjusting for the effects of the student's grade and household pets.

Suppose that the School data set contains the stratum-specific numbers of cases of respiratory symptoms in a given school year for a school district. Variables Pet and Grade are the grouping variables that form the strata in the standardization, and the variable Smoking identifies students who have smokers in their households. The variables Case and Student indicate the number of cases with respiratory symptoms and the total number of students, respectively.

```
data School;
   input Smoking $ Pet $ Grade $ Case Student;
   datalines;
Yes   Yes   K-1   109    807
Yes   Yes   2-3   106    791
Yes   Yes   4-5   112    868
Yes   No    K-1   168   1329
Yes   No    2-3   162   1337
Yes   No    4-5   183   1594
No    Yes   K-1   284   2403
No    Yes   2-3   266   2237
No    Yes   4-5   273   2279
No    No    K-1   414   3398
No    No    2-3   372   3251
No    No    4-5   382   3270
;
```

The following statements invoke the STDRATE procedure and compute the Mantel-Haenszel rate difference statistic between students with household smokers and students without household smokers:

```
ods graphics on;
proc stdrate data=School
             method=mh
             stat=risk
             effect=diff
             plots=all
             ;
   population group=Smoking event=Case total=Student;
   strata Pet Grade / order=data stats(cl=none) effect;
run;
```

The ORDER=DATA option in the STRATA statement sorts the strata by the order of appearance in the input data set.

The "Standardization Information" table in Output 116.2.1 displays the standardization information.

Output 116.2.1 Standardization Information

The STDRATE Procedure

Standardization Information	
Data Set	WORK.SCHOOL
Group Variable	Smoking
Method	Mantel-Haenszel
Statistic	Risk
Number of Strata	6

The STATS option in the STRATA statement requests that the STDRATE procedure display a "Mantel-Haenszel Standardized Strata Statistics" table for study populations, as shown in Output 116.2.2. The table displays the strata information and the expected number of events in each stratum. The Expected Events column shows the expected number of events when the Mantel-Haenszel weights are applied to the corresponding stratum-specific risks in the study populations. The CL=NONE suboption requests that confidence limits for strata risks not be displayed.

Output 116.2.2 Mantel-Haenszel Standardized Strata Statistics

The STDRATE Procedure

Mantel-Haenszel Standardized Strata Statistics									
				Study Population					
					Number of Observations			Mantel-Haenszel	
Smoking	Stratum Index	Pet	Grade	Observed Events	Value	Proportion	Crude Risk	Weight	Expected Events
No	**1**	Yes	K-1	284	2403	0.1427	0.118186	604.12	71.398
No	**2**	Yes	2-3	266	2237	0.1329	0.118909	584.37	69.487
No	**3**	Yes	4-5	273	2279	0.1353	0.119789	628.59	75.298
No	**4**	No	K-1	414	3398	0.2018	0.121836	955.35	116.396
No	**5**	No	2-3	372	3251	0.1931	0.114426	947.38	108.405
No	**6**	No	4-5	382	3270	0.1942	0.116820	1071.62	125.187
Yes	**1**	Yes	K-1	109	807	0.1200	0.135068	604.12	81.597
Yes	**2**	Yes	2-3	106	791	0.1176	0.134008	584.37	78.310
Yes	**3**	Yes	4-5	112	868	0.1291	0.129032	628.59	81.108
Yes	**4**	No	K-1	168	1329	0.1976	0.126411	955.35	120.767
Yes	**5**	No	2-3	162	1337	0.1988	0.121167	947.38	114.791
Yes	**6**	No	4-5	183	1594	0.2370	0.114806	1071.62	123.028

With ODS Graphics enabled, the PLOTS=ALL option displays all appropriate plots. With the METHOD=MH and STAT=RISK options, these plots include the strata distribution plot, strata risk plot, and strata effect plot.

The strata distribution plot displays proportions for stratum-specific numbers of students in the study populations, as shown in Output 116.2.3.

Output 116.2.3 Strata Distribution Plot

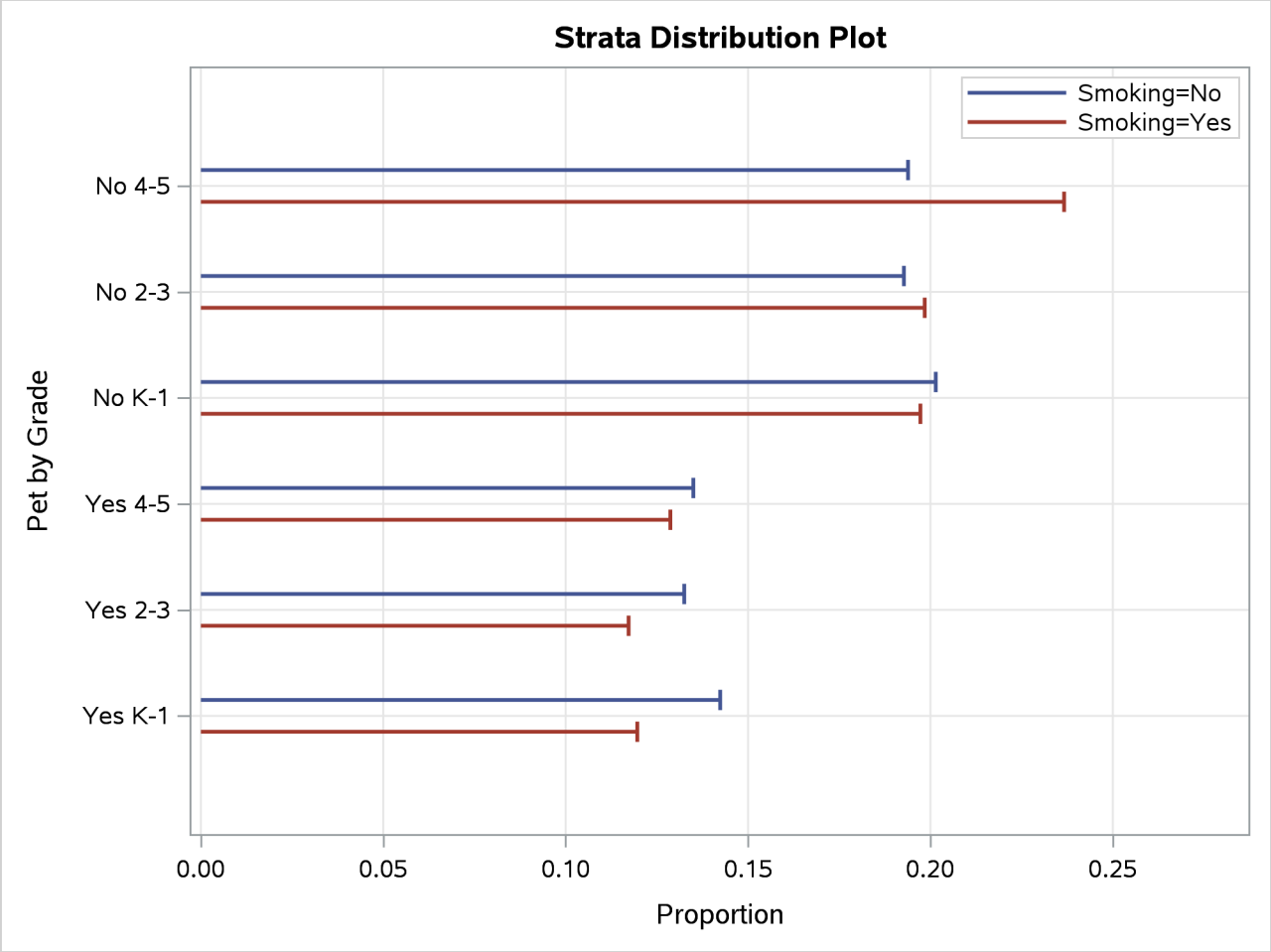

The strata risk plot displays stratum-specific risk estimates with confidence limits in the study populations, as shown in Output 116.2.4. This plot displays stratum-specific risk estimates in the "Mantel-Haenszel Standardized Strata Statistics" table in Output 116.2.2. In addition, the overall crude risks for the two study populations are also displayed. By default, strata levels are displayed on the vertical axis.

Output 116.2.4 Strata Risk Plot

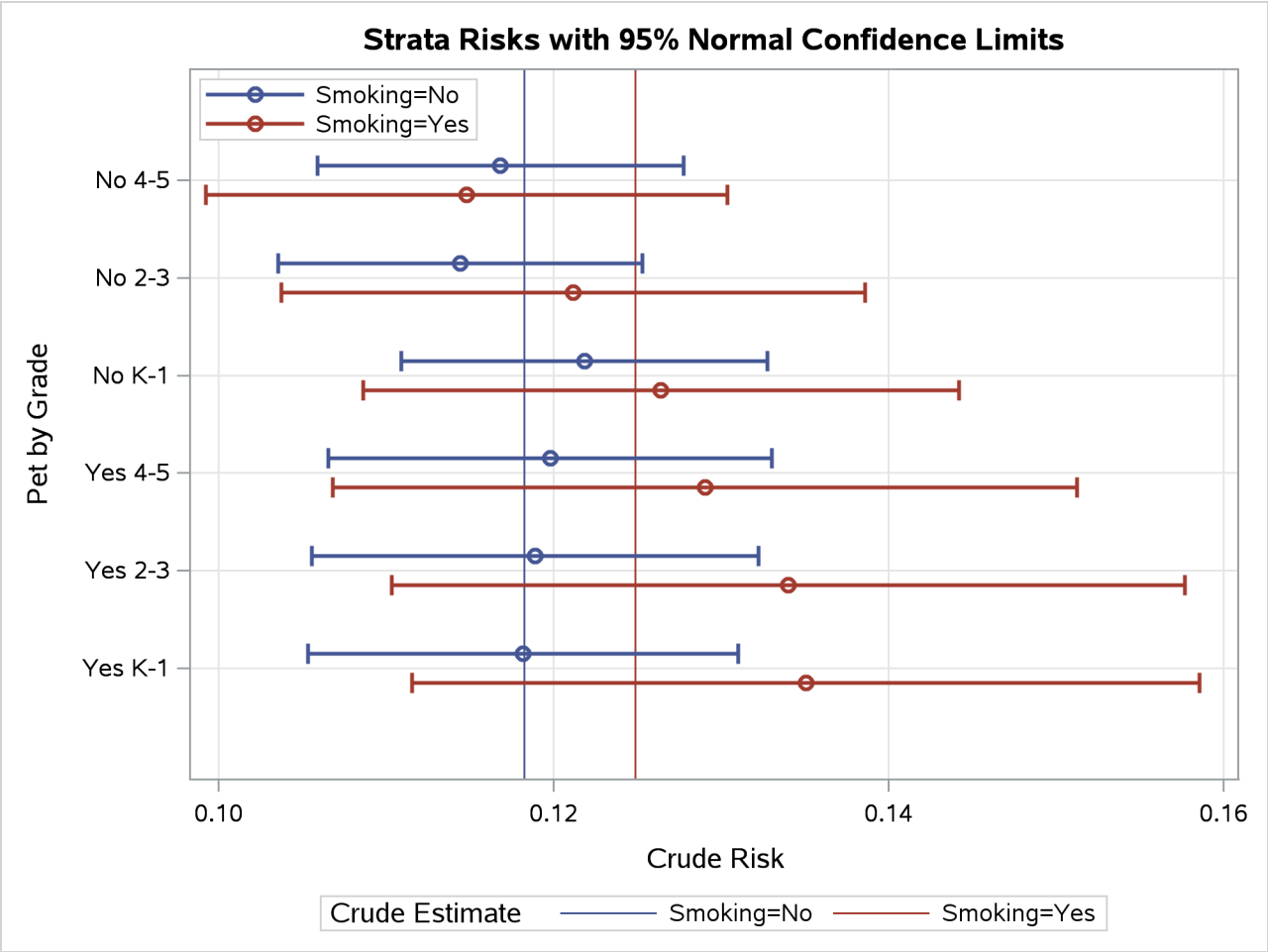

The EFFECT option in the STRATA statement requests that the "Strata Risk Effect Estimates" table be displayed, as shown in Output 116.2.5. The EFFECT=DIFF option in the PROC STDRATE statement requests that strata risk differences be displayed.

Output 116.2.5 Strata Effect Estimates

Strata Risk Effect Estimates

Stratum Index	Pet	Grade	Smoking No	Smoking Yes	Risk Difference	Standard Error	95% Normal Confidence Limits	
1	Yes	K-1	0.11819	0.13507	-.016883	0.013716	-.043766	0.010001
2	Yes	2-3	0.11891	0.13401	-.015098	0.013912	-.042366	0.012169
3	Yes	4-5	0.11979	0.12903	-.009243	0.013257	-.035225	0.016740
4	No	K-1	0.12184	0.12641	-.004574	0.010704	-.025554	0.016405
5	No	2-3	0.11443	0.12117	-.006740	0.010527	-.027373	0.013892
6	No	4-5	0.11682	0.11481	0.002014	0.009762	-.017120	0.021148

The "Strata Risk Effect Estimates" table shows that for the stratum of students without household pets in

Grade 4–5, the risk is higher for students without household smokers than for students with household smokers. For all other strata, the risk is lower for students without household smokers than for students with household smokers. The difference is not significant in each stratum because the null value 0 is between the lower and upper confidence limits.

With ODS Graphics enabled, the PLOTS=EFFECT option displays the plot with the stratum-specific risk effect measures and their associated confidence limits, as shown in Output 116.2.6. The EFFECT=DIFF option requests that the risk difference be displayed. By default, confidence limits are generated with 95% confidence level. This plot displays the stratum-specific risk differences in the "Strata Risk Effect Estimates" table in Output 116.2.5.

Output 116.2.6 Strata Risk Plot

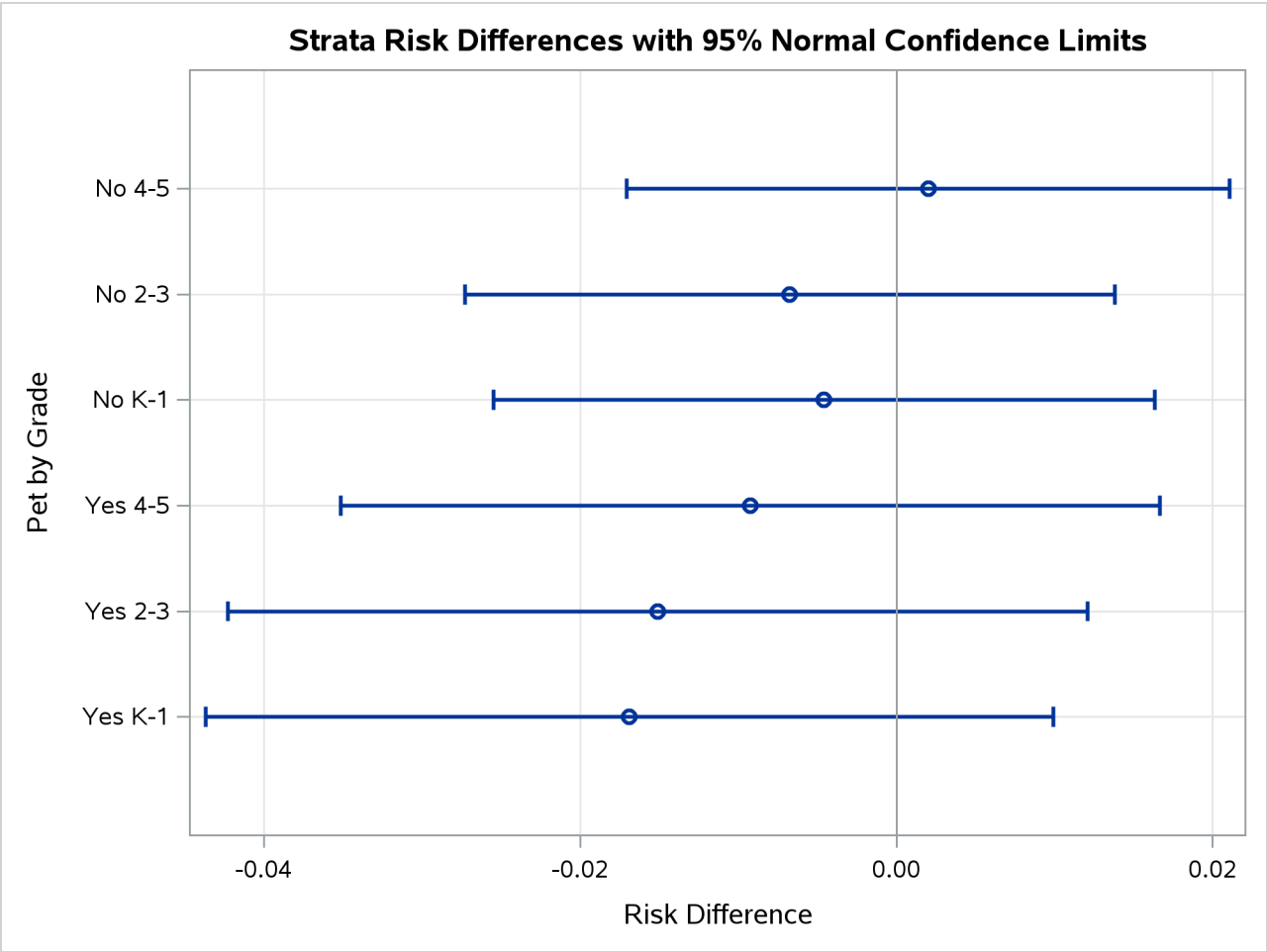

The "Mantel-Haenszel Standardized Risk Estimates" table in Output 116.2.7 displays the Mantel-Haenszel standardized risks and related statistics.

Output 116.2.7 Standardized Risk Estimates (Mantel-Haenszel Estimation)

Mantel-Haenszel Standardized Risk Estimates									
	Study Population			Mantel-Haenszel		Standardized Risk			
Smoking	Observed Events	Number of Observations	Crude Risk	Expected Events	Weight	Estimate	Standard Error	95% Normal Confidence Limits	
No	1991	16838	0.1182	566.172	4791.43	0.1182	0.00250	0.1133	0.1231
Yes	840	6726	0.1249	599.602	4791.43	0.1251	0.00404	0.1172	0.1331

The EFFECT=DIFF option requests that the "Risk Effect Estimates" table display the risk difference statistic for the two directly standardized risks, as shown in Output 116.2.8.

Output 116.2.8 Mantel-Haenszel Effect Estimates

Risk Effect Estimates							
Smoking							
No	Yes	Risk Difference	95% Normal Confidence Limits		Standard Error	Z	Pr > |Z|
0.1182	0.1251	-0.00698	-.016284	0.002330	0.00475	-1.47	0.1418

The table shows that although the standardized risk for students without household smokes is lower than the standardized risk for students with household smokes, the difference (–0.00698) is not significant at the 5% significance level, (p-value 0.1418).

Example 116.3: Computing Attributable Fraction Estimates

(View the complete code for this example (sratex3.sas) in the example repository.)

This example computes the excess event risk fraction that is attributable to a specific chemical exposure for workers in a factory.

Suppose that the Factory data set contains the stratum-specific event information for exposure to a specific chemical agent. The variable Age is the grouping variable that forms the strata. The variables Event_E and Count_E indicate the number of events and number of workers for workers with the specific chemical exposure, respectively. The variables Event_NE and Count_NE indicate the number of events and number of workers for workers without the specific chemical exposure, respectively.

```
data Factory;
   input Age $ Event_E Count_E Event_NE Count_NE;
   datalines;
20-29   31  352  143  2626
30-39   57  486  392  4124
40-49   62  538  459  4662
50-59   50  455  337  3622
60-69   38  322  199  2155
70+      9   68   35   414
;
```

The following statements invoke the STDRATE procedure and compute the attributable risk and population attributable risk for the chemical exposure:

```
ods graphics on;
proc stdrate data=Factory
             refdata=Factory
             method=indirect(af)
             stat=risk
             plots(stratum=horizontal)
             ;
   population event=Event_E  total=Count_E;
   reference  event=Event_NE total=Count_NE;
   strata Age / stats;
run;
```

The "Standardization Information" table in Output 116.3.1 displays the standardization information.

Output 116.3.1 Standardization Information

The STDRATE Procedure

Standardization Information	
Data Set	WORK.FACTORY
Reference Data Set	WORK.FACTORY
Method	Indirect Standardization
Statistic	Risk
Number of Strata	6

The STATS option in the STRATA statement requests that the "Indirectly Standardized Strata Statistics" table in Output 116.3.2 display the strata information and the expected number of events at each stratum. The Expected Events column shows the expected numbers of events when the stratum-specific risks in the reference data set are applied to the corresponding numbers of workers in the study data set.

Output 116.3.2 Strata Information (Indirect Standardization)

The STDRATE Procedure

Indirectly Standardized Strata Statistics											
			Study Population						Reference Population		
			Number of Observations						Number of Observations		
Stratum Index	Age	Observed Events	Value	Proportion	Crude Risk	Standard Error	95% Normal Confidence Limits		Value	Proportion	Crude Risk
1	20-29	31	352	0.1585	0.088068	0.015105	0.058463	0.117673	2626	0.1492	0.05446
2	30-39	57	486	0.2188	0.117284	0.014595	0.088678	0.145890	4124	0.2343	0.09505
3	40-49	62	538	0.2422	0.115242	0.013767	0.088260	0.142224	4662	0.2648	0.09846
4	50-59	50	455	0.2049	0.109890	0.014662	0.081153	0.138627	3622	0.2058	0.09304
5	60-69	38	322	0.1450	0.118012	0.017979	0.082774	0.153251	2155	0.1224	0.09234
6	70+	9	68	0.0306	0.132353	0.041095	0.051809	0.212897	414	0.0235	0.08454

Indirectly Standardized Strata Statistics	
Stratum Index	Expected Events
1	19.1683
2	46.1959
3	52.9691
4	42.3343
5	29.7346
6	5.7488

With ODS Graphics enabled and the specified STAT=RISK option, the default PLOTS=RISK option displays the stratum-specific risk estimates in the study and reference populations, as shown in Output 116.3.3. The STRATUM=HORIZONTAL global option in the PLOTS option displays the strata information on the horizontal axis. The plot displays the stratum-specific risk estimates in the "Indirect Standardized Strata Statistics" table in Output 116.3.2. In addition, confidence limits for the risk estimates in the study population and the overall crude risks for the two populations are also displayed

Output 116.3.3 Strata Risk Plot

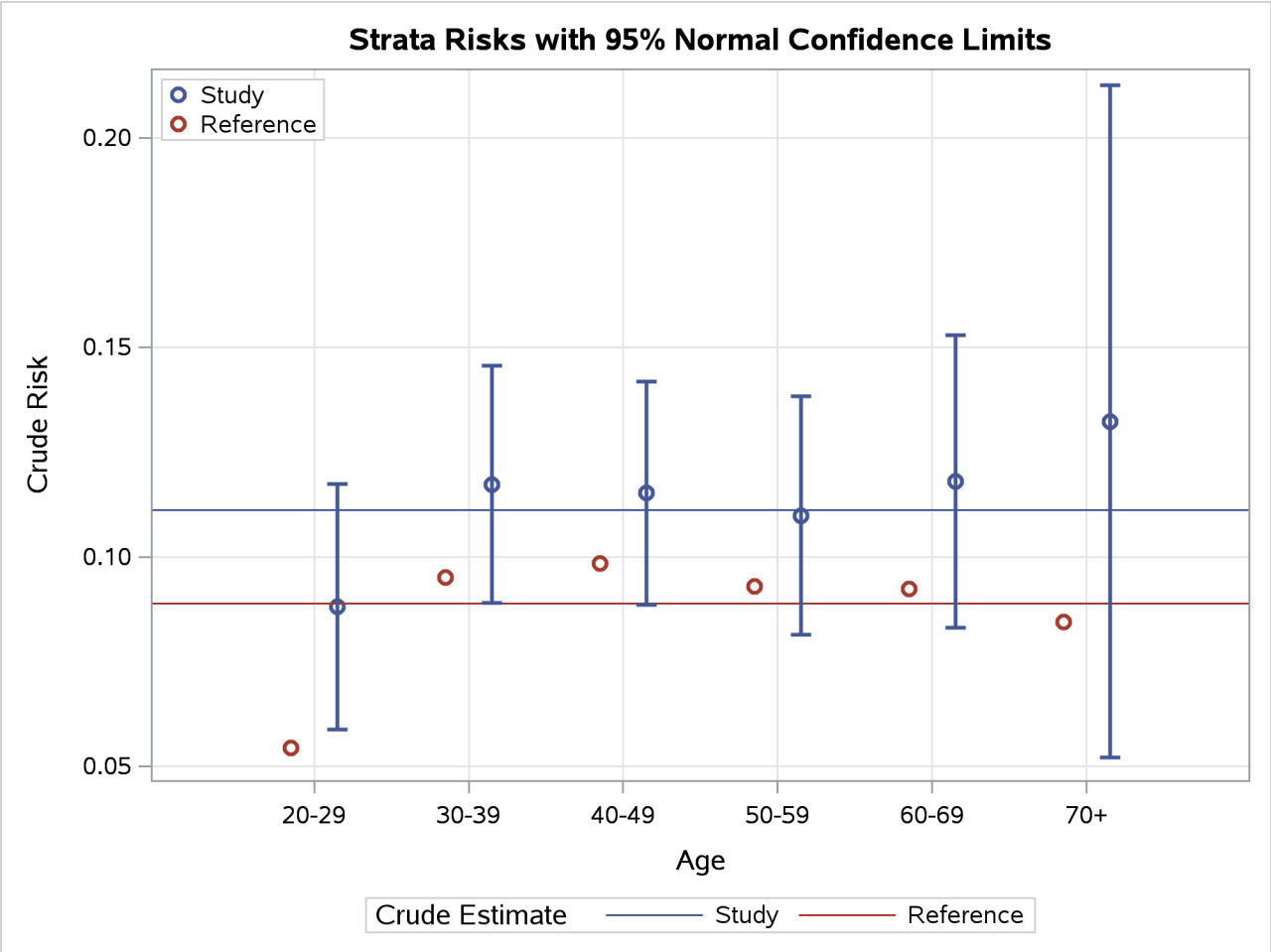

The METHOD=INDIRECT option requests that the "Standardized Morbidity/Mortality Ratio" table in Output 116.3.4 display the SMR, its 95% confidence limits, and the test for the null hypothesis H_0 : SMR $=$ 1.

Output 116.3.4 Standardized Morbidity/Mortality Ratio

Standardized Morbidity/Mortality Ratio							
Observed Events	Expected Events	SMR	Standard Error	95% Normal Confidence Limits		Z	Pr > |Z|
247	196.151	1.2592	0.0755	1.1113	1.4072	3.43	0.0006

The "Standardized Morbidity/Mortality Ratio" table shows that SMR=1.259, the 95% confidence limits do not contain the null value SMR=1, and the null hypothesis of SMR=1 is rejected at $\alpha = 0.05$ level from the normal test.

The "Indirectly Standardized Risk Estimates" table in Output 116.3.5 displays the standardized risks and related statistics.

Output 116.3.5 Standardized Risks (Indirect Standardization)

Indirectly Standardized Risk Estimates									
Study Population						Standardized Risk			
Observed Events	Number of Observations	Crude Risk	Reference Crude Risk	Expected Events	SMR	Estimate	Standard Error	95% Normal Confidence Limits	
247	2221	0.1112	0.0889	196.151	1.2592	0.1120	0.00671	0.0988	0.1251

The AF suboption in the METHOD=INDIRECT option requests that the "Attributable Fraction Estimates" table display the attributable risk and population attributable risk, as shown in Output 116.3.6

Output 116.3.6 Attributable Fraction Estimates

Attributable Fraction Estimates			
Parameter	Estimate	95% Confidence Limits	
Attributable Risk	0.20587	0.10013	0.28937
Population Attributable Risk	0.02806	0.01159	0.04426

The attributable risk fraction 0.206 indicates that 20.6% of all events in the chemical exposure group are attributed to the chemical exposure, and the population attributable risk fraction 0.028 indicates that about 2.8% of all events in the total population are attributed to the chemical exposure.

The Attributable fraction can also be computed by using Mantel-Haenszel method.

Suppose that the Factory1 data set contains the stratum-specific event information for exposure to a specific chemical agent. The variable Age is the grouping variable that forms the strata, and the variable Exposure identifies workers with chemical exposure. The variables Event and Count indicate the number of events and number of workers, respectively.

```
data Factory1;
   input Exposure $ Age $ Event Count;
   datalines;
Yes  20-29   31   352
Yes  30-39   57   486
Yes  40-49   62   538
Yes  50-59   50   455
Yes  60-69   38   322
Yes  70+      9    68
No   20-29  143  2626
No   30-39  392  4124
No   40-49  459  4662
No   50-59  337  3622
No   60-69  199  2155
No   70+     35   414
;
```

The following statements invoke the STDRATE procedure and compute the attributable risk and population attributable risk for the chemical exposure:

```
proc stdrate data=Factory1
             method=mh(af)
             stat=risk
             effect
             ;
   population group(order=data exposed='Yes')=Exposure
              event=Event total=Count;
   strata Age;
run;
```

The GROUP=EXPOSURE option specifies the variable Exposure, whose values identify the various populations. The ORDER= suboption specifies the order in which the values of Exposure are to be displayed, and the EXPOSED= option identifies the exposed group in the derivation of the attributable fraction.

The "Standardization Information" table in Output 116.3.7 displays the standardization information.

Output 116.3.7 Standardization Information

The STDRATE Procedure

Standardization Information	
Data Set	WORK.FACTORY1
Group Variable	Exposure
Method	Mantel-Haenszel
Statistic	Risk
Number of Strata	6

The "Mantel-Haenszel Standardized Risk Estimates" table in Output 116.3.8 displays the Mantel-Haenszel standardized risks and related statistics.

Output 116.3.8 Standardized Risk Estimates (Mantel-Haenszel Estimation)

Mantel-Haenszel Standardized Risk Estimates									
	Study Population			Mantel-Haenszel		Standardized Risk			
Exposure	Observed Events	Number of Observations	Crude Risk	Expected Events	Weight	Estimate	Standard Error	95% Normal Confidence Limits	
Yes	247	2221	0.1112	219.122	1970.26	0.1112	0.00667	0.0981	0.1243
No	1565	17603	0.0889	174.134	1970.26	0.0884	0.00214	0.0842	0.0926

The EFFECT option requests that the "Risk Effect Estimates" table display the risk ratio statistic for the two directly standardized risks, as shown in Output 116.3.9.

Output 116.3.9 Mantel-Haenszel Effect Estimates

Risk Effect Estimates								
Exposure								
Yes	No	Risk Ratio	95% Lognormal Confidence Limits		Log Risk Ratio	Standard Error	Z	Pr > |Z|
0.1112	0.0884	1.2584	1.10851	1.42845	0.2298	0.0647	3.55	0.0004

The AF suboption in the METHOD=MH option requests that the "Attributable Fraction Estimates" table display the attributable risk and population attributable risk, as shown in Output 116.3.10

Output 116.3.10 Attributable Fraction Estimates

Attributable Fraction Estimates Exposed = Yes			
Parameter	Estimate	95% Confidence Limits	
Attributable Risk	0.20531	0.09789	0.29994
Population Attributable Risk	0.02799	0.01070	0.04497

Similar to the results of using the SMR estimates, the attributable risk fraction (0.205) indicates that 20.5% of all events in the chemical exposure group are attributed to the chemical exposure, and the population attributable risk fraction (0.028) indicates that about 2.8% of all events in the total population are attributed to the chemical exposure.

Example 116.4: Displaying SMR Results from BY Groups

(View the complete code for this example (sratex4.sas) in the example repository.)

This example illustrates the use of ODS OUTPUT statement to save standardized mortality ratios for different causes and to display these statistics together in a table and in a plot.

The Florida_Cs data set contains the stratum-specific mortality information for stomach cancer and skin cancer in year 2000 for the state of Florida (Florida Department of Health 2000, 2020). The variable Age is the grouping variable that forms the strata in the standardization. The variables Event_C16, Event_C43, and PYear identify the number of events for stomach cancer, the number of events for skin cancer, and the person-years, respectively. The COMMA9. format is specified in the DATA step to input numerical values that contain commas in PYear.

```
data Florida_Cs;
   input Age $1-5 Event_C16 Event_C43 PYear:comma9.;
   datalines;
00-04    0    0    953,785
05-14    0    0  1,997,935
15-24    0    4  1,885,014
25-34    1   14  1,957,573
35-44   19   43  2,356,649
45-54   64   72  2,088,000
55-64  114   70  1,548,371
65-74  201  126  1,447,432
75-84  294  136  1,087,524
85+    136   73    335,944
;
```

The following statements construct and list the mortality information by cancer cause:

```
data Florida_Cs;
   set Florida_Cs;
   Cause='Stomach';  Event=Event_C16;  output;
   Cause='Skin';     Event=Event_C43;  output;
   drop Event_C16 Event_C43;
run;

proc sort data=Florida_Cs;
   by Cause;
run;

proc print data=Florida_Cs;
   var Cause Age Event PYear;
run;
```

Output 116.4.1 Florida Data

Obs	Cause	Age	Event	PYear
1	Skin	00-04	0	953785
2	Skin	05-14	0	1997935
3	Skin	15-24	4	1885014
4	Skin	25-34	14	1957573
5	Skin	35-44	43	2356649
6	Skin	45-54	72	2088000
7	Skin	55-64	70	1548371
8	Skin	65-74	126	1447432
9	Skin	75-84	136	1087524
10	Skin	85+	73	335944
11	Stomach	00-04	0	953785
12	Stomach	05-14	0	1997935
13	Stomach	15-24	0	1885014
14	Stomach	25-34	1	1957573
15	Stomach	35-44	19	2356649
16	Stomach	45-54	64	2088000
17	Stomach	55-64	114	1548371
18	Stomach	65-74	201	1447432
19	Stomach	75-84	294	1087524
20	Stomach	85+	136	335944

The US_Cs data set contains the corresponding stratum-specific mortality information for the United States (Miniño et al. 2002; US Bureau of the Census 2011). The variable Age is the grouping variable that forms the strata in the standardization. The variables Event_C16, Event_C43, and PYear identify the number of events for stomach cancer, the number of events for skin cancer, and the person-years, respectively.

```
data US_Cs;
   input Age $1-5 Event_C16 Event_C43 PYear:comma10.;
   datalines;
00-04     0     0  19,175,798
05-14     1     1  41,077,577
15-24    14    41  39,183,891
25-34   124   186  39,892,024
```

```
35-44   484   626  45,148,527
45-54  1097  1199  37,677,952
55-64  1804  1303  24,274,684
65-74  3054  1637  18,390,986
75-84  3833  1624  12,361,180
85+    2234   803   4,239,587
;
```

The following statements construct and list the mortality information by cancer cause:

```
data US_Cs;
   set US_Cs;
   Cause='Stomach';  Event=Event_C16;  output;
   Cause='Skin';     Event=Event_C43;  output;
   drop Event_C16 Event_C43;
run;

proc sort data=US_Cs;
   by Cause;
run;

proc print data=US_Cs;
   var Cause Age Event PYear;
run;
```

Output 116.4.2 lists the mortality information by cancer cause.

Output 116.4.2 Florida Data

Obs	Cause	Age	Event	PYear
1	Skin	00-04	0	19175798
2	Skin	05-14	1	41077577
3	Skin	15-24	41	39183891
4	Skin	25-34	186	39892024
5	Skin	35-44	626	45148527
6	Skin	45-54	1199	37677952
7	Skin	55-64	1303	24274684
8	Skin	65-74	1637	18390986
9	Skin	75-84	1624	12361180
10	Skin	85+	803	4239587
11	Stomach	00-04	0	19175798
12	Stomach	05-14	1	41077577
13	Stomach	15-24	14	39183891
14	Stomach	25-34	124	39892024
15	Stomach	35-44	484	45148527
16	Stomach	45-54	1097	37677952
17	Stomach	55-64	1804	24274684
18	Stomach	65-74	3054	18390986
19	Stomach	75-84	3833	12361180
20	Stomach	85+	2234	4239587

The following statements invoke the STDRATE procedure and request indirect standardization to compute the skin and stomach SMR estimates for the state of Florida. The BY statement requests separate analyses of

causes that are defined by the Cause variable.

```
ods graphics on;
ods select StdInfo StrataSmrPlot Smr;
proc stdrate data=Florida_Cs refdata=US_Cs
             stat=rate
             method=indirect
             plots=smr
             ;
   population event=Event total=PYear;
   reference  event=Event total=PYear;
   strata Age;
   by Cause;
ods output smr=Smr_Cs;
run;
```

Only the tables and plots that are specified in the ODS SELECT statement are displayed.

The STDINFO option in the ODS SELECT statement requests that the "Standardization Information" table display the standardization information for the first BY group, skin cancer, as shown in Output 116.4.3

Output 116.4.3 Standardization Information

The STDRATE Procedure

Cause=Skin

Standardization Information	
Data Set	WORK.FLORIDA_CS
Reference Data Set	WORK.US_CS
Method	Indirect Standardization
Statistic	Rate
Number of Strata	10
Rate Multiplier	100000

The STRATASMRPLOT option in the ODS SELECT statement requests that the strata SMR plot display stratum-specific SMR estimates for skin cancer with confidence limits, as shown in Output 116.4.4.

Output 116.4.4 Strata SMR Plot

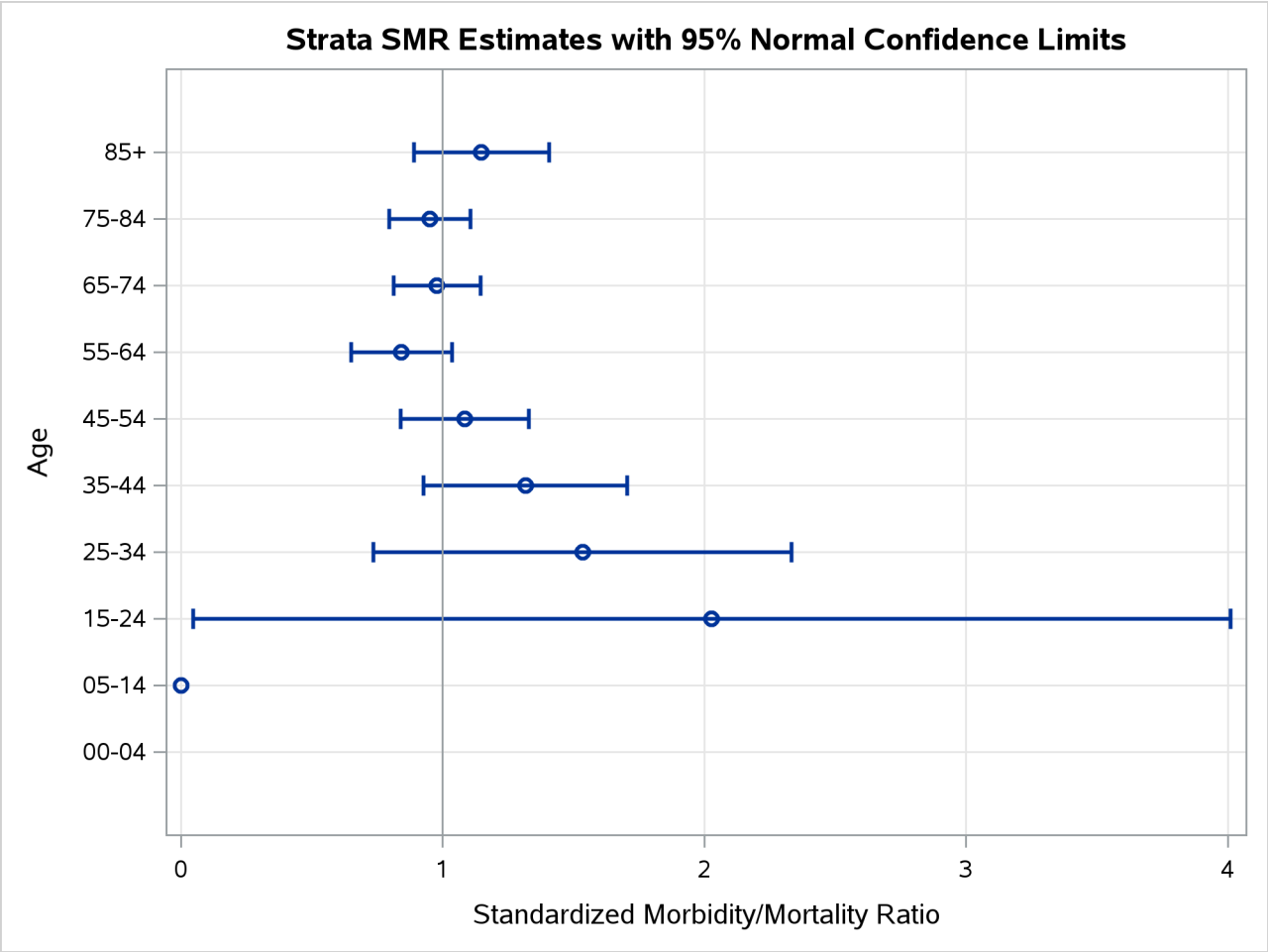

The SMR option in the ODS SELECT statement requests that the "Standardized Morbidity/Mortality Ratio" table display the SMR, its confidence limits, and the test for the null hypothesis H_0 : SMR $= 1$ for skin cancer, as shown in Output 116.4.5. With the default ALPHA=0.05, 95% confidence limits are constructed.

Output 116.4.5 Standardized Morbidity/Mortality Ratio

Cause=Skin

Standardized Morbidity/Mortality Ratio							
				95% Normal Confidence Limits			
Observed Events	Expected Events	SMR	Standard Error			Z	Pr > |Z|
538	528.726	1.0175	0.0439	0.9316	1.1035	0.40	0.6893

Similarly, the "Standardization Information" table in Output 116.4.6 displays the standardization information for the second BY group, stomach cancer.

Output 116.4.6 Standardization Information

The STDRATE Procedure

Cause=Stomach

Standardization Information	
Data Set	WORK.FLORIDA_CS
Reference Data Set	WORK.US_CS
Method	Indirect Standardization
Statistic	Rate
Number of Strata	10
Rate Multiplier	100000

The "Strata SMR Plot" displays stratum-specific SMR estimates with confidence limits for stomach cancer, as shown in Output 116.4.7.

Output 116.4.7 Strata SMR Plot

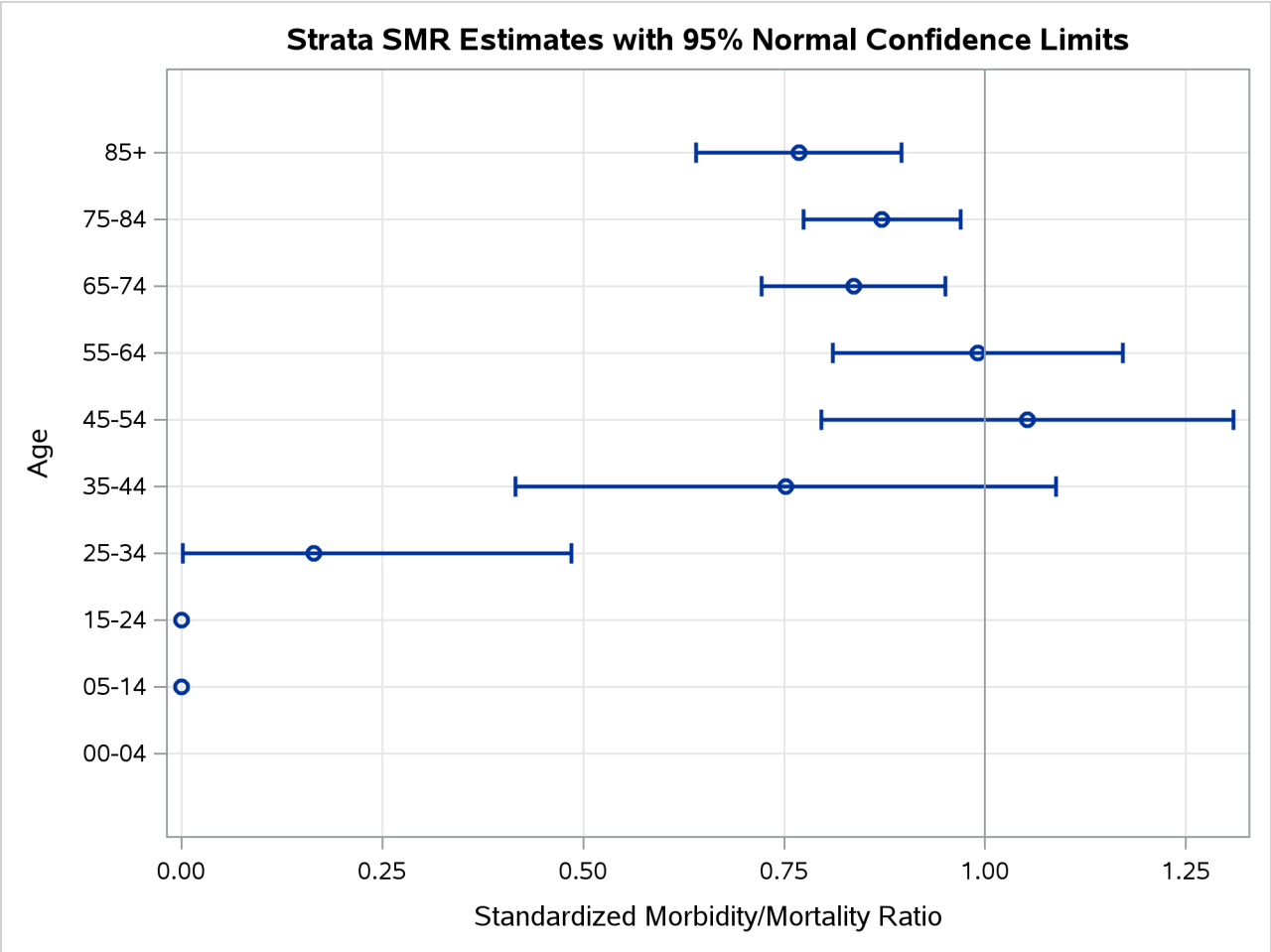

The "Standardized Morbidity/Mortality Ratio" table displays the SMR, its confidence limits, and the test for the null hypothesis H_0 : SMR $= 1$ for stomach cancer, as shown in Output 116.4.8.

Output 116.4.8 Standardized Morbidity/Mortality Ratio

Cause=Stomach

Standardized Morbidity/Mortality Ratio							
Observed Events	Expected Events	SMR	Standard Error	95% Normal Confidence Limits		Z	Pr > |Z|
829	962.537	0.8613	0.0299	0.8026	0.9199	-4.64	<.0001

The ODS OUTPUT SMR=SMR_CS statement requests that the "Standardized Morbidity/Mortality Ratio" tables for the two cancer causes be saved in the data set Smr_Cs. The following statements display the selected output variables for the data set:

```
proc print data=Smr_Cs;
   var Cause ObservedEvents ExpectedEvents Smr SmrLcl SmrUcl;
run;
```

Output 116.4.9 SMR Results from BY Groups

Obs	Cause	ObservedEvents	ExpectedEvents	Smr	SmrLcl	SmrUcl
1	Skin	538	528.726	1.0175	0.9316	1.1035
2	Stomach	829	962.537	0.8613	0.8026	0.9199

The table in Output 116.4.9 shows that the study population (state of Florida) has a higher skin cancer rate and a lower stomach cancer rate than the reference population (United States), but only the lower stomach cancer rate is significant because its corresponding SMR upper confidence limit (0.9199) is less than 1.

The following statements display the standardized morbidity/mortality ratios for the two causes in a plot:

```
proc sgplot data=Smr_Cs;
   scatter y=Cause x=Smr / group=Cause;
   highlow y=Cause high=SmrUcl low=SmrLcl / highcap=serif lowcap=serif;
   yaxis type=discrete;
   xaxis label="SMR";
   refline 1 / axis=x transparency=0.5;
run;
```

Output 116.4.10 SMR Results

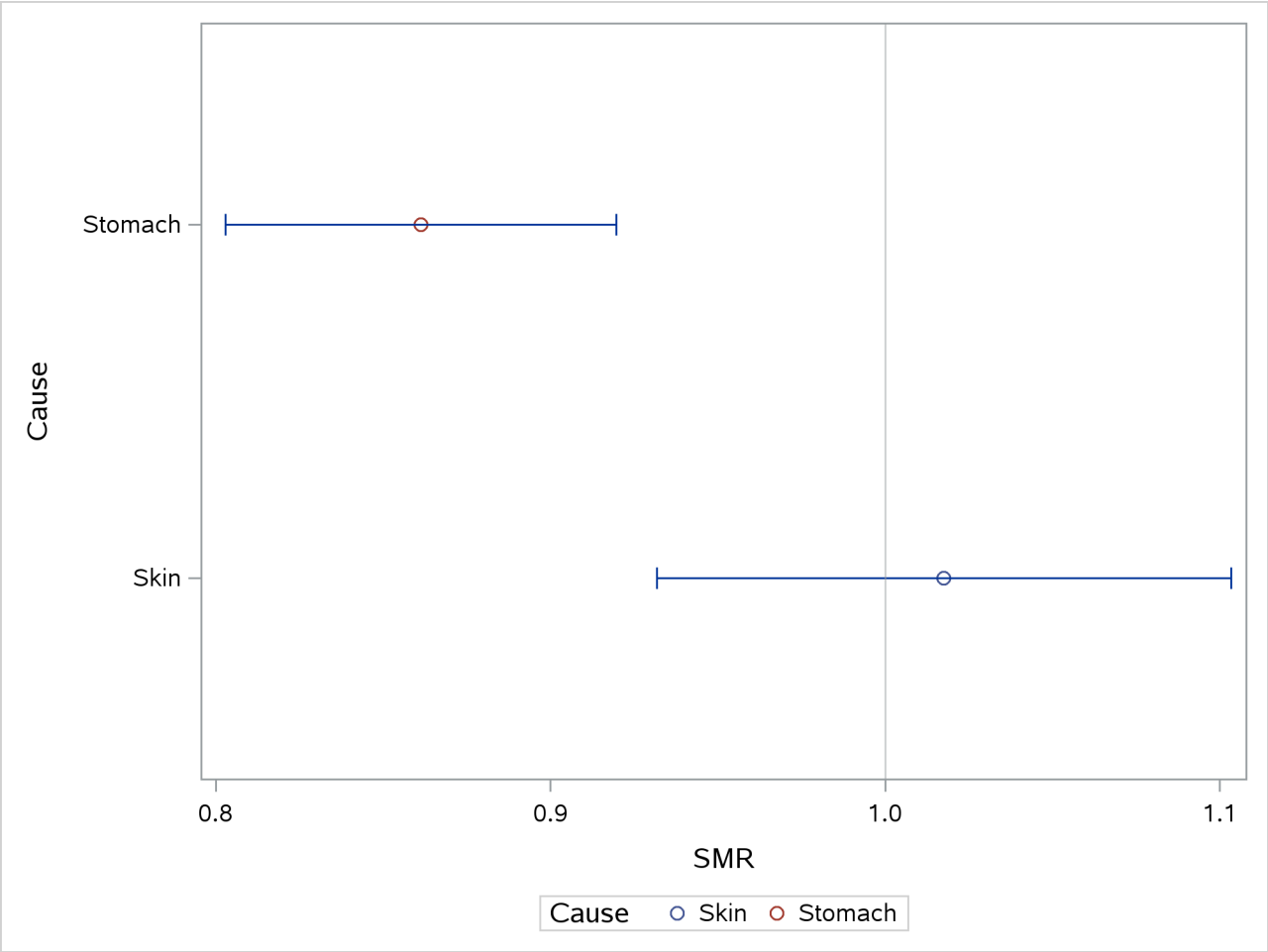

Alternatively, you can also use the following statements to obtain separate analyses for the two cancer causes, and then to display these standardized mortality ratios together in a table and in a plot:

```
/*----- Perform Separate Analyses for Different Causes ------*/
ods graphics on;
ods select StdInfo StrataSmrPlot Smr;
proc stdrate data=Florida_Cs refdata=US_Cs
             stat=rate
             method=indirect
             plots=smr
             ;
   population event=Event_C43 total=PYear;
   reference  event=Event_C43 total=PYear;
   strata Age;
ods output smr=Smr_c43;
run;
```

```
ods select StdInfo StrataSmrPlot Smr;
proc stdrate data=Florida_Cs refdata=US_Cs
             stat=rate
             method=indirect
             plots=smr
             ;
   population event=Event_C16 total=PYear;
   reference  event=Event_C16 total=PYear;
   strata Age;
ods output smr=Smr_c16;
run;

/*------------ Combine SMRs -----------*/
data Smr_C43;
   set Smr_C43;
   length Cause $ 7.;
   Cause='Skin';
run;

data Smr_C16;
   set Smr_C16;
   length Cause $ 7.;
   Cause='Stomach';
run;

data Smr_Cs;
   set Smr_C43 Smr_C16;
run;

/*------- Display the Cause-Specific SMRs --------*/
proc print data=Smr_Cs;
   var Cause ObservedEvents ExpectedEvents Smr SmrLcl SmrUcl;
run;

proc sgplot data=Smr_Cs;
   scatter y=Cause x=Smr / group=Cause;
   highlow y=Cause high=SmrUcl low=SmrLcl / highcap=serif lowcap=serif;
   yaxis type=discrete;
   xaxis label="SMR";
   refline 1 / axis=x transparency=0.5;
run;
```

References

Alaska Bureau of Vital Statistics (2000a). "2000 Annual Report, Appendix I: Population Overview." Accessed February 2012. http://dhss.alaska.gov/dph/VitalStats/Documents/PDFs/2000/annual_report/Appendix_I.pdf.

Alaska Bureau of Vital Statistics (2000b). "2000 Annual Report: Deaths." Accessed Febru-

ary 2012. http://dhss.alaska.gov/dph/VitalStats/Documents/PDFs/2000/annual_report/2000_Annual_Report.pdf.

Breslow, N. E., and Day, N. E. (1987). *The Design and Analysis of Cohort Studies.* Statistical Methods in Cancer Research, IARC Scientific Publications, vol. 2, no. 82. Lyon: International Agency for Research on Cancer.

Elandt-Johnson, R. C., and Johnson, N. L. (1980). *Survival Models and Data Analysis*. New York: John Wiley & Sons.

Fay, M. P., and Feuer, E. J. (1997). "Confidence Intervals for Directly Standardized Rates: A Method Based on the Gamma Distribution." *Statistics in Medicine* 16:791–801.

Florida Department of Health (2000). "Florida Vital Statistics Annual Report 2000." Accessed February 2012. http://www.flpublichealth.com/VSBOOK/pdf/2000/Population.pdf.

Florida Department of Health (2020). "FLHealthCHARTS—Death Count Query System." Accessed June 14, 2021. http://www.flhealthcharts.com/FLQUERY_New/Death/Count.

Friss, R. H., and Sellers, T. A. (2009). *Epidemiology for Public Health Practice*. 4th ed. Sudbury, MA: Jones & Bartlett.

Greenland, S. (2008). "Applications of Stratified Analysis Methods." In *Modern Epidemiology*, 3rd ed., edited by K. J. Rothman, S. Greenland, and T. L. Lash, 283–302. Philadelphia: Lippincott Williams & Wilkins.

Greenland, S., and Robins, J. M. (1985). "Estimation of a Common Effect Parameter from Sparse Follow-Up Data." *Biometrics* 41:55–68.

Greenland, S., and Rothman, K. J. (2008). "Introduction to Stratified Analysis." In *Modern Epidemiology*, 3rd ed., edited by K. J. Rothman, S. Greenland, and T. L. Lash, 258–282. Philadelphia: Lippincott Williams & Wilkins.

Kleinbaum, D. G., Kupper, L. L., and Morgenstern, H. (1982). *Epidemiologic Research: Principles and Quantitative Methods*. New York: Van Nostrand Reinhold.

Miniño, A. M., Arias, E., Kochanek, K. D., Murphy, S. L., and Smith, B. L. (2002). "Deaths: Final Data for 2000." Centers for Disease Control and Prevention. Accessed February 2012. http://www.cdc.gov/nchs/data/nvsr/nvsr50/nvsr50_15.pdf.

Porta, M. (2008). *A Dictionary of Epidemiology*. 5th ed. New York: Oxford University Press.

Tiwari, R. C., Clegg, L. X., and Zou, Z. (2006). "Efficient Interval Estimation for Age-Adjusted Cancer Rates." *Statistics Methods in Medical Research* 15:547–569.

US Bureau of the Census (2011). "Age and Sex Composition: 2010." Accessed February 2012. https://www.census.gov/content/dam/Census/library/publications/2011/dec/c2010br-03.pdf.

Chapter 117
The STEPDISC Procedure

Contents

Overview: STEPDISC Procedure **9743**
Getting Started: STEPDISC Procedure **9745**
Syntax: STEPDISC Procedure **9749**
PROC STEPDISC Statement 9749
BY Statement 9753
CLASS Statement 9754
FREQ Statement 9754
VAR Statement 9754
WEIGHT Statement 9754
Details: STEPDISC Procedure **9755**
Missing Values 9755
Input Data Sets 9755
Computational Resources 9756
Displayed Output 9757
ODS Table Names 9759
Example: STEPDISC Procedure **9760**
Example 117.1: Performing a Stepwise Discriminant Analysis 9760
References **9765**

Overview: STEPDISC Procedure

Given a classification variable and several quantitative variables, the STEPDISC procedure performs a stepwise discriminant analysis to select a subset of the quantitative variables for use in discriminating among the classes. The set of variables that make up each class is assumed to be multivariate normal with a common covariance matrix. The STEPDISC procedure can use forward selection, backward elimination, or stepwise selection (Klecka 1980). The STEPDISC procedure is a useful prelude to further analyses with the CANDISC procedure or the DISCRIM procedure.

With PROC STEPDISC, variables are chosen to enter or leave the model according to one of two criteria:

- the significance level of an F test from an analysis of covariance, where the variables already chosen act as covariates and the variable under consideration is the dependent variable
- the squared partial correlation for predicting the variable under consideration from the CLASS variable, controlling for the effects of the variables already selected for the model

Forward selection begins with no variables in the model. At each step, PROC STEPDISC enters the variable that contributes most to the discriminatory power of the model as measured by Wilks' lambda, the likelihood ratio criterion. When none of the unselected variables meet the entry criterion, the forward selection process stops.

Backward elimination begins with all variables in the model except those that are linearly dependent on previous variables in the VAR statement. At each step, the variable that contributes least to the discriminatory power of the model as measured by Wilks' lambda is removed. When all remaining variables meet the criterion to stay in the model, the backward elimination process stops.

Stepwise selection begins, like forward selection, with no variables in the model. At each step, the model is examined. If the variable in the model that contributes least to the discriminatory power of the model as measured by Wilks' lambda fails to meet the criterion to stay, then that variable is removed. Otherwise, the variable not in the model that contributes most to the discriminatory power of the model is entered. When all variables in the model meet the criterion to stay and none of the other variables meet the criterion to enter, the stepwise selection process stops. Stepwise selection is the default method of variable selection.

It is important to realize that, in the selection of variables for entry, only one variable can be entered into the model at each step. The selection process does not take into account the relationships between variables that have not yet been selected. Thus, some important variables could be excluded in the process. Also, Wilks' lambda might not be the best measure of discriminatory power for your application. However, if you use PROC STEPDISC carefully, in combination with your knowledge of the data and careful cross validation, it can be a valuable aid in selecting variables for a discrimination model.

As with any stepwise procedure, it is important to remember that when many significance tests are performed, each at a level of, for example, 5% (0.05), the overall probability of rejecting at least one true null hypothesis is much larger than 5%. If you want to prevent including any variables that do not contribute to the discriminatory power of the model in the population, you should specify a very small significance level. In most applications, all variables considered have some discriminatory power, however small. To choose the model that provides the best discrimination by using the sample estimates, you need only to guard against estimating more parameters than can be reliably estimated with the given sample size.

Costanza and Afifi (1979) use Monte Carlo studies to compare alternative stopping rules that can be used with the forward selection method in the two-group multivariate normal classification problem. Five different numbers of variables, ranging from 10 to 30, are considered in the studies. The comparison is based on conditional and estimated unconditional probabilities of correct classification. They conclude that the use of a moderate significance level, in the range of 10 to 25 percent, often performs better than the use of a much larger or a much smaller significance level.

The significance level and the squared partial correlation criteria select variables in the same order, although they might select different numbers of variables. Increasing the sample size tends to increase the number of variables selected when you are using significance levels, but it has little effect on the number selected by using squared partial correlations.

See Chapter 11, "Introduction to Discriminant Procedures," for more information about discriminant analysis.

Getting Started: STEPDISC Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/stepdgs.sas.)

The data in this example are measurements of 159 fish caught in Finland's Lake Laengelmaevesi; this data set is available from the Puranen. For each of the seven species (bream, roach, whitefish, parkki, perch, pike, and smelt) the weight, length, height, and width of each fish are tallied. Three different length measurements are recorded: from the nose of the fish to the beginning of its tail, from the nose to the notch of its tail, and from the nose to the end of its tail. The height and width are recorded as percentages of the third length variable. The fish data set is available from the Sashelp library. PROC STEPDISC will select a subset of the six quantitative variables that might be useful for differentiating between the fish species. This subset is used in conjunction with PROC CANDISC and PROC DISCRIM to develop discrimination models.

The following steps use PROC STEPDISC to select a subset of potential discriminator variables. By default, PROC STEPDISC uses stepwise selection on all numeric variables that are not listed in other statements, and the significance levels for a variable to enter the subset and to stay in the subset are set to 0.15. The following statements produce Figure 117.1 through Figure 117.5:

```
title 'Fish Measurement Data';

proc stepdisc data=sashelp.fish;
   class Species;
run;
```

PROC STEPDISC begins by displaying summary information about the analysis (see Figure 117.1). This information includes the number of observations with nonmissing values, the number of classes in the classification variable (specified by the CLASS statement), the number of quantitative variables under consideration, the significance criteria for variables to enter and to stay in the model, and the method of variable selection being used. The frequency of each class is also displayed.

Figure 117.1 Summary Information

Fish Measurement Data

The STEPDISC Procedure

The Method for Selecting Variables is STEPWISE			
Total Sample Size	158	Variable(s) in the Analysis	6
Class Levels	7	Variable(s) Will Be Included	0
		Significance Level to Enter	0.15
		Significance Level to Stay	0.15

Number of Observations Read	159
Number of Observations Used	158

Figure 117.1 *continued*

Class Level Information				
Species	**Variable Name**	**Frequency**	**Weight**	**Proportion**
Bream	Bream	34	34.0000	0.215190
Parkki	Parkki	11	11.0000	0.069620
Perch	Perch	56	56.0000	0.354430
Pike	Pike	17	17.0000	0.107595
Roach	Roach	20	20.0000	0.126582
Smelt	Smelt	14	14.0000	0.088608
Whitefish	Whitefish	6	6.0000	0.037975

For each entry step, the statistics for entry are displayed for all variables not currently selected (see Figure 117.2). The variable selected to enter at this step (if any) is displayed, as well as all the variables currently selected. Next are multivariate statistics that take into account all previously selected variables and the newly entered variable.

Figure 117.2 Step 1: Variable HEIGHT Selected for Entry

Fish Measurement Data

The STEPDISC Procedure
Stepwise Selection: Step 1

Statistics for Entry, DF = 6, 151				
Variable	**R-Square**	**F Value**	**Pr > F**	**Tolerance**
Weight	0.3750	15.10	<.0001	1.0000
Length1	0.6017	38.02	<.0001	1.0000
Length2	0.6098	39.32	<.0001	1.0000
Length3	0.6280	42.49	<.0001	1.0000
Height	0.7553	77.69	<.0001	1.0000
Width	0.4806	23.29	<.0001	1.0000

Variable Height will be entered.

Variable(s) That Have Been Entered
Height

Multivariate Statistics					
Statistic	**Value**	**F Value**	**Num DF**	**Den DF**	**Pr > F**
Wilks' Lambda	0.244670	77.69	6	151	<.0001
Pillai's Trace	0.755330	77.69	6	151	<.0001
Average Squared Canonical Correlation	0.125888				

For each removal step (Figure 117.3), the statistics for removal are displayed for all variables currently entered. The variable to be removed at this step (if any) is displayed. If no variable meets the criterion to be removed and the maximum number of steps as specified by the MAXSTEP= option has not been attained,

then the procedure continues with another entry step.

Figure 117.3 Step 2: No Variable Is Removed; Variable Length2 Added

Fish Measurement Data

The STEPDISC Procedure
Stepwise Selection: Step 2

Statistics for Removal, DF = 6, 151			
Variable	**R-Square**	**F Value**	**Pr > F**
Height	0.7553	77.69	<.0001

No variables can be removed.

Statistics for Entry, DF = 6, 150				
Variable	**Partial R-Square**	**F Value**	**Pr > F**	**Tolerance**
Weight	0.7388	70.71	<.0001	0.4690
Length1	0.9220	295.35	<.0001	0.6083
Length2	0.9229	299.31	<.0001	0.5892
Length3	0.9173	277.37	<.0001	0.5056
Width	0.8783	180.44	<.0001	0.3699

Variable Length2 will be entered.

Variable(s) That Have Been Entered
Length2 Height

Multivariate Statistics					
Statistic	**Value**	**F Value**	**Num DF**	**Den DF**	**Pr > F**
Wilks' Lambda	0.018861	157.04	12	300	<.0001
Pillai's Trace	1.554349	87.78	12	302	<.0001
Average Squared Canonical Correlation	0.259058				

The stepwise procedure terminates either when no variable can be removed and no variable can be entered or when the maximum number of steps as specified by the MAXSTEP= option has been attained. In this example at step 7 no variables can be either removed or entered (Figure 117.4). Steps 3 through 6 are not displayed in this document.

Figure 117.4 Step 7: No Variables Entered or Removed

Fish Measurement Data

The STEPDISC Procedure
Stepwise Selection: Step 7

Statistics for Removal, DF = 6, 146			
Variable	Partial R-Square	F Value	Pr > F
Weight	0.4521	20.08	<.0001
Length1	0.2987	10.36	<.0001
Length2	0.5250	26.89	<.0001
Length3	0.7948	94.25	<.0001
Height	0.7257	64.37	<.0001
Width	0.5757	33.02	<.0001

No variables can be removed.

PROC STEPDISC ends by displaying a summary of the steps.

Figure 117.5 Step Summary

No further steps are possible.

Fish Measurement Data

The STEPDISC Procedure

Stepwise Selection Summary

Step	Number In	Entered	Removed	Partial R-Square	F Value	Pr > F	Wilks' Lambda	Pr < Lambda	Average Squared Canonical Correlation	Pr > ASCC
1	1	Height		0.7553	77.69	<.0001	0.24466983	<.0001	0.12588836	<.0001
2	2	Length2		0.9229	299.31	<.0001	0.01886065	<.0001	0.25905822	<.0001
3	3	Length3		0.8826	186.77	<.0001	0.00221342	<.0001	0.38427100	<.0001
4	4	Width		0.5775	33.72	<.0001	0.00093510	<.0001	0.45200732	<.0001
5	5	Weight		0.4461	19.73	<.0001	0.00051794	<.0001	0.49488458	<.0001
6	6	Length1		0.2987	10.36	<.0001	0.00036325	<.0001	0.51744189	<.0001

All the variables in the data set are found to have potential discriminatory power. These variables are used to develop discrimination models in both the CANDISC and DISCRIM procedure chapters.

Syntax: STEPDISC Procedure

The following statements are available in the STEPDISC procedure:

```
PROC STEPDISC <options> ;
    CLASS variable ;
    BY variables ;
    FREQ variable ;
    VAR variables ;
    WEIGHT variable ;
```

The BY, CLASS, FREQ, VAR, and WEIGHT statements are described after the PROC STEPDISC statement.

PROC STEPDISC Statement

```
PROC STEPDISC <options> ;
```

The PROC STEPDISC statement invokes the STEPDISC procedure. Table 117.1 summarizes the options available in the PROC STEPDISC statement.

Table 117.1 STEPDISC Procedure Options

Option	Description
Input Data Set	
DATA=	Specifies input SAS data set
Method Details	
MAXMACRO=	Specifies maximum macro variable lists
METHOD=	Specifies method
SINGULAR=	Specifies singularity
Control Stepwise Selection	
SLENTRY=	Specifies entry significance
SLSTAY=	Specifies staying significance
PR2ENTRY=	Specifies entry partial R square
PR2STAY=	Specifies staying partial R square
INCLUDE=	Forces inclusion of variables
MAXSTEP=	Specifies maximum number of steps
START=	Specifies variables to begin
STOP=	Specifies number of variables in final model
Control Displayed Output	
ALL	Displays all
BCORR	Displays between correlations
BCOV	Displays between covariances
BSSCP	Displays between SSCPs
PCORR	Displays pooled correlations
PCOV	Displays pooled covariances

Table 117.1 *continued*

Option	Description
PSSCP	Displays pooled SSCPs
SHORT	Suppresses output
SIMPLE	Displays descriptive statistics
STDMEAN	Displays standardized class means
TCORR	Displays total correlations
TCOV	Displays total covariances
TSSCP	Displays total SSCPs
WCORR	Displays within correlations
WCOV	Displays within covariances
WSSCP	Displays within SSCPs

ALL
activates all of the display options.

BCORR
displays between-class correlations.

BCOV
displays between-class covariances. The between-class covariance matrix equals the between-class SSCP matrix divided by $n(c-1)/c$, where n is the number of observations and c is the number of classes. The between-class covariances should be interpreted in comparison with the total-sample and within-class covariances, not as formal estimates of population parameters.

BSSCP
displays the between-class SSCP matrix.

DATA=*SAS-data-set*
specifies the data set to be analyzed. The data set can be an ordinary SAS data set or one of several specially structured data sets created by statistical procedures available with SAS/STAT software. These specially structured data sets include TYPE=CORR, COV, CSSCP, and SSCP. If the DATA= option is omitted, the procedure uses the most recently created SAS data set.

INCLUDE=*n*
includes the first *n* variables in the VAR statement in every model. By default, INCLUDE=0.

MAXMACRO=*n*
specifies the maximum number of macro variables with independent variable lists to create. By default, MAXMACRO=100. PROC STEPDISC saves the list of selected variables in a macro variable, &_StdVar. Suppose your input variable list consists of x1-x10; then &_StdVar would be set to x1 x3 x4 x10 if, for example, the first, third, fourth, and tenth variables were selected for the model. This list can be used, for example, in a subsequent procedure's VAR statement as follows:

```
var &_stdvar;
```

With BY processing, one macro variable is created for each BY group, and the macro variables are indexed by the BY-group number. The MAXMACRO= option can be used to either limit or increase the number of these macro variables in processing data sets with many BY groups. The macro variables are created as follows:

With no BY processing, PROC STEPDISC creates the following:

_StdVar	selected variables
_StdVar1	selected variables
_StdNumBys	number of BY groups (1)
_StdNumMacroBys	number of _StdVar*i* macro variables actually made (1)

With BY processing, PROC STEPDISC creates the following:

_StdVar	selected variables for BY group 1
_StdVar1	selected variables for BY group 1
_StdVar2	selected variables for BY group 2
.	
.	
.	
_StdVar*m*	selected variables for BY group *m*, where a number is substituted for *m*
_StdNumBys	*n*, the number of BY groups
_StdNumMacroBys	the number *m* of _StdVar*i* macro variables actually made. This value might be less than _StdNumbys = *n*, and it is less than or equal to the MAXMACRO= value.

MAXSTEP=*n*

specifies the maximum number of steps. By default, MAXSTEP= two times the number of variables in the VAR statement.

METHOD=BACKWARD | BW

METHOD=FORWARD | FW

METHOD=STEPWISE | SW

specifies the method used to select the variables in the model. The BACKWARD method specifies backward elimination, FORWARD specifies forward selection, and STEPWISE specifies stepwise selection. By default, METHOD=STEPWISE.

PCORR

displays pooled within-class correlations (partial correlations based on the pooled within-class covariances).

PCOV

displays pooled within-class covariances.

PR2ENTRY=*p*

PR2E=*p*

specifies the partial R square for adding variables in the forward selection mode, where $p \leq 1$.

PR2STAY=*p*

PR2S=*p*

specifies the partial R square for retaining variables in the backward elimination mode, where $p \leq 1$.

PSSCP

displays the pooled within-class corrected SSCP matrix.

SHORT

suppresses the displayed output from each step.

SIMPLE

displays simple descriptive statistics for the total sample and within each class.

SINGULAR=*p*

specifies the singularity criterion for entering variables, where $0 < p < 1$. PROC STEPDISC precludes the entry of a variable if the squared multiple correlation of the variable with the variables already in the model exceeds $1 - p$. With more than one variable already in the model, PROC STEPDISC also excludes a variable if it would cause any of the variables already in the model to have a squared multiple correlation (with the entering variable and the other variables in the model) exceeding $1 - p$. By default, SINGULAR= 1E–8.

SLENTRY=*p*

SLE=*p*

specifies the significance level for adding variables in the forward selection mode, where $0 \leq p \leq 1$. The default value is 0.15.

SLSTAY=*p*

SLS=*p*

specifies the significance level for retaining variables in the backward elimination mode, where $0 \leq p \leq 1$. The default value is 0.15.

START=*n*

specifies that the first *n* variables in the VAR statement be used to begin the selection process. When you specify METHOD=FORWARD or METHOD=STEPWISE, the default value is 0; when you specify METHOD=BACKWARD, the default value is the number of variables in the VAR statement.

STDMEAN

displays total-sample and pooled within-class standardized class means.

STOP=*n*

specifies the number of variables in the final model. The STEPDISC procedure stops the selection process when a model with *n* variables is found. This option applies only when you specify METHOD=FORWARD or METHOD=BACKWARD. When you specify METHOD=FORWARD, the default value is the number of variables in the VAR statement; when you specify METHOD=BACKWARD, the default value is 0.

TCORR
displays total-sample correlations.

TCOV
displays total-sample covariances.

TSSCP
displays the total-sample corrected SSCP matrix.

WCORR
displays within-class correlations for each class level.

WCOV
displays within-class covariances for each class level.

WSSCP
displays the within-class corrected SSCP matrix for each class level.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC STEPDISC to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the STEPDISC procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* **;**

The values of the CLASS variable define the groups for analysis. Class levels are determined by the formatted values of the CLASS variable. The CLASS variable can be numeric or character. A CLASS statement is required.

FREQ Statement

FREQ *variable* **;**

If a variable in the data set represents the frequency of occurrence for the other values in the observation, include the name of the variable in a FREQ statement. The procedure then treats the data set as if each observation appears *n* times, where *n* is the value of the FREQ variable for the observation. The total number of observations is considered to be equal to the sum of the FREQ variable when the procedure determines degrees of freedom for significance probabilities.

If the value of the FREQ variable is missing or is less than one, the observation is not used in the analysis. If the value is not an integer, the value is truncated to an integer.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the quantitative variables eligible for selection. The default is all numeric variables not listed in other statements.

WEIGHT Statement

WEIGHT *variable* **;**

To use relative weights for each observation in the input data set, place the weights in a variable in the data set and specify the name in a WEIGHT statement. This is often done when the variance associated with each observation is different and the values of the WEIGHT variable are proportional to the reciprocals of the variances. If the value of the WEIGHT variable is missing or is less than zero, then a value of zero for the weight is assumed.

The WEIGHT and FREQ statements have a similar effect except that the WEIGHT statement does not alter the degrees of freedom.

Details: STEPDISC Procedure

Missing Values

Observations containing missing values are omitted from the analysis.

Input Data Sets

The input data set can be an ordinary SAS data set or one of several specially structured data sets created by statistical procedures available with SAS/STAT software. For more information about these data sets, see Appendix A, "Special SAS Data Sets." The BY variable in these data sets becomes the CLASS variable in PROC STEPDISC. These specially structured data sets include the following:

- TYPE=CORR data sets created by PROC CORR by using a BY statement
- TYPE=COV data sets created by PROC PRINCOMP by using both the COV option and a BY statement
- TYPE=CSSCP data sets created by PROC CORR by using the CSSCP option and a BY statement, where the OUT= data set is assigned TYPE=CSSCP with the TYPE= data set option
- TYPE=SSCP data sets created by PROC REG by using both the OUTSSCP= option and a BY statement

When the input data set is TYPE=CORR, TYPE=COV, or TYPE=CSSCP, the STEPDISC procedure reads the number of observations for each class from the observations with _TYPE_='N' and the variable means in each class from the observations with _TYPE_='MEAN'. The procedure then reads the within-class correlations from the observations with _TYPE_='CORR', the standard deviations from the observations with _TYPE_='STD' (data set TYPE=CORR), the within-class covariances from the observations with _TYPE_='COV' (data set TYPE=COV), or the within-class corrected sums of squares and crossproducts from the observations with _TYPE_='CSSCP' (data set TYPE=CSSCP).

When the data set does not include any observations with _TYPE_='CORR' (data set TYPE=CORR), _TYPE_='COV' (data set TYPE=COV), or _TYPE_='CSSCP' (data set TYPE=CSSCP) for each class, PROC STEPDISC reads the pooled within-class information from the data set. In this case, the STEPDISC procedure reads the pooled within-class correlations from the observations with _TYPE_='PCORR', the pooled within-class standard deviations from the observations with _TYPE_='PSTD' (data set TYPE=CORR), the pooled within-class covariances from the observations with _TYPE_='PCOV' (data set TYPE=COV), or the pooled within-class corrected SSCP matrix from the observations with_TYPE_='PSSCP' (data set TYPE=CSSCP).

When the input data set is TYPE=SSCP, the STEPDISC procedure reads the number of observations for each class from the observations with _TYPE_='N', the sum of weights of observations from the variable INTERCEPT in observations with _TYPE_='SSCP' and _NAME_='INTERCEPT', the variable sums from the analysis variables in observations with _TYPE_='SSCP' and _NAME_='INTERCEPT', and the uncorrected sums of squares and crossproducts from the analysis variables in observations with _TYPE_='SSCP' and _NAME_=*variable-names*.

Computational Resources

In the following discussion, let

$$
\begin{aligned}
n &= \text{number of observations} \\
c &= \text{number of class levels} \\
v &= \text{number of variables in the VAR list} \\
l &= \text{length of the CLASS variable} \\
t &= v + c - 1
\end{aligned}
$$

Memory Requirements

The amount of memory in bytes for temporary storage needed to process the data is

$$c(4v^2 + 28v + 3l + 4c + 72) + 16v^2 + 92v + 4t^2 + 20t + 4l$$

Additional temporary storage of 72 bytes at each step is also required to store the results.

Time Requirements

The following factors determine the time requirements of a stepwise discriminant analysis:

- The time needed for reading the data and computing covariance matrices is proportional to nv^2. The STEPDISC procedure must also look up each class level in the list. This is faster if the data are sorted by the CLASS variable. The time for looking up class levels is proportional to a value ranging from n to $n \ln(c)$.
- The time needed for stepwise discriminant analysis is proportional to the number of steps required to select the set of variables in the discrimination model. The number of steps required depends on the data set itself and the selection method and criterion used in the procedure. Each forward or backward step takes time proportional to $(v + c)^2$.

Displayed Output

The displayed output from PROC STEPDISC includes the class level information table. For each level of the classification variable, the following information is provided: the output data set variable name, frequency sum, weight sum, and the proportion of the total sample.

The optional output from PROC STEPDISC includes the following:

The optional output includes the following:

- Within-class SSCP matrices for each group
- Pooled within-class SSCP matrix
- Between-class SSCP matrix
- Total-sample SSCP matrix
- Within-class covariance matrices for each group
- Pooled within-class covariance matrix
- Between-class covariance matrix, equal to the between-class SSCP matrix divided by $n(c-1)/c$, where n is the number of observations and c is the number of classes
- Total-sample covariance matrix
- Within-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the within-class population correlation coefficients are zero
- Pooled within-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the partial population correlation coefficients are zero
- Between-class correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the between-class population correlation coefficients are zero
- Total-sample correlation coefficients and $\mathrm{Pr} > |r|$ to test the hypothesis that the total population correlation coefficients are zero
- Simple statistics, including N (the number of observations), sum, mean, variance, and standard deviation for the total sample and within each class
- Total-sample standardized class means, obtained by subtracting the grand mean from each class mean and dividing by the total-sample standard deviation
- Pooled within-class standardized class means, obtained by subtracting the grand mean from each class mean and dividing by the pooled within-class standard deviation

At each step, the following statistics are displayed:

- for each variable considered for entry or removal: partial R-square, the squared (partial) correlation, the F statistic, and $\mathrm{Pr} > F$, the probability level, from a one-way analysis of covariance

- the minimum tolerance for entering each variable. A variable is entered only if its tolerance and the tolerances for all variables already in the model are greater than the value specified in the SINGULAR= option. The tolerance for the entering variable is $1 - R^2$ from regressing the entering variable on the other variables already in the model. The tolerance for a variable already in the model is $1 - R^2$ from regressing that variable on the entering variable and the other variables already in the model. With m variables already in the model, for each entering variable, $m + 1$ multiple regressions are performed by using the entering variable and each of the m variables already in the model as a dependent variable. These $m + 1$ tolerances are computed for each entering variable, and the minimum tolerance is displayed for each.

  The tolerance is computed by using the total-sample correlation matrix. It is customary to compute tolerance by using the pooled within-class correlation matrix (Jennrich 1977), but it is possible for a variable with excellent discriminatory power to have a high total-sample tolerance and a low pooled within-class tolerance. For example, PROC STEPDISC enters a variable that yields perfect discrimination (that is, produces a canonical correlation of one), but a program that uses pooled within-class tolerance does not.

- the variable label, if any

- the name of the variable chosen

- the variables already selected or removed

- Wilks' lambda and the associated F approximation with degrees of freedom and $\Pr < F$, the associated probability level after the selected variable has been entered or removed. Wilks' lambda is the likelihood ratio statistic for testing the hypothesis that the means of the classes on the selected variables are equal in the population (see the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures.") Lambda is close to zero if any two groups are well separated.

- Pillai's trace and the associated F approximation with degrees of freedom and $\Pr > F$, the associated probability level after the selected variable has been entered or removed. Pillai's trace is a multivariate statistic for testing the hypothesis that the means of the classes on the selected variables are equal in the population (see the section "Multivariate Tests" on page 98 in Chapter 4, "Introduction to Regression Procedures").

- Average squared canonical correlation (ASCC). The ASCC is Pillai's trace divided by the number of groups minus 1. The ASCC is close to 1 if all groups are well separated and if all or most directions in the discriminant space show good separation for at least two groups.

- Summary to give statistics associated with the variable chosen at each step. The summary includes the following:

  - Step number
  - Variable entered or removed
  - Number in, the number of variables in the model
  - Partial R-square
  - the F value for entering or removing the variable
  - $\Pr > F$, the probability level for the F statistic
  - Wilks' lambda

- $\Pr < \text{Lambda}$ based on the F approximation to Wilks' lambda
- Average squared canonical correlation
- $\Pr > \text{ASCC}$ based on the F approximation to Pillai's trace
- the variable label, if any

ODS Table Names

PROC STEPDISC assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 117.2 along with the PROC STEPDISC statement options needed to produce the table. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 117.2 ODS Tables Produced by PROC STEPDISC

ODS Table Name	Description	Option
BCorr	Between-class correlations	BCORR
BCov	Between-class covariances	BCOV
BSSCP	Between-class SSCP matrix	BSSCP
Counts	Number of observations, variables, classes, df	default
CovDF	Nonprinting table of df for covariance matrices	any *COV option
Levels	Class level information	default
Messages	Entry/removal messages	default
Multivariate	Multivariate statistics	default
NObs	Number of observations	default
PCorr	Pooled within-class correlations	PCORR
PCov	Pooled within-class covariances	PCOV
PSSCP	Pooled within-class SSCP matrix	PSSCP
PStdMeans	Pooled standardized class means	STDMEAN
SimpleStatistics	Simple statistics	SIMPLE
Steps	Stepwise selection entry/removal	default
Summary	Stepwise selection summary	default
TCorr	Total-sample correlations	TCORR
TCov	Total-sample covariances	TCOV
TSSCP	Total-sample SSCP matrix	TSSCP
TStdMeans	Total standardized class means	STDMEAN
Variables	Variable lists	default
WCorr	Within-class correlations	WCORR
WCov	Within-class covariances	WCOV
WSSCP	Within-class SSCP matrices	WSSCP

Example: STEPDISC Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 117.1: Performing a Stepwise Discriminant Analysis

(View the complete code for this example (stepdex.sas) in the example repository.)

The iris data published by Fisher (1936) have been widely used for examples in discriminant analysis and cluster analysis. The sepal length, sepal width, petal length, and petal width are measured in millimeters on 50 iris specimens from each of three species: *Iris setosa*, *I. versicolor*, and *I. virginica*. The iris data set is available from the Sashelp library.

A stepwise discriminant analysis is performed by using stepwise selection.

In the PROC STEPDISC statement, the BSSCP and TSSCP options display the between-class SSCP matrix and the total-sample corrected SSCP matrix. By default, the significance level of an F test from an analysis of covariance is used as the selection criterion. The variable under consideration is the dependent variable, and the variables already chosen act as covariates. The following SAS statements produce Output 117.1.1 through Output 117.1.8:

```
title 'Fisher (1936) Iris Data';

%let _stdvar = ;
proc stepdisc data=sashelp.iris bsscp tsscp;
   class Species;
   var SepalLength SepalWidth PetalLength PetalWidth;
run;
```

Output 117.1.1 Iris Data: Summary Information

Fisher (1936) Iris Data

The STEPDISC Procedure

The Method for Selecting Variables is STEPWISE			
Total Sample Size	150	Variable(s) in the Analysis	4
Class Levels	3	Variable(s) Will Be Included	0
		Significance Level to Enter	0.15
		Significance Level to Stay	0.15

Number of Observations Read	150
Number of Observations Used	150

Output 117.1.1 *continued*

Class Level Information				
Species	**Variable Name**	**Frequency**	**Weight**	**Proportion**
Setosa	Setosa	50	50.0000	0.333333
Versicolor	Versicolor	50	50.0000	0.333333
Virginica	Virginica	50	50.0000	0.333333

Output 117.1.2 Iris Data: Between-Class and Total-Sample SSCP Matrices

Fisher (1936) Iris Data

The STEPDISC Procedure

Between-Class SSCP Matrix					
Variable	**Label**	**SepalLength**	**SepalWidth**	**PetalLength**	**PetalWidth**
SepalLength	Sepal Length (mm)	6321.21333	-1995.26667	16524.84000	7127.93333
SepalWidth	Sepal Width (mm)	-1995.26667	1134.49333	-5723.96000	-2293.26667
PetalLength	Petal Length (mm)	16524.84000	-5723.96000	43710.28000	18677.40000
PetalWidth	Petal Width (mm)	7127.93333	-2293.26667	18677.40000	8041.33333

Total-Sample SSCP Matrix					
Variable	**Label**	**SepalLength**	**SepalWidth**	**PetalLength**	**PetalWidth**
SepalLength	Sepal Length (mm)	10216.83333	-632.26667	18987.30000	7692.43333
SepalWidth	Sepal Width (mm)	-632.26667	2830.69333	-4911.88000	-1812.42667
PetalLength	Petal Length (mm)	18987.30000	-4911.88000	46432.54000	19304.58000
PetalWidth	Petal Width (mm)	7692.43333	-1812.42667	19304.58000	8656.99333

In step 1, the tolerance is 1.0 for each variable under consideration because no variables have yet entered the model. The variable PetalLength is selected because its F statistic, 1180.161, is the largest among all variables.

Output 117.1.3 Iris Data: Stepwise Selection Step 1

Fisher (1936) Iris Data

The STEPDISC Procedure
Stepwise Selection: Step 1

Statistics for Entry, DF = 2, 147					
Variable	**Label**	**R-Square**	**F Value**	**Pr > F**	**Tolerance**
SepalLength	Sepal Length (mm)	0.6187	119.26	<.0001	1.0000
SepalWidth	Sepal Width (mm)	0.4008	49.16	<.0001	1.0000
PetalLength	Petal Length (mm)	0.9414	1180.16	<.0001	1.0000
PetalWidth	Petal Width (mm)	0.9289	960.01	<.0001	1.0000

Variable PetalLength will be entered.

Output 117.1.3 *continued*

Variable(s) That Have Been Entered
PetalLength

Multivariate Statistics					
Statistic	**Value**	**F Value**	**Num DF**	**Den DF**	**Pr > F**
Wilks' Lambda	0.058628	1180.16	2	147	<.0001
Pillai's Trace	0.941372	1180.16	2	147	<.0001
Average Squared Canonical Correlation	0.470686				

In step 2, with the variable PetalLength already in the model, PetalLength is tested for removal before a new variable is selected for entry. Since PetalLength meets the criterion to stay, it is used as a covariate in the analysis of covariance for variable selection. The variable SepalWidth is selected because its F statistic, 43.035, is the largest among all variables not in the model and because its associated tolerance, 0.8164, meets the criterion to enter. The process is repeated in steps 3 and 4. The variable PetalWidth is entered in step 3, and the variable SepalLength is entered in step 4.

Output 117.1.4 Iris Data: Stepwise Selection Step 2

Fisher (1936) Iris Data

The STEPDISC Procedure
Stepwise Selection: Step 2

Statistics for Removal, DF = 2, 147				
Variable	**Label**	**R-Square**	**F Value**	**Pr > F**
PetalLength	Petal Length (mm)	0.9414	1180.16	<.0001

No variables can be removed.

Statistics for Entry, DF = 2, 146					
Variable	**Label**	**Partial R-Square**	**F Value**	**Pr > F**	**Tolerance**
SepalLength	Sepal Length (mm)	0.3198	34.32	<.0001	0.2400
SepalWidth	Sepal Width (mm)	0.3709	43.04	<.0001	0.8164
PetalWidth	Petal Width (mm)	0.2533	24.77	<.0001	0.0729

Variable SepalWidth will be entered.

Variable(s) That Have Been Entered
SepalWidth PetalLength

Multivariate Statistics					
Statistic	**Value**	**F Value**	**Num DF**	**Den DF**	**Pr > F**
Wilks' Lambda	0.036884	307.10	4	292	<.0001
Pillai's Trace	1.119908	93.53	4	294	<.0001
Average Squared Canonical Correlation	0.559954				

Output 117.1.5 Iris Data: Stepwise Selection Step 3

Fisher (1936) Iris Data

The STEPDISC Procedure
Stepwise Selection: Step 3

Statistics for Removal, DF = 2, 146

Variable	Label	Partial R-Square	F Value	Pr > F
SepalWidth	Sepal Width (mm)	0.3709	43.04	<.0001
PetalLength	Petal Length (mm)	0.9384	1112.95	<.0001

No variables can be removed.

Statistics for Entry, DF = 2, 145

Variable	Label	Partial R-Square	F Value	Pr > F	Tolerance
SepalLength	Sepal Length (mm)	0.1447	12.27	<.0001	0.1323
PetalWidth	Petal Width (mm)	0.3229	34.57	<.0001	0.0662

Variable PetalWidth will be entered.

Variable(s) That Have Been Entered
SepalWidth PetalLength PetalWidth

Multivariate Statistics

Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.024976	257.50	6	290	<.0001
Pillai's Trace	1.189914	71.49	6	292	<.0001
Average Squared Canonical Correlation	0.594957				

Output 117.1.6 Iris Data: Stepwise Selection Step 4

Fisher (1936) Iris Data

The STEPDISC Procedure
Stepwise Selection: Step 4

Statistics for Removal, DF = 2, 145

Variable	Label	Partial R-Square	F Value	Pr > F
SepalWidth	Sepal Width (mm)	0.4295	54.58	<.0001
PetalLength	Petal Length (mm)	0.3482	38.72	<.0001
PetalWidth	Petal Width (mm)	0.3229	34.57	<.0001

No variables can be removed.

Output 117.1.6 *continued*

Statistics for Entry, DF = 2, 144					
Variable	Label	Partial R-Square	F Value	Pr > F	Tolerance
SepalLength	Sepal Length (mm)	0.0615	4.72	0.0103	0.0320

Variable SepalLength will be entered.

All variables have been entered.

Multivariate Statistics					
Statistic	Value	F Value	Num DF	Den DF	Pr > F
Wilks' Lambda	0.023439	199.15	8	288	<.0001
Pillai's Trace	1.191899	53.47	8	290	<.0001
Average Squared Canonical Correlation	0.595949				

Since no more variables can be added to or removed from the model, the procedure stops at step 5 and displays a summary of the selection process.

Output 117.1.7 Iris Data: Stepwise Selection Step 5

Fisher (1936) Iris Data

The STEPDISC Procedure
Stepwise Selection: Step 5

Statistics for Removal, DF = 2, 144				
Variable	Label	Partial R-Square	F Value	Pr > F
SepalLength	Sepal Length (mm)	0.0615	4.72	0.0103
SepalWidth	Sepal Width (mm)	0.2335	21.94	<.0001
PetalLength	Petal Length (mm)	0.3308	35.59	<.0001
PetalWidth	Petal Width (mm)	0.2570	24.90	<.0001

No variables can be removed.

Output 117.1.8 Iris Data: Stepwise Selection Summary

No further steps are possible.

Output 117.1.8 *continued*

Fisher (1936) Iris Data

The STEPDISC Procedure

Stepwise Selection Summary

Step	Number In	Entered	Removed	Label	Partial R-Square	F Value	Pr > F	Wilks' Lambda	Pr < Lambda	Average Squared Canonical Correlation	Pr > ASCC
1	1	PetalLength		Petal Length (mm)	0.9414	1180.16	<.0001	0.05862828	<.0001	0.47068586	<.0001
2	2	SepalWidth		Sepal Width (mm)	0.3709	43.04	<.0001	0.03688411	<.0001	0.55995394	<.0001
3	3	PetalWidth		Petal Width (mm)	0.3229	34.57	<.0001	0.02497554	<.0001	0.59495691	<.0001
4	4	SepalLength		Sepal Length (mm)	0.0615	4.72	0.0103	0.02343863	<.0001	0.59594941	<.0001

PROC STEPDISC automatically creates a list of the selected variables and stores it in a macro variable. You can submit the following statement to see the list of selected variables:

```
* print the macro variable list;
%put &_stdvar;
```

The macro variable _StdVar contains the following variable list:

```
SepalLength SepalWidth PetalLength PetalWidth
```

You could use this macro variable if you want to analyze these variables in subsequent steps as follows:

```
proc discrim data=sashelp.iris;
   class Species;
   var &_stdvar;
run;
```

The results of this step are not shown.

References

Costanza, M. C., and Afifi, A. A. (1979). "Comparison of Stopping Rules in Forward Stepwise Discriminant Analysis." *Journal of the American Statistical Association* 74:777–785.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Jennrich, R. I. (1977). "Stepwise Discriminant Analysis." In *Statistical Methods for Digital Computers*, edited by K. Enslein, A. Ralston, and H. Wilf, 76–96. New York: John Wiley & Sons.

Klecka, W. R. (1980). *Discriminant Analysis.* Vol. 07-019 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Puranen, J. (1917). "Fish Catch data set (1917)." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Chapter 118
The SURVEYFREQ Procedure

Contents

Overview: SURVEYFREQ Procedure **9768**
Getting Started: SURVEYFREQ Procedure **9769**
Syntax: SURVEYFREQ Procedure **9777**
- PROC SURVEYFREQ Statement 9777
- BY Statement 9789
- CLUSTER Statement 9790
- REPWEIGHTS Statement 9790
- STRATA Statement 9793
- TABLES Statement 9794
- WEIGHT Statement 9821

Details: SURVEYFREQ Procedure **9822**
- Specifying the Sample Design 9822
- Domain Analysis 9824
- Missing Values 9825
- Variance Estimation 9827
  - Taylor Series Variance Estimation 9827
  - Replication Variance Estimation 9828
  - Bootstrap Method 9828
  - Balanced Repeated Replication (BRR) Method 9830
  - Fay's BRR Method 9832
  - Jackknife Method 9834
- Statistical Computations 9836
  - Definitions and Notation 9836
  - Totals 9837
  - Covariances of Frequency Estimates 9838
  - Proportions 9839
  - Row and Column Proportions 9840
  - Confidence Limits for Totals 9841
  - Confidence Limits for Proportions 9842
  - Degrees of Freedom 9846
  - Coefficient of Variation 9847
  - Design Effect 9847
  - Expected Weighted Frequency 9848
  - Sensitivity and Specificity 9849
  - Risks and Risk Difference 9850
  - Odds Ratio and Relative Risks 9852

Discordant Proportion Difference 9854
Kappa Coefficients 9854
Rao-Scott Chi-Square Test 9858
Rao-Scott Likelihood Ratio Chi-Square Test 9864
Wald Chi-Square Test 9865
Wald Log-Linear Chi-Square Test 9867
Output Data Sets 9868
Displayed Output 9869
ODS Table Names 9877
ODS Graphics 9879
Examples: SURVEYFREQ Procedure **9880**
Example 118.1: Two-Way Tables 9880
Example 118.2: Multiway Tables (Domain Analysis) 9883
Example 118.3: Output Data Sets 9885
References **9887**

Overview: SURVEYFREQ Procedure

The SURVEYFREQ procedure produces one-way to *n*-way frequency and crosstabulation tables from sample survey data. These tables include estimates of population totals, population proportions, and their standard errors. Confidence limits, coefficients of variation, and design effects are also available. The procedure provides a variety of options to customize the table display.

For one-way frequency tables, PROC SURVEYFREQ provides Rao-Scott chi-square goodness-of-fit tests, which are adjusted for the sample design. You can test a null hypothesis of equal proportions for a one-way frequency table, or you can input custom null hypothesis proportions for the test. For two-way tables, PROC SURVEYFREQ provides design-adjusted tests of independence or no association between the row and column variables. These tests include the Rao-Scott chi-square test, the Rao-Scott likelihood ratio test, the Wald chi-square test, and the Wald log-linear chi-square test. For 2×2 tables, PROC SURVEYFREQ computes estimates and confidence limits for risks (row proportions), the risk difference, the odds ratio, and relative risks. For two-way tables, PROC SURVEYFREQ provides kappa statistics and other measures of agreement.

PROC SURVEYFREQ computes variance estimates based on the sample design used to obtain the survey data. The design can be a complex multistage survey design with stratification, clustering, and unequal weighting. PROC SURVEYFREQ provides a choice of variance estimation methods, which include Taylor series linearization, bootstrap, balanced repeated replication (BRR), and jackknife.

PROC SURVEYFREQ uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For more information about the statistical graphics that PROC SURVEYFREQ produces, see the PLOTS= option in the TABLES statement and the section "ODS Graphics" on page 9879.

Getting Started: SURVEYFREQ Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/surfgs.sas.)

The following example shows how to use PROC SURVEYFREQ to construct and analyze frequency tables from sample survey data. This example uses simulated data from a customer satisfaction survey for a student information system (SIS).

Suppose a company conducted a survey of school personnel who use the SIS, which is a software product that provides modules for student registration, class scheduling, attendance, grade reporting, and other functions. A probability sample of SIS users was selected from the study population, which included SIS users at middle schools and high schools in the states of Georgia, South Carolina, and North Carolina. The sample design for this survey was a two-stage stratified design. A first-stage sample of schools was selected from the list of schools in the three states that use the SIS. The list of schools, which are the primary sampling units (PSU), was stratified by state and by customer status (whether the school was a new user or a renewal user of the system). Within the strata, schools were selected with probability proportional to size and with replacement, where the size measure was school enrollment. From each sample school, five staff members were randomly selected with replacement as the second-stage sample to complete the SIS satisfaction questionnaire.

The SAS data set SIS_Survey contains the survey results and the sample design information that is needed to analyze the data. This data set contains the following variables:

- State: state where the school is located
- NewUser: 'New Customer' or 'Renewal Customer'
- School: school identification (PSU)
- SchoolType: 'High School' or 'Middle School'
- Department: 'Faculty' or 'Admin/Guidance'
- SamplingWeight: sampling weight
- Response: response, from "Very Unsatisfied" to "Very Satisfied"

The variables State and NewUser identify the first-stage strata from which schools were selected. The variable School identifies the primary sampling units (clusters). The variable SamplingWeight contains the sampling weight for each respondent. Sampling weights were computed from the selection probabilities at each stage of sampling and were adjusted for nonresponse.

The following PROC SURVEYFREQ statements request a one-way frequency table for the variable Response:

```
title 'Student Information System Survey';
proc surveyfreq data=SIS_Survey;
   tables  Response;
   strata  State NewUser;
   cluster School;
   weight  SamplingWeight;
```

```
run;
```

The PROC SURVEYFREQ statement invokes the procedure and identifies the input data set to be analyzed. The TABLES statement requests a one-way frequency table for the variable Response. The table request syntax in PROC SURVEYFREQ is identical to the table request syntax in PROC FREQ. You can specify one-way, two-way, and multiway table requests. You can specify more than one table request in the same TABLES statement, and you can specify multiple TABLES statements in the same invocation of the procedure.

The STRATA, CLUSTER, and WEIGHT statements provide sample design information for the procedure, which performs the analysis according to the survey design. The STRATA statement names the variables State and NewUser, which identify the first-stage strata. The CLUSTER statement names the variable School, which identifies the primary sampling units (clusters). The WEIGHT statement names the sampling weight variable.

Figure 118.1 and Figure 118.2 display the procedure output, which includes the "Data Summary" table and the one-way frequency table, "Table of Response." The "Data Summary" table is produced by default unless you specify the NOSUMMARY option. This table shows there are 6 strata, 370 clusters (schools), and 1,850 observations (respondents) in the SIS_Survey data set. The sum of the sampling weights is approximately 39,000, which can be used as an estimate of the population size (number of school personnel in the study area that use the SIS).

Figure 118.1 SIS_Survey Data Summary

Student Information System Survey

The SURVEYFREQ Procedure

Data Summary	
Number of Strata	6
Number of Clusters	370
Number of Observations	1850
Sum of Weights	38899.6482

Figure 118.2 displays the one-way table of Response, which provides estimates of the population total (weighted frequency) and the population percentage for each category (level) of the variable Response. The frequency of the response 'Very Unsatisfied' is 304, which means that 304 respondents in the sample reported this response. The weighted frequency of 'Very Unsatisfied' is 6,789, which is an estimate of the total frequency of this level in the study population. The standard error of this estimate is 501. The percentage of 'Very Unsatisfied' is 17.71%, which is an estimate of the percentage of this level in the study population. The standard error of this estimate is 1.20%. The standard errors that PROC SURVEYFREQ computes are based on the survey design; this differs from some traditional analysis procedures, which assume that the design is simple random sampling from an infinite population.

Figure 118.2 One-Way Table of Response

Table of Response					
Response	Frequency	Weighted Frequency	Std Err of Wgt Freq	Percent	Std Err of Percent
Very Unsatisfied	304	6678	501.61039	17.1676	1.2872
Unsatisfied	326	6907	495.94101	17.7564	1.2712
Neutral	581	12291	617.20147	31.5965	1.5795
Satisfied	455	9309	572.27868	23.9311	1.4761
Very Satisfied	184	3714	370.66577	9.5483	0.9523
Total	1850	38900	129.85268	100.0000	

The following PROC SURVEYFREQ statements request confidence limits for the weighted frequencies (totals), a chi-square goodness-of-fit test, and a weighted frequency plot for the one-way table of Response. The ODS GRAPHICS ON statement enables ODS Graphics, which is required in order to produce plots.

```
title 'Student Information System Survey';
ods graphics on;
proc surveyfreq data=SIS_Survey nosummary;
   tables  Response / clwt nopct chisq
                      plots=WtFreqPlot;
   strata  State  NewUser;
   cluster School;
   weight  SamplingWeight;
run;
ods graphics off;
```

The NOSUMMARY option in the PROC SURVEYFREQ statement suppresses the “Data Summary” table. The CLWT option in the TABLES statement requests confidence limits for the weighted frequencies (totals). The NOPCT option suppresses display of the percentages and their standard errors. The CHISQ option requests a Rao-Scott chi-square goodness-of-fit test, and the PLOTS= option requests a weighted frequency plot.

Figure 118.3 shows the one-way frequency table of Response, which displays weighted frequencies together with their standard errors and confidence limits. The 95% confidence limits for the total number of school personnel who are ‘Very Unsatisfied’ are 5,692 and 7,665. You can change the confidence level by specifying the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

Figure 118.3 Confidence Limits for Response Totals

Student Information System Survey

The SURVEYFREQ Procedure

Table of Response					
Response	Frequency	Weighted Frequency	Std Err of Wgt Freq	95% Confidence Limits for Wgt Freq	
Very Unsatisfied	304	6678	501.61039	5692	7665
Unsatisfied	326	6907	495.94101	5932	7882
Neutral	581	12291	617.20147	11077	13505
Satisfied	455	9309	572.27868	8184	10435
Very Satisfied	184	3714	370.66577	2985	4443
Total	1850	38900	129.85268	38644	39155

Figure 118.4 shows the weighted frequency plot of Response. This plot displays weighted frequencies (totals) together with their confidence limits in the form of a vertical bar chart. You can use the PLOTS= option to request a dot plot instead of a bar chart or to plot percentages instead of weighted frequencies.

Figure 118.4 Bar Chart of Response Totals

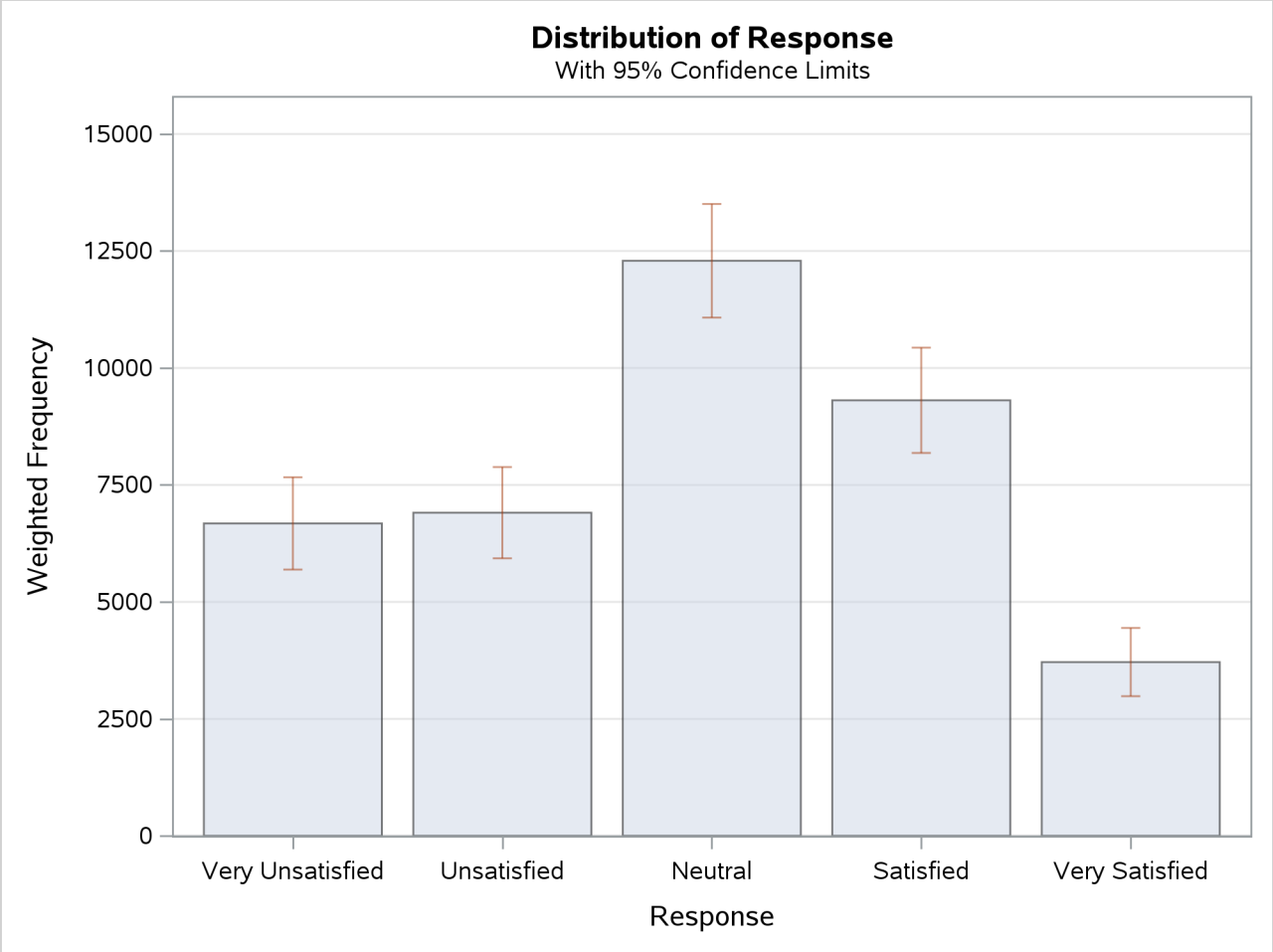

Figure 118.5 shows the chi-square goodness-of-fit test for the distribution of Response. The null hypothesis for this test is equal proportions for the levels of the one-way frequency table. (To test a null hypothesis of proportions that are not equal, you can use the TESTP= option to specify null hypothesis proportions.)

The CHISQ option produces the Rao-Scott design-adjusted chi-square test, which takes into account the sample design and provides inferences for the study population. To compute the Rao-Scott chi-square statistic, PROC SURVEYFREQ computes the Pearson chi-square statistic by using the weighted frequencies and adjusts this value by using a design correction. The procedure also provides an F approximation. For the table of Response, the F value is 30.0972 and the corresponding p-value is <0.0001, which indicates rejection of the null hypothesis of equal proportions.

Figure 118.5 Chi-Square Goodness-of-Fit Test for Response

Rao-Scott Chi-Square Test	
Pearson Chi-Square	251.8105
Design Correction	2.0916
Rao-Scott Chi-Square	120.3889
DF	4
Pr > ChiSq	<.0001
F Value	30.0972
Num DF	4
Den DF	1456
Pr > F	<.0001
Sample Size = 1850	

The following PROC SURVEYFREQ statements request a two-way crosstabulation of SchoolType by Response:

```
title 'Student Information System Survey';
ods graphics on;
proc surveyfreq data=SIS_Survey nosummary;
   tables  SchoolType * Response /
      plots=wtfreqplot(type=dot scale=percent groupby=row);
   strata  State NewUser;
   cluster School;
   weight  SamplingWeight;
run;
ods graphics off;
```

The STRATA, CLUSTER, and WEIGHT statements do not change from the one-way analysis because the sample design and the input data set are the same. The PROC SURVEYFREQ statements request a different table but specify the same sample design information.

The ODS GRAPHICS ON statement enables ODS Graphics. The PLOTS= option in the TABLES statement requests a plot of SchoolType by Response, and the TYPE=DOT *plot-option* specifies a dot plot instead of the default bar chart. The SCALE=PERCENT *plot-option* requests a plot of percentages instead of totals. The GROUPBY=ROW *plot-option* groups the graph cells by the row variable (SchoolType).

Figure 118.6 shows the two-way crosstabulation table for SchoolType by Response. The first variable in the two-way table request, SchoolType, is the *row variable*, and the second variable, Response, is the *column variable*. Two-way tables display all column variable levels for each row variable level. This two-way table lists all levels of the column variable Response for each level of the row variable SchoolType, 'Middle School' and 'High School'. SchoolType='Total' shows the distribution of Response overall for both types of schools. Response='Total' provides totals over all levels of response for each type of school and overall. To suppress these totals, you can specify the NOTOTAL option.

Figure 118.6 Two-Way Table of SchoolType by Response

Student Information System Survey

The SURVEYFREQ Procedure

Table of SchoolType by Response

SchoolType	Response	Frequency	Weighted Frequency	Std Err of Wgt Freq	Percent	Std Err of Percent
Middle School	Very Unsatisfied	116	2496	351.43834	6.4155	0.9030
	Unsatisfied	109	2389	321.97957	6.1427	0.8283
	Neutral	234	4856	504.20553	12.4847	1.2953
	Satisfied	197	4064	443.71188	10.4467	1.1417
	Very Satisfied	94	1952	302.17144	5.0193	0.7758
	Total	750	15758	1000	40.5089	2.5691
High School	Very Unsatisfied	188	4183	431.30589	10.7521	1.1076
	Unsatisfied	217	4518	446.31768	11.6137	1.1439
	Neutral	347	7434	574.17175	19.1119	1.4726
	Satisfied	258	5245	498.03221	13.4845	1.2823
	Very Satisfied	90	1762	255.67158	4.5290	0.6579
	Total	1100	23142	1003	59.4911	2.5691
Total	Very Unsatisfied	304	6678	501.61039	17.1676	1.2872
	Unsatisfied	326	6907	495.94101	17.7564	1.2712
	Neutral	581	12291	617.20147	31.5965	1.5795
	Satisfied	455	9309	572.27868	23.9311	1.4761
	Very Satisfied	184	3714	370.66577	9.5483	0.9523
	Total	1850	38900	129.85268	100.0000	

Figure 118.7 displays the weighted frequency dot plot that PROC SURVEYFREQ produces for the table of SchoolType and Response. The GROUPBY=ROW *plot-option* groups the graph cells by the row variable (SchoolType). If you do not specify GROUPBY=ROW, the procedure groups the graph cells by the column variable by default. You can plot percentages instead of weighted frequencies by specifying the SCALE=PERCENT *plot-option*. You can use other *plot-options* to change the orientation of the plot or to request a different two-way layout.

Figure 118.7 Dot Plot of Percentages for SchoolType by Response

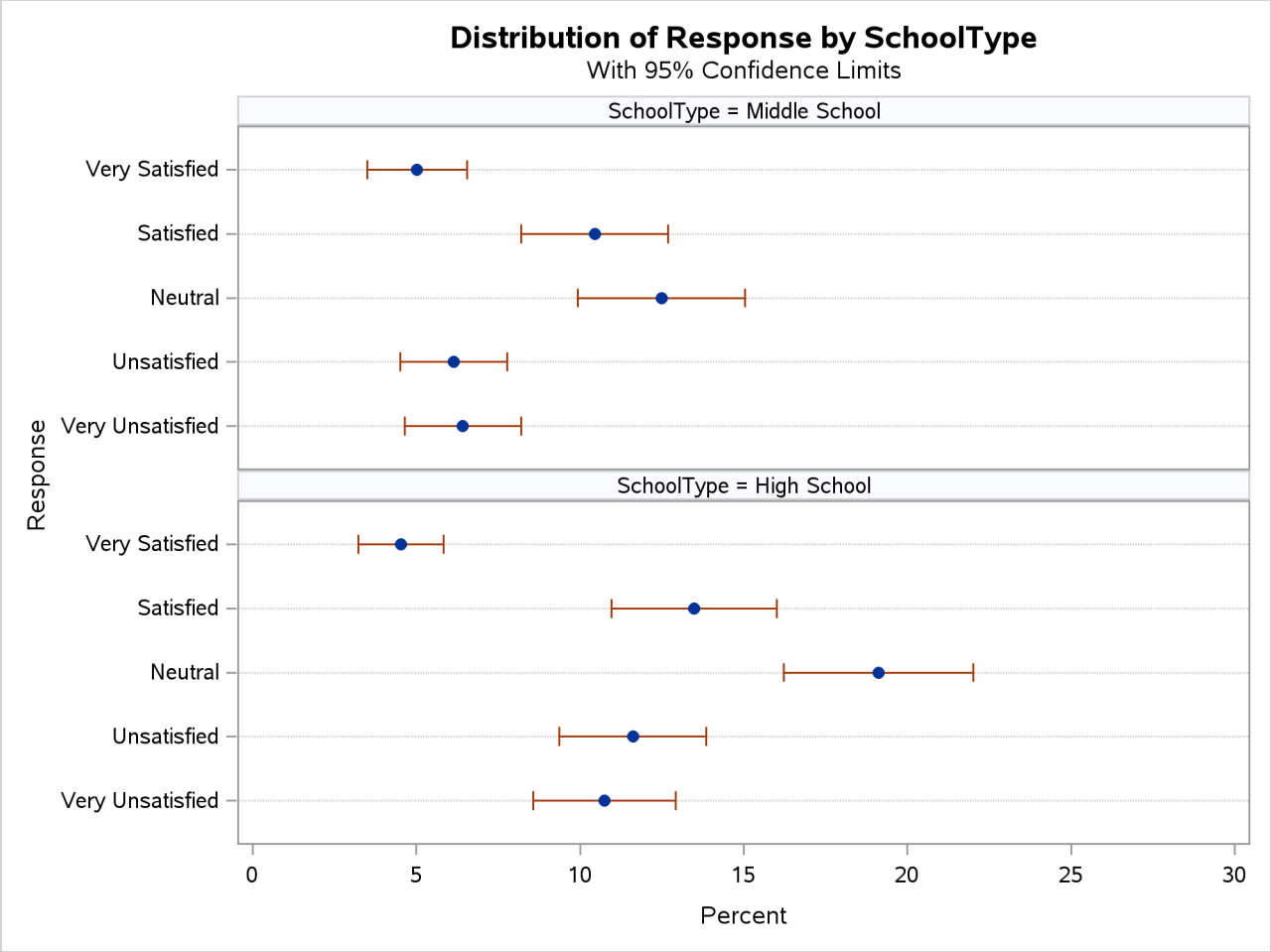

The following PROC SURVEYFREQ statements request a two-way table of SchoolType by Response that displays row percentages and a chi-square test of association between the two variables:

```
title 'Student Information System Survey';
proc surveyfreq data=SIS_Survey nosummary;
   tables  SchoolType * Response / row nowt chisq;
   strata  State NewUser;
   cluster School;
   weight  SamplingWeight;
run;
```

The ROW option in the TABLES statement requests row percentages, which provide the distribution of Response within each level of the row variable SchoolType. The NOWT option suppresses display of the weighted frequencies and their standard errors. The CHISQ option requests a Rao-Scott chi-square test of association between SchoolType and Response.

Figure 118.8 displays the two-way table of SchoolType by Response. For middle schools, it is estimated that 25.79% of school personnel are satisfied with the student information system and 12.39% are very satisfied. For high schools, these estimates are 22.67% and 7.61%, respectively.

Figure 118.9 displays the chi-square test results. The Rao-Scott chi-square statistic is 9.04, and the corre-

sponding F value is 2.26 with a p-value of 0.0605. This indicates an association between school type (middle school or high school) and satisfaction with the student information system at the 10% significance level.

Figure 118.8 Two-Way Table with Row Percentages

Student Information System Survey

The SURVEYFREQ Procedure

Table of SchoolType by Response

SchoolType	Response	Frequency	Percent	Std Err of Percent	Row Percent	Std Err of Row Percent
Middle School	Very Unsatisfied	116	6.4155	0.9030	15.8373	1.9920
	Unsatisfied	109	6.1427	0.8283	15.1638	1.8140
	Neutral	234	12.4847	1.2953	30.8196	2.5173
	Satisfied	197	10.4467	1.1417	25.7886	2.2947
	Very Satisfied	94	5.0193	0.7758	12.3907	1.7449
	Total	750	40.5089	2.5691	100.0000	
High School	Very Unsatisfied	188	10.7521	1.1076	18.0735	1.6881
	Unsatisfied	217	11.6137	1.1439	19.5218	1.7280
	Neutral	347	19.1119	1.4726	32.1255	2.0490
	Satisfied	258	13.4845	1.2823	22.6663	1.9240
	Very Satisfied	90	4.5290	0.6579	7.6128	1.0557
	Total	1100	59.4911	2.5691	100.0000	
Total	Very Unsatisfied	304	17.1676	1.2872		
	Unsatisfied	326	17.7564	1.2712		
	Neutral	581	31.5965	1.5795		
	Satisfied	455	23.9311	1.4761		
	Very Satisfied	184	9.5483	0.9523		
	Total	1850	100.0000			

Figure 118.9 Chi-Square Test of No Association

Rao-Scott Chi-Square Test	
Pearson Chi-Square	18.7829
Design Correction	2.0766
Rao-Scott Chi-Square	9.0450
DF	4
Pr > ChiSq	0.0600
F Value	2.2613
Num DF	4
Den DF	1456
Pr > F	0.0605
Sample Size = 1850	

Syntax: SURVEYFREQ Procedure

The following statements are available in the SURVEYFREQ procedure:

PROC SURVEYFREQ < *options* > ;
BY *variables* ;
CLUSTER *variables* ;
REPWEIGHTS *variables* < / *options* > ;
STRATA *variables* < / *option* > ;
TABLES *requests* < / *options* > ;
WEIGHT *variable* ;

The PROC SURVEYFREQ statement invokes the procedure, identifies the data set to be analyzed, and specifies the variance estimation method to use. The PROC SURVEYFREQ statement is required.

The TABLES statement specifies frequency or crosstabulation tables and requests tests and statistics for those tables. The STRATA statement lists the variables that form the strata in a stratified sample design. The CLUSTER statement specifies cluster identification variables in a clustered sample design. The WEIGHT statement names the sampling weight variable. The REPWEIGHTS statement names replicate weight variables for replication variance estimation. The BY statement requests completely separate analyses of groups defined by the BY variables.

All statements can appear multiple times except the PROC SURVEYFREQ statement and the WEIGHT statement, which can appear only once.

The rest of this section gives detailed syntax information for the BY, CLUSTER, REPWEIGHTS, STRATA, TABLES, and WEIGHT statements in alphabetical order after the description of the PROC SURVEYFREQ statement.

PROC SURVEYFREQ Statement

PROC SURVEYFREQ < *options* > ;

The PROC SURVEYFREQ statement invokes the SURVEYFREQ procedure. It also identifies the data set to be analyzed, specifies the variance estimation method to use, and provides sample design information. The DATA= option names the input data set to be analyzed. The VARMETHOD= option specifies the variance estimation method, which is the Taylor series method by default. For Taylor series and bootstrap variance estimation, you can include a finite population correction factor in the analysis by providing either the sampling rate or population total in the RATE= or TOTAL= option, respectively. If your design is stratified with different sampling rates or totals for different strata, you can input these stratum rates or totals in a SAS data set that contains the stratification variables.

Table 118.1 summarizes the *options* available in the PROC SURVEYFREQ statement.

Table 118.1 PROC SURVEYFREQ Statement Options

Option	Description
DATA=	Names the input SAS data set
DEFF	Specifies design effect details

Table 118.1 *continued*

Option	Description
MISSING	Treats missing values as a valid level
NOMCAR	Treats missing values as not missing completely at random
NOSUMMARY	Suppresses the display of the "Data Summary" table
ORDER=	Specifies the order of variable levels
PAGE	Displays only one table per page
RATE=	Specifies the first-stage sampling rate
TOTAL=	Specifies the total number of primary sampling units
VARHEADER=	Specifies the variable identification to display
VARMETHOD=	Specifies the variance estimation method

You can specify the following *options*:

DATA=*SAS-data-set*
names the *SAS-data-set* to be analyzed by PROC SURVEYFREQ. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

DEFF(*options***)**
specifies details of the design effect computations.

This option applies to all design effects in the analyses that you request in the TABLES statements. The DEFF option in the TABLES statement displays design effects of percentages in the frequency and crosstabulation tables. PROC SURVEYFREQ also uses design effects to compute Rao-Scott chi-square tests (CHISQ) and modified confidence limits for proportions (CL). For more information, see the sections "Rao-Scott Chi-Square Test" on page 9858 and "Modified Confidence Limits" on page 9843.

You can specify the following *options*:

FPC=YES | NO
controls the inclusion or exclusion of the finite population correction (*fpc*) in the simple random sampling (SRS) variance component of the design effect. For more information, see the section "Design Effect" on page 9847.

This option is available when you specify a finite population correction by providing sampling rates or population totals in the RATE= or TOTAL= option, respectively. For more information, see the section "Population Totals and Sampling Rates" on page 9823.

FPC=YES includes the *fpc* and FPC=NO excludes the *fpc* from the SRS variance component of the design effect. By default, FPC=YES when you use Taylor series variance estimation and FPC=NO when you use replication variance estimation.

VARDEF=N
uses the number of observations (n) as the divisor in the simple random sampling (SRS) variance component of the design effect. By default, PROC SURVEYFREQ uses ($n - 1$) as the divisor. For more information, see the section "Design Effect" on page 9847.

MISSING

treats missing values as a valid (nonmissing) category for all categorical variables, which include TABLES, STRATA, and CLUSTER variables.

By default (if you do not specify the MISSING option), PROC SURVEYFREQ excludes an observation from the analysis if the observation contains a missing value for any STRATA or CLUSTER variable. By default, PROC SURVEYFREQ also excludes an observation from the analysis if the observation contains a missing value for any variable in the table request. For more information, see the section "Missing Values" on page 9825.

NOMCAR

includes observations with missing values of TABLES variables in the variance computation as *not missing completely at random* (NOMCAR) for Taylor series variance estimation. When you specify the NOMCAR option, PROC SURVEYFREQ computes variance estimates by analyzing the nonmissing values as a domain (subpopulation), where the entire population includes both nonmissing and missing domains. For more information, see the section "Missing Values" on page 9825.

By default, PROC SURVEYFREQ completely excludes an observation from a frequency or crosstabulation table (and the corresponding variance computations) if that observation has a missing value for any of the variables in the table request, unless you specify the MISSING option. The NOMCAR option has no effect when you specify the MISSING option, which treats missing values as a valid nonmissing level.

The NOMCAR option applies only to Taylor series variance estimation; it does not apply to replication variance estimation.

NOSUMMARY

suppresses the display of the "Data Summary" table, which PROC SURVEYFREQ produces by default. For information about this table, see the section "Data Summary Table" on page 9869.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the order of the variable levels in the frequency and crosstabulation tables, which you request in the TABLES statement. The ORDER= option also controls the order of the STRATA variable levels in the "Stratum Information" table.

You can specify one of the following orders:

ORDER=	Levels Ordered By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=INTERNAL. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. ORDER=FREQ orders levels by unweighted frequency (number of observations) rather than weighted frequency.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

PAGE

displays only one table per page. Otherwise, PROC SURVEYFREQ displays multiple tables per page as space permits.

RATE=*value* | *SAS-data-set*

R=*value* | *SAS-data-set*

specifies the sampling rate, which PROC SURVEYFREQ uses to compute a finite population correction in Taylor series and bootstrap variance estimation. This option is not used in BRR, jackknife, and replicate weight variance estimation.

If your sample design has multiple stages, you should specify the *first-stage sampling rate*, which is the ratio of the number of primary sampling units (PSUs) in the sample to the total number of PSUs in the population.

You can specify one of the following forms:

RATE=*value*

specifies a single sampling rate *value* to use for all strata or for a nonstratified design.

RATE=*SAS-data-set*

names a *SAS-data-set* that contains the stratum sampling rates. You must provide the sampling rates in a data set variable named _RATE_.

The *SAS-data-set* must contain all stratification variables that you specify in the STRATA statement. It must also contain all stratum levels that appear in the DATA= input data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets.

You can specify sampling rates as numbers between 0 and 1. Or you can specify sampling rates in percentage form as numbers between 1 and 100, which PROC SURVEYFREQ converts to proportions. The procedure treats the value 1 as 100% instead of 1%.

For more information, see the section "Population Totals and Sampling Rates" on page 9823.

If you do not specify either the RATE= or TOTAL= option, the variance estimation does not include a finite population correction. You cannot specify both the RATE= and TOTAL= options.

TOTAL=*value* | *SAS-data-set*

N=*value* | *SAS-data-set*

specifies the total number of primary sampling units (PSUs) in the population. PROC SURVEYFREQ uses this information to compute a finite population correction in Taylor series and bootstrap variance estimation. This option is not used in BRR, jackknife, and replicate weight variance estimation.

You can specify one of the following forms:

TOTAL=*value*
specifies a single total *value* to use for all strata or for a nonstratified design. The *value* must be a positive number.

TOTAL=*SAS-data-set*
names a *SAS-data-set* that contains the stratum totals. You must provide the stratum totals in a data set variable named _TOTAL_. The totals must be positive numbers.

The *SAS-data-set* must contain all stratification variables that you specify in the STRATA statement. It must also contain all stratum levels that appear in the DATA= input data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets.

For more information, see the section "Population Totals and Sampling Rates" on page 9823.

If you do not specify either the TOTAL= or RATE= option, the variance estimation does not include a finite population correction. You cannot specify both the TOTAL= and RATE= options.

VARHEADER=LABEL | NAME | NAMELABEL
specifies the variable identification to use in the displayed output. This option controls the headings of the variable columns in one-way frequency tables, crosstabulation tables, and the "Stratum Information" table. This option also controls the variable identification in table titles. By default, VARHEADER=NAME.

The following table shows how PROC SURVEYFREQ interprets values of the VARHEADER= option.

VARHEADER=	Variable Identification Displayed
LABEL	Variable label
NAME	Variable name
NAMELABEL	Variable name and label, as *Name* (*Label*)

VARMETHOD=*method* **<(***method-options***)>**
specifies the variance estimation *method*. PROC SURVEYFREQ provides the Taylor series method and the following replication (resampling) methods: bootstrap, balanced repeated replication (BRR), and jackknife.

Table 118.2 summarizes the available *methods* and *method-options*.

Table 118.2 Variance Estimation Methods

method	**Variance Estimation Method**	*method-options*
BOOTSTRAP	Bootstrap	CENTER=FULLSAMPLE | REPLICATES DFADJ MH=*values* | *SAS-data-set* OUTWEIGHTS=*SAS-data-set* REPS=*number* REPWARN SEED=*number*
BRR	Balanced repeated replication	CENTER=FULLSAMPLE | REPLICATES DFADJ FAY <=*value*> HADAMARD=*SAS-data-set* OUTWEIGHTS=*SAS-data-set* PRINTH REPS=*number* REPWARN
JACKKNIFE | JK	Jackknife	CENTER=FULLSAMPLE | REPLICATES DFADJ OUTJKCOEFS=*SAS-data-set* OUTWEIGHTS=*SAS-data-set* REPWARN
TAYLOR	Taylor series linearization	None

For VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, and VARMETHOD=JACKKNIFE, you can specify *method-options* in parentheses after the variance *method*. For example:

```
varmethod=BRR(reps=60 outweights=myReplicateWeights)
```

By default, VARMETHOD=JACKKNIFE if you specify a REPWEIGHTS statement; otherwise, VARMETHOD=TAYLOR by default.

You can specify one of the following *methods*:

BOOTSTRAP <(*method-options*)>

requests variance estimation by the bootstrap method. For more information, see the section "Bootstrap Method" on page 9828.

The bootstrap method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you use a REPWEIGHTS statement to provide replicate weights.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

specifies how to compute the deviations for bootstrap variance estimation. You can specify one of the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Bootstrap Method" on page 9828.

DFADJ

computes the degrees of freedom by using the number of nonmissing strata and clusters for the current table request. If you specify this *method-option*, PROC SURVEYFREQ does not count any empty strata or clusters that occur when observations are excluded because of missing values of a TABLES variable in the current request.

By default, PROC SURVEYFREQ computes the degrees of freedom by counting the number of nonmissing strata and clusters for all valid observations in the input data set. The degrees of freedom for VARMETHOD=BOOTSTRAP equals the number of clusters minus the number of strata.

For more information, see the section "Degrees of Freedom" on page 9846. For information about valid observations, see the section "Data Summary Table" on page 9869.

This *method-option* has no effect when you specify the MISSING option, which treats missing values as a valid nonmissing level.

This *method-option* is not used when you specify the degrees of freedom in the DF= option in the TABLES statement or when you specify a REPWEIGHTS statement to provide replicate weights. When you specify a REPWEIGHTS statement, the degrees of freedom is the number of REPWEIGHTS variables (replicates) unless you specify the DF= option in the REPWEIGHTS or the TABLES statement.

MH=*value* | **(***values***)** | *SAS-data-set*

specifies the number of PSUs to select for the bootstrap replicate samples. You can provide bootstrap stratum sample sizes m_h by specifying a list of *values* or a *SAS-data-set*. Alternatively, you can provide a single bootstrap sample size *value* to use for all strata or for a nonstratified design. You can specify the number of replicate samples in the REPS= option. For more information, see the section "Bootstrap Method" on page 9828.

Each bootstrap sample size m_h must be a positive integer and must be less than n_h, which is the total number of PSUs in stratum h. By default, $m_h = n_h - 1$ for a stratified design. For a nonstratified design, the bootstrap sample size *value* must be less than n (the total number of PSUs in the sample). By default, $m = n - 1$ for a nonstratified design.

You can provide bootstrap sample sizes by specifying one of the following forms:

MH=*value*
specifies a single bootstrap sample size *value* to use for all strata or for a nonstratified design.

MH=(*values***)**
specifies a list of stratum bootstrap sample size *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. The number of values must not be less than the number of strata in the DATA= input data set.

The order of the stratum sample size values must match the order of the stratum levels in the DATA= input data set. Each stratum sample size value must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

MH=*SAS-data-set*
names a *SAS-data-set* that contains the stratum bootstrap sample sizes. You must provide the sample sizes in a data set variable named _NSIZE_ or SampleSize.

The *SAS-data-set* must contain all stratification variables that you specify in the STRATA statement. It must also contain all stratum levels that appear in the DATA= input data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets.

Each value of the _NSIZE_ or SampleSize variable must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

OUTWEIGHTS=*SAS-data-set*
names a *SAS-data-set* to store the bootstrap replicate weights that PROC SURVEYFREQ creates. For information about replicate weights, see the section "Bootstrap Method" on page 9828. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weight Output Data Set" on page 9868.

This *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

REPS=*number*
specifies the *number* of replicates for bootstrap variance estimation. The value of *number* must be an integer greater than 1. Increasing the number of replicates improves the estimation precision but also increases the computation time. By default, REPS=250.

REPWARN < =NO | YES >
controls the log message that PROC SURVEYFREQ provides when a parameter cannot be estimated from one or more replicates. REPWARN or REPWARN=YES displays this log message as a warning. By default, REPWARN=NO, which displays the log message as a note.

For more information, see the subsection "Variance Estimation" in the section "Bootstrap Method" on page 9828.

SEED=*number*

specifies the initial seed for random number generation for bootstrap replicate sampling.

If you do not specify this option or if you specify a *number* that is negative or 0, PROC SURVEYFREQ uses the time of day from the system clock to obtain an initial seed.

To reproduce the same bootstrap replicate weights and the same analysis in a subsequent execution of PROC SURVEYFREQ, you can specify the same initial seed that was used in the original analysis.

PROC SURVEYFREQ displays the value of the initial seed in the "Variance Estimation" table.

BRR < (*method-options*) >

requests variance estimation by balanced repeated replication (BRR). This method requires a stratified sample design where each stratum contains two primary sampling units (PSUs). When you specify this method, you must also specify a STRATA statement unless you provide replicate weights by using the REPWEIGHTS statement. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 9830.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

specifies how to compute the deviations for BRR variance estimation. You can specify one of the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 9830.

DFADJ

computes the degrees of freedom as the number of nonmissing strata for the current table request. If you specify this *method-option*, PROC SURVEYFREQ does not count any empty strata that occur when observations are excluded because of missing values of a TABLES variable in the current request.

By default, PROC SURVEYFREQ computes the degrees of freedom by counting the number of nonmissing strata for all valid observations in the input data set.

For more information, see the section "Degrees of Freedom" on page 9846. For information about valid observations, see the section "Data Summary Table" on page 9869.

This *method-option* has no effect when you specify the MISSING option, which treats missing values as a valid nonmissing level.

This *method-option* is not used when you specify the degrees of freedom in the DF= option in the TABLES statement or when you specify a REPWEIGHTS statement to provide replicate weights. When you specify a REPWEIGHTS statement, the degrees of freedom is the number of REPWEIGHTS variables (replicates) unless you specify the DF= option in the REPWEIGHTS or the TABLES statement.

FAY < =*value* >

requests Fay's method, which is a modification of the BRR method. For more information, see the section "Fay's BRR Method" on page 9832.

You can specify the *value* of the Fay coefficient, which is used in converting the original sampling weights to replicate weights. The Fay coefficient must be a nonnegative number less than 1. By default, the Fay coefficient is 0.5.

HADAMARD=*SAS-data-set*

H=*SAS-data-set*

names a *SAS-data-set* that contains the Hadamard matrix for BRR replicate construction. If you do not specify this *method-option*, PROC SURVEYFREQ generates an appropriate Hadamard matrix for replicate construction. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 9830 and "Hadamard Matrix" on page 9832.

If a Hadamard matrix of a particular dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a *SAS-data-set* in this *method-option*.

In the HADAMARD= input data set, each variable corresponds to a column and each observation corresponds to a row of the Hadamard matrix. You can use any variable names in the HADAMARD= data set. All values in the data set must equal either 1 or –1. You must ensure that the matrix you provide is indeed a Hadamard matrix—that is, $\mathbf{A}'\mathbf{A} = R\,\mathbf{I}$, where $\mathbf{A}$ is the Hadamard matrix of dimension R and $\mathbf{I}$ is an identity matrix. PROC SURVEYFREQ does not check the validity of the Hadamard matrix that you provide.

The HADAMARD= input data set must contain at least H variables, where H denotes the number of first-stage strata in your design. If the data set contains more than H variables, PROC SURVEYFREQ uses only the first H variables. Similarly, the HADAMARD= input data set must contain at least H observations.

If you do not specify the REPS= *method-option*, the number of replicates is assumed to be the number of observations in the HADAMARD= input data set. If you specify the number of replicates—for example, REPS=*nreps*—the first *nreps* observations in the HADAMARD= data set are used to construct the replicates.

You can specify the PRINTH *method-option* to display the Hadamard matrix that PROC SURVEYFREQ uses to construct replicates for BRR.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* to store the replicate weights that PROC SURVEYFREQ creates for BRR variance estimation. For information about replicate weights, see the section "Balanced Repeated Replication (BRR) Method" on page 9830. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weight Output Data Set" on page 9868.

The OUTWEIGHTS= *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

PRINTH

displays the Hadamard matrix that PROC SURVEYFREQ uses to construct replicates for BRR variance estimation. When you provide the Hadamard matrix in the HADAMARD= *method-option*, PROC SURVEYFREQ displays only the rows and columns that are actually used to construct replicates. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 9830 and "Hadamard Matrix" on page 9832.

The PRINTH *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement because the procedure does not use a Hadamard matrix in this case.

REPS=*number*

specifies the *number* of replicates for BRR variance estimation. The value of *number* must be an integer greater than 1.

If you do not use the HADAMARD= *method-option* to provide a Hadamard matrix, the number of replicates should be greater than the number of strata and should be a multiple of 4. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 9830. If PROC SURVEYFREQ cannot construct a Hadamard matrix for the REPS= value that you specify, the value is increased until a Hadamard matrix of that dimension can be constructed. Therefore, the actual number of replicates that PROC SURVEYFREQ uses might be larger than *number*.

If you use the HADAMARD= *method-option* to provide a Hadamard matrix, the value of *number* must not be greater than the number of rows in the Hadamard matrix. If you provide a Hadamard matrix and do not specify the REPS= *method-option*, the number of replicates is the number of rows in the Hadamard matrix.

If you do not specify the REPS= or the HADAMARD= *method-option* and do not use a REPWEIGHTS statement, the number of replicates is the smallest multiple of 4 that is greater than the number of strata.

If you use a REPWEIGHTS statement to provide replicate weights, PROC SURVEYFREQ does not use the REPS= *method-option*; the number of replicates is the number of REPWEIGHTS variables.

REPWARN < =NO | YES >

controls the log message that PROC SURVEYFREQ provides when a parameter cannot be estimated from one or more replicates. REPWARN or REPWARN=YES displays this log message as a warning. By default, REPWARN=NO, which displays the log message as a note.

For more information, see the subsection "Variance Estimation" in the section "Balanced Repeated Replication (BRR) Method" on page 9830.

JACKKNIFE < (*method-options***) >**

JK < (*method-options***) >**

requests variance estimation by the delete-1 jackknife method. For more information, see the section "Jackknife Method" on page 9834. If you use a REPWEIGHTS statement to provide replicate weights, VARMETHOD=JACKKNIFE is the default variance estimation method.

The delete-1 jackknife method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you use a REPWEIGHTS statement to provide replicate weights.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

specifies how to compute the deviations for jackknife variance estimation. You can specify one of the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Jackknife Method" on page 9834.

DFADJ

computes the degrees of freedom by using the number of nonmissing strata and clusters for the current table request. If you specify this *method-option*, PROC SURVEYFREQ does not count any empty strata or clusters that occur when observations are excluded because of missing values of a TABLES variable in the current request.

By default, PROC SURVEYFREQ computes the degrees of freedom by counting the number of nonmissing strata and clusters for all valid observations in the input data set. The degrees of freedom for VARMETHOD=JACKKNIFE equals the number of clusters minus the number of strata.

For more information, see the section "Degrees of Freedom" on page 9846. For information about valid observations, see the section "Data Summary Table" on page 9869.

This *method-option* has no effect when you specify the MISSING option, which treats missing values as a valid nonmissing level.

This *method-option* is not used when you specify the degrees of freedom in the DF= option in the TABLES statement or when you specify a REPWEIGHTS statement to provide replicate weights. When you specify a REPWEIGHTS statement, the degrees of freedom is the number of REPWEIGHTS variables (replicates) unless you specify the DF= option in the REPWEIGHTS or the TABLES statement.

OUTJKCOEFS=*SAS-data-set*

names a *SAS-data-set* to store the jackknife coefficients. For information about jackknife coefficients, see the section "Jackknife Method" on page 9834. For information about the contents of the OUTJKCOEFS= data set, see the section "Jackknife Coefficient Output Data Set" on page 9868.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* to store the replicate weights that PROC SURVEYFREQ creates for jackknife variance estimation. For information about replicate weights, see the section "Jackknife Method" on page 9834. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weight Output Data Set" on page 9868.

This *method-option* is not available when you use a REPWEIGHTS statement to provide replicate weights.

REPWARN < =NO | YES >

controls the log message that PROC SURVEYFREQ provides when a parameter cannot be estimated from one or more replicates. REPWARN or REPWARN=YES displays this log message as a warning. By default, REPWARN=NO, which displays the log message as a note.

For more information, see the subsection "Variance Estimation" in the section "Jackknife Method" on page 9834.

TAYLOR

requests Taylor series variance estimation. This is the default method if you do not specify the VARMETHOD= option or a REPWEIGHTS statement. For more information, see the section "Taylor Series Variance Estimation" on page 9827.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SURVEYFREQ to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SURVEYFREQ procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Using a BY statement provides completely separate analyses of the BY groups. It does not provide a statistically valid domain (subpopulation) analysis, where the total number of units in the subpopulation is not known with certainty. You should include the domain variable(s) in your TABLES request to obtain domain analysis. For more information, see the section "Domain Analysis" on page 9824.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLUSTER Statement

CLUSTER *variables* **;**

The CLUSTER statement names one or more *variables* that identify the first-stage clusters in a clustered sample design. First-stage clusters are also known as primary sampling units (PSUs). The combinations of levels of the CLUSTER variables define the clusters in the sample. If there is a STRATA statement, clusters are nested within strata.

If your sample design has clustering at multiple stages, you should specify only the first-stage clusters (PSUs) in the CLUSTER statement. For more information, see the section "Specifying the Sample Design" on page 9822.

If you provide replicate weights for replication variance estimation by using the REPWEIGHTS statement, you do not need to specify a CLUSTER statement.

The CLUSTER variables are one or more variables in the DATA= input data set. These variables can be either character or numeric, but the procedure treats them as categorical variables. The formatted values of the CLUSTER variables determine the CLUSTER variable levels. Thus, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

An observation is excluded from the analysis if it has a missing value for any CLUSTER variable unless you specify the MISSING option in the PROC SURVEYFREQ statement. For more information, see the section "Missing Values" on page 9825.

You can use multiple CLUSTER statements to specify CLUSTER variables. The procedure uses variables from all CLUSTER statements to create clusters.

REPWEIGHTS Statement

REPWEIGHTS *variables* < / *options* > **;**

The REPWEIGHTS statement names *variables* that provide replicate weights for replication variance estimation methods (which include bootstrap, BRR, and jackknife). Each *variable* should contain the weights for a single replicate, and the number of *variables* should correspond to the number of replicates. The *variables* must be numeric, and their values must be nonnegative unless you specify the NEGATIVE option.

You can provide replicate coefficients by specifying the JKCOEFS= or REPCOEFS= option. These options are available for bootstrap and jackknife variance estimation (VARMETHOD=BOOTSTRAP and VARMETHOD=JACKKNIFE), in addition to general replication variance estimation (when the VARMETHOD= option is not specified). When you specify VARMETHOD=BRR, the replicate coefficient is fixed at $1/R$, where R is the total number of replicates.

For information about replicate weights, replicate coefficients, and replication variance estimation, see the sections "Bootstrap Method" on page 9828, "Balanced Repeated Replication (BRR) Method" on page 9830, "Fay's BRR Method" on page 9832, and "Jackknife Method" on page 9834. When you provide replicate weights but do not specify the VARMETHOD= option, PROC SURVEYFREQ computes the variance estimates as described in the section "Jackknife Method" on page 9834.

When you provide replicate weights, you do not need to specify a CLUSTER or STRATA statement.

If you specify a REPWEIGHTS statement but do not include a WEIGHT statement, PROC SURVEYFREQ uses the average of each observation's replicate weights as the observation's weight.

You can specify the following *options*:

DF=*df*

specifies the degrees of freedom for the analysis. The value of *df* must be a positive number. By default, the value of *df* is the number of REPWEIGHTS variables. For more information, see the section "Degrees of Freedom" on page 9846.

PROC SURVEYFREQ uses *df* to obtain the *t* distribution percentile for confidence limits (for proportions, totals, and other statistics). For more information, see the section "Confidence Limits for Proportions" on page 9842. PROC SURVEYFREQ also uses *df* to compute the denominator degrees of freedom for the *F* statistics in the Rao-Scott and Wald chi-square tests. For more information, see the sections "Rao-Scott Chi-Square Test" on page 9858, "Rao-Scott Likelihood Ratio Chi-Square Test" on page 9864, "Wald Chi-Square Test" on page 9865, and "Wald Log-Linear Chi-Square Test" on page 9867.

JKCOEFS=*value* | **<(>***values* **<)>** | *SAS-data-set*

specifies the jackknife coefficients for jackknife variance estimation (VARMETHOD=JACKKNIFE). You can provide a single jackknife coefficient *value* to use for all replicates, or you can provide a separate value for each replicate by specifying a list of *values* or a *SAS-data-set*. The jackknife coefficient values must be nonnegative numbers. For more information, see the section "Jackknife Method" on page 9834.

You can provide jackknife coefficients by specifying one of the following forms:

JKCOEFS=*value*

specifies a single jackknife coefficient *value* to use for all replicates, where *value* must be a nonnegative number.

JKCOEFS=(*values***)**

specifies a list of jackknife coefficient *values*, where each value corresponds to a single replicate that is identified by a REPWEIGHTS variable. You can separate the values with blanks or commas, and you can enclose the list of values in parentheses. The coefficient values must be nonnegative numbers. The number of coefficient values must equal the number of replicate weight variables that you specify in the REPWEIGHTS statement.

List the jackknife coefficient values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

JKCOEFS=*SAS-data-set*

names a *SAS-data-set* that contains the jackknife coefficients. You must provide the jackknife coefficients in a variable named JKCoefficient in the *SAS-data-set*. Each coefficient value must be a nonnegative number. Each observation in the *SAS-data-set* is assumed to correspond to a replicate that is identified by a REPWEIGHTS variable. The number of observations in the *SAS-data-set* must not be less than the number of REPWEIGHTS variables.

The order of the jackknife coefficients in the *SAS-data-set* should match the order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

By default, the jackknife coefficient is $(R-1)/R$, where R is the total number of replicates.

NEGATIVE | NEG

removes the restriction that the REPWEIGHTS variable values must be nonnegative. When you specify this option, PROC SURVEYFREQ accepts negative replicate weight values and includes them in the analysis.

By default, replicate weight values must be nonnegative; when PROC SURVEYFREQ encounters a negative replicate weight value, it displays an error message and does not perform the analysis unless you specify the NEGATIVE option.

REPCOEFS=*value* | **<(>***values* **<)>** | *SAS-data-set*

specifies the replicate coefficients for replicate-based variance estimation. You can provide a single coefficient *value* to use for all replicates, or you can provide a separate value for each replicate by specifying a list of *values* or a *SAS-data-set*. The replicate coefficient values must be nonnegative numbers.

This option is available for bootstrap and jackknife variance estimation (VARMETHOD=BOOTSTRAP and VARMETHOD=JACKKNIFE, respectively) in addition to general replication variance estimation (when the VARMETHOD= option is not specified).

When you specify VARMETHOD=BOOTSTRAP, the default value of the replicate coefficient is $1/R$, where R is the total number of replicates. When you specify VARMETHOD=JACKKNIFE, or when you do not specify the VARMETHOD= option, the default value of the replicate coefficient is $(R-1)/R$. For more information, see the sections "Bootstrap Method" on page 9828 and "Jackknife Method" on page 9834.

You can provide replicate coefficients by specifying one of the following forms:

REPCOEFS=*value*

specifies a single coefficient *value* to use for all replicates, where *value* must be a nonnegative number.

REPCOEFS=(*values***)**

specifies a list of coefficient *values*, where each *value* corresponds to a single replicate that is identified by a REPWEIGHTS *variable*. You can separate the values with blanks or commas, and you can enclose the list of values in parentheses. The coefficient values must be nonnegative numbers. The number of coefficient values must equal the number of replicate weight variables that you specify in the REPWEIGHTS statement.

List the replicate coefficient values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

REPCOEFS=*SAS-data-set*

names a *SAS-data-set* that contains the replicate coefficients. You must provide the coefficients in a variable named RepCoefficient in the *SAS-data-set*. Each coefficient value must be a nonnegative number. Each observation in the *SAS-data-set* is assumed to correspond to a replicate that is identified by a REPWEIGHTS variable. The number of observations in the *SAS-data-set* must not be less than the number of REPWEIGHTS variables.

The order of the replicate coefficients in the *SAS-data-set* should match the order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

REPWARN < =NO | YES >

controls the log message that PROC SURVEYFREQ provides when a parameter cannot be estimated from one or more replicates. REPWARN or REPWARN=YES displays this log message as a warning. By default, REPWARN=NO, which displays the log message as a note.

For more information, see the subsection "Variance Estimation" in the section "Jackknife Method" on page 9834.

STRATA Statement

STRATA *variables* < / *option* > **;**

The STRATA statement names one or more *variables* that identify the first-stage strata in a stratified sample design. The combinations of levels of STRATA variables define the strata in the sample, where strata are nonoverlapping subgroups that were sampled independently.

If your sample design has stratification at multiple stages, you should specify only the first-stage strata in the STRATA statement. For more information, see the section "Specifying the Sample Design" on page 9822.

If you use a REPWEIGHTS statement to provide replicate weights for variance estimation, you do not need to specify a STRATA statement.

The STRATA variables are one or more variables in the DATA= input data set. These variables can be either character or numeric, but the procedure treats them as categorical variables. The formatted values of the STRATA variables determine the STRATA variable levels. Thus, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

PROC SURVEYFREQ excludes an observation from the analysis if it has a missing value for any STRATA variable unless you specify the MISSING option in the PROC SURVEYFREQ statement. For more information, see the section "Missing Values" on page 9825.

You can use multiple STRATA statements to specify STRATA variables. The procedure uses variables from all STRATA statements to define strata.

You can specify the following *option*:

LIST

displays the "Stratum Information" table, which lists all strata together with the corresponding values of the STRATA variables. This table provides the number of observations and the number of clusters in each stratum. This table also provides the sampling fraction in each stratum when you specify the RATE= or TOTAL= option in the PROC SURVEYFREQ statement. For more information, see the section "Stratum Information Table" on page 9869.

TABLES Statement

TABLES *requests* < / *options* > **;**

The TABLES statement requests one-way to *n*-way frequency and crosstabulation tables and statistics for these tables.

If you omit the TABLES statement, PROC SURVEYFREQ generates one-way frequency tables for all DATA= data set variables that are not listed in the other statements.

The following argument is required in the TABLES statement:

requests

specify the frequency and crosstabulation tables to produce. A *request* is composed of one variable name or several variable names separated by asterisks. To request a one-way frequency table, use a single variable. To request a two-way crosstabulation table, use an asterisk between two variables. To request a multiway table (an *n*-way table, where $n > 2$), separate the variables with asterisks. The unique values of these variables form the rows, columns, and layers of the table.

You can specify multiple TABLES statements in a single PROC SURVEYFREQ step. You can also specify any number of table requests in a single TABLES statement.

For two-way tables to multiway tables, the values of the last variable form the crosstabulation table *columns* and the values of the next-to-last variable form the *rows*. Each level (or combination of levels) of the other variables forms one *layer*. By default, PROC SURVEYFREQ produces a separate (two-way) crosstabulation table for each layer. For example, a specification of A * B * C * D in a TABLES statement produces *k* two-way tables, where *k* is the number of different combinations of levels for A and B. Each table lists the levels for D (columns) within each level of C (rows).

Subsetting Multiway Tables When you specify a multiway table request, you can select which two-way subtables to display by specifying level values of the layer variables. For example, the table request Year ['2012' '2013'] * A * B displays two-way subtables of A * B for Year '2012' and Year '2013'; this request suppresses the two-way subtables for levels of Year that are not specified. To subset a multiway table, you can specify *formatted-level-values* in square brackets after the layer variable names. You must enclose each *formatted-level-value* in single or double quotation marks. You can specify one or more levels of each layer variable; when you specify more than one level, separate the levels by a space or a comma.

Variable-level subsetting is available only for layer variables in multiway tables. It is not available for row or column variables in multiway or two-way tables, or for classification variables in one-way tables.

Variable-level subsetting controls only the display of multiway tables. It does not subset the data set, change the degrees of freedom, or otherwise affect the variance estimation.

Grouping Syntax To specify multiple table requests quickly, use a grouping syntax by placing parentheses around several variables and joining other variables or variable combinations. Table 118.3 shows some examples of grouping syntax.

Table 118.3 Grouping Syntax

TABLES Request	Equivalent to
A*(B C)	A*B A*C
(A B)*(C D)	A*C B*C A*D B*D
(A B C)*D	A*D B*D C*D
A – C	A B C
(A – C)*D	A*D B*D C*D

Formatted Variable Levels The TABLES statement variables are one or more variables from the DATA= input data set. These variables can be either character or numeric, but the procedure treats them as categorical variables. PROC SURVEYFREQ uses the formatted values of the TABLES variable to determine the categorical variable levels. If you assign a format to a variable by using a FORMAT statement, PROC SURVEYFREQ formats the values before dividing observations into the levels of a frequency or crosstabulation table. For more information, see the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

Order of Variable Levels By default, the frequency or crosstabulation table lists the values of both character and numeric variables in ascending order based on internal (unformatted) variable values. You can change the order of the values in the table by specifying the ORDER= option in the PROC SURVEYFREQ statement. To list the values in ascending order by formatted value, use ORDER=FORMATTED.

Without Options

If you request a frequency or crosstabulation table without specifying any options, PROC SURVEYFREQ produces the following for each table level or cell:

- frequency (sample size)
- weighted frequency, which estimates the population total
- standard error of the weighted frequency
- percentage, which estimates the population proportion
- standard error of the percentage

The table displays weighted frequencies if your analysis includes a WEIGHT statement, or if you specify the WTFREQ option in the TABLES statement. The table also displays the number of observations that have missing values. For more information, see the sections "One-Way Frequency Tables" on page 9870 and "Crosstabulation Tables" on page 9872.

Options

Table 118.4 summarizes the *options* available in the TABLES statement. Descriptions of the *options* follow the table in alphabetical order.

Table 118.4 TABLES Statement Options

Option	Description
Control Statistical Analysis	
AGREE	Requests kappa coefficients
ALPHA=	Sets level for confidence limits
CHISQ	Requests Rao-Scott chi-square test
CL	Requests confidence limits for percentages and specifies confidence limit type for percentages
CLWT	Requests confidence limits for weighted frequencies
COV	Requests covariances of frequency estimates
COVP	Requests covariances of proportion estimates
DF=	Specifies degrees of freedom
DISCORDDIFF	Requests discordant proportion difference
KAPPA	Requests simple kappa coefficient
LRCHISQ	Requests Rao-Scott likelihood ratio test
OR	Requests odds ratio and relative risks
RISK	Requests risks and risk difference
RISKDIFF	Requests risk difference
SENSPEC	Requests sensitivity and specificity
TESTP=	Specifies null proportions for one-way chi-square test
WCHISQ	Requests Wald chi-square test
WLLCHISQ	Requests Wald log-linear chi-square test
WTKAPPA	Requests weighted kappa coefficient
Request Additional Table Information	
CELLCHI2	Displays cell contributions to the Pearson chi-square
COLUMN	Displays column percentages and standard errors
CV	Displays coefficients of variation for percentages
CVWT	Displays coefficients of variation for weighted frequencies
DEFF	Displays design effects for percentages
DEVIATION	Displays deviations of weighted frequencies
EXPECTED	Displays expected weighted frequencies
ONEWAYTABLE	Displays additional statistics in one-way tables
PEARSONRES	Displays Pearson residuals
ROW	Displays row percentages and standard errors
VAR	Displays variances of percentages
VARWT	Displays variances of weighted frequencies
WTFREQ	Displays totals and standard errors when there is no WEIGHT statement
Control Displayed Output	
DOMAIN=ROW	Displays one-way row domain analysis
NOCELLPERCENT	Suppresses display of overall percentages

Table 118.4 *continued*

Option	Description
NOFREQ	Suppresses display of frequency counts
NOPERCENT	Suppresses display of all percentages
NOPRINT	Suppresses display of tables but displays statistical tests
NOSPARSE	Suppresses display of zero-frequency rows and columns
NOSTD	Suppresses display of standard errors for all estimates
NOTOTAL	Suppresses display of row and column totals
NOWT	Suppresses display of weighted frequencies
Produce Statistical Graphics	
PLOTS=	Requests plots from ODS Graphics

You can specify the following *options*:

AGREE < (*options*) >

requests the simple and weighted kappa coefficients together with their standard errors and confidence limits. You can specify *options* to request additional agreement coefficients and statistics.

Agreement coefficients can be computed for square two-way tables, where the number of rows is the same as the number of columns. For 2×2 tables, the weighted kappa coefficient equals the simple kappa coefficient, and PROC SURVEYFREQ displays only the simple kappa coefficient. For more information, see the section "Kappa Coefficients" on page 9854.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

Agreement coefficients are available for replication variance estimation methods (which you can request by specifying VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE or by including a REPWEIGHTS statement); agreement coefficients are not available for the Taylor series method (VARMETHOD=TAYLOR).

The weighted kappa coefficient is computed by using agreement weights that reflect the relative agreement between pairs of variable levels. Agreement weights are not the same as *sampling weights*, which you provide by specifying the WEIGHT statement. PROC SURVEYFREQ uses sampling weights to compute both the simple and weighted kappa coefficients. For more information, see the section "Weighted Kappa Coefficient" on page 9856.

You can request the simple kappa coefficient or the weighted kappa coefficient separately by specifying the KAPPA or WTKAPPA option, respectively.

You can specify the following *options*:

AC1

requests the AC1 agreement coefficient. For more information, see the section "AC1 Agreement Coefficient" on page 9857.

DETAILS

displays the "Kappa Details" table, which includes the following statistics for the simple kappa coefficient: observed agreement, chance-expected agreement, maximum kappa, and the B_n measure. If the two-way table is 2×2, the "Kappa Details" table also includes the prevalence index and the bias index. For more information, see the section "Simple Kappa Coefficient" on page 9854.

If the two-way table is larger than 2×2, this option also displays the "Weighted Kappa Details" table, which includes the observed agreement and chance-expected agreement components of the weighted kappa coefficient. For more information, see the section "Weighted Kappa Coefficient" on page 9856.

KAPPADETAILS

displays the "Kappa Details" table, which includes the following statistics for the simple kappa coefficient: observed agreement, chance-expected agreement, maximum kappa, and the B_n measure. If the two-way table is 2×2, the "Kappa Details" table also includes the prevalence index and the bias index. For more information, see the section "Simple Kappa Coefficient" on page 9854.

PABAK

requests the prevalence-adjusted bias-adjusted kappa coefficient. For more information, see the section "Prevalence-Adjusted Bias-Adjusted Kappa Coefficient" on page 9858.

PRINTKWTS

displays the agreement weights that PROC SURVEYFREQ uses to compute the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. By default, PROC SURVEYFREQ uses the Cicchetti-Allison form of agreement weights. If you specify the WT=FC option, the procedure uses the Fleiss-Cohen form of agreement weights. For more information, see the section "Kappa Agreement Weights" on page 9856.

WT=FC

requests Fleiss-Cohen agreement weights for the weighted kappa computation. By default, PROC SURVEYFREQ uses Cicchetti-Allison agreement weights to compute the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. For more information, see the section "Kappa Agreement Weights" on page 9856.

WTKAPDETAILS

displays the "Weighted Kappa Details" table, which includes the observed agreement and chance-expected agreement components of the weighted kappa coefficient. This information is available when the two-way table is larger than 2×2. For more information, see the section "Weighted Kappa Coefficient" on page 9856.

ALPHA=α

specifies the level for confidence limits. The value of α must be between 0 and 1; a confidence level of α produces $100(1-\alpha)\%$ confidence limits. By default, ALPHA=0.05, which produces 95% confidence limits.

You can request confidence limits for percentages by specifying the CL option, and you can request confidence limits for weighted frequencies by specifying the CLWT option. For more information, see the sections "Confidence Limits for Proportions" on page 9842 and "Confidence Limits for Totals" on page 9841.

The ALPHA= option also applies to confidence limits for the risks and risk difference (which you can request by specifying the RISK option) and to confidence limits for the odds ratio and relative risks (which you can request by specifying the OR option). For more information, see the sections "Risks and Risk Difference" on page 9850 and "Odds Ratio and Relative Risks" on page 9852.

CELLCHI2

displays each table cell's contribution to the Pearson chi-square statistic in the crosstabulation table. The cell chi-square is computed as $(\textit{weighted frequency} - \textit{expected})^2/\textit{expected}$, where *weighted frequency* is the weighted frequency of the table cell and *expected* is the expected weighted frequency, which is computed under the null hypothesis that the row and column variables are independent. You can display the expected weighted frequencies by specifying the EXPECTED option, and you can display the deviations (*weighted frequency* − *expected*) by specifying the DEVIATION option. For more information, see the sections "Expected Weighted Frequency" on page 9848 and "Rao-Scott Chi-Square Test" on page 9858.

This option has no effect on one-way frequency tables. To display cell chi-squares in one-way frequency tables, you can specify the ONEWAYTABLE(CELLCHI2) option.

CHISQ < (*options*) >

requests the Rao-Scott chi-square test. This is a design-adjusted test that is computed by applying a design correction to the weighted Pearson chi-square statistic. By default, PROC SURVEYFREQ provides a first-order Rao-Scott chi-square test. If you specify CHISQ(SECONDORDER), the procedure provides a second-order (Satterthwaite) Rao-Scott chi-square test. For more information, see the section "Rao-Scott Chi-Square Test" on page 9858.

For one-way tables, the CHISQ option produces a design-based goodness-of-fit test. By default, this is a goodness-of-fit test for equal proportions. If you specify the null hypothesis proportions in the TESTP= option, the CHISQ option produces a chi-square goodness-of-fit test for the specified proportions.

By default for one-way tables, and for first-order tests for two-way tables, the design correction is computed from proportion estimates. If you specify CHISQ(MODIFIED), the design correction is computed from null hypothesis proportions. For second-order tests for two-way tables, the design correction is always computed from null hypothesis proportions.

You can specify the following *options*:

FIRSTORDER

requests a first-order Rao-Scott chi-square test. This is the default for the CHISQ option; if you do not specify CHISQ(SECONDORDER), the procedure provides a first-order Rao-Scott test.

MODIFIED

uses the null hypothesis proportions to compute the Rao-Scott design correction. By default (if you do not specify CHISQ(MODIFIED)), the procedure uses proportion estimates to compute the design correction for all first-order tests and for second-order tests for one-way tables. For second-order tests for two-way tables, the procedure always uses null hypothesis proportions to compute the design correction.

SECONDORDER

requests a second-order (Satterthwaite) Rao-Scott chi-square test. For more information, see the section "Rao-Scott Chi-Square Test" on page 9858.

CL < (*cl-options*) >

requests confidence limits for the percentages (proportions) in frequency and crosstabulation tables. You can use *cl-options* to specify the confidence limit type and computational details. You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

PROC SURVEYFREQ provides the following design-based confidence limit types for percentages: Agresti-Coull, Clopper-Pearson (exact), Jeffreys, logit, uniform, Wald, and Wilson (score). By default, PROC SURVEYFREQ computes Wald-type ("linear") confidence limits by using the variance estimates that are based on the sample design. For more information, see the section "Confidence Limits for Proportions" on page 9842. You can specify the PSMALL or NSMALL= option to request a different confidence limit type for frequency or crosstabulation table cells that have small percentages or frequencies, respectively.

The CL option requests confidence limits for all percentages that you include in the crosstabulation table. To request confidence limits for only row or column percentages, you can specify the ROW(CL) or COLUMN(CL) option, respectively. To suppress confidence limits for row or column percentages, you can specify the ROW(CL=NO) or COLUMN(CL=NO) option, respectively.

You can specify the following *cl-options*:

ADJUST=YES | DP | NO

controls the degrees-of-freedom adjustment to the effective sample size in the computation of modified confidence limits (Agresti-Coull, Clopper-Pearson, Jeffreys, uniform, and Wilson). For more information, see the subsection "Degrees-of-Freedom Adjustment" in the section "Modified Confidence Limits" on page 9843.

By default, ADJUST=YES, which adjusts the effective sample size by using the Korn and Graubard (1998) adjustment factor. If you specify ADJUST=DP, PROC SURVEYFREQ uses the Dean and Pagano (2015) adjustment factor. If you specify ADJUST=NO, the effective sample size is not adjusted.

NSMALL=*n*

requests an alternative confidence limit type for frequency or crosstabulation table cells that have small frequencies. When the (unweighted) table cell frequency is less than or equal to *n*, PROC SURVEYFREQ computes the confidence limit type that you specify in the TYPE= *cl-option*. When the table cell frequency is greater than *n*, PROC SURVEYFREQ computes Wald confidence limits.

The frequency value *n* must be a positive integer. The NSMALL= *cl-option* is available when you specify one of the following TYPE= *cl-options*: AGRESTICOULL, CLOPPERPEARSON, JEFFREYS, LOGIT, UNIFORM, or WILSON. For more information, see the section "Confidence Limits for Proportions" on page 9842.

PSMALL <=*p*>

requests an alternative confidence limit type for extreme (small and large) proportions. When the proportion estimate is less than or equal to *p* or greater than or equal to (1 – *p*), PROC SURVEYFREQ computes the confidence limit type that you specify in the TYPE= *cl-option*. When the proportion estimate is between *p* and (1 – *p*), PROC SURVEYFREQ computes Wald confidence limits.

By default, *p*=0.25; when the proportion estimate is less than or equal to 0.25 or greater than or equal to 0.75, PROC SURVEYFREQ computes the alternative confidence limit type. When the proportion estimate is between 0.25 and 0.75, PROC SURVEYFREQ computes Wald confidence limits.

The value of *p* must be a nonnegative number. You can specify *p* as a proportion between 0 and 0.5. Or you can specify *p* in percentage form as a number between 1 and 50, and PROC SURVEYFREQ converts that number to a proportion. The procedure treats the value 1 as the percentage form 1%.

The PSMALL *cl-option* is available when you specify one of the following TYPE= *cl-options*: AGRESTICOULL, CLOPPERPEARSON, JEFFREYS, LOGIT, UNIFORM, or WILSON. For more information, see the section "Confidence Limits for Proportions" on page 9842.

SENSPEC

uses the confidence limit type that you specify in the CL(TYPE=) option to compute confidence limits for statistics in the "Sensitivity and Specificity" table (which you can request by specifying the SENSPEC option in the TABLES statement). By default, PROC SURVEYFREQ computes Wald-type confidence limits for the SENSPEC statistics.

When you specify the SENSPEC *cl-option* together with a modified confidence limit type (Agresti-Coull, Clopper-Pearson, Jeffreys, uniform, or Wilson), the ADJUST=, TRUNCATE=, and VARDEF=N *cl-options* also apply to the SENSPEC confidence limits.

The CL(SENSPEC) option is equivalent to the SENSPEC(CLTYPE) option.

TRUNCATE=YES | NO

controls the truncation of the effective sample size in the computation of modified confidence limits (Agresti-Coull, Clopper-Pearson, Jeffreys, uniform, and Wilson). By default, TRUNCATE=YES truncates the effective sample size if it is larger than the original sample size. If you specify TRUNCATE=NO, the effective sample size is not truncated. For more information, see the subsection "Truncation" in the section "Modified Confidence Limits" on page 9843.

TYPE=*type*

specifies the *type* of confidence limits to compute for proportions. By default, PROC SURVEYFREQ computes Wald confidence limits (TYPE=WALD).

If you specify the PSMALL *cl-option*, the procedure computes the specified confidence limit type for extreme (small and large) proportions and computes Wald confidence limits otherwise. If you specify the NSMALL=*n cl-option*, the procedure computes the specified confidence limit type when the table cell frequency is less than or equal to *n*. If you omit the PSMALL and NSMALL= *cl-options*, PROC SURVEYFREQ computes the confidence limit type that you specify for all proportions.

You can specify one of the following confidence limit *types*:

AGRESTICOULL

AC

requests modified Agresti-Coull confidence limits for proportions. For more information, see the section "Modified Agresti-Coull Confidence Limits" on page 9844.

CLOPPERPEARSON

CP

requests modified Clopper-Pearson (exact) confidence limits for proportions. For more information, see the section "Modified Clopper-Pearson Confidence Limits" on page 9845.

JEFFREYS

requests modified Jeffreys confidence limits for proportions. For more information, see the section "Modified Jeffreys Confidence Limits" on page 9845.

LOGIT

requests logit confidence limits for proportions. For more information, see the section "Logit Confidence Limits" on page 9843.

UNIFORM

requests modified uniform confidence limits for proportions. For more information, see the section "Modified Uniform Confidence Limits" on page 9845.

WALD

requests Wald ("linear") confidence limits for proportions. This is the default confidence limit type if you do not specify the TYPE= option. For more information, see the section "Wald Confidence Limits" on page 9842.

WILSON

SCORE

requests modified Wilson (score) confidence limits for proportions. For more information, see the section "Modified Wilson Confidence Limits" on page 9845.

VARDEF=N

uses the number of observations (n) as the divisor in the simple random sampling (SRS) variance component of the design effect for modified confidence limits. By default, PROC SURVEYFREQ uses ($n - 1$) as the divisor. For more information, see the section "Design Effect" on page 9847.

This option applies only to design effects that are used to compute effective sample sizes for modified confidence limits (Agresti-Coull, Clopper-Pearson, Jeffreys, uniform, and Wilson). For more information, see the subsection "Effective Sample Size" in the section "Modified Confidence Limits" on page 9843. To control the divisor for all design effects that PROC SURVEYFREQ computes, you can specify the VARDEF=N option in the PROC SURVEYFREQ statement.

CLWT

requests confidence limits for the weighted frequencies (totals) in the crosstabulation table. You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits. For more information, see the section "Confidence Limits for Totals" on page 9841.

COLUMN < (*column-options*) >

displays the column percentage (estimated proportion of the column total) for each cell in a two-way table. This option also provides standard errors of the column percentages. For more information, see the section "Row and Column Proportions" on page 9840. This option has no effect on one-way frequency tables.

You can specify the following *column-options*:

CL < (*cl-options*)| =NO >

requests confidence limits for the column percentages (proportions). For more information, see the section "Confidence Limits for Proportions" on page 9842.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

You can specify the CL=NO *column-option* to suppress the display of confidence limits for column percentages (while displaying confidence limits for other percentages). By default, the overall CL option controls the display of confidence limits for all percentages (overall, row, and column percentages).

You can specify *cl-options* to control the confidence limit computations. The COLUMN *cl-options* are the same *cl-options* that are available for the overall CL option, which requests confidence limits for all percentages (overall, row, and column percentages). For a description of the *cl-options*, see the overall CL option. In a single TABLES statement, PROC SURVEYFREQ accepts only one value of each *cl-option*; for example, you cannot specify one TYPE= value for the overall CL option and a different TYPE= value for the COLUMN(CL) option.

CV < =NO >

displays coefficients of variation for the column percentages. For more information, see the section "Coefficient of Variation" on page 9847.

You can specify the CV=NO *column-option* to suppress the display of coefficients of variation for column percentages (while displaying coefficients of variation for other percentages). By default, the overall CV option controls display of coefficients of variation for all percentages (overall, row, and column percentages).

DEFF

displays design effects for the column percentages. For more information, see the section "Design Effect" on page 9847.

NOSTD

suppresses the display of standard errors for column percentages.

VAR < =NO >

displays variance estimates for the column percentages. For more information, see the section "Proportions" on page 9839.

You can specify the VAR=NO *column-option* to suppress the display of variances estimates for column percentages (while displaying variance estimates for other percentages). By default, the overall VAR option controls the display of variance estimates for all percentages (overall, row, and column percentages).

COV

requests the covariance matrix of the table cell frequency estimates. For more information, see the section "Covariances of Frequency Estimates" on page 9838.

COVP

requests the covariance matrix of the proportion estimates.

CV

displays the coefficient of variation for each percentage (proportion) estimate in the crosstabulation table. For more information, see the section "Coefficient of Variation" on page 9847.

To display coefficients of variation only for row or column percentages, you can specify the ROW(CV) or COLUMN(CV) option, respectively.

CVWT

displays the coefficient of variation for each weighted frequency (estimated total), in the crosstabulation table. For more information, see the section "Coefficient of Variation" on page 9847.

DEFF

displays the design effect for each overall percentage (proportion) estimate in the crosstabulation table. For more information, see the section "Design Effect" on page 9847.

To include or exclude the finite population correction (*fpc*) in the simple random sampling (SRS) component of the design effect, you can specify the DEFF(FPC=) option in the PROC SURVEYFREQ statement.

To use n instead of $(n - 1)$ as the divisor in the SRS component of the design effect, you can specify the DEFF(VARDEF=N) option in the PROC SURVEYFREQ statement.

To display design effects for row or column percentages, you can specify the ROW(DEFF) or COLUMN(DEFF) option, respectively.

DEVIATION

displays the deviations of the weighted frequencies from the expected weighted frequencies (*weighted frequency – expected*) in the crosstabulation table. The expected weighted frequencies are computed under the null hypothesis that the row and column variables are independent. You can display the expected values by specifying the EXPECTED option. For more information, see the section "Expected Weighted Frequency" on page 9848.

This option has no effect on one-way frequency tables. To display deviations in one-way frequency tables, you can specify the ONEWAYTABLE(DEVIATION) option.

DF=*df*

specifies the degrees of freedom for the analysis. The value of *df* must be a nonnegative number. By default, PROC SURVEYFREQ computes the degrees of freedom as described in the section "Degrees of Freedom" on page 9846.

PROC SURVEYFREQ uses the value *df* to obtain the t distribution's percentile for confidence limits for proportions, totals, and other statistics. For more information, see the section "Confidence Limits for Proportions" on page 9842. PROC SURVEYFREQ also uses *df* to compute the denominator degrees of freedom for the F statistics in the Rao-Scott and Wald chi-square tests. For more information, see the sections "Rao-Scott Chi-Square Test" on page 9858, "Rao-Scott Likelihood Ratio Chi-Square Test" on page 9864, "Wald Chi-Square Test" on page 9865, and "Wald Log-Linear Chi-Square Test" on page 9867.

DISCORDDIFF

requests the difference between discordant proportions for 2×2 tables. This option displays the discordant proportion difference together with its standard error, confidence limits, and *t* test. For more information, see the section "Discordant Proportion Difference" on page 9854.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

DOMAIN=ROW

ONEWAY

displays a separate one-way frequency table for each row variable level in a two-way or multiway crosstabulation. Each one-way frequency table provides the distribution of the column (last) variable in the domain that is defined by the row variable level (together with the layer variable levels if the crosstabulation is multiway). The percentages, standard errors, confidence limits, and other statistics in a DOMAIN=ROW one-way frequency table are the same values that you can display in the corresponding two-way table by specifying the ROW option. For more information, see the sections "Domain Analysis" on page 9824 and "Row and Column Proportions" on page 9840.

All display and statistic options that apply to general one-way table analysis are available for DOMAIN=ROW one-way table analysis. When you specify the DOMAIN=ROW option, the CHISQ and LRCHISQ options produce one-way chi-square tests for the row-level domains. You can specify null proportions for these chi-square tests in the TESTP= option.

EXPECTED

displays the expected weighted frequencies for the cells in the crosstabulation table. The expected weighted frequencies are computed under the null hypothesis that the row and column variables are independent. For more information, see the section "Expected Weighted Frequency" on page 9848.

This option has no effect on one-way frequency tables. To display expected weighted frequencies in one-way frequency tables, you can specify the ONEWAYTABLE(EXPECTED) option.

KAPPA <(*option*)>

requests the simple kappa coefficient with its standard error and confidence limits. The kappa coefficient can be computed for square two-way tables, where the number of rows equals the number of columns. For more information, see the section "Simple Kappa Coefficient" on page 9854.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

The kappa coefficient is available for replication variance estimation methods (which you can request by specifying VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE or by including a REPWEIGHTS statement); the kappa coefficient is not available for the Taylor series method (VARMETHOD=TAYLOR).

You can specify the following *option*:

DETAILS

KAPPADETAILS

displays the "Kappa Details" table, which includes the following statistics: observed agreement, chance-expected agreement, maximum possible kappa, and the B_n measure. If the two-way table is 2×2, the "Kappa Details" table also includes the prevalence index and the bias index. For more information, see the section "Simple Kappa Coefficient" on page 9854.

LRCHISQ < (*options*) >

requests the Rao-Scott likelihood ratio chi-square test. This is a design-adjusted test that is computed by applying a design correction to the weighted likelihood ratio chi-square statistic. By default, PROC SURVEYFREQ provides a first-order Rao-Scott likelihood ratio test. If you specify LRCHISQ(SECONDORDER), the procedure provides a second-order (Satterthwaite) Rao-Scott likelihood ratio test. For more information, see the section "Rao-Scott Likelihood Ratio Chi-Square Test" on page 9864.

For one-way tables, the LRCHISQ option produces a design-based likelihood ratio goodness-of-fit test. By default, the null hypothesis is equal proportions. If you specify null hypothesis proportions in the TESTP= option, the LRCHISQ option produces a design-based likelihood ratio test for the specified proportions.

By default for one-way tables, and for first-order tests for two-way tables, the design correction is computed from proportion estimates. If you specify LRCHISQ(MODIFIED), the design correction is computed from null hypothesis proportions. For second-order tests for two-way tables, the design correction is always computed from null hypothesis proportions.

You can specify the following *options*:

FIRSTORDER

requests a first-order Rao-Scott likelihood ratio test. This is the default for the LRCHISQ option; if you do not specify LRCHISQ(SECONDORDER), the procedure provides a first-order Rao-Scott test.

MODIFIED

uses the null hypothesis proportions to compute the Rao-Scott design correction. By default (if you do not specify LRCHISQ(MODIFIED)), the procedure uses proportion estimates to compute the design correction for all first-order tests and for second-order tests for one-way tables. For second-order tests for two-way tables, the procedure always uses null hypothesis proportions to compute the design correction.

SECONDORDER

requests a second-order (Satterthwaite) Rao-Scott likelihood ratio test. For more information, see the section "Rao-Scott Likelihood Ratio Chi-Square Test" on page 9864.

NOCELLPERCENT

suppresses the display of overall cell percentages (and their standard errors) in the crosstabulation table. This option does not suppress the display of row or column percentages, which you can request by specifying the ROW or COLUMN option.

NOFREQ

suppresses the display of cell frequencies in the crosstabulation table. The NOFREQ option also suppresses the display of row, column, and overall table frequencies.

NOPERCENT

suppresses the display of all percentages in the crosstabulation table. The NOPERCENT option also suppresses the display of standard errors of the percentages. You can use the NOCELLPERCENT option to suppress display of overall cell percentages but allow display of row or column percentages.

NOPRINT

suppresses the display of frequency and crosstabulation tables but displays all requested statistical tests. This option disables the Output Delivery System (ODS) for the suppressed tables. For more information, see Chapter 23, "Using the Output Delivery System."

NOSPARSE

suppresses the display of zero-frequency variable levels in two-way tables. By default, PROC SURVEYFREQ displays all levels of the column variable within each level of the row variable, including any column variable levels that have frequencies of 0 in the row. By default for multiway tables, PROC SURVEYFREQ displays all levels of the row variable within each layer of the table, including any row variable levels that have frequencies of 0 in the layer.

NOSTD

suppresses the display of all standard errors in the crosstabulation table.

NOTOTAL

suppresses the display of row totals, column totals, and overall totals in the crosstabulation table.

NOWT

suppresses the display of weighted frequencies in the crosstabulation table. The NOWT option also suppresses the display of standard errors of the weighted frequencies.

ONEWAYTABLE (*options*)

displays additional statistics in one-way frequency tables. For more information about the contents of one-way frequency tables, see the section "One-Way Frequency Tables" on page 9870.

You can specify the following *options*:

CELLCHI2

displays each table cell's contribution to the Pearson chi-square statistic in the one-way frequency table. The cell chi-square is computed as $(\textit{weighted frequency} - \textit{expected})^2/\textit{expected}$, where *weighted frequency* is the weighted frequency of the table cell and *expected* is the expected weighted frequency under the null hypothesis of the chi-square test. For more information, see the subsection "One-Way Tables" in the section "Rao-Scott Chi-Square Test" on page 9858.

You can display the expected weighted frequencies by specifying the ONEWAYTABLE(EXPECTED) option, and you can display the deviations (*weighted frequency* − *expected*) by specifying the ONEWAYTABLE(DEVIATION) option.

DEVIATION

displays the deviation of the weighted frequency from the expected weighted frequency (*weighted frequency* − *expected*) for each level (cell) in the one-way frequency table, where *expected* is the expected weighted frequency under the null hypothesis of the chi-square test. For more information, see the subsection "One-Way Tables" in the section "Rao-Scott Chi-Square Test" on page 9858. You can display the expected weighted frequencies by specifying the ONEWAYTABLE(EXPECTED) option.

EXPECTED

displays the expected weighted frequency for each level (cell) in the one-way frequency table. The expected weighted frequencies are computed under the null hypothesis of the chi-square test. By default, the expected weighted frequency for each level is the total weighted frequency divided by the number of levels. Alternatively, the expected weighted frequencies are computed by using the null hypothesis proportions that you provide in the TESTP= option. For more information, see the subsection "One-Way Tables" in the section "Rao-Scott Chi-Square Test" on page 9858.

PEARSONRES

displays the Pearson residual for each level (cell) in the one-way frequency table. The Pearson residual is the square root of the level's contribution to the Pearson chi-square statistic. The Pearson residual is computed as $(\textit{weighted frequency} - \textit{expected})/\sqrt{\textit{expected}}$, where *weighted frequency* is the weighted frequency of the level and *expected* is the expected weighted frequency of the level under the null hypothesis of the chi-square test. For more information, see the subsection "One-Way Tables" in the section "Rao-Scott Chi-Square Test" on page 9858.

OR

RELRISK

requests estimates of the odds ratio, the column 1 relative risk, and the column 2 relative risk for 2×2 tables. The OR option also provides confidence limits for these statistics. For more information, see the section "Odds Ratio and Relative Risks" on page 9852.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

PEARSONRES

displays each crosstabulation table cell's Pearson residual, which is the square root of the table cell's contribution to the Pearson chi-square statistic. The Pearson residual is computed as $(\textit{weighted frequency} - \textit{expected})/\sqrt{\textit{expected}}$, where *weighted frequency* is the weighted frequency of the table cell and *expected* is the expected weighted frequency, which is computed under the null hypothesis that the row and column variables are independent. You can display the expected values, the deviations, and the cell chi-squares by specifying the EXPECTED, DEVIATION, and CELLCHI2 options, respectively. For more information, see the sections "Expected Weighted Frequency" on page 9848 and "Rao-Scott Chi-Square Test" on page 9858.

This option has no effect on one-way frequency tables. To display Pearson residuals in one-way frequency tables, you can specify the ONEWAYTABLE(PEARSONRES) option.

PLOTS < (*global-plot-options*) > < =*plot-request* < (*plot-options*) > >

PLOTS < (*global-plot-options*) > < =(*plot-request* < (*plot-options*) > < . . . *plot-request* < (*plot-options*) > >) >

controls the plots that are produced through ODS Graphics. *Plot-requests* identify the plots, and *plot-options* control the appearance and content of the plots. You can specify *plot-options* in parentheses after a *plot-request*. A *global-plot-option* applies to all plots for which it is available unless it is altered by a specific *plot-option*. You can specify *global-plot-options* in parentheses after the PLOTS option.

When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. For example:

```
plots=all
plots=wtfreqplot
plots=(wtfreqplot oddsratioplot)
plots(only)=(riskdiffplot relriskplot)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;
proc surveyfreq;
   tables treatment*response / chisq plots=wtfreqplot;
   weight wt;
run;
ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, PROC SURVEYFREQ produces all plots that are associated with the analyses that you request, with the exception of weighted frequency plots and mosaic plots. To produce a weighted frequency plot or mosaic plot when ODS Graphics is enabled, you must specify the WTFREQPLOT or MOSAICPLOT *plot-request*, or you must specify the PLOTS=ALL option. PROC SURVEYFREQ produces the remaining plots (listed in Table 118.5) by default when you request the corresponding TABLES statement options.

You can suppress default plots and request specific plots by using the PLOTS(ONLY)= option; PLOTS(ONLY)=(*plot-requests*) produces only the plots that are specified as *plot-requests*. You can suppress all plots by specifying the PLOTS=NONE option.

See Figure 118.4 and Figure 118.7 for examples of plots that PROC SURVEYFREQ produces. For information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Plot Requests

Table 118.5 lists the available *plot-requests* together with their required TABLES statement options. Descriptions of the *plot-requests* follow the table in alphabetical order.

Table 118.5 Plot Requests

Plot Request	Description	Required TABLES Statement Option
ALL	All plots	None
KAPPAPLOT	Kappa plot	AGREE or KAPPA ($h \times r \times r$ table)
MOSAICPLOT	Mosaic plot	Crosstabulation table request
NONE	No plots	None
ODDSRATIOPLOT	Odds ratio plot	OR ($h \times 2 \times 2$ table)
RELRISKPLOT	Relative risk plot	OR ($h \times 2 \times 2$ table)
RISKDIFFPLOT	Risk difference plot	RISK or RISKDIFF, ($h \times 2 \times 2$ table)
WTFREQPLOT	Weighted frequency plot	Frequency or crosstabulation table request
WTKAPPAPLOT	Weighted kappa plot	AGREE or WTKAPPA ($h \times r \times r$ table, $r > 2$)

You can specify the following *plot-requests*:

ALL

requests all plots that are associated with the specified analyses. If you specify the PLOTS=ALL option, PROC SURVEYFREQ also produces the weighted frequency and mosaic plots that are associated with the tables that you request. (PROC SURVEYFREQ does not produce weighted frequency and mosaic plots by default when ODS Graphics is enabled.)

KAPPAPLOT < (*plot-options*) >

requests a plot of kappa coefficients with confidence limits. Kappa plots are available for multiway square tables and display the simple kappa coefficient (with confidence limits) for each two-way table layer. To produce a kappa plot, you must specify the KAPPA or AGREE option in the TABLES statement to compute kappa coefficients.

Table 118.6 lists the *plot-options* that are available for kappa plots. For descriptions of the *plot-options*, see the subsection "Plot Options."

Table 118.6 Plot Options for KAPPAPLOT and WTKAPPAPLOT

Plot Option	Description	Values
CLDISPLAY=	Error bar type	**BAR**, **LINE**, **LINEARROW**, **SERIF***, or **SERIFARROW**
NPANELPOS=	Statistics per graphic	Number (All*)
ORDER=	Order of two-way levels	**ASCENDING** or **DESCENDING**
RANGE=	Range to display	Values or **CLIP**
STATS	Statistic values	None

*Default

MOSAICPLOT < (*plot-options*) >

requests a mosaic plot. Mosaic plots are available for crosstabulation tables. For multiway tables, PROC SURVEYFREQ provides a mosaic plot for each two-way table layer.

To produce a mosaic plot, you must specify the MOSAICPLOT *plot-request* in the PLOTS= option, or you must specify the PLOTS=ALL option. PROC SURVEYFREQ does not produce mosaic plots by default when ODS Graphics is enabled.

Mosaic plots display tiles that correspond to the crosstabulation table cells. The areas of the tiles are proportional to the weighted frequencies of the table cells. The column variable is displayed on the X axis, and the tile widths are proportional to the relative weighted frequencies of the column variable levels. The row variable is displayed on the Y axis, and the tile heights are proportional to the relative weighted frequencies of the row levels within column levels. For more information, see Friendly (2000).

By default, the colors of the tiles correspond to the row variable levels. If you specify the COLORSTAT *plot-option*, the tiles are colored according to the values of the Pearson residuals.

You can specify the following *plot-options*:

COLORSTAT < =PEARSONRES >

colors the mosaic plot tiles according to the values of the Pearson residuals. A table cell's Pearson residual is the square root of its contribution to the Pearson chi-square statistic. The Pearson residual is computed as $(\textit{weighted frequency} - \textit{expected})/\sqrt{\textit{expected}}$, where *weighted frequency* is the weighted frequency of the table cell and *expected* is the expected weighted frequency. You can specify the PEARSONRES option to display the Pearson residuals in the crosstabulation table.

SQUARE

produces a square mosaic plot, where the height of the Y axis equals the width of the X axis. In a square mosaic plot, the scale of the relative weighted frequencies is the same on both axes. By default, PROC SURVEYFREQ produces a rectangular mosaic plot.

NONE

suppresses all plots.

ODDSRATIOPLOT < (*plot-options*) >

requests a plot of odds ratios with confidence limits. Odds ratio plots are available for multiway 2×2 tables and display the odds ratio (with confidence limits) for each 2×2 table layer. To produce an odds ratio plot, you must specify the OR option in the TABLES statement for a multiway 2×2 table.

Table 118.7 lists the *plot-options* that are available for odds ratio plots. For descriptions of the *plot-options*, see the subsection "Plot Options."

Table 118.7 Plot Options for ODDSRATIOPLOT, RELRISKPLOT, and RISKDIFFPLOT

Plot Option	Description	Values
CLDISPLAY=	Error bar type	**BAR**, **LINE**, **LINEARROW**, **SERIF***, or **SERIFARROW**
COLUMN=**	Risk column	**1*** or **2**
LOGBASE=***	Axis scale	**2**, **E**, or **10**
NPANELPOS=	Statistics per graphic	Number (All*)
ORDER=	Order of two-way levels	**ASCENDING** or **DESCENDING**
RANGE=	Range to display	Values or **CLIP**
STATS	Statistic values	None

*Default

**Available for RELRISKPLOT and RISKDIFFPLOT

***Available for ODDSRATIOPLOT and RELRISKPLOT

RELRISKPLOT < (*plot-options*) >

requests a plot of relative risks with confidence limits. Relative risk plots are available for multiway 2×2 tables and display the relative risk (with confidence limits) for each 2×2 table layer. To produce a relative risk plot, you must specify the OR option in the TABLES statement for a multiway 2×2 table.

Table 118.7 lists the *plot-options* that are available for relative risk plots. For descriptions of the *plot-options*, see the subsection "Plot Options."

RISKDIFFPLOT < (*plot-options*) >

requests a plot of risk differences with confidence limits. Risk difference plots are available for multiway 2 × 2 tables and display the risk difference (with confidence limits) for each 2 × 2 table layer. To produce a risk difference plot, you must specify the RISK or RISKDIFF option in the TABLES statement for a multiway 2 × 2 table.

Table 118.7 lists the *plot-options* that are available for risk difference plots. For descriptions of the *plot-options*, see the subsection "Plot Options."

WTFREQPLOT < (*plot-options*) >

requests a weighted frequency plot. Weighted frequency plots are available for frequency and crosstabulation tables. For multiway tables, PROC SURVEYFREQ provides a two-way weighted frequency plot for each two-way table layer.

To produce a weighted frequency plot, you must specify the WTFREQPLOT *plot-request* in the PLOTS= option, or you must specify the PLOTS=ALL option. PROC SURVEYFREQ does not produce weighted frequency plots by default when ODS Graphics is enabled.

By default, PROC SURVEYFREQ displays weighted frequency plots as bar charts. You can specify the TYPE=DOTPLOT *plot-option* to display frequency plots as dot plots. You can plot weighted percentages instead of frequencies by specifying the SCALE=PERCENT *plot-option*. There are four frequency plot layouts available, which you can request by specifying the TWOWAY= *plot-option*. For more information, see the subsection "Plot Options."

By default, the primary grouping of graph cells in a two-way layout is by column variable. Row variable levels are then displayed within column variable levels. You can specify the GROUPBY=ROW *plot-option* to group first by row variable.

Weighted frequency plots for one-way tables display confidence limits by default. For two-way tables, weighted frequency plots display confidence limits by default in the TWOWAY=GROUPVERTICAL and TWOWAY=GROUPHORIZONTAL layouts. You can suppress confidence limits by specifying the CLBAR=NO *plot-option*. Confidence limits are not available for two-way plots in the TWOWAY=CLUSTER and TWOWAY=STACKED layouts.

Table 118.8 lists the *plot-options* that are available for weighted frequency plots. For descriptions of the *plot-options*, see the subsection "Plot Options."

Table 118.8 Plot Options for WTFREQPLOT

Plot Option	Description	Values
CLBAR=	Confidence limit bars	**NO** or **YES***
GROUPBY=**	Primary group	**COLUMN*** or **ROW**
NPANELPOS=**	Sections per panel	Number (4*)
ORIENT=	Orientation	**HORIZONTAL** or **VERTICAL***
SCALE=	Scale	**PERCENT** or **WTFREQ***
TWOWAY=**	Two-way layout	**CLUSTER**, **GROUPHORIZONTAL**, **GROUPVERTICAL***, or **STACKED**
TYPE=	Type	**BARCHART*** or **DOTPLOT**

*Default

**For two-way tables

WTKAPPAPLOT < (*plot-options*) >

requests a plot of weighted kappa coefficients with confidence limits. Weighted kappa plots are available for multiway square tables and display the weighted kappa coefficient (with confidence limits) for each two-way table layer. To produce a weighted kappa plot, you must specify the WTKAPPA or AGREE option in the TABLES statement to compute weighted kappa coefficients, and the table dimension must be greater than 2.

Table 118.6 lists the *plot-options* that are available for weighted kappa plots. For descriptions of the *plot-options*, see the subsection "Plot Options."

Global Plot Options

A *global-plot-option* applies to all plots for which the option is available unless it is altered by an individual *plot-option*.

You can specify *global-plot-options* in parentheses after the PLOTS option. For example:

```
plots(order=ascending stats)=(riskdiffplot oddsratioplot)
plots(only)=wtfreqplot
```

You can specify any *plot-option* that is listed in Table 118.8 or Table 118.7 as a *global-plot-option*. In addition to these *plot-options*, you can specify the following *global-plot-option*:

ONLY

suppresses the default plots and requests only the plots that are specified as *plot-requests*.

Plot Options

You can specify the following *plot-options*:

CLBAR=NO | YES

controls the confidence limit error bars in weighted frequency plots (WTFREQPLOT). By default, CLBAR=YES, which displays confidence limits error bars; CLBAR=NO suppresses confidence limit error bars.

This *plot-option* applies to all weighted frequency plots except those two-way plots that are displayed in the TWOWAY=CLUSTER or TWOWAY=STACKED layout. Confidence limit error bars are not available in the TWOWAY=CLUSTER and TWOWAY=STACKED layouts.

CLDISPLAY=BAR < *width* > **| LINE | LINEARROW | SERIF | SERIFARROW**

controls the appearance of the confidence limit error bars. This *plot-option* is available for the following plots: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

By default, CLDISPLAY=SERIF, which displays the confidence limits as lines with serifs. CLDISPLAY=LINE displays the confidence limits as plain lines without serifs. The CLDISPLAY=SERIFARROW and CLDISPLAY=LINEARROW *plot-options* display arrowheads on any error bars that are clipped by the RANGE= *plot-option*; if an entire error bar is cut from the plot, the plot displays an arrowhead that points toward the statistic.

CLDISPLAY=BAR displays the confidence limits as bars. By default, the width of the bars equals the size of the marker for the estimate. You can control the width of the bars and the size of the marker by specifying the value of *width* as a percentage of the distance between bars, $0 < width \leq 1$. The bar might disappear when the value of *width* is very small.

COLUMN=1 | 2

specifies the 2×2 table column for which to compute the risks (proportions) for the relative risk plot (RELRISKPLOT) and the risk difference plot (RISKDIFFPLOT). If you specify COLUMN=1, the plot displays the column 1 relative risks or risk differences; if you specify COLUMN=2, the plot displays the column 2 relative risks or risk differences.

If you omit this option for the risk difference plot, the plot uses the table column that you specify for the risk difference statistics in the RISK(COLUMN=) or RISKDIFF(COLUMN=) option. If you omit these COLUMN= options, then by default, COLUMN=1 for the risk difference plot. For the relative risk plot, by default, COLUMN=1.

GROUPBY=COLUMN | ROW

specifies the primary grouping for two-way weighted frequency plots, which you can request by specifying the WTFREQPLOT *plot-request*.

By default, GROUPBY=COLUMN, which groups graph cells first by column variable and displays row variable levels within column variable levels. You can specify GROUPBY=ROW to group first by row variable. In two-way and multiway table requests, the column variable is the last variable specified and forms the columns of the crosstabulation table. The row variable is the next-to-last variable specified and forms the rows of the table.

By default for a bar chart that is displayed in the TWOWAY=STACKED layout, bars correspond to the column variable levels and row levels are displayed (stacked) within each column bar. By default for a bar chart that is displayed in the TWOWAY=CLUSTER layout, bars are first grouped by column variable levels, and row levels are displayed as adjacent bars within each column-level group. You can reverse the default row and column variable groupings by specifying GROUPBY=ROW.

LOGBASE=2 | E | 10

applies to the odds ratio plot (ODDSRATIOPLOT) and the relative risk plot (RELRISKPLOT). This *plot-option* displays the odds ratio or relative risk axis on the log scale that you specify.

NPANELPOS=*n*

divides the plot into multiple panels that display at most $|n|$ statistics or sections.

If *n* is positive, the number of statistics or sections per panel is balanced; if *n* is negative, the number of statistics per panel is not balanced. For example, suppose you want to display 21 odds ratios. NPANELPOS=20 displays two panels, the first with 11 odds ratios and the second with 10 odds ratios; NPANELPOS=–20 displays 20 odds ratios in the first panel but only 1 in the second panel. This *plot-option* is available for all plots except mosaic plots and one-way weighted frequency plots.

For two-way weighted frequency plots (WTFREQPLOT), NPANELPOS=*n* requests that panels display at most $|n|$ sections, where sections correspond to row or column variable levels, depending on the type of plot and the grouping. By default, *n*=4 and each panel includes at most four sections. This *plot-option* applies to two-way plots that are displayed in the TWOWAY=GROUPVERTICAL or TWOWAY=GROUPHORIZONTAL lay-

out. The NPANELPOS= *plot-option* does not apply to the TWOWAY=CLUSTER and TWOWAY=STACKED layouts, which are always displayed in a single panel.

For plots that display statistics with confidence limits, NPANELPOS=*n* requests that panels display at most |*n*| statistics. By default, *n*=0 and all statistics are displayed in a single panel. This *plot-option* applies to the following plots: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

ORDER=ASCENDING | DESCENDING

displays the two-way table (layer) statistics in order of the statistic value. If you specify ORDER=ASCENDING or ORDER=DESCENDING, the plot displays the statistics in ascending or descending order, respectively. By default, the order of the statistics in the plot matches the order that the two-way table layers appear in the multiway table.

This *plot-option* is available for the following plots: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

ORIENT=HORIZONTAL | VERTICAL

controls the orientation of weighted frequency plots (WTFREQPLOT). This *plot-option* places the variable levels on the Y axis and the weighted frequencies or percentages on the X axis. ORIENT=VERTICAL places the variable levels on the X axis. The default orientation is ORIENT=VERTICAL for bar charts (TYPE=BARCHART) and ORIENT=HORIZONTAL for dot plots (TYPE=DOTPLOT).

RANGE=(< *min* > < , *max* >)| CLIP

specifies the range of values to display. If you specify RANGE=CLIP, the confidence limits are clipped and the display range is determined by the minimum and maximum values of the estimates. By default, the display range includes all confidence limits.

This *plot-option* is available for the following plots: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

SCALE=PERCENT | WTFREQ

specifies the scale of the frequencies in weighted frequency plots (WTFREQPLOT). SCALE=WTFREQ displays weighted frequencies (totals), and SCALE=PERCENT displays percentages. The default scale is SCALE=WTFREQ.

STATS

displays the values of the statistics and their confidence limits on the right side of the plot. If you do not specify this *plot-option*, the statistic values are not displayed.

This *plot-option* is available for the following plots: KAPPAPLOT, ODDSRATIOPLOT, RELRISKPLOT, RISKDIFFPLOT, and WTKAPPAPLOT.

TWOWAY=CLUSTER | GROUPHORIZONTAL | GROUPVERTICAL | STACKED

specifies the layout for two-way weighted frequency plots (WTFREQPLOT).

All TWOWAY= layouts are available for bar charts (TYPE=BARCHART). All TWOWAY= layouts except TWOWAY=CLUSTER are available for dot plots (TYPE=DOTPLOT). Confidence limits (CLBAR=) can be displayed in the GROUPVERTICAL and GROUPHORIZONTAL layouts. Confidence limits are not available in the STACKED and CLUSTER layouts. The ORIENT= and GROUPBY= *plot-options* are available for all TWOWAY= layouts.

The default two-way layout is TWOWAY=GROUPVERTICAL, which produces a grouped plot that has a vertical common baseline. By default for bar charts (TYPE=BARCHART, ORIENT=VERTICAL), the X axis displays column variable levels, and the Y axis displays weighted frequencies. The plot includes a vertical (Y-axis) block for each row variable level. The relative positions of the graph cells in this plot layout are the same as the relative positions of the table cells in the crosstabulation table. You can reverse the default row and column grouping by specifying the GROUPBY=ROW *plot-option*.

The TWOWAY=GROUPHORIZONTAL layout produces a grouped plot that has a horizontal common baseline. By default (GROUPBY=COLUMN), the plot displays a block on the X axis for each column variable level. Within each column-level block, the plot displays row variable levels.

The TWOWAY=STACKED layout produces stacked displays of weighted frequencies. By default (GROUPBY=COLUMN) in a stacked bar chart, the bars correspond to column variable levels, and row levels are stacked within each column level. By default in a stacked dot plot, the dotted lines correspond to column levels, and cell weighted frequencies are plotted as data dots on the corresponding column line. The dot color identifies the row level.

The TWOWAY=CLUSTER layout, which is available only for bar charts, displays groups of adjacent bars. By default, the primary grouping is by column variable level, and row levels are displayed within each column level.

You can reverse the default row and column grouping in any layout by specifying the GROUPBY=ROW *plot-option*. By default, GROUPBY=COLUMN, which groups first by column variable.

TYPE=BARCHART | DOTPLOT

specifies the type (form) of the weighted frequency plots (WTFREQPLOT). TYPE=BARCHART produces a bar chart and TYPE=DOTPLOT produces a dot plot. The default type is TYPE=BARCHART.

RISK < (*option*) >

requests risk statistics for 2 × 2 tables. Risk statistics include the row 1 risk (proportion of row 1), row 2 risk, overall risk, and risk difference (row 1 risk – row 2 risk), together with their standard errors and confidence limits. For more information, see the section "Risks and Risk Difference" on page 9850.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits. To display only the risk difference (and suppress display of the row 1, row 2, and overall risks), you can specify the RISKDIFF(ONLY) option.

You can specify the following *option*:

COLUMN=1 | 2 | BOTH

specifies the table column for which to compute the risk statistics. By default, COLUMN=BOTH.

RISKDIFF < (*options*) >

requests risk statistics for 2 × 2 tables. By default, risk statistics include the row 1 risk (proportion of row 1), row 2 risk, overall risk, and risk difference (row 1 risk – row 2 risk), together with their standard errors and confidence limits. For more information, see the section "Risks and Risk Difference" on page 9850.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

You can specify the following *options*:

COLUMN=1 | 2 | BOTH
specifies the table column for which to compute the risk statistics. By default, COLUMN=BOTH.

ONLY
displays only the risk difference together with its standard error and confidence limits. This option suppresses display of the row 1 risk, row 2 risk, and overall risk.

ROW < (*row-options*) >
displays the row percentage (estimated proportion of the row total) for each cell in a two-way table. This option also provides the standard errors of the row percentages. For more information, see the section "Row and Column Proportions" on page 9840. This option has no effect on one-way frequency tables.

To display the row percentages and statistics in separate one-way tables for each row (instead of in a two-way table), you can specify the DOMAIN=ROW option.

You can specify the following *row-options*:

CL < (*cl-options*)| =NO >
requests confidence limits for the row percentages (proportions). For more information, see the section "Confidence Limits for Proportions" on page 9842.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

You can specify the CL=NO *row-option* to suppress the display of confidence limits for row percentages (while displaying confidence limits for other percentages). By default, the overall CL option controls display of confidence limits for all percentages (overall, row, and column percentages).

You can specify *cl-options* to control the confidence limit computations. The ROW *cl-options* are the same *cl-options* that are available for the overall CL option, which requests confidence limits for all percentages (table cell, row, and column percentages). For a description of the *cl-options*, see the overall CL option. For more information, see the section "Confidence Limits for Proportions" on page 9842. In a single TABLES statement, PROC SURVEYFREQ accepts only one value of each *cl-option*; for example, you cannot specify one TYPE= value in the overall CL option and a different TYPE= value in the ROW(CL) option.

CV < =NO >
displays coefficients of variation for the row percentages. For more information, see the section "Coefficient of Variation" on page 9847.

You can specify the CV=NO *row-option* to suppress the display of coefficients of variation for row percentages (while displaying coefficients of variation for other percentages). By default, the overall CV option controls display of coefficients of variation for all percentages (overall, row, and column percentages).

DEFF

displays design effects for the row percentages. For more information, see the section "Design Effect" on page 9847.

NOSTD

suppresses the display of standard errors for the row percentages.

VAR < =NO >

displays variance estimates for the row percentages. For more information, see the section "Proportions" on page 9839.

You can specify the VAR=NO *row-option* to suppress the display of variances estimates for row percentages (while displaying variances estimates for other percentages). By default, the overall VAR option controls display of variance estimates for all percentages (overall, row, and column percentages).

SENSPEC < (*options*) >

requests estimates of sensitivity, specificity, accuracy, positive predictive value, and negative predictive value for 2×2 tables. The "Sensitivity and Specificity" table provides these estimates together with their standard errors and confidence limits. For more information, see the section "Sensitivity and Specificity" on page 9849.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

You can specify the following *options*:

CLTYPE

uses the confidence limit type that you specify in the CL(TYPE=) option to compute confidence limits for statistics in the "Sensitivity and Specificity" table. Available confidence limit types include Agresti-Coull, Clopper-Pearson (exact), Jeffreys, logit, uniform, Wald, and Wilson (score). For more information, see the description of the CL option and the section "Confidence Limits for Proportions" on page 9842.

By default, PROC SURVEYFREQ computes Wald-type ("linear") confidence limits for the SENSPEC statistics by using the variance estimates that are based on the sample design.

If you specify the SENSPEC(CLTYPE) option together with a modified confidence limit type (Agresti-Coull, Clopper-Pearson, Jeffreys, uniform, or Wilson) in the CL(TYPE=) option, the CL(ADJUST=), CL(TRUNCATE=), and CL(VARDEF=N) options also apply to the SENSPEC confidence limits.

The SENSPEC(CLTYPE) option is equivalent to the CL(SENSPEC) option.

NOTE< =YES | NO >

controls the footnote that identifies the reference level (true positives) in the "Sensitivity and Specificity" table. NOTE=NO suppresses this footnote and NOTE< =YES > displays it. By default, PROC SURVEYFREQ displays the footnote only when you specify the reference level by using the REFCELL=, REFCOLUMN=, and/or REFROW= options.

REFCELL=*number*

specifies the 2 × 2 table cell to use as the true-positive classification (positive row variable level and positive column variable level) in the SENSPEC computations. For more information, see the section "Sensitivity and Specificity" on page 9849.

You can specify the true-positive cell *number* as 1, 2, 3, or 4; these numbers correspond to the cells of the 2 × 2 crosstabulation table. By default, REFCELL=1, which uses table cell (1,1) as the true-positive cell (and the corresponding row 1 and column 1 as the positive row and column levels, respectively). REFCELL=2, 3, and 4 use table cells (1,2), (2,1), and (2,2), respectively, as the true-positive cells.

You cannot specify this option together with the REFROW= or REFCOLUMN= option. The true-positive cell *number* determines the corresponding positive row and column levels.

REFCOLUMN=*number* | '*level-value*'

specifies the column level to use as the positive column level in the SENSPEC computations. For more information, see the section "Sensitivity and Specificity" on page 9849.

You can specify the positive column level *number* as 1 or 2, which is the order in which the level appears in the 2 × 2 crosstabulation table. Or you can specify the positive column *level-value*, which is the formatted value of the level. The *level-value* must be enclosed in single quotes. By default, REFCOLUMN=1.

REFROW=*number* | '*level-value*'

specifies the row level to use as the positive row level in the SENSPEC computations. For more information, see the section "Sensitivity and Specificity" on page 9849.

You can specify the positive row level *number* as 1 or 2, which is the order in which the level appears in the 2 × 2 crosstabulation table. Or you can specify the positive row *level-value*, which is the formatted value of the level. The *level-value* must be enclosed in single quotes. By default, REFROW=1.

TESTP=(*values***)**

specifies null hypothesis proportions (test percentages) for chi-square tests for one-way tables (goodness-of-fit tests). You can separate *values* with blanks or commas, and you must enclose the list of values in parentheses. Specify *values* in probability form as numbers between 0 and 1, where the proportions sum to 1. Or specify *values* in percentage form as numbers between 0 and 100, where the percentages sum to 100. PROC SURVEYFREQ treats the value 1 as the percentage form 1%. The number of TESTP= values must equal the number of variable levels in the one-way table. List these values in the same order in which the corresponding variable levels appear in the output.

When you specify the TESTP= option, PROC SURVEYFREQ displays the specified test percentages in the one-way frequency table. The TESTP= option has no effect on two-way tables.

PROC SURVEYFREQ uses the TESTP= values for the one-way Rao-Scott chi-square test (CHISQ) and for the one-way Rao-Scott likelihood ratio chi-square test (LRCHISQ). For more information, see the sections "Rao-Scott Chi-Square Test" on page 9858 and "Rao-Scott Likelihood Ratio Chi-Square Test" on page 9864.

VAR

displays the variance estimate for each percentage in the crosstabulation table. For more information, see the section "Proportions" on page 9839. By default, PROC SURVEYFREQ displays the standard errors of the percentages.

To display variance estimates only for row or column percentages, specify the ROW(VAR) or COLUMN(VAR) option, respectively.

VARWT

displays the variance estimate for each weighted frequency, or estimated total, in the crosstabulation table. For more information, see the section "Totals" on page 9837. By default, PROC SURVEYFREQ displays the standard errors of the weighted frequencies.

WCHISQ

requests the Wald chi-square test for two-way tables. For more information, see the section "Wald Chi-Square Test" on page 9865.

WLLCHISQ

requests the Wald log-linear chi-square test for two-way tables. For more information, see the section "Wald Log-Linear Chi-Square Test" on page 9867.

WTFREQ

displays totals (weighted frequencies) and their standard errors when you do not specify a WEIGHT or REPWEIGHTS statement. By default, PROC SURVEYFREQ displays the weighted frequencies only when you specify a WEIGHT or REPWEIGHTS statement. When you do not specify a WEIGHT or REPWEIGHTS statement, PROC SURVEYFREQ assigns all observations a weight of one.

WTKAPPA < (*options*) >

requests the weighted kappa coefficient with its standard error and confidence limits. Weighted kappa coefficients can be computed for square two-way tables, where the number of rows is the same as the number of columns. For 2×2 tables, the weighted kappa coefficient equals the simple kappa coefficient, and PROC SURVEYFREQ displays only the simple kappa coefficient. For more information, see the section "Weighted Kappa Coefficient" on page 9856.

You can specify the confidence level in the ALPHA= option. By default, ALPHA=0.05, which produces 95% confidence limits.

Weighted kappa coefficients are available for replication variance estimation methods (which you can request by specifying VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE or by including a REPWEIGHTS statement); weighted kappa coefficients are not available for the Taylor series method (VARMETHOD=TAYLOR).

Weighted kappa coefficients are computed by using agreement weights that reflect the relative agreement between pairs of variable levels. Agreement weights are not the same as *sampling weights*, which you provide by specifying the WEIGHT statement. PROC SURVEYFREQ uses the sampling weights to compute both the simple kappa and weighted kappa coefficients. For more information, see the section "Weighted Kappa Coefficient" on page 9856.

You can specify the following *options*:

DETAILS

WTKAPDETAILS

displays the observed agreement and chance-expected agreement components of the weighted kappa coefficient. For more information, see the section "Weighted Kappa Coefficient" on page 9856.

PRINTKWTS

displays the agreement weights that PROC SURVEYFREQ uses to compute the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. By default, PROC SURVEYFREQ computes these weights by using the Cicchetti-Allison form. If you specify the WT=FC option, the procedure uses the Fleiss-Cohen form of agreement weights. For more information, see the section "Weighted Kappa Coefficient" on page 9856.

WT=FC

requests Fleiss-Cohen agreement weights for the weighted kappa computation. By default, PROC SURVEYFREQ uses Cicchetti-Allison agreement weights to compute the weighted kappa coefficient. Agreement weights reflect the relative agreement between pairs of variable levels. For more information, see the section "Weighted Kappa Coefficient" on page 9856.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement names the *variable* that contains the sampling weights. This variable must be numeric, and the sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, then the procedure omits that observation from the analysis. For more information, see the section "Missing Values" on page 9825. If you specify more than one WEIGHT statement, the procedure uses only the first WEIGHT statement and ignores the rest.

If you do not specify a WEIGHT statement but provide replicate weights by specifying a REPWEIGHTS statement, PROC SURVEYFREQ uses the average of each observation's replicate weights as the observation's weight.

If you do not specify a WEIGHT statement or a REPWEIGHTS statement, PROC SURVEYFREQ assigns all observations a weight of one.

Details: SURVEYFREQ Procedure

Specifying the Sample Design

PROC SURVEYFREQ produces tables and statistics that are based on the sample design used to obtain the survey data. PROC SURVEYFREQ can be used for single-stage or multistage designs, with or without stratification, and with or without unequal weighting. To analyze your survey data with PROC SURVEYFREQ, you need to provide sample design information for the procedure. This information can include design strata, clusters, and sampling weights. You can provide sample design information by using the STRATA, CLUSTER, and WEIGHT statements, and by specifying the RATE= or TOTAL= option in the PROC SURVEYFREQ statement.

If you provide replicate weights for variance estimation, you do not need to specify a STRATA or CLUSTER statement. Otherwise, you should specify STRATA and CLUSTER statements whenever your design includes stratification and clustering.

When there are clusters (PSUs) in the sample design, the procedure estimates the variance by using the PSUs, as described in the section "Statistical Computations" on page 9836. For a multistage sample design, the variance estimation depends only on the first stage of the sample design. Therefore, the required input includes only first-stage cluster (PSU) and first-stage stratum identification. You do not need to input design information about any additional stages of sampling.

Stratification

If your sample design is stratified at the first stage of sampling, use the STRATA statement to name the variables that form the strata. The combinations of categories of STRATA variables define the strata in the sample, where strata are nonoverlapping subgroups that were sampled independently. If your sample design has stratification at multiple stages, you should identify only the first-stage strata in the STRATA statement.

If you use a REPWEIGHTS statement to provide replicate weights for variance estimation, you do not need to specify a STRATA statement. Otherwise, you should specify a STRATA statement whenever your design includes stratification. If you do not specify a STRATA statement or a REPWEIGHTS statement, then PROC SURVEYFREQ assumes there is no stratification at the first stage.

Clustering

If your sample design selects clusters at the first stage of sampling, use the CLUSTER statement to name the variables that identify the first-stage clusters, which are also called primary sampling units (PSUs). The combinations of categories of CLUSTER variables define the clusters in the sample. If there is a STRATA statement, clusters are nested within strata. If your sample design has clustering at multiple stages, you should specify only the first-stage clusters (PSUs) in the CLUSTER statement. PROC SURVEYFREQ assumes that each cluster that is defined by the CLUSTER statement variables represents a PSU in the sample, and that each observation belongs to one PSU.

If you use a REPWEIGHTS statement to provide replicate weights for variance estimation, you do not need to specify a CLUSTER statement. Otherwise, you should specify a CLUSTER statement whenever your design includes clustering at the first stage of sampling. If you do not specify a CLUSTER statement, then PROC SURVEYFREQ treats each observation as a PSU.

Weighting

If your sample design includes unequal weighting, use the WEIGHT statement to name the variable that contains the sampling weights. Sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, the procedure omits that observation from the analysis. For more information, see the section "Missing Values" on page 9825.

If you do not specify a WEIGHT statement but include a REPWEIGHTS statement, PROC SURVEYFREQ uses the average of each observation's replicate weights as the observation's weight. If you do not specify a WEIGHT statement or a REPWEIGHTS statement, PROC SURVEYFREQ assigns all observations a weight of one.

Population Totals and Sampling Rates

To include a finite population correction (*fpc*) in Taylor series or bootstrap variance estimation, you can input either the sampling rate or the population total by using the RATE= or TOTAL= option in the PROC SURVEYFREQ statement. (You cannot specify both of these options in the same PROC SURVEYFREQ statement.) The RATE= and TOTAL= options apply only to Taylor series and bootstrap variance estimation. PROC SURVEYFREQ does not use a finite population correction in BRR, jackknife, or replicate weight variance estimation.

If you do not specify the RATE= or TOTAL= option, the variance estimation does not include a finite population correction. For fairly small sampling fractions, it is appropriate to ignore this correction. For more information, see Cochran (1977) and Kish (1965).

If your design has multiple stages of selection and you are specifying the RATE= option, you should input the first-stage sampling rate, which is the ratio of the number of PSUs in the sample to the total number of PSUs in the study population. If you are specifying the TOTAL= option for a multistage design, you should input the total number of PSUs in the study population.

For a nonstratified sample design, or for a stratified sample design with the same sampling rate or the same population total in all strata, you can use the RATE=*value* or TOTAL=*value* option. If your sample design is stratified with different sampling rates or population totals in different strata, use the RATE=*SAS-data-set* or TOTAL=*SAS-data-set* option to name a SAS data set that contains the stratum sampling rates or totals. This data set is called a *secondary data set*, as opposed to the *primary data set* that you specify in the DATA= option.

The secondary data set must contain all the stratification variables listed in the STRATA statement and all the variables in the BY statement. Furthermore, the BY groups must appear in the same order as in the primary data set. If there are formats that are associated with the STRATA variables and the BY variables, then the formats must be consistent in the primary and the secondary data sets. If you specify the TOTAL=*SAS-data-set* option, the secondary data set must have a variable named _TOTAL_ that contains the stratum population totals. If you specify the RATE=*SAS-data-set* option, the secondary data set must have a variable named _RATE_ that contains the stratum sampling rates. If the secondary data set contains more than one observation for any one stratum, the procedure uses the first value of _TOTAL_ or _RATE_ for that stratum and ignores the rest.

The *value* in the RATE= option or the values of _RATE_ in the secondary data set must be nonnegative numbers. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC SURVEYFREQ converts that number to a proportion. The procedure treats the value 1 as 100% instead of 1%.

If you specify the TOTAL=*value* option, *value* must not be less than the sample size. If you provide stratum population totals in a secondary data set, these values must not be less than the corresponding stratum sample sizes.

Domain Analysis

You can perform domain analysis in PROC SURVEYFREQ by using multiway table requests, the ROW option, the COLUMN option, or the DOMAIN=ROW option. *Domain analysis* refers to the computation of statistics for domains (subpopulations) in addition to the computation of statistics for the entire study population. Formation of subpopulations can be unrelated to the sample design and thus the domain sample sizes can actually be random variables. Domain analysis takes this variability into account by using the entire sample to estimate the variance of domain estimates. Domain analysis is also known as subgroup analysis, subpopulation analysis, and subdomain analysis. For more information about domain analysis, see Lohr (2022), Cochran (1977), Fuller et al. (1989).

You can perform domain analysis by including the domain variable(s) in a multiway table request. For example, you can specify DOMAIN * A * B in a TABLES statement to produce separate two-way tables of A by B for each level of DOMAIN. If your domains are formed by more than one variable, you can specify DomainVariable_1 * DomainVariable_2 * A * B, for example, to obtain two-way tables of A by B for each domain formed by the combinations of levels of DomainVariable_1 and DomainVariable_2. For an example of domain analysis, see Example 118.2.

If you specify a two-way table request in a TABLES statement (for example, DOMAIN * A), the values of the variable DOMAIN form the table rows and the values of the variable A form the table columns. The two-way table displays the levels of the variable A within each level of the row variable DOMAIN. You can specify the ROW option in the TABLES statement to obtain the row percentages, standard errors, confidence limits, and other statistics. This provides the one-way distribution of A in each domain (level of the variable DOMAIN). Alternatively, you can display a separate one-way table for each row variable level by specifying the DOMAIN=ROW option. This option enables you to produce one-way chi-square tests for the row-level domains by also specifying the CHISQ option.

A domain analysis (where the variance computations are based on the entire sample) is not the same as the analysis that you obtain by using a BY statement or subsetting the input data set; a BY statement provides completely separate analyses of the BY groups. You can use a BY statement to analyze subgroups of the data, but it is critical to note that this does *not* produce a valid domain analysis; the BY statement is appropriate only when the number of units in each subgroup is known with certainty. For example, you can use a BY statement to obtain stratum level estimates when the stratum sample sizes are fixed. But when the subgroup sample sizes are not fixed, you should perform domain analysis by including the domain variables in the TABLES statement request.

Missing Values

WEIGHT Variable

If an observation has a missing or nonpositive value for the WEIGHT variable, PROC SURVEYFREQ excludes that observation from the analysis.

REPWEIGHTS Variables

If you provide replicate weights by specifying a REPWEIGHTS statement, all REPWEIGHTS variable values must be nonmissing. Similarly, if you provide jackknife coefficients by specifying the JKCOEFS= option in the REPWEIGHTS statement, all values of the JKCoefficient variable must be nonmissing. If any replicate weight or jackknife coefficient is missing, PROC SURVEYFREQ does not perform the analysis.

STRATA and CLUSTER Variables

If an observation has a missing value for any STRATA or CLUSTER variable, PROC SURVEYFREQ excludes that observation from the analysis unless you specify the MISSING option in the PROC SURVEYFREQ statement. If you specify the MISSING option, PROC SURVEYFREQ treats missing values as a valid (nonmissing) category for all categorical variables, which include STRATA, CLUSTER, and TABLES variables.

TABLES Variables

If an observation has a missing value for any variable in the TABLES request, PROC SURVEYFREQ excludes that observation from the crosstabulation table (and all associated analyses) unless you specify the MISSING or NOMCAR option in the PROC SURVEYFREQ statement. When the procedure excludes observations with missing values from a table, it displays the total frequency of missing observations below the table.

If you specify the MISSING option, PROC SURVEYFREQ treats missing values as a valid (nonmissing) level for each TABLES variable. The procedure displays these levels in the crosstabulation table and includes them in the computation of totals, percentages, and all other table statistics.

If you specify the NOMCAR option in the PROC SURVEYFREQ statement for Taylor series variance estimation, the procedure includes observations with missing values of TABLES variables in the variance computations. The NOMCAR option does not display missing levels in the crosstabulation table or compute percentages and totals for missing levels.

The NOMCAR Option

The NOMCAR option in the PROC SURVEYFREQ statement includes observations with missing values of TABLES variables in the variance computations as *not missing completely at random* (NOMCAR) for Taylor series variance estimation. By default, observations are completely excluded from the analysis if they have missing values for any of the variables in the current TABLES request. This default treatment is based on the assumption that the values are *missing completely at random* (MCAR), and assumes that the analysis results should not be substantially different between the missing and nonmissing groups. For more information, see the section "Analysis Considerations" on page 9827.

When you specify the NOMCAR option, PROC SURVEYFREQ computes variance estimates by analyzing the nonmissing values as a domain (subpopulation), where the entire population includes both nonmissing and missing domains.

The NOMCAR option has no effect when you specify the MISSING option, which treats missing values as a valid nonmissing level. The NOMCAR option does not affect the inclusion of observations with missing values of the WEIGHT, CLUSTER, or STRATA variables. Observations with missing values of the WEIGHT variable are always excluded from the analysis. Observations with missing values of the CLUSTER or STRATA variables are excluded unless you specify the MISSING option.

The NOMCAR option applies only to Taylor series variance estimation VARMETHOD=TAYLOR. The replication methods (which you can request by specifying VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or the REPWEIGHTS statement) do not use the NOMCAR option.

Degrees of Freedom

PROC SURVEYFREQ computes the degrees of freedom to obtain the t distribution's percentile for confidence limits for proportions, totals, and other statistics. The procedure also uses the degrees of freedom for the F statistics in the Rao-Scott and Wald chi-square tests. The degrees of freedom computation depends on the sample design and the variance estimation method. For more information, see the section "Degrees of Freedom" on page 9846. Missing values can affect the degrees of freedom computation.

Taylor Series Variance Estimation

The degrees of freedom can depend on the number of clusters, the number of strata, and the number of observations. For Taylor series variance estimation, these numbers are based on the observations included in the analysis of the individual table. These numbers do not count observations that are excluded from the table because of missing values. If all values in a stratum are excluded from the analysis of a table as missing values, then that stratum is called an *empty stratum*. Empty strata are not counted in the total number of strata for the table. Similarly, empty clusters and missing observations are not included in the total counts of clusters and observations that are used to compute the degrees of freedom for the analysis.

If you specify the MISSING option, missing values are treated as valid nonmissing levels and are included in computing degrees of freedom. If you specify the NOMCAR option for Taylor series variance estimation, observations with missing values of the TABLES variables are included in computing degrees of freedom.

Replicate Variance Estimation

By default for replicate variance estimation, PROC SURVEYFREQ computes the degrees of freedom by using all valid observations in the input data set. An observation is valid if it has a positive value of the WEIGHT variable and nonmissing values of the STRATA and CLUSTER variables unless you specify the MISSING option. For more information, see the section "Data Summary Table" on page 9869.

If you specify the DFADJ *method-option*, PROC SURVEYFREQ computes the degrees of freedom by using only the nonmissing observations for the current table request. This excludes any empty strata or clusters that occur when observations are excluded because of missing values of a TABLES variable.

Table Summary Output Data Set

For each table request, PROC SURVEYFREQ produces a nondisplayed ODS table, "Table Summary," which contains the number of (nonmissing) observations, strata, and clusters that are included in the analysis of the individual table. If there are missing observations, empty strata, or empty clusters excluded from the analysis, the "Table Summary" data set also contains this information. If you request any confidence limits or chi-square tests for the table, which require degrees of freedom, the "Table Summary" data set provides the degrees of freedom.

Because of missing values, the number of observations used for an individual table analysis can differ from the number of valid observations in the input data set, which is reported in the "Data Summary" table. Similarly, a difference can also occur for the number of clusters or strata. For more information about the "Table Summary" output data set, see Example 118.3.

If you specify the NOMCAR option for Taylor series variance estimation, the "Table Summary" data set reflects all observations used for variance estimation, which includes those observations with missing values of the TABLES variables.

Analysis Considerations

If you have missing values in your survey data for any reason (such as nonresponse), this can compromise the quality of your survey results. An observation without missing values is called a *complete respondent*, and an observation with missing values is called an *incomplete respondent*. If the complete respondents are different from the incomplete respondents with regard to a survey effect or outcome, then survey estimates will be biased and will not accurately represent the survey population. There are a variety of techniques in sample design and survey operations that can reduce nonresponse. After data collection is complete, you can use imputation to replace missing values with acceptable values, and you can use sampling weight adjustments to compensate for nonresponse. You should complete this data preparation and adjustment before you analyze your data with PROC SURVEYFREQ. For more information, see Cochran (1977), Kalton and Kasprzyk (1986), and Brick and Kalton (1996).

Variance Estimation

PROC SURVEYFREQ provides the Taylor series linearization method and replication (resampling) methods to estimate sampling errors of estimators that are based on complex sample designs. For more information, see Lohr (2022), Fuller (2009), Wolter (2007), Särndal, Swensson, and Wretman (1992), Lee, Forthofer, and Lorimor (1989), and Kalton (1983). You can use the VARMETHOD= option in the PROC SURVEYFREQ statement to specify the variance estimation method; by default, PROC SURVEYFREQ uses the Taylor series method.

Taylor Series Variance Estimation

The Taylor series linearization method can be used to estimate standard errors of proportions and other statistics for crosstabulation tables. For sample survey data, the proportion estimator is a ratio estimator formed from estimators of totals. For example, to estimate the proportion in a crosstabulation table cell, the procedure uses the ratio of the estimator of the cell total frequency to the estimator of the overall population total, where these totals are linear statistics computed from the survey data. The Taylor series expansion method obtains a first-order linear approximation for the ratio estimator and then uses the variance estimate

for this approximation to estimate the variance of the estimate itself (Woodruff 1971; Fuller 1975). For more information about Taylor series variance estimation for sample survey data, see Lohr (2022), Wolter (2007), Särndal, Swensson, and Wretman (1992), and Lee, Forthofer, and Lorimor (1989).

When there are clusters (PSUs) in the sample design, the Taylor series method estimates variance from the variance among PSUs. When the design is stratified, the procedure combines stratum variance estimates to compute the overall variance estimate. For a multistage sample design, the variance estimation depends only on the first stage of the sample design. Therefore, the required input includes only first-stage cluster (PSU) and first-stage stratum identification. You do not need to input design information about any additional stages of sampling. This variance estimation method assumes that the first-stage sampling fraction is small, or the first-stage sample is drawn with replacement, as it often is in practice.

For more information about Taylor series variance estimation, see the sections "Proportions" on page 9839, "Row and Column Proportions" on page 9840, "Risks and Risk Difference" on page 9850, and "Odds Ratio and Relative Risks" on page 9852.

Replication Variance Estimation

Replication methods for variance estimation draw multiple replicates (subsamples) from the full sample by following a specific resampling scheme. Commonly used resampling schemes include *bootstrap*, *balanced repeated replication* (BRR), and *jackknife*. PROC SURVEYFREQ estimates the parameter of interest (a proportion, total, odds ratio, or other statistic) from each replicate, and then uses the variability among replicate estimates to estimate the overall variance of the parameter estimate. For more information, see Wolter (1985) and Lohr (2022).

The balanced repeated replication (BRR) variance estimation method requires a stratified sample design with two PSUs in each stratum. Each replicate is created by deleting one PSU per stratum according to the corresponding Hadamard matrix and adjusting the original weights of the remaining PSUs. The adjusted weights are called *replicate weights*. PROC SURVEYFREQ also provides Fay's BRR method, which is a modification of the BRR method. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 9830 and "Fay's BRR Method" on page 9832.

The jackknife method deletes one PSU at a time from the full sample to create replicates and adjusts the original weights to obtain replicate weights. The total number of replicates is the number of PSUs. If the sample design is stratified, each stratum must contain at least two PSUs, and the jackknife method is applied separately within each stratum. For more information, see the section "Jackknife Method" on page 9834.

Instead of having PROC SURVEYFREQ construct replicate weights for the analysis, you can input your own replicate weights by using a REPWEIGHTS statement. This can be useful if you need to do multiple analyses with the same set of replicate weights, or if you have access to replicate weights without complete design information. For more information, see the REPWEIGHTS statement.

Bootstrap Method

The VARMETHOD=BOOTSTRAP option requests the bootstrap method for variance estimation. This method can be used for stratified sample designs and for designs that have no stratification. If your design is stratified, the bootstrap method requires at least two PSUs in each stratum. You can provide bootstrap replicate weights for the analysis by using a REPWEIGHTS statement, or the procedure can construct bootstrap replicate weights for the analysis. PROC SURVEYFREQ estimates the parameter of interest (a proportion, total, odds ratio, or other statistic) from each replicate, and then uses the variability among replicate estimates to estimate the overall variance of the parameter estimate.

This bootstrap method for complex survey data is similar to the method of Rao, Wu, and Yue (1992) and is also known as the bootstrap weights method (Mashreghi, Haziza, and Léger 2016). For more information, see Lohr (2022, Section 9.3.3), Wolter (2007, Chapter 5), Beaumont and Patak (2012), Fuller (2009, Section 4.5), and Shao and Tu (1995, Section 6.2.4). McCarthy and Snowden (1985), Rao and Wu (1988), Sitter (1992b), and Sitter (1992a) provide several adjusted bootstrap variance estimators that are consistent for complex survey data. The naive bootstrap variance estimator that is suitable for infinite populations is not consistent for complex survey data.

Replicate Weight Construction

If you do not provide replicate weights by using a REPWEIGHTS statement, PROC SURVEYFREQ constructs bootstrap replicate weights for the analysis. The procedure selects replicate bootstrap samples by with-replacement random sampling of PSUs within strata. You can specify the number of bootstrap replicates in the REPS= *method-option*; by default, the number of replicates is 250. (Increasing the number of replicates can improve the estimation precision but also increases the computation time.) You can specify the bootstrap sample sizes m_h in the MH= *method-option*; by default, $m_h = n_h - 1$, where n_h is the number of PSUs in stratum h.

In each replicate sample, the original sampling weights of the selected units are adjusted to reflect the full sample. These adjusted weights are the bootstrap replicate weights. In replicate r, the bootstrap replicate weight for observation j in PSU i in stratum h is computed as

$$w_{hij}^{(r)} = w_{hij}\left\{1 - \sqrt{(1-f_h)m_h/(n_h-1)} + \sqrt{(1-f_h)m_h/(n_h-1)}(n_h/m_h)k_{hi}^{(r)}\right\}$$

where $k_{hi}^{(r)}$ is the number of times PSU i is selected in replicate sample r, and f_h is the sampling fraction in stratum h.

You can use the OUTWEIGHTS= *method-option* to store the bootstrap replicate weights in a SAS data set. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weight Output Data Set" on page 9868. You can provide these replicate weights to the procedure for subsequent analyses by using a REPWEIGHTS statement.

Variance Estimation

Let θ denote the population parameter to be estimated—for example, a proportion, total, odds ratio, or other statistic. Let $\hat{\theta}$ denote the estimate of θ from the full sample, and let $\hat{\theta}_r$ denote the estimate from the rth bootstrap replicate, which is computed by using the bootstrap replicate weights. The bootstrap variance estimate for $\hat{\theta}$ is computed as

$$\widehat{V}(\hat{\theta}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\theta}_r - \hat{\theta}\right)^2$$

where R is the total number of replicates and α_r is the coefficient for replicate r.

By default, $\alpha_r = 1/R$ for the bootstrap method. If you provide bootstrap weights in the REPWEIGHTS statement, you can also provide replicate coefficients in the REPCOEFS= option.

If you specify the CENTER=REPLICATES *method-option*, the bootstrap variance estimate is computed as

$$\widehat{V}(\hat{\theta}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\theta}_r - \bar{\theta}\right)^2$$

where $\bar{\theta}$ is the average of the replicate estimates and is computed as follows:

$$\bar{\theta} = \frac{1}{R}\sum_{r=1}^{R}\hat{\theta}_r$$

If a parameter cannot be estimated from one or more replicates, the variance estimate is computed by using those replicates from which the parameter can be estimated. For example, suppose the parameter is a column proportion—the proportion of column j for table cell (i, j). If a replicate r contains no observations in column j, then the column j proportion is not estimable from replicate r. In this case, the bootstrap variance estimate is computed as

$$\widehat{V}(\hat{\theta}) = \frac{R}{R'}\sum_{r=1}^{R'}\alpha_r\left(\hat{\theta}_r - \hat{\theta}\right)^2$$

where the summation is over the replicates for which the parameter θ is estimable and where R' is the number of those replicates.

Balanced Repeated Replication (BRR) Method

When you specify the VARMETHOD=BRR option, PROC SURVEYFREQ uses balanced repeated replication (BRR) for variance estimation. The BRR variance estimation method requires a stratified sample design with two PSUs in each stratum. You can provide replicate weights for BRR variance estimation by using a REPWEIGHTS statement, or the procedure can construct replicate weights for the analysis. PROC SURVEYFREQ estimates the parameter of interest (a proportion, total, odds ratio, or other statistic) from each replicate, and then uses the variability among replicate estimates to estimate the overall variance of the parameter estimate. For more information about BRR variance estimation, see Wolter (1985) and Lohr (2022).

Replicate Weight Construction

If you do not provide replicate weights by using a REPWEIGHTS statement, PROC SURVEYFREQ constructs replicates based on the stratified design with two PSUs in each stratum. This section describes replicate construction by the traditional BRR method. If you specify the FAY *method-option* for VARMETHOD=BRR, the procedure uses Fay's modified BRR method, which is described in the section "Fay's BRR Method" on page 9832.

With the traditional BRR method, each replicate is obtained by deleting one PSU per stratum according to the corresponding Hadamard matrix of dimension R, where R is the number of replicates. The number of replicates is the smallest multiple of 4 that is greater than the number of strata H. Alternatively, you can specify the number of replicates with the REPS= *method-option*. If a Hadamard matrix cannot be constructed for the REPS= value that you specify, the value is increased until a Hadamard matrix of that dimension can be constructed. Therefore, it is possible for the actual number of replicates used to be larger than the REPS= value that you specify.

You can provide a Hadamard matrix for BRR replicate construction by using the HADAMARD= *method-option*. Otherwise, PROC SURVEYFREQ generates an appropriate Hadamard matrix. For more information, see the section "Hadamard Matrix" on page 9832. You can display the Hadamard matrix by specifying the PRINTH *method-option*.

PROC SURVEYFREQ constructs replicates by using the first H columns of the $R \times R$ Hadamard matrix, where H denotes the number of strata. The rth replicate ($r = 1, 2, \ldots, R$) is drawn from the full sample according to the rth row of the Hadamard matrix as follows:

- If element (r, h) of the Hadamard matrix is 1, then the first PSU of stratum h is included in the rth replicate, and the second PSU of stratum h is excluded.
- If element (r, h) of the Hadamard matrix is –1, then the second PSU of stratum h is included in the rth replicate, and the first PSU of stratum h is excluded.

For the PSUs included in replicate r, the original weights are doubled to form the replicate r weights. For the PSUs not included in replicate r, the replicate r weights equal 0. You can use the OUTWEIGHTS= *method-option* to store the replicate weights in a SAS data set. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weight Output Data Set" on page 9868. You can provide these replicate weights to the procedure for subsequent analyses by using a REPWEIGHTS statement.

Variance Estimation

Let θ denote the population parameter to be estimated—for example, a proportion, total, odds ratio, or other statistic. Let $\hat{\theta}$ denote the estimate of θ from the full sample, and let $\hat{\theta}_r$ denote the estimate from the rth BRR replicate, which is computed by using the replicate weights. The BRR variance estimate for $\hat{\theta}$ is computed as

$$\widehat{V}(\hat{\theta}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\theta}_r - \hat{\theta}\right)^2$$

where R is the total number of replicates.

If you specify the CENTER=REPLICATES *method-option*, the BRR variance estimate is computed as

$$\widehat{V}(\hat{\theta}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\theta}_r - \bar{\theta}\right)^2$$

where $\bar{\theta}$ is the average of the replicate estimates and is computed as follows:

$$\bar{\theta} = \frac{1}{R}\sum_{r=1}^{R}\hat{\theta}_r$$

If a parameter cannot be estimated from one or more replicates, the variance estimate is computed by using those replicates from which the parameter can be estimated. For example, suppose the parameter is a column proportion—the proportion of column j for table cell (i, j). If a replicate r contains no observations in column j, then the column j proportion is not estimable from replicate r. In this case, the BRR variance estimate is computed as

$$\widehat{V}(\hat{\theta}) = \frac{1}{R'}\sum_{r=1}^{R'}\left(\hat{\theta}_r - \hat{\theta}\right)^2$$

where the summation is over the replicates for which the parameter θ is estimable and where R' is the number of those replicates.

Hadamard Matrix

PROC SURVEYFREQ uses a Hadamard matrix to construct replicates for BRR variance estimation. You can provide a Hadamard matrix for replicate construction in the HADAMARD= *method-option* for VARMETHOD=BRR. Otherwise, PROC SURVEYFREQ generates an appropriate Hadamard matrix. You can display the Hadamard matrix by specifying the PRINTH *method-option*.

A Hadamard matrix $\mathbf{A}$ of dimension R is a square matrix that has all elements equal to 1 or –1. A Hadamard matrix must satisfy the requirement that $\mathbf{A}'\mathbf{A} = R\,\mathbf{I}$, where $\mathbf{I}$ is an identity matrix. The dimension of a Hadamard matrix must equal 1, 2, or a multiple of 4.

For example, the following matrix is a Hadamard matrix of dimension $k = 8$:

$$
\begin{array}{rrrrrrrr}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{array}
$$

For BRR replicate construction, the dimension of the Hadamard matrix must be at least H, where H denotes the number of first-stage strata in your design. If a Hadamard matrix of a given dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a SAS data set in the HADAMARD=*SAS-data-set method-option*. You must ensure that the matrix that you provide is actually a Hadamard matrix; PROC SURVEYFREQ does not check the validity of your Hadamard matrix.

For information about how the Hadamard matrix is used to construct replicates for BRR variance estimation, see the sections "Balanced Repeated Replication (BRR) Method" on page 9830 and "Fay's BRR Method" on page 9832.

Fay's BRR Method

When you specify the FAY *method-option* for VARMETHOD=BRR, PROC SURVEYFREQ uses Fay's BRR method, which is a modification of the traditional BRR variance estimation method. Fay's BRR method also requires a stratified sample design with two PSUs in each stratum. You can provide replicate weights by using a REPWEIGHTS statement, or the procedure can construct replicate weights for the analysis. PROC SURVEYFREQ estimates the parameter of interest (a proportion, total, odds ratio, or other statistic) from each replicate, and then uses the variability among replicate estimates to estimate the overall variance of the parameter estimate.

Replicate Weight Construction

If you do not provide replicate weights by using a REPWEIGHTS statement, PROC SURVEYFREQ constructs replicates based on the stratified design with two PSUs in each stratum. As for traditional BRR, the number of replicates R is the smallest multiple of 4 that is greater than the number of strata H, or you can specify the number of replicates with the REPS= *method-option*. You can provide a Hadamard matrix for replicate construction in the HADAMARD= *method-option*, or PROC SURVEYFREQ generates an appropriate Hadamard matrix.

The traditional BRR method constructs half-sample replicates by deleting one PSU per stratum according to the Hadamard matrix and doubling the original weights to form replicate weights. Fay's BRR method adjusts the original weights by a coefficient ϵ, where $0 \leq \epsilon < 1$. You can specify the value of ϵ with the FAY= *method-option*. If you do not specify the value of ϵ, PROC SURVEYFREQ uses $\epsilon = 0.5$ by default. For information about the value of the Fay coefficient, see Judkins (1990) and Rao and Shao (1999). When $\epsilon = 0$, Fay's method becomes the traditional BRR method. For more information, see Dippo, Fay, and Morganstein (1984), Fay (1989), and Judkins (1990).

PROC SURVEYFREQ constructs Fay BRR replicates by using the first H columns of the $R \times R$ Hadamard matrix, where H denotes the number of strata. The rth replicate ($r = 1, 2, \ldots, R$) is drawn from the full sample according to the rth row of the Hadamard matrix as follows:

- If element (r, h) of the Hadamard matrix is 1, the sampling weight of the first PSU in stratum h is multiplied by ϵ, and the sampling weight of the second PSU is multiplied by $(2 - \epsilon)$ to form the rth replicate weights.
- If element (r, h) of the Hadamard matrix is –1, then the sampling weight of the second PSU in stratum h is multiplied by ϵ, and the sampling weight of the first PSU is multiplied by $(2 - \epsilon)$ to form the rth replicate weights.

You can use the OUTWEIGHTS= *method-option* to store the replicate weights in a SAS data set. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weight Output Data Set" on page 9868. You can provide these replicate weights to the procedure for subsequent analyses by using a REPWEIGHTS statement.

Variance Estimation

Let θ denote the population parameter to be estimated—for example, a proportion, total, odds ratio, or other statistic. Let $\hat{\theta}$ denote the estimate of θ from the full sample, and let $\hat{\theta}_r$ denote the estimate from the rth BRR replicate, which is computed by using the replicate weights. The Fay BRR variance estimate for $\hat{\theta}$ is computed as

$$\widehat{V}(\hat{\theta}) = \frac{1}{R(1-\epsilon)^2} \sum_{r=1}^{R} \left(\hat{\theta}_r - \hat{\theta}\right)^2$$

where R is the total number of replicates and ϵ is the Fay coefficient.

If you request Fay's BRR method and also include a REPWEIGHTS statement, PROC SURVEYFREQ uses the replicate weights that you provide and includes the Fay coefficient ϵ in the denominator of the variance estimate in the preceding expression.

If you specify the CENTER=REPLICATES *method-option*, the Fay BRR variance estimate is computed as

$$\widehat{V}(\hat{\theta}) = \frac{1}{R(1-\epsilon)^2} \sum_{r=1}^{R} \left(\hat{\theta}_r - \bar{\theta}\right)^2$$

where $\bar{\theta}$ is the average of the replicate estimates and is computed as follows:

$$\bar{\theta} = \frac{1}{R} \sum_{r=1}^{R} \hat{\theta}_r$$

If a parameter cannot be estimated from one or more replicates, the variance estimate is computed by using those replicates from which the parameter can be estimated. For example, suppose the parameter is a column proportion—the proportion of column j for table cell (i, j). If a replicate r contains no observations in column j, then the column j proportion is not estimable from replicate r. In this case, the BRR variance estimate is computed as

$$\widehat{V}(\hat{\theta}) = \frac{1}{R'(1-\epsilon)^2} \sum_{r=1}^{R'} \left(\hat{\theta}_r - \hat{\theta}\right)^2$$

where the summation is over the replicates for which the parameter θ is estimable and where R' is the number of those replicates.

Jackknife Method

When you specify the VARMETHOD=JACKKNIFE option, PROC SURVEYFREQ uses the delete-1 jackknife method for variance estimation. The jackknife method can be used for stratified sample designs and for designs with no stratification. If your design is stratified, the jackknife method requires at least two PSUs in each stratum. You can provide replicate weights for jackknife variance estimation by using a REPWEIGHTS statement, or the procedure can construct replicate weights for the analysis. PROC SURVEYFREQ estimates the parameter of interest (a proportion, total, odds ratio, or other statistic) from each replicate, and then uses the variability among replicate estimates to estimate the overall variance of the parameter estimate. For more information about jackknife variance estimation, see Wolter (1985) and Lohr (2022).

Replicate Weight Construction

If you do not provide replicate weights by using a REPWEIGHTS statement, PROC SURVEYFREQ constructs the replicates. The number of replicates R is the number of PSUs, and the procedure deletes one PSU from the full sample to form each replicate. The sampling weights are modified by the jackknife coefficient for the replicate to create the replicate weights.

If your design is not stratified (no STRATA statement), the jackknife coefficient has the same value for each replicate r. The jackknife coefficient is

$$\alpha_r = (R-1)/R \quad \text{for } r = 1, 2, \ldots, R$$

where R is the total number of replicates (or total number of PSUs). For the PSUs included in a replicate, the replicate weights are computed by dividing the original sampling weights by the jackknife coefficient. For the deleted PSU, which is not included in the replicate, the replicate weights equal 0. The replicate weight for the jth member of the ith PSU can be expressed as follows when the design is not stratified:

$$W_{ij}^r = \begin{cases} W_{ij}/\alpha_r & \text{if PSU } i \text{ is included in replicate } r \\ 0 & \text{otherwise} \end{cases}$$

where W_{ij} is the original sampling weight of unit (ij), r is the replicate number, and α_r is the jackknife coefficient.

If your design is stratified, the jackknife method requires at least two PSUs in each stratum. Let stratum h'_r be the stratum from which a PSU is deleted to form the rth replicate. Stratum h'_r is called the *donor stratum*. The jackknife coefficients are defined as

$$\alpha_r = (n_{h'_r} - 1)/n_{h'_r} \quad \text{for } r = 1, 2, \ldots, R$$

where $n_{h'_r}$ is the total number of PSUs in the donor stratum for replicate r. For all strata other than the donor stratum, the replicate r weights equal the original sampling weights. For PSUs included from the donor stratum, the replicate weights are computed by dividing the original sampling weights by the jackknife coefficient. For the deleted PSU, which is not included in the replicate, the replicate weights equal 0. The replicate weight for the jth member of the ith PSU in stratum h can be expressed as

$$W^r_{hij} = \begin{cases} W_{hij} & \text{if } h \neq h'_r \\ W_{hij}/\alpha_r & \text{if } h = h'_r \text{ and PSU } (hi) \text{ is included in replicate } r \\ 0 & \text{if } h = h'_r \text{ and PSU } (hi) \text{ is not included in replicate } r \end{cases}$$

You can use the OUTWEIGHTS= *method-option* to store the replicate weights in a SAS data set. You can also use the OUTJKCOEFS= *method-option* to store the jackknife coefficients in a SAS data set. For information about the contents of these output data sets, see the sections "Replicate Weight Output Data Set" on page 9868 and "Jackknife Coefficient Output Data Set" on page 9868. You can provide replicate weights and jackknife coefficients to the procedure for subsequent analyses by using a REPWEIGHTS statement. If you provide replicate weights but do not provide jackknife coefficients, PROC SURVEYFREQ uses $\alpha_r = (R-1)/R$ as the jackknife coefficient for all replicates.

Variance Estimation

Let θ denote the population parameter to be estimated—for example, a proportion, total, odds ratio, or other statistic. Let $\hat{\theta}$ denote the estimate of θ from the full sample, and let $\hat{\theta}_r$ be the estimate from the rth jackknife replicate, which is computed by using the replicate weights. The jackknife variance estimate for $\hat{\theta}$ is computed as

$$\widehat{V}(\hat{\theta}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\theta}_r - \hat{\theta}\right)^2$$

where R is the total number of replicates and α_r is the jackknife coefficient for replicate r.

If you specify the CENTER=REPLICATES *method-option*, the jackknife variance estimate is computed as

$$\widehat{V}(\hat{\theta}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\theta}_r - \bar{\theta}\right)^2$$

where $\bar{\theta}$ is the average of the replicate estimates and is computed as follows:

$$\bar{\theta} = \frac{1}{R} \sum_{r=1}^{R} \hat{\theta}_r$$

If a parameter cannot be estimated from one or more replicates, the variance estimate is computed by using those replicates from which the parameter can be estimated. For example, suppose the parameter is a column proportion—the proportion of column j for table cell (i, j). If a replicate r contains no observations in column j, then the column j proportion is not estimable from replicate r. In this case, the jackknife variance estimate is computed as

$$\widehat{V}(\hat{\theta}) = \frac{R}{R'} \sum_{r=1}^{R'} \alpha_r \left(\hat{\theta}_r - \hat{\theta}\right)^2$$

where the summation is over the replicates for which the parameter θ is estimable and where R' is the number of those replicates.

Statistical Computations

Definitions and Notation

For a stratified clustered sample design, define the following:

h	$=$	$1, 2, \ldots, H$	is the stratum number, with a total of H strata
i	$=$	$1, 2, \ldots, n_h$	is the cluster number within stratum h, with a total of n_h sample clusters in stratum h
j	$=$	$1, 2, \ldots, m_{hi}$	is the unit number within cluster i of stratum h with a total of m_{hi} sample units from cluster i of stratum h
n	$=$	$\sum_{h=1}^{H} \sum_{i=1}^{n_h} m_{hi}$	is the total number of observations in the sample

and

f_h	$=$	first-stage sampling rate for stratum h
W_{hij}	$=$	sampling weight of unit j in cluster i of stratum h

The sampling rate f_h, which is used in Taylor series and bootstrap variance estimation, is the fraction of first-stage units (PSUs) selected for the sample. You can specify the stratum sampling rates in the RATE= option. Or you can specify the stratum population totals in the TOTAL= option, and PROC SURVEYFREQ computes the f_h as the ratio of stratum sample sizes (PSUs) to stratum totals. For more information, see the section "Population Totals and Sampling Rates" on page 9823. If you do not specify the RATE= option or TOTAL= option, the procedure assumes that the stratum sampling rates f_h are negligible and does not use a finite population correction in variance computation.

This notation is also applicable to other sample designs. For example, for a design without stratification, you can let $H = 1$; for a sample design without clustering, you can let $m_{hi} = 1$ for every h and i, which replaces clusters with individual sampling units.

For a two-way table representing the crosstabulation of two variables, define the following, where there are R levels of the row variable and C levels of the column variable:

r	$=$	$1, 2, \ldots, R$	is the row number, with a total of R rows
c	$=$	$1, 2, \ldots, C$	is the column number, with a total of C columns
N_{rc}			is the population total in row r and column c
$N_{r\cdot}$	$=$	$\sum_{c=1}^{C} N_{rc}$	is the total in row r
$N_{\cdot c}$	$=$	$\sum_{r=1}^{R} N_{rc}$	is the total in column c
N	$=$	$\sum_{r=1}^{R} \sum_{c=1}^{C} N_{rc}$	is the overall total

$$
\begin{aligned}
P_{rc} &= N_{rc} \;/\; N && \text{is the population proportion in row } r \text{ and column } c \\
P_{r.} &= N_{r\cdot} \;/\; N && \text{is the proportion in row } r \\
P_{.c} &= N_{\cdot c} \;/\; N && \text{is the proportion in column } c \\
P_{rc}^{\;r} &= N_{rc} \;/\; N_{r\cdot} && \text{is the row proportion for table cell } (r,c) \\
P_{rc}^{\;c} &= N_{rc} \;/\; N_{\cdot c} && \text{is the column proportion for table cell } (r,c)
\end{aligned}
$$

For a specified observation (identified by stratum, cluster, and unit number within the cluster), define the following to indicate whether or not that observation belongs to cell (r, c), row r and column c, of the two-way table, for $r = 1, 2, \ldots, R$ and $c = 1, 2, \ldots, C$:

$$
\delta_{hij}(r,c) = \begin{cases} 1 & \text{if observation } (hij) \text{ is in cell } (r,c) \\ 0 & \text{otherwise} \end{cases}
$$

Similarly, define the following functions to indicate the observation's row and column classification:

$$
\delta_{hij}(r\,\cdot) = \begin{cases} 1 & \text{if observation } (hij) \text{ is in row } r \\ 0 & \text{otherwise} \end{cases}
$$

$$
\delta_{hij}(\cdot\,c) = \begin{cases} 1 & \text{if observation } (hij) \text{ is in column } c \\ 0 & \text{otherwise} \end{cases}
$$

Totals

PROC SURVEYFREQ estimates population frequency totals for the specified crosstabulation tables, including totals for two-way table cells, rows, columns, and overall totals. The procedure computes the estimate of the total frequency in table cell (r,c) as the weighted frequency sum,

$$
\widehat{N}_{rc} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \delta_{hij}(r,c)\; W_{hij}
$$

Similarly, PROC SURVEYFREQ computes estimates of row totals, column totals, and overall totals as

$$
\widehat{N}_{r\cdot} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \delta_{hij}(r\,\cdot)\; W_{hij}
$$

$$
\widehat{N}_{\cdot c} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \delta_{hij}(\cdot\,c)\; W_{hij}
$$

$$
\widehat{N} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} W_{hij}
$$

PROC SURVEYFREQ estimates the variances of totals by using the variance estimation method that you request. If you request a replication method (bootstrap, BRR, jackknife, or replicate weights), the procedure estimates the variances as described in the section "Replication Variance Estimation" on page 9828.

By default, PROC SURVEYFREQ estimates variances by using the Taylor series method (which you can also request by specifying the VARMETHOD=TAYLOR option). Because totals are linear statistics, their variances can be estimated directly, without the approximation that is used for proportions and other nonlinear statistics. PROC SURVEYFREQ estimates the variance of the total frequency in table cell (r, c) as

$$\widehat{\mathrm{Var}}(\widehat{N}_{rc}) = \sum_{h=1}^{H} \widehat{\mathrm{Var}}_h(\widehat{N}_{rc})$$

where if $n_h > 1$,

$$\begin{aligned}
\widehat{\mathrm{Var}}_h(\widehat{N}_{rc}) &= \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (n_{rc}^{hi} - \bar{n}_{rc}^{h})^2 \\
n_{rc}^{hi} &= \sum_{j=1}^{m_{hi}} \delta_{hij}(r,c)\, W_{hij} \\
\bar{n}_{rc}^{h} &= \left(\sum_{i=1}^{n_h} n_{rc}^{hi} \right) / \, n_h
\end{aligned}$$

and if $n_h = 1$,

$$\widehat{\mathrm{Var}}_h(\widehat{N}_{rc}) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

The standard error of the total is computed as

$$\mathrm{StdErr}(\widehat{N}_{rc}) = \sqrt{\widehat{\mathrm{Var}}(\widehat{N}_{rc})}$$

The variances and standard errors are computed in a similar manner for row totals, column totals, and overall table totals.

Covariances of Frequency Estimates

The covariance matrix of the table cell frequency estimates is an $rc \times rc$ matrix that contains the pairwise cell frequency covariances. $\widehat{\mathbf{V}}(\widehat{\mathbf{N}})$ denotes the covariance matrix and $\widehat{\mathrm{Cov}}(\widehat{N}_{rc}, \widehat{N}_{ab})$ denotes the pairwise covariances, for $r = 1, \ldots, R$; $c = 1, \ldots, C$; $a = 1, \ldots, R$; and $b = 1, \ldots, C$.

PROC SURVEYFREQ estimates the covariances by using the variance estimation method that you request. If you request a replication method (bootstrap, BRR, jackknife, or replicate weights), the procedure estimates the covariances as described in the section "Replication Variance Estimation" on page 9828.

By default, or if you request Taylor series variance estimation, PROC SURVEYFREQ estimates the covariance between frequency estimates for table cells (r,c) and (a,b) as

$$\widehat{\mathrm{Cov}}(\widehat{N}_{rc}, \widehat{N}_{ab}) = \sum_{h=1}^{H} \left(\frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (n_{rc}^{hi} - \bar{n}_{rc}^{h})\,(n_{ab}^{hi} - \bar{n}_{ab}^{h}) \right)$$

Proportions

PROC SURVEYFREQ computes the estimate of the proportion in table cell (r, c) as the ratio of the estimated total for the table cell to the estimated overall total,

$$\begin{aligned}\widehat{P}_{rc} &= \widehat{N}_{rc} \,/\, \widehat{N} \\ &= \left(\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} \delta_{hij}(r,c)\, W_{hij}\right) \Big/ \left(\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} W_{hij}\right)\end{aligned}$$

PROC SURVEYFREQ estimates the variances of the proportion estimates by using the variance estimation method that you request. If you request a replication method (bootstrap, BRR, jackknife, or replicate weights), the procedure estimates the variances as described in the section "Replication Variance Estimation" on page 9828. By default, PROC SURVEYFREQ estimates variances by using the Taylor series method (which you can also request by specifying the VARMETHOD=TAYLOR option).

By using Taylor series linearization, the variance of a proportion estimate can be expressed as

$$\widehat{\mathrm{Var}}(\widehat{P}_{rc}) = \sum_{h=1}^{H} \widehat{\mathrm{Var}}_h(\widehat{P}_{rc})$$

where if $n_h > 1$,

$$\begin{aligned}\widehat{\mathrm{Var}}_h(\widehat{P}_{rc}) &= \frac{n_h(1-f_h)}{n_h-1}\sum_{i=1}^{n_h}(e_{rc}^{hi} - \bar{e}_{rc}^{\,h})^2 \\ e_{rc}^{hi} &= \left(\sum_{j=1}^{m_{hi}} (\delta_{hij}(r,c) - \widehat{P}_{rc})\, W_{hij}\right) \Big/ \widehat{N} \\ \bar{e}_{rc}^{\,h} &= \left(\sum_{i=1}^{n_h} e_{rc}^{hi}\right) \Big/ n_h\end{aligned}$$

and if $n_h = 1$,

$$\widehat{\mathrm{Var}}_h(\widehat{P}_{rc}) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

The standard error of the proportion is computed as

$$\mathrm{StdErr}(\widehat{P}_{rc}) = \sqrt{\widehat{\mathrm{Var}}(\widehat{P}_{rc})}$$

Similarly, the estimate of the proportion in row r is

$$\widehat{P}_{r\cdot} = \widehat{N}_{r\cdot} \,/\, \widehat{N}$$

And its variance estimate is

$$\widehat{\mathrm{Var}}(\widehat{P}_{r\cdot}) = \sum_{h=1}^{H} \widehat{\mathrm{Var}}_h(\widehat{P}_{r\cdot})$$

where if $n_h > 1$,

$$\begin{aligned}
\widehat{\mathrm{Var}}_h(\widehat{P}_{r\cdot}) &= \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (e_{r\cdot}^{hi} - \bar{e}_{r\cdot}^{h})^2 \\
e_{r\cdot}^{hi} &= \left(\sum_{j=1}^{m_{hi}} (\delta_{hij}(r\,\cdot) - \widehat{P}_{r\cdot})\, W_{hij} \right) \Big/ \widehat{N} \\
\bar{e}_{r\cdot}^{h} &= \left(\sum_{i=1}^{n_h} e_{r\cdot}^{hi} \right) \Big/ n_h
\end{aligned}$$

and if $n_h = 1$,

$$\widehat{\mathrm{Var}}_h(\widehat{P}_{r\cdot}) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

The standard error of the proportion in row r is computed as

$$\mathrm{StdErr}(\widehat{P}_{r\cdot}) = \sqrt{\widehat{\mathrm{Var}}(\widehat{P}_{r\cdot})}$$

Computations for the proportion in column c are done in the same way.

Row and Column Proportions

PROC SURVEYFREQ computes the estimate of the row proportion for table cell (r, c) as the ratio of the estimated total for the table cell to the estimated total for row r,

$$\begin{aligned}
\widehat{P}_{rc}^{\,r} &= \widehat{N}_{rc} \,/\, \widehat{N}_{r\cdot} \\
&= \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \delta_{hij}(r,c)\, W_{hij} \right) \Big/ \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \delta_{hij}(r\,\cdot)\, W_{hij} \right)
\end{aligned}$$

Similarly, PROC SURVEYFREQ estimates the column proportion for table cell (r, c) as the ratio of the estimated total for the table cell to the estimated total for column c,

$$\begin{aligned}
\widehat{P}_{rc}^{\,c} &= \widehat{N}_{rc} \,/\, \widehat{N}_{\cdot c} \\
&= \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \delta_{hij}(r,c)\, W_{hij} \right) \Big/ \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \delta_{hij}(\cdot\, c)\, W_{hij} \right)
\end{aligned}$$

PROC SURVEYFREQ estimates the variances of the row and column proportion estimates by using the variance estimation method that you request. If you request a replication method (bootstrap, BRR, jackknife, or replicate weights), the procedure estimates the variances as described in the section "Replication Variance Estimation" on page 9828. By default, PROC SURVEYFREQ estimates variances by using the Taylor series method (which you can also request by specifying the VARMETHOD=TAYLOR option).

By using Taylor series linearization, the variance of the row proportion estimate can be expressed as

$$\widehat{\mathrm{Var}}(\widehat{P}^{\,r}_{rc}) = \sum_{h=1}^{H} \widehat{\mathrm{Var}}_h(\widehat{P}_{rc})$$

where if $n_h > 1$,

$$\begin{aligned}
\widehat{\mathrm{Var}}_h(\widehat{P}^{\,r}_{rc}) &= \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (g^{hi}_{rc} - \bar{g}^{\,h}_{rc})^2 \\
g^{hi}_{rc} &= \left(\sum_{j=1}^{m_{hi}} (\delta_{hij}(r,c) - \widehat{P}^{\,r}_{rc}\, \delta_{hij}(r\,\cdot))\, W_{hij} \right) \Big/ \widehat{N}_{r\cdot} \\
\bar{g}^{\,h}_{rc} &= \left(\sum_{i=1}^{n_h} g^{hi}_{rc} \right) \Big/ n_h
\end{aligned}$$

and if $n_h = 1$,

$$\widehat{\mathrm{Var}}_h(\widehat{P}^{\,r}_{rc}) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

The standard error of the row proportion is computed as

$$\mathrm{StdErr}(\widehat{P}^{\,r}_{rc}) = \sqrt{\widehat{\mathrm{Var}}(\widehat{P}^{\,r}_{rc})}$$

The Taylor series variance estimate for the column proportion is computed as described previously for the row proportion, but with

$$g^{hi}_{rc} = \left(\sum_{j=1}^{m_{hi}} (\delta_{hij}(r,c) - \widehat{P}^{\,c}_{rc}\, \delta_{hij}(\cdot\, c))\, W_{hij} \right) \Big/ \widehat{N}_{\cdot c}$$

Confidence Limits for Totals

If you specify the CLWT option in the TABLES statement, PROC SURVEYFREQ computes confidence limits for the weighted frequencies (totals) in the crosstabulation tables.

For the total in table cell (r, c), the confidence limits are computed as

$$\widehat{N}_{rc} \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\widehat{N}_{rc}) \right)$$

where $\widehat{N}_{rc}$ is the estimate of the total frequency in table cell (r, c), $\mathrm{StdErr}(\widehat{N}_{rc})$ is the standard error of the estimate, and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The confidence level α is determined by the value of the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

The confidence limits for row totals, column totals, and the overall total are computed similarly to the confidence limits for table cell totals.

For each table request, PROC SURVEYFREQ produces a nondisplayed ODS table, "Table Summary," which contains the number of observations, strata, and clusters that are included in the analysis of the requested table. When you request confidence limits, the "Table Summary" data set also contains the degrees of freedom df and the value of $t_{df,\alpha/2}$ that is used to compute the confidence limits. For more information about this output data set, see Example 118.3.

Confidence Limits for Proportions

If you specify the CL option in the TABLES statement, PROC SURVEYFREQ computes confidence limits for the proportions in the frequency and crosstabulation tables. The procedure provides Wald-type confidence limits, logit confidence limits, and the following modified confidence limits: Agresti-Coull, Clopper-Pearson (exact), Jeffreys, uniform, and Wilson (score). By default, PROC SURVEYFREQ computes Wald-type ("linear") confidence limits by using the variance estimates that are based on the sample design.

For information about design-based confidence limits for proportions (including comparisons of their performance), see Korn and Graubard (1999), Korn and Graubard (1998), Franco et al. (2019), Curtin et al. (2006), and Sukasih and Jang (2005). For more information about binomial confidence limits, see Brown, Cai, and DasGupta (2001) and Agresti and Coull (1998), in addition to the references cited in the following sections.

PROC SURVEYFREQ provides the option to compute an alternative confidence limit type for extreme (small and large) proportions or for small-frequency table cells. For more information, see Curtin et al. (2006). If you specify the PSMALL=*p cl-option*, PROC SURVEYFREQ computes the alternative confidence limit type when the proportion estimate is less than or equal to p or greater than or equal to $(1 - p)$. When the proportion estimate is between p and $(1 - p)$, PROC SURVEYFREQ computes Wald confidence limits. If you specify the NSMALL=*n cl-option*, PROC SURVEYFREQ computes the alternative confidence limit type when the table cell frequency is less than or equal to n.

For each table request, PROC SURVEYFREQ produces a nondisplayed ODS table, "Table Summary," which contains the number of observations, strata, and clusters that are included in the analysis of the requested table. When you request confidence limits, the "Table Summary" data set also contains the degrees of freedom df and the corresponding value of $t_{df,\alpha/2}$ that is used to compute the confidence limits. For more information about this output data set, see Example 118.3.

Wald Confidence Limits

By default, PROC SURVEYFREQ computes Wald-type ("linear") confidence limits for proportions. These confidence limits use variance estimates that are based on the sample design. For the proportion in table cell (r, c), the Wald confidence limits are computed as

$$\widehat{P}_{rc} \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\widehat{P}_{rc})\right)$$

where $\widehat{P}_{rc}$ is the estimate of the proportion in table cell (r, c), $\mathrm{StdErr}(\widehat{P}_{rc})$ is the standard error of the estimate, and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For

more information, see the section "Degrees of Freedom" on page 9846.) The confidence level α is determined by the value of the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

The confidence limits for row proportions and column proportions are computed similarly to the confidence limits for table cell proportions.

Logit Confidence Limits

If you specify the CL(TYPE=LOGIT) option, PROC SURVEYFREQ computes logit confidence limits for proportions. For more information, see Agresti (2013) and Korn and Graubard (1998).

Logit confidence limits for proportions are based on the logit transformation $Y = \log(\hat{p}/(1-\hat{p}))$. The logit confidence limits P_L and P_U are computed as

$$P_L = \exp(Y_L) \,/\, (\,1 + \exp(Y_L)\,)$$

$$P_U = \exp(Y_U) \,/\, (\,1 + \exp(Y_U)\,)$$

where

$$(\,Y_L,\ Y_U\,) = \log(\hat{p}/(1-\hat{p})) \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\hat{p}) \,/\, (\hat{p}(1-\hat{p}))\right)$$

where $\hat{p}$ is the estimate of the proportion, $\mathrm{StdErr}(\hat{p})$ is the standard error of the estimate, and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The confidence level α is determined by the value of the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

Modified Confidence Limits

PROC SURVEYFREQ uses the Korn and Graubard (1998) method to compute the following design-based modified confidence limits: Agresti-Coull, Clopper-Pearson (exact), Jeffreys, uniform, and Wilson (score). This method substitutes the degrees-of-freedom-adjusted effective sample size for the original sample size in the confidence limit computations. For more information, see Franco et al. (2019) and Dean and Pagano (2015).

Effective Sample Size The effective sample size n_e is computed as

$$n_e = n \,/\, \mathrm{Deff}$$

where n is the original sample size (unweighted frequency) that corresponds to the total domain of the proportion estimate, and Deff is the design effect.

If the proportion is computed for a table cell of a two-way table, then the domain is the two-way table, and the sample size n is the frequency of the two-way table. If the proportion is a row proportion, which is based on a two-way table row, then the domain is the row, and the sample size n is the frequency of the row.

The design effect for an estimate is the ratio of the actual variance (estimated based on the sample design) to the variance of a simple random sample with the same number of observations. For more information, see the section "Design Effect" on page 9847.

By default, PROC SURVEYFREQ uses (n – 1) as the divisor in the SRS component of the design effect. To use n as the divisor for the design effects in the confidence limit computations, you can specify the CL(VARDEF=N) option in the TABLES statement. To use n as the divisor for all design effects that PROC SURVEYFREQ computes, you can specify the DEFF(VARDEF=N) option in the PROC SURVEYFREQ statement.

Degrees-of-Freedom Adjustment The adjusted effective sample size n_e^* is computed by applying a degrees-of-freedom adjustment to the effective sample size n_e. By default, PROC SURVEYFREQ uses the Korn and Graubard (1998) adjustment factor and computes the adjusted sample size as

$$n_e^* = n_e \left(\frac{t_{(n-1),\alpha/2}}{t_{df,\alpha/2}} \right)^2$$

where df is the degrees of freedom and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. The degrees of freedom depend on the sample design and the variance estimation method. For more information, see the section "Degrees of Freedom" on page 9846. The confidence level α is determined by the value of the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

If you specify the CL(ADJUST=DP) option, PROC SURVEYFREQ uses the Dean and Pagano (2015) adjustment factor, which replaces the t quantile in the numerator with a normal quantile and computes the adjusted effective sample size as

$$n_e^* = n_e \left(\frac{z_{\alpha/2}}{t_{df,\alpha/2}} \right)^2$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution.

If you specify the CL(ADJUST=NO) option, PROC SURVEYFREQ omits the adjustment and uses the (unadjusted) effective sample size n_e instead of n_e^* in the confidence limit computations.

Truncation The design effect is usually greater than 1 for complex survey designs, and in this case the effective sample size is less than the actual sample size. If the adjusted effective sample size n_e^* is greater than the actual sample size n, then by default, PROC SURVEYFREQ truncates the value of n_e^* to n, as recommended by Korn and Graubard (1998). If you specify the CL(TRUNCATE=NO) option, PROC SURVEYFREQ does not truncate the value of n_e^* if it exceeds n.

Modified Agresti-Coull Confidence Limits The modified Agresti-Coull confidence limits are constructed by applying the Korn and Graubard (1998) method to the Agresti-Coull form (Agresti and Coull 1998; Brown, Cai, and DasGupta 2001). The adjusted effective sample size n_e^* is substituted for the sample size, and the adjusted effective sample size times the proportion estimate ($n_e^* \times \hat{p}$) is substituted for the number of positive responses. For more information, see Franco et al. (2019).

PROC SURVEYFREQ computes modified Agresti-Coull confidence limits for the proportion as

$$\tilde{p} \pm \left(z_{\alpha/2} \times \sqrt{\tilde{p}\,(1-\tilde{p}) \,/\, \tilde{n}_e^*} \right)$$

where

$$\begin{aligned} \tilde{n}_1 &= (n_e^* \times \hat{p}) + z_{\alpha/2}^2/2 \\ \tilde{n} &= n_e^* + z_{\alpha/2}^2 \\ \tilde{p} &= \tilde{n}_1 \,/\, \tilde{n} \end{aligned}$$

and $\hat{p}$ is the design-based proportion estimate.

Modified Clopper-Pearson Confidence Limits Clopper-Pearson (exact) confidence limits for the binomial proportion are constructed by inverting the exact equal-tailed test that is based on the binomial distribution. This method is attributed to Clopper and Pearson (1934). For a derivation of the F distribution expression for the confidence limits, see Leemis and Trivedi (1996).

Modified Clopper-Pearson confidence limits are constructed by using the Korn and Graubard (1998) method. The adjusted effective sample size n_e^* is substituted for the sample size in the standard Clopper-Pearson expression, and the proportion estimate times the adjusted effective sample size ($\hat{p} \times n_e^*$) is substituted for the number of positive responses.

The modified Clopper-Pearson confidence limits for a proportion (P_L and P_U) are computed as

$$P_L = \left(1 + \frac{n_e^* - \hat{p}\, n_e^* + 1}{\hat{p}\, n_e^*\ F(\,\alpha/2,\ 2\hat{p}\, n_e^*,\ 2(n_e^* - \hat{p}\, n_e^* + 1)\,)}\right)^{-1}$$

$$P_U = \left(1 + \frac{n_e^* - \hat{p}\, n_e^*}{(\hat{p}\, n_e^* + 1)\ F(\,1-\alpha/2,\ 2(\hat{p}\, n_e^* + 1),\ 2(n_e^* - \hat{p}\, n_e^*)\,)}\right)^{-1}$$

where $F(\alpha/2, b, c)$ is the $(\alpha/2)$th percentile of the F distribution with b and c degrees of freedom, n_e^* is the adjusted effective sample size, and $\hat{p}$ is the design-based proportion estimate.

Modified Jeffreys Confidence Limits The Jeffreys confidence interval is an equal-tailed interval that is based on the noninformative Jeffreys prior for a binomial proportion. For more information, see Brown, Cai, and DasGupta (2001) and Berger (1985).

PROC SURVEYFREQ computes modified Jeffreys confidence limits for a proportion as

$$\big(\ \beta(\alpha/2,\ n_1 + 1/2,\ n_e^* - n_1 + 1/2),\ \ \beta(1-\alpha/2,\ n_1 + 1/2,\ n_e^* - n_1 + 1/2)\ \big)$$

where $n_1 = n_e^* \times \hat{p}$ and $\beta(\alpha, b, c)$ is the αth percentile of the beta distribution with shape parameters b and c. The lower confidence limit is set to 0 when $n_1 = 0$, and the upper confidence limit is set to 1 when $n_1 = n$. For more information, see Franco et al. (2019) and Carlin and Louis (2009).

Modified Uniform Confidence Limits PROC SURVEYFREQ computes modified uniform confidence limits for a proportion as

$$\big(\ \beta(\alpha/2,\ n_1 + 1,\ n_e^* - n_1 + 1),\ \ \beta(1-\alpha/2,\ n_1 + 1,\ n_e^* - n_1 + 1)\ \big)$$

where $n_1 = n_e^* \times \hat{p}$ and $\beta(\alpha, b, c)$ is the αth percentile of the beta distribution with shape parameters b and c. The lower confidence limit is set to 0 when $n_1 = 0$, and the upper confidence limit is set to 1 when $n_1 = n$. For more information, see Franco et al. (2019) and Carlin and Louis (2009).

Modified Wilson Confidence Limits Wilson confidence limits for the binomial proportion are also known as score confidence limits and are attributed to Wilson (1927). The confidence limits are based on inverting the normal test that uses the null proportion in the variance (the score test). For more information, see Agresti and Coull (1998), Newcombe (1998), and Brown, Cai, and DasGupta (2001).

PROC SURVEYFREQ computes modified Wilson confidence limits by substituting the adjusted effective sample size n_e^* for the original sample size in the standard Wilson computation. For more information, see Korn and Graubard (1999).

The modified Wilson confidence limits for a proportion are computed as

$$\left(\hat{p} \ + \ \kappa^2/2n_e^* \ \pm \ \kappa\sqrt{\left(\hat{p}(1-\hat{p}) + \kappa^2/4n_e^*\right)/n_e^*}\ \right) \ / \ \left(1 + \kappa^2/n_e^*\right)$$

where n_e^* is the adjusted effective sample size and $\hat{p}$ is the design-based estimate of the proportion. By default, $\kappa = z_{\alpha/2}$. If you specify the CL(ADJUST=NO) option to use the unadjusted effective sample size n_e instead of n_e^*, then $\kappa = t_{df,\alpha/2}$. For more information, see Curtin et al. (2006).

Degrees of Freedom

PROC SURVEYFREQ uses the degrees of freedom of the variance estimator to obtain the *t* distribution's percentile for confidence limits for proportions, totals, and other statistics. The procedure also uses the degrees of freedom to compute the *F* statistics in the Rao-Scott and Wald chi-square tests.

PROC SURVEYFREQ computes the degrees of freedom based on the variance estimation method and the sample design. Alternatively, you can use the DF= option in the TABLES statement to specify the degrees of freedom.

For Taylor series variance estimation, PROC SURVEYFREQ calculates the degrees of freedom (*df*) as the number of clusters minus the number of strata. If there are no clusters, then *df* is the number of observations minus the number of strata. If the design is not stratified, then *df* is the number of clusters minus one. These numbers are based on the observations included in the analysis of the individual table request. These numbers do not count observations that are excluded from the table because of missing values. For more information, see the section "Missing Values" on page 9825. If you specify the MISSING option, missing values are treated as valid nonmissing levels and are included when computing degrees of freedom. If you specify the NOMCAR option for Taylor series variance estimation, observations with missing values of the TABLES variables are included when computing degrees of freedom.

If you use a REPWEIGHTS statement to provide replicate weights, the degrees of freedom equals the number of replicates, which is the number of REPWEIGHTS variables that you provide. Alternatively, you can use the DF= option in the REPWEIGHTS or the TABLES statement to specify the degrees of freedom.

For BRR variance estimation (when you do not use a REPWEIGHTS statement), PROC SURVEYFREQ calculates the degrees of freedom as the number of strata. By default, PROC SURVEYFREQ computes the number of strata by using all valid observations in the data set. If you specify the DFADJ *method-option*, PROC SURVEYFREQ computes the number of strata as the number of nonmissing strata for the current table request. This excludes any empty strata that occur when observations are excluded because of missing values of a TABLES variable. For more information, see the section "Missing Values" on page 9825.

For bootstrap and jackknife variance estimation (when you do not use a REPWEIGHTS statement), PROC SURVEYFREQ calculates the degrees of freedom (*df*) as the number of clusters minus the number of strata. If there are no clusters, then *df* is the number of observations minus the number of strata. If the design is not stratified, then *df* is the number of clusters minus one. By default, PROC SURVEYFREQ computes the stratum and cluster counts by using all valid observations in the data set. If you specify the DFADJ *method-option*, PROC SURVEYFREQ computes the stratum and cluster counts by using only nonmissing observations for the current table request. This excludes any empty strata or clusters that occur when observations are excluded because of missing values of a TABLES variable. For more information, see the section "Missing Values" on page 9825.

Coefficient of Variation

If you specify the CV option in the TABLES statement, PROC SURVEYFREQ computes the coefficients of variation for the proportion estimates in the frequency and crosstabulation tables. The coefficient of variation is the ratio of the standard error to the estimate.

For the proportion in table cell (r, c), the coefficient of variation is computed as

$$\mathrm{CV}(\widehat{P}_{rc}) = \mathrm{StdErr}(\widehat{P}_{rc}) \,/\, \widehat{P}_{rc}$$

where $\widehat{P}_{rc}$ is the estimate of the proportion in table cell (r, c), and $\mathrm{StdErr}(\widehat{P}_{rc})$ is the standard error of the estimate. The coefficients of variation for row proportions and column proportions are computed similarly.

If you specify the CVWT option in the TABLES statement, PROC SURVEYFREQ computes the coefficients of variation for the weighted frequencies (estimated totals) in the crosstabulation tables. For the total in table cell (r, c), the coefficient of variation is computed as

$$\mathrm{CV}(\widehat{N}_{rc}) = \mathrm{StdErr}(\widehat{N}_{rc}) \,/\, \widehat{N}_{rc}$$

where $\widehat{N}_{rc}$ is the estimate of the total in table cell (r, c), and $\mathrm{StdErr}(\widehat{N}_{rc})$ is the standard error of the estimate. The coefficients of variation for row totals, column totals, and the overall total are computed similarly.

Design Effect

PROC SURVEYFREQ provides design effects for the overall proportion estimates when you specify the DEFF option in the TABLES statement. The procedure also uses design effects to compute Rao-Scott chi-square tests (CHISQ) and modified confidence limits for proportions (CL). For more information, see the sections "Rao-Scott Chi-Square Test" on page 9858 and "Modified Confidence Limits" on page 9843.

The design effect for an estimate is the ratio of the actual variance (estimated based on the sample design) to the variance of a simple random sample (SRS) with the same number of observations. For more information, see Lohr (2022) and Kish (1965).

PROC SURVEYFREQ computes the design effect for the proportion in table cell (r, c) as

$$\begin{aligned}
\mathrm{Deff}(\widehat{P}_{rc}) &= \widehat{\mathrm{Var}}(\widehat{P}_{rc}) \,/\, \widehat{\mathrm{Var}}_{\mathrm{srs}}(\widehat{P}_{rc}) \\
&= \widehat{\mathrm{Var}}(\widehat{P}_{rc}) \,/\, \Big((1-f)\,\widehat{P}_{rc}\,(1-\widehat{P}_{rc}) \,/\, (n-1)\Big)
\end{aligned}$$

where $\widehat{P}_{rc}$ is the estimate of the proportion in table cell (r, c); $\widehat{\mathrm{Var}}(\widehat{P}_{rc})$ is the variance of the estimate; n is the sample size (unweighted frequency) for the two-way table; and f is the overall sampling fraction, which is described in the section "Sampling Fraction" on page 9848.

If you specify the DEFF(VARDEF=N) option in the PROC SURVEYFREQ statement, the procedure uses n instead of (n – 1) as the divisor in the SRS variance $\widehat{\mathrm{Var}}_{\mathrm{srs}}(\widehat{P}_{rc})$.

For Taylor series variance estimation, by default, PROC SURVEYFREQ includes the finite population correction (1 –f) in the SRS variance when you specify sampling rates or population totals in the RATE= or TOTAL= option, respectively. To exclude the finite population correction, you can specify the DEFF(FPC=NO) option in the PROC SURVEYFREQ statement.

For replication variance estimation, by default, PROC SURVEYFREQ does not include the finite population correction $(1 - f)$ in the SRS variance. To include the finite population correction, you can specify the DEFF(FPC=YES) option in the PROC SURVEYFREQ statement and provide sampling rates or population totals in the RATE= or TOTAL= option, respectively.

PROC SURVEYFREQ computes design effects in the same way for proportions in one-way frequency tables and for row and column proportions in two-way tables. In these design effect computations, the value of n is the sample size (unweighted frequency) that corresponds to the total domain of the proportion estimate. For table cell proportions of a two-way table, the domain is the two-way table and the sample size n is the frequency of the two-way table. For row proportions, which are based on a two-way table row, the domain is the row and the sample size n is the row frequency. For column proportions, the sample size n is the column frequency.

Sampling Fraction

PROC SURVEYFREQ determines the sampling fraction f for the design effect by using the sampling rates or population totals that you provide in the RATE= or TOTAL= option, respectively. If you omit both of these options, PROC SURVEYFREQ assumes that the sampling fraction f is negligible and does not include a finite population correction in the analysis, as described in the section "Population Totals and Sampling Rates" on page 9823.

If you specify RATE=*value*, PROC SURVEYFREQ uses this value as the overall sampling fraction f. If you specify TOTAL=*value*, PROC SURVEYFREQ computes f as the ratio of the number of PSUs in the sample to the total value.

If you provide stratum sampling rates by using the RATE=*SAS-data-set* option, PROC SURVEYFREQ computes each stratum total as the number of sample PSUs in the stratum divided by the stratum sampling rate. Or you can provide the stratum totals by using the TOTAL=*SAS-data-set* option. The overall total is then computed as the sum of the stratum totals, and the overall sampling fraction f is computed as the ratio of the number of sample PSUs to the overall total.

Expected Weighted Frequency

If you specify the EXPECTED option in the TABLES statement, PROC SURVEYFREQ computes expected weighted frequencies for the table cells in two-way tables. The expected weighted frequencies are computed under the null hypothesis that the row and column variables are independent. The expected weighted frequency for table cell (r, c) is

$$E_{rc} = \widehat{N}_{r\cdot}\, \widehat{N}_{\cdot c} \,/\, \widehat{N}$$

where $\widehat{N}_{r\cdot}$ is the estimated total for row r, $\widehat{N}_{\cdot c}$ is the estimated total for column c, and $\widehat{N}$ is the estimated overall total. Equivalently, the expected weighted frequency can be expressed as

$$E_{rc} = \widehat{P}_{r\cdot}\, \widehat{P}_{\cdot c}\, \widehat{N}$$

These expected values are used in the design-based chi-square tests of independence, as described in the sections "Rao-Scott Chi-Square Test" on page 9858 and "Wald Chi-Square Test" on page 9865.

Sensitivity and Specificity

The SENSPEC option in the TABLES statement provides estimates of sensitivity, specificity, positive predictive value, and negative predictive value for 2×2 tables. These measures are conditional (row and column) proportions in the 2×2 crosstabulation table. In sensitivity analysis, the row variable might represent a positive or negative diagnostic test result, and the column variable might represent the presence or absence of a condition. For more information, see Fleiss, Levin, and Paik (2003). The SENSPEC option also provides an estimate of accuracy, which is the proportion of correct classifications.

By default, the SENSPEC computations use table cell (1,1) as the reference cell that represents the true positives (positive row value and positive column value). You can specify a different reference cell by using the SENSPEC REFCELL= suboption or the REFCOLUMN= and REFROW= suboptions. For example, REFCELL=4 specifies table cell (2,2) as the true-positive cell (and row 2 and column 2 as the positive row and column levels, respectively). The notation in this section assumes that table cell (1,1) is the reference cell.

Sensitivity is computed as the column proportion for table cell (1,1), which is the ratio of the total in table cell (1,1) to the total in column 1 of the 2×2 table. Sensitivity is denoted by

$$\mathrm{SN} = \widehat{N}_{11} \,/\, \widehat{N}_{\cdot 1}$$

Specificity is computed as the column proportion for table cell (2,2), which is the ratio of the total in table cell (2,2) to the total in column 2. Specificity is denoted by

$$\mathrm{SP} = \widehat{N}_{22} \,/\, \widehat{N}_{\cdot 2}$$

The positive predictive value is the row proportion for table cell (1,1), which is computed as

$$\mathrm{PPV} = \widehat{N}_{11} \,/\, \widehat{N}_{1\cdot}$$

The negative predictive value is the row proportion for table cell (2,2), which is computed as

$$\mathrm{NPV} = \widehat{N}_{22} \,/\, \widehat{N}_{2\cdot}$$

For more information, see the section "Row and Column Proportions" on page 9840.

Accuracy (correct classification rate) is the overall proportion in table cells (1,1) and (2,2), which is computed as

$$\mathrm{A} = \left(\widehat{N}_{11} + \widehat{N}_{22}\right) \,/\, \widehat{N}$$

PROC SURVEYFREQ estimates the variance of these proportions by using the method that you specify (Taylor series, bootstrap, BRR, jackknife, or replicate weights). For more information, see the sections "Taylor Series Variance Estimation" on page 9827 and "Replication Variance Estimation" on page 9828.

The "Sensitivity and Specificity" table provides the estimates together with their standard errors and confidence limits. By default, PROC SURVEYFREQ computes Wald-type (linear) confidence limits by using the variance estimates that are based on the sample design. If you specify the SENSPEC(CLTYPE) option, PROC SURVEYFREQ uses the confidence limit type that you request in the CL(TYPE=) option. Available confidence limit types include Agresti-Coull, Clopper-Pearson (exact), Jeffreys, logit, uniform, and Wilson (score). For more information, see the section "Confidence Limits for Proportions" on page 9842.

The Wald confidence limits are computed as

$$\widehat{S} \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\widehat{S}) \right)$$

where $\mathrm{StdErr}(\widehat{S})$ is the standard error of the estimate S and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits. For more information, see the section "Wald Confidence Limits" on page 9842.

Risks and Risk Difference

The RISK option provides estimates of risks (binomial proportions) and risk differences for 2×2 tables, together with their standard errors and confidence limits. Risk statistics include the row 1 risk, row 2 risk, overall risk, and risk difference.

When you specify the RISK option, PROC SURVEYFREQ provides both column 1 and column 2 risks by default. You can request only column 1 (or only column 2) risks by specifying the RISK(COLUMN=) option. To display only the risk difference (and suppress display of the row 1, row 2, and overall risks), you can specify the RISKDIFF(ONLY) option.

The column 1 risk for row 1 is the row 1 proportion for table cell (1,1). The column 1 risk estimate is computed as the ratio of the estimated total for table cell (1,1) to the estimated total for row 1,

$$\widehat{P}_{11}^{(1)} = \widehat{N}_{11} \,/\, \widehat{N}_{1\cdot}$$

where the total estimates are computed as described in the section "Totals" on page 9837. The column 1 risk for row 2 is the row 2 proportion for table cell (2,1), which is estimated as

$$\widehat{P}_{21}^{(2)} = \widehat{N}_{21} \,/\, \widehat{N}_{2\cdot}$$

The overall column 1 risk is the overall proportion in column 1, and its estimate is computed as

$$\widehat{P}_{\cdot 1} = \widehat{N}_{\cdot 1} \,/\, \widehat{N}$$

The column 2 risk estimates are computed similarly.

The row 1 and row 2 risks are the same as the row proportions for a 2×2 table, and their variances are computed as described in the section "Row and Column Proportions" on page 9840. The overall risk is the overall proportion in the column, and its variance computation is described in the section "Proportions" on page 9839. Confidence limits for the column 1 risk for row 1 are computed as

$$\widehat{P}_{11}^{(1)} \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\widehat{P}_{11}^{(1)}) \right)$$

where $\mathrm{StdErr}(\widehat{P}_{11}^{(1)})$ is the standard error of the risk estimate and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits. Confidence limits for the other risks are computed similarly.

The risk difference is defined as the row 1 risk minus the row 2 risk. The estimate of the column 1 risk difference $\widehat{\mathrm{RD}}_1$ is computed as

$$\begin{aligned}\widehat{\mathrm{RD}}_1 &= \widehat{P}_{11}^{(1)} - \widehat{P}_{21}^{(2)} \\ &= \left(\widehat{N}_{11} \,/\, \widehat{N}_{1\cdot}\right) - \left(\widehat{N}_{21} \,/\, \widehat{N}_{2\cdot}\right)\end{aligned}$$

The column 2 risk difference is computed similarly.

PROC SURVEYFREQ estimates the variance of the risk difference by using the variance estimation method that you request. If you request a replication method (bootstrap, BRR, jackknife, or replicate weights), the procedure estimates the variance as described in the section "Replication Variance Estimation" on page 9828. By default, PROC SURVEYFREQ estimates the variance by using the Taylor series method.

By using Taylor series linearization, the variance estimate for the column 1 risk difference $\widehat{\mathrm{Var}}(\widehat{\mathrm{RD}}_1)$ can be expressed as

$$\widehat{\mathrm{Var}}(\widehat{\mathrm{RD}}_1) = \widehat{\mathbf{D}}\,\widehat{\mathbf{V}}(\widehat{\mathbf{X}})\,\widehat{\mathbf{D}}'$$

where $\widehat{\mathbf{V}}(\widehat{\mathbf{X}})$ is the covariance matrix of $\widehat{\mathbf{X}}$,

$$\widehat{\mathbf{X}} = \left(\widehat{N}_{11},\ \widehat{N}_{1\cdot},\ \widehat{N}_{21},\ \widehat{N}_{2\cdot}\right)$$

and $\widehat{\mathbf{D}}$ is an array that contains the partial derivatives of the risk difference with respect to the elements of $\widehat{\mathbf{X}}$,

$$\widehat{\mathbf{D}} = \left(1/\widehat{N}_{1\cdot},\quad -\widehat{N}_{11}/\widehat{N}_{1\cdot}^{2},\quad -1/\widehat{N}_{2\cdot},\quad -\widehat{N}_{21}/\widehat{N}_{2\cdot}^{2}\right)$$

For more information, see Wolter (1985, pp. 239–242). The variance estimate for the column 2 risk difference is computed similarly.

The standard error of the column 1 risk difference is

$$\mathrm{StdErr}(\widehat{\mathrm{RD}}_1) = \sqrt{\widehat{\mathrm{Var}}(\widehat{\mathrm{RD}}_1)}$$

Confidence limits for the column 1 risk difference are computed as

$$\widehat{\mathrm{RD}}_1 \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\widehat{\mathrm{RD}}_1)\right)$$

where $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits. Confidence limits for the column 2 risk difference are computed in the same way.

Odds Ratio and Relative Risks

The OR option provides estimates of the odds ratio, the column 1 relative risk, and the column 2 relative risk for 2×2 tables, together with their confidence limits.

Odds Ratio

For a 2×2 table, the odds of a positive (column 1) response in row 1 is N_{11}/N_{12}. Similarly, the odds of a positive response in row 2 is N_{21}/N_{22}. The odds ratio is formed as the ratio of the row 1 odds to the row 2 odds. The estimate of the odds ratio is computed as

$$\widehat{\mathrm{OR}} = \frac{\widehat{N}_{11} \,/\, \widehat{N}_{12}}{\widehat{N}_{21} \,/\, \widehat{N}_{22}} = \frac{\widehat{N}_{11}\,\widehat{N}_{22}}{\widehat{N}_{12}\,\widehat{N}_{21}}$$

The value of the odds ratio can be any nonnegative number. When the row and column variables are independent, the true value of the odds ratio is 1. An odds ratio greater than 1 indicates that the odds of a positive response are higher in row 1 than in row 2. An odds ratio less than 1 indicates that the odds of positive response are higher in row 2. The strength of association increases with the deviation from 1. For more information, see Stokes, Davis, and Koch (2000) and Agresti (2007).

PROC SURVEYFREQ constructs confidence limits for the odds ratio by using the log transform. The $100(1-\alpha)\%$ confidence limits for the odds ratio are computed as

$$\left(\widehat{\mathrm{OR}} \times \exp(-t_{df,\alpha/2}\,\sqrt{v}),\ \widehat{\mathrm{OR}} \times \exp(t_{df,\alpha/2}\,\sqrt{v}) \right)$$

where

$$v = \widehat{\mathrm{Var}}(\log(\widehat{\mathrm{OR}})) = \widehat{\mathrm{Var}}(\widehat{\mathrm{OR}}) \,/\, \widehat{\mathrm{OR}}^{\,2}$$

is the estimate of the variance of the log odds ratio and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

PROC SURVEYFREQ estimates the variance of the odds ratio by using the variance estimation method that you request. If you request a replication variance estimation method (bootstrap, BRR, jackknife, or replicate weights), PROC SURVEYFREQ estimates the variance of the odds ratio as described in the section "Replication Variance Estimation" on page 9828. The default variance estimation method is Taylor series.

By using Taylor series linearization, the variance estimate for the odds ratio can be expressed as

$$\widehat{\mathrm{Var}}(\widehat{\mathrm{OR}}) = \widehat{\mathbf{D}}\,\widehat{\mathbf{V}}(\widehat{\mathbf{N}})\,\widehat{\mathbf{D}}'$$

where $\widehat{\mathbf{V}}(\widehat{\mathbf{N}})$ is the covariance matrix of the estimates of the cell totals $\widehat{\mathbf{N}}$,

$$\widehat{\mathbf{N}} = \left(\widehat{N}_{11},\ \widehat{N}_{12},\ \widehat{N}_{21},\ \widehat{N}_{22} \right)$$

and $\widehat{\mathbf{D}}$ is an array that contains the partial derivatives of the odds ratio with respect to the elements of $\widehat{\mathbf{N}}$. The section "Covariances of Frequency Estimates" on page 9838 describes the computation of $\widehat{\mathbf{V}}(\widehat{\mathbf{N}})$. The array $\widehat{\mathbf{D}}$ is computed as

$$\widehat{\mathbf{D}} = \left(\widehat{N}_{22}/(\widehat{N}_{12}\widehat{N}_{21}),\quad -\widehat{N}_{11}\widehat{N}_{22}/(\widehat{N}_{21}\widehat{N}_{12}^{2}),\quad -\widehat{N}_{11}\widehat{N}_{22}/(\widehat{N}_{12}\widehat{N}_{21}^{2}),\quad \widehat{N}_{11}/(\widehat{N}_{12}\widehat{N}_{21}) \right)$$

For more information, see Wolter (1985, pp. 239–242).

Relative Risks

For a 2×2 table, the column 1 relative risk is the ratio of the column 1 risks for row 1 to row 2. As described in the section "Risks and Risk Difference" on page 9850, the column 1 risk for row 1 is the proportion of row 1 observations classified in column 1, and the column 1 risk for row 2 is the proportion of row 2 observations classified in column 1. The estimate of the column 1 relative risk is computed as

$$\widehat{\mathrm{RR}}_1 = \frac{\widehat{N}_{11} \,/\, \widehat{N}_{1\cdot}}{\widehat{N}_{21} \,/\, \widehat{N}_{2\cdot}}$$

Similarly, the estimate of the column 2 relative risk is computed as

$$\widehat{\mathrm{RR}}_2 = \frac{\widehat{N}_{12} \,/\, \widehat{N}_{1\cdot}}{\widehat{N}_{22} \,/\, \widehat{N}_{2\cdot}}$$

A relative risk greater than 1 indicates that the probability of positive response is greater in row 1 than in row 2. Similarly, a relative risk less than 1 indicates that the probability of positive response is less in row 1 than in row 2. The strength of association increases with the deviation from 1. For more information, see Stokes, Davis, and Koch (2000) and Agresti (2007).

PROC SURVEYFREQ constructs confidence limits for the relative risk by using the log transform, which is similar to the odds ratio computations described previously. The $100(1-\alpha)\%$ confidence limits for the column 1 relative risk are computed as

$$\left(\widehat{\mathrm{RR}}_1 \times \exp(-t_{df,\alpha/2}\sqrt{v}),\ \widehat{\mathrm{RR}}_1 \times \exp(t_{df,\alpha/2}\sqrt{v}) \right)$$

where

$$v = \widehat{\mathrm{Var}}(\log(\widehat{\mathrm{RR}}_1)) = \widehat{\mathrm{Var}}(\widehat{\mathrm{RR}}_1) \,/\, \widehat{\mathrm{RR}}_1^{\,2}$$

is the estimate of the variance of the log column 1 relative risk and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

PROC SURVEYFREQ estimates the variance of the relative risk by using the variance estimation method that you request. If you request a replication variance estimation method (bootstrap, BRR, jackknife, or replicate weights), PROC SURVEYFREQ estimates the variance of the relative risk as described in the section "Replication Variance Estimation" on page 9828. The default variance estimation method is Taylor series.

By using Taylor series linearization, the variance estimate for the column 1 relative risk can be expressed as

$$\widehat{\mathrm{Var}}(\widehat{\mathrm{RR}}_1) = \widehat{\mathbf{D}}\,\widehat{\mathbf{V}}(\widehat{\mathbf{X}})\,\widehat{\mathbf{D}}'$$

where $\widehat{\mathbf{V}}(\widehat{\mathbf{X}})$ is the covariance matrix of $\widehat{\mathbf{X}}$,

$$\widehat{\mathbf{X}} = \left(\widehat{N}_{11},\ \widehat{N}_{1\cdot},\ \widehat{N}_{21},\ \widehat{N}_{2\cdot} \right)$$

and $\widehat{\mathbf{D}}$ is an array that contains the partial derivatives of the column 1 relative risk with respect to the elements of $\widehat{\mathbf{X}}$,

$$\widehat{\mathbf{D}} = \left(\widehat{N}_{2\cdot}/(\widehat{N}_{21}\widehat{N}_{1\cdot}),\quad -\widehat{N}_{11}\widehat{N}_{2\cdot}/(\widehat{N}_{21}\widehat{N}_{1\cdot}^{\,2}),\quad -\widehat{N}_{11}\widehat{N}_{2\cdot}/(\widehat{N}_{1\cdot}\widehat{N}_{21}^{\,2}),\quad \widehat{N}_{11}/(\widehat{N}_{21}\widehat{N}_{1\cdot}) \right)$$

For more information, see Wolter (1985, pp. 239–242).

Confidence limits for the column 2 relative risk are computed similarly.

Discordant Proportion Difference

The DISCORDDIFF option provides the estimate of the difference between discordant proportions for 2×2 tables (McNemar 1947). The difference is computed as

$$\widehat{D} = \widehat{P}_{12} - \widehat{P}_{21}$$

where the proportion estimates are described in the section "Proportions" on page 9839.

PROC SURVEYFREQ estimates the variance of the difference by using the variance estimation method that you request. If you request a replication method (bootstrap, BRR, jackknife, or replicate weights), the procedure estimates the variance as described in the section "Replication Variance Estimation" on page 9828. By default, PROC SURVEYFREQ estimates the variance by using the Taylor series method.

Confidence limits for the discordant difference are computed as

$$\widehat{D} \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\widehat{D}) \right)$$

where $\mathrm{StdErr}(\widehat{D})$ is the standard error of the estimate and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

PROC SURVEYFREQ also provides the corresponding t test of the null hypothesis that the discordant proportion difference is 0. The test statistic is computed as

$$t = \widehat{D} \,/\, \mathrm{StdErr}(\widehat{D})$$

The two-sided p-value is

$$\mathrm{Prob}(|T| > |t|)$$

where T has a t distribution with df degrees of freedom.

Kappa Coefficients

Simple Kappa Coefficient

The KAPPA option (or the AGREE option) provides an estimate of the simple kappa coefficient, its standard error, and the confidence limits. This statistic is available for replication variance estimation methods (which you can request by specifying VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE or by including a REPWEIGHTS statement).

The simple kappa coefficient (Cohen 1960) is a measure of interrater agreement, where the row and column variables of the two-way table are viewed as two independent ratings. When there is perfect agreement between the two ratings, the kappa coefficient is +1. When the observed agreement exceeds chance agreement, the value of kappa is positive, and its magnitude reflects the strength of agreement. The minimum value of kappa is between –1 and 0, depending on the marginal proportions. For more information, see Fleiss, Levin, and Paik (2003).

PROC SURVEYFREQ computes the simple kappa coefficient as

$$\hat{\kappa} = (P_o - P_e) \;/\; (1 - P_e)$$

where

$$P_o = \sum_i \widehat{P}_{ii}$$

$$P_e = \sum_i \widehat{P}_{i\cdot}\widehat{P}_{\cdot i}$$

where $\widehat{P}_{ii}$ is the estimate of the proportion in table cell (i, i), $\widehat{P}_{i\cdot}$ is the estimate of the proportion in row i, and $\widehat{P}_{\cdot i}$ is the estimate of the proportion in column i. For information about how PROC SURVEYFREQ computes the proportion estimates, see the section "Proportions" on page 9839.

PROC SURVEYFREQ estimates the variance of the simple kappa coefficient as described in the section "Replication Variance Estimation" on page 9828. PROC SURVEYFREQ computes confidence limits for the simple kappa coefficient as

$$\hat{\kappa} \pm (\, t_{df,\alpha/2} \times \mathrm{StdErr}(\hat{\kappa})\,)$$

where $\mathrm{StdErr}(\hat{\kappa})$ is the standard error of the kappa coefficient and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

Kappa Details The kappa component P_o measures the observed agreement, and the component P_e measures the chance-expected agreement. These values are displayed in the "Kappa Details" table when you specify the AGREE(DETAILS) or KAPPA(DETAILS) option. The "Kappa Details" table also displays the maximum possible value of the simple kappa coefficient given the marginal proportions of the two-way table. The maximum kappa is computed as

$$\max(\kappa) = (\max(P_o) - P_e) \;/\; (1 - P_e)$$

where

$$\max(P_o) = \left(\sum_i \min(\hat{N}_{i\cdot}, \hat{N}_{\cdot i})\right) / \hat{N}$$

The "Kappa Details" table displays the B_n measure (Bangdiwala 1988; Bangdiwala et al. 2008), which is computed as

$$B_n = \left(\sum_i \widehat{N}_{ii}^2\right) / \left(\sum_i \sum_j \widehat{N}_{i\cdot}\widehat{N}_{\cdot i}\right)$$

When the two-way table is 2×2, the "Kappa Details" table displays the prevalence index and the bias index. The prevalence index is the absolute difference between the agreement proportions, $|\widehat{P}_{11} - \widehat{P}_{22}|$. The bias index is the absolute difference between the disagreement proportions, $|\widehat{P}_{12} - \widehat{P}_{21}|$. For more information, see Sim and Wright (2005) and Byrt, Bishop, and Carlin (1993).

Weighted Kappa Coefficient

The WTKAPPA option (or the AGREE option) provides an estimate of the weighted kappa coefficient, its standard error, and the confidence limits. This statistic is available for replication variance estimation methods.

The weighted kappa coefficient is a generalization of the simple kappa coefficient that uses agreement weights to quantify the relative difference between categories (levels). By default, PROC SURVEYFREQ uses Cicchetti-Allison agreement weights to compute the weighted kappa coefficient; if you specify the WTKAPPA(WT=FC) option, the procedure uses Fleiss-Cohen agreement weights. For information about how the agreement weights are computed, see the section "Kappa Agreement Weights" on page 9856. For more information, see Fleiss, Cohen, and Everitt (1969) and Fleiss, Levin, and Paik (2003).

For 2×2 tables, the weighted kappa coefficient equals the simple kappa coefficient; PROC SURVEYFREQ displays the weighted kappa coefficient only for tables larger than 2×2.

PROC SURVEYFREQ computes the weighted kappa coefficient as

$$\hat{\kappa}_w = \left(P_{o(w)} - P_{e(w)}\right) \;/\; \left(1 - P_{e(w)}\right)$$

where

$$P_{o(w)} = \sum_i \sum_j w_{ij} \widehat{P}_{ij}$$

$$P_{e(w)} = \sum_i \sum_j w_{ij} \widehat{P}_{i.} \widehat{P}_{.j}$$

where w_{ij} is the agreement weight for table cell (i, j), $\widehat{P}_{ij}$ is the estimate of the proportion in table cell (i, j), $\widehat{P}_{i.}$ is the estimate of the proportion in row i, and $\widehat{P}_{.i}$ is the estimate of the proportion in column i. For information about how PROC SURVEYFREQ computes the proportion estimates, see the section "Proportions" on page 9839.

The weighted kappa component $P_{o(w)}$ measures the observed agreement, and the component $P_{e(w)}$ measures the chance-expected agreement. These values are displayed in the "Weighted Kappa Details" table when you specify the AGREE(DETAILS) or WTKAPPA(DETAILS) option.

PROC SURVEYFREQ estimates the variance of the simple kappa coefficient as described in the section "Replication Variance Estimation" on page 9828. PROC SURVEYFREQ computes confidence limits for the weighted kappa coefficient as

$$\hat{\kappa}_w \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\hat{\kappa}_w) \right)$$

where $\mathrm{StdErr}(\hat{\kappa}_w)$ is the standard error of the weighted kappa coefficient and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

Kappa Agreement Weights PROC SURVEYFREQ computes the weighted kappa coefficient by using the Cicchetti-Allison form (by default) or the Fleiss-Cohen form of agreement weights. These weights are based on the scores of the column variable in the two-way table request. If the column variable is numeric, the column scores are the numeric values of the column levels. If the column variable is a character variable,

the column scores are the column numbers, where the columns are numbered in the order in which they appear in the crosstabulation table.

PROC SURVEYFREQ computes Cicchetti-Allison agreement weights as

$$w_{ij} = 1 - \left(|C_i - C_j| \,/\, (C_c - C_1)\right)$$

where C_i is the score for column i and c is the number of columns (categories). For more information, see Cicchetti and Allison (1971).

PROC SURVEYFREQ computes Fleiss-Cohen agreement weights as

$$w_{ij} = 1 - \left((C_i - C_j) \,/\, (C_c - C_1)\right)^2$$

For more information, see Fleiss and Cohen (1973).

The agreement weights w_{ij} are constructed so that $w_{ii} = 1$ for all i, and $w_{ij} = w_{ji}$. For $i \neq j$, the agreement weights must be nonnegative and less than 1, which is always true for character variables (where the scores are the column numbers). For numeric variables, you should assign numeric variable levels (scores) so that all agreement weights are nonnegative and less than 1.

You can assign numeric values to the variable levels in a way that reflects their degree of similarity. For example, suppose the column variable is numeric and has four levels, which you order according to similarity. If you assign the values 0, 2, 4, and 10 to the column variable levels, the Cicchetti-Allison agreement weights take the following values: $w_{12} = 0.8$, $w_{13} = 0.6$, $w_{14} = 0.0$, $w_{23} = 0.8$, $w_{24} = 0.2$, and $w_{34} = 0.4$. For this example, the Fleiss-Cohen agreement weights are as follows: $w_{12} = 0.96$, $w_{13} = 0.84$, $w_{14} = 0.00$, $w_{23} = 0.96$, $w_{24} = 0.36$, and $w_{34} = 0.64$.

To display the kappa agreement weights, you can specify the WTKAPPA(PRINTKWTS) option.

AC1 Agreement Coefficient

When you specify the AGREE(AC1) option, PROC SURVEYFREQ provides Gwet's first-order agreement coefficient, AC1 (Gwet 2008), its standard error, and the confidence limits. This option is available for replication variance estimation methods.

The AC1 agreement coefficient is computed as

$$\hat{\gamma} = \left(P_o - P_{e(\gamma)}\right) \;/\; \left(1 - P_{e(\gamma)}\right)$$

where $P_o = \sum_i \widehat{P}_{ii}$, $P_e = \sum_i e_i(1 - e_i)/(R - 1)$, and $e_i = (\widehat{P}_{i\cdot} + \widehat{P}_{\cdot i})/2$. The component P_o is the proportion of observed agreement, and the component $P_{e(\gamma)}$ represents the proportion of chance-expected agreement. For more information, see Xie (2013) and Blood and Spratt (2007).

PROC SURVEYFREQ estimates the variance of the AC1 coefficient as described in the section "Replication Variance Estimation" on page 9828 and computes the confidence limits as

$$\hat{\gamma} \pm \left(t_{df,\alpha/2} \times \mathrm{StdErr}(\hat{\gamma})\right)$$

where $\mathrm{StdErr}(\hat{\gamma})$ is the standard error of the AC1 coefficient and $t_{df,\alpha/2}$ is the $100(1 - \alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

Prevalence-Adjusted Bias-Adjusted Kappa Coefficient

When you specify the AGREE(PABAK) option, PROC SURVEYFREQ provides the prevalence-adjusted bias-adjusted kappa coefficient (PABAK) (Byrt, Bishop, and Carlin 1993). This statistic is available for replication variance estimation methods.

This coefficient is computed as

$$\hat{\kappa}_a = (P_o - 1/R) \ / \ (1 - 1/R)$$

where $P_o = \sum_i \widehat{P}_{ii}$ and R is the dimension of the square, two-way table. The component P_o is the proportion of observed agreement, and the component $1/R$ represents the chance-expected agreement. When the table is 2×2, $\hat{\kappa}_a = 2P_o - 1$. For more information, see Sim and Wright (2005), Xie (2013), and Holley and Guilford (1964).

PROC SURVEYFREQ estimates the variance of PABAK as described in the section "Replication Variance Estimation" on page 9828 and computes the confidence limits as

$$\hat{\kappa}_a \pm (\, t_{df,\alpha/2} \times \mathrm{StdErr}(\hat{\kappa}_a)\,)$$

where $\mathrm{StdErr}(\hat{\kappa}_a)$ is the standard error of PABAK coefficient and $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) The value of the confidence coefficient α is determined by the ALPHA= option; by default, ALPHA=0.05, which produces 95% confidence limits.

Rao-Scott Chi-Square Test

The Rao-Scott chi-square test is a design-adjusted version of the Pearson chi-square test, which involves differences between observed and expected frequencies. For information about design-adjusted chi-square tests, see Lohr (2022, Section 10.3.2), Rao and Scott (1981), Rao and Scott (1984), Rao and Scott (1987), and Thomas, Singh, and Roberts (1996).

PROC SURVEYFREQ provides a first-order Rao-Scott chi-square test by default. If you specify the CHISQ(SECONDORDER) option, PROC SURVEYFREQ provides a second-order (Satterthwaite) Rao-Scott chi-square test. The first-order design correction depends only on the design effects of the table cell proportion estimates and, for two-way tables, the design effects of the marginal proportion estimates. The second-order design correction requires computation of the full covariance matrix of the proportion estimates. The second-order test requires more computational resources than the first-order test, but it can provide some performance advantages (for Type I error and power), particularly when the design effects are variable (Thomas and Rao 1987; Rao and Thomas 1989).

One-Way Tables

For one-way tables, the CHISQ option provides a Rao-Scott (design-based) goodness-of-fit test for one-way tables. By default, this is a test for the null hypothesis of equal proportions. If you specify null hypothesis proportions in the TESTP= option, the goodness-of-fit test uses the specified proportions.

First-Order Test The first-order Rao-Scott chi-square statistic for the goodness-of-fit test is computed as

$$Q_{\mathrm{RS1}} = Q_P \ / \ D$$

where Q_P is the Pearson chi-square based on the estimated totals and D is the first-order design correction described in the section "First-Order Design Correction" on page 9859. For more information, see Rao and Scott (1979), Rao and Scott (1981), Rao and Scott (1984).

For a one-way table with C levels, the Pearson chi-square is computed as

$$Q_P = (n/\widehat{N}) \sum_c (\widehat{N}_c - E_c)^2 \,/\, E_c$$

where n is the sample size, $\widehat{N}$ is the estimated overall total, $\widehat{N}_c$ is the estimated total for level c, and E_c is the expected total for level c under the null hypothesis. For the null hypothesis of equal proportions, the expected total for each level is

$$E_c = \widehat{N} \,/\, C$$

For specified null proportions, the expected total for level c is

$$E_c = \widehat{N} \times P_c^{\,0}$$

where $P_c^{\,0}$ is the null proportion that you specify for level c.

Under the null hypothesis, the first-order Rao-Scott chi-square Q_{RS1} approximately follows a chi-square distribution with $(C - 1)$ degrees of freedom. A better approximation can be obtained by the F statistic,

$$F_1 = Q_{\mathrm{RS1}} \,/\, (C - 1)$$

which has an F distribution with $(C - 1)$ and $\kappa(C - 1)$ degrees of freedom under the null hypothesis (Thomas and Rao 1984, 1987). The value of κ is the degrees of freedom for the variance estimator. The degrees of freedom computation depends on the sample design and the variance estimation method. For more information, see the section "Degrees of Freedom" on page 9846.

First-Order Design Correction By default for one-way tables, the first-order design correction is computed from the proportion estimates as

$$D = \sum_c (1 - \widehat{P}_c)\, \mathrm{Deff}(\widehat{P}_c) \,/\, (C - 1)$$

where

$$\begin{aligned} \mathrm{Deff}(\widehat{P}_c) &= \widehat{\mathrm{Var}}(\widehat{P}_c) \,/\, \widehat{\mathrm{Var}}_{\mathrm{srs}}(\widehat{P}_c) \\ &= \widehat{\mathrm{Var}}(\widehat{P}_c) \,/\, \Big((1 - f)\, \widehat{P}_c\, (1 - \widehat{P}_c) \,/\, (n - 1)\Big) \end{aligned}$$

as described in the section "Design Effect" on page 9847. $\widehat{P}_c$ is the proportion estimate for level c, $\widehat{\mathrm{Var}}(\widehat{P}_c)$ is the variance of the estimate, f is the overall sampling fraction, and n is the number of observations in the sample. The factor $(1 - f)$ is included only for Taylor series variance estimation (VARMETHOD=TAYLOR) when you specify the RATE= or TOTAL= option. For more information, see the section "Design Effect" on page 9847.

If you specify the CHISQ(MODIFIED) or LRCHISQ(MODIFIED) option, the design correction is computed by using null hypothesis proportions instead of proportion estimates. By default, null hypothesis proportions are equal proportions for all levels of the one-way table. Alternatively, you can specify null proportion values in the TESTP= option. The modified design correction D_0 is computed from null hypothesis proportions as

$$D_0 = \sum_c (1 - P_c^{\,0})\, \mathrm{Deff}_0(\widehat{P}_c) \,/\, (C - 1)$$

where

$$\begin{aligned}\mathrm{Deff}_0(\widehat{P}_c) &= \widehat{\mathrm{Var}}(\widehat{P}_c) \,/\, \widehat{\mathrm{Var}}_{\mathrm{srs}}({P_c}^0) \\ &= \widehat{\mathrm{Var}}(\widehat{P}_c) \,/\, \big((1-f){P_c}^0\,(1-{P_c}^0)\,/\,(n-1)\big)\end{aligned}$$

The null hypothesis proportion ${P_c}^0$ is $1/C$ for equal proportions (the default), or ${P_c}^0$ is the null proportion that you specify for level c if you use the TESTP= option.

Second-Order Test The second-order (Satterthwaite) Rao-Scott chi-square statistic for the goodness-of-fit test is computed as

$$Q_{\mathrm{RS2}} = Q_{\mathrm{RS1}} \,/\, (1+\hat{a}^2)$$

where Q_{RS1} is the first-order Rao-Scott chi-square statistic described in the section "First-Order Test" on page 9858 and $\hat{a}^2$ is the second-order design correction described in the section "Second-Order Design Correction" on page 9860. For more information, see Rao and Scott (1979), Rao and Scott (1981), and Rao and Thomas (1989).

Under the null hypothesis, the second-order Rao-Scott chi-square Q_{RS2} approximately follows a chi-square distribution with $(C-1)/(1+\hat{a}^2)$ degrees of freedom. The corresponding F statistic is

$$F_{\mathrm{RS2}} = Q_{\mathrm{RS2}} \,/\, (C-1)$$

which has an F distribution with $(C-1)/(1+\hat{a}^2)$ and $\kappa(C-1)/(1+\hat{a}^2)$ degrees of freedom under the null hypothesis (Thomas and Rao 1984, 1987). The value of κ is the degrees of freedom for the variance estimator. The degrees of freedom computation depends on the sample design and the variance estimation method. For more information, see the section "Degrees of Freedom" on page 9846.

Second-Order Design Correction The second-order (Satterthwaite) design correction for one-way tables is computed from the eigenvalues of the estimated design effects matrix $\widehat{\mathbf{\Delta}}$, which are known as *generalized design effects*. The design effects matrix is computed as

$$\widehat{\mathbf{\Delta}} \;=\; (n-1)/(1-f)\,\Big(\mathbf{Cov}_{\mathbf{srs}}(\widehat{\mathbf{P}})^{-1}\;\widehat{\mathbf{Cov}}(\widehat{\mathbf{P}})\Big)$$

where $\mathbf{Cov}_{\mathbf{srs}}(\widehat{\mathbf{P}})$ is the covariance under multinomial sampling (*srs* with replacement) and $\widehat{\mathbf{Cov}}(\widehat{\mathbf{P}})$ is the covariance matrix of the first $(C-1)$ proportion estimates. For more information, see Rao and Scott (1979), Rao and Scott (1981), and Rao and Thomas (1989).

By default, the *srs* covariance matrix is computed from the proportion estimates as

$$\mathbf{Cov}_{\mathbf{srs}}(\widehat{\mathbf{P}}) \;=\; \mathrm{Diag}(\widehat{\mathbf{P}}) - \widehat{\mathbf{P}}\,\widehat{\mathbf{P}}'$$

where $\widehat{\mathbf{P}}$ is an array of $(C-1)$ proportion estimates. If you specify the CHISQ(MODIFIED) or LRCHISQ(MODIFIED) option, the *srs* covariance matrix is computed from the null hypothesis proportions $\mathbf{P_0}$ as

$$\mathbf{Cov}_{\mathbf{srs}}(\mathbf{P_0}) \;=\; \mathrm{Diag}(\mathbf{P_0}) - \mathbf{P_0}\,\mathbf{P_0}'$$

where $\mathbf{P_0}$ is an array of $(C-1)$ null hypothesis proportions. The null hypothesis proportions equal $1/C$ by default. If you use the TESTP= option to specify null hypothesis proportions, $\mathbf{P_0}$ is an array of $(C-1)$ proportions that you specify.

The second-order design correction is computed as

$$\hat{a}^2 = \left(\sum_{c=1}^{C-1} d_c^2 / (C-1)\bar{d}^{\,2} \right) - 1$$

where d_c are the eigenvalues of the design effects matrix $\widehat{\mathbf{\Delta}}$ and $\bar{d}$ is the average of the eigenvalues.

Two-Way Tables

For two-way tables, the CHISQ option provides a Rao-Scott (design-based) test of association between the row and column variables. PROC SURVEYFREQ provides a first-order Rao-Scott chi-square test by default. If you specify the CHISQ(SECONDORDER) option, PROC SURVEYFREQ provides a second-order (Satterthwaite) Rao-Scott chi-square test.

First-Order Test The first-order Rao-Scott chi-square statistic is computed as

$$Q_{\mathrm{RS1}} = Q_P \,/\, D$$

where Q_P is the Pearson chi-square based on the estimated totals and D is the design correction described in the section "First-Order Design Correction" on page 9861. For more information, see Rao and Scott (1979), Rao and Scott (1984), and Rao and Scott (1987).

For a two-way tables with R rows and C columns, the Pearson chi-square is computed as

$$Q_P = (n/\widehat{N}) \sum_r \sum_c (\widehat{N}_{rc} - E_{rc})^2 \,/\, E_{rc}$$

where n is the sample size, $\widehat{N}$ is the estimated overall total, $\widehat{N}_{rc}$ is the estimated total for table cell (r, c), and E_{rc} is the expected total for table cell (r,c) under the null hypothesis of no association,

$$E_{rc} = \widehat{N}_{r\cdot}\, \widehat{N}_{\cdot c} \,/\, \widehat{N}$$

Under the null hypothesis of no association, the first-order Rao-Scott chi-square Q_{RS1} approximately follows a chi-square distribution with $(R-1)(C-1)$ degrees of freedom. A better approximation can be obtained by the F statistic,

$$F_1 = Q_{\mathrm{RS1}} \,/\, (R-1)(C-1)$$

which has an F distribution with $(R-1)(C-1)$ and $\kappa(R-1)(C-1)$ degrees of freedom under the null hypothesis (Thomas and Rao 1984, 1987). The value of κ is the degrees of freedom for the variance estimator. The degrees of freedom computation depends on the sample design and the variance estimation method. For more information, see the section "Degrees of Freedom" on page 9846.

First-Order Design Correction By default for a first-order test, PROC SURVEYFREQ computes the design correction from proportion estimates. If you specify the CHISQ(MODIFIED) or LRCHISQ(MODIFIED) option for a first-order test, the procedure computes the design correction from null hypothesis proportions.

Second-order tests, which you request by specifying the CHISQ(SECONDORDER) or LRCHISQ(SECONDORDER) option, are computed by applying both first-order and second-order design corrections to the weighted chi-square statistic. For second-order tests for two-way tables, PROC

SURVEYFREQ always uses null hypothesis proportions to compute both the first-order and second-order design corrections.

The first-order design correction D that is based on proportion estimates is computed as

$$D = \Big(\sum_r \sum_c (1 - \widehat{P}_{rc}) \operatorname{Deff}(\widehat{P}_{rc}) - \sum_r (1 - \widehat{P}_{r\cdot}) \operatorname{Deff}(\widehat{P}_{r\cdot}) - \sum_c (1 - \widehat{P}_{\cdot c}) \operatorname{Deff}(\widehat{P}_{\cdot c}) \Big) \,/\, (R-1)(C-1)$$

where

$$\begin{aligned} \operatorname{Deff}(\widehat{P}_{rc}) &= \widehat{\operatorname{Var}}(\widehat{P}_{rc}) \,/\, \widehat{\operatorname{Var}}_{\text{srs}}(\widehat{P}_{rc}) \\ &= \widehat{\operatorname{Var}}(\widehat{P}_{rc}) \,/\, \Big((1-f)\, \widehat{P}_{rc}\, (1 - \widehat{P}_{rc}) \,/\, (n-1) \Big) \end{aligned}$$

as described in the section "Design Effect" on page 9847. $\widehat{P}_{rc}$ is the estimate of the proportion in table cell (r, c), $\widehat{\operatorname{Var}}(\widehat{P}_{rc})$ is the variance of the estimate, f is the overall sampling fraction, and n is the number of observations in the sample. The factor $(1 - f)$ is included only for Taylor series variance estimation (VARMETHOD=TAYLOR) when you specify the RATE= or TOTAL= option. For more information, see the section "Design Effect" on page 9847.

The design effects for the estimate of the proportion in row r and the estimate of the proportion in column c ($\operatorname{Deff}(\widehat{P}_{r\cdot})$ and $\operatorname{Deff}(\widehat{P}_{\cdot c})$, respectively) are computed in the same way.

If you specify the CHISQ(MODIFIED) or LRCHISQ(MODIFIED) option for a first-order Rao-Scott test, or if you request a second-order test for a two-way table (CHISQ(SECONDORDER) or LRCHISQ(SECONDORDER)), the procedure computes the design correction from the null hypothesis cell proportions instead of the estimated cell proportions. For two-way tables, the null hypothesis cell proportions are computed as the products of the corresponding row and column proportion estimates. The modified design correction D_0 (based on null hypothesis proportions) is computed as

$$D_0 = \Big(\sum_r \sum_c (1 - P^0_{rc}) \operatorname{Deff}_0(\widehat{P}_{rc}) - \sum_r (1 - \widehat{P}_{r\cdot}) \operatorname{Deff}(\widehat{P}_{r\cdot}) - \sum_c (1 - \widehat{P}_{\cdot c}) \operatorname{Deff}(\widehat{P}_{\cdot c}) \Big) \,/\, (R-1)(C-1)$$

where

$$P^0_{rc} = \widehat{P}_{r\cdot} \times \widehat{P}_{\cdot c}$$

and

$$\begin{aligned} \operatorname{Deff}_0(\widehat{P}_{rc}) &= \widehat{\operatorname{Var}}(\widehat{P}_{rc}) \,/\, \operatorname{Var}_{\text{srs}}(P^0_{rc}) \\ &= \widehat{\operatorname{Var}}(\widehat{P}_{rc}) \,/\, \big((1-f)\, P^0_{rc}\, (1 - P^0_{rc}) \,/\, (n-1) \big) \end{aligned}$$

Second-Order Test The second-order (Satterthwaite) Rao-Scott chi-square statistic for two-way tables is computed as

$$Q_{\mathrm{RS2}} = Q_{\mathrm{RS1}} \,/\, (1 + \hat{a}^2)$$

where Q_{RS1} is the first-order Rao-Scott chi-square statistic described in the section "First-Order Test" on page 9861 and $\hat{a}^2$ is the second-order design correction described in the section "Second-Order Design Correction" on page 9863. For more information, see Rao and Scott (1979), Rao and Scott (1981), and Rao and Thomas (1989).

Under the null hypothesis, the second-order Rao-Scott chi-square Q_{RS2} approximately follows a chi-square distribution with $(R-1)(C-1)/(1+\hat{a}^2)$ degrees of freedom. The corresponding F statistic is

$$F_{\mathrm{RS2}} = Q_{\mathrm{RS2}}\,(1 + \hat{a}^2) \,/\, (R-1)(C-1)$$

which has an F distribution with $(R-1)(C-1)/(1+\hat{a}^2)$ and $\kappa(R-1)(C-1)/(1+\hat{a}^2)$ degrees of freedom under the null hypothesis (Thomas and Rao 1984, 1987). The value of κ is the degrees of freedom for the variance estimator. The degrees of freedom computation depends on the sample design and the variance estimation method. For more information, see the section "Degrees of Freedom" on page 9846.

Second-Order Design Correction The second-order (Satterthwaite) design correction for two-way tables is computed from the eigenvalues of the estimated design effects matrix $\widehat{\mathbf{\Delta}}$, which are known as *generalized design effects*. The design effects matrix is defined as

$$\widehat{\mathbf{\Delta}} \;=\; (n-1)/(1-f)\,\left(\mathbf{Cov}_{\mathbf{srs}}(\widehat{\mathbf{P}})^{-1}\,\mathbf{H}\,\widehat{\mathbf{Cov}}(\widehat{\mathbf{P}})\,\mathbf{H}'\right)$$

where $\widehat{\mathbf{Cov}}(\widehat{\mathbf{P}})$ is the covariance matrix of the $R \times C$ proportion estimates and $\mathbf{Cov}_{\mathbf{srs}}(\widehat{\mathbf{P}})$ is the covariance under multinomial sampling (*srs* with replacement). For more information, see Rao and Scott (1979), Rao and Scott (1981), and Rao and Thomas (1989).

The second-order design correction is computed from the design effects matrix $\widehat{\mathbf{\Delta}}$ as

$$\hat{a}^2 \;=\; \left(\sum_{i=1}^{K} d_c^2 / K\bar{d}^{\,2}\right) - 1$$

where $K = (R-1)(C-1)$, d_c are the eigenvalues of $\widehat{\mathbf{\Delta}}$, and $\bar{d}$ is the average eigenvalue.

The *srs* covariance matrix is computed as

$$\mathbf{Cov}_{\mathbf{srs}}(\widehat{\mathbf{P}}) \;=\; \widehat{\mathbf{P}}_{\mathbf{r}} \otimes \widehat{\mathbf{P}}_{\mathbf{c}}$$

where $\widehat{\mathbf{P}}_{\mathbf{r}}$ is an $(R-1) \times (R-1)$ matrix that is constructed from the array of $(R-1)$ row proportion estimates $\hat{\mathbf{p}}_{\mathbf{r}}$ as

$$\widehat{\mathbf{P}}_{\mathbf{r}} \;=\; \mathrm{Diag}(\hat{\mathbf{p}}_{\mathbf{r}}) - \hat{\mathbf{p}}_{\mathbf{r}}\,\hat{\mathbf{p}}_{\mathbf{r}}'$$

Similarly, $\widehat{\mathbf{P}}_{\mathbf{c}}$ is a $(C-1) \times (C-1)$ matrix that is constructed from the array of $(C-1)$ column proportion estimates $\hat{\mathbf{p}}_{\mathbf{c}}$ as

$$\widehat{\mathbf{P}}_{\mathbf{c}} \;=\; \mathrm{Diag}(\hat{\mathbf{p}}_{\mathbf{c}}) - \hat{\mathbf{p}}_{\mathbf{c}}\,\hat{\mathbf{p}}_{\mathbf{c}}'$$

The $(R-1)(C-1) \times (R-1)(C-1)$ matrix $\mathbf{H}$ is computed as

$$\mathbf{H} \;=\; \mathbf{J}_{\mathbf{r}} \otimes \mathbf{J}_{\mathbf{c}} \;-\; (\hat{\mathbf{p}}_{\mathbf{r}}\,\mathbf{1}_{\mathbf{r}}') \otimes \mathbf{J}_{\mathbf{c}} \;-\; \mathbf{J}_{\mathbf{r}} \otimes (\hat{\mathbf{p}}_{\mathbf{r}}\,\mathbf{1}_{\mathbf{r}}')$$

where $\mathbf{J}_{\mathbf{r}} = (\mathbf{I}_{(R-1)}|\mathbf{0})$, $\mathbf{J}_{\mathbf{c}} = (\mathbf{I}_{(C-1)}|\mathbf{0})$, $\mathbf{1}_{\mathbf{r}}$ is an $(R \times 1)$ array of ones, and $\mathbf{1}_{\mathbf{c}}$ is an $(C \times 1)$ array of ones. For more information, see Rao and Scott (1979, p. 61).

Rao-Scott Likelihood Ratio Chi-Square Test

The Rao-Scott likelihood ratio chi-square test is a design-adjusted version of the likelihood ratio test, which involves ratios of observed and expected frequencies. For information about design-adjusted chi-square tests, see Lohr (2022, Section 10.3.2), Rao and Scott (1981), Rao and Scott (1984), Rao and Scott (1987), and Thomas, Singh, and Roberts (1996).

PROC SURVEYFREQ provides a first-order Rao-Scott likelihood ratio test by default. If you specify the LRCHISQ(SECONDORDER) option, PROC SURVEYFREQ provides a second-order (Satterthwaite) likelihood ratio chi-square test.

The procedure computes the Rao-Scott likelihood ratio test by applying design adjustments to the weighted likelihood ratio statistic that is based on estimated totals. This computation is identical to the Rao-Scott chi-square test computation except that it uses the likelihood ratio statistic G^2 in place of the Pearson chi-square statistic Q_P. For more information, see the section "Rao-Scott Chi-Square Test" on page 9858.

One-Way Tables

For one-way tables, the LRCHISQ option provides a Rao-Scott (design-based) goodness-of-fit test for one-way tables. By default, this is a test for the null hypothesis of equal proportions. If you specify null hypothesis proportions in the TESTP= option, the goodness-of-fit test uses the specified proportions.

The Rao-Scott likelihood ratio test uses the likelihood ratio statistic that is based on the estimated totals,

$$G^2 = 2\,(n/\widehat{N}) \sum_c \widehat{N}_c \,\log\left(\widehat{N}_c \,/\, E_c\right)$$

where n is the sample size, $\widehat{N}$ is the estimated overall total, $\widehat{N}_c$ is the estimated total for level c, and E_c is the expected total for level c under the null hypothesis. For the null hypothesis of equal proportions, the expected total for each level is

$$E_c = \widehat{N} \,/\, C$$

For specified null proportions, the expected total for level c is

$$E_c = \widehat{N} \times {P_c}^0$$

where ${P_c}^0$ is the null proportion that you specify for level c.

The computation of the Rao-Scott likelihood ratio test for one-way tables uses G^2 in place of Q_P in the Rao-Scott chi-square test computation and is otherwise identical to the chi-square test computation. For more information, see the sections "First-Order Test" on page 9858 and "Second-Order Test" on page 9860.

If you specify the LRCHISQ(MODIFIED) option, PROC SURVEYFREQ computes the design corrections by using null hypothesis proportions instead of proportion estimates. By default, null hypothesis proportions are equal proportions for all levels of the one-way table. Alternatively, you can specify null proportion values in the TESTP= option.

Two-Way Tables

For two-way tables, the LRCHISQ option provides a Rao-Scott (design-based) test of association between the row and column variables.

The Rao-Scott likelihood ratio test uses the likelihood ratio statistic that is based on the estimated totals,

$$G^2 = 2\,(n/\widehat{N}) \sum_r \sum_c \widehat{N}_{rc} \,\log\left(\widehat{N}_{rc} \,/\, E_{rc}\right)$$

where n is the sample size, $\widehat{N}$ is the estimated overall total, $\widehat{N}_{rc}$ is the estimated total for table cell (r, c), and E_{rc} is the expected total for cell (r, c) under the null hypothesis of no association. The expected total for cell (r, c) is

$$E_{rc} = \widehat{N}_{r\cdot}\, \widehat{N}_{\cdot c} \,/\, \widehat{N}$$

The computation of the Rao-Scott likelihood ratio test for two-way tables uses G^2 in place of Q_P in the Rao-Scott chi-square test computation and is otherwise identical to the chi-square test computation. For more information, see the sections "First-Order Test" on page 9861 and "Second-Order Test" on page 9863.

By default for a first-order test, PROC SURVEYFREQ computes the design correction from proportion estimates. If you specify the LRCHISQ(MODIFIED) option for a first-order test, the procedure computes the design correction from null hypothesis proportions.

Second-order tests, which you request by specifying the LRCHISQ(SECONDORDER) option, are computed by applying both first-order and second-order design corrections to the weighted likelihood ratio statistic. For second-order tests for two-way tables, PROC SURVEYFREQ always uses null hypothesis proportions to compute both the first-order and second-order design corrections.

Wald Chi-Square Test

PROC SURVEYFREQ provides two Wald chi-square tests for independence of the row and column variables in a two-way table: a Wald chi-square test based on the difference between observed and expected weighted cell frequencies, and a Wald log-linear chi-square test based on the log odds ratios. These statistics test for independence of the row and column variables in two-way tables, taking into account the complex survey design. For information about Wald statistics and their applications to categorical data analysis, see Bedrick (1983), Koch, Freeman, and Freeman (1975), and Wald (1943).

For these two tests, PROC SURVEYFREQ computes the generalized Wald chi-square statistic, the corresponding F statistic, and also an adjusted F statistic for tables larger than 2×2. Under the null hypothesis of independence, the Wald chi-square statistic approximately follows a chi-square distribution with $(R-1)(C-1)$ degrees of freedom for large samples. However, it has been shown that this test can perform poorly in terms of actual significance level and power, especially for tables with a large number of cells or for samples with a relatively small number of clusters. For more information, see Thomas and Rao (1984), Thomas and Rao (1985), and Lohr (2022). For information about the adjusted F statistic, see Fellegi (1980) and Hidiroglou, Fuller, and Hickman (1980). Thomas and Rao (1984) found that the adjusted F statistic provides a more stable test than the chi-square statistic, although its power can be low when the number of sample clusters is not large. See also Korn and Graubard (1990) and Thomas, Singh, and Roberts (1996).

If you specify the WCHISQ option in the TABLES statement, PROC SURVEYFREQ computes a Wald test for independence in the two-way table based on the differences between the observed (weighted) cell frequencies and the expected frequencies.

Under the null hypothesis of independence of the row and column variables, the expected cell frequencies are computed as

$$E_{rc} = \widehat{N}_{r\cdot}\, \widehat{N}_{\cdot c} \,/\, \widehat{N}$$

where $\widehat{N}_{r\cdot}$ is the estimated total for row r, $\widehat{N}_{\cdot c}$ is the estimated total for column c, and $\widehat{N}$ is the estimated overall total, as described in the section "Expected Weighted Frequency" on page 9848. The null hypothesis that the population weighted frequencies equal the expected frequencies can be expressed as

$$H_0\colon Y_{rc} = N_{rc} - E_{rc} = 0$$

for all $r = 1, \ldots (R-1)$ and $c = 1, \ldots (C-1)$. This null hypothesis can be stated equivalently in terms of cell proportions, with the expected cell proportions computed as the products of the marginal row and column proportions.

The generalized Wald chi-square statistic Q_W is computed as

$$Q_W = \widehat{\mathbf{Y}}' \left(\mathbf{H}\, \widehat{\mathbf{V}}(\widehat{\mathbf{N}})\, \mathbf{H}'\right)^{-1} \widehat{\mathbf{Y}}$$

where $\widehat{\mathbf{Y}}$ is an array of $(R-1)(C-1)$ differences between the observed and expected weighted frequencies $(\widehat{N}_{rc} - E_{rc})$, and $(\mathbf{H}\, \widehat{\mathbf{V}}(\widehat{\mathbf{N}})\, \mathbf{H}')$ estimates the variance of $\widehat{\mathbf{Y}}$.

$\widehat{\mathbf{V}}(\widehat{\mathbf{N}})$ is the covariance matrix of the estimates $\widehat{N}_{rc}$, and its computation is described in the section "Covariances of Frequency Estimates" on page 9838.

$\mathbf{H}$ is an $(R-1)(C-1)$ by RC matrix that contains the partial derivatives of the elements of $\widehat{\mathbf{Y}}$ with respect to the elements of $\widehat{\mathbf{N}}$. The elements of $\mathbf{H}$ are computed as follows, where a denotes a row different from row r, and b denotes a column different from column c:

$$\begin{aligned}
\partial\widehat{Y}_{rc}/\partial\widehat{N}_{rc} &= 1 - \left(\widehat{N}_{r\cdot} + \widehat{N}_{\cdot c} - \widehat{N}_{\cdot c}\, \widehat{N}_{r\cdot} \,/\, \widehat{N}\right) \,/\, \widehat{N} \\
\partial\widehat{Y}_{rc}/\partial\widehat{N}_{ac} &= -\left(\widehat{N}_{r\cdot} - \widehat{N}_{r\cdot}\, \widehat{N}_{\cdot c} \,/\, \widehat{N}\right) \,/\, \widehat{N} \\
\partial\widehat{Y}_{rc}/\partial\widehat{N}_{rb} &= -\left(\widehat{N}_{\cdot c} - \widehat{N}_{r\cdot}\, \widehat{N}_{\cdot c} \,/\, \widehat{N}\right) \,/\, \widehat{N} \\
\partial\widehat{Y}_{rc}/\partial\widehat{Y}_{ab} &= \widehat{N}_{r\cdot}\, \widehat{N}_{\cdot c} \,/\, \widehat{N}^{\,2}
\end{aligned}$$

Under the null hypothesis of independence, the statistic Q_W approximately follows a chi-square distribution with $(R-1)(C-1)$ degrees of freedom for large samples.

PROC SURVEYFREQ computes the Wald F statistic as

$$F_W = Q_W \,/\, (R-1)(C-1)$$

Under the null hypothesis of independence, F_W approximately follows an F distribution with $(R-1)(C-1)$ numerator degrees of freedom. The denominator degrees of freedom is the degrees of freedom for the variance estimator and depends on the sample design and the variance estimation method. For more information, see the section "Degrees of Freedom" on page 9846. Alternatively, you can use the DF= option in the TABLES statement to specify the denominator degrees of freedom.

For tables larger than 2×2, PROC SURVEYFREQ also computes the adjusted Wald F statistic as

$$F_{\text{Adj_W}} = Q_W\, (s - k + 1) \,/\, ks$$

where $k = (R-1)(C-1)$, and s is the degrees of freedom. (For more information, see the section "Degrees of Freedom" on page 9846.) Alternatively, you can use the DF= option in the TABLES statement to specify the value of s. For 2×2 tables, $k = (R-1)(C-1) = 1$, and therefore the adjusted Wald F statistic equals the (unadjusted) Wald F statistic and has the same numerator and denominator degrees of freedom.

Under the null hypothesis, $F_{\text{Adj_W}}$ approximately follows an F distribution with k numerator degrees of freedom and $(s - k + 1)$ denominator degrees of freedom.

Wald Log-Linear Chi-Square Test

If you specify the WLLCHISQ option in the TABLES statement, PROC SURVEYFREQ computes a Wald test for independence based on the log odds ratios. For more information about Wald tests, see the section "Wald Chi-Square Test" on page 9865.

For a two-way table of R rows and C columns, the Wald log-linear test is based on the $(R-1)(C-1)$-dimensional array of elements $\widehat{Y}_{rc}$,

$$\widehat{Y}_{rc} = \log \widehat{N}_{rc} - \log \widehat{N}_{rC} - \log \widehat{N}_{Rc} + \log \widehat{N}_{RC}$$

where $\widehat{N}_{rc}$ is the estimated total for table cell (r, c). The null hypothesis of independence between the row and column variables can be expressed as $H_0: Y_{rc} = 0$ for all $r = 1, \ldots (R-1)$ and $c = 1, \ldots (C-1)$. This null hypothesis can be stated equivalently in terms of cell proportions.

The generalized Wald log-linear chi-square statistic is computed as

$$Q_L = \widehat{\mathbf{Y}}' \, \widehat{\mathbf{V}}(\widehat{\mathbf{Y}})^{-1} \, \widehat{\mathbf{Y}}$$

where $\widehat{\mathbf{Y}}$ is the $(R-1)(C-1)$-dimensional array of the $\widehat{Y}_{rc}$, and $\widehat{\mathbf{V}}(\widehat{\mathbf{Y}})$ estimates the variance of $\widehat{\mathbf{Y}}$,

$$\widehat{\mathbf{V}}(\widehat{\mathbf{Y}}) = \mathbf{A}\,\mathbf{D}^{-1} \, \widehat{V}(\widehat{\mathbf{N}}) \, \mathbf{D}^{-1} \, \mathbf{A}'$$

where $\widehat{\mathbf{V}}(\widehat{\mathbf{N}})$ is the covariance matrix of the estimates $\widehat{N}_{rc}$, which is computed as described in the section "Covariances of Frequency Estimates" on page 9838. $\mathbf{D}$ is a diagonal matrix with the estimated totals $\widehat{N}_{rc}$ on the diagonal, and $\mathbf{A}$ is the $(R-1)(C-1)$ by $RC \times RC$ linear contrast matrix.

Under the null hypothesis of independence, the statistic Q_L approximately follows a chi-square distribution with $(R-1)(C-1)$ degrees of freedom for large samples.

PROC SURVEYFREQ computes the Wald log-linear F statistic as

$$F_L = Q_L \, / \, (R-1)(C-1)$$

Under the null hypothesis of independence, F_L approximately follows an F distribution with $(R-1)(C-1)$ numerator degrees of freedom. PROC SURVEYFREQ computes the denominator degrees of freedom as described in the section "Degrees of Freedom" on page 9846. Alternatively, you can use the DF= option in the TABLES statement to specify the denominator degrees of freedom.

For tables larger than 2×2, PROC SURVEYFREQ also computes the adjusted Wald log-linear F statistic as

$$F_{\text{Adj_L}} = Q_L \, (s - k + 1) \, / \, ks$$

where $k = (R-1)(C-1)$, and s is the denominator degrees of freedom, which is computed as described in the section "Degrees of Freedom" on page 9846. Alternatively, you can use the DF= option in the TABLES statement to specify the value of s. For 2×2 tables, $k = (R-1)(C-1) = 1$, and therefore the adjusted Wald F statistic equals the (unadjusted) Wald F statistic and has the same numerator and denominator degrees f freedom.

Under the null hypothesis, $F_{\text{Adj_L}}$ approximately follows an F distribution with k numerator degrees of freedom and $(s-k+1)$ denominator degrees of freedom.

Output Data Sets

You can use the Output Delivery System to create a SAS data set from any piece of PROC SURVEYFREQ output. For more information, see the section "ODS Table Names" on page 9877 and Example 118.3.

PROC SURVEYFREQ also provides an output data set that stores the replicate weights for bootstrap, BRR or jackknife variance estimation and an output data set that stores the jackknife coefficients for jackknife variance estimation.

Replicate Weight Output Data Set

If you specify the OUTWEIGHTS= *method-option* for VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE, PROC SURVEYFREQ stores the replicate weights in an output data set. The OUTWEIGHTS= output data set contains all observations from the DATA= input data set that are valid (used in the analysis). A valid observation must have a positive value of the WEIGHT variable. A valid observations must also have nonmissing values of the STRATA and CLUSTER variables unless you specify the MISSING option in the PROC SURVEYFREQ statement. For information about valid observations, see the section "Data Summary Table" on page 9869.

The OUTWEIGHTS= data set contains the following variables:

- all variables in the DATA= input data set
- RepWt_1, RepWt_2, . . ., RepWt_n, which are the replicate weight variables, where n is the total number of replicates in the analysis

Each replicate weight variable contains the replicate weights for the corresponding replicate. Replicate weights equal 0 for those observations not included in the replicate.

After the procedure creates and stores replicate weights for a particular input data set and survey design, you can use them again in subsequent analyses, either in PROC SURVEYFREQ or in another survey procedure. You use a REPWEIGHTS statement to provide replicate weights to the procedure.

Jackknife Coefficient Output Data Set

If you specify the OUTJKCOEFS= *method-option* for VARMETHOD=JACKKNIFE, PROC SURVEYFREQ stores the jackknife coefficients in an output data set. The OUTJKCOEFS= output data set contains one observation for each replicate. The OUTJKCOEFS= data set contains the following variables:

- Replicate, which is the replicate number for the jackknife coefficient
- JKCoefficient, which is the jackknife coefficient
- DonorStratum, which is the stratum of the PSU that was deleted to construct the replicate, if you specify a STRATA statement

After the procedure creates jackknife coefficients for a particular input data set and survey design, you can use the OUTJKCOEFS= *method-option* to store these coefficients and then use them again in subsequent analyses, either in PROC SURVEYFREQ or in another survey procedure. You use the JKCOEFS= option in the REPWEIGHTS statement to provide jackknife coefficients for the procedure.

Displayed Output

Data Summary Table

The "Data Summary" table provides information about the input data set and the sample design. PROC SURVEYFREQ displays this table unless you specify the NOSUMMARY option in the PROC SURVEYFREQ statement.

The "Data Summary" table displays the total number of valid observations. To be considered *valid*, an observation must have a nonmissing, positive sampling weight value if you specify a WEIGHT statement. If you do not specify the MISSING option, a valid observation must also have nonmissing values for all STRATA and CLUSTER variables. The number of valid observations can differ from the number of nonmissing observations for an individual table request, which the procedure displays in the frequency or crosstabulation tables. For more information, see the section "Missing Values" on page 9825.

PROC SURVEYFREQ displays the following information in the "Data Summary" table:

- Number of Strata, if you specify a STRATA statement
- Number of Clusters, if you specify a CLUSTER statement
- Number of Observations, which is the total number of valid observations
- Sum of Weights, which is the sum over all valid observations, if you specify a WEIGHT or REPWEIGHTS statement

Stratum Information Table

If you specify the LIST option in the STRATA statement, PROC SURVEYFREQ displays a "Stratum Information" table. This table provides the following information for each stratum:

- Stratum Index, which is a sequential stratum identification number
- STRATA variables, which list the levels of STRATA variables for the stratum
- Number of Observations, which is the number of valid observations in the stratum
- Population Total for the stratum, if you specify the TOTAL= option
- Sampling Rate for the stratum, if you specify the TOTAL= or RATE= option. If you specify the TOTAL= option, the sampling rate is based on the number of valid observations in the stratum.
- Number of Clusters, which is the number of clusters in the stratum, if you specify a CLUSTER statement

Variance Estimation Table

If you specify VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, VARMETHOD=JACKKNIFE, the NOMCAR option, or the REPWEIGHTS statement, PROC SURVEYFREQ displays a "Variance Estimation" table. If you do not specify any of these options, the procedure creates a "Variance Estimation" table but does not display it. You can store this nondisplayed table in an output data set by using the Output Delivery System (ODS). For more information, see the section "ODS Table Names" on page 9877.

The "Variance Estimation" table provides the following information:

- Method, which is the variance estimation method—Taylor Series, Bootstrap, Balanced Repeated Replication, or Jackknife
- Center Replicates, if you specify the CENTER=REPLICATES *method-option* for a replication variance estimation method
- Replicate Weights input data set name, if you use a REPWEIGHTS statement to provide replicate weights
- Number of Replicates, if you specify VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or the REPWEIGHTS statement
- Bootstrap Seed, if you specify VARMETHOD=BOOTSTRAP
- Hadamard Data Set name, if you specify the HADAMARD= *method-option* for VARMETHOD=BRR
- Fay Coefficient, if you specify the FAY *method-option* for VARMETHOD=BRR
- Missing Levels Included (MISSING), if you specify the MISSING option
- Missing Levels Included (NOMCAR), if you specify the NOMCAR option

Hadamard Matrix

If you specify the PRINTH *method-option* for VARMETHOD=BRR, PROC SURVEYFREQ displays the Hadamard matrix that it uses to construct replicates for BRR variance estimation. If you provide a Hadamard matrix by specifying the HADAMARD= *method-option* for VARMETHOD=BRR but the procedure does not use the entire matrix, the procedure displays only the rows and columns that are actually used to construct replicates.

One-Way Frequency Tables

PROC SURVEYFREQ displays one-way frequency tables for all one-way table requests in the TABLES statements, unless you specify the NOPRINT option in the TABLES statement. A one-way table shows the sample frequency distribution of a single variable, and provides estimates for its population distribution in terms of totals and proportions.

If you request a one-way table without specifying options, PROC SURVEYFREQ displays the following information for each level of the variable:

- Frequency count, which is the number of sample observations in the level

- Weighted Frequency, which estimates the population total for the level
- Standard Error of Weighted Frequency
- Percent, which estimates the population proportion for the level
- Standard Error of Percent

The one-way table displays weighted frequencies if your analysis includes a WEIGHT or REPWEIGHTS statement, or if you specify the WTFREQ option in the TABLES statement.

The one-way table also displays the Frequency Missing, which is the number of observations with missing values.

You can suppress the frequency counts by specifying the NOFREQ option in the TABLES statement. Also, the NOWT option suppresses the weighted frequencies and their standard errors. The NOPERCENT option suppresses the percentages and their standard errors. The NOSTD option suppresses the standard errors of percentages and weighted frequencies. The NOTOTAL option suppresses the total row of the one-way table.

PROC SURVEYFREQ optionally displays the following information in a one-way table:

- Expected Weighted Frequency, if you specify the ONEWAYTABLE(EXPECTED) option
- Deviation from Expected Weighted Frequency, if you specify the ONEWAYTABLE(DEVIATION) option
- Pearson Residual, if you specify the ONEWAYTABLE(PEARSONRES) option
- Cell Chi-Square, if you specify the ONEWAYTABLE(CELLCHI2) option
- Variance of Weighted Frequency, if you specify the VARWT option
- Confidence Limits for Weighted Frequency, if you specify the CLWT option
- Coefficient of Variation for Weighted Frequency, if you specify the CVWT option
- Test Percent, if you specify the TESTP= option
- Variance of Percent, if you specify the VAR option
- Confidence Limits for Percent, if you specify the CL option
- Coefficient of Variation for Percent, if you specify the CV option
- Design Effect for Percent, if you specify the DEFF option

Crosstabulation Tables

PROC SURVEYFREQ displays all table requests in the TABLES statements, unless you specify the NOPRINT option in the TABLES statement. For two-way to multiway crosstabulation tables, the values of the last variable in the table request form the table columns. The values of the next-to-last variable form the rows. Each level (or combination of levels) of the other variables forms one layer. PROC SURVEYFREQ produces a separate two-way crosstabulation table for each layer of a multiway table.

For each layer, the crosstabulation table displays the row and column variable names and values (levels). Each two-way table lists levels of the column variable within each level of the row variable.

By default, PROC SURVEYFREQ displays all levels of the column variable within each level of the row variable, including any column variable levels that have frequencies of 0 in the row level. By default for multiway tables, PROC SURVEYFREQ displays all levels of the row variable within each layer of the table, including any row levels that have frequencies of 0 in the layer. You can suppress the display of zero-frequency levels by specifying the NOSPARSE option.

If you request a crosstabulation table without specifying options, the table displays the following information for each combination of variable levels (table cell):

- Frequency, which is the number of sample observations in the table cell
- Weighted Frequency, which estimates the population total for the table cell
- Standard Error of Weighted Frequency
- Percent, which estimates the population proportion for the table cell
- Standard Error of Percent

The two-way table displays weighted frequencies if your analysis includes a WEIGHT or REPWEIGHTS statement, or if you specify the WTFREQ option in the TABLES statement.

The two-way table also displays the Frequency Missing, which is the number of observations with missing values.

You can suppress the frequency counts by specifying the NOFREQ option in the TABLES statement. Also, the NOWT option suppresses the weighted frequencies and their standard errors. The NOPERCENT option suppresses all percentages and their standard errors. The NOCELLPERCENT option suppresses overall cell percentages and their standard errors, but displays any other percentages (and standard errors) that you request, such as row or column percentages. The NOSTD option suppresses the standard errors of percentages and weighted frequencies. The NOTOTAL option suppresses the row totals, column totals, and overall total.

PROC SURVEYFREQ optionally displays the following information in a two-way table:

- Expected Weighted Frequency, if you specify the EXPECTED option
- Deviation from Expected Weighted Frequency, if you specify the DEVIATION option
- Pearson Residual, if you specify the PEARSONRES option
- Cell Chi-Square, if you specify the CELLCHI2 option

- Variance of Weighted Frequency, if you specify the VARWT option
- Confidence Limits for Weighted Frequency, if you specify the CLWT option
- Coefficient of Variation for Weighted Frequency, if you specify the CVWT option
- Variance of Percent, if you specify the VAR option
- Confidence Limits for Percent, if you specify the CL option
- Coefficient of Variation for Percent, if you specify the CV option
- Design Effect for Percent, if you specify the DEFF option
- Row Percent, which estimates the population proportion of the row total, if you specify the ROW option
- Standard Error of Row Percent, if you specify the ROW option
- Variance of Row Percent, if you specify the VAR option and the ROW option
- Confidence Limits for Row Percent, if you specify the CL option and the ROW option
- Coefficient of Variation for Row Percent, if you specify the CV option and the ROW option
- Design Effect for Row Percent, if you specify the ROW(DEFF) option
- Column Percent, which estimates the population proportion of the column total, if you specify the COLUMN option
- Standard Error of Column Percent, if you specify the COLUMN option
- Variance of Column Percent, if you specify the VAR option and the COLUMN option
- Confidence Limits for Column Percent, if you specify the CL option and the COLUMN option
- Coefficient of Variation for Column Percent, if you specify the CV option and the COLUMN option
- Design Effects for Column Percent, if you specify the COLUMN(DEFF) option

Covariance Matrices of Estimates

If you specify the COV option, PROC SURVEYFREQ displays the covariance matrix of the cell total frequency estimates. If you specify the COVP option, PROC SURVEYFREQ displays the covariance matrix of the proportion estimates.

Statistical Tests

If you specify the CHISQ option for the Rao-Scott chi-square test or the LRCHISQ option for the Rao-Scott likelihood ratio chi-square test, PROC SURVEYFREQ displays the following information:

- Pearson Chi-Square, if you specify the CHISQ option
- Likelihood Ratio Chi-Square, if you specify the LRCHISQ option
- Design Correction
- Rao-Scott Chi-Square, by default or if you specify the FIRSTORDER option
- First-Order Chi-Square, if you specify the SECONDORDER option
- Second-Order Chi-Square, if you specify the SECONDORDER option
- DF, which is the degrees of freedom for the chi-square test
- Pr > ChiSq, which is the p-value for the chi-square test
- F Value
- Num DF, which is the numerator degrees of freedom for F
- Den DF, which is the denominator degrees of freedom for F
- Pr > F, which is the p-value for the F test

If you specify the WCHISQ option for the Wald chi-square test or the WLLCHISQ option for the Wald log-linear chi-square test, PROC SURVEYFREQ displays the following information:

- Wald Chi-Square, if you specify the WCHISQ option
- Wald Log-Linear Chi-Square, if you specify the WLLCHISQ option
- F Value
- Num DF, which is the numerator degrees of freedom for F
- Den DF, which is the denominator degrees of freedom for F
- Pr > F, which is the p-value for the F test
- Adjusted F Value, for tables larger than 2×2
- Num DF, which is the numerator degrees of freedom for Adjusted F
- Den DF, which is the denominator degrees of freedom for Adjusted F
- Pr > Adj F, which is the p-value for the Adjusted F test

Sensitivity and Specificity

If you specify the SENSPEC option in the TABLES statement for a 2 × 2 table, PROC SURVEYFREQ displays the "Sensitivity and Specificity" table. This table includes the following information:

- Statistic, which identifies the statistic as Sensitivity, Specificity, Accuracy, Positive Predictive Value, or Negative Predictive Value
- Estimate
- Standard Error
- Confidence Limits

Risks and Risk Difference

If you specify the RISK or RISKDIFF option in the TABLES statement for a 2 × 2 table, PROC SURVEYFREQ displays "Column 1 Risks" and "Column 2 Risks" tables. You can display only column 1 or column 2 risks by specifying the COLUMN= option.

By default, each "Risks" table displays the following information for Row 1, Row 2, Total, and Difference:

- Risk, which identifies the risk as Row 1, Row 2, Total, or Difference
- Estimate
- Standard Error
- Confidence Limits

In the "Column 1 Risks" table, the row 1 risk is the column 1 percentage of row 1. The row 2 risk is the column 1 percentage of row 2, and the total risk is the column 1 percentage of the entire table. The risk difference is the row 1 risk minus the row 2 risk. In the "Column 2 Risks" table, these computations are based on column 2.

If you specify the RISKDIFF ONLY option, the table displays only the risk difference (together with its standard error and confidence limits).

Odds Ratio and Relative Risks

If you specify the OR option in the TABLES statement for a 2 × 2 table, PROC SURVEYFREQ displays the "Odds Ratio" table. This table includes the following information:

- Statistic, which identifies the statistic as the Odds Ratio, Column 1 Relative Risk, or Column 2 Relative Risk
- Estimate
- Confidence Limits

Difference between Discordant Proportions

If you specify the DISCORDDIFF option in the TABLES statement for a 2×2 table, PROC SURVEYFREQ displays the "Discordant Proportion Difference" table. This table includes the following information:

- Estimate
- Standard Error
- Confidence Limits
- DF
- t Value
- Pr > |t|

Kappa Statistics

If you specify the AGREE, KAPPA, or WTKAPPA option in the TABLES statement for a square table, PROC SURVEYFREQ displays the "Kappa Statistics" table. This table includes the following information:

- Statistic, which identifies the statistic as the Simple Kappa, Weighted Kappa, P-B Adjusted Kappa, or Chance-Corrected AC1
- Estimate
- Standard Error
- Confidence Limits

Kappa Details

If you specify the AGREE(DETAILS) or KAPPA(DETAILS) option in the TABLES statement for a square table, PROC SURVEYFREQ displays the "Kappa Details" table. This table includes the following information:

- Observed Agreement
- Chance-Expected Agreement
- Maximum Kappa
- Prevalence Index, if the table is 2×2
- Bias Index, if the table is 2×2
- B_N Measure

Weighted Kappa Details

If you specify the AGREE(DETAILS) or WTKAPPA(DETAILS) option for a square table whose dimension is greater than 2, PROC SURVEYFREQ displays the "Weighted Kappa Details" table. This table includes the following information:

- Observed Agreement
- Chance-Expected Agreement

Kappa Coefficient Weights

If you specify the AGREE(PRINTKWTS) or WTKAPPA(PRINTKWTS) option for a square table whose dimension is greater than 2, PROC SURVEYFREQ displays the "Kappa Weights" table. This table provides the matrix of kappa agreement weights that the procedure uses to compute the weighted kappa coefficient. The matrix contains an agreement weight for each pair of column variable levels.

ODS Table Names

PROC SURVEYFREQ assigns a name to each table that it creates. You can use these names to refer to tables when you use the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System." See Example 118.3 for examples of storing PROC SURVEYFREQ tables as output data sets.

Table 118.9 lists the ODS table names together with their descriptions and the options required to produce the tables.

Table 118.9 ODS Tables Produced by PROC SURVEYFREQ

ODS Table Name	Description	Statement	Option
ChiSq	Chi-square test	TABLES	CHISQ
ChiSq1	Modified chi-square test	TABLES	CHISQ(MODIFIED)
Cov	Covariances of frequencies	TABLES	COV
CovP	Covariances of proportions	TABLES	COVP
CrossTabs	Crosstabulation table	TABLES	n-way table request, $n > 1$
DiscordDiff	Discordant proportion difference	TABLES	DISCORDDIFF (2×2 table)
HadamardMatrix	Hadamard matrix	PROC	VARMETHOD=BRR(PRINTH)
Kappa	Kappa statistics	TABLES	AGREE, KAPPA, or WTKAPPA ($r \times r$ table)
KappaDetails	Kappa details	TABLES	KAPPA(DETAILS) or AGREE(DETAILS) ($r \times r$ table)
KappaWeights	Kappa agreement weights	TABLES	WTKAPPA(PRINTKWTS) ($r \times r$ table, $r > 2$)
LRChiSq	Likelihood ratio test	TABLES	LRCHISQ
LRChiSq1	Modified likelihood ratio test	TABLES	LRCHISQ(MODIFIED)
OddsRatio	Odds ratio and relative risks	TABLES	OR (2×2 table)
OneWay	One-way frequency table	PROC or TABLES	No TABLES statement One-way table request
Risk1	Column 1 risks and difference	TABLES	RISK or RISKDIFF (2×2 table)
Risk2	Column 2 risks and difference	TABLES	RISK or RISKDIFF (2×2 table)
SenSpec	Sensitivity and specificity	TABLES	SENSPEC (2×2 table)
StrataInfo	Stratum information	STRATA	LIST
Summary	Data summary	PROC	Default
TableSummary	Table summary (not displayed)	TABLES	Default
VarianceEstimation	Variance estimation	PROC	VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or NOMCAR
WChiSq	Wald chi-square test	TABLES	WCHISQ (two-way table)
WLLChiSq	Wald log-linear chi-square test	TABLES	WLLCHISQ (two-way table)
WtKappaDetails	Weighted kappa details	TABLES	WTKAPPA(DETAILS) or AGREE(DETAILS) ($r \times r$ table, $r > 2$)

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, you can request specific plots by specifying the PLOTS= option in the TABLES statement. To produce a weighted frequency plot or mosaic plot, you must specify the WTFREQPLOT or MOSAICPLOT *plot-request* in the PLOTS= option, or you must specify the PLOTS=ALL option. By default, PROC SURVEYFREQ produces all other plots that are associated with the analyses that you request in the TABLES statement. You can suppress default plots and request specific plots by using the PLOTS(ONLY)= option. For more information, see the description of the PLOTS= option.

PROC SURVEYFREQ assigns a name to each graph that it creates by using ODS Graphics. You can use these names to refer to the graphs. Table 118.10 lists the names of the graphs that PROC SURVEYFREQ generates together with their descriptions, their PLOTS= options (*plot-requests*), and the TABLES statement options that are required to produce the graphs.

Table 118.10 ODS Graphs Produced by PROC SURVEYFREQ

ODS Graph Name	Description	PLOTS= Option	TABLES Statement Option
KappaPlot	Kappa plot	KAPPAPLOT	AGREE or KAPPA ($h \times r \times r$ table)
MosaicPlot	Mosaic plot	MOSAICPLOT	Two-way or multiway table request
ORPlot	Odds ratio plot	ODDSRATIOPLOT	OR ($h \times 2 \times 2$ table)
RelRiskPlot	Relative risk plot	RELRISKPLOT	OR ($h \times 2 \times 2$ table)
RiskDiffPlot	Risk difference plot	RISKDIFFPLOT	RISK or RISKDIFF ($h \times 2 \times 2$ table)
WtFreqPlot	Weighted frequency plot	WTFREQPLOT	Any table request
WtKappaPlot	Weighted kappa plot	WTKAPPAPLOT	AGREE or WTKAPPA ($h \times r \times r$ table, $r > 2$)

Examples: SURVEYFREQ Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-stat.

Example 118.1: Two-Way Tables

(View the complete code for this example (surfex1.sas) in the example repository.)

This example uses the SIS_Survey data set from the section "Getting Started: SURVEYFREQ Procedure" on page 9769. The data set contains results from a customer satisfaction survey for a student information system (SIS).

The following PROC SURVEYFREQ statements request a two-way table for Department by Response and customize the crosstabulation table display:

```
title 'Student Information System Survey';
proc surveyfreq data=SIS_Survey;
   tables  Department * Response / cv deff nowt nostd nototal;
   strata  State NewUser / list;
   cluster School;
   weight  SamplingWeight;
run;
```

The TABLES statement requests a two-way table of Department by Response. The CV option requests coefficients of variation for the percentage estimates. The DEFF option requests design effects for the percentage estimates. The NOWT option suppresses display of the weighted frequencies, and the NOSTD option suppresses display of standard errors for the estimates. The NOTOTAL option suppresses the row totals, column totals, and overall totals.

The STRATA, CLUSTER, and WEIGHT statements provide sample design information for the procedure, so that the analysis is done according to the sample design used for the survey. The STRATA statement names the variables State and NewUser, which identify the first-stage strata. The LIST option in the STRATA statement requests a "Stratum Information" table. The CLUSTER statement names the variable School, which identifies the clusters (primary sampling units). The WEIGHT statement names the sampling weight variable.

Output 118.1.1 displays the "Data Summary" and "Stratum Information" tables produced by PROC SURVEYFREQ. The "Stratum Information" table lists the six strata in the survey and shows the number of observations and the number of clusters (schools) in each stratum.

Output 118.1.1 Data Summary and Stratum Information

Student Information System Survey

The SURVEYFREQ Procedure

Data Summary	
Number of Strata	6
Number of Clusters	370
Number of Observations	1850
Sum of Weights	38899.6482

Stratum Information

Stratum Index	State	NewUser	Number of Obs	Number of Clusters
1	GA	Renewal Customer	315	63
2	GA	New Customer	355	71
3	NC	Renewal Customer	280	56
4	NC	New Customer	420	84
5	SC	Renewal Customer	210	42
6	SC	New Customer	270	54

Output 118.1.2 displays the two-way table of Department by Response. According to the TABLES statement options that are specified, this two-way table includes coefficients of variation and design effects for the percentage estimates, and it does not show the weighted frequencies or the standard errors of the estimates. It also does not show the row, column, and overall totals.

Output 118.1.2 Two-Way Table of Department by Response

Table of Department by Response

Department	Response	Frequency	Percent	CV for Percent	Design Effect
Faculty	Very Unsatisfied	209	13.4987	0.0865	2.1586
	Unsatisfied	203	13.0710	0.0868	2.0962
	Neutral	346	22.4127	0.0629	2.1157
	Satisfied	254	16.2006	0.0806	2.3232
	Very Satisfied	98	6.2467	0.1362	2.2842
Admin/Guidance	Very Unsatisfied	95	3.6690	0.1277	1.1477
	Unsatisfied	123	4.6854	0.1060	1.0211
	Neutral	235	9.1838	0.0700	0.9166
	Satisfied	201	7.7305	0.0756	0.8848
	Very Satisfied	86	3.3016	0.1252	0.9892

The following PROC SURVEYFREQ statements request a two-way table of Department by Response that includes row percentages, and also a Wald chi-square test of association between the two table variables:

```
title 'Student Information System Survey';
proc surveyfreq data=SIS_Survey nosummary;
   tables  Department * Response / row nowt wchisq;
   strata  State NewUser;
   cluster School;
```

```
    weight  SamplingWeight;
  run;
```

Output 118.1.3 displays the two-way table. The row percentages show the distribution of Response for Department = 'Faculty' and for Department = 'Admin/Guidance'. This is equivalent to a domain (subpopulation) analysis of Response, where the domains are Department = 'Faculty' and Department = 'Admin/Guidance'.

Output 118.1.4 displays the Wald chi-square test of association between Department and Response. The Wald chi-square is 11.44, and the corresponding adjusted F value is 2.84 with a p-value of 0.0243. This indicates a significant association between department (faculty or administration/guidance) and satisfaction with the student information system.

Output 118.1.3 Table of Department by Response with Row Percentages

Student Information System Survey

The SURVEYFREQ Procedure

Table of Department by Response

Department	Response	Frequency	Percent	Std Err of Percent	Row Percent	Std Err of Row Percent
Faculty	Very Unsatisfied	209	13.4987	1.1675	18.8979	1.6326
	Unsatisfied	203	13.0710	1.1350	18.2992	1.5897
	Neutral	346	22.4127	1.4106	31.3773	1.9705
	Satisfied	254	16.2006	1.3061	22.6805	1.8287
	Very Satisfied	98	6.2467	0.8506	8.7452	1.1918
	Total	1110	71.4297	0.1468	100.0000	
Admin/Guidance	Very Unsatisfied	95	3.6690	0.4684	12.8419	1.6374
	Unsatisfied	123	4.6854	0.4966	16.3995	1.7446
	Neutral	235	9.1838	0.6430	32.1447	2.2300
	Satisfied	201	7.7305	0.5842	27.0579	2.0406
	Very Satisfied	86	3.3016	0.4133	11.5560	1.4466
	Total	740	28.5703	0.1468	100.0000	
Total	Very Unsatisfied	304	17.1676	1.2872		
	Unsatisfied	326	17.7564	1.2712		
	Neutral	581	31.5965	1.5795		
	Satisfied	455	23.9311	1.4761		
	Very Satisfied	184	9.5483	0.9523		
	Total	1850	100.0000			

Output 118.1.4 Wald Chi-Square Test

Wald Chi-Square Test	
Chi-Square	11.4454
F Value	2.8613
Num DF	4
Den DF	364
Pr > F	0.0234
Adj F Value	2.8378
Num DF	4
Den DF	361
Pr > Adj F	0.0243
Sample Size = 1850	

Example 118.2: Multiway Tables (Domain Analysis)

(View the complete code for this example (surfex2.sas) in the example repository.)

Continuing to use the SIS_Survey data set from the section "Getting Started: SURVEYFREQ Procedure" on page 9769, this example shows how to produce multiway tables. The following PROC SURVEYFREQ statements request a table of Department by SchoolType by Response for the student information system survey:

```
title 'Student Information System Survey';
proc surveyfreq data=SIS_Survey;
   tables  Department * SchoolType * Response
           SchoolType * Response;
   strata  State NewUser;
   cluster School;
   weight  SamplingWeight;
run;
```

The TABLES statement requests a multiway table with SchoolType as the row variable, Response as the column variable, and Department as the layer variable. This request produces a separate two-way table of SchoolType by Response for each level of the variable Department. The TABLES statement also requests a two-way table of SchoolType by Response, which totals the multiway table over both levels of Department. As in the previous examples, the STRATA, CLUSTER, and WEIGHT statements provide sample design information, so that the analysis will be done according to the design used for this survey.

Output 118.2.1 displays the multiway table produced by PROC SURVEYFREQ, which includes a table of SchoolType by Response for Department = 'Faculty' and for Department = 'Admin/Guidance'. This is equivalent to a domain (subpopulation) analysis of SchoolType by Response, where the domains are Department = 'Faculty' and Department = 'Admin/Guidance'.

Output 118.2.1 Multiway Table of Department by SchoolType by Response

Student Information System Survey

The SURVEYFREQ Procedure

Table of SchoolType by Response
Controlling for Department=Faculty

SchoolType	Response	Frequency	Weighted Frequency	Std Err of Wgt Freq	Percent	Std Err of Percent
Middle School	Very Unsatisfied	74	1846	301.22637	6.6443	1.0838
	Unsatisfied	78	1929	283.11476	6.9428	1.0201
	Neutral	130	3289	407.80855	11.8369	1.4652
	Satisfied	113	2795	368.85087	10.0597	1.3288
	Very Satisfied	55	1378	261.63311	4.9578	0.9411
	Total	450	11237	714.97120	40.4415	2.5713
High School	Very Unsatisfied	135	3405	389.42313	12.2536	1.3987
	Unsatisfied	125	3155	384.56734	11.3563	1.3809
	Neutral	216	5429	489.37826	19.5404	1.7564
	Satisfied	141	3507	417.54773	12.6208	1.5040
	Very Satisfied	43	1052	221.59367	3.7874	0.7984
	Total	660	16549	719.61536	59.5585	2.5713
Total	Very Unsatisfied	209	5251	454.82598	18.8979	1.6326
	Unsatisfied	203	5085	442.39032	18.2992	1.5897
	Neutral	346	8718	550.81735	31.3773	1.9705
	Satisfied	254	6302	507.01711	22.6805	1.8287
	Very Satisfied	98	2430	330.97602	8.7452	1.1918
	Total	1110	27786	119.25529	100.0000	

Output 118.2.1 *continued*

Table of SchoolType by Response
Controlling for Department=Admin/Guidance

SchoolType	Response	Frequency	Weighted Frequency	Std Err of Wgt Freq	Percent	Std Err of Percent
Middle School	Very Unsatisfied	42	649.43427	133.06194	5.8435	1.1947
	Unsatisfied	31	460.35557	100.80158	4.1422	0.9076
	Neutral	104	1568	186.99946	14.1042	1.6804
	Satisfied	84	1269	165.71127	11.4142	1.4896
	Very Satisfied	39	574.93878	110.37243	5.1732	0.9942
	Total	300	4521	287.86832	40.6774	2.5801
High School	Very Unsatisfied	53	777.77725	136.41869	6.9983	1.2285
	Unsatisfied	92	1362	175.40862	12.2573	1.5806
	Neutral	131	2005	212.34804	18.0404	1.8990
	Satisfied	117	1739	190.07798	15.6437	1.7118
	Very Satisfied	47	709.37033	126.54394	6.3828	1.1371
	Total	440	6593	288.92483	59.3226	2.5801
Total	Very Unsatisfied	95	1427	182.28132	12.8419	1.6374
	Unsatisfied	123	1823	193.43045	16.3995	1.7446
	Neutral	235	3572	250.22739	32.1447	2.2300
	Satisfied	201	3007	226.82311	27.0579	2.0406
	Very Satisfied	86	1284	160.83434	11.5560	1.4466
	Total	740	11114	60.78850	100.0000	

Example 118.3: Output Data Sets

(View the complete code for this example (surfex3.sas) in the example repository.)

PROC SURVEYFREQ uses the Output Delivery System (ODS) to create output data sets. This is a departure from older SAS procedures that provide OUTPUT statements for similar functionality. By using ODS, you can create a SAS data set from any piece of PROC SURVEYFREQ output. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

When selecting tables for ODS output data sets, you refer to tables by their ODS table names. Each table created by PROC SURVEYFREQ is assigned a name. See the section "ODS Table Names" on page 9877 for a list of the table names provided by PROC SURVEYFREQ.

To save the one-way table of Response from Figure 118.3 in an output data set, use an ODS OUTPUT statement as follows:

```
proc surveyfreq data=SIS_Survey;
   tables  Response / cl nowt;
   ods output OneWay=ResponseTable;
   strata  State NewUser;
   cluster School;
   weight  SamplingWeight;
run;
```

Output 118.3.1 displays the output data set ResponseTable, which contains the one-way table of Response. This data set has six observations, and each of these observations corresponds to a row of the one-way table. The first five observations correspond to the five levels of Response, as they are ordered in the one-way table display, and the last observation corresponds to the overall total, which is the last row of the one-way table. The data set ResponseTable includes a variable corresponding to each column of the one-way table. For example, the variable Percent contains the percentage estimates, and the variables LowerCL and UpperCL contain the lower and upper confidence limits for the percentage estimates.

Output 118.3.1 ResponseTable Output Data Set

Obs	Table	Response	Frequency	Percent	StdErr	LowerCL	UpperCL
1	Table Response	Very Unsatisfied	304	17.1676	1.2872	14.6364	19.6989
2	Table Response	Unsatisfied	326	17.7564	1.2712	15.2566	20.2562
3	Table Response	Neutral	581	31.5965	1.5795	28.4904	34.7026
4	Table Response	Satisfied	455	23.9311	1.4761	21.0285	26.8338
5	Table Response	Very Satisfied	184	9.5483	0.9523	7.6756	11.4210
6	Table Response	.	1850	100.0000	_	_	_

PROC SURVEYFREQ also creates a table summary that is not displayed. Some of the information in this table is similar to the information in the "Data Summary" table, but the "Table Summary" table describes the data that are used to analyze the specified table, whereas the "Data Summary" table describes the entire input data set. Because of missing values, for example, the number of observations (or strata or clusters) used to analyze a particular table can differ from the number of observations (or strata or clusters) in the input data set. For more information, see the section "Missing Values" on page 9825. If you request confidence limits, the "Table Summary" table also contains the degrees of freedom and the *t*-value used to compute the confidence limits.

The following statements store the nondisplayed "Table Summary" table in the output data set ResponseSummary:

```
proc surveyfreq data=SIS_Survey;
   tables  Response / cl nowt;
   ods output TableSummary=ResponseSummary;
   strata  State NewUser;
   cluster School;
   weight  SamplingWeight;
run;
```

Output 118.3.2 displays the output data set ResponseSummary.

Output 118.3.2 ResponseSummary Output Data Set

Obs	Table	Number of Observations	Number of Strata	Number of Clusters	Degrees of Freedom	t Percentile
1	Table Response	1850	6	370	364	1.966503

References

Agresti, A. (2007). *An Introduction to Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Agresti, A. (2013). *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Agresti, A., and Coull, B. A. (1998). "Approximate Is Better Than 'Exact' for Interval Estimation of Binomial Proportions." *American Statistician* 52:119–126.

Bangdiwala, S. I. (1988). *The Agreement Chart*. Technical report, Department of Biostatistics, University of North Carolina at Chapel Hill.

Bangdiwala, S. I., Haedo, A. S., Natal, M. L., and Villaveces, A. (2008). "The Agreement Chart as an Alternative to the Receiver-Operating Characteristic Curve for Diagnostic Tests." *Journal of Clinical Epidemiology* 61:866–874.

Beaumont, J.-F., and Patak, Z. (2012). "On the Generalized Bootstrap for Sample Surveys with Special Attention to Poisson Sampling." *International Statistical Review* 80:127–148.

Bedrick, E. J. (1983). "Adjusted Chi-Squared Tests for Cross-Classified Tables of Survey Data." *Biometrika* 70:591–596.

Berger, J. O. (1985). *Statistical Decision Theory and Bayesian Analysis*. 2nd ed. New York: Springer-Verlag.

Blood, E., and Spratt, K. F. (2007). "Disagreement on Agreement: Two Alternative Agreement Coefficients." In *Proceedings of the SAS Global Forum 2007 Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/forum2007/186-2007.pdf.

Brick, J. M., and Kalton, G. (1996). "Handling Missing Data in Survey Research." *Statistical Methods in Medical Research* 5:215–238.

Brown, L. D., Cai, T. T., and DasGupta, A. (2001). "Interval Estimation for a Binomial Proportion." *Statistical Science* 16:101–133.

Byrt, T., Bishop, J., and Carlin, J. B. (1993). "Bias, Prevalence, and Kappa." *Journal of Clinical Epidemiology* 46:423–429.

Carlin, B. P., and Louis, T. A. (2009). *Bayesian Methods for Data Analysis*. 3rd ed. Boca Raton, FL: Chapman & Hall/CRC.

Cicchetti, D. V., and Allison, T. (1971). "A New Procedure for Assessing Reliability of Scoring EEG Sleep Recordings." *American Journal of EEG Technology* 11:101–109.

Clopper, C. J., and Pearson, E. S. (1934). "The Use of Confidence or Fiducial Limits Illustrated in the Case of the Binomial." *Biometrika* 26:404–413.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Cohen, J. (1960). "A Coefficient of Agreement for Nominal Scales." *Educational and Psychological Measurement* 20:37–46.

Curtin, L. R., Kruszon-Moran, D., Carroll, M., and Li, X. (2006). "Estimation and Analytic Issues for Rare Events in NHANES." In *Proceedings of the Survey Research Methods Section*, 2893–2903. Alexandria, VA: American Statistical Association.

Dean, N., and Pagano, M. (2015). "Evaluating Confidence Interval Methods for Binomial Proportions in Clustered Surveys." *Journal of Survey Statistics and Methodology* 3:484–503.

Dippo, C. S., Fay, R. E., and Morganstein, D. H. (1984). "Computing Variances from Complex Samples with Replicate Weights." In *Proceedings of the Survey Research Methods Section*, 489–494. Alexandria, VA: American Statistical Association.

Fay, R. E. (1989). "Theory and Application of Replicate Weighting for Variance Calculations." In *Proceedings of the Survey Research Methods Section*, 212–217. Alexandria, VA: American Statistical Association.

Fellegi, I. P. (1980). "Approximate Tests of Independence and Goodness of Fit Based on Stratified Multistage Samples." *Journal of the American Statistical Association* 75:261–268.

Fienberg, S. E. (1980). *The Analysis of Cross-Classified Categorical Data*. 2nd ed. Cambridge, MA: MIT Press.

Fleiss, J. L. (1981). *Statistical Methods for Rates and Proportions*. 2nd ed. New York: John Wiley & Sons.

Fleiss, J. L., and Cohen, J. (1973). "The Equivalence of Weighted Kappa and the Intraclass Correlation Coefficient as Measures of Reliability." *Educational and Psychological Measurement* 33:613–619.

Fleiss, J. L., Cohen, J., and Everitt, B. S. (1969). "Large-Sample Standard Errors of Kappa and Weighted Kappa." *Psychological Bulletin* 72:323–327.

Fleiss, J. L., Levin, B., and Paik, M. C. (2003). *Statistical Methods for Rates and Proportions*. 3rd ed. Hoboken, NJ: John Wiley & Sons.

Franco, C., Little, R. J. A., Louis, T. A., and Slud, E. V. (2019). "Comparative Study of Confidence Intervals for Proportions in Complex Sample Surveys." *Journal of Survey Statistics and Methodology* 7:334–364.

Friendly, M. (2000). *Visualizing Categorical Data*. Cary, NC: SAS Institute Inc.

Fuller, W. A. (1975). "Regression Analysis for Sample Survey." *Sankhyā, Series C* 37:117–132.

Fuller, W. A. (2009). *Sampling Statistics*. Hoboken, NJ: John Wiley & Sons.

Fuller, W. A., Kennedy, W. J., Schnell, D., Sullivan, G., and Park, H. J. (1989). *PC CARP*. Ames: Iowa State University Statistical Laboratory.

Gwet, K. L. (2008). "Computing Inter-rater Reliability and Its Variance in the Presence of High Agreement." *British Journal of Mathematical and Statistical Psychology* 61:29–48.

Hansen, M. H., Hurwitz, W. N., and Madow, W. G. (1953). *Sample Survey Methods and Theory.* 2 vols. New York: John Wiley & Sons.

Hidiroglou, M. A., Fuller, W. A., and Hickman, R. D. (1980). *SUPER CARP*. Ames: Iowa State University Statistical Laboratory.

Holley, J. W., and Guilford, J. P. (1964). "A Note on the G Index of Agreement." *Educational and Psychological Measurement* 24:749–753.

Judkins, D. R. (1990). "Fay's Method for Variance Estimation." *Journal of Official Statistics* 6:223–239.

Kalton, G. (1983). *Introduction to Survey Sampling.* Vol. 07-035 of University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Kalton, G., and Kasprzyk, D. (1986). "The Treatment of Missing Survey Data." *Survey Methodology* 12:1–16.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Koch, G. G., Freeman, D. H., and Freeman, J. L. (1975). "Strategies in the Multivariate Analysis of Data from Complex Surveys." *International Statistical Review* 43:59–78.

Koch, G. G., Landis, J. R., Freeman, J. L., Freeman, D. H., and Lehnen, R. G. (1977). "A General Methodology for the Analysis of Experiments with Repeated Measurement of Categorical Data." *Biometrics* 33:133–158.

Korn, E. L., and Graubard, B. I. (1990). "Simultaneous Testing with Complex Survey Data: Use of Bonferroni t-Statistics." *American Statistician* 44:270–276.

Korn, E. L., and Graubard, B. I. (1998). "Confidence Intervals for Proportions with Small Expected Number of Positive Counts Estimated from Survey Data." *Survey Methodology* 24:193–201.

Korn, E. L., and Graubard, B. I. (1999). *Analysis of Health Surveys*. New York: John Wiley & Sons.

Lee, E. S., Forthofer, R. N., and Lorimor, R. J. (1989). *Analyzing Complex Survey Data.* Vol. 07-071 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Leemis, L. M., and Trivedi, K. S. (1996). "A Comparison of Approximate Interval Estimators for the Bernoulli Parameter." *American Statistician* 50:63–68.

Levy, P. S., and Lemeshow, S. (1999). *Sampling of Populations: Methods and Applications*. 3rd ed. New York: John Wiley & Sons.

Lohr, S. L. (2022). *Sampling: Design and Analysis*. 3rd ed. Boca Raton, FL: CRC Press.

Mashreghi, Z., Haziza, D., and Léger, C. (2016). "A Survey of Bootstrap Methods in Finite Population Sampling." *Statistics Surveys* 10:1–52.

McCarthy, P. J., and Snowden, C. B. (1985). *The Bootstrap and Finite Population Sampling.* Vital and Health Statistics, Series 2, no. 95. DHHS Publication Number 85–1369. Washington, DC: US Government Printing Office. https://www.cdc.gov/nchs/data/series/sr_02/sr02_095.pdf.

McNemar, Q. (1947). "Note on the Sampling Error of the Difference between Correlated Proportions or Percentages." *Psychometrika* 12:153–157.

Nathan, G. (1975). "Tests for Independence in Contingency Tables from Stratified Samples." *Sankhyā, Series C* 37:77–87.

Newcombe, R. G. (1998). "Two-Sided Confidence Intervals for the Single Proportion: Comparison of Seven Methods." *Statistics in Medicine* 17:857–872.

Rao, J. N. K., and Scott, A. J. (1979). "Chi-Squared Tests for Analysis of Categorical Data from Complex Surveys." In *Proceedings of the Survey Research Methods Section*, 58–66. Washington, DC: American Statistical Association.

Rao, J. N. K., and Scott, A. J. (1981). "The Analysis of Categorical Data from Complex Surveys: Chi-Squared Tests for Goodness of Fit and Independence in Two-Way Tables." *Journal of the American Statistical Association* 76:221–230.

Rao, J. N. K., and Scott, A. J. (1984). "On Chi-Squared Tests for Multiway Contingency Tables with Cell Properties Estimated from Survey Data." *Annals of Statistics* 12:46–60.

Rao, J. N. K., and Scott, A. J. (1987). "On Simple Adjustments to Chi-Square Tests with Survey Data." *Annals of Statistics* 15:385–397.

Rao, J. N. K., and Shao, J. (1996). "On Balanced Half-Sample Variance Estimation in Stratified Random Sampling." *Journal of the American Statistical Association* 91:343–348.

Rao, J. N. K., and Shao, J. (1999). "Modified Balanced Repeated Replication for Complex Survey Data." *Biometrika* 86:403–415.

Rao, J. N. K., and Thomas, D. R. (1989). "Chi-Squared Tests for Contingency Tables." In *Analysis of Complex Surveys*, edited by C. J. Skinner, D. Holt, and T. M. F. Smith, 89–114. New York: John Wiley & Sons.

Rao, J. N. K., and Wu, C. F. J. (1988). "Resampling Inference with Complex Survey Data." *Journal of the American Statistical Association* 83:231–241.

Rao, J. N. K., Wu, C. F. J., and Yue, K. (1992). "Some Recent Work on Resampling Methods for Complex Surveys." *Survey Methodology* 18:209–217.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Satterthwaite, F. E. (1946). "An Approximate Distribution of Estimates of Variance Components." *Biometrics Bulletin* 2:110–114.

Shao, J., and Tu, D. (1995). *The Jackknife and Bootstrap*. New York: Springer-Verlag.

Sim, J., and Wright, C. C. (2005). "The Kappa Statistic in Reliability Studies: Use, Interpretation, and Sample Size Requirements." *Physical Therapy* 85:257–268.

Sitter, R. R. (1992a). "Comparing Three Bootstrap Methods for Survey Data." *Canadian Journal of Statistics* 20:135–154.

Sitter, R. R. (1992b). "A Resampling Procedure for Complex Survey Data." *Journal of the American Statistical Association* 87:755–765.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2000). *Categorical Data Analysis Using the SAS System*. 2nd ed. Cary, NC: SAS Institute Inc.

Sukasih, A., and Jang, D. (2005). "An Application of Confidence Interval Methods for Small Proportions in the Health Care Survey of DoD Beneficiaries." In *Proceedings of the Survey Research Methods Section*, 3608–3612. Alexandria, VA: American Statistical Association.

Thomas, D. R., and Rao, J. N. K. (1984). "A Monte Carlo Study of Exact Levels of Goodness-of-Fit Statistics under Cluster Sampling." In *Proceedings of the Survey Research Methods Section*, 207–211. Alexandria, VA: American Statistical Association.

Thomas, D. R., and Rao, J. N. K. (1985). "On the Power of Some Goodness-of-Fit Tests under Cluster Sampling." In *Proceedings of the Survey Research Methods Section*, 291–296. Alexandria, VA: American Statistical Association.

Thomas, D. R., and Rao, J. N. K. (1987). "Small-Sample Comparisons of Level and Power for Simple Goodness-of-Fit Statistics under Cluster Sampling." *Journal of the American Statistical Association* 82:630–636.

Thomas, D. R., Singh, A. C., and Roberts, G. R. (1996). "Tests of Independence on Two-Way Tables under Cluster Sampling: An Evaluation." *International Statistical Review* 64:295–311.

Wald, A. (1943). "Tests of Statistical Hypotheses Concerning Several Parameters When the Number of Observations Is Large." *Transactions of the American Mathematical Society* 54:426–482.

Wilson, E. B. (1927). "Probable Inference, the Law of Succession, and Statistical Inference." *Journal of the American Statistical Association* 22:209–212.

Wolter, K. M. (1985). *Introduction to Variance Estimation*. New York: Springer-Verlag.

Wolter, K. M. (2007). *Introduction to Variance Estimation*. 2nd ed. New York: Springer.

Woodruff, R. S. (1971). "A Simple Method for Approximating the Variance of a Complicated Estimate." *Journal of the American Statistical Association* 66:411–414.

Xie, Q. (2013). "Agree or Disagree? A Demonstration of an Alternative Statistic to Cohen's Kappa for Measuring the Extent and Reliability of Agreement between Observers." Paper presented at FCSM Research Conference, Nov. 7–9, Washington, DC.

Chapter 119
The SURVEYIMPUTE Procedure

Contents

Overview: SURVEYIMPUTE Procedure **9894**
Getting Started: SURVEYIMPUTE Procedure **9895**
Syntax: SURVEYIMPUTE Procedure **9901**
- PROC SURVEYIMPUTE Statement 9902
- BY Statement 9911
- CELLS Statement 9912
- CLASS Statement 9912
- CLUSTER Statement 9913
- ID Statement 9913
- IMPJOINT Statement 9914
- OUTPUT Statement 9914
- REPWEIGHTS Statement 9916
- STRATA Statement 9917
- VAR Statement 9917
- WEIGHT Statement 9918

Details: SURVEYIMPUTE Procedure **9918**
- Definitions 9918
- Specifying the Sample Design 9919
- Missing Values 9921
- Missing Data Patterns 9923
- Random Number Generation 9923
- Fully Efficient Fractional Imputation 9924
- Two-Stage Fully Efficient Fractional Imputation 9928
- Fractional Hot-Deck Imputation 9940
- Hot-Deck Imputation 9952
- Replication Variance Estimation 9954
- Output Data Sets 9961
- Displayed Output 9961
- ODS Table Names 9964

Examples: SURVEYIMPUTE Procedure **9965**
- Example 119.1: Approximate Bayesian Bootstrap Imputation 9965
- Example 119.2: Fully Efficient Fractional Imputation 9970
- Example 119.3: Fully Efficient Fractional Imputation, Fay's Balanced Repeated Replication, and Domain Analysis 9974
- Example 119.4: Fractional Hot-Deck Imputation for Mixed Variables 9982

References **9990**

Overview: SURVEYIMPUTE Procedure

The SURVEYIMPUTE procedure imputes missing values of an item in a data set by replacing them with observed values from the same item. The principles by which the imputation is performed are particularly useful for survey data. PROC SURVEYIMPUTE also computes replicate weights (such as jackknife weights) that account for the imputation and that can be used for replication-based variance estimation for complex surveys. The procedure implements a fractional hot-deck imputation technique (Kim and Fuller 2004; Fuller 2009; Kim and Shao 2014) in addition to some traditional hot-deck imputation techniques (Andridge and Little 2010).

Nonresponse is a common problem in almost all surveys of human populations. Estimators that are based on survey data that include nonresponse can suffer from nonresponse bias if the nonrespondents are different from the respondents. Estimators that use complete cases (only the observed units) might also be less precise. Imputation techniques are important tools for reducing nonresponse bias and producing efficient estimators.

The main objectives of any imputation technique are to eliminate the nonresponse bias and to provide an imputed data set that results in consistent analyses conducted with the imputed data. In addition, a variance estimator must be available that accounts for both the sampling variance and the imputation variance. Imputation techniques use implicit or explicit models. Some model-based imputation techniques include multiple imputation, mean imputation, and regression imputation. For more information about multiple imputation in SAS/STAT, see Chapter 82, "The MI Procedure," and Chapter 83, "The MIANALYZE Procedure."

Imputation techniques that do not use explicit models include hot-deck imputation, cold-deck imputation, and fractional imputation. PROC SURVEYIMPUTE implements imputation techniques that do not use explicit models. It also produces replicate weights that can be used with any survey analysis procedure in SAS/STAT to estimate both the sampling variability and the imputation variability.

Hot-deck imputation is the most commonly used imputation technique for survey data. A donor is selected for a recipient unit, and the observed values of the donor are imputed for the missing items of the recipient. Although the imputation method is straightforward, the variance estimator that accounts for imputation variance might not be simple and is often ignored in practice. PROC SURVEYIMPUTE does not create imputation-adjusted replicate weights for hot-deck imputation.

Fractional hot-deck imputation (Kalton and Kish 1984; Fay 1996; Kim and Fuller 2004; Fuller and Kim 2005), also known as fractional imputation (FI), is a variation of hot-deck imputation in which one missing item for a recipient is imputed from multiple donors. Each donor donates a fraction of the original weight of the recipient such that the sum of the fractional weights from all the donors is equal to the original weight of the recipient. For fully efficient fractional imputation (FEFI), all observed values in an imputation cell are used as donors for a recipient unit in that cell (Kim and Fuller 2004).

The SURVEYIMPUTE procedure implements single and multiple hot-deck imputation, FEFI and FHDI. Available donor selection techniques include simple random selection with or without replacement, probability proportional to weights selection (Rao and Shao 1992), and approximate Bayesian bootstrap selection (Rubin and Schenker 1986).

The remaining sections of this chapter are organized as follows:

- "Getting Started: SURVEYIMPUTE Procedure" on page 9895 introduces PROC SURVEYIMPUTE with an example.

- "Syntax: SURVEYIMPUTE Procedure" on page 9901 describes the syntax of the procedure.
- "Details: SURVEYIMPUTE Procedure" on page 9918 summarizes the imputation techniques that PROC SURVEYIMPUTE uses.
- "Examples: SURVEYIMPUTE Procedure" on page 9965 includes some additional examples of useful applications.

Getting Started: SURVEYIMPUTE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/svimpgs.sas.)

This example shows how you can use PROC SURVEYIMPUTE to impute missing values and compute imputation-adjusted statistics for sample survey data. The example uses simulated data from a customer satisfaction survey for a student information system (SIS), which is a software product that provides modules for student registration, class scheduling, attendance, grade reporting, and other functions.

The software company conducted a survey of school personnel who use the SIS. A probability sample of SIS users was selected from the study population, which included SIS users at middle schools and high schools in three states, Georgia, South Carolina, and North Carolina. The sample design for this survey was a two-stage stratified design. A first-stage sample of schools was selected from the list of schools in the three states that use the SIS. The list of schools, which are the primary sampling units (PSU), was stratified by state and by customer status (whether the school was a new user or a renewal user of the system). Within the strata, schools were selected with probability proportional to size and with replacement, where the size measure was school enrollment. From each sample school, five staff members were randomly selected with replacement as the second-stage units to complete the SIS satisfaction questionnaire. These staff members include both teachers and administrators.

The SAS data set SIS_Survey_Sub contains the survey results and the sample design information that is needed to analyze the data. The data set contains the following items:

- State: state where the school is located
- NewUser: 1 if the school is a new user of SIS or 0 if not
- School: school identification (PSU)
- SamplingWeight: sampling weight
- Department: 0 for teachers and 1 for administrators
- Response: coded from 1 to 5, where 1 represents "Very Unsatisfied" and 5 represents "Very Satisfied"

```
data SIS_Survey_Sub;
   input State $ NewUser School SamplingWeight Department Response @@;
   datalines;
GA 1  1 25 1 1  GA 1  1 25 1 2  GA 1  1 25 1 2  GA 1  1 15 . .  GA 1  1 15 0 3
GA 1  2 25 . 3  GA 1  2 25 1 1  GA 1  2 25 . 1  GA 1  2 15 0 .  GA 1  2 15 0 1
```

```
GA 1  3 25 1 .  GA 1  3 25 1 4  GA 1  3 25 1 4  GA 1  3 15 . 5  GA 1  3 15 0 .
GA 1  4 25 0 3  GA 1  4 25 0 4  GA 1  4 25 0 .  GA 1  4 15 1 3  GA 1  4 15 1 5
GA 1  5 25 0 3  GA 1  5 25 0 2  GA 1  5 25 0 2  GA 1  5 15 1 4  GA 1  5 15 1 .
GA 1  6 25 0 4  GA 1  6 25 0 .  GA 1  6 25 0 3  GA 1  6 15 1 .  GA 1  6 15 1 4
GA 1  7 25 0 .  GA 1  7 25 0 .  GA 1  7 25 0 3  GA 1  7 15 0 5  GA 1  7 15 0 1
GA 1  8 25 1 2  GA 1  8 25 1 4  GA 1  8 25 1 4  GA 1  8 15 0 4  GA 1  8 15 0 .
GA 1  9 25 1 4  GA 1  9 25 1 .  GA 1  9 25 1 .  GA 1  9 15 0 5  GA 1  9 15 . 5
GA 1 10 25 0 4  GA 1 10 25 0 4  GA 1 10 25 0 4  GA 1 10 15 0 4  GA 1 10 15 0 3
GA 0 11 25 . 5  GA 0 11 25 1 2  GA 0 11 25 1 .  GA 0 11 15 . 3  GA 0 11 15 1 3
GA 0 12 25 1 4  GA 0 12 25 1 2  GA 0 12 25 1 2  GA 0 12 15 1 5  GA 0 12 15 1 .
GA 0 13 25 1 3  GA 0 13 25 1 1  GA 0 13 25 1 1  GA 0 13 15 0 1  GA 0 13 15 0 4
GA 0 14 25 1 .  GA 0 14 25 1 3  GA 0 14 25 1 .  GA 0 14 15 1 4  GA 0 14 15 1 2
GA 0 15 25 1 .  GA 0 15 25 1 5  GA 0 15 25 1 5  GA 0 15 15 1 4  GA 0 15 15 1 .
GA 0 16 25 0 5  GA 0 16 25 0 2  GA 0 16 25 0 2  GA 0 16 15 0 .  GA 0 16 15 0 2
GA 0 17 25 0 1  GA 0 17 25 0 1  GA 0 17 25 0 1  GA 0 17 15 0 2  GA 0 17 15 0 3
GA 0 18 25 1 4  GA 0 18 25 1 4  GA 0 18 25 1 4  GA 0 18 15 0 .  GA 0 18 15 0 .
GA 0 19 25 0 3  GA 0 19 25 0 5  GA 0 19 25 . 5  GA 0 19 15 0 4  GA 0 19 15 0 5
GA 0 20 25 1 1  GA 0 20 25 1 4  GA 0 20 25 1 4  GA 0 20 15 0 .  GA 0 20 15 0 .
GA 0 21 25 1 2  GA 0 21 25 1 2  GA 0 21 25 1 2  GA 0 21 15 0 3  GA 0 21 15 0 3
NC 1 22 30 1 3  NC 1 22 30 . 3  NC 1 22 30 1 3  NC 1 22 20 . 4  NC 1 22 20 1 4
NC 1 23 30 0 3  NC 1 23 30 0 3  NC 1 23 30 0 .  NC 1 23 20 0 5  NC 1 23 20 0 .
NC 1 24 30 . 4  NC 1 24 30 0 .  NC 1 24 30 . .  NC 1 24 20 1 .  NC 1 24 20 1 4
NC 1 25 30 0 3  NC 1 25 30 0 3  NC 1 25 30 0 3  NC 1 25 20 1 2  NC 1 25 20 1 2
NC 1 26 30 . 5  NC 1 26 30 1 5  NC 1 26 30 1 5  NC 1 26 20 1 1  NC 1 26 20 1 .
NC 1 27 30 1 .  NC 1 27 30 1 1  NC 1 27 30 . 1  NC 1 27 20 0 1  NC 1 27 20 . 1
NC 1 28 30 0 .  NC 1 28 30 0 .  NC 1 28 30 . 3  NC 1 28 20 1 3  NC 1 28 20 1 .
NC 1 29 30 1 1  NC 1 29 30 . 1  NC 1 29 30 1 .  NC 1 29 20 1 5  NC 1 29 20 1 5
NC 1 30 30 0 1  NC 1 30 30 0 1  NC 1 30 30 0 1  NC 1 30 20 1 2  NC 1 30 20 1 2
NC 1 31 30 0 3  NC 1 31 30 0 3  NC 1 31 30 0 3  NC 1 31 20 1 2  NC 1 31 20 1 2
NC 1 32 30 0 5  NC 1 32 30 . .  NC 1 32 30 0 5  NC 1 32 20 . 2  NC 1 32 20 0 2
NC 1 33 30 1 3  NC 1 33 30 1 3  NC 1 33 30 1 3  NC 1 33 20 0 1  NC 1 33 20 0 1
NC 1 34 30 0 3  NC 1 34 30 0 .  NC 1 34 30 0 .  NC 1 34 20 0 .  NC 1 34 20 0 5
NC 0 35 35 0 4  NC 0 35 35 0 2  NC 0 35 35 0 2  NC 0 35 20 1 3  NC 0 35 20 1 3
NC 0 36 35 0 2  NC 0 36 35 0 .  NC 0 36 35 0 .  NC 0 36 20 1 2  NC 0 36 20 . .
NC 0 37 35 1 4  NC 0 37 35 1 1  NC 0 37 35 1 1  NC 0 37 20 1 5  NC 0 37 20 1 5
NC 0 38 35 1 3  NC 0 38 35 . 3  NC 0 38 35 1 3  NC 0 38 20 . 2  NC 0 38 20 0 2
NC 0 39 35 0 3  NC 0 39 35 0 .  NC 0 39 35 . .  NC 0 39 20 0 .  NC 0 39 20 0 3
NC 0 40 35 1 4  NC 0 40 35 1 2  NC 0 40 35 1 2  NC 0 40 20 0 1  NC 0 40 20 0 1
SC 1 41 50 . 2  SC 1 41 50 0 5  SC 1 41 50 0 5  SC 1 41 40 1 .  SC 1 41 40 1 .
SC 1 42 50 1 3  SC 1 42 50 1 1  SC 1 42 50 1 1  SC 1 42 40 1 4  SC 1 42 40 1 4
SC 1 43 50 0 .  SC 1 43 50 0 .  SC 1 43 50 0 .  SC 1 43 40 0 .  SC 1 43 40 0 .
SC 1 44 50 0 .  SC 1 44 50 0 .  SC 1 44 50 0 3  SC 1 44 40 1 3  SC 1 44 40 1 .
SC 0 45 55 1 1  SC 0 45 55 1 .  SC 0 45 55 1 4  SC 0 45 48 1 4  SC 0 45 48 1 4
SC 0 46 55 1 5  SC 0 46 55 1 3  SC 0 46 55 1 3  SC 0 46 48 . 1  SC 0 46 48 0 1
SC 0 47 55 0 .  SC 0 47 55 0 2  SC 0 47 55 0 2  SC 0 47 48 0 2  SC 0 47 48 . 2
;
```

The following statements request the imputation of missing values for Department and Response by using the fully efficient fractional imputation (FEFI) method:

```
proc surveyimpute data=SIS_Survey_Sub method=fefi varmethod=jackknife;
   class Department Response;
   var Department Response;
   strata State NewUser;
```

```
    cluster School;
    weight SamplingWeight;
    output out=SIS_Survey_Imputed outjkcoefs=SIS_JKCoefs;
run;
```

The PROC SURVEYIMPUTE statement invokes the procedure. The DATA= option in the PROC SURVEYIMPUTE statement specifies the input data set containing the missing values, the METHOD=FEFI option requests the fully efficient fractional imputation method, and the VARMETHOD= option requests the imputation-adjusted jackknife replicate weights. The CLASS statement specifies the classification variables. The STRATA, CLUSTER, and WEIGHT statements specify the strata, clusters (PSUs), and weight variables. The VAR statement specifies the variables to be imputed (Department and Response). By default, both the variables Department and Response are imputed jointly. Therefore, the missing values for Department will be imputed conditionally on the observed levels of Response, and the missing values for Response will be imputed conditionally on the observed levels of Department. Observations that contain missing values for both Department and Response will be imputed by using the joint observed levels of Department and Response. The OUT= option in the OUTPUT statement names a SAS data set to save the imputed data. The OUTJKCOEFS= option in the OUTPUT statement names a SAS data set to save the jackknife coefficients.

Summary information about the data, CLASS levels, and survey design is shown in Figure 119.1. The "Imputation Information" table summarizes the imputation information. The "Number of Observations" table displays the number of observations that PROC SURVEYIMPUTE reads and uses. This table also displays the sum of weights that are read and used. The sum of weights read (6,468) can be used as an estimator of the population size. For example, the 235 observation units in the SIS_Survey_Sub data set represent 6,468 teachers and administrative staff in the population. The "Class Level Information" table shows that Department has two levels and Response has five levels. The "Design Summary" table shows that 47 schools are selected in the sample from six strata.

Figure 119.1 Summary Information

The SURVEYIMPUTE Procedure

Imputation Information	
Data Set	WORK.SIS_SURVEY_SUB
Weight Variable	SamplingWeight
Stratum Variables	State
	NewUser
Cluster Variable	School
Imputation Method	FEFI
Replication Method	JACKKNIFE
Number of Replicates	47

Number of Observations Read	235
Number of Observations Used	235
Sum of Weights Read	6468
Sum of Weights Used	6468

Class Level Information		
Class	**Levels**	**Values**
Department	2	0 1
Response	5	1 2 3 4 5

Figure 119.1 *continued*

Design Summary	
Number of Strata	6
Number of Clusters	47

The "Missing Data Patterns" table in Figure 119.2 lists distinct missing data patterns along with their corresponding frequencies and weighted percentages. An "X" means that the variable is observed in the corresponding group, and a "." means that the variable is missing. The table also displays group-specific variable means. In this hypothetical example, five respondents have unit nonresponse (both variables in the VAR statement contain missing values), 73 respondents have item nonresponse (only one variable in the VAR statement contains a missing value), and 157 respondents have complete response (no variables in the VAR statement contain missing values). Among the 73 item nonrespondents, for 52 respondents, Department is observed but Response is not observed; for 21 respondents, Response is observed but Department is not observed. The estimated percentages in the sample for unit nonresponse, item nonresponse, and complete response are 2.1%, 31.1%, and 66.8%, respectively.

Figure 119.2 Missing Data Patterns

Missing Data Patterns

Group	Department	Response	Freq	Sum of Weights	Unweighted Percent	Weighted Percent
1	X	X	157	4272	66.81	66.05
2	X	.	52	1480	22.13	22.88
3	.	X	21	586	8.94	9.06
4	.	.	5	130	2.13	2.01

Missing Data Patterns

	Group Means						
Group	Department 0	Department 1	Response 1	Response 2	Response 3	Response 4	Response 5
1	0.440309	0.559691	0.184457	0.206695	0.265684	0.209738	0.133427
2	0.641892	0.358108	.	.	.	.	.
3	.	.	0.261092	0.235495	0.230375	0.085324	0.187713
4	.	.	.	.	.	.	.

The "Imputation Summary" table in Figure 119.3 lists the number of nonmissing observations, missing observations, and imputed observations. There are 78 observations that have missing values for at least one variable, and all 78 missing observations are imputed.

Figure 119.3 Imputation Summary

Imputation Summary

Observation Status	Number of Observations	Sum of Weights
Nonmissing	157	4272
Missing	78	2196
Missing, Imputed	78	2196
Missing, Not Imputed	0	0

The output data set SIS_Survey_Imputed contains the observed data and the imputed values for Department and Response. In addition, this data set contains the imputation-adjusted full-sample weight (ImpWt), observation unit identification (UnitId), recipient index (ImpIndex), and imputation-adjusted jackknife replicate weights (ImpRepWt_1, . . . , ImpRepWt_47).

Suppose you want to compute frequency tables by using the imputed data set. The following statements request one-way tables for Department and Response and a two-way table for Department by Response. The analyses include the imputed values and account for both the design variance and the imputation variance.

```
proc surveyfreq data=SIS_Survey_Imputed varmethod=jackknife;
   table department response  department*response;
   weight ImpWt;
   repweights ImpRepWt: / jkcoefs=SIS_JKCoefs;
run;
```

The DATA= option in the PROC SURVEYFREQ statement specifies the input data set for analysis, SIS_Survey_Imputed, which contains the observed values and the imputed values for Department and Response. The FEFI technique uses multiple donor cells for a missing item. Therefore, the number of rows in the SIS_Survey_Imputed data set is greater than the number of rows in the observed data set, SIS_Survey_Sub. Each row in the SIS_Survey_Sub data set represents an observation unit, but this is not true for the SIS_Survey_Imputed data set. Therefore, it is very important to use only the weighted statistics from SIS_Survey_Imputed. The WEIGHT statement specifies the weight variable ImpWt, which is adjusted for the FEFI method. The imputation-adjusted jackknife replicate weights are saved in the variables ImpRepWt_1, . . . , ImpRepWt_47 in the SIS_Survey_Imputed data set. The REPWEIGHTS statement names the replicate weight variables and the jackknife coefficients data set, SIS_JKCOEFS. You should not use the unadjusted full-sample weights (SamplingWeight) or unadjusted replicate weights along with the imputed data.

Figure 119.4 displays some summary information. Note that the sum of weights in Figure 119.4 matches the sum of weights read from Figure 119.1, but the number of observations in Figure 119.4 (509) does not match the number of observations from Figure 119.1 (235). The sum of weights from both PROC SURVEYIMPUTE and PROC SURVEYFREQ represents the population size. The number of observations in Figure 119.1 represents the number of observation units, but the number of observations in Figure 119.4 represents the number of rows in the data set that include the observed units and the imputed rows. The number of replicates is 47, which is the same as the number of schools (PSUs).

Figure 119.4 One-Way Table

The SURVEYFREQ Procedure

Data Summary	
Number of Observations	509
Sum of Weights	6468

Variance Estimation	
Method	Jackknife
Replicate Weights	SIS_SURVEY_IMPUTED
Number of Replicates	47

Figure 119.5 displays one-way tables for Department and Response. The Frequency column does not represent frequencies for observation units from the SIS_Survey_Sub data set. These frequencies represent

the frequency of data lines in the SIS_Survey_Sub data set, which also include the imputed rows. The Weighted Frequency, Std Err of Wgt Freq, Percent, and Std Err of Percent columns use the imputation-adjusted full-sample weight and replicate weights. You should use the weighted statistics from these columns. For example, an estimated 49.47% of SIS users are teachers, with a standard error of 6.64%. An estimate of "Very Satisfied" users is 14.19%, with a standard error of 3.77%.

Figure 119.5 One-Way Table

Table of Department

Department	Frequency	Weighted Frequency	Std Err of Wgt Freq	Percent	Std Err of Percent
0	278	3200	429.52229	49.4729	6.6407
1	231	3268	429.52229	50.5271	6.6407
Total	509	6468	0	100.0000	

Table of Response

Response	Frequency	Weighted Frequency	Std Err of Wgt Freq	Percent	Std Err of Percent
1	100	1256	291.92305	19.4153	4.5133
2	103	1371	361.02585	21.1976	5.5817
3	112	1710	305.26968	26.4371	4.7197
4	100	1213	283.69298	18.7598	4.3861
5	94	917.82544	243.87967	14.1903	3.7706
Total	509	6468	0	100.0000	

Figure 119.6 displays the two-way table for Department by Response. The Weighted Frequency, Std Err of Wgt Freq, Percent, and Std Err of Percent columns use the imputation-adjusted full-sample weight and replicate weights. You should use the weighted statistics from these columns. Among the school personnel, an estimated 8.10% are teachers who are "Very Satisfied", and 6.09% are administrators who are "Very Satisfied". The standard errors are 3.11% and 2.43%, respectively.

Figure 119.6 Crosstabulation

Table of Department by Response						
Department	Response	Frequency	Weighted Frequency	Std Err of Wgt Freq	Percent	Std Err of Percent
0	1	57	637.83724	246.18741	9.8614	3.8062
	2	55	743.50947	334.28335	11.4952	5.1683
	3	64	951.95811	258.23015	14.7180	3.9924
	4	49	342.84458	150.44168	5.3006	2.3259
	5	53	523.75680	200.99126	8.0977	3.1075
	Total	278	3200	429.52229	49.4729	6.6407
1	1	43	617.94159	209.04386	9.5538	3.2320
	2	48	627.55128	185.53346	9.7024	2.8685
	3	48	757.99401	237.82609	11.7191	3.6770
	4	51	870.53830	262.37514	13.4592	4.0565
	5	41	394.06863	156.95381	6.0926	2.4266
	Total	231	3268	429.52229	50.5271	6.6407
Total	1	100	1256	291.92305	19.4153	4.5133
	2	103	1371	361.02585	21.1976	5.5817
	3	112	1710	305.26968	26.4371	4.7197
	4	100	1213	283.69298	18.7598	4.3861
	5	94	917.82544	243.87967	14.1903	3.7706
	Total	509	6468	0	100.0000	

Syntax: SURVEYIMPUTE Procedure

The following statements are available in the SURVEYIMPUTE procedure. Items within < > are optional.

```
PROC SURVEYIMPUTE <options> ;
    BY variables ;
    CELLS variables ;
    CLASS variable <(options)> <... variable <(options)>> </ options> ;
    CLUSTER variables ;
    ID variable ;
    IMPJOINT <variables> ;
    OUTPUT <OUT=SAS-data-set><OUTJKCOEFS=SAS-data-set><keyword=name
            ... keyword=name> ;
    REPWEIGHTS variables ;
    STRATA variables </ option> ;
    VAR variable <(options)> <... variable <(options)>> </ options> ;
    WEIGHT variable ;
```

The PROC SURVEYIMPUTE and VAR statements are required.

The following sections describe the PROC SURVEYIMPUTE statement and then describe the other statements in alphabetical order.

PROC SURVEYIMPUTE Statement

PROC SURVEYIMPUTE < *options* > **;**

The PROC SURVEYIMPUTE statement invokes the SURVEYIMPUTE procedure. The DATA= option identifies the data set to be analyzed. Table 119.1 summarizes the options available in the PROC SURVEYIMPUTE statement.

Table 119.1 Options Available in the PROC SURVEYIMPUTE Statement

Option	Description
DATA=	Names the input data set
METHOD=	Specifies the imputation method
NDONORS=	Specifies the number of donors for a recipient
NOPRINT	Suppresses all displayed output
ORDER=	Specifies the sort order of CLASS variables
RATE=	Specifies the sampling rate for the primary sampling units
SEED=	Specifies the random number seed
TOTAL=	Specifies the total number of primary sampling units
VARMETHOD=	Specifies the variance estimation method

You can specify the following *options*.

DATA=*SAS-data-set*

names the SAS data set that contains the data to be analyzed. If you omit the DATA= option, PROC SURVEYIMPUTE uses the most recently created SAS data set.

METHOD=FEFI | FHDI | HOTDECK | HD < (*method-option***) >**

specifies the imputation method to impute missing values for all variables in the VAR statement.

Table 119.2 summarizes the available *method-options*.

Table 119.2 Imputation Methods

METHOD=	Imputation Method	*Method-Options*
FEFI	Fully efficient fractional imputation method	ABSEMWTCONV= MAXDONORCELLS= MAXEMITER= RELEMWTCONV=
FHDI	Fractional hot-deck imputation method	ABSEMWTCONV= DISP=MEAN DISP=SSCP MAXDONORCELLS= MAXEMITER= RELEMWTCONV= REPWTADJ=RATIO REPWTADJ=NEIGHBOR REPWTADJ=NONE SELECTION=PPSPERPATN SELECTION=PPSPEROBS
HOTDECK | HD	Approximate Bayesian bootstrap Simple random sampling without replacement Simple random sampling with replacement Weighted selection	SELECTION=ABB SELECTION=SRSWOR SELECTION=SRSWR SELECTION=WEIGHTED

By default, if all variables that you specify in the VAR statement are also specified in the CLASS statement, then METHOD=FEFI. Otherwise, the default imputation method is METHOD=HOTDECK. You can specify the following values:

FEFI < (*method-options*) >

requests the fully efficient fractional imputation (FEFI) method. For more information, see the section "Fully Efficient Fractional Imputation" on page 9924.

You can specify the following *method-options*:

ABSEMWTCONV=*r*

specifies the absolute weighted convergence criterion. The expectation maximization (EM) algorithm stops when the maximum absolute difference between the fractional weights from the previous iteration and the fractional weights from the current iteration is less than *r*. The default value of *r* is 0.00001. For more information, see the section "FEFI Algorithm" on page 9924.

MAXDONORCELLS=*i*

specifies the maximum number (*i*) of donor cells allowed for a recipient unit. If the maximum number of donor cells exceeds MAXDONORCELLS=, then no imputation is performed. By default, MAXDONORCELLS=5000.

MAXEMITER=*i*
specifies the maximum number (*i*) of iterations for the expectation maximization (EM) algorithm. By default, MAXEMITER=100.

RELEMWTCONV=*r*
specifies a relative weighted convergence criterion. The expectation maximization (EM) algorithm stops when the maximum absolute relative difference between the weights from the previous iteration and the weights from the current iteration is less than *r*. For more information, see the section "FEFI Algorithm" on page 9924. The default value of *r* is 0.001.

FHDI < (*method-options***) >**
requests the fractional hot-deck imputation (FHDI) method. For more information, see the section "Fractional Hot-Deck Imputation" on page 9940.

You can specify the following *method-options* in parentheses:

ABSEMWTCONV=*r*
specifies the absolute weighted convergence criterion. The expectation maximization (EM) algorithm stops when the maximum absolute difference between the first-stage fractional weights from the previous iteration and the first-stage fractional weights from the current iteration is less than *r*. For more information, see the section "FEFI Algorithm" on page 9924. The default value of *r* is 0.00001.

DISP=*stat*
displays the weighted mean or the crossproduct of the weighted mean sum of squares for variables that are specified in the VAR statement but not in the CLASS statement.

You can specify one of the following *stats*:

MEAN displays the weighted mean.

SSCP displays both the weighted mean and the crossproduct of the weighted mean sum of squares.

Displayed statistics are from both two-stage FEFI and FHDI because PROC SURVEYIMPUTE must perform two-stage FEFI in order to determine the donor sets for FHDI. The closer the means and crossproducts for FHDI are to the ones for two-stage FEFI, the more confidence you can have that FHDI is as efficient as FEFI.

MAXDONORCELLS=*i*
specifies the maximum number (*i*) of second-stage donor cells allowed for a recipient unit. If the maximum number of second-stage donor cells exceeds *i*, then no imputation is performed. By default, MAXDONORCELLS=5000.

MAXEMITER=*i*
specifies the maximum number (*i*) of iterations for the expectation maximization (EM) algorithm for the first-stage FEFI. By default, MAXEMITER=100.

RELEMWTCONV=*r*

specifies the relative weighted convergence criterion. The expectation maximization (EM) algorithm for the first-stage FEFI stops when the maximum absolute relative difference between the weights from the previous iteration and the weights from the current iteration is less than *r*. For more information, see the section "FEFI Algorithm" on page 9924. The default value of *r* is 0.001.

REPWTADJ=*replicate-adjustment-option*

adjusts the replicate weights for FHDI. For more information, see the section "Replicate Weight Adjustments for FHDI" on page 9959. You can specify one of the following values for *replicate-adjustment-option*:

NEIGHBOR adjusts replicate weights by using the sum of replicate fractional weights in neighborhoods that are defined by the full sample fractional weights from two-stage FEFI.

NONE does not adjust the replicate weights for the selection of donors after two-stage FEFI.

RATIO adjusts replicate weights by using the ratio of replicate fractional weights and the full sample fractional weights from two-stage FEFI.

By default, REPWTADJ=NEIGHBOR.

SELECTION=*selection-option*

specifies how to perform the probability proportional to size (PPS) with replacement selection. For more information, see "Second-Stage Selection" in section "Fractional Hot-Deck Imputation Algorithm" on page 9943. You can specify one of the following *selection-options*:

PPSPEROBS performs independent selection of second-stage donor cells for every observation unit that is a recipient for the second-stage imputation.

PPSPERPATN performs one selection of second-stage donor cells for all observation units that are recipients for the second-stage imputation and have the same first-stage FEFI levels.

By default, SELECTION=PPSPERPATN for METHOD=FHDI.

HOTDECK < (SELECTION=*selection-option***) >**

HD < (SELECTION=*selection-option***) >**

requests the hot-deck imputation method. For more information, see the section "Hot-Deck Imputation" on page 9952.

By default, SELECTION=SRSWR for METHOD=HOTDECK if you do not use the WEIGHT statement, and SELECTION=WEIGHTED for METHOD=HOTDECK if you use the WEIGHT statement. You can specify one of the following donor selection *selection-options*:

ABB
requests donor selection by using the approximate Bayesian bootstrap method. For more information, see the section "Approximate Bayesian Bootstrap" on page 9953.

SRSWOR
requests donor selection by using simple random samples without replacement. For more information, see the section "Simple Random Samples without Replacement" on page 9953.

SRSWR
requests donor selection by using simple random samples with replacement. For more information, see the section "Simple Random Samples with Replacement" on page 9954.

WEIGHTED
requests donor selection by using probability proportional to respondent weights with replacement. For more information, see the section "Weighted Selection" on page 9954.

NDONORS=*r*
specifies the number of donor units *r*, where *r* is either of the following:

- the number of donor units to be used to impute every recipient unit when METHOD=HOTDECK
- the maximum number of second-stage donor cells to be used to impute second-stage missing items conditional on the first-stage FEFI levels when METHOD=FHDI

If you specify NDONORS=0 for METHOD=HOTDECK, then no imputation is performed.

When METHOD=FEFI, the SURVEYIMPUTE procedure performs fully efficient fractional imputation, for which the NDONORS= option does not apply.

By default, NDONORS=1 for METHOD=HOTDECK and NDONORS=10 for METHOD=FHDI.

NOPRINT
suppresses all displayed output. This option temporarily disables the Output Delivery System (ODS); for more information about ODS, see Chapter 23, "Using the Output Delivery System."

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

RATE=*value* | *SAS-data-set*

R=*value* | *SAS-data-set*

specifies the sampling rate, which PROC SURVEYIMPUTE uses to compute a finite population correction for bootstrap replicate weights. This option is ignored for the BRR and jackknife replication methods.

If your sample design has multiple stages, you should specify the *first-stage sampling rate*, which is the ratio of the number of primary sampling units (PSUs) that are selected to the total number of PSUs in the population.

You can specify the sampling rate in either of the following ways:

value specifies a nonnegative number to use for a nonstratified design or for a stratified design that has the same sampling rate in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the sampling rates for a stratified design that has different sampling rates in the strata. You must provide the sampling rates in the data set variable named _RATE_.

The sampling rates must be nonnegative numbers. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC SURVEYIMPUTE converts that number to a proportion. The procedure treats the value 1 as 100% instead of 1%.

For more information, see the section "Population Totals and Sampling Rates" on page 9920.

If you do not specify the RATE= or TOTAL= option, then the bootstrap replicate weights do not include a finite population correction. You cannot specify both the TOTAL= option and the RATE= option in the same PROC SURVEYIMPUTE statement.

SEED=*number*

specifies the initial random number generation seed for selecting donor units for METHOD=HOTDECK or for selecting second-stage donor cells for METHOD=FHDI. The *number* should be a positive integer. If you do not specify this option or if *number* is 0, PROC SURVEYIMPUTE uses the time of day from the computer's clock to obtain the initial seed. For more information, see the section "Random Number Generation" on page 9923.

TOTAL=*value* | *SAS-data-set*

specifies the total number of primary sampling units (PSUs) in the population. PROC SURVEYIMPUTE uses the *value* to compute a finite population correction for bootstrap replicate weights. This option is ignored for the BRR and jackknife replication methods.

You can specify the total number of PSUs in either of the following ways:

value specifies a positive number to use for a nonstratified design or for a stratified design that has the same population total in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the population totals for a stratified design that has different population totals in the strata. You must provide the stratum totals in the data set variable named _TOTAL_.

The stratum totals must be positive numbers.

For more information, see the section "Population Totals and Sampling Rates" on page 9920.

If you do not specify the TOTAL= or RATE= option, then the bootstrap replicate weights do not include a finite population correction. You cannot specify both the TOTAL= option and the RATE= option in the same PROC SURVEYIMPUTE statement.

VARMETHOD=*method* **<(***method-options***)>**
REPWEIGHTSTYPE=*method* **<(***method-options***)>**
computes imputation-adjusted replicate weights.

Table 119.3 summarizes the available *methods* and *method-options*.

Table 119.3 Replicate Weights Options

method	**Replicate Weights Method**	*method-options*
BOOTSTRAP	Bootstrap	MH=*number* REPS=*number*
BRR	Balanced repeated replication	FAY <=*value*> HADAMARD=*SAS-data-set* PRINTH REPS=*number*
JACKKNIFE	Jackknife	None
NONE	No replicate weight computation	None

By default, VARMETHOD=JACKKNIFE when METHOD=FEFI or METHOD=FHDI, and VARMETHOD=NONE when METHOD=HOTDECK.

You can specify the following *methods*:

BOOTSTRAP <(*method-options***)>**
computes imputation-adjusted bootstrap replicate weights. The bootstrap method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you provide replicate weights by using a REPWEIGHTS statement. For more information, see the section "Bootstrap Method" on page 9955.

You can specify the following *method-options*:

MH=*value* | (*values*)

specifies the number of PSUs to select for the bootstrap replicate samples. You can provide bootstrap stratum sample sizes m_h by specifying a list of *values*. Alternatively, you can provide a single bootstrap sample size *value* to use for all strata or for a nonstratified design. For more information, see the section "Bootstrap Method" on page 9955.

Each bootstrap sample size, m_h, must be a positive integer and must be less than n_h, which is the total number of PSUs in stratum h. By default, $m_h = n_h - 1$ for a stratified design. For a nonstratified design, the bootstrap sample size *value* must be less than n (the total number of PSUs in the sample). By default, $m = n - 1$ for a nonstratified design. You can provide the bootstrap sample size by specifying one of the following forms:

MH=*value*

specifies a single bootstrap sample size *value* to use for all strata or for a nonstratified design.

MH=(*values***)**

specifies a list of stratum bootstrap sample size *values*. You can separate the values by using blanks or commas, and you must enclose the list of values in parentheses. The number of values must not be less than the number of strata in the DATA= input data set.

The order of the stratum sample size values must match the order of the stratum levels in the DATA= input data set. Each stratum sample size value must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

REPS=*number*

specifies the *number* of replicates for bootstrap variance estimation, where *number* must be an integer greater than 1. Increasing the number of replicates improves the estimation precision but also increases the computation time. By default, REPS=250.

BRR < (*method-options***) >**

computes the imputation-adjusted balanced repeated replication (BRR) weights. The BRR method requires a stratified sample design with two primary sampling units (PSUs) in each stratum. If you specify the VARMETHOD=BRR option, you must also use a STRATA statement unless you provide replicate weights by using a REPWEIGHTS statement. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 9955.

You can specify the following *method-options* in parentheses after the VARMETHOD=BRR option:

FAY < =*value* **>**

requests Fay's method, which is a modification of the BRR method. For more information, see the section "Unadjusted Fay's BRR Replicate Weights" on page 9956.

You can specify the *value* of the Fay coefficient, which is used in converting the original sampling weights to replicate weights. The Fay coefficient must be a nonnegative number less than 1. By default, the value of the Fay coefficient is 0.5.

HADAMARD=*SAS-data-set*

H=*SAS-data-set*

names a SAS data set that contains the Hadamard matrix for BRR replicate construction. If you do not provide a Hadamard matrix by using this *method-option*, PROC SURVEYIMPUTE generates an appropriate Hadamard matrix for replicate construction. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 9955 and "Hadamard Matrix" on page 9957.

If a Hadamard matrix of a particular dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a SAS data set in the HADAMARD= *method-option*.

In the HADAMARD= input data set, each variable corresponds to a column of the Hadamard matrix, and each observation corresponds to a row of the matrix. You can use any variable names in the HADAMARD= data set. All values in the data set must equal either 1 or –1. You must ensure that the matrix you provide is indeed a Hadamard matrix—that is, $\mathbf{A}'\mathbf{A} = R\mathbf{I}$, where $\mathbf{A}$ is the Hadamard matrix of dimension R and $\mathbf{I}$ is an identity matrix. PROC SURVEYIMPUTE does not check the validity of the Hadamard matrix that you provide.

The HADAMARD= input data set must contain at least H variables, where H denotes the number of first-stage strata in your design. If the data set contains more than H variables, PROC SURVEYIMPUTE uses only the first H variables. Similarly, the HADAMARD= input data set must contain at least H observations.

If you do not specify the REPS= *method-option*, then the number of replicates is equal to the number of observations in the HADAMARD= input data set. If you specify the number of replicates—for example, REPS=*nreps*—then the procedure uses the first *nreps* observations in the HADAMARD= data set to construct the replicates.

You can specify the PRINTH *method-option* to display the Hadamard matrix that the procedure uses to construct replicates for BRR.

PRINTH

displays the Hadamard matrix that is used to construct replicates for BRR. When you provide the Hadamard matrix in the HADAMARD= *method-option*, PROC SURVEYIMPUTE displays only the rows and columns that are actually used to construct replicates. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 9955 and "Hadamard Matrix" on page 9957.

The PRINTH *method-option* is not available when you use a REPWEIGHTS statement to provide replicate weights, because the procedure does not use a Hadamard matrix in this case.

REPS=*number*

specifies the number of replicates for BRR variance estimation. The value of *number* must be an integer greater than 1.

If you do not provide a Hadamard matrix by using the HADAMARD= *method-option*, the number of replicates should be greater than the number of strata and should be a multiple of 4. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 9955. If a Hadamard matrix cannot be constructed for the REPS= value that you

specify, the value is increased until a Hadamard matrix of that dimension can be constructed. Therefore, it is possible for the actual number of replicates used to be larger than the REPS= value that you specify.

If you provide a Hadamard matrix by using the HADAMARD= *method-option*, the value of REPS= must not be greater than the number of rows in the Hadamard matrix. If you provide a Hadamard matrix and do not specify the REPS= *method-option*, the number of replicates equals the number of rows in the Hadamard matrix.

If you do not specify the REPS= or HADAMARD= *method-option* and do not include a REPWEIGHTS statement, the number of replicates equals the smallest multiple of 4 that is greater than the number of strata.

If you provide replicate weights by using a REPWEIGHTS statement, PROC SURVEYIMPUTE does not use the REPS= *method-option*. When you use a REPWEIGHTS statement, the number of replicates equals the number of REPWEIGHTS variables.

JACKKNIFE
JK

computes the imputation-adjusted jackknife replicate weights. For more information, see the section "Jackknife Method" on page 9957.

NONE

does not compute replicate weights.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SURVEYIMPUTE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SURVEYIMPUTE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Note that using a BY statement provides completely separate imputation within the BY groups. For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CELLS Statement

CELLS *variables* **;**

The CELLS statement names the variables that identify the imputation cells. The imputation cells divide the data into groups of similar units. The combination of levels of CELLS variables defines the imputation cells. If you do not use this statement, then all observation units are assumed to be in one imputation cell.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *options* > **;**

The CLASS statement names the classification variables for the analysis. Most *options* can be specified either as individual variable *options* or as global *options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify global *options* for the CLASS statement by placing the *options* after a slash (/). Global *options* are applied to every variable that is specified without an *option* in the CLASS statement. However, individual CLASS variable *options* override the global *options*.

You can specify the following *options* either as global options or as individual options:

DESCENDING
DESC

reverses the sort order of the classification variable. If you specify both the DESCENDING and ORDER= options, PROC SURVEYIMPUTE orders the categories according to the ORDER= option and then reverses that order.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables.

The following table shows how PROC SURVEYIMPUTE interprets values of the ORDER= option:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values. For ORDER=FREQ, the frequency counts are unweighted. For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

TRUNCATE <=*n*>

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify the TRUNCATE option without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

CLUSTER Statement

CLUSTER *variables* **;**

The CLUSTER statement names variables that identify the first-stage clusters in a clustered sample design. First-stage clusters are also known as primary sampling units (PSUs). The combinations of categories of CLUSTER variables define the clusters in the sample. If you also use the STRATA statement, clusters are nested within strata.

If your sample design has clustering at multiple stages, you should specify only the first-stage clusters (PSUs) in the CLUSTER statement. For more information, see the section "Specifying the Sample Design" on page 9919.

If you provide replicate weights in a REPWEIGHTS statement, you do not need to use a CLUSTER statement.

The CLUSTER *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric, but PROC SURVEYIMPUTE treats them as categorical variables. The formatted values of the CLUSTER variables determine the cluster variable levels. Thus, you can use formats to group values into levels. For more information, see the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

You can use multiple CLUSTER statements to specify CLUSTER variables. PROC SURVEYIMPUTE uses variables from all CLUSTER statements to create clusters.

ID Statement

ID *variable* **;**

The ID statement names a *variable* in the DATA= input data set to identify observation units. When you use an OUTPUT statement, the OUT= data set includes the ID variable. PROC SURVEYIMPUTE uses the ID variable values to identify the donor observations. If you do not use an ID statement, the procedure creates a new variable named UnitID that uses the observation numbers to identify the donor observations.

IMPJOINT Statement

IMPJOINT < *variables* > **;**

The IMPJOINT statement specifies the names of variables that are to be imputed jointly for the fully efficient fractional imputation (FEFI) method. If you do not use the IMPJOINT statement, then all the variables that you specify in the VAR statement are imputed jointly. You can use multiple IMPJOINT statements. The levels of the variables in the IMPJOINT statement describe a nonparametric imputation model for the expectation maximization (EM) step for the fractional imputation.

If you use the following IMPJOINT statements, then the variables y1, y2, and y3 are imputed jointly, and the variables y4 and y5 are imputed jointly:

```
IMPJOINT y1 y2 y3;
IMPJOINT y4 y5;
```

Analysis variables, which you specify in the VAR statement, can appear in only one IMPJOINT statement. However, auxiliary variables that you do not specify in the VAR statement can be specified in multiple IMPJOINT statements.

If you do not specify any variable names in the IMPJOINT statement, then every variable in the VAR statement is imputed marginally.

The IMPJOINT statement is ignored when you specify the METHOD=HOTDECK or METHOD=FHDI option in the PROC SURVEYIMPUTE statement.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < **OUTJKCOEFS=***SAS-data-set* > < *keyword***=***name* . . . *keyword***=***name* >
;

The OUTPUT statement creates a SAS data set that contains the imputed data. You must use the OUTPUT statement to store the imputed data in a SAS data set. If you use multiple OUTPUT statements, then PROC SURVEYIMPUTE uses only the first OUTPUT statement and ignores the rest. The OUTPUT OUT= data set contains all the variables from the DATA= input data set, imputed values for missing values for the variables in the VAR statement, and some observation-level quantities. These quantities can include the fractionally adjusted weights, replicate weights, recipient numbers, and donor identifications.

You can specify the following in the OUTPUT statement:

OUT=*SAS-data-set*
names the output data set. If you use the OUTPUT statement but omit the OUT= option, then the output data set is named by using the DATA*n* convention. For more information, see the section "OUT= Output Data Set" on page 9961.

OUTJKCOEFS=*SAS-data-set*
names a SAS data set that contains the jackknife coefficients for VARMETHOD=JACKKNIFE.

keyword < =*name* >

specifies the quantities to include in the output data set and optionally names the new variables that contain the quantities. Specify a *keyword* for each desired quantity (see the following list of *keywords*), optionally followed by an equal sign and a variable name to contain the quantity. If you specify a *keyword* without a variable *name*, then PROC SURVEYIMPUTE uses default names. You can specify the following *keywords*:

DONORID< =*name* >

requests a name for the identification variable for the donor units. If you do not specify this keyword, the donor IDs are not saved in the output data set. If you specify this keyword but do not specify a *name*, then the donor IDs are stored in a new variable named DonorID. This keyword is available only when METHOD=HOTDECK.

FRACTIONALWEIGHTS=*name*

includes the fractional weights of donor cells in the output data set and specifies the corresponding variable *name*. If you do not specify this keyword, the fractional weights are not saved in the output data set. This keyword is available when METHOD=FEFI or METHOD=FHDI.

IMPADJWEIGHTS=*name*

includes the imputation-adjusted weights in the output data set and specifies the corresponding variable *name*. The imputation-adjusted weights are computed by multiplying the base weights by the fractional weights. If you do not specify this keyword, then the imputation-adjusted weights are stored in a new variable named ImpWt. This keyword is available when METHOD=FEFI or METHOD=FHDI.

IMPSTATUS=*name*

includes an imputation status index with the values shown in Table 119.4.

Table 119.4 Imputation Status Index

Index	Imputation Status
0	All items are observed
1	All missing items are imputed
2	No missing items are imputed
3	Some missing items are imputed but some missing items are not imputed
4	Invalid observation; observation is not used in the imputation

OBSID=*name*

includes an index variable to contain the unique numeric identification of every unit from the input data set in the output data and specifies the corresponding variable *name*. If you do not specify this keyword, then the default unit ID is stored in a new variable named UnitID. The OBSID= option is not applicable when the ID statement is specified.

IMPINDEX=*name*

includes the imputation index in the output data set and specifies the corresponding variable *name*. The imputation index can be 0 (which indicates a nonmissing unit) or any positive integer (which represents multiple donor units for a recipient unit). If you do not specify this keyword, then the imputation index is stored in a new variable named ImpIndex.

OUTCONTLEVELS < =YES | NO >

specifies whether to include in the OUTPUT OUT= data set the imputed values for the variables that are specified in the CLEVVAR= option in the VAR statement or the imputed levels for the imputation bins (when the CLEVELS= option is specified in the VAR statement). This option does not apply when METHOD=HOTDECK.

By default, the imputed values for the variables that are specified in the CLEVVAR= option or the imputed levels for the imputation bins are included in the OUTPUT OUT= data. Optionally, you can specify the following keywords:

YES includes the imputed values for CLEVVAR= variables or the imputed levels for the imputation bins (when the CLEVELS= option is used) in the OUTPUT OUT= data set.

NO does not include the imputed values for CLEVVAR= variables or the imputed levels for the imputation bins in the OUTPUT OUT= data set.

REPWEIGHTS Statement

REPWEIGHTS *variables* **;**

The REPWEIGHTS statement names variables that provide replicate weights.

If you use a REPWEIGHTS statement and you specify METHOD=HOTDECK in the PROC SURVEYIMPUTE statement, the procedure does not adjust the replicate weights.

If you use a REPWEIGHTS statement and you specify METHOD=FEFI or METHOD=FHDI in the PROC SURVEYIMPUTE statement, the procedure adjusts the replicate weights.

Each REPWEIGHTS variable should contain the weights for a single replicate, and the number of replicates should equal the number of REPWEIGHTS variables. The REPWEIGHTS variables must be numeric, and the variable values must be nonnegative numbers.

If you provide replicate weights in a REPWEIGHTS statement, you do not need to use a CLUSTER or STRATA statement.

If you use a REPWEIGHTS statement but do not use a WEIGHT statement, PROC SURVEYIMPUTE uses the average of each observation's replicate weights as the observation's weight.

STRATA Statement

STRATA *variables* < / *option* > **;**

The STRATA statement names one or more *variables* that form the strata in a stratified sample design. The combinations of levels of STRATA variables define the strata in the sample, where strata are nonoverlapping subgroups that were sampled independently.

If your sample design has stratification at multiple stages, you should identify only the first-stage strata in the STRATA statement. For more information, see the section "Specifying the Sample Design" on page 9919.

If you provide replicate weights in a REPWEIGHTS statement, you do not need to use a STRATA statement.

The STRATA *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric, but PROC SURVEYIMPUTE treats them as categorical variables. The formatted values of the STRATA variables determine the STRATA variable levels. Thus, you can use formats to group values into levels. For more information, see the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in the *SAS Formats and Informats: Reference*.

You can specify the following *option* in the STRATA statement after a slash (/):

LIST

displays a "Stratum Information" table, which lists all strata together with the corresponding values of the STRATA variables. This table provides the number of observations, the number of clusters, and the sum of weights in each stratum, as well as the sampling fraction if you specify the RATE= or TOTAL= option.

VAR Statement

VAR *variable* < (*options*) > < . . . *variable* < (*options*) > > < / *global-options* > **;**

The VAR statement names the analysis variables to be imputed. The analysis variables can be either character or numeric. The categorical variables in the VAR statement, which can be either character or numeric, must also be specified in the CLASS statement. Only variables that you specify in the VAR statement are imputed. If you specify METHOD=FEFI or METHOD=FHDI, then you must also specify a CLEVELS= or CLEVVAR= option for all numeric variables that are not specified in the CLASS statement.

A *variable* in the VAR statement should not appear in any of the BY, CLUSTER, REPWEIGHTS, STRATA, and WEIGHT statements.

You can specify the following *global-options* or *options* in the VAR statement:

CLEVELS=*k*

specifies the number of levels by which to categorize numeric variables that are in the VAR statement but not in the CLASS statement. The procedure divides the range of these variables into *k* equally spaced bins. You can specify the CLEVELS= option either as a *global-option* or as an individual *option*.

CLEVVAR=*variable*
specifies a variable that contains the bins for a numeric variable that is specified in the VAR statement but not in the CLASS statement. Both the CLEVVAR= *variable* and the numeric variable to which the CLEVVAR=*variable* applies to should have the same missing values.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement names the variable that contains the sampling weights. This variable must be numeric, and the sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, then PROC SURVEYIMPUTE omits that observation from the analysis. For more information, see the section "Missing Values" on page 9921.

If you do not use a WEIGHT statement but you provide replicate weights in a REPWEIGHTS statement, PROC SURVEYIMPUTE uses the average of each observation's replicate weights as the observation's weight.

If you use neither a WEIGHT statement nor a REPWEIGHTS statement, PROC SURVEYIMPUTE assigns all observations a weight of 1.

Details: SURVEYIMPUTE Procedure

Definitions

The following definitions are used in this chapter:

- *Analysis variables:* The variables whose missing values are imputed. These variables are specified in the VAR statement.
- *Augmented (output) data:* The data set that contains both observed data and imputed data.
- *Auxiliary variables:* The variables that are used to impute other variables. Missing values in the auxiliary variables are not imputed. Auxiliary variables are specified in the IMPJOINT statement or CELLS statement, but not in the VAR statement.
- *Discretized variable:* The variable that contains the discretized levels of a continuous variable. These levels are used as imputation bins for the continuous variable. See also *imputation bins*.
- *Donor cell:* The collection of donor units that have the same observed levels for all items. Donor cells that have the same observed levels as the observed levels for the nonmissing items in a recipient unit are used to impute the missing values in that recipient unit.
- *Donor unit:* The observation unit that provides a value for imputation.
- *Fractional replicate weight:* The fraction of weight that a donor cell contributes to the recipient unit in a replicate sample. Fractional replicate weights are proportions that lie between 0 and 1.

- *Fractional weight:* The fraction of weight that a donor cell contributes to the recipient unit. Fractional weights are proportions that lie between 0 and 1.
- *Fully efficient fractional weight:* The fractional weight from the fully efficient fractional imputation.
- *Imputation-adjusted replicate weight:* The fraction of the recipient weight that a donor cell contributes to the recipient unit in a replicate sample.
- *Imputation-adjusted weight:* The fraction of the recipient weight that a donor cell contributes to the recipient unit.
- *Imputation bins:* A partition of the range of observed values for a continuous variable on which the first-stage imputation is applied. See also *discretized variables*.
- *Imputation cell:* A partition of the input data within which imputation is performed independently.
- *Observation row:* The same as the *observation unit* for observation units that have no missing values. Otherwise, an observation row is one realization of imputed values. Each row in the output data set represents one observation row.
- *Observation unit:* "An object on which a measurement is taken." (Lohr 2022).
- *Recipient unit:* The *observation unit* that contains missing values.
- *Sampling unit:* An object that can be selected in a sample. Probabilities of selection are assigned to the sampling units.
- *Set of donor cells:* The set of all possible configurations (vectors of observed values) that a particular observation unit that has missing items can take for a particular imputation model. Each configuration represents a donor cell. See also *donor cells*.

Specifying the Sample Design

PROC SURVEYIMPUTE produces replicate weights that are based on the sample design that is used to collect the survey data. You can use PROC SURVEYIMPUTE for single-stage or multistage designs, with or without stratification, and with or without unequal weighting. To create imputation-adjusted replicate weights for your survey data, you need to provide sample design information to PROC SURVEYIMPUTE. This information can include design (or variance) strata, clusters, and sampling weights. You provide sample design information by specifying the STRATA, CLUSTER, WEIGHT, and REPWEIGHTS statements and the RATE= or TOTAL= option in the PROC SURVEYIMPUTE statement.

If you use the REPWEIGHTS statement to provide replicate weights, you do not need to use a STRATA or CLUSTER statement. Otherwise, you should use STRATA and CLUSTER statements whenever your design includes stratification and clustering. If your design includes unequal sampling weights, you should use the WEIGHT statement.

For a multistage sample design, PROC SURVEYIMPUTE uses only the first stage of the sample design to create replicate weights. Therefore, the required input includes only the first-stage cluster (PSU) identification and first-stage stratum identification. You do not need to input design information about any additional stages of sampling.

Stratification

If your sample design is stratified at the first stage of sampling, use the STRATA statement to name the variables that form the strata. The combinations of categories of STRATA variables define the strata in the sample, where strata are nonoverlapping subgroups that were sampled independently. If your sample design has stratification at multiple stages, then identify only the first-stage strata in the STRATA statement.

If you use a REPWEIGHTS statement to provide replicate weights, you do not need to use a STRATA statement. Otherwise, you should use a STRATA statement whenever your design includes stratification. If you do not use a STRATA statement or a REPWEIGHTS statement, then PROC SURVEYIMPUTE assumes there is no stratification at the first stage; that is, the procedure assumes that all observation units are in the same stratum.

Clustering

If your sample design selects clusters at the first stage of sampling, use the CLUSTER statement to name the variables that identify the first-stage clusters, which are also called primary sampling units (PSUs). The combinations of categories of CLUSTER variables define the clusters in the sample. If there is a STRATA statement, clusters are nested within strata. If your sample design has clustering at multiple stages, you should specify only the first-stage clusters (PSUs) in the CLUSTER statement. PROC SURVEYIMPUTE assumes that each cluster that is defined by the variables in the CLUSTER statement represents a PSU in the sample.

If you use a REPWEIGHTS statement to provide replicate weights, you do not need to use a CLUSTER statement. Otherwise, you should use a CLUSTER statement whenever your design includes clustering at the first stage of sampling. If you do not use a CLUSTER statement, then PROC SURVEYIMPUTE treats each observation as a PSU.

Weighting

If your sample design includes unequal weighting, use the WEIGHT statement to name the variable that contains the sampling weights. Sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, then PROC SURVEYIMPUTE omits that observation from the analysis. For more information, see the section "Missing Values" on page 9921.

If you do not use a WEIGHT statement but you include a REPWEIGHTS statement, PROC SURVEYIMPUTE uses the average of each observation's replicate weights as the observation's weight. If you use neither a WEIGHT statement nor a REPWEIGHTS statement, PROC SURVEYIMPUTE assumes that all observations have a weight of 1.

Population Totals and Sampling Rates

To include a finite population correction (*fpc*) in the bootstrap replicate weights, you can specify either the sampling rate or the population total by using the RATE= or TOTAL= option, respectively, in the PROC SURVEYIMPUTE statement. You cannot specify both of these options in the same PROC SURVEYIMPUTE statement. The procedure does not use a finite population correction for BRR or jackknife variance estimation.

If you do not specify the RATE= or TOTAL= option, the bootstrap replicate weights do not include a finite population correction. For fairly small sampling fractions, this correction is often ignored. For more information, see Cochran (1977) and Kish (1965).

If your design has multiple stages of selection and you are specifying the RATE= option, you should use the first-stage sampling rate, which is the ratio of the number of PSUs in the sample to the total number of PSUs in the population. If you are specifying the TOTAL= option for a multistage design, you should use the total number of PSUs in the population.

For a nonstratified sample design, or for a stratified sample design that has the same sampling rate or same population total in all strata, you can use the RATE=*value* or TOTAL=*value* option. If your sample design is stratified with different sampling rates or population totals in different strata, use the RATE=*SAS-data-set* or TOTAL=*SAS-data-set* option to name a SAS data set that contains the stratum sampling rates or totals. This data set is called a *secondary data set*, as opposed to the *primary data set* that you specify by using the DATA= option.

The secondary data set must contain all the stratification variables that are listed in the STRATA statement and all the variables in the BY statement. Furthermore, the BY groups must appear in the same order as in the primary data set. If there are formats associated with the STRATA variables and BY variables, then the formats must be consistent in the primary and secondary data sets. If you specify the TOTAL=*SAS-data-set* option, the secondary data set must have a variable named _TOTAL_ that contains the stratum population totals. If you specify the RATE=*SAS-data-set* option, the secondary data set must have a variable named _RATE_ that contains the stratum sampling rates. If the secondary data set contains more than one observation for any one stratum, the procedure uses the first value of _TOTAL_ or _RATE_ for that stratum and ignores the rest.

The *value* in the RATE= option or the values of _RATE_ in the secondary data set must be nonnegative numbers. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC SURVEYIMPUTE converts that number to a proportion. The procedure treats the value 1 as 100% instead of 1%.

If you specify the TOTAL=*value* option, *value* must not be less than the sample size. If you provide stratum population totals in a secondary data set, these values must not be less than the corresponding stratum sample sizes.

Replicate Weights

If you have replicate weights available for your survey data, use the REPWEIGHTS statement to name the variables that contain the replicate weights. Replicate weights must be positive numbers. If an observation has a replicate weight that is nonpositive or missing, then PROC SURVEYIMPUTE does not perform any imputation. For more information, see the section "Missing Values" on page 9921.

Missing Values

You might have missing values in your data set for several reasons. Some common reasons are data entry error, ineligible items or units, and nonresponse. You should complete your data preparation (identify data entry error or ineligibility) and adjustment (fill in deterministic values or edits) before using the SURVEYIMPUTE procedure. Use PROC SURVEYIMPUTE to impute missing values that arise only from nonresponse. If you have observations that have missing values in variables that are not specified in the VAR statement, then the procedure does not impute those observations. The following subsections describe how PROC SURVEYIMPUTE treats missing values in the variables that are specified in some statements.

WEIGHT Statement Variable

If an observation has a missing value or a nonpositive value for the variable in the WEIGHT statement, then PROC SURVEYIMPUTE excludes that observation from the analysis. However, if you use the OUTPUT statement, the observation is included in the output data set.

REPWEIGHTS Statement Variables

If you provide replicate weights by using a REPWEIGHTS statement, the values for all variables in that statement must be nonmissing and nonnegative. PROC SURVEYIMPUTE does not perform the analysis when any replicate weight value is missing or nonpositive.

Variables in the CLUSTER and STRATA Statements

An observation is excluded from the analysis if it has a missing value for any variable in a CLUSTER or STRATA statement. However, if you use the OUTPUT statement, the observation is included in the output data set.

Variables in the CELLS Statement

An observation is excluded from the imputation if it has a missing value for any variable in the CELLS statement. However, if you specify the VARMETHOD=JK option in the PROC SURVEYIMPUTE statement, then the observation unit is used to create replicate weights, unless the observation unit has missing values in any of the variables in the STRATA, CLUSTER, or WEIGHT statement. If you use the OUTPUT statement, the observation is also included in the output data set.

Auxiliary Variables in the IMPJOINT Statement

Variables that you specify in the IMPJOINT statement but do not specify in the VAR statement are used as auxiliary variables in the imputation. If you have missing values in the auxiliary variables, then that observation unit is not used in the imputation. However, if you specify the VARMETHOD=JK option in the PROC SURVEYIMPUTE statement, then the observation unit is used to create replicate weights, unless the observation unit has missing values in any of the variables in the STRATA, CLUSTER, or WEIGHT statement. If you use the OUTPUT statement, the observation unit is also included in the output data set.

Variable in the ID Statement

If an observation unit has a missing value for the variable in the ID statement, then that observation is used in the imputation unless it also has missing values for variables in the STRATA, CLUSTER, or WEIGHT or CELLS statement. If the observation is selected as a donor unit for a recipient unit, then the donor identification for that recipient unit will also be missing.

Missing Data Patterns

The SURVEYIMPUTE procedure displays the missing data patterns in groups in a "Missing Data Patterns" table; the groups are based on whether the analysis variables are observed or missing. The input data set does not need to be sorted in any order.

For example, when a data set contains variables Z_1, Z_2, and Z_3 (in that order), up to eight groups of observations can be formed from the data set. Figure 119.7 displays the eight groups of observations and an unique missing pattern for each group.

Figure 119.7 Missing Data Patterns

Missing Data Patterns

Group	Z1	Z2	Z3
1	X	X	X
2	X	X	.
3	X	.	X
4	X	.	.
5	.	X	X
6	.	X	.
7	.	.	X
8	.	.	.

An "X" in Figure 119.7 means that the variable is observed, and a "." means that the variable is missing.

The order of the groups is determined by the order in which you list the variables in the VAR statement. If you specify a different order of variables in the VAR statement, then the results are different even if the other specifications remain the same.

Random Number Generation

The donor selection methods available in PROC SURVEYIMPUTE use random numbers in their selection algorithms. PROC SURVEYIMPUTE uses a uniform random number function to generate streams of pseudorandom numbers from an initial starting point, or seed. You can use the SEED= option to specify the initial seed. You can specify the same SEED= value (along with the same options and the same input data) to reproduce the imputation. If you do not specify the SEED= option, PROC SURVEYIMPUTE uses the time of day from the computer's clock to obtain the initial seed. For information about specifying the initial seed, see the SEED= option.

PROC SURVEYIMPUTE uses the Mersenne twister random number generator. The Mersenne twister generator (Matsumoto and Nishimura 1998) has a very long period ($2^{19937}-1$) and good statistical properties. The algorithm is a twisted generalized feedback shift register. This is the same random number generator that PROC SURVEYSELECT uses by default and that the RAND function provides for the uniform distribution. For more information, see *SAS Functions and CALL Routines: Reference*.

Fully Efficient Fractional Imputation

The fully efficient fractional imputation (FEFI) method uses multiple donor units for a recipient unit. The observation unit that contains the missing values is known as the recipient unit, and the observation unit that provides the value for imputation is known as the donor unit. The number of donor units for a recipient unit is equal to the number of observed levels for the missing items, given the observed levels for the nonmissing items of the recipient unit. Each donor donates a fraction of the original weight of the recipient unit such that the sum of the fractional weights from all the donors is equal to the original weight of the recipient. The fraction of the recipient weight that a donor unit contributes to the recipient unit is known as the *fractional weight*. The method is called fully efficient because it does not introduce additional variability that is caused by the selection of donor units (Kim and Fuller 2004). One disadvantage of the FEFI method is that it can greatly increase the size of the imputed data set. For more information, see Kalton and Kish (1984), Fuller (2009, Section 5.2.2), and Kim and Shao (2014, Section 4.6).

FEFI Algorithm

Suppose you want to impute P items jointly (by using the IMPJOINT statement in PROC SURVEYIMPUTE). Let $\mathbf{Z}_i = (Z_{i1}, \ldots, Z_{iP})$ be the true response for the P items for unit i. $\mathbf{Z}_i$ is completely known if all P items are observed for unit i. However, the true response might not be known for some units because of item nonresponse. Let Z_{ij} be categorical and have J levels for item j. Denote $\mathbf{Z}_{i,\text{obs}}$ as the observed part and $\mathbf{Z}_{i,\text{miss}}$ as the missing part of $\mathbf{Z}_i$. Let $\pi(\kappa_1\kappa_2\cdots\kappa_P)$ be the population proportion that falls in category $(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2, \cdots, Z_{iP} = \kappa_P)$. Assume that it is possible to estimate the population proportion from the observed sample. That is, for example, the conditional probability, $P(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2 | Z_{i3} = \kappa_3, \ldots, Z_{iP} = \kappa_p)$, in the observed data is the same as the conditional probability in the data where $(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2)$ are missing. The conditional probabilities are estimated by

$$\hat{P}(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2 | Z_{i3} = \kappa_3, \ldots, Z_{iP} = \kappa_p) = \left\{ \sum_{\kappa_1\kappa_2} \hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P) \right\}^{-1} \hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P)$$

where

$$\hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P) = \left\{ \sum_i w_i \right\}^{-1} \sum_i w_i I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

is the estimated joint probability, $I(.)$ is an indicator function, and w_i is the observation weight for unit i.

The FEFI method uses an EM-by-weighting algorithm similar to that of Ibrahim (1990). The detailed algorithm is described in Kim and Fuller (2013). The following steps describe the imputation technique. If you do not specify imputation cells by using the CELLS statement, PROC SURVEYIMPUTE uses the entire data set as one imputation cell. If you specify imputation cells, then all the probabilities in these steps are computed by using observations from the same imputation cell as the recipient unit. To simplify notation, subscripts are not used for imputation cells in the following description.

For given i, let $\mathbf{Z}_{i,\text{obs}}$ and $\mathbf{Z}_{i,\text{miss}}$ be the observed part and the missing part, respectively, of unit i. Let $\mathcal{A}_c$ be the index set for the complete respondents. Suppose you want to impute the missing part of $\mathbf{Z}_i$, $\mathbf{Z}_{i,\text{miss}}$. The index set $d_i = \{k : k \in \mathcal{A}_c \wedge \mathbf{Z}_{k,\text{obs}} = \mathbf{Z}_{i,\text{obs}}\}$ contains the indexes for the all possible donor units for $\mathbf{Z}_i$. Let $l = 1, 2, \ldots, M_l$ be all the observed combinations of $\{\mathbf{Z}_{k,\text{miss}} : k \in d_i\}$. The set of all observed combinations for unit i defines the donor cells (all possible realizations) for unit i. Let $\mathbf{Z}_{i,\text{miss}[l]}$ be the

lth imputed value of $\mathbf{Z}_{i,\text{miss}}$. You must assume that at least one imputed value is available; otherwise the observation is not imputed.

1. *Initialization:* For each observation that has missing items, determine the number of donor cells by using the number of unique combinations of observed levels for the missing items for the responding units in the imputation cell. Compute the initial fractional weight from donor cell l to unit i, $w_{il(0)}$, by

$$w_{il(0)} = \left\{ \sum_{k=1}^{M_l} \tilde{\pi}_{(0)}(\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{miss}[k]}) \right\}^{-1} \tilde{\pi}_{(0)}(\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{miss}[l]})$$

where $l = 1, 2, \ldots, M_l$ is the number of donor cells and

$$\tilde{\pi}_{(0)}(\kappa_1 \cdots \kappa_P) = \left\{ \sum_{i \in \mathcal{A}_c} w_i \right\}^{-1} \sum_{i \in \mathcal{A}_c} w_i I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

The sum of the fractional weights over all the donor cells is 1 for every observation unit; that is, $\sum_l w_{il(0)} = 1$, for all i. The lth imputed row for unit i is created by keeping the observed items unchanged, replacing the missing items with the observed levels from the lth donor cell, and computing the fractional weight by $w_i w_{il(0)}$. Only the complete respondents are used to compute the fractional weights in this step. If unit i has no missing items, then $w_{i1(0)} = 1$. The initial FEFI data set contains all the observed units, the imputed rows for observation that had missing items, and the corresponding fractional weights.

2. *M-step:* The tth M-step computes the joint probabilities by using the fractional weights from the (t–1)th E-step,

$$\tilde{\pi}_{(t)}(\kappa_1 \cdots \kappa_P) = \left\{ \sum_i \sum_l w_i w_{il(t-1)} \right\}^{-1} \sum_i \sum_l w_i w_{il(t-1)} I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

for all i, all l, and $t > 0$. Note that for $t > 0$, $\tilde{\pi}_{(t)}$ uses all observation units, including observations where missing items are imputed in the initialization step.

3. *E-step:* The tth E-step computes the fractional weights by using the joint probabilities $\tilde{\pi}_{(t)}(\kappa_1 \cdots \kappa_P)$ from the tth M-step. The tth fractional weight for unit i and donor cell l is given by

$$w_{il(t)} = \left\{ \sum_{k=1}^{M_l} \tilde{\pi}_{(t)}(\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{miss}[k]}) \right\}^{-1} \tilde{\pi}_{(t)}(\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{miss}[l]})$$

4. *Repetition:* The EM-steps are repeated for $t = 1, 2, \ldots,$ until the changes in fractional weights over all observation units between two successive EM-steps are negligible or the maximum number of EM repetitions is reached.

The maximum absolute difference convergence criterion, ϵ_{AD}, at step t is defined as

$$\max_{i,l} |w_{il(t)} - w_{il(t-1)}| \leq \epsilon_{\text{AD}}$$

The maximum absolute relative difference convergence criterion, ϵ_{RD}, at step t is defined as

$$\max_{i,l} |w_{il(t)} - w_{il(t-1)}| / w_{il(t-1)} \leq \epsilon_{\text{RD}}$$

where $w_{il(t-1)} > 0$.

The replicate weights are created by computing a replicated version of $\tilde{\pi}_{(t)}(\kappa_1\kappa_2\cdots\kappa_p)$, $\tilde{\pi}^{(k)}_{(t)}(\kappa_1\kappa_2\cdots\kappa_p)$, and by repeating the EM-by-weighting algorithm as described earlier. For the kth replicate sample, $\tilde{\pi}^{(k)}_{(t)}(\kappa_1\kappa_2\cdots\kappa_p)$ is computed by

$$\tilde{\pi}^{(k)}_{(t)}(\kappa_1\cdots\kappa_P) = \left\{\sum_i\sum_l w_i^{(k)} w_{il(t-1)}^{(k)}\right\}^{-1} \sum_i\sum_l w_i^{(k)} w_{il(t-1)}^{(k)} I(Z_{i1}=\kappa_1,\ldots,Z_{iP}=\kappa_P)$$

Example of FEFI

The small data set shown in Figure 119.8 is used to illustrate the imputation technique. The data set contains nine observation units, and each unit has two items (X and Y). The variable Unit contains the observation identification. In this example, X is missing for units 5 and 9, and Y is missing for units 2 and 9.

Figure 119.8 Sample Data with Missing Items

Unit	X	Y
1	0	0
2	0	.
3	0	1
4	0	0
5	.	1
6	1	0
7	1	1
8	1	1
9	.	.

The following SAS statements request joint imputation of X and Y by using the FEFI method. These statements also request imputation-adjusted replicate weights for the jackknife replication method. The CLASS statement specifies that both X and Y are CLASS variables. The OUTPUT statement stores the imputed values in the data set Imputed and stores the jackknife coefficients in the data set Ojkc. The FRACTIONALWEIGHTS= option in the OUTPUT statement saves the fractional weights in the Imputed data set.

```
proc surveyimpute data=test varmethod=jackknife;
   class x y;
   var x y;
   id Unit;
   output out=Imputed fractionalweights=FracWgt outjkcoefs=Ojkc;
run;
```

The initial fractional weights, FracWgt, after the initialization step are displayed in Figure 119.9.

- Observation unit 1 has no missing value. Therefore, the ImpIndex value is 0, the FracWgt value is 1, and the values of X and Y are the same as the observed values for observation unit 1 in Figure 119.9. Because all observation units have a weight of 1, the fractional weights, FracWgt, and the imputation-adjusted weights, ImpWt, are the same for all rows.

- Observation unit 2 has a missing Y. The observed level for X for unit 2 is 0. For X = 0, two levels for Y are observed: Y = 0, which has a proportion (FracWgt) of 0.67, and Y = 1, which has a proportion of 0.33. Therefore, observation unit 2 receives two donor cells (ImpIndex = 1 and ImpIndex = 2), whose initial fractional weights are 0.67 and 0.33, respectively. Because X is observed, the X values in both rows for unit 2 are the same as the observed value. However, the first recipient row for unit 2 has an imputed Y value of 0, the second recipient row for unit 2 has an imputed Y value of 1, and each has a corresponding initial fractional weight.

- Observation unit 5 has a missing X. The observed level for Y for unit 5 is 1. To impute X, note that two levels of X are observed when Y = 1: X = 0 with a proportion of 0.33 and X = 1 with a proportion of 0.67. The two recipient rows for observation unit 5 contain the initial fractional weights in the FracWgt column and the imputed X values.

- Observation unit 9 has missing values for both X and Y. From the observed data, X and Y can take the following values: (X = 0, Y = 0) with probability 0.33, (X = 0, Y = 1) with probability 0.17, (X = 1, Y = 0) with probability 0.17, and (X = 1, Y = 1) with probability 0.33. The four imputed rows (ImpIndex 1, ImpIndex 2, ImpIndex 3, and ImpIndex 4) for observation unit 9 represent the four observed combinations for X and Y along with their initial fractional weights.

The resulting data set contains 14 rows. There are six rows for fully observed units (ImpIndex = 0), two rows for unit 2, two rows for unit 5, and four rows for unit 9. The sum of initial fractional weights is 1 for all units.

Figure 119.9 Fractional Imputation after Initialization

Unit	ImpIndex	ImpWt	FracWgt	X	Y
1	0	1.00000	1.00000	0	0
2	1	0.66667	0.66667	0	0
2	2	0.33333	0.33333	0	1
3	0	1.00000	1.00000	0	1
4	0	1.00000	1.00000	0	0
5	1	0.33333	0.33333	0	1
5	2	0.66667	0.66667	1	1
6	0	1.00000	1.00000	1	0
7	0	1.00000	1.00000	1	1
8	0	1.00000	1.00000	1	1
9	1	0.33333	0.33333	0	0
9	2	0.16667	0.16667	0	1
9	3	0.16667	0.16667	1	0
9	4	0.33333	0.33333	1	1

The EM algorithm repeats the computation of the joint probabilities and the fractional weights until convergence. The fractional weights, FracWgt, after the EM step and the imputation-adjusted replicate weights (ImpRepWt_1, . . . , ImpRepWt_9) are displayed in Figure 119.10.

Figure 119.10 Fractional Imputation after the EM

Unit	ImpIndex	ImpWt	FracWgt	X	Y	ImpRepWt_1	ImpRepWt_2	ImpRepWt_3	ImpRepWt_4
1	0	1.00000	1.00000	0	0	0.00000	1.12500	1.12500	1.12500
2	1	0.58601	0.58601	0	0	0.46072	0.00000	1.12498	0.46072
2	2	0.41399	0.41399	0	1	0.66428	0.00000	0.00002	0.66428
3	0	1.00000	1.00000	0	1	1.12500	1.12500	0.00000	1.12500
4	0	1.00000	1.00000	0	0	1.12500	1.12500	1.12500	0.00000
5	1	0.41399	0.41399	0	1	0.49821	0.37510	0.00002	0.49821
5	2	0.58601	0.58601	1	1	0.62679	0.74990	1.12498	0.62679
6	0	1.00000	1.00000	1	0	1.12500	1.12500	1.12500	1.12500
7	0	1.00000	1.00000	1	1	1.12500	1.12500	1.12500	1.12500
8	0	1.00000	1.00000	1	1	1.12500	1.12500	1.12500	1.12500
9	1	0.32330	0.32330	0	0	0.22659	0.32143	0.48214	0.22659
9	2	0.22840	0.22840	0	1	0.32669	0.21434	0.00001	0.32669
9	3	0.12500	0.12500	1	0	0.16071	0.16071	0.16071	0.16071
9	4	0.32330	0.32330	1	1	0.41101	0.42851	0.48214	0.41101

ImpRepWt_5	ImpRepWt_6	ImpRepWt_7	ImpRepWt_8	ImpRepWt_9
1.12500	1.12500	1.12500	1.12500	1.12500
0.75009	0.65906	0.62682	0.62682	0.65877
0.37491	0.46594	0.49818	0.49818	0.46623
1.12500	1.12500	1.12500	1.12500	1.12500
1.12500	1.12500	1.12500	1.12500	1.12500
0.00000	0.46601	0.66443	0.66443	0.46623
0.00000	0.65899	0.46057	0.46057	0.65877
1.12500	0.00000	1.12500	1.12500	1.12500
1.12500	1.12500	0.00000	1.12500	1.12500
1.12500	1.12500	1.12500	0.00000	1.12500
0.42862	0.41563	0.41109	0.41109	0.00000
0.21424	0.29384	0.32672	0.32672	0.00000
0.16071	0.00000	0.16071	0.16071	0.00000
0.32143	0.41553	0.22648	0.22648	0.00000

Two-Stage Fully Efficient Fractional Imputation

The two-stage fully efficient fractional imputation method uses multiple donor units for a recipient unit. Each donor donates a fraction of the original weight of the recipient unit such that the sum of the fractional weights from all the donors is equal to the original weight of the recipient. The fraction of the recipient weight that a donor unit contributes to the recipient unit is known as the *fractional weight*. The method is called fully efficient because it does not introduce additional variability that is caused by the selection of donor units (Kim and Fuller 2004). Two-stage FEFI has two hierarchical imputation stages. One disadvantage of the two-stage FEFI method is that it can greatly increase the size of the imputed data set.

Two-stage FEFI is useful when you want to impute variables that have many unique observed values. FEFI creates many imputed rows if the variable that you are imputing has many unique observed values. Two-stage FEFI imputes these variables conditional on the imputed levels from the first-stage FEFI. Thus, two-stage FEFI often uses fewer imputed rows compared to a similar first-stage FEFI.

Variables that have many observed levels are grouped into imputation bins. The first-stage imputation is performed for all categorical variables by using the FEFI method. The categorical variables include the character variables, the CLASS variables that you also specify in the VAR statement, and the variables that contain the imputation bins of the continuous variables.

The second-stage imputation is performed for the continuous variables within each first-stage donor cell. Observations that contain missing values in any of the continuous items are the recipients, and observations that contain observed values for these missing items are the donors. The second-stage donor cells are defined by the unique combinations of the observed values for the continuous variables within the first-stage donor cells.

Imputation-adjusted replicate weights are computed by repeating both the first-stage and second-stage imputation in every replicate sample independently.

The method is similar to Im, Kim, and Fuller (2015).

Two-Stage FEFI Algorithm

Suppose you want to impute P items jointly. Let $\mathbf{X}_i = (X_{i1}, \ldots, X_{iP_1})$ be the response for P_1 items in unit i, and let $\mathbf{Y}_i = (Y_{i1}, \ldots, Y_{iP_2})$ be the response for P_2 items in unit i, where $P = P_1 + P_2$. Let X_{ij} be categorical with J levels for item j, and let Y_{ij} be continuous. Further assume that $\tilde{\mathbf{Y}} = (\tilde{Y}_{i1}, \ldots, \tilde{Y}_{iP_2})$ contains the discretized levels (imputation bins) for Y, where $\tilde{Y}_{ij}$ has J levels. Define $Z_{ij} = (X_{ij}, \tilde{Y}_{ij})$. Then Z_{ij} is categorical and has J levels for item j. Denote $\mathbf{Z}_{i,\text{obs}}$ as the observed part and $\mathbf{Z}_{i,\text{miss}}$ as the missing part of $\mathbf{Z}_i$.

Let $\pi(\kappa_1\kappa_2\cdots\kappa_P)$ be the population proportion that falls in category $\kappa_1\kappa_2\cdots\kappa_P$. Assume that it is possible to estimate the population categories from the observed sample. For example, the conditional probability, $P(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2 | Z_{i3} = \kappa_3, \ldots, Z_{iP} = \kappa_p)$, is the same for the observed data as it is for the data where $(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2)$ are missing. The initial conditional probabilities are estimated by

$$\hat{P}(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2 | Z_{i3} = \kappa_3, \ldots, Z_{iP} = \kappa_p) = \left\{ \sum_{\kappa_1\kappa_2} \hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P) \right\}^{-1} \hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P)$$

where

$$\hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P) = \left\{ \sum_i w_i \right\}^{-1} \sum_i w_i I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

is the estimated joint probability, $I(.)$ is an indicator function, and w_i is the observation weight for unit i.

Let $l = 1, 2, \ldots, M_{il_1}$ be all the observed combinations of $\mathbf{Z}_{k:k\notin i,\text{miss}}$ in the sample. Let $\mathbf{Z}_{i,\text{miss}[l]}$ be the lth realization of $\mathbf{Z}_{i,\text{miss}}$ in the sample. You must assume that at least one realization is available; otherwise, the observation is not imputed.

The two-stage FEFI method first computes the fully efficient fractional weights by using an EM-by-weighting algorithm like that of Kim and Fuller (2013) to impute the missing values in $\mathbf{Z}_i$. The missing values in $\mathbf{Y}_i$ are imputed in the second-stage imputation. Two-stage FEFI weights for imputing $\mathbf{Y}_i$ are computed independently in every imputed level of $\mathbf{Z}_{i,\text{miss}[l]}$, where $l = 1, \ldots, M_{l_1}$ is the number of first-stage donor cells.

The following steps describe the two-stage FEFI technique. If you do not use the CELL statement to specify imputation cells, PROC SURVEYIMPUTE uses the entire data set as one imputation cell. If you specify

imputation cells, then all the probabilities are computed by using observations from the same imputation cell as the recipient unit. To simplify notation, subscripts are not used for imputation cells in the following description. Imputation cells are defined for the first-stage imputation. Steps 1 to 4 describe the first-stage FEFI for the categorical variables, which also include the imputation bins for the continuous variables. Step 5 describes the second-stage FEFI.

1. *Initialization*: For each observation that has missing items, determine the number of first-stage donor cells. The first-stage donor cells are determined by using the number of unique combinations of observed levels in $\mathbf{Z}_i$ for imputing the missing items in $\mathbf{Z}_i$. Only the responding units in the imputation cell are used to determine the number of first-stage donor cells. Compute the initial fractional weight from donor cell l to unit i, $w_{il(0)}$, by

$$w_{il(0)} = \left\{ \sum_{k=1}^{M_{il_1}} \tilde{\pi}_{(0)}(\mathbf{Z}_{i,\mathrm{obs}}, \mathbf{Z}_{i,\mathrm{miss}[k]}) \right\}^{-1} \tilde{\pi}_{(0)}(\mathbf{Z}_{i,\mathrm{obs}}, \mathbf{Z}_{i,\mathrm{miss}[l]})$$

where $l = 1, 2, \ldots, M_{l_1}$ is the number of first-stage donor cells and

$$\tilde{\pi}_{(0)}(\kappa_1, \ldots, \kappa_P) = \left\{ \sum_{i \in \mathcal{A}_c} w_i \right\}^{-1} \sum_{i \in \mathcal{A}_c} w_i I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

The sum of the fractional weights over all the donor cells is 1 for every observation unit; that is, $\sum_l w_{il(0)} = 1$ for all i. The lth imputed row for unit i is created by keeping the observed items unchanged, replacing the missing items with the observed levels from the lth donor cell, and computing the fractional weight by $w_i w_{il(0)}$. Only the complete observations (observations that have no missing items) are used to compute the fractional weights in this step. If unit i has no missing items, then $w_{i1(0)} = 1$. The initial FEFI data set contains all the observed units, the imputed rows for observations that have missing items, and the corresponding fractional weights.

2. *M-step*: The tth maximization step (M-step) computes the joint probabilities by using the fractional weights from the (t–1)th expectation-step,

$$\tilde{\pi}_{(t)}(\kappa_1, \ldots, \kappa_P) = \left\{ \sum_i \sum_l w_i w_{il(t-1)} \right\}^{-1} \sum_i \sum_l w_i w_{il(t-1)} I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

for all i, all l, and $t > 0$. Note that for $t > 0$, $\tilde{\pi}_{(t)}$ uses all observation units, including observations that have missing items that are imputed in the initialization step.

3. *E-step*: The tth expectation (E-step) computes the fractional weights by using the joint probabilities $\tilde{\pi}_{(t)}(\kappa_1, \ldots, \kappa_P)$ from the tth M-step. The tth fractional weight for unit i and donor cell l is given by

$$w_{il(t)} = \left\{ \sum_{k=1}^{M_{il_1}} \tilde{\pi}_{(t)}(\mathbf{Z}_{i,\mathrm{obs}}, \mathbf{Z}_{i,\mathrm{miss}[k]}) \right\}^{-1} \tilde{\pi}_{(t)}(\mathbf{Z}_{i,\mathrm{obs}}, \mathbf{Z}_{i,\mathrm{miss}[l]})$$

4. *Repetition*: The expectation maximization steps (EM-steps, steps 2 and 3) are repeated for $t = 1, 2, \ldots,$ until the changes in fractional weights over all observation units between two successive EM-steps are negligible or the maximum number of EM repetitions is reached.

The maximum absolute difference convergence criterion, ϵ_{AD}, at step t is defined as

$$\max_{i,l} |w_{il(t)} - w_{il(t-1)}| \leq \epsilon_{\mathrm{AD}}$$

The maximum absolute relative difference convergence criterion, ϵ_{RD}, at step t is defined as

$$\max_{i,l} |w_{il(t)} - w_{il(t-1)}| / w_{il(t-1)} \leq \epsilon_{\mathrm{RD}}$$

where $w_{il(t-1)} > 0$.

5. *Second-stage imputation*: The second-stage imputation replaces the missing values in the continuous variables by using the observed values within each selected first-stage donor cell. This step is similar to step 1 but is applied to impute the continuous variables.

   For a particular observation unit i, let $\mathbf{Z}_{i,\mathrm{dcell}[l_1]} = (\mathbf{Z}_{i,\mathrm{obs}}, \mathbf{Z}_{i,\mathrm{mis}[l_1]})$ be the l_1th donor cell from the first-stage imputation, where l_1 ranges from 1 to M_{l_1}. For each observation unit, i, the possible number of second-stage donor cells is equal to the number of unique combinations of the observed levels for the missing items in $\mathbf{Y}_i$ from the responding units in the first-stage donor cell l_1.

   Let $\pi_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1}\cdots\kappa_{P_2|l_1})$ be the population proportion that falls in category $\kappa_{1|l_1}\kappa_{2|l_1}\cdots\kappa_{P_2|l_1}$. Assume that it is possible to estimate the population categories from the observed sample. For example, the conditional probability, $P(Y_{i1|l_1} = \kappa_{1|l_1}, Y_{i2|l_1} = \kappa_{2|l_1} | Y_{i3|l_1} = \kappa_{3|l_1}, \ldots, Y_{iP_2|l_1} = \kappa_{P_2|l_1})$, is the same for the observed data as it is for the data in which $(Y_{i1|l_1} = \kappa_{1|l_1}, Y_{i2|l_1} = \kappa_{2|l_1})$ are missing. The conditional probabilities are estimated by

$$\hat{P}(Y_{i1|l_1} = \kappa_{1|l_1}, Y_{i2|l_1} = \kappa_{2|l_1} | Y_{i3|l_1} = \kappa_{3|l_1}, \ldots, Y_{iP_2|l_1} = \kappa_{P_2|l_1}) = \left\{\sum_{\kappa_{1|l_1}\kappa_{2|l_1}} \hat{\pi}_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1}\cdots\kappa_{P_2|l_1})\right\}^{-1} \hat{\pi}_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1}\cdots\kappa_{P_2|l_1})$$

   where

$$\hat{\pi}_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1}\cdots\kappa_{P_2|l_1}) = \left\{\sum_{i\in\mathbf{Z}_{i,\mathrm{dcell}[l_1]}} w_i\right\}^{-1} \sum_{i\in\mathbf{Z}_{i,\mathrm{dcell}[l_1]}} w_i I(Y_{i1|l_1} = \kappa_{1|l_1}, \ldots, Y_{iP_2|l_1} = \kappa_{P_2|l_1})$$

   is the estimated joint probability, $I(.)$ is an indicator function, and w_i is the observation weight for unit i.

   Let $l = 1, 2, \ldots, M_{il_2|l_1}$ be all the observed combinations of $\mathbf{Y}_{k:k\in\mathbf{Z}_{i,\mathrm{dcell}[l_1]},k\neq i,\mathrm{miss}}$ in the sample. Let $\mathbf{Y}_{i,\mathrm{miss}[l]}$ be the lth realization of $\mathbf{Y}_{i,\mathrm{miss}}$ in the sample. You must assume that at least one realization is available; otherwise, missing values in the continuous items for the observation are not imputed.

   Compute the second-stage fractional weight from the second-stage donor cell l_2 conditional on the first-stage donor cell l_1 for unit i, $w_{il_2|l_1}$:

$$w_{il_2|l_1} = \left\{\sum_{k=1}^{M_{il_2|l_1}} \tilde{\pi}_{2|l_1}(\mathbf{Y}_{i,\mathrm{obs}}, \mathbf{Y}_{i,\mathrm{miss}[k]})\right\}^{-1} \tilde{\pi}_{2|l_1}(\mathbf{Y}_{i,\mathrm{obs}}, \mathbf{Y}_{i,\mathrm{miss}[l_2]})$$

   where $l = 1, 2, \ldots, M_{il_2|l_1}$ is the number of second-stage donor cells and

$$\tilde{\pi}_{2|l_1}(\kappa_1, \ldots, \kappa_{P_2}) = \left\{\sum_{i\in\mathbf{Z}_{i,\mathrm{dcell}[l_1]}} w_i\right\}^{-1} \sum_{i\in\mathbf{Z}_{i,\mathrm{dcell}[l_1]}} w_i I(Y_{i1|l_1} = \kappa_1, \ldots, Y_{iP_2|l_1} = \kappa_{P_2})$$

The sum of the second-stage fractional weights over all second-stage donor cells is 1 for every observation unit; that is, $\sum_{l_2} w_{il_2|l_1} = 1$ for all l_1 and i. The l_2th second-stage imputed row in the l_1th first-stage imputed row for unit i is created by keeping the observed items unchanged, replacing the missing items in $\mathbf{Y}_i$ with the observed values from the l_2th second-stage donor cell, and computing the two-stage fractional weight by $w_{il_1l_2} = w_{il_1} w_{il_2|l_1}$, where w_{il_1} is the first-stage fractional weight for the first-stage donor cell l_1. The maximum number of donor cells for unit i is $M_{il_1} M_{il_2|l_1}$. Only the complete observations are used to compute the second-stage fractional weights.

The imputation-adjusted replicate weights are created by using the following:

1. The first-stage FEFI is repeated by using replicate weights in every replicate sample.
2. The second-stage replicate fractional weights for the kth replicate are computed by using the estimated joint probabilities from the kth replicate sample:

$$\hat{\pi}^{(k)}_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1}\cdots\kappa_{P_2|l_1}) = \left\{ \sum_{i\in\mathbf{Z}_{i,\mathrm{dcell}[l_1]}} w_i^{(k)} \right\}^{-1} \sum_{i\in\mathbf{Z}_{i,\mathrm{dcell}[l_1]}} w_i^{(k)} I(Y_{i1|l_1} = \kappa_{1|l_1}, \ldots, Y_{iP_2|l_1} = \kappa_{P_2|l_1})$$

where $w_i^{(k)}$ are the unadjusted replicate weights.

Example of Two-Stage FEFI

The small data set shown in Figure 119.11 is used to illustrate the imputation technique. The data set contains 18 observation units, and each unit has four items (X, CX, Y, and CY). The variable Unit contains the observation identification. Variables CX and CY contains the imputation bins for variables X and Y, respectively. In this example, X and CX are missing for units 14 and 18, and Y and CY are missing for units 5 and 18.

Figure 119.11 Sample Data with Missing Items

Unit	X	CX	Y	CY
1	0.3	0	-0.54	0
2	0.2	0	-0.77	0
3	1.7	0	-0.59	0
4	1.7	0	-0.59	0
5	1.0	0	.	.
6	1.8	0	-0.03	1
7	2.0	0	0.95	1
8	1.9	0	0.78	1
9	6.7	1	-0.15	0
10	6.0	1	-1.01	0
11	3.3	1	-1.86	0
12	7.3	1	-0.21	0
13	6.7	1	0.80	1
14	.	.	1.23	1
15	2.9	1	0.65	1
16	9.6	1	0.95	1
17	10.0	1	0.13	1
18	.	.	.	.

The following statements request joint imputation of X and Y by using the two-stage FEFI method. Two CLEVVAR= options specify variables CX and CY, which contain the imputation bins for variables X and Y, respectively. The following statements also request imputation-adjusted replicate weights for the jackknife replication method. The OUTPUT statement stores the imputed values in the Imputed data set and stores the jackknife coefficients in the OJKC data set. The FRACTIONALWEIGHTS= option in the OUTPUT statement saves the fractional weights in the Imputed data set.

```
proc surveyimpute data=Example method=fefi;
   var X (clevvar=CX) Y (clevvar=CY);
   output out=Imputed fractionalweights=FracWt outjkcoefs=OJKC;
run;
```

The first-stage FEFI imputes the imputation bin variables CX and CY by using the FEFI method. The imputed data set after the first-stage imputation is displayed in Figure 119.12. Variables X and Y are not imputed in the first-stage imputation.

Figure 119.12 First-Stage Fractional Imputation

Unit	ImpIndex	ImpWt	FracWt	X	CX	Y	CY
1	0	1.0000	1.0000	0.3	0	-0.54	0
2	0	1.0000	1.0000	0.2	0	-0.77	0
3	0	1.0000	1.0000	1.7	0	-0.59	0
4	0	1.0000	1.0000	1.7	0	-0.59	0
5	1	0.5360	0.5360	1.0	0	.	0
5	2	0.4640	0.4640	1.0	0	.	1
6	0	1.0000	1.0000	1.8	0	-0.03	1
7	0	1.0000	1.0000	2.0	0	0.95	1
8	0	1.0000	1.0000	1.9	0	0.78	1
9	0	1.0000	1.0000	6.7	1	-0.15	0
10	0	1.0000	1.0000	6.0	1	-1.01	0
11	0	1.0000	1.0000	3.3	1	-1.86	0
12	0	1.0000	1.0000	7.3	1	-0.21	0
13	0	1.0000	1.0000	6.7	1	0.80	1
14	1	0.4640	0.4640	.	0	1.23	1
14	2	0.5360	0.5360	.	1	1.23	1
15	0	1.0000	1.0000	2.9	1	0.65	1
16	0	1.0000	1.0000	9.6	1	0.95	1
17	0	1.0000	1.0000	10.0	1	0.13	1
18	1	0.2668	0.2668	.	0	.	0
18	2	0.2310	0.2310	.	0	.	1
18	3	0.2353	0.2353	.	1	.	0
18	4	0.2668	0.2668	.	1	.	1

The first-stage FEFI is described as follows:

- Observation unit 1 has no missing value. Therefore, the ImpIndex value is 0; the FracWt value is 1; and the values of X, CX, Y, and CY are the same as the observed values for observation unit 1 in Figure 119.12. Because all observation units have a weight of 1, the fractional weights (FracWt) and the imputation-adjusted weights (ImpWt) are the same for all rows.

- Observation unit 5 has missing values in Y and CY. In the first-stage, only CY is imputed conditional on the observed level of CX. The observed level for CX for observation unit 5 is 0. For CX=0, two levels for CY are observed: CY=0, and CY=1. Therefore, observation unit 5 receives two donor cells (ImpIndex=1 and ImpIndex=2). The fractional weights for these two donor cells are computed by applying FEFI on variables CX and CY. For more information about FEFI, see the section "Example of FEFI" on page 9926. The fractional weights after the first-stage imputation are 0.536 and 0.464. Because CX is observed, CX values in both rows for observation unit 5 are the same as the observed value. However, the first recipient row for observation unit 5 has an imputed CY value of 0, the second recipient row for observation unit 5 has an imputed CY value of 1, and each of these rows has a corresponding fractional weight. Because no imputation is performed for Y in the first-stage, both rows for observation unit 5 contain missing values for Y.

- Observation unit 14 has missing values in X and CX. In the first-stage, only CX is imputed conditional on the observed level of CY. The observed level of CY for unit 14 is 1. For CY=1, two levels for CX are observed: CX=0, and CX=1. Therefore, observation unit 14 receives two donor cells (ImpIndex=1

and ImpIndex=2). The fractional weights for these two donor cells are computed by applying FEFI on variables CX and CY. For more information about FEFI, see the section "Example of FEFI" on page 9926. The fractional weights after the first-stage imputation are 0.464 and 0.536. Because CY is observed, CY values in both rows for unit 14 are the same as the observed value. However, the first recipient row for unit 14 has an imputed CX value of 0, the second recipient row for unit 14 has an imputed CX value of 1, and each of these rows has a corresponding fractional weight. Because no imputation is performed for X in the first-stage, both rows for unit 14 contain missing values for X.

- Observation unit 18 has missing values in all variables X, CX, Y, and CY. Only variables CX and CY are imputed in the first-stage. From the observed data, CX and CY can take the following values (CX=0, CY=0), (CX=0, CY=1), (CX=1, CY=0), and (CX=1, CY=1). The four imputed rows (ImpIndex 1, ImpIndex 2, ImpIndex 3, and ImpIndex 4) for observation unit 18 represent the four observed combinations for CX and CY along with their fractional weights. The fractional weights for these four donor cells are computed by applying FEFI on variables CX and CY. For more information about FEFI, see the section "Example of FEFI" on page 9926.

The second-stage FEFI imputes the missing values in X and Y conditional on the imputed levels for imputation bin variables CX and CY from the first-stage imputation. The imputed data set after the second-stage imputation is displayed in Figure 119.13. Variables X and Y are imputed in the second stage.

Figure 119.13 Two-Stage Fractional Imputation

Unit	ImpIndex	ImpWt	FracWt	X	CX	Y	CY
1	0	1.0000	1.0000	0.3	0	-0.54	0
2	0	1.0000	1.0000	0.2	0	-0.77	0
3	0	1.0000	1.0000	1.7	0	-0.59	0
4	0	1.0000	1.0000	1.7	0	-0.59	0
5	1	0.1340	0.1340	1.0	0	-0.77	0
5	2	0.1340	0.1340	1.0	0	-0.54	0
5	3	0.2680	0.2680	1.0	0	-0.59	0
5	4	0.1547	0.1547	1.0	0	-0.03	1
5	5	0.1547	0.1547	1.0	0	0.78	1
5	6	0.1547	0.1547	1.0	0	0.95	1
6	0	1.0000	1.0000	1.8	0	-0.03	1
7	0	1.0000	1.0000	2.0	0	0.95	1
8	0	1.0000	1.0000	1.9	0	0.78	1
9	0	1.0000	1.0000	6.7	1	-0.15	0
10	0	1.0000	1.0000	6.0	1	-1.01	0
11	0	1.0000	1.0000	3.3	1	-1.86	0
12	0	1.0000	1.0000	7.3	1	-0.21	0
13	0	1.0000	1.0000	6.7	1	0.80	1
14	1	0.1547	0.1547	1.8	0	1.23	1
14	2	0.1547	0.1547	1.9	0	1.23	1
14	3	0.1547	0.1547	2.0	0	1.23	1
14	4	0.1340	0.1340	2.9	1	1.23	1
14	5	0.1340	0.1340	6.7	1	1.23	1
14	6	0.1340	0.1340	9.6	1	1.23	1
14	7	0.1340	0.1340	10.0	1	1.23	1
15	0	1.0000	1.0000	2.9	1	0.65	1
16	0	1.0000	1.0000	9.6	1	0.95	1
17	0	1.0000	1.0000	10.0	1	0.13	1
18	1	0.0667	0.0667	0.2	0	-0.77	0
18	2	0.0667	0.0667	0.3	0	-0.54	0
18	3	0.1334	0.1334	1.7	0	-0.59	0
18	4	0.0770	0.0770	1.8	0	-0.03	1
18	5	0.0770	0.0770	1.9	0	0.78	1
18	6	0.0770	0.0770	2.0	0	0.95	1
18	7	0.0588	0.0588	3.3	1	-1.86	0
18	8	0.0588	0.0588	6.0	1	-1.01	0
18	9	0.0588	0.0588	6.7	1	-0.15	0
18	10	0.0588	0.0588	7.3	1	-0.21	0
18	11	0.0667	0.0667	2.9	1	0.65	1
18	12	0.0667	0.0667	6.7	1	0.80	1
18	13	0.0667	0.0667	9.6	1	0.95	1
18	14	0.0667	0.0667	10.0	1	0.13	1

The second-stage FEFI is described as follows:

- Observation unit 1 has no missing value. Therefore, the ImpIndex value is 0; the FracWt value is 1; and the values of X, CX, Y, and CY are the same as the observed values for observation unit 1 in

Figure 119.13. Because all observation units have a weight of 1, the fractional weights (FracWt) and the imputation-adjusted weights (ImpWt) are the same for all rows.

- Observation unit 5 has missing values in Y and CY. The variable CY has two imputed levels (0 and 1) from the first-stage imputation. The observed level of CX for observation unit 5 is 0.

  The row that contains Unit 5 and ImpIndex=1 in Figure 119.12 has CX=0 and CY=0. Units 1, 2, 3, and 4 in the complete data have CX=0 and CY=0. These units are possible donors for a missing Y when CX=0 and CY=0. The four donor units have three unique values for Y: –0.54, –0.59, and –0.77. These three unique values define three donor cells to impute Y when CX=0 and CY=0. The missing value in Y for CX=0 and CY=0 is replaced by all three possible observed values. Because the weight for the donor cell that is defined by Y=–0.59 is double the weight for the other two donor cells, the imputed row that contains Y = –0.59 is assigned a second-stage fractional weight of 1/2 and the other two rows are each assigned a second-stage fractional weight of 1/4. The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.54) for CX=0, CY=0, and ImpIndex=1 to obtain the two-stage FEFI weight.

  The row that contains Unit 5 and ImpIndex=2 in Figure 119.12 has CX=0 and CY=1. Units 6, 7, and 8 in the complete data have CX=0 and CY=1. These units are donors for missing Y when CX=0 and CY=1. The three donor units have three unique values for Y: –0.03, 0.95, and 0.78. These three unique values define three donor cells for imputing Y when CX=0 and CY=1. The missing value in Y for CX=0 and CY=1 is replaced by all three possible observed values. Because all three donor cells have equal weights, all three imputed rows are assigned a second-stage fractional weight of 1/3. The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.46) for CX=0, CY=1, and ImpIndex=2 to obtain the two-stage FEFI weight.

  Therefore, after two-stage FEFI, missing values in Y are replaced by six imputed values that have fractional weights proportional to the observed weighted frequencies of the second-stage donor cells conditional on the first-stage FEFI.

- Observation unit 14 has missing values in X and CX. The variable CX has two imputed levels (0 and 1) from the first-stage imputation. The observed level for CY for unit 14 is 1.

  The row that contains Unit 14 and ImpIndex=1 in Figure 119.12 has CX=0 and CY=1. Units 6, 7, and 8 in the complete data have CX=0 and CY=1. These units are possible donors for missing X when CX=0 and CY=1. The three donor units have three unique values for X: 1.8, 1.9, and 2.0. These three unique values define three donor cells for imputing X when CX=0 and CY=1. The missing value in X for CX=0 and CY=1 is replaced by all three possible observed values. Because all three donor cells have equal weights, all three imputed rows are assigned a second-stage fractional weight of 1/3. The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.46) for CX=0, CY=1, and ImpIndex=1 to obtain the two-stage FEFI weight.

  The row that contains Unit 14 and ImpIndex=2 in Figure 119.12 has CX=1 and CY=1. Units 13, 15, 16 and 17 in the complete data have CX=1 and CY=1. These units are donors for missing X when CX=1 and CY=1. The four donor units have four unique values for X: 2.9, 6.7, 9.6, and 10.0. These four unique values define four donor cells for imputing X when CX=1 and CY=1. The missing value in X for CX=1 and CY=1 is replaced by all four possible observed values. Because all four donor cells have equal weights, all four imputed rows are assigned a second-stage fractional weight of 1/4. The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.54) for CX=1 CX=1 to obtain the two-stage FEFI weight.

Therefore, after two-stage FEFI, missing values in X are replaced by seven imputed values that have fractional weights proportional to the observed weighted frequencies of the second-stage donor cells conditional on the first-stage FEFI.

- Observation unit 18 has missing values in all four variables X, CX, Y, and CY. The variable CX has two imputed levels (0 and 1), and the variable CY has two imputed levels (0 and 1) from the first-stage imputation.

  The row that contains Unit 18 and ImpIndex=1 in Figure 119.12 has CX=0 and CY=0. Units 1, 2, 3, and 4 in the complete data have CX=0 and CY=0. These units are possible donors for missing X and Y when CX=0 and CY=0. The four donor units have three unique values for (X, Y): (0.3, –0.54), (0.2, –0.77), and (1.7, –0.59). These three unique values define three donor cells for imputing (X, Y) when CX=0 and CY=0. The missing value in (X, Y) for CX=0 and CY=0 is replaced by all three values. Because the weight for the donor cell that is defined by (X, Y) = (1.7, –0.59) is double the weight for the other two donor cells, the imputed row that contains X = 1.7 and Y = –0.59 is assigned a second-stage fractional weight of 1/2 and the other two rows each are assigned a second-stage fractional weight of 1/4. The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.27) for CX=0, CY=0, and ImpIndex=1 to obtain the two-stage FEFI weight.

  The row that contains Unit 18 and ImpIndex=2 in Figure 119.12 has CX=0 and CY=1. Units 6, 7, and 8 in the complete data have CX=0 and CY=1. These units are donors for the missing (X, Y) when CX=0 and CY=1. The three donor units have three unique values for (X, Y): (1.8, -0.3), (1.9, 0.78), and (2.0, 0.95). These three unique values define three donor cells for imputing (X, Y) when CX=0 and CY=1. The missing value in (X, Y) for CX=0 and CY=1 is replaced by all three possible observed values. Because all three donor cells have equal weights, all three imputed rows are assigned a second-stage fractional weight of 1/3. The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.23) for CX=0 and CX=1 to obtain the two-stage FEFI weight.

  Missing values in (X, Y) in rows that contain Unit 18, ImpIndex=3, and ImpIndex=4 in Figure 119.12 are imputed similarly.

  Thus, after two-stage FEFI, missing values in (X, Y) are replaced by 14 imputed values that have fractional weights proportional to the observed weighted frequencies of the second-stage donor cells conditional on the first-stage FEFI.

The resulting data set has 42 rows. Fifteen rows for fully observed units (ImpIndex = 0), six rows for unit 5, seven rows for unit 14, and fourteen rows for unit 18. The sum of the fractional weights is 1 for all units. The imputation-adjusted replicate weights are computed by applying the first-stage and the second-stage imputation independently in each replicate sample as discussed in the previous list. The imputed data set along with first four imputation-adjusted replicate weights is displayed in Figure 119.14.

Figure 119.14 Two-Stage Fractional Imputation with the Imputation-Adjusted Replicate Weights

Unit	ImpIndex	ImpWt	FracWt	X	CX	Y	CY	ImpRepWt_1	ImpRepWt_2	ImpRepWt_3	ImpRepWt_4
1	0	1.0000	1.0000	0.3	0	-0.54	0	0	1.0588	1.0588	1.0588
2	0	1.0000	1.0000	0.2	0	-0.77	0	1.0588	0	1.0588	1.0588
3	0	1.0000	1.0000	1.7	0	-0.59	0	1.0588	1.0588	0	1.0588
4	0	1.0000	1.0000	1.7	0	-0.59	0	1.0588	1.0588	1.0588	0
5	1	0.1340	0.1340	1.0	0	-0.77	0	0.1637	0	0.1637	0.1637
5	2	0.1340	0.1340	1.0	0	-0.54	0	0	0.1637	0.1637	0.1637
5	3	0.2680	0.2680	1.0	0	-0.59	0	0.3274	0.3274	0.1637	0.1637
5	4	0.1547	0.1547	1.0	0	-0.03	1	0.1893	0.1893	0.1893	0.1893
5	5	0.1547	0.1547	1.0	0	0.78	1	0.1893	0.1893	0.1893	0.1893
5	6	0.1547	0.1547	1.0	0	0.95	1	0.1893	0.1893	0.1893	0.1893
6	0	1.0000	1.0000	1.8	0	-0.03	1	1.0588	1.0588	1.0588	1.0588
7	0	1.0000	1.0000	2.0	0	0.95	1	1.0588	1.0588	1.0588	1.0588
8	0	1.0000	1.0000	1.9	0	0.78	1	1.0588	1.0588	1.0588	1.0588
9	0	1.0000	1.0000	6.7	1	-0.15	0	1.0588	1.0588	1.0588	1.0588
10	0	1.0000	1.0000	6.0	1	-1.01	0	1.0588	1.0588	1.0588	1.0588
11	0	1.0000	1.0000	3.3	1	-1.86	0	1.0588	1.0588	1.0588	1.0588
12	0	1.0000	1.0000	7.3	1	-0.21	0	1.0588	1.0588	1.0588	1.0588
13	0	1.0000	1.0000	6.7	1	0.80	1	1.0588	1.0588	1.0588	1.0588
14	1	0.1547	0.1547	1.8	0	1.23	1	0.1656	0.1656	0.1656	0.1656
14	2	0.1547	0.1547	1.9	0	1.23	1	0.1656	0.1656	0.1656	0.1656
14	3	0.1547	0.1547	2.0	0	1.23	1	0.1656	0.1656	0.1656	0.1656
14	4	0.1340	0.1340	2.9	1	1.23	1	0.1405	0.1405	0.1405	0.1405
14	5	0.1340	0.1340	6.7	1	1.23	1	0.1405	0.1405	0.1405	0.1405
14	6	0.1340	0.1340	9.6	1	1.23	1	0.1405	0.1405	0.1405	0.1405
14	7	0.1340	0.1340	10.0	1	1.23	1	0.1405	0.1405	0.1405	0.1405
15	0	1.0000	1.0000	2.9	1	0.65	1	1.0588	1.0588	1.0588	1.0588
16	0	1.0000	1.0000	9.6	1	0.95	1	1.0588	1.0588	1.0588	1.0588
17	0	1.0000	1.0000	10.0	1	0.13	1	1.0588	1.0588	1.0588	1.0588
18	1	0.0667	0.0667	0.2	0	-0.77	0	0.0764	0	0.0764	0.0764
18	2	0.0667	0.0667	0.3	0	-0.54	0	0	0.0764	0.0764	0.0764
18	3	0.1334	0.1334	1.7	0	-0.59	0	0.1528	0.1528	0.0764	0.0764
18	4	0.0770	0.0770	1.8	0	-0.03	1	0.0884	0.0884	0.0884	0.0884
18	5	0.0770	0.0770	1.9	0	0.78	1	0.0884	0.0884	0.0884	0.0884
18	6	0.0770	0.0770	2.0	0	0.95	1	0.0884	0.0884	0.0884	0.0884
18	7	0.0588	0.0588	3.3	1	-1.86	0	0.0662	0.0662	0.0662	0.0662
18	8	0.0588	0.0588	6.0	1	-1.01	0	0.0662	0.0662	0.0662	0.0662
18	9	0.0588	0.0588	6.7	1	-0.15	0	0.0662	0.0662	0.0662	0.0662
18	10	0.0588	0.0588	7.3	1	-0.21	0	0.0662	0.0662	0.0662	0.0662
18	11	0.0667	0.0667	2.9	1	0.65	1	0.0750	0.0750	0.0750	0.0750
18	12	0.0667	0.0667	6.7	1	0.80	1	0.0750	0.0750	0.0750	0.0750
18	13	0.0667	0.0667	9.6	1	0.95	1	0.0750	0.0750	0.0750	0.0750
18	14	0.0667	0.0667	10.0	1	0.13	1	0.0750	0.0750	0.0750	0.0750

Fractional Hot-Deck Imputation

The fractional hot-deck imputation (FHDI) method uses multiple donor units for a recipient unit. Each donor donates a fraction of the original weight of the recipient unit such that the sum of the fractional weights from all the donors is equal to the original weight of the recipient. The fraction of the recipient weight that a donor unit contributes to the recipient unit is known as the *fractional weight*. The donors are selected by using probability proportional to size (PPS) selection in which the two-stage FEFI weights are used as the size measure.

FHDI is useful for reducing the size of the imputed data when two-stage FEFI creates many imputed rows. FHDI follows the same imputation steps as those of two-stage FEFI, but FHDI selects a subset of second-stage donor cells from all possible second-stage donor cells for the imputation.

Similar to two-stage FEFI, variables that have many unique observed levels are grouped into imputation bins. The first imputation stage is performed for all categorical variables by using the FEFI method. The categorical variables include the character variables, the CLASS variables that you also specify in the VAR statement, and the variables that contain the imputation bins of the continuous variables.

The second imputation stage is performed for the continuous variables within each first-stage donor cell. Observations that contain missing values for any of the continuous items are considered to be the recipients, and observations that contain observed values for all items are the donors. The second-stage donor cells are defined by the unique vectors of the observed values for the continuous variables within the first-stage donor cells.

If you specify NDONORS=m_2, then m_2 second-stage donor cells are selected within each first-stage donor cell, provided that more than m_2 second-stage donor cells are available for the continuous variables in that donor cell. Second-stage donor cells are selected within each first-stage donor cell by using PPS selection with replacement, where the fractional weights of the second-stage donor cells are used as the size measures. If the number of second-stage donor cells in a first-stage donor cell is less than or equal to m_2, then all available second-stage donor cells are used to impute the missing values for the continuous variables in that first-stage donor cell.

If no second-stage selection is performed, then the second-stage fractional weights are the same as the fractional weights from two-stage FEFI. If a second-stage selection is performed, then the second-stage fractional weights are computed by multiplying the first-stage fractional weights by the number of times a second-stage donor cell is selected divided by the second-stage sample size.

Imputation-adjusted replicate weights are computed by repeating both the first-stage and second-stage imputation in every replicate sample independently.

Selection of donor cells is not repeated in the creation of replicate samples. The donor cells that are selected in the full sample are retained in all replicate samples, and the replicate weights are adjusted to compensate for the selection of donor cells in both stages. If no selection is performed in either stage in the full sample (FEFI is used in both stages), then the replicate weights are not adjusted further.

The method is similar to Im, Kim, and Fuller (2015).

For more information about replication-weight adjustment, see the section "Replicate Weight Adjustments for FHDI" on page 9959.

Fractional Hot-Deck Imputation Notations and Assumptions

Suppose you want to impute P items jointly. Let $\mathbf{X}_i = (X_{i1}, \ldots, X_{iP_1})$ be the response for P_1 items in unit i, and let $\mathbf{Y}_i = (Y_{i1}, \ldots, Y_{iP_2})$ be the response for P_2 items in unit i, where $P = P_1 + P_2$. Let X_{ij} be categorical with J_j levels for item j, and let Y_{ij} be continuous. Further assume that $\tilde{\mathbf{Y}} = (\tilde{Y}_{i1}, \ldots, \tilde{Y}_{iP_2})$ contains the discretized levels (imputation bins) for $\mathbf{Y}$, where $\tilde{Y}_{ij}$ has J_j levels. Define $Z_{ij} = (X_{ij}, \tilde{Y}_{ij})$. Then Z_{ij} is categorical and has J_j levels for item j. Denote $\mathbf{Z}_{i,\text{obs}}$ as the observed part and $\mathbf{Z}_{i,\text{miss}}$ as the missing part of $\mathbf{Z}_i$.

Let $\pi(\kappa_1\kappa_2\cdots\kappa_P)$ be the population proportion that falls in category $Z_1 = \kappa_1, Z_2 = \kappa_2, \cdots Z_P = \kappa_P$. FEFI computes the fractional weights by using conditional probabilities of observing an imputed value in the data in which the observed levels for the nonmissing items in the recipient unit are equal to the observed levels for the same items in the donor units. For example, consider observation unit i in which items Z_{i1} and Z_{i2} are missing and items $Z_{i3} = \kappa_3$, $Z_{i4} = \kappa_4$, ..., and $Z_{iP} = \kappa_P$ are observed. The initial fractional weight for imputing (Z_{i1}, Z_{i2}) with (κ_1, κ_2) is computed by using the estimated conditional probability of observing $(Z_{i1}, Z_{i2}) = (\kappa_1, \kappa_2)$ when $(Z_{i3}, Z_{i4}, \ldots, Z_{iP}) = (\kappa_3, \kappa_4, \ldots, \kappa_P)$ in the complete data. The initial conditional probabilities are estimated by

$$\hat{P}(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2 | Z_{i3} = \kappa_3, \ldots, Z_{iP} = \kappa_p) = \left\{ \sum_{\kappa_1=1}^{J_1} \sum_{\kappa_2=1}^{J_2} \hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P) \right\}^{-1} \hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P)$$

where

$$\hat{\pi}(\kappa_1\kappa_2\cdots\kappa_P) = \left\{ \sum_{i\in\mathcal{A}_c} w_i \right\}^{-1} \sum_{i\in\mathcal{A}_c} w_i I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

is the estimated joint probability, $I(.)$ is an indicator function, w_i is the sampling weight for observation unit i, and $\mathcal{A}_c$ is the set of indices for observation units without any missing items.

The first-stage FEFI replaces the missing items in $\mathbf{Z}_i = (\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{mis}})$ by using observed values from all donor cells for observation unit i. The l_1th imputed value $\mathbf{Z}_{i,\text{dcell}[l_1]} = (\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{mis}[l_1]})$ uses the observed values from the l_1th donor cell (realization), where l_1 ranges from 1 to M_{l_1}.

The second-stage FEFI imputes the missing values in the continuous variables Y_{ij} where $j = 1, 2, \ldots, P_2$. The second-stage imputation is conditional on the imputed levels from the first-stage imputation. For observation unit i and first-stage donor cell l_1, the number of second-stage donor cells is equal to the number of unique combinations of the observed levels for the missing items in $\mathbf{Y}_i = (\mathbf{Y}_{i,\text{obs}}, \mathbf{Y}_{i,\text{mis}})$ among the responding units in the first-stage donor cell l_1. Unlike the first-stage imputation, the second-stage imputation does not depend on the observed levels for the nonmissing items in $\mathbf{Y}_i = (\mathbf{Y}_{i,\text{obs}}, \mathbf{Y}_{i,\text{mis}})$.

The second-stage fractional weights are computed by using the estimated probability of observing an imputed value for continuous variables conditional on the first-stage imputed level. For example, consider observation unit i in which items Y_{i1} and Y_{i2} are missing. Assume that the l_1th first-stage donor cell containing observation units that have $Z_1 = \kappa_1$, $Z_2 = \kappa_2$, ..., and $Z_P = \kappa_P$ provides the l_1th imputed value for the first-stage imputation for observation unit i. The second-stage fractional weight for imputing (Y_{i1}, Y_{i2}) with $(\kappa_{21}, \kappa_{22})$ in the first-stage donor cell l_1 is computed by using the estimated conditional probability of observing $(Y_{i1}, Y_{i2}) = (\kappa_{21}, \kappa_{22})$ by using the nonmissing units in the l_1th first-stage donor cell.

The conditional probabilities are estimated by

$$\hat{P}(Y_{i1} = \kappa_{21}, Y_{i2} = \kappa_{22} | Z_1 = \kappa_1, \ldots, Z_P = \kappa_P) = \hat{\pi}(\kappa_{21}\kappa_{22}|l_1) = \left\{\sum_{i \in \mathcal{A}_c} w_i I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_p)\right\}^{-1} \sum_{i \in \mathcal{A}_c} w_i I(Y_{i1} = \kappa_{21}, Y_{i2} = \kappa_{22}, Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_p)$$

The second-stage FEFI replaces the missing values in $\mathbf{Y}_i = (\mathbf{Y}_{i,\text{obs}}, \mathbf{Y}_{i,\text{miss}})$ when $(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$ by using all observed values for $\mathbf{Y}_{i,\text{miss}}$ from observation rows that contain $(Z_1 = \kappa_1, \ldots, Z_P = \kappa_P)$. Each second-stage imputed value (configuration) defines a second-stage donor cell. The l_2th imputed value in the l_1th first-stage donor cell is $\mathbf{Y}_{i,\text{dcell}[l_2|l_1]} = (\mathbf{Y}_{i,\text{obs}}, \mathbf{Y}_{i,\text{miss}[l_2]}, Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$, where l_2 ranges from 1 to $M_{il_2|l_1}$, and the l_1th first-stage donor cell is defined by observation units that contain $Z_1 = \kappa_1, \ldots, Z_P = \kappa_P$.

The two-stage imputation-adjusted fractional weights are computed by multiplying the first-stage fractional weights by the second-stage fractional weights. The sampling weight for observation unit i is denoted by w_i. Let w_{il} be the first-stage fractional weight for donor cell l, and let $w_{il(t)}$ be the first-stage fractional weight for donor cell l at the tth EM iteration. Let $w_{il_2|l_1}$ be the second-stage fractional weight, and let $w_{il_1l_2}$ be the two-stage fractional weight for first-stage donor cell l_1 and second-stage donor cell l_2. Thus, the imputation-adjusted weight for observation unit i, first-stage donor cell l_1, and second-stage donor cell l_2 is $w_i w_{il_1l_2}$.

Replicate weights are adjusted for imputation by applying both the first-stage and the second-stage imputation in every replicate sample. Let $w_i^{(r)}$ be the unadjusted replicate weight for observation unit i in replicate sample r. The unadjusted replicate weights are created by using a replication method such as the bootstrap, BRR, or delete-1 jackknife. For more information about how the unadjusted replicate weights are created, see sections "Unadjusted Bootstrap Replicate Weights" on page 9955, "Unadjusted BRR Replicate Weights" on page 9956, and "Unadjusted Jackknife Replicate Weights" on page 9958. Let $w_{il}^{(r)}$ be the first-stage fractional replicate weight, $w_{il_2|l_1}^{(r)}$ be the second-stage fractional replicate weight, and $w_{il_1l_2}^{(r)}$ be the two-stage fractional replicate weight for replicate sample r, observation unit i, first-stage donor cell l_1, and second-stage donor cell l_2. Thus, the imputation-adjusted replicate weight for observation unit i, first-stage donor cell l_1, second-stage donor cell l_2, and replicate sample r is $w_i^{(r)} w_{il_1l_2}^{(r)}$.

The following assumptions are necessary:

- The conditional probability of observing an imputed value in the missing data is the same as the conditional probability of observing the value in the observed data in each imputation cell. For example, the conditional probability, $P(Z_{i1} = \kappa_1, Z_{i2} = \kappa_2 | Z_{i3} = \kappa_3, \ldots, Z_{iP} = \kappa_P)$, is the same for the observed data as it is for the data where Z_1 and Z_2 are missing.
- The conditional probability of observing a second-stage imputed value in the missing data is the same as the conditional probability of observing the value in the observed data in every first-stage donor cell. For example, the conditional probability in the first-stage donor cell $(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$, $P(Y_{i1} = \kappa_{21}, Y_{i2} = \kappa_{22} | Z_1 = \kappa_1, \ldots, Z_P = \kappa_p)$, is the same for the observed data as it is for the data in which (Y_1 and Y_2) are missing and $Z_1 = \kappa_1, \ldots, Z_P = \kappa_P$.
- For every observation unit that contains missing items, at least one realization for the missing items is available in the complete data; otherwise the observation is not imputed.
- For variance estimation, at least two realizations for the missing items from two different PSUs must be available in the complete data for every observation unit that contains missing items. If the condition is not satisfied, the variance due to imputing the missing items in that observation is ignored.

Fractional Hot-Deck Imputation Algorithm

The FHDI method first computes the fully efficient fractional weights by using an EM-by-weighting algorithm like that of Kim and Fuller (2013) to impute the missing values in $\mathbf{Z}_i$. The missing values in $\mathbf{Y}_i$ are imputed in the second-stage imputation. For the second-stage imputation, FEFI weights to impute $\mathbf{Y}_i$ are computed independently in every imputed level of $\mathbf{Z}_{i,\text{miss}[l]}$, where $l = 1, \ldots, M_{l_1}$ is the number of first-stage donor cells. If the number of second-stage donor cells is greater than m_2 (where m_2 is the value of the NDONORS= option in the PROC SURVEYIMPUTE statement), then m_2 second-stage donor cells are selected by using PPS sampling in which the second-stage fractional weights are used as the size measure.

The following steps describe the FHDI technique. If you do not use the CELL statement to specify imputation cells, PROC SURVEYIMPUTE uses the entire data set as one imputation cell. If you specify imputation cells, then all the probabilities are computed by using observations from the same imputation cell as the recipient unit. To simplify notation, subscripts are not used for imputation cells in the following description. Imputation cells are defined for the first-stage imputation. Steps 1 to 5 describe the two-stage FEFI. Step 6 describes donor selection for FHDI.

1. *Initialization*: For each observation that has missing items, determine the number of first-stage donor cells. The first-stage donor cells are determined by using the number of unique combinations of observed levels in $\mathbf{Z}_i$ for imputing the missing items in $\mathbf{Z}_i$. Only the responding units in the imputation cell are used to determine the number of first-stage donor cells. Compute the initial fractional weight from donor cell l to unit i, $w_{il(0)}$:

$$w_{il(0)} = \left\{ \sum_{k=1}^{M_{il_1}} \tilde{\pi}_{(0)}(\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{miss}[k]}) \right\}^{-1} \tilde{\pi}_{(0)}(\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{miss}[l]})$$

   where $l = 1, 2, \ldots, M_{l_1}$ is the number of first-stage donor cells and

$$\tilde{\pi}_{(0)}(\kappa_1, \ldots, \kappa_P) = \left\{ \sum_{i \in \mathcal{A}_c} w_i \right\}^{-1} \sum_{i \in \mathcal{A}_c} w_i I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

   The sum of the fractional weights over all the donor cells is 1 for every observation unit; that is, $\sum_l w_{il(0)} = 1$ for all i. The lth imputed row for unit i is created by keeping the observed items unchanged, replacing the missing items with the observed levels from the lth donor cell, and computing the fractional weight by $w_i w_{il(0)}$. Only the complete observations (observations that have no missing items) are used to compute the fractional weights in this step. If unit i has no missing items, then $w_{i1(0)} = 1$. The initial FEFI data set contains all the observed units, the imputed rows for observations that have missing items, and the corresponding fractional weights.

2. *M-step*: The tth maximization step (M-step) computes the joint probabilities by using the fractional weights from the (t–1)th expectation-step,

$$\tilde{\pi}_{(t)}(\kappa_1, \ldots, \kappa_P) = \left\{ \sum_i \sum_l w_i w_{il(t-1)} \right\}^{-1} \sum_i \sum_l w_i w_{il(t-1)} I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

   for all i, all l, and $t > 0$. Note that for $t > 0$, $\tilde{\pi}_{(t)}$ uses all observation units, including observations that have missing items and are imputed in the initialization step.

3. *E-step*: The tth expectation step (E-step) computes the fractional weights by using the joint probabilities $\tilde{\pi}_{(t)}(\kappa_1, \ldots, \kappa_P)$ from the tth M-step. The tth fractional weight for unit i and donor cell l is given by

$$w_{il(t)} = \left\{ \sum_{k=1}^{M_{il_1}} \tilde{\pi}_{(t)}(\mathbf{Z}_{i,\mathrm{obs}}, \mathbf{Z}_{i,\mathrm{miss}[k]}) \right\}^{-1} \tilde{\pi}_{(t)}(\mathbf{Z}_{i,\mathrm{obs}}, \mathbf{Z}_{i,\mathrm{miss}[l]})$$

4. *Repetition*: The expectation maximization steps (EM-steps, step 2 and 3) are repeated for $t = 1, 2, \ldots,$ until the changes in fractional weights over all observation units between two successive EM-steps are negligible or the maximum number of EM repetitions is reached.

   The maximum absolute difference convergence criterion, ϵ_{AD}, at step t is defined as

$$\max_{i,l} |w_{il(t)} - w_{il(t-1)}| \leq \epsilon_{\mathrm{AD}}$$

   The maximum absolute relative difference convergence criterion, ϵ_{RD}, at step t is defined as

$$\max_{i,l} |w_{il(t)} - w_{il(t-1)}| / w_{il(t-1)} \leq \epsilon_{\mathrm{RD}}$$

   where $w_{il(t-1)} > 0$.

5. *Second-stage imputation*: The second-stage imputation replaces the missing values in the continuous variables by using the observed values within each selected first-stage donor cell. This step is similar to step 1 but is applied in order to impute the continuous variables.

   For a particular observation unit i, let $\mathbf{Z}_{i,\mathrm{dcell}[l_1]} = (\mathbf{Z}_{i,\mathrm{obs}}, \mathbf{Z}_{i,\mathrm{mis}[l_1]})$ be the l_1th donor cell from the first-stage imputation, where l_1 ranges from 1 to M_{l_1}. For each observation unit, i, the possible number of second-stage donor cells is equal to the number of unique combinations of the observed levels for the missing items in $\mathbf{Y}_i$ from the responding units in the first-stage donor cell l_1.

   Let $\pi_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1} \cdots \kappa_{P_2|l_1})$ be the population proportion that falls in category $\kappa_{1|l_1}\kappa_{2|l_1} \cdots \kappa_{P_2|l_1}$. Assume that it is possible to estimate the population categories from the observed sample. For example, the conditional probability, $P(Y_{i1|l_1} = \kappa_{1|l_1}, Y_{i2|l_1} = \kappa_{2|l_1} | Y_{i3|l_1} = \kappa_{3|l_1}, \ldots, Y_{iP_2|l_1} = \kappa_{P_2|l_1})$, is the same for the observed data as it is for the data in which $(Y_{i1|l_1} = \kappa_{1|l_1}, Y_{i2|l_1} = \kappa_{2|l_1})$ are missing. The conditional probabilities are estimated by

$$\hat{P}(Y_{i1|l_1} = \kappa_{1|l_1}, Y_{i2|l_1} = \kappa_{2|l_1} | Y_{i3|l_1} = \kappa_{3|l_1}, \ldots, Y_{iP_2|l_1} = \kappa_{P_2|l_1}) = \left\{ \sum_{\kappa_{1|l_1}\kappa_{2|l_1}} \hat{\pi}_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1} \cdots \kappa_{P_2|l_1}) \right\}^{-1} \hat{\pi}_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1} \cdots \kappa_{P_2|l_1})$$

   where

$$\hat{\pi}_{2|l_1}(\kappa_{1|l_1}\kappa_{2|l_1} \cdots \kappa_{P_2|l_1}) = \left\{ \sum_{i \in \mathbf{Z}_{i,\mathrm{dcell}[l_1]}} w_i \right\}^{-1} \sum_{i \in \mathbf{Z}_{i,\mathrm{dcell}[l_1]}} w_i I(Y_{i1|l_1} = \kappa_{1|l_1}, \ldots, Y_{iP_2|l_1} = \kappa_{P_2|l_1})$$

   is the estimated joint probability, $I(.)$ is an indicator function, and w_i is the observation weight for unit i.

   Let $l = 1, 2, \ldots, M_{il_2|l_1}$ be all the observed combinations of $\mathbf{Y}_{k:k \in \mathbf{Z}_{i,\mathrm{dcell}[l_1]}, k \neq i,\mathrm{miss}}$ in the sample. Let $\mathbf{Y}_{i,\mathrm{miss}[l]}$ be the lth realization of $\mathbf{Y}_{i,\mathrm{miss}}$ in the sample. You must assume that at least one realization is available; otherwise, missing values in the continuous items for the observation are not imputed.

Compute the second-stage fractional weight from the second-stage donor cell l_2 conditional on the first-stage donor cell l_1 for unit i, $w_{il_2|l_1}$, by

$$w_{il_2|l_1} = \left\{ \sum_{k=1}^{M_{il_2|l_1}} \tilde{\pi}_{2|l_1}(\mathbf{Y}_{i,\text{obs}}, \mathbf{Y}_{i,\text{miss}[k]}) \right\}^{-1} \tilde{\pi}_{2|l_1}(\mathbf{Y}_{i,\text{obs}}, \mathbf{Y}_{i,\text{miss}[l_2]})$$

where $l = 1, 2, \ldots, M_{il_2|l_1}$ is the number of second-stage donor cells and

$$\tilde{\pi}_{2|l_1}(\kappa_1, \ldots, \kappa_{P_2}) = \left\{ \sum_{i \in \mathbf{Z}_{i,\text{dcell}[l_1]}} w_i \right\}^{-1} \sum_{i \in \mathbf{Z}_{i,\text{dcell}[l_1]}} w_i I(Y_{i1|l_1} = \kappa_1, \ldots, Y_{iP_2|l_1} = \kappa_{P_2})$$

The sum of the second-stage fractional weights over all second-stage donor cells is 1 for every observation unit; that is, $\sum_{l_2} w_{il_2|l_1} = 1$ for all l_1 and i. The l_2th second-stage imputed row in the l_1th first-stage imputed row for unit i is created by keeping the observed items unchanged, replacing the missing items in $\mathbf{Y}_i$ with the observed values from the l_2th second-stage donor cell, and computing the two-stage fractional weight by $w_{il_1l_2} = w_{il_1} w_{il_2|l_1}$, where w_{il_1} is the first-stage fractional weight for the first-stage donor cell l_1. The maximum number of donor cells for unit i is $M_{il_1} M_{il_2|l_1}$. Only the complete observations are used to compute the second-stage fractional weights.

6. *Second-stage selection*: Because all observed levels are used as the imputed values for missing items in the continuous variables in the previous step, the number of second-stage donor cells, $M_{il_2|l_1}$, is usually large. Suppose you want to use only m_2 second-stage donor cells (where m_2 is the value of the NDONORS= option in the PROC SURVEYIMPUTE statement). The second-stage selection chooses m_2 second-stage donor cells for every first-stage donor cell.

   The selection step selects a random sample of second-stage donor cells of size m_2 for every recipient unit and every first-stage donor cell in which the number of second-stage donor cells is greater than m_2. Consider the set of all observations in which the values of the observed items are the same and they all have the same missing items. The index set $p_i = \{j : j \in \mathcal{A}, \mathbf{Z}_{j,\text{obs}} = \mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{j,\text{miss}} = \mathbf{Z}_{i,\text{miss}}\}$ contains the indices for all units that have the same missing pattern and the same observed values in the nonmissing items. Let n_{pi} be the number of units in p_i. Note that the number of donor cells, $M_{il_2|l_1}$, and the fractional weights for all n_{pi} recipient units indexed by p_i are the same. If $M_{il_2|l_1} > m_2$, then select a PPS sample with replacement of $m_2 n_{pi}$ donor cells from $M_{il_2|l_1} n_{pi}$ donor cells by using the two-stage fractional weights of the donor cells as the size measure.

   The donor selection algorithm can be described as follows:

   a) Sort the second-stage donor cells by observed values of $\mathbf{Y}_i$ within each first-stage donor cell.
   b) Compute the cumulative sums of the fractional weights for the second-stage donor cells, and normalize the cumulative sums to 1.
   c) Select a random number between 0 and 1, and divide it by the sample size, $m_2 n_{pi}$.
   d) Select a PPS sample with replacement that starts from the scaled random number and uses $1/m_2 n_{pi}$ as the step length. For more information, see Särndal, Swensson, and Wretman (1992, p. 97).
   e) Use a random permutation to sort recipient units in a random order.

f) Distribute the selected donor cells to the recipient units in a cyclic manner, such that the first selected donor cell is allocated to the first recipient unit, the second selected donor cell is allocated to the second recipient unit, the $(n_{pi} + 1)$th selected donor cell is allocated to the first recipient unit, the $(n_{pi} + 2)$th selected donor cell is allocated to the second recipient unit, and so on. Because of the random sorting in the previous step, the first recipient unit after the sorting is not the same as the first recipient unit in the input data.

The second-stage fractional weights, $w_{il_2|l_1}$, are used as the size measure in the PPS sampling. Only one PPS sampling is performed for all recipient units in one first-stage donor cell.

Alternatively, if you specify METHOD=FHDI(SELECTION=PPSPEROBS), then independent selection is performed for every recipient unit and every first-stage donor cell in which the number of second-stage donor cells is greater than m_2. Thus the procedure selects n_{p_i} independent PPS samples with replacement of size m_2 for n_{pi} recipient units.

The imputation-adjusted weights for the second-stage donor cells are equal to w_{il_2} for observation units when no second-stage selection is performed and equal to $w_{il}h_{l_2}/m_2$ for observation units when the second-stage selection is performed, where h_{l_2} is the number of times the second-stage donor cell l_2 is selected. Combining the first-stage and the second-stage imputation, the total number of donor cells for an observation unit is greater than or equal to $M_{l_1}M_{l_2}$ and less than $M_{l_1}m_2$.

7. *Replicate weight adjustments*: The unadjusted replicate weights are created as if all rows of data were observed data by using a replication procedure such as the delete-1 jackknife, BRR, or bootstrap. The unadjusted replicate weights for rows that contain imputed data are then adjusted for two-stage FEFI by replicating all steps for two-stage FEFI for each replicate sample.

   Replication weights for FHDI rows are further adjusted to account for the selection of second-stage donor cells. Although every step in the two-stage FEFI procedure is applied in each replicate sample, the selection of second-stage donor cells is not repeated in FHDI. The fractional weights are adjusted instead.

   For available adjustments in the replicate weights, see the section "Replicate Weight Adjustments for FHDI" on page 9959.

Example of Two-Stage Fractional Hot-Deck Imputation

The small data set shown in Figure 119.15 is used to illustrate the FHDI technique. The data set contains 18 observation units, and each unit has four items (X, CX, Y, and CY). The variable Unit contains the observation identification. Variables CX and CY contains the imputation bins for variables X and Y, respectively. In this example, X and CX are missing for units 14 and 18, and Y and CY are missing for units 5 and 18.

Figure 119.15 Sample Data with Missing Items

Unit	X	CX	Y	CY
1	0.3	0	-0.54	0
2	0.2	0	-0.77	0
3	1.7	0	-0.59	0
4	1.7	0	-0.59	0
5	1.0	0	.	.
6	1.8	0	-0.03	1
7	2.0	0	0.95	1
8	1.9	0	0.78	1
9	6.7	1	-0.15	0
10	6.0	1	-1.01	0
11	3.3	1	-1.86	0
12	7.3	1	-0.21	0
13	6.7	1	0.80	1
14	.	.	1.23	1
15	2.9	1	0.65	1
16	9.6	1	0.95	1
17	10.0	1	0.13	1
18	.	.	.	.

The following statements request joint imputation of X and Y by using the FHDI method. The two CLEVVAR= options specify variables CX and CY to contain the imputation bins for variables X and Y, respectively. These statements also request imputation-adjusted replicate weights for the jackknife replication method. The OUTPUT statement stores the imputed values in the data set ImputedD3 and stores the jackknife coefficients in the data set OJKC. The FRACTIONALWEIGHTS= option in the OUTPUT statement saves the fractional weights in the ImputedD3 data set. The SEED= option specifies a seed for the random number generator, and the NDONORS=3 option requests three second-stage donor cells for each first-stage FEFI level.

```
proc surveyimpute data=Example method=fhdi seed=8943028 ndonors=3;
   var X (clevvar=CX) Y (clevvar=CY);
   output out=ImputedD3 fractionalweights=FracWt outjkcoefs=OJKC;
run;
```

The procedure first imputes the missing values in CX, CY, X, and Y by using the two-stage FEFI, as described in the section "Example of Two-Stage FEFI" on page 9932. The two-stage FEFI imputed values along with the imputation-adjusted weights and the imputation-adjusted replicate weights are shown in Figure 119.14.

The FHDI selects three second-stage donor cells in each first-stage FEFI cell. The imputed data set after the FHDI is displayed in Figure 119.16.

Figure 119.16 Fractional Hot-Deck Imputation Using Three Donors

Unit	ImpIndex	ImpWt	FracWt	X	CX	Y	CY
1	0	1.0000	1.0000	0.3	0	-0.54	0
2	0	1.0000	1.0000	0.2	0	-0.77	0
3	0	1.0000	1.0000	1.7	0	-0.59	0
4	0	1.0000	1.0000	1.7	0	-0.59	0
5	1	0.1340	0.1340	1.0	0	-0.77	0
5	2	0.1340	0.1340	1.0	0	-0.54	0
5	3	0.2680	0.2680	1.0	0	-0.59	0
5	4	0.1547	0.1547	1.0	0	-0.03	1
5	5	0.1547	0.1547	1.0	0	0.78	1
5	6	0.1547	0.1547	1.0	0	0.95	1
6	0	1.0000	1.0000	1.8	0	-0.03	1
7	0	1.0000	1.0000	2.0	0	0.95	1
8	0	1.0000	1.0000	1.9	0	0.78	1
9	0	1.0000	1.0000	6.7	1	-0.15	0
10	0	1.0000	1.0000	6.0	1	-1.01	0
11	0	1.0000	1.0000	3.3	1	-1.86	0
12	0	1.0000	1.0000	7.3	1	-0.21	0
13	0	1.0000	1.0000	6.7	1	0.80	1
14	1	0.1547	0.1547	1.8	0	1.23	1
14	2	0.1547	0.1547	1.9	0	1.23	1
14	3	0.1547	0.1547	2.0	0	1.23	1
14	4	0.1787	0.1787	2.9	1	1.23	1
14	5	0.1787	0.1787	6.7	1	1.23	1
14	6	0.1787	0.1787	10.0	1	1.23	1
15	0	1.0000	1.0000	2.9	1	0.65	1
16	0	1.0000	1.0000	9.6	1	0.95	1
17	0	1.0000	1.0000	10.0	1	0.13	1
18	1	0.0667	0.0667	0.2	0	-0.77	0
18	2	0.0667	0.0667	0.3	0	-0.54	0
18	3	0.1334	0.1334	1.7	0	-0.59	0
18	4	0.0770	0.0770	1.8	0	-0.03	1
18	5	0.0770	0.0770	1.9	0	0.78	1
18	6	0.0770	0.0770	2.0	0	0.95	1
18	7	0.0784	0.0784	3.3	1	-1.86	0
18	8	0.0784	0.0784	6.7	1	-0.15	0
18	9	0.0784	0.0784	7.3	1	-0.21	0
18	10	0.0889	0.0889	6.7	1	0.80	1
18	11	0.0889	0.0889	9.6	1	0.95	1
18	12	0.0889	0.0889	10.0	1	0.13	1

The FHDI is described as follows:

- Observation unit 1 has no missing value. Therefore, the ImpIndex value is 0; the FracWt value is 1; and the values of X, CX, Y, and CY are the same as the observed values for observation unit 1 in Figure 119.16. Because all observation units have a weight of 1, the fractional weights (FracWt) and the imputation-adjusted weights (ImpWt) are the same for all rows.

- Observation unit 5 has missing values in Y and CY. The variable CY has two imputed levels (0 and 1) from the first-stage imputation. The observed level of CX for observation unit 5 is 0.

  There are six rows for observation unit 5 after two-stage FEFI (Figure 119.14). ImpIndex values 1 to 3 contain imputed values when CY=0, and ImpIndex values 4 to 6 contain imputed values when CY=1. Because there are only three rows for each imputed level of CY and NDONORS=3 is specified in the PROC SURVEYIMPUTE statement, no selection is performed for FHDI for observation unit 5. Thus, all six imputed rows from FHDI (Figure 119.16) are the same as the six imputed rows from two-stage FEFI for observation unit 5 (Figure 119.14).

- Observation unit 14 has missing values in X and CX. The variable CX has two imputed levels (0 and 1) from the first-stage imputation. The observed level of CY for observation unit 14 is 1. There are seven rows for observation unit 14 after two-stage FEFI (Figure 119.14).

  ImpIndex values 1 to 3 for observation unit 14 contain imputed values for X when CX=0. Because there are only three rows for CX=0 and NDONORS=3 is specified in the PROC SURVEYIMPUTE statement, no selection is performed for FHDI for observation unit 14 and CX=0.

  ImpIndex values 4 to 7 for observation unit 14 contain imputed values for X when CX=1. Because the number of imputed rows for observation unit 14 and CX=1 is greater than 3, the procedure selects three donor cells by using a PPS sample with replacement. Donor cells that have imputed X values of 2.9, 6.7, and 10.0 are selected. Each row is assigned a second-stage fractional weight of 1/3. The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.54) for observation unit 14 when CX=1 to obtain the FHDI weight.

  Although three donor cells are selected for observation unit 14 when CX=0, the procedure selects six donor cells from four donor cells for observation units 14 and 18 together by using one PPS sample with replacement. Note that observation unit 14 for ImpIndex values 4 to 7 and observation unit 18 for ImpIndex values 11 to 14 have the same levels for the first-stage imputation variables CX (=1) and CY (=1).

  Thus, there are six rows for observation unit 14 after FHDI in Figure 119.16.

- Observation unit 18 has missing values in all four variables, X, CX, Y, and CY. The variable CX has two imputed levels (0 and 1) and the variable CY has two imputed levels (0 and 1) from the first-stage imputation. There are 14 rows for observation unit 18 after two-stage FEFI (Figure 119.14).

  ImpIndex values 1 to 3 for observation unit 18 after two-stage FEFI contain three imputed values for X and Y when CX=0 and CY=0 (Figure 119.14). Because there are only three imputed rows for CX=0 and CY=0 and NDONORS=3 is specified in the PROC SURVEYIMPUTE statement, no selection is performed for FHDI for observation unit 18 when CX=0 and CY=0. Similarly, no selection is performed for FHDI for observation unit 18 when CX=0 and CY=1. Thus, all rows for observation unit 18 and ImpIndex values 1 to 6 from FHDI (Figure 119.16) are the same as the rows for observation unit 18 and ImpIndex values 1 to 6 from two-stage FEFI (Figure 119.14).

  Three donor cells are selected from four donor cells for observation unit 18 and ImpIndex values 7 to 10, where CX=1 and CY=0. A PPS sample with replacement is used, where the fractional weights of the donor cells are used as the size measure. The selected donor cells are (3.3, –1.86), (6.7, –0.15), and (7.3, –0.21) for (X, Y). Each row is assigned a second-stage fractional weight of 1/3. The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.24) for the observation unit 18 in which CX=1 and CY=0 in order to obtain the FHDI weight.

  Similarly, donor cells (6.7, 0.80), (9.6, 0.95), and (10.0, 0.13) are selected for (X, Y) for the observation unit 18 in which CX=1 and CX=1. Each row is assigned a second-stage fractional weight of 1/3.

The second-stage fractional weight is then multiplied by the first-stage FEFI weight (0.27) for the observation unit 18 in which CX=1 and CY=1 in order to obtain the FHDI weight.

Thus, there are 12 imputed rows after FHDI for observation unit 18.

The resulting data set has 39 rows: 15 rows for fully observed units (ImpIndex=0), six rows for unit 5, six rows for unit 14, and 12 rows for unit 18. The sum of the fractional weights is 1 for all observation units.

The imputation-adjusted replicate weights are computed by applying the first-stage imputation, second-stage imputation, and neighbor adjustment (as discussed in section "Replicate Weight Adjustments for FHDI" on page 9959) independently in each replicate sample. The imputed data set along with first four imputation-adjusted replicate weights is displayed in Figure 119.17.

Figure 119.17 Fractional Hot-Deck Imputation with Imputation-Adjusted Replicate Weights

Unit	ImpIndex	ImpWt	FracWt	X	CX	Y	CY	ImpRepWt_1	ImpRepWt_2	ImpRepWt_3	ImpRepWt_4
1	0	1.0000	1.0000	0.3	0	-0.54	0	0	1.0588	1.0588	1.0588
2	0	1.0000	1.0000	0.2	0	-0.77	0	1.0588	0	1.0588	1.0588
3	0	1.0000	1.0000	1.7	0	-0.59	0	1.0588	1.0588	0	1.0588
4	0	1.0000	1.0000	1.7	0	-0.59	0	1.0588	1.0588	1.0588	0
5	1	0.1340	0.1340	1.0	0	-0.77	0	0.1637	0	0.1637	0.1637
5	2	0.1340	0.1340	1.0	0	-0.54	0	0	0.1637	0.1637	0.1637
5	3	0.2680	0.2680	1.0	0	-0.59	0	0.3274	0.3274	0.1637	0.1637
5	4	0.1547	0.1547	1.0	0	-0.03	1	0.1893	0.1893	0.1893	0.1893
5	5	0.1547	0.1547	1.0	0	0.78	1	0.1893	0.1893	0.1893	0.1893
5	6	0.1547	0.1547	1.0	0	0.95	1	0.1893	0.1893	0.1893	0.1893
6	0	1.0000	1.0000	1.8	0	-0.03	1	1.0588	1.0588	1.0588	1.0588
7	0	1.0000	1.0000	2.0	0	0.95	1	1.0588	1.0588	1.0588	1.0588
8	0	1.0000	1.0000	1.9	0	0.78	1	1.0588	1.0588	1.0588	1.0588
9	0	1.0000	1.0000	6.7	1	-0.15	0	1.0588	1.0588	1.0588	1.0588
10	0	1.0000	1.0000	6.0	1	-1.01	0	1.0588	1.0588	1.0588	1.0588
11	0	1.0000	1.0000	3.3	1	-1.86	0	1.0588	1.0588	1.0588	1.0588
12	0	1.0000	1.0000	7.3	1	-0.21	0	1.0588	1.0588	1.0588	1.0588
13	0	1.0000	1.0000	6.7	1	0.80	1	1.0588	1.0588	1.0588	1.0588
14	1	0.1547	0.1547	1.8	0	1.23	1	0.1656	0.1656	0.1656	0.1656
14	2	0.1547	0.1547	1.9	0	1.23	1	0.1656	0.1656	0.1656	0.1656
14	3	0.1547	0.1547	2.0	0	1.23	1	0.1656	0.1656	0.1656	0.1656
14	4	0.1787	0.1787	2.9	1	1.23	1	0.1873	0.1873	0.1873	0.1873
14	5	0.1787	0.1787	6.7	1	1.23	1	0.1873	0.1873	0.1873	0.1873
14	6	0.1787	0.1787	10.0	1	1.23	1	0.1873	0.1873	0.1873	0.1873
15	0	1.0000	1.0000	2.9	1	0.65	1	1.0588	1.0588	1.0588	1.0588
16	0	1.0000	1.0000	9.6	1	0.95	1	1.0588	1.0588	1.0588	1.0588
17	0	1.0000	1.0000	10.0	1	0.13	1	1.0588	1.0588	1.0588	1.0588
18	1	0.0667	0.0667	0.2	0	-0.77	0	0.0764	0	0.0764	0.0764
18	2	0.0667	0.0667	0.3	0	-0.54	0	0	0.0764	0.0764	0.0764
18	3	0.1334	0.1334	1.7	0	-0.59	0	0.1528	0.1528	0.0764	0.0764
18	4	0.0770	0.0770	1.8	0	-0.03	1	0.0884	0.0884	0.0884	0.0884
18	5	0.0770	0.0770	1.9	0	0.78	1	0.0884	0.0884	0.0884	0.0884
18	6	0.0770	0.0770	2.0	0	0.95	1	0.0884	0.0884	0.0884	0.0884
18	7	0.0784	0.0784	3.3	1	-1.86	0	0.0882	0.0882	0.0882	0.0882
18	8	0.0784	0.0784	6.7	1	-0.15	0	0.0882	0.0882	0.0882	0.0882
18	9	0.0784	0.0784	7.3	1	-0.21	0	0.0882	0.0882	0.0882	0.0882
18	10	0.0889	0.0889	6.7	1	0.80	1	0.0999	0.0999	0.0999	0.0999
18	11	0.0889	0.0889	9.6	1	0.95	1	0.0999	0.0999	0.0999	0.0999
18	12	0.0889	0.0889	10.0	1	0.13	1	0.0999	0.0999	0.0999	0.0999

Hot-Deck Imputation

Imputation techniques that use observed values from the sample to impute (fill in) missing values are known as hot-deck imputation. For more information, see Fellegi and Holt (1976), Lohr (2022, Section 8.7.3), Andridge and Little (2010), Fuller (2009, Section 5.2.1), Särndal and Lundström (2005), and Bethlehem (2009, Section 8.3). The observation unit that contains the missing values is known as the recipient unit, and the observation unit that provides the value for imputation is known as the donor unit. It is common to group similar observation units in one imputation cell and then select the donor units from the same imputation cell as the recipient unit. This imputation technique is also known as hot-deck imputation within classes (Särndal, Swensson, and Wretman 1992, p. 593). If the donor unit is selected randomly for a recipient unit, then the imputation technique is called random hot-deck imputation.

PROC SURVEYIMPUTE implements cell-based random hot-deck imputation methods. You identify imputation cells by using the CELLS statement and specify a random selection method by using the SELECTION= suboption for METHOD=HOTDECK in the PROC SURVEYIMPUTE statement. If an observation unit does not contain any missing values in the analysis variables, then the observation unit is considered as a donor unit. If an observation unit contains at least one missing value in the analysis variables, then the observation unit is treated as a recipient unit. You specify the analysis variables in the VAR statement. If no donors are found for an observation unit in the imputation cell, then the missing items are not imputed for that observation unit.

To illustrate the technique, consider the data set in Figure 119.18. Assume these six units are in the same imputation cell and you want to use the hot-deck imputation to impute the missing values in units 3, 4, and 6.

Figure 119.18 Units in an Imputation Cell

Unit	Age	Gender	Pregnancy
1	48	Male	0
2	22	Female	1
3	31		2
4	.	Male	0
5	22	Female	0
6	35		.

The following SAS statements use units 1, 2, and 5, which contain no missing values as the donor units for all recipient units 3, 4, and 6. For every recipient unit, PROC SURVEYIMPUTE selects a donor unit at random from all available donor units and replaces the missing values in the recipient unit with the observed values from the selected donor unit.

Both Age and Pregnancy are missing for unit 6, and PROC SURVEYIMPUTE uses the same donor to impute both items. Using the same donor unit to impute multiple items helps preserve the observed multivariate relationship.

However, it is possible to generate impossible responses. For example, if observation unit 1 is randomly selected as the donor unit for observation unit 3, then observation unit 3 will have Gender=Male but Pregnancy=2—a biological impossibility! To deal with such situations, consider filling the deterministic values before using imputation. For example, unit 3 is reported to be pregnant twice, and thus must be a female respondent. So you assign Gender=Female for unit 3 before using PROC SURVEYIMPUTE.

```
proc surveyimpute method=hotdeck(selection=srswr);
   var Age Gender Pregnancy;
   output out=JointHotDeck;
run;
```

If you do not want to preserve the multivariate relationship among the items, then you can impute the items marginally. The following SAS statements impute Age marginally. The recipient unit is 4, and the possible donor units are 1, 2, 3, 5, and 6.

```
proc surveyimpute method=hotdeck(selection=srswr);
   var Age;
   output out=MarginalHotDeck;
run;
```

The random selection of donors preserves the expectations within the imputation cells, but the random selection process increases the variance (Fuller 2009, p. 289). The variance estimator must include both the sampling variability and the imputation variability (Särndal and Lundström 2005).

PROC SURVEYIMPUTE implements the random selection methods that are described in the following subsections.

Approximate Bayesian Bootstrap

Suppose there are m recipient units and r donor units in an imputation cell. The approximate Bayesian bootstrap technique uses the following two steps for donor selection:

1. Select a sample of size r from the r donor units by using a simple random sample with replacement. The selected set is called the donor set for this imputation cell.
2. Select m donor units from the donor set by using a simple random sample with replacement.

To account for the imputation variance, you must select multiple donor units for every recipient unit. You can use the NDONORS= option in the PROC SURVEYIMPUTE statement to select multiple donor units. The procedure repeats the preceding two steps independently to select multiple donor units for every recipient unit. If you have a stratified design, Little and Rubin (2002, p. 89) suggest defining the imputation cells that are nested within strata. By default, the procedure does not assume that the cells are nested within the strata. You must specify the STRATA variables in the CELLS statement to define the imputation cells that are nested within the strata. For more information about the approximate Bayesian bootstrap method, see Rubin and Schenker (1986), Little and Rubin (2002, p. 89), and Kim (2002).

Simple Random Samples without Replacement

Suppose there are m recipient units and r donor units in an imputation cell. PROC SURVEYIMPUTE selects a simple random sample without replacement of size m from the r donors. One requirement for this selection method is that the number of donor units must be greater than or equal to the number of the recipient units. PROC SURVEYIMPUTE uses the selection-rejection method described in Tillé (2006, p. 48). If you select multiple (d) donor units for each recipient unit (by using the NDONORS= option in the PROC SURVEYIMPUTE statement), then the procedure selects d simple random samples independently.

Simple Random Samples with Replacement

Suppose there are m recipient units and r donor units in an imputation cell. PROC SURVEYIMPUTE selects a simple random sample with replacement of size m from the r donors. If you select multiple (d) donor units for each recipient unit (by using the NDONORS= option in the PROC SURVEYIMPUTE statement), then the procedure selects d simple random samples independently.

Weighted Selection

Suppose there are m recipient units and r donor units in an imputation cell. Let w_i be the weight of the donor unit i. PROC SURVEYIMPUTE selects a probability proportional to donor weight, w_i, with replacement sample of size m from the r donors. The procedure uses the probability proportional to size sampling algorithm described in Särndal, Swensson, and Wretman (1992, p. 97). For more information about the weighted hot-deck method, see Shao and Tu (1995, p. 271), and Rao and Shao (1992).

Replication Variance Estimation

Replication methods are useful for estimating variances that account for both the sampling variability and the imputation variability. If you specify the METHOD=FEFI or the METHOD=FHDI option in the PROC SURVEYIMPUTE statement, then, by default, the procedure creates imputation-adjusted jackknife replicate weights unless you also specify the VARMETHOD=NONE option in the same statement. If you specify your own replicate weights by using the REPWEIGHTS statement and if you specify the METHOD=FEFI or the METHOD=FHDI option in the PROC SURVEYIMPUTE statement, then the procedure creates new replicate weights by adjusting the replicate weights that you provide for imputation. It does not create imputation-adjusted replicate weights when you specify the METHOD=HOTDECK option in the PROC SURVEYIMPUTE statement.

The SURVEYIMPUTE procedure does not compute any variances. The replicate weights that are created can be used in any SAS/STAT survey procedure for variance computation. For an example, see the section "Getting Started: SURVEYIMPUTE Procedure" on page 9895.

Replication methods draw multiple replicates (also called subsamples) from a full sample according to a specific resampling scheme. The most commonly used resampling schemes are the balanced repeated replication (BRR) method, the jackknife method, and the bootstrap method. For each replicate, the original weights are modified for the primary sampling units (PSUs) in the replicates to create replicate weights. The parameters of interest are estimated by using the replicate weights for each replicate. These estimates are also known as replicate estimates. Then the variances of parameters of interest are estimated by estimating variability among the replicate estimates. The SURVEYIMPUTE procedure automatically creates replicate weights based on the replication method that you specify; alternatively you can use the REPWEIGHTS statement to provide your own replicate weights.

The following subsections provide details about how the replication weights are created for each variance estimation method.

Bootstrap Method

The naive bootstrap variance estimator that is suitable for infinite population is not consistent when applied to complex surveys. Bootstrap replicate samples for complex surveys are created by using a simple random sample with replacement of primary sampling units (PSUs) within each stratum. PSUs in different strata are sampled independently. The original sampling weights are then adjusted in each replicate to reflect the full sample. These adjusted weights are also called bootstrap replicate weights. McCarthy and Snowden (1985), Rao and Wu (1988), Sitter (1992b), and Sitter (1992a) provide several adjusted bootstrap variance estimators that are consistent for complex surveys. For more information about bootstrap variance estimation for complex surveys, see Mashreghi, Haziza, and Léger (2016), Beaumont and Patak (2012), Lohr (2022, Section 9.3.3), Fuller (2009, Section 4.5), Wolter (2007, Chapter 5), and Shao and Tu (1995, Section 6.2.4).

If you do not provide replicate weights by using the REPWEIGHTS statement, then the BOOTSTRAP option in the PROC SURVEYIMPUTE statement creates bootstrap replicate weights for you. This bootstrap method is similar to the method of Rao, Wu, and Yue (1992) and is also known as bootstrap weights method (Mashreghi, Haziza, and Léger 2016).

If you use the FEFI or the FHDI method, then the unadjusted bootstrap weights are adjusted for the imputation to create the *imputation-adjusted replicate weights*. The section "Unadjusted Bootstrap Replicate Weights" on page 9955 describes how the unadjusted replicate weights are created, and sections "Imputation-Adjusted Replicate Weights" on page 9958 and "Replicate Weight Adjustments for FHDI" on page 9959 describe how the imputation-adjusted replicate weights are created.

Unadjusted Bootstrap Replicate Weights

Each replicate is obtained by selecting a simple random sample with replacement of m_h PSUs from stratum h. The rth bootstrap replicate weight for observation unit j in PSU i and stratum h is given by

$$w_{hij}^{(r)} = w_{hij}\left\{1 - \sqrt{(1-f_h)m_h/(n_h-1)} + \sqrt{(1-f_h)m_h/(n_h-1)}(n_h/m_h)k_{hi}^{(r)}\right\}$$

where $k_{hi}^{(r)}$ is the number of times PSU i in stratum h is selected in replicate sample r, and f_h is the sampling fraction in stratum h.

If you use the hot-deck imputation method, then you can use the OUTPUT statement to store the unadjusted replicate weights. The unadjusted replicate weights are not saved when you use the FEFI or FHDI method. You should use the imputation-adjusted replicate weights for variance estimation from a fractionally imputed data set.

For more information about how the bootstrap variance estimators are computed for related statistics, see the section "Bootstrap Method" in each of the following chapters: Chapter 118, "The SURVEYFREQ Procedure," Chapter 120, "The SURVEYLOGISTIC Procedure," Chapter 121, "The SURVEYMEANS Procedure," Chapter 122, "The SURVEYPHREG Procedure," and Chapter 123, "The SURVEYREG Procedure."

Balanced Repeated Replication (BRR) Method

The balanced repeated replication (BRR) method requires that the full sample be drawn by using a stratified sample design with two primary sampling units (PSUs) per stratum. The BRR method constructs half-sample replicates by deleting one PSU per stratum according to a Hadamard matrix and doubling the original weight of the other PSU in that stratum. If you use the FEFI or the FHDI method, then the unadjusted BRR weights are adjusted for the imputation to create the *imputation-adjusted replicate weights*. The sections "Unadjusted BRR Replicate Weights" on page 9956 and "Unadjusted Fay's BRR Replicate Weights" on page 9956

describe how the unadjusted replicate weights are created, and the section "Imputation-Adjusted Replicate Weights" on page 9958 describes how the imputation-adjusted replicate weights are created.

Unadjusted BRR Replicate Weights

Let H be the total number of strata. The total number of replicates, R, is the smallest multiple of 4 that is greater than H. However, if you prefer a larger number of replicates, you can specify the REPS=*n method-option*. If an $n \times n$ Hadamard matrix cannot be constructed, the number of replicates is increased until a Hadamard matrix becomes available.

Each replicate is obtained by deleting one PSU per stratum according to a corresponding Hadamard matrix and adjusting the original weights for the remaining PSUs. The new weights are called replicate weights.

Replicates are constructed by using the first H columns of the $R \times R$ Hadamard matrix. The rth ($r = 1, 2, \ldots, R$) replicate is drawn from the full sample according to the rth row of the Hadamard matrix as follows:

- If the (r, h) element of the Hadamard matrix is 1, then the first PSU of stratum h is included in the rth replicate and the second PSU of stratum h is excluded.
- If the (r, h) element of the Hadamard matrix is –1, then the second PSU of stratum h is included in the rth replicate and the first PSU of stratum h is excluded.

The replicate weights of the remaining PSUs in each half sample are then doubled to their original weights. For more information about the BRR method, see Wolter (2007) and Lohr (2022).

By default, PROC SURVEYIMPUTE generates an appropriate Hadamard matrix automatically to create the replicates. You can display the Hadamard matrix by specifying the VARMETHOD=BRR(PRINTH) *method-option*. If you provide a Hadamard matrix by specifying the VARMETHOD=BRR(HADAMARD=) *method-option*, then the replicates are generated according to the provided Hadamard matrix.

For more information about how the BRR variance estimators are computed for related statistics, see the section "Balanced Repeated Replication (BRR) Method" in each of the following chapters: Chapter 118, "The SURVEYFREQ Procedure," Chapter 120, "The SURVEYLOGISTIC Procedure," Chapter 121, "The SURVEYMEANS Procedure," Chapter 122, "The SURVEYPHREG Procedure," and Chapter 123, "The SURVEYREG Procedure."

Unadjusted Fay's BRR Replicate Weights

The traditional BRR method constructs half-sample replicates by deleting one PSU per stratum according to a Hadamard matrix and doubling the original weight of the other PSU. Fay's BRR method uses the Fay coefficient, ϵ ($0 \leq \epsilon < 1$), and instead of deleting one PSU per stratum, it multiplies the original weight by the coefficient ϵ. The original weight of the remaining PSU in that stratum is multiplied by $2 - \epsilon$. PROC SURVEYIMPUTE uses $\epsilon = 0.5$ as the default value; alternatively, you can specify a value for ϵ by using the FAY= *method-option*. When $\epsilon = 0$, Fay's method becomes the traditional BRR method. For more information, see Dippo, Fay, and Morganstein (1984); Fay (1984, 1989); Judkins (1990). Because the traditional BRR method uses only half of the total sample in every replicate, some observed levels of the analysis variables might not be available in the replicate samples. Fay's BRR method is especially useful in this situation because it uses all the sampled units in every replicate.

For more information about how Fay's BRR variance estimators are computed for related statistics, see the section "Balanced Repeated Replication (BRR) Method" in each of the following chapters: Chapter 118,

"The SURVEYFREQ Procedure," Chapter 120, "The SURVEYLOGISTIC Procedure," Chapter 121, "The SURVEYMEANS Procedure," Chapter 122, "The SURVEYPHREG Procedure," and Chapter 123, "The SURVEYREG Procedure."

Hadamard Matrix

PROC SURVEYIMPUTE uses a Hadamard matrix to construct replicates for BRR variance estimation. You can provide a Hadamard matrix for replicate construction by using the HADAMARD= *method-option* for VARMETHOD=BRR. Otherwise, PROC SURVEYIMPUTE generates an appropriate Hadamard matrix. You can display the Hadamard matrix by specifying the PRINTH *method-option*.

A Hadamard matrix $\mathbf{A}$ of dimension R is a square matrix that has all elements equal to 1 or –1 such that $\mathbf{A}'\mathbf{A} = R\mathbf{I}$, where $\mathbf{I}$ is an identity matrix of appropriate order. The dimension of a Hadamard matrix must equal 1, 2, or a multiple of 4.

For example, the following matrix is a Hadamard matrix of dimension $k = 8$:

$$\begin{array}{rrrrrrrr}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{array}$$

For BRR replicate construction, the dimension of the Hadamard matrix must be at least H, where H denotes the number of first-stage strata in your design. If a Hadamard matrix of a particular dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a SAS data set in the HADAMARD= *method-option*. You must ensure that the matrix that you provide is actually a Hadamard matrix; PROC SURVEYIMPUTE does not check the validity of your Hadamard matrix.

For more information about how the Hadamard matrix is used to construct replicates for BRR variance estimation, see the section "Unadjusted BRR Replicate Weights" on page 9956.

Jackknife Method

The jackknife method of variance estimation deletes one PSU at a time from the full sample to create replicates. This method is also known as the delete-1 jackknife method because it deletes exactly one PSU in every replicate. The total number of replicates R is the same as the total number of PSUs. In each replicate, the sampling weights of the remaining PSUs are modified by the jackknife coefficient α_r. The modified weights are called *replicate weights*. If you use the FEFI or the FHDI method, then the unadjusted replicate weights are adjusted for the imputation to create the *imputation-adjusted replicate weights*. The section "Unadjusted Jackknife Replicate Weights" on page 9958 describes how the unadjusted replicate weights are created, and the section "Imputation-Adjusted Replicate Weights" on page 9958 describes how the imputation-adjusted replicate weights are created.

Unadjusted Jackknife Replicate Weights

Let PSU i in stratum h_r be omitted for the rth replicate. Then the jackknife coefficient, α_r, and replicate weights, $w_{hij}^{(r)}$, are computed as

$$\alpha_r = \begin{cases} \frac{n_{h_r}-1}{n_{h_r}} & \text{for a stratified design} \\ \frac{R-1}{R} & \text{for designs without stratification} \end{cases}$$

$$w_{hij}^{(r)} = \begin{cases} w_{hij} & \text{if observation unit } j \text{ is not in donor stratum } h_r \\ 0 & \text{if observation unit } j \text{ is in PSU } i \text{ of donor stratum } h_r \\ w_{hij}/\alpha_r & \text{if observation unit } j \text{ is not in PSU } i \text{ but is in donor stratum } h_r \end{cases}$$

If you use the hot-deck imputation method, then you can use the OUTPUT statement in PROC SURVEYIMPUTE to store the unadjusted replicate weights. The unadjusted replicate weights are not saved for the FEFI or the FHDI method. You should use the imputation-adjusted replicate weights for variance estimation from a fractionally imputed data set. Use the OUTJKCOEFS= option in the OUTPUT statement to store the jackknife coefficients in a SAS data set.

For more information about how the jackknife variance estimators are computed for related statistics, see the section "Jackknife Method" in each of the following chapters: Chapter 118, "The SURVEYFREQ Procedure," Chapter 120, "The SURVEYLOGISTIC Procedure," Chapter 121, "The SURVEYMEANS Procedure," Chapter 122, "The SURVEYPHREG Procedure," and Chapter 123, "The SURVEYREG Procedure."

Imputation-Adjusted Replicate Weights

If you use the hot-deck imputation technique by specifying the METHOD=HOTDECK option in the PROC SURVEYIMPUTE statement, the procedure does not create imputation-adjusted replicate weights. Naive variance estimators that do not use imputation-adjusted replicate weights and assume the imputed data as the observed data might underestimate the true variance. For more information, see Haziza (2009); Särndal and Lundström (2005); Rao and Shao (1992).

If specify the METHOD=FEFI or the METHOD=FHDI option in the PROC SURVEYIMPUTE statement, the procedure adjusts the replicate weights for imputation. The imputation-adjusted replicate weights should be used with other SAS/STAT survey procedures to estimate the variance of an estimator that uses the imputed data. For more information, see Fuller (2009, Section 5.2.2) and Kim and Shao (2014, Section 4.6).

Let $w_i^{(r)}$ be the unadjusted replicate weight for observation unit i. To facilitate discussion, separate subscripts for strata, clusters, and imputation cells are omitted. The unadjusted replicate weights can come from a jackknife method as described in the section "Unadjusted Jackknife Replicate Weights" on page 9958, from a BRR method as described in the section "Unadjusted BRR Replicate Weights" on page 9956, or from a bootstrap method as described in the section "Unadjusted Bootstrap Replicate Weights" on page 9955, or they can be specified by using the REPWEIGHTS statement. The adjustment follows the similar EM-by-weighting algorithm that is described in the section "Fully Efficient Fractional Imputation" on page 9924 but uses the replicate weights, $w_i^{(r)}$, instead of the full sample weight, w_i.

In particular, the joint probabilities for the tth M-step and the rth replicate weight are computed by

$$\tilde{\pi}_{(t)}^{(r)}(\kappa_1 \cdots \kappa_P) = \left\{ \sum_i \sum_l w_i^{(r)} w_{il(t-1)}^{(r)} \right\}^{-1} \sum_i \sum_l w_i^{(r)} w_{il(t-1)}^{(r)} I(Z_{i1} = \kappa_1, \ldots, Z_{iP} = \kappa_P)$$

for all i, l, and $t > 0$.

The rth replicate fractional weights for the tth E-step is computed by

$$w_{il(t)}^{(r)} = \left\{ \sum_{k=1}^{M_l} \tilde{\pi}_{(t)}^{(r)}(\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{miss}[k]}) \right\}^{-1} \tilde{\pi}_{(t)}^{(r)}(\mathbf{Z}_{i,\text{obs}}, \mathbf{Z}_{i,\text{miss}[l]})$$

where M_l is the number of donor cells.

Replicate Weight Adjustments for FHDI

If you use the FHDI method (by specifying the METHOD=FHDI option in the PROC SURVEYIMPUTE statement), the procedure adjusts the replicate weights for imputation. You must use the imputation-adjusted replicate weights with other SAS/STAT survey procedures to estimate the variance of an estimator that uses the imputed data.

Let m_2 be the number of second-stage donor cells that are requested for FHDI (the value of the NDONORS= option in the PROC SURVEYIMPUTE statement), and let $m_{2u,i}$ be the unique number of selected second-stage donor cells for observation unit i. Further assume that $h_{l_2|l_1}$ is the number of times the second-stage donor cell l_2 is selected for first-stage donor cell l_1.

Let $w_i^{(r)}$ be the unadjusted replicate weight, $w_{il_1}^{(r)}$ be the first-stage fractional replicate weight, and $w_{il_2|l_1}^{(r)}$ be the second-stage replicate weight from two-stage FEFI for observation unit i, first-stage donor cell l_1, second-stage donor cell l_2, and replicate sample r. Let $M_{il_2|l_1}$ be the total number of second-stage donor cells conditional on the first-stage donor cell l_1 for observation unit i.

The following weight adjustments are available:

- *No adjustment*: If you specify the REPWTADJ=NONE option, then PROC SURVEYIMPUTE does not adjust the replicate weights for FHDI.

  The two-stage imputation-adjusted fractional replicate weight for the rth replicate sample for the second-stage donor cell l_2 in the first-stage donor cell l_1 for observation unit i is

  $$w_{il_1l_2}^{(r)} = \begin{cases} w_{il_1}^{(r)} w_{il_2|l_1}^{(r)} & , \quad M_{il_2|l_1} \le m_2 \\ w_{il_1}^{(r)} h_{l_2|l_1}/m_2 & , \quad M_{il_2|l_1} > m_2 \end{cases}$$

  where $l_2 = 1, 2, \ldots, \min(M_{il_2|l_1}, m_{2u,i})$.

  The two-stage imputation-adjusted replicate weight for observation unit i, first-stage donor cell l_1, and second-stage donor cell l_2 in the rth replicate sample is $w_i^{(r)} w_{il_1l_2}^{(r)}$.

- *Ratio adjustment*: If you specify the REPWTADJ=RATIO option, then PROC SURVEYIMPUTE adjusts the replicate weights for FHDI by using ratio adjustment.

  The two-stage imputation-adjusted fractional replicate weight for the rth replicate sample for the second-stage donor cell l_2 in the first-stage donor cell l_1 for observation unit i is

  $$w_{il_1l_2}^{(r)} = \begin{cases} w_{il_1}^{(r)} w_{il_2|l_1}^{(r)} & , \quad M_{il_2|l_1} \le m_2 \\ w_{il_1}^{(r)} \left[s_i^{(r)} \right]^{-1} h_{l_2|l_1} \left[w_{il_2|l_1}^{(r)} / w_{il_2|l_1} \right] & , \quad M_{il_2|l_1} > m_2 \end{cases}$$

where $l_2 = 1, 2, \ldots, \min(M_{il_2|l_1}, m_{2u,i})$ and $s_i^{(r)} = \sum_{l_2=1}^{m_{2u,i}} \left[w_{il_2|l_1}^{(r)} / w_{il_2|l_1} \right] h_{l_2|l_1}$ is the sum of the ratios of the second-stage replicate fractional weight to the second-stage full sample fractional weight for observation unit i.

If $w_{il_2|l_1}^{(r)} = 0$ for all selected second-stage donor cells in the rth replicate sample, then each selected second-stage donor cell is assigned a second-stage fractional weight of $h_{l_2|l_1}/m_2$.

The two-stage imputation-adjusted replicate weight for observation unit i, first-stage donor cell l_1, and second-stage donor cell l_2 in the rth replicate sample is $w_i^{(r)} w_{il_1l_2}^{(r)}$.

- *Neighbor adjustment*: If you specify the REPWTADJ=NEIGHBOR option or if you do not specify the REPWTADJ= option, then PROC SURVEYIMPUTE adjusts the replicate weights for FHDI by using neighbor adjustment.

  Neighbor adjustment first computes the proportion of the full-sample fractional weights that fall in each of m_2 equally spaced intervals and then adjusts the replicate sample fractional weights by using the proportions from the full sample. Let $S_k = \sum_{j_2=1}^{k} w_{i\,j_2|l_1}$ be the cumulative sum of the second-stage full-sample fractional weights for the first k second-stage donor cells. S_k depends on both i and l_1 but, for simplicity, subscripts i and l_1 are not used in S_k. By construction, $S_{k-1} \le S_k$. Define factors α_{dk} according to how the interval $[S_{k-1}, S_k]$ for the kth second-stage donor cell overlaps with the dth equally spaced interval $[(d-1)/m_2, d/m_2]$:

$$\alpha_{dk} = \begin{cases} \frac{\min(S_k, \frac{d}{m_2}) - \max(S_{k-1}, \frac{d-1}{m_2})}{S_k - S_{k-1}} & , \quad \min(S_k, \frac{d}{m_2}) > \max(S_{k-1}, \frac{d-1}{m_2}) \\ 0 & , \quad \text{otherwise} \end{cases}$$

  for $d = 1, 2, \ldots, m_2$ and $k = 1, 2, \ldots, M_{il_2|l_1}$. The replicate fraction for the second-stage donor cell l_2 is computed as

$$w_{il_2|l_1}^{*(r)} = \sum_{k=1}^{M_{il_2|l_1}} w_{ik|l_1}^{(r)} \alpha_{l_2k}$$

  for all $r = 1, 2, \ldots, R$, and $l_2 = 1, 2, \ldots, m_2$. Let $s_i^{*(r)} = \sum_{l_2=1}^{m_2} w_{il_2|l_1}^{*(r)}$.

  The two-stage imputation-adjusted fractional replicate weight for the rth replicate sample for the second-stage donor cell l_2 in the first-stage donor cell l_1 for observation unit i is

$$w_{il_1l_2}^{(r)} = \begin{cases} w_{il_1}^{(r)} w_{il_2|l_1}^{(r)} & , \quad M_{il_2|l_1} \le m_2 \\ w_{il_1}^{(r)} \left[s_i^{*(r)} \right]^{-1} w_{il_2|l_1}^{*(r)} & , \quad M_{il_2|l_1} > m_2 \end{cases}$$

  where $l_2 = 1, 2, \ldots, \min(M_{il_2|l_1}, m_{2u,i})$.

  If $w_{il_2|l_1}^{(r)} = 0$ for all selected second-stage donor cells in the rth replicate sample, then each selected second-stage donor cell is assigned a second-stage fractional weight of $h_{l_2|l_1}/m_2$, where $h_{l_2|l_1}$ is the number of times the second-stage donor cell l_2 is selected.

  The two-stage imputation-adjusted replicate weight for observation unit i, first-stage donor cell l_1, and second-stage donor cell l_2 in the rth replicate sample is $w_i^{(r)} w_{il_1l_2}^{(r)}$.

Output Data Sets

PROC SURVEYIMPUTE creates an output data set to store the imputed data and the replicate weights, and an output data set to store the jackknife coefficients for jackknife variance estimation. You can use the Output Delivery System (ODS) to create a SAS data set from any piece of PROC SURVEYIMPUTE output. For more information, see the section "Displayed Output" on page 9961.

OUT= Output Data Set

You can use the OUTPUT statement to create a data set to store the imputed data. The OUTPUT OUT= data set contains all the variables from the input data set, imputed values for missing values for the variables in the VAR statement, and some observation-level quantities. These quantities can include the fractionally adjusted weights, replicate weights, recipient numbers, and donor identifications.

Jackknife Coefficients Output Data Set

If you specify the OUTJKCOEFS= option in the OUTPUT statement, PROC SURVEYIMPUTE stores the jackknife coefficients in an output data set. The OUTJKCOEFS= output data set contains one observation for each replicate. The OUTJKCOEFS= data set contains the following variables:

- Replicate: the replicate number for the jackknife coefficient
- JKCoefficient: the jackknife coefficient for the replicate
- DonorStratum: the stratum of the PSU that was deleted to construct the replicate, if you use a STRATA statement

You can use the JKCOEFS= option in the REPWEIGHTS statement in any SAS/STAT survey procedure to provide jackknife coefficients for that procedure. If the jackknife coefficients are different from $(R-1)/R$, where R is the total number of replicates, then you must provide the jackknife coefficients to correctly estimate the variance.

Displayed Output

If you use the NOPRINT option in the PROC SURVEYIMPUTE statement, the procedure does not display any output. Otherwise, PROC SURVEYIMPUTE displays results of the imputation in a collection of tables, which are described in the following subsections.

Class Level Information

If you use a CLASS statement, PROC SURVEYIMPUTE displays a "Class Level Information" table, which lists the categories of every CLASS variable that is used in the imputation. The ODS name of the "Class Level Information" table is ClassLevelInfo.

Convergence Status

If you specify METHOD=FEFI or METHOD=FHDI, PROC SURVEYIMPUTE displays a "Convergence Status" table, which shows the convergence status of the EM optimization routine. If the optimization routine converges, then the Status is set to 0; otherwise, the Status is set to 1. The ODS name of the "Convergence Status" table is ConvergenceStatus.

Design Summary

If you use a STRATA or CLUSTER statement, PROC SURVEYIMPUTE displays a "Design Summary" table, which provides information about the sample design. The table displays the total number of strata that are read and used, and the total number of clusters that are read and used. The ODS name of the "Design Summary" table is DesignSummary.

Donor Count

If you specify METHOD=FEFI or METHOD=FHDI, PROC SURVEYIMPUTE displays a "Donor Count" table, which shows the number of donor cells in the first-stage FEFI and the number of recipient units that use those donor cells. The ODS name of the "Donor Count" table is DonorCount.

Hadamard Matrix

If you specify the PRINTH *method-option* for VARMETHOD=BRR, PROC SURVEYIMPUTE displays the Hadamard matrix that is used to construct replicates for BRR variance estimation. If you provide a Hadamard matrix by using the HADAMARD= *method-option* for VARMETHOD=BRR but the procedure does not use the entire matrix, the procedure displays only the rows and columns that are actually used to construct replicates. The ODS name of the "Hadamard Matrix" table is HadamardMatrix.

Imputation Information

By default, PROC SURVEYIMPUTE displays an "Imputation Information" table, which provides information about the imputation method. The table displays the two-level name of the input data set, the name and label of the WEIGHT variable, the name and label of each STRATA variable, the name and label of each CLUSTER variable, the name of the imputation method used, the random number generator seed, the name of the replication method, and the number of replicates. The ODS name of the "Imputation Information" table is ImputationInfo.

Imputation Summary

By default, PROC SURVEYIMPUTE displays an "Imputation Summary" table, which provides summary information about the imputation. The table displays the number of observations and the sum of weights for the following:

- Nonmissing observations – all variables specified in the VAR statement have nonmissing values
- Missing – at least one variable specified in the VAR statement has a missing value
- Missing, Imputed – all missing values have been imputed
- Missing, Not Imputed – no missing values are imputed

- Missing, Partially Imputed – missing values in some variables are imputed, but missing values in some other variables are not imputed

The ODS name of the "Imputation Summary" table is ImputationSummary.

Iteration History

If you specify METHOD=FEFI or METHOD=FHDI, PROC SURVEYIMPUTE displays an "Iteration History" table, which provides information about the iteration history for the EM algorithm. The table displays iteration numbers, maximum absolute differences, and maximum relative absolute differences for the fractional weights over all the observations. The ODS name of the "Iteration History" table is IterationHistory.

Missing Data Patterns

By default, PROC SURVEYIMPUTE displays a "Missing Data Patterns" table, which provides information about the missing data patterns. The table displays the missing data pattern groups, "X" if the variable is observed in the group, and "." if the variable is missing in that group. In addition, it displays observation frequencies, the sum of weights, unweighted percentages, and weighted percentages for each group. The ODS name of the "Missing Data Patterns" table is MissPattern.

Number of Observations

By default, PROC SURVEYIMPUTE displays a "Number of Observations" table, which shows the number of observations that are read and used, and the sum of weights that are read and used in the imputation. The ODS name of the "Number of Observations" table is NObs.

Stratum Information

If you specify the LIST option in the STRATA statement, PROC SURVEYIMPUTE displays a "Stratum Information" table. This table provides the following information for each stratum:

- Stratum Index, which is a sequential stratum identification number
- STRATA variables, which list the levels of STRATA variables for the stratum
- Number of Observations, which is the number of observations used in the stratum
- Number of Clusters, which is the number of clusters in the stratum, if you specify a CLUSTER statement
- Sum of Weights, which is the sum of the weights for observations in the stratum
- Sampling Rate for the stratum, if you specify the TOTAL= or RATE= option

The ODS name of the "Stratum Information" table is StrataInfo.

Weighted Moments

If you specify METHOD=FHDI(DISP=MEAN) or METHOD=FHDI(DISP=SSCP), PROC SURVEYIMPUTE displays a "Weighted Moments" table, which shows the weighted means and weighted mean-squared deviations from both two-stage FEFI and FHDI for the variables that are specified in the VAR statement but not in the CLASS statement. The weighted moments for two-stage FEFI are computed by using imputation-adjusted weights from two-stage FEFI, and the weighted moments for FHDI are computed by using imputation-adjusted weights from FHDI. The ODS name of the "Weighted Moments" table is ContMean.

Weighted Mean SSCP for FEFI

If you specify METHOD=FHDI(DISP=SSCP), PROC SURVEYIMPUTE displays a "Weighted Mean SSCP for FEFI" table, which shows the weighted mean SSCP from two-stage FEFI for the variables that are specified in the VAR statement but not in the CLASS statement. The weighted mean SSCP matrix is computed by using imputation-adjusted weights from two-stage FEFI. The ODS name of the "Weighted Mean SSCP for FEFI" table is ContMSFEFI.

Weighted Mean SSCP for FHDI

If you specify METHOD=FHDI(DISP=SSCP), PROC SURVEYIMPUTE displays a "Weighted Mean SSCP for FHDI" table, which shows the weighted mean SSCP from FHDI for the variables that are specified in the VAR statement but not in the CLASS statement. The weighted mean SSCP matrix is computed by using imputation-adjusted weights from FHDI. The ODS name of the "Weighted Mean SSCP for FHDI" table is ContMSFHDI.

ODS Table Names

PROC SURVEYIMPUTE assigns a name to each table that it creates. You can use this name to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System." Table 119.5 lists the table names.

Table 119.5 ODS Tables Produced by PROC SURVEYIMPUTE

ODS Table Name	Description	Statement	Option
ClassLevelInfo	CLASS variable levels	CLASS	
ContMean	Weighted moments	PROC	METHOD=FHDI(DISP=MEAN | SSCP)
ContMSFEFI	Weighted mean SSCP for FEFI	PROC	METHOD=FHDI(DISP=SSCP)
ContMSFHDI	Weighted mean SSCP for FHDI	PROC	METHOD=FHDI(DISP=SSCP)
ConvergenceStatus	Convergence status	PROC	METHOD=FEFI | FHDI
DesignSummary	Design summary	STRATA or CLUSTER	
DonorCount	Donor count	PROC	METHOD=FEFI | FHDI

Table 119.5 *continued*

ODS Table Name	Description	Statement	Option
HadamardMatrix	Hadamard matrix	PROC	VARMETHOD=BRR (PRINTH)
ImputationInfo	Imputation information	PROC	
ImputationSummary	Imputation summary	PROC	
IterationHistory	Iteration history	PROC	METHOD= FEFI | FHDI
MissPattern	Missing data patterns	PROC	
NObs	Number of observations	PROC	
StrataInfo	Stratum information	STRATA / LIST	

Examples: SURVEYIMPUTE Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-stat`.

Example 119.1: Approximate Bayesian Bootstrap Imputation

(View the complete code for this example (svimpex1.sas) in the example repository.)

This example illustrates the approximate Bayesian bootstrap hot-deck imputation method by using a simulated data set from a fictitious survey of drug abusers. A stratified clustered sample of drug abuse treatment centers is taken from a list of available treatment centers. The list is first stratified based on geographic locations. From each strata, two or three treatment centers are sampled as the primary sampling units (PSU). Data are collected from individual patients within the selected treatment centers. The survey collects information about the substances that the patients used (such as drugs, alcohol, and marijuana) along with insurance information and treatment information.

The data set contains 736 observation units in 35 PSUs and 10 strata. The sum of the weights is 19,600. Therefore, the survey data represent a population of 19,600 patients from the study area. Some participants did not respond to all questions. The data set contains missing values in many variables.

To impute the missing items, you first need to decide whether to impute within imputation cells. Imputation cells divide the data into groups of similar units such that the recipient units share similar characteristics with the donor units in the same group. For example, it is reasonable to believe that different age groups, races, and income categories might have different responses to the drug abuse survey. You can use these characteristics to create imputation cells. Characteristics for imputation cells might come from the same survey or might come from other sources such as census data or previous surveys. In this example, assume that the imputation cells are available as a variable called ImputationCell in the data set.

The data set DrugAbuse contains the following items:

- Strata: stratum identification

- PSU: PSU identification (treatment centers)
- ObsWeight: observation weight for patients
- ImputationCell: imputation cell identification
- Age: age, in years
- Sex: 1 for female and 2 for male
- Race: 1 for white, 2 for black, and 3 for others
- Insurance: 1 if the patient has any insurance, and 2 otherwise
- Drug: 1 if the patient used any drugs in the past three months, and 2 otherwise
- Alcohol: 1 if the patient consumed any alcohol in the past month, and 2 otherwise
- Treatment: 1 if the patient is being treated for the first time, and 2 otherwise

```
data DrugAbuse;
   input Strata PSU ObsWeight ImputationCell Age Sex Race Insurance
         Drug Alcohol Treatment;
   datalines;
 1    1     5        1      74  1    1      3       2      2        1
 1    1     5        1      20  0    3      1       2      2        1
 1    1     5        3      42  1    2      1       1      2        1
 1    1     5        3      65  1    3      2       1      2        1
 1    1     5        2      53  1    1      1       1      1        1
 1    1     5        3      49  1    1      1       1      2        1
 1    1     5        3      51  0    2      1       2      1        1
 1    1     5        2      77  0    3      1       1      1        1
 1    1     5        2      26  1    1      1       1      2        1
 1    1     5        3      28  0    3      1       1      1        1
 1    1     5        1      71  1    1      1       2      2        1
 1    1     5        2      72  1    1      3       2      2        1
 1    1     5        3      24  1    1      1       1      1        1
 1    1     5        2      65  1    1      2       1      1        2
 1    1     5        3      47  1    1      1       1      1        1
 1    1     5        2      37  1    1      2       1      2        1
 1    1     5        2      46  1    1      3       1      1        1
 1    1     5        2      52  1    1      1       1      2        2
 1    1     5        3      60  0    3      1       1      2        1
 1    1     5        1      31  0    1      1       1      2        1
 1    2     5        1      23  0    3      3       1      1        1
 1    2     5        2      78  1    1      .       .      .        .
 1    2     5        2      29  1    1      1       1      1        1
 1    2     5        2      21  .    .      .       .      .        .

   ... more lines ...

10    4  55.5556      1      40  0    3      2       1      1        2
10    4  55.5556      1      32  1    3      1       2      2        1
10    4  55.5556      3      68  0    1      2       2      1        2
```

```
10    4  55.5556      3      35  1    1      2       1      2        2
;
```

The following statements request that the missing items be imputed by using the approximate Bayesian bootstrap hot-deck imputation method:

```
proc surveyimpute data=DrugAbuse method=hotdeck(selection=abb)
                  ndonors=5 seed=773269;
   var Sex Race Insurance Drug Alcohol Treatment;
   cells ImputationCell;
   output out=DrugAbuseABB;
run;
```

The PROC SURVEYIMPUTE statement invokes the procedure, the DATA= option specifies the input data set DrugAbuse, the METHOD= option requests the hot-deck imputation method, the METHOD=HOTDECK(SELECTION=ABB) option requests the approximate Bayesian bootstrap method, the NDONORS= option requests five donor units for every recipient unit, and the SEED= option specifies the random number generator seed. The VAR statement specifies the variables that are to be imputed, the CELLS statement identifies the imputation cell variable ImputationCell, and the OUT= option in the OUTPUT statement names the output data set DrugAbuseABB.

You do not need to use WEIGHTS, STRATA, and CLUSTER statements for the approximate Bayesian bootstrap method unless you want to create the jackknife replication weights by including the VARMETHOD=JACKKNIFE option in the PROC SURVEYIMPUTE statement. The selection of donors does not use the design information. However, if you want to select donors from the same strata or the same group of clusters, then you must include that information in the imputation cell.

Summary information about the imputation method, number of observations, and missing data patterns is shown in Output 119.1.1. The "Imputation Information" table summarizes the imputation method. The "Number of Observations" table shows that PROC SURVEYIMPUTE read and used all 736 observations. The "Missing Data Pattern" table displays the missing patterns in the data set. There are four different missing data pattern groups: all items observed, one item missing, four items missing, and all items missing. Of the observation units, 92.53% have all items observed; 4.64% have missing values in Treatment; 1.77% have missing values in Insurance, Drug, Alcohol, and Treatment; and 1.09% have missing values in all variables. Because the WEIGHT statement is not specified, these percentages represent the percentages of missing units in the input data.

Output 119.1.1 Imputation Summary

The SURVEYIMPUTE Procedure

Imputation Information	
Data Set	WORK.DRUGABUSE
Imputation Method	HOTDECK
Selection Method	ABB
Random Number Seed	773269
Replication Method	NONE

Number of Observations Read	736
Number of Observations Used	736

Output 119.1.1 *continued*

Missing Data Patterns

Group	Sex	Race	Insurance	Drug	Alcohol	Treatment	Freq	Sum of Weights	Unweighted Percent	Weighted Percent
1	X	X	X	X	X	X	681	681	92.53	92.53
2	X	X	X	X	X	.	34	34	4.62	4.62
3	X	X	.	.	.	.	13	13	1.77	1.77
4	.	.	.	.	.	.	8	8	1.09	1.09

Missing Data Patterns

Group Means

Group	Sex	Race	Insurance	Drug	Alcohol	Treatment
1	0.566814	1.491924	1.718062	1.292217	1.716593	1.201175
2	0.500000	1.441176	1.588235	1.294118	1.588235	.
3	0.692308	1.230769	.	.	.	.
4	.	.	.	.	.	.

Imputation Summary

Observation Status	Number of Observations	Sum of Weights
Nonmissing	681	681
Missing	55	55
Missing, Imputed	55	55
Missing, Not Imputed	0	0

Some selected observations from the output data set are displayed in Output 119.1.2. The output data set DrugAbuseABB contains the unit identification, the recipient index, and all the variables from the input data set DrugAbuse. Units that are complete respondents have one row, but units that are incomplete respondents have five rows in the output data set. For example, unit 21 is a complete respondent, so it has only one row in the output data set and its ImpIndex value is 0. Unit 22 is an incomplete respondent, so it has five rows in the output data set and its ImpIndex values range from 1 to 5.

Output 119.1.2 Observations for Some Selected Units

UnitID	ImpIndex	Strata	PSU	ObsWeight	ImputationCell	Age	Sex	Race	Insurance	Drug	Alcohol	Treatment
20	0	1	1	5	1	31	0	1	1	1	2	1
21	0	1	2	5	1	23	0	3	3	1	1	1
22	1	1	2	5	2	78	1	1	1	1	1	1
22	2	1	2	5	2	78	1	1	1	1	1	1
22	3	1	2	5	2	78	1	1	2	1	2	1
22	4	1	2	5	2	78	1	1	3	1	2	1
22	5	1	2	5	2	78	1	1	1	1	2	2
23	0	1	2	5	2	29	1	1	1	1	1	1
24	1	1	2	5	2	21	1	2	2	1	2	2
24	2	1	2	5	2	21	1	3	1	1	1	1
24	3	1	2	5	2	21	1	2	2	1	2	1
24	4	1	2	5	2	21	1	1	2	1	2	1
24	5	1	2	5	2	21	1	2	2	2	1	1
25	0	1	2	5	3	85	0	1	3	1	2	1

Suppose you want to perform a logistic regression analysis by using the imputed data set. If you want to use the multiple imputation variance estimator that is available in the MIANALYZE procedure with the imputed data set, then you need to create one complete data set for every imputation. The following SAS statements create five complete data sets and then merge the five data sets into one. Each complete data set contains the complete respondents and only one donor unit for the incomplete respondents. Each data set also contains the imputation number (_Imputation_).

```
data DAIMP;
   set DrugAbuseABB;
   if (ImpIndex = 0) then do;  /* Include complete respondents */
      do _Imputation_=1 to 5;   /* in all imputations.          */
         output;
      end;
   end;
   else do;                     /* Put incomplete respondents   */
      _Imputation_ = ImpIndex; /* in separate imputations.     */
      output;
   end;
proc sort data=DAIMP;
   by _Imputation_ UnitID;
run;
```

The following SAS statements first use the SURVEYLOGISTIC procedure (see Chapter 120, "The SURVEYLOGISTIC Procedure") to perform separate logistic regression analyses within the imputed data sets and use the MIANALYZE procedure (Chapter 83, "The MIANALYZE Procedure") to combine the logistic regression results from five imputed data sets:

```
ods select none;
proc surveylogistic data=DAIMP;
   by _imputation_;
   class Treatment Insurance Sex Race;
   strata Strata;
   cluster PSU;
   weight ObsWeight;
   model Drug=Treatment Insurance Age Sex Race / covb;
   ods output parameterestimates=Estimates covb=Covariances;
run;
ods select all;

proc mianalyze parms(classvar=classval)=Estimates
               covb(effectvar=stacking)=Covariances
               edf=25;
   class Treatment Insurance Sex Race;
   modeleffects Intercept Treatment Insurance Age Sex Race;
   ods output parameterestimates=ABBLogisticAnalysis;
run;
```

Although the survey design information was not directly used in the imputation, you must use the complete design information, including strata, clusters, and weights, to estimate the design variance within each imputed data set. The STRATA, CLUSTER, and WEIGHT statements in PROC SURVEYLOGISTIC specify the design information. However, separate logistic regression results from any single imputed data set should not be used for inference.

Degrees of freedom values for survey data are often much less than the number of observation units. In this

example, there are 736 observation units, but there are 35 PSUs in 10 strata. The degrees of freedom for the Taylor series linearized variance estimator is 25 (35 – 10). You should specify the reduced degrees of freedom by using the EDF= option in PROC MIANALYZE. For more information, see the section "EDF=*number*" on page 6709 in Chapter 83, "The MIANALYZE Procedure"; also see Barnard and Rubin (1999).

The estimated regression parameters and their standard errors from a multiply imputed data set are shown in Output 119.1.3.

Output 119.1.3 Logistic Regression Analysis Using a Multiply Imputed Data Set

The MIANALYZE Procedure

Parameter Estimates (5 Imputations)

Parameter	Treatment	Insurance	Sex	Race	Estimate	Std Error	95% Confidence Limits		DF	Minimum	Maximum
Intercept					0.527080	0.232094	0.04656	1.007603	22.658	0.492600	0.574832
Treatment	1				0.094937	0.153297	-0.22225	0.412121	22.916	0.078459	0.114233
Insurance		1			-0.128475	0.144781	-0.42822	0.171268	22.671	-0.157661	-0.108998
Insurance		2			-0.038353	0.119353	-0.28536	0.208658	22.816	-0.059264	-0.024690
Age					0.001926	0.004635	-0.00767	0.011524	22.577	0.000858	0.002539
Sex			0		-0.088823	0.088702	-0.27265	0.095000	22.279	-0.101860	-0.063631
Race				1	0.436564	0.156414	0.11307	0.760060	23.092	0.418673	0.443484
Race				2	-0.111091	0.195368	-0.51509	0.292904	23.16	-0.118899	-0.096429

Parameter Estimates (5 Imputations)

Parameter	Treatment	Insurance	Sex	Race	Theta0	t for H0: Parameter=Theta0	Pr > |t|
Intercept					0	2.27	0.0330
Treatment	1				0	0.62	0.5418
Insurance		1			0	-0.89	0.3842
Insurance		2			0	-0.32	0.7509
Age					0	0.42	0.6817
Sex			0		0	-1.00	0.3274
Race				1	0	2.79	0.0104
Race				2	0	-0.57	0.5751

Example 119.2: Fully Efficient Fractional Imputation

(View the complete code for this example (svimpex2.sas) in the example repository.)

This example illustrates the fully efficient fractional imputation (FEFI) method by using the data set DrugAbuse from a fictitious survey of drug abusers from Example 119.1. The survey collects information about substance that are used (such as drugs, alcohol, and marijuana) along with insurance information and treatment information. Some participants did not respond to all questions. The data set contains 736 observation units in 35 PSUs and 10 strata. The sum of the weights is 19,600. The data set contains missing values for many variables.

As in Example 119.1, to impute the missing items, you first need to decide whether to impute within imputation cells. Imputation cells divide the data into groups of similar units such that the recipient units share similar characteristics with the donor units in the same group. For example, it is reasonable to believe that different age groups, races, and income categories might have different responses to the drug abuse

survey. You can use these characteristics to create imputation cells. Characteristics for imputation cells might come from the same survey or from other sources such as census data or previous surveys. In this example, assume that the imputation cells are available as a variable called ImputationCell in the data set.

The following statements request that the missing items be imputed by using the FEFI method:

```
proc surveyimpute data=DrugAbuse method=FEFI varmethod=Jackknife;
   class Sex Race Insurance Drug Alcohol Treatment;
   var   Sex Race Insurance Drug Alcohol Treatment;
   cells ImputationCell;
   strata Strata;
   cluster PSU;
   weight ObsWeight;
   output out=DrugAbuseFEFI outjkcoefs=DrugAbuseJKCOEFS;
run;
```

The PROC SURVEYIMPUTE statement invokes the procedure, the DATA= option specifies the input data set DrugAbuse, the METHOD= option requests the FEFI method, and the VARMETHOD= option requests that imputation-adjusted jackknife replicate weights be created. The VAR statement specifies the variables that are to be imputed, and the CELLS statement identifies the imputation cell variable ImputationCell. Because no IMPJOINT statements are specified, all the variables in the VAR statement are to be imputed by using their joint categories. For more information, see the section "IMPJOINT Statement" on page 9914. The STRATA, CLUSTER, and WEIGHT statements specify the strata, cluster, and weight variables. The OUT= option in the OUTPUT statement names the output data set DrugAbuseFEFI to store the imputed values, and the OUTJKCOEFS= option in the OUTPUT statement names the output data set DrugAbuseJKCOEFS to store the jackknife coefficients.

Summary information about the imputation method, number of observations, and survey design is shown in Output 119.2.1. The "Imputation Information" table summarizes the imputation method. The "Number of Observations" table displays the number of observations that are read and used (736) and the weighted number of observation that are read and used (19,600) by PROC SURVEYIMPUTE. The "Design Information" table shows that there are 35 PSUs and 10 strata.

Output 119.2.1 Imputation Information

The SURVEYIMPUTE Procedure

Imputation Information	
Data Set	WORK.DRUGABUSE
Weight Variable	ObsWeight
Stratum Variable	Strata
Cluster Variable	PSU
Imputation Method	FEFI
Replication Method	JACKKNIFE
Number of Replicates	35

Number of Observations Read	736
Number of Observations Used	736
Sum of Weights Read	19599.9937
Sum of Weights Used	19599.9937

Output 119.2.1 *continued*

Design Summary	
Number of Strata	10
Number of Clusters	35

Selected observations for some variables from the output data set are displayed in Output 119.2.2. The output data set, DrugAbuseFEFI, contains unit identification, the recipient index, imputation-adjusted full-sample weights, imputation-adjusted jackknife weights, and all the variables from the input data set DrugAbuse. The output data set contains 35 sets of replicate weights, but only the first three sets of replicate weights are shown in Output 119.2.2. Units that are complete respondents have one row, but units that are incomplete respondents have multiple rows in the output data set. For example, unit 21 is a complete respondent, so it has only one row in the output data set and its ImpIndex value is 0. Unit 22 is an incomplete respondent; it has 20 rows in the output data set, its ImpIndex values range from 1 to 20, and its imputation-adjusted full-sample weights (ImpWt) range from 0.02 to 1.02. The sum of ImpWt for all rows (donor cells) for observation unit 22 is 5, which is the full sample weight for unit 22.

Output 119.2.2 Observations for Selected Units

UnitID	ImpIndex	ImpWt	Strata	PSU	ObsWeight	ImputationCell	Sex	Race	Insurance	Drug
20	0	5.00000	1	1	5	1	0	1	1	1
21	0	5.00000	1	2	5	1	0	3	3	1
22	1	0.49238	1	2	5	2	1	1	1	1
22	2	1.01597	1	2	5	2	1	1	1	1
22	3	0.14661	1	2	5	2	1	1	1	1
22	4	0.11496	1	2	5	2	1	1	1	2
22	5	0.03894	1	2	5	2	1	1	1	2
22	6	0.32730	1	2	5	2	1	1	1	2
22	7	0.21661	1	2	5	2	1	1	1	2
22	8	0.24024	1	2	5	2	1	1	2	1
22	9	0.10453	1	2	5	2	1	1	2	1
22	10	0.46726	1	2	5	2	1	1	2	1
22	11	0.19382	1	2	5	2	1	1	2	1
22	12	0.37531	1	2	5	2	1	1	2	2
22	13	0.03697	1	2	5	2	1	1	2	2
22	14	0.15533	1	2	5	2	1	1	3	1
22	15	0.08672	1	2	5	2	1	1	3	1
22	16	0.69416	1	2	5	2	1	1	3	1
22	17	0.03688	1	2	5	2	1	1	3	1
22	18	0.02106	1	2	5	2	1	1	3	2
22	19	0.18442	1	2	5	2	1	1	3	2
22	20	0.05053	1	2	5	2	1	1	3	2
23	0	5.00000	1	2	5	2	1	1	1	1

Alcohol	Treatment	ImpRepWt_1	ImpRepWt_2	ImpRepWt_3
2	1	0.00000	6.66667	6.66667
1	1	6.66667	0.00000	6.66667
1	1	0.65757	0.00000	0.66513
2	1	1.36836	0.00000	1.33813
2	2	0.18561	0.00000	0.19845
1	1	0.15441	0.00000	0.15268
1	2	0.05231	0.00000	0.05172
2	1	0.44358	0.00000	0.43861
2	2	0.29074	0.00000	0.28749
1	1	0.32268	0.00000	0.31907
1	2	0.12908	0.00000	0.14255
2	1	0.61595	0.00000	0.62442
2	2	0.26067	0.00000	0.25730
2	1	0.50409	0.00000	0.49845
2	2	0.04966	0.00000	0.04911
1	1	0.19731	0.00000	0.21002
1	2	0.11647	0.00000	0.11517
2	1	0.93613	0.00000	0.92566
2	2	0.04953	0.00000	0.04897
1	1	0.02828	0.00000	0.02796
2	1	0.23638	0.00000	0.24865
2	2	0.06788	0.00000	0.06712
1	1	6.66667	0.00000	6.66667

You can use the imputed data set and the imputation-adjusted replicate weights to compute any estimators from your imputed data. You can use the REPWEIGHTS statement in any SAS/STAT survey analysis procedures to specify the imputation-adjusted replicate weights. For example, the following statements use PROC SURVEYLOGISTIC to perform logistic regression analysis of the imputed data:

```
proc surveylogistic data=DrugAbuseFEFI varmethod=Jackknife;
   class Treatment Insurance Sex Race;
   model Drug=Treatment Insurance Age Sex Race;
   weight ImpWt;
   repweights ImpRepWt_: / jkcoefs=DrugAbuseJKCOEFS;
   ods output parameterestimates=FEFILogisticAnalysis;
run;
```

The WEIGHT statement specifies the imputation-adjusted full-sample weight (ImpWt), and the REPWEIGHTS statement specifies the imputation-adjusted replicate weights (ImpRepWt_1, . . . , ImpRepWt_35). The JKCOEFS= option in the REPWEIGHTS statement specifies the jackknife coefficients.

The parameter estimates and their standard errors are displayed in Output 119.2.3. The variance estimators correctly account for both the design variability and the imputation variability.

Output 119.2.3 Logistic Regression Analysis of the Fractionally Imputed Data Set

The SURVEYLOGISTIC Procedure

Analysis of Maximum Likelihood Estimates

Parameter		Estimate	Standard Error	t Value	Pr > |t|
Intercept		0.5105	0.2281	2.24	0.0317
Treatment	1	0.1048	0.1600	0.65	0.5168
Insurance	1	-0.1152	0.1454	-0.79	0.4337
Insurance	2	-0.0461	0.1183	-0.39	0.6988
Age		0.00195	0.00459	0.42	0.6736
Sex	0	-0.0719	0.0931	-0.77	0.4452
Race	1	0.4463	0.1574	2.84	0.0075
Race	2	-0.1212	0.2110	-0.57	0.5695

NOTE:
The degrees of freedom for the t tests is 35.

Example 119.3: Fully Efficient Fractional Imputation, Fay's Balanced Repeated Replication, and Domain Analysis

(View the complete code for this example (svimpex3.sas) in the example repository.)

This example demonstrates the FEFI method by using data from the third National Health and Nutrition Examination Survey (NHANES III). The data set contains a set of BRR replicate weights. The REPWEIGHTS statement in PROC SURVEYIMPUTE is used to create imputation-adjusted replicate weights. The imputed data set and the imputation-adjusted replicate weights are then used in PROC SURVEYFREQ to create crosstabulation tables and to perform domain analysis.

The objective of NHANES is to study the health and nutritional status of the US population. NHANES uses a multistage stratified area sample with typically two PSUs per stratum. Strata are created based on

geographic location, Metropolitan Statistical Areas (MSAs), and other demographics. An MSA or a group of counties are selected as PSUs from each stratum. Sampling weights are unequal because of different selection probabilities among different subgroups and for reasons such as nonresponse and undercoverage. For more information about NHANES, see http://www.cdc.gov/nchs/nhanes/about_nhanes.htm.

NHANES III data contain missing values in many items. Multiple imputation was used to impute some of the missing items. Five multiply imputed data sets are available for public use. Because FEFI will be used in this example to impute the missing values, you need the observed data, the missing (or imputation) flag for every item, and only one imputed data set. The data sets Core and IMP1 have been downloaded from the Centers for Disease Control and Prevention's website (https://www.cdc.gov/). The Core data set contains the demographic variables, full sample weights, replicate weights, and imputation flags. The replicate weights are created by using Fay's BRR method with a Fay coefficient of 0.3. The IMP1 data set contains the first version of the five multiply imputed data sets.

For this example, a new data set, Smoke, is created by merging the Core and IMP1 data sets by the observation sequence number, SEQN. The Smoke data set contains the following items:

- SEQN: observation sequence number
- WTPFQX6: observation weight
- WTPQRP1 to WTPQRP52: 52 replicate weights from the BRR method
- DMARETHN: race-ethnicity; 1 for white, 2 for black, 3 for Mexican American, and 4 for other
- HSSEX: gender; 1 for male and 2 for female
- HFF1IF: imputation flag for HFF1MI; 1 for observed and 2 for imputed
- HAN6SRIF: imputation flag for HAN6SRMI; 0 for not applicable, 1 for observed, and 2 for imputed
- HAR3RIF: imputation flag for HAR3RMI; 0 for not applicable, 1 for observed, and 2 for imputed
- HAT28IF: imputation flag for HAT28MI; 0 for not applicable, 1 for observed, and 2 for imputed
- HFF1MI: anyone smokes cigarettes in the home; 1 for yes, and 2 for no
- HAN6SRMI: beer, wine, or liquor per month; –9 for not applicable, 1 for 0 time in the past month, 2 for 1 to 10 times in the past month, and 3 for more than 10 times in the past month
- HAR3RMI: smoke cigarettes now; –9 for not applicable, 1 for yes, and 2 for no
- HAT28MI: activity level compared to others; –9 for not applicable, 1 for more active, 2 for less active, and 3 for about the same
- Education: highest education attained; levels are elementary, high school, college, and unknown

For donor-based imputation methods, auxiliary information is used to create imputation cells. Imputation cells divide the data into groups of similar units such that the recipient units share similar characteristics with the donor units in the same group. Characteristics for imputation cells might come from the same survey or from other auxiliary sources such as census data or previous surveys. The cell identification is known for every unit in the sample. Categorical levels of auxiliary variables are often used to create imputation cells. For a helpful review, see Brick and Kalton (1996). For the purpose of this example, seven imputation

cells were created by using only two demographic variables: race-ethnicity status (DMARETHN) and gender (HSSEX). Both variables are available in the Core data set, and both have no missing values. The imputation cells are identified by the variable ImputationCells in the Smoke data set.

The following DATA step creates the imputation cells and the variable Education, and replaces the multiply imputed values with missing values:

```
*---Create education levels and imputation cells, ---;
*---and assign . to missing items                 ---;
data Smoke; set Smoke;
   if HFA7 <=8                 then Education='Elementary ';
   if HFA7 > 8  and HFA7 <= 12 then Education='High School';
   if HFA7 > 12 and HFA7 <= 17 then Education='College    ';
   if HFA7 > 17                then Education='Unknown    ';
   if DMARETHN = 1 & HSSEX = 1 then ImputationCells=1;
   if DMARETHN = 1 & HSSEX = 2 then ImputationCells=2;
   if DMARETHN = 2 & HSSEX = 1 then ImputationCells=3;
   if DMARETHN = 2 & HSSEX = 2 then ImputationCells=4;
   if DMARETHN = 3 & HSSEX = 1 then ImputationCells=5;
   if DMARETHN = 3 & HSSEX = 2 then ImputationCells=6;
   if DMARETHN = 4             then ImputationCells=7;
   if HFF1IF   = 2 then HFF1MI   = .;
   if HAN6SRIF = 2 then HAN6SRMI = .;
   if HAR3RIF  = 2 then HAR3RMI  = .;
   if HAT28IF  = 2 then HAT28MI  = .;
run;
```

The following statements request that the missing values be imputed by using the FEFI method:

```
proc surveyimpute data=Smoke method=FEFI varmethod=BRR;
   weight wtpfqx6;
   repweights wtpqrp:;
   id seqn;
   class hff1mi han6srmi har3rmi hat28mi;
   var   hff1mi han6srmi har3rmi hat28mi;
   cells ImputationCells;
   output out=SmokeImputed;
run;
```

The PROC SURVEYIMPUTE statement invokes the procedure, the DATA= option specifies the input data set Smoke, the METHOD= option requests the FEFI method, and the VARMETHOD= option requests the imputation-adjusted BRR replication weights. The WEIGHT statement specifies the weight variable, and the REPWEIGHTS statement specifies the unadjusted BRR replicate weights. Because you specify replicate weights by using the REPWEIGHTS statement, you do not need to specify the Fay coefficient in PROC SURVEYIMPUTE. The variable SEQN in the ID statement identifies the observation units. The VAR statement specifies the variables to be imputed, the CELLS statement identifies the imputation cell variable ImputationCells, and the OUT= option in the OUTPUT statement names the output data set SmokeImputed. You request that all four variables be imputed jointly and that the imputed data be saved in the SmokeImputed data set.

Note that this example creates imputation-adjusted BRR replicate weights from the unadjusted BRR replicate weights that are available for these data. If the unadjusted BRR replicate weights are not available to you, then PROC SURVEYIMPUTE first creates the unadjusted BRR replicate weights and then updates the unadjusted

weights for imputation to create the imputation-adjusted BRR replicate weights. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 9955.

Summary information about the number of observations and class level information are shown in Output 119.3.1. The "Number of Observations" table displays the number of observations (33,994) that are read and used. The weighted number of observations that are read shows that the 33,994 observation units in the sample represent over 251,000,000 observation units in the population. The "Class Level Information" table displays the observed levels for the analysis variables. The "Missing Data Patterns" table shows an arbitrary missing pattern. There are 12 different missing pattern groups. An "X" denotes that the variable is observed in that group, and a "." denotes that the variable is missing. Almost 94% of the observation units have no missing values (Group 1), 4.5% of the observation units have missing values for the variable HAN6SRMI (Group 4), and 1% of the observation units have missing values for the variable HAT28MI (Group 2).

Output 119.3.1 Imputation Information

The SURVEYIMPUTE Procedure

Number of Observations Read	33994
Number of Observations Used	33994
Sum of Weights Read	251097002
Sum of Weights Used	251097002

Class Level Information

Class	Levels	Values
HFF1MI	2	1 2
HAN6SRMI	4	-9 1 2 3
HAR3RMI	3	-9 1 2
HAT28MI	4	-9 1 2 3

Output 119.3.2 Missing Data Patterns

Missing Data Patterns											
									Group Means		
Group	HFF1MI	HAN6SRMI	HAR3RMI	HAT28MI	Freq	Sum of Weights	Unweighted Percent	Weighted Percent	HFF1MI 1	HFF1MI 2	HAN6SRMI -9
1	X	X	X	X	31916	229183908	93.89	91.27	0.370883	0.629117	0.275976
2	X	X	X	.	383	3584032.5	1.13	1.43	0.312278	0.687722	0
3	X	X	.	X	2	6696.18	0.01	0.00	0	1.000000	0
4	X	.	X	X	1536	17201675.8	4.52	6.85	0.417227	0.582773	.
5	X	.	X	.	25	255366.75	0.07	0.10	0.512251	0.487749	.
6	X	.	.	.	1	1137.27	0.00	0.00	0	1.000000	.
7	.	X	X	X	106	723161.98	0.31	0.29	.	.	0.277402
8	.	X	X	.	5	21175.81	0.01	0.01	.	.	0
9	.	X	.	.	2	43867.66	0.01	0.02	.	.	0
10	.	.	X	X	4	11751.6	0.01	0.00	.	.	.
11	.	.	X	.	1	4483.45	0.00	0.00	.	.	.
12	.	.	.	.	13	59745.61	0.04	0.02	.	.	.

Missing Data Patterns										
				Group Means						
Group	HAN6SRMI 1	HAN6SRMI 2	HAN6SRMI 3	HAR3RMI -9	HAR3RMI 1	HAR3RMI 2	HAT28MI -9	HAT28MI 1	HAT28MI 2	HAT28MI 3
1	0.365368	0.219609	0.139048	0.275976	0.199395	0.524629	0.275976	0.239253	0.160019	0.324752
2	0.533112	0.322946	0.143942	0	0.231400	0.768600	.	.	.	.
3	0	0.695886	0.304114	.	.	.	0	0.695886	0	0.304114
4	.	.	.	0	0.362741	0.637259	0	0.339890	0.222725	0.437384
5	.	.	.	0	0.371613	0.628387	.	.	.	.
6	.	.	.	.	.	.	.	.	.	.
7	0.570801	0.099198	0.052599	0.277402	0.156784	0.565813	0.277402	0.185712	0.192402	0.344483
8	0.414365	0.585635	0	0	0.078044	0.921956	.	.	.	.
9	0.262239	0.737761	0	.	.	.	.	.	.	.
10	.	.	.	0	0	1.000000	0	0.664584	0	0.335416
11	.	.	.	0	0	1.000000	.	.	.	.
12	.	.	.	.	.	.	.	.	.	.

The "Iteration History" table shown in Output 119.3.3 displays the maximum absolute and relative differences of the fractional weights for the EM algorithm for the full sample. The algorithm converged after four iterations. The "Imputation Summary" table shown in Output 119.3.4 displays the number of observed units (31,961), the number of missing units (2,078), and the number of imputed units. All units that have missing values have been imputed.

Output 119.3.3 Iteration History for the EM

Iteration History		
Iteration Number	Maximum Absolute Difference	Maximum Relative Difference
1	830.4733	0.18278
2	93.33655	0.00904
3	16.14668	0.00237
4	4.138731	0.00061

Output 119.3.4 Imputation Summary

Imputation Summary		
Observation Status	Number of Observations	Sum of Weights
Nonmissing	31916	229183908
Missing	2078	21913094.6
Missing, Imputed	2078	21913094.6
Missing, Not Imputed	0	0

The imputed data set SmokeImputed contains the imputation-adjusted weight (ImpWt) and 52 imputation-adjusted replicate weights (ImpRepWt_1 to ImpRepWt_52). The SmokeImputed data set has 38,701 data lines. The number of imputed values for an observation unit ranges from two to six, but around 80% of the units are imputed by using two or three imputed values.

You can use the imputed data set, the imputation-adjusted replicate weights, and the appropriate Fay coefficient to compute any estimators from your imputed data. You should use the REPWEIGHTS statement in SAS/STAT survey analysis procedures to specify the imputation-adjusted replicate weights. This example uses PROC SURVEYFREQ to perform the following analyses:

- estimate the percentage of smokers and nonsmokers in the population
- describe the smoking habits of an individual and of anyone who smokes in the home
- perform a domain analysis of activity levels for different levels of education

The PROC SURVEYFREQ statement invokes the procedure, the DATA= option names the imputed data set SmokeImputed, and the VARMETHOD=option requests the BRR variance estimation. The FAY= option for VARMETHOD=BRR specifies the Fay coefficient 0.3. Because your replicate weights come from Fay's BRR method, you must specify the FAY= option in the SAS/STAT survey analysis procedures to appropriately estimate the variance. The VARHEADER=LABEL option in the PROC SURVEYFREQ statement requests that the labels of the variables be displayed in the output. The WEIGHT statement specifies the imputation-adjusted full sample weights, and the REPWEIGHTS statement specifies the imputation-adjusted replicate weights. Note that the imputation-adjusted full sample and replicate weights are created by PROC SURVEYIMPUTE, and they are different from the unadjusted weights available in the Smoke data set. The first TABLES statement requests a two-way frequency analysis for HFF1MI and HAR3RMI. The second TABLES statement requests a domain analysis for HAT28MI, where the variable Education is used as the domain variable. The ROW option in the TABLES statement is required in order to compute the

distribution of HAT28MI for different levels of Education. The NOTOTAL, NOFREQ, and NOWT options suppress some output columns.

```
proc surveyfreq data=SmokeImputed varmethod=brr(fay=0.3) varheader=label;
   weight ImpWt;
   repweights ImpRepWt_:;
   table HFF1MI*HAR3RMI;
   table Education*HAT28MI / row nototal nofreq nowt;
run;
```

The data summary and the variance estimation information are displayed in Output 119.3.5. There are 38,701 data lines in the SmokeImputed data set. These 38,701 data lines represent the 33,994 observation units in the Smoke data set. The observation units are identified by the variable SEQN. The sum of weights is over 251,000,000, which is the same as the sum of weights in the Smoke data set. The sum of weights is an estimate of the population size. The "Variance Estimation" table shows that 52 replicate weights from Fay's BRR method are used for variance estimation with the Fay coefficient 0.3.

Output 119.3.5 Summary Information

The SURVEYFREQ Procedure

Data Summary	
Number of Observations	38701
Sum of Weights	251097002

Variance Estimation	
Method	BRR
Replicate Weights	SMOKEIMPUTED
Number of Replicates	52
Fay Coefficient	0.300

A two-way table for the smoking habit of the observation unit and smoking in the home is shown in Output 119.3.6. There are 21% smokers and 54% nonsmokers in the population. Nearly 19% of the individuals are smokers and live in a home where at least one person smokes in the home, but only 2% of the individuals are smokers and live in a home where no other household member smokes in the home. However, almost 9% of the individuals are nonsmokers but live in a home where at least one household member smokes in the home. The standard errors that are reported in the table properly account for the imputation.

Output 119.3.6 Two-Way Table for Smoking Status

Table of Anyone living here smoke cigs in home by Smoke cigarettes now (recode)						
Anyone living here smoke cigs in home	**Smoke cigarettes now (recode)**	**Frequency**	**Weighted Frequency**	**Std Err of Wgt Freq**	**Percent**	**Std Err of Percent**
1	**-9**	5431	24762830	717089	9.8619	0.2856
	1	5788	47166758	1361381	18.7843	0.5422
	2	3341	21790932	614088	8.6783	0.2446
	Total	14560	93720520	2180565	37.3244	0.8684
2	**-9**	8582	38702422	701855	15.4133	0.2795
	1	881	5837874	397358	2.3249	0.1582
	2	14678	112836186	1602093	44.9373	0.6380
	Total	24141	157376482	2180565	62.6756	0.8684
Total	**-9**	14013	63465252	260068	25.2752	0.1036
	1	6669	53004633	1276926	21.1092	0.5085
	2	18019	134627118	1328833	53.6156	0.5292
	Total	38701	251097002	2.25270	100.0000	

Suppose you want to perform a domain analysis by using the imputed data. If a list of domain variables is available before the imputation, then sometimes it is desirable to use the domain variables to create the imputation cells. However, requests for domain analyses often come after the imputation. In addition, data users might use domain variables that are different from what are used to create the imputation cells. In this example, the domain variable Education was not used to create the imputation cells. Although education level is not used in the imputation, it is reasonable to use the imputed data to perform domain analysis for every level of education. Domain analysis for activity levels for different education levels is shown in Output 119.3.7. If the highest education level is college, then 38% are reported as more active and 21% are reported as less active than their peers. If the highest education level is high school, then 28% are reported as more active and 20% are reported as less active than their peers. The standard errors that are reported in the table properly account for the imputation.

Output 119.3.7 Domain Analysis for Activity Levels by Education

Table of Education by Compare own activity level to others					
Education	Compare own activity level to others	Percent	Std Err of Percent	Row Percent	Std Err of Row Percent
College	**-9**	0.0017	0.0016	0.0055	0.0052
	1	11.9908	0.4117	38.3446	0.9792
	2	6.5292	0.3253	20.8795	0.8461
	3	12.7494	0.4641	40.7704	0.9978
Elementary	**-9**	21.8961	0.1269	74.2986	0.9154
	1	1.8571	0.1193	6.3015	0.3596
	2	2.0051	0.1500	6.8039	0.4513
	3	3.7121	0.1970	12.5961	0.5420
High School	**-9**	3.2191	0.1238	8.3653	0.3251
	1	10.6977	0.2990	27.7997	0.6634
	2	7.8486	0.2782	20.3959	0.6333
	3	16.7160	0.4958	43.4391	0.7062
Unknown	**-9**	0.1583	0.0289	20.3674	3.2674
	1	0.1981	0.0463	25.4966	3.7217
	2	0.1502	0.0346	19.3311	3.3973
	3	0.2704	0.0426	34.8049	3.2211

Example 119.4: Fractional Hot-Deck Imputation for Mixed Variables

(View the complete code for this example (svimpex4.sas) in the example repository.)

This example imputes missing values in both categorical and continuous variables by applying the fractional hot-deck imputation (FHDI) method to a data set from the third National Health and Nutrition Examination Survey (NHANES III). The data set contains a set of balanced repeated replication (BRR) replicate weights. The REPWEIGHTS statement in PROC SURVEYIMPUTE accepts the BRR weights and creates imputation-adjusted replicate weights. The imputed data set and the imputation-adjusted replicate weights are then used in PROC SURVEYMEANS and PROC SURVEYREG to perform domain analysis and regression analysis, respectively.

The objective of NHANES is to study the health and nutritional status of the US population. NHANES uses a multistage stratified area sample that typically has two PSUs per stratum. Strata are created on the basis of geographic location, metropolitan statistical area (MSA), and other demographic information. MSAs or a group of counties are selected as PSUs from each stratum. Sampling weights are unequal because of different selection probabilities among different subgroups and for reasons such as nonresponse and undercoverage. For more information about NHANES, see `http://www.cdc.gov/nchs/nhanes/about_nhanes.htm`.

NHANES III data contain missing values in many items. Multiple imputation was used to impute some of the missing items. Five multiply imputed data sets are available for public use. Because FHDI is used in this example to impute the missing values, you need the observed data, the missing (or imputation) flag for every item, and only one imputed data set. The data sets Core and IMP1 have been downloaded from the Centers for Disease Control and Prevention's website (`https://www.cdc.gov/`). The Core data set contains the demographic variables, full sample weights, replicate weights, and imputation flags. The replicate weights are created by using Fay's BRR method, with a Fay coefficient of 0.3. The IMP1 data set contains the first version of the five multiply imputed data sets.

The data set HealthMiss is obtained by merging the Core and IMP1 data sets by using the observation sequence number SEQN. The HealthMiss data set contains observation units that are between 17 and 60 years of age. Missing values are added according to the imputation flag in the Core data set. The following items are available in the HealthMiss data set for each observation unit:

- SEQN: observation sequence number
- WTPFQX6: observation weight, ranging from 220.18 to 140916.28
- WTPQRP1 to WTPQRP52: 52 replicate weights from the BRR method
- HSSEX: gender; 1 for male and 2 for female
- HFF1MI: whether anyone smokes cigarettes in the home; 1 for yes and 2 for no
- HAT28MI: activity level compared to others; –9 for not applicable, 1 for more active, 2 for less active, and 3 for about the same
- BMPHTMI: standing height (cm), ranging from 130.6 to 206.5
- BMPWTMI: body weight (kg), ranging from 26.75 to 241.80
- PEP6G3MI: K5 diastolic blood pressure (mmHg), ranging from 0 to 136
- HSAGEIR: age in years, ranging from 17 to 60
- HSHSIZER: household size; categories from 1 to 10
- Married: marital status; 1 for married and 0 for not married

There are no missing values in the variables SEQN, HSAGEIR, HSHSIZER, Married, and WTPFQX6 or in the replicate weight variables. The variables HFF1MI, HAT28MI, BMPHTMI, BMPWTMI, and PEP6G3MI contain missing values, which are imputed in this example. Although HFF1MI and HAT28MI have two and four observed levels, respectively, BMPHTMI, BMPWTMI, and PEP6G3MI have many observed levels. Because these last three variables have many observed levels, FHDI is applied to impute missing values jointly in all five variables.

Before you apply the FHDI method to a data set, you should (1) incorporate auxiliary information in the imputation by creating imputation cells and (2) create bins for variables that have many levels.

Imputation cells divide the data into groups of similar units such that the recipient units have characteristics similar to those of the donor units in the same group. Characteristics of imputation cells might come from the same survey or from other sources, such as census data or previous surveys. The cell identification is known for every unit in the sample. For a helpful review of handling missing data, see Brick and Kalton (1996). For the purpose of this example, a cluster variable is created by using two demographic variables, HSAGEIR and HSHSIZER, and by using the FASTCLUS procedure in SAS/STAT. Both variables are available in the Core data set, and they do not contain missing values. These clusters are identified by the variable Cluster in the HealthMiss data set. Levels of the variables Cluster and Married are used to create imputation cells.

If you request FHDI, then the variables that have many levels (these are identified by the variables that you specify in the VAR statement but not in the CLASS statement) are first levelized to create bins. You can use the CLEVVAR= option to specify the variable that contains the bins for a numeric variable. Alternatively, you use the CLEVELS=*k* option to divide the observed range of the numeric variable into *k* equally spaced

bins. First-stage FEFI is applied to the CLEVVAR= variables and to the variables that you specify in the CLASS statement. In this example, the bins are created for a numeric variable by using the observed 33rd and 66th percentiles for that variable. The CLEVVAR= variable should contain a missing value for every observation unit in which the corresponding numeric variable has a missing value. The following statements create the CLEVVAR= variables:

```
*---Create bins for continuous variables---;
data HealthMiss; set HealthMiss;
   if bmphtmi       = .     then bmphtlev=.;
   else if bmphtmi <= 162.6 then bmphtlev=1;
   else if bmphtmi <= 171.5 then bmphtlev=2;
   else                          bmphtlev=3;
   if bmpwtmi       = .    then bmpwtlev=.;
   else if bmpwtmi <= 65.7 then bmpwtlev=1;
   else if bmpwtmi <= 80.2 then bmpwtlev=2;
   else                         bmpwtlev=3;
   if pep6g3mi       = .    then pep6g3lev=.;
   else if pep6g3mi <= 65.3 then pep6g3lev=1;
   else if pep6g3mi <= 75.8 then pep6g3lev=2;
   else                          pep6g3lev=4;
   label bmphtlev  = "Bin values for BMPHTMI"
         bmpwtlev  = "Bin values for BMPWTMI"
         pep6g3lev = "Bin values for PEP6G3MI";
run;
```

The following statements use the FHDI method to impute the missing values:

```
proc surveyimpute data=HealthMiss method=fhdi varmethod=brr
                      ndonors=5 seed=9388401;
   id seqn;
   class hff1mi hat28mi;
   weight wtpfqx6;
   repweights wtpqrp:;
   cells cluster married;
   var hff1mi hat28mi bmphtmi (clevvar=bmphtlev)
                      bmpwtmi (clevvar=bmpwtlev)
                      pep6g3mi(clevvar=pep6g3lev);
   output out=HealthFHDI;
run;
```

The PROC SURVEYIMPUTE statement invokes the procedure, the DATA= option specifies the input data set HealthMiss, the METHOD= option requests the FHDI method, the VARMETHOD= option requests the imputation-adjusted BRR replication weights, the NDONORS= option specifies the maximum number of second-stage donors, and the SEED= option specifies the random number generator seed. The variable SEQN in the ID statement identifies the observation units. The WEIGHT statement identifies the weight variable, and the REPWEIGHTS statement identifies the variables that contain the unadjusted BRR replicate weights. The CELLS statement identifies the imputation cell variables Cluster and Married, and the OUT= option in the OUTPUT statement names the output data set HealthFHDI.

The VAR statement specifies the variables in which the missing values are to be imputed. The CLASS statement identifies the categorical variables. Only first-stage FEFI will be applied to the CLASS variables. Second-stage FEFI and FHDI will be applied to the variables that are specified in the VAR statement but not in the CLASS statement. The CLEVVAR= option for these variables identifies the corresponding bin

variables in which first-stage FEFI will be performed. In this example, you requested first-stage FEFI for the variables HFF1MI, HAT28MI, BMPHTLEV, BMPWTLEV, and PEP6G3LEV; and second-stage FEFI and FHDI for the variables BMPHTMI, BMPWTMI, and PEP6G3MI.

You request that all five variables be imputed jointly and that the imputed data be saved in the HealthFHDI data set.

The number of observations and the CLASS level information are shown in Output 119.4.1. The Sum of Weights Read row shows that the 13,721 observation units in the sample represent over 149 million observation units in the population. The "Class Level Information" table displays the observed levels for the CLASS variables.

Output 119.4.1 Imputation Information

The SURVEYIMPUTE Procedure

Number of Observations Read	13721
Number of Observations Used	13721
Sum of Weights Read	149546400
Sum of Weights Used	149546400

Class Level Information

Class	Levels	Values
HFF1MI	2	1 2
HAT28MI	3	1 2 3

The "Missing Data Patterns" table shows an arbitrary missing pattern. There are 13 different missing pattern groups. An "X" denotes that the variable is observed in that group, and a "." denotes that the variable is missing. Almost 87.42% of the observation units have no missing values (Group 1); 7.17% of the observation units have missing values for the variables BMPHTMI, BMPWTMI, and PEP6G3MI (Group 6); and 0.03% of the observation units have missing values in all five variables (Group 13).

Output 119.4.2 Missing Data Patterns

Missing Data Patterns

Group	HFF1MI	HAT28MI	BMPHTMI	BMPWTMI	PEP6G3MI	Freq	Sum of Weights	Unweighted Percent	Weighted Percent
1	X	X	X	X	X	11995	129773812	87.42	86.78
2	X	X	X	X	.	475	3493749.6	3.46	2.34
3	X	X	X	.	X	10	16064.94	0.07	0.01
4	X	X	X	.	.	1	12796.75	0.01	0.01
5	X	X	.	.	X	6	49022.69	0.04	0.03
6	X	X	.	.	.	984	13080725.9	7.17	8.75
7	X	.	X	X	X	167	2193493.99	1.22	1.47
8	X	.	X	X	.	11	138241.25	0.08	0.09
9	X	.	.	X	X	1	2742.56	0.01	0.00
10	X	.	.	.	.	20	307483.94	0.15	0.21
11	.	X	X	X	X	15	42007.38	0.11	0.03
12	.	X	.	.	.	32	390724.57	0.23	0.26
13	.	.	.	.	.	4	45534.74	0.03	0.03

Missing Data Patterns

Group Means

Group	BMPHTMI	BMPWTMI	PEP6G3MI	HFF1MI 1	HFF1MI 2	HAT28MI 1	HAT28MI 2	HAT28MI 3
1	169.387674	75.258624	71.746316	0.403270	0.596730	0.310384	0.231793	0.457823
2	168.189385	79.263271	.	0.464648	0.535352	0.301674	0.252490	0.445836
3	154.554452	.	53.348310	0.370184	0.629816	0	0.164085	0.835915
4	156.500000	.	.	0	1.000000	0	0	1.000000
5	.	.	87.096205	0.870574	0.129426	0	0.468602	0.531398
6	.	.	.	0.402061	0.597939	0.336804	0.214523	0.448674
7	168.361303	76.971926	72.511286	0.322006	0.677994	.	.	.
8	168.749387	79.516017	.	0.470061	0.529939	.	.	.
9	.	70.450000	84.000000	0	1.000000	.	.	.
10	.	.	.	0.483659	0.516341	.	.	.
11	166.962303	84.023762	75.389658	.	.	0.158780	0.312686	0.528534
12	.	.	.	.	.	0.255939	0.237827	0.506235
13	.	.	.	.	.	.	.	.

The "Imputation Summary" table in Output 119.4.3 displays the number of observation units that contain no missing items (11,995), the number of observation units that contain at least one missing item (1,726), and the number of units in which the missing values are imputed. Missing values in all 1,726 units are imputed.

Output 119.4.3 Imputation Summary

Imputation Summary

Observation Status	Number of Observations	Sum of Weights
Nonmissing	11995	129773812
Missing	1726	19772588.3
Missing, Imputed	1726	19772588.3
Missing, Not Imputed	0	0

Because fractional imputation replaces one observed unit with several observation rows that contain imputed values, the 13,721 observed units in the input data set HealthMiss generate 127,002 observation rows in the imputed data set HealthFHDI. The following note displays the number of observation rows (127,007) in this imputed data set:

```
NOTE: The data set WORK.HEALTHFHDI has 127007 observations and 134 variables.
```

You can use the imputed data set, the imputation-adjusted replicate weights, and the appropriate Fay coefficient to compute any estimators from your imputed data. However, you must use the REPWEIGHTS statement in SAS/STAT survey analysis procedures to specify the imputation-adjusted replicate weights. The following two examples describe a domain analysis and a regression analysis that use the imputed data.

The following PROC SURVEYMEANS statements estimate the mean diastolic blood pressure in the year 2000 population and in the subpopulation of smokers and nonsmokers:

```
ods graphics on;
proc surveymeans data=HealthFHDI varmethod=brr(Fay=0.3) plots=domain;
   weight ImpWt;
   repweights ImpRepWt_:;
   var pep6g3mi;
   domain hff1mi;
run;
```

The "Data Summary" table in Output 119.4.4 displays the number of observation rows (127,007) and the sum of weights (149,546,400). Because fractional imputation is used, the number of observation rows is not equal to the number of observation units (13,721). However, the sum of weights from the observation rows, which is an estimate of the population size, is the same as the sum of weights from the observation units. The "Variance Estimation" table in Output 119.4.4 shows that Fay's BRR with 52 replicate weights and a Fay coefficient of 0.3 are used for variance estimation.

Output 119.4.4 Data Summary

The SURVEYMEANS Procedure

Data Summary	
Number of Observations	127007
Sum of Weights	149546400

Variance Estimation	
Method	BRR
Replicate Weights	HEALTHFHDI
Number of Replicates	52
Fay Coefficient	0.3

The first table in Output 119.4.5 displays the mean diastolic blood pressure for the overall population as 71.78, with a standard error of 1.34. The second table shows data for smokers and nonsmokers. For smokers, the mean diastolic blood pressure is 70.67, with a standard error of 1.63. For nonsmokers, it is 72.55, with a standard error of 1.45. The N column in both tables displays the number of observation rows, not the number of observation units.

Output 119.4.5 Mean Diastolic Blood Pressure

Statistics						
Variable	Label	N	Mean	Std Error of Mean	95% CL for Mean	
PEP6G3MI	K5, diastolic, for 1st BP (mmHg)	127007	71.788404	0.268699	71.2492203	72.3275871

The SURVEYMEANS Procedure

Statistics for HFF1MI Domains							
HFF1MI	Variable	Label	N	Mean	Std Error of Mean	95% CL for Mean	
1	PEP6G3MI	K5, diastolic, for 1st BP (mmHg)	49876	70.665325	0.326577	70.0100015	71.3206493
2	PEP6G3MI	K5, diastolic, for 1st BP (mmHg)	77131	72.548913	0.289446	71.9680973	73.1297289

A box plot of the weighted distribution of diastolic blood pressure is displayed in Output 119.4.6. The first box is for the overall population, and the other two boxes are for the two domains defined by smoking habits.

Output 119.4.6 Diastolic Blood Pressure

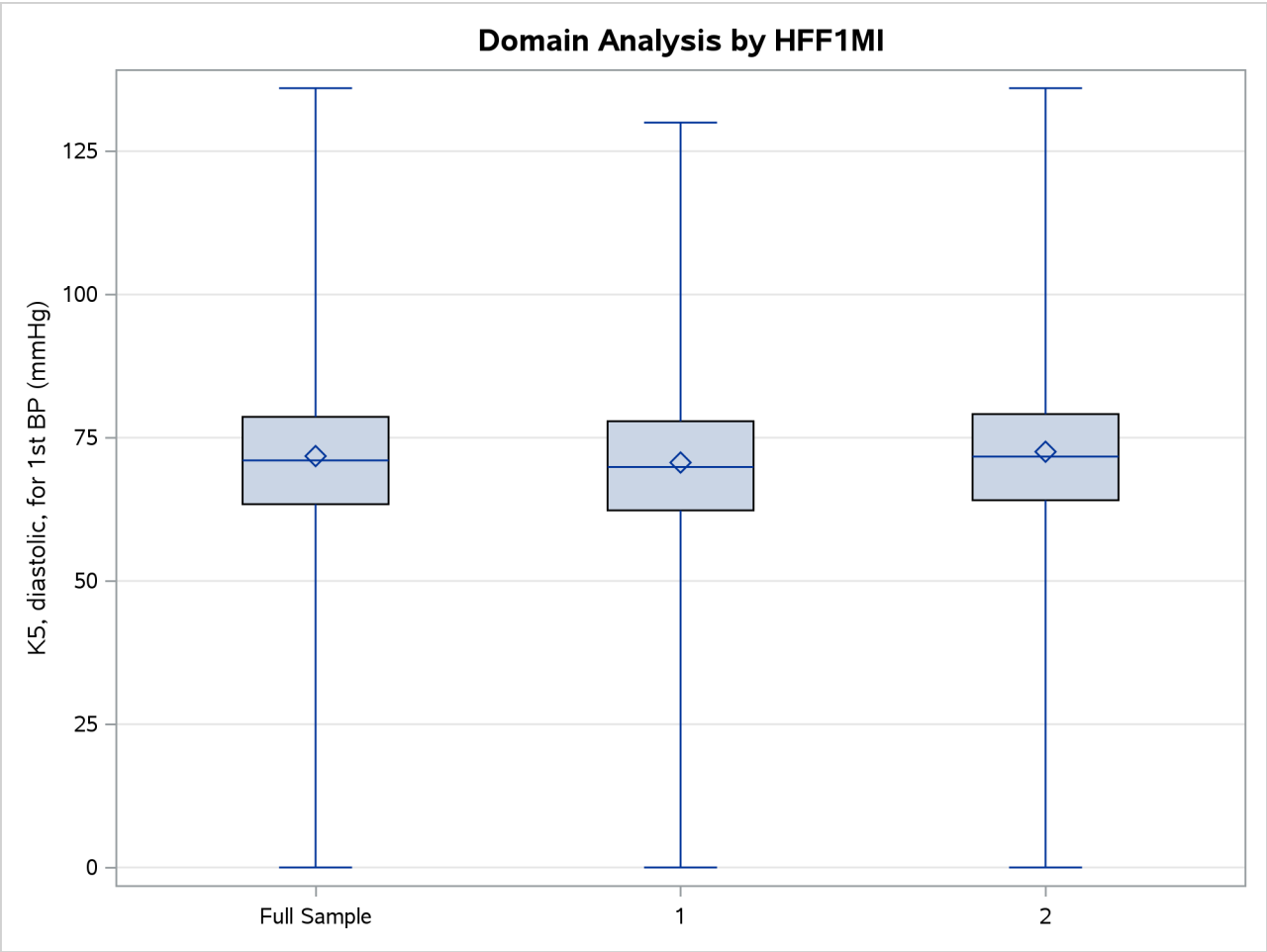

The following PROC SURVEYREG statements estimate the regression coefficients for regressing diastolic blood pressure on smoking status, gender, height, weight, and age. Imputation-adjusted weights and

imputation-adjusted replicate weights are used for point estimation and variance estimation, respectively. The SOLUTION option in the MODEL statement displays the parameter estimates. Estimated values of the regression parameters, along with their standard errors, are shown in Output 119.4.7. The OUT= option in the OUTPUT statement saves the residuals and the fitted values in the SAS data set Resid.

```
proc surveyreg data=HealthFHDI varmethod=brr(Fay=0.3);
   weight ImpWt;
   repweights ImpRepWt_:;
   class hff1mi hssex;
   model pep6g3mi = hff1mi hssex bmphtmi bmpwtmi hsageir / solution;
   output out=Resid residual=Residuals predicted=Fitted;
run;
```

Estimated regression parameters and their standard errors are displayed in Output 119.4.7. All covariates except height (BMPHTMI) have small standard errors compared to their estimated values. Thus, they are all important in describing the regression relationship in the NHANES III population between 17 and 60 years of age. The degrees of freedom for the *t* tests is 52, which is equal to the number of BRR replicates.

Output 119.4.7 Parameter Estimates

The SURVEYREG Procedure

Regression Analysis for Dependent Variable PEP6G3MI

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	45.6902005	4.00157249	11.42	<.0001
HFF1MI 1	-1.3855994	0.33509056	-4.13	0.0001
HFF1MI 2	0.0000000	0.00000000	.	.
HSSEX 1	3.2312257	0.28893940	11.18	<.0001
HSSEX 2	0.0000000	0.00000000	.	.
BMPHTMI	0.0038192	0.02220508	0.17	0.8641
BMPWTMI	0.1792811	0.00940559	19.06	<.0001
HSAGEIR	0.3021222	0.02373061	12.73	<.0001

Note: The degrees of freedom for the t tests is 52.
Matrix X'WX is singular and a generalized inverse was used to solve the normal equations. Estimates are not unique.

The following PROC SURVEYREG statements request a residual plot by using the residuals and fitted values from the Resid data set. Imputation-adjusted weights are displayed by using a heat map, as shown in Output 119.4.8. The SHAPE=HEXAGONAL option requests hexagonal bins, and the NBINS=60 option specifies that 60 bins be used. For more information about how to create customized graphs by using ODS graphs, see Kuhfeld (2016). The weighted residual plot does not reveal any major violations of the model assumptions—namely, linearity and equal variance.

Because fractional imputation increases the number of observation rows, you must use imputation-adjusted weights in all analyses that involve fractionally imputed data sets.

```
ods graphics on;
proc surveyreg data=resid plots(nbins=60)=fit(shape=hexagonal);
   model Residuals=Fitted;
   weight impwt;
run;
```

Output 119.4.8 Plot of Residuals versus Fitted Values

References

Andridge, R. R., and Little, R. J. A. (2010). "A Review of Hot Deck Imputation for Survey Non-response." *International Statistical Review* 78:40–64.

Barnard, J., and Rubin, D. B. (1999). "Small-Sample Degrees of Freedom with Multiple Imputation." *Biometrika* 86:948–955.

Beaumont, J.-F., and Patak, Z. (2012). "On the Generalized Bootstrap for Sample Surveys with Special Attention to Poisson Sampling." *International Statistical Review* 80:127–148.

Bethlehem, J. (2009). *Applied Survey Methods: A Statistical Perspective*. Hoboken, NJ: John Wiley & Sons.

Brick, J. M., and Kalton, G. (1996). "Handling Missing Data in Survey Research." *Statistical Methods in Medical Research* 5:215–238.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Dippo, C. S., Fay, R. E., and Morganstein, D. H. (1984). "Computing Variances from Complex Samples with Replicate Weights." In *Proceedings of the Survey Research Methods Section*, 489–494. Alexandria, VA: American Statistical Association.

Fay, R. E. (1984). "Some Properties of Estimates of Variance Based on Replication Methods." In *Proceedings of the Survey Research Methods Section*, 495–500. Alexandria, VA: American Statistical Association.

Fay, R. E. (1989). "Theory and Application of Replicate Weighting for Variance Calculations." In *Proceedings of the Survey Research Methods Section*, 212–217. Alexandria, VA: American Statistical Association.

Fay, R. E. (1993). "Valid Inferences from Imputed Survey Data." In *Proceedings of the Survey Research Methods Section*, 41–48. Alexandria, VA: American Statistical Association.

Fay, R. E. (1996). "Alternative Paradigms for the Analysis of Imputed Survey Data." *Journal of the American Statistical Association* 91:490–498.

Fellegi, I. P., and Holt, D. (1976). "A Systematic Approach to Automatic Edit and Imputation." *Journal of the American Statistical Association* 71:17–35.

Fuller, W. A. (2009). *Sampling Statistics*. Hoboken, NJ: John Wiley & Sons.

Fuller, W. A., and Kim, J. K. (2005). "Hot Deck Imputation for the Response Model." *Survey Methodology* 31:139–149.

Haziza, D. (2009). "Imputation and Inference in the Presence of Missing Data." In *Sample Surveys: Design, Methods, and Applications*, edited by D. Pfeffermann and C. R. Rao, 215–246. Vol. 29A of Handbook of Statistics. Amsterdam: North-Holland.

Ibrahim, J. G. (1990). "Incomplete Data in Generalized Linear Models." *Journal of the American Statistical Association* 85:765–769.

Im, J., Kim, J. K., and Fuller, W. A. (2015). "Two-Phase Sampling Approach to Fractional Hot Deck Imputation." In *Proceedings of the Survey Research Methods Section*, 1030–1043. Alexandria, VA: American Statistical Association.

Judkins, D. R. (1990). "Fay's Method for Variance Estimation." *Journal of Official Statistics* 6:223–239.

Kalton, G., and Kasprzyk, D. (1986). "The Treatment of Missing Survey Data." *Survey Methodology* 12:1–16.

Kalton, G., and Kish, L. (1984). "Some Efficient Random Imputation Methods." *Communications in Statistics—Theory and Methods* 13:1919–1939.

Kim, J. K. (2002). "A Note on Approximate Bayesian Bootstrap Imputation." *Biometrika* 89:470–477.

Kim, J. K., and Fuller, W. A. (2004). "Fractional Hot Deck Imputation." *Biometrika* 91:559–578.

Kim, J. K., and Fuller, W. A. (2013). "Hot Deck Imputation for Multivariate Missing Data." In *Proceedings of the Fifty-Ninth ISI World Statistics Congress*, 924–929. The Hague: International Statistics Institute.

Kim, J. K., and Shao, J. (2014). *Statistical Methods for Handling Incomplete Data*. Boca Raton, FL: CRC Press.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Kuhfeld, W. F. (2016). "Highly Customized Graphs Using ODS Graphics." In *Proceedings of the SAS Global Forum 2016 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings16/SAS1800-2016.pdf.

Little, R. J. A., and Rubin, D. B. (2002). *Statistical Analysis with Missing Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Lohr, S. L. (2022). *Sampling: Design and Analysis*. 3rd ed. Boca Raton, FL: CRC Press.

Mashreghi, Z., Haziza, D., and Léger, C. (2016). "A Survey of Bootstrap Methods in Finite Population Sampling." *Statistics Surveys* 10:1–52.

Matsumoto, M., and Nishimura, T. (1998). "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-random Number Generator." *ACM Transactions on Modeling and Computer Simulation* 8:3–30.

McCarthy, P. J., and Snowden, C. B. (1985). *The Bootstrap and Finite Population Sampling*. Vital and Health Statistics, Series 2, no. 95. DHHS Publication Number 85–1369. Washington, DC: US Government Printing Office. https://www.cdc.gov/nchs/data/series/sr_02/sr02_095.pdf.

Myers, T. A. (2011). "Goodbye, Listwise Deletion: Presenting Hot Deck Imputation as an Easy and Effective Tool for Handling Missing Data." *Communication Methods and Measures* 5:297–310.

Rao, J. N. K., and Shao, J. (1992). "Jackknife Variance Estimation with Survey Data under Hot Deck Imputation." *Biometrika* 79:811–822.

Rao, J. N. K., and Wu, C. F. J. (1988). "Resampling Inference with Complex Survey Data." *Journal of the American Statistical Association* 83:231–241.

Rao, J. N. K., Wu, C. F. J., and Yue, K. (1992). "Some Recent Work on Resampling Methods for Complex Surveys." *Survey Methodology* 18:209–217.

Rubin, D. B., and Schenker, N. (1986). "Multiple Imputation for Interval Estimation from Simple Random Samples with Ignorable Nonresponse." *Journal of the American Statistical Association* 81:366–374.

Särndal, C.-E., and Lundström, S. (2005). *Estimation in Surveys with Nonresponse*. Chichester, UK: John Wiley & Sons.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Shao, J., and Tu, D. (1995). *The Jackknife and Bootstrap*. New York: Springer-Verlag.

Sitter, R. R. (1992a). "Comparing Three Bootstrap Methods for Survey Data." *Canadian Journal of Statistics* 20:135–154.

Sitter, R. R. (1992b). "A Resampling Procedure for Complex Survey Data." *Journal of the American Statistical Association* 87:755–765.

Tillé, Y. (2006). *Sampling Algorithms*. New York: Springer.

Wolter, K. M. (2007). *Introduction to Variance Estimation*. 2nd ed. New York: Springer.

Chapter 120
The SURVEYLOGISTIC Procedure

Contents

Overview: SURVEYLOGISTIC Procedure **9994**
Getting Started: SURVEYLOGISTIC Procedure **9996**
Syntax: SURVEYLOGISTIC Procedure **10000**
- PROC SURVEYLOGISTIC Statement 10001
- BY Statement 10010
- CLASS Statement 10010
- CLUSTER Statement 10012
- CONTRAST Statement 10013
- DOMAIN Statement 10015
- EFFECT Statement 10017
- ESTIMATE Statement 10018
- FREQ Statement 10019
- LSMEANS Statement 10020
- LSMESTIMATE Statement 10021
- MARGINS Statement 10022
- MODEL Statement 10023
- OUTPUT Statement 10032
- REPWEIGHTS Statement 10035
- SLICE Statement 10037
- STORE Statement 10037
- STRATA Statement 10037
- TEST Statement 10038
- UNITS Statement 10039
- WEIGHT Statement 10039

Details: SURVEYLOGISTIC Procedure **10040**
- Missing Values 10040
- Model Specification 10041
- Model Fitting 10046
- Survey Design Information 10051
- Logistic Regression Models and Parameters 10052
- Variance Estimation 10056
- Domain Analysis 10064
- Hypothesis Testing and Estimation 10064
- Linear Predictor, Predicted Probability, and Confidence Limits 10074
- Output Data Sets 10075
- Displayed Output 10077

ODS Table Names 10082
ODS Graphics 10083
Examples: SURVEYLOGISTIC Procedure **10084**
Example 120.1: Stratified Sampling 10084
Example 120.2: The Medical Expenditure Panel Survey (MEPS) 10090
References **10095**

Overview: SURVEYLOGISTIC Procedure

Categorical responses arise extensively in sample survey. Common examples of responses include the following:

- binary: for example, attended graduate school or not
- ordinal: for example, mild, moderate, and severe pain
- nominal: for example, ABC, NBC, CBS, FOX TV network viewed at a certain hour

Logistic regression analysis is often used to investigate the relationship between such discrete responses and a set of explanatory variables. For a description of logistic regression for sample survey data, see Binder (1981, 1983); Roberts, Rao, and Kumar (1987); Skinner, Holt, and Smith (1989); Morel (1989); Lehtonen and Pahkinen (1995).

For binary response models, the response of a sampling unit can take a specified value or not (for example, attended graduate school or not). Suppose $\mathbf{x}$ is a row vector of explanatory variables and π is the response probability to be modeled. The linear logistic model has the form

$$\text{logit}(\pi) \equiv \log\left(\frac{\pi}{1-\pi}\right) = \alpha + \mathbf{x}\boldsymbol{\beta}$$

where α is the intercept parameter and $\boldsymbol{\beta}$ is the vector of slope parameters.

The logistic model shares a common feature with the more general class of generalized linear models—namely, that a function $g = g(\mu)$ of the expected value, μ, of the response variable is assumed to be linearly related to the explanatory variables. Since μ implicitly depends on the stochastic behavior of the response, and since t he explanatory variables are assumed to be fixed, the function g provides the link between the random (stochastic) component and the systematic (deterministic) component of the response variable. For this reason, Nelder and Wedderburn (1972) refer to $g(\cdot)$ as a link function. One advantage of the logit function over other link functions is that differences on the logistic scale are interpretable regardless of whether the data are sampled prospectively or retrospectively (McCullagh and Nelder 1989, Chapter 4). Other link functions that are widely used in practice are the probit function and the complementary log-log function. The SURVEYLOGISTIC procedure enables you to choose one of these link functions, resulting in fitting a broad class of binary response models of the form

$$g(\pi) = \alpha + \mathbf{x}\boldsymbol{\beta}$$

For ordinal response models, the response Y of an individual or an experimental unit might be restricted to one of a usually small number of ordinal values, denoted for convenience by $1, \ldots, D, D+1$ ($D \geq 1$).

For example, pain severity can be classified into three response categories as 1=mild, 2=moderate, and 3=severe. The SURVEYLOGISTIC procedure fits a common slopes cumulative model, which is a parallel lines regression model based on the cumulative probabilities of the response categories rather than on their individual probabilities. The cumulative model has the form

$$g(\Pr(Y \le d \mid \mathbf{x})) = \alpha_d + \mathbf{x}\boldsymbol{\beta}, \quad 1 \le d \le D$$

where $\alpha_1, \ldots, \alpha_k$ are k intercept parameters and $\boldsymbol{\beta}$ is the vector of slope parameters. This model has been considered by many researchers. Aitchison and Silvey (1957) and Ashford (1959) employ a probit scale and provide a maximum likelihood analysis; Walker and Duncan (1967) and Cox and Snell (1989) discuss the use of the log-odds scale. For the log-odds scale, the cumulative logit model is often referred to as the *proportional odds* model.

For nominal response logistic models, where the $D + 1$ possible responses have no natural ordering, the logit model can also be extended to a *generalized logit* model, which has the form

$$\log\left(\frac{\Pr(Y = i \mid \mathbf{x})}{\Pr(Y = D + 1 \mid \mathbf{x})}\right) = \alpha_i + \mathbf{x}\boldsymbol{\beta}_i, \quad i = 1, \ldots, D$$

where the $\alpha_1, \ldots, \alpha_D$ are D intercept parameters and the $\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_D$ are D vectors of parameters. These models were introduced by McFadden (1974) as the *discrete choice* model, and they are also known as *multinomial* models.

The SURVEYLOGISTIC procedure fits linear logistic regression models for discrete response survey data by the method of maximum likelihood. For statistical inferences, PROC SURVEYLOGISTIC incorporates complex survey sample designs, including designs with stratification, clustering, and unequal weighting.

The maximum likelihood estimation is carried out with either the Fisher scoring algorithm or the Newton-Raphson algorithm. You can specify starting values for the parameter estimates. The logit link function in the ordinal logistic regression models can be replaced by the probit function or the complementary log-log function.

Odds ratio estimates are displayed along with parameter estimates. You can also specify the change in the explanatory variables for which odds ratio estimates are desired.

Variances of the regression parameters and odds ratios are computed by using either the Taylor series (linearization) method or replication (resampling) methods to estimate sampling errors of estimators based on complex sample designs (Binder 1983; Särndal, Swensson, and Wretman 1992; Wolter 2007; Rao, Wu, and Yue 1992).

The SURVEYLOGISTIC procedure enables you to specify categorical variables (also known as CLASS variables) as explanatory variables. It also enables you to specify interaction terms in the same way as in the LOGISTIC procedure.

Like many procedures in SAS/STAT software that allow the specification of CLASS variables, the SURVEYLOGISTIC procedure provides a CONTRAST statement for specifying customized hypothesis tests concerning the model parameters. The CONTRAST statement also provides estimation of individual rows of contrasts, which is particularly useful for obtaining odds ratio estimates for various levels of the CLASS variables.

Getting Started: SURVEYLOGISTIC Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/svlgs.sas.)

The SURVEYLOGISTIC procedure is similar to the LOGISTIC procedure and other regression procedures in the SAS System. See Chapter 79, "The LOGISTIC Procedure," for general information about how to perform logistic regression by using SAS. PROC SURVEYLOGISTIC is designed to handle sample survey data, and thus it incorporates the sample design information into the analysis.

The following example illustrates how to use PROC SURVEYLOGISTIC to perform logistic regression for sample survey data.

In the customer satisfaction survey example in the section "Getting Started: SURVEYSELECT Procedure" on page 10414 in Chapter 124, "The SURVEYSELECT Procedure," an Internet service provider conducts a customer satisfaction survey. The survey population consists of the company's current subscribers from four states: Alabama (AL), Florida (FL), Georgia (GA), and South Carolina (SC). The company plans to select a sample of customers from this population, interview the selected customers and ask their opinions on customer service, and then make inferences about the entire population of subscribers from the sample data. A stratified sample is selected by using the probability proportional to size (PPS) method. The sample design divides the customers into strata depending on their types ('Old' or 'New') and their states (AL, FL, GA, SC). There are eight strata in all. Within each stratum, customers are selected and interviewed by using the PPS with replacement method, where the size variable is Usage. The stratified PPS sample contains 192 customers. The data are stored in the SAS data set SampleStrata. Figure 120.1 displays the first 10 observations of this data set.

Figure 120.1 Stratified PPS Sample (First 10 Observations)

Customer Satisfaction Survey
Stratified PPS Sampling
(First 10 Observations)

OBS	State	Type	CustomerID	Rating	Usage	SamplingWeight
1	AL	New	24394278	Neutral	13.17	26.358
2	AL	New	64798692	Extremely Unsatisfied	15.53	22.352
3	AL	New	75375074	Unsatisfied	99.11	3.501
4	AL	New	262831809	Neutral	5.40	64.228
5	AL	New	294428658	Extremely Satisfied	1.17	297.488
6	AL	New	336222949	Unsatisfied	38.69	8.970
7	AL	New	351929023	Extremely Satisfied	2.72	127.475
8	AL	New	366142640	Satisfied	2.61	132.958
9	AL	New	371478614	Neutral	14.36	24.173
10	AL	New	477172230	Neutral	4.06	85.489

In the SAS data set SampleStrata, the variable CustomerID uniquely identifies each customer. The variable State contains the state of the customer's address. The variable Type equals 'Old' if the customer has subscribed to the service for more than one year; otherwise, the variable Type equals 'New'. The variable Usage contains the customer's average monthly service usage, in hours. The variable Rating contains the customer's responses to the survey. The sample design uses an unequal probability sampling method, with

the sampling weights stored in the variable SamplingWeight.

The following SAS statements fit a cumulative logistic model between the satisfaction levels and the Internet usage by using the stratified PPS sample:

```
title 'Customer Satisfaction Survey';
proc surveylogistic data=SampleStrata;
   strata state type/list;
   model Rating (order=internal) = Usage;
   weight SamplingWeight;
run;
```

The PROC SURVEYLOGISTIC statement invokes the SURVEYLOGISTIC procedure. The STRATA statement specifies the stratification variables State and Type that are used in the sample design. The LIST option requests a summary of the stratification. In the MODEL statement, Rating is the response variable and Usage is the explanatory variable. The ORDER=internal is used for the response variable Rating to ask the procedure to order the response levels by using the internal numerical value (1–5) instead of the formatted character value. The WEIGHT statement specifies the variable SamplingWeight that contains the sampling weights.

The results of this analysis are shown in the following figures.

Figure 120.2 Stratified PPS Sample, Model Information

Customer Satisfaction Survey

The SURVEYLOGISTIC Procedure

Model Information		
Data Set	WORK.SAMPLESTRATA	
Response Variable	Rating	
Number of Response Levels	5	
Stratum Variables	State	
	Type	
Number of Strata	8	
Weight Variable	SamplingWeight	Sampling Weight
Model	Cumulative Logit	
Optimization Technique	Fisher's Scoring	
Variance Adjustment	Degrees of Freedom (DF)	

PROC SURVEYLOGISTIC first lists the following model fitting information and sample design information in Figure 120.2:

- The link function is the logit of the cumulative of the lower response categories.
- The Fisher scoring optimization technique is used to obtain the maximum likelihood estimates for the regression coefficients.
- The response variable is Rating, which has five response levels.
- The stratification variables are State and Type.
- There are eight strata in the sample.

- The weight variable is SamplingWeight.
- The variance adjustment method used for the regression coefficients is the default degrees of freedom adjustment.

Figure 120.3 lists the number of observations in the data set and the number of observations used in the analysis. Since there is no missing value in this example, observations in the entire data set are used in the analysis. The sums of weights are also reported in this table.

Figure 120.3 Stratified PPS Sample, Number of Observations

Number of Observations Read	194
Number of Observations Used	194
Sum of Weights Read	14200.58
Sum of Weights Used	14200.58

The "Response Profile" table in Figure 120.4 lists the five response levels, their ordered values, and their total frequencies and total weights for each category. Due to the ORDER=INTERNAL option for the response variable Rating, the category "Extremely Unsatisfied" has the Ordered Value 1, the category "Unsatisfied" has the Ordered Value 2, and so on.

Figure 120.4 Stratified PPS Sample, Response Profile

Response Profile			
Ordered Value	**Rating**	**Total Frequency**	**Total Weight**
1	Extremely Unsatisfied	61	2049.6278
2	Unsatisfied	42	3084.6615
3	Neutral	49	2295.9993
4	Satisfied	33	3754.7221
5	Extremely Satisfied	9	3015.5727

Probabilities modeled are cumulated over the lower Ordered Values.

Figure 120.5 displays the output of the stratification summary. There are a total of eight strata, and each stratum is defined by the customer types within each state. The table also shows the number of customers within each stratum.

Figure 120.5 Stratified PPS Sample, Stratification Summary

Stratum Information			
Stratum Index	**State**	**Type**	**N Obs**
1	AL	New	24
2		Old	25
3	FL	New	25
4		Old	23
5	GA	New	24
6		Old	25
7	SC	New	24
8		Old	24

Figure 120.6 shows the iteration algorithm converged to obtain the MLE for this example. The "Model Fit Statistics" table contains the Akaike information criterion (AIC), the Schwarz criterion (SC), and the negative of twice the log likelihood ($-2 \log L$) for the intercept-only model and the fitted model. AIC and SC can be used to compare different models, and the ones with smaller values are preferred.

Figure 120.6 Stratified PPS Sample, Model Fitting Information

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	45064.144	44143.526
SC	45094.389	44181.332
-2 Log L	45056.144	44133.526

The table "Testing Global Null Hypothesis: BETA=0" in Figure 120.7 shows the likelihood ratio test and the Wald test for testing the significance of the explanatory variable (Usage). Both tests are significant.

Figure 120.7 Stratified PPS Sample

Testing Global Null Hypothesis: BETA=0				
Test	F Value	Num DF	Den DF	Pr > F
Likelihood Ratio	12.95	1	186	0.0004
Wald	5.93	1	186	0.0159

NOTE:
First-order Rao-Scott design correction 0.9734
applied to the likelihood ratio test.

Figure 120.8 shows the parameter estimates of the logistic regression and their standard errors.

Figure 120.8 Stratified PPS Sample, Parameter Estimates

Analysis of Maximum Likelihood Estimates					
Parameter		Estimate	Standard Error	t Value	Pr > |t|
Intercept	Extremely Unsatisfied	-2.1252	0.3727	-5.70	<.0001
Intercept	Unsatisfied	-0.8649	0.3616	-2.39	0.0178
Intercept	Neutral	-0.1768	0.3645	-0.49	0.6282
Intercept	Satisfied	1.0953	0.5255	2.08	0.0385
Usage		0.0441	0.0181	2.43	0.0159

NOTE: The degrees of freedom for the t tests is 186.

Figure 120.9 displays the odds ratio estimate and its confidence intervals.

Figure 120.9 Stratified PPS Sample, Odds Ratios

Odds Ratio Estimates			
Effect	Point Estimate	95% Confidence Limits	
Usage	1.045	1.008	1.083

NOTE: The degrees of freedom in computing the confidence limits is 186.

Syntax: SURVEYLOGISTIC Procedure

The following statements are available in the SURVEYLOGISTIC procedure:

```
PROC SURVEYLOGISTIC <options> ;
   BY variables ;
   CLASS variable <(v-options)> <variable <(v-options)> ... > </ v-options> ;
   CLUSTER variables ;
   CONTRAST 'label' effect values <, ... effect values> </ options> ;
   DOMAIN variables <variable*variable variable*variable*variable ... > ;
   EFFECT name = effect-type (variables </ options>) ;
   ESTIMATE <'label'> estimate-specification </ options> ;
   FREQ variable ;
   LSMEANS <model-effects> </ options> ;
   LSMESTIMATE model-effect lsmestimate-specification </ options> ;
   MARGINS effects </ options> ;
   MODEL events/trials = <effects </ options>> ;
   MODEL variable <(v-options)> = <effects> </ options> ;
   OUTPUT <OUT=SAS-data-set> <options> </ option> ;
   REPWEIGHTS variables </ options> ;
   SLICE model-effect </ options> ;
   STORE <OUT=>item-store-name </ LABEL='label'> ;
   STRATA variables </ option> ;
   <label:> TEST equation1 <, ..., equationk> </ options> ;
   UNITS independent1 = list1 <... independentk = listk> </ option> ;
   WEIGHT variable ;
```

The PROC SURVEYLOGISTIC and MODEL statements are required.

The CLASS, CLUSTER, CONTRAST, EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, REPWEIGHTS, SLICE, STRATA, and TEST statements can appear multiple times. You should use only one of each following statements: MODEL, WEIGHT, STORE, OUTPUT, and UNITS.

The CLASS and EFFECT statements (if specified) must precede the MODEL statement, and the CONTRAST, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, and SLICE statements (if specified) must follow the MODEL statement.

The rest of this section provides detailed syntax information for each of the preceding statements, except the EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, SLICE, and STORE statements. These statements are also available in many other procedures. Summary descriptions of functionality and syntax for these statements are shown in this chapter, and full documentation about them is available in Chapter 20, "Shared Concepts and Topics."

The syntax descriptions begin with the PROC SURVEYLOGISTIC statement; the remaining statements are covered in alphabetical order.

PROC SURVEYLOGISTIC Statement

PROC SURVEYLOGISTIC < *options* > **;**

The PROC SURVEYLOGISTIC statement invokes the SURVEYLOGISTIC procedure. Optionally, it identifies input data sets, controls the ordering of the response levels, and specifies the variance estimation method. The PROC SURVEYLOGISTIC statement is required.

Table 120.1 summarizes the *options* available in the PROC SURVEYLOGISTIC statement.

Table 120.1 PROC SURVEYLOGISTIC Statement Options

Option	Description
ALPHA=	Sets the confidence level for confidence intervals
DATA=	Names the SAS data set containing the data to be analyzed
INEST=	Names the SAS data set that contains initial estimates
MAXRESPONSELEVELS=	Specifies the maximum number of response levels allowed
MISSING	Treats missing values as a valid category
NAMELEN=	Specifies the length of effect names
NOMCAR	Treats missing values as *not missing completely at random*
NOSORT	Suppresses the internal sorting process
ORDER=	Specifies the sort order
RATE=	Specifies the sampling rate
TOTAL=	Specifies the total number of primary sampling units
VARMETHOD=	Specifies the variance estimation method

ALPHA=*value*

sets the confidence level for confidence intervals. The value of the ALPHA= option must be between 0 and 1, and the default value is 0.05. A confidence level of α produces $100(1-\alpha)\%$ confidence intervals. The default of ALPHA=0.05 produces 95% confidence intervals.

DATA=*SAS-data-set*

names the SAS data set containing the data to be analyzed. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

INEST=*SAS-data-set*

names the SAS data set that contains initial estimates for all the parameters in the model. BY-group processing is allowed in setting up the INEST= data set. See the section "INEST= Data Set" on page 10050 for more information.

MAXRESPONSELEVELS=*number*

specifies the maximum number of response levels that are allowed in your data set. By default, MAXRESPONSELEVELS=100. If you have more response levels than the maximum number allowed, then a message is displayed in the SAS log that provides the value of *number* required to continue the analysis, and the procedure stops.

MISSING

treats missing values as a valid (nonmissing) category for all categorical variables, which include CLASS, STRATA, CLUSTER, and DOMAIN variables.

By default, if you do not specify the MISSING option, an observation is excluded from the analysis if it has a missing value. For more information, see the section "Missing Values" on page 10040.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 200. The default length is 20 characters.

NOMCAR

requests that the procedure treat missing values in the variance computation as *not missing completely at random* (NOMCAR) for Taylor series variance estimation. When you specify the NOMCAR option, PROC SURVEYLOGISTIC computes variance estimates by analyzing the nonmissing values as a domain or subpopulation, where the entire population includes both nonmissing and missing domains. See the section "Missing Values" on page 10040 for more details.

By default, PROC SURVEYLOGISTIC completely excludes an observation from analysis if that observation has a missing value, unless you specify the MISSING option. Note that the NOMCAR option has no effect on a classification variable when you specify the MISSING option, which treats missing values as a valid nonmissing level.

The NOMCAR option applies only to Taylor series variance estimation; it is ignored for replication methods.

NOSORT

suppresses the internal sorting process to shorten the computation time if the data set is presorted by the STRATA and CLUSTER variables. By default, the procedure sorts the data by the STRATA variables if you use the STRATA statement; then the procedure sorts the data by the CLUSTER variables within strata. If your data are already stored by the order of STRATA and CLUSTER variables, then you can specify this option to omit this sorting process to reduce the usage of computing resources, especially when your data set is very large. However, if you specify this NOSORT option while your data are not presorted by STRATA and CLUSTER variables, then any changes in these variables creates a new stratum or cluster.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the response variable. This option, except for ORDER=FREQ, also determines the sort order for the levels of CLUSTER and DOMAIN variables and controls STRATA variable levels in the "Stratum Information" table. By default, ORDER=INTERNAL. However, if an ORDER= option is specified after the response variable, in the MODEL statement, it overrides this option for the response variable. This option does not affect the ordering of the CLASS variable levels; see the ORDER= option in the CLASS statement for more information.

RATE=*value* | *SAS-data-set*

R=*value* | *SAS-data-set*

specifies the sampling rate, which PROC SURVEYLOGISTIC uses to compute a finite population correction for Taylor series or bootstrap variance estimation. This option is ignored for the jackknife or balanced repeated replication (BRR) variance estimation method.

If your sample design has multiple stages, you should specify the *first-stage sampling rate*, which is the ratio of the number of primary sampling units (PSUs) in the sample to the total number of PSUs in the population.

You can specify the sampling rate in either of the following ways:

value specifies a nonnegative number to use for a nonstratified design or for a stratified design that has the same sampling rate in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the sampling rates for a stratified design that has different sampling rates in the strata. You must provide the sampling rates in the data set variable named _RATE_. The sampling rates must be nonnegative numbers.

You can specify sampling rates as numbers between 0 and 1. Or you can specify sampling rates in percentage form as numbers between 1 and 100, which PROC SURVEYLOGISTIC converts to proportions. The procedure treats the value 1 as 100% instead of 1%.

For more information, see the section "Specification of Population Totals and Sampling Rates" on page 10051.

If you do not specify either the RATE= or TOTAL= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. You cannot specify both the RATE= and TOTAL= options.

TOTAL=*value* | *SAS-data-set*

N=*value* | *SAS-data-set*

specifies the total number of primary sampling units (PSUs) in the study population. PROC SURVEYLOGISTIC uses this information to compute a finite population correction for Taylor series or bootstrap variance estimation. This option is ignored for the jackknife or BRR variance estimation method.

You can specify the total number of PSUs in either of the following ways:

value specifies a positive number to use for a nonstratified design or for a stratified design that has the same population total in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the population totals for a stratified design that has different population totals in the strata. You must provide the stratum totals in the data set variable named _TOTAL_. The stratum totals must be positive numbers.

For more information, see the section "Specification of Population Totals and Sampling Rates" on page 10051.

If you do not specify either the TOTAL= or RATE= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. You cannot specify both the TOTAL= and RATE= options.

VARMETHOD=*method* **<(***method-options***)>**
specifies the variance estimation *method*. PROC SURVEYLOGISTIC provides the Taylor series method and the following replication (resampling) methods: balanced repeated replication (BRR), bootstrap, and jackknife.

Table 120.2 summarizes the available *methods* and *method-options*.

Table 120.2 Variance Estimation Methods

method	**Variance Estimation Method**	*method-options*
BOOTSTRAP	Bootstrap	CENTER= MH=*value* | *SAS-data-set* OUTWEIGHTS=*SAS-data-set* REPS=*number* SEED=*number*
BRR	Balanced repeated replication	CENTER= FAY <=*value*> HADAMARD=*SAS-data-set* OUTWEIGHTS=*SAS-data-set* PRINTH REPS=*number*
JACKKNIFE | JK	Jackknife	CENTER= OUTJKCOEFS=*SAS-data-set* OUTWEIGHTS=*SAS-data-set*
TAYLOR	Taylor series linearization	None

For VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, and VARMETHOD=JACKKNIFE, you can specify *method-options* in parentheses after the variance estimation *method*. For example:

```
varmethod=BRR(reps=60 outweights=myReplicateWeights)
```

By default, VARMETHOD=JACKKNIFE if you also specify a REPWEIGHTS statement; otherwise, VARMETHOD=TAYLOR by default.

You can specify the following *methods*:

BOOTSTRAP < (*method-options*) >

requests variance estimation by the bootstrap method. For more information, see the section "Bootstrap Method" on page 10058.

The bootstrap method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you use a REPWEIGHTS statement to provide replicate weights.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Bootstrap Method" on page 10058.

MH=*value* | (*values*) | *SAS-data-set*

specifies the number of PSUs to select for the bootstrap replicate samples. You can provide bootstrap stratum sample sizes m_h by specifying a list of *values* or a *SAS-data-set*. Alternatively, you can provide a single bootstrap sample size *value* to use for all strata or for a nonstratified design. You can specify the number of replicate samples in the REPS= option. For more information, see the section "Bootstrap Method" on page 10058.

Each bootstrap sample size m_h must be a positive integer and must be less than n_h, which is the total number of PSUs in stratum h. By default, $m_h = n_h - 1$ for a stratified design. For a nonstratified design, the bootstrap sample size *value* must be less than n (the total number of PSUs in the sample). By default, $m = n - 1$ for a nonstratified design.

You can provide bootstrap sample sizes by specifying one of the following forms:

MH=*value*

specifies a single bootstrap sample size *value* to use for all strata or for a nonstratified design.

MH=(*values***)**

specifies a list of stratum bootstrap sample size *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. The number of values must not be less than the number of strata in the DATA= input data set.

Each stratum sample size value must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

MH=*SAS-data-set*

names a *SAS-data-set* that contains the stratum bootstrap sample sizes. You must provide the sample sizes in a data set variable named _NSIZE_ or SampleSize.

The *SAS-data-set* must contain all stratification variables that you specify in the STRATA statement. It must also contain all stratum levels that appear in the DATA= input data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets.

Each value of the _NSIZE_ or SampleSize variable must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* in which to store the bootstrap replicate weights that PROC SURVEYLOGISTIC creates. For information about replicate weights, see the section "Bootstrap Method" on page 10058. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10076.

This *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

REPS=*number*

specifies the *number* of replicates for bootstrap variance estimation. The value of *number* must be an integer greater than 1. Increasing the number of replicates improves the estimation precision but also increases the computation time. By default, REPS=250.

SEED=*number*

specifies the initial seed for random number generation for bootstrap replicate sampling.

If you do not specify this option or if you specify a *number* that is negative or 0, PROC SURVEYLOGISTIC uses the time of day from the system clock to obtain an initial seed.

To reproduce the same bootstrap replicate weights and the same analysis in a subsequent execution of PROC SURVEYLOGISTIC, you can specify the same initial seed that was used in the original analysis.

PROC SURVEYLOGISTIC displays the value of the initial seed in the "Variance Estimation" table.

BRR <(*method-options***)>**

requests variance estimation by balanced repeated replication (BRR). This method requires a stratified sample design where each stratum contains two primary sampling units (PSUs). When you specify this method, you must also specify a STRATA statement unless you provide replicate weights by using the REPWEIGHTS statement. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10060.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10060.

FAY < =*value* >

requests Fay's method, which is a modification of the BRR method. For more information, see the section "Fay's BRR Method" on page 10061.

You can specify the *value* of the Fay coefficient, which is used in converting the original sampling weights to replicate weights. The Fay coefficient must be a nonnegative number less than 1. By default, the Fay coefficient is 0.5.

HADAMARD=*SAS-data-set*

H=*SAS-data-set*

names a *SAS-data-set* that contains the Hadamard matrix for BRR replicate construction. If you do not specify this *method-option*, PROC SURVEYLOGISTIC generates an appropriate Hadamard matrix for replicate construction. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 10060 and "Hadamard Matrix" on page 10064.

If a Hadamard matrix of a particular dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a *SAS-data-set* in this *method-option*.

In this *SAS-data-set*, each variable corresponds to a column and each observation corresponds to a row of the Hadamard matrix. You can use any variable names in this data set. All values in the data set must equal either 1 or –1. You must ensure that the matrix you provide is indeed a Hadamard matrix—that is, $\mathbf{A}'\mathbf{A} = R\,\mathbf{I}$, where $\mathbf{A}$ is the Hadamard matrix of dimension R and $\mathbf{I}$ is an identity matrix. PROC SURVEYLOGISTIC does not check the validity of the Hadamard matrix that you provide.

The *SAS-data-set* must contain at least H variables, where H denotes the number of first-stage strata in your design. If the data set contains more than H variables, PROC SURVEYLOGISTIC uses only the first H variables. Similarly, this data set must contain at least H observations.

If you do not specify the REPS= *method-option*, the number of replicates is assumed to be the number of observations in the *SAS-data-set*. If you specify the number of replicates—for example, REPS=*nreps*—the first *nreps* observations in the *SAS-data-set* are used to construct the replicates.

You can specify the PRINTH *method-option* to display the Hadamard matrix that PROC SURVEYLOGISTIC uses to construct replicates for BRR.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* in which to store the replicate weights that PROC SURVEYLOGISTIC creates for BRR variance estimation. For information about replicate weights, see the section "Balanced Repeated Replication (BRR) Method" on page 10060. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10076.

This *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

PRINTH

displays the Hadamard matrix that PROC SURVEYLOGISTIC uses to construct replicates for BRR variance estimation. When you provide the Hadamard matrix in the HADAMARD= *method-option*, PROC SURVEYLOGISTIC displays only the rows and columns that are actually used to construct replicates. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 10060 and "Hadamard Matrix" on page 10064.

The PRINTH *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement because the procedure does not use a Hadamard matrix in this case.

REPS=*number*

specifies the *number* of replicates for BRR variance estimation. The value of *number* must be an integer greater than 1.

If you do not use the HADAMARD= *method-option* to provide a Hadamard matrix, the number of replicates should be greater than the number of strata and should be a multiple of 4. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10060. If PROC SURVEYLOGISTIC cannot construct a Hadamard matrix for the REPS= value that you specify, the value is increased until a Hadamard matrix of that dimension can be constructed. Therefore, the actual number of replicates that PROC SURVEYLOGISTIC uses might be larger than *number*.

If you use the HADAMARD= *method-option* to provide a Hadamard matrix, the value of *number* must not be greater than the number of rows in the Hadamard matrix. If you provide a Hadamard matrix and do not specify the REPS= *method-option*, the number of replicates is the number of rows in the Hadamard matrix.

If you do not specify the REPS= or the HADAMARD= *method-option* and do not use a REPWEIGHTS statement, the number of replicates is the smallest multiple of 4 that is greater than the number of strata.

If you use a REPWEIGHTS statement to provide replicate weights, PROC SURVEYLOGISTIC does not use the REPS= *method-option*; the number of replicates is the number of REPWEIGHTS variables.

JACKKNIFE <(*method-options***)>**

JK <(*method-options***)>**

requests variance estimation by the delete-1 jackknife method. For more information, see the section "Jackknife Method" on page 10062. If you use a REPWEIGHTS statement to provide replicate weights, VARMETHOD=JACKKNIFE is the default variance estimation method.

The delete-1 jackknife method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you use a REPWEIGHTS statement to provide replicate weights.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES
defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Jackknife Method" on page 10062.

OUTJKCOEFS=*SAS-data-set*
names a *SAS-data-set* in which to store the jackknife coefficients. For information about jackknife coefficients, see the section "Jackknife Method" on page 10062. For information about the contents of the OUTJKCOEFS= data set, see the section "Jackknife Coefficients Output Data Set" on page 10076.

OUTWEIGHTS=*SAS-data-set*
names a *SAS-data-set* in which to store the replicate weights that PROC SURVEYLOGISTIC creates for jackknife variance estimation. For information about replicate weights, see the section "Jackknife Method" on page 10062. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10076.

This *method-option* is not available when you use a REPWEIGHTS statement to provide replicate weights.

TAYLOR
requests Taylor series variance estimation. This is the default method if you do not specify the VARMETHOD= option or a REPWEIGHTS statement. For more information, see the section "Taylor Series (Linearization)" on page 10056.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SURVEYLOGISTIC to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SURVEYLOGISTIC procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Note that using a BY statement provides completely separate analyses of the BY groups. It does not provide a statistically valid domain (subpopulation) analysis, where the total number of units in the subpopulation is not known with certainty. You should use the DOMAIN statement to obtain domain analysis. For more information about subpopulation analysis for sample survey data, see Cochran (1977).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(***v-options***)** > < *variable* < **(***v-options***)** > . . . > < / *v-options* > **;**

The CLASS statement names the classification variables to be used in the analysis. The CLASS statement must precede the MODEL statement. You can specify various *v-options* for each variable by enclosing them in parentheses after the variable name. You can also specify global *v-options* for the CLASS statement by placing them after a slash (/). Global *v-options* are applied to all the variables specified in the CLASS statement. However, individual CLASS variable *v-options* override the global *v-options*.

CPREFIX= *n*

specifies that, at most, the first *n* characters of a CLASS variable name be used in creating names for the corresponding dummy variables. The default is $32 - \min(32, \max(2, f))$, where f is the formatted length of the CLASS variable.

DESCENDING
DESC
reverses the sort order of the classification variable.

LPREFIX= *n*
specifies that, at most, the first *n* characters of a CLASS variable label be used in creating labels for the corresponding dummy variables.

ORDER=DATA | FORMATTED | FREQ | INTERNAL
specifies the order in which to sort the levels of the classification variables. This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PARAM=*keyword*
specifies the parameterization method for the classification variable or variables. Design matrix columns are created from CLASS variables according to the following coding schemes; the default is PARAM=EFFECT.

EFFECT specifies effect coding

GLM specifies less-than-full-rank, reference cell coding; this option can be used only as a global option

ORDINAL specifies the cumulative parameterization for an ordinal CLASS variable

POLYNOMIAL | POLY specifies polynomial coding

REFERENCE | REF specifies reference cell coding

ORTHEFFECT orthogonalizes PARAM=EFFECT

ORTHORDINAL | ORTHOTHERM orthogonalizes PARAM=ORDINAL

ORTHPOLY orthogonalizes PARAM=POLYNOMIAL

ORTHREF orthogonalizes PARAM=REFERENCE

If PARAM=ORTHPOLY or PARAM=POLY, and the CLASS levels are numeric, then the ORDER= option in the CLASS statement is ignored, and the internal, unformatted values are used.

EFFECT, POLYNOMIAL, REFERENCE, ORDINAL, and their orthogonal parameterizations are full rank. The REF= option in the CLASS statement determines the reference level for EFFECT, REFERENCE, and their orthogonal parameterizations.

Parameter names for a CLASS predictor variable are constructed by concatenating the CLASS variable name with the CLASS levels. However, for the POLYNOMIAL and orthogonal parameterizations, parameter names are formed by concatenating the CLASS variable name and *keywords* that reflect the parameterization.

REFERENCE=*'level'* | *keyword*

REF=*'level'* | *keyword*

specifies the reference level for PARAM=EFFECT or PARAM=REFERENCE. For an individual (but not a global) variable REF= *option*, you can specify the *level* of the variable to use as the reference level. For a global or individual variable REF= *option*, you can use one of the following *keywords*. The default is REF=LAST.

FIRST designates the first-ordered level as reference

LAST designates the last-ordered level as reference

CLUSTER Statement

CLUSTER *variables* **;**

The CLUSTER statement names variables that identify the clusters in a clustered sample design. The combinations of categories of CLUSTER variables define the clusters in the sample. If there is a STRATA statement, clusters are nested within strata.

If you provide replicate weights for BRR or jackknife variance estimation with the REPWEIGHTS statement, you do not need to specify a CLUSTER statement.

If your sample design has clustering at multiple stages, you should identify only the first-stage clusters (primary sampling units (PSUs)), in the CLUSTER statement. See the section "Primary Sampling Units (PSUs)" on page 10052 for more information.

The CLUSTER *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the CLUSTER variables determine the CLUSTER variable levels. Thus, you can use formats to group values into levels. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference* for more information.

When determining levels of a CLUSTER variable, an observation with missing values for this CLUSTER variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10040.

You can use multiple CLUSTER statements to specify cluster variables. The procedure uses variables from all CLUSTER statements to create clusters.

CONTRAST Statement

CONTRAST '*label*' *row-description* < **,** . . . **,** *row-description* < / *options* > > **;**

where a *row-description* is defined as follows:

effect values < , . . . , *effect values* >

The CONTRAST statement provides a mechanism for obtaining customized hypothesis tests. It is similar to the CONTRAST statement in PROC LOGISTIC and PROC GLM, depending on the coding schemes used with any classification variables involved.

The CONTRAST statement enables you to specify a matrix, $\mathbf{L}$, for testing the hypothesis $\mathbf{L}\boldsymbol{\theta} = 0$, where $\boldsymbol{\theta}$ is the parameter vector. You must be familiar with the details of the model parameterization that PROC SURVEYLOGISTIC uses (for more information, see the PARAM= option in the section "CLASS Statement" on page 10010). Optionally, the CONTRAST statement enables you to estimate each row, $\mathbf{l}_i\boldsymbol{\theta}$, of $\mathbf{L}\boldsymbol{\theta}$ and test the hypothesis $\mathbf{l}_i\boldsymbol{\theta} = 0$. For more information, see the section "Testing Linear Hypotheses about the Regression Coefficients" on page 10069.

There is no limit to the number of CONTRAST statements that you can specify, but they must appear after the MODEL statement.

The following parameters can be specified in the CONTRAST statement:

label identifies the contrast on the output. A label is required for every contrast specified, and it must be enclosed in quotes.

effect identifies an effect that appears in the MODEL statement. The name INTERCEPT can be used as an effect when one or more intercepts are included in the model. You do not need to include all effects that are included in the MODEL statement.

values are constants that are elements of the $\mathbf{L}$ matrix associated with the effect. To correctly specify your contrast, it is crucial to know the ordering of parameters within each effect and the variable levels associated with any parameter. The "Class Level Information" table shows the ordering of levels within variables. The E option, described later in this section, enables you to verify the proper correspondence of *values* to parameters.

The rows of $\mathbf{L}$ are specified in order and are separated by commas. Multiple degree-of-freedom hypotheses can be tested by specifying multiple *row-descriptions*. For any of the full-rank parameterizations, if an effect is not specified in the CONTRAST statement, all of its coefficients in the $\mathbf{L}$ matrix are set to 0. If too many values are specified for an effect, the extra ones are ignored. If too few values are specified, the remaining ones are set to 0.

When you use effect coding (by default or by specifying PARAM=EFFECT in the CLASS statement), all parameters are directly estimable (involve no other parameters).

For example, suppose an effect that is coded CLASS variable A has four levels. Then there are three parameters $(\alpha_1, \alpha_2, \alpha_3)$ that represent the first three levels, and the fourth parameter is represented by

$$-\alpha_1 - \alpha_2 - \alpha_3$$

To test the first versus the fourth level of A, you would test

$$\alpha_1 = -\alpha_1 - \alpha_2 - \alpha_3$$

or, equivalently,

$$2\alpha_1 + \alpha_2 + \alpha_3 = 0$$

which, in the form $\mathbf{L}\boldsymbol{\theta} = 0$, is

$$\begin{bmatrix} 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = 0$$

Therefore, you would use the following CONTRAST statement:

```
contrast '1 vs. 4' A 2 1 1;
```

To contrast the third level with the average of the first two levels, you would test

$$\frac{\alpha_1 + \alpha_2}{2} = \alpha_3$$

or, equivalently,

$$\alpha_1 + \alpha_2 - 2\alpha_3 = 0$$

Therefore, you would use the following CONTRAST statement:

```
contrast '1&2 vs. 3' A 1 1 -2;
```

Other CONTRAST statements are constructed similarly. For example:

```
contrast '1 vs. 2    '  A  1 -1  0;
contrast '1&2 vs. 4  '  A  3  3  2;
contrast '1&2 vs. 3&4'  A  2  2  0;
contrast 'Main Effect'  A  1  0  0,
                        A  0  1  0,
                        A  0  0  1;
```

When you use the less-than-full-rank parameterization (by specifying PARAM=GLM in the CLASS statement), each row is checked for estimability. If PROC SURVEYLOGISTIC finds a contrast to be nonestimable, it displays missing values in corresponding rows in the results. PROC SURVEYLOGISTIC handles missing level combinations of classification variables in the same manner as PROC LOGISTIC. Parameters corresponding to missing level combinations are not included in the model. This convention can affect the way in which you specify the **L** matrix in your CONTRAST statement. If the elements of **L** are not specified for an effect that contains a specified effect, then the elements of the specified effect are distributed over the levels of the higher-order effect just as the LOGISTIC procedure does for its CONTRAST and ESTIMATE statements. For example, suppose that the model contains effects A and B and their interaction A*B. If you specify a CONTRAST statement involving A alone, the **L** matrix contains nonzero terms for both A and A*B, since A*B contains A.

The degrees of freedom is the number of linearly independent constraints implied by the CONTRAST statement—that is, the rank of **L**.

You can specify the following *options* after a slash (/):

ALPHA=*value*

sets the confidence level for confidence intervals. The value of the ALPHA= option must be between 0 and 1, and the default value is 0.05. A confidence level of α produces $100(1-\alpha)\%$ confidence intervals. The default of ALPHA=0.05 produces 95% confidence intervals.

E

requests that the **L** matrix be displayed.

ESTIMATE=*keyword*

requests that each individual contrast (that is, each row, $\mathbf{l}_i\boldsymbol{\beta}$, of $\mathbf{L}\boldsymbol{\beta}$) or exponentiated contrast ($e^{\mathbf{l}_i\boldsymbol{\beta}}$) be estimated and tested. PROC SURVEYLOGISTIC displays the point estimate, its standard error, a *t* or Wald confidence interval, and a *t* or Wald chi-square test for each contrast. The significance level of the confidence interval is controlled by the ALPHA= option. You can estimate the contrast or the exponentiated contrast ($e^{\mathbf{l}_i\boldsymbol{\beta}}$), or both, by specifying one of the following *keywords*:

PARM	specifies that the contrast itself be estimated
EXP	specifies that the exponentiated contrast be estimated
BOTH	specifies that both the contrast and the exponentiated contrast be estimated

SINGULAR=*value*

tunes the estimability checking. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the largest absolute value of the elements of $\mathbf{v}$. For a row vector $\mathbf{l}$ of the matrix $\mathbf{L}$, define

$$c = \begin{cases} \text{ABS}(\mathbf{l}) & \text{if ABS}(\mathbf{l}) > 0 \\ 1 & \text{otherwise} \end{cases}$$

If ABS($\mathbf{l} - \mathbf{lH}$) is greater than c**value*, then $\mathbf{l}\boldsymbol{\beta}$ is declared nonestimable. The $\mathbf{H}$ matrix is the Hermite form matrix $\mathbf{I}_0^{-}\mathbf{I}_0$, where $\mathbf{I}_0^{-}$ represents a generalized inverse of the information matrix $\mathbf{I}_0$ of the null model. The *value* must be between 0 and 1; the default is 10^{-4}.

DOMAIN Statement

DOMAIN *variables* < *variable*∗*variable variable*∗*variable*∗*variable . . .* > **;**

The DOMAIN statement requests analysis for domains (subpopulations) in addition to analysis for the entire study population. The DOMAIN statement names the *variables* that identify domains, which are called domain variables.

A domain variable can be either character or numeric. The procedure treats domain variables as categorical variables. If a *variable* appears by itself in a DOMAIN statement, each level of this variable determines a domain in the study population. If two or more *variables* are joined by asterisks (∗), then every possible combination of levels of these variables determines a domain. The procedure performs a descriptive analysis within each domain that is defined by the domain variables.

The formatted values of the domain variables determine the categorical variable levels. Thus, you can use formats to group values into levels. For more information, see the FORMAT procedure in *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

When the procedure is determining levels of a DOMAIN variable, an observation with missing values for this DOMAIN variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10040.

It is common practice to compute statistics for domains. Because formation of these domains might be unrelated to the sample design, the sample sizes for the domains are random variables. Use a DOMAIN statement to incorporate this variability into the variance estimation.

A DOMAIN statement is different from a BY statement. In a BY statement, you treat the sample sizes as fixed in each subpopulation, and you perform analysis within each BY group independently. For more information, see the section "Domain Analysis" on page 10064. Similarly, you should use a DOMAIN statement to perform a domain analysis over the entire data set. Creating a new data set from a single domain and analyzing that with PROC SURVEYLOGISTIC yields inappropriate estimates of variance.

By default, the SURVEYLOGISTIC procedure displays analyses for all levels of domains that are formed by the variables in a DOMAIN statement. Optionally, you can specify particular levels of each DOMAIN variable to be displayed by listing quoted *formatted-level-values* in parentheses after each variable name. You must enclose each *formatted-level-value* in single or double quotation marks. You can specify one or more levels of each variable; when you specify more than one level, separate the levels by a space or a comma. These examples illustrate the syntax:

```
domain Race*Gender(''Female'');
domain Race('White','Asian') Gender;
```

For example, **Race*Gender("Female")** requests that the procedure display analysis only for females within each race category, and **Race('White','Asian')** requests that the procedure display domain analysis only for people whose race is either white or Asian.

Specifying the same domain multiple times but with different levels for each corresponding domain variables is equivalent to specifying the union of different levels for the same variables. However, if you do not specify levels for a variable in a domain that is specified multiple times, only the specified levels are rendered. For example, the following two specifications together

```
domain Race('White')*Gender('Female');
domain Race('Asian')*Gender;
```

have the same effect as a single specification:

```
domain Race('White' 'Asian')*Gender('Female');
```

Also, the following specification

```
domain Race('White')*Gender Race('Asian')*Gender;
```

is equivalent to

```
domain Race('White' 'Asian')*Gender;
```

This syntax controls only the display of domain analysis results; it does not subset the data set, change the degrees of freedom, or otherwise affect the variance estimation.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 120.3 summarizes the *options* available in the EFFECT statement.

Table 120.3 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.

Table 120.3 *continued*

Option	Description
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 120.4 summarizes the *options* available in the ESTIMATE statement.

Table 120.4 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference

Table 120.4 *continued*

Option	Description
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected *p*-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the **L** matrix
JOINT	Produces a joint *F* or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement" on page 475 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a variable that contains the frequency of occurrence of each observation. PROC SURVEYLOGISTIC treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If it is not an integer, the frequency value is truncated to an integer. If the frequency value is less than 1 or missing, the observation is not used in the model fitting. When the FREQ statement is not specified, each observation is assigned a frequency of 1.

If you use the events/trials syntax in the MODEL statement, the FREQ statement is not allowed because the event and trial variables represent the frequencies in the data set.

If you use the FREQ statement and you specify the VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or VARMETHOD=BOOTSTRAP option in the PROC SURVEYLOGISTIC statement to estimate the variance,

then you must use the CLUSTER statement to identify the primary sampling units unless you also provide replicate weights with a REPWEIGHTS statement.

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted margins*—that is, they estimate the marginal means over a hypothetical balanced population on the linked scale. For example, in a binomial model with logit link, the least squares means are predicted population margins of the logits.

Table 120.5 the summarizes available *options* in the LSMEANS statement.

Table 120.5 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers

Table 120.5 *continued*

Option	Description
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For details about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 120.6 summarizes the *options* available in the LSMESTIMATE statement.

Table 120.6 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference

Table 120.6 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MARGINS Statement

MARGINS *model-effects* < / *options* > **;**

The MARGINS statement computes and compares predictive margins of fixed effects. The predictive margin for a specific level (group) of a fixed effect represents the average predicted response if all the observations in the data set were in that group (Lane and Nelder 1982; Chang, Gelman, and Pagano 1982). You can compute predictive margins for any effect in the MODEL statement that involves only classification variables.

Table 120.7 summarizes the *options* available in the MARGINS statement.

Table 120.7 MARGINS Statement Options

Option	Description
Construction and Computation of Predictive Margins	
AT	Modifies the covariate value in computing predictive margins
DIFF	Computes differences of predictive margins
SLICEBY=	Partitions tests of interaction effects
SLICEDIFF	Computes differences of sliced predictive margins and determines the types of differences
Degrees of Freedom and *p*-Values	
ADJUST=	Specifies the method of multiple comparison adjustment of predictive margin differences
ALPHA=α	Specifies the confidence level $(1-\alpha)$
DF=	Assigns a specific value to degrees of freedom for t tests and confidence limits
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion
Statistical Output	
CL	Constructs confidence limits for predictive margins and/or predictive margin differences

For more information about the syntax of the MARGINS statement, see the section "MARGINS Statement" on page 525 in Chapter 20, "Shared Concepts and Topics."

NOTE: If your model has classification variables, then the MARGINS statement is allowed only if you also specify the PARAM=GLM option.

MODEL Statement

MODEL *events/trials* **=** < *effects* < / *options* > > **;**

MODEL *variable* < **(***v-options***)** > **=** < *effects* > < / *options* > **;**

The MODEL statement names the response variable and the explanatory effects, including covariates, main effects, interactions, and nested effects; see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure," for more information. If you omit the explanatory variables, the procedure fits an intercept-only model. Model options can be specified after a slash (/).

Two forms of the MODEL statement can be specified. The first form, referred to as *single-trial* syntax, is applicable to binary, ordinal, and nominal response data. The second form, referred to as *events/trials* syntax, is restricted to the case of binary response data. The single-trial syntax is used when each observation in the DATA= data set contains information about only a single trial, such as a single subject in an experiment. When each observation contains information about multiple binary-response trials, such as the counts of the number of subjects observed and the number responding, then events/trials syntax can be used.

In the events/trials syntax, you specify two variables that contain count data for a binomial experiment. These two variables are separated by a slash. The value of the first variable, *events*, is the number of positive

responses (or events), and it must be nonnegative. The value of the second variable, *trials*, is the number of trials, and it must not be less than the value of *events*.

In the single-trial syntax, you specify one variable (on the left side of the equal sign) as the response variable. This variable can be character or numeric. Options specific to the response variable can be specified immediately after the response variable with parentheses around them.

For both forms of the MODEL statement, explanatory *effects* follow the equal sign. Variables can be either continuous or classification variables. Classification variables can be character or numeric, and they must be declared in the CLASS statement. When an effect is a classification variable, the procedure enters a set of coded columns into the design matrix instead of directly entering a single column containing the values of the variable.

Response Variable Options

You specify the following *options* by enclosing them in parentheses after the response variable:

DESCENDING
DESC

reverses the order of response categories. If both the DESCENDING and the ORDER= options are specified, PROC SURVEYLOGISTIC orders the response categories according to the ORDER= option and then reverses that order. See the section "Response Level Ordering" on page 10041 for more detail.

EVENT='*category***' | *keyword***

specifies the event category for the binary response model. PROC SURVEYLOGISTIC models the probability of the event category. The EVENT= option has no effect when there are more than two response categories. You can specify the value (formatted if a format is applied) of the event category in quotes or you can specify one of the following *keywords*. The default is EVENT=FIRST.

FIRST designates the first-ordered category as the event

LAST designates the last-ordered category as the event

One of the most common sets of response levels is {0,1}, with 1 representing the event for which the probability is to be modeled. Consider the example where Y takes the values 1 and 0 for event and nonevent, respectively, and Exposure is the explanatory variable. To specify the value 1 as the event category, use the following MODEL statement:

```
model Y(event='1') = Exposure;
```

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the response variable.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=INTERNAL. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

REFERENCE='*category*' | *keyword*

REF='*category*' | *keyword*

specifies the reference category for the generalized logit model and the binary response model. For the generalized logit model, each nonreference category is contrasted with the reference category. For the binary response model, specifying one response category as the reference is the same as specifying the other response category as the event category. You can specify the value (formatted if a format is applied) of the reference category in quotes or you can specify one of the following *keywords*. The default is REF=LAST.

FIRST designates the first-ordered category as the reference

LAST designates the last-ordered category as the reference

Model Options

Model *options* can be specified after a slash (/). Table 120.8 summarizes the *options* available in the MODEL statement.

Table 120.8 MODEL Statement Options

Option	Description
Model Specification Options	
LINK=	Specifies link function
NOINT	Suppresses intercept(s)
OFFSET=	Specifies offset variable
Convergence Criterion Options	
ABSFCONV=	Specifies absolute function convergence criterion
FCONV=	Specifies relative function convergence criterion
GCONV=	Specifies relative gradient convergence criterion
XCONV=	Specifies relative parameter convergence criterion
MAXITER=	Specifies maximum number of iterations

Table 120.8 (continued)

Option	Description
NOCHECK	Suppresses checking for infinite parameters
RIDGING=	Specifies technique used to improve the log-likelihood function when its value is worse than that of the previous step
SINGULAR=	Specifies tolerance for testing singularity
TECHNIQUE=	Specifies iterative algorithm for maximization
Options for Adjustment to Variance Estimation	
VADJUST=	Chooses variance estimation adjustment method
Options for Confidence Intervals	
DF=	Specifies the degrees of freedom
ALPHA=	Specifies α for the $100(1-\alpha)\%$ confidence intervals
CHISQ	Specifies the type of likelihood ratio chi-square test
CLPARM	Computes confidence intervals for parameters
CLODDS	Computes confidence intervals for odds ratios
Options for Display of Details	
CORRB	Displays correlation matrix
COVB	Displays covariance matrix
EXPB	Displays exponentiated values of estimates
GRADIENT	Displays gradients evaluated at null hypothesis
ITPRINT	Displays iteration history
NODUMMYPRINT	Suppresses "Class Level Information" table
PARMLABEL	Displays parameter labels
RSQUARE	Displays generalized R^2
STB	Displays standardized estimates

The following list describes these *options*:

ABSFCONV=*value*

specifies the absolute function convergence criterion. Convergence requires a small change in the log-likelihood function in subsequent iterations:

$$|l^{(i)} - l^{(i-1)}| < value$$

where $l^{(i)}$ is the value of the log-likelihood function at iteration i. See the section "Convergence Criteria" on page 10048.

ALPHA=*value*

sets the level of significance α for $100(1-\alpha)\%$ confidence intervals for regression parameters or odds ratios. The value α must be between 0 and 1. By default, α is equal to the value of the ALPHA= option in the PROC SURVEYLOGISTIC statement, or $\alpha = 0.05$ if the ALPHA= option is not specified. This option has no effect unless confidence intervals for the parameters or odds ratios are requested.

CHISQ (FIRSTORDER | NOADJUST | SECONDORDER)

specifies the type of likelihood ratio chi-square test. If you specify CHISQ(FIRSTORDER) or CHISQ(SECONDORDER), PROC SURVEYLOGISTIC provides a first-order or second-order (Satterthwaite) Rao-Scott likelihood ratio chi-square test, which is a design-adjusted test. If you specify CHISQ(NOADJUST), the procedure computes a chi-square test without the Rao-Scott design correction.

If you do not specify the CHISQ option, the default test that PROC SURVEYLOGISTIC uses depends on the design and model as follows:

- If you do not use a STRATA, CLUSTER, or REPWEIGHTS statement, then the default is CHISQ(NOADJUST).
- If you use a STRATA, CLUSTER, or REPWEIGHTS statement, and you need to estimate only one parameter excluding the intercepts in the model, then the default is CHISQ(FIRSTORDER).
- If you use a STRATA, CLUSTER, or REPWEIGHTS statement, and you need to estimate more than one parameter excluding the intercepts in the model, then the default is CHISQ(SECONDORDER).

For more information, see the section "Rao-Scott Likelihood Ratio Chi-Square Test" on page 10066.

Note that unless you specify the DF=INFINITY option, PROC SURVEYLOGISTIC displays an F test instead of a chi-square test.

CLODDS

requests confidence intervals for the odds ratios. Computation of these confidence intervals is based on individual t tests or Wald tests. The degrees of freedom for a t test is described in the section "Degrees of Freedom" on page 10064. The confidence coefficient can be specified with the ALPHA= option. See the section "Wald Confidence Intervals for Parameters" on page 10068 for more information.

CLPARM

requests confidence intervals for the parameters. Computation of these confidence intervals is based on the t tests or Wald tests. The degrees of freedom for a t test is described in the section "Degrees of Freedom" on page 10064. You can specify the confidence level by using the ALPHA= option.

CORRB

displays the correlation matrix of the parameter estimates.

COVB

displays the covariance matrix of the parameter estimates.

DF=*types* **<(***value***)>**

determines the denominator degrees of freedom (df) for F statistics in hypothesis testing, as well as the degrees of freedom in t tests for parameter estimates and odds ratio estimates, and for computing t distribution percentiles for confidence limits of these estimates.

You can specify *type* to be DESIGN, INFINITY, or PARMADJ. When you specify DF=DESIGN or DF=PARMADJ, you can optionally specify a positive *value* in parentheses to overwrite the default design degrees of freedom.

DF=PARMADJ is the default for the Taylor variance estimation method, and DF=DESIGN is the default for the replication variance estimation method.

For more information, see the section "Degrees of Freedom" on page 10064.

If you specify both DF=DESIGN(*value*) in the MODEL statement and the DF= option in a REPWEIGHTS statement, PROC SURVEYLOGISTIC uses the *value* in DF=DESIGN(*value*) in the MODEL statement to determine the *df* and ignores the one in the REPWEIGHTS statement.

You can specify one of the following *types*:

DESIGN

DESIGN < (*value*) >

specifies the *df* to be the design degrees of freedom. If you specify a positive *value* in DF=DESIGN(*value*), then *df*=*value*.

If you specify DF=DESIGN without the optional positive *value*, then *df* is determined as the design degrees of freedom.

For more information, see the section "Degrees of Freedom" on page 10064.

INFINITY

NONE

specifies that the *df* is infinite. As the denominator degrees of freedom grows, an *F* distribution approaches a chi-square distribution, and similarly a *t* distribution approaches a normal distribution. Therefore, when you specify DF=INFINITY, PROC SURVEYLOGISTIC uses chi-square tests and normal distribution percentiles to construct confidence intervals.

PARMADJ

PARMADJ < (*value*) >

requests that the *df* be modified as *f*–*r*+*1*, where *f* is the default design degrees of freedom or the *value* specified in this option, and *r* is the rank of the contrast of model parameters to be tested.

This option applies only when the Taylor variance estimation method is used (either by default or when you specify VARMETHOD=TAYLOR). This option can be useful when you have many parameters relative to the default design degrees of freedom.

EXPB

EXPEST

displays the exponentiated values ($e^{\hat{\theta}_i}$) of the parameter estimates $\hat{\theta}_i$ in the "Analysis of Maximum Likelihood Estimates" table for the logit model. These exponentiated values are the estimated odds ratios for the parameters corresponding to the continuous explanatory variables.

FCONV=*value*

specifies the relative function convergence criterion. Convergence requires a small relative change in the log-likelihood function in subsequent iterations:

$$\frac{|l^{(i)} - l^{(i-1)}|}{|l^{(i-1)}| + 1\text{E–}6} < value$$

where $l^{(i)}$ is the value of the log likelihood at iteration i. See the section "Convergence Criteria" on page 10048 for details.

GCONV=*value*

specifies the relative gradient convergence criterion. Convergence requires that the normalized prediction function reduction is small:

$$\frac{\mathbf{g}^{(i)\prime}\mathbf{I}^{(i)}\mathbf{g}^{(i)}}{|l^{(i)}| + 1\text{E–}6} < value$$

where $l^{(i)}$ is the value of the log-likelihood function, $\mathbf{g}^{(i)}$ is the gradient vector, and $\mathbf{I}^{(i)}$ the (expected) information matrix. All of these functions are evaluated at iteration i. This is the default convergence criterion, and the default value is 1E–8. For more information, see the section "Convergence Criteria" on page 10048.

GRADIENT

displays the gradient vector, which is evaluated at the global null hypothesis.

ITPRINT

displays the iteration history of the maximum-likelihood model fitting. The ITPRINT option also displays the last evaluation of the gradient vector and the final change in the $-2\log L$.

LINK=*keyword*

L=*keyword*

specifies the link function that links the response probabilities to the linear predictors. You can specify one of the following *keywords*. The default is LINK=LOGIT.

CLOGLOG specifies the complementary log-log function. PROC SURVEYLOGISTIC fits the binary complementary log-log model for binary response and fits the cumulative complementary log-log model when there are more than two response categories. Aliases: CCLOGLOG, CCLL, CUMCLOGLOG.

GLOGIT specifies the generalized logit function. PROC SURVEYLOGISTIC fits the generalized logit model where each nonreference category is contrasted with the reference category. You can use the response variable option REF= to specify the reference category.

LOGIT specifies the cumulative logit function. PROC SURVEYLOGISTIC fits the binary logit model when there are two response categories and fits the cumulative logit model when there are more than two response categories. Aliases: CLOGIT, CUMLOGIT.

PROBIT specifies the inverse standard normal distribution function. PROC SURVEYLOGISTIC fits the binary probit model when there are two response categories and fits the cumulative probit model when there are more than two response categories. Aliases: NORMIT, CPROBIT, CUMPROBIT.

See the section "Link Functions and the Corresponding Distributions" on page 10045 for details.

MAXITER=*n*

specifies the maximum number of iterations to perform. By default, MAXITER=25. If convergence is not attained in *n* iterations, the displayed output created by the procedure contains results that are based on the last maximum likelihood iteration.

NOCHECK

disables the checking process to determine whether maximum likelihood estimates of the regression parameters exist. If you are sure that the estimates are finite, this option can reduce the execution time when the estimation takes more than eight iterations. For more information, see the section "Existence of Maximum Likelihood Estimates" on page 10048.

NODUMMYPRINT

suppresses the "Class Level Information" table, which shows how the design matrix columns for the CLASS variables are coded.

NOINT

suppresses the intercept for the binary response model or the first intercept for the ordinal response model.

OFFSET=*name*

names the offset variable. The regression coefficient for this variable is fixed at 1.

PARMLABEL

displays the labels of the parameters in the "Analysis of Maximum Likelihood Estimates" table.

RIDGING=ABSOLUTE | RELATIVE | NONE

specifies the technique used to improve the log-likelihood function when its value in the current iteration is less than that in the previous iteration. If you specify the RIDGING=ABSOLUTE option, the diagonal elements of the negative (expected) Hessian are inflated by adding the ridge value. If you specify the RIDGING=RELATIVE option, the diagonal elements are inflated by a factor of 1 plus the ridge value. If you specify the RIDGING=NONE option, the crude line search method of taking half a step is used instead of ridging. By default, RIDGING=RELATIVE.

RSQUARE

requests a generalized R^2 measure for the fitted model.

For more information, see the section "Generalized Coefficient of Determination" on page 10050.

SINGULAR=*value*

specifies the tolerance for testing the singularity of the Hessian matrix (Newton-Raphson algorithm) or the expected value of the Hessian matrix (Fisher scoring algorithm). The Hessian matrix is the matrix of second partial derivatives of the log likelihood. The test requires that a pivot for sweeping this matrix be at least this *value* times a norm of the matrix. Values of the SINGULAR= option must be numeric. By default, SINGULAR=10^{-12}.

STB

displays the standardized estimates for the parameters for the continuous explanatory variables in the "Analysis of Maximum Likelihood Estimates" table. The standardized estimate of θ_i is given by $\hat{\theta}_i/(s/s_i)$, where s_i is the total sample standard deviation for the ith explanatory variable and

$$s = \begin{cases} \pi/\sqrt{3} & \text{Logistic} \\ 1 & \text{Normal} \\ \pi/\sqrt{6} & \text{Extreme-value} \end{cases}$$

For the intercept parameters and parameters associated with a CLASS variable, the standardized estimates are set to missing.

TECHNIQUE=FISHER | NEWTON

TECH=FISHER | NEWTON

specifies the optimization technique for estimating the regression parameters. NEWTON (or NR) is the Newton-Raphson algorithm and FISHER (or FS) is the Fisher scoring algorithm. Both techniques yield the same estimates, but the estimated covariance matrices are slightly different except for the case where the LOGIT link is specified for binary response data. The default is TECHNIQUE=FISHER. If the LINK=GLOGIT option is specified, then Newton-Raphson is the default and only available method. See the section "Iterative Algorithms for Model Fitting" on page 10046 for details.

VADJUST=DF | MOREL <(*Morel-options*)> | NONE

specifies an adjustment to the variance estimation for the regression coefficients.

By default, PROC SURVEYLOGISTIC uses the degrees of freedom adjustment VADJUST=DF.

If you do not want to use any variance adjustment, you can specify the VADJUST=NONE option. You can specify the VADJUST=MOREL option for the variance adjustment proposed by Morel (1989).

You can specify the following *Morel-options* within parentheses after the VADJUST=MOREL option:

ADJBOUND=ϕ

sets the upper bound coefficient ϕ in the variance adjustment. This upper bound must be positive. By default, the procedure uses $\phi = 0.5$. See the section "Adjustments to the Variance Estimation" on page 10057 for more details on how this upper bound is used in the variance estimation.

DEFFBOUND=δ

sets the lower bound of the estimated design effect in the variance adjustment. This lower bound must be positive. By default, the procedure uses $\delta = 1$. See the section "Adjustments to the Variance Estimation" on page 10057 for more details about how this lower bound is used in the variance estimation.

XCONV=*value*

specifies the relative parameter convergence criterion. Convergence requires a small relative parameter change in subsequent iterations:

$$\max_j |\delta_j^{(i)}| < value$$

where

$$\delta_j^{(i)} = \begin{cases} \theta_j^{(i)} - \theta_j^{(i-1)} & |\theta_j^{(i-1)}| < 0.01 \\ \frac{\theta_j^{(i)} - \theta_j^{(i-1)}}{\theta_j^{(i-1)}} & \text{otherwise} \end{cases}$$

and $\theta_j^{(i)}$ is the estimate of the jth parameter at iteration i. See the section "Convergence Criteria" on page 10048 for details.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *options* > < / *option* > **;**

The OUTPUT statement creates a new SAS data set that contains all the variables in the input data set and, optionally, the estimated linear predictors and their standard error estimates, the estimates of the cumulative or individual response probabilities, and the confidence limits for the cumulative probabilities. Formulas for the statistics are given in the section "Linear Predictor, Predicted Probability, and Confidence Limits" on page 10074.

If you use the single-trial syntax, the data set also contains a variable named _LEVEL_, which indicates the level of the response that the given row of output is referring to. For example, the value of the cumulative probability variable is the probability that the response variable is as large as the corresponding value of _LEVEL_. For details, see the section "OUT= Data Set in the OUTPUT Statement" on page 10075.

The estimated linear predictor, its standard error estimate, all predicted probabilities, and the confidence limits for the cumulative probabilities are computed for all observations in which the explanatory variables have no missing values, even if the response is missing. By adding observations with missing response values to the input data set, you can compute these statistics for new observations, or for settings of the explanatory variables not present in the data, without affecting the model fit.

Table 120.9 summarizes the *options* available in the OUTPUT statement.

Table 120.9 OUTPUT Statement Options

Option	Description
ALPHA=	Sets the level of significance
LOWER=	Names the variable that contains the lower confidence limits
OUT=	Names the output data set
PREDICTED=	Names the variable that contains the predicted probabilities
PREDPROBS=	Requests predicted probabilities
STDXBETA=	Names the variable that contains the standard error estimates
UPPER=	Names the variable that contains the upper confidence limits
XBETA=	Names the variable that contains the estimates of the linear predictor

You can specify the following *options* in the OUTPUT statement:

LOWER | L=*name*

names the variable that contains the lower confidence limits for π, where π is the probability of the event response if events/trials syntax or the single-trial syntax with binary response is specified; π is cumulative probability (that is, the probability that the response is less than or equal to the value of _LEVEL_) for a cumulative model; and π is the individual probability (that is, the probability that the response category is represented by the value of _LEVEL_) for the generalized logit model. See the ALPHA= option for information about setting the confidence level.

OUT=*SAS-data-set*
names the output data set. If you omit the OUT= option, the output data set is created and given a default name by using the DATA*n* convention.

The statistic options in the OUTPUT statement specify the statistics to be included in the output data set and name the new variables that contain the statistics.

PREDICTED | P=*name*
names the variable that contains the predicted probabilities. For the events/trials syntax or the single-trial syntax with binary response, it is the predicted event probability. For a cumulative model, it is the predicted cumulative probability (that is, the probability that the response variable is less than or equal to the value of _LEVEL_); and for the generalized logit model, it is the predicted individual probability (that is, the probability of the response category represented by the value of _LEVEL_).

PREDPROBS=(*keywords***)**
requests individual, cumulative, or cross validated predicted probabilities. Descriptions of the *keywords* are as follows.

INDIVIDUAL | I requests the predicted probability of each response level. For a response variable Y with three levels, 1, 2, and 3, the individual probabilities are Pr(Y=1), Pr(Y=2), and Pr(Y=3).

CUMULATIVE | C requests the cumulative predicted probability of each response level. For a response variable Y with three levels, 1, 2, and 3, the cumulative probabilities are Pr(Y$\leq$1), Pr(Y$\leq$2), and Pr(Y$\leq$3). The cumulative probability for the last response level always has the constant value of 1. For generalized logit models, the cumulative predicted probabilities are not computed and are set to missing.

CROSSVALIDATE | XVALIDATE | X requests the cross validated individual predicted probability of each response level. These probabilities are derived from the leave-one-out principle; that is, dropping the data of one subject and reestimating the parameter estimates. PROC SURVEYLOGISTIC uses a less expensive one-step approximation to compute the parameter estimates. This option is valid only for binary response models; for nominal and ordinal models, the cross validated probabilities are not computed and are set to missing.

See the section "Details of the PREDPROBS= Option" on page 10034 at the end of this section for further details.

STDXBETA=*name*
names the variable that contains the standard error estimates of XBETA (the definition of which follows).

UPPER | U=*name*
names the variable that contains the upper confidence limits for π, where π is the probability of the event response if events/trials syntax or single-trial syntax with binary response is specified; π is cumulative probability (that is, the probability that the response is less than or equal to the value of _LEVEL_) for a cumulative model; and π is the individual probability (that is, the probability that the response category is represented by the value of _LEVEL_) for the generalized logit model. See the ALPHA= option for information about setting the confidence level.

XBETA=*name*
names the variable that contains the estimates of the linear predictor $\alpha_i + \mathbf{x}\boldsymbol{\beta}$, where i is the corresponding ordered value of _LEVEL_.

You can specify the following *option* in the OUTPUT statement after a slash (/):

ALPHA=*value*
sets the level of significance α for $100(1-\alpha)\%$ confidence limits for the appropriate response probabilities. The value α must be between 0 and 1. By default, α is equal to the value of the ALPHA= option in the PROC SURVEYLOGISTIC statement, or 0.05 if the ALPHA= option is not specified.

Details of the PREDPROBS= Option

You can request any of the three given types of predicted probabilities. For example, you can request both the individual predicted probabilities and the cross validated probabilities by specifying PREDPROBS=(I X).

When you specify the PREDPROBS= option, two automatic variables _FROM_ and _INTO_ are included for the single-trial syntax and only one variable, _INTO_, is included for the events/trials syntax. The _FROM_ variable contains the formatted value of the observed response. The variable _INTO_ contains the formatted value of the response level with the largest individual predicted probability.

If you specify PREDPROBS=INDIVIDUAL, the OUTPUT data set contains k additional variables representing the individual probabilities, one for each response level, where k is the maximum number of response levels across all BY groups. The names of these variables have the form IP_*xxx*, where *xxx* represents the particular level. The representation depends on the following situations:

- If you specify the events/trials syntax, *xxx* is either Event or Nonevent. Thus, the variable that contains the event probabilities is named IP_Event and the variable containing the nonevent probabilities is named IP_Nonevent.
- If you specify the single-trial syntax with more than one BY group, *xxx* is 1 for the first-ordered level of the response, 2 for the second ordered level of the response, and so forth, as given in the "Response Profile" table. The variable that contains the predicted probabilities Pr(Y=1) is named IP_1, where Y is the response variable. Similarly, IP_2 is the name of the variable containing the predicted probabilities Pr(Y=2), and so on.
- If you specify the single-trial syntax with no BY-group processing, *xxx* is the left-justified formatted value of the response level (the value can be truncated so that IP_*xxx* does not exceed 32 characters). For example, if Y is the response variable with response levels 'None,' 'Mild,' and 'Severe,' the variables representing individual probabilities Pr(Y='None'), Pr(Y='Mild'), and Pr(Y='Severe') are named IP_None, IP_Mild, and IP_Severe, respectively.

If you specify PREDPROBS=CUMULATIVE, the OUTPUT data set contains k additional variables that represent the cumulative probabilities, one for each response level, where k is the maximum number of response levels across all BY groups. The names of these variables have the form CP_*xxx*, where *xxx* represents the particular response level. The naming convention is similar to that given by PREDPROBS=INDIVIDUAL. The PREDPROBS=CUMULATIVE values are the same as those output by the PREDICT=keyword, but they are arranged in variables in each output observation rather than in multiple output observations.

If you specify PREDPROBS=CROSSVALIDATE, the OUTPUT data set contains k additional variables representing the cross validated predicted probabilities of the k response levels, where k is the maximum number

of response levels across all BY groups. The names of these variables have the form XP_*xxx*, where *xxx* represents the particular level. The representation is the same as that given by PREDPROBS=INDIVIDUAL, except that for the events/trials syntax there are four variables for the cross validated predicted probabilities instead of two:

XP_EVENT_R1E is the cross validated predicted probability of an event when a current event trial is removed.

XP_NONEVENT_R1E is the cross validated predicted probability of a nonevent when a current event trial is removed.

XP_EVENT_R1N is the cross validated predicted probability of an event when a current nonevent trial is removed.

XP_NONEVENT_R1N is the cross validated predicted probability of a nonevent when a current nonevent trial is removed.

REPWEIGHTS Statement

REPWEIGHTS *variables* < / *options* > **;**

The REPWEIGHTS statement names *variables* that provide replicate weights for bootstrap, BRR, or jackknife variance estimation, which you request with the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option, respectively, in the PROC SURVEYLOGISTIC statement. For more information about the replication methods, see the section "Variance Estimation" on page 10056.

Each REPWEIGHTS *variable* contains the weights for a single replicate, and the number of replicates equals the number of REPWEIGHTS variables. The REPWEIGHTS variables must be numeric, and the variable values must be nonnegative numbers.

For more information about replicate weights that the SURVEYLOGISTIC procedure creates, see the sections "Balanced Repeated Replication (BRR) Method" on page 10060 and "Jackknife Method" on page 10062.

If you provide replicate weights with a REPWEIGHTS statement, you do not need to specify a CLUSTER or STRATA statement. If you use a REPWEIGHTS statement and do not specify the VARMETHOD= option in the PROC SURVEYLOGISTIC statement, the procedure uses VARMETHOD=JACKKNIFE by default.

If you specify a REPWEIGHTS statement but do not include a WEIGHT statement, the procedure uses the average of replicate weights of each observation as the observation's weight.

You can specify the following *options* in the REPWEIGHTS statement after a slash (/):

DF=*df*

specifies the degrees of freedom for the analysis. The value of *df* must be a positive number. By default, the value of *df* is the number of REPWEIGHTS variables.

If you know the number of PSUs and the number of strata from which the replicate weights are generated, you should specify the number of PSUs minus the number of strata as the degrees of freedom in this option.

JKCOEFS=*value* | **<(>***values* **<)>** | *SAS-data-set*

specifies jackknife coefficients for the VARMETHOD=JACKKNIFE option in the PROC SURVEYLOGISTIC statement. The jackknife coefficient values must be nonnegative numbers. For more information about jackknife coefficients, see the section "Jackknife Method" on page 10062.

You can provide jackknife coefficients by specifying one of the following forms:

value

specifies a single jackknife coefficient *value* to use for all replicates, where *value* must be a nonnegative number.

values

specifies a list of jackknife coefficients, where each value in the *values* is a nonnegative number that corresponds to a single replicate that is identified by a REPWEIGHTS variable. You can separate the values with blanks or commas, and you can optionally enclose the *values* in parentheses. The number of values in the *values* must equal the number of replicate weight variables that you specify in the REPWEIGHTS statement.

You must list the jackknife coefficient values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

SAS-data-set

names a *SAS-data-set* that contains the jackknife coefficients, where each coefficient value must be a nonnegative number. You must provide the jackknife coefficients in the data set variable named JKCoefficient. Each observation in this data set must correspond to a replicate that is identified by a REPWEIGHTS variable. The number of observations in the *SAS-data-set* must not be less than the number of REPWEIGHTS variables.

REPCOEFS=*value* | **<(>***values* **<)>** | *SAS-data-set*

specifies replicate coefficients for the VARMETHOD=BOOTSTRAP or VARMETHOD=JACKKNIFE option in the PROC SURVEYLOGISTIC statement, where each coefficient corresponds to an individual replicate weight that is identified by a REPWEIGHTS variable. The replicate coefficient values must be nonnegative numbers.

You can provide replicate coefficients by specifying one of the following forms:

value

specifies a single replicate coefficient *value* to use for all replicates, where *value* must be a nonnegative number.

values

specifies a list of replicate coefficients, where each value in the *values* is a nonnegative number that corresponds to a single replicate that is identified by a REPWEIGHTS variable. You can separate the values with blanks or commas, and you can optionally enclose the *values* in parentheses. The number of values in the *values* must equal the number of replicate weight variables that you specify in the REPWEIGHTS statement.

You must list the replicate coefficient values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

SAS-data-set

names a *SAS-data-set* that contains the replicate coefficients, where each coefficient value must be a nonnegative number. You must provide the replicate coefficients in the data set variable named JKCoefficient. Each observation in this data set must correspond to a replicate that is identified by a REPWEIGHTS variable. The number of observations in the *SAS-data-set* must not be less than the number of REPWEIGHTS variables.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

STRATA Statement

STRATA *variables* < / *option* > **;**

The STRATA statement specifies variables that form the strata in a stratified sample design. The combinations of categories of STRATA variables define the strata in the sample.

If your sample design has stratification at multiple stages, you should identify only the first-stage strata in the STRATA statement. See the section "Specification of Population Totals and Sampling Rates" on page 10051 for more information.

If you provide replicate weights for BRR or jackknife variance estimation with the REPWEIGHTS statement, you do not need to specify a STRATA statement.

The STRATA *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the STRATA variables determine the levels. Thus, you can use formats to group values into levels. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference* for more information.

When determining levels of a STRATA variable, an observation with missing values for this STRATA variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10040.

You can use multiple STRATA statements to specify stratum variables.

You can specify the following *option* in the STRATA statement after a slash (/):

LIST

displays a "Stratum Information" table, which includes values of the STRATA variables and the number of observations, number of clusters, population total, and sampling rate for each stratum. See the section "Stratum Information" on page 10079 for more details.

TEST Statement

< *label*: > **TEST** *equation1* < , *equation2*, . . . > < / *option* > ;

The TEST statement tests linear hypotheses about the regression coefficients. The Wald test is used to jointly test the null hypotheses (H_0: $\mathbf{L}\boldsymbol{\theta} = \mathbf{c}$) specified in a single TEST statement. When $\mathbf{c} = \mathbf{0}$ you should specify a CONTRAST statement instead.

Each *equation* specifies a linear hypothesis (a row of the $\mathbf{L}$ matrix and the corresponding element of the $\mathbf{c}$ vector); multiple *equations* are separated by commas. The label, which must be a valid SAS name, is used to identify the resulting output and should always be included. You can submit multiple TEST statements.

The form of an *equation* is as follows:

term < ± *term* . . . > < = ± *term* < ± *term* . . . >>

where *term* is a parameter of the model, or a constant, or a constant times a parameter. For a binary response model, the intercept parameter is named INTERCEPT; for an ordinal response model, the intercept parameters are named INTERCEPT, INTERCEPT2, INTERCEPT3, and so on. When no equal sign appears, the expression is set to 0. The following illustrates possible uses of the TEST statement:

```
proc surveylogistic;
   model y= a1 a2 a3 a4;
   test1: test intercept + .5 * a2 = 0;
   test2: test intercept + .5 * a2;
   test3: test a1=a2=a3;
   test4: test a1=a2, a2=a3;
run;
```

Note that the first and second TEST statements are equivalent, as are the third and fourth TEST statements.

You can specify the following *option* in the TEST statement after a slash (/):

PRINT

displays intermediate calculations in the testing of the null hypothesis H_0: $\mathbf{L}\boldsymbol{\theta} = \mathbf{c}$. This includes $\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}})\mathbf{L}'$ bordered by $(\mathbf{L}\hat{\boldsymbol{\theta}} - \mathbf{c})$ and $[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}})\mathbf{L}']^{-1}$ bordered by $[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}})\mathbf{L}']^{-1}(\mathbf{L}\hat{\boldsymbol{\theta}} - \mathbf{c})$, where $\hat{\boldsymbol{\theta}}$ is the pseudo-estimator of $\boldsymbol{\theta}$ and $\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}})$ is the estimated covariance matrix of $\hat{\boldsymbol{\theta}}$.

For more information, see the section "Testing Linear Hypotheses about the Regression Coefficients" on page 10069.

UNITS Statement

UNITS *independent1* **=** *list1* < . . . *independentk* **=** *listk* > < / *option* > **;**

The UNITS statement enables you to specify units of change for the continuous explanatory variables so that customized odds ratios can be estimated. An estimate of the corresponding odds ratio is produced for each unit of change specified for an explanatory variable. The UNITS statement is ignored for CLASS variables. If the CLODDS option is specified in the MODEL statement, the corresponding confidence intervals for the odds ratios are also displayed.

The term *independent* is the name of an explanatory variable, and *list* represents a list of units of change, separated by spaces, that are of interest for that variable. Each unit of change in a list has one of the following forms:

- *number*
- **SD** or –**SD**
- *number* * **SD**

where *number* is any nonzero number and SD is the sample standard deviation of the corresponding independent variable. For example, $X = -2$ requests an odds ratio that represents the change in the odds when the variable X is decreased by two units. $X = 2*\text{SD}$ requests an estimate of the change in the odds when X is increased by two sample standard deviations.

You can specify the following *option* in the UNITS statement after a slash (/):

DEFAULT=*list*

gives a list of units of change for all explanatory variables that are not specified in the UNITS statement. Each unit of change can be in any of the forms described previously. If the DEFAULT= option is not specified, PROC SURVEYLOGISTIC does not produce customized odds ratio estimates for any explanatory variable that is not listed in the UNITS statement.

For more information, see the section "Odds Ratio Estimation" on page 10070.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement names the variable that contains the sampling weights. This variable must be numeric, and the sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, then the procedure omits that observation from the analysis. See the section "Missing Values" on page 10040 for more information. If you specify more than one WEIGHT statement, the procedure uses only the first WEIGHT statement and ignores the rest.

If you do not specify a WEIGHT statement but provide replicate weights with a REPWEIGHTS statement, PROC SURVEYLOGISTIC uses the average of replicate weights of each observation as the observation's weight.

If you do not specify a WEIGHT statement or a REPWEIGHTS statement, PROC SURVEYLOGISTIC assigns all observations a weight of one.

Details: SURVEYLOGISTIC Procedure

Missing Values

If you have missing values in your survey data for any reason, such as nonresponse, this can compromise the quality of your survey results. If the respondents are different from the nonrespondents with regard to a survey effect or outcome, then survey estimates might be biased and cannot accurately represent the survey population. There are a variety of techniques in sample design and survey operations that can reduce nonresponse. After data collection is complete, you can use imputation to replace missing values with acceptable values, and/or you can use sampling weight adjustments to compensate for nonresponse. You should complete this data preparation and adjustment before you analyze your data with PROC SURVEYLOGISTIC. For more information, see Cochran (1977); Kalton and Kasprzyk (1986); Brick and Kalton (1996).

If an observation has a missing value or a nonpositive value for the WEIGHT or FREQ variable, then that observation is excluded from the analysis.

An observation is also excluded if it has a missing value for any design (STRATA, CLUSTER, or DOMAIN) variable, unless you specify the MISSING option in the PROC SURVEYLOGISTIC statement. If you specify the MISSING option, the procedure treats missing values as a valid (nonmissing) category for all categorical variables.

By default, if an observation contains missing values for the response, offset, or any explanatory variables used in the independent effects, the observation is excluded from the analysis. This treatment is based on the assumption that the missing values are missing completely at random (MCAR). However, this assumption is not true sometimes. For example, evidence from other surveys might suggest that observations with missing values are systematically different from observations without missing values. If you believe that missing values are not missing completely at random, then you can specify the NOMCAR option to include these observations with missing values in the dependent variable and the independent variables in the variance estimation.

Whether or not the NOMCAR option is used, observations with missing or invalid values for WEIGHT, FREQ, STRATA, CLUSTER, or DOMAIN variables are always excluded, unless the MISSING option is also specified.

When you specify the NOMCAR option, the procedure treats observations with and without missing values for variables in the regression model as two different domains, and it performs a domain analysis in the domain of nonmissing observations.

If you use a REPWEIGHTS statement, all REPWEIGHTS variables must contain nonmissing values.

Model Specification

Response Level Ordering

Response level ordering is important because, by default, PROC SURVEYLOGISTIC models the probabilities of response levels with lower *Ordered Values.* Ordered Values, displayed in the "Response Profile" table, are assigned to response levels in ascending sorted order. That is, the lowest response level is assigned Ordered Value 1, the next lowest is assigned Ordered Value 2, and so on. For example, if your response variable Y takes values in $\{1, \ldots, D+1\}$, then the functions of the response probabilities modeled with the cumulative model are

$$\text{logit}(\Pr(Y \leq i|\mathbf{x})), i = 1, \ldots, D$$

and for the generalized logit model they are

$$\log\left(\frac{\Pr(Y = i|\mathbf{x})}{\Pr(Y = D+1|\mathbf{x})}\right), i = 1, \ldots, D$$

where the highest Ordered Value $Y = D + 1$ is the reference level. You can change these default functions by specifying the EVENT=, REF=, DESCENDING, or ORDER= response variable options in the MODEL statement.

For binary response data with event and nonevent categories, the procedure models the function

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right)$$

where p is the probability of the response level assigned to Ordered Value 1 in the "Response Profiles" table. Since

$$\text{logit}(p) = -\text{logit}(1-p)$$

the effect of reversing the order of the two response levels is to change the signs of α and $\boldsymbol{\beta}$ in the model $\text{logit}(p) = \alpha + \mathbf{x}\boldsymbol{\beta}$.

If your event category has a higher Ordered Value than the nonevent category, the procedure models the nonevent probability. You can use response variable options to model the event probability. For example, suppose the binary response variable Y takes the values 1 and 0 for event and nonevent, respectively, and Exposure is the explanatory variable. By default, the procedure assigns Ordered Value 1 to response level Y=0, and Ordered Value 2 to response level Y=1. Therefore, the procedure models the probability of the nonevent (Ordered Value=1) category. To model the event probability, you can do the following:

- Explicitly state which response level is to be modeled by using the response variable option EVENT= in the MODEL statement:

```
model Y(event='1') = Exposure;
```

- Specify the response variable option DESCENDING in the MODEL statement:

```
model Y(descending)=Exposure;
```

- Specify the response variable option REF= in the MODEL statement as the nonevent category for the response variable. This option is most useful when you are fitting a generalized logit model.

```
model Y(ref='0') = Exposure;
```

- Assign a format to Y such that the first formatted value (when the formatted values are put in sorted order) corresponds to the event. For this example, Y=1 is assigned formatted value 'event' and Y=0 is assigned formatted value 'nonevent.' Since ORDER= FORMATTED by default, Ordered Value 1 is assigned to response level Y=1 so the procedure models the event.

```
proc format;
   value Disease 1='event' 0='nonevent';
run;

proc surveylogistic;
   format Y Disease.;
   model Y=Exposure;
run;
```

CLASS Variable Parameterization

Consider a model with one CLASS variable A with four levels: 1, 2, 5, and 7. Details of the possible choices for the PARAM= option follow.

EFFECT Three columns are created to indicate group membership of the nonreference levels. For the reference level, all three dummy variables have a value of –1. For instance, if the reference level is 7 (REF=7), the design matrix columns for A are as follows.

	Design Matrix		
A	**A1**	**A2**	**A5**
1	1	0	0
2	0	1	0
5	0	0	1
7	–1	–1	–1

For CLASS main effects that use the EFFECT coding scheme, individual parameters correspond to the difference between the effect of each nonreference level and the average over all four levels.

GLM As in PROC GLM, four columns are created to indicate group membership. The design matrix columns for A are as follows.

	Design Matrix			
A	**A1**	**A2**	**A5**	**A7**
1	1	0	0	0
2	0	1	0	0
5	0	0	1	0
7	0	0	0	1

For CLASS main effects that use the GLM coding scheme, individual parameters correspond to the difference between the effect of each level and the last level.

ORDINAL Three columns are created to indicate group membership of the higher levels of the effect. For the first level of the effect (which for A is 1), all three dummy variables have a value of 0. The design matrix columns for A are as follows.

	Design Matrix		
A	**A2**	**A5**	**A7**
1	0	0	0
2	1	0	0
5	1	1	0
7	1	1	1

The first level of the effect is a control or baseline level.

For CLASS main effects that use the ORDINAL coding scheme, the first level of the effect is a control or baseline level; individual parameters correspond to the difference between effects of the current level and the preceding level. When the parameters for an ordinal main effect have the same sign, the response effect is monotonic across the levels.

POLYNOMIAL | POLY Three columns are created. The first represents the linear term (x), the second represents the quadratic term (x^2), and the third represents the cubic term (x^3), where x is the level value. If the CLASS levels are not numeric, they are translated into 1, 2, 3, . . . according to their sort order. The design matrix columns for A are as follows.

	Design Matrix		
A	**APOLY1**	**APOLY2**	**APOLY3**
1	1	1	1
2	2	4	8
5	5	25	125
7	7	49	343

REFERENCE | REF Three columns are created to indicate group membership of the nonreference levels. For the reference level, all three dummy variables have a value of 0. For instance, if the reference level is 7 (REF=7), the design matrix columns for A are as follows.

	Design Matrix		
A	**A1**	**A2**	**A5**
1	1	0	0
2	0	1	0
5	0	0	1
7	0	0	0

For CLASS main effects that use the REFERENCE coding scheme, individual parameters correspond to the difference between the effect of each nonreference level and the reference level.

ORTHEFFECT The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=EFFECT. The design matrix columns for A are as follows.

	Design Matrix		
A	**AOEFF1**	**AOEFF2**	**AOEFF3**
1	1.41421	–0.81650	–0.57735
2	0.00000	1.63299	–0.57735
5	0.00000	0.00000	1.73205
7	–1.41421	–0.81649	–0.57735

ORTHORDINAL | ORTHOTHERM The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=ORDINAL. The design matrix columns for A are as follows.

	Design Matrix		
A	**AOORD1**	**AOORD2**	**AOORD3**
1	–1.73205	0.00000	0.00000
2	0.57735	–1.63299	0.00000
5	0.57735	0.81650	–1.41421
7	0.57735	0.81650	1.41421

ORTHPOLY The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=POLY. The design matrix columns for A are as follows.

	Design Matrix		
A	**AOPOLY1**	**AOPOLY2**	**AOPOLY5**
1	–1.153	0.907	–0.921
2	–0.734	–0.540	1.473
5	0.524	–1.370	–0.921
7	1.363	1.004	0.368

ORTHREF The columns are obtained by applying the Gram-Schmidt orthogonalization to the columns for PARAM=REFERENCE. The design matrix columns for A are as follows.

	Design Matrix		
A	**AOREF1**	**AOREF2**	**AOREF3**
1	1.73205	0.00000	0.00000
2	–0.57735	1.63299	0.00000
5	–0.57735	–0.81650	1.41421
7	–0.57735	–0.81650	–1.41421

Link Functions and the Corresponding Distributions

Four link functions are available in the SURVEYLOGISTIC procedure. The logit function is the default. To specify a different link function, use the LINK= option in the MODEL statement. The link functions and the corresponding distributions are as follows:

- The **logit** function

  $$g(p) = \log\left(\frac{p}{1-p}\right)$$

  is the inverse of the cumulative logistic distribution function, which is

  $$F(x) = \frac{1}{1+e^{-x}}$$

- The **probit** (or normit) function

  $$g(p) = \Phi^{-1}(p)$$

  is the inverse of the cumulative standard normal distribution function, which is

  $$F(x) = \Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{1}{2}z^2}dz$$

  Traditionally, the probit function includes an additive constant 5, but throughout PROC SURVEYLOGISTIC, the terms probit and normit are used interchangeably, previously defined as $g(p)$.

- The **complementary log-log** function

  $$g(p) = \log(-\log(1-p))$$

  is the inverse of the cumulative extreme-value function (also called the Gompertz distribution), which is

  $$F(x) = 1 - e^{-e^{x}}$$

- The **generalized logit** function extends the binary logit link to a vector of levels $(\pi_1, \ldots, \pi_{k+1})$ by contrasting each level with a fixed level

  $$g(\pi_i) = \log\left(\frac{\pi_i}{\pi_{k+1}}\right) \quad i = 1, \ldots, k$$

The variances of the normal, logistic, and extreme-value distributions are not the same. Their respective means and variances are

Distribution	Mean	Variance
Normal	0	1
Logistic	0	$\pi^2/3$
Extreme-value	$-\gamma$	$\pi^2/6$

where γ is the Euler constant. In comparing parameter estimates that use different link functions, you need to take into account the different scalings of the corresponding distributions and, for the complementary log-log function, a possible shift in location. For example, if the fitted probabilities are in the neighborhood of 0.1 to 0.9, then the parameter estimates from using the logit link function should be about $\pi/\sqrt{3} \approx 1.8$ larger than the estimates from the probit link function.

Model Fitting

Determining Observations for Likelihood Contributions

If you use the events/trials syntax, each observation is split into two observations. One has the response value 1 with a frequency equal to the value of the *events* variable. The other observation has the response value 2 and a frequency equal to the value of (*trials* – *events*). These two observations have the same explanatory variable values and the same WEIGHT values as the original observation.

For either the single-trial or the events/trials syntax, let j index all observations. In other words, for the single-trial syntax, j indexes the actual observations. And, for the events/trials syntax, j indexes the observations after splitting (as described previously). If your data set has 30 observations and you use the single-trial syntax, j has values from 1 to 30; if you use the events/trials syntax, j has values from 1 to 60.

Suppose the response variable in a cumulative response model can take on the ordered values $1, \ldots, k, k+1$, where k is an integer ≥ 1. The likelihood for the jth observation with ordered response value y_j and explanatory variables vector (row vectors) $\mathbf{x}_j$ is given by

$$L_j = \begin{cases} F(\alpha_1 + \mathbf{x}_j\boldsymbol{\beta}) & y_j = 1 \\ F(\alpha_i + \mathbf{x}_j\boldsymbol{\beta}) - F(\alpha_{i-1} + \mathbf{x}_j\boldsymbol{\beta}) & 1 < y_j = i \leq k \\ 1 - F(\alpha_k + \mathbf{x}_j\boldsymbol{\beta}) & y_j = k+1 \end{cases}$$

where $F(.)$ is the logistic, normal, or extreme-value distribution function; $\alpha_1, \ldots, \alpha_k$ are ordered intercept parameters; and $\boldsymbol{\beta}$ is the slope parameter vector.

For the generalized logit model, letting the $k+1$st level be the reference level, the intercepts $\alpha_1, \ldots, \alpha_k$ are unordered and the slope vector $\boldsymbol{\beta}_i$ varies with each logit. The likelihood for the jth observation with ordered response value y_j and explanatory variables vector $\mathbf{x}_j$ (row vectors) is given by

$$\begin{aligned} L_j &= \Pr(Y = y_j | \mathbf{x}_j) \\ &= \begin{cases} \dfrac{e^{\alpha_i + \mathbf{x}_j\boldsymbol{\beta}_i}}{1 + \sum_{i=1}^{k} e^{\alpha_i + \mathbf{x}_j\boldsymbol{\beta}_i}} & 1 \leq y_j = i \leq k \\ \dfrac{1}{1 + \sum_{i=1}^{k} e^{\alpha_i + \mathbf{x}_j\boldsymbol{\beta}_i}} & y_j = k+1 \end{cases} \end{aligned}$$

Iterative Algorithms for Model Fitting

Two iterative maximum likelihood algorithms are available in PROC SURVEYLOGISTIC to obtain the pseudo-estimate $\hat{\boldsymbol{\theta}}$ of the model parameter $\boldsymbol{\theta}$. The default is the Fisher scoring method, which is equivalent to fitting by iteratively reweighted least squares. The alternative algorithm is the Newton-Raphson method. Both algorithms give the same parameter estimates; the covariance matrix of $\hat{\boldsymbol{\theta}}$ is estimated in the section "Variance Estimation" on page 10056. For a generalized logit model, only the Newton-Raphson technique is available. You can use the TECHNIQUE= option in the MODEL statement to select a fitting algorithm.

Iteratively Reweighted Least Squares Algorithm (Fisher Scoring)

Let Y be the response variable that takes values $1, \ldots, k, k+1$ ($k \geq 1$). Let j index all observations and Y_j be the value of response for the jth observation. Consider the multinomial variable $\mathbf{Z}_j = (Z_{1j}, \ldots, Z_{kj})'$ such that

$$Z_{ij} = \begin{cases} 1 & \text{if } Y_j = i \\ 0 & \text{otherwise} \end{cases}$$

and $Z_{(k+1)j} = 1 - \sum_{i=1}^{k} Z_{ij}$. With π_{ij} denoting the probability that the jth observation has response value i, the expected value of $\mathbf{Z}_j$ is $\boldsymbol{\pi}_j = (\pi_{1j}, \ldots, \pi_{kj})'$, and $\pi_{(k+1)j} = 1 - \sum_{i=1}^{k} \pi_{ij}$. The covariance matrix of $\mathbf{Z}_j$ is $\mathbf{V}_j$, which is the covariance matrix of a multinomial random variable for one trial with parameter vector $\boldsymbol{\pi}_j$. Let $\boldsymbol{\theta}$ be the vector of regression parameters—for example, $\boldsymbol{\theta} = (\alpha_1, \ldots, \alpha_k, \boldsymbol{\beta}')'$ for cumulative logit model. Let $\mathbf{D}_j$ be the matrix of partial derivatives of $\boldsymbol{\pi}_j$ with respect to $\boldsymbol{\theta}$. The estimating equation for the regression parameters is

$$\sum_j \mathbf{D}_j' \mathbf{W}_j (\mathbf{Z}_j - \boldsymbol{\pi}_j) = \mathbf{0}$$

where $\mathbf{W}_j = w_j f_j \mathbf{V}_j^{-1}$, and w_j and f_j are the WEIGHT and FREQ values of the jth observation.

With a starting value of $\boldsymbol{\theta}^{(0)}$, the pseudo-estimate of $\boldsymbol{\theta}$ is obtained iteratively as

$$\boldsymbol{\theta}^{(i+1)} = \boldsymbol{\theta}^{(i)} + (\sum_j \mathbf{D}_j' \mathbf{W}_j \mathbf{D}_j)^{-1} \sum_j \mathbf{D}_j' \mathbf{W}_j (\mathbf{Z}_j - \boldsymbol{\pi}_j)$$

where $\mathbf{D}_j$, $\mathbf{W}_j$, and $\boldsymbol{\pi}_j$ are evaluated at the ith iteration $\boldsymbol{\theta}^{(i)}$. The expression after the plus sign is the step size. If the log likelihood evaluated at $\boldsymbol{\theta}^{(i+1)}$ is less than that evaluated at $\boldsymbol{\theta}^{(i)}$, then $\boldsymbol{\theta}^{(i+1)}$ is recomputed by step-halving or ridging. The iterative scheme continues until convergence is obtained—that is, until $\boldsymbol{\theta}^{(i+1)}$ is sufficiently close to $\boldsymbol{\theta}^{(i)}$. Then the maximum likelihood estimate of $\boldsymbol{\theta}$ is $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(i+1)}$.

By default, starting values are zero for the slope parameters, and starting values are the observed cumulative logits (that is, logits of the observed cumulative proportions of response) for the intercept parameters. Alternatively, the starting values can be specified with the INEST= option in the PROC SURVEYLOGISTIC statement.

Newton-Raphson Algorithm

Let

$$\begin{aligned} \mathbf{g} &= \sum_j w_j f_j \frac{\partial l_j}{\partial \boldsymbol{\theta}} \\ \mathbf{H} &= \sum_j -w_j f_j \frac{\partial^2 l_j}{\partial \boldsymbol{\theta}^2} \end{aligned}$$

be the gradient vector and the Hessian matrix, where $l_j = \log L_j$ is the log likelihood for the jth observation. With a starting value of $\boldsymbol{\theta}^{(0)}$, the pseudo-estimate $\hat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}$ is obtained iteratively until convergence is obtained:

$$\boldsymbol{\theta}^{(i+1)} = \boldsymbol{\theta}^{(i)} + \mathbf{H}^{-1}\mathbf{g}$$

where $\mathbf{H}$ and $\mathbf{g}$ are evaluated at the ith iteration $\boldsymbol{\theta}^{(i)}$. If the log likelihood evaluated at $\boldsymbol{\theta}^{(i+1)}$ is less than that evaluated at $\boldsymbol{\theta}^{(i)}$, then $\boldsymbol{\theta}^{(i+1)}$ is recomputed by step-halving or ridging. The iterative scheme continues until convergence is obtained—that is, until $\boldsymbol{\theta}^{(i+1)}$ is sufficiently close to $\boldsymbol{\theta}^{(i)}$. Then the maximum likelihood estimate of $\boldsymbol{\theta}$ is $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(i+1)}$.

Convergence Criteria

Four convergence criteria are allowed: ABSFCONV=, FCONV=, GCONV=, and XCONV=. If you specify more than one convergence criterion, the optimization is terminated as soon as one of the criteria is satisfied. If none of the criteria is specified, the default is GCONV=1E–8.

Existence of Maximum Likelihood Estimates

The likelihood equation for a logistic regression model does not always have a finite solution. Sometimes there is a nonunique maximum on the boundary of the parameter space, at infinity. The existence, finiteness, and uniqueness of pseudo-estimates for the logistic regression model depend on the patterns of data points in the observation space (Albert and Anderson 1984; Santner and Duffy 1986).

Consider a binary response model. Let Y_j be the response of the ith subject, and let $\mathbf{x}_j$ be the row vector of explanatory variables (including the constant 1 associated with the intercept). There are three mutually exclusive and exhaustive types of data configurations: complete separation, quasi-complete separation, and overlap.

Complete separation

There is a complete separation of data points if there exists a vector $\mathbf{b}$ that correctly allocates all observations to their response groups; that is,

$$\begin{cases} \mathbf{x}_j\mathbf{b} > 0 & Y_j = 1 \\ \mathbf{x}_j\mathbf{b} < 0 & Y_j = 2 \end{cases}$$

This configuration gives nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, the log likelihood diminishes to zero, and the dispersion matrix becomes unbounded.

Quasi-complete separation

The data are not completely separable, but there is a vector $\mathbf{b}$ such that

$$\begin{cases} \mathbf{x}_j\mathbf{b} \geq 0 & Y_j = 1 \\ \mathbf{x}_j\mathbf{b} \leq 0 & Y_j = 2 \end{cases}$$

and equality holds for at least one subject in each response group. This configuration also yields nonunique infinite estimates. If the iterative process of maximizing the likelihood function is allowed to continue, the dispersion matrix becomes unbounded and the log likelihood diminishes to a nonzero constant.

Overlap

If neither complete nor quasi-complete separation exists in the sample points, there is an overlap of sample points. In this configuration, the pseudo-estimates exist and are unique.

Complete separation and quasi-complete separation are problems typically encountered with small data sets. Although complete separation can occur with any type of data, quasi-complete separation is not likely with truly continuous explanatory variables.

The SURVEYLOGISTIC procedure uses a simple empirical approach to recognize the data configurations that lead to infinite parameter estimates. The basis of this approach is that any convergence method of maximizing the log likelihood must yield a solution that gives complete separation, if such a solution exists. In maximizing the log likelihood, there is no checking for complete or quasi-complete separation if convergence is attained in eight or fewer iterations. Subsequent to the eighth iteration, the probability of the observed response is computed for each observation. If the probability of the observed response is one for all observations, there is a complete separation of data points and the iteration process is stopped. If the complete separation of data has not been determined and an observation is identified to have an extremely large probability ($\geq$0.95) of the observed response, there are two possible situations. First, there is overlap in the data set, and the observation is an atypical observation of its own group. The iterative process, if allowed to continue, stops when a maximum is reached. Second, there is quasi-complete separation in the data set, and the asymptotic dispersion matrix is unbounded. If any of the diagonal elements of the dispersion matrix for the standardized observations vectors (all explanatory variables standardized to zero mean and unit variance) exceeds 5,000, quasi-complete separation is declared and the iterative process is stopped. If either complete separation or quasi-complete separation is detected, a warning message is displayed in the procedure output.

Checking for quasi-complete separation is less foolproof than checking for complete separation. The NOCHECK option in the MODEL statement turns off the process of checking for infinite parameter estimates. In cases of complete or quasi-complete separation, turning off the checking process typically results in the procedure failing to converge.

Model Fitting Statistics

Suppose the model contains s explanatory effects. For the jth observation, let $\hat{\pi}_j$ be the estimated probability of the observed response. The three criteria displayed by the SURVEYLOGISTIC procedure are calculated as follows:

- –2 log likelihood:

  $$-2 \text{ Log L} = -2 \sum_j w_j f_j \log(\hat{\pi}_j)$$

  where w_j and f_j are the weight and frequency values, respectively, of the jth observation. For binary response models that use the events/trials syntax, this is equivalent to

  $$-2 \text{ Log L} = -2 \sum_j w_j f_j \{r_j \log(\hat{\pi}_j) + (n_j - r_j) \log(1 - \hat{\pi}_j)\}$$

  where r_j is the number of events, n_j is the number of trials, and $\hat{\pi}_j$ is the estimated event probability.

- Akaike information criterion:

  $$\text{AIC} = -2 \text{ Log L} + 2p$$

  where p is the number of parameters in the model. For cumulative response models, $p = k + s$, where k is the total number of response levels minus one, and s is the number of explanatory effects. For the generalized logit model, $p = k(s + 1)$.

- Schwarz criterion:

$$\mathrm{SC} = -2\ \mathrm{Log\ L} + p \log(\sum_j w_j f_j)$$

where p is the number of parameters in the model. For cumulative response models, $p = k + s$, where k is the total number of response levels minus one, and s is the number of explanatory effects. For the generalized logit model, $p = k(s + 1)$.

The –2 log likelihood statistic has a chi-square distribution under the null hypothesis (that all the explanatory effects in the model are zero), and the procedure produces a p-value for this statistic. The AIC and SC statistics give two different ways of adjusting the –2 log likelihood statistic for the number of terms in the model and the number of observations used.

Generalized Coefficient of Determination

Cox and Snell (1989, pp. 208–209) propose the following generalization of the coefficient of determination to a more general linear model:

$$R^2 = 1 - \left\{ \frac{L(\mathbf{0})}{L(\hat{\boldsymbol{\theta}})} \right\}^{\frac{2}{N}}$$

where $L(\mathbf{0})$ is the likelihood of the intercept-only model, $L(\hat{\boldsymbol{\theta}})$ is the likelihood of the specified model, and N is the population size. The quantity R^2 achieves a maximum of less than 1 for discrete models, where the maximum is given by

$$R^2_{\max} = 1 - \{L(\mathbf{0})\}^{\frac{2}{N}}$$

Nagelkerke (1991) proposes the following adjusted coefficient, which can achieve a maximum value of 1:

$$\tilde{R}^2 = \frac{R^2}{R^2_{\max}}$$

Properties and interpretation of R^2 and $\tilde{R}^2$ are provided in Nagelkerke (1991). In the "Testing Global Null Hypothesis: BETA=0" table, R^2 is labeled as "RSquare" and $\tilde{R}^2$ is labeled as "Max-rescaled RSquare." Use the RSQUARE option to request R^2 and $\tilde{R}^2$.

INEST= Data Set

You can specify starting values for the iterative algorithm in the INEST= data set.

The INEST= data set contains one observation for each BY group. The INEST= data set must contain the intercept variables (named Intercept for binary response models and Intercept, Intercept2, Intercept3, and so forth, for ordinal response models) and all explanatory variables in the MODEL statement. If BY processing is used, the INEST= data set should also include the BY variables, and there must be one observation for each BY group. If the INEST= data set also contains the _TYPE_ variable, only observations with _TYPE_ value 'PARMS' are used as starting values.

Survey Design Information

Specification of Population Totals and Sampling Rates

To include a finite population correction (*fpc*) in Taylor series or bootstrap variance estimation, you can input either the sampling rate or the population total by using the RATE= or TOTAL= option, respectively, in the PROC SURVEYLOGISTIC statement. (You cannot specify both of these options in the same PROC SURVEYLOGISTIC statement.) The RATE= and TOTAL= options apply only to Taylor series or bootstrap variance estimation. The procedure does not use a finite population correction for BRR or jackknife variance estimation.

If you do not specify the RATE= or TOTAL= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. For fairly small sampling fractions, it is appropriate to ignore this correction. For more information, see Cochran (1977) and Kish (1965).

If your design has multiple stages of selection and you are specifying the RATE= option, you should input the first-stage sampling rate, which is the ratio of the number of primary sampling units (PSUs) in the sample to the total number of PSUs in the study population. If you are specifying the TOTAL= option for a multistage design, you should input the total number of PSUs in the study population. For more information, see the section "Primary Sampling Units (PSUs)" on page 10052.

For a nonstratified sample design, or for a stratified sample design that has the same sampling rate or the same population total in all strata, you can use the RATE=*value* or TOTAL=*value* option. If your sample design is stratified with different sampling rates or population totals in different strata, use the RATE=*SAS-data-set* or TOTAL=*SAS-data-set* option to name a SAS data set that contains the stratum sampling rates or totals. This data set is called a *secondary data set*, as opposed to the *primary data set* that you specify in the DATA= option.

The secondary data set must contain all the stratification variables that are listed in the STRATA statement and all the variables that are listed in the BY statement. If there are formats associated with the STRATA variables and the BY variables, then the formats must be consistent in the primary and the secondary data sets. If you specify the TOTAL=*SAS-data-set* option, the secondary data set must have a variable named _TOTAL_ that contains the stratum population totals. Or if you specify the RATE=*SAS-data-set* option, the secondary data set must have a variable named _RATE_ that contains the stratum sampling rates. If the secondary data set contains more than one observation for any one stratum, then the procedure uses the first value of _TOTAL_ or _RATE_ for that stratum and ignores the rest.

The *value* in the RATE= option or the values of _RATE_ in the secondary data set must be nonnegative numbers. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC SURVEYLOGISTIC converts that number to a proportion. The procedure treats the value 1 as 100% instead of 1%.

If you specify the TOTAL=*value* option, *value* must not be less than the sample size. If you provide stratum population totals in a secondary data set, these values must not be less than the corresponding stratum sample sizes.

Primary Sampling Units (PSUs)

When you have clusters, or primary sampling units (PSUs), in your sample design, the procedure estimates variance from the variation among PSUs when the Taylor series variance method is used. For more information, see the section "Taylor Series (Linearization)" on page 10056.

BRR or jackknife variance estimation methods draw multiple replicates (also called subsamples) from the full sample by following a specific resampling scheme. These subsamples are constructed by deleting PSUs from the full sample.

The bootstrap variance estimation method repeatedly draws a simple random sample with replacement of primary sampling units (PSUs) from the full sample to form each replicate.

If you use a REPWEIGHTS statement to provide replicate weights for BRR, jackknife, bootstrap variance estimation, you do not need to specify a CLUSTER statement. Otherwise, you should specify a CLUSTER statement whenever your design includes clustering at the first stage of sampling. If you do not specify a CLUSTER statement, then PROC SURVEYLOGISTIC treats each observation as a PSU.

Logistic Regression Models and Parameters

The SURVEYLOGISTIC procedure fits a logistic regression model and estimates the corresponding regression parameters. Each model uses the link function you specified in the LINK= option in the MODEL statement. There are four types of model you can use with the procedure: cumulative logit model, complementary log-log model, probit model, and generalized logit model.

Notation

Let Y be the response variable with categories $1, 2, \ldots, D, D+1$. The p covariates are denoted by a p-dimension row vector $\mathbf{x}$.

For a stratified clustered sample design, each observation is represented by a row vector, $(w_{hij}, \mathbf{y}'_{hij}, y_{hij(D+1)}, \mathbf{x}_{hij})$, where

- $h = 1, 2, \ldots, H$ is the stratum index
- $i = 1, 2, \ldots, n_h$ is the cluster index within stratum h
- $j = 1, 2, \ldots, m_{hi}$ is the unit index within cluster i of stratum h
- w_{hij} denotes the sampling weight
- $\mathbf{y}_{hij}$ is a D-dimensional column vector whose elements are indicator variables for the first D categories for variable Y. If the response of the jth unit of the ith cluster in stratum h falls in category d, the dth element of the vector is one, and the remaining elements of the vector are zero, where $d = 1, 2, \ldots, D$.
- $y_{hij(D+1)}$ is the indicator variable for the $(D+1)$ category of variable Y
- $\mathbf{x}_{hij}$ denotes the k-dimensional row vector of explanatory variables for the jth unit of the ith cluster in stratum h. If there is an intercept, then $x_{hij1} \equiv 1$.
- $\tilde{n} = \sum_{h=1}^{H} n_h$ is the total number of clusters in the sample

- $n = \sum_{h=1}^{H} \sum_{i=1}^{n_h} m_{hi}$ is the total sample size

The following notations are also used:

- f_h denotes the sampling rate for stratum h
- $\boldsymbol{\pi}_{hij}$ is the expected vector of the response variable:

$$\begin{aligned} \boldsymbol{\pi}_{hij} &= E(\mathbf{y}_{hij}|\mathbf{x}_{hij}) \\ &= (\pi_{hij1}, \pi_{hij2}, \ldots, \pi_{hijD})' \\ \pi_{hij(D+1)} &= E(y_{hij(D+1)}|\mathbf{x}_{hij}) \end{aligned}$$

Note that $\pi_{hij(D+1)} = 1 - \mathbf{1}'\boldsymbol{\pi}_{hij}$, where $\mathbf{1}$ is a D-dimensional column vector whose elements are 1.

Logistic Regression Models

If the response categories of the response variable Y can be restricted to a number of ordinal values, you can fit cumulative probabilities of the response categories with a cumulative logit model, a complementary log-log model, or a probit model. Details of cumulative logit models (or proportional odds models) can be found in McCullagh and Nelder (1989). If the response categories of Y are nominal responses without natural ordering, you can fit the response probabilities with a generalized logit model. Formulation of the generalized logit models for nominal response variables can be found in Agresti (2002). For each model, the procedure estimates the model parameter $\boldsymbol{\theta}$ by using a pseudo-log-likelihood function. The procedure obtains the pseudo-maximum likelihood estimator $\hat{\boldsymbol{\theta}}$ by using iterations described in the section "Iterative Algorithms for Model Fitting" on page 10046 and estimates its variance described in the section "Variance Estimation" on page 10056.

Cumulative Logit Model

A cumulative logit model uses the **logit** function

$$g(t) = \log\left(\frac{t}{1-t}\right)$$

as the link function.

Denote the cumulative sum of the expected proportions for the first d categories of variable Y by

$$F_{hijd} = \sum_{r=1}^{d} \pi_{hijr}$$

for $d = 1, 2, \ldots, D$. Then the cumulative logit model can be written as

$$\log\left(\frac{F_{hijd}}{1-F_{hijd}}\right) = \alpha_d + \mathbf{x}_{hij}\boldsymbol{\beta}$$

with the model parameters

$$\begin{aligned} \boldsymbol{\beta} &= (\beta_1, \beta_2, \ldots, \beta_k)' \\ \boldsymbol{\alpha} &= (\alpha_1, \alpha_2, \ldots, \alpha_D)', \quad \alpha_1 < \alpha_2 < \cdots < \alpha_D \\ \boldsymbol{\theta} &= (\boldsymbol{\alpha}', \boldsymbol{\beta}')' \end{aligned}$$

Complementary Log-Log Model

A complementary log-log model uses the **complementary log-log** function

$$g(t) = \log(-\log(1-t))$$

as the link function. Denote the cumulative sum of the expected proportions for the first d categories of variable Y by

$$F_{hijd} = \sum_{r=1}^{d} \pi_{hijr}$$

for $d = 1, 2, \ldots, D$. Then the complementary log-log model can be written as

$$\log(-\log(1 - F_{hijd})) = \alpha_d + \mathbf{x}_{hij}\boldsymbol{\beta}$$

with the model parameters

$$\begin{aligned}
\boldsymbol{\beta} &= (\beta_1, \beta_2, \ldots, \beta_k)' \\
\boldsymbol{\alpha} &= (\alpha_1, \alpha_2, \ldots, \alpha_D)', \quad \alpha_1 < \alpha_2 < \cdots < \alpha_D \\
\boldsymbol{\theta} &= (\boldsymbol{\alpha}', \boldsymbol{\beta}')'
\end{aligned}$$

Probit Model

A probit model uses the **probit** (or normit) function, which is the inverse of the cumulative standard normal distribution function,

$$g(t) = \Phi^{-1}(t)$$

as the link function, where

$$\Phi(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{t} e^{-\frac{1}{2}z^2} dz$$

Denote the cumulative sum of the expected proportions for the first d categories of variable Y by

$$F_{hijd} = \sum_{r=1}^{d} \pi_{hijr}$$

for $d = 1, 2, \ldots, D$. Then the probit model can be written as

$$F_{hijd} = \Phi(\alpha_d + \mathbf{x}_{hij}\boldsymbol{\beta})$$

with the model parameters

$$\begin{aligned}
\boldsymbol{\beta} &= (\beta_1, \beta_2, \ldots, \beta_k)' \\
\boldsymbol{\alpha} &= (\alpha_1, \alpha_2, \ldots, \alpha_D)', \quad \alpha_1 < \alpha_2 < \cdots < \alpha_D \\
\boldsymbol{\theta} &= (\boldsymbol{\alpha}', \boldsymbol{\beta}')'
\end{aligned}$$

Generalized Logit Model

For nominal response, a generalized logit model is to fit the ratio of the expected proportion for each response category over the expected proportion of a reference category with a logit link function.

Without loss of generality, let category $D+1$ be the reference category for the response variable Y. Denote the expected proportion for the dth category by π_{hijd} as in the section "Notation" on page 10052. Then the generalized logit model can be written as

$$\log\left(\frac{\pi_{hijd}}{\pi_{hij(D+1)}}\right) = \mathbf{x}_{hij}\boldsymbol{\beta}_d$$

for $d = 1, 2, \ldots, D$, with the model parameters

$$\begin{aligned}\boldsymbol{\beta}_d &= (\beta_{d1}, \beta_{d2}, \ldots, \beta_{dk})' \\ \boldsymbol{\theta} &= (\boldsymbol{\beta}_1', \boldsymbol{\beta}_2', \ldots, \boldsymbol{\beta}_D')'\end{aligned}$$

Likelihood Function

Let $\mathbf{g}(\cdot)$ be a link function such that

$$\boldsymbol{\pi} = \mathbf{g}(\mathbf{x}, \boldsymbol{\theta})$$

where $\boldsymbol{\theta}$ is a column vector for regression coefficients. The pseudo-log likelihood is

$$l(\boldsymbol{\theta}) = \sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}\left((\log(\boldsymbol{\pi}_{hij}))'\mathbf{y}_{hij} + \log(\pi_{hij(D+1)})y_{hij(D+1)}\right)$$

Denote the pseudo-estimator as $\hat{\boldsymbol{\theta}}$, which is a solution to the estimating equations:

$$\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}\mathbf{D}_{hij}\left(\mathrm{diag}(\boldsymbol{\pi}_{hij}) - \boldsymbol{\pi}_{hij}\boldsymbol{\pi}_{hij}'\right)^{-1}(\mathbf{y}_{hij} - \boldsymbol{\pi}_{hij}) = \mathbf{0}$$

where $\mathbf{D}_{hij}$ is the matrix of partial derivatives of the link function $\mathbf{g}$ with respect to $\boldsymbol{\theta}$.

To obtain the pseudo-estimator $\hat{\boldsymbol{\theta}}$, the procedure uses iterations with a starting value $\boldsymbol{\theta}^{(0)}$ for $\boldsymbol{\theta}$. See the section "Iterative Algorithms for Model Fitting" on page 10046 for more details.

Variance Estimation

Due to the variability of characteristics among items in the population, researchers apply scientific sample designs in the sample selection process to reduce the risk of a distorted view of the population, and they make inferences about the population based on the information from the sample survey data. In order to make statistically valid inferences for the population, they must incorporate the sample design in the data analysis.

The SURVEYLOGISTIC procedure fits linear logistic regression models for discrete response survey data by using the maximum likelihood method. In the variance estimation, the procedure uses the Taylor series (linearization) method or replication (resampling) methods to estimate sampling errors of estimators based on complex sample designs, including designs with stratification, clustering, and unequal weighting (Binder 1981, 1983; Roberts, Rao, and Kumar 1987; Skinner, Holt, and Smith 1989; Binder and Roberts 2003; Morel 1989; Lehtonen and Pahkinen 1995; Woodruff 1971; Fuller 1975; Särndal, Swensson, and Wretman 1992; Fuller 2009; Wolter 2007; Rust 1985; Dippo, Fay, and Morganstein 1984; Rao and Shao 1999; Rao, Wu, and Yue 1992; Rao and Shao 1996).

You can use the VARMETHOD= option to specify a variance estimation method to use. By default, the Taylor series method is used. However, replication methods have recently gained popularity for estimating variances in complex survey data analysis. One reason for this popularity is the relative simplicity of replication-based estimates, especially for nonlinear estimators; another is that modern computational capacity has made replication methods feasible for practical survey analysis.

Replication methods draw multiple replicates (also called subsamples) from a full sample according to a specific resampling scheme. The most commonly used resampling schemes are the *balanced repeated replication* (BRR) method, the *jackknife* method, and the *bootstrap* method. For each replicate, the original weights are modified for the PSUs in the replicates in order to create replicate weights. The parameters of interest are estimated by using the replicate weights for each replicate. Then the variances of parameters of interest are estimated by the variability among the estimates derived from these replicates. You can use the REPWEIGHTS statement to provide your own replicate weights for variance estimation.

The following sections provide details about how the variance-covariance matrix of the estimated regression coefficients is estimated for each variance estimation method.

Taylor Series (Linearization)

The Taylor series (linearization) method is the most commonly used method to estimate the covariance matrix of the regression coefficients for complex survey data. It is the default variance estimation method used by PROC SURVEYLOGISTIC.

Using the notation described in the section "Notation" on page 10052, the estimated covariance matrix of model parameters $\hat{\boldsymbol{\theta}}$ by the Taylor series method is

$$\hat{V}(\hat{\boldsymbol{\theta}}) = \hat{\mathbf{Q}}^{-1}\hat{\mathbf{G}}\hat{\mathbf{Q}}^{-1}$$

where

$$
\begin{aligned}
\hat{\mathbf{Q}} &= \sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}\hat{\mathbf{D}}_{hij}\left(\mathrm{diag}(\hat{\boldsymbol{\pi}}_{hij}) - \hat{\boldsymbol{\pi}}_{hij}\hat{\boldsymbol{\pi}}'_{hij}\right)^{-1}\hat{\mathbf{D}}'_{hij} \\
\hat{\mathbf{G}} &= \frac{n-1}{n-p}\sum_{h=1}^{H}\frac{n_h(1-f_h)}{n_h-1}\sum_{i=1}^{n_h}(\mathbf{e}_{hi\cdot} - \bar{\mathbf{e}}_{h\cdot\cdot})(\mathbf{e}_{hi\cdot} - \bar{\mathbf{e}}_{h\cdot\cdot})' \\
\mathbf{e}_{hi\cdot} &= \sum_{j=1}^{m_{hi}} w_{hij}\hat{\mathbf{D}}_{hij}\left(\mathrm{diag}(\hat{\boldsymbol{\pi}}_{hij}) - \hat{\boldsymbol{\pi}}_{hij}\hat{\boldsymbol{\pi}}'_{hij}\right)^{-1}(\mathbf{y}_{hij} - \hat{\boldsymbol{\pi}}_{hij}) \\
\bar{\mathbf{e}}_{h\cdot\cdot} &= \frac{1}{n_h}\sum_{i=1}^{n_h}\mathbf{e}_{hi\cdot}
\end{aligned}
$$

and $\mathbf{D}_{hij}$ is the matrix of partial derivatives of the inverse link function $\mathbf{g}$ with respect to $\boldsymbol{\theta}$ and $\hat{\mathbf{D}}_{hij}$ and the response probabilities $\hat{\boldsymbol{\pi}}_{hij}$ are evaluated at $\hat{\boldsymbol{\theta}}$.

If you specify the TECHNIQUE=NEWTON option in the MODEL statement to request the Newton-Raphson algorithm, the matrix $\hat{\mathbf{Q}}$ is replaced by the negative (expected) Hessian matrix when the estimated covariance matrix $\hat{V}(\hat{\boldsymbol{\theta}})$ is computed.

Adjustments to the Variance Estimation

The factor $(n-1)/(n-p)$ in the computation of the matrix $\hat{\mathbf{G}}$ reduces the small sample bias associated with using the estimated function to calculate deviations (Morel 1989; Hidiroglou, Fuller, and Hickman 1980). For simple random sampling, this factor contributes to the degrees-of-freedom correction applied to the residual mean square for ordinary least squares in which p parameters are estimated. By default, the procedure uses this adjustment in Taylor series variance estimation. It is equivalent to specifying the VADJUST=DF option in the MODEL statement. If you do not want to use this multiplier in the variance estimation, you can specify the VADJUST=NONE option in the MODEL statement to suppress this factor.

In addition, you can specify the VADJUST=MOREL option to request an adjustment to the variance estimator for the model parameters $\hat{\boldsymbol{\theta}}$, introduced by Morel (1989):

$$\hat{V}(\hat{\boldsymbol{\theta}}) = \hat{\mathbf{Q}}^{-1}\hat{\mathbf{G}}\hat{\mathbf{Q}}^{-1} + \kappa\lambda\hat{\mathbf{Q}}^{-1}$$

where for given nonnegative constants δ and ϕ,

$$
\begin{aligned}
\kappa &= \max\left(\delta,\ p^{-1}\mathrm{tr}\left(\hat{\mathbf{Q}}^{-1}\hat{\mathbf{G}}\right)\right) \\
\lambda &= \min\left(\phi, \frac{p}{\tilde{n}-p}\right)
\end{aligned}
$$

The adjustment $\kappa\lambda\hat{\mathbf{Q}}^{-1}$ does the following:

- reduces the small sample bias reflected in inflated Type I error rates
- guarantees a positive-definite estimated covariance matrix provided that $\hat{\mathbf{Q}}^{-1}$ exists

- is close to zero when the sample size becomes large

In this adjustment, κ is an estimate of the design effect, which has been bounded below by the positive constant δ. You can use DEFFBOUND=δ in the VADJUST=MOREL option in the MODEL statement to specify this lower bound; by default, the procedure uses $\delta = 1$. The factor λ converges to zero when the sample size becomes large, and λ has an upper bound ϕ. You can use ADJBOUND=ϕ in the VADJUST=MOREL option in the MODEL statement to specify this upper bound; by default, the procedure uses $\phi = 0.5$.

Bootstrap Method

The VARMETHOD=BOOTSTRAP option in the PROC SURVEYLOGISTIC statement requests the bootstrap method for variance estimation. This method can be used for stratified sample designs and for designs that have no stratification. If your design is stratified, the bootstrap method requires at least two PSUs in each stratum. You can provide your own bootstrap replicate weights for the analysis by using a REPWEIGHTS statement, or the procedure can construct bootstrap replicate weights for the analysis.

PROC SURVEYLOGISTIC estimates the parameter of interest (or requested statistics) from each replicate, and then uses the variability among replicate estimates to estimate the overall variance of these statistics.

This bootstrap method for complex survey data is similar to the method of Rao, Wu, and Yue (1992) and is also known as the bootstrap weights method (Mashreghi, Haziza, and Léger 2016). For more information, see Lohr (2022, Section 9.3.3), Wolter (2007, Chapter 5), Beaumont and Patak (2012), Fuller (2009, Section 4.5), and Shao and Tu (1995, Section 6.2.4). McCarthy and Snowden (1985), Rao and Wu (1988), Sitter (1992b), and Sitter (1992a) provide several adjusted bootstrap variance estimators that are consistent for complex survey data. The naive bootstrap variance estimator that is suitable for infinite populations is not consistent for complex survey data.

Replicate Weight Construction

If you do not provide replicate weights by using a REPWEIGHTS statement, PROC SURVEYLOGISTIC constructs bootstrap replicate weights for the analysis. The procedure selects replicate bootstrap samples by with-replacement random sampling of PSUs within strata. You can specify the number of bootstrap replicates in the REPS= *method-option*; by default, the number of replicates is 250. (Increasing the number of replicates can improve the estimation precision but also increases the computation time.) You can specify the bootstrap sample sizes m_h in the MH= *method-option*; by default, $m_h = n_h - 1$, where n_h is the number of PSUs in stratum h.

In each replicate sample, the original sampling weights of the selected units are adjusted to reflect the full sample. These adjusted weights are the *bootstrap replicate weights*. In replicate r, the bootstrap replicate weight for observation j in PSU i in stratum h is computed as

$$\tilde{w}_{hij}^{(r)} = \left\{ 1 + \sqrt{\frac{1 - f_h}{m_h(n_h - 1)}} n_h k_{hi}^{(r)} - \sqrt{\frac{m_h(1 - f_h)}{n_h - 1}} \right\} w_{hij}$$

where $k_{hi}^{(r)}$ is the number of times PSU i is selected for replicate r, and f_h is the sampling fraction in stratum h that you can input by using either the RATE= or TOTAL= option in the PROC SURVEYLOGISTIC statement.

You can use the OUTWEIGHTS=*SAS-data-set* method-option to store the bootstrap replicate weights in a SAS data set. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate

Weights Output Data Set" on page 10076. You can provide these replicate weights to the procedure for subsequent analyses by using a REPWEIGHTS statement.

Variance Estimation

Let R be the total number of bootstrap replicates weights. Denote α_r as the replicate coefficient for the rth replicate ($r = 1, 2, \ldots, R$).

When the procedure generates the bootstrap replicate weights, $\alpha_r = 1/R$.

If you provide your own bootstrap replicate weights by including a REPWEIGHTS statement, you can specify α_r in the REPCOEFS= option. By default, $\alpha_r = 1/R$.

Let $\hat{\boldsymbol{\theta}}$ be the estimated regression coefficients from the full sample for $\boldsymbol{\theta}$, and let $\hat{\boldsymbol{\theta}}_r$ be the estimated regression coefficient when the rth set of bootstrap replicate weights are used. PROC SURVEYLOGISTIC estimates the covariance matrix of $\hat{\boldsymbol{\theta}}$ by the following equation:

$$\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\theta}}_r - \hat{\boldsymbol{\theta}}\right)\left(\hat{\boldsymbol{\theta}}_r - \hat{\boldsymbol{\theta}}\right)'$$

Here, the degrees of freedom is the number of clusters minus the number of strata. If there are no clusters, then the degrees of freedom equals the number of observations minus the number of strata. If the design is not stratified, then the degrees of freedom equals the number of PSUs minus one.

If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYLOGISTIC computes the covariance matrix of $\hat{\boldsymbol{\theta}}$ by

$$\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\theta}}_r - \overline{\hat{\boldsymbol{\theta}}}\right)\left(\hat{\boldsymbol{\theta}}_r - \overline{\hat{\boldsymbol{\theta}}}\right)'$$

where $\overline{\hat{\boldsymbol{\theta}}}$ is the average of the replicate estimates and is calculated as follows:

$$\overline{\hat{\boldsymbol{\theta}}} = \frac{1}{R}\sum_{r=1}^{R} \hat{\boldsymbol{\theta}}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom and if R' is less than the number of PSUs minus the number of strata, then the degrees of freedom is set to R'.

Balanced Repeated Replication (BRR) Method

The balanced repeated replication (BRR) method requires that the full sample be drawn by using a stratified sample design with two primary sampling units (PSUs) per stratum. Let H be the total number of strata. The total number of replicates R is the smallest multiple of 4 that is greater than H. However, if you prefer a larger number of replicates, you can specify the REPS=*number* option. If a *number* × *number* Hadamard matrix cannot be constructed, the number of replicates is increased until a Hadamard matrix becomes available.

Each replicate is obtained by deleting one PSU per stratum according to the corresponding Hadamard matrix and adjusting the original weights for the remaining PSUs. The new weights are called replicate weights.

Replicates are constructed by using the first H columns of the $R \times R$ Hadamard matrix. The rth ($r = 1, 2, \ldots, R$) replicate is drawn from the full sample according to the rth row of the Hadamard matrix as follows:

- If the (r, h) element of the Hadamard matrix is 1, then the first PSU of stratum h is included in the rth replicate and the second PSU of stratum h is excluded.
- If the (r, h) element of the Hadamard matrix is –1, then the second PSU of stratum h is included in the rth replicate and the first PSU of stratum h is excluded.

Note that the "first" and "second" PSUs are determined by data order in the input data set. Thus, if you reorder the data set and perform the same analysis by using BRR method, you might get slightly different results, because the contents in each replicate sample might change.

The replicate weights of the remaining PSUs in each half-sample are then doubled to their original weights. For more information about the BRR method, see Wolter (2007) and Lohr (2022).

By default, an appropriate Hadamard matrix is generated automatically to create the replicates. You can request that the Hadamard matrix be displayed by specifying the VARMETHOD=BRR(PRINTH) *method-option*. If you provide a Hadamard matrix by specifying the VARMETHOD=BRR(HADAMARD=) *method-option*, then the replicates are generated according to the provided Hadamard matrix.

You can use the VARMETHOD=BRR(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

Let $\hat{\boldsymbol{\theta}}$ be the estimated regression coefficients from the full sample for $\boldsymbol{\theta}$, and let $\hat{\boldsymbol{\theta}}_r$ be the estimated regression coefficient from the rth replicate by using replicate weights. PROC SURVEYLOGISTIC estimates the covariance matrix of $\hat{\boldsymbol{\theta}}$ by

$$\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\theta}}_r - \hat{\boldsymbol{\theta}}\right)\left(\hat{\boldsymbol{\theta}}_r - \hat{\boldsymbol{\theta}}\right)'$$

with H degrees of freedom, where H is the number of strata. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYLOGISTIC computes the covariance matrix of $\hat{\boldsymbol{\theta}}$ by

$$\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\theta}}_r - \overline{\hat{\boldsymbol{\theta}}}\right)\left(\hat{\boldsymbol{\theta}}_r - \overline{\hat{\boldsymbol{\theta}}}\right)'$$

where $\overline{\hat{\boldsymbol{\theta}}}$ is the average of the replicate estimates and is calculated as follows:

$$\overline{\hat{\boldsymbol{\theta}}} = \frac{1}{R}\sum_{r=1}^{R}\hat{\boldsymbol{\theta}}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals the minimum between R' and the number of strata, H.

Fay's BRR Method

Fay's method is a modification of the BRR method, and it requires a stratified sample design with two primary sampling units (PSUs) per stratum. The total number of replicates R is the smallest multiple of 4 that is greater than the total number of strata H. However, if you prefer a larger number of replicates, you can specify the REPS= *method-option*.

For each replicate, Fay's method uses a Fay coefficient $0 \le \epsilon < 1$ to impose a perturbation of the original weights in the full sample that is gentler than using only half-samples, as in the traditional BRR method. The Fay coefficient $0 \le \epsilon < 1$ can be set by specifying the FAY = ϵ *method-option*. By default, $\epsilon = 0.5$ if the FAY *method-option* is specified without providing a value for ϵ (Judkins 1990; Rao and Shao 1999). When $\epsilon = 0$, Fay's method becomes the traditional BRR method. For more information, see Dippo, Fay, and Morganstein (1984); Fay (1984, 1989); Judkins (1990).

Let H be the number of strata. Replicates are constructed by using the first H columns of the $R \times R$ Hadamard matrix, where R is the number of replicates, $R > H$. The rth ($r = 1, 2, \ldots, R$) replicate is created from the full sample according to the rth row of the Hadamard matrix as follows:

- If the (r, h) element of the Hadamard matrix is 1, then the full sample weight of the first PSU in stratum h is multiplied by ϵ and the full sample weight of the second PSU is multiplied by $2 - \epsilon$ to obtain the rth replicate weights.
- If the (r, h) element of the Hadamard matrix is –1, then the full sample weight of the first PSU in stratum h is multiplied by $2 - \epsilon$ and the full sample weight of the second PSU is multiplied by ϵ to obtain the rth replicate weights.

You can use the VARMETHOD=BRR(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

By default, an appropriate Hadamard matrix is generated automatically to create the replicates. You can request that the Hadamard matrix be displayed by specifying the VARMETHOD=BRR(PRINTH) *method-option*. If you provide a Hadamard matrix by specifying the VARMETHOD=BRR(HADAMARD=) *method-option*, then the replicates are generated according to the provided Hadamard matrix.

Let $\hat{\boldsymbol{\theta}}$ be the estimated regression coefficients from the full sample for $\boldsymbol{\theta}$. Let $\hat{\boldsymbol{\theta}}_r$ be the estimated regression coefficient obtained from the rth replicate by using replicate weights. PROC SURVEYLOGISTIC estimates the covariance matrix of $\hat{\boldsymbol{\theta}}$ by

$$\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}}) = \frac{1}{R(1-\epsilon)^2}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\theta}}_r - \hat{\boldsymbol{\theta}}\right)\left(\hat{\boldsymbol{\theta}}_r - \hat{\boldsymbol{\theta}}\right)'$$

with H degrees of freedom, where H is the number of strata. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYLOGISTIC computes the covariance matrix of $\hat{\boldsymbol{\theta}}$ by

$$\hat{\mathrm{V}}(\hat{\boldsymbol{\theta}}) = \frac{1}{R(1-\epsilon)^2} \sum_{r=1}^{R} \left(\hat{\boldsymbol{\theta}}_r - \overline{\hat{\boldsymbol{\theta}}}\right) \left(\hat{\boldsymbol{\theta}}_r - \overline{\hat{\boldsymbol{\theta}}}\right)'$$

where $\overline{\hat{\boldsymbol{\theta}}}$ is the average of the replicate estimates and is calculated as follows:

$$\overline{\hat{\boldsymbol{\theta}}} = \frac{1}{R} \sum_{r=1}^{R} \hat{\boldsymbol{\theta}}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals the minimum between R' and the number of strata, H.

Jackknife Method

The jackknife method of variance estimation deletes one PSU at a time from the full sample to create replicates. The total number of replicates R is the same as the total number of PSUs. In each replicate, the sample weights of the remaining PSUs are modified by the jackknife coefficient α_r. The modified weights are called replicate weights.

The jackknife coefficient and replicate weights are described as follows.

Without Stratification If there is no stratification in the sample design (no STRATA statement), the jackknife coefficients α_r are the same for all replicates:

$$\alpha_r = \frac{R-1}{R} \quad \text{where } r = 1, 2, \ldots, R$$

Denote the original weight in the full sample for the jth member of the ith PSU as w_{ij}. If the ith PSU is included in the rth replicate ($r = 1, 2, \ldots, R$), then the corresponding replicate weight for the jth member of the ith PSU is defined as

$$w_{ij}^{(r)} = w_{ij}/\alpha_r$$

With Stratification If the sample design involves stratification, each stratum must have at least two PSUs to use the jackknife method.

Let stratum $\tilde{h}_r$ be the stratum from which a PSU is deleted for the rth replicate. Stratum $\tilde{h}_r$ is called the *donor stratum.* Let $n_{\tilde{h}_r}$ be the total number of PSUs in the donor stratum $\tilde{h}_r$. The jackknife coefficients are defined as

$$\alpha_r = \frac{n_{\tilde{h}_r} - 1}{n_{\tilde{h}_r}} \quad \text{where } r = 1, 2, \ldots, R$$

Denote the original weight in the full sample for the *j*th member of the *i*th PSU as w_{ij}. If the *i*th PSU is included in the *r*th replicate ($r = 1, 2, \ldots, R$), then the corresponding replicate weight for the *j*th member of the *i*th PSU is defined as

$$w_{ij}^{(r)} = \begin{cases} w_{ij} & \text{if } i\text{th PSU is not in the donor stratum } \tilde{h}_r \\ w_{ij}/\alpha_r & \text{if } i\text{th PSU is in the donor stratum } \tilde{h}_r \end{cases}$$

You can use the VARMETHOD=JACKKNIFE(OUTJKCOEFS=) *method-option* to save the jackknife coefficients into a SAS data set and use the VARMETHOD=JACKKNIFE(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

If you provide your own replicate weights in a REPWEIGHTS statement, then you can also provide corresponding jackknife coefficients in the JKCOEFS= or REPCOEFS= option. If you provide replicate weights but do not provide jackknife coefficients, PROC SURVEYLOGISTIC uses $\alpha_r = (R-1)/R$ as the jackknife coefficient for all replicates by default.

Let $\hat{\boldsymbol{\theta}}$ be the estimated regression coefficients from the full sample for $\boldsymbol{\theta}$. Let $\hat{\boldsymbol{\theta}}_r$ be the estimated regression coefficient obtained from the *r*th replicate by using replicate weights. PROC SURVEYLOGISTIC estimates the covariance matrix of $\hat{\boldsymbol{\theta}}$ by

$$\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\theta}}_r - \hat{\boldsymbol{\theta}}\right)\left(\hat{\boldsymbol{\theta}}_r - \hat{\boldsymbol{\theta}}\right)'$$

with $R - H$ degrees of freedom, where R is the number of replicates and H is the number of strata, or $R - 1$ when there is no stratification. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYLOGISTIC computes the covariance matrix of $\hat{\boldsymbol{\theta}}$ by

$$\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\theta}}_r - \overline{\hat{\boldsymbol{\theta}}}\right)\left(\hat{\boldsymbol{\theta}}_r - \overline{\hat{\boldsymbol{\theta}}}\right)'$$

where $\overline{\hat{\boldsymbol{\theta}}}$ is the average of the replicate estimates and is calculated as follows:

$$\overline{\hat{\boldsymbol{\theta}}} = \frac{1}{R}\sum_{r=1}^{R} \hat{\boldsymbol{\theta}}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals $R' - H$.

Hadamard Matrix

A Hadamard matrix $\mathbf{H}$ is a square matrix whose elements are either 1 or –1 such that

$$\mathbf{H}\mathbf{H}' = k\mathbf{I}$$

where k is the dimension of $\mathbf{H}$ and $\mathbf{I}$ is the identity matrix of order k. The order k is necessarily 1, 2, or a positive integer that is a multiple of 4.

For example, the following matrix is a Hadamard matrix of dimension $k = 8$:

$$\begin{array}{rrrrrrrr}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{array}$$

Domain Analysis

A DOMAIN statement requests that the procedure perform logistic regression analysis for each domain.

For a domain Ω, let I_Ω be the corresponding indicator variable:

$$I_\Omega(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to } \Omega \\ 0 & \text{otherwise} \end{cases}$$

Let

$$v_{hij} = w_{hij} I_\Omega(h,i,j) = \begin{cases} w_{hij} & \text{if observation } (h,i,j) \text{ belongs to } \Omega \\ 0 & \text{otherwise} \end{cases}$$

The regression in domain Ω uses v as the weight variable.

Hypothesis Testing and Estimation

Degrees of Freedom

In this section, degrees of freedom (df) refers to the denominator degrees of freedom for F statistics in hypothesis testing. It also refers to the degrees of freedom in t tests for parameter estimates and odds ratio estimates, and for computing t distribution percentiles for confidence limits of these estimates. The value of df is determined by the design degrees of freedom f and by what you specify in the DF= option in the MODEL statement.

The default df is determined as

$$df = \begin{cases} f - r + 1 & \text{for Taylor variance estimation method} \\ f & \text{for replication variance estimation methods} \end{cases}$$

where f is the design degrees of freedom and r is the rank of the contrast of model parameters to be tested.

Design Degrees of Freedom f

The design degrees of freedom f is determined by the survey design and the variance estimation method.

Design Degrees of Freedom f for the Taylor Series Method For Taylor series variance estimation, the design degrees of freedom f can depend on the number of clusters, the number of strata, and the number of observations. These numbers are based on the observations that are included in the analysis; they do not count observations that are excluded from the analysis because of missing values. If all values in a stratum are excluded from the analysis as missing values, then that stratum is called an *empty stratum*. Empty strata are not counted in the total number of strata for the analysis. Similarly, empty clusters and missing observations are not included in the totals counts of clusters and observations that are used to compute the f for the analysis.

If you specify the MISSING option in the CLASS statement, missing values are treated as valid nonmissing levels and are included in determining the f. If you specify the NOMCAR option for Taylor series variance estimation, observations that have missing values for variables in the regression model are included. For more information about missing values, see the section "Missing Values" on page 10040.

Using the notation that is defined in the section "Notation" on page 10052, let $\tilde{n}$ be the total number of clusters if the design has a CLUSTER statement; let n be the total sample size; and let H be the number of strata if there is a STRATA statement, or 1 otherwise. Then for Taylor series variance estimation, the design degrees of freedom is

$$f = \begin{cases} \tilde{n} - H & \text{if the design contains clusters} \\ n - H & \text{if the design does not contain clusters} \end{cases}$$

Design Degrees of Freedom f for Replication Methods For replication variance estimation methods that include a REPWEIGHTS statement, the design degrees of freedom f equals the number of REPWEIGHTS variables, unless you specify an alternative in the DF= option in a REPWEIGHTS statement.

When the BRR method (including Fay's method) is requested but no REPWEIGHTS statement is specified, the design degrees of freedom f equals the number of strata.

When the jackknife method or the bootstrap method is requested but no REPWEIGHTS statement is specified, the design degrees of freedom f is the same as in the Taylor series method described in the previous section.

Setting Design Degrees of Freedom f to a Specific Value

If you do not want to use the default design degrees of freedom, then you can specify the DF=DESIGN(*value*) or DF=PARMADJ(*value*) (for Taylor method only) option in the MODEL statement, where *value* is a positive number. Then f=*value*.

However, if you specify the *value* in the DF= option in the MODEL statement as well as with the DF= option in a REPWEIGHTS statement, then the *df* is determined by the *value* in the MODEL statement, and the DF= option in the REPWEIGHTS statement is ignored.

Setting Design Degrees of Freedom to Infinity

If you specify DF=INFINITY in the MODEL statement, then the *df* is set to be infinite.

As the denominator degrees of freedom grows, an F distribution approaches a chi-square distribution, and similarly a t distribution approaches a normal distribution. Therefore, when you specify DF=INFINITY in the MODEL statement), PROC SURVEYLOGISTIC uses chi-square tests and normal distribution percentiles to construct confidence intervals.

Modifying Degrees of Freedom with the Number of Parameters

When you use Taylor series variance estimation (by default or when you specify VARMETHOD=TAYLOR in the MODEL statement), and you are fitting a model that has many parameters relative to the design degrees of freedom, it is appropriate to modify the design degrees of freedom by using the number of nonsingular parameters p in the model (Korn and Graubard (1999, section 5.2), Rao, Scott, and Skinner (1998)). You can specify DF=PARMADJ or DF=PARMADJ(*value*) in the MODEL statement to request this modification only for Taylor series variance estimation method; and this option does not apply to the replication variance estimation method.

Let f be the design degrees of freedom that is described in the section "Design Degrees of Freedom f for the Taylor Series Method" on page 10065. By default, or if you specify the DF=PARMADJ option, the df is modified as $df = f - p + 1$.

Testing Global Null Hypothesis: BETA=0

The *global null hypothesis* refers to the null hypothesis that all the explanatory effects can be eliminated and the model can contain only intercepts. By using the notations in the section "Logistic Regression Models" on page 10053, the global null hypothesis is defined as follows:

- If you have a cumulative model whose model parameters are $\boldsymbol{\theta} = (\boldsymbol{\alpha}', \boldsymbol{\beta}')'$, where $\boldsymbol{\alpha}$ are the parameters for the intercepts and $\boldsymbol{\beta}$ are the parameters for the explanatory effects, then $H_0\colon \boldsymbol{\beta} = \mathbf{0}$. The number of restrictions r that are imposed on $\boldsymbol{\theta}$ is the number of parameters in slope parameter $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_k)'$: $r = k$.
- If you have a generalized logit model whose model parameters are $\boldsymbol{\theta} = (\boldsymbol{\beta}'_1, \boldsymbol{\beta}'_2, \ldots, \boldsymbol{\beta}'_D)'$ and $\boldsymbol{\beta}_d = (\beta_{d1}, \beta_{d2}, \ldots, \beta_{dk})'$ $(d = 1, 2, \ldots, D)$, then $H_0\colon (\beta_{d2}, \ldots, \beta_{dk})' = \mathbf{0}$ $(d = 1, 2, \ldots, D)$. The number of restrictions r that are imposed on $\boldsymbol{\theta}$ is the total number of slope parameters in $\boldsymbol{\beta}'_1, \boldsymbol{\beta}'_2, \ldots, \boldsymbol{\beta}'_D$: $r = (k - 1) * D$.

PROC SURVEYLOGISTIC displays these tests in the "Testing Global Null Hypothesis: BETA=0" table.

Rao-Scott Likelihood Ratio Chi-Square Test

For complex survey design, you can use a design-adjusted Rao-Scott likelihood ratio chi-square test to test the global null hypothesis. For information about design-adjusted chi-square tests, see Lohr (2022, Section 10.3.2), Rao and Scott (1981), Rao and Scott (1984), Rao and Scott (1987), Thomas and Rao (1987), Rao and Thomas (1989), and Thomas, Singh, and Roberts (1996).

If you specify the CHISQ(NOADJUST) option, PROC SURVEYLOGISTIC computes the likelihood ratio chi-square test without the Rao-Scott design correction. If you specify the CHISQ(FIRSTORDER) option, PROC SURVEYLOGISTIC performs a first-order Rao-Scott likelihood ratio chi-square test. If you specify the CHISQ(SECONDORDER) option, PROC SURVEYLOGISTIC performs a second-order Rao-Scott likelihood ratio chi-square test.

If you do not specify the CHISQ option, the default test depends on the design and the model. By default, PROC SURVEYLOGISTIC performs a first-order or second-order Rao-Scott likelihood ratio chi-square (Satterthwaite) test if your design is not simple random sampling or when you provide replicate weights. Otherwise, if your design is simple random sampling and you do not provide replicate weights, PROC SURVEYLOGISTIC does not make any adjustment for the likelihood ratio test. In other words:

- If your design does not contain stratification nor clustering, and you do not provide replicate weights, then by default PROC SURVEYLOGISTIC performs a likelihood ratio chi-square test without any adjustment.
- If your design contains either stratification or clustering, or if you provide replicate weights, then by default PROC SURVEYLOGISTIC performs a likelihood ratio chi-square test with Rao-Scott adjustment. However, the default order of the adjustment depends on the number of model parameters excluding the intercepts.
  - If there is more than one nonintercept parameter in the model, the default is the second-order Rao-Scott likelihood ratio test.
  - If there is only one nonintercept parameter in the model, there is no need to compute the second-order adjustment. Therefore, the default is the first-order Rao-Scott likelihood ratio test.

Let $\hat{\boldsymbol{\theta}}$ be the estimated parameters, let $\hat{\boldsymbol{\theta}}_{H_0}$ be the estimated parameters under the global null hypothesis, and let r be the restrictions imposed on $\boldsymbol{\theta}$ under the global null hypothesis H_0. Let $L(\boldsymbol{\theta})$ be the log-likelihood function that is computed by using normalized weights.

Denote the estimated covariance matrix of $\hat{\boldsymbol{\theta}}$ under simple random sampling as $\hat{V}^{\text{srs}}(\hat{\boldsymbol{\theta}})$, and its partition corresponding to the r slope parameters as $\hat{V}^{\text{srs}}_{rr}(\hat{\boldsymbol{\theta}})$. Similarly, denote the estimated covariance matrix of $\hat{\boldsymbol{\theta}}$ under the sample design as $\hat{V}(\hat{\boldsymbol{\theta}})$, and its partition corresponding to the r slope parameters as $\hat{V}_{rr}(\hat{\boldsymbol{\theta}})$.

Define the *design effect matrix* E as

$$E = \hat{V}_{rr}(\hat{\boldsymbol{\theta}}) \left(\hat{V}^{\text{srs}}_{rr}(\hat{\boldsymbol{\theta}})\right)^{-1}$$

Denote r^* as the rank of E and the positive eigenvalues of the design matrix E as $\delta_1 \geq \delta_2 \geq \cdots \geq \delta_{r^*} > 0$.

Likelihood Ratio Chi-Square Test

Without the Rao-Scott design correction, the global null hypothesis is tested using either the chi-square statistics,

$$Q_{\chi^2} = 2\left[L(\hat{\boldsymbol{\theta}}) - L(\hat{\boldsymbol{\theta}}_{H_0})\right]$$

with r degrees of freedom, or an equivalent F statistics,

$$F = 2\left[L(\hat{\boldsymbol{\theta}}) - L(\hat{\boldsymbol{\theta}}_{H_0})\right]/r$$

with (r, ∞) degrees of freedom.

Rao-Scott First-Order Chi-Square Test

To address the impact of a complex survey design on the significance level of the likelihood ratio test, Rao and Scott (1984) proposed a first-order correction to the chi-square statistics as

$$Q_{RS1} = Q_{\chi^2}/\bar{\delta}.$$

where the first-order design correction,

$$\bar{\delta}. = \sum_{i=1}^{r^*} \delta_i / r^*$$

is the average of positive eigenvalues of the design effect matrix E.

Under the null hypothesis, the first-order Rao-Scott chi-square Q_{RS1} approximately follows a chi-square distribution with r^* degrees of freedom.

The corresponding F statistic is

$$F_{RS1} = Q_{RS1}/r^*$$

which has an F distribution with r^* and $df \cdot r^*$ degrees of freedom under the null hypothesis (Thomas and Rao 1984, 1987), and df is the design degrees of freedom as described in the section "Design Degrees of Freedom f for the Taylor Series Method" on page 10065.

Rao-Scott Second-Order Chi-Square Test

Rao and Scott (1987) further proposed the second-order (Satterthwaite) Rao-Scott chi-square statistic as

$$Q_{RS2} = Q_{RS1}/(1 + \hat{a}^2)$$

where Q_{RS1} is the first-order Rao-Scott chi-square statistic and the *second-order design correction* is computed from the coefficient of variation of the eigenvalues of the design effect matrix E as

$$\hat{a}^2 = \frac{1}{r^* - 1} \sum_{i=1}^{r^*} (\delta_i - \bar{\delta}.)^2 / \bar{\delta}.^2$$

Under the null hypothesis, the second-order Rao-Scott chi-square Q_{RS2} approximately follows a chi-square distribution with $r^*/(1 + \hat{a}^2)$ degrees of freedom.

The corresponding F statistic is

$$F_{RS2} = Q_{RS2}(1 + \hat{a}^2)/r^*$$

which has an F distribution with $r^*/(1 + \hat{a}^2)$ and $df \cdot r^*/(1 + \hat{a}^2)$ degrees of freedom under the null hypothesis (Thomas and Rao 1984, 1987), and df is the design degrees of freedom as described in the section "Design Degrees of Freedom f for the Taylor Series Method" on page 10065.

Wald Confidence Intervals for Parameters

Wald confidence intervals are sometimes called normal confidence intervals. They are based on the asymptotic normality of the parameter estimators. The $100(1 - \alpha)\%$ Wald confidence interval for θ_j is given by

$$\hat{\theta}_j \pm z_{1-\alpha/2} \hat{\sigma}_j$$

where $z_{1-\alpha/2}$ is the $100(1 - \alpha/2)$th percentile of the standard normal distribution, $\hat{\theta}_j$ is the pseudo-estimate of θ_j, and $\hat{\sigma}_j$ is the standard error estimate of $\hat{\theta}_j$ in the section "Variance Estimation" on page 10056.

Testing Linear Hypotheses about the Regression Coefficients

Linear hypotheses for $\boldsymbol{\theta}$ can be expressed in matrix form as

$$H_0\colon \mathbf{L}\boldsymbol{\theta} = \mathbf{c}$$

where $\mathbf{L}$ is a matrix of coefficients for the linear hypotheses and $\mathbf{c}$ is a vector of constants whose rank is r. The vector of regression coefficients $\boldsymbol{\theta}$ includes both slope parameters and intercept parameters.

Let $\hat{\boldsymbol{\theta}}$ be the MLE of $\boldsymbol{\theta}$, and let $\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}})$ be the estimated covariance matrix that is described in the section "Variance Estimation" on page 10056.

For the Taylor series variance estimation method, PROC SURVEYLOGISTIC computes the test statistic for the null hypothesis H_0 as

$$W_F = \left(\frac{f - r + 1}{f\ r}\right)(\mathbf{L}\hat{\boldsymbol{\theta}} - \mathbf{c})'[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}})\mathbf{L}']^{-1}(\mathbf{L}\hat{\boldsymbol{\theta}} - \mathbf{c})$$

where f is the design degrees of freedom as described in the section "Design Degrees of Freedom f for the Taylor Series Method" on page 10065.

For the replication variance estimation method, PROC SURVEYLOGISTIC computes the test statistic for the null hypothesis H_0 as

$$W_F = \frac{1}{r}(\mathbf{L}\hat{\boldsymbol{\theta}} - \mathbf{c})'[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}})\mathbf{L}']^{-1}(\mathbf{L}\hat{\boldsymbol{\theta}} - \mathbf{c})$$

Under the H_0, W_F has an F distribution with (r, df) degrees of freedom, and the denominator degrees of freedom df is described in the section "Degrees of Freedom" on page 10064.

As the denominator degrees of freedom grows, an F distribution approaches a chi-square distribution, and similarly a t distribution approaches a normal distribution. If you specify DF=INFINITY in the MODEL statement, PROC SURVEYLOGISTIC computes the test statistic for both Taylor series and replication methods for testing the null hypothesis H_0 as

$$W_{\chi^2} = (\mathbf{L}\hat{\boldsymbol{\theta}} - \mathbf{c})'[\mathbf{L}\hat{\mathbf{V}}(\hat{\boldsymbol{\theta}})\mathbf{L}']^{-1}(\mathbf{L}\hat{\boldsymbol{\theta}} - \mathbf{c})$$

Under H_0, χ^2_W has an asymptotic chi-square distribution with r degrees of freedom.

Type 3 Tests

For models that use less-than-full-rank parameterization (as specified by the PARAM=GLM option in the CLASS statement), a Type 3 test of an effect of interest (main effect or interaction) is a test of the Type III estimable functions that are defined for that effect. When the model contains no missing cells, performing the Type 3 test of a main effect corresponds to testing the hypothesis of equal marginal means. For more information about Type III estimable functions, see Chapter 53, "The GLM Procedure," and Chapter 16, "The Four Types of Estimable Functions." Also see Littell, Freund, and Spector (1991).

For models that use full-rank parameterization, all parameters are estimable when there are no missing cells, so it is unnecessary to define estimable functions. The standard test of an effect of interest in this case is the joint test that the values of the parameters associated with that effect are zero. For a model that uses effects parameterization (as specified by the PARAM=EFFECT option in the CLASS statement), performing the joint test for a main effect is equivalent to testing the equality of marginal means. For a model that uses reference parameterization (as specified by the PARAM=REF option in the CLASS statement), performing

the joint test is equivalent to testing the equality of cell means at the reference level of the other model effects. For more information about the coding scheme and the associated interpretation of results, see Muller and Fetterman (2002, Chapter 14).

If there is no interaction term, the Type 3 test of an effect for a model that uses GLM parameterization is the same as the joint test of the effect for the model that uses full-rank parameterization. In this situation, the joint test is also called the Type 3 test. For a model that contains an interaction term and no missing cells, the Type 3 test of a component main effect under GLM parameterization is the same as the joint test of the component main effect under effect parameterization. Both test the equality of cell means. But this Type 3 test differs from the joint test under reference parameterization, which tests the equality of cell means at the reference level of the other component main effect. If some cells are missing, you can obtain meaningful tests only by testing a Type III estimable function, so in this case you should use GLM parameterization.

The results of a Type 3 test or a joint test do not depend on the order in which you specify the terms in the MODEL statement.

Odds Ratio Estimation

Consider a dichotomous response variable with outcomes *event* and *nonevent*. Let a dichotomous risk factor variable X take the value 1 if the risk factor is present and 0 if the risk factor is absent. According to the logistic model, the log odds function, $g(X)$, is given by

$$g(X) \equiv \log\left(\frac{\Pr(\ event \mid X)}{\Pr(\ nonevent \mid X)}\right) = \beta_0 + \beta_1 X$$

The odds ratio ψ is defined as the ratio of the odds for those with the risk factor ($X = 1$) to the odds for those without the risk factor ($X = 0$). The log of the odds ratio is given by

$$\log(\psi) \equiv \log(\psi(X = 1, X = 0)) = g(X = 1) - g(X = 0) = \beta_1$$

The parameter, β_1, associated with X represents the change in the log odds from $X = 0$ to $X = 1$. So the odds ratio is obtained by simply exponentiating the value of the parameter associated with the risk factor. The odds ratio indicates how the odds of *event* change as you change X from 0 to 1. For instance, $\psi = 2$ means that the odds of an event when $X = 1$ are twice the odds of an event when $X = 0$.

Suppose the values of the dichotomous risk factor are coded as constants a and b instead of 0 and 1. The odds when $X = a$ become $\exp(\beta_0 + a\beta_1)$, and the odds when $X = b$ become $\exp(\beta_0 + b\beta_1)$. The odds ratio corresponding to an increase in X from a to b is

$$\psi = \exp[(b - a)\beta_1] = [\exp(\beta_1)]^{b-a} \equiv [\exp(\beta_1)]^c$$

Note that for any a and b such that $c = b - a = 1, \psi = \exp(\beta_1)$. So the odds ratio can be interpreted as the change in the odds for any increase of one unit in the corresponding risk factor. However, the change in odds for some amount other than one unit is often of greater interest. For example, a change of one pound in body weight might be too small to be considered important, while a change of 10 pounds might be more meaningful. The odds ratio for a change in X from a to b is estimated by raising the odds ratio estimate for a unit change in X to the power of $c = b - a$, as shown previously.

For a polytomous risk factor, the computation of odds ratios depends on how the risk factor is parameterized. For illustration, suppose that Race is a risk factor with four categories: White, Black, Hispanic, and Other.

For the effect parameterization scheme (PARAM=EFFECT) with White as the reference group, the design variables for Race are as follows.

	Design Variables		
Race	X_1	X_2	X_3
Black	1	0	0
Hispanic	0	1	0
Other	0	0	1
White	–1	–1	–1

The log odds for Black is

$$\begin{aligned} g(\text{Black}) &= \beta_0 + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) \\ &= \beta_0 + \beta_1 \end{aligned}$$

The log odds for White is

$$\begin{aligned} g(\text{White}) &= \beta_0 + \beta_1(X_1 = -1) + \beta_2(X_2 = -1) + \beta_3(X_3 = -1)) \\ &= \beta_0 - \beta_1 - \beta_2 - \beta_3 \end{aligned}$$

Therefore, the log odds ratio of Black versus White becomes

$$\begin{aligned} \log(\psi(\text{Black}, \text{White})) &= g(\text{Black}) - g(\text{White}) \\ &= 2\beta_1 + \beta_2 + \beta_3 \end{aligned}$$

For the reference cell parameterization scheme (PARAM=REF) with White as the reference cell, the design variables for race are as follows.

	Design Variables		
Race	X_1	X_2	X_3
Black	1	0	0
Hispanic	0	1	0
Other	0	0	1
White	0	0	0

The log odds ratio of Black versus White is given by

$$\begin{aligned} \log(\psi(\text{Black}, \text{White})) &= g(\text{Black}) - g(\text{White}) \\ &= (\beta_0 + \beta_1(X_1 = 1) + \beta_2(X_2 = 0)) + \beta_3(X_3 = 0)) - \\ &\quad (\beta_0 + \beta_1(X_1 = 0) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0)) \\ &= \beta_1 \end{aligned}$$

For the GLM parameterization scheme (PARAM=GLM), the design variables are as follows.

	Design Variables			
Race	X_1	X_2	X_3	X_4
Black	1	0	0	0
Hispanic	0	1	0	0
Other	0	0	1	0
White	0	0	0	1

The log odds ratio of Black versus White is

$$\begin{aligned}
\log(\psi(\text{Black}, \text{White})) &= g(\text{Black}) - g(\text{White}) \\
&= (\beta_0 + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) + \beta_4(X_4 = 0)) - \\
&\quad (\beta_0 + \beta_1(X_1 = 0) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) + \beta_4(X_4 = 1)) \\
&= \beta_1 - \beta_4
\end{aligned}$$

Consider the hypothetical example of heart disease among race in Hosmer and Lemeshow (2000, p. 51). The entries in the following contingency table represent counts.

	Race			
Disease Status	**White**	**Black**	**Hispanic**	**Other**
Present	5	20	15	10
Absent	20	10	10	10

The computation of odds ratio of Black versus White for various parameterization schemes is shown in Table 120.10.

Table 120.10 Odds Ratio of Heart Disease Comparing Black to White

	Parameter Estimates				
PARAM=	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{\beta}_4$	**Odds Ratio Estimates**
EFFECT	0.7651	0.4774	0.0719		$\exp(2 \times 0.7651 + 0.4774 + 0.0719) = 8$
REF	2.0794	1.7917	1.3863		$\exp(2.0794) = 8$
GLM	2.0794	1.7917	1.3863	0.0000	$\exp(2.0794) = 8$

Since the log odds ratio ($\log(\psi)$) is a linear function of the parameters, the Wald confidence interval for $\log(\psi)$ can be derived from the parameter estimates and the estimated covariance matrix. Confidence intervals for the odds ratios are obtained by exponentiating the corresponding confidence intervals for the log odd ratios. In the displayed output of PROC SURVEYLOGISTIC, the "Odds Ratio Estimates" table contains the odds ratio estimates and the corresponding *t* or Wald confidence intervals computed by using the covariance matrix in the section "Variance Estimation" on page 10056. For continuous explanatory variables, these odds ratios correspond to a unit increase in the risk factors.

To customize odds ratios for specific units of change for a continuous risk factor, you can use the UNITS statement to specify a list of relevant units for each explanatory variable in the model. Estimates of these customized odds ratios are given in a separate table. Let (L_j, U_j) be a confidence interval for $\log(\psi)$. The corresponding lower and upper confidence limits for the customized odds ratio $\exp(c\beta_j)$ are $\exp(cL_j)$

and $\exp(cU_j)$, respectively, (for $c > 0$); or $\exp(cU_j)$ and $\exp(cL_j)$, respectively, (for $c < 0$). You use the CLODDS option in the MODEL statement to request confidence intervals for the odds ratios.

For a generalized logit model, odds ratios are computed similarly, except D odds ratios are computed for each effect, corresponding to the D logits in the model.

Rank Correlation of Observed Responses and Predicted Probabilities

The predicted mean score of an observation is the sum of the ordered values (shown in the "Response Profile" table) minus one, weighted by the corresponding predicted probabilities for that observation; that is, the predicted means score is $\sum_{d=1}^{D+1}(d-1)\hat{\pi}_d$, where $D + 1$ is the number of response levels and $\hat{\pi}_d$ is the predicted probability of the dth (ordered) response.

A pair of observations with different observed responses is said to be *concordant* if the observation with the lower-ordered response value has a lower predicted mean score than the observation with the higher-ordered response value. If the observation with the lower-ordered response value has a higher predicted mean score than the observation with the higher-ordered response value, then the pair is *discordant*. If the pair is neither concordant nor discordant, it is a *tie*. Enumeration of the total numbers of concordant and discordant pairs is carried out by categorizing the predicted mean score into intervals of length $D/500$ and accumulating the corresponding frequencies of observations.

Let N be the sum of observation frequencies in the data. Suppose there are a total of t pairs with different responses, n_c of them are concordant, n_d of them are discordant, and $t - n_c - n_d$ of them are tied. PROC SURVEYLOGISTIC computes the following four indices of rank correlation for assessing the predictive ability of a model:

$$c = (n_c + 0.5(t - n_c - n_d))/t$$
$$\text{Somers' } D = (n_c - n_d)/t$$
$$\text{Goodman-Kruskal gamma} = (n_c - n_d)/(n_c + n_d)$$
$$\text{Kendall's tau-}a = (n_c - n_d)/(0.5N(N-1))$$

Note that c also gives an estimate of the area under the receiver operating characteristic (ROC) curve when the response is binary (Hanley and McNeil 1982).

For binary responses, the predicted mean score is equal to the predicted probability for Ordered Value 2.

Linear Predictor, Predicted Probability, and Confidence Limits

This section describes how predicted probabilities and confidence limits are calculated by using the pseudo-estimates (MLEs) obtained from PROC SURVEYLOGISTIC. For a specific example, see the section "Getting Started: SURVEYLOGISTIC Procedure" on page 9996. Predicted probabilities and confidence limits can be output to a data set with the OUTPUT statement.

Let $\Delta_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile point of a standard normal distribution or a t distribution according to the DF= specification:

$$\Delta_{\alpha/2} = \begin{cases} 100(1-\alpha/2)\text{th percentile point of a standard normal distribution } z_{\alpha/2} & \text{if DF=INFINITY} \\ 100(1-\alpha/2)\text{th percentile point of a } t \text{ distribution } t_{\alpha/2} & \text{otherwise} \end{cases}$$

Cumulative Response Models

For a row vector of explanatory variables $\mathbf{x}$, the linear predictor

$$\eta_i = g(\Pr(Y \le i \mid \mathbf{x})) = \alpha_i + \mathbf{x}\boldsymbol{\beta}, \quad 1 \le i \le k$$

is estimated by

$$\hat{\eta}_i = \hat{\alpha}_i + \mathbf{x}\hat{\boldsymbol{\beta}}$$

where $\hat{\alpha}_i$ and $\hat{\boldsymbol{\beta}}$ are the MLEs of α_i and $\boldsymbol{\beta}$. The estimated standard error of η_i is $\hat{\sigma}(\hat{\eta}_i)$, which can be computed as the square root of the quadratic form $(1, \mathbf{x}')\hat{\mathbf{V}}_{\mathbf{b}}(1, \mathbf{x}')'$, where $\hat{\mathbf{V}}_{\mathbf{b}}$ is the estimated covariance matrix of the parameter estimates. The asymptotic $100(1-\alpha)\%$ confidence interval for η_i is given by

$$\hat{\eta}_i \pm \Delta_{\alpha/2}\hat{\sigma}(\hat{\eta}_i)$$

The predicted value and the $100(1-\alpha)\%$ confidence limits for $\Pr(Y \le i \mid \mathbf{x})$ are obtained by back-transforming the corresponding measures for the linear predictor.

Link	**Predicted Probability**	$100(1-\alpha)$ **Confidence Limits**
LOGIT	$1/(1+e^{-\hat{\eta}_i})$	$1/(1+e^{-\hat{\eta}_i \pm \Delta_{\alpha/2}\hat{\sigma}(\hat{\eta}_i)})$
PROBIT	$\Phi(\hat{\eta}_i)$	$\Phi(\hat{\eta}_i \pm \Delta_{\alpha/2}\hat{\sigma}(\hat{\eta}_i))$
CLOGLOG	$1-e^{-e^{\hat{\eta}_i}}$	$1-e^{-e^{\hat{\eta}_i \pm \Delta_{\alpha/2}\hat{\sigma}(\hat{\eta}_i)}}$

Generalized Logit Model

For a vector of explanatory variables $\mathbf{x}$, let π_i denote the probability of obtaining the response value i:

$$\pi_i = \begin{cases} \pi_{k+1} e^{\alpha_i + \mathbf{x}\boldsymbol{\beta}_i} & 1 \le i \le k \\ \dfrac{1}{1 + \sum_{j=1}^{k} e^{\alpha_j + \mathbf{x}\boldsymbol{\beta}_j}} & i = k+1 \end{cases}$$

By the *delta method*,

$$\sigma^2(\pi_i) = \left(\frac{\partial \pi_i}{\partial \boldsymbol{\theta}}\right)' \mathbf{V}(\boldsymbol{\theta}) \frac{\partial \pi_i}{\partial \boldsymbol{\theta}}$$

A $100(1-\alpha)\%$ confidence level for π_i is given by

$$\hat{\pi}_i \pm \Delta_{\alpha/2} \hat{\sigma}(\hat{\pi}_i)$$

where $\hat{\pi}_i$ is the estimated expected probability of response i and $\hat{\sigma}(\hat{\pi}_i)$ is obtained by evaluating $\sigma(\pi_i)$ at $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$.

Output Data Sets

You can use the Output Delivery System (ODS) to create a SAS data set from any piece of PROC SURVEYLOGISTIC output. See the section "ODS Table Names" on page 10082 for more information. For a more detailed description of using ODS, see Chapter 23, "Using the Output Delivery System."

PROC SURVEYLOGISTIC also provides an OUTPUT statement to create a data set that contains estimated linear predictors, the estimates of the cumulative or individual response probabilities, and their confidence limits.

If you use replication variance estimation, PROC SURVEYLOGISTIC provides an output data set that stores the replicate weights and an output data set that stores the jackknife coefficients for jackknife variance estimation.

OUT= Data Set in the OUTPUT Statement

The OUT= data set in the OUTPUT statement contains all the variables in the input data set along with statistics you request by using *keyword=name* options or the PREDPROBS= option in the OUTPUT statement. In addition, if you use the single-trial syntax and you request any of the XBETA=, STDXBETA=, PREDICTED=, LCL=, and UCL= options, the OUT= data set contains the automatic variable _LEVEL_. The value of _LEVEL_ identifies the response category upon which the computed values of XBETA=, STDXBETA=, PREDICTED=, LCL=, and UCL= are based.

When there are more than two response levels, only variables named by the XBETA=, STDXBETA=, PREDICTED=, LOWER=, and UPPER= options and the variables given by PREDPROBS=(INDIVIDUAL CUMULATIVE) have their values computed; the other variables have missing values. If you fit a generalized logit model, the cumulative predicted probabilities are not computed.

When there are only two response categories, each input observation produces one observation in the OUT= data set.

If there are more than two response categories and you specify only the PREDPROBS= option, then each input observation produces one observation in the OUT= data set. However, if you fit an ordinal (cumulative) model and specify options other than the PREDPROBS= options, each input observation generates as many output observations as one fewer than the number of response levels, and the predicted probabilities and their confidence limits correspond to the cumulative predicted probabilities. If you fit a generalized logit model and specify options other than the PREDPROBS= options, each input observation generates as many output observations as the number of response categories; the predicted probabilities and their confidence limits correspond to the probabilities of individual response categories.

For observations in which only the response variable is missing, values of the XBETA=, STDXBETA=, PREDICTED=, UPPER=, LOWER=, and PREDPROBS= options are computed even though these observations do not affect the model fit. This enables, for instance, predicted probabilities to be computed for new observations.

Replicate Weights Output Data Set

If you specify the OUTWEIGHTS= *method-option* for VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or VARMETHOD=BOOTSTRAP, PROC SURVEYLOGISTIC stores the replicate weights in an output data set. The OUTWEIGHTS= output data set contains all observations from the DATA= input data set that are valid (used in the analysis). (A valid observation is an observation that has a positive value of the WEIGHT variable. Valid observations must also have nonmissing values of the STRATA and CLUSTER variables, unless you specify the MISSING option.)

The OUTWEIGHTS= data set contains all variables in the DATA= input data set and the replicate weight variables, RepWt_1, RepWt_2, . . ., RepWt_n, where n is the total number of replicates in the analysis. Each replicate weight variable contains the replicate weights for the corresponding replicate. Replicate weights equal 0 for observations that are not included in the replicate.

After the procedure creates replicate weights for a particular input data set and survey design, you can use the OUTWEIGHTS= *method-option* to store these replicate weights and then use them again in subsequent analyses, either in PROC SURVEYLOGISTIC or in other survey procedures. You can use the REPWEIGHTS statement to provide replicate weights for the procedure.

Jackknife Coefficients Output Data Set

If you specify the OUTJKCOEFS= *method-option* for VARMETHOD=JACKKNIFE, PROC SURVEYLOGISTIC stores the jackknife coefficients in an output data set. The OUTJKCOEFS= output data set contains one observation for each replicate. The OUTJKCOEFS= data set contains the following variables:

- Replicate, which is the replicate number for the jackknife coefficient
- JKCoefficient, which is the jackknife coefficient
- DonorStratum, which is the stratum of the PSU that was deleted to construct the replicate, if you specify a STRATA statement

After the procedure creates jackknife coefficients for a particular input data set and survey design, you can use the OUTJKCOEFS= *method-option* to store these coefficients and then use them again in subsequent analyses, either in PROC SURVEYLOGISTIC or in the other survey procedures. You can use the JKCOEFS= option in the REPWEIGHTS statement to provide jackknife coefficients for the procedure.

Displayed Output

The SURVEYLOGISTIC procedure produces output that is described in the following sections.

Output that is generated by the EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements are not listed below. For information about the output that is generated by these statements, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Model Information

By default, PROC SURVEYLOGISTIC displays the following information in the "Model Information" table:

- name of the input Data Set
- name and label of the Response Variable if the single-trial syntax is used
- number of Response Levels
- name of the Events Variable if the events/trials syntax is used
- name of the Trials Variable if the events/trials syntax is used
- name of the Offset Variable if the OFFSET= option is specified
- name of the Frequency Variable if the FREQ statement is specified
- name(s) of the Stratum Variable(s) if the STRATA statement is specified
- total Number of Strata if the STRATA statement is specified
- name(s) of the Cluster Variable(s) if the CLUSTER statement is specified
- total Number of Clusters if the CLUSTER statement is specified
- name of the Weight Variable if the WEIGHT statement is specified
- Variance Adjustment method
- Upper Bound ADJBOUND parameter used in the VADJUST=MOREL(ADJBOUND=) option
- Lower Bound DEFFBOUND parameter used in the VADJUST=MOREL(DEFFBOUND=) option
- whether *fpc* (finite population correction) is used

Variance Estimation

By default, PROC SURVEYLOGISTIC displays the following variance estimation information in the "Variance Estimation" table:

- Method, which is the variance estimation method
- Variance Adjustment method

- Initial seed for the random number that the bootstrap method uses
- Upper Bound ADJBOUND parameter specified in the VADJUST=MOREL(ADJBOUND=) option
- Lower Bound DEFFBOUND parameter specified in the VADJUST=MOREL(DEFFBOUND=) option
- whether *fpc* (finite population correction) is used
- Number of Replicates
- Number of Replicates Used, which is the number of replicates that are used in variance estimation (the rest are excluded because of nonconvergence)
- Hadamard Data Set name, if you specify the VARMETHOD=BRR(HADAMARD=) *method-option*
- Fay Coefficient, if you specify the VARMETHOD=BRR(FAY) *method-option*
- Replicate Weights input data set name, if you use a REPWEIGHTS statement
- whether Missing Levels are created for categorical variables by the MISSING option
- whether observations with Missing Values are included in the analysis by the NOMCAR option

Data Summary

By default, PROC SURVEYLOGISTIC displays the following information for the entire data set:

- Number of Observations read from the input data set
- Number of Observations used in the analysis

If there is a DOMAIN statement, PROC SURVEYLOGISTIC also displays the following:

- Number of Observations in the current domain
- Number of Observations not in the current domain

If there is a FREQ statement, PROC SURVEYLOGISTIC also displays the following:

- Sum of Frequencies of all the observations read from the input data set
- Sum of Frequencies of all the observations used in the analysis

If there is a WEIGHT statement, PROC SURVEYLOGISTIC also displays the following:

- Sum of Weights of all the observations read from the input data set
- Sum of Weights of all the observations used in the analysis
- Sum of Weights of all the observations in the current domain, if DOMAIN statement is also specified.

Response Profile

By default, PROC SURVEYLOGISTIC displays a “Response Profile” table, which gives, for each response level, the ordered value (an integer between one and the number of response levels, inclusive); the value of the response variable if the single-trial syntax is used or the values “EVENT” and “NO EVENT” if the events/trials syntax is used; the count or frequency; and the sum of weights if the WEIGHT statement is specified.

Class Level Information

If you use a CLASS statement to name classification variables, PROC SURVEYLOGISTIC displays a "Class Level Information" table. This table contains the following information for each classification variable:

- Class, which lists each CLASS variable name
- Value, which lists the values of the classification variable. The values are separated by a white space character; therefore, to avoid confusion, you should not include a white space character within a classification variable value.
- Design Variables, which lists the parameterization used for the classification variables

Stratum Information

When you specify the LIST option in the STRATA statement, PROC SURVEYLOGISTIC displays a "Stratum Information" table, which provides the following information for each stratum:

- Stratum Index, which is a sequential stratum identification number
- STRATA variable(s), which lists the levels of STRATA variables for the stratum
- Population Total, if you specify the TOTAL= option
- Sampling Rate, if you specify the TOTAL= or RATE= option. If you specify the TOTAL= option, the sampling rate is based on the number of nonmissing observations in the stratum.
- N Obs, which is the number of observations
- number of Clusters, if you specify a CLUSTER statement

Maximum Likelihood Iteration History

The “Maximum Likelihood Iterative Phase” table gives the iteration number, the step size (in the scale of 1.0, 0.5, 0.25, and so on) or the ridge value, –2 log likelihood, and parameter estimates for each iteration. Also displayed are the last evaluation of the gradient vector and the last change in the –2 log likelihood. You need to use the ITPRINT option in the MODEL statement to obtain this table.

Model Fit Statistics

By default, PROC SURVEYLOGISTIC displays the following information in the "Model Fit Statistics" table:

- "Model Fit Statistics" and "Testing Global Null Hypothesis: BETA=0" tables, which give the various criteria (–2 Log L, AIC, SC) based on the likelihood for fitting a model with intercepts only and for fitting a model with intercepts and explanatory variables. If you specify the NOINT option, these statistics are calculated without considering the intercept parameters. The third column of the table gives the chi-square statistics and p-values for the –2 Log L statistic and for the Score statistic. These test the joint effect of the explanatory variables included in the model. The Score criterion is always missing for the models identified by the first two columns of the table. Note also that the first two rows of the Chi-Square column are always missing, since tests cannot be performed for AIC and SC.
- generalized R^2 measures for the fitted model if you specify the RSQUARE option in the MODEL statement

Type III Analysis of Effects

PROC SURVEYLOGISTIC displays the "Type III Analysis of Effects" table if the model contains an effect that involves a CLASS variable. This table gives the degrees of freedom for F statistics or Wald chi-square statistic, and the p-value for each effect in the model.

Analysis of Maximum Likelihood Estimates

By default, PROC SURVEYLOGISTIC displays the following information in the "Analysis of Maximum Likelihood Estimates" table:

- maximum likelihood estimate of the parameter
- estimated standard error of the parameter estimate, computed as the square root of the corresponding diagonal element of the estimated covariance matrix
- t value, which is the t statistic for testing H_0: Parameter $= 0$
- Pr $> \mid t \mid$, which is the two-sided p-value for the t test
- $100(1-\alpha)\%$ confidence intervals for estimated parameters. You need to specify the CLPARM option in the MODEL statement to display these estimates.
- standardized estimate for the slope parameter, given by $\hat{\beta}_i/(s/s_i)$, where s_i is the total sample standard deviation for the ith explanatory variable and

$$s = \begin{cases} \pi/\sqrt{3} & \text{logistic} \\ 1 & \text{normal} \\ \pi/\sqrt{6} & \text{extreme-value} \end{cases}$$

  You need to specify the STB option in the MODEL statement to obtain these estimates. Standardized estimates of the intercept parameters are set to missing.

- value of ($e^{\hat{\beta}_i}$) for each slope parameter β_i if you specify the EXPB option in the MODEL statement. For continuous variables, this is equivalent to the estimated odds ratio for a one-unit change.
- label of the variable (if space permits) if you specify the PARMLABEL option in the MODEL statement. Because of constraints on the line size, the variable label might be suppressed in order to display the table in one panel. Use the SAS system option LINESIZE= to specify a larger line size to accommodate variable labels. A shorter line size can break the table into two panels, allowing labels to be displayed.

Odds Ratio Estimates

The "Odds Ratio Estimates" table displays the odds ratio estimates and the corresponding 95% confidence intervals. For continuous explanatory variables, these odds ratios correspond to a unit increase in the risk factors.

Association of Predicted Probabilities and Observed Responses

The "Association of Predicted Probabilities and Observed Responses" table displays measures of association between predicted probabilities and observed responses, which include a breakdown of the number of pairs with different responses, and four rank correlation indexes: Somers' D, Goodman-Kruskal gamma, and Kendall's tau-a, and c.

Estimated Covariance Matrix

PROC SURVEYLOGISTIC displays the following information in the "Estimated Covariance Matrix" table:

- estimated covariance matrix of the parameter estimates if you use the COVB option in the MODEL statement
- estimated correlation matrix of the parameter estimates if you use the CORRB option in the MODEL statement

Linear Hypotheses Testing Results

The "Linear Hypothesis Testing" table gives the result of the F test or Wald test for each TEST statement (if specified).

Hadamard Matrix

If you specify the VARMETHOD=BRR(PRINTH) *method-option* in the PROC SURVEYLOGISTIC statement, the procedure displays the Hadamard matrix.

When you provide a Hadamard matrix with the VARMETHOD=BRR(HADAMARD=) *method-option*, the procedure displays only used rows and columns of the Hadamard matrix.

ODS Table Names

PROC SURVEYLOGISTIC assigns a name to each table it creates; these names are listed in Table 120.11. You can use these names to refer the table when using the Output Delivery System (ODS) to select tables and create output data sets. The EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, and SLICE statements also create tables, which are not listed in Table 120.11. For information about these tables, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 120.11 ODS Tables Produced by PROC SURVEYLOGISTIC

ODS Table Name	Description	Statement	Option
Association	Association of predicted probabilities and observed responses	MODEL	Default
ClassLevelInfo	CLASS variable levels and design variables	MODEL	Default (with CLASS variables)
CLOdds	Confidence intervals for odds ratios	MODEL	CLODDS
CLparm	Confidence intervals for parameters	MODEL	CLPARM
ContrastCoeff	**L** matrix from CONTRAST	CONTRAST	E
ContrastEstimate	Estimates from CONTRAST	CONTRAST	ESTIMATE=
ContrastTest	Wald test for CONTRAST	CONTRAST	Default
ConvergenceStatus	Convergence status	MODEL	Default
CorrB	Estimated correlation matrix of parameter estimators	MODEL	CORRB
CovB	Estimated covariance matrix of parameter estimators (Ordinal response)	MODEL	COVB
DomainSummary	Domain summary	DOMAIN	Default
FitStatistics	Model fit statistics	MODEL	Default
GlobalTests	Test for global null hypothesis	MODEL	Default
Gradient	Gradient evaluated at global null hypothesis	MODEL	GRADIENT
HadamardMatrix	Hadamard matrix	PROC	PRINTH
IterHistory	Iteration history	MODEL	ITPRINT
LastGradient	Last evaluation of gradient	MODEL	ITPRINT
Linear	Linear combination	PROC	Default
LogLikeChange	Final change in the log likelihood	MODEL	ITPRINT
ModelANOVA	Joint or Type 3 tests of effects	MODEL	Default (with CLASS variables)
ModelInfo	Model information	PROC	Default
NObs	Number of observations	PROC	Default
OddsEst	Adjusted odds ratios	UNITS	Default
OddsRatios	Odds ratios	MODEL	Default

Table 120.11 *continued*

ODS Table Name	Description	Statement	Option
ParameterEstimates	Maximum likelihood estimates of model parameters	MODEL	Default
RSquare	R-square	MODEL	RSQUARE
ResponseProfile	Response profile	PROC	Default
StrataInfo	Stratum information	STRATA	LIST
TestPrint1	$\mathbf{L}[\mathrm{cov}(\mathbf{b})]\mathbf{L}'$ and $\mathbf{Lb} - \mathbf{c}$	TEST	PRINT
TestPrint2	$\mathrm{Ginv}(\mathbf{L}[\mathrm{cov}(\mathbf{b})]\mathbf{L}')$ and $\mathrm{Ginv}(\mathbf{L}[\mathrm{cov}(\mathbf{b})]\mathbf{L}')(\mathbf{Lb} - \mathbf{c})$	TEST	PRINT
TestStmts	Linear hypotheses testing results	TEST	Default
VarianceEstimation	Variance estimation	PROC	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, then the ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements can produce plots that are associated with their analyses. For information about these plots, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Examples: SURVEYLOGISTIC Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 120.1: Stratified Sampling

(View the complete code for this example (svlex1.sas) in the example repository.)

A market research firm conducts a survey among undergraduate students at a certain university to evaluate three new Web designs for a commercial Web site targeting undergraduate students at the university.

The sample design is a stratified sample where the strata are students' classes. Within each class, 300 students are randomly selected by using simple random sampling without replacement. The total number of students in each class in the fall semester of 2001 is shown in the following table:

Class	Enrollment
1 - Freshman	3,734
2 - Sophomore	3,565
3 - Junior	3,903
4 - Senior	4,196

This total enrollment information is saved in the SAS data set Enrollment by using the following SAS statements:

```
proc format;
   value Class
      1='Freshman' 2='Sophomore'
      3='Junior'   4='Senior';
run;

data Enrollment;
   format Class Class.;
   input Class _TOTAL_;
   datalines;
1 3734
2 3565
3 3903
4 4196
;
```

In the data set Enrollment, the variable _TOTAL_ contains the enrollment figures for all classes. They are also the population size for each stratum in this example.

Each student selected in the sample evaluates one randomly selected Web design by using the following scale:

1	Dislike very much
2	Dislike
3	Neutral
4	Like
5	Like very much

The survey results are collected and shown in the following table, with the three different Web designs coded as A, B, and C.

Evaluation of New Web Designs

		Rating Counts				
Strata	**Design**	**1**	**2**	**3**	**4**	**5**
Freshman	A	10	34	35	16	15
	B	5	6	24	30	25
	C	11	14	20	34	21
Sophomore	A	19	12	26	18	25
	B	10	18	32	23	26
	C	15	22	34	9	20
Junior	A	8	21	23	26	22
	B	1	4	15	33	47
	C	16	19	30	23	12
Senior	A	11	14	24	33	18
	B	8	15	25	30	22
	C	2	34	30	18	16

The survey results are stored in a SAS data set WebSurvey by using the following SAS statements:

```
proc format;
   value Design 1='A' 2='B' 3='C';
   value Rating
      1='dislike very much'
      2='dislike'
      3='neutral'
      4='like'
      5='like very much';
run;

data WebSurvey;
   format Class Class. Design Design. Rating Rating.;
   do Class=1 to 4;
      do Design=1 to 3;
         do Rating=1 to 5;
            input Count @@;
            output;
         end;
      end;
   end;
   datalines;
```

```
10 34 35 16 15   8 21 23 26 22   5 10 24 30 21
 1 14 25 23 37  11 14 20 34 21  16 19 30 23 12
19 12 26 18 25  11 14 24 33 18  10 18 32 23 17
 8 15 35 30 12  15 22 34  9 20   2 34 30 18 16
;

data WebSurvey;
   set WebSurvey;
   if Class=1 then Weight=3734/300;
   if Class=2 then Weight=3565/300;
   if Class=3 then Weight=3903/300;
   if Class=4 then Weight=4196/300;
run;
```

The data set WebSurvey contains the variables Class, Design, Rating, Count, and Weight. The variable class is the stratum variable, with four strata: freshman, sophomore, junior, and senior. The variable Design specifies the three new Web designs: A, B, and C. The variable Rating contains students' evaluations of the new Web designs. The variable counts gives the frequency with which each Web design received each rating within each stratum. The variable weight contains the sampling weights, which are the reciprocals of selection probabilities in this example.

Output 120.1.1 shows the first 20 observations of the data set.

Output 120.1.1 Web Design Survey Sample (First 20 Observations)

OBS	Class	Design	Rating	Count	Weight
1	Freshman	A	dislike very much	10	12.4467
2	Freshman	A	dislike	34	12.4467
3	Freshman	A	neutral	35	12.4467
4	Freshman	A	like	16	12.4467
5	Freshman	A	like very much	15	12.4467
6	Freshman	B	dislike very much	8	12.4467
7	Freshman	B	dislike	21	12.4467
8	Freshman	B	neutral	23	12.4467
9	Freshman	B	like	26	12.4467
10	Freshman	B	like very much	22	12.4467
11	Freshman	C	dislike very much	5	12.4467
12	Freshman	C	dislike	10	12.4467
13	Freshman	C	neutral	24	12.4467
14	Freshman	C	like	30	12.4467
15	Freshman	C	like very much	21	12.4467
16	Sophomore	A	dislike very much	1	11.8833
17	Sophomore	A	dislike	14	11.8833
18	Sophomore	A	neutral	25	11.8833
19	Sophomore	A	like	23	11.8833
20	Sophomore	A	like very much	37	11.8833

The following SAS statements perform the logistic regression:

```
proc surveylogistic data=WebSurvey total=Enrollment;
   stratum Class;
   freq Count;
   class Design;
   model Rating (order=internal) = design;
   weight Weight;
run;
```

The PROC SURVEYLOGISTIC statement invokes the procedure. The TOTAL= option specifies the data set Enrollment, which contains the population totals in the strata. The population totals are used to calculate the finite population correction factor in the variance estimates. The response variable Rating is in the ordinal scale. A cumulative logit model is used to investigate the responses to the Web designs. In the MODEL statement, rating is the response variable, and Design is the effect in the regression model. The ORDER=INTERNAL option is used for the response variable Rating to sort the ordinal response levels of Rating by its internal (numerical) values rather than by the formatted values (for example, 'like very much'). Because the sample design involves stratified simple random sampling, the STRATA statement is used to specify the stratification variable Class. The WEIGHT statement specifies the variable Weight for sampling weights.

The sample and analysis summary is shown in Output 120.1.2. There are five response levels for the Rating, with 'dislike very much' as the lowest ordered value. The regression model is modeling lower cumulative probabilities by using logit as the link function. Because the TOTAL= option is used, the finite population correction is included in the variance estimation. The sampling weight is also used in the analysis.

Output 120.1.2 Web Design Survey, Model Information

The SURVEYLOGISTIC Procedure

Model Information	
Data Set	WORK.WEBSURVEY
Response Variable	Rating
Number of Response Levels	5
Frequency Variable	Count
Stratum Variable	Class
Number of Strata	4
Weight Variable	Weight
Model	Cumulative Logit
Optimization Technique	Fisher's Scoring
Variance Adjustment	Degrees of Freedom (DF)
Finite Population Correction	Used

Response Profile

Ordered Value	Rating	Total Frequency	Total Weight
1	dislike very much	116	1489.0733
2	dislike	227	2933.0433
3	neutral	338	4363.3767
4	like	283	3606.8067
5	like very much	236	3005.7000

Probabilities modeled are cumulated over the lower Ordered Values.

An alternative model is to use the generalized logit model with the LINK=GLOGIT option, as shown in the following SAS statements:

```
proc surveylogistic data=WebSurvey total=Enrollment;
   stratum Class;
   freq Count;
   class Design;
   model Rating (ref='neutral') = Design /link=glogit;
   weight Weight;
run;
```

The REF='neutral' option is used for the response variable Rating to indicate that all other response levels are referenced to the level 'neutral.' The option LINK=GLOGIT option requests that the procedure fit a generalized logit model.

The summary of the analysis is shown in Output 120.1.3, which indicates that the generalized logit model is used in the analysis.

Output 120.1.3 Web Design Survey, Model Information

The SURVEYLOGISTIC Procedure

Model Information	
Data Set	WORK.WEBSURVEY
Response Variable	Rating
Number of Response Levels	5
Frequency Variable	Count
Stratum Variable	Class
Number of Strata	4
Weight Variable	Weight
Model	Generalized Logit
Optimization Technique	Newton-Raphson
Variance Adjustment	Degrees of Freedom (DF)
Finite Population Correction	Used

Response Profile

Ordered Value	Rating	Total Frequency	Total Weight
1	dislike	227	2933.0433
2	dislike very much	116	1489.0733
3	like	283	3606.8067
4	like very much	236	3005.7000
5	neutral	338	4363.3767

Logits modeled use Rating='neutral' as the reference category.

Output 120.1.4 shows the parameterization for the main effect Design.

Output 120.1.4 Web Design Survey, Class Level Information

Class Level Information			
Class	Value	Design Variables	
Design	A	1	0
	B	0	1
	C	-1	-1

The parameter and odds ratio estimates are shown in Output 120.1.5. For each odds ratio estimate, the 95% confidence intervals shown in the table contain the value 1.0. Therefore, no conclusion about which Web design is preferred can be made based on this survey.

Output 120.1.5 Web Design Survey, Parameter and Odds Ratio Estimates

Analysis of Maximum Likelihood Estimates					
Parameter	Rating	Estimate	Standard Error	t Value	Pr > |t|
Intercept	dislike	-0.3964	0.0832	-4.77	<.0001
Intercept	dislike very much	-1.0826	0.1045	-10.36	<.0001
Intercept	like	-0.1892	0.0780	-2.43	0.0154
Intercept	like very much	-0.3767	0.0824	-4.57	<.0001
Design A	dislike	-0.0942	0.1166	-0.81	0.4196
Design A	dislike very much	-0.0647	0.1469	-0.44	0.6597
Design A	like	-0.1370	0.1104	-1.24	0.2149
Design A	like very much	0.0446	0.1130	0.39	0.6934
Design B	dislike	0.0391	0.1201	0.33	0.7451
Design B	dislike very much	0.2721	0.1448	1.88	0.0605
Design B	like	0.1669	0.1102	1.52	0.1300
Design B	like very much	0.1420	0.1174	1.21	0.2265
NOTE: The degrees of freedom for the t tests is 1196.					

Odds Ratio Estimates				
Effect	Rating	Point Estimate	95% Confidence Limits	
Design A vs C	dislike	0.861	0.583	1.272
Design A vs C	dislike very much	1.153	0.691	1.924
Design A vs C	like	0.899	0.618	1.306
Design A vs C	like very much	1.260	0.851	1.866
Design B vs C	dislike	0.984	0.658	1.471
Design B vs C	dislike very much	1.615	0.975	2.677
Design B vs C	like	1.218	0.838	1.769
Design B vs C	like very much	1.389	0.924	2.087
NOTE: The degrees of freedom in computing the confidence limits is 1196.				

Example 120.2: The Medical Expenditure Panel Survey (MEPS)

(View the complete code for this example (svlex2.sas) in the example repository.)

The U.S. Department of Health and Human Services conducts the Medical Expenditure Panel Survey (MEPS) to produce national and regional estimates of various aspects of health care. The MEPS has a complex sample design that includes both stratification and clustering. The sampling weights are adjusted for nonresponse and raked with respect to population control totals from the Current Population Survey. See the MEPS (2006) and Machlin, Yu, and Zodet (2005) for details.

In this example, the 1999 full-year consolidated data file HC-038 (MEPS HC-038, 2002) from the MEPS is used to investigate the relationship between medical insurance coverage and the demographic variables. The data can be downloaded directly from the Agency for Healthcare Research and Quality (AHRQ) Web site at https://meps.ahrq.gov/mepsweb/data_stats/download_data_files_detail.jsp?cboPufNumber=HC-038 in either ASCII format or SAS transport format. The Web site includes a detailed description of the data as well as the SAS program used to access and format it.

For this example, the SAS transport format data file for HC-038 is downloaded to 'C:H38.ssp' on a Windows-based PC. The instructions on the Web site lead to the following SAS statements for creating a SAS data set MEPS, which contains only the sample design variables and other variables necessary for this analysis.

```
proc format;
   value racex
      -9 = 'NOT ASCERTAINED'
      -8 = 'DK'
      -7 = 'REFUSED'
      -1 = 'INAPPLICABLE'
      1 = 'AMERICAN INDIAN'
      2 = 'ALEUT, ESKIMO'
      3 = 'ASIAN OR PACIFIC ISLANDER'
      4 = 'BLACK'
      5 = 'WHITE'
      91 = 'OTHER'
      ;
   value sex
      -9 = 'NOT ASCERTAINED'
      -8 = 'DK'
      -7 = 'REFUSED'
      -1 = 'INAPPLICABLE'
      1 = 'MALE'
      2 = 'FEMALE'
      ;
   value povcat9h
      1 = 'NEGATIVE OR POOR'
      2 = 'NEAR POOR'
      3 = 'LOW INCOME'
      4 = 'MIDDLE INCOME'
      5 = 'HIGH INCOME'
      ;
   value inscov9f
      1 = 'ANY PRIVATE'
      2 = 'PUBLIC ONLY'
```

```
      3 = 'UNINSURED'
      ;
run;

libname mylib '';
filename in1 'H38.SSP';
proc xcopy in=in1 out=mylib import;
run;

data meps;
   set mylib.H38;
   label racex= sex= inscov99= povcat99=
      varstr99= varpsu99= perwt99f= totexp99=;
   format racex racex. sex sex.
      povcat99 povcat9h. inscov99 inscov9f.;
   keep inscov99 sex racex povcat99 varstr99
      varpsu99 perwt99f totexp99;
run;
```

There are a total of 24,618 observations in this SAS data set. Each observation corresponds to a person in the survey. The stratification variable is VARSTR99, which identifies the 143 strata in the sample. The variable VARPSU99 identifies the 460 PSUs in the sample. The sampling weights are stored in the variable PERWT99F. The response variable is the health insurance coverage indicator variable, INSCOV99, which has three values:

1	The person had any private insurance coverage any time during 1999
2	The person had only public insurance coverage during 1999
3	The person was uninsured during all of 1999

The demographic variables include gender (SEX), race (RACEX), and family income level as a percent of the poverty line (POVCAT99). The variable RACEX has five categories:

1	American Indian
2	Aleut, Eskimo
3	Asian or Pacific Islander
4	Black
5	White

The variable POVCAT99 is constructed by dividing family income by the applicable poverty line (based on family size and composition), with the resulting percentages grouped into five categories:

1	Negative or poor (less than 100%)
2	Near poor (100% to less than 125%)
3	Low income (125% to less than 200%)
4	Middle income (200% to less than 400%)
5	High income (greater than or equal to 400%)

The data set also contains the total health care expenditure in 1999, TOTEXP99, which is used as a covariate in the analysis.

Output 120.2.1 displays the first 30 observations of this data set.

Output 120.2.1 1999 Full-Year MEPS (First 30 Observations)

OBS	SEX	RACEX	POVCAT99	INSCOV99	TOTEXP99	PERWT99F	VARSTR99	VARPSU99
1	MALE	WHITE	MIDDLE INCOME	PUBLIC ONLY	2735	14137.86	131	2
2	FEMALE	WHITE	MIDDLE INCOME	ANY PRIVATE	6687	17050.99	131	2
3	MALE	WHITE	MIDDLE INCOME	ANY PRIVATE	60	35737.55	131	2
4	MALE	WHITE	MIDDLE INCOME	ANY PRIVATE	60	35862.67	131	2
5	FEMALE	WHITE	MIDDLE INCOME	ANY PRIVATE	786	19407.11	131	2
6	MALE	WHITE	MIDDLE INCOME	ANY PRIVATE	345	18499.83	131	2
7	MALE	WHITE	MIDDLE INCOME	ANY PRIVATE	680	18499.83	131	2
8	MALE	WHITE	MIDDLE INCOME	ANY PRIVATE	3226	22394.53	136	1
9	FEMALE	WHITE	MIDDLE INCOME	ANY PRIVATE	2852	27008.96	136	1
10	MALE	WHITE	MIDDLE INCOME	ANY PRIVATE	112	25108.71	136	1
11	MALE	WHITE	MIDDLE INCOME	ANY PRIVATE	3179	17569.81	136	1
12	MALE	WHITE	MIDDLE INCOME	ANY PRIVATE	168	21478.06	136	1
13	FEMALE	WHITE	MIDDLE INCOME	ANY PRIVATE	1066	21415.68	136	1
14	MALE	WHITE	NEGATIVE OR POOR	PUBLIC ONLY	0	12254.66	125	1
15	MALE	WHITE	NEGATIVE OR POOR	ANY PRIVATE	0	17699.75	125	1
16	FEMALE	WHITE	NEGATIVE OR POOR	UNINSURED	0	18083.15	125	1
17	MALE	BLACK	NEGATIVE OR POOR	PUBLIC ONLY	230	6537.97	78	10
18	MALE	WHITE	LOW INCOME	UNINSURED	408	8951.36	95	2
19	FEMALE	WHITE	LOW INCOME	UNINSURED	0	11833.00	95	2
20	MALE	WHITE	LOW INCOME	UNINSURED	40	12754.07	95	2
21	FEMALE	WHITE	LOW INCOME	UNINSURED	51	14698.57	95	2
22	MALE	WHITE	LOW INCOME	UNINSURED	0	3890.20	92	19
23	FEMALE	WHITE	LOW INCOME	UNINSURED	610	5882.29	92	19
24	MALE	WHITE	LOW INCOME	PUBLIC ONLY	24	8610.47	92	19
25	FEMALE	BLACK	MIDDLE INCOME	UNINSURED	1758	0.00	64	1
26	MALE	BLACK	MIDDLE INCOME	PUBLIC ONLY	551	7049.70	64	1
27	MALE	BLACK	MIDDLE INCOME	ANY PRIVATE	65	34067.03	64	1
28	FEMALE	BLACK	NEGATIVE OR POOR	PUBLIC ONLY	0	9313.84	73	12
29	FEMALE	BLACK	NEGATIVE OR POOR	PUBLIC ONLY	10	14697.03	73	12
30	MALE	BLACK	NEGATIVE OR POOR	PUBLIC ONLY	0	4574.73	73	12

The following SAS statements fit a generalized logit model for the 1999 full-year consolidated MEPS data:

```
proc surveylogistic data=meps;
   stratum VARSTR99;
   cluster VARPSU99;
   weight PERWT99F;
   class SEX RACEX POVCAT99;
   model INSCOV99 = TOTEXP99 SEX RACEX POVCAT99 / link=glogit;
run;
```

The STRATUM statement specifies the stratification variable VARSTR99. The CLUSTER statement specifies the PSU variable VARPSU99. The WEIGHT statement specifies the sample weight variable PERWT99F. The demographic variables SEX, RACEX, and POVCAT99 are listed in the CLASS statement to indicate that they are categorical independent variables in the MODEL statement. In the MODEL statement, the response variable is INSCOV99, and the independent variables are TOTEXP99 along with the selected demographic variables. The LINK= option requests that the procedure fit the generalized logit model because the response variable INSCOV99 has nominal responses.

The results of this analysis are shown in the following outputs.

PROC SURVEYLOGISTIC lists the model fitting information and sample design information in Output 120.2.2.

Output 120.2.2 MEPS, Model Information

The SURVEYLOGISTIC Procedure

Model Information	
Data Set	WORK.MEPS
Response Variable	INSCOV99
Number of Response Levels	3
Stratum Variable	VARSTR99
Number of Strata	143
Cluster Variable	VARPSU99
Number of Clusters	460
Weight Variable	PERWT99F
Model	Generalized Logit
Optimization Technique	Newton-Raphson
Variance Adjustment	Degrees of Freedom (DF)

Output 120.2.3 displays the number of observations and the total of sampling weights both in the data set and used in the analysis. Only the observations with positive person-level weight are used in the analysis. Therefore, 1,053 observations with zero person-level weights were deleted.

Output 120.2.3 MEPS, Number of Observations

Number of Observations Read	24618
Number of Observations Used	23565
Sum of Weights Read	2.7641E8
Sum of Weights Used	2.7641E8

Output 120.2.4 lists the three insurance coverage levels for the response variable INSCOV99. The "UNINSURED" category is used as the reference category in the model.

Output 120.2.4 MEPS, Response Profile

Response Profile

Ordered Value	INSCOV99	Total Frequency	Total Weight
1	ANY PRIVATE	16130	204403997
2	PUBLIC ONLY	4241	41809572
3	UNINSURED	3194	30197198

Logits modeled use INSCOV99='UNINSURED' as the reference category.

Output 120.2.5 shows the parameterization in the regression model for each categorical independent variable.

Output 120.2.5 MEPS, Classification Levels

Class Level Information					
Class	Value	Design Variables			
SEX	FEMALE	1			
	MALE	-1			
RACEX	ALEUT, ESKIMO	1	0	0	0
	AMERICAN INDIAN	0	1	0	0
	ASIAN OR PACIFIC ISLANDER	0	0	1	0
	BLACK	0	0	0	1
	WHITE	-1	-1	-1	-1
POVCAT99	HIGH INCOME	1	0	0	0
	LOW INCOME	0	1	0	0
	MIDDLE INCOME	0	0	1	0
	NEAR POOR	0	0	0	1
	NEGATIVE OR POOR	-1	-1	-1	-1

Output 120.2.6 displays the parameter estimates and their standard errors.

Output 120.2.6 MEPS, Parameter Estimates

Analysis of Maximum Likelihood Estimates						
Parameter		INSCOV99	Estimate	Standard Error	t Value	Pr > |t|
Intercept		ANY PRIVATE	2.7703	0.1906	14.53	<.0001
Intercept		PUBLIC ONLY	1.9216	0.1562	12.30	<.0001
TOTEXP99		ANY PRIVATE	0.000215	0.000071	3.03	0.0026
TOTEXP99		PUBLIC ONLY	0.000241	0.000072	3.34	0.0009
SEX	FEMALE	ANY PRIVATE	0.1208	0.0248	4.87	<.0001
SEX	FEMALE	PUBLIC ONLY	0.1741	0.0308	5.65	<.0001
RACEX	ALEUT, ESKIMO	ANY PRIVATE	7.1457	0.6977	10.24	<.0001
RACEX	ALEUT, ESKIMO	PUBLIC ONLY	7.6303	0.5023	15.19	<.0001
RACEX	AMERICAN INDIAN	ANY PRIVATE	-2.0904	0.2615	-7.99	<.0001
RACEX	AMERICAN INDIAN	PUBLIC ONLY	-1.8992	0.2910	-6.53	<.0001
RACEX	ASIAN OR PACIFIC ISLANDER	ANY PRIVATE	-1.8055	0.2299	-7.85	<.0001
RACEX	ASIAN OR PACIFIC ISLANDER	PUBLIC ONLY	-1.9914	0.2285	-8.71	<.0001
RACEX	BLACK	ANY PRIVATE	-1.7517	0.1984	-8.83	<.0001
RACEX	BLACK	PUBLIC ONLY	-1.7038	0.1692	-10.07	<.0001
POVCAT99	HIGH INCOME	ANY PRIVATE	1.4560	0.0685	21.26	<.0001
POVCAT99	HIGH INCOME	PUBLIC ONLY	-0.6092	0.0903	-6.75	<.0001
POVCAT99	LOW INCOME	ANY PRIVATE	-0.3066	0.0666	-4.60	<.0001
POVCAT99	LOW INCOME	PUBLIC ONLY	-0.0239	0.0754	-0.32	0.7512
POVCAT99	MIDDLE INCOME	ANY PRIVATE	0.6467	0.0587	11.01	<.0001
POVCAT99	MIDDLE INCOME	PUBLIC ONLY	-0.3496	0.0807	-4.33	<.0001
POVCAT99	NEAR POOR	ANY PRIVATE	-0.8015	0.1076	-7.45	<.0001
POVCAT99	NEAR POOR	PUBLIC ONLY	0.2985	0.0952	3.14	0.0019
NOTE: The degrees of freedom for the t tests is 317.						

Output 120.2.7 displays the odds ratio estimates and their standard errors.

Output 120.2.7 MEPS, Odds Ratios

Odds Ratio Estimates					
Effect			INSCOV99	Point Estimate	95% Confidence Limits	
TOTEXP99			ANY PRIVATE	1.000	1.000	1.000
TOTEXP99			PUBLIC ONLY	1.000	1.000	1.000
SEX	FEMALE vs MALE		ANY PRIVATE	1.273	1.155	1.404
SEX	FEMALE vs MALE		PUBLIC ONLY	1.417	1.255	1.599
RACEX	ALEUT, ESKIMO	vs WHITE	ANY PRIVATE	>999.999	>999.999	>999.999
RACEX	ALEUT, ESKIMO	vs WHITE	PUBLIC ONLY	>999.999	>999.999	>999.999
RACEX	AMERICAN INDIAN	vs WHITE	ANY PRIVATE	0.553	0.339	0.903
RACEX	AMERICAN INDIAN	vs WHITE	PUBLIC ONLY	1.146	0.602	2.182
RACEX	ASIAN OR PACIFIC ISLANDER	vs WHITE	ANY PRIVATE	0.735	0.499	1.084
RACEX	ASIAN OR PACIFIC ISLANDER	vs WHITE	PUBLIC ONLY	1.045	0.656	1.667
RACEX	BLACK	vs WHITE	ANY PRIVATE	0.776	0.638	0.944
RACEX	BLACK	vs WHITE	PUBLIC ONLY	1.394	1.133	1.715
POVCAT99	HIGH INCOME	vs NEGATIVE OR POOR	ANY PRIVATE	11.595	9.293	14.467
POVCAT99	HIGH INCOME	vs NEGATIVE OR POOR	PUBLIC ONLY	0.274	0.213	0.353
POVCAT99	LOW INCOME	vs NEGATIVE OR POOR	ANY PRIVATE	1.990	1.606	2.466
POVCAT99	LOW INCOME	vs NEGATIVE OR POOR	PUBLIC ONLY	0.492	0.395	0.615
POVCAT99	MIDDLE INCOME	vs NEGATIVE OR POOR	ANY PRIVATE	5.162	4.197	6.348
POVCAT99	MIDDLE INCOME	vs NEGATIVE OR POOR	PUBLIC ONLY	0.356	0.280	0.452
POVCAT99	NEAR POOR	vs NEGATIVE OR POOR	ANY PRIVATE	1.213	0.901	1.632
POVCAT99	NEAR POOR	vs NEGATIVE OR POOR	PUBLIC ONLY	0.680	0.526	0.878
NOTE: The degrees of freedom in computing the confidence limits is 317.						

For example, after adjusting for the effects of sex, race, and total health care expenditures, a person with high income is estimated to be 11.595 times more likely than a poor person to choose private health care insurance over no insurance, but only 0.274 times as likely to choose public health insurance over no insurance.

References

Agresti, A. (1984). *Analysis of Ordinal Categorical Data*. New York: John Wiley & Sons.

Agresti, A. (2002). *Categorical Data Analysis*. 2nd ed. New York: John Wiley & Sons.

Aitchison, J., and Silvey, S. (1957). "The Generalization of Probit Analysis to the Case of Multiple Responses." *Biometrika* 44:131–140.

Albert, A., and Anderson, J. A. (1984). "On the Existence of Maximum Likelihood Estimates in Logistic Regression Models." *Biometrika* 71:1–10.

Ashford, J. R. (1959). "An Approach to the Analysis of Data for Semi-quantal Responses in Biology Response." *Biometrics* 15:573–581.

Beaumont, J.-F., and Patak, Z. (2012). "On the Generalized Bootstrap for Sample Surveys with Special Attention to Poisson Sampling." *International Statistical Review* 80:127–148.

Binder, D. A. (1981). "On the Variances of Asymptotically Normal Estimators from Complex Surveys." *Survey Methodology* 7:157–170.

Binder, D. A. (1983). "On the Variances of Asymptotically Normal Estimators from Complex Surveys." *International Statistical Review* 51:279–292.

Binder, D. A., and Roberts, G. R. (2003). "Design-Based and Model-Based Methods for Estimating Model Parameters." In *Analysis of Survey Data*, edited by C. Skinner and R. Chambers, 29–48. New York: John Wiley & Sons.

Brick, J. M., and Kalton, G. (1996). "Handling Missing Data in Survey Research." *Statistical Methods in Medical Research* 5:215–238.

Chang, I.-M., Gelman, R., and Pagano, M. (1982). "Corrected Group Prognostic Curves and Summary Statistics." *Journal of Chronic Diseases* 35:669–674.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Collett, D. (1991). *Modelling Binary Data*. London: Chapman & Hall.

Cox, D. R., and Snell, E. J. (1989). *The Analysis of Binary Data*. 2nd ed. London: Chapman & Hall.

Dippo, C. S., Fay, R. E., and Morganstein, D. H. (1984). "Computing Variances from Complex Samples with Replicate Weights." In *Proceedings of the Survey Research Methods Section*, 489–494. Alexandria, VA: American Statistical Association.

Fay, R. E. (1984). "Some Properties of Estimates of Variance Based on Replication Methods." In *Proceedings of the Survey Research Methods Section*, 495–500. Alexandria, VA: American Statistical Association.

Fay, R. E. (1989). "Theory and Application of Replicate Weighting for Variance Calculations." In *Proceedings of the Survey Research Methods Section*, 212–217. Alexandria, VA: American Statistical Association.

Freeman, D. H., Jr. (1987). *Applied Categorical Data Analysis*. New York: Marcel Dekker.

Fuller, W. A. (1975). "Regression Analysis for Sample Survey." *Sankhyā, Series C* 37:117–132.

Fuller, W. A. (2009). *Sampling Statistics*. Hoboken, NJ: John Wiley & Sons.

Hanley, J. A., and McNeil, B. J. (1982). "The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve." *Radiology* 143:29–36.

Hidiroglou, M. A., Fuller, W. A., and Hickman, R. D. (1980). *SUPER CARP*. Ames: Iowa State University Statistical Laboratory.

Hosmer, D. W., Jr., and Lemeshow, S. (2000). *Applied Logistic Regression*. 2nd ed. New York: John Wiley & Sons.

Judkins, D. R. (1990). "Fay's Method for Variance Estimation." *Journal of Official Statistics* 6:223–239.

Kalton, G., and Kasprzyk, D. (1986). "The Treatment of Missing Survey Data." *Survey Methodology* 12:1–16.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Korn, E. L., and Graubard, B. I. (1999). *Analysis of Health Surveys*. New York: John Wiley & Sons.

Lancaster, H. O. (1961). "Significance Tests in Discrete Distributions." *Journal of the American Statistical Association* 56:223–234.

Lane, P. W., and Nelder, J. A. (1982). "Analysis of Covariance and Standardization as Instances of Prediction." *Biometrics* 38:613–621.

Lehtonen, R., and Pahkinen, E. (1995). *Practical Methods for Design and Analysis of Complex Surveys*. Chichester, UK: John Wiley & Sons.

Littell, R. C., Freund, R. J., and Spector, P. C. (1991). *SAS System for Linear Models*. 3rd ed. Cary, NC: SAS Institute Inc.

Lohr, S. L. (2022). *Sampling: Design and Analysis*. 3rd ed. Boca Raton, FL: CRC Press.

Machlin, S., Yu, W., and Zodet, M. (2005). "Computing Standard Errors for MEPS Estimates." Rockville, MD: Agency for Healthcare Research and Quality. http://www.meps.ahrq.gov/mepsweb/survey_comp/standard_errors.jsp.

Mashreghi, Z., Haziza, D., and Léger, C. (2016). "A Survey of Bootstrap Methods in Finite Population Sampling." *Statistics Surveys* 10:1–52.

McCarthy, P. J., and Snowden, C. B. (1985). *The Bootstrap and Finite Population Sampling*. Vital and Health Statistics, Series 2, no. 95. DHHS Publication Number 85–1369. Washington, DC: US Government Printing Office. https://www.cdc.gov/nchs/data/series/sr_02/sr02_095.pdf.

McCullagh, P., and Nelder, J. A. (1989). *Generalized Linear Models*. 2nd ed. London: Chapman & Hall.

McFadden, D. (1974). "Conditional Logit Analysis of Qualitative Choice Behavior." In *Frontiers in Econometrics*, edited by P. Zarembka, 105–142. New York: Academic Press.

MEPS (2002). "MEPS HC-038: 1999 Full Year Consolidated Data File." Rockville, MD: Agency for Healthcare Research and Quality. Accessed July 22, 2011. http://www.meps.ahrq.gov/mepsweb/data_stats/download_data_files_detail.jsp?cboPufNumber=HC-038.

MEPS (2006). "MEPS Survey Background." Rockville, MD: Agency for Healthcare Research and Quality. Accessed July 22, 2011. http://www.meps.ahrq.gov/mepsweb/about_meps/survey_back.jsp.

Morel, J. G. (1989). "Logistic Regression under Complex Survey Designs." *Survey Methodology* 15:203–223.

Muller, K. E., and Fetterman, B. A. (2002). *Regression and ANOVA: An Integrated Approach Using SAS Software*. Cary, NC: SAS Institute Inc.

Nagelkerke, N. J. D. (1991). "A Note on a General Definition of the Coefficient of Determination." *Biometrika* 78:691–692.

Nelder, J. A., and Wedderburn, R. W. M. (1972). "Generalized Linear Models." *Journal of the Royal Statistical Society, Series A* 135:370–384.

Rao, J. N. K., and Scott, A. J. (1981). "The Analysis of Categorical Data from Complex Surveys: Chi-Squared Tests for Goodness of Fit and Independence in Two-Way Tables." *Journal of the American Statistical Association* 76:221–230.

Rao, J. N. K., and Scott, A. J. (1984). "On Chi-Squared Tests for Multiway Contingency Tables with Cell Properties Estimated from Survey Data." *Annals of Statistics* 12:46–60.

Rao, J. N. K., and Scott, A. J. (1987). "On Simple Adjustments to Chi-Square Tests with Survey Data." *Annals of Statistics* 15:385–397.

Rao, J. N. K., Scott, A. J., and Skinner, C. J. (1998). "Quasi-score Tests with Survey Data." *Statistica Sinica* 8:1059–1070.

Rao, J. N. K., and Shao, J. (1996). "On Balanced Half-Sample Variance Estimation in Stratified Random Sampling." *Journal of the American Statistical Association* 91:343–348.

Rao, J. N. K., and Shao, J. (1999). "Modified Balanced Repeated Replication for Complex Survey Data." *Biometrika* 86:403–415.

Rao, J. N. K., and Thomas, D. R. (1989). "Chi-Squared Tests for Contingency Tables." In *Analysis of Complex Surveys*, edited by C. J. Skinner, D. Holt, and T. M. F. Smith, 89–114. New York: John Wiley & Sons.

Rao, J. N. K., and Wu, C. F. J. (1988). "Resampling Inference with Complex Survey Data." *Journal of the American Statistical Association* 83:231–241.

Rao, J. N. K., Wu, C. F. J., and Yue, K. (1992). "Some Recent Work on Resampling Methods for Complex Surveys." *Survey Methodology* 18:209–217.

Roberts, G., Rao, J. N. K., and Kumar, S. (1987). "Logistic Regression Analysis of Sample Survey Data." *Biometrika* 74:1–12.

Rust, K. (1985). "Variance Estimation for Complex Estimators in Sample Surveys." *Journal of Official Statistics* 1:381–397.

Rust, K., and Kalton, G. (1987). "Strategies for Collapsing Strata for Variance Estimation." *Journal of Official Statistics* 3:69–81.

Santner, T. J., and Duffy, D. E. (1986). "A Note on A. Albert and J. A. Anderson's Conditions for the Existence of Maximum Likelihood Estimates in Logistic Regression Models." *Biometrika* 73:755–758.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Shao, J., and Tu, D. (1995). *The Jackknife and Bootstrap*. New York: Springer-Verlag.

Sitter, R. R. (1992a). "Comparing Three Bootstrap Methods for Survey Data." *Canadian Journal of Statistics* 20:135–154.

Sitter, R. R. (1992b). "A Resampling Procedure for Complex Survey Data." *Journal of the American Statistical Association* 87:755–765.

Skinner, C. J., Holt, D., and Smith, T. M. F. (1989). *Analysis of Complex Surveys*. New York: John Wiley & Sons.

Stokes, M. E., Davis, C. S., and Koch, G. G. (2000). *Categorical Data Analysis Using the SAS System*. 2nd ed. Cary, NC: SAS Institute Inc.

Thomas, D. R., and Rao, J. N. K. (1984). "A Monte Carlo Study of Exact Levels of Goodness-of-Fit Statistics under Cluster Sampling." In *Proceedings of the Survey Research Methods Section*, 207–211. Alexandria, VA: American Statistical Association.

Thomas, D. R., and Rao, J. N. K. (1987). "Small-Sample Comparisons of Level and Power for Simple Goodness-of-Fit Statistics under Cluster Sampling." *Journal of the American Statistical Association* 82:630–636.

Thomas, D. R., Singh, A. C., and Roberts, G. R. (1996). "Tests of Independence on Two-Way Tables under Cluster Sampling: An Evaluation." *International Statistical Review* 64:295–311.

Walker, S. H., and Duncan, D. B. (1967). "Estimation of the Probability of an Event as a Function of Several Independent Variables." *Biometrika* 54:167–179.

Wolter, K. M. (2007). *Introduction to Variance Estimation*. 2nd ed. New York: Springer.

Woodruff, R. S. (1971). "A Simple Method for Approximating the Variance of a Complicated Estimate." *Journal of the American Statistical Association* 66:411–414.

Chapter 121
The SURVEYMEANS Procedure

Contents

Overview: SURVEYMEANS Procedure . . . **10102**
Getting Started: SURVEYMEANS Procedure . . . **10102**
Syntax: SURVEYMEANS Procedure . . . **10109**
- PROC SURVEYMEANS Statement . . . 10109
- BY Statement . . . 10126
- CLASS Statement . . . 10126
- CLUSTER Statement . . . 10127
- DOMAIN Statement . . . 10127
- POSTSTRATA Statement . . . 10130
- RATIO Statement . . . 10132
- REPWEIGHTS Statement . . . 10134
- STRATA Statement . . . 10136
- VAR Statement . . . 10137
- WEIGHT Statement . . . 10137

Details: SURVEYMEANS Procedure . . . **10138**
- Missing Values . . . 10138
- Survey Data Analysis . . . 10139
- Statistical Computations . . . 10140
- Replication Methods for Variance Estimation . . . 10169
- Computational Resources . . . 10175
- Output Data Sets . . . 10177
- Displayed Output . . . 10180
- ODS Table Names . . . 10186
- ODS Graphics . . . 10187

Examples: SURVEYMEANS Procedure . . . **10188**
- Example 121.1: Stratified Cluster Sample Design . . . 10188
- Example 121.2: Domain Analysis . . . 10191
- Example 121.3: Ratio Analysis . . . 10195
- Example 121.4: Analyzing Survey Data with Missing Values . . . 10195
- Example 121.5: Variance Estimation Using Replication Methods . . . 10197
- Example 121.6: Comparing Domain Means . . . 10200

References . . . **10203**

Overview: SURVEYMEANS Procedure

The SURVEYMEANS procedure estimates characteristics of a survey population by using statistics computed from a survey sample. It enables you to estimate statistics such as means, totals, proportions, quantiles, geometric means, and ratios. PROC SURVEYMEANS also provides domain analysis, which computes estimates for subpopulations or domains. PROC SURVEYMEANS also estimates variances and confidence limits and performs *t* tests for these statistics. PROC SURVEYMEANS uses either the Taylor series (linearization) method or replication (subsampling) methods to estimate sampling errors of estimators based on complex sample designs. The sample design can be a complex survey sample design with stratification, clustering, and unequal weighting. For more information, see Fuller (2009); Lohr (2022); Särndal, Swensson, and Wretman (1992); Wolter (2007).

PROC SURVEYMEANS uses the Output Delivery System (ODS), a SAS subsystem that provides capabilities for displaying and controlling the output from SAS procedures. ODS enables you to convert any of the output from PROC SURVEYMEANS into a SAS data set. For more information, see the section "ODS Table Names" on page 10186.

PROC SURVEYMEANS uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the SURVEYMEANS procedure, see the PLOTS= option in the PROC SURVEYMEANS statement and the section "ODS Graphics" on page 10187.

Getting Started: SURVEYMEANS Procedure

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/svmgs.sas`.)

This section demonstrates how you can use the SURVEYMEANS procedure to produce descriptive statistics from sample survey data. For a complete description of PROC SURVEYMEANS, see the section "Syntax: SURVEYMEANS Procedure" on page 10109. The section "Examples: SURVEYMEANS Procedure" on page 10188 provides more complicated examples to illustrate the applications of PROC SURVEYMEANS.

Simple Random Sampling

This example illustrates how you can use PROC SURVEYMEANS to estimate population means and proportions from sample survey data. The study population is a junior high school with a total of 4,000 students in grades 7, 8, and 9. Researchers want to know how much these students spend weekly for ice cream, on average, and what percentage of students spend at least $10 weekly for ice cream.

To answer these questions, 40 students were selected from the entire student population by using simple random sampling (SRS). Selection by simple random sampling means that all students have an equal chance of being selected and no student can be selected more than once. Each student selected for the sample was asked how much he or she spends for ice cream per week, on average. The SAS data set IceCream saves the responses of the 40 students:

```
data IceCream;
   input Grade Spending @@;
   if (Spending < 10) then Group='less';
   else Group='more';
   datalines;
7 7  7  7  8 12  9 10  7  1  7 10  7  3  8 20  8 19  7 2
7 2  9 15  8 16  7  6  7  6  7  6  9 15  8 17  8 14  9 8
9 8  9  7  7  3  7 12  7  4  9 14  8 18  9  9  7  2  7 1
7 4  7 11  9  8  8 10  8 13  7  2  9  6  9 11  7  2  7 9
;
```

The variable Grade contains a student's grade. The variable Spending contains a student's response regarding how much he spends per week for ice cream, in dollars. The variable Group is created to indicate whether a student spends at least $10 weekly for ice cream: Group='more' if a student spends at least $10, or Group='less' if a student spends less than $10.

You can use PROC SURVEYMEANS to produce estimates for the entire student population, based on this random sample of 40 students:

```
ods graphics on;
title1 'Analysis of Ice Cream Spending';
title2 'Simple Random Sample Design';
proc surveymeans data=IceCream total=4000;
   var Spending Group;
run;
```

The PROC SURVEYMEANS statement invokes the procedure. The TOTAL=4000 option specifies the total number of students in the study population, or school. PROC SURVEYMEANS uses this total to adjust variance estimates for the effects of sampling from a finite population. The VAR statement names the variables to analyze, Spending and Group.

Figure 121.1 displays the results from this analysis. There are a total of 40 observations used in the analysis. The "Class Level Information" table lists the two levels of the variable Group. This variable is a character variable, and so PROC SURVEYMEANS provides a categorical analysis for it, estimating the relative frequency or proportion for each level. If you want a categorical analysis for a numeric variable, you can name that variable in the CLASS statement.

Figure 121.1 Analysis of Ice Cream Spending

Analysis of Ice Cream Spending
Simple Random Sample Design

The SURVEYMEANS Procedure

Data Summary	
Number of Observations	40

Class Level Information		
Variable	Levels	Values
Group	2	less more

Figure 121.1 *continued*

Statistics						
Variable	Level	N	Mean	Std Error of Mean	95% CL for Mean	
Spending		40	8.750000	0.845139	7.04054539	10.4594546
Group	less	23	0.575000	0.078761	0.41568994	0.7343101
	more	17	0.425000	0.078761	0.26568994	0.5843101

The "Statistics" table displays the estimates for each analysis variable. By default, PROC SURVEYMEANS displays the number of observations, the estimate of the mean, its standard error, and the 95% confidence limits for the mean. You can obtain other statistics by specifying the corresponding *statistic-keywords* in the PROC SURVEYMEANS statement.

The estimate of the average weekly ice cream expense is $8.75 for students at this school. The standard error of this estimate if $0.85, and the 95% confidence interval for weekly ice cream expense is from $7.04 to $10.46. The analysis variable Group is a character variable, and so PROC SURVEYMEANS analyzes it as categorical, estimating the relative frequency or proportion for each level or category. These estimates are displayed in the Mean column of the "Statistics" table. It is estimated that 57.5% of all students spend less than $10 weekly on ice cream, while 42.5% of the students spend at least $10 weekly. The standard error of each estimate is 7.9%.

When ODS Graphics is enabled, PROC SURVEYMEANS also displays plots that depict the distribution of the continuous variables. Figure 121.2 displays such a plot for the variable Spending.

Figure 121.2 Distribution of Spending

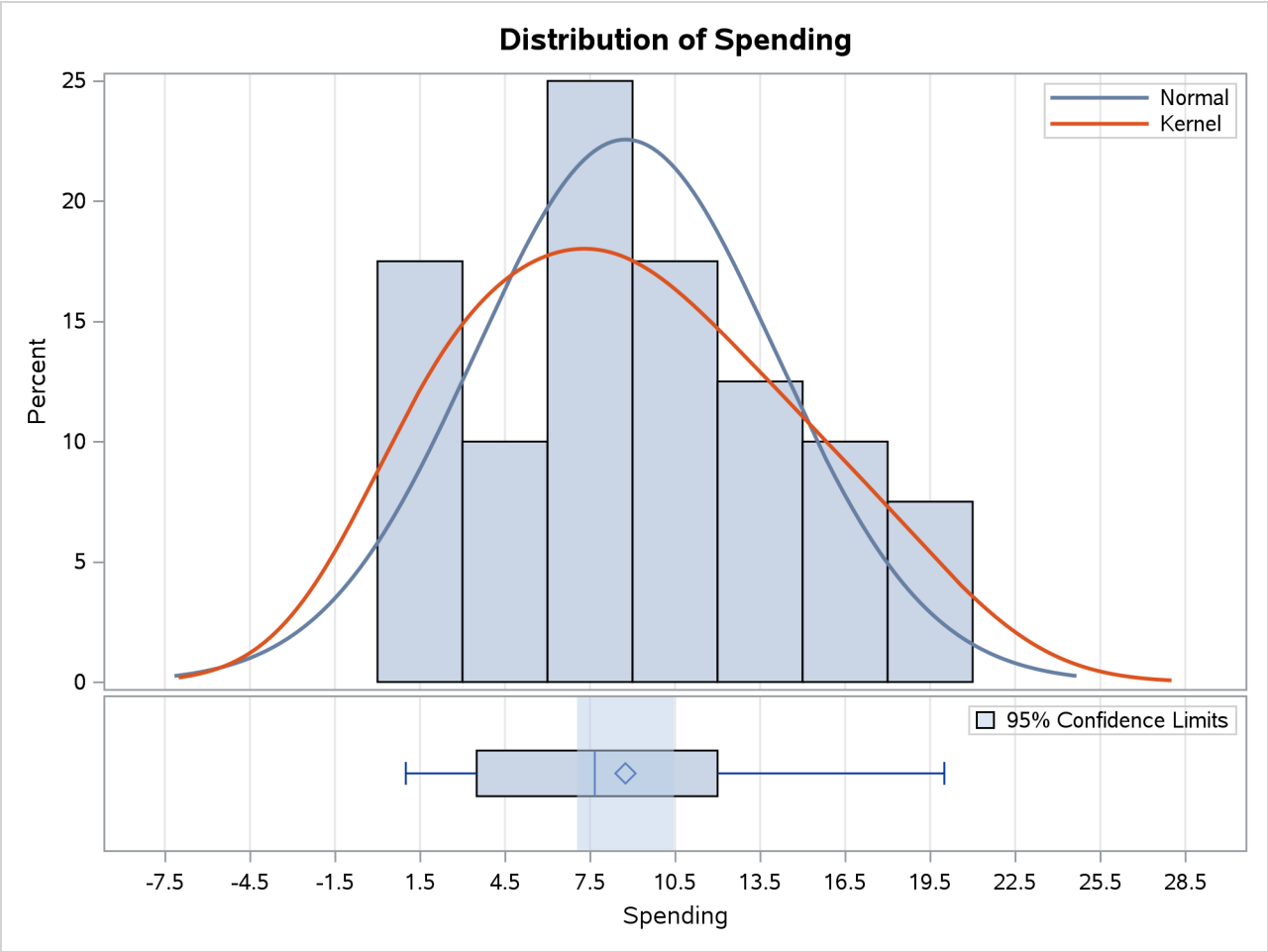

Stratified Sampling

Suppose that the sample of students described in the previous section was actually selected by using stratified random sampling. In stratified sampling, the study population is divided into nonoverlapping strata, and samples are selected from each stratum independently.

The list of students in this junior high school was stratified by grade, yielding three strata: grades 7, 8, and 9. A simple random sample of students was selected from each grade. Table 121.1 shows the total number of students in each grade.

Table 121.1 Number of Students by Grade

Grade	Number of Students
7	1,824
8	1,025
9	1,151
Total	4,000

To analyze this stratified sample, you need to provide the population totals for each stratum to PROC SURVEYMEANS. The SAS data set StudentTotals contains the information from Table 121.1:

```
data StudentTotals;
   input Grade _total_;
   datalines;
7 1824
8 1025
9 1151
;
```

The variable Grade is the stratum identification variable, and the variable _TOTAL_ contains the total number of students for each stratum. PROC SURVEYMEANS requires you to use the variable name _TOTAL_ for the stratum population totals.

PROC SURVEYMEANS uses the stratum population totals to adjust variance estimates for the effects of sampling from a finite population. If you do not provide population totals or sampling rates, then PROC SURVEYMEANS assumes that the proportion of the population in the sample is very small, and the computation does not involve a finite population correction.

In a stratified sample design, when the sampling rates in the strata are unequal, you need to use sampling weights to reflect this information in order to produce an unbiased mean estimator. In this example, the appropriate sampling weights are reciprocals of the probabilities of selection. You can use the following DATA step to create the sampling weights:

```
data IceCream;
   set IceCream;
   if Grade=7 then Prob=20/1824;
   if Grade=8 then Prob=9/1025;
   if Grade=9 then Prob=11/1151;
   Weight=1/Prob;
run;
```

When you use PROC SURVEYSELECT to select your sample, the procedure creates these sampling weights for you.

The following SAS statements perform the stratified analysis of the survey data:

```
title1 'Analysis of Ice Cream Spending';
title2 'Stratified Sample Design';
proc surveymeans data=IceCream total=StudentTotals;
   stratum Grade / list;
   var Spending Group;
   weight Weight;
run;
```

The PROC SURVEYMEANS statement invokes the procedure. The DATA= option names the SAS data set IceCream as the input data set to be analyzed. The TOTAL= option names the data set StudentTotals as the input data set that contains the stratum population totals. Comparing this to the analysis in the section "Simple Random Sampling" on page 10102, notice that the TOTAL=StudentTotals option is used here instead of the TOTAL=4000 option. In this stratified sample design, the population totals are different for different strata, and so you need to provide them to PROC SURVEYMEANS in a SAS data set.

The STRATA statement identifies the stratification variable Grade. The LIST option in the STRATA statement requests that PROC SURVEYMEANS display stratum information. The WEIGHT statement tells PROC

SURVEYMEANS that the variable Weight contains the sampling weights.

Figure 121.3 displays information about the input data set. There are three strata in the design and 40 observations in the sample. The categorical variable Group has two levels, 'less' and 'more.'

Figure 121.4 displays information for each stratum. The table displays a stratum index and the values of the STRATA variable. The stratum index identifies each stratum by a sequentially assigned number. For each stratum, the table gives the population total (total number of students), the sampling rate, and the sample size. The stratum sampling rate is the ratio of the number of students in the sample to the number of students in the population for that stratum. The table also lists each analysis variable and the number of stratum observations for that variable. For categorical variables, the table lists each level and the number of sample observations in that level.

Figure 121.3 Data Summary

Analysis of Ice Cream Spending
Stratified Sample Design

The SURVEYMEANS Procedure

Data Summary	
Number of Strata	3
Number of Observations	40
Sum of Weights	4000

Class Level Information		
Variable	Levels	Values
Group	2	less more

Figure 121.4 Stratum Information

Stratum Information							
Stratum Index	Grade	Pop. Total	Sampling Rate	N Obs	Variable	Level	N
1	7	1824	1.10%	20	Spending		20
					Group	less	17
						more	3
2	8	1025	0.88%	9	Spending		9
					Group	less	0
						more	9
3	9	1151	0.96%	11	Spending		11
					Group	less	6
						more	5

Figure 121.5 shows the following:

- The estimate of average weekly ice cream expense is \$9.14 for students in this school, with a standard error of \$0.53, and a 95% confidence interval from \$8.06 to \$10.22.
- An estimate of 54.5% of all students spend less than \$10 weekly on ice cream, and 45.5% spend more, with a standard error of 5.8%.

Figure 121.5 Analysis of Ice Cream Spending

Statistics						
Variable	Level	N	Mean	Std Error of Mean	95% CL for Mean	
Spending		40	9.141298	0.531799	8.06377052	10.2188254
Group	less	23	0.544555	0.058424	0.42617678	0.6629323
	more	17	0.455445	0.058424	0.33706769	0.5738232

Output Data Sets

PROC SURVEYMEANS uses the Output Delivery System (ODS) to create output data sets. This is a departure from older SAS procedures that provide OUTPUT statements for similar functionality. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

For example, to save the "Statistics" table shown in Figure 121.5 in the previous section in an output data set, you use the ODS OUTPUT statement as follows:

```
title1 'Analysis of Ice Cream Spending';
title2 'Stratified Sample Design';
proc surveymeans data=IceCream total=StudentTotals;
   stratum Grade / list;
   var Spending Group;
   weight Weight;
   ods output Statistics=MyStat;
run;
```

The statement

```
ods output Statistics=MyStat;
```

requests that the "Statistics" table that appears in Figure 121.5 be placed in a SAS data set MyStat.

The PRINT procedure displays observations of the data set MyStat:

```
proc print data=MyStat;
run;
```

Figure 121.6 displays the data set MyStat. The section "ODS Table Names" on page 10186 gives the complete list of tables produced by PROC SURVEYMEANS.

Figure 121.6 Output Data Set MyStat

Analysis of Ice Cream Spending
Stratified Sample Design

Obs	VarName	VarLevel	N	Mean	StdErr	LowerCLMean	UpperCLMean
1	Spending		40	9.141298	0.531799	8.06377052	10.2188254
2	Group	less	23	0.544555	0.058424	0.42617678	0.6629323
3	Group	more	17	0.455445	0.058424	0.33706769	0.5738232

Syntax: SURVEYMEANS Procedure

The following statements are available in the SURVEYMEANS procedure:

PROC SURVEYMEANS < *options* > < *statistic-keywords* > **;**
BY *variables* **;**
CLASS *variables* **;**
CLUSTER *variables* **;**
DOMAIN *variables* < *variable*∗*variable variable*∗*variable*∗*variable . . .* > < / *option* > **;**
POSTSTRATA *variables* / **PSTOTAL=** < *option* > **;**
POSTSTRATA *variables* / **PSPCT=** < *option* > **;**
RATIO < '*label*' > *variables* / *variables* < *(options)* > **;**
REPWEIGHTS *variables* < / *options* > **;**
STRATA *variables* < / *option* > **;**
VAR *variables* **;**
WEIGHT *variable* **;**

The PROC SURVEYMEANS statement invokes the procedure. It optionally names the input data sets, specifies statistics for the procedure to compute, and specifies the variance estimation method. The PROC SURVEYMEANS statement is required.

The VAR statement identifies the variables to be analyzed. The CLASS statement identifies numeric variables that are to be analyzed as categorical variables. The STRATA statement lists the variables that form the strata in a stratified sample design. The CLUSTER statement specifies cluster identification variables in a clustered sample design. The DOMAIN statement lists the variables that define domains for subpopulation analysis. The RATIO statement requests ratio analysis for means or proportions of analysis variables. The WEIGHT statement names the sampling weight variable. The POSTSTRATA statement lists the variables that are used to form poststrata for poststratification. The REPWEIGHTS statement names replicate weight variables that are used by replication variance estimation methods. You can use a BY statement with PROC SURVEYMEANS to obtain separate analyses for groups that are defined by the BY variables.

All statements can appear multiple times except the PROC SURVEYMEANS statement, POSTSTRATA statement, and WEIGHT statement, each of which can appear only once.

The following sections describe the PROC SURVEYMEANS statement and then describe the other statements in alphabetical order.

PROC SURVEYMEANS Statement

PROC SURVEYMEANS < *options* > *statistic-keywords* **;**

The PROC SURVEYMEANS statement invokes the SURVEYMEANS procedure. In this statement, you identify the data set to be analyzed, specify the variance estimation method, and provide sample design information. The DATA= option names the input data set to be analyzed. The VARMETHOD= option specifies the variance estimation method, which is the Taylor series method by default. For Taylor series variance estimation, you can include a finite population correction factor in the analysis by providing either the sampling rate or population total with the RATE= or TOTAL= option. If your design is stratified, with different sampling rates or totals for different strata, then you can input these stratum rates or totals in a SAS data set that contains the stratification variables.

In the PROC SURVEYMEANS statement, you also can use *statistic-keywords* to specify statistics, such as population mean and population total, for PROC SURVEYMEANS to compute. You can also request data set summary information and sample design information.

Table 121.2 summarizes the *options* available in the PROC SURVEYMEANS statement.

Table 121.2 PROC SURVEYMEANS Statement Options

Option	Description
ALPHA=	Sets the confidence level for confidence limits
DATA=	Specifies the SAS data set to be analyzed
MISSING	Treats missing values as a valid category
NOMCAR	Computes variance estimates by analyzing the nonmissing values as a domain
NONSYMCL	Requests nonsymmetric confidence limits for quantiles
NOSPARSE	Suppresses the display of analysis variables with zero frequency
ORDER=	Specifies the order in which to report the values of the categorical variables
PERCENTILE=	Specifies percentiles that you want the procedure to compute
PLOTS=	Requests plots from ODS Graphics
QUANTILE=	Specifies quantiles that you want the procedure to compute
RATE=	Specifies the sampling rate
STACKING	Produces the output data sets by using a stacking table structure
TOTAL=	Specifies the total number of primary sampling units
VARHEADER=	Specifies the variable identification to display
VARMETHOD=	Specifies the variance estimation method

You can specify the following *options* in the PROC SURVEYMEANS statement:

ALPHA=α

sets the confidence level for confidence limits. The value of the ALPHA= option must be between 0 and 1, and the default value is 0.05. A confidence level of α produces $100(1-\alpha)\%$ confidence limits. The default of ALPHA=0.05 produces 95% confidence limits.

DATA=*SAS-data-set*

specifies the SAS data set to be analyzed by PROC SURVEYMEANS. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

MISSING

treats missing values as a valid (nonmissing) category for all categorical variables, which include CLASS, STRATA, CLUSTER, DOMAIN, and POSTSTRATA variables.

By default, if you do not specify the MISSING option, an observation is excluded from the analysis if it has a missing value. For more information, see the section "Missing Values" on page 10138.

NOMCAR

treats missing values in the variance computation as *not missing completely at random* (NOMCAR) for Taylor series variance estimation. When you specify this option, PROC SURVEYMEANS computes variance estimates by analyzing the nonmissing values as a domain (subpopulation), where the entire population includes both nonmissing and missing domains. For more information, see the section "Missing Values" on page 10138.

By default, PROC SURVEYMEANS completely excludes an observation from analysis if that observation has a missing value, unless you specify the MISSING option for categorical variables. Note that the NOMCAR option has no effect on a categorical variable when you specify the MISSING option, which treats missing values as a valid nonmissing level.

The NOMCAR option applies only to Taylor series variance estimation; it is ignored for replication methods.

NONSYMCL

requests nonsymmetric confidence limits for quantiles when you request quantiles with PERCENTILE= or QUANTILE= option. This option applies only to the default VARMETHOD=TAYLOR option. For more details, see the section "Confidence Limits" on page 10156.

NOSPARSE

suppresses the display of analysis variables with zero frequency. By default, the procedure displays all continuous variables and all levels of categorical variables.

ORDER=DATA | FORMATTED | INTERNAL

specifies the order in which the values of the categorical variables are to be reported.

This option also determines the sort order for the levels of ClUSTER and DOMAIN variables and controls STRATA variable levels in the "Stratum Information" table.

The following shows how PROC SURVEYMEANS interprets values of the ORDER= option:

DATA orders values according to their order in the input data set.

FORMATTED orders values by their formatted values. This order is operating environment dependent. By default, the order is ascending.

INTERNAL orders values by their unformatted values, which yields the same order that the SORT procedure does. This order is operating environment dependent.

By default, ORDER=INTERNAL. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PERCENTILE=(*values*)

specifies percentiles you want the procedure to compute. You can separate values with blanks or commas. Each value must be between 0 and 100. You can also use the *statistic-keywords* DECILES, MEDIAN, Q1, Q3, and QUARTILES to request common percentiles.

PROC SURVEYMEANS uses Woodruff's method (Dorfman and Valliant 1993; Särndal, Swensson, and Wretman 1992; Francisco and Fuller 1991) to estimate the variances of quantiles. For more details, see the section "Quantiles" on page 10153.

PLOTS < (*global-plot-options*) > < = *plot-request* < (*plot-option*) > >

PLOTS < (*global-plot-options*) > < = (*plot-request* < (*plot-option*) > < . . . *plot-request* < (*plot-option*) > >) >

controls the plots that are produced through ODS Graphics.

A *plot-request* identifies the plot, and a *plot-option* controls the appearance and content of the plot. You can specify *plot-options* in parentheses after a *plot-request*. A *global-plot-option* applies to all plots for which it is available. You can specify *global-plot-options* in parentheses after the PLOTS option.

When you specify only one *plot-request*, you can omit the parentheses around it. Here are a few examples of requesting plots:

```
plots=all
plots(unpack)=summary
plots=(summary(unpack) domain)
plots=boxplot
plots=(domain(packvar) histogram)
```

You can suppress default plots and request specific plots by specifying the PLOTS(ONLY)= option; PLOTS(ONLY)=(*plot-requests*) produces only the plots that are specified as *plot-requests*.

ODS Graphics must be enabled before you can request plots. For example:

```
ods graphics on;
proc surveymeans plots=boxplot;
   variable income;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled but you do not specify the PLOTS= option, PROC SURVEYMEANS produces summary plots, and it also produces domain plots when you specify a DOMAIN statement. You can suppress all plots by specifying PLOTS=NONE.

For a continuous analysis variable, PROC SURVEYMEANS provides a summary plot, which contains a box plot and a histogram plot. For a categorical variable, PROC SURVEYFREQ provides corresponding plots for it.

For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Global Plot Option

A *global-plot-option* applies to all plots for which the option is available. You can specify the following *global-plot-options*:

ONLY
suppresses the default plots and requests only the plots that are specified as *plot-requests*.

NBINS=*value*
specifies the number of bins in a histogram plot. If you do not specify this option, then by default the number of bins is determined by the method of Terrell and Scott (1985).

UNPACK
requests that the procedure create a histogram with overlaid densities and a box plot along with a confidence interval band separately.

Plot Requests

You can specify the following *plot-requests*:

ALL
requests all appropriate plots.

BOXPLOT | BOX
requests a box plot for continuous variables.

DOMAIN < (*plot-options* **) >**
requests box plots for domain statistics for each domain definition. By default, the procedure plots each domain in a single panel for all continuous analysis variables. This plot is produced by default if you specify a DOMAIN statement. You can specify the following *plot-options*:

EXCLUDE
requests that the procedure create box plots for every domain level of a domain but exclude the box plot for the full sample. By default, the box plot includes the full sample box plot.

PACKDOMAIN
requests box plots for all domain definitions in one panel for each analysis variable.

PACKVAR
requests box plots for all analysis variables in one panel for each domain definition. This is the default when you do not specify the UNPACK option.

UNPACK
requests a box plot for each domain and for each analysis variable in a single panel.

HISTOGRAM < (*plot-option*) >

HIST < (*plot-option*) >

requests a histogram with overlaid normal and kernel densities. You can specify the following *plot-option*:

NBINS=*value*

specifies the number of bins in a histogram plot. If you do not specify this option, then by default the number of bins is determined by the method of Terrell and Scott (1985).

NONE

suppresses all plots.

SUMMARY < (*plot-options*) >

requests that a histogram and a box plot be displayed together in a single panel, sharing the same X axis. This packed plot is produced by default. You can specify the following *plot-options*:

NBINS=*value*

specifies the number of bins in a histogram plot. If you do not specify this option, then by default the number of bins is determined by the method of Terrell and Scott (1985).

UNPACK

requests that a histogram with overlaid densities be displayed in one panel and a box plot along with a confidence interval band be displayed separately. Note that specifying PLOTS(ONLY)=SUMMARY(UNPACK) is exactly the same as specifying PLOTS(ONLY)=(BOX HISTOGRAM).

PLOTS=SUMMARY overwrites the PLOTS=BOX and the PLOTS=HISTOGRAM *plot-requests*. That is, if you do not specify the UNPACK option, PROC SURVEYMEANS does not display a histogram plot or a box plot by itself when PLOTS=SUMMARY is specified.

QUANTILE=(*values*)

specifies quantiles you want the procedure to compute. You can separate values with blanks or commas. Each value must be between 0 and 1. You can also use the *statistic-keywords* DECILES, MEDIAN, Q1, Q3, and QUARTILES to request common quantiles.

PROC SURVEYMEANS uses Woodruff's method (Dorfman and Valliant 1993; Särndal, Swensson, and Wretman 1992; Francisco and Fuller 1991) to estimate the variances of quantiles. For more details, see the section "Quantiles" on page 10153.

RATE=*value* | *SAS-data-set*

R=*value* | *SAS-data-set*

specifies the sampling rate, which PROC SURVEYMEANS uses to compute a finite population correction for Taylor series or bootstrap variance estimation. This option is ignored for the jackknife or balanced repeated replication (BRR) variance estimation method.

If your sample design has multiple stages, you should specify the *first-stage sampling rate*, which is the ratio of the number of primary sampling units (PSUs) in the sample to the total number of PSUs in the population.

You can specify the sampling rate in either of the following ways:

value specifies a nonnegative number to use for a nonstratified design or for a stratified design that has the same sampling rate in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the sampling rates for a stratified design that has different sampling rates in the strata. You must provide the sampling rates in the data set variable named _RATE_. The sampling rates must be nonnegative numbers.

You can specify sampling rates as numbers between 0 and 1. Or you can specify sampling rates in percentage form as numbers between 1 and 100, which PROC SURVEYMEANS converts to proportions. The procedure treats the value 1 as 100% instead of 1%.

For more information, see the section "Specification of Population Totals and Sampling Rates" on page 10139.

If you do not specify either the RATE= or TOTAL= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. You cannot specify both the RATE= and TOTAL= options.

STACKING

requests that the procedure produce the output data sets by using a stacking table structure, which was the default before SAS 9. The new default is to produce a rectangular table structure in the output data sets.

A rectangular structure creates one observation for each analysis variable in the data set. A stacking structure creates only one observation in the output data set for all analysis variables.

The STACKING option affects the following tables:

- Domain
- Statistics
- StrataInfo

For more information, see the section "Rectangular and Stacking Structures in an Output Data Set" on page 10178.

TOTAL=*value* | *SAS-data-set*

N=*value* | *SAS-data-set*

specifies the total number of primary sampling units (PSUs) in the study population. PROC SURVEYMEANS uses this information to compute a finite population correction for Taylor series or bootstrap variance estimation. This option is ignored for the jackknife or BRR variance estimation method.

You can specify the total number of PSUs in either of the following ways:

value specifies a positive number to use for a nonstratified design or for a stratified design that has the same population total in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the population totals for a stratified design that has different population totals in the strata. You must provide the stratum totals in the data set variable named _TOTAL_. The stratum totals must be positive numbers.

For more information, see the section "Specification of Population Totals and Sampling Rates" on page 10139.

If you do not specify either the TOTAL= or RATE= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. You cannot specify both the TOTAL= and RATE= options.

statistic-keywords

specifies the statistics for the procedure to compute. If you do not specify any *statistic-keywords*, PROC SURVEYMEANS computes the NOBS, MEAN, STDERR, and CLM statistics by default.

The statistics produced depend on the type of the analysis variable. If you name a numeric variable in the CLASS statement, then the procedure analyzes that variable as a categorical variable. The procedure always analyzes character variables as categorical. For more information, see the section "CLASS Statement" on page 10126.

PROC SURVEYMEANS computes MIN, MAX, and RANGE for numeric variables but not for categorical variables. For numeric variables, the keyword MEAN produces the mean, but for categorical variables it produces the proportion in each category or level. Also, for categorical variables, the keyword NOBS produces the number of observations for each variable level, and the keyword NMISS produces the number of missing observations for each level. If you request the keyword NCLUSTER for a categorical variable, PROC SURVEYMEANS displays for each level the number of clusters with observations in that level. PROC SURVEYMEANS computes SUMWGT in the same way for both categorical and numeric variables, as the sum of the weights over all nonmissing observations.

PROC SURVEYMEANS performs univariate analysis, analyzing each variable separately. Thus the number of nonmissing and missing observations might not be the same for all analysis variables. For more information, see the section "Missing Values" on page 10138.

The following statistics are available for ratios (which you request with a RATIO statement): N, NCLU, SUMWGT, RATIO, STDERR, DF, T, PROBT, and CLM, as shown in the following list. If no statistics are requested, the procedure computes the ratio and its standard error by default.

You can specify the following *statistic-keywords*:

ALL requests all available statistics except those that are associated with geometric means.

ALLGEO requests all available statistics that are associated with geometric means.

CLM requests the $100(1-\alpha)\%$ two-sided confidence limits for MEAN, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

CLSUM requests the $100(1-\alpha)\%$ two-sided confidence limits for SUM, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

CV requests the coefficient of variation for MEAN.

CVSUM requests the coefficient of variation for SUM.

DECILES requests the 10th through the 90th percentiles, including their standard errors and confidence limits.

DEFF requests the design effect for MEAN.

DF requests the degrees of freedom for the *t* test.

GEOMEAN requests the geometric mean of a numeric variable that contains positive values.

GMCLM requests the $100(1-\alpha)\%$ two-sided confidence limits for GEOMEAN, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

GMSTDERR requests the standard error of GEOMEAN. When you specify GEOMEAN, SURVEYMEANS procedure computes GMSTDERR by default.

LCLM requests the $100(1-\alpha)\%$ one-sided lower confidence limit for MEAN, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

LCLSUM requests the $100(1-\alpha)\%$ one-sided lower confidence limit for SUM, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

LGMCLM requests the $100(1-\alpha)\%$ one-sided lower confidence limit for GEOMEAN, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

MAX requests the maximum value.

MEAN requests the mean for a numeric variable, or the proportion in each category for a categorical variable.

MEDIAN requests the median (50th percentile) for a numeric variable.

MIN requests the minimum value.

NCLUSTER requests the number of clusters.

NMISS requests the number of missing observations.

NOBS requests the number of nonmissing observations.

Q1 requests the lower quartile (25th percentile).

Q3 requests the upper quartile (75th percentile).

QUARTILES requests Q1 (25th percentile), MEDIAN (50th percentile), and Q3 (75th percentile), including their standard errors and confidence limits.

RANGE requests the range, MAX–MIN.

RATIO requests the ratio of means or proportions.

STD requests the standard deviation of SUM. When you request SUM, the procedure computes STD by default.

STDERR requests the standard error of MEAN or RATIO. When you request MEAN or RATIO, the procedure computes STDERR by default.

SUM requests the weighted sum, $\sum w_i y_i$, or estimated population total when the appropriate sampling weights are used.

SUMWGT requests the sum of the weights, $\sum w_i$.

T requests the t value and its corresponding p-value with DF degrees of freedom for $H_0\colon \theta = 0$, where θ is a requested statistic.

UCLM requests the $100(1-\alpha)\%$ one-sided upper confidence limit for MEAN, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

UCLSUM requests the $100(1-\alpha)\%$ one-sided upper confidence limit for SUM, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

UGMCLM requests the $100(1-\alpha)\%$ one-sided upper confidence limit for GEOMEAN, where α is determined by the ALPHA= option; the default is $\alpha = 0.05$.

VAR requests the variance of MEAN or RATIO.

VARSUM requests the variance of SUM.

For details about how PROC SURVEYMEANS computes these statistics, see the section "Statistical Computations" on page 10140.

VARHEADER=LABEL | NAME | NAMELABEL

specifies the variable identification to use in the displayed output. This option controls the headings of the DOMAIN variable in domain analysis output and of the STRATUM variables in the "Stratum Information" table. By default, VARHEADER=NAME.

You can specify the following values:

LABEL displays the variable label.

NAME displays the variable name.

NAMELABEL displays the variable name and label as *Name* (*Label*).

This option has no effect on tables that use the STACKING option.

VARMETHOD=*method* **<(***method-options***)>**

specifies the variance estimation *method*. PROC SURVEYMEANS provides the Taylor series method and the following replication (resampling) methods: balanced repeated replication (BRR), bootstrap, and jackknife.

Table 121.3 summarizes the available *methods* and *method-options*.

Table 121.3 Variance Estimation Methods

method	**Variance Estimation Method**	*method-options*
BOOTSTRAP	Bootstrap	CENTER= DFADJ MH=*value* | *SAS-data-set* OUTWEIGHTS=*SAS-data-set* REPS=*number* SEED=*number*
BRR	Balanced repeated replication	CENTER= DFADJ FAY <=*value*> HADAMARD=*SAS-data-set* NAIVEQVAR OUTWEIGHTS=*SAS-data-set* PRINTH REPS=*number*
JACKKNIFE | JK	Jackknife	CENTER= DFADJ NAIVEQVAR OUTJKCOEFS=*SAS-data-set* OUTWEIGHTS=*SAS-data-set*
TAYLOR	Taylor series linearization	None

For VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, and VARMETHOD=JACKKNIFE, you can specify *method-options* in parentheses after the variance estimation *method*. For example:

```
varmethod=BRR(reps=60 outweights=myReplicateWeights)
```

By default, VARMETHOD=JACKKNIFE if you also specify a REPWEIGHTS statement; otherwise, VARMETHOD=TAYLOR by default.

You can specify the following *methods*:

BOOTSTRAP <(*method-options*)>

requests variance estimation by the bootstrap method. For more information, see the section "Bootstrap Method" on page 10169.

The bootstrap method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you use a REPWEIGHTS statement to provide replicate weights.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Bootstrap Method" on page 10169.

DFADJ

computes the degrees of freedom by using the number of nonempty strata for an analysis variable. The degrees of freedom for VARMETHOD=JACKKNIFE equals the number of clusters (or number of observations if there are no clusters) minus the number of strata (or one if there are no strata). By default, the number of strata is based on all valid observations in the data set. But if you specify this *method-option*, PROC SURVEYMEANS does not count any empty strata that are caused by all observations containing missing values for an analysis variable.

For more information, see the section "Degrees of Freedom" on page 10144. For more information about valid observations, see the section "Data and Sample Design Summary" on page 10180.

This *method-option* has no effect on categorical variables when you specify the MISSING option, which treats missing values as a valid nonmissing level.

This *method-option* cannot be used when you provide replicate weights in a REPWEIGHTS statement. When you use a REPWEIGHTS statement, the degrees of freedom equals the number of REPWEIGHTS variables (replicates) unless you specify an alternative value in the DF= option in the REPWEIGHTS statement.

MH=*value* | **(***values***)** | *SAS-data-set*

specifies the number of PSUs to select for the bootstrap replicate samples. You can provide bootstrap stratum sample sizes m_h by specifying a list of *values* or a *SAS-data-set*. Alternatively, you can provide a single bootstrap sample size *value* to use for all strata or for a nonstratified design. You can specify the number of replicate samples in the REPS= option. For more information, see the section "Bootstrap Method" on page 10169.

Each bootstrap sample size m_h must be a positive integer and must be less than n_h, which is the total number of PSUs in stratum h. By default, $m_h = n_h - 1$ for a stratified design. For a nonstratified design, the bootstrap sample size *value* must be less than n (the total number of PSUs in the sample). By default, $m = n - 1$ for a nonstratified design.

You can provide bootstrap sample sizes by specifying one of the following forms:

MH=*value*
specifies a single bootstrap sample size *value* to use for all strata or for a nonstratified design.

MH=(*values***)**
specifies a list of stratum bootstrap sample size *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. The number of values must not be less than the number of strata in the DATA= input data set.

Each stratum sample size value must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

MH=*SAS-data-set*
names a *SAS-data-set* that contains the stratum bootstrap sample sizes. You must provide the sample sizes in a data set variable named _NSIZE_ or SampleSize.

The *SAS-data-set* must contain all stratification variables that you specify in the STRATA statement. It must also contain all stratum levels that appear in the DATA= input data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets.

Each value of the _NSIZE_ or SampleSize variable must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

OUTWEIGHTS=*SAS-data-set*
names a *SAS-data-set* in which to store the bootstrap replicate weights that PROC SURVEYMEANS creates. For information about replicate weights, see the section "Bootstrap Method" on page 10169. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10177.

This *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

REPS=*number*
specifies the *number* of replicates for bootstrap variance estimation. The value of *number* must be an integer greater than 1. Increasing the number of replicates improves the estimation precision but also increases the computation time. By default, REPS=250.

SEED=*number*
specifies the initial seed for random number generation for bootstrap replicate sampling.

If you do not specify this option or if you specify a *number* that is negative or 0, PROC SURVEYMEANS uses the time of day from the system clock to obtain an initial seed.

To reproduce the same bootstrap replicate weights and the same analysis in a subsequent execution of PROC SURVEYMEANS, you can specify the same initial seed that was used in the original analysis.

PROC SURVEYMEANS displays the value of the initial seed in the "Variance Estimation" table.

BRR < (*method-options*) >

requests variance estimation by balanced repeated replication (BRR). This method requires a stratified sample design where each stratum contains two primary sampling units (PSUs). When you specify this method, you must also specify a STRATA statement unless you provide replicate weights by using the REPWEIGHTS statement. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10171.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10171.

DFADJ

computes the degrees of freedom by using the number of nonempty strata for an analysis variable. The degrees of freedom for VARMETHOD=BRR equals the number of strata; by default, that number is based on all valid observations in the data set. But if you specify this *method-option*, PROC SURVEYMEANS does not count any empty strata that are caused by all observations containing missing values for an analysis variable.

For more information, see the section "Degrees of Freedom" on page 10144. For more information about valid observations, see the section "Data and Sample Design Summary" on page 10180.

This *method-option* has no effect on categorical variables when you specify the MISSING option, which treats missing values as a valid nonmissing level.

This *method-option* cannot be used when you provide replicate weights in a REPWEIGHTS statement. When you use a REPWEIGHTS statement, the degrees of freedom equals the number of REPWEIGHTS variables (replicates) unless you specify an alternative value in the DF= option in the REPWEIGHTS statement.

FAY < =*value* >

requests Fay's method, which is a modification of the BRR method. For more information, see the section "Fay's BRR Method" on page 10172.

You can specify the *value* of the Fay coefficient, which is used in converting the original sampling weights to replicate weights. The Fay coefficient must be a nonnegative number less than 1. By default, the Fay coefficient is 0.5.

HADAMARD=*SAS-data-set*

H=*SAS-data-set*

names a *SAS-data-set* that contains the Hadamard matrix for BRR replicate construction. If you do not specify this *method-option*, PROC SURVEYMEANS generates an appropriate Hadamard matrix for replicate construction. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 10171 and "Hadamard Matrix" on page 10175.

If a Hadamard matrix of a particular dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a *SAS-data-set* in this *method-option*.

In this *SAS-data-set*, each variable corresponds to a column and each observation corresponds to a row of the Hadamard matrix. You can use any variable names in this data set. All values in the data set must equal either 1 or –1. You must ensure that the matrix you provide is indeed a Hadamard matrix—that is, $\mathbf{A}'\mathbf{A} = R\,\mathbf{I}$, where $\mathbf{A}$ is the Hadamard matrix of dimension R and $\mathbf{I}$ is an identity matrix. PROC SURVEYMEANS does not check the validity of the Hadamard matrix that you provide.

The *SAS-data-set* must contain at least H variables, where H denotes the number of first-stage strata in your design. If the data set contains more than H variables, PROC SURVEYMEANS uses only the first H variables. Similarly, this data set must contain at least H observations.

If you do not specify the REPS= *method-option*, the number of replicates is assumed to be the number of observations in the *SAS-data-set*. If you specify the number of replicates—for example, REPS=*nreps*—the first *nreps* observations in the *SAS-data-set* are used to construct the replicates.

You can specify the PRINTH *method-option* to display the Hadamard matrix that PROC SURVEYMEANS uses to construct replicates for BRR.

NAIVEQVAR

requests that naive replication variance estimates be used to estimate the variances for quantiles. For more information, see the section "Replication Methods" on page 10154.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* in which to store the replicate weights that PROC SURVEYMEANS creates for BRR variance estimation. For information about replicate weights, see the section "Balanced Repeated Replication (BRR) Method" on page 10171. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10177.

This *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

PRINTH

displays the Hadamard matrix that PROC SURVEYMEANS uses to construct replicates for BRR variance estimation. When you provide the Hadamard matrix in the HADAMARD= *method-option*, PROC SURVEYMEANS displays only the rows and columns that are actually used to construct replicates. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 10171 and "Hadamard Matrix" on page 10175.

The PRINTH *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement because the procedure does not use a Hadamard matrix in this case.

REPS=*number*

specifies the *number* of replicates for BRR variance estimation. The value of *number* must be an integer greater than 1.

If you do not use the HADAMARD= *method-option* to provide a Hadamard matrix, the number of replicates should be greater than the number of strata and should be a multiple of 4. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10171. If PROC SURVEYMEANS cannot construct a Hadamard matrix for the REPS= value that you specify, the value is increased until a Hadamard matrix of that dimension can be constructed. Therefore, the actual number of replicates that PROC SURVEYMEANS uses might be larger than *number*.

If you use the HADAMARD= *method-option* to provide a Hadamard matrix, the value of *number* must not be greater than the number of rows in the Hadamard matrix. If you provide a Hadamard matrix and do not specify the REPS= *method-option*, the number of replicates is the number of rows in the Hadamard matrix.

If you do not specify the REPS= or the HADAMARD= *method-option* and do not use a REPWEIGHTS statement, the number of replicates is the smallest multiple of 4 that is greater than the number of strata.

If you use a REPWEIGHTS statement to provide replicate weights, PROC SURVEYMEANS does not use the REPS= *method-option*; the number of replicates is the number of REPWEIGHTS variables.

JACKKNIFE <(*method-options***)>**

JK <(*method-options***)>**

requests variance estimation by the delete-1 jackknife method. For more information, see the section "Jackknife Method" on page 10173. If you use a REPWEIGHTS statement to provide replicate weights, VARMETHOD=JACKKNIFE is the default variance estimation method.

The delete-1 jackknife method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you use a REPWEIGHTS statement to provide replicate weights.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Jackknife Method" on page 10173.

DFADJ

computes the degrees of freedom by using the number of nonempty strata for an analysis variable. The degrees of freedom for VARMETHOD=JACKKNIFE equals the number of clusters (or number of observations if there are no clusters) minus the number of strata (or one if there are no strata). By default, the number of strata is based on all valid observations in the data set. But if you specify this *method-option*, PROC SURVEYMEANS does not count any empty strata that are caused by all observations containing missing values for an analysis variable.

For more information, see the section "Degrees of Freedom" on page 10144. For more information about valid observations, see the section "Data and Sample Design Summary" on page 10180.

This *method-option* has no effect on categorical variables when you specify the MISSING option, which treats missing values as a valid nonmissing level.

This *method-option* cannot be used when you provide replicate weights in a REPWEIGHTS statement. When you use a REPWEIGHTS statement, the degrees of freedom equals the number of REPWEIGHTS variables (replicates) unless you specify an alternative value in the DF= option in the REPWEIGHTS statement.

NAIVEQVAR

requests that naive replication variance estimates be used to estimate the variances for quantiles. For more information, see the section "Replication Methods" on page 10154.

OUTJKCOEFS=*SAS-data-set*

names a *SAS-data-set* in which to store the jackknife coefficients. For information about jackknife coefficients, see the section "Jackknife Method" on page 10173. For information about the contents of the OUTJKCOEFS= data set, see the section "Jackknife Coefficients Output Data Set" on page 10177.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* in which to store the replicate weights that PROC SURVEYMEANS creates for jackknife variance estimation. For information about replicate weights, see the section "Jackknife Method" on page 10173. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10177.

This *method-option* is not available when you use a REPWEIGHTS statement to provide replicate weights unless you specify a POSTSTRATA statement.

TAYLOR

requests Taylor series variance estimation. This is the default method if you do not specify the VARMETHOD= option or a REPWEIGHTS statement. For more information, see the section "Taylor Series Method" on page 10142.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SURVEYMEANS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SURVEYMEANS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Note that using a BY statement provides completely separate analyses of the BY groups. It does not provide a statistically valid domain (subpopulation) analysis, where the total number of units in the subpopulation is not known with certainty. You should use the DOMAIN statement to obtain domain analysis. For more information about subpopulation analysis for sample survey data, see Cochran (1977).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* **;**

The CLASS statement names variables to be analyzed as categorical variables. For categorical variables, PROC SURVEYMEANS estimates the proportion in each category or level, instead of the overall mean. PROC SURVEYMEANS always analyzes character variables as categorical. If you want categorical analysis for a numeric variable, you must include that variable in the CLASS statement.

The CLASS *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the CLASS variables determine the categorical variable levels. Thus, you can use formats to group values into levels. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference* for more information.

When determining levels of a CLASS variable, an observation with missing values for this CLASS variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10138.

You can use multiple CLASS statements to specify categorical variables.

When you specify classification variables, you can use the SAS system option SUMSIZE= to limit (or to specify) the amount of memory that is available for data analysis. See the chapter on SAS system options in *SAS System Options: Reference* for a description of the SUMSIZE= option.

CLUSTER Statement

CLUSTER *variables* **;**

The CLUSTER statement names variables that identify the clusters in a clustered sample design. The combinations of categories of CLUSTER variables define the clusters in the sample. If there is a STRATA statement, clusters are nested within strata.

If you provide replicate weights for BRR or jackknife variance estimation with the REPWEIGHTS statement, you do not need to specify a CLUSTER statement.

If your sample design has clustering at multiple stages, you should identify only the first-stage clusters (primary sampling units (PSUs)), in the CLUSTER statement. See the section "Primary Sampling Units (PSUs)" on page 10140 for more information.

The CLUSTER *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the CLUSTER variables determine the CLUSTER variable levels. Thus, you can use formats to group values into levels. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference* for more information.

When determining levels of a CLUSTER variable, an observation with missing values for this CLUSTER variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10138.

You can use multiple CLUSTER statements to specify cluster variables. The procedure uses variables from all CLUSTER statements to create clusters.

DOMAIN Statement

DOMAIN *variables* < *variable*∗*variable variable*∗*variable*∗*variable . . .* > < / *options* > **;**

DOMAIN *variable* < ('*formatted-level-value*' . . .'*formatted-level-value*') > < *variable* < ('*formatted-level-value*' . . .'*formatted-level-value*') >∗*variable* < ('*formatted-level-value*' . . .'*formatted-level-value*') > > < / *options* > **;**

The DOMAIN statement requests analysis for domains (subpopulations) in addition to analysis for the entire study population. The DOMAIN statement names the *variables* that identify domains, which are called domain variables.

A domain variable can be either character or numeric. The procedure treats domain variables as categorical variables. If a *variable* appears by itself in a DOMAIN statement, each level of this variable determines a domain in the study population. If two or more *variables* are joined by asterisks (∗), then every possible combination of levels of these variables determines a domain. The procedure performs a descriptive analysis within each domain that is defined by the domain variables.

The formatted values of the domain variables determine the categorical variable levels. Thus, you can use formats to group values into levels. For more information, see the FORMAT procedure in *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

When the procedure is determining levels of a DOMAIN variable, an observation with missing values for this DOMAIN variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10138.

It is common practice to compute statistics for domains. Because formation of these domains might be unrelated to the sample design, the sample sizes for the domains are random variables. Use a DOMAIN statement to incorporate this variability into the variance estimation.

A DOMAIN statement is different from a BY statement. In a BY statement, you treat the sample sizes as fixed in each subpopulation, and you perform analysis within each BY group independently. For more information, see the section "Domain Analysis" on page 10140. Similarly, you should use a DOMAIN statement to perform a domain analysis over the entire data set. Creating a new data set from a single domain and analyzing that with PROC SURVEYMEANS yields inappropriate estimates of variance.

By default, the SURVEYMEANS procedure displays analyses for all levels of domains that are formed by the variables in a DOMAIN statement. Optionally, you can specify particular levels of each DOMAIN variable to be displayed by listing quoted *formatted-level-values* in parentheses after each variable name. You must enclose each *formatted-level-value* in single or double quotation marks. You can specify one or more levels of each variable; when you specify more than one level, separate the levels by a space or a comma. These examples illustrate the syntax:

```
domain Race*Gender(''Female'');
domain Race('White','Asian') Gender;
```

For example, **Race*Gender("Female")** requests that the procedure display analysis only for females within each race category, and **Race('White','Asian')** requests that the procedure display domain analysis only for people whose race is either white or Asian.

Specifying the same domain multiple times but with different levels for each corresponding domain variables is equivalent to specifying the union of different levels for the same variables. However, if you do not specify levels for a variable in a domain that is specified multiple times, only the specified levels are rendered. For example, the following two specifications together

```
domain Race('White')*Gender('Female');
domain Race('Asian')*Gender;
```

have the same effect as a single specification:

```
domain Race('White' 'Asian')*Gender('Female');
```

Also, the following specification

```
domain Race('White')*Gender Race('Asian')*Gender;
```

is equivalent to

```
domain Race('White' 'Asian')*Gender;
```

This syntax controls only the display of domain analysis results; it does not subset the data set, change the degrees of freedom, or otherwise affect the variance estimation.

You can specify the following *options* in the DOMAIN statement after a slash (/):

ADJUST=BON

requests a Bonferroni multiple comparison adjustment of the p-values, and adjusted confidence limits for the difference of domain means if the CLDIFF option is also specified. The adjusted p-values and confidence limits are displayed in addition to the unadjusted quantities. For a description of the adjustments, see the section "p-Value Adjustments" on page 7072 in Chapter 86, "The MULTTEST Procedure."

This option also invokes the DIFFMEANS option.

CLDIFF

requests t type confidence limits for each difference of domain means. You can specify the confidence level α in the ALPHA= option in the PROC SURVEYMEANS statement. By default, $\alpha = 0.05$, which produces 95% confidence limits. If you specify the ADJUST=BON option, then the adjusted confidence limits for Bonferroni multiplicity are also displayed.

This option also invokes the DIFFMEANS option.

COV

displays the estimated covariance matrix of domain means.

DFADJ

computes the degrees of freedom by using the number of non-empty strata for an analysis variable in a domain.

In a domain analysis, it is possible that some strata contain no sampling units for a specific domain. Or some strata in the domain might be empty due to missing values. By default, the procedure counts these empty strata when computing the degrees of freedom.

However, if you specify the DFADJ option, the procedure excludes any empty strata when computing the degrees of freedom. Prior to SAS 9.2, the procedure excluded empty strata by default.

The DFADJ option has no effect on categorical variables when you specify the MISSING option, which treats missing values as a valid nonmissing level.

For more information about valid observations, see the section "Data and Sample Design Summary" on page 10180. For more information about degrees of freedom, see the section "Degrees of Freedom" on page 10144.

DIFFMEANS | DIFF

requests the comparison of domain means for each continuous analysis variable that you specify in the VAR statement; this option does not provide comparisons for categorical analysis variables. If you specify this option, the SURVEYMEANS procedure provides differences between domain means for pairwise levels of a defined domain. You can specify which variable levels to include by listing the quoted *formatted-level-values* in parentheses after the variable names. By default, PROC SURVEYMEANS includes all pairwise domain levels.

Specifying DIFF is equivalent to specifying both DIFFRATIOS and DIFFMEANS.

For each pair of domain levels, the procedure displays the difference between domain means, the standard error of the difference, and the t test. You can also specify the CLDIFF option to request confidence limits for the differences and the ADJUST=BON option to request a Bonferroni multiple comparison adjustment of the p-values and confidence limits.

For more information, see the section "Difference of Domain Means" on page 10150.

DIFFRATIOS | DIFF

requests the comparison of domain ratios that you specify in the RATIO statement. If you specify this option, PROC SURVEYMEANS provides pairwise differences between domain ratios for levels of each domain that you define. You can specify which variable levels to include by listing the quoted *formatted-level-values* in parentheses after the variable names. By default, PROC SURVEYMEANS includes all pairs of domain levels.

For each pair of domain levels, the procedure displays the difference between domain ratios, the standard error of the difference, and the *t* test. You can also specify the CLDIFF option to request confidence limits for the differences and the ADJUST=BON option to request a Bonferroni multiple comparison adjustment of the *p*-values and confidence limits.

Specifying DIFF is equivalent to specifying both DIFFRATIOS and DIFFMEANS.

Domain ratio comparisons are available for replication variance estimation methods (which you can request by specifying VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or VARMETHOD=BOOTSTRAP or by including a REPWEIGHTS statement). They are not available for the Taylor series method (VARMETHOD=TAYLOR).

For more information, see the section "Difference of Domain Ratios" on page 10152.

POSTSTRATA Statement

POSTSTRATA *variables* / **PSTOTAL=***SAS-data-set* | **(***value-list***)** < *option* > **;**

POSTSTRATA *variables* / **PSPCT=***SAS-data-set* | **(***value-list***)** < *option* > **;**

The POSTSTRATA statement names *variables* that form the poststrata to adjust the sampling weights for analyzing the survey. The combinations of categories of *variables* define the poststrata in the sample.

The *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the *variables* determine the categorical levels. Thus, you can use formats to group values into levels. For more information, see the FORMAT procedure in *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

You must specify either poststratification totals (in a PSTOTAL= option) or poststratification proportions (in a PSPCT= option), but not both, after a slash (/).

PSTOTAL=*SAS-data-set* | **(***value-list***)**

POSTTOTAL=*SAS-data-set* | **(***value-list***)**

PSCONTROL=*SAS-data-set* | **(***value-list***)**

specifies poststratum totals, which the SURVEYMEANS procedure uses to compute weight adjustment for poststratification.

You can specify poststratification totals in either of the following ways:

SAS-data-set names a SAS data set that contains the poststratification variables and the poststratum totals. This data set is called the *poststratum total data set*.

A poststratum total data set must contain all the poststratification variables that are listed in the POSTSTRATA statement and all the variables listed in the BY statement. If there are formats associated with POSTSTRATA variables and the BY

variables, then the formats in the poststratum total data set for these variables must be consistent with those in the DATA= data set in the PROC SURVEYMEANS statement.

A poststratum total data set must have a variable named _PSTOTAL_ that contains the poststratum totals. The values of _PSTOTAL_ must be positive.

value-list specifies poststratum totals as a list of positive numbers when their corresponding poststratum levels are easy to identify. You must enclose this list in parentheses.

The ORDER=FORMATTED option in the PROC SURVEYMEANS statement is used to order the levels of poststratum levels.

The number of values in the *value-list* must equal the number of poststrata in the data. List the values in the *value-list* in the order of the corresponding poststratum level and separate them with blanks or commas.

PSPCT=*SAS-data-set* | **(***value-list***)**

POSTPCT=*SAS-data-set* | **(***value-list***)**

specifies the poststratum proportions, which the SURVEYMEANS procedure uses to compute weight adjustment for poststratification.

You can specify the poststratification proportions in one of the following ways:

SAS-data-set names a SAS data set that contains the poststratification variables and the poststratum proportions. This data set is called the *poststratum proportion data set*.

A poststratum proportion data set must contain all the poststratification variables that are listed in the POSTSTRATA statement and all the variables listed in the BY statement. If there are formats associated with the POSTSTRATA variables and the BY variables, then the formats in the poststratum proportion data set for these variables must be consistent with those in the DATA= data set in the PROC SURVEYMEANS statement.

A poststratum proportion data set must have a variable named _PSPCT_ that contains the poststratum proportions. The values of _PSPCT_ must be positive.

You can provide poststratum proportions either as positive decimal numbers between 0 and 1 for all poststrata or as positive percentages that must be less than 100 for all poststrata. If any of the proportion values is greater than 1, the procedure treats all proportions as percentages instead of decimal numbers.

value-list specifies the poststratum proportions as a list of positive numbers that correspond to poststrata. You must enclose this list in parentheses.

The ORDER=FORMATTED option in the PROC SURVEYMEANS statement is used to order the levels of poststratum levels.

The number of values in the *value-list* must equal the number of poststrata in the data. List the values in the *value-list* in the order of the corresponding poststratum level and separate them with blanks or commas.

If you provide the proportions as decimal numbers, then the sum of these values over all poststrata must be 1. If you provide the proportions as percentages, then the sum of these percentages over all poststrata must be 100.

You can also specify the following *option*:

OUTPSWGT=*SAS-data-set*

OUT=*SAS-data-set*

names a *SAS-data-set* to contain poststratification weights. For information about poststratification weights, see the section "Poststratification" on page 10162.

This option is ignored if you also specify an OUTWEIGHTS= *method-option* for VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or VARMETHOD=BOOTSTRAP in the PROC SURVEYMEANS statement. In this case, poststratification weights for the full sample and the replication weights adjusted for poststratification are stored in the OUTWEIGHTS= data set.

For more information about the contents of the OUTPSWGT= data set, see the section "Poststratification Weights Output Data Set" on page 10178. For more information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10177.

RATIO Statement

RATIO <'*label*'> *variables / variables* <*(options)*> **;**

The RATIO statement requests ratio analysis for means or proportions of analysis variables. A ratio statement names the variables whose means are used as numerators or denominators in a ratio. Variables that appear before the slash (/) are called *numerator variables* and are used as numerators. Variables that appear after the slash (/) are called *denominator variables* and are used as denominators. These *variables* can be any number of analysis variables, either continuous or categorical, except those named in the BY, CLUSTER, STRATA, DOMAIN, POSTSTRATA, REPWEIGHTS, and WEIGHT statements.

You can optionally specify a label for each RATIO statement to identify the ratios in the output. Labels must be enclosed in single quotes.

The computation of ratios depends on whether the numerator and denominator variables are continuous or categorical.

For continuous variables, ratios are calculated from the variable means. For example, for continuous variables X, Y, Z, and T, the following RATIO statement requests that the procedure analyze the ratios $\bar{x}/\bar{z}$, $\bar{x}/\bar{t}$, $\bar{y}/\bar{z}$, and $\bar{y}/\bar{t}$:

```
ratio x y / z t;
```

If a continuous variable appears as both a numerator and a denominator variable, the ratio of this variable to itself is ignored.

For categorical variables, ratios are calculated with the proportions for the categories. For example, if the categorical variable Gender has the values 'Male' and 'Female,' with the proportions p_m = Pr(Gender='Male') and p_f = Pr(Gender='Female'), and Y is a continuous variable, then the following RATIO statement requests that the procedure analyze the ratios p_m/p_f, p_f/p_m, $\bar{y}/p_m$, and $\bar{y}/p_f$:

```
ratio Gender y / Gender;
```

If a categorical variable appears as both a numerator and denominator variable, then the ratios of the proportions for all categories are computed, except the ratio of each category to itself.

You can have more than one RATIO statement. Each RATIO statement produces ratios independently by using its own numerator and denominator variables. Each RATIO statement also produces its own ratio analysis table. You can specify the following *options* in parentheses for each individual RATIO statement:

LowerCLBound=*min*

specifies *min* as the lower bound for the lower $100(1-\alpha)\%$ confidence limit of each ratio in the current RATIO statement. If the lower confidence limit is computed to be smaller than *min*, then the lower confidence limit is set to *min*. This truncation might result in less coverage than the specified $100(1-\alpha)\%$ confidence level. When a confidence limit is truncated, it is marked with an asterisk in the output.

When the ratio estimate is less than *min*, PROC SURVEYMEANS displays the lower confidence limit as missing.

By default, LowerCLBound=0 for a ratio of two proportions. For all other ratios, the confidence limits are not truncated.

UpperCLBound=*max*

specifies *max* as the upper bound for the upper $100(1-\alpha)\%$ confidence limit of each ratio in the current RATIO statement. If the upper confidence limit is computed to be larger than *max*, then the upper confidence limit is set to *max*. This truncation might result in less coverage than the specified $100(1-\alpha)\%$ confidence level. When a confidence limit is truncated, it is marked with an asterisk in the output.

When the ratio estimate is greater than *max*, PROC SURVEYMEANS displays the upper confidence limit as missing.

By default, the confidence limits are not truncated.

Available statistics for a ratio are as follows:

- N, number of observations used to compute the ratio
- NCLU, number of clusters
- SUMWGT, sum of weights
- RATIO, ratio
- STDERR, standard error of ratio
- VAR, variance of ratio
- T, t value of ratio
- PROBT, p-value of t
- DF, degrees of freedom of t
- CLM, two-sided confidence limits for ratio
- UCLM, one-sided upper confidence limit for ratio
- LCLM, one-sided lower confidence limit for ratio

The procedure calculates these statistics based on the *statistic-keywords* that you specify in the PROC SURVEYMEANS statement. If a *statistic-keyword* is not appropriate for a RATIO statement, that *statistic-keyword* is ignored for the ratios. If no valid statistics are requested for a RATIO statement, the procedure computes the ratio and its standard error by default.

When the means or proportions for the numerator and denominator variables in a ratio are calculated, an observation is excluded if it has a missing value for a continuous numerator or denominator variable. The procedure also excludes an observation with a missing value for a categorical numerator or denominator variable unless you specify the MISSING option.

When the denominator for a ratio is zero, then the value of the ratio is displayed as '–Infty', 'Infty', or a missing value, depending on whether the numerator is negative, positive, or zero, respectively, and the corresponding internal value is the special missing value '.M', the special missing value '.I', or the usual missing value, respectively.

REPWEIGHTS Statement

REPWEIGHTS *variables* < / *options* > **;**

The REPWEIGHTS statement names *variables* that provide replicate weights for bootstrap, BRR, or jackknife variance estimation, which you request with the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option, respectively, in the PROC SURVEYMEANS statement. For more information about the replication methods, see the section "Replication Methods for Variance Estimation" on page 10169.

Each REPWEIGHTS *variable* contains the weights for a single replicate, and the number of replicates equals the number of REPWEIGHTS variables. The REPWEIGHTS variables must be numeric, and the variable values must be nonnegative unless you specify the NEGATIVE option.

For more information about replicate weights that the SURVEYMEANS procedure creates, see the sections "Balanced Repeated Replication (BRR) Method" on page 10171 and "Jackknife Method" on page 10173.

If you provide replicate weights with a REPWEIGHTS statement, you do not need to specify a CLUSTER or STRATA statement. If you use a REPWEIGHTS statement and do not specify the VARMETHOD= option in the PROC SURVEYMEANS statement, the procedure uses VARMETHOD=JACKKNIFE by default.

If you specify a REPWEIGHTS statement but do not include a WEIGHT statement, the procedure uses the average of replicate weights of each observation as the observation's weight.

You can specify the following *options* in the REPWEIGHTS statement after a slash (/):

DF=*df*

specifies the degrees of freedom for the analysis. The value of *df* must be a positive number. By default, the value of *df* is the number of REPWEIGHTS variables.

If you know the number of PSUs and the number of strata from which the replicate weights are generated, you should specify the number of PSUs minus the number of strata as the degrees of freedom in this option. For more information, see the section "Degrees of Freedom" on page 10144.

JKCOEFS=*value* | **<(>***values***<)>** | *SAS-data-set*

specifies jackknife coefficients for the VARMETHOD=JACKKNIFE option in the PROC SURVEYMEANS statement. The jackknife coefficient values must be nonnegative numbers. For more information about jackknife coefficients, see the section "Jackknife Method" on page 10173.

You can provide jackknife coefficients by specifying one of the following forms:

value

specifies a single jackknife coefficient *value* to use for all replicates, where *value* must be a nonnegative number.

values

specifies a list of jackknife coefficients, where each value in the *values* is a nonnegative number that corresponds to a single replicate that is identified by a REPWEIGHTS variable. You can separate the values with blanks or commas, and you can optionally enclose the *values* in parentheses. The number of values in the *values* must equal the number of replicate weight variables that you specify in the REPWEIGHTS statement.

You must list the jackknife coefficient values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

SAS-data-set

names a *SAS-data-set* that contains the jackknife coefficients, where each coefficient value must be a nonnegative number. You must provide the jackknife coefficients in the data set variable named JKCoefficient. Each observation in this data set must correspond to a replicate that is identified by a REPWEIGHTS variable. The number of observations in the *SAS-data-set* must not be less than the number of REPWEIGHTS variables.

NEGATIVE | NEG

removes the restriction that the REPWEIGHTS variable values must be nonnegative. When you specify this option, PROC SURVEYMEANS accepts negative replicate weight values and includes them in the analysis.

By default, replicate weight values must be nonnegative; when PROC SURVEYMEANS encounters a negative replicate weight value, it displays an error message and does not perform the analysis unless you specify the NEGATIVE option.

REPCOEFS=*value* | **<(>***values***<)>** | *SAS-data-set*

specifies replicate coefficients for the VARMETHOD=BOOTSTRAP or VARMETHOD=JACKKNIFE option in the PROC SURVEYMEANS statement, where each coefficient corresponds to an individual replicate weight that is identified by a REPWEIGHTS variable. The replicate coefficient values must be nonnegative numbers.

You can provide replicate coefficients by specifying one of the following forms:

value

specifies a single replicate coefficient *value* to use for all replicates, where *value* must be a nonnegative number.

values

specifies a list of replicate coefficients, where each value in the *values* is a nonnegative number that corresponds to a single replicate that is identified by a REPWEIGHTS variable. You can separate the values with blanks or commas, and you can optionally enclose the *values* in parentheses. The number of values in the *values* must equal the number of replicate weight variables that you specify in the REPWEIGHTS statement.

You must list the replicate coefficient values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

SAS-data-set

names a *SAS-data-set* that contains the replicate coefficients, where each coefficient value must be a nonnegative number. You must provide the replicate coefficients in the data set variable named JKCoefficient. Each observation in this data set must correspond to a replicate that is identified by a REPWEIGHTS variable. The number of observations in the *SAS-data-set* must not be less than the number of REPWEIGHTS variables.

STRATA Statement

STRATA *variables* < / *option* > **;**

The STRATA statement specifies variables that form the strata in a stratified sample design. The combinations of categories of STRATA variables define the strata in the sample.

If your sample design has stratification at multiple stages, you should identify only the first-stage strata in the STRATA statement. See the section "Specification of Population Totals and Sampling Rates" on page 10139 for more information.

If you provide replicate weights for BRR or jackknife variance estimation with the REPWEIGHTS statement, you do not need to specify a STRATA statement.

The STRATA *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the STRATA variables determine the levels. Thus, you can use formats to group values into levels. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference* for more information.

When determining levels of a STRATA variable, an observation with missing values for this STRATA variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10138.

You can use multiple STRATA statements to specify stratum variables.

You can specify the following *option* in the STRATA statement after a slash (/):

LIST

displays a "Stratum Information" table, which includes values of the STRATA variables and the number of observations, number of clusters, population total, and sampling rate for each stratum. See the section "Stratum Information" on page 10180 for more details.

VAR Statement

VAR *variables* **;**

The VAR statement names the variables to be analyzed.

A *variable* in the VAR statement should not appear in any of the BY, CLUSTER, DOMAIN, POSTSTRATA, REPWEIGHTS, STRATA, and WEIGHT statements.

If you want a categorical analysis for a numeric variable, you must also name that variable in the CLASS statement. For categorical variables, PROC SURVEYMEANS estimates the proportion in each category or level, instead of the overall mean. Character variables are always analyzed as categorical variables. For more information, see the section "CLASS Statement" on page 10126.

When you specify a variable in a RATIO statement but not in a VAR statement, PROC SURVEYMEANS includes this variable as an analysis variable.

If you do not specify a VAR statement, but you have a RATIO statement, then PROC SURVEYMEANS analyzes only the variables in the RATIO statement.

If you do not specify a VAR statement nor a RATIO statement, then PROC SURVEYMEANS analyzes all variables in the DATA= input data set, except those named in the BY, CLUSTER, DOMAIN, POSTSTRATA, REPWEIGHTS, STRATA, and WEIGHT statements.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement names the variable that contains the sampling weights. This variable must be numeric, and the sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, then the procedure omits that observation from the analysis. See the section "Missing Values" on page 10138 for more information. If you specify more than one WEIGHT statement, the procedure uses only the first WEIGHT statement and ignores the rest.

If you do not specify a WEIGHT statement but provide replicate weights with a REPWEIGHTS statement, PROC SURVEYMEANS uses the average of replicate weights of each observation as the observation's weight.

If you do not specify a WEIGHT statement or a REPWEIGHTS statement, PROC SURVEYMEANS assigns all observations a weight of one.

Details: SURVEYMEANS Procedure

Missing Values

If you have missing values in your survey data for any reason, such as nonresponse, this can compromise the quality of your survey results. If the respondents are different from the nonrespondents with regard to a survey effect or outcome, then survey estimates might be biased and cannot accurately represent the survey population. There are a variety of techniques in sample design and survey operations that can reduce nonresponse. After data collection is complete, you can use imputation to replace missing values with acceptable values, and/or you can use sampling weight adjustments to compensate for nonresponse. You should complete this data preparation and adjustment before you analyze your data with PROC SURVEYMEANS. For more information, see Cochran (1977); Kalton and Kasprzyk (1986); Brick and Kalton (1996).

If an observation has a missing value or a nonpositive value for the WEIGHT variable, then that observation is excluded from the analysis.

An observation is also excluded from the analysis if it has a missing value for any design (STRATA, CLUSTER, or POSTSTRATA) variable, unless you specify the MISSING option in the PROC SURVEYMEANS statement. If you specify the MISSING option, the procedure treats missing values as a valid (nonmissing) category for all categorical variables. An observation is also excluded from a domain analysis if it has a missing value for a DOMAIN variable that defines a domain.

By default, when computing statistics for an analysis variable, PROC SURVEYMEANS omits observations with missing values for that analysis variable. The procedure computes statistics for each variable based only on observations that have nonmissing values for that variable. This treatment is based on the assumption that the missing values are missing completely at random (MCAR). However, this assumption is sometimes not true. For example, evidence from other surveys might suggest that observations with missing values are systematically different from observations without missing values. If you believe that missing values are not missing completely at random, then you can specify the NOMCAR option to let variance estimation include these observations with missing values in the analysis variables.

Whether or not you specify the NOMCAR option, PROC SURVEYMEANS always excludes observations that have missing or invalid values for the WEIGHT, STRATA, and CLUSTER variables unless you specify the MISSING option. Similarly, the procedure always excludes observations that have missing or invalid values for DOMAIN variables in domain analysis unless you specify the MISSING option.

When you specify the NOMCAR option, the procedure treats observations with and without missing values for analysis variables as two different domains, and it performs a domain analysis in the domain of nonmissing observations.

The procedure performs univariate analysis and analyzes each VAR variable separately. Thus, the number of missing observations might be different for different variables. You can specify the keyword NMISS in the PROC SURVEYMEANS statement to display the number of missing values for each analysis variable in the "Statistics" table.

When you specify a RATIO statement, the procedure excludes any observation that has a missing value for a continuous numerator or denominator variable. The procedure also excludes an observation with a missing value for a categorical numerator or denominator variable unless you specify the MISSING option.

If you use a REPWEIGHTS statement, all REPWEIGHTS variables must contain nonmissing values.

Survey Data Analysis

Specification of Population Totals and Sampling Rates

To include a finite population correction (*fpc*) in Taylor series or bootstrap variance estimation, you can input either the sampling rate or the population total by using the RATE= or TOTAL= option, respectively, in the PROC SURVEYMEANS statement. (You cannot specify both of these options in the same PROC SURVEYMEANS statement.) The RATE= and TOTAL= options apply only to Taylor series or bootstrap variance estimation. The procedure does not use a finite population correction for BRR or jackknife variance estimation.

If you do not specify the RATE= or TOTAL= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. For fairly small sampling fractions, it is appropriate to ignore this correction. For more information, see Cochran (1977) and Kish (1965).

If your design has multiple stages of selection and you are specifying the RATE= option, you should input the first-stage sampling rate, which is the ratio of the number of primary sampling units (PSUs) in the sample to the total number of PSUs in the study population. If you are specifying the TOTAL= option for a multistage design, you should input the total number of PSUs in the study population. For more information, see the section "Primary Sampling Units (PSUs)" on page 10140.

For a nonstratified sample design, or for a stratified sample design that has the same sampling rate or the same population total in all strata, you can use the RATE=*value* or TOTAL=*value* option. If your sample design is stratified with different sampling rates or population totals in different strata, use the RATE=*SAS-data-set* or TOTAL=*SAS-data-set* option to name a SAS data set that contains the stratum sampling rates or totals. This data set is called a *secondary data set*, as opposed to the *primary data set* that you specify in the DATA= option.

The secondary data set must contain all the stratification variables that are listed in the STRATA statement and all the variables that are listed in the BY statement. If there are formats associated with the STRATA variables and the BY variables, then the formats must be consistent in the primary and the secondary data sets. If you specify the TOTAL=*SAS-data-set* option, the secondary data set must have a variable named _TOTAL_ that contains the stratum population totals. Or if you specify the RATE=*SAS-data-set* option, the secondary data set must have a variable named _RATE_ that contains the stratum sampling rates. If the secondary data set contains more than one observation for any one stratum, then the procedure uses the first value of _TOTAL_ or _RATE_ for that stratum and ignores the rest.

The *value* in the RATE= option or the values of _RATE_ in the secondary data set must be nonnegative numbers. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC SURVEYMEANS converts that number to a proportion. The procedure treats the value 1 as 100% instead of 1%.

If you specify the TOTAL=*value* option, *value* must not be less than the sample size. If you provide stratum population totals in a secondary data set, these values must not be less than the corresponding stratum sample sizes.

Primary Sampling Units (PSUs)

When you have clusters, or primary sampling units (PSUs), in your sample design, the procedure estimates variance from the variation among PSUs when the Taylor series variance method is used. For more information, see the section "Taylor Series Method" on page 10142.

BRR or jackknife variance estimation methods draw multiple replicates (also called subsamples) from the full sample by following a specific resampling scheme. These subsamples are constructed by deleting PSUs from the full sample.

The bootstrap variance estimation method repeatedly draws a simple random sample with replacement of primary sampling units (PSUs) from the full sample to form each replicate.

If you use a REPWEIGHTS statement to provide replicate weights for BRR, jackknife, bootstrap variance estimation, you do not need to specify a CLUSTER statement. Otherwise, you should specify a CLUSTER statement whenever your design includes clustering at the first stage of sampling. If you do not specify a CLUSTER statement, then PROC SURVEYMEANS treats each observation as a PSU.

Domain Analysis

It is common practice to compute statistics for domains (subpopulations), in addition to computing statistics for the entire study population. Analysis for domains that uses the entire sample is called *domain analysis* (also called subgroup analysis, subpopulation analysis, or subdomain analysis). The formation of these subpopulations of interest might be unrelated to the sample design. Therefore, the sample sizes for the subpopulations might actually be random variables.

Use a DOMAIN statement to incorporate this variability into the variance estimation. Note that using a BY statement provides completely separate analyses of the BY groups. It does not provide a statistically valid subpopulation or domain analysis, where the total number of units in the subpopulation is not known with certainty.

For more detailed information about domain analysis, see Kish (1965).

Statistical Computations

The SURVEYMEANS procedure uses the Taylor series (linearization) method or replication (resampling) methods to estimate sampling errors of estimators based on complex sample designs. For more information, see Fuller (2009); Wolter (2007); Lohr (2022); Kalton (1983); Hidiroglou, Fuller, and Hickman (1980); Fuller et al. (1989); Lee, Forthofer, and Lorimor (1989); Cochran (1977); Kish (1965); Hansen, Hurwitz, and Madow (1953); Rust (1985); Dippo, Fay, and Morganstein (1984); Rao and Shao (1999); Rao, Wu, and Yue (1992); Rao and Shao (1996). You can use the VARMETHOD= option to specify a variance estimation method to use. By default, the Taylor series method is used.

The Taylor series method obtains a linear approximation for the estimator and then uses the variance estimate for this approximation to estimate the variance of the estimate itself (Woodruff 1971; Fuller 1975). When there are clusters, or PSUs, in the sample design, the procedure estimates variance from the variation among PSUs. When the design is stratified, the procedure pools stratum variance estimates to compute the overall variance estimate. For t tests of the estimates, the degrees of freedom equals the number of clusters minus the number of strata in the sample design.

For a multistage sample design, the Taylor series estimation depends only on the first stage of the sample design. Therefore, the required input includes only first-stage cluster (PSU) and first-stage stratum identification. You do not need to input design information about any additional stages of sampling. This variance estimation method assumes that the first-stage sampling fraction is small, or that the first-stage sample is drawn with replacement, as it often is in practice.

Quite often in complex surveys, respondents have unequal weights, which reflect unequal selection probabilities and adjustments for nonresponse. In such surveys, the appropriate sampling weights must be used to obtain valid estimates for the study population.

However, replication methods have recently gained popularity for estimating variances in complex survey data analysis. One reason for this popularity is the relative simplicity of replication-based estimates, especially for nonlinear estimators; another is that modern computational capacity has made replication methods feasible for practical survey analysis.

Replication methods draw multiple replicates (also called subsamples) from a full sample according to a specific resampling scheme. The most commonly used resampling schemes are the *balanced repeated replication* (BRR) method, the *jackknife* method, and the *bootstrap* method. For each replicate, the original weights are modified for the PSUs in the replicates to create replicate weights. The population parameters of interest are estimated by using the replicate weights for each replicate. Then the variances of parameters of interest are estimated by the variability among the estimates derived from these replicates. You can use a REPWEIGHTS statement to provide your own replicate weights for variance estimation. For more information about using replication methods to analyze sample survey data, see the section "Replication Methods for Variance Estimation" on page 10169.

Definitions and Notation

For a stratified clustered sample design, together with the sampling weights, the sample can be represented by an $n \times (P+1)$ matrix

$$\begin{aligned}(\mathbf{w}, \mathbf{Y}) &= \left(w_{hij}, \mathbf{y}_{hij}\right) \\ &= \left(w_{hij}, y_{hij}^{(1)}, y_{hij}^{(2)}, \ldots, y_{hij}^{(P)}\right)\end{aligned}$$

where

- $h = 1, 2, \ldots, H$ is the stratum index
- $i = 1, 2, \ldots, n_h$ is the cluster index within stratum h
- $j = 1, 2, \ldots, m_{hi}$ is the unit index within cluster i of stratum h
- $p = 1, 2, \ldots, P$ is the analysis variable number, with a total of P variables
- $n = \sum_{h=1}^{H} \sum_{i=1}^{n_h} m_{hi}$ is the total number of observations in the sample
- w_{hij} denotes the sampling weight for unit j in cluster i of stratum h
- $\mathbf{y}_{hij} = \left(y_{hij}^{(1)}, y_{hij}^{(2)}, \ldots, y_{hij}^{(P)}\right)$ are the observed values of the analysis variables for unit j in cluster i of stratum h, including both the values of numerical variables and the values of indicator variables for levels of categorical variables.

For a categorical variable C, let l denote the number of levels of C, and denote the level values as $c_1, c_2, \ldots, c_l$. Let $y^{(q)}$ ($q \in \{1, 2, \ldots, P\}$) be an indicator variable for the category $C = c_k$ ($k = 1, 2, \ldots, l$) with the observed value in unit j in cluster i of stratum h:

$$y_{hij}^{(q)} = I_{\{C=c_k\}}(h, i, j) = \begin{cases} 1 & \text{if } C_{hij} = c_k \\ 0 & \text{otherwise} \end{cases}$$

Note that the indicator variable $y_{hij}^{(q)}$ is set to missing when C_{hij} is missing. Therefore, the total number of analysis variables, P, is the total number of numerical variables plus the total number of levels of all categorical variables.

The sampling rate f_h for stratum h, which is used in Taylor series and bootstrap variance estimation, is the fraction of first-stage units (PSUs) that are selected for the sample. You can use the TOTAL= or RATE= option to input population totals or sampling rates. For more information, see the section "Specification of Population Totals and Sampling Rates" on page 10139. If you input stratum totals, PROC SURVEYMEANS computes f_h as the ratio of the stratum sample size to the stratum total. If you input stratum sampling rates, PROC SURVEYMEANS uses these values directly for f_h. If you do not specify the TOTAL= or RATE= option, then the procedure assumes that the stratum sampling rates f_h are negligible, and a finite population correction is not used when computing variances. Replication methods that are specified by the VARMETHOD=BRR or VARMETHOD=JACKKNIFE option do not use this finite population correction f_h.

Mean

When you specify the keyword MEAN, the procedure computes the estimate of the mean (mean per element) from the survey data. Also, the procedure computes the mean by default if you do not specify any *statistic-keywords* in the PROC SURVEYMEANS statement.

PROC SURVEYMEANS computes the estimate of the mean as

$$\hat{\bar{Y}} = \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}\ y_{hij} \right) / \ w_{...}$$

where

$$w_{...} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}$$

is the sum of the weights over all observations in the sample.

Variance and Standard Error of the Mean

When you specify the keyword STDERR, the procedure computes the standard error of the mean. Also, the procedure computes the standard error by default if you specify the keyword MEAN, or if you do not specify any *statistic-keywords* in the PROC SURVEYMEANS statement. The keyword VAR requests the variance of the mean.

Taylor Series Method

When you use VARMETHOD=TAYLOR, or by default if you do not specify the VARMETHOD= option, PROC SURVEYMEANS uses the Taylor series method to estimate the variance of the mean $\hat{\bar{Y}}$. The

procedure computes the estimated variance as

$$\hat{V}(\hat{\bar{Y}}) = \sum_{h=1}^{H} \hat{V}_h(\hat{\bar{Y}})$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h(\hat{\bar{Y}}) &= \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (e_{hi\cdot} - \bar{e}_{h\cdot\cdot})^2 \\
e_{hi\cdot} &= \left(\sum_{j=1}^{m_{hi}} w_{hij}\ (y_{hij} - \hat{\bar{Y}}) \right) / \ w_{\cdots} \\
\bar{e}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} e_{hi\cdot} \right) / \ n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h(\hat{\bar{Y}}) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

Replication Methods

When you specify the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option, the procedure computes the variance $\hat{V}(\hat{\bar{Y}})$ with replication methods by using the variability among replicate estimates to estimate the overall variance. See the section "Replication Methods for Variance Estimation" on page 10169 for more details.

Standard Error

The standard error of the mean is the square root of the estimated variance.

$$\text{StdErr}(\hat{\bar{Y}}) = \sqrt{\hat{V}(\hat{\bar{Y}})}$$

t Test for the Mean

If you specify the keyword T, PROC SURVEYMEANS computes the *t*-value for testing that the population mean equals zero, $H_0\colon \bar{Y} = 0$. The test statistic equals

$$t(\hat{\bar{Y}}) = \hat{\bar{Y}} \ / \ \text{StdErr}(\hat{\bar{Y}})$$

The two-sided *p*-value for this test is

$$\text{Prob}(\ |T| > |t(\hat{\bar{Y}})|\ )$$

where T is a random variable with the t distribution with df degrees of freedom.

Degrees of Freedom

PROC SURVEYMEANS computes the degrees of freedom *df* to obtain the *t* distribution's percentile to construct the $100(1-\alpha)\%$ confidence limits for means, proportions, totals, ratios, and other statistics, and for hypothesis testing. The degrees of freedom computation depends on the variance estimation method that you request. Missing values can affect the degrees of freedom computation. For more information, see the section "Missing Values" on page 10138.

Taylor Series Variance Estimation

For the Taylor series method, PROC SURVEYMEANS calculates the degrees of freedom for the *t* test as the number of clusters minus the number of strata. If there are no clusters, then the degrees of freedom equals the number of observations minus the number of strata. If the design is not stratified, then the degrees of freedom equals the number of PSUs minus one.

If all observations in a stratum are excluded from the analysis due to missing values, then that stratum is called an *empty stratum*. Empty strata are not counted in the total number of strata for the table. Similarly, empty clusters and missing observations are not included in the total counts of cluster and observations that are used to compute the degrees of freedom for the analysis.

If you specify the MISSING option, missing values are treated as valid nonmissing levels for a categorical variable and are included in computing degrees of freedom. If you specify the NOMCAR option for Taylor series variance estimation, observations with missing values for an analysis variable are included in computing degrees of freedom.

Replication Variance Estimation with Replicate Weights

When a REPWEIGHTS statement is specified, PROC SURVEYMEANS computes the degrees of freedom as the number of REPWEIGHTS variables, unless you specify an alternative in the DF= option in a REPWEIGHTS statement.

Replication Variance Estimation without Replicate Weights

For replication variance estimation when no REPWEIGHT statement is specified, PROC SURVEYMEANS by default computes the degrees of freedom by using all valid observations in the input data set. A valid observation is an observation that has a positive value of the variable that you specify in a WEIGHT statement. A valid observation also must have nonmissing values for the variables that are specified in STRATA, CLUSTER, and POSTSTRATA statements unless you specify the MISSING option. For more information about valid observations, see the section "Data and Sample Design Summary" on page 10180.

For VARMETHOD=BRR variance estimation (including Fay's method) without a REPWEIGHTS statement, the degrees of freedom equals the number of strata.

For the VARMETHOD=JACKKNIFE and the VARMETHOD=BOOTSTRAP variance estimation methods without a REPWEIGHTS statement, PROC SURVEYMEANS calculates the degrees of freedom the same as it does when VARMETHOD=TAYLOR.

When you specify the DFADJ option, the procedure computes the degrees of freedom from the number of nonmissing strata and clusters for an analysis variable. This excludes any empty strata or clusters that occur when observations that have missing values of an analysis variable are removed.

The procedure displays the degrees of freedom for the *t* test if you specify the keyword DF in the PROC SURVEYMEANS statement.

Confidence Limits for the Mean

If you specify the keyword CLM, the procedure computes two-sided confidence limits for the mean. Also, the procedure includes the confidence limits by default if you do not specify any *statistic-keywords* in the PROC SURVEYMEANS statement.

The confidence coefficient is determined by the value of the ALPHA= option, which by default equals 0.05 and produces 95% confidence limits. The confidence limits are computed as

$$\hat{\bar{Y}} \pm \text{StdErr}(\hat{\bar{Y}})\ t_{df,\ \alpha/2}$$

where $\hat{\bar{Y}}$ is the estimate of the mean, $\text{StdErr}(\hat{\bar{Y}})$ is the standard error of the mean, and $t_{df,\ \alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df calculated as in the section "*t* Test for the Mean" on page 10143.

If you specify the keyword UCLM, the procedure computes the one-sided upper $100(1-\alpha)\%$ confidence limit for the mean:

$$\hat{\bar{Y}} + \text{StdErr}(\hat{\bar{Y}})\ t_{df,\ \alpha}$$

If you specify the keyword LCLM, the procedure computes the one-sided lower $100(1-\alpha)\%$ confidence limit for the mean:

$$\hat{\bar{Y}} - \text{StdErr}(\hat{\bar{Y}})\ t_{df,\ \alpha}$$

Coefficient of Variation

If you specify the keyword CV, PROC SURVEYMEANS computes the coefficient of variation, which is the ratio of the standard error of the mean to the estimated mean:

$$\text{cv}(\bar{Y}) = \text{StdErr}(\hat{\bar{Y}})\ /\ \hat{\bar{Y}}$$

If you specify the keyword CVSUM, PROC SURVEYMEANS computes the coefficient of variation for the estimated total, which is the ratio of the standard deviation of the sum to the estimated total:

$$\text{cv}(Y) = \text{Std}(\hat{Y})\ /\ \hat{Y}$$

Proportions

If you specify the keyword MEAN for a categorical variable, PROC SURVEYMEANS estimates the proportion, or relative frequency, for each level of the categorical variable. If you do not specify any *statistic-keywords* in the PROC SURVEYMEANS statement, the procedure estimates the proportions for levels of the categorical variables, together with their standard errors and confidence limits.

The procedure estimates the proportion in level c_k for variable C as

$$\hat{p} = \frac{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}}\ w_{hij}\ y_{hij}^{(q)}}{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}}\ w_{hij}}$$

where $y_{hij}^{(q)}$ is the value of the indicator function for level $C = c_k$, defined in the section "Definitions and Notation" on page 10141, and $y_{hij}^{(q)}$ equals 1 if the observed value of variable C equals c_k, and $y_{hij}^{(q)}$ equals 0 otherwise. Since the proportion estimator is actually an estimator of the mean for an indicator variable, the procedure computes its variance and standard error according to the method outlined in the section "Variance and Standard Error of the Mean" on page 10142. Similarly, the procedure computes confidence limits for proportions as in the section "Confidence Limits for the Mean" on page 10145.

Total

If you specify the keyword SUM, the procedure computes the estimate of the population total from the survey data. The estimate of the total is the weighted sum over the sample:

$$\hat{Y} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}\ y_{hij}$$

For a categorical variable level, $\hat{Y}$ estimates its total frequency in the population.

Variance and Standard Deviation of the Total

When you specify the keyword STD or the keyword SUM, the procedure estimates the standard deviation of the total. The keyword VARSUM requests the variance of the total.

Taylor Series Method

When you use VARMETHOD=TAYLOR, or by default, PROC SURVEYMEANS uses the Taylor series method to estimate the variance of the total as

$$\hat{V}(\hat{Y}) = \sum_{h=1}^{H} \hat{V}_h(\hat{Y})$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h(\hat{Y}) &= \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (y_{hi\cdot} - \bar{y}_{h\cdot\cdot})^2 \\
y_{hi\cdot} &= \sum_{j=1}^{m_{hi}} w_{hij}\ y_{hij} \\
\bar{y}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} y_{hi\cdot} \right) / n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h(\hat{Y}) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

Replication Methods

When you specify the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option, the procedure computes the variance $\hat{V}(\hat{Y})$ by using replication methods and measuring the variability among the estimates derived from these replicates. For more information, see the section "Replication Methods for Variance Estimation" on page 10169.

Standard Deviation

The standard deviation of the total equals

$$\mathrm{Std}(\hat{Y}) = \sqrt{\hat{V}(\hat{Y})}$$

Confidence Limits for the Total

If you specify the keyword CLSUM, the procedure computes confidence limits for the total. The confidence coefficient is determined by the value of the ALPHA= option, which by default equals 0.05 and produces 95% confidence limits. The confidence limits are computed as

$$\hat{Y} \pm \mathrm{Std}(\hat{Y})\, t_{df,\,\alpha/2}$$

where $\hat{Y}$ is the estimate of the total, $\mathrm{Std}(\hat{Y})$ is the estimated standard deviation, and $t_{df,\,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df calculated as described in the section "*t* Test for the Mean" on page 10143.

If you specify the keyword UCLSUM, the procedure computes the one-sided upper $100(1-\alpha)\%$ confidence limit for the sum:

$$\hat{Y} + \mathrm{Std}(\hat{Y})\, t_{df,\,\alpha}$$

If you specify the keyword LCLSUM, the procedure computes the one-sided lower $100(1-\alpha)\%$ confidence limit for the sum:

$$\hat{Y} - \mathrm{Std}(\hat{Y})\, t_{df,\,\alpha}$$

Design Effect

If you specify the keyword DEFF in the PROC SURVEYMEANS statement, the procedure calculates the design effects for the mean. The design effect is the ratio of the variance computed under the sample design to the variance computed under the assumption of simple random sampling:

$$\mathrm{DEFF} = \frac{\text{variance under the sample design}}{\text{variance under simple random sampling}}$$

For more information, see Kish (1965, p. 258).

In the numerator, PROC SURVEYMEANS computes the variance of the mean according to the method outlined in the section "Variance and Standard Error of the Mean" on page 10142. And the denominator is computed under the assumption that the sample design is simple random sampling, with no stratification and no clustering.

For Taylor series or bootstrap variance estimation, PROC SURVEYMEANS computes the overall sampling fraction f_{SRS} in the simple random sampling variance by using the value of the RATE= or TOTAL= option.

If you do not specify either of these options, PROC SURVEYMEANS assumes that the value of f_{SRS} is negligible and does not use a finite population correction in the analysis, as described in the section "Specification of Population Totals and Sampling Rates" on page 10139.

If you specify RATE=*value*, PROC SURVEYMEANS uses this *value* as the overall sampling fraction f_{SRS}. If you specify TOTAL=*value*, PROC SURVEYMEANS computes f_{SRS} as the ratio of the number of PSUs in the sample to the specified total.

If you specify stratum sampling rates by using the RATE=*SAS-data-set* option, then PROC SURVEYMEANS computes stratum totals based on these stratum sampling rates and the number of sample PSUs in each stratum. The procedure sums the stratum totals to form the overall total, and it computes f_{SRS} as the ratio of the number of sample PSUs to the overall total. Alternatively, if you specify stratum totals by using the TOTAL=*SAS-data-set* option, then PROC SURVEYMEANS sums these totals to compute the overall total. The overall sampling fraction f_{SRS} is then computed as the ratio of the number of sample PSUs to the overall total.

Ratio

When you use a RATIO statement, the procedure produces statistics requested by the *statistic-keywords* in the PROC SURVEYMEANS statement.

Suppose that you want to calculate the ratio of variable Y to variable X. Let x_{hij} be the value of variable X for the jth member in cluster i in the hth stratum.

The ratio of Y to X is

$$\hat{R} = \frac{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}\ y_{hij}\ /\ \sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}}{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}\ x_{hij}\ /\ \sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}}$$

PROC SURVEYMEANS uses the Taylor series method to estimate the variance of the ratio $\hat{R}$ as

$$\hat{V}(\hat{R}) = \sum_{h=1}^{H} \hat{V}_h(\hat{R})$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h(\hat{R}) &= \frac{n_h(1-f_h)}{n_h-1}\sum_{i=1}^{n_h}(g_{hi\cdot} - \bar{g}_{h\cdot\cdot})^2 \\
g_{hi\cdot} &= \frac{\sum_{j=1}^{m_{hi}} w_{hij}\ (y_{hij} - x_{hij}\hat{R})}{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w_{hij}\ x_{hij}} \\
\bar{g}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} g_{hi\cdot}\right) / \ n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h(\hat{R}) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

The standard error of the ratio is the square root of the estimated variance:

$$\text{StdErr}(\hat{R}) = \sqrt{\hat{V}(\hat{R})}$$

When the denominator for a ratio is zero, then the value of the ratio is displayed as '–Infty', 'Infty', or a missing value, depending on whether the numerator is negative, positive, or zero, respectively; and the corresponding internal value is the special missing value '.M', the special missing value '.I', or the usual missing value, respectively.

Domain Statistics

When you use a DOMAIN statement to request a domain analysis, the procedure computes the requested statistics for each domain level.

For a domain D, let I_D be the corresponding indicator variable:

$$I_D(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to } D \\ 0 & \text{otherwise} \end{cases}$$

Let

$$z_{hij} \quad = \quad y_{hij} I_D(h,i,j) \quad = \begin{cases} y_{hij} & \text{if observation } (h,i,j) \text{ belongs to } D \\ 0 & \text{otherwise} \end{cases}$$

Let

$$v_{hij} = w_{hij} I_D(h,i,j) = \begin{cases} w_{hij} & \text{if observation } (h,i,j) \text{belongs to} D \\ 0 & \text{otherwise} \end{cases}$$

The requested statistics for variable y in domain D are computed by using the new weights v.

Note that z_{hij} is set to missing if y_{hij} represents a level of a categorical variable and y_{hij} is missing.

Domain Mean

The estimated mean of Y in the domain D is

$$\hat{\bar{Y}}_D = \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} v_{hij} \; y_{hij} \right) / \; v_{...}$$

where

$$v_{...} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} v_{hij}$$

When you use the Taylor series method, the variance of $\hat{\bar{Y}}_D$ is estimated by

$$\hat{V}(\hat{\bar{Y}}_D) = \sum_{h=1}^{H} \hat{V}_h(\hat{\bar{Y}}_D)$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h(\hat{\bar{Y}}_D) &= \frac{n_h(1-f_h)}{n_h-1}\sum_{i=1}^{n_h}(r_{hi\cdot}-\bar{r}_{h\cdot\cdot})^2 \\
r_{hi\cdot} &= \left(\sum_{j=1}^{m_{hi}} v_{hij}\ (y_{hij}-\hat{\bar{Y}}_D)\right) / \ v_{\cdots} \\
\bar{r}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} r_{hi\cdot}\right) / \ n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h(\hat{\bar{Y}}_D) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

If you use replication methods to estimate the variance (by specifying the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option), PROC SURVEYMEANS estimates the variance of $\hat{\bar{Y}}_D$ by using the variability among replicate estimates to estimate the overall variance. For more information, see the section "Replication Methods for Variance Estimation" on page 10169.

Difference of Domain Means

If you specify the DIFFMEANS option in a DOMAIN statement, PROC SURVEYMEANS compares the means of a continuous analysis variable at different domain levels.

Let D be a domain definition that is specified in a DOMAIN statement. Let $D_1, D_2, \ldots, D_r$ be the r levels of D, and let the corresponding indicator variables be

$$I_{D_k}(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to } D_k \\ 0 & \text{otherwise} \end{cases}$$

for $k = 1, \ldots, r$.

For variable Y, the difference between the means for domain levels D_{k1} and D_{k2} $(1 \le k1 \ne k2 \le r)$ can be expressed as

$$\Delta(Y, D, k1, k2) = \hat{\bar{Y}}_{D_{k1}} - \hat{\bar{Y}}_{D_{k2}}$$

The estimated variance for this difference is

$$\hat{V}(\Delta(Y, D, k1, k2)) = \hat{V}(\hat{\bar{Y}}_{D_{k1}}) + \hat{V}(\hat{\bar{Y}}_{D_{k2}}) - 2\hat{\mathrm{Cov}}(\hat{\bar{Y}}_{D_{k1}}, \hat{\bar{Y}}_{D_{k2}})$$

where the estimated variances $\hat{V}(\hat{\bar{Y}}_{D_{k1}})$ and $\hat{V}(\hat{\bar{Y}}_{D_{k2}})$ for means at corresponding domain levels (in addition to the covariance between these two domain means) are described as in the section "Domain Mean" on page 10149.

For variable Y, PROC SURVEYMEANS computes the t statistics to test the significance of the difference of the domain means between two levels, D_{k1} and D_{k2}, as the following ratio with df degrees of freedom:

$$\frac{\Delta(Y, D, k1, k2)}{\sqrt{\hat{V}(\Delta(Y, D, k1, k2))}}$$

For more information about *df*, see the section "Degrees of Freedom" on page 10144. For more information about computing the domain means and the variance and covariance of domain means with poststratification, see the section "Variance of the Domain Mean and Sum" on page 10165. The corresponding t statistics are then computed accordingly.

Domain Total

The estimated total in domain D is

$$\hat{Y}_D = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} v_{hij}\ y_{hij}$$

and its estimated variance is

$$\hat{V}(\hat{Y}_D) = \sum_{h=1}^{H} \hat{V}_h(\hat{Y}_D)$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h(\hat{Y}_D) &= \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (z_{hi\cdot} - \bar{z}_{h\cdot\cdot})^2 \\
z_{hi\cdot} &= \sum_{j=1}^{m_{hi}} v_{hij}\ z_{hij} \\
\bar{z}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} z_{hi\cdot} \right) \ / \ n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h(\hat{Y}_D) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

Domain Ratio

The estimated ratio of Y to X in domain D is

$$\hat{R}_D = \frac{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}}\ v_{hij}\ y_{hij}}{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}}\ v_{hij}\ x_{hij}}$$

and its estimated variance is

$$\hat{V}(\hat{R}_D) = \sum_{h=1}^{H} \hat{V}_h(\hat{R}_D)$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h(\hat{R}_D) &= \frac{n_h(1-f_h)}{n_h-1}\sum_{i=1}^{n_h}(g_{hi\cdot}-\bar{g}_{h\cdot\cdot})^2 \\
g_{hi\cdot} &= \frac{\sum_{j=1}^{m_{hi}} v_{hij}\,(y_{hij}-x_{hij}\hat{R}_D)}{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} v_{hij}\,x_{hij}} \\
\bar{g}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} g_{hi\cdot}\right)/\,n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h(\hat{R}_D) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1,2,\ldots,H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

For domain analysis with poststratification, see the section "Poststratification" on page 10162. For quantile estimation in a domain, see the section "Domain Quantile" on page 10157. For quantile estimation in a domain with poststratification, see the section "Domain Quantile Estimation with Poststratification" on page 10159.

Difference of Domain Ratios

If you specify the DIFFRATIOS option in a DOMAIN statement and also specify a RATIO statement, PROC SURVEYMEANS compares each requested ratio at different domain levels.

Let D be a domain definition that you specify in a DOMAIN statement. Let $D_1, D_2, \ldots, D_r$ be the r levels of D, and let the corresponding indicator variables be

$$I_{D_k}(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to } D_k \\ 0 & \text{otherwise} \end{cases}$$

for $k = 1,\ldots,r$.

For the ratio of Y to X, the difference between domain levels D_{k1} and D_{k2} $(1 \le k1 \ne k2 \le r)$ in domain D can be expressed as

$$\Delta(Y/X, D, k1, k2) = \hat{R}_{D_{k1}} - \hat{R}_{D_{k2}}$$

The estimated variance for this difference, $\hat{V}(\Delta(Y/X, D, k1, k2))$, is computed by using replication methods and measuring the variability among the estimates that are derived from replicates. For more information, see the section "Replication Methods for Variance Estimation" on page 10169.

For the ratio of Y to X, PROC SURVEYMEANS computes the t statistic to test the significance of the difference of the domain ratios between two levels, D_{k1} and D_{k2}, as

$$\frac{\Delta(Y/X, D, k1, k2)}{\sqrt{\hat{V}(\Delta(Y/X, D, k1, k2))}}$$

Under the null hypothesis of no difference between the two levels, this statistic approximately follows a t distribution with df degrees of freedom. For more information about df, see the section "Degrees of Freedom" on page 10144.

Quantiles

Let Y be the variable of interest in a complex survey. Denote $F(t) = \Pr(Y \le t)$ as the cumulative distribution function of Y. For $0 < p < 1$, the pth quantile of the population cumulative distribution function is

$$Q(p) = \inf\{y : F(y) \ge p\}$$

Estimate of Quantile

Let $\{y_{hij}, w_{hij}\}$ be the observed values for variable Y that are associated with sampling weights, where (h, i, j) are the stratum index, cluster index, and member index, respectively, as shown in the section "Definitions and Notation" on page 10141. Let $y_{(1)} < y_{(2)} < \cdots < y_{(n)}$ denote the sample order statistics for variable Y.

An estimate of quantile $Q(p)$ is

$$\hat{Q}(p) = \begin{cases} y_{(1)} & \text{if } p < \hat{F}(y_{(1)}) \\ y_{(k)} + \dfrac{p - \hat{F}(y_{(k)})}{\hat{F}(y_{(k+1)}) - \hat{F}(y_{(k)})}(y_{(k+1)} - y_{(k)}) & \text{if } \hat{F}(y_{(k)}) \le p < \hat{F}(y_{(k+1)}) \\ y_{(n)} & \text{if } p = 1 \end{cases}$$

where $\hat{F}(t)$ is the estimated cumulative distribution for Y,

$$\hat{F}(t) = \frac{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij} I(y_{hij} \le t)}{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}}$$

and $I(\cdot)$ is the indicator function.

Standard Error

How PROC SURVEYMEANS estimates the standard error of a quantile depends on which variance method you specify in the VARMETHOD= option.

Taylor Series Method When you specify VARMETHOD=TAYLOR, or by default if you do not specify the VARMETHOD= option, PROC SURVEYMEANS uses Woodruff's method (Dorfman and Valliant 1993; Särndal, Swensson, and Wretman 1992; Francisco and Fuller 1991) to estimate the variances of quantiles. This method first constructs a confidence interval on a quantile $\hat{Q}(p)$. Then it uses the width of the confidence interval to estimate the standard error of $\hat{Q}(p)$. To construct the confidence interval, PROC SURVEYMEANS first estimates the variance of the estimated distribution function $\hat{F}(\hat{Q}(p))$ by

$$\hat{V}(\hat{F}(\hat{Q}(p))) = \sum_{h=1}^{H} \frac{n_h(1 - f_h)}{n_h - 1} \sum_{i=1}^{n_h} (e_{hi\cdot} - \bar{e}_{h\cdot\cdot})^2$$

where

$$\begin{aligned} e_{hi\cdot} &= \left(\sum_{j=1}^{m_{hi}} w_{hij} \left(I(y_{hij} \le \hat{Q}(p)) - \hat{F}(\hat{Q}(p)) \right) \right) / \, w_{\cdots} \\ \bar{e}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} e_{hi\cdot} \right) / \, n_h \\ w_{\cdots} &= \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij} \end{aligned}$$

Then $100(1-\alpha)\%$ confidence limits for $\hat{F}(\hat{Q}(p))$ can be constructed by

$$(\hat{p}_L, \quad \hat{p}_U) = \left(\hat{F}(\hat{Q}(p)) - t_{df,\,\alpha/2}\sqrt{\hat{V}(\hat{F}(\hat{Q}(p)))}, \qquad \hat{F}(\hat{Q}(p)) + t_{df,\,\alpha/2}\sqrt{\hat{V}(\hat{F}(\hat{Q}(p)))}\right)$$

where $t_{df,\,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom, described in the section "Degrees of Freedom" on page 10144.

When $(\hat{p}_L, \hat{p}_U)$ is out of the range of [0,1], the procedure does not compute the standard error of $\hat{Q}(p)$.

The $\hat{p}_L$th quantile is defined as

$$\hat{Q}(\hat{p}_L) = \begin{cases} y_{(1)} & \text{if } \hat{p}_L < \hat{F}(y_{(1)}) \\ y_{(k_L)} + \dfrac{\hat{p}_L - \hat{F}(y_{(k_L)})}{\hat{F}(y_{(k_L+1)}) - \hat{F}(y_{(k_L)})}(y_{(k_L+1)} - y_{(k_L)}) & \text{if } \hat{F}(y_{(k_L)}) \le \hat{p}_L < \hat{F}(y_{(k_L+1)}) \\ y_{(d)} & \text{if } \hat{p}_L = 1 \end{cases}$$

and the $\hat{p}_U$th quantile is defined as

$$\hat{Q}(\hat{p}_U) = \begin{cases} y_{(1)} & \text{if } \hat{p}_U < \hat{F}(y_{(1)}) \\ y_{(k_U)} + \dfrac{\hat{p}_U - \hat{F}(y_{(k_U)})}{\hat{F}(y_{(k_U+1)}) - \hat{F}(y_{(k_U)})}(y_{(k_U+1)} - y_{(k_U)}) & \text{if } \hat{F}(y_{(k_U)}) \le \hat{p}_U < \hat{F}(y_{(k_U+1)}) \\ y_{(d)} & \text{if } \hat{p}_U = 1 \end{cases}$$

The standard error of $\hat{Q}(p)$ is then estimated by

$$\hat{\text{StdErr}}(\hat{Q}(p)) = \frac{\hat{Q}(\hat{p}_U) - \hat{Q}(\hat{p}_L)}{2t_{df,\,\alpha/2}}$$

where $t_{df,\,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom.

Replication Methods When you use the jackknife replication method as described in the section "Replication Methods for Variance Estimation" on page 10169, the following naive replication variance estimate for a quantile $\hat{Q}(p)$ can have poor properties:

$$\hat{V}(\hat{Q}(p)) = \sum_{r=1}^{R} \alpha_r \left(\hat{Q}^{(r)}(p) - \hat{Q}(p)\right)^2$$

Here α_r is a coefficient that corresponds to each replicate, R is the number of replicates, and $\hat{Q}^{(r)}(p)$ is the estimated quantile in the rth replicate.

Fuller (2009) proposes using a smoothing method similar to the following to modify the quantile estimates $\hat{Q}^{(r)}(p)$ for each replicate before using this naive variance estimate for a quantile $\hat{Q}(p)$.

In the rth replicate ($r = 1, 2, \ldots, R$), denote $\{y_{hij}^{(r)}, w_{hij}^{(r)}\}$ to be the observed values of variable Y that are associated with the replicate weights. Let $y_{(1)}^{(r)} < y_{(2)}^{(r)} < \cdots < y_{(n^{(r)})}^{(r)}$ be the sample order statistics for variable Y in the rth replicate, where $n^{(r)}$ is the total number of observations whose replicate weights $w_{hij}^{(r)} > 0$.

Let $\hat{Q}^{(r)}(p)$ be the estimated pth quantile and $\hat{F}^{(r)}(t)$ be the estimated cumulative distribution for Y,

$$\hat{Q}^{(r)}(p) = \begin{cases} y^{(r)}_{(1)} & \text{if } p < \hat{F}^{(r)}(y^{(r)}_{(1)}) \\ y^{(r)}_{(k)} + \dfrac{p - \hat{F}^{(r)}(y^{(r)}_{(k)})}{\hat{F}^{(r)}(y^{(r)}_{(k+1)}) - \hat{F}^{(r)}(y^{(r)}_{(k)})}(y^{(r)}_{(k+1)} - y^{(r)}_{(k)}) & \text{if } \hat{F}^{(r)}(y^{(r)}_{(k)}) \le p < \hat{F}^{(r)}(y^{(r)}_{(k+1)}) \\ y^{(r)}_{(n^{(r)})} & \text{if } p = 1 \end{cases}$$

and

$$\hat{F}^{(r)}(t) = \frac{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w^{(r)}_{hij} I(y^{(r)}_{hij} \le t)}{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w^{(r)}_{hij}}$$

where $I(\cdot)$ is the indicator function.

Choose a segment in the distribution function by two points, $\tilde{p}^{(r)}_L$ and $\tilde{p}^{(r)}_U$,

$$\begin{aligned} \tilde{p}^{(r)}_L &= \max\left(\frac{w^{(r)}_{(1)}}{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} w^{(r)}_{hij}},\ \hat{F}^{(r)}(\hat{Q}^{(r)}(p)) - 2\sqrt{p(1-p)/n^{(r)}}\right) \\ \tilde{p}^{(r)}_U &= \min\left(1,\ \hat{F}^{(r)}(\hat{Q}^{(r)}(p)) + 2\sqrt{p(1-p)/n^{(r)}}\right) \end{aligned}$$

where $w^{(r)}_{(1)}$ is the replicate weight that corresponds to $y^{(r)}_{(1)}$ in the rth replicate. Then a modified pth quantile in the rth replicate is defined by

$$\tilde{Q}^{(r)}(p) = \hat{Q}^{(r)}(\tilde{p}^{(r)}_L) + \frac{\hat{Q}^{(r)}(\tilde{p}^{(r)}_U) - \hat{Q}^{(r)}(\tilde{p}^{(r)}_L)}{\tilde{p}^{(r)}_U - \tilde{p}^{(r)}_L}\left(p - \tilde{p}^{(r)}_L\right)$$

PROC SURVEYMEANS then uses $\tilde{Q}^{(r)}(p)$ as the quantile estimate in the rth replicate and its deviation from the mean of $\tilde{Q}^{(r)}(p)$ to estimate the variance of $\hat{Q}(p)$:

$$\begin{aligned} \hat{V}(\hat{Q}(p)) &= \sum_{r=1}^{R} \alpha_r \left(\tilde{Q}^{(r)}(p) - \bar{\tilde{Q}}(p)\right)^2 \\ \bar{\tilde{Q}}(p) &= \frac{1}{R}\sum_{r=1}^{R} \tilde{Q}^{(r)}(p) \end{aligned}$$

If you want to use the naive replication variance estimates by using $\hat{Q}^{(r)}(p)$ instead of the smoothed $\tilde{Q}^{(r)}(p)$ to estimate the variance of $\hat{Q}(p)$, you can specify the *method-option* NAIVEQVAR as VARMETHOD=BRR(NAIVEQVAR) or VARMETHOD=JACKKNIFE(NAIVEQVAR).

Confidence Limits

Symmetric $100(1-\alpha)\%$ confidence limits are computed as

$$\left(\hat{Q}(p) - \hat{\text{StdErr}}(\hat{Q}(p))\, t_{df,\,\alpha/2}, \quad \hat{Q}(p) + \hat{\text{StdErr}}(\hat{Q}(p))\, t_{df,\,\alpha/2}\right)$$

If you specify the NONSYMCL option in the PROC SURVEYMEANS statement, when you use the VARMETHOD=TAYLOR option, the procedure computes $100(1-\alpha)\%$ nonsymmetric confidence limits:

$$\left(\hat{Q}(\hat{p}_L), \quad \hat{Q}(\hat{p}_U)\right)$$

Quantile Estimation with Poststratification

When you specify a POSTSTRATA statement, the quantile estimation and its variance estimation incorporate poststratification. For more information about poststratification, see the section "Poststratification" on page 10162.

For a selected sample, let $r = 1, 2, \ldots, R$ be the poststratum index; let $Z_1, Z_2, \ldots, Z_R$ be the population totals for each corresponding poststratum, and let I_r be the indicator variable for the poststratum r that is defined by

$$I_r(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to the } r\text{th poststratum} \\ 0 & \text{otherwise} \end{cases}$$

Denote the total sum of original weights in the sample for each poststratum as

$$\psi_r = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}\, I_r(h,i,j)$$

Assume that the observation *(h, i, j)* belongs to the *r*th poststratum. Then the poststratification weight for the observation *(h, i, j)* is

$$\tilde{w}_{hij} = w_{hij} \frac{Z_r}{\psi_r}$$

Then the estimated cumulative distribution function of Y, $\hat{F}(t)$ and the estimated pth quantile estimation $\hat{Q}(p)$ can be computed as in the section "Estimate of Quantile" on page 10153 by replacing the original weights, w_{hij}, with the poststratification weights, $\tilde{w}_{hij}$.

When you specify VARMETHOD=TAYLOR (or by default), the variance of $\hat{Q}(p)$ is estimated as in the section "Standard Error" on page 10153, except that the variance of the estimated distribution function $\hat{F}(\hat{Q}(p))$ is computed as follows.

For each poststratum $r = 1, 2, \ldots, r$, define

$$\hat{\theta}^{(r)}(\hat{Q}(p)) = Z_r^{-1} \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} I_r(h,i,j)\, \tilde{w}_{hij}\, \left(I(y_{hij} \le \hat{Q}(p)) - \hat{F}(\hat{Q}(p))\right)$$

where $I(\cdot)$ is the indicator function.

Assume that the observation *(h, i, j)* belongs to the *r*th poststratum. Let

$$\tilde{y}_{hij} = I(y_{hij} \leq \hat{Q}(p)) - \hat{F}(\hat{Q}(p)) - \hat{\theta}^{(r)}(\hat{Q}(p))$$

PROC SURVEYMEANS estimates the variance of the estimated distribution function $\hat{F}(\hat{Q}(p))$ with poststratification by

$$\hat{V}(\hat{F}(\hat{Q}(p))) = \sum_{h=1}^{H} \frac{n_h(1-f_h)}{n_h - 1} \sum_{i=1}^{n_h} (u_{hi\cdot} - \bar{u}_{h\cdot\cdot})^2$$

where

$$\begin{aligned} u_{hi\cdot} &= \left(\sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ \tilde{y}_{hij} \right) / \ \tilde{w}_{\cdots} \\ \bar{u}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} u_{hi\cdot} \right) / \ n_h \\ \tilde{w}_{\cdots} &= \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \tilde{w}_{hij} \end{aligned}$$

Domain Quantile

Let Y be the variable of interest in a complex survey, and let a subpopulation of interest be domain D. Denote $F_D(t)$ as the cumulative distribution function of Y in domain D. For $0 < p < 1$, the pth quantile of the population cumulative distribution function is

$$Q_D(p) = \inf\{y : F_D(y) \geq p\}$$

Let I_D be the corresponding indicator variable:

$$I_D(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to } D \\ 0 & \text{otherwise} \end{cases}$$

Assume that there are a total of d observations among the n observations in the entire sample that belong to domain D. Let $y_{(1)} < y_{(2)} < \cdots < y_{(d)}$ denote the order statistics of variable Y for these d observations that fall in domain D.

The cumulative distribution function of Y in domain D is estimated by

$$\hat{F}_D(t) = \frac{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij} I(y_{hij} \leq t) I_D(h,i,j)}{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij} I_D(h,i,j)}$$

and $I(\cdot)$ is the indicator function. Then the estimated quantile in domain D is

$$\hat{Q}_D(p) = \begin{cases} y_{(1)} & \text{if } p < \hat{F}_D(y_{(1)}) \\ y_{(k)} + \dfrac{p - \hat{F}_D(y_{(k)})}{\hat{F}_D(y_{(k+1)}) - \hat{F}_D(y_{(k)})}(y_{(k+1)} - y_{(k)}) & \text{if } \hat{F}_D(y_{(k)}) \le p < \hat{F}_D(y_{(k+1)}) \\ y_{(d)} & \text{if } p = 1 \end{cases}$$

In order to estimate the variance for $\hat{Q}_D(p)$, PROC SURVEYMEANS first estimates the variance of the estimated distribution function $\hat{F}_D(\hat{Q}_D(p))$ in domain D. When you specify VARMETHOD=TAYLOR (or by default), the variance of $\hat{F}_D(\hat{Q}_D(p))$ is estimated by

$$\hat{V}(\hat{F}_D(\hat{Q}_D(p))) = \sum_{h=1}^{H} \frac{n_h(1-f_h)}{n_h - 1} \sum_{i=1}^{n_h} (d_{hi\cdot} - \bar{d}_{h\cdot\cdot})^2$$

where

$$\begin{aligned} v_{hij} &= I_D(h,i,j)w_{hij} \\ v_{\cdots} &= \sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} v_{hij} \\ d_{hi\cdot} &= \left(\sum_{j=1}^{m_{hi}} v_{hij}\ (I(y_{hij} \le \hat{Q}_D(p)) - \hat{F}_D(\hat{Q}_D(p)))\right) / \ v_{\cdots} \\ \bar{d}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} d_{hi\cdot}\right) / \ n_h \end{aligned}$$

Then $100(1-\alpha)\%$ confidence limits for $\hat{F}_D(\hat{Q}_D(p))$ can be constructed by $(\hat{p}_{DL}, \hat{p}_{DU})$, where

$$\begin{aligned} \hat{p}_{DL} &= \hat{F}_D(\hat{Q}_D(p)) - t_{df,\,\alpha/2}\sqrt{\hat{V}(\hat{F}_D(\hat{Q}_D(p)))} \\ \hat{p}_{DU} &= \hat{F}_D(\hat{Q}_D(p)) + t_{df,\,\alpha/2}\sqrt{\hat{V}(\hat{F}_D(\hat{Q}_D(p)))} \end{aligned}$$

and $t_{df,\,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom, described in the section "Degrees of Freedom" on page 10144. When $(\hat{p}_{DL}, \hat{p}_{DU})$ is out of the range of [0,1], PROC SURVEYMEANS does not compute the standard error of $\hat{Q}_D(p)$.

The $\hat{p}_{DL}$th quantile is then estimated as

$$\hat{Q}_D(\hat{p}_{DL}) = \begin{cases} y_{(1)} & \text{if } \hat{p}_{DL} < \hat{F}_D(y_{(1)}) \\ y_{(k_L)} + \dfrac{(\hat{p}_{DL} - \hat{F}_D(y_{(k_L)}))(y_{(k_L+1)} - y_{(k_L)})}{\hat{F}_D(y_{(k_L+1)}) - \hat{F}_D(y_{(k_L)})} & \text{if } \hat{F}_D(y_{(k_L)}) \le \hat{p}_{DL} < \hat{F}_D(y_{(k_L+1)}) \\ y_{(d)} & \text{if } \hat{p}_{DL} = 1 \end{cases}$$

The $\hat{p}_{DU}$th quantile is then estimated as

$$\hat{Q}_D(\hat{p}_{DU}) = \begin{cases} y_{(1)} & \text{if } \hat{p}_{DU} < \hat{F}_D(y_{(1)}) \\ y_{(k_U)} + \dfrac{(\hat{p}_{DU} - \hat{F}_D(y_{(k_U)}))(y_{(k_U+1)} - y_{(k_U)})}{\hat{F}_D(y_{(k_U+1)}) - \hat{F}_D(y_{(k_U)})} & \text{if } \hat{F}_D(y_{(k_U)}) \le \hat{p}_{DU} < \hat{F}_D(y_{(k_U+1)}) \\ y_{(d)} & \text{if } \hat{p}_{DU} = 1 \end{cases}$$

The standard error of $\hat{Q}_D(p)$ is then estimated by

$$\hat{\text{StdErr}}(\hat{Q}_D(p)) = \frac{\hat{Q}_D(\hat{p}_{DU}) - \hat{Q}_D(\hat{p}_{DL})}{2t_{df,\,\alpha/2}}$$

where $t_{df,\,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution with df degrees of freedom.

Symmetric $100(1-\alpha)\%$ confidence limits for $\hat{Q}_D(p)$ are computed as

$$\left(\hat{Q}_D(p) - \hat{\text{StdErr}}(\hat{Q}_D(p))\, t_{df,\,\alpha/2}, \quad \hat{Q}_D(p) + \hat{\text{StdErr}}(\hat{Q}_D(p))\, t_{df,\,\alpha/2}\right)$$

If you specify the NONSYMCL option in the PROC SURVEYMEANS statement, the procedure displays $100(1-\alpha)\%$ nonsymmetric confidence limits as

$$\left(\hat{Q}_D(\hat{p}_{DL}), \quad \hat{Q}_D(\hat{p}_{DU})\right)$$

Domain Quantile Estimation with Poststratification

When you specify both a POSTSTRATA statement and a DOMAIN statement, the domain quantile estimation and its variance estimation incorporate poststratification. For more information about poststratification, see the section "Poststratification" on page 10162.

For a selected sample, let $r = 1, 2, \ldots, R$ be the poststratum index, let $Z_1, Z_2, \ldots, Z_R$ be the population totals for each corresponding poststratum, and let I_r be the indicator variable for the poststratum r:

$$I_r(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to the } r\text{th poststratum} \\ 0 & \text{otherwise} \end{cases}$$

The poststratification weights, $\tilde{w}_{hij}$, are defined as in the section "Quantile Estimation with Poststratification" on page 10156.

For domain D, let I_D be the corresponding indicator variable:

$$I_D(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to } D \\ 0 & \text{otherwise} \end{cases}$$

With poststratification, for variable Y, the estimated cumulative distribution in domain D, $\hat{F}_D(t)$, and its pth quantile estimation, $\hat{Q}_D(p)$, can be computed as in the section "Domain Quantile" on page 10157 by replacing the original weights, w_{hij}, with the poststratification weights, $\tilde{w}_{hij}$. However, the variance of

$\hat{F}_D(\hat{Q}_D(p))$, which is described in the section "Domain Quantile" on page 10157, is computed as follows when you specify the VARMETHOD=TAYLOR option (or by default).

Define

$$
\begin{aligned}
\hat{\theta}_D^{(r)}(\hat{Q}_D(p)) &= Z_r^{-1} \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} I_D(h,i,j) I_r(h,i,j)\ \tilde{w}_{hij}\ (I(y_{hij} \le \hat{Q}_D(p)) - \hat{F}_D(\hat{Q}_D(p))) \\
\hat{\bar{I}}_D^{(r)} &= Z_r^{-1} \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} I_r(h,i,j)\ I_D(h,i,j)\ \tilde{w}_{hij} \\
\hat{\theta}_D(p) &= \frac{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} I_D(h,i,j)\ \tilde{w}_{hij}\ (I(y_{hij} \le \hat{Q}_D(p)) - \hat{F}_D(\hat{Q}_D(p)))}{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} I_D(h,i,j)\ \tilde{w}_{hij}}
\end{aligned}
$$

Assume that the observation *(h, i, j)* belongs to the *r*th poststratum. Then the variance of $\hat{F}_D(\hat{Q}_D(p))$ is estimated by

$$
\begin{aligned}
\hat{V}(\hat{F}_D(\hat{Q}_D(p))) &= \sum_{h=1}^{H} \frac{n_h(1-f_h)}{n_h - 1} \sum_{i=1}^{n_h} (e_{hi\cdot} - \bar{e}_{h\cdot\cdot})^2 \\
e_{hij} &= I_D(h,i,j) \Big(I(y_{hij} \le \hat{Q}_D(p)) - \hat{F}_D(\hat{Q}_D(p)) \Big) - \hat{\theta}_D^{(r)}(\hat{Q}_D(p)) \\
&\quad - \left(I_D(h,i,j) - \hat{\bar{I}}_D^{(r)} \right) \hat{\theta}_D(p) \\
e_{hi\cdot} &= \sum_{j=1}^{m_{hi}} \tilde{w}_{hij} e_{hij} / \tilde{w}_{\cdots} \\
\bar{e}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} e_{hi\cdot} \right) / n_h
\end{aligned}
$$

Geometric Mean

For a continuous variable Y that has positive values, the SURVEYMEANS procedure can compute its geometric mean and associated standard error and confidence limits. To request these statistics, you can specify *statistic-keywords* such as GEOMEAN, GMSTDERR, and GMCLM.

The geometric mean of Y from a sample is computed as

$$
\begin{aligned}
\hat{\bar{Y}}_G &= \left(\prod_{h=1}^{H} \prod_{i=1}^{n_h} \prod_{j=1}^{m_{hi}} {y_{hij}}^{w_{hij}} \right)^{\frac{1}{w_{\cdots}}} \\
&= \exp\left(\frac{1}{w_{\cdots}} \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij} \ln(y_{hij}) \right)
\end{aligned}
$$

where

$$w_{\cdots} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}$$

is the sum of the weights over all observations in the data set.

When you use the Taylor series method, the variance estimation for the geometric mean is computed as

$$\hat{V}(\hat{\bar{Y}}_G) \quad = \quad \left(\hat{\bar{Y}}_G\right)^2 \sum_{h=1}^{H} \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (r_{hi\cdot} - \bar{r}_{h\cdot\cdot})^2$$

where

$$r_{hi\cdot} \quad = \quad \left(\sum_{j=1}^{m_{hi}} w_{hij} \; (\ln(y_{hij}) - \ln(\hat{\bar{Y}}_G)) \right) / \; w_{\cdots}$$

$$\bar{r}_{h\cdot\cdot} \quad = \quad \left(\sum_{i=1}^{n_h} r_{hi\cdot} \right) / \; n_h$$

The standard error of the geometric mean is the square root of the estimated variance:

$$\text{StdErr}(\hat{\bar{Y}}_G) = \sqrt{\hat{V}(\hat{\bar{Y}}_G)}$$

The confidence limits for the geometric means are computed based on the confidence limits for the log transformation of the Y variable as

$$\left(\exp(\ln(\hat{\bar{Y}}_G) \; - \; \gamma), \quad \exp(\ln(\hat{\bar{Y}}_G) \; + \; \gamma)\right)$$

where

$$\gamma = t_{df,\,\alpha/2} * \text{StdErr}(\hat{\bar{Y}}_G)/\hat{\bar{Y}}_G$$

and $t_{df,\,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution, with df calculated as in the section "*t* Test for the Mean" on page 10143.

If you use replication methods to estimate the variance by specifying the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option, the procedure computes the variance of a geometric means $\hat{V}_R(\hat{\bar{Y}}_G)$ by using the variability among replicate estimates to estimate the overall variance. See the section "Replication Methods for Variance Estimation" on page 10169 for more information.

Then the standard error is the square root of the estimated variance:

$$\text{StdErr}_R(\hat{\bar{Y}}_G) = \sqrt{\hat{V}_R(\hat{\bar{Y}}_G)}$$

The confidence limits for the geometric means are computed based on the confidence limits for the log transformation of the variable Y as

$$\left(\exp(\ln(\hat{\bar{Y}}_G) - \lambda), \quad \exp(\ln(\hat{\bar{Y}}_G) + \lambda)\right)$$

where

$$\lambda = t_{df,\,\alpha/2} * \text{StdErr}_R(\hat{\bar{Y}}_G)/\hat{\bar{Y}}_G$$

and $t_{df,\,\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the t distribution, with df calculated as in the section "*t* Test for the Mean" on page 10143.

Poststratification

After a probability sample is drawn and survey data are collected, researchers sometimes want to stratify the sample according to auxiliary information about the sampled population. This process is often called *poststratification.*

When poststratification is done properly, it can improve efficiency. It can also be used to adjust the sampling weights such that the marginal distribution of the sampling weights is in agreement with known auxiliary information from other resources, such as the census. The adjusted weight is often called the *poststratification weight*.

It is quite common for researchers to use poststratification techniques in survey data analysis.

Poststratification is also used by epidemiologists, who frequently analyze health survey data. They often compute statistics based on a process called *direct standardization*, a form of poststratification. For example, certain diseases, such as cancer, are more common among older populations. Therefore, to compare the prevalence rates among geographic regions that are populated with different age groups, it is necessary to make adjustments according to such demographic categories and to compute relative prevalence rates of the diseases.

For more information about poststratification, see Fuller (2009); Lohr (2022); Wolter (2007); Rao, Yung, and Hidiroglou (2002).

After you provide the population controls for each poststratum that is defined by the poststratification variables, the SURVEYMEANS procedure creates the poststratification weights accordingly. Then the procedure computes statistics that you request by using poststratification weights.

You can save the poststratification weights in an OUTPSWGT= data set to be used in subsequent analyses.

For a selected sample, let $p = 1, 2, \ldots, P$ be the poststratum index; let $Z_1, Z_2, \ldots, Z_P$ be the population totals (or poststratum totals) for the corresponding poststrata, and let I_p be a corresponding indicator variable for poststratum p defined by

$$I_p(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to poststratum } p \\ 0 & \text{otherwise} \end{cases}$$

Denote the total sum of original weights in the sample for each poststratum as

$$\psi_p = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij} I_p(h, i, j)$$

Then the poststratification weight for observation *(h, i, j)* is

$$\tilde{w}_{hij} = w_{hij} \frac{Z_p}{\psi_p}$$

The SURVEYMEANS procedure computes statistics by using the poststratification weights $\tilde{w}_{hij}$ instead of the original weights w_{hij}.

The standard error and confidence intervals of computed statistics are based on the estimated variances, which are computed by using either a replication method or the Taylor series method.

Replication Methods

When you specify the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option, PROC SURVEYMEANS computes the variance of a statistic by using replication methods, as described in the section "Replication Methods for Variance Estimation" on page 10169. However, with poststratification, an extra step is needed to adjust the weights.

First, PROC SURVEYMEANS constructs a replicate and computes appropriate replicate weights for the replicate. Then, by using the poststratification control totals, the procedure adjusts these replicate weights in the same way as described previously for constructing the poststratification weights for the full sample. Finally, PROC SURVEYMEANS computes the estimate for a desired statistics by using the poststratification weights that are adjusted from the replicate weights in the current replicate. Then the final variance is estimated by the variability among replicate estimates, as described in the section "Replication Methods for Variance Estimation" on page 10169.

Taylor Series Method

When you specify VARMETHOD=TAYLOR, or by default when you do not specify the VARMETHOD= option, PROC SURVEYMEANS uses the Taylor series method to estimate the variances of requested statistics.

Variance of the Mean and Sum The sum and mean of variable Y under poststratification are

$$\begin{aligned} \hat{Y}^{(PS)} &= \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ y_{hij} \\ \hat{\bar{Y}}^{(PS)} &= \hat{Y}^{(PS)} / \tilde{w}_{...} \end{aligned}$$

where

$$\tilde{w}_{...} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \tilde{w}_{hij}$$

is the sum of the poststratification weights over all observations in the sample.

For each poststratum $p = 1, 2, \ldots, P$, let the mean of variable Y be

$$\hat{\bar{Y}}^{(p)} = \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} I_p(h,i,j)\ \tilde{w}_{hij}\ y_{hij} \right) / \ Z_p$$

where Z_p is the total of the poststratification weights in poststratum p.

For observation *(h, i, j)*, assume that it belongs to the pth poststratum. Let

$$\tilde{y}_{hij} = y_{hij} - \hat{\bar{Y}}^{(p)}$$

PROC SURVEYMEANS estimates the variance of $\hat{\bar{Y}}^{(PS)}$ as

$$\hat{V}\left(\hat{\bar{Y}}^{(PS)}\right) = \sum_{h=1}^{H} \hat{V}_h\left(\hat{\bar{Y}}^{(PS)}\right)$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h\left(\hat{\bar{Y}}^{(PS)}\right) &= \frac{n_h(1-f_h)}{n_h - 1} \sum_{i=1}^{n_h} (e_{hi\cdot} - \bar{e}_{h\cdot\cdot})^2 \\
e_{hi\cdot} &= \left(\sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ \tilde{y}_{hij} \right) / \tilde{w}_{...} \\
\bar{e}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} e_{hi\cdot} \right) / \ n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h\left(\hat{\bar{Y}}^{(PS)}\right) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

PROC SURVEYMEANS estimates the variance of $\hat{Y}^{(PS)}$ as

$$\hat{V}\left(\hat{Y}^{(PS)}\right) = \hat{V}\left(\hat{\bar{Y}}^{(PS)}\right)\ \tilde{w}_{...}^2$$

Variance of the Domain Mean and Sum For a domain D, let I_D be the corresponding indicator variable:

$$I_D(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{ belongs to } D \\ 0 & \text{otherwise} \end{cases}$$

Let

$$\tilde{v}_{hij} = \tilde{w}_{hij} I_D(h,i,j) = \begin{cases} \tilde{w}_{hij} & \text{if observation } (h,i,j) \text{ belongs to } D \\ 0 & \text{otherwise} \end{cases}$$

The sum and mean of variable Y under poststratification in domain D are

$$\begin{aligned} \hat{Y}_D^{(PS)} &= \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \tilde{v}_{hij}\ y_{hij} \\ \hat{\bar{Y}}_D^{(PS)} &= \hat{Y}_D^{(PS)} / \tilde{v}_{...} \end{aligned}$$

where

$$\tilde{v}_{...} = \sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \tilde{v}_{hij}$$

is the sum of the poststratification weights over all observations in the sample in domain D. For each poststratum $p = 1, 2, \ldots, P$, let the mean of variable Y and the mean of the domain indicator variable in each poststratum be

$$\begin{aligned} \hat{\bar{Y}}_D^{(p)} &= \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} I_p(h,i,j)\ I_D(h,i,j)\ \tilde{w}_{hij}\ y_{hij} \right) / Z_p \\ \hat{\bar{I}}_D^{(p)} &= \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} I_p(h,i,j)\ I_D(h,i,j)\ \tilde{w}_{hij} \right) / Z_p \end{aligned}$$

Assume that the observation *(h, i, j)* belongs to the *p*th poststratum. Let

$$\begin{aligned} d_{hij} &= y_{hij} I_D(h,i,j) - \hat{\bar{Y}}_D^{(p)} \\ e_{hij} &= d_{hij} - \left(I_D(h,i,j) - \hat{\bar{I}}_D^{(p)} \right) \hat{\bar{Y}}_D^{(PS)} \end{aligned}$$

Then PROC SURVEYMEANS estimates the variance of domain sum $\hat{Y}_D^{(PS)}$ as

$$\hat{V}\left(\hat{Y}_D^{(PS)}\right) = \sum_{h=1}^{H} \hat{V}_h\left(\hat{Y}_D^{(PS)}\right)$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h\left(\hat{Y}_D^{(PS)}\right) &= \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (d_{hi\cdot} - \bar{d}_{h\cdot\cdot})^2 \\
d_{hi\cdot} &= \sum_{j=1}^{m_{hi}} \tilde{w}_{hij} d_{hij} \\
\bar{d}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} d_{hi\cdot}\right) / \, n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h\left(\hat{Y}_D^{(PS)}\right) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

Then PROC SURVEYMEANS estimates the variance of domain mean $\hat{\bar{Y}}_D^{(PS)}$ as

$$\hat{V}\left(\hat{\bar{Y}}_D^{(PS)}\right) = \sum_{h=1}^{H} \hat{V}_h\left(\hat{\bar{Y}}_D^{(PS)}\right)$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h\left(\hat{\bar{Y}}_D^{(PS)}\right) &= \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (e_{hi\cdot} - \bar{e}_{h\cdot\cdot})^2 \\
e_{hi\cdot} &= \sum_{j=1}^{m_{hi}} \tilde{v}_{hij} e_{hij} / \tilde{v}_{\cdots} \\
\bar{e}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} e_{hi\cdot}\right) / \, n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h\left(\hat{\bar{Y}}_D^{(PS)}\right) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

Variance of the Ratio Suppose you want to calculate the ratio of variable Y to variable X. Let x_{hij} and y_{hij} be the values of variable X and variable Y, respectively, for observation *(h, i, j)*.

The ratio of Y to X after poststratification is

$$\hat{R}^{(PS)} = \frac{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ y_{hij}}{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ x_{hij}}$$

where $\tilde{w}_{hij}$ is the poststratification weight for observation (h, i, j).

Assume that the observation *(h, i, j)* belongs to the pth poststratum. Let

$$\begin{aligned} \tilde{y}_{hij} &= y_{hij} - \hat{\bar{Y}}^{(p)} \\ \tilde{x}_{hij} &= x_{hij} - \hat{\bar{X}}^{(p)} \end{aligned}$$

where $\hat{\bar{Y}}^{(p)}$ and $\hat{\bar{X}}^{(p)}$ are the means of variable Y and variable X, respectively, in poststratum p.

The variance of $\hat{R}^{(PS)}$ is estimated by

$$\hat{V}(\hat{R}^{(PS)}) = \sum_{h=1}^{H} \hat{V}_h(\hat{R}^{(PS)})$$

where, if $n_h > 1$, then

$$\begin{aligned} \hat{V}_h(\hat{R}^{(PS)}) &= \frac{n_h(1-f_h)}{n_h - 1} \sum_{i=1}^{n_h} (g_{hi\cdot} - \bar{g}_{h\cdot\cdot})^2 \\ g_{hi\cdot} &= \frac{\sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ (\tilde{y}_{hij} - \tilde{x}_{hij}\hat{R}^{(PS)})}{\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ x_{hij}} \\ \bar{g}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} g_{hi\cdot} \right) / n_h \end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h(\hat{R}^{(PS)}) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

Variance of the Domain Ratio For a domain D, let I_D be the corresponding indicator variable:

$$I_D(h, i, j) = \begin{cases} 1 & \text{if observation } (h, i, j) \text{ belongs to } D \\ 0 & \text{otherwise} \end{cases}$$

Let

$$\tilde{v}_{hij} = \tilde{w}_{hij} I_D(h, i, j) = \begin{cases} \tilde{w}_{hij} & \text{if observation } (h, i, j) \text{ belongs to } D \\ 0 & \text{otherwise} \end{cases}$$

The ratio of variable Y to variable X in domain D after poststratification is estimated by

$$\hat{R}_D^{(PS)} = \frac{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ y_{hij}\ I_D(h,i,j)}{\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} \tilde{w}_{hij}\ x_{hij}\ I_D(h,i,j)}$$

For each poststratum $p = 1, 2, \ldots, P$, let the mean of variable X and Y in each poststratum be

$$\begin{aligned}
\hat{\bar{Y}}_D^{(p)} &= \left(\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} I_p(h,i,j)\ \tilde{v}_{hij}\ x_{hij}\right) / Z_p \\
\hat{\bar{X}}_D^{(p)} &= \left(\sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} I_p(h,i,j)\ \tilde{v}_{hij}\ y_{hij}\right) / Z_p
\end{aligned}$$

Assume that the observation *(h, i, j)* belongs to the *p*th poststratum. Let

$$r_{hij} = y_{hij} I_D(h,i,j) - \hat{\bar{Y}}_D^{(p)} - \left(x_{hij} I_D(h,i,j) - \hat{\bar{X}}_D^{(p)} \right) \hat{R}_D^{(PS)}$$

Then PROC SURVEYMEANS estimates the variance of domain ratio $\hat{R}_D^{(PS)}$ after poststratification as

$$\hat{V}\left(\hat{R}_D^{(PS)}\right) = \sum_{h=1}^{H} \hat{V}_h\left(\hat{R}_D^{(PS)}\right)$$

where, if $n_h > 1$, then

$$\begin{aligned}
\hat{V}_h\left(\hat{R}_D^{(PS)}\right) &= \frac{n_h(1-f_h)}{n_h - 1} \sum_{i=1}^{n_h} (r_{hi\cdot} - \bar{r}_{h\cdot\cdot})^2 \\
r_{hi\cdot} &= \sum_{j=1}^{m_{hi}} \tilde{w}_{hij} r_{hij} / \sum_{h=1}^{H}\sum_{i=1}^{n_h}\sum_{j=1}^{m_{hi}} \tilde{v}_{hij}\ x_{hij} \\
\bar{r}_{h\cdot\cdot} &= \left(\sum_{i=1}^{n_h} r_{hi\cdot}\right) / n_h
\end{aligned}$$

and if $n_h = 1$, then

$$\hat{V}_h\left(\hat{R}_D^{(PS)}\right) = \begin{cases} \text{missing} & \text{if } n_{h'} = 1 \text{ for } h' = 1, 2, \ldots, H \\ 0 & \text{if } n_{h'} > 1 \text{ for some } 1 \le h' \le H \end{cases}$$

Replication Methods for Variance Estimation

Recently replication methods have gained popularity for estimating variances in complex survey data analysis. One reason for this popularity is the relative simplicity of replication-based estimates, especially for nonlinear estimators; another is that modern computational capacity has made replication methods feasible for practical survey analysis. For more information, see Lohr (2022); Wolter (2007); Rust (1985); Dippo, Fay, and Morganstein (1984); Rao and Shao (1999); Rao, Wu, and Yue (1992); Rao and Shao (1996).

Replication methods draw multiple replicates (also called subsamples) from a full sample according to a specific resampling scheme. The most commonly used resampling schemes are the *balanced repeated replication* (BRR) method, the *jackknife* method, and the *bootstrap* method. For each replicate, the original weights are modified for the PSUs in the replicates to create replicate weights. The statistics of interest are estimated by using the replicate weights for each replicate. Then the variances of parameters of interest are estimated by the variability among the estimates derived from these replicates. You can use the REPWEIGHTS statement to provide your own replicate weights for variance estimation.

Bootstrap Method

The VARMETHOD=BOOTSTRAP option in the PROC SURVEYMEANS statement requests the bootstrap method for variance estimation. This method can be used for stratified sample designs and for designs that have no stratification. If your design is stratified, the bootstrap method requires at least two PSUs in each stratum. You can provide your own bootstrap replicate weights for the analysis by using a REPWEIGHTS statement, or the procedure can construct bootstrap replicate weights for the analysis.

PROC SURVEYMEANS estimates the parameter of interest (or requested statistics) from each replicate, and then uses the variability among replicate estimates to estimate the overall variance of these statistics.

This bootstrap method for complex survey data is similar to the method of Rao, Wu, and Yue (1992) and is also known as the bootstrap weights method (Mashreghi, Haziza, and Léger 2016). For more information, see Lohr (2022, Section 9.3.3), Wolter (2007, Chapter 5), Beaumont and Patak (2012), Fuller (2009, Section 4.5), and Shao and Tu (1995, Section 6.2.4). McCarthy and Snowden (1985), Rao and Wu (1988), Sitter (1992b), and Sitter (1992a) provide several adjusted bootstrap variance estimators that are consistent for complex survey data. The naive bootstrap variance estimator that is suitable for infinite populations is not consistent for complex survey data.

Replicate Weight Construction

If you do not provide replicate weights by using a REPWEIGHTS statement, PROC SURVEYMEANS constructs bootstrap replicate weights for the analysis. The procedure selects replicate bootstrap samples by with-replacement random sampling of PSUs within strata. You can specify the number of bootstrap replicates in the REPS= *method-option*; by default, the number of replicates is 250. (Increasing the number of replicates can improve the estimation precision but also increases the computation time.) You can specify the bootstrap sample sizes m_h in the MH= *method-option*; by default, $m_h = n_h - 1$, where n_h is the number of PSUs in stratum h.

In each replicate sample, the original sampling weights of the selected units are adjusted to reflect the full sample. These adjusted weights are the *bootstrap replicate weights*. In replicate r, the bootstrap replicate weight for observation j in PSU i in stratum h is computed as

$$\tilde{w}_{hij}^{(r)} = \left\{ 1 + \sqrt{\frac{1-f_h}{m_h(n_h-1)}} n_h k_{hi}^{(r)} - \sqrt{\frac{m_h(1-f_h)}{n_h-1}} \right\} w_{hij}$$

where $k_{hi}^{(r)}$ is the number of times PSU i is selected for replicate r, and f_h is the sampling fraction in stratum h that you can input by using either the RATE= or TOTAL= option in the PROC SURVEYMEANS statement.

You can use the OUTWEIGHTS=*SAS-data-set* method-option to store the bootstrap replicate weights in a SAS data set. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10177. You can provide these replicate weights to the procedure for subsequent analyses by using a REPWEIGHTS statement.

Variance Estimation

Let R be the total number of bootstrap replicates weights. Denote α_r as the replicate coefficient for the rth replicate ($r = 1, 2, \ldots, R$).

When the procedure generates the bootstrap replicate weights, $\alpha_r = 1/R$.

If you provide your own bootstrap replicate weights by including a REPWEIGHTS statement, you can specify α_r in the REPCOEFS= option. By default, $\alpha_r = 1/R$.

Suppose that θ is a population parameter of interest. Let $\hat{\theta}$ be the estimate from the full sample for θ, and let $\hat{\theta}_r$ be the estimate from the rth replicate subsample by using bootstrap replicate weights. PROC SURVEYMEANS estimates the variance of $\hat{\theta}$ by the following equation:

$$\hat{V}(\hat{\theta}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\theta}_r - \hat{\theta} \right)^2$$

Here, the degrees of freedom is the number of clusters minus the number of strata. If there are no clusters, then the degrees of freedom equals the number of observations minus the number of strata. If the design is not stratified, then the degrees of freedom equals the number of PSUs minus one.

If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYMEANS computes the variance of $\hat{\theta}$ by

$$\hat{V}(\hat{\theta}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\theta}_r - \bar{\theta} \right)^2$$

where $\bar{\theta}$ is the average of the replicate estimates and is calculated as follows:

$$\bar{\theta} = \frac{1}{R} \sum_{r=1}^{R} \hat{\theta}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom and if R' is less than the number of PSUs minus the number of strata, then the degrees of freedom is set to R'.

Balanced Repeated Replication (BRR) Method

The balanced repeated replication (BRR) method requires that the full sample be drawn by using a stratified sample design with two primary sampling units (PSUs) per stratum. Let H be the total number of strata. The total number of replicates R is the smallest multiple of 4 that is greater than H. However, if you prefer a larger number of replicates, you can specify the REPS=*number* option. If a *number* × *number* Hadamard matrix cannot be constructed, the number of replicates is increased until a Hadamard matrix becomes available.

Each replicate is obtained by deleting one PSU per stratum according to the corresponding Hadamard matrix and adjusting the original weights for the remaining PSUs. The new weights are called replicate weights.

Replicates are constructed by using the first H columns of the $R \times R$ Hadamard matrix. The rth ($r = 1, 2, \ldots, R$) replicate is drawn from the full sample according to the rth row of the Hadamard matrix as follows:

- If the (r, h) element of the Hadamard matrix is 1, then the first PSU of stratum h is included in the rth replicate and the second PSU of stratum h is excluded.
- If the (r, h) element of the Hadamard matrix is –1, then the second PSU of stratum h is included in the rth replicate and the first PSU of stratum h is excluded.

Note that the "first" and "second" PSUs are determined by data order in the input data set. Thus, if you reorder the data set and perform the same analysis by using BRR method, you might get slightly different results, because the contents in each replicate sample might change.

The replicate weights of the remaining PSUs in each half-sample are then doubled to their original weights. For more information about the BRR method, see Wolter (2007) and Lohr (2022).

By default, an appropriate Hadamard matrix is generated automatically to create the replicates. You can request that the Hadamard matrix be displayed by specifying the VARMETHOD=BRR(PRINTH) *method-option*. If you provide a Hadamard matrix by specifying the VARMETHOD=BRR(HADAMARD=) *method-option*, then the replicates are generated according to the provided Hadamard matrix.

You can use the VARMETHOD=BRR(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

Suppose that θ is a population parameter of interest. Let $\hat{\theta}$ be the estimate from the full sample for θ. Let $\hat{\theta}_r$ be the estimate from the rth replicate subsample by using replicate weights. PROC SURVEYMEANS estimates the variance of $\hat{\theta}$ by

$$\hat{V}(\hat{\theta}) = \frac{1}{R} \sum_{r=1}^{R} \left(\hat{\theta}_r - \hat{\theta}\right)^2$$

with H degrees of freedom, where H is the number of strata. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYMEANS computes the variance of $\hat{\theta}$ by

$$\hat{V}(\hat{\theta}) = \frac{1}{R} \sum_{r=1}^{R} \left(\hat{\theta}_r - \bar{\theta}\right)^2$$

where $\bar{\theta}$ is the average of the replicate estimates and is calculated as follows:

$$\bar{\theta} = \frac{1}{R}\sum_{r=1}^{R}\hat{\theta}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals the minimum between R' and the number of strata, H.

Fay's BRR Method

Fay's method is a modification of the BRR method, and it requires a stratified sample design with two primary sampling units (PSUs) per stratum. The total number of replicates R is the smallest multiple of 4 that is greater than the total number of strata H. However, if you prefer a larger number of replicates, you can specify the REPS= *method-option*.

For each replicate, Fay's method uses a Fay coefficient $0 \le \epsilon < 1$ to impose a perturbation of the original weights in the full sample that is gentler than using only half-samples, as in the traditional BRR method. The Fay coefficient $0 \le \epsilon < 1$ can be set by specifying the FAY = ϵ *method-option*. By default, $\epsilon = 0.5$ if the FAY *method-option* is specified without providing a value for ϵ (Judkins 1990; Rao and Shao 1999). When $\epsilon = 0$, Fay's method becomes the traditional BRR method. For more information, see Dippo, Fay, and Morganstein (1984); Fay (1984, 1989); Judkins (1990).

Let H be the number of strata. Replicates are constructed by using the first H columns of the $R \times R$ Hadamard matrix, where R is the number of replicates, $R > H$. The rth ($r = 1, 2, \ldots, R$) replicate is created from the full sample according to the rth row of the Hadamard matrix as follows:

- If the (r, h) element of the Hadamard matrix is 1, then the full sample weight of the first PSU in stratum h is multiplied by ϵ and the full sample weight of the second PSU is multiplied by $2 - \epsilon$ to obtain the rth replicate weights.
- If the (r, h) element of the Hadamard matrix is –1, then the full sample weight of the first PSU in stratum h is multiplied by $2 - \epsilon$ and the full sample weight of the second PSU is multiplied by ϵ to obtain the rth replicate weights.

You can use the VARMETHOD=BRR(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

By default, an appropriate Hadamard matrix is generated automatically to create the replicates. You can request that the Hadamard matrix be displayed by specifying the VARMETHOD=BRR(PRINTH) *method-option*. If you provide a Hadamard matrix by specifying the VARMETHOD=BRR(HADAMARD=) *method-option*, then the replicates are generated according to the provided Hadamard matrix.

Suppose that θ is a population parameter of interest. Let $\hat{\theta}$ be the estimate from the full sample for θ. Let $\hat{\theta}_r$ be the estimate from the rth replicate subsample by using replicate weights. PROC SURVEYMEANS estimates the variance of $\hat{\theta}$ by

$$\hat{V}(\hat{\theta}) = \frac{1}{R(1-\epsilon)^2}\sum_{r=1}^{R}\left(\hat{\theta}_r - \hat{\theta}\right)^2$$

with H degrees of freedom, where H is the number of strata. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYMEANS computes the variance of $\hat{\theta}$ by

$$\hat{V}(\hat{\theta}) = \frac{1}{R(1-\epsilon)^2} \sum_{r=1}^{R} \left(\hat{\theta}_r - \bar{\theta}\right)^2$$

where $\bar{\theta}$ is the average of the replicate estimates and is calculated as follows:

$$\bar{\theta} = \frac{1}{R} \sum_{r=1}^{R} \hat{\theta}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals the minimum between R' and the number of strata, H.

Jackknife Method

The jackknife method of variance estimation deletes one PSU at a time from the full sample to create replicates. The total number of replicates R is the same as the total number of PSUs. In each replicate, the sample weights of the remaining PSUs are modified by the jackknife coefficient α_r. The modified weights are called replicate weights.

The jackknife coefficient and replicate weights are described as follows.

Without Stratification If there is no stratification in the sample design (no STRATA statement), the jackknife coefficients α_r are the same for all replicates:

$$\alpha_r = \frac{R-1}{R} \quad \text{where } r = 1, 2, \ldots, R$$

Denote the original weight in the full sample for the jth member of the ith PSU as w_{ij}. If the ith PSU is included in the rth replicate ($r = 1, 2, \ldots, R$), then the corresponding replicate weight for the jth member of the ith PSU is defined as

$$w_{ij}^{(r)} = w_{ij}/\alpha_r$$

With Stratification If the sample design involves stratification, each stratum must have at least two PSUs to use the jackknife method.

Let stratum $\tilde{h}_r$ be the stratum from which a PSU is deleted for the rth replicate. Stratum $\tilde{h}_r$ is called the *donor stratum.* Let $n_{\tilde{h}_r}$ be the total number of PSUs in the donor stratum $\tilde{h}_r$. The jackknife coefficients are defined as

$$\alpha_r = \frac{n_{\tilde{h}_r} - 1}{n_{\tilde{h}_r}} \quad \text{where } r = 1, 2, \ldots, R$$

Denote the original weight in the full sample for the *j*th member of the *i*th PSU as w_{ij}. If the *i*th PSU is included in the *r*th replicate ($r = 1, 2, \ldots, R$), then the corresponding replicate weight for the *j*th member of the *i*th PSU is defined as

$$w_{ij}^{(r)} = \begin{cases} w_{ij} & \text{if } i\text{th PSU is not in the donor stratum } \tilde{h}_r \\ w_{ij}/\alpha_r & \text{if } i\text{th PSU is in the donor stratum } \tilde{h}_r \end{cases}$$

You can use the VARMETHOD=JACKKNIFE(OUTJKCOEFS=) *method-option* to save the jackknife coefficients into a SAS data set and use the VARMETHOD=JACKKNIFE(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

If you provide your own replicate weights in a REPWEIGHTS statement, then you can also provide corresponding jackknife coefficients in the JKCOEFS= or REPCOEFS= option. If you provide replicate weights but do not provide jackknife coefficients, PROC SURVEYMEANS uses $\alpha_r = (R-1)/R$ as the jackknife coefficient for all replicates by default.

Suppose that θ is a population parameter of interest. Let $\hat{\theta}$ be the estimate from the full sample for θ. Let $\hat{\theta}_r$ be the estimate from the *r*th replicate subsample by using replicate weights. PROC SURVEYMEANS estimates the variance of $\hat{\theta}$ by

$$\hat{V}(\hat{\theta}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\theta}_r - \hat{\theta}\right)^2$$

with $R - H$ degrees of freedom, where *R* is the number of replicates and *H* is the number of strata, or *R* – 1 when there is no stratification. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates *R*.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYMEANS computes the variance of $\hat{\theta}$ by

$$\hat{V}(\hat{\theta}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\theta}_r - \bar{\theta}\right)^2$$

where $\bar{\theta}$ is the average of the replicate estimates and is calculated as follows:

$$\bar{\theta} = \frac{1}{R} \sum_{r=1}^{R} \hat{\theta}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, *R*, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals $R' - H$.

Hadamard Matrix

A Hadamard matrix **H** is a square matrix whose elements are either 1 or –1 such that

$$\mathbf{HH}' = k\mathbf{I}$$

where k is the dimension of **H** and **I** is the identity matrix of order k. The order k is necessarily 1, 2, or a positive integer that is a multiple of 4.

For example, the following matrix is a Hadamard matrix of dimension $k = 8$:

$$\begin{array}{rrrrrrrr}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{array}$$

Computational Resources

Due to the complex nature of survey data analysis, the SURVEYMEANS procedure usually requires more memory than an analysis by the MEANS procedure for the same analysis variables. PROC SURVEYMEANS requires memory resources to keep a a copy of each unique value of the STRATUM, CLUSTER, and DOMAIN variables in addition to the memory needed for the categorical analysis variables and other computations.

The estimated memory needed by the SURVEYMEANS procedure is described as follows.

Let:

- T_{str} be the total number of STRATUM variables
- $L_{\text{str}}(t)$ be the number of unique values for the tth STRATUM variable, where $t = 1, 2, \ldots, T_{\text{str}}$
- H be the total number of strata
- T_{clu} be the total number of CLUSTER variables
- $L_{\text{clu}}(t)$ be the number of unique values for the tth CLUSTER variable, where $t = 1, 2, \ldots, T_{\text{clu}}$
- T_{dom} be the total number of DOMAIN variables in a domain (you might have multiple domains defined in a DOMAIN statement)
- $L_{\text{dom}}(t)$ be the number of unique values for the tth DOMAIN variable, where $t = 1, 2, \ldots, T_{\text{dom}}$
- D be the total number of domains
- T_{cont} be the total number of continuous analysis variables
- T_{clas} be the total number of categorical analysis variables (CLASS variable)

- $L_{\text{clas}}(t)$ be the number of unique values for the tth CLASS variable, where $t = 1, 2, \ldots, T_{\text{clas}}$
- T_{ratio} be the total number of ratios
- T_{pctl} be the total number of percentiles
- c be a constant on the order of 32 bytes (64 for 64-bit architectures) plus the maximum combined unformatted and formatted length among all the STRATUM, CLUSTER, DOMAIN, and CLASS variables

If all combinations of levels of categorical variables exist, the maximum potential memory (in bytes) requirements for the analysis is estimated by

$$c * P * Q + 2000 * (H + 1) * (D + 1) * Q$$

where

$$\begin{aligned} P &= \prod_{t=1}^{T_{\text{str}}} L_{\text{str}}(t) \prod_{t=1}^{T_{\text{clu}}} L_{\text{clu}}(t) \prod_{t=1}^{T_{\text{dom}}} L_{\text{dom}}(t) \\ Q &= T_{\text{cont}} + \sum_{t=1}^{T_{\text{clas}}} L_{\text{clas}}(t) + T_{\text{ratio}} + T_{\text{pctl}} \end{aligned}$$

A relatively small amount of memory, compared to the memory usage described in the preceding calculation, is also needed for the analysis.

When the data-dependent memory usage overwhelms what is available in the computer system, the procedure might open one or more utility files to complete the analysis. This process can be controlled by the SAS system option SUMSIZE=, which sets the memory threshold where utility file operations begin. For best results, set SUMSIZE= to be less than the amount of real memory that is likely to be available for the task. See the chapter on SAS system options in *SAS System Options: Reference* for a description of the SUMSIZE= option.

If PROC SURVEYMEANS reports that there is insufficient memory, increase SUMSIZE=. A SUMSIZE= value greater than MEMSIZE= has no effect. Therefore, you might also need to increase MEMSIZE=.

The MEMSIZE option can be specified at system invocation, on the SAS command line, or in a configuration file. However, the MEMSIZE system option is not available in some operating environments. See the *SAS Companion* for your operating environment for more information and for the syntax specification.

To report a procedure's memory consumption, you can use the FULLSTIMER option. The syntax is described in the *SAS Companion* for your operating environment.

Also see the *SAS System Options: Reference* for more information about how to adjust your computation resource parameters for your operating environment.

For additional information about the memory usage for categorical variables, see the section "Computational Resources" in the chapter "The MEANS Procedure" in the *Base SAS Procedures Guide: Statistical Procedures*.

Output Data Sets

You can use the Output Delivery System to create a SAS data set from any piece of PROC SURVEYMEANS output. See the section "ODS Table Names" on page 10186 for more information.

PROC SURVEYMEANS also provides an output data set that stores the replicate weights for BRR, jackknife, or bootstrap variance estimation and an output data set that stores the jackknife coefficients for jackknife variance estimation.

Replicate Weights Output Data Set

If you specify the OUTWEIGHTS= *method-option* for VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or VARMETHOD=BOOTSTRAP, PROC SURVEYMEANS stores the replicate weights in an output data set. The OUTWEIGHTS= output data set contains all observations from the DATA= input data set that are valid (used in the analysis). (A valid observation is an observation that has a positive value of the WEIGHT variable. Valid observations must also have nonmissing values of the STRATA and CLUSTER variables, unless you specify the MISSING option.)

The OUTWEIGHTS= data set contains all variables in the DATA= input data set and the replicate weight variables, RepWt_1, RepWt_2, . . ., RepWt_n, where n is the total number of replicates in the analysis. Each replicate weight variable contains the replicate weights for the corresponding replicate. Replicate weights equal 0 for observations that are not included in the replicate.

After the procedure creates replicate weights for a particular input data set and survey design, you can use the OUTWEIGHTS= *method-option* to store these replicate weights and then use them again in subsequent analyses, either in PROC SURVEYMEANS or in other survey procedures. You can use the REPWEIGHTS statement to provide replicate weights for the procedure.

Jackknife Coefficients Output Data Set

If you specify the OUTJKCOEFS= *method-option* for VARMETHOD=JACKKNIFE, PROC SURVEYMEANS stores the jackknife coefficients in an output data set. The OUTJKCOEFS= output data set contains one observation for each replicate. The OUTJKCOEFS= data set contains the following variables:

- Replicate, which is the replicate number for the jackknife coefficient
- JKCoefficient, which is the jackknife coefficient
- DonorStratum, which is the stratum of the PSU that was deleted to construct the replicate, if you specify a STRATA statement

After the procedure creates jackknife coefficients for a particular input data set and survey design, you can use the OUTJKCOEFS= *method-option* to store these coefficients and then use them again in subsequent analyses, either in PROC SURVEYMEANS or in the other survey procedures. You can use the JKCOEFS= option in the REPWEIGHTS statement to provide jackknife coefficients for the procedure.

Poststratification Weights Output Data Set

If you specify the OUTPSWGT= option in the POSTSTRATA statement, PROC SURVEYMEANS stores the poststratification weights in an output data set. This output data set contains all observations from the DATA= input data set that are valid. (A valid observation is an observation that has a positive value of the WEIGHT variable and is used in the analysis. Valid observations must also have nonmissing values of the STRATA and CLUSTER variables, unless you specify the MISSING option. For more information about valid observations, see the section "Data and Sample Design Summary" on page 10180.)

The OUTPSWGT= data set contains the following variables:

- all variables in the DATA= input data set
- _PSWt_, which contains poststratification weights

For VARMETHOD=BRR, VARMETHOD=JACKKNIFE, and VARMETHOD=BOOTSTRAP, the output data set that is created by the OUTPSWGT= option in the POSTSTRATA statement also contains replicate weights that are adjusted for poststratification. These replicates weights are similar to those in a data set that is created if you specify the OUTWEIGHTS= *method-option* in the PROC SURVEYMEANS statement. For more information about the replicate weights that are created by the OUTWEIGHTS= *method-option* in the PROC SURVEYMEANS statement, see the section "Replicate Weights Output Data Set" on page 10177.

However, if you specify both the OUTWEIGHTS= *method-option* in the PROC SURVEYMEANS statement and the OUTPSWGT= option in the POSTSTRATA statement, but with different data set names, the data set name in OUTPSWGT= option is ignored, and the name in the OUTWEIGHTS= *method-option* is used to store the poststratification replicate weights.

If you provide your own replicate weights by using a REPWEIGHTS statement for VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE, the poststratification replicate weights replace the original replicate weights in the OUTPSWGT= data set.

Rectangular and Stacking Structures in an Output Data Set

When you use an ODS output statement to create SAS data sets for certain tables in PROC SURVEYMEANS, there are two possible types of table structure for the output data sets: *rectangular* and *stacking*. A rectangular structure creates one observation for each analysis variable in the data set. A stacking structure creates only one observation in the output data set for all analysis variables.

Before SAS 9, the stacking table structure, similar to the table structure in PROC MEANS, was the default in PROC SURVEYMEANS. Since SAS 9, the new default is to produce a rectangular table in the output data sets. You can use the STACKING option to request that the procedure produce the output data sets by using a stacking table structure.

The STACKING option affects the following tables:

- Domain
- Ratio
- Statistics
- StrataInfo

Figure 121.7 and Figure 121.8 shows these two structures for analyzing the following data set:

```
data new;
   input sex$ x;
   datalines;
M 12
F 5
M 13
F 23
F 11
;
```

The following statements request the default rectangular structure of the output data set for the statistics table:

```
proc surveymeans data=new mean;
   ods output statistics=rectangle;
run;

proc print data=rectangle;
run;
```

Figure 121.7 shows the rectangular structure.

Figure 121.7 Rectangular Structure in the Output Data Set

Rectangular Structure in the Output Data Set

Obs	VarName	VarLevel	Mean	StdErr
1	x		12.800000	2.905168
2	sex	F	0.600000	0.244949
3	sex	M	0.400000	0.244949

The following statements specify the STACKING option to request that the output data set have a stacking structure:

```
proc surveymeans data=new mean stacking;
   ods output statistics=stacking;
run;

proc print data=stacking;
run;
```

Figure 121.8 shows the stacking structure of the output data set for the statistics table requested by the STACKING option.

Figure 121.8 Stacking Structure in the Output Data Set Requested by the STACKING option

Stacking Structure in the Output Data Set

Obs	x	x_Mean	x_StdErr	sex_F	sex_F_Mean	sex_F_StdErr	sex_M	sex_M_Mean	sex_M_StdErr
1	x	12.800000	2.905168	sex=F	0.600000	0.244949	sex=M	0.400000	0.244949

Displayed Output

The SURVEYMEANS procedure produces output that is described in the following sections.

Data and Sample Design Summary

The "Data Summary" table provides information about the input data set and the sample design. This table displays the total number of valid observations, where an observation is considered *valid* if it has nonmissing values for all procedure variables other than the analysis variables—that is, for all specified STRATA, CLUSTER, DOMAIN, POSTSTRATA, and WEIGHT variables. This number might differ from the number of nonmissing observations for an individual analysis variable, which the procedure displays in the "Statistics" table. See the section "Missing Values" on page 10138 for more information.

PROC SURVEYMEANS displays the following information in the "Data Summary" table:

- Number of Strata, if you specify a STRATA statement
- Number of Poststrata, if you specify a POSTSTRATA statement
- Number of Clusters, if you specify a CLUSTER statement
- Number of Observations, which is the total number of valid observations
- Sum of Weights, which is the sum over all valid observations, if you specify a WEIGHT statement

Class Level Information

If you use a CLASS statement to name classification variables for categorical analysis, or if you list any character variables in the VAR statement, then PROC SURVEYMEANS displays a "Class Level Information" table. This table contains the following information for each classification variable:

- CLASS Variable, which lists each CLASS variable name
- Levels, which is the number of values or levels of the classification variable
- Missing, which is the number of observations that have missing values for the classification variable
- Values, which lists the values of the classification variable. The values are separated by a white space character; therefore, to avoid confusion, you should not include a white space character within a classification variable value.

Stratum Information

If you specify the LIST option in the STRATA statement, PROC SURVEYMEANS displays a "Stratum Information" table. This table displays the number of valid observations in each stratum, as well as the number of nonmissing stratum observations for each analysis variable. The "Stratum Information" table provides the following for each stratum:

- Stratum Index, which is a sequential stratum identification number

- STRATA variable(s), which lists the levels of STRATA variables for the stratum
- Population Total, if you specify the TOTAL= option
- Sampling Rate, if you specify the TOTAL= or RATE= option. If you specify the TOTAL= option, the sampling rate is based on the number of valid observations in the stratum.
- N Obs, which is the number of valid observations
- Variable, which lists each analysis variable name
- Levels, which identifies each level for categorical variables
- N, which is the number of nonmissing observations for the analysis variable
- Clusters, which is the number of clusters, if you specify a CLUSTER statement

Variance Estimation

If the variance method is not Taylor series or if the NOMCAR option is used, by default, PROC SURVEYMEANS displays the following variance estimation specifications in the "Variance Estimation" table:

- Method, which is the variance estimation method
- Replicate Weights Data Set, which is the name of the SAS data set that contains the replicate weights
- Number of Replicates
- Initial seed for the random number that the bootstrap method uses
- Hadamard Data Set, which is the name of the SAS data set for the HADAMARD matrix if you specify the VARMETHOD=BRR(HADAMARD=) *method-option*
- Fay Coefficient, which is the value of the FAY coefficient if you specify the VARMETHOD=BRR(FAY) *method-option*
- Missing Levels Included (MISSING), if you specify the MISSING option
- Missing Levels Included (NOMCAR), if you specify the NOMCAR option

Statistics

The "Statistics" table displays all of the statistics that you request with *statistic-keywords* in the PROC SURVEYMEANS statement, except DECILES, MEDIAN, Q1, Q3, and QUARTILES, which are displayed in the "Quantiles" table. If you do not specify any *statistic-keywords*, then by default this table displays the following information for each analysis variable: the sample size, the mean, the standard error of the mean, and the confidence limits for the mean. The "Statistics" table can contain the following information for each analysis variable, depending on which *statistic-keywords* you request:

- Variable name

- Variable Label
- Level, which identifies each level for categorical variables
- N, which is the number of nonmissing observations
- N Miss, which is the number of missing observations
- Minimum
- Maximum
- Range
- Number of Clusters
- Sum of Weights
- DF, which is the degrees of freedom for the t test
- Mean
- Std Error of Mean, which is the standard error of the mean
- Var of Mean, which is the variance of the mean
- t Value, for testing H_0: population MEAN $= 0$
- Pr $> \mid t \mid$, which is the two-sided p-value for the t test
- $100(1-\alpha)\%$ CL for Mean, which are two-sided confidence limits for the mean
- $100(1-\alpha)\%$ Upper CL for Mean, which is a one-sided upper confidence limit for the mean
- $100(1-\alpha)\%$ Lower CL for Mean, which is a one-sided lower confidence limit for the mean
- Coeff of Variation, which is the coefficient of variation for the mean
- Sum
- Std Error of Sum, which is the standard error of the sum
- Var of Sum, which is the variance of the sum
- $100(1-\alpha)\%$ CL for Sum, which are two-sided confidence limits for the sum
- $100(1-\alpha)\%$ Upper CL for Sum, which is a one-sided upper confidence limit for the sum
- $100(1-\alpha)\%$ Lower CL for Sum, which is a one-sided lower confidence limit for the Sum
- Coeff of Variation for sum, which is the coefficient of variation for the sum

Quantiles

The "Quantiles" table displays all the quantiles that you request with either *statistic-keywords* such as DECILES, MEDIAN, Q1, Q3, and QUARTILES, or the PERCENTILE= option, or the QUANTILE= option in the PROC SURVEYMEANS statement.

The "Quantiles" table contains the following information for each quantile:

- Variable name
- Variable Label
- Percentile, which is the requested quantile in the format of %
- Percentile Label, which is the corresponding common name for a percentile if it exists—for example, *Median* for 50th percentile
- Estimate, which is the estimate for a requested quantile with respect to the population distribution
- Std Error, which is the standard error of the quantile
- $100(1-\alpha)\%$ Confidence Limits, which are two-sided confidence limits for the quantile

Domain Analysis

If you specify a DOMAIN statement, the procedure displays domain statistics in a "Domain Analysis" table. A "Domain Analysis" table displays all the requested statistics for each level of the domain request. The procedure produces a separate "Domain Analysis" for each separate domain request. For example, the DOMAIN statement

```
domain A B*C*D A*C C;
```

specifies four domain requests:

- A: all the levels of A
- C: all the levels of C
- A*C: all the interactive levels of A and C
- B*C*D: all the interactive levels of B, C, and D

The procedure displays four "Domain Analysis" tables, one for each domain definition. If you use an ODS OUTPUT statement to create an output data set for domain analysis, the output data set contains a variable Domain whose values are these domain definitions. It contains all the columns in the "Statistics" table plus columns of domain variable values.

If you specify a subset of domain levels in a DOMAIN statement by formatted domain variable values, then only the domain levels that contain these specific values are displayed in the output. For example, the following statement displays statistics only for white females and Asian females:

```
domain Gender('Female')*Race('White' 'Asian');
```

Differences of Domain Means

If you specify the DIFFMEANS option in a DOMAIN statement, the procedure displays the difference of domain means for each continuous analysis variable at pairwise levels. The procedure also estimates the standard error of the difference, along with the *t* test, the degrees of freedom, and the *p*-value for the *t* test. If you specify the CLDIFF option in a DOMAIN statement, the confidence limits for the estimated difference are also displayed. If you specify the ADJUST=BON option in a DOMAIN statement, then the procedure displays both the unadjusted *p*-value and confidence limits and the adjusted *p*-value for Bonferroni multiplicity and corresponding confidence limits.

Covariance Matrix of Domain Means

If you specify the COV option in a DOMAIN statement, the procedure displays the covariance matrix of domain means for each domain definition and for each continuous analysis variable, along with the identification for each domain level.

Domain Quantiles

If you specify a DOMAIN statement, and if you request statistics by specifying either *statistic-keywords* such as DECILES, MEDIAN, Q1, Q3, and QUARTILES, or the PERCENTILE= option, or the QUANTILE= option in the PROC SURVEYMEANS statement, then the procedure displays domain quantiles in a "Domain Quantiles" table. This table displays all the quantile statistics for each level of the domain request. It contains all the columns in the "Quantiles" table plus columns of DOMAIN variable values.

Ratio Analysis

The "Ratio Analysis" table displays statistics for all the ratios that you request in the RATIO statement. If you do not specify any *statistic-keywords* in the PROC SURVEYMEANS statement, then by default this table displays the ratios and standard errors. The "Ratio Analysis" table can contain the following information for each ratio, depending on which *statistic-keywords* you request:

- Numerator, which identifies the numerator variable of the ratio
- Denominator, which identifies the denominator variable of the ratio
- N, which is the number of observations used in the ratio analysis
- number of Clusters
- Sum of Weights
- DF, which is the degrees of freedom for the *t* test
- Ratio
- Std Err of Ratio, which is the standard error of the ratio
- Var, which is the variance of the ratio

- t Value, for testing H_0: population RATIO $= 0$
- Pr $> | t |$, which is the two-sided p-value for the t test
- $100(1-\alpha)\%$ CL for Ratio, which are two-sided confidence limits for the Ratio
- Upper $100(1-\alpha)\%$ CL for Ratio, which are one-sided upper confidence limits for the Ratio
- Lower $100(1-\alpha)\%$ CL for Ratio, which are one-sided lower confidence limits for the Ratio

When you use the ODS OUTPUT statement to create an output data set, if you use labels for your RATIO statement, these labels are saved in the variable Ratio Statement in the output data set.

Domain Ratio Analysis

If you specify a DOMAIN statement with a RATIO statement, the procedure displays domain ratios in a "Domain Ratio Analysis" table. A "Domain Ratio Analysis" table displays all the ratio statistics for each level of the domain request. It contains all the columns in the "Ratio Analysis" table plus columns of domain variable values.

Hadamard Matrix

If you specify the VARMETHOD=BRR(PRINTH) *method-option* in the PROC SURVEYMEANS statement, PROC SURVEYMEANS displays the Hadamard matrix that is used to construct replicates for BRR variance estimation.

If you provide a Hadamard matrix with the VARMETHOD=BRR(HADAMARD=) *method-option* but the procedure does not use the entire matrix, the procedure displays only the rows and columns that are actually used to construct replicates.

Geometric Means

The "Geometric Means" table displays all the statistics related to geometric mean that you request with *statistic-keywords* in the PROC SURVEYMEANS statement. The "Geometric Means" table can contain the following information for each analysis variable, depending on which *statistic-keywords* you request:

- Variable Name
- Variable Label
- Geometric Mean
- Std Error of Geometric Mean
- $100(1-\alpha)\%$ CL for Geometric Mean, which are two-sided confidence limits for the geometric mean
- $100(1-\alpha)\%$ Lower CL for Geometric Mean, which is a one-sided lower confidence limit for the geometric mean
- $100(1-\alpha)\%$ Upper CL for Geometric Mean, which is a one-sided upper confidence limit for the geometric mean

Domain Geometric Means

If you specify a DOMAIN statement and request any statistics related to geometric mean with *statistic-keywords* in the PROC SURVEYMEANS statement, the procedure displays these statistics for each domain level in a "Domain Geometric Means" table. It contains all the columns in the "Geometric Means" table plus columns of domain variable values.

ODS Table Names

PROC SURVEYMEANS assigns a name to each table that it creates. These names are listed in Table 121.4. You can use these names to refer to tables when you use the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 121.4 ODS Tables Produced by PROC SURVEYMEANS

ODS Table Name	Description	Statement	Option
ClassVarInfo	Class level information	CLASS	Default
Domain	Statistics in domains	DOMAIN	Default
DomainDiffs	Differences of domain means	DOMAIN	DIFFMEANS | DIFF
DomainGeoMeans	Statistics related to geometric means in domains	PROC and DOMAIN	Keywords
DomainMeanCov	Covariance matrix of domain means	DOMAIN	COV
DomainQuantiles	Quantiles in domains	DOMAIN	Default
DomainRatio	Statistics for ratios in domains	DOMAIN and RATIO	Default
DomainRatioDiffs	Differences of domain ratios	DOMAIN and RATIO	DIFFRATIOS | DIFF
GeometricMeans	Statistics related to geometric means	PROC	Keywords
HadamardMatrix	Hadamard matrix	PROC	PRINTH
Quantiles	Quantiles	PROC	Default
Ratio	Statistics for ratios	RATIO	Default
Statistics	Statistics	PROC	Default
StrataInfo	Stratum information	STRATA	LIST
Summary	Data summary	PROC	Default
VarianceEstimation	Variance estimation	PROC	VARMETHOD= or NOMCAR

For example, the following statements create the output data set MyStrata, which contains the StrataInfo table, and the output data set MyStat, which contains the Statistics table for the ice cream study discussed in the section "Stratified Sampling" on page 10105:

```
title1 'Analysis of Ice Cream Spending';
proc surveymeans data=IceCream total=StudentTotals;
   strata Grade / list;
   var Spending Group;
   weight Weight;
   ods output
      StrataInfo = MyStrata
      Statistics = MyStat;
run;
```

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, you can request specific plots by specifying the PLOTS= option in the PROC SURVEYMEANS statement.

PROC SURVEYMEANS provides a summary plot that includes a box plot and a histogram plot for continuous analysis variables. For categorical variables, plots are available in PROC SURVEYFREQ.

By default, PROC SURVEYMEANS produces summary plots. When you specify a DOMAIN statement, PROC SURVEYMEANS also produce domain plots by default. You can suppress default plots and request specific plots by specifying the PLOTS(ONLY)= option. For more information, see the description of the PLOTS= option.

PROC SURVEYMEANS assigns a name to each graph that it creates using ODS Graphics. You can use these names to refer to the graphs. Table 121.5 lists the names of the graphs that PROC SURVEYMEANS generates, together with their descriptions and the PLOTS= options (*plot-requests*) and statements that produce them.

Table 121.5 ODS Graphs Produced by PROC SURVEYMEANS

ODS Graph Name	Description	PLOTS= Option	Statement
BoxPlot	Box plots	BOXPLOT	PROC
DomainPlot	Box plots for domain statistics for each domain definition	DOMAIN	DOMAIN
Histogram	Histograms with overlaid kernel densities and normal densities	HISTOGRAM	PROC
SummaryPanel	Histograms with overlaid kernel densities and normal densities, and box plots in a single panel	SUMMARY	PROC

Examples: SURVEYMEANS Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

The section "Getting Started: SURVEYMEANS Procedure" on page 10102 contains examples of analyzing data from simple random sampling and stratified simple random sample designs. This section provides more examples that illustrate how to use PROC SURVEYMEANS.

Example 121.1: Stratified Cluster Sample Design

(View the complete code for this example (svmex1.sas) in the example repository.)

Consider the example in the section "Stratified Sampling" on page 10105. The study population is a junior high school with a total of 4,000 students in grades 7, 8, and 9. Researchers want to know how much these students spend weekly for ice cream, on the average, and what percentage of students spend at least $10 weekly for ice cream.

The example in the section "Stratified Sampling" on page 10105 assumes that the sample of students was selected using a stratified simple random sample design. This example shows analysis based on a more complex sample design.

Suppose that every student belongs to a study group and that study groups are formed within each grade level. Each study group contains between two and four students. Table 121.6 shows the total number of study groups for each grade.

Table 121.6 Study Groups and Students by Grade

Grade	Number of Study Groups	Number of Students
7	608	1,824
8	252	1,025
9	403	1,151
Total	1263	4,000

It is quicker and more convenient to collect data from students in the same study group than to collect data from students individually. Therefore, this study uses a stratified clustered sample design. The primary sampling units, or clusters, are study groups. The list of all study groups in the school is stratified by grade level. From each grade level, a sample of study groups is randomly selected, and all students in each selected study group are interviewed. The sample consists of eight study groups from the 7th grade, three groups from the 8th grade, and five groups from the 9th grade.

The SAS data set IceCreamStudy saves the responses of the selected students:

```
data IceCreamStudy;
   input Grade StudyGroup Spending @@;
   if (Spending < 10) then Group='less';
      else Group='more';
```

```
   datalines;
7  34  7     7  34  7    7 412  4     9  27 14
7  34  2     9 230 15    9  27 15     7 501  2
9 230  8     9 230  7    7 501  3     8  59 20
7 403  4     7 403 11    8  59 13     8  59 17
8 143 12     8 143 16    8  59 18     9 235  9
8 143 10     9 312  8    9 235  6     9 235 11
9 312 10     7 321  6    8 156 19     8 156 14
7 321  3     7 321 12    7 489  2     7 489  9
7  78  1     7  78 10    7 489  2     7 156  1
7  78  6     7 412  6    7 156  2     9 301  8
;
```

In the data set IceCreamStudy, the variable Grade contains a student's grade. The variable StudyGroup identifies a student's study group. It is possible for students from different grades to have the same study group number because study groups are sequentially numbered within each grade. The variable Spending contains a student's response regarding how much he spends per week for ice cream, in dollars. The variable GROUP indicates whether a student spends at least $10 weekly for ice cream. It is not necessary to store the data in order of grade and study group.

The SAS data set StudyGroup is created to provide PROC SURVEYMEANS with the sample design information shown in Table 121.6:

```
data StudyGroups;
   input Grade _total_;
   datalines;
7 608
8 252
9 403
;
```

The variable Grade identifies the strata, and the variable _TOTAL_ contains the total number of study groups in each stratum. As discussed in the section "Specification of Population Totals and Sampling Rates" on page 10139, the population totals stored in the variable _TOTAL_ should be expressed in terms of the primary sampling units (PSUs), which are study groups in this example. Therefore, the variable _TOTAL_ contains the total number of study groups for each grade, rather than the total number of students.

In order to obtain unbiased estimates, you create sampling weights by using the following SAS statements:

```
data IceCreamStudy;
   set IceCreamStudy;
   if Grade=7 then Prob=8/608;
   if Grade=8 then Prob=3/252;
   if Grade=9 then Prob=5/403;
   Weight=1/Prob;
run;
```

The sampling weights are the reciprocals of the probabilities of selections. The variable Weight contains the sampling weights. Because the sampling design is clustered and all students from each selected cluster are interviewed, the sampling weights equal the inverse of the cluster (or study group) selection probabilities.

The following SAS statements perform the analysis for this sample design:

```
title1 'Analysis of Ice Cream Spending';
proc surveymeans data=IceCreamStudy total=StudyGroups;
   strata Grade / list;
   cluster StudyGroup;
   var Spending Group;
   weight Weight;
run;
```

Output 121.1.1 provides information about the sample design and the input data set. There are three strata in the sample design, and the sample contains 16 clusters and 40 observations. The variable Group has two levels, 'less' and 'more.'

Output 121.1.1 Data Summary and Class Information

Analysis of Ice Cream Spending

The SURVEYMEANS Procedure

Data Summary	
Number of Strata	3
Number of Clusters	16
Number of Observations	40
Sum of Weights	3162.6

Class Level Information		
Variable	Levels	Values
Group	2	less more

Output 121.1.2 displays information for each stratum. Since the primary sampling units in this design are study groups, the population totals shown in Output 121.1.2 are the total numbers of study groups for each stratum or grade. This differs from Output 121.4, which provides the population totals in terms of students since students were the primary sampling units for that design. Output 121.1.2 also displays the number of clusters for each stratum and analysis variable.

Output 121.1.2 Stratum Information

Stratum Information								
Stratum Index	Grade	Pop. Total	Sampling Rate	N Obs	Variable	Level	N	Clusters
1	7	608	1.32%	20	Spending		20	8
					Group	less	17	8
						more	3	3
2	8	252	1.19%	9	Spending		9	3
					Group	less	0	0
						more	9	3
3	9	403	1.24%	11	Spending		11	5
					Group	less	6	4
						more	5	4

Output 121.1.3 displays the estimates of the average weekly ice cream expenditure and the percentage of students spending at least $10 weekly for ice cream.

Output 121.1.3 Statistics

Statistics						
Variable	Level	N	Mean	Std Error of Mean	95% CL for Mean	
Spending		40	8.923860	0.650859	7.51776370	10.3299565
Group	less	23	0.561437	0.056368	0.43966057	0.6832130
	more	17	0.438563	0.056368	0.31678698	0.5603394

Example 121.2: Domain Analysis

(View the complete code for this example (svmex2.sas) in the example repository.)

Suppose that you are studying profiles of 800 top-performing companies to provide information about their impact on the economy. You are also interested in the company profiles within each market type. A sample of 66 companies is selected with unequal probability across market types. However, market type is not included in the sample design. Thus, the number of companies within each market type is a random variable in your sample. To obtain statistics within each market type, you should use domain analysis. The data of the 66 companies are saved in the following data set:

```
data Company;
   length Type $14;
   input Type$ Asset Sale Value Profit Employee Weight;
   datalines;
Other            2764.0  1828.0  1850.3   144.0   18.7   9.6
Energy          13246.2  4633.5  4387.7   462.9   24.3  42.6
Finance          3597.7   377.8    93.0    14.0    1.1  12.2
Transportation   6646.1  6414.2  2377.5   348.2   47.1  21.8
HiTech           1068.4  1689.8  1430.2    72.9    4.6   4.3
Manufacturing    1125.0  1719.4  1057.5    98.1   20.4   4.5
Other            1459.0  1241.4   452.7    24.5   20.1   5.5
Finance          2672.3   262.5   296.2    23.1    2.2   9.3
Finance           311.0   566.2   932.0    52.8    2.7   1.9
Energy           1148.6  1014.6   485.1    60.6    4.0   4.5
Finance          5327.0   572.4   372.9    25.2    4.2  17.7
Energy           1602.7   678.4   653.0    75.6    2.8   6.0
Energy           5808.8  1288.4  2007.0   318.8    5.9  19.2
Medical           268.8   204.4   820.9    45.6    3.7   1.8
Transportation   5222.6  2627.8  1910.0   245.6   22.8  17.4
Other             872.7  1419.4   939.3    69.7   12.2   3.7
Retail           4461.7  8946.8  4662.7   289.0  132.1  15.0
HiTech           6719.2  6942.0  8240.2   381.3   85.8  22.1
Retail            833.4  1538.8  1090.3    64.9   15.4   3.5
Finance           415.9   167.3  1126.8    56.8    0.7   2.2
HiTech            442.4  1139.9  1039.9    57.6   22.7   2.3
Other             801.5  1157.0   664.2    56.9   15.5   3.4
Finance          4954.8   468.8   366.4    41.7    3.0  16.5
Finance          2661.9   257.9   181.1    21.2    2.1   9.3
Finance          5345.8   530.1   337.4    36.4    4.3  17.8
Energy           3334.3  1644.7  1407.8   157.6    6.4  11.4
Manufacturing    1826.6  2671.7   483.2    71.3   25.3   6.7
```

```
Retail            618.8  2354.7   767.7    58.6   19.0   2.9
Retail           1529.1  6534.0   826.3    58.3   65.8   5.7
Manufacturing    4458.4  4824.5  3132.1    28.9   67.0  15.0
HiTech           5831.7  6611.1  9464.7   459.6   86.7  19.3
Medical          6468.3  4199.2  3170.4   270.1   59.5  21.3
Energy           1720.7   473.1   811.1    86.6    1.6   6.3
Energy           1679.7  1379.9   721.1    91.8    4.5   6.2
Retail           4018.2 16823.4  2038.3   178.1  162.0  13.6
Other             227.1   575.8  1083.8    62.6    1.9   1.6
Finance          3872.8   362.0   209.3    27.6    2.4  13.1
Retail           3359.3  4844.7  2651.4   224.1   75.6  11.5
Energy           1295.6   356.9   180.8   162.3    0.6   5.0
Energy           1658.0   626.6   688.0   126.0    3.5   6.1
Finance         12156.7  1345.5   680.7   106.6    9.4  39.2
HiTech           3982.6  4196.0  3946.8   313.9   64.3  13.5
Finance          8760.7   886.4  1006.9    90.0    7.5  28.5
Manufacturing    2362.2  3153.3  1080.0   137.0   25.2   8.4
Transportation   2499.9  3419.0   992.6    47.2   25.3   8.8
Energy           1430.4  1610.0   664.3    77.7    3.5   5.4
Energy          13666.5 15465.4  2736.7   411.4   26.6  43.9
Manufacturing    4069.3  4174.7  2907.6   289.2   38.2  13.7
Energy           2924.7   711.9  1067.8   146.7    3.4  10.1
Transportation   1262.1  1716.0   364.3    71.2   14.5   4.9
Medical           684.4   672.9   287.4    61.8    6.0   3.1
Energy           3069.3  1719.0  1439.0   196.4    4.9  10.6
Medical           246.5   318.8   924.1    43.8    3.1   1.7
Finance         11562.2  1128.5   580.4    64.2    6.7  37.3
Finance          9316.0  1059.4   816.5    95.9    8.0  30.2
Retail           1094.3  3848.0   563.3    29.4   44.7   4.4
Retail           1102.1  4878.3   932.4    65.2   47.3   4.4
HiTech            466.4   675.8   845.7    64.5    5.2   2.4
Manufacturing   10839.4  5468.7  1895.4   232.8   47.8  35.0
Manufacturing     733.5  2135.3    96.6    10.9    2.7   3.2
Manufacturing   10354.2 14477.4  5607.2   321.9  188.5  33.5
Energy           1902.1  2697.9   329.3    34.2    2.2   6.9
Other            2245.2  2132.2  2230.4   198.9    8.0   8.0
Transportation    949.4  1248.3   298.9    35.4   10.4   3.9
Retail           2834.4  2884.6   458.2    41.2   49.8   9.8
Retail           2621.1  6173.8  1992.7   183.7  115.1   9.2
;
```

For each company in your sample, the variables are defined as follows:

- Type identifies the type of market for the company.
- Asset contains the company's assets, in millions of dollars.
- Sale contains sales, in millions of dollars.
- Value contains the market value of the company, in millions of dollars.
- Profit contains the profit, in millions of dollars.
- Employee contains the number of employees, in thousands.

- Weight contains the sampling weight.

The following SAS statements use PROC SURVEYMEANS to perform the domain analysis, estimating means, and other statistics for the overall population and also for the subpopulations (or domain) defined by market type. The DOMAIN statement specifies Type as the domain variable:

```
ods graphics on;
title 'Top Companies Profile Study';
proc surveymeans data=Company total=800 mean sum;
   var Asset;
   weight Weight;
   domain Type;
run;
```

Output 121.2.1 shows that there are 66 observations in the sample. The sum of the sampling weights equals 799.8, which is close to the total number of companies in the study population.

Output 121.2.1 Company Profile Study

Top Companies Profile Study

The SURVEYMEANS Procedure

Data Summary	
Number of Observations	66
Sum of Weights	799.8

Statistics				
Variable	Mean	Std Error of Mean	Sum	Std Error of Sum
Asset	6523.488510	720.557075	5217486	1073829

The "Statistics" table in Output 121.2.1 displays the estimates of the mean and total for all analysis variables for the entire set of 800 companies, while Output 121.2.2 shows the mean and total estimates for each company type.

When ODS Graphics is enabled, PROC SURVEYMEANS also displays Figure 121.2.3, which depicts the domain statistics for each company type in addition to the statistics in the full sample.

Output 121.2.2 Domain Analysis for Company Profile Study

Top Companies Profile Study

The SURVEYMEANS Procedure

Statistics for Type Domains					
Type	**Variable**	**Mean**	**Std Error of Mean**	**Sum**	**Std Error of Sum**
Energy	**Asset**	7868.302932	1941.699163	1449341	785962
Finance	**Asset**	7890.190264	1057.185336	1855773	704506
HiTech	**Asset**	5031.959781	732.436967	321542	183302
Manufacturing	**Asset**	7403.004250	1454.921083	888361	492577
Medical	**Asset**	5046.570609	1218.444638	140799	131942
Other	**Asset**	1850.250000	338.128984	58838	31375
Retail	**Asset**	2939.845750	393.692369	235188	94605
Transportation	**Asset**	4712.047359	888.954411	267644	163516

Output 121.2.3 Domain Analysis for Company Profile Study

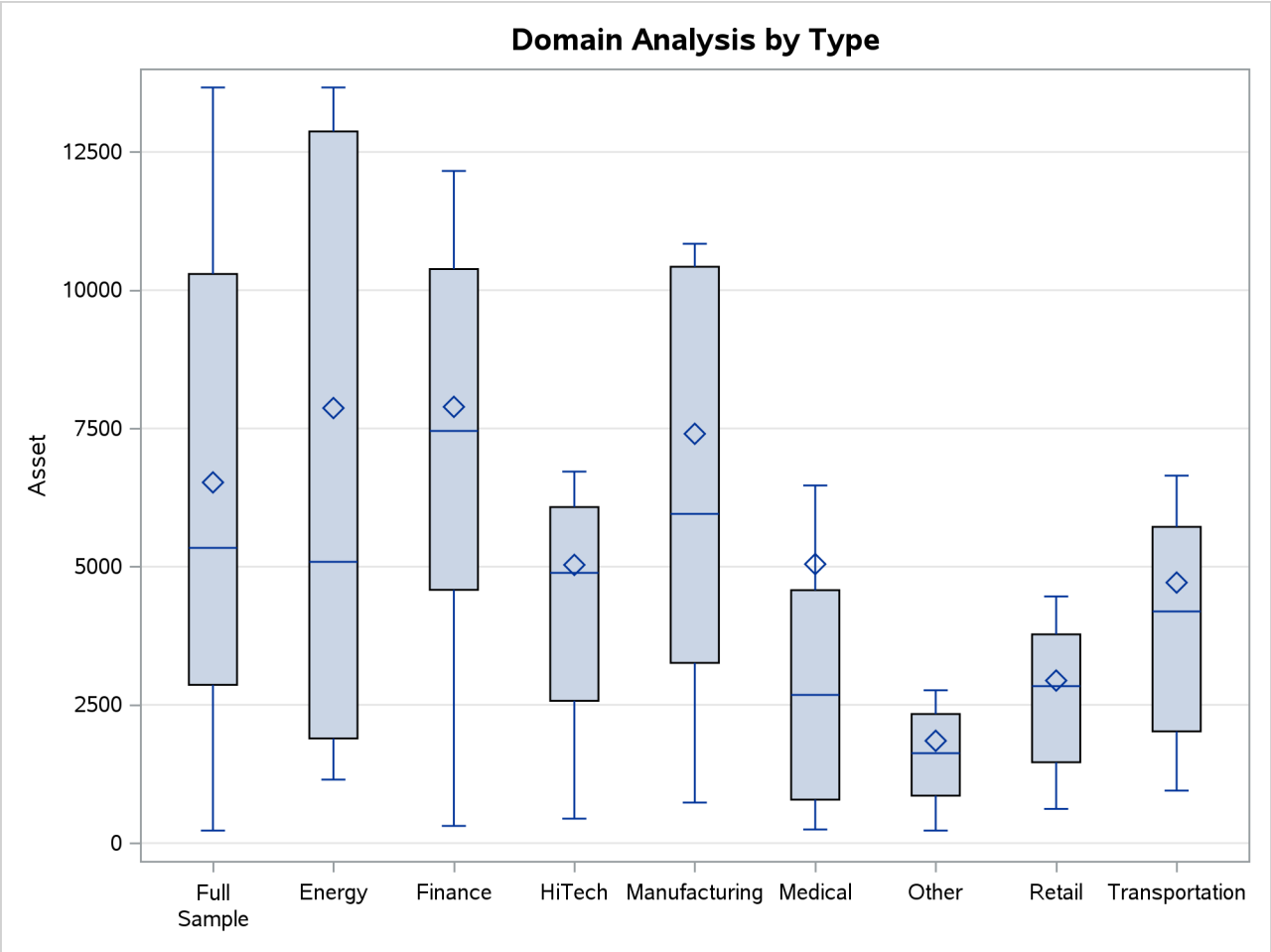

Example 121.3: Ratio Analysis

(View the complete code for this example (svmex3.sas) in the example repository.)

Suppose you are interested in the profit per employee and the sale per employee among the 800 top-performing companies in the data in the previous example. The following SAS statements illustrate how you can use PROC SURVEYMEANS to estimate these ratios:

```
title 'Ratio Analysis in Top Companies Profile Study';
proc surveymeans data=Company total=800 ratio;
   var Profit Sale Employee;
   weight Weight;
   ratio Profit Sale / Employee;
run;
```

The RATIO statement requests the ratio of the profit and the sales to the number of employees.

Output 121.3.1 shows the estimated ratios and their standard errors. Because the profit and the sales figures are in millions of dollars, and the employee numbers are in thousands, the profit per employee is estimated as $5,120 with a standard error of $1,059, and the sales per employee are $114,332 with a standard error of $20,503.

Output 121.3.1 Estimate Ratios

Ratio Analysis in Top Companies Profile Study

The SURVEYMEANS Procedure

Ratio Analysis			
Numerator	Denominator	Ratio	Std Error
Sale	Employee	114.332497	20.502742
Profit	Employee	5.119698	1.058939

Example 121.4: Analyzing Survey Data with Missing Values

(View the complete code for this example (svmex4.sas) in the example repository.)

As described in the section "Missing Values" on page 10138, the SURVEYMEANS procedure excludes an observation from the analysis if it has a missing value for the analysis variable or a nonpositive value for the WEIGHT variable.

However, if there is evidence indicating that the nonrespondents are different from the respondents for your study, you can use the NOMCAR option to compute descriptive statistics among respondents while still counting the number of nonrespondents.

This example continues the ice cream example in the section "Stratified Sampling" on page 10105 to illustrate how to perform a similar analysis when you have missing values.

Suppose that some of the students failed to provide the amounts spent on ice cream, as shown in the following data set, IceCream:

```
data IceCream;
   input Grade Spending @@;
   if Grade=7 then Prob=20/1824;
   if Grade=8 then Prob=9/1025;
   if Grade=9 then Prob=11/1151;
   Weight=1/Prob;
   datalines;
7 7  7  7  8  .  9 10  7  .  7 10  7  3  8 20  8 19  7 2
7 .  9 15  8 16  7  6  7  6  7  6  9 15  8 17  8 14  9 .
9 8  9  7  7  3  7 12  7  4  9 14  8 18  9  9  7  2  7 1
7 4  7 11  9  8  8  .  8 13  7  .  9  .  9 11  7  2  7 9
;

data StudentTotals;
   input Grade _total_;
   datalines;
7 1824
8 1025
9 1151
;
```

Considering the possibility that those students who did not respond spend differently than those students who did respond, you can use the NOMCAR option to request the analysis to treat the respondents as a domain rather than exclude the nonrespondents.

The following SAS statements produce the desired analysis:

```
title 'Analysis of Ice Cream Spending';
proc surveymeans data=IceCream total=StudentTotals nomcar mean sum;
   strata Grade;
   var Spending;
   weight Weight;
run;
```

Output 121.4.1 summarizes the analysis including the variance estimation method.

Output 121.4.1 Analysis of Incomplete Ice Cream Data Excluding Observations with Missing Values

Analysis of Ice Cream Spending

The SURVEYMEANS Procedure

Data Summary	
Number of Strata	3
Number of Observations	40
Sum of Weights	4000

Variance Estimation	
Method	Taylor Series
Missing Values	NOMCAR

Output 121.4.2 shows the mean and total estimates when treating respondents as a domain in the student population. Although the point estimates are the same as the analysis without the NOMCAR option, for this

particular example, the variance estimations are slightly higher when you assume that the missingness is not completely at random.

Output 121.4.2 Analysis of Incomplete Ice Cream Data Excluding Observations with Missing Values

Statistics				
Variable	Mean	Std Error of Mean	Sum	Std Error of Sum
Spending	9.770542	0.652347	32139	3515.126876

Example 121.5: Variance Estimation Using Replication Methods

(View the complete code for this example (svmex5.sas) in the example repository.)

In order to improve service, the San Francisco Municipal Railway (MUNI) conducts a survey to estimated passenger's average waiting time for MUNI's subway system.

The study uses a stratified cluster sample design. Each MUNI subway line is a stratum. The subway lines included in the study are 'J-Church,' 'K-Ingleside,' 'L-Taraval,' 'M-Ocean View,' 'N-Judah,' and the street car 'F-Market & Wharves.' The MUNI vehicles in service for these lines during a day are primary sampling units. Within each stratum, two vehicles (PSUs) are randomly selected. Then the waiting times of passengers for a selected MUNI vehicle are collected.

Table 121.7 shows the number of passengers that are interviewed in each of the selected MUNI vehicles.

Table 121.7 The Sample of the MUNI Waiting Time Study

MUNI Line	Vehicle	Number of Passengers
F-Market & Wharves	1	65
	2	102
J-Church	1	101
	2	142
K-Ingleside	1	145
	2	180
L-Taraval	1	135
	2	185
M-Ocean View	1	139
	2	203
N-Judah	1	306
	2	234

The collected data are saved in the SAS data set MUNIsurvey. The variable Line indicates which MUNI line a passenger is riding. The variable vehicle identifies the vehicle that a passenger is boarding. The variable Waittime is the time (in minutes) that a passenger waited. The variable weight contains the sampling weights, which are determined by selection probabilities within each stratum.

Output 121.5.1 displays the first 10 observations of the data set MUNIsurvey.

Output 121.5.1 First 10 Observations in the Data Set from the MUNI Subway Survey

MUNI Subway Passenger Waiting Time Survey Data

Obs	line	vehicle	passenger	waittime	weight
1	F-Market & Wharves	1	1	18	59
2	F-Market & Wharves	1	2	0	59
3	F-Market & Wharves	1	3	16	59
4	F-Market & Wharves	1	4	13	59
5	F-Market & Wharves	1	5	5	59
6	F-Market & Wharves	1	6	13	59
7	F-Market & Wharves	1	7	7	59
8	F-Market & Wharves	1	8	5	59
9	F-Market & Wharves	1	9	16	59
10	F-Market & Wharves	1	10	5	59

Using the VARMETHOD=BRR option, the following SAS statements analyze the MUNI subway survey by using the BRR method to estimate the variance:

```
title 'MUNI Passenger Waiting Time Analysis Using BRR';
proc surveymeans data=MUNIsurvey mean varmethod=brr mean clm;
   strata line;
   cluster vehicle;
   var waittime;
   weight weight;
run;
```

The STRATUM variable is line, which corresponds to MUNI lines. The two clusters within each stratum are identified by the variable vehicle. The sampling weights are stored in the variable weight. The mean and confident limits for passenger waiting time (in minutes) are requested statistics.

Output 121.5.2 summarizes the data and indicates that the variance estimation method is BRR with 8 replicates.

Output 121.5.2 MUNI Passenger Waiting Time Analysis Using the BRR Method

MUNI Passenger Waiting Time Analysis Using BRR

The SURVEYMEANS Procedure

Data Summary	
Number of Strata	6
Number of Clusters	12
Number of Observations	1937
Sum of Weights	143040

Variance Estimation	
Method	BRR
Number of Replicates	8

Output 121.5.3 reports that the average passenger waiting time for a MUNI vehicle is 7.33 minutes, with an estimated standard of 0.24 minutes, using the BRR method. The 95% confident limits for the mean are

estimated as 6.75 to 7.91 minutes.

Output 121.5.3 MUNI Passenger Waiting Time Analysis Using the BRR Method

Statistics				
Variable	Mean	Std Error of Mean	95% CL for Mean	
waittime	7.333012	0.237557	6.75172983	7.91429366

Alternatively, the variance can be estimated using the jackknife method if the VARMETHOD=JACKKNIFE option is used. The following SAS statements analyze the MUNI subway survey by using the jackknife method to estimate the variance:

```
title 'MUNI Passenger Waiting Time Analysis Using Jackknife';
proc surveymeans data=MUNIsurvey mean varmethod=jackknife mean clm;
   strata line;
   cluster vehicle;
   var waittime;
   weight weight;
run;
```

Output 121.5.4 summarizes the data and indicates that the variance estimation method is jackknife with 12 replicates.

Output 121.5.4 MUNI Passenger Waiting Time Analysis Using the Jackknife Method

MUNI Passenger Waiting Time Analysis Using Jackknife

The SURVEYMEANS Procedure

Data Summary	
Number of Strata	6
Number of Clusters	12
Number of Observations	1937
Sum of Weights	143040

Variance Estimation	
Method	Jackknife
Number of Replicates	12

Output 121.5.5 reports the statistics computed using the jackknife method. Although the average passenger waiting time remains the same (7.33 minutes), the standard error is slightly smaller 0.23 minutes when the jackknife method is used, as opposed to 0.24 minutes when the BRR method is used. The 95% confidence limits are between 6.76 and 7.90 minutes when the jackknife method is used.

Output 121.5.5 MUNI Passenger Waiting Time Analysis Using the Jackknife Method

Statistics				
Variable	Mean	Std Error of Mean	95% CL for Mean	
waittime	7.333012	0.232211	6.76481105	7.90121244

Example 121.6: Comparing Domain Means

(View the complete code for this example (svmex6.sas) in the example repository.)

Recall the example in the section "Stratified Sampling" on page 10105, which analyzed a stratified simple random sample from a junior high school to estimate students' average weekly spending for ice cream. When the data are collected, a student's gender is also recorded in the data. Then you can use the same sample to analyze the average weekly spending among male and female students.

The data set follows:

```
data IceCreamDataDomain;
   input Grade Spending Gender$ @@;
   datalines;
7   7  M   7   7  F   8  12  F
9  10  M   7   1  M   7  10  M
7   3  M   8  20  F   8  19  M
7   2  M   7   2  F   9  15  F
8  16  F   7   6  F   7   6  M
7   6  M   9  15  M   8  17  F
8  14  M   9   8  M   9   8  F
9   7  F   7   3  F   7  12  M
7   4  M   9  14  F   8  18  M
9   9  F   7   2  F   7   1  M
7   4  M   7  11  M   9   8  M
8  10  M   8  13  F   7   2  F
9   6  F   9  11  M   7   2  F
7   9  F
;
data IceCreamDataDomain;
   set IceCreamDataDomain;
   if Grade=7 then Prob=20/1824;
   if Grade=8 then Prob=9/1025;
   if Grade=9 then Prob=11/1151;
   Weight=1/Prob;
run;
data StudentTotals;
   input Grade _TOTAL_;
   datalines;
7 1824
8 1025
9 1151
;
```

In the IceCreamDataDomain data set, the Grade variable indicates a student's grade, which is the stratification variable. The Spending variable contains the dollar amount of each student's average weekly spending for ice cream. The Gender variable indicates a student's gender. The sampling weights are created by using the reciprocals of the probabilities of selection.

In the StudentTotals data set, the variable Grade is the stratification variable, and the _TOTAL_ variable contains the total numbers of students in the strata in the survey population.

Suppose you are interested in comparing differences in spending among grades. In particular, you want to examine whether average ice cream spending differs significantly between male and female students in the

eighth grade. You can use the DIFFMEANS option in a DOMAIN statement to request these analyses as follows:

```
ods graphics on;
proc surveymeans data=IceCreamDataDomain total=StudentTotals;
   strata Grade;
   var spending;
   domain Grade / diffmeans adjust=bon;
   domain Gender*Grade('8') / diffmeans;
   weight Weight;
run;
```

The first DOMAIN statement defines a domain definition by the Grade variable, which has three levels in the data: Grade='7', Grade='8', and Grade='9'. The DIFFMEANS option requests pairwise comparison of spending among three grades. The ADJUST=BON option requests a Bonferroni multiple comparison adjustment for the *p*-values for testing the pairwise differences in spending among grades.

Output 121.6.1 displays the domain statistics that PROC SURVEYMEANS produces for the first DOMAIN statement.

Output 121.6.1 Average Spending in Each Grade

The SURVEYMEANS Procedure

Statistics for Grade Domains

Grade	Variable	N	Mean	Std Error of Mean	95% CL for Mean	
7	Spending	20	5.000000	0.763551	3.4018694	6.5981306
8	Spending	9	15.444444	1.126773	12.8461011	18.0427878
9	Spending	11	10.090909	0.971908	7.9253640	12.2564542

Output 121.6.2 displays the comparison of spending among grades, as requested by the first DOMAIN statement.

Output 121.6.2 Comparison of Average Spending in Each Grade

Differences of Spending Means for Grade Domains

Grade	-Grade	Diff Estimate	Std Error	DF	t Value	Pr > |t|	Adj P
7	8	-10.444444	1.361113	19	-7.67	<.0001	<.0001
7	9	-5.090909	1.235967	19	-4.12	0.0006	0.0018
8	9	5.353535	1.488026	8	3.60	0.0070	0.0210

Figure 121.6.3 displays the plot for average spending in each grade.

Output 121.6.3 Average Spending in Each Grade

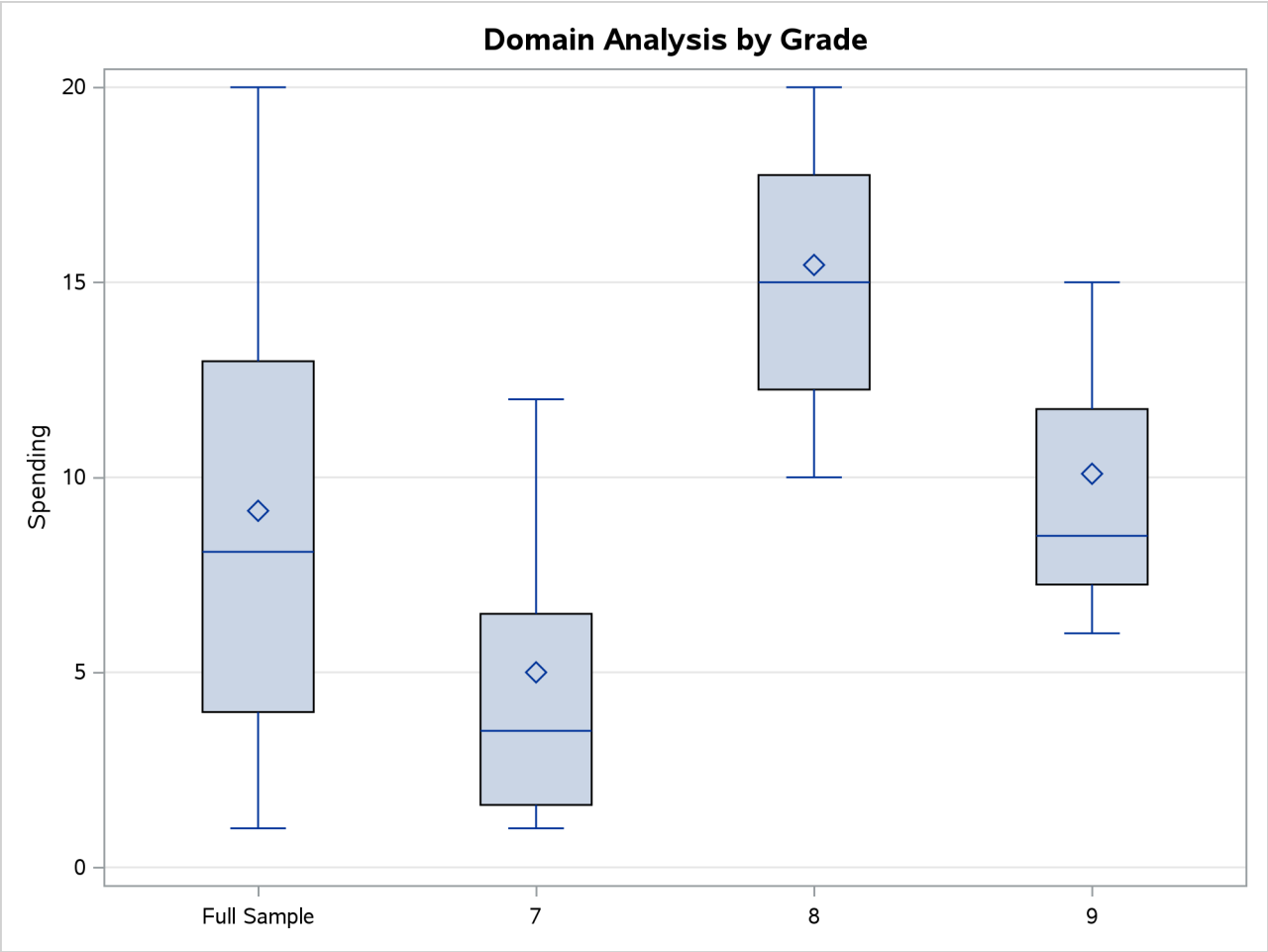

The second DOMAIN statement uses the combination of levels of two DOMAIN variables (Gender and Grade) to form domain levels. In addition to the three levels of the Grade variable, the Gender variable has two levels Gender='F' and Gender='M'. However, specification of `Grade('8')` indicates that only the results associated with the eighth grade are of interest. The DIFFMEANS option requests that PROC SURVEYMEANS compare the difference between domain means under this specification, which are the average spending for male and female in the eighth grade.

Therefore, PROC SURVEYMEANS compares the average spending only between male and female students within Grade 8.

Output 121.6.4 displays the domain statistics by gender only for students in the eighth grade.

Output 121.6.4 Average Spending by Gender in Grade 8

The SURVEYMEANS Procedure

Statistics for Gender*Grade Domains

Gender	Grade	Variable	N	Mean	Std Error of Mean	95% CL for Mean	
F	8	Spending	5	15.600000	1.355626	12.4739216	18.7260784
M	8	Spending	4	15.250000	1.880699	10.9131011	19.5868989

Output 121.6.5 displays the comparison of average spending among male and female students in the eighth grade, as requested by the second DOMAIN statement.

Output 121.6.5 Comparison of Average Spending between Gender in Grade 8

Differences of Spending Means for Gender*Grade Domains								
Gender	-Gender	Grade	-Grade	Diff Estimate	Std Error	DF	t Value	Pr > |t|
F	M	8	8	0.350000	5.593545	8	0.06	0.9516

Figure 121.6.6 displays the plot for domain means that correspond to the second DOMAIN statement to compare the difference between male and female students in the eighth grade.

Output 121.6.6 Average Spending between Gender in Grade 8

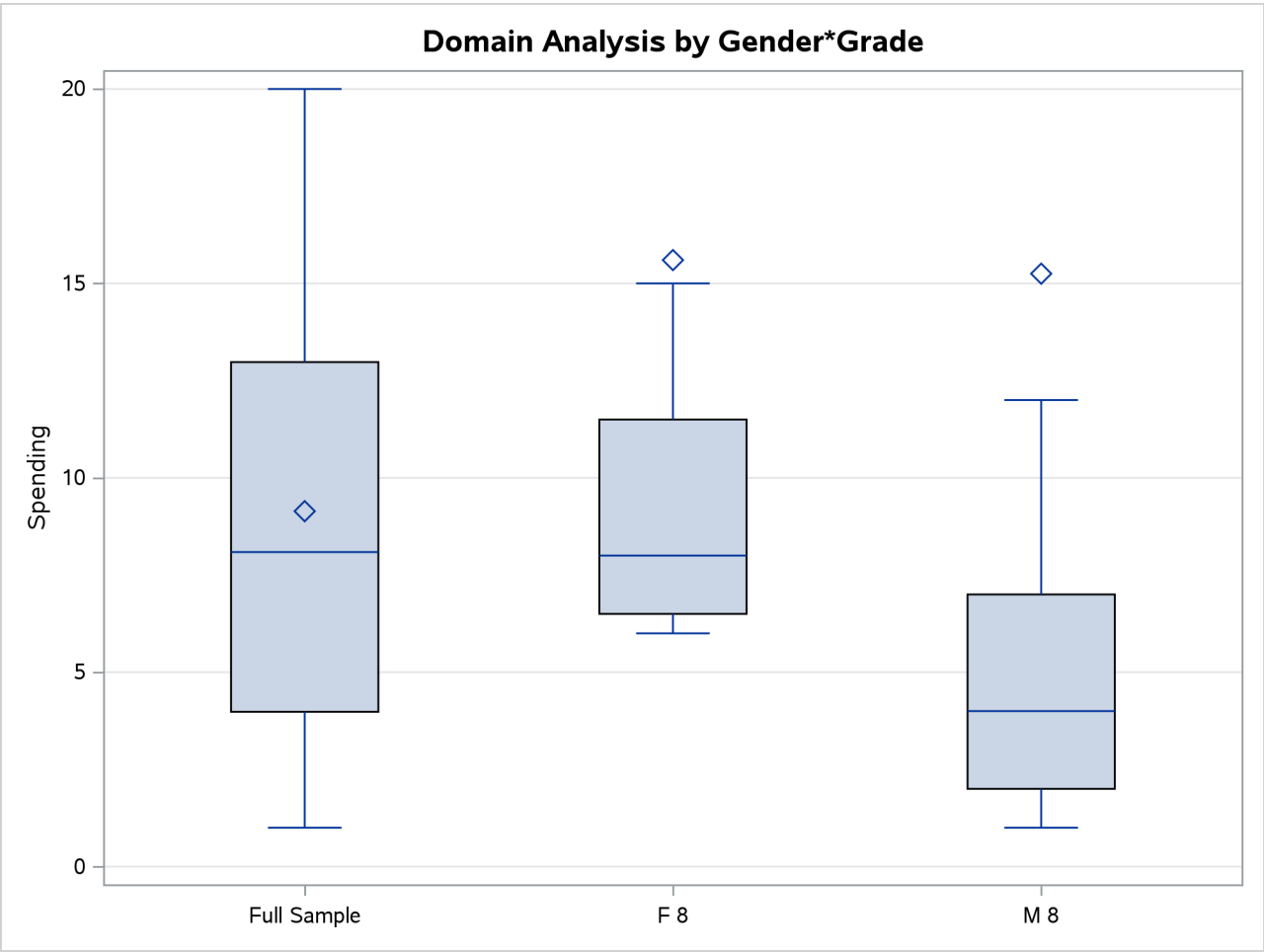

References

Beaumont, J.-F., and Patak, Z. (2012). "On the Generalized Bootstrap for Sample Surveys with Special Attention to Poisson Sampling." *International Statistical Review* 80:127–148.

Brick, J. M., and Kalton, G. (1996). "Handling Missing Data in Survey Research." *Statistical Methods in Medical Research* 5:215–238.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Dippo, C. S., Fay, R. E., and Morganstein, D. H. (1984). "Computing Variances from Complex Samples with Replicate Weights." In *Proceedings of the Survey Research Methods Section*, 489–494. Alexandria, VA: American Statistical Association.

Dorfman, A. H., and Valliant, R. (1993). "Quantile Variance Estimators in Complex Surveys." In *Proceedings of the Survey Research Methods Section*, 866–871. Alexandria, VA: American Statistical Association.

Fay, R. E. (1984). "Some Properties of Estimates of Variance Based on Replication Methods." In *Proceedings of the Survey Research Methods Section*, 495–500. Alexandria, VA: American Statistical Association.

Fay, R. E. (1989). "Theory and Application of Replicate Weighting for Variance Calculations." In *Proceedings of the Survey Research Methods Section*, 212–217. Alexandria, VA: American Statistical Association.

Francisco, C. A., and Fuller, W. A. (1991). "Quantile Estimation with a Complex Survey Design." *Annals of Statistics* 19:454–469.

Fuller, W. A. (1975). "Regression Analysis for Sample Survey." *Sankhyā, Series C* 37:117–132.

Fuller, W. A. (2009). *Sampling Statistics*. Hoboken, NJ: John Wiley & Sons.

Fuller, W. A., Kennedy, W. J., Schnell, D., Sullivan, G., and Park, H. J. (1989). *PC CARP*. Ames: Iowa State University Statistical Laboratory.

Hansen, M. H., Hurwitz, W. N., and Madow, W. G. (1953). *Sample Survey Methods and Theory.* 2 vols. New York: John Wiley & Sons.

Hidiroglou, M. A., Fuller, W. A., and Hickman, R. D. (1980). *SUPER CARP*. Ames: Iowa State University Statistical Laboratory.

Judkins, D. R. (1990). "Fay's Method for Variance Estimation." *Journal of Official Statistics* 6:223–239.

Kalton, G. (1983). *Introduction to Survey Sampling.* Vol. 07-035 of University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Kalton, G., and Kasprzyk, D. (1986). "The Treatment of Missing Survey Data." *Survey Methodology* 12:1–16.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Lee, E. S., Forthofer, R. N., and Lorimor, R. J. (1989). *Analyzing Complex Survey Data.* Vol. 07-071 of Sage University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Lohr, S. L. (2022). *Sampling: Design and Analysis*. 3rd ed. Boca Raton, FL: CRC Press.

Mashreghi, Z., Haziza, D., and Léger, C. (2016). "A Survey of Bootstrap Methods in Finite Population Sampling." *Statistics Surveys* 10:1–52.

McCarthy, P. J., and Snowden, C. B. (1985). *The Bootstrap and Finite Population Sampling*. Vital and Health Statistics, Series 2, no. 95. DHHS Publication Number 85–1369. Washington, DC: US Government Printing Office. https://www.cdc.gov/nchs/data/series/sr_02/sr02_095.pdf.

Rao, J. N. K., and Shao, J. (1996). "On Balanced Half-Sample Variance Estimation in Stratified Random Sampling." *Journal of the American Statistical Association* 91:343–348.

Rao, J. N. K., and Shao, J. (1999). "Modified Balanced Repeated Replication for Complex Survey Data." *Biometrika* 86:403–415.

Rao, J. N. K., and Wu, C. F. J. (1988). "Resampling Inference with Complex Survey Data." *Journal of the American Statistical Association* 83:231–241.

Rao, J. N. K., Wu, C. F. J., and Yue, K. (1992). "Some Recent Work on Resampling Methods for Complex Surveys." *Survey Methodology* 18:209–217.

Rao, J. N. K., Yung, W., and Hidiroglou, M. A. (2002). "Estimating Equations for the Analysis of Survey Data Using Poststratification Information." *Sankhyā, Series A* 64:364–378.

Rust, K. (1985). "Variance Estimation for Complex Estimators in Sample Surveys." *Journal of Official Statistics* 1:381–397.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Shao, J., and Tu, D. (1995). *The Jackknife and Bootstrap*. New York: Springer-Verlag.

Sitter, R. R. (1992a). "Comparing Three Bootstrap Methods for Survey Data." *Canadian Journal of Statistics* 20:135–154.

Sitter, R. R. (1992b). "A Resampling Procedure for Complex Survey Data." *Journal of the American Statistical Association* 87:755–765.

Terrell, G. R., and Scott, D. W. (1985). "Oversmoothed Nonparametric Density Estimates." *Journal of the American Statistical Association* 80:209–214.

Wolter, K. M. (2007). *Introduction to Variance Estimation*. 2nd ed. New York: Springer.

Woodruff, R. S. (1971). "A Simple Method for Approximating the Variance of a Complicated Estimate." *Journal of the American Statistical Association* 66:411–414.

Chapter 122
The SURVEYPHREG Procedure

Contents

Overview: SURVEYPHREG Procedure . . . **10209**
Getting Started: SURVEYPHREG Procedure . . . **10210**
Syntax: SURVEYPHREG Procedure . . . **10213**
- PROC SURVEYPHREG Statement . . . 10214
- BY Statement . . . 10222
- CLASS Statement . . . 10223
- CLUSTER Statement . . . 10225
- DOMAIN Statement . . . 10226
- ESTIMATE Statement . . . 10227
- FREQ Statement . . . 10228
- HAZARDRATIO Statement . . . 10229
- LSMEANS Statement . . . 10231
- LSMESTIMATE Statement . . . 10232
- MARGINS Statement . . . 10233
- MODEL Statement . . . 10234
- NLOPTIONS Statement . . . 10240
- OUTPUT Statement . . . 10240
- Programming Statements . . . 10241
- REPWEIGHTS Statement . . . 10243
- SLICE Statement . . . 10245
- STORE Statement . . . 10245
- STRATA Statement . . . 10245
- TEST Statement . . . 10246
- WEIGHT Statement . . . 10246

Details: SURVEYPHREG Procedure . . . **10247**
- Notation and Estimation . . . 10247
- Failure Time Distribution . . . 10248
- Time and CLASS Variable Usage . . . 10249
- Partial Likelihood Function for the Cox Model . . . 10252
- Counting Process Style of Input . . . 10253
- Left-Truncation of Failure Times . . . 10254
- The Multiplicative Hazards Model . . . 10254
- Firth's Modification for Maximum Likelihood Estimation . . . 10255
- Specifying the Sample Design . . . 10257
- Missing Values . . . 10259
- Variance Estimation . . . 10262

Taylor Series Linearization 10262
Balanced Repeated Replication (BRR) Method 10263
Bootstrap Method 10266
Jackknife Method 10267
Replicate Weights Method 10268
Degrees of Freedom 10270
Variance Adjustment Factors 10271
Variance Ratios and Standard Error Ratios 10272
Domain Analysis 10273
Hypothesis Tests, Confidence Intervals, and Residuals 10274
Testing the Global Null Hypothesis 10274
Model Fit Statistics 10275
Contrasts and Hypothesis Tests 10275
Confidence Intervals 10276
Hazard Ratios 10277
Residuals 10277
Hazard Ratios 10280
Output Data Sets 10282
Displayed Output 10283
ODS Table Names 10287
ODS Graphics 10288
Examples: SURVEYPHREG Procedure **10289**
Example 122.1: Analysis of Clustered Data 10289
Example 122.2: Stratification, Clustering, and Unequal Weights 10292
Example 122.3: Domain Analysis 10296
Example 122.4: Variance Estimation by Using Replicate Weights 10300
Example 122.5: A Test of the Proportional Hazards Assumption Using Programming Statements 10303
References **10304**

Overview: SURVEYPHREG Procedure

The SURVEYPHREG procedure performs regression analysis based on the Cox proportional hazards model for sample survey data. Cox's semiparametric model is widely used in the analysis of survival data to estimate hazard rates when adequate explanatory variables are available. The procedure provides design-based variance estimates, confidence intervals, and hypothesis tests concerning the parameters and model effects. See Chapter 3, "Introduction to Statistical Modeling with SAS/STAT Software," and Chapter 15, "Introduction to Survey Procedures," for an introduction to the basic concepts of survey data analysis; see Chapter 14, "Introduction to Survival Analysis Procedures," for an introduction to the basic concepts of survival analysis.

The survival time of each member of a finite population is assumed to follow its own hazard function, $\lambda_i(t)$, expressed as

$$\lambda_i(t) = \lambda(t; \mathbf{Z}_i(t)) = \lambda_0(t)\ \exp(\mathbf{Z}_i'(t)\boldsymbol{\beta})$$

where $\lambda_0(t)$ is an arbitrary and unspecified baseline hazard function, $\mathbf{Z}_i(t)$ is the vector of explanatory variables for the ith population unit at time t, and $\boldsymbol{\beta}$ is the vector of unknown regression parameters.

The finite population regression parameter $\boldsymbol{\beta}_N$ is defined as the maximizer of the partial log likelihood when the entire finite population is observed. The SURVEYPHREG procedure produces a sample-based estimate $\hat{\boldsymbol{\beta}}$ of the proportional hazards regression parameters $\boldsymbol{\beta}_N$ for the finite population by maximizing the partial pseudo-log-likelihood $l_\pi(\boldsymbol{\beta}; \mathbf{Z}_i(t), t_i)$ based on observed covariates $\mathbf{Z}_i(t)$ and observed survival time t_i. The procedure also produces an estimate of the sampling variance $\mathrm{V}(\hat{\boldsymbol{\beta}}|\mathcal{F}_N)$, which assumes that the values of the finite population $\mathcal{F}_N$ are fixed. For statistical inference, PROC SURVEYPHREG incorporates complex survey sample designs, including designs with stratification, clustering, and unequal weighting.

The procedure also allows time-dependent explanatory variables. An explanatory variable is time-dependent if its value for any given individual can change over time. Time-dependent variables have many useful applications in survival analysis. You can include time-dependent variables such as blood pressure or blood chemistry measures that vary with time during the course of a study. You can also use time-dependent variables to test the validity of the proportional hazards model.

Several optimization techniques are available in SURVEYPHREG to maximize the log likelihood. Hazard ratio estimates can also be obtained along with parameter estimates. Sampling errors of the regression parameters and hazard ratios are computed by using either the Taylor series (linearization) method or one of the replication (resampling) methods that are based on complex sample designs (Binder 1983; Wolter 2007; Särndal, Swensson, and Wretman 1992; Binder 1992; Lohr 2022; Fuller 2009). These variance estimators essentially assume the finite population as fixed and estimate the variability due to the random sample selection mechanism.

The remaining sections of this chapter contain information about how to use PROC SURVEYPHREG, information about the underlying statistical methodology, and some applications of the procedure. The section "Getting Started: SURVEYPHREG Procedure" on page 10210 introduces PROC SURVEYPHREG with an example. The section "Syntax: SURVEYPHREG Procedure" on page 10213 describes the syntax of the procedure. The section "Details: SURVEYPHREG Procedure" on page 10247 summarizes the statistical techniques employed in PROC SURVEYPHREG. The section "Examples: SURVEYPHREG Procedure" on page 10289 includes some additional examples of useful applications. Experienced SAS/STAT software users might decide to proceed to the "Syntax" section, while other users might choose to read both the "Getting Started" and "Examples" sections before proceeding to "Syntax" and "Details."

Getting Started: SURVEYPHREG Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/surphgs.sas.)

This section uses a data set that is obtained by stratified random sampling from a simulated finite population to illustrate some of the basic features of PROC SURVEYPHREG.

Suppose the library system for a small county wants to study the length of time that books are borrowed over a specified study period, adjusting for the age of the borrower and accounting for the fact that some books are never returned. Suppose there are 10 branch libraries in the county. Assume that a list of 11,617 (simulated) transactions is available for the study period October 1, 2008, to December 31, 2008, and assume that this list can be used as the sampling frame. A stratified random sample with replacement is used to select 100 transactions, where branch libraries are the strata. The total number of transactions within branches range from 510 to 2,011 for the study period. The total sample size of 100 transactions is allocated proportionally across branches based on the number of transactions. For each selected transaction, telephone interviews were conducted to find out additional characteristics of the borrower. The data set LibrarySurvey contains the following variables for all units (transactions) in the sample:

- Branch, the library branch from which the book was borrowed
- SampleWeight, the survey sampling weight for the transaction
- CheckOut, the date the book was borrowed
- CheckIn, the date the book was returned, with a missing value if the book was not returned by December 31, 2008
- Age, the age of the borrower

```
data LibrarySurvey;
   input Branch         2.
         SamplingWeight 7.2
         CheckOut       date10.
         CheckIn        date10.
         Age;
   datalines;
 1 103.60 08NOV2008 13NOV2008 18
 1 103.60 01OCT2008 07OCT2008 30
 1 103.60 05NOV2008 06NOV2008 73
 1 103.60 25OCT2008 26OCT2008 53
 1 103.60 09NOV2008 10NOV2008 55
 2 127.50 10DEC2008 15DEC2008 39
 2 127.50 19DEC2008         . 33
 2 127.50 26NOV2008 27NOV2008 41
 2 127.50 03NOV2008 07NOV2008 33

   ... more lines ...

10 118.35 14NOV2008 17NOV2008 29
```

```
10 118.35 11DEC2008 13DEC2008 35
10 118.35 21NOV2008 23NOV2008 46
;

data LibrarySurvey;
   set LibrarySurvey;
   Returned = (CheckIn ^= .);
   if (Returned) then
      lenBorrow = CheckIn                - CheckOut;
   else
   lenBorrow = input('31Dec2008',date9.) - CheckOut;
run;
```

PROC SURVEYPHREG can be used to estimate the regression parameters of a proportional hazards model and the design-based variance of the estimated coefficients. The design-based variance is useful when the finite population is considered fixed, as in this example. See Lohr (2022) and Särndal, Swensson, and Wretman (1992) for details.

The following statements request a proportional hazards regression of lenBorrow on Age with Returned as the censor indicator. A transaction is considered to be censored if its check-in date is missing. The WEIGHT statement specifies the sampling weight variable (SamplingWeight), and the STRATA statement specifies the stratification variable (Branch).

```
proc surveyphreg data = LibrarySurvey;
   weight SamplingWeight;
   strata Branch;
   model lenBorrow*Returned(0) = Age;
run;
```

Summary information about the model, number of observations, survey design, censored values, and variance estimation method are shown in Figure 122.1. The "Model Information" table summarizes the model you fit. The "Number of Observations" table displays the number of observations read and used by the procedure. This table also displays the sum of weights read and used. The sum of weights read (11,616.79) can be used as an estimator of the population size, and the sum of weights used can be used as an estimator of the respondent size in the population. The "Design Summary" table displays survey design information such as stratification and clustering. This example implements a stratified design with 10 strata. The "Censored Summary" and "Weighted Censored Summary" tables display the (weighted) number of censored and event units. Weighted counts can be used as estimators of the corresponding finite population quantities. For example, Figure 122.1 shows that 10% of the sampled units are censored and an estimated 10.05% of the population units are censored.

Figure 122.1 Summary Statistics

The SURVEYPHREG Procedure

Model Information	
Data Set	WORK.LIBRARYSURVEY
Dependent Variable	lenBorrow
Censoring Variable	Returned
Censoring Value(s)	0
Weight Variable	SamplingWeight
Stratum Variable	Branch
Ties Handling	BRESLOW

Figure 122.1 *continued*

Number of Observations Read	100
Number of Observations Used	100
Sum of Weights Read	11616.79
Sum of Weights Used	11616.79

Design Summary	
Number of Strata	10

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
100	90	10	10.00

Summary of the Weighted Number of Event and Censored Values

Total	Event	Censored	Percent Censored
11616.79	10449.22	1167.57	10.05

Variance Estimation	
Method	Taylor Series

Parameter estimates and their standard errors are shown in Figure 122.2. The estimated regression coefficient is highly significant with a value of 0.062, indicating a positive association between age and the length of time books are borrowed (recall that these are simulated data). In this example, the procedure uses the STRATA and WEIGHT statements to incorporate stratification and unequal weighting, respectively, into variance estimation. The degrees of freedom are calculated as the number of sampling units (100) minus the number of strata (10). Note that the estimated variance reported in Figure 122.2 ignores the finite population correction (*fpc*). You can use the TOTAL= or RATE= option in the PROC statement to include an *fpc* in your variance estimator.

Figure 122.2 Weighted Estimates and Their Standard Errors

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Hazard Ratio
Age	90	0.061593	0.008366	7.36	<.0001	1.064

Syntax: SURVEYPHREG Procedure

The following statements are available in the SURVEYPHREG procedure. Items within < > are optional.

```
PROC SURVEYPHREG <options> ;
    BY variables ;
    CLASS variable <(options)> <... variable <(options)>> </ options> ;
    CLUSTER variables ;
    DOMAIN variable <( 'formatted-level-value' ... 'formatted-level-value' )> < variable <( 'formatted-
            level-value' ... 'formatted-level-value' )>*variable <( 'formatted-level-value' ... 'formatted-
            level-value' )> > ;
    ESTIMATE <'label'> estimate-specification </ options> ;
    FREQ variable ;
    HAZARDRATIO <'label'> variable /options> ;
    LSMEANS <model-effects> </ options> ;
    LSMESTIMATE model-effect lsmestimate-specification </ options> ;
    MARGINS effects </ options> ;
    MODEL response <* censor(list)> = effects </ options> ;
    MODEL (t1, t2) <* censor(list)> = effects </ options> ;
    NLOPTIONS <options> ;
    OUTPUT <OUT=SAS-data-set> <keyword=name ... keyword=name> </ options> ;
    Programming statements ;
    REPWEIGHTS variables </ options> ;
    SLICE model-effect </ options> ;
    STRATA variables </ option> ;
    STORE <OUT= >item-store-name <' LABEL='label'> ;
    TEST <model-effects> </ options> ;
    WEIGHT variable ;
```

The PROC SURVEYPHREG and MODEL statements are required. The CLASS statement, if present, must precede the MODEL statement.

The following sections describe the PROC SURVEYPHREG statement and then describe the other statements in alphabetical order.

The ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, SLICE, STORE, and TEST statements are also available in other procedures. Summary descriptions of functionality and syntax for these statements are provided in this chapter, and you can find full documentation about them in Chapter 20, "Shared Concepts and Topics."

PROC SURVEYPHREG Statement

PROC SURVEYPHREG < *options* > **;**

The PROC SURVEYPHREG statement invokes the SURVEYPHREG procedure. It also identifies the data set to be analyzed. Table 122.1 summarizes the *options* available in the PROC SURVEYPHREG statement.

Table 122.1 PROC SURVEYPHREG Statement Options

Option	Description
ATRISK	Displays a table that contains the sum of weights for the number of units and the sum of weights for the corresponding number of events in the risk sets
DATA=	Names the input SAS data set
MISSING	Treats missing values as a valid category
NAMELEN=	Specifies the length of effect names
NOMCAR	Uses missing observations specified as *not missing completely at random*
NOPRINT	Suppresses all displayed output
ORDER=	Specifies the sort order of CLASS variables
RATE=	Specifies the sampling rate
TOTAL=	Specifies the total number of primary sampling units
VARMETHOD=	Specifies the variance estimation method

You can specify the following *options* in the PROC SURVEYPHREG statement:

ATRISK

displays a table that contains the sum of weights for the number of units at risk at each distinct event time and the sum of weights for the corresponding number of events in the risk sets. For example, the risk set information in Figure 122.3 is displayed if the ATRISK option is specified in the example in the section "Getting Started: SURVEYPHREG Procedure" on page 10210.

Figure 122.3 Risk Set Information

The SURVEYPHREG Procedure

Risk Set Sum of Weights		
lenBorrow	At Risk	Event
1	11616.79	5440.11
2	6176.68	1177.71
3	4998.97	926.55
4	4072.42	1411.07
5	2661.35	461.89
6	2199.46	565.01
7	1634.45	236.58
8	1397.87	230.3

DATA=*SAS-data-set*

names the SAS data set that contains the data to be analyzed. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

MISSING

treats missing values as a valid (nonmissing) category for all categorical variables, which include CLASS, STRATA, CLUSTER, and DOMAIN variables. By default, if you do not specify the MISSING option, an observation is excluded from the analysis if it has a missing value for any of these categorical variables. For more information, see the section "Missing Values" on page 10259.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 20 and 200, inclusive. By default, NAMELEN=20.

NOMCAR

includes observations with missing values of the analysis variables that are specified in the MODEL statement as *not missing completely at random* (NOMCAR) for Taylor series variance estimation. When you specify the NOMCAR option, PROC SURVEYPHREG computes variance estimates by analyzing the nonmissing values as a domain (subpopulation), where the entire population includes both nonmissing and missing domains. See the section "Missing Values" on page 10259 for details.

By default, PROC SURVEYPHREG excludes an observation from analyses (and the corresponding variance computations) if that observation has a missing value for any of the variables in the MODEL statement. Note that if you specify the MISSING option for classification variables, then the procedure treats the missing values as a valid nonmissing level.

The NOMCAR option applies only to Taylor series variance estimation. Other replication methods do not use the NOMCAR option.

NOPRINT

suppresses all displayed output. Note that this option temporarily disables the Output Delivery System (ODS); see Chapter 23, "Using the Output Delivery System," for more information.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

RATE=*value* | *SAS-data-set*

R=*value* | *SAS-data-set*

specifies the sampling rate, which PROC SURVEYPHREG uses to compute a finite population correction for Taylor series or bootstrap variance estimation. This option is ignored for BRR and jackknife variance estimation.

If your sample design has multiple stages, you should specify the *first-stage sampling rate*, which is the ratio of the number of primary sampling units (PSUs) that are selected to the total number of PSUs in the population.

You can specify the sampling rate in either of the following ways:

value specifies a nonnegative number to use for a nonstratified design or for a stratified design that has the same sampling rate in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the sampling rates for a stratified design that has different sampling rates in the strata. You must provide the sampling rates in the data set variable named _RATE_.

The sampling rates must be nonnegative numbers. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC SURVEYPHREG converts that number to a proportion. The procedure treats the value 1 as 100% instead of 1%.

For more information, see the section "Population Totals and Sampling Rates" on page 10258.

If you do not specify the RATE= or TOTAL= option, then the Taylor series or bootstrap variance estimation does not include a finite population correction. You cannot specify both the TOTAL= option and the RATE= option in the same PROC SURVEYPHREG statement.

TOTAL=*value* | *SAS-data-set*

N=*value* | *SAS-data-set*

specifies the total number of primary sampling units (PSUs) in the population. PROC SURVEYPHREG uses the *value* to compute a finite population correction for Taylor series or bootstrap variance estimation. This option is ignored for BRR and jackknife variance estimation.

You can specify the total number of PSUs in either of the following ways:

value specifies a positive number to use for a nonstratified design or for a stratified design that has the same population total in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the population totals for a stratified design that has different population totals in the strata. You must provide the stratum totals in the data set variable named _TOTAL_.

The stratum totals must be positive numbers.

For more information, see the section "Population Totals and Sampling Rates" on page 10258.

If you do not specify the TOTAL= or RATE= option, then the Taylor series or bootstrap variance estimation does not include a finite population correction. You cannot specify both the TOTAL= option and the RATE= option in the same PROC SURVEYPHREG statement.

VARMETHOD=*method* **<(***method-options***)>**
specifies the variance estimation *method*. PROC SURVEYPHREG provides the Taylor series method and balanced repeated replication (BRR), jackknife, and bootstrap replication (resampling) methods.

Table 122.2 summarizes the available *methods* and *method-options*.

Table 122.2 Variance Estimation Options

method	**Variance Estimation Method**	*method-options*
BOOTSTRAP	Bootstrap	CENTER=FULLSAMPLE | REPLICATES DETAILS MH=*number* | *SAS-data-set* OUTWEIGHTS=*SAS-data-set* REPS=*number* SEED=*number*
BRR	Balanced repeated replication	CENTER=FULLSAMPLE | REPLICATES DETAILS FAY <=*value*> HADAMARD=*SAS-data-set* OUTWEIGHTS=*SAS-data-set* PRINTH REPS=*number*
JACKKNIFE	Jackknife	CENTER=FULLSAMPLE | REPLICATES DETAILS OUTJKCOEFS=*SAS-data-set* OUTWEIGHTS=*SAS-data-set*
TAYLOR	Taylor series linearization	None

By default, VARMETHOD=JACKKNIFE if you also specify a REPWEIGHTS statement; otherwise, VARMETHOD=TAYLOR by default.

You can specify the following *methods*:

BOOTSTRAP <(*method-options***)>**
requests variance estimation by the bootstrap method. The bootstrap method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you provide replicate weights by using a REPWEIGHTS statement. For more information, see the section "Bootstrap Method" on page 10266.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

For more information, see the section "Bootstrap Method" on page 10266. By default, CENTER=FULLSAMPLE.

DETAILS

displays the maximum likelihood estimates of model parameters for replicate samples when the replicate parameter estimates are available. A replicate sample might not provide useful parameter estimates (replicate estimates) for reasons such as nonconvergence of the optimization or inestimability of some parameters in that replicate sample.

MH=*value* | **(***values***)** | *SAS-data-set*

specifies the number of PSUs to select for the bootstrap replicate samples. You can provide bootstrap stratum sample sizes m_h by specifying a list of *values* or a *SAS-data-set*. Alternatively, you can provide a single bootstrap sample size *value* to use for all strata or for a nonstratified design. For more information, see the section "Bootstrap Method" on page 10266.

Each bootstrap sample size m_h must be a positive integer and must be less than n_h, which is the total number of PSUs in stratum h. By default, $m_h = n_h - 1$ for a stratified design. For a nonstratified design, the bootstrap sample size *value* must be less than n (the total number of PSUs in the sample). By default, $m = n - 1$ for a nonstratified design.

You can provide the bootstrap sample size by specifying one of the following forms:

MH=*value*

specifies a single bootstrap sample size *value* to use for all strata or for a nonstratified design.

MH=(*values***)**

specifies a list of stratum bootstrap sample size *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. The number of values must not be less than the number of strata in the DATA= input data set.

The order of the stratum sample size values must match the order of the stratum levels in the DATA= input data set. Each stratum sample size value must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

MH=*SAS-data-set*

names a *SAS-data-set* that contains the stratum bootstrap sample sizes. You must provide the sample sizes in a data set variable named _NSIZE_ or SampleSize.

The *SAS-data-set* must contain all stratification variables that you specify in the STRATA statement. It must also contain all stratum levels that appear in the DATA=

input data set. The order of the stratum levels in the *SAS-data-set* must match the order of the levels in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets.

Each value of the _NSIZE_ or SampleSize variable must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* in which to store the replicate weights that PROC SURVEYPHREG creates for bootstrap variance estimation. For information about replicate weights, see the section "Bootstrap Method" on page 10266. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10282.

This *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

REPS=*number*

specifies the *number* of replicates for bootstrap variance estimation, where *number* must be an integer greater than 1. Increasing the number of replicates improves the estimation precision but also increases the computation time. By default, REPS=250.

SEED=*number*

specifies the initial seed for random number generation, where *number* must be a positive integer.

If you do not specify this option or if you specify a *number* that is negative or 0, PROC SURVEYPHREG uses the time of day from the computer's clock to obtain an initial seed.

The seed that is used is displayed in the "Variance Estimation" table.

To reproduce the same bootstrap replicate weights and the same analysis in a subsequent execution of PROC SURVEYPHREG, you can specify the same initial seed that was used in the original analysis.

BRR < (*method-options***) >**

requests variance estimation by balanced repeated replication (BRR). The BRR method requires a stratified sample design with two primary sampling units (PSUs) in each stratum. If you specify the VARMETHOD=BRR option, you must also specify a STRATA statement unless you provide replicate weights with a REPWEIGHTS statement. See the section "Balanced Repeated Replication (BRR) Method" on page 10263 for details.

You can specify the following *method-options* in parentheses after the VARMETHOD=BRR option:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the BRR method. CENTER=FULLSAMPLE is the default, which computes the deviations of the replicate estimates from the full sample estimate. Alternatively, you can specify CENTER=REPLICATES to compute the deviations of the replicate estimates from the average of the replicate estimates. See the section "Balanced Repeated Replication (BRR) Method" on page 10263 for details.

DETAILS

displays the maximum likelihood estimates of model parameters for replicate samples when the replicate parameter estimates are available. A replicate sample might not provide useful parameter estimates (replicate estimates), for reasons such as nonconvergence of the optimization or inestimability of some parameters in that replicate sample.

FAY < =*value* >

requests Fay's method, which is a modification of the BRR method. See the section "Fay's BRR Method" on page 10264 for details.

You can specify the *value* of the Fay coefficient, which is used in converting the original sampling weights to replicate weights. The Fay coefficient must be a nonnegative number less than 1. By default, the value of the Fay coefficient equals 0.5.

HADAMARD=*SAS-data-set*

H=*SAS-data-set*

names a SAS data set that contains the Hadamard matrix for BRR replicate construction. If you do not provide a Hadamard matrix with the HADAMARD= *method-option*, PROC SURVEYPHREG generates an appropriate Hadamard matrix for replicate construction. See the sections "Balanced Repeated Replication (BRR) Method" on page 10263 and "Hadamard Matrix" on page 10265 for details.

If a Hadamard matrix of a given dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a SAS data set in the HADAMARD= *method-option*.

In the HADAMARD= input data set, each variable corresponds to a column of the Hadamard matrix, and each observation corresponds to a row of the matrix. You can use any variable names in the HADAMARD= data set. All values in the data set must equal either 1 or –1. You must ensure that the matrix you provide is indeed a Hadamard matrix—that is, $\mathbf{A}'\mathbf{A} = R\mathbf{I}$, where $\mathbf{A}$ is the Hadamard matrix of dimension R and $\mathbf{I}$ is an identity matrix. PROC SURVEYPHREG does not check the validity of the Hadamard matrix that you provide.

The HADAMARD= input data set must contain at least H variables, where H denotes the number of first-stage strata in your design. If the data set contains more than H variables, PROC SURVEYPHREG uses only the first H variables. Similarly, the HADAMARD= input data set must contain at least H observations.

If you do not specify the REPS= *method-option*, then the number of replicates is equal to the number of observations in the HADAMARD= input data set. If you specify the number of replicates—for example, REPS=*nreps*—then the first *nreps* observations in the HADAMARD= data set are used to construct the replicates.

You can specify the PRINTH *method-option* to display the Hadamard matrix that PROC SURVEYPHREG uses to construct replicates for BRR variance estimation.

OUTWEIGHTS=*SAS-data-set*

names an output SAS data set to store the replicate weights that PROC SURVEYPHREG creates for BRR variance estimation. For more information about replicate weights, see the section "Balanced Repeated Replication (BRR) Method" on page 10263. For more information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10282.

The OUTWEIGHTS= *method-option* is not available when you provide replicate weights by using a REPWEIGHTS statement.

PRINTH

displays the Hadamard matrix that is used to construct replicates for BRR variance estimation. When you provide the Hadamard matrix in the HADAMARD= *method-option*, PROC SURVEYPHREG displays only the rows and columns that are actually used to construct replicates. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 10263 and "Hadamard Matrix" on page 10265.

The PRINTH *method-option* is not available when you provide replicate weights by using a REPWEIGHTS statement, because PROC SURVEYPHREG does not use a Hadamard matrix in this case.

REPS=*number*

specifies the number of replicates for BRR variance estimation. The value of *number* must be an integer greater than 1.

If you do not provide a Hadamard matrix by using the HADAMARD= *method-option*, the number of replicates should be greater than the number of strata and should be a multiple of 4. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10263. If a Hadamard matrix cannot be constructed for the REPS= value that you specify, the value is increased until a Hadamard matrix of that dimension can be constructed. Therefore, it is possible for the actual number of replicates to be larger than the REPS= value that you specify.

If you provide a Hadamard matrix by using the HADAMARD= *method-option*, the value of REPS= must not be greater than the number of rows in the Hadamard matrix. If you provide a Hadamard matrix and do not specify the REPS= *method-option*, the number of replicates equals the number of rows in the Hadamard matrix.

If you do not specify the REPS= or HADAMARD= *method-option* and do not include a REPWEIGHTS statement, the number of replicates equals the smallest multiple of 4 that is greater than the number of strata.

If you provide replicate weights with a REPWEIGHTS statement, the procedure does not use the REPS= *method-option*. With a REPWEIGHTS statement, the number of replicates equals the number of REPWEIGHTS variables.

JACKKNIFE | JK < (*method-options***) >**

requests variance estimation by the delete-1 jackknife method. See the section "Jackknife Method" on page 10267 for details. If you provide replicate weights with a REPWEIGHTS statement, VARMETHOD=JACKKNIFE is the default variance estimation method. The JACKKNIFE method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you provide replicate weights with a REPWEIGHTS statement.

You can specify the following *method-options* in parentheses following VARMETHOD=JACKKNIFE:

CENTER=FULLSAMPLE | REPLICATES
defines how to compute the deviations for the jackknife method. CENTER=FULLSAMPLE is the default, which computes the deviations of the replicate estimates from the full sample estimate. Alternatively, you can specify CENTER=REPLICATES to compute the deviations of the replicate estimates from the average of the replicate estimates. See the section "Jackknife Method" on page 10267 for details.

DETAILS
displays the maximum likelihood estimates of model parameters for replicate samples when the replicate parameter estimates are available. A replicate sample might not provide useful parameter estimates (replicate estimates), for reasons such as nonconvergence of the optimization or inestimability of some parameters in that replicate sample.

OUTJKCOEFS=*SAS-data-set*
names an output SAS data set that contains jackknife coefficients. See the section "Jackknife Coefficients Output Data Set" on page 10283 for more details about the contents of the OUTJKCOEFS= data set.

OUTWEIGHTS=*SAS-data-set*
names an output SAS data set that contains replicate weights. See the section "Jackknife Method" on page 10267 for more information about replicate weights. See the section "Replicate Weights Output Data Set" on page 10282 for more details about the contents of the OUTWEIGHTS= data set.

The OUTWEIGHTS= *method-option* is not available when you provide replicate weights with a REPWEIGHTS statement.

TAYLOR
requests Taylor series variance estimation. This is the default method if you do not specify the VARMETHOD= option or a REPWEIGHTS statement. See the section "Taylor Series Linearization" on page 10262 for more information.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SURVEYPHREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SURVEYPHREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the

data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.

- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Note that using a BY statement provides completely separate analyses of the BY groups. It does not provide a domain (subpopulation) analysis, where the number of sampling units in the subpopulation is not known at the time the survey is designed. For such an analysis use the DOMAIN statement.

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < (*options*) > . . . < *variable* < (*options*) > > < / *options* > **;**

The CLASS statement names the classification variables to be used as explanatory variables in the analysis.

The CLASS statement must precede the MODEL statement. Most *options* can be specified either as individual variable *options* or as global *options*. You can specify *options* for each variable by enclosing the options in parentheses after the variable name. You can also specify global *options* for the CLASS statement by placing the *options* after a slash (/). Global *options* are applied to all the variables specified in the CLASS statement. If you specify more than one CLASS statement, the global *options* specified in any one CLASS statement apply to all CLASS statements. However, individual CLASS variable *options* override the global *options*. The following *options* are available:

DESCENDING
DESC

reverses the sort order of the classification variable. If both the DESCENDING and ORDER= options are specified, PROC SURVEYPHREG orders the categories according to the ORDER= option and then reverses that order.

MISSING

treats missing values (".", ._, .A, . . . , .Z for numeric variables and blanks for character variables) as valid values for the CLASS variable.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of classification variables. This ordering determines which parameters in the model correspond to each level in the data, so the ORDER= option can be useful when you use the CONTRAST statement. By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent. When ORDER=FORMATTED is in effect for numeric variables for which you have supplied no explicit format, the levels are ordered by their internal values.

The following table shows how PROC SURVEYPHREG interprets values of the ORDER= option.

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted values, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) values
FREQ	Descending frequency count; levels with more observations come earlier in the order
INTERNAL	Unformatted value

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in *SAS Programmers Guide: Essentials*.

PARAM=*keyword*

specifies the parameterization method for the classification variable or variables. If the PARAM= option is not specified together with any individual CLASS variable, then by default, PARAM=GLM. Otherwise, the default is PARAM=EFFECT. You can specify any of the *keywords* shown in the following table.

Design matrix columns are created from CLASS variables according to the corresponding coding schemes.

Value of PARAM=	Coding
EFFECT	Effect coding
GLM	Less-than-full-rank reference cell coding (this *keyword* can be used only in a global option)
ORDINAL **THERMOMETER**	Cumulative parameterization for an ordinal CLASS variable
POLYNOMIAL **POLY**	Polynomial coding
REFERENCE **REF**	Reference cell coding
ORTHEFFECT	Orthogonalizes PARAM=EFFECT coding
ORTHORDINAL **ORTHOTHERM**	Orthogonalizes PARAM=ORDINAL coding
ORTHPOLY	Orthogonalizes PARAM=POLYNOMIAL coding
ORTHREF	Orthogonalizes PARAM=REFERENCE coding

All parameterizations are full rank, except for the GLM parameterization. The REF= option in the CLASS statement determines the reference level for EFFECT and REFERENCE coding and for their orthogonal parameterizations. It also indirectly determines the reference level for a singular GLM parameterization through the order of levels.

If a PARAM= option is specified as a variable option for some variables, then any variables for which PARAM= is not specified use either the EFFECT parameterization if the global PARAM= option is not specified, or the full-rank parameterization indicated in the global PARAM= option if specified. If

the global PARAM=GLM option is specified and PARAM= is also specified for some variables, GLM parameterization is used for all variables.

If PARAM=ORTHPOLY or PARAM=POLY and the classification variable is numeric, then the ORDER= option in the CLASS statement is ignored, and the internal unformatted values are used. For more information, see the section "Other Parameterizations" on page 416 in Chapter 20, "Shared Concepts and Topics."

REF=*'level' | keyword*

specifies the reference level for PARAM=EFFECT, PARAM=REFERENCE, and their orthogonalizations. For PARAM=GLM, the REF= option specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the linear estimates with a singular parameterization.

For an individual variable REF= option (but not for a global REF= option), you can specify the *level* of the variable to use as the reference level. Specify the formatted value of the variable if a format is assigned. For a global or individual variable REF= option, you can use one of the following *keywords*:

FIRST designates the first ordered level as reference.

LAST designates the last ordered level as reference.

By default, REF=LAST.

TRUNCATE< =*n* >

specifies the length *n* of CLASS variable values to use in determining CLASS variable levels. The default is to use the full formatted length of the CLASS variable. If you specify TRUNCATE without the length *n*, the first 16 characters of the formatted values are used. The TRUNCATE option is available only as a global option.

CLUSTER Statement

CLUSTER *variables* **;**

The CLUSTER statement names variables that identify the first-stage clusters in a clustered sample design. First-stage clusters are also known as primary sampling units (PSUs). The combinations of categories of CLUSTER variables define the clusters in the sample. If a STRATA statement is specified, clusters are nested within strata.

If your sample design has clustering at multiple stages, you should specify only the first-stage clusters (PSUs) in the CLUSTER statement. For more information, see the section "Specifying the Sample Design" on page 10257.

If you provide replicate weights for replication variance estimation by specifying a REPWEIGHTS statement, you do not need to specify a CLUSTER statement.

The CLUSTER *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric, but the procedure treats them as categorical variables. The formatted values of the CLUSTER variables determine the CLUSTER variable levels. Thus, you can use formats to group values into levels. For more information, see the discussion of the FORMAT procedure in the *Base SAS*

Procedures Guide and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

You can use multiple CLUSTER statements to specify CLUSTER variables. The procedure uses variables from all CLUSTER statements to create clusters. Cluster variables must not occur in the CLASS statement.

DOMAIN Statement

DOMAIN *variable* < ('*formatted-level-value*' . . . '*formatted-level-value*')> < *variable* < ('*formatted-level-value*' . . . '*formatted-level-value*')>∗*variable* < ('*formatted-level-value*' . . . '*formatted-level-value*')> > **;**

The DOMAIN statement requests analysis for domains (subpopulations), in addition to analysis for the entire study population. The DOMAIN statement names the variables that identify domains, which are called domain variables.

It is common practice to compute statistics for domains. The formation of these domains might not be known at the design stage. Therefore, the sample sizes for the domains are often random. Use a DOMAIN statement to incorporate this variability into the variance estimation.

Note that a DOMAIN statement is different from a BY statement. In a BY statement, you treat the sample sizes as fixed in each subpopulation, and you perform analysis within each BY group independently.

Use the DOMAIN statement on the entire data set to perform a domain analysis. Creating a new data set from a single domain and analyzing that with PROC SURVEYPHREG can yield inappropriate estimates of variance for domain statistics.

A domain variable can be either character or numeric. The procedure treats domain variables as categorical variables. If a variable appears by itself in a DOMAIN statement, each level of this variable determines a domain in the study population. If two or more variables are joined by asterisks (*), then every possible combination of levels of these variables determines a domain. The procedure performs a descriptive analysis within each domain that is defined by the domain variables. Domain variables must not occur in the CLASS statement.

The formatted values of the domain variables determine the categorical variable levels. Thus, you can use formats to group values into levels. For more information, see the FORMAT procedure in the *Base SAS Procedures Guide*.

By default, the SURVEYPHREG procedure performs analyses for all levels of domains that are formed by the variables in the DOMAIN statement. Optionally, you can specify domain analyses for particular levels of a DOMAIN variable by listing quoted *formatted-level-values* in parentheses after the variable name. You must enclose each *formatted-level-value* in single or double quotation marks. You can specify one or more levels of each variable; when you specify more than one level, separate the levels by a space or a comma. The following example requests domain analysis only for females within each race category:

```
domain Race*Gender(''Female'');
```

The following example requests domain analyses only for white and Asian races, and separate domain analyses for both genders:

```
domain Race('White','Asian') Gender;
```

If a domain variable appears more than once in any domain cross-classification but the specified levels for that domain variable are not the same, then PROC SURVEYPHREG includes all specified levels of that variable in the domain cross-classification.

In the following example, two different levels for Race are specified in two DOMAIN statements:

```
domain Race('White')*Gender;
domain Race('Asian')*Gender;
```

Thus, the preceding specification is equivalent to the following:

```
domain Race('Asian' 'White')*Gender;
```

However, if a domain variable appears more than once in cross-classifications but the levels for that domain variable are not specified in all cross-classifications, then PROC SURVEYPHREG includes only the specified levels.

In the following example, a level for Gender is specified in the first DOMAIN statement but no levels for Gender are specified in the second DOMAIN statement:

```
domain Race('White')*Gender('Female');
domain Race('Asian')*Gender;
```

Thus, the preceding specification is equivalent to the following:

```
domain Race('White' 'Asian')*Gender('Female');
```

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 122.3 summarizes the *options* available in the ESTIMATE statement.

Table 122.3 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference

Table 122.3 *continued*

Option	Description
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected *p*-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the **L** matrix
JOINT	Produces a joint *F* or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement" on page 475 in Chapter 20, "Shared Concepts and Topics."

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a numeric *variable* that provides a frequency for each observation in the input data set. PROC SURVEYPHREG treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If not an integer, the frequency value is truncated to an integer. If the frequency value is missing, the observation is not used in the analysis. The FREQ statement allows one frequency variable.

If you use the FREQ statement and request the jackknife or BRR variance estimator by specifying the VARMETHOD=JACKKNIFE or VARMETHOD=BRR option in the PROC SURVEYPHREG statement, then you must identify the primary sampling units with a CLUSTER statement unless you also provide replicate weights with a REPWEIGHTS statement.

HAZARDRATIO Statement

HAZARDRATIO <'*label*'> *variable* <*/ options*> **;**

The HAZARDRATIO statement enables you to request hazard ratios for any variable in the model at customized settings. For example, if the model contains the interaction of a CLASS variable A and a continuous variable X, the following specification displays a table of hazard ratios that compares the hazards of each pair of levels of A at X=3:

```
hazardratio A / at (X=3);
```

The HAZARDRATIO statement identifies the variable whose hazard ratios are to be evaluated. If the variable is a continuous variable, the hazard ratio compares the hazards for a specified change (by default, an increase of 1 unit) in the variable. For a CLASS variable, a hazard ratio compares the hazards of two levels of the variable. You can specify more than one HAZARDRATIO statement, and an optional label (specified as a quoted string) helps identify the output.

Table 122.4 summarizes the *options* available in the HAZARDRATIO statement.

Table 122.4 HAZARDRATIO Statement Options

Option	Description
ALPHA=	Specifies the alpha level
AT	Specifies the variables that interact with the variable of interest
DIFF=	Specifies which differences to compute
E	Displays the coefficients for the log-hazard ratio
UNITS=	Specifies the units of change

You can specify the following *options* in the HAZARDRATIO statement:

ALPHA=*number*

specifies the alpha level of the interval estimates for the hazard ratios, where *number* must be between 0 and 1. The default is the value of the ALPHA= option in the PROC SURVEYPHREG statement, or 0.05 if that option is not specified.

AT (*variable***=ALL | REF |** *list* **< . . .** *variable***=ALL | REF |** *list* **>)**

specifies the variables that interact with the variable of interest and the corresponding values of the interacting variables. If the interacting variable is continuous and a numeric *list* is specified after the equal sign, hazard ratios are computed for each value in the list. If the interacting variable is a CLASS variable, you can specify, after the equal sign, a list of quoted strings that correspond to various levels of the CLASS variable, or you can specify the keyword ALL or REF. Hazard ratios are computed at each value of the list if a *list* is specified, or at each level of the interacting variable if ALL is specified, or at the reference level of the interacting variable if REF is specified.

If this option is not specified, PROC SURVEYPHREG finds all the variables that interact with the variable of interest. If an interacting variable is a CLASS variable, *variable*=ALL is the default; if the interacting variable is continuous, *variable*=*m* is the default, where *m* is the weighted average of the observed values of the continuous variable.

Suppose the model contains two interactions: an interaction A*B of CLASS variables A and B, and another interaction A*X of A with a continuous variable X. If 3.5 is the weighted average of the values of X, the following two HAZARDRATIO statements are equivalent:

```
hazardratio A;
hazardratio A / at (B=ALL X=3.5);
```

DIFF=*diff-request*

specifies which differences to consider for the level comparisons of a CLASS variable. This option is ignored in the estimation of hazard ratios for a continuous variable. You can specify the following *diff-requests*:

DISTINCT

DISTINCTPAIRS

requests all comparisons of only the distinct combinations of pairs.

PAIRWISE

requests all possible pairwise comparisons of levels.

REFERENCE

REF

requests comparisons between the reference level and all other levels of the CLASS variable.

For example, let A be a CLASS variable that has three levels (A1, A2, and A3), where A3 is specified as the reference level. The following table depicts the hazard ratios that are displayed for the three alternatives of the DIFF= option.

	Hazard Ratios Displayed					
DIFF=*option*	**A1 vs A2**	**A2 vs A1**	**A1 vs A3**	**A3 vs A1**	**A2 vs A3**	**A3 vs A2**
DISTINCT	√		√		√	
PAIRWISE	√	√	√	√	√	√
REF			√		√	

By default, DIFF=DISTINCT.

E

displays the vector $\mathbf{l}$ of linear coefficients such that $\mathbf{l}'\boldsymbol{\beta}$ is the log-hazard ratio, where $\boldsymbol{\beta}$ is the vector of regression coefficients.

UNITS=*value*

specifies the units of change in the continuous explanatory variable for which the customized hazard ratio is estimated. This option is ignored in the computation of the hazard ratios for a CLASS variable. By default, UNITS=1.

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted population margins*—that is, they estimate the marginal means over a balanced population. In a sense, LS-means are to unbalanced designs as class and subclass arithmetic means are to balanced designs.

Table 122.5 summarizes the *options* available in the LSMEANS statement.

Table 122.5 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers

Table 122.5 *continued*

Option	Description
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For more information about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 122.6 summarizes the *options* available in the LSMESTIMATE statement.

Table 122.6 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference

Table 122.6 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint F or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MARGINS Statement

MARGINS *model-effects* < / *options* > **;**

The MARGINS statement computes and compares predictive margins of fixed effects. The predictive margin for a specific level (group) of a fixed effect represents the average predicted response if all the observations in the data set were in that group (Lane and Nelder 1982; Chang, Gelman, and Pagano 1982). You can compute predictive margins for any effect in the MODEL statement that involves only classification variables.

Table 122.7 summarizes the *options* available in the MARGINS statement.

Table 122.7 MARGINS Statement Options

Option	Description
Construction and Computation of Predictive Margins	
AT	Modifies the covariate value in computing predictive margins
DIFF	Computes differences of predictive margins
SLICEBY=	Partitions tests of interaction effects
SLICEDIFF	Computes differences of sliced predictive margins and determines the types of differences
Degrees of Freedom and *p*-Values	
ADJUST=	Specifies the method of multiple comparison adjustment of predictive margin differences
ALPHA=α	Specifies the confidence level $(1-\alpha)$
DF=	Assigns a specific value to degrees of freedom for t tests and confidence limits
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion
Statistical Output	
CL	Constructs confidence limits for predictive margins and/or predictive margin differences

For more information about the syntax of the MARGINS statement, see the section "MARGINS Statement" on page 525 in Chapter 20, "Shared Concepts and Topics."

NOTE: If your model has classification variables, then the MARGINS statement is allowed only if you also specify the PARAM=GLM option.

MODEL Statement

MODEL *response* < ∗ *censor* (*list*) > **=** *effects* < / *options* > **;**

MODEL (*t1*, *t2*) < ∗ *censor*(*list*) > **=** *effects* < / *options* > **;**

The MODEL statement identifies the variables to be used as the failure time variables, the optional censoring variable, and the explanatory effects, including covariates, main effects, and interactions. For more information about explanatory effects, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure." A note of caution: specifying the effect T*A in the MODEL statement, where T is the time variable and A is a CLASS variable, does not make the effect time-dependent.

You must specify exactly one MODEL statement. Specify either of two forms of MODEL syntax: the first form allows one time variable, and the second form allows two time variables for the counting process style of input. For more information on the counting process style of input, see the section "Counting Process Style of Input" on page 10253.

For the first form of the MODEL statement, the name of the failure time variable (*response*) precedes the equal sign. This variable can optionally be followed by an asterisk, the name of the censoring variable, and a *list* of censoring values (separated by blanks or commas if there is more than one) enclosed in parentheses.

If the censoring variable takes on one of these values, the corresponding failure time is considered to be censored. The variables following the equal sign (*effects*) are the explanatory variables (sometimes called independent variables or covariates) for the model.

Instead of using a single failure time variable, the second form of the MODEL statement identifies a pair of failure time variables. Their names are enclosed in parentheses, and they signify the endpoints of a semiclosed interval $(t1, t2]$ during which the subject is at risk. If the censoring variable takes on one of the censoring values, the time $t2$ is considered to be censored.

The censoring variable must be numeric. The failure time variable must contain nonnegative values. Any observation that has a negative failure time is excluded from the analysis, as is any observation that has a missing value for any of the variables listed in the MODEL statement. For more information, see the section "Missing Values" on page 10259. Failure time variables that have a SAS date format are not recommended because the dates might be translated into negative numbers and consequently the corresponding observation would be discarded.

Table 122.8 summarizes the *options* available in the MODEL statement, which can be specified after a slash (/).

Table 122.8 MODEL Statement Options

Option	Description
ALPHA=	Specifies α for the $100(1-\alpha)\%$ confidence limits
CLPARM	Computes confidence limits for regression parameters
COVB	Displays covariance matrix
DF=	Specifies the denominator degrees of freedom
ENTRYTIME=	Specifies the delayed entry time variable
FIRTH	Specifies Firth's penalized likelihood method
HESS	Displays the Hessian matrix
INVHESS	Displays the inverse of the Hessian matrix
RISKLIMITS	Computes confidence limits for the exponentials of the regression parameters
SERATIO=	Computes the ratio of two standard errors for the regression coefficients
SINGULAR=	Specifies tolerance for testing singularity
TIES=	Specifies the method of handling ties in failure times
VADJUST=	Specifies a variance adjustment factor
VARRATIO=	Computes the ratio of two variances for the regression coefficients

ALPHA=α

sets the level of the confidence limits for the estimated regression parameters and the hazard ratios. The value of *alpha* must be between 0 and 1, and the default is 0.05. A confidence level of α produces $100(1-\alpha)\%$ confidence limits. The default of ALPHA=0.05 produces 95% confidence limits.

The ALPHA= option has no effect unless you also specify the CLPARM or RISKLIMITS option.

CLPARM

produces confidence limits for regression parameters of Cox proportional hazards models. You can specify the confidence coefficient by using the ALPHA= option. Classification main effects that use parameterizations other than REF, EFFECT, or GLM are ignored. For more information, see the section "Confidence Intervals" on page 10276.

COVB

displays the estimated covariance matrix of the parameter estimates.

DF=*value* | *keyword* **<(***value***)>**

specifies the denominator degrees of freedom for hypothesis tests, specifies the degrees of freedom for confidence limits, and requests adjustments to the Wald test statistics. If you specify a *value*, it must be a nonnegative number.

In the description that follows, d denotes the usual degrees of freedom computed from the survey data by using the number of strata, clusters, or replicate weights. For more information, see the section "Degrees of Freedom" on page 10270.

By default, DF=PARMADJ when you use the Taylor series linearized variance estimator, and DF=DESIGN when you use the replication variance estimator. Alternatively, you can specify a nonnegative *value* for the degrees of freedom, or you can specify one of the following *keywords*:

ALLREPS

computes the denominator degrees of freedom for replication methods by using the total number of replicate samples. By default, PROC SURVEYPHREG computes the denominator degrees of freedom based on the number of replicate samples that are used. Some replicate samples might not be usable, in the sense that they cannot be used for variance estimation because of factors such as inestimability or nonconvergence. These replicate samples are not accounted for in the denominator degrees of freedom unless you specify DF=ALLREPS. For more information, see the section "Degrees of Freedom" on page 10270.

DESIGN

computes the denominator degrees of freedom as d. When you specify DF=DESIGN, the corresponding Wald F statistics do not account for the number of parameters in the model. This option is useful if you do not want to apply the adjustment described in Korn and Graubard (1999, p. 93). For more information, see the section "Testing the Global Null Hypothesis" on page 10274.

DESIGN (*value***)**

computes the denominator degrees of freedom as *value*. When you specify DF=DESIGN (*value*), the corresponding Wald F statistics do not account for the number of parameters in the model. This option is useful if you do not want to apply the adjustment described in Korn and Graubard (1999, p. 93) and you want to specify the denominator degrees of freedom. You might want to specify a denominator degrees of freedom other than d for reasons such as missing values or domain estimation for relatively small domains. For more information, see the section "Testing the Global Null Hypothesis" on page 10274.

DESIGNADJ

computes the denominator degrees of freedom as d. When you specify DF=DESIGNADJ, the corresponding Wald F statistics account for the number of parameters in the model. This option is useful if you are fitting a model that has many parameters relative to d but you want to use d as the denominator degrees of freedom. For more information, see the section "Testing the Global Null Hypothesis" on page 10274.

NONE

specifies the denominator degrees of freedom to be infinite. This option is useful if you want to compute chi-square tests and normal confidence intervals. For more information, see the section "Testing the Global Null Hypothesis" on page 10274.

PARMADJ

computes the denominator degrees of freedom as d minus the number of nonsingular parameters plus 1. When you specify DF=PARMADJ, the corresponding Wald F statistics account for the number of parameters in the model. This option is useful if you are fitting a model that has many parameters relative to d. For more information, see the section "Testing the Global Null Hypothesis" on page 10274.

PARMADJ (*value*)

computes the denominator degrees of freedom as *value*. When you specify DF=PARMADJ (*value*), the corresponding Wald F statistics account for the number of parameters in the model. This option is useful if you are fitting a model that has many parameters relative to d and you want to specify the denominator degrees of freedom. You might want to specify the denominator degrees of freedom for reasons such as missing values or domain estimation for relatively small domains. For more information, see the section "Testing the Global Null Hypothesis" on page 10274.

ENTRYTIME=*variable*

ENTRY=*variable*

specifies the name of the variable that represents the left-truncation time. This option has no effect when the counting process style of input is specified. For more information, see the section "Left-Truncation of Failure Times" on page 10254.

FIRTH

performs Firth's penalized maximum likelihood estimation (Mukhopadhyay 2020; Heinze and Schemper 2001; Firth 1993). This method is useful when the likelihood is monotone—that is, the likelihood converges to a finite value, but at least one estimate diverges to infinity. This option is available only for the Breslow likelihood. When you specify this option, the likelihood ratio statistics are computed using the unadjusted likelihoods, and only the Wald test for the overall null hypothesis is available. For more information, see the section "Firth's Modification for Maximum Likelihood Estimation" on page 10255.

HESS

displays the last evaluation of the Hessian matrix.

INVHESS

displays the inverse of the Hessian matrix that is evaluated at the estimated regression parameters.

RISKLIMITS

RL

produces confidence limits for hazard ratios and related quantities. For more information, see the section "Hazard Ratios" on page 10277. You can specify the confidence coefficient by using the ALPHA= option. You must take great care with any interpretation of the estimates and their confidence limits if interaction effects are involved in the model or if parameterizations other than REF, EFFECT, or GLM are used.

SERATIO=ALL | MODEL | IND | SRSWOR | SRSWR

computes the ratio of two standard errors for the regression parameters. The standard error in the numerator uses the complete design information that you specify. You can specify the following options to compute different standard errors for the denominator:

ALL

requests IND, MODEL, and either SRSWR or SRSWOR standard error ratios. If you specify the RATE= or the TOTAL= option in the PROC SURVEYPHREG statement, then SRSWOR standard error ratios are computed; otherwise, SRSWR standard error ratios are computed.

IND

computes the standard errors in the denominator by ignoring stratification and clustering. For more information, see the section "Variance Ratios and Standard Error Ratios" on page 10272.

MODEL

computes the standard errors in the denominator as the square root of the diagonals of the inverse Hessian matrix evaluated at the estimated regression parameters. For more information, see the section "Variance Ratios and Standard Error Ratios" on page 10272.

SRSWOR

computes the standard errors in the denominator as the square root of the diagonals of a scaled inverse Hessian matrix evaluated at the estimated regression parameters. If you specify the RATE= or the TOTAL= option in the PROC SURVEYPHREG statement, then the scaling factor also includes the sampling fractions. For more information, see the section "Variance Ratios and Standard Error Ratios" on page 10272.

SRSWR

computes the standard errors in the denominator as the square root of the diagonals of a scaled inverse Hessian matrix evaluated at the estimated regression parameters. For more information, see the section "Variance Ratios and Standard Error Ratios" on page 10272.

SINGULAR=*value*

specifies the singularity criterion for determining linear dependencies in the set of explanatory variables. The default value is 10^{-12}.

TIES=*method*

specifies how to handle ties in the failure time. You can specify the following *methods*:

BRESLOW

uses the approximate partial likelihood of Breslow (1974).

EFRON

uses the approximate partial likelihood of Efron (1977).

If there are no ties, both methods result in the same likelihood and yield identical estimates. By default, TIES=BRESLOW, which is the most efficient method when there are no ties.

VADJUST=DF | PARMADJ | NONE | AVGREPSS

specifies variance adjustment factors. You can specify the following keywords:

DF

PARMADJ

requests the degrees-of-freedom adjustment $(n-1)/(n-p)$ in the computation of the matrix $\mathbf{G}$ for the Taylor series linearization variance estimation.

NONE

excludes the degrees-of-freedom adjustment $(n-1)/(n-p)$ from the computation of the matrix $\mathbf{G}$ for the Taylor series linearization variance estimation. By default, VADJUST=NONE.

AVGREPSS

use the average sum of squares from all the usable replicate samples for the unusable replicates. This option is applicable only for the jackknife replication method. VADJUST=AVGREPSS multiplies the default jackknife variance estimator by the factor R/R_a, where R_a is the number of usable replicates and R is the total number of replicates. For more information, see the section "Variance Adjustment Factors" on page 10271.

VARRATIO=ALL | MODEL | IND | SRSWOR | SRSWR

computes the ratio of two variances for the regression parameters. The variance in the numerator uses the complete design information. You can specify the following options to compute different variances for the denominator:

ALL

requests IND, MODEL, and either SRSWR or SRSWOR variance ratios. If you specify the RATE= or the TOTAL= option in the PROC SURVEYPHREG statement, then SRSWOR variance ratios are computed; otherwise, SRSWR variance ratios are computed.

IND

computes the variances in the denominator by ignoring stratification and clustering. For more information, see the section "Variance Ratios and Standard Error Ratios" on page 10272.

MODEL

computes the variances in the denominator as the diagonals of the inverse Hessian matrix evaluated at the estimated regression parameters. For more information, see the section "Variance Ratios and Standard Error Ratios" on page 10272.

SRSWOR
computes the variances in the denominator as the diagonals of a scaled inverse Hessian matrix evaluated at the estimated regression parameters. If you specify the RATE= or the TOTAL= option in the PROC SURVEYPHREG statement, then the scaling factor also includes the sampling fractions. For more information, see the section "Variance Ratios and Standard Error Ratios" on page 10272.

SRSWR
computes the variances in the denominator as the diagonals of a scaled inverse Hessian matrix evaluated at the estimated regression parameters. For more information, see the section "Variance Ratios and Standard Error Ratios" on page 10272.

NLOPTIONS Statement

NLOPTIONS < *options* > **;**

The NLOPTIONS statement specifies details of the nonlinear optimization used by PROC SURVEYPHREG to maximize the log likelihood function. By default, the procedure uses the Newton-Raphson optimization technique. For more information about the NLOPTIONS statement, see the section "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics."

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *keyword*=*name* . . . *keyword*=*name* > < / *options* > **;**

The OUTPUT statement creates a new SAS data set that contains statistics that are calculated for each observation unit. These statistics can include the estimated linear predictor ($\mathbf{z}'_j\hat{\boldsymbol{\beta}}$) and its standard error, residuals, and influence statistics. In addition, this data set includes all the variables from the DATA= input data set.

Only score residuals are available in the OUTPUT data set if the model contains a time-dependent variable that is defined by means of programming statements.

The following list explains specifications in the OUTPUT statement:

OUT=*SAS-data-set*
names the output data set. If you omit the OUT= option, the OUTPUT data set is named by using the DATA*n* convention. See the section "OUT= Data Set for the OUTPUT Statement" on page 10282 for more information.

keyword=name
specifies the statistics to include in the OUTPUT data set and names the new variables that contain the statistics. Specify a *keyword* for each desired statistic (see the following list of *keywords*), and optionally an equal sign with either a variable or a list of variables in parentheses to contain the statistics. If you specify a *keyword* without a variable name, then the procedure uses default names. The *keywords* that accept a list of variables are RESSCH, RESSCO, and WTRESSCH. For these *keywords*, you can specify as many names in *name* as the number of explanatory variables in the MODEL statement. If you specify k names and k is less than the total number of explanatory variables,

only the first k names are taken from the list; the procedure assigns default names for the rest of the statistics. The *keywords* and the corresponding statistics are as follows:

ATRISK
specifies the number of subjects at risk at the observation time τ_j (or at the right endpoint of the at-risk interval when a counting-process specification is used in the MODEL statement, as described in the section "Counting Process Style of Input" on page 10253).

RESDEV
specifies the deviance residual $\hat{D}_j$. This is a transform of the martingale residual to achieve a more symmetric distribution.

RESMART
specifies the martingale residual $\hat{M}_j$. The residual at the observation time τ_j can be interpreted as the difference over $[0, \tau_j]$ in the observed number of events minus the expected number of events given by the model.

RESSCH
specifies the Schoenfeld residuals. These residuals are useful in assessing the proportional hazards assumption.

RESSCO
specifies the score residuals. These residuals are a decomposition of the first partial derivative of the log likelihood. They can be used to assess the leverage that is exerted by each subject in the parameter estimation. They are also useful in constructing design-based variance estimators.

STDXBETA
specifies the standard error of the estimated linear predictor, $\sqrt{\mathbf{z}_j' \hat{\mathbf{V}}(\hat{\boldsymbol{\beta}}|F)\mathbf{z}_j}$.

WTATRISK
specifies the weighted number of subjects at risk at the observation time τ_j (or at the right endpoint of the at-risk interval when a counting-process specification is used in the MODEL statement, as described in the section "Counting Process Style of Input" on page 10253).

XBETA
specifies the estimate of the linear predictor, $\mathbf{z}_j' \hat{\boldsymbol{\beta}}$.

Programming Statements

Programming statements are used to create or modify the values of the explanatory variables in the MODEL statement. They are especially useful in fitting models with time-dependent explanatory variables. Programming statements can also be used to create explanatory variables that are not time-dependent. PROC SURVEYPHREG programming statements cannot be used to create or modify the values of the response variable, the censoring variable, the frequency variable, the WEIGHT variable, the CLASS variables, the STRATA variables, the CLUSTER variables, or the DOMAIN variables. You cannot use programming statements in PROC SURVEYPHREG to modify the observed values of the explanatory variables that are available in the input data set.

The following DATA step statements are available in PROC SURVEYPHREG:

ABORT;
ARRAY *arrayname* < **[** *dimensions* **]** > < **$** > < *variables-and-constants* >**;**
CALL *name* < **(***expression* < **,** *expression* . . . >**)** >**;**
DELETE;
DO < *variable* **=** *expression* < **TO** *expression* > < **BY** *expression* > >
< **,** *expression* < **TO** *expression* > < **BY** *expression* > > . . .
< **WHILE** *expression* > < **UNTIL** *expression* >**;**
END;
GOTO *statement-label***;**
IF *expression***;**
IF *expression* **THEN** *program-statement***;**
ELSE *program-statement***;**
variable **=** *expression***;**
variable **+** *expression***;**
LINK *statement-label***;**
PUT < *variable* > < **=** > . . . **;**
RETURN;
SELECT < **(***expression***)** >**;**
STOP;
SUBSTR(*variable***,** *index***,** *length***)=** *expression***;**
WHEN (*expression***)***program-statement***;**
OTHERWISE *program-statement***;**

By default, the PUT statement in PROC SURVEYPHREG writes results to the output instead of the log. If you want the results of the PUT statements to go to the log, add the following statement before the PUT statement:

```
FILE LOG;
```

DATA step functions are also available. Use these programming statements the same way you use them in the DATA step. For detailed information, see the *SAS Functions and CALL Routines: Reference*.

Consider the following example of using programming statements in PROC SURVEYPHREG. Suppose blood pressure is measured at multiple times during the course of a study that investigates the effect of blood pressure on some survival time. By treating the blood pressure as a time-dependent explanatory variable, you can use the value of the most recent blood pressure at each specific point of time in the modeling process rather than using the initial blood pressure or the final blood pressure. The values of the following variables are recorded for each patient, if they are available. Otherwise, the variables contain missing values.

Time	survival time
Censor	censoring indicator (with 0 as the censoring value)
BP0	blood pressure on entry to the study
T1	time 1
BP1	blood pressure at T1
T2	time 2
BP2	blood pressure at T2
WT	design weight

PSU identification of primary sampling units

The following programming statements create a variable BP. At each time T, the value of BP is the blood pressure reading for that time, if available. Otherwise, it is the last blood pressure reading.

```
proc surveyphreg;
   weight WT;
   model Time*Censor(0)=BP;
   cluster PSU;
   BP = BP0;
   if Time>=T1 and T1^=. then BP=BP1;
   if Time>=T2 and T2^=. then BP=BP2;
run;
```

REPWEIGHTS Statement

REPWEIGHTS *variables* < / *options* > **;**

The REPWEIGHTS statement names *variables* that provide replicate weights for replication variance estimation, which you request with the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option in the PROC SURVEYPHREG statement. If you do not provide a REPWEIGHTS statement for VARMETHOD=BRR or VARMETHOD=JACKKNIFE, then PROC SURVEYPHREG constructs replicate weights for the analysis. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 10263 and "Jackknife Method" on page 10267. For VARMETHOD=BOOTSTRAP, you must specify the REPWEIGHTS statement to provide replicate weights. For more information, see the section "Bootstrap Method" on page 10266.

Each REPWEIGHTS variable should contain the weights for a single replicate, and the number of replicates equals the number of REPWEIGHTS variables. The REPWEIGHTS variables must be numeric, and the variable values must be nonnegative numbers.

If you provide replicate weights with a REPWEIGHTS statement, you do not need to specify a CLUSTER or STRATA statement. If you use a REPWEIGHTS statement and do not specify the VARMETHOD= option in the PROC SURVEYPHREG statement, the procedure uses VARMETHOD=JACKKNIFE by default.

If you specify a REPWEIGHTS statement but do not include a WEIGHT statement, PROC SURVEYPHREG uses the average of each observation's replicate weights as the observation's weight.

You can specify the following *options* in the REPWEIGHTS statement after a slash (/):

JKCOEFS=*jackknife-coefficient-specification*

specifies jackknife coefficients for VARMETHOD=JACKKNIFE. The default value for the jackknife coefficient is $(R-1)/R$, where R is the total number of replicates. You can specify an alternative value with one of the following three forms:

JKCOEFS=*value*

specifies a single jackknife coefficient for all replicates. The coefficient *value* must be a nonnegative number.

JKCOEFS=(*values*)
specifies jackknife coefficients for VARMETHOD=JACKKNIFE, where each coefficient corresponds to an individual replicate identified by a REPWEIGHTS variable. You can separate *values* with blanks or commas. The coefficient *values* must be nonnegative numbers. The number of *values* must equal the number of replicate weight variables named in the REPWEIGHTS statement. List these values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

JKCOEFS=*SAS-data-set*
names a SAS data set that contains the jackknife coefficients for VARMETHOD=JACKKNIFE. You provide the jackknife coefficients in the JKCOEFS= data set variable JKCoefficient. Each coefficient value must be a nonnegative number. The observations in the JKCOEFS= data set should correspond to the replicates that are identified by the REPWEIGHTS variables. Arrange the coefficients or observations in the JKCOEFS= data set in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement. The number of observations in the JKCOEFS= data set must not be less than the number of REPWEIGHTS variables.

See the section "Jackknife Method" on page 10267 for details about jackknife coefficients.

REPCOEFS=*replication-coefficient-specification*
specifies replicate coefficients for replication methods. When you specify VARMETHOD=JACKKNIFE, the default value for the replicate coefficient is $(R-1)/R$, where R is the total number of replicates. When you specify VARMETHOD=BOOTSTRAP or VARMETHOD=BRR, the default value for the replicate coefficient is $1/R$.

For VARMETHOD=BOOTSTRAP or VARMETHOD=JACKKNIFE, you can specify one of the following three *replication-coefficient-specifications*:

REPCOEFS=*value*
specifies a single replicate coefficient for all replicates, where *value* must be a nonnegative number.

REPCOEFS=(*values*)
specifies a list of replicate coefficients, where each coefficient corresponds to an individual replicate that is identified by a REPWEIGHTS variable. You can separate *values* with blanks or commas. The coefficient *values* must be nonnegative numbers. The number of *values* must equal the number of replicate weight variables specified in the REPWEIGHTS statement. List these values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

REPCOEFS=*SAS-data-set*
names a *SAS-data-set* that contains the replicate coefficients. You must provide the replicate coefficients in a variable named Coefficient or RepCoefficient in the *SAS-data-set*. Each coefficient value must be a nonnegative number. The observations in the *SAS-data-set* should correspond to the replicates that are identified by the variables that are specified in the REPWEIGHTS statement. Arrange the coefficients or observations in the *SAS-data-set* in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement. The number of observations in the *SAS-data-set* must not be less than the number of variables specified in the REPWEIGHTS statement.

For more information about replication coefficients, see the section "Replicate Weights Method" on page 10268.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

STRATA Statement

STRATA *variables* < / *option* > **;**

The STRATA statement names variables that form the strata in a stratified sample design. The combinations of levels of STRATA variables define the strata in the sample, where strata are nonoverlapping subgroups that were sampled independently.

If your sample design has stratification at multiple stages, you should identify only the first-stage strata in the STRATA statement. See the section "Specifying the Sample Design" on page 10257 for more information.

If you provide replicate weights for replication variance estimation in a REPWEIGHTS statement, you do not need to specify a STRATA statement.

The STRATA *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric, but the procedure treats them as categorical variables. The formatted values of the STRATA variables determine the STRATA variable levels. Thus, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in the *SAS Formats and Informats: Reference*. Strata variables must not occur in the CLASS statement.

The STRATA statement in PROC SURVEYPHREG is different from the STRATA statement in PROC PHREG (Chapter 92, "The PHREG Procedure"). PROC PHREG fits different baseline hazard functions in different strata, which is useful if the proportional hazards assumption is not satisfied.

You can specify the following *option* in the STRATA statement after a slash (/):

LIST

displays a "Stratum Information" table, which lists all strata together with the corresponding values of the STRATA variables. This table provides the number of observations and the number of clusters in each stratum, as well as the sampling fraction if you specify the RATE= or TOTAL= option.

TEST Statement

TEST < *model-effects* > < / *options* > **;**

The TEST statement enables you to perform *F* tests for model effects that test Type I, Type II, or Type III hypotheses. For more information about constructing Type I, II, and III estimable functions, see Chapter 16, "The Four Types of Estimable Functions."

Table 122.9 summarizes the *options* available in the TEST statement.

Table 122.9 TEST Statement Options

Option	Description
CHISQ	Requests chi-square tests
DDF=	Specifies denominator degrees of freedom for fixed effects
E	Requests Type I, Type II, and Type III coefficients
E1	Requests Type I coefficients
E2	Requests Type II coefficients
E3	Requests Type III coefficients
HTYPE=	Indicates the type of hypothesis test to perform
INTERCEPT	Adds a row that corresponds to the overall intercept

For more information about the syntax of the TEST statement, see the section "TEST Statement" on page 556 in Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement names the variable that contains the sampling weights. This variable must be numeric, and the sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, then the procedure omits that observation from the analysis. See the section "Missing Values" on page 10259 for more information. The WEIGHT statement allows one weight variable.

If you do not specify a WEIGHT statement but provide replicate weights with a REPWEIGHTS statement, PROC SURVEYPHREG uses the average of each observation's replicate weights as the observation's weight.

If you specify neither a WEIGHT statement nor a REPWEIGHTS statement, PROC SURVEYPHREG assigns all observations a weight of one.

Details: SURVEYPHREG Procedure

Notation and Estimation

Let $U = \{1, 2, \ldots, N\}$ be the set of indices and let $\mathcal{F}_N$ be the set of values for a finite population of size N. The survival time of each member of the finite population is assumed to follow its own hazard function, $\lambda_i(t)$, expressed as

$$\lambda_i(t) = \lambda(t; \mathbf{Z}_i(t)) = \lambda_0(t)\ \exp(\mathbf{Z}_i'(t)\boldsymbol{\beta})$$

where $\lambda_0(t)$ is an arbitrary and unspecified baseline hazard function, $\mathbf{Z}_i(t)$ is the vector of explanatory variables for the ith unit at time t, and $\boldsymbol{\beta}$ is the vector of unknown regression parameters that are associated with the explanatory variables. The vector $\boldsymbol{\beta}$ is assumed to be the same for all individuals.

The partial likelihood function introduced by Cox (1972, 1975) eliminates the unknown baseline hazard $\lambda_0(t)$ and accounts for censored survival times. If the entire population is observed, then this partial likelihood can be used to estimate $\boldsymbol{\beta}$. Let $\boldsymbol{\beta}_N$ be the desired estimator. Assuming a working model with uncorrelated responses, $\boldsymbol{\beta}_N$ is obtained by maximizing the partial log likelihood,

$$l(\boldsymbol{\beta}) = \sum_{i \in U} \log\left\{\mathrm{L}(\boldsymbol{\beta}; \mathbf{Z}_i(t), t_i)\right\}$$

with respect to $\boldsymbol{\beta}$, where $\mathrm{L}(\boldsymbol{\beta}; \mathbf{Z}_i(t), t_i)$ is Cox's partial likelihood function.

Assume that probability sample A is selected from the finite population U and π_i is the selection probability for unit i. Further assume that covariates $\mathbf{Z}_i(t)$ and survival time t_i are available for every unit in the sample A. An estimator of the finite population log likelihood is

$$l_\pi(\boldsymbol{\beta}) = \sum_{i \in A} \pi_i^{-1} \log\left\{\mathrm{L}(\boldsymbol{\beta}; \mathbf{Z}_i(t), t_i)\right\}$$

See "Partial Likelihood Function for the Cox Model" on page 10252 for more details.

A sample-based estimator $\hat{\boldsymbol{\beta}}$ for the finite population quantity $\boldsymbol{\beta}_N$ can be obtained by maximizing the partial pseudo-log-likelihood $l_\pi(\boldsymbol{\beta}; \mathbf{Z}_i(t), t_i)$ with respect to $\boldsymbol{\beta}$. The design-based variance for $\hat{\boldsymbol{\beta}}$ is obtained by assuming the set of finite population values $\mathcal{F}_N$ as fixed. For more information about maximum pseudo-likelihood estimators and other inferential approaches for survey data, see Kish and Frankel (1974); Godambe and Thompson (1986); Pfeffermann (1993), Korn and Graubard (1999, chapter 3), Chambers and Skinner (2003, chapter 2), and Fuller (2009, section 6.5). Maximum pseudo-likelihood estimators and their properties for Cox's proportional hazards model for survey data are discussed in Binder (1990, 1992); Lin and Wei (1989); Lin (2000); Boudreau and Lawless (2006).

Without loss of generality, the rest of this section uses indices for stratified clustered designs. For a stratified clustered sample design, observations are represented by a matrix

$$(\mathbf{w}, \mathbf{t}, \boldsymbol{\Delta}, \mathbf{Z}) = (w_{hij}, t_{hij}, \Delta_{hij}, \mathbf{z}_{hij})$$

where

- $\mathbf{w}$ denotes the vector of sampling weights
- $\mathbf{t}$ denotes the event time variable
- $\mathbf{\Delta}$ denotes the event indicator
- $\mathbf{Z}$ denotes the $n \times p$ matrix of auxiliary information
- $h = 1, 2, \ldots, H$ is the stratum index
- $i = 1, 2, \ldots, n_h$ is the cluster index within stratum h
- $j = 1, 2, \ldots, m_{hi}$ is the unit index within cluster i of stratum h
- p is the total number of parameters
- $n = \sum_{h=1}^{H} \sum_{i=1}^{n_h} m_{hi}$ is the total number of observations in the sample
- $y_{hij}(t) = I(t_{hij} \geq t)$, where $I(\cdot)$ is an indicator function
- $n_{hij}(t) = I(t_{hij} \leq t)$, where $I(\cdot)$ is an indicator function

Let $\sum_B = \sum_{(h,i,j)\in B}$ denote the summation over the set of indices such that the observation unit j in PSU i and stratum h belongs to the index set B. Typically, B is the set of all population indices that are in the sample, the risk set, or the set of all units with a failure.

The first-stage sampling rate (fraction of PSUs selected for the sample) is denoted by f_h. The first-stage sampling rate is used in Taylor series or bootstrap variance estimation. You can specify the stratum sampling rates with the RATE= option. Or if you specify population totals with the TOTAL= option, PROC SURVEYPHREG computes f_h as the ratio of stratum sample size to the stratum total, in terms of PSUs. See the section "Population Totals and Sampling Rates" on page 10258 for details. If you do not specify the RATE= option or the TOTAL= option, then the procedure assumes that the stratum sampling rates f_h are negligible and does not use a finite population correction when computing variances.

Failure Time Distribution

Let T be a nonnegative random variable that represents the failure time of an individual from a homogeneous superpopulation. The survival distribution function (also known as the survivor function) of T is written as

$$S(t) = \Pr(T \geq t)$$

A mathematically equivalent way of specifying the distribution of T is through its hazard function. The hazard function $\lambda(t)$ specifies the instantaneous failure rate at t. If T is a continuous random variable, $\lambda(t)$ is expressed as

$$\lambda(t) = \lim_{\Delta t \to 0^+} \frac{\Pr(t \leq T < t + \Delta t \mid T \geq t)}{\Delta t} = \frac{f(t)}{S(t)}$$

where $f(t)$ is the probability density function of T.

Time and CLASS Variable Usage

The following DATA step creates an artificial data set, Test, to be used in this section. Test contains six variables: T contains the failure times; Status is the censoring indicator variable, whose value is 1 for an uncensored failure time and 0 for a censored failure time; A is an auxiliary variable whose value ranges from 1.0 to 3.9; MirrorT is an exact copy of T; W is the observation weight; and S is the strata indicator.

```
data Test;
   input T Status A W S @@;
   MirrorT = T;
   datalines;
 23    1    1   10   1    7    0   1   20   2
 23    1    1   10   1   10    1   1   20   2
 20    0    1   10   1   13    0   1   20   2
 24    1    1   10   1   10    1   1   20   2
 18    1    2   10   1    6    1   2   20   2
 18    0    2   10   1    6    1   2   20   2
 13    0    2   10   1   13    1   2   20   2
  9    0    2   10   1   15    1   2   20   2
  8    1    3   10   1    6    1   3   20   2
 12    0    3   10   1    4    1   3   20   2
 11    1    3   10   1    8    1   1   20   2
  6    1    3   10   1    7    1   3   20   2
  7    1    3   10   1   12    1   3   20   2
  9    1    2   10   1   15    1   2   20   2
  3    1    2   10   1   14    0   3   20   2
  6    1    1   10   1   13    1   2   20   2
;
```

Time Variable on the Right Side of the MODEL Statement

You cannot use the time variable explicitly as an explanatory effect in the MODEL statement. The following statements produce an error message:

```
proc surveyphreg data=Test;
   weight W;
   strata S;
   model T*Status(0)=A*T;
run;
```

To use the time variable as an explanatory effect, replace T with MirrorT as an effect, as in the following statements. MirrorT is an exact copy of T.

```
proc surveyphreg data=Test;
   weight W;
   strata S;
   class A;
   model T*Status(0)=A*MirrorT;
run;
```

Note that neither T*A nor MirrorT*A in the MODEL statement is time-dependent.

CLASS Variables and Programming Statements

In PROC SURVEYPHREG, the levels of CLASS variables are determined by the CLASS statement and the input data and are not affected by user-supplied programming statements. Consider the following statements, which produce the results in Figure 122.4. Variable A is declared as a CLASS variable in the CLASS statement.

```
proc surveyphreg data=Test;
   weight W;
   strata S;
   class A;
   model T*Status(0)=A;
run;
```

Figure 122.4 shows the parameters that correspond to A and their respective regression coefficients estimates.

Figure 122.4 Design Variable and Regression Coefficient Estimates

The SURVEYPHREG Procedure

Class Level Information		
Class	Levels	Values
A	3	1 2 3

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Hazard Ratio
A 1	30	-1.162184	0.644483	-1.80	0.0814	0.313
A 2	30	-0.616962	0.513355	-1.20	0.2388	0.540
A 3	30	0	.	.	.	1.000

Now consider the programming statement that attempts to change the value of the CLASS variable A as in the following specification:

```
proc surveyphreg data=Test;
   weight W;
   strata S;
   class A;
   model T*Status(0)=A;
   if A=3 then A=2;
run;
```

Results of this analysis are shown in Figure 122.5 and are identical to those in Figure 122.4. The **`if A=3 then A=2`** programming statement has no effect on the explanatory variable for A, which have already been determined.

Figure 122.5 Design Variable and Regression Coefficient Estimates

The SURVEYPHREG Procedure

Class Level Information		
Class	**Levels**	**Values**
A	3	1 2 3

Analysis of Maximum Likelihood Estimates						
Parameter	**DF**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**	**Hazard Ratio**
A 1	30	-1.162184	0.644483	-1.80	0.0814	0.313
A 2	30	-0.616962	0.513355	-1.20	0.2388	0.540
A 3	30	0	.	.	.	1.000

Additionally any variable used in a programming statement that has already been declared in the CLASS statement is *not* treated as a collection of the corresponding design variables. Consider the following statements:

```
proc surveyphreg data=Test;
   class A;
   model T*Status(0)=A X;
   X=T*A;
run;
```

The CLASS variable A generates two design variables as explanatory variables. The variable X created by the **X=T*A** programming statement is a single time-dependent covariate whose values are evaluated using the exact values of A given in the data, not the dummy coded values that represent A. In the data set Test, A has the values of 1, 2, and 3, and these values are multiplied by the values of T to produce X. If A were a character variable with values 'Bird', 'Cat', and 'Dog', the programming statement X=T*A would have produced an error in the attempt to multiply a number with a character value.

Figure 122.6 Single Time-Dependent Variable X*A

The SURVEYPHREG Procedure

Analysis of Maximum Likelihood Estimates						
Parameter	**DF**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**	**Hazard Ratio**
A 1	31	0.158010	1.182556	0.13	0.8946	1.171
A 2	31	0.008993	0.652504	0.01	0.9891	1.009
A 3	31	0	.	.	.	1.000
X	31	0.092679	0.071328	1.30	0.2034	1.097

The following statements are not the same as in the preceding program. If you want to create time-dependent covariates from the values of a CLASS variable, you could use syntax like the following:

```
proc surveyphreg data=Test;
   class A;
   model T*Status(0)=A X1 X2;
   X1= T*(A=1);
```

```
    X2= T*(A=2);
run;
```

The Boolean parenthetical expressions (A=1) and (A=2) resolve to a value of 1 or 0, depending on whether the expression is true or false, respectively.

Results of this test are shown in Figure 122.7.

Figure 122.7 Simple Test of Proportional Hazards Assumption

The SURVEYPHREG Procedure

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Hazard Ratio
A 1	31	-0.007655	1.221122	-0.01	0.9950	0.992
A 2	31	-0.881383	1.743507	-0.51	0.6168	0.414
A 3	31	0	.	.	.	1.000
X1	31	-0.155220	0.164334	-0.94	0.3522	0.856
X2	31	0.011554	0.188932	0.06	0.9516	1.012

In general, when your model contains a categorical explanatory variable that is time-dependent, it might be necessary to use hardcoded dummy variables to represent the categories of the categorical variable.

Partial Likelihood Function for the Cox Model

Let $t_{(1)} < t_{(2)} < \ldots < t_{(K)}$ denote the K distinct, ordered event times. Let d_k denote the multiplicity of failures at $t_{(k)}$; that is, d_k is the size of the set $\mathcal{D}_k$ of individuals that fail at $t_{(k)}$. Let w_{hij} be the weight associated with the jth observation unit in the ith cluster in stratum h. Using this notation, the pseudo-likelihood functions used in PROC SURVEYPHREG to estimate $\boldsymbol{\beta}_N$ are described in the following sections.

Let $\mathcal{R}_k$ denote the risk set just before the kth ordered event time $t_{(k)}$.

The Breslow likelihood is expressed as

$$L_{\text{Breslow}}(\boldsymbol{\beta}) = \prod_{k=1}^{K} \frac{\exp\left(\boldsymbol{\beta}' \sum_{\mathcal{D}_k} w_{hij} \mathbf{Z}_{hij}(t)\right)}{\left\{\sum_{\mathcal{R}_k} w_{hij} \exp(\boldsymbol{\beta}' \mathbf{Z}_{hij}(t))\right\}^{\sum_{\mathcal{D}_k} w_{hij}}}$$

The Efron likelihood is expressed as

$$L_{\text{Efron}}(\boldsymbol{\beta}) = \prod_{k=1}^{K} \frac{\exp\left(\boldsymbol{\beta}' \sum_{\mathcal{D}_k} w_{hij} \mathbf{Z}_{hij}(t)\right)}{\{\phi(\boldsymbol{\beta}, \mathbf{Z}, \mathbf{w}, k)\}^{\frac{1}{d_k} \sum_{\mathcal{D}_k} w_{hij}}}$$

where $\phi(\boldsymbol{\beta}, \mathbf{Z}, \mathbf{w}, k)$ is

$$\phi(\boldsymbol{\beta}, \mathbf{Z}, \mathbf{w}, k) = \prod_{l=1}^{d_k} \left\{\sum_{\mathcal{R}_k} w_{hij} \exp\left(\boldsymbol{\beta}' \mathbf{Z}_{hij}(t)\right) - \frac{l-1}{d_k} \sum_{\mathcal{D}_k} w_{hij} \exp\left(\boldsymbol{\beta}' \mathbf{Z}_{hij}(t)\right)\right\}$$

Counting Process Style of Input

In the counting process formulation, data for each subject are identified by a triple $\{N, Y, \mathbf{Z}\}$ of counting, at-risk, and covariate processes. $N(t)$ indicates the sum of weights for all events that the subject experiences over the time interval $(0, t]$, $Y(t)$ indicates whether the subject is at risk at time t (1 if at risk and 0 otherwise), and $\mathbf{Z}(t)$ is a vector of explanatory variables for the subject at time t. The sample path of N is a step function with jumps at the event times, and $N(0) = 0$. Unless $\mathbf{Z}(t)$ changes continuously with time, the data for each subject can be represented by multiple observations, each of which identifies by a semiclosed time interval $(t_1, t_2]$, the values of the explanatory variables over that interval, and the event status at t_2. The subject remains at risk during the interval $(t_1, t_2]$, and an event might occur at t_2. Values of the explanatory variables for the subject remain unchanged in the interval. This style of data input was originated by Therneau (1994).

For example, suppose a patient (ID=1) with an analysis weight of 10 has a tumor recurrence at weeks 3, 10, and 15 and is followed up until week 23. Consider three fixed explanatory variables Trt (treatment), Number (initial tumor number), and Size (initial tumor size), one weight variable Weight (analysis weight), one patient identification variable ID, and one time-dependent covariate Z that represents a hormone level. The value of Z might change during the follow-up period. The data for this patient are represented by the following four observations:

ID	Weight	T1	T2	Status	Trt	Number	Size	Z
1	10	0	3	1	1	1	3	12.3
1	10	3	10	1	1	1	3	14.7
1	10	10	15	1	1	1	3	13.8
1	10	15	23	0	1	1	3	15.5

Here (T1,T2] contains the at-risk intervals. The variable Status indicates whether a recurrence has occurred at T2: a value of 1 indicates a tumor recurrence, and a value of 0 indicates non-recurrence. Assume the patients are selected independently. Because there are multiple observation rows for every patient, you should use the CLUSTER statement to identify each individual patient. The CLUSTER statement computes the variability between the patients. The following statements fit a multiplicative hazards model with baseline covariates Trt, Number, and Size, and a time-varying covariate Z. For more information, see the section "The Multiplicative Hazards Model" on page 10254.

```
proc surveyphreg;
   weight Weight;
   cluster ID;
   model (T1,T2) * Status(0) = Trt Number Size Z;
run;
```

Another useful application of the counting process formulation is the delayed entry of subjects into the risk set. For example, in studying the mortality of workers exposed to a carcinogen, the survival time is chosen to be the worker's age at death by malignant neoplasm. Any worker who joins the workplace at an age later than the failure time of an event is not included in the corresponding risk set. The variables for a worker consist of Entry (age at which the worker entered the workplace), Age (age at death or age censored), Status (an indicator of whether the observation time is censored, with the value 0 identifying a censored time), and X1 and X2 (explanatory variables thought to be related to survival). The specification for such an application is as follows:

```
proc surveyphreg;
   model (Entry, Age) * Status(0) = X1 X2;
run;
```

Left-Truncation of Failure Times

Left-truncation occurs when individuals are not observed at the natural time origin of the phenomenon under study but come under observation at some known later time (called the left-truncation time). The risk set just prior to an event time does not include individuals whose left-truncation times exceed that event time. Thus, any contribution to the likelihood must be conditional on the truncation limit having been exceeded.

You use the ENTRY= option to specify the variable that represents the left-truncation time. Suppose T1 and T2 represent the left-truncation time and the survival time, respectively. To account for left-truncation, you specify the following statements:

```
proc surveyphreg;
   model T2*Dead(0)=X1-X10/entry=T1;
   title 'The ENTRY= option is Specified';
run;
```

Equivalently, you can use the counting process style of input for left-truncation:

```
proc surveyphreg;
   model (T1,T2)*Dead(0)=X1-X10;
   title 'Counting Process Style of Input';
run;
```

The Multiplicative Hazards Model

Consider a set of n subjects such that the counting process $N_i \equiv \{N_i(t), t \geq 0\}$ for the ith subject represents the number of observed events that are experienced over time t. The sample paths of the process N_i are step functions with jumps of size 1, with $N_i(0) = 0$. Let $\boldsymbol{\beta}$ denote the vector of unknown regression coefficients. The multiplicative hazards function $\Lambda(t, \mathbf{Z}_i(t))$ for N_i is given by

$$Y_i(t)d\Lambda(t, \mathbf{Z}_i(t)) = Y_i(t)\exp(\boldsymbol{\beta}'\mathbf{Z}_i(t))d\Lambda_0(t)$$

where

- $Y_i(t)$ indicates whether the ith subject is at risk at time t (specifically, $Y_i(t) = 1$ if at risk and $Y_i(t) = 0$ otherwise)
- $\mathbf{Z}_i(t)$ is the vector of explanatory variables for the ith subject at time t
- $\Lambda_0(t)$ is an unspecified baseline hazard function

See Fleming and Harrington (1991) and Andersen et al. (1992). The Cox model is a special case of this multiplicative hazards model, where $Y_i(t) = 1$ until the first event or censoring, and $Y_i(t) = 0$ thereafter.

The partial likelihood for n independent triplets $(N_i, Y_i, \mathbf{Z}_i), i = 1, \ldots, n$, has the form

$$\mathcal{L}(\boldsymbol{\beta}) = \prod_{i=1}^{n} \prod_{t \geq 0} \left\{ \frac{Y_i(t) \exp(\boldsymbol{\beta}' \mathbf{Z}_i(t))}{\sum_{j=1}^{n} w_j Y_j(t) \exp(\boldsymbol{\beta}' \mathbf{Z}_j(t))} \right\}^{w_i \Delta N_i(t)}$$

where w_i is the weight for subject i, $\Delta N_i(t) = 1$ if $N_i(t) - N_i(t-) = 1$, and $\Delta N_i(t) = 0$ otherwise.

A CLUSTER statement is necessary in order to appropriately estimate variances from multiplicative hazards models. If your design has a primary sampling unit (PSU), then use the PSU identification in the CLUSTER statement. Otherwise, use the subject identification in the CLUSTER statement.

Firth's Modification for Maximum Likelihood Estimation

In fitting a Cox model, the phenomenon of monotone likelihood is observed if the likelihood converges to a finite value while at least one parameter diverges (Mukhopadhyay 2020; Heinze and Schemper 2001).

Firth (1993) recommended using the penalized likelihood $L^*(\boldsymbol{\beta}) = L(\boldsymbol{\beta})|\mathcal{I}(\boldsymbol{\beta})|^{0.5}$ to reduce the first-order bias in estimating the canonical parameters of an exponential family model, where $L(\boldsymbol{\beta})$ and $\mathcal{I}(\boldsymbol{\beta})$ are the unpenalized likelihood and information matrix, respectively.

Heinze (1999) and Heinze and Schemper (2001) applied the idea of Firth (1993) by maximizing the penalized partial log likelihood

$$l^*(\boldsymbol{\beta}) = l(\boldsymbol{\beta}) + 0.5 \log(|\mathcal{I}(\boldsymbol{\beta})|)$$

to obtain estimates of regression parameters when a monotone likelihood is observed.

The score function $\mathbf{U}(\boldsymbol{\beta})$ is replaced by the penalized score function, $\mathbf{U}^*(\boldsymbol{\beta}) \equiv (U^*(\beta_1), \ldots, U^*(\beta_p))'$, where

$$U^*(\beta_r) = U(\beta_r) + 0.5 \mathrm{tr} \left\{ \mathcal{I}^{-1}(\boldsymbol{\beta}) \frac{\partial \mathcal{I}(\boldsymbol{\beta})}{\partial \beta_r} \right\} \qquad r = 1, \ldots, p$$

The Firth estimate is obtained iteratively as

$$\boldsymbol{\beta}^{(s+1)} = \boldsymbol{\beta}^{(s)} + \mathcal{I}^{-1}(\boldsymbol{\beta}^{(s)}) \mathbf{U}^*(\boldsymbol{\beta}^{(s)})$$

Although the estimated regression parameters, $\hat{\boldsymbol{\beta}}$, are obtained by maximizing the penalized partial likelihood, the Taylor series linearized variance estimator uses the score residuals and the information matrix from the unpenalized likelihood that are evaluated at $\hat{\boldsymbol{\beta}}$. For more information, see the section "Taylor Series Linearization" on page 10262.

The replication variance estimation methods use the replicated version of the penalized score function to obtain replicate estimates, $\hat{\boldsymbol{\beta}}^{(r)}$, for the regression parameters. The replicate estimates are then used in the replication variance estimation, as described in the sections "Balanced Repeated Replication (BRR) Method" on page 10263, "Bootstrap Method" on page 10266, "Jackknife Method" on page 10267, and "Replicate Weights Method" on page 10268.

Explicit Formulas for the Score Function, Fisher Information, and Partial Derivatives for the Information Matrix

Mukhopadhyay (2020) recommended using normalized weights to construct the penalized log partial likelihood for weighted data. Using the notation in the sections "Notation and Estimation" on page 10247 and "Partial Likelihood Function for the Cox Model" on page 10252, the Breslow unpenalized log partial likelihood is given by

$$l(\boldsymbol{\beta}) = \log(L_{\text{Breslow}}(\boldsymbol{\beta})) = \sum_{k=1}^{K}\left\{\boldsymbol{\beta}' \sum_{hij\in\mathcal{D}_k} \tilde{w}_{hij}\mathbf{Z}_{hij}(t_k) - \Big(\sum_{hij\in\mathcal{D}_k} \tilde{w}_{hij}\Big) \log \sum_{hij\in\mathcal{R}_k} \tilde{w}_{hij} \exp\big(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t_k)\big)\right\}$$

where $\tilde{w}_{hij} = n\big(\sum_{hij\in\mathcal{A}} w_{hij}\big)^{-1} w_{hij}$ is the normalized weight, n is the number of observation units, and w_{hij} is the weight for unit j in PSU i and stratum h.

Denote

$$\mathbf{S}_k^{(a)}(\boldsymbol{\beta}) = \sum_{hij\in\mathcal{R}_k} \tilde{w}_{hij} \exp\big(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t_k)\big)[\mathbf{Z}_{hij}(t_k)]^{\otimes a}$$

where $a = 0, 1, 2$.

Then the score function is given by

$$\begin{aligned}
\mathbf{U}(\boldsymbol{\beta}) &\equiv (U(\beta_1), \ldots, U(\beta_p))' \\
&= \frac{\partial l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \\
&= \sum_{k=1}^{K}\left\{\sum_{hij\in\mathcal{D}_k} \tilde{w}_{hij}\mathbf{Z}_{hij}(t_k) - \sum_{hij\in\mathcal{D}_k} \tilde{w}_{hij} \frac{\mathbf{S}_k^{(1)}(\boldsymbol{\beta})}{S_k^{0}(\boldsymbol{\beta})}\right\}
\end{aligned}$$

and the Fisher information matrix is given by

$$\begin{aligned}
\mathcal{I}(\boldsymbol{\beta}) &= -\frac{\partial^2 l(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}^2} \\
&= \sum_{k=1}^{K}\sum_{hij\in\mathcal{D}_k} \tilde{w}_{hij}\left\{\frac{\mathbf{S}_k^{(2)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} - \left[\frac{\mathbf{S}_k^{(1)}(\boldsymbol{\beta})}{\mathbf{S}_k^{(0)}(\boldsymbol{\beta})}\right]\left[\frac{\mathbf{S}_k^{(1)}(\boldsymbol{\beta})}{\mathbf{S}_k^{(0)}(\boldsymbol{\beta})}\right]'\right\}
\end{aligned}$$

Denote

$$\mathbf{Q}_{kr}^{(a)}(\boldsymbol{\beta}) = \sum_{hij\in\mathcal{R}_k} \tilde{w}_{hij} \exp\big(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t_k)\big) Z_{hij,r}(t_k)[\mathbf{Z}_{hij}(t_k)]^{\otimes a}$$

where $a = 0, 1, 2$; $r = 1, \ldots, p$; and $\mathbf{Z}_{hij}(t) = (Z_{hij,1}(t), \ldots, Z_{hij,p}(t))$. Then

$$
\begin{aligned}
\frac{\partial \mathcal{I}(\boldsymbol{\beta})}{\partial \beta_r} \quad = \quad \sum_{k=1}^{K} \sum_{hij \in \mathcal{D}_k} \tilde{w}_{hij} \Bigg\{ & \left[\frac{\mathbf{Q}_{kr}^{(2)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} - \frac{\mathbf{Q}_{kr}^{(0)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} \frac{\mathbf{S}_k^{(2)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} \right] \\
& - \left[\frac{\mathbf{Q}_{kr}^{(1)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} - \frac{\mathbf{Q}_{kr}^{(0)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} \frac{\mathbf{S}_k^{(1)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} \right] \left[\frac{\mathbf{S}_k^{(1)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} \right]' \\
& - \left[\frac{\mathbf{S}_k^{(1)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} \right] \left[\frac{\mathbf{Q}_{kr}^{(1)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} - \frac{\mathbf{Q}_{kr}^{(0)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} \frac{\mathbf{S}_k^{(1)}(\boldsymbol{\beta})}{S_k^{(0)}(\boldsymbol{\beta})} \right]' \Bigg\}
\end{aligned}
$$

where $r = 1, \ldots, p$.

Specifying the Sample Design

PROC SURVEYPHREG produces statistics that are based on the sample design used to obtain the survey data. PROC SURVEYPHREG can be used for single-stage or multistage designs, with or without stratification, and with or without unequal weighting. To analyze your survey data with PROC SURVEYPHREG, you need to provide sample design information for the procedure. This information can include design (or variance) strata, clusters, and sampling weights. You provide sample design information with the STRATA, CLUSTER, WEIGHT, and REPWEIGHTS statements, and with the RATE= or TOTAL= option in the PROC SURVEYPHREG statement.

If you provide replicate weights for replication variance estimation, you do not need to specify a STRATA or CLUSTER statement. Otherwise, you should specify STRATA and CLUSTER statements whenever your design includes stratification and clustering.

When there are clusters (PSUs) in the sample design, the procedure estimates variance by using the PSUs, as described in the section "Variance Estimation" on page 10262. For a multistage sample design, PROC SURVEYPHREG uses only the first stage of the sample design for variance estimation. So, the required input includes only first-stage cluster (PSU) and first-stage stratum identification. You do not need to input design information about any additional stages of sampling.

Stratification

If your sample design is stratified at the first stage of sampling, use the STRATA statement to name the variables that form the strata. The combinations of categories of STRATA variables define the strata in the sample, where strata are nonoverlapping subgroups that were sampled independently. If your sample design has stratification at multiple stages, you should identify only the first-stage strata in the STRATA statement.

If you use a REPWEIGHTS statement to provide replicate weights for replication variance estimation, you do not need to specify a STRATA statement. Otherwise, you should specify a STRATA statement whenever your design includes stratification. If you do not specify a STRATA statement or a REPWEIGHTS statement, then PROC SURVEYPHREG assumes there is no stratification at the first stage. In other words, in this case, the procedure assumes that all observation units are in the same stratum.

Clustering

If your sample design selects clusters at the first stage of sampling, use the CLUSTER statement to name the variables that identify the first-stage clusters, which are also called primary sampling units (PSUs). The combinations of categories of CLUSTER variables define the clusters in the sample. If there is a STRATA statement, clusters are nested within strata. If your sample design has clustering at multiple stages, you should specify only the first-stage clusters (PSUs) in the CLUSTER statement. PROC SURVEYPHREG assumes that each cluster that is defined by the CLUSTER statement variables represents a PSU in the sample.

If you use a REPWEIGHTS statement to provide replicate weights for replication variance estimation, you do not need to specify a CLUSTER statement. Otherwise, you should specify a CLUSTER statement whenever your design includes clustering at the first stage of sampling. If you do not specify a CLUSTER statement, then PROC SURVEYPHREG treats each observation as a PSU.

Weighting

If your sample design includes unequal weighting, use the WEIGHT statement to name the variable that contains the sampling weights. Sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, then the procedure omits that observation from the analysis. See the section "Missing Values" on page 10259 for more information.

If you do not specify a WEIGHT statement but include a REPWEIGHTS statement, PROC SURVEYPHREG uses the average of each observation's replicate weights as the observation's weight. If you specify neither a WEIGHT statement nor a REPWEIGHTS statement, PROC SURVEYPHREG assumes all observations have a weight of one.

Population Totals and Sampling Rates

To include a finite population correction (*fpc*) in Taylor series or bootstrap variance estimation, you can specify either the sampling rate or the population total by using the RATE= or TOTAL= option, respectively, in the PROC SURVEYPHREG statement. You cannot specify both of these options in the same PROC SURVEYPHREG statement. The RATE= and TOTAL= options apply only to Taylor series or bootstrap variance estimation. The procedure does not use a finite population correction for BRR or jackknife variance estimation.

If you do not specify the RATE= or TOTAL= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. For fairly small sampling fractions, this correction is often ignored. For more information, see Cochran (1977) and Kish (1965).

If your design has multiple stages of selection and you are specifying the RATE= option, you should input the first-stage sampling rate, which is the ratio of the number of PSUs in the sample to the total number of PSUs in the study population. If you are specifying the TOTAL= option for a multistage design, you should input the total number of PSUs in the study population.

For a nonstratified sample design, or for a stratified sample design with the same sampling rate or the same population total in all strata, you can use the RATE=*value* or TOTAL=*value* option. If your sample design is stratified with different sampling rates or population totals in different strata, use the RATE=*SAS-data-set* or TOTAL=*SAS-data-set* option to name a SAS data set that contains the stratum sampling rates or totals. This data set is called a *secondary data set*, as opposed to the *primary data set* that you specify with the DATA= option.

The secondary data set must contain all the stratification variables listed in the STRATA statement and all the variables in the BY statement. Furthermore, the BY groups must appear in the same order as in the primary data set. If there are formats associated with the STRATA variables and the BY variables, then the formats must be consistent in the primary and the secondary data sets. If you specify the TOTAL=*SAS-data-set* option, the secondary data set must have a variable named _TOTAL_ that contains the stratum population totals. If you specify the RATE=*SAS-data-set* option, the secondary data set must have a variable named _RATE_ that contains the stratum sampling rates. If the secondary data set contains more than one observation for any one stratum, the procedure uses the first value of _TOTAL_ or _RATE_ for that stratum and ignores the rest.

The *value* in the RATE= option or the values of _RATE_ in the secondary data set must be nonnegative numbers. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC SURVEYPHREG converts that number to a proportion. The procedure treats the value 1 as 100% instead of 1%.

If you specify the TOTAL=*value* option, *value* must not be less than the sample size. If you provide stratum population totals in a secondary data set, these values must not be less than the corresponding stratum sample sizes.

Replicate Weights

If you have replicate weights available for your survey data, use the REPWEIGHTS statement to name the variables that contain the replicate weights. Replicate weights must be positive numbers. If an observation has a replicate weight that is nonpositive or missing, then PROC SURVEYPHREG does not perform any analyses. For more information, see the section "Missing Values" on page 10259.

If you have replicate coefficients available for your survey data, use the REPCOEFS= option in the REPWEIGHTS statement to specify the replicate coefficients.

Missing Values

Missing values in your survey data can compromise the quality of your survey results. Some missing values for survey data are because of nonresponses. An observation whose response to every survey item is available is called a *complete respondent*, and an observation whose response to one or more survey items are missing is called an *incomplete respondent*. If the complete respondents are different from the incomplete respondents with regard to a survey effect or outcome, then survey estimates will be biased and will not accurately represent the survey population. There are a variety of techniques in sample design and survey operations that can reduce nonresponse. After data collection is complete, you can use imputation to replace missing values with acceptable values, and you can use sampling weight adjustments to compensate for nonresponse. You should complete this data preparation and adjustment before you analyze your data with PROC SURVEYPHREG. For more details, see Cochran (1977); Kalton and Kasprzyk (1986); Brick and Kalton (1996).

WEIGHT Variable

If an observation has a missing value or a nonpositive value for the WEIGHT variable, then PROC SURVEYPHREG excludes that observation from the analysis.

REPWEIGHTS Variables

If you provide replicate weights in a REPWEIGHTS statement for replication variance estimation, all REPWEIGHTS variable values must be nonmissing. Similarly, if you provide jackknife coefficients in the JKCOEFS= option in the REPWEIGHTS statement, all values of the JKCoefficient variable must be nonmissing. The procedure does not perform the analysis when any replicate weight or jackknife coefficient value is missing.

CLASS, STRATA, CLUSTER, and DOMAIN Variables

An observation is excluded from the analysis if it has a missing value for any CLASS, STRATA, CLUSTER, or DOMAIN variable, unless you specify the MISSING option in the PROC SURVEYPHREG statement. If you specify the MISSING option, the procedure treats missing values as a valid (nonmissing) category for all categorical variables, which include STRATA variables, CLUSTER variables, CLASS variables, and DOMAIN variables.

Analysis Variables

By default, PROC SURVEYPHREG excludes an observation from the likelihood estimation and all associated analyses if the observation has a missing value for any of the variables in the MODEL statement, unless you specify the MISSING or NOMCAR option in the PROC SURVEYPHREG statement. When the procedure excludes observations with missing values from analyses, it displays the total frequency of observations used in the NObs table.

If you specify time-dependent covariates by using programming statements, the procedure computes the values of the covariates for all observations in the risk set at every event time. If an observation contains missing values for any of the time-dependent covariates at a given event time, then the observation is not used at that event time. However, that same observation can be used at some other event times where it contains no missing values. Therefore, an observation with missing time-dependent covariates can be used at some event times but ignored at other event times, depending on whether any of the corresponding time-dependent covariates are missing.

If you specify the MISSING option, the procedure treats missing levels as a valid (nonmissing) level for each categorical analysis variable.

If you specify the NOMCAR option for Taylor series variance estimation, the procedure includes observations with missing values of analysis variables in the variance computations.

The NOMCAR Option

When you specify the NOMCAR option, PROC SURVEYPHREG computes variance estimates by analyzing the nonmissing values for variables in the regression model as a domain or subpopulation, where the entire population includes both nonmissing and missing domains. By default, if an observation contains missing values for the dependent variable or for any variable used in the independent effects, the observation is excluded from the analysis. See the section "Missing Values" on page 10259 for more information.

Note that the NOMCAR option has no effect on categorical predictors when you specify the MISSING option, which treats missing values as a valid nonmissing level. The NOMCAR option does not affect the inclusion of observations that have missing values in the WEIGHT, FREQ, CLUSTER, STRATA, or DOMAIN variables. Observations that have missing values of the WEIGHT and FREQ variables are always excluded from the analysis. Observations that have missing values of the CLUSTER, DOMAIN, or STRATA variables are excluded unless you specify the MISSING option.

The NOMCAR option applies only to Taylor series variance estimation. The replication methods, which you request by specifying VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE, do not use the NOMCAR option.

Degrees of Freedom

PROC SURVEYPHREG uses the degrees of freedom of the variance estimator to obtain t confidence limits and Wald-type F tests. The procedure computes the degrees of freedom based on the variance estimation method, the sample design, and the number of estimable parameters. For more information, see the section "Degrees of Freedom" on page 10270. This section describes how missing values can affect the computation of the degrees of freedom.

Taylor Series Variance Estimation

The degrees of freedom can depend on the number of clusters, the number of strata, and the number of observations. For Taylor series variance estimation, these numbers are based on the observations that are included in the analysis. These numbers do not count observations that are excluded from the analysis because they have missing values. If all values in a stratum are excluded from the analysis as missing values, then that stratum is called an *empty stratum*. Empty strata are not counted in the total number of strata for the analysis. Similarly, empty clusters and missing observations are not included in the totals counts of clusters and observations that are used to compute the degrees of freedom for the analysis.

If you specify the MISSING option, missing values are treated as valid nonmissing levels and are included in computing the degrees of freedom. If you specify the NOMCAR option for Taylor series variance estimation, observations that have missing values for variables in the regression model are included in computing the degrees of freedom.

Replicate-Based Variance Estimation

For BRR or jackknife variance estimation, by default PROC SURVEYPHREG computes the degrees of freedom by using all valid observations in the input data set. A valid observation is an observation that has a positive value of the WEIGHT variable and nonmissing values of the STRATA and CLUSTER variables unless you specify the MISSING option.

Variance Estimation

PROC SURVEYPHREG uses the Taylor series method or replication (resampling) methods to estimate sampling errors of estimators that are based on complex sample designs (Fuller 1975; Särndal, Swensson, and Wretman 1992; Wolter 2007; Rust 1985; Dippo, Fay, and Morganstein 1984; Rao and Shao 1999, 1996; and Binder 1992). You can use the VARMETHOD= option in the PROC statement to specify the variance estimation method. By default, PROC SURVEYPHREG uses the Taylor series method.

However, replication methods have recently gained popularity for estimating variances in complex survey data analysis. One reason for this popularity is the relative simplicity of replication-based estimates, especially for nonlinear estimators; another is that modern computational capacity has made replication methods feasible for practical survey analysis.

Replication methods draw multiple replicates (also called subsamples) from a full sample according to a specific resampling scheme. The most commonly used resampling schemes are the *balanced repeated replication* (BRR) method and the *jackknife* method. For each replicate, the original weights are modified for the PSUs in the replicates to create replicate weights. The parameters of interest are estimated by using the replicate weights for each replicate. Then the variances of parameters of interest are estimated by the variability among the estimates derived from these replicates. The procedure automatically creates replicate weights based on the replication method you specify; alternatively you can use the REPWEIGHTS statement to provide your own replicate weights for variance estimation.

The following sections provide details about how the variance-covariance matrix of the estimated regression coefficients is estimated for each variance estimation method.

Taylor Series Linearization

The Taylor series linearization method is the default variance estimation method used by PROC SURVEYPHREG. See the section "Notation and Estimation" on page 10247 for definitions of the notation used in this section. Let

$$S^{(r)}(\boldsymbol{\beta}, t) = \sum_{A} w_{hij}\, y_{hij}(t) \exp\left(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t)\right) \mathbf{Z}_{hij}^{\otimes r}(t)$$

where $r = 0, 1$. Let A be the set of indices in the selected sample. Let

$$\mathbf{a}^{\otimes r} = \begin{cases} \mathbf{a}\mathbf{a}' & , \quad r = 1 \\ I_{\dim(\mathbf{a})} & , \quad r = 0 \end{cases}$$

and let $I_{\dim(\mathbf{a})}$ be the identity matrix of appropriate dimension.

Let $\bar{\mathbf{Z}}(\boldsymbol{\beta}, t) = \frac{S^{(1)}(\boldsymbol{\beta},t)}{S^{(0)}(\boldsymbol{\beta},t)}$. The score residual for the (h, i, j) subject is

$$\begin{aligned} \mathbf{L}_{hij}(\boldsymbol{\beta}) &= \Delta_{hij}\left\{\mathbf{Z}_{hij}(t_{hij}) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, t_{hij})\right\} \\ &\quad - \sum_{(h',i',j')\in A} \Delta_{h'i'j'} \frac{w_{h'i'j'} Y_{hij}(t_{h'i'j'}) \exp\left(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t_{h'i'j'})\right)}{S^{(0)}(\boldsymbol{\beta}, t_{h'i'j'})} \left\{\mathbf{Z}_{hij}(t_{h'i'j'}) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, t_{h'i'j'})\right\} \end{aligned}$$

For TIES=EFRON, the computation of the score residuals is modified to comply with the Efron partial likelihood. See the section "Residuals" on page 10277 for more information.

The Taylor series estimate of the covariance matrix of $\hat{\boldsymbol{\beta}}$ is

$$\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})\mathbf{G}\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})$$

where $\mathcal{I}(\hat{\boldsymbol{\beta}})$ is the observed information matrix and the $p \times p$ matrix $\mathbf{G}$ is defined as

$$\mathbf{G} = \frac{n-1}{n-p}\sum_{h=1}^{H}\frac{n_h(1-f_h)}{n_h-1}\sum_{i=1}^{n_h}(\mathbf{e}_{hi+} - \bar{\mathbf{e}}_{h\cdot\cdot})'(\mathbf{e}_{hi+} - \bar{\mathbf{e}}_{h\cdot\cdot})$$

The observed residuals, their sums and means are defined as follows:

$$\begin{aligned}
\mathbf{e}_{hij} &= w_{hij}\mathbf{L}_{hij}(\hat{\boldsymbol{\beta}}) \\
\mathbf{e}_{hi+} &= \sum_{j=1}^{m_{hi}}\mathbf{e}_{hij} \\
\bar{\mathbf{e}}_{h\cdot\cdot} &= \frac{1}{n_h}\sum_{i=1}^{n_h}\mathbf{e}_{hi+}
\end{aligned}$$

The factor $(n-1)/(n-p)$ in the computation of the matrix $\mathbf{G}$ reduces the small sample bias that is associated with using the estimated function to calculate deviations (Fuller et al. (1989), pp. 77–81). For simple random sampling, this factor contributes to the degrees of freedom correction applied to the residual mean square for ordinary least squares in which p parameters are estimated. By default, the procedure uses this adjustment in the variance estimation. If you do not want to use this multiplier in the variance estimator, then specify the VADJUST=NONE option in the MODEL statement.

Balanced Repeated Replication (BRR) Method

The balanced repeated replication (BRR) method requires that the full sample be drawn by using a stratified sample design with two primary sampling units (PSUs) per stratum. The BRR method constructs half-sample replicates by deleting one PSU per stratum according to a Hadamard matrix and doubling the original weight of the other PSU in that stratum. Let H be the total number of strata. The total number of replicates R is the smallest multiple of 4 that is greater than H. However, if you prefer a larger number of replicates, you can specify the REPS=*n* *method-option*. If a $n \times n$ Hadamard matrix cannot be constructed, the number of replicates is increased until a Hadamard matrix becomes available.

Each replicate is obtained by deleting one PSU per stratum according to a corresponding Hadamard matrix and adjusting the original weights for the remaining PSUs. The new weights are called replicate weights.

Replicates are constructed by using the first H columns of the $R \times R$ Hadamard matrix. The rth ($r = 1, 2, \ldots, R$) replicate is drawn from the full sample according to the rth row of the Hadamard matrix as follows:

- If the (r, h) element of the Hadamard matrix is 1, then the first PSU of stratum h is included in the rth replicate and the second PSU of stratum h is excluded.
- If the (r, h) element of the Hadamard matrix is –1, then the second PSU of stratum h is included in the rth replicate and the first PSU of stratum h is excluded.

The replicate weights of the remaining PSUs in each half sample are then doubled to their original weights. For more detail about the BRR method, see Wolter (2007) and Lohr (2022).

By default, an appropriate Hadamard matrix is generated automatically to create the replicates. You can display the Hadamard matrix by specifying the VARMETHOD=BRR(PRINTH) *method-option*. If you provide a Hadamard matrix by specifying the VARMETHOD=BRR(HADAMARD=) *method-option*, then the replicates are generated according to the provided Hadamard matrix. You can use the VARMETHOD=BRR(OUTWEIGHTS=) *method-option* to store the replicate weights in a SAS data set.

Let $\hat{\boldsymbol{\beta}}$ be the estimated proportional hazards regression coefficients from the full sample, and let $\hat{\boldsymbol{\beta}}_r$ be the estimated proportional hazards regression coefficients from the rth replicate by using replicate weights. PROC SURVEYPHREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with H degrees of freedom, where H is the number of strata.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYPHREG computes the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\beta}}_r - \overline{\hat{\boldsymbol{\beta}}_r}\right)\left(\hat{\boldsymbol{\beta}}_r - \overline{\hat{\boldsymbol{\beta}}_r}\right)'$$

where $\overline{\hat{\boldsymbol{\beta}}_r}$ is the average of the replicate estimates as follows:

$$\overline{\hat{\boldsymbol{\beta}}_r} = \frac{1}{R}\sum_{r=1}^{R}\hat{\boldsymbol{\beta}}_r$$

If one or more components of $\hat{\boldsymbol{\beta}}_r$ cannot be calculated for some replicates, then the variance estimate is computed by using only the replicates for which the proportional hazards regression coefficients can be estimated. Estimability and nonconvergence are the two most common reasons why $\hat{\boldsymbol{\beta}}_r$ might not be available for a replicate sample even if $\hat{\boldsymbol{\beta}}$ is defined for the full sample. Let R_a be the number of replicates where $\hat{\boldsymbol{\beta}}_r$ is available, and let $R - R_a$ be the number of replicates where $\hat{\boldsymbol{\beta}}_r$ is not available. Without loss of generality, assume that $\hat{\boldsymbol{\beta}}_r$ is available only for the first R_a replicates; then the BRR variance estimator is

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R_a}\sum_{r=1}^{R_a}\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with degrees of freedom equal to the minimum of H and R_a, where H is the number of strata. Alternatively, you can use the FAY= *method-option* to request Fay's BRR method, as discussed in the following section.

Fay's BRR Method

The traditional BRR method constructs half-sample replicates by deleting one PSU per stratum according to a Hadamard matrix and doubling the original weight of the other PSU. Fay's BRR method uses the Fay coefficient, ϵ $(0 \le \epsilon < 1)$, and instead of deleting one PSU per stratum, it multiplies the original weight by

the coefficient ϵ. The original weight of the remaining PSU in that stratum is multiplied by $2 - \epsilon$. PROC SURVEYPHREG uses $\epsilon = 0.5$ as the default value; alternatively, you can specify a value for ϵ with the FAY= *method-option*. When $\epsilon = 0$, Fay's method becomes the traditional BRR method. For more details, see Dippo, Fay, and Morganstein (1984); Fay (1984, 1989); Judkins (1990). Because the traditional BRR method uses only half of the total sample in every replicate, several replicate estimators ($\hat{\boldsymbol{\beta}}_r$) might be undefined even when the full sample estimator ($\hat{\boldsymbol{\beta}}$) is defined. Fay's BRR method is especially useful for this situation because it uses all the sampled units in every replicate.

Let $\hat{\boldsymbol{\beta}}$ be the estimated proportional hazards regression coefficients from the full sample, and let $\hat{\boldsymbol{\beta}}_r$ be the estimated regression coefficients that are obtained from the rth replicate by using replicate weights. PROC SURVEYPHREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R(1-\epsilon)^2} \sum_{r=1}^{R} \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right) \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with H degrees of freedom, where H is the number of strata.

Hadamard Matrix

PROC SURVEYPHREG uses a Hadamard matrix to construct replicates for BRR variance estimation. You can provide a Hadamard matrix for replicate construction by using the HADAMARD= *method-option* for VARMETHOD=BRR. Otherwise, PROC SURVEYPHREG generates an appropriate Hadamard matrix. You can display the Hadamard matrix by specifying the PRINTH *method-option*.

A Hadamard matrix $\mathbf{A}$ of dimension R is a square matrix that has all elements equal to 1 or –1 such that $\mathbf{A}'\mathbf{A} = R\mathbf{I}$, where $\mathbf{I}$ is an identity matrix of appropriate order. The dimension of a Hadamard matrix must equal 1, 2, or a multiple of 4.

For example, the following matrix is a Hadamard matrix of dimension $k = 8$:

$$\begin{array}{rrrrrrrr}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{array}$$

For BRR replicate construction, the dimension of the Hadamard matrix must be at least H, where H denotes the number of first-stage strata in your design. If a Hadamard matrix of a given dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a SAS data set in the HADAMARD= *method-option*. You must ensure that the matrix that you provide is actually a Hadamard matrix; PROC SURVEYPHREG does not check the validity of your Hadamard matrix.

See the section "Balanced Repeated Replication (BRR) Method" on page 10263 for details about how the Hadamard matrix is used to construct replicates for BRR variance estimation.

Bootstrap Method

The naive bootstrap variance estimator that is suitable for infinite population is not consistent when applied to complex surveys. Bootstrap replicate samples for complex surveys are created by using a simple random sample with replacement of primary sampling units (PSUs) within each stratum. PSUs in different strata are sampled independently. The original sampling weights are then adjusted in each replicate to reflect the full sample. These adjusted weights are also called bootstrap replicate weights. McCarthy and Snowden (1985), Rao and Wu (1988), Sitter (1992b), and Sitter (1992a) provide several adjusted bootstrap variance estimators that are consistent for complex surveys. For more information about bootstrap variance estimation for complex surveys, see Mashreghi, Haziza, and Léger (2016), Beaumont and Patak (2012), Lohr (2022, Section 9.3.3), Fuller (2009, Section 4.5), Wolter (2007, Chapter 5), and Shao and Tu (1995, Section 6.2.4).

If you do not provide replicate weights by using the REPWEIGHTS statement, then the BOOTSTRAP option in the PROC SURVEYPHREG statement creates bootstrap replicate weights for you. This bootstrap method is similar to the method of Rao, Wu, and Yue (1992) and is also known as bootstrap weights method (Mashreghi, Haziza, and Léger 2016).

Each replicate is obtained by selecting a simple random sample with replacement of m_h PSUs from stratum h. The rth bootstrap replicate weight for observation unit j in PSU i and stratum h is given by

$$w_{hij}^{(r)} = w_{hij}\left\{1 - \sqrt{(1-f_h)m_h/(n_h-1)} + \sqrt{(1-f_h)m_h/(n_h-1)}(n_h/m_h)k_{hi}^{(r)}\right\}$$

where $k_{hi}^{(r)}$ is the number of times PSU i is selected in replicate sample r, and f_h is the sampling fraction in stratum h.

Let $\hat{\boldsymbol{\beta}}$ be the estimated proportional hazards regression coefficients from the full sample, and let $\hat{\boldsymbol{\beta}}_r$ be the estimated proportional hazards regression coefficients from the rth replicate by using replicate weights. PROC SURVEYPHREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with $\sum_{h=1}^{H} n_h - H$ degrees of freedom, where n_h is the number of PSUs in stratum h, and H is the number of strata.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYPHREG computes the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\beta}}_r - \overline{\hat{\boldsymbol{\beta}}_r}\right)\left(\hat{\boldsymbol{\beta}}_r - \overline{\hat{\boldsymbol{\beta}}_r}\right)'$$

where $\overline{\hat{\boldsymbol{\beta}}_r}$ is the average of the replicate estimates as follows:

$$\overline{\hat{\boldsymbol{\beta}}_r} = \frac{1}{R}\sum_{r=1}^{R}\hat{\boldsymbol{\beta}}_r$$

If one or more components of $\hat{\boldsymbol{\beta}}_r$ cannot be calculated for some replicates, then the variance estimate is computed by using only the replicates for which the proportional hazards regression coefficients can be

estimated. Estimability and nonconvergence are the two most common reasons why $\hat{\boldsymbol{\beta}}_r$ might not be available for a replicate sample even if $\hat{\boldsymbol{\beta}}$ is defined for the full sample. Let R_a be the number of replicates where $\hat{\boldsymbol{\beta}}_r$ is available, and let $R - R_a$ be the number of replicates where $\hat{\boldsymbol{\beta}}_r$ is not available. Without loss of generality, assume that $\hat{\boldsymbol{\beta}}_r$ is available only for the first R_a replicates; then the bootstrap variance estimator is

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R_a} \sum_{r=1}^{R_a} \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right) \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with degrees of freedom equal to the minimum of $\sum_{h=1}^{H} n_h - H$ and R_a, where n_h is the number of PSUs in stratum h, and H is the number of strata.

Although PROC SURVEYPHREG creates bootstrap weights only from the bootstrap weights method (Rao, Wu, and Yue 1992), bootstrap weights that are generated from any bootstrap methods can be used in the REPWEIGHTS statement. If the bootstrap replicate weights are available to you for a survey, then you can use the REPWEIGHTS statement to name the variables that contain the bootstrap replicate weights and specify the VARMETHOD=BOOTSTRAP option in the PROC SURVEYPHREG statement. The SURVEYPHREG procedure uses $1/R$ as the default bootstrap replicate coefficient when you specify the VARMETHOD=BOOTSTRAP option, where R is the total number of replicates. Alternatively, you can specify different replicate coefficients by using the REPCOEFS= option in the REPWEIGHTS statement.

For more information, see the section "Replicate Weights Method" on page 10268.

Jackknife Method

The jackknife method of variance estimation deletes one PSU at a time from the full sample to create replicates. This method is also known as the delete-1 jackknife method because it deletes exactly one PSU in every replicate. The total number of replicates R is the same as the total number of PSUs. In each replicate, the sampling weights of the remaining PSUs are modified by the *jackknife coefficient* α_r. The modified weights are called replicate weights.

Let PSU i in stratum h_r be omitted for the rth replicate; then the jackknife coefficient and replicate weights are computed as

$$\alpha_r = \begin{cases} \frac{n_{h_r}-1}{n_{h_r}} & \text{for a stratified design} \\ \frac{R-1}{R} & \text{for designs without stratification} \end{cases}$$

and

$$w_{hij}^{(r)} = \begin{cases} w_{hij} & \text{if observation unit } j \text{ is not in donor stratum } h_r \\ 0 & \text{if observation unit } j \text{ is in PSU } i \text{ of donor stratum } h_r \\ w_{hij}/\alpha_r & \text{if observation unit } j \text{ is not in PSU } i \text{ but in donor stratum } h_r \end{cases}$$

You can use the VARMETHOD=JACKKNIFE(OUTJKCOEFS=) *method-option* to store the jackknife coefficients in a SAS data set and use the VARMETHOD=JACKKNIFE(OUTWEIGHTS=) *method-option* to store the replicate weights in a SAS data set.

If you provide your own replicate weights with a REPWEIGHTS statement, then you can also provide corresponding jackknife coefficients with the JKCOEFS= option. If you provide replicate weights with a

REPWEIGHTS statement but do not provide jackknife coefficients, then the procedure uses $(R-1)/R$ as the default jackknife coefficient for every replicate, where R is the total number of replicates.

Let $\hat{\boldsymbol{\beta}}$ be the estimated proportional hazards regression coefficients from the full sample, and let $\hat{\boldsymbol{\beta}}_r$ be the estimated regression coefficients for the rth replicate. PROC SURVEYPHREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with $R-H$ degrees of freedom, where R is the number of replicates and H is the number of strata, or $R-1$ when there is no stratification.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYPHREG computes the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \overline{\hat{\boldsymbol{\beta}}_r}\right)\left(\hat{\boldsymbol{\beta}}_r - \overline{\hat{\boldsymbol{\beta}}_r}\right)'$$

where $\overline{\hat{\boldsymbol{\beta}}_r}$ is the average of the replicate estimates as follows:

$$\overline{\hat{\boldsymbol{\beta}}_r} = \frac{1}{R}\sum_{r=1}^{R} \hat{\boldsymbol{\beta}}_r$$

If one or more components of $\hat{\boldsymbol{\beta}}_r$ cannot be calculated for some replicates, then the variance estimator uses only the replicates for which the proportional hazards regression coefficients can be estimated. Estimability and nonconvergence are two common reasons why $\hat{\boldsymbol{\beta}}_r$ might not be available for a replicate sample even if $\hat{\boldsymbol{\beta}}$ is defined for the full sample. Let R_a be the number of replicates where $\hat{\boldsymbol{\beta}}_r$ are available, and let $R-R_a$ be the number of replicates where $\hat{\boldsymbol{\beta}}_r$ are not available. Without loss of generality, assume that $\hat{\boldsymbol{\beta}}_r$ is available only for the first R_a replicates; then the jackknife variance estimator is

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R_a} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with R_a-H degrees of freedom, where H is the number of strata. Alternatively, you can use the VADJUST=AVGREPSS option in the MODEL statement to use the average sum of squares for the invalid replicate samples. See "Variance Adjustment Factors" on page 10271 for details.

Replicate Weights Method

The replicate weights variance estimation method is a general-purpose variance estimation method that uses the replicate weights and replicate coefficients that you provide by using the REPWEIGHTS statement and the REPCOEFS= option, respectively.

If you provide your own replicate weights in a REPWEIGHTS statement but do not specify replicate coefficients in a REPCOEFS= option, then the default replicate coefficient depends on the VARMETHOD= option in the PROC SURVEYPHREG statement as shown in the following table:

Value of VARMETHOD=	Default Replicate Coefficient
None specified	$(R-1)/R$
BOOTSTRAP	$1/R$
BRR	$1/R$
JACKKNIFE	$(R-1)/R$

Let $\hat{\boldsymbol{\beta}}$ be the estimated proportional hazards regression coefficients from the full sample, and let $\hat{\boldsymbol{\beta}}_r$ and α_r be the estimated regression coefficients and the replicate coefficient for the rth replicate, respectively. PROC SURVEYPHREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with R degrees of freedom, where R is the number of replicates.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYPHREG computes the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \overline{\hat{\boldsymbol{\beta}}_r}\right)\left(\hat{\boldsymbol{\beta}}_r - \overline{\hat{\boldsymbol{\beta}}_r}\right)'$$

where $\overline{\hat{\boldsymbol{\beta}}_r}$ is the average of the replicate estimates as follows:

$$\overline{\hat{\boldsymbol{\beta}}_r} = \frac{1}{R}\sum_{r=1}^{R} \hat{\boldsymbol{\beta}}_r$$

If one or more components of $\hat{\boldsymbol{\beta}}_r$ cannot be calculated for some replicates, then the variance estimator uses only the replicates for which the proportional hazards regression coefficients can be estimated. Estimability and nonconvergence are two common reasons why $\hat{\boldsymbol{\beta}}_r$ might not be available for a replicate sample even if $\hat{\boldsymbol{\beta}}$ is defined for the full sample. Let R_a be the number of replicates where $\hat{\boldsymbol{\beta}}_r$ are available, and let $R - R_a$ be the number of replicates where $\hat{\boldsymbol{\beta}}_r$ are not available. Without loss of generality, assume that $\hat{\boldsymbol{\beta}}_r$ are available only for the first R_a replicates; then the jackknife variance estimator is

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R_a} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with R_a degrees of freedom. Alternatively, you can use the VADJUST=AVGREPSS option in the MODEL statement to use the average sum of squares for the invalid replicate samples. For more information, see "Variance Adjustment Factors" on page 10271.

Degrees of Freedom

PROC SURVEYPHREG uses the degrees of freedom of the variance estimator to obtain *t* confidence limits and Wald-type *F* tests. The procedure computes the degrees of freedom based on the variance estimation method, the sample design, and the number of estimable parameters. Alternatively, you can specify the degrees of freedom by using the DF= option in the MODEL statement. This section describes how PROC SURVEYPHREG computes different values of the degrees of freedom based on the variance estimation method and the sample design. For more information about how degrees of freedom depend on the number of estimable parameters and the DF= option in the MODEL statement, see the section "Hypothesis Tests, Confidence Intervals, and Residuals" on page 10274.

For Taylor series variance estimation, PROC SURVEYPHREG calculates the degrees of freedom (*df*) as the number of clusters minus the number of strata. If the CLUSTER statement is not specified, then the procedure treats each observation as a cluster. If the STRATA statement is not specified, then the procedure assumes that all observations are in the same stratum. These numbers are based on the observations that are included in the analysis. These numbers do not count observations that are excluded from the analysis because they have missing values. For more information, see the section "Missing Values" on page 10259. If you specify the MISSING option in the CLASS statement, missing values are treated as valid nonmissing levels and are included in computing the degrees of freedom. If you specify the NOMCAR option for Taylor series variance estimation, observations that have missing values of the analysis variables are included in computing the degrees of freedom.

If you provide replicate weights by using the REPWEIGHTS statement, the degrees of freedom is equal to the number of replicates used, which is the number of REPWEIGHTS variables that provide replicate estimates. Alternatively, you can specify DF=ALLREPS in the MODEL statement to specify that *df* equals the number of replicates. For bootstrap variance estimation, the number of replicates is typically much larger than the number of independent variance units. Use the DF= option in the MODEL statement to specify that *df* equals the number of independent variance units from your survey design.

For BRR variance estimation (when you do not use the REPWEIGHTS statement), PROC SURVEYPHREG calculates the degrees of freedom as the number of strata. The procedure bases the number of strata on all valid observations in the data set. If some replicate samples are not usable, in the sense that they cannot be used for parameter estimation because of factors such as nonconvergence or inestimability, then *df* equals the minimum of the number of strata and the number of replicates used. Alternatively, you can specify DF=ALLREPS in the MODEL statement to specify that *df* equals the number of strata.

For bootstrap variance estimation (when you do not use the REPWEIGHTS statement), PROC SURVEYPHREG calculates the degrees of freedom as the number of clusters minus the number of strata. If you do not specify the CLUSTER statement, then the procedure treats each observation as a cluster. If you do not specify the STRATA statement, then the procedure assumes that all observations are in the same stratum. For bootstrap variance estimation, PROC SURVEYPHREG bases the number of strata and clusters on all valid observations in the data set. If some replicate samples are not usable, in the sense that they cannot be used for parameter estimation because of factors such as nonconvergence or inestimability, then *df* equals the lesser of the number of clusters minus the number of strata and the number of replicate samples that produces parameter estimates. Alternatively, you can specify DF=ALLREPS in the MODEL statement to specify that *df* equals the number of clusters minus the number of strata.

For jackknife variance estimation (when you do not use the REPWEIGHTS statement), PROC SURVEYPHREG calculates the degrees of freedom as the number of clusters minus the number of strata. If you do not specify the CLUSTER statement, then the procedure treats each observation as a cluster. If you do not specify the STRATA statement, then the procedure assumes that all observations are in the same stratum.

For jackknife variance estimation, PROC SURVEYPHREG bases the number of strata and clusters on all valid observations in the data set. If some replicate samples are not usable, in the sense that they cannot be used for parameter estimation because of factors such as nonconvergence or inestimability, then *df* equals the number of clusters (or observations if no CLUSTER statement is specified) minus the number of strata (or 1 if no STRATA statement is specified) minus the number of replicate samples that are not used. Alternatively, you can specify DF=ALLREPS in the MODEL statement to specify that *df* equals the number of clusters minus the number of strata.

Variance Adjustment Factors

PROC SURVEYPHREG provides options for adjusting the default variance estimators. VADJUST=NONE and VADJUST=DF are available for the Taylor series linearization variance estimator. VADJUST=AVGREPSS is available for the jackknife replication variance estimators.

For models with large number of parameters, it is reasonable to adjust the Taylor series linearized variance estimator by the number of estimable parameters in the analysis model. Fuller et al. (1989, pp. 77–81) use an adjustment factor $(n-1)/(n-p)$ to estimate the linearized variance for regression coefficients, where n is the total number of observation units and p is the number of estimable parameters in the analysis model. By default, PROC SURVEYPHREG uses this adjustment in the computation of the matrix $\mathbf{G}$ for the Taylor series linearization variance estimation. If you do not want to use this adjustment, then specify VADJUST=NONE.

Variance adjustment factors can be useful for replication variance estimations, especially if some replicate samples are not usable. A replicate sample might not provide useful parameter estimates (replicate estimates) for reasons such as nonconvergence of the optimization or inestimability of some parameters in that subsample. For example, consider the jackknife variance estimator with R replicates. Suppose that only $R_a (< R)$ replicates are used to obtain replicate estimates and $R - R_a$ replicates cannot be used due to, say, nonconvergence of the optimization. Without loss of generality, assume that the first R_a replicates are used. By default SURVEYPHREG uses

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R_a} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

as the jackknife variance estimator. An alternative estimator is

$$\begin{aligned}\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) &= \sum_{r=1}^{R_a} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)' + (R - R_a)\left\{\frac{1}{R_a}\sum_{r=1}^{R_a} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'\right\} \\ &= \frac{R}{R_a}\sum_{r=1}^{R_a} \alpha_r (\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}})(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}})'\end{aligned}$$

which uses the average replicate sum of squares for the $R - R_a$ unusable replicate samples. If you specify the VADJUST=AVGREPSS option, PROC SURVEYPHREG uses the second variance estimator for the jackknife replication method. Note that you can specify the FAY *method-option* for the BRR method to avoid nonconvergence of the optimization or inestimability of some parameters in subsamples.

Variance Ratios and Standard Error Ratios

PROC SURVEYPHREG provides options to compute the following variance ratios and standard error ratios:

- If you specify the VARRATIO=MODEL option, then the procedure computes the variance ratio of the estimated regression parameter $\hat{\beta}_j$ as $\frac{\hat{V}(\hat{\beta}_j)}{\hat{V}_M(\hat{\beta}_j)}$, where $\hat{V}(\hat{\beta}_j)$, the estimated variance of $\hat{\beta}_j$, uses the complete design information and $\hat{V}_M(\hat{\beta}_j)$ is the jth diagonal element of the observed information matrix $\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})$.

- If you specify the VARRATIO=IND option, then the procedure computes the variance ratio of the estimated regression parameter $\hat{\beta}_j$ as $\frac{\hat{V}(\hat{\beta}_j)}{\hat{V}_{\mathrm{IND}}(\hat{\beta}_j)}$, where $\hat{V}(\hat{\beta}_j)$, the estimated variance of $\hat{\beta}_j$, uses the complete design information and $\hat{V}_{\mathrm{IND}}(\hat{\beta}_j)$ is the jth diagonal element of $\hat{V}_{\mathrm{IND}}(\hat{\boldsymbol{\beta}})$. $\hat{V}_{\mathrm{IND}}(\hat{\boldsymbol{\beta}})$ is the sandwich variance estimator, which ignores the strata and the clusters and is computed as

$$\hat{V}_{\mathrm{IND}}(\hat{\boldsymbol{\beta}}) = \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \left\{ \frac{n}{n-1}(1-f) \sum_h \sum_i \sum_j (\mathbf{e}_{hij} - \bar{\mathbf{e}}_{\cdots})'(\mathbf{e}_{hij} - \bar{\mathbf{e}}_{\cdots}) \right\} \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}})$$

  where e_{hij} are the weighted score residuals, f is the overall sampling fraction, and n is the number of observation units. The three sums are over the observation units (j) across the PSUs (i) and the strata (h).

- If you specify the VARRATIO=SRSWR option, then the procedure computes the variance ratio of the estimated regression parameter $\hat{\beta}_j$ as $\frac{\hat{V}(\hat{\beta}_j)}{\hat{V}_{\mathrm{SRSWR}}(\hat{\beta}_j)}$, where $\hat{V}(\hat{\beta}_j)$, the estimated variance of $\hat{\beta}_j$, uses the complete design information and $\hat{V}_{\mathrm{SRSWR}}(\hat{\beta}_j)$ is the jth diagonal element of $\hat{V}_{\mathrm{SRSWR}}(\hat{\boldsymbol{\beta}})$. $\hat{V}_{\mathrm{SRSWR}}(\hat{\boldsymbol{\beta}})$ is given by

$$\hat{V}_{\mathrm{SRSWR}}(\hat{\boldsymbol{\beta}}) = \mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \sum_h \sum_i \sum_j w_{hij}/n$$

  where w_{hij} is the observation weight for unit (h, i, j) and n is the number of observation units. The three sums are over the observation units (j) across the PSUs (i) and the strata (h). The matrix $[\hat{V}_{\mathrm{SRSWR}}(\hat{\boldsymbol{\beta}})]^{-1}\hat{V}(\hat{\boldsymbol{\beta}})$ is often called the generalized design effect matrix (Rao, Scott, and Skinner 1998).

- If you specify the VARRATIO=SRSWOR option, then the procedure computes the variance ratio of the estimated regression parameter $\hat{\beta}_j$ as $\frac{\hat{V}(\hat{\beta}_j)}{\hat{V}_{\mathrm{SRSWOR}}(\hat{\beta}_j)}$, where $\hat{V}(\hat{\beta}_j)$, the estimated variance of $\hat{\beta}_j$, uses the complete design information and $\hat{V}_{\mathrm{SRSWOR}}(\hat{\beta}_j)$ is the jth diagonal element of $\hat{V}_{\mathrm{SRSWOR}}(\hat{\boldsymbol{\beta}})$. $\hat{V}_{\mathrm{SRSWOR}}(\hat{\boldsymbol{\beta}})$ is given by

$$\hat{V}_{\mathrm{SRSWR}}(\hat{\boldsymbol{\beta}}) = (1-f)\mathcal{I}^{-1}(\hat{\boldsymbol{\beta}}) \sum_h \sum_i \sum_j w_{hij}/n$$

  where w_{hij} is the observation weight for unit (h, i, j), f is the overall sampling fraction, and n is the number of observation units. The three sums are over the observation units (j) across the PSUs (i) and the strata (h).

For Taylor series or bootstrap variance estimation, PROC SURVEYPHREG determines the value of f, the overall sampling fraction, based on the RATE= or TOTAL= option. If you do not specify either of these options, PROC SURVEYPHREG assumes that the value of f is negligible and does not use a finite population correction in the analysis. If you specify RATE=*value*, PROC SURVEYPHREG uses *value* as the overall sampling fraction f. If you specify TOTAL=*value*, PROC SURVEYPHREG computes f as the ratio of the number of PSUs in the sample to the specified total.

If you specify stratum sampling rates by using the RATE=*SAS-data-set* option, then PROC SURVEYPHREG computes stratum totals based on these stratum sampling rates and the number of sample PSUs in each stratum. The procedure sums the stratum totals to form the overall total and then computes f as the ratio of the number of sample PSUs to the overall total. Alternatively, if you specify stratum totals with the TOTAL=*SAS-data-set* option, then PROC SURVEYPHREG sums these totals to compute the overall total. The overall sampling fraction f is then computed as the ratio of the number of sample PSUs to the overall total.

The replication methods do not use the finite population correction factor $(1 - f)$ in the denominator.

Standard error ratios are computed as the square root of the variance ratios.

Domain Analysis

Domain analysis refers to the computation of statistics for domains (subpopulations). Formation of subpopulations can be unrelated to the sample design, and so the domain sample sizes can actually be random variables. Domain analysis takes this variability into account to compute variance estimates for estimated model parameters. Domain analysis is also known as subgroup analysis, subpopulation analysis, and subdomain analysis. For more information about domain analysis, see Lohr (2022); Särndal, Swensson, and Wretman (1992); Cochran (1977).

To request domain analysis with PROC SURVEYPHREG, use the DOMAIN statement. If your domains are formed by more than one variable, you can specify DomainVariable_1 * DomainVariable_2 in the DOMAIN statement. If you use the DOMAIN statement, the procedure performs separate analyses for all domains, in addition to the overall analysis.

Including the domain variables in a DOMAIN statement request provides a different analysis from that obtained by using a BY statement, which provides completely separate analyses of the BY groups. The BY statement can also be used to analyze the data set by subgroups, but it is critical to note that this does *not* account for random sample sizes that often occur for domain analyses. The BY statement is appropriate only when the number of units in each subgroup is known with certainty. For example, the BY statement can be used to obtain stratum level estimates when you have fixed sample sizes for the strata. When the subgroup sample size is random, include the domain variables in DOMAIN statement.

Hypothesis Tests, Confidence Intervals, and Residuals

Testing the Global Null Hypothesis

The following statistics are available to test the global null hypothesis $H_0\colon \boldsymbol{\beta} = \mathbf{0}$. Let d be the usual degrees of freedom computed from the survey data by using the number of strata, clusters, or replicate weights; and let p be the number of estimable parameters in the null hypothesis H_0. For more information about computing d, see the section "Degrees of Freedom" on page 10270.

The unadjusted likelihood ratio test statistic is expressed as

$$\chi^2_{\mathrm{LR}} = 2\left[\log\left\{\mathrm{L}(\hat{\boldsymbol{\beta}})\right\} - \log\left\{\mathrm{L}(\mathbf{0})\right\}\right]$$

where $\mathrm{L}(\cdot)$ denotes the partial pseudo-likelihood that is described in the section "Partial Likelihood Function for the Cox Model" on page 10252 and $\hat{\boldsymbol{\beta}}$ denotes the estimated regression parameters. The p-value for the unadjusted test is computed by using a chi-square distribution with p degrees of freedom.

The unadjusted likelihood ratio statistic is sensitive to the scaling of the weights. PROC SURVEYPHREG computes an adjusted likelihood ratio test statistic that is invariant to the scaling of the weights. The adjusted test is similar to the second-order adjusted Rao-Scott chi-square tests. For more information, see Rao, Scott, and Skinner (1998), and Lumley and Scott (2013). The adjusted likelihood ratio test statistic is expressed as

$$\chi^2_{\mathrm{RS2}} = \frac{(n/\hat{N})\chi^2_{\mathrm{LR}}}{\bar{\delta}_{.}(1+\hat{a}^2)}$$

where $\delta_1, \delta_2, \ldots, \delta_r$ are the positive eigenvalues from the generalized design effect matrix ("Variance Ratios and Standard Error Ratios" on page 10272), $\bar{\delta}_{.} = 1/r\sum_{i=1}^{r}\delta_i$ is the mean of the positive eigenvalues, $\hat{a}^2 = (r-1)^{-1}\sum_{i=1}^{r}(\delta_i - \bar{\delta}_{.})^2/\bar{\delta}_{.}^2$ is the squared coefficient of variations of the positive eigenvalues, n is the number of observation units, and $\hat{N} = \sum_{hij} w_{hij}$ is the sum of the weights over all observation units. The p-value for the adjusted test is computed by using a chi-square distribution with $r/(1+\hat{a}^2)$ degrees of freedom.

The usual assumptions that are required for a likelihood ratio test do not hold for the pseudo-likelihood that is used by PROC SURVEYPHREG (Rao, Scott, and Skinner 1998), leading to other methods of testing the global null hypothesis, such as the Wald test discussed in the following paragraph.

The Wald test uses the variance estimator that accounts for complex sampling such as stratification, clustering, and unequal weighting. Let $Q = \hat{\boldsymbol{\beta}}'\left[\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\right]^{-1}\hat{\boldsymbol{\beta}}$, where $\hat{\boldsymbol{\beta}}$ is the estimated regression parameters and $\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}})$ is the estimated covariance matrix for $\hat{\boldsymbol{\beta}}$. You can request the Wald tests that are described in the following table by using the DF= option in the MODEL statement.

Value of DF=	Test Request	Test Statistic	Numerator Degrees of Freedom	Denominator Degrees of Freedom
NONE	Chi-square	Q	p	∞
v	Customized F	vQ/pd	p	v
DESIGN	Unadjusted F	Q/p	p	d
DESIGN (v)	Unadjusted F	Q/p	p	v
PARMADJ	Adjusted F	$(d-p+1)Q/pd$	p	$d-p+1$
PARMADJ (v)	Adjusted F	$(v-p+1)Q/pv$	p	$v-p+1$
DESIGNADJ	Adjusted F	$(d-p+1)Q/pd$	p	d

Model Fit Statistics

Suppose the model contains p estimable parameters. Then the following two criteria are displayed for model fit statistics:

- –2 log likelihood:

  $$-2 \text{ Log L} = -2 \log(L(\hat{\boldsymbol{\beta}}))$$

  where $L(.)$ is a partial pseudo-likelihood function for the corresponding TIES= option as described in the section "Partial Likelihood Function for the Cox Model" on page 10252, and $\hat{\boldsymbol{\beta}}$ is the maximum pseudo-log-likelihood estimate of the proportional hazards regression coefficients.

- Akaike's information criterion (AIC):

  $$\text{AIC} = -2 \text{ Log L} + 2p$$

The AIC statistic provides a different way of adjusting the log-likelihood statistic for the number of estimable parameters in the model.

Neither of these criteria is adjusted for the complex sample design, and both criteria are sensitive to the scale of the weights.

Contrasts and Hypothesis Tests

For a testable hypothesis $H_0\colon \mathbf{L}\boldsymbol{\beta} = 0$, you can request different Wald tests by using the DF= option in the MODEL statement.

Let

$$Q = (\mathbf{L}^*\hat{\boldsymbol{\beta}})'(\mathbf{L}^*\widehat{\mathbf{V}}\mathbf{L}^{*\prime})^{-}(\mathbf{L}^*\hat{\boldsymbol{\beta}})$$

where $\mathbf{L}$ is a contrast vector or matrix that you specify, $\boldsymbol{\beta}$ is the vector of regression parameters, $\hat{\boldsymbol{\beta}}$ is the estimated regression coefficients, $\widehat{\mathbf{V}}$ is the estimated covariance matrix of $\hat{\boldsymbol{\beta}}$, and $\mathbf{L}^*$ is a matrix such that the following are true:

- $\mathbf{L}^*$ has the same number of columns as $\mathbf{L}$.
- $\mathbf{L}^*$ has full row rank.
- The rank of $\mathbf{L}^*$ equals the rank of the $\mathbf{L}$ matrix.
- All rows of $\mathbf{L}^*$ are estimable functions.
- The Wald F statistic that is computed by using the $\mathbf{L}^*$ matrix is equivalent to the Wald F statistic computed by using the $\mathbf{L}$ matrix.

If $\mathbf{L}$ is a full-rank matrix and all rows of $\mathbf{L}$ are estimable functions, then $\mathbf{L}^*$ is the same as $\mathbf{L}$. It is possible that such an $\mathbf{L}^*$ matrix cannot be constructed for a given set of linear contrasts, in which case the contrasts are not testable. Let r be the rank of $\mathbf{L}\hat{\mathbf{V}}\mathbf{L}'$. Table 122.10 describes the Wald tests available in PROC SURVEYPHREG.

Table 122.10 Summary of Wald Tests

Value of DF=	Test Request	Test Statistic	Numerator Degrees of Freedom	Denominator Degrees of Freedom
NONE	Chi-square	Q	r	∞
v	Customized F	vQ/rd	r	v
DESIGN	Unadjusted F	Q/r	r	d
DESIGN (v)	Unadjusted F	Q/r	r	v
PARMADJ	Adjusted F	$(d-r+1)Q/rd$	r	$d-r+1$
PARMADJ (v)	Adjusted F	$(v-r+1)Q/rv$	r	$v-r+1$
DESIGNADJ	Adjusted F	Q/r	r	d

The preceding development for Wald tests assumes that the estimated variance of $\hat{\boldsymbol{\beta}}$, $\hat{\mathbf{V}}$, is of the form $\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}$ for some estimate $\mathbf{V}$ of the variance of $\mathbf{Y}$. In this case, estimability, $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}(\mathbf{X}'\mathbf{X}) = \mathbf{L}$, ensures that this F statistic has a unique value no matter which kind of generalized inverse is used to compute it. However, when a design-based variance estimator is used to estimate the variability of $\hat{\boldsymbol{\beta}}$, estimability does not ensure uniqueness. In this case, the F value is invariant to the choice of the generalized inverse if and only if $\mathbf{L}$ is estimable and $\mathbf{L}'\left[\mathbf{L}\hat{\mathbf{V}}\mathbf{L}'\right]^{-}\left[\mathbf{L}\hat{\mathbf{V}}\mathbf{L}'\right] = \mathbf{L}'$.

Although it is extremely rare, it is possible in practice that the preceding uniqueness condition is not satisfied. For example, if the number of clusters is less than the number of nonsingular parameters in the model, then the matrix of coefficients for testing the overall null does not satisfy the uniqueness condition. If this condition is not satisfied, then the F statistic for testing $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ is not invariant to the choice of the g_2-inverse of $\mathbf{L}\hat{\mathbf{V}}\mathbf{L}'$. In practical applications, the test statistic is compared with an F distribution as described in Table 122.10, but the value of the test statistic and therefore the inference might be different when a different g_2-inverse is used. This F test is not recommended when the uniqueness condition is not satisfied. An alternative approach would be to increase the number of clusters or to find a parsimonious model so that the number of parameters is less than the number of clusters.

Confidence Intervals

By default, the SURVEYPHREG procedure computes t confidence limits for the estimated regression coefficients. Alternatively, you can specify DF=NONE in the MODEL statement to request standard normal confidence intervals. The t confidence interval for a linear combination $\mathbf{l}'\boldsymbol{\beta}$ of the regression coefficients is computed as

$$\left(\mathbf{l}'\hat{\boldsymbol{\beta}} \pm t_{df,\alpha/2}\sqrt{\mathbf{l}'\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{l}}\right)$$

where $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$ percentile point of the t distribution with df degrees of freedom. See the section "Degrees of Freedom" on page 10270 for more information about df. If you use the DF=NONE option in the MODEL statement, then the procedure uses the $100(1-\alpha/2)$ percentile point of the standard normal distribution.

Hazard Ratios

The hazard ratio for a quantitative effect with regression coefficient $\beta_j = \mathbf{e}_j'\boldsymbol{\beta}$ is defined as $\exp(\beta_j)$, where $\mathbf{e}_j$ denotes the jth unit vector. In general, a log-hazard ratio can be written as $\mathbf{l}'\boldsymbol{\beta}$, a linear combination of the regression coefficients, and the hazard ratio $\exp(\mathbf{l}'\boldsymbol{\beta})$ is obtained by replacing e_j with $\mathbf{l}$.

The confidence intervals for hazard ratios are obtained by exponentiating the confidence limits of the corresponding linear combination. Thus, the $100(1-\alpha)$ confidence limits are

$$\exp\left(\mathbf{e}_j'\hat{\boldsymbol{\beta}} \pm t_{df,\alpha/2}\sqrt{\mathbf{e}_j'\hat{\mathbf{V}}(\hat{\boldsymbol{\beta}})\mathbf{e}_j}\right)$$

where $t_{df,\alpha/2}$ is the $100(1-\alpha/2)$ percentile point of the t distribution with df degrees of freedom. For more information about hazard ratios see the section "Hazard Ratios" on page 10280, and for more information about df see the section "Degrees of Freedom" on page 10270. If you use the DF=NONE option in the MODEL statement, then the procedure uses the $100(1-\alpha/2)$ percentile point of the standard normal distribution.

Residuals

This section describes the computation of residuals (RESMART, RESDEV, RESSCH, and RESSCO in the OUTPUT statement). See the section "Notation and Estimation" on page 10247 for definition of notation that is used in this section. The residuals are calculated based on the TIES= option in the MODEL statement.

TIES=BRESLOW

This is the default option. Let

$$S^{(r)}(\boldsymbol{\beta},t) = \sum_A w_{hij}\, y_{hij}(t) \exp\left(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t)\right) \mathbf{Z}_{hij}^{\otimes r}(t)$$

$$\bar{\mathbf{Z}}(\boldsymbol{\beta},t) = \frac{S^{(1)}(\boldsymbol{\beta},t)}{S^{(0)}(\boldsymbol{\beta},t)}$$

where $r = 0, 1$; and A be the set of indices in the selected sample.

Further let

$$\begin{aligned} d\Lambda_0(\boldsymbol{\beta},t) &= \sum_A \frac{w_{hij}\, dn_{hij}(t)}{S^{(0)}(\boldsymbol{\beta},t)} \\ dM_{hij}(\boldsymbol{\beta},t) &= dn_{hij}(t) - y_{hij}(t)\exp\left(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t)\right) d\Lambda_0(\boldsymbol{\beta},t) \end{aligned}$$

The martingale residual at t is defined as

$$\begin{aligned} \hat{M}_{hij}(t) &= \int_0^t dM_{hij}(\hat{\boldsymbol{\beta}},\tau) \\ &= n_{hij}(t) - \int_0^t y_{hij}(\tau)\exp\left(\hat{\boldsymbol{\beta}}'\mathbf{Z}_{hij}(\tau)\right) d\Lambda_0(\hat{\boldsymbol{\beta}},\tau) \end{aligned}$$

Here $\hat{M}_{hij}(t)$ estimates the difference over $(0,t]$ between the observed number of events for the (h,i,j) observation unit and a conditional expected number of events. The quantity $\hat{M}_{hij} \equiv \hat{M}_{hij}(\infty)$ is referred

to as the martingale residual for the (h, i, j) observation unit. For the Cox model with no time-dependent explanatory variables, the martingale residual for the (h, i, j) unit with observation time $t_{(h,i,j)}$ and event status $\Delta_{(h,i,j)}$ is

$$\hat{M}_{(h,i,j)} = \Delta_{(h,i,j)} - \mathrm{e}^{\hat{\boldsymbol{\beta}}'\mathbf{Z}_{(h,i,j)}} \int_0^{t_{(h,i,j)}} d\Lambda_0(\hat{\boldsymbol{\beta}}, s)$$

The deviance residual D_{hij} for the (h, i, j) observation unit is a transformation of the corresponding martingale residuals,

$$D_{hij} = \mathrm{sign}(\hat{M}_{hij})\sqrt{2\left[-\hat{M}_{hij} - n_{hij}(\infty)\log\left(\frac{n_{hij}(\infty) - \hat{M}_{hij}}{n_{hij}(\infty)}\right)\right]}$$

The square root shrinks large negative martingale residuals, while the logarithmic transformation expands martingale residuals that are close to unity. As such, the deviance residuals are more symmetrically distributed around zero than the martingale residuals. For the Cox model, the deviance residual reduces to the form

$$D_{hij} = \mathrm{sign}(\hat{M}_{hij})\sqrt{2[-\hat{M}_{hij} - \Delta_{hij}\log(\Delta_{hij} - \hat{M}_{hij})]}$$

The Schoenfeld (1982) residual vector is calculated on a per-event-time basis. At the kth event time $t_{hij,k}$ of the (h, i, j) observation unit, the Schoenfeld residual

$$\hat{\mathbf{U}}_{hij}(t_{hij,k}) = \mathbf{Z}_{hij}(t_{hij,k}) - \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, t_{hij,k})$$

is the difference between the observed covariate vector for the (h, i, j) observation unit and the average of the covariate vectors over the risk set at $t_{hij,k}$. Under the proportional hazards assumption, the Schoenfeld residuals have the sample path of a random walk; therefore, they are useful in assessing time trend or lack of proportionality.

The score process for the (h, i, j) subject at time t is

$$\mathbf{L}_{hij}(\boldsymbol{\beta}, t) = \int_0^t [\mathbf{Z}_{hij}(\tau) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, \tau)]dM_{hij}(\boldsymbol{\beta}, \tau)$$

The vector $\hat{\mathbf{L}}_{hij} \equiv \mathbf{L}_{hij}(\hat{\boldsymbol{\beta}}, \infty)$ is the score residual for the (h, i, j) observation unit.

The score residuals are a decomposition of the first partial derivative of the log likelihood. They are useful in assessing the influence of each subject on individual parameter estimates. They also play an important role in the computation of the variance estimators.

TIES=EFRON

For TIES=EFRON, the preceding computation is modified to comply with the Efron partial likelihood. For a given uncensored time t, let $\delta_{hij}(t) = 1$ if t is an event time for the (h, i, j) observation, and 0 otherwise. Let $d(t) = \sum_{hij \in A} \delta_{hij}(t)$, which is the number of observation units that have an event at t. For $1 \le l \le d(t)$,

let

$$
\begin{aligned}
S^{(r)}(\boldsymbol{\beta}, l, t) &= \sum_A w_{hij}\, y_{hij}(t)\left\{1 - \frac{l-1}{d(t)}\delta_{hij}(t)\right\} \exp\left(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t)\right) \mathbf{Z}_{hij}^{\otimes r}(t) \\
\bar{\mathbf{Z}}(\boldsymbol{\beta}, l, t) &= \frac{S^{(1)}(\boldsymbol{\beta}, l, t)}{S^{(0)}(\boldsymbol{\beta}, l, t)} \\
d\Lambda_0(\boldsymbol{\beta}, l, t) &= \sum_A \frac{w_{hij}\, dn_{hij}(t)}{S^{(0)}(\boldsymbol{\beta}, l, t)} \\
dM_{hij}(\boldsymbol{\beta}, l, t) &= dn_{hij}(t) - y_{hij}(t)\left(1 - \delta_{hij}(t)\frac{l-1}{d(t)}\right) \exp\left(\boldsymbol{\beta}'\mathbf{Z}_{hij}(t)\right) d\Lambda_0(\boldsymbol{\beta}, l, t)
\end{aligned}
$$

where $r = 0, 1$, and A are the set of indices in the selected sample.

The martingale residual at t for the (h, i, j) observation unit is defined as

$$
\begin{aligned}
\hat{M}_{hij}(t) &= \int_0^t \frac{1}{d(\tau)} \sum_{l=1}^{d(\tau)} dM_{hij}(\hat{\boldsymbol{\beta}}, l, \tau) \\
&= n_{hij}(t) - \int_0^t \frac{1}{d(\tau)} \sum_{l=1}^{d(\tau)} y_{hij}(\tau)\left(1 - \delta_{hij}(\tau)\frac{l-1}{d(\tau)}\right) \exp\left(\hat{\boldsymbol{\beta}}'\mathbf{Z}_{hij}(\tau)\right) d\Lambda_0(\hat{\boldsymbol{\beta}}, l, \tau)
\end{aligned}
$$

Deviance residuals are computed by using the same transform on the corresponding martingale residuals as in TIES=BRESLOW.

The Schoenfeld residual vector for the (h, i, j) observation unit at event time $t_{hij,k}$ is

$$
\hat{\mathbf{U}}_{hij}(t_{hij,k}) = \mathbf{Z}_{hij}(t_{hij,k}) - \frac{1}{d(t_{hij,k})} \sum_{l=1}^{d(t_{hij,k})} \bar{\mathbf{Z}}(\hat{\boldsymbol{\beta}}, l, t_{hij,k})
$$

The score process for the (h, i, j) observation unit at time t is

$$
\mathbf{L}_{hij}(\boldsymbol{\beta}, t) = \int_0^t \frac{1}{d(\tau)} \sum_{l=1}^{d(\tau)} \left(\mathbf{Z}_{hij}(\tau) - \bar{\mathbf{Z}}(\boldsymbol{\beta}, l, \tau)\right) dM_{hij}(\boldsymbol{\beta}, l, \tau)
$$

The vector $\hat{\mathbf{L}}_{hij} \equiv \mathbf{L}_{hij}(\hat{\boldsymbol{\beta}}, \infty)$ is the score residual for the (h, i, j) observation unit.

Hazard Ratios

Consider a dichotomous risk factor variable X that takes the value 1 if the risk factor is present and 0 if the risk factor is absent. The log-hazard function is given by

$$\log[\lambda(t|X)] = \log[\lambda_0(t)] + \beta_1 X$$

where $\lambda_0(t)$ is the baseline hazard function.

The hazard ratio ψ is defined as the ratio of the hazard for those with the risk factor (X = 1) to the hazard for those without the risk factor (X = 0). The log of the hazard ratio is given by

$$\log(\psi) \equiv \log[\psi(X = 1, X = 0)] = \log[\lambda(t|X = 1)] - \log[\lambda(t|X = 0)] = \beta_1$$

In general, the hazard ratio can be computed by exponentiating the difference of the log-hazard between any two population profiles. This is the approach taken by the HAZARDRATIO statement, so the computations are available regardless of parameterization, interactions, and nestings. However, as shown in the preceding equation for $\log(\psi)$, hazard ratios of main effects can be computed as functions of the parameter estimates. The remainder of this section is concerned with this methodology.

The parameter β_1, which is associated with X, represents the change in the log-hazard rate from X=0 to X=1. So the hazard ratio is obtained by simply exponentiating the value of the parameter that is associated with the risk factor. The hazard ratio indicates how the hazard changes as you change X from 0 to 1. For example, $\psi = 2$ means that the hazard when X=1 is twice the hazard when X=0.

Suppose the values of the dichotomous risk factor are coded as constants a and b instead of 0 and 1. The hazard when X=a becomes $\lambda(t)\exp(a\beta_1)$, and the hazard when X=b becomes $\lambda(t)\exp(b\beta_1)$. The hazard ratio that corresponds to an increase in X from a to b is

$$\psi = \exp[(b-a)\beta_1] = [\exp(\beta_1)]^{b-a} \equiv [\exp(\beta_1)]^c$$

Note that for any a and b such that $c = b - a = 1$, $\psi = \exp(\beta_1)$. So the hazard ratio can be interpreted as the change in the hazard for any increase of one unit in the corresponding risk factor. However, the change in hazard for some amount other than one unit is often of greater interest. For example, a change of one pound in body weight might be too small to be considered important, whereas a change of 10 pounds might be more meaningful. The hazard ratio for a change in X from a to b is estimated by raising the hazard ratio estimate for a unit change in X to the power of $c = b - a$ as shown previously.

For a polytomous risk factor, the computation of hazard ratios depends on how the risk factor is parameterized. For illustration, suppose that Cell is a risk factor that has four categories: Adeno, Large, Small, and Squamous.

For the effect parameterization scheme (PARAM=EFFECT) with Squamous as the reference group, the design variables for Cell are as follows:

	Design Variables for Cell		
Cell	X_1	X_2	X_3
Adeno	1	0	0
Large	0	1	0
Small	0	0	1
Squamous	–1	–1	–1

The log-hazard for Adeno is

$$\begin{aligned}\log[\lambda(t|\text{Adeno})] &= \log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) \\ &= \lambda_0(t) + \beta_1\end{aligned}$$

The log-hazard for Squamous is

$$\begin{aligned}\log[\lambda(t|\text{Squamous})] &= \log[\lambda_0(t)] + \beta_1(X_1 = -1) + \beta_2(X_2 = -1) + \beta_3(X_3 = -1)) \\ &= \log[\lambda_0(t)] - \beta_1 - \beta_2 - \beta_3\end{aligned}$$

Therefore, the log-hazard ratio of Adeno versus Squamous

$$\begin{aligned}\log[\psi(\text{Adeno, Squamous})] &= \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})] \\ &= 2\beta_1 + \beta_2 + \beta_3\end{aligned}$$

For the reference cell parameterization scheme (PARAM=REF) with Squamous as the reference cell, the design variables for Cell are as follows:

	Design Variables for Cell		
Cell	X_1	X_2	X_3
Adeno	1	0	0
Large	0	1	0
Small	0	0	1
Squamous	0	0	0

The log-hazard ratio of Adeno versus Squamous is given by

$$\begin{aligned}&\log(\psi(\text{Adeno, Squamous})) \\ &= \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})] \\ &= (\log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0)) - \\ &\quad (\log[\lambda_0(t)] + \beta_1(X_1 = 0) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0)) \\ &= \beta_1\end{aligned}$$

For the GLM parameterization scheme (PARAM=GLM), the design variables for Cell are as follows:

	Design Variables for Cell			
Cell	X_1	X_2	X_3	X_4
Adeno	1	0	0	0
Large	0	1	0	0
Small	0	0	1	0
Squamous	0	0	0	1

The log-hazard ratio of Adeno versus Squamous is

$$
\begin{aligned}
&\log(\psi(\text{Adeno}, \text{Squamous})) \\
&\quad = \log[\lambda(t|\text{Adeno})] - \log[\lambda(t|\text{Squamous})] \\
&\quad = \log[\lambda_0(t)] + \beta_1(X_1 = 1) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) + \beta_4(X_4 = 0)) - \\
&\qquad (\log(\lambda_0(t)) + \beta_1(X_1 = 0) + \beta_2(X_2 = 0) + \beta_3(X_3 = 0) + \beta_4(X_4 = 1)) \\
&\quad = \beta_1 - \beta_4
\end{aligned}
$$

The fact that the log-hazard ratio ($\log(\psi) = \mathbf{l}^T\boldsymbol{\beta}$) is a linear function of the parameters enables the HAZARDRATIO statement to estimate the hazard ratio of the main effect even in the presence of interactions and nest effects. The log-hazard ratio is estimated by $\mathbf{l}^T\hat{\boldsymbol{\beta}}$, and the variance of the estimated log-hazard ratio is computed by $\mathbf{l}^T\hat{V}(\hat{\boldsymbol{\beta}})\mathbf{l}$, where $\hat{V}(\hat{\boldsymbol{\beta}})$ is the estimated variance for $\hat{\boldsymbol{\beta}}$.

To customize hazard ratios for specific units of change for a continuous risk factor, you can use the UNITS= option in a HAZARDRATIO statement to specify a list of relevant units for each explanatory variable in the model. Estimates of these customized hazard ratios are given in a separate table. Let (L_j, U_j) be a confidence interval for $\log(\psi)$. The corresponding lower and upper confidence limits for the customized hazard ratio $\exp(c\beta_j)$ are $\exp(cL_j)$ and $\exp(cU_j)$, respectively, (for $c > 0$), or $\exp(cU_j)$ and $\exp(cL_j)$, respectively, (for $c < 0$).

Output Data Sets

You can use the Output Delivery System to create a SAS data set from any piece of PROC SURVEYPHREG output. See the section "ODS Table Names" on page 10287 for more information. PROC SURVEYPHREG also provides an output data set to store observation-level statistics, an output data set to store the replicate weights for BRR or jackknife variance estimation, and an output data set to store the jackknife coefficients for jackknife variance estimation.

OUT= Data Set for the OUTPUT Statement

The OUTPUT statement can be used to store observation-level statistics, such as the predicted values and their standard errors, the (weighted) number of observation units at risk, martingale residuals, Schoenfeld residuals, score residuals, and deviance residuals. See the section "Residuals" on page 10277 for details about how these statistics are calculated.

Replicate Weights Output Data Set

If you specify the OUTWEIGHTS= *method-option* for VARMETHOD=BOOTSTRAP, BRR, or JACKKNIFE, PROC SURVEYPHREG stores the replicate weights in an output data set. The OUTWEIGHTS= output data set contains all observations that are used in the analysis or all valid observations in the DATA= input data set. See the section "Missing Values" on page 10259 for details about valid observations.

The OUTWEIGHTS= data set contains the following variables:

- all variables in the DATA= input data set

- RepWt_1, RepWt_2, . . ., RepWt_R, which are the replicate weight variables, where R is the total number of replicates in the analysis

Each replicate weight variable contains the replicate weights for the corresponding replicate. Replicate weights equal zero for those observations not included in the replicate.

After the procedure creates replicate weights for a particular input data set and survey design, you can use the OUTWEIGHTS= *method-option* to store these replicate weights and then use them again in subsequent analyses, either in PROC SURVEYPHREG or in the other survey procedures. You can use the REPWEIGHTS statement to provide replicate weights for the procedure.

Jackknife Coefficients Output Data Set

If you specify the OUTJKCOEFS= *method-option* for VARMETHOD=JACKKNIFE, PROC SURVEYPHREG stores the jackknife coefficients in an output data set. The OUTJKCOEFS= output data set contains one observation for each replicate. The OUTJKCOEFS= data set contains the following variables:

- Replicate, which is the replicate number for the jackknife coefficient
- JKCoefficient, which is the jackknife coefficient for the replicate
- DonorStratum, which is the stratum of the PSU that was deleted to construct the replicate, if you specify a STRATA statement

After the procedure creates jackknife coefficients for a particular input data set and survey design, you can use the OUTJKCOEFS= *method-option* to store these coefficients and then use them again in subsequent analyses, either in PROC SURVEYPHREG or in the other survey procedures. You can use the JKCOEFS= option in the REPWEIGHTS statement to provide jackknife coefficients for the procedure.

Displayed Output

If you use the NOPRINT option in the PROC SURVEYPHREG statement, the procedure does not display any output. Otherwise, PROC SURVEYPHREG displays results of the analysis in a collection of tables.

Model Information

The “Model Information” table displays the two-level name of the input data set, the name and label of the failure time variable, the name and label of the censoring variable and the values that indicate censored times, the model, the name and label of the FREQ variable, the name and label of the WEIGHT variable, the name and label of the STRATA variables, the name and label of the CLUSTER variables, and the method of handling ties in the failure time for the Cox model. The ODS name of the “Model Information” table is ModelInfo.

Number of Observations

The "Number of Observations" table displays the number of observations that are read and used, the sum of frequencies read and used, the sum of weights read and used, and the weighted sum of frequencies that are read and used in the analysis. The ODS name of the "Number of Observations" table is NObs.

Summary of the Number of Event and Censored Values

The "Summary of the Number of Event and Censored Values" table displays the number of events and censored values. The ODS name of the "Summary of the Number of Event and Censored Values" table is CensoredSummary.

Summary of the Weighted Number of Event and Censored Values

The "Summary of the Weighted Number of Event and Censored Values" table displays the weighted number of events and censored values. The ODS name of the "Summary of the Weighted Number of Event and Censored Values" table is WeightedCensoredSummary.

Class Level Information

The "Class Level Information" table is displayed when there are CLASS variables in the model. The table lists the categories of every CLASS variable that is used in the model and the corresponding design variable values. The ODS name of the "Class Level Information" table is ClassLevelInfo.

Design Summary Table

The "Design Summary" table provides information about the sample design. The table displays the total number of strata that are read and used, and the total number of clusters read and used. The table is displayed only if you specify a STRATA or CLUSTER statement. The ODS name of the "Design Summary" table is DesignSummary.

Stratum Information Table

If you specify the LIST option in the STRATA statement, PROC SURVEYPHREG displays a "Stratum Information" table. The ODS name of the "Stratum Information Table" is StrataInfo. This table provides the following information for each stratum:

- Stratum Index, which is a sequential stratum identification number
- STRATA variables, which list the levels of STRATA variables for the stratum
- Number of Observations, which is the number of observations used in the stratum
- Population Total for the stratum, if you specify the TOTAL= option
- Sampling Rate for the stratum, if you specify the TOTAL= or RATE= option. If you specify the TOTAL= option, the sampling rate is based on the number of valid observations in the stratum.
- Number of Clusters, which is the number of clusters in the stratum, if you specify a CLUSTER statement

Convergence Status

The "Convergence Status" table displays the convergence status of the optimization routine. The procedure displays this table only when you specify the NLOPTIONS statement. The ODS name of the "Convergence Status" table is ConvergenceStatus.

Model Fit Statistics

The "Model Fit Statistics" table displays the values of –2 log likelihood and the AIC for the null model and the fitted model. The ODS name of the "Model Fit Statistics" table is FitStatistics.

Testing Global Null Hypothesis: BETA=0

The "Testing Global Null Hypothesis: BETA=0" table displays results of the likelihood ratio test and the Wald test for testing the hypothesis that all parameters are zero. The ODS name of the "Testing Global Null Hypothesis: BETA=0" table is GlobalTests.

Analysis of Maximum Likelihood Estimates

The "Analysis of Maximum Likelihood Estimates" table displays the denominator degrees of freedom, which is computed as described in the section "Degrees of Freedom" on page 10270; the maximum likelihood estimate of the parameter; the estimated standard error, computed as the square root of the corresponding diagonal element of the estimated covariance matrix; the t statistic, computed as the parameter estimate divided by the standard error; the p-value of the t statistic with respect to a t distribution with denominator degrees of freedom; and the hazard ratio estimate. The t confidence limits for the parameter estimates and estimated hazard ratios are displayed if you specify the CLPARM or RISKLIMITS option in the MODEL statement. You can specify the DF=NONE option in the MODEL statement to request p-values and confidence intervals from a standard normal distribution. If you specify the VARRATIO=ALL | MODEL | IND option in the MODEL statement, then the variance ratios for model or independence (or both) are displayed. If you specify the SERATIO=ALL | MODEL | IND option in the MODEL statement, then the standard error ratios for model or independence (or both) are displayed.

The ODS name of the "Analysis of Maximum Likelihood Estimates" table is ParameterEstimates.

Covariance Matrix

The "Covariance Matrix" table is displayed if you specify the COVB option in the MODEL statement. The table contains the estimated covariance matrix for the parameter estimates. The ODS name of the "Covariance Matrix" table is CovB.

Hessian Matrix

The "Hessian Matrix" table is displayed if you specify the HESS option in the MODEL statement. The table contains the Hessian matrix that is evaluated at the estimated regression parameters. The ODS name of the "Hessian Matrix" table is Hessian.

Inverse Hessian Matrix

The "Inverse Hessian Matrix" table is displayed if you specify the INVHESS option in the MODEL statement. The table contains the inverse of the Hessian matrix evaluated at the estimated regression parameters. The ODS name of the "Inverse Hessian Matrix" table is InvHessian.

Risk Set Sum of Weights

The "Risk Set Sum of Weights" table is displayed if you specify the ATRISK option in the PROC SURVEYPHREG statement. The table displays, for each event time, the sum of weights of units at risk and the sum of weights of units that experience the event. The ODS name of the "Risk Set Sum of Weights" table is RiskSetInfo.

Variance Estimation Table

The "Variance Estimation" table provides the following information:

- Method, which is the variance estimation method—Taylor Series, Balanced Repeated Replication, or Jackknife
- Replicate Weights input data set name, if you provide replicate weights with a REPWEIGHTS statement
- Number of Replicates, for VARMETHOD=BRR VARMETHOD=BOOTSTRAP, or VARMETHOD=JACKKNIFE
- Hadamard Data Set name, if you specify the HADAMARD= *method-option* for VARMETHOD=BRR
- Fay Coefficient, if you specify the FAY *method-option* for VARMETHOD=BRR
- Missing Values Included (MISSING), if you specify the MISSING option
- Missing Values Included (NOMCAR), if you specify the NOMCAR option
- Missing Values Excluded, if you have missing values and you do not specify the NOMCAR option
- Bootstrap Seed, for VARMETHOD=BOOTSTRAP

The ODS name of the "Variance Estimation" table is VarianceEstimation.

Hadamard Matrix

If you specify the PRINTH *method-option* for VARMETHOD=BRR, PROC SURVEYPHREG displays the Hadamard matrix that is used to construct replicates for BRR variance estimation. If you provide a Hadamard matrix with the HADAMARD= *method-option* for VARMETHOD=BRR but the procedure does not use the entire matrix, the procedure displays only the rows and columns that are actually used to construct replicates. The ODS name of the "Hadamard Matrix" table is HadamardMatrix.

Maximum Likelihood Estimates for Replicate Samples

If you specify the DETAILS *method-option* for VARMETHOD=BRR, VARMETHOD=BOOTSTRAP, or VARMETHOD=JACKKNIFE, PROC SURVEYPHREG displays the "Maximum Likelihood Estimates for Replicate Samples" table. The Replicate Number column displays the replication number, the Replicate Weight column displays the name of the replicate weight variable, and the Status column displays the convergence status. The replicate number for the full sample is set to 0. If you do not specify replicate weights, then PROC SURVEYPHRG uses default names to identify the replicate weights. For more information, see the section "Replicate Weights Output Data Set" on page 10282.

Define AXTOL as the maximum absolute difference of the regression parameters between the last two iterations,

$$\text{AXTOL} = \max_j |\beta_j^{(t)} - \beta_j^{(t-1)}|$$

where $\beta_j^{(t)}$ is the jth component of $\boldsymbol{\beta}^{(t)}$ at iteration t.

The convergence status for a replicate sample is 1 if at least one convergence criterion for maximum likelihood estimation is satisfied, 2 if at least one convergence criterion for maximum likelihood estimation is satisfied but AXTOL is greater than 0.1, and 0 if no maximum likelihood convergence criteria are satisfied and thus the maximum likelihood estimates are not available in the replicate sample. If the maximum likelihood estimates are not available for a replicate sample, then the parameter estimates are set to missing for that replicate.

The ODS name of the "Maximum Likelihood Estimates for Replicate Samples" table is RepEstimates.

Hazard Ratios for *label*

The "Hazard Ratios for *label*" table is displayed if you specify the HAZARDRATIO statement. The table displays the estimate and confidence limits for each hazard ratio. The ODS name of the "Hazard Ratios for *label*" table is HazardRatios.

ODS Table Names

PROC SURVEYPHREG assigns a name to each table it creates. You can use this name to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System." Table 122.11 lists the table names, along with the corresponding analysis options.

Table 122.11 ODS Tables Produced by PROC SURVEYPHREG

ODS Table Name	Description	Statement / Option
CensoredSummary	Summary of event and censored observations	Default
ClassLevelInfo	CLASS variable levels	CLASS
ConvergenceStatus	Convergence status	NLOPTIONS / PALL
CovB	Covariance of parameter estimates	MODEL / COVB
DesignSummary	Design summary	STRATA or CLUSTER
FitStatistics	Model fit statistics	Default

Table 122.11 *continued*

ODS Table Name	Description	Statement / Option
GlobalTests	Tests of the global null hypothesis	Default
HadamardMatrix	Hadamard matrix	PROC / VARMETHOD=BRR(PRINTH)
HazardRatios	Hazard ratios and confidence limits	HAZARDRATIO
Hessian	Observed Hessian matrix	MODEL / HESSIAN
InvHessian	Inverse Hessian matrix	MODEL / INVHESS
IterHist	Iteration history	NLOPTIONS / PHISTORY or PHISTPARMS
ModelInfo	Model information	Default
NObs	Number of observations	Default
ParameterEstimates	Maximum likelihood estimates	Default
ParameterEstimatesStart	Initial parameter values	NLOPTIONS / PALL
RepEstimates	Maximum likelihood estimates for replicate samples	PROC / VARMETHOD=BRR(DETAILS), VARMETHOD=JACKKNIFE(DETAILS) or VARMETHOD=BOOTSTRAP(DETAILS)
RiskSetInfo	Risk set information	PROC / ATRISK
StrataInfo	Stratum information	STRATA / LIST
VarianceEstimation	Variance estimation	Default
WeightedCensoredSummary	Summary of weighted number of event and censored observations	WEIGHT

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, the ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements can produce plots that are associated with their analyses. For information about these plots, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Examples: SURVEYPHREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 122.1: Analysis of Clustered Data

(View the complete code for this example (surphex1.sas) in the example repository.)

When experimental units are naturally or artificially clustered, failure times of experimental units within a cluster are correlated. Lee, Wei, and Amato (1992) estimate the regression parameters in the Cox model by maximizing a partial likelihood function under an independent working correlation assumption and estimate the variance of the estimated regression coefficients by using a robust sandwich variance estimator that accounts for the intracluster dependence.

The Diabetic Retinopathy Study (DRS) is a randomized, controlled clinical trial of more than 1,700 patients across 15 medical centers. One objective of this study was to determine if photocoagulation treatment delays the occurrence of blindness. One eye of each patient was randomly assigned to treatment and the other eye to control. For more information about the data set and other alternative analyses, see Example 92.11 in Chapter 92, "The PHREG Procedure."

Each patient is a cluster that contributes two observations to the input data set, one for each eye. The following variables are available:

- ID, patient's identification
- Time, failure time
- Status, event indicator (0=censored, and 1=uncensored)
- Treatment, treatment received (1=laser photocoagulation, and 0=otherwise)
- DiabeticType, type of diabetes (0=juvenile onset with age of onset at 20 or under, and 1= adult onset with age of onset over 20)

The following DATA step creates the data set Blind, which represents 197 diabetic patients from the DRS:

```
proc format;
   value type 0='Juvenile' 1='Adult';
   value Rx  1='Laser' 0='Others';
run;

data Blind;
   format Treatment Rx.
          DiabeticType Type.;
   input ID Time Status DiabeticType Treatment @@;
   datalines;
   5 46.23 0 1 1    5 46.23 0 1 0   14 42.50 0 0 1   14 31.30 1 0 0
  16 42.27 0 0 1   16 42.27 0 0 0   25 20.60 0 0 1   25 20.60 0 0 0
```

```
  29 38.77 0 0 1   29  0.30 1 0 0   46 65.23 0 0 1   46 54.27 1 0 0
  49 63.50 0 0 1   49 10.80 1 0 0   56 23.17 0 0 1   56 23.17 0 0 0
  61  1.47 0 0 1   61  1.47 0 0 0   71 58.07 0 1 1   71 13.83 1 1 0
 100 46.43 1 1 1  100 48.53 0 1 0  112 44.40 0 1 1  112  7.90 1 1 0
 120 39.57 0 1 1  120 39.57 0 1 0  127 30.83 1 1 1  127 38.57 1 1 0
 133 66.27 0 1 1  133 14.10 1 1 0  150 20.17 1 0 1  150  6.90 1 0 0

   ... more lines ...

1705  8.00 0 0 1 1705  8.00 0 0 0 1717 51.60 0 1 1 1717 42.33 1 1 0
1727 49.97 0 1 1 1727  2.90 1 1 0 1746 45.90 0 0 1 1746  1.43 1 0 0
1749 41.93 0 1 1 1749 41.93 0 1 0
;
```

The following statements request a proportional hazards regression of Time on Treatment, DiabeticType, and the Treatment × DiabeticType interaction, with Status as the censoring indicator. The CLUSTER statement indicates the observations that came from the same patient. The HAZARDRATIO statement requests estimated hazard ratios for the treatments.

```
proc surveyphreg data=Blind;
   class Treatment DiabeticType / param=ref;
   model Time*Status(0) = Treatment DiabeticType Treatment*DiabeticType;
   cluster ID;
   hazardratio Treatment;
run;
```

Output 122.1.1 displays some summary information. There are 394 observations and 197 patients (clusters). Almost 61% of the observations are censored. The p-values for the null model are less than 0.0001 for both the adjusted likelihood ratio test and the Wald test (Output 122.1.2), indicating that the survival time is highly dependent on Treatment and DiabeticType. In this example, the adjusted likelihood ratio statistic has an approximate chi-square distribution with 2.7 degrees of freedom, and the Wald statistic has an approximate F distribution with 3 numerator degrees of freedom and 194 denominator degrees of freedom. The denominator degrees of freedom is calculated as the number of clusters (197) minus the number of estimable parameters (3). Although the adjusted and the unadjusted likelihood ratio tests are very similar for this data set, the unadjusted likelihood ratio test does not account for clustering. It is recommended that you use the adjusted likelihood ratio test for clustered data.

Output 122.1.1 Summary Information

The SURVEYPHREG Procedure

Number of Observations Read	394
Number of Observations Used	394

Design Summary	
Number of Clusters	197

Summary of the Number of Event and Censored Values			
Total	**Event**	**Censored**	**Percent Censored**
394	155	239	60.66

Output 122.1.1 *continued*

Variance Estimation	
Method	Taylor Series

Output 122.1.2 Global Test Results

Testing Global Null Hypothesis: BETA=0				
Test	**Test Statistic**	**Num DF**	**Den DF**	**p-Value**
Likelihood Ratio (Unadj.)	28.4556	3	Infty	<.0001
Likelihood Ratio (Adj.)	28.1668	2.703	Infty	<.0001
Wald	11.4455	3	194	<.0001

Output 122.1.3 displays parameter estimates, standard errors, t statistics, denominator degrees of freedom, p-values, and hazard ratios. In this example data set, Treatment and the Treatment × DiabeticType interaction are significant with p-values 0.023 and 0.006, respectively. Because the model contains the Treatment × DiabeticType interaction, the exponential of the estimated regression coefficient is not the hazard ratio. The HAZARDRATIO statement requests the hazard ratios.

Output 122.1.3 Parameter Estimates

Analysis of Maximum Likelihood Estimates						
Parameter	**DF**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**	**Hazard Ratio**
Treatment Laser	196	-0.424672	0.185438	-2.29	0.0231	0.654
DiabeticType Adult	196	0.340841	0.196076	1.74	0.0837	1.406
Treatment*DiabeticTy Laser Adult	196	-0.845665	0.304303	-2.78	0.0060	0.429

Output 122.1.4 displays the estimated hazard ratios and corresponding 95% Wald confidence intervals. For both types of diabetes, the 95% confidence interval for the hazard ratio lies below 1, indicating the effectiveness of the treatment for both types of diabetes.

Output 122.1.4 Hazard Ratios

Hazard Ratios for Treatment			
Description	**Point Estimate**	**95% Wald Confidence Limits**	
Treatment Laser vs Others At DiabeticType=Adult	0.281	0.174	0.453
Treatment Laser vs Others At DiabeticType=Juvenile	0.654	0.454	0.943

Example 122.2: Stratification, Clustering, and Unequal Weights

(View the complete code for this example (surphex2.sas) in the example repository.)

This example uses a data set from the National Health and Nutrition Examination Survey I (NHANES I) Epidemiologic Followup Study (NHEFS). The NHEFS is a national longitudinal survey that is conducted by the National Center for Health Statistics, the National Institute on Aging, and some other agencies of the Public Health Service in the United States. Some important objectives of this survey are to determine the relationships between clinical, nutritional, and behavioral factors; to determine mortality and hospital utilizations; and to monitor changes in risk factors for the initial cohort that represents the NHANES I population. A cohort of size 14,407, which includes all persons 25 to 74 years old who completed a medical examination at NHANES I in 1971–1975, was selected for the NHEFS. Personal interviews were conducted for every selected unit during the first wave of data collection from the year 1982 to 1984. Follow-up studies were conducted in 1986, 1987, and 1992. In the year 1986, only nondeceased persons 55 to 74 years old (as reported in the base year survey) were interviewed. The 1987 and 1992 NHEFS contain the entire nondeceased NHEFS cohort. Vital and tracing status data, interview data, health care facility stay data, and mortality data for all four waves are available for public use. For more information about the survey and the data sets, see the Center for Disease Control and Prevention's website (`https://www.cdc.gov/`).

For illustration purposes, 1,018 observations from the 1987 NHEFS public use interview data are used to create the data set cancer. The observations are obtained from 10 strata that contain 596 PSUs. The sum of observation weights for these selected units is over 19 million. Observation weights range from 359 to 129,359 with a mean of 18,747.69 and a median of 11,414. Several observation weights have large values; therefore it is reasonable to rescale the observation weights to facilitate the optimization routine. Different scaling techniques are proposed in the literature. For example, Binder (1992) uses scaled weights such that the sum of weights over the sampled units is one. Without loss of generality, the analysis weights in this example are obtained by dividing each observation weight by a large number (130,000). Because of this rescaling, you must be careful interpreting some results from PROC SURVEYPHREG.

The following variables are used in this example:

- ObsNo, unit identification
- Strata, stratum identification
- PSU, identification for primary sampling units
- ObservationWt, sampling weight associated with each unit
- AnalysisWt, obtained from the sampling weights by dividing each ObservationWt by 130,000
- Smoke, smoking status (–1 = not applicable, 1 = never smoked, 2 = current or former smoker in 1982–1984 follow-up, and 3 = current or former smoker in 1987 follow-up)
- Age, the event-time variable, defined as follows:
  - age of the subject when the first cancer was reported for subjects with reported cancer
  - age of the subject at death for deceased subjects without reported cancer
  - age of the subject as reported in 1987 follow-up (this value is used for nondeceased subjects who never reported cancer)

  - age of the subject for the entry year 1971–1975 survey if the subject has cancer (or is deceased) but the date of incident is not reported

- Cancer, cancer indicator (1 = cancer reported, 0 = cancer not reported)

- BodyWeight, body weight of the subject as reported in the 1987 follow-up, or an imputed body weight based on the subject's age in the entry year 1971–1975 survey

The following SAS statements create the data set cancer. Note that BodyWeight for a few observations (8%) is imputed based on Age by using a deterministic regression imputation model (Särndal and Lundström (2005, chapter 12)). The imputed values are treated as observed values in this example. In other words, this example treats the data set cancer as the observed data set.

```
data cancer;
   input ObsNo Strata PSU AnalysisWt ObservationWt Smoke
         Age Cancer BodyWeight;
   datalines;
   1  3  002  0.02927    3805   2  53  1  175
   2  3  002  0.04698    6107   2  77  0  175
   3  3  039  0.02283    2968   2  50  0  160
   4  3  084  0.23414    30438  2  52  0  145
   5  3  007  0.03908    5081   1  80  0  127
   6  3  009  0.02993    3891   1  62  0  180
   7  3  009  0.02754    3580   2  50  0  157
   8  3  022  0.02283    2968   2  56  0  142
   9  3  050  0.18268    23748  2  60  0  140

   ... more lines ...

1016  4  002   0.02068   2689   2  40  0  120
1017  4  092   0.35298  45888   2  52  0  166
1018  4  035   0.03344   4347  -1  58  0  156
;
```

Suppose you want to study the occurrence of cancer for the base year survey population and its relation to smoking status and body weight. The following statements request a proportional hazards regression of Age on BodyWeight and Smoke with Cancer as the censor indicator. The STRATA, CLUSTER, and WEIGHT statements identify the variance strata, PSUs, and analysis weights respectively. The CLASS statement specifies that Smoke is a categorical variable, and the MODEL statement provides information about the analysis model. The TIES= option in the MODEL statement requests the Efron likelihood to handle tied events. If you do not specify the TIES= option in the MODEL statement, then the procedure uses the Breslow likelihood. The PHISTORY option in the NLOPTIONS statement is used to display the iteration history table. The ESTIMATE statement computes a contrast between subjects who are reported as current (or former) smokers and the others. The EXP option in the ESTIMATE statement requests that the linear contrast be estimated in the exponential scale, which is the hazard ratio. The TEST statement requests the Type 3 test for each effect that is specified in the MODEL statement.

```
proc surveyphreg data = cancer;
   strata strata;
   cluster psu;
   weight analysiswt;
   class smoke;
```

```
    model age*cancer(0) = bodyweight smoke / ties = efron;
    nloptions phistory;
    estimate smoke 0.5 0.5 -0.5 -0.5 / exp;
    test ;
run;
```

Some summary statistics are shown in Output 122.2.1. The "Model Information" table contains information about the model such as the names for the dependent and censoring variables, and the likelihood. The "Number of Observations" table displays the number of observations and the sum of weights. A total of 1,018 observations are read from the cancer data set, but one observation is not used in the analysis because it has a zero sampling weight. The sum of weights is 146.81, which gives an estimated 19,085,105 ($= 146.8085 \times 130{,}000$) observation units in the population. Note that the estimated number of observation units in the population would be 19,085,151 if you use the sampling weights (ObservationWt) instead of the analysis weights (AnalysisWt). The difference is due to the rounding errors in AnalysisWt. For simplicity, analysis weights are rounded at the fifth decimal place. The "Design Summary" table shows that there are 596 PSUs and 10 strata. From the censored summary tables, 11.7% subjects in the sample have reported cancer and an estimated 11.6% subjects in the study population have cancer. The "Variance Estimation" table shows that the Taylor series linearization variance estimation method is used and the observation units with missing values are excluded from the analysis. Note that the only missing unit in this data set has a zero sampling weight and hence it is not included in the analysis.

Output 122.2.1 Model Information, Data Summary, Design Summary, and Information about Variance Estimation

The SURVEYPHREG Procedure

Model Information	
Data Set	WORK.CANCER
Dependent Variable	Age
Censoring Variable	Cancer
Censoring Value(s)	0
Weight Variable	AnalysisWt
Stratum Variable	Strata
Cluster Variable	PSU
Ties Handling	EFRON

Number of Observations Read	1018
Number of Observations Used	1017
Sum of Weights Read	146.8085
Sum of Weights Used	146.8085

Design Summary	
Number of Strata	10
Number of Clusters	596

Summary of the Number of Event and Censored Values			
Total	**Event**	**Censored**	**Percent Censored**
1017	119	898	88.30

Output 122.2.1 *continued*

Summary of the Weighted Number of Event and Censored Values			
Total	Event	Censored	Percent Censored
146.8085	17.01185	129.7966	88.41

Variance Estimation	
Method	Taylor Series
Missing Values	Excluded

The "Iteration History" table in Output 122.2.2 shows that the procedure converged after four iterations. The "Objective Function" column contains the value of the likelihood after every iteration. The "Objective Function Change" column measures the change in the objective function between iterations; however, this is not the monitored convergence criterion. The SURVEYPHREG procedure monitors several features simultaneously to determine whether to stop an optimization.

Output 122.2.2 Iteration History

Maximum Likelihood Iteration History								
Iteration	Restarts	Function Calls	Active Constraints	Objective Function	Objective Function Change	Max Abs Gradient Element	Ridge	Ratio Between Actual and Predicted Change
1	0	4	0	-63.34004	1.6501	21.9620	0	0.916
2	0	6	0	-63.29819	0.0418	0.2005	0	1.052
3	0	8	0	-63.29776	0.000430	0.00293	0	1.012
4	0	10	0	-63.29776	1.528E-7	1.102E-6	0	1.000

Estimates for proportional hazards regression coefficients and their standard errors are shown in Output 122.2.3. The categorical variable Smoke has four levels, and GLM parameterization is used by PROC SURVEYPHREG. You can use the PARAM= option in the CLASS statement to specify other types of parameterizations. The estimated regression coefficient for BodyWeight is 0.012 with a standard error of 0.003. The degrees of freedom for the t test are equal to the number of PSUs (596) minus the number of strata (10). The "Estimates" table displays the estimated contrast and the corresponding hypothesis test. The estimated value for the contrast is –0.75. The estimated hazard for the nonsmokers is 0.47 times the estimated hazard for the current or former smokers. In this example data set, the contrast of interest is not significant at 0.05 levels. The "Type III Tests of Model Effects" table displays the Type 3 analysis. The effect variable Smoke has four levels. The F Value for Smoke is 1.49 with three numerator degrees of freedom and 584 denominator degrees of freedom.

Output 122.2.3 Parameter Estimates

Analysis of Maximum Likelihood Estimates						
Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Hazard Ratio
BodyWeight	586	0.011920	0.003150	3.78	0.0002	1.012
Smoke -1	586	-1.174048	0.738358	-1.59	0.1124	0.309
Smoke 1	586	-1.006515	0.577955	-1.74	0.0821	0.365
Smoke 2	586	-0.674183	0.557587	-1.21	0.2271	0.510
Smoke 3	586	0	.	.	.	1.000

Type III Tests of Model Effects				
Effect	Num DF	Den DF	F Value	Pr > F
BodyWeight	1	586	14.32	0.0002
Smoke	3	584	1.49	0.2162

Estimate						
Label	Estimate	Standard Error	DF	t Value	Pr > |t|	Exponentiated
Row 1	-0.7532	0.3864	586	-1.95	0.0518	0.4709

Example 122.3: Domain Analysis

(View the complete code for this example (surphex3.sas) in the example repository.)

This example uses a data set from the NHANES I Epidemiologic Followup Study (NHEFS); see Example 122.2 for more information about the NHEFS.

For illustration purposes, 1,891 observations from the 1992 NHEFS vital and tracing status data set are used to estimate the regression coefficients of a proportional hazards model. The observations are obtained from 22 strata; each stratum contains either two or three primary sampling units. The sum of observation weights for these selected units is almost 103 million. Observation weights range from 1,498 to 470,154 with a mean of 54,457.11 and a median of 45,246. The following variables are used in this example. Although this example uses the observation weights directly, Binder (1992) suggests that a scaled version of the observation weights would be useful to improve the performance of the optimization routine.

The following variables are created in the data set mortality:

- ID, unit identification
- VARSTRATA, stratum identification
- VARPSU, identification for primary sampling units
- SWEIGHT, sampling weight associated with each unit
- AGE, the subject's reported age at the 1992 interview if the subject was alive at that time; otherwise, the subject's age at death
- VITALSTATUS, vital status of subject in 1992 (1 = alive, 3 = dead, 4 = unknown, 5 = traced alive with direct subject contact, 6 = traced alive without direct subject contact)

- POVARIND, indicator for poverty area where subject's household was located at NHANES I (1971–1975) exam, (1 = poverty area, 2 = non-poverty area)
- GENDER, (1 = male, 2 = female)

```
data mortality;
   input ID VARSTRATA VARPSU SWEIGHT AGE VITALSTATUS POVARIND GENDER;
   datalines;
      1  03  1  13312    66  1   1   1
      2  03  1   7941    71  3   1   2
      3  03  1  16048     .  4   1   1
      4  03  3   9298    58  3   1   1
      5  03  2  15336    56  3   1   2
      6  03  1  14744    63  1   1   1
      7  03  2  83729    70  1   2   2
      8  03  3 106492    57  1   2   1
      9  03  3  78083    81  3   2   2
     10  03  3  55957    79  3   2   1

   ... more lines ...

   1890  13  1  88939  59  1  2   1
   1891  13  1  59218  75  1  2   2
;
```

Suppose you want to estimate the hazard function for mortality time after adjusting for the poverty area indicator in the base year survey population. The following SAS statements request a proportional hazards regression of age (AGE) on poverty indicator (POVARIND):

```
proc surveyphreg data = mortality nomcar;
   class povarind;
   strata varstrata;
   cluster varpsu;
   weight sweight;
   model age*vitalstatus(1 4 5 6) = povarind;
   domain gender;
run;
```

Subjects with VITALSTATUS 1, 4, 5, or 6 are considered alive. The CLASS statement specifies that POVARIND is a categorical variable, the WEIGHT statement identifies the sampling weights, the STRATA statement identifies variance strata, and the CLUSTER statement identifies variance PSUs. The DOMAIN statement requests three separate analyses: for the overall data set, the male subpopulation, and the female subpopulation respectively. There are 223 observation units with missing values on age. All the units with missing age have vital status 1, 4, 5, or 6. Therefore, these subjects are considered to be alive in the current survey year 1992. Age for every observation unit in the base year survey was known from 1971–1975 NHANES I. One reasonable approach is to determine the age of these 223 units based on their age from the NHANES I data set. However, for illustration purposes, this example does not include the observation units with missing age when estimating the regression coefficients. Instead, an analysis of just the set of respondents is requested by specifying the NOMCAR option in the PROC SURVEYPHREG statement. This option uses a variance estimator that accounts for the random size of the set of respondents.

Output 122.3.1 shows summary statistics for the overall analysis. A total of 1,891 observations are read from the input DATA= data set, but only 1,668 observations are used in the analysis. The remaining 223

observations have missing values in the variable age. The respondent data set represents almost 89.5 million units in the population. There are 22 strata and 55 clusters. Although only 57% observation units in the sample are alive, an estimated 69% observation units in the population are alive. This difference is reasonable because selection probabilities for observation units are not the same. If you do not use the sampling weights, then your sample-based estimators might be biased for the corresponding finite population quantities. The "Variance Estimation" table indicates that the NOMCAR option is used for variance estimation.

Output 122.3.1 Summary Statistics for the Entire Population

The SURVEYPHREG Procedure

Number of Observations Read	1891
Number of Observations Used	1668
Sum of Weights Read	1.0298E8
Sum of Weights Used	89439590

Design Summary	
Number of Strata	22
Number of Clusters	55

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
1668	717	951	57.01

Summary of the Weighted Number of Event and Censored Values

Total	Event	Censored	Percent Censored
89439590	27650348	61789242	69.08

Variance Estimation	
Method	Taylor Series
Missing Values	NOMCAR

Output 122.3.2 displays the estimated regression coefficients and their standard errors. Poverty index has two levels, and only one level is estimable. By default, PROC SURVEYPHREG estimates the first level (POVARIND 1) and assigns a zero value for the second level. The estimated regression coefficient is 0.385 with a standard error of 0.078. The estimated hazard for the poverty areas is 1.47 times higher than the estimated hazard for the non-poverty areas. The degrees of freedom are equal to the number of PSUs (55) minus the number of strata (22).

Output 122.3.2 Inference for the Entire Population

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Hazard Ratio
POVARIND 1	33	0.384961	0.077586	4.96	<.0001	1.470
POVARIND 2	33	0	.	.	.	1.000

Output 122.3.3 shows that 813 observation units in the sample are male, and they account for over 42.6 million males in the base year survey population. Approximately half of these observation units in the sample are censored, and an estimated 64.5% observation units are censored for the male subpopulation.

Output 122.3.3 Summary Statistics for the Male Subpopulation

The SURVEYPHREG Procedure

Domain Analysis for domain GENDER=1

Number of Observations Read	1891
Number of Observations Used	813
Sum of Weights Read	48887067
Sum of Weights Used	42629905

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
813	404	409	50.31

Summary of the Weighted Number of Event and Censored Values

Total	Event	Censored	Percent Censored
42629905	15126321	27503584	64.52

Output 122.3.4 shows that the estimated regression coefficient for POVARIND 1 is 0.425 with a standard error of 0.157. The estimated hazard for the males in the poverty areas is 1.53 times higher than the estimated hazard for the males in the non-poverty areas. The degrees of freedom for the t significant test for the male subpopulation are equal to the total number of PSUs (55) minus the total number of strata (22).

Output 122.3.4 Inference for the Male Subpopulation

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Hazard Ratio
POVARIND 1	33	0.424922	0.156583	2.71	0.0105	1.529
POVARIND 2	33	0	.	.	.	1.000

Output 122.3.5 displays some summary statistics for the female subpopulation. There are 855 observation units for females in the sample, and they represent over 46.8 million females in the base year survey population. Although 63.4% females in the sample are alive, an estimated 73.2% females in the subpopulation are alive.

Output 122.3.5 Summary Statistics for the Female Subpopulation

The SURVEYPHREG Procedure

Domain Analysis for domain GENDER=2

Number of Observations Read	1891
Number of Observations Used	855
Sum of Weights Read	54091604
Sum of Weights Used	46809685

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
855	313	542	63.39

Summary of the Weighted Number of Event and Censored Values

Total	Event	Censored	Percent Censored
46809685	12524027	34285658	73.24

Output 122.3.6 shows that the estimated proportional hazards regression coefficients for POVARIND for the females subpopulation (0.435) is higher than the estimated proportional hazards regression coefficients for POVARIND for the males subpopulation. The estimated hazard for the females in the poverty areas is 1.54 times higher than the estimated hazard for the females in the non-poverty areas. The degrees of freedom for the *t* significant test for the female subpopulation are equal to the total number of PSUs (55) minus the total number of strata (22).

Output 122.3.6 Inference for the Female Subpopulation

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Hazard Ratio
POVARIND 1	33	0.434579	0.115766	3.75	0.0007	1.544
POVARIND 2	33	0	.	.	.	1.000

Example 122.4: Variance Estimation by Using Replicate Weights

(View the complete code for this example (surphex4.sas) in the example repository.)

Consider the data set LibrarySurvey from "Getting Started: SURVEYPHREG Procedure" on page 10210. The selected sample contains 100 transactions from ten branch libraries. A set of replicate weights and jackknife coefficients are created by randomly assigning observation units in disjoint groups of nearly equal size within each stratum. A total of 46 different groups are created. The data set LibraryRepWeights is similar to the data set LibrarySurvey except that it also contains replicate weights repwt_1 to repwt_46. Each column of replicate weights is obtained by deleting one group of observations and adjusting the sampling weights for the other groups in that stratum (Rust 1985).

The data set LibraryJKCOEF contains the jackknife coefficient for every replicate sample. The variable replicate denotes the replicate number, donorstratum denotes the stratum identification for that replicate, and jkcoefficient denotes the jackknife coefficient for that replicate sample.

```
data LibrarySurvey;
   set LibrarySurvey;
   randomorder = ranuni(12345);
run;
proc sort data = LibrarySurvey out = LibrarySurvey;
   by Branch randomorder;
run;
data LibrarySurvey;
   set LibrarySurvey;
   array nGroup{10} (2 2 2 4 4 4 4 8 8 8);
   GroupPSU = mod(_N_,nGroup{Branch});
   drop randomorder nGroup1 nGroup2 nGroup3 nGroup4
        nGroup5 nGroup6 nGroup7 nGroup8 nGroup9 nGroup10;
run;

proc surveymeans data = LibrarySurvey varmethod = jk
               (outweights = LibraryRepWeights outjkcoefs = LibraryJKCOEF);
   weight SamplingWeight;
   strata Branch;
   cluster GroupPSU;
   var Age;
run;
```

It is not necessary to provide replicate weights to compute jackknife variance estimates using the SURVEYPHREG procedure. If you do not specify the replicate weights, then the procedure creates replicate weights for you. For this illustration, assume that LibraryRepWeights and LibraryJKCOEF are the only two data sets available for analysis.

The following SAS statements request a proportional hazards regression of lenBorrow on Age. The variable Returned is the censor indicator, and the value 0 indicates a censored observation. The WEIGHT statement specifies the sampling weight variable, and the REPWEIGHTS statement specifies replicate weight variables RepWt_1 to RepWt_46. The JKCOEFS= option in the REPWEIGHTS statement specifies the jackknife coefficient for each replicate sample. The VARMETHOD= option in the MODEL statement requests the jackknife variance estimation method. A STRATA statement is not required when the REPWEIGHTS statement is specified.

```
proc surveyphreg data = LibraryRepWeights varmethod = jk;
   weight SamplingWeight;
   repweights RepWt_: / jkcoefs = LibraryJKCOEF;
   model lenBorrow*Returned(0) = Age;
run;
```

Output 122.4.1 displays some summary information. The “Number of Observations,” “Censored Summary,” and “Weighted Censored Summary” tables are exactly the same as in the example discussed in “Getting Started: SURVEYPHREG Procedure” on page 10210. The “Variance Estimation” table displays information about the variance estimation, such as the name of the variance estimation method and the number of replicate samples.

Output 122.4.1 Summary Statistics for Overall Analysis

The SURVEYPHREG Procedure

Number of Observations Read	100
Number of Observations Used	100
Sum of Weights Read	11616.79
Sum of Weights Used	11616.79

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
100	90	10	10.00

Summary of the Weighted Number of Event and Censored Values

Total	Event	Censored	Percent Censored
11616.79	10449.22	1167.57	10.05

Variance Estimation	
Method	Jackknife
Replicate Weights	WORK.LIBRARYREPWEIGHTS
Number of Replicates	46

Output 122.4.2 shows that the estimated regression coefficient is 0.0616 with a standard error of 0.009. The denominator degrees of freedom (46) for the t test is equal to the number of replicates used. Note that the estimated proportional hazards regression coefficient is the same as the estimated proportional hazards regression coefficient in the example in "Getting Started: SURVEYPHREG Procedure" on page 10210, but the standard error and the denominator degrees of freedom are different. This is not surprising because these two examples use the same estimator to estimate the regression coefficients but different estimators to estimate the variance.

Output 122.4.2 Inferences Based on Survey Design for Overall Analysis

Analysis of Maximum Likelihood Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|	Hazard Ratio
Age	46	0.061593	0.009159	6.73	<.0001	1.064

Example 122.5: A Test of the Proportional Hazards Assumption Using Programming Statements

(View the complete code for this example (surphex5.sas) in the example repository.)

You can use programming statements in PROC SURVEYPHREG to create time-dependent covariates to test the proportional hazards assumption for complex survey data. Consider the data set mortality from Example 122.3. The data set contains 1,891 observations from the 1992 NHANES I Epidemiologic Follow-up Study (NHEFS) vital and tracing status.

Suppose you want to fit a proportional hazards model to these data and construct a test of the proportional hazards assumption on gender. The following statements request a proportional hazards regression of age on gender and x, where the time-dependent covariate x is created using the programming statements. The explanatory variable x assumes the value of the time variable age for the male subgroup. The variable vitalstatus is the censor indicator, and a value of 1, 4, 5, or 6 indicates a censored observation. The WEIGHT statement specifies the sampling weight, and the CLASS statement specifies that gender is a classification variable.

```
proc surveyphreg data = mortality nomcar;
   class gender;
   strata varstrata;
   cluster varpsu;
   weight sweight;
   model age*vitalstatus(1 4 5 6) = gender x;
   x = age*(gender=1);
run;
```

Output 122.5.1 displays some summary information. The "Number of Observations," "Censored Summary," and "Weighted Censored Summary" tables are exactly the same as in "Example 122.3: Domain Analysis" on page 10296.

Output 122.5.1 Data Summary, Censored Summary, and Information about Variance Estimation

The SURVEYPHREG Procedure

Number of Observations Read	1891
Number of Observations Used	1668
Sum of Weights Read	1.0298E8
Sum of Weights Used	89439590

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
1668	717	951	57.01

Summary of the Weighted Number of Event and Censored Values

Total	Event	Censored	Percent Censored
89439590	27650348	61789242	69.08

Output 122.5.1 *continued*

Variance Estimation	
Method	Taylor Series
Missing Values	NOMCAR

Output 122.5.2 displays the estimated regression coefficients and their standard errors. The variable gender has two levels, and only one level is estimable. By default, PROC SURVEYPHREG estimates the first level (GENDER 1) and assigns a zero value for the second level. The estimated regression coefficient is 1.61 with a standard error of 0.71. The estimated regression coefficient for x is –0.02 with a standard error of 0.01. The t statistic for x is –1.55 with a p-value of 0.13 on 33 degrees of freedom. This test suggests that an interaction between the time variable age and gender is not significant. Therefore, there is little evidence of an exponential trend over time in the hazard ratio for gender.

Output 122.5.2 Parameter Estimates

Analysis of Maximum Likelihood Estimates						
Parameter	**DF**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**	**Hazard Ratio**
GENDER 1	33	1.605505	0.709081	2.26	0.0303	4.980
GENDER 2	33	0	.	.	.	1.000
x	33	-0.015648	0.010079	-1.55	0.1301	0.984

References

Andersen, P. K., Borgan, Ø., Gill, R. D., and Keiding, N. (1992). *Statistical Models Based on Counting Processes*. New York: Springer-Verlag.

Beaumont, J.-F., and Patak, Z. (2012). "On the Generalized Bootstrap for Sample Surveys with Special Attention to Poisson Sampling." *International Statistical Review* 80:127–148.

Binder, D. A. (1983). "On the Variances of Asymptotically Normal Estimators from Complex Surveys." *International Statistical Review* 51:279–292.

Binder, D. A. (1990). "Fitting Cox's Proportional Hazards Models from Survey Data." In *Proceedings of the Survey Research Methods Section*, 342–347. Alexandria, VA: American Statistical Association.

Binder, D. A. (1992). "Fitting Cox's Proportional Hazards Models from Survey Data." *Biometrika* 79:139–147.

Boudreau, C., and Lawless, J. F. (2006). "Survival Analysis Based on the Proportional Hazards Model and Survey Data." *Canadian Journal of Statistics* 34:203–216.

Breslow, N. E. (1974). "Covariance Analysis of Censored Survival Data." *Biometrics* 30:89–99.

Brick, J. M., and Kalton, G. (1996). "Handling Missing Data in Survey Research." *Statistical Methods in Medical Research* 5:215–238.

Chambers, R. L., and Skinner, C. J. (2003). *Analysis of Survey Data*. Chichester, UK: John Wiley & Sons.

Chang, I.-M., Gelman, R., and Pagano, M. (1982). "Corrected Group Prognostic Curves and Summary Statistics." *Journal of Chronic Diseases* 35:669–674.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220. With discussion.

Cox, D. R. (1975). "Partial Likelihood." *Biometrika* 62:269–276.

Dippo, C. S., Fay, R. E., and Morganstein, D. H. (1984). "Computing Variances from Complex Samples with Replicate Weights." In *Proceedings of the Survey Research Methods Section*, 489–494. Alexandria, VA: American Statistical Association.

Efron, B. (1977). "The Efficiency of Cox's Likelihood Function for Censored Data." *Journal of the American Statistical Association* 72:557–565.

Fay, R. E. (1984). "Some Properties of Estimates of Variance Based on Replication Methods." In *Proceedings of the Survey Research Methods Section*, 495–500. Alexandria, VA: American Statistical Association.

Fay, R. E. (1989). "Theory and Application of Replicate Weighting for Variance Calculations." In *Proceedings of the Survey Research Methods Section*, 212–217. Alexandria, VA: American Statistical Association.

Firth, D. (1993). "Bias Reduction of Maximum Likelihood Estimates." *Biometrika* 80:27–38.

Fleming, T. R., and Harrington, D. P. (1991). *Counting Processes and Survival Analysis*. New York: John Wiley & Sons.

Fuller, W. A. (1975). "Regression Analysis for Sample Survey." *Sankhyā, Series C* 37:117–132.

Fuller, W. A. (2009). *Sampling Statistics*. Hoboken, NJ: John Wiley & Sons.

Fuller, W. A., Kennedy, W. J., Schnell, D., Sullivan, G., and Park, H. J. (1989). *PC CARP*. Ames: Iowa State University Statistical Laboratory.

Godambe, V. P., and Thompson, M. E. (1986). "Parameters of Superpopulation and Survey Population: Their Relationships and Estimation." *International Statistical Review* 54:127–138.

Harrell, F. E. (1986). "The PHGLM Procedure." In *SUGI Supplemental Library Guide, Version 5 Edition*. Cary, NC: SAS Institute Inc.

Heinze, G. (1999). *The Application of Firth's Procedure to Cox and Logistic Regression*. Technical Report 10, updated January 2001, Department of Medical Computer Sciences, Section of Clinical Biometrics, University of Vienna.

Heinze, G., and Schemper, M. (2001). "A Solution to the Problem of Monotone Likelihood in Cox Regression." *Biometrics* 51:114–119.

Judkins, D. R. (1990). "Fay's Method for Variance Estimation." *Journal of Official Statistics* 6:223–239.

Kalbfleisch, J. D., and Prentice, R. L. (2002). *The Statistical Analysis of Failure Time Data*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Kalton, G., and Kasprzyk, D. (1986). "The Treatment of Missing Survey Data." *Survey Methodology* 12:1–16.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Kish, L., and Frankel, M. R. (1974). "Inference from Complex Samples." *Journal of the Royal Statistical Society, Series B* 36:1–37.

Korn, E. L., and Graubard, B. I. (1999). *Analysis of Health Surveys*. New York: John Wiley & Sons.

Lane, P. W., and Nelder, J. A. (1982). "Analysis of Covariance and Standardization as Instances of Prediction." *Biometrics* 38:613–621.

Lawless, J. F. (2003). *Statistical Model and Methods for Lifetime Data*. 2nd ed. New York: John Wiley & Sons.

Lee, E. W., Wei, L. J., and Amato, D. A. (1992). "Cox-Type Regression Analysis for Large Numbers of Small Groups of Correlated Failure Time Observations." In *Survival Analysis: State of the Art*, edited by J. P. Klein and P. K. Goel, 237–247. Dordrecht, Netherlands: Kluwer Academic.

Lin, D. Y. (2000). "On Fitting Cox's Proportional Hazards Models to Survey Data." *Biometrika* 87:37–47.

Lin, D. Y., and Wei, L. J. (1989). "The Robust Inference for the Proportional Hazards Model." *Journal of the American Statistical Association* 84:1074–1078.

Lohr, S. L. (2022). *Sampling: Design and Analysis*. 3rd ed. Boca Raton, FL: CRC Press.

Lumley, T., and Scott, A. (2013). "Partial Likelihood Ratio Tests for the Cox Model under Complex Sampling." *Statistics in Medicine* 32:110–123.

Mashreghi, Z., Haziza, D., and Léger, C. (2016). "A Survey of Bootstrap Methods in Finite Population Sampling." *Statistics Surveys* 10:1–52.

McCarthy, P. J., and Snowden, C. B. (1985). *The Bootstrap and Finite Population Sampling*. Vital and Health Statistics, Series 2, no. 95. DHHS Publication Number 85–1369. Washington, DC: US Government Printing Office. https://www.cdc.gov/nchs/data/series/sr_02/sr02_095.pdf.

Mukhopadhyay, P. K. (2020). "Firth's Penalized Likelihood for Proportional Hazards Regressions for Complex Surveys." *Survey Methodology* 46:215–241. https://www150.statcan.gc.ca/n1/en/catalogue/12-001-X202000200004.

Pfeffermann, D. (1993). "The Role of Sampling Weights When Modeling Survey Data." *International Statistical Review* 61:317–337.

Rao, J. N. K., Scott, A. J., and Skinner, C. J. (1998). "Quasi-score Tests with Survey Data." *Statistica Sinica* 8:1059–1070.

Rao, J. N. K., and Shao, J. (1996). "On Balanced Half-Sample Variance Estimation in Stratified Random Sampling." *Journal of the American Statistical Association* 91:343–348.

Rao, J. N. K., and Shao, J. (1999). "Modified Balanced Repeated Replication for Complex Survey Data." *Biometrika* 86:403–415.

Rao, J. N. K., and Wu, C. F. J. (1988). "Resampling Inference with Complex Survey Data." *Journal of the American Statistical Association* 83:231–241.

Rao, J. N. K., Wu, C. F. J., and Yue, K. (1992). "Some Recent Work on Resampling Methods for Complex Surveys." *Survey Methodology* 18:209–217.

Rust, K. (1985). "Variance Estimation for Complex Estimators in Sample Surveys." *Journal of Official Statistics* 1:381–397.

Särndal, C.-E., and Lundström, S. (2005). *Estimation in Surveys with Nonresponse*. Chichester, UK: John Wiley & Sons.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Schoenfeld, D. A. (1982). "Partial Residuals for the Proportional Hazards Regression Model." *Biometrika* 69:239–241.

Shao, J., and Tu, D. (1995). *The Jackknife and Bootstrap*. New York: Springer-Verlag.

Sitter, R. R. (1992a). "Comparing Three Bootstrap Methods for Survey Data." *Canadian Journal of Statistics* 20:135–154.

Sitter, R. R. (1992b). "A Resampling Procedure for Complex Survey Data." *Journal of the American Statistical Association* 87:755–765.

Therneau, T. M. (1994). *A Package for Survival Analysis in S*. Technical Report 53, Section of Biostatistics, Mayo Clinic, Rochester, MN.

Therneau, T. M., Grambsch, P. M., and Fleming, T. R. (1990). "Martingale-Based Residuals and Survival Models." *Biometrika* 77:147–160.

Wolter, K. M. (2007). *Introduction to Variance Estimation*. 2nd ed. New York: Springer.

Chapter 123
The SURVEYREG Procedure

Contents

Overview: SURVEYREG Procedure **10310**
Getting Started: SURVEYREG Procedure **10311**
Syntax: SURVEYREG Procedure **10317**
- PROC SURVEYREG Statement 10318
- BY Statement 10331
- CLASS Statement 10332
- CLUSTER Statement 10333
- CONTRAST Statement 10333
- DOMAIN Statement 10335
- EFFECT Statement 10336
- ESTIMATE Statement 10338
- LSMEANS Statement 10339
- LSMESTIMATE Statement 10340
- MARGINS Statement 10341
- MODEL Statement 10342
- OUTPUT Statement 10345
- REPWEIGHTS Statement 10346
- SLICE Statement 10348
- STORE Statement 10348
- STRATA Statement 10348
- TEST Statement 10349
- WEIGHT Statement 10350

Details: SURVEYREG Procedure **10350**
- Missing Values 10350
- Survey Design Information 10351
- Computational Details 10352
- Variance Estimation 10356
- Testing 10364
- Domain Analysis 10365
- Computational Resources 10366
- Output Data Sets 10366
- Displayed Output 10368
- ODS Table Names 10372
- ODS Graphics 10374

Examples: SURVEYREG Procedure **10375**
- Example 123.1: Simple Random Sampling 10375

Example 123.2: Cluster Sampling 10377
Example 123.3: Regression Estimator for Simple Random Sample 10380
Example 123.4: Stratified Sampling 10381
Example 123.5: Regression Estimator for Stratified Sample 10388
Example 123.6: Stratum Collapse 10391
Example 123.7: Domain Analysis 10395
Example 123.8: Comparing Domain Statistics 10398
Example 123.9: Variance Estimate Using the Jackknife Method 10404
Example 123.10: Plotting a Smoothed Trend Line 10407
References **10408**

Overview: SURVEYREG Procedure

The SURVEYREG procedure performs regression analysis for sample survey data. This procedure can handle complex survey sample designs, including designs with stratification, clustering, and unequal weighting. The procedure fits linear models for survey data and computes regression coefficients and their variance-covariance matrix. PROC SURVEYREG also provides significance tests for the model effects and for any specified estimable linear functions of the model parameters. Using the regression model, the procedure can compute predicted values for the sample survey data.

PROC SURVEYREG uses elementwise regression to compute the regression coefficient estimators by generalized least squares estimation. The procedure assumes that the regression coefficients are the same across strata and primary sampling units (PSUs). To estimate the variance-covariance matrix for the regression coefficients, PROC SURVEYREG uses either the Taylor series (linearization) method or replication (resampling) methods to estimate sampling errors of estimators, based on complex sample designs. For details see Woodruff (1971); Fuller (1975); Särndal, Swensson, and Wretman (1992); Wolter (2007); Rust (1985); Dippo, Fay, and Morganstein (1984); Rao and Shao (1999); Rao, Wu, and Yue (1992); and Rao and Shao (1996).

PROC SURVEYREG uses the Output Delivery System (ODS), a SAS subsystem that provides capabilities for displaying and controlling the output from SAS procedures. ODS enables you to convert any of the output from PROC SURVEYREG into a SAS data set. For more information, see the section "ODS Table Names" on page 10372.

PROC SURVEYREG uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the SURVEYREG procedure, see the PLOTS= option in the PROC SURVEYREG statement and the section "ODS Graphics" on page 10374.

Getting Started: SURVEYREG Procedure

This section demonstrates how you can use PROC SURVEYREG to perform a regression analysis for sample survey data. For a complete description of the usage of PROC SURVEYREG, see the section "Syntax: SURVEYREG Procedure" on page 10317. The section "Examples: SURVEYREG Procedure" on page 10375 provides more detailed examples that illustrate the applications of PROC SURVEYREG.

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/svrgs.sas.)

Simple Random Sampling

Suppose that, in a junior high school, there are a total of 4,000 students in grades 7, 8, and 9. You want to know how household income and the number of children in a household affect students' average weekly spending for ice cream.

In order to answer this question, you draw a sample by using simple random sampling from the student population in the junior high school. You randomly select 40 students and ask them their average weekly expenditure for ice cream, their household income, and the number of children in their household. The answers from the 40 students are saved as the following SAS data set IceCream:

```
data IceCream;
   input Grade Spending Income Kids @@;
   datalines;
7   7  39  2   7   7  38  1   8  12  47  1
9  10  47  4   7   1  34  4   7  10  43  2
7   3  44  4   8  20  60  3   8  19  57  4
7   2  35  2   7   2  36  1   9  15  51  1
8  16  53  1   7   6  37  4   7   6  41  2
7   6  39  2   9  15  50  4   8  17  57  3
8  14  46  2   9   8  41  2   9   8  41  1
9   7  47  3   7   3  39  3   7  12  50  2
7   4  43  4   9  14  46  3   8  18  58  4
9   9  44  3   7   2  37  1   7   1  37  2
7   4  44  2   7  11  42  2   9   8  41  2
8  10  42  2   8  13  46  1   7   2  40  3
9   6  45  1   9  11  45  4   7   2  36  1
7   9  46  1
;
```

In the data set IceCream, the variable Grade indicates a student's grade. The variable Spending contains the dollar amount of each student's average weekly spending for ice cream. The variable Income specifies the household income, in thousands of dollars. The variable Kids indicates how many children are in a student's family.

The following PROC SURVEYREG statements request a regression analysis:

```
title1 'Ice Cream Spending Analysis';
title2 'Simple Random Sample Design';
proc surveyreg data=IceCream total=4000;
   class Kids;
```

```
   model Spending = Income Kids / solution;
run;
```

The PROC SURVEYREG statement invokes the procedure. The TOTAL=4000 option specifies the total in the population from which the sample is drawn. The CLASS statement requests that the procedure use the variable Kids as a classification variable in the analysis. The MODEL statement describes the linear model that you want to fit, with Spending as the dependent variable and Income and Kids as the independent variables. The SOLUTION option in the MODEL statement requests that the procedure output the regression coefficient estimates.

Figure 123.1 displays the summary of the data, the summary of the fit, and the levels of the classification variable Kids. The "Fit Statistics" table displays the denominator degrees of freedom, which are used in F tests and t tests in the regression analysis.

Figure 123.1 Summary of Data

Ice Cream Spending Analysis
Simple Random Sample Design

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Spending

Data Summary	
Number of Observations	40
Mean of Spending	8.75000
Sum of Spending	350.00000

Fit Statistics	
R-Square	0.8132
Root MSE	2.4506
Denominator DF	39

Class Level Information		
CLASS Variable	Levels	Values
Kids	4	1 2 3 4

Figure 123.2 displays the tests for model effects. The effect Income is significant in the linear regression model, while the effect Kids is not significant at the 5% level.

Figure 123.2 Testing Effects in the Regression

Tests of Model Effects			
Effect	Num DF	F Value	Pr > F
Model	4	119.15	<.0001
Intercept	1	153.32	<.0001
Income	1	324.45	<.0001
Kids	3	0.92	0.4385

Note: The denominator degrees of freedom for the F tests is 39.

The regression coefficient estimates and their standard errors and associated t tests are displayed in Figure 123.3.

Figure 123.3 Regression Coefficients

Estimated Regression Coefficients				
Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	-26.084677	2.46720403	-10.57	<.0001
Income	0.775330	0.04304415	18.01	<.0001
Kids 1	0.897655	1.12352876	0.80	0.4292
Kids 2	1.494032	1.24705263	1.20	0.2381
Kids 3	-0.513181	1.33454891	-0.38	0.7027
Kids 4	0.000000	0.00000000	.	.

Note: The degrees of freedom for the t tests is 39.
Matrix X'X is singular and a generalized inverse was used to solve the normal equation
Estimates are not unique.

Stratified Sampling

Suppose that the previous student sample is actually selected by using a stratified sample design. The strata are the grades in the junior high school: 7, 8, and 9. Within the strata, simple random samples are selected. Table 123.1 provides the number of students in each grade.

Table 123.1 Students in Grades

Grade	Number of Students
7	1,824
8	1,025
9	1,151
Total	**4,000**

In order to analyze this sample by using PROC SURVEYREG, you need to input the stratification information by creating a SAS data set that contains the information in Table 123.1. The following SAS statements create such a data set, named StudentTotals:

```
data StudentTotals;
   input Grade _TOTAL_;
   datalines;
7 1824
8 1025
9 1151
;
```

The variable Grade is the stratification variable, and the variable _TOTAL_ contains the total numbers of students in each stratum in the survey population. PROC SURVEYREG requires you to use the keyword _TOTAL_ as the name of the variable that contains the population totals.

When the sample design is stratified and the stratum sampling rates are unequal, you should use sampling weights to reflect this information in the analysis. For this example, the appropriate sampling weights are the

reciprocals of the probabilities of selection. You can use the following DATA step to create the sampling weights:

```
data IceCream;
   set IceCream;
   if Grade=7 then Prob=20/1824;
   if Grade=8 then Prob=9/1025;
   if Grade=9 then Prob=11/1151;
   Weight=1/Prob;
run;
```

If you use PROC SURVEYSELECT to select your sample, PROC SURVEYSELECT creates these sampling weights for you.

The following statements demonstrate how you can fit a linear model while incorporating the sample design information (stratification and unequal weighting):

```
ods graphics on;
title1 'Ice Cream Spending Analysis';
title2 'Stratified Sample Design';
proc surveyreg data=IceCream total=StudentTotals;
   strata Grade /list;
   model Spending = Income;
   weight Weight;
run;
```

Comparing these statements to those in the section "Simple Random Sampling" on page 10311, you can see how the TOTAL=StudentTotals option replaces the previous TOTAL=4000 option.

The STRATA statement specifies the stratification variable Grade. The LIST option in the STRATA statement requests that the stratification information be displayed. The WEIGHT statement specifies the weight variable.

Figure 123.4 summarizes the data information, the sample design information, and the fit information. Because of the stratification, the denominator degrees of freedom for F tests and t tests are 37, which are different from those in the analysis in Figure 123.1.

Figure 123.4 Summary of the Regression

Ice Cream Spending Analysis
Stratified Sample Design

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Spending

Data Summary	
Number of Observations	40
Sum of Weights	4000.0
Weighted Mean of Spending	9.14130
Weighted Sum of Spending	36565.2

Design Summary	
Number of Strata	3

Figure 123.4 *continued*

Fit Statistics	
R-Square	0.8037
Root MSE	2.4371
Denominator DF	37

Figure 123.5 displays the following information for each stratum: the value of the stratification variable, the number of observations (sample size), the total population size, and the sampling rate (fraction).

Figure 123.5 Stratification Information

Stratum Information				
Stratum Index	Grade	N Obs	Population Total	Sampling Rate
1	7	20	1824	1.10%
2	8	9	1025	0.88%
3	9	11	1151	0.96%

Figure 123.6 displays the tests for significance of the model effects. The Income effect is strongly significant at the 5% level.

Figure 123.6 Testing Effects

Tests of Model Effects			
Effect	Num DF	F Value	Pr > F
Model	1	492.39	<.0001
Intercept	1	225.81	<.0001
Income	1	492.39	<.0001

Note: The denominator degrees of freedom for the F tests is 37.

Figure 123.7 displays the regression coefficient estimates, their standard errors, and the associated *t* tests for the stratified sample.

Figure 123.7 Regression Coefficients

Estimated Regression Coefficients				
Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	-23.416322	1.55827214	-15.03	<.0001
Income	0.731052	0.03294520	22.19	<.0001

Note: The degrees of freedom for the t tests is 37.

You can request other statistics and tests by using PROC SURVEYREG. You can also analyze data from a more complex sample design. The remainder of this chapter provides more detailed information.

When ODS Graphics is enabled and the model contains a single continuous regressor, PROC SURVEYREG provides a fit plot that displays the regression line and the confidence limits of the mean predictions. Figure 123.8 displays the fit plot for the regression model of Spending as a function of Income. The regression line and confidence limits of mean prediction are overlaid by a bubble plot of the data, in which

the bubble area is proportional to the sampling weight of an observation.

Figure 123.8 Regression Fitting

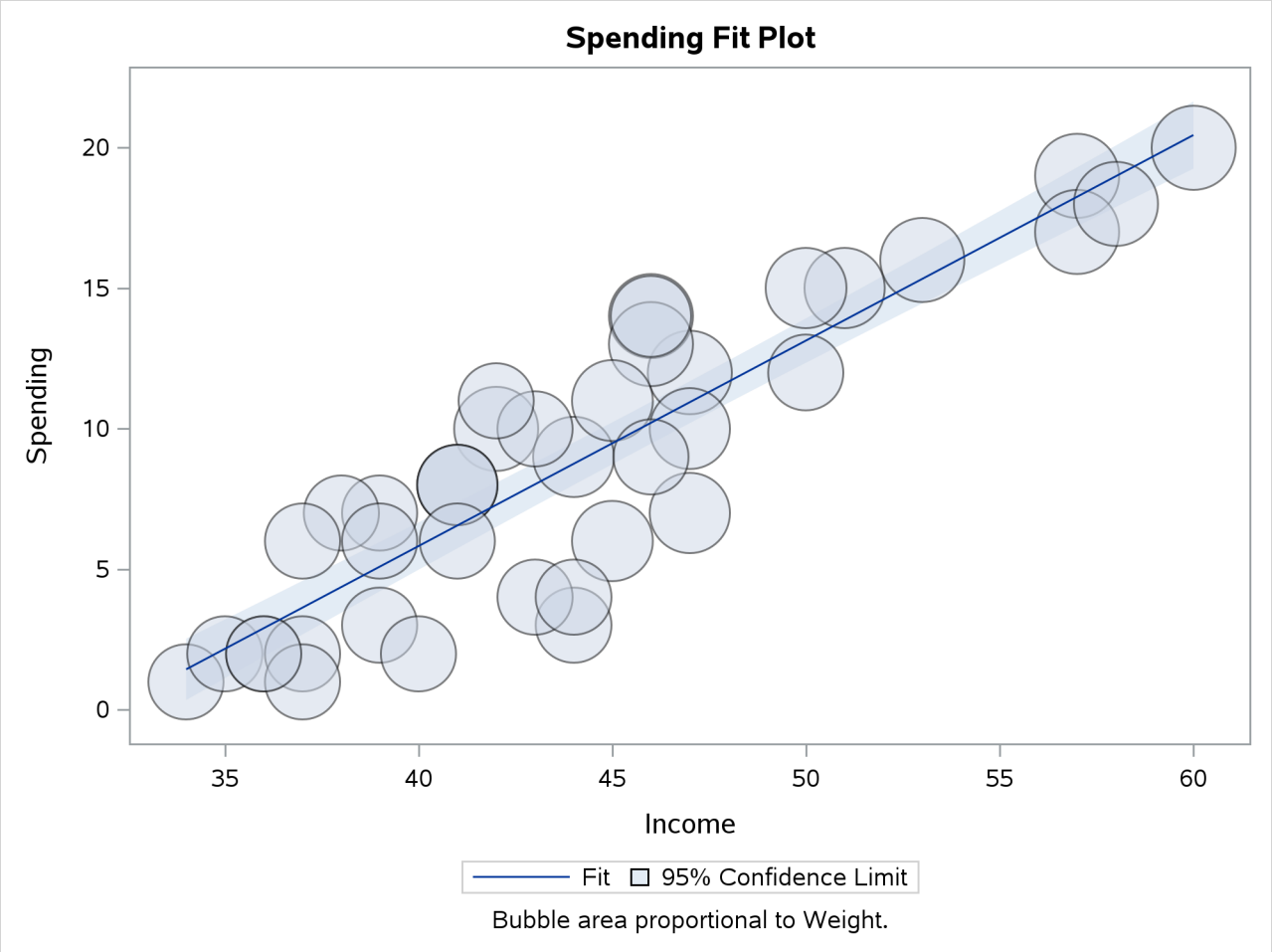

Output Data Sets

You can use the OUTPUT statement to create a new SAS data set that contains the estimated linear predictors and their standard error estimates, the residuals from the linear regression, and the confidence limits for the predictors. See the section "OUTPUT Statement" on page 10345 for more details.

You can use the Output Delivery System (ODS) to create SAS data sets that capture the outputs from PROC SURVEYREG. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

For example, to save the ParameterEstimates table (Figure 123.7) in the previous section in an output data set, you use the ODS OUTPUT statement as follows:

```
title1 'Ice Cream Spending Analysis';
title2 'Stratified Sample Design';
proc surveyreg data=IceCream total=StudentTotals;
   strata Grade /list;
   model Spending = Income;
   weight Weight;
   ods output ParameterEstimates = MyParmEst;
```

```
run;
```

The statement

```
ods output ParameterEstimates = MyParmEst;
```

requests that the ParameterEstimates table that appears in Figure 123.7 be placed into a SAS data set MyParmEst.

The PRINT procedure displays observations of the data set MyParmEst:

```
proc print data=MyParmEst;
run;
```

Figure 123.9 displays the observations in the data set MyParmEst. The section "ODS Table Names" on page 10372 gives the complete list of the tables produced by PROC SURVEYREG.

Figure 123.9 The Data Set MyParmEst

Ice Cream Spending Analysis
Stratified Sample Design

OBS	Parameter	Estimate	StdErr	DenDF	tValue	Probt
1	Intercept	-23.416322	1.55827214	37	-15.03	<.0001
2	Income	0.731052	0.03294520	37	22.19	<.0001

Syntax: SURVEYREG Procedure

The following statements are available in the SURVEYREG procedure:

PROC SURVEYREG < *options* > ;
- **BY** *variables* ;
- **CLASS** *variables* ;
- **CLUSTER** *variables* ;
- **CONTRAST** *'label' effect values* < . . . *effect values* > < / *options* > ;
- **DOMAIN** *variables* < *variable*∗*variable variable*∗*variable*∗*variable* . . . > ;
- **EFFECT** *name* = *effect-type* (*variables* < / *options*>) ;
- **ESTIMATE** < *'label'*> *estimate-specification* < / *options* > ;
- **LSMEANS** < *model-effects* > < / *options* > ;
- **LSMESTIMATE** *model-effect lsmestimate-specification* < / *options* > ;
- **MARGINS** *effects* < / *options* > ;
- **MODEL** *dependent* = < *effects* > < / *options* > ;
- **OUTPUT** < *keyword*< =*variable-name* > . . . *keyword*< =*variable-name* > > < / *option* > ;
- **REPWEIGHTS** *variables* < / *options* > ;
- **SLICE** *model-effect* < / *options* > ;
- **STORE** < **OUT=** >*item-store-name* < /**LABEL=***'label'*> ;
- **STRATA** *variables* < / *options* > ;
- **TEST** < *model-effects* > < / *options* > ;
- **WEIGHT** *variable* ;

The PROC SURVEYREG and MODEL statements are required. If your model contains classification effects, you must list the classification variables in a CLASS statement.

The CLASS and EFFECT statements (if specified) must precede the MODEL statement, and the CONTRAST, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, and SLICE statements (if specified) must follow the MODEL statement.

The CLASS, CLUSTER, CONTRAST, EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, REPWEIGHTS, SLICE, STRATA, and TEST statements can appear multiple times. You should use only one of each of the following statements: MODEL, WEIGHT, STORE, and OUTPUT.

The rest of this section provides detailed syntax information for each of the preceding statements, except the EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, SLICE, STORE, and TEST statements. These statements are also available in many other procedures. Summary descriptions of functionality and syntax for these statements are shown in this chapter, and full documentation about them is available in Chapter 20, "Shared Concepts and Topics."

The syntax descriptions begin with the PROC SURVEYREG statement; the remaining statements are covered in alphabetical order.

PROC SURVEYREG Statement

PROC SURVEYREG < *options* > **;**

The PROC SURVEYREG statement invokes the SURVEYREG procedure. It optionally names the input data sets and specifies the variance estimation method.

Table 123.2 summarizes the *options* available in the PROC SURVEYREG statement.

Table 123.2 PROC SURVEYREG Statement Options

Option	Description
ALPHA=	Sets the confidence level
DATA=	Specifies the SAS data set to be analyzed
MISSING	Treats missing values as a valid category
NAMELEN=	Specifies the length of effect names
NOMCAR	Treats missing values as *not missing completely at random*
ORDER=	Specifies the sort order
PLOTS=	Requests plots from ODS Graphics
RATE=	Specifies the sampling rate
TOTAL=	Specifies the total number of primary sampling units
TRUNCATE	Specifies class levels using no more than the first 16 characters of the formatted values
VARMETHOD=	Specifies the variance estimation method

You can specify the following *options* in the PROC SURVEYREG statement:

ALPHA=*α*

sets the confidence level for confidence limits. The value of the ALPHA= option must be between 0 and 1, and the default value is 0.05. A confidence level of α produces $100(1-\alpha)\%$ confidence limits. The default of ALPHA=0.05 produces 95% confidence limits.

DATA=*SAS-data-set*

specifies the SAS data set to be analyzed by PROC SURVEYREG. If you omit the DATA= option, the procedure uses the most recently created SAS data set.

MISSING

treats missing values as a valid (nonmissing) category for all categorical variables, which include CLASS, STRATA, CLUSTER, and DOMAIN variables.

By default, if you do not specify the MISSING option, an observation is excluded from the analysis if it has a missing value. For more information, see the section "Missing Values" on page 10350.

NAMELEN=*n*

specifies the length of effect names in tables and output data sets to be *n* characters, where *n* is a value between 40 and 200. The default length is 40 characters.

NOMCAR

treats that missing values in the variance computation as *not missing completely at random* (NOMCAR) for Taylor series variance estimation. When you specify this option, PROC SURVEYREG computes variance estimates by analyzing the nonmissing values as a domain or subpopulation, where the entire population includes both nonmissing and missing domains. For more information, see the section "Missing Values" on page 10350.

By default, PROC SURVEYREG completely excludes an observation from analysis if that observation has a missing value, unless you specify the MISSING option. Note that the NOMCAR option has no effect on a classification variable when you specify the MISSING option, which treats missing values as a valid nonmissing level.

The NOMCAR option applies only to Taylor series variance estimation; it is ignored for replication methods.

ORDER=DATA | FORMATTED | FREQ | INTERNAL

specifies the sort order for the levels of the classification variables (which are specified in the CLASS statement).

This option also determines the sort order for the levels of DOMAIN variables.

This option applies to the levels for all classification variables, except when you use the (default) ORDER=FORMATTED option with numeric classification variables that have no explicit format. In that case, the levels of such variables are ordered by their internal value.

The ORDER= option can take the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set
FORMATTED	External formatted value, except for numeric variables with no explicit format, which are sorted by their unformatted (internal) value
FREQ	Descending frequency count; levels with the most observations come first in the order
INTERNAL	Unformatted value

By default, ORDER=FORMATTED. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

PLOTS < (*global-plot-options*) > < = *plot-request* < (*plot-option*) > >
PLOTS < (*global-plot-options*) > < = (*plot-request* < (*plot-option*) > < . . . *plot-request* < (*plot-option*) > >) >

controls the plots that are produced through ODS Graphics.

When ODS Graphics is enabled and when the regression model depends on at most one continuous variable as a regressor, excluding the intercept, the PLOTS= option in the PROC SURVEYREG statement controls fit plots for the regression.

A *plot-request* identifies the plot, and a *plot-option* controls the appearance and content of the plot. You can specify *plot-options* in parentheses after a *plot-request*. A *global-plot-option* applies to all plots for which it is available unless it is altered by a specific *plot-option*. You can specify *global-plot-options* in parentheses after the PLOTS option.

When you specify only one *plot-request*, you can omit the parentheses around it. Here are a few examples of requesting plots:

```
plots=all
plots(weight=heatmap)=fit
```

When the regression model depends on at most one continuous variable as a regressor, excluding the intercept, PROC SURVEYREG provides a bubble plot or a heat map for model fitting. In a bubble plot, the bubble area is proportional to the weight of an observation. In a heat map, the heat color represents the sum of the weights at the corresponding location. The default plot depends on the number of observations in your data. That is, for a data set that contains 100 observations or less, a bubble plot is the default. For a data set that contains more than 100 observations, a heat map is the default.

ODS Graphics must be enabled before you can request a plot. For example:

```
ods graphics on;
proc surveyreg plots=fit;
   model height=weight;
run;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, the ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements can produce plots that are associated with their analyses. For information about these plots, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Global Plot Options

A *global-plot-option* applies to all plots for which the option is available unless it is altered by a specific *plot-option*. You can specify the following *global-plot-options*:

MAXSIZE=*Bmax*
specifies the maximum size of the radius of the bubble plot. Any observation that maps to a larger size than *Bmax* is displayed at the size specified for *Bmax*. *Bmax* must be larger than the *Bmin* value specified in the MINSIZE= option. By default, MAXSIZE=21. This option is ignored when WEIGHT=HEATMAP.

MAXWVALUE=*Wmax*
displays all observations for which weight is greater than *Wmax* with a bubble whose radius is set by the MAXSIZE= option. By default, *Wmax* equals the largest weight in the data. This option is ignored when WEIGHT=HEATMAP.

MINSIZE=*Bmin*
specifies the minimum size of the radius of the bubble plot. Any observation that maps to a smaller size than *Bmin* is displayed at the size specified for *Bmin*. *Bmin* must be smaller than the *Bmax* value specified in the MAXSIZE= option. By default, MINSIZE=3.5. This option is ignored when WEIGHT=HEATMAP.

NBINS=*nbin1* < *nbin2* >
specifies the number of bins for the heat map of the observation weights in the fit plot. The number of bins also depends on the value of the SHAPE= option:

- If SHAPE=RECTANGULAR, the number of bins is calculated as follows:
  - If you specify only one number, *nbin1*, then it is used for both the horizontal and vertical axes. For example, if you specify NBINS=10, then PROC SURVEYREG creates 100 bins (10×10).
  - If you specify two numbers, then *nbin1* is used for the horizontal axis and *nbin2* is used for the vertical axis. For example, if you specify NBINS=10 20, then PROC SURVEYREG creates 200 bins (10×20).
- If you specify SHAPE=HEXAGONAL, then PROC SURVEYREG calculates the number of rectangular bins by the method previously described. Then the procedure creates a heat map that has hexagonal bins of approximately the same size as those rectangular bins.

If you specify this option, then by default WEIGHT=HEATMAP.

If you do not specify this option, then the number of bins is determined by first using the algorithm that is discussed in the section "ODS Graphics" on page 5527 in Chapter 73, "The KDE Procedure," and then multiplying the resulting numbers of bins by three.

ONLY
suppresses the default plots and requests only the plots that are specified as *plot-requests*.

WEIGHT=BUBBLE | HEATMAP
requests either a bubble plot or a heat map of the data as an overlay on the regression line and confidence limits band of the prediction in a fit plot. You can specify the following options:

BUBBLE overlays a bubble plot in which the bubble area is proportional to the weight of an observation.

HEATMAP overlays a heat map in which the heat color represents the sum of the weights at the corresponding location.

If you specify the NBINS= option, then by default WEIGHT=HEATMAP.

If you do not specify this option, then the default plot depends on the number of observations in your data. For a data set that contains 100 observations or less, the default is WEIGHT=BUBBLE. For a data set that contains more than 100 observations, the default is WEIGHT=HEATMAP.

Plot Requests

You can specify the following *plot-requests*:

ALL
requests all appropriate plots.

FIT < (*plot-options*) >
requests a plot that displays the model fitting for a model that depends on at most one regressor, excluding the intercept. The plot is either a bubble plot or a heat map that is overlaid with the regression line and confidence band of the prediction.

The FIT plot request has the following *plot-options*:

MAXSIZE=*Bmax*
specifies the maximum size of the radius of the bubble plot. Any observation that maps to a larger size than *Bmax* is displayed at the size specified for *Bmax*. *Bmax* must be larger than the *Bmin* value specified in the MINSIZE= option. By default, MAXSIZE=21. This option is ignored when WEIGHT=HEATMAP.

MAXWVALUE=*Wmax*
displays all observations for which weight is greater than *Wmax* with a bubble whose radius is set by the MAXSIZE= option. By default, *Wmax* equals the largest weight in the data. This option is ignored when WEIGHT=HEATMAP.

MINSIZE=*Bmin*
specifies the minimum size of the radius of the bubble plot. Any observation that maps to a smaller size than *Bmin* is displayed at the size specified for *Bmin*. *Bmin* must be smaller than the *Bmax* value specified in the MAXSIZE= option. By default, MINSIZE=3.5. This option is ignored when WEIGHT=HEATMAP.

NBINS=*nbin1* < *nbin2* >

specifies the number of bins for the heat map of the observation weights in the fit plot. The number of bins also depends on the value of the SHAPE= option:

- If SHAPE=RECTANGULAR, the number of bins is calculated as follows:
  - If you specify only one number, *nbin1*, then it is used for both the horizontal and vertical axes. For example, if you specify NBINS=10, then PROC SURVEYREG creates 100 bins (10×10).
  - If you specify two numbers, then *nbin1* is used for the horizontal axis and *nbin2* is used for the vertical axis. For example, if you specify NBINS=10 20, then PROC SURVEYREG creates 200 bins (10×20).
- If you specify SHAPE=HEXAGONAL, then PROC SURVEYREG calculates the number of rectangular bins by the method previously described. Then the procedure creates a heat map that has hexagonal bins of approximately the same size as those rectangular bins.

If you specify this option, then by default WEIGHT=HEATMAP.

If you do not specify this option, then the number of bins is determined by first using the algorithm that is discussed in the section "ODS Graphics" on page 5527 in Chapter 73, "The KDE Procedure," and then multiplying the resulting numbers of bins by three.

WEIGHT=BUBBLE | HEATMAP

requests either a bubble plot or a heat map of the data as an overlay on the regression line and confidence limits band of the prediction in a fit plot. You can specify the following options:

BUBBLE overlays a bubble plot in which the bubble area is proportional to the weight of an observation.

HEATMAP overlays a heat map in which the heat color represents the sum of the weights at the corresponding location.

If you specify the NBINS= option, then by default WEIGHT=HEATMAP.

If you do not specify this option, then the default plot depends on the number of observations in your data. For a data set that contains 100 observations or less, the default is WEIGHT=BUBBLE. For a data set that contains more than 100 observations, the default is WEIGHT=HEATMAP.

SHAPE=RECTANGULAR | HEXAGONAL

SHAPE=REC | HEX

requests either rectangular or hexagonal bins for a heat map of the data. Thus, this option implies WEIGHT=HEATMAP by default. By default, SHAPE=RECTANGULAR.

NONE
suppresses all plots.

RATE=*value* | *SAS-data-set*
R=*value* | *SAS-data-set*
specifies the sampling rate, which PROC SURVEYREG uses to compute a finite population correction for Taylor series or bootstrap variance estimation. This option is ignored for the jackknife or balanced repeated replication (BRR) variance estimation method.

If your sample design has multiple stages, you should specify the *first-stage sampling rate*, which is the ratio of the number of primary sampling units (PSUs) in the sample to the total number of PSUs in the population.

You can specify the sampling rate in either of the following ways:

value specifies a nonnegative number to use for a nonstratified design or for a stratified design that has the same sampling rate in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the sampling rates for a stratified design that has different sampling rates in the strata. You must provide the sampling rates in the data set variable named _RATE_. The sampling rates must be nonnegative numbers.

You can specify sampling rates as numbers between 0 and 1. Or you can specify sampling rates in percentage form as numbers between 1 and 100, which PROC SURVEYREG converts to proportions. The procedure treats the value 1 as 100% instead of 1%.

For more information, see the section "Specification of Population Totals and Sampling Rates" on page 10351.

If you do not specify either the RATE= or TOTAL= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. You cannot specify both the RATE= and TOTAL= options.

TOTAL=*value* | *SAS-data-set*
N=*value* | *SAS-data-set*
specifies the total number of primary sampling units (PSUs) in the study population. PROC SURVEYREG uses this information to compute a finite population correction for Taylor series or bootstrap variance estimation. This option is ignored for the jackknife or BRR variance estimation method.

You can specify the total number of PSUs in either of the following ways:

value specifies a positive number to use for a nonstratified design or for a stratified design that has the same population total in each stratum.

SAS-data-set specifies a *SAS-data-set* that contains the stratification variables and the population totals for a stratified design that has different population totals in the strata. You must provide the stratum totals in the data set variable named _TOTAL_. The stratum totals must be positive numbers.

For more information, see the section "Specification of Population Totals and Sampling Rates" on page 10351.

If you do not specify either the TOTAL= or RATE= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. You cannot specify both the TOTAL= and RATE= options.

TRUNCATE

specifies that class levels should be determined using no more than the first 16 characters of the formatted values of the CLASS, STRATA, and CLUSTER variables.

VARMETHOD=*method* **<(***method-options***)>**

specifies the variance estimation *method*. PROC SURVEYREG provides the Taylor series method and the following replication (resampling) methods: balanced repeated replication (BRR), bootstrap, and jackknife.

Table 123.3 summarizes the available *methods* and *method-options*.

Table 123.3 Variance Estimation Methods

method	**Variance Estimation Method**	*method-options*
BOOTSTRAP	Bootstrap	CENTER= MH=*value* | *SAS-data-set* OUTWEIGHTS=*SAS-data-set* REPS=*number* REPWARN <=NO | YES> SEED=*number*
BRR	Balanced repeated replication	CENTER= FAY <=*value*> HADAMARD=*SAS-data-set* OUTWEIGHTS=*SAS-data-set* PRINTH REPS=*number* REPWARN <=NO | YES>
JACKKNIFE | JK	Jackknife	CENTER= OUTJKCOEFS=*SAS-data-set* OUTWEIGHTS=*SAS-data-set* REPWARN <=NO | YES>
TAYLOR	Taylor series linearization	None

For VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, and VARMETHOD=JACKKNIFE, you can specify *method-options* in parentheses after the variance estimation *method*. For example:

```
varmethod=BRR(reps=60 outweights=myReplicateWeights)
```

By default, VARMETHOD=JACKKNIFE if you also specify a REPWEIGHTS statement; otherwise, VARMETHOD=TAYLOR by default.

You can specify the following *methods*:

BOOTSTRAP <(*method-options*)>

requests variance estimation by the bootstrap method. For more information, see the section "Bootstrap Method" on page 10357.

The bootstrap method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you use a REPWEIGHTS statement to provide replicate weights.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Bootstrap Method" on page 10357.

MH=*value* | **(***values***)** | *SAS-data-set*

specifies the number of PSUs to select for the bootstrap replicate samples. You can provide bootstrap stratum sample sizes m_h by specifying a list of *values* or a *SAS-data-set*. Alternatively, you can provide a single bootstrap sample size *value* to use for all strata or for a nonstratified design. You can specify the number of replicate samples in the REPS= option. For more information, see the section "Bootstrap Method" on page 10357.

Each bootstrap sample size m_h must be a positive integer and must be less than n_h, which is the total number of PSUs in stratum h. By default, $m_h = n_h - 1$ for a stratified design. For a nonstratified design, the bootstrap sample size *value* must be less than n (the total number of PSUs in the sample). By default, $m = n - 1$ for a nonstratified design.

You can provide bootstrap sample sizes by specifying one of the following forms:

MH=*value*

specifies a single bootstrap sample size *value* to use for all strata or for a nonstratified design.

MH=(*values***)**

specifies a list of stratum bootstrap sample size *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. The number of values must not be less than the number of strata in the DATA= input data set.

Each stratum sample size value must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

MH=*SAS-data-set*

names a *SAS-data-set* that contains the stratum bootstrap sample sizes. You must provide the sample sizes in a data set variable named _NSIZE_ or SampleSize.

The *SAS-data-set* must contain all stratification variables that you specify in the STRATA statement. It must also contain all stratum levels that appear in the DATA= input data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets.

Each value of the _NSIZE_ or SampleSize variable must be a positive integer and must be less than the total number of PSUs in the corresponding stratum.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* in which to store the bootstrap replicate weights that PROC SURVEYREG creates. For information about replicate weights, see the section "Bootstrap Method" on page 10357. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10367.

This *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

REPS=*number*

specifies the *number* of replicates for bootstrap variance estimation. The value of *number* must be an integer greater than 1. Increasing the number of replicates improves the estimation precision but also increases the computation time. By default, REPS=250.

REPWARN < =NO | YES >

controls the log message that PROC SURVEYREG provides when the parameters cannot be estimated from one or more replicates because they are not estimable in those replicates. The procedure displays this log message as a warning if you specify REPWARN or REPWARN=YES. By default, the procedure displays the log message as a note, as if you specified REPWARN=NO.

For more information, see the subsection "Variance Estimation" on page 10358 in the section "Bootstrap Method" on page 10357.

SEED=*number*

specifies the initial seed for random number generation for bootstrap replicate sampling.

If you do not specify this option or if you specify a *number* that is negative or 0, PROC SURVEYREG uses the time of day from the system clock to obtain an initial seed.

To reproduce the same bootstrap replicate weights and the same analysis in a subsequent execution of PROC SURVEYREG, you can specify the same initial seed that was used in the original analysis.

PROC SURVEYREG displays the value of the initial seed in the "Variance Estimation" table.

BRR < (*method-options*) >

requests variance estimation by balanced repeated replication (BRR). This method requires a stratified sample design where each stratum contains two primary sampling units (PSUs). When you specify this method, you must also specify a STRATA statement unless you provide replicate weights by using the REPWEIGHTS statement. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10359.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES

defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10359.

FAY < =*value* >

requests Fay's method, which is a modification of the BRR method. For more information, see the section "Fay's BRR Method" on page 10360.

You can specify the *value* of the Fay coefficient, which is used in converting the original sampling weights to replicate weights. The Fay coefficient must be a nonnegative number less than 1. By default, the Fay coefficient is 0.5.

HADAMARD=*SAS-data-set*

H=*SAS-data-set*

names a *SAS-data-set* that contains the Hadamard matrix for BRR replicate construction. If you do not specify this *method-option*, PROC SURVEYREG generates an appropriate Hadamard matrix for replicate construction. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 10359 and "Hadamard Matrix" on page 10363.

If a Hadamard matrix of a particular dimension exists, it is not necessarily unique. Therefore, if you want to use a specific Hadamard matrix, you must provide the matrix as a *SAS-data-set* in this *method-option*.

In this *SAS-data-set*, each variable corresponds to a column and each observation corresponds to a row of the Hadamard matrix. You can use any variable names in this data set. All values in the data set must equal either 1 or –1. You must ensure that the matrix you provide is indeed a Hadamard matrix—that is, $\mathbf{A}'\mathbf{A} = R\,\mathbf{I}$, where $\mathbf{A}$ is the Hadamard matrix of dimension R and $\mathbf{I}$ is an identity matrix. PROC SURVEYREG does not check the validity of the Hadamard matrix that you provide.

The *SAS-data-set* must contain at least H variables, where H denotes the number of first-stage strata in your design. If the data set contains more than H variables, PROC

SURVEYREG uses only the first *H* variables. Similarly, this data set must contain at least *H* observations.

If you do not specify the REPS= *method-option*, the number of replicates is assumed to be the number of observations in the *SAS-data-set*. If you specify the number of replicates—for example, REPS=*nreps*—the first *nreps* observations in the *SAS-data-set* are used to construct the replicates.

You can specify the PRINTH *method-option* to display the Hadamard matrix that PROC SURVEYREG uses to construct replicates for BRR.

OUTWEIGHTS=*SAS-data-set*

names a *SAS-data-set* in which to store the replicate weights that PROC SURVEYREG creates for BRR variance estimation. For information about replicate weights, see the section "Balanced Repeated Replication (BRR) Method" on page 10359. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10367.

This *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement.

PRINTH

displays the Hadamard matrix that PROC SURVEYREG uses to construct replicates for BRR variance estimation. When you provide the Hadamard matrix in the HADAMARD= *method-option*, PROC SURVEYREG displays only the rows and columns that are actually used to construct replicates. For more information, see the sections "Balanced Repeated Replication (BRR) Method" on page 10359 and "Hadamard Matrix" on page 10363.

The PRINTH *method-option* is not available when you provide replicate weights in a REPWEIGHTS statement because the procedure does not use a Hadamard matrix in this case.

REPS=*number*

specifies the *number* of replicates for BRR variance estimation. The value of *number* must be an integer greater than 1.

If you do not use the HADAMARD= *method-option* to provide a Hadamard matrix, the number of replicates should be greater than the number of strata and should be a multiple of 4. For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10359. If PROC SURVEYREG cannot construct a Hadamard matrix for the REPS= value that you specify, the value is increased until a Hadamard matrix of that dimension can be constructed. Therefore, the actual number of replicates that PROC SURVEYREG uses might be larger than *number*.

If you use the HADAMARD= *method-option* to provide a Hadamard matrix, the value of *number* must not be greater than the number of rows in the Hadamard matrix. If you provide a Hadamard matrix and do not specify the REPS= *method-option*, the number of replicates is the number of rows in the Hadamard matrix.

If you do not specify the REPS= or the HADAMARD= *method-option* and do not use a REPWEIGHTS statement, the number of replicates is the smallest multiple of 4 that is greater than the number of strata.

If you use a REPWEIGHTS statement to provide replicate weights, PROC SURVEYREG does not use the REPS= *method-option*; the number of replicates is the number of REPWEIGHTS variables.

REPWARN < =NO | YES >
controls the log message that PROC SURVEYREG provides when the parameters cannot be estimated from one or more replicates because they are not estimable in those replicates. The procedure displays this log message as a warning if you specify REPWARN or REPWARN=YES. By default, the procedure displays the log message as a note, as if you specified REPWARN=NO.

For more information, see the section "Balanced Repeated Replication (BRR) Method" on page 10359.

JACKKNIFE < (*method-options*) >

JK < (*method-options*) >
requests variance estimation by the delete-1 jackknife method. For more information, see the section "Jackknife Method" on page 10361. If you use a REPWEIGHTS statement to provide replicate weights, VARMETHOD=JACKKNIFE is the default variance estimation method.

The delete-1 jackknife method requires at least two primary sampling units (PSUs) in each stratum for stratified designs unless you use a REPWEIGHTS statement to provide replicate weights.

You can specify the following *method-options*:

CENTER=FULLSAMPLE | REPLICATES
defines how to compute the deviations for the bootstrap method. You can specify the following values:

FULLSAMPLE computes the deviations of the replicate estimates from the full sample estimate.

REPLICATES computes the deviations of the replicate estimates from the average of the replicate estimates.

By default, CENTER=FULLSAMPLE. For more information, see the section "Jackknife Method" on page 10361.

OUTJKCOEFS=*SAS-data-set*
names a *SAS-data-set* in which to store the jackknife coefficients. For information about jackknife coefficients, see the section "Jackknife Method" on page 10361. For information about the contents of the OUTJKCOEFS= data set, see the section "Jackknife Coefficients Output Data Set" on page 10367.

OUTWEIGHTS=*SAS-data-set*
names a *SAS-data-set* in which to store the replicate weights that PROC SURVEYREG creates for jackknife variance estimation. For information about replicate weights, see the section "Jackknife Method" on page 10361. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10367.

This *method-option* is not available when you use a REPWEIGHTS statement to provide replicate weights.

REPWARN < =NO | YES >
controls the log message that PROC SURVEYREG provides when the parameters cannot be estimated from one or more replicates because they are not estimable in those replicates. The procedure displays this log message as a warning if you specify REPWARN or REPWARN=YES. By default, the procedure displays the log message as a note, as if you specified REPWARN=NO.

For more information, see the section "Jackknife Method" on page 10361.

TAYLOR
requests Taylor series variance estimation. This is the default method if you do not specify the VARMETHOD= option or a REPWEIGHTS statement. For more information, see the section "Taylor Series (Linearization)" on page 10356.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC SURVEYREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the SURVEYREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

Note that using a BY statement provides completely separate analyses of the BY groups. It does not provide a statistically valid domain (subpopulation) analysis, where the total number of units in the subpopulation is not known with certainty. You should use the DOMAIN statement to obtain domain analysis. For more information about subpopulation analysis for sample survey data, see Cochran (1977).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* < **(REF=** *option***)** > . . . < *variable* < **(REF=** *option***)** > > < / *global-options* > **;**

The CLASS statement names the classification variables to be used in the model. Typical classification variables are Treatment, Sex, Race, Group, and Replication. If you use the CLASS statement, it must appear before the MODEL statement.

Classification variables can be either character or numeric. By default, class levels are determined from the entire set of formatted values of the CLASS variables.

In any case, you can use formats to group values into levels. See the discussion of the FORMAT procedure in the *Base SAS Procedures Guide* and the discussions of the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*. You can adjust the order of CLASS variable levels with the ORDER= option in the PROC SURVEYREG statement.

You can specify the following REF= option to indicate how the levels of an individual classification variable are to be ordered by enclosing it in parentheses after the variable name:

REF=*'level'* **| FIRST | LAST**

specifies a level of the classification variable to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC SURVEYREG's singular parameterization. You can specify the *level* of the variable to use as the reference level; specify a value that corresponds to the formatted value of the variable if a format is assigned. Alternatively, you can specify REF=FIRST to designate that the first ordered level serve as the reference, or REF=LAST to designate that the last ordered level serve as the reference. To specify that REF=FIRST or REF=LAST be used for all classification variables, use the REF= *global-option* after the slash (/) in the CLASS statement.

You can specify the following *global-options* in the CLASS statement after a slash (/):

REF=FIRST | LAST

specifies a level of all classification variables to be put at the end of the list of levels. This level thus corresponds to the reference level in the usual interpretation of the estimates with PROC SURVEYREG's singular parameterization. Specify REF=FIRST to designate that the first ordered level for each classification variable serve as the reference. Specify REF=LAST to designate that the last ordered level serve as the reference. This option applies to all the variables specified in the CLASS statement. To specify different reference levels for different classification variables, use REF= options for individual variables.

CLUSTER Statement

CLUSTER *variables* **;**

The CLUSTER statement names variables that identify the clusters in a clustered sample design. The combinations of categories of CLUSTER variables define the clusters in the sample. If there is a STRATA statement, clusters are nested within strata.

If you provide replicate weights for BRR or jackknife variance estimation with the REPWEIGHTS statement, you do not need to specify a CLUSTER statement.

If your sample design has clustering at multiple stages, you should identify only the first-stage clusters (primary sampling units (PSUs)), in the CLUSTER statement. See the section "Primary Sampling Units (PSUs)" on page 10352 for more information.

The CLUSTER *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the CLUSTER variables determine the CLUSTER variable levels. Thus, you can use formats to group values into levels. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference* for more information.

When determining levels of a CLUSTER variable, an observation with missing values for this CLUSTER variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10350.

You can use multiple CLUSTER statements to specify cluster variables. The procedure uses variables from all CLUSTER statements to create clusters.

Prior to SAS 9, clusters were determined by using no more than the first 16 characters of the formatted values. If you want to revert to this previous behavior, you can use the TRUNCATE option in the PROC SURVEYREG statement.

CONTRAST Statement

CONTRAST *'label' effect values < / options >* **;**

CONTRAST *'label' effect values < . . . effect values > < / options >* **;**

The CONTRAST statement provides custom hypothesis tests for linear combinations of the regression parameters $H_0\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$, where $\mathbf{L}$ is the vector or matrix you specify and $\boldsymbol{\beta}$ is the vector of regression parameters. Thus, to use this feature, you must be familiar with the details of the model parameterization used by PROC SURVEYREG. For information about the parameterization, see the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

Each term in the MODEL statement, called an *effect*, is a variable or a combination of variables. You can specify an effect with a variable name or a special notation by using variable names and operators. For more details about how to specify an effect, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

For each CONTRAST statement, PROC SURVEYREG computes Wald's F test. The procedure displays this value with the degrees of freedom, and identifies it with the contrast label. The numerator degrees of freedom

for Wald's F test equal rank($\mathbf{L}$). The denominator degrees of freedom equal the number of clusters (or the number of observations if there is no CLUSTER statement) minus the number of strata. Alternatively, you can use the DF= option in the MODEL statement to specify the denominator degrees of freedom.

You can specify any number of CONTRAST statements, but they must appear after the MODEL statement.

In the CONTRAST statement,

label identifies the contrast in the output. A label is required for every contrast specified. Labels must be enclosed in single quotes.

effect identifies an effect that appears in the MODEL statement. You can use the INTERCEPT keyword as an effect when an intercept is fitted in the model. You do not need to include all effects that are in the MODEL statement.

values are constants that are elements of $\mathbf{L}$ associated with the effect.

You can specify the following *options* in the CONTRAST statement after a slash (/):

E

displays the entire coefficient $\mathbf{L}$ vector or matrix.

NOFILL

requests no filling in higher-order effects. When you specify only certain portions of $\mathbf{L}$, by default PROC SURVEYREG constructs the remaining elements from the context. (For more information, see the section "Specification of ESTIMATE Expressions" on page 4227 in Chapter 53, "The GLM Procedure.")

When you specify the NOFILL option, PROC SURVEYREG does not construct the remaining portions and treats the vector or matrix $\mathbf{L}$ as it is defined in the CONTRAST statement.

SINGULAR=*value*

tunes the estimability checking. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the largest absolute value of the elements of $\mathbf{v}$. For a row vector $\mathbf{l}$ of the matrix $\mathbf{L}$, define

$$c = \begin{cases} \text{ABS}(\mathbf{l}) & \text{if ABS}(\mathbf{l}) > 0 \\ 1 & \text{otherwise} \end{cases}$$

If ABS($\mathbf{l} - \mathbf{lH}$) is greater than c**value*, then $\mathbf{l}\boldsymbol{\beta}$ is declared nonestimable. Here, $\mathbf{H}$ is the matrix $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$. The *value* must be between 0 and 1; the default is 10^{-7}.

As stated previously, the CONTRAST statement enables you to perform hypothesis tests $H_0\colon \mathbf{L}\boldsymbol{\beta} = 0$.

If the $\mathbf{L}$ matrix contains more than one contrast, then you can separate the rows of the $\mathbf{L}$ matrix with commas.

For example, for the model

```
proc surveyreg;
   class A B;
   model Y=A B;
run;
```

with A at 5 levels and B at 2 levels, the parameter vector is

$$(\mu \;\; \alpha_1 \;\; \alpha_2 \;\; \alpha_3 \;\; \alpha_4 \;\; \alpha_5 \;\; \beta_1 \;\; \beta_2)$$

To test the hypothesis that the pooled A linear and A quadratic effect is zero, you can use the following **L** matrix:

$$\mathbf{L} = \begin{bmatrix} 0 & -2 & -1 & 0 & 1 & 2 & 0 & 0 \\ 0 & 2 & -1 & -2 & -1 & 2 & 0 & 0 \end{bmatrix}$$

The corresponding CONTRAST statement is

```
contrast 'A Linear & Quadratic'
         a -2 -1  0  1  2,
         a  2 -1 -2 -1  2;
```

DOMAIN Statement

DOMAIN *variables* < *variable*∗*variable variable*∗*variable*∗*variable . . .* > **;**

The DOMAIN statement requests analysis for domains (subpopulations) in addition to analysis for the entire study population. The DOMAIN statement names the *variables* that identify domains, which are called domain variables.

A domain variable can be either character or numeric. The procedure treats domain variables as categorical variables. If a *variable* appears by itself in a DOMAIN statement, each level of this variable determines a domain in the study population. If two or more *variables* are joined by asterisks (∗), then every possible combination of levels of these variables determines a domain. The procedure performs a descriptive analysis within each domain that is defined by the domain variables.

The formatted values of the domain variables determine the categorical variable levels. Thus, you can use formats to group values into levels. For more information, see the FORMAT procedure in *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

When the procedure is determining levels of a DOMAIN variable, an observation with missing values for this DOMAIN variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10350.

It is common practice to compute statistics for domains. Because formation of these domains might be unrelated to the sample design, the sample sizes for the domains are random variables. Use a DOMAIN statement to incorporate this variability into the variance estimation.

A DOMAIN statement is different from a BY statement. In a BY statement, you treat the sample sizes as fixed in each subpopulation, and you perform analysis within each BY group independently. For more information, see the section "Domain Analysis" on page 10365. Similarly, you should use a DOMAIN statement to perform a domain analysis over the entire data set. Creating a new data set from a single domain and analyzing that with PROC SURVEYREG yields inappropriate estimates of variance.

By default, the SURVEYREG procedure displays analyses for all levels of domains that are formed by the variables in a DOMAIN statement. Optionally, you can specify particular levels of each DOMAIN variable to be displayed by listing quoted *formatted-level-values* in parentheses after each variable name. You must enclose each *formatted-level-value* in single or double quotation marks. You can specify one or more levels of each variable; when you specify more than one level, separate the levels by a space or a comma. These examples illustrate the syntax:

```
domain Race*Gender(''Female'');
domain Race('White','Asian') Gender;
```

For example, **Race*Gender("Female")** requests that the procedure display analysis only for females within each race category, and **Race('White','Asian')** requests that the procedure display domain analysis only for people whose race is either white or Asian.

Specifying the same domain multiple times but with different levels for each corresponding domain variables is equivalent to specifying the union of different levels for the same variables. However, if you do not specify levels for a variable in a domain that is specified multiple times, only the specified levels are rendered. For example, the following two specifications together

```
domain Race('White')*Gender('Female');
domain Race('Asian')*Gender;
```

have the same effect as a single specification:

```
domain Race('White' 'Asian')*Gender('Female');
```

Also, the following specification

```
domain Race('White')*Gender Race('Asian')*Gender;
```

is equivalent to

```
domain Race('White' 'Asian')*Gender;
```

This syntax controls only the display of domain analysis results; it does not subset the data set, change the degrees of freedom, or otherwise affect the variance estimation.

EFFECT Statement

EFFECT *name*=*effect-type* (*variables* < / *options* >) ;

The EFFECT statement enables you to construct special collections of columns for design matrices. These collections are referred to as *constructed effects* to distinguish them from the usual model effects that are formed from continuous or classification variables, as discussed in the section "GLM Parameterization of Classification Variables and Effects" on page 412 in Chapter 20, "Shared Concepts and Topics."

You can specify the following *effect-types*:

COLLECTION specifies a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. The variables in a collection are considered as a unit for estimation and inference.

LAG specifies a classification effect in which the level that is used for a particular period corresponds to the level in the preceding period.

MULTIMEMBER | MM specifies a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement.

POLYNOMIAL | POLY specifies a multivariate polynomial effect in the specified numeric variables.

SPLINE specifies a regression spline effect whose columns are univariate spline expansions of one or more variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

Table 123.4 summarizes the *options* available in the EFFECT statement.

Table 123.4 EFFECT Statement Options

Option	Description
Collection Effects Options	
DETAILS	Displays the constituents of the collection effect
Lag Effects Options	
DESIGNROLE=	Names a variable that controls to which lag design an observation is assigned
DETAILS	Displays the lag design of the lag effect
NLAG=	Specifies the number of periods in the lag
PERIOD=	Names the variable that defines the period. This option is required.
WITHIN=	Names the variable or variables that define the group within which each period is defined. This option is required.
Multimember Effects Options	
NOEFFECT	Specifies that observations with all missing levels for the multimember variables should have zero values in the corresponding design matrix columns
WEIGHT=	Specifies the weight variable for the contributions of each of the classification effects
Polynomial Effects Options	
DEGREE=	Specifies the degree of the polynomial
MDEGREE=	Specifies the maximum degree of any variable in a term of the polynomial
STANDARDIZE=	Specifies centering and scaling suboptions for the variables that define the polynomial
Spline Effects Options	
BASIS=	Specifies the type of basis (B-spline basis or truncated power function basis) for the spline effect
DEGREE=	Specifies the degree of the spline effect
KNOTMETHOD=	Specifies how to construct the knots for the spline effect

For more information about the syntax of these *effect-types* and how columns of constructed effects are computed, see the section "EFFECT Statement" on page 427 in Chapter 20, "Shared Concepts and Topics."

ESTIMATE Statement

ESTIMATE <'*label*'> *estimate-specification* <(*divisor*=*n*)>
<, . . . <'*label*'> *estimate-specification* <(*divisor*=*n*)>>
</ *options*> ;

The ESTIMATE statement provides a mechanism for obtaining custom hypothesis tests. Estimates are formed as linear estimable functions of the form $\mathbf{L}\boldsymbol{\beta}$. You can perform hypothesis tests for the estimable functions, construct confidence limits, and obtain specific nonlinear transformations.

Table 123.5 summarizes the *options* available in the ESTIMATE statement.

Table 123.5 ESTIMATE Statement Options

Option	Description
Construction and Computation of Estimable Functions	
DIVISOR=	Specifies a list of values to divide the coefficients
NOFILL	Suppresses the automatic fill-in of coefficients for higher-order effects
SINGULAR=	Tunes the estimability checking difference
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple comparison adjustment of estimates
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiplicity-corrected p-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference
Statistical Output	
CL	Constructs confidence limits
CORR	Displays the correlation matrix of estimates
COV	Displays the covariance matrix of estimates
E	Prints the $\mathbf{L}$ matrix
JOINT	Produces a joint F or chi-square test for the estimable functions
PLOTS=	Produces ODS statistical graphics if the analysis is sampling-based
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays estimates
ILINK	Computes and displays estimates and standard errors on the inverse linked scale

For more information about the syntax of the ESTIMATE statement, see the section "ESTIMATE Statement"

on page 475 in Chapter 20, "Shared Concepts and Topics."

LSMEANS Statement

LSMEANS < *model-effects* > < / *options* > **;**

The LSMEANS statement computes and compares least squares means (LS-means) of fixed effects. LS-means are *predicted margins*—that is, they estimate the marginal means over a hypothetical balanced population.

Table 123.6 the summarizes available *options* in the LSMEANS statement.

Table 123.6 LSMEANS Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies the covariate value in computing LS-means
BYLEVEL	Computes separate margins
DIFF	Computes differences of LS-means
OM=	Specifies the weighting scheme for LS-means computation as determined by the input data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
Statistical Output	
ASC	Sorts LS-means in ascending order in results that you display by specifying the LINES option
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
LINES	Uses connecting lines to indicate nonsignificantly different subsets of LS-means
LINESTABLE	Displays the results of the LINES option as a table
MEANS	Prints the LS-means
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers

Table 123.6 *continued*

Option	Description
Generalized Linear Modeling	
EXP	Exponentiates and displays estimates of LS-means or LS-means differences
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale
ODDSRATIO	Reports (simple) differences of least squares means in terms of odds ratios if permitted by the link function

For details about the syntax of the LSMEANS statement, see the section "LSMEANS Statement" on page 492 in Chapter 20, "Shared Concepts and Topics."

LSMESTIMATE Statement

LSMESTIMATE *model-effect* < '*label*' > *values* < *divisor*=*n* >
< , . . . < '*label*' > *values* < *divisor*=*n* > >
< / *options* > ;

The LSMESTIMATE statement provides a mechanism for obtaining custom hypothesis tests among least squares means.

Table 123.7 summarizes the *options* available in the LSMESTIMATE statement.

Table 123.7 LSMESTIMATE Statement Options

Option	Description
Construction and Computation of LS-Means	
AT	Modifies covariate values in computing LS-means
BYLEVEL	Computes separate margins
DIVISOR=	Specifies a list of values to divide the coefficients
OM=	Specifies the weighting scheme for LS-means computation as determined by a data set
SINGULAR=	Tunes estimability checking
Degrees of Freedom and *p*-Values	
ADJUST=	Determines the method of multiple-comparison adjustment of LS-means differences
ALPHA=α	Determines the confidence level $(1-\alpha)$
LOWER	Performs one-sided, lower-tailed inference
STEPDOWN	Adjusts multiple-comparison *p*-values further in a step-down fashion
TESTVALUE=	Specifies values under the null hypothesis for tests
UPPER	Performs one-sided, upper-tailed inference

Table 123.7 *continued*

Option	Description
Statistical Output	
CL	Constructs confidence limits for means and mean differences
CORR	Displays the correlation matrix of LS-means
COV	Displays the covariance matrix of LS-means
E	Prints the **L** matrix
ELSM	Prints the **K** matrix
JOINT	Produces a joint *F* or chi-square test for the LS-means and LS-means differences
PLOTS=	Produces graphs of means and mean comparisons
SEED=	Specifies the seed for computations that depend on random numbers
Generalized Linear Modeling	
CATEGORY=	Specifies how to construct estimable functions for multinomial data
EXP	Exponentiates and displays LS-means estimates
ILINK	Computes and displays estimates and standard errors of LS-means (but not differences) on the inverse linked scale

For more information about the syntax of the LSMESTIMATE statement, see the section "LSMESTIMATE Statement" on page 513 in Chapter 20, "Shared Concepts and Topics."

MARGINS Statement

MARGINS *model-effects* < / *options* > **;**

The MARGINS statement computes and compares predictive margins of fixed effects. The predictive margin for a specific level (group) of a fixed effect represents the average predicted response if all the observations in the data set were in that group (Lane and Nelder 1982; Chang, Gelman, and Pagano 1982). You can compute predictive margins for any effect in the MODEL statement that involves only classification variables.

Table 123.8 summarizes the *options* available in the MARGINS statement.

Table 123.8 MARGINS Statement Options

Option	Description
Construction and Computation of Predictive Margins	
AT	Modifies the covariate value in computing predictive margins
DIFF	Computes differences of predictive margins
SLICEBY=	Partitions tests of interaction effects
SLICEDIFF	Computes differences of sliced predictive margins and determines the types of differences
Degrees of Freedom and *p*-Values	
ADJUST=	Specifies the method of multiple comparison adjustment of predictive margin differences
ALPHA=α	Specifies the confidence level $(1-\alpha)$
DF=	Assigns a specific value to degrees of freedom for t tests and confidence limits
STEPDOWN	Adjusts multiple comparison p-values further in a step-down fashion
Statistical Output	
CL	Constructs confidence limits for predictive margins and/or predictive margin differences

For more information about the syntax of the MARGINS statement, see the section "MARGINS Statement" on page 525 in Chapter 20, "Shared Concepts and Topics."

MODEL Statement

MODEL *dependent* **=** < *effects* > < / *options* > **;**

The MODEL statement specifies the dependent (response) variable and the independent (regressor) variables or effects. The dependent variable must be numeric. Each term in a MODEL statement, called an *effect*, is a variable or a combination of variables. You can specify an effect with a variable name or with special notation by using variable names and operators. For more information about how to specify an effect, see the section "Specification of Effects" on page 4207 in Chapter 53, "The GLM Procedure."

Only one MODEL statement is allowed for each PROC SURVEYREG statement. If you specify more than one MODEL statement, the procedure uses the first model and ignores the rest.

Table 123.9 summarizes the *options* available in the MODEL statement.

Table 123.9 MODEL Statement Options

Option	Description
ADJRSQ	Compute the adjusted multiple R-square
ANOVA	Produces the ANOVA table
CLPARM	Requests confidence limits

Table 123.9 *continued*

Option	Description
COVB	Displays the estimated covariance matrix
DEFF	Displays design effects
DF=	Specifies the denominator degrees of freedom
I	Displays the inverse or the generalized inverse of the $\mathbf{X}'\mathbf{X}$ matrix
NOINT	Omits the intercept
PARMLABEL	Displays the labels of the parameters
SINGULAR=	Tunes the estimability checking
SOLUTION	Displays parameter estimates
STB	Displays standardized parameter estimates
VADJUST=	Specifies whether to use degrees of freedom adjustment
X	Displays the $\mathbf{X}'\mathbf{X}$ matrix, or the $\mathbf{X}'\mathbf{W}\mathbf{X}$ matrix

You can specify the following *options* in the MODEL statement after a slash (/):

ADJRSQ
requests the procedure compute the adjusted multiple R-square.

ANOVA
requests the ANOVA table be produced in the output. By default, the ANOVA table is not printed in the output.

CLPARM
requests confidence limits for the parameter estimates. The SURVEYREG procedure determines the confidence coefficient by using the ALPHA= option, which by default equals 0.05 and produces 95% confidence bounds. The CLPARM option also requests confidence limits for all the estimable linear functions of regression parameters in the ESTIMATE statements.

Note that when there is a CLASS statement, you need to use the SOLUTION option with the CLPARM option to obtain the parameter estimates and their confidence limits.

COVB
displays the estimated covariance matrix of the estimated regression estimates.

DEFF
displays design effects for the regression coefficient estimates.

DF=*value*
specifies the denominator degrees of freedom for the F tests and the degrees of freedom for the t tests. For details about the default denominator degrees of freedom, see the section "Denominator Degrees of Freedom" on page 10363 for details.

I | INVERSE
displays the inverse or the generalized inverse of the $\mathbf{X}'\mathbf{X}$ matrix. When there is a WEIGHT variable, the procedure displays the inverse or the generalized inverse of the $\mathbf{X}'\mathbf{W}\mathbf{X}$ matrix, where $\mathbf{W}$ is the diagonal matrix constructed from WEIGHT variable values.

NOINT

omits the intercept from the model.

PARMLABEL

displays the labels of the parameters in the "Estimated Regression Coefficients" table, if the effect contains a single continuous variable that has a label.

SINGULAR=*value*

tunes the estimability checking. If $\mathbf{v}$ is a vector, define ABS($\mathbf{v}$) to be the largest absolute value of the elements of $\mathbf{v}$. For a row vector $\mathbf{l}$ of the matrix $\mathbf{L}$, define

$$c = \begin{cases} \text{ABS}(\mathbf{l}) & \text{if ABS}(\mathbf{l}) > 0 \\ 1 & \text{otherwise} \end{cases}$$

If ABS($\mathbf{l} - \mathbf{lH}$) is greater than c**value*, then $\mathbf{l}\boldsymbol{\beta}$ is declared nonestimable. Here, $\mathbf{H}$ is the matrix $(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{X}$. The *value* must be between 0 and 1; the default is 10^{-7}.

SOLUTION

displays a solution to the normal equations, which are the parameter estimates. The SOLUTION option is useful only when you use a CLASS statement. If you do not specify a CLASS statement, PROC SURVEYREG displays parameter estimates by default. But if you specify a CLASS statement, PROC SURVEYREG does not display parameter estimates unless you also specify the SOLUTION option.

STB

produces standardized regression coefficients. A standardized regression coefficient is computed by dividing a parameter estimate by the ratio of the sample standard deviation of the dependent variable to the sample standard deviation of the regressor.

VADJUST=DF | NONE

specifies whether to use degrees of freedom adjustment $(n-1)/(n-p)$ in the computation of the matrix $\mathbf{G}$ for the variance estimation. If you do not specify the VADJUST= option, by default, PROC SURVEYREG uses the degrees-of-freedom adjustment that is equivalent to the VARADJ=DF option. If you do not want to use this variance adjustment, you can specify the VADJUST=NONE option.

X | XPX

displays the $\mathbf{X}'\mathbf{X}$ matrix, or the $\mathbf{X}'\mathbf{W}\mathbf{X}$ matrix when there is a WEIGHT variable, where $\mathbf{W}$ is the diagonal matrix constructed from WEIGHT variable values. The X option also displays the crossproducts vector $\mathbf{X}'\mathbf{y}$ or $\mathbf{X}'\mathbf{W}\mathbf{y}$.

OUTPUT Statement

OUTPUT < **OUT=***SAS-data-set* > < *keyword*< **=***variable-name* > . . . *keyword*< **=***variable-name* >>
< / *option* > **;**

The OUTPUT statement creates a new SAS data set that contains all the variables in the input data set and, optionally, the estimated linear predictors and their standard error estimates, the residuals from the linear regression, and the confidence limits for the predictors.

You can specify the following *options* in the OUTPUT statement:

OUT=*SAS-data-set*
gives the name of the new output data set. By default, the procedure uses the DATA*n* convention to name the new data set.

keyword < *=variable-name* >
specifies the statistics to include in the output data set and names the new variables that contain the statistics. You can specify a *keyword* for each desired statistic (see the following list of *keywords*). Optionally, you can name a statistic by providing a variable name followed an equal sign to contain the statistic. For example,

```
output out=myOutDataSet p=myPredictor;
```

creates a SAS data set myOutDataSet that contains the predicted values in the variable myPredictor.

The *keywords* allowed and the statistics they represent are as follows:

LCLM | L lower bound of a $100(1-\alpha)\%$ confidence interval for the expected value (mean) of the predicted value. The α level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the ALPHA= option in the PROC SURVEYREG statement. If neither of these options is set, then $\alpha = 0.05$ by default, resulting in the lower bound for a 95% confidence interval. If no variable name is given for this keyword, the default variable name is _LCLM_.

PREDICTED | PRED | P predicted values. If no variable name is given for this keyword, the default variable name is _PREDICTED_.

RESIDUAL | R residuals, calculated as ACTUAL – PREDICTED. If no variable name is given for this keyword, the default variable name is _RESIDUAL_.

STDP | STD standard error of the mean predicted value. If no variable name is given for this keyword, the default variable name is _STD_.

UCLM | U upper bound of a $100(1-\alpha)\%$ confidence interval for the expected value (mean) of the predicted value. The α level is equal to the value of the ALPHA= option in the OUTPUT statement or, if this option is not specified, to the ALPHA= option in the PROC SURVEYREG statement. If neither of these options is set, then $\alpha = 0.05$ by default, resulting in the upper bound for a 95% confidence interval. If no variable name is given for this keyword, the default variable name is _UCLM_.

The following *option* is available in the OUTPUT statement and is specified after a slash (/):

ALPHA=α

specifies the level of significance α for $100(1-\alpha)\%$ confidence intervals. By default, α is equal to the value of the ALPHA= option in the PROC SURVEYREG statement or 0.05 if that option is not specified. You can use values between 0 and 1.

REPWEIGHTS Statement

REPWEIGHTS *variables* < / *options* > **;**

The REPWEIGHTS statement names *variables* that provide replicate weights for bootstrap, BRR, or jackknife variance estimation, which you request with the VARMETHOD=BOOTSTRAP, VARMETHOD=BRR, or VARMETHOD=JACKKNIFE option, respectively, in the PROC SURVEYREG statement. For more information about the replication methods, see the section "Variance Estimation" on page 10356.

Each REPWEIGHTS *variable* contains the weights for a single replicate, and the number of replicates equals the number of REPWEIGHTS variables. The REPWEIGHTS variables must be numeric, and the variable values must be nonnegative numbers.

For more information about replicate weights that the SURVEYREG procedure creates, see the sections "Balanced Repeated Replication (BRR) Method" on page 10359 and "Jackknife Method" on page 10361.

If you provide replicate weights with a REPWEIGHTS statement, you do not need to specify a CLUSTER or STRATA statement. If you use a REPWEIGHTS statement and do not specify the VARMETHOD= option in the PROC SURVEYREG statement, the procedure uses VARMETHOD=JACKKNIFE by default.

If you specify a REPWEIGHTS statement but do not include a WEIGHT statement, the procedure uses the average of replicate weights of each observation as the observation's weight.

You can specify the following *options* in the REPWEIGHTS statement after a slash (/):

DF=*df*

specifies the degrees of freedom for the analysis. The value of *df* must be a positive number. By default, the value of *df* is the number of REPWEIGHTS variables.

If you know the number of PSUs and the number of strata from which the replicate weights are generated, you should specify the number of PSUs minus the number of strata as the degrees of freedom in this option.

JKCOEFS=*value* | **<(>***values* **<)>** | *SAS-data-set*

specifies jackknife coefficients for the VARMETHOD=JACKKNIFE option in the PROC SURVEYREG statement. The jackknife coefficient values must be nonnegative numbers. For more information about jackknife coefficients, see the section "Jackknife Method" on page 10361.

You can provide jackknife coefficients by specifying one of the following forms:

value

specifies a single jackknife coefficient *value* to use for all replicates, where *value* must be a nonnegative number.

values
specifies a list of jackknife coefficients, where each value in the *values* is a nonnegative number that corresponds to a single replicate that is identified by a REPWEIGHTS variable. You can separate the values with blanks or commas, and you can optionally enclose the *values* in parentheses. The number of values in the *values* must equal the number of replicate weight variables that you specify in the REPWEIGHTS statement.

You must list the jackknife coefficient values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

SAS-data-set
names a *SAS-data-set* that contains the jackknife coefficients, where each coefficient value must be a nonnegative number. You must provide the jackknife coefficients in the data set variable named JKCoefficient. Each observation in this data set must correspond to a replicate that is identified by a REPWEIGHTS variable. The number of observations in the *SAS-data-set* must not be less than the number of REPWEIGHTS variables.

REPCOEFS=*value* | **<(>***values***<)>** | *SAS-data-set*
specifies replicate coefficients for the VARMETHOD=BOOTSTRAP or VARMETHOD=JACKKNIFE option in the PROC SURVEYREG statement, where each coefficient corresponds to an individual replicate weight that is identified by a REPWEIGHTS variable. The replicate coefficient values must be nonnegative numbers.

You can provide replicate coefficients by specifying one of the following forms:

value
specifies a single replicate coefficient *value* to use for all replicates, where *value* must be a nonnegative number.

values
specifies a list of replicate coefficients, where each value in the *values* is a nonnegative number that corresponds to a single replicate that is identified by a REPWEIGHTS variable. You can separate the values with blanks or commas, and you can optionally enclose the *values* in parentheses. The number of values in the *values* must equal the number of replicate weight variables that you specify in the REPWEIGHTS statement.

You must list the replicate coefficient values in the same order in which you list the corresponding replicate weight variables in the REPWEIGHTS statement.

SAS-data-set
names a *SAS-data-set* that contains the replicate coefficients, where each coefficient value must be a nonnegative number. You must provide the replicate coefficients in the data set variable named JKCoefficient. Each observation in this data set must correspond to a replicate that is identified by a REPWEIGHTS variable. The number of observations in the *SAS-data-set* must not be less than the number of REPWEIGHTS variables.

SLICE Statement

SLICE *model-effect* < / *options* > **;**

The SLICE statement provides a general mechanism for performing a partitioned analysis of the LS-means for an interaction. This analysis is also known as an analysis of simple effects.

This statement uses the same *options* as the LSMEANS statement, which are summarized in Table 20.24 in Chapter 20, "Shared Concepts and Topics." For more information about the syntax of the SLICE statement, see the section "SLICE Statement" on page 550 in Chapter 20, "Shared Concepts and Topics."

STORE Statement

STORE < **OUT=** >*item-store-name* < / **LABEL=**'*label*' > **;**

The STORE statement saves the context and results of the statistical analysis. The resulting item store has a binary file format that cannot be modified. The contents of the item store can be processed using the PLM procedure. For more information about the syntax of the STORE statement, see the section "STORE Statement" on page 554 in Chapter 20, "Shared Concepts and Topics."

STRATA Statement

STRATA *variables* < / *options* > **;**

The STRATA statement specifies variables that form the strata in a stratified sample design. The combinations of categories of STRATA variables define the strata in the sample.

If your sample design has stratification at multiple stages, you should identify only the first-stage strata in the STRATA statement. See the section "Specification of Population Totals and Sampling Rates" on page 10351 for more information.

If you provide replicate weights for BRR, jackknife, or bootstrap variance estimation by including the REPWEIGHTS statement, you do not need to specify a STRATA statement.

The STRATA *variables* are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the STRATA variables determine the levels. Thus, you can use formats to group values into levels. See the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference* for more information.

When determining levels of a STRATA variable, an observation with missing values for this STRATA variable is excluded, unless you specify the MISSING option. For more information, see the section "Missing Values" on page 10350.

You can use multiple STRATA statements to specify stratum variables.

You can specify the following *options* in the STRATA statement after a slash (/):

LIST

displays a "Stratum Information" table, which includes values of the STRATA variables and the number of observations, number of clusters, population total, and sampling rate for each stratum. See the section "Stratum Information" on page 10370 for more details.

NOCOLLAPSE

prevents the procedure from collapsing (combining) strata that have only one sampling unit for the Taylor series variance estimation. By default, the procedure collapses strata that contain only one sampling unit for the Taylor series method. See the section "Stratum Collapse" on page 10354 for details.

TEST Statement

TEST < *model-effects* > < / *options* > **;**

The TEST statement enables you to perform *F* tests for model effects that test Type I, Type II, or Type III hypotheses. For more information about constructing Type I, II, and III estimable functions, see Chapter 16, "The Four Types of Estimable Functions."

Table 123.10 summarizes the *options* available in the TEST statement.

Table 123.10 TEST Statement Options

Option	Description
CHISQ	Requests chi-square tests
DDF=	Specifies denominator degrees of freedom for fixed effects
E	Requests Type I, Type II, and Type III coefficients
E1	Requests Type I coefficients
E2	Requests Type II coefficients
E3	Requests Type III coefficients
HTYPE=	Indicates the type of hypothesis test to perform
INTERCEPT	Adds a row that corresponds to the overall intercept

For more information about the syntax of the TEST statement, see the section "TEST Statement" on page 556 in Chapter 20, "Shared Concepts and Topics."

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement names the variable that contains the sampling weights. This variable must be numeric, and the sampling weights must be positive numbers. If an observation has a weight that is nonpositive or missing, then the procedure omits that observation from the analysis. See the section "Missing Values" on page 10350 for more information. If you specify more than one WEIGHT statement, the procedure uses only the first WEIGHT statement and ignores the rest.

If you do not specify a WEIGHT statement but provide replicate weights with a REPWEIGHTS statement, PROC SURVEYREG uses the average of replicate weights of each observation as the observation's weight.

If you do not specify a WEIGHT statement or a REPWEIGHTS statement, PROC SURVEYREG assigns all observations a weight of one.

Details: SURVEYREG Procedure

Missing Values

If you have missing values in your survey data for any reason, such as nonresponse, this can compromise the quality of your survey results. If the respondents are different from the nonrespondents with regard to a survey effect or outcome, then survey estimates might be biased and cannot accurately represent the survey population. There are a variety of techniques in sample design and survey operations that can reduce nonresponse. After data collection is complete, you can use imputation to replace missing values with acceptable values, and/or you can use sampling weight adjustments to compensate for nonresponse. You should complete this data preparation and adjustment before you analyze your data with PROC SURVEYREG. For more information, see Cochran (1977); Kalton and Kasprzyk (1986); Brick and Kalton (1996).

If an observation has a missing value or a nonpositive value for the WEIGHT variable, then that observation is excluded from the analysis.

An observation is also excluded from the analysis if it has a missing value for any design (STRATA, CLUSTER, or DOMAIN) variable, unless you specify the MISSING option in the PROC SURVEYREG statement. If you specify the MISSING option, the procedure treats missing values as a valid (nonmissing) category for all categorical variables.

By default, if an observation contains missing values for the dependent variable or for any variable used in the independent effects, the observation is excluded from the analysis. This treatment is based on the assumption that the missing values are missing completely at random (MCAR). However, this assumption sometimes is not true. For example, evidence from other surveys might suggest that observations with missing values are systematically different from observations without missing values. If you believe that missing values are not missing completely at random, then you can specify the NOMCAR option to include these observations with missing values in the dependent variable and the independent variables in the variance estimation.

Whether or not you specify the NOMCAR option, the procedure always excludes observations with missing or invalid values for the WEIGHT, STRATA, CLUSTER, and DOMAIN variables, unless you specify the MISSING option.

When you specify the NOMCAR option, the procedure treats observations with and without missing values for variables in the regression model as two different domains, and it performs a domain analysis in the domain of nonmissing observations.

If you use a REPWEIGHTS statement, all REPWEIGHTS variables must contain nonmissing values.

Survey Design Information

Specification of Population Totals and Sampling Rates

To include a finite population correction (*fpc*) in Taylor series or bootstrap variance estimation, you can input either the sampling rate or the population total by using the RATE= or TOTAL= option, respectively, in the PROC SURVEYREG statement. (You cannot specify both of these options in the same PROC SURVEYREG statement.) The RATE= and TOTAL= options apply only to Taylor series or bootstrap variance estimation. The procedure does not use a finite population correction for BRR or jackknife variance estimation.

If you do not specify the RATE= or TOTAL= option, the Taylor series or bootstrap variance estimation does not include a finite population correction. For fairly small sampling fractions, it is appropriate to ignore this correction. For more information, see Cochran (1977) and Kish (1965).

If your design has multiple stages of selection and you are specifying the RATE= option, you should input the first-stage sampling rate, which is the ratio of the number of primary sampling units (PSUs) in the sample to the total number of PSUs in the study population. If you are specifying the TOTAL= option for a multistage design, you should input the total number of PSUs in the study population. For more information, see the section "Primary Sampling Units (PSUs)" on page 10352.

For a nonstratified sample design, or for a stratified sample design that has the same sampling rate or the same population total in all strata, you can use the RATE=*value* or TOTAL=*value* option. If your sample design is stratified with different sampling rates or population totals in different strata, use the RATE=*SAS-data-set* or TOTAL=*SAS-data-set* option to name a SAS data set that contains the stratum sampling rates or totals. This data set is called a *secondary data set*, as opposed to the *primary data set* that you specify in the DATA= option.

The secondary data set must contain all the stratification variables that are listed in the STRATA statement and all the variables that are listed in the BY statement. If there are formats associated with the STRATA variables and the BY variables, then the formats must be consistent in the primary and the secondary data sets. If you specify the TOTAL=*SAS-data-set* option, the secondary data set must have a variable named _TOTAL_ that contains the stratum population totals. Or if you specify the RATE=*SAS-data-set* option, the secondary data set must have a variable named _RATE_ that contains the stratum sampling rates. If the secondary data set contains more than one observation for any one stratum, then the procedure uses the first value of _TOTAL_ or _RATE_ for that stratum and ignores the rest.

The *value* in the RATE= option or the values of _RATE_ in the secondary data set must be nonnegative numbers. You can specify *value* as a number between 0 and 1. Or you can specify *value* in percentage form as a number between 1 and 100, and PROC SURVEYREG converts that number to a proportion. The procedure treats the value 1 as 100% instead of 1%.

If you specify the TOTAL=*value* option, *value* must not be less than the sample size. If you provide stratum population totals in a secondary data set, these values must not be less than the corresponding stratum sample sizes.

Primary Sampling Units (PSUs)

When you have clusters, or primary sampling units (PSUs), in your sample design, the procedure estimates variance from the variation among PSUs when the Taylor series variance method is used. For more information, see the section "Taylor Series (Linearization)" on page 10356.

BRR or jackknife variance estimation methods draw multiple replicates (also called subsamples) from the full sample by following a specific resampling scheme. These subsamples are constructed by deleting PSUs from the full sample.

The bootstrap variance estimation method repeatedly draws a simple random sample with replacement of primary sampling units (PSUs) from the full sample to form each replicate.

If you use a REPWEIGHTS statement to provide replicate weights for BRR, jackknife, bootstrap variance estimation, you do not need to specify a CLUSTER statement. Otherwise, you should specify a CLUSTER statement whenever your design includes clustering at the first stage of sampling. If you do not specify a CLUSTER statement, then PROC SURVEYREG treats each observation as a PSU.

Computational Details

Notation

For a stratified clustered sample design, observations are represented by an $n \times (p+2)$ matrix

$$(\mathbf{w}, \mathbf{y}, \mathbf{X}) = (w_{hij}, y_{hij}, \mathbf{x}_{hij})$$

where

- $\mathbf{w}$ denotes the sampling weight vector
- $\mathbf{y}$ denotes the dependent variable
- $\mathbf{X}$ denotes the $n \times p$ design matrix. (When an effect contains only classification variables, the columns of $\mathbf{X}$ that correspond this effect contain only 0s and 1s; no reparameterization is made.)
- $h = 1, 2, \ldots, H$ is the stratum index
- $i = 1, 2, \ldots, n_h$ is the cluster index within stratum h
- $j = 1, 2, \ldots, m_{hi}$ is the unit index within cluster i of stratum h
- p is the total number of parameters (including an intercept if the INTERCEPT effect is included in the MODEL statement)
- $n = \sum_{h=1}^{H} \sum_{i=1}^{n_h} m_{hi}$ is the total number of observations in the sample

Also, f_h denotes the sampling rate for stratum h. You can use the TOTAL= or RATE= option to input population totals or sampling rates. See the section "Specification of Population Totals and Sampling Rates" on page 10351 for details. If you input stratum totals, PROC SURVEYREG computes f_h as the ratio of the stratum sample size to the stratum total. If you input stratum sampling rates, PROC SURVEYREG uses these values directly for f_h. If you do not specify the TOTAL= or RATE= option, then the procedure assumes that the stratum sampling rates f_h are negligible, and a finite population correction is not used when computing variances.

Regression Coefficients

PROC SURVEYREG solves the normal equations $\mathbf{X}'\mathbf{W}\mathbf{X}\boldsymbol{\beta} = \mathbf{X}'\mathbf{W}\mathbf{y}$ by using a modified sweep routine that produces a generalized (g2) inverse $(\mathbf{X}'\mathbf{W}\mathbf{X})^{-}$ and a solution (Pringle and Rayner 1971)

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-}\mathbf{X}'\mathbf{W}\mathbf{y}$$

where $\mathbf{W}$ is the diagonal matrix constructed from WEIGHT variable values.

For models with CLASS variables, there are more design matrix columns than there are degrees of freedom (*df*) for the effect. Thus, there are linear dependencies among the columns. In this case, the parameters are not estimable; there is an infinite number of least squares solutions. PROC SURVEYREG uses a generalized (g2) inverse to obtain values for the estimates. The solution values are not displayed unless you specify the SOLUTION option in the MODEL statement. The solution has the characteristic that estimates are zero whenever the design column for that parameter is a linear combination of previous columns. (In strict terms, the solution values should not be called estimates.) With this full parameterization, hypothesis tests are constructed to test linear functions of the parameters that are estimable.

Design Effect

If you specify the DEFF option in the MODEL statement, PROC SURVEYREG calculates the design effects for the regression coefficients. The design effect of an estimate is the ratio of the actual variance to the variance computed under the assumption of simple random sampling:

$$\text{DEFF} = \frac{\text{variance under the sample design}}{\text{variance under simple random sampling}}$$

For more information, see Kish (1965, p. 258).

PROC SURVEYREG computes the numerator as described in the section "Variance Estimation" on page 10356. And the denominator is computed under the assumption that the sample design is simple random sampling, with no stratification and no clustering.

For Taylor series or bootstrap variance estimation, PROC SURVEYREG computes the overall sampling fraction f_{SRS} in the simple random sampling variance by using the value of the RATE= or TOTAL= option.

If you do not specify either of these options, PROC SURVEYREG assumes that the value of f_{SRS} is negligible and does not use a finite population correction in the analysis, as described in the section "Specification of Population Totals and Sampling Rates" on page 10351.

If you specify RATE=*value*, PROC SURVEYREG uses this *value* as the overall sampling fraction f_{SRS}. If you specify TOTAL=*value*, PROC SURVEYREG computes f_{SRS} as the ratio of the number of PSUs in the sample to the specified total.

If you specify stratum sampling rates by using the RATE=*SAS-data-set* option, then PROC SURVEYREG computes stratum totals based on these stratum sampling rates and the number of sample PSUs in each stratum. The procedure sums the stratum totals to form the overall total, and it computes f_{SRS} as the ratio of the number of sample PSUs to the overall total. Alternatively, if you specify stratum totals by using the TOTAL=*SAS-data-set* option, then PROC SURVEYREG sums these totals to compute the overall total. The overall sampling fraction f_{SRS} is then computed as the ratio of the number of sample PSUs to the overall total.

Stratum Collapse

If there is only one sampling unit in a stratum, then PROC SURVEYREG cannot estimate the variance for this stratum for the Taylor series method. To estimate stratum variances, by default the procedure collapses, or combines, those strata that contain only one sampling unit. If you specify the NOCOLLAPSE option in the STRATA statement, PROC SURVEYREG does not collapse strata and uses a variance estimate of zero for any stratum that contains only one sampling unit.

Note that stratum collapse only applies to Taylor series variance estimation (the default method, also specified by VARMETHOD=TAYLOR). The procedure does not collapse strata for replication methods.

If you do not specify the NOCOLLAPSE option for the Taylor series method, PROC SURVEYREG collapses strata according to the following rules. If there are multiple strata that contain only one sampling unit each, then the procedure collapses, or combines, all these strata into a new pooled stratum. If there is only one stratum with a single sampling unit, then PROC SURVEYREG collapses that stratum with the preceding stratum, where strata are ordered by the STRATA variable values. If the stratum with one sampling unit is the first stratum, then the procedure combines it with the following stratum.

If you specify stratum sampling rates by using the RATE=*SAS-data-set* option, PROC SURVEYREG computes the sampling rate for the new pooled stratum as the weighted average of the sampling rates for the collapsed strata. See the section "Computational Details" on page 10352 for details. If the specified sampling rate equals 0 for any of the collapsed strata, then the pooled stratum is assigned a sampling rate of 0. If you specify stratum totals by using the TOTAL=*SAS-data-set* option, PROC SURVEYREG combines the totals for the collapsed strata to compute the sampling rate for the new pooled stratum.

Sampling Rate of the Pooled Stratum from Collapse

Assuming that PROC SURVEYREG collapses single-unit strata $h_1, h_2, \ldots, h_c$ into the pooled stratum, the procedure calculates the sampling rate for the pooled stratum as

$$f_{\text{Pooled Stratum}} = \begin{cases} 0 & \text{if any of } f_{h_l} = 0 \text{ where } l = 1, 2, \ldots, c \\ \left(\sum_{l=1}^{c} n_{h_l} f_{h_l}^{-1} \right)^{-1} \sum_{l=1}^{c} n_{h_l} & \text{otherwise} \end{cases}$$

Analysis of Variance (ANOVA)

PROC SURVEYREG produces an analysis of variance table for the model specified in the MODEL statement. This table is identical to the one produced by the GLM procedure for the model. PROC SURVEYREG computes ANOVA table entries by using the sampling weights, but not the sample design information about stratification and clustering.

The degrees of freedom (*df*) displayed in the ANOVA table are the same as those in the ANOVA table produced by PROC GLM. The Total DF is the total degrees of freedom used to obtain the regression coefficient estimates. The Total DF equals the total number of observations minus 1 if the model includes an intercept. If the model does not include an intercept, the Total DF equals the total number of observations. The Model DF equals the degrees of freedom for the effects in the MODEL statement, not including the intercept. The Error DF equals the Total DF minus the Model DF.

Multiple R-Square

PROC SURVEYREG computes a multiple R-square for the weighted regression as

$$R^2 = 1 - \frac{\mathrm{SS}_{error}}{\mathrm{SS}_{total}}$$

where SS_{error} is the error sum of squares in the ANOVA table

$$\mathrm{SS}_{error} = \mathbf{r}'\mathbf{W}\mathbf{r}$$

and SS_{total} is the total sum of squares

$$\mathrm{SS}_{total} = \begin{cases} \mathbf{y}'\mathbf{W}\mathbf{y} & \text{if no intercept} \\ \mathbf{y}'\mathbf{W}\mathbf{y} - \left(\sum_{h=1}^{H} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij} y_{hij} \right)^2 / \, w_{...} & \text{otherwise} \end{cases}$$

where $w_{...}$ is the sum of the sampling weights over all observations.

Adjusted R-Square

If you specify the ADJRSQ option in the MODEL statement, PROC SURVEYREG computes an multiple R-square adjusted as the weighted regression as

$$\mathrm{ADJRSQ} = \begin{cases} 1 - \dfrac{n(1-R^2)}{n-p} & \text{if no intercept} \\ 1 - \dfrac{(n-1)(1-R^2)}{n-p} & \text{otherwise} \end{cases}$$

where R^2 is the multiple R-square.

Root Mean Square Errors

PROC SURVEYREG computes the square root of mean square errors as

$$\sqrt{\mathrm{MSE}} = \sqrt{n \; \mathrm{SS}_{error} \; / \; (n-p) \; w_{...}}$$

where $w_{...}$ is the sum of the sampling weights over all observations.

Variance Estimation

PROC SURVEYREG uses the Taylor series method or replication (resampling) methods to estimate sampling errors of estimators based on complex sample designs (Fuller 2009; Woodruff 1971; Fuller 1975; Fuller et al. 1989; Särndal, Swensson, and Wretman 1992; Wolter 2007; Rust 1985; Dippo, Fay, and Morganstein 1984; Rao and Shao 1999; Rao, Wu, and Yue 1992; Rao and Shao 1996). You can use the VARMETHOD= option to specify a variance estimation method to use. By default, the Taylor series method is used. However, replication methods have recently gained popularity for estimating variances in complex survey data analysis. One reason for this popularity is the relative simplicity of replication-based estimates, especially for nonlinear estimators; another is that modern computational capacity has made replication methods feasible for practical survey analysis.

Replication methods draw multiple replicates (also called subsamples) from a full sample according to a specific resampling scheme. The most commonly used resampling schemes are the *balanced repeated replication* (BRR) method, the *jackknife* method, and the *bootstrap* method. For each replicate, the original weights are modified for the PSUs in the replicates to create replicate weights. The parameters of interest are estimated by using the replicate weights for each replicate. Then the variances of parameters of interest are estimated by the variability among the estimates derived from these replicates. You can use the REPWEIGHTS statement to provide your own replicate weights for variance estimation.

The following sections provide details about how the variance-covariance matrix of the estimated regression coefficients is estimated for each variance estimation method.

Taylor Series (Linearization)

The Taylor series (linearization) method is the most commonly used method to estimate the covariance matrix of the regression coefficients for complex survey data. It is the default variance estimation method used by PROC SURVEYREG.

Use the notation described in the section "Notation" on page 10352 to denote the residuals from the linear regression as

$$\mathbf{r} = \mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}$$

with r_{hij} as its elements. Let the $p \times p$ matrix $\mathbf{G}$ be defined as

$$\mathbf{G} = \frac{n-1}{n-p} \sum_{h=1}^{H} \frac{n_h(1-f_h)}{n_h-1} \sum_{i=1}^{n_h} (\mathbf{e}_{hi\cdot} - \bar{\mathbf{e}}_{h\cdot\cdot})'(\mathbf{e}_{hi\cdot} - \bar{\mathbf{e}}_{h\cdot\cdot})$$

where

$$\begin{aligned}
\mathbf{e}_{hij} &= w_{hij} r_{hij} \mathbf{x}_{hij} \\
\mathbf{e}_{hi\cdot} &= \sum_{j=1}^{m_{hi}} \mathbf{e}_{hij} \\
\bar{\mathbf{e}}_{h\cdot\cdot} &= \frac{1}{n_h} \sum_{i=1}^{n_h} \mathbf{e}_{hi\cdot}
\end{aligned}$$

The Taylor series estimate of the covariance matrix of $\hat{\boldsymbol{\beta}}$ is

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-}\mathbf{G}(\mathbf{X}'\mathbf{W}\mathbf{X})^{-}$$

The factor $(n-1)/(n-p)$ in the computation of the matrix $\mathbf{G}$ reduces the small sample bias associated with using the estimated function to calculate deviations (Hidiroglou, Fuller, and Hickman 1980). For simple random sampling, this factor contributes to the degrees of freedom correction applied to the residual mean square for ordinary least squares in which p parameters are estimated. By default, the procedure use this adjustment in the variance estimation. If you do not want to use this multiplier in variance estimation, you can specify the VADJUST=NONE option in the MODEL statement to suppress this factor.

Bootstrap Method

The VARMETHOD=BOOTSTRAP option in the PROC SURVEYREG statement requests the bootstrap method for variance estimation. This method can be used for stratified sample designs and for designs that have no stratification. If your design is stratified, the bootstrap method requires at least two PSUs in each stratum. You can provide your own bootstrap replicate weights for the analysis by using a REPWEIGHTS statement, or the procedure can construct bootstrap replicate weights for the analysis.

PROC SURVEYREG estimates the parameter of interest (or requested statistics) from each replicate, and then uses the variability among replicate estimates to estimate the overall variance of these statistics.

This bootstrap method for complex survey data is similar to the method of Rao, Wu, and Yue (1992) and is also known as the bootstrap weights method (Mashreghi, Haziza, and Léger 2016). For more information, see Lohr (2022a, Section 9.3.3), Wolter (2007, Chapter 5), Beaumont and Patak (2012), Fuller (2009, Section 4.5), and Shao and Tu (1995, Section 6.2.4). McCarthy and Snowden (1985), Rao and Wu (1988), Sitter (1992b), and Sitter (1992a) provide several adjusted bootstrap variance estimators that are consistent for complex survey data. The naive bootstrap variance estimator that is suitable for infinite populations is not consistent for complex survey data.

Replicate Weight Construction

If you do not provide replicate weights by using a REPWEIGHTS statement, PROC SURVEYREG constructs bootstrap replicate weights for the analysis. The procedure selects replicate bootstrap samples by with-replacement random sampling of PSUs within strata. You can specify the number of bootstrap replicates in the REPS= *method-option*; by default, the number of replicates is 250. (Increasing the number of replicates can improve the estimation precision but also increases the computation time.) You can specify the bootstrap sample sizes m_h in the MH= *method-option*; by default, $m_h = n_h - 1$, where n_h is the number of PSUs in stratum h.

In each replicate sample, the original sampling weights of the selected units are adjusted to reflect the full sample. These adjusted weights are the *bootstrap replicate weights*. In replicate r, the bootstrap replicate weight for observation j in PSU i in stratum h is computed as

$$\tilde{w}_{hij}^{(r)} = \left\{ 1 + \sqrt{\frac{1-f_h}{m_h(n_h-1)}}\, n_h k_{hi}^{(r)} - \sqrt{\frac{m_h(1-f_h)}{n_h-1}} \right\} w_{hij}$$

where $k_{hi}^{(r)}$ is the number of times PSU i is selected for replicate r, and f_h is the sampling fraction in stratum h that you can input by using either the RATE= or TOTAL= option in the PROC SURVEYREG statement.

You can use the OUTWEIGHTS=*SAS-data-set* method-option to store the bootstrap replicate weights in a SAS data set. For information about the contents of the OUTWEIGHTS= data set, see the section "Replicate Weights Output Data Set" on page 10367. You can provide these replicate weights to the procedure for subsequent analyses by using a REPWEIGHTS statement.

Variance Estimation

Let R be the total number of bootstrap replicates weights. Denote α_r as the replicate coefficient for the rth replicate ($r = 1, 2, \ldots, R$).

When the procedure generates the bootstrap replicate weights, $\alpha_r = 1/R$.

If you provide your own bootstrap replicate weights by including a REPWEIGHTS statement, you can specify α_r in the REPCOEFS= option. By default, $\alpha_r = 1/R$.

Let $\hat{\boldsymbol{\beta}}$ be the estimated regression coefficients from the full sample for $\boldsymbol{\beta}$, and let $\hat{\boldsymbol{\beta}}_r$ be the estimated regression coefficient when the rth set of bootstrap replicate weights is used. PROC SURVEYREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by the following equation

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

Here, the degrees of freedom is the number of clusters minus the number of strata. If there are no clusters, then the degrees of freedom equals the number of observations minus the number of strata. If the design is not stratified, then the degrees of freedom equals the number of PSUs minus one.

If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYREG computes the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \bar{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \bar{\boldsymbol{\beta}}\right)'$$

where $\bar{\boldsymbol{\beta}}$ is the average of the replicate estimates and is calculated as follows:

$$\bar{\boldsymbol{\beta}} = \frac{1}{R}\sum_{r=1}^{R} \hat{\boldsymbol{\beta}}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom and if R' is less than the number of PSUs minus the number of strata, then the degrees of freedom is set to R'.

Balanced Repeated Replication (BRR) Method

The balanced repeated replication (BRR) method requires that the full sample be drawn by using a stratified sample design with two primary sampling units (PSUs) per stratum. Let H be the total number of strata. The total number of replicates R is the smallest multiple of 4 that is greater than H. However, if you prefer a larger number of replicates, you can specify the REPS=*number* option. If a *number* × *number* Hadamard matrix cannot be constructed, the number of replicates is increased until a Hadamard matrix becomes available.

Each replicate is obtained by deleting one PSU per stratum according to the corresponding Hadamard matrix and adjusting the original weights for the remaining PSUs. The new weights are called replicate weights.

Replicates are constructed by using the first H columns of the $R \times R$ Hadamard matrix. The rth ($r = 1, 2, \ldots, R$) replicate is drawn from the full sample according to the rth row of the Hadamard matrix as follows:

- If the (r, h) element of the Hadamard matrix is 1, then the first PSU of stratum h is included in the rth replicate and the second PSU of stratum h is excluded.
- If the (r, h) element of the Hadamard matrix is –1, then the second PSU of stratum h is included in the rth replicate and the first PSU of stratum h is excluded.

Note that the "first" and "second" PSUs are determined by data order in the input data set. Thus, if you reorder the data set and perform the same analysis by using BRR method, you might get slightly different results, because the contents in each replicate sample might change.

The replicate weights of the remaining PSUs in each half-sample are then doubled to their original weights. For more information about the BRR method, see Wolter (2007) and Lohr (2022a).

By default, an appropriate Hadamard matrix is generated automatically to create the replicates. You can request that the Hadamard matrix be displayed by specifying the VARMETHOD=BRR(PRINTH) *method-option*. If you provide a Hadamard matrix by specifying the VARMETHOD=BRR(HADAMARD=) *method-option*, then the replicates are generated according to the provided Hadamard matrix.

You can use the VARMETHOD=BRR(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

Let $\hat{\boldsymbol{\beta}}$ be the estimated regression coefficients from the full sample for $\boldsymbol{\beta}$, and let $\hat{\boldsymbol{\beta}}_r$ be the estimated regression coefficient from the rth replicate by using replicate weights. PROC SURVEYREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with H degrees of freedom, where H is the number of strata. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYREG computes the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\beta}}_r - \bar{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \bar{\boldsymbol{\beta}}\right)'$$

where $\bar{\boldsymbol{\beta}}$ is the average of the replicate estimates and is calculated as follows:

$$\bar{\boldsymbol{\beta}} = \frac{1}{R}\sum_{r=1}^{R}\hat{\boldsymbol{\beta}}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals the minimum between R' and the number of strata, H.

Fay's BRR Method

Fay's method is a modification of the BRR method, and it requires a stratified sample design with two primary sampling units (PSUs) per stratum. The total number of replicates R is the smallest multiple of 4 that is greater than the total number of strata H. However, if you prefer a larger number of replicates, you can specify the REPS= *method-option*.

For each replicate, Fay's method uses a Fay coefficient $0 \le \epsilon < 1$ to impose a perturbation of the original weights in the full sample that is gentler than using only half-samples, as in the traditional BRR method. The Fay coefficient $0 \le \epsilon < 1$ can be set by specifying the FAY = ϵ *method-option*. By default, $\epsilon = 0.5$ if the FAY *method-option* is specified without providing a value for ϵ (Judkins 1990; Rao and Shao 1999). When $\epsilon = 0$, Fay's method becomes the traditional BRR method. For more information, see Dippo, Fay, and Morganstein (1984); Fay (1984, 1989); Judkins (1990).

Let H be the number of strata. Replicates are constructed by using the first H columns of the $R \times R$ Hadamard matrix, where R is the number of replicates, $R > H$. The rth ($r = 1, 2, \ldots, R$) replicate is created from the full sample according to the rth row of the Hadamard matrix as follows:

- If the (r, h) element of the Hadamard matrix is 1, then the full sample weight of the first PSU in stratum h is multiplied by ϵ and the full sample weight of the second PSU is multiplied by $2 - \epsilon$ to obtain the rth replicate weights.
- If the (r, h) element of the Hadamard matrix is –1, then the full sample weight of the first PSU in stratum h is multiplied by $2 - \epsilon$ and the full sample weight of the second PSU is multiplied by ϵ to obtain the rth replicate weights.

You can use the VARMETHOD=BRR(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

By default, an appropriate Hadamard matrix is generated automatically to create the replicates. You can request that the Hadamard matrix be displayed by specifying the VARMETHOD=BRR(PRINTH) *method-option*. If you provide a Hadamard matrix by specifying the VARMETHOD=BRR(HADAMARD=) *method-option*, then the replicates are generated according to the provided Hadamard matrix.

Let $\hat{\boldsymbol{\beta}}$ be the estimated regression coefficients from the full sample for $\boldsymbol{\beta}$. Let $\hat{\boldsymbol{\beta}}_r$ be the estimated regression coefficient obtained from the rth replicate by using replicate weights. PROC SURVEYREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R(1-\epsilon)^2}\sum_{r=1}^{R}\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with H degrees of freedom, where H is the number of strata. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates R.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYREG computes the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \frac{1}{R(1-\epsilon)^2} \sum_{r=1}^{R} \left(\hat{\boldsymbol{\beta}}_r - \bar{\boldsymbol{\beta}}\right) \left(\hat{\boldsymbol{\beta}}_r - \bar{\boldsymbol{\beta}}\right)'$$

where $\bar{\boldsymbol{\beta}}$ is the average of the replicate estimates and is calculated as follows:

$$\bar{\boldsymbol{\beta}} = \frac{1}{R} \sum_{r=1}^{R} \hat{\boldsymbol{\beta}}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, R, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals the minimum between R' and the number of strata, H.

Jackknife Method

The jackknife method of variance estimation deletes one PSU at a time from the full sample to create replicates. The total number of replicates R is the same as the total number of PSUs. In each replicate, the sample weights of the remaining PSUs are modified by the jackknife coefficient α_r. The modified weights are called replicate weights.

The jackknife coefficient and replicate weights are described as follows.

Without Stratification If there is no stratification in the sample design (no STRATA statement), the jackknife coefficients α_r are the same for all replicates:

$$\alpha_r = \frac{R-1}{R} \quad \text{where } r = 1, 2, \ldots, R$$

Denote the original weight in the full sample for the jth member of the ith PSU as w_{ij}. If the ith PSU is included in the rth replicate ($r = 1, 2, \ldots, R$), then the corresponding replicate weight for the jth member of the ith PSU is defined as

$$w_{ij}^{(r)} = w_{ij}/\alpha_r$$

With Stratification If the sample design involves stratification, each stratum must have at least two PSUs to use the jackknife method.

Let stratum $\tilde{h}_r$ be the stratum from which a PSU is deleted for the rth replicate. Stratum $\tilde{h}_r$ is called the *donor stratum.* Let $n_{\tilde{h}_r}$ be the total number of PSUs in the donor stratum $\tilde{h}_r$. The jackknife coefficients are defined as

$$\alpha_r = \frac{n_{\tilde{h}_r} - 1}{n_{\tilde{h}_r}} \quad \text{where } r = 1, 2, \ldots, R$$

Denote the original weight in the full sample for the *j*th member of the *i*th PSU as w_{ij}. If the *i*th PSU is included in the *r*th replicate ($r = 1, 2, \ldots, R$), then the corresponding replicate weight for the *j*th member of the *i*th PSU is defined as

$$w_{ij}^{(r)} = \begin{cases} w_{ij} & \text{if } i\text{th PSU is not in the donor stratum } \tilde{h}_r \\ w_{ij}/\alpha_r & \text{if } i\text{th PSU is in the donor stratum } \tilde{h}_r \end{cases}$$

You can use the VARMETHOD=JACKKNIFE(OUTJKCOEFS=) *method-option* to save the jackknife coefficients into a SAS data set and use the VARMETHOD=JACKKNIFE(OUTWEIGHTS=) *method-option* to save the replicate weights into a SAS data set.

If you provide your own replicate weights in a REPWEIGHTS statement, then you can also provide corresponding jackknife coefficients in the JKCOEFS= or REPCOEFS= option. If you provide replicate weights but do not provide jackknife coefficients, PROC SURVEYREG uses $\alpha_r = (R-1)/R$ as the jackknife coefficient for all replicates by default.

Let $\hat{\boldsymbol{\beta}}$ be the estimated regression coefficients from the full sample for $\boldsymbol{\beta}$. Let $\hat{\boldsymbol{\beta}}_r$ be the estimated regression coefficient obtained from the *r*th replicate by using replicate weights. PROC SURVEYREG estimates the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \hat{\boldsymbol{\beta}}\right)'$$

with *R*–*H* degrees of freedom, where *R* is the number of replicates and *H* is the number of strata, or *R*–1 when there is no stratification. If you provide your own replicate weights without specifying the DF= option, the degrees of freedom is set to be the number of replicates *R*.

If you specify the CENTER=REPLICATES *method-option*, then PROC SURVEYREG computes the covariance matrix of $\hat{\boldsymbol{\beta}}$ by

$$\widehat{\mathbf{V}}(\hat{\boldsymbol{\beta}}) = \sum_{r=1}^{R} \alpha_r \left(\hat{\boldsymbol{\beta}}_r - \bar{\boldsymbol{\beta}}\right)\left(\hat{\boldsymbol{\beta}}_r - \bar{\boldsymbol{\beta}}\right)'$$

where $\bar{\boldsymbol{\beta}}$ is the average of the replicate estimates and is calculated as follows:

$$\bar{\boldsymbol{\beta}} = \frac{1}{R}\sum_{r=1}^{R} \hat{\boldsymbol{\beta}}_r$$

If a parameter cannot be computed from one or more replicates, then the procedure computes the variance estimate by using those replicates from which the parameter can be estimated, and the number of those replicates, R', replaces the original number of replicates, *R*, in the variance estimation.

If you do not provide your own value for the degrees of freedom, then the degrees of freedom equals $R' - H$.

Hadamard Matrix

A Hadamard matrix **H** is a square matrix whose elements are either 1 or –1 such that

$$\mathbf{H}\mathbf{H}' = k\mathbf{I}$$

where k is the dimension of **H** and **I** is the identity matrix of order k. The order k is necessarily 1, 2, or a positive integer that is a multiple of 4.

For example, the following matrix is a Hadamard matrix of dimension $k = 8$:

$$\begin{array}{rrrrrrrr}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\
1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\
1 & -1 & -1 & 1 & -1 & 1 & 1 & -1
\end{array}$$

Degrees of Freedom

PROC SURVEYREG produces tests for the significance of model effects, regression parameters, estimable functions specified in the ESTIMATE statement, and contrasts specified in the CONTRAST statement. It computes all these tests taking into account the sample design. The degrees of freedom for these tests differ from the degrees of freedom for the ANOVA table, which does not consider the sample design.

Denominator Degrees of Freedom

The denominator *df* refers to the denominator degrees of freedom for F tests and to the degrees of freedom for t tests in the analysis.

For the Taylor series method, the denominator *df* equals the number of clusters minus the actual number of strata. If there are no clusters, the denominator *df* equals the number of observations minus the actual number of strata. The *actual number of strata* equals the following:

- one, if there is no STRATA statement
- the number of strata in the sample, if there is a STRATA statement but the procedure does not collapse any strata
- the number of strata in the sample after collapsing, if there is a STRATA statement and the procedure collapses strata that have only one sampling unit

For replication variance estimation methods that include a REPWEIGHTS statement, the denominator *df* equals the number of REPWEIGHTS variables, unless you specify an alternative in the DF= option in the REPWEIGHTS statement.

When the BRR method (including Fay's method) is specified but no REPWEIGHTS statement is included, the denominator *df* equals the number of strata.

When the jackknife method or bootstrap method is specified but no REPWEIGHTS statement is included, the denominator *df* is the same as the Taylor series method described earlier.

Alternatively, you can specify your own denominator *df* by using the DF= option in the MODEL statement.

Numerator Degrees of Freedom

The numerator *df* refers to the numerator degrees of freedom for the Wald *F* statistic associated with an effect or with a contrast. The procedure computes the Wald *F* statistic for an effect as a Type III test; that is, the test has the following properties:

- The hypothesis for an effect does not involve parameters of other effects except for containing effects (which it must involve to be estimable).
- The hypotheses to be tested are invariant to the ordering of effects in the model.

See the section "Testing Effects" on page 10364 for more information. The numerator *df* for the Wald *F* statistic for a contrast is the rank of the **L** matrix that defines the contrast.

Testing

Testing Effects

For each effect in the model, PROC SURVEYREG computes an **L** matrix such that every element of $\mathbf{L}\boldsymbol{\beta}$ is estimable; the **L** matrix has the maximum possible rank that is associated with the effect. To test the effect, the procedure uses the Wald *F* statistic for the hypothesis $H_0\colon \mathbf{L}\boldsymbol{\beta} = 0$. The Wald *F* statistic equals

$$F_{\text{Wald}} = \frac{(\mathbf{L}\hat{\boldsymbol{\beta}})'(\mathbf{L}\widehat{\mathbf{V}}\mathbf{L}')^{-}(\mathbf{L}\hat{\boldsymbol{\beta}})}{\text{rank}(\mathbf{L}\widehat{\mathbf{V}}\mathbf{L}')}$$

with numerator degrees of freedom equal to $\text{rank}(\mathbf{L}\widehat{\mathbf{V}}\mathbf{L}')$.

In the Taylor series method, the denominator degrees of freedom is equal to the number of clusters minus the number of strata (unless you specify the denominator degrees of freedom with the DF= option in the MODEL statement). For details about denominator degrees of freedom in replication methods, see the section "Denominator Degrees of Freedom" on page 10363. It is possible that the **L** matrix cannot be constructed for an effect, in which case that effect is not testable. For more information about how the matrix **L** is constructed, see the discussion in Chapter 16, "The Four Types of Estimable Functions."

You can use the TEST statement to perform *F* tests that test Type I, Type II, or Type III hypotheses. For details about the syntax of the TEST statement, see the section "TEST Statement" on page 556 in Chapter 20, "Shared Concepts and Topics."

Contrasts

You can use the CONTRAST statement to perform custom hypothesis tests. If the hypothesis is testable in the univariate case, the Wald *F* statistic for $H_0 : \mathbf{L}\boldsymbol{\beta} = 0$ is computed as

$$F_{\text{Wald}} = \frac{(\mathbf{L}_{\text{Full}}\hat{\boldsymbol{\beta}})'(\mathbf{L}_{\text{Full}}\widehat{\mathbf{V}}\mathbf{L}'_{\text{Full}})^{-}(\mathbf{L}_{\text{Full}}\hat{\boldsymbol{\beta}})}{\text{rank}(\mathbf{L})}$$

where **L** is the contrast vector or matrix you specify, $\boldsymbol{\beta}$ is the vector of regression parameters, $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-}\mathbf{X}'\mathbf{W}\mathbf{Y}$, $\widehat{\mathbf{V}}$ is the estimated covariance matrix of $\hat{\boldsymbol{\beta}}$, rank(**L**) is the rank of **L**, and $\mathbf{L}_{\text{Full}}$ is a matrix such that

- $\mathbf{L}_{\text{Full}}$ has the same number of columns as $\mathbf{L}$
- $\mathbf{L}_{\text{Full}}$ has full row rank
- the rank of $\mathbf{L}_{\text{Full}}$ equals the rank of the $\mathbf{L}$ matrix
- all rows of $\mathbf{L}_{\text{Full}}$ are estimable functions
- the Wald F statistic computed using the $\mathbf{L}_{\text{Full}}$ matrix is equivalent to the Wald F statistic computed by using the $\mathbf{L}$ matrix with any row deleted that is a linear combination of previous rows

If $\mathbf{L}$ is a full-rank matrix and all rows of $\mathbf{L}$ are estimable functions, then $\mathbf{L}_{\text{Full}}$ is the same as $\mathbf{L}$. It is possible that $\mathbf{L}_{\text{Full}}$ matrix cannot be constructed for contrasts in a CONTRAST statement, in which case the contrasts are not testable.

A Note on F_{Wald}

The preceding tests that use the F_{Wald} statistic assume that the estimated variance of $\hat{\boldsymbol{\beta}}$, $\widehat{\mathbf{V}}$, is of the form $\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}$ for some estimate $\mathbf{V}$ of the variance of $\mathbf{Y}$. In this case, estimability, $\mathbf{L}(\mathbf{X}'\mathbf{X})^{-}(\mathbf{X}'\mathbf{X}) = \mathbf{L}$, ensures that this F_{Wald} statistic has a unique value no matter which kind of generalized inverse is used to compute it. However, when a design-based variance estimator is used to estimate the variability of $\hat{\boldsymbol{\beta}}$, estimability does not ensure uniqueness. In this case, the F_{Wald} value is invariant to the choice of the generalized inverse if and only if $\mathbf{L}$ is estimable and $\mathbf{L}'\left[\mathbf{L}\widehat{\mathbf{V}}\mathbf{L}'\right]^{-}\left[\mathbf{L}\widehat{\mathbf{V}}\mathbf{L}'\right] = \mathbf{L}'$.

Although it is extremely rare, it is possible in practice that the preceding uniqueness condition is not satisfied. For example, if the number of PSUs is less than the number of nonsingular parameters in the model, then the matrix of coefficients for testing the overall null does not satisfy the uniqueness condition. If this condition is not satisfied, then the F_{Wald} statistic for testing $H\colon \mathbf{L}\boldsymbol{\beta} = \mathbf{0}$ is not invariant to the choice of the g_2-inverse of $\mathbf{L}\widehat{\mathbf{V}}\mathbf{L}'$. In practical applications, the test statistic is compared with an F distribution, but the value of the test statistic and therefore the inference might be different when a different g_2-inverse is used. Thus, this F test is not recommended when the uniqueness condition is not satisfied. An alternative approach would be to increase the number of PSUs or to find a parsimonious model so that the number of parameters is less than the number of PSUs.

Domain Analysis

A DOMAIN statement requests that the procedure perform regression analysis for each domain.

For a domain D, let I_D be the corresponding indicator variable:

$$I_D(h,i,j) = \begin{cases} 1 & \text{if observation } (h,i,j) \text{belongs to domain } D \\ 0 & \text{otherwise} \end{cases}$$

Let

$$v_{hij} = w_{hij} I_D(h,i,j) = \begin{cases} w_{hij} & \text{if observation } (h,i,j) \text{ belongs to domain } D \\ 0 & \text{otherwise} \end{cases}$$

The regression in domain D uses v as the weight variable.

Computational Resources

Due to the complex nature of survey data analysis, the SURVEYREG procedure requires more memory than an analysis of the same regression model by the GLM procedure. For details about the amount of memory related to the modeling, see the section "Computational Resources" on page 4261 in Chapter 53, "The GLM Procedure."

The memory needed by the SURVEYREG procedure to handle the survey design is described as follows.

Let

- H be the total number of strata
- n_c be the total number of clusters in your sample across all H strata, if you specify a CLUSTER statement
- p be the total number of parameters in the model

The memory needed (in bytes) is

$$48H + 8pH + 4p(p+1)H$$

For a cluster sample, the additional memory needed (in bytes) is

$$48H + 8pH + 4p(p+1)H + 4p(p+1)n_c + 16n_c$$

The SURVEYREG procedure also uses other small amounts of additional memory. However, when you have a large number of clusters or strata, or a large number of parameters in your model, the memory described previously dominates the total memory required by the procedure.

Output Data Sets

You can use the Output Delivery System (ODS) to create a SAS data set from any piece of PROC SURVEYREG output. See the section "ODS Table Names" on page 10372 for more information. For a more detailed description of using ODS, see Chapter 23, "Using the Output Delivery System."

PROC SURVEYREG also provides an OUTPUT statement to create a data set that contains estimated linear predictors and their standard error estimates, the residuals from the linear regression, and the confidence limits for the predictors.

If you use replication variance estimation, PROC SURVEYREG provides an output data set that stores the replicate weights and an output data set that stores the jackknife coefficients for jackknife variance estimation.

OUT= Data Set Created by the OUTPUT Statement

The OUTPUT statement produces an output data set that contains the following:

- all original data from the SAS data set input to PROC SURVEYREG
- the new variables corresponding to the diagnostic measures specified with statistics *keywords* in the OUTPUT statement (PREDICTED=, RESIDUAL=, and so on)

When any independent variable in the analysis (including all classification variables) is missing for an observation, then all new variables that correspond to diagnostic measures are missing for the observation in the output data set.

When a dependent variable in the analysis is missing for an observation, then the residual variable that corresponds to R is also missing in the output data set. However, the variables corresponding to LCLM, P, STDP, and UCLM are not missing.

Replicate Weights Output Data Set

If you specify the OUTWEIGHTS= *method-option* for VARMETHOD=BRR, VARMETHOD=JACKKNIFE, or VARMETHOD=BOOTSTRAP, PROC SURVEYREG stores the replicate weights in an output data set. The OUTWEIGHTS= output data set contains all observations from the DATA= input data set that are valid (used in the analysis). (A valid observation is an observation that has a positive value of the WEIGHT variable. Valid observations must also have nonmissing values of the STRATA and CLUSTER variables, unless you specify the MISSING option.)

The OUTWEIGHTS= data set contains all variables in the DATA= input data set and the replicate weight variables, RepWt_1, RepWt_2, . . ., RepWt_n, where n is the total number of replicates in the analysis. Each replicate weight variable contains the replicate weights for the corresponding replicate. Replicate weights equal 0 for observations that are not included in the replicate.

After the procedure creates replicate weights for a particular input data set and survey design, you can use the OUTWEIGHTS= *method-option* to store these replicate weights and then use them again in subsequent analyses, either in PROC SURVEYREG or in other survey procedures. You can use the REPWEIGHTS statement to provide replicate weights for the procedure.

Jackknife Coefficients Output Data Set

If you specify the OUTJKCOEFS= *method-option* for VARMETHOD=JACKKNIFE, PROC SURVEYREG stores the jackknife coefficients in an output data set. The OUTJKCOEFS= output data set contains one observation for each replicate. The OUTJKCOEFS= data set contains the following variables:

- Replicate, which is the replicate number for the jackknife coefficient
- JKCoefficient, which is the jackknife coefficient
- DonorStratum, which is the stratum of the PSU that was deleted to construct the replicate, if you specify a STRATA statement

After the procedure creates jackknife coefficients for a particular input data set and survey design, you can use the OUTJKCOEFS= *method-option* to store these coefficients and then use them again in subsequent analyses, either in PROC SURVEYREG or in the other survey procedures. You can use the JKCOEFS= option in the REPWEIGHTS statement to provide jackknife coefficients for the procedure.

Displayed Output

The SURVEYREG procedure produces output that is described in the following sections.

Output that is generated by the EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements is not listed below. For information about the output that is generated by these statements, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Data Summary

By default, PROC SURVEYREG displays the following information in the "Data Summary" table:

- Number of Observations, which is the total number of observations used in the analysis, excluding observations with missing values
- Sum of Weights, if you specify a WEIGHT statement
- Mean of the dependent variable in the MODEL statement, or Weighted Mean if you specify a WEIGHT statement
- Sum of the dependent variable in the MODEL statement, or Weighted Sum if you specify a WEIGHT statement

Design Summary

When you specify a CLUSTER statement or a STRATA statement, the procedure displays a "Design Summary" table, which provides the following sample design information:

- Number of Strata, if you specify a STRATA statement
- Number of Strata Collapsed, if the procedure collapses strata
- Number of Clusters, if you specify a CLUSTER statement
- Overall Sampling Rate used to calculate the design effect, if you specify the DEFF option in the MODEL statement

Domain Summary

By default, PROC SURVEYREG displays the following information in the "Domain Summary" table:

- Number of Observations, which is the total number of observations used in the analysis
- total number of observations in the current domain
- total number of observations not in the current domain
- Sum of Weights for the observations in the current domain, if you specify a WEIGHT statement

Fit Statistics

By default, PROC SURVEYREG displays the following regression statistics in the "Fit Statistics" table:

- R-square for the regression
- Root MSE, which is the square root of the mean square error
- Denominator DF, which is the denominator degrees of freedom for the F tests and also the degrees of freedom for the t tests produced by the procedure

Variance Estimation

If the variance method is not Taylor series (see the section "Variance Estimation" on page 10356) or if the NOMCAR option is used, by default, PROC SURVEYREG displays the following variance estimation information in the "Variance Estimation" table:

- Method, which is the variance estimation method
- Number of Replicates
- Initial seed for the random number that the bootstrap method uses
- Hadamard Data Set name, if you specify the VARMETHOD=BRR(HADAMARD=) *method-option*
- Fay Coefficient, if you specify the VARMETHOD=BRR(FAY) *method-option*
- Replicate Weights input data set name, if you provide replicate weights with a REPWEIGHTS statement
- Missing Levels, which indicates whether missing levels of categorical variables are included by the MISSING option
- Missing Values, which indicates whether observations with missing values are included in the analysis by the NOMCAR option

Stratum Information

When you specify the LIST option in the STRATA statement, PROC SURVEYREG displays a "Stratum Information" table, which provides the following information for each stratum:

- Stratum Index, which is a sequential stratum identification number
- STRATA variable(s), which lists the levels of STRATA variables for the stratum
- Population Total, if you specify the TOTAL= option
- Sampling Rate, if you specify the TOTAL= option or the RATE= option. If you specify the TOTAL= option, the sampling rate is based on the number of nonmissing observations in the stratum.
- N Obs, which is the number of observations
- number of Clusters, if you specify a CLUSTER statement
- Collapsed, which has the value 'Yes' if the stratum is collapsed with another stratum before analysis

If PROC SURVEYREG collapses strata, the "Stratum Information" table also displays stratum information for the new, collapsed stratum. The new stratum has a Stratum Index of 0 and is labeled 'Pooled.'

Class Level Information

If you use a CLASS statement to name classification variables, PROC SURVEYREG displays a "Class Level Information" table. This table contains the following information for each classification variable:

- CLASS Variable, which lists each CLASS variable name
- Levels, which is the number of values or levels of the classification variable
- Values, which lists the values of the classification variable. The values are separated by a white space character; therefore, to avoid confusion, you should not include a white space character within a classification variable value.

X'X Matrix

If you specify the XPX option in the MODEL statement, PROC SURVEYREG displays the $\mathbf{X}'\mathbf{X}$ matrix. When there is a WEIGHT variable, the procedure displays the $\mathbf{X}'\mathbf{W}\mathbf{X}$ matrix. This option also displays the crossproducts vector $\mathbf{X}'\mathbf{y}$ or $\mathbf{X}'\mathbf{W}\mathbf{y}$, where $\mathbf{y}$ is the response vector (dependent variable).

Inverse Matrix of X'X

If you specify the INVERSE option in the MODEL statement, PROC SURVEYREG displays the inverse or the generalized inverse of the $\mathbf{X}'\mathbf{X}$ matrix. When there is a WEIGHT variable, the procedure displays the inverse or the generalized inverse of the $\mathbf{X}'\mathbf{W}\mathbf{X}$ matrix.

ANOVA for Dependent Variable

If you specify the ANOVA option in the model statement, PROC SURVEYREG displays an analysis of variance table for the dependent variable. This table is identical to the ANOVA table displayed by the GLM procedure.

Tests of Model Effects

By default, PROC SURVEYREG displays a "Tests of Model Effects" table, which provides Wald's F test for each effect in the model. The table contains the following information for each effect:

- Effect, which is the effect name
- Num DF, which is the numerator degrees of freedom for Wald's F test
- F Value, which is Wald's F statistic
- Pr > F, which is the significance probability corresponding to the F Value

A footnote displays the denominator degrees of freedom, which is the same for all effects.

Estimated Regression Coefficients

PROC SURVEYREG displays the "Estimated Regression Coefficients" table by default when there is no CLASS statement. Also, the procedure displays this table when you specify a CLASS statement and also specify the SOLUTION option in the MODEL statement. This table contains the following information for each regression parameter:

- Parameter, which identifies the effect or regressor variable
- Estimate, which is the estimate of the regression coefficient
- Standardized Estimate, which is the standardized regression coefficient
- Standard Error, which is the standard error of the estimate
- t Value, which is the t statistic for testing H_0: Parameter $= 0$
- Pr > | t |, which is the two-sided significance probability corresponding to the t Value

Covariance of Estimated Regression Coefficients

When you specify the COVB option in the MODEL statement, PROC SURVEYREG displays the "Covariance of Estimated Regression Coefficients" matrix.

Coefficients of Contrast

When you specify the E option in a CONTRAST statement, PROC SURVEYREG displays a "Coefficients of Contrast" table for the contrast. You can use this table to check the coefficients you specified in the CONTRAST statement. Also, this table gives a note for a nonestimable contrast.

Analysis of Contrasts

If you specify a CONTRAST statement, PROC SURVEYREG produces an "Analysis of Contrasts" table, which displays Wald's F test for the contrast. If you use more than one CONTRAST statement, the procedure displays all results in the same table. The "Analysis of Contrasts" table contains the following information for each contrast:

- Contrast, which is the label of the contrast
- Num DF, which is the numerator degrees of freedom for Wald's F test
- F Value, which is Wald's F statistic for testing H_0: Contrast $= 0$
- Pr > F, which is the significance probability corresponding to the F Value

Hadamard Matrix

If you specify the VARMETHOD=BRR(PRINTH) *method-option* in the PROC SURVEYREG statement, the procedure displays the Hadamard matrix.

When you provide a Hadamard matrix with the VARMETHOD=BRR(HADAMARD=) *method-option* but the procedure does not use the entire matrix, the procedure displays only the rows and columns that are actually used to construct replicates.

ODS Table Names

PROC SURVEYREG assigns a name to each table it creates; these names are listed in Table 123.11. You can use these names to refer to tables when you use the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

The EFFECT, ESTIMATE, LSMEANS, LSMESTIMATE, MARGINS, and SLICE statements also create tables, which are not listed in Table 123.11. For information about these tables, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

Table 123.11 ODS Tables Produced by PROC SURVEYREG

ODS Table Name	Description	Statement	Option
ANOVA	ANOVA for dependent variable	MODEL	ANOVA
ClassVarInfo	Class level information	CLASS	Default
ContrastCoef	Coefficients of contrast	CONTRAST	E
Contrasts	Analysis of contrasts	CONTRAST	Default

Table 123.11 *continued*

ODS Table Name	Description	Statement	Option
CovB	Covariance of estimated regression coefficients	MODEL	COVB
DataSummary	Data summary	PROC	Default
DesignSummary	Design summary	STRATA | CLUSTER	Default
DomainSummary	Domain summary	DOMAIN	Default
Effects	Tests of model effects	MODEL	Default
FitStatistics	Fit statistics	MODEL	Default
HadamardMatrix	Hadamard matrix	PROC	PRINTH
InvXPX	Inverse matrix of $\mathbf{X}'\mathbf{X}$	MODEL	I
ParameterEstimates	Estimated regression coefficients	MODEL	SOLUTION
StrataInfo	Stratum information	STRATA	LIST
VarianceEstimation	Variance estimation	PROC	Default
XPX	$\mathbf{X}'\mathbf{X}$ matrix	MODEL	XPX

By referring to the names of such tables, you can use the ODS OUTPUT statement to place one or more of these tables in output data sets.

For example, the following statements create an output data set MyStrata, which contains the StrataInfo table, an output data set MyParmEst, which contains the ParameterEstimates table, and an output data set Cov, which contains the CovB table for the ice cream study discussed in the section "Stratified Sampling" on page 10313:

```
title1 'Ice Cream Spending Analysis';
title2 'Stratified Sample Design';
proc surveyreg data=IceCream total=StudentTotals;
   strata Grade /list;
   class Kids;
   model Spending = Income Kids / solution covb;
   weight Weight;
   ods output StrataInfo = MyStrata
              ParameterEstimates = MyParmEst
              CovB = Cov;
run;
```

Note that the option CovB is specified in the MODEL statement in order to produce the covariance matrix table.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

When ODS Graphics is enabled, the ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements can produce plots that are associated with their analyses. For information about these plots, see the corresponding sections of Chapter 20, "Shared Concepts and Topics."

When ODS Graphics is enabled and when the regression model depends on at most one continuous variable as a regressor, excluding the intercept, the PLOTS= option in the PROC SURVEYREG statement controls fit plots for the regression.

PROC SURVEYREG provides a bubble plot or a heat map for model fitting. You can request a specific type of presentation of the weights by specifying the PLOTS(WEIGHT=) global plot option to request either a bubble plot or a heat map plot of the data that overlays the regression line and confidence limits band of the prediction in a fit plot. If you do not specify this option, the default plot depends on the number of observations in your data. That is, for a data set that contains 100 observations or less, the default is a bubble plot, in which the bubble area is proportional to the sampling weight of an observation. For a data set that contains more than 100 observations, the default is a heat map, in which the color of heat represents the sum of weights at the corresponding location.

PROC SURVEYREG assigns a name to each graph that it creates using ODS Graphics. You can use the name to refer to the graph. Table 123.12 lists the name of the graph that PROC SURVEYREG generates, together with its description and the PLOTS= option *plot-request* that produces it.

Table 123.12 ODS Graphs Produced by PROC SURVEYREG

ODS Graph Name	Description	PLOTS= Option
FitPlot	Regression line and confidence limits band of the prediction overlaid on a bubble plot or a heat map of the data	FIT

Examples: SURVEYREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 123.1: Simple Random Sampling

(View the complete code for this example (svrex1.sas) in the example repository.)

This example investigates the relationship between the labor force participation rate (LFPR) of women in 1968 and 1972 in large cities in the United States. A simple random sample of 19 cities is drawn from a total of 200 cities. For each selected city, the LFPRs are recorded and saved in a SAS data set Labor. In the following DATA step, LFPR in 1972 is contained in the variable LFPR1972, and the LFPR in 1968 is identified by the variable LFPR1968:

```
data Labor;
   input City $ 1-16 LFPR1972 LFPR1968;
   datalines;
New York        .45     .42
Los Angeles     .50     .50
Chicago         .52     .52
Philadelphia    .45     .45
Detroit         .46     .43
San Francisco   .55     .55
Boston          .60     .45
Pittsburgh      .49     .34
St. Louis       .35     .45
Connecticut     .55     .54
Washington D.C. .52     .42
Cincinnati      .53     .51
Baltimore       .57     .49
Newark          .53     .54
Minn/St. Paul   .59     .50
Buffalo         .64     .58
Houston         .50     .49
Patterson       .57     .56
Dallas          .64     .63
;
```

Assume that the LFPRs in 1968 and 1972 have a linear relationship, as shown in the following model:

$$\text{LFPR1972} = \beta_0 + \beta_1 * \text{LFPR1968} + \text{error}$$

You can use PROC SURVEYREG to obtain the estimated regression coefficients and estimated standard errors of the regression coefficients. The following statements perform the regression analysis:

```
ods graphics on;
title 'Study of Labor Force Participation Rates of Women';
proc surveyreg data=Labor total=200;
   model LFPR1972 = LFPR1968;
```

```
run;
```

Here, the TOTAL=200 option specifies the finite population total from which the simple random sample of 19 cities is drawn. You can specify the same information by using the sampling rate option RATE=0.095 (19/200=.095).

Output 123.1.1 summarizes the data information and the fit information.

Output 123.1.1 Summary of Regression Using Simple Random Sampling

Study of Labor Force Participation Rates of Women

The SURVEYREG Procedure

Regression Analysis for Dependent Variable LFPR1972

Data Summary	
Number of Observations	19
Mean of LFPR1972	0.52684
Sum of LFPR1972	10.01000

Fit Statistics	
R-Square	0.3970
Root MSE	0.05657
Denominator DF	18

Output 123.1.2 presents the significance tests for the model effects and estimated regression coefficients. The F tests and t tests for the effects in the model are also presented in these tables.

Output 123.1.2 Regression Coefficient Estimates

Tests of Model Effects

Effect	Num DF	F Value	Pr > F
Model	1	13.84	0.0016
Intercept	1	4.63	0.0452
LFPR1968	1	13.84	0.0016

Note: The denominator degrees of freedom for the F tests is 18.

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	0.20331056	0.09444296	2.15	0.0452
LFPR1968	0.65604048	0.17635810	3.72	0.0016

Note: The degrees of freedom for the t tests is 18.

From the regression performed by PROC SURVEYREG, you obtain a positive estimated slope for the linear relationship between the LFPR in 1968 and the LFPR in 1972. The regression coefficients are all significant at the 5% level. The effects Intercept and LFPR1968 are significant in the model at the 5% level. In this example, the F test for the overall model without intercept is the same as the effect LFPR1968.

When ODS graphics is enabled and you have only one regressor in the model, PROC SURVEYREG displays a plot of the model fitting, which is shown in Output 123.1.3.

Output 123.1.3 Regression Fitting

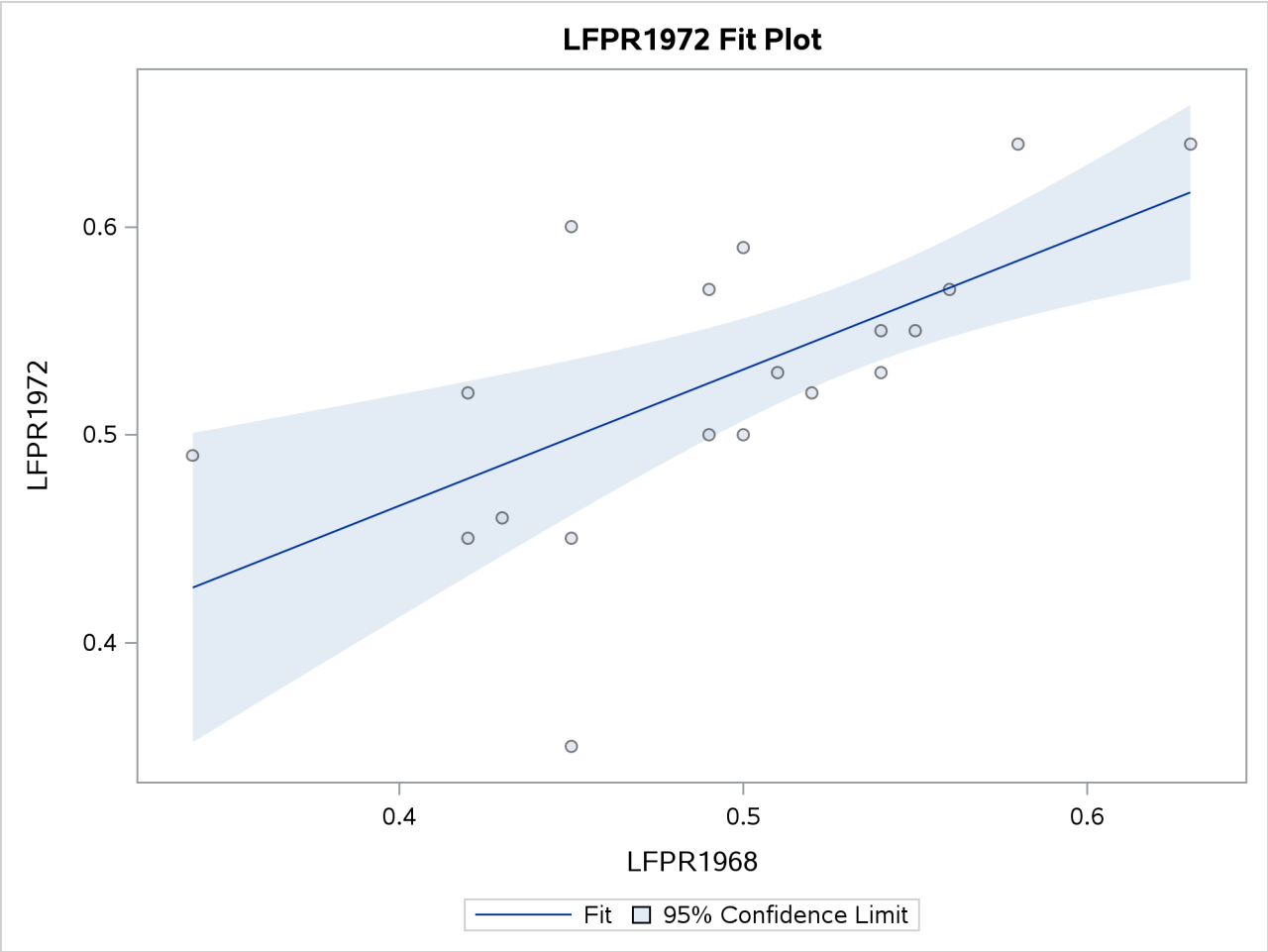

Example 123.2: Cluster Sampling

(View the complete code for this example (svrex2.sas) in the example repository.)

This example illustrates the use of regression analysis in a simple random cluster sample design. The data are from Särndal, Swensson, and Wretman (1992, p. 652). A total of 284 Swedish municipalities are grouped into 50 clusters of neighboring municipalities. Five clusters with a total of 32 municipalities are randomly selected. The results from the regression analysis in which clusters are used in the sample design are compared to the results of a regression analysis that ignores the clusters. The linear relationship between the population in 1975 and in 1985 is investigated.

The 32 selected municipalities in the sample are saved in the data set Municipalities:

```
data Municipalities;
   input Municipality Cluster Population85 Population75;
   datalines;
205   37    5    5
206   37   11   11
207   37   13   13
```

```
208   37    8    8
209   37   17   19
  6    2   16   15
  7    2   70   62
  8    2   66   54
  9    2   12   12
 10    2   60   50
 94   17    7    7
 95   17   16   16
 96   17   13   11
 97   17   12   11
 98   17   70   67
 99   17   20   20
100   17   31   28
101   17   49   48
276   50    6    7
277   50    9   10
278   50   24   26
279   50   10    9
280   50   67   64
281   50   39   35
282   50   29   27
283   50   10    9
284   50   27   31
 52   10    7    6
 53   10    9    8
 54   10   28   27
 55   10   12   11
 56   10  107  108
;
```

The variable Municipality identifies the municipalities in the sample; the variable Cluster indicates the cluster to which a municipality belongs; and the variables Population85 and Population75 contain the municipality populations in 1985 and in 1975 (in thousands), respectively. A regression analysis is performed by PROC SURVEYREG with a CLUSTER statement:

```
title1 'Regression Analysis for Swedish Municipalities';
title2 'Cluster Sampling';
proc surveyreg data=Municipalities total=50;
   cluster Cluster;
   model Population85=Population75;
run;
```

The TOTAL=50 option specifies the total number of clusters in the sampling frame.

Output 123.2.1 displays the data and design summary. Since the sample design includes clusters, the procedure displays the total number of clusters in the sample in the “Design Summary” table.

Output 123.2.1 Regression Analysis for Cluster Sampling

Regression Analysis for Swedish Municipalities
Cluster Sampling

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Population85

Data Summary	
Number of Observations	32
Mean of Population85	27.50000
Sum of Population85	880.00000

Design Summary	
Number of Clusters	5

Output 123.2.2 displays the fit statistics and regression coefficient estimates. In the "Estimated Regression Coefficients" table, the estimated slope for the linear relationship is 1.05, which is significant at the 5% level; but the intercept is not significant. This suggests that a regression line crossing the original can be established between populations in 1975 and in 1985.

Output 123.2.2 Regression Analysis for Cluster Sampling

Fit Statistics	
R-Square	0.9860
Root MSE	3.0488
Denominator DF	4

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	-0.0191292	0.89204053	-0.02	0.9839
Population75	1.0546253	0.05167565	20.41	<.0001

Note: The degrees of freedom for the t tests is 4.

The CLUSTER statement is necessary in PROC SURVEYREG in order to incorporate the sample design. If you do not specify a CLUSTER statement in the regression analysis, as in the following statements, the standard deviation of the regression coefficients are incorrectly estimated.

```
title1 'Regression Analysis for Swedish Municipalities';
title2 'Simple Random Sampling';
proc surveyreg data=Municipalities total=284;
   model Population85=Population75;
run;
```

The analysis ignores the clusters in the sample, assuming that the sample design is a simple random sampling. Therefore, the TOTAL= option specifies the total number of municipalities, which is 284.

Output 123.2.3 displays the regression results ignoring the clusters. Compared to the results in Output 123.2.2, the regression coefficient estimates are the same. However, without using clusters, the regression coefficients have a smaller variance estimate, as in Output 123.2.3. By using clusters in the analysis, the estimated

regression coefficient for effect Population75 is 1.05, with the estimated standard error 0.05, as displayed in Output 123.2.2; without using the clusters, the estimate is 1.05, but with the estimated standard error 0.04, as displayed in Output 123.2.3. To estimate the variance of the regression coefficients correctly, you should include the clustering information in the regression analysis.

Output 123.2.3 Regression Analysis for Simple Random Sampling

Regression Analysis for Swedish Municipalities
Simple Random Sampling

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Population85

Data Summary	
Number of Observations	32
Mean of Population85	27.50000
Sum of Population85	880.00000

Fit Statistics	
R-Square	0.9860
Root MSE	3.0488
Denominator DF	31

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	-0.0191292	0.67417606	-0.03	0.9775
Population75	1.0546253	0.03668414	28.75	<.0001

Note: The degrees of freedom for the t tests is 31.

Example 123.3: Regression Estimator for Simple Random Sample

(View the complete code for this example (svrex3.sas) in the example repository.)

By using auxiliary information, you can construct regression estimators to provide more accurate estimates of population characteristics. With ESTIMATE statements in PROC SURVEYREG, you can specify a regression estimator as a linear function of the regression parameters to estimate the population total. This example illustrates this application by using the data set Municipalities from Example 123.2.

In this sample, a linear model between the Swedish populations in 1975 and in 1985 is established:

$$\text{Population85} = \alpha + \beta * \text{Population75} + \text{error}$$

Assuming that the total population in 1975 is known to be 8200 (in thousands), you can use the ESTIMATE statement to predict the 1985 total population by using the following statements:

```
title1 'Regression Analysis for Swedish Municipalities';
title2 'Estimate Total Population';
proc surveyreg data=Municipalities total=50;
   cluster Cluster;
```

```
    model Population85=Population75;
    estimate '1985 population' Intercept 284 Population75 8200;
  run;
```

Since each observation in the sample is a municipality and there is a total of 284 municipalities in Sweden, the coefficient for Intercept (α) in the ESTIMATE statement is 284 and the coefficient for Population75 (β) is the total population in 1975 (8.2 million).

Output 123.3.1 displays the regression results and the estimation of the total population. By using the linear model, you can predict the total population in 1985 to be 8.64 million, with a standard error of 0.26 million.

Output 123.3.1 Use the Regression Estimator to Estimate the Population Total

Regression Analysis for Swedish Municipalities
Estimate Total Population

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Population85

Estimate					
Label	Estimate	Standard Error	DF	t Value	Pr > |t|
1985 population	8642.49	258.56	4	33.43	<.0001

Example 123.4: Stratified Sampling

(View the complete code for this example (svrex4.sas) in the example repository.)

This example illustrates the use of the SURVEYREG procedure to perform a regression in a stratified sample design. Consider a population of 235 farms producing corn in Nebraska and Iowa. You are interested in the relationship between corn yield (CornYield) and total farm size (FarmArea).

Each state is divided into several regions, and each region is used as a stratum. Within each stratum, a simple random sample with replacement is drawn. A total of 19 farms is selected by using a stratified simple random sample. The sample size and population size within each stratum are displayed in Table 123.13.

Table 123.13 Number of Farms in Each Stratum

			Number of Farms	
Stratum	**State**	**Region**	**Population**	**Sample**
1	Iowa	1	100	3
2		2	50	5
3		3	15	3
4	Nebraska	1	30	6
5		2	40	2
Total			**235**	**19**

The following three models are considered:

- Model I — Common intercept and slope:

$$\text{Corn Yield} = \alpha + \beta * \text{Farm Area}$$

- Model II — Common intercept, different slope:

$$\text{Corn Yield} = \begin{cases} \alpha + \beta_{\text{Iowa}} * \text{Farm Area} & \text{if the farm is in Iowa} \\ \alpha + \beta_{\text{Nebraska}} * \text{Farm Area} & \text{if the farm is in Nebraska} \end{cases}$$

- Model III — Different intercept and different slope:

$$\text{Corn Yield} = \begin{cases} \alpha_{\text{Iowa}} + \beta_{\text{Iowa}} * \text{Farm Area} & \text{if the farm is in Iowa} \\ \alpha_{\text{Nebraska}} + \beta_{\text{Nebraska}} * \text{Farm Area} & \text{if the farm is in Nebraska} \end{cases}$$

Data from the stratified sample are saved in the SAS data set Farms. The variable Weight contains the sampling weights, which are reciprocals of the selection probabilities.

```
data Farms;
   input State $ Region FarmArea CornYield Weight;
   datalines;
Iowa     1 100  54 33.333
Iowa     1  83  25 33.333
Iowa     1  25  10 33.333
Iowa     2 120  83 10.000
Iowa     2  50  35 10.000
Iowa     2 110  65 10.000
Iowa     2  60  35 10.000
Iowa     2  45  20 10.000
Iowa     3  23   5  5.000
Iowa     3  10   8  5.000
Iowa     3 350 125  5.000
Nebraska 1 130  20  5.000
Nebraska 1 245  25  5.000
Nebraska 1 150  33  5.000
Nebraska 1 263  50  5.000
Nebraska 1 320  47  5.000
Nebraska 1 204  25  5.000
Nebraska 2  80  11 20.000
Nebraska 2  48   8 20.000
;
```

The SAS data set StratumTotals contains the stratum population sizes.

```
data StratumTotals;
   input State $ Region _TOTAL_;
   datalines;
Iowa     1 100
Iowa     2  50
Iowa     3  15
Nebraska 1  30
Nebraska 2  40
;
```

Using the sample data from the data set Farms and the control information data from the data set StratumTotals, you can fit Model I by using the following statements in PROC SURVEYREG:

```
ods graphics on;
title1 'Analysis of Farm Area and Corn Yield';
title2 'Model I: Same Intercept and Slope';
proc surveyreg data=Farms total=StratumTotals;
   strata State Region / list;
   model CornYield = FarmArea / covB;
   weight Weight;
run;
```

Output 123.4.1 displays the data summary and stratification information fitting Model I. The sampling rates are automatically computed by the procedure based on the sample sizes and the population totals in strata.

Output 123.4.1 Data Summary and Stratum Information Fitting Model I

Analysis of Farm Area and Corn Yield
Model I: Same Intercept and Slope

The SURVEYREG Procedure

Regression Analysis for Dependent Variable CornYield

Data Summary	
Number of Observations	19
Sum of Weights	234.99900
Weighted Mean of CornYield	31.56029
Weighted Sum of CornYield	7416.6

Design Summary	
Number of Strata	5

Fit Statistics	
R-Square	0.3882
Root MSE	20.6422
Denominator DF	14

Stratum Information

Stratum Index	State	Region	N Obs	Population Total	Sampling Rate
1	Iowa	1	3	100	3.00%
2		2	5	50	10.0%
3		3	3	15	20.0%
4	Nebraska	1	6	30	20.0%
5		2	2	40	5.00%

Output 123.4.2 displays tests of model effects and the estimated regression coefficients.

Output 123.4.2 Estimated Regression Coefficients and the Estimated Covariance Matrix

Tests of Model Effects

Effect	Num DF	F Value	Pr > F
Model	1	21.74	0.0004
Intercept	1	4.93	0.0433
FarmArea	1	21.74	0.0004

Note: The denominator degrees of freedom for the F tests is 14.

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	11.8162978	5.31981027	2.22	0.0433
FarmArea	0.2126576	0.04560949	4.66	0.0004

Note: The degrees of freedom for the t tests is 14.

Covariance of Estimated Regression Coefficients

	Intercept	FarmArea
Intercept	28.300381277	-0.146471538
FarmArea	-0.146471538	0.0020802259

Output 123.4.3 displays the fit of the regression.

Output 123.4.3 Regression Fitting

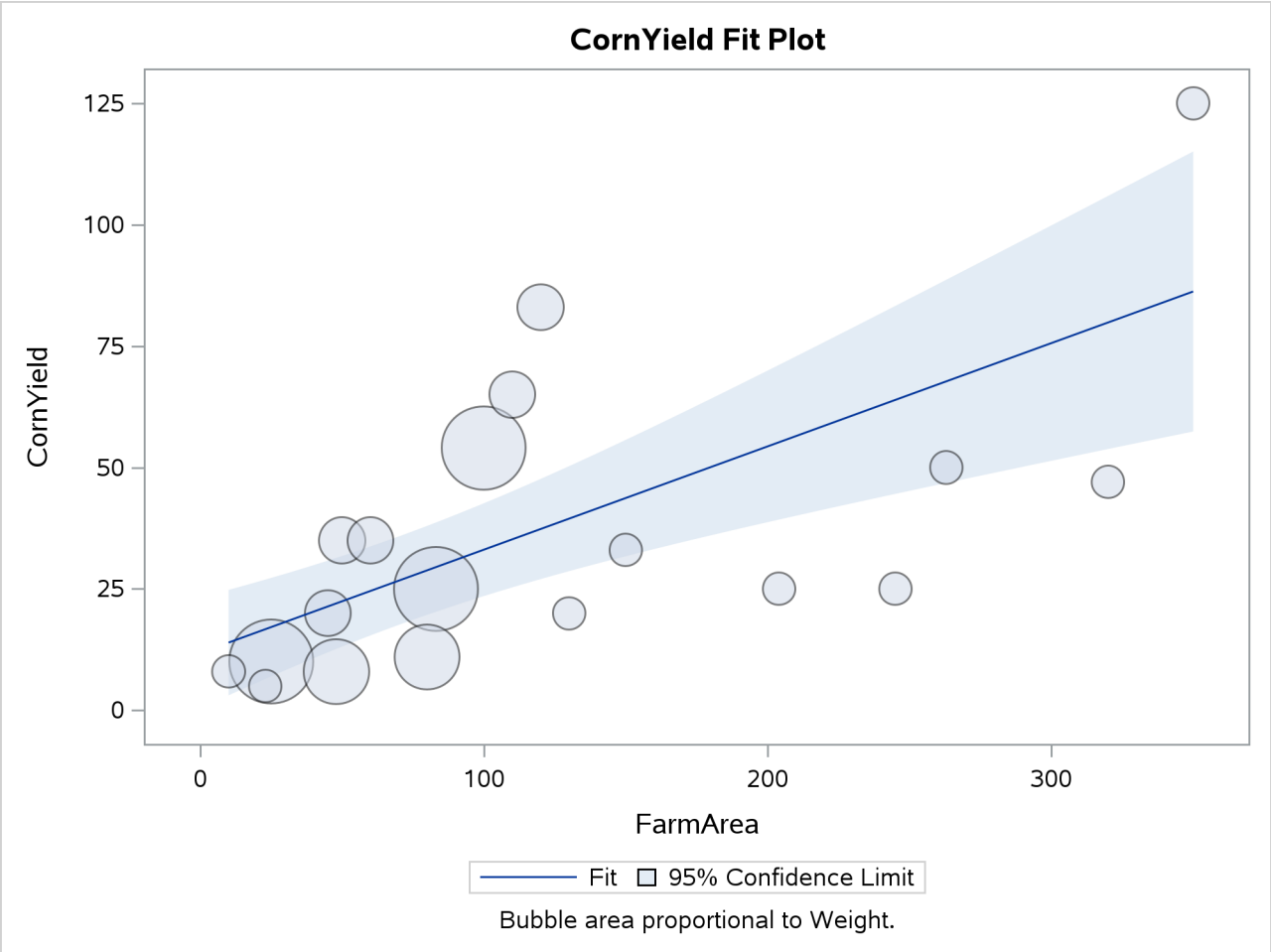

Alternatively, you can assume that the linear relationship between corn yield (CornYield) and farm area (FarmArea) is different among the states (Model II). In order to analyze the data by using this model, you create auxiliary variables FarmAreaNE and FarmAreaIA to represent farm area in different states:

$$\text{FarmAreaNE} = \begin{cases} 0 & \text{if the farm is in Iowa} \\ \text{FarmArea} & \text{if the farm is in Nebraska} \end{cases}$$

$$\text{FarmAreaIA} = \begin{cases} \text{FarmArea} & \text{if the farm is in Iowa} \\ 0 & \text{if the farm is in Nebraska} \end{cases}$$

The following statements create these variables in a new data set called FarmsByState and use PROC SURVEYREG to fit Model II:

```
data FarmsByState;
   set Farms;
   if State='Iowa' then do;
      FarmAreaIA=FarmArea;
      FarmAreaNE=0;
   end;
```

```
   else do;
      FarmAreaIA=0;
      FarmAreaNE=FarmArea;
   end;
run;
```

The following statements perform the regression by using the new data set FarmsByState. The analysis uses the auxiliary variables FarmAreaIA and FarmAreaNE as the regressors:

```
title1 'Analysis of Farm Area and Corn Yield';
title2 'Model II: Same Intercept, Different Slopes';
proc surveyreg data=FarmsByState total=StratumTotals;
   strata State Region;
   model CornYield = FarmAreaIA FarmAreaNE / covB;
   weight Weight;
run;
```

Output 123.4.4 displays the fit statistics and parameter estimates. The estimated slope parameters for each state are quite different from the estimated slope in Model I. The results from the regression show that Model II fits these data better than Model I.

Output 123.4.4 Regression Results from Fitting Model II

Analysis of Farm Area and Corn Yield
Model II: Same Intercept, Different Slopes

The SURVEYREG Procedure

Regression Analysis for Dependent Variable CornYield

Fit Statistics	
R-Square	0.8158
Root MSE	11.6759
Denominator DF	14

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	4.04234816	3.80934848	1.06	0.3066
FarmAreaIA	0.41696069	0.05971129	6.98	<.0001
FarmAreaNE	0.12851012	0.02495495	5.15	0.0001

Note: The degrees of freedom for the t tests is 14.

Covariance of Estimated Regression Coefficients

	Intercept	FarmAreaIA	FarmAreaNE
Intercept	14.511135861	-0.118001232	-0.079908772
FarmAreaIA	-0.118001232	0.0035654381	0.0006501109
FarmAreaNE	-0.079908772	0.0006501109	0.0006227496

For Model III, different intercepts are used for the linear relationship in two states. The following statements illustrate the use of the NOINT option in the MODEL statement associated with the CLASS statement to fit Model III:

```
title1 'Analysis of Farm Area and Corn Yield';
title2 'Model III: Different Intercepts and Slopes';
proc surveyreg data=FarmsByState total=StratumTotals;
   strata State Region;
   class State;
   model CornYield = State FarmAreaIA FarmAreaNE / noint covB solution;
   weight Weight;
run;
```

The model statement includes the classification effect State as a regressor. Therefore, the parameter estimates for effect State present the intercepts in two states.

Output 123.4.5 displays the regression results for fitting Model III, including parameter estimates, and covariance matrix of the regression coefficients. The estimated covariance matrix shows a lack of correlation between the regression coefficients from different states. This suggests that Model III might be the best choice for building a model for farm area and corn yield in these two states.

However, some statistics remain the same under different regression models—for example, Weighted Mean of CornYield. These estimators do not rely on the particular model you use.

Output 123.4.5 Regression Results for Fitting Model III

Analysis of Farm Area and Corn Yield
Model III: Different Intercepts and Slopes

The SURVEYREG Procedure

Regression Analysis for Dependent Variable CornYield

Fit Statistics	
R-Square	0.9300
Root MSE	11.9810
Denominator DF	14

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
State Iowa	5.27797099	5.27170400	1.00	0.3337
State Nebraska	0.65275201	1.70031616	0.38	0.7068
FarmAreaIA	0.40680971	0.06458426	6.30	<.0001
FarmAreaNE	0.14630563	0.01997085	7.33	<.0001

Note: The degrees of freedom for the t tests is 14.

Covariance of Estimated Regression Coefficients

	State Iowa	State Nebraska	FarmAreaIA	FarmAreaNE
State Iowa	27.790863033	0	-0.205517205	0
State Nebraska	0	2.8910750385	0	-0.027354011
FarmAreaIA	-0.205517205	0	0.0041711265	0
FarmAreaNE	0	-0.027354011	0	0.0003988349

Example 123.5: Regression Estimator for Stratified Sample

(View the complete code for this example (svrex5.sas) in the example repository.)

This example uses the corn yield data set FARMS from Example 123.4 to illustrate how to construct a regression estimator for a stratified sample design.

As in Example 123.3, by incorporating auxiliary information into a regression estimator, the procedure can produce more accurate estimates of the population characteristics that are of interest. In this example, the sample design is a stratified sample design. The auxiliary information is the total farm areas in regions of each state, as displayed in Table 123.14. You want to estimate the total corn yield by using this information under the three linear models given in Example 123.4.

Table 123.14 Information for Each Stratum

			Number of Farms		
Stratum	**State**	**Region**	**Population**	**Sample**	**Total Farm Area**
1	Iowa	1	100	3	
2		2	50	5	13,200
3		3	15	3	
4	Nebraska	1	30	6	8,750
5		2	40	2	
Total			**235**	**19**	**21,950**

The regression estimator to estimate the total corn yield under Model I can be obtained by using PROC SURVEYREG with an ESTIMATE statement:

```
title1 'Estimate Corn Yield from Farm Size';
title2 'Model I: Same Intercept and Slope';
proc surveyreg data=Farms total=StratumTotals;
   strata State Region / list;
   class  State Region;
   model  CornYield = FarmArea State*Region /solution;
   weight Weight;
   estimate 'Estimate of CornYield under Model I'
           INTERCEPT 235 FarmArea 21950
           State*Region 100 50 15 30 40 /e;
run;
```

To apply the constraint in each stratum that the weighted total number of farms equals to the total number of farms in the stratum, you can include the strata as an effect in the MODEL statement, effect State*Region. Thus, the CLASS statement must list the STRATA variables, State and Region, as classification variables. The following ESTIMATE statement specifies the regression estimator, which is a linear function of the regression parameters:

```
   estimate 'Estimate of CornYield under Model I'
           INTERCEPT 235 FarmArea 21950
           State*Region 100 50 15 30 40 /e;
```

This linear function contains the total for each explanatory variable in the model. Because the sampling units are farms in this example, the coefficient for Intercept in the ESTIMATE statement is the total number of farms (235); the coefficient for FarmArea is the total farm area listed in Table 123.14 (21950); and the coefficients for effect State*Region are the total number of farms in each strata (as displayed in Table 123.14).

Output 123.5.1 displays the results of the ESTIMATE statement. The regression estimator for the total of CornYield in Iowa and Nebraska is 7464 under Model I, with a standard error of 927.

Output 123.5.1 Regression Estimator for the Total of CornYield under Model I

Estimate Corn Yield from Farm Size
Model I: Same Intercept and Slope

The SURVEYREG Procedure

Regression Analysis for Dependent Variable CornYield

	Estimate				
Label	Estimate	Standard Error	DF	t Value	Pr > |t|
Estimate of CornYield under Model I	7463.52	926.84	14	8.05	<.0001

Under Model II, a regression estimator for totals can be obtained by using the following statements:

```
title1 'Estimate Corn Yield from Farm Size';
title2 'Model II: Same Intercept, Different Slopes';
proc surveyreg data=FarmsByState total=StratumTotals;
   strata State Region;
   class  State Region;
   model  CornYield = FarmAreaIA FarmAreaNE
                      state*region /solution;
   weight Weight;
   estimate 'Total of CornYield under Model II'
           INTERCEPT 235 FarmAreaIA 13200 FarmAreaNE 8750
           State*Region 100 50 15 30 40 /e;
run;
```

In this model, you also need to include strata as a fixed effect in the MODEL statement. Other regressors are the auxiliary variables FarmAreaIA and FarmAreaNE (defined in Example 123.4). In the following ESTIMATE statement, the coefficient for Intercept is still the total number of farms; and the coefficients for FarmAreaIA and FarmAreaNE are the total farm area in Iowa and Nebraska, respectively, as displayed in Table 123.14. The total number of farms in each strata are the coefficients for the strata effect:

```
   estimate 'Total of CornYield under Model II'
           INTERCEPT 235 FarmAreaIA 13200 FarmAreaNE 8750
           State*Region 100 50 15 30 40 /e;
```

Output 123.5.2 displays that the results of the regression estimator for the total of corn yield in two states under Model II is 7580 with a standard error of 859. The regression estimator under Model II has a slightly smaller standard error than under Model I.

Output 123.5.2 Regression Estimator for the Total of CornYield under Model II

Estimate Corn Yield from Farm Size
Model II: Same Intercept, Different Slopes

The SURVEYREG Procedure

Regression Analysis for Dependent Variable CornYield

	Estimate				
Label	Estimate	Standard Error	DF	t Value	Pr > |t|
Total of CornYield under Model II	7580.49	859.18	14	8.82	<.0001

Finally, you can apply Model III to the data and estimate the total corn yield. Under Model III, you can also obtain the regression estimators for the total corn yield for each state. Three ESTIMATE statements are used in the following statements to create the three regression estimators:

```
title1 'Estimate Corn Yield from Farm Size';
title2 'Model III: Different Intercepts and Slopes';
proc surveyreg data=FarmsByState total=StratumTotals;
   strata State Region;
   class  State Region;
   model  CornYield = state FarmAreaIA FarmAreaNE
      State*Region /noint solution;
   weight Weight;
   estimate 'Total CornYield in Iowa under Model III'
             State 165 0 FarmAreaIA 13200 FarmAreaNE  0
             State*region 100 50 15  0  0 /e;
   estimate 'Total CornYield in Nebraska under Model III'
             State 0 70 FarmAreaIA 0 FarmAreaNE 8750
             State*Region 0 0 0 30 40 /e;
   estimate 'Total CornYield in both states under Model III'
             State 165 70 FarmAreaIA 13200 FarmAreaNE 8750
             State*Region 100 50 15 30 40 /e;
run;
```

The fixed effect State is added to the MODEL statement to obtain different intercepts in different states, by using the NOINT option. Among the ESTIMATE statements, the coefficients for explanatory variables are different depending on which regression estimator is estimated. For example, in the ESTIMATE statement

```
   estimate 'Total CornYield in Iowa under Model III'
             State 165 0 FarmAreaIA 13200 FarmAreaNE  0
             State*region 100 50 15  0  0 /e;
```

the coefficients for the effect State are 165 and 0, respectively. This indicates that the total number of farms in Iowa is 165 and the total number of farms in Nebraska is 0, because the estimation is the total corn yield in Iowa only. Similarly, the total numbers of farms in three regions in Iowa are used for the coefficients of the strata effect State*Region, as displayed in Table 123.14.

Output 123.5.3 displays the results from the three regression estimators by using Model III. Since the estimations are independent in each state, the total corn yield from both states is equal to the sum of the estimated total of corn yield in Iowa and Nebraska, $6246 + 1334 = 7580$. This regression estimator is the same as the one under Model II. The variance of regression estimator of the total corn yield in both states is

the sum of variances of regression estimators for total corn yield in each state. Therefore, it is not necessary to use Model III to obtain the regression estimator for the total corn yield unless you need to estimate the total corn yield for each individual state.

Output 123.5.3 Regression Estimator for the Total of CornYield under Model III

Estimate Corn Yield from Farm Size
Model III: Different Intercepts and Slopes

The SURVEYREG Procedure

Regression Analysis for Dependent Variable CornYield

Label	Estimate	Standard Error	DF	t Value	Pr > |t|
Total CornYield in Iowa under Model III	6246.11	851.27	14	7.34	<.0001

Example 123.6: Stratum Collapse

(View the complete code for this example (svrex6.sas) in the example repository.)

In a stratified sample, it is possible that some strata might have only one sampling unit. When this happens, PROC SURVEYREG collapses the strata that contain a single sampling unit into a pooled stratum. For more detailed information about stratum collapse, see the section "Stratum Collapse" on page 10354.

Suppose that you have the following data:

```
data Sample;
   input Stratum X Y W;
   datalines;
10 0 0 5
10 1 1 5
11 1 1 10
11 1 2 10
12 3 3 16
33 4 4 45
14 6 7 50
12 3 4 16
;
```

The variable Stratum is again the stratification variable, the variable X is the independent variable, and the variable Y is the dependent variable. You want to regress Y on X. In the data set Sample, both Stratum=33 and Stratum=14 contain one observation. By default, PROC SURVEYREG collapses these strata into one pooled stratum in the regression analysis.

To input the finite population correction information, you create the SAS data set StratumTotals:

```
data StratumTotals;
   input Stratum _TOTAL_;
   datalines;
10 10
11 20
```

```
12 32
33 40
33 45
14 50
15  .
66 70
;
```

The variable Stratum is the stratification variable, and the variable _TOTAL_ contains the stratum totals. The data set StratumTotals contains more strata than the data set Sample. Also in the data set StratumTotals, more than one observation contains the stratum totals for Stratum=33:

```
33 40
33 45
```

PROC SURVEYREG allows this type of input. The procedure simply ignores strata that are not present in the data set Sample; for the multiple entries of a stratum, the procedure uses the first observation. In this example, Stratum=33 has the stratum total _TOTAL_=40.

The following SAS statements perform the regression analysis:

```
title1 'Stratified Sample with Single Sampling Unit in Strata';
title2 'With Stratum Collapse';
proc surveyreg data=Sample total=StratumTotals;
   strata Stratum/list;
   model Y=X;
   weight W;
run;
```

Output 123.6.1 shows that there are a total of five strata in the input data set and two strata are collapsed into a pooled stratum. The denominator degrees of freedom is 4, due to the collapse (see the section "Denominator Degrees of Freedom" on page 10363).

Output 123.6.1 Summary of Data and Regression

Stratified Sample with Single Sampling Unit in Strata
With Stratum Collapse

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Y

Data Summary	
Number of Observations	8
Sum of Weights	157.00000
Weighted Mean of Y	4.31210
Weighted Sum of Y	677.00000

Design Summary	
Number of Strata	5
Number of Strata Collapsed	2

Output 123.6.1 *continued*

Fit Statistics	
R-Square	0.9564
Root MSE	0.5111
Denominator DF	4

Output 123.6.2 displays the stratification information, including stratum collapse. Under the column Collapsed, the fourth stratum (Stratum=14) and the fifth (Stratum=33) are marked as 'Yes,' which indicates that these two strata are collapsed into the pooled stratum (Stratum Index=0). The sampling rate for the pooled stratum is 2% (see the section "Sampling Rate of the Pooled Stratum from Collapse" on page 10354).

Output 123.6.3 displays the parameter estimates and the tests of the significance of the model effects.

Output 123.6.2 Stratification Information

Stratum Information					
Stratum Index	Collapsed	Stratum	N Obs	Population Total	Sampling Rate
1		10	2	10	20.0%
2		11	2	20	10.0%
3		12	2	32	6.25%
4	Yes	14	1	50	2.00%
5	Yes	33	1	40	2.50%
0	Pooled		2	90	2.22%

Note: Strata with only one observation are collapsed into the stratum with Stratum Index "0".

Output 123.6.3 Parameter Estimates and Effect Tests

Tests of Model Effects			
Effect	Num DF	F Value	Pr > F
Model	1	173.01	0.0002
Intercept	1	0.00	0.9961
X	1	173.01	0.0002

Note: The denominator degrees of freedom for the F tests is 4.

Estimated Regression Coefficients				
Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	0.00179469	0.34306373	0.01	0.9961
X	1.12598708	0.08560466	13.15	0.0002

Note: The degrees of freedom for the t tests is 4.

Alternatively, if you prefer not to collapse strata with a single sampling unit, you can specify the NOCOLLAPSE option in the STRATA statement:

```
title1 'Stratified Sample with Single Sampling Unit in Strata';
title2 'Without Stratum Collapse';
proc surveyreg data=Sample total=StratumTotals;
   strata Stratum/list nocollapse;
```

```
    model Y = X;
    weight W;
run;
```

Output 123.6.4 does not contain the stratum collapse information displayed in Output 123.6.1, and the denominator degrees of freedom are 3 instead of 4.

Output 123.6.4 Summary of Data and Regression

Stratified Sample with Single Sampling Unit in Strata Without Stratum Collapse

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Y

Data Summary	
Number of Observations	8
Sum of Weights	157.00000
Weighted Mean of Y	4.31210
Weighted Sum of Y	677.00000

Design Summary	
Number of Strata	5

Fit Statistics	
R-Square	0.9564
Root MSE	0.5111
Denominator DF	3

In Output 123.6.5, although the fourth stratum and the fifth stratum contain only one observation, no stratum collapse occurs.

Output 123.6.5 Stratification Information

Stratum Information				
Stratum Index	Stratum	N Obs	Population Total	Sampling Rate
1	10	2	10	20.0%
2	11	2	20	10.0%
3	12	2	32	6.25%
4	14	1	50	2.00%
5	33	1	40	2.50%

As a result of not collapsing strata, the standard error estimates of the parameters, shown in Output 123.6.6, are different from those in Output 123.6.3, as are the tests of the significance of model effects.

Output 123.6.6 Parameter Estimates and Effect Tests

Tests of Model Effects			
Effect	**Num DF**	**F Value**	**Pr > F**
Model	1	347.27	0.0003
Intercept	1	0.00	0.9962
X	1	347.27	0.0003

Note: The denominator degrees of freedom for the F tests is 3.

Estimated Regression Coefficients				
Parameter	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	0.00179469	0.34302581	0.01	0.9962
X	1.12598708	0.06042241	18.64	0.0003

Note: The degrees of freedom for the t tests is 3.

Example 123.7: Domain Analysis

(View the complete code for this example (svrex7.sas) in the example repository.)

You can use PROC SURVEYREG to perform domain analysis in a subgroup of your interest. To illustrate, this example uses a data set from the National Health and Nutrition Examination Survey I (NHANES I) Epidemiologic Followup Study (NHEFS), described in Example 122.2 in Chapter 122, "The SURVEYPHREG Procedure."

The NHEFS is a national longitudinal survey that is conducted by the National Center for Health Statistics, the National Institute on Aging, and some other agencies of the Public Health Service in the United States. Some important objectives of this survey are to determine the relationships between clinical, nutritional, and behavioral factors; to determine the relationship between mortality and hospital utilization; and to monitor changes in risk factors for the initial cohort that represents the NHANES I population. A cohort of size 14,407, which includes all persons 25 to 74 years old who completed a medical examination at NHANES I in 1971–1975, was selected for the NHEFS. Personal interviews were conducted for every selected unit during the first wave of data collection from the year 1982 to 1984. Follow-up studies were conducted in 1986, 1987, and 1992. In the year 1986, only nondeceased persons 55 to 74 years old (as reported in the base year survey) were interviewed. The 1987 and 1992 NHEFS contain the entire nondeceased NHEFS cohort. Vital and tracing status data, interview data, health care facility stay data, and mortality data for all four waves are available for public use. For more information about the survey and the data sets, see the Center of Disease Control and Prevention's website (`http://www.cdc.gov/`).

For illustration purposes, 1,018 observations from the 1987 NHEFS public use interview data are used to create the data set cancer. The observations are obtained from 10 strata that contain 596 PSUs. The sum of observation weights for these selected units is over 19 million. Observation weights range from 359 to 129,359 with a mean of 18,747.69 and a median of 11,414.

The following variables are used in this example:

- ObsNo, unit identification
- Strata, stratum identification

- PSU, identification for primary sampling units
- ObservationWt, sampling weight associated with each unit
- Age, the event-time variable, defined as follows:
  - age of the subject when the first cancer was reported for subjects with reported cancer
  - age of the subject at death for deceased subjects without reported cancer
  - age of the subject as reported in 1987 follow-up (this value is used for nondeceased subjects who never reported cancer)
  - age of the subject for the entry year 1971–1975 survey if the subject has cancer (or is deceased) but the date of incident is not reported
- Cancer, cancer indicator (1 = cancer reported, 0 = cancer not reported)
- BodyWeight, body weight of the subject as reported in the 1987 follow-up, or an imputed body weight based on the subject's age in the entry year 1971–1975 survey

The following SAS statements create the data set cancer. Note that BodyWeight for a few observations (8%) is imputed based on Age by using a deterministic regression imputation model (Särndal and Lundström (2005, chapter 12)). The imputed values are treated as observed values in this example. In other words, this example treats the data set Cancer as the observed data set.

```
data cancer;
   input ObsNo Strata PSU ObservationWt Age Cancer BodyWeight;
   datalines;
   1  3  002  3805   53  1  175
   2  3  002  6107   77  0  175
   3  3  039  2968   50  0  160
   4  3  084  30438  52  0  145
   5  3  007  5081   80  0  127

   ... more lines ...

1016  4  002   2689  40  0  120
1017  4  092  45888  52  0  166
1018  4  035   4347  58  0  156
;
```

Suppose you want to study how aging affects body weight in the subgroup of cancer patients for the base year survey population. Because whether an individual has cancer or not is unrelated to the design of the sample, this kind of analysis is called domain analysis (subgroup analysis).

The following statements request a linear regression of BodyWeight on Age among cancer patients. The STRATA, CLUSTER, and WEIGHT statements identify the variance strata, PSUs, and analysis weights, respectively. The DOMAIN statement defines the subgroups of people who have been diagnosed with cancer and people who do not have cancer. The option ('1') after the DOMAIN variable Cancer requests that the procedure display only the analysis in the subgroup Cancer=1 in the output. The PLOT= option in the PROC statement requests that weights be represented as a heat map with hexagonal bins.

```
title1 'Study of Body Weight and Age among Cancer Patients';
ods graphics on;
proc surveyreg data=cancer plot=fit(weight=heatmap shape=hex);
   strata strata;
   cluster psu;
   weight ObservationWt;
   model bodyweight = age;
   domain cancer ('1');
run;
```

Output 123.7.1 gives a summary of the data and the parameter estimates of the linear regression in the domain Cancer=1. The analysis indicates that aging does not significantly affect body weight among cancer patients.

Output 123.7.1 Domain Analysis among Cancer Patients

Study of Body Weight and Age among Cancer Patients

The SURVEYREG Procedure

Cancer=1

Domain Regression Analysis for Variable BodyWeight

Domain Summary	
Number of Observations	1017
Number of Observations in Domain	119
Number of Observations Not in Domain	898
Sum of Weights in Domain	2211545.0
Weighted Mean of BodyWeight	164.87655
Weighted Sum of BodyWeight	364631909

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	189.614789	14.9467889	12.69	<.0001
Age	-0.398556	0.2398447	-1.66	0.0971

Note: The degrees of freedom for the t tests is 586.

When ODS Graphics is enabled and the model contains a single continuous regressor, PROC SURVEYREG displays a plot of the model fitting, which is shown in Figure 123.7.2.

Output 123.7.2 Regression Fitting

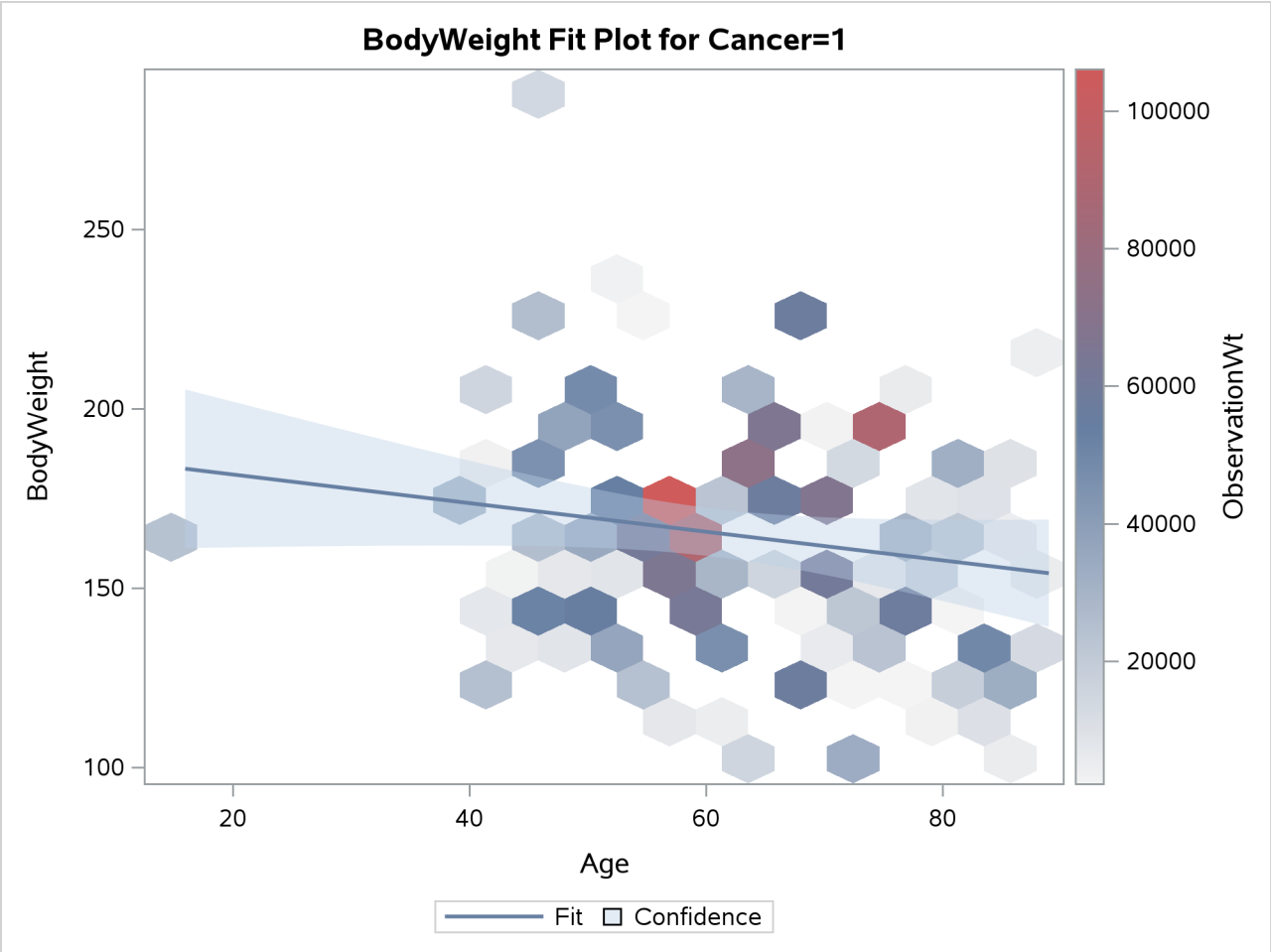

Example 123.8: Comparing Domain Statistics

(View the complete code for this example (svrex8.sas) in the example repository.)

Recall the example in the section "Getting Started: SURVEYREG Procedure" on page 10311, which analyzed a stratified simple random sample from a junior high school to examine how household income and the number of children in a household affect students' average weekly spending for ice cream. You can use the same sample to analyze the average weekly spending among male and female students. Because student gender is unrelated to the design of the sample, this kind of analysis is called domain analysis (subgroup analysis).

The data set follows:

```
data IceCreamDataDomain;
   input Grade Spending Income Gender$ @@;
   datalines;
7   7  39  M   7   7  38  F   8  12  47  F
9  10  47  M   7   1  34  M   7  10  43  M
7   3  44  M   8  20  60  F   8  19  57  M
```

```
7   2  35  M   7   2  36  F   9  15  51  F
8  16  53  F   7   6  37  F   7   6  41  M
7   6  39  M   9  15  50  M   8  17  57  F
8  14  46  M   9   8  41  M   9   8  41  F
9   7  47  F   7   3  39  F   7  12  50  M
7   4  43  M   9  14  46  F   8  18  58  M
9   9  44  F   7   2  37  F   7   1  37  M
7   4  44  M   7  11  42  M   9   8  41  M
8  10  42  M   8  13  46  F   7   2  40  F
9   6  45  F   9  11  45  M   7   2  36  F
7   9  46  F
;
data IceCreamDataDomain;
   set IceCreamDataDomain;
   if Grade=7 then Prob=20/1824;
   if Grade=8 then Prob=9/1025;
   if Grade=9 then Prob=11/1151;
   Weight=1/Prob;
run;
data StudentTotals;
   input Grade _TOTAL_;
   datalines;
7 1824
8 1025
9 1151
;
```

In the data set IceCreamDataDomain, the variable Grade indicates a student's grade, which is the stratification variable. The variable Spending contains the dollar amount of each student's average weekly spending for ice cream. The variable Income specifies the household income, in thousands of dollars. The variable Gender indicates a student's gender. The sampling weights are created by using the reciprocals of the probabilities of selection.

In the data set StudentTotals, the variable Grade is the stratification variable, and the variable _TOTAL_ contains the total numbers of students in the strata in the survey population.

Suppose that you are now interested in estimating the gender domain means of weekly ice cream spending (that is, the average spending for males and females, respectively). You can use the SURVEYMEANS procedure to produce these domain statistics by using the following statements:

```
proc surveymeans data=IceCreamDataDomain total=StudentTotals;
   strata Grade;
   var spending;
   domain Gender;
   weight Weight;
run;
```

Output 123.8.1 shows the estimated spending among male and female students.

Output 123.8.1 Estimated Domain Means

The SURVEYMEANS Procedure

Statistics for Gender Domains						
Gender	Variable	N	Mean	Std Error of Mean	95% CL for Mean	
F	Spending	19	9.376111	1.077927	7.19202418	11.5601988
M	Spending	21	8.923052	1.003423	6.88992385	10.9561807

You can also use PROC SURVEYREG to estimate these domain means. The benefit of this alternative approach is that PROC SURVEYREG provides more tools for additional analysis, such as domain means comparisons in a LSMEANS statement.

Suppose that you want to test whether there is a significant difference for the ice cream spending between male and female students. You can use the following statements to perform the test:

```
title1 'Ice Cream Spending Analysis';
title2 'Compare Domain Statistics';
proc surveyreg data=IceCreamDataDomain total=StudentTotals;
   strata Grade;
   class Gender;
   model Spending = Gender / vadjust=none;
   lsmeans Gender / diff;
   weight Weight;
run;
```

The variable Gender is used as a model effect. The VADJUST=NONE option is used to produce variance estimates for domain means that are identical to those produced by PROC SURVEYMEANS. The LSMEANS statement requests that PROC SURVEYREG estimate the average spending in each gender group. The DIFF option requests that the procedure compute the difference among domain means.

Output 123.8.2 displays the estimated weekly spending on ice cream among male and female students, respectively, and their standard errors. Female students spend $9.38 per week on average, and male students spend $8.92 per week on average. These domain means, including their standard errors, are identical to those in Output 123.8.1 which are produced by PROC SURVEYMEANS.

Output 123.8.2 Domain Means between Gender

Ice Cream Spending Analysis
Compare Domain Statistics

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Spending

Gender Least Squares Means					
Gender	Estimate	Standard Error	DF	t Value	Pr > |t|
F	9.3761	1.0779	37	8.70	<.0001
M	8.9231	1.0034	37	8.89	<.0001

Output 123.8.3 shows the estimated difference for weekly ice scream spending between the two gender

groups. The female students spend $0.45 more than male students on average, and the difference is not statistically significant based on the *t* test.

Output 123.8.3 Domain Means Comparison

Differences of Gender Least Squares Means						
Gender	_Gender	Estimate	Standard Error	DF	t Value	Pr > |t|
F	M	0.4531	1.7828	37	0.25	0.8008

If you want to investigate whether there is any significant difference in ice cream spending among grades, you can use the following similar statements to compare:

```
ods graphics on;
title1 'Ice Cream Spending Analysis';
title2 'Compare Domain Statistics';
proc surveyreg data=IceCreamDataDomain total=StudentTotals;
   strata Grade;
   class Grade;
   model Spending = Grade / vadjust=none;
   lsmeans Grade / diff plots=(diff meanplot(cl));
   weight Weight;
run;
```

The Grade is specified in the CLASS statement to be used as an effect in the MODEL statement. The DIFF option in the LSMEANS statement requests that the procedure compute the difference among the domain means for the effect Grade. The ODS GRAPHICS statement enables ODS to create graphics. The PLOTS=(DIFF MEANPLOT(CL)) option requests two graphics: the domain means plot MeanPlot and their pairwise difference plot DiffPlot. The CL suboption requests the MeanPlot to display confidence. For information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS."

Output 123.8.4 shows the estimated weekly spending on ice cream for students within each grade. Students in Grade 7 spend the least, only $5.00 per week. Students in Grade 8 spend the most, $15.44 per week. Students in Grade 9 spend a little less at $10.09 per week.

Output 123.8.4 Domain Means among Grades

Ice Cream Spending Analysis
Compare Domain Statistics

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Spending

Grade Least Squares Means					
Grade	Estimate	Standard Error	DF	t Value	Pr > |t|
7	5.0000	0.7636	37	6.55	<.0001
8	15.4444	1.1268	37	13.71	<.0001
9	10.0909	0.9719	37	10.38	<.0001

Output 123.8.5 plots the weekly spending results that are shown in Output 123.8.4.

Output 123.8.5 Plot of Means of Ice Cream Spending within Grades

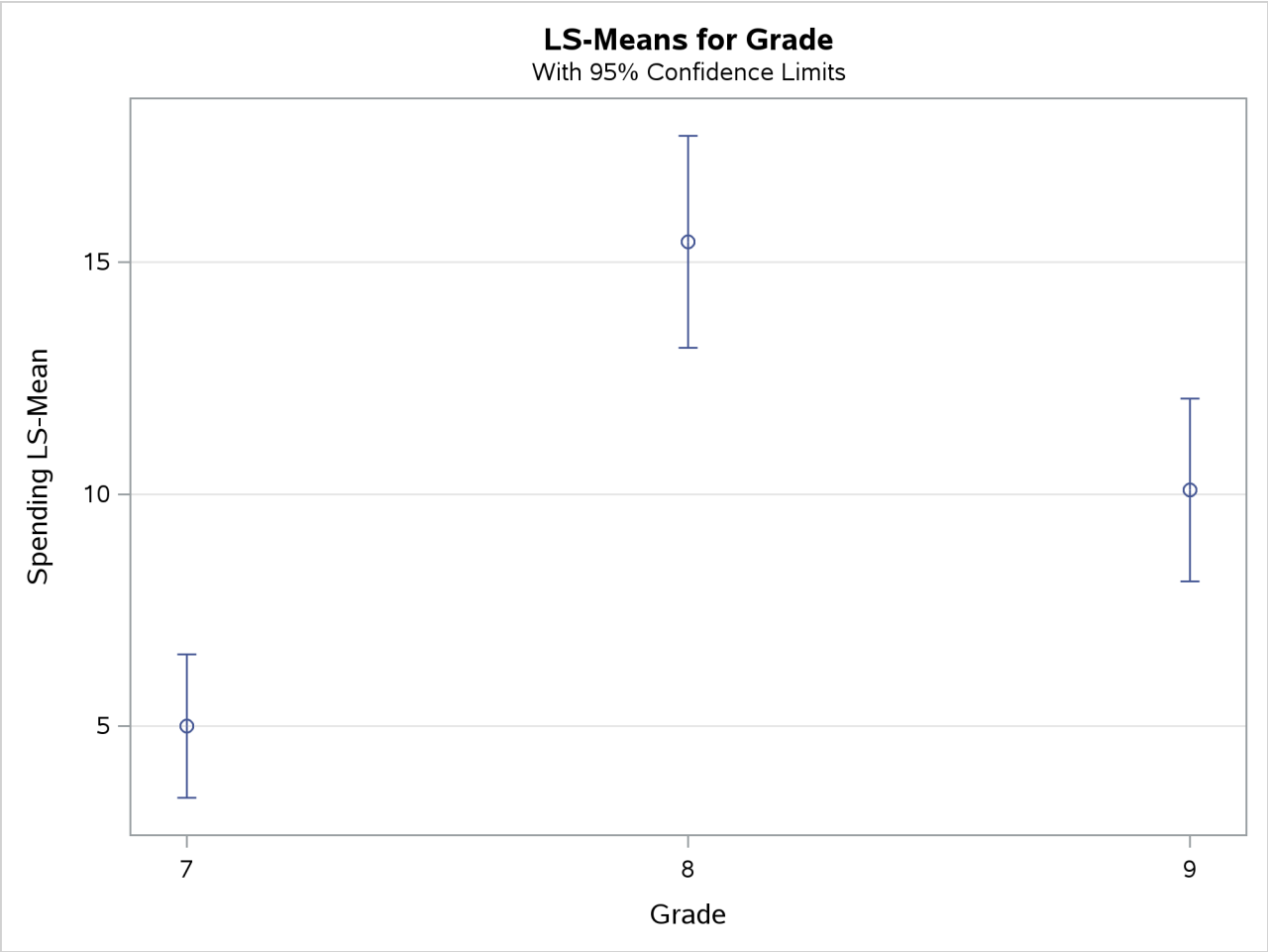

Output 123.8.6 displays pairwise comparisons for weekly ice scream spending among grades. All the differences are significant based on *t* tests.

Output 123.8.6 Domain Means Comparison

Differences of Grade Least Squares Means						
Grade	_Grade	Estimate	Standard Error	DF	t Value	Pr > |t|
7	8	-10.4444	1.3611	37	-7.67	<.0001
7	9	-5.0909	1.2360	37	-4.12	0.0002
8	9	5.3535	1.4880	37	3.60	0.0009

Output 123.8.7 plots the comparisons that are shown in Output 123.8.6.

Output 123.8.7 Plot of Pairwise Comparisons of Spending among Grades

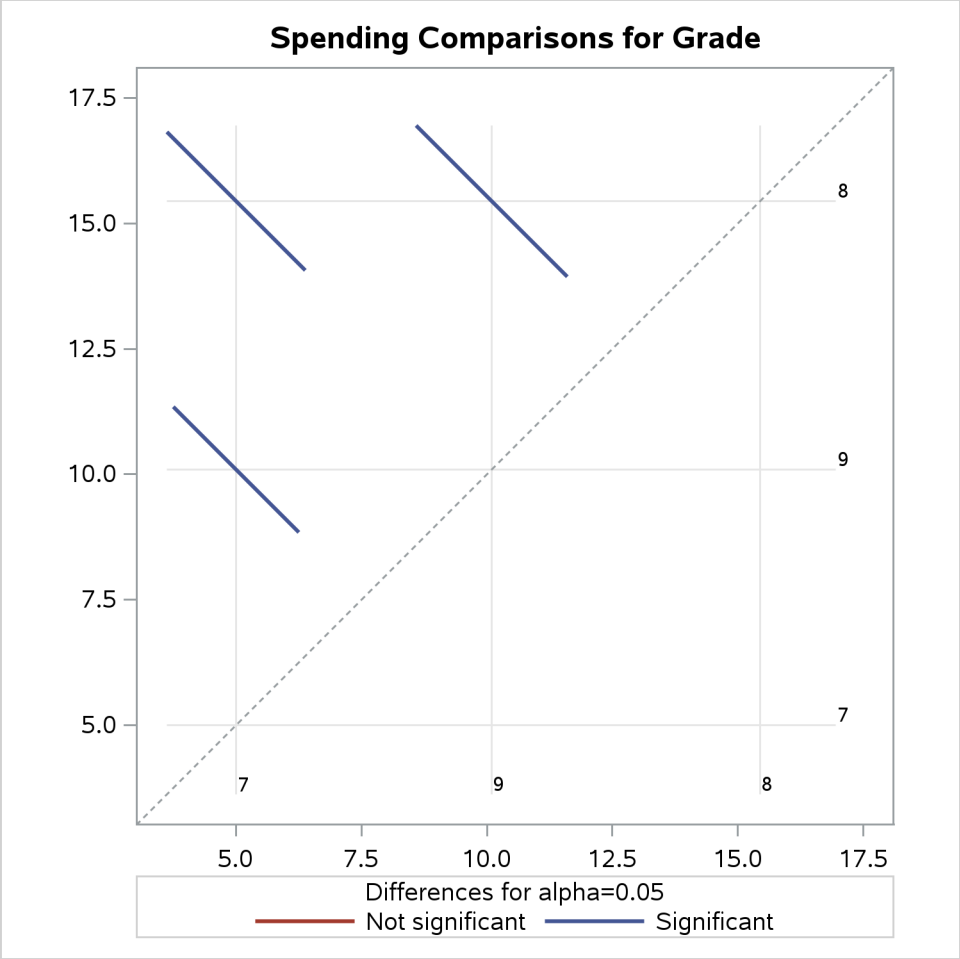

In Output 123.8.7, the spending for each grade is shown in the background grid on both axes. Comparisons for each pair of domain means are shown by colored bars at intersections of these grids. The length of each bar represents the width of the confidence intervals for the corresponding difference between domain means. The significance of these pairwise comparisons are indicated in the plot by whether these bars cross the 45-degree background dash-line across the plot. Since none of the three bars cross the dash-line, all pairwise comparisons are significant, as shown in Output 123.8.6.

Example 123.9: Variance Estimate Using the Jackknife Method

(View the complete code for this example (svrex9.sas) in the example repository.)

This example uses the stratified sample from the section "Getting Started: SURVEYREG Procedure" on page 10311 to illustrate how to estimate the variances with replication methods.

As shown in the section "Stratified Sampling" on page 10313, the sample is saved in the SAS data set IceCream. The variable Grade that indicates a student's grade is the stratification variable. The variable Spending contains the dollar amount of each student's average weekly spending for ice cream. The variable Income specifies the household income, in thousands of dollars. The variable Kids indicates how many children are in a student's family. The variable Weight contains sampling weights.

In this example, the procedure uses the jackknife method to estimate the variance, saving the replicate weights that PROC SURVEYREG generates in a SAS data set:

```
title1 'Ice Cream Spending Analysis';
title2 'Use the Jackknife Method to Estimate the Variance';
proc surveyreg data=IceCream
   varmethod=JACKKNIFE(outweights=JKWeights);
   strata Grade;
   class Kids;
   model Spending = Income Kids / solution;
   weight Weight;
run;
```

The VARMETHOD=JACKKNIFE option requests the procedure to estimate the variance by using the jackknife method. The OUTWEIGHTS=*JKWeights* option provides a SAS data set named JKWeights that contains the replicate weights used in the computation.

Output 123.9.1 shows the summary of the data and the variance estimation method. There are a total of 40 replicates generated by the procedure.

Output 123.9.1 Variance Estimation Using the Jackknife Method

Ice Cream Spending Analysis
Use the Jackknife Method to Estimate the Variance

The SURVEYREG Procedure

Regression Analysis for Dependent Variable Spending

Data Summary	
Number of Observations	40
Sum of Weights	4000.0
Weighted Mean of Spending	9.14130
Weighted Sum of Spending	36565.2

Design Summary	
Number of Strata	3

Variance Estimation	
Method	Jackknife
Number of Replicates	40

Output 123.9.2 displays the parameter estimates and their standard errors, as well as the tests of model effects that use the jackknife method.

Output 123.9.2 Variance Estimation Using the Jackknife Method

Tests of Model Effects

Effect	Num DF	F Value	Pr > F
Model	4	110.48	<.0001
Intercept	1	133.30	<.0001
Income	1	289.16	<.0001
Kids	3	0.90	0.4525

Note: The denominator degrees of freedom for the F tests is 37.

Estimated Regression Coefficients

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	-26.086882	2.58771182	-10.08	<.0001
Income	0.776699	0.04567521	17.00	<.0001
Kids 1	0.888631	1.12799263	0.79	0.4358
Kids 2	1.545726	1.25598146	1.23	0.2262
Kids 3	-0.526817	1.42555453	-0.37	0.7138
Kids 4	0.000000	0.00000000	.	.

Note: The degrees of freedom for the t tests is 37.
Matrix X'WX is singular and a generalized inverse was used to solve the normal equations. Estimates are not unique.

Output 123.9.3 prints the first 6 observation in the output data set JKWeights, which contains the replicate weights.

The data set JKWeights contains all the variable in the data set IceCream, in addition to the replicate weights variables named RepWt_1, RepWt_2, . . . , RepWt_40.

For example, the first observation (student) from stratum Grade=7 is deleted to create the first replicate. Therefore, stratum Grade=7 is the donor stratum for the first replicate, and the corresponding replicate weights are saved in the variable RepWt_1.

Because the first observation is deleted in the first replicate, RepWt_1=0 for the first observation. For observations from strata other than the donor stratum Grade=7, their replicate weights remain the same as in the variable Weight, while the rest of the observations in stratum Grade=7 are multiplied by the reciprocal of the corresponding jackknife coefficient, 0.95 for the first replicate.

Output 123.9.3 The Jackknife Replicate Weights for the First 6 Observations

The Jackknife Weights for the First 6 Obs

OBS	Grade	Spending	Income	Kids	Prob	Weight	RepWt_1	RepWt_2	RepWt_3	RepWt_4	RepWt_5
1	7	7	39	2	0.010965	91.200	0.000	96.000	91.200	91.200	96.000
2	7	7	38	1	0.010965	91.200	96.000	0.000	91.200	91.200	96.000
3	8	12	47	1	0.008780	113.889	113.889	113.889	0.000	113.889	113.889
4	9	10	47	4	0.009557	104.636	104.636	104.636	104.636	0.000	104.636
5	7	1	34	4	0.010965	91.200	96.000	96.000	91.200	91.200	0.000
6	7	10	43	2	0.010965	91.200	96.000	96.000	91.200	91.200	96.000

OBS	RepWt_6	RepWt_7	RepWt_8	RepWt_9	RepWt_10	RepWt_11	RepWt_12	RepWt_13	RepWt_14
1	96.000	96.000	91.200	91.200	96.000	96.000	91.200	91.200	96.000
2	96.000	96.000	91.200	91.200	96.000	96.000	91.200	91.200	96.000
3	113.889	113.889	128.125	128.125	113.889	113.889	113.889	128.125	113.889
4	104.636	104.636	104.636	104.636	104.636	104.636	115.100	104.636	104.636
5	96.000	96.000	91.200	91.200	96.000	96.000	91.200	91.200	96.000
6	0.000	96.000	91.200	91.200	96.000	96.000	91.200	91.200	96.000

OBS	RepWt_15	RepWt_16	RepWt_17	RepWt_18	RepWt_19	RepWt_20	RepWt_21	RepWt_22	RepWt_23
1	96.000	96.000	91.200	91.200	91.200	91.200	91.200	91.200	96.000
2	96.000	96.000	91.200	91.200	91.200	91.200	91.200	91.200	96.000
3	113.889	113.889	113.889	128.125	128.125	113.889	113.889	113.889	113.889
4	104.636	104.636	115.100	104.636	104.636	115.100	115.100	115.100	104.636
5	96.000	96.000	91.200	91.200	91.200	91.200	91.200	91.200	96.000
6	96.000	96.000	91.200	91.200	91.200	91.200	91.200	91.200	96.000

OBS	RepWt_24	RepWt_25	RepWt_26	RepWt_27	RepWt_28	RepWt_29	RepWt_30	RepWt_31	RepWt_32
1	96.000	96.000	91.200	91.200	91.200	96.000	96.000	96.000	96.000
2	96.000	96.000	91.200	91.200	91.200	96.000	96.000	96.000	96.000
3	113.889	113.889	113.889	128.125	113.889	113.889	113.889	113.889	113.889
4	104.636	104.636	115.100	104.636	115.100	104.636	104.636	104.636	104.636
5	96.000	96.000	91.200	91.200	91.200	96.000	96.000	96.000	96.000
6	96.000	96.000	91.200	91.200	91.200	96.000	96.000	96.000	96.000

OBS	RepWt_33	RepWt_34	RepWt_35	RepWt_36	RepWt_37	RepWt_38	RepWt_39	RepWt_40
1	91.200	91.200	91.200	96.000	91.200	91.200	96.000	96.000
2	91.200	91.200	91.200	96.000	91.200	91.200	96.000	96.000
3	113.889	128.125	128.125	113.889	113.889	113.889	113.889	113.889
4	115.100	104.636	104.636	104.636	115.100	115.100	104.636	104.636
5	91.200	91.200	91.200	96.000	91.200	91.200	96.000	96.000
6	91.200	91.200	91.200	96.000	91.200	91.200	96.000	96.000

Example 123.10: Plotting a Smoothed Trend Line

(View the complete code for this example (svrex10.sas) in the example repository.)

Statistical graphics can be useful in data visualization and exploration. You can use statements in the SURVEYREG procedure to plot statistical graphics and gain insights about your data.

This example shows how to use PROC SURVEYREG to plot a smoothed trend line, as discussed in Chapter 7 of Lohr (2022b).

To better demonstrate the relationship between the response variable and a covariate in your model, you can fit a smoothed trend line (which is a curved line instead of a straight line) in a bubble plot or heat map of your observed data. The curved line is constructed by splines at your data points. You can control the construction of the splines by using spline effect options in the EFFECT statement. For more information about regression splines and spline bases, see the section "Splines and Spline Bases" on page 441. For more information about using splines, see the section "Using Splines and Knots" on page 10632.

This example uses the data set cancer, which is created in Example 123.7 and based on the National Health and Nutrition Examination Survey I (NHANES I) Epidemiologic Follow-Up Study (NHEFS):

```
data cancer;
   input ObsNo Strata PSU ObservationWt Age Cancer BodyWeight;
   datalines;
   1  3  002  3805   53  1  175
   2  3  002  6107   77  0  175
   3  3  039  2968   50  0  160
   4  3  084  30438  52  0  145
   5  3  007  5081   80  0  127

   ... more lines ...

1016  4  002   2689  40  0  120
1017  4  092  45888  52  0  166
1018  4  035   4347  58  0  156
;
```

To visually examine how aging relates to body weight and determine whether this relationship differs between cancer patients and non-cancer patients, you can construct a spline effect (SplineAge) from the variable Age by using the following EFFECT statement:

```
effect SplineAge = spline(Age);
```

To control how the spline is constructed, you can specify various spline options, which are described in the section "Using Splines and Knots" on page 10632. In the following statements, the KNOTMETHOD=PERCENTILES(5) option requests that internal knots be constructed at five equally spaced percentiles of the variable Age, and the BASIS=BSPLINE option specifies a B-spline basis for the spline expansion. The PLOTS=FIT option requests a plot of the fitted curve by using the spline effect SplineAge against the response variable BodyWeight in the MODEL statement. The DOMAIN statement generates a plot for each subgroup (cancer patients and non-cancer patients). The STRATA, CLUSTER, and WEIGHT statements specify the survey design information.

```
title1 'Plot a Smoothed Trend Line';
ods graphics on;
proc surveyreg data=cancer plot=fit;
   strata Strata;
   cluster PSU;
   weight ObservationWt;
   effect SplineAge = spline(Age /
                 basis=bspline
                 knotmethod=percentiles(5));
   model BodyWeight = SplineAge;
   domain Cancer;
run;
```

As shown in Figure 123.10, the trend line of body weight changes differently with age for non-cancer patients and for cancer patients. For non-cancer patients, body weight tends to have a monotonic decreasing trend as age increases. However, for cancer patients, body weight has a decreasing trend in general, but it increases during the age range of 60 to 70 years. The trend lines give the impression that weight gain between ages 60 and 70 might be related to the risk of having cancer.

Figure 123.10 Trend Lines Comparison

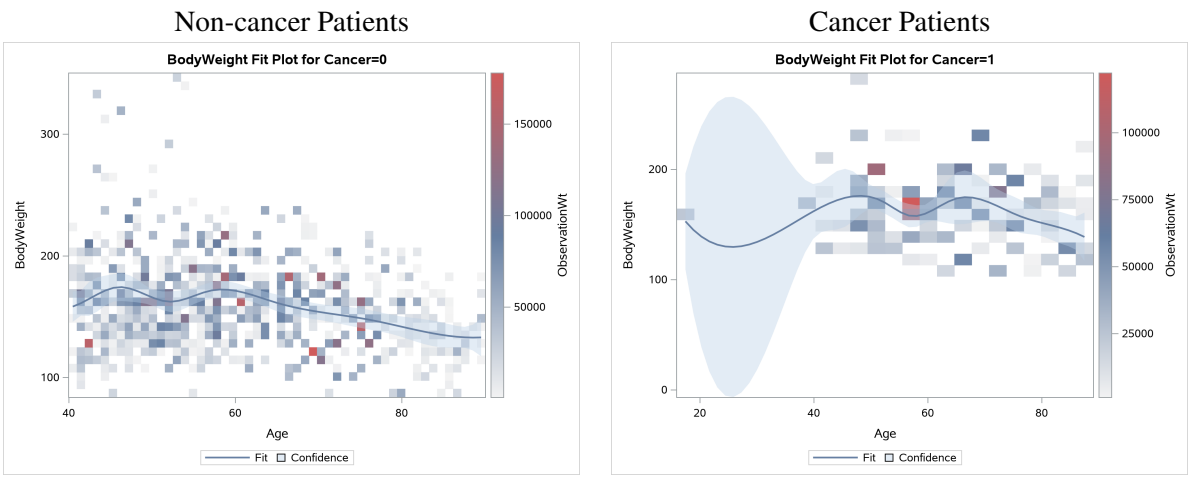

References

Beaumont, J.-F., and Patak, Z. (2012). "On the Generalized Bootstrap for Sample Surveys with Special Attention to Poisson Sampling." *International Statistical Review* 80:127–148.

Brick, J. M., and Kalton, G. (1996). "Handling Missing Data in Survey Research." *Statistical Methods in Medical Research* 5:215–238.

Chang, I.-M., Gelman, R., and Pagano, M. (1982). "Corrected Group Prognostic Curves and Summary Statistics." *Journal of Chronic Diseases* 35:669–674.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Dippo, C. S., Fay, R. E., and Morganstein, D. H. (1984). "Computing Variances from Complex Samples with Replicate Weights." In *Proceedings of the Survey Research Methods Section*, 489–494. Alexandria, VA: American Statistical Association.

Fay, R. E. (1984). "Some Properties of Estimates of Variance Based on Replication Methods." In *Proceedings of the Survey Research Methods Section*, 495–500. Alexandria, VA: American Statistical Association.

Fay, R. E. (1989). "Theory and Application of Replicate Weighting for Variance Calculations." In *Proceedings of the Survey Research Methods Section*, 212–217. Alexandria, VA: American Statistical Association.

Fuller, W. A. (1975). "Regression Analysis for Sample Survey." *Sankhyā, Series C* 37:117–132.

Fuller, W. A. (2009). *Sampling Statistics*. Hoboken, NJ: John Wiley & Sons.

Fuller, W. A., Kennedy, W. J., Schnell, D., Sullivan, G., and Park, H. J. (1989). *PC CARP*. Ames: Iowa State University Statistical Laboratory.

Hidiroglou, M. A., Fuller, W. A., and Hickman, R. D. (1980). *SUPER CARP*. Ames: Iowa State University Statistical Laboratory.

Judkins, D. R. (1990). "Fay's Method for Variance Estimation." *Journal of Official Statistics* 6:223–239.

Kalton, G., and Kasprzyk, D. (1986). "The Treatment of Missing Survey Data." *Survey Methodology* 12:1–16.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Lane, P. W., and Nelder, J. A. (1982). "Analysis of Covariance and Standardization as Instances of Prediction." *Biometrics* 38:613–621.

Lohr, S. L. (2022a). *Sampling: Design and Analysis*. 3rd ed. Boca Raton, FL: CRC Press.

Lohr, S. L. (2022b). *SAS Software Companion for Sampling Design and Analysis, Third Edition*. Boca Raton, FL: CRC Press.

Mashreghi, Z., Haziza, D., and Léger, C. (2016). "A Survey of Bootstrap Methods in Finite Population Sampling." *Statistics Surveys* 10:1–52.

McCarthy, P. J., and Snowden, C. B. (1985). *The Bootstrap and Finite Population Sampling*. Vital and Health Statistics, Series 2, no. 95. DHHS Publication Number 85–1369. Washington, DC: US Government Printing Office. https://www.cdc.gov/nchs/data/series/sr_02/sr02_095.pdf.

Pringle, R. M., and Rayner, A. A. (1971). *Generalized Inverse Matrices with Applications to Statistics*. New York: Hafner Publishing.

Rao, J. N. K., and Shao, J. (1996). "On Balanced Half-Sample Variance Estimation in Stratified Random Sampling." *Journal of the American Statistical Association* 91:343–348.

Rao, J. N. K., and Shao, J. (1999). "Modified Balanced Repeated Replication for Complex Survey Data." *Biometrika* 86:403–415.

Rao, J. N. K., and Wu, C. F. J. (1988). "Resampling Inference with Complex Survey Data." *Journal of the American Statistical Association* 83:231–241.

Rao, J. N. K., Wu, C. F. J., and Yue, K. (1992). "Some Recent Work on Resampling Methods for Complex Surveys." *Survey Methodology* 18:209–217.

Rust, K. (1985). "Variance Estimation for Complex Estimators in Sample Surveys." *Journal of Official Statistics* 1:381–397.

Särndal, C.-E., and Lundström, S. (2005). *Estimation in Surveys with Nonresponse*. Chichester, UK: John Wiley & Sons.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Shao, J., and Tu, D. (1995). *The Jackknife and Bootstrap*. New York: Springer-Verlag.

Sitter, R. R. (1992a). "Comparing Three Bootstrap Methods for Survey Data." *Canadian Journal of Statistics* 20:135–154.

Sitter, R. R. (1992b). "A Resampling Procedure for Complex Survey Data." *Journal of the American Statistical Association* 87:755–765.

Wolter, K. M. (2007). *Introduction to Variance Estimation*. 2nd ed. New York: Springer.

Woodruff, R. S. (1971). "A Simple Method for Approximating the Variance of a Complicated Estimate." *Journal of the American Statistical Association* 66:411–414.

Chapter 124
The SURVEYSELECT Procedure

Contents

Overview: SURVEYSELECT Procedure . . . **10412**
Getting Started: SURVEYSELECT Procedure . . . **10414**
- Simple Random Sampling . . . 10415
- Stratified Sampling . . . 10417
- Stratified Sampling with Control Sorting . . . 10420

Syntax: SURVEYSELECT Procedure . . . **10421**
- PROC SURVEYSELECT Statement . . . 10422
- CONTROL Statement . . . 10445
- FREQ Statement . . . 10446
- ID Statement . . . 10446
- SAMPLINGUNIT | CLUSTER Statement . . . 10446
- SIZE Statement . . . 10448
- STRATA Statement . . . 10449

Details: SURVEYSELECT Procedure . . . **10455**
- Missing Values . . . 10455
- Sorting by CONTROL Variables . . . 10455
- Random Number Generation . . . 10456
- Sample Selection Methods . . . 10457
  - Simple Random Sampling . . . 10458
  - Unrestricted Random Sampling . . . 10458
  - Balanced Bootstrap Sampling . . . 10458
  - Systematic Random Sampling . . . 10459
  - Sequential Random Sampling . . . 10459
  - Bernoulli Sampling . . . 10461
  - Poisson Sampling . . . 10461
  - Sequential Poisson Sampling . . . 10462
  - PPS Sampling without Replacement . . . 10462
  - PPS Sampling with Replacement . . . 10464
  - PPS Systematic Sampling . . . 10464
  - PPS Sequential Sampling . . . 10465
  - Brewer's PPS Method . . . 10466
  - Murthy's PPS Method . . . 10467
  - Sampford's PPS Method . . . 10467
- Sample Size Allocation . . . 10468
  - Proportional Allocation . . . 10468
  - Optimal Allocation . . . 10469

Neyman Allocation 10470
Specifying the Margin of Error 10470
Secondary Input Data Set 10472
Sample Output Data Set 10473
Allocation Output Data Set 10477
Random Assignment Output Data Set 10477
Displayed Output 10478
ODS Table Names 10482
Examples: SURVEYSELECT Procedure **10483**
Example 124.1: Replicated Sampling 10483
Example 124.2: PPS Selection of Two Units per Stratum 10486
Example 124.3: PPS (Dollar-Unit) Sampling 10489
Example 124.4: Proportional Allocation 10492
References **10494**

Overview: SURVEYSELECT Procedure

The SURVEYSELECT procedure provides a variety of methods for selecting probability-based random samples. The procedure can select a simple random sample or can sample according to a complex multistage design that includes stratification, clustering, and unequal probabilities of selection. When you use probability sampling, each unit in the survey population has a known, positive probability of selection. This property of probability sampling avoids selection bias and enables you to use statistical theory to make valid inferences from the sample to the survey population.

To select a sample by using PROC SURVEYSELECT, you provide a SAS data set that contains the sampling frame (the list of units from which the sample is to be selected). The sampling units can be individual observations or groups of observations (clusters). You can also specify the selection method, the sample size or sampling rate, and other selection parameters. PROC SURVEYSELECT selects the sample and produces an output data set that contains the selected units, their selection probabilities, and their sampling weights. To select a sample in multiple stages, you can invoke the procedure separately for each stage of selection by providing the sampling frame and selection parameters for each stage.

PROC SURVEYSELECT provides methods for both equal probability sampling and probability proportional to size (PPS) sampling. In equal probability sampling, each unit in the sampling frame (or stratum) has the same probability of selection. In PPS sampling, each unit's selection probability is proportional to its size measure. For information about probability sampling methods, see Lohr (2010), Kish (1965), Kish (1987), Kalton (1983), and Cochran (1977).

PROC SURVEYSELECT provides the following equal probability sampling methods:

- simple random sampling (without replacement)
- unrestricted random sampling (with replacement)
- systematic random sampling

- sequential random sampling
- balanced bootstrap sampling
- Bernoulli sampling

The procedure also provides Poisson sampling and the following probability proportional to size (PPS) sampling methods:

- PPS sampling without replacement
- PPS sampling with replacement
- PPS systematic sampling
- PPS algorithms for selecting two units per stratum
- sequential PPS sampling with minimum replacement

PROC SURVEYSELECT uses fast, efficient algorithms for sample selection. Thus, it performs well even for large input data sets (sampling frames).

PROC SURVEYSELECT can perform stratified sampling by selecting samples independently within strata, which are nonoverlapping subgroups of the survey population. Stratification controls the distribution of the sample size in the strata. It is widely used in practice toward meeting a variety of survey objectives. For example, you can use stratification to ensure adequate sample sizes for subgroups of interest (including small subgroups), or you can use stratification to improve the precision of overall estimates. When you use a systematic or sequential selection method, PROC SURVEYSELECT can sort by control variables within strata for the additional control of implicit stratification.

For stratified sampling, PROC SURVEYSELECT provides survey design methods to allocate the total sample size among the strata. Available allocation methods include proportional, Neyman, and optimal allocation. Optimal allocation maximizes the estimation precision within the available resources by taking into account stratum sizes, costs, and variances.

PROC SURVEYSELECT provides replicated sampling, where the total sample is composed of a set of replicates, and each replicate is selected in the same way. You can use replicated sampling to study variable nonsampling errors, such as variability in the results obtained by different interviewers. You can also use replicated sampling to estimate standard errors for combined sample estimates and to perform a variety of other resampling and simulation tasks.

Getting Started: SURVEYSELECT Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/sursgs.sas.)

The following examples show how to use PROC SURVEYSELECT to select probability-based random samples. These examples use simulated data for a customer satisfaction survey.

Suppose an internet service provider plans to conduct a customer satisfaction survey by selecting a random sample of customers from all current customers (the survey population). The company plans to interview the selected customers and make inferences about the survey population from the sample data.

The SAS data set Customers contains the sampling frame, which is the list of units in the survey population. The sample of customers will be selected from this sampling frame. The data set Customers is constructed from the company's customer database and contains 13,471 observations (one observation for each customer).

The following PROC PRINT statements display the first 10 observations of the data set Customers and produce the table shown in Figure 124.1:

```
title1 'Customer Satisfaction Survey';
title2 'First 10 Observations';
proc print data=Customers(obs=10);
run;
```

Figure 124.1 Customers Data Set (First 10 Observations)

Customer Satisfaction Survey
First 10 Observations

Obs	CustomerID	State	Type	Usage
1	416-87-4322	AL	New	839
2	288-13-9763	GA	Old	224
3	339-00-8654	GA	Old	2451
4	118-98-0542	GA	New	349
5	421-67-0342	FL	New	562
6	623-18-9201	SC	New	68
7	324-55-0324	FL	Old	137
8	832-90-2397	AL	Old	1563
9	586-45-0178	GA	New	615
10	801-24-5317	SC	New	728

The variable CustomerID contains the unique customer identification number. The variable State contains the state where the customer is located. The value of the variable Type is 'Old' if the customer has subscribed to the service for more than one year; otherwise, the value of Type is 'New'. The variable Usage contains the customer's average monthly service usage in minutes.

The following examples show how to use PROC SURVEYSELECT to implement three different probability sample designs. The first design is simple random sampling without stratification. The second design is a stratified design in which the list of customers is stratified by state and type; the sample is then selected by simple random sampling within strata. The third design is a stratified design that uses control sorting;

customers are ordered by service usage within strata, and the sample is selected by systematic random sampling.

Simple Random Sampling

The following PROC SURVEYSELECT statements select a probability sample of customers from the Customers data set by using simple random sampling:

```
title1 'Customer Satisfaction Survey';
title2 'Simple Random Sampling';
proc surveyselect data=Customers method=srs n=100
                  out=SampleSRS;
run;
```

The PROC SURVEYSELECT statement invokes the procedure. The DATA= option names the SAS data set Customers as the input data set from which to select the sample. The METHOD=SRS option specifies simple random sampling as the sample selection method. In simple random sampling, each sampling unit (observation) has an equal probability of selection, and sampling is performed without replacement. (Without-replacement sampling means that a unit cannot be selected more than once.) The N= option specifies a sample size of 100 customers. The OUT= option stores the sample in the SAS data set named SampleSRS.

Figure 124.2 displays the output from PROC SURVEYSELECT, which summarizes the sample selection. The "Sample Selection Method" table shows that the selection method is simple random sampling. The "Sample Selection Summary" table shows that a sample of 100 customers is selected from the input data set Customers.

When you use simple random sampling (without stratification), all sampling units have the same selection probability. In this example, the selection probability for each customer is 0.007423, which is the sample size (100) divided by the population size (13,471). The sampling weight for each customer in the sample is 134.71, which is the inverse of the selection probability. If you specify the STATS option, PROC SURVEYSELECT includes the selection probabilities and sampling weights in the output data set. For more complex designs, PROC SURVEYSELECT includes this information in the output data set by default.

The "Sample Selection Summary" table also displays the initial seed that PROC SURVEYSELECT uses for random number generation (39647). When you do not specify the SEED= option, PROC SURVEYSELECT uses the time of day from the computer's clock to obtain an initial seed. To reproduce this same sample, you can specify SEED=39647 (for the same input data set and sample selection method).

Figure 124.2 Sample Selection Summary

Customer Satisfaction Survey
Simple Random Sampling

The SURVEYSELECT Procedure

Selection Method	Simple Random Sampling

Figure 124.2 *continued*

Input Data Set	CUSTOMERS
Random Number Seed	39647
Sample Size	100
Selection Probability	0.007423
Sampling Weight	134.71
Output Data Set	SAMPLESRS

The sample of 100 customers is stored in the SAS data set SampleSRS. PROC SURVEYSELECT does not display this output data set. The following PROC PRINT statements display the first 20 observations of SampleSRS:

```
title1 'Customer Satisfaction Survey';
title2 'Sample of 100 Customers, Selected by SRS';
title3 '(First 20 Observations)';
proc print data=SampleSRS(obs=20);
run;
```

Figure 124.3 displays the first 20 observations of the output data set SampleSRS, which contains the sample of customers. This data set includes all variables in the input data set Customers. If you do not want to include all variables, you can use the ID statement to specify which variables to copy from the input data set to the output (sample) data set.

Figure 124.3 Customer Sample (First 20 Observations)

Customer Satisfaction Survey
Sample of 100 Customers, Selected by SRS
(First 20 Observations)

Obs	CustomerID	State	Type	Usage
1	017-27-4096	GA	New	168
2	026-37-3895	AL	New	59
3	038-54-9276	GA	New	785
4	046-40-3131	FL	New	60
5	070-37-6924	GA	New	524
6	100-58-3342	FL	New	302
7	107-61-9029	AL	New	235
8	110-95-0432	FL	New	12
9	112-81-9251	SC	New	347
10	137-33-0478	GA	New	551
11	143-83-4677	AL	New	203
12	147-19-9164	GA	New	172
13	159-51-0606	FL	New	102
14	164-14-7799	GA	Old	388
15	165-05-7323	SC	New	606
16	174-69-3566	AL	Old	111
17	177-69-6934	FL	New	202
18	181-58-3508	AL	Old	261
19	207-41-8446	AL	Old	183
20	207-64-7308	GA	New	193

Stratified Sampling

This example shows how to use PROC SURVEYSELECT to select a stratified random sample. The sampling frame (list of customers) is stratified by the variables State and Type. This stratification divides the sampling frame into nonoverlapping subgroups that are determined by the values of State and Type. Samples are then selected independently within the strata.

PROC SURVEYSELECT requires that the input data set be sorted by the STRATA variables. The following PROC SORT statements sort the Customers data set by the stratification variables State and Type:

```
proc sort data=Customers;
   by State Type;
run;
```

The following PROC FREQ statements display the crosstabulation of the Customers data set by State and Type:

```
title1 'Customer Satisfaction Survey';
title2 'Strata of Customers';
proc freq data=Customers;
   tables State*Type;
run;
```

Figure 124.4 shows the table of State by Type for the 13,471 customers. There are four states and two levels of Type, which form eight strata.

Figure 124.4 Stratification of Customers by State and Type

Customer Satisfaction Survey
Strata of Customers

The FREQ Procedure

Frequency Percent Row Pct Col Pct	Table of State by Type		
		Type	
State	New	Old	Total
AL	1238	706	1944
	9.19	5.24	14.43
	63.68	36.32	
	14.43	14.43	
FL	2170	1370	3540
	16.11	10.17	26.28
	61.30	38.70	
	25.29	28.01	
GA	3488	1940	5428
	25.89	14.40	40.29
	64.26	35.74	
	40.65	39.66	
SC	1684	875	2559
	12.50	6.50	19.00
	65.81	34.19	
	19.63	17.89	
Total	8580	4891	13471
	63.69	36.31	100.00

The following PROC SURVEYSELECT statements select a probability sample of customers from the Customers data set according to the stratified sample design:

```
title1 'Customer Satisfaction Survey';
title2 'Stratified Sampling';
proc surveyselect data=Customers method=srs n=15
                  seed=1953 out=SampleStrata;
   strata State Type;
run;
```

The STRATA statement names the stratification variables State and Type. In the PROC SURVEYSELECT statement, the METHOD=SRS option specifies simple random sampling, and the N= option specifies a sample size of 15 customers in each stratum. If you want to specify different sample sizes for different strata, you can use the N=*SAS-data-set* option to name a secondary data set that contains the stratum sample sizes. The SEED= option specifies 1953 as the initial seed for random number generation.

Figure 124.5 displays the output from PROC SURVEYSELECT, which summarizes the sample selection. A total of 120 customers are selected.

Figure 124.5 Sample Selection Summary

Customer Satisfaction Survey
Stratified Sampling

The SURVEYSELECT Procedure

Selection Method	Simple Random Sampling
Strata Variables	State
	Type

Input Data Set	CUSTOMERS
Random Number Seed	1953
Stratum Sample Size	15
Number of Strata	8
Total Sample Size	120
Output Data Set	SAMPLESTRATA

The following PROC PRINT statements display the first 30 observations of the output data set SampleStrata:

```
title1 'Customer Satisfaction Survey';
title2 'Sample Selected by Stratified Design';
title3 '(First 30 Observations)';
proc print data=SampleStrata(obs=30);
run;
```

Figure 124.6 displays the first 30 observations of the output data set SampleStrata, which contains the sample of 120 customers (15 customers from each of the eight strata). The variable SelectionProb contains the selection probability for each customer in the sample. Because customers are selected with equal probability within strata, the selection probability is the stratum sample size (15) divided by the stratum population size. The selection probabilities differ from stratum to stratum because the stratum population sizes differ. The selection probability for each customer in the first stratum (State='AL' and Type='New') is 0.012116, and the selection probability for customers in the second stratum (State='AL' and Type='Old') is 0.021246.

The variable SamplingWeight contains the sampling weights, which are computed as inverse selection probabilities.

Figure 124.6 Customer Sample (First 30 Observations)

Customer Satisfaction Survey
Sample Selected by Stratified Design
(First 30 Observations)

Obs	State	Type	CustomerID	Usage	SelectionProb	SamplingWeight
1	AL	New	015-57-9903	26	0.012116	82.5333
2	AL	New	052-18-5029	576	0.012116	82.5333
3	AL	New	064-72-0145	88	0.012116	82.5333
4	AL	New	291-22-2497	1221	0.012116	82.5333
5	AL	New	305-62-6833	187	0.012116	82.5333
6	AL	New	309-63-9722	534	0.012116	82.5333
7	AL	New	413-76-0209	435	0.012116	82.5333
8	AL	New	492-18-7867	70	0.012116	82.5333
9	AL	New	508-16-8324	189	0.012116	82.5333
10	AL	New	561-82-0366	392	0.012116	82.5333
11	AL	New	685-24-1718	74	0.012116	82.5333
12	AL	New	800-20-2155	21	0.012116	82.5333
13	AL	New	857-94-2672	77	0.012116	82.5333
14	AL	New	918-29-9618	540	0.012116	82.5333
15	AL	New	963-93-4916	33	0.012116	82.5333
16	AL	Old	182-45-1938	160	0.021246	47.0667
17	AL	Old	210-85-9046	184	0.021246	47.0667
18	AL	Old	211-14-1373	88	0.021246	47.0667
19	AL	Old	229-87-9527	362	0.021246	47.0667
20	AL	Old	239-16-9426	22	0.021246	47.0667
21	AL	Old	283-78-3723	595	0.021246	47.0667
22	AL	Old	293-90-2342	124	0.021246	47.0667
23	AL	Old	360-78-7048	375	0.021246	47.0667
24	AL	Old	432-96-1275	2283	0.021246	47.0667
25	AL	Old	534-79-2367	167	0.021246	47.0667
26	AL	Old	668-77-4832	30	0.021246	47.0667
27	AL	Old	681-88-8208	2133	0.021246	47.0667
28	AL	Old	794-79-7878	1274	0.021246	47.0667
29	AL	Old	954-40-0057	30	0.021246	47.0667
30	AL	Old	954-98-4646	1038	0.021246	47.0667

Stratified Sampling with Control Sorting

This example shows how to use PROC SURVEYSELECT to select a stratified random sample by implementing control sorting and systematic random sampling within strata. The sampling frame (list of customers) is stratified by the variable State, and the sampling units (customers) are sorted by the variables Type and Usage within each stratum (state). Customers are then selected by systematic random sampling within strata. Systematic sampling together with control sorting distributes the sample uniformly over the range of type and usage values within each state.

The following PROC SURVEYSELECT statements select a probability sample of customers from the Customers data set by using this design:

```
title1 'Customer Satisfaction Survey';
title2 'Stratified Sampling with Control Sorting';
proc surveyselect data=Customers method=sys rate=.02
                  seed=1234 out=SampleControl;
   strata State;
   control Type Usage;
run;
```

The STRATA statement names the stratification variable State. The CONTROL statement names the control variables Type and Usage. In the PROC SURVEYSELECT statement, the METHOD=SYS option requests systematic random sampling, and the RATE= option specifies a sampling rate of 2% for each stratum. The SEED= option specifies the initial seed for random number generation.

Figure 124.7 displays the output from PROC SURVEYSELECT, which summarizes the sample selection. A sample of 270 customers is selected by using systematic random sampling within strata that are determined by the variable State. The sampling frame Customers is sorted by the control variables Type and Usage within strata. Sorting is performed by using hierarchic serpentine sorting, which is the default type of sorting. For more information, see the section "Sorting by CONTROL Variables" on page 10455.

By default, the sorted data set replaces the input data set. To store the sorted input data in another data set, you can specify the OUTSORT= option. The output data set SampleControl contains the sample of customers.

Figure 124.7 Sample Selection Summary

Customer Satisfaction Survey
Stratified Sampling with Control Sorting

The SURVEYSELECT Procedure

Selection Method	Systematic Random Sampling
Strata Variable	State
Control Variables	Type
	Usage
Control Sorting	Serpentine

Figure 124.7 *continued*

Input Data Set	CUSTOMERS
Random Number Seed	1234
Stratum Sampling Rate	0.02
Number of Strata	4
Total Sample Size	270
Output Data Set	SAMPLECONTROL

Syntax: SURVEYSELECT Procedure

The following statements are available in the SURVEYSELECT procedure:

PROC SURVEYSELECT *options* ;
  CONTROL *variables* ;
  FREQ *variable* ;
  ID *variables* ;
  SAMPLINGUNIT | CLUSTER *variables* < / *options* > ;
  SIZE *variable* ;
  STRATA *variables* < / *options* > ;

The PROC SURVEYSELECT statement invokes the SURVEYSELECT procedure. Optionally, it identifies input and output data sets. It also specifies the selection method, the sample size, and other sample design parameters. The PROC SURVEYSELECT statement is required.

The SIZE statement identifies the variable that contains the size measures of the sampling units. This statement is required for any probability proportional to size (PPS) selection method unless you specify the PPS option in the SAMPLINGUNIT statement.

The remaining statements are optional. The STRATA statement identifies a variable or set of variables that stratify the input data set. When you specify a STRATA statement, PROC SURVEYSELECT selects samples independently from the strata that are formed by the STRATA variables. The STRATA statement also provides options to allocate the total sample size among the strata.

The SAMPLINGUNIT statement identifies a variable or set of variables that group the input data set observations into sampling units (clusters). Sampling units are nested within strata. When you specify a SAMPLINGUNIT statement, PROC SURVEYSELECT selects clusters instead of individual observations.

The CONTROL statement identifies variables for ordering units within strata. It can be used for systematic and sequential sampling methods. The ID statement identifies variables to copy from the input data set to the output data set of selected units.

The FREQ statement identifies a variable that contains the frequency of occurrence for each observation. The FREQ statement is available only for sample allocation when no sample is selected, which you can request by specifying the ALLOC= and NOSAMPLE options in the STRATA statement.

The following sections describe the PROC SURVEYSELECT statement and then describe the other statements in alphabetical order.

PROC SURVEYSELECT Statement

PROC SURVEYSELECT *options* **;**

The PROC SURVEYSELECT statement invokes the SURVEYSELECT procedure. Optionally, it identifies input and output data sets. If you do not name a DATA= input data set, the procedure selects the sample from the most recently created SAS data set. If you do not name an OUT= output data set to contain the sample of selected units, the procedure still creates an output data set and names it according to the DATA*n* convention.

The PROC SURVEYSELECT statement also specifies the sample selection method, the sample size, and other sample design parameters.

If you do not specify a selection method, PROC SURVEYSELECT uses simple random sampling (METHOD=SRS) by default unless you specify a SIZE statement or the PPS option in the SAMPLINGUNIT statement. If you specify a SIZE statement (or the PPS option), PROC SURVEYSELECT uses probability proportional to size selection without replacement (METHOD=PPS) by default. For more information, see the description of the METHOD= option.

You can use the SAMPSIZE=*n* option to specify the sample size, or you can use the SAMPSIZE=*SAS-data-set* option to name a secondary input data set that contains stratum sample sizes. You must specify a sample size or sampling rate except when you request one of the following: random assignment (GROUPS=); balanced bootstrap selection (METHOD=BALBOOTSTRAP); Poisson sampling (METHOD=POISSON); Brewer's method or Murthy's method, either of which selects two units from each stratum (METHOD=PPS_BREWER or METHOD=PPS_MURTHY); or sample allocation for a specified margin (MARGIN=).

You can provide stratum sample sizes, sampling rates, initial seeds, minimum size measures, maximum size measures, and certainty size measures in a secondary input data set. For more information, see the descriptions of the SAMPSIZE=, SAMPRATE=, SEED=, MINSIZE=, MAXSIZE=, CERTSIZE=, and CERTSIZE=P= options. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT. For more information, see the section "Secondary Input Data Set" on page 10472.

Table 124.1 summarizes the options available in the PROC SURVEYSELECT statement. Descriptions of the options follow in alphabetical order.

Table 124.1 PROC SURVEYSELECT Statement Options

Option	Description
Input and Output Data Sets	
DATA=	Names the input SAS data set
OUT=	Names the output SAS data set that contains the sample
OUTORDER=RANDOM	Randomly orders the observations in the output data set
OUTSORT=	Names an output SAS data set that stores the sorted input data set
Selection Method	
METHOD=	Specifies the sample selection method
Sample Size	
SAMPSIZE=	Specifies the sample size
SELECTALL	Selects all stratum units when the sample size exceeds the total

Table 124.1 *continued*

Option	Description
Sampling Rate	
NMAX=	Specifies the maximum stratum sample size
NMIN=	Specifies the minimum stratum sample size
ROUND=	Specifies the rounding type
SAMPRATE=	Specifies the sampling rate
Replicated Sampling	
REPS=	Specifies the number of replicate samples
Size Measures	
CERTSIZE=	Specifies the certainty size measure
CERTSIZE=P=	Specifies the certainty proportion
MAXSIZE=	Specifies the maximum size measure
MINSIZE=	Specifies the minimum size measure
Control Sorting	
SORT=	Specifies the type of sorting
Random Number Generation	
RANUNI	Requests the RANUNI random number generator
SEED=	Specifies the initial seed
STRATUMSEED=	Controls the stratum initial seeds
Random Assignment	
GROUPS=	Requests random assignment
Displayed Output	
NOPRINT	Suppresses the display of all output
OUT= Data Set Contents	
CERTUNITS=	Includes number of certainty units
JTPROBS	Includes joint probabilities of selection
OUTALL	Includes all observations from the DATA= input data set
OUTHITS	Includes a distinct copy of each selected unit
OUTSEED	Includes the initial seed for each stratum
OUTSIZE	Includes additional design and sampling frame information
STATS	Includes selection probabilities and sampling weights

You can specify the following *options*:

CERTSIZE < =*value* | *SAS-data-set* >

specifies the certainty size value. When you specify this option, PROC SURVEYSELECT selects with certainty all sampling units whose size measures are greater than or equal to the certainty size value. After removing these certainty units, the procedure selects the remainder of the sample by using the method that you specify in the METHOD= option. You provide the size measures by using the SIZE statement.

You can provide a single certainty *value* for the entire sample selection, or you can provide stratum-level certainty values by specifying a *SAS-data-set*. The certainty size values must be positive numbers.

When you specify this option, the OUT= output data set contains a variable named Certain that identifies units that are selected with certainty. The selection probability of each certainty unit is one.

The CERTSIZE= option is available for the following PPS selection methods: METHOD=PPS, METHOD=PPS_SAMPFORD, METHOD=PPS_SYS, METHOD=PPS_WR, and METHOD=SEQ_POISSON. The CERTSIZE= option is not available when you specify a SAMPLINGUNIT statement.

You can provide certainty size values by specifying one of the following forms:

CERTSIZE

indicates that certainty size values are provided in a secondary input data set that you name in another option (for example, the SAMPSIZE=*SAS-data-set* option). This data set should include a variable named _CERTSIZE_ that contains the certainty values. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

CERTSIZE=*value*

specifies a single certainty size *value*, which must be a positive number. If you request a stratified sample design by specifying the STRATA statement, PROC SURVEYSELECT uses the certainty value to determine certainty selections for all strata.

CERTSIZE=*SAS-data-set*

names a *SAS-data-set* that contains stratum-level certainty size values. You should provide the certainty values in the data set variable named _CERTSIZE_. Each observation in this data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary input data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= data set. The order of the stratum groups in the CERTSIZE= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

CERTSIZE=P <=*p* | *SAS-data-set*>

specifies the certainty proportion to use for iterative certainty selection. When you specify this option, PROC SURVEYSELECT selects with certainty the sampling units whose size measure proportions (of the total size) are greater than or equal to the certainty proportion *p*. After removing the selected certainty units, the procedure repeats certainty selection by using the remaining sampling units, the total size of the remaining units, and the certainty proportion *p*. This process continues until no more certainty units are selected. PROC SURVEYSELECT then selects the remainder of the sample by using the method that you specify in the METHOD= option.

You can provide a single certainty proportion *p* for the entire sample, or you can provide stratum-level certainty proportions by specifying a *SAS-data-set*.

The certainty proportions must be positive numbers. You can specify a certainty proportion as a number between 0 and 1. Or you can specify a proportion in percentage form as a number between 1 and 100, which PROC SURVEYSELECT converts to a proportion. The procedure treats the value 1 as 100% instead of 1%.

When you specify this option, the OUT= output data set contains a variable named Certain that identifies units that are selected with certainty. The selection probability of each certainty unit is one.

You use the SIZE statement to provide size measures for the sampling units. The CERTSIZE=P= option is available for METHOD=PPS, METHOD=PPS_SAMPFORD, and METHOD=SEQ_POISSON. This option is not available when you specify a SAMPLINGUNIT statement.

You can provide certainty size proportions by specifying one of the following forms:

CERTSIZE=P
indicates that certainty size proportions are provided in a secondary input data set that you name in another option (for example, the SAMPSIZE=*SAS-data-set* option). You should provide the certainty proportions in the data set variable named _CERTP_. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

CERTSIZE=P=*p*
specifies a single certainty size proportion *p*, which must be a positive number. If you request a stratified sample design by specifying the STRATA statement, PROC SURVEYSELECT uses the certainty proportion *p* to determine certainty selections for all strata.

CERTSIZE=P=*SAS-data-set*
names a *SAS-data-set* that contains stratum-level certainty size proportions. You should provide the certainty proportions in the data set variable named _CERTP_. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary input data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= input data set. The order of the stratum groups in the CERTSIZE=P= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

CERTUNITS=*certain-option* | (*certain-options*)
controls the display and output of the number of certainty units. This option is available when you specify the CERTSIZE= or CERTSIZE=P= option.

You can specify one or both of the following *certain-options*. If you specify both *certain-options*, enclose the values in parentheses after CERTUNITS=.

NOPRINT
suppresses display of the number of certainty units in the "Sample Selection Summary" table. For more information, see the section "Displayed Output" on page 10478.

OUTPUT
includes the number of certainty units in the output data set. For more information, see the section "Sample Output Data Set" on page 10473.

DATA=*SAS-data-set*

names the *SAS-data-set* from which PROC SURVEYSELECT selects the sample. If you omit the DATA= option, the procedure uses the most recently created SAS data set. In sampling terminology, the input data set is the *sampling frame* (the list of units from which the sample is selected).

By default, the procedure uses input data set observations as sampling units and selects a sample of these units. Alternatively, you can use the SAMPLINGUNIT statement to define sampling units as groups of observations (clusters).

GROUPS=*n* | (*values*)| **P=(***values***)< (ROUND=RANDOM) >**

requests random assignment (partitioning) of the input data set observations into groups. You can specify the number of equal-size groups, the number of observations in each group, or the proportion of observations in each group.

When you specify a STRATA statement, PROC SURVEYSELECT performs the random assignment independently for each stratum. Otherwise, PROC SURVEYSELECT performs the random assignment for the entire input data set.

You can specify one of the following forms:

GROUPS=*n* **< (ROUND=RANDOM) >**

requests random assignment of the observations in the input data set (or stratum) to *n* groups. The value of *n* must be a positive integer.

When you specify this form, PROC SURVEYSELECT assigns the same, or nearly the same, number of observations to each group. For example, if the data set contains 100 observations and you specify GROUPS=3, PROC SURVEYSELECT randomly assigns observations to three groups, and the group sizes are 33, 33, and 34 observations.

By default, when the group sizes are not equal, sequential group identification numbers (in the output data set variable GroupID) are designated in order of group size. In the example where GROUPS=3, GroupID=1 and GroupID=2 contain 33 observations, and GroupID=3 contains 34 observations by default. If you specify ROUND=RANDOM, the group identification numbers are designated randomly.

GROUPS=(*values***)**

specifies the number of observations to be assigned to each group. The group size *values* must be positive integers. The sum of the group size values must equal the total number of observations in the input data set (or in the stratum, if you specify a STRATA statement).

The number of groups is the number of group size values that you list. You can separate the values by using blanks or commas, and you must enclose the list of values in parentheses.

GROUPS=P=(*values***)< (ROUND=RANDOM) >**

specifies the proportion of observations to be assigned to each group. The group proportion *values* must be positive numbers between 0 and 1. The sum of the group proportion values must equal 1.

The number of groups is the number of proportion values that you list. You can separate the values by using blanks or commas, and you must enclose the list of values in parentheses.

PROC SURVEYSELECT converts the group proportions to positive, integer-valued group sizes. The procedure multiplies the total number of observations (in the input data set or in the stratum)

by the group proportions to obtain the target group sizes. When a target group size is less than 1, it is always rounded up to 1. By default, PROC SURVEYSELECT computes the integer group sizes by rounding up the target group sizes in order of the fractional parts until the group size total equals the total number of observations. If you specify ROUND=RANDOM, PROC SURVEYSELECT rounds up the target group sizes in random order until the total is met.

The following options in the PROC SURVEYSELECT statement are available when you specify the GROUPS= option: the SEED=, RANUNI, and OUTSEED options, which pertain to random number generation; the REPS= option, which provides independent replicates of the random assignment; the NOPRINT option, which suppresses display of the "Random Assignment" table; and the OUTSIZE option, which adds information to the output data set.

The GROUPS= option does not select a sample, and therefore sample selection options (for example, METHOD= or SAMPSIZE=) are not available when you specify the GROUPS= option. The SAMPLINGUNIT statement is also not available when you specify the GROUPS= option.

When you specify the GROUPS= option, the OUT= output data set includes a variable named GroupID that identifies the group assignment of each observation. When you specify the OUTSIZE option, the output data set also includes the following variables: GroupSize, which is the number of observations in the group; Total, which is the total number of observations (in the data set or in the stratum); and NGroups, which is the number of groups (in the data set or in the stratum). For more information, see the section "Random Assignment Output Data Set" on page 10477.

JTPROBS

includes joint probabilities of selection in the OUT= output data set. This option is available for the following probability proportional to size selection methods: METHOD=PPS, METHOD=PPS_SAMPFORD, and METHOD=PPS_WR. By default, PROC SURVEYSELECT outputs joint selection probabilities for METHOD=PPS_BREWER and METHOD=PPS_MURTHY, which select two units per stratum.

For information about joint selection probabilities for a particular sampling method, see the method description in the section "Sample Selection Methods" on page 10457. For more information about the contents of the output data set, see the section "Sample Output Data Set" on page 10473.

MAXSIZE < =*value* | *SAS-data-set* >

specifies the maximum size value, which PROC SURVEYSELECT uses to adjust size measures before sample selection. When a size measure exceeds the maximum size value, PROC SURVEYSELECT replaces that size measure with the maximum size value.

You can provide a single maximum size *value* for the entire sample selection, or you can provide stratum-level maximum size values by specifying a *SAS-data-set*. The maximum size values must be positive numbers.

You provide size measures by specifying the SIZE statement or by specifying the PPS option in the SAMPLINGUNIT statement.

Unless you specify a SAMPLINGUNIT statement, the OUT= data set includes a variable named AdjustedSize that contains the adjusted size measures.

If you use a SAMPLINGUNIT statement to define sampling units (clusters), PROC SURVEYSELECT adjusts the sampling unit sizes (instead of the observation sizes). If you specify a SIZE statement, the size of a sampling unit is the sum of the size measures of the observations in the unit. If you specify the SAMPLINGUNIT PPS option, the size of a sampling unit is the number of observations in the unit.

When you use a SAMPLINGUNIT statement, the OUT= data set includes a variable named UnitSize that contains the adjusted sampling unit size measures.

You can provide maximum size values by specifying one of the following forms:

MAXSIZE
indicates that maximum size values are provided in a secondary input data set that you name in another option (for example, the SAMPSIZE=*SAS-data-set* option). You should provide the maximum size values in the data set variable named _MAXSIZE_. For more information, see the section "Secondary Input Data Set" on page 10472. You can specify only one secondary input data set in each invocation of PROC SURVEYSELECT.

MAXSIZE=*value*
specifies a single maximum size *value*, which must be a positive number. If you request a stratified sample design by specifying the STRATA statement, PROC SURVEYSELECT uses the value to adjust size measures in all strata.

MAXSIZE=*SAS-data-set*
names a *SAS-data-set* that contains stratum-level maximum size values. You should provide the maximum size values in the data set variable named _MAXSIZE_. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary input data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= data set. The order of the stratum groups in the MAXSIZE= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

METHOD=*method*
M=*method*
specifies the sample selection *method*.

By default, METHOD=PPS if you specify a SIZE statement or the PPS option in the SAMPLINGUNIT statement; otherwise, METHOD=SRS by default.

You can specify one of the following *methods*:

BALBOOTSTRAP
BALBOOT
requests balanced bootstrap selection. This method selects r bootstrap samples of N units with equal probability and with replacement, where N is the total number of sampling units (in the stratum or in the data set) and r is the number of samples (replicates) that you specify in the REPS= option. The bootstrap selection is balanced so that the overall total number of selections is r for each sampling unit. For more information, see the section "Balanced Bootstrap Sampling" on page 10458.

When you request this method, you must specify the number of bootstrap samples r in the REPS= option. The sample size for each bootstrap replicate is fixed at N units; therefore, you cannot specify the SAMPSIZE=, SAMPRATE=, or ALLOC= option together with METHOD=BALBOOTSTRAP.

For balanced bootstrap sampling, the output data set contains an observation for each unit that is selected (in each replicate sample). If you specify the OUTHITS option, the output data set also includes the variable NumberHits, which provides the number of selections in the replicate for each selected unit.

BERNOULLI

requests Bernoulli sampling, which consists of N independent selection trials, each with constant inclusion probability π, where N is the total number of sampling units in the stratum or data set. The sample size is not fixed but is a random variable. For more information, see the section "Bernoulli Sampling" on page 10461.

When you specify this method, you must provide the sampling rate (inclusion probability π) in the SAMPRATE= option. For stratified sampling (which you request with the STRATA statement), you can specify the same sampling rate for each stratum in the SAMPRATE=*value* option. Or you can specify different sampling rates for different strata in the SAMPRATE=(*values*) or SAMPRATE=*SAS-data-set* option.

Because Bernoulli sampling is based on a specified inclusion probability instead of a fixed sample size, METHOD=BERNOULLI does not use the SAMPSIZE= option. Also, the ALLOC= option in the STRATA statement (which allocates the total sample size among strata) is not available with METHOD=BERNOULLI.

POISSON

requests Poisson sampling. A generalization of Bernoulli sampling, Poisson sampling consists of N independent selection trials with a separate inclusion probability specified for each unit, where N is the total number of sampling units in the stratum or data set. The sample size is not fixed but is a random variable. For more information, see the section "Poisson Sampling" on page 10461. For a fixed-sample-size modification of Poisson sampling, see METHOD=SEQ_POISSON.

You must provide inclusion probabilities for Poisson sampling in the SIZE variable. The probability values should be between 0 and 1. If a value of the SIZE variable is missing, nonpositive, or greater than 1, PROC SURVEYSELECT omits the observation from sample selection.

Because Poisson sampling is based on specified inclusion probabilities instead of on a fixed sample size, you cannot specify the SAMPSIZE= option together with METHOD=POISSON. Similarly, you cannot specify the ALLOC= option (which allocates the total sample size among strata) together with METHOD=POISSON.

The SAMPLINGUNIT statement is not available when you specify METHOD=POISSON.

When you specify the SAMPRATE= option for METHOD=POISSON but do not specify a SIZE statement, PROC SURVEYSELECT uses METHOD=BERNOULLI.

PPS <(*method-option*)>

requests selection with probability proportional to size and without replacement. PROC SURVEYSELECT performs PPS selection by using the Hanurav-Vijayan algorithm. For more information, see the section "PPS Sampling without Replacement" on page 10462. When you specify this method, you must name a size measure variable in the SIZE statement or specify the PPS option in the SAMPLINGUNIT statement.

You can specify the following *method-option*:

OUTSORT=*SAS-data-set*

names an output data set to store the input data that are sorted by size measures within strata. By default, PROC SURVEYSELECT does not save this data set. (Sorting by size measures is part of the Hanurav-Vijayan algorithm for PPS selection. For more information, see the section "PPS Sampling without Replacement" on page 10462.) This option is not available when you specify the SAMPLINGUNIT statement.

PPS_BREWER

BREWER

requests selection according to Brewer's method. Brewer's method selects two units from each stratum with probability proportional to size and without replacement. For more information, see the section "Brewer's PPS Method" on page 10466. When you specify this method, you must name a size measure variable in the SIZE statement or specify the PPS option in the SAMPLINGUNIT statement. You do not need to specify the sample size in the SAMPSIZE= option because Brewer's method selects two units from each stratum.

PPS_MURTHY

MURTHY

requests selection according to Murthy's method. Murthy's method selects two units from each stratum with probability proportional to size and without replacement. For more information, see the section "Murthy's PPS Method" on page 10467. When you specify this method, you must name a size measure variable in the SIZE statement or specify the PPS option in the SAMPLINGUNIT statement. You do not need to specify the sample size in the SAMPSIZE= option because Murthy's method selects two units from each stratum.

PPS_SAMPFORD

SAMPFORD

requests selection according to Sampford's method. Sampford's method selects units with probability proportional to size and without replacement. For more information, see the section "Sampford's PPS Method" on page 10467. When you specify this method, you must name a size measure variable in the SIZE statement or specify the PPS option in the SAMPLINGUNIT statement.

PPS_SEQ

CHROMY

requests sequential selection with probability proportional to size and with minimum replacement. This method is also known as Chromy's method. For more information, see the section "PPS Sequential Sampling" on page 10465. When you specify this method, you must name a size measure variable in the SIZE statement or specify the PPS option in the SAMPLINGUNIT statement.

PPS_SYS < (*method-options*) >

requests systematic selection with probability proportional to size. For more information, see the section "PPS Systematic Sampling" on page 10464. When you specify this method, you must provide size measures by specifying the SIZE statement or the PPS option in the SAMPLINGUNIT statement.

You can specify the following *method-options*:

DETAILS

displays the random start and the systematic interval in the "Sample Selection Summary" table when the design does not include strata or replicates. For more information, see the section "Displayed Output" on page 10478.

INTERVAL=*value*

specifies the interval *value* for PPS systematic selection. The interval value must be a positive number. It must not exceed the total of the size measures in the data set (or in each stratum if you specify a STRATA statement). By default, the systematic interval is the ratio of the size measure total to the sample size (which you provide in the SAMPSIZE= option). For more information, see the section "PPS Systematic Sampling" on page 10464.

You cannot use the INTERVAL= *method-option* when you specify a sample size in the SAMPSIZE= option or when you specify the ALLOC= option, which allocates the total sample size among strata.

START=*value*

specifies the starting *value* for PPS systematic selection. The starting value must be a positive number that is less than the systematic interval. By default, PROC SURVEYSELECT randomly determines a starting point in the systematic interval. For more information, see the section "PPS Systematic Sampling" on page 10464.

When you use this option to specify a systematic starting point (instead of allowing the procedure to randomly determine the starting point), the following options for random number generation have no effect: SEED=, RANUNI, and OUTSEED. You cannot use the REPS= option when you specify the START= *method-option*.

When the starting value that you provide is not randomly determined, the resulting selection is not a probability-based sample.

PPS_WR

requests selection with probability proportional to size and with replacement. For more information, see the section "PPS Sampling with Replacement" on page 10464. When you specify this method, you must name a size measure variable in the SIZE statement or specify the PPS option in the SAMPLINGUNIT statement.

SEQ

CHROMY

requests sequential selection according to Chromy's method. If you specify this method and do not specify a SIZE statement (or the PPS option in the SAMPLINGUNIT statement), PROC SURVEYSELECT uses sequential zoned selection with equal probability and without replacement. For more information, see the section "Sequential Random Sampling" on page 10459.

If you specify METHOD=SEQ and also specify a SIZE statement (or the PPS option in the SAMPLINGUNIT statement), PROC SURVEYSELECT uses METHOD=PPS_SEQ, which is sequential selection with probability proportional to size and with minimum replacement. For more information, see the section "PPS Sequential Sampling" on page 10465.

SEQ_POISSON

requests sequential Poisson sampling, which is a fixed-sample-size modification of Poisson sampling (METHOD=POISSON). For more information, see the section "Sequential Poisson Sampling" on page 10462.

When you request this method, you must provide size measures in the SIZE variable and you must specify the sample size in the SAMPSIZE= option.

If you request this method, you cannot specify a SAMPLINGUNIT statement.

SRS

requests simple random sampling, which is selection with equal probability and without replacement. For more information, see the section "Simple Random Sampling" on page 10458. METHOD=SRS is the default selection method if you do not specify the METHOD= option and also do not specify a SIZE statement (or the PPS option in the SAMPLINGUNIT statement).

SYS < (*method-options*) >

requests systematic random sampling. If you specify this method and do not specify a SIZE statement (or the PPS option in the SAMPLINGUNIT statement), PROC SURVEYSELECT uses systematic random sampling with equal probability. For more information, see the section "Systematic Random Sampling" on page 10459.

If you specify this method and also specify a SIZE statement (or the PPS option in the SAMPLINGUNIT statement), PROC SURVEYSELECT uses systematic random sampling with probability proportional to size (METHOD=PPS_SYS). For more information, see the section "PPS Systematic Sampling" on page 10464.

You can specify the following *method-options*:

DETAILS

displays the random start and the systematic interval in the "Sample Selection Summary" table when the design does not include strata or replicates. For more information, see the section "Displayed Output" on page 10478.

INTERVAL=*value*

specifies the interval for systematic random sampling. The interval *value* must be a positive number and must not exceed the number of sampling units in the data set (or the number of units in each stratum, if you specify a STRATA statement).

By default, PROC SURVEYSELECT determines the systematic interval from the sampling rate or sample size that you provide in the SAMPRATE= or SAMPSIZE= option, respectively. When you specify the sampling rate, PROC SURVEYSELECT computes the systematic interval as the inverse of the sampling rate. When you specify the sample size, the procedure computes the interval as the ratio of the number of sampling units to the sample size. For more information, see the section "Systematic Random Sampling" on page 10459.

You cannot use the INTERVAL= *method-option* when you specify the SAMPSIZE= option, the SAMPRATE= option, or the ALLOC= option (which allocates the total sample size among strata).

START=*value*

specifies the starting *value* for systematic selection. The starting value must be a positive number that is less than the systematic interval. By default, PROC SURVEYSELECT randomly determines a starting point in the systematic interval. For more information, see the section "Systematic Random Sampling" on page 10459.

When you use this option to specify a systematic starting point (instead of allowing the procedure to randomly determine the starting point), the following options for random number generation have no effect: SEED=, RANUNI, and OUTSEED. You cannot use the REPS= option when you specify the START= *method-option*.

When the starting value that you provide is not randomly determined, the resulting selection is not a probability-based sample.

URS

requests unrestricted random sampling, which is selection with equal probability and with replacement. For more information, see the section "Unrestricted Random Sampling" on page 10458.

MINSIZE < =*value* | *SAS-data-set* >

specifies the minimum size measure, which PROC SURVEYSELECT uses to adjust size measures before sample selection. When a size measure is less than the minimum size value, PROC SURVEYSELECT replaces that size measure with the minimum size value.

You can provide a single minimum size *value* for the entire sample selection, or you can provide stratum-level minimum size values by specifying a *SAS-data-set*. The minimum size values must be positive numbers.

You provide size measures by specifying the SIZE statement or by specifying the PPS option in the SAMPLINGUNIT statement.

Unless you specify a SAMPLINGUNIT statement, the OUT= data set includes a variable named AdjustedSize that contains the adjusted size measures.

If you use a SAMPLINGUNIT statement to define sampling units (clusters), PROC SURVEYSELECT adjusts the sampling unit sizes (not the observation sizes). If you specify a SIZE statement, the size of a sampling unit is the sum of the size measures of the observations in the unit. If you specify the SAMPLINGUNIT PPS option, the size of a sampling unit is the number of observations in the unit.

When you use a SAMPLINGUNIT statement, the OUT= data set includes a variable named UnitSize that contains the adjusted sampling unit size measures.

You can provide minimum size values by specifying one of the following forms:

MINSIZE

indicates that minimum size values are provided in a secondary input data set that you name in another option (for example, the SAMPSIZE=*SAS-data-set* option). You should provide the minimum size values in the data set variable named _MINSIZE_. For more information, see the section "Secondary Input Data Set" on page 10472. You can specify only one secondary input data set in each invocation of PROC SURVEYSELECT.

MINSIZE=*value*

specifies a single minimum size *value*, which must be a positive number. If you request a stratified sample design by specifying the STRATA statement, PROC SURVEYSELECT uses the minimum value to adjust size measures in all strata.

MINSIZE=*SAS-data-set*

names a *SAS-data-set* that contains stratum-level minimum size values. You should provide the minimum size values in the data set variable named _MINSIZE_. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary input data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= input data set. The order of the stratum groups in the MINSIZE= data set must match the order of the groups in the DATA= input data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

NMAX=*n*

specifies the maximum stratum sample size *n* for the SAMPRATE= option. When you specify the SAMPRATE= option, PROC SURVEYSELECT calculates the stratum sample size by multiplying the total number of units in the stratum by the specified sampling rate. If this sample size is greater than the value NMAX=*n*, PROC SURVEYSELECT selects only *n* units.

The maximum sample size *n* must be a positive integer. The NMAX= option is available only with the SAMPRATE= option, which you can specify for equal probability selection methods (METHOD=SRS, METHOD=URS, METHOD=SYS, and METHOD=SEQ). The NMAX= option is not available with METHOD=BERNOULLI, where the SAMPRATE= option specifies the constant inclusion probability.

NMIN=*n*

specifies the minimum stratum sample size *n* for the SAMPRATE= option. When you specify the SAMPRATE= option, PROC SURVEYSELECT calculates the stratum sample size by multiplying the total number of units in the stratum by the specified sampling rate. If this sample size is less than the value NMIN=*n*, PROC SURVEYSELECT selects *n* units.

The minimum sample size *n* must be a positive integer. The NMIN= option is available only with the SAMPRATE= option, which you can specify for equal probability selection methods (METHOD=SRS, METHOD=URS, METHOD=SYS, and METHOD=SEQ). The NMIN= option is not available with METHOD=BERNOULLI, where the SAMPRATE= option specifies the constant inclusion probability.

NOPRINT

suppresses the display of all output. You can use the NOPRINT option when you want only to create an output data set. This option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OUT=*SAS-data-set*

names the output data set. If you omit the OUT= option, the data set is named DATA*n*, where *n* is the smallest integer that makes the name unique. If you request sample selection by specifying the METHOD= option, the output data set contains the observations that are selected for the sample. If you request sample allocation without sample selection by specifying the ALLOC= and NOSAMPLE options in the STRATA statement, the output data set contains the allocated sample sizes. If you request random assignment by specifying the GROUPS= option, the output data set contains all observations in the input data set together with their assigned group identification.

When PROC SURVEYSELECT selects a sample, the output data set contains the units that are selected, sample design information, and selection statistics. You can specify options that control the information to include in the output data set. For more information, see the descriptions of the following options: JTPROBS, OUTALL, OUTHITS, OUTSEED, OUTSIZE, and STATS. For more information, see the section "Sample Output Data Set" on page 10473.

By default, the sample output data set contains only those units that are selected for the sample. To include all observations from the input data set in the output data set, use the OUTALL option.

By default, the sample output data set includes one copy of each selected unit, even when a unit is selected more than once, which can occur when you use with-replacement or with-minimum-replacement selection methods. For with-replacement or with-minimum-replacement selection methods, the output data set includes a variable NumberHits that records the number of hits (selections) for each unit. To include a distinct copy of each selection in the output data set when the same unit is selected more than once, use the OUTHITS option.

When you specify the ALLOC= and NOSAMPLE options in the STRATA statement, PROC SURVEYSELECT allocates the total sample size among the strata but does not select a sample. In this case, the OUT= data set contains the allocated sample sizes. For more information, see the section "Allocation Output Data Set" on page 10477.

When you specify the GROUPS= option, PROC SURVEYSELECT randomly assigns observations to groups; it does not select a sample. In this case, the OUT= data set contains all observations from the input data set and includes a variable named GroupID that identifies group assignments. For more information, see the section "Random Assignment Output Data Set" on page 10477.

OUTALL < (ZEROSTRATA) >

includes all observations from the sampling frame in the OUT= output data set. By default, the output data set includes only those units selected for the sample. When you specify the OUTALL option, the output data set includes all observations in the sampling frame along with a variable (Selected) that indicates each observation's selection status. The value of Selected is 1 for an observation that is selected or 0 for an observation that is not selected. For information about the contents of the output data set, see the section "Sample Output Data Set" on page 10473.

The OUTALL option is available for equal probability selection methods (METHOD=SRS, METHOD=URS, METHOD=SYS, METHOD=SEQ, and METHOD=BERNOULLI). and for METHOD=POISSON.

If you specify a sample size of 0 for a stratum, PROC SURVEYSELECT omits this stratum from the sampling frame. By default, PROC SURVEYSELECT also omits this stratum from the output data set when you specify the OUTALL option. You can specify the OUTALL(ZEROSTRATA) option to include strata that have sample sizes of 0 in the output data set. For more information, see the description of the SAMPSIZE= option.

OUTHITS

includes a distinct copy of each selected unit in the OUT= output data set when the same sampling unit is selected more than once. By default, the output data set contains a single copy of each unit selected, even when a unit is selected more than once, and the variable NumberHits records the number of hits (selections) for each unit. If you specify the OUTHITS option, the output data set contains *m* copies of a sampling unit for which NumberHits is *m*; for example, the output data set contains three copies of a unit that is selected three times (NumberHits is 3).

A sampling unit can be selected more than once by with-replacement and with-minimum-replacement selection methods, which include METHOD=URS, METHOD=BALBOOTSTRAP, METHOD=PPS_WR, METHOD=PPS_SYS, and METHOD=PPS_SEQ. The OUTHITS option is available for these selection methods.

For information about the contents of the output data set, see the section "Sample Output Data Set" on page 10473.

OUTORDER=RANDOM < (*option*) >

OUTRANDOM < (*option*) >

randomly orders the sample observations in the OUT= output data set. If you specify a STRATA statement, the sample observations are randomly ordered within stratum groups. If you specify the REPS= option, the sample observations are randomly ordered within sample replicates (nested within stratum groups). This option does not affect the selection of the sample; it randomly orders the selected observations after sample selection is complete.

By default for most sample selection methods, the order of the sample observations in the OUT= data set corresponds to the order in the DATA= input data set. If you specify a CONTROL statement to sort the input data set for systematic selection methods, the order in the output data set is the same as the sorted order of the input data set. If you specify a sequential selection method, the order in the output data set is the selection order (which begins at a randomly chosen sampling unit). By default for METHOD=PPS, the order of the sample observations corresponds to the ascending order of the size measures.

The OUTORDER=RANDOM option is not available when you specify the GROUPS= option, a SAMPLINGUNIT statement, or the NOSAMPLE option in the STRATA statement.

You can specify the following *option*:

SEED=*value*

specifies the initial random number seed for randomly ordering the sample observations. To initialize random number generation, *value* must be a positive integer. If you do not specify this option or if you specify a *value* that is negative or 0, PROC SURVEYSELECT uses the existing random number stream that is used for sample selection. For more information, see the SEED= option and the section "Random Number Generation" on page 10456.

OUTSEED

includes the initial seed for each stratum in the OUT= output data set. The variable InitialSeed contains the stratum initial seeds. For information about the contents of the output data set, see the section "Sample Output Data Set" on page 10473. The OUTSEED option is not available when you specify the STRATUMSEED=NONE option for a stratified sample.

To reproduce the same sample for any stratum in a subsequent execution of PROC SURVEYSELECT, you can specify the same stratum initial seed in the SEED=*SAS-data-set* option together with the same

sample selection parameters. For more information, see the section "Random Number Generation" on page 10456.

The "Sample Selection Summary" table displays the initial random number seed for the entire sample selection, which is the initial seed for the first stratum when the design is stratified. To reproduce the entire sample, you can specify this same seed value in the SEED= option, along with the same sample selection parameters.

OUTSIZE

includes additional design and sampling frame information in the OUT= output data set.

If you use a STRATA statement, the OUTSIZE option provides stratum-level values in the output data set. Otherwise, the OUTSIZE option provides overall values.

The OUTSIZE option includes the sample size or sampling rate in the output data set, depending on whether you specify the SAMPSIZE= option or the SAMPRATE= option, respectively. For PPS selection methods, the OUTSIZE option includes the total size measure in the output data set. If you do not provide size measures, or if you specify a SAMPLINGUNIT statement, the OUTSIZE option includes the total number of sampling units in the output data set.

If you request size measure adjustment or certainty selection, the OUTSIZE option includes the following information in the output data set: the minimum size measure if you specify the MINSIZE= option, the maximum size measure if you specify the MAXSIZE= option, the certainty size measure if you specify the CERTSIZE= option, and the certainty proportion if you specify the CERTSIZE=P= option.

For METHOD=BERNOULLI, the OUTSIZE option includes the following information in the output data set: total number of sampling units, selection probability (sampling rate), expected sample size, and actual sample size. See the section "Bernoulli Sampling" on page 10461 for descriptions of these statistics.

For more information about the contents of the output data set, see the section "Sample Output Data Set" on page 10473.

If you specify the GROUPS= option for random assignment, the OUTSIZE option adds the following information to the output data set: total number of units, number of groups, and number of units in the group. For more information, see the section "Random Assignment Output Data Set" on page 10477.

OUTSORT=*SAS-data-set*

names an output data set to store the sorted input data set. This option is available when you specify a CONTROL statement to sort the DATA= input data set for systematic or sequential selection methods (METHOD=SYS, METHOD=PPS_SYS, METHOD=SEQ, and METHOD=PPS_SEQ).

If you specify CONTROL variables but do not name an output data set in the OUTSORT= option, the sorted data set replaces the input data set.

RANUNI

requests uniform random number generation by the method of Fishman and Moore (1982), which is the random number generator that the RANUNI function provides. For more information, see the section "Random Number Generation" on page 10456. For information about the RANUNI function, see *SAS Functions and CALL Routines: Reference*.

By default, PROC SURVEYSELECT uses the Mersenne twister random number generator (Matsumoto and Nishimura 1998). The Mersenne twister random number generator has a very long period and

good statistical properties. This is the random number generator that the RAND function provides for the uniform distribution. For information about the RAND function, see *SAS Functions and CALL Routines: Reference*.

If you use the RANUNI option to select a sample, you can reproduce this same sample by specifying the same SEED= value together with the RANUNI option (for the same input data set and sample selection parameters).

In releases before SAS/STAT 12.1, PROC SURVEYSELECT uses the RANUNI random number generator by default. To reproduce samples from releases before SAS/STAT 12.1, you can specify the same SEED= value together with the RANUNI option (for the same input data set and sample selection parameters).

REPS=*nreps* **<(***option***)>**

requests replicated sampling and specifies the number of replicate samples. The value of *nreps* must be a positive integer.

When you specify this option, PROC SURVEYSELECT selects *nreps* independent replicate samples. Each replicate sample is selected by using the same sample size (or sampling rate) and design parameters that you specify.

By default, the variable named Replicate in the OUT= data set contains replicate identification numbers for the sample observations. You can specify a different name for the replicate identification variable in the REPNAME= suboption.

You can use replicated sampling to provide a simple method of variance estimation for any form of statistic and to evaluate variable nonsampling errors such as interviewer differences. For more information, see Lohr (2010), Wolter (2007), Kish (1965), Kish (1987), and Kalton (1983). You can also use the REPS= option to perform a variety of other resampling and simulation tasks. For more information, see Cassell (2007).

You can specify the following *option*:

REPNAME=*name*

specifies the *name* of the replicate identification variable in the OUT= data set. By default, the variable name is Replicate. For more information, see the section "Sample Output Data Set" on page 10473.

ROUND=*type*

specifies the type of rounding to use when the sampling rate is converted to a positive, integer-valued sample size. This option is available when you specify the SAMPRATE= option for one of the following equal probability selection methods: METHOD=SRS, METHOD=URS, METHOD=SYS, and METHOD=SEQ.

For these selection methods, PROC SURVEYSELECT converts the sampling rate that you specify in the SAMPRATE= option to an integer-valued sample size before selecting the sample. PROC SURVEYSELECT multiplies the total number of units (in the stratum or in the data set) by the sampling rate to obtain the target sample size. If the target sample size is not an integer, the procedure rounds this number to a positive integer value. By default, PROC SURVEYSELECT always rounds the target sample size up to the nearest integer (ROUND=UP).

To provide positive selection probabilities for all sampling units, all stratum sample sizes must be greater than 0. Therefore, when a target sample size is less than 1, it is always rounded up to 1 (instead of down to 0), regardless of the ROUND=*type*.

You can specify one of the following *types*:

ALTERNATE
alternates the rounding direction (up or down) by strata. This option rounds up for the first stratum, down for the second stratum, up for the third stratum, and so on, where the strata are processed in the order in which they appear in the input data set. The alternating sequence skips strata for which the target sample size is an integer (and therefore does not require rounding). This option has no effect unless you request stratified sampling by specifying a STRATA statement.

DOWN
FLOOR
rounds down to the largest integer that does not exceed the target sample size.

NEAREST < (HALF=DOWN) >
rounds the target sample size to the nearest integer. This option rounds up when the fractional part of the target sample size is greater than 0.5 and rounds down when the fractional part is less than 0.5. By default, the sample size is rounded up when the fractional part is exactly 0.5. When you specify ROUND=NEAREST(HALF=DOWN), the sample size is rounded down when the fractional part is 0.5.

RANDOM
determines the rounding direction (up or down) randomly. Each rounding direction is equally likely (with probability of 0.5).

UP
CEILING
rounds up to the smallest integer that is not less than the target sample size. This option is the default.

SAMPRATE=*value* | **(***values***)**| *SAS-data-set*
RATE=*value* | **(***values***)**| *SAS-data-set*
specifies the sampling rate, which is the proportion of units to select for the sample. You can provide a single sampling rate *value* for the entire sample selection, or you can provide stratum sampling rates by specifying *values* or a *SAS-data-set*.

The sampling rate value must be a positive number. The stratum sampling rate values and the stratum sampling rates that you provide in the *SAS-data-set* must be nonnegative numbers. You can specify a sampling rate as a number between 0 and 1. Or you can specify a rate in percentage form as a number between 1 and 100, which PROC SURVEYSELECT converts to a proportion. The procedure treats the value 1 as 100% instead of 1%.

This option is available for the equal probability selection methods, as follows:

- For METHOD=SYS (systematic random sampling), PROC SURVEYSELECT computes the selection interval as the inverse of the sampling rate. For more information, see the section "Systematic Random Sampling" on page 10459.
- For METHOD=BERNOULLI (Bernoulli sampling), the procedure uses the sampling rate as the inclusion probability. For more information, see the section "Bernoulli Sampling" on page 10461.

- For the other equal probability selection methods (METHOD=SRS, METHOD=URS, METHOD=SYS, and METHOD=SEQ), the procedure converts the sampling rate to an integer-valued sample size before selecting the sample.

To convert the sampling rate to an integer-valued sample size, PROC SURVEYSELECT multiplies the total number of units (in the stratum or data set) by the sampling rate that you specify to obtain the target sample size. If this number is not an integer, the target sample size is rounded to a positive integer. By default, PROC SURVEYSELECT always rounds the target sample size up to the nearest integer. You can specify other types of rounding by using the ROUND= option.

You cannot specify the SAMPRATE= option together with the SAMPSIZE= option.

You can provide sampling rates by specifying one of the following forms:

SAMPRATE=*value*

RATE=*value*

specifies a single sampling rate *value*, which must be a positive number. If you request a stratified sample design by specifying the STRATA statement, PROC SURVEYSELECT uses the rate value for all strata.

SAMPRATE=(*values***)**

RATE=(*values***)**

specifies a list of stratum sampling rate *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. The number of stratum sampling rate values should equal the number of strata in the input data set.

The order of the stratum sampling rate values must match the order of the stratum groups in the DATA= input data set. When you specify a list of values, the input data set must be sorted by the STRATA variables in ascending order; you cannot use the DESCENDING or NOTSORTED option in the STRATA statement.

The stratum sampling rate values must be nonnegative numbers. If you specify a stratum sampling rate of 0, PROC SURVEYSELECT does not select a sample from the stratum. This has the effect of subsetting the input data set before sample selection; the stratum that you omit is not included in the sampling frame or represented in the sample.

SAMPRATE=*SAS-data-set*

RATE=*SAS-data-set*

names a *SAS-data-set* that contains stratum sampling rates. You should provide the sampling rates in the data set variable named _RATE_. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary input data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= input data set. The order of the stratum groups in the SAMPRATE= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

The stratum sampling rates must be nonnegative numbers. If you specify a stratum sampling rate of 0, PROC SURVEYSELECT does not select a sample from the stratum. This has the effect of

subsetting the input data set before sample selection; the stratum that you omit is not included in the sampling frame or represented in the sample.

SAMPSIZE=*n* |(*values*)| *SAS-data-set*

N=*n* | (*values*)| *SAS-data-set*

specifies the sample size, which is the number of units to select for the sample. You can provide a single sample size *n* for the entire sample selection, or you can provide stratum sample sizes by specifying *values* or a *SAS-data-set*.

The value of *n* must be a positive integer. The stratum sample size values and the stratum sample sizes that you provide in the *SAS-data-set* must be nonnegative numbers. For selection methods that select without replacement, the sample size must not exceed the total number of units in the data set (or the number of units in the stratum, if you specify a STRATA statement).

This option specifies the number of sampling units to select. If you do not specify a SAMPLINGUNIT statement, PROC SURVEYSELECT defines sampling units as observations and selects the number of observations that you specify. If you specify a SAMPLINGUNIT statement, PROC SURVEYSELECT defines sampling units as groups of observations (clusters) and selects the number of clusters that you specify.

If you specify SAMPSIZE=*n* and the ALLOC= option in the STRATA statement, PROC SURVEYSELECT allocates the sample size *n* among the strata according to the allocation method that you request. For more information, see the section "Sample Size Allocation" on page 10468. You cannot specify SAMPSIZE=*values* or SAMPSIZE=*SAS-data-set* when you use the ALLOC= option. You cannot specify SAMPSIZE= with the MARGIN= option, which determines stratum sample sizes that provide the specified margin of error. For more information, see the section "Specifying the Margin of Error" on page 10470.

You cannot specify both the SAMPSIZE= option and the SAMPRATE= option.

You can provide sample size values by specifying one of the following forms:

SAMPSIZE=*n*

N=*n*

specifies a single sample size value *n*, which must be a positive integer. If you request a stratified sample design, PROC SURVEYSELECT selects *n* units from each stratum (unless you also specify the ALLOC= option in the STRATA statement, which allocates the total sample size among the strata).

For methods that select without replacement, the sample size *n* must not exceed the number of units in the stratum unless you also specify the SELECTALL option. If you specify the SELECTALL option, PROC SURVEYSELECT selects all stratum units when the stratum sample size exceeds the total number of units in the stratum.

SAMPSIZE=(*values***)**

N=(*values***)**

specifies a list of stratum sample size *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. The number of sample size values must equal the number of strata in the input data set.

The order of the stratum sample size values must match the order of the stratum groups in the DATA= input data set. When you specify a list of values, the input data set must be sorted by

the STRATA variables in ascending order; you cannot use the DESCENDING or NOTSORTED option in the STRATA statement.

The values of the stratum sample sizes must be nonnegative numbers. If you specify a stratum sample size of 0, PROC SURVEYSELECT does not select a sample from the stratum. This has the effect of subsetting the input data set before sample selection; the stratum that you omit is not included in the sampling frame or represented in the sample.

SAMPSIZE=*SAS-data-set*

N=*SAS-data-set*

names a *SAS-data-set* that contains stratum sample sizes. You should provide the sample sizes in the data set variable named _NSIZE_ or SampleSize. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= input data set. The order of the stratum groups in the SAMPSIZE= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

The stratum sample sizes must be nonnegative numbers. If you specify a stratum sample size of 0, PROC SURVEYSELECT does not select a sample from the stratum. This has the effect of subsetting the input data set before sample selection; the stratum that you omit is not included in the sampling frame or represented in the sample.

SEED < =*value* | *SAS-data-set* >

specifies the initial seed for random number generation. You can provide a single seed *value* for the entire sample selection, or you can provide stratum initial seeds by specifying a *SAS-data-set*. To initialize random number generation, a seed must be a positive integer. If you do not specify this option, or if you specify an initial seed that is negative or 0, PROC SURVEYSELECT uses the time of day from the computer's clock to obtain an initial seed. For more information, see the section "Random Number Generation" on page 10456.

PROC SURVEYSELECT displays the value of the initial seed in the "Sample Selection Summary" table. To reproduce the same sample in a subsequent execution of PROC SURVEYSELECT, you can specify the same initial seed in the SEED= option (for the same input data set and sample selection parameters).

If you specify a STRATA statement, you can provide stratum initial seeds in a *SAS-data-set*. If you do not provide stratum initial seeds and if you use the RANUNI random number generator, PROC SURVEYSELECT generates random numbers continuously across strata (from a single pseudorandom number stream). You can specify the OUTSEED option to include the stratum initial seeds in the output data set.

If you do not provide stratum initial seeds and if you use the (default) Mersenne twister random number generator, PROC SURVEYSELECT generates separate pseudorandom number streams for the strata by default. You can specify the STRATUMSEED=NONE option to use a single Mersenne twister stream across all strata. For more information, see the STRATUMSEED= option.

You can provide initial seeds by specifying one of the following forms:

SEED

indicates that stratum initial seeds are provided in a secondary input data set that you name in another option (for example, the SAMPSIZE=*SAS-data-set* option). You should provide the initial seeds in the data set variable named _SEED_ or InitialSeed. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

SEED=*value*

specifies a single initial seed *value* for random number generation. To initialize random number generation, the value must be a positive integer.

SEED=*SAS-data-set*

names a *SAS-data-set* that contains stratum initial seeds. You should provide the stratum initial seeds in the data set variable named _SEED_ or InitialSeed. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary input data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= input data set. The order of the stratum groups in the SEED= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

The OUTSEED option includes the stratum initial seeds in the OUT= output data set. You can reproduce the same sample in a subsequent execution of PROC SURVEYSELECT by specifying the same stratum initial seeds (for the same input data set and sample selection parameters). If you need to reproduce the same sample for only a subset of the strata, you can use the same initial seeds for the strata in the subset.

SELECTALL

requests that PROC SURVEYSELECT select all stratum units when the stratum sample size exceeds the total number of units in the stratum. By default, PROC SURVEYSELECT does not allow you to specify a stratum sample size that is greater than the total number of units in the stratum unless you are using a with-replacement selection method.

The SELECTALL option is available for the following without-replacement selection methods: METHOD=SRS, METHOD=SYS, METHOD=SEQ, METHOD=PPS, and METHOD=PPS_SAMPFORD.

The SELECTALL option is not available for with-replacement selection methods, with-minimum-replacement methods, or those PPS methods that select two units per stratum.

SORT=NEST | SERP

specifies the type of sorting to perform when you specify a CONTROL statement for systematic or sequential sample selection.

SORT=NEST requests nested sorting, and SORT=SERP requests hierarchic serpentine sorting. For more information, see the section "Sorting by CONTROL Variables" on page 10455.

By default, SORT=SERP. Where there is only one CONTROL variable, the two types of sorting are equivalent.

The SORT= option is available when you specify a CONTROL statement for systematic or sequential selection methods (METHOD=SYS, METHOD=PPS_SYS, METHOD=SEQ, and METHOD=PPS_SEQ). When you specify a CONTROL statement, PROC SURVEYSELECT sorts the input data set by the CONTROL variables within strata before selecting the sample.

When you specify a CONTROL statement, you can use the OUTSORT= option to name an output data set that contains the sorted input data set. Otherwise, if you do not specify the OUTSORT= option, the sorted data set replaces the input data set.

The SORT= option and the CONTROL statement are not available when you specify a SAMPLINGUNIT statement.

STATS

includes the selection probability and sampling weight in the OUT= output data set for equal probability selection methods when you do not specify a STRATA statement. By default, the output data set does not include these values for equal probability selection methods unless you specify a STRATA statement. The STATS option applies to the following selection methods: METHOD=SRS, METHOD=URS, METHOD=SYS, METHOD=SEQ, METHOD=BALBOOTSTRAP, and METHOD=BERNOULLI.

In addition to the selection probability and sampling weight, the STATS option includes the following statistics in the output data set for METHOD=BERNOULLI: total number of sampling units, expected sample size, actual sample size, and adjusted sampling weight. For more information, see the section "Bernoulli Sampling" on page 10461.

For PPS selection methods, the output data set contains selection probabilities and sampling weights by default. The STATS option has no effect for PPS methods.

For more information about the contents of the output data set, see the section "Sample Output Data Set" on page 10473.

STRATUMSEED=NONE | RESTORE

controls stratum initial seeds for the Mersenne twister random number generator, which PROC SURVEYSELECT uses by default unless you specify the RANUNI option.

By default, PROC SURVEYSELECT uses separate, independent pseudorandom number streams for the strata. You can store the stratum initial seeds in the output data set by specifying the OUTSEED option. You can reproduce the entire sample (over all strata) by specifying the same initial seed in the SEED= option. You can also reproduce individual stratum samples by specifying the corresponding stratum initial seeds in the SEED= option.

The STRATUMSEED= option applies only to stratified selection, which you request by specifying a STRATA statement. The STRATUMSEED= option applies only to the Mersenne twister random number generator; you cannot specify this option together with the RANUNI option (which requests the RANUNI random number generator).

You cannot specify the STRATUMSEED= option when you provide stratum initial seeds by using the SEED=*SAS-data-set* option.

For more information, see the section "Random Number Generation" on page 10456.

You can specify one of the following keywords:

NONE

uses a single pseudorandom number stream across all strata instead of separate pseudorandom number streams for the strata.

When you specify this option, stratum-level initial seeds are not available; you can reproduce the entire sample (over all strata), but you cannot reproduce individual stratum samples separately (apart from the overall sample). To reproduce an entire sample that the procedure selects when you specify the STRATUMSEED=NONE option, you can specify the same SEED= value, input data set, and selection parameters (along with the STRATUMSEED=NONE option). The OUTSEED option, which stores stratum initial seeds in the output data set, is not available when you specify STRATUMSEED=NONE.

RESTORE

reproduces the stratum initial seeds that PROC SURVEYSELECT uses in releases before SAS/STAT 14.1 for the Mersenne twister random number generator. To reproduce a stratified sample that PROC SURVEYSELECT selects in releases before SAS/STAT 14.1, you can specify STRATUMSEED=RESTORE along with the same SEED= value, input data set, and selection parameters.

CONTROL Statement

CONTROL *variables* **;**

The CONTROL statement names one or more *variables* for sorting the input data set before sample selection. The CONTROL variables can be character or numeric. If you also specify a STRATA statement, PROC SURVEYSELECT sorts by CONTROL variables within strata.

Control sorting is available for systematic and sequential selection methods (METHOD=SYS, METHOD=PPS_SYS, METHOD=SEQ, and METHOD=PPS_SEQ). Ordering the sampling units before systematic or sequential selection can provide additional control over the distribution of the sample.

Control sorting is not available when you use a SAMPLINGUNIT statement, which defines groups of observations as units (clusters) for sample selection. See the description of the SAMPLINGUNIT statement for information about ordering clusters before systematic or sequential selection.

By default (or if you specify the SORT=SERP option in the PROC SURVEYSELECT statement), PROC SURVEYSELECT uses hierarchic serpentine sorting by the CONTROL variables. If you specify the SORT=NEST option, the procedure uses nested sorting. For more information about serpentine and nested sorting, see the section "Sorting by CONTROL Variables" on page 10455.

You can use the OUTSORT= option in the PROC SURVEYSELECT statement to name an output data set that contains the sorted input data set. If you do not specify the OUTSORT= option when you use the CONTROL statement, then the sorted data set replaces the input data set.

FREQ Statement

FREQ *variable* **;**

The FREQ statement names a numeric *variable* that contains the frequency of occurrence of each observation. If you use a FREQ statement, PROC SURVEYSELECT assumes that an observation represents *n* observations, where *n* is the value of the FREQ variable for the observation. The FREQ statement is not available when you specify a SAMPLINGUNIT statement.

The FREQ statement is available only for sample allocation when no sample is selected, which you can request by specifying the ALLOC= and NOSAMPLE options in the STRATA statement. The ALLOC= option requests allocation of the total sample size among the strata, and the NOSAMPLE option requests that no sample be selected after allocation. When you specify the NOSAMPLE option, the procedure computes stratum sample sizes according to the allocation method that you request, but does not select the sample. For more information, see the section "Sample Size Allocation" on page 10468.

The sum of the FREQ variable values (frequencies) represents the total number of sampling units. The sum of the frequencies in a stratum represents the total number of sampling units in the stratum. When you use a FREQ statement, the sample size allocation is based on the expanded total and stratum frequencies.

Values of the FREQ variable must be nonmissing and nonnegative. If a value of the FREQ variable is 0, PROC SURVEYSELECT ignores the observation. If a value of the FREQ variable is not an integer, PROC SURVEYSELECT uses only the integer portion as the frequency of the observation.

ID Statement

ID *variables* **;**

The ID statement names one or more *variables* from the DATA= input data set to include in the OUT= output data set of selected units. If there is no ID statement, PROC SURVEYSELECT includes all variables from the input data set in the output data set. The ID variables can be either character or numeric.

SAMPLINGUNIT | CLUSTER Statement

SAMPLINGUNIT | CLUSTER *variables* < / *options* > **;**

The SAMPLINGUNIT statement names one or more *variables* that identify the sampling units as groups of observations (clusters). The combinations of categories of SAMPLINGUNIT variables define the sampling units. If there is a STRATA statement, sampling units are nested within strata.

When you use a SAMPLINGUNIT statement to define units (clusters), PROC SURVEYSELECT selects a sample of these units by using the selection method and design parameters that you specify in the PROC SURVEYSELECT statement. If you do not use a SAMPLINGUNIT statement, then PROC SURVEYSELECT uses the input data set observations as sampling units by default.

The SAMPLINGUNIT variables are one or more variables in the DATA= input data set. These variables can be either character or numeric. The formatted values of the SAMPLINGUNIT variables determine the SAMPLINGUNIT variable levels. Thus, you can use formats to group values into levels. For more

information, see the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

The SAMPLINGUNIT statement is available with all selection methods except Poisson sampling (METHOD=POISSON) and sequential Poisson sampling (METHOD=SEQ_POISSON).

If you specify the PPS option in the SAMPLINGUNIT statement and do not specify a SIZE statement, the procedure computes sampling unit size as the number of observations in the sampling unit. If you specify a SIZE statement and a SAMPLINGUNIT statement, the procedure computes sampling unit size by summing the size measures of all observations in the sampling unit.

By default, PROC SURVEYSELECT sorts the input data set by the SAMPLINGUNIT variables within strata before sample selection. This groups the observations into sampling units and orders the sampling units by the SAMPLINGUNIT variables. If you do not want the procedure to sort the input data set by the SAMPLINGUNIT variables, then specify the PRESORTED option in the SAMPLINGUNIT statement. By using the PRESORTED option, you can provide the order of the sampling units for systematic and sequential selection methods. The CONTROL statement is not available with the SAMPLINGUNIT statement.

The SAMPLINGUNIT statement defines groups of observations (clusters) to use as sampling units, and PROC SURVEYSELECT selects a sample of these units. When you specify a SAMPLINGUNIT statement, PROC SURVEYSELECT does not select samples of observations from within the sampling units (clusters). To select independent samples within groups, you can use the STRATA statement.

You can specify the following *options*:

ID

includes identification numbers for the selected sampling units (clusters) in the OUT= output data set variable named ClusterID. The ID numbers are consecutive integers that begin with the number 1. For more information, see the section "Sample Output Data Set" on page 10473.

When the selection method is with-replacement or with-minimum-replacement and a cluster is selected more than once, PROC SURVEYSELECT assigns a separate ID number to each selection. (With-replacement and with-minimum-replacement selection methods include METHOD=URS, METHOD=BALBOOTSTRAP, METHOD=PPS_WR, METHOD=PPS_SYS, and METHOD=PPS_SEQ.)

When you specify a STRATA statement, PROC SURVEYSELECT assigns cluster ID numbers within each stratum. When you specify replicated sampling (by using the REPS= option), PROC SURVEYSELECT assigns cluster ID numbers within each replicate.

For with-replacement and with-minimum-replacement selection methods, the ID option invokes the OUTHITS option, which includes a distinct copy of each selected unit in the output data set. When you specify the ID option together with the OUTALL option in the PROC SURVEYSELECT statement, the value of ClusterID is 0 for sampling units that are not selected.

PPS

computes a sampling unit's size measure as the number of observations in the sampling unit. The procedure then uses these size measures to select a sample according to the PPS selection method that you specify in the METHOD= option in the PROC SURVEYSELECT statement.

This option has no effect when you specify a SIZE statement. When you specify a SIZE statement, the procedure computes sampling unit size by summing the size measures of all observations that belong to the sampling unit.

PRESORTED

requests that PROC SURVEYSELECT not sort the input data set by the SAMPLINGUNIT variables within strata. By default, the procedure sorts the input data set by the SAMPLINGUNIT variables, which groups the observations into sampling units and orders the units by the SAMPLINGUNIT variables.

The PRESORTED option enables you to provide the order of the sampling units. For systematic and sequential selection methods, ordering provides additional control over the distribution of the sample and gives some benefits of proportionate stratification. Systematic and sequential methods include METHOD=SYS, METHOD=PPS_SYS, METHOD=SEQ, and METHOD=PPS_SEQ. For more information, see the descriptions of these methods in the section "Sample Selection Methods" on page 10457.

When you specify the PRESORTED option, the procedure treats the sampling unit groups as NOTSORTED. Like the BY statement option NOTSORTED, this does not mean that the data are unsorted by the SAMPLINGUNIT variables, but rather that the data are arranged in groups (according to values of the SAMPLINGUNIT variables) and that these groups are not necessarily in alphabetical or increasing numeric order. For more information about the BY statement NOTSORTED option, see *SAS Programmers Guide: Essentials*.

SIZE Statement

SIZE *variable* **;**

The SIZE statement names one and only one *variable* that contains size measures that are used for PPS selection. The SIZE variable must be numeric.

If you specify a SAMPLINGUNIT statement together with a SIZE statement, the procedure computes a sampling unit's size by summing the size measures of all observations that belong to the sampling unit. Alternatively, if you specify the PPS option in the SAMPLINGUNIT statement and do not specify a SIZE statement, the procedure computes sampling unit size as the number of observations in the sampling unit.

When the value of a sampling unit's size measure is missing or nonpositive, that sampling unit is excluded from the sample selection. For more information, see the section "Missing Values" on page 10455.

You can adjust the size measure values by using the MAXSIZE= option, the MINSIZE= option, or both of these options in the PROC SURVEYSELECT statement.

All PPS selection methods require size measures, which you can provide by specifying a SIZE statement (or by specifying the PPS option in the SAMPLINGUNIT statement). PPS selection methods include the following: METHOD=PPS, METHOD=PPS_BREWER, METHOD=PPS_MURTHY, METHOD=PPS_SAMPFORD, METHOD=PPS_SEQ, METHOD=PPS_SYS, METHOD=PPS_WR, and METHOD=SEQ_POISSON. For information about how size measures are used in sample selection, see the descriptions of PPS selection methods in the section "Sample Selection Methods" on page 10457.

A sampling unit's size measure, which you provide for PPS selection by specifying a SIZE statement, is not the same as the *sample size*. The sample size is the number of units to select for the sample; you specify the sample size in the SAMPSIZE= option in the PROC SURVEYSELECT statement.

For METHOD=POISSON, the variable that you specify in the SIZE statement provides inclusion probabilities for Poisson sampling. For more information, see the section "Poisson Sampling" on page 10461. When the

value of the SIZE variable is missing, nonpositive, or greater than 1, the sampling unit is not included in the sample selection.

STRATA Statement

STRATA *variables* < / *options* > **;**

You can specify a STRATA statement to obtain stratified sampling. The STRATA statement names one or more *variables* that partition the input data set into nonoverlapping groups (strata). The combinations of levels of the STRATA variables define the strata. PROC SURVEYSELECT independently selects samples from the strata according to the selection method and design parameters that you specify in the PROC SURVEYSELECT statement. For information about stratification in sample design, see Lohr (2010), Kalton (1983), Kish (1965), Kish (1987), and Cochran (1977).

The STRATA variables are one or more variables in the DATA= input data set. These variables can be either character or numeric, but PROC SURVEYSELECT treats them as categorical variables. The formatted values of the STRATA variables determine the STRATA variable levels. Thus, you can use formats to group values into levels. For more information, see the FORMAT procedure in the *Base SAS Procedures Guide* and the FORMAT statement and SAS formats in *SAS Formats and Informats: Reference*.

The STRATA variables function much like BY variables, and PROC SURVEYSELECT expects the input data set to be sorted by the STRATA variables. The BY statement options DESCENDING and NOTSORTED are available in the STRATA statement. For more information about these BY statement options, see *SAS Programmers Guide: Essentials*.

If you specify a CONTROL statement or METHOD=PPS in the PROC SURVEYSELECT statement, the input data set must be sorted by the STRATA variables in ascending order. In this case, you cannot specify the NOTSORTED or DESCENDING option in the STRATA statement.

If your input data set is not sorted by the STRATA variables, use one of the following alternatives:

- Sort the data by using the SORT procedure with the STRATA variables in a BY statement.
- Specify the NOTSORTED or DESCENDING option in the STRATA statement (if you do not specify a CONTROL statement or METHOD=PPS in the PROC SURVEYSELECT statement). The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the STRATA variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the STRATA variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the discussion in *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

Table 124.2 summarizes the options available in the STRATA statement. Descriptions of the options follow in alphabetical order.

Table 124.2 STRATA Statement Options for Sample Allocation

Option	Description
ALLOC=*name*	Specifies the allocation method
ALLOC=(*values*)	Provides allocation proportions
ALLOCMAX=	Specifies the maximum sample size per stratum
ALLOCMIN=	Specifies the minimum sample size per stratum
ALPHA=	Specifies the confidence level for the MARGIN= option
COST=	Provides stratum costs
MARGIN=	Specifies the margin of error
NOSAMPLE	Allocates but does not select the sample
STATS	Displays additional allocation statistics
VAR=	Provides stratum variances

You can specify the following *options*:

ALLOC=*name* | (*values*)| *SAS-data-set*

specifies the allocation method *name* or specifies the stratum allocation proportions as a list of *values* or a *SAS-data-set*. You can use the ALLOC= option with any selection method (which you specify in the PROC SURVEYSELECT statement) except METHOD=PPS_BREWER and METHOD=PPS_MURTHY, either of which selects two units from each stratum.

You can specify the sample size allocation by using one of the following forms:

ALLOC=*name*

specifies the method for allocating the total sample size among the strata. You can specify one of the following values for *name*:

NEYMAN

requests Neyman allocation, which allocates the total sample size among the strata in proportion to the stratum sizes and variances. For more information, see the section "Neyman Allocation" on page 10470. If you specify ALLOC=NEYMAN, you must provide the stratum variances by also specifying the VAR= option.

OPTIMAL

OPT

requests optimal allocation, which allocates the total sample size among the strata in proportion to the stratum sizes, stratum variances, and stratum costs. For more information, see the section "Optimal Allocation" on page 10469. If you specify ALLOC=OPTIMAL, you must provide the stratum variances by also specifying the VAR= option, and you must provide the stratum costs by also specifying the COST= option.

PROPORTIONAL

PROP

requests proportional allocation, which allocates the total sample size in proportion to the stratum sizes, where stratum size is the number of sampling units in the stratum. For more information, see the section "Proportional Allocation" on page 10468.

ALLOC=(*values*)

specifies a list of stratum allocation proportion *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. Each value should correspond to a stratum group, and the number of values must equal the number of strata in the input data set.

A stratum allocation proportion specifies the proportion of the total sample size to allocate to the stratum. The sum of the allocation proportions must be 1 or 100%.

The allocation proportions must be positive numbers. You can specify the proportion values as numbers between 0 and 1. Or you can specify the values in percentage form (as numbers between 1 and 100), and PROC SURVEYSELECT converts the numbers to proportions. PROC SURVEYSELECT treats the value 1 as 100% instead of 1%.

The order of the stratum allocation proportions must match the order of the stratum groups in the DATA= input data set. When you specify a list of proportion values, the input data set must be sorted by the STRATA variables in ascending order; you cannot use the DESCENDING or NOTSORTED option in the STRATA statement.

ALLOC=*SAS-data-set*

names a *SAS-data-set* that contains stratum allocation proportions. You should provide the stratum allocation proportions in the data set variable named _ALLOC_. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

A stratum allocation proportion specifies the proportion of the total sample size to allocate to the corresponding stratum. The sum of the allocation proportions must be 1 or 100%.

The allocation proportions must be positive numbers. You can specify the proportion values as numbers between 0 and 1. Or you can specify the values in percentage form (as numbers between 1 and 100), and PROC SURVEYSELECT converts the numbers to proportions. PROC SURVEYSELECT treats the value 1 as 100% instead of 1%.

The ALLOC= data set, which is a secondary input data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= input data set. The order of the stratum groups in the ALLOC= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent between the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary data set in each invocation of PROC SURVEYSELECT.

ALLOCMAX=*n*

specifies the maximum sample size *n* to allocate to a stratum. When an allocated stratum sample size is greater than *n*, PROC SURVEYSELECT allocates only *n* units to the stratum. The procedure then allocates the remaining sample size among the remaining strata by using the allocation method or proportions that you specify. For more information, see the section "Proportional Allocation" on page 10468.

For without-replacement methods, the allocated stratum size sample must also not exceed the number of sampling units in the stratum.

The maximum stratum sample size *n* must be a positive integer. The value of *n* times the number of strata must not be less than the total sample size to be allocated.

This option is available when you specify the ALLOC=PROPORTIONAL, ALLOC=(*values*), or ALLOC=*SAS-data-set* option.

When you specify both the ALLOCMAX= and ALLOCMIN= options, the ALLOCMAX= value must be greater than the ALLOCMIN= value.

ALLOCMIN=*n*

specifies the minimum sample size *n* to allocate to a stratum. When you specify this option, PROC SURVEYSELECT allocates at least *n* sampling units to each stratum.

The minimum stratum sample size *n* must be a positive integer. The value of *n* times the number of strata must not exceed the total sample size to be allocated. For without-replacement selection methods, the value of *n* must not exceed the number of sampling units in any stratum.

By default, PROC SURVEYSELECT allocates at least one sampling unit to each stratum.

ALPHA=*α*

specifies the confidence level that PROC SURVEYSELECT uses in the MARGIN= computations. For more information, see the section "Specifying the Margin of Error" on page 10470.

The value of α must be between 0 and 1; a confidence level of α produces a $100(1-\alpha)\%$ confidence interval. By default, ALPHA=0.05, which produces a 95% confidence interval.

COST < =*values* | *SAS-data-set* >

specifies the stratum-level costs that PROC SURVEYSELECT uses to compute optimal allocation when you specify ALLOC=OPTIMAL. For more information, see the section "Optimal Allocation" on page 10469. The stratum costs must be positive numbers. A stratum cost represents the per-unit cost, which is the survey cost of a single unit in the stratum.

You can provide stratum costs by specifying one of the following forms:

COST

indicates that stratum costs are provided in a secondary input data set that you name in another option (for example, the VAR=*SAS-data-set* option). You should provide the stratum costs in the data set variable named _COST_. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

COST=(*values***)**

specifies a list of stratum cost *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. Each value should correspond to a stratum group, and the number of values must equal the number of strata in the input data set.

The order of the stratum cost values must match the order of the stratum groups in the DATA= input data set. When you specify a list of values, the input data set must be sorted by the STRATA variables in ascending order; you cannot use the DESCENDING or NOTSORTED option in the STRATA statement.

COST=*SAS-data-set*

names a *SAS-data-set* that contains the stratum costs. You should provide the stratum costs in the data set variable named _COST_. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= input data set. The order of the stratum groups in the COST= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

MARGIN=*value*

specifies the margin of error for the estimate of the overall mean from the stratified sample. When you specify this option, PROC SURVEYSELECT determines the stratum sample sizes that achieve the margin *value* by using the allocation method or proportions that you specify in the ALLOC= option. For more information, see the section "Specifying the Margin of Error" on page 10470.

The *value* must be a positive number. When you specify this option, you must also provide the stratum variances in the VAR= option.

You can use the ALPHA= option to specify the confidence level for the MARGIN= computations. By default, ALPHA=0.05, which produces a 95% confidence interval.

You can specify the MARGIN= option with any allocation method (proportional, optimal, or Neyman) or with allocation proportions that you provide (ALLOC=(*values*) or ALLOC=*SAS-data-set*).

Allocation to achieve a specified margin is an alternative approach to the allocation of a specified total sample size. Therefore, when you specify the MARGIN= option, you cannot also specify a total sample size in the SAMPSIZE= option in the PROC SURVEYSELECT statement.

NOSAMPLE

requests that PROC SURVEYSELECT not select a sample after computing the allocation. When you specify this option, the OUT= output data set contains the stratum sample sizes that PROC SURVEYSELECT computes. For more information, see the section "Allocation Output Data Set" on page 10477. (By default, PROC SURVEYSELECT selects a sample after computing the allocation.)

STATS

displays sample allocation statistics. When you specify the MARGIN= option, the STATS option displays the expected margin of error for the allocation. For more information, see the section "Specifying the Margin of Error" on page 10470. When you specify ALLOC=OPTIMAL or ALLOC=NEYMAN but do not specify the MARGIN= option, the STATS option displays the expected variance, which is computed from the stratum variances that you provide and the allocated stratum sample sizes. When you specify ALLOC=OPTIMAL, the STATS option also displays the total stratum-level cost, which is computed from the stratum costs that you provide and the allocated stratum sample sizes.

VAR < =*values* | *SAS-data-set* >

specifies the stratum variances that PROC SURVEYSELECT uses to compute optimal allocation (ALLOC=OPTIMAL), Neyman allocation (ALLOC=NEYMAN), or allocation for a specified margin (MARGIN=). The stratum variances must be positive numbers.

You can provide stratum variances by specifying one of the following forms:

VAR

indicates that stratum variances are provided in a secondary input data set that you name in another option (for example, the COST=*SAS-data-set* option). You should provide the stratum variances in the data set variable named _VAR_. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

VAR=(*values***)**

specifies a list of stratum variance *values*. You can separate the values with blanks or commas, and you must enclose the list of values in parentheses. Each value should correspond to a stratum group, and the number of values must equal the number of strata in the input data set.

The order of the stratum variance values must match the order of the stratum groups in the DATA= input data set. When you specify a list of values, the input data set must be sorted by the STRATA variables in ascending order; you cannot use the DESCENDING or NOTSORTED option in the STRATA statement.

VAR=*SAS-data-set*

names a *SAS-data-set* that contains the stratum variances. You should provide the stratum variances in the data set variable named _VAR_. Each observation in the data set should correspond to a stratum group, which is determined by the values of the STRATA variables.

This data set, which is a secondary data set, must contain all stratification variables that you specify in the STRATA statement. The data set must also contain all stratum groups that appear in the DATA= input data set. The order of the stratum groups in the VAR= data set must match the order of the groups in the DATA= data set. If formats are associated with the STRATA variables, the formats must be consistent in the two data sets. For more information, see the section "Secondary Input Data Set" on page 10472. You can name only one secondary input data set in each invocation of PROC SURVEYSELECT.

Details: SURVEYSELECT Procedure

Missing Values

PROC SURVEYSELECT treats missing values of STRATA and SAMPLINGUNIT variables like any other STRATA or SAMPLINGUNIT variable value. The missing values form a separate, valid variable level.

When you use a FREQ statement for sample size allocation, all values of the frequency variable must be nonmissing. If there is a missing or nonpositive frequency, PROC SURVEYSELECT does not perform the allocation.

When you specify a SIZE variable, any sampling units that have missing or nonpositive size measures are excluded from the sample selection. The procedure provides a log note that reports the number of observations omitted because of missing or nonpositive size measures.

If you do not use a SAMPLINGUNIT statement with the SIZE statement, your sampling units are input data set observations, and observations that have missing or nonpositive size measures are excluded from the sample selection. If you do use a SAMPLINGUNIT statement with the SIZE statement, the procedure computes sampling unit size by summing the size measures of all observations in the unit. When summing the observation size measures, the procedure omits any observations that have missing or nonpositive size measures. If the size of an entire sampling unit is missing or nonpositive, the procedure excludes that unit from the sample selection. When a sampling unit is selected, the output data set includes all observations that belong to the selected unit, regardless of whether an observation's size measure is missing.

If you provide stratum-level design or allocation information in a secondary input data set, the variable values should be nonmissing. For example, if a stratum value of _NSIZE_ (or SampleSize) in the SAMPSIZE= secondary input data set is missing or negative, PROC SURVEYSELECT cannot select a sample from the stratum. The procedure gives an error message and skips the stratum. Similarly, if other secondary data set variables have missing values for a stratum, a sample cannot be selected from the stratum. These variables include _NRATE_, _MINSIZE_, _MAXSIZE_, _CERTSIZE_, and _CERTP_. Additionally, if any of the sample allocation variables in the secondary input data set have missing or nonpositive values, PROC SURVEYSELECT cannot compute the sample allocation. Variables that provide information for allocation include _ALLOC_, _VAR_, and _COST_. For more information, see the section "Secondary Input Data Set" on page 10472.

Sorting by CONTROL Variables

If you specify a CONTROL statement, PROC SURVEYSELECT sorts the input data set by the CONTROL variables before selecting the sample. If you also specify a STRATA statement, the procedure sorts by CONTROL variables within strata. Sorting by CONTROL variables is available for systematic and sequential selection methods, which include METHOD=SYS, METHOD=PPS_SYS, METHOD=SEQ, and METHOD=PPS_SEQ. Sorting provides additional control over the distribution of the sample and gives some benefits of proportionate stratification.

Control sorting is not available when you use a SAMPLINGUNIT statement, which defines groups of observations as units (clusters) for sample selection. See the description of the SAMPLINGUNIT statement for information about ordering clusters before systematic or sequential selection.

When you specify a CONTROL statement, the sorted data set replaces the input data set by default. Alternatively, you can use the OUTSORT= option to name an output data set that contains the sorted input data set.

PROC SURVEYSELECT provides two types of sorting: hierarchic serpentine sorting and nested sorting. By default (or if you specify the SORT=SERP option), the procedure uses serpentine sorting. If you specify the SORT=NEST option, then the procedure sorts by the CONTROL variables according to nested sorting. These two types of sorting are equivalent when there is only one CONTROL variable.

If you request nested sorting, PROC SURVEYSELECT sorts observations in the same order as PROC SORT does for an ascending sort by the CONTROL variables. For more information, see the chapter "The SORT Procedure" in the *Base SAS Procedures Guide*. PROC SURVEYSELECT sorts within strata if you also specify a STRATA statement. The procedure first arranges the input observations in ascending order of the first CONTROL variable. Then within each level of the first control variable, the procedure arranges the observations in ascending order of the second CONTROL variable. This continues for all CONTROL variables that are specified.

In hierarchic serpentine sorting, PROC SURVEYSELECT sorts by the first CONTROL variable in ascending order. Then within the first level of the first CONTROL variable, the procedure sorts by the second CONTROL variable in ascending order. Within the second level of the first CONTROL variable, the procedure sorts by the second CONTROL variable in descending order. Sorting by the second CONTROL variable continues to alternate between ascending and descending sorting throughout all levels of the first CONTROL variable. If there is a third CONTROL variable, the procedure sorts by that variable within levels formed from the first two CONTROL variables, again alternating between ascending and descending sorting. This continues for all CONTROL variables that are specified. This sorting algorithm minimizes the change from one observation to the next with respect to the CONTROL variable values, thus making nearby observations more similar. For more information about serpentine sorting, see Chromy (1979) and Williams and Chromy (1980).

Random Number Generation

The probability sampling methods that PROC SURVEYSELECT provides use random numbers in their selection algorithms, as described in the following sections and in the references cited. PROC SURVEYSELECT uses a uniform random number function to generate streams of pseudorandom numbers from an initial starting point, or *seed*. You can use the SEED= option to specify the initial seed. If you do not specify the SEED= option, PROC SURVEYSELECT uses the time of day from the computer's clock to obtain the initial seed. For information about specifying initial seeds for strata, storing stratum seeds in the output data set, and reproducing samples, see the description of the SEED= option.

By default, PROC SURVEYSELECT uses the Mersenne twister uniform random number generator (Matsumoto and Nishimura 1998) The Mersenne twister generator has a very long period ($2^{19937} - 1$) and good statistical properties. The algorithm is a twisted generalized feedback shift register. This is the same random number generator that the RAND function provides for the uniform distribution. For more information, see *SAS Functions and CALL Routines: Reference*.

If you specify the RANUNI option, PROC SURVEYSELECT uses the RANUNI random number generator. This uniform random number generator is based on the method of Fishman and Moore (1982), which uses a prime modulus multiplicative generator with modulus 2^{31} and multiplier 397,204,094. This is the same uniform random number generator that the RANUNI function provides. For more information, see *SAS Functions and CALL Routines: Reference*.

When you use the RANUNI random number generator for stratified sampling, PROC SURVEYSELECT generates a single pseudorandom number stream across all strata. You can store the stratum initial seeds in the output data set by specifying the OUTSEED option, and you can use the stratum seeds to reproduce stratum samples (separately, apart from the entire sample).

When you use the Mersenne twister random number generator for stratified sampling, PROC SURVEYSELECT generates separate, independent pseudorandom number streams for the strata by default. To use a single Mersenne twister pseudorandom number stream across all strata, you can specify the STRATUMSEED=NONE option. When you specify this option, stratum initial seeds are not available in the output data set.

In releases before SAS/STAT 14.1, PROC SURVEYSELECT uses a different method to initialize the stratum (Mersenne twister) pseudorandom number streams. To reproduce stratified samples that PROC SURVEYSELECT selects by using the Mersenne twister random number generator in releases before SAS/STAT 14.1, you can specify the STRATUMSEED=RESTORE option (along with the same SEED= value, input data set, and selection parameters).

In releases before SAS/STAT 12.1, PROC SURVEYSELECT uses the RANUNI random number generator by default. To reproduce samples that PROC SURVEYSELECT selects in releases before SAS/STAT 12.1, you can specify the RANUNI option along with the same SEED= value (for the same input data set and selection parameters).

Sample Selection Methods

PROC SURVEYSELECT provides a variety of methods for selecting probability-based random samples. With probability sampling, each unit in the survey population has a known, positive probability of selection. This property of probability sampling avoids selection bias and enables you to use statistical theory to make valid inferences from the sample to the survey population. For more information about probability sampling, see Lohr (2010), Kish (1965), Kish (1987), Kalton (1983), and Cochran (1977).

In equal probability sampling, each unit in the sampling frame, or in a stratum, has the same probability of being selected for the sample. PROC SURVEYSELECT provides the following methods that select units with equal probability: simple random sampling, unrestricted random sampling, systematic random sampling, sequential random sampling, and Bernoulli sampling. In simple random sampling, units are selected *without replacement*, which means that a unit cannot be selected more than once. Both systematic and sequential equal probability sampling are also without replacement. In unrestricted random sampling, units are selected *with replacement*, which means that a unit can be selected more than once. In with-replacement sampling, the *number of hits* refers to the number of times a unit is selected.

In probability proportional to size (PPS) sampling, a unit's selection probability is proportional to its size measure. PROC SURVEYSELECT provides the following methods that select units with probability proportional to size (PPS): PPS sampling without replacement, PPS sampling with replacement, PPS systematic sampling, PPS sequential sampling, Brewer's method, Murthy's method, and Sampford's method. PPS sampling is often used in cluster sampling, where you select clusters (or groups of sampling units) of varying size in the first stage of selection. For example, clusters might be schools, hospitals, or geographical areas, and the final sampling units might be students, patients, or citizens. Cluster sampling can provide efficiencies in frame construction and other survey operations. For more information, see Lohr (2010), Kalton (1983), and Kish (1965), in addition to the other references cited in the following sections.

The following sections give detailed descriptions of the sample selection methods available in PROC SURVEYSELECT. In these sections, n_h denotes the sample size (the number of units in the sample) for stratum h, and N_h denotes the population size (number of units in the population) for stratum h, for $h = 1, 2, \ldots, H$. When the sample design is not stratified, n denotes the sample size, and N denotes the population size. For PPS sampling, M_{hi} represents the size measure for unit i in stratum h, $M_{h\cdot}$ is the total of all size measures for the population of stratum h, and $Z_{hi} = M_{hi}/M_{h\cdot}$ is the relative size of unit i in stratum h.

Simple Random Sampling

The method of simple random sampling (METHOD=SRS) selects units with equal probability and without replacement. Each possible sample of n different units out of N has the same probability of being selected. The selection probability for each individual unit is n/N. When you request stratified sampling by using a STRATA statement, PROC SURVEYSELECT selects samples independently within strata. The selection probability for a unit in stratum h is n_h/N_h for stratified simple random sampling.

By default, PROC SURVEYSELECT uses Floyd's ordered hash table algorithm for simple random sampling. This algorithm is fast, efficient, and appropriate for large data sets. For more information, see Bentley and Floyd (1987) and Bentley and Knuth (1986).

If there is not enough memory available for Floyd's algorithm, PROC SURVEYSELECT switches to the sequential algorithm of Fan, Muller, and Rezucha (1962), which requires less memory but might require more time to select the sample. When PROC SURVEYSELECT uses the alternative sequential algorithm, it writes a note to the log. To request the sequential algorithm, even if enough memory is available for Floyd's algorithm, you can specify METHOD=SRS2 in the PROC SURVEYSELECT statement.

Unrestricted Random Sampling

The method of unrestricted random sampling (METHOD=URS) selects units with equal probability and with replacement. Because units are selected with replacement, a unit can be selected for the sample more than once. The expected number of hits (selections) for each unit is n/N when sampling without stratification. For stratified sampling, the expected number of hits for a unit in stratum h is n_h/N_h. The expected number of hits exceeds one when the sample size n is greater than the population size N.

For unrestricted random sampling, by default, the output data set contains a single copy of each unit selected, even when a unit is selected more than once, and the variable NumberHits records the number of hits (selections) for each unit. If you specify the OUTHITS option, the output data set contains m copies of a sampling unit for which NumberHits is m; for example, the output data set contains three copies of a sampling unit that is selected three times (NumberHits is three). For information about the contents of the output data set, see the section "Sample Output Data Set" on page 10473.

Balanced Bootstrap Sampling

Balanced bootstrap sampling (METHOD=BALBOOTSTRAP) selects r bootstrap samples of N units, where N is the total number of sampling units (in the stratum or in the data set) and r is the number of samples (replicates) that you specify. Units are selected with equal probability and with replacement. The bootstrap selection is balanced so that the total number of selections (over all replicate samples) is r selections for each sampling unit. For more information, see Davison, Hinkley, and Schechtman (1986). PROC SURVEYSELECT performs balanced bootstrap selection by using the algorithm of Gleason (1988).

Systematic Random Sampling

Systematic random sampling (METHOD=SYS) selects units at a fixed interval throughout the sampling frame (or stratum) after a random start. If you request stratified sampling by specifying a STRATA statement, PROC SURVEYSELECT independently selects systematic samples from the strata. PROC SURVEYSELECT applies systematic selection to sampling units in the order of their appearance in the input data set, or in their sorted order if you specify a CONTROL statement.

This section describes equal-probability systematic sampling, where each sampling unit in the sampling frame (or stratum) has the same probability of selection. For information about PPS systematic sampling, see the section "PPS Systematic Sampling" on page 10464.

When you specify the sample size in the SAMPSIZE= option, PROC SURVEYSELECT computes the systematic selection interval as the ratio of the total number of sampling units to the sample size (N/n, or N_h/n_h for stratified sampling). The procedure uses a fractional systematic interval to provide the specified sample size exactly. The selection probability for each unit is computed as n/N (or n_h/N_h for stratified sampling).

When you specify the sampling rate in the SAMPRATE= option, PROC SURVEYSELECT computes the systematic selection interval as the inverse of the sampling rate. The selection probability for each unit is the sampling rate.

Instead of specifying the sample size or sampling rate, you can directly specify the systematic interval in the INTERVAL= option. When you specify the interval, PROC SURVEYSELECT computes the selection probability as the inverse of the interval value.

By default, PROC SURVEYSELECT randomly determines a starting value in the selection interval. Optionally, you can specify the starting value in the START= option. The random component of systematic sampling is the random selection of a starting value in the systematic interval. If you use the START= option to provide a purposely chosen (nonrandom) starting value, the resulting systematic selection does not provide a random, probability-based sample.

Systematic sampling controls the distribution of the sample by spreading the selections throughout the sampling frame (or stratum) at equal intervals and thus provides implicit stratification. You can specify a CONTROL statement to order the input data set by CONTROL variables before sample selection. If you also specify a STRATA statement, PROC SURVEYSELECT sorts by the CONTROL variables within strata. If you do not specify a CONTROL statement, PROC SURVEYSELECT applies systematic selection to the observations in the order in which they appear in the input data set.

Sequential Random Sampling

If you specify the METHOD=SEQ option and do not include a SIZE statement, PROC SURVEYSELECT uses the equal probability version of Chromy's method for sequential random sampling. This method selects units sequentially with equal probability and without replacement. For more information, see Chromy (1979) and Williams and Chromy (1980). For information about Chromy's PPS selection method, see the section "PPS Sequential Sampling" on page 10465.

Sequential random sampling controls the distribution of the sample by spreading it throughout the sampling frame or stratum, thus providing implicit stratification according to the order of units in the frame or stratum. You can use the CONTROL statement to sort the input data set by the CONTROL variables before sample selection. If you also use a STRATA statement, PROC SURVEYSELECT sorts by the CONTROL variables within strata. By default (or if you specify the SORT=SERP option), the procedure uses hierarchic serpentine

ordering for sorting. If you specify the SORT=NEST option, the procedure uses nested sorting. See the section "Sorting by CONTROL Variables" on page 10455 for descriptions of serpentine and nested sorting. If you do not specify a CONTROL statement, PROC SURVEYSELECT applies sequential selection to the observations in the order in which they appear in the input data set.

Following Chromy's method of sequential selection, PROC SURVEYSELECT randomly chooses a starting unit from the entire stratum (or frame, if the design is not stratified). With this unit as the first one, the procedure treats the stratum units as a closed loop. This is done so that all pairwise (joint) selection probabilities are positive and an unbiased variance estimator can be obtained. The procedure numbers units sequentially from the random start to the end of the stratum and then continues from the beginning of the stratum until all units are numbered.

Beginning with the randomly chosen starting unit, PROC SURVEYSELECT accumulates the expected number of selections (hits), where the expected number of selections $\mathrm{E}(S_{hi})$ is n_h/N_h for all units i in stratum h. The procedure computes

$$I_{hi} = \mathrm{Int}\Big(\sum_{j=1}^{i} \mathrm{E}(S_{hj})\Big) = \mathrm{Int}(i\, n_h/N_h)$$

$$F_{hi} = \mathrm{Frac}\Big(\sum_{j=1}^{i} \mathrm{E}(S_{hj})\Big) = \mathrm{Frac}(i\, n_h/N_h)$$

where $\mathrm{Int}(\cdot)$ denotes the integer part of the number, and $\mathrm{Frac}(\cdot)$ denotes the fractional part.

Considering each unit sequentially, Chromy's method determines whether unit i is selected by comparing the total number of selections for the first $(i - 1)$ units,

$$T_{h(i-1)} = \sum_{j=1}^{i-1} S_{hj}$$

with the value of $I_{h(i-1)}$.

If $T_{h(i-1)} = I_{h(i-1)}$, Chromy's method determines whether or not unit i is selected as follows. If $F_{hi} = 0$ or $F_{h(i-1)} > F_{hi}$, then unit i is selected with certainty. Otherwise, unit i is selected with probability

$$(F_{hi} - F_{h(i-1)})/(1 - F_{h(i-1)})$$

If $T_{h(i-1)} = (I_{h(i-1)} + 1)$, Chromy's method determines whether or not unit i is selected as follows. If $F_{hi} = 0$ or $F_{hi} > F_{h(i-1)}$, then the unit is not selected. Otherwise, unit i is selected with probability

$$F_{hi}/F_{h(i-1)}$$

Bernoulli Sampling

Bernoulli sampling, which you request by specifying the METHOD=BERNOULLI option, is an equal probability selection method for which the total sample size is not fixed. PROC SURVEYSELECT performs an independent random selection trial for each of the N sampling units in the input data set by using the constant inclusion probability (sampling rate) that you specify. You can specify a single value of the inclusion probability π to use for all N sampling units, or you can specify separate stratum-level values of π_h to use for the N_h units in each stratum.

You provide the inclusion probability (or probabilities) by specifying the SAMPRATE= option. For stratified sampling (which you request by using the STRATA statement), you can specify the same sampling rate for each stratum in the SAMPRATE=*value* option. Or you can specify different sampling rates for different strata by using the SAMPRATE=(*values*) or SAMPRATE=*SAS-data-set* option.

In Bernoulli sampling, the sample size n (number of units selected) is not fixed; it is a random variable that has a binomial distribution with parameters N and π. The possible values of n range from 0 to N. The expected value of the sample size is πN (or $\pi_h N_h$ for stratified sampling), and the variance of the sample size is $\pi(1-\pi)N$.

For Bernoulli sampling, the selection probability is the inclusion probability that you specify in the SAMPRATE= option. PROC SURVEYSELECT computes the sampling weight as the inverse of the selection probability, which is $1/\pi$. For Bernoulli sampling, the procedure also computes an adjusted sampling weight as the ratio of the total number of sampling units to the actual sample size, N/n (or N_h/n_h for stratified sampling). The joint selection probability for any two distinct units is π^2. For more information, see Särndal, Swensson, and Wretman (1992).

You can specify the STATS option to include the following information in the OUT= output data set for METHOD=BERNOULLI: total number of sampling units, selection probability, expected sample size, actual sample size, sampling weight, and adjusted sampling weight.

Poisson Sampling

Poisson sampling, which you request by specifying the METHOD=POISSON option, is an unequal probability sampling method for which the total sample size is not fixed. A generalization of Bernoulli sampling, Poisson sampling also consists of independent random selection trials for the N sampling units in the input data set, but the sampling units can have different inclusion probabilities. You provide inclusion probabilities for Poisson sampling in the variable that you specify in the SIZE statement.

The expected value of the sample size for Poisson sampling is $\sum_i \pi_i$, where π_i is the inclusion probability for sampling unit i. The variance of the sample size is $\sum_i \pi_i(1-\pi_i)$.

For Poisson sampling, the selection probability for unit i is the inclusion probability π_i that you specify by using the SIZE statement. PROC SURVEYSELECT computes the sampling weight for unit i as the inverse of the selection probability, which is $1/\pi_i$. The joint selection probability for any two distinct units i and j is $\pi_i \pi_j$. For more information, see Särndal, Swensson, and Wretman (1992).

Sequential Poisson Sampling

Sequential Poisson sampling, which you request by specifying the METHOD=SEQ_POISSON option, is a fixed-sample-size modification of Poisson sampling. For information about Poisson sampling, see the section "Poisson Sampling" on page 10461.

PROC SURVEYSELECT performs sequential Poisson sampling by using the method of Ohlsson (1998). A *transformed random number* is computed for each sampling unit as X_{hi}/M_{hi}, where M_{hi} is the size measure of unit i in stratum h and X_{hi} is a uniform random number (from the procedure's pseudorandom number stream). For more information about random number generation, see the SEED= option and the section "Random Number Generation" on page 10456.

The N_h transformed random numbers are ordered, and the stratum h sample consists of the n_h sampling units that correspond to the n_h smallest transformed random numbers.

Although this algorithm produces a sample of the fixed size that you specify, the sample selection is considered to be only approximately probability proportional to size (PPS); it is not strictly PPS. For more information, see Ohlsson (1998). The (approximate) selection probability for unit i in stratum h is computed as $n_h Z_{hi}$, where n_h is the sample size for stratum h and Z_{hi} is the relative size of unit i in stratum h. The relative size is computed as $M_{hi}/M_{h\cdot}$, which is the ratio of the size measure for unit i in stratum h (M_{hi}) to the total of all size measures for stratum h ($M_{h\cdot}$)

The relative size of each sampling unit cannot exceed $1/n_h$ because the selection probability (n_h times the relative size) cannot exceed 1. This requirement can be expressed as $Z_{hi} \leq 1/n_h$, or equivalently as $M_{hi} \leq M_{h\cdot}/n_h$. If your size measures do not meet this requirement, you can adjust the size measures by using the MAXSIZE= or MINSIZE= option. Or you can select the larger units with certainty by using the CERTSIZE= or CERTSIZE=P= option. Alternatively, you can use a selection method that does not have a relative size restriction, such as PPS with minimum replacement (METHOD=PPS_SEQ).

PPS Sampling without Replacement

When you specify the METHOD=PPS option, PROC SURVEYSELECT selects units with probability proportional to size and without replacement. The selection probability for unit i in stratum h is $n_h Z_{hi}$, where n_h is the sample size for stratum h and Z_{hi} is the relative size of unit i in stratum h. The relative size Z_{hi} is computed as $M_{hi}/M_{h\cdot}$, which is the ratio of the size measure of unit i in stratum h to the total of all size measures in stratum h.

Because selection probabilities cannot exceed 1, the relative size for each unit must not exceed $1/n_h$ for METHOD=PPS. This requirement can be expressed as $Z_{hi} \leq 1/n_h$, or equivalently as $M_{hi} \leq M_{h\cdot}/n_h$. If your size measures do not meet this requirement, you can adjust the size measures by using the MAXSIZE= or MINSIZE= option. Or you can request certainty selection for the larger units by using the CERTSIZE= or CERTSIZE=P= option. Alternatively, you can use a selection method that does not have this relative size restriction, such as PPS with minimum replacement (METHOD=PPS_SEQ).

PROC SURVEYSELECT performs PPS selection by using the Hanurav-Vijayan algorithm. Hanurav (1967) introduced this algorithm for the selection of two units per stratum, and Vijayan (1968) generalized it for the selection of more than two units. This algorithm enables computation of joint selection probabilities and provides joint selection probability values that usually ensure nonnegativity and stability of the Sen-Yates-Grundy variance estimator. For more information, see Fox (1989), Golmant (1990), and Watts (1991).

The notation in the remainder of this section drops the stratum subscript h for simplicity. If you specify a stratified design, n now denotes the sample size for the current stratum, N denotes the stratum population

size, M_i denotes the size measure for unit i in the stratum, and M denotes the total of size measures in the stratum. For a stratified design, PROC SURVEYSELECT selects samples independently within strata by using the same selection method in each stratum.

PROC SURVEYSELECT performs the Hanurav-Vijayan selection algorithm as described by Fox (1989, p. 169). For the definition of $P_k^{(i)}$, see Golmant (1990). The sampling units are first sorted in ascending order by size measure so that $M_1 \le M_2 \le \cdots \le M_N$. The procedure then selects a PPS sample of n units as follows:

1. The procedure randomly chooses one of the integers $1, 2, \ldots, n$ with probability $\theta_1, \theta_2, \ldots, \theta_n$, where

   $$\theta_i = n\,(Z_{N-n+i+1} - Z_{N-n+i})\,(T + i\,Z_{N-n+1})\,/\,T$$

   where $Z_j = M_j/M$ and

   $$T = \sum_{j=1}^{N-n} Z_j$$

   By definition, $Z_{N+1} = 1/n$ to ensure that $\sum_{i=1}^{n} \theta_i = 1$.

2. If the integer i is selected in step 1, the procedure includes the last $(n-i)$ units in the sample (where the units are ordered by their size measures). The procedure then selects the remaining i units by following steps 3 through 6.

3. The procedure defines new normed size measures for the remaining $(N-n+i)$ units that were not selected in steps 1 and 2:

   $$Z_j^*(i) = \begin{cases} Z_j\,/\,(T + i\,Z_{N-n+1}) & \text{for } j = 1, \ldots, N-n+1 \\ Z_{N-n+1}\,/\,(T + i\,Z_{N-n+1}) & \text{for } j = N-n+2, \ldots, N-n+i \end{cases}$$

4. The procedure selects the next unit from the first $(N-n+1)$ units with probability proportional to $a_j(1)$, where

   $$\begin{aligned} a_1(1) &= i\,Z_1^*(i) \\ a_j(1) &= i\,Z_j^*(i) \textstyle\prod_{k=1}^{j-1}\left(1 - (i-1)\,P_k^{(i)}\right) \quad \text{for } j = 2, \ldots, N-n+1 \end{aligned}$$

   and

   $$P_k^{(i)} = M_k/(M_{k+1} + M_{k+2} + \cdots + M_{N-n+i})$$

5. Where j_1 denotes the unit that is selected in step 4, the procedure selects the next unit from units (j_1+1) through $(N-n+2)$ with probability proportional to $a_j(2, j_1)$, where

   $$a_{j_1+1}(2, j_1) = (i-1)\,Z_{j_1+1}^*(i)$$

   $$a_j(2, j_1) = (i-1)\,Z_j^*(i) \prod_{k=j_1+1}^{j-1} \left(1 - (i-2)\,P_k^{(i)}\right) \quad \text{for } j = j_1+2, \ldots, N-n+2$$

6. The procedure repeats step 5 until all n sample units are selected.

If you specify the JTPROBS option, PROC SURVEYSELECT computes the joint selection probabilities for all pairs of selected units in each stratum. The joint selection probability for units i and j is

$$P_{(ij)} = \sum_{r=1}^{n} \theta_r K_{ij}^{(r)}$$

where

$$K_{ij}^{(r)} = \begin{cases} 1 & N-n+r < i \le N-1 \\ r\,Z_{N-n+1} \,/\, (T + r\,Z_{N-n+1}) & N-n < i \le N-n+r, \quad j > N-n+r \\ r\,Z_i \,/\, (T + r\,Z_{N-n+1}) & 1 \le i \le N-n, \quad j > N-n+r \\ \pi_{ij}^{(r)} & j \le N-n+r \end{cases}$$

$$\pi_{ij}^{(r)} = r(r-1)\,P_i^{(r)}\,Z_j^*(r) \prod_{k=1}^{i-1} (1 - P_k^{(r)})$$

$$P_k^{(r)} = M_k/(M_{k+1} + M_{k+2} + \cdots + M_{N-n+r})$$

PPS Sampling with Replacement

If you specify the METHOD=PPS_WR option, PROC SURVEYSELECT selects units with probability proportional to size and with replacement. The procedure makes n_h independent random selections from the stratum of N_h units, selecting with probability $Z_{hi} = M_{hi}/M_{h\cdot}$. Because units are selected with replacement, a unit can be selected for the sample more than once. The expected number of hits (selections) for unit i in stratum h is $n_h Z_{hi}$. If you specify the JTPROBS option, PROC SURVEYSELECT computes the joint expected number of hits for all pairs of selected units in each stratum. The joint expected number of hits for units i and j in stratum h is

$$P_{h(ij)} = \begin{cases} n_h(n_h-1)Z_{hi}Z_{hj} & \text{for } j \ne i \\ n_h(n_h-1)Z_{hi}Z_{hi}/2 & \text{for } j = i \end{cases}$$

PPS Systematic Sampling

If you specify the METHOD=PPS_SYS option, PROC SURVEYSELECT selects the sample by using systematic random sampling with probability proportional to size. Systematic sampling selects units at a fixed interval throughout the sampling frame (or stratum) after a random start. If you request stratified sampling by specifying a STRATA statement, PROC SURVEYSELECT independently selects systematic samples from the strata. PROC SURVEYSELECT applies systematic selection to sampling units in the order of their appearance in the input data set, or in their sorted order if you specify a CONTROL statement.

When you specify the sample size in the SAMPSIZE= option, PROC SURVEYSELECT computes the systematic selection interval as the ratio of the total size to the sample size (M/n, or $M_{h\cdot}/n_h$ for stratified sampling). The procedure uses a fractional systematic interval to provide the specified sample size exactly. Depending on the sample size and the values of the size measures, it might be possible for a sampling unit to be selected more than once. The expected number of hits (selections) for unit i in stratum h is computed as $n_h M_{hi}/M_{h\cdot} = n_h Z_{hi}$. For more information, see Cochran (1977, pp. 265–266) and Madow (1949).

Instead of specifying the sample size for systematic sampling, you can directly specify the systematic interval in the INTERVAL= option. When you specify the interval, PROC SURVEYSELECT computes the expected number of hits as the inverse of the interval value.

By default, PROC SURVEYSELECT randomly determines a starting value in the selection interval. Optionally, you can specify the starting value in the START= option. The random component of systematic sampling is the random selection of a starting value in the systematic interval. If you use the START= option to provide a purposely chosen (nonrandom) starting value, the resulting systematic selection does not provide a random, probability-based sample.

Systematic sampling controls the distribution of the sample by spreading the selections throughout the sampling frame (or stratum) at equal intervals and thus provides implicit stratification. You can specify a CONTROL statement to order the input data set by the CONTROL variables before sample selection. If you also specify a STRATA statement, PROC SURVEYSELECT sorts by the CONTROL variables within strata. If you do not specify a CONTROL statement, PROC SURVEYSELECT applies systematic selection to the observations in the order in which they appear in the input data set.

PPS Sequential Sampling

If you specify the METHOD=PPS_SEQ option, PROC SURVEYSELECT uses Chromy's method of sequential random sampling. For more information, see Chromy (1979) and Williams and Chromy (1980). Chromy's method selects units sequentially with probability proportional to size and with minimum replacement. Selection *with minimum replacement* means that the actual number of hits for a unit can equal the integer part of the expected number of hits for that unit, or the next largest integer. This can be compared to selection *without replacement*, where each unit can be selected only once, so the number of hits can equal 0 or 1. The other alternative is selection *with replacement*, where there is no restriction on the number of hits for each unit, so the number of hits can equal $0, 1, \ldots, n_h$, where n_h is the stratum sample size.

Sequential random sampling controls the distribution of the sample by spreading it throughout the sampling frame or stratum, thus providing implicit stratification according to the order of units in the frame or stratum. You can use the CONTROL statement to sort the input data set by the CONTROL variables before sample selection. If you also use a STRATA statement, PROC SURVEYSELECT sorts by the CONTROL variables within strata. By default (or if you specify the SORT=SERP option), the procedure uses hierarchic serpentine ordering to sort the sampling frame by the CONTROL variables within strata. If you specify the SORT=NEST option, the procedure uses nested sorting. See the section "Sorting by CONTROL Variables" on page 10455 for descriptions of serpentine and nested sorting. If you do not specify a CONTROL statement, PROC SURVEYSELECT applies sequential selection to the observations in the order in which they appear in the input data set.

According to Chromy's method of sequential selection, PROC SURVEYSELECT first chooses a starting unit randomly from the entire stratum, with probability proportional to size. The procedure uses this unit as the first one and treats the stratum observations as a closed loop. This is done so that all pairwise (joint) expected number of hits are positive and an unbiased variance estimator can be obtained. The procedure numbers observations sequentially from the random start to the end of the stratum and then continues from the beginning of the stratum until all units are numbered.

Beginning with the randomly chosen starting unit, Chromy's method partitions the ordered stratum sampling frame into n_h zones of equal size. There is one selection from each zone and a total of n_h hits (selections), although fewer than n_h distinct units might be selected. Beginning with the random start, the procedure

accumulates the expected number of hits and computes

$$\mathrm{E}(S_{hi}) = n_h Z_{hi}$$

$$I_{hi} = \mathrm{Int}\Big(\sum_{j=1}^{i} \mathrm{E}(S_{hj})\Big)$$

$$F_{hi} = \mathrm{Frac}\Big(\sum_{j=1}^{i} \mathrm{E}(S_{hj})\Big)$$

where $\mathrm{E}(S_{hi})$ represents the expected number of hits for unit i in stratum h, $\mathrm{Int}(\cdot)$ denotes the integer part of the number, and $\mathrm{Frac}(\cdot)$ denotes the fractional part.

Considering each unit sequentially, Chromy's method determines the actual number of hits for unit i by comparing the total number of hits for the first $(i - 1)$ units,

$$T_{h(i-1)} = \sum_{j=1}^{i-1} S_{hj}$$

with the value of $I_{h(i-1)}$.

If $T_{h(i-1)} = I_{h(i-1)}$, Chromy's method determines the total number of hits for the first i units as follows. If $F_{hi} = 0$ or $F_{h(i-1)} > F_{hi}$, then $T_{hi} = I_{hi}$. Otherwise, $T_{hi} = I_{hi} + 1$ with probability

$$(F_{hi} - F_{h(i-1)})/(1 - F_{h(i-1)})$$

And the number of hits for unit i is $T_{hi} - T_{h(i-1)}$.

If $T_{h(i-1)} = (I_{h(i-1)} + 1)$, Chromy's method determines the total number of hits for the first i units as follows. If $F_{hi} = 0$, then $T_{hi} = I_{hi}$. If $F_{hi} > F_{h(i-1)}$, then $T_{hi} = I_{hi} + 1$. Otherwise, $T_{hi} = I_{hi} + 1$ with probability

$$F_{hi}/F_{h(i-1)}$$

Brewer's PPS Method

Brewer's method (METHOD=PPS_BREWER) selects two units from each stratum, with probability proportional to size and without replacement. The selection probability for unit i in stratum h is $2M_{hi}/M_{h\cdot} = 2Z_{hi}$. (Because selection probabilities cannot exceed 1, the relative size for each unit, Z_{hi}, must not exceed $1/2$.)

Brewer's algorithm first selects a unit with probability

$$\frac{Z_{hi}(1 - Z_{hi})}{D_h(1 - 2Z_{hi})}$$

where

$$D_h = \sum_{i=1}^{N_h} \frac{Z_{hi}(1 - Z_{hi})}{1 - 2Z_{hi}}$$

Then a second unit is selected from the remaining units with probability

$$\frac{Z_{hj}}{1 - Z_{hi}}$$

where unit i is the first unit selected. The joint selection probability for units i and j in stratum h is

$$P_{h(ij)} = \frac{2Z_{hi}Z_{hj}}{D_h}\left(\frac{1 - Z_{hi} - Z_{hj}}{(1 - 2Z_{hi})(1 - 2Z_{hj})}\right)$$

For more information, see Cochran (1977, pp. 261–263) and Brewer (1963). Brewer's method yields the same selection probabilities and joint selection probabilities as Durbin's method (Cochran 1977; Durbin 1967).

Murthy's PPS Method

Murthy's method (METHOD=PPS_MURTHY) selects two units from each stratum, with probability proportional to size and without replacement. The selection probability for unit i in stratum h is

$$P_{hi} = Z_{hi}\big(1 + K_h - (Z_{hi}/(1 - Z_{hi}))\big)$$

where $Z_{hi} = M_{hi}/M_{h\cdot}$ and

$$K_h = \sum_{j=1}^{N_h}\big(Z_{hj}/(1 - Z_{hj})\big)$$

Murthy's algorithm first selects a unit with probability Z_{hi}. Then a second unit is selected from the remaining units with probability $Z_{hj}/(1 - Z_{hi})$, where unit i is the first unit selected. The joint selection probability for units i and j in stratum h is

$$P_{h(ij)} = Z_{hi}Z_{hj}\left(\frac{2 - Z_{hi} - Z_{hj}}{(1 - Z_{hi})(1 - Z_{hj})}\right)$$

For more information, see Cochran (1977, pp. 263–265) and Murthy (1957).

Sampford's PPS Method

Sampford's method (METHOD=PPS_SAMPFORD) is an extension of Brewer's method that selects more than two units from each stratum, with probability proportional to size and without replacement. The selection probability for unit i in stratum h is $n_h M_{hi}/M_{h\cdot} = n_h Z_{hi}$. (Because selection probabilities cannot exceed 1, the relative size for each unit, Z_{hi}, must not exceed $1/n_h$.)

Sampford's method first selects a unit from stratum h with probability Z_{hi}. Then subsequent units are selected with probability proportional to

$$\lambda_{hi} = Z_{hi} \,/\, (1 - n_h Z_{hi})$$

and with replacement. If the same unit appears more than once in the sample of size n_h, then Sampford's algorithm rejects that sample and selects a new sample. The sample is accepted if it contains n_h distinct units.

If you specify the JTPROBS option, PROC SURVEYSELECT computes the joint selection probabilities for all pairs of selected units in each stratum. The joint selection probability for units i and j in stratum h is

$$P_{h(ij)} = K_h \, \lambda_{hi} \, \lambda_{hj} \sum_{t=2}^{n_h} \Big(\big[t - n_h (Z_{hi} + Z_{hj}) \big] \; L_{h,(n_h - t)}(\overline{ij}) \Big) \, / \, n_h^{t-2}$$

where

$$K_h = 1 \, / \sum_{t=1}^{n_h} \big(t \; L_{h,(n_h - t)} \, / \, n_h^t \big)$$

$$L_{h,m} = \sum_{S_h(m)} \lambda_{hi_1} \, \lambda_{hi_2} \cdots \lambda_{hi_m}$$

and $S_h(m)$ denotes all possible samples of size m, for $m = 1, 2, \ldots, N_h$. The sum $L_{h,m}(\overline{ij})$ is defined similarly to $L_{h,m}$ but sums over all possible samples of size m that do not include units i and j. For more information, see Cochran (1977, pp. 262–263) and Sampford (1967).

Sample Size Allocation

If you specify the ALLOC= option in the STRATA statement, PROC SURVEYSELECT allocates the total sample size among the strata according to the method that you request. PROC SURVEYSELECT provides proportional allocation (ALLOC=PROPORTIONAL), optimal allocation (ALLOC=OPTIMAL), and Neyman allocation (ALLOC=NEYMAN). For more information about these allocation methods, see Lohr (2010), Kish (1965), and Cochran (1977). You can also directly provide the allocation proportions by using the ALLOC=(*values*) option or the ALLOC=*SAS-data-set* option. Then PROC SURVEYSELECT allocates the sample size among the strata according to the proportions that you provide. Allocation proportions are the relative stratum sample sizes, n_h/n, where n_h is the sample size for stratum h and n is the total sample size.

You can use the SAMPSIZE=*n* option in the PROC SURVEYSELECT statement to specify the total sample size to allocate among the strata. Or you can specify the required margin of error in the MARGIN= option in the STRATA statement, and PROC SURVEYSELECT computes the stratum sample sizes necessary to achieve that margin of error for the allocation method that you request. For more information, see the section "Specifying the Margin of Error" on page 10470.

Proportional Allocation

When you specify the ALLOC=PROPORTIONAL option in the STRATA statement, PROC SURVEYSELECT allocates the total sample size among the strata in proportion to the stratum sizes, where the stratum size is the number of sampling units in the stratum. The allocation proportion of the total sample size for stratum h is

$$f_h^* = N_h / N$$

where N_h is the number of sampling units in stratum h and N is the total number of sampling units for all strata. PROC SURVEYSELECT computes the target sample size for stratum h as

$$n_h^* = f_h^* \times n$$

where n is the total sample size that you specify in the SAMPSIZE= option in the PROC SURVEYSELECT statement.

If you specify a minimum stratum sample size n_{min} in the ALLOCMIN= option in the STRATA statement, then all stratum sample sizes are required to be at least n_{min}. By default, all stratum sample sizes are required to be at least 1 (to ensure that at least one sampling unit is selected from each stratum). If a target sample size is less than the required minimum value, PROC SURVEYSELECT sets the target sample size equal to the minimum value.

PROC SURVEYSELECT computes the allocated stratum sample sizes n_h (which must be integers) by rounding the target sample size values in order of the fractional parts until the total sample size n is achieved.

PROC SURVEYSELECT provides the target allocation proportions f_h^* in the output data set variable AllocProportion. The variable ActualProportion contains the actual proportions for the allocated sample sizes n_h. For stratum h, the actual proportion is computed as

$$f_h = n_h / n$$

where n_h is the allocated sample size for stratum h and n is the total sample size. The actual proportions f_h can differ from the target allocation proportions f_h^* because of rounding and the requirement that $n_h \geq 1$ (or $n_h \geq n_{min}$).

Optimal Allocation

When you specify the ALLOC=OPTIMAL option in the STRATA statement, PROC SURVEYSELECT allocates the total sample size among the strata in proportion to stratum sizes, stratum costs, and stratum variances. You provide the stratum costs and variances in the COST= and VAR= options, respectively.

Optimal allocation minimizes the overall variance for a specified cost, or equivalently minimizes the overall cost for a specified variance. For more information, see Lohr (2010), Cochran (1977), and Kish (1965). For optimal allocation, PROC SURVEYSELECT computes the proportion of the total sample size for stratum h as

$$f_h^* = \frac{N_h S_h}{\sqrt{C_h}} \Big/ \sum_{i=1}^{H} \frac{N_i S_i}{\sqrt{C_i}}$$

where N_h is the number of sampling units in stratum h, S_h is the standard deviation within stratum h, C_h is the unit cost within stratum h, and H is the total number of strata.

PROC SURVEYSELECT computes the target sample size for stratum h as

$$n_h^* = f_h^* \times n$$

where n is the total sample size that you specify in the SAMPSIZE= option in the PROC SURVEYSELECT statement.

If you specify a minimum stratum sample size n_{min} in the ALLOCMIN= option in the STRATA statement, then all stratum sample sizes are required to be at least n_{min}. By default, all stratum sample sizes are required to be at least 1 (to ensure that at least one sampling unit is selected from each stratum). If a target sample size is less than the required minimum value, PROC SURVEYSELECT sets the target sample size equal to the minimum value.

For without-replacement selection methods, a stratum sample size cannot exceed the number of sampling units in the stratum (N_h). If a target stratum sample size exceeds the number of units in the stratum, PROC

SURVEYSELECT allocates the number of available units (N_h) to the stratum and allocates the remaining sample size proportionally among the remaining strata.

PROC SURVEYSELECT computes the allocated stratum sample sizes n_h (which must be integers) by rounding the target sample size values in order of the fractional parts until the total sample size n is achieved.

PROC SURVEYSELECT provides the target allocation proportions f_h^* in the output data set variable AllocProportion. The variable ActualProportion contains the actual proportions for the allocated sample sizes n_h. For stratum h, the actual proportion is computed as

$$f_h = n_h / n$$

where n_h is the allocated sample size for stratum h and n is the total sample size. The actual proportions f_h can differ from the target allocation proportions f_h^* because of rounding and the requirement that $n_h \geq 1$ (or $n_h \geq n_{min}$).

Neyman Allocation

When you specify the ALLOC=NEYMAN option in the STRATA statement, PROC SURVEYSELECT allocates the total sample size among the strata in proportion to stratum sizes and stratum variances. Neyman allocation is a special case of optimal allocation (which is described in the section "Optimal Allocation" on page 10469), where the costs per unit are the same for all strata. For Neyman allocation, PROC SURVEYSELECT computes the proportion of the total sample size for stratum h as

$$f_h^* = N_h S_h \Big/ \sum_{i=1}^{H} N_i S_i$$

where N_h is the number of sampling units in stratum h, S_h is the standard deviation within stratum h, and H is the total number of strata.

PROC SURVEYSELECT computes the target sample size for stratum h as

$$n_h^* = f_h^* \times n$$

where n is the total sample size that you specify in the SAMPSIZE= option in the PROC SURVEYSELECT statement.

The allocated sample sizes n_h are computed from the target sample sizes as described in the section "Optimal Allocation" on page 10469.

Specifying the Margin of Error

Instead of specifying the total sample size to allocate among the strata, you can specify the margin of error for the estimate of the overall mean from the stratified sample. Based on the requested allocation method and the stratum variances that you provide, PROC SURVEYSELECT computes the stratum sample sizes that are required to achieve this margin of error. You specify the margin of error in the MARGIN= option in the STRATA statement, and you provide stratum variances in the VAR= option. You can use the MARGIN= option with any allocation method (proportional, optimal, or Neyman) or with allocation proportions that you provide (ALLOC=(*values*) or ALLOC=*SAS-data-set*).

The margin of error e is the half-width of the $100(1-\alpha)\%$ confidence interval for the overall mean based on the stratified sample,

$$e = z_{\alpha/2} \sqrt{\mathrm{Var}(\bar{y}_{str})}$$

where $\mathrm{Var}(\bar{y}_{str})$ is the variance of the estimate of the mean from the stratified sample and $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution. You can specify the value of α in the ALPHA= option in the STRATA statement. By default, PROC SURVEYSELECT uses a 95% confidence interval (ALPHA=0.05).

For the specified margin of error e, PROC SURVEYSELECT computes the target stratum sample sizes n_h^* for without-replacement selection methods as

$$n_h^* = f_h^* \left(\sum_{i=1}^{H} N_i^2 S_i^2 / f_i^* \right) \Big/ \left((e\, N/z_{\alpha/2})^2 + \sum_{i=1}^{H} N_i S_i^2 \right)$$

where N_i is the number of sampling units in stratum i, S_i^2 is the variance within stratum i, N is the total number of sampling units for all strata, and H is the total number of strata.

The values of f_h^* are the stratum allocation proportions, which PROC SURVEYSELECT computes according to the allocation method that you request. For more information, see the sections "Proportional Allocation" on page 10468, "Optimal Allocation" on page 10469, and "Neyman Allocation" on page 10470.

For with-replacement selection methods, PROC SURVEYSELECT computes the target stratum sample sizes as

$$n_h^* = f_h^* \left(\sum_{i=1}^{H} N_i^2 S_i^2 / f_i^* \right) \Big/ \left(e\, N/z_{\alpha/2} \right)^2$$

For more information, see Lohr (2010, p. 91), Cochran (1977, Chapter 5), and Arkin (1984, Chapter 10).

The target sample size values n_h^* might not be integers, but the stratum sample sizes are required to be integers. PROC SURVEYSELECT rounds all fractional target sample sizes up to integer sample sizes. If you specify a minimum stratum sample size n_{min} in the ALLOCMIN= option in the STRATA statement, then all stratum sample sizes n_h are required to be at least n_{min}.

For without-replacement selection methods, a stratum sample size cannot exceed the number of units in the stratum. If a target stratum sample size does exceed the number of units in the stratum, the procedure sets $n_h = N_h$ for that stratum, removes the stratum from the variance computation (because it contributes nothing to the sampling error), revises the allocation proportions f_h^* for the remaining strata, and computes the stratum sample sizes again. If a stratum sample size equals the number of units in its stratum, the procedure also removes that stratum from the variance computation and revises the sample sizes for the remaining strata. For more information, see Cochran (1977, p. 104) and Arkin (1984, p. 176).

When you specify the STATS option with the MARGIN= option in the STRATA statement, PROC SURVEYSELECT displays the expected margin of error for the sample allocation. The expected margin of error (for the overall mean based on the stratified sample) is computed from the stratum sizes (N_i), the stratum variances that you provide (S_i^2), and the allocated stratum sample sizes that the procedure computes (n_i). For without-replacement selection methods, the expected margin of error is

$$e = (z_{\alpha/2}/N) \sqrt{\sum_{i=1}^{H} (N_i^2 S_i^2 / n_i)\,(1 - n_i/N)}$$

For with-replacement selection methods, the expected margin of error is

$$e = (z_{\alpha/2}/N) \sqrt{\sum_{i=1}^{H} (N_i^2 S_i^2 / n_i)}$$

The expected margin of error should be less than or equal to the value specified in the MARGIN= option. Any difference between the expected margin and the specified value is due to rounding the target stratum sample sizes up to integer values and increasing stratum sample sizes to equal the required minimum value (ALLOCMIN=).

Secondary Input Data Set

The primary input data set for PROC SURVEYSELECT is the DATA= data set, which contains the list of units from which the sample is selected. You can use a secondary input data set to provide stratum-level design and selection information, such as sample sizes or rates, certainty size values, or stratum costs. This secondary input data set is sometimes called the SAMPSIZE= input data set. You can provide stratum sample sizes in the _NSIZE_ (or SampleSize) variable in the SAMPSIZE= data set.

The secondary input data set must contain all the STRATA variables, with the same type and length as in the DATA= data set. The STRATA groups should appear in the same order in the secondary data set as in the DATA= data set. You can name only one secondary data set in each invocation of PROC SURVEYSELECT.

You must name the secondary input data set in the appropriate PROC SURVEYSELECT or STRATA option, and use the designated variable name to provide the stratum-level values. For example, if you want to provide stratum-level costs for sample allocation, you name the secondary data set in the COST=*SAS-data-set* option in the STRATA statement. The data set must include the stratum costs in a variable named _COST_. You can use the secondary input data set for more than one option if it is appropriate for your design. For example, the secondary data set can include both stratum costs and stratum variances, which are required for optimal allocation (ALLOC=OPTIMAL).

Instead of using a separate secondary input data set, you can include secondary information in the DATA= data set along with the sampling frame. When you include secondary information in the DATA= data set, name the DATA= data set in the appropriate options, and include the required variables in the DATA= data set.

Table 124.3 lists the available secondary data set variables, together with their descriptions and the corresponding options.

Table 124.3 PROC SURVEYSELECT Secondary Data Set Variables

Variable	Description	Statement	Option
ALLOC	Allocation proportion	STRATA	ALLOC=
CERTP	Certainty proportion	PROC	CERTSIZE=P=
CERTSIZE	Certainty size	PROC	CERTSIZE=
COST	Cost	STRATA	COST=
MAXSIZE	Maximum size	PROC	MAXSIZE=
MINSIZE	Minimum size	PROC	MINSIZE=
NSIZE	Sample size	PROC	SAMPSIZE=
RATE	Sampling rate	PROC	SAMPRATE=
SEED	Random number seed	PROC	SEED=
VAR	Variance	STRATA	VAR=

Sample Output Data Set

PROC SURVEYSELECT selects a sample and creates a SAS data set that contains the sample of selected units unless you specify the NOSAMPLE option in the STRATA statement or the GROUPS= option in the PROC SURVEYSELECT statement. When you specify the NOSAMPLE option, PROC SURVEYSELECT allocates the total sample size among strata but does not select a sample; the output data set contains the allocated sample sizes. For more information, see the section "Allocation Output Data Set" on page 10477. When you specify the GROUPS= option, PROC SURVEYSELECT randomly assigns observations to groups and does not select a sample. For more information, see the section "Random Assignment Output Data Set" on page 10477.

You can specify the name of the sample output data set in the OUT= option in the PROC SURVEYSELECT statement. If you omit the OUT= option, the data set is named DATA*n*, where *n* is the smallest integer that makes the name unique.

The output data set contains the units that are selected for the sample. These units are either observations or groups of observations (clusters) that you define by specifying the SAMPLINGUNIT statement. If you do not specify the SAMPLINGUNIT statement to define units (clusters), then PROC SURVEYSELECT uses observations as sampling units by default.

By default, the output data set contains only those units that are selected for the sample. If you specify the OUTALL option, the output data set includes all observations from the input data set and also contains a variable that indicates each observation's selection status. For an observation that is selected, the value of the variable Selected is 1; for an observation that is not selected, the value of Selected is 0. The OUTALL option is available for equal probability selection methods.

By default, the output data set contains a single copy of each selected unit, even if the unit is selected more than once, and the variable NumberHits records the number of hits (selections) for each unit. A unit can be selected more than once if you use a with-replacement or with-minimum-replacement selection method (METHOD=URS, METHOD=PPS_WR, METHOD=PPS_SYS, or METHOD=PPS_SEQ). If you specify the OUTHITS option, the output data set includes a distinct copy of each selection in the output data set; for example, the output data set includes three copies of a unit that is selected three times (NumberHits is three).

The output data set also contains design information and selection statistics, depending on the selection method and output options that you specify. The output data set can include the following variables:

- Selected, which indicates whether or not the observation is selected for the sample. This variable is included if you specify the OUTALL option. For an observation that is selected, the value of the variable Selected is 1; for an observation that is not selected, the value of Selected is 0.
- STRATA variables, which you specify in the STRATA statement.
- Replicate, which is the sample replicate number. This variable is included when you specify the REPS= option. You can specify a different name for this variable in the REPNAME= option.
- SAMPLINGUNIT (CLUSTER) variables, which you specify in the SAMPLINGUNIT statement.
- ClusterID, which is a sampling unit (cluster) ID number. This variable is included when you specify the ID option in the SAMPLINGUNIT statement.
- ID variables, which you name in the ID statement.

- CONTROL variables, which you specify in the CONTROL statement.
- Zone, which is the selection zone. This variable is included for METHOD=PPS_SEQ.
- SIZE variable, which you specify in the SIZE statement.
- AdjustedSize, which is the adjusted size measure. This variable is included if you request adjusted sizes with the MINSIZE= or MAXSIZE= option when your sampling units are observations.
- UnitSize, which is the sampling unit (or cluster) size measure. This variable is included if you specify the SAMPLINGUNIT statement.
- Certain, which indicates certainty selection. This variable is included if you specify the CERTSIZE= or CERTSIZE=P= option. For units that are selected with certainty (because their size measures exceed the certainty size value or the certainty proportion), the value of Certain is 1; for other units, the value of Certain is 0.
- NumberHits, which is the number of hits (selections). This variable is included for selection methods that are with replacement or with minimum replacement (METHOD=URS, METHOD=PPS_WR, METHOD=PPS_SYS, and METHOD=PPS_SEQ).

The output data set includes the following variables if you request a PPS selection method or if you specify the STATS option in the PROC SURVEYSELECT statement for other methods:

- ExpectedHits, which is the expected number of hits (selections). This variable is included for selection methods that are with replacement or with minimum replacement, where the same unit can be selected more than once (METHOD=URS, METHOD=BALBOOTSTRAP, METHOD=PPS_WR, METHOD=PPS_SYS, and METHOD=PPS_SEQ).
- SelectionProb, which is the probability of selection. This variable is included for selection methods that are without replacement.
- SamplingWeight, which is the sampling weight. The value of this variable is the inverse of ExpectedHits or SelectionProb.

If you specify the STATS or OUTSIZE option for METHOD=BERNOULLI, the output data set contains the following variables. If you specify a STRATA statement, the output data set includes stratum-level values of these variables; otherwise, the output data set includes overall values.

- Total, which is the total number of sampling units
- SelectionProb, which is the selection probability that you specify in the SAMPRATE= option
- ExpectedN, which is the expected value of the sample size
- SampleSize, which is the actual sample size

If you specify the STATS option for METHOD=BERNOULLI, the output data set also contains the following variable:

- AdjSamplingWeight, which is the adjusted sampling weight

For METHOD=PPS_BREWER and METHOD=PPS_MURTHY, either of which selects two units from each stratum with probability proportional to size, the output data set contains the following variable:

- JtSelectionProb, which is the joint probability of selection for the two units selected from the stratum.

If you specify the JTPROBS option to compute joint probabilities of selection for METHOD=PPS or METHOD=PPS_SAMPFORD, then the output data set contains the following variables:

- Unit, which is an identification variable that numbers the selected units sequentially within each stratum.
- JtProb_1, JtProb_2, JtProb_3, . . . , where the variable JtProb_1 contains the joint probability of selection for the current unit and unit 1. Similarly, JtProb_2 contains the joint probability of selection for the current unit and unit 2, and so on.

If you specify the JTPROBS option for METHOD=PPS_WR, then the output data set contains the following variables:

- Unit, which is an identification variable that numbers the selected units sequentially within each stratum.
- JtHits_1, JtHits_2, JtHits_3, . . . , where the variable JtHits_1 contains the joint expected number of hits for the current unit and unit 1. Similarly, JtHits_2 contains the joint expected number of hits for the current unit and unit 2, and so on.

If you specify the OUTSIZE option, the output data set contains the following variables. If you specify a STRATA statement, the output data set includes stratum-level values of these variables; otherwise, the output data set includes overall values.

- MinimumSize, which is the minimum size measure that you specify in the MINSIZE= option. This variable is included if you specify the MINSIZE= option.
- MaximumSize, which is the maximum size measure that you specify in the MAXSIZE= option. This variable is included if you specify the MAXSIZE= option.
- CertaintySize, which is the certainty size measure that you specify in the CERTSIZE= option. This variable is included if you specify the CERTSIZE= option.
- CertaintyProp, which is the certainty proportion that you specify in the CERTSIZE=P= option. This variable is included if you specify the CERTSIZE=P= option.
- Total, which is the total number of sampling units in the stratum. This variable is included if you do not specify a SIZE statement or a SAMPLINGUNIT statement.
- TotalSize, which is the total of size measures in the stratum. This variable is included if you specify a SIZE statement or the PPS option in the SAMPLINGUNIT statement.

- TotalAdjSize, which is the total of adjusted size measures in the stratum. This variable is included if you request adjusted sizes in the MAXSIZE= or MINSIZE= option.
- SamplingRate, which is the sampling rate. This variable is included if you specify the SAMPRATE= option.
- SampleSize, which is the sample size. This variable is included if you specify the SAMPSIZE= option, or if you specify METHOD=PPS_BREWER or METHOD=PPS_MURTHY, either of which selects two units from each stratum.
- Interval, which is the specified systematic interval. This variable is included if you specify the INTERVAL= option for METHOD=SYS or METHOD=PPS_SYS.
- NCertain, which is the number of certainty units. This variable is included if you specify the CERTSIZE= or CERTSIZE=P= option and CERTUNITS=OUTPUT.

If you specify the OUTSEED option, the output data set contains the following variable:

- InitialSeed, which is the initial seed for the stratum.

If you specify the ALLOC= option in the STRATA statement, the output data set contains the following variables:

- Total, which is the total number of sampling units in the stratum.
- Variance, which is the stratum variance. This variable is included if you specify the VAR, VAR=(*values*), or VAR=*SAS-data-set* option for the ALLOC=OPTIMAL, ALLOC=NEYMAN, or MARGIN= allocation option.
- Cost, which is the stratum cost. This variable is included if you specify the COST, COST=(*values*), or COST=*SAS-data-set* option for ALLOC=OPTIMAL.
- AllocProportion, which is the target allocation proportion (the proportion of the total sample size to allocate to the stratum). PROC SURVEYSELECT computes this proportion by using the allocation method that you specify.
- SampleSize, which is the sample size allocated to the stratum.
- ActualProportion, which is the actual proportion allocated to the stratum. The value of ActualProportion equals the allocated stratum sample size divided by the total sample size. This value can differ from the target AllocProportion because of rounding and other restrictions. For more information, see the section "Sample Size Allocation" on page 10468.

Allocation Output Data Set

When you specify the NOSAMPLE option in the STRATA statement, PROC SURVEYSELECT allocates the total sample size among the strata but does not select the sample. In this case, the OUT= data set contains the allocated sample sizes.

You can specify the name of the allocation output data set with the OUT= option in the PROC SURVEYSELECT statement. If you omit the OUT= option, the data set is named DATA*n*, where *n* is the smallest integer that makes the name unique.

The allocation output data set contains one observation for each stratum. The data set can include the following variables:

- STRATA variables, which you specify in the STRATA statement.
- Total, which is the total number of sampling units in the stratum.
- Variance, which is the stratum variance. This variable is included if you specify the VAR, VAR=(*values*), or VAR=*SAS-data-set* option for the ALLOC=OPTIMAL, ALLOC=NEYMAN, or MARGIN= allocation option.
- Cost, which is the stratum cost. This variable is included if you specify the COST, COST=(*values*), or COST=*SAS-data-set* option for ALLOC=OPTIMAL.
- AllocProportion, which is the target allocation proportion (the proportion of the total sample size to allocate to the stratum). PROC SURVEYSELECT computes this proportion by using the allocation method that you specify.
- SampleSize, which is the sample size allocated to the stratum.
- ActualProportion, which is the actual proportion allocated to the stratum. The value of ActualProportion equals the allocated stratum sample size divided by the total sample size. This value can differ from the target AllocProportion because of rounding and other restrictions. For more information, see the section "Sample Size Allocation" on page 10468.

Random Assignment Output Data Set

When you specify the GROUPS= option, PROC SURVEYSELECT provides random assignment of the observations in the DATA= input data set. The OUT= output data set contains all observations in the input data set and identifies the assigned groups. If you do not specify an ID statement, the output data set contains all variables in the input data set. If you specify an ID statement, PROC SURVEYSELECT copies those variable that you specify from the input data set to the output data set.

You can specify the name of the output data set in the OUT= option in the PROC SURVEYSELECT statement. If you omit the OUT= option, the data set is named DATA*n*, where *n* is the smallest integer that makes the name unique.

The random assignment output data set can include the following variables:

- STRATA variables, if you specify a STRATA statement
- Replicate, which is the replicate identification number. This variable is included when you specify the REPS= option.
- ID variables, if you specify an ID statement
- GroupID, which is the group identification number. If you specify a STRATA statement, PROC SURVEYSELECT performs random assignment independently within strata, and the groups are nested within strata.
- InitialSeed, which is the initial seed for random number generation

If you specify the OUTSIZE option, the random assignment output data set also includes the following variables:

- Total, which is the total number of units in the data set, or the total in the stratum if you specify a STRATA statement
- NGroups, which is the number of groups in the data set, or the number in the stratum if you specify a STRATA statement
- GroupSize, which is the number of units in the observation's group

Displayed Output

By default, PROC SURVEYSELECT displays two tables that summarize the sample selection: the "Sample Selection Method" table and the "Sample Selection Summary" table.

If you request sample allocation but no sample selection, PROC SURVEYSELECT displays two tables that summarize the allocation: the "Sample Allocation Method" table and the "Sample Allocation Summary" table.

If you request random assignment, the procedure displays the "Random Assignment" table.

You can suppress display of these tables by specifying the NOPRINT option.

PROC SURVEYSELECT creates an output data set that contains the units that are selected for the sample. Or if you request sample allocation but no sample selection, PROC SURVEYSELECT creates an output data set that contains the sample size allocation results. If you request random assignment, the procedure creates an output data set that contains the assignments. For more information, see the sections "Sample Output Data Set" on page 10473, "Allocation Output Data Set" on page 10477, and "Random Assignment Output Data Set" on page 10477. The procedure does not display the output data set that it creates. Use PROC PRINT, PROC REPORT, or any other SAS reporting tool to display the output data set.

Sample Selection Method Table

PROC SURVEYSELECT displays the following information in the "Sample Selection Method" table:

- Selection Method
- Sampling Unit Variables, if you specify a SAMPLINGUNIT statement
- Size Measure variable, if you specify a SIZE statement
- Size Measure: Number of Observations, if you specify the PPS option in the SAMPLINGUNIT statement and do not specify a SIZE statement
- Minimum Size Measure, if you specify the MINSIZE= option
- Maximum Size Measure, if you specify the MAXSIZE= option
- Certainty Size Measure, if you specify the CERTSIZE= option
- Certainty Proportion, if you specify the CERTSIZE=P= option
- Strata Variables, if you specify a STRATA statement
- Control Variables, if you specify a CONTROL statement
- Control Sorting (Serpentine or Nested), if you specify a CONTROL statement
- Allocation (Proportional, Neyman, Optimal, or Input), if you specify the ALLOC= option in the STRATA statement
- Margin of Error, if you specify the MARGIN= option in the STRATA statement
- Confidence Level, if you specify the ALPHA= option in the STRATA statement

Sample Selection Summary Table

PROC SURVEYSELECT displays the following information in the "Sample Selection Summary" table:

- Input Data Set name
- Sorted Data Set name, if you specify the OUTSORT= option
- Random Number Seed
- Sample Size or Stratum Sample Size, if you specify the SAMPSIZE=*n* option
- Sample Size Data Set, if you specify the SAMPSIZE=*SAS-data-set* option
- Sampling Rate or Stratum Sampling Rate, if you specify the SAMPRATE=*value* option for METHOD=SRS, METHOD=URS, METHOD=SYS, or METHOD=SEQ.
- Selection Probability or Stratum Selection Probability, if you specify the SAMPRATE=*value* option for METHOD=BERNOULLI
- Sampling Rate Data Set, if you specify the SAMPRATE=*SAS-data-set* option

- Minimum Sample Size or Stratum Minimum Sample Size, if you specify the NMIN= option in the SAMPRATE= option
- Maximum Sample Size or Stratum Maximum Sample Size, if you specify the NMAX= option in the SAMPRATE= option
- Number of Certainty Units, if you specify the CERTSIZE= or CERTSIZE=P= option and do not specify a STRATA statement
- Specified Start, if you specify the START= option for METHOD=SYS or METHOD=PPS_SYS
- Random Start, if you specify the DETAILS option for METHOD=SYS or METHOD=PPS_SYS and do not specify a STRATA statement or the REPS= option
- Specified Interval, if you specify the INTERVAL= option for METHOD=SYS or METHOD=PPS_SYS
- Systematic Interval, if you specify the DETAILS option for METHOD=SYS or METHOD=PPS_SYS and do not specify a STRATA statement or the REPS= option
- Sample Size, if you specify the INTERVAL= option for METHOD=SYS or METHOD=PPS_SYS and do not specify a STRATA statement or the REPS= option
- Allocation Input Data Set name, if you specify the ALLOC=*SAS-data-set* option in the STRATA statement
- Variance Input Data Set name, if you specify the VAR=*SAS-data-set* option in the STRATA statement
- Cost Input Data Set name, if you specify the COST=*SAS-data-set* option in the STRATA statement
- Selection Probability, if you specify METHOD=SRS, METHOD=SYS, or METHOD=SEQ and do not specify a SIZE statement or a STRATA statement
- Expected Number of Hits, if you specify METHOD=URS and do not specify a STRATA statement
- Total Number of Units, if you specify METHOD=BERNOULLI or METHOD=POISSON and do not specify a STRATA statement
- Expected Sample Size, if you specify METHOD=BERNOULLI or METHOD=POISSON and do not specify a STRATA statement
- Sample Size, if you specify METHOD=BERNOULLI or METHOD=POISSON and do not specify a STRATA statement
- Sampling Weight, if you specify an equal probability selection method (METHOD=SRS, METHOD=URS, METHOD=SYS, METHOD=SEQ, or METHOD=BERNOULLI) and do not specify a STRATA statement
- Adjusted Sampling Weight, if you specify METHOD=BERNOULLI and do not specify a STRATA statement
- Number of Strata, if you specify a STRATA statement
- Stratum Minimum Sample Size, if you specify the ALLOCMIN= option in the STRATA statement
- Number of Replicates, if you specify the REPS= option

- Total Sample Size, if you specify a STRATA statement or the REPS= option
- Expected Margin of Error, if you specify the STATS option with the MARGIN= option in the STRATA statement
- Expected Variance, if you specify the STATS option without the MARGIN= option in the STRATA statement for ALLOC=OPTIMAL or ALLOC=NEYMAN
- Total Stratum Costs, if you specify the STATS option with ALLOC=OPTIMAL in the STRATA statement
- Output Data Set name

Sample Allocation Method Table

If you specify the NOSAMPLE option in the STRATA statement, PROC SURVEYSELECT allocates the total sample among the strata but does not select the sample. When you specify the NOSAMPLE option, PROC SURVEYSELECT displays the "Sample Allocation Method" table and the "Sample Allocation Summary" table. The "Sample Allocation Method" table includes the following information:

- Allocation (Proportional, Neyman, Optimal, or Input)
- Margin of Error, if you specify the MARGIN= option in the STRATA statement
- Confidence Level, if you specify the ALPHA= option in the STRATA statement
- Sampling Unit Variables, if you specify a SAMPLINGUNIT statement
- Strata Variables
- Frequency Variable
- Selection Method, if you specify the METHOD= option

Sample Allocation Summary Table

PROC SURVEYSELECT displays the following information in the "Sample Allocation Summary" table.

- Input Data Set name
- Allocation Input Data Set name, if you specify the ALLOC=*SAS-data-set* option in the STRATA statement
- Variance Input Data Set name, if you specify the VAR=*SAS-data-set* option in the STRATA statement
- Cost Input Data Set name, if you specify the COST=*SAS-data-set* option in the STRATA statement
- Number of Strata
- Stratum Minimum Sample Size, if you specify the ALLOCMIN= option in the STRATA statement
- Total Sample Size

- Expected Margin of Error, if you specify the STATS option with the MARGIN= option in the STRATA statement
- Expected Variance, if you specify the STATS option without the MARGIN= option in the STRATA statement for ALLOC=OPTIMAL or ALLOC=NEYMAN
- Total Stratum Costs, if you specify the STATS option with ALLOC=OPTIMAL in the STRATA statement
- Allocation Output Data Set name

Random Assignment Table

If you specify the GROUPS= option, PROC SURVEYSELECT displays the following information in the "Random Assignment" table:

- Input Data Set name
- Strata Variables, if you specify a STRATA statement
- Random Number Seed
- Number of Groups
- Total Number of Units, if you specify the GROUPS= option and do not specify a STRATA statement
- Number of Units per Group, if you specify the GROUPS= option and do not specify a STRATA statement
- Number of Replicates, if you specify the REPS= option
- Number of Strata, if you specify a STRATA statement
- Total Number of Groups, if you specify a STRATA statement or the REPS= option
- Output Data Set name

ODS Table Names

PROC SURVEYSELECT assigns a name to each table that it creates. You can use these names to refer to tables when you use the Output Delivery System (ODS) to select tables and create output data sets. For more information about ODS, see Chapter 23, "Using the Output Delivery System." Table 124.4 lists the table names.

Table 124.4 ODS Tables Produced by PROC SURVEYSELECT

ODS Table Name	Description	Statement	Option
Groups	Random assignment summary	PROC	GROUPS=
Method	Sample selection method	PROC	Default
Method	Sample allocation method	STRATA	NOSAMPLE
Summary	Sample selection summary	PROC	Default
Summary	Sample allocation summary	STRATA	NOSAMPLE

Examples: SURVEYSELECT Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-stat`.

Example 124.1: Replicated Sampling

(View the complete code for this example (sursex1.sas) in the example repository.)

This example uses the Customers data set from the section "Getting Started: SURVEYSELECT Procedure" on page 10414. The data set Customers contains an Internet service provider's current subscribers, and the service provider wants to select a sample from this population for a customer satisfaction survey.

This example illustrates replicated sampling, which selects multiple samples from the survey population according to the same design. You can use replicated sampling to provide a simple method of variance estimation, or to evaluate variable nonsampling errors such as interviewer differences. For information about replicated sampling, see Lohr (2010), Wolter (2007), Kish (1965), Kish (1987), and Kalton (1983).

This design includes four replicates, which each have a sample size of 50 customers. The sampling frame is stratified by State and sorted by Type and Usage within strata. Customers are selected by sequential random sampling with equal probability within strata. The following PROC SURVEYSELECT statements select a probability sample of customers from the Customers data set by using this design:

```
title1 'Customer Satisfaction Survey';
title2 'Replicated Sampling';
proc surveyselect data=Customers method=seq n=(8 12 20 10)
                  reps=4 seed=40070 ranuni out=SampleRep;
   strata State;
   control Type Usage;
run;
```

The STRATA statement names the stratification variable State. The CONTROL statement names the control variables Type and Usage.

In the PROC SURVEYSELECT statement, the METHOD=SEQ option requests sequential random sampling. The REPS= option specifies 4 replicates of this sample. The N=(8 12 20 10) option specifies the stratum sample sizes in each replicate. The N= option lists the stratum sample sizes in the same order as the strata appear in the Customers data set, which is sorted by State. The sample size of 8 customers corresponds to

the first stratum, (State = ‘AL’); the sample size of 12 customers corresponds to the second stratum (State = ‘FL’), and so on.

The SEED= option specifies 40070 as the initial seed for random number generation. The RANUNI option requests random number generation by the RANUNI random number generator.

Output 124.1.1 displays the output from PROC SURVEYSELECT, which summarizes the sample selection. A total of 200 customers is selected in 4 replicates. PROC SURVEYSELECT selects each replicate by using sequential random sampling within strata that are determined by State. The sampling frame Customers is sorted by the control variables Type and Usage within strata, according to hierarchic serpentine sorting. The output data set SampleRep contains the sample.

Output 124.1.1 Sample Selection Summary

Customer Satisfaction Survey
Replicated Sampling

The SURVEYSELECT Procedure

Selection Method	Sequential Random Sampling With Equal Probability
Strata Variable	State
Control Variables	Type Usage
Control Sorting	Serpentine

Input Data Set	CUSTOMERS
Random Number Seed	40070
Number of Strata	4
Number of Replicates	4
Total Sample Size	200
Output Data Set	SAMPLEREP

The following PROC PRINT statements display the selected customers for the first stratum (State = ‘AL’) in the output data set SampleRep:

```
title1 'Customer Satisfaction Survey';
title2 'Sample Selected by Replicated Design';
title3 '(First Stratum)';
proc print data=SampleRep;
   where State = 'AL';
run;
```

Output 124.1.2 displays the 32 sample customers in the first stratum (State = ‘AL’) from the output data set SampleRep, which includes the entire sample of 200 customers. The variable SelectionProb contains the selection probability, and SamplingWeight contains the sampling weight. Because customers are selected with equal probability within strata in this design, all customers in the same stratum have the same selection probability. These selection probabilities and sampling weights apply to a single replicate, and the variable Replicate contains the sample replicate number.

Output 124.1.2 Customer Sample (First Stratum)

Customer Satisfaction Survey
Sample Selected by Replicated Design
(First Stratum)

Obs	State	Replicate	CustomerID	Type	Usage	SelectionProb	SamplingWeight
1	AL	1	882-37-7496	New	572	.004115226	243
2	AL	1	581-32-5534	New	863	.004115226	243
3	AL	1	980-29-2898	Old	571	.004115226	243
4	AL	1	172-56-4743	Old	128	.004115226	243
5	AL	1	998-55-5227	Old	35	.004115226	243
6	AL	1	625-44-3396	New	60	.004115226	243
7	AL	1	627-48-2509	New	114	.004115226	243
8	AL	1	257-66-6558	New	172	.004115226	243
9	AL	2	622-83-1680	New	22	.004115226	243
10	AL	2	343-57-1186	New	53	.004115226	243
11	AL	2	976-05-3796	New	110	.004115226	243
12	AL	2	859-74-0652	New	303	.004115226	243
13	AL	2	476-48-1066	New	839	.004115226	243
14	AL	2	109-27-8914	Old	2102	.004115226	243
15	AL	2	743-25-0298	Old	376	.004115226	243
16	AL	2	722-08-2215	Old	105	.004115226	243
17	AL	3	668-57-7696	New	200	.004115226	243
18	AL	3	300-72-0129	New	471	.004115226	243
19	AL	3	073-60-0765	New	656	.004115226	243
20	AL	3	526-87-0258	Old	672	.004115226	243
21	AL	3	726-61-0387	Old	150	.004115226	243
22	AL	3	632-29-9020	Old	51	.004115226	243
23	AL	3	417-17-8378	New	56	.004115226	243
24	AL	3	091-26-2366	New	93	.004115226	243
25	AL	4	336-04-1288	New	419	.004115226	243
26	AL	4	827-04-7407	New	650	.004115226	243
27	AL	4	317-70-6496	Old	452	.004115226	243
28	AL	4	002-38-4582	Old	206	.004115226	243
29	AL	4	181-83-3990	Old	33	.004115226	243
30	AL	4	675-34-7393	New	47	.004115226	243
31	AL	4	228-07-6671	New	65	.004115226	243
32	AL	4	298-46-2434	New	161	.004115226	243

Example 124.2: PPS Selection of Two Units per Stratum

(View the complete code for this example (sursex2.sas) in the example repository.)

This example describes hospital selection for a survey by using PROC SURVEYSELECT. A state health agency plans to conduct a statewide survey of a variety of different hospital services. The agency plans to select a probability sample of individual discharge records within hospitals by using a two-stage sample design. First-stage units are hospitals, and second-stage units are patient discharges during the study period. Hospitals are stratified first according to geographic region and then by rural/urban type and size of hospital. Two hospitals are selected from each stratum with probability proportional to size.

The data set HospitalFrame contains all hospitals in the first geographical region of the state:

```
data HospitalFrame;
   input Hospital$ Type$ SizeMeasure @@;
   if (SizeMeasure < 20) then Size='Small ';
      else if (SizeMeasure < 50) then Size='Medium';
      else Size='Large ';
   datalines;
034 Rural  0.870   107 Rural  1.316
079 Rural  2.127   223 Rural  3.960
236 Rural  5.279   165 Rural  5.893
086 Rural  0.501   141 Rural 11.528
042 Urban  3.104   124 Urban  4.033
006 Urban  4.249   261 Urban  4.376
195 Urban  5.024   190 Urban 10.373
038 Urban 17.125   083 Urban 40.382
259 Urban 44.942   129 Urban 46.702
133 Urban 46.992   218 Urban 48.231
026 Urban 61.460   058 Urban 65.931
119 Urban 66.352
;
```

In the SAS data set HospitalFrame, the variable Hospital identifies the hospital. The variable Type equals ‘Urban’ if the hospital is located in an urban area, and ‘Rural’ otherwise. The variable SizeMeasure contains the hospital’s size measure, which is constructed from past data on service utilization for the hospital together with the target sampling rates for each service. This size measure reflects the amount of relevant survey information expected from the hospital. For information about this type of size measure, see Drummond et al. (1982). The value of the variable Size is ‘Small’, ‘Medium’, or ‘Large’, depending on the value of the hospital’s size measure.

The following PROC PRINT statements display the data set Hospital Frame and produce Output 124.2.1:

```
title1 'Hospital Utilization Survey';
title2 'Sampling Frame, Region 1';
proc print data=HospitalFrame;
run;
```

Output 124.2.1 Sampling Frame

Hospital Utilization Survey
Sampling Frame, Region 1

Obs	Hospital	Type	SizeMeasure	Size
1	034	Rural	0.870	Small
2	107	Rural	1.316	Small
3	079	Rural	2.127	Small
4	223	Rural	3.960	Small
5	236	Rural	5.279	Small
6	165	Rural	5.893	Small
7	086	Rural	0.501	Small
8	141	Rural	11.528	Small
9	042	Urban	3.104	Small
10	124	Urban	4.033	Small
11	006	Urban	4.249	Small
12	261	Urban	4.376	Small
13	195	Urban	5.024	Small
14	190	Urban	10.373	Small
15	038	Urban	17.125	Small
16	083	Urban	40.382	Medium
17	259	Urban	44.942	Medium
18	129	Urban	46.702	Medium
19	133	Urban	46.992	Medium
20	218	Urban	48.231	Medium
21	026	Urban	61.460	Large
22	058	Urban	65.931	Large
23	119	Urban	66.352	Large

The following PROC SURVEYSELECT statements select a probability sample of hospitals from the HospitalFrame data set by using a stratified design with PPS selection of two units from each stratum:

```
title1 'Hospital Utilization Survey';
title2 'Stratified PPS Sampling';
proc surveyselect data=HospitalFrame method=pps_brewer
                  seed=48702 out=SampleHospitals;
   size SizeMeasure;
   strata Type Size notsorted;
run;
```

The STRATA statement names the stratification variables Type and Size. The NOTSORTED option specifies that observations with the same STRATA variable values are grouped together but are not necessarily sorted in alphabetical or increasing numerical order. In the HospitalFrame data set, Size = ‘Small’ precedes Size = ‘Medium’.

In the PROC SURVEYSELECT statement, the METHOD=PPS_BREWER option requests sample selection by Brewer’s method, which selects two units per stratum with probability proportional to size. The SEED= option specifies 48702 as the initial seed for random number generation. The SIZE statement names SizeMeasure as the size measure variable. It is not necessary to specify the sample size in the N= option because Brewer’s method selects two units from each stratum.

Output 124.2.2 displays the output from PROC SURVEYSELECT. A total of 8 hospitals are selected from the 4 strata. The data set SampleHospitals contains the selected hospitals.

Output 124.2.2 Sample Selection Summary

Hospital Utilization Survey
Stratified PPS Sampling

The SURVEYSELECT Procedure

Selection Method	Brewer's PPS Method
Size Measure	SizeMeasure
Strata Variables	Type
	Size

Input Data Set	HOSPITALFRAME
Random Number Seed	48702
Stratum Sample Size	2
Number of Strata	4
Total Sample Size	8
Output Data Set	SAMPLEHOSPITALS

The following PROC PRINT statements display the sample hospitals and produce Output 124.2.3:

```
title1 'Hospital Utilization Survey';
title2 'Sample Selected by Stratified PPS Design';
proc print data=SampleHospitals;
run;
```

Output 124.2.3 Sample Hospitals

Hospital Utilization Survey
Sample Selected by Stratified PPS Design

Obs	Type	Size	Hospital	SizeMeasure	SelectionProb	SamplingWeight	JtSelectionProb
1	Rural	Small	165	5.893	0.37447	2.67046	0.22465
2	Rural	Small	141	11.528	0.73254	1.36511	0.22465
3	Urban	Small	190	10.373	0.42967	2.32739	0.25370
4	Urban	Small	038	17.125	0.70934	1.40975	0.25370
5	Urban	Medium	083	40.382	0.35540	2.81374	0.08953
6	Urban	Medium	133	46.992	0.41357	2.41795	0.08953
7	Urban	Large	026	61.460	0.63445	1.57617	0.31940
8	Urban	Large	119	66.352	0.68495	1.45996	0.31940

The variable SelectionProb contains the selection probability for each hospital in the sample. The variable JtSelectionProb contains the joint probability of selection for the two sample hospitals in the same stratum. The variable SamplingWeight contains the sampling weight component for this first stage of the design. The final-stage weight components, which correspond to patient record selection within hospitals, can be multiplied by the hospital weight components to obtain the overall sampling weights.

Example 124.3: PPS (Dollar-Unit) Sampling

(View the complete code for this example (sursex3.sas) in the example repository.)

A small company wants to audit employee travel expenses in an effort to improve the expense reporting procedure and possibly reduce expenses. The company does not have resources to examine all expense reports and wants to use statistical sampling to objectively select expense reports for audit.

The data set TravelExpense contains the dollar amount of all employee travel expense transactions during the past month:

```
data TravelExpense;
   input ID$ Amount @@;
   if (Amount < 500) then Level='1_Low ';
      else if (Amount > 1500) then Level='3_High';
      else Level='2_Avg ';
   datalines;
110  237.18   002  567.89   234  118.50
743   74.38   411 1287.23   782  258.10
216  325.36   174  218.38   568 1670.80
302  134.71   285 2020.70   314   47.80
139 1183.45   775  330.54   425  780.10
506  895.80   239  620.10   011  420.18
672  979.66   142  810.25   738  670.85
192  314.58   243   87.50   263 1893.40
496  753.30   332  540.65   486 2580.35
614  230.56   654  185.60   308  688.43
784  505.14   017  205.48   162  650.42
289 1348.34   691   30.50   545 2214.80
517  940.35   382  217.85   024  142.90
478  806.90   107  560.72
;
```

In the SAS data set TravelExpense, the variable ID identifies the travel expense report. The variable Amount contains the dollar amount of the reported expense. The variable Level equals '1_Low', '2_Avg', or '3_High', depending on the value of Amount.

In the sample design for this audit, expense reports are stratified by Level. This ensures that each of these expense levels is included in the sample and also permits a disproportionate allocation of the sample, selecting proportionately more of the expense reports from the higher levels. Within strata, the sample of expense reports is selected with probability proportional to the amount of the expense, thus giving a greater chance of selection to larger expenses. In auditing terms, this is known as monetary-unit sampling. For more information, see Wilburn (1984).

PROC SURVEYSELECT requires that the input data set be sorted by the STRATA variables. The following PROC SORT statements sort the TravelExpense data set by the stratification variable Level.

```
proc sort data=TravelExpense;
   by Level;
run;
```

Output 124.3.1 displays the sampling frame data set TravelExpense, which contains 41 observations.

Output 124.3.1 Sampling Frame

Travel Expense Audit

Obs	ID	Amount	Level
1	110	237.18	1_Low
2	234	118.50	1_Low
3	743	74.38	1_Low
4	782	258.10	1_Low
5	216	325.36	1_Low
6	174	218.38	1_Low
7	302	134.71	1_Low
8	314	47.80	1_Low
9	775	330.54	1_Low
10	011	420.18	1_Low
11	192	314.58	1_Low
12	243	87.50	1_Low
13	614	230.56	1_Low
14	654	185.60	1_Low
15	017	205.48	1_Low
16	691	30.50	1_Low
17	382	217.85	1_Low
18	024	142.90	1_Low
19	002	567.89	2_Avg
20	411	1287.23	2_Avg
21	139	1183.45	2_Avg
22	425	780.10	2_Avg
23	506	895.80	2_Avg
24	239	620.10	2_Avg
25	672	979.66	2_Avg
26	142	810.25	2_Avg
27	738	670.85	2_Avg
28	496	753.30	2_Avg
29	332	540.65	2_Avg
30	308	688.43	2_Avg
31	784	505.14	2_Avg
32	162	650.42	2_Avg
33	289	1348.34	2_Avg
34	517	940.35	2_Avg
35	478	806.90	2_Avg
36	107	560.72	2_Avg
37	568	1670.80	3_High
38	285	2020.70	3_High
39	263	1893.40	3_High
40	486	2580.35	3_High
41	545	2214.80	3_High

The following PROC SURVEYSELECT statements select a probability sample of expense reports from the TravelExpense data set by using the stratified design with PPS selection within strata:

```
title1 'Travel Expense Audit';
title2 'Stratified PPS (Dollar-Unit) Sampling';
proc surveyselect data=TravelExpense method=pps n=(6 10 4)
                  seed=47279 out=AuditSample;
   size Amount;
   strata Level;
run;
```

The STRATA statement names the stratification variable Level. The SIZE statement specifies the size measure variable Amount. In the PROC SURVEYSELECT statement, the METHOD=PPS option requests sample selection with probability proportional to size and without replacement. The N=(6 10 4) option specifies the stratum sample sizes by listing the sample sizes in the same order as the strata appear in the TravelExpense data set. The sample size of 6 corresponds to the first stratum (Level = '1_Low'); the sample size of 10 corresponds to the second stratum (Level = '2_Avg'); and the sample size of 4 corresponds to the last stratum (Level = '3_High'). The SEED= option specifies 47279 as the initial seed for random number generation.

Output 124.3.2 displays the output from PROC SURVEYSELECT. A total of 20 expense reports are selected for audit. The data set AuditSample contains the sample of travel expense reports.

Output 124.3.2 Sample Selection Summary

Travel Expense Audit
Stratified PPS (Dollar-Unit) Sampling

The SURVEYSELECT Procedure

Selection Method	PPS, Without Replacement
Size Measure	Amount
Strata Variable	Level

Input Data Set	TRAVELEXPENSE
Random Number Seed	47279
Number of Strata	3
Total Sample Size	20
Output Data Set	AUDITSAMPLE

The following PROC PRINT statements display the audit sample, which is shown in Output 124.3.3:

```
title1 'Travel Expense Audit';
title2 'Sample Selected by Stratified PPS Design';
proc print data=AuditSample;
run;
```

Output 124.3.3 Audit Sample

Travel Expense Audit
Sample Selected by Stratified PPS Design

Obs	Level	ID	Amount	SelectionProb	SamplingWeight
1	1_Low	024	142.90	0.23949	4.17553
2	1_Low	614	230.56	0.38640	2.58797
3	1_Low	110	237.18	0.39750	2.51574
4	1_Low	782	258.10	0.43256	2.31183
5	1_Low	192	314.58	0.52721	1.89676
6	1_Low	216	325.36	0.54528	1.83392
7	2_Avg	239	620.10	0.42503	2.35278
8	2_Avg	308	688.43	0.47186	2.11925
9	2_Avg	496	753.30	0.51633	1.93676
10	2_Avg	478	806.90	0.55307	1.80810
11	2_Avg	142	810.25	0.55536	1.80063
12	2_Avg	517	940.35	0.64454	1.55151
13	2_Avg	672	979.66	0.67148	1.48925
14	2_Avg	139	1183.45	0.81116	1.23280
15	2_Avg	411	1287.23	0.88229	1.13341
16	2_Avg	289	1348.34	0.92418	1.08204
17	3_High	568	1670.80	0.64385	1.55316
18	3_High	263	1893.40	0.72963	1.37056
19	3_High	545	2214.80	0.85348	1.17167
20	3_High	486	2580.35	0.99435	1.00568

Example 124.4: Proportional Allocation

(View the complete code for this example (sursex4.sas) in the example repository.)

This example uses the Customers data set from the section "Getting Started: SURVEYSELECT Procedure" on page 10414. The data set Customers contains an Internet service provider's current subscribers, and the service provider wants to select a sample from this population for a customer satisfaction survey. This example illustrates proportional allocation, which allocates the total sample size among the strata in proportion to the strata sizes.

The section "Getting Started: SURVEYSELECT Procedure" on page 10414 gives an example of stratified sampling, where the list of customers is stratified by State and Type. Figure 124.4 displays the strata in a table of State by Type for the 13,471 customers. There are four states and two levels of Type, forming a total of eight strata. A sample of 15 customers was selected from each stratum by using the following PROC SURVEYSELECT statements:

```
title1 'Customer Satisfaction Survey';
title2 'Stratified Sampling';
proc surveyselect data=Customers method=srs n=15
                  seed=1953 out=SampleStrata;
   strata State Type;
run;
```

The STRATA statement names the stratification variables State and Type. In the PROC SURVEYSELECT statement, the N= option specifies a sample size of 15 customers in each stratum.

Instead of specifying the number of customers to select from each stratum, you can specify the total sample size and request allocation of the total sample size among the strata. The following PROC SURVEYSELECT statements request proportional allocation, which allocates the total sample size in proportion to the stratum sizes:

```
title1 'Customer Satisfaction Survey';
title2 'Proportional Allocation';
proc surveyselect data=Customers n=1000
                  out=SampleSizes;
   strata State Type / alloc=prop nosample;
run;
```

The STRATA statement names the stratification variables State and Type. In the STRATA statement, the ALLOC=PROP option requests proportional allocation. The NOSAMPLE option requests that no sample be selected after the procedure computes the sample size allocation. In the PROC SURVEYSELECT statement, the N= option specifies a total sample size of 1000 customers to be allocated among the strata.

Output 124.4.1 displays the output from PROC SURVEYSELECT, which summarizes the sample allocation. The total sample size of 1000 is allocated among the eight strata by using proportional allocation. The allocated sample sizes are stored in the SAS data set SampleSizes.

Output 124.4.1 Proportional Allocation Summary

Customer Satisfaction Survey
Proportional Allocation

The SURVEYSELECT Procedure

Allocation	Proportional
Strata Variables	State
	Type

Input Data Set	CUSTOMERS
Number of Strata	8
Total Sample Size	1000
Allocation Output Data Set	SAMPLESIZES

The following PROC PRINT statements display the allocation output data set SampleSizes, which is shown in Output 124.4.2:

```
title1 'Customer Satisfaction Survey';
title2 'Proportional Allocation';
proc print data=SampleSizes;
run;
```

Output 124.4.2 Stratum Sample Sizes

Customer Satisfaction Survey
Proportional Allocation

Obs	State	Type	Total	AllocProportion	SampleSize	ActualProportion
1	AL	New	1238	0.09190	92	0.092
2	AL	Old	706	0.05241	52	0.052
3	FL	New	2170	0.16109	161	0.161
4	FL	Old	1370	0.10170	102	0.102
5	GA	New	3488	0.25893	259	0.259
6	GA	Old	1940	0.14401	144	0.144
7	SC	New	1684	0.12501	125	0.125
8	SC	Old	875	0.06495	65	0.065

The output data set SampleSizes includes one observation for each of the eight strata, which are identified by the stratification variables State and Type. The variable Total contains the number of sampling units in the stratum, and the variable AllocProportion contains the proportion of the total sample size to allocate to the stratum. The variable SampleSize contains the allocated stratum sample size. For the first stratum (State='AL' and Type='New'), the total number of sampling units is 1238 customers, the allocation proportion is 0.09190, and the allocated sample size is 92 customers. The sum of the allocated sample sizes equals the requested total sample size of 1000 customers.

The output data set also includes the variable ActualProportion, which contains actual stratum proportions of the total sample size. The actual proportion for a stratum is the stratum sample size divided by the total sample size. For the first stratum (State='AL' and Type='New'), the actual proportion is 0.092, whereas the allocation proportion is 0.09190. The target sample sizes computed from the allocation proportions are often not integers, and PROC SURVEYSELECT uses a rounding algorithm to obtain integer sample sizes and maintain the requested total sample size. Because of rounding and other restrictions, the actual proportions can differ from the target allocation proportions. For more information, see the section "Sample Size Allocation" on page 10468.

To use the allocated sample sizes in a later invocation of PROC SURVEYSELECT, you can name the allocation data set in the N=*SAS-data-set* option, as shown in the following PROC SURVEYSELECT statements:

```
title1 'Customer Satisfaction Survey';
title2 'Stratified Sampling';
proc surveyselect data=Customers method=srs n=SampleSizes
                  seed=1953 out=SampleStrata;
   strata State Type;
run;
```

References

Arkin, H. (1984). *Handbook of Sampling for Auditing and Accounting*. New York: McGraw-Hill.

Bentley, J. L., and Floyd, R. W. (1987). "Programming Pearls: A Sample of Brilliance." *Communications of the Association for Computing Machinery* 30:754–757.

Bentley, J. L., and Knuth, D. E. (1986). "Literate Programming." *Communications of the Association for Computing Machinery* 29:364–369.

Brewer, K. W. R. (1963). "A Model of Systematic Sampling with Unequal Probabilities." *Australian Journal of Statistics* 5:93–105.

Cassell, D. L. (2007). "Don't Be Loopy: Re-sampling and Simulation the SAS Way." In *Proceedings of the SAS Global Forum 2007 Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/forum2007/183-2007.pdf.

Chromy, J. R. (1979). "Sequential Sample Selection Methods." In *Proceedings of the Survey Research Methods Section*, 401–406. Washington, DC: American Statistical Association.

Cochran, W. G. (1977). *Sampling Techniques*. 3rd ed. New York: John Wiley & Sons.

Davison, A. C., Hinkley, D. V., and Schechtman, E. (1986). "Efficient Bootstrap Computation." *Biometrika* 73:555–566.

Drummond, D., Lessler, J., Watts, D., and Williams, S. (1982). "A Design for Achieving Prespecified Levels of Representation for Multiple Domains in Health Record Samples." In *Proceedings of the Fourth Conference on Health Survey Research Methods.* DHHS Publication No. (PHS) 84-3346, 233–248. Washington, DC: National Center for Health Services Research.

Durbin, J. (1967). "Design of Multi-stage Surveys for the Estimation of Sampling Errors." *Journal of the Royal Statistical Society, Series C* 16:152–164.

Fan, C. T., Muller, M. E., and Rezucha, I. (1962). "Development of Sampling Plans by Using Sequential (Item by Item) Selection Techniques and Digital Computers." *Journal of the American Statistical Association* 57:387–402.

Fishman, G. S., and Moore, L. R. (1982). "A Statistical Evaluation of Multiplicative Congruential Generators with Modulus $(2^{31} - 1)$." *Journal of the American Statistical Association* 77:129–136.

Fox, D. R. (1989). "Computer Selection of Size-Biased Samples." *American Statistician* 43:168–171.

Gleason, J. R. (1988). "Algorithms for Balanced Bootstrap Simulations." *American Statistician* 42:263–266.

Golmant, J. (1990). "Correction: Computer Selection of Size-Biased Samples." *American Statistician* 44:194.

Hanurav, T. V. (1967). "Optimum Utilization of Auxiliary Information: π_{ps} Sampling of Two Units from a Stratum." *Journal of the Royal Statistical Society, Series B* 29:374–391.

Kalton, G. (1983). *Introduction to Survey Sampling.* Vol. 07-035 of University Paper Series on Quantitative Applications in the Social Sciences. Beverly Hills, CA: Sage Publications.

Kish, L. (1965). *Survey Sampling*. New York: John Wiley & Sons.

Kish, L. (1987). *Statistical Design for Research*. New York: John Wiley & Sons.

Lohr, S. L. (2010). *Sampling: Design and Analysis*. 2nd ed. Boston: Brooks/Cole.

Madow, W. G. (1949). "On the Theory of Systematic Sampling, II." *Annals of Mathematical Statistics* 20:333–354.

Matsumoto, M., and Nishimura, T. (1998). "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-random Number Generator." *ACM Transactions on Modeling and Computer Simulation* 8:3–30.

McLeod, A. I., and Bellhouse, D. R. (1983). "A Convenient Algorithm for Drawing a Simple Random Sample." *Journal of the Royal Statistical Society, Series C* 32:182–183.

Murthy, M. N. (1957). "Ordered and Unordered Estimators in Sampling without Replacement." *Sankhyā* 18:379–390.

Murthy, M. N. (1967). *Sampling Theory and Methods*. Calcutta: Statistical Publishing Society.

Ohlsson, E. (1998). "Sequential Poisson Sampling." *Journal of Official Statistics* 14:149–162.

Sampford, M. R. (1967). "On Sampling without Replacement with Unequal Probabilities of Selection." *Biometrika* 54:499–513.

Särndal, C.-E., Swensson, B., and Wretman, J. (1992). *Model Assisted Survey Sampling*. New York: Springer-Verlag.

Vijayan, K. (1968). "An Exact π_{ps} Sampling Scheme: Generalization of a Method of Hanurav." *Journal of the Royal Statistical Society, Series B* 30:556–566.

Watts, D. L. (1991). "Correction: Computer Selection of Size-Biased Samples." *American Statistician* 45:172.

Wilburn, A. J. (1984). *Practical Statistical Sampling for Auditors*. New York: Marcel Dekker.

Williams, R. L., and Chromy, J. R. (1980). "SAS Sample Selection Macros." In *Proceedings of the Fifth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI80/Sugi-80-71%20Williams%20Chromy.pdf.

Wolter, K. M. (2007). *Introduction to Variance Estimation*. 2nd ed. New York: Springer.

Chapter 125
The TPSPLINE Procedure

Contents

Overview: TPSPLINE Procedure **10498**
- Penalized Least Squares Estimation 10499
- PROC TPSPLINE with Large Data Sets 10500

Getting Started: TPSPLINE Procedure **10500**

Syntax: TPSPLINE Procedure **10509**
- PROC TPSPLINE Statement 10510
- BY Statement 10514
- FREQ Statement 10515
- ID Statement 10515
- MODEL Statement 10515
- OUTPUT Statement 10518
- SCORE Statement 10519

Details: TPSPLINE Procedure **10520**
- Computational Formulas 10520
- ODS Table Names 10524
- ODS Graphics 10524

Examples: TPSPLINE Procedure **10525**
- Example 125.1: Partial Spline Model Fit 10526
- Example 125.2: Spline Model with Higher-Order Penalty 10528
- Example 125.3: Multiple Minima of the GCV Function 10532
- Example 125.4: Large Data Set Application 10538
- Example 125.5: Computing a Bootstrap Confidence Interval 10542

References **10549**

Overview: TPSPLINE Procedure

The TPSPLINE procedure uses the penalized least squares method to fit a nonparametric regression model. It computes thin-plate smoothing splines to approximate smooth multivariate functions observed with noise. The TPSPLINE procedure allows great flexibility in the possible form of the regression surface. In particular, PROC TPSPLINE makes no assumptions of a parametric form for the model. The generalized cross validation (GCV) function can be used to select the amount of smoothing.

The TPSPLINE procedure complements the methods provided by the standard SAS regression procedures such as the GLM, REG, and NLIN procedures. These procedures can handle most situations in which you specify the regression model and the model is known up to a fixed number of parameters. However, when you have no prior knowledge about the model, or when you know that the data cannot be represented by a model with a fixed number of parameters, you can use the TPSPLINE procedure to model the data.

The TPSPLINE procedure uses the penalized least squares method to fit the data with a flexible model in which the number of effective parameters can be as large as the number of unique design points. Hence, as the sample size increases, the model space also increases, enabling the thin-plate smoothing spline to fit more complicated situations.

The main features of the TPSPLINE procedure are as follows:

- provides penalized least squares estimates
- supports the use of multidimensional data
- supports multiple SCORE statements
- fits both semiparametric models and nonparametric models
- provides options for handling large data sets
- supports multiple dependent variables
- enables you to choose a particular model by specifying the model degrees of freedom or smoothing parameter
- produces graphs with ODS Graphics

Penalized Least Squares Estimation

Penalized least squares estimation provides a way to balance fitting the data closely and avoiding excessive roughness or rapid variation. A penalized least squares estimate is a surface that minimizes the penalized squared error over the class of all surfaces that satisfy sufficient regularity conditions.

Define $\mathbf{x}_i$ as a d-dimensional covariate vector from an $n \times d$ matrix $\mathbf{X}$, $\mathbf{z}_i$ as a p-dimensional covariate vector, and y_i as the observation associated with $(\mathbf{x}_i, \mathbf{z}_i)$. Assuming that the relation between $\mathbf{z}_i$ and y_i is linear but the relation between $\mathbf{x}_i$ and y_i is unknown, you can fit the data by using a semiparametric model as follows:

$$y_i = f(\mathbf{x}_i) + \mathbf{z}_i \boldsymbol{\beta} + \epsilon_i$$

where f is an unknown function that is assumed to be reasonably smooth, $\epsilon_i, i = 1, \ldots, n$, are independent, zero-mean random errors, and $\boldsymbol{\beta}$ is a p-dimensional unknown parameter vector.

This model consists of two parts. The $\mathbf{z}_i \boldsymbol{\beta}$ is the parametric part of the model, and the $\mathbf{z}_i$ are the regression variables. The $f(\mathbf{x}_i)$ is the nonparametric part of the model, and the $\mathbf{x}_i$ are the smoothing variables. The ordinary least squares method estimates $f(\mathbf{x}_i)$ and $\boldsymbol{\beta}$ by minimizing the quantity:

$$\frac{1}{n} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i) - \mathbf{z}_i \boldsymbol{\beta})^2$$

However, the functional space of $f(\mathbf{x})$ is so large that you can always find a function f that interpolates the data points. In order to obtain an estimate that fits the data well and has some degree of smoothness, you can use the penalized least squares method.

The penalized least squares function is defined as

$$S_\lambda(f) = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i) - \mathbf{z}_i \boldsymbol{\beta})^2 + \lambda J_2(f)$$

where $J_2(f)$ is the penalty on the roughness of f and is defined, in most cases, as the integral of the square of the second derivative of f.

The first term measures the goodness of fit and the second term measures the smoothness associated with f. The λ term is the smoothing parameter, which governs the trade-off between smoothness and goodness of fit. When λ is large, it more heavily penalizes rougher fits. Conversely, a small value of λ puts more emphasis on the goodness of fit.

The estimate f_λ is selected from a reproducing kernel Hilbert space, and it can be represented as a linear combination of a sequence of basis functions. Hence, the final estimates of f can be written as

$$\hat{f}_\lambda(\mathbf{x}_i) = \theta_0 + \sum_{j=1}^{d} \theta_j \mathbf{x}_{ij} + \sum_{j=1}^{p} \delta_j B_j(\mathbf{x}_j)$$

where B_j is the basis function, which depends on where the data $\mathbf{x}_i$ are located, and $\boldsymbol{\theta} = \{\theta_0, \ldots, \theta_d\}$ and $\boldsymbol{\delta} = \{\delta_1, \ldots, \delta_p\}$ are the coefficients that need to be estimated.

For a fixed λ, the coefficients $(\boldsymbol{\theta}, \boldsymbol{\delta}, \boldsymbol{\beta})$ can be estimated by solving an $n \times n$ system.

The smoothing parameter can be chosen by minimizing the generalized cross validation (GCV) function.

If you write

$$\hat{\mathbf{y}} = \mathbf{A}(\lambda)\mathbf{y}$$

then $\mathbf{A}(\lambda)$ is referred to as the *hat* or *smoothing* matrix, and the GCV function $\mathrm{GCV}(\lambda)$ is defined as

$$\mathrm{GCV}(\lambda) = \frac{(1/n)\|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y}\|^2}{[(1/n)\mathrm{tr}(\mathbf{I} - \mathbf{A}(\lambda))]^2}$$

PROC TPSPLINE with Large Data Sets

The calculation of the penalized least squares estimate is computationally intensive. The amount of memory and CPU time needed for the analysis depends on the number of unique design points, which corresponds to the number of unknown parameters to be estimated.

You can specify the D= option in the MODEL statement to reduce the number of unknown parameters. The option groups design points by the specified range (see the D= option on page 10516).

PROC TPSPLINE selects one design point from the group and treats all observations in the group as replicates of that design point. Calculation of the thin-plate smoothing spline estimates is based on the reprocessed data. The way to choose the design point from a group depends on the order of the data. Hence, different orders of input data might result in different estimates.

By combining several design points into one, this option reduces the number of unique design points, thereby approximating the original data. The value you specify for the D= option determines the width of the range used to group the data.

Getting Started: TPSPLINE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/tpsplgs.sas.)

The following example demonstrates how you can use the TPSPLINE procedure to fit a semiparametric model.

Suppose that y is a continuous variable and x1 and x2 are two explanatory variables of interest. To fit a bivariate thin-plate spline model, you can use a MODEL statement similar to that used in many regression procedures in the SAS System:

```
proc tpspline;
   model y = (x1 x2);
run;
```

The TPSPLINE procedure can fit semiparametric models; the parentheses in the preceding MODEL statement separate the smoothing variables from the regression variables. The following statements illustrate this syntax:

```
proc tpspline;
   model y = z1 (x1 x2);
run;
```

This model assumes a linear relation with z1 and an unknown functional relation with x1 and x2.

If you want to fit several responses by using the same explanatory variables, you can save computation time by using the multiple responses feature in the MODEL statement. For example, if y1 and y2 are two response variables, the following MODEL statement can be used to fit two models. Separate analyses are then performed for each response variable.

```
proc tpspline;
   model y1 y2 = (x1 x2);
run;
```

The following example illustrates the use of PROC TPSPLINE. The data are from Bates et al. (1987).

```
data Measure;
   input x1 x2 y @@;
   datalines;
-1.0 -1.0   15.54483570   -1.0 -1.0    15.76312613
 -.5 -1.0   18.67397826    -.5 -1.0    18.49722167
  .0 -1.0   19.66086310     .0 -1.0    19.80231311
  .5 -1.0   18.59838649     .5 -1.0    18.51904737
 1.0 -1.0   15.86842815    1.0 -1.0    16.03913832
-1.0  -.5   10.92383867   -1.0  -.5    11.14066546
 -.5  -.5   14.81392847    -.5  -.5    14.82830425
  .0  -.5   16.56449698     .0  -.5    16.44307297
  .5  -.5   14.90792284     .5  -.5    15.05653924
 1.0  -.5   10.91956264    1.0  -.5    10.94227538
-1.0   .0    9.61492010   -1.0   .0     9.64648093
 -.5   .0   14.03133439    -.5   .0    14.03122345
  .0   .0   15.77400253     .0   .0    16.00412514
  .5   .0   13.99627680     .5   .0    14.02826553
 1.0   .0    9.55700164    1.0   .0     9.58467047
-1.0   .5   11.20625177   -1.0   .5    11.08651907
 -.5   .5   14.83723493    -.5   .5    14.99369172
  .0   .5   16.55494349     .0   .5    16.51294369
  .5   .5   14.98448603     .5   .5    14.71816070
 1.0   .5   11.14575565    1.0   .5    11.17168689
-1.0  1.0   15.82595514   -1.0  1.0    15.96022497
 -.5  1.0   18.64014953    -.5  1.0    18.56095997
  .0  1.0   19.54375504     .0  1.0    19.80902641
  .5  1.0   18.56884576     .5  1.0    18.61010439
 1.0  1.0   15.86586951    1.0  1.0    15.90136745
;
```

The data set Measure contains three variables x1, x2, and y. Suppose that you want to fit a surface by using the variables x1 and x2 to model the response y. The variables x1 and x2 are spaced evenly on a $[-1 \times 1] \times [-1 \times 1]$ square, and the response y is generated by adding a random error to a function $f(x_1, x_2)$. The raw data are plotted in three-dimensional scatter plot by using the G3D procedure. In order to visualize those replicates, half of the data are shifted a little bit by adding a small value (0.001) to x1 values, as in the following statements:

```
data Measure1;
   set Measure;
run;

proc sort data=Measure1;
   by x2 x1;
run;

data Measure1;
   set Measure1;
   if mod(_N_, 2) = 0 then x1=x1+0.001;
run;

proc g3d data=Measure1;
   scatter x2*x1=y /size=.5
                    zmin=9 zmax=21
                    zticknum=4;
   title "Raw Data";
run;
```

Figure 125.1 displays the raw data.

Figure 125.1 Plot of Data Set MEASURE

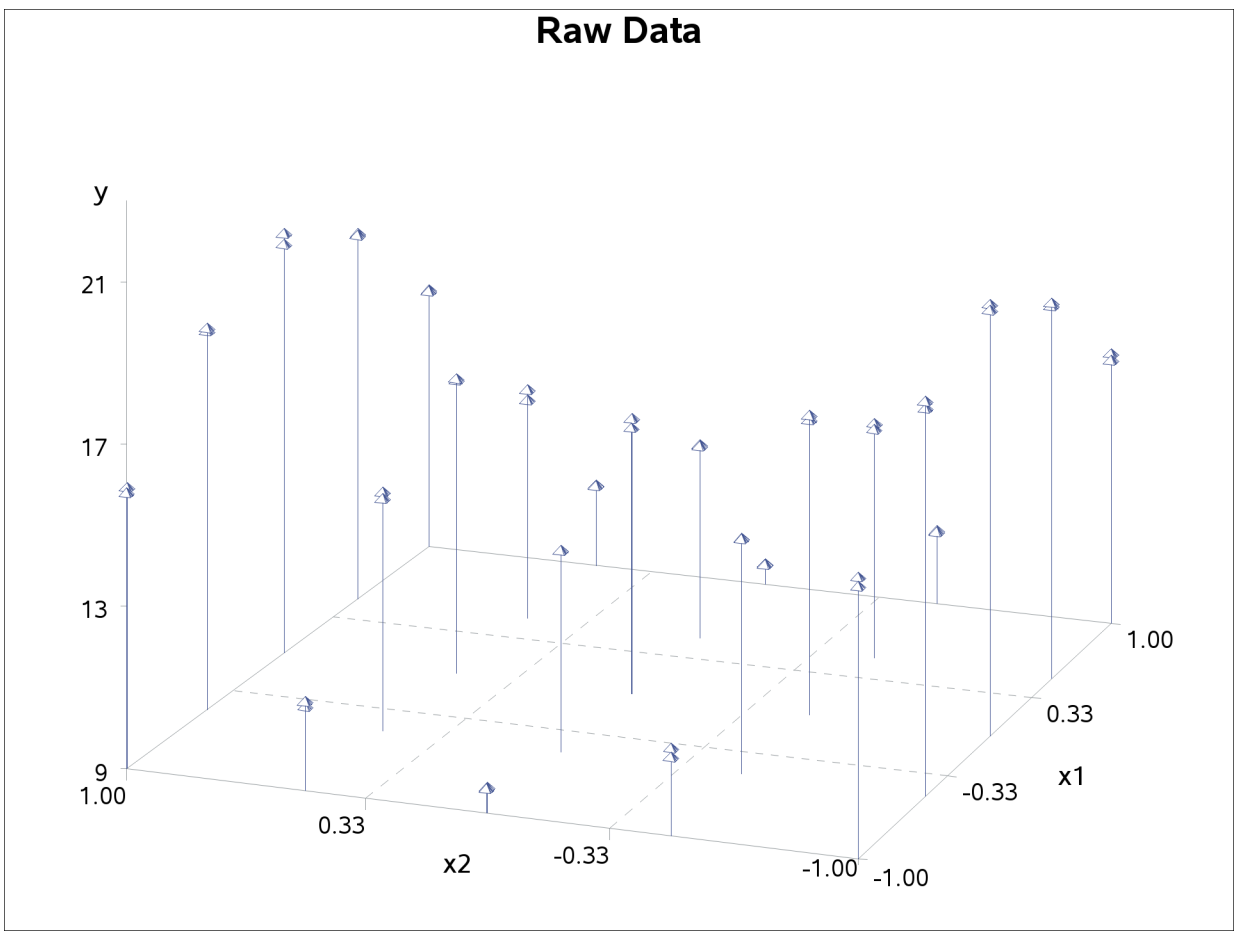

The following statements invoke the TPSPLINE procedure, to analyze the Measure data set as input. In the MODEL statement, the x1 and x2 variables are listed as smoothing variables. The LOGNLAMBDA= option specifies that PROC TPSPLINE examine a list of models with $\log_{10}(n\lambda)$ ranging from –4 to –2.5. The OUTPUT statement creates the data set estimate to contain the predicted values and the 95% upper and lower confidence limits from the best model selected by the GCV criterion.

```
ods graphics on;

proc tpspline data=Measure;
   model y=(x1 x2) /lognlambda=(-4 to -2.5 by 0.1);
   output out=estimate pred uclm lclm;
run;

proc print data=estimate;
run;
```

When ODS Graphics is enabled, PROC TPSPLINE produces several default plots. One of the default plots is the contour plot of the fitted surface, shown in Figure 125.2. The surface exhibits nonlinear patterns along the directions of both predictors.

Figure 125.2 Fitted Surface from PROC TPSPLINE

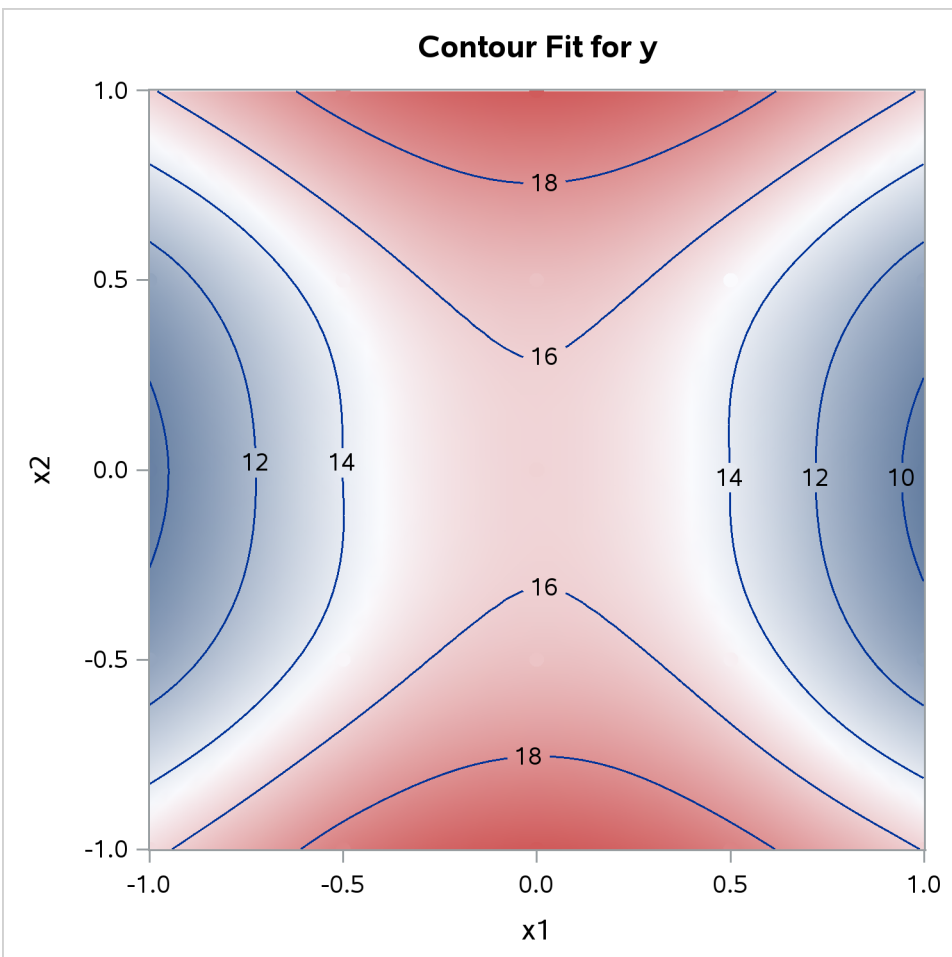

Figure 125.3 shows the "Criterion Plot" that provides a graphical display of the GCV selection process. Three sets of values are shown in the plot: the specified smoothing values and their GCV values, the examined smoothing values and their GCV values during the optimization process, and the best smoothing parameter and its GCV value. The final thin-plate smoothing spline estimate is based on $\log_{10}(n\lambda) = -3.4762$, which minimizes the GCV.

Figure 125.3 The GCV Criterion by $\log_{10}(n\lambda)$

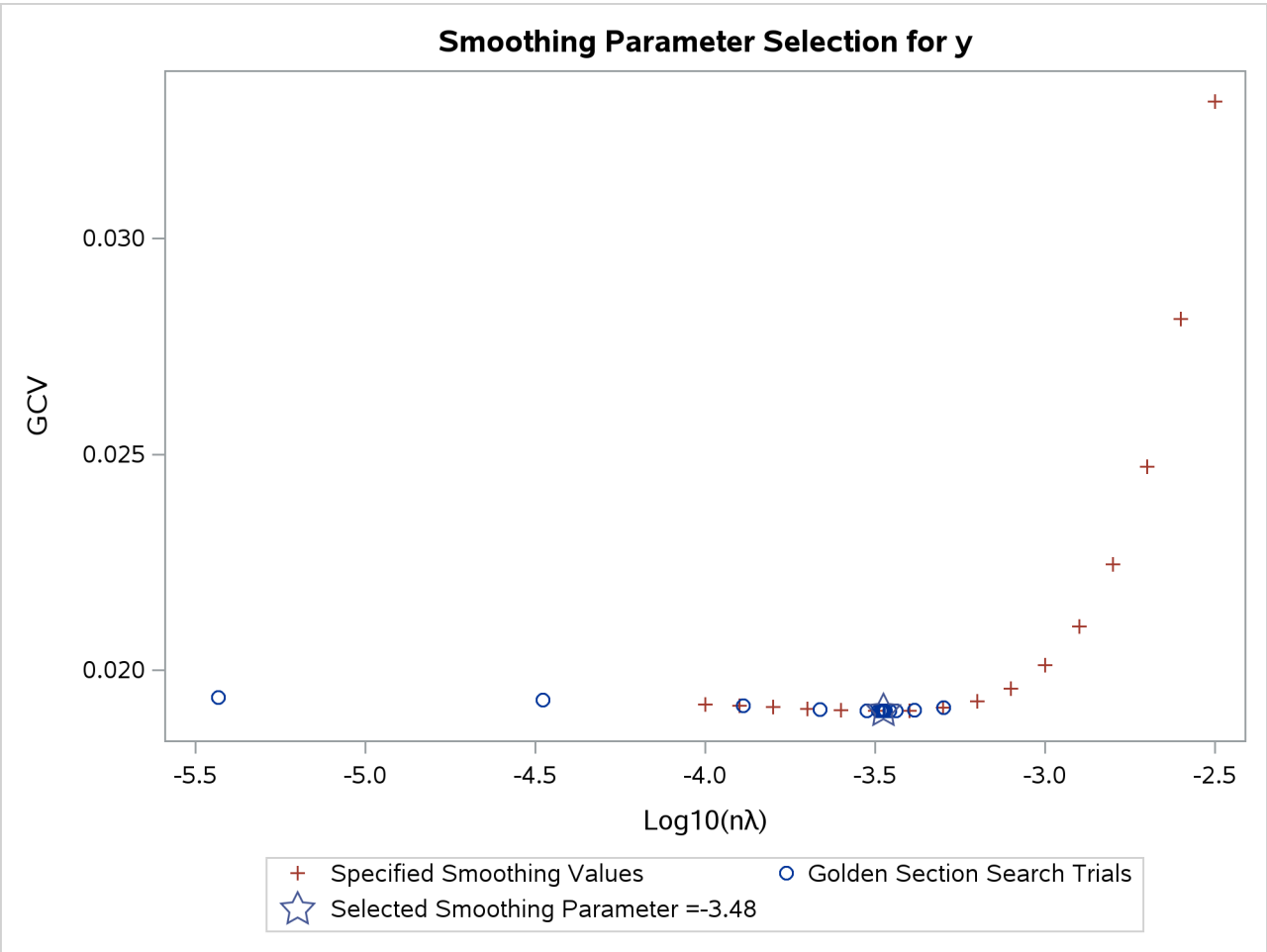

Figure 125.4 shows that the data set Measure contains 50 observations with 25 unique design points. The final model contains no parametric regression terms and two smoothing variables. The order of the derivative in the penalty is 2 by default, and the dimension of polynomial space is 3. See the section "Computational Formulas" on page 10520 for definitions.

Figure 125.4 also lists the GCV values along with the supplied values of $\log_{10}(n\lambda)$. The value that minimizes the GCV function is –3.5 among the given list of $\log_{10}(n\lambda)$.

The residual sum of squares from the fitted model is 0.246110, and the model degrees of freedom are 24.593203. The standard deviation, defined as $\mathrm{RSS}/(\mathrm{tr}(\mathbf{I} - \mathbf{A}))$, is 0.098421. The predictions and 95% confidence limits are displayed in Figure 125.5.

Figure 125.4 Fitted Model Summaries from PROC TPSPLINE

The TPSPLINE Procedure
Dependent Variable: y

Summary of Input Data Set	
Number of Non-Missing Observations	50
Number of Missing Observations	0
Unique Smoothing Design Points	25

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	2
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	3

GCV Function		
log10(n*Lambda)	GCV	
-4.000000	0.019215	
-3.900000	0.019183	
-3.800000	0.019148	
-3.700000	0.019113	
-3.600000	0.019082	
-3.500000	0.019064	*
-3.400000	0.019074	
-3.300000	0.019135	
-3.200000	0.019286	
-3.100000	0.019584	
-3.000000	0.020117	
-2.900000	0.021015	
-2.800000	0.022462	
-2.700000	0.024718	
-2.600000	0.028132	
-2.500000	0.033165	

Note: * indicates minimum GCV value.

Summary Statistics of Final Estimation	
log10(n*Lambda)	-3.4762
Smoothing Penalty	2558.1432
Residual SS	0.2461
Tr(I-A)	25.4068
Model DF	24.5932
Standard Deviation	0.0984
GCV	0.0191

Figure 125.5 Data Set ESTIMATE

Obs	x1	x2	y	P_y	LCLM_y	UCLM_y
1	-1.0	-1.0	15.5448	15.6474	15.5115	15.7832
2	-1.0	-1.0	15.7631	15.6474	15.5115	15.7832
3	-0.5	-1.0	18.6740	18.5783	18.4430	18.7136
4	-0.5	-1.0	18.4972	18.5783	18.4430	18.7136
5	0.0	-1.0	19.6609	19.7270	19.5917	19.8622
6	0.0	-1.0	19.8023	19.7270	19.5917	19.8622
7	0.5	-1.0	18.5984	18.5552	18.4199	18.6905
8	0.5	-1.0	18.5190	18.5552	18.4199	18.6905
9	1.0	-1.0	15.8684	15.9436	15.8077	16.0794
10	1.0	-1.0	16.0391	15.9436	15.8077	16.0794
11	-1.0	-0.5	10.9238	11.0467	10.9114	11.1820
12	-1.0	-0.5	11.1407	11.0467	10.9114	11.1820
13	-0.5	-0.5	14.8139	14.8246	14.6896	14.9597
14	-0.5	-0.5	14.8283	14.8246	14.6896	14.9597
15	0.0	-0.5	16.5645	16.5102	16.3752	16.6452
16	0.0	-0.5	16.4431	16.5102	16.3752	16.6452
17	0.5	-0.5	14.9079	14.9812	14.8461	15.1162
18	0.5	-0.5	15.0565	14.9812	14.8461	15.1162
19	1.0	-0.5	10.9196	10.9497	10.8144	11.0850
20	1.0	-0.5	10.9423	10.9497	10.8144	11.0850
21	-1.0	0.0	9.6149	9.6372	9.5019	9.7724
22	-1.0	0.0	9.6465	9.6372	9.5019	9.7724
23	-0.5	0.0	14.0313	14.0188	13.8838	14.1538
24	-0.5	0.0	14.0312	14.0188	13.8838	14.1538
25	0.0	0.0	15.7740	15.8822	15.7472	16.0171
26	0.0	0.0	16.0041	15.8822	15.7472	16.0171
27	0.5	0.0	13.9963	14.0006	13.8656	14.1356
28	0.5	0.0	14.0283	14.0006	13.8656	14.1356
29	1.0	0.0	9.5570	9.5769	9.4417	9.7122
30	1.0	0.0	9.5847	9.5769	9.4417	9.7122
31	-1.0	0.5	11.2063	11.1614	11.0261	11.2967
32	-1.0	0.5	11.0865	11.1614	11.0261	11.2967
33	-0.5	0.5	14.8372	14.9182	14.7831	15.0532
34	-0.5	0.5	14.9937	14.9182	14.7831	15.0532
35	0.0	0.5	16.5549	16.5386	16.4036	16.6736
36	0.0	0.5	16.5129	16.5386	16.4036	16.6736
37	0.5	0.5	14.9845	14.8549	14.7199	14.9900
38	0.5	0.5	14.7182	14.8549	14.7199	14.9900
39	1.0	0.5	11.1458	11.1727	11.0374	11.3080
40	1.0	0.5	11.1717	11.1727	11.0374	11.3080
41	-1.0	1.0	15.8260	15.8851	15.7493	16.0210
42	-1.0	1.0	15.9602	15.8851	15.7493	16.0210
43	-0.5	1.0	18.6401	18.5946	18.4593	18.7299
44	-0.5	1.0	18.5610	18.5946	18.4593	18.7299
45	0.0	1.0	19.5438	19.6729	19.5376	19.8081
46	0.0	1.0	19.8090	19.6729	19.5376	19.8081
47	0.5	1.0	18.5688	18.5832	18.4478	18.7185
48	0.5	1.0	18.6101	18.5832	18.4478	18.7185

Figure 125.5 *continued*

Obs	x1	x2	y	P_y	LCLM_y	UCLM_y
49	1.0	1.0	15.8659	15.8761	15.7402	16.0120
50	1.0	1.0	15.9014	15.8761	15.7402	16.0120

You can also use the TEMPLATE and SGRENDER procedures to create a perspective plot for visualizing the fitted surface. Because the data in the data set Measure are very sparse, the fitted surface is not smooth. To produce a smoother surface, the following statements generate the data set pred in order to obtain a finer grid. The LOGNLAMBDA0= option requests that PROC TPSPLINE fit a model with a fixed $\log_{10}(n\lambda)$ value of –3.4762. The SCORE statement evaluates the fitted surface at those new design points.

```
data pred;
   do x1=-1 to 1 by 0.1;
      do x2=-1 to 1 by 0.1;
         output;
      end;
   end;
run;

proc tpspline data=measure;
   model y=(x1 x2)/lognlambda0=-3.4762;
   score data=pred out=predy;
run;

proc template;
   define statgraph surface;
      dynamic _X _Y _Z _T;
      begingraph /designheight=360;
         entrytitle _T;
         layout overlay3d/rotate=120 cube=false  xaxisopts=(label="x1")
                         yaxisopts=(label="x2") zaxisopts=(label="P_y");
            surfaceplotparm x=_X y=_Y z=_Z;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=predy template=surface;
   dynamic _X='x1' _Y='x2' _Z='P_y'
           _T='Plot of Fitted Surface on a Fine Grid';
run;
```

The surface plot based on the finer grid is displayed in Figure 125.6. The plot indicates that a parametric model with quadratic terms of x1 and x2 provides a reasonable fit to the data.

Figure 125.6 Plot of TPSPLINE Fit

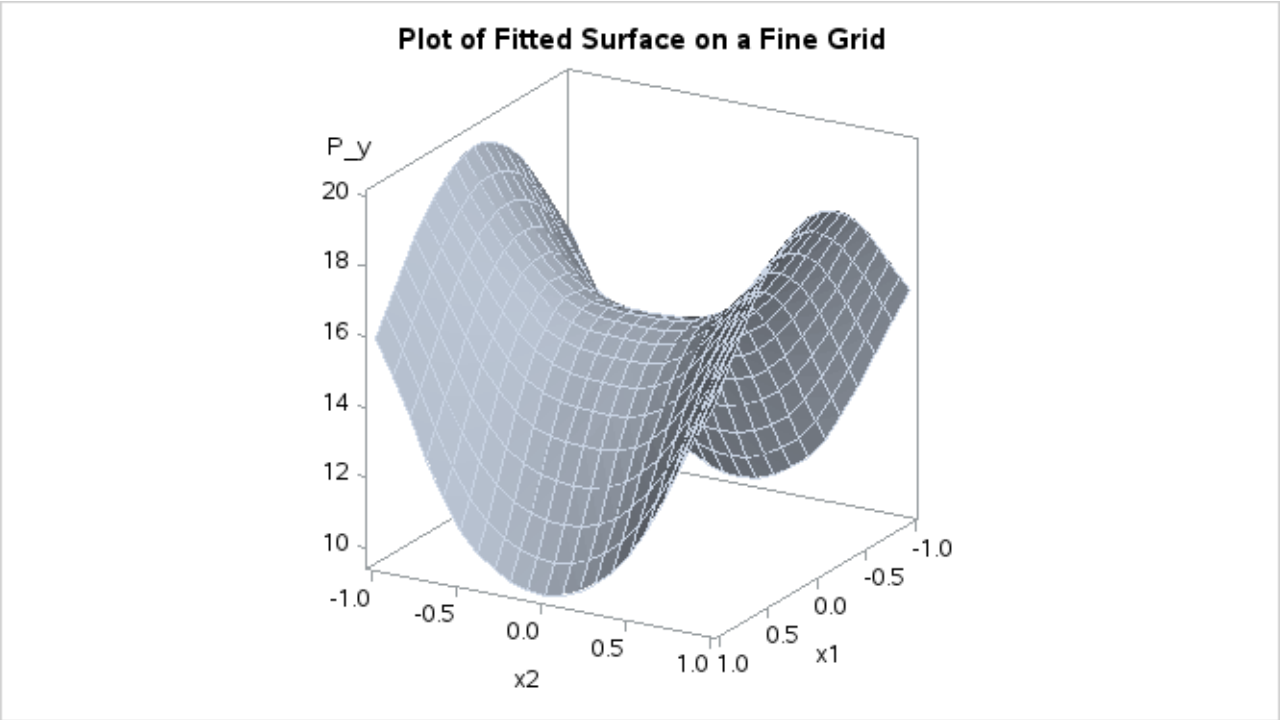

Figure 125.7 shows a panel of fit diagnostics for the selected model that indicate a reasonable fit:

- The predicted values closely approximate the observed values.
- The residuals are approximately normally distributed and do not show obvious systematic patterns.
- The RFPLOT shows that much variation in the response variable is addressed by the fit and only a little remains in the residuals.

Figure 125.7 Fit Diagnostics

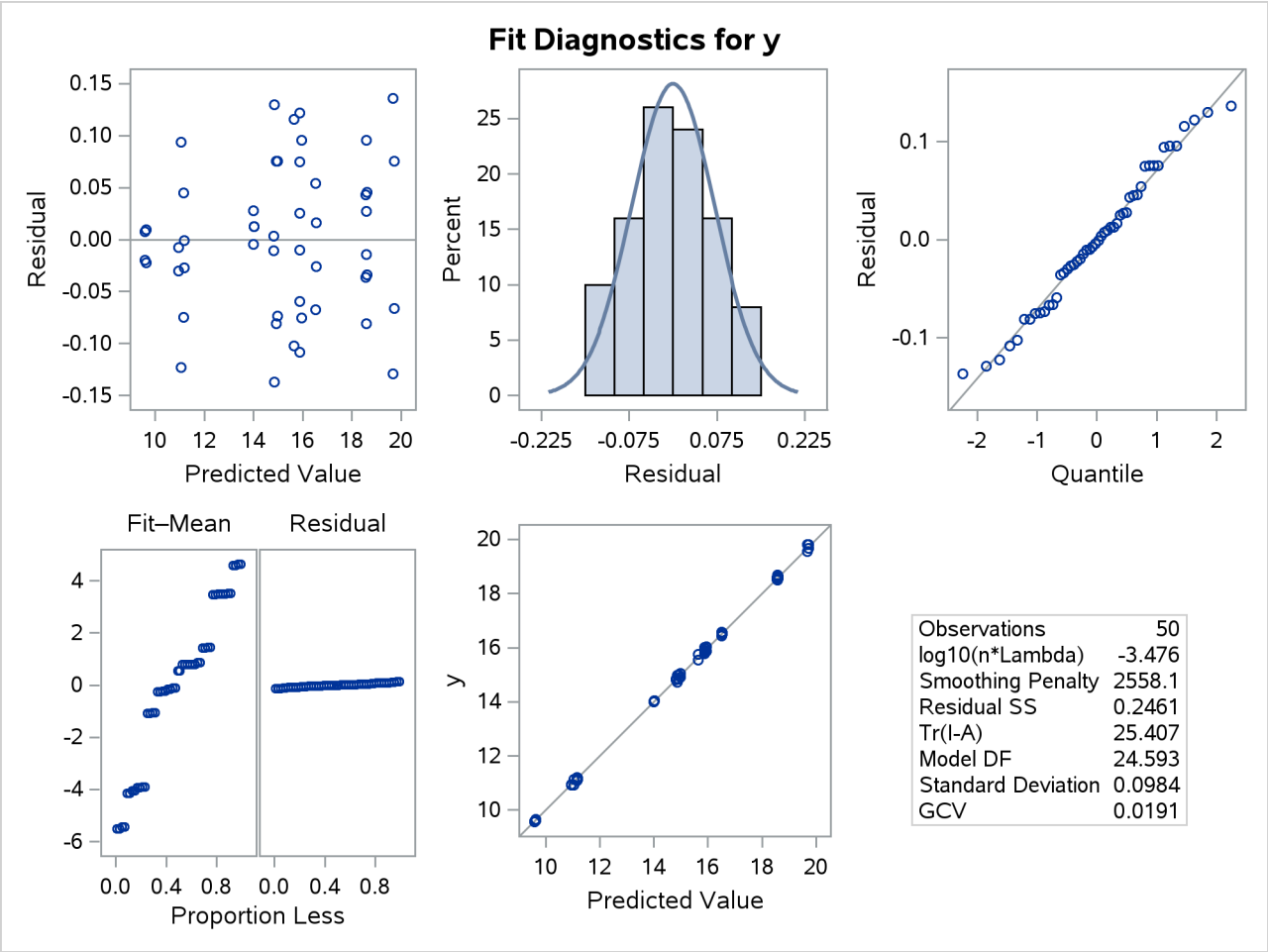

Syntax: TPSPLINE Procedure

The following statements are available in the TPSPLINE procedure:

PROC TPSPLINE < *options* > **;**
  MODEL *dependents* **=** < *variables* > **(***variables***)**< / *options* > **;**
  SCORE DATA=*SAS-data-set* **OUT=***SAS-data-set* < *keyword . . . keyword* > **;**
  OUTPUT < **OUT=***SAS-data-set* > *keyword . . . keyword* **;**
  BY *variables* **;**
  FREQ *variable* **;**
  ID *variables* **;**

The syntax in PROC TPSPLINE is similar to that of other regression procedures in the SAS System. The PROC TPSPLINE and MODEL statements are required. The SCORE statement can appear multiple times; all other statements appear no more than once.

The statements available for PROC TPSPLINE are described in alphabetical order after the description of the PROC TPSPLINE statement.

PROC TPSPLINE Statement

PROC TPSPLINE < *options* > **;**

The PROC TPSPLINE statement invokes the TPSPLINE procedure. Table 125.1 summarizes the *options* available in the TPSPLINE statement.

Table 125.1 PROC TPSPLINE Statement Options

Option	Description
DATA=	Specifies the SAS data set to be read
PLOTS	Controls the plots that are produced through ODS Graphics

You can specify the following *options*:

DATA=*SAS-data-set*
specifies the SAS data set to be read by PROC TPSPLINE. The default value is the most recently created data set.

PLOTS < (*global-plot-options*) > < = *plot-request*< (*options*) > >

PLOTS < (*global-plot-options*) > < = (*plot-request*< (*options*) > < . . . *plot-request*< (*options*) > >) >
controls the plots that are produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=none
plots=residuals(smooth)
plots(unpack)=diagnostics
plots(only)=(fit residualHistogram)
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc tpspline;
   model y = (x);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS= option, then PROC TPSPLINE produces a default set of plots. The following table lists the plots that are produced.

Table 125.2 Graphs Produced

Plot	Conditional on:
ContourByLambdaPanel	LAMBDA= or LOGNLAMBDA= option specified in the MODEL statement
Contour	Model with two predictors
CriterionPlot	Multiple values for the smoothing parameter
DiagnosticsPanel	Unconditional
ResidByLambdaPanel	LAMBDA= or LOGNLAMBDA= option specified in the MODEL statement
ResidPanel	Unconditional
FitByLambdaPanel	LAMBDA= or LOGNLAMBDA= option specified in the MODEL statement
FitPlot	Model with one predictor
ScorePlot	One or more SCORE statements and a model with one predictor

For models with multiple dependent variables, separate plots are produced for each dependent variable. For models in which multiple smoothing parameters are specified with the LAMBDA= or LOGNLAMBDA= option in the MODEL statement, the plots are produced for the selected model only.

Global Plot Options

The *global-plot-options* apply to all relevant plots generated by the TPSPLINE procedure, unless they are overridden by a *specific-plot-option*. The following *global-plot-options* are supported by the TPSPLINE procedure:

ONLY
suppresses the default plots. Only the plots specifically requested are produced.

UNPACK
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACK to get each plot individually. You can specify PLOTS(UNPACK) to unpack the default plots. You can also specify UNPACK as a suboption with the CONTOURFITPANEL, DIAGNOSTICS, FITPANEL, RESIDUALS and RESIDUALSBYSMOOTH options.

Plot Requests

You can specify the following specific *plot-requests* and controls for them:

ALL
produces all plots appropriate for the particular analysis. You can specify other options with ALL; for example, to request that all plots be produced and that only the residual plots be unpacked, specify PLOTS=(ALL RESIDUALS(UNPACK)).

CONTOURFIT < (OBS=*contour-options*) >
produces a contour plot of the fitted surface overlaid with a scatter plot of the data for models with two predictors. You can use the following *contour-options* to control how the observations are displayed:

GRADIENT
displays observations as circles colored by the observed response. The same color gradient is used to display the fitted surface and the observations. Observations where the predicted response is close to the observed response have similar colors—the greater the contrast between the color of an observation and the surface, the larger the residual is at that point. OBS=GRADIENT is the default if you do not specify any *contour-options*.

NONE
suppresses the observations.

OUTLINE
displays observations as circles with a border but with a completely transparent fill.

OUTLINEGRADIENT
is the same as OBS=GRADIENT except that a border is shown around each observation. This option is useful for identifying the location of observations where the residuals are small, because at these points the color of the observations and the color of the surface are indistinguishable.

CONTOURFITPANEL < (*options*) >
produces panels of contour plots overlaid with a scatter plot of the data for each smoothing parameter specified in the LAMBDA= or LOGNLAMBDA= option in the MODEL statement, for models with two predictors. If you do not specify the LAMBDA= or LOGNLAMBDA= option or if the model does not have two predictors, then this plot is not produced. Each panel contains at most six plots, and multiple panels are used when there are more than six smoothing parameters in the LAMBDA= or LOGNLAMBDA= option. The following *options* are available:

OBS=*contour-options*
specifies how the observations are displayed. See *contour-options* for the CONTOURFIT option for details.

UNPACK
suppresses paneling.

CRITERIONPLOT | CRITERION < (NOPATH) >
displays a scatter plot of the value of the GCV criterion versus the smoothing parameter value for all smoothing parameter values examined in the selection process. This plot is not produced when you specify one smoothing parameter with either the LAMBDA0= or LOGNLAMBDA0= option in the MODEL statement. When you supply a list of values for the smoothing parameter with the LAMBDA= or LOGNLAMBDA= option and PROC TPSPLINE obtains the optimal smoothing parameter by minimizing the GCV criterion, then the plot contains the supplied list of smoothing values and the optimal smoothing parameter in addition to the values examined during the optimization process. You can use the NOPATH suboption to disable the display of the optimization path in the plot in this case.

DIAGNOSTICSPANEL | DIAGNOSTICS < (UNPACK) >
produces a summary panel of fit diagnostics that consists of the following:

- residuals versus the predicted values
- a histogram of the residuals
- a normal quantile plot of the residuals

- a "Residual-Fit" (RF) plot that consists of side-by-side quantile plots of the centered fit and the residuals
- response values versus the predicted values

You can request the five plots in this panel as individual plots by specifying the UNPACK option. You can also request individual plots in the panel by name without having to unpack the panel. The fit diagnostics panel is produced by default whenever ODS Graphics is enabled.

FITPANEL < (*options*) >
produces panels of plots that show the fitted TPSPLINE curve overlaid on a scatter plot of the input data for each smoothing parameter specified in the LAMBDA= or LOGNLAMBDA= option in the MODEL statement. If you do not specify the LAMBDA= or LOGNLAMBDA= option or the model has more than one predictor, then this plot is not produced. Each panel contains at most six plots, and multiple panels are used when there are more than six smoothing parameters in the LAMBDA= or LOGNLAMBDA= option. The following *options* are available:

CLM
includes a confidence band at the significance level specified in the ALPHA= option in the MODEL statement in each plot in the panels.

UNPACK
suppresses paneling.

FITPLOT | FIT < (CLM) >
produces a scatter plot of the input data with the fitted TPSPLINE curve overlaid for models with a single predictor. If the CLM option is specified, then a confidence band at the significance level specified in the ALPHA= option in the MODEL statement is included in the plot.

NONE
suppresses all plots.

OBSERVEDBYPREDICTED
produces a scatter plot of the dependent variable values by the predicted values.

QQPLOT | QQ
produces a normal quantile plot of the residuals.

RESIDUALBYSMOOTH < (SMOOTH) >
produces, for each predictor, panels of plots that show the residuals of the TPSPLINE fit versus the predictor for each smoothing parameter specified in the LAMBDA= or LOGNLAMBDA= option in the MODEL statement. If you do not specify the LAMBDA= or LOGNLAMBDA= option, then this plot is not produced. Each panel contains at most six plots, and multiple panels are used when there are more than six smoothing parameters in the LAMBDA= or LOGNLAMBDA= option in the MODEL statement. The SMOOTH option displays a nonparametric fit line in each plot in the panel. The type of nonparametric fit and the options used are controlled by the underlying template for this plot. In the standard template that is provided, the nonparametric smooth is specified to be a loess fit that corresponds to the default options of PROC LOESS, except that the PRESEARCH suboption in the SELECT statement is always used. It is important to note that the loess fit that is shown in each of the residual plots is computed independently of the smoothing spline fit that is used to obtain the residuals.

RESIDUALBYPREDICTED
produces a scatter plot of the residuals by the predicted values.

RESIDUALHISTOGRAM
produces a histogram of the residuals.

RESIDUALPANEL | RESIDUALS <(*options*)>
produces panels of the residuals versus the predictors in the model. Each panel contains at most six plots, and multiple panels are used when there are more than six predictors in the model.

The following *options* are available:

SMOOTH
displays a nonparametric fit line in each plot in the panel. The type of nonparametric fit and the options used are controlled by the underlying template for this plot. In the standard template that is provided, the nonparametric smooth is specified to be a loess fit that corresponds to the default options of PROC LOESS, except that the PRESEARCH suboption in the SELECT statement is always used. It is important to note that the loess fit that is shown in each of the residual plots is computed independently of the smoothing spline fit that is used to obtain the residuals.

UNPACK
suppresses paneling.

RFPLOT | RF
produces a "Residual-Fit" (RF) plot that consists of side-by-side quantile plots of the centered fit and the residuals. This plot "shows how much variation in the data is explained by the fit and how much remains in the residuals" (Cleveland 1993).

SCOREPLOT | SCORE
produces a scatter plot of the scored values at the score points for each SCORE statement. SCORE plots are not produced for models with more than one predictor.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC TPSPLINE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the TPSPLINE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.

- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If one variable in your input data set represents the frequency of occurrence for other values in the observation, specify the variable's name in a FREQ statement. PROC TPSPLINE treats the data as if each observation appears *n* times, where *n* is the value of the FREQ variable for the observation. If the value of the FREQ variable is less than one, the observation is not used in the analysis. Only the integer portion of the value is used.

ID Statement

ID *variables* **;**

The ID statement is optional, and more than one ID statement can be used. If variables are specified in the ID statement, their values are displayed in tooltips to identify observations in the plots produced by PROC TPSPLINE.

MODEL Statement

MODEL *dependent-variables* **=** < *regression-variables* > **(***smoothing-variables***)**< / *options* > **;**

The MODEL statement specifies the dependent variables, the independent regression variables, which are listed with no parentheses, and the independent smoothing variables, which are listed inside parentheses.

The regression variables are optional. At least one smoothing variable is required, and it must be listed after the regression variables. No variables can be listed in both the regression variable list and the smoothing variable list.

If you specify more than one dependent variable, PROC TPSPLINE calculates a thin-plate smoothing spline estimate for each dependent variable by using the regression variables and smoothing variables specified on the right side.

If you specify regression variables, PROC TPSPLINE fits a semiparametric model by using the regression variables as the linear part of the model.

Table 125.3 summarizes the *options* available in the MODEL statement.

Table 125.3 MODEL Statement Options

Option	Description
ALPHA=	Specifies the significance level
DF=	Specifies the degrees of freedom
DISTANCE=	Defines a range in which points are treated as replicates
LAMBDA0=	Specifies the smoothing parameter
LAMBDA=	Specifies a set of values for the λ parameter
LOGNLAMBDA0=	Specifies the smoothing parameter on the $\log_{10}(n\lambda)$ scale
LOGNLAMBDA=	Specifies a set of values for the λ parameter on the $\log_{10}(n\lambda)$ scale
M=	Specifies the order of the derivative
RANGE=	Specifies the range for smoothing values to be evaluated

You can specify the following *options* in the MODEL statement:

ALPHA=*number*

specifies the significance level α of the confidence limits on the final thin-plate smoothing spline estimate when you request confidence limits to be included in the output data set. Specify *number* as a value between 0 and 1. The default value is 0.05. See the section "OUTPUT Statement" on page 10518 for more information about the OUTPUT statement.

DF=*df*

specifies the degrees of freedom of the thin-plate smoothing spline estimate, defined as

$$df = \mathrm{tr}(\mathbf{A}(\lambda))$$

where $\mathbf{A}(\lambda)$ is the *hat* matrix. Specify *df* as a value between zero and the number of unique design points n_q. Smaller *df* values cause more penalty on the roughness and thus smoother fits.

DISTANCE=*number*

D=*number*

defines a range such that if the L_∞ distance between two data points $(\mathbf{x}_i, \mathbf{z}_i)$ and $(\mathbf{x}_j, \mathbf{z}_j)$ satisfies

$$\|\mathbf{x}_i - \mathbf{x}_j\|_\infty \leq D/2$$

then these data points are treated as replicates, where $\mathbf{x}_i$ are the smoothing variables and $\mathbf{z}_i$ are the regression variables.

You can use the DISTANCE= option to reduce the number of unique design points by treating nearby data as replicates. This can be useful when you have a large data set. Larger DISTANCE= option values cause fewer n_q points. The default value is 0.

PROC TPSPLINE uses the DISTANCE= value to group points as follows: The data are first sorted by the smoothing variables in the order in which they appear in the MODEL statement. The first point in the sorted data becomes the first unique point. Subsequent points have their values set equal to that point until the first point where the maximum distance in one dimension is larger than $D/2$. This point becomes the next unique point, and so on. Because of this sequential processing, the set of unique points differs depending on the order of the smoothing variables in the MODEL statement.

For example, with a model that has two smoothing variables (x1, x2), the data are first sorted by x1 and x2 (in that order), and then uniqueness is assessed sequentially. The first point in the sorted data $\mathbf{x}_1 = (\mathsf{x1}_1, \mathsf{x2}_1)$ becomes the first unique point, $\mathbf{u}_1 = (\mathsf{u1}_1, \mathsf{u2}_1)$. Subsequent points $\mathbf{x}_i = (\mathsf{x1}_i, \mathsf{x2}_i)$ are set equal to $\mathbf{u}_1$ until the algorithm comes to a point with $\max(|\mathsf{x1}_i - \mathsf{u1}_1|, |\mathsf{x2}_i - \mathsf{u2}_1|) > D/2$. This point becomes the second unique point $\mathbf{u}_2$, and data sorting proceeds from there.

LAMBDA0=*number*

specifies the smoothing parameter, λ_0, to be used in the thin-plate smoothing spline estimate. By default, PROC TPSPLINE uses the λ parameter that minimizes the GCV function for the final fit. The LAMBDA0= value must be positive. Larger λ_0 values cause smoother fits.

LAMBDA=*list-of-values*

specifies a set of values for the λ parameter. PROC TPSPLINE returns a GCV value for each λ point that you specify. You can use the LAMBDA= option to study the GCV function curve for a set of values for λ. All values listed in the LAMBDA= option must be positive.

LOGNLAMBDA0=*number*

LOGNL0=*number*

specifies the smoothing parameter λ_0 on the $\log_{10}(n\lambda)$ scale. If you specify both the LOGNL0= and LAMBDA0= options, only the value provided by the LOGNL0= option is used. Larger $\log_{10}(n\lambda_0)$ values cause smoother fits. By default, PROC TPSPLINE uses the λ parameter that minimizes the GCV function for the estimate.

LOGNLAMBDA=*list-of-values*

LOGNL=*list-of-values*

specifies a set of values for the λ parameter on the $\log_{10}(n\lambda)$ scale. PROC TPSPLINE returns a GCV value for each λ point that you specify. You can use the LOGNLAMBDA= option to study the GCV function curve for a set of λ values. If you specify both the LOGNL= and LAMBDA= options, only the list of values provided by the LOGNL= option is used.

In some cases, the LOGNL= option might be preferred over the LAMBDA= option. Because the LAMBDA= value must be positive, a small change in that value can result in a major change in the GCV value. If you instead specify λ on the $\log_{10}(n\lambda)$ scale, the allowable range is enlarged to include negative values. Thus, the GCV function is less sensitive to changes in LOGNLAMBDA.

The DF= option, LAMBDA0= option, and LOGNLAMBDA0= option all specify exact smoothness of a nonparametric fit. If you want to fit a model with specified smoothness, the DF= option is preferable to the other two options because $(0, n_q)$, the range of df, is much smaller in length than $(0, \infty)$ of λ and $(-\infty, \infty)$ of $\log_{10}(n\lambda)$.

M=*number*

specifies the order of the derivative in the penalty term. The *number* must be a positive integer. The default value is $\max(2, \mathrm{int}(d/2) + 1)$, where d is the number of smoothing variables.

RANGE=(*lower***,** *upper***)**

specifies that on the $\log_{10}(n\lambda)$ scale only smoothing values greater than or equal to *lower* and less than or equal to *upper* be evaluated to minimize the GCV function.

OUTPUT Statement

OUTPUT OUT=*SAS-data-set* < *keyword* . . . *keyword* > **;**

The OUTPUT statement creates a new SAS data set that contains diagnostic measures calculated after fitting the model.

All the variables in the original data set are included in the new data set, along with variables created by specifying *keywords* in the OUTPUT statement. These new variables contain the values of a variety of statistics and diagnostic measures that are calculated for each observation in the data set. If no *keyword* is present, the data set contains only the original data set and predicted values.

Details about the specifications in the OUTPUT statement are as follows.

OUT=*SAS-data-set*
specifies the name of the new data set to contain the diagnostic measures. This specification is required.

keyword
specifies the statistics to include in the output data set. The names of the new variables that contain the statistics are formed by using a prefix of one or more characters to identify the statistic, followed by an underscore (_), followed by the dependent variable name.

For example, suppose that you have two dependent variables—say, y1 and y2—and you specify the keywords PRED, ADIAG, and UCLM. The output SAS data set will contain the following variables:

- P_y1 and P_y2
- ADIAG_y1 and ADIAG_y2
- UCLM_y1 and UCLM_y2

The *keywords* and the statistics they represent are as follows:

RESID | R residual values, calculated as fitted values subtracted from the observed response values: $y - \hat{y}$. The default prefix is R_.

PRED predicted values. The default prefix is P_.

STD standard error of the mean predicted value. The default prefix is STD_.

UCLM upper limit of the Bayesian confidence interval for the expected value of the dependent variables. By default, PROC TPSPLINE computes 95% confidence limits. The default prefix is UCLM_.

LCLM lower limit of the Bayesian confidence interval for the expected value of the dependent variables. By default, PROC TPSPLINE computes 95% confidence limits. The default prefix is LCLM_.

ADIAG diagonal element of the hat matrix associated with the observation. The default prefix is ADIAG_.

COEF coefficients arranged in the order of $(\theta_0, \theta_1, \ldots, \theta_d, \delta_1, \ldots, \delta_{n_q})$, where n_q is the number of unique data points. This option can be used only when there is only one dependent variable in the model. The default prefix is COEF_.

SCORE Statement

SCORE DATA=*SAS-data-set* **OUT=***SAS-data-set* < *keyword . . . keyword* > **;**

The SCORE statement calculates predicted statistics for a new data set. If you have multiple data sets to predict, you can specify multiple SCORE statements. You must use a SCORE statement for each data set.

You can request diagnostic measures that are calculated for each observation in the SCORE data set. The new data set contains all the variables in the SCORE data set in addition to the requested variables. If no *keyword* is present, the data set contains only the predicted values.

The following *keywords* must be specified in the SCORE statement:

DATA=*SAS-data-set*
specifies the input SAS data set that contains the smoothing variables $\mathbf{x}$ and regression variables $\mathbf{z}$. The predicted response ($\hat{y}$) value is computed for each $(\mathbf{x}, \mathbf{z})$ pair. The data set must include all independent variables specified in the MODEL statement.

OUT=*SAS-data-set*
specifies the name of the SAS data set to contain the predictions.

keyword
specifies the statistics to include in the output data set for the current SCORE statement. The names of the new variables that contain the statistics are formed by using a prefix of one or more characters to identify the statistic, followed by an underscore (_), followed by the dependent variable name. The *keywords* and the statistics they represent are as follows:

PRED	predicted values
STD	standard error of the mean predicted value
UCLM	upper limit of the Bayesian confidence interval for the expected value of the dependent variables. By default, PROC TPSPLINE computes 95% confidence limits.
LCLM	lower limit of the Bayesian confidence interval for the expected value of the dependent variables. By default, PROC TPSPLINE computes 95% confidence limits.

Details: TPSPLINE Procedure

Computational Formulas

The theoretical foundations for the thin-plate smoothing spline are described in Duchon (1976, 1977) and Meinguet (1979). Further results and applications are given in: Wahba and Wendelberger (1980); Hutchinson and Bischof (1983); Seaman and Hutchinson (1985).

Suppose that $\mathcal{H}_m$ is a space of functions whose partial derivatives of total order m are in $L_2(E^d)$, where E^d is the domain of $\mathbf{x}$.

Now, consider the data model

$$y_i = f(\mathbf{x}_i) + \epsilon_i, i = 1, \ldots, n$$

where $f \in \mathcal{H}_m$.

Using the notation from the section "Penalized Least Squares Estimation" on page 10499, for a fixed λ, estimate f by minimizing the penalized least squares function

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i) - \mathbf{z}_i\boldsymbol{\beta})^2 + \lambda J_m(f)$$

$\lambda J_m(f)$ is the penalty term to enforce smoothness on f. There are several ways to define $J_m(f)$. For the thin-plate smoothing spline, with $\mathbf{x} = (x_1, \ldots, x_d)$ of dimension d, define $J_m(f)$ as

$$J_m(f) = \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\sum\frac{m!}{\alpha_1!\cdots\alpha_d!}\left(\frac{\partial^m f}{\partial {x_1}^{\alpha_1}\cdots\partial {x_d}^{\alpha_d}}\right)^2 \mathrm{d}x_1\cdots\mathrm{d}x_d$$

where $\sum_i \alpha_i = m$. Under this definition, $J_m(f)$ gives zero penalty to some functions. The space that is spanned by the set of polynomials that contribute zero penalty is called the polynomial space. The dimension of the polynomial space M is a function of dimension d and order m of the smoothing penalty, $M = \binom{m+d-1}{d}$.

Given the condition that $2m > d$, the function that minimizes the penalized least squares criterion has the form

$$\hat{f}(\mathbf{x}) = \sum_{j=1}^{M}\theta_j\phi_j(\mathbf{x}) + \sum_{i=1}^{n}\delta_i\eta_{md}(\|\mathbf{x} - \mathbf{x}_i\|)$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\delta}$ are vectors of coefficients to be estimated. The M functions ϕ_j are linearly independent polynomials that span the space of functions for which $J_m(f)$ is zero. The basis functions η_{md} are defined as

$$\eta_{md}(r) = \begin{cases} \frac{(-1)^{m+1+d/2}}{2^{2m-1}\pi^{d/2}(m-1)!(m-d/2)!}r^{2m-d}\log(r) & \text{if } di \text{ is even} \\ \frac{\Gamma(d/2-m)}{2^{2m}\pi^{d/2}(m-1)!}r^{2m-d} & \text{if } d \text{ is odd} \end{cases}$$

When $d = 2$ and $m = 2$, then $M = \binom{3}{2} = 3$, $\phi_1(\mathbf{x}) = 1$, $\phi_2(\mathbf{x}) = x_1$, and $\phi_3(\mathbf{x}) = x_2$. $J_m(f)$ is as follows:

$$J_2(f) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\left(\frac{\partial^2 f}{\partial {x_1}^2}\right)^2 + 2\left(\frac{\partial^2 f}{\partial x_1\partial x_2}\right)^2 + \left(\frac{\partial^2 f}{\partial {x_2}^2}\right)^2\right)\mathrm{d}x_1\mathrm{d}x_2$$

For the sake of simplicity, the formulas and equations that follow assume $m = 2$. See Wahba (1990) and Bates et al. (1987) for more details.

Duchon (1976) showed that f_λ can be represented as

$$f_\lambda(\mathbf{x}_i) = \theta_0 + \sum_{j=1}^{d} \theta_j \mathbf{x}_{i\,j} + \sum_{j=1}^{n} \delta_j E_2(\mathbf{x}_i - \mathbf{x}_j)$$

where $E_2(\mathrm{s}) = \frac{1}{2^3\pi}\|\mathrm{s}\|^2 \log(\|\mathrm{s}\|)$ for $d = 2$. For derivations of $E_2(\mathrm{s})$ for other values of d, see Villalobos and Wahba (1987).

If you define $\mathbf{K}$ with elements $\mathbf{K}_{ij} = E_2(\mathbf{x}_i - \mathbf{x}_j)$ and $\mathbf{T}$ with elements $\mathbf{T}_{ij} = (\mathbf{X}_{ij})$, the goal is to find vectors of coefficients $\boldsymbol{\beta}, \boldsymbol{\theta}$, and $\boldsymbol{\delta}$ that minimize

$$S_\lambda(\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\delta}) = \frac{1}{n}\|\mathbf{y} - \mathbf{T}\boldsymbol{\theta} - \mathbf{K}\boldsymbol{\delta} - \mathbf{Z}\boldsymbol{\beta}\|^2 + \lambda\boldsymbol{\delta}'\mathbf{K}\boldsymbol{\delta}$$

A unique solution is guaranteed if the matrix $\mathbf{T}$ is of full rank and $\boldsymbol{\delta}'\mathbf{K}\boldsymbol{\delta} \geq 0$.

If $\boldsymbol{\alpha} = \begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{\beta} \end{pmatrix}$ and $\mathbf{X} = (\mathbf{T}\ \mathbf{Z})$, the expression for S_λ becomes

$$\frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\alpha} - \mathbf{K}\boldsymbol{\delta}\|^2 + \lambda\boldsymbol{\delta}'\mathbf{K}\boldsymbol{\delta}$$

The coefficients α and δ can be obtained by solving

$$\begin{aligned} (\mathbf{K} + n\lambda\mathbf{I}_n)\boldsymbol{\delta} + \mathbf{X}\boldsymbol{\alpha} &= \mathbf{y} \\ \mathbf{X}'\boldsymbol{\delta} &= \mathbf{0} \end{aligned}$$

To compute $\boldsymbol{\alpha}$ and $\boldsymbol{\delta}$, let the QR decomposition of $\mathbf{X}$ be

$$\mathbf{X} = (\mathbf{Q}_1\ \mathbf{Q}_2)\begin{pmatrix} \mathbf{R} \\ \mathbf{0} \end{pmatrix}$$

where $(\mathbf{Q}_1\ \mathbf{Q}_2)$ is an orthogonal matrix and $\mathbf{R}$ is an upper triangular, with $\mathbf{X}'\mathbf{Q}_2 = 0$ (Dongarra et al. 1979).

Since $\mathbf{X}'\boldsymbol{\delta} = 0$, $\boldsymbol{\delta}$ must be in the column space of $\mathbf{Q}_2$. Therefore, $\boldsymbol{\delta}$ can be expressed as $\delta = \mathbf{Q}_2\boldsymbol{\gamma}$ for a vector $\boldsymbol{\gamma}$. Substituting $\boldsymbol{\delta} = \mathbf{Q}_2\boldsymbol{\gamma}$ into the preceding equation and multiplying through by $\mathbf{Q}_2'$ gives

$$\mathbf{Q}_2'(\mathbf{K} + n\lambda\mathbf{I})\mathbf{Q}_2\boldsymbol{\gamma} = \mathbf{Q}_2'\mathbf{y}$$

or

$$\boldsymbol{\delta} = \mathbf{Q}_2\boldsymbol{\gamma} = \mathbf{Q}_2[\mathbf{Q}_2'(\mathbf{K} + n\lambda\mathbf{I})\mathbf{Q}_2]^{-1}\mathbf{Q}_2'\mathbf{y}$$

The coefficient $\boldsymbol{\alpha}$ can be obtained by solving

$$\mathbf{R}\boldsymbol{\alpha} = \mathbf{Q}_1'[\mathbf{y} - (\mathbf{K} + n\lambda\mathbf{I})\boldsymbol{\delta}]$$

The influence matrix $\mathbf{A}(\lambda)$ is defined as

$$\hat{\mathbf{y}} = \mathbf{A}(\lambda)\mathbf{y}$$

and has the form

$$\mathbf{A}(\lambda) = \mathbf{I} - n\lambda\mathbf{Q}_2[\mathbf{Q}_2'(\mathbf{K} + n\lambda\mathbf{I})\mathbf{Q}_2]^{-1}\mathbf{Q}_2'$$

Similar to the regression case, if you consider the trace of $\mathbf{A}(\lambda)$ as the degrees of freedom for the model and the trace of $(\mathbf{I} - \mathbf{A}(\lambda))$ as the degrees of freedom for the error, the estimate σ^2 can be represented as

$$\hat{\sigma}^2 = \frac{\mathrm{RSS}(\lambda)}{\mathrm{tr}(\mathbf{I} - \mathbf{A}(\lambda))}$$

where $\mathrm{RSS}(\lambda)$ is the residual sum of squares. Theoretical properties of these estimates have not yet been published. However, good numerical results in simulation studies have been described by several authors. For more information, see O'Sullivan and Wong (1987); Nychka (1986b, a, 1988); Hall and Titterington (1987).

Confidence Intervals

Viewing the spline model as a Bayesian model, Wahba (1983) proposed Bayesian confidence intervals for smoothing spline estimates as

$$\hat{f}_\lambda(\mathbf{x}_i) \pm z_{\alpha/2}\sqrt{\hat{\sigma}^2 a_{ii}(\lambda)}$$

where $a_{ii}(\lambda)$ is the ith diagonal element of the $\mathbf{A}(\lambda)$ matrix and $z_{\alpha/2}$ is the $1 - \alpha/2$ quantile of the standard normal distribution. The confidence intervals are interpreted as intervals "across the function" as opposed to pointwise intervals.

For SCORE data sets, the hat matrix $\mathbf{A}(\lambda)$ is not available. To compute the Bayesian confidence interval for a new point $\mathbf{x}_{\mathrm{new}}$, let

$$\mathbf{S} = \mathbf{X},\ \mathbf{M} = \mathbf{K} + n\lambda\mathbf{I}$$

and let $\boldsymbol{\xi}$ be an $n \times 1$ vector with ith entry

$$\eta_{md}(\|\mathbf{x}_{\mathrm{new}} - \mathbf{x}_i\|)$$

When $d = 2$ and $m = 2$, ξ_i is computed with

$$E_2(\mathbf{x}_i - \mathbf{x}_{\mathrm{new}}) = \frac{1}{2^3\pi}\|\mathbf{x}_i - \mathbf{x}_{\mathrm{new}}\|^2 \log(\|\mathbf{x}_i - \mathbf{x}_{\mathrm{new}}\|)$$

$\boldsymbol{\phi}$ is a vector of evaluations of $\mathbf{x}_{\mathrm{new}}$ by the polynomials that span the functional space where $J_m(f)$ is zero. The details for $\mathbf{X}$, $\mathbf{K}$, and E_2 are discussed in the previous section. Wahba (1983) showed that the Bayesian posterior variance of $\mathbf{x}_{\mathrm{new}}$ satisfies

$$n\lambda\mathrm{Var}(\mathbf{x}_{\mathrm{new}}) = \boldsymbol{\phi}'(\mathbf{S}'\mathbf{M}^{-1}\mathbf{S})^{-1}\boldsymbol{\phi} - 2\boldsymbol{\phi}'\mathbf{d}_\xi - \boldsymbol{\xi}'\mathbf{c}_\xi$$

where

$$\begin{aligned}\mathbf{c}_\xi &= (\mathbf{M}^{-1} - \mathbf{M}^{-1}\mathbf{S}(\mathbf{S}'\mathbf{M}^{-1}\mathbf{S})^{-1}\mathbf{S}'\mathbf{M}^{-1})\boldsymbol{\xi} \\ \mathbf{d}_\xi &= (\mathbf{S}'\mathbf{M}^{-1}\mathbf{S})^{-1}\mathbf{S}'\mathbf{M}^{-1}\boldsymbol{\xi}\end{aligned}$$

Suppose that you fit a spline estimate that consists of a true function f and a random error term ϵ_i to experimental data. In repeated experiments, it is likely that about $100(1 - \alpha)\%$ of the confidence intervals cover the corresponding true values, although some values are covered every time and other values are not covered by the confidence intervals most of the time. This effect is more pronounced when the true surface or surface has small regions of particularly rapid change.

Smoothing Parameter

The quantity λ is called the smoothing parameter, which controls the balance between the goodness of fit and the smoothness of the final estimate.

A large λ heavily penalizes the mth derivative of the function, thus forcing $f^{(m)}$ close to 0. A small λ places less of a penalty on rapid change in $f^{(m)}(\mathbf{x})$, resulting in an estimate that tends to interpolate the data points.

The smoothing parameter greatly affects the analysis, and it should be selected with care. One method is to perform several analyses with different values for λ and compare the resulting final estimates.

A more objective way to select the smoothing parameter λ is to use the "leave-out-one" cross validation function, which is an approximation of the predicted mean squares error. A generalized version of the leave-out-one cross validation function is proposed by Wahba (1990) and is easy to calculate. This generalized cross validation (GCV) function is defined as

$$\mathrm{GCV}(\lambda) = \frac{(1/n)\|(\mathbf{I} - \mathbf{A}(\lambda))\mathbf{y}\|^2}{[(1/n)\mathrm{tr}(\mathbf{I} - \mathbf{A}(\lambda))]^2}$$

The justification for using the GCV function to select λ relies on asymptotic theory. Thus, you cannot expect good results for very small sample sizes or when there is not enough information in the data to separate the model from the error component. Simulation studies suggest that for independent and identically distributed Gaussian noise, you can obtain reliable estimates of λ for n greater than 25 or 30. Note that, even for large values of n (say, $n \geq 50$), in extreme Monte Carlo simulations there might be a small percentage of unwarranted extreme estimates in which $\hat{\lambda} = 0$ or $\hat{\lambda} = \infty$ (Wahba 1983). Generally, if σ^2 is known to within an order of magnitude, the occasional extreme case can be readily identified. As n gets larger, the effect becomes weaker.

The GCV function is fairly robust against nonhomogeneity of variances and non-Gaussian errors (Villalobos and Wahba 1987). Andrews (1988) has provided favorable theoretical results when variances are unequal. However, this selection method is likely to give unsatisfactory results when the errors are highly correlated.

The GCV value might be suspect when λ is extremely small because computed values might become indistinguishable from zero. In practice, calculations with $\lambda = 0$ or λ near 0 can cause numerical instabilities that result in an unsatisfactory solution. Simulation studies have shown that a λ with $\log_{10}(n\lambda) > -8$ is small enough that the final estimate based on this λ almost interpolates the data points. A GCV value based on a $\lambda \leq 10^{-8}$ might not be accurate.

ODS Table Names

PROC TPSPLINE assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. Table 125.4 lists these names. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 125.4 ODS Tables Produced by PROC TPSPLINE

ODS Table Name	Description	Statement	Option
DataSummary	Data summary	PROC	Default
FitSummary	Fit parameters and fit summary	PROC	Default
FitStatistics	Model fit statistics	PROC	Default
GCVFunction	GCV table	MODEL	LOGNLAMBDA, LAMBDA

By referring to the names of such tables, you can use the ODS OUTPUT statement to place one or more of these tables in output data sets.

For example, the following statements create an output data set named FitStats which contains the FitStatistics table, an output data set named DataInfo which contains the DataSummary table, an output data set named ModelInfo which contains the FitSummary table, and an output data set named GCVFunc which contains the GCVFunction table.

```
proc tpspline data=Melanoma;
   model Incidences=Year /LOGNLAMBDA=(-4 to 0 by 0.2);
   ods output FitStatistics = FitStats
              DataSummary   = DataInfo
              FitSummary    = ModelInfo
              GCVFunction   = GCVFunc;
run;
```

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

You can reference every graph produced through ODS Graphics with a name. Table 125.5 lists the names of the graphs, along with the relevant PLOTS= options.

Table 125.5 Graphs Produced by PROC TPSPLINE

ODS Graph Name	Plot Description	PLOTS Option
ContourByLambdaPanel	Panel of thin-plate spline contour surfaces overlaid on scatter plots of data	CONTOURFITPANEL
Contour	Thin-plate spline contour surface overlaid on scatter plot of data	CONTOURFITPANEL
DiagnosticsPanel	Panel of fit diagnostics	DIAGNOSTICS
FitByLambdaPanel	Panel of thin-plate spline curves overlaid on scatter plots of data	FITPANEL
FitPlot	Thin-plate spline curve overlaid on scatter plot of data	FIT
ObservedByPredicted	Dependent variable versus thin-plate spline fit	OBSERVEDBYPREDICTED
QQPlot	Normal quantile plot of residuals	QQPLOT
ResidByLambdaPanel	Panel of residuals versus predictor by smoothing parameter values	RESIDUALBYSMOOTH
ResidualByPredicted	Residuals versus thin-plate spline fit	RESIDUALBYPREDICTED
ResidualHistogram	Histogram of fit residuals	RESIDUALHISTOGRAM
ResidPanel	Panel of residuals versus predictors for fixed smoothing parameter value	RESIDUALS
RFPlot	Side-by-side plots of quantiles of centered fit and residuals	RFPLOT
ScorePlot	Thin-plate spline fit evaluated at scoring points	SCOREPLOT
CriterionPlot	GCV criterion versus smoothing parameter	CRITERION

Examples: TPSPLINE Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 125.1: Partial Spline Model Fit

(View the complete code for this example (tpsplex1.sas) in the example repository.)

This example analyzes the data set Measure that was introduced in the section "Getting Started: TPSPLINE Procedure" on page 10500. That analysis determined that the final estimated surface can be represented by a quadratic function for one or both of the independent variables. This example illustrates how you can use PROC TPSPLINE to fit a partial spline model. The data set Measure is fit by using the following model:

$$y = \beta_0 + \beta_1 x_1 + \beta x_1^2 + f(x_2)$$

The model has a parametric component (associated with the x_1 variable) and a nonparametric component (associated with the x_2 variable). The following statements fit a partial spline model:

```
data Measure;
   set Measure;
   x1sq = x1*x1;
run;

data pred;
   do x1=-1 to 1 by 0.1;
      do x2=-1 to 1 by 0.1;
         x1sq = x1*x1;
         output;
      end;
   end;
run;

proc tpspline data= measure;
   model y = x1 x1sq (x2);
   score data = pred out  = predy;
run;
```

Output 125.1.1 displays the results from these statements.

Output 125.1.1 Output from PROC TPSPLINE

The TPSPLINE Procedure
Dependent Variable: y

Summary of Input Data Set	
Number of Non-Missing Observations	50
Number of Missing Observations	0
Unique Smoothing Design Points	5

Summary of Final Model	
Number of Regression Variables	2
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	4

Output 125.1.1 *continued*

Summary Statistics of Final Estimation	
log10(n*Lambda)	-2.2374
Smoothing Penalty	205.3461
Residual SS	8.5821
Tr(I-A)	43.1534
Model DF	6.8466
Standard Deviation	0.4460
GCV	0.2304

As displayed in Output 125.1.1, there are five unique design points for the smoothing variable x_2 and two regression variables in the model (x_1, x_1^2). The dimension of the polynomial space is the number of columns in $(\{1, x_1, x_1^2, x_2\}) = 4$. The standard deviation of the estimate is much larger than the one based on the model with both x_1 and x_2 as smoothing variables (0.445954 compared to 0.098421). One of the many possible explanations might be that the number of unique design points of the smoothing variable is too small to warrant an accurate estimate for $f(x_2)$.

The following statements produce a surface plot for the partial spline model by using the surface template that is defined in the section "Getting Started: TPSPLINE Procedure" on page 10500.

```
proc sgrender data=predy template=surface;
   dynamic _X='x1' _Y='x2' _Z='P_y' _T='Plot of Fitted Surface on a Fine Grid';
run;
```

The surface displayed in Output 125.1.2 is similar to the one estimated by using the full nonparametric model (displayed in Output 125.2 and Output 125.6).

Output 125.1.2 Plot of PROC TPSPLINE Fit from the Partial Spline Model

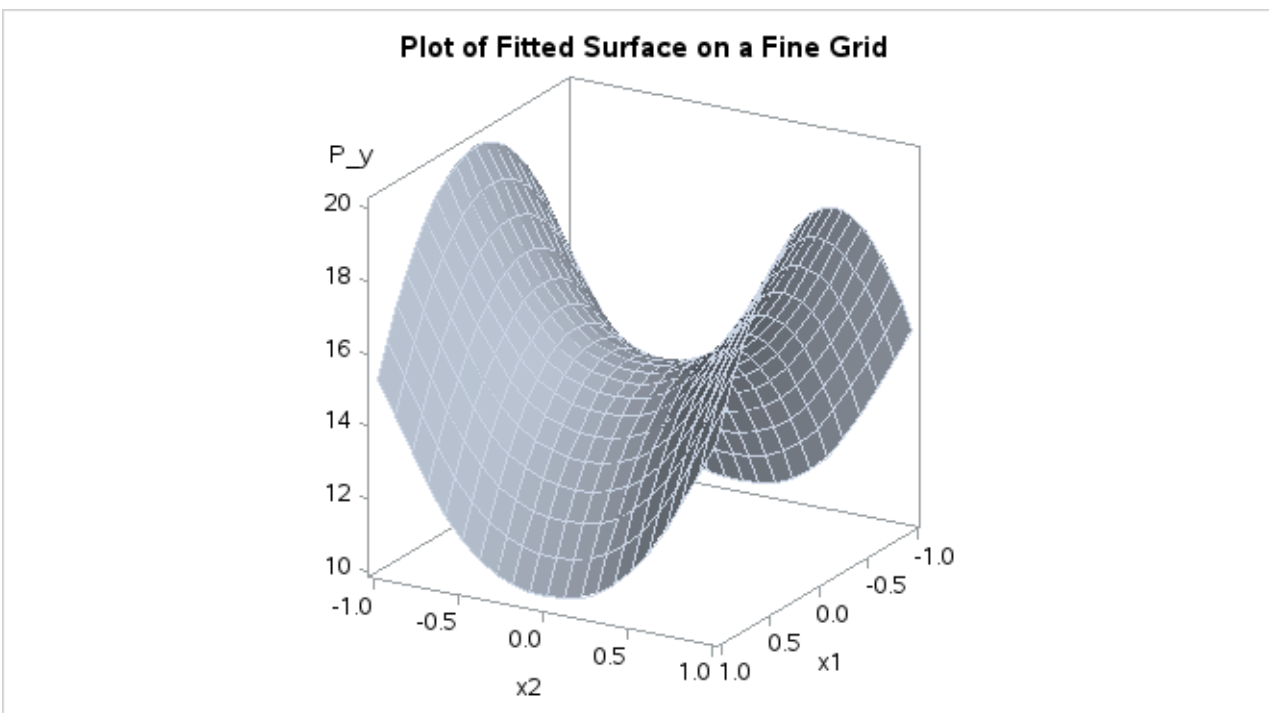

Example 125.2: Spline Model with Higher-Order Penalty

(View the complete code for this example (tpsplex2.sas) in the example repository.)

This example continues the analysis of the data set Measure to illustrate how you can use PROC TPSPLINE to fit a spline model with a higher-order penalty term. Spline models with high-order penalty terms move low-order polynomial terms into the polynomial space. Hence, there is no penalty for these terms, and they can vary without constraint.

As shown in the previous analyses, the final model for the data set Measure must include quadratic terms for both x_1 and x_2. This example fits the following model:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \beta_3 x_2 + \beta_4 x_2^2 + \beta_5 x_1 x_2 + f(x_1, x_2)$$

The model includes quadratic terms for both variables, although it differs from the usual linear model. The nonparametric term $f(x_1, x_2)$ explains the variation of the data that is unaccounted for by a simple quadratic surface.

To modify the order of the derivative in the penalty term, specify the M= option. The following statements specify the option M=3 in order to include the quadratic terms in the polynomial space:

```
data Measure;
   set Measure;
   x1sq = x1*x1;
   x2sq = x2*x2;
   x1x2 = x1*x2;
run;

proc tpspline data= Measure;
   model y = (x1 x2) / m=3;
   score data = pred out = predy;
run;
```

Output 125.2.1 displays the results from these statements.

Output 125.2.1 Output from PROC TPSPLINE with M=3

The TPSPLINE Procedure
Dependent Variable: y

Summary of Input Data Set	
Number of Non-Missing Observations	50
Number of Missing Observations	0
Unique Smoothing Design Points	25

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	2
Order of Derivative in the Penalty	3
Dimension of Polynomial Space	6

Output 125.2.1 *continued*

Summary Statistics of Final Estimation	
log10(n*Lambda)	-3.7831
Smoothing Penalty	2092.4495
Residual SS	0.2731
Tr(I-A)	29.1716
Model DF	20.8284
Standard Deviation	0.0968
GCV	0.0160

The model contains six terms in the polynomial space is the number of columns in $((\{1, x_1, x_1^2, x_1x_2, x_2, x_2^2\}) = 6)$. Compare Output 125.2.1 with Output 125.1.1: the $\log_{10}(n\lambda)$ value and the smoothing penalty differ significantly. In general, these terms are not directly comparable for different models. The final estimate based on this model is close to the estimate based on the model by using the default, M=2.

In the following statements, the REG procedure fits a quadratic surface model to the data set Measure:

```
proc reg data= Measure;
   model y = x1 x1sq x2 x2sq x1x2;
run;
```

The results are displayed in Output 125.2.2.

Output 125.2.2 Quadratic Surface Model: The REG Procedure

The REG Procedure
Model: MODEL1
Dependent Variable: y

Analysis of Variance					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	5	443.20502	88.64100	436.33	<.0001
Error	44	8.93874	0.20315		
Corrected Total	49	452.14376			

Root MSE	0.45073	**R-Square**	0.9802
Dependent Mean	15.08548	**Adj R-Sq**	0.9780
Coeff Var	2.98781		

Parameter Estimates					
Variable	**DF**	**Parameter Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	14.90834	0.12519	119.09	<.0001
x1	1	0.01292	0.09015	0.14	0.8867
x1sq	1	-4.85194	0.15237	-31.84	<.0001
x2	1	0.02618	0.09015	0.29	0.7729
x2sq	1	5.20624	0.15237	34.17	<.0001
x1x2	1	-0.04814	0.12748	-0.38	0.7076

The REG procedure produces slightly different results. To fit a similar model with PROC TPSPLINE, you can use a MODEL statement that specifies the degrees of freedom with the DF= option. You can also use a large value for the LOGNLAMBDA0= option to force a parametric model fit.

Because there is one degree of freedom for each of the terms intercept, x1, x2, x1sq, x2sq, and x1x2, the DF=6 option is used as follows:

```
proc tpspline data=measure;
   model  y=(x1 x2) /m=3 df=6 lognlambda=(-4 to 1 by 0.5);
   score data = pred
         out = predy;
run;
```

The fit statistics are displayed in Output 125.2.3.

Output 125.2.3 Output from PROC TPSPLINE Using M=3 and DF=6

The TPSPLINE Procedure
Dependent Variable: y

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	2
Order of Derivative in the Penalty	3
Dimension of Polynomial Space	6

GCV Function		
log10(n*Lambda)	GCV	
-4.000000	0.016330	*
-3.500000	0.016889	
-3.000000	0.027496	
-2.500000	0.067672	
-2.000000	0.139642	
-1.500000	0.195727	
-1.000000	0.219512	
-0.500000	0.227306	
0	0.229740	
0.500000	0.230504	
1.000000	0.230745	

Note: * indicates minimum GCV value.

Summary Statistics of Final Estimation	
log10(n*Lambda)	2.3830
Smoothing Penalty	0.0000
Residual SS	8.9384
Tr(I-A)	43.9997
Model DF	6.0003
Standard Deviation	0.4507
GCV	0.2309

Output 125.2.4 shows the GCV values for the list of supplied $\log_{10}(n\lambda)$ values in addition to the fitted model

with fixed degrees of freedom 6. The fitted model has a larger GCV value than all other examined models.

Output 125.2.4 Criterion Plot

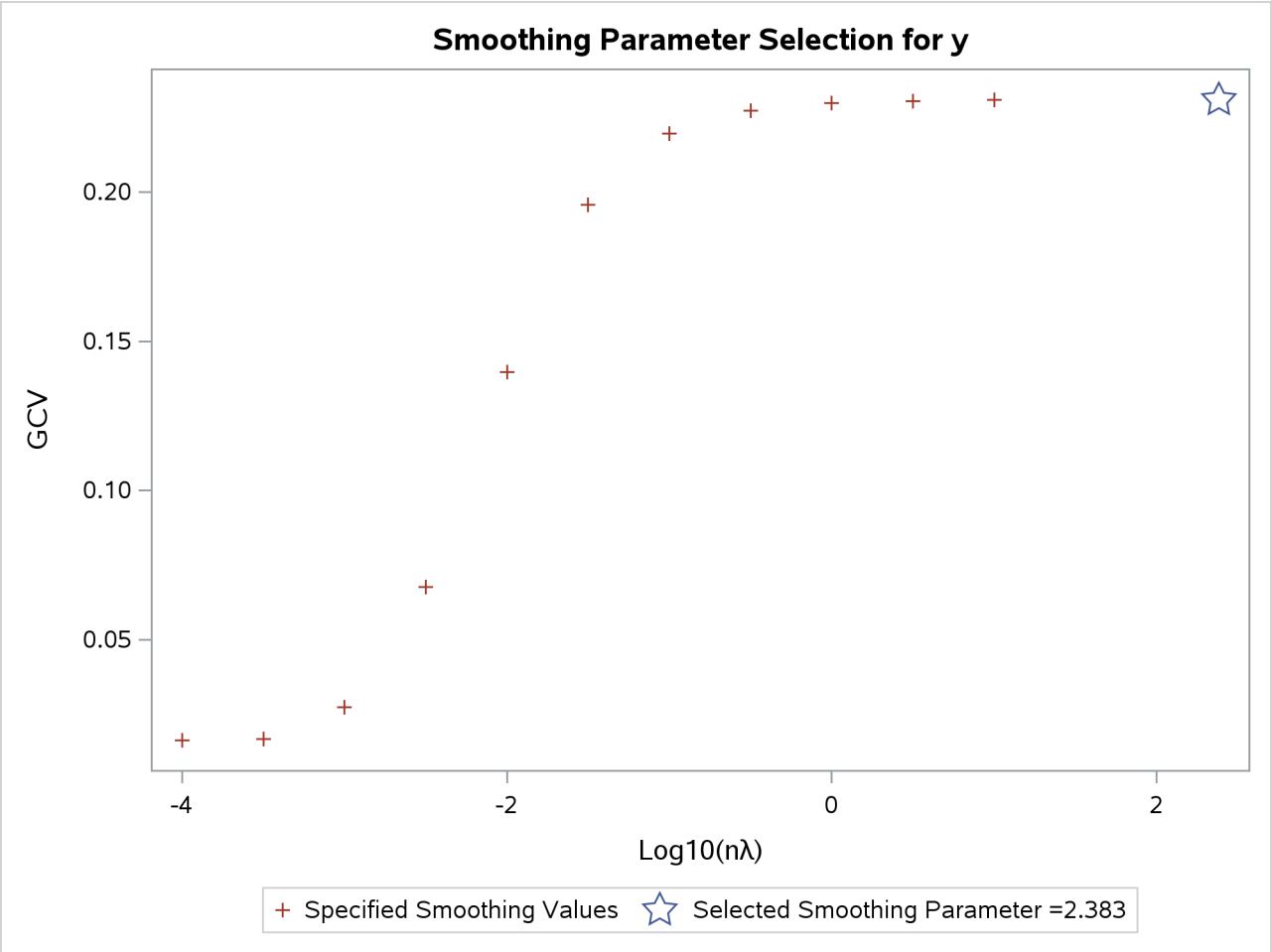

The final estimate is based on 6.000330 degrees of freedom because there are already 6 degrees of freedom in the polynomial space and the search range for λ is not large enough (in this case, setting DF=6 is equivalent to setting $\lambda = \infty$).

The standard deviation and RSS (Output 125.2.3) are close to the sum of squares for the error term and the root MSE from the linear regression model (Output 125.2.2), respectively.

For this model, the optimal $\log_{10}(n\lambda)$ is around –3.8, which produces a standard deviation estimate of 0.096765 (see Output 125.2.1) and a GCV value of 0.016051, while the model that specifies DF=6 results in a $\log_{10}(n\lambda)$ larger than 1 and a GCV value larger than 0.23074. The nonparametric model, based on the GCV, should provide better prediction, but the linear regression model can be more easily interpreted.

Example 125.3: Multiple Minima of the GCV Function

(View the complete code for this example (tpsplex3.sas) in the example repository.)

The data in this example represent the deposition of sulfate (SO_4) at 179 sites in the 48 contiguous states of the United States in 1990. Each observation records the latitude and longitude of the site in addition to the SO_4 deposition at the site measured in grams per square meter (g/m^2).

You can use PROC TPSPLINE to fit a surface that reflects the general trend and that reveals underlying features of the data, which are shown in the following DATA step:

```
data so4;
   input latitude longitude so4 @@;
   datalines;
32.45833  87.24222 1.403 34.28778  85.96889 2.103
33.07139 109.86472 0.299 36.07167 112.15500 0.304
31.95056 112.80000 0.263 33.60500  92.09722 1.950
34.17944  93.09861 2.168 36.08389  92.58694 1.578

   ... more lines ...

43.87333 104.19222 0.306 44.91722 110.42028 0.210
45.07611  72.67556 2.646
;

data pred;
   do latitude = 25 to 47 by 1;
      do longitude = 68 to 124 by 1;
         output;
      end;
   end;
run;
```

The preceding statements create the SAS data set so4 and the data set pred in order to make predictions on a regular grid. The following statements fit a surface for SO_4 deposition:

```
ods graphics on;
proc tpspline data=so4 plots(only)=criterion;
   model so4 = (latitude longitude) /lognlambda=(-6 to 1 by 0.1);
   score data=pred out=prediction1;
run;
```

Partial output from these statements is displayed in Output 125.3.1 and Output 125.3.2.

Output 125.3.1 Partial Output from PROC TPSPLINE for Data Set SO_4

The TPSPLINE Procedure
Dependent Variable: so4

Summary of Input Data Set	
Number of Non-Missing Observations	179
Number of Missing Observations	0
Unique Smoothing Design Points	179

Output 125.3.1 *continued*

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	2
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	3

Output 125.3.2 Partial Output from PROC TPSPLINE for Data Set SO_4

Summary Statistics of Final Estimation	
log10(n*Lambda)	0.2770
Smoothing Penalty	2.4588
Residual SS	12.4450
Tr(I-A)	140.2750
Model DF	38.7250
Standard Deviation	0.2979
GCV	0.1132

Output 125.3.3 displays the CriterionPlot of the GCV function versus $\log_{10}(n\lambda)$.

Output 125.3.3 GCV Function of SO_4 Data Set

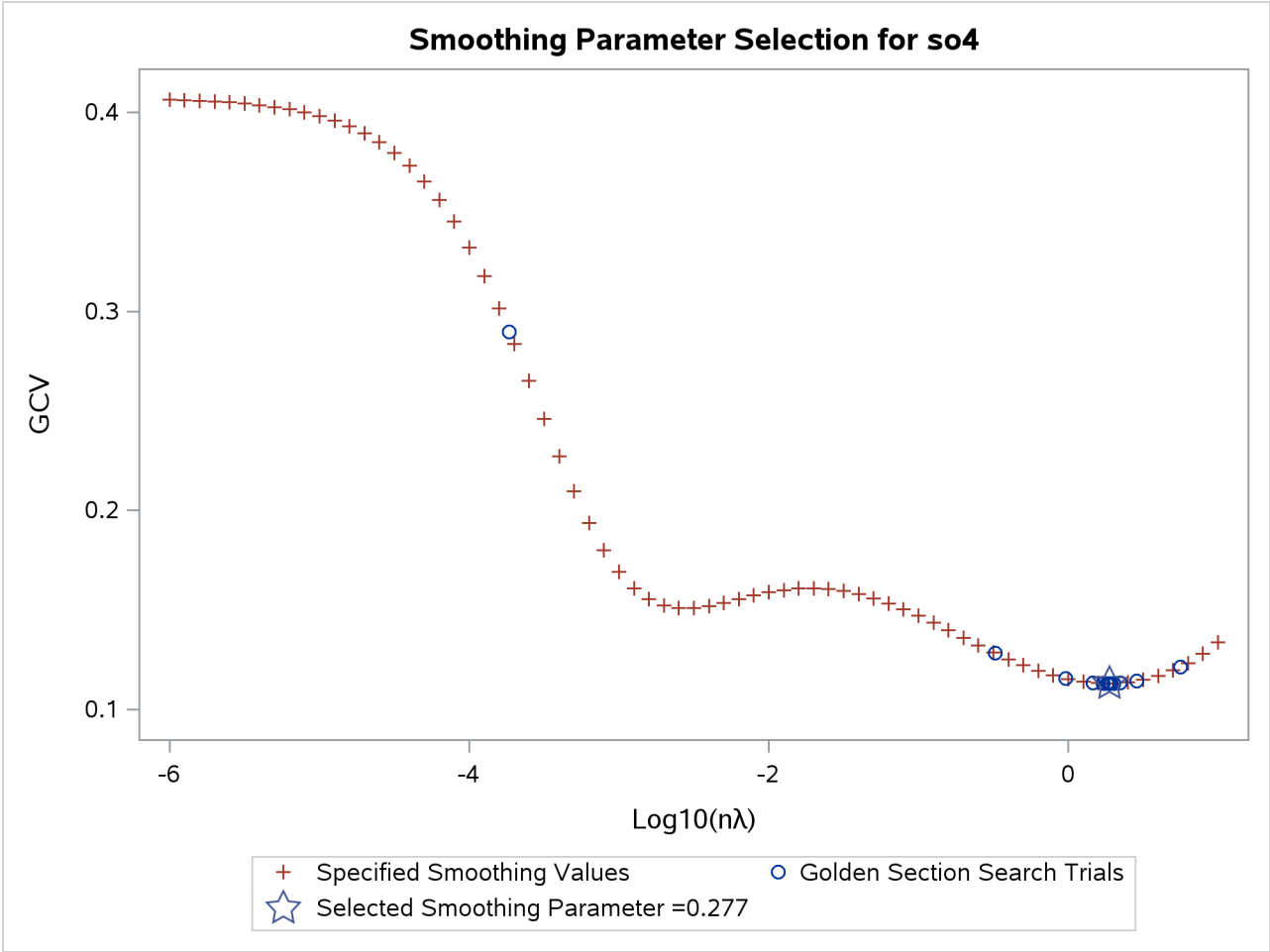

The GCV function has two minima. PROC TPSPLINE locates the global minimum at 0.277005. The plot also displays a local minimum located around –2.56. The TPSPLINE procedure might not always find the global minimum, although it did in this case. If there is a predetermined search range based on prior knowledge, you can use the RANGE= option to narrow the search range in order to find a desired smoothing value. For example, if you believe a better smoothing parameter should be within the $(-4, -2)$ range, you can obtain the model with $\log_{10}(n\lambda) = -2.56$ with the following statements.

```
proc tpspline data=so4;
   model so4 = (latitude longitude) / range=(-4,-2);
   score data=pred out=prediction2;
run;
```

Output 125.3.4 displays the output from PROC TPSPLINE with a specified search range from the smoothing parameter.

Output 125.3.4 Output from PROC TPSPLINE for Data Set SO_4 with $\log_{10}(n\lambda) = -2.56$

The TPSPLINE Procedure
Dependent Variable: so4

Summary of Input Data Set	
Number of Non-Missing Observations	179
Number of Missing Observations	0
Unique Smoothing Design Points	179

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	2
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	3

Summary Statistics of Final Estimation	
log10(n*Lambda)	-2.5600
Smoothing Penalty	177.2160
Residual SS	0.0438
Tr(I-A)	7.2083
Model DF	171.7917
Standard Deviation	0.0779
GCV	0.1508

The smoothing penalty in Output 125.3.4 is much larger than that displayed in Output 125.3.2. The estimate in Output 125.3.2 uses a large λ value; therefore, the surface is smoother than the estimate by using $\log_{10}(n\lambda) = -2.56$ (Output 125.3.4).

The estimate based on $\log_{10}(n\lambda) = -2.56$ has a larger value of degrees of freedom, and it has a much smaller standard deviation.

However, a smaller standard deviation in nonparametric regression does not necessarily mean that the estimate is good: a small λ value always produces an estimate closer to the data and, therefore, a smaller standard deviation.

When ODS Graphics is enabled, you can compare the two fits by supplying 0.277 and –2.56 to the LOGNLAMBDA= option:

```
proc tpspline data=so4;
   model so4 = (latitude longitude) / lognlambda=(0.277 -2.56);
run;
```

Output 125.3.5 shows the contour surfaces of two models with the two minima. The fit that corresponds to the global minimum 0.277 shows a smoother fit that captures the general structure in the data set. The fit at the local minimum –2.56 is a rougher fit that captures local details. The response values are also displayed as circles with the same color gradient by the default GRADIENT *contour-option*. The contrast between the predicted and observed SO_4 deposition is greater for the smoother fit than for the other one, which means the smoother fit has larger absolute residual values.

Output 125.3.5 Panel of Contour Fit Plots by 0.277 and –2.56

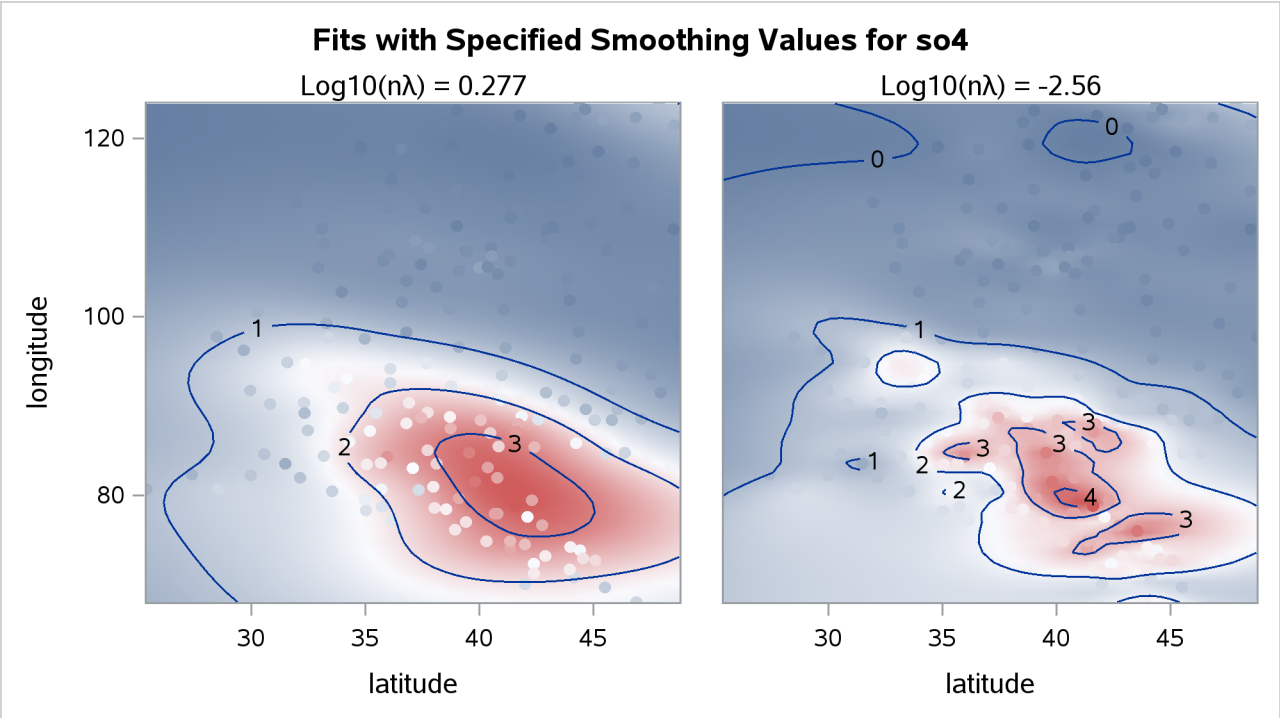

The residuals for the two fits can be visualized in RESIDUALBYSMOOTH panels. Output 125.3.6 is a panel of plots of residuals against smoothing variable Latitude. Output 125.3.7 is a panel of plots of residuals against smoothing variable Longitude. Both panels show that the residuals from the model with the global minimum are larger in absolute values than the ones from the local minimum. This is expected, since the optimal model achieves the smallest GCV value by significantly increasing the smoothness of fit and sacrificing a little in the goodness of fit.

Output 125.3.6 Panel of Residuals by Latitude Plots

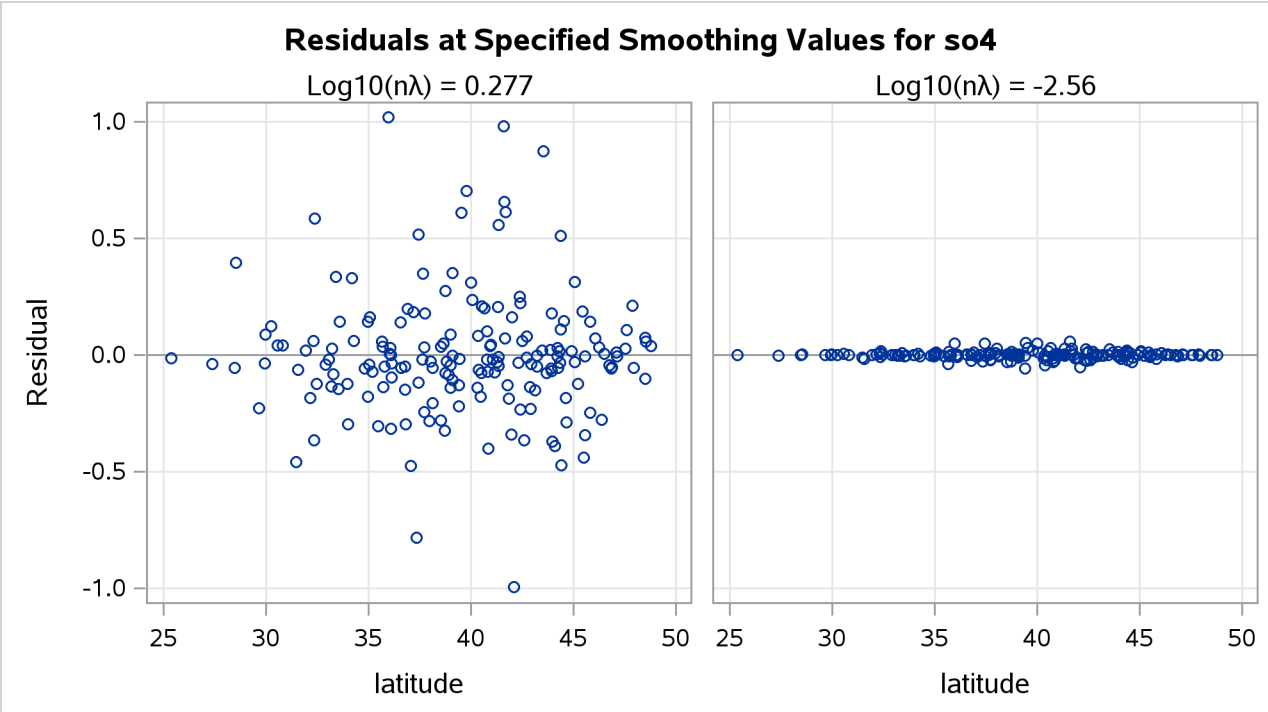

Output 125.3.7 Panel of Residuals by Longitude Plots

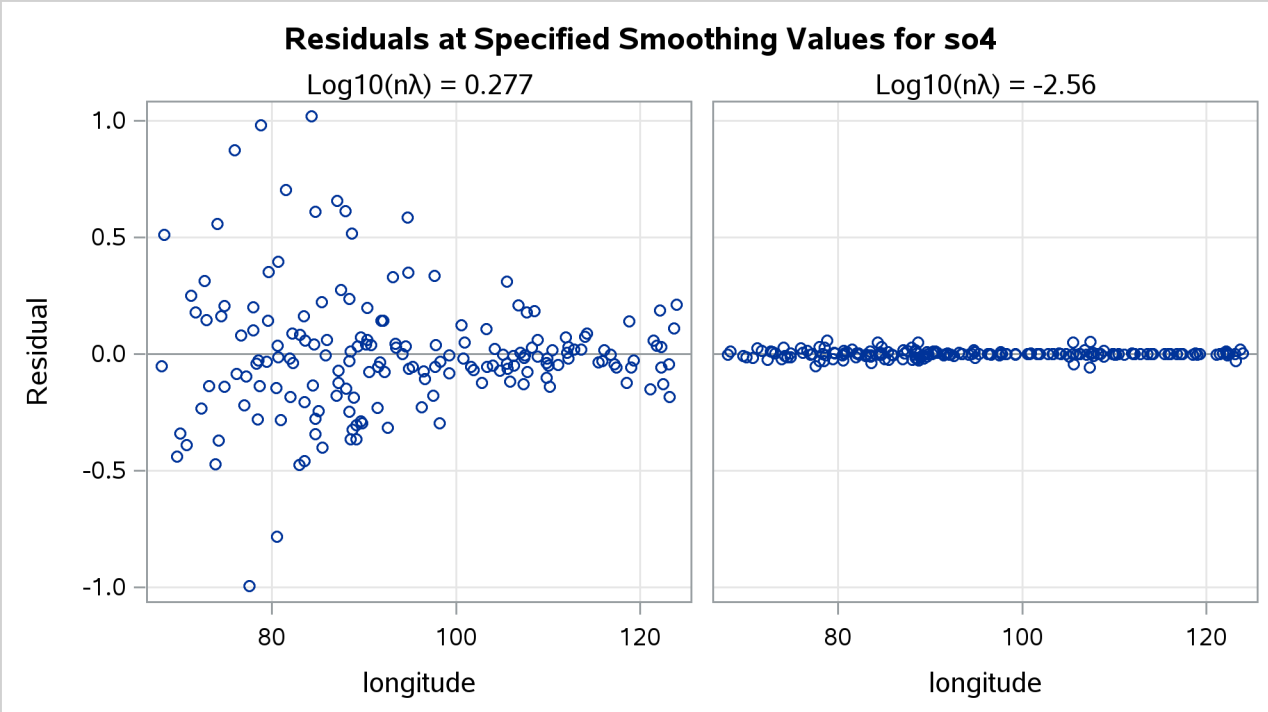

In summary, the fit with $\log_{10}(n\lambda) = 0.277$ represents the underlying surface, while the fit with the

$\log_{10}(n\lambda) = -2.56$ overfits the data and captures the additional noise component.

Example 125.4: Large Data Set Application

(View the complete code for this example (tpsplex4.sas) in the example repository.)

This example illustrates how you can use the D= option to decrease the computation time needed by the TPSPLINE procedure. Although the D= option can be helpful in decreasing computation time for large data sets, it might produce unexpected results when used with small data sets.

The following statements generate the data set large:

```
data large;
   do x=-5 to 5 by 0.02;
      y=5*sin(3*x)+1*rannor(57391);
      output;
   end;
run;
```

The data set large contains 501 observations with one independent variable x and one dependent variable y. The following statements invoke PROC TPSPLINE to produce a thin-plate smoothing spline estimate and the associated 99% confidence interval. The output statistics are saved in the data set fit1.

```
proc tpspline data=large;
   model y  =(x) /lognlambda=(-5 to -1 by 0.2) alpha=0.01;
   output out=fit1 pred lclm uclm;
run;
```

The results from this MODEL statement are displayed in Output 125.4.1.

Output 125.4.1 Output from PROC TPSPLINE without the D= Option

The TPSPLINE Procedure
Dependent Variable: y

Summary of Input Data Set	
Number of Non-Missing Observations	501
Number of Missing Observations	0
Unique Smoothing Design Points	501

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	2

Output 125.4.1 *continued*

GCV Function		
log10(n*Lambda)	**GCV**	
-5.000000	1.258652	
-4.800000	1.228743	
-4.600000	1.205835	
-4.400000	1.188371	
-4.200000	1.174644	
-4.000000	1.163102	
-3.800000	1.152627	
-3.600000	1.142590	
-3.400000	1.132700	
-3.200000	1.122789	
-3.000000	1.112755	
-2.800000	1.102642	
-2.600000	1.092769	
-2.400000	1.083779	
-2.200000	1.076636	
-2.000000	1.072763	*
-1.800000	1.074636	
-1.600000	1.087152	
-1.400000	1.120339	
-1.200000	1.194023	
-1.000000	1.344213	

Note: * indicates minimum GCV value.

Summary Statistics of Final Estimation	
log10(n*Lambda)	-1.9483
Smoothing Penalty	9953.7066
Residual SS	475.0984
Tr(I-A)	471.0861
Model DF	29.9139
Standard Deviation	1.0042
GCV	1.0726

The following statements specify an identical model, but with the additional specification of the D= option. The estimates are obtained by treating nearby points as replicates.

```
proc tpspline data=large;
   model y  =(x) /lognlambda=(-5 to -1 by 0.2) d=0.05 alpha=0.01;
   output out=fit2 pred lclm uclm;
run;
```

The output is displayed in Output 125.4.2.

Output 125.4.2 Output from PROC TPSPLINE with the D= Option

The TPSPLINE Procedure
Dependent Variable: y

Summary of Input Data Set	
Number of Non-Missing Observations	501
Number of Missing Observations	0
Unique Smoothing Design Points	251

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	2

GCV Function		
log10(n*Lambda)	GCV	
-5.000000	1.306536	
-4.800000	1.261692	
-4.600000	1.226881	
-4.400000	1.200060	
-4.200000	1.179284	
-4.000000	1.162776	
-3.800000	1.149072	
-3.600000	1.137120	
-3.400000	1.126220	
-3.200000	1.115884	
-3.000000	1.105766	
-2.800000	1.095730	
-2.600000	1.085972	
-2.400000	1.077066	
-2.200000	1.069954	
-2.000000	1.066076	*
-1.800000	1.067929	
-1.600000	1.080419	
-1.400000	1.113564	
-1.200000	1.187172	
-1.000000	1.337252	

Note: * indicates minimum GCV value.

Summary Statistics of Final Estimation	
log10(n*Lambda)	-1.9477
Smoothing Penalty	9943.5614
Residual SS	472.1424
Tr(I-A)	471.0901
Model DF	29.9099
Standard Deviation	1.0011
GCV	1.0659

The difference between the two estimates is minimal. However, the CPU time for the second MODEL statement is only about 1/7 of the CPU time used in the first model fit.

The following statements produce a plot for comparison of the two estimates:

```
data fit2;
   set fit2;
   P1_y     = P_y;
   LCLM1_y  = LCLM_y;
   UCLM1_y  = UCLM_y;
   drop P_y LCLM_y UCLM_y;
run;

proc sort data=fit1;
   by x y;
run;

proc sort data=fit2;
   by x y;
run;

data comp;
   merge fit1 fit2;
      by x y;
   label p1_y   ="Yhat1" p_y="Yhat0"
         lclm_y ="Lower CL"
         uclm_y ="Upper CL";
run;

proc sgplot data=comp;
   title "Comparison of Two Estimates";
   title2 "with and without the D= Option";

   yaxis label="Predicted y Values";
   xaxis label="x";

   band x=x lower=lclm_y upper=uclm_y /name="range"
                     legendlabel="99% CI of Predicted y without D=";
   series x=x y=P_y/ name="P_y" legendlabel="Predicted y without D="
                     lineattrs=graphfit(thickness=1px pattern=shortdash);
   series x=x y=P1_y/ name="P1_y" legendlabel="Predicted y with D="
                     lineattrs=graphfit(thickness=1px color=red);
   discretelegend "range" "P_y" "P1_y";
run;
```

The estimates from fit1 and fit2 are displayed in Output 125.4.3 with the 99% confidence interval from the fit1 output data set.

Output 125.4.3 Comparison of Two PROC TPSPLINE Fits with and without the D= Option

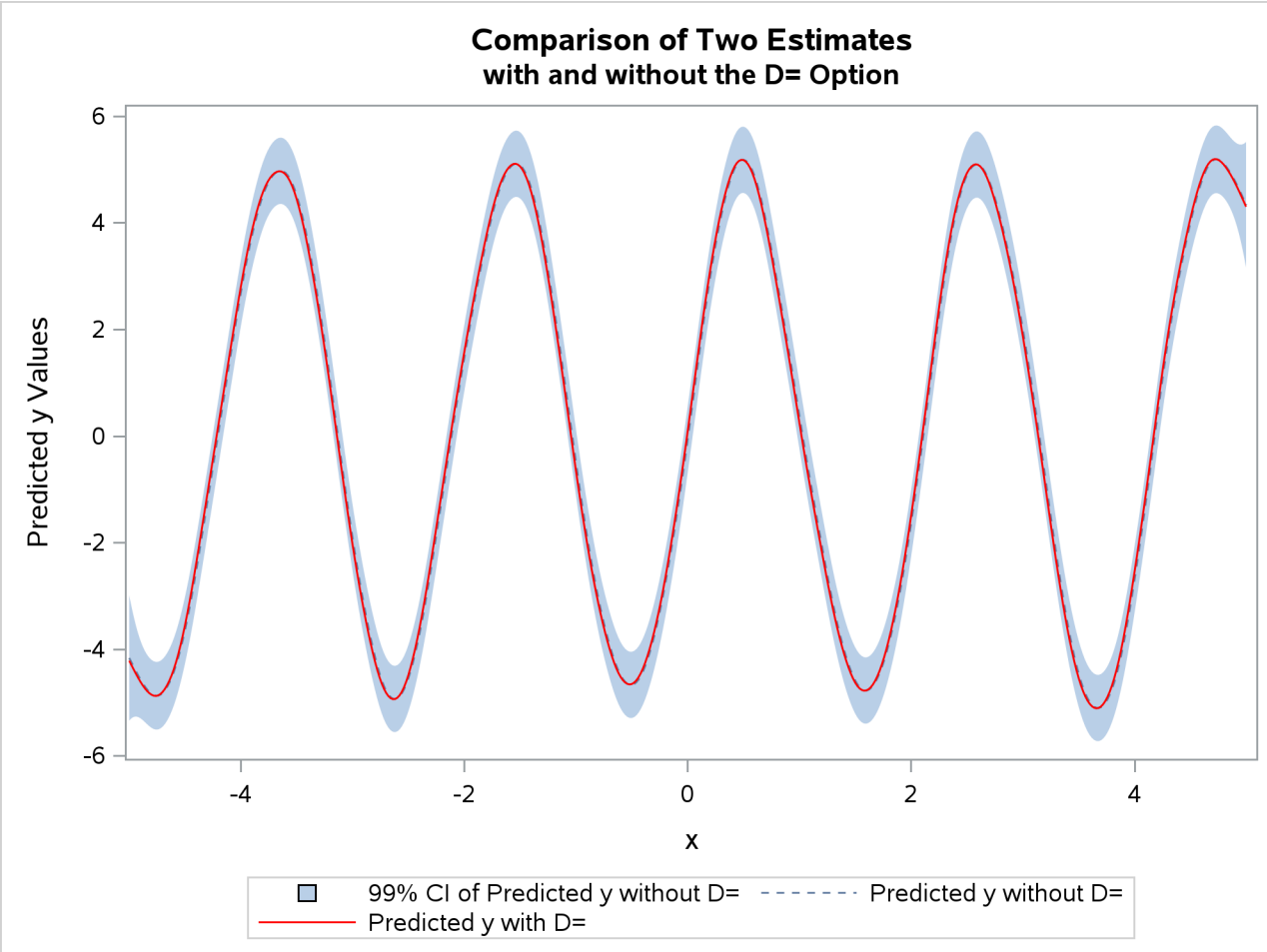

Example 125.5: Computing a Bootstrap Confidence Interval

(View the complete code for this example (tpsplex5.sas) in the example repository.)

This example illustrates how you can construct a bootstrap confidence interval by using the multiple responses feature in PROC TPSPLINE.

Numerous epidemiological observations have indicated that exposure to solar radiation is an important factor in the etiology of melanoma. The following data present age-adjusted melanoma incidences for 37 years from the Connecticut Tumor Registry (Houghton, Flannery, and Viola 1980). The data are analyzed by Ramsay and Silverman (1997).

```
data melanoma;
   input  year incidences @@;
   datalines;
1936    0.9   1937   0.8  1938   0.8  1939   1.3
1940    1.4   1941   1.2  1942   1.7  1943   1.8
1944    1.6   1945   1.5  1946   1.5  1947   2.0
1948    2.5   1949   2.7  1950   2.9  1951   2.5
```

```
1952    3.1   1953   2.4  1954   2.2  1955   2.9
1956    2.5   1957   2.6  1958   3.2  1959   3.8
1960    4.2   1961   3.9  1962   3.7  1963   3.3
1964    3.7   1965   3.9  1966   4.1  1967   3.8
1968    4.7   1969   4.4  1970   4.8  1971   4.8
1972    4.8
;
```

The variable incidences records the number of melanoma cases per 100,000 people for the years 1936 to 1972. The following model fits the data and requests a 90% Bayesian confidence interval along with the estimate:

```
ods graphics on;
proc tpspline data=melanoma plots(only)=(criterionplot fitplot(clm));
   model incidences = (year) /alpha = 0.1;
   output out = result pred uclm lclm;
run;
```

The output is displayed in Output 125.5.1

Output 125.5.1 Output from PROC TPSPLINE for the MELANOMA Data

The TPSPLINE Procedure
Dependent Variable: incidences

Summary of Input Data Set	
Number of Non-Missing Observations	37
Number of Missing Observations	0
Unique Smoothing Design Points	37

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	2

Summary Statistics of Final Estimation	
log10(n*Lambda)	-0.0607
Smoothing Penalty	0.5171
Residual SS	1.2243
Tr(I-A)	22.5852
Model DF	14.4148
Standard Deviation	0.2328
GCV	0.0888

The estimated curve is displayed with 90% confidence interval bands in Output 125.5.2. The number of melanoma incidences exhibits a periodic pattern and increases over the years. The periodic pattern is related to sunspot activity and the accompanying fluctuations in solar radiation.

Output 125.5.2 PROC TPSPLINE Estimate and 90% Confidence Interval of Data Set MELANOMA

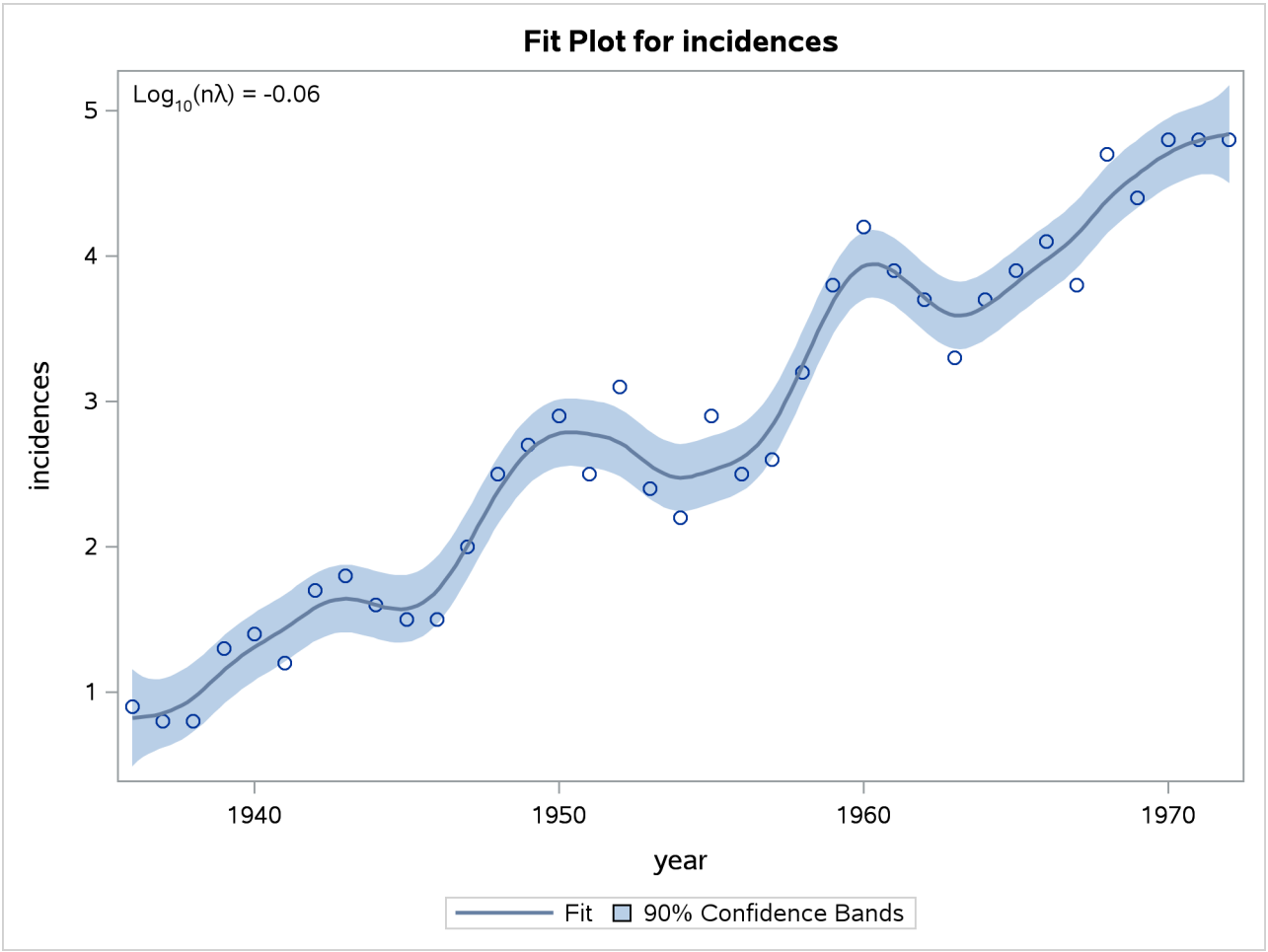

Wang and Wahba (1995) compare several bootstrap confidence intervals to Bayesian confidence intervals for smoothing splines. Both bootstrap and Bayesian confidence intervals are across-the-curve intervals, not pointwise intervals. They concluded that bootstrap confidence intervals work as well as Bayesian intervals concerning average coverage probability. Additionally, bootstrap confidence intervals appear to be better for small sample sizes. Based on their simulation, the "percentile-t interval" bootstrap interval performs better than the other types of bootstrap intervals.

Suppose that $\hat{f}_{\hat{\lambda}}$ and $\hat{\sigma}$ are the estimates of f and σ from the data. Assume that $\hat{f}_{\hat{\lambda}}$ is the "true" f, and generate the bootstrap sample as

$$y_i^* = \hat{f}_{\hat{\lambda}}(\mathbf{x}_i) + \epsilon_i^*, \quad i = 1, \ldots, n$$

where $\boldsymbol{\epsilon}^* = (\epsilon_1^*, \ldots, \epsilon_n^*)' \sim N(0, \hat{\sigma}\mathbf{I})$. Denote $f_{\hat{\lambda}}^*(\mathbf{x}_i)$ as the random variable of the bootstrap estimate at $\mathbf{x}_i$. Repeat this process K times, so that at each point $\mathbf{x}_i$, you have K bootstrap estimates $\hat{f}_{\hat{\lambda}}(\mathbf{x}_i)$ or K realizations of $f_{\hat{\lambda}}^*(\mathbf{x}_i)$. For each fixed $\mathbf{x}_i$, consider the statistic D_i^*, which is similar to the Student's t statistic,

$$D_i^* = \left(f_{\hat{\lambda}}^*(\mathbf{x}_i) - \hat{f}_{\hat{\lambda}}(\mathbf{x}_i)\right) / \hat{\sigma}_i{}^*$$

where $\hat{\sigma}_i{}^*$ is the estimate of $\hat{\sigma}$ based on the ith bootstrap sample.

Suppose $\chi_{\alpha/2}$ and $\chi_{1-\alpha/2}$ are the lower and upper $\alpha/2$ points, respectively, of the empirical distribution of D_i^*. The $(1-\alpha)100\%$ bootstrap confidence interval is defined as

$$\left(\hat{f}_{\hat{\lambda}}(\mathbf{x}_i) - \chi_{1-\alpha/2}\hat{\sigma},\ \ \hat{f}_{\hat{\lambda}}(\mathbf{x}_i) - \chi_{\alpha/2}\hat{\sigma}\right)$$

Bootstrap confidence intervals are easy to interpret and can be used with any distribution. However, because they require K model fits, their construction is computationally intensive.

The feature of multiple dependent variables in PROC TPSPLINE enables you to fit multiple models with the same independent variables. The procedure calculates the matrix decomposition part of the calculations only once, regardless of the number of dependent variables in the model. These calculations are responsible for most of the computing time used by the TPSPLINE procedure. This feature is particularly useful when you need to generate a bootstrap confidence interval.

To construct a bootstrap confidence interval, perform the following tasks:

- Fit the data by using PROC TPSPLINE and obtain estimates $\hat{f}_{\hat{\lambda}}(\mathbf{x}_i)$ and $\hat{\sigma}$.
- Generate K bootstrap samples based on $\hat{f}_{\hat{\lambda}}(\mathbf{x}_i)$ and $\hat{\sigma}$.
- Fit the K bootstrap samples with the TPSPLINE procedure to obtain estimates of $\hat{f}^*_{\hat{\lambda}}(\mathbf{x}_i)$ and $\hat{\sigma}_i^*$.
- Compute D_i^* and the values $\chi_{\alpha/2}$ and $\chi_{1-\alpha/2}$.

The following statements illustrate this process:

```
proc tpspline data=melanoma plots(only)=fitplot(clm);
   model incidences = (year) /alpha = 0.1;
   output out=result pred uclm lclm;
run;
```

The output from the initial PROC TPSPLINE analysis is displayed in Output 125.5.3. The data set result contains the predicted values and confidence limits from the analysis.

Output 125.5.3 Output from PROC TPSPLINE for the MELANOMA Data

The TPSPLINE Procedure
Dependent Variable: incidences

Summary of Input Data Set	
Number of Non-Missing Observations	37
Number of Missing Observations	0
Unique Smoothing Design Points	37

Summary of Final Model	
Number of Regression Variables	0
Number of Smoothing Variables	1
Order of Derivative in the Penalty	2
Dimension of Polynomial Space	2

Output 125.5.3 *continued*

Summary Statistics of Final Estimation	
log10(n*Lambda)	-0.0607
Smoothing Penalty	0.5171
Residual SS	1.2243
Tr(I-A)	22.5852
Model DF	14.4148
Standard Deviation	0.2328
GCV	0.0888

The following statements illustrate how you can obtain a bootstrap confidence interval for the Melanoma data set. The following statements create the data set bootstrap. The observations are created with information from the preceding PROC TPSPLINE execution; as displayed in Output 125.5.3, $\hat{\sigma} = 0.232823$. The values of $\hat{f}_{\hat{\lambda}}(\mathrm{x}_i)$ are stored in the data set result in the variable P_incidence.

```
data bootstrap;
   set result;
   array y{1070} y1-y1070;
   do i=1 to 1070;
      y{i} = p_incidences + 0.232823*rannor(123456789);
   end;
   keep y1-y1070 p_incidences year;
run;

ods exclude all;
proc tpspline data=bootstrap plots=none;
   ods output FitStatistics=FitResult;
   id p_incidences;
   model y1-y1070 = (year);
   output out=result2;
run;
ods exclude none;
```

The DATA step generates 1,070 bootstrap samples based on the previous estimate from PROC TPSPLINE. For this data set, some of the bootstrap samples result in λs (selected by the GCV function) that cause problematic behavior. Thus, an additional 70 bootstrap samples are generated.

Displayed output is excluded before PROC TPSPLINE is invoked. The PLOTS=NONE option suppresses all graphical output. The model fits all the y1...y1070 variables as dependent variables, and the models are fit for all bootstrap samples simultaneously. The output data set result2 contains the variables year, y1...y1070, p_y1...p_y1070, and p_incidences.

The ODS OUTPUT statement writes the FitStatistics table to the data set FitResult. The data set FitResult contains the two variables Parameter and Value. The FitResult data set is used in subsequent calculations for D_i^*.

In the data set FitResult, there are 63 estimates with a standard deviation of zero, suggesting that the estimates provide perfect fits of the data and are caused by $\hat{\lambda}$s that are approximately equal to zero. For small sample sizes, there is a positive probability that the λ chosen by the GCV function will be zero (Wang and Wahba 1995).

In the following steps, these cases are removed from the bootstrap samples as "bad" samples: they represent failure of the GCV function.

The following SAS statements manipulate the data set FitResult, retaining the standard deviations for all bootstrap samples and merging FitResult with the data set result2, which contains the estimates for bootstrap samples. In the final data set boot, the D_i^* statistics are calculated.

```
data FitResult;
   set FitResult;
   if Parameter="Standard Deviation";
   keep Value;
run;

proc transpose data=FitResult out=sd prefix=sd;

data result2;
   if _N_ = 1 then set sd;
   set result2;
run;

data boot;
   set result2;
   array y{1070}  p_y1-p_y1070;
   array sd{1070} sd1-sd1070;
   do i=1 to 1070;
      if sd{i} > 0 then do;
         d = (y{i} - P_incidences)/sd{i};
         obs = _N_;
         output;
      end;
   end;
   keep d obs P_incidences year;
run;
```

The following SAS statements retain the first 1,000 bootstrap samples and calculate the values $\chi_{\alpha/2}$ and $\chi_{1-\alpha/2}$ with $\alpha = 0.1$.

```
proc sort data=boot;
   by obs;
run;

data boot;
   set boot;
      by obs;
   retain n;

   if first.obs then n=1;
      else n=n+1;
   if n > 1000 then delete;
run;

proc sort data=boot;
   by obs d;
run;
```

```
data chi1 chi2 ;
   set boot;
   if (_N_ = (obs-1)*1000+50)  then output chi1;
   if (_N_ = (obs-1)*1000+950) then output chi2;
run;

proc sort data=result;
   by year;
run;

proc sort data=chi1;
   by year;
run;

proc sort data=chi2;
   by year;
run;

data result;
   merge result
      chi1(rename=(d=chi05))
      chi2(rename=(d=chi95));
   keep year incidences P_incidences lower upper
        LCLM_incidences UCLM_incidences;

   lower = -chi95*0.232823 + P_incidences;
   upper = -chi05*0.232823 + P_incidences;

   label  lower="Lower 90% CL (Bootstrap)"
          upper="Upper 90% CL (Bootstrap)"
          lclm_incidences="Lower 90% CL (Bayesian)"
          uclm_incidences="Upper 90% CL (Bayesian)";
run;
```

The data set result contains the variables year and incidences, the PROC TPSPLINE estimate P_incidences, and the 90% Bayesian and 90% bootstrap confidence intervals.

The following statements produce Output 125.5.4:

```
proc sgplot data=result;
   title "Age-adjusted Melanoma Incidence for 37 Years";

   xaxis label="year";
   yaxis label="Incidences";

   band x=year lower=lclm_incidences upper=uclm_incidences/name="bayesian"
               legendlabel="90% Bayesian CI of Predicted incidences"
               fillattrs=(color=red);
   band x=year lower=lower upper=upper/name="bootstrap"
               legendlabel="90% Bootstrap CI of Predicted incidences"
               transparency=0.05;
   scatter x=year y=incidences/name="obs" legendlabel="incidences";
   series x=year y=p_incidences/name="pred"
               legendlabel="predicted values of incidences"
```

```
                lineattrs=graphfit(thickness=1px);

   discretelegend "bayesian" "bootstrap" "obs" "pred";
run;

ods graphics off;
```

Output 125.5.4 displays the plot of the variable incidences, the predicted values, and the Bayesian and bootstrap confidence intervals.

The plot shows that the bootstrap confidence interval is similar to the Bayesian confidence interval. However, the Bayesian confidence interval is symmetric around the estimates, while the bootstrap confidence interval is not.

Output 125.5.4 Comparison of Bayesian and Bootstrap Confidence Interval for Data Set MELANOMA

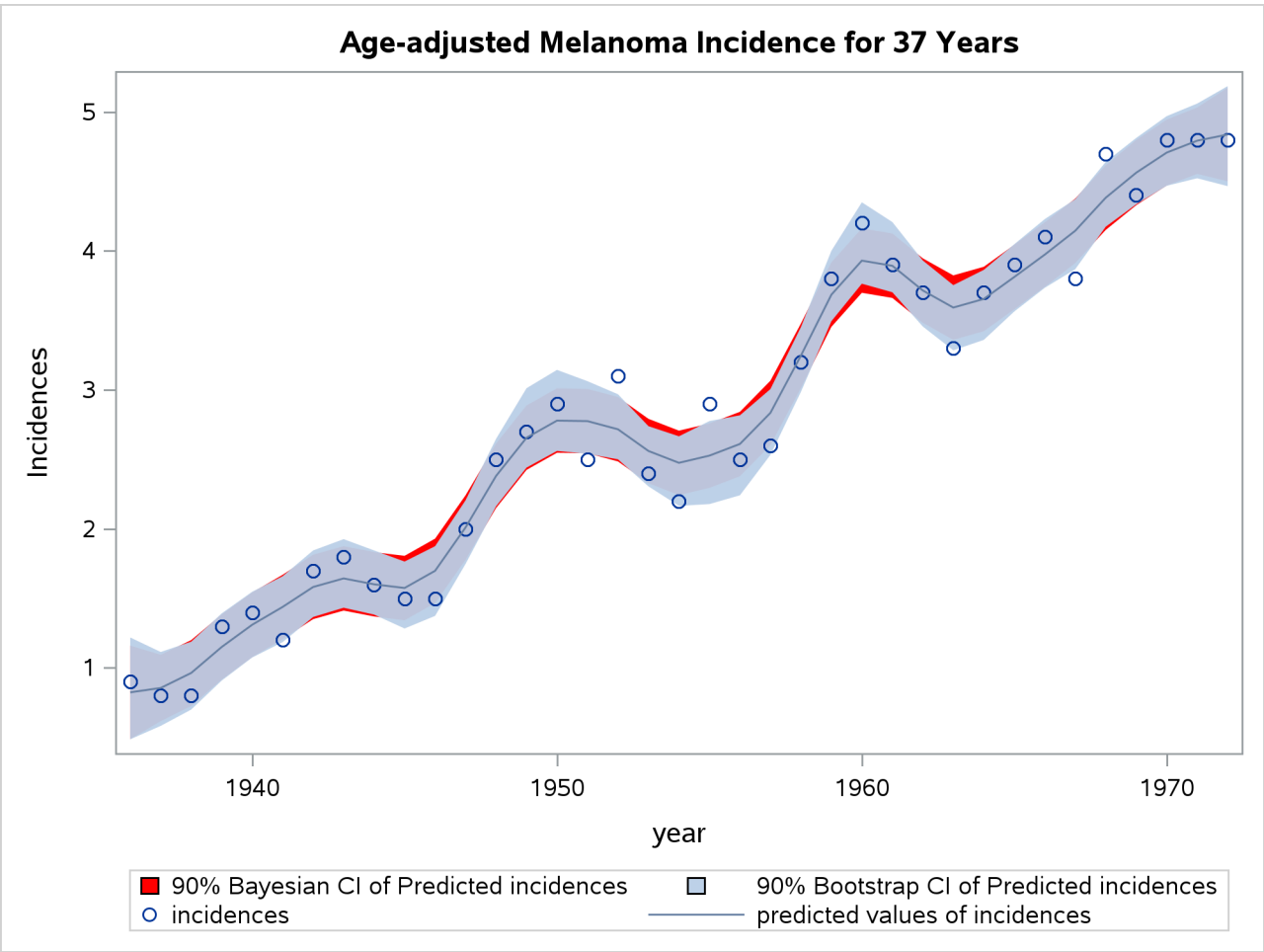

References

Andrews, D. W. K. (1988). *Asymptotic Optimality of Generalized C_L, Cross-Validation, and Generalized Cross-Validation in Regression with Heteroskedastic Errors*. Cowles Foundation Discussion Paper 906,

Cowles Foundation for Research in Economics at Yale University. Revised May 1989.

Bates, D. M., Lindstrom, M. J., Wahba, G., and Yandell, B. S. (1987). "GCVPACK—Routines for Generalized Cross Validation." *Communications in Statistics—Simulation and Computation* 16:263–297.

Cleveland, W. S. (1993). *Visualizing Data*. Summit, NJ: Hobart Press.

Dongarra, J. J., Bunch, J. R., Moler, C. B., and Steward, G. W. (1979). *Linpack Users' Guide*. Philadelphia: SIAM.

Duchon, J. (1976). "Fonctions-spline et espérances conditionnelles de champs gaussiens [Spline functions and conditional expectations of Gaussian fields]." *Annales scientifiques de l'Université de Clermont-Ferrand 2, Série Mathématique* 14:19–27.

Duchon, J. (1977). "Splines Minimizing Rotation-Invariant Semi-norms in Sobolev Spaces." In *Constructive Theory of Functions of Several Variables*, edited by W. Schempp and K. Zeller, 85–100. New York: Springer-Verlag.

Hall, P., and Titterington, D. M. (1987). "Common Structure of Techniques for Choosing Smoothing Parameters in Regression Problems." *Journal of the Royal Statistical Society, Series B* 49:184–198.

Houghton, A. N., Flannery, J., and Viola, M. V. (1980). "Malignant Melanoma in Connecticut and Denmark." *International Journal of Cancer* 25:95–104.

Hutchinson, M., and Bischof, R. (1983). "A New Method for Estimating the Spatial Distribution of Mean Seasonal and Annual Rainfall Applied to the Hunter Valley." *Australian Meteorological Magazine* 31:179–184.

Meinguet, J. (1979). "Multivariate Interpolation at Arbitrary Points Made Simple." *Journal of Applied Mathematics and Physics* 30:292–304.

Nychka, D. (1986a). *A Frequency Interpretation of Bayesian "Confidence" Intervals for Smoothing Splines*. Technical Report 1699, North Carolina State University.

Nychka, D. (1986b). *The Mean Posterior Variance of a Smoothing Spline and a Consistent Estimate of the Mean Squared Error*. Technical Report 1688, North Carolina State University.

Nychka, D. (1988). "Bayesian Confidence Intervals for Smoothing Splines." *Journal of the American Statistical Association* 83:1134–1143.

O'Sullivan, F., and Wong, T. (1987). *Determining a Function Diffusion Coefficient in the Heat Equation*. Technical report, Department of Statistics, University of California, Berkeley.

Ramsay, J. O., and Silverman, B. W. (1997). *Functional Data Analysis*. New York: Springer-Verlag.

Seaman, R., and Hutchinson, M. (1985). "Comparative Real Data Tests of Some Objective Analysis Methods by Withholding." *Australian Meteorological Magazine* 33:37–46.

Villalobos, M., and Wahba, G. (1987). "Inequality Constrained Multivariate Smoothing Splines with Application to the Estimation of Posterior Probabilities." *Journal of the American Statistical Association* 82:239–248.

Wahba, G. (1983). "Bayesian 'Confidence Intervals' for the Cross Validated Smoothing Spline." *Journal of the Royal Statistical Society, Series B* 45:133–150.

Wahba, G. (1990). *Spline Models for Observational Data*. Philadelphia: SIAM.

Wahba, G., and Wendelberger, J. (1980). "Some New Mathematical Methods for Variational Objective Analysis Using Splines and Cross Validation." *Monthly Weather Review* 108:1122–1145.

Wang, Y., and Wahba, G. (1995). "Bootstrap Confidence Intervals for Smoothing Splines and Their Comparison to Bayesian Confidence Intervals." *Journal of Statistical Computation and Simulation* 51:263–279.

Chapter 126
The TRANSREG Procedure

Contents

Overview: TRANSREG Procedure . . . **10554**
Getting Started: TRANSREG Procedure . . . **10556**
- Fitting a Curve through a Scatter Plot . . . 10556
- Main-Effects ANOVA . . . 10570

Syntax: TRANSREG Procedure . . . **10573**
- PROC TRANSREG Statement . . . 10574
- BY Statement . . . 10581
- FREQ Statement . . . 10581
- ID Statement . . . 10582
- MODEL Statement . . . 10582
- OUTPUT Statement . . . 10609
- WEIGHT Statement . . . 10619

Details: TRANSREG Procedure . . . **10620**
- Model Statement Usage . . . 10620
- Box-Cox Transformations . . . 10623
- Using Splines and Knots . . . 10632
- Scoring Spline Variables . . . 10643
- Linear and Nonlinear Regression Functions . . . 10647
- Simultaneously Fitting Two Regression Functions . . . 10651
- Penalized B-Splines . . . 10657
- Smoothing Splines . . . 10660
- ANOVA Codings . . . 10664
- Missing Values . . . 10681
- Missing Values, UNTIE, and Hypothesis Tests . . . 10682
- Controlling the Number of Iterations . . . 10683
- Using the REITERATE Algorithm Option . . . 10684
- Avoiding Constant Transformations . . . 10685
- Constant Variables . . . 10686
- Character OPSCORE Variables . . . 10686
- Convergence and Degeneracies . . . 10686
- Implicit and Explicit Intercepts . . . 10687
- Passive Observations . . . 10687
- Point Models . . . 10687
- Redundancy Analysis . . . 10688
- Optimal Scaling . . . 10691
- OPSCORE, MONOTONE, UNTIE, and LINEAR Transformations . . . 10692

SPLINE and MSPLINE Transformations . . . 10693
Specifying the Number of Knots . . . 10694
SPLINE, BSPLINE, and PSPLINE Comparisons . . . 10696
Hypothesis Tests . . . 10696
Output Data Set . . . 10698
OUTTEST= Output Data Set . . . 10706
Computational Resources . . . 10707
Unbalanced ANOVA without CLASS Variables . . . 10708
Hypothesis Tests for Simple Univariate Models . . . 10709
Hypothesis Tests with Monotonicity Constraints . . . 10715
Hypothesis Tests with Dependent Variable Transformations . . . 10716
Hypothesis Tests with One-Way ANOVA . . . 10719
Using the DESIGN Output Option . . . 10722
Discrete Choice Experiments: DESIGN, NORESTORE, NOZERO . . . 10726
Centering . . . 10727
Displayed Output . . . 10728
ODS Table Names . . . 10728
ODS Graphics . . . 10730
Examples: TRANSREG Procedure . . . **10735**
Example 126.1: Transformation Regression of Exhaust Emissions Data . . . 10735
Example 126.2: Box-Cox Transformations . . . 10742
Example 126.3: Penalized B-Spline . . . 10746
Example 126.4: Nonmetric Conjoint Analysis of Tire Data . . . 10752
Example 126.5: Metric Conjoint Analysis of Tire Data . . . 10755
Example 126.6: Preference Mapping of Automobile Data . . . 10767
References . . . **10773**

Overview: TRANSREG Procedure

The TRANSREG (transformation regression) procedure fits linear models, optionally with smooth, spline, Box-Cox, and other nonlinear transformations of the variables. You can use PROC TRANSREG to fit a curve through a scatter plot or fit multiple curves, one for each level of a classification variable. You can also constrain the functions to be parallel or monotone or have the same intercept. PROC TRANSREG can be used to code experimental designs and classification variables prior to their use in other analyses.

The TRANSREG procedure fits many types of linear models, including the following:

- ordinary regression and ANOVA
- metric and nonmetric conjoint analysis (Green and Wind 1975; De Leeuw, Young, and Takane 1976)
- linear models with Box-Cox (1964) transformations of the dependent variables

- regression with a smooth (Reinsch 1967), spline (De Boor 1978; Van Rijckevorsel 1982), monotone spline (Winsberg and Ramsay 1980), or penalized B-spline (Eilers and Marx 1996) fit function
- metric and nonmetric vector and ideal point preference mapping (Carroll 1972)
- simple, multiple, and multivariate regression with variable transformations (Young, de Leeuw, and Takane 1976; Winsberg and Ramsay 1980; Breiman and Friedman 1985)
- redundancy analysis (Stewart and Love 1968) with variable transformations (Israels 1984)
- canonical correlation analysis with variable transformations (Van der Burg and de Leeuw 1983)
- response surface regression (Myers 1976; Khuri and Cornell 1987) with variable transformations

The data set can contain variables measured on nominal, ordinal, interval, and ratio scales (Siegel 1956). You can specify any mix of these variable types for the dependent and independent variables. PROC TRANSREG can do the following:

- transform nominal variables by scoring the categories to minimize squared error (Fisher 1938), or treat nominal variables as classification variables
- transform ordinal variables by monotonically scoring the ordered categories so that order is weakly preserved (adjacent categories can be merged) and squared error is minimized. Ties can be optimally untied or left tied (Kruskal 1964). Ordinal variables can also be transformed to ranks.
- transform interval and ratio scale of measurement variables linearly or nonlinearly with spline (De Boor 1978; Van Rijckevorsel 1982), monotone spline (Winsberg and Ramsay 1980), penalized B-spline (Eilers and Marx 1996), smooth (Reinsch 1967), or Box-Cox (Box and Cox 1964) transformations. In addition, logarithmic, exponential, power, logit, and inverse trigonometric sine transformations are available.

Transformations produced by the PROC TRANSREG multiple regression algorithm, requesting spline transformations, are often similar to transformations produced by the ACE smooth regression method of Breiman and Friedman (1985). However, ACE does not explicitly optimize a loss function (De Leeuw 1986), while PROC TRANSREG explicitly minimizes a squared-error criterion.

PROC TRANSREG extends the ordinary general linear model by providing optimal variable transformations that are iteratively derived. PROC TRANSREG iterates until convergence, alternating two major steps: finding least squares estimates of the model parameters given the current scoring of the data, and finding least squares estimates of the scoring parameters given the current set of model parameters. This is called the method of alternating least squares (Young 1981).

For more background on alternating least squares optimal scaling methods and transformation regression methods, see Young, de Leeuw, and Takane (1976); Winsberg and Ramsay (1980); Young (1981); Gifi (1990); Schiffman, Reynolds, and Young (1981); Van der Burg and de Leeuw (1983); Israels (1984); Breiman and Friedman (1985); Hastie and Tibshirani (1986). (These are just a few of the many relevant sources.)

Getting Started: TRANSREG Procedure

This section provides several examples that illustrate a few of the more basic features of PROC TRANSREG.

Fitting a Curve through a Scatter Plot

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/treggs1.sas.)

PROC TRANSREG can fit curves through data and detect nonlinear relationships among variables. This example uses a subset of the data from an experiment in which nitrogen oxide emissions from a single cylinder engine are measured for various combinations of fuel and equivalence ratio (Brinkman 1981). This gas data set is available from the Sashelp library. The following step creates a subset of the data for analysis:

```
title 'Gasoline and Emissions Data';

data gas;
   set sashelp.gas;
   if fuel in ('Ethanol', '82rongas', 'Gasohol');
run;
```

The next step fits a spline or curve through the data and displays the regression results. For information about splines and knots, see the sections "Smoothing Splines" on page 10660, "Linear and Nonlinear Regression Functions" on page 10647, "Simultaneously Fitting Two Regression Functions" on page 10651, and "Using Splines and Knots" on page 10632, as well as Example 126.1. The following statements produce Figure 126.1:

```
ods graphics on;

* Request a Spline Transformation of Equivalence Ratio;
proc transreg data=Gas solve ss2 plots=(transformation obp residuals);
   model identity(nox) = spline(EqRatio / nknots=4);
run;
```

The SOLVE algorithm option, or *a-option*, requests a direct solution for both the transformation and the parameter estimates. For many models, PROC TRANSREG with the SOLVE *a-option* can produce exact results without iteration. The SS2 (Type II sums of squares) *a-option* requests regression and ANOVA results. The PLOTS= option requests plots of the variable transformations, a plot of the observed values by the predicted values, and a plot of the residuals. The dependent variable NOx was specified with an IDENTITY transformation, which means that it will not be transformed, just as in ordinary regression. The independent variable EqRatio, in contrast, is transformed by using a cubic spline with four knots. The NKNOTS= option is known as a transformation option, or *t-option*. Graphical results are enabled when ODS Graphics is enabled. The results are shown in Figure 126.1 through Figure 126.5.

Figure 126.1 Iteration, ANOVA, and Regression Results

Gasoline and Emissions Data

The TRANSREG Procedure

Dependent Variable Identity(NOx)
Nitrogen Oxide

Number of Observations Read	112
Number of Observations Used	110

TRANSREG MORALS Algorithm Iteration History for Identity(NOx)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	1.04965	3.46121	0.00917		
1	0.00000	0.00000	0.82429	0.81512	Converged

Algorithm converged.

The TRANSREG Procedure Hypothesis Tests for Identity(NOx)
Nitrogen Oxide

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	180.0951	25.72788	68.36	<.0001
Error	102	38.3891	0.37636		
Corrected Total	109	218.4842			

Root MSE	0.61348	**R-Square**	0.8243
Dependent Mean	2.25022	**Adj R-Sq**	0.8122
Coeff Var	27.26334		

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Intercept	1	8.3165407	324.065	324.065	861.04	<.0001	Intercept
Spline(EqRatio)	7	-6.5740158	180.095	25.728	68.36	<.0001	Equivalence Ratio

PROC TRANSREG increases the squared multiple correlation from the original value of 0.00917 to 0.82429. Iteration 0 shows the fit before the data are transformed, and iteration 1 shows the fit after the transformation, which was directly solved for in the initial iteration. The change values for iteration 0 show the change from the original EqRatio variable to the transformed EqRatio variable. For this model, no improvement on the initial solution is possible, so in iteration 1, all change values are zero. The ANOVA and regression results show that you are fitting a model with 7 model parameters, 4 knots plus a degree 3 or cubic spline. The overall model fit is identical to the test for the spline transformation, since there is only one term in the model besides the intercept, and the results are significant at the 0.0001 level. The transformations are shown next in Figure 126.2.

Figure 126.2 Transformations

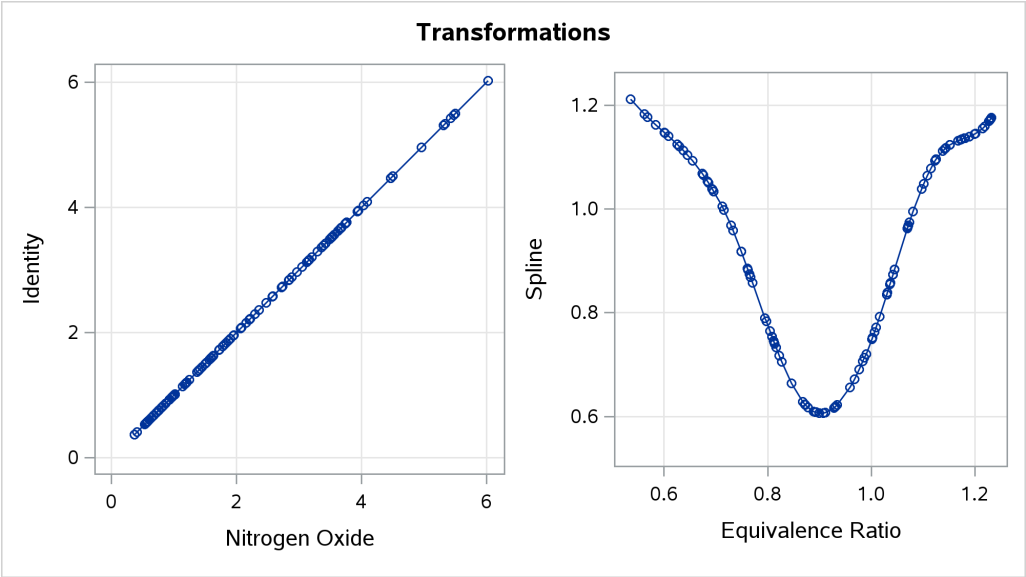

The transformation plots show the identity transformation of NOx and the nonlinear spline transformation of EqRatio. These plots are requested with the PLOTS=TRANSFORMATION option. The plot on the left shows that NOx is unchanged, which is always the case with the IDENTITY transformation. In contrast, the spline transformation of EqRatio is nonlinear. It is this nonlinear transformation of EqRatio that accounts for the increase in fit that is shown in the iteration history table.

Figure 126.3 Residuals

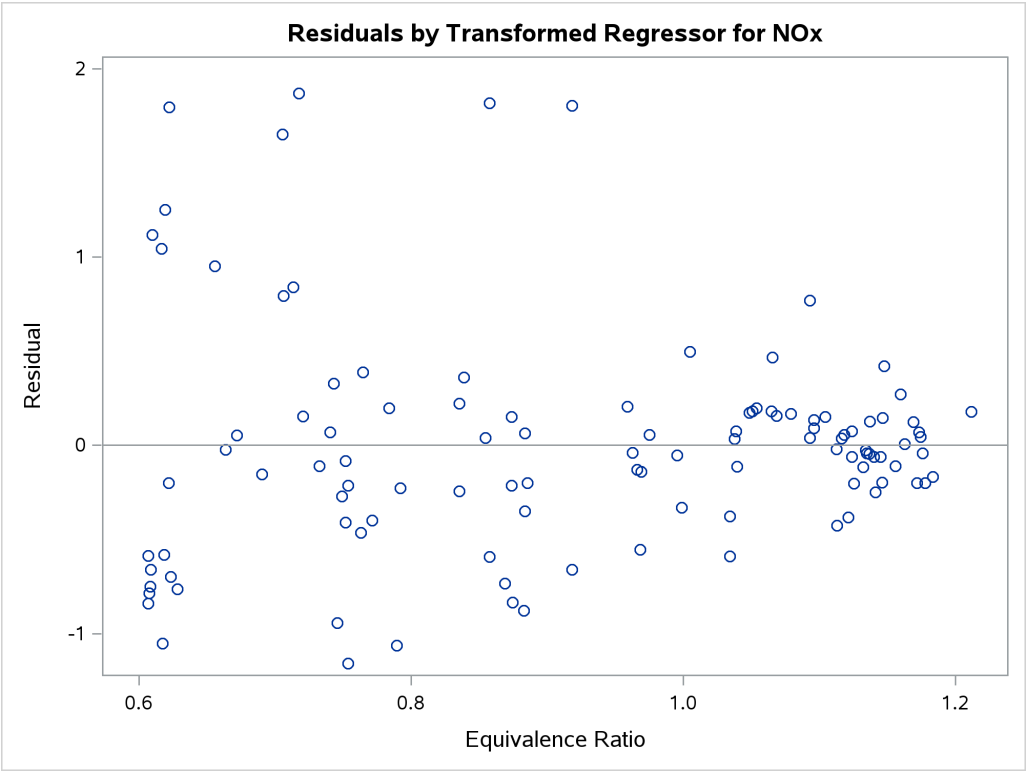

The residuals plot in Figure 126.3 shows the residuals as a function of the transformed independent variable.

The "Spline Regression Fit" plot in Figure 126.4 displays the nonlinear regression function plotted through the original data, along with 95% confidence and prediction limits. This plot clearly shows that nitrous oxide emissions are largest in the middle range of equivalence ratio, 0.08 to 1.0, and are much lower for the extreme values of equivalence ratio, such as around 0.6 and 1.2.

Figure 126.4 Fitting a Curve through a Scatter Plot

This plot is produced by default when ODS Graphics is enabled and when there is an IDENTITY dependent variable and one non-CLASS independent variable. The plot consists of an ordinary scatter plot of NOx plotted as a function of EqRatio. It also contains the predicted values of NOx, which are a function of the spline transformation of EqRatio (or TEqRatio shown previously), and are plotted as a function of EqRatio. Similarly, it contains confidence limits based on NOx and TEqRatio.

The "Observed by Predicted" values plot in Figure 126.5 displays the dependent variable plotted as a function of the regression predicted values along with a linear regression line, which for this plot always has a slope of 1. This plot was requested with the OBP or OBSERVEDBYPREDICTED suboption in the PLOTS= option. The residual differences between the transformed data and the regression line show how well the nonlinearly transformed data fit a linear-regression model. The residuals look mostly random; however, they are larger for larger values of NOx, suggesting that maybe this is not the optimal model. You can also see this by examining the fit of the function through the original scatter plot in Figure 126.4. Near the middle of the function, the residuals are much larger. You can refit the model, this time requesting separate functions for each type of fuel. You can request the original scatter plot, without any regression information and before the variables are transformed, by specifying the SCATTER suboption in the PLOTS= option.

Figure 126.5 Observed by Predicted

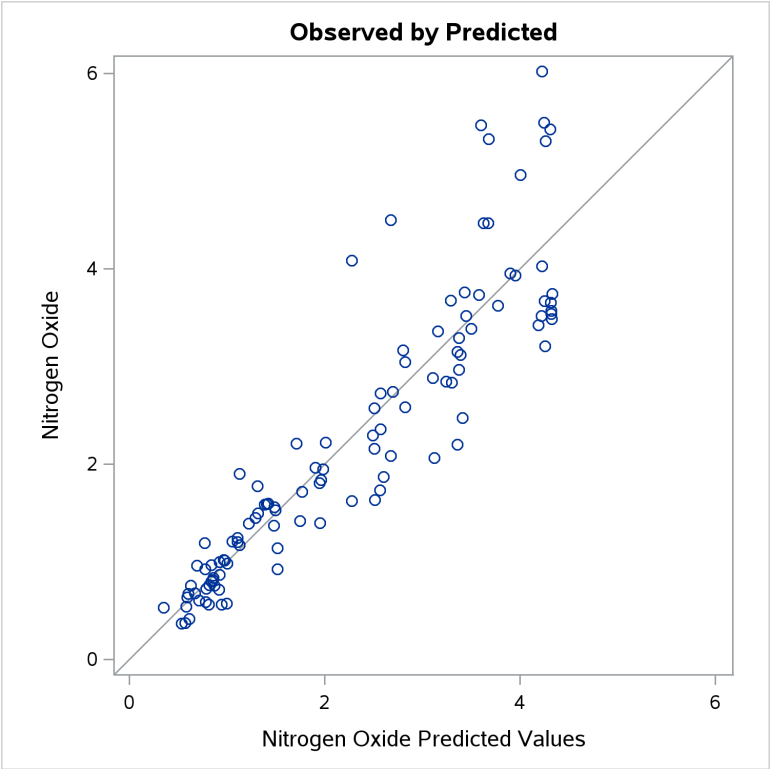

These next statements fit an additive model with separate functions for each of the different fuels. The statements produce Figure 126.6 through Figure 126.9.

```
* Separate Curves and Intercepts;
proc transreg data=Gas solve ss2 additive plots=(transformation obp);
   model identity(nox) = class(Fuel / zero=none) |
                         spline(EqRatio / nknots=4 after);
run;
```

The ADDITIVE *a-option* requests an additive model, where the regression coefficients are absorbed into the transformations, and so the final regression coefficients are all one. The specification CLASS(Fuel / ZERO=NONE) recodes fuel into a set of three binary variables, one for each of the three fuels in this data set. The vertical bar between the CLASS and SPLINE specifications request both main effects and interactions. For this model, it requests both a separate intercept and a separate spline function for each fuel. The original two variables, Fuel and EqRatio, are replaced by six variables—three binary intercept terms and three spline variables. The three spline variables are zero when their corresponding intercept binary variable is zero, and nonzero otherwise. The nonzero parts are optimally transformed by the analysis. The AFTER *t-option* specified with the SPLINE transformation specifies that the four knots should be selected independently for each of the three spline transformations, *after* EqRatio is crossed with the CLASS variable. Alternatively, and by default, the knots are chosen by examining EqRatio before it is crossed with the CLASS variable, and the same knots are used for all three transformations. The results are shown in Figure 126.6.

Figure 126.6 Iteration, ANOVA, and Regression Results

Gasoline and Emissions Data

The TRANSREG Procedure

Dependent Variable Identity(NOx)
Nitrogen Oxide

Class Level Information		
Class	**Levels**	**Values**
Fuel	3	82rongas Ethanol Gasohol

Number of Observations Read	112
Number of Observations Used	110
Implicit Intercept Model	

TRANSREG MORALS Algorithm Iteration History for Identity(NOx)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	0.12476	1.13866	0.18543		
1	0.00000	0.00000	0.95870	0.77327	Converged

Algorithm converged.

Hypothesis Test Iterations Excluding Spline(Fuel82rongasEqRatio)

TRANSREG MORALS Algorithm Iteration History for Identity(NOx)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	0.00000	0.00000	0.80234		
1	0.00000	0.00000	0.80234	-.00000	Converged

Algorithm converged.

Hypothesis Test Iterations Excluding Spline(FuelEthanolEqRatio)

TRANSREG MORALS Algorithm Iteration History for Identity(NOx)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	0.00000	0.00000	0.48801		
1	0.00000	0.00000	0.48801	-.00000	Converged

Algorithm converged.

Figure 126.6 *continued*

Hypothesis Test Iterations Excluding Spline(FuelGasoholEqRatio)

TRANSREG MORALS Algorithm Iteration History for Identity(NOx)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	0.00000	0.00000	0.80052		
1	0.00000	0.00000	0.80052	-.00000	Converged

Algorithm converged.

The TRANSREG Procedure Hypothesis Tests for Identity(NOx) Nitrogen Oxide

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	23	209.4613	9.107012	86.80	<.0001
Error	86	9.0229	0.104918		
Corrected Total	109	218.4842			

Root MSE	0.32391	**R-Square**	0.9587
Dependent Mean	2.25022	**Adj R-Sq**	0.9477
Coeff Var	14.39461		

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Class.Fuel82rongas	1	1.00000000	32.634	32.6338	311.04	<.0001	Fuel 82rongas
Class.FuelEthanol	1	1.00000000	97.406	97.4058	928.40	<.0001	Fuel Ethanol
Class.FuelGasohol	1	1.00000000	34.672	34.6720	330.47	<.0001	Fuel Gasohol
Spline(Fuel82rongasEqRatio)	7	1.00000000	34.162	4.8803	46.52	<.0001	Fuel 82rongas * Equivalence Ratio
Spline(FuelEthanolEqRatio)	7	1.00000000	102.840	14.6914	140.03	<.0001	Fuel Ethanol * Equivalence Ratio
Spline(FuelGasoholEqRatio)	7	1.00000000	34.561	4.9372	47.06	<.0001	Fuel Gasohol * Equivalence Ratio

ZERO=SUM and ZERO=NONE coefficient tests are not exact when there are iterative transformations. Those tests are performed holding all transformations fixed, and so are generally liberal.

The first iteration history table in Figure 126.6 shows that PROC TRANSREG increases the squared multiple correlation from the original value of 0.18543 to 0.95870. The remaining iteration histories pertain to PROC TRANSREG's process of comparing models to test hypotheses. The important thing to look for is convergence in all of the tables.

Figure 126.7 Transformations

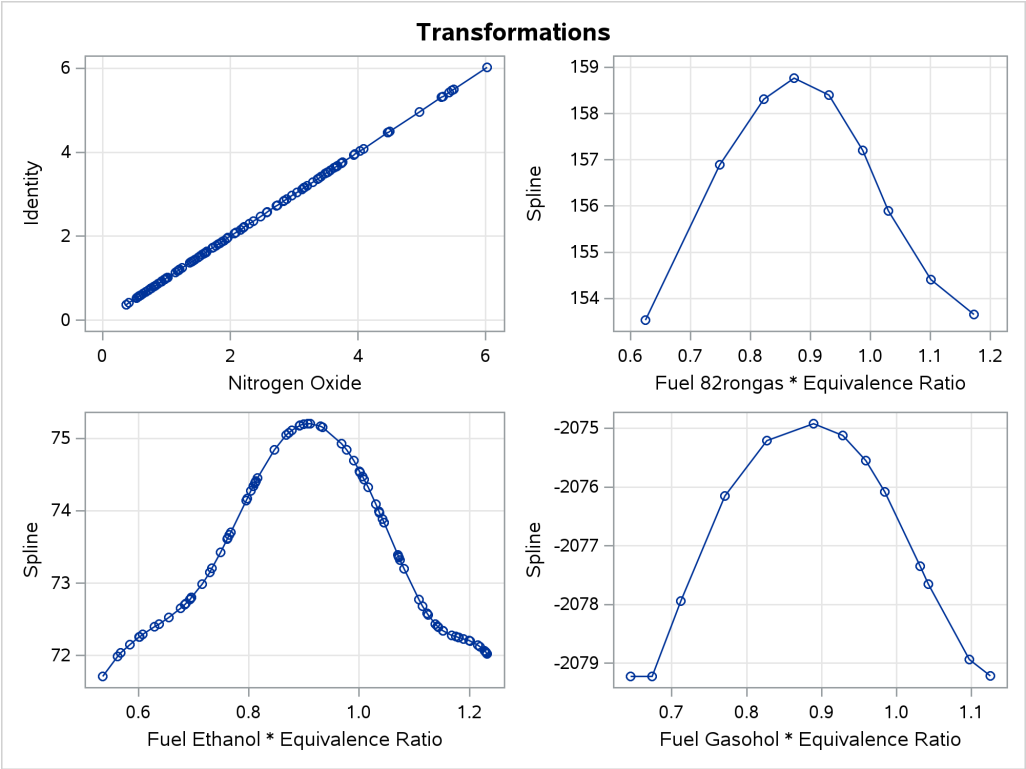

The transformations, shown in Figure 126.7, show that for all three groups, the transformation of EqRatio is approximately quadratic.

Figure 126.8 Fitting Curves through a Scatter Plot

The fit plot, shown in Figure 126.8, shows that there are in fact three distinct functions in the data. The increase in fit over the previous model comes from individually fitting each group instead of providing an aggregate fit.

Figure 126.9 Observed by Predicted

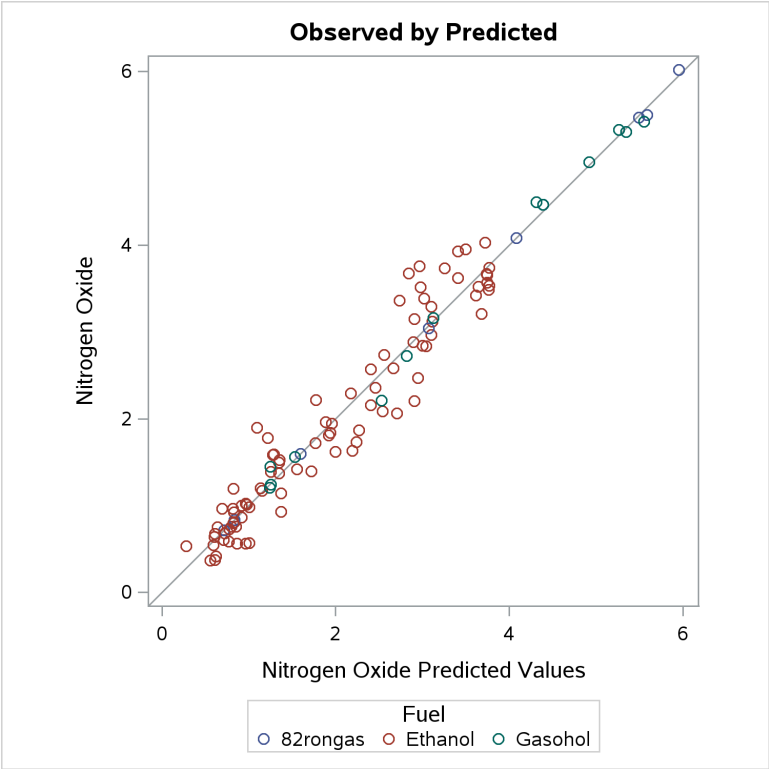

The residuals in the observed by predicted plot displayed in Figure 126.9 are much better for this analysis.

You could fit a model that is "in between" the two models shown previously. This next model provides for separate intercepts for each group, but calls for a common function. There are still three functions, one per group, but their shapes are the same, and they are equidistant or parallel. This model is requested by omitting the vertical bar so that separate intercepts are requested, but not separate curves within each group. The following statements fit the separate intercepts model and create Figure 126.10:

```
* Separate Intercepts;
proc transreg data=Gas solve ss2 additive;
   model identity(nox) = class(Fuel / zero=none)
                         spline(EqRatio / nknots=4);
run;
```

The ANOVA table and fit plot are shown in Figure 126.10.

Figure 126.10 Separate Intercepts Only

Gasoline and Emissions Data

The TRANSREG Procedure

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	9	196.7548	21.86165	100.61	<.0001
Error	100	21.7294	0.21729		
Corrected Total	109	218.4842			

Figure 126.10 *continued*

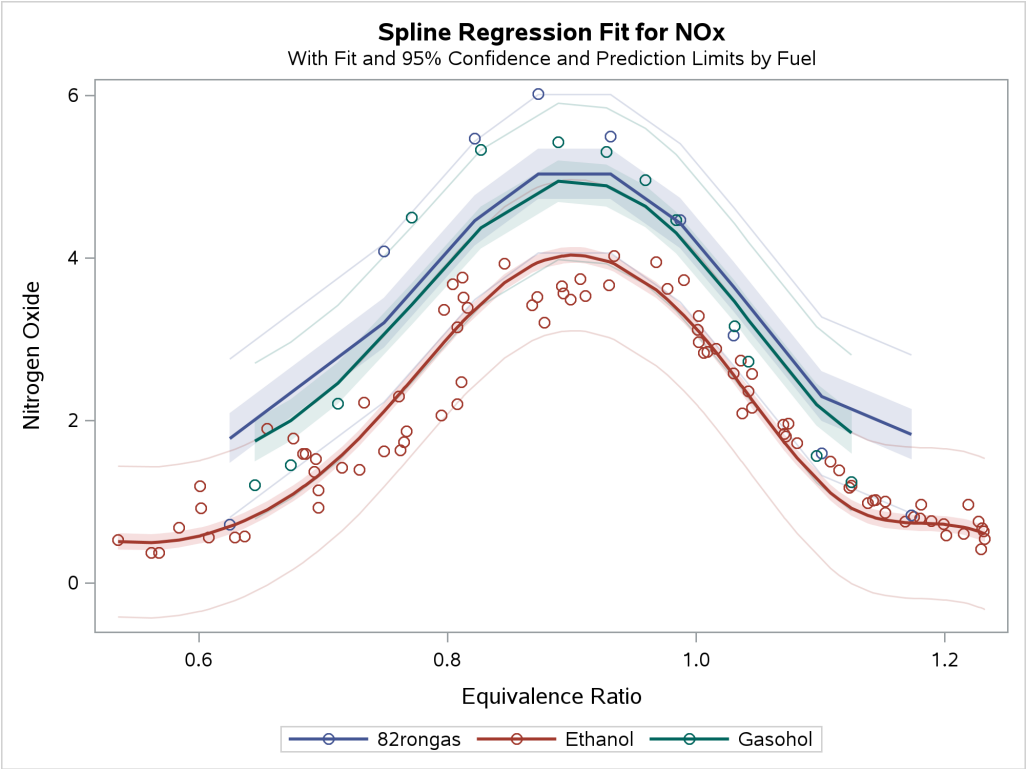

Now, squared multiple correlation is 0.9005, which is smaller than the model with the unconstrained separate curves, but larger than the model with only one curve. Because of the restrictions on the shapes, these curves do not track the data as well as the previous model. However, this model is more parsimonious with many fewer parameters.

There are other ways to fit curves through scatter plots in PROC TRANSREG. For example, you could use smoothing splines or penalized B-splines, as is illustrated next. The following statements fit separate curves through each group by using penalized B-splines and produce Figure 126.11:

```
* Separate Curves and Intercepts with Penalized B-Splines;
proc transreg data=Gas ss2 plots=transformation lprefix=0;
   model identity(nox) = class(Fuel / zero=none) * pbspline(EqRatio);
run;
```

This example asks for a separate penalized B-spline transformation, PBSPLINE, of equivalence ratio for each type of fuel. The LPREFIX=0 *a-option* is specified in the PROC statement so that zero characters of the CLASS variable name (Fuel) are used in constructing the labels for the coded variables. The result is label components like "Ethanol" instead of the more redundant "Fuel Ethanol". The results of this analysis are shown in Figure 126.11.

Figure 126.11 Penalized B-Splines

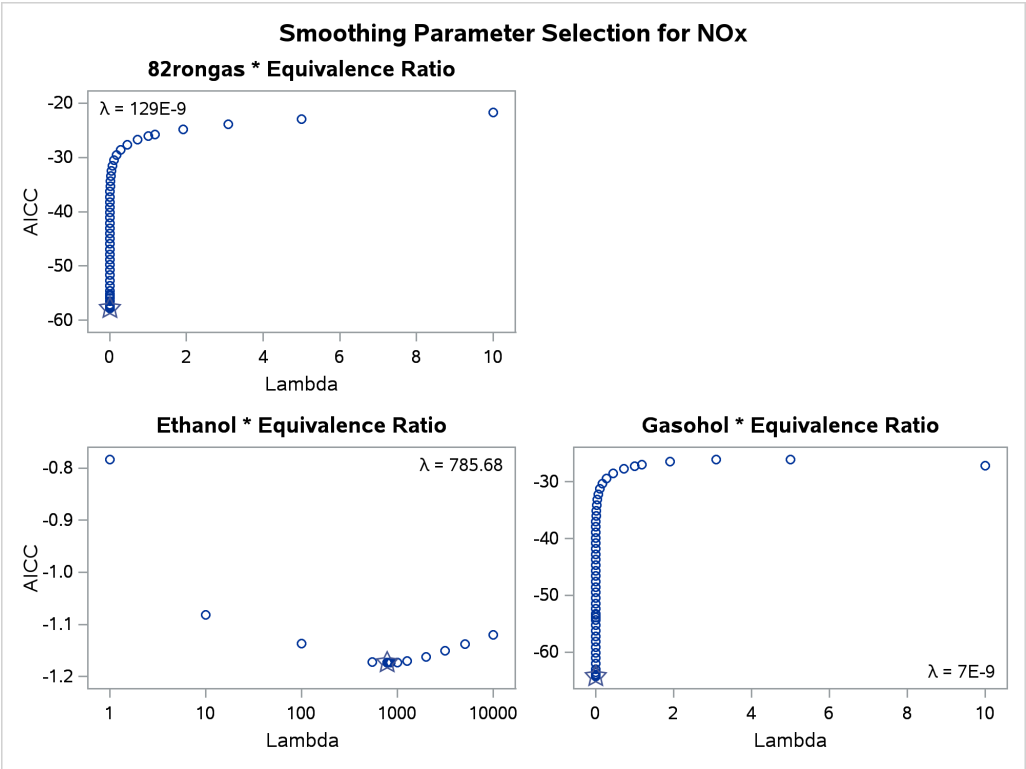

Figure 126.11 *continued*

Dependent Variable Identity(NOx)
Nitrogen Oxide

Class Level Information		
Class	**Levels**	**Values**
Fuel	3	82rongas Ethanol Gasohol

Number of Observations Read	112
Number of Observations Used	110
Implicit Intercept Model	

Figure 126.11 *continued*

TRANSREG Univariate Algorithm Iteration History for Identity(NOx)

Iteration Number	Average Change	Maximum Change	Note
1	0.00000	0.00000	Converged

Algorithm converged.

The TRANSREG Procedure Hypothesis Tests for Identity(NOx) Nitrogen Oxide

Univariate ANOVA Table, Penalized B-Spline Transformation

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	33.194	211.4818	6.371106	68.97	<.0001
Error	75.806	7.0024	0.092373		
Corrected Total	109	218.4842			

Root MSE	0.30393	**R-Square**	0.9680
Dependent Mean	2.25022	**Adj R-Sq**	0.9539
Coeff Var	13.50663		

Penalized B-Spline Transformation

Variable	DF	Coefficient	Lambda	AICC	Label
Pbspline(Fuel82rongasEqRatio)	9.000	1.000	1.287E-7	-57.7840	82rongas * Equivalence Ratio
Pbspline(FuelEthanolEqRatio)	12.19	1.000	785.7	-1.1736	Ethanol * Equivalence Ratio
Pbspline(FuelGasoholEqRatio)	13.00	1.000	7.019E-9	-64.2961	Gasohol * Equivalence Ratio

Figure 126.11 *continued*

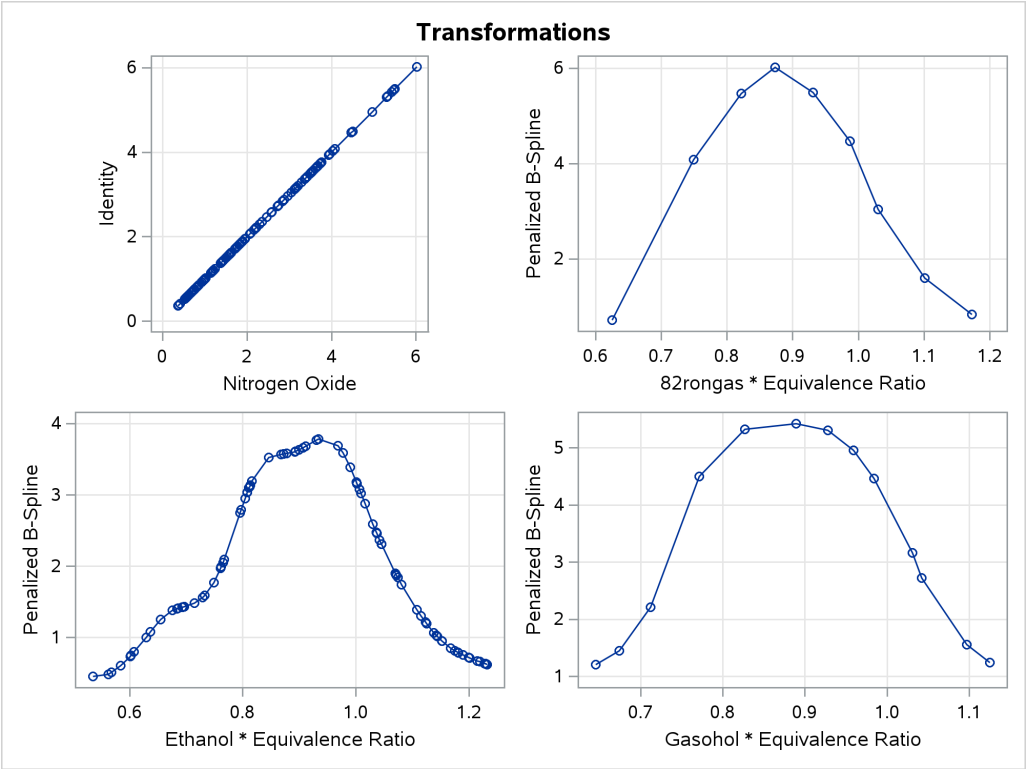

Figure 126.11 *continued*

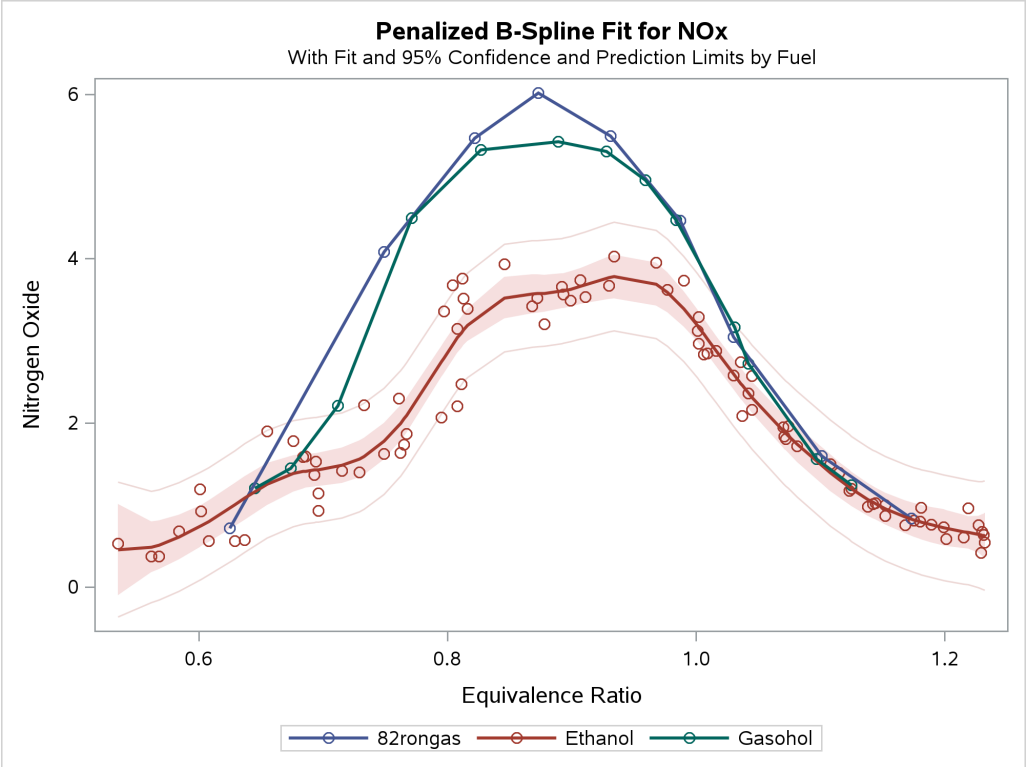

With penalized B-splines, the degrees of freedom are based on the trace of the transformation hat matrix and are typically not integers. The first panel of plots shows AICC as a function of lambda, the smoothing parameter. The smoothing parameter is automatically chosen, and since the smoothing parameters range from essentially 0 to almost 800, it is clear that some functions are smoother than others. The plots of the criterion (AICC in this example) as a function of lambda use a linear scale for the horizontal axis when the range of lambdas is small, as in the first and third plot, and a log scale when the range is large, as in the second plot. The transformation for equivalence ratio for Ethanol required more smoothing than for the other two fuels. All three have an overall quadratic shape, but for Ethanol, the function more closely follows the smaller variations in the data. You could get similar results with SPLINE by using more knots.

For other examples of curve fitting by using PROC TRANSREG, see the sections "Smoothing Splines" on page 10660, "Linear and Nonlinear Regression Functions" on page 10647, "Simultaneously Fitting Two Regression Functions" on page 10651, and "Using Splines and Knots" on page 10632, as well as Example 126.3. These examples include cases where multiple curves are fit through scatter plots with multiple groups. Special cases include linear models with separate slopes and separate intercepts. Many constraints on the slopes, curves, and intercepts are possible.

Main-Effects ANOVA

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/treggs2.sas.)

This example shows how to use PROC TRANSREG to code and fit a main-effects ANOVA model. PROC TRANSREG has very extensive and versatile options for coding or creating so-called dummy variables. PROC TRANSREG is commonly used to code classification variables before they are used for analysis in other procedures. See the sections "Using the DESIGN Output Option" on page 10722 and "Discrete Choice Experiments: DESIGN, NORESTORE, NOZERO" on page 10726. In this example, the input data set contains the dependent variables y, factors x1 and x2, and 12 observations. PROC TRANSREG can be useful for coding even before running procedures with a CLASS statement because of its detailed options that enable you to control how the coded variable names and labels are constructed. The following statements perform a main-effects ANOVA and display the results in Figure 126.12 and Figure 126.13:

```
title 'Introductory Main-Effects ANOVA Example';

data a;
   input y x1 $ x2 $;
   datalines;
8 a a
7 a a
4 a b
3 a b
5 b a
4 b a
2 b b
1 b b
8 c a
7 c a
5 c b
2 c b
;
```

```
* Fit a main-effects ANOVA model with 1, 0, -1 coding;
proc transreg ss2;
   model identity(y) = class(x1 x2 / effects);
   output coefficients replace;
run;

* Display TRANSREG output data set;
proc print label;
   format intercept -- x2a 5.2;
run;
```

The SS2 *a-option* requests results based on Type II sums of squares. The simple ANOVA model is fit by designating y as an IDENTITY variable, which specifies no transformation. The independent variables are specified with a CLASS expansion, which replaces them with coded variables. There are $(3-1)+(2-1) = 3$ coded variables created by the CLASS specification, since the two CLASS variables have 3 and 2 different values or levels. In this case, the EFFECTS *t-option* is specified. This option requests an *effects coding* (displayed in Figure 126.13), which is also called a deviations from means or 0, 1, –1 coding. The OUTPUT statement requests an output data set with the data and coded variables. The COEFFICIENTS output option, or *o-option*, adds the parameter estimates and marginal means to the data set. The REPLACE *o-option* specifies that the transformed variables should replace the original variables in the output data set. The output data set variable names are the same as the original variable name. In an example like this, there are no nonlinear transformations; the transformed variables are the same as the original variables. The REPLACE *o-option* is used to eliminate unnecessary and redundant transformed variables from the output data set. The results of the PROC TRANSREG step are shown in Figure 126.12.

Figure 126.12 ANOVA Example Output from PROC TRANSREG

Introductory Main-Effects ANOVA Example

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	**Levels**	**Values**
x1	3	a b c
x2	2	a b

Number of Observations Read	12
Number of Observations Used	12

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	3	57.00000	19.00000	19.83	0.0005
Error	8	7.66667	0.95833		
Corrected Total	11	64.66667			

Figure 126.12 *continued*

Root MSE	0.97895	**R-Square**	0.8814
Dependent Mean	4.66667	**Adj R-Sq**	0.8370
Coeff Var	20.97739		

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Intercept	1	4.6666667	261.333	261.333	272.70	<.0001	Intercept
Class.x1a	1	0.8333333	4.167	4.167	4.35	0.0705	x1 a
Class.x1b	1	-1.6666667	16.667	16.667	17.39	0.0031	x1 b
Class.x2a	1	1.8333333	40.333	40.333	42.09	0.0002	x2 a

Figure 126.12 shows the ANOVA results, fit statistics, and regression tables. The output data set, with the coded design, parameter estimates and means, is shown in Figure 126.13. For more information about PROC TRANSREG for ANOVA and other codings, see the section "ANOVA Codings" on page 10664.

Figure 126.13 Output Data Set from PROC TRANSREG

Introductory Main-Effects ANOVA Example

Obs	_TYPE_	_NAME_	y	Intercept	x1 a	x1 b	x2 a	x1	x2
1	SCORE	ROW1	8	1.00	1.00	0.00	1.00	a	a
2	SCORE	ROW2	7	1.00	1.00	0.00	1.00	a	a
3	SCORE	ROW3	4	1.00	1.00	0.00	-1.00	a	b
4	SCORE	ROW4	3	1.00	1.00	0.00	-1.00	a	b
5	SCORE	ROW5	5	1.00	0.00	1.00	1.00	b	a
6	SCORE	ROW6	4	1.00	0.00	1.00	1.00	b	a
7	SCORE	ROW7	2	1.00	0.00	1.00	-1.00	b	b
8	SCORE	ROW8	1	1.00	0.00	1.00	-1.00	b	b
9	SCORE	ROW9	8	1.00	-1.00	-1.00	1.00	c	a
10	SCORE	ROW10	7	1.00	-1.00	-1.00	1.00	c	a
11	SCORE	ROW11	5	1.00	-1.00	-1.00	-1.00	c	b
12	SCORE	ROW12	2	1.00	-1.00	-1.00	-1.00	c	b
13	M COEFFI	y	.	4.67	0.83	-1.67	1.83		
14	MEAN	y	.	.	5.50	3.00	6.50		

The output data set has three kinds of observations, identified by values of _TYPE_ as follows:

- When _TYPE_='SCORE', the observation contains the following information about the dependent and independent variables:
  - y is the original dependent variable.
  - x1 and x2 are the independent classification variables, and the Intercept through x2 a columns contain the main-effects design matrix that PROC TRANSREG creates. The variable names are Intercept, x1a, x1b, and x2a. Their labels are shown in the listing.

- When _TYPE_='M COEFFI', the observation contains coefficients of the final linear model (parameter estimates).
- When _TYPE_='MEAN', the observation contains the marginal means.

The observations with _TYPE_='SCORE' form the score or data partition of the output data set, and the observations with _TYPE_='M COEFFI' and _TYPE_='MEAN' form the output statistics partition of the output data set.

Syntax: TRANSREG Procedure

The following statements are available in the TRANSREG procedure:

```
PROC TRANSREG <DATA=SAS-data-set>
              <PLOTS=(plot-requests)>
              <OUTTEST=SAS-data-set> <a-options> <o-options> ;
   MODEL <transform(dependents <|t-options>)>
         <transform(dependents <|t-options>)...=>
           transform(independents <|t-options>)
         <transform(independents <|t-options>)...> <|a-options> ;
   OUTPUT <OUT=SAS-data-set> <o-options> ;
   ID variables ;
   FREQ variable ;
   WEIGHT variable ;
   BY variables ;
```

To use PROC TRANSREG, you need both the PROC TRANSREG and MODEL statements. To produce an OUT= output data set, the OUTPUT statement is required. PROC TRANSREG enables you to specify the same options in more than one statement. All of the MODEL statement *a-options* (algorithm options) and all of the OUTPUT statement *o-options* (output options) can also be specified in the PROC TRANSREG statement. You can abbreviate all *a-options*, *o-options*, and *t-options* (transformation options) to their first three letters. This is a special feature of PROC TRANSREG and is not generally true of other SAS/STAT procedures. See Table 126.1 for a list of options available in the PROC TRANSREG statement.

The PROC TRANSREG statement starts the TRANSREG procedure. Optionally, this statement identifies an input and an OUTTEST= data set, specifies the algorithm and other computational details, requests displayed output, and controls the contents of the OUT= data set (which is created with the OUTPUT statement). The DATA= and OUTTEST= options can appear only in the PROC TRANSREG statement. All *a-options* and *o-options* are described in the sections on either the MODEL or OUTPUT statement, in which these options can also be specified.

The rest of this section provides detailed syntax information for each of the preceding statements, beginning with the PROC TRANSREG statement. The remaining statements are described in alphabetical order.

PROC TRANSREG Statement

PROC TRANSREG < **DATA=***SAS-data-set* >
< **PLOTS=(***plot-requests***)** >
< **OUTTEST=***SAS-data-set* > < *a-options* > < *o-options* > **;**

The PROC TRANSREG statement invokes the TRANSREG procedure. Optionally, this statement identifies an input and an OUTTEST= data set, specifies the algorithm and other computational details, requests displayed output, and controls the contents of the OUT= data set (which is created with the OUTPUT statement). The DATA=, OUTTEST=, and PLOTS= options can appear only in the PROC TRANSREG statement. Table 126.1 summarizes the options available in the PROC TRANSREG statement. The *a-options* are also available in the MODEL statement, and the *o-options* are also available in the OUTPUT statement.

Table 126.1 Options Available in the PROC TRANSREG Statement

Option	Description
Data Set Options (PROC Statement)	
DATA=	Specifies input SAS data set
OUTTEST=	Specifies output test statistics data set
ODS Graphics (PROC Statement)	
PLOTS=	Specifies ODS Graphics selection
Input Control (PROC or MODEL)	
REITERATE	Restarts the iterations
TYPE=	Specifies input observation type
Method and Iterations (PROC or MODEL)	
CCONVERGE=	Specifies minimum criterion change
CONVERGE=	Specifies minimum data change
MAXITER=	Specifies maximum number of iterations
METHOD=	Specifies iterative algorithm
NCAN=	Specifies number of canonical variables
NSR	Specifies no restrictions on smoothing models
SINGULAR=	Specifies singularity criterion
SOLVE	Attempts direct solution instead of iteration
Missing Data Handling (PROC or MODEL)	
INDIVIDUAL	Fits each model individually (METHOD=MORALS)
MONOTONE=	Includes monotone special missing values
NOMISS	Excludes observations with missing values
UNTIE=	Unties special missing values
Intercept and CLASS Variables (PROC or MODEL)	
CPREFIX=	Specifies CLASS coded variable name prefix
LPREFIX=	Specifies CLASS coded variable label prefix
NOINT	Specifies no intercept or centering
ORDER=	Specifies order of CLASS variable levels
REFERENCE=	Controls output of reference levels

Table 126.1 *continued*

Option	Description
SEPARATORS=	Controls CLASS coded variable label separators
Control Displayed Output (PROC or MODEL)	
ALPHA=	Specifies confidence limits alpha
CL	Displays parameter estimate confidence limits
DETAIL	Displays model specification details
HISTORY	Displays iteration histories
NOPRINT	Suppresses displayed output
PBOXCOXTABLE	Prints the Box-Cox log likelihood table
RSQUARE	Displays the R square
SHORT	Suppresses the iteration histories
SS2	Displays regression results
TEST	Displays ANOVA table
TSUFFIX=	Shortens transformed variable labels
UTILITIES	Displays conjoint part-worth utilities
Standardization (PROC or MODEL)	
ADDITIVE	Fits additive model
NOZEROCONSTANT	Does not zero constant variables
TSTANDARD=	Specifies transformation standardization
Predicted Values, Residuals, Scores (PROC or OUTPUT)	
CANONICAL	Outputs canonical scores
CLI	Outputs individual confidence limits
CLM	Outputs mean confidence limits
DESIGN=	Specifies design matrix coding
DREPLACE	Replaces dependent variables
IREPLACE	Replaces independent variables
LEVERAGE	Outputs leverage
NORESTOREMISSING	Does not restore missing values
NOSCORES	Suppresses output of scores
PREDICTED	Outputs predicted values
REDUNDANCY=	Outputs redundancy variables
REPLACE	Replaces all variables
RESIDUALS	Outputs residuals
Output Data Set Coefficients (PROC or OUTPUT)	
COEFFICIENTS	Outputs coefficients
COORDINATES=	Outputs ideal point coordinates
MEANS	Outputs marginal means
MREDUNDANCY	Outputs redundancy analysis coefficients
Output Data Set Variable Name Prefixes (PROC or OUTPUT)	
ADPREFIX=	Specifies dependent variable approximations
AIPREFIX=	Specifies independent variable approximations
CDPREFIX=	Specifies canonical dependent variables

Table 126.1 *continued*

Option	Description
CILPREFIX=	Specifies conservative individual lower CL
CIPREFIX=	Specifies canonical independent variables
CIUPREFIX=	Specifies conservative-individual-upper CL
CMLPREFIX=	Specifies conservative-mean-lower CL
CMUPREFIX=	Specifies conservative-mean-upper CL
DEPENDENT=	Specifies METHOD=MORALS untransformed dependent
LILPREFIX=	Specifies liberal-individual-lower CL
LIUPREFIX=	Specifies liberal-individual-upper CL
LMLPREFIX=	Specifies liberal-mean-lower CL
LMUPREFIX=	Specifies liberal-mean-upper CL
PPREFIX=	Specifies predicted values
RDPREFIX=	Specifies residuals
RPREFIX=	Specifies redundancy variables
TDPREFIX=	Specifies transformed dependents
TIPREFIX=	Specifies transformed independents
Macros Variables (PROC or OUTPUT)	
MACRO	Creates macro variables
Other Options (PROC or OUTPUT)	
APPROXIMATIONS	Outputs dependent and independent approximations
CCC	Outputs canonical correlation coefficients
CEC	Outputs canonical elliptical point coordinates
CPC	Outputs canonical point coordinates
CQC	Outputs canonical quadratic point coordinates
DAPPROXIMATIONS	Outputs approximations to transformed dependents
IAPPROXIMATIONS	Outputs approximations to transformed independents
MEC	Outputs elliptical point coordinates
MPC	Outputs point coordinates
MQC	Outputs quadratic point coordinates
MRC	Outputs multiple regression coefficients

DATA=*SAS-data-set*

specifies the SAS data set to be analyzed. If you do not specify the DATA= option, PROC TRANSREG uses the most recently created SAS data set. The data set must be an ordinary SAS data set; it cannot be a special TYPE= data set.

OUTTEST=*SAS-data-set*

specifies an output data set to contain hypothesis tests results. When you specify the OUTTEST= option, the data set contains ANOVA results. When you specify the SS2 *a-option*, regression tables are also output. When you specify the UTILITIES *o-option*, conjoint analysis part-worth utilities are also output. For more information about the OUTTEST= data set, see the section "OUTTEST= Output Data Set" on page 10706.

PLOTS < (*global-plot-options*) > < = *plot-request* < (*options*) > >

PLOTS < (*global-plot-options*) > < = (*plot-request* < (*options*) > < . . . *plot-request* < (*options*) > >) >

controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=none
plots=(residuals transformation)
plots(unpack)=boxcox
plots(unpack)=(transformation boxcox(p=0))
plots=(residuals(unpack) transformation(dep unp) boxcox(t rmse))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc transreg plots=all;
   model identity(y) = pbspline(x);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled, but you do not specify the PLOTS= option, then PROC TRANSREG produces a default set of plots. The fit, scatter, residual, and observed-by-predicted plots are available with METHOD=MORALS and also with METHOD=UNIVARIATE when there is only one dependent variable. When no method is specified and there is more than one dependent variable, and when regression plots are requested, the default method is set to METHOD=MORALS. When there is more than one dependent variable, when METHOD= is not specified, or when METHOD=MORALS is specified and PLOTS=ALL is specified, the plots that are produced might be different from those you would see with METHOD=UNIVARIATE and PLOTS=ALL. Certain plots appear by default when ODS Graphics is enabled and certain combinations of options are specified. The Box-Cox $F = t^2$ and log-likelihood plots appear when a BOXCOX dependent variable transformation is specified. The regression fit plot appears for models with a single dependent variable that is not transformed (for example, IDENTITY(y)), a single quantitative independent variable that might or might not be transformed, and at most one CLASS independent variable. Preference mapping plots appear when the COORDINATES *o-option* is used.

The global plot options include the following:

INTERPOLATE

INT

uses observations that are excluded from the analysis for interpolation in the fit and transformation plots. By default, observations with zero weight are excluded from all plots. These include observations with a zero, negative, or missing weight or frequency and observations excluded due to missing and invalid values. You can specify PLOTS(INTERPOLATE)=(*plot-requests*) to include some of these observations in the plots. You might want to use this option, for example, with sparse data sets to show smoother functions over the range of the data (see the section "The

PLOTS(INTERPOLATE) Option" on page 10731). Observations with missing values in CLASS variables are excluded from the plots even when PLOTS(INTERPOLATE) is specified.

ONLY
ONL
suppresses the default plots. Only plots specifically requested are displayed.

UNPACKPANEL
UNPACK
UNP
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACKPANEL to get each plot in a separate panel. You can specify PLOTS(UNPACKPANEL) to unpack the default plots. You can also specify UNPACKPANEL as a suboption with TRANSFORMATION, RESIDUALS, PBSPLINE, and BOXCOX.

The plot requests include the following:

ALL
produces all appropriate plots. You can specify other options with ALL; for example, to request all plots and unpack only the residuals, specify PLOTS=(ALL RES(UNP)).

BOXCOX < (*options*) >
BOX < (*options*) >
requests a display of the results of the Box-Cox transformation. These results are displayed by default when there is a Box-Cox transformation. The BOXCOX plot request has the following options:

P=*n*
adds t or $F = t^2$ curves to the legend for the functions where $p(t) < n$, where t is the t statistic corresponding to the optimal lambda. You can specify P=0 to suppress the legend and P=1 to see all curves in the legend. The default value comes from the BOXCOX(variable / ALPHA=p) specification, which by default is 0.05.

RMSE
RMS
plots the root mean square error as a function of lambda.

T
plots t statistics rather than $F = t^2$ statistics.

UNPACKPANEL
UNPACK
UNP
plots the t or $F = t^2$ and log-likelihood plots in separate panels.

FIT < (*options*) >

requests a regression fit plot. This plot is produced by default whenever it is appropriate. It is produced when the dependent variable is specified with the IDENTITY *transform*, and when there is one quantitative independent variable (for example, IDENTITY for linear fit, SPLINE or one of the other transformations for a nonlinear fit, or PSPLINE) and at most one CLASS variable. When there is a CLASS variable, separate fits are produced within levels based on your model. You would specify the FIT plot request only to specify a FIT option or with the ONLY global plot option. The FIT plot request has the following options:

FORMULA

FOR

displays the fit function as an equation in regression fit plots. This option is valid when a fit plot is produced and either an IDENTITY *transform* or a PSPLINE *expansion* with degree less than ten and no knots is specified for a single independent variable. When this option is specified, you can output the formula to a data set by using the **`ods output formula=`***SAS-data-set* statement. This is the formula, complete with Unicode specifications for polynomials, that is used in the fit plot template to make the formula.

NOCLM

suppresses the confidence limits in regression fit plots.

NOCLI

suppresses the individual prediction limits in regression fit plots.

NOOBS

suppresses the observations showing only the fit function and optionally the confidence and prediction limits.

NONE

suppresses all plots.

OBSERVEDBYPREDICTED

OBP

OBS

plots the transformed dependent variable as a function of the regression predicted values.

PBSPLINE < (UNPACKPANEL) >

PBS < (UNPACK) >

requests the penalized B-spline criterion plots. You would specify the PBSPLINE plot request only to specify a PBSPLINE option or with the ONLY global plot option. The PBSPLINE plot request has the following option:

UNPACKPANEL

UNPACK

UNP

plots each criterion plot in a separate panel.

PREFMAP

PRE

plots ideal point or vector preference mapping results when either two IDENTITY or two POINT independent variables are specified along with the COORDINATES option.

RESIDUALS < (*options*) >

RES < (*options*) >

plots the residuals as a function of each of the transformed independent variables, except coded CLASS variables. The RESIDUALS plot request has the following options:

CLASS

CLA

plots the residuals as a function of each of the transformed independent variables, including coded CLASS variables. Note that the ALL plot request, which you use to request all plots, specifies the RESIDUALS plot request without the CLASS option.

UNPACKPANEL

UNPACK

UNP

plots the residuals in separate plots, not several per panel.

SMOOTH

SMO

adds a LOESS smooth function to the residuals plots.

SCATTER

SCA

plots the scatter plot of observed data, before the transformations, for models with a single quantitative dependent variable, a single quantitative independent variable, and at most one CLASS independent variable.

TRANSFORMATION < (*options*) >

TRA < (*options*) >

plots the variable transformations. The TRANSFORMATION plot request has the following options:

DEPENDENTS

DEP

plots only the dependent variable transformations.

INDEPENDENTS

IND

plots only the independent variable transformations.

UNPACKPANEL
UNPACK
UNP
plots the transformations in separate plots, not several per panel.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC TRANSREG to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the TRANSREG procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If one variable in the input data set represents the frequency of occurrence for other values in the observation, specify the variable's name in a FREQ statement. PROC TRANSREG then treats the data set as if each observation appeared n times, where n is the value of the FREQ variable for the observation. Noninteger values of the FREQ variable are truncated to the largest integer less than the FREQ value. The observation is used in the analysis only if the value of the FREQ statement variable is greater than or equal to 1.

ID Statement

ID *variables* **;**

The ID statement includes additional character or numeric variables in the OUT= data set. The variables must be contained in the input data set. The first variable is used to label points in PREFMAP plots. These variables are also used in some plots as tip variables.

MODEL Statement

MODEL < *transform*(*dependents* < / *t-options* >) >
< *transform*(*dependents* < / *t-options* >)**. . . = >**
transform(*independents* < / *t-options* >)
< *transform*(*independents* < / *t-options* >)**. . .** > < / *a-options* > **;**

The MODEL statement specifies the dependent and independent variables (*dependents* and *independents*, respectively) and specifies the transformation (*transform*) to apply to each variable. Only one MODEL statement can appear in PROC TRANSREG. The *t-options* are transformation options, and the *a-options* are algorithm options. The *t-options* provide details for the transformation; these depend on the *transform* chosen. The *t-options* are listed after a slash in the parentheses that enclose the variable list (either *dependents* or *independents*). The *a-options* control the algorithm used, details of iteration, details of how the intercept and coded variables are generated, and displayed output details. The *a-options* are listed after the entire model specification (the *dependents*, *independents*, transformations, and *t-options*) and after a slash. You can also specify the algorithm options in the PROC TRANSREG statement. When you specify the DESIGN *o-option*, *dependents* and an equal sign are not required. The operators *, |, and @ from the GLM procedure are available for interactions with the CLASS expansion and the IDENTITY transformation. They are used as follows:

```
   Class(a * b ...
         c | d ...
         e | f ... @ n)
Identity(a * b ...
         c | d ...
         e | f ... @ n)
```

In addition, transformations and spline expansions can be crossed with classification variables as follows:

transform **(var) * class(group)**
transform **(var) | class(group)**

See the section "Types of Effects" on page 4207 in Chapter 53, "The GLM Procedure," for a description of the @, *, and | operators and see the section "Model Statement Usage" on page 10620 for information about how to use these operators in PROC TRANSREG. Note that nesting is not implemented in PROC TRANSREG.

The next three sections discuss the transformations available (*transforms*) (see the section "Families of Transformations" on page 10583), the transformation options (*t-options*) (see the section "Transformation Options (t-options)" on page 10590), and the algorithm options (*a-options*) (see the section "Algorithm Options (a-options)" on page 10600).

Families of Transformations

In the MODEL statement, *transform* specifies a transformation in one of the following five families:

Variable expansions	preprocess the specified variables, replacing them with more variables.
Nonoptimal transformations	preprocess the specified variables, replacing each one with a single new nonoptimal, nonlinear transformation.
Nonlinear fit transformations	preprocess the specified variable, replacing it with a smooth transformation, fitting one or more nonlinear functions through a scatter plot.
Optimal transformations	replace the specified variables with new, iteratively derived optimal transformation variables that fit the specified model better than the original variable (except for contrived cases where the transformation fits the model exactly as well as the original variable).
Other transformations	are the IDENTITY and SSPLINE transformations. These do not fit into the preceding categories.

The transformations and expansions listed in Table 126.2 are available in the MODEL statement.

Table 126.2 Transformation Families

Transformation	Description
Variable Expansions	
BSPLINE	B-spline basis
CLASS	set of coded variables
EPOINT	elliptical response surface
POINT	circular response surface & PREFMAP
PSPLINE	piecewise polynomial basis
QPOINT	quadratic response surface
Nonoptimal Transformations	
ARSIN	inverse trigonometric sine
EXP	exponential
LOG	logarithm
LOGIT	logit
POWER	raises variables to specified power
RANK	transforms to ranks
Nonlinear Fit Transformations	
BOXCOX	Box-Cox
PBSPLINE	penalized B-splines
SMOOTH	noniterative smoothing spline
Optimal Transformations	
LINEAR	linear
MONOTONE	monotonic, ties preserved
MSPLINE	monotonic B-spline
OPSCORE	optimal scoring
SPLINE	B-spline

Table 126.2 *continued*

Transformation	Description
UNTIE	monotonic, ties not preserved
Other Transformations	
IDENTITY	identity, no transformation
SSPLINE	iterative smoothing spline

You can use any transformation with either dependent or independent variables (except the SMOOTH and PBSPLINE transformations, which can be used only with independent variables, and BOXCOX, which can be used only with dependent variables). However, the variable expansions are usually more appropriate for independent variables.

The *transform* is followed by a variable (or list of variables) enclosed in parentheses. Here is an example:

```
model log(y) = class(x);
```

This example finds a LOG transformation of y and performs a CLASS expansion of x. Optionally, depending on the *transform*, the parentheses can also contain *t-options*, which follow the variables and a slash. Here is an example:

```
model identity(y) = spline(x1 x2 / nknots=3);
```

The preceding statement finds SPLINE transformations of x1 and x2. The NKNOTS= *t-option* used with the SPLINE transformation specifies three knots. The **identity(y)** transformation specifies that y is not to be transformed.

The rest of this section provides syntax details for members of the five families of transformations listed at the beginning of this section. The *t-options* are discussed in the section "Transformation Options (t-options)" on page 10590.

Variable Expansions

PROC TRANSREG performs variable expansions before iteration begins. Variable expansions expand the original variables into a typically larger set of new variables. The original variables are those that are listed in parentheses after *transform*, and they are sometimes referred to by the name of the *transform*. For example, in CLASS(x1 x2), x1 and x2 are sometimes referred to as CLASS expansion variables or simply CLASS variables, and the expanded variables are referred to as coded or sometimes "dummy" variables. Similarly, in POINT(Dim1 Dim2), Dim1 and Dim2 are sometimes referred to as POINT variables.

The resulting variables are not transformed by the iterative algorithms after the initial preprocessing. Observations with missing values for these types of variables are excluded from the analysis.

The POINT, EPOINT, and QPOINT variable expansions are used in preference mapping analyses (also called PREFMAP, external unfolding, ideal point regression) (Carroll 1972) and for response surface regressions. These three expansions create circular, elliptical, and quadratic response or preference surfaces (see the section "Point Models" on page 10687 and Example 126.6). The CLASS variable expansion is used for main-effects ANOVA.

The following list provides syntax and details for the variable expansion *transforms*.

BSPLINE

BSP

expands each variable to a B-spline basis. You can specify the DEGREE=, KNOTS=, NKNOTS=, and EVENLY= *t-options* with the BSPLINE expansion. When DEGREE=n (3 by default) with k knots (0 by default), $n + k + 1$ variables are created. In addition, the original variable appears in the OUT= data set before the ID variables. For example, **`bspline(x)`** expands x into x_0 x_1 x_2 x_3 and outputs x as well. The x_: variables contain the B-spline basis vectors (which are the same basis vectors that the SPLINE and MSPLINE transformations use internally). The columns of the BSPLINE expansion sum to a column of ones, so an implicit intercept model is fit when the BSPLINE expansion is specified. If you specify the BSPLINE expansion for more than one variable, the model is less than full rank. Variables specified in a BSPLINE expansion must be numeric, and they are typically continuous. See the sections "SPLINE and MSPLINE Transformations" on page 10693 and "SPLINE, BSPLINE, and PSPLINE Comparisons" on page 10696 for more information about B-splines.

CLASS

CLA

expands the variables to a set of coded or "dummy" variables. PROC TRANSREG uses the values of the formatted variables to determine class membership. The specification **`class(x1 x2)`** fits a simple main-effects model, **`class(x1 | x2)`** fits a main-effects and interactions model, and **`class(x1|x2|x3|x4@2 x1*x2*x3)`** fits a model with all main effects, all two-way interactions, and one three-way interaction. Variables specified with the CLASS expansion can be either character or numeric; numeric variables should be discrete. See the section "ANOVA Codings" on page 10664 for more information about CLASS variables. See the section "Model Statement Usage" on page 10620 for information about how to use the operators @, *, and | in PROC TRANSREG.

EPOINT

EPO

expands the variables for an elliptical response surface regression or for an elliptical ideal point regression. Specify the COORDINATES *o-option* to output PREFMAP ideal elliptical point model coordinates to the OUT= data set. Each axis of the ellipse (or ellipsoid) is oriented in the same direction as one of the variables. The EPOINT expansion creates a new variable for each original variable. The value of each new variable is the square of each observed value for the corresponding original variable. The regression analysis then uses both sets of variables (original and squared). Variables specified with the EPOINT expansion must be numeric, and they are typically continuous. See the section "Point Models" on page 10687 and Example 126.6 for more information about point models.

POINT

POI

expands the variables for a circular response surface regression or for a circular ideal point regression. Specify the COORDINATES *o-option* to output PREFMAP ideal point model coordinates to the OUT= data set. The POINT expansion creates a new variable having a value for each observation that is the sum of squares of all the POINT variables. This new variable is added to the set of variables and is used in the regression analysis. For more information about ideal point regression, see Carroll (1972). Variables specified with the POINT expansion must be numeric, and they are typically continuous. See the section "Point Models" on page 10687 and Example 126.6 for more information about point models.

PSPLINE

PSP

expands each variable to a piecewise polynomial basis. You can specify the DEGREE=, KNOTS=, NKNOTS=, and EVENLY *t-options* with PSPLINE. When DEGREE=*n* (3 by default) with *k* knots (0 by default), $n + k$ variables are created. In addition, the original variable appears in the OUT= data set before the ID variables. For example, **`pspline(x / nknots=1)`** expands x into x_1 x_2 x_3 x_4 and outputs x as well. Unlike BSPLINE, an intercept is not implicit in the columns of PSPLINE. Variables specified with the PSPLINE expansion must be numeric, and they are typically continuous. See the sections "SPLINE, BSPLINE, and PSPLINE Comparisons" on page 10696 and "Using Splines and Knots" on page 10632 for more information about splines. Also see Smith (1979) for a good introduction to piecewise polynomial splines.

QPOINT

QPO

expands the variables for a quadratic response surface regression or for a quadratic ideal point regression. Specify the COORDINATES *o-option* to output PREFMAP quadratic ideal point model coordinates to the OUT= data set. For *m* QPOINT variables, $m(m + 1)/2$ new variables are created containing the squares and crossproducts of the original variables. The regression analysis uses both sets (original and crossed). Variables specified with the QPOINT expansion must be numeric, and they are typically continuous. See the section "Point Models" on page 10687 and Example 126.6 for more information about point models.

Nonoptimal Transformations

The nonoptimal transformations, like the variable expansions, are computed before the iterative algorithm begins. Nonoptimal transformations create a single new transformed variable that replaces the original variable. The new variable is not transformed by the subsequent iterative algorithms (except for a possible linear transformation with missing value estimation). The following list provides syntax and details for nonoptimal variable transformations.

ARSIN

ARS

finds an inverse trigonometric sine transformation. Variables specified in the ARSIN *transform* must be numeric and in the interval $(-1.0 \leq x \leq 1.0)$, and they are typically continuous.

EXP

exponentiates variables (x is transformed to a^x). To specify the value of a, use the PARAMETER= *t-option*. By default, a is the mathematical constant $e = 2.718\ldots$. Variables specified with the EXP *transform* must be numeric, and they are typically continuous.

LOG

transforms variables to logarithms (x is transformed to $\log_a(x)$). To specify the base of the logarithm, use the PARAMETER= *t-option*. The default is a natural logarithm with base $e = 2.718\ldots$. Variables specified with the LOG *transform* must be numeric and positive, and they are typically continuous.

LOGIT

finds a logit transformation on the variables. The logit of x is $\log(x/(1-x))$. Unlike other transformations, LOGIT does not have a three-letter abbreviation. Variables specified with the LOGIT *transform* must be numeric and in the interval ($0.0 < x < 1.0$), and they are typically continuous.

POWER

POW

raises variables to a specified power (x is transformed to x^a). You must specify the power parameter a by specifying the PARAMETER= *t-option* following the variables. Here is an example:

```
power(variable / parameter=number)
```

You can use POWER for squaring variables (PARAMETER=2), reciprocal transformations (PARAMETER=–1), square roots (PARAMETER=0.5), and so on. Variables specified with the POWER *transform* must be numeric, and they are typically continuous.

RANK

RAN

transforms variables to ranks. Ranks are averaged within ties. The smallest input value is assigned the smallest rank. Variables specified in the RANK *transform* must be numeric.

Nonlinear Fit Transformations

Nonlinear fit transformations, like nonoptimal transformations, are computed before the iterative algorithm begins. Nonlinear fit transformations create a single new transformed variable that replaces the original variable and provides one or more smooth functions through a scatter plot. The new variable is not transformed by the subsequent iterative algorithms. The nonlinear fit transformations, unlike the nonoptimal transformations, use information in the other variables in the model to find the transformations. The nonlinear fit transformations, unlike the optimal transformations, do not minimize a squared-error criterion. The following list provides syntax and details for nonoptimal variable transformations.

BOXCOX

BOX

finds a Box-Cox (1964) transformation of the specified variables. The BOXCOX transformation can be used only with dependent variables. The ALPHA=, CLL=, CONVENIENT, GEOMETRICMEAN, LAMBDA=, and PARAMETER= *t-options* can be used with the BOXCOX transformation. Variables specified in the BOXCOX *transform* must be numeric, and they are typically continuous. See the section "Box-Cox Transformations" on page 10623 and Example 126.2 for more information about Box-Cox transformations.

PBSPLINE

PBS

is a noniterative penalized B-spline transformation (Eilers and Marx 1996). The PBSPLINE transformation can be used only with independent variables. By default with PBSPLINE, a cubic spline is fit with 100 evenly spaced knots, three evenly spaced exterior knots, and a difference matrix of order three (DEGREE=3 NKNOTS=100 EVENLY=3 PARAMETER=3). Variables specified in the PBSPLINE *transform* must be numeric, and they are typically continuous. See the section "Penalized B-Splines" on page 10657 and Example 126.3 for more information about penalized B-splines.

SMOOTH

SMO

is a noniterative smoothing spline transformation (Reinsch 1967). You can specify the smoothing parameter with either the SM= or the PARAMETER= *t-option*. The default smoothing parameter is SM=0. The SMOOTH transformation can be used only with independent variables. Variables specified with the SMOOTH *transform* must be numeric, and they are typically continuous. See the section "Smoothing Splines" on page 10660 for more information about smoothing splines.

Optimal Transformations

Optimal transformations are iteratively derived. Missing values for these types of variables can be optimally estimated (see the section "Missing Values" on page 10681). The following list provides syntax and details for optimal transformations.

LINEAR

LIN

finds an optimal linear transformation of each variable. For variables with no missing values, the transformed variable is the same as the original variable. For variables with missing values, the transformed nonmissing values have a different scale and origin than the original values. Variables specified in the LINEAR *transform* must be numeric. See the section "OPSCORE, MONOTONE, UNTIE, and LINEAR Transformations" on page 10692 for more information about optimal scaling.

MONOTONE

MON

finds a monotonic transformation of each variable, with the restriction that ties are preserved. The Kruskal (1964) secondary least squares monotonic transformation is used. This transformation weakly preserves order and category membership (ties). Variables specified with the MONOTONE *transform* must be numeric, and they are typically discrete. See the section "OPSCORE, MONOTONE, UNTIE, and LINEAR Transformations" on page 10692 for more information about optimal scaling.

MSPLINE

MSP

finds a monotonically increasing B-spline transformation with monotonic coefficients (De Boor 1978; De Leeuw 1986) of each variable. You can specify the DEGREE=, KNOTS=, NKNOTS=, and EVENLY= *t-options* with MSPLINE. By default, PROC TRANSREG fits a quadratic spline with no knots. Variables specified with the MSPLINE *transform* must be numeric, and they are typically continuous. See the section "SPLINE and MSPLINE Transformations" on page 10693 for more information about monotone splines.

OPSCORE

OPS

finds an optimal scoring of each variable. The OPSCORE transformation assigns scores to each class (level) of the variable. The Fisher (1938) optimal scoring method is used. Variables specified with the OPSCORE *transform* can be either character or numeric; numeric variables should be discrete. See the sections "Character OPSCORE Variables" on page 10686 and "OPSCORE, MONOTONE, UNTIE, and LINEAR Transformations" on page 10692 for more information about optimal scaling.

SPLINE

SPL

finds a B-spline transformation (De Boor 1978) of each variable. By default, PROC TRANSREG fits a cubic spline with no knots. You can specify the DEGREE=, KNOTS=, NKNOTS=, and EVENLY= *t-options* with SPLINE. Variables specified with the SPLINE *transform* must be numeric, and they are typically continuous. See the sections "SPLINE and MSPLINE Transformations" on page 10693, "Specifying the Number of Knots" on page 10694, and "SPLINE, BSPLINE, and PSPLINE Comparisons" on page 10696, and "Using Splines and Knots" on page 10632 for more information about splines.

UNTIE

UNT

finds a monotonic transformation of each variable without the restriction that ties are preserved. PROC TRANSREG uses the Kruskal (1964) primary least squares monotonic transformation method. This transformation weakly preserves order but not category membership (it might untie some previously tied values). Variables specified with the UNTIE *transform* must be numeric, and they are typically discrete. See the section "OPSCORE, MONOTONE, UNTIE, and LINEAR Transformations" on page 10692 for more information about optimal scaling.

Other Transformations

IDENTITY

IDE

specifies variables that are not changed by the iterations. Typically, the IDENTITY transformation is used with a simple variable list, such as **`identity(x1-x5)`**. However, you can also specify interaction terms. For example, **`identity(x1 | x2)`** creates x1, x2, and the product x1*x2; and **`identity(x1 | x2 | x3)`** creates x1, x2, x1*x2, x3, x1*x3, x2*x3, and x1*x2*x3. See the section "Model Statement Usage" on page 10620 for information about how to use the operators @, *, and | in PROC TRANSREG. Variables specified in the IDENTITY *transform* must be numeric.

The IDENTITY transformation is used for variables when no transformation and no missing data estimation are desired. However, the REFLECT *t-option*, the ADDITIVE *a-option*, and the TSTANDARD=Z, and TSTANDARD=CENTER options can linearly transform all variables, including IDENTITY variables, after the iterations. Observations with missing values in IDENTITY variables are excluded from the analysis, and no optimal scores are computed for missing values in IDENTITY variables.

SSPLINE

SSP

finds an iterative smoothing spline transformation of each variable. The SSPLINE transformation does not generally minimize squared error. You can specify the smoothing parameter with either the SM= *t-option* or the PARAMETER= *t-option*. The default smoothing parameter is SM=0. Variables specified with the SSPLINE *transform* must be numeric, and they are typically continuous.

Transformation Options (t-options)

If you use a nonoptimal, nonlinear fit, optimal, or other transformation, you can use *t-options*, which specify additional details of the transformation. The *t-options* are specified within the parentheses that enclose variables and are listed after a slash. You can use *t-options* with both the dependent and the independent variables. Here is an example of using just one *t-option*:

```
proc transreg;
   model identity(y)=spline(x / nknots=3);
   output;
run;
```

The preceding statements find an optimal variable transformation (SPLINE) of the independent variable, and they use a *t-option* to specify the number of knots (NKNOTS=). The following is a more complex example:

```
proc transreg;
   model mspline(y / nknots=3)=class(x1 x2 / effects);
   output;
run;
```

These statements find a monotone spline transformation (MSPLINE with three knots) of the dependent variable and perform a CLASS expansion with effects coding of the independents.

Table 126.3 summarizes the *t-options* available in the MODEL statement.

Table 126.3 Transformation Options

Option	Description
Nonoptimal Transformation	
ORIGINAL	Uses original mean and variance
Parameter Specification	
PARAMETER=	Specifies miscellaneous parameters
SM=	Specifies smoothing parameter
Penalized B-Spline	
AIC	Uses Akaike's information criterion
AICC	Uses corrected AIC
CV	Uses cross validation criterion
GCV	Uses generalized cross validation criterion
LAMBDA=	Specifies smoothing parameter list or range
RANGE	Specifies a LAMBDA= range, not a list
SBC	Uses Schwarz's Bayesian criterion
Spline	
DEGREE=	Specifies the degree of the spline
EVENLY=	Spaces the knots evenly
EXKNOTS=	Specifies exterior knots
KNOTS=	Specifies the interior knots or break points
NKNOTS=	Creates *n* knots
CLASS Variable	
CPREFIX=	Specifies CLASS coded variable name prefix

Table 126.3 *continued*

Option	Description
DEVIATIONS	Specifies a deviations-from-means coding
EFFECTS	Specifies a deviations-from-means coding
LPREFIX=	Specifies CLASS coded variable label prefix
ORDER=	Specifies order of CLASS variable levels
ORTHOGONAL	Specifies an orthogonal-contrast coding
SEPARATORS=	Specifies CLASS coded variable label separators
STANDORTH	Specifies a standardized-orthogonal coding
ZERO=	Controls reference levels
Box-Cox	
ALPHA=	Specifies confidence interval alpha
CLL=	Specifies convenient lambda list
CONVENIENT	Uses a convenient lambda
GEOMETRICMEAN	Scales transformation using geometric mean
LAMBDA=	Specifies power parameter list
Other t-options	
AFTER	Specifies operations occur after the expansion
CENTER	Specifies center before the analysis begins
NAME=	Renames variables
REFLECT	Reflects the variable around the mean
TSTANDARD=	Specifies transformation standardization
Z	Standardizes before the analysis begins

The following sections discuss the *t-options* available for nonoptimal, nonlinear fit, optimal, and other transformations.

Nonoptimal Transformation t-options

ORIGINAL

ORI

matches the variable's final mean and variance to the mean and variance of the original variable. By default, the mean and variance are based on the transformed values. The ORIGINAL *t-option* is available for all of the nonoptimal transformations.

Parameter t-options

PARAMETER=*number*

PAR=*number*

specifies the transformation parameter. The PARAMETER= *t-option* is available for the BOXCOX, EXP, LOG, POWER, SMOOTH, SSPLINE, and PBSPLINE transformations. For BOXCOX, the parameter is the value to add to each value of the variable before a Box-Cox transformation. For EXP, the parameter is the value to be exponentiated; for LOG, the parameter is the base value; and for POWER, the parameter is the power. For SMOOTH and SSPLINE, the parameter is the raw smoothing parameter. (See the SM= option for an alternative way to specify the smoothing parameter.) The

default for the PARAMETER= *t-option* for the BOXCOX transformation is 0 and for the LOG and EXP transformations is $e = 2.718\ldots$. The default parameter for SMOOTH and SSPLINE is computed from SM=0. For the POWER transformation, you must specify the PARAMETER= *t-option*; there is no default. For PBSPLINE, the parameter is the order of the difference matrix, which provides some control over the smoothness of the transformation. The default order parameter with PBSPLINE is the maximum of the DEGREE= *t-option*, and 1. With PBSPLINE, the default is DEGREE=3 and PARAMETER=3, which works well for most problems.

SM=*n*

specifies a smoothing parameter in the range 0 to 100, just like PROC GPLOT uses. For example, SM=50 in PROC TRANSREG is equivalent to I=SM50 in the SYMBOL statement with PROC GPLOT. You can specify the SM= *t-option* only with the SMOOTH and SSPLINE transformations. The smoothness of the function increases as the value of the smoothing parameter increases. By default, SM=0.

Spline t-options

The following *t-options* are available with the SPLINE, MSPLINE and PBSPLINE transformations and with the PSPLINE and BSPLINE expansions.

DEGREE=*n*

DEG=*n*

specifies the degree of the spline transformation. The degree must be a nonnegative integer. The defaults are DEGREE=3 for SPLINE, PSPLINE, and BSPLINE variables and DEGREE=2 for MSPLINE variables.

The polynomial degree should be a small integer, usually 0, 1, 2, or 3. Larger values are rarely useful. If you have any doubt as to what degree to specify, use the default.

EVENLY<=*n***>**

EVE<=*n***>**

is used with the NKNOTS= *t-option* to space the knots evenly. The differences between adjacent knots are constant.

If you specify NKNOTS=*k* and EVENLY, *k* knots are created at

$$\text{minimum} + i((\text{maximum} - \text{minimum})/(k+1))$$

for $i = 1, \ldots, k$. Here is an example:

```
spline(x / nknots=2 evenly)
```

When the variable x has a minimum of 4 and a maximum of 10, then the two interior knots are 6 and 8. Without the EVENLY *t-option*, the NKNOTS= *t-option* places knots at percentiles, so the knots are not evenly spaced. By default for the BSPLINE expansion and the SPLINE and MSPLINE transformations, the smaller exterior knots are all the same and all just a little smaller than the minimum. Similarly, by default, the larger exterior knots are all the same and all just a little larger than the maximum. However, if you specify EVENLY=*n*, then the *n* exterior knots are evenly spaced as well. The number of exterior knots must be greater than or equal to the degree. You can specify values larger than the degree when

you want to interpolate slightly beyond the range or your data. The exterior knots must be less than the minimum or greater than the maximum; hence the knots across all sets are not precisely equally spaced. For example, with data ranging from 0 to 10, and with EVENLY=3 and NKNOTS=4, the first exterior knots are –4.000000000001, –2.000000000001, and –0.000000000001, the interior knots are 2, 4, 6, and 8, and the second exterior knots are 10.000000000001, 12.000000000001, and 14.000000000001.

With the BSPLINE and PSPLINE expansions and the SPLINE and MSPLINE transformations, evenly spaced knots are not the default. With the PBSPLINE transformation, evenly spaced interior and exterior knots are the default. If you want unevenly spaced knots with PBSPLINE, you must use the KNOTS= *t-option*.

EXKNOTS=*number-list*

EXK=*number-list*

specifies exterior knots for SPLINE and MSPLINE transformations and BSPLINE expansions. Usually, this *t-option* is not needed; PROC TRANSREG automatically picks suitable exterior knots. The only time you need to use this option is when you want to ensure that the exact same basis is used for different splines, such as when you apply coefficients from one spline transformation to a variable in a different data set (see the section "Scoring Spline Variables" on page 10643).

Specify one or two values. If the minimum EXKNOTS= value is less than the minimum data value, it is used as the exterior knot. If the maximum EXKNOTS= value is greater than the maximum data value, it is used as the exterior knot. Otherwise these values are ignored. When EXKNOTS= is specified with the CENTER or Z *t-options*, the knots apply to the original variable, not to the centered or standardized variable.

The B-spline transformations and expansions use a knot list consisting of exterior knots (values just smaller than the minimum), the specified (interior) knots, and exterior knots (values just larger than the minimum). You can use the DETAIL *a-option* to see all of these knots. If you use different exterior knots, you get different but equivalent B-spline bases. You can specify exterior knots in either the KNOTS= or EXKNOTS= *t-options*; however, for the BSPLINE expansion, the KNOTS= *t-option* creates extra all-zero basis columns, whereas the EXKNOTS= *t-option* gives you the correct basis. See the EVENLY= *t-option* for an alternative way to specify exterior knots.

KNOTS=*number-list* | *n* **TO** *m* **BY** *p*

KNO=*number-list* | *n* **TO** *m* **BY** *p*

specifies the interior knots or break points. By default, there are no knots. The first time you specify a value in the knot list, it indicates a discontinuity in the nth (from DEGREE=n) derivative of the transformation function at the value of the knot. The second mention of a value indicates a discontinuity in the $(n-1)$ derivative of the transformation function at the value of the knot. Knots can be repeated any number of times for decreasing smoothness at the break points, but the values in the knot list can never decrease.

You cannot use the KNOTS= *t-option* with the NKNOTS= *t-option*. You should keep the number of knots small (see the section "Specifying the Number of Knots" on page 10694).

NKNOTS=*n*

NKN=*n*

creates n knots, the first at the $100/(n+1)$th percentile, the second at the $200/(n+1)$th percentile, and so on. Knots are always placed at data values; there is no interpolation. For example, if NKNOTS=3, knots are placed at the 25th percentile, the median, and the 75th percentile. You can use the EVENLY=

t-option along with NKNOTS= to get evenly spaced knots. By default, with the BSPLINE and PSPLINE expansions and the SPLINE and MSPLINE transformations, NKNOTS=0. By default, with the PBSPLINE transformation, NKNOTS=100.

The value specified for the NKNOTS= *t-option* must be ≥ 0.

You cannot use the NKNOTS= *t-option* with the KNOTS= *t-option*.

You should keep the number of knots small (see the section "Specifying the Number of Knots" on page 10694).

Penalized B-Spline t-options

The following *t-options* are available with the PBSPLINE transformation.

AIC

specifies that the procedure should select the smoothing parameter, λ, that minimizes the (Akaike 1973) information criterion (AIC). By default, the (AICC) criterion is minimized.

AICC

specifies that the procedure should select the smoothing parameter, λ, that minimizes the corrected Akaike information criterion (Hurvich, Simonoff, and Tsai 1998). This is the default criterion unless the AIC, CV, GCV, or SBC *t-option* is specified.

CV

specifies that the procedure should select the smoothing parameter, λ, that minimizes the cross validation criterion (CV). By default, the (AICC) criterion is minimized.

GCV

specifies that the procedure should select the smoothing parameter, λ, that minimizes the generalized cross validation criterion (Craven and Wahba 1979). By default, the (AICC) criterion is minimized.

LAMBDA=*number-list*

LAM=*number-list*

specifies a list of penalized B-spline smoothing parameters. By default, PROC TRANSREG considers lambdas in the range 0 to 1E6. Alternatively, you can specify the RANGE *t-option* with LAMBDA=, such as LAMBDA=1E3 1E5 RANGE, to only consider lambdas in a narrower range. Note that the algorithm might not actually evaluate the criterion at the minimum and maximum if it does not have to. In particular, it avoids evaluating the criterion at LAMBDA=0 (no smoothing) unless it is the only LAMBDA= value specified. You can also specify a list of lambdas, such as LAMBDA=1 TO 10, and the procedure selects the best lambda from the list. In all cases, the lambda that minimizes the specified criterion (or AICC by default) is chosen.

RANGE

RAN

specifies that the LAMBDA= *t-option* specifies two lambdas that define a range of values, from which an optimal lambda is selected. By default, PROC TRANSREG considers lambdas in the range 0 to 1E6.

SBC

specifies that the procedure should select the smoothing parameter, λ, that minimizes Schwarz's Bayesian criterion (Schwarz 1978; Judge et al. 1980). By default, the (AICC) criterion is minimized.

Class Variable t-options

CPREFIX=*n* | *number-list*

CPR=*n* | *number-list*

specifies the number of first characters of a CLASS expansion variable's name to use in constructing names for coded variables. When you specify CPREFIX= as an *a-option* or an *o-option*, it specifies the default for all CLASS variables. When you specify CPREFIX= as a *t-option*, it overrides the default only for selected variables. A different CPREFIX= value can be specified for each CLASS variable by specifying the CPREFIX=number-list *t-option*, like the ZERO=*formatted-value t-option*.

DEVIATIONS

DEV

requests a deviations-from-means coding of CLASS variables. The coded design matrix has values of 0, 1, and –1 for reference levels. This coding is referred to as "deviations-from-means," "effects," "center-point," or "full-rank" coding. For example, here is the coding for two-, three-, four-, and five-level factors:

	Number of Levels									
	Two	Three		Four			Five			
a	1	1	0	1	0	0	1	0	0	0
b	-1	0	1	0	1	0	0	1	0	0
c		-1	-1	0	0	1	0	0	1	0
d				-1	-1	-1	0	0	0	1
e							-1	-1	-1	-1

EFFECTS

EFF

See the DEVIATIONS *t-option*.

LPREFIX=*n* | *number-list*

LPR=*n* | *number-list*

specifies the number of first characters of a CLASS expansion variable's label (or name if no label is specified) to use in constructing labels for the coded variables. When you specify LPREFIX= as an *a-option* or an *o-option*, it specifies the default for all CLASS variables. When you specify LPREFIX= as a *t-option*, it overrides the default only for selected variables. A different LPREFIX= value can be specified for each CLASS variable by specifying the LPREFIX=number-list *t-option*, like the ZERO=*formatted-value t-option*.

ORDER=DATA | FREQ | FORMATTED | INTERNAL

ORD=DAT | FRE | FOR | INT

specifies the order in which the CLASS variable levels are to be reported. The default is ORDER=INTERNAL. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine dependent. When you specify ORDER= as an *a-option* or an *o-option*, it specifies the default ordering for all CLASS variables. When you specify ORDER= as a *t-option*, it overrides the default ordering only for selected variables. You can specify a different ORDER= value for each CLASS specification.

ORTHOGONAL

ORT

requests an orthogonal-contrast coding of CLASS variables. For example, here is the orthogonal-contrast coding for two-, three-, four-, and five-level factors:

	Number of Levels									
	Two	Three		Four			Five			
a	1	1	-1	1	-1	-1	1	-1	-1	-1
b	-1	0	2	0	2	-1	0	2	-1	-1
c		-1	-1	0	0	3	0	0	3	-1
d				-1	-1	-1	0	0	0	4
e							-1	-1	-1	-1

The sum of the coded values within each column is zero, all columns within a factor are orthogonal, and the *i*th column represents a contrast between the *i*th level and the combination of all preceding levels and the last level. The $\mathbf{X}$ matrix is orthogonal and $\mathbf{X}'\mathbf{X}$ is diagonal with this coding only if the experimental design is orthogonal.

SEPARATORS='*string-1*' **<** '*string-2*' **>**

SEP='*string-1*' **<** '*string-2*' **>**

specifies separators for creating CLASS expansion variable labels. By default, SEPARATORS=' ' ' * ' ("blank" and "blank asterisk blank"). When you specify SEPARATORS= as an *a-option* or an *o-option*, it specifies the default separators for all CLASS variables. When you specify SEPARATORS= as a *t-option*, it overrides the default only for selected variables. You can specify a different SEPARATORS= value for each CLASS specification.

STANDORTH

STA

ORTHEFFECT

requests a standardized-orthogonal coding of CLASS variables. For example, here is the standardized-orthogonal coding for two-, three-, four-, and five-level factors:

	Number of Levels									
	Two	Three		Four			Five			
a	1	1.22	-0.71	1.41	-0.82	-0.58	1.58	-0.91	-0.65	-0.50
b	-1	0.00	1.41	0.00	1.63	-0.58	0.00	1.83	-0.65	-0.50
c		-1.22	-0.71	0.00	0.00	1.73	0.00	0.00	1.94	-0.50
d				-1.41	-0.82	-0.58	0.00	0.00	0.00	2.00
e							-1.58	-0.91	-0.65	-0.50

The sum of the coded values within each column is zero, the sum of squares of the coded values within each column is equal to the number of levels, all columns within a factor are orthogonal, and the *i*th column represents a contrast between the *i*th level and the combination of all preceding levels and the last level. The $\mathbf{X}$ matrix is orthogonal and $\mathbf{X}'\mathbf{X}$ is diagonal ($\mathbf{X}'\mathbf{X} = n\mathbf{I}$, the number of observations times an identity matrix) with this coding only if the experimental design is orthogonal.

ZERO=FIRST | LAST | NONE | SUM

ZER=FIR | LAS | NON | SUM

ZERO='*formatted-value*' **<** '*formatted-value*' **... >**

is used with CLASS variables. The default is ZERO=LAST.

The specification CLASS(variable / ZERO=FIRST) sets to missing the coded variable for the first of the sorted categories, implying a zero coefficient for that category.

The specification CLASS(variable / ZERO=LAST) sets to missing the coded variable for the last of the sorted categories, implying a zero coefficient for that category.

The specification CLASS(variable / ZERO='*formatted-value*') sets to missing the coded variable for the category with a formatted value that matches '*formatted-value*', implying a zero coefficient for that category. With ZERO=*formatted-value*, the first formatted value applies to the first variable in the specification, the second formatted value applies to the next variable that was not previously mentioned, and so on. For example, **`class(a a*b b b*c c / zero='x' 'y' 'z')`** specifies that the reference level for a is 'x', for b is 'y', and for c is 'z'. With ZERO='*formatted-value*', the procedure first looks for exact matches between the formatted values and the specified value. If none are found, leading blanks are stripped from both and the values are compared again. If zero or two or more matches are found, warnings are issued.

The specifications ZERO=FIRST, ZERO=LAST, and ZERO='*formatted-value*' are used for reference cell models. The Intercept parameter estimate is the marginal mean for the reference cell, and the other marginal means are obtained by adding the intercept to the coded variable coefficients.

The specification CLASS(variable / ZERO=NONE) sets to missing none of the coded variables. The columns of the expansion sum to a column of ones, so an implicit intercept model is fit. If you specify ZERO=NONE for more than one variable, the model is less than full rank. In the model **`model identity(y) = class(x / zero=none)`**, the coefficients are cell means.

The specification CLASS(variable / ZERO=SUM) sets to missing none of the coded variables, and the coefficients for the coded variables created from the variable sum to 0. This creates a less-than-full-rank model, but the coefficients are uniquely determined due to the sum-to-zero constraint.

In the presence of iterative transformations, hypothesis tests for ZERO=NONE and ZERO=SUM levels are not exact; they are liberal because a model with an explicit intercept is fit inside the iterations. There is no provision for adjusting the transformations while setting to 0 a parameter that is redundant given the explicit intercept and the other parameters.

Box-Cox t-options

The following *t-options* are available only with the BOXCOX transformation of the dependent variable (see the section "Box-Cox Transformations" on page 10623 and Example 126.2).

ALPHA=*p*

ALP=*p*

specifies the Box-Cox alpha for the confidence interval for the power parameter. By default, ALPHA=0.05.

CLL=*number-list*

specifies the Box-Cox convenient lambda list. The default is CLL=1 0 0.5 –1 –0.5 2 –2 3 –3. By default, a linear transformation ($\lambda = 1$) is preferred over log ($\lambda = 0$), square root ($\lambda = 0.5$), inverse ($\lambda = -1$), inverse square root ($\lambda = -0.5$), quadratic ($\lambda = 2$), inverse quadratic ($\lambda = -2$), cubic ($\lambda = 3$), and inverse cubic ($\lambda = -3$) transformations. A value in the CLL= list is considered only if it is also in the LAMBDA= list. When the confidence interval for the power parameter includes one of the parameters in the CLL= list, PROC TRANSREG reports it. In addition, if you specify the CONVENIENT *t-option*, then PROC TRANSREG uses the first convenient power parameter in the CLL=*number-list* that is in the confidence interval. For example, if the optimal power parameter is 0.25 and 0.0 is in the confidence interval but 1.0 is not, then the convenient power parameter is 0.0.

CONVENIENT

CON

requests that PROC TRANSREG use a power parameter from the CLL= *t-option* for the final transformation instead of using the optimal LAMBDA= *t-option* value if a CLL= power parameter is in both the confidence interval and the LAMBDA=*number-list*. For more information, see the CLL= *t-option*.

GEOMETRICMEAN

GEO

divides the Box-Cox transformation by $\dot{y}^{\lambda-1}$, where $\dot{y}$ is the geometric mean of the variable to be transformed. This form of the Box-Cox transformation essentially converts the transformation back to original units, and hence it permits direct comparison of the residual sums of squares for models with different power parameters.

LAMBDA=*number-list*

LAM=*number-list*

specifies a list of Box-Cox power parameters. By default, LAMBDA=–3 TO 3 BY 0.25. PROC TRANSREG evaluates the log likelihood for each power parameter in the list and chooses the best one. However, when the CONVENIENT *t-option* is specified, PROC TRANSREG can choose a convenient parameter from the confidence interval instead of choosing the optimal power parameter from the LAMBDA= list. For example, if the optimal power parameter is 0.25 and 0.0 is in the confidence interval but 1.0 is not, then the convenient power parameter 0.0 (log transformation) is chosen instead of the more optimal parameter 0.25. You can specify the convenient power parameters by using the CLL= *t-option*.

Other t-options

AFTER

AFT

requests that certain operations occur after the expansion. This *t-option* affects the NKNOTS= *t-option* when the SPLINE or MSPLINE transformation is crossed with a CLASS specification. For example, if the original spline variable (1 2 3 4 5 6 7 8 9) is expanded into the three variables (1 2 3 0 0 0 0 0 0), (0 0 0 4 5 6 0 0 0), and (0 0 0 0 0 0 7 8 9), then, by default, NKNOTS=1 would use the overall median of 5 as the knot for all three variables. When you specify the AFTER *t-option*, the knots for the three variables are 2, 5, and 8. Note that the structural zeros are ignored when the internal knot list is created, but they are not ignored for the exterior knots.

You can also specify the AFTER *t-option* with the RANK, SMOOTH, and PBSPLINE transformations. The following specifications compute ranks and smooth transformations within groups, after crossing, ignoring the structural zeros:

```
class(x / zero=none) | rank(z / after)
class(x / zero=none) | smooth(z / after)
```

CENTER
CEN

centers the variables before the analysis begins (in contrast to the TSTANDARD=CENTER option, which centers after the analysis ends). The CENTER *t-option* can be used instead of running PROC STANDARD before PROC TRANSREG (see the section "Centering" on page 10727). When the KNOTS= *t-option* is specified with CENTER, the knots apply to the original variable, not to the centered variable. PROC TRANSREG centers the knots.

NAME=(*variable-list***)**
NAM=(*variable-list***)**

renames variables as they are used in the MODEL statement. This *t-option* lets you use a variable more than once.

For example, if x is a character variable, then the following step stores both the original character variable x and a numeric variable xc that contains category numbers in the OUT= data set:

```
proc transreg data=a;
   model identity(y) = opscore(x / name=(xc));
   output;
   id x;
run;
```

With the CLASS and IDENTITY transformations, which can contain interaction effects, the first name applies to the first variable in the specification, the second name applies to the next variable that was not previously mentioned, and so on. For example, **identity(a a * b b b * c c / name=(g h i))** specifies that the new name for a is g, for b is h, and for c is i. The same assignment is used for the (not useful) specification **identity(a a b b c c / name=(g h i))**. For all *transforms* other than CLASS and IDENTITY (all those in which interactions are not supported), repeated variables are not handled specially. For example, **spline(a a b b c c / name=(a g b h c i))** creates six variables: a copy of a named a, another copy of a named g, a copy of b named b, another copy of b named h, a copy of c named c, and another copy of c named i.

REFLECT
REF

reflects the transformation

$$y = -(y - \bar{y}) + \bar{y}$$

after the iterations are completed and before the final standardization and results calculations. This *t-option* is particularly useful with the dependent variable in a conjoint analysis. When the dependent variable consists of ranks with the most preferred combination assigned 1.0, the REFLECT *t-option* reflects the transformation so that positive utilities mean high preference. (See Example 126.4.)

TSTANDARD=CENTER | NOMISS | ORIGINAL | Z

TST=CEN | NOM | ORI | Z

specifies the standardization of the transformed variables for the hypothesis tests and in the OUT= data set (see the section "Centering" on page 10727). By default, TSTANDARD=ORIGINAL. When you specify TSTANDARD= as an *a-option* or an *o-option*, it determines the default standardization for all variables. When you specify TSTANDARD= as a *t-option*, it overrides the default standardization only for selected variables. You can specify a different TSTANDARD= value for each transformation. For example, to perform a redundancy analysis with standardized dependent variables, specify the following:

```
model identity(y1-y4 / tstandard=z) = identity(x1-x10);
```

Z

centers and standardizes the variables to variance one before the analysis begins (in contrast to the TSTANDARD=Z option, which standardizes after the analysis ends). The Z *t-option* can be used instead of running PROC STANDARD before PROC TRANSREG (see the section "Centering" on page 10727). When the KNOTS= *t-option* is specified with Z, the knots apply to the original variable, not to the standardized variable. PROC TRANSREG standardizes the knots.

Algorithm Options (a-options)

This section discusses the options that can appear in the PROC TRANSREG or MODEL statement as *a-options*. They are listed after the entire model specification and after a slash. Here is an example:

```
proc transreg;
   model spline(y / nknots=3)=log(x1 x2 / parameter=2)
         / nomiss maxiter=50;
   output;
run;
```

In the preceding statements, NOMISS and MAXITER= are *a-options*. (SPLINE and LOG are *transforms*, and NKNOTS= and PARAMETER= are *t-options*.) The statements find a spline transformation with 3 knots on y and a base 2 logarithmic transformation on x1 and x2. The NOMISS *a-option* excludes all observations with missing values, and the MAXITER= *a-option* specifies the maximum number of iterations.

Table 126.4 summarizes the *a-options* available in the PROC TRANSREG or MODEL statement.

Table 126.4 Options Available in the PROC TRANSREG or MODEL Statement

Option	Description
Input Control	
REITERATE	Restarts iterations
TYPE=	Specifies input observation type
Method and Iterations	
CCONVERGE=	Specifies minimum criterion change
CONVERGE=	Specifies minimum data change
MAXITER=	Specifies maximum number of iterations

Table 126.4 *continued*

Option	Description
METHOD=	Specifies iterative algorithm
NCAN=	Specifies number of canonical variables
NSR	Specifies no restrictions on smoothing models
SINGULAR=	Specifies singularity criterion
SOLVE	Attempts direct solution instead of iteration
Missing Data Handling	
INDIVIDUAL	Fits each model individually (METHOD=MORALS)
MONOTONE=	Includes monotone special missing values
NOMISS	Excludes observations with missing values
UNTIE=	Unties special missing values
Intercept and CLASS Variables	
CPREFIX=	Specifies CLASS coded variable name prefix
LPREFIX=	Specifies CLASS coded variable label prefix
NOINT	Specifies no intercept or centering
ORDER=	Specifies order of CLASS variable levels
REFERENCE=	Controls output of reference levels
SEPARATORS=	Specifies CLASS coded variable label separators
Control Displayed Output	
ALPHA=	Specifies confidence limits alpha
CL	Displays parameter estimate confidence limits
DETAIL	Displays model specification details
HISTORY	Displays iteration histories
NOPRINT	Suppresses displayed output
PBOXCOXTABLE	Prints the Box-Cox log likelihood table
RSQUARE	Displays the R square
SHORT	Suppresses the iteration histories
SS2	Displays regression results
TEST	Displays ANOVA table
TSUFFIX=	Shortens transformed variable labels
UTILITIES	Displays conjoint part-worth utilities
Standardization	
ADDITIVE	Fits additive model
NOZEROCONSTANT	Does not zero constant variables
TSTANDARD=	Specifies transformation standardization

The following list provides details about these *a-options*. The *a-options* are available in the PROC TRANSREG or MODEL statement.

ADDITIVE

ADD

creates an additive model by multiplying the values of each independent variable (after the TSTANDARD= standardization) by that variable's corresponding multiple regression coefficient. This process scales the independent variables so that the predicted-values variable for the final dependent variable is simply the sum of the final independent variables. An additive model is a univariate multiple regression model. As a result, the ADDITIVE *a-option* is not valid if METHOD=CANALS, or if METHOD=REDUNDANCY or METHOD=UNIVARIATE with more than one dependent variable.

ALPHA=*number*

ALP=*number*

specifies the level of significance for all of the confidence limits. By default, ALPHA=0.05.

CCONVERGE=*n*

CCO=*n*

specifies the minimum change in the criterion being optimized (squared multiple correlation for METHOD=MORALS and METHOD=UNIVARIATE, average squared multiple correlation for METHOD=REDUNDANCY, average squared canonical correlation for METHOD=CANALS) that is required to continue iterating. By default, CCONVERGE=0.0.

CL

requests confidence limits on the parameter estimates in the displayed output.

CONVERGE=*n*

CON=*n*

specifies the minimum average absolute change in standardized variable scores that is required to continue iterating. By default, CONVERGE=0.00001. Average change is computed over only those variables that can be transformed by the iterations; that is, all LINEAR, OPSCORE, MONOTONE, UNTIE, SPLINE, MSPLINE, and SSPLINE variables and nonoptimal transformation variables with missing values.

CPREFIX=*n*

CPR=*n*

specifies the number of first characters of a CLASS expansion variable's name to use in constructing names for coded variables. Let v be the maximum variable length (for example, 32 for VALIDVARNAME=V7). Coded variable names are constructed from the first n characters of the CLASS expansion variable's name and the first $v - n$ characters of the formatted CLASS expansion variable's value. For example, if the variable ClassVariable has values 1, 2, and 3, then, by default, the coded variables are named ClassVariable1, ClassVariable2, and ClassVariable3. However, with CPREFIX=5, the coded variables are named Class1, Class2, and Class3. When CPREFIX=0, coded variable names are created entirely from the CLASS expansion variable's formatted values. Valid values range from –1 to 31, where –1 indicates the default calculation and 0 to 31 are the number of prefix characters to use. The default, –1, sets n to $v - \min(v, \max(2, fl))$, where fl is the format length. When you specify CPREFIX= as an *a-option* or an *o-option*, it specifies the default for all CLASS variables. When you specify CPREFIX= as a *t-option*, it overrides the default only for selected variables.

DETAIL

DET

reports on details of the model specification. For example, it reports the knots and coefficients for splines, reference levels for CLASS variables, Box-Cox results, the smoothing parameter, and so on. The DETAIL option can take two optional suboptions, NOCOEFFICIENTS and NOKNOTS (or NOC and NOK). To suppress knots from the details listing, specify DETAIL(NOKNOTS). To suppress coefficients from the details listing, specify DETAIL(NOCOEFFICIENTS). To suppress both knots and coefficients from the details listing, specify DETAIL(NOKNOTS NOCOEFFICIENTS).

SOLVE

SOL

DUMMY

DUM

provides a canonical initialization. When there are no monotonicity constraints, when there is at most one canonical variable in each set, and when there is enough available memory, PROC TRANSREG (with the SOLVE *a-option*) can usually directly solve for the optimal solution in only one iteration. The initialization iteration is number 0, which is slower and uses more memory than other iterations. However, for some models, specifying the SOLVE *a-option* can greatly decrease the amount of time required to find the optimal transformations. During iteration 0, each variable is replaced by an expanded variable and the model is fit to the larger, expanded set of variables. For example, an OPSCORE variable is expanded into coded (or "dummy") variables, as if CLASS were specified, and a SPLINE variable is expanded into a B-spline basis, as if BSPLINE were specified. Then for each expanded variable, the results of iteration zero are constructed by multiplying the expanded basis times the $\boldsymbol{\beta}$ subvector to get the optimal transformation. This *a-option* can be useful even in models where a direct solution is not possible, because it provides good initial transformations of all the variables.

HISTORY

HIS

displays the iteration histories even when the NOPRINT *a-option* is specified.

INDIVIDUAL

IND

fits each model for each dependent variable individually. This means, for example, that when INDIVIDUAL is specified, missing values in one dependent variable will not cause that observation to be deleted for the other models with the other dependent variables. In contrast, by default, missing values in any variable in any model can cause the observation to be deleted for all models. The INDIVIDUAL *a-option* can be specified only with METHOD=MORALS.

This *a-option* also affects the order of the output. By default, the number of observations table is printed once at the beginning of the output. With INDIVIDUAL, a number of observations table appears for each model.

LPREFIX=*n*

LPR=*n*

specifies the number of first characters of a CLASS expansion variable's label (or name if no label is specified) to use in constructing labels for coded variables. Coded variable labels are constructed from the first *n* characters of the CLASS expansion variable's name and the first 127 – *n* characters of the formatted CLASS expansion variable's value. Valid values range from –1 to 127. Values of 0

to 127 specify the number of name or label characters to use. The default is –1, which specifies that PROC TRANSREG should pick a value depending on the length of the prefix and the formatted class value. When you specify LPREFIX= as an *a-option* or an *o-option*, it determines the default for all CLASS variables. When you specify LPREFIX= as a *t-option*, it overrides the default only for selected variables.

MAXITER=*n*

MAX=*n*

specifies the maximum number of iterations (see the section "Controlling the Number of Iterations" on page 10683). By default, MAXITER=30. You can specify MAXITER=0 to save time when no transformations are requested.

METHOD=CANALS | MORALS | REDUNDANCY | UNIVARIATE

MET=CAN | MOR | RED | UNI

specifies the iterative algorithm. By default, METHOD=UNIVARIATE, unless you specify options that cannot be handled by the UNIVARIATE algorithm. Specifically, the default is METHOD=MORALS for the following situations:

- if you specify LINEAR, OPSCORE, MONOTONE, UNTIE, SPLINE, MSPLINE, or SSPLINE transformations for the independent variables
- if you specify the ADDITIVE *a-option* with more than one dependent variable
- if you specify the IAPPROXIMATIONS *o-option*
- if you specify the INDIVIDUAL *a-option*
- if ODS Graphics is enabled, regression plots are produced, and there is more than one dependent variable

CANALS specifies canonical correlation with alternating least squares. This jointly transforms all dependent and independent variables to maximize the average of the first *n* squared canonical correlations, where *n* is the value of the NCAN= *a-option*.

MORALS specifies multiple optimal regression with alternating least squares. This transforms each dependent variable, along with the set of independent variables, to maximize the squared multiple correlation.

REDUNDANCY jointly transforms all dependent and independent variables to maximize the average of the squared multiple correlations (see the section "Redundancy Analysis" on page 10688).

UNIVARIATE transforms each dependent variable to maximize the squared multiple correlation, while the independent variables are not transformed.

MONOTONE=*two-letters*

MON=*two-letters*

specifies the first and last special missing value in the list of those special missing values to be estimated with within-variable order and category constraints. By default, there are no order constraints on missing value estimates. The *two-letters* value must consist of two letters in alphabetical order. For example, MONOTONE=DF means that the estimate of .D must be less than or equal to the estimate of .E, which must be less than or equal to the estimate of .F; no order constraints are placed on estimates of ._, .A through .C, and .G through .Z. For details, see the section "Missing Values" on page 10681.

NCAN=*n*

NCA=*n*

specifies the number of canonical variables to use in the METHOD=CANALS algorithm. By default, NCAN=1. The value of the NCAN= *a-option* must be ≥ 1.

When canonical coefficients and coordinates are included in the OUT= data set, the NCAN= *a-option* also controls the number of rows of the canonical coefficient matrices in the data set. If you specify an NCAN= value larger than the minimum of the number of dependent variables and the number of independent variables, PROC TRANSREG displays a warning and sets the NCAN= *a-option* to the maximum value.

NOINT

NOI

omits the intercept from the OUT= data set and suppresses centering of data. You cannot specify the NOINT *a-option* with iterative transformations since there is no provision for optimal scaling without an intercept. The NOINT *a-option* can be specified only when there is no implicit intercept and when all of the data in a BY group absolutely will not change during the iterations.

NOMISS

NOM

excludes all observations with missing values from the analysis, but does not exclude them from the OUT= data set. If you omit the NOMISS *a-option*, PROC TRANSREG simultaneously computes the optimal transformations of the nonmissing values and estimates the missing values that minimize squared error. For details, see the section "Missing Values" on page 10681.

Casewise deletion of observations with missing values occurs when the NOMISS *a-option* is specified, when there are missing values in expansions, when there are missing values in METHOD=UNIVARIATE independent variables, when there are weights less than or equal to 0, or when there are frequencies less than 1. Excluded observations are output with a blank value for the _TYPE_ variable, and they have a weight of 0. They do not contribute to the analysis but are scored and transformed as *supplementary* or passive observations.

See the section "Passive Observations" on page 10687 for more information about excluded observations.

NOPRINT

NOP

suppresses the display of all output unless you specify the HISTORY *a-option*. The NOPRINT *a-option* without the HISTORY *a-option* disables the Output Delivery System (ODS), including ODS Graphics, for the duration of the procedure run. The NOPRINT *a-option* with the HISTORY *a-option* disables all output except the iteration history, again including ODS Graphics, for the duration of the procedure run. For more information, see Chapter 23, "Using the Output Delivery System."

NOZEROCONSTANT

NOZERO

NOZ

specifies that constant variables are expected and should not be zeroed. By default, constant variables are zeroed. This option is useful when PROC TRANSREG is used to code experimental designs for discrete choice models (see the section "Discrete Choice Experiments: DESIGN, NORESTORE,

NOZERO" on page 10726). When these designs are very large, it might be more efficient to use the DESIGN=*n a-option*. It might be that attributes are constant within a block of *n* observations, so you need to specify the NOZEROCONSTANT *a-option* to get the correct results. You can specify this option in the PROC TRANSREG, MODEL, and OUTPUT statements.

NSR

specifies that no restrictions are placed on the use of SMOOTH and SSPLINE and the ordinary least squares is used to find the coefficients and predicted values. By default, only certain types of models can be specified with SMOOTH and ordinary least squares is not used to find the coefficients and predicted values.

ORDER=DATA | FREQ | FORMATTED | INTERNAL

ORD=DAT | FRE | FOR | INT

specifies the order in which the CLASS variable levels are to be reported. The default is ORDER=INTERNAL. For ORDER=FORMATTED and ORDER=INTERNAL, the sort order is machine dependent. When you specify ORDER= as an *a-option* or an *o-option*, it determines the default ordering for all CLASS variables. When you specify ORDER= as a *t-option*, it overrides the default ordering only for selected variables.

DATA	sorts by order of appearance in the input data set.
FORMATTED	sorts by formatted value.
FREQ	sorts by descending frequency count; levels with the most observations appear first.
INTERNAL	sorts by unformatted value.

PBOXCOXTABLE

PBO

prints the Box-Cox table with the log likelihood displayed as a function of lambda. The important information in this table is displayed in the Box-Cox plot, so when ODS Graphics is enabled and the plot is produced, the table is not produced by default. When ODS Graphics is not enabled or when the plot is not produced, the table is produced by default. Specify the PBOXCOXTABLE option if you want to see the table in addition to the plot.

REFERENCE=NONE | MISSING | ZERO

REF=NON | MIS | ZER

specifies how reference levels of CLASS variables are to be treated. The options are REFERENCE=NONE, the default, in which reference levels are suppressed; REFERENCE=MISSING, in which reference levels are displayed and output with missing values; and REFERENCE=ZERO, in which reference levels are displayed and output with zeros. You can specify the REFERENCE= option in the PROC TRANSREG, MODEL, or OUTPUT statement, and you can specify it independently for the OUT= data set and the displayed output. When you specify it in only one statement, it sets the option for both the displayed output and the OUT= data set.

REITERATE

REI

enables PROC TRANSREG to use previous transformations as starting points. The REITERATE *a-option* affects only variables that are iteratively transformed (specified as LINEAR, OPSCORE,

MONOTONE, UNTIE, SPLINE, MSPLINE, and SSPLINE). For iterative transformations, the REITERATE *a-option* requests a search in the input data set for a variable that consists of the value of the TDPREFIX= or TIPREFIX= *o-option* followed by the original variable name. If such a variable is found, it is used to provide the initial values for the first iteration. The final transformation is a member of the transformation family defined by the original variable, not the transformation family defined by the initialization variable. See the section "Using the REITERATE Algorithm Option" on page 10684 for more information about the REITERATE option.

RSQUARE

RSQ

prints a table with only the model R square.

SEPARATORS='*string-1*' **<** '*string-2*' **>**

SEP='*string-1*' **<** '*string-2*' **>**

specifies separators for creating CLASS expansion variable labels. By default, SEPARATORS=' ' ' * ' ("blank" and "blank asterisk blank"). The first value is used to separate variable names and values in interactions. The second value is used to separate interaction components. For example, the label for the coded variable for the A=1 and B=2 cell is, by default, 'A 1 * B 2'. If SEPARATORS='=' 'x' is specified, then the label is 'A=1xB=2'. When you specify SEPARATORS= as an *a-option* or an *o-option*, it determines the default separators for all CLASS variables. When you specify SEPARATORS= as a *t-option*, it overrides the default only for selected variables.

SHORT

SHO

suppresses the iteration histories.

SINGULAR=*n*

SIN=*n*

specifies the largest value within rounding error of zero. By default, SINGULAR=1E–12. PROC TRANSREG uses the value of the SINGULAR= *a-option* for checking $1 - \mathrm{R}^2$ when constructing full-rank matrices of predictor variables, checking denominators before dividing, and so on. PROC TRANSREG computes the regression coefficients by sweeping with rational pivoting.

SS2

produces a regression table based on Type II sums of squares. Tests of the contribution of each transformation to the overall model are displayed and output to the OUTTEST= data set when you specify the OUTTEST= option. When you specify the SS2 *a-option*, the TEST *a-option* is automatically specified for you. See the section "Hypothesis Tests" on page 10696 for more information about the TEST and SS2 options. You can suppress the variable labels in the regression tables by specifying the NOLABEL option in the OPTIONS statement.

TEST

TES

generates an ANOVA table. PROC TRANSREG tests the null hypothesis that the vector of scoring coefficients for all of the transformations is zero. See the section "Hypothesis Tests" on page 10696 for more information about the TEST option.

TSUFFIX=*n*

TSU=*n*

specifies the number of characters in "Transformation" to append to variable labels for transformed variables. By default, all characters are used.

TSTANDARD=CENTER | NOMISS | ORIGINAL | Z

TST=CEN | NOM | ORI | Z

specifies the standardization of the transformed variables for the hypothesis tests and in the OUT= data set. By default, TSTANDARD=ORIGINAL. When you specify TSTANDARD= as an *a-option* or an *o-option*, it determines the default standardization for all variables. When you specify TSTANDARD= as a *t-option*, it overrides the default standardization only for selected variables.

CENTER centers the output variables to mean zero, but the variances are the same as the variances of the input variables.

NOMISS sets the means and variances of the transformed variables in the OUT= data set, computed over all output values that correspond to nonmissing values in the input data set, to the means and variances computed from the nonmissing observations of the original variables. The TSTANDARD=NOMISS specification is useful with missing data. When a variable is linearly transformed, the final variable contains the original nonmissing values and the missing value estimates. In other words, the nonmissing values are unchanged. If your data have no missing values, TSTANDARD=NOMISS and TSTANDARD=ORIGINAL produce the same results.

ORIGINAL sets the means and variances of the transformed variables to the means and variances of the original variables. This is the default.

Z standardizes the variables to mean zero, variance one.

The final standardization is affected by other options. If you also specify the ADDITIVE *a-option*, the TSTANDARD= option specifies an intermediate step in computing the final means and variances. The final independent variables, along with their means and standard deviations, are scaled by the regression coefficients, creating an additive model with all coefficients equal to one.

For nonoptimal variable transformations, the means and variances of the original variables are actually the means and variances of the nonlinearly transformed variables, unless you specify the ORIGINAL nonoptimal *t-option* in the MODEL statement. For example, if a variable x with no missing values is specified as LOG, then, by default, the final transformation of x is simply the log of x, not the log of x standardized to the mean of x and variance of x.

TYPE='*text***'|***name*

TYP='*text***'|***name*

specifies the valid value for the _TYPE_ variable in the input data set. If PROC TRANSREG finds an input _TYPE_ variable, it uses only observations with a _TYPE_ value that matches the TYPE= value. This enables a PROC TRANSREG OUT= data set containing coefficients to be used as input to PROC TRANSREG without requiring a WHERE statement to exclude the coefficients. If a _TYPE_ variable is not in the data set, all observations are used. The default is TYPE='SCORE', so if you do not specify the TYPE= *a-option*, only observations with _TYPE_='SCORE' are used. Do not confuse this *a-option* with the data set TYPE= option. The DATA= data set must be an ordinary SAS data set.

PROC TRANSREG displays a note when it reads observations with blank values of _TYPE_, but it does not automatically exclude those observations. Data sets created by the TRANSREG and

PRINQUAL procedures have blank _TYPE_ values for those observations that were excluded from the analysis due to nonpositive weights, nonpositive frequencies, or missing data. When these observations are read again, they are excluded for the same reason that they were excluded from their original analysis, not because their _TYPE_ value is blank.

UNTIE=*two-letters*

UNT=*two-letters*

specifies the first and last special missing values in the list of those special missing values that are to be estimated with within-variable order constraints but no category constraints. The *two-letters* value must consist of two letters in alphabetical order. By default, there are category constraints but no order constraints on special missing value estimates. For details, see the sections "Missing Values" on page 10681 and "Optimal Scaling" on page 10691.

UTILITIES

UTI

produces a table of the part-worth utilities from a conjoint analysis. Utilities, their standard errors, and the relative importance of each factor are displayed and output to the OUTTEST= data set when you specify the OUTTEST= option. When you specify the UTILITIES *a-option*, the TEST *a-option* is automatically specified for you. See Example 126.4 and Example 126.5 for more information about conjoint analysis.

OUTPUT Statement

OUTPUT OUT=*SAS-data-set* < *o-options* > **;**

The OUTPUT statement creates a new SAS data set that contains coefficients, marginal means, and information about the original and transformed variables. The information about original and transformed variables composes the score partition of the data set; observations have _TYPE_='SCORE'. The coefficients and marginal means compose the coefficient partition of the data set; observations have _TYPE_='M COEFFI' or _TYPE_='MEAN'. Other values of _TYPE_ are possible; for details, see "_TYPE_ and _NAME_ Variables" later in this chapter. For details about data set structure, see the section "Output Data Set" on page 10698. To specify the name of the output data set, use the OUT= option.

OUT=*SAS-data-set*

specifies the output data set for the data, transformed data, predicted values, residuals, scores, coefficients, and so on. When you use an OUTPUT statement but do not use the OUT= specification, PROC TRANSREG creates a data set and uses the DATA*n* convention. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

To control the contents of the data set and variable names, use one or more of the *o-options*. You can also specify these options in the PROC TRANSREG statement.

Output Options (o-options)

Table 126.5 summarizes the options available in the OUTPUT statement. These options include the OUT= option and all of the *o-options*. Many of the statistics created in the OUTPUT statement are exactly the same as statistics created by PROC REG. More details are given in the sections "Predicted and Residual Values" on page 8801, "Model Fit and Diagnostic Statistics" on page 8808 in Chapter 105, "The REG Procedure," and Chapter 4, "Introduction to Regression Procedures."

Table 126.5 Options Available in the OUTPUT Statement

Option	Description
Identify Output Data Set	
OUT=	Outputs data set
Predicted Values, Residuals, Scores	
CANONICAL	Outputs canonical scores
CLI	Outputs individual confidence limits
CLM	Outputs mean confidence limits
DESIGN=	Specifies design matrix coding
DREPLACE	Replaces dependent variables
IREPLACE	Replaces independent variables
LEVERAGE	Outputs leverage
NORESTOREMISSING	Does not restore missing values
NOSCORES	Suppresses output of scores
PREDICTED	Outputs predicted values
REDUNDANCY=	Outputs redundancy variables
REPLACE	Replaces all variables
RESIDUALS	Outputs residuals
Output Data Set Coefficients	
COEFFICIENTS	Outputs coefficients
COORDINATES=	Outputs ideal point coordinates
MEANS	Outputs marginal means
MREDUNDANCY	Outputs redundancy analysis coefficients
Output Data Set Variable Name Prefixes	
ADPREFIX=	Specifies dependent variable approximations
AIPREFIX=	Specifies independent variable approximations
CDPREFIX=	Specifies canonical dependent variables
CILPREFIX=	Specifies conservative individual lower CL
CIPREFIX=	Specifies canonical independent variables
CIUPREFIX=	Specifies conservative-individual-upper CL
CMLPREFIX=	Specifies conservative-mean-lower CL
CMUPREFIX=	Specifies conservative-mean-upper CL
DEPENDENT=	Specifies METHOD=MORALS untransformed dependent
LILPREFIX=	Specifies liberal-individual-lower CL
LIUPREFIX=	Specifies liberal-individual-upper CL
LMLPREFIX=	Specifies liberal-mean-lower CL
LMUPREFIX=	Specifies liberal-mean-upper CL

Table 126.5 *continued*

Option	Description
RDPREFIX=	Specifies residuals
PPREFIX=	Specifies predicted values
RPREFIX=	Specifies redundancy variables
TDPREFIX=	Specifies transformed dependents
TIPREFIX=	Specifies transformed independents
Macros Variables	
MACRO	Creates macro variables
Other Options	
APPROXIMATIONS	Outputs dependent and independent approximations
CCC	Outputs canonical correlation coefficients
CEC	Outputs canonical elliptical point coordinates
CPC	Outputs canonical point coordinates
CQC	Outputs canonical quadratic point coordinates
DAPPROXIMATIONS	Outputs approximations to transformed dependents
IAPPROXIMATIONS	Outputs approximations to transformed independents
MEC	Outputs elliptical point coordinates
MPC	Outputs point coordinates
MQC	Outputs quadratic point coordinates
MRC	Outputs multiple regression coefficients

For the coefficients partition, the COEFFICIENTS, COORDINATES, and MEANS *o-options* provide the coefficients that are appropriate for your model. For more explicit control of the coefficient partition, use the options that control details and prefixes. The following list provides details about these options.

ADPREFIX=*name*

ADP=*name*

specifies a prefix for naming the dependent variable predicted values. The default is ADPREFIX=P when you specify the PREDICTED *o-option*; otherwise, it is ADPREFIX=A. When you specify the ADPREFIX= *o-option*, the PREDICTED *o-option* is automatically specified for you. The ADPREFIX= *o-option* is the same as the PPREFIX= *o-option*.

AIPREFIX=*name*

AIP=*name*

specifies a prefix for naming the independent variable approximations. The default is AIPREFIX=A. When you specify the AIPREFIX= *o-option*, the IAPPROXIMATIONS *o-option* is automatically specified for you.

APPROXIMATIONS

APPROX

APP

is equivalent to specifying both the DAPPROXIMATIONS and the IAPPROXIMATIONS *o-options*. If you specify METHOD=UNIVARIATE, then the APPROXIMATIONS *o-option* specifies only the DAPPROXIMATIONS *o-option*.

CANONICAL
CAN

outputs canonical variables to the OUT= data set. When you specify METHOD=CANALS, the CANONICAL *o-option* is automatically specified for you. The CDPREFIX= *o-option* specifies a prefix for naming the dependent canonical variables (default Cand), and the CIPREFIX= *o-option* specifies a prefix for naming the independent canonical variables (default Cani).

CCC

outputs canonical correlation coefficients to the OUT= data set.

CDPREFIX=*name*
CDP=*name*

provides a prefix for naming the canonical dependent variables. The default is CDPREFIX=Cand. When you specify the CDPREFIX= *o-option*, the CANONICAL *o-option* is automatically specified for you.

CEC

outputs canonical elliptical point model coordinates to the OUT= data set.

CILPREFIX=*name*
CIL=*name*

specifies a prefix for naming the conservative-individual-lower confidence limits. The default prefix is CIL. When you specify the CILPREFIX= *o-option*, the CLI *o-option* is automatically specified for you.

CIPREFIX=*name*
CIP=*name*

provides a prefix for naming the canonical independent variables. The default is CIPREFIX=Cani. When you specify the CIPREFIX= *o-option*, the CANONICAL *o-option* is automatically specified for you.

CIUPREFIX=*name*
CIU=*name*

specifies a prefix for naming the conservative-individual-upper confidence limits. The default prefix is CIU. When you specify the CIUPREFIX= *o-option*, the CLI *o-option* is automatically specified for you.

CLI

outputs individual confidence limits to the OUT= data set. The names of the confidence limits variables are constructed from the original dependent variable names and the prefixes specified in the following *o-options*: LILPREFIX= (default LIL for liberal individual lower), CILPREFIX= (default CIL for conservative individual lower), LIUPREFIX= (default LIU for liberal individual upper), and CIUPREFIX= (default CIU for conservative individual upper). When there are no monotonicity constraints, the liberal and conservative limits are the same.

CLM

outputs mean confidence limits to the OUT= data set. The names of the confidence limits variables are constructed from the original dependent variable names and the prefixes specified in the following *o-options*: LMLPREFIX= (default LML for liberal mean lower), CMLPREFIX= (default CML for conservative mean lower), LMUPREFIX= (default LMU for liberal mean upper), and CMUPREFIX= (default CMU for conservative mean upper). When there are no monotonicity constraints, the liberal and conservative limits are the same.

CMLPREFIX=*name*

CML=*name*

specifies a prefix for naming the conservative-mean-lower confidence limits. The default prefix is CML. When you specify the CMLPREFIX= *o-option*, the CLM *o-option* is automatically specified for you.

CMUPREFIX=*name*

CMU=*name*

specifies a prefix for naming the conservative-mean-upper confidence limits. The default prefix is CMU. When you specify the CMUPREFIX= *o-option*, the CLM *o-option* is automatically specified for you.

COEFFICIENTS

COE

outputs either multiple regression coefficients or raw canonical coefficients to the OUT= data set. If you specify METHOD=CANALS (in the MODEL or PROC TRANSREG statement), then the COEFFICIENTS *o-option* outputs the first *n* canonical variables, where *n* is the value of the NCAN= *a-option* (specified in the MODEL or PROC TRANSREG statement). Otherwise, the COEFFICIENTS *o-option* includes multiple regression coefficients in the OUT= data set. In addition, when you specify the CLASS expansion for any independent variable, the COEFFICIENTS *o-option* also outputs marginal means.

COORDINATES<=*n*>

COO<=*n*>

outputs either ideal point or vector model coordinates for preference mapping to the OUT= data set. When METHOD=CANALS, these coordinates are computed from canonical coefficients; otherwise, the coordinates are computed from multiple regression coefficients. For details, see the section "Point Models" on page 10687.

When ODS Graphics is enabled and vector model coordinates are requested, a plot is produced with points for each row and vectors for each column. If the vectors are plotted based on the actual computed coordinates, then often the vectors are short. A better graphical display is produced when the vectors are stretched. The absolute lengths of each vector can optionally be changed by specifying COORDINATES=*n*. Then the vector coordinates are all multiplied by *n*. Usually, *n* is a value such as 2, 2.5, or 3. The default is 2.5. Specify COORDINATES=1 if you want to see the vectors without any stretching. The relative lengths of the different vectors are important and interpretable, and these are preserved by the stretching.

CPC

outputs canonical point model coordinates to the OUT= data set.

CQC

outputs canonical quadratic point model coordinates to the OUT= data set.

DAPPROXIMATIONS

DAP

outputs the approximations of the transformed dependent variables to the OUT= data set. These are the target values for the optimal transformations. With METHOD=UNIVARIATE and METHOD=MORALS, the dependent variable approximations are the ordinary predicted values from the linear model. The names of the approximation variables are constructed from the ADPREFIX= *o-option* (default A) and the original dependent variable names. For ordinary predicted values, use the

PREDICTED *o-option* instead of the DAPPROXIMATIONS *o-option*, since the PREDICTED *o-option* uses a more relevant prefix ("P" instead of "A") and a more relevant variable label suffix ("Predicted Values" instead of "Approximations").

DESIGN< =*n*>

DES< =*n*>

specifies that your primary goal is design matrix coding, not analysis. Specifying the DESIGN *o-option* makes the procedure run faster. The DESIGN *o-option* sets the default method to UNIVARIATE and the default MAXITER= value to zero. It suppresses computing the regression coefficients, unless they are needed for some other option. Furthermore, when the DESIGN *o-option* is specified, the MODEL statement is not required to have an equal sign. When no MODEL statement equal sign is specified, all variables are considered independent variables, all options that require dependent variables are ignored, and the IREPLACE *o-option* is automatically specified for you.

You can use DESIGN=*n* for coding very large data sets, where *n* is the number of observations to code at one time. For example, to code a data set with a large number of observations, you can specify DESIGN=100 or DESIGN=1000 to process the data set in blocks of 100 or 1000 observations. If you specify the DESIGN *o-option* rather than DESIGN=*n*, PROC TRANSREG tries to process all observations at once, which might not work with very large data sets. Specify the NOZEROCONSTANT *a-option* with DESIGN=*n* to ensure that constant variables within blocks are not zeroed. See the sections "Using the DESIGN Output Option" on page 10722 and "Discrete Choice Experiments: DESIGN, NORESTORE, NOZERO" on page 10726 for more information about the DESIGN option.

DEPENDENT=*name*

DEP=*name*

specifies the untransformed dependent variable for OUT= data sets with METHOD=MORALS when there is more than one dependent variable. The default is DEPENDENT=_DEPEND_.

DREPLACE

DRE

replaces the original dependent variables with the transformed dependent variables in the OUT= data set. The names of the transformed variables in the OUT= data set correspond to the names of the original dependent variables in the input data set. By default, both the original dependent variables and the transformed dependent variables (with names constructed from the TDPREFIX= (default T) *o-option* and the original dependent variable names) are included in the OUT= data set.

IAPPROXIMATIONS

IAP

outputs the approximations of the transformed independent variables to the OUT= data set. These are the target values for the optimal transformations. The names of the approximation variables are constructed from the AIPREFIX= *o-option* (default A) and the original independent variable names. When you specify the AIPREFIX= *o-option*, the IAPPROXIMATIONS *o-option* is automatically specified for you. The IAPPROXIMATIONS *o-option* is not valid when METHOD=UNIVARIATE.

IREPLACE

IRE

replaces the original independent variables with the transformed independent variables in the OUT= data set. The names of the transformed variables in the OUT= data set correspond to the names of the original independent variables in the input data set. By default, both the original independent variables and the transformed independent variables (with names constructed from the TIPREFIX= *o-option* (default T) and the original independent variable names) are included in the OUT= data set.

LEVERAGE<=*name*>

LEV<=*name*>

creates a variable with the specified name in the OUT= data set that contains leverages. Specifying the LEVERAGE *o-option* is equivalent to specifying LEVERAGE=Leverage.

LILPREFIX=*name*

LIL=*name*

specifies a prefix for naming the liberal-individual-lower confidence limits. The default prefix is LIL. When you specify the LILPREFIX= *o-option*, the CLI *o-option* is automatically specified for you.

LIUPREFIX=*name*

LIU=*name*

specifies a prefix for naming the liberal-individual-upper confidence limits. The default prefix is LIU. When you specify the LIUPREFIX= *o-option*, the CLI *o-option* is automatically specified for you.

LMLPREFIX=*name*

LML=*name*

specifies a prefix for naming the liberal-mean-lower confidence limits. The default prefix is LML. When you specify the LMLPREFIX= *o-option*, the CLM *o-option* is automatically specified for you.

LMUPREFIX=*name*

LMU=*name*

specifies a prefix for naming the liberal-mean-upper confidence limits. The default prefix is LMU. When you specify the LMUPREFIX= *o-option*, the CLM *o-option* is automatically specified for you.

MACRO(*keyword=name...*)

MAC(*keyword=name...*)

creates macro variables. Most of the options available within the MACRO *o-option* are rarely needed. By default, PROC TRANSREG creates a macro variable named _TrgInd with a complete list of independent variables created by the procedure. When PROC TRANSREG is being used for design matrix creation prior to running a procedure without a CLASS statement, this macro provides a convenient way to use the results from PROC TRANSREG. For example, a PROC LOGISTIC step that uses a design matrix coded by PROC TRANSREG can use the following MODEL statement:

```
model y=&_trgind;
```

PROC TRANSREG, also by default, creates a macro variable named _TrgIndN, which contains the number of variables in the _TrgInd list. These macro variables can be used in an ARRAY statement as follows:

```
array indvars[&_trgindn] &_trgind;
```

See the sections "Using the DESIGN Output Option" on page 10722 and "Discrete Choice Experiments: DESIGN, NORESTORE, NOZERO" on page 10726 for examples of using the default macro variables.

The available *keywords* are as follows.

DN=*name* specifies the name of a macro variable that contains the number of dependent variables. By default, a macro variable named _TrgDepN is created. This is the number of variables in the DL= list and the number of macro variables created by the DV= and DE= specifications.

IN=*name* specifies the name of a macro variable that contains the number of independent variables. By default, a macro variable named _TrgIndN is created. This is the number of variables in the IL= list and the number of macro variables created by the IV= and IE= specifications.

DL=*name* specifies the name of a macro variable that contains the list of the dependent variables. By default, a macro variable named _TrgDep is created. These are the variable names of the final transformed variables in the OUT= data set. For example, if there are three dependent variables, y1–y3, then _TrgDep contains, by default, Ty1 Ty2 Ty3 (or y1 y2 y3 if you specify the REPLACE *o-option*).

IL=*name* specifies the name of a macro variable that contains the list of the independent variables. By default, a macro variable named _TrgInd is created. These are the variable names of the final transformed variables in the OUT= data set. For example, if there are three independent variables, x1–x3, then _TrgInd contains, by default, Tx1 Tx2 Tx3 (or x1 x2 x3 if you specify the REPLACE *o-option*).

DV=*prefix* specifies a prefix for creating a list of macro variables, each of which contains one dependent variable name. For example, if there are three dependent variables, y1–y3, and you specify **`macro(dv=Dep)`**, then three macro variables, Dep1, Dep2, and Dep3, are created, containing Ty1, Ty2, and Ty3, respectively (or y1, y2, and y3 if you specify the REPLACE *o-option*). By default, no list is created.

IV=*prefix* specifies a prefix for creating a list of macro variables, each of which contains one independent variable name. For example, if there are three independent variables, x1–x3, and you specify **`macro(iv=Ind)`**, then three macro variables, Ind1, Ind2, and Ind3, are created, containing Tx1, Tx2, and TX3, respectively (or x1, x2, and x3 if you specify the REPLACE *o-option*). By default, no list is created.

DE=*prefix* specifies a prefix for creating a list of macro variables, each of which contains one dependent variable effect. This list shows the origin of each model term. Let *v* be the maximum variable length (for example, 32 for VALIDVARNAME=V7). Each effect consists of two or more parts, and each part consists of a value in *v* columns followed by a blank. If you specify **`macro(de=d)`**, then a macro variable d1 is created for **`identity(y)`**. The d1 macro variable is shown next, wrapped onto two lines:

```
4                                TY
IDENTITY                         Y
```

The first part is the number of parts (4), the second part is the transformed variable name, the third part is the transformation, and the last part is the input variable name. By default, no list is created.

IE=*prefix*
specifies a prefix for creating a list of macro variables, each of which contains one independent variable effect. This list shows the origin of each model term. Let *v* be the maximum variable length (for example, 32 for VALIDVARNAME=V7). Each effect consists of two or more parts, and each part consists of a value in *v* columns followed by a blank. If you specify **macro(ie=I)**, then three macro variables, I1, I2, and I3, are created for **class(x1 | x2)** when both x1 and x2 have values of 1 and 2. These macro variables are shown next, with extra white space removed:

```
5     Tx11     CLASS    x1   1
5     Tx21     CLASS    x2   1
8     Tx11x21  CLASS    x1   1      CLASS    x2   1
```

For CLASS variables, the formatted level appears after the variable name. The first two effects are the main effects, and the last is the interaction term. By default, no list is created.

MEANS
MEA
outputs marginal means for CLASS variable expansions to the OUT= data set.

MEC
outputs multiple regression elliptical point model coordinates to the OUT= data set.

MPC
outputs multiple regression point model coordinates to the OUT= data set.

MQC
outputs multiple regression quadratic point model coordinates to the OUT= data set.

MRC
outputs multiple regression coefficients to the OUT= data set.

MREDUNDANCY
MRE
outputs multiple redundancy analysis coefficients to the OUT= data set.

NORESTOREMISSING
NORESTORE
NOR
specifies that missing values should not be restored when the OUT= data set is created. By default, the coded CLASS variable contains a row of missing values for observations in which the CLASS variable is missing. When you specify the NORESTOREMISSING *o-option*, these observations contain a row of zeros instead. This is useful when PROC TRANSREG is used to code experimental designs for discrete choice models and there is a constant alternative indicated by a missing value.

NOSCORES

NOS

excludes original variables, transformed variables, predicted values, residuals, and scores from the OUT= data set. You can use the NOSCORES *o-option* with various other options to create an OUT= data set that contains only a coefficient partition (for example, a data set consisting entirely of coefficients and coordinates).

PREDICTED

PRE

P

outputs predicted values, which for METHOD=UNIVARIATE and METHOD=MORALS are the ordinary predicted values from the linear model, to the OUT= data set. The names of the predicted values' variables are constructed from the PPREFIX= *o-option* (default P) and the original dependent variable names. When you specify the PPREFIX= *o-option*, the PREDICTED *o-option* is automatically specified for you.

PPREFIX=*name*

PDPREFIX=*name*

PDP=*name*

specifies a prefix for naming the dependent variable predicted values. The default is PPREFIX=P when you specify the PREDICTED *o-option*; otherwise, it is PPREFIX=A. When you specify the PPREFIX= *o-option*, the PREDICTED *o-option* is automatically specified for you. The PPREFIX= *o-option* is the same as the ADPREFIX= *o-option*.

RDPREFIX=*name*

RDP=*name*

specifies a prefix for naming the residual (dependent) variables to the OUT= data set. The default is RDPREFIX=R. When you specify the RDPREFIX= *o-option*, the RESIDUALS *o-option* is automatically specified for you.

REDUNDANCY< =STANDARDIZE | UNSTANDARDIZE >

RED< =STA | UNS >

outputs redundancy variables to the OUT= data set, either standardized or unstandardized. Specifying the REDUNDANCY *o-option* is the same as specifying REDUNDANCY=STANDARDIZE. The results of the REDUNDANCY *o-option* depends on the TSTANDARD= option. You must specify TSTANDARD=Z to get results based on standardized data. The TSTANDARD= option controls how the data that go into the redundancy analysis are scaled, and REDUNDANCY=STANDARDIZE|UNSTANDARDIZE controls how the redundancy variables are scaled. The REDUNDANCY *o-option* is automatically specified for you when you specify the METHOD=REDUNDANCY *a-option*. The RPREFIX= *o-option* specifies a prefix (default Red) for naming the redundancy variables.

REFERENCE=NONE | MISSING | ZERO

REF=NON | MIS | ZER

specifies how reference levels of CLASS variables are to be treated. The options are REFERENCE=NONE, the default, in which reference levels are suppressed; REFERENCE=MISSING, in which reference levels are displayed and output with missing values; and REFERENCE=ZERO, in which reference levels are displayed and output with zeros. You can specify the REFERENCE= option

in the PROC TRANSREG, MODEL, or OUTPUT statement, and you can specify it independently for the OUT= data set and the displayed output. When you specify it in only one statement, it sets the option for both the displayed output and the OUT= data set.

REPLACE

REP

is equivalent to specifying both the DREPLACE and the IREPLACE *o-options*.

RESIDUALS

RES

R

outputs the differences between the transformed dependent variables and their predicted values. The names of the residual variables are constructed from the RDPREFIX= *o-option* (default R) and the original dependent variable names.

RPREFIX=*name*

RPR=*name*

provides a prefix for naming the redundancy variables. The default is RPREFIX=Red. When you specify the RPREFIX= *o-option*, the REDUNDANCY *o-option* is automatically specified for you.

TDPREFIX=*name*

TDP=*name*

specifies a prefix for naming the transformed dependent variables. By default, TDPREFIX=T. The TDPREFIX= *o-option* is ignored when you specify the DREPLACE *o-option*.

TIPREFIX=*name*

TIP=*name*

specifies a prefix for naming the transformed independent variables. By default, TIPREFIX=T. The TIPREFIX= *o-option* is ignored when you specify the IREPLACE *o-option*.

WEIGHT Statement

WEIGHT *variable* **;**

When you use a WEIGHT statement, a weighted residual sum of squares is minimized. The WEIGHT statement has no effect on degrees of freedom or number of observations, but the weights affect most other calculations. The observation is used in the analysis only if the value of the WEIGHT statement variable is greater than 0.

Details: TRANSREG Procedure

Model Statement Usage

MODEL < *transform*(*dependents* < / *t-options* >) >
< *transform*(*dependents* < / *t-options* >)**. . . = >**
transform(*independents* < / *t-options* >)
< *transform*(*independents* < / *t-options* >)**. . .** > < / *a-options* > **;**

Here are some examples of model statements:

- linear regression

```
model identity(y) = identity(x);
```

- a linear model with a nonlinear regression function

```
model identity(y) = spline(x / nknots=5);
```

- multiple regression

```
model identity(y) = identity(x1-x5);
```

- multiple regression with nonlinear transformations

```
model spline(y / nknots=3) = spline(x1-x5 / nknots=3);
```

- multiple regression with nonlinear but monotone transformations

```
model mspline(y / nknots=3) = mspline(x1-x5 / nknots=3);
```

- multivariate multiple regression

```
model identity(y1-y4) = identity(x1-x5);
```

- canonical correlation

```
model identity(y1-y4) = identity(x1-x5) / method=canals;
```

- redundancy analysis

```
model identity(y1-y4) = identity(x1-x5) / method=redundancy;
```

- preference mapping, vector model (Carroll 1972)

```
model identity(Attrib1-Attrib3) = identity(Dim1-Dim2);
```

- preference mapping, ideal point model (Carroll 1972)

```
model identity(Attrib1-Attrib3) = point(Dim1-Dim2);
```

- preference mapping, ideal point model, elliptical (Carroll 1972)

```
model identity(Attrib1-Attrib3) = epoint(Dim1-Dim2);
```

- preference mapping, ideal point model, quadratic (Carroll 1972)

```
model identity(Attrib1-Attrib3) = qpoint(Dim1-Dim2);
```

- metric conjoint analysis

```
model identity(Subj1-Subj50) = class(a b c d e f / zero=sum);
```

- nonmetric conjoint analysis

```
model monotone(Subj1-Subj50) = class(a b c d e f / zero=sum);
```

- main effects, two-way interaction

```
model identity(y) = class(a|b);
```

- less-than-full-rank model—main effects and two-way interaction are constrained to sum to zero

```
model identity(y) = class(a|b / zero=sum);
```

- main effects and all two-way interactions

```
model identity(y) = class(a|b|c@2);
```

- main effects and all two- and three-way interactions

```
model identity(y) = class(a|b|c);
```

- main effects and only the b*c two-way interaction

```
model identity(y) = class(a b c b*c);
```

- seven main effects, three two-way interactions

```
model identity(y) = class(a b c d e f g a*b a*c a*d);
```

- deviations-from-means (effects or $(1, 0, -1)$) coding, with an a reference level of '1' and a b reference level of '2'

```
model identity(y) = class(a|b / deviations zero='1' '2');
```

- cell-means coding (implicit intercept)

```
model identity(y) = class(a*b / zero=none);
```

- reference cell model

```
model identity(y) = class(a|b / zero='1' '1');
```

- reference line with change in line parameters

```
model identity(y) = class(a) | identity(x);
```

- reference curve with change in curve parameters

```
model identity(y) = class(a) | spline(x);
```

- separate curves and intercepts

```
model identity(y) = class(a / zero=none) | spline(x);
```

- quantitative effects with interaction

```
model identity(y) = identity(x1 | x2);
```

- separate quantitative effects with interaction within each cell

```
model identity(y) = class(a * b / zero=none) | identity(x1 | x2);
```

Box-Cox Transformations

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/tregbox.sas.)

Box-Cox (1964) transformations are used to find potentially nonlinear transformations of a dependent variable. The Box-Cox transformation has the form

$$\begin{array}{ll} (y^{\lambda}-1)/\lambda & \lambda \neq 0 \\ \log(y) & \lambda = 0 \end{array}$$

This family of transformations of the positive dependent variable y is controlled by the parameter λ. Transformations linearly related to square root, inverse, quadratic, cubic, and so on are all special cases. The limit as λ approaches 0 is the log transformation. More generally, Box-Cox transformations of the following form can be fit:

$$\begin{array}{ll} ((y+c)^{\lambda}-1)/(\lambda g) & \lambda \neq 0 \\ \log(y+c)/g & \lambda = 0 \end{array}$$

By default, $c = 0$. The parameter c can be used to rescale y so that it is strictly positive. By default, $g = 1$. Alternatively, g can be $\dot{y}^{\lambda-1}$, where $\dot{y}$ is the geometric mean of y.

The BOXCOX transformation in PROC TRANSREG can be used to perform a Box-Cox transformation of the dependent variable. You can specify a list of power parameters by using the LAMBDA= *t-option*. By default, LAMBDA=–3 TO 3 BY 0.25. The procedure chooses the optimal power parameter by using a maximum likelihood criterion (Draper and Smith 1981, pp. 225–226). You can specify the PARAMETER=*c* transformation option when you want to shift the values of y, usually to avoid negatives. To divide by $\dot{y}^{\lambda-1}$, specify the GEOMETRICMEAN *t-option*.

Here are three examples of using the LAMBDA= *t-option*:

```
model BoxCox(y / lambda=0) = identity(x1-x5);
model BoxCox(y / lambda=-2 to 2 by 0.1) = identity(x1-x5);
model BoxCox(y) = identity(x1-x5);
```

Here is the first example:

```
model BoxCox(y / lambda=0) = identity(x1-x5);
```

LAMBDA=0 specifies a Box-Cox transformation with a power parameter of 0. Since a single value of 0 was specified for LAMBDA=, there is no difference between the following models:

```
model BoxCox(y / lambda=0) = identity(x1-x5);
model log(y) = identity(x1-x5);
```

Here is the second example:

```
model BoxCox(y / lambda=-2 to 2 by 0.1) = identity(x1-x5);
```

LAMBDA= specifies a list of power parameters. PROC TRANSREG tries each power parameter in the list and picks the best transformation. A maximum likelihood approach (Draper and Smith 1981, pp. 225–226) is used. With Box-Cox transformations, PROC TRANSREG finds the transformation before the usual iterations begin. Note that this is quite different from PROC TRANSREG's usual approach of iteratively finding optimal transformations with ordinary and alternating least squares. It is analogous to SMOOTH and PBSPLINE, which also find transformations before the iterations begin based on a criterion other than least squares.

Here is the third example:

```
model BoxCox(y) = identity(x1-x5);
```

The default LAMBDA= list of –3 TO 3 BY 0.25 is used.

The procedure prints the optimal power parameter, a confidence interval on the power parameter (based on the ALPHA= *t-option*), a "convenient" power parameter (selected from the CLL= *t-option* list), and the log likelihood for each power parameter tried (see Example 126.2).

To illustrate how Box-Cox transformations work, data were generated from the model

$$y = e^{x+\epsilon}$$

where $\epsilon \sim \mathrm{N}(0, 1)$. The transformed data can be fit with a linear model

$$\log(y) = x + \epsilon$$

The following statements produce Figure 126.14 through Figure 126.15:

```
title 'Basic Box-Cox Example';

data x;
   do x = 1 to 8 by 0.025;
      y = exp(x + normal(7));
      output;
   end;
run;

ods graphics on;

title2 'Default Options';

proc transreg data=x test;
   model BoxCox(y) = identity(x);
run;
```

Figure 126.14 Basic Box-Cox Example, Default Output

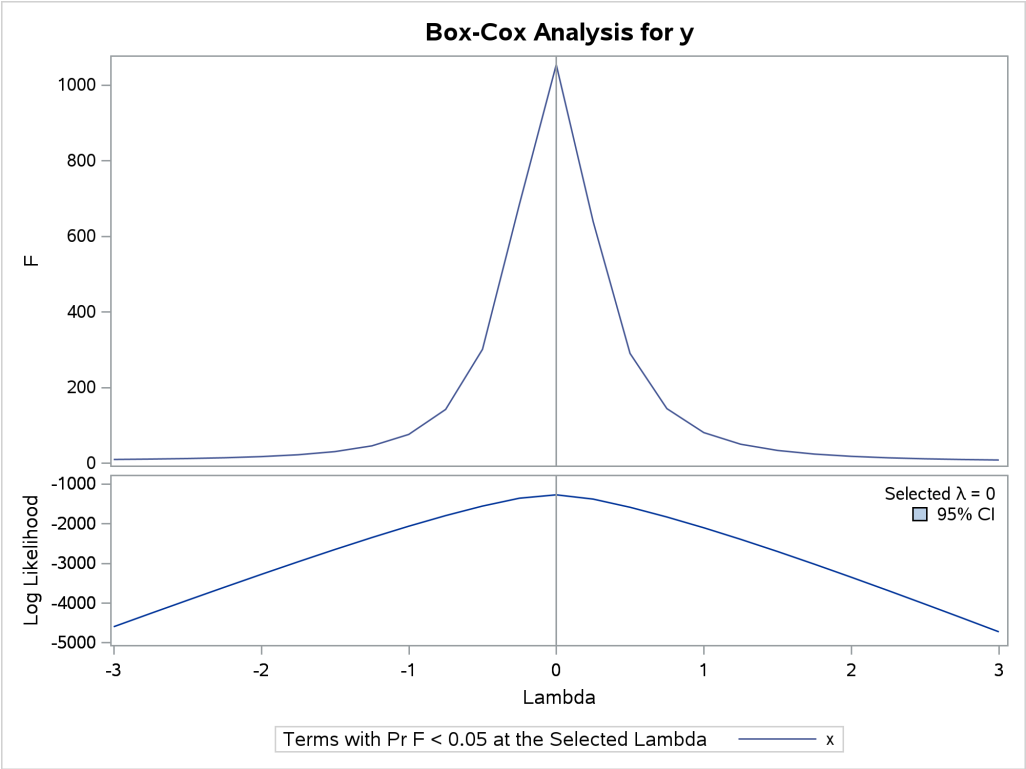

Figure 126.14 shows that PROC TRANSREG correctly selects the log transformation $\lambda = 0$, with a narrow confidence interval. The $F = t^2$ plot shows that F is at its largest in the vicinity of the optimal Box-Cox transformation.

The rest of the output, which contains the ANOVA results, is shown in Figure 126.15.

Figure 126.15 Basic Box-Cox Example, Default Output

Dependent Variable BoxCox(y)

Number of Observations Read	281
Number of Observations Used	281

The TRANSREG Procedure Hypothesis Tests for BoxCox(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Liberal p
Model	1	1145.884	1145.884	1053.66	>= <.0001
Error	279	303.421	1.088		
Corrected Total	280	1449.305			

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Figure 126.15 *continued*

Root MSE	1.04285	**R-Square**	0.7906
Dependent Mean	4.49653	**Adj R-Sq**	0.7899
Coeff Var	23.19225	**Lambda**	0.0000

This next example uses several options. The LAMBDA= *t-option* specifies power parameters sparsely from –2 to –0.5 and 0.5 to 2 just to get the general shape of the log-likelihood function in that region. Between –0.5 and 0.5, more power parameters are tried. The CONVENIENT *t-option* is specified so that if a power parameter like $\lambda = 1$ or $\lambda = 0$ is found in the confidence interval, it is used instead of the optimal power parameter. PARAMETER=2 is specified to add 2 to each y before performing the transformations. ALPHA=0.00001 specifies a wide confidence interval.

These next statements perform the Box-Cox analysis and produce Figure 126.16 and Figure 126.17:

```
title2 'Several Options Demonstrated';

proc transreg data=x ss2 details
              plots=(transformation(dependent) scatter
                     observedbypredicted);
   model BoxCox(y / lambda=-2 -1 -0.5 to 0.5 by 0.05 1 2
                    convenient parameter=2 alpha=0.00001) =
         identity(x);
run;
```

Figure 126.16 Basic Box-Cox Example, Several Options Demonstrated

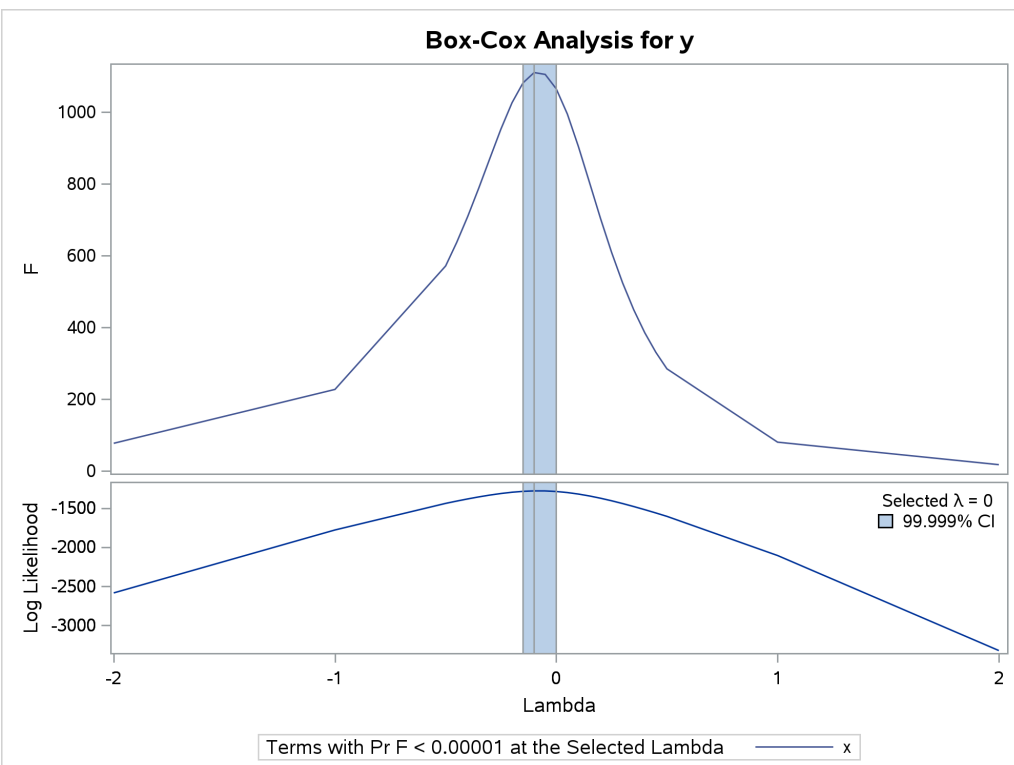

The results in Figure 126.16 and Figure 126.17 show that the optimal power parameter is –0.1, but 0 is in

the confidence interval, and hence a log transformation is chosen. The actual Box-Cox transformation, the original scatter plot, and observed by predicted values plot are shown in Figure 126.17.

Figure 126.17 Basic Box-Cox Example, Several Options Demonstrated

Dependent Variable BoxCox(y)

Number of Observations Read	281
Number of Observations Used	281

Model Statement Specification Details

Type	DF	Variable	Description	Value
Dep	1	BoxCox(y)	Lambda Used	0
			Lambda	-0.1
			Log Likelihood	-1280.1
			Conv. Lambda	0
			Conv. Lambda LL	-1287.7
			CI Limit	-1289.9
			Alpha	0.00001
			Parameter	2
			Options	Convenient Lambda Used
Ind	1	Identity(x)	DF	1

The TRANSREG Procedure Hypothesis Tests for BoxCox(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Liberal p
Model	1	999.438	999.4381	1064.82	>= <.0001
Error	279	261.868	0.9386		
Corrected Total	280	1261.306			

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Root MSE	0.96881	**R-Square**	0.7924
Dependent Mean	4.61429	**Adj R-Sq**	0.7916
Coeff Var	20.99591	**Lambda**	0.0000

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Liberal p
Intercept	1	0.42939328	8.746	8.746	9.32	>= 0.0025
Identity(x)	1	0.92997620	999.438	999.438	1064.82	>= <.0001

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Figure 126.17 *continued*

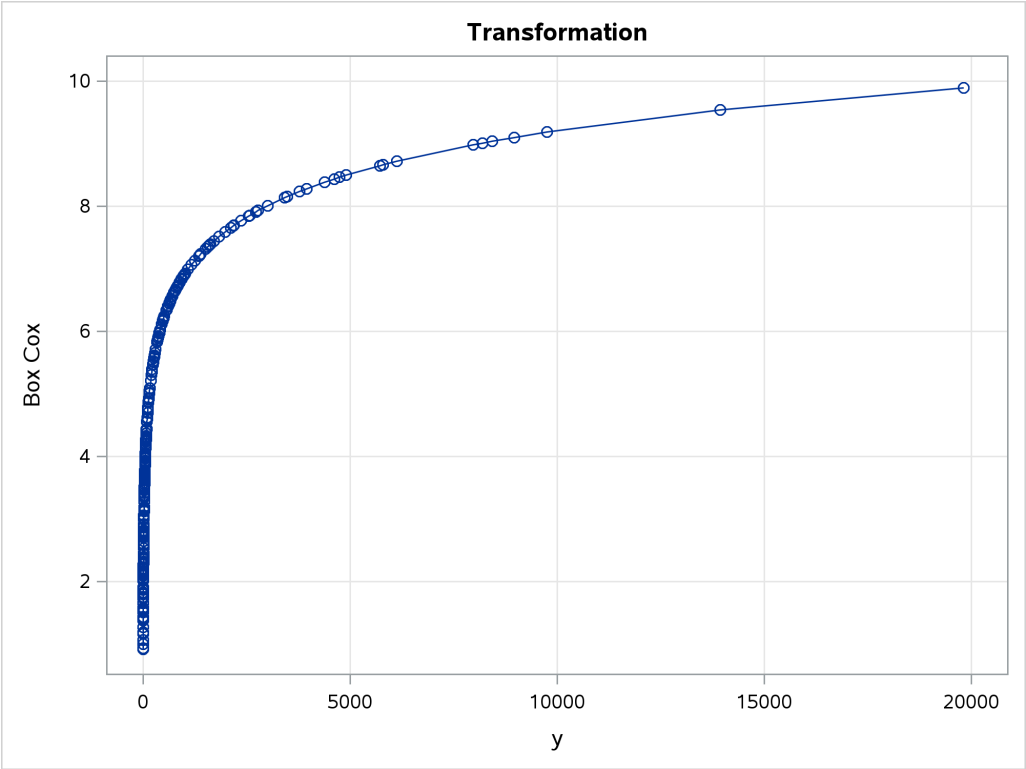

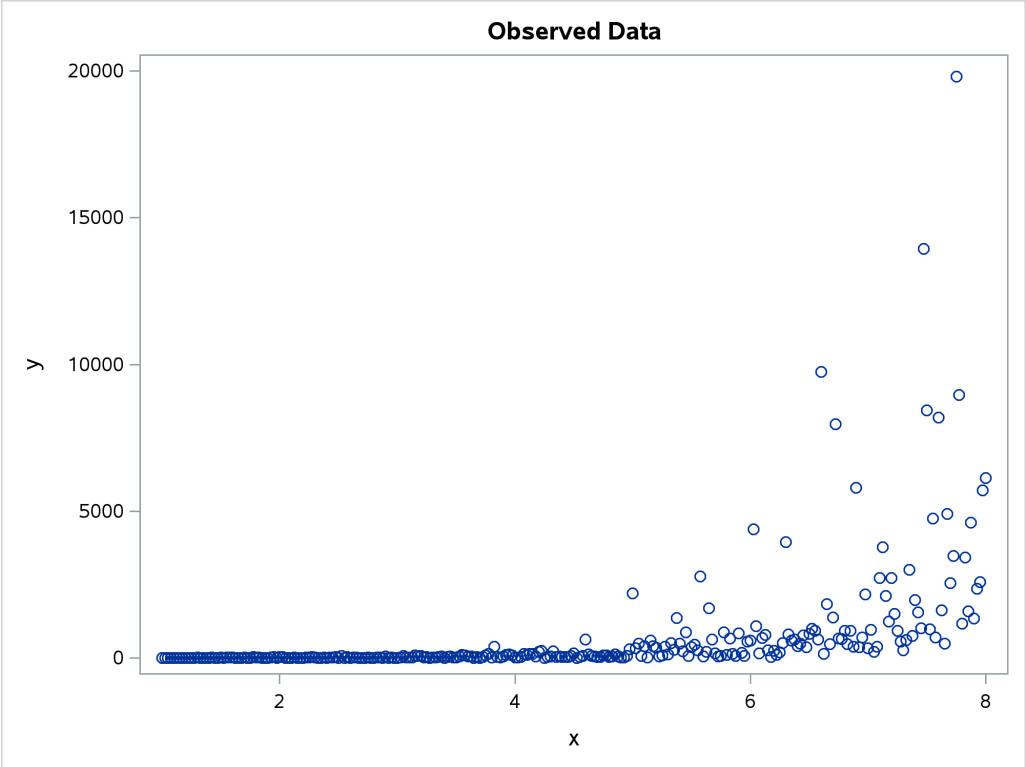

Figure 126.17 *continued*

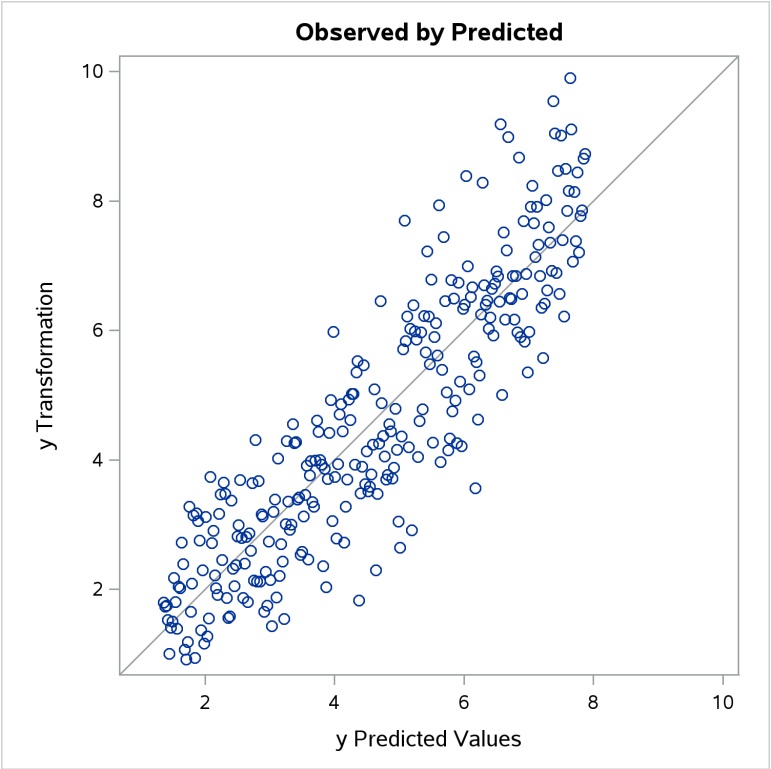

The next example shows how to find a Box-Cox transformation without an independent variable. This seeks to normalize the univariate histogram. This example generates 500 random observations from a lognormal distribution. In addition, a constant variable z is created that is all zero. This is because PROC TRANSREG requires some independent variable to be specified, even if it is constant. Two options are specified in the PROC TRANSREG statement. MAXITER=0 is specified because the Box-Cox transformation is performed before any iterations are begun. No iterations are needed since no other work is required. The NOZEROCONSTANT *a-option* (which can be abbreviated NOZ) is specified so that PROC TRANSREG does not print any warnings when it encounters the constant independent variable. The MODEL statement asks for a Box-Cox transformation of y and an IDENTITY transformation (which does nothing) of the constant variable z. Finally, PROC UNIVARIATE is run to show a histogram of the original variable y, and the Box-Cox transformation, Ty. The following statements fit the univariate Box-Cox model and produce Figure 126.18:

```
title 'Univariate Box-Cox';

data x;
   call streaminit(17);
   z = 0;
   do i = 1 to 500;
      y = rand('lognormal');
      output;
   end;
run;

proc transreg maxiter=0 nozeroconstant;
```

```
   model BoxCox(y) = identity(z);
   output;
run;

proc univariate noprint;
   histogram y ty;
run;
```

The PROC TRANSREG results in Figure 126.18 show that zero is chosen for lambda, so a log transformation is chosen. The first histogram shows that the original data are skewed, but a log transformation makes the data appear much more nearly normal.

Figure 126.18 Box-Cox with No Independent Variable

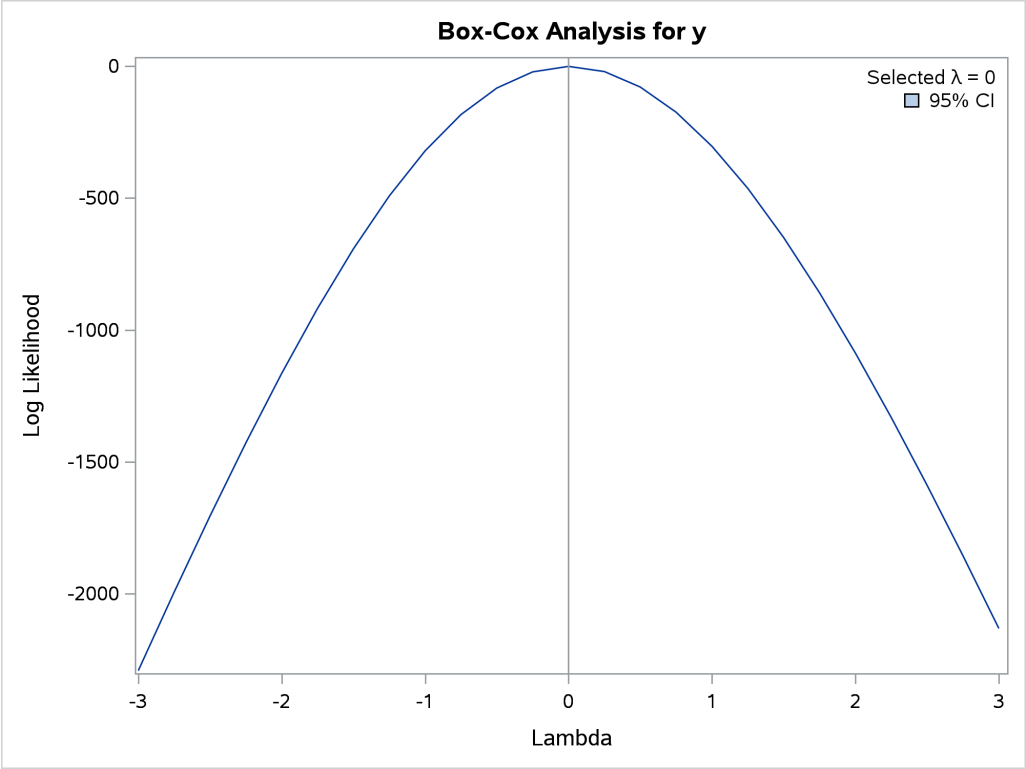

Figure 126.18 *continued*

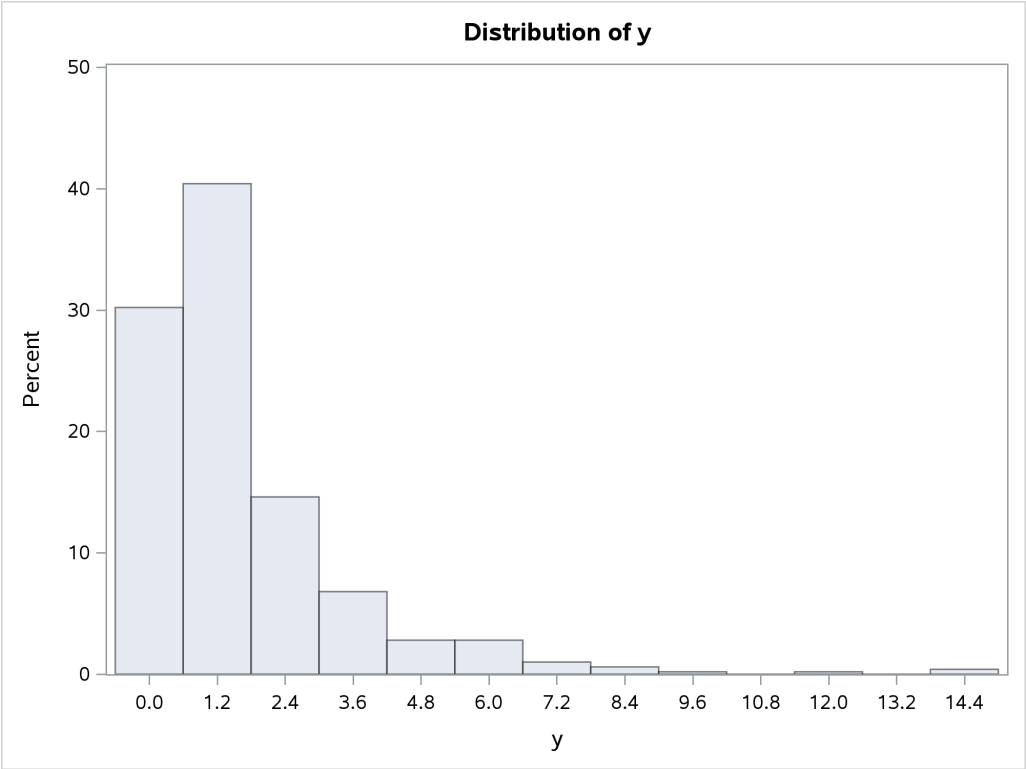

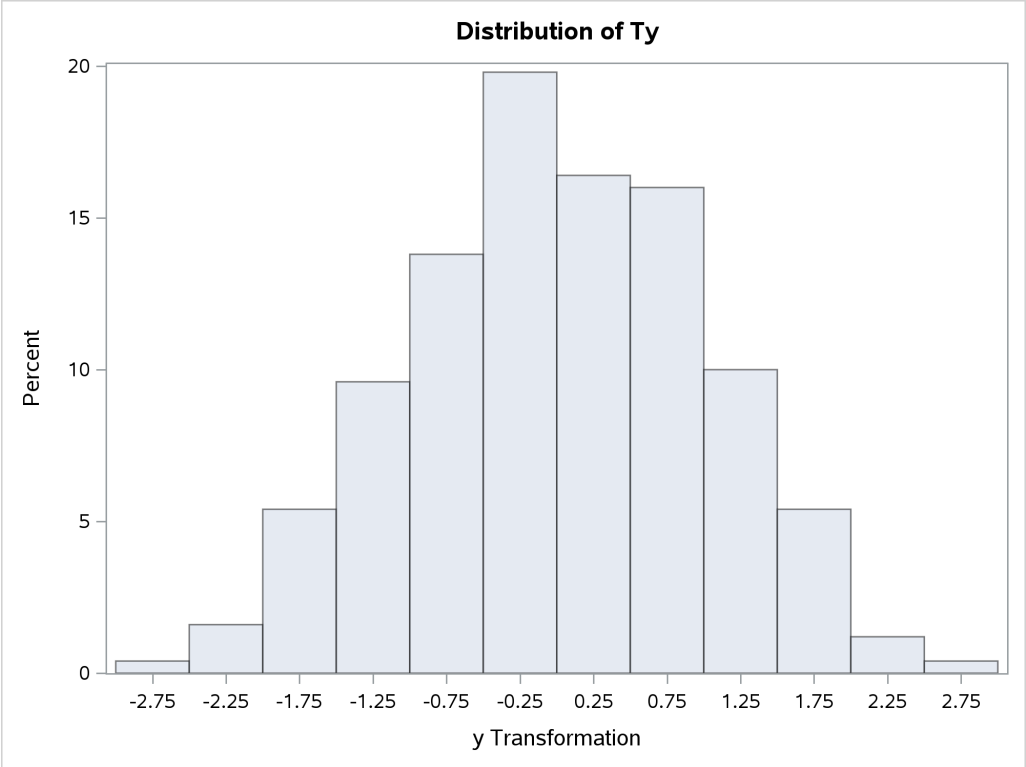

Using Splines and Knots

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/tregspl.sas.)

This section illustrates some properties of splines. *Splines* are curves, and they are usually required to be continuous and smooth. Splines are usually defined as piecewise polynomials of degree n with function values and first $n - 1$ derivatives that agree at the points where they join. The abscissa or X-axis values of the join points are called *knots*. The term "spline" is also used for polynomials (splines with no knots) and piecewise polynomials with more than one discontinuous derivative. Splines with no knots are generally smoother than splines with knots, which are generally smoother than splines with multiple discontinuous derivatives. Splines with few knots are generally smoother than splines with many knots; however, increasing the number of knots usually increases the fit of the spline function to the data. Knots give the curve freedom to bend to more closely follow the data. See Smith (1979) for an excellent introduction to splines.

In this section, an artificial data set is created with a variable y that is a discontinuous function of x. (See Figure 126.20.) Notice that the function has four unconnected parts, each of which is a curve. Notice too that there is an overall quadratic trend—that is, ignoring the shapes of the individual curves, at first the y values tend to decrease as x increases, then y values tend to increase. While these artificial data are clearly not realistic, their distinct pattern helps illustrate how splines work. The following statements create the data set, fit a simple linear regression model, and produce Figure 126.19 through Figure 126.20:

```
title 'An Illustration of Splines and Knots';

* Create in y a discontinuous function of x.;

data a;
   x = -0.000001;
   do i = 0 to 199;
      if mod(i, 50) = 0 then do;
         c = ((x / 2) - 5)**2;
         if i = 150 then c = c + 5;
         y = c;
      end;
      x = x + 0.1;
      y = y - sin(x - c);
      output;
   end;
run;

ods graphics on;

title2 'A Linear Regression Fit';
proc transreg data=a plots=scatter rsquare;
   model identity(y) = identity(x);
run;
```

The R square for the linear regression is 0.1006. The linear fit results in Figure 126.19 show the predicted values of y given x. It can clearly be seen in Figure 126.19 that the linear regression model is not appropriate for these data.

Figure 126.19 A Linear Regression Fit

An Illustration of Splines and Knots
A Linear Regression Fit

The TRANSREG Procedure

The TRANSREG Procedure Hypothesis Tests for Identity(y)

R-Square	0.1006

Figure 126.19 *continued*

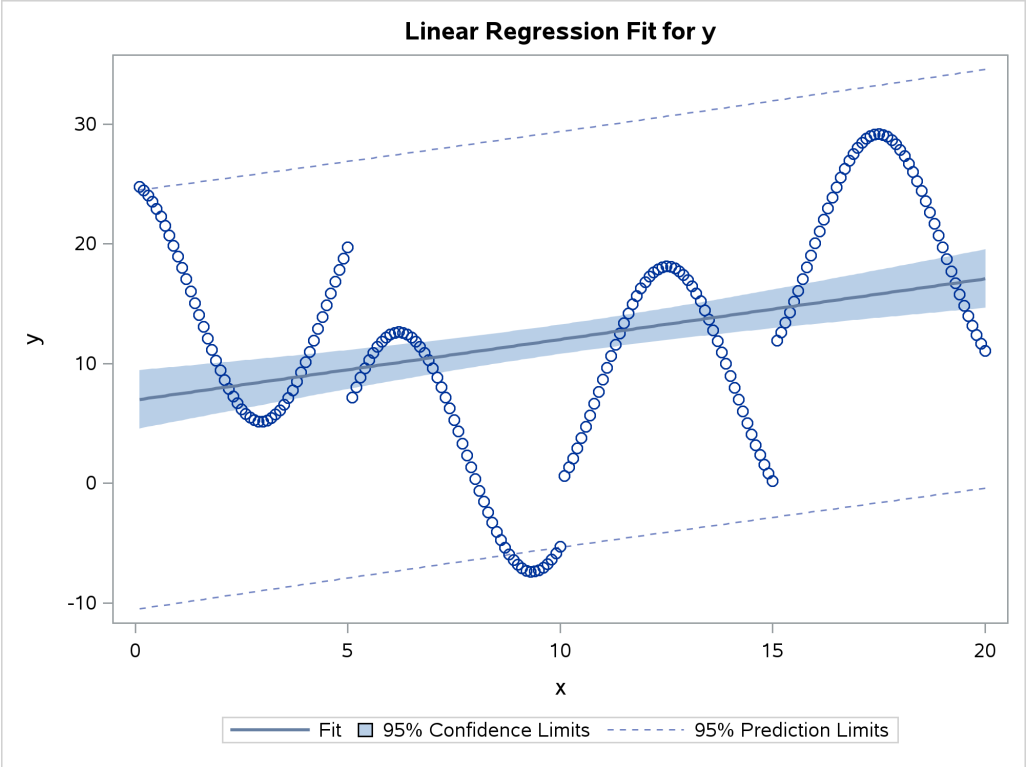

Figure 126.20 The Original Scatter Plot

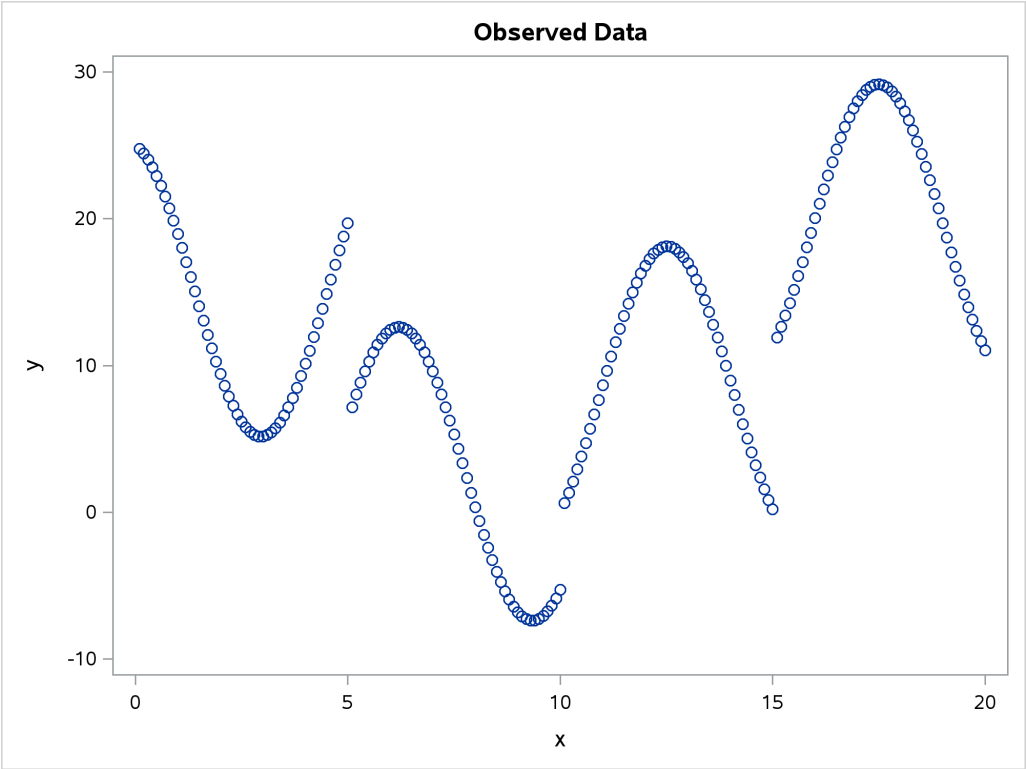

The next PROC TRANSREG step finds a degree-two spline transformation with no knots, which is a quadratic polynomial. The spline is a weighted sum of a single constant, a single straight line, and a single quadratic curve. The following statements perform the quadratic analysis and produce Figure 126.21:

```
title2 'A Quadratic Polynomial Fit';

proc transreg data=A;
   model identity(y)=spline(x / degree=2);
run;
```

The R square in Figure 126.21 increases from 0.10061, which is the linear fit value from before, to 0.40720. The plot shows that the quadratic regression function does not fit any of the individual curves well, but it does follow the overall trend in the data. Since the overall trend is quadratic, if you were to fit a degree-three spline with no knots (not shown) would increase R square by only a small amount.

Figure 126.21 A Quadratic Polynomial Fit

An Illustration of Splines and Knots
A Quadratic Polynomial Fit

The TRANSREG Procedure

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
1	0.82127	2.77121	0.10061		
2	0.00000	0.00000	0.40720	0.30659	Converged

Algorithm converged.

Figure 126.21 *continued*

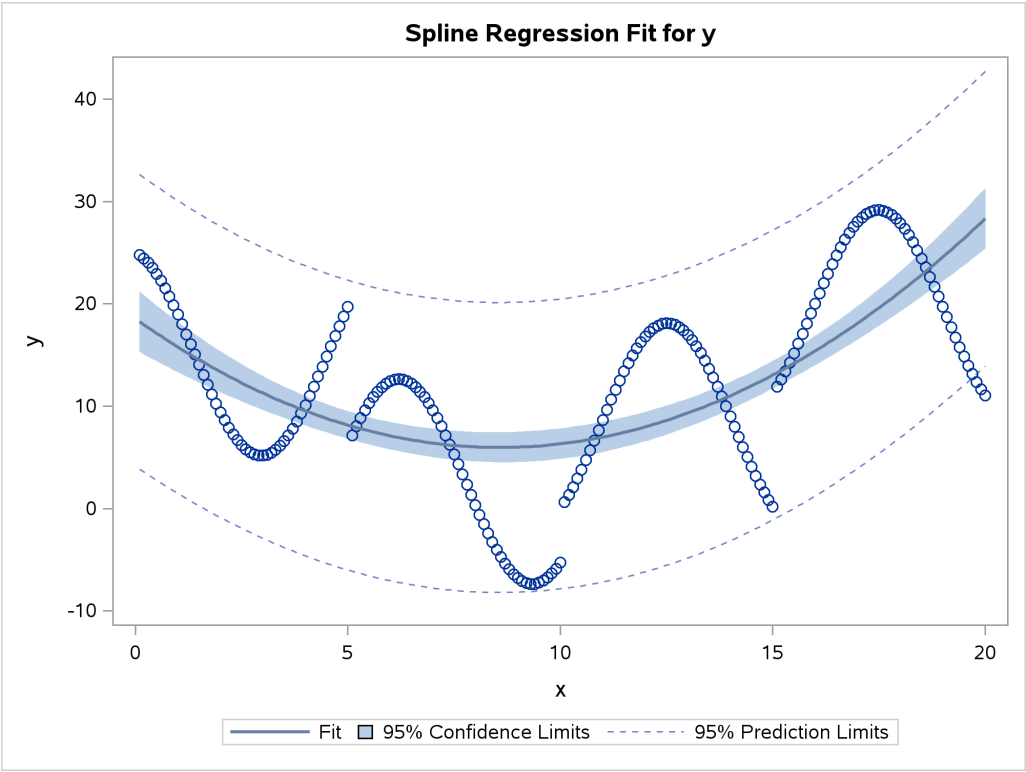

The next step uses the default degree of three, for a piecewise cubic polynomial, and requests knots at the known break points, x=5, 10, and 15. This requests a spline that is continuous, has continuous first and second derivatives, and has a third derivative that is discontinuous at 5, 10, and 15. The spline is a weighted sum of a single constant, a single straight line, a single quadratic curve, a cubic curve for the portion of x less than 5, a different cubic curve for the portion of x between 5 and 10, a different cubic curve for the portion of x between 10 and 15, and another cubic curve for the portion of x greater than 15. The following statements fit the spline model and produce Figure 126.22:

```
title2 'A Cubic Spline Fit with Knots at X=5, 10, 15';

proc transreg data=a;
   model identity(y) = spline(x / knots=5 10 15);
run;
```

The new R square in Figure 126.22 is 0.61730. The plot shows that the spline is less smooth than the quadratic polynomial and follows the data more closely than the quadratic polynomial.

Figure 126.22 A Cubic Spline Fit

An Illustration of Splines and Knots
A Cubic Spline Fit with Knots at X=5, 10, 15

The TRANSREG Procedure

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
1	0.85367	3.88449	0.10061		
2	0.00000	0.00000	0.61730	0.51670	Converged

Algorithm converged.

Figure 126.22 *continued*

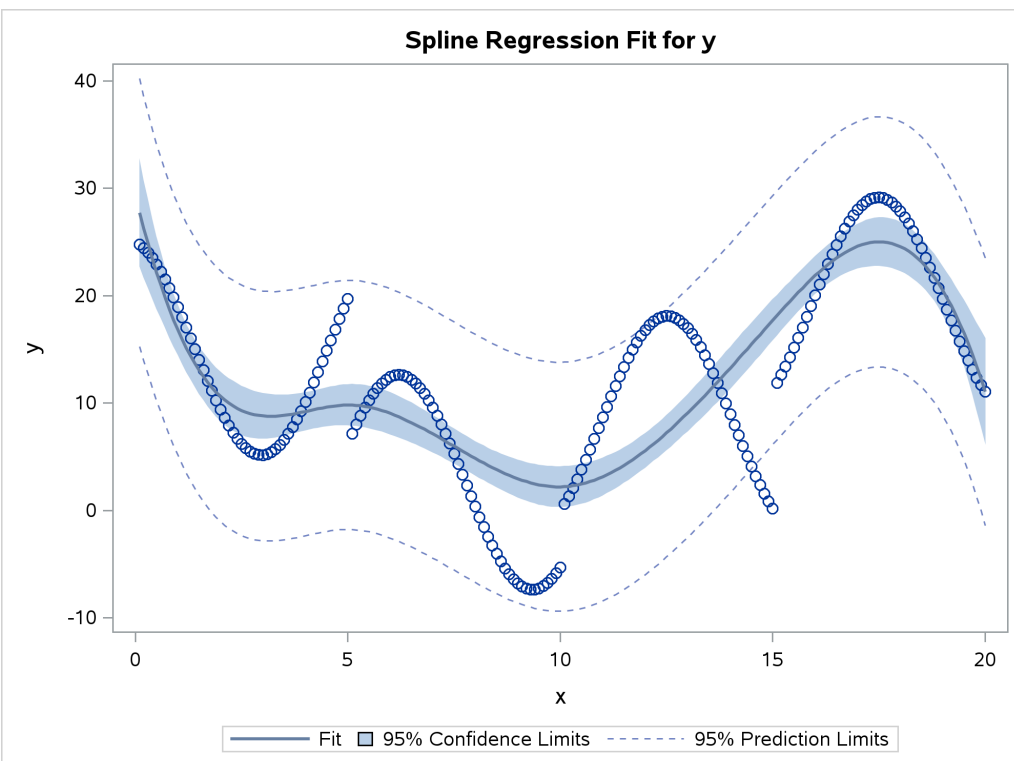

The same model could be fit with a DATA step and PROC REG, as follows:

```
data b;              /* A is the data set used by transreg */
   set a(keep=x y);
   x1=x;                        /* x                       */
   x2=x**2;                     /* x squared               */
   x3=x**3;                     /* x cubed                 */
   x4=(x> 5)*((x-5)**3);        /* change in x**3 after  5 */
   x5=(x>10)*((x-10)**3);       /* change in x**3 after 10 */
   x6=(x>15)*((x-15)**3);       /* change in x**3 after 15 */
run;

proc reg;
   model y=x1-x6;
quit;
```

The output from these previous statements is not displayed. The assignment statements and comments show how you can construct terms that can be used to fit the same model.

In the next step, each knot is repeated three times, so the first, second, and third derivatives are discontinuous at x=5, 10, and 15, but the spline is continuous at the knots. The spline is a weighted sum of the following:

- a single constant
- a line for the portion of x less than 5
- a quadratic curve for the portion of x less than 5
- a cubic curve for the portion of x less than 5
- a different line for the portion of x between 5 and 10
- a different quadratic curve for the portion of x between 5 and 10
- a different cubic curve for the portion of x between 5 and 10
- a different line for the portion of x between 10 and 15
- a different quadratic curve for the portion of x between 10 and 15
- a different cubic curve for the portion of x between 10 and 15
- another line for the portion of x greater than 15
- another quadratic curve for the portion of x greater than 15
- another cubic curve for the portion of x greater than 15

The spline is continuous since there is not a separate constant or separate intercept in the formula for the spline for each knot. The following statements perform this analysis and produce Figure 126.23:

```
title3 'First - Third Derivatives Discontinuous at X=5, 10, 15';

proc transreg data=a;
   model identity(y) = spline(x / knots=5 5 5 10 10 10 15 15 15);
run;
```

Now the R square in Figure 126.23 is 0.95542, and the spline closely follows the data, except at the knots.

Figure 126.23 Spline with Discontinuous Derivatives

An Illustration of Splines and Knots
A Cubic Spline Fit with Knots at X=5, 10, 15
First - Third Derivatives Discontinuous at X=5, 10, 15

The TRANSREG Procedure

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
1	0.92492	3.50038	0.10061		
2	0.00000	0.00000	0.95542	0.85481	Converged

Algorithm converged.

Figure 126.23 *continued*

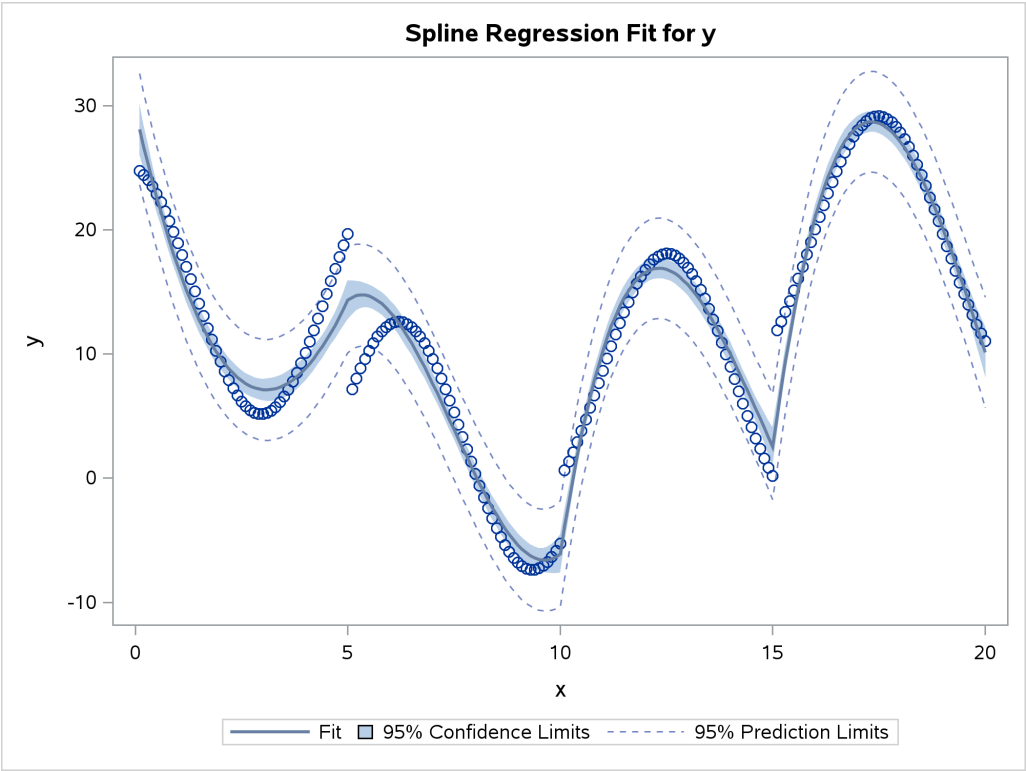

The same model could be fit with a DATA step and PROC REG, as follows:

```
data b;
   set a(keep=x y);
   x1=x;                        /* x                        */
   x2=x**2;                     /* x squared                */
   x3=x**3;                     /* x cubed                  */
   x4=(x>5)   * (x- 5);         /* change in x    after  5 */
   x5=(x>10)  * (x-10);         /* change in x    after 10 */
```

```
   x6=(x>15)  * (x-15);         /* change in x    after 15 */
   x7=(x>5)   * ((x-5)**2);     /* change in x**2 after  5 */
   x8=(x>10)  * ((x-10)**2);    /* change in x**2 after 10 */
   x9=(x>15)  * ((x-15)**2);    /* change in x**2 after 15 */
   x10=(x>5)  * ((x-5)**3);     /* change in x**3 after  5 */
   x11=(x>10) * ((x-10)**3);    /* change in x**3 after 10 */
   x12=(x>15) * ((x-15)**3);    /* change in x**3 after 15 */
run;

proc reg;
   model y=x1-x12;
quit;
```

The output from these previous statements is not displayed. The assignment statements and comments show how you can construct terms that can be used to fit the same model.

Each knot is repeated four times in the next step. Now the spline function is discontinuous at the knots, and it can follow the data more closely. The following statements perform this analysis and produce Figure 126.24:

```
title3 'Discontinuous Function and Derivatives';

proc transreg data=a;
   model identity(y) = spline(x / knots=5 5 5 5 10 10 10 10
                                        15 15 15 15);
run;
```

Now the R square in Figure 126.24 is 0.99254. In this step, each separate curve is approximated by a cubic polynomial (with no knots within the separate polynomials). (Note, however, that the separate functions are connected in the plot, because PROC TRANSREG cannot currently produce separate functions for a model like this. Usually, you would use a CLASS variable to get separate functions.)

Figure 126.24 Discontinuous Spline Fit

An Illustration of Splines and Knots
A Cubic Spline Fit with Knots at X=5, 10, 15
Discontinuous Function and Derivatives

The TRANSREG Procedure

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
1	0.90271	3.29184	0.10061		
2	0.00000	0.00000	0.99254	0.89193	Converged

Algorithm converged.

Figure 126.24 *continued*

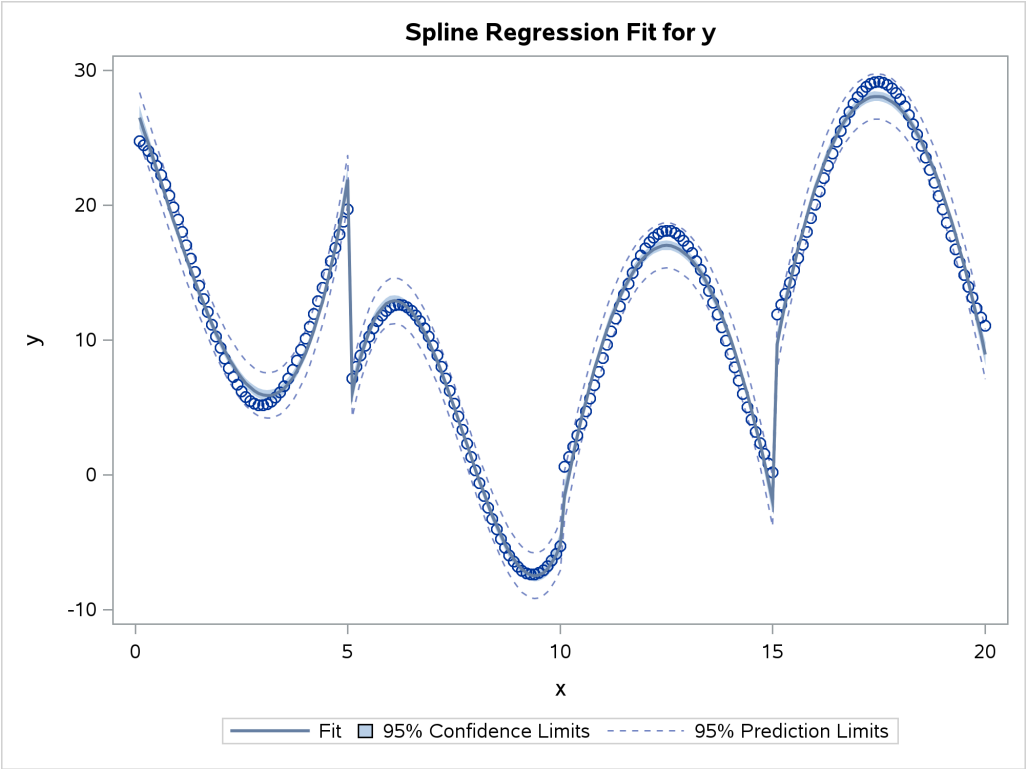

To solve this problem with a DATA step and PROC REG, you would need to create all of the variables in the preceding DATA step (the B data set for the piecewise polynomial with discontinuous third derivatives), plus the following three variables:

```
x13=(x >  5);   /* intercept change after  5 */
x14=(x > 10);   /* intercept change after 10 */
x15=(x > 15);   /* intercept change after 15 */
```

The next two examples use the NKNOTS= *t-option* to specify the number of knots but not their location. NKNOTS=4 places knots at the quintiles, whereas NKNOTS=9 places knots at the deciles. The spline and its first two derivatives are continuous. The following statements produce Figure 126.25 and Figure 126.26:

```
title3 'Four Knots';

proc transreg data=a;
   model identity(y) = spline(x / nknots=4);
run;

title3 'Nine Knots';

proc transreg data=a;
   model identity(y) = spline(x / nknots=9);
run;
```

The R-square values displayed in Figure 126.25 and Figure 126.26 are 0.74450 and 0.95256, respectively. Even though the knots are not optimally placed, the spline can closely follow the data with NKNOTS=9.

Figure 126.25 Spline Fit with Knots at the Quintiles

An Illustration of Splines and Knots
A Cubic Spline Fit with Knots at X=5, 10, 15
Four Knots

The TRANSREG Procedure

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
1	0.90305	4.46027	0.10061		
2	0.00000	0.00000	0.74450	0.64389	Converged

Algorithm converged.

Figure 126.25 *continued*

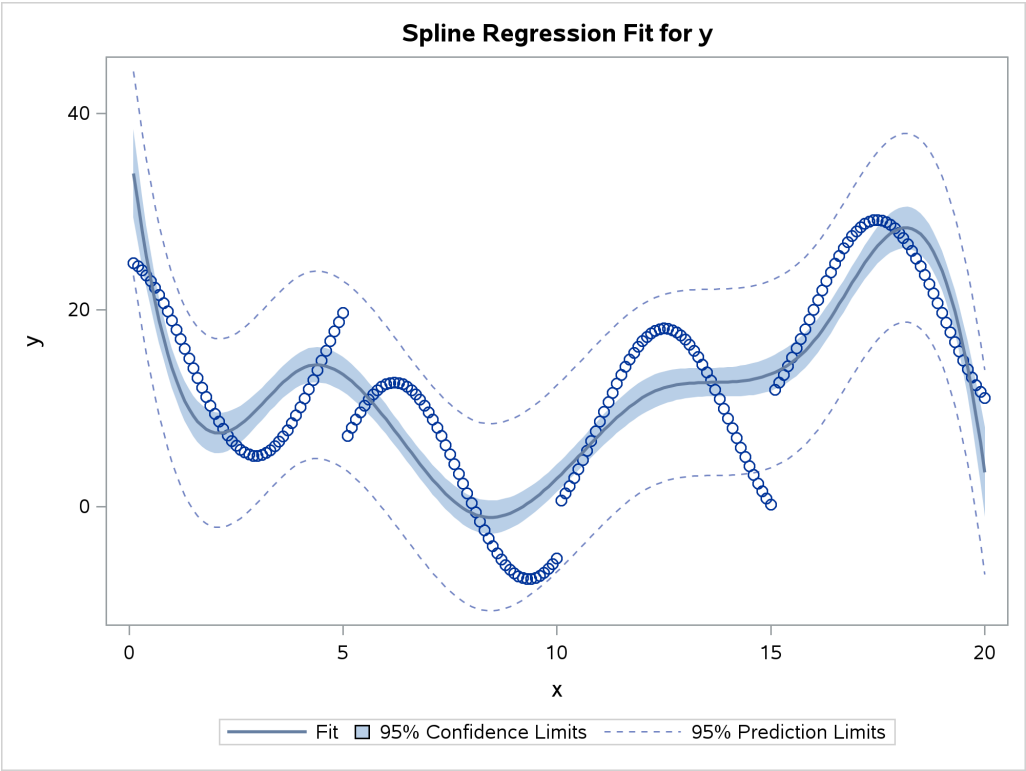

Figure 126.26 Spline Fit with Knots at the Deciles

An Illustration of Splines and Knots
A Cubic Spline Fit with Knots at X=5, 10, 15
Nine Knots

The TRANSREG Procedure

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
1	0.94832	3.03488	0.10061		
2	0.00000	0.00000	0.95256	0.85196	Converged

Algorithm converged.

Figure 126.26 *continued*

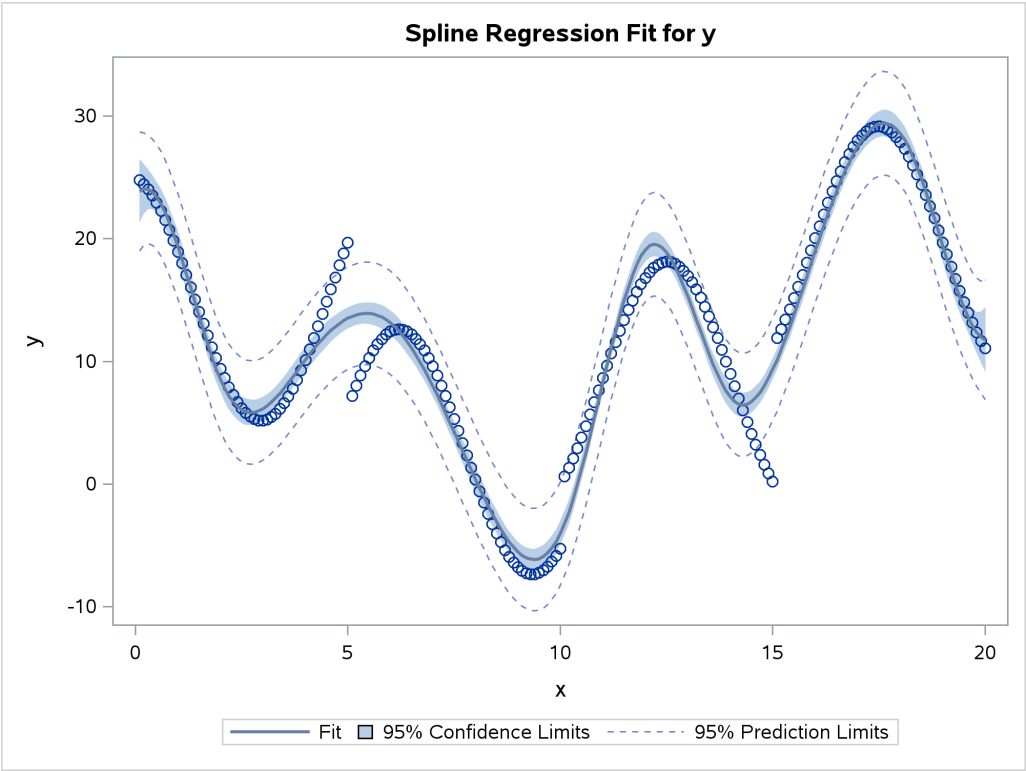

Scoring Spline Variables

This section shows you how to find spline transformations of variables in one data set and apply the same transformations to variables in another data set. This is illustrated with artificial data. In these data sets, the variable y is approximately a linear function of nonlinear transformations of the variables x, w, and z. The model is fit using data set X, and those results are used to score data set Z. The following statements create the two data sets:

```
title 'An Illustration of Splines and Knots';
title2 'Scoring Spline Variables';

data x;
   do i = 1 to 5000;
      w = normal(7);
      x = normal(7);
      z = normal(7);
      y = w * w + log(5 + x) + sin(z) + normal(7);
      output;
   end;
run;

data z;
   do i = 1 to 5000;
      w = normal(1);
      x = normal(1);
      z = normal(1);
      y = w * w + log(5 + x) + sin(z) + normal(1);
      output;
   end;
run;
```

First, you run PROC TRANSREG to fit the transformation regression model asking for spline transformations of the three independent variables. You must use the EXKNOTS= *t-option*, because you need to use the same knots, both interior and exterior, with both data sets. By default, the exterior knots will be different if the minima and maxima are different in the two data sets, so you get the wrong results if you do not specify the EXKNOTS= *t-option* with values less than the minima and greater than the maxima of the six x, y, and w variables. If the ranges in all three pairs were different, you would need separate spline transformation for each variable with different knot and exterior knot specifications. The following statements fit the spline model:

```
proc transreg data=x solve details ss2;
   ods output splinecoef=c;
   model identity(y) = spline(w x z / knots=-1.5 to 1.5 by 0.5
                                      exknots=-5 5);
   output out=d;
run;
```

The results of this step are not displayed. The nonprinting SplineCoef table is output to a SAS data set. This data set contains the coefficients that were used to get the spline transformations and can be used to transform variables in other data sets. These coefficients are also in the details table. However, in the SplineCoef table, they are in a form directly suitable for use with PROC SCORE.

The next step reads the second input data set, Z, and generates an output data set with the B-spline basis for each of the variables:

```
proc transreg data=z design;
   model bspl(w x z / knots=-1.5 to 1.5 by 0.5 exknots=-5 5);
   output out=b;
run;
```

Note that the same interior and exterior knots are used in both of the previous steps. The next three steps score the B-spline bases created in the previous step by using the coefficients generated in the first PROC TRANSREG step. PROC SCORE is run once for each SPLINE variable in the statements that follow:

```
proc score data=b score=c out=o1(rename=(spline=bw w=nw));
   var w:;
run;

proc score data=b score=c out=o2(rename=(spline=bx x=nx));
   var x:;
run;

proc score data=b score=c out=o3(rename=(spline=bz z=nz));
   var z:;
run;
```

The following steps merge the three transformations with the original data and plot the results:

```
data all;
   merge d(keep=w x z tw tx tz) o1(keep=nw bw)
         o2(keep=nx bx) o3(keep=nz bz);
run;

proc template;
   define statgraph twobytwo;
      begingraph;
         layout lattice / rows=2 columns=2;
            layout overlay;
               seriesplot y=tw x=w  / connectorder=xaxis;
               seriesplot y=bw x=nw / connectorder=xaxis;
            endlayout;
            layout overlay;
               seriesplot y=tx x=x  / connectorder=xaxis;
               seriesplot y=bx x=nx / connectorder=xaxis;
            endlayout;
            layout overlay;
               seriesplot y=tz x=z  / connectorder=xaxis;
               seriesplot y=bz x=nz / connectorder=xaxis;
            endlayout;
         endlayout;
      endgraph;
   end;
run;

proc sgrender data=all template=twobytwo;
run;
```

The plots in Figure 126.27 show that the two transformations for each variable, original and scored, are the same function. The two functions in each plot are on top of each other and are indistinguishable. Furthermore, PROC TRANSREG found the functional forms that were used to generate the data: quadratic for w, log for x, and sine for z.

Figure 126.27 Scoring Spline Variables Example

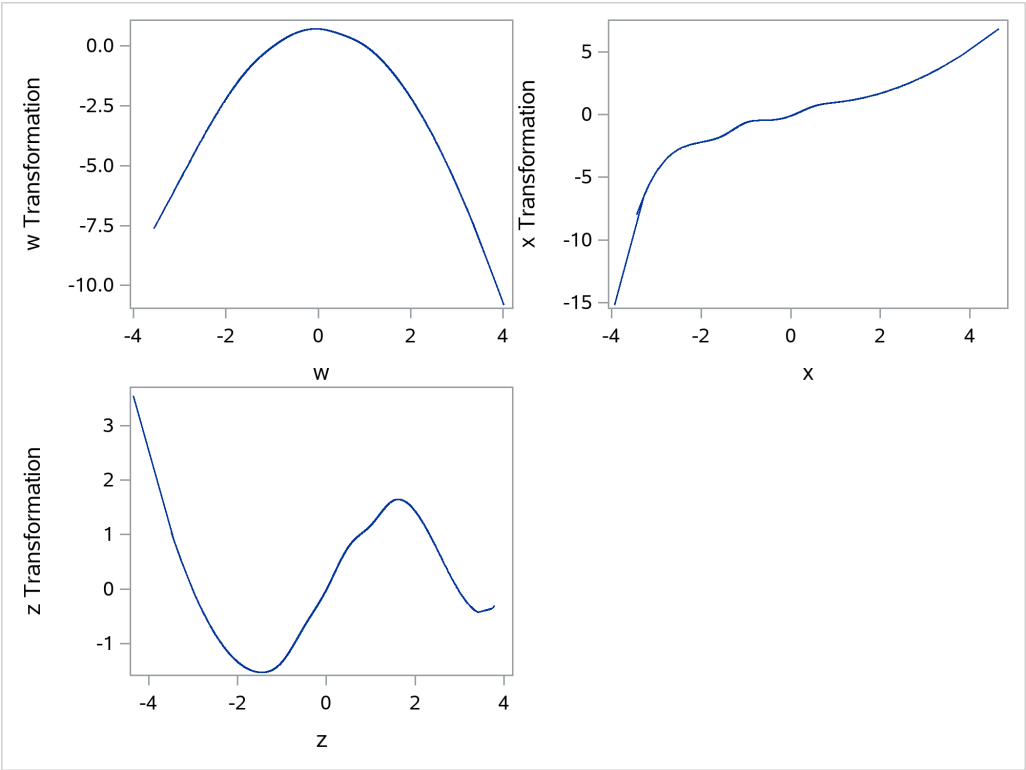

The next statements show how to run PROC TRANSREG, output the interior and exterior knots to an output data set with ODS, extract the knots, and use them in a DATA step to re-create the B-spline basis that PROC TRANSREG makes. In practice, you would never need to use a DATA step to make the B-spline basis since PROC TRANSREG does it automatically. The following statements show how you could do it yourself:

```
data x;
   input x @@;
   datalines;
1 2 3 4 5 6 7 8 9 10
;

ods output details=d;
proc transreg details design;
   model bspline(x / nkn=3);
   output out=y;
run;

%let k = 0;
data d;
   set d;
   length d $ 20;
```

```
   retain d ' ';
   if description ne ' ' then d = description;
   if d = 'Degree' then call symput('d', compress(formattedvalue));
   if d = 'Number of Knots'
      then call symput('k', compress(formattedvalue));
   if index(d, 'Knots') and not index(d, 'Number');
   keep d numericvalue;
run;

%let nkn = %eval(&d * 2 + &k); /* total number of knots   */
%let nb  = %eval(&d + 1 + &k); /* number of cols in basis */

proc transpose data=d out=k(drop=_name_) prefix=Knot;
run;

proc print; format k: 20.16;
run;

data b(keep=x:);
   if _n_ = 1 then set k; /* read knots from transreg */
   array k[&nkn] knot1-knot&nkn;        /* knots      */
   array b[&nb] x_0 - x_%eval(&nb - 1); /* basis      */
   array w[%eval(2 * &d)];              /* work       */
   set x;
   do i = 1 to &nb; b[i] = 0; end;

   * find the index of first knot greater than current data value;
   do ki = 1 to &nkn while(k[ki] le x); end;
   kki = ki - &d - 1;

   * make the basis;
   b[1 + kki] = 1;
   do j = 1 to &d;
      w[&d + j] = k[ki + j - 1] - x;
      w[j] = x - k[ki - j];
      s = 0;
      do i = 1 to j;
         t = w[&d + i] + w[j + 1 - i];
         if t ne 0.0 then t = b[i + kki] / t;
         b[i + kki] = s + w[&d + i] * t;
         s = w[j + 1 - i] * t;
      end;
      b[j + 1 + kki] = s;
   end;
run;

proc compare data=y(keep=x:) compare=b
   criterion=1e-12 note nosummary;
   title3 "should be no differences";
run;
```

The output from these steps is not shown. There are several things to note about the DATA step. It produces the same basis as PROC TRANSREG only because it uses exactly the same interior and exterior knots. The exterior knots (0.999999999999 and 10.000000000001) are just slightly smaller than 1 (the minimum in x)

and just slightly greater than 10 (the maximum in x). Both exterior knots appear in the list three times, because a cubic (degree 3) polynomial was requested. The complete knot list is: 0.999999999999 0.999999999999 0.999999999999 3 6 8 10.000000000001 10.000000000001 10.000000000001. The exterior knots do not have any particular interpretation, but they are needed by the algorithm to construct the proper basis. The construction method computes differences between each value and the nearby knots. The algorithm that makes the B-spline basis is not very obvious, particularly compared to the polynomial spline basis. However, the B-spline basis is much better behaved numerically than a polynomial-spline basis, so that is why it is used.

Linear and Nonlinear Regression Functions

This section shows how to use PROC TRANSREG in simple regression (one dependent variable and one independent variable) to find the optimal regression line, a nonlinear but monotone regression function, and a nonlinear and nonmonotone regression function. To find a linear regression function, specify the IDENTITY transformation of the independent variable. For a monotone curve, specify the MSPLINE transformation of the independent variable. To relax the monotonicity constraint, specify the SPLINE transformation. You can get more flexibility in spline functions by specifying knots. The more knots you specify, the more freedom the function has to follow minor variations in the data. This example uses artificial data. While these artificial data are clearly not realistic, their distinct pattern helps illustrate how splines work. The following statements generate the data and produce Figure 126.28 through Figure 126.31:

```
title 'Linear and Nonlinear Regression Functions';

* Generate an Artificial Nonlinear Scatter Plot;
data a;
   do i = 1 to 500;
      x = i / 2.5;
      y = -((x/50)-1.5)**2 + sin(x/8) + sqrt(x)/5 + 2*log(x) + cos(x);
      x = x / 21;
      if y > 2 then output;
   end;
run;

ods graphics on;
ods select fitplot(persist);

title2 'Linear Regression';

proc transreg data=a;
   model identity(y)=identity(x);
run;

title2 'A Monotone Regression Function';

proc transreg data=a;
   model identity(y)=mspline(x / nknots=9);
run;

title2 'A Nonlinear Regression Function';
```

```
proc transreg data=a;
   model identity(y)=spline(x / nknots=9);
run;

title2 'A Nonlinear Regression Function, 100 Knots';

proc transreg data=a;
   model identity(y)=spline(x / nknots=100);
run;

ods select all;
```

Figure 126.28 Linear Regression

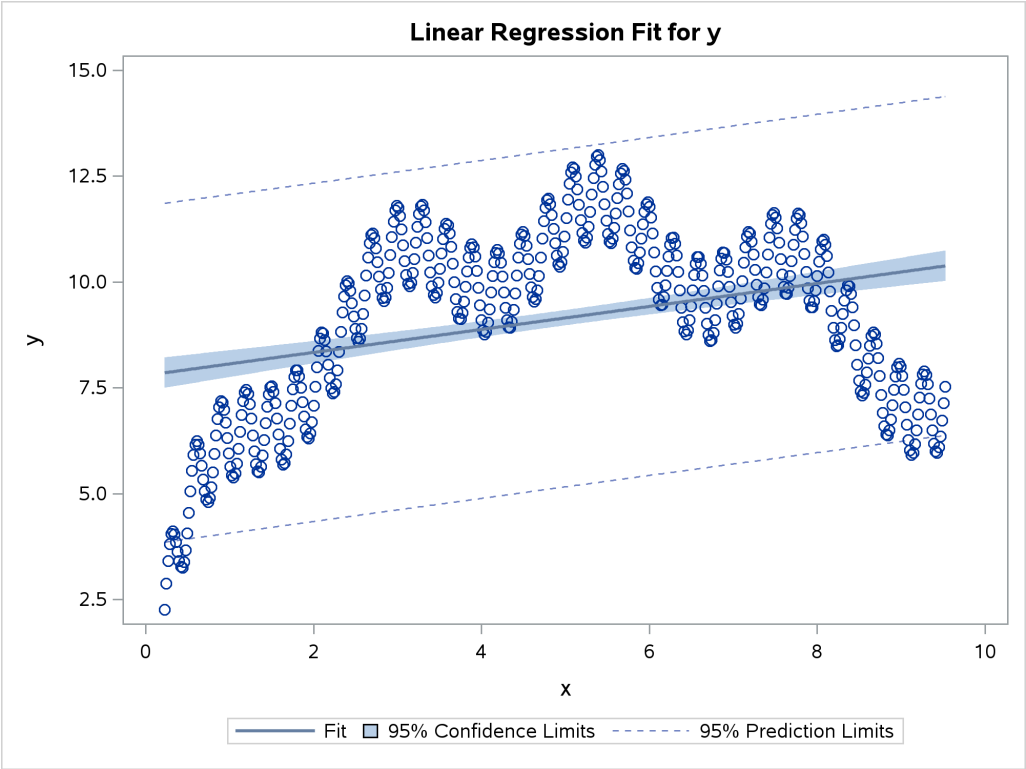

Figure 126.29 A Monotone Regression Function

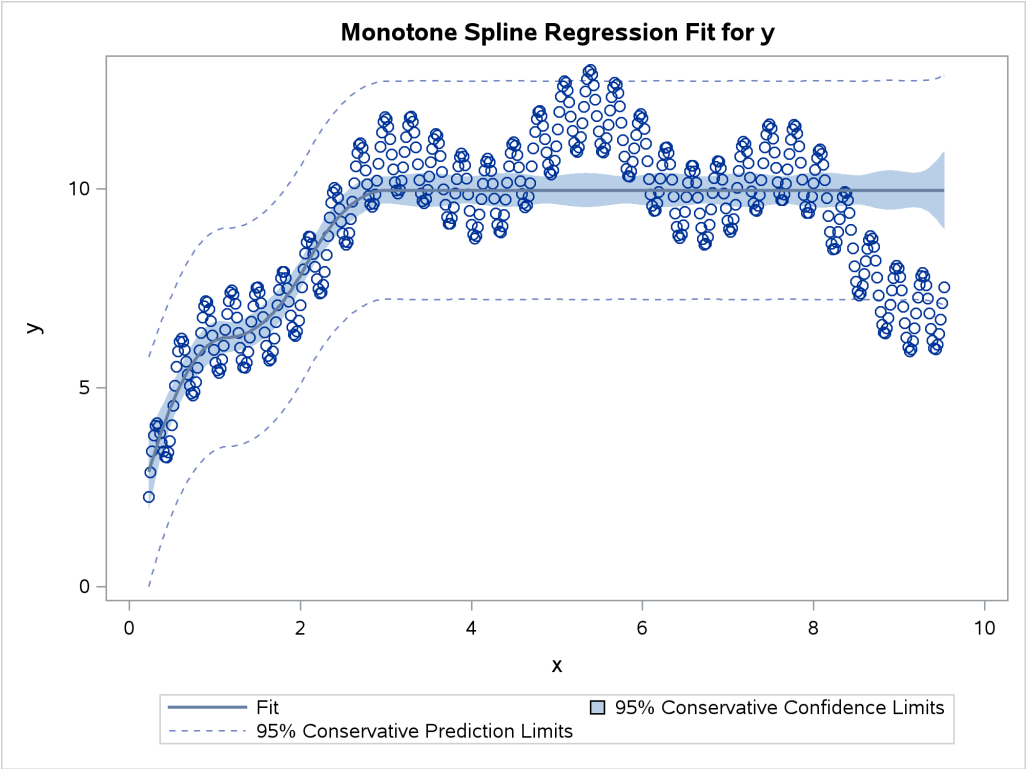

Figure 126.30 A Nonlinear Regression Function

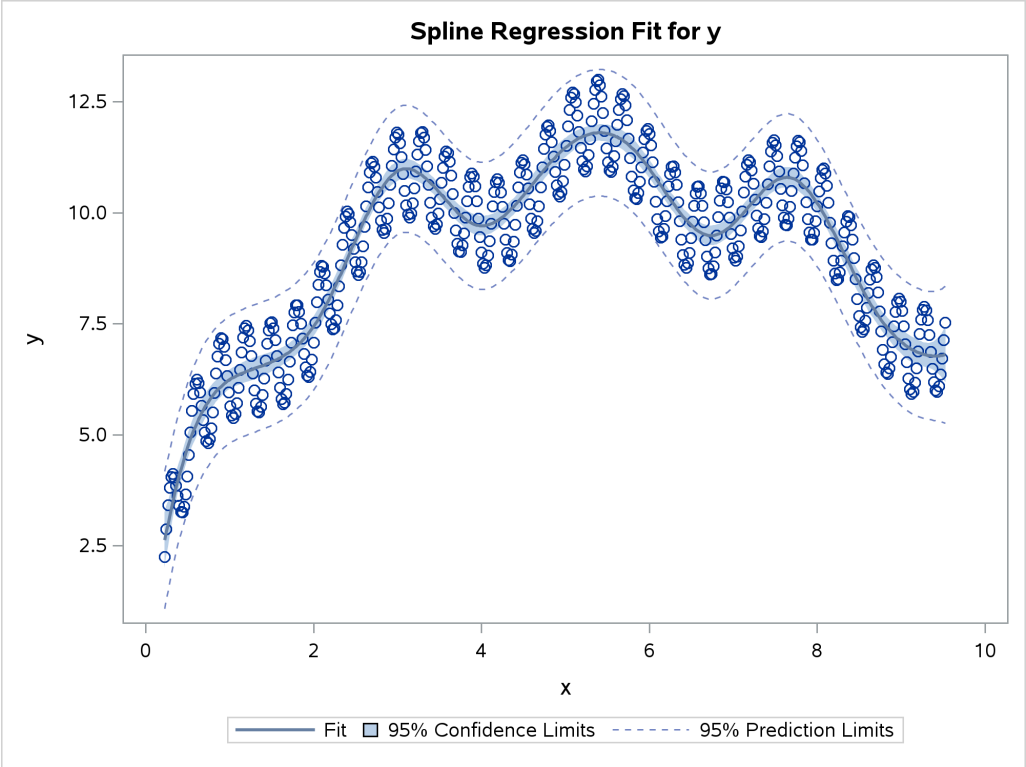

Figure 126.31 A Less-Smooth Nonlinear Regression Function

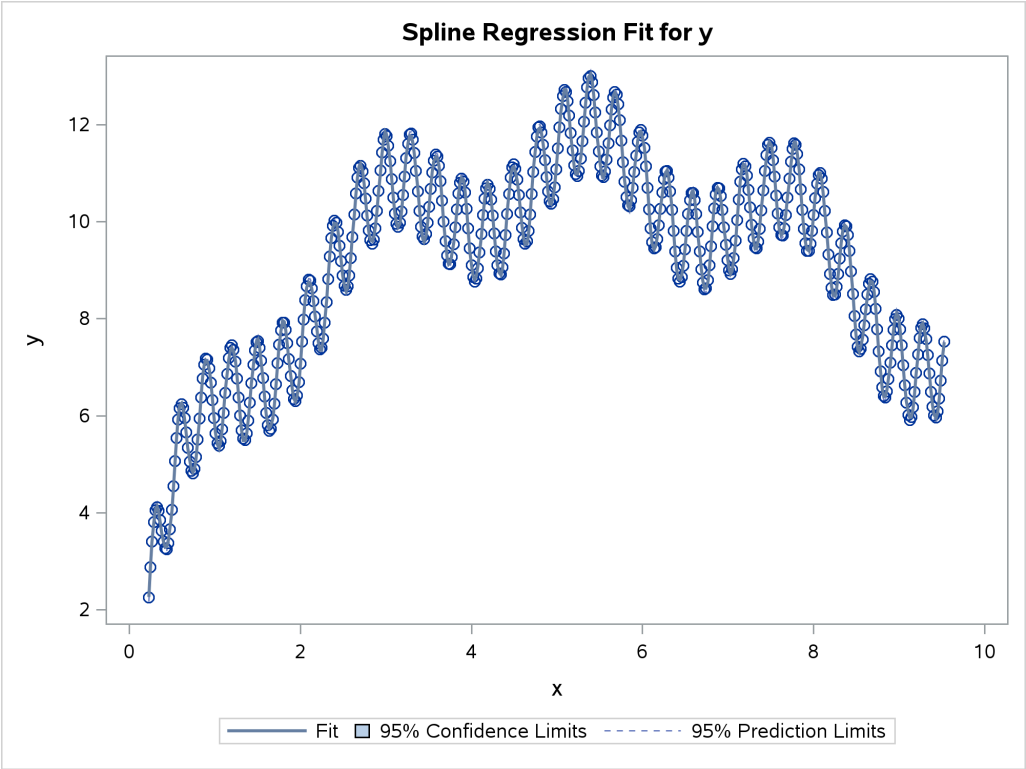

The squared correlation is only 0.15 for the linear regression in Figure 126.28. Clearly, a simple linear regression model is not appropriate for these data. By relaxing the constraints placed on the regression line, the proportion of variance accounted for increases from 0.15 (linear) to 0.61 (monotone in Figure 126.29) to 0.90 (nonlinear but smooth in Figure 126.30) to almost 1.0 with 100 knots (nonlinear and not very smooth in Figure 126.28). Relaxing the linearity constraint permits the regression function to bend and more closely follow the right portion of the scatter plot. Relaxing the monotonicity constraint permits the regression function to follow the periodic portion of the left side of the plot more closely. The nonlinear MSPLINE transformation is a quadratic spline with knots at the deciles. The first nonlinear nonmonotonic SPLINE transformation is a cubic spline with knots at the deciles.

Different knots and different degrees would produce slightly different results. The two nonlinear regression functions could be closely approximated by simpler piecewise linear regression functions. The monotone function could be approximated by a two-piece line with a single knot at the elbow. The first nonmonotone function could be approximated by a six-piece function with knots at the five elbows.

With this type of problem (one dependent variable with no missing values that is not transformed and one independent variable that is nonlinearly transformed), PROC TRANSREG always iterates exactly twice (although only one iteration is necessary). The first iteration reports the R square for the linear regression line and finds the optimal transformation of x. Since the data change in the first iteration, a second iteration is performed, which reports the R square for the final nonlinear regression function, and zero data change. The predicted values, which are a linear function of the optimal transformation of x, contain the Y coordinates for the nonlinear regression function. The variance of the predicted values divided by the variance of y is the R square for the fit of the nonlinear regression function. When x is monotonically transformed, the transformation of x is always monotonically increasing, but the predicted values increase if the correlation is positive and decrease for negative correlations.

Simultaneously Fitting Two Regression Functions

One application of ordinary multiple regression is fitting two or more regression lines through a single scatter plot. With PROC TRANSREG, this application can easily be generalized to fit separate or parallel curves. To illustrate, consider a data set with two groups and a group membership variable g that has the value 1 for one group and 2 for the other group. The data set also has a continuous independent variable x and a continuous dependent variable y. When g is crossed with x, the variables g1x and g2x both have a large partition of zeros. For this reason, the KNOTS= *t-option* is specified instead of the NKNOTS= *t-option*. (The latter would put a number of knots in the partition of zeros.) The following example generates an artificial data set with two curves. While these artificial data are clearly not realistic, their distinct pattern helps illustrate how fitting simultaneous regression functions works. The following statements generate data and show how PROC TRANSREG fits lines, curves, and monotone curves through a scatter plot:

```
title 'Separate Curves, Separate Intercepts';

data a;
   do x = -2 to 3 by 0.025;
      g = 1;
      y = 8*(x*x + 2*cos(x*6)) + 15*normal(7654321);
      output;
      g = 2;
      y = 4*(-x*x + 4*sin(x*4)) - 40 + 15*normal(7654321);
      output;
   end;
run;

ods graphics on;
ods select fitplot(persist);

title 'Parallel Lines, Separate Intercepts';

proc transreg data=a solve;
   model identity(y)=class(g) identity(x);
run;

title 'Parallel Monotone Curves, Separate Intercepts';

proc transreg data=a;
   model identity(y)=class(g) mspline(x / knots=-1.5 to 2.5 by 0.5);
run;

title 'Parallel Curves, Separate Intercepts';

proc transreg data=a solve;
   model identity(y)=class(g) spline(x / knots=-1.5 to 2.5 by 0.5);
run;

title 'Separate Slopes, Same Intercept';

proc transreg data=a;
```

```
   model identity(y)=class(g / zero=none) * identity(x);
run;

title 'Separate Monotone Curves, Same Intercept';

proc transreg data=a;
   model identity(y) = class(g / zero=none) *
                       mspline(x / knots=-1.5 to 2.5 by 0.5);
run;

title 'Separate Curves, Same Intercept';

proc transreg data=a solve;
   model identity(y) = class(g / zero=none) *
                       spline(x / knots=-1.5 to 2.5 by 0.5);
run;

title 'Separate Slopes, Separate Intercepts';

proc transreg data=a;
   model identity(y) = class(g / zero=none) | identity(x);
run;

title 'Separate Monotone Curves, Separate Intercepts';

proc transreg data=a;
   model identity(y) = class(g / zero=none) |
                       mspline(x / knots=-1.5 to 2.5 by 0.5);
run;

title 'Separate Curves, Separate Intercepts';

proc transreg data=a solve;
   model identity(y) = class(g / zero=none) |
                       spline(x / knots=-1.5 to 2.5 by 0.5);
run;
ods select all;
```

The previous statements produce Figure 126.32 through Figure 126.40. Only the fit plots are generated and displayed.

Figure 126.32 Parallel Lines, Separate Intercepts

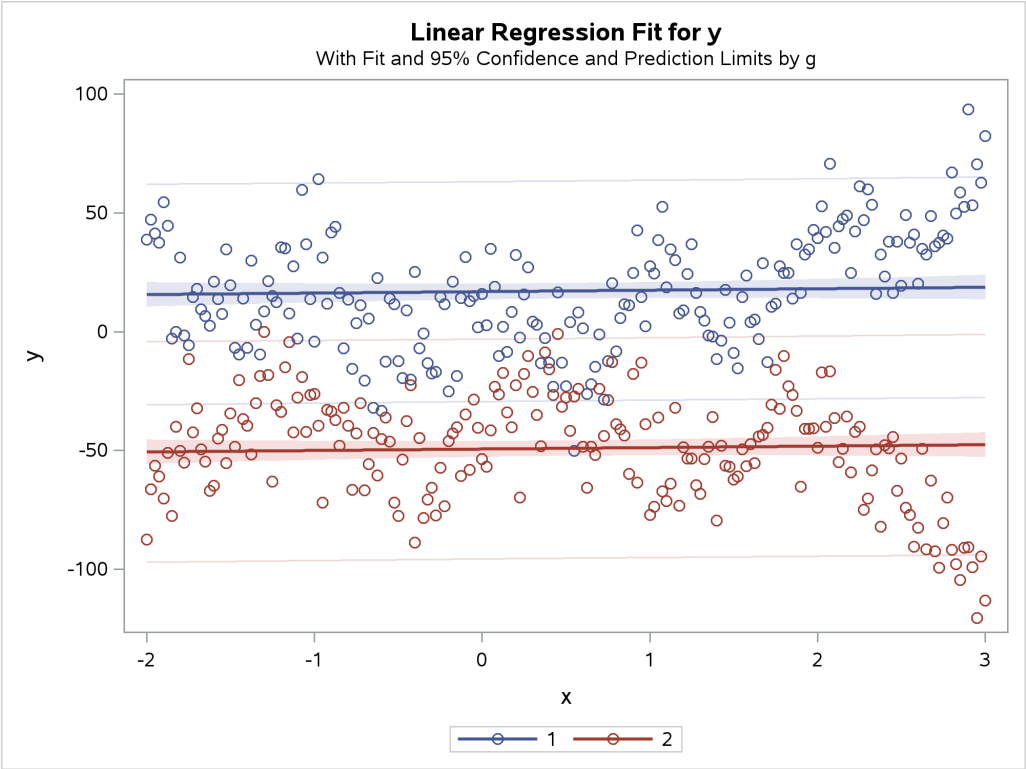

Figure 126.33 Parallel Monotone Curves, Separate Intercepts

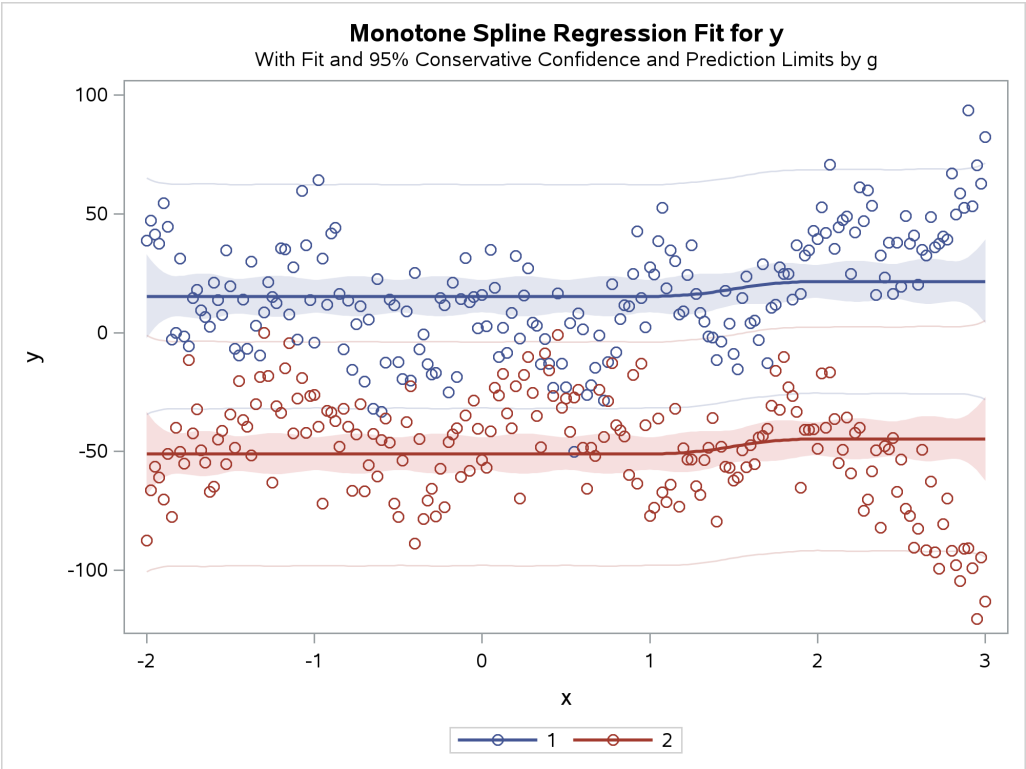

Figure 126.34 Parallel Curves, Separate Intercepts

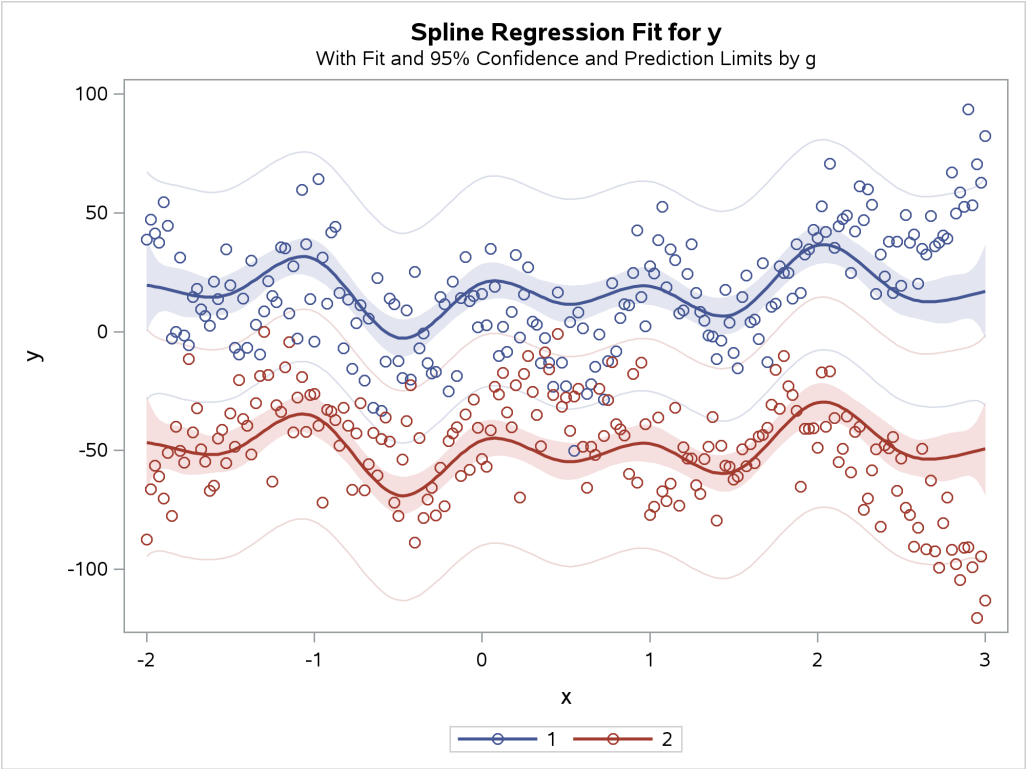

Figure 126.35 Separate Slopes, Same Intercept

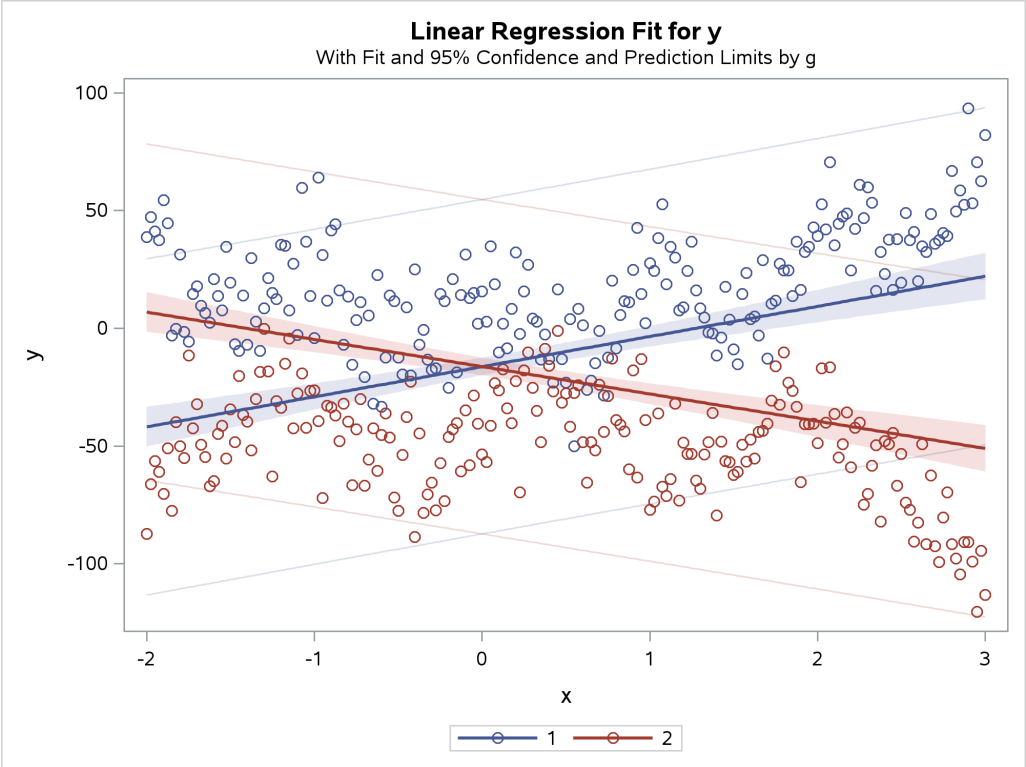

Figure 126.36 Separate Monotone Curves, Same Intercept

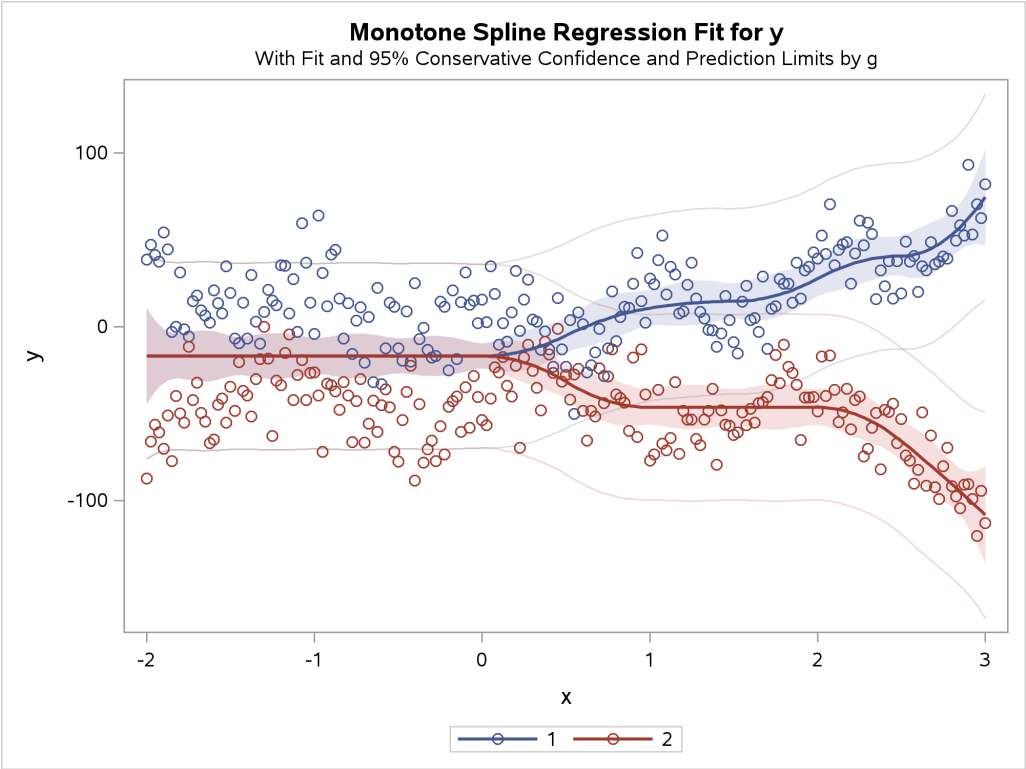

Figure 126.37 Separate Curves, Same Intercept

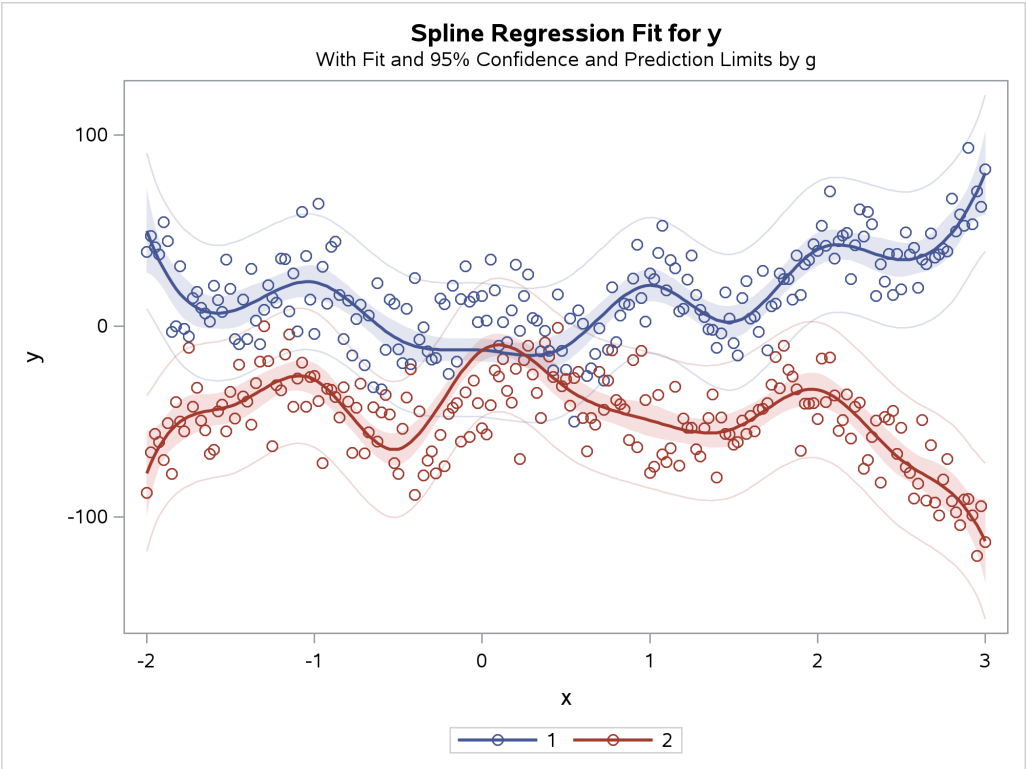

Figure 126.38 Separate Slopes, Separate Intercepts

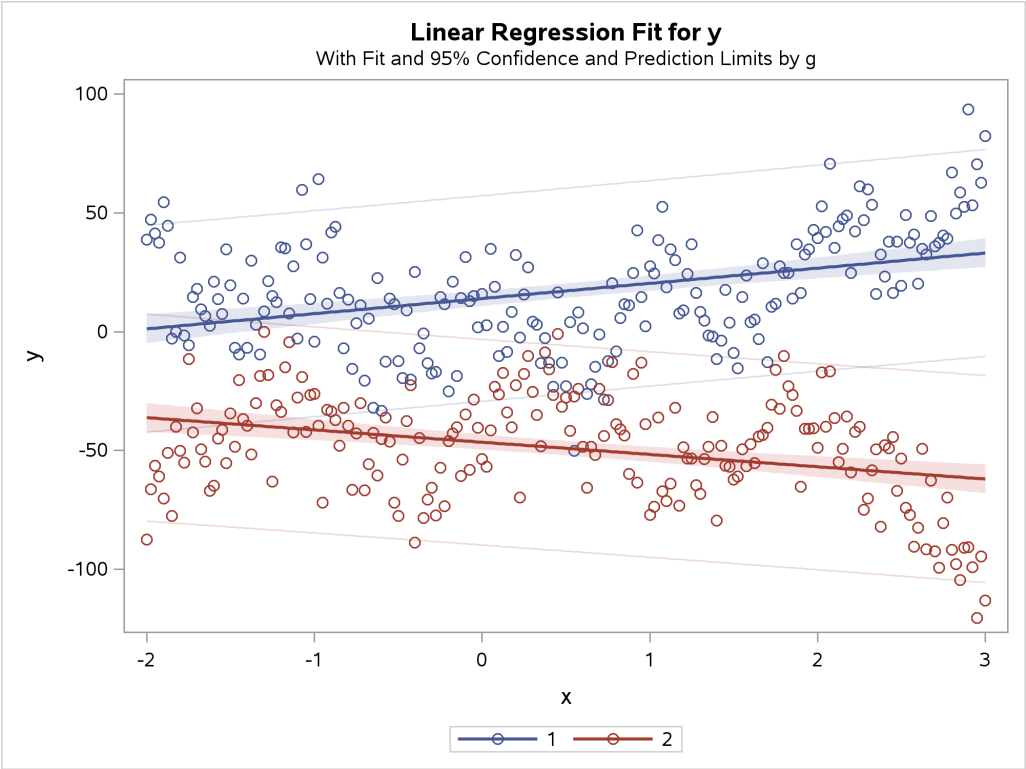

Figure 126.39 Separate Monotone Curves, Separate Intercepts

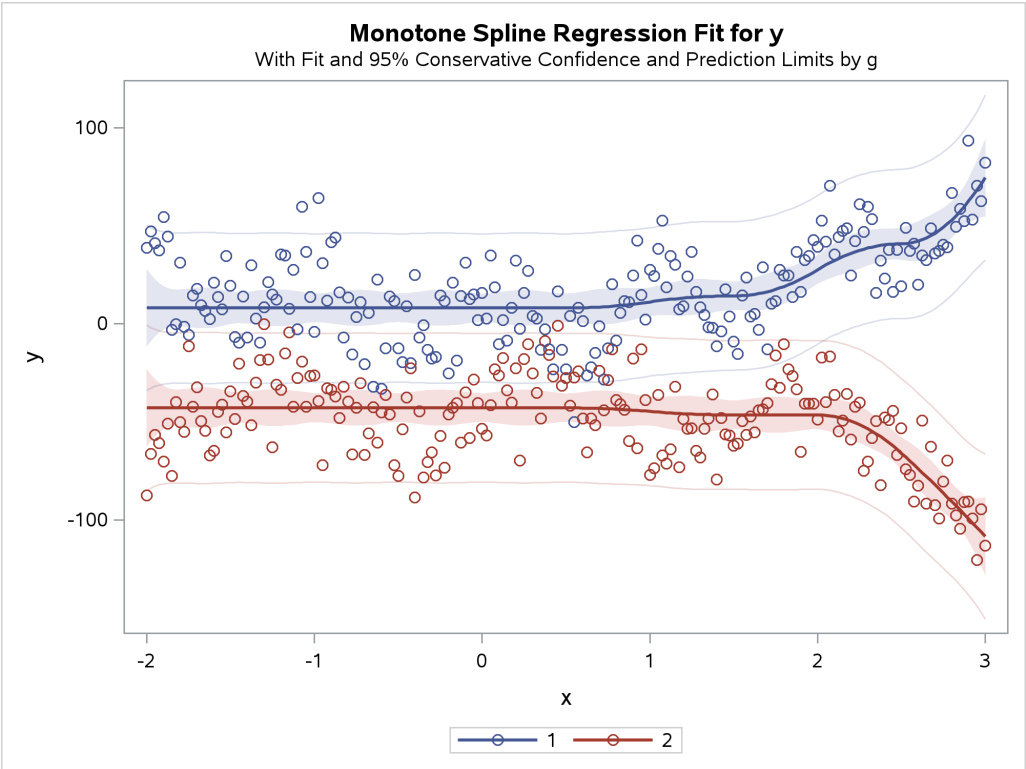

Figure 126.40 Separate Curves, Separate Intercepts

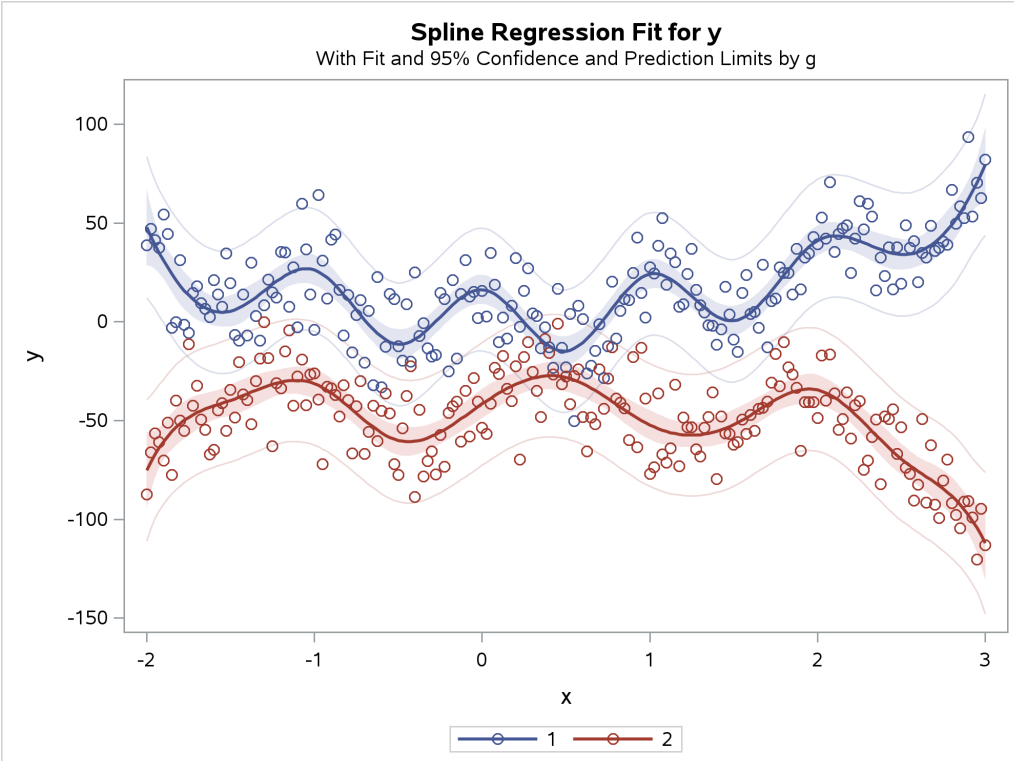

Penalized B-Splines

You can use penalized B-splines (Eilers and Marx 1996) to fit a smooth curve through a scatter plot with an automatic selection of the smoothing parameter. See Example 126.3 for an example. With penalized B-splines, you can find a transformation that minimizes any of the following criteria: CV, GCV, AIC, AICC, or SBC. These criteria are all functions of λ. For many problems, all of these criteria produce nearly identical results. However, for some problems, the choice of criterion can have a large effect. When the default results are not satisfactory, try the other criteria. Information criteria such as AIC and AICC are defined in different ways in the statistical literature, and these differences can be seen in different SAS procedures. Typically, the definitions differ only by a positive (additive or multiplicative) constant, so they are equivalent, and each of the definitions of the same criterion produces the same selection of λ. The definitions that PROC TRANSREG uses match the definitions that PROC REG uses. The penalized B-spline matrices, statistics, and criteria are defined as follows:

n	number of observations
$\mathbf{y}$	dependent variable
$\mathbf{W}$	diagonal matrix of observation weights
w_i	weight for the ith observation
$\mathbf{B}$	B-spline basis for the independent variable
λ	nonnegative smoothing parameter
$\mathbf{D}$	difference matrix, penalizes lack of smoothness
$\mathbf{H} = \mathbf{B}(\mathbf{B}'\mathbf{W}\mathbf{B} + \lambda\mathbf{D}'\mathbf{D})^{-1}\mathbf{B}'\mathbf{W}$	hat matrix
h_{ii}	ith diagonal element of $\mathbf{H}$

$\hat{\mathbf{y}} = \mathbf{H}\mathbf{y}$	penalized B-spline transformation of $\mathbf{y}$
$\mathrm{SSE} = \sum_{i=1}^{n} w_i (y_i - \hat{y}_i)^2$	error sum of squares
$t = \sum_{i=1}^{n} h_{ii}$	trace of $\mathbf{H}$
$\sum_{i=1}^{n} w_i \left(\frac{y_i - \hat{y}_i}{1 - h_{ii}} \right)^2$	CV - cross validation criterion
$\sum_{i=1}^{n} w_i \left(\frac{y_i - \hat{y}_i}{n - t} \right)^2$	GCV - generalized cross validation criterion
$n \log(\mathrm{SSE}/n) + 2t$	AIC - Akaike's information criterion
$1 + \log(\mathrm{SSE}/n) + \frac{2(t+1)}{n-t-2}$	AICC - corrected AIC (default)
$n \log(\mathrm{SSE}/n) + t \log(n)$	SBC - Schwarz's Bayesian criterion

For more information about constructing the B-spline basis, see "SPLINE and MSPLINE Transformations" on page 10693 and the section "Using Splines and Knots" on page 10632. The nonzero elements of $\mathbf{D}$, order 1 are (1 –1), order 2 are (1 –2 1), order 3 (the default) are (1 –3 3 –1), order 4 are (1 –4 6 –4 1), and so on. The nonzero elements for each order are made from the nonzero elements from the preceding order by subtraction: $\mathbf{d}'_{i+1} = (\mathbf{d}'_i \;\; 0) - (0 \;\; \mathbf{d}'_i)$. Within an order, the first nonzero element of row i is in column i—that is, each row of $\mathbf{D}$ is made from the preceding row by shifting the nonzero elements to the right one position. For example, with $k = 4$ knots, order $o = 3$, and degree $d = 3$, $\mathbf{D}$ is the $((d + 1 + k - o) \times (d + 1 + k))$ matrix:

$$\begin{bmatrix} 1 & -3 & 3 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -3 & 3 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -3 & 3 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -3 & 3 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -3 & 3 & -1 \end{bmatrix}$$

where

$$\begin{array}{rrrrr} & 1 & -1 & 0 & \\ - & 0 & 1 & -1 & \\ \hline & 1 & -2 & 1 & 0 \\ - & 0 & 1 & -2 & 1 \\ \hline & 1 & -3 & 3 & -1 \end{array}$$

The trace of the hat matrix, $t = \sum_{i=1}^{n} h_{ii}$, provides an estimate of the number of parameters needed to find the transformation and is used in *df* calculations. Note, however, that in some cases, particularly with error-free or nearly error-free data, this value can be *much* larger than you might expect. You might be able to directly create a function by using SPLINE or BSPLINE with many fewer parameters that fits essentially just as well as the penalized B-spline function.

By default with PBSPLINE, a cubic spline is fit with 100 evenly spaced knots, three evenly spaced exterior knots, and a difference matrix of order three. Options are specified as follows: PBSPLINE(x / DEGREE=3 NKNOTS=100 EVENLY=3 PARAMETER=3). By default, PROC TRANSREG searches for an optimal lambda in the range 0 to 1E6 by using parabolic interpolation and Brent's (Brent 1973; Press et al. 1989) method. Alternatively, you can specify a lambda range or a list of lambdas by using the LAMBDA= option. Be aware, however, LAMBDA=0 and values near zero might cause numerical problems including floating point errors. Also be aware that larger lambdas might cause numerical problems—for example, the error sum of squares for the model, $\Sigma(y - \hat{y})^2$, might be greater than the total sum of squares, $\Sigma(y - \bar{y})^2$—implying that the model with the transformation fits less well than simply predicting by using the mean. When this happens, you will see this message: ERROR: Degenerate transformation with PBSPLINE.

You can fit a single curve through a scatter plot (y × x) as follows:

```
model identity(y) = pbspline(x);
```

Alternatively, you can fit multiple curves through a scatter plot, one for each level of Group, as follows:

```
model identity(y) = class(group / zero=none) * pbspline(x);
```

There are several options for how the smoothing parameter, λ, is chosen. Usually, you do not specify the smoothing parameter, λ, and you let PROC TRANSREG choose λ for you by minimizing one of the information or cross validation criteria. By default, PROC TRANSREG first considers ranges defined by $\lambda = 0$ and $\lambda = 1, 10, 100, 1{,}000, 10{,}000, 100{,}000, 1{,}000{,}000$. If it finds a range that includes the minimum, it stops and does not consider larger λ values. Then it performs further searches in that range. For example, if the initial evaluations at $\lambda = 1$ and $\lambda = 10$ show that there is at least a local minimum in the range 0 to 10, then larger values are not considered. Note that the zero smoothing case, $\lambda = 0$, provides a boundary on the range even though the criterion is not evaluated at $\lambda = 0$. The criterion is not evaluated at $\lambda = 0$ unless LAMBDA=0 is the only value specified. Also note that the default approach is not the same as specifying the options LAMBDA=0 1E6 RANGE. When a range of values is specified, along with the RANGE *t-option*, PROC TRANSREG does not try to find smaller ranges based on powers of 10.

PROC TRANSREG avoids evaluating the criterion for LAMBDA= values at or near zero unless you force it to consider them. This is because zero smoothing is rarely interesting and the results are numerically unstable. Values of λ at or near zero often result in predicted values that are far outside the range of the data, particularly with interpolation and x values that do not appear in the data set. Also, zero smoothing is prone to numerical problems including floating point errors. This is particularly true when there is a small number of observations, a large number of knots, a high degree, or a perfect or near perfect fit. If you force PROC TRANSREG to evaluate the criterion at or near $\lambda = 0$, you can easily get bad results.

Note that when some observations appear more than once, such as when you have the kind of data where you can use a FREQ statement, then you should consider directly specifying lambda based on a preliminary analysis, ignoring the frequencies. Alternatively, specify a range of λ values, such as LAMBDA=0.1 1E6 RANGE, that steers λ away from values near zero. With the default lambda list, a cross validation criterion does not perform well in choosing a smoothing parameter with replicated data. Leaving one observation out of the computations changes the frequency for that observation from one positive integer to the next smaller positive integer, so in some sense, the point corresponding to that observation is never really left out of any computations. The resulting fit will be undersmoothed unless you specify a larger λ.

Smoothing Splines

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/tregsmo.sas.)

You can use PROC TRANSREG to plot and output to a SAS data set the same smoothing spline function that the GPLOT procedure creates. You request a smoothing spline transformation by specifying SMOOTH in the MODEL statement. The smoothing parameter can be specified with either the SM= or the PARAMETER= *o-option*. The results are saved in the independent variable transformation (for example, Tx, when the independent variable is x) and the predicted values variable (for example, Py, when the dependent variable is y).

You can display the smoothing spline by using PROC TRANSREG and ODS Graphics (as shown in Figure 126.41). The following statements produce Figure 126.41:

```
title h=1.5 'Smoothing Splines';

ods graphics on;

data x;
   do x = 1 to 100 by 2;
      do rep = 1 to 3;
         y = log(x) + sin(x / 10) + normal(7);
         output;
      end;
   end;
run;

proc transreg;
   model identity(y) = smooth(x / sm=50);
   output p;
run;
```

Figure 126.41 Smoothing Spline Displayed with ODS Graphics

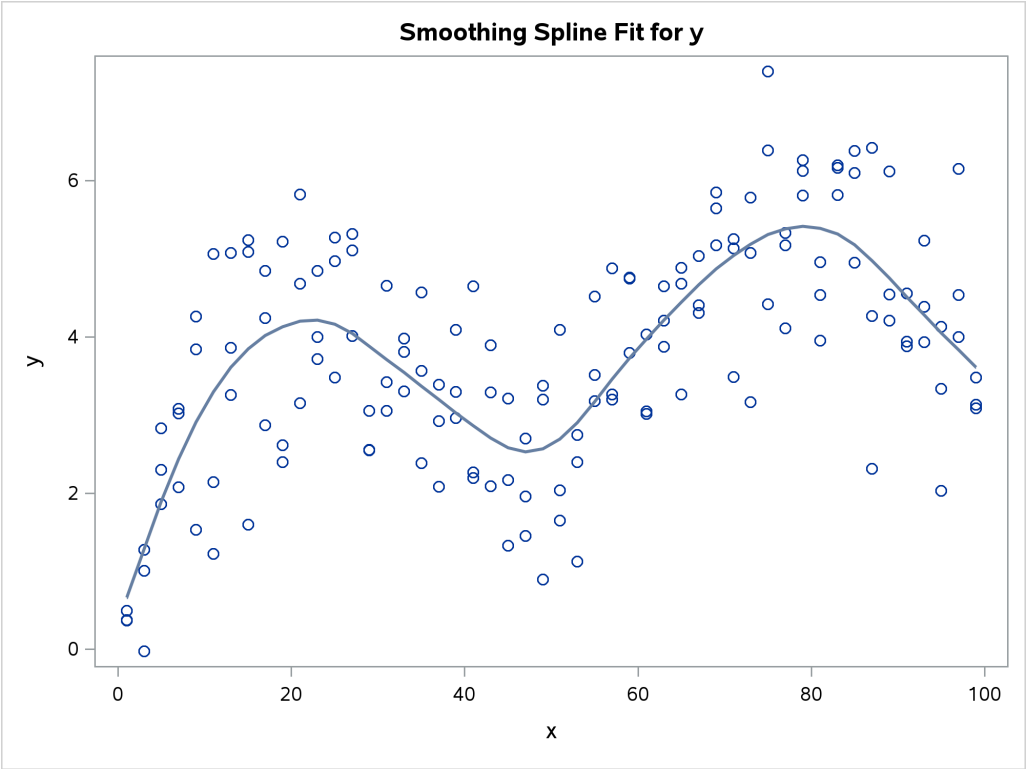

You can also use PROC GPLOT to verify that the two procedures produce the same results. The PROC GPLOT plot request `y * x = 1` displays the data as stars. The specification `y * x = 2` with I=SM50 requests the smooth curve through the scatter plot. It is overlaid with `Py * x = 3`, which displays with large dots the smooth function created by PROC TRANSREG. The results of the following step are not displayed:

```
proc gplot;
   axis1 minor=none label=(angle=90 rotate=0);
   axis2 minor=none;
   symbol1 color=blue v=circle i=none;  /* data              */
   symbol2 color=blue v=none   i=sm50;  /* gplot's smooth    */
   symbol3 color=red  v=dot    i=none;  /* transreg's smooth */
   plot y*x=1 y*x=2 py*x=3 / overlay haxis=axis2 vaxis=axis1 frame;
run; quit;
```

You can plot multiple nonlinear functions, one for each of several groups as defined by the levels of a CLASS variable. When you cross a SMOOTH variable with a CLASS variable, specify ZERO=NONE with the CLASS expansion. The following statements create artificial data and produce Figure 126.42:

```
title2 'Two Groups';

data x;
   do x = 1 to 100;
      Group = 1;
      do rep = 1 to 3;
         y = log(x) + sin(x / 10) + normal(7);
         output;
```

```
      end;
      group = 2;
      do rep = 1 to 3;
         y = -log(x) + cos(x / 10) + normal(7);
         output;
      end;
   end;
run;

proc transreg ss2 data=x;
   model identity(y) = class(group / zero=none) *
                       smooth(x / sm=50);
   output p;
run;
```

The ANOVA table in Figure 126.42 shows the overall model fit. The degrees of freedom are based on the trace of the transformation hat matrix, and are typically not integers. The "Smooth Transformation" table reports the degrees of freedom for each term, which includes an intercept for each group; the regression coefficients, which are always 1 with smoothing splines; the 0 to 100 smoothing parameter (like the one PROC GPLOT uses); the actual computed smoothing parameter; and the name and label for each term.

Figure 126.42 Smoothing Spline Example 2

Smoothing Splines
Two Groups

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	**Levels**	**Values**
Group	2	1 2

Number of Observations Read	600
Number of Observations Used	600
Implicit Intercept Model	

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table, Smooth Transformation					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	16.794	9195.493	547.5365	562.03	<.0001
Error	582.21	567.195	0.9742		
Corrected Total	599	9762.688			

Root MSE	0.98702	**R-Square**	0.9419
Dependent Mean	0.03651	**Adj R-Sq**	0.9402
Coeff Var	2703.13908		

Figure 126.42 *continued*

Smooth Transformation					
Variable	**DF**	**Coefficient**	**SM**	**Parameter**	**Label**
Smooth(Group1x)	8.8971	1.000	50	2405.265	Group 1 * x
Smooth(Group2x)	8.8971	1.000	50	2405.265	Group 2 * x

Figure 126.42 *continued*

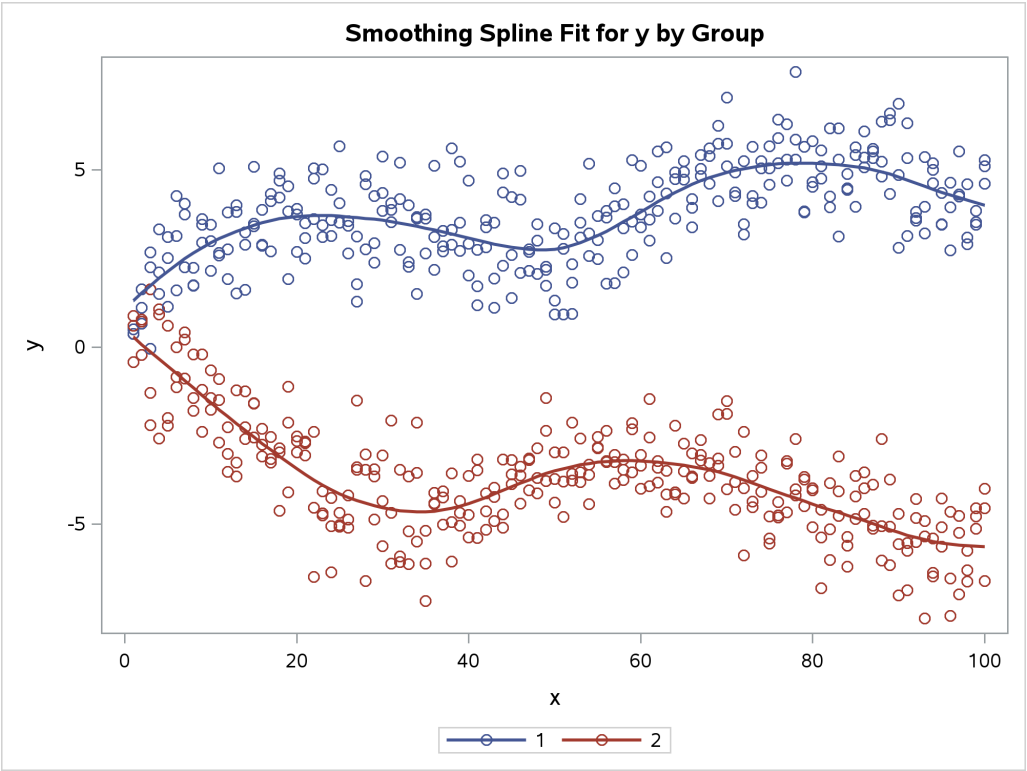

The SMOOTH transformation is valid only with independent variables. Typically, it is used only, as in the two preceding examples, in models with a single dependent variable, a single independent variable, and optionally, a single classification variable that is crossed with the independent variable. The various standardization options such as TSTANDARD=, CENTER, Z, and REFLECT are by default not permitted when the SMOOTH transformation is part of the model.

The SMOOTH transformation can also be used in other ways, but only when you specify the NSR *a-option*. When you specify the NSR *a-option*, and there are multiple independent variables designated as SMOOTH, PROC TRANSREG tries to smooth the *i*th independent variable by using the *i*th dependent variable as a target. When there are more independent variables than dependent variables, the last dependent variable is reused as often as is necessary. For example, consider the following statements:

```
proc transreg nsr;
   model identity(y1-y3) = smooth(x1-x5);
run;
```

Smoothing is based on the pairs (y1, x1), (y2, x2), (y3, x3), (y3, x4), and (y3, x5).

The SMOOTH transformation is a noniterative transformation. The smoothing of each variable occurs before the iterations begin. In contrast, SSPLINE provides an iterative smoothing spline transformation. It does not generally minimize squared error; hence, divergence is possible with SSPLINE.

ANOVA Codings

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/tregano.sas.)

This section illustrates several different codings of classification variables and hence several different ways of fitting two-way ANOVA models to some data. Each example fits an ANOVA model, displays the ANOVA table and parameter estimates, and displays the coded design matrix. Note throughout that the ANOVA tables and R squares are identical for all of the models, showing that the codings are equivalent. For each model, the parameter estimates are stated as a function of the cell means. The formulas are appropriate for a design such as this one, which is balanced and orthogonal (every level and every pair of levels occurs equally often). They will not work with unequal frequencies. Since this data set has $3 \times 2 = 6$ cells, the full-rank codings all have six parameters. The following statements create the input data set, and display it in Figure 126.43:

```
title 'Two-Way ANOVA Models';

data x;
   input a b @@;
   do i = 1 to 2; input y @@; output; end;
   drop i;
   datalines;
1 1   16 14         1 2   15 13
2 1    1  9         2 2   12 20
3 1   14  8         3 2   18 20
;

proc print label;
run;
```

Figure 126.43 Input Data Set

Two-Way ANOVA Models

Obs	a	b	y
1	1	1	16
2	1	1	14
3	1	2	15
4	1	2	13
5	2	1	1
6	2	1	9
7	2	2	12
8	2	2	20
9	3	1	14
10	3	1	8
11	3	2	18
12	3	2	20

The following statements fit a cell-means model and produce Figure 126.44 and Figure 126.45:

```
proc transreg data=x ss2 short;
   title2 'Cell-Means Model';
   model identity(y) = class(a * b / zero=none);
   output replace;
run;

proc print label;
run;
```

Figure 126.44 Cell-Means Model

Two-Way ANOVA Models
Cell-Means Model

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	**Levels**	**Values**
a	3	1 2 3
b	2	1 2

Number of Observations Read	12
Number of Observations Used	12
Implicit Intercept Model	

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	5	234.6667	46.93333	3.20	0.0946
Error	6	88.0000	14.66667		
Corrected Total	11	322.6667			

Root MSE	3.82971	**R-Square**	0.7273
Dependent Mean	13.33333	**Adj R-Sq**	0.5000
Coeff Var	28.72281		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	**DF**	**Coefficient**	**Type II Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**	**Label**
Class.a1b1	1	15.0000000	450.000	450.000	30.68	0.0015	a 1 * b 1
Class.a1b2	1	14.0000000	392.000	392.000	26.73	0.0021	a 1 * b 2
Class.a2b1	1	5.0000000	50.000	50.000	3.41	0.1144	a 2 * b 1
Class.a2b2	1	16.0000000	512.000	512.000	34.91	0.0010	a 2 * b 2
Class.a3b1	1	11.0000000	242.000	242.000	16.50	0.0066	a 3 * b 1
Class.a3b2	1	19.0000000	722.000	722.000	49.23	0.0004	a 3 * b 2

The parameter estimates are

$$
\begin{aligned}
\hat{\mu}_{11} &= \overline{y}_{11} = 15 \\
\hat{\mu}_{12} &= \overline{y}_{12} = 14 \\
\hat{\mu}_{21} &= \overline{y}_{21} = 5 \\
\hat{\mu}_{22} &= \overline{y}_{22} = 16 \\
\hat{\mu}_{31} &= \overline{y}_{31} = 11 \\
\hat{\mu}_{32} &= \overline{y}_{32} = 19
\end{aligned}
$$

Figure 126.45 Cell-Means Model, Design Matrix

Two-Way ANOVA Models
Cell-Means Model

Obs	_TYPE_	_NAME_	y	Intercept	a 1*b 1	a 1*b 2	a 2*b 1	a 2*b 2	a 3*b 1	a 3*b 2	a	b
1	SCORE	ROW1	16	.	1	0	0	0	0	0	1	1
2	SCORE	ROW2	14	.	1	0	0	0	0	0	1	1
3	SCORE	ROW3	15	.	0	1	0	0	0	0	1	2
4	SCORE	ROW4	13	.	0	1	0	0	0	0	1	2
5	SCORE	ROW5	1	.	0	0	1	0	0	0	2	1
6	SCORE	ROW6	9	.	0	0	1	0	0	0	2	1
7	SCORE	ROW7	12	.	0	0	0	1	0	0	2	2
8	SCORE	ROW8	20	.	0	0	0	1	0	0	2	2
9	SCORE	ROW9	14	.	0	0	0	0	1	0	3	1
10	SCORE	ROW10	8	.	0	0	0	0	1	0	3	1
11	SCORE	ROW11	18	.	0	0	0	0	0	1	3	2
12	SCORE	ROW12	20	.	0	0	0	0	0	1	3	2

The next model is a reference cell model, and the default reference cell is the last cell, which in this case is the (3,2) cell. The following statements fit a reference cell model and produce Figure 126.46 and Figure 126.47:

```
proc transreg data=x ss2 short;
   title2 'Reference Cell Model, (3,2) Reference Cell';
   model identity(y) = class(a | b);
   output replace;
run;

proc print label;
run;
```

Figure 126.46 Reference Cell Model, (3,2) Reference Cell

Two-Way ANOVA Models
Reference Cell Model, (3,2) Reference Cell

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	**Levels**	**Values**
a	3	1 2 3
b	2	1 2

Number of Observations Read	12
Number of Observations Used	12

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	5	234.6667	46.93333	3.20	0.0946
Error	6	88.0000	14.66667		
Corrected Total	11	322.6667			

Root MSE	3.82971	**R-Square**	0.7273
Dependent Mean	13.33333	**Adj R-Sq**	0.5000
Coeff Var	28.72281		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	**DF**	**Coefficient**	**Type II Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**	**Label**
Intercept	1	19.0000000	722.000	722.000	49.23	0.0004	Intercept
Class.a1	1	-5.0000000	25.000	25.000	1.70	0.2395	a 1
Class.a2	1	-3.0000000	9.000	9.000	0.61	0.4632	a 2
Class.b1	1	-8.0000000	64.000	64.000	4.36	0.0817	b 1
Class.a1b1	1	9.0000000	40.500	40.500	2.76	0.1476	a 1 * b 1
Class.a2b1	1	-3.0000000	4.500	4.500	0.31	0.5997	a 2 * b 1

The parameter estimates are

$$
\begin{aligned}
\hat{\mu}_{32} &= \overline{y}_{32} = 19 \\
\hat{\alpha}_1 &= \overline{y}_{12} - \overline{y}_{32} = 14 - 19 = -5 \\
\hat{\alpha}_2 &= \overline{y}_{22} - \overline{y}_{32} = 16 - 19 = -3 \\
\hat{\beta}_1 &= \overline{y}_{31} - \overline{y}_{32} = 11 - 19 = -8 \\
\hat{\gamma}_{11} &= \overline{y}_{11} - (\hat{\mu}_{32} + \hat{\alpha}_1 + \hat{\beta}_1) = 15 - (19 + -5 + -8) = 9 \\
\hat{\gamma}_{21} &= \overline{y}_{21} - (\hat{\mu}_{32} + \hat{\alpha}_2 + \hat{\beta}_1) = 5 - (19 + -3 + -8) = -3
\end{aligned}
$$

Figure 126.47 Reference Cell Model, (3,2) Reference Cell, Design Matrix

Two-Way ANOVA Models
Reference Cell Model, (3,2) Reference Cell

Obs	_TYPE_	_NAME_	y	Intercept	a 1	a 2	b 1	a 1*b 1	a 2*b 1	a	b
1	SCORE	ROW1	16	1	1	0	1	1	0	1	1
2	SCORE	ROW2	14	1	1	0	1	1	0	1	1
3	SCORE	ROW3	15	1	1	0	0	0	0	1	2
4	SCORE	ROW4	13	1	1	0	0	0	0	1	2
5	SCORE	ROW5	1	1	0	1	1	0	1	2	1
6	SCORE	ROW6	9	1	0	1	1	0	1	2	1
7	SCORE	ROW7	12	1	0	1	0	0	0	2	2
8	SCORE	ROW8	20	1	0	1	0	0	0	2	2
9	SCORE	ROW9	14	1	0	0	1	0	0	3	1
10	SCORE	ROW10	8	1	0	0	1	0	0	3	1
11	SCORE	ROW11	18	1	0	0	0	0	0	3	2
12	SCORE	ROW12	20	1	0	0	0	0	0	3	2

The next model is a deviations-from-means model. This coding is also called effects coding. The default reference cell is the last cell (3,2). The following statements produce Figure 126.48 and Figure 126.49:

```
proc transreg data=x ss2 short;
   title2 'Deviations from Means, (3,2) Reference Cell';
   model identity(y) = class(a | b / deviations);
   output replace;
run;

proc print label;
run;
```

Figure 126.48 Deviations-from-Means Model, (3,2) Reference Cell

Two-Way ANOVA Models
Deviations from Means, (3,2) Reference Cell

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	Levels	Values
a	3	1 2 3
b	2	1 2

Number of Observations Read	12
Number of Observations Used	12

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	234.6667	46.93333	3.20	0.0946
Error	6	88.0000	14.66667		
Corrected Total	11	322.6667			

Root MSE	3.82971	R-Square	0.7273
Dependent Mean	13.33333	Adj R-Sq	0.5000
Coeff Var	28.72281		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Intercept	1	13.3333333	2133.33	2133.33	145.45	<.0001	Intercept
Class.a1	1	1.1666667	8.17	8.17	0.56	0.4837	a 1
Class.a2	1	-2.8333333	48.17	48.17	3.28	0.1199	a 2
Class.b1	1	-3.0000000	108.00	108.00	7.36	0.0349	b 1
Class.a1b1	1	3.5000000	73.50	73.50	5.01	0.0665	a 1 * b 1
Class.a2b1	1	-2.5000000	37.50	37.50	2.56	0.1609	a 2 * b 1

The parameter estimates are

$$
\begin{aligned}
\hat{\mu} &= \overline{y} = 13.3333 \\
\hat{\alpha}_1 &= (\overline{y}_{11} + \overline{y}_{12})/2 - \overline{y} = (15 + 14)/2 - 13.3333 = 1.1667 \\
\hat{\alpha}_2 &= (\overline{y}_{21} + \overline{y}_{22})/2 - \overline{y} = (5 + 16)/2 - 13.3333 = -2.8333 \\
\hat{\beta}_1 &= (\overline{y}_{11} + \overline{y}_{21} + \overline{y}_{31})/3 - \overline{y} = (15 + 5 + 11)/3 - 13.3333 = -3 \\
\hat{\gamma}_{11} &= \overline{y}_{11} - (\overline{y} + \hat{\alpha}_1 + \hat{\beta}_1) = 15 - (13.3333 + 1.1667 + -3) = 3.5 \\
\hat{\gamma}_{21} &= \overline{y}_{21} - (\overline{y} + \hat{\alpha}_2 + \hat{\beta}_1) = 5 - (13.3333 + -2.8333 + -3) = -2.5
\end{aligned}
$$

Figure 126.49 Deviations-from-Means Model, (3,2) Reference Cell, Design Matrix

Two-Way ANOVA Models
Deviations from Means, (3,2) Reference Cell

Obs	_TYPE_	_NAME_	y	Intercept	a 1	a 2	b 1	a*b 1 1	a*b 2 1	a	b
1	SCORE	ROW1	16	1	1	0	1	1	0	1	1
2	SCORE	ROW2	14	1	1	0	1	1	0	1	1
3	SCORE	ROW3	15	1	1	0	-1	-1	0	1	2
4	SCORE	ROW4	13	1	1	0	-1	-1	0	1	2
5	SCORE	ROW5	1	1	0	1	1	0	1	2	1
6	SCORE	ROW6	9	1	0	1	1	0	1	2	1
7	SCORE	ROW7	12	1	0	1	-1	0	-1	2	2
8	SCORE	ROW8	20	1	0	1	-1	0	-1	2	2
9	SCORE	ROW9	14	1	-1	-1	1	-1	-1	3	1
10	SCORE	ROW10	8	1	-1	-1	1	-1	-1	3	1
11	SCORE	ROW11	18	1	-1	-1	-1	1	1	3	2
12	SCORE	ROW12	20	1	-1	-1	-1	1	1	3	2

The next model is a less-than-full-rank model. The parameter estimates are constrained to sum to zero within each effect. The following statements produce Figure 126.50 and Figure 126.51:

```
proc transreg data=x ss2 short;
   title2 'Less-Than-Full-Rank Model';
   model identity(y) = class(a | b / zero=sum);
   output replace;
run;

proc print label;
run;
```

Figure 126.50 Less-Than-Full-Rank Model

Two-Way ANOVA Models
Less-Than-Full-Rank Model

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	**Levels**	**Values**
a	3	1 2 3
b	2	1 2

Number of Observations Read	12
Number of Observations Used	12

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	5	234.6667	46.93333	3.20	0.0946
Error	6	88.0000	14.66667		
Corrected Total	11	322.6667			

Root MSE	3.82971	**R-Square**	0.7273
Dependent Mean	13.33333	**Adj R-Sq**	0.5000
Coeff Var	28.72281		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	**DF**	**Coefficient**	**Type II Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**	**Label**
Intercept	1	13.3333333	2133.33	2133.33	145.45	<.0001	Intercept
Class.a1	1	1.1666667	8.17	8.17	0.56	0.4837	a 1
Class.a2	1	-2.8333333	48.17	48.17	3.28	0.1199	a 2
Class.a3	1	1.6666667	16.67	16.67	1.14	0.3274	a 3
Class.b1	1	-3.0000000	108.00	108.00	7.36	0.0349	b 1
Class.b2	1	3.0000000	108.00	108.00	7.36	0.0349	b 2
Class.a1b1	1	3.5000000	73.50	73.50	5.01	0.0665	a 1 * b 1
Class.a1b2	1	-3.5000000	73.50	73.50	5.01	0.0665	a 1 * b 2
Class.a2b1	1	-2.5000000	37.50	37.50	2.56	0.1609	a 2 * b 1
Class.a2b2	1	2.5000000	37.50	37.50	2.56	0.1609	a 2 * b 2
Class.a3b1	1	-1.0000000	6.00	6.00	0.41	0.5461	a 3 * b 1
Class.a3b2	1	1.0000000	6.00	6.00	0.41	0.5461	a 3 * b 2

The sum of the regression table DF's, minus one for the intercept, will be greater than the model df when there are ZERO=SUM constraints.

The parameter estimates are

$$
\begin{aligned}
\hat{\mu} &= \overline{y} = 13.3333 \\
\hat{\alpha}_1 &= (\overline{y}_{11} + \overline{y}_{12})/2 - \overline{y} = (15 + 14)/2 - 13.3333 = 1.1667 \\
\hat{\alpha}_2 &= (\overline{y}_{21} + \overline{y}_{22})/2 - \overline{y} = (5 + 16)/2 - 13.3333 = -2.8333 \\
\hat{\alpha}_3 &= (\overline{y}_{31} + \overline{y}_{32})/2 - \overline{y} = (11 + 19)/2 - 13.3333 = 1.6667 \\
\hat{\beta}_1 &= (\overline{y}_{11} + \overline{y}_{21} + \overline{y}_{31})/3 - \overline{y} = (15 + 5 + 11)/3 - 13.3333 = -3 \\
\hat{\beta}_2 &= (\overline{y}_{12} + \overline{y}_{22} + \overline{y}_{32})/3 - \overline{y} = (14 + 16 + 19)/3 - 13.3333 = 3 \\
\hat{\gamma}_{11} &= \overline{y}_{11} - (\overline{y} + \hat{\alpha}_1 + \hat{\beta}_1) = 15 - (13.3333 + 1.1667 + -3) = 3.5 \\
\hat{\gamma}_{12} &= \overline{y}_{12} - (\overline{y} + \hat{\alpha}_1 + \hat{\beta}_2) = 14 - (13.3333 + 1.1667 + 3) = -3.5 \\
\hat{\gamma}_{21} &= \overline{y}_{21} - (\overline{y} + \hat{\alpha}_2 + \hat{\beta}_1) = 5 - (13.3333 + -2.8333 + -3) = -2.5 \\
\hat{\gamma}_{22} &= \overline{y}_{22} - (\overline{y} + \hat{\alpha}_2 + \hat{\beta}_2) = 16 - (13.3333 + -2.8333 + 3) = 2.5 \\
\hat{\gamma}_{31} &= \overline{y}_{31} - (\overline{y} + \hat{\alpha}_3 + \hat{\beta}_1) = 11 - (13.3333 + 1.6667 + -3) = -1 \\
\hat{\gamma}_{32} &= \overline{y}_{32} - (\overline{y} + \hat{\alpha}_3 + \hat{\beta}_2) = 19 - (13.3333 + 1.6667 + 3) = 1
\end{aligned}
$$

The constraints are

$$\alpha_1 + \alpha_2 + \alpha_3 \equiv \beta_1 + \beta_2 \equiv 0$$

$$\gamma_{11} + \gamma_{12} \equiv \gamma_{21} + \gamma_{22} \equiv \gamma_{31} + \gamma_{32} \equiv \gamma_{11} + \gamma_{21} + \gamma_{31} \equiv \gamma_{12} + \gamma_{22} + \gamma_{32} \equiv 0$$

Only four of the five interaction constraints are needed. The fifth constraint is implied by the other four. (Given a 2×3 table with four marginal sum-to-zero constraints, you can freely fill in only two cells. The values in the other four cells are determined from the first two cells and the constraints.) A full-rank model has six estimable parameters. This less-than-full-rank model has one parameter for the intercept, two for the first main effect (plus one more as determined by the first constraint), one for the second main effect (plus one more as determined by the second constraint), and two for the interactions (plus four more as determined by the next four constraints). Six of the twelve parameters are determined given the other six and the constraints. Notice that $\hat{\mu}, \hat{\alpha}_1, \hat{\alpha}_2, \hat{\beta}_1, \hat{\gamma}_{11}$, and $\hat{\gamma}_{21}$ match the corresponding estimates from the effects coding.

Figure 126.51 Less-Than-Full-Rank Model, Design Matrix

Two-Way ANOVA Models
Less-Than-Full-Rank Model

Obs	_TYPE_	_NAME_	y	Intercept	a 1	a 2	a 3	b 1	b 2	a 1*b 1	a 1*b 2	a 2*b 1	a 2*b 2	a 3*b 1	a 3*b 2	a	b
1	SCORE	ROW1	16	1	1	0	0	1	0	1	0	0	0	0	0	1	1
2	SCORE	ROW2	14	1	1	0	0	1	0	1	0	0	0	0	0	1	1
3	SCORE	ROW3	15	1	1	0	0	0	1	0	1	0	0	0	0	1	2
4	SCORE	ROW4	13	1	1	0	0	0	1	0	1	0	0	0	0	1	2
5	SCORE	ROW5	1	1	0	1	0	1	0	0	0	1	0	0	0	2	1
6	SCORE	ROW6	9	1	0	1	0	1	0	0	0	1	0	0	0	2	1
7	SCORE	ROW7	12	1	0	1	0	0	1	0	0	0	1	0	0	2	2
8	SCORE	ROW8	20	1	0	1	0	0	1	0	0	0	1	0	0	2	2
9	SCORE	ROW9	14	1	0	0	1	1	0	0	0	0	0	1	0	3	1
10	SCORE	ROW10	8	1	0	0	1	1	0	0	0	0	0	1	0	3	1
11	SCORE	ROW11	18	1	0	0	1	0	1	0	0	0	0	0	1	3	2
12	SCORE	ROW12	20	1	0	0	1	0	1	0	0	0	0	0	1	3	2

The next model is a reference cell model, but this time the reference cell is the first cell (1,1). The following statements produce Figure 126.52 and Figure 126.53:

```
proc transreg data=x ss2 short;
   title2 'Reference Cell Model, (1,1) Reference Cell';
   model identity(y) = class(a | b / zero=first);
   output replace;
run;

proc print label;
run;
```

Figure 126.52 Reference Cell Model, (1,1) Reference Cell

Two-Way ANOVA Models
Reference Cell Model, (1,1) Reference Cell

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	Levels	Values
a	3	1 2 3
b	2	1 2

Number of Observations Read	12
Number of Observations Used	12

Figure 126.52 *continued*

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	234.6667	46.93333	3.20	0.0946
Error	6	88.0000	14.66667		
Corrected Total	11	322.6667			

Root MSE	3.82971	R-Square	0.7273
Dependent Mean	13.33333	Adj R-Sq	0.5000
Coeff Var	28.72281		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Intercept	1	15.000000	450.000	450.000	30.68	0.0015	Intercept
Class.a2	1	-10.000000	100.000	100.000	6.82	0.0401	a 2
Class.a3	1	-4.000000	16.000	16.000	1.09	0.3365	a 3
Class.b2	1	-1.000000	1.000	1.000	0.07	0.8027	b 2
Class.a2b2	1	12.000000	72.000	72.000	4.91	0.0686	a 2 * b 2
Class.a3b2	1	9.000000	40.500	40.500	2.76	0.1476	a 3 * b 2

The parameter estimates are

$$
\begin{aligned}
\hat{\mu}_{11} &= \overline{y}_{11} = 15 \\
\hat{\alpha}_2 &= \overline{y}_{21} - \overline{y}_{11} = 5 - 15 = -10 \\
\hat{\alpha}_3 &= \overline{y}_{31} - \overline{y}_{11} = 11 - 15 = -4 \\
\hat{\beta}_2 &= \overline{y}_{12} - \overline{y}_{11} = 14 - 15 = -1 \\
\hat{\gamma}_{22} &= \overline{y}_{22} - (\hat{\mu}_{11} + \hat{\alpha}_2 + \hat{\beta}_2) = 16 - (15 + -10 + -1) = 12 \\
\hat{\gamma}_{32} &= \overline{y}_{32} - (\hat{\mu}_{11} + \hat{\alpha}_3 + \hat{\beta}_2) = 19 - (15 + -4 + -1) = 9
\end{aligned}
$$

Figure 126.53 Reference Cell Model, (1,1) Reference Cell, Design Matrix

Two-Way ANOVA Models
Reference Cell Model, (1,1) Reference Cell

Obs	_TYPE_	_NAME_	y	Intercept	a 2	a 3	b 2	a*b 2	a*b 3	a*b 2	a	b
1	SCORE	ROW1	16	1	0	0	0	0	0	1	1	
2	SCORE	ROW2	14	1	0	0	0	0	0	1	1	
3	SCORE	ROW3	15	1	0	0	1	0	0	1	2	
4	SCORE	ROW4	13	1	0	0	1	0	0	1	2	
5	SCORE	ROW5	1	1	1	0	0	0	0	2	1	
6	SCORE	ROW6	9	1	1	0	0	0	0	2	1	
7	SCORE	ROW7	12	1	1	0	1	1	0	2	2	
8	SCORE	ROW8	20	1	1	0	1	1	0	2	2	
9	SCORE	ROW9	14	1	0	1	0	0	0	3	1	
10	SCORE	ROW10	8	1	0	1	0	0	0	3	1	
11	SCORE	ROW11	18	1	0	1	1	0	1	3	2	
12	SCORE	ROW12	20	1	0	1	1	0	1	3	2	

The next model is a deviations-from-means model, but this time the reference cell is the first cell (1,1). This coding is also called effects coding. The following statements produce Figure 126.54 and Figure 126.55:

```
proc transreg data=x ss2 short;
   title2 'Deviations from Means, (1,1) Reference Cell';
   model identity(y) = class(a | b / deviations zero=first);
   output replace;
run;

proc print label;
run;
```

Figure 126.54 Deviations-from-Means Model, (1,1) Reference Cell

Two-Way ANOVA Models
Deviations from Means, (1,1) Reference Cell

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	Levels	Values
a	3	1 2 3
b	2	1 2

Number of Observations Read	12
Number of Observations Used	12

Figure 126.54 *continued*

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	234.6667	46.93333	3.20	0.0946
Error	6	88.0000	14.66667		
Corrected Total	11	322.6667			

Root MSE	3.82971	R-Square	0.7273
Dependent Mean	13.33333	Adj R-Sq	0.5000
Coeff Var	28.72281		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Intercept	1	13.3333333	2133.33	2133.33	145.45	<.0001	Intercept
Class.a2	1	-2.8333333	48.17	48.17	3.28	0.1199	a 2
Class.a3	1	1.6666667	16.67	16.67	1.14	0.3274	a 3
Class.b2	1	3.0000000	108.00	108.00	7.36	0.0349	b 2
Class.a2b2	1	2.5000000	37.50	37.50	2.56	0.1609	a 2 * b 2
Class.a3b2	1	1.0000000	6.00	6.00	0.41	0.5461	a 3 * b 2

The parameter estimates are

$$
\begin{aligned}
\hat{\mu} &= \overline{y} = 13.3333 \\
\hat{\alpha}_2 &= (\overline{y}_{21} + \overline{y}_{22})/2 - \overline{y} = (5+16)/2 - 13.3333 = -2.8333 \\
\hat{\alpha}_3 &= (\overline{y}_{31} + \overline{y}_{32})/2 - \overline{y} = (11+19)/2 - 13.3333 = 1.6667 \\
\hat{\beta}_2 &= (\overline{y}_{12} + \overline{y}_{22} + \overline{y}_{32})/3 - \overline{y} = (14+16+19)/3 - 13.3333 = 3 \\
\hat{\gamma}_{22} &= \overline{y}_{22} - (\overline{y} + \hat{\alpha}_2 + \hat{\beta}_2) = 16 - (13.3333 + -2.8333 + 3) = 2.5 \\
\hat{\gamma}_{32} &= \overline{y}_{32} - (\overline{y} + \hat{\alpha}_3 + \hat{\beta}_2) = 19 - (13.3333 + 1.6667 + 3) = 1
\end{aligned}
$$

Notice that all of the parameter estimates match the corresponding estimates from the less-than-full-rank coding.

Figure 126.55 Deviations-from-Means Model, (1,1) Reference Cell, Design Matrix

Two-Way ANOVA Models
Deviations from Means, (1,1) Reference Cell

Obs	_TYPE_	_NAME_	y	Intercept	a 2	a 3	b 2	a*b 2	a*b 2	a	b
1	SCORE	ROW1	16	1	-1	-1	-1	1	1	1	1
2	SCORE	ROW2	14	1	-1	-1	-1	1	1	1	1
3	SCORE	ROW3	15	1	-1	-1	1	-1	-1	1	2
4	SCORE	ROW4	13	1	-1	-1	1	-1	-1	1	2
5	SCORE	ROW5	1	1	1	0	-1	-1	0	2	1
6	SCORE	ROW6	9	1	1	0	-1	-1	0	2	1
7	SCORE	ROW7	12	1	1	0	1	1	0	2	2
8	SCORE	ROW8	20	1	1	0	1	1	0	2	2
9	SCORE	ROW9	14	1	0	1	-1	0	-1	3	1
10	SCORE	ROW10	8	1	0	1	-1	0	-1	3	1
11	SCORE	ROW11	18	1	0	1	1	0	1	3	2
12	SCORE	ROW12	20	1	0	1	1	0	1	3	2

The following statements fit a model with an orthogonal-contrast coding and produce Figure 126.56 and Figure 126.57:

```
proc transreg data=x ss2 short;
   title2 'Orthogonal Contrast Coding';
   model identity(y) = class(a | b / orthogonal);
   output replace;
run;

proc print label;
run;
```

Figure 126.56 Orthogonal-Contrast Coding

Two-Way ANOVA Models
Orthogonal Contrast Coding

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	Levels	Values
a	3	1 2 3
b	2	1 2

Number of Observations Read	12
Number of Observations Used	12

Figure 126.56 *continued*

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	234.6667	46.93333	3.20	0.0946
Error	6	88.0000	14.66667		
Corrected Total	11	322.6667			

Root MSE	3.82971	R-Square	0.7273
Dependent Mean	13.33333	Adj R-Sq	0.5000
Coeff Var	28.72281		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Intercept	1	13.3333333	2133.33	2133.33	145.45	<.0001	Intercept
Class.a1	1	-0.2500000	0.50	0.50	0.03	0.8596	a 1
Class.a2	1	-1.4166667	48.17	48.17	3.28	0.1199	a 2
Class.b1	1	-3.0000000	108.00	108.00	7.36	0.0349	b 1
Class.a1b1	1	2.2500000	40.50	40.50	2.76	0.1476	a 1 * b 1
Class.a2b1	1	-1.2500000	37.50	37.50	2.56	0.1609	a 2 * b 1

The parameter estimates are

$$
\begin{aligned}
\hat{\mu} &= \overline{y} = 13.3333 \\
\hat{\alpha}_1 &= ((\overline{y}_{11} + \overline{y}_{12}) - (\overline{y}_{31} + \overline{y}_{32}))/4 = ((15 + 14) - (11 + 19))/4 = -0.25 \\
\hat{\alpha}_2 &= ((\overline{y}_{21} + \overline{y}_{22}) - (\overline{y}_{11} + \overline{y}_{12} + \overline{y}_{31} + \overline{y}_{32})/2)/6 \\
&= ((5 + 16) - (15 + 14 + 11 + 19)/2)/6 = -1.417 \\
\hat{\beta}_1 &= ((\overline{y}_{11} + \overline{y}_{21} + \overline{y}_{31}) - (\overline{y}_{12} + \overline{y}_{22} + \overline{y}_{32}))/6 \\
&= ((15 + 5 + 11) - (14 + 16 + 19))/6 = -3 \\
\hat{\gamma}_{11} &= (\overline{y}_{11} - \overline{y}_{12} - \overline{y}_{31} + \overline{y}_{32})/4 = (15 - 14 - 11 + 19)/4 = 2.25 \\
\hat{\gamma}_{21} &= ((-\overline{y}_{11} + \overline{y}_{12} - \overline{y}_{31} + \overline{y}_{32})/2 + (\overline{y}_{21} - \overline{y}_{22}))/6 \\
&= ((-15 + 14 - 11 + 19)/2 + (5 - 16))/6 = -1.25
\end{aligned}
$$

Figure 126.57 Orthogonal-Contrast Coding, Design Matrix

Two-Way ANOVA Models
Orthogonal Contrast Coding

Obs	_TYPE_	_NAME_	y	Intercept	a 1	a 2	b 1	a*b 1	a*b 2	a 1	b 1
1	SCORE	ROW1	16	1	1	-1	1	1	-1	1	1
2	SCORE	ROW2	14	1	1	-1	1	1	-1	1	1
3	SCORE	ROW3	15	1	1	-1	-1	-1	1	1	2
4	SCORE	ROW4	13	1	1	-1	-1	-1	1	1	2
5	SCORE	ROW5	1	1	0	2	1	0	2	2	1
6	SCORE	ROW6	9	1	0	2	1	0	2	2	1
7	SCORE	ROW7	12	1	0	2	-1	0	-2	2	2
8	SCORE	ROW8	20	1	0	2	-1	0	-2	2	2
9	SCORE	ROW9	14	1	-1	-1	1	-1	-1	3	1
10	SCORE	ROW10	8	1	-1	-1	1	-1	-1	3	1
11	SCORE	ROW11	18	1	-1	-1	-1	1	1	3	2
12	SCORE	ROW12	20	1	-1	-1	-1	1	1	3	2

The following statements fit a model with a standardized-orthogonal coding and produce Figure 126.58 and Figure 126.59:

```
proc transreg data=x ss2 short;
   title2 'Standardized-Orthogonal Coding';
   model identity(y) = class(a | b / standorth);
   output replace;
run;

proc print label;
run;
```

Figure 126.58 Standardized-Orthogonal Coding

Two-Way ANOVA Models
Standardized-Orthogonal Coding

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	Levels	Values
a	3	1 2 3
b	2	1 2

Number of Observations Read	12
Number of Observations Used	12

Figure 126.58 *continued*

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	234.6667	46.93333	3.20	0.0946
Error	6	88.0000	14.66667		
Corrected Total	11	322.6667			

Root MSE	3.82971	R-Square	0.7273
Dependent Mean	13.33333	Adj R-Sq	0.5000
Coeff Var	28.72281		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Intercept	1	13.3333333	2133.33	2133.33	145.45	<.0001	Intercept
Class.a1	1	-0.2041241	0.50	0.50	0.03	0.8596	a 1
Class.a2	1	-2.0034692	48.17	48.17	3.28	0.1199	a 2
Class.b1	1	-3.0000000	108.00	108.00	7.36	0.0349	b 1
Class.a1b1	1	1.8371173	40.50	40.50	2.76	0.1476	a 1 * b 1
Class.a2b1	1	-1.7677670	37.50	37.50	2.56	0.1609	a 2 * b 1

The parameter estimates are

$$
\begin{aligned}
\hat{\mu} &= \overline{y} = 13.3333 \\
\hat{\alpha}_1 &= (((\overline{y}_{11} + \overline{y}_{12}) - (\overline{y}_{31} + \overline{y}_{32}))/4) \times \sqrt{2/3} \\
&= (((15 + 14) - (11 + 19))/4) \times \sqrt{2/3} = -0.2041 \\
\hat{\alpha}_2 &= (((\overline{y}_{21} + \overline{y}_{22}) - (\overline{y}_{11} + \overline{y}_{12} + \overline{y}_{31} + \overline{y}_{32})/2)/6) \times \sqrt{6/3} \\
&= (((5 + 16) - (15 + 14 + 11 + 19)/2)/6) \times \sqrt{6/3} = -2.0035 \\
\hat{\beta}_1 &= (((\overline{y}_{11} + \overline{y}_{21} + \overline{y}_{31}) - (\overline{y}_{12} + \overline{y}_{22} + \overline{y}_{32}))/6) \times \sqrt{2/2} \\
&= (((15 + 5 + 11) - (14 + 16 + 19))/6) \times \sqrt{2/2} = -3 \\
\hat{\gamma}_{11} &= ((\overline{y}_{11} - \overline{y}_{12} - \overline{y}_{31} + \overline{y}_{32})/4) \times \sqrt{2/3} \times \sqrt{2/2} \\
&= ((15 - 14 - 11 + 19)/4) \times \sqrt{2/3} \times \sqrt{2/2} = 1.8371 \\
\hat{\gamma}_{21} &= (((-\overline{y}_{11} + \overline{y}_{12} - \overline{y}_{31} + \overline{y}_{32})/2 + (\overline{y}_{21} - \overline{y}_{22}))/6) \times \sqrt{6/3} \times \sqrt{2/2} \\
&= (((-15 + 14 - 11 + 19)/2 + (5 - 16))/6) \times \sqrt{6/3} \times \sqrt{2/2} = -1.7678
\end{aligned}
$$

The numerators in the square roots are sums of squares of the coded values for the unstandardized-orthogonal codings, and the denominators are the numbers of levels. These terms convert the estimates from the orthogonal contrast coding to the standardized-orthogonal coding. The term $\sqrt{2/2}$, which is 1 and could be dropped, is included in the preceding formulas to show the general pattern. Notice the regression tables for

the orthogonal-contrast coding and the standardized-orthogonal coding. Some of the coefficients are different, but the rest of the table is the same since the coded variables for the two models differ only by a constant.

Figure 126.59 Standardized-Orthogonal Coding, Design Matrix

Two-Way ANOVA Models
Standardized-Orthogonal Coding

Obs	_TYPE_	_NAME_	y	Intercept	a 1	a 2	b 1	a 1 * b 1	a 2 * b 1	a	b
1	SCORE	ROW1	16	1	1.22474	-0.70711	1	1.22474	-0.70711	1	1
2	SCORE	ROW2	14	1	1.22474	-0.70711	1	1.22474	-0.70711	1	1
3	SCORE	ROW3	15	1	1.22474	-0.70711	-1	-1.22474	0.70711	1	2
4	SCORE	ROW4	13	1	1.22474	-0.70711	-1	-1.22474	0.70711	1	2
5	SCORE	ROW5	1	1	0.00000	1.41421	1	0.00000	1.41421	2	1
6	SCORE	ROW6	9	1	0.00000	1.41421	1	0.00000	1.41421	2	1
7	SCORE	ROW7	12	1	0.00000	1.41421	-1	0.00000	-1.41421	2	2
8	SCORE	ROW8	20	1	0.00000	1.41421	-1	0.00000	-1.41421	2	2
9	SCORE	ROW9	14	1	-1.22474	-0.70711	1	-1.22474	-0.70711	3	1
10	SCORE	ROW10	8	1	-1.22474	-0.70711	1	-1.22474	-0.70711	3	1
11	SCORE	ROW11	18	1	-1.22474	-0.70711	-1	1.22474	0.70711	3	2
12	SCORE	ROW12	20	1	-1.22474	-0.70711	-1	1.22474	0.70711	3	2

Missing Values

PROC TRANSREG can estimate missing values, with or without category or monotonicity constraints, so that the regression model fit is optimized. Several approaches to missing data handling are provided. All observations with missing values in IDENTITY, CLASS, POINT, EPOINT, QPOINT, SMOOTH, PBSPLINE, PSPLINE, and BSPLINE variables are excluded from the analysis. When METHOD=UNIVARIATE (specified in the PROC TRANSREG or MODEL statement), observations with missing values in any of the independent variables are excluded from the analysis. When you specify the NOMISS *a-option*, observations with missing values in the other analysis variables are excluded. Otherwise, missing data are estimated, and the variable means are the initial estimates.

You can specify the LINEAR, OPSCORE, MONOTONE, UNTIE, SPLINE, MSPLINE, SSPLINE, LOG, LOGIT, POWER, ARSIN, BOXCOX, RANK, and EXP transformations in any combination with nonmissing values, ordinary missing values, and special missing values, as long as the nonmissing values in each variable have positive variance. No category or order restrictions are placed on the estimates of ordinary missing values. You can force missing value estimates within a variable to be identical by using special missing values (see "DATA Step Processing" in *SAS Programmers Guide: Essentials*. You can specify up to 27 categories of missing values, in which within-category estimates must be the same, by coding the missing values with ._ and .A through .Z.

You can also specify an ordering of some missing value estimates. You can use the MONOTONE= *a-option* in the PROC TRANSREG or MODEL statement to indicate a range of special missing values (a subset of the list from .A to .Z) with estimates that must be weakly ordered within each variable in which they appear. For example, if MONOTONE=AI, the nine classes, .A, .B, . . . , .I, are monotonically scored and optimally scaled just as MONOTONE transformation values are scored. In this case, category but not order restrictions are placed on the missing values ._ and .J through .Z. You can also use the UNTIE= *a-option* (in the PROC

TRANSREG or MODEL statement) to indicate a range of special missing values with estimates that must be weakly ordered within each variable in which they appear but can be untied.

The missing value estimation facilities enable you to have partitioned or mixed-type variables. For example, a variable can be considered part nominal and part ordinal. Nominal classes of otherwise ordinal variables are coded with special missing values. This feature can be useful with survey research. The class "unfamiliar with the product" in the variable "Rate your preference for 'Brand X' on a 1 to 9 scale, or if you are unfamiliar with the product, check 'unfamiliar with the product"' is an example. You can code "unfamiliar with the product" as a special missing value, such as .A. The 1s to 9s can be monotonically transformed, while no monotonic restrictions are placed on the quantification of the "unfamiliar with the product" class.

A variable specified for a LINEAR transformation, with special missing values and ordered categorical missing values, can be part interval, part ordinal, and part nominal. A variable specified for a MONOTONE transformation can have two independent ordinal parts. A variable specified for an UNTIE transformation can have an ordered categorical part and an ordered part without category restrictions. Many other mixes are possible.

Missing Values, UNTIE, and Hypothesis Tests

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/tregmod.sas.)

PROC TRANSREG can estimate missing data and monotonically transform variables while untying tied values. Estimates of ordinary missing values (.) are all permitted to be different. Analyses with UNTIE transformations, the UNTIE= *a-option*, and ordinary missing data estimation are all prone to degeneracy problems. Consider the following example. A perfect fit is found by collapsing all observations except the one with two missing values into a single value in y and x1. The following statements produce Figure 126.60:

```
title 'Missing Data';

data x;
   input y x1 x2 @@;
   datalines;
1 3 7    8 3 9    1 8 6    . . 9    3 3 9
8 5 1    6 7 3    2 7 2    1 8 2    . 9 1
;

proc transreg solve;
   model linear(y) = linear(x1 x2);
   output;
run;

proc print;
run;
```

Figure 126.60 Missing Values Example

Missing Data

Obs	_TYPE_	_NAME_	y	Ty	Intercept	x1	x2	TIntercept	Tx1	Tx2
1	SCORE	ROW1	1	2.7680	1	3	7	1	5.1233	7
2	SCORE	ROW2	8	2.7680	1	3	9	1	5.1233	9
3	SCORE	ROW3	1	2.7680	1	8	6	1	5.1233	6
4	SCORE	ROW4	.	12.5878	1	.	9	1	12.7791	9
5	SCORE	ROW5	3	2.7680	1	3	9	1	5.1233	9
6	SCORE	ROW6	8	2.7680	1	5	1	1	5.1233	1
7	SCORE	ROW7	6	2.7680	1	7	3	1	5.1233	3
8	SCORE	ROW8	2	2.7680	1	7	2	1	5.1233	2
9	SCORE	ROW9	1	2.7680	1	8	2	1	5.1233	2
10	SCORE	ROW10	.	2.7680	1	9	1	1	5.1233	1

Generally, the use of ordinary missing data estimation, the UNTIE transformation, and the UNTIE= *a-option* should be avoided, particularly with hypothesis tests. With these options, parameters are estimated based on only a single observation, and they can exert tremendous influence over the results. Each of these parameters has one model degree of freedom associated with it, so small or zero error degrees of freedom can also be a problem.

Controlling the Number of Iterations

Several *a-options* in the PROC TRANSREG or MODEL statement control the number of iterations performed. Iteration terminates when any one of the following conditions is satisfied:

- The number of iterations equals the value of the MAXITER= *a-option*.
- The average absolute change in variable scores from one iteration to the next is less than the value of the CONVERGE= *a-option*.
- The criterion change is less than the value of the CCONVERGE= *a-option*.

You can specify negative values for either convergence *a-option* if you want to define convergence only in terms of the other option. The criterion change can become negative when the data have converged, so it is numerically impossible, within machine precision, to increase the criterion. Usually, a negative criterion change is the result of very small amounts of rounding error, since the algorithms are (usually) convergent. However, there are cases where a negative criterion change is a sign of divergence, which is not necessarily an error. When you specify an SSPLINE transformation or the REITERATE or SOLVE *a-option*, divergence is perfectly normal.

When there are no monotonicity constraints and there is only one canonical variable in each set, PROC TRANSREG (with the SOLVE *a-option*) can usually find the optimal solution in only one iteration. (There are no monotonicity constraints when none of the following is specified: MONOTONE, MSPLINE, or UNTIE transformation or the UNTIE= or MONOTONE= *a-option*. There is only one canonical variable in each set when METHOD=MORALS or METHOD=UNIVARIATE, or when METHOD=REDUNDANCY with only one dependent variable, or when METHOD=CANALS and NCAN=1.)

The initialization iteration is number 0. When there are no monotonicity constraints and there is only one canonical variable in each set, the next iteration shows no change, and iteration stops. At least two iterations (0 and 1) are performed with the SOLVE *a-option* even if nothing changes in iteration 0. The MONOTONE, MSPLINE, and UNTIE variables are not transformed by the canonical initialization. Note that divergence with the SOLVE *a-option*, particularly in the second iteration, is not an error. The initialization iteration is slower and uses more memory than other iterations. However, for many models, specifying the SOLVE *a-option* can greatly decrease the amount of time required to find the optimal transformations.

You can increase the number of iterations to ensure convergence by increasing the value of the MAXITER= *a-option* and decreasing the value of the CONVERGE= *a-option*. Since the average absolute change in standardized variable scores seldom decreases below 1E–11, you should not specify a value for the CONVERGE= *a-option* less than 1E–8 or 1E–10. Most of the data changes occur during the first few iterations, but the data can still change after 50 or even 100 iterations. You can try different combinations of values for the CONVERGE= and MAXITER= *a-options* to ensure convergence without extreme overiteration. If the data do not converge with the default specifications, try CONVERGE=1E–8 and MAXITER=50, or CONVERGE=1E–10 and MAXITER=200. Note that you can specify the REITERATE *a-option* to start iterating where the previous analysis stopped.

Using the REITERATE Algorithm Option

You can use the REITERATE *a-option* to perform additional iterations when PROC TRANSREG stops before the data have adequately converged. For example, suppose that you execute the following step:

```
proc transreg data=a;
   model mspline(y) = mspline(x1-x5);
   output out=b coefficients;
run;
```

If the transformations do not converge in the default 30 iterations, you can perform more iterations without repeating the first 30 iterations, as follows:

```
proc transreg data=b reiterate;
   model mspline(y) = mspline(x1-x5);
   output out=b coefficients;
run;
```

Note that a WHERE statement is not necessary to exclude the coefficient observations. They are automatically excluded because their _TYPE_ value is not SCORE.

You can also use the REITERATE *a-option* to specify starting values other than the original values for the transformations. Providing alternate starting points might help avoid local optima. Here are two examples:

```
proc transreg data=a;
   model rank(y) = rank(x1-x5);
   output out=b;
run;

proc transreg data=b reiterate;
   /* Use ranks as the starting point. */
   model mspline(y) = mspline(x1-x5);
   output out=c coefficients;
```

```
run;

data b;
   set a;
   array tx[6] ty tx1-tx5;
   do j = 1 to 6;
      tx[j] = normal(7);
   end;
run;

proc transreg data=b reiterate;
   /* Use a random starting point. */
   model mspline(y) = mspline(x1-x5);
   output out=c coefficients;
run;
```

Note that divergence with the REITERATE *a-option*, particularly in the second iteration, is not an error since the initial transformation is not required to be a valid member of the transformation family. When you specify the REITERATE *a-option*, the iteration does not terminate when the criterion change is negative during the first 10 iterations.

Avoiding Constant Transformations

There are times when the optimal scaling produces a constant transformed variable. This can happen with the MONOTONE, UNTIE, and MSPLINE transformations when the target is negatively correlated with the original input variable. It can happen with all transformations when the target is uncorrelated with the original input variable. When this happens, the procedure modifies the target to avoid a constant transformation. This strategy avoids certain nonoptimal solutions.

If the transformation is monotonic and a constant transformed variable results, the procedure multiplies the target by –1 and tries the optimal scaling again. If the transformation is not monotonic or if the multiplication by –1 did not help, the procedure tries using a random target. If the transformation is still constant, the previous nonconstant transformation is retained. When a constant transformation is avoided by any strategy, this message is displayed: "A constant transformation was avoided for *name*."

With extreme collinearity, small amounts of rounding error might interact with the instability of the coefficients to produce target vectors that are not positively correlated with the original scaling. If a regression coefficient for a variable is zero, the formula for the target for that variable contains a zero divide. In a multiple regression model, after many iterations, one independent variable can be scaled the same way as the current scaling of the dependent variable, so the other independent variables have coefficients of zero. When the constant transformation warning appears, you should interpret your results with extreme caution, and recheck your model.

Constant Variables

Constant and almost constant variables are zeroed and ignored. When constant variables are expected and should not be zeroed, specify the NOZEROCONSTANT *a-option*.

Character OPSCORE Variables

Character OPSCORE variables are replaced by a numeric variable containing category numbers before the iterations, and the character values are discarded. Only the first eight characters are considered in determining category membership. If you want the original character variable in the output data set, give it a different name in the OPSCORE specification (OPSCORE(x / name=(x2)) and name the original variable in the ID statement (ID x;).

Convergence and Degeneracies

When you specify the SSPLINE transformation, divergence is normal. The rest of this section assumes that you did not specify SSPLINE. For all the methods available in PROC TRANSREG, the algorithms are convergent, in terms of both the criterion being optimized and the parameters being estimated. The value of the criterion being maximized (squared multiple correlation, average squared multiple correlation, or average squared canonical correlation) can, theoretically, never decrease from one iteration to the next. The values of the parameters being solved for (the scores and weights of the transformed variables) become stable after sufficient iteration.

In practice, the criterion being maximized can decrease with overiteration. When the statistic has very nearly reached its maximum, further iterations might report a decrease in the criterion in the last few decimal places. This is a normal result of very small amounts of rounding error. By default, iteration terminates when this occurs because, by default, CCONVERGE=0.0. Specifying CCONVERGE=–1, an impossible change, turns off this check for convergence.

Even though the algorithms are convergent, they might not converge to a global optimum. Also, under extreme circumstances, the solution might degenerate. Because two points always form a straight line, the algorithms sometimes try to reach this degenerate optimum. This sometimes occurs when one observation is an ordinal outlier (when one observation has the extreme rank on all variables). The algorithm can reach an optimal solution that ties all other categories producing two points. Similar results can occur when there are many missing values. More generally, whenever there are very few constraints on the scoring of one or more points, degeneracies can be a problem. In a well-behaved analysis, the maximum data change, average data change, and criterion change all decrease at a rapid rate with each iteration. When the rate of change increases for several iterations, the solution might be degenerating.

Implicit and Explicit Intercepts

Depending on several options, the model intercept is nonzero, zero, or implicit, or there is no intercept. Ordinarily, the model contains an explicit nonzero intercept, and the Intercept variable in the OUT= data set contains ones. When TSTANDARD=CENTER or TSTANDARD=Z is specified, the model contains an explicit, zero intercept and the Intercept variable contains zeros. When METHOD=CANALS, the model is fit with centered variables and the Intercept variable is set to missing.

If you specify CLASS with ZERO=NONE or BSPLINE for one or more independent variables, and TSTANDARD=NOMISS or TSTANDARD=ORIGINAL (the default), an implicit intercept model is fit. The intercept is implicit in a set of the independent variables since there exists a set of independent variables the sum of which is a column of ones. All statistics are mean corrected. The implicit intercept is not an option; it is implied by the model. Specifying SMOOTH or PBSPLINE also implies an implicit intercept model.

With METHOD=CANALS, the Intercept variable contains the *canonical intercept* for canonical coefficients observations: $\hat{\beta}_0 = \overline{\mathbf{y}}'\hat{\boldsymbol{\alpha}} - \overline{\mathbf{x}}'\hat{\boldsymbol{\beta}}$ where $\mathbf{Y}\hat{\boldsymbol{\alpha}} \approx \mathbf{X}\hat{\boldsymbol{\beta}}$.

Passive Observations

Observations can be excluded from the analysis for several reasons; these include zero weight; zero frequency; missing values in variables designated IDENTITY, CLASS, POINT, EPOINT, QPOINT, SMOOTH, PBSPLINE, PSPLINE, or BSPLINE; and missing values with the NOMISS *a-option* specified. These observations are passive in that they do not contribute to determining transformations, R square, sums of squares, degrees of freedom, and so on. However, some information can be computed for them. For example, if no independent variable values are missing, predicted values and redundancy variable values can both be computed. Residuals can be computed for observations with a nonmissing dependent and nonmissing predicted value. Canonical variables for dependent variables can be computed when no dependent variables are missing; canonical variables for independent variables can be computed when no independent variables are missing, and so on. Passive observations in the OUT= data set have a blank value for _TYPE_.

Point Models

The expanded set of independent variables generated from the POINT, EPOINT, and QPOINT expansions can be used to perform ideal point regressions (Carroll 1972) and compute ideal point coordinates for plotting in a biplot (Gabriel 1981). The three types of ideal point coordinates can all be described as transformed coefficients. Assume that m independent variables are specified in one of the three point expansions. Let $\mathbf{b}'$ be a $1 \times m$ row vector of coefficients for these variables and one of the dependent variables. Let $\mathbf{R}$ be a matrix created from the coefficients of the extra variables. When coordinates are requested with the MPC, MEC, or MQC *o-option*, $\mathbf{b}'$ and $\mathbf{R}$ are created from multiple regression coefficients. When coordinates are requested with the CPC, CEC, or CQC *o-option*, $\mathbf{b}'$ and $\mathbf{R}$ are created from canonical coefficients.

If you specify the POINT expansion in the MODEL statement, $\mathbf{R}$ is an $m \times m$ identity matrix times the coefficient for the sums of squares (_ISSQ_) variable. If you specify the EPOINT expansion, $\mathbf{R}$ is an $m \times m$ diagonal matrix of coefficients from the squared variables. If you specify the QPOINT expansion, $\mathbf{R}$ is an $m \times m$ symmetric matrix of coefficients from the squared variables on the diagonal and crossproduct variables off the diagonal. The MPC, MEC, MQC, CPC, CEC, and CQC ideal point coordinates are defined

as $-0.5\mathbf{b}'\mathbf{R}^{-1}$. When $\mathbf{R}$ is singular, the ideal point coordinates are infinitely far away and are set to missing, so you should try a simpler version of the model. The version that is simpler than the POINT model is the vector model, where no extra variables are created. In the vector model, designate all independent variables as IDENTITY. Then draw vectors from the origin to the COEFFICIENTS points.

Typically, when you request ideal point coordinates, the MODEL statement should consist of a single transformation for the dependent variables (usually IDENTITY, MONOTONE, or MSPLINE) and a single expansion for the independent variables (one of POINT, EPOINT, or QPOINT).

Redundancy Analysis

Redundancy analysis (Stewart and Love 1968) is a principal component analysis of multivariate regression predicted values. These first steps show the redundancy analysis results produced by PROC TRANSREG. The specification TSTANDARD=Z sets all variables to mean zero and variance one. METHOD=REDUNDANCY specifies redundancy analysis and outputs the redundancy variables to the OUT= data set. The MREDUNDANCY *o-option* outputs two sets of redundancy analysis coefficients to the OUT= data set.

The following statements produce Figure 126.61:

```
title 'Redundancy Analysis';

data x;
   input y1-y3 x1-x4;
   datalines;
 6  8  8 15 18 26 27
 1 12 16 18  9 20  8
 5  6 15 20 17 29 31
 6  9 15 14 10 16 22
 7  5 12 14  6 13  9
 3  6  7  2 14 26 22
 3  5  9 13 18 10 22
 6  3 11  3 15 22 29
 6  3  7 10 20 21 27
 7  5  9  8 10 12 18
;

proc transreg data=x tstandard=z method=redundancy;
   model identity(y1-y3) = identity(x1-x4);
   output out=red mredundancy replace;
run;

proc print data=red(drop=Intercept);
   format _numeric_ 4.1;
run;
```

Figure 126.61 Redundancy Analysis Example

Redundancy Analysis

Obs	_TYPE_	_NAME_	y1	y2	y3	x1	x2	x3	x4	Red1	Red2	Red3
1	SCORE	ROW1	0.5	0.6	-0.8	0.6	0.9	1.0	0.7	0.2	-0.5	-0.9
2	SCORE	ROW2	-2.0	2.1	1.5	1.1	-1.0	0.1	-1.7	1.6	-1.5	0.4
3	SCORE	ROW3	0.0	-0.1	1.2	1.4	0.7	1.5	1.2	1.0	0.8	-1.3
4	SCORE	ROW4	0.5	1.0	1.2	0.4	-0.8	-0.5	0.1	0.5	1.7	0.1
5	SCORE	ROW5	1.0	-0.4	0.3	0.4	-1.6	-1.0	-1.6	1.0	0.1	0.9
6	SCORE	ROW6	-1.0	-0.1	-1.1	-1.6	0.1	1.0	0.1	-0.8	-0.9	1.4
7	SCORE	ROW7	-1.0	-0.4	-0.6	0.2	0.9	-1.5	0.1	-1.0	-0.4	-1.3
8	SCORE	ROW8	0.5	-1.2	0.0	-1.5	0.3	0.4	1.0	-1.2	0.8	0.7
9	SCORE	ROW9	0.5	-1.2	-1.1	-0.3	1.3	0.2	0.7	-1.0	-0.9	-0.8
10	SCORE	ROW10	1.0	-0.4	-0.6	-0.6	-0.8	-1.1	-0.4	-0.4	0.8	0.7
11	M REDUND	Red1	.	.	.	0.7	-0.6	0.4	-0.1	.	.	.
12	M REDUND	Red2	.	.	.	0.3	-1.5	-0.6	1.9	.	.	.
13	M REDUND	Red3	.	.	.	-0.7	-0.7	0.3	-0.3	.	.	.
14	R REDUND	x1	.	.	.	.	.	.	.	0.8	-0.0	-0.6
15	R REDUND	x2	.	.	.	.	.	.	.	-0.6	-0.2	-0.7
16	R REDUND	x3	.	.	.	.	.	.	.	0.1	-0.2	-0.1
17	R REDUND	x4	.	.	.	.	.	.	.	-0.5	0.3	-0.5

The _TYPE_='SCORE' observations of the Red1–Red3 variables contain the redundancy variables. The nonmissing "M REDUND" values are coefficients for predicting the redundancy variables from the independent variables. The nonmissing "R REDUND" values are coefficients for predicting the independent variables from the redundancy variables.

The next steps show how to generate the same results manually. The data set is standardized, predicted values are computed, and principal components of the predicted values are computed. The following statements produce the redundancy variables, shown in Figure 126.62:

```
proc standard data=x out=std m=0 s=1;
   title2 'Manually Generate Redundancy Variables';
run;

proc reg noprint data=std;
   model y1-y3 = x1-x4;
   output out=p p=ay1-ay3;
quit;

proc princomp data=p cov noprint std out=p;
   var ay1-ay3;
run;

proc print data=p(keep=Prin:);
   format _numeric_ 4.1;
run;
```

Figure 126.62 Redundancy Analysis Example

Redundancy Analysis
Manually Generate Redundancy Variables

Obs	Prin1	Prin2	Prin3
1	0.2	-0.5	-0.9
2	1.6	-1.5	0.4
3	1.0	0.8	-1.3
4	0.5	1.7	0.1
5	1.0	0.1	0.9
6	-0.8	-0.9	1.4
7	-1.0	-0.4	-1.3
8	-1.2	0.8	0.7
9	-1.0	-0.9	-0.8
10	-0.4	0.8	0.7

The following statements produce the coefficients for predicting the redundancy variables from the independent variables, shown in Figure 126.63:

```
proc reg data=p outest=redcoef noprint;
   title2 'Manually Create Redundancy Coefficients';
   model Prin1-Prin3 = x1-x4;
quit;

proc print data=redcoef(keep=x1-x4);
   format _numeric_ 4.1;
run;
```

Figure 126.63 Redundancy Analysis Example

Redundancy Analysis
Manually Create Redundancy Coefficients

Obs	x1	x2	x3	x4
1	0.7	-0.6	0.4	-0.1
2	0.3	-1.5	-0.6	1.9
3	-0.7	-0.7	0.3	-0.3

The following statements produce the coefficients for predicting the independent variables from the redundancy variables, shown in Figure 126.64:

```
proc reg data=p outest=redcoef2 noprint;
   title2 'Manually Create Other Coefficients';
   model x1-x4 = prin1-prin3;
quit;

proc print data=redcoef2(keep=Prin1-Prin3);
   format _numeric_ 4.1;
run;
```

Figure 126.64 Redundancy Analysis Example

Redundancy Analysis
Manually Create Other Coefficients

Obs	Prin1	Prin2	Prin3
1	0.8	-0.0	-0.6
2	-0.6	-0.2	-0.7
3	0.1	-0.2	-0.1
4	-0.5	0.3	-0.5

Optimal Scaling

An alternating least squares optimal scaling algorithm can be divided into two major stages. The first major stage estimates the parameters of the linear model. These parameters are used to create the predicted values or target for each variable that can be transformed. Each target minimizes squared error (as explained in the discussion of the algorithms in *SAS Technical Report R-108*). The definition of the target depends on many factors, such as whether a variable is independent or dependent, which algorithm is used (for example, regression, redundancy, CANALS, or principal components), and so on. The definition of the target is independent of the transformation family you specify for the variable. However, the target values for a variable typically do not fit the prescribed transformation family for the variable. They might not have the right category structure; they might not have the right order; they might not be a linear combination of the columns of a B-spline basis; and so on.

The second major stage is optimal scaling. Optimal scaling can be defined as a possibly constrained, least squares regression problem. When you specify an optimal transformation, or when missing data are estimated for any variable, the full representation of the variable is not simply a vector; it is a matrix with more than one column. The optimal scaling phase finds the vector that is a linear combination of the columns of this matrix that is closest to the target (in terms of minimum squared error), among those that do not violate any of the constraints imposed by the transformation family. Optimal scaling methods are independent of the data analysis method that generated the target. In all cases, optimal scaling can be accomplished by creating a design matrix based on the original scaling of the variable and the transformation family specified for that variable. The optimally scaled variable is a linear combination of the columns of the design matrix. The coefficients of the linear combination are found by using (possibly constrained) least squares. Many optimal scaling problems are solved without actually constructing design and projection matrices. The next two sections describe the algorithms used by PROC TRANSREG for optimal scaling. The first section discusses optimal scaling for OPSCORE, MONOTONE, UNTIE, and LINEAR transformations, including how missing values are handled. The second section addresses SPLINE and MSPLINE transformations.

OPSCORE, MONOTONE, UNTIE, and LINEAR Transformations

Two vectors of information are needed to produce the optimally scaled variable: the initial variable scaling vector $\mathbf{x}$ and the target vector $\mathbf{y}$. For convenience, both vectors are first sorted on the values of the initial scaling vector. If you request an UNTIE transformation, the target vector is sorted within ties in the initial scaling vector. The normal SAS collating sequence for missing and nonmissing values is used. Sorting simply permits the constraints to be specified in terms of relationships among adjoining coefficients. The sorting process partitions $\mathbf{x}$ and $\mathbf{y}$ into missing and nonmissing parts $(\mathbf{x}_m'\mathbf{x}_n')'$, and $(\mathbf{y}_m'\mathbf{y}_n')'$.

Next, PROC TRANSREG determines category membership. Every ordinary missing value (.) forms a separate category. (Three ordinary missing values form three categories.) Every special missing value within the range specified in the UNTIE= *a-option* forms a separate category. (If UNTIE= BC and there are three .B and two .C missing values, five categories are formed from them.) For all other special missing values, a separate category is formed for each different value. (If there are four .A missing values, one category is formed from them.)

Each distinct nonmissing value forms a separate category for OPSCORE and MONOTONE transformations (1 1 1 2 2 3 form three categories). Each nonmissing value forms a separate category for all other transformations (1 1 1 2 2 3 form six categories). When category membership is determined, category means are computed. Here is an example:

$\mathbf{x}$:	`(. . .A .A .B 1 1 1 2 2 3 3 3 4)'`
$\mathbf{y}$:	`(5 6 2 4 2 1 2 3 4 6 4 5 6 7)'`
OPSCORE and MONOTONE means:	`(5 6 3 2 2 5 5 7)'`
other means:	`(5 6 3 2 1 2 3 4 6 4 5 6 7)'`

The category means are the coefficients of a category indicator design matrix. The category means are the Fisher (1938) optimal scores. For MONOTONE and UNTIE transformations, order constraints are imposed on the category means for the nonmissing partition by merging categories that are out of order. The algorithm checks upward until an order violation is found, and then averages downward until the order violation is averaged away. (The average of $\bar{x}_1$ computed from n_1 observations and $\bar{x}_2$ computed from n_2 observations is $(n_1\bar{x}_1 + n_2\bar{x}_2)/(n_1 + n_2)$.) The MONOTONE algorithm (Kruskal 1964, secondary approach to ties) for this example with means for the nonmissing values $(2\ 5\ 5\ 7)'$ would do the following checks: $2 < 5$: OK, $5 = 5$: OK, $5 < 7$: OK. The means are in the proper order, so no work is needed.

The UNTIE transformation (Kruskal 1964, primary approach to ties) uses the same algorithm on the means of the nonmissing values $(1\ 2\ 3\ 4\ 6\ 4\ 5\ 6\ 7)'$ but with different results for this example: $1 < 2$: OK, $2 < 3$: OK, $3 < 4$: OK, $4 < 6$: OK, $6 > 4$: average 6 and 4 and replace 6 and 4 by the average. The new means of the nonmissing values are $(1\ 2\ 3\ 4\ 5\ 5\ 5\ 6\ 7)'$. The check resumes: $4 < 5$: OK, $5 = 5$: OK, $5 = 5$: OK, $5 < 6$: OK, $6 < 7$: OK. If some of the special missing values are ordered, the upward-checking, downward-averaging algorithm is applied to them also, independently of the other missing and nonmissing partitions. When the means conform to any required category or order constraints, an optimally scaled vector is produced from the means. The following example results from a MONOTONE transformation:

$\mathbf{x}$:	`(. . .A .A .B 1 1 1 2 2 3 3 3 4)'`
$\mathbf{y}$:	`(5 6 2 4 2 1 2 3 4 6 4 5 6 7)'`
result:	`(5 6 3 3 2 2 2 2 5 5 5 5 5 7)'`

The upward-checking, downward-averaging algorithm is equivalent to creating a category indicator design matrix, solving for least squares coefficients with order constraints, and then computing the linear combination of design matrix columns.

For the optimal transformation LINEAR and for nonoptimal transformations, missing values are handled as just described. The nonmissing target values are regressed onto the matrix defined by the nonmissing initial scaling values and an intercept. In this example, the target vector $y_n = (1\ 2\ 3\ 4\ 6\ 4\ 5\ 6\ 7)'$ is regressed onto the design matrix

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 & 2 & 3 & 3 & 3 & 4 \end{bmatrix}'$$

Although only a linear transformation is performed, the effect of a linear regression optimal scaling is not eliminated by the later standardization step (unless the variable has no missing values). In the presence of missing values, the linear regression is necessary to minimize squared error.

SPLINE and MSPLINE Transformations

The missing portions of variables subjected to SPLINE or MSPLINE transformations are handled the same way as for OPSCORE, MONOTONE, UNTIE, and LINEAR transformations (see the previous section). The nonmissing partition is handled by first creating a B-spline basis of the specified degree with the specified knots for the nonmissing partition of the initial scaling vector and then regressing the target onto the basis. The optimally scaled vector is a linear combination of the B-spline basis vectors. Ordinary least squares regression coefficients are used. An algorithm for generating the B-spline basis is given in De Boor (1978, pp. 134–135). B-splines are both a computationally accurate and efficient way of constructing a basis for piecewise polynomials; however, they are not the most natural method of describing splines.

Consider an initial scaling vector $x = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9)'$ and a degree-three spline with interior knots at 3.5 and 6.5. The B-spline basis for the transformation is the left matrix, and the natural piecewise polynomial spline basis is the right matrix.

B-Spline Basis

$$\begin{bmatrix} 1.000 & 0.000 & 0.000 & 0.000 & 0 & 0 \\ 0.216 & 0.608 & 0.167 & 0.009 & 0 & 0 \\ 0.008 & 0.458 & 0.461 & 0.073 & 0 & 0 \\ 0 & 0.172 & 0.585 & 0.241 & 0.001 & 0 \\ 0 & 0.037 & 0.463 & 0.463 & 0.037 & 0 \\ 0 & 0.001 & 0.241 & 0.585 & 0.172 & 0 \\ 0 & 0 & 0.073 & 0.461 & 0.458 & 0.008 \\ 0 & 0 & 0.009 & 0.167 & 0.608 & 0.216 \\ 0 & 0 & 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix}$$

Piecewise Polynomial Splines

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 2 & 4 & 8 & 0 & 0 \\ 1 & 3 & 9 & 27 & 0 & 0 \\ 1 & 4 & 16 & 64 & 0.125 & 0 \\ 1 & 5 & 25 & 125 & 3.375 & 0 \\ 1 & 6 & 36 & 216 & 15.625 & 0 \\ 1 & 7 & 49 & 343 & 42.875 & 0.125 \\ 1 & 8 & 64 & 512 & 91.125 & 3.375 \\ 1 & 9 & 81 & 729 & 166.375 & 15.625 \end{bmatrix}$$

The two matrices span the same column space. The natural basis has an intercept, a linear term, a quadratic term, a cubic term, and two more terms since there are two interior knots. These terms are generated (for knot k and **x** element x) by the formula $(x-k)^3 \times I_{(x>k)}$. The indicator variable $I_{(x>k)}$ evaluates to 1.0 if x is greater than k and to 0.0 otherwise. If knot k had been repeated, there would be a $(x-k)^2 \times I_{(x>k)}$ term

also. Notice that the fifth column makes no contribution to the curve before 3.5, makes zero contribution at 3.5 (the transformation is continuous), and makes an increasing contribution beyond 3.5. The same pattern of results holds for the last term with knot 6.5. The coefficient of the fifth column represents the change in the cubic portion of the curve after 3.5. The coefficient of the sixth column represents the change in the cubic portion of the curve after 6.5.

The numbers in the B-spline basis do not have a simple interpretation like the numbers in the natural piecewise polynomial basis. The B-spline basis has a diagonally banded structure. The band shifts one column to the right after every knot. The number of entries in each row that can potentially be nonzero is one greater than the degree. The elements within a row always sum to one. The B-spline basis is accurate because of the smallness of the numbers and the lack of extreme collinearity inherent in the natural polynomials. B-splines are efficient because PROC TRANSREG can take advantage of the sparseness of the B-spline basis when it accumulates crossproducts. The number of required multiplications and additions to accumulate the crossproduct matrix does not increase with the number of knots but does increase with the degree of the spline, so it is much more computationally efficient to increase the number of knots than to increase the degree of the polynomial.

MSPLINE transformations are handled like SPLINE transformations except that constraints are placed on the coefficients to ensure monotonicity. When the coefficients of the B-spline basis are monotonically increasing, the transformation is monotonically increasing. When the polynomial degree is two or less, monotone coefficient splines, integrated splines (Winsberg and Ramsay 1980), and the general class of all monotone splines are equivalent.

Specifying the Number of Knots

Keep the number of knots small (usually less than 10, although you can specify more). A degree-three spline with nine knots, one at each decile, can closely follow a large variety of curves. Each spline transformation of degree p with q knots fits a model with $p + q$ parameters. The total number of parameters should be much less than the number of observations. Usually in regression analyses, it is recommended that there be at least five or ten observations for each parameter in order to get stable results. For example, when spline transformations of degree three with nine knots are requested for six variables, the number of observations in the data set should be at least 5 or 10 times 72 (since $6 \times (3 + 9)$ is the total number of parameters). The overall model can also have a parameter for the intercept and one or more parameters for each nonspline variable in the model.

Increasing the number of knots gives the spline more freedom to bend and follow the data. Increasing the degree also gives the spline more freedom, but to a lesser extent. Specifying a large number of knots is much better than increasing the degree beyond three.

When you specify NKNOTS=q for a variable with n observations, then each of the $q + 1$ segments of the spline contains $n/(q + 1)$ observations on the average. When you specify KNOTS=number-list, make sure that there is a reasonable number of observations in each interval.

The following statements find a cubic polynomial transformation of x and no transformation of y:

```
proc transreg;
   model identity(y)=spline(x);
   output;
run;
```

The following statements find a cubic-spline transformation for x that consists of the weighted sum of a single constant, a single straight line, a quadratic curve for the portion of the variable less than 3.0, a different quadratic curve for the portion greater than 3.0 (since the 3.0 knot is repeated), and a different cubic curve for each of the intervals: (minimum to 1.5), (1.5 to 2.4), (2.4 to 3.0), (3.0 to 4.0), and (4.0 to maximum):

```
proc transreg;
   model identity(y)=spline(x / knots=1.5 2.4 3.0 3.0 4.0);
   output;
run;
```

The transformation is continuous everywhere, its first derivative is continuous everywhere, its second derivative is continuous everywhere except at 3.0, and its third derivative is continuous everywhere except at 1.5, 2.4, 3.0, and 4.0.

The following statements find a quadratic spline transformation that consists of a polynomial $\mathsf{x_t} = b_0 + b_1\mathsf{x} + b_2\mathsf{x}^2$ for the range ($\mathsf{x} < 3.0$) and a completely different polynomial $\mathsf{x_t} = b_3 + b_4\mathsf{x} + b_5\mathsf{x}^2$ for the range ($\mathsf{x} > 3.0$):

```
proc transreg;
   model identity(y)=spline(x / knots=3 3 3 degree=2);
   output;
run;
```

The two curves are not required to be continuous at 3.0.

The following statements categorize y into 10 intervals and find a step-function transformation:

```
proc transreg;
   model identity(y)=spline(x / degree=0 nknots=9);
   output;
run;
```

One aspect of this transformation family is unlike all other optimal transformation families. The initial scaling of the data does not fit the restrictions imposed by the transformation family. This is because the initial variable can be continuous, but a discrete step-function transformation is sought. Zero-degree spline variables are categorized before the first iteration.

The following statements find a continuous, piecewise linear transformation of x:

```
proc transreg;
   model identity(y)=spline(x / degree=1 nknots=8);
   output;
run;
```

SPLINE, BSPLINE, and PSPLINE Comparisons

SPLINE is a transformation. It takes a variable as input and produces a transformed variable as output. Internally, with SPLINE, a B-spline basis is used to find the transformation, which is a linear combination of the columns of the B-spline basis. However, with SPLINE, the basis is not made available in any output.

BSPLINE is an expansion. It takes a variable as input and produces more than one variable as output. The output variables are the same B-spline basis that is used internally by SPLINE.

PSPLINE is an expansion. It takes a variable as input and produces more than one variable as output. The difference between PSPLINE and BSPLINE is that PSPLINE produces a piecewise polynomial, whereas BSPLINE produces a B-spline. A matrix consisting of a piecewise polynomial basis and an intercept spans the same space as the B-spline matrix, but the basis vectors are quite different. The numbers in the piecewise polynomials can get quite large; the numbers in the B-spline basis range between 0 and 1. There are many more zeros in the B-spline basis.

Interchanging SPLINE, BSPLINE, and PSPLINE should have no effect on the fit of the overall model except for the fact that PSPLINE is much more prone to numerical problems. Similarly, interchanging a CLASS expansion and an OPSCORE transformation should have no effect on the fit of the overall model.

Hypothesis Tests

PROC TRANSREG has a set of options for testing hypotheses in models with a single dependent variable. The TEST *a-option* produces an ANOVA table. It tests the null hypothesis that the vector of coefficients for all of the transformations is zero. The SS2 *a-option* produces a regression table with Type II tests of the contribution of each transformation to the overall model. In some cases, exact tests are provided; in other cases, the tests are approximate, liberal, or conservative.

There are two reasons why it is typically not appropriate to test hypotheses by using the output from PROC TRANSREG as input to other procedures such as the REG procedure. First, PROC REG has no way of determining how many degrees of freedom were used for each transformation. Second, the Type II sums of squares for the tests of the individual regression coefficients are not correct for the transformation regression model because PROC REG, as it evaluates the effect of each variable, cannot change the transformations of the other variables. PROC TRANSREG uses the correct degrees of freedom and sums of squares.

In an ordinary univariate linear model, there is one parameter for each independent variable, including the intercept. In the transformation regression model, many of the "variables" are used internally in the bases for the transformations. Each basis column has one parameter or *scoring* coefficient, and each linearly independent column has one model degree of freedom associated with it. Coefficients applied to transformed variables, *model coefficients*, do not enter into the degrees-of-freedom calculations. They are byproducts of the standardizations and can be absorbed into the transformations by specifying the ADDITIVE *a-option*. The word *parameter* is reserved for model and scoring coefficients that have a degree of freedom associated with them.

For expansions, there is one model parameter for each variable created by the expansion (except for all missing CLASS columns and expansions that have an implicit intercept). Each IDENTITY variable has one model parameter. If there are m POINT variables, they expand to $m + 1$ variables and hence have $m + 1$ model parameters. For m EPOINT variables, there are $2m$ model parameters. For m QPOINT variables, there are $m(m + 3)/2$ model parameters. If a variable with m categories is designated CLASS, there are

$m - 1$ parameters. For BSPLINE and PSPLINE variables of DEGREE=n with NKNOTS=k, there are $n + k$ parameters. Note that one of the $n + k + 1$ BSPLINE columns and one of the m CLASS(variable / ZERO=NONE) columns are not counted due to the implicit intercept.

There are scoring parameters for missing values in nonexcluded observations. Each ordinary missing value (.) has one scoring parameter. Each different special missing value (._ and .A through .Z) within each variable has one scoring parameter. Missing values specified in the UNTIE= and MONOTONE= options follow the rules for UNTIE and MONOTONE transformations, which are described later in this chapter.

For all nonoptimal transformations (LOG, LOGIT, ARSIN, POWER, EXP, RANK, BOXCOX), there is one parameter per variable in addition to any missing value scoring parameters.

For SPLINE, OPSCORE, and LINEAR transformations, the number of scoring parameters is the number of basis columns that are used internally to find the transformations minus 1 for the intercept. The number of scoring parameters for SPLINE variables is the same as the number of model parameters for BSPLINE and PSPLINE variables. If DEGREE=n and NKNOTS=k, there are $n + k$ scoring parameters. The number of scoring parameters for OPSCORE, SMOOTH, and SSPLINE variables is the same as the number of model parameters for CLASS variables. If there are m categories, there are $m - 1$ scoring parameters. There is one parameter for each LINEAR variable. For SPLINE, OPSCORE, LINEAR, MONOTONE, UNTIE, and MSPLINE transformations, missing value scoring parameters are computed as described previously with the nonoptimal transformations.

The number of scoring parameters for MONOTONE, UNTIE, and MSPLINE transformations is less precise than for SPLINE, OPSCORE, and LINEAR transformations. One way of handling a MONOTONE transformation is to treat it as if it were the same as an OPSCORE transformation. If there are m categories, there are $m - 1$ potential scoring parameters. However, there are typically fewer than $m - 1$ unique parameter estimates, since some of those $m - 1$ scoring parameter estimates might be tied during the optimal scaling to impose the order constraints. Imposing ties on the scoring parameter estimates is equivalent to fitting a model with fewer parameters. So there are two available scoring parameter counts: $m - 1$ and a smaller number that is determined during the analysis. Using $m - 1$ as the model degrees of freedom for MONOTONE variables (treating OPSCORE and MONOTONE transformations the same way) is *conservative*, since the MONOTONE scoring parameter estimates are more restricted than the OPSCORE scoring parameter estimates. Using the smaller count (the number of scoring parameter estimates that are different, minus 1 for the intercept) in the model degrees of freedom is *liberal*, since the data and the model together are being used to determine the number of parameters. PROC TRANSREG reports tests that use both liberal and conservative degrees of freedom to provide lower and upper bounds on the "true" p-values.

For the UNTIE transformation, the conservative scoring parameter count is the number of distinct observations, whereas the liberal scoring parameter count is the number of scoring parameter estimates that are different, minus 1 for the intercept. Hence, when you specify UNTIE, conservative tests have zero error degrees of freedom unless there are replicated observations.

For MSPLINE variables of DEGREE=n and NKNOTS=k, the conservative scoring parameter count is $n + k$, whereas the liberal parameter count is the number of scoring parameter estimates that are different, minus 1 for the intercept. A liberal degrees of freedom of 1 does not necessarily imply a linear transformation. It implies only that n plus k minus the number of ties imposed equals 1. An example of a one-degree-of-freedom nonlinear transformation is a two-piece linear transformation in which the slope of one piece is 0.

The number of scoring parameters is determined during each iteration. After the last iteration, enough information is available for the TEST *a-option* to produce an ANOVA table that reports the overall fit of the model. If you specify the SS2 *a-option*, further iterations are necessary to test the contribution of each transformation to the overall model.

The liberal tests do not compensate for overparameterization. For example, requesting a spline transformation with k knots when a linear transformation will suffice results in "liberal" tests that are actually conservative because too many degrees of freedom are being used for the transformations. To avoid this problem, use as few knots as possible.

In ordinary multiple regression, an F test of the null hypothesis that the coefficient for variable x_j is zero can be constructed by comparing two linear models. One model is the full model with all parameters, and the other is a reduced model that has all parameters except the parameter for variable x_j. The difference between the model sum of squares for the full model and the model sum of squares for the reduced model is the Type II sum of squares for the test of the null hypothesis that the coefficient for variable x_j is 0. The numerator of the F test has one degree of freedom. The mean square error for the full model is the denominator of the F test of variable x_j. Note that the estimates of the coefficients for the two models are not usually the same. When variable x_j is removed, the coefficients for the other variables change to compensate for the removal of x_j. In a transformation regression model, the transformations of the other variables must be permitted to change and the numerator degrees of freedom are not always ones. It is not correct to simply let the model coefficients for the transformed variables change and apply the new model coefficients to the old transformations computed with the old scoring parameter estimates. In a transformation regression model, further iteration is needed to test each transformation, because all the scoring parameter estimates for other variables must be permitted to change to test the effect of variable x_j. This can be quite time-consuming for a large model if the SOLVE *a-option* cannot be used to solve directly for the transformations.

Output Data Set

The OUT= output data set can contain a great deal of information; however, in most cases, the output data set contains a small portion of the entire range of available information.

Output Data Set Examples

This section provides three brief examples, illustrating some typical OUT= output data sets. See the section "Output Data Set Contents" on page 10704 for a complete list of the contents of the OUT= data set.

The first example shows the output data set from a two-way ANOVA model. The following statements produce Figure 126.65:

```
title 'ANOVA Output Data Set Example';

data ReferenceCell;
   input y x1 $ x2 $;
   datalines;
11  a  a
12  a  a
10  a  a
 4  a  b
 5  a  b
 3  a  b
 5  b  a
 6  b  a
 4  b  a
 2  b  b
 3  b  b
```

```
 1  b  b
;

* Fit Reference Cell Two-Way ANOVA Model;
proc transreg data=ReferenceCell;
   model identity(y) = class(x1 | x2);
   output coefficients replace predicted residuals;
run;

* Print the Results;
proc print;
run;

proc contents position;
   ods select position;
run;
```

Figure 126.65 ANOVA Example Output Data Set Contents

ANOVA Output Data Set Example

Obs	_TYPE_	_NAME_	y	Py	Ry	Intercept	x1a	x2a	x1ax2a	x1	x2
1	SCORE	ROW1	11	11	0	1	1.0	1	1	a	a
2	SCORE	ROW2	12	11	1	1	1.0	1	1	a	a
3	SCORE	ROW3	10	11	-1	1	1.0	1	1	a	a
4	SCORE	ROW4	4	4	0	1	1.0	0	0	a	b
5	SCORE	ROW5	5	4	1	1	1.0	0	0	a	b
6	SCORE	ROW6	3	4	-1	1	1.0	0	0	a	b
7	SCORE	ROW7	5	5	0	1	0.0	1	0	b	a
8	SCORE	ROW8	6	5	1	1	0.0	1	0	b	a
9	SCORE	ROW9	4	5	-1	1	0.0	1	0	b	a
10	SCORE	ROW10	2	2	0	1	0.0	0	0	b	b
11	SCORE	ROW11	3	2	1	1	0.0	0	0	b	b
12	SCORE	ROW12	1	2	-1	1	0.0	0	0	b	b
13	M COEFFI	y	.	.	.	2	2.0	3	4		
14	MEAN	y	.	.	.	.	7.5	8	11		

Figure 126.65 *continued*

ANOVA Output Data Set Example

The CONTENTS Procedure

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	_TYPE_	Char	8	
2	_NAME_	Char	32	
3	y	Num	8	
4	Py	Num	8	y Predicted Values
5	Ry	Num	8	y Residuals
6	Intercept	Num	8	Intercept
7	x1a	Num	8	x1 a
8	x2a	Num	8	x2 a
9	x1ax2a	Num	8	x1 a * x2 a
10	x1	Char	32	
11	x2	Char	32	

The _TYPE_ variable indicates observation type: score, multiple regression coefficient (parameter estimates), and marginal means. The _NAME_ variable contains the default observation labels, “ROW1”, “ROW2”, and so on, and contains the dependent variable name (y) for the remaining observations. If you specify an ID statement, _NAME_ contains the values of the first ID variable for score observations. The y variable is the dependent variable, Py contains the predicted values, Ry contains the residuals, and the variables Intercept through x1ax2a contain the design matrix. The x1 and x2 variables are the original CLASS variables.

The next example shows the contents of the output data set from fitting a curve through a scatter plot. The following statements produce Figure 126.66:

```
title 'Output Data Set for Curve Fitting Example';

data a;
   do x = 1 to 100;
      y = log(x) + sin(x / 10) + normal(7);
      output;
   end;
run;

proc transreg;
   model identity(y) = spline(x / nknots=9);
   output predicted out=b;
run;

proc contents position;
   ods select position;
run;
```

Figure 126.66 Predicted Values Example Output Data Set Contents

Output Data Set for Curve Fitting Example

The CONTENTS Procedure

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	_TYPE_	Char	8	
2	_NAME_	Char	32	
3	y	Num	8	
4	Ty	Num	8	y Transformation
5	Py	Num	8	y Predicted Values
6	Intercept	Num	8	Intercept
7	x	Num	8	
8	TIntercept	Num	8	Intercept Transformation
9	Tx	Num	8	x Transformation

The OUT= data set contains _TYPE_ and _NAME_ variables. Since no coefficients or coordinates are requested, all observations are _TYPE_='SCORE'. The y variable is the original dependent variable, Ty is the transformed dependent variable, Py contains the predicted values, x is the original independent variable, and Tx is the transformed independent variable. The data set also contains an Intercept and transformed intercept TIntercept variable. (In this case, the transformed intercept is the same as the intercept. However, if you specify the TSTANDARD= and ADDITIVE options, these are not always the same.)

The following example shows the results from specifying METHOD=MORALS when there is more than one dependent variable:

```
title 'METHOD=MORALS Output Data Set Example';

data x;
   input y1 y2 x1 $ x2 $;
   datalines;
11 1 a a
10 4 b a
 5 2 a b
 5 9 b b
 4 3 c c
 3 6 b a
 1 8 a b
;

* Fit Reference Cell Two-Way ANOVA Model;
proc transreg data=x noprint solve;
   model spline(y1 y2) = opscore(x1 x2 / name=(n1 n2));
   output coefficients predicted residuals;
   id x1 x2;
run;

* Print the Results;
proc print;
run;
```

```
proc contents position;
   ods select position;
run;
```

These statements produce Figure 126.67.

Figure 126.67 METHOD=MORALS Rolled Output Data Set

METHOD=MORALS Output Data Set Example

Obs	_DEPVAR_	_TYPE_	_NAME_	_DEPEND_	T_DEPEND_	P_DEPEND_
1	Spline(y1)	SCORE	a	11	13.1600	11.1554
2	Spline(y1)	SCORE	b	10	6.1931	6.8835
3	Spline(y1)	SCORE	a	5	2.4467	4.7140
4	Spline(y1)	SCORE	b	5	2.4467	0.4421
5	Spline(y1)	SCORE	c	4	4.2076	4.2076
6	Spline(y1)	SCORE	b	3	5.5693	6.8835
7	Spline(y1)	SCORE	a	1	4.9766	4.7140
8	Spline(y1)	M COEFFI	y1	.	.	.
9	Spline(y2)	SCORE	a	1	-0.5303	-0.5199
10	Spline(y2)	SCORE	b	4	5.5487	4.5689
11	Spline(y2)	SCORE	a	2	3.8940	4.5575
12	Spline(y2)	SCORE	b	9	9.6358	9.6462
13	Spline(y2)	SCORE	c	3	5.6210	5.6210
14	Spline(y2)	SCORE	b	6	3.5994	4.5689
15	Spline(y2)	SCORE	a	8	5.2314	4.5575
16	Spline(y2)	M COEFFI	y2	.	.	.

Obs	R_DEPEND_	Intercept	n1	n2	TIntercept	Tn1	Tn2	x1	x2
1	2.00464	1	0	0	1.0000	0.06711	-0.09384	a	a
2	-0.69041	1	1	0	1.0000	1.51978	-0.09384	b	a
3	-2.26724	1	0	1	1.0000	0.06711	1.32038	a	b
4	2.00464	1	1	1	1.0000	1.51978	1.32038	b	b
5	0.00000	1	2	2	1.0000	0.23932	1.32038	c	c
6	-1.31422	1	1	0	1.0000	1.51978	-0.09384	b	a
7	0.26261	1	0	1	1.0000	0.06711	1.32038	a	b
8	.	.	.	.	10.9253	-2.94071	-4.55475	y1	y1
9	-0.01043	1	0	0	1.0000	0.03739	-0.09384	a	a
10	0.97988	1	1	0	1.0000	1.51395	-0.09384	b	a
11	-0.66347	1	0	1	1.0000	0.03739	1.32038	a	b
12	-0.01043	1	1	1	1.0000	1.51395	1.32038	b	b
13	0.00000	1	2	2	1.0000	0.34598	1.32038	c	c
14	-0.96945	1	1	0	1.0000	1.51395	-0.09384	b	a
15	0.67390	1	0	1	1.0000	0.03739	1.32038	a	b
16	.	.	.	.	-0.3119	3.44636	3.59024	y2	y2

Figure 126.67 *continued*

METHOD=MORALS Output Data Set Example

The CONTENTS Procedure

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	_DEPVAR_	Char	42	Dependent Variable Transformation(Name)
2	_TYPE_	Char	8	
3	_NAME_	Char	32	
4	_DEPEND_	Num	8	Dependent Variable
5	T_DEPEND_	Num	8	Dependent Variable Transformation
6	P_DEPEND_	Num	8	Dependent Variable Predicted Values
7	R_DEPEND_	Num	8	Dependent Variable Residuals
8	Intercept	Num	8	Intercept
9	n1	Num	8	
10	n2	Num	8	
11	TIntercept	Num	8	Intercept Transformation
12	Tn1	Num	8	n1 Transformation
13	Tn2	Num	8	n2 Transformation
14	x1	Char	32	
15	x2	Char	32	

If you specify METHOD=MORALS with multiple dependent variables, PROC TRANSREG performs separate univariate analyses and stacks the results in the OUT= data set. For this example, the results of the first analysis are in the partition designated by _DEPVAR_='Spline(y1)' and the results of the second analysis are in the partition designated by _DEPVAR_='Spline(y2)', which are the transformation and dependent variable names. Each partition has _TYPE_='SCORE' observations for the variables and a _TYPE_='M COEFFI' observation for the coefficients. In this example, an ID variable is specified, so the _NAME_ variable contains the formatted values of the first ID variable. Since both dependent variables have to go into the same column, the dependent variable is given a new name, _DEPEND_. The dependent variable transformation is named T_DEPEND_, the predicted values variable is named P_DEPEND_, and the residuals variable is named R_DEPEND_.

The independent variables are character OPSCORE variables. By default, PROC TRANSREG replaces character OPSCORE variables with category numbers and discards the original character variables. To avoid this, the input variables are renamed from x1 and x2 to n1 and n2 and the original x1 and x2 are added to the data set as ID variables. The n1 and n2 variables contain the initial values for the OPSCORE transformations, and the Tn1 and Tn2 variables contain optimal scores. The data set also contains an Intercept and transformed intercept TIntercept variable. The regression coefficients are in the transformation columns, which also contain the variables to which they apply.

Output Data Set Contents

Table 126.7 summarizes the various matrices that can result from PROC TRANSREG processing and that appear in the OUT= data set. The exact contents of an OUT= data set depends on many options.

Table 126.7 PROC TRANSREG OUT= Data Set Contents

TYPE	Contents	Options, Default Prefix
SCORE	dependent variables	DREPLACE not specified
SCORE	independent variables	IREPLACE not specified
SCORE	transformed dependent variables	default, TDPREFIX=T
SCORE	transformed independent variables	default, TIPREFIX=T
SCORE	predicted values	PREDICTED, PPREFIX=P
SCORE	residuals	RESIDUALS, RDPREFIX=R
SCORE	leverage	LEVERAGE, LEVERAGE=Leverage
SCORE	lower individual confidence limits	CLI, LILPREFIX=LIL, CILPREFIX=CIL
SCORE	upper individual confidence limits	CLI, LIUPREFIX=LIU, CIUPREFIX=CIU
SCORE	lower mean confidence limits	CLM, LMLPREFIX=LML, CMLPREFIX=CML
SCORE	upper mean confidence limits	CLM, LMUPREFIX=LMU, CMUPREFIX=CMU
SCORE	dependent canonical variables	CANONICAL, CDPREFIX=Cand
SCORE	independent canonical variables	CANONICAL, CIPREFIX=Cani
SCORE	redundancy variables	REDUNDANCY, RPREFIX=Red
SCORE	ID, CLASS, BSPLINE variables	ID, CLASS, BSPLINE,
SCORE	independent variables approximations	IAPPROXIMATIONS, AIPREFIX=A
M COEFFI	multiple regression coefficients	COEFFICIENTS, MRC
C COEFFI	canonical coefficients	COEFFICIENTS, CCC
MEAN	marginal means	COEFFICIENTS, MEANS
M REDUND	multiple redundancy coefficients	MREDUNDANCY
R REDUND	multiple redundancy coefficients	MREDUNDANCY
M POINT	point coordinates	COORDINATES or MPC, POINT
M EPOINT	elliptical point coordinates	COORDINATES or MEC, EPOINT
M QPOINT	quadratic point coordinates	COORDINATES or MQC, QPOINT
C POINT	canonical point coordinates	COORDINATES or CPC, POINT
C EPOINT	canonical elliptical point coordinates	COORDINATES or CEC, EPOINT
C QPOINT	canonical quadratic point coordinates	COORDINATES or CQC, QPOINT

The independent and dependent variables are created from the original input data. Several potential differences exist between these variables and the actual input data. An intercept variable can be added, new variables can be added for POINT, EPOINT, QPOINT, CLASS, IDENTITY, PSPLINE, and BSPLINE variables, and category numbers are substituted for character OPSCORE variables. These matrices are not always what is input to the first iteration. After the expanded data set is stored for inclusion in the output data set, several things happen to the data before they are input to the first iteration: column means are substituted for missing

values; zero-degree SPLINE and MSPLINE variables are transformed so that the iterative algorithms get step-function data as input, which conform to the zero-degree transformation family restrictions; and the nonoptimal transformations are performed.

Details for the UNIVARIATE Method

When you specify METHOD=UNIVARIATE (in the MODEL or PROC TRANSREG statement), PROC TRANSREG can perform several analyses, one for each dependent variable. While each dependent variable can be transformed, their independent variables are not transformed. The OUT= data set optionally contains all of the _TYPE_='SCORE' observations, optionally followed by coefficients or coordinates.

Details for the MORALS Method

When you specify METHOD=MORALS (in the MODEL or PROC TRANSREG statement), successive analyses are performed, one for each dependent variable. Each analysis transforms one dependent variable and the entire set of the independent variables. All information for the first dependent variable (scores then, optionally, coefficients) appears first. Then all information for the second dependent variable (scores then, optionally, coefficients) appears next. This arrangement is repeated for all dependent variables.

Details for the CANALS and REDUNDANCY Methods

For METHOD=CANALS and METHOD=REDUNDANCY (specified in either the MODEL or PROC TRANSREG statement), one analysis is performed that simultaneously transforms all dependent and independent variables. The OUT= data set optionally contains all of the _TYPE_='SCORE' observations, optionally followed by coefficients or coordinates.

Variable Names

As shown in the preceding examples, some variables in the output data set directly correspond to input variables, and some are created. All original optimal and nonoptimal transformation variable names are unchanged.

The names of the POINT, QPOINT, and EPOINT expansion variables are also left unchanged, but new variables are created. When independent POINT variables are present, the sum-of-squares variable _ISSQ_ is added to the output data set. For each EPOINT and QPOINT variable, a new squared variable is created by appending "_2". For example, Dim1 and Dim2 are expanded into Dim1, Dim2, Dim1_2, and Dim2_2. In addition, for each pair of QPOINT variables, a new crossproduct variable is created by combining the two names—for example, Dim1Dim2.

The names of the CLASS variables are constructed from original variable names and levels. Lengths are controlled by the CPREFIX= *a-option*. For example, when x1 and x2 both have values of 'a' and 'b', CLASS(x1 | x2 / ZERO=NONE) creates x1 main-effect variable names x1a x1b, x2 main-effect variable names x2a x2b, and interaction variable names x1ax2a x1ax2b x1bx2a x1bx2b.

PROC TRANSREG then uses these variable names when creating the transformed, predicted, and residual variable names by affixing the relevant prefix and dropping extra characters if necessary.

METHOD=MORALS Variable Names

When you specify METHOD=MORALS and only one dependent variable is present, the output data set is structured exactly as if METHOD=REDUNDANCY (see the section "Details for the CANALS and REDUNDANCY Methods" on page 10705). When more than one dependent variable is present, the dependent variables are output in the variable _DEPEND_, transformed dependent variables are output in the

variable T_DEPEND_, predicted values are output in the variable P_DEPEND_, and residuals are output in the variable R_DEPEND_. You can partition the data set into BY groups, one per dependent variable, by referring to the character variable _DEPVAR_, which contains the original dependent variable names and transformations.

Duplicate Variable Names

When the same name is generated from multiple variables in the OUT= data set, new names are created by appending '2', '3', or '4', and so on, until a unique name is created. Let v be the maximum variable length (for example, 32 for VALIDVARNAME=V7). For v-character names, the last character is replaced with a numeric suffix until a unique name is created. For example, if there are two output variables that otherwise would be named x, then x and x2 are created instead. If $v = 32$ and there are two output variables that otherwise would be named ThisIsAThirtyTwoCharacterVarName, then ThisIsAThirtyTwoCharacterVarName and ThisIsAThirtyTwoCharacterVarNam2 are created instead.

OUTTEST= Output Data Set

The OUTTEST= data set contains hypothesis test results. The OUTTEST= data set always contains ANOVA results. When you specify the SS2 *a-option*, regression tables are also output. When you specify the UTILITIES *a-option*, conjoint analysis part-worth utilities are also output. The OUTTEST= data set has the following variables:

DEPVAR	is a 42-character variable that contains the dependent variable transformation and name.
TYPE	is an 8-character variable that contains the table type. The first character is "U" for univariate or "M" for multivariate. The second character is blank. The third character is "A" for ANOVA, "2" for Type II sum of squares, or "U" for UTILITIES. The fourth character is blank. The fifth character is "L" for liberal tests, "C" for conservative tests, or "U" for the usual tests.
Title	is an 80-character variable that contains the table title.
Variable	is a 42-character variable that contains the independent variable transformations and names for regression tables and blanks for ANOVA tables.
Coefficient	contains the multiple regression coefficients for regression tables and underscore special missing values for ANOVA tables.
Statistic	is a 24-character variable that contains the names for statistics in other variables, such as Value.
Value	contains multivariate test statistics and all other information that does not fit in one of the other columns including R square, dependent mean, adjusted R square, and coefficient of variation. Whenever Value is not an underscore special missing value, the Statistic variable describes the contents of the Value variable.
NumDF	contains numerator degrees of freedom for F tests.
DenDF	contains denominator degrees of freedom for F tests.
SSq	contains sums of squares.
MeanSquare	contains mean squares.
F	contains F statistics.

NumericP	contains the p-value for the F statistic, stored in a numeric variable.
P	is a 9-character variable that contains the formatted p-value for the F statistic, including the appropriate ~, <=, >=, or blank symbols.
LowerLimit	contains lower confidence limits on the parameter estimates.
UpperLimit	contains upper confidence limits on the parameter estimates.
StdError	contains standard errors. For SS2 and UTILITIES tables, standard errors are output for each coefficient with one degree of freedom.
Importance	contains the relative importance of each factor for UTILITIES tables.
Label	is a 256-character variable that contains variable labels.

There are several possible tables in the OUTTEST= data set corresponding to combinations of univariate and multivariate tests; ANOVA and regression results; and liberal, conservative, and the usual tests. Each table is composed of only a subset of the variables. Numeric variables contain underscore special missing values when they are not a column in a table. Ordinary missing values (.) appear in variables that are part of a table when a nonmissing value cannot be produced. For example, the F is missing for a test with zero degrees of freedom.

Computational Resources

This section provides information about the computational resources required to use PROC TRANSREG.

Let

$$
\begin{aligned}
n &= \text{number of observations} \\
q &= \text{number of expanded independent variables} \\
r &= \text{number of expanded dependent variables} \\
k &= \text{maximum spline degree} \\
p &= \text{maximum number of knots}
\end{aligned}
$$

More than $56(q + r)$ plus the maximum of the data matrix size, the optimal scaling work space, and the covariance matrix size bytes of array space are required. The data matrix size is $8n(q + r)$ bytes. The optimal scaling work space requires less than $8(6n + (p + k + 2)(p + k + 11))$ bytes. The covariance matrix size is $4(q + r)(q + r + 1)$ bytes.

PROC TRANSREG tries to store the original and transformed data in memory. If there is not enough memory, a utility data set is used, potentially resulting in a large increase in execution time. The amount of memory for the preceding data formulas is an underestimate of the amount of memory needed to handle most problems. These formulas give the absolute minimum amount of memory required. If a utility data set is used, and if memory can be used with perfect efficiency, then roughly the amount of memory stated previously is needed. In reality, most problems require at least two or three times the minimum.

PROC TRANSREG sorts the data once. The sort time is roughly proportional to $(q + r)n^{3/2}$.

One regression analysis per iteration is required to compute model parameters (or two canonical correlation analyses per iteration for METHOD=CANALS). The time required to accumulate the crossproducts matrix is roughly proportional to $n(q+r)^2$. The time required to compute the regression coefficients is roughly proportional to q^3.

Each optimal scaling is a multiple regression problem, although some transformations are handled with faster special-case algorithms. The number of regressors for the optimal scaling problems depends on the original values of the variable and the type of transformation. For each monotone spline transformation, an unknown number of multiple regressions is required to find a set of coefficients that satisfies the constraints. The B-spline basis is generated twice for each SPLINE and MSPLINE transformation for each iteration. The time required to generate the B-spline basis is roughly proportional to nk^2.

Unbalanced ANOVA without CLASS Variables

This section illustrates that an analysis of variance model can be formulated as a simple regression model with optimal scoring. The purpose of the example is to explain one aspect of how PROC TRANSREG works, not to propose an alternative way of performing an analysis of variance.

Finding the overall fit of a large, unbalanced analysis of variance model can be handled as an optimal scoring problem without creating large, sparse design matrices. For example, consider an unbalanced full main-effects and interactions ANOVA model with six factors. Assume that a SAS data set is created with factor-level indicator variables c1 through c6 and dependent variable y. If each factor level consists of nonblank single characters, you can create a cell indicator in a DATA step with the statement as follows:

```
x=compress(c1||c2||c3||c4||c5||c6);
```

The following statements optimally score x (by using the OPSCORE transformation) and do not transform y:

```
proc transreg;
   model identity(y)=opscore(x);
   output;
run;
```

The final R square reported is the R square for the full analysis of variance model. This R square is the same R square that would be reported by both of the following PROC GLM steps:

```
proc glm;
   class x;
   model y=x;
run;

proc glm;
   class c1-c6;
   model y=c1|c2|c3|c4|c5|c6;
run;
```

PROC TRANSREG optimally scores the classes of x, within the space of a single variable with values linearly related to the cell means, so the full ANOVA problem is reduced to a simple regression problem with an optimal independent variable. PROC TRANSREG requires only one iteration to find the optimal scoring of x but, by default, performs a second iteration, which reports no data changes.

Hypothesis Tests for Simple Univariate Models

If the dependent variable has one parameter (IDENTITY, LINEAR with no missing values, and so on) and if there are no monotonicity constraints, PROC TRANSREG fits univariate models, which can also be fit with a DATA step and PROC REG. This is illustrated with the following artificial data set:

```
data htex;
   do i = 0.5 to 10 by 0.5;
      x1 = log(i);
      x2 = sqrt(i) + sin(i);
      x3 = 0.05 * i * i + cos(i);
      y  = x1 - x2 + x3 + 3 * normal(7);
      x1 = x1 + normal(7);
      x2 = x2 + normal(7);
      x3 = x3 + normal(7);
      output;
   end;
run;
```

Both PROC TRANSREG and PROC REG are run to fit the same polynomial regression model as follows:

```
proc transreg data=htex ss2 short;
   title 'Fit a Polynomial Regression Model with PROC TRANSREG';
   model identity(y) = spline(x1);
run;

data htex2;
   set htex;
   x1_1 = x1;
   x1_2 = x1 * x1;
   x1_3 = x1 * x1 * x1;
run;

proc reg;
   title 'Fit a Polynomial Regression Model with PROC REG';
   model y = x1_1 - x1_3;
quit;
```

The ANOVA and regression tables from PROC TRANSREG are displayed in Figure 126.68. The ANOVA and regression tables from PROC REG are displayed in Figure 126.69. The SHORT *a-option* is specified with PROC TRANSREG to suppress the iteration history.

Figure 126.68 ANOVA and Regression Output from PROC TRANSREG

Fit a Polynomial Regression Model with PROC TRANSREG

The TRANSREG Procedure

Dependent Variable Identity(y)

Number of Observations Read	20
Number of Observations Used	20

Figure 126.68 *continued*

Identity(y)
Algorithm converged.

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	5.8365	1.94550	0.14	0.9329
Error	16	218.3073	13.64421		
Corrected Total	19	224.1438			

Root MSE	3.69381	R-Square	0.0260
Dependent Mean	0.85490	Adj R-Sq	-0.1566
Coeff Var	432.07258		

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F
Intercept	1	1.4612767	18.8971	18.8971	1.38	0.2565
Spline(x1)	3	-0.3924013	5.8365	1.9455	0.14	0.9329

Figure 126.69 ANOVA and Regression Output from PROC REG

Fit a Polynomial Regression Model with PROC REG

The REG Procedure
Model: MODEL1
Dependent Variable: y

Number of Observations Read	20
Number of Observations Used	20

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	5.83651	1.94550	0.14	0.9329
Error	16	218.30729	13.64421		
Corrected Total	19	224.14380			

Root MSE	3.69381	R-Square	0.0260
Dependent Mean	0.85490	Adj R-Sq	-0.1566
Coeff Var	432.07258		

Figure 126.69 *continued*

Parameter Estimates					
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	1.22083	1.47163	0.83	0.4190
x1_1	1	0.79743	1.75129	0.46	0.6550
x1_2	1	-0.49381	1.50449	-0.33	0.7470
x1_3	1	0.04422	0.32956	0.13	0.8949

The PROC TRANSREG regression table differs in several important ways from the parameter estimate table produced by PROC REG. The REG procedure displays standard errors and t statistics. PROC TRANSREG displays Type II sums of squares, mean squares, and F statistics. The difference is because the numerator degrees of freedom are not always 1, so t tests are not uniformly appropriate. When the degrees of freedom for variable x_j is 1, the following relationships hold between the standard errors (s_{β_j}) and the Type II sums of squares (SS_j):

$$s_{\beta_j} = (\hat{\beta}_j^2 / F_j)^{1/2}$$

and

$$\mathrm{SS}_j = \hat{\beta}_j^2 \times \mathrm{MSE}/s_{\beta_j}^2$$

PROC TRANSREG does not provide tests of the individual terms that go into the transformation. (However, it could if BSPLINE or PSPLINE had been specified instead of SPLINE.) The test of **spline(x1)** is the same as the test of the overall model. The intercepts are different due to the different numbers of variables and their standardizations.

In the next example, both x1 and x2 are transformed in the first PROC TRANSREG step, and PROC TRANSREG is used instead of a DATA step to create the polynomials for PROC REG. Both PROC TRANSREG and PROC REG fit the same polynomial regression model. The following statements run PROC TRANSREG and PROC REG and produce Figure 126.70 and Figure 126.71:

```
title 'Two-Variable Polynomial Regression';

proc transreg data=htex ss2 solve;
   model identity(y) = spline(x1 x2);
run;

proc transreg noprint data=htex maxiter=0;
   /* Use PROC TRANSREG to prepare input to PROC REG */
   model identity(y) = pspline(x1 x2);
   output out=htex2;
run;

proc reg data=htex2;
   model y = x1_1-x1_3 x2_1-x2_3;
   test x1_1, x1_2, x1_3;
   test x2_1, x2_2, x2_3;
quit;
```

Figure 126.70 Two-Variable Polynomial Regression Output from PROC TRANSREG

Two-Variable Polynomial Regression

The TRANSREG Procedure

Dependent Variable Identity(y)

Number of Observations Read	20
Number of Observations Used	20

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	0.69502	4.73421	0.08252		
1	0.00000	0.00000	0.17287	0.09035	Converged

Algorithm converged.

Hypothesis Test Iterations Excluding Spline(x1)

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	0.03575	0.32390	0.15097		
1	0.00000	0.00000	0.15249	0.00152	Converged

Algorithm converged.

Hypothesis Test Iterations Excluding Spline(x2)

TRANSREG MORALS Algorithm Iteration History for Identity(y)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	0.45381	1.43736	0.00717		
1	0.00000	0.00000	0.02604	0.01886	Converged

Algorithm converged.

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	38.7478	6.45796	0.45	0.8306
Error	13	185.3960	14.26123		
Corrected Total	19	224.1438			

Root MSE	3.77640	**R-Square**	0.1729
Dependent Mean	0.85490	**Adj R-Sq**	-0.2089
Coeff Var	441.73431		

Figure 126.70 *continued*

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F
Intercept	1	3.5437125	35.2282	35.2282	2.47	0.1400
Spline(x1)	3	0.3644562	4.5682	1.5227	0.11	0.9546
Spline(x2)	3	-1.3551738	32.9112	10.9704	0.77	0.5315

There are three iteration histories: one for the overall model and two for the two independent variables. The first PROC TRANSREG iteration history shows the R square of 0.17287 for the fit of the overall model. The second is for the following model:

```
model identity(y) = spline(x2);
```

This model excludes **spline(x1)**. The third iteration history is for the following model:

```
model identity(y) = spline(x1);
```

This model excludes **spline(x2)**. The difference between the first and second R square times the total sum of squares is the model sum of squares for **spline(x1)**:

$$(0.17287 - 0.15249) \times 224.143800 = 4.568165$$

The difference between the first and third R square times the total sum of squares is the model sum of squares for **spline(x2)**:

$$(0.17287 - 0.02604) \times 224.143800 = 32.911247$$

Figure 126.71 displays the PROC REG results. The TEST statement in PROC REG tests the null hypothesis that the vector of parameters for x1_1 x1_2 x1_3 is zero. This is the same test as the **spline(x1)** test used by PROC TRANSREG. Similarly, the PROC REG test that the vector of parameters for x2_1 x2_2 x2_3 is zero is the same as the PROC TRANSREG SPLINE(x2) test. So for models with no monotonicity constraints and no dependent variable transformations, PROC TRANSREG provides little more than a different packaging of standard least squares methodology.

Figure 126.71 Two-Variable Polynomial Regression Output from PROC REG

Two-Variable Polynomial Regression

The REG Procedure
Model: MODEL1
Dependent Variable: y

Number of Observations Read	20
Number of Observations Used	20

Figure 126.71 *continued*

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	6	38.74775	6.45796	0.45	0.8306
Error	13	185.39605	14.26123		
Corrected Total	19	224.14380			

Root MSE	3.77640	**R-Square**	0.1729
Dependent Mean	0.85490	**Adj R-Sq**	-0.2089
Coeff Var	441.73431		

Parameter Estimates						
Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	Intercept	1	10.77824	7.55244	1.43	0.1771
x1_1	x1 1	1	0.40112	1.81024	0.22	0.8281
x1_2	x1 2	1	0.25652	1.66023	0.15	0.8796
x1_3	x1 3	1	-0.11639	0.36775	-0.32	0.7567
x2_1	x2 1	1	-14.07054	12.50521	-1.13	0.2809
x2_2	x2 2	1	5.95610	5.97952	1.00	0.3374
x2_3	x2 3	1	-0.80608	0.87291	-0.92	0.3726

Two-Variable Polynomial Regression

The REG Procedure
Model: MODEL1

Test 1 Results for Dependent Variable y				
Source	DF	Mean Square	F Value	Pr > F
Numerator	3	1.52272	0.11	0.9546
Denominator	13	14.26123		

Two-Variable Polynomial Regression

The REG Procedure
Model: MODEL1

Test 2 Results for Dependent Variable y				
Source	DF	Mean Square	F Value	Pr > F
Numerator	3	10.97042	0.77	0.5315
Denominator	13	14.26123		

Hypothesis Tests with Monotonicity Constraints

Now consider a model with monotonicity constraints. This model has no counterpart in PROC REG. The following statements fit a monotone-spline model and produce Figure 126.72:

```
title 'Monotone Splines';

proc transreg data=htex ss2 short;
   model identity(y) = mspline(x1-x3 / nknots=3);
run;
```

The SHORT *a-option* is specified to suppress the iteration histories. Two ANOVA tables are displayed—one by using liberal degrees of freedom and one by using conservative degrees of freedom. All sums of squares and the R squares are the same for both tables. What differs are the degrees of freedom and statistics that use degrees of freedom. The liberal test has 8 model degrees of freedom and 11 error degrees of freedom, whereas the conservative test has 15 model degrees of freedom and only 4 error degrees of freedom. The "true" p-value is between 0.8462 and 0.9997, so clearly you would fail to reject the null hypothesis. Unfortunately, results are not always this clear. (See Figure 126.72.)

Figure 126.72 Monotone Spline Transformations

Monotone Splines

The TRANSREG Procedure

Dependent Variable Identity(y)

Number of Observations Read	20
Number of Observations Used	20

Identity(y)
Algorithm converged.

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on Liberal Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Liberal p
Model	8	58.0534	7.25667	0.48	>= 0.8462
Error	11	166.0904	15.09913		
Corrected Total	19	224.1438			

Root MSE	3.88576	**R-Square**	0.2590
Dependent Mean	0.85490	**Adj R-Sq**	-0.2799
Coeff Var	454.52581		

Figure 126.72 *continued*

Univariate ANOVA Table Based on Conservative Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Conservative p
Model	15	58.0534	3.87022	0.09	<= 0.9997
Error	4	166.0904	41.52261		
Corrected Total	19	224.1438			

Root MSE	6.44380	**R-Square**	0.2590
Dependent Mean	0.85490	**Adj R-Sq**	-2.5197
Coeff Var	753.74578		

Univariate Regression Table Based on Liberal Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Liberal p
Intercept	1	4.8687676	54.7372	54.7372	3.63	>= 0.0834
Mspline(x1)	2	-0.6886834	12.1943	6.0972	0.40	>= 0.6773
Mspline(x2)	3	-1.8237319	46.3155	15.4385	1.02	>= 0.4199
Mspline(x3)	3	0.8646155	24.6840	8.2280	0.54	>= 0.6616

Univariate Regression Table Based on Conservative Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Conservative p
Intercept	1	4.8687676	54.7372	54.7372	1.32	<= 0.3149
Mspline(x1)	5	-0.6886834	12.1943	2.4389	0.06	<= 0.9959
Mspline(x2)	5	-1.8237319	46.3155	9.2631	0.22	<= 0.9344
Mspline(x3)	5	0.8646155	24.6840	4.9368	0.12	<= 0.9809

Hypothesis Tests with Dependent Variable Transformations

PROC TRANSREG can also provide approximate tests of hypotheses when the dependent variable is transformed, but the output is more complicated. When a dependent variable has more than one degree of freedom, the problem becomes multivariate. Hypothesis tests are performed in the context of a multivariate linear model with the number of dependent variables equal to the number of scoring parameters for the dependent variable transformation. The transformation regression model with a dependent variable transformation differs from the usual multivariate linear model in two important ways. First, the usual assumption of multivariate normality is always violated. This fact is simply ignored. This is one reason why all hypothesis tests in the presence of a dependent variable transformation should be considered approximate at best. Multivariate normality is assumed even though it is known that the assumption is violated.

The second difference concerns the usual multivariate test statistics: Pillai's trace, Wilks' lambda, Hotelling-Lawley trace, and Roy's greatest root. The first three statistics are defined in terms of all the squared canonical correlations. Here, there is only one linear combination (the transformation), and hence only one squared canonical correlation of interest, which is equal to the R square. It might seem that Roy's greatest root, which

uses only the largest squared canonical correlation, is the only statistic of interest. Unfortunately, Roy's greatest root is very liberal and provides only a lower bound on the p-value. Approximate upper bounds are provided by adjusting the other three statistics for the one linear combination case. Wilks' lambda, Pillai's trace, and Hotelling-Lawley trace are a conservative adjustment of the usual statistics.

These statistics are normally defined in terms of the squared canonical correlations, which are the eigenvalues of the matrix $\mathbf{H}(\mathbf{H}+\mathbf{E})^{-1}$, where $\mathbf{H}$ is the hypothesis sum-of-squares matrix and $\mathbf{E}$ is the error sum-of-squares matrix. Here the R square is used for the first eigenvalue, and all other eigenvalues are set to 0 since only one linear combination is used. Degrees of freedom are computed assuming that all linear combinations contribute to the lambda and trace statistics, so the F tests for those statistics are conservative. The p-values for the liberal and conservative statistics provide approximate lower and upper bounds on p. In practice, the adjusted Pillai's trace is very conservative—perhaps too conservative to be useful. Wilks' lambda is less conservative, and the Hotelling-Lawley trace seems to be the least conservative. The conservative statistics and the liberal Roy's greatest root provide a bound on the true p-value. Unfortunately, they sometimes report a bound of 0.0001 and 1.0000.

The following example has a dependent variable transformation and produces Figure 126.73:

```
title 'Transform Dependent and Independent Variables';

proc transreg data=htex ss2 solve short;
   model spline(y) = spline(x1-x3);
run;
```

The univariate results match Roy's greatest root results. Clearly, the proper action is to fail to reject the null hypothesis. However, as stated previously, results are not always this clear.

Figure 126.73 Transform Dependent and Independent Variables

Transform Dependent and Independent Variables

The TRANSREG Procedure

Dependent Variable Spline(y)

Number of Observations Read	20
Number of Observations Used	20

Spline(y)
Algorithm converged.

The TRANSREG Procedure Hypothesis Tests for Spline(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Liberal p
Model	9	110.8822	12.32025	1.09	>= 0.4452
Error	10	113.2616	11.32616		
Corrected Total	19	224.1438			

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Figure 126.73 *continued*

Root MSE	3.36544	**R-Square**	0.4947
Dependent Mean	0.85490	**Adj R-Sq**	0.0399
Coeff Var	393.66234		

Adjusted Multivariate ANOVA Table Based on the Usual Degrees of Freedom

Dependent Variable Scoring Parameters=3 S=3 M=2.5 N=3

Statistic	Value	F Value	Num DF	Den DF	p
Wilks' Lambda	0.505308	0.23	27	24.006	<= 0.9998
Pillai's Trace	0.494692	0.22	27	30	<= 0.9999
Hotelling-Lawley Trace	0.978992	0.26	27	11.589	<= 0.9980
Roy's Greatest Root	0.978992	1.09	9	10	>= 0.4452

The Wilks' Lambda, Pillai's Trace, and Hotelling-Lawley Trace statistics are a conservative adjustment of the normal statistics. Roy's Greatest Root is liberal. These statistics are normally defined in terms of the squared canonical correlations which are the eigenvalues of the matrix H*inv(H+E). Here the R-Square is used for the first eigenvalue and all other eigenvalues are set to zero since only one linear combination is used. Degrees of freedom are computed assuming all linear combinations contribute to the Lambda and Trace statistics, so the F tests for those statistics are conservative. The p values for the liberal and conservative statistics provide approximate lower and upper bounds on p. A liberal test statistic with conservative degrees of freedom and a conservative test statistic with liberal degrees of freedom yield at best an approximate p value, which is indicated by a "~" before the p value.

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Liberal p
Intercept	1	6.9089087	117.452	117.452	10.37	>= 0.0092
Spline(x1)	3	-1.0832321	32.493	10.831	0.96	>= 0.4504
Spline(x2)	3	-2.1539191	45.251	15.084	1.33	>= 0.3184
Spline(x3)	3	0.4779207	10.139	3.380	0.30	>= 0.8259

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Figure 126.73 *continued*

Adjusted Multivariate Regression Table Based on the Usual Degrees of Freedom							
Variable	**Coefficient**	**Statistic**	**Value**	**F Value**	**Num DF**	**Den DF**	**p**
Intercept	6.9089087	Wilks' Lambda	0.49092	2.77	3	8	0.1112
		Pillai's Trace	0.50908	2.77	3	8	0.1112
		Hotelling-Lawley Trace	1.036993	2.77	3	8	0.1112
		Roy's Greatest Root	1.036993	2.77	3	8	0.1112
Spline(x1)	-1.0832321	Wilks' Lambda	0.777072	0.24	9	19.621	<= 0.9840
		Pillai's Trace	0.222928	0.27	9	30	<= 0.9787
		Hotelling-Lawley Trace	0.286883	0.24	9	9.8113	<= 0.9784
		Roy's Greatest Root	0.286883	0.96	3	10	>= 0.4504
Spline(x2)	-2.1539191	Wilks' Lambda	0.714529	0.32	9	19.621	<= 0.9572
		Pillai's Trace	0.285471	0.35	9	30	<= 0.9494
		Hotelling-Lawley Trace	0.399524	0.33	9	9.8113	<= 0.9424
		Roy's Greatest Root	0.399524	1.33	3	10	>= 0.3184
Spline(x3)	0.4779207	Wilks' Lambda	0.917838	0.08	9	19.621	<= 0.9998
		Pillai's Trace	0.082162	0.09	9	30	<= 0.9996
		Hotelling-Lawley Trace	0.089517	0.07	9	9.8113	<= 0.9997
		Roy's Greatest Root	0.089517	0.30	3	10	>= 0.8259

These statistics are adjusted in the same way as the multivariate statistics above.

Hypothesis Tests with One-Way ANOVA

One-way ANOVA models are fit with either an explicit or implicit intercept. In implicit intercept models, the ANOVA table of PROC TRANSREG is the correct table for a model with an intercept, and the regression table is the correct table for a model that does not have a separate explicit intercept. The PROC TRANSREG implicit intercept ANOVA table matches the PROC REG table when the NOINT *a-option* is not specified, and the PROC TRANSREG implicit intercept regression table matches the PROC REG table when the NOINT *a-option* is specified. The following statements illustrate this relationship and produce Figure 126.74:

```
data oneway;
   input y x $;
   datalines;
0 a
1 a
2 a
7 b
8 b
9 b
3 c
4 c
5 c
;

title 'Implicit Intercept Model';

proc transreg ss2 data=oneway short;
   model identity(y) = class(x / zero=none);
```

```
   output out=oneway2;
run;

proc reg data=oneway2;
   model y = xa xb xc;         /* Implicit Intercept ANOVA      */
   model y = xa xb xc / noint; /* Implicit Intercept Regression */
quit;
```

Figure 126.74 Implicit Intercept Model

Implicit Intercept Model

The TRANSREG Procedure

Dependent Variable Identity(y)

Class Level Information		
Class	**Levels**	**Values**
x	3	a b c

Number of Observations Read	9
Number of Observations Used	9
Implicit Intercept Model	

The TRANSREG Procedure Hypothesis Tests for Identity(y)

Univariate ANOVA Table Based on the Usual Degrees of Freedom					
Source	**DF**	**Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**
Model	2	74.00000	37.00000	37.00	0.0004
Error	6	6.00000	1.00000		
Corrected Total	8	80.00000			

Root MSE	1.00000	**R-Square**	0.9250
Dependent Mean	4.33333	**Adj R-Sq**	0.9000
Coeff Var	23.07692		

Univariate Regression Table Based on the Usual Degrees of Freedom							
Variable	**DF**	**Coefficient**	**Type II Sum of Squares**	**Mean Square**	**F Value**	**Pr > F**	**Label**
Class.xa	1	1.00000000	3.000	3.000	3.00	0.1340	x a
Class.xb	1	8.00000000	192.000	192.000	192.00	<.0001	x b
Class.xc	1	4.00000000	48.000	48.000	48.00	0.0004	x c

Figure 126.74 *continued*

Implicit Intercept Model

The REG Procedure
Model: MODEL1
Dependent Variable: y

Number of Observations Read	9
Number of Observations Used	9

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	74.00000	37.00000	37.00	0.0004
Error	6	6.00000	1.00000		
Corrected Total	8	80.00000			

Root MSE	1.00000	**R-Square**	0.9250
Dependent Mean	4.33333	**Adj R-Sq**	0.9000
Coeff Var	23.07692		

Note: Model is not full rank. Least-squares solutions for the parameters are not unique. Some statistics will be misleading. A reported DF of 0 or B means that the estimate is biased.

Note: The following parameters have been set to 0, since the variables are a linear combination of other variables as shown.

xc =	Intercept - xa - xb

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	Intercept	B	4.00000	0.57735	6.93	0.0004
xa	x a	B	-3.00000	0.81650	-3.67	0.0104
xb	x b	B	4.00000	0.81650	4.90	0.0027
xc	x c	0	0	.	.	.

Implicit Intercept Model

The REG Procedure
Model: MODEL2
Dependent Variable: y

Number of Observations Read	9
Number of Observations Used	9

Note: No intercept in model. R-Square is redefined.

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	243.00000	81.00000	81.00	<.0001
Error	6	6.00000	1.00000		
Uncorrected Total	9	249.00000			

Figure 126.74 *continued*

Root MSE	1.00000	**R-Square**	0.9759
Dependent Mean	4.33333	**Adj R-Sq**	0.9639
Coeff Var	23.07692		

Parameter Estimates

Variable	Label	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
xa	x a	1	1.00000	0.57735	1.73	0.1340
xb	x b	1	8.00000	0.57735	13.86	<.0001
xc	x c	1	4.00000	0.57735	6.93	0.0004

Using the DESIGN Output Option

This example uses PROC TRANSREG and the DESIGN *o-option* to prepare an input data set with classification variables for the LOGISTIC procedure. The DESIGN *o-option* specifies that the goal is design matrix creation, not analysis. When you specify DESIGN, dependent variables are not required. The DEVIATIONS (or EFFECTS) *t-option* requests a deviations-from-means $(1, 0, -1)$ coding of the classification variables, which is the same coding the CATMOD procedure uses. PROC TRANSREG automatically creates a macro variable &_TrgInd that contains the list of independent variables created. This macro is used in the PROC LOGISTIC MODEL statement. (See Figure 126.75.) For comparison, the same analysis is also performed with PROC CATMOD. The following statements create Figure 126.75:

```
title 'Using PROC TRANSREG to Create a Design Matrix';

data a;
   do y = 1, 2;
      do a = 1 to 4;
         do b = 1 to 3;
            w = ceil(uniform(1) * 10 + 10);
            output;
         end;
      end;
   end;
run;

proc transreg data=a design;
   model class(a b / deviations);
   id y w;
   output out=coded;
run;

proc print;
   title2 'PROC TRANSREG Output Data Set';
run;

title2 'PROC LOGISTIC with Classification Variables';

proc logistic;
```

```
   freq w;
   model y = &_trgind;
run;

title2 'PROC CATMOD Should Produce the Same Results';

proc catmod data=a;
   model y = a b;
   weight w;
run;
```

Figure 126.75 The PROC TRANSREG Design Matrix

Using PROC TRANSREG to Create a Design Matrix
PROC LOGISTIC with Classification Variables

The LOGISTIC Procedure

Model Information	
Data Set	WORK.CODED
Response Variable	y
Number of Response Levels	2
Frequency Variable	w
Model	binary logit
Optimization Technique	Fisher's scoring

Number of Observations Read	24
Number of Observations Used	24
Sum of Frequencies Read	375
Sum of Frequencies Used	375

Response Profile

Ordered Value	y	Total Frequency
1	1	188
2	2	187

Probability modeled is y=1.

Model Convergence Status
Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Intercept Only	Intercept and Covariates
AIC	521.858	524.378
SC	525.785	547.939
-2 Log L	519.858	512.378

Figure 126.75 *continued*

Testing Global Null Hypothesis: BETA=0			
Test	**Chi-Square**	**DF**	**Pr > ChiSq**
Likelihood Ratio	7.4799	5	0.1873
Score	7.4312	5	0.1905
Wald	7.3356	5	0.1969

Analysis of Maximum Likelihood Estimates					
Parameter	**DF**	**Estimate**	**Standard Error**	**Wald Chi-Square**	**Pr > ChiSq**
Intercept	1	-0.00040	0.1044	0.0000	0.9969
a1	1	-0.0802	0.1791	0.2007	0.6542
a2	1	0.2001	0.1800	1.2363	0.2662
a3	1	-0.1350	0.1819	0.5514	0.4578
b1	1	-0.2392	0.1500	2.5436	0.1107
b2	1	0.3433	0.1474	5.4223	0.0199

Odds Ratio Estimates			
Effect	**Point Estimate**	**95% Wald Confidence Limits**	
a1	0.923	0.650	1.311
a2	1.222	0.858	1.738
a3	0.874	0.612	1.248
b1	0.787	0.587	1.056
b2	1.410	1.056	1.882

Association of Predicted Probabilities and Observed Responses			
Percent Concordant	54.0	**Somers' D**	0.163
Percent Discordant	37.8	**Gamma**	0.177
Percent Tied	8.2	**Tau-a**	0.082
Pairs	35156	**c**	0.581

Using PROC TRANSREG to Create a Design Matrix PROC CATMOD Should Produce the Same Results

The CATMOD Procedure

Data Summary			
Response	y	**Response Levels**	2
Weight Variable	w	**Populations**	12
Data Set	A	**Total Frequency**	375
Frequency Missing	0	**Observations**	24

Figure 126.75 *continued*

Population Profiles			
Sample	**a**	**b**	**Sample Size**
1	1	1	31
2	1	2	31
3	1	3	34
4	2	1	26
5	2	2	33
6	2	3	37
7	3	1	36
8	3	2	29
9	3	3	28
10	4	1	26
11	4	2	35
12	4	3	29

Response Profiles	
Response	**y**
1	1
2	2

Maximum Likelihood Analysis
Maximum likelihood computations converged.

Maximum Likelihood Analysis of Variance			
Source	**DF**	**Chi-Square**	**Pr > ChiSq**
Intercept	1	0.00	0.9969
a	3	1.50	0.6823
b	2	5.64	0.0597
Likelihood Ratio	6	2.81	0.8329

Analysis of Maximum Likelihood Estimates					
Parameter		**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Intercept		-0.00040	0.1044	0.00	0.9969
a	**1**	-0.0802	0.1791	0.20	0.6542
	2	0.2001	0.1800	1.24	0.2662
	3	-0.1350	0.1819	0.55	0.4578
b	**1**	-0.2392	0.1500	2.54	0.1107
	2	0.3434	0.1474	5.42	0.0199

Discrete Choice Experiments: DESIGN, NORESTORE, NOZERO

A discrete choice experiment is constructed consisting of four product brands, each available at three different prices, $1.49, $1.99, $2.49. In addition, each choice set contains a constant "other" alternative available at $1.49. In the fifth choice set, price is constant. PROC TRANSREG is used to code the design, and the PHREG procedure fits the multinomial logit choice model (not shown). See Kuhfeld (2010) for more information about discrete choice modeling and the multinomial logit model; look for the latest "Discrete Choice" report. The following statements produce Figure 126.76:

```
title 'Choice Model Coding';

data design;
   array p[4];
   input p1-p4 @@;
   set = _n_;
   do brand = 1 to 4;
      price = p[brand];
      output;
   end;
   brand = .; price = 1.49; output; /* constant alternative */
   keep set brand price;
   datalines;
1.49 1.99 1.49 1.99 1.99 1.99 2.49 1.49 1.99 1.49 1.99 1.49
1.99 1.49 2.49 1.99 1.49 1.49 1.49 1.49 2.49 1.49 1.99 2.49
1.49 1.49 2.49 2.49 2.49 2.49 1.49 1.49 1.49 2.49 2.49 1.99
2.49 2.49 2.49 1.49 1.99 2.49 1.49 2.49 2.49 1.99 2.49 2.49
2.49 1.49 1.49 1.99 1.49 1.99 1.99 1.49 2.49 1.99 1.99 1.99
1.99 1.99 1.49 2.49 1.99 2.49 1.99 1.99 1.49 2.49 1.99 2.49
;

proc transreg data=design design norestoremissing nozeroconstant;
   model class(brand / zero=none) identity(price);
   output out=coded;
   by set;
run;

proc print data=coded(firstobs=21 obs=25);
   var set brand &_trgind;
run;
```

In the interest of space, only the fifth choice set is displayed in Figure 126.76.

Figure 126.76 The Fifth Choice Set

Choice Model Coding

Obs	set	brand	brand1	brand2	brand3	brand4	price
21	5	1	1	0	0	0	1.49
22	5	2	0	1	0	0	1.49
23	5	3	0	0	1	0	1.49
24	5	4	0	0	0	1	1.49
25	5	.	0	0	0	0	1.49

For the constant alternative (Brand = .), the brand coding is a row of zeros due to the NORESTOREMISSING *o-option*, and Price is a constant $1.49 (instead of 0) due to the NOZEROCONSTANT.

The data set was coded by choice set (BY set;). This is a small problem. With very large problems, it might be necessary to restrict the number of observations that are coded at one time so that the procedure uses less time and memory. Coding by choice set is one option. When coding is performed after the data are merged in, coding by subject and choice set combinations is another option. Alternatively, you can specify DESIGN=*n*, where *n* is the number of observations to code at one time. For example, you can specify DESIGN=100 or DESIGN=1000 to process the data set in blocks of 100 or 1000 observations. Specify the NOZEROCONSTANT *a-option* to ensure that constant variables within blocks are not zeroed. When you specify DESIGN=*n*, or perform coding after the data are merged in, specify the dependent variable and any other variables needed for analysis as ID variables.

Centering

You can use transformation options to center and standardize the variables in several ways. For example, the following MODEL statement creates three independent variables, x, x^2, and x^3:

```
model identity(y) = pspline(x);
```

The variables are not centered.

When the CENTER *t-option* is specified, as in the following statement, the independent variable is centered before squaring and cubing:

```
model identity(y) = pspline(x / center);
```

The three independent variables are $x - \bar{x}$, $(x - \bar{x})^2$, and $(x - \bar{x})^3$.

Since operations such as squaring occur after the centering, the resulting variables are not always centered. The CENTER *t-option* is particularly useful with polynomials since centering before squaring and cubing can help reduce collinearity and numerical problems. For example, if one of your variables is year, with values all greater than 1900, squaring and cubing without centering first will create variables that are all essentially perfectly correlated.

When the TSTANDARD=CENTER *t-option* is specified, as in the following model, the three independent variables are squared and cubed and then centered:

```
model identity(y) = pspline(x / tstandard=center);
```

The three independent variables are $x - \bar{x}$, $x^2 - \overline{x^2}$, and $x^3 - \overline{x^3}$.

You can specify both the CENTER and TSTANDARD=CENTER *t-options* to center the variables, then square and cube them, and then center the results, as in the following statement:

```
model identity(y) = pspline(x / center tstandard=center);
```

The three independent variables are $x - \bar{x}$, $(x - \bar{x})^2 - \overline{(x - \bar{x})^2}$, and $(x - \bar{x})^3 - \overline{(x - \bar{x})^3}$.

Displayed Output

The display options control the amount of displayed output. The displayed output can contain the following:

- an iteration history and convergence status table (by default when there are iterations)
- an ANOVA table when the TEST, SS2, or UTILITIES *a-option* is specified
- a regression table when the SS2 *a-option* is specified
- conjoint analysis part-worth utilities when the UTILITIES *a-option* is specified
- model details when the DETAIL *a-option* is specified
- a multivariate ANOVA table when the dependent variable is transformed and the TEST or SS2 *a-option* is specified
- a multivariate regression table when the dependent variable is transformed and it is specified
- liberal and conservative ANOVA, multivariate ANOVA, regression, and multivariate regression tables when there is a MONOTONE, UNTIE, or MSPLINE transformation and the TEST or SS2 *a-option* is specified

ODS Table Names

PROC TRANSREG assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 126.8. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 126.8 ODS Tables Produced by PROC TRANSREG

ODS Table Name	Description	Statement & Option
ANOVA	ANOVA	MODEL/PROC, TEST/SS2
BoxCox	Box-Cox transformation results	MODEL, BOXCOX
CANALS	CANALS iteration history	MODEL/PROC, METHOD=CANALS
ClassLevels	ANOVA	MODEL/PROC, TEST/SS2
Coef	Regression results	MODEL/PROC, SS2
ConservANOVA	ANOVA, *1	MODEL/PROC, TEST/SS2
ConservCoef	Regression results, *1	MODEL/PROC, SS2
ConservFitStatistics	Fit statistics, *1	MODEL/PROC, TEST/SS2
ConservMVANOVA	Multivariate ANOVA, *1, *2	MODEL/PROC, TEST/SS2
ConservMVCoef	Multivariate regression results, *1, *2	MODEL/PROC, SS2
ConservUtilities	Conjoint analysis utilities, *1	MODEL/PROC, UTILITIES
ConvergenceStatus	Convergence status	default

Table 126.8 *continued*

ODS Table Name	Description	Statement & Option
Details	Model Details	MODEL/PROC, DETAIL
Equation	Linear dependency equation	less-than-full-rank model
FitStatistics	Fit statistics like R square	MODEL/PROC, TEST/SS2
Footnotes	Iteration history footnotes	default
Formula	Fit plot formula (nonprinting)	PROC, PLOTS=FIT(FORMULA)
LiberalANOVA	ANOVA, *1	MODEL/PROC, TEST/SS2
LiberalCoef	Regression results, *1	MODEL/PROC, SS2
LiberalFitStatistics	Fit statistics, *1	MODEL/PROC, TEST/SS2
LiberalMVANOVA	Multivariate ANOVA, *1, *2	MODEL/PROC, TEST/SS2
LiberalMVCoef	Multivariate regression results, *1, *2	MODEL/PROC, SS2
LiberalUtilities	Conjoint analysis utilities, *1	MODEL/PROC, UTILITIES
MORALS	MORALS iteration history	MODEL/PROC, METHOD=MORALS
MVANOVA	Multivariate ANOVA, *2	MODEL/PROC, TEST/SS2
MVCoef	Multivariate regression results, *2	MODEL/PROC, SS2
NObs	ANOVA	MODEL/PROC, TEST/SS2
PBSplineCriteria	Penalized B-spline criteria (nonprinting)	MODEL, PBSPLINE
RSquare	R square	MODEL/PROC, RSQUARE
Redundancy	Redundancy iteration history	MODEL/PROC, METHOD=REDUNDANCY
SplineCoef	Spline coefficients (nonprinting)	MODEL, SPLINE/MSPLINE
TestIterations	Hypothesis test iterations iteration history	MODEL/PROC, SS2
Univariate	Univariate iteration history	MODEL/PROC, METHOD=UNIVARIATE
Utilities	Conjoint analysis utilities	MODEL/PROC, UTILITIES

*1. Liberal and conservative test tables are produced when a MONOTONE, UNTIE, or MSPLINE transformation is requested.

*2. Multivariate tables are produced when the dependent variable is iteratively transformed.

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

Some graphs are produced by default; other graphs are produced by using statements and options. You can reference every graph produced through ODS Graphics with a name. The names of the graphs that PROC TRANSREG generates are listed in Table 126.9, along with the required statements and options.

Table 126.9 Graphs Produced by PROC TRANSREG

ODS Graph Name	Plot Description	Statement & Option
BoxCoxFPlot	Box-Cox $F = t^2$	MODEL & PROC, BOXCOX *transform* & PLOTS(UNPACK)
BoxCoxLogLikePlot	Box-Cox Log Likelihood	MODEL & PROC, BOXCOX *transform* & PLOTS(UNPACK)
BoxCoxPlot	Box-Cox t or $F = t^2$ & Log Likelihood	MODEL, BOXCOX *transform*
BoxCoxtPlot	Box-Cox t	MODEL & PROC, BOXCOX *transform* & PLOTS(UNPACK)=BOXCOX(T)
FitPlot	Simple Regression and Separate Group Regressions	MODEL, a dependent variable that is not transformed, one non-CLASS independent variable, and at most one CLASS variable
ObservedByPredicted	Dependent Variable by Predicted Values	MODEL, PLOTS=OBSERVEDBYPREDICTED
PBSPlineCritPlot	Penalized B-Spline Criterion Plot	MODEL, PBSPLINE *transform*
PrefMapVecPlot	Preference Mapping Vector Plot	MODEL & PROC, IDENTITY *transform* & COORDINATES
PrefMapIdealPlot	Preference Mapping Ideal Point Plot	MODEL & PROC, POINT expansion & COORDINATES
ResidualPlot	Residuals	PROC, PLOTS=RESIDUALS
RMSEPlot	Box-Cox Root Mean Square Error	MODEL & PROC, BOXCOX *transform* & PLOTS=BOXCOX(RMSE)
ScatterPlot	Scatter Plot of Observed Data	MODEL, one non-CLASS independent variable, and at most one CLASS variable, PLOTS=SCATTER
TransformationPlot	Variable Transformations	PROC, PLOTS=TRANSFORMATION

The PLOTS(INTERPOLATE) Option

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/tregint.sas.)

This section illustrates one use of the PLOTS(INTERPOLATE) option for use with ODS Graphics. The data set has two groups of observations, c = 1 and c = 2. Each group is sparse, having only five observations, so the plots of the transformations and fit functions are not smooth. A second DATA step adds additional observations to the data set, over the range of x, with y missing. These observations do not contribute to the analysis, but they are used in computations of transformed and predicted values. The resulting plots are much smoother in the latter case than in the former. The other results of the analysis are the same. The following statements produce Figure 126.77 and Figure 126.78:

```
title 'Smoother Interpolation with PLOTS(INTERPOLATE)';

data a;
   input c y x;
   output;
   datalines;
1 1 1
1 2 2
1 4 3
1 6 4
1 7 5
2 3 1
2 4 2
2 5 3
2 4 4
2 5 5
;

ods graphics on;

proc transreg data=a plots=(tran fit) ss2;
   model ide(y) = pbs(x) * class(c / zero=none);
run;

data b;
   set a end=eof;
   output;
   if eof then do;
      y = .;
      do x = 1 to 5 by 0.05;
         c = 1; output;
         c = 2; output;
      end;
   end;
run;

proc transreg data=b plots(interpolate)=(tran fit) ss2;
   model ide(y) = pbs(x) * class(c / zero=none);
run;
```

The results with no interpolation are shown in Figure 126.77. The transformation and fit functions are not at

all smooth. The results with interpolation are shown in Figure 126.78. The transformation and fit functions are smooth in Figure 126.78, because there are intermediate points to plot.

Figure 126.77 No Interpolation

Smoother Interpolation with PLOTS(INTERPOLATE)

The TRANSREG Procedure

Univariate ANOVA Table, Penalized B-Spline Transformation

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	9	28.90000	3.211111	Infty	<.0001
Error	18E-9	0.00000	0.000000		
Corrected Total	9	28.90000			

Root MSE	0	R-Square	1.0000
Dependent Mean	4.10000	Adj R-Sq	1.0000
Coeff Var	0		

Penalized B-Spline Transformation

Variable	DF	Coefficient	Lambda	AICC	Label
Pbspline(xc1)	5.0000	1.000	2.642E-7	-66.4330	x * c 1
Pbspline(xc2)	5.0000	1.000	2.513E-7	-60.6440	x * c 2

Figure 126.77 *continued*

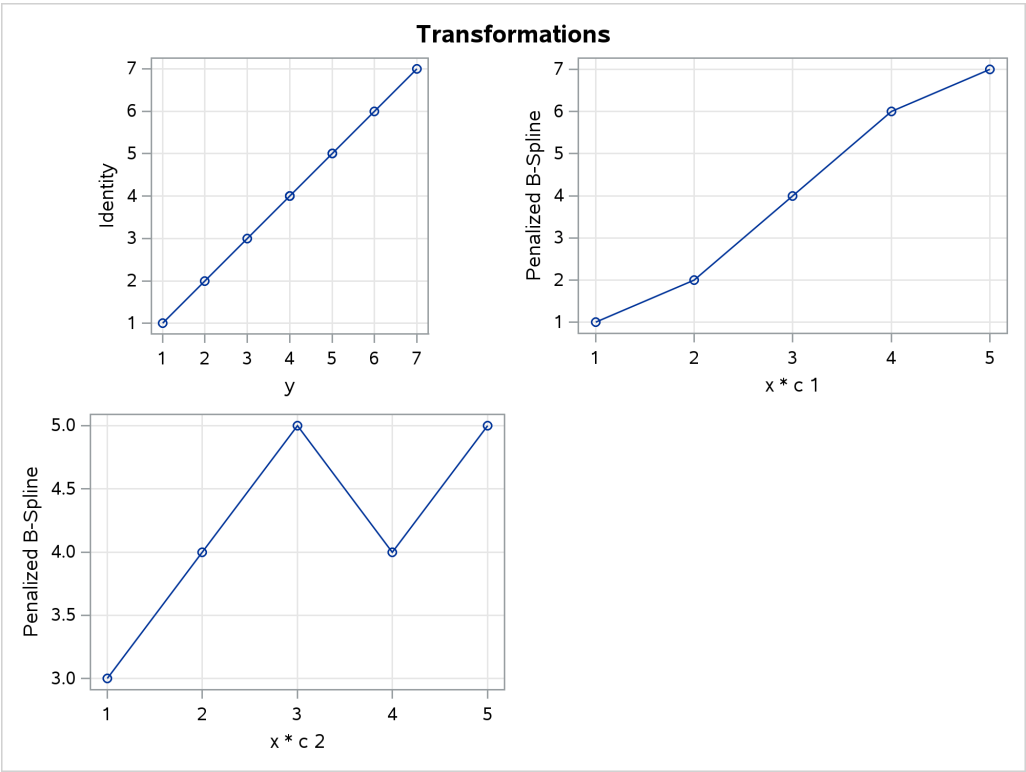

Figure 126.77 *continued*

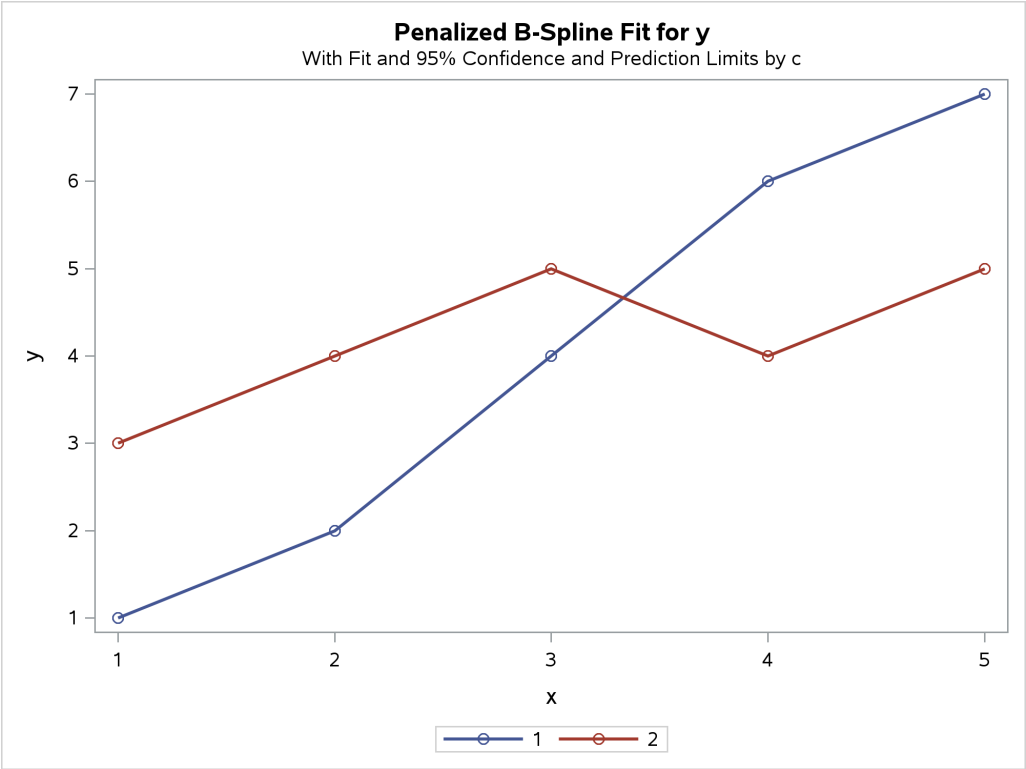

Figure 126.78 Interpolation with PLOTS(INTERPOLATE)

Smoother Interpolation with PLOTS(INTERPOLATE)

The TRANSREG Procedure

Univariate ANOVA Table, Penalized B-Spline Transformation

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	9	28.90000	3.211111	Infty	<.0001
Error	18E-9	0.00000	0.000000		
Corrected Total	9	28.90000			

Root MSE	0	**R-Square**	1.0000
Dependent Mean	4.10000	**Adj R-Sq**	1.0000
Coeff Var	0		

Penalized B-Spline Transformation

Variable	DF	Coefficient	Lambda	AICC	Label
Pbspline(xc1)	5.0000	1.000	2.642E-7	-66.4330	x * c 1
Pbspline(xc2)	5.0000	1.000	2.513E-7	-60.6440	x * c 2

Figure 126.78 *continued*

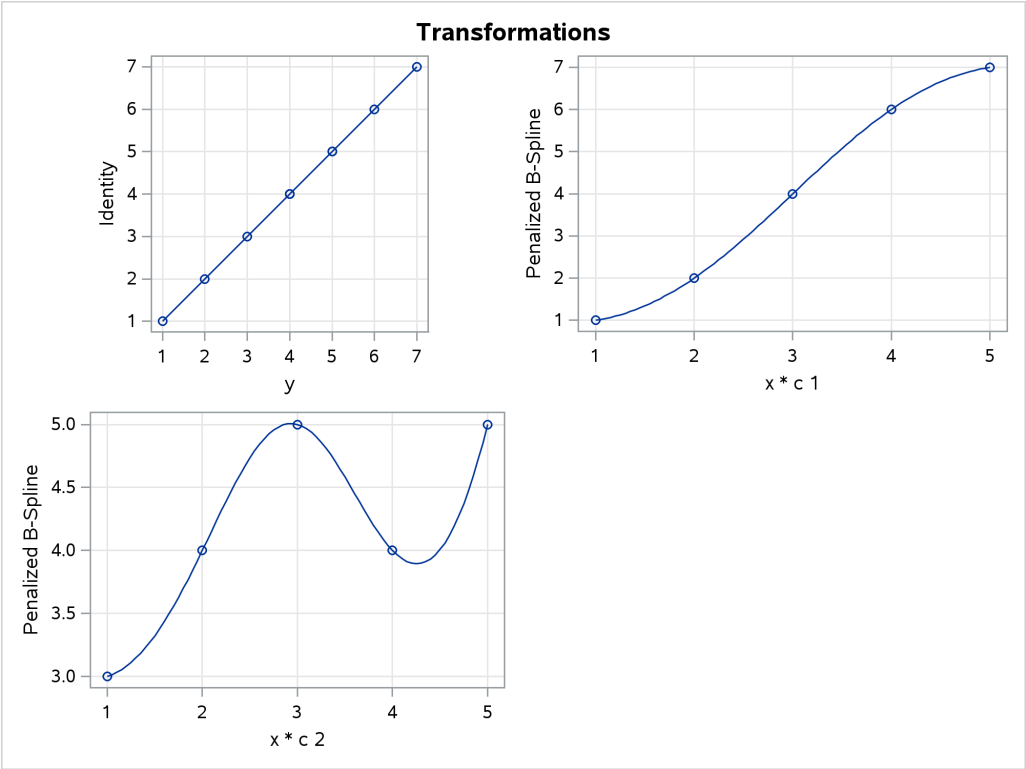

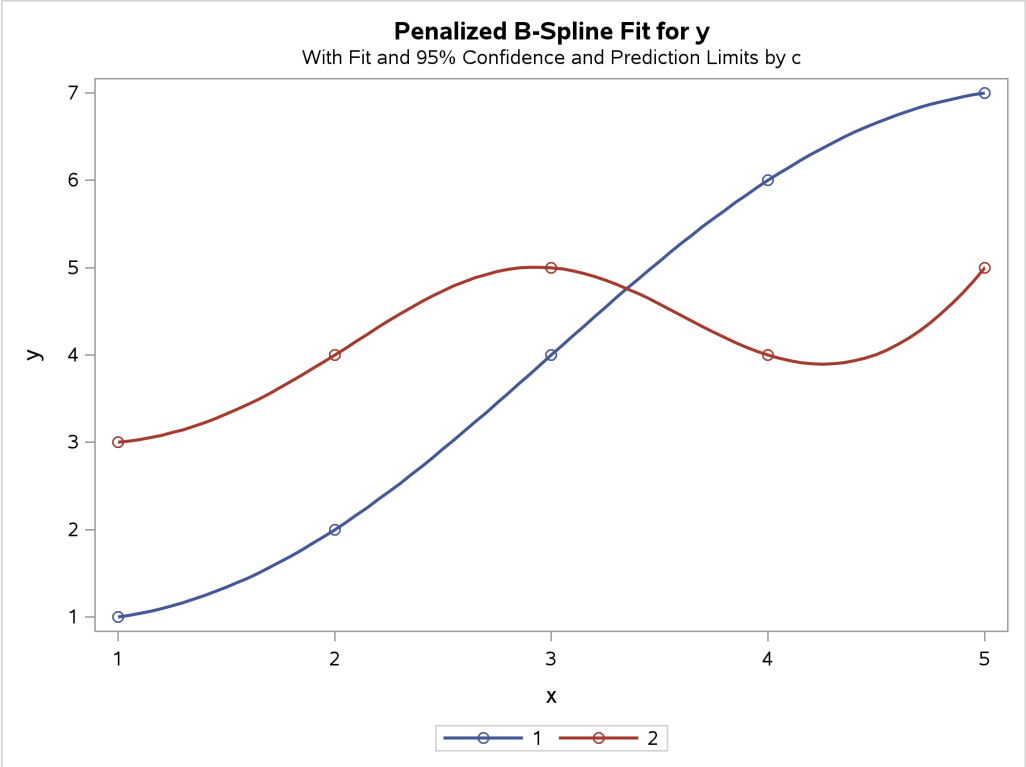

Examples: TRANSREG Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 126.1: Transformation Regression of Exhaust Emissions Data

(View the complete code for this example (tregex1.sas) in the example repository.)

In this example, the data are from an experiment in which nitrogen oxide emissions from a single cylinder engine are measured for various combinations of fuel, compression ratio, and equivalence ratio. The data are provided by Brinkman (1981). This gas data set is available from the Sashelp library.

The equivalence ratio and nitrogen oxide variables are continuous and numeric, so spline transformations of these variables are requested. The spline transformation of the dependent variable is restricted to be monotonic. Each spline is degree three with nine knots (one at each decile) in order to give PROC TRANSREG a great deal of freedom in finding transformations. The compression ratio variable has only five discrete values, so an optimal scoring is requested with monotonicity constraints. The character variable Fuel is nominal, so it is optimally scored without any monotonicity constraints. Observations with missing values are excluded with the NOMISS *a-option*.

```
ods graphics on;

title 'Gasoline Example';
title2 'Iteratively Estimate NOx, CpRatio, EqRatio, and Fuel';

* Fit the Nonparametric Model;
proc transreg data=sashelp.Gas solve test nomiss plots=all;
   ods exclude where=(_path_ ? 'MV');
   model mspline(NOx / nknots=9) = spline(EqRatio / nknots=9)
                                   monotone(CpRatio) opscore(Fuel);
run;
```

Output 126.1.1 Transformation Regression Example: The Nonparametric Model

Gasoline Example
Iteratively Estimate NOx, CpRatio, EqRatio, and Fuel

The TRANSREG Procedure

Dependent Variable Mspline(NOx)
Nitrogen Oxide

Number of Observations Read	171
Number of Observations Used	169

Output 126.1.1 *continued*

TRANSREG MORALS Algorithm Iteration History for Mspline(NOx)

Iteration Number	Average Change	Maximum Change	R-Square	Criterion Change	Note
0	0.41900	3.80550	0.05241		
1	0.11984	0.83327	0.91028	0.85787	
2	0.03727	0.17688	0.93981	0.02953	
3	0.02795	0.10880	0.94969	0.00987	
4	0.02088	0.07279	0.95382	0.00413	
5	0.01530	0.05031	0.95582	0.00201	
6	0.01130	0.03922	0.95688	0.00106	
7	0.00852	0.03197	0.95748	0.00060	
8	0.00657	0.02531	0.95783	0.00035	
9	0.00510	0.01975	0.95805	0.00022	
10	0.00398	0.01534	0.95818	0.00013	
11	0.00314	0.01200	0.95827	0.00009	
12	0.00250	0.00953	0.95832	0.00005	
13	0.00199	0.00752	0.95836	0.00003	
14	0.00159	0.00594	0.95838	0.00002	
15	0.00127	0.00470	0.95839	0.00001	
16	0.00102	0.00373	0.95840	0.00001	
17	0.00081	0.00297	0.95841	0.00001	
18	0.00065	0.00237	0.95841	0.00000	
19	0.00052	0.00189	0.95841	0.00000	
20	0.00042	0.00151	0.95842	0.00000	
21	0.00033	0.00120	0.95842	0.00000	
22	0.00027	0.00096	0.95842	0.00000	
23	0.00021	0.00077	0.95842	0.00000	
24	0.00017	0.00061	0.95842	0.00000	
25	0.00014	0.00049	0.95842	0.00000	
26	0.00011	0.00039	0.95842	0.00000	
27	0.00009	0.00031	0.95842	0.00000	
28	0.00007	0.00025	0.95842	0.00000	
29	0.00006	0.00020	0.95842	0.00000	
30	0.00005	0.00016	0.95842	0.00000	Not Converged

WARNING: Failed to converge, however criterion change is less than 0.0001.

Output 126.1.1 *continued*

The TRANSREG Procedure Hypothesis Tests for Mspline(NOx) Nitrogen Oxide

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Liberal p
Model	21	326.0176	15.52465	161.35	>= <.0001
Error	147	14.1443	0.09622		
Corrected Total	168	340.1619			

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Root MSE	0.31019	**R-Square**	0.9584
Dependent Mean	2.34593	**Adj R-Sq**	0.9525
Coeff Var	13.22262		

Output 126.1.1 *continued*

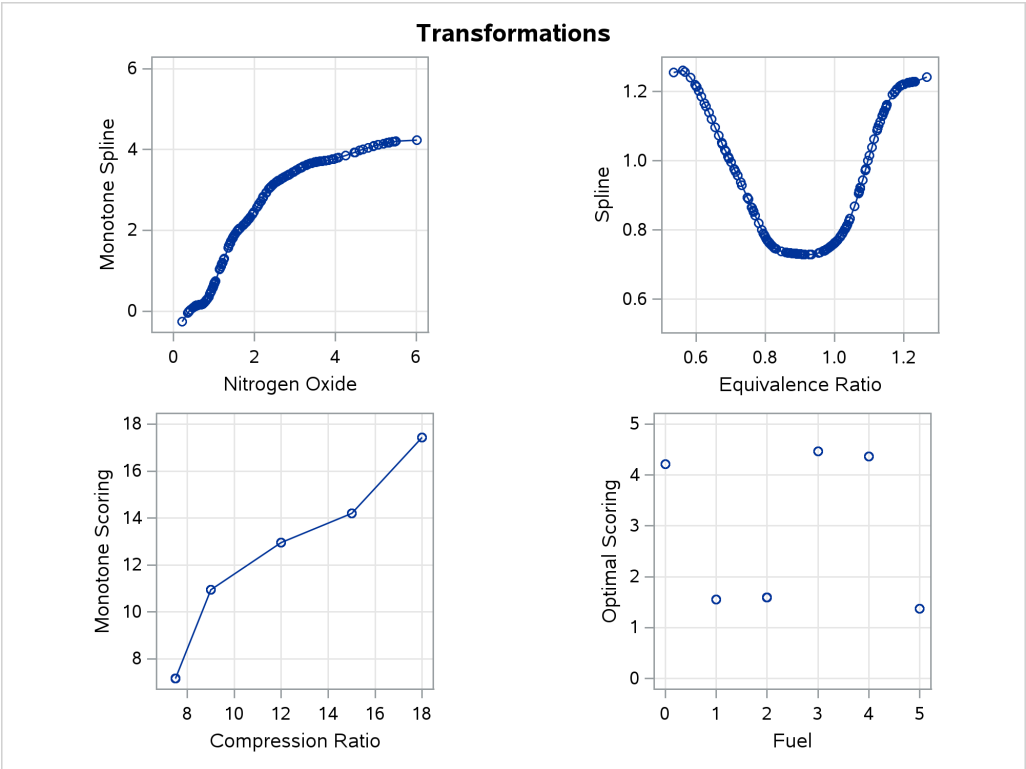

Output 126.1.1 *continued*

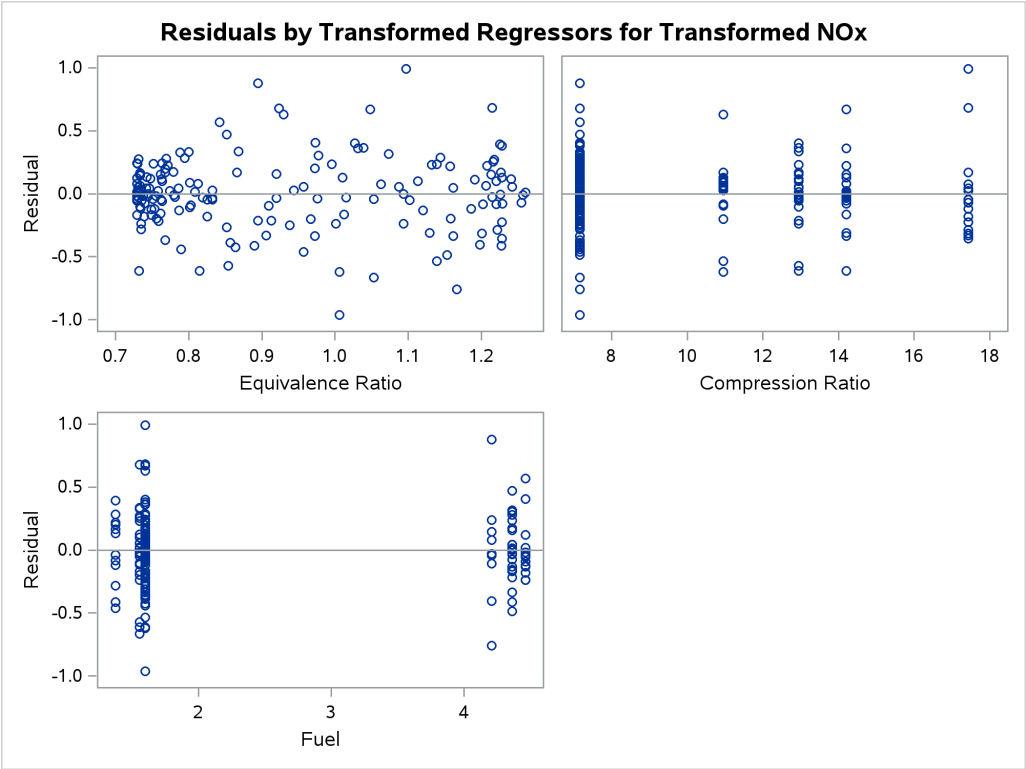

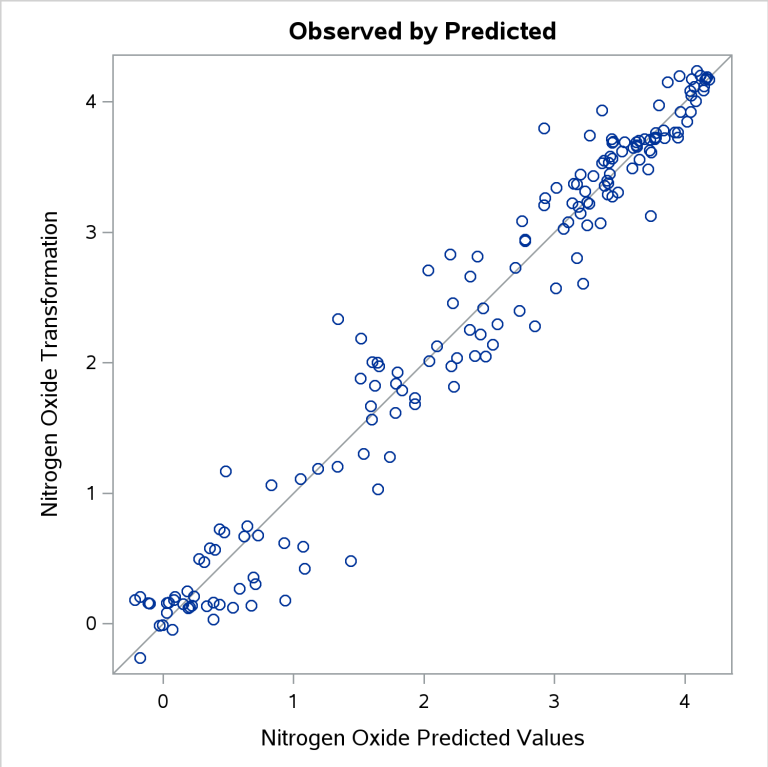

The squared multiple correlation for the initial model is approximately 0.05. PROC TRANSREG increases

the R square to over 0.95 by transforming the variables. The transformation plots show how each variable is transformed. The transformation of compression ratio (TCpRatio) is nearly linear. The transformation of equivalence ratio (TEqRatio) is nearly parabolic. It can be seen from this plot that the optimal transformation of equivalence ratio is nearly uncorrelated with the original scoring. This suggests that the large increase in R square is due to this transformation. The transformation of nitrogen oxide (TNOx) is similar to a log transformation. The final plot shows the transformed dependent variable plotted as a function of the predicted values. This plot is reasonably linear, showing that the nonlinearities in the data are being accounted for fairly well by the TRANSREG model.

These results suggest the parametric model

$$\begin{aligned}\log(\text{NOX}) &= b_0 + b_1 \times \text{EqRatio} + b_2 \times \text{EqRatio}^2 + b_3 \times \text{CpRatio} \\ &\quad + \sum_j b_j \text{class}_j(\text{Fuel}) + \text{error}\end{aligned}$$

You can perform this analysis with PROC TRANSREG. The following statements produce Output 126.1.2:

```
title2 'Now fit log(NOx) = b0 + b1*EqRatio + b2*EqRatio**2 +';
title3 'b3*CpRatio + Sum b(j)*Fuel(j) + Error';

*-Fit the Parametric Model Suggested by the Nonparametric Analysis-;
proc transreg data=sashelp.Gas solve ss2 short nomiss plots=all;
   model log(NOx) = pspline(EqRatio / deg=2) identity(CpRatio)
                    opscore(Fuel);
run;
```

Output 126.1.2 Transformation Regression Example: The Parametric Model

Gasoline Example
Now fit log(NOx) = b0 + b1*EqRatio + b2*EqRatio2 +**
b3*CpRatio + Sum b(j)*Fuel(j) + Error

The TRANSREG Procedure

Dependent Variable Log(NOx)
Nitrogen Oxide

Number of Observations Read	171
Number of Observations Used	169

Log(NOx)
Algorithm converged.

Output 126.1.2 *continued*

The TRANSREG Procedure Hypothesis Tests for Log(NOx) Nitrogen Oxide

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	8	79.33838	9.917298	213.09	<.0001
Error	160	7.44659	0.046541		
Corrected Total	168	86.78498			

Root MSE	0.21573	**R-Square**	0.9142
Dependent Mean	0.63130	**Adj R-Sq**	0.9099
Coeff Var	34.17294		

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Pr > F	Label
Intercept	1	-15.274649	57.1338	57.1338	1227.60	<.0001	Intercept
Pspline.EqRatio_1	1	35.102914	62.7478	62.7478	1348.22	<.0001	Equivalence Ratio 1
Pspline.EqRatio_2	1	-19.386468	64.6430	64.6430	1388.94	<.0001	Equivalence Ratio 2
Identity(CpRatio)	1	0.032058	1.4445	1.4445	31.04	<.0001	Compression Ratio
Opscore(Fuel)	5	0.158388	5.5619	1.1124	23.90	<.0001	Fuel

Output 126.1.2 *continued*

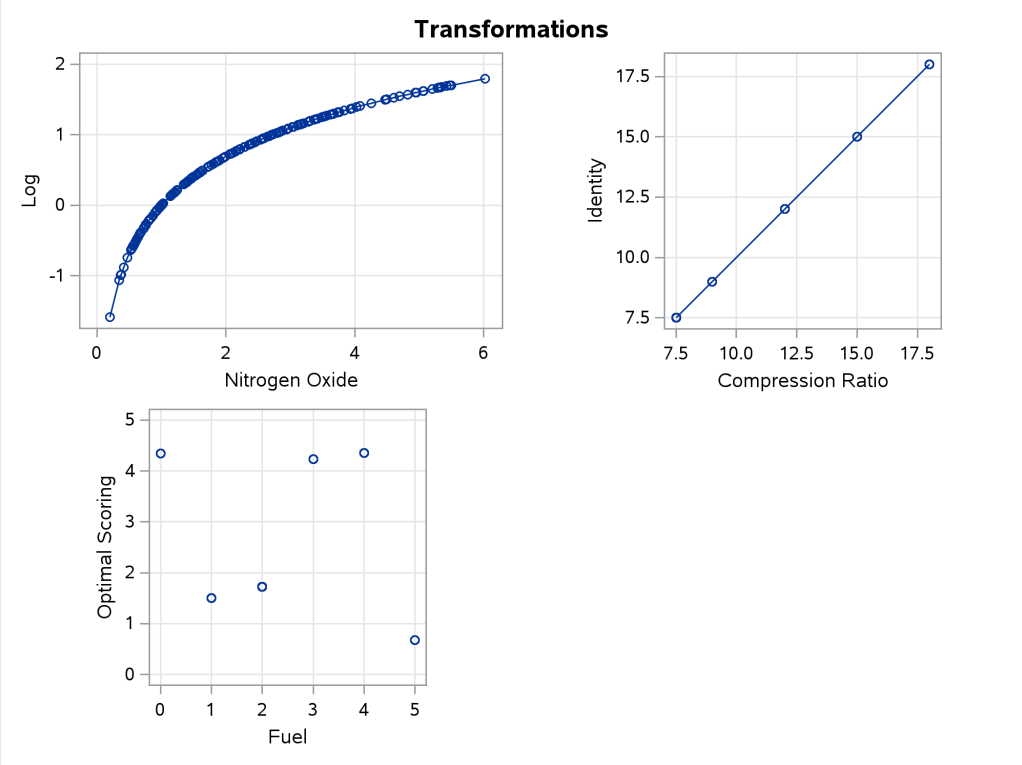

Output 126.1.2 *continued*

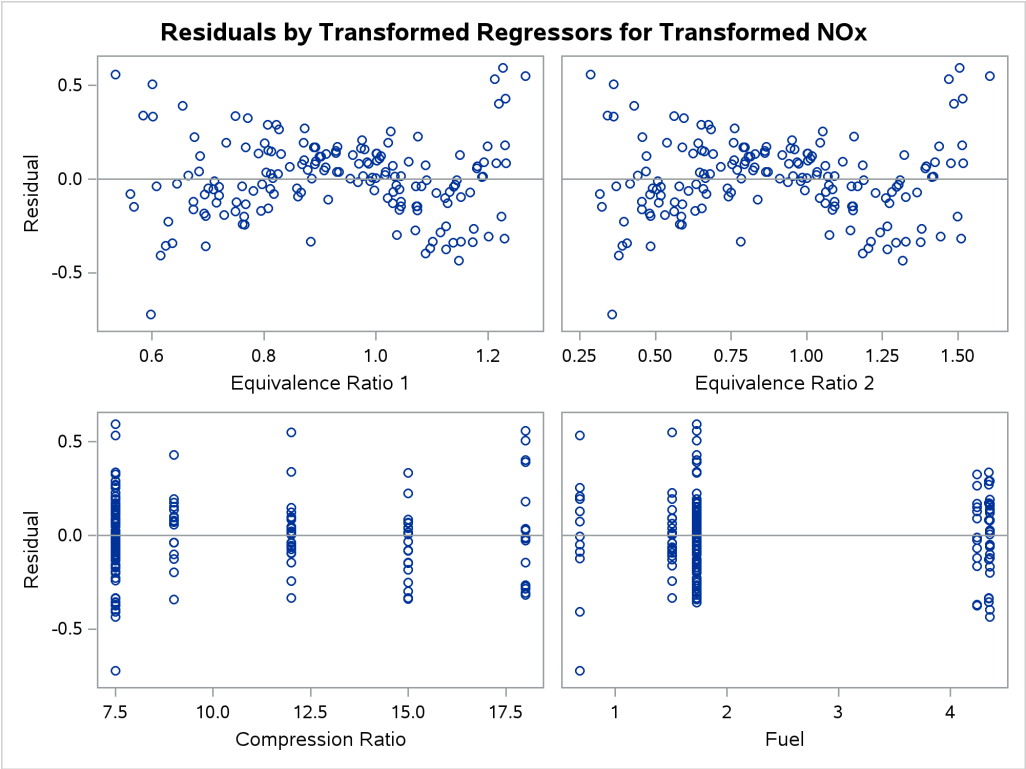

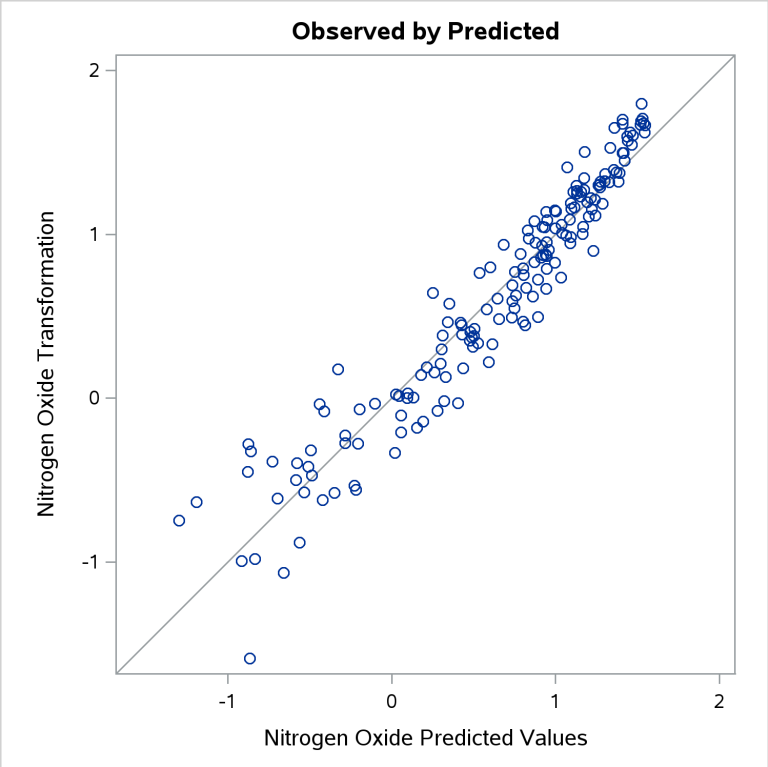

The LOG transformation computes the natural log. The PSPLINE expansion expands EqRatio into a linear

term, EqRatio, and a squared term, EqRatio^2. An identity transformation of CpRatio and an optimal scoring of Fuel is requested. These should provide a good parametric operationalization of the optimal transformations. The final model has an R square of 0.91 (smaller than before since the model has fewer parameters, but still quite good).

Example 126.2: Box-Cox Transformations

(View the complete code for this example (tregex2.sas) in the example repository.)

This example shows Box-Cox transformations with a yarn failure data set. For more information about Box-Cox transformations, including using a Box-Cox transformation in a model with no independent variable, to normalize the distribution of the data, see the section "Box-Cox Transformations" on page 10623. In this example, a simple 3^3 design was used to study the effects of different factors on the failure of a yarn manufacturing process. The design factors are as follows:

- the length of test specimens of yarn, with levels of 250, 300, and 350 mm
- the amplitude of the loading cycle, with levels of 8, 9, and 10 mmd
- the load with levels of 40, 45, and 50 grams

The measured response was time (in cycles) until failure. However, you could just as well have measured the inverse of time until failure (in other words, the failure rate). Hence, the correct metric with which to analyze the response is not apparent. You can use PROC TRANSREG to find an optimum power transformation for the analysis. The following statements create the input SAS data set:

```
title 'Yarn Strength';

proc format;
   value a -1 =   8 0 =   9 1 =  10;
   value l -1 = 250 0 = 300 1 = 350;
   value o -1 =  40 0 =  45 1 =  50;
run;

data yarn;
   input Fail Amplitude Length Load @@;
   format amplitude a. length l. load o.;
   label fail = 'Time in Cycles until Failure';
   datalines;
 674 -1 -1 -1    370 -1 -1  0    292 -1 -1  1    338  0 -1 -1
 266  0 -1  0    210  0 -1  1    170  1 -1 -1    118  1 -1  0
  90  1 -1  1   1414 -1  0 -1   1198 -1  0  0    634 -1  0  1
1022  0  0 -1    620  0  0  0    438  0  0  1    442  1  0 -1
 332  1  0  0    220  1  0  1   3636 -1  1 -1   3184 -1  1  0
2000 -1  1  1   1568  0  1 -1   1070  0  1  0    566  0  1  1
1140  1  1 -1    884  1  1  0    360  1  1  1
;
```

PROC TRANSREG is run to find the Box-Cox transformation. The lambda list is –2 TO 2 BY 0.05, which produces 81 lambdas, and a convenient lambda is requested. This many power parameters makes a nice graphical display with plenty of detail around the confidence interval. In the interest of space, only part of this table is displayed. The independent variables are designated with the QPOINT expansion. QPOINT, for

quadratic point model, gets its name from PROC TRANSREG's ideal point modeling capabilities, which process variables for a response surface analysis. What QPOINT does is create a set of independent variables consisting of the following: the m original variables (Length Amplitude Load), the m original variables squared (Length_2 Amplitude_2 Load_2), and the $m \times (m-1)/2 = 3$ pairs of products between the m variables (LengthAmplitude LengthLoad AmplitudeLoad). The following statements produce Output 126.2.1:

```
ods graphics on;

proc transreg details data=yarn ss2
              plots=(transformation(dependent) obp);
   model BoxCox(fail / convenient lambda=-2 to 2 by 0.05) =
         qpoint(length amplitude load);
run;
```

Output 126.2.1 Box-Cox Yarn Data

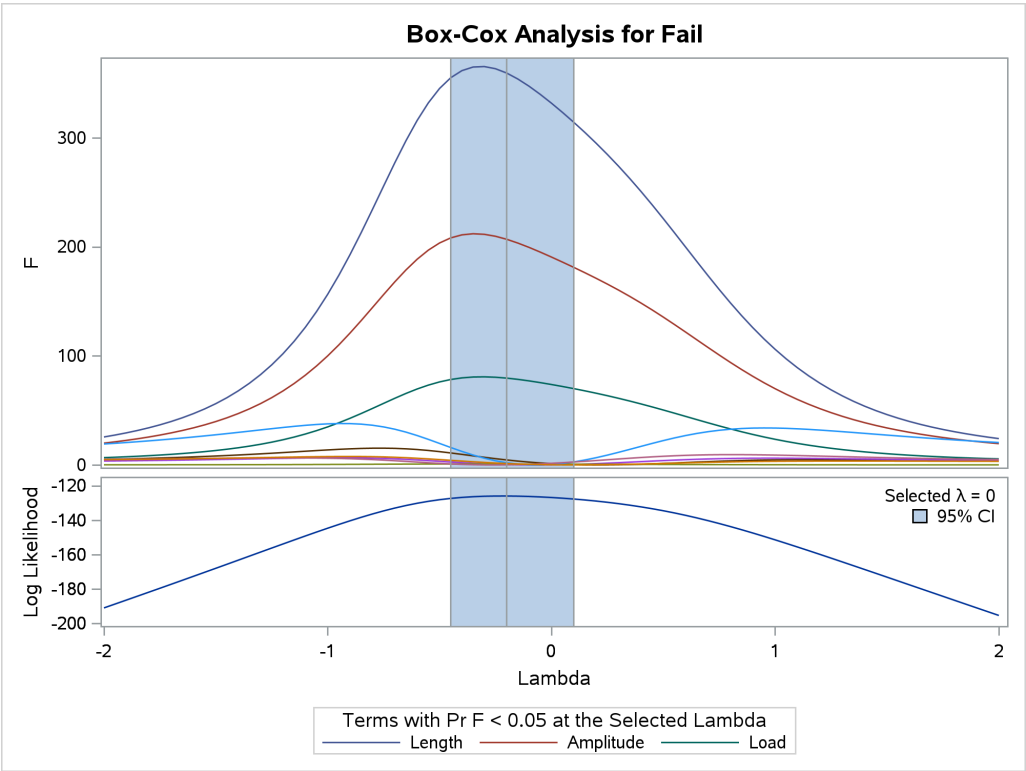

Output 126.2.1 *continued*

Dependent Variable BoxCox(Fail)
Time in Cycles until Failure

Number of Observations Read	27
Number of Observations Used	27

Output 126.2.1 *continued*

Model Statement Specification Details				
Type	**DF**	**Variable**	**Description**	**Value**
Dep	1	BoxCox(Fail)	Lambda Used	0
			Lambda	-0.2
			Log Likelihood	-125.9
			Conv. Lambda	0
			Conv. Lambda LL	-126.7
			CI Limit	-127.8
			Alpha	0.05
			Options	Convenient Lambda Used
			Label	Time in Cycles until Failure
Ind	1	Qpoint.Length	DF	1
Ind	1	Qpoint.Amplitude	DF	1
Ind	1	Qpoint.Load	DF	1
Ind	1	Qpoint.Length_2	DF	1
Ind	1	Qpoint.Amplitude_2	DF	1
Ind	1	Qpoint.Load_2	DF	1
Ind	1	Qpoint.LengthAmplitude	DF	1
Ind	1	Qpoint.LengthLoad	DF	1
Ind	1	Qpoint.AmplitudeLoad	DF	1

The TRANSREG Procedure Hypothesis Tests for BoxCox(Fail) Time in Cycles until Failure

Univariate ANOVA Table Based on the Usual Degrees of Freedom

Source	DF	Sum of Squares	Mean Square	F Value	Liberal p
Model	9	22.56498	2.507220	66.73	>= <.0001
Error	17	0.63871	0.037571		
Corrected Total	26	23.20369			

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Root MSE	0.19383	**R-Square**	0.9725
Dependent Mean	6.33466	**Adj R-Sq**	0.9579
Coeff Var	3.05987	**Lambda**	0.0000

Output 126.2.1 *continued*

Univariate Regression Table Based on the Usual Degrees of Freedom

Variable	DF	Coefficient	Type II Sum of Squares	Mean Square	F Value	Liberal p	Label
Intercept	1	6.4206207	159.008	159.008	4232.19	>= <.0001	Intercept
Qpoint.Length	1	0.8323842	12.472	12.472	331.94	>= <.0001	Length
Qpoint.Amplitude	1	-0.6309916	7.167	7.167	190.75	>= <.0001	Amplitude
Qpoint.Load	1	-0.3924940	2.773	2.773	73.80	>= <.0001	Load
Qpoint.Length_2	1	-0.0856974	0.044	0.044	1.17	>= 0.2939	Length_2
Qpoint.Amplitude_2	1	0.0242183	0.004	0.004	0.09	>= 0.7633	Amplitude_2
Qpoint.Load_2	1	-0.0674555	0.027	0.027	0.73	>= 0.4058	Load_2
Qpoint.LengthAmplitude	1	-0.0382414	0.018	0.018	0.47	>= 0.5035	LengthAmplitude
Qpoint.LengthLoad	1	-0.0684146	0.056	0.056	1.49	>= 0.2381	LengthLoad
Qpoint.AmplitudeLoad	1	-0.0208340	0.005	0.005	0.14	>= 0.7142	AmplitudeLoad

The above statistics are not adjusted for the fact that the dependent variable was transformed and so are generally liberal.

Output 126.2.1 *continued*

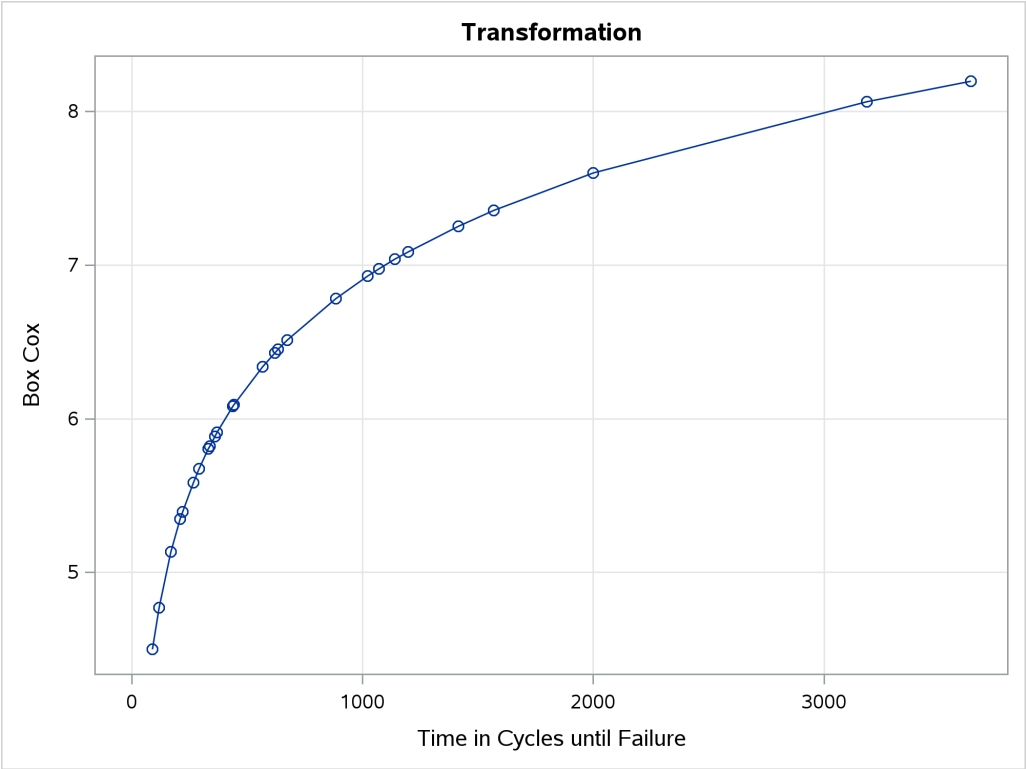

Output 126.2.1 *continued*

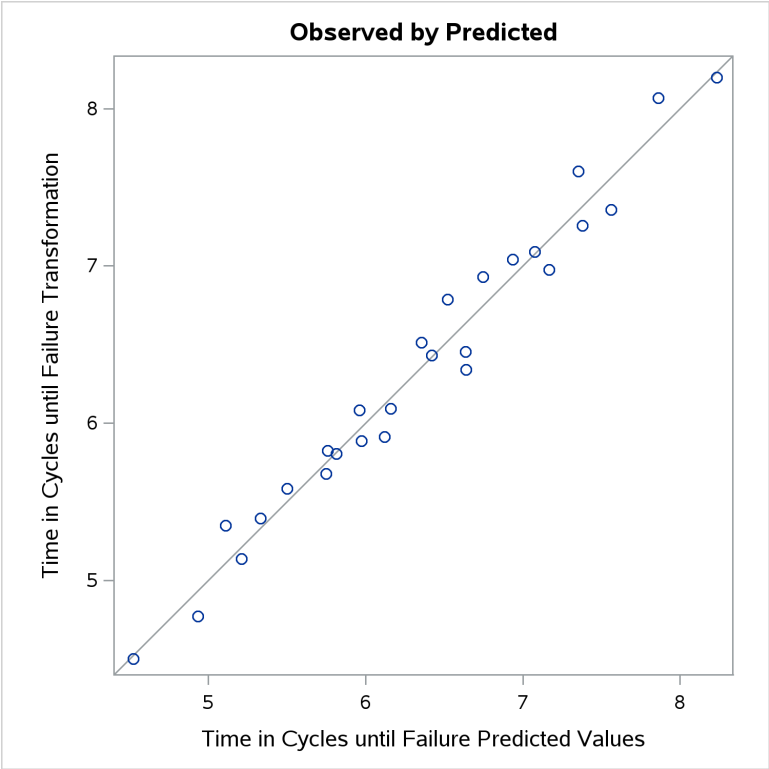

The optimal power parameter is –0.20, but since 0.0 is in the confidence interval, and since the CONVENIENT *t-option* was specified, the procedure chooses a log transformation. The $F = t^2$ plot shows in the vicinity of the optimal Box-Cox transformation, the parameters for the three original variables (Length Amplitude Load), particularly Length, are significant and the others become essentially zero.

Example 126.3: Penalized B-Spline

(View the complete code for this example (tregex3.sas) in the example repository.)

The ENSO data set contains measurements of monthly averaged atmospheric pressure differences between Easter Island and Darwin, Australia, for a period of 168 months (National Institute of Standards and Technology 1998). The ENSO data set is available from the Sashelp library.

You can fit a curve through these data by using a penalized B-spline (Eilers and Marx 1996) function and the following statements:

```
title 'Atmospheric Pressure Changes Between'
      ' Easter Island & Darwin, Australia';
ods graphics on;

proc transreg data=sashelp.enso;
   model identity(pressure) = pbspline(year);
run;
```

The dependent variable Pressure is specified along with an IDENTITY transformation, so Pressure is analyzed as is, with no transformations. The independent variable Year is specified with a PBSPLINE transformation, so a penalized B-spline model is fit. By default, a DEGREE=3 B-spline basis is used along with 100 evenly spaced knots and three evenly spaced exterior knots on each side of the data. The penalized spline function is typically much smoother than you would get by using a SPLINE transformation or a BSPLINE expansion since changes in the coefficients of the basis are penalized to make a smoother fit. The output is shown next in Output 126.3.1.

Output 126.3.1 Change in Atmospheric Pressure, AICC

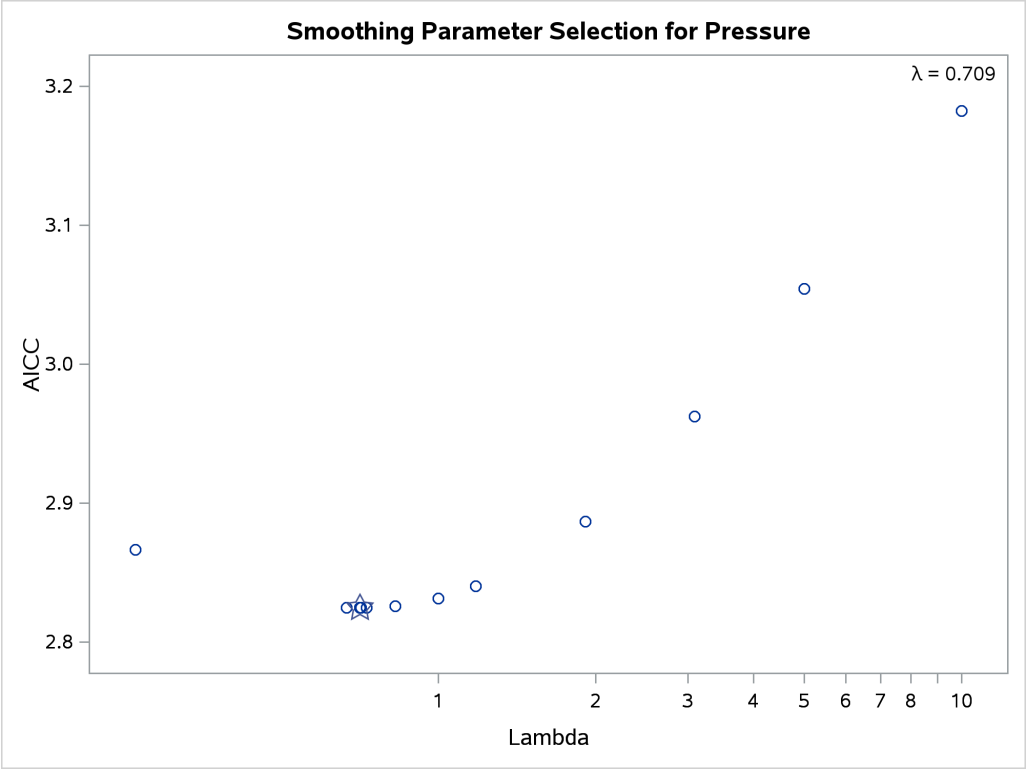

Output 126.3.1 *continued*

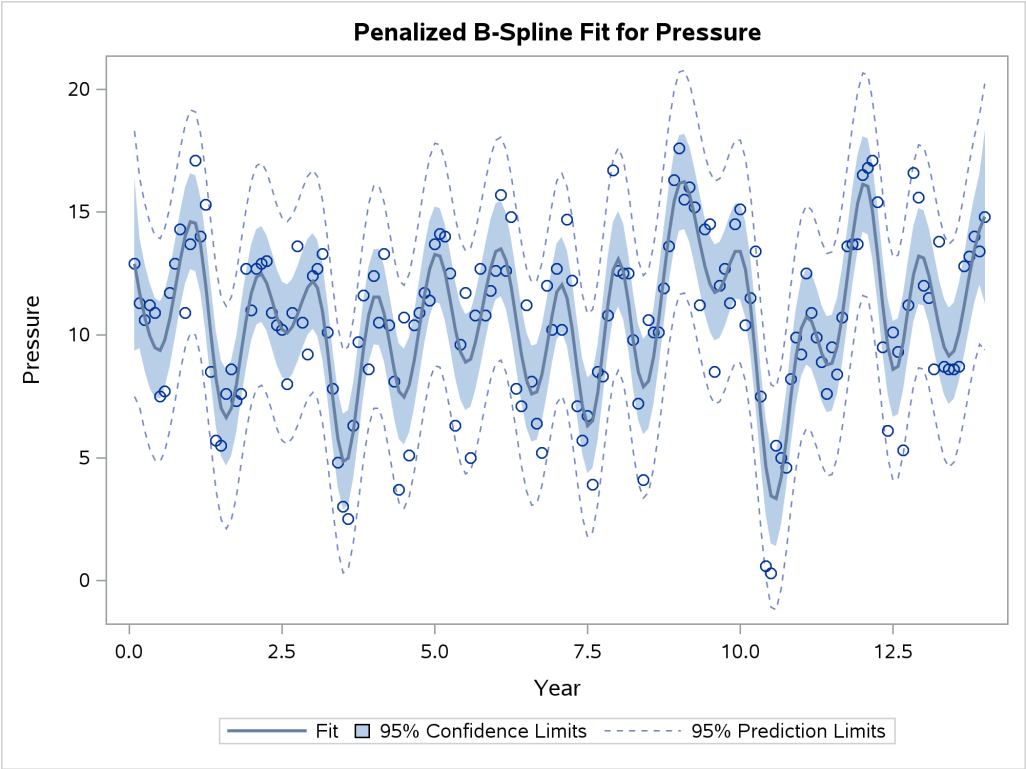

The results show a yearly cycle of pressure change. The procedure chose a smoothing parameter of $\lambda = 0.709$. With data such as these, with many peaks and valleys, it might be useful to perform another analysis, this time asking for a smoother plot. The Schwarz Bayesian criterion (SBC) is sometimes a better choice than the default criterion when you want a smoother plot. The following PROC TRANSREG step requests a penalized B-spline analysis minimizing the SBC criterion:

```
proc transreg data=sashelp.enso;
   model identity(pressure) = pbspline(year / sbc);
run;
```

The plot of SBC as a function of λ is shown in Output 126.3.2.

Output 126.3.2 Change in Atmospheric Pressure, SBC

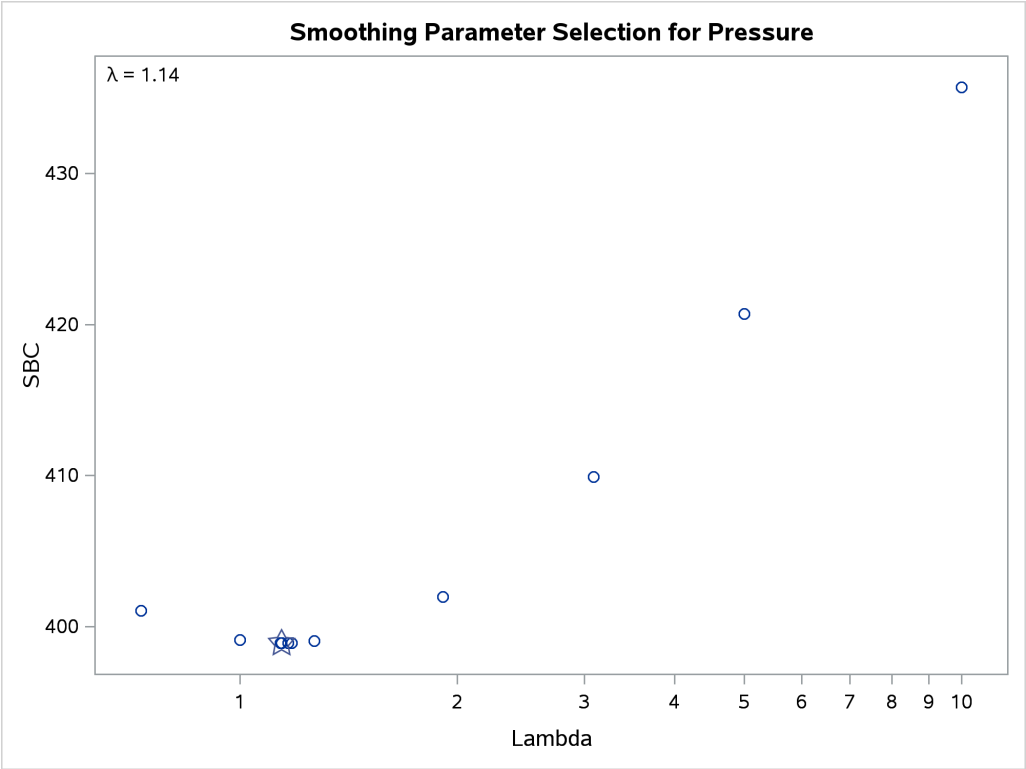

The fit plot (not shown) is essentially the same as the one shown in Output 126.3.1 due to the similar choice of smoothing parameters: $\lambda = 0.709$ versus $\lambda = 1.14$. You can analyze these data again, this time forcing PROC TRANSREG to consider only larger smoothing parameters. The specification LAMBDA=2 10000 RANGE eliminates from consideration the two lambdas that you previously saw and considers only $2 \leq \lambda \leq 10{,}000$. The following statements produce Output 126.3.3:

```
proc transreg data=sashelp.enso;
   model identity(pressure) = pbspline(year / sbc lambda=2 10000 range);
run;
```

Output 126.3.3 Change in Atmospheric Pressure, SBC, Lambda > 1

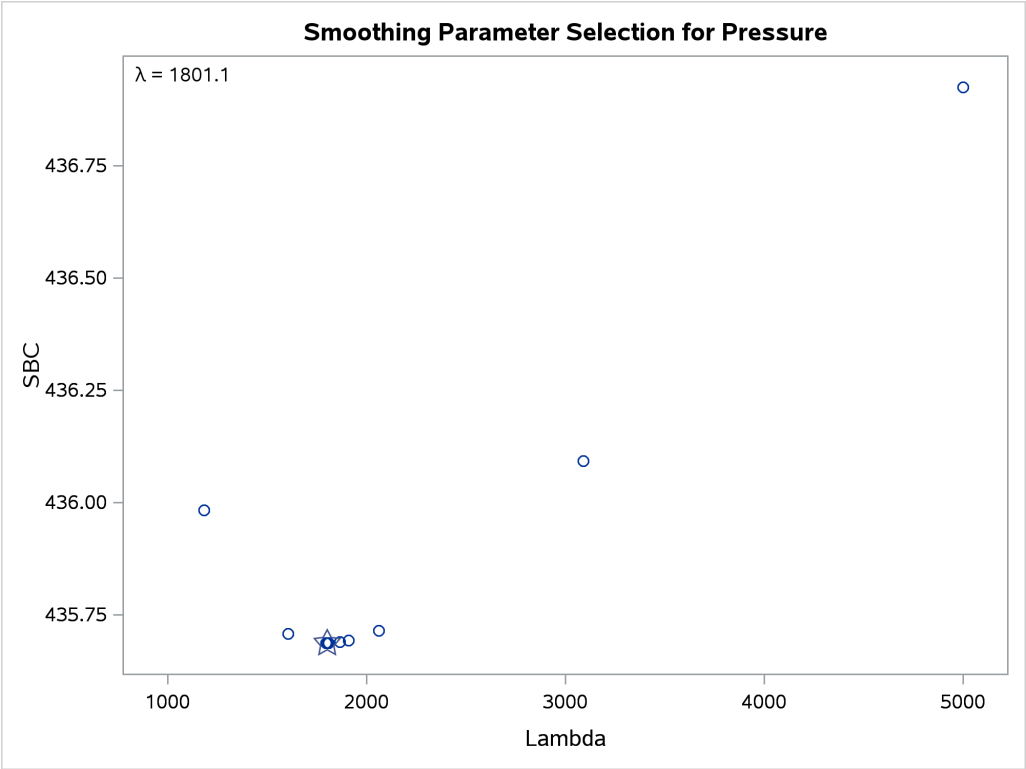

Output 126.3.3 *continued*

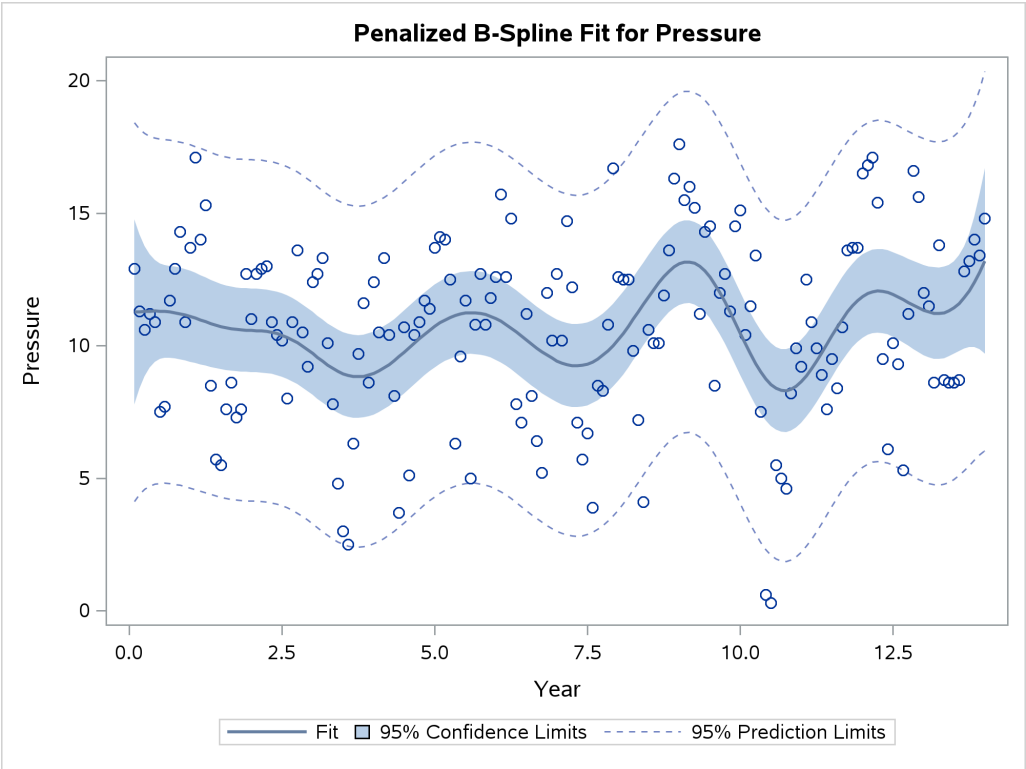

The results clearly show that there is a local minimum in the SBC(λ) function at $\lambda = 1801.1$. Using this lambda results in a much smoother regression function with a longer cycle than you saw previously. This second cycle can be identified as the periodic warming of the Pacific Ocean known as "El Niño." The SBC(λ) function has at least two minima since there are at least two trends in the data. In the first analysis, PROC TRANSREG found what is probably the globally optimal solution, and in the second set of analyses, with a little nudging away from the global optimum, it found a very interesting locally optimal solution.

You can specify a list of lambdas to see SBC as a function of lambda over the range that includes both minima as follows:

```
proc transreg data=sashelp.enso;
   model identity(pressure) = pbspline(year / sbc lambda=.1 .5 1 5
                                        10 50 100 500 to 2500 by 500);
run;
```

The plot of SBC as a function of λ is shown in Output 126.3.4.

Output 126.3.4 Change in Atmospheric Pressure, SBC, Over the Range of Both Minima

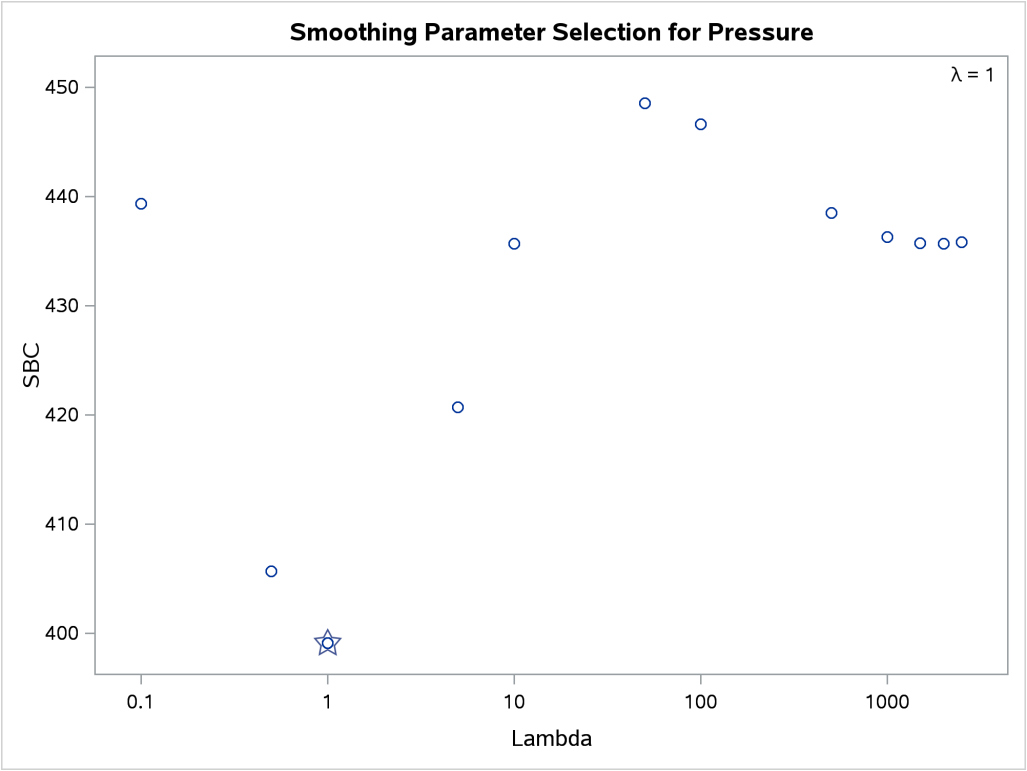

Example 126.4: Nonmetric Conjoint Analysis of Tire Data

(View the complete code for this example (tregex4.sas) in the example repository.)

This example uses PROC TRANSREG to perform a nonmetric conjoint analysis of tire preference data. Conjoint analysis decomposes rank-ordered evaluation judgments of products or services into components based on qualitative product attributes. For each level of each attribute of interest, a numerical "part-worth utility" value is computed. The sum of the part-worth utilities for each product is an estimate of the utility for that product. The goal is to compute part-worth utilities such that the product utilities are as similar as possible to the original rank ordering. (This example is a greatly simplified introductory example.)

The stimuli for the experiment are 18 hypothetical tires. The stimuli represent different brands (Goodstone, Pirogi, Machismo),[1] prices ($69.99, $74.99, $79.99), expected tread life (50,000, 60,000, 70,000 miles), and road hazard insurance plans (Yes, No). There are $3 \times 3 \times 3 \times 2 = 54$ possible combinations. From these, 18 combinations are selected that form an efficient experimental design for a main-effects model. The combinations are then ranked from 1 (most preferred) to 18 (least preferred). In this simple example, there is one set of rankings. A real conjoint study would have many more.

First, the FORMAT procedure is used to specify the meanings of the factor levels, which are entered as numbers in the DATA step along with the ranks. PROC TRANSREG is used to perform the conjoint analysis. A maximum of 50 iterations is requested. The specification **monotone(Rank / reflect)** in the MODEL statement requests that the dependent variable Rank should be monotonically transformed and reflected so that positive utilities mean high preference. The variables Brand, Price, Life, and Hazard are designated as CLASS variables, and the part-worth utilities are constrained by ZERO=SUM to sum to zero within each factor. The UTILITIES *a-option* displays the conjoint analysis results.

The importance column of the utilities table shows that price is the most important attribute in determining preference (57%), followed by expected tread life (18%), brand (15%), and road hazard insurance (10%). Looking at the utilities table for the maximum part-worth utility within each attribute, you see from the results that the most preferred combination is Pirogi brand tires, at $69.99, with a 70,000-mile expected tread life and road hazard insurance. This product is not actually in the data set. The sum of the part-worth utilities for this combination is as follows:

$$20.64 = 9.50 + 1.90 + 5.87 + 2.41 + 0.96$$

The following statements produce Output 126.4.1.

```
title 'Nonmetric Conjoint Analysis of Ranks';

proc format;
   value BrandF
              1 = 'Goodstone'
              2 = 'Pirogi   '
              3 = 'Machismo ';
   value PriceF
              1 = '$69.99'
              2 = '$74.99'
```

[1] In real conjoint experiments, real brand names would be used.

```
              3 = '$79.99';
   value LifeF
              1 = '50,000'
              2 = '60,000'
              3 = '70,000';
   value HazardF
              1 = 'Yes'
              2 = 'No ';
run;

data Tires;
   input Brand Price Life Hazard Rank;
   format Brand BrandF9. Price PriceF9. Life LifeF6. Hazard HazardF3.;
   datalines;
1 1 2 1  3
1 1 3 2  2
1 2 1 2 14
1 2 2 2 10
1 3 1 1 17
1 3 3 1 12
2 1 1 2  7
2 1 3 2  1
2 2 1 1  8
2 2 3 1  5
2 3 2 1 13
2 3 2 2 16
3 1 1 1  6
3 1 2 1  4
3 2 2 2 15
3 2 3 1  9
3 3 1 2 18
3 3 3 2 11
;

proc transreg maxiter=50 utilities short;
   ods select TestsNote ConvergenceStatus FitStatistics Utilities;
   model monotone(Rank / reflect) =
         class(Brand Price Life Hazard / zero=sum);
   output ireplace predicted;
run;

proc print label;
   var Rank TRank PRank Brand Price Life Hazard;
   label PRank = 'Predicted Ranks';
run;
```

Output 126.4.1 Simple Conjoint Analysis

Nonmetric Conjoint Analysis of Ranks

The TRANSREG Procedure

Monotone(Rank)
Algorithm converged.

Output 126.4.1 *continued*

The TRANSREG Procedure Hypothesis Tests for Monotone(Rank)

Root MSE	0.49759	**R-Square**	0.9949
Dependent Mean	9.50000	**Adj R-Sq**	0.9913
Coeff Var	5.23783		

Utilities Table Based on the Usual Degrees of Freedom

Label	Utility	Standard Error	Importance (% Utility Range)	Variable
Intercept	9.5000	0.11728		Intercept
Brand Goodstone	-1.1718	0.16586	15.463	Class.BrandGoodstone
Brand Pirogi	1.8980	0.16586		Class.BrandPirogi
Brand Machismo	-0.7262	0.16586		Class.BrandMachismo
Price $69.99	5.8732	0.16586	56.517	Class.Price_69_99
Price $74.99	-0.5261	0.16586		Class.Price_74_99
Price $79.99	-5.3471	0.16586		Class.Price_79_99
Life 50,000	-1.2350	0.16586	18.361	Class.Life50_000
Life 60,000	-1.1751	0.16586		Class.Life60_000
Life 70,000	2.4101	0.16586		Class.Life70_000
Hazard Yes	0.9588	0.11728	9.659	Class.HazardYes
Hazard No	-0.9588	0.11728		Class.HazardNo

The standard errors are not adjusted for the fact that the dependent variable was transformed and so are generally liberal (too small).

Output 126.4.1 *continued*

Nonmetric Conjoint Analysis of Ranks

Obs	Rank	Rank Transformation	Predicted Ranks	Brand	Price	Life	Hazard
1	3	14.4462	13.9851	Goodstone	$69.99	60,000	Yes
2	2	15.6844	15.6527	Goodstone	$69.99	70,000	No
3	14	5.7229	5.6083	Goodstone	$74.99	50,000	No
4	10	5.7229	5.6682	Goodstone	$74.99	60,000	No
5	17	2.6699	2.7049	Goodstone	$79.99	50,000	Yes
6	12	5.7229	6.3500	Goodstone	$79.99	70,000	Yes
7	7	14.4462	15.0774	Pirogi	$69.99	50,000	No
8	1	18.7699	18.7225	Pirogi	$69.99	70,000	No
9	8	11.1143	10.5957	Pirogi	$74.99	50,000	Yes
10	5	14.4462	14.2408	Pirogi	$74.99	70,000	Yes
11	13	5.7229	5.8346	Pirogi	$79.99	60,000	Yes
12	16	3.8884	3.9170	Pirogi	$79.99	60,000	No
13	6	14.4462	14.3708	Machismo	$69.99	50,000	Yes
14	4	14.4462	14.4307	Machismo	$69.99	60,000	Yes
15	15	5.7229	6.1139	Machismo	$74.99	60,000	No
16	9	11.1143	11.6166	Machismo	$74.99	70,000	Yes
17	18	1.1905	1.2330	Machismo	$79.99	50,000	No
18	11	5.7229	4.8780	Machismo	$79.99	70,000	No

Example 126.5: Metric Conjoint Analysis of Tire Data

(View the complete code for this example (tregex5.sas) in the example repository.)

This example, which is more detailed than the previous one, uses PROC TRANSREG to perform a metric conjoint analysis of tire preference data. Conjoint analysis can be used to decompose preference ratings of products or services into components based on qualitative product attributes. For each level of each attribute of interest, a numerical "part-worth utility" value is computed. The sum of the part-worth utilities for each product is an estimate of the utility for that product. The goal is to compute part-worth utilities such that the product utilities are as similar as possible to the original ratings. Metric conjoint analysis, as shown in this example, fits an ordinary linear model directly to data assumed to be measured on an interval scale. Nonmetric conjoint analysis, as shown in Example 126.4, finds an optimal monotonic transformation of original data before fitting an ordinary linear model to the transformed data.

This example has three parts. In the first part, an experimental design is created. In the second part, a DATA step creates descriptions of the stimuli for the experiment. The third part of the example performs the conjoint analyses.

The stimuli for the experiment are 18 hypothetical tires. The stimuli represent different brands (Goodstone, Pirogi, Machismo),[2] prices ($69.99, $74.99, $79.99), expected tread life (50,000, 60,000, 70,000 miles), and road hazard insurance plans (Yes, No).

For a conjoint study such as this, you need to create an experimental design with 3 three-level factors, 1 two-level factor, and 18 combinations or *runs*. The easiest way to get this design is with the %MktEx autocall macro. The %MktEx macro requires you to specify the number of levels of each of the four factors, followed by N=18, the number of runs. Specifying a random number seed, while not strictly necessary, helps ensure that the design is reproducible. The %MktLab macro assigns the actual factor names instead of the default names x1, x2, and so on, and it assigns formats to the factor levels. The %MktEval macro helps you evaluate the design. It shows how correlated or independent the factors are, how often each factor level appears in the design, how often each pair occurs for every factor pair, and how often each product profile or run occurs in the design. See Kuhfeld (2010) for more information about experimental design and conjoint analysis; look for the latest "Conjoint Analysis" report. The following statements create, evaluate, and display the design:

```
title 'Tire Study, Experimental Design';

proc format;
   value BrandF
              1 = 'Goodstone'
              2 = 'Pirogi   '
              3 = 'Machismo ';
   value PriceF
              1 = '$69.99'
              2 = '$74.99'
              3 = '$79.99';
   value LifeF
              1 = '50,000'
              2 = '60,000'
              3 = '70,000';
   value HazardF
```

[2] In real conjoint experiments, real brand names would be used.

```
                1 = 'Yes'
                2 = 'No ';
run;

%mktex(3 3 3 2, n=18, seed=448)

%mktlab(vars=Brand Price Life Hazard, out=sasuser.TireDesign,
        statements=format Brand BrandF9. Price PriceF9.
                   Life LifeF6. Hazard HazardF3.)

%mkteval;

proc print data=sasuser.TireDesign;
run;
```

The %MktEx macro (Kuhfeld 2010) output displayed in Output 126.5.1 shows you that the design is 100% efficient, which means it is orthogonal and balanced. The %MktEval macro output displayed in Output 126.5.2 shows you that all of the factors are uncorrelated or orthogonal, the design is balanced (each level occurs once), and every pair of factor levels occurs equally often (again showing that the design is orthogonal). The *n*-way frequencies show that each product profile occurs once (there are no duplicates). The design is shown in Output 126.5.3. The design is automatically randomized (the profiles were sorted into a random order and the original levels are randomly reassigned). Orthogonality, balance, randomization, and other design concepts are discussed in detail in Kuhfeld (2010), in the "Experimental Design, Efficiency, Coding, and Choice Designs" report.

Output 126.5.1 Tire Study, Design Efficiency

Tire Study, Experimental Design

Algorithm Search History

Design	Row,Col	Current D-Efficiency	Best D-Efficiency	Notes
1	Start	100.0000	100.0000	Tab
1	End	100.0000		

Tire Study, Experimental Design

The OPTEX Procedure

Class Level Information

Class	Levels	Values
x1	3	1 2 3
x2	3	1 2 3
x3	3	1 2 3
x4	2	1 2

Output 126.5.1 *continued*

Tire Study, Experimental Design

Design Number	D-Efficiency	A-Efficiency	G-Efficiency	Average Prediction Standard Error
1	100.0000	100.0000	100.0000	0.6667

Output 126.5.2 Tire Study, Design Evaluation

Tire Study, Experimental Design
Canonical Correlations Between the Factors
There are 0 Canonical Correlations Greater Than 0.316

	Brand	Price	Life	Hazard
Brand	1	0	0	0
Price	0	1	0	0
Life	0	0	1	0
Hazard	0	0	0	1

Tire Study, Experimental Design
Summary of Frequencies
There are 0 Canonical Correlations Greater Than 0.316

	Frequencies
Brand	6 6 6
Price	6 6 6
Life	6 6 6
Hazard	9 9
Brand Price	2 2 2 2 2 2 2 2 2
Brand Life	2 2 2 2 2 2 2 2 2
Brand Hazard	3 3 3 3 3 3
Price Life	2 2 2 2 2 2 2 2 2
Price Hazard	3 3 3 3 3 3
Life Hazard	3 3 3 3 3 3
N-Way	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Output 126.5.3 Tire Study, Design

Tire Study, Experimental Design

Obs	Brand	Price	Life	Hazard
1	Pirogi	$79.99	50,000	No
2	Machismo	$79.99	60,000	No
3	Machismo	$74.99	70,000	Yes
4	Machismo	$74.99	50,000	No
5	Goodstone	$74.99	60,000	Yes
6	Pirogi	$69.99	60,000	Yes
7	Goodstone	$69.99	50,000	Yes
8	Machismo	$69.99	50,000	Yes
9	Pirogi	$74.99	60,000	Yes
10	Pirogi	$74.99	50,000	No
11	Goodstone	$79.99	60,000	No
12	Goodstone	$69.99	70,000	No
13	Pirogi	$79.99	70,000	Yes
14	Goodstone	$74.99	70,000	No
15	Machismo	$69.99	60,000	No
16	Machismo	$79.99	70,000	Yes
17	Pirogi	$69.99	70,000	No
18	Goodstone	$79.99	50,000	Yes

The %MktEx macro requires SAS/STAT, SAS/QC, and SAS/IML software. Alternatively, you can make a design for this experiment using the %MktDes macro, which requires only SAS/STAT and SAS/QC software. The %MktDes macro contains a small subset of the functionality of the %MktEx macro. It can be used as follows:

```
%mktdes(factors=Brand=3 Price=3 Life=3 Hazard=2, n=18)
```

The results of this step are not shown or used.

Next, the questionnaires are printed and given to the subjects, who are asked to rate the tires.

The following statements produce Output 126.5.4:

```
data _null_;
   title;
   set sasuser.TireDesign;
   file print;
   if mod(_n_,4) eq 1 then do;
      put _page_;
      put +55 'Subject ________';
   end;
   length hazardstring $ 7.;
   if put(hazard, hazardf3.) = 'Yes'
      then hazardstring = 'with';
      else hazardstring = 'without';

   s = 3 + (_n_ >= 10);
   put // _n_ +(-1) ') For your next tire purchase, '
          'how likely are you to buy this product?'
```

```
        // +s Brand 'brand tires at ' Price +(-1) ','
        /  +s 'with a ' Life 'tread life guarantee, '
        /  +s 'and ' hazardstring 'road hazard insurance.'
        // +s 'Definitely Would                 Definitely Would'
        /  +s 'Not Purchase                            Purchase'
        // +s '1     2     3     4     5     6     7     8     9 ';
  run;
```

This output in Output 126.5.4 is abbreviated in the interest of conserving space; the statements actually produce stimuli for all combinations.

Output 126.5.4 Conjoint Analysis, Stimuli Descriptions

```
                                                    Subject _________

1) For your next tire purchase, how likely are you to buy this product?

   Pirogi brand tires at $79.99,
   with a 50,000 tread life guarantee,
   and without road hazard insurance.

   Definitely Would                 Definitely Would
   Not Purchase                             Purchase

   1     2     3     4     5     6     7     8     9


2) For your next tire purchase, how likely are you to buy this product?

   Machismo brand tires at $79.99,
   with a 60,000 tread life guarantee,
   and without road hazard insurance.

   Definitely Would                 Definitely Would
   Not Purchase                             Purchase

   1     2     3     4     5     6     7     8     9


3) For your next tire purchase, how likely are you to buy this product?

   Machismo brand tires at $74.99,
   with a 70,000 tread life guarantee,
   and with road hazard insurance.

   Definitely Would                 Definitely Would
   Not Purchase                             Purchase

   1     2     3     4     5     6     7     8     9


4) For your next tire purchase, how likely are you to buy this product?

   Machismo brand tires at $74.99,
   with a 50,000 tread life guarantee,
   and without road hazard insurance.

   Definitely Would                 Definitely Would
   Not Purchase                             Purchase

   1     2     3     4     5     6     7     8     9
```

The third part of the example performs the conjoint analyses. The DATA step reads the data. Only the ratings are entered, one row per subject. Real conjoint studies have many more subjects than five. The TRANSPOSE procedure transposes this (5×18) data set into an (18×5) data set that can be merged with the factor level

data set sasuser.TireDesign. The next DATA step does the merge. The PRINT procedure displays the input data set.

PROC TRANSREG fits the five individual conjoint models, one for each subject. The UTILITIES *a-option* displays the conjoint analysis results. The SHORT *a-option* suppresses the iteration histories, OUTTEST=UTILS creates an output data set with all of the conjoint results, and the SEPARATORS= option requests that the labels constructed for each category contain two blanks between the variable name and the level value. The ODS SELECT statement is used to limit the displayed output. The MODEL statement specifies IDENTITY for the ratings, which specifies a metric conjoint analysis—the ratings are not transformed. The variables Brand, Price, Life, and Hazard are designated as CLASS variables, and the part-worth utilities are constrained to sum to zero within each factor.

The following statements produce Output 126.5.5:

```
title 'Tire Study, Data Entry, Preprocessing';

data Results;
   input (c1-c18) (1.);
   datalines;
233279766526376493
124467885349168274
262189456534275794
184396375364187754
133379775526267493
;

* Create an Object by Subject Data Matrix;
proc transpose data=Results out=Results(drop=_name_) prefix=Subj;
run;

* Merge the Factor Levels with the Data Matrix;
data Both;
   merge sasuser.TireDesign Results;
run;

proc print;
   title2 'Data Set for Conjoint Analysis';
run;

title 'Tire Study, Individual Conjoint Analyses';

* Fit Each Subject Individually;
proc transreg data=Both utilities short outtest=utils separators='  ';
   ods select TestsNote FitStatistics Utilities;
   model identity(Subj1-Subj5) =
         class(Brand Price Life Hazard / zero=sum);
run;
```

The output contains two tables per subject, one with overall fit statistics and one with the conjoint analysis results.

Output 126.5.5 Conjoint Analysis

Tire Study, Data Entry, Preprocessing Data Set for Conjoint Analysis

Obs	Brand	Price	Life	Hazard	Subj1	Subj2	Subj3	Subj4	Subj5
1	Pirogi	$79.99	50,000	No	2	1	2	1	1
2	Machismo	$79.99	60,000	No	3	2	6	8	3
3	Machismo	$74.99	70,000	Yes	3	4	2	4	3
4	Machismo	$74.99	50,000	No	2	4	1	3	3
5	Goodstone	$74.99	60,000	Yes	7	6	8	9	7
6	Pirogi	$69.99	60,000	Yes	9	7	9	6	9
7	Goodstone	$69.99	50,000	Yes	7	8	4	3	7
8	Machismo	$69.99	50,000	Yes	6	8	5	7	7
9	Pirogi	$74.99	60,000	Yes	6	5	6	5	5
10	Pirogi	$74.99	50,000	No	5	3	5	3	5
11	Goodstone	$79.99	60,000	No	2	4	3	6	2
12	Goodstone	$69.99	70,000	No	6	9	4	4	6
13	Pirogi	$79.99	70,000	Yes	3	1	2	1	2
14	Goodstone	$74.99	70,000	No	7	6	7	8	6
15	Machismo	$69.99	60,000	No	6	8	5	7	7
16	Machismo	$79.99	70,000	Yes	4	2	7	7	4
17	Pirogi	$69.99	70,000	No	9	7	9	5	9
18	Goodstone	$79.99	50,000	Yes	3	4	4	4	3

Output 126.5.5 *continued*

Tire Study, Individual Conjoint Analyses

The TRANSREG Procedure

The TRANSREG Procedure Hypothesis Tests for Identity(Subj1)

Root MSE	1.34164	**R-Square**	0.8043
Dependent Mean	5.00000	**Adj R-Sq**	0.6674
Coeff Var	26.83282		

Output 126.5.5 *continued*

Utilities Table Based on the Usual Degrees of Freedom				
Label	**Utility**	**Standard Error**	**Importance (% Utility Range)**	**Variable**
Intercept	5.0000	0.31623		Intercept
Brand Goodstone	0.3333	0.44721	20.833	Class.BrandGoodstone
Brand Pirogi	0.6667	0.44721		Class.BrandPirogi
Brand Machismo	-1.0000	0.44721		Class.BrandMachismo
Price $69.99	2.1667	0.44721	54.167	Class.Price_69_99
Price $74.99	0.0000	0.44721		Class.Price_74_99
Price $79.99	-2.1667	0.44721		Class.Price_79_99
Life 50,000	-0.8333	0.44721	16.667	Class.Life50_000
Life 60,000	0.5000	0.44721		Class.Life60_000
Life 70,000	0.3333	0.44721		Class.Life70_000
Hazard Yes	0.3333	0.31623	8.333	Class.HazardYes
Hazard No	-0.3333	0.31623		Class.HazardNo

Tire Study, Individual Conjoint Analyses

The TRANSREG Procedure

The TRANSREG Procedure Hypothesis Tests for Identity(Subj2)

Root MSE	0.56765	**R-Square**	0.9710
Dependent Mean	4.94444	**Adj R-Sq**	0.9506
Coeff Var	11.48049		

Utilities Table Based on the Usual Degrees of Freedom				
Label	**Utility**	**Standard Error**	**Importance (% Utility Range)**	**Variable**
Intercept	4.9444	0.13380		Intercept
Brand Goodstone	1.2222	0.18922	25.658	Class.BrandGoodstone
Brand Pirogi	-0.9444	0.18922		Class.BrandPirogi
Brand Machismo	-0.2778	0.18922		Class.BrandMachismo
Price $69.99	2.8889	0.18922	65.132	Class.Price_69_99
Price $74.99	-0.2778	0.18922		Class.Price_74_99
Price $79.99	-2.6111	0.18922		Class.Price_79_99
Life 50,000	-0.2778	0.18922	7.895	Class.Life50_000
Life 60,000	0.3889	0.18922		Class.Life60_000
Life 70,000	-0.1111	0.18922		Class.Life70_000
Hazard Yes	0.0556	0.13380	1.316	Class.HazardYes
Hazard No	-0.0556	0.13380		Class.HazardNo

Output 126.5.5 *continued*

Tire Study, Individual Conjoint Analyses

The TRANSREG Procedure

The TRANSREG Procedure Hypothesis Tests for Identity(Subj3)

Root MSE	2.48104	**R-Square**	0.3902
Dependent Mean	4.94444	**Adj R-Sq**	-0.0367
Coeff Var	50.17832		

Utilities Table Based on the Usual Degrees of Freedom

Label	Utility	Standard Error	Importance (% Utility Range)	Variable
Intercept	4.9444	0.58479		Intercept
Brand Goodstone	0.0556	0.82701	18.261	Class.BrandGoodstone
Brand Pirogi	0.5556	0.82701		Class.BrandPirogi
Brand Machismo	-0.6111	0.82701		Class.BrandMachismo
Price $69.99	1.0556	0.82701	31.304	Class.Price_69_99
Price $74.99	-0.1111	0.82701		Class.Price_74_99
Price $79.99	-0.9444	0.82701		Class.Price_79_99
Life 50,000	-1.4444	0.82701	41.739	Class.Life50_000
Life 60,000	1.2222	0.82701		Class.Life60_000
Life 70,000	0.2222	0.82701		Class.Life70_000
Hazard Yes	0.2778	0.58479	8.696	Class.HazardYes
Hazard No	-0.2778	0.58479		Class.HazardNo

Tire Study, Individual Conjoint Analyses

The TRANSREG Procedure

The TRANSREG Procedure Hypothesis Tests for Identity(Subj4)

Root MSE	1.90321	**R-Square**	0.6185
Dependent Mean	5.05556	**Adj R-Sq**	0.3514
Coeff Var	37.64598		

Output 126.5.5 *continued*

Utilities Table Based on the Usual Degrees of Freedom

Label	Utility	Standard Error	Importance (% Utility Range)	Variable
Intercept	5.0556	0.44859		Intercept
Brand Goodstone	0.6111	0.63440	36.885	Class.BrandGoodstone
Brand Pirogi	-1.5556	0.63440		Class.BrandPirogi
Brand Machismo	0.9444	0.63440		Class.BrandMachismo
Price $69.99	0.2778	0.63440	12.295	Class.Price_69_99
Price $74.99	0.2778	0.63440		Class.Price_74_99
Price $79.99	-0.5556	0.63440		Class.Price_79_99
Life 50,000	-1.5556	0.63440	49.180	Class.Life50_000
Life 60,000	1.7778	0.63440		Class.Life60_000
Life 70,000	-0.2222	0.63440		Class.Life70_000
Hazard Yes	0.0556	0.44859	1.639	Class.HazardYes
Hazard No	-0.0556	0.44859		Class.HazardNo

Tire Study, Individual Conjoint Analyses

The TRANSREG Procedure

The TRANSREG Procedure Hypothesis Tests for Identity(Subj5)

Root MSE	1.36219	**R-Square**	0.8162
Dependent Mean	4.94444	**Adj R-Sq**	0.6875
Coeff Var	27.54987		

Utilities Table Based on the Usual Degrees of Freedom

Label	Utility	Standard Error	Importance (% Utility Range)	Variable
Intercept	4.9444	0.32107		Intercept
Brand Goodstone	0.2222	0.45406	9.023	Class.BrandGoodstone
Brand Pirogi	0.2222	0.45406		Class.BrandPirogi
Brand Machismo	-0.4444	0.45406		Class.BrandMachismo
Price $69.99	2.5556	0.45406	67.669	Class.Price_69_99
Price $74.99	-0.1111	0.45406		Class.Price_74_99
Price $79.99	-2.4444	0.45406		Class.Price_79_99
Life 50,000	-0.6111	0.45406	15.789	Class.Life50_000
Life 60,000	0.5556	0.45406		Class.Life60_000
Life 70,000	0.0556	0.45406		Class.Life70_000
Hazard Yes	0.2778	0.32107	7.519	Class.HazardYes
Hazard No	-0.2778	0.32107		Class.HazardNo

The next steps summarize the results. Three tables are displayed, showing the following: all of the importance values, the average importance, and the part-worth utilities. The first DATA step selects the importance information from the UTILS data set. The final assignment statement stores just the variable name from the label, relying on the fact that the separator is two blanks. PROC TRANSPOSE creates the data set of

importances, one row per subject, and PROC PRINT displays the results. The MEANS procedure displays the average importance of each attribute across the subjects. The next DATA step selects the part-worth utilities information from the UTILS data set. PROC TRANSPOSE creates the data set of utilities, one row per subject, and PROC PRINT displays the results. The following statements produce Output 126.5.6:

```
title 'Tire Study Results';

* Gather the Importance Values;
data Importance;
   set utils(keep=_depvar_ Importance Label);
   if n(Importance);
   label = substr(label, 1, index(label, '  '));
run;

proc transpose out=Importance2(drop=_:);
   by _depvar_;
   id Label;
run;

proc print;
   title2 'Importance Values';
run;

proc means;
   title2 'Average Importance';
run;

* Gather the Part-Worth Utilities;
data Utilities;
   set utils(keep=_depvar_ Coefficient Label);
   if n(Coefficient);
run;

proc transpose out=Utilities2(drop=_:);
   by _depvar_;
   id Label;
   idlabel Label;
run;

proc print label;
   title2 'Utilities';
run;
```

Output 126.5.6 Summary of Conjoint Analysis Results

Tire Study Results
Importance Values

Obs	Brand	Price	Life	Hazard
1	20.8333	54.1667	16.6667	8.33333
2	25.6579	65.1316	7.8947	1.31579
3	18.2609	31.3043	41.7391	8.69565
4	36.8852	12.2951	49.1803	1.63934
5	9.0226	67.6692	15.7895	7.51880

Output 126.5.6 *continued*

Tire Study Results
Average Importance

The MEANS Procedure

Variable	N	Mean	Std Dev	Minimum	Maximum
Brand	5	22.1319800	10.2301014	9.0225564	36.8852459
Price	5	46.1133697	23.7391251	12.2950820	67.6691729
Life	5	26.2540671	18.0547195	7.8947368	49.1803279
Hazard	5	5.5005832	3.6989117	1.3157895	8.6956522

Output 126.5.6 *continued*

Tire Study Results
Utilities

Obs	Intercept	Brand Goodstone	Brand Pirogi	Brand Machismo	Price $69.99	Price $74.99	Price $79.99	Life 50,000	Life 60,000	Life 70,000	Hazard Yes	Hazard No
1	5.00000	0.33333	0.66667	-1.00000	2.16667	0.00000	-2.16667	-0.83333	0.50000	0.33333	0.33333	-0.33333
2	4.94444	1.22222	-0.94444	-0.27778	2.88889	-0.27778	-2.61111	-0.27778	0.38889	-0.11111	0.05556	-0.05556
3	4.94444	0.05556	0.55556	-0.61111	1.05556	-0.11111	-0.94444	-1.44444	1.22222	0.22222	0.27778	-0.27778
4	5.05556	0.61111	-1.55556	0.94444	0.27778	0.27778	-0.55556	-1.55556	1.77778	-0.22222	0.05556	-0.05556
5	4.94444	0.22222	0.22222	-0.44444	2.55556	-0.11111	-2.44444	-0.61111	0.55556	0.05556	0.27778	-0.27778

Based on the importance values, price is the most important attribute for some of the respondents, but expected tread life is most important for others. On the average, price is most important, followed by expected tread life and brand. Road hazard insurance is less important. Each of the brands is preferred by some of the respondents. All respondents preferred a lower price over a higher price, a longer tread life, and road hazard insurance.

Example 126.6: Preference Mapping of Automobile Data

(View the complete code for this example (tregex6.sas) in the example repository.)

This example uses PROC TRANSREG to perform a preference mapping (PREFMAP) analysis (Carroll 1972) of automobile preference data after a PROC PRINQUAL principal component analysis. The PREFMAP analysis is a response surface regression that locates ideal points for each dependent variable in a space defined by the independent variables.

The data are ratings obtained from 25 judges of their preference for each of 17 automobiles. The ratings were made on a scale of zero (very weak preference) to nine (very strong preference). These judgments were made in 1980 about that year's products. There are two character variables that indicate the manufacturer and model of the automobile. The data set also contains three ratings: miles per gallon (MPG), projected reliability (Reliability), and quality of the ride (Ride). These ratings are on a scale of one (bad) to five (good). PROC PRINQUAL creates an OUT= data set containing standardized principal component scores (Prin1 and Prin2), along with the ID variables Model, MPG, Reliability, and Ride.

While this data set contains all of the information needed for the subsequent preference mapping, you can make slightly more informative plots by adding new variable labels to the principal component score variables. The default labels are 'Component 1', 'Component 2', and so on. These are by necessity rather generic since they are created before any data are read, and they must be appropriate across BY groups when a BY variable is specified. In contrast, the MDPREF plot in PROC PRINQUAL has axis labels of the form 'Component 1 (43.54%)' and 'Component 2 (23.4%)' that show the proportion of variance accounted for by each component. You can create an output data set from the MDPREF plot by using the ODS OUTPUT statement and then use only the label information from it to reset the labels in the output data set from PROC PRINQUAL. In the DATA PLOT step, the SET statement for the MD data set is specified before the SET statement for the PRESULTS data set. The **if 0** ensures that no data are actually read from it, but nevertheless the properties of the Prin1 and Prin2 variables including the variable labels are set based on the properties of those variables in the MD data set.

The first PROC TRANSREG step fits univariate regression models for MPG and Reliability. All variables are designated IDENTITY. A vector drawn in the plot of Prin1 and Prin2 from the origin to the point defined by an attribute's regression coefficients approximately shows how the autos differ on that attribute. See Carroll (1972) for more information. The Prin1 and Prin2 columns of the TResult1 OUT= data set contain the automobile coordinates (_Type_='SCORE' observations) and endpoints of the MPG and Reliability vectors (_Type_='M COEFFI' observations).

The second PROC TRANSREG step fits a univariate regression model with Ride designated IDENTITY, and Prin1 and Prin2 designated POINT. The POINT expansion creates an additional independent variable _ISSQ_, which contains the sum of Prin1 squared and Prin2 squared. The OUT= data set TResult2 contains no _Type_='SCORE' observations, only ideal point (_Type_='M POINT') coordinates for Ride. The coordinates of both the vectors and the ideal points are output by specifying COORDINATES in the OUTPUT statement in PROC TRANSREG.

A vector model is used for MPG and Reliability because perfectly efficient and reliable automobiles do not exist in the data set. The ideal points for MPG and Reliability are far removed from the plot of the automobiles. It is more likely that an ideal point for quality of the ride is in the plot, so an ideal point model is used for the ride variable. See Carroll (1972) and Schiffman, Reynolds, and Young (1981) for discussions of the vector model and point models (including the EPOINT and QPOINT versions of the point model that are not used in this example). For the vector model, the default coordinates stretch factor of 2.5 was used. This extends the vectors by a factor of 2.5 from their standard lengths, making a better graphical display. Sometimes the default vectors are short and near the origin, and they look better when they are extended.

The following statements produce Output 126.6.1 through Output 126.6.5:

```
title 'Preference Ratings for Automobiles Manufactured in 1980';

options validvarname=any;

data CarPreferences;
   input Make $ 1-10 Model $ 12-22 @25 ('1'n-'25'n) (1.)
         MPG Reliability Ride;
   datalines;
Cadillac   Eldorado     8007990491240508971093809 3 2 4
Chevrolet  Chevette     0051200423451043003515698 5 3 2
Chevrolet  Citation     4053305814161643544747795 4 1 5
Chevrolet  Malibu       6027400723121345545668658 3 3 4
Ford       Fairmont     2024006715021443530648655 3 3 4
Ford       Mustang      5007197705021101850657555 3 2 2
```

```
Ford       Pinto        00210003030302015005140 78 4 1 1
Honda      Accord       59568976096999529989750 78 5 5 3
Honda      Civic        48367095074888525677650 75 5 5 3
Lincoln    Continental  70089905922304099620919 09 2 4 5
Plymouth   Gran Fury    70060004341011073334587 08 2 1 5
Plymouth   Horizon      30050056354613024446756 55 4 3 3
Plymouth   Volare       40050036140216027544765 55 2 1 3
Pontiac    Firebird     01078956132012069582659 07 1 1 5
Volkswagen Dasher       48586965088777953778950 00 5 3 4
Volkswagen Rabbit       48585097096957954878850 00 5 4 3
Volvo      DL           99899989099999879899190 00 4 5 5
;

ods graphics on;

* Compute Coordinates for a 2-Dimensional Scatter Plot of Automobiles;
proc prinqual data=CarPreferences out=PResults(drop='1'n-'25'n)
              n=2 replace standard scores mdpref=2;
   id Model MPG Reliability Ride;
   transform identity('1'n-'25'n);
   title2 'Multidimensional Preference (MDPREF) Analysis';
   ods output mdprefplot=md;
run;

options validvarname=v7;

title2 'Preference Mapping (PREFMAP) Analysis';

* Add the Labels from the Plot to the Results Data Set;
data plot;
   if 0 then set md(keep=prin:);
   set presults;
run;

* Compute Endpoints for MPG and Reliability Vectors;
proc transreg data=plot rsquare;
   Model identity(MPG Reliability)=identity(Prin1 Prin2);
   output tstandard=center coordinates replace out=TResult1;
   id Model;
run;

* Compute Ride Ideal Point Coordinates;
proc transreg data=plot rsquare;
   Model identity(Ride)=point(Prin1 Prin2);
   output tstandard=center coordinates replace noscores out=TResult2;
   id Model;
run;

proc print;
run;
```

Output 126.6.1 Preference Ratings Example Output

Preference Ratings for Automobiles Manufactured in 1980
Multidimensional Preference (MDPREF) Analysis

The PRINQUAL Procedure

PRINQUAL MTV Algorithm Iteration History

Iteration Number	Average Change	Maximum Change	Proportion of Variance	Criterion Change	Note
1	0.00000	0.00000	0.66946		Converged

Algorithm converged.

Output 126.6.2 MDPREF Plot

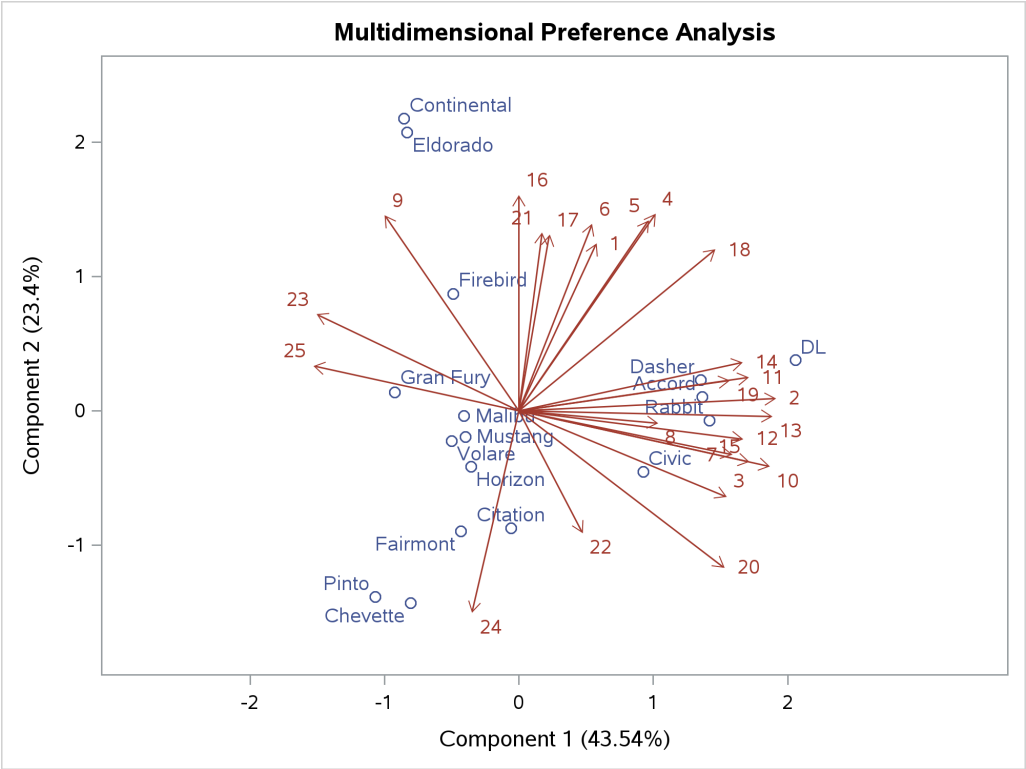

Output 126.6.3 shows that an unreliable-to-reliable direction extends from the left and slightly below the origin to the right and slightly above the origin. The Japanese and European automobiles are rated, on the average, as more reliable. A low MPG to good MPG direction extends from the top left of the plot to the bottom right. The smaller automobiles, on the average, get better gas mileage.

Output 126.6.3 Preference Mapping Vector Plot

Preference Ratings for Automobiles Manufactured in 1980
Preference Mapping (PREFMAP) Analysis

The TRANSREG Procedure

The TRANSREG Procedure Hypothesis Tests for Identity(MPG)

R-Square	0.5720

The TRANSREG Procedure Hypothesis Tests for Identity(Reliability)

R-Square	0.5086

Output 126.6.3 *continued*

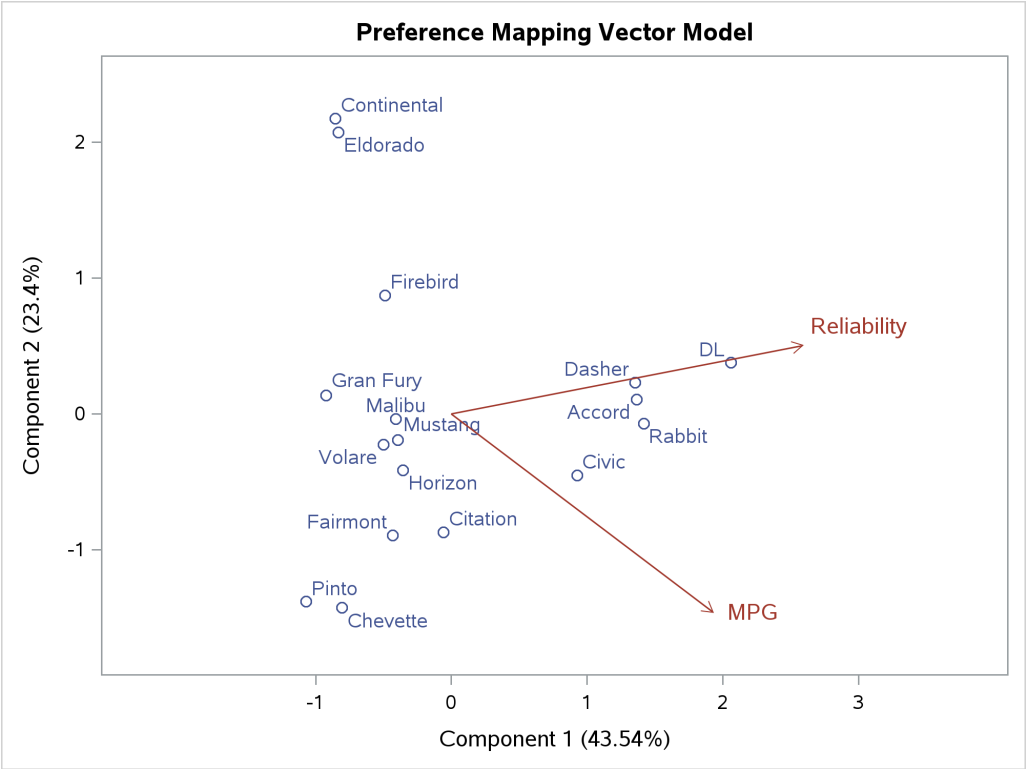

The ideal point for Ride in Output 126.6.4 is in the top, just right of the center of the plot. Automobiles near the Ride ideal point tend to have a better ride than automobiles far away. It can be seen from the R squares that none of these ratings perfectly fits the model, so all of the interpretations are approximate.

Output 126.6.4 Preference Mapping Ideal Point Plot

Preference Ratings for Automobiles Manufactured in 1980
Preference Mapping (PREFMAP) Analysis

The TRANSREG Procedure

The TRANSREG Procedure Hypothesis Tests for Identity(Ride)

R-Square	0.3780

Output 126.6.4 *continued*

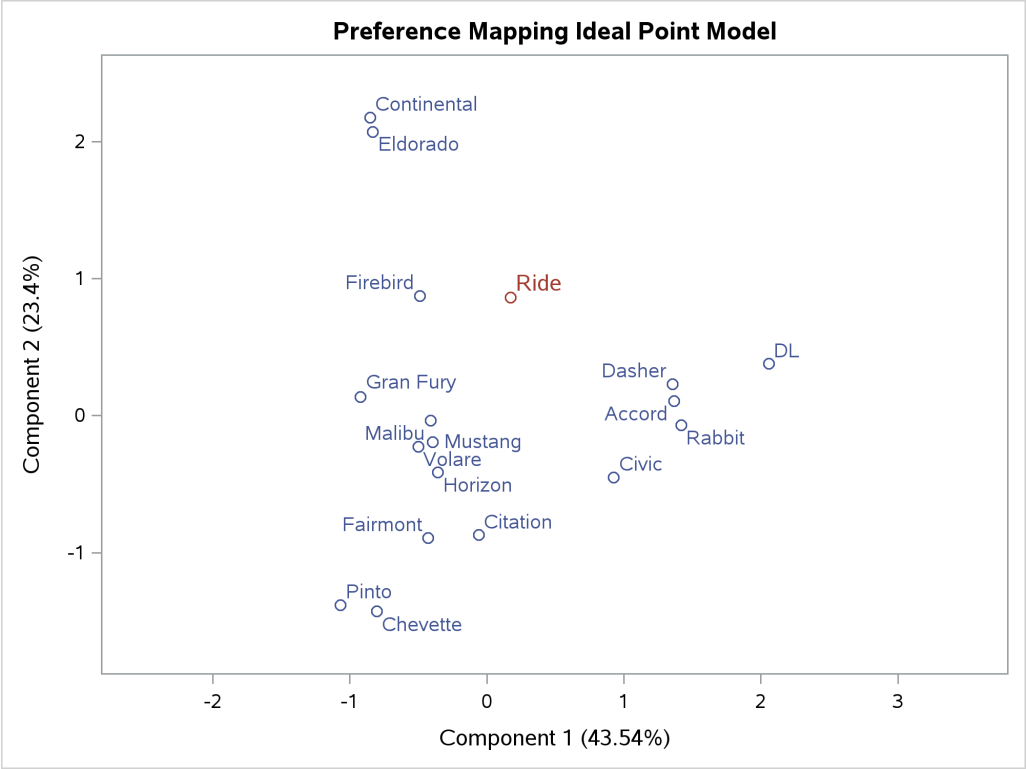

The Ride point is a "negative-negative" ideal point. The point models assume that small ratings mean the object (automobile) is similar to the rating name and large ratings imply dissimilarity to the rating name. Because the opposite scoring is used, the interpretation of the Ride point must be reversed to a negative ideal point (bad ride). However, the coefficient for the _ISSQ_ variable in Output 126.6.5 is negative, so the interpretation is reversed again, back to the original interpretation.

Output 126.6.5 Preference Mapping Ideal Point Coefficients

Preference Ratings for Automobiles Manufactured in 1980
Preference Mapping (PREFMAP) Analysis

Obs	_TYPE_	_NAME_	Ride	Intercept	Prin1	Prin2	_ISSQ_	Model
1	M POINT	Ride	.	.	0.49461	2.46539	-0.17448	Ride

References

Akaike, H. (1973). "Information Theory and an Extension of the Maximum Likelihood Principle." In *Proceedings of the Second International Symposium on Information Theory*, edited by B. N. Petrov and F. Csáki, 267–281. Budapest: Akademiai Kiado.

Box, G. E. P., and Cox, D. R. (1964). "An Analysis of Transformations." *Journal of the Royal Statistical Society, Series B* 26:211–234.

Breiman, L., and Friedman, J. H. (1985). "Estimating Optimal Transformations for Multiple Regression and Correlation." *Journal of the American Statistical Association* 77:580–619. With discussion.

Brent, R. P. (1973). *Algorithms for Minimization without Derivatives*. Englewood Cliffs, NJ: Prentice-Hall. Chapter 5.

Brinkman, N. D. (1981). "Ethanol Fuel: A Single-Cylinder Engine Study of Efficiency and Exhaust Emissions." *Society of Automotive Engineers Transactions* 90:1410–1424.

Carroll, J. D. (1972). "Individual Differences and Multidimensional Scaling." In *Multidimensional Scaling: Theory and Applications in the Behavioral Sciences*, vol. 1, edited by R. N. Shepard, A. K. Romney, and S. B. Nerlove, 105–155. New York: Seminar Press.

Craven, P., and Wahba, G. (1979). "Smoothing Noisy Data with Spline Functions." *Numerical Mathematics* 31:377–403.

De Boor, C. (1978). *A Practical Guide to Splines*. New York: Springer-Verlag.

De Leeuw, J. (1986). *Regression with Optimal Scaling of the Dependent Variable*. Leiden, Netherlands: Department of Data Theory, University of Leiden.

De Leeuw, J., Young, F. W., and Takane, Y. (1976). "Additive Structure in Qualitative Data: An Alternating Least Squares Approach with Optimal Scaling Features." *Psychometrika* 41:471–503.

Draper, N. R., and Smith, H. (1981). *Applied Regression Analysis*. 2nd ed. New York: John Wiley & Sons.

Eilers, P. H. C., and Marx, B. D. (1996). "Flexible Smoothing with *B*-Splines and Penalties." *Statistical Science* 11:89–121. With discussion.

Fisher, R. A. (1938). *Statistical Methods for Research Workers*. 10th ed. Edinburgh: Oliver & Boyd.

Gabriel, K. R. (1981). "Biplot Display of Multivariate Matrices for Inspection of Data and Diagnosis." In *Interpreting Multivariate Data*, edited by V. Barnett, 571–572. Chichester, UK: John Wiley & Sons.

Gifi, A. (1990). *Nonlinear Multivariate Analysis*. New York: John Wiley & Sons.

Green, P. E., and Wind, Y. (1975). "New Way to Measure Consumers' Judgments." *Harvard Business Review* 53:107–117.

Hastie, T. J., and Tibshirani, R. J. (1986). "Generalized Additive Models." *Statistical Science* 1:297–318.

Hurvich, C. M., Simonoff, J. S., and Tsai, C.-L. (1998). "Smoothing Parameter Selection in Nonparametric Regression Using an Improved Akaike Information Criterion." *Journal of the Royal Statistical Society, Series B* 60:271–293.

Israels, A. Z. (1984). "Redundancy Analysis for Qualitative Variables." *Psychometrika* 49:331–346.

Judge, G. G., Griffiths, W. E., Hill, R. C., and Lee, T.-C. (1980). *The Theory and Practice of Econometrics*. New York: John Wiley & Sons.

Khuri, A. I., and Cornell, J. A. (1987). *Response Surfaces*. New York: Marcel Dekker.

Kruskal, J. B. (1964). "Nonmetric Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis." *Psychometrika* 29:1–27.

Kuhfeld, W. F. (2010). *Marketing Research Methods in SAS*. Technical report, SAS Institute Inc., Cary, NC. http://support.sas.com/resources/papers/tnote/tnote_marketresearch.html.

Myers, R. H. (1976). *Response Surface Methodology*. Blacksburg: Virginia Polytechnic Institute and State University.

National Institute of Standards and Technology (1998). "Statistical Reference Data Sets." Accessed June 6, 2011. http://www.itl.nist.gov/div898/strd/general/dataarchive.html.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1989). *Numerical Recipes in PASCAL*. Cambridge: Cambridge University Press.

Reinsch, C. H. (1967). "Smoothing by Spline Functions." *Numerische Mathematik* 10:177–183.

SAS Institute Inc. (1993). *Algorithms for the PRINQUAL and TRANSREG Procedures*. Technical Report R-108, SAS Institute Inc., Cary, NC. http://support.sas.com/publishing/pubcat/techreports/59040.pdf.

Schiffman, S. S., Reynolds, M. L., and Young, F. W. (1981). *Introduction to Multidimensional Scaling*. New York: Academic Press.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Siegel, S. (1956). *Nonparametric Statistics*. New York: McGraw-Hill.

Smith, P. L. (1979). "Splines as a Useful and Convenient Statistical Tool." *American Statistician* 33:57–62.

Stewart, D. K., and Love, W. A. (1968). "A General Canonical Correlation Index." *Psychological Bulletin* 70:160–163.

Van der Burg, E., and de Leeuw, J. (1983). "Non-linear Canonical Correlation." *British Journal of Mathematical and Statistical Psychology* 36:54–80.

Van Rijckevorsel, J. L. (1982). "Canonical Analysis with B-Splines." In *COMPUSTAT 1982, Part I*, edited by H. Caussinus, P. Ettinger, and R. Tomassone, 393–398. Vienna: Physica-Verlag.

Winsberg, S., and Ramsay, J. O. (1980). "Monotonic Transformations to Additivity Using Splines." *Biometrika* 67:669–674.

Young, F. W. (1981). "Quantitative Analysis of Qualitative Data." *Psychometrika* 46:357–388.

Young, F. W., de Leeuw, J., and Takane, Y. (1976). "Regression with Qualitative and Quantitative Variables: An Alternating Least Squares Approach with Optimal Scaling Features." *Psychometrika* 41:505–529.

Chapter 127
The TREE Procedure

Contents

Overview: TREE Procedure **10778**
Getting Started: TREE Procedure **10778**
Syntax: TREE Procedure **10783**
- PROC TREE Statement 10783
- BY Statement 10789
- COPY Statement 10789
- FREQ Statement 10789
- HEIGHT Statement 10790
- ID Statement 10790
- NAME Statement 10790
- PARENT Statement 10790

Details: TREE Procedure **10791**
- Missing Values 10791
- Output Data Set 10791
- Displayed Output 10792
- ODS Table Names 10792

Examples: TREE Procedure **10793**
- Example 127.1: Mammals' Teeth 10793
- Example 127.2: Iris Data 10801

References **10802**

Overview: TREE Procedure

The TREE procedure reads a data set created by the CLUSTER or VARCLUS procedure and produces a tree diagram (also known as a *dendrogram* or *phenogram*), which displays the results of a hierarchical clustering analysis as a tree structure. The TREE procedure uses the data set to produce a diagram of the tree structure in the style of Johnson (1967), with the root at the top. Alternatively, the diagram can be oriented horizontally, with the root at the left. Any numeric variable in the output data set can be used to specify the heights of the clusters. PROC TREE can also create an output data set that contains a variable to indicate the disjoint clusters at a specified level in the tree.

Tree diagrams are discussed in the context of cluster analysis by Duran and Odell (1974); Hartigan (1975); Everitt (1980). Knuth (1973) provides a general treatment of tree diagrams in computer programming.

The literature on tree diagrams contains a mixture of botanical and genealogical terminology. The objects that are clustered are *leaves*. The cluster that contains all objects is the *root*. A cluster that contains at least two objects but not all of them is a *branch*. The general term for leaves, branches, and roots is *node*. If a cluster A is the union of clusters B and C, then A is the *parent* of B and C, and B and C are *children* of A. A leaf is thus a node with no children, and a root is a node with no parent. If every cluster has at most two children, the tree diagram is a *binary tree*. The CLUSTER procedure always produces binary trees. The VARCLUS procedure can produce tree diagrams with clusters that have many children.

Getting Started: TREE Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/treegs.sas.)

The TREE procedure creates tree diagrams from a SAS data set that contains the tree structure. You can create this type of data set with the CLUSTER or VARCLUS procedure. See Chapter 40, "The CLUSTER Procedure," and Chapter 129, "The VARCLUS Procedure," for more information.

In the following example, the VARCLUS procedure is used to divide a set of variables into hierarchical clusters and to create the SAS data set that contains the tree structure. The TREE procedure then generates the tree diagrams.

The following data, from Hand et al. (1994), represent the amount of protein consumed from nine food groups for each of 25 European countries. The nine food groups are red meat (RedMeat), white meat (WhiteMeat), eggs (Eggs), milk (Milk), fish (Fish), cereal (Cereal), starch (Starch), nuts (Nuts), and fruits and vegetables (FruVeg).

The following SAS statements create the data set Protein:

```
data Protein;
   input Country $15. RedMeat WhiteMeat Eggs Milk
      Fish Cereal Starch Nuts FruVeg;
   datalines;
Albania        10.1  1.4  0.5   8.9  0.2  42.3  0.6  5.5  1.7
Austria         8.9 14.0  4.3  19.9  2.1  28.0  3.6  1.3  4.3
Belgium        13.5  9.3  4.1  17.5  4.5  26.6  5.7  2.1  4.0
```

```
Bulgaria        7.8  6.0  1.6   8.3  1.2  56.7  1.1  3.7  4.2
Czechoslovakia  9.7 11.4  2.8  12.5  2.0  34.3  5.0  1.1  4.0
Denmark        10.6 10.8  3.7  25.0  9.9  21.9  4.8  0.7  2.4
E Germany       8.4 11.6  3.7  11.1  5.4  24.6  6.5  0.8  3.6
Finland         9.5  4.9  2.7  33.7  5.8  26.3  5.1  1.0  1.4
France         18.0  9.9  3.3  19.5  5.7  28.1  4.8  2.4  6.5
Greece         10.2  3.0  2.8  17.6  5.9  41.7  2.2  7.8  6.5
Hungary         5.3 12.4  2.9   9.7  0.3  40.1  4.0  5.4  4.2
Ireland        13.9 10.0  4.7  25.8  2.2  24.0  6.2  1.6  2.9
Italy           9.0  5.1  2.9  13.7  3.4  36.8  2.1  4.3  6.7
Netherlands     9.5 13.6  3.6  23.4  2.5  22.4  4.2  1.8  3.7
Norway          9.4  4.7  2.7  23.3  9.7  23.0  4.6  1.6  2.7
Poland          6.9 10.2  2.7  19.3  3.0  36.1  5.9  2.0  6.6
Portugal        6.2  3.7  1.1   4.9 14.2  27.0  5.9  4.7  7.9
Romania         6.2  6.3  1.5  11.1  1.0  49.6  3.1  5.3  2.8
Spain           7.1  3.4  3.1   8.6  7.0  29.2  5.7  5.9  7.2
Sweden          9.9  7.8  3.5   4.7  7.5  19.5  3.7  1.4  2.0
Switzerland    13.1 10.1  3.1  23.8  2.3  25.6  2.8  2.4  4.9
UK             17.4  5.7  4.7  20.6  4.3  24.3  4.7  3.4  3.3
USSR            9.3  4.6  2.1  16.6  3.0  43.6  6.4  3.4  2.9
W Germany      11.4 12.5  4.1  18.8  3.4  18.6  5.2  1.5  3.8
Yugoslavia      4.4  5.0  1.2   9.5  0.6  55.9  3.0  5.7  3.2
;
```

The data set Protein contains the character variable Country and the nine numeric variables that represent the food groups. The **$15.** in the INPUT statement specifies that the variable Country is a character variable with a length of 15.

The following statements cluster the variables in the data set Protein:

```
ods graphics on;

proc varclus data=Protein outtree=Tree centroid maxclusters=4
             plots=dendrogram(vertical height=ncl);
   var RedMeat--FruVeg;
run;
```

The OUTTREE= option creates an output data set named Tree to contain the tree structure. The CENTROID option specifies the centroid clustering method, and the MAXCLUSTERS= option specifies that the largest number of clusters desired is four. The VAR statement specifies that all numeric variables (RedMeat—FruVeg) are used by the procedure. Since ODS Graphics is enabled, PROC VARCLUS creates a dendrogram, which is displayed in Figure 127.1. The option **plots=dendrogram(vertical height=ncl)** specifies a vertical dendrogram with the number of clusters on the vertical axis. The default is a horizontal dendrogram with, for this cluster analysis, the proportion of variance explained on the horizontal axis.

Figure 127.1 Dendrogram from PROC VARCLUS and ODS Graphics

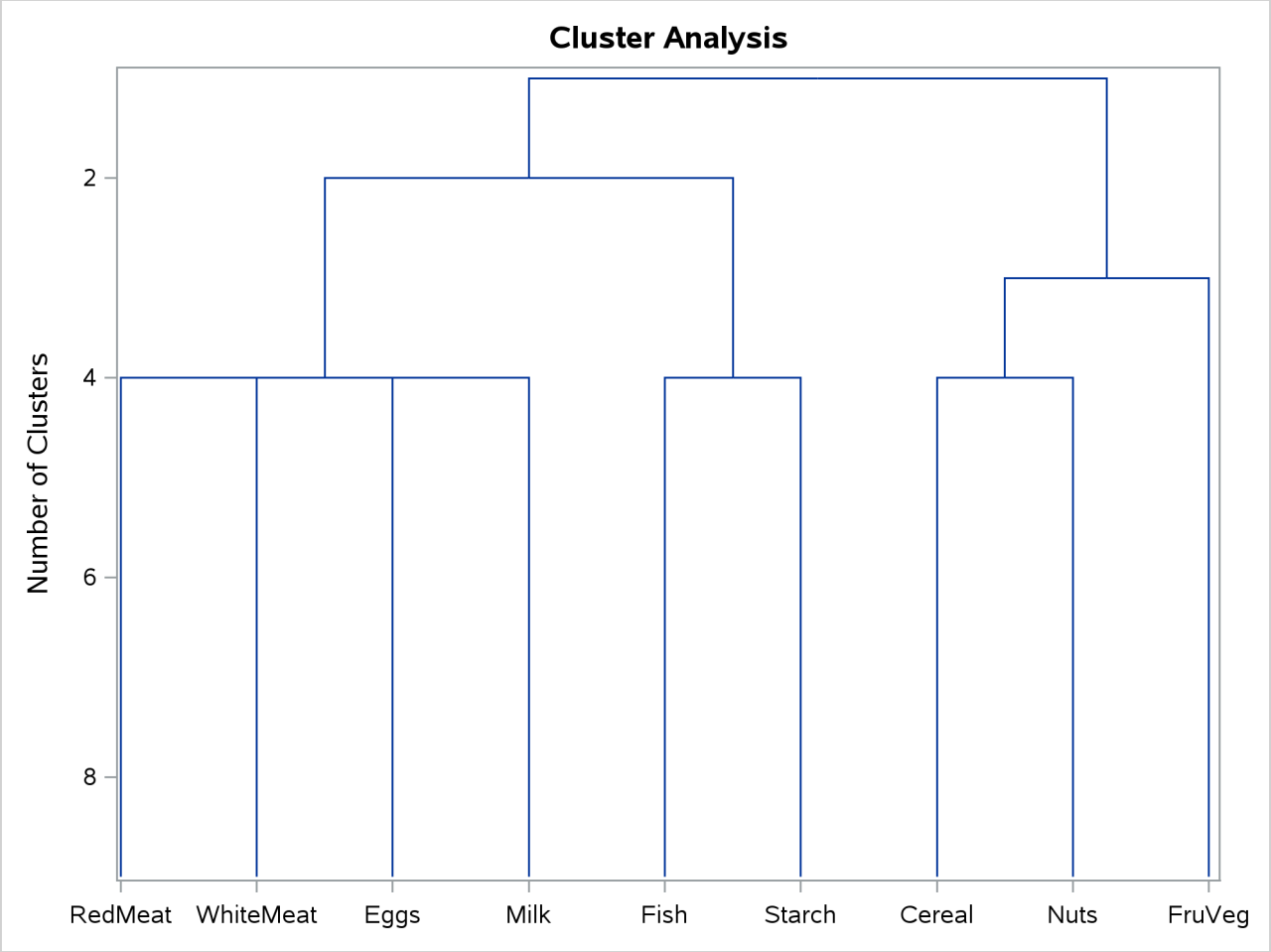

The output data set Tree, created by the OUTTREE= option in the previous statements, contains the following variables:

NAME	the name of the cluster
PARENT	the parent of the cluster
NCL	the number of clusters
VAREXP	the amount of variance explained by the cluster
PROPOR	the proportion of variance explained by the clusters at the current level of the tree diagram
MINPRO	the minimum proportion of variance explained by a cluster
MAXEIGEN	the maximum second eigenvalue of a cluster

Figure 127.2 Graphical Tree Diagram from PROC TREE

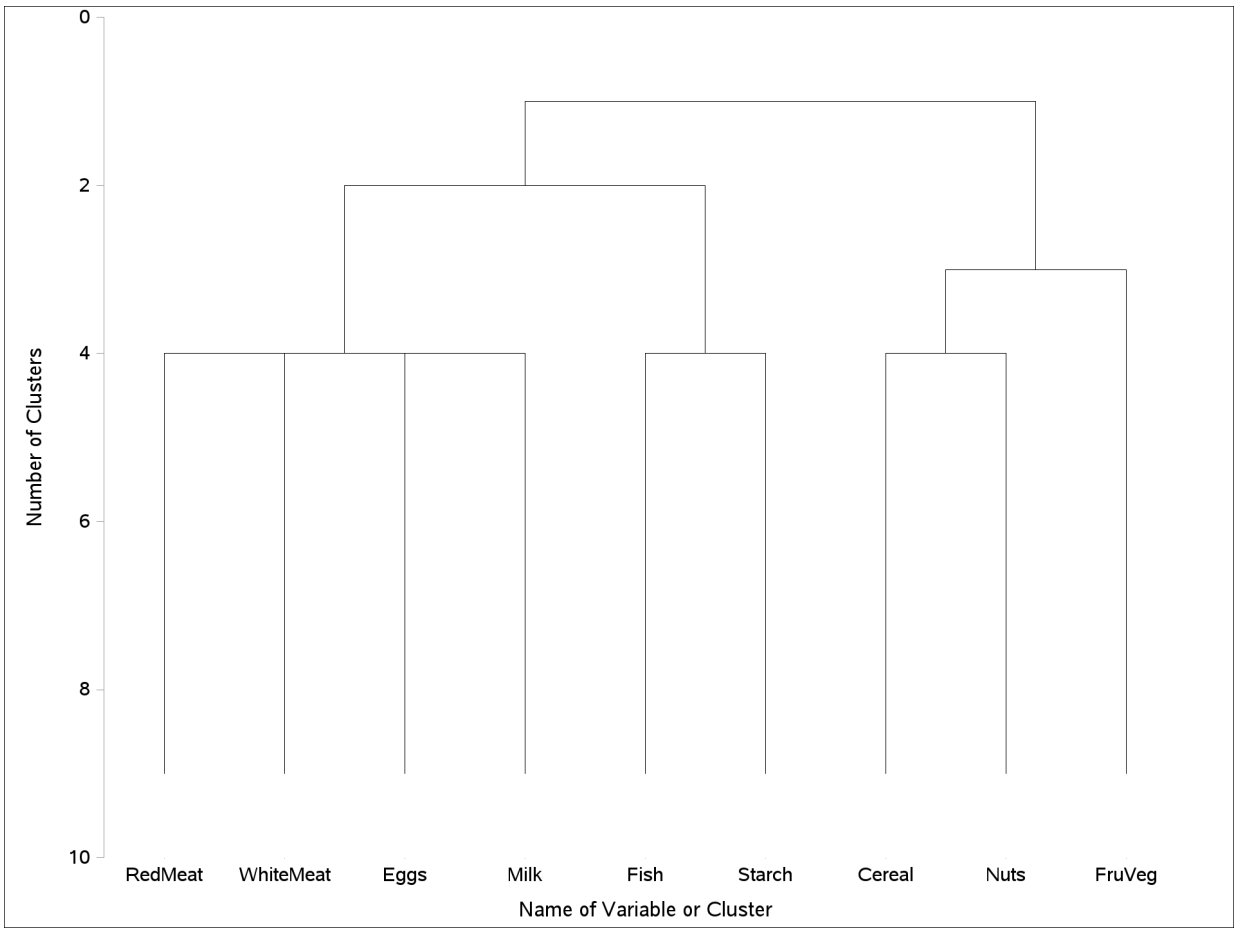

The following statements use PROC TREE to a produce tree diagram of the clusters created by PROC VARCLUS:

```
proc tree data=tree vaxis=axis1;
   axis1 label=(angle=90);
run;
```

The AXIS1 statement rotates the Y-axis label so that it is displayed in the same way as in the dendrogram from PROC VARCLUS. Figure 127.2 displays the tree diagram.

In each diagram, the name of the cluster is displayed on the horizontal axis and the number of clusters is displayed on the vertical (height) axis.

As you look up from the bottom of either diagram, clusters are progressively joined until a single, all-encompassing cluster is formed at the top (or root) of the tree. Clusters exist at each level of the diagram. For example, at the level where the diagram indicates three clusters, the clusters are as follows:

- Cluster 1: RedMeat WhiteMeat Eggs Milk
- Cluster 2: Fish Starch
- Cluster 3: Cereal Nuts FruVeg

As you proceed up the diagram one level, the number of clusters is two:

- Cluster 1: RedMeat WhiteMeat Eggs Milk Fish Starch
- Cluster 2: Cereal Nuts FruVeg

The following statements illustrate how you can specify the numeric variable that defines the height of each node (cluster) in the tree:

```
proc tree data=Tree horizontal haxis=axis1;
   axis1 order=(0 to 1 by 0.2);
   height _PROPOR_;
   label _name_ = 'Variable';
run;
```

The HORIZONTAL option in the PROC TREE statement orients the tree diagram horizontally. The HAXIS= option specifies that the AXIS1 statement be used to customize the appearance of the horizontal axis. The ORDER= option in the AXIS1 statement (see *SAS/GRAPH: Reference*) specifies the data values in the order in which they are to appear on the axis. The HEIGHT statement specifies the variable _PROPOR_ (the proportion of variance explained) as the height variable. The LABEL statement supplies a short Y-axis label.

The resulting tree diagram is shown in Figure 127.3.

Figure 127.3 Horizontal Tree Diagram with _PROPOR_ as the HEIGHT Variable

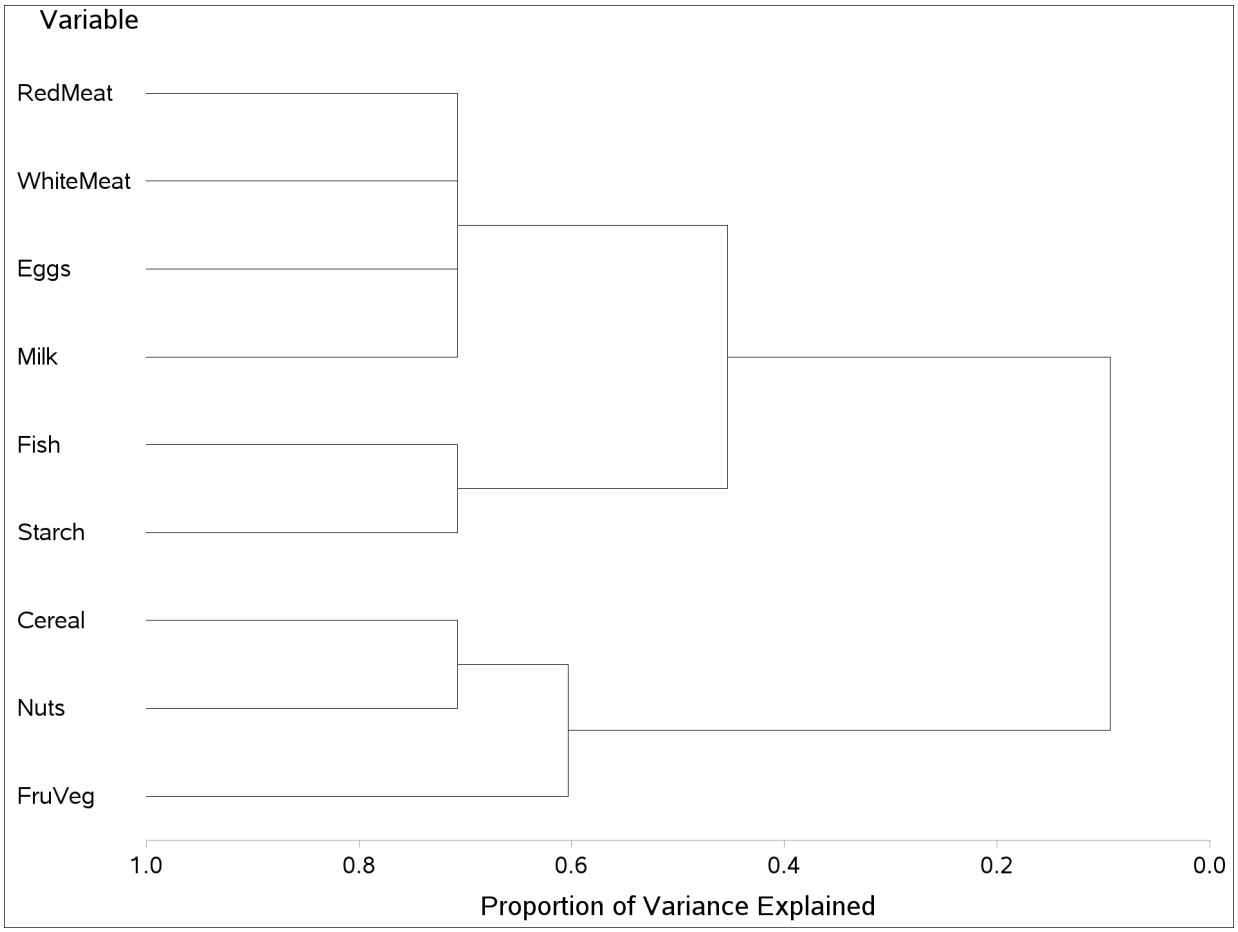

As you look from left to right in the diagram, objects and clusters are progressively joined until a single, all-encompassing cluster is formed at the right (or root) of the tree.

Clusters exist at each level of the diagram, represented by horizontal line segments. Each vertical line segment represents a point where leaves and branches are connected into progressively larger clusters.

For example, three clusters are formed at the leftmost point along the axis where three horizontal line segments exist. At that point, where a vertical line segment connects the Cereal-Nuts and FruVeg clusters, the proportion of variance explained is about 0.6 (_PROPOR_ = 0.6). At the next clustering level the variables Fish and Starch are clustered with variables RedMeat through Milk, resulting in a total of two clusters. The proportion of variance explained is about 0.45 at that point.

Syntax: TREE Procedure

The following statements are available in the TREE procedure:

PROC TREE < *options* > **;**
  NAME *variables* **;**
  HEIGHT *variable* **;**
  PARENT *variables* **;**
  BY *variables* **;**
  COPY *variables* **;**
  FREQ *variable* **;**
  ID *variable* **;**

If the input data set has been created by CLUSTER or VARCLUS, the only statement required is the PROC TREE statement. The BY, COPY, FREQ, HEIGHT, ID, NAME, and PARENT statements are described after the PROC TREE statement.

PROC TREE Statement

PROC TREE < *options* > **;**

The PROC TREE statement invokes the TREE procedure.

Table 127.1 summarizes the *options* available in the PROC TREE statement.

Table 127.1 PROC TREE Statement Options

Option	Description
Data Sets	
DATA=	specifies input data set
DOCK=	specifies that small clusters not be counted in OUT= data set
LEVEL=	defines disjoint cluster in OUT= data set
NCLUSTERS=	specifies number of clusters in OUT= data set
OUT=	specifies output data set

Table 127.1 *continued*

Option	Description
ROOT=	displays root of a subtree
Cluster Heights	
HEIGHT=	specifies variable for the height axis
DISSIMILAR	specifies that height values indicate dissimilarity
SIMILAR	specifies that height values indicate similarity
Horizontal Trees	
HORIZONTAL	specifies that height axis be horizontal
Sort Order	
DESCENDING	reverses sort order
SORT	sorts children by HEIGHT variable
Displayed Output	
INC=	specifies increment between tick values
LINEPRINTER	displays tree by using line printer graphics
LIST	displays all nodes in tree
MAXHEIGHT=	specifies maximum value on axis
MINHEIGHT=	specifies minimum value on axis
NOPRINT	suppresses display of tree
NTICK=	specifies number of tick intervals
Graphics	
CFRAME=	specifies color of the frame
DESCRIPTION=	specifies catalog description
GOUT=	specifies catalog name
HAXIS=	customizes horizontal axis
HORDISPLAY=	displays horizontal tree with leaves on right
HPAGES=	specifies number of pages to expand tree horizontally
LINES=	specifies line color and thickness, dots at nodes
NAME=	specifies name of graph in catalog
VAXIS=	customizes vertical axis
VPAGES=	specifies number of pages to expand tree vertically
Line Printer Graphics	
PAGES=	specifies number of pages
POS=	specifies number of column positions
SPACES=	specifies number of spaces between objects
TICKPOS=	specifies number of column positions between ticks
FILLCHAR=	specifies fill character between unjoined leaves
JOINCHAR=	specifies character displayed between joined leaves
LEAFCHAR=	specifies character representing clusters with no children
TREECHAR=	specifies character representing clusters with children

CFRAME=*color*
specifies a color for the frame, which is the rectangle bounded by the axes.

DATA=*SAS-data-set*
specifies the input data set that defines the tree. If you omit the DATA= option, the most recently created SAS data set is used.

DESCENDING
DES
reverses the sort order for the SORT option.

DESCRIPTION=*entry-description*
specifies a description for the graph in the GOUT= catalog. The default is "Proc Tree Graph Output."

DISSIMILAR
DIS
specifies that the values of the HEIGHT variable are dissimilarities; that is, a large height value means that the clusters are very dissimilar or far apart.

If neither the SIMILAR nor the DISSIMILAR option is specified, PROC TREE attempts to infer from the data whether the height values are similarities or dissimilarities. If PROC TREE cannot tell this from the data, it issues an error message and does not display a tree diagram.

DOCK=*n*
causes observations in the OUT= data set that have a frequency of *n* or less to be given missing values for the output variables CLUSTER and CLUSNAME. If the NCLUSTERS= option is also specified, DOCK= also prevents clusters with a frequency of *n* or less from being counted toward the number of clusters requested by the NCLUSTERS= option. By default, DOCK=0.

FILLCHAR=*'c'*
FC=*'c'*
specifies the character displayed between leaves that are not joined into a cluster. The character should be enclosed in single quotes. The default is a blank. The LINEPRINTER option must also be specified.

GOUT=< *libref.* >*member-name*
specifies the catalog in which the generated graph is stored. The default is Work.Gseg.

HAXIS=AXIS*n*
specifies that the AXIS*n* statement be used to customize the appearance of the horizontal axis.

HEIGHT=*name*
H=*name*
specifies certain conventional variables to be used for the height axis of the tree diagram. For many situations, the only option you need is the HEIGHT= option. Valid values for *name* and their meanings are as follows:

HEIGHT | H	specifies the _HEIGHT_ variable.
LENGTH | L	defines the height of each node as its path length from the root. This can also be interpreted as the number of ancestors of the node.
MODE | M	specifies the _MODE_ variable.

NCL | N specifies the _NCL_ (number of clusters) variable.

RSQ | R specifies the _RSQ_ variable.

See also the section "HEIGHT Statement" on page 10790. The HEIGHT statement can specify any variable in the input data set to be used for the height axis. In rare cases, you might need to specify either the DISSIMILAR option or the SIMILAR option.

HORDISPLAY=RIGHT

specifies that the graph be oriented horizontally with the leaf nodes on the right side, when the HORIZONTAL option is also specified. By default, the leaf nodes are on the left side.

HORIZONTAL

HOR

displays the tree diagram with the height axis oriented horizontally. The leaf nodes are on the side specified in the HORDISPLAY= option. If you do not specify the HORIZONTAL option, the height axis is vertical, with the root at the top. When the tree takes up more than one page, horizontal orientation can make the tree diagram considerably easier to read.

HPAGES=*n1*

specifies that the original graph be enlarged to cover *n1* pages. If you also specify the VPAGES=*n2* option, the original graph is enlarged to cover $n1 \times n2$ graphs. For example, if HPAGES=2 and VPAGES=3, then the original graph is generated, followed by $2 \times 3 = 6$ more graphs. In these six graphs, the original is enlarged by a factor of 2 in the horizontal direction and by a factor of 3 in the vertical direction. The graphs are generated in left-to-right and top-to-bottom order.

INC=*n*

specifies the increment between tick values on the height axis. If the HEIGHT variable is _NCL_, the default is usually 1, although a different value can be specified for consistency with other options. For any other HEIGHT variable, the default is some power of 10 times 1, 2, 2.5, or 5.

JOINCHAR=*'c'*

JC=*'c'*

specifies the character displayed between leaves that are joined into a cluster. The character should be enclosed in single quotes. The default is 'X'. The LINEPRINTER option must also be specified.

LEAFCHAR=*'c'*

LC=*'c'*

specifies the character used to represent clusters that have no children. The character should be enclosed in single quotes. The default is a period. The LINEPRINTER option must also be specified.

LEVEL=*n*

specifies the level of the tree that defines disjoint clusters for the OUT= data set. The LEVEL= option also causes only clusters between the root and a height of *n* to be displayed. The clusters in the output data set are those that exist at a height of *n* on the tree diagram. For example, if the HEIGHT variable is _NCL_ (number of clusters) and LEVEL=5 is specified, then the OUT= data set contains five disjoint clusters. If the HEIGHT variable is _RSQ_ (R square) and LEVEL=0.9 is specified, then the OUT= data set contains the smallest number of clusters that yields an R square of at least 0.9.

LINEPRINTER

specifies that the tree diagram be displayed using line printer graphics.

LINES=(< COLOR=*color* > < WIDTH=*n* > < DOTS >)

specifies the color and the thickness of the lines of the tree, and whether a dot is drawn at each leaf node. If the frame and the lines are specified to be the same color, PROC TREE selects a different color for the lines.

LIST

lists all the nodes in the tree, displaying the height, parent, and children of each node.

MAXHEIGHT=*n*

MAXH=*n*

specifies the maximum value displayed on the height axis.

MINHEIGHT=*n*

MINH=*n*

specifies the minimum value displayed on the height axis.

NAME=*name*

specifies the entry name for the generated graph in the GOUT= catalog. Each time another graph is generated with the same name, the name is modified by appending a number to make it unique.

NCLUSTERS=*n*

NCL=*n*

N=*n*

specifies the number of clusters desired in the OUT= data set. The number of clusters obtained might not equal the number specified if (1) there are fewer than *n* leaves in the tree, (2) there are more than *n* unconnected trees in the data set, (3) a multiway tree does not contain a level with the specified number of clusters, or (4) the DOCK= option eliminates too many clusters.

The NCLUSTERS= option uses the _NCL_ variable to determine the order in which the clusters are formed. If there is no _NCL_ variable, the height variable (as determined by the HEIGHT statement or HEIGHT= option) is used instead.

NTICK=*n*

specifies the number of tick intervals on the height axis. The default depends on the values of other options.

NOPRINT

suppresses the display of the tree. Specify the NOPRINT option if you want only to create an OUT= data set.

OUT=*SAS-data-set*

creates an output data set that contains one observation for each object in the tree or subtree being processed and variables called CLUSTER and CLUSNAME that show cluster membership at any specified level in the tree. If you specify the OUT= option, you must also specify either the NCLUSTERS= or LEVEL= option in order to define the output partition level. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PAGES=*n*

specifies the number of pages over which the tree diagram (from root to leaves) is to extend. The default is 1. The LINEPRINTER option must also be specified.

POS=*n*

specifies the number of column positions on the height axis. The default depends on the value of the PAGES= option, the orientation of the tree diagram, and the values specified by the PAGESIZE= and LINESIZE= options. The LINEPRINTER option must also be specified.

ROOT=*'name'*

specifies the value of the NAME statement variable for the root of a subtree to be displayed if you do not want to display the entire tree. If you also specify the OUT= option, the output data set contains only objects that belong to the subtree specified by the ROOT= option.

SIMILAR

SIM

specifies that the values of the HEIGHT variable represent similarities; that is, a large height value means that the clusters are very similar or close together.

If neither the SIMILAR nor the DISSIMILAR option is specified, PROC TREE attempts to infer from the data whether the height values are similarities or dissimilarities. If PROC TREE cannot tell this from the data, it issues an error message and does not display a tree diagram.

SORT

sorts the children of each node by the HEIGHT variable, in the order of cluster formation. See the DESCENDING option for details.

SPACES=*s*

S=*s*

specifies the number of spaces between objects in the output. The default depends on the number of objects, the orientation of the tree diagram, and the values specified by the PAGESIZE= and LINESIZE= options. The LINEPRINTER option must also be specified.

TICKPOS=*n*

specifies the number of column positions per tick interval on the height axis. The default value is usually between 5 and 10, although a different value can be specified for consistency with other options.

TREECHAR=*'c'*

TC=*'c'*

specifies the character used to represent clusters with children. The character should be enclosed in single quotes. The default is 'X'. The LINEPRINTER option must also be specified.

VAXIS=AXIS*n*

specifies that the AXIS*n* statement be used to customize the appearance of the vertical axis.

VPAGES=*n2*

specifies that the original graph be enlarged to cover *n2* pages. If you also specify the HPAGES=*n1* option, the original graph is enlarged to cover $n1 \times n2$ pages. For example, if HPAGES=2 and VPAGES=3, then the original graph is generated, followed by $2 \times 3 = 6$ more graphs. In these six graphs, the original is enlarged by a factor of 2 in the horizontal direction and by a factor of 3 in the vertical direction. The graphs are generated in left-to-right and top-to-bottom order.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC TREE to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the TREE procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

COPY Statement

COPY *variables* **;**

The COPY statement specifies one or more character or numeric variables to be copied to the OUT= data set.

FREQ Statement

FREQ *variable* **;**

The FREQ statement specifies one numeric variable that tells how many clustering observations belong to the cluster. If the FREQ statement is omitted, PROC TREE uses the variable _FREQ_ to specify the number of observations per cluster. If neither the FREQ statement nor the _FREQ_ variable is present, each leaf is assumed to represent one clustering observation, and the frequency for each internal node is found by summing the frequencies of its children.

HEIGHT Statement

HEIGHT *variable* **;**

The HEIGHT statement specifies the name of a numeric variable to define the height of each node (cluster) in the tree. The height variable can also be specified by the HEIGHT= option in the PROC TREE statement. If both the HEIGHT statement and the HEIGHT= option are omitted, PROC TREE uses the variable _HEIGHT_. If the data set does not contain _HEIGHT_, PROC TREE uses the variable _NCL_. If _NCL_ is not present either, the height of each node is defined to be its path length from the root.

ID Statement

ID *variable* **;**

The ID variable is used to identify the objects (leaves) in the tree on the output. The ID variable can be a character or numeric variable of any length. If the ID statement is omitted, the variable in the NAME statement is used instead. If both the ID and NAME statements are omitted, PROC TREE uses the variable _NAME_. If the _NAME_ variable is not found in the data set, PROC TREE issues an error message and stops. The ID variable is copied to the OUT= data set.

NAME Statement

NAME *variable* **;**

The NAME statement specifies a character or numeric variable that identifies the node represented by each observation. The NAME statement variable and the PARENT statement variable jointly define the tree structure. If the NAME statement is omitted, PROC TREE uses the variable _NAME_. If the _NAME_ variable is not found in the data set, PROC TREE issues an error message and stops.

PARENT Statement

PARENT *variable* **;**

The PARENT statement specifies a character or numeric variable that identifies the node in the tree that is the parent of each observation. The PARENT statement variable must have the same formatted length as the NAME statement variable. If the PARENT statement is omitted, PROC TREE uses the variable _PARENT_. If the _PARENT_ variable is not found in the data set, PROC TREE issues an error message and stops.

Details: TREE Procedure

Missing Values

An observation with a missing value for the NAME statement variable is omitted from processing. If the PARENT statement variable has a missing value but the NAME statement variable is present, the observation is treated as the root of a tree. A data set can contain several roots and, hence, several trees.

Missing values of the HEIGHT variable are set to upper or lower bounds determined from the nonmissing values under the assumption that the heights are monotonic with respect to the tree structure.

Missing values of the FREQ variable are inferred from nonmissing values where possible; otherwise, they are treated as zero.

Output Data Set

The OUT= data set contains one observation for each leaf in the tree or subtree being processed. The variables are as follows:

- the BY variables, if any
- the ID variable, or the NAME statement variable if the ID statement is not used
- the COPY variables
- a numeric variable CLUSTER that takes values from 1 to c, where c is the number of disjoint clusters. The cluster to which the first observation belongs is given the number 1, the cluster to which the next observation belongs that does not belong to cluster 1 is given the number 2, and so on.
- a character variable CLUSNAME that gives the value of the NAME statement variable of the cluster to which the observation belongs

The CLUSTER and CLUSNAME variables are missing if the corresponding leaf has a nonpositive frequency.

Displayed Output

The displayed output from the TREE procedure includes the following:

- the names of the objects in the tree
- the height axis
- the tree diagram.

The leaves of the tree diagram are displayed at the bottom of the graph. Horizontal lines connect the leaves into branches, while the topmost horizontal line indicates the root.

If the LINEPRINTER option is specified, the root (the cluster that contains all the objects) is indicated by a solid line of the character specified by the TREECHAR= option (the default character is 'X'). At each level of the tree, clusters are shown by unbroken lines of the TREECHAR= symbol with the FILLCHAR= symbol (the default is a blank) separating the clusters. The LEAFCHAR= symbol (the default character is a period) represents single-member clusters.

By default, the tree diagram is oriented with the height axis vertical and the object names at the top of the diagram. If the HORIZONTAL option is specified, then the height axis is horizontal and the object names are on the left.

ODS Table Names

PROC TREE assigns a name to each table it creates. You can use table names to refer to tables when using the Output Delivery System (ODS) to select tables and create output data sets. The name of PROC TREE's only table is listed in Table 127.2. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 127.2 ODS Tables Produced by PROC TREE

ODS Table Name	Description	Statement	Option
TreeListing	Listing of all nodes in the tree	PROC	LIST

Examples: TREE Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 127.1: Mammals' Teeth

(View the complete code for this example (treeex1.sas) in the example repository.)

The following statements produce a data set that contains the numbers of different kinds of teeth for a variety of mammals:

```
data teeth;
   title 'Mammals'' Teeth';
   input mammal & $16. v1-v8 @@;
   label v1='Right Top Incisors'
         v2='Right Bottom Incisors'
         v3='Right Top Canines'
         v4='Right Bottom Canines'
         v5='Right Top Premolars'
         v6='Right Bottom Premolars'
         v7='Right Top Molars'
         v8='Right Bottom Molars';
   datalines;
Brown Bat         2 3 1 1 3 3 3 3   Mole              3 2 1 0 3 3 3 3
Silver Hair Bat   2 3 1 1 2 3 3 3   Pigmy Bat         2 3 1 1 2 2 3 3
House Bat         2 3 1 1 1 2 3 3   Red Bat           1 3 1 1 2 2 3 3
Pika              2 1 0 0 2 2 3 3   Rabbit            2 1 0 0 3 2 3 3
Beaver            1 1 0 0 2 1 3 3   Groundhog         1 1 0 0 2 1 3 3
Gray Squirrel     1 1 0 0 1 1 3 3   House Mouse       1 1 0 0 0 0 3 3
Porcupine         1 1 0 0 1 1 3 3   Wolf              3 3 1 1 4 4 2 3
Bear              3 3 1 1 4 4 2 3   Raccoon           3 3 1 1 4 4 3 2
Marten            3 3 1 1 4 4 1 2   Weasel            3 3 1 1 3 3 1 2
Wolverine         3 3 1 1 4 4 1 2   Badger            3 3 1 1 3 3 1 2
River Otter       3 3 1 1 4 3 1 2   Sea Otter         3 2 1 1 3 3 1 2
Jaguar            3 3 1 1 3 2 1 1   Cougar            3 3 1 1 3 2 1 1
Fur Seal          3 2 1 1 4 4 1 1   Sea Lion          3 2 1 1 4 4 1 1
Grey Seal         3 2 1 1 3 3 2 2   Elephant Seal     2 1 1 1 4 4 1 1
Reindeer          0 4 1 0 3 3 3 3   Elk               0 4 1 0 3 3 3 3
Deer              0 4 0 0 3 3 3 3   Moose             0 4 0 0 3 3 3 3
;
```

The following statements use the CLUSTER procedure to cluster the mammals by average linkage and use ODS Graphics and the TREE procedure to produce a horizontal tree diagram that uses the average-linkage distance as its height axis:

```
ods graphics on;

proc cluster method=average std pseudo noeigen outtree=tree;
   id mammal;
```

```
   var v1-v8;
run;

proc tree horizontal;
   label _name_ = 'Animal';
run;
```

Output 127.1.1 displays the information about how the clusters are joined. For example, the cluster history shows that the mammals 'Wolf' and 'Bear' form cluster 29, which is merged with 'Raccoon' to form cluster 11.

Output 127.1.1 Output from PROC CLUSTER

Mammals' Teeth

The CLUSTER Procedure
Average Linkage Cluster Analysis

The data have been standardized to mean 0 and variance 1

Root-Mean-Square Total-Sample Standard Deviation	1

Root-Mean-Square Distance Between Observations	4

Output 127.1.1 *continued*

Cluster History

Number of Clusters	Clusters Joined		Freq	Pseudo F Statistic	Pseudo t-Squared	Norm RMS Distance	Tie
31	Beaver	Groundhog	2	.	.	0	T
30	Gray Squirrel	Porcupine	2	.	.	0	T
29	Wolf	Bear	2	.	.	0	T
28	Marten	Wolverine	2	.	.	0	T
27	Weasel	Badger	2	.	.	0	T
26	Jaguar	Cougar	2	.	.	0	T
25	Fur Seal	Sea Lion	2	.	.	0	T
24	Reindeer	Elk	2	.	.	0	T
23	Deer	Moose	2	.	.	0	
22	Pigmy Bat	Red Bat	2	281	.	0.2289	
21	CL28	River Otter	3	139	.	0.2292	
20	CL31	CL30	4	83.2	.	0.2357	T
19	Brown Bat	Silver Hair Bat	2	76.7	.	0.2357	T
18	Pika	Rabbit	2	73.2	.	0.2357	
17	CL27	Sea Otter	3	67.4	.	0.2462	
16	CL22	House Bat	3	62.9	1.7	0.2859	
15	CL21	CL17	6	47.4	6.8	0.3328	
14	CL25	Elephant Seal	3	45.0	.	0.3362	
13	CL19	CL16	5	40.8	3.5	0.3672	
12	CL15	Grey Seal	7	38.9	2.8	0.4078	
11	CL29	Raccoon	3	38.0	.	0.423	
10	CL18	CL20	6	34.5	10.3	0.4339	
9	CL12	CL26	9	30.0	7.3	0.5071	
8	CL24	CL23	4	28.7	.	0.5473	
7	CL9	CL14	12	25.7	7.0	0.5668	
6	CL10	House Mouse	7	28.3	4.1	0.5792	
5	CL11	CL7	15	26.8	6.9	0.6621	
4	CL13	Mole	6	31.9	7.2	0.7156	
3	CL4	CL8	10	31.0	12.7	0.8799	
2	CL3	CL6	17	27.8	16.1	1.0316	
1	CL2	CL5	32	.	27.8	1.1938	

Output 127.1.2 shows the tree diagram produced by PROC CLUSTER.

Output 127.1.2 Dendrogram from PROC CLUSTER

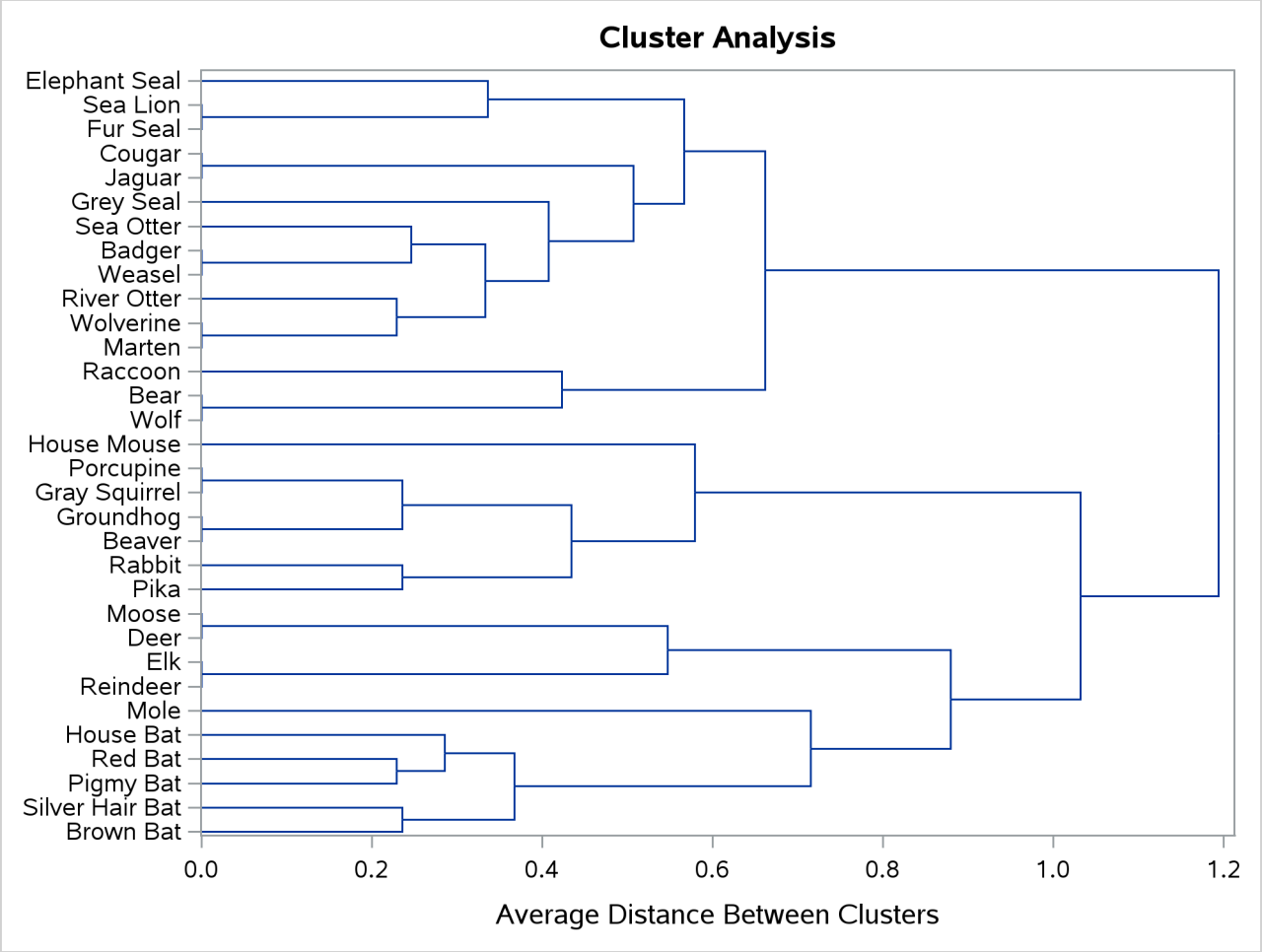

Output 127.1.3 shows the corresponding tree diagram produced by PROC TREE.

Output 127.1.3 Tree Diagram of Mammal Teeth Clusters

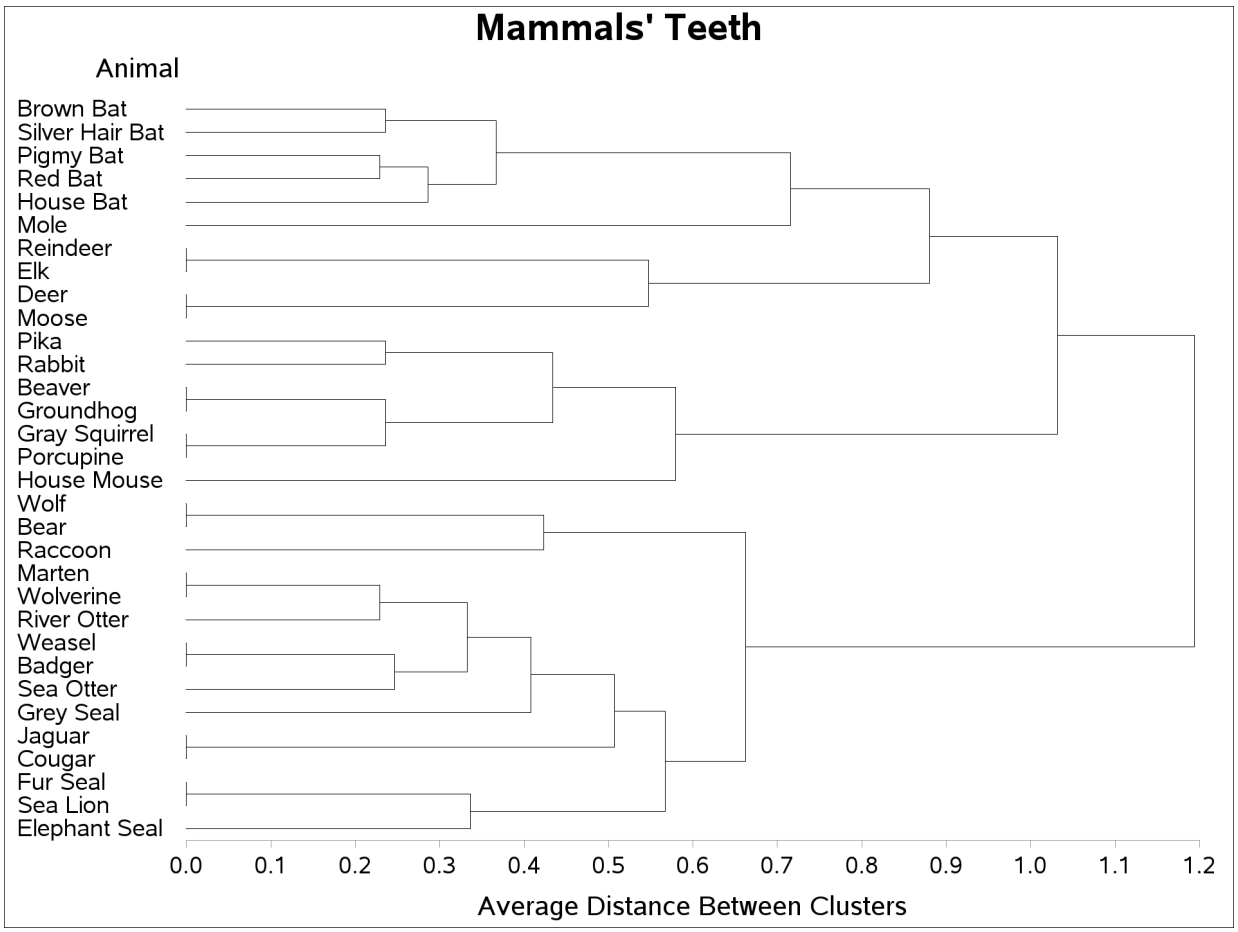

As you view the diagram in Output 127.1.3 from left to right, objects and clusters are progressively joined until a single, all-encompassing cluster is formed at the right (or root) of the tree. Clusters exist at each level of the diagram, and every vertical line connects leaves and branches into progressively larger clusters. For example, the five bats form a cluster at the 0.6 level, while the next cluster consists only of the mole. The mammals 'Reindeer', 'Elk', 'Deer', and 'Moose' form the next cluster at the 0.6 level, the mammals 'Pika' through 'House Mouse' are in the fourth cluster, the mammals 'Wolf', 'Bear', and 'Raccoon' form the fifth cluster, and the last cluster contains the mammals 'Marten' through 'Elephant Seal'.

The following statements create the same tree with line printer graphics in a vertical orientation:

```
options ps=40;
proc tree lineprinter;
run;
```

The line printer plot is not displayed.

The next statements sort the clusters at each branch in order of formation and use the number of clusters as the height axis:

```
proc tree sort height=n horizontal;
   label _name_ = 'Animal';
run;
```

The resulting tree is displayed in Output 127.1.4.

Output 127.1.4 PROC TREE with SORT and HEIGHT= Options

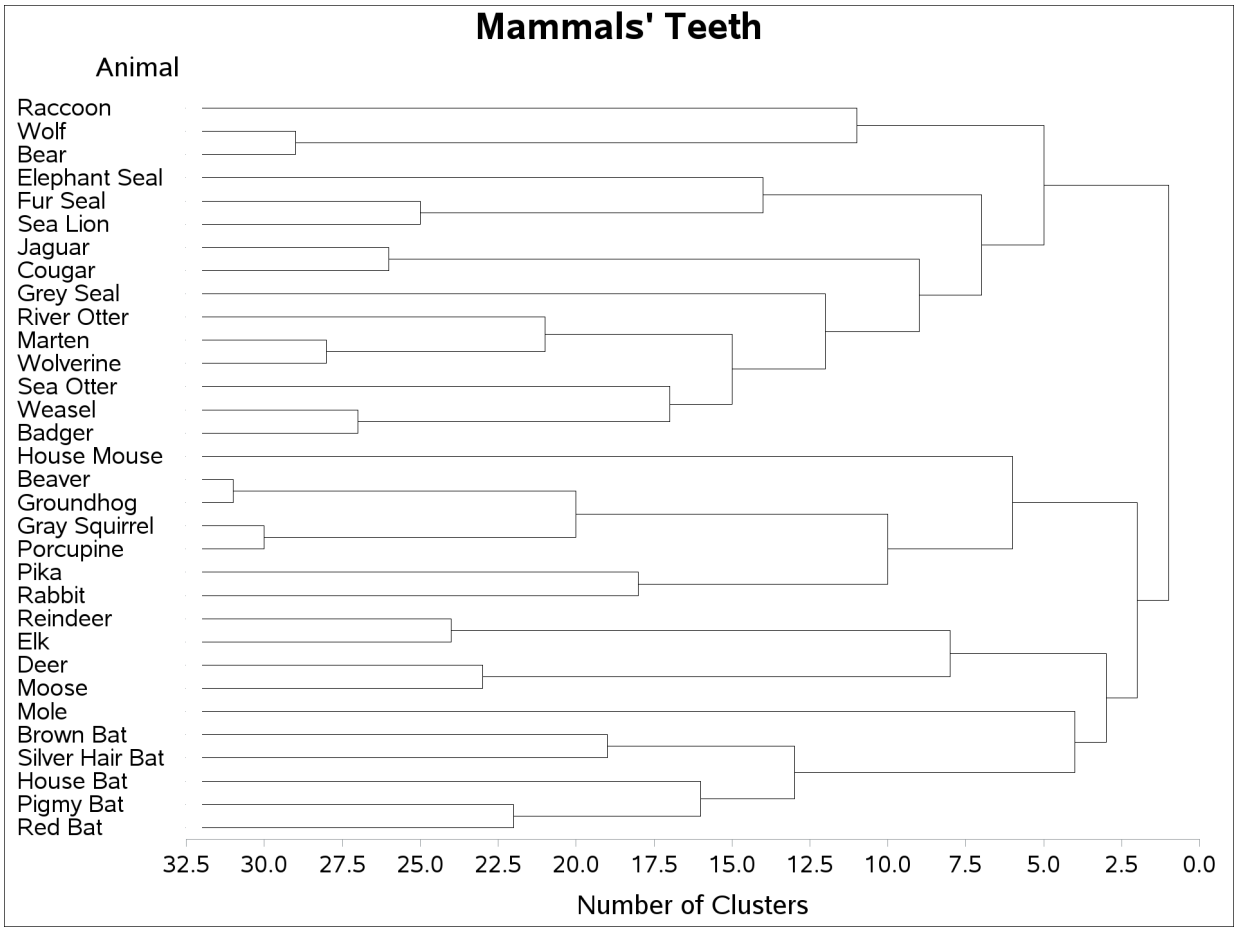

Because the CLUSTER procedure always produces binary trees, the number of internal (root and branch) nodes in the tree is one less than the number of leaves. Therefore 31 clusters are formed from the 32 mammals in the input data set. These are represented by the 31 vertical line segments in the tree diagram, each at a different value along the horizontal axis.

As you examine the tree from left to right, the first vertical line segment is where ‘Beaver’ and ‘Groundhog’ are clustered and the number of clusters is 31. The next cluster is formed from ‘Gray Squirrel’ and ‘Porcupine’. The third contains ‘Wolf’ and ‘Bear’. Note how the tree graphically displays the clustering order information that was presented in tabular form by the CLUSTER procedure in Output 127.1.1.

The same clusters as in Output 127.1.3 can be seen at the six-cluster level of the tree diagram in Output 127.1.4, although the SORT and HEIGHT= options make them appear in a different order.

The following statements create these six clusters and save the result in the output data set part:

```
proc tree noprint out=part nclusters=6;
   id mammal;
   copy v1-v8;
run;

proc sort;
   by cluster;
run;
```

PROC TREE with the NOPRINT option displays no output but creates an output data set that indicates the cluster to which each observation belongs at the six-cluster level in the tree. The following statements print the data set part, with the results shown in Output 127.1.5:

```
proc print label uniform;
   id mammal;
   var v1-v8;
   format v1-v8 1.;
   by cluster;
run;
```

Output 127.1.5 PROC TREE OUT= Data Set

Mammals' Teeth

CLUSTER=1

mammal	Right Top Incisors	Right Bottom Incisors	Right Top Canines	Right Bottom Canines	Right Top Premolars	Right Bottom Premolars	Right Top Molars	Right Bottom Molars
Beaver	1	1	0	0	2	1	3	3
Groundhog	1	1	0	0	2	1	3	3
Gray Squirrel	1	1	0	0	1	1	3	3
Porcupine	1	1	0	0	1	1	3	3
Pika	2	1	0	0	2	2	3	3
Rabbit	2	1	0	0	3	2	3	3
House Mouse	1	1	0	0	0	0	3	3

CLUSTER=2

mammal	Right Top Incisors	Right Bottom Incisors	Right Top Canines	Right Bottom Canines	Right Top Premolars	Right Bottom Premolars	Right Top Molars	Right Bottom Molars
Wolf	3	3	1	1	4	4	2	3
Bear	3	3	1	1	4	4	2	3
Raccoon	3	3	1	1	4	4	3	2

Output 127.1.5 *continued*

CLUSTER=3

mammal	Right Top Incisors	Right Bottom Incisors	Right Top Canines	Right Bottom Canines	Right Top Premolars	Right Bottom Premolars	Right Top Molars	Right Bottom Molars
Marten	3	3	1	1	4	4	1	2
Wolverine	3	3	1	1	4	4	1	2
Weasel	3	3	1	1	3	3	1	2
Badger	3	3	1	1	3	3	1	2
Jaguar	3	3	1	1	3	2	1	1
Cougar	3	3	1	1	3	2	1	1
Fur Seal	3	2	1	1	4	4	1	1
Sea Lion	3	2	1	1	4	4	1	1
River Otter	3	3	1	1	4	3	1	2
Sea Otter	3	2	1	1	3	3	1	2
Elephant Seal	2	1	1	1	4	4	1	1
Grey Seal	3	2	1	1	3	3	2	2

CLUSTER=4

mammal	Right Top Incisors	Right Bottom Incisors	Right Top Canines	Right Bottom Canines	Right Top Premolars	Right Bottom Premolars	Right Top Molars	Right Bottom Molars
Reindeer	0	4	1	0	3	3	3	3
Elk	0	4	1	0	3	3	3	3
Deer	0	4	0	0	3	3	3	3
Moose	0	4	0	0	3	3	3	3

CLUSTER=5

mammal	Right Top Incisors	Right Bottom Incisors	Right Top Canines	Right Bottom Canines	Right Top Premolars	Right Bottom Premolars	Right Top Molars	Right Bottom Molars
Pigmy Bat	2	3	1	1	2	2	3	3
Red Bat	1	3	1	1	2	2	3	3
Brown Bat	2	3	1	1	3	3	3	3
Silver Hair Bat	2	3	1	1	2	3	3	3
House Bat	2	3	1	1	1	2	3	3

CLUSTER=6

mammal	Right Top Incisors	Right Bottom Incisors	Right Top Canines	Right Bottom Canines	Right Top Premolars	Right Bottom Premolars	Right Top Molars	Right Bottom Molars
Mole	3	2	1	0	3	3	3	3

Example 127.2: Iris Data

(View the complete code for this example (treeex2.sas) in the example repository.)

Fisher (1936)'s iris data give sepal and petal dimensions for three different species of iris. The data, which are available in the Sashelp library, are clustered by *k*th-nearest-neighbor density linkage by using the CLUSTER procedure with K=8. Observations are identified by species ('Setosa', 'Versicolor', or 'Virginica') in the tree diagram, which is oriented with the height axis horizontal.

The following statements produce the results shown in Output 127.2.1:

```
title 'Fisher (1936) Iris Data';
ods graphics on;

proc cluster data=sashelp.iris method=twostage print=10
             outtree=tree k=8 noeigen;
   var SepalLength SepalWidth PetalLength PetalWidth;
   copy Species;
run;

proc tree data=tree horizontal lineprinter pages=1 maxh=10;
   id species;
run;
```

The PAGES=1 option specifies that the tree diagram extend over one page from tree to root. Since the HORIZONTAL option is also specified, the horizontal extent of the diagram is one page. The number of vertical pages required for the diagram is dictated by the number of leaves in the tree.

The MAXH=10 limits the values displayed on the height axis to a maximum of 10. This prunes the tree diagram so that only the portion from the leaves to level 10 is produced. The line printer plot is not displayed.

Output 127.2.1 Clustering of Fisher's Iris Data

Fisher (1936) Iris Data

The CLUSTER Procedure
Two-Stage Density Linkage Clustering

K = 8

Root-Mean-Square Total-Sample Standard Deviation	10.69224

Output 127.2.1 *continued*

Cluster History						
				Maximum Density in Each Cluster		
Number of Clusters	Clusters Joined		Freq	Normalized Fusion Density	Lesser	Greater	Tie
10	CL11	OB79	49	0.2879	0.1479	8.3678	
9	CL13	OB112	45	0.2802	0.2005	3.5156	
8	CL10	OB113	50	0.2699	0.1372	8.3678	
7	CL8	OB91	51	0.2586	0.1372	8.3678	
6	CL9	OB120	46	0.1412	0.0832	3.5156	
5	CL6	OB118	47	0.107	0.0605	3.5156	
4	CL5	OB110	48	0.0969	0.0541	3.5156	
3	CL4	OB135	49	0.0715	0.0370	3.5156	
2	CL7	CL3	100	2.6277	3.5156	8.3678	

3 modal clusters have been formed.

References

Duran, B. S., and Odell, P. L. (1974). *Cluster Analysis*. New York: Springer-Verlag.

Everitt, B. S. (1980). *Cluster Analysis*. 2nd ed. London: Heineman Educational Books.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Hand, D. J., Daly, F., Lunn, A. D., McConway, K. J., and Ostrowski, E. (1994). *A Handbook of Small Data Sets*. London: Chapman & Hall.

Hartigan, J. A. (1975). *Clustering Algorithms*. New York: John Wiley & Sons.

Johnson, S. C. (1967). "Hierarchical Clustering Schemes." *Psychometrika* 32:241–254.

Knuth, D. E. (1973). *Fundamental Algorithms*. Vol. 1 of The Art of Computer Programming. Reading, MA: Addison-Wesley.

Chapter 128
The TTEST Procedure

Contents

Overview: TTEST Procedure . . . **10804**
Getting Started: TTEST Procedure . . . **10805**
One-Sample *t* Test . . . 10805
Comparing Group Means . . . 10808
Syntax: TTEST Procedure . . . **10812**
PROC TTEST Statement . . . 10813
BOOTSTRAP Statement . . . 10822
BY Statement . . . 10824
CLASS Statement . . . 10825
FREQ Statement . . . 10825
PAIRED Statement . . . 10826
VAR Statement . . . 10826
WEIGHT Statement . . . 10827
Details: TTEST Procedure . . . **10827**
Input Data Set of Statistics . . . 10827
Missing Values . . . 10828
Computational Methods . . . 10828
Common Notation . . . 10828
Arithmetic and Geometric Means . . . 10829
Coefficient of Variation . . . 10829
One-Sample Design . . . 10829
Paired Design . . . 10832
Two-Independent-Sample Design . . . 10833
AB/BA Crossover Design . . . 10838
TOST Equivalence Test . . . 10839
Bootstrap Methods . . . 10840
Displayed Output . . . 10849
ODS Table Names . . . 10853
ODS Graphics . . . 10853
ODS Graph Names . . . 10854
Interpreting Graphs . . . 10857
Examples: TTEST Procedure . . . **10859**
Example 128.1: Using Summary Statistics to Compare Group Means . . . 10859
Example 128.2: One-Sample Comparison with the FREQ Statement . . . 10862
Example 128.3: Paired Comparisons . . . 10865
Example 128.4: AB/BA Crossover Design . . . 10870

Example 128.5: Equivalence Testing with Lognormal Data 10879
Example 128.6: Bootstrap with Two-Sample Design 10884
References **10898**

Overview: TTEST Procedure

The TTEST procedure performs *t* tests and computes confidence limits for one sample, paired observations, two independent samples, and the AB/BA crossover design. Two-sided, TOST (two one-sided test) equivalence, and upper and lower one-sided hypotheses are supported for means, mean differences, and mean ratios for either normal or lognormal data. PROC TTEST also computes bootstrap standard error, bias estimates, and confidence limits.

Table 128.1 summarizes the designs, analysis criteria, hypotheses, and distributional assumptions supported in the TTEST procedure, along with the syntax that you use to specify them.

Table 128.1 Features Supported in the TTEST Procedure

Feature	Syntax
Design	
One-sample	VAR statement
Paired	PAIRED statement
Two-independent-sample	CLASS statement, VAR statement
AB/BA crossover	VAR / CROSSOVER=
Analysis Criterion	
Mean difference	PROC TTEST TEST=DIFF
Mean ratio	PROC TTEST TEST=RATIO
Hypothesis	
Two-sided	PROC TTEST SIDES=2
Equivalence	PROC TTEST TOST (< *lower* , > *upper*)
Lower one-sided	PROC TTEST SIDES=L
Upper one-sided	PROC TTEST SIDES=U
Distribution	
Normal	PROC TTEST DIST=NORMAL
Lognormal	PROC TTEST DIST=LOGNORMAL
Empirical, resampled with replacement	BOOTSTRAP statement

FREQ and WEIGHT statements are available. Data can be input in the form of observations or, in certain cases, summary statistics. Output includes summary statistics; confidence limits for means, standard deviations, and coefficients of variation; hypothesis tests; bootstrap analyses; and a variety of graphical displays for both the input data set and bootstrap samples, including histograms, densities, box plots, confidence intervals, Q-Q plots, profiles, agreement plots, and correlation plots.

PROC TTEST uses ODS Graphics to create graphs as part of its output. For general information about ODS

Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the statistical graphics available with the TTEST procedure, see the PLOTS option in the PROC TTEST statement and the section "ODS Graphics" on page 10853.

Getting Started: TTEST Procedure

One-Sample *t* Test

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/ttegs1.sas.)

A one-sample *t* test can be used to compare a sample mean to a given value. This example, taken from Huntsberger and Billingsley (1989, p. 290), tests whether the mean length of a certain type of court case is more than 80 days by using 20 randomly chosen cases. The data are read by the following DATA step:

```
data time;
   input time @@;
   datalines;
 43  90  84  87  116   95  86   99   93  92
121  71  66  98   79  102  60  112  105  98
;
```

The only variable in the data set, time, is assumed to be normally distributed. The trailing at signs (@@) indicate that there is more than one observation on a line. The following statements invoke PROC TTEST for a one-sample *t* test:

```
ods graphics on;

proc ttest h0=80 plots(showh0) sides=u alpha=0.1;
   var time;
run;

ods graphics off;
```

The VAR statement indicates that the time variable is being studied, while the H0= option specifies that the mean of the time variable should be compared to the null value 80 rather than the default of 0. The PLOTS(SHOWH0) option requests that this null value be displayed on all relevant graphs. The SIDES=U option reflects the focus of the research question, namely whether the mean court case length is *greater than* 80 days, rather than *different than* 80 days (in which case you would use the default SIDES=2 option). The ALPHA=0.1 option requests 90% confidence intervals rather than the default 95% confidence intervals. The output is displayed in Figure 128.1.

Figure 128.1 One-Sample *t* Test Results

The TTEST Procedure

Variable: time

N	Mean	Std Dev	Std Err	Minimum	Maximum
20	89.8500	19.1456	4.2811	43.0000	121.0

Mean	90% CL Mean		Std Dev	90% CL Std Dev	
89.8500	84.1659	Infty	19.1456	15.2002	26.2374

DF	t Value	Pr > t
19	2.30	0.0164

Summary statistics appear at the top of the output. The sample size (N), mean, standard deviation, and standard error are displayed with the minimum and maximum values of the time variable. The 90% confidence limits for the mean and standard deviation are shown next. Due to the SIDES=U option, the interval for the mean is an upper one-sided interval with a finite lower bound (84.1659 days). The limits for the standard deviation are the equal-tailed variety, per the default CI=EQUAL option in the PROC TTEST statement. At the bottom of the output are the degrees of freedom, t statistic value, and p-value for the t test. At the 10% α level, this test indicates that the mean length of the court cases is significantly greater than from 80 days (t = 2.30, p = 0.0164).

The summary panel in Figure 128.2 shows a histogram with overlaid normal and kernel densities, a box plot, the 90% confidence interval for the mean, and the null value of 80 days.

Figure 128.2 Summary Panel

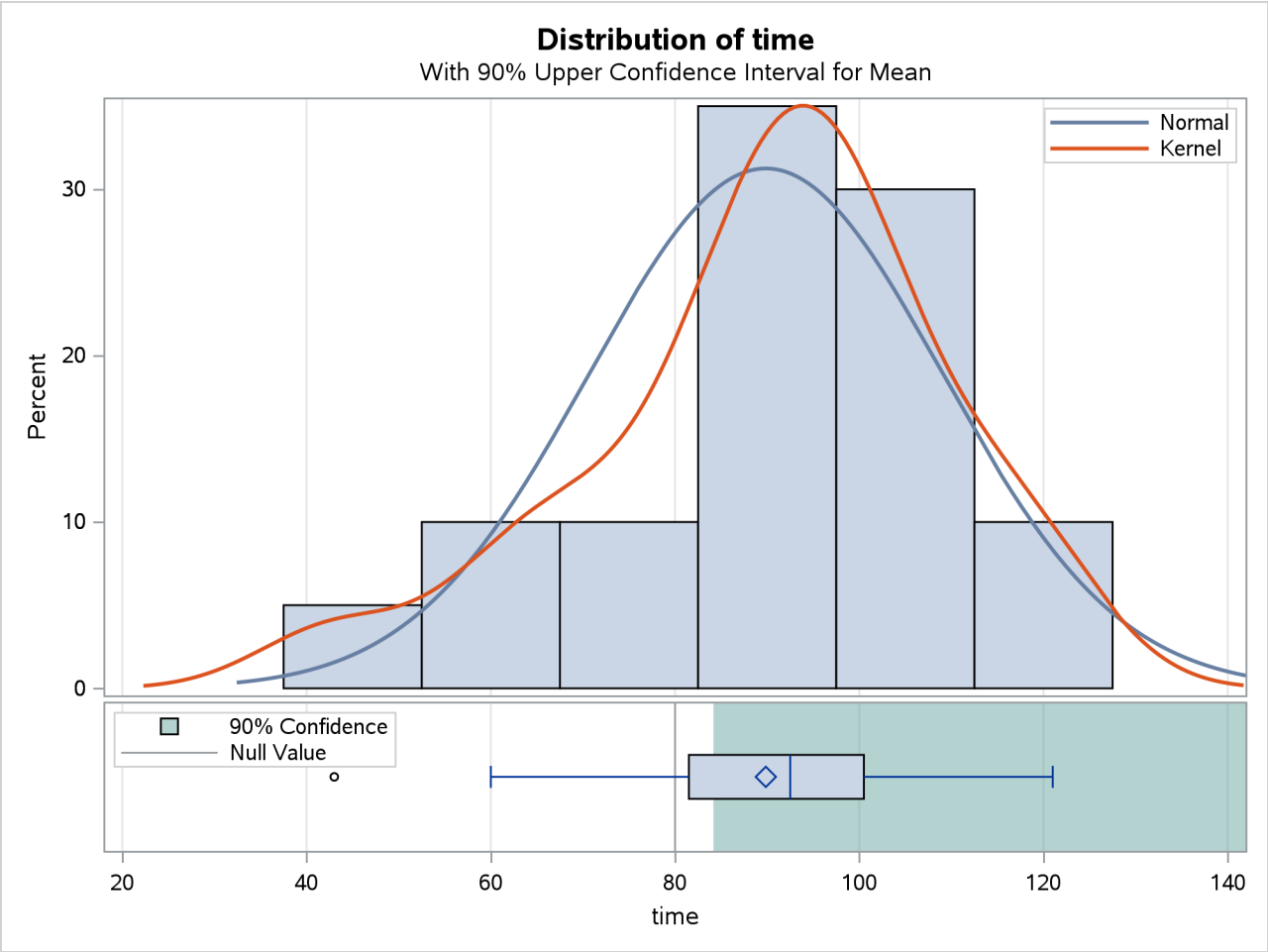

The confidence interval excludes the null value, consistent with the rejection of the null hypothesis at $\alpha = 0.1$.

The Q-Q plot in Figure 128.3 assesses the normality assumption.

Figure 128.3 Q-Q Plot

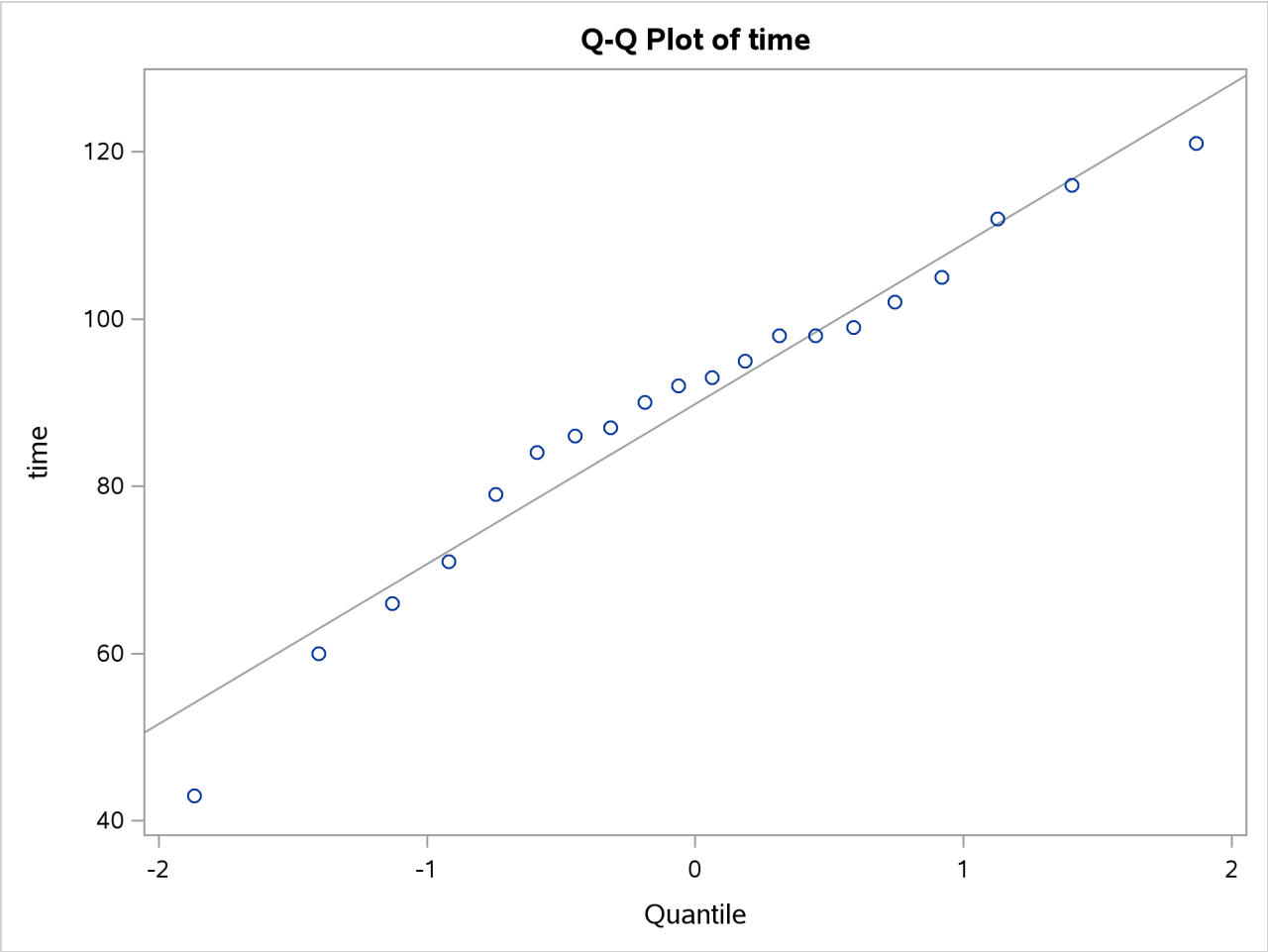

The curvilinear shape of the Q-Q plot suggests a possible slight deviation from normality. You could use the UNIVARIATE procedure with the NORMAL option to numerically check the normality assumptions.

Comparing Group Means

(View the complete code for this example at `https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/ttegs2.sas`.)

If you want to compare values obtained from two different groups, and if the groups are independent of each other and the data are normally or lognormally distributed in each group, then a group *t* test can be used. Examples of such group comparisons include the following:

- test scores for two third-grade classes, where one of the classes receives tutoring
- fuel efficiency readings of two automobile nameplates, where each nameplate uses the same fuel
- sunburn scores for two sunblock lotions, each applied to a different group of people

- political attitude scores of males and females

In the following example, the golf scores for males and females in a physical education class are compared. The sample sizes from each population are equal, but this is not required for further analysis. The scores are thought to be approximately normally distributed within gender. The data are read by the following statements:

```
data scores;
   input Gender $ Score @@;
   datalines;
f 75  f 76  f 80  f 77  f 80  f 77  f 73
m 82  m 80  m 85  m 85  m 78  m 87  m 82
;
```

The dollar sign ($) following Gender in the INPUT statement indicates that Gender is a character variable. The trailing at signs (@@) enable the procedure to read more than one observation per line.

You can use a group *t* test to determine whether the mean golf score for the men in the class differs significantly from the mean score for the women. If you also suspect that the distributions of the golf scores of males and females have unequal variances, then you might want to specify the COCHRAN option in order to use the Cochran approximation (in addition to the Satterthwaite approximation, which is included by default). The following statements invoke PROC TTEST for the case of unequal variances, along with both types of confidence limits for the pooled standard deviation.

```
ods graphics on;

proc ttest cochran ci=equal umpu;
   class Gender;
   var Score;
run;

ods graphics off;
```

The CLASS statement contains the variable that distinguishes the groups being compared, and the VAR statement specifies the response variable to be used in calculations. The COCHRAN option produces *p*-values for the unequal variance situation by using the Cochran and Cox (1950) approximation. Equal-tailed and uniformly most powerful unbiased (UMPU) confidence intervals for σ are requested by the CI= option. Output from these statements is displayed in Figure 128.4 through Figure 128.7.

Figure 128.4 Simple Statistics

The TTEST Procedure

Variable: Score

Gender	Method	N	Mean	Std Dev	Std Err	Minimum	Maximum
f		7	76.8571	2.5448	0.9619	73.0000	80.0000
m		7	82.7143	3.1472	1.1895	78.0000	87.0000
Diff (1-2)	Pooled		-5.8571	2.8619	1.5298		
Diff (1-2)	Satterthwaite		-5.8571		1.5298		

Simple statistics for the two populations being compared, as well as for the difference of the means between

the populations, are displayed in Figure 128.4. The Gender column indicates the population that corresponds to the statistics in that row, and the Method column indicates the method for estimating the standard deviation, either pooled (assuming equal variances for males and females) or Satterthwaite (assuming unequal variances). The sample size (N), mean, standard deviation, standard error, and minimum and maximum values are displayed.

Confidence limits for means and standard deviations are shown in Figure 128.5.

Figure 128.5 Confidence Limits

Gender	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev		95% UMPU CL Std Dev	
f		76.8571	74.5036	79.2107	2.5448	1.6399	5.6039	1.5634	5.2219
m		82.7143	79.8036	85.6249	3.1472	2.0280	6.9303	1.9335	6.4579
Diff (1-2)	**Pooled**	-5.8571	-9.1902	-2.5241	2.8619	2.0522	4.7242	2.0019	4.5727
Diff (1-2)	**Satterthwaite**	-5.8571	-9.2064	-2.5078					

For the mean difference, both pooled and Satterthwaite 95% intervals are shown. Equal-tailed and UMPU confidence limits are shown for the standard deviation under the assumption of equal variances.

The test statistics, associated degrees of freedom, and p-values are displayed in Figure 128.6.

Figure 128.6 t Tests

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	12	-3.83	0.0024
Satterthwaite	Unequal	11.496	-3.83	0.0026
Cochran	Unequal	6	-3.83	0.0087

The Method column denotes which t test is being used for that row, and the Variances column indicates what assumption about variances is being made. The pooled test assumes that the two populations have equal variances and uses degrees of freedom $n_1 + n_2 - 2$, where n_1 and n_2 are the sample sizes for the two populations. The remaining two tests do not assume that the populations have equal variances. The Satterthwaite test uses the Satterthwaite approximation for degrees of freedom, while the Cochran test uses the Cochran and Cox approximation for the p-value. All three tests result in highly significant p-values, supporting the conclusion of a significant difference between males' and females' golf scores.

The "Equality of Variances" test in Figure 128.7 reveals insufficient evidence of unequal variances (the Folded F statistic $F' = 1.53$, with $p = 0.6189$).

Figure 128.7 Tests of Equality of Variances

Equality of Variances				
Method	Num DF	Den DF	F Value	Pr > F
Folded F	6	6	1.53	0.6189

The summary panel in Figure 128.8 shows comparative histograms, normal and kernel densities, and box plots, comparing the distribution of golf scores between genders.

Figure 128.8 Summary Panel

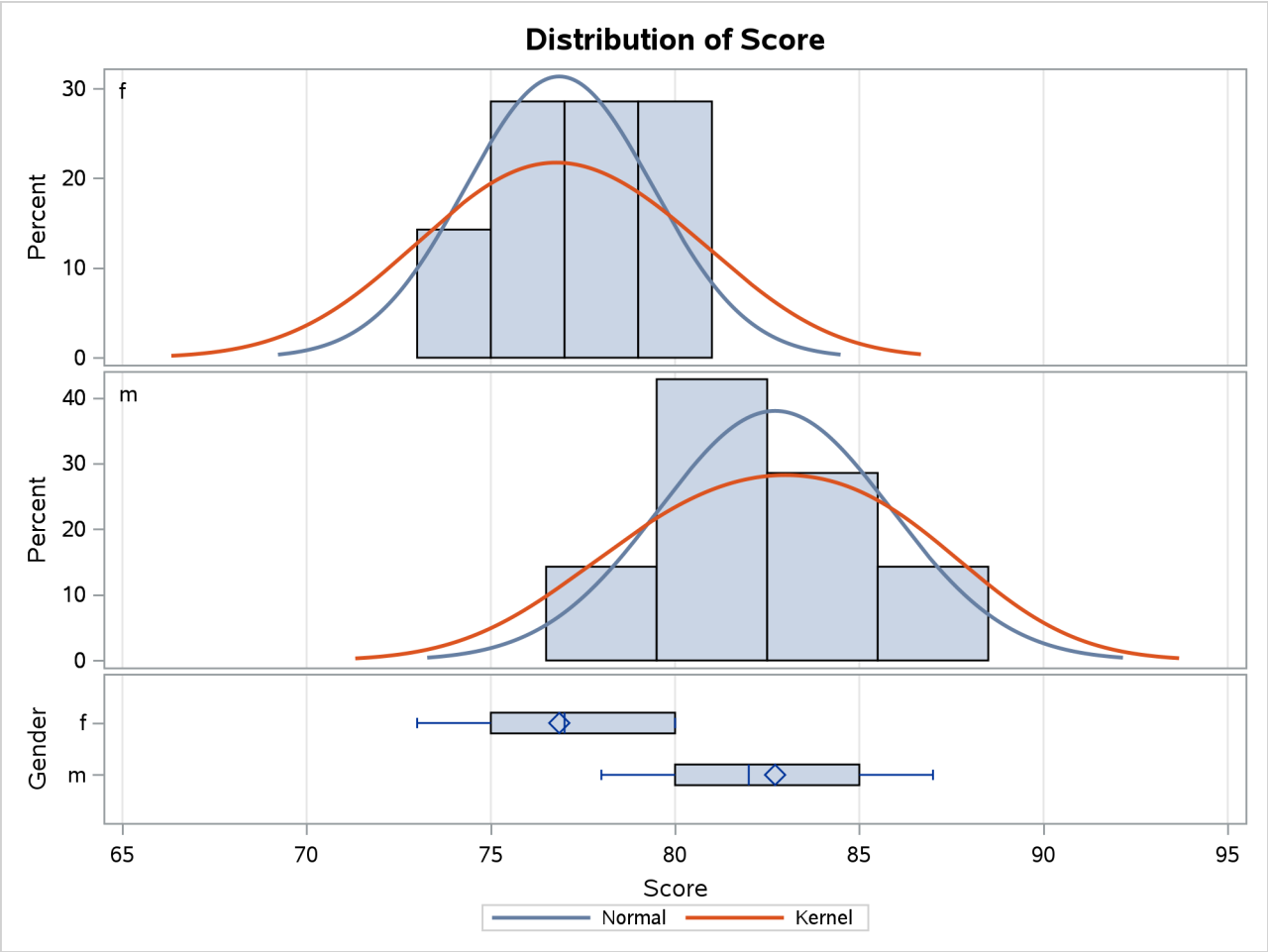

The Q-Q plots in Figure 128.9 assess the normality assumption for each gender.

Figure 128.9 Q-Q Plot

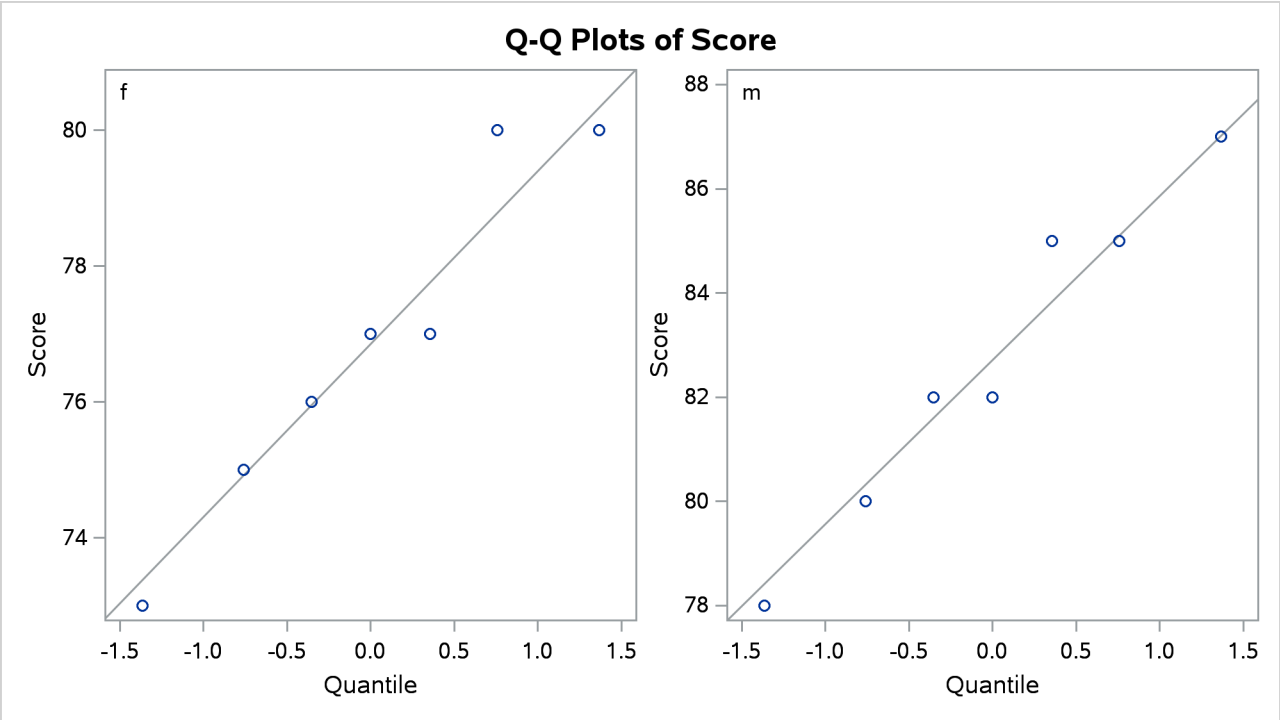

The plots for both males and females show no obvious deviations from normality. You can check the assumption of normality more rigorously by using PROC UNIVARIATE with the NORMAL option; if the assumption of normality is not reasonable, you should analyze the data with the nonparametric Wilcoxon rank sum test by using PROC NPAR1WAY.

Syntax: TTEST Procedure

The following statements are available in the TTEST procedure:

```
PROC TTEST <options> ;
    BOOTSTRAP </options> ;
    CLASS variable ;
    PAIRED variables ;
    BY variables ;
    VAR variables </options> ;
    FREQ variable ;
    WEIGHT variable ;
```

No statement can be used more than once. There is no restriction on the order of the statements after the PROC TTEST statement. The following sections describe the PROC TTEST statement and then describe the other statements in alphabetical order.

PROC TTEST Statement

PROC TTEST < *options* > **;**

The PROC TTEST statement invokes the TTEST procedure. Table 128.2 summarizes the *options* available in the PROC TTEST statement. The *options* are then described fully in alphabetical order.

Table 128.2 PROC TTEST Statement Options

Option	Description
Basic Options	
DATA=	Specifies input data set
ORDER=	Determines sort order of CLASS variable or CROSSOVER= treatment variables
Analysis Options	
ALPHA=	Specifies 1 – confidence level
DIST=	Specifies distributional assumption (normal or lognormal)
H0=	Specifies null value
SIDES=	Specifies number of sides and direction
TEST=	Specifies test criterion (difference or ratio)
TOST	Requests equivalence test and specifies bounds
Displayed Output	
CI=	Requests confidence interval for standard deviation or coefficient of variation
COCHRAN	Requests Cochran *t* test
PLOTS	Produces ODS statistical graphics
Output Ordering	
BYVAR	Groups results by variables that are specified in the PAIRED or VAR statement
NOBYVAR	Groups results by tables

You can specify the following *options*:

ALPHA=*p*

specifies that confidence intervals (except test-based mean confidence intervals when you specify the TOST option) are to be $100(1-p)\%$ confidence intervals, where $0 < p < 1$. When you specify the TOST option, the test-based mean confidence intervals are $100(1-2p)\%$ confidence intervals, where $0 < p < 0.5$. For the PLOTS=BOOTSTRAP(CORRELATION) option, the ALPHA= option specifies that the elliptical prediction region is to be a $100(1-p)\%$ prediction region. If $p \leq 0$ or $p \geq 1$, or if $p \geq 0.5$ when you specify the TOST option, then an error message is printed. By default, ALPHA=0.05.

BYVAR

groups the results by the variables that are specified in the PAIRED or VAR statement. The BYVAR option is enabled by default. Note that this represents a change from previous releases for how the results are grouped with respect to variables and tables. Prior to SAS 9.2, multiple variables were included in each table, similar to the new NOBYVAR option.

CI=EQUAL | UMPU | NONE

CL=EQUAL | UMPU | NONE

specifies whether a confidence interval is displayed for σ and, if so, what kind. You can specify one or more of the following values:

EQUAL specifies an equal-tailed confidence interval.

UMPU specifies an interval based on the uniformly most powerful unbiased test of $H_0: \sigma = \sigma_0$.

NONE requests that no confidence interval be displayed for σ.

The values EQUAL and UMPU together request that both types of confidence intervals be displayed. If the value NONE is specified with one or both of the values EQUAL and UMPU, NONE takes precedence. For more information, see the section "Two-Independent-Sample Design" on page 10833. By default, CI=EQUAL.

COCHRAN

requests the Cochran and Cox (1950) approximation of the probability level for the unequal variances situation. For more information, see the section "Two-Independent-Sample Design" on page 10833.

DATA=*SAS-data-set*

names the SAS data set for the procedure to use. By default, PROC TTEST uses the most recently created SAS data set. The input data set can contain summary statistics of the observations instead of the observations themselves. The number, mean, and standard deviation of the observations are required for each BY group (one sample and paired differences) or for each class within each BY group (two samples). For more information about the DATA= option, see the section "Input Data Set of Statistics" on page 10827.

DIST=LOGNORMAL | NORMAL

specifies the underlying distribution assumed for the data. You can specify the following values:

LOGNORMAL specifies that the underlying distribution is lognormal.

NORMAL specifies that the underlying distribution is normal.

By default, DIST=NORMAL, unless TEST=RATIO is specified, in which case the default is DIST=LOGNORMAL.

H0=*m*

requests tests against a null value of *m*, unless the TOST option is used, in which case *m* is merely used to derive the lower and upper equivalence bounds. For the crossover design, the value *m* applies for both treatment and period tests. By default, H0=0 when TEST=DIFF (or DIST=NORMAL for a one-sample design) and H0=1 when TEST=RATIO (or DIST=LOGNORMAL for a one-sample design).

NOBYVAR

includes all variables that are specified in the PAIRED or VAR statement together in each output table. If the NOBYVAR option is not specified, then the BYVAR option is enabled, grouping the results by the PAIRED and VAR variables.

ORDER=DATA | FORMATTED | FREQ | INTERNAL | MIXED

specifies the order in which to sort the levels of the classification variables (which are specified in the CLASS statement) and treatment variables (which are specified in the CROSSOVER= option in the VAR statement).

This option applies to the levels for all classification or treatment variables, except when you use the ORDER=FORMATTED option with numeric classification or treatment variables that have no explicit format. With this option, the levels of such variables are ordered by their internal value.

You can specify the following values:

Value of ORDER=	Levels Sorted By
DATA	Order of appearance in the input data set.
FORMATTED	External formatted value, except for numeric variables that have no explicit format, which are sorted by their unformatted (internal) value. The sort order is machine-dependent.
FREQ	Descending frequency count; levels that have the greatest number of observations come first in the order. In the event of a tie, ORDER=MIXED is used.
INTERNAL	Unformatted value. The sort order is machine-dependent.
MIXED	Same as ORDER=FORMATTED if the unformatted variable is character-valued; same as ORDER=INTERNAL otherwise (the unformatted variable is numeric-valued).

For FORMATTED and INTERNAL, the sort order is machine-dependent.

For more information about sort order, see the chapter on the SORT procedure in the *Base SAS Procedures Guide* and the discussion of BY-group processing in the "Grouping Data" section of *SAS Programmers Guide: Essentials*.

By default, ORDER=MIXED, which corresponds to the ordering in releases previous to SAS 9.2.

PLOTS <(*global-plot-options*)> <= *plot-request*<(*options*)>>

PLOTS <(*global-plot-options*)> <= (*plot-request*<(*options*)> <... *plot-request*<(*options*)>>)>

controls the plots produced through ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around the *plot-request*. Here are some examples:

```
plots=none
plots=(histogram boxplot interval qq profiles agreement)
plots(unpack)=summary
plots(showh0)=interval(type=pergroup)
plots=(summary(unpack) interval(type=period))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc ttest plots=all;
   var oxygen;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you do not specify the PLOTS option, then PROC TTEST produces a default set of plots. (**NOTE:** The graphical results are unavailable if your input data set contains summary statistics rather than observation values.)

For a one-sample design, the default plots are the following:

- summary plot (histogram with overlaid normal and kernel densities, box plot, and confidence interval band) for the input data set
- Q-Q plot for the input data set
- summary plot of bootstrap statistics, if you specify the BOOTSTRAP statement
- Q-Q plot of bootstrap statistics, if you specify the BOOTSTRAP statement
- correlation plot of bootstrap statistics, if you specify the BOOTSTRAP statement

For a two-independent-sample design, the default plots are the following:

- summary plot (comparative histograms with overlaid densities and box plots) for the input data set
- Q-Q plot for the input data set
- summary plot of bootstrap statistics, if you specify the BOOTSTRAP statement
- Q-Q plot of bootstrap statistics, if you specify the BOOTSTRAP statement
- correlation plot of bootstrap statistics, if you specify the BOOTSTRAP statement

For a paired design, the default plots are the following:

- summary plot (histogram, densities, box plot, and confidence interval) of the difference or ratio for the input data set
- Q-Q plot of the difference or ratio for the input data set
- profiles plot for the input data set
- agreement plot for the input data set
- summary plot of bootstrap statistics, if you specify the BOOTSTRAP statement
- Q-Q plot of bootstrap statistics, if you specify the BOOTSTRAP statement
- correlation plot of bootstrap statistics, if you specify the BOOTSTRAP statement

For a crossover design, the default plots are the following:

- comparative histograms with overlaid densities by treatment and period for the input data set
- comparative box plots by treatment and period for the input data set
- Q-Q plots by treatment and period for the input data set
- profiles over treatment plot for the input data set
- agreement of treatments plot for the input data set

For more detailed descriptions of plots, see the section "Interpreting Graphs" on page 10857.

You can specify the following *global-plot-options*:

ONLY
suppresses the default plots. Only plots that you specifically request are displayed.

SHOWH0
SHOWNULL
shows the null value (as specified by the H0= option in the PROC TTEST statement) in all relevant plots. For one-sample and paired designs, the null value can appear when you also specify the SUMMARY, BOOTSTRAP(SUMMARY), BOOTSTRAP(BOX), BOOTSTRAP(INTERVAL), BOX, or INTERVAL *plot-request*. For two-independent-sample designs and crossover designs, the null value can appear when you specify the BOOTSTRAP(INTERVAL) or INTERVAL *plot-request*. For crossover designs, the null value can appear only when you specify the INTERVAL *plot-request*.

UNPACKPANEL
UNPACK
suppresses paneling. By default, multiple plots can appear in some output panels. Specify this option to get each plot in a separate panel. You can specify PLOTS(UNPACKPANEL) to unpack the default plots.

You can specify the following *plot-requests*:

ALL
produces all appropriate plots. You can specify other *plot-requests* with ALL; for example, to request all plots and specify that intervals should be for the period difference in a crossover design, specify PLOTS=(ALL INTERVAL(TYPE=PERIOD)).

AGREEMENT < TYPE=*type* >
AGREEMENTPLOT < TYPE=*type* >
produces an agreement plot for the input data set. This plot is produced by default for paired and crossover designs, the only designs for which this option is valid.

For paired designs, the second response in each pair is plotted against the first response. For more information, see the section "Agreement Plots for Paired Designs" on page 10857.

For crossover designs, you can specify the following options:

TYPE=PERIOD
plots the response in the second period against the response in the first period. For more information, see the section "Period Agreement Plots for Crossover Designs" on page 10857.

TYPE=TREATMENT
plots the response associated with the second treatment against the response associated with the first treatment. For more information, see the section "Treatment Agreement Plots for Crossover Designs" on page 10857.

By default, TYPE=TREATMENT for crossover designs.

BOOTSTRAP < (*bootstrap-plot-requests*) >
BOOT < (*bootstrap-plot-requests*) >
controls the bootstrap-related plots that are produced through ODS Graphics.

You can specify the following *bootstrap-plot-requests*:

BOX
BOXPLOT
produces box plots of bootstrap mean and standard deviation.

A bootstrap confidence interval is shown as a band in the background if the specified value of the BOOTCI= option in the BOOTSTRAP statement is capable of producing confidence intervals for the bootstrap statistic.

For more information, see the section "Box Plots" on page 10858.

CORRELATION
CORR
produces a scatter plot of bootstrap standard deviation versus mean statistics with an overlaid elliptical prediction region.

This correlation plot is produced by default for one-sample, two-sample, and paired designs if you specify the BOOTSTRAP statement.

For more information, see the section "Bootstrap Correlation Plots" on page 10859.

HISTOGRAM
HIST
HISTDENS
produces histograms along with overlaid normal and kernel densities for bootstrap mean and standard deviation.

For more information, see the section "Histograms" on page 10858.

INTERVAL
INTERVALPLOT
produces plots of bootstrap confidence intervals for the mean and standard deviation.

For more information, see the section "Confidence Intervals" on page 10858.

QQ

QQPLOT

produces normal quantile-quantile (Q-Q) plots of the bootstrap mean and standard deviation. These plots are produced by default for one-sample, two-sample, and paired designs if you specify the BOOTSTRAP statement.

For more information, see the section "Q-Q Plots" on page 10859.

SUMMARY < UNPACK >

SUMMARYPLOT < UNPACK >

produces bootstrap histogram and box plots together in a single panel, where the plots share common X axes. These plots are produced by default for one-sample, paired, and two-independent-sample designs if you specify the BOOTSTRAP statement. For more information, see the documentation for BOX and HISTOGRAM *bootstrap-plot-requests*. You can specify the following option:

UNPACK

plots bootstrap histograms along with overlaid densities in one panel and bootstrap box plots (along with confidence interval bands, if applicable) in another panel. Note that specifying PLOTS(ONLY)=BOOTSTRAP(SUMMARY(UNPACK)) is exactly the same as specifying PLOTS(ONLY)=BOOTSTRAP(BOX HISTOGRAM).

By default, if you specify the BOOTSTRAP statement, then the SUMMARY, QQ, and CORRELATION *bootstrap-plot-requests* are included unless you exclude them by specifying the ONLY *global-plot-option*.

The BOOTSTRAP *plot-request* is ignored if you omit the BOOTSTRAP statement.

BOX

BOXPLOT

produces a box plot or comparative box plots for the input data set. A box plot is produced by default for crossover designs. For other designs, a box plot appears by default if you specify the SUMMARY or ALL *plot-request*.

For one-sample and paired designs, a confidence interval for the mean is shown as a band in the background, along with the equivalence bounds if the TOST option is used in the PROC TTEST statement.

For a two-independent-sample design, comparative box plots (one for each class) are shown. For a crossover design, comparative box plots for all four combinations of the two treatments and two periods are shown.

For more information, see the section "Box Plots" on page 10858.

HISTOGRAM

HIST

HISTDENS

produces a histogram or comparative histograms along with overlaid normal and kernel densities for the input data set. A histogram is produced by default for crossover designs. For other designs, it appears by default if you specify the SUMMARY or ALL *plot-request*.

For one-sample and paired designs, the histogram and densities are based on the test criterion (which is the mean difference or ratio for a paired design). For a two-independent-sample design, comparative histograms (one for each class) are shown. For a crossover design, histograms for all four combinations of the two treatments and two periods are shown.

For more information, see the section "Histograms" on page 10858.

INTERVAL < TYPE=*type* >

INTERVALPLOT < TYPE=*type* >

produces plots of confidence interval for means of the input data set.

For a two-independent-sample design, you can specify one of the following options:

TYPE=PERGROUP

shows two separate two-sided confidence intervals, one for each class. You cannot use this option along with the SHOWH0 *global-plot-option*.

TYPE=TEST

shows pooled and Satterthwaite confidence intervals.

By default, TYPE=TEST for two-independent-sample designs.

For a crossover design, you can specify the following options:

TYPE=PERGROUP

shows four separate two-sided intervals, one for each treatment-by-period combination. You cannot use this option along with the SHOWH0 *global-plot-option*.

TYPE=PERIOD

shows pooled and Satterthwaite confidence intervals for the period difference or ratio. This option is invalid if you specify the IGNOREPERIOD option in the VAR statement.

TYPE=TREATMENT

shows pooled and Satterthwaite confidence intervals for the treatment difference or ratio.

By default, TYPE=TREATMENT for crossover designs.

For more information, see the section "Confidence Intervals" on page 10858.

NONE

suppresses all plots.

PROFILES < TYPE=*type* >

PROFILESPLOT < TYPE=*type* >

produces a profiles plot for the input data set. This plot is produced by default for paired and crossover designs, the only designs for which this option is valid.

For paired designs, a line is drawn for each observation from left to right connecting the first response to the second response. For more information, see the section "Profiles for Paired Designs" on page 10858.

For crossover designs, you can specify one of the following options:

TYPE=PERIOD
shows response profiles over period, connecting the first period on the left to the second period on the right for each subject. For more information, see the section "Profiles over Period for Crossover Designs" on page 10858.

TYPE=TREATMENT
shows response profiles over treatment values, connecting the first treatment on the left to the second treatment on the right for each observation. For more information, see the section "Profiles over Treatment for Crossover Designs" on page 10859.

By default, TYPE=TREATMENT for crossover designs.

QQ
QQPLOT
produces a normal quantile-quantile (Q-Q) plot for the input data set. This plot is produced by default for all designs.

For two-sample designs, separate plots are shown for each class in a single panel. For crossover design, separate plots are shown for each treatment-by-period combination in a single panel.

For more information, see the section "Q-Q Plots" on page 10859.

SUMMARY < UNPACK >
SUMMARYPLOT < UNPACK >
produces histogram and box plots for the input data set together in a single panel, where the plots share common X axes. These plots are produced by default for one-sample, paired, and two-independent-sample designs, the only designs for which this option is valid. For more information, see the documentation for the BOX and HISTOGRAM *plot-requests*. You can specify the following option:

UNPACK
plots histograms along with overlaid densities in one panel and box plots (along with confidence interval bands, for one-sample and paired designs) in another panel. Note that specifying PLOTS(ONLY)=SUMMARY(UNPACK) is exactly the same as specifying PLOTS(ONLY)=(BOX HISTOGRAM).

SIDES=2 | L | U
SIDED=2 | L | U
SIDE=2 | L | U
specifies the number of sides (or tails) and direction of the statistical tests and test-based confidence intervals. The values are interpreted as follows:

2 specifies two-sided tests and confidence intervals.

L specifies lower one-sided tests (in which the alternative hypothesis indicates a parameter value less than the null value) and lower one-sided confidence intervals between minus infinity and the upper confidence limit.

U specifies upper one-sided tests (in which the alternative hypothesis indicates a parameter value greater than the null value) and upper one-sided confidence intervals between the lower confidence limit and infinity.

By default, SIDES=2.

TEST=DIFF | RATIO

specifies the test criterion. This option is ignored for one-sample designs. You can specify the following values:

DIFF tests the difference of means.

RATIO tests the ratio of means.

By default, TEST=DIFF, unless you specify DIST=LOGNORMAL, in which case the default is TEST=RATIO.

TOST (< *lower* **, >** *upper* **)**

performs Schuirmann's TOST equivalence test. The *upper* equivalence bound must be specified. If TEST=DIFF, then the default value for the *lower* equivalence bound is $2m-$*upper*, where m is the value of the H0= option. If TEST=RATIO, then the default value for *lower* is m / *upper*.

BOOTSTRAP Statement

BOOTSTRAP < / *options* > **;**

The BOOTSTRAP statement requests bootstrap standard error, bias estimates, and confidence intervals.

These bootstrap statistics are currently available only for one-sample, paired, and two-sample designs and only for analyses that assume normal data—although the bootstrap methods themselves do not necessarily assume normality.

Bootstrap results are unavailable if you specify the TEST=RATIO, DIST=LOGNORMAL, or TOST option in the PROC TTEST statement; if you specify the CROSSOVER= option in the VAR statement; or if you specify the WEIGHT statement. They are also unavailable if your input data set contains summary statistics rather than raw observed values.

The sample statistics for which bootstrap standard error, bias estimates, and confidence intervals are provided are as follows:

- For a one-sample design: mean and standard deviation of the observations
- For a paired design: mean and standard deviation of the paired difference—that is, the difference between the first and second members of an observation pair
- For a two-sample design: mean, pooled standard deviation, and unpooled standard deviation of the class difference—that is, the difference between an observation from the first class and an observation from the second class

For more information about how these statistics are computed, see the section "Statistics That Are Resampled" on page 10844.

You can specify the BOOTSTRAP *plot-request* option in the PROC TTEST statement to request plots that are based on bootstrap samples.

Summary of Options

Table 128.4 summarizes the *options* available in the BOOTSTRAP statement.

Table 128.4 Summary of Options in BOOTSTRAP Statement

Option	Description
BOOTCI=	Produces bootstrap confidence intervals for the mean and variability parameter estimates
BOOTDATA=	Specifies the bootstrap output data set
NSAMPLES=	Specifies the number of bootstrap sample data sets (replicates)
SEED=	Provides the seed that initializes the random number stream

Dictionary of Options

BOOTCI < = < BC | BOOTT | EXPANDEDPERC | NORMAL | PERCENTILE | TBOOTSE >>

produces bootstrap-based confidence intervals. You can request the following types of bootstrap confidence interval:

BC produces bias-corrected percentile intervals.

BOOTT produces bootstrap *t* intervals, which use a traditional standard error estimate and quantiles of the bootstrap distribution of the *t* statistic.

EXPANDEDPERC produces percentile-based confidence intervals that include a narrowness bias adjustment.

NORMAL produces normal-based confidence intervals that use the bootstrap standard error estimate.

PERCENTILE produces percentile-based confidence intervals.

TBOOTSE produces *t*-based confidence intervals that use the bootstrap standard error estimate.

The default option is BOOTCI=BC.

Table 128.5 shows which analysis parameters are supported for each type of bootstrap confidence interval. A bullet enclosed in parentheses indicates that the confidence limits are the same as for the other method for the same parameter.

Table 128.5 Parameters That Are Supported for Each Type of Bootstrap Confidence Interval in the BOOTSTRAP Statement

			BOOTCI Type					
Design	**Parameter**	**Method**	BC	BOOTT	EXPERC	NORMAL	PERC	TBOOTSE
One-sample	Mean		•	•	•	•	•	•
	Std Dev		•				•	
Paired	Mean (1–2)		•	•	•	•	•	•
	Std Dev (1–2)		•				•	
Two-sample	Mean (1–2)	Pooled	•	•	•	•	•	•

Table 128.5 *continued*

			BOOTCI Type					
Design	**Parameter**	**Method**	BC	BOOTT	EXPERC	NORMAL	PERC	TBOOTSE
	Mean (1–2)	Satterthwaite	(•)	•	•	(•)	(•)	•
	Std Dev (1–2)	Pooled	•				•	
	Std Dev (1–2)	Satterthwaite	•				•	

All six types include confidence intervals for the mean parameter estimate on which the usual hypothesis test is based, for example, the mean for a one-sample design or the mean difference for a paired or two-sample design. For BOOTCI=BC and BOOTCI=PERCENTILE, confidence intervals based on the variability parameter estimate are also produced; these are based on the variability parameter estimate that is used to compute the standard error of the usual hypothesis test for the mean parameter, for example, the standard deviation for a one-sample design or the standard deviation of the difference for a paired or two-sample design.

The ALPHA= and SIDES= options in the PROC TTEST statement set the direction and level of significance that is used in constructing the bootstrap confidence intervals.

For more information about the bootstrap confidence intervals supported by PROC TTEST, see the section "Bootstrap Confidence Intervals" on page 10846.

BOOTDATA=*SAS-data-set*
specifies the SAS data set that contains the bootstrap sample data when you use a BOOTSTRAP statement. This data set has the number of observations that you specify in the NSAMPLES= option and contains the mean and standard deviation estimates that are calculated for each bootstrap sample.

NSAMPLES=*n*
specifies the number of bootstrap sample data sets (replicates). The value must be greater than 1. By default, NSAMPLES=10000.

SEED=*n*
provides the seed that initializes the random number stream for generating the bootstrap sample data sets (replicates). If you do not specify the SEED= value, or if you specify a value less than or equal to 0, the seed is generated from reading the time of day from the computer's clock.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC TTEST to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.

- Specify the NOTSORTED or DESCENDING option in the BY statement in the TTEST procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variable* **;**

A CLASS statement giving the name of the classification (or grouping) variable must accompany the PROC TTEST statement in the two-independent-sample case. It should be omitted for the one-sample, paired, and AB/BA crossover designs. If it is used without the VAR statement, all numeric variables in the input data set (except those that appear in the CLASS, BY, FREQ, or WEIGHT statement) are included in the analysis.

The classification variable must have two, and only two, levels. PROC TTEST divides the observations into the two groups for the *t* test by using the levels of this variable. You can use either a numeric or a character variable in the CLASS statement.

Classification levels are determined from the formatted values of the CLASS variable. Thus, you can use formats to define group levels. For more information, see the discussions of the FORMAT procedure, the FORMAT statement, formats, and informats in *SAS Formats and Informats: Reference*.

FREQ Statement

FREQ *variable* **;**

The *variable* in the FREQ statement identifies a variable that contains the frequency of occurrence of each observation. PROC TTEST treats each observation as if it appears *n* times, where *n* is the value of the FREQ variable for the observation. If the value is not an integer, only the integer portion is used. If the frequency value is less than 1 or is missing, the observation is not used in the analysis. When the FREQ statement is not specified, each observation is assigned a frequency of 1. The FREQ statement cannot be used if the DATA= data set contains statistics instead of the original observations.

PAIRED Statement

PAIRED *pair-lists* **;**

The *pair-lists* in the PAIRED statement identifies the variables to be compared in paired comparisons. You can use one or more *pair-lists*. Variables or lists of variables are separated by an asterisk (*) or a colon (:). The asterisk requests that each variable on the left be compared with each variable on the right. The colon requests that the first variable on the left be compared with the first on the right, the second on the left with the second on the right, and so on. The number of variables on the left must equal the number on the right when the colon is used. The differences are calculated by taking the variable on the left minus the variable on the right for both the asterisk and colon. A pair that is formed by a variable with itself is ignored. Use the PAIRED statement only for paired comparisons. The CLASS and VAR statements cannot be used with the PAIRED statement.

Examples of the use of the asterisk and the colon are shown in Table 128.6.

Table 128.6 PAIRED Statement in the TTEST Procedure

These PAIRED Statements	Yield These Comparisons
`PAIRED A*B;`	A-B
`PAIRED A*B C*D;`	A-B and C-D
`PAIRED (A B)*(C D);`	A-C, A-D, B-C, and B-D
`PAIRED (A B)*(C B);`	A-C, A-B, and B-C
`PAIRED (A1-A2)*(B1-B2);`	A1-B1, A1-B2, A2-B1, and A2-B2
`PAIRED (A1-A2):(B1-B2);`	A1-B1 and A2-B2

VAR Statement

VAR *variables* < / *options* > **;**

The VAR statement names the variables to be used in the analyses. One-sample comparisons are conducted when the VAR statement is used without the CROSSOVER= option or CLASS statement. Two-independent-sample comparisons are conducted when the VAR statement is used with a CLASS statement.

An AB/BA crossover analysis is conducted when the CROSSOVER= option is used in the VAR statement. In this case, you must specify an even number of variables. Each set of two variables represents the responses in the first and second periods of the AB/BA crossover design. For example, if you use the CROSSOVER= option and specify `VAR x1 x2 x3 x4`, then you will get two analyses. One analysis will have x1 as the period 1 response and x2 as the period 2 response. The other analysis will have x3 as the period 1 response and x4 as the period 2 response.

The VAR statement cannot be used with the PAIRED statement. If the VAR statement is omitted, all numeric variables in the input data set (except a numeric variable that appears in the BY, CLASS, FREQ, or WEIGHT statement) are included in the analysis.

You can specify the following *options* after a slash (/):

CROSSOVER= (*variable1 variable2* **)**

specifies the variables that represent the treatment applied in each of the two periods in an AB/BA crossover design. The treatment variables must have two, and only two, levels. For any particular observation, the levels for the two variables must be different, due to the restrictions of the AB/BA crossover design. You can use either numeric or character variables.

Treatment levels are determined from the formatted values of the variables. Thus, you can use formats to define the treatment levels. For more information, see the discussions of the FORMAT procedure, the FORMAT statement, formats, and informats in *SAS Formats and Informats: Reference*.

IGNOREPERIOD

ignores the period effect—that is, the period effect is assumed to be equal to 0 (if TEST=DIFF) or 1 (if TEST=RATIO). This assumption increases the degrees of freedom for the test of the treatment difference by 1 and is usually more powerful, but it risks incorrect results if there is actually a period effect.

WEIGHT Statement

WEIGHT *variable* **;**

The WEIGHT statement weights each observation in the input data set by the value of the *variable*. The values of the *variable* can be nonintegral, and they are not truncated. Observations that have negative, zero, or missing values for the *variable* are not used in the analyses. Each observation is assigned a weight of 1 when the WEIGHT statement is not used. The WEIGHT statement cannot be used with an input data set of summary statistics.

Details: TTEST Procedure

Input Data Set of Statistics

PROC TTEST accepts data containing either observation values or summary statistics. Observation values are supported for all analyses, whereas summary statistics are supported only for a subset of analyses. If the analysis involves the paired design, the AB/BA crossover design, or the lognormal distributional assumption (DIST=LOGNORMAL), then observation values must be used. The graphical results are unavailable if your input data set contains summary statistics rather than raw observed values.

PROC TTEST assumes that the DATA= data set contains statistics if it contains a character variable with name _TYPE_ or _STAT_. The TTEST procedure expects this character variable to contain the names of statistics. If both _TYPE_ and _STAT_ variables exist and are of type character, PROC TTEST expects _TYPE_ to contain the names of statistics including ‘N’, ‘MEAN’, and ‘STD’ for each BY group (or for each class within each BY group for two-sample *t* tests). If no ‘N’, ‘MEAN’, or ‘STD’ statistics exist, an error message is printed.

FREQ, WEIGHT, and PAIRED statements cannot be used with input data sets of statistics. BY, CLASS, and VAR statements are the same regardless of data set type. For paired comparisons, see the _DIF_ values

for the _TYPE_=T observations in output produced by the OUTSTATS= option in the PROC COMPARE statement (see the *Base SAS Procedures Guide*).

Missing Values

An observation is omitted from the calculations if it has a missing value for either the CLASS variable, a CROSSOVER= variable, a PAIRED variable, the variable to be tested (in a one-sample or two-independent-sample design), or either of the two response variables (in a crossover design). If more than one variable or pair of variables is listed in the VAR statement, a missing value in one variable or pair does not eliminate the observation from the analysis of other nonmissing variables or variable pairs.

Computational Methods

This section describes the computational formulas for the estimates, confidence limits, and tests for each analysis in the TTEST procedure. The first subsection defines some common notation. The second subsection discusses the distinction between arithmetic and geometric means. The third subsection explains the concept of the coefficient of variation. The next four subsections address the four supported designs (one-sample, paired, two-independent-sample, and AB/BA crossover). The content in each of those subsections is divided into separate discussions according to different values of the DIST= and TEST= options in the PROC TTEST statement. The next-to-last subsection describes TOST equivalence analyses. The last subsection discusses bootstrap methods.

Common Notation

Table 128.7 displays notation for some of the commonly used symbols.

Table 128.7 Common Notation

Symbol	Description
μ	Population value of (arithmetic) mean
μ_0	Null value of test (value of H0= option in PROC TTEST statement)
σ^2	Population variance
σ	Population value of standard deviation
γ	Population value of geometric mean
CV	Population value of coefficient of variation (ratio of population standard deviation and population arithmetic mean)
α	Value of ALPHA= option in PROC TTEST statement
$t_{p,\nu}$	pth percentile of t distribution with ν degrees of freedom (df)
F_{p,ν_1,ν_2}	pth percentile of F distribution with ν_1 numerator df and ν_2 denominator df
$\chi^2_{p,\nu}$	pth percentile of chi-square distribution with ν df

Arithmetic and Geometric Means

The *arithmetic mean* (more commonly called simply the *mean*) of the distribution of a random variable X is its expected value, $E(X)$. The arithmetic mean is the natural parameter of interest for a normal distribution because the distribution of the difference of normal random variables has a known normal distribution, and the arithmetic mean of a normal difference is equal to the difference of the individual arithmetic means. (No such convenient property holds for geometric means with normal data, with either differences or ratios.)

The usual estimate of an arithmetic mean is the sum of the values divided by the number of values:

$$\text{arithmetic mean} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

The *geometric mean* of the distribution of a random variable X is $\exp(E(\log(X))$, the exponentiation of the mean of the natural logarithm. The geometric mean is the natural parameter of interest for a lognormal distribution because the distribution of a ratio of lognormal random variables has a known lognormal distribution, and the geometric mean of a lognormal ratio is equal to the ratio of the individual geometric means. (No such convenient property holds for arithmetic means with lognormal data, with either differences or ratios.)

The usual estimate of a geometric mean is the product of the values raised to the power $1/n$, where n is the number of values:

$$\text{geometric mean} = \left(\prod_{i=1}^{n} y_i\right)^{\frac{1}{n}}$$

Coefficient of Variation

The *coefficient of variation* (abbreviated "CV") of the distribution of a random variable X is the ratio of the standard deviation to the (arithmetic) mean, or $\sqrt{\mathrm{Var}(X)}/E(X)$. Conceptually, it is a measure of the variability of X expressed in units corresponding to the mean of X.

For lognormal data, the CV is the natural measure of variability (rather than the standard deviation) because the CV is invariant to multiplication of a lognormal variable by a constant. For a two-independent-sample design, the assumption of equal CVs on a lognormal scale is analogous to the assumption of equal variances on the normal scale. When the CVs of two independent samples of lognormal data are assumed equal, the pooled estimate of variability is used.

One-Sample Design

Define the following notation:

$$\begin{aligned}
n^\star &= \text{number of observations in data set}\\
y_i &= \text{value of } i\text{th observation},\ \ i \in \{1,\ldots,n^\star\}\\
f_i &= \text{frequency of } i\text{th observation},\ \ i \in \{1,\ldots,n^\star\}\\
w_i &= \text{weight of } i\text{th observation},\ \ i \in \{1,\ldots,n^\star\}\\
n &= \text{sample size} = \sum_{i}^{n^\star} f_i
\end{aligned}$$

Normal Data (DIST=NORMAL)

The mean estimate $\bar{y}$, standard deviation estimate s, and standard error SE are computed as follows:

$$\bar{y} = \frac{\sum_i^{n^\star} f_i w_i y_i}{\sum_i^{n^\star} f_i w_i}$$

$$s = \left(\frac{\sum_i^{n^\star} f_i w_i (y_i - \bar{y})^2}{n-1} \right)^{\frac{1}{2}}$$

$$\mathrm{SE} = \frac{s}{\left(\sum_i^{n^\star} f_i w_i \right)^{\frac{1}{2}}}$$

The $100(1-\alpha)\%$ confidence interval for the mean μ is

$$\begin{aligned} \left(\bar{y} - t_{1-\frac{\alpha}{2},n-1}\mathrm{SE} \ , \ \bar{y} + t_{1-\frac{\alpha}{2},n-1}\mathrm{SE}\right), & \quad \text{SIDES=2} \\ \left(-\infty \ , \ \bar{y} + t_{1-\alpha,n-1}\mathrm{SE}\right), & \quad \text{SIDES=L} \\ \left(\bar{y} - t_{1-\alpha,n-1}\mathrm{SE} \ , \ \infty\right), & \quad \text{SIDES=U} \end{aligned}$$

The t value for the test is computed as

$$t = \frac{\bar{y} - \mu_0}{\mathrm{SE}}$$

The p-value of the test is computed as

$$p\text{-value} = \begin{cases} P\left(T > t^2\right), & \text{two-sided} \\ P\left(T < t\right), & \text{lower one-sided} \\ P\left(T > t\right), & \text{upper one-sided} \end{cases}$$

where T has a t distribution with $n-1$ degrees of freedom.

The equal-tailed confidence interval for the standard deviation (CI=EQUAL) is based on the acceptance region of the test of $H_0{:}\,\sigma = \sigma_0$ that places an equal amount of area $(\frac{\alpha}{2})$ in each tail of the chi-square distribution:

$$\left\{ \chi^2_{\frac{\alpha}{2},n-1} \le \frac{(n-1)s^2}{\sigma_0^2} \le \chi^2_{\frac{1-\alpha}{2},n-1} \right\}$$

The acceptance region can be algebraically manipulated to give the following $100(1-\alpha)\%$ confidence interval for σ^2:

$$\left(\frac{(n-1)s^2}{\chi^2_{1-\frac{\alpha}{2},n-1}} \ , \ \frac{(n-1)s^2}{\chi^2_{\frac{\alpha}{2},n-1}} \right)$$

Taking the square root of each side yields the $100(1-\alpha)\%$ CI=EQUAL confidence interval for σ:

$$\left(\left(\frac{(n-1)s^2}{\chi^2_{1-\frac{\alpha}{2},n-1}} \right)^{\frac{1}{2}} \ , \ \left(\frac{(n-1)s^2}{\chi^2_{\frac{\alpha}{2},n-1}} \right)^{\frac{1}{2}} \right)$$

The other confidence interval for the standard deviation (CI=UMPU) is derived from the uniformly most powerful unbiased test of $H_0: \sigma = \sigma_0$ (Lehmann 1986). This test has acceptance region

$$\left\{ c_1 \leq \frac{(n-1)s^2}{\sigma_0^2} \leq c_2 \right\}$$

where the critical values c_1 and c_2 satisfy

$$\int_{c_1}^{c_2} f_{n-1}(y)dy = 1 - \alpha$$

and

$$\int_{c_1}^{c_2} y f_{n-1}(y)dy = (n-1)(1-\alpha)$$

where $f_\nu(y)$ is the PDF of the chi-square distribution with ν degrees of freedom. This acceptance region can be algebraically manipulated to arrive at

$$P\left\{ \frac{(n-1)s^2}{c_2} \leq \sigma^2 \leq \frac{(n-1)s^2}{c_1} \right\} = 1 - \alpha$$

where c_1 and c_2 solve the preceding two integrals. To find the area in each tail of the chi-square distribution to which these two critical values correspond, solve $c_1 = \chi^2_{1-\alpha_2,n-1}$ and $c_2 = \chi^2_{\alpha_1,n-1}$ for α_1 and α_2; the resulting α_1 and α_2 sum to α. Hence, a $100(1-\alpha)\%$ confidence interval for σ^2 is given by

$$\left(\frac{(n-1)s^2}{\chi^2_{1-\alpha_2,n-1}} \ , \ \frac{(n-1)s^2}{\chi^2_{\alpha_1,n-1}} \right)$$

Taking the square root of each side yields the $100(1-\alpha)\%$ CI=UMPU confidence interval for σ:

$$\left(\left(\frac{(n-1)s^2}{\chi^2_{1-\alpha_2,n-1}} \right)^{\frac{1}{2}} \ , \ \left(\frac{(n-1)s^2}{\chi^2_{\alpha_1,n-1}} \right)^{\frac{1}{2}} \right)$$

Lognormal Data (DIST=LOGNORMAL)

The DIST=LOGNORMAL analysis is handled by log-transforming the data and null value, performing a DIST=NORMAL analysis, and then transforming the results back to the original scale. This simple technique is based on the properties of the lognormal distribution as discussed in Johnson, Kotz, and Balakrishnan (1994, Chapter 14).

Taking the natural logarithms of the observation values and the null value, define

$$z_i = \log(y_i) \ , \ i \in \{1, \ldots, n^\star\}$$
$$\gamma_0 = \log(\mu_0)$$

First, a DIST=NORMAL analysis is performed on $\{z_i\}$ with the null value γ_0, producing the mean estimate $\bar{z}$, the standard deviation estimate s_z, a t value, and a p-value. The geometric mean estimate $\hat{\gamma}$ and the CV estimate $\widehat{CV}$ of the original lognormal data are computed as follows:

$$\hat{\gamma} = \exp(\bar{z})$$
$$\widehat{CV} = \left(\exp(s_z^2) - 1\right)^{\frac{1}{2}}$$

The t value and p-value remain the same. The confidence limits for the geometric mean and CV on the original lognormal scale are computed from the confidence limits for the arithmetic mean and standard deviation in the DIST=NORMAL analysis on the log-transformed data, in the same way that $\hat{\gamma}$ is derived from $\bar{z}$ and $\widehat{CV}$ is derived from s_z.

Paired Design

Define the following notation:

$$
\begin{aligned}
n^\star &= \text{number of observations in data set} \\
y_{1i} &= \text{value of } i\text{th observation for first PAIRED variable, } \; i \in \{1, \ldots, n^\star\} \\
y_{2i} &= \text{value of } i\text{th observation for second PAIRED variable, } \; i \in \{1, \ldots, n^\star\} \\
f_i &= \text{frequency of } i\text{th observation, } \; i \in \{1, \ldots, n^\star\} \\
w_i &= \text{weight of } i\text{th observation, } \; i \in \{1, \ldots, n^\star\} \\
n &= \text{sample size} = \sum_i^{n^\star} f_i
\end{aligned}
$$

Normal Difference (DIST=NORMAL TEST=DIFF)

The analysis is the same as the analysis for the one-sample design in the section "Normal Data (DIST=NORMAL)" on page 10830 based on the differences

$$d_i = y_{1i} - y_{2i} \quad , \quad i \in \{1, \ldots, n^\star\}$$

Lognormal Ratio (DIST=LOGNORMAL TEST=RATIO)

The analysis is the same as the analysis for the one-sample design in the section "Lognormal Data (DIST=LOGNORMAL)" on page 10831 based on the ratios

$$r_i = y_{1i}/y_{2i} \quad , \quad i \in \{1, \ldots, n^\star\}$$

Normal Ratio (DIST=NORMAL TEST=RATIO)

The hypothesis $H_0\colon \mu_1/\mu_2 = \mu_0$, where μ_1 and μ_2 are the means of the first and second PAIRED variables, respectively, can be rewritten as $H_0\colon \mu_1 - \mu_0\mu_2 = 0$. The t value and p-value are computed in the same way as in the one-sample design in the section "Normal Data (DIST=NORMAL)" on page 10830 based on the transformed values

$$z_i = y_{1i} - \mu_0 y_{2i} \quad , \quad i \in \{1, \ldots, n^\star\}$$

Estimates and confidence limits are not computed for this situation.

Two-Independent-Sample Design

Define the following notation:

$$
\begin{aligned}
n_1^\star &= \text{number of observations at first class level} \\
n_2^\star &= \text{number of observations at second class level} \\
y_{1i} &= \text{value of } i\text{th observation at first class level, } \; i \in \{1, \ldots, n_1^\star\} \\
y_{2i} &= \text{value of } i\text{th observation at second class level, } \; i \in \{1, \ldots, n_2^\star\} \\
f_{1i} &= \text{frequency of } i\text{th observation at first class level, } \; i \in \{1, \ldots, n_1^\star\} \\
f_{2i} &= \text{frequency of } i\text{th observation at second class level, } \; i \in \{1, \ldots, n_2^\star\} \\
w_{1i} &= \text{weight of } i\text{th observation at first class level, } \; i \in \{1, \ldots, n_1^\star\} \\
w_{2i} &= \text{weight of } i\text{th observation at second class level, } \; i \in \{1, \ldots, n_2^\star\} \\
n_1 &= \text{sample size for first class level} = \sum_i^{n_1^\star} f_{1i} \\
n_2 &= \text{sample size for second class level} = \sum_i^{n_2^\star} f_{2i}
\end{aligned}
$$

Normal Difference (DIST=NORMAL TEST=DIFF)

Observations at the first class level are assumed to be distributed as $\mathrm{N}(\mu_1, \sigma_1^2)$, and observations at the second class level are assumed to be distributed as $\mathrm{N}(\mu_2, \sigma_2^2)$, where μ_1, μ_2, σ_1, and σ_2 are unknown.

The within-class-level mean estimates ($\bar{y}_1$ and $\bar{y}_2$), standard deviation estimates (s_1 and s_2), standard errors (SE_1 and SE_2), and confidence limits for means and standard deviations are computed in the same way as for the one-sample design in the section "Normal Data (DIST=NORMAL)" on page 10830.

The mean difference $\mu_1 - \mu_2 = \mu_d$ is estimated by

$$\bar{y}_d = \bar{y}_1 - \bar{y}_2$$

Under the assumption of equal variances ($\sigma_1^2 = \sigma_2^2$), the pooled estimate of the common standard deviation is

$$s_p = \left(\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2} \right)^{\frac{1}{2}}$$

The pooled standard error (the estimated standard deviation of $\bar{y}_d$ assuming equal variances) is

$$\mathrm{SE}_p = s_p \left(\frac{1}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} + \frac{1}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}} \right)^{\frac{1}{2}}$$

The pooled $100(1-\alpha)\%$ confidence interval for the mean difference μ_d is

$$
\begin{aligned}
\left(\bar{y}_d - t_{1-\frac{\alpha}{2}, n_1+n_2-2}\mathrm{SE}_p \;\; , \;\; \bar{y}_d + t_{1-\frac{\alpha}{2}, n_1+n_2-2}\mathrm{SE}_p\right), &\quad \text{SIDES=2} \\
\left(-\infty \;\; , \;\; \bar{y}_d + t_{1-\alpha, n_1+n_2-2}\mathrm{SE}_p\right), &\quad \text{SIDES=L} \\
\left(\bar{y}_d - t_{1-\alpha, n_1+n_2-2}\mathrm{SE}_p \;\; , \;\; \infty\right), &\quad \text{SIDES=U}
\end{aligned}
$$

The t value for the pooled test is computed as

$$t_p = \frac{\bar{y}_d - \mu_0}{\mathrm{SE}_p}$$

The p-value of the test is computed as

$$p\text{-value} = \begin{cases} P\left(T > t_p^2\right), & \text{two-sided} \\ P\left(T < t_p\right), & \text{lower one-sided} \\ P\left(T > t_p\right), & \text{upper one-sided} \end{cases}$$

where T has a t distribution with $n_1 + n_2 - 2$ degrees of freedom.

Under the assumption of unequal variances (the Behrens-Fisher problem), the unpooled standard error is computed as

$$\mathrm{SE}_u = \left(\frac{s_1^2}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} + \frac{s_2^2}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}} \right)^{\frac{1}{2}}$$

Satterthwaite's (1946) approximation for the degrees of freedom, extended to accommodate weights, is computed as

$$\mathrm{df}_u = \frac{\mathrm{SE}_u^4}{\frac{s_1^4}{(n_1-1)\left(\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}\right)^2} + \frac{s_2^4}{(n_2-1)\left(\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}\right)^2}}$$

The unpooled Satterthwaite $100(1-\alpha)\%$ confidence interval for the mean difference μ_d is

$$\begin{aligned} \left(\bar{y}_d - t_{1-\frac{\alpha}{2},\mathrm{df}_u}\mathrm{SE}_u \ , \ \bar{y}_d + t_{1-\frac{\alpha}{2},\mathrm{df}_u}\mathrm{SE}_u\right), & \quad \text{SIDES=2} \\ \left(-\infty \ , \ \bar{y}_d + t_{1-\alpha,\mathrm{df}_u}\mathrm{SE}_u\right), & \quad \text{SIDES=L} \\ \left(\bar{y}_d - t_{1-\alpha,\mathrm{df}_u}\mathrm{SE}_u \ , \ \infty\right), & \quad \text{SIDES=U} \end{aligned}$$

The t value for the unpooled Satterthwaite test is computed as

$$t_u = \frac{\bar{y}_d - \mu_0}{\mathrm{SE}_u}$$

The p-value of the unpooled Satterthwaite test is computed as

$$p\text{-value} = \begin{cases} P\left(T > t_u^2\right), & \text{two-sided} \\ P\left(T < t_u\right), & \text{lower one-sided} \\ P\left(T > t_u\right), & \text{upper one-sided} \end{cases}$$

where T has a t distribution with df_u degrees of freedom.

When the COCHRAN option is specified in the PROC TTEST statement, the Cochran and Cox (1950) approximation of the p-value of the t_u statistic is the value of p such that

$$t_u = \frac{\left(\frac{s_1^2}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}}\right) t_1 + \left(\frac{s_2^2}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}}\right) t_2}{\left(\frac{s_1^2}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}}\right) + \left(\frac{s_2^2}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}}\right)}$$

where t_1 and t_2 are the critical values of the t distribution corresponding to a significance level of p and sample sizes of n_1 and n_2, respectively. The number of degrees of freedom is undefined when $n_1 \neq n_2$. In general, the Cochran and Cox test tends to be conservative (Lee and Gurland 1975).

The $100(1-\alpha)\%$ CI=EQUAL and CI=UMPU confidence intervals for the common population standard deviation σ assuming equal variances are computed as discussed in the section "Normal Data (DIST=NORMAL)" on page 10830 for the one-sample design, except replacing s^2 by s_p^2 and $(n-1)$ by (n_1+n_2-1).

The folded form of the F statistic, F', tests the hypothesis that the variances are equal (Steel and Torrie 1980), where

$$F' = \frac{\max(s_1^2, s_2^2)}{\min(s_1^2, s_2^2)}$$

A test of F' is a two-tailed F test because you do not specify which variance you expect to be larger. The p-value (Steel and Torrie 1980) is equal-tailed and is computed as

$$\begin{aligned} p\text{-value} &= 2P\left(F_{\mathrm{df}_a,\mathrm{df}_b} > F'\right) \\ &= \begin{cases} P\left(F_{\mathrm{df}_1,\mathrm{df}_2} > s_1^2/s_2^2\right) + P\left(F_{\mathrm{df}_2,\mathrm{df}_1} \leq s_2^2/s_1^2\right), & s_1^2/s_2^2 \geq 1 \\ P\left(F_{\mathrm{df}_1,\mathrm{df}_2} \leq s_1^2/s_2^2\right) + P\left(F_{\mathrm{df}_2,\mathrm{df}_1} > s_2^2/s_1^2\right), & s_1^2/s_2^2 < 1 \end{cases} \end{aligned}$$

where $F_{i,j}$ has an F distribution with i numerator degrees of freedom and j denominator degrees of freedom and df_1, df_2, df_a, and df_b are the degrees of freedom that correspond to s_1^2, s_1^2, $\max(s_1^2, s_2^2)$, and $\min(s_1^2, s_2^2)$, respectively.

Note that the p-value is similar to the probability $p^\star$ of a greater F' value under the null hypothesis that $\sigma_1^2 = \sigma_2^2$,

$$p^\star = \begin{cases} P\left(F_{\mathrm{df}_1,\mathrm{df}_2} > s_1^2/s_2^2\right) + P\left(F_{\mathrm{df}_1,\mathrm{df}_2} \leq s_2^2/s_1^2\right), & s_1^2/s_2^2 \geq 1 \\ P\left(F_{\mathrm{df}_1,\mathrm{df}_2} \leq s_1^2/s_2^2\right) + P\left(F_{\mathrm{df}_1,\mathrm{df}_2} > s_2^2/s_1^2\right), & s_1^2/s_2^2 < 1 \end{cases}$$

The F' test is not very robust to violations of the assumption that the data are normally distributed, and thus it is not recommended without confidence in the normality assumption.

Lognormal Ratio (DIST=LOGNORMAL TEST=RATIO)

The DIST=LOGNORMAL analysis is handled by log-transforming the data and null value, performing a DIST=NORMAL analysis, and then transforming the results back to the original scale. See the section "Normal Data (DIST=NORMAL)" on page 10830 for the one-sample design for details on how the DIST=NORMAL computations for means and standard deviations are transformed into the DIST=LOGNORMAL results for geometric means and CVs. As mentioned in the section "Coefficient of Variation" on page 10829, the assumption of equal CVs on the lognormal scale is analogous to the assumption of equal variances on the normal scale.

Normal Ratio (DIST=NORMAL TEST=RATIO)

The distributional assumptions, equality of variances test, and within-class-level mean estimates ($\bar{y}_1$ and $\bar{y}_2$), standard deviation estimates (s_1 and s_2), standard errors (SE_1 and SE_2), and confidence limits for means and standard deviations are the same as in the section "Normal Difference (DIST=NORMAL TEST=DIFF)" on page 10833 for the two-independent-sample design.

The mean ratio $\mu_1/\mu_2 = \mu_r$ is estimated by

$$\hat{\mu}_r = \bar{y}_1/\bar{y}_2$$

No estimates or confidence intervals for the ratio of standard deviations are computed.

Under the assumption of equal variances ($\sigma_1^2 = \sigma_2^2$), the pooled confidence interval for the mean ratio is the Fieller (1954) confidence interval, extended to accommodate weights. Let

$$a_p = \frac{s_p^2 t^2_{1-\frac{\alpha}{2},n_1+n_2-2}}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}} - \bar{y}_2^2$$
$$b_p = \bar{y}_1 \bar{y}_2$$
$$c_p = \frac{s_p^2 t^2_{1-\frac{\alpha}{2},n_1+n_2-2}}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} - \bar{y}_1^2$$

where s_p is the pooled standard deviation defined in the section "Normal Difference (DIST=NORMAL TEST=DIFF)" on page 10833 for the two-independent-sample design. If $a_p \geq 0$ (which occurs when $\bar{y}_2$ is too close to zero), then the pooled two-sided 100(1 – α)% Fieller confidence interval for μ_r does not exist. If $a < 0$, then the interval is

$$\left(-\frac{b_p}{a_p} + \frac{\left(b_p^2 - a_p c_p\right)^{\frac{1}{2}}}{a_p} \ , -\frac{b_p}{a_p} - \frac{\left(b_p^2 - a_p c_p\right)^{\frac{1}{2}}}{a_p} \right)$$

For the one-sided intervals, let

$$a_p^\star = \frac{s_p^2 t^2_{1-\alpha,n_1+n_2-2}}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}} - \bar{y}_2^2$$
$$c_p^\star = \frac{s_p^2 t^2_{1-\alpha,n_1+n_2-2}}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} - \bar{y}_1^2$$

which differ from a_p and c_p only in the use of α in place of $\alpha/2$. If $a_p^\star \geq 0$, then the pooled one-sided 100(1 – α)% Fieller confidence intervals for μ_r do not exist. If $a_p^\star < 0$, then the intervals are

$$\left(-\infty \ , -\frac{b_p}{a_p^\star} - \frac{\left(b_p^2 - a_p^\star c_p^\star\right)^{\frac{1}{2}}}{a_p^\star} \right), \quad \text{SIDES=L}$$
$$\left(-\frac{b_p}{a_p^\star} + \frac{\left(b_p^2 - a_p^\star c_p^\star\right)^{\frac{1}{2}}}{a_p^\star} \ , \ \infty \right), \quad \text{SIDES=U}$$

The pooled t test assuming equal variances is the Sasabuchi (1988a, b) test. The hypothesis $H_0\colon \mu_r = \mu_0$ is rewritten as $H_0\colon \mu_1 - \mu_0\mu_2 = 0$, and the pooled t test in the section "Normal Difference (DIST=NORMAL TEST=DIFF)" on page 10833 for the two-independent-sample design is conducted on the original y_{1i} values ($i \in \{1, \ldots, n_1^\star\}$) and transformed values of y_{2i}

$$y_{2i}^\star = \mu_0 y_{2i} \quad , \quad i \in \{1, \ldots, n_2^\star\}$$

with a null difference of 0. The t value for the Sasabuchi pooled test is computed as

$$t_p = \frac{\bar{y}_1 - \mu_0 \bar{y}_2}{s_p \left(\frac{1}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} + \frac{\mu_0^2}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}} \right)^{\frac{1}{2}}}$$

The p-value of the test is computed as

$$p\text{-value} = \begin{cases} P\left(F > t_p^2\right), & \text{two-sided} \\ P\left(T < t_p\right), & \text{lower one-sided} \\ P\left(T > t_p\right), & \text{upper one-sided} \end{cases}$$

where F has an F distribution with one numerator degree of freedom and $n_1 + n_2 - 2$ denominator degrees of freedom and T has a t distribution with $n_1 + n_2 - 2$ degrees of freedom.

Under the assumption of unequal variances, the unpooled Satterthwaite-based confidence interval for the mean ratio μ_r is computed according to the method in Dilba, Schaarschmidt, and Hothorn (2007, the section "Two-sample Problem" on page 20), extended to accommodate weights. The degrees of freedom for the confidence interval are based on the same approximation as in Tamhane and Logan (2004) for the unpooled t test but with the null mean ratio μ_0 replaced by the maximum likelihood estimate $\hat{\mu}_r = \bar{y}_1/\bar{y}_2$:

$$\mathrm{df}_u = \frac{\left(\frac{s_1^2}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} + \frac{\hat{\mu}_r^2 s_2^2}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}}\right)^2}{\frac{s_1^4}{(n_1-1)\left(\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}\right)^2} + \frac{\hat{\mu}_r^4 s_2^4}{(n_2-1)\left(\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}\right)^2}}$$

Let

$$a_u = \frac{s_2^2 t_{1-\frac{\alpha}{2},\mathrm{df}_u}^2}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}} - \bar{y}_2^2$$

$$b_u = \bar{y}_1 \bar{y}_2$$

$$c_u = \frac{s_1^2 t_{1-\frac{\alpha}{2},\mathrm{df}_u}^2}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} - \bar{y}_1^2$$

where s_1 and s_2 are the within-class-level standard deviations defined in the section "Normal Difference (DIST=NORMAL TEST=DIFF)" on page 10833 for the two-independent-sample design. If $a_u \geq 0$ (which occurs when $\bar{y}_2$ is too close to zero), then the unpooled Satterthwaite-based two-sided $100(1-\alpha)\%$ confidence interval for μ_r does not exist. If $a_u < 0$, then the interval is

$$\left(-\frac{b_u}{a_u} + \frac{\left(b_u^2 - a_u c_u\right)^{\frac{1}{2}}}{a_u}, -\frac{b_u}{a_u} - \frac{\left(b_u^2 - a_u c_u\right)^{\frac{1}{2}}}{a_u}\right)$$

The t test assuming unequal variances is the test derived in Tamhane and Logan (2004). The hypothesis $H_0: \mu_r = \mu_0$ is rewritten as $H_0: \mu_1 - \mu_0 \mu_2 = 0$, and the Satterthwaite t test in the section "Normal Difference (DIST=NORMAL TEST=DIFF)" on page 10833 for the two-independent-sample design is conducted on the original y_{1i} values ($i \in \{1, \ldots, n_1^\star\}$) and transformed values of y_{2i}

$$y_{2i}^\star = \mu_0 y_{2i} \quad , \quad i \in \{1, \ldots, n_2^\star\}$$

with a null difference of 0. The degrees of freedom are computed as

$$\mathrm{df}_u = \frac{\left(\frac{s_1^2}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} + \frac{\mu_0^2 s_2^2}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}}\right)^2}{\frac{s_1^4}{(n_1-1)\left(\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}\right)^2} + \frac{\mu_0^4 s_2^4}{(n_2-1)\left(\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}\right)^2}}$$

The t value for the Satterthwaite-based unpooled test is computed as

$$t_u = \frac{\bar{y}_1 - \mu_0 \bar{y}_2}{\left(\frac{s_1^2}{\sum_{i=1}^{n_1^\star} f_{1i} w_{1i}} + \frac{\mu_0^2 s_2^2}{\sum_{i=1}^{n_2^\star} f_{2i} w_{2i}} \right)^{\frac{1}{2}}}$$

The p-value of the test is computed as

$$p\text{-value} = \begin{cases} P\left(F > t_u^2\right), & \text{two-sided} \\ P\left(T < t_u\right), & \text{lower one-sided} \\ P\left(T > t_u\right), & \text{upper one-sided} \end{cases}$$

where F has an F distribution with one numerator degree of freedom and $\mathrm{df}_u^\star$ denominator degrees of freedom and T has a t distribution with $\mathrm{df}_u^\star$ degrees of freedom.

AB/BA Crossover Design

Let "A" and "B" denote the two treatment values. Define the following notation:

$$\begin{aligned} n_1^\star &= \text{number of observations with treatment sequence AB} \\ n_2^\star &= \text{number of observations with treatment sequence BA} \\ y_{11i} &= \text{response value of } i\text{th observation in sequence AB during period 1}, \ i \in \{1, \ldots, n_1^\star\} \\ y_{12i} &= \text{response value of } i\text{th observation in sequence AB during period 2}, \ i \in \{1, \ldots, n_1^\star\} \\ y_{21i} &= \text{response value of } i\text{th observation in sequence BA during period 1}, \ i \in \{1, \ldots, n_2^\star\} \\ y_{22i} &= \text{response value of } i\text{th observation in sequence BA during period 2}, \ i \in \{1, \ldots, n_2^\star\} \end{aligned}$$

So $\{y_{11i}, \ldots, y_{11n_1^\star}\}$ and $\{y_{22i}, \ldots, y_{22n_2^\star}\}$ are all observed at treatment level A, and $\{y_{12i}, \ldots, y_{12n_2^\star}\}$ and $\{y_{21i}, \ldots, y_{21n_1^\star}\}$ are all observed at treatment level B.

Define the *period difference* for an observation as the difference between period 1 and period 2 response values:

$$\mathrm{pd}_{kji} = y_{k1i} - y_{k2i}$$

for $k \in \{1, 2\}$ and $i \in \{1, \ldots, n_k^\star\}$. Similarly, the *period ratio* is the ratio between period 1 and period 2 response values:

$$\mathrm{pr}_{kji} = y_{k1i} / y_{k2i}$$

The *crossover difference* for an observation is the difference between treatment A and treatment B response values:

$$\mathrm{cd}_{kji} = \begin{cases} y_{k1i} - y_{k2i} \ , & k = 1 \\ y_{k2i} - y_{k1i} \ , & k = 2 \end{cases}$$

Similarly, the *crossover ratio* is the ratio between treatment A and treatment B response values:

$$\mathrm{cr}_{kji} = \begin{cases} y_{k1i} / y_{k2i} \ , & k = 1 \\ y_{k2i} / y_{k1i} \ , & k = 2 \end{cases}$$

In the absence of the IGNOREPERIOD option in the PROC TTEST statement, the data are split into two groups according to treatment sequence and analyzed as a two-independent-sample design. If DIST=NORMAL, then the analysis of the treatment effect is based on the half period differences $\{\mathrm{pd}_{kji}/2\}$, and the analysis for the period effect is based on the half crossover differences $\{\mathrm{cd}_{kji}/2\}$. The computations for the normal difference analysis are the same as in the section "Normal Difference (DIST=NORMAL TEST=DIFF)" on page 10833 for the two-independent-sample design. The normal ratio analysis without the IGNOREPERIOD option is not supported for the AB/BA crossover design. If DIST=LOGNORMAL, then the analysis of the treatment effect is based on the square root of the period ratios $\{\sqrt{\mathrm{pr}_{kji}}\}$, and the analysis for the period effect is based on the square root of the crossover ratios $\{\sqrt{\mathrm{cr}_{kji}}\}$. The computations are the same as in the section "Lognormal Ratio (DIST=LOGNORMAL TEST=RATIO)" on page 10835 for the two-independent-sample design.

If the IGNOREPERIOD option is specified, then the treatment effect is analyzed as a paired analysis on the (treatment A, treatment B) response value pairs, regardless of treatment sequence. So the set of pairs is taken to be the concatenation of $\{(y_{111}, y_{121}), \ldots, (y_{11n_1^\star}, y_{12n_1^\star})\}$ and $\{(y_{221}, y_{211}), \ldots, (y_{22n_2^\star}, y_{22n_2^\star})\}$. The computations are the same as in the section "Paired Design" on page 10832.

See Senn (2002, Chapter 3) for a more detailed discussion of the AB/BA crossover design.

TOST Equivalence Test

The hypotheses for an equivalence test are

$$H_0{:}\mu < \theta_L \quad \text{or} \quad \mu > \theta_U$$
$$H_1{:}\theta_L \le \mu \le \theta_U$$

where θ_L and θ_U are the lower and upper bounds specified in the TOST option in the PROC TTEST statement, and μ is the analysis criterion (mean, mean ratio, or mean difference, depending on the analysis). Following the two one-sided tests (TOST) procedure of Schuirmann (1987), the equivalence test is conducted by performing two separate tests:

$$H_{a0}{:}\mu < \theta_L$$
$$H_{a1}{:}\mu \ge \theta_L$$

and

$$H_{b0}{:}\mu > \theta_U$$
$$H_{b1}{:}\mu \le \theta_U$$

The overall p-value is the larger of the two p-values of those tests.

Rejection of H_0 in favor of H_1 at significance level α occurs if and only if the $100(1-2\,\alpha)\%$ confidence interval for μ is contained completely within (θ_L, θ_U). So, the $100(1-2\,\alpha)\%$ confidence interval for μ is displayed in addition to the usual $100(1-\alpha)\%$ interval.

For further discussion of equivalence testing for the designs supported in the TTEST procedure, see Phillips (1990); Diletti, Hauschke, and Steinijans (1991); Hauschke et al. (1999).

Bootstrap Methods

Overview of the Bootstrap

The bootstrap is based on the *plug-in principle* and is an extension of the practice of replacing unknown parameters with estimates (for example, substituting a sample mean for a population mean). The extension goes all the way to the entire population F from which the data being analyzed are a sample.

The most popular variety of bootstrap is the *nonparametric bootstrap*, which relies on random sampling with replacement from the data to estimate the distribution of a sample estimate (or the joint distribution of multiple sample estimates).

The bootstrap methods in PROC TTEST are all based on the nonparametric bootstrap. The other two main varieties are the *parametric bootstrap* (sampling from a model that has estimated parameters) and *smoothed bootstrap* (sampling from a continuous distribution estimate).

The heuristic for the nonparametric bootstrap is as follows:

1. Draw n observations with replacement from the original n data points to create a "bootstrap sample."
2. Calculate a statistic of interest, θ, from the bootstrap sample and denote its computed value as $\hat{\theta}$.
3. Repeat for a total of r samples.

The statistics of primary interest in PROC TTEST are the sample mean and sample standard deviation.

Purpose of the Bootstrap

The main purpose of bootstrapping is to assess the accuracy and precision of one or more sample estimates in terms of bias, standard error, and confidence intervals.

In typical situations, the bootstrap is *not* useful for estimating a population parameter or the CDF or quantiles of sample estimates. This is because the bootstrap distribution is centered around the observed statistic, not the population parameter. For example, the bootstrap cannot improve on a sample mean estimate.

Bootstrapping can also be a useful tool for inference in various situations, such as the following:

- Parametric assumptions are violated. For example, χ^2 intervals for a variance and F-based intervals for the ratio of variances are not robust to deviations from normality, and their coverage does not improve even with increasing sample size.
- It is too difficult to derive formulas.
- The data are stored in a way that makes calculating formulas impractical.

Useful Applications and Notable Shortcomings

Popular and useful applications of bootstrapping include the following:

- Better standard error estimates
- Bias estimates
- Percentile intervals, optionally with corrections for median bias or narrowness bias (or both)

- t-based intervals, which are traditional t-based confidence intervals either with the bootstrap standard error in place of the traditional standard error or with bootstrap quantiles of the t statistic in place of t distribution quantiles

The two most notable shortcomings of the bootstrap are as follows:

- It tends to perform poorly for small samples.
- Bootstrap bias-corrected *estimates* are usually worse than estimates that are based on the original sample. Even though they tend to be more accurate, they also tend to have much higher variance.

Educational Value

Hesterberg (2015) points out several educational benefits of the bootstrap:

- Because the bootstrap works the same way with a wide variety of statistics, students can focus on ideas rather than formulas. They can also focus on statistics that are appropriate rather than "well-behaved."
- Plots of the bootstrap distribution can help make the abstract concrete for concepts such as sampling distributions, standard errors, bias, the central limit theorem, and confidence intervals.
- The action of drawing bootstrap samples reinforces the role that random sampling plays in statistics.
- The relationship between the bootstrap distribution and the original sample is fundamentally the same as the relationship between the original sample and the population. Patterns that are observed in bootstrap samples (for example, excessive narrowness) usually imply similar patterns in random sampling from the population (for example, the same narrowness that is corrected for with the $n/(n-1)$ factor in the traditional sample standard deviation estimate).

Politis (2016) explains how bootstrapping can help ease students into understanding the notion of resampling from an empirical distribution, instilling confidence in mean and variance estimates without relying on the (often unjustifiable) assumption of normality. He suggests a three-stage approach for guiding students through this transition:

1. Introduce Monte Carlo simulation as an alternative to distribution theory.
2. Demonstrate the parametric bootstrap as an alternative to critical value tables.
3. Abandon the parametric paradigm altogether by generating quantiles and percentile intervals from the resampling distribution. Show when the bootstrap works better or worse than the parametric approach.

Weights and Frequencies

The TTEST procedure does not support the use of the WEIGHT statement with the bootstrap because there is no consensus on weighted bootstrap methods.

The FREQ statement is supported with the bootstrap.

Review of Common Notation and Formulas

Most notation and formulas involved in the descriptions of bootstrap methods in subsequent sections have already been discussed in previous sections, but they are presented here for easier reference. Estimates that involve the empirical distribution derived from the data are newly presented in this section and are denoted as "ZE," sometimes with a subscript to distinguish among alternative assumptions.

Table 128.8 summarizes the basic notation for each design that is supported in bootstrap methods in PROC TTEST.

Table 128.8 Common Notation

Symbol	Description
One-Sample Design	
n	Number of observations
μ	Population mean
σ^2	Population variance
α	Value of ALPHA= option in PROC TTEST statement, such that the confidence level for all bootstrap confidence intervals is $100(1-\alpha)\%$
y_i	Value of ith observation, $i \in \{1, \ldots, n\}$
Two-Sample Design	
n_1	Number of observations at the first class level
n_2	Number of observations at the second class level
y_{1i}	Value of ith observation at the first class level, $i \in \{1, \ldots, n_1\}$
y_{2i}	Value of ith observation at the second class level, $i \in \{1, \ldots, n_2\}$
General	
z_p	$100p$ percentile of standard normal distribution
$t_{p,\nu}$	$100p$ percentile of t distribution with ν degrees of freedom

The standard error of $\hat{\theta}$ is the standard deviation of its sampling distribution. The degrees of freedom discussed in this section reflect the values that would be used for t tests for the corresponding designs in PROC TTEST.

One-sample estimates for mean ($\bar{y}$), standard deviation (s), standard error of mean (SE), and degrees of freedom (df) are as follows:

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

$$s = \left(\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2\right)^{\frac{1}{2}}$$

$$\mathrm{SE} = s/\sqrt{n}$$

$$\mathrm{df} = n - 1$$

Two-sample estimates for within-group means ($\bar{y_1}$ and $\bar{y_2}$), mean difference ($\bar{y_d}$), and within-group standard

deviations (s_1 and s_2) are as follows:

$$\bar{y_1} = \frac{1}{n_1}\sum_{i=1}^{n_1} y_{1i}$$
$$\bar{y_2} = \frac{1}{n_2}\sum_{i=1}^{n_2} y_{2i}$$
$$\bar{y_d} = \bar{y_1} - \bar{y_2}$$
$$s_1 = \left(\frac{1}{n_1 - 1}\sum_{i=1}^{n_1}(y_{1i} - \bar{y_1})^2\right)^{\frac{1}{2}}$$
$$s_2 = \left(\frac{1}{n_2 - 1}\sum_{i=1}^{n_2}(y_{2i} - \bar{y_2})^2\right)^{\frac{1}{2}}$$

Two-sample pooled estimates for standard deviation that is assumed to be common within groups (s_p), standard error of mean difference (SE_p), and degrees of freedom (df_p) are as follows:

$$s_p = \left(\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}\right)^{\frac{1}{2}}$$
$$\mathrm{SE}_p = s_p\left(\frac{1}{n_1} + \frac{1}{n_2}\right)^{\frac{1}{2}}$$
$$\mathrm{df}_p = n_1 + n_2 - 2$$

Note that s^2, s_1^2, s_2^2, and s_p^2 are all unbiased estimators of their respective variances.

The two-sample unpooled standard error estimate of the mean difference (SE_u) and degrees of freedom estimate for unpooled (Satterthwaite) t statistic (df_u) are as follows:

$$\mathrm{SE}_u = \left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^{\frac{1}{2}}$$
$$\mathrm{df}_u = \frac{\mathrm{SE}_u^4}{\frac{s_1^4}{(n_1-1)n_1^2} + \frac{s_2^4}{(n_2-1)n_2^2}}$$

The one-sample variance of the empirical distribution ($\hat{\sigma^2}$) and the standard error of the empirical distribution of the mean (ZE) are as follows:

$$\hat{\sigma^2} = \left(\frac{n-1}{n}\right)s^2$$
$$\mathrm{ZE} = \hat{\sigma}/\sqrt{n}$$

The two-sample pooled variance of the empirical distribution ($\hat{\sigma_p^2}$) and the pooled standard error estimate of the empirical distribution of mean difference (ZE_p) are as follows:

$$\hat{\sigma_p^2} = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2}$$

$$\mathrm{ZE}_p = \hat{\sigma}_p \left(\frac{1}{n_1} + \frac{1}{n_2} \right)^{\frac{1}{2}}$$

The two-sample unpooled standard error of the empirical distribution of mean difference is defined as

$$\mathrm{ZE}_u = \left(\frac{(n_1 - 1)s_1^2}{n_1^2} + \frac{(n_2 - 1)s_2^2}{n_2^2} \right)^{\frac{1}{2}}$$

Resampling

For the nonparametric bootstrap for a one-sample design, a *bootstrap sample* is a random draw of n observations with replacement from the original data set, where $\hat{\theta}$ is the statistic that is calculated from a sample of n iid observations (for example, $\bar{y}$ or s), r is the number of independent bootstrap samples, and $\hat{\theta}_i^\star$ is the value of $\hat{\theta}$ for the ith bootstrap sample from the original data, where $i \in \{1, \ldots, r\}$.

The bootstrap for a paired design is identical to the bootstrap for a one-sample design if y_i is defined as the difference between the first and second members of the ith pair.

In a bootstrap for a two-sample design, random draws of size n_1 and n_2 are taken with replacement from the first and second groups, respectively, and combined to produce a single bootstrap sample.

Statistics That Are Resampled

The sample estimates $\hat{\theta}$ for statistics θ that are supported in bootstrap analyses are computed as follows.

For a one-sample design, the mean μ is estimated by

$$\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$$

and the standard deviation σ is estimated by

$$s = \left(\frac{1}{n-1} \sum_{i=1}^{n} (y_i - \bar{y})^2 \right)^{\frac{1}{2}}$$

For a paired design, the mean of the paired difference $d_i = y_{1i} - y_{2i}$ is estimated by

$$\bar{d} = \frac{1}{n} \sum_{i=1}^{n} d_i$$

and the standard deviation of the paired difference is estimated by

$$s_d = \left(\frac{1}{n-1} \sum_{i=1}^{n} (d_i - \bar{d})^2 \right)^{\frac{1}{2}}$$

For a two-sample design, the mean μ_d of the class difference $d_{ij} = y_{1i} - y_{2j}$ is estimated by

$$\bar{y}_d = \bar{y}_1 - \bar{y}_2 = \frac{1}{n_1}\sum_{i=1}^{n_1} y_{1i} - \frac{1}{n_2}\sum_{j=1}^{n_2} y_{2j}$$

Under the assumption of equal variances ($\sigma_1^2 = \sigma_2^2$), the pooled estimate of the standard deviation of the class difference is

$$s_{pd} = \sqrt{2}s_p = \sqrt{2}\left(\frac{(n_1-1)s_1^2 + (n_2-1)s_2^2}{n_1+n_2-2}\right)^{\frac{1}{2}}$$

Under the assumption of unequal variances, the Satterthwaite estimate of the standard deviation of the class difference is

$$s_{ud} = \sqrt{s_1^2 + s_2^2}$$

Bootstrap Standard Error, Bias Estimate, and Quantiles

The *bootstrap standard error* is the sample standard deviation of the bootstrap distribution:

$$s_b = \left(\frac{1}{r-1}\sum_{i=1}^{r}(\hat{\theta}_i^\star - \bar{\hat{\theta}}^\star)^2\right)^{\frac{1}{2}}$$

The *bootstrap bias estimate* is

$$\hat{\text{bias}}_b = \left(\frac{1}{r}\sum_{i=1}^{r}\hat{\theta}_i^\star\right) - \hat{\theta}$$

Several confidence intervals in the next section are based on quantiles of bootstrap samples. Following the convention in Efron and Tibshirani (1993, section 12.5), the quantile for an ambiguous case is chosen as the nearest sample value in the direction toward the center of the bootstrap distribution. This choice ensures that confidence intervals that are constructed from the quantiles satisfy the desired coverage. In particular, the pth quantile (100p percentile) q_p of the bootstrap distribution of $\hat{\theta}_i^\star$ (or some function of $\hat{\theta}_i^\star$) is computed as follows:

- If rp is an integer, then q_p is the rpth largest value.
- Otherwise, if $p \le 0.5$ and $L = \text{floor}((r+1)p) \ge 1$, then q_p is the Lth largest value.
- Otherwise, if $p > 0.5$ and $U = \text{ceil}((r+1)p) \le r$, then q_p is the Uth largest value.
- Otherwise (either $p \le 0.5$ and $L = \text{floor}((r+1)p) < 1$, or $p > 0.5$ and $U = \text{ceil}((r+1)p) > r$), then q_p is undefined and the bootstrap sample must be larger to yield a valid quantile-based confidence interval.

Bootstrap Confidence Intervals

The bootstrap confidence intervals that PROC TTEST implements are based primarily on recommendations from Hesterberg (2015). The recommendations are based on a combination of educational value and good performance in practice.

See Table 128.5 for a summary of which parameters are supported for each type of confidence interval.

For the following sections, let $\hat{S}$ denote the estimate of the standard error based on unbiased variance estimates—that is, SE for a one-sample or paired design, SE_p for a pooled analysis for a two-sample design, or SE_u for an unpooled analysis for a two-sample design. Similarly, let $\hat{Z}$ denote the estimate of the standard error based on the variances of the empirical distribution—that is, ZE, ZE_p, or ZE_u. Finally, let $\hat{\mathrm{df}}$ denote the degrees of freedom that would be used for t tests for the corresponding designs—that is, df, df_p, or df_u.

Normal Interval with Bootstrap Standard Error

Perhaps the most crude confidence interval based on the bootstrap is the *normal interval with bootstrap standard error*, which is simply the normal-based confidence interval with the usual standard error replaced by the bootstrap standard error:

$$\begin{cases} \left(\hat{\theta} - z_{1-\alpha/2} s_b, \hat{\theta} + z_{1-\alpha/2} s_b\right) \quad , & \text{two-sided} \\ \left(-\infty, \hat{\theta} + z_{1-\alpha} s_b\right) \quad , & \text{lower one-sided} \\ \left(\hat{\theta} - z_{1-\alpha} s_b, \infty\right) \quad , & \text{upper one-sided} \end{cases}$$

In PROC TTEST, the normal interval with bootstrap standard error is computed only for the mean or mean difference. Standard confidence intervals for standard deviations are based on the chi-square distribution rather than on the normal distribution and thus do not have a bootstrap analog of this type.

Bootstrap Percentile Interval

The *bootstrap percentile interval* is recommended by Hesterberg (2015) as one of two "quick and dirty" intervals to begin with when introducing students to the bootstrap. Depending on the sidedness, it is the middle, lower, or upper $100(1-\alpha)\%$ of the bootstrap distribution:

$$\begin{cases} \left(q_{\frac{\alpha}{2}}, q_{1-\frac{\alpha}{2}}\right) \quad , & \text{two-sided} \\ (-\infty, q_{1-\alpha}) \quad , & \text{lower one-sided} \\ (q_{\alpha}, \infty) \quad , & \text{upper one-sided} \end{cases}$$

where q are quantiles of $\hat{\theta}^{\star}$.

This interval is usually the most intuitive one for students. It is robust to skewness in the data, but it performs poorly for small sample sizes. It tends to be too narrow, and it is only "first-order accurate." For a one-sample design, first-order accuracy means that the one-sided coverage probability differs from the nominal value by $O(n^{-\frac{1}{2}})$.

t Interval with Bootstrap Standard Error

The other "quick and dirty interval" is the *t interval with bootstrap standard error*, which is the traditional t-based confidence interval with the usual standard error replaced by the bootstrap standard error:

$$\begin{cases} \left(\hat{\theta} - t_{1-\alpha/2,\hat{\mathrm{df}}} s_b, \hat{\theta} + t_{1-\alpha/2,\hat{\mathrm{df}}} s_b\right) \quad , & \text{two-sided} \\ \left(-\infty, \hat{\theta} + t_{1-\alpha,\hat{\mathrm{df}}} s_b\right) \quad , & \text{lower one-sided} \\ \left(\hat{\theta} - t_{1-\alpha,\hat{\mathrm{df}}} s_b, \infty\right) \quad , & \text{upper one-sided} \end{cases}$$

This interval is also the same as the normal interval with bootstrap standard error where normal quantiles are replaced by t quantiles.

In PROC TTEST, the t interval with bootstrap standard error is computed only for the mean or mean difference. Standard confidence intervals for standard deviations are based on the chi-square distribution rather than on the t distribution and thus do not have a bootstrap analog of this type.

The t interval with bootstrap standard error can help students learn formula methods. It performs relatively well for small n but is not robust to skewness in the data.

Students can compare percentile and t intervals: if they are similar, then they are both probably acceptable.

Bootstrap Expanded Percentile Interval

Whereas the usual bootstrap percentile interval has coverage properties similar to the normal interval with bootstrap standard error (robustness to skewness notwithstanding), the *expanded bootstrap percentile interval* alleviates the narrowness bias by "upgrading" the coverage properties to be more like the t interval with bootstrap standard error. The expanded bootstrap percentile is produced by replacing the α in the bootstrap percentile interval with the value α' that solves the equation $h_z(\alpha') = h_t(\alpha)$, where $h_z(p)$ is the half-width of the normal-based $100(1-p)\%$ confidence interval that uses the variance of the empirical distribution and $h_t(p)$ is the half-width of the t-based $100(1-p)\%$ confidence interval that uses the unbiased variance estimate. The *half-width* of a two-sided interval is the length of the interval divided by two, and the half-width of a one-sided interval is the absolute difference between the point estimate and the finite limit.

The general solution of $h_z(\alpha') = h_t(\alpha)$ is

$$\alpha' = d\Phi\left(\frac{\hat{S}}{\hat{Z}} t_{\alpha/d,\hat{\mathrm{df}}}\right)$$

where d is the number of sides.

The solutions for different designs are as follows:

$$\alpha' = \begin{cases} d\Phi\left(\sqrt{\frac{n}{n-1}}\, t_{\alpha/d,n-1}\right) \;, & \text{one-sample or paired analysis} \\ d\Phi\left(\sqrt{\frac{n_1+n_2}{n_1+n_2-2}}\, t_{\alpha/d,n_1+n_2-2}\right) \;, & \text{two-sample pooled analysis} \\ d\Phi\left(\frac{\mathrm{SE}_u}{\mathrm{ZE}_u} t_{\alpha/d,\mathrm{df}_u}\right) \;, & \text{two-sample unpooled analysis} \end{cases}$$

The resulting expanded percentile interval for each case is

$$\begin{cases} \left(q_{\frac{\alpha'}{2}}, q_{1-\frac{\alpha'}{2}}\right) \;, & \text{two-sided} \\ (-\infty, q_{1-\alpha'}) \;, & \text{lower one-sided} \\ (q_{\alpha'}, \infty) \;, & \text{upper one-sided} \end{cases}$$

where q are quantiles of $\hat{\theta}^\star$.

In PROC TTEST, the bootstrap expanded percentile interval is computed only for the mean or mean difference. Standard confidence intervals for standard deviations are based on the chi-square distribution rather than on the normal or t distributions and thus do not have a bootstrap analog of this type.

The expanded interval is better than the bootstrap percentile interval and the t interval with bootstrap standard error but not as good as the bootstrap t interval, which is described in the following section.

Bootstrap *t* Interval

The *bootstrap t interval* eschews the assumption of the t statistic having a t distribution and instead uses quantiles of its bootstrap distribution, along with traditional standard error estimates,

$$\begin{cases} \left(\hat{\theta} - q_{1-\frac{\alpha}{2}}\hat{S}, \hat{\theta} - q_{\frac{\alpha}{2}}\hat{S}\right) \quad , & \text{two-sided} \\ \left(-\infty, \hat{\theta} - q_{\alpha}\hat{S}\right) \quad , & \text{lower one-sided} \\ \left(\hat{\theta} - q_{1-\alpha}\hat{S}, \infty\right) \quad , & \text{upper one-sided} \end{cases}$$

where q are quantiles of $\left(\hat{\theta}^{\star} - \hat{\theta}\right)/\hat{S}^{\star}$ and $\hat{S}$ is the (non-bootstrap) standard error estimate of $\hat{\theta}$ based on unbiased variance estimates.

In PROC TTEST, the bootstrap t interval is computed only for the mean or mean difference. There is no reasonable general formula for the standard error of the sample standard deviation.

The bootstrap t interval allows for asymmetry and is "second-order accurate," satisfying the following properties for a one-sample design:

- $O(n^{-1})$ difference in one-sided coverage probability from nominal value
- robust to bias
- robust to skewness
- transformation-invariant—that is, intervals for some function of θ can be obtained by applying the same transformation to the endpoints

Bootstrap Bias-Corrected Percentile Interval

The *bootstrap bias-corrected percentile interval (BC)* is

$$\begin{cases} (q_{\alpha_1}, q_{\alpha_2}) \quad , & \text{two-sided} \\ (-\infty, q_{\alpha_3}) \quad , & \text{lower one-sided} \\ (q_{\alpha_4}, \infty) \quad , & \text{upper one-sided} \end{cases}$$

where

$$\begin{aligned} \alpha_1 &= \Phi\left(2z_0 + z_{\alpha/2}\right) \\ \alpha_2 &= \Phi\left(2z_0 + z_{1-\alpha/2}\right) \\ \alpha_3 &= \Phi\left(2z_0 + z_{1-\alpha}\right) \\ \alpha_4 &= \Phi\left(2z_0 + z_{\alpha}\right) \\ z_0 &= \Phi^{-1}\left(\#\{\hat{\theta}_i^{\star} < \hat{\theta}\}/r\right) \end{aligned}$$

and q are quantiles of $\hat{\theta}^{\star}$.

The BC interval is the default bootstrap confidence interval in PROC TTEST (and also in the NLIN and CAUSALTRT procedures). It corrects for median bias, which occurs when the median of the sampling distribution differs from θ. The two-sided version is given in Efron and Tibshirani (1993, equation 14.10), and the one-sided version is given in Carpenter and Bithell (2000, equation 9).

Displayed Output

For an AB/BA crossover design, the CrossoverVarInfo table shows the variables that are specified for the response and treatment values in each period of the design.

The summary statistics in the Statistics table and confidence limits in the ConfLimits table are displayed for certain variables and/or transformations or subgroups of the variables in the analysis, depending on the design. For a one-sample design, results are displayed for all variables. For a paired design, results are displayed for the paired difference if you specify the TEST=DIFF option in the PROC TTEST statement, or for the paired ratio if you specify TEST=RATIO.

For a two-independent-sample design, most of the results are displayed for each of the two classes, for the difference (if TEST=DIFF) or ratio of means between classes (if TEST=RATIO), and for the assumed-common within-class standard deviation (if DIST=NORMAL) or coefficient of variation (if TEST=RATIO). Results are *not* displayed for the standard *deviation* of the class difference or ratio or for the coefficient of variation of the class ratio. However, the standard *errors* that are displayed if you specify TEST=DIFF are the standard deviation estimates of the means of each of the two classes and the pooled and unpooled (Satterthwaite) standard deviation estimates of the mean class difference. These standard errors are the same ones that are used in hypothesis tests and confidence limits for means and mean differences.

For an AB/BA crossover design, statistics and confidence limits are displayed for each of the four cells in the design (all four combinations of the two periods and two treatments). If the IGNOREPERIOD option is specified in the VAR statement, then results are also displayed for the overall treatment difference or ratio, the same as they are for the paired design. If the IGNOREPERIOD option is absent, then results are also displayed for the treatment difference or ratio within each sequence and overall, and also for the period difference or ratio, the same as they are for the two-sample design.

The Statistics table displays the following summary statistics:

- the names of the variables, displayed if the NOBYVAR option is used in the PROC TTEST statement
- the name of the classification variable (if the two-independent-sample design is used) or treatment and period (if the AB/BA crossover design is used)
- the Method for estimating standard deviation and standard error for a two-independent-sample design, either pooled (for the equal-variance assumption) or Satterthwaite (for the unequal-variance assumption)
- N, the number of nonmissing values
- the (arithmetic) Mean, displayed if the DIST=NORMAL option is specified in the PROC TTEST statement
- the Geometric Mean, displayed if the DIST=LOGNORMAL option is specified in the PROC TTEST statement
- Std Dev, the standard deviation, displayed if the DIST=NORMAL option is specified in the PROC TTEST statement
- the Coefficient of Variation, displayed if the DIST=LOGNORMAL option is specified in the PROC TTEST statement

- Std Err, the standard error of the mean, displayed if the DIST=NORMAL option is specified in the PROC TTEST statement
- the Minimum value
- the Maximum value

The ConfLimits table displays the following:

- the names of the variables, displayed if the NOBYVAR option is used in the PROC TTEST statement
- the name of the classification variable (if the two-independent-sample design is used) or treatment and period (if the AB/BA crossover design is used)
- the method for estimating standard deviation and standard error for a two-independent-sample design, either pooled (for the equal-variance assumption) or Satterthwaite (for the unequal-variance assumption)
- the (arithmetic) Mean, displayed if the DIST=NORMAL option is specified in the PROC TTEST statement
- the Geometric Mean, displayed if the DIST=LOGNORMAL option is specified in the PROC TTEST statement
- $100(1-\alpha)\%$ CL Mean, the lower and upper confidence limits for the mean. Separate pooled and Satterthwaite confidence limits are shown for the difference or ratio transformations in two-independent-sample designs and AB/BA crossover designs without the IGNOREPERIOD option.
- Std Dev, the standard deviation, displayed if the DIST=NORMAL option is specified in the PROC TTEST statement
- the Coefficient of Variation, displayed if the DIST=LOGNORMAL option is specified in the PROC TTEST statement
- $100(1-\alpha)\%$ CL Std Dev, the equal-tailed confidence limits for the standard deviation, displayed if the DIST=NORMAL and CI=EQUAL options are specified in the PROC TTEST statement
- $100(1-\alpha)\%$ UMPU CL Std Dev, the UMPU confidence limits for the standard deviation, displayed if the DIST=NORMAL and CI=UMPU options are specified in the PROC TTEST statement
- $100(1-\alpha)\%$ CL CV, the equal-tailed confidence limits for the coefficient of variation, displayed if the DIST=LOGNORMAL and CI=EQUAL options are specified in the PROC TTEST statement
- $100(1-\alpha)\%$ UMPU CL CV, the UMPU confidence limits for the coefficient of variation, displayed if the DIST=LOGNORMAL and CI=UMPU options are specified in the PROC TTEST statement

The confidence limits in the EquivLimits table and test results in the TTests and EquivTests tables are displayed only for the test criteria—that is, the variables or transformations being tested. For a one-sample design, results are displayed for all variables in the analysis. For a paired design, results are displayed for the difference if you specify the TEST=DIFF option in the PROC TTEST statement, or for the ratio if you specify TEST=RATIO. For a two-independent-sample design, the results for the difference (if TEST=DIFF) or ratio (if TEST=RATIO) are displayed. For an AB/BA crossover design, results are displayed for the

treatment difference (if TEST=DIFF) or ratio (if TEST=RATIO). If the IGNOREPERIOD option is absent, then results are also displayed for the period difference (if TEST=DIFF) or ratio (if TEST=RATIO).

The EquivLimits table, produced only if the TOST option is specified in the PROC TTEST statement, displays the following:

- the name of the variable(s), displayed if the NOBYVAR option is used in the PROC TTEST statement
- the (arithmetic) Mean, displayed if the DIST=NORMAL option is specified in the PROC TTEST statement
- the Geometric Mean, displayed if the DIST=LOGNORMAL option is specified in the PROC TTEST statement
- Lower Bound, the lower equivalence bound for the mean specified in the TOST option in the PROC TTEST statement
- $100(1-2\alpha)\%$ CL Mean, the lower and upper confidence limits for the mean relevant to the equivalence test. Separate pooled and Satterthwaite confidence limits are shown for two-independent-sample designs and AB/BA crossover designs without the IGNOREPERIOD option.
- Upper Bound, the upper equivalence bound for the mean specified in the TOST option in the PROC TTEST statement
- Assessment, the result of the equivalence test at the significance level specified by the ALPHA= option in the PROC TTEST statement, either “Equivalent” or “Not equivalent”

The TTests table is produced only if the TOST option is *not* specified in the PROC TTEST statement. Separate results for pooled and Satterthwaite tests (and also the Cochran and Cox test, if the COCHRAN option is specified in the PROC TTEST statement) are displayed for two-independent-sample designs and AB/BA crossover designs without the IGNOREPERIOD option. The table includes the following results:

- the name of the variable(s), displayed if the NOBYVAR option is used in the PROC TTEST statement
- t Value, the t statistic for comparing the mean to the null value as specified by the H0= option in the PROC TTEST statement
- DF, the degrees of freedom
- the p-value, the probability of obtaining a t statistic at least as extreme as the observed t value under the null hypothesis

The EquivTests table is produced only if the TOST option is specified in the PROC TTEST statement. Separate results for pooled and Satterthwaite tests are displayed for two-independent-sample designs and AB/BA crossover designs without the IGNOREPERIOD option. Each test consists of two separate one-sided tests. The overall p-value is the larger p-value from these two tests. The table includes the following results:

- the name of the variable(s), displayed if the NOBYVAR option is used in the PROC TTEST statement
- Null, the lower equivalence bound for the Upper test or the upper equivalence bound for the Lower test, as specified by the TOST option in the PROC TTEST statement

- t Value, the t statistic for comparing the mean to the Null value
- DF, the degrees of freedom
- the p-value, the probability of obtaining a t statistic at least as extreme as the observed t value under the null hypothesis

The Equality table gives the results of the test of equality of variances. It is displayed for two-independent-sample designs and AB/BA crossover designs without the IGNOREPERIOD option. The table includes the following results:

- the name of the variable(s), displayed if the NOBYVAR option is used in the PROC TTEST statement
- Num DF and Den DF, the numerator and denominator degrees of freedom
- F Value, the F' (folded) statistic
- Pr > F, the probability of a greater F' value. This is the two-tailed p-value.

The Bootstrap table displays bootstrap standard error, bias estimates, and confidence limits. These results are displayed for certain variables in the analysis, their transformations, or their subgroups (or any combination of these), depending on the design and the analysis options. Results are currently displayed only for one-sample, paired, and two-sample designs and for analyses that assume normal data and involve means and standard deviations of either variables or differences between variables or class levels. Bootstrap standard error and bias estimates are always displayed for these designs and analyses, and Table 128.5 shows which analysis parameters are supported for each type of bootstrap confidence interval.

Bootstrap results are unavailable if you specify any of the TEST=RATIO, DIST=LOGNORMAL, or TOST options in the PROC TTEST statement; if you specify the CROSSOVER= option in the VAR statement; or if you specify the WEIGHT statement. They are also unavailable if your input data set contains summary statistics rather than raw observed values.

For a one-sample design, results are displayed for all variables. For a paired design, results are displayed for the paired difference.

For a two-sample design, results are displayed for the class difference—that is, the difference between an observation from the first class and an observation from the second class. Note in particular that whereas the limits displayed for standard deviation in the ConfLimits table are for the assumed-common within-class standard deviation, the limits displayed for standard deviation in the Bootstrap table are for the standard deviation of the *difference*. The estimated standard deviation of the difference is equal to $\sqrt{2}s_p$ under the equal-variance assumption and is equal to $\sqrt{s_1^2 + s_2^2}$ under the unequal-variance assumption.

The Bootstrap table includes the following results:

- the names of the variables, displayed if the NOBYVAR option is used in the PROC TTEST statement
- the name of the classification variable (if the two-independent-sample design is used)
- the method for estimating standard deviation for a two-independent-sample design, either pooled (for the equal-variance assumption) or Satterthwaite (for the unequal-variance assumption); this choice of method determines both the form of the standard deviation difference estimate on which the bootstrap standard error and bias estimates are based and also the formulas for some of the confidence limits for both mean and standard deviation parameters

- the parameter, either mean or standard deviation
- the bootstrap estimate of the standard error of the statistic
- the bootstrap estimate of the bias
- $100(1-\alpha)\%$ CL, the lower and upper bootstrap confidence limits for the relevant transformation of each parameter, based on the choice of BOOTCI method

ODS Table Names

PROC TTEST assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 128.9. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 128.9 ODS Tables Produced by PROC TTEST

ODS Table Name	Description	Syntax
ConfLimits	$100(1-\alpha)\%$ confidence limits for means, standard deviations, and/or coefficients of variation	By default
Equality	Tests for equality of variance	CLASS statement or VAR / CROSSOVER=
EquivLimits	$100(1-2\alpha)\%$ confidence limits for means	PROC TTEST TOST
EquivTests	Equivalence *t* tests	PROC TTEST TOST
Statistics	Univariate summary statistics	By default
TTests	*t* tests	By default
Bootstrap	Bootstrap standard error and bias estimates and $100(1-\alpha)\%$ bootstrap confidence limits	BOOTSTRAP statement

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

ODS Graph Names

You can refer to every graph that is produced through ODS Graphics by its name. The names of the graphs that PROC TTEST generates are listed in Table 128.10 and Table 128.11. Table 128.10, which is alphabetized by graph name, shows required statements and options. Table 128.11 is organized by required *bootstrap-plot-request*.

Table 128.10 Graphs Produced by PROC TTEST

ODS Graph Name	Plot Description	Statement / Option
AgreementOfPeriods	Plot of period 2 against period 1 response values for an AB/BA crossover design	VAR / CROSSOVER= PLOTS=AGREEMENT(TYPE=PERIOD)
AgreementOfTreatments	Plot of second treatment against first treatment response values for an AB/BA crossover design	VAR / CROSSOVER= PLOTS=AGREEMENT
AgreementPlot	Plot of second response against first response for a paired design	PAIRED statement PLOTS=AGREEMENT
BoxPlot	Box plots, also with confidence band for one-sample or paired design	PLOTS=BOX PLOTS=SUMMARY(UNPACK)
Histogram	Histograms with overlaid kernel densities, and also normal densities if DIST=NORMAL	PLOTS=HISTOGRAM PLOTS=SUMMARY(UNPACK)
Interval	Confidence intervals for means	PLOTS=INTERVAL
ProfilesOverPrd	Plot of response profiles over periods 1 and 2 for an AB/BA crossover design	VAR / CROSSOVER= PLOTS=PROFILES(TYPE=PERIOD)
ProfilesOverTrt	Plot of response profiles over first and second treatments for an AB/BA crossover design	VAR / CROSSOVER= PLOTS=PROFILES
ProfilesPlot	Plot of response profiles over first and second response values for a paired design	PAIRED statement PLOTS=PROFILES
QQPlot	Normal quantile-quantile plots	PLOTS=QQ
SummaryPanel	Histograms with overlaid kernel densities (and also normal densities if DIST=NORMAL) and box plots (and also with confidence band for one-sample or paired design)	PLOTS=SUMMARY
(various)	Bootstrap plots (see Table 128.11)	BOOTSTRAP statement PLOTS=BOOTSTRAP(*bootstrap-plot-request*)

Table 128.11 Bootstrap Graphs Produced by PROC TTEST

ODS Graph Name	Plot Description	Statistic	Design	*bootstrap-plot-request*
BootstrapMeanBoxPlot	Box plots and confidence bands	Mean	One-sample Paired Certain two-sample cases	BOX SUMMARY(UNPACK)
BootMeanPooledBoxPlot	Box plots and pooled confidence bands	Mean	Certain two-sample cases	BOX SUMMARY(UNPACK)
BootMeanSattBoxPlot	Box plots and Satterthwaite confidence bands	Mean	Certain two-sample cases	BOX SUMMARY(UNPACK)
BootstrapStdDevBoxPlot	Box plots and confidence bands	Standard deviation	One-sample Paired	BOX SUMMARY(UNPACK)
BootPooledStdDevBoxPlot	Box plots and confidence bands	Pooled standard deviation	Two-sample	BOX SUMMARY(UNPACK)
BootSattStdDevBoxPlot	Box plots and confidence bands	Satterthwaite standard deviation	Two-sample	BOX SUMMARY(UNPACK)
BootCorrMeanStdDevPlot	Scatter plots with overlaid elliptical prediction regions	Mean and standard deviation	One-sample Paired	CORRELATION
BootCorrMeanPoolStdDevPlot	Scatter plots with overlaid elliptical prediction regions	Mean and pooled standard deviation	Two-sample	CORRELATION
BootCorrMeanSattStdDevPlot	Scatter plots with overlaid elliptical prediction regions	Mean and Satterthwaite standard deviation	Two-sample	CORRELATION
BootstrapMeanHistogram	Histograms with overlaid normal and kernel densities	Mean	One-sample Paired Two-sample	HISTOGRAM SUMMARY(UNPACK)
BootstrapStdDevHistogram	Histograms with overlaid normal and kernel densities	Standard deviation	One-sample Paired	HISTOGRAM SUMMARY(UNPACK)
BootPooledStdDevHistogram	Histograms with overlaid normal and kernel densities	Pooled standard deviation	Two-sample	HISTOGRAM SUMMARY(UNPACK)
BootSattStdDevHistogram	Histograms with overlaid normal and kernel densities	Satterthwaite standard deviation	Two-sample	HISTOGRAM SUMMARY(UNPACK)
BootstrapMeanIntervalPlot	Bootstrap confidence intervals	Mean	One-sample Paired Two-sample	INTERVAL

Table 128.11 *continued*

ODS Graph Name	Plot Description	Statistic	Design	*bootstrap-plot-request*
BootstrapStdDevIntervalPlot	Bootstrap confidence intervals	Standard deviation Pooled and Satterthwaite standard deviations	One-sample Paired Two-sample	INTERVAL
BootstrapMeanQQPlot	Normal quantile-quantile plots	Mean	One-sample Paired Two-sample	QQ
BootstrapStdDevQQPlot	Normal quantile-quantile plots	Standard deviation	One-sample Paired	QQ
BootPooledStdDevQQPlot	Normal quantile-quantile plots	Pooled standard deviation	Two-sample	QQ
BootSattStdDevQQPlot	Normal quantile-quantile plots	Satterthwaite standard deviation	Two-sample	QQ
BootstrapMeanSummaryPanel	Histograms with overlaid normal and kernel densities, box plots, and confidence bands	Mean	One-sample Paired Certain two-sample cases	SUMMARY
BootMeanPooledSummaryPanel	Histograms with overlaid normal and kernel densities, box plots, and pooled confidence bands	Mean	Certain two-sample cases	SUMMARY
BootMeanSattSummaryPanel	Histograms with overlaid normal and kernel densities, box plots, and Satterthwaite confidence bands	Mean	Certain two-sample cases	SUMMARY
BootstrapStdDevSummaryPanel	Histograms with overlaid normal and kernel densities, box plots, and confidence bands	Standard deviation	One-sample Paired	SUMMARY
BootPoolStdDevSummaryPanel	Histograms with overlaid normal and kernel densities, box plots, and confidence bands	Pooled standard deviation	Two-sample	SUMMARY

Table 128.11 *continued*

ODS Graph Name	Plot Description	Statistic	Design	*bootstrap-plot-request*
BootSattStdDevSummaryPanel	Histograms with overlaid normal and kernel densities, box plots, and confidence bands	Satterthwaite standard deviation	Two-sample	SUMMARY

Interpreting Graphs

Agreement Plots for Paired Designs

For paired designs, the second response of each pair is plotted against the first response, with the mean shown as a large bold symbol. If the WEIGHT statement is used, then the mean is the weighted mean. A diagonal line with a slope of 1 and a y-intercept of 0 is overlaid. The location of the points with respect to the diagonal line reveals the strength and direction of the difference or ratio. The tighter the clustering along the same direction as the line, the stronger the positive correlation of the two measurements for each subject. Clustering along a direction perpendicular to the line indicates negative correlation.

Period Agreement Plots for Crossover Designs

The response in the second period is plotted against the response in the first period, with plot symbols distinguishing the two treatment sequences and the two sequence means shown larger in bold. If the WEIGHT statement is used, then the means are weighted means. A diagonal line with a slope of 1 and a y-intercept of 0 is overlaid.

In the absence of a strong period effect, the points from each sequence will appear as mirror images about the diagonal line, farther apart with stronger treatment effects. Deviations from symmetry about the diagonal line indicate a period effect. The spread of points within each treatment sequence is an indicator of between-subject variability. The tighter the clustering along the same direction as the line (within each treatment sequence), the stronger the positive correlation of the two measurements for each subject. Clustering along a direction perpendicular to the line indicates negative correlation.

The period agreement plot is usually less informative than the treatment agreement plot. The exception is when the period effect is stronger than the treatment effect.

Treatment Agreement Plots for Crossover Designs

The response associated with the second treatment is plotted against the response associated with the first treatment, with plot symbols distinguishing the two treatment sequences and the two sequence means shown larger in bold. If the WEIGHT statement is used, then the means are weighted means. A diagonal line with a slope of 1 and a y-intercept of 0 is overlaid.

The location of the points with respect to the diagonal line reveals the strength and direction of the treatment effect. Substantial location differences between the two sequences indicates a strong period effect. The spread of points within each treatment sequence is an indicator of between-subject variability. The tighter the clustering along the same direction as the line (within each treatment sequence), the stronger the positive correlation of the two measurements for each subject. Clustering along a direction perpendicular to the line indicates negative correlation.

Box Plots

The box is drawn from the 25th percentile (lower quartile) to the 75th percentile (upper quartile). The vertical line inside the box shows the location of the median. If DIST=NORMAL, then a diamond symbol shows the location of the mean. The whiskers extend to the minimum and maximum observations, and circles beyond the whiskers identify outliers.

For one-sample and paired designs, a confidence interval for the mean is shown as a band in the background. If the analysis is an equivalence analysis (with the TOST option in the PROC TTEST statement), then the interval is a $100(1 - 2\alpha)\%$ confidence interval shown along with the equivalence bounds. The inclusion of this interval completely within the bounds is indicative of a significant p-value. If the analysis is not an equivalence analysis, then the confidence level is $100(1 - \alpha)\%$. If the SHOWH0 global plot option is used, then the null value for the test is shown. If the WEIGHT statement is used, then weights are incorporated in the confidence intervals.

Histograms

The WEIGHT statement is ignored in the computation of the normal and kernel densities.

Confidence Intervals

If the analysis is an equivalence analysis (with the TOST option in the PROC TTEST statement), then unless the TYPE=PERGROUP option is used, the interval is a $100(1 - 2\alpha)\%$ mean confidence interval shown along with the equivalence bounds. The inclusion of this interval completely within the bounds is indicative of a significant p-value.

If the analysis is not an equivalence analysis, or if the TYPE=PERGROUP option is used, then the confidence level is $100(1 - \alpha)\%$. If the SHOWH0 *global-plot-option* is used, then the null value for the test is shown.

If the SIDES=L or SIDES=U option is used in the PROC TTEST statement, then the unbounded side of the one-sided interval is represented with an arrowhead. Note that the actual location of the arrowhead is irrelevant.

If the WEIGHT statement is used, then weights are incorporated in the confidence intervals.

Profiles for Paired Designs

For paired designs, a line is drawn for each observation from left to right connecting the first response to the second response. The mean first response and mean second response are connected with a bold line. If the WEIGHT statement is used, then the means are weighted means. The more extreme the slope, the stronger the effect. A wide spread of profiles indicates high between-subject variability. Consistent positive slopes indicate strong positive correlation. Widely varying slopes indicate lack of correlation, while consistent negative slopes indicate strong negative correlation.

Profiles over Period for Crossover Designs

For each observation, the response for the first period is connected to the response for second period, regardless of the treatment applied in each period. The means for each treatment sequence are shown in bold. If the WEIGHT statement is used, then the means are weighted means.

In the absence of a strong period effect, the profiles for each sequence will appear as mirror images about an imaginary horizontal line in the center. Deviations from symmetry about this imaginary horizontal line indicate a period effect. A wide spread of profiles within sequence indicates high between-subject variability.

The TYPE=PERIOD plot is usually less informative than the TYPE=TREATMENT plot. The exception is when the period effect is stronger than the treatment effect.

Profiles over Treatment for Crossover Designs

For each observation, the response for the first treatment is connected to the response for the second treatment, regardless of the periods in which they occur. The means for each treatment sequence are shown in bold. If the WEIGHT statement is used, then the means are weighted means.

In general, the more extreme the slope, the stronger the treatment effect. Slope differences between the two treatment sequences measure the period effect. A wide spread of profiles within sequence indicates high between-subject variability.

Q-Q Plots

Q-Q plots are useful for diagnosing violations of the normality and homoscedasticity assumptions. If the data in a Q-Q plot come from a normal distribution, the points will cluster tightly around the reference line. You can use the UNIVARIATE procedure with the NORMAL option to numerically check the normality assumption.

Bootstrap Correlation Plots

The bootstrap correlation plot is useful for visualizing the correlation between the bootstrap mean and standard deviation across bootstrap samples. The elliptical prediction region is the $100(1-p)\%$ prediction region that assumes a bivariate normal distribution, where p is the value of the ALPHA= option in the PROC TTEST statement. The more elongated the shape of the ellipse, the higher the correlation.

Examples: TTEST Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 128.1: Using Summary Statistics to Compare Group Means

(View the complete code for this example (tteex01.sas) in the example repository.)

This example, taken from Huntsberger and Billingsley (1989), compares two grazing methods using 32 steers. Half of the steers are allowed to graze continuously while the other half are subjected to controlled grazing time. The researchers want to know if these two grazing methods affect weight gain differently. The data are read by the following DATA step:

```
data graze;
   length GrazeType $ 10;
   input GrazeType $ WtGain @@;
   datalines;
controlled  45   controlled  62
controlled  96   controlled 128
controlled 120   controlled  99
controlled  28   controlled  50
controlled 109   controlled 115
controlled  39   controlled  96
controlled  87   controlled 100
```

```
controlled  76   controlled  80
continuous  94   continuous  12
continuous  26   continuous  89
continuous  88   continuous  96
continuous  85   continuous 130
continuous  75   continuous  54
continuous 112   continuous  69
continuous 104   continuous  95
continuous  53   continuous  21
;
```

The variable GrazeType denotes the grazing method: "controlled" is controlled grazing and "continuous" is continuous grazing. The dollar sign ($) following GrazeType makes it a character variable, and the trailing at signs (@@) tell the procedure that there is more than one observation per line.

If you have summary data—that is, just means and standard deviations, as computed by PROC MEANS—then you can still use PROC TTEST to perform a simple *t* test analysis. This example demonstrates this mode of input for PROC TTEST. Note, however, that graphics are unavailable when summary statistics are used as input.

The MEANS procedure is invoked to create a data set of summary statistics with the following statements:

```
proc sort;
   by GrazeType;
run;

proc means data=graze noprint;
   var WtGain;
   by GrazeType;
   output out=newgraze;
run;
```

The NOPRINT option eliminates all printed output from the MEANS procedure. The VAR statement tells PROC MEANS to compute summary statistics for the WtGain variable, and the BY statement requests a separate set of summary statistics for each level of GrazeType. The OUTPUT OUT= statement tells PROC MEANS to put the summary statistics into a data set called newgraze so that it can be used in subsequent procedures. This new data set is displayed in Output 128.1.1 by using PROC PRINT as follows:

```
proc print data=newgraze;
run;
```

The _STAT_ variable contains the names of the statistics, and the GrazeType variable indicates which group the statistic is from.

Output 128.1.1 Output Data Set of Summary Statistics

Obs	GrazeType	_TYPE_	_FREQ_	_STAT_	WtGain
1	continuous	0	16	N	16.000
2	continuous	0	16	MIN	12.000
3	continuous	0	16	MAX	130.000
4	continuous	0	16	MEAN	75.188
5	continuous	0	16	STD	33.812
6	controlled	0	16	N	16.000
7	controlled	0	16	MIN	28.000
8	controlled	0	16	MAX	128.000
9	controlled	0	16	MEAN	83.125
10	controlled	0	16	STD	30.535

The following statements invoke PROC TTEST with the newgraze data set, as denoted by the DATA= option:

```
proc ttest data=newgraze;
   class GrazeType;
   var WtGain;
run;
```

The CLASS statement contains the variable that distinguishes between the groups being compared, in this case GrazeType. The summary statistics and confidence intervals are displayed first, as shown in Output 128.1.2.

Output 128.1.2 Summary Statistics and Confidence Limits

The TTEST Procedure

Variable: WtGain

GrazeType	Method	N	Mean	Std Dev	Std Err	Minimum	Maximum
continuous		16	75.1875	33.8117	8.4529	12.0000	130.0
controlled		16	83.1250	30.5350	7.6337	28.0000	128.0
Diff (1-2)	Pooled		-7.9375	32.2150	11.3897		
Diff (1-2)	Satterthwaite		-7.9375		11.3897		

GrazeType	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev	
continuous		75.1875	57.1705	93.2045	33.8117	24.9768	52.3300
controlled		83.1250	66.8541	99.3959	30.5350	22.5563	47.2587
Diff (1-2)	Pooled	-7.9375	-31.1984	15.3234	32.2150	25.7434	43.0609
Diff (1-2)	Satterthwaite	-7.9375	-31.2085	15.3335			

In Output 128.1.2, The GrazeType column specifies the group for which the statistics are computed. For each class, the sample size, mean, standard deviation and standard error, and maximum and minimum values are displayed. The confidence bounds for the mean are also displayed.

Output 128.1.3 shows the results of tests for equal group means and equal variances.

Output 128.1.3 *t* Tests

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	30	-0.70	0.4912
Satterthwaite	Unequal	29.694	-0.70	0.4913

Equality of Variances				
Method	**Num DF**	**Den DF**	**F Value**	**Pr > F**
Folded F	15	15	1.23	0.6981

A group test statistic for the equality of means is reported for both equal and unequal variances. Both tests indicate a lack of evidence for a significant difference between grazing methods ($t = -0.70$ and $p = 0.4912$ for the pooled test, $t = -0.70$ and $p = 0.4913$ for the Satterthwaite test). The equality of variances test does not indicate a significant difference in the two variances ($F' = 1.23$, $p = 0.6981$). Note that this test assumes that the observations in both data sets are normally distributed; this assumption can be checked in PROC UNIVARIATE by using the NORMAL option with the raw data.

Although the ability to use summary statistics as input is useful if you lack access to the original data, some of the output that would otherwise be produced in an analysis on the original data is unavailable. There are also limitations on the designs and distributional assumptions that can be used with summary statistics as input. For more information, see the section "Input Data Set of Statistics" on page 10827.

Example 128.2: One-Sample Comparison with the FREQ Statement

(View the complete code for this example (tteex02.sas) in the example repository.)

This example examines children's reading skills. The data consist of Degree of Reading Power (DRP) test scores from 44 third-grade children and are taken from Moore (1995, p. 337). Their scores are given in the following DATA step:

```
data read;
   input score count @@;
   datalines;
40 2   47 2   52 2   26 1   19 2
25 2   35 4   39 1   26 1   48 1
14 2   22 1   42 1   34 2   33 2
18 1   15 1   29 1   41 2   44 1
51 1   43 1   27 2   46 2   28 1
49 1   31 1   28 1   54 1   45 1
;
```

The following statements invoke the TTEST procedure to test if the mean test score is equal to 30.

```
ods graphics on;

proc ttest data=read h0=30;
   var score;
   freq count;
run;

ods graphics off;
```

The count variable contains the frequency of occurrence of each test score; this is specified in the FREQ statement. The output, shown in Output 128.2.1, contains the results.

Output 128.2.1 TTEST Results

The TTEST Procedure

Variable: score

Frequency: count

N	Mean	Std Dev	Std Err	Minimum	Maximum
44	34.8636	11.2303	1.6930	14.0000	54.0000

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
34.8636	31.4493	38.2780	11.2303	9.2788	14.2291

DF	t Value	Pr > |t|
43	2.87	0.0063

The SAS log states that 30 observations and two variables have been read. However, the sample size given in the TTEST output is N=44. This is due to specifying the count variable in the FREQ statement. The test is significant ($t = 2.87$, $p = 0.0063$) at the 5% level, so you can conclude that the mean test score is different from 30.

The summary panel in Output 128.2.2 shows a histogram with overlaid normal and kernel densities, a box plot, and the 95% confidence interval for the mean.

Output 128.2.2 Summary Panel

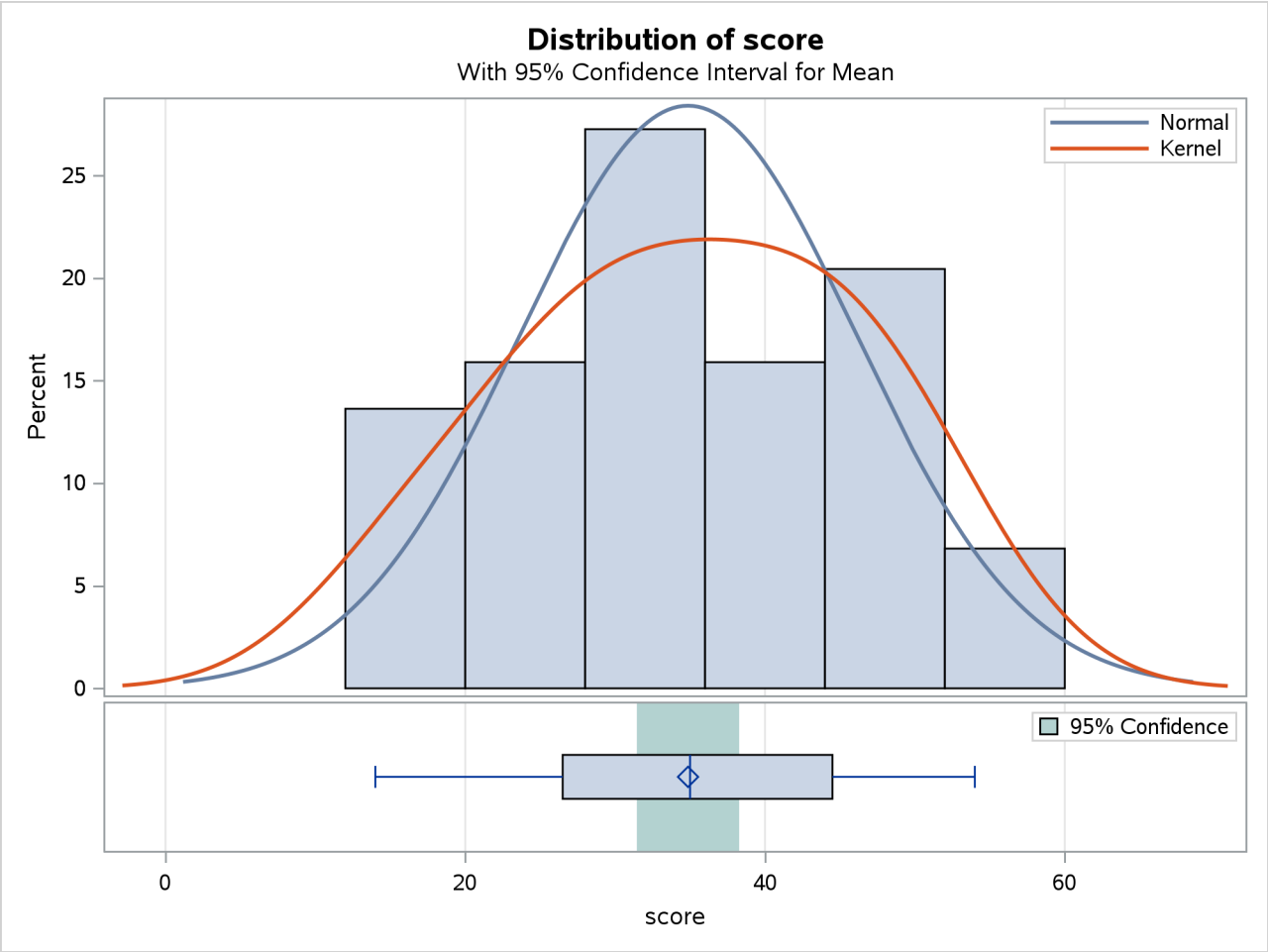

The Q-Q plot in Output 128.2.3 assesses the normality assumption.

Output 128.2.3 Q-Q Plot

The tight clustering of the points around the diagonal line is consistent with the normality assumption. You could use the UNIVARIATE procedure with the NORMAL option to numerically check the normality assumption.

Example 128.3: Paired Comparisons

(View the complete code for this example (tteex03.sas) in the example repository.)

When it is not feasible to assume that two groups of data are independent, and a natural pairing of the data exists, it is advantageous to use an analysis that takes the correlation into account. Using this correlation results in higher power to detect existing differences between the means. The differences between paired observations are assumed to be normally distributed. Some examples of this natural pairing are as follows:

- pre- and post-test scores for a student receiving tutoring
- fuel efficiency readings of two fuel types observed on the same automobile
- sunburn scores for two sunblock lotions, one applied to the individual's right arm, one to the left arm

- political attitude scores of husbands and wives

In this example, taken from the *SUGI Supplemental Library User's Guide, Version 5 Edition*, a stimulus is being examined to determine its effect on systolic blood pressure. Twelve men participate in the study. Each man's systolic blood pressure is measured both before and after the stimulus is applied. The following statements input the data:

```
data pressure;
   input SBPbefore SBPafter @@;
   datalines;
120 128   124 131   130 131   118 127
140 132   128 125   140 141   135 137
126 118   130 132   126 129   127 135
;
```

The variables SBPbefore and SBPafter denote the systolic blood pressure before and after the stimulus, respectively.

The statements to perform the test follow:

```
ods graphics on;

proc ttest;
   paired SBPbefore*SBPafter;
run;

ods graphics off;
```

The PAIRED statement is used to test whether the mean change in systolic blood pressure is significantly different from zero. The tabular output is displayed in Output 128.3.1.

Output 128.3.1 TTEST Results

The TTEST Procedure

Difference: SBPbefore - SBPafter

N	Mean	Std Dev	Std Err	Minimum	Maximum
12	-1.8333	5.8284	1.6825	-9.0000	8.0000

Mean	95% CL Mean		Std Dev	95% CL Std Dev	
-1.8333	-5.5365	1.8698	5.8284	4.1288	9.8958

DF	t Value	Pr > |t|
11	-1.09	0.2992

The variables SBPbefore and SBPafter are the paired variables with a sample size of 12. The summary statistics of the difference are displayed (mean, standard deviation, and standard error) along with their confidence limits. The minimum and maximum differences are also displayed. The t test is not significant ($t = -1.09$, $p = 0.2992$), indicating that the stimuli did not significantly affect systolic blood pressure.

The summary panel in Output 128.3.2 shows a histogram, normal and kernel densities, box plot, and $100(1 - \alpha)\% = 95\%$ confidence interval of the SBPbefore – SBPafter difference.

Output 128.3.2 Summary Panel

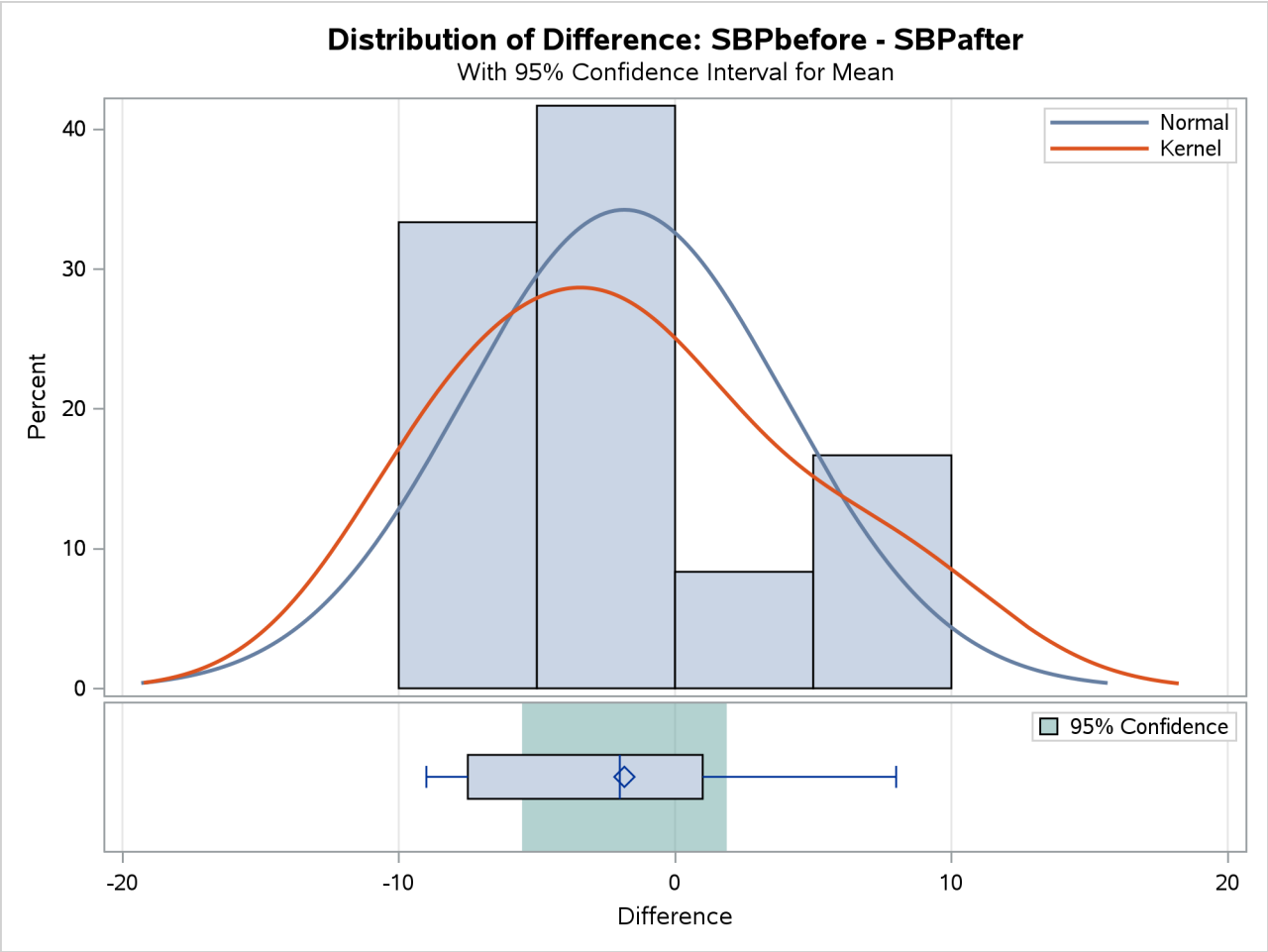

The agreement plot in Output 128.3.3 reveals that only three men have higher blood pressure before the stimulus than after.

Output 128.3.3 Agreement of Treatments

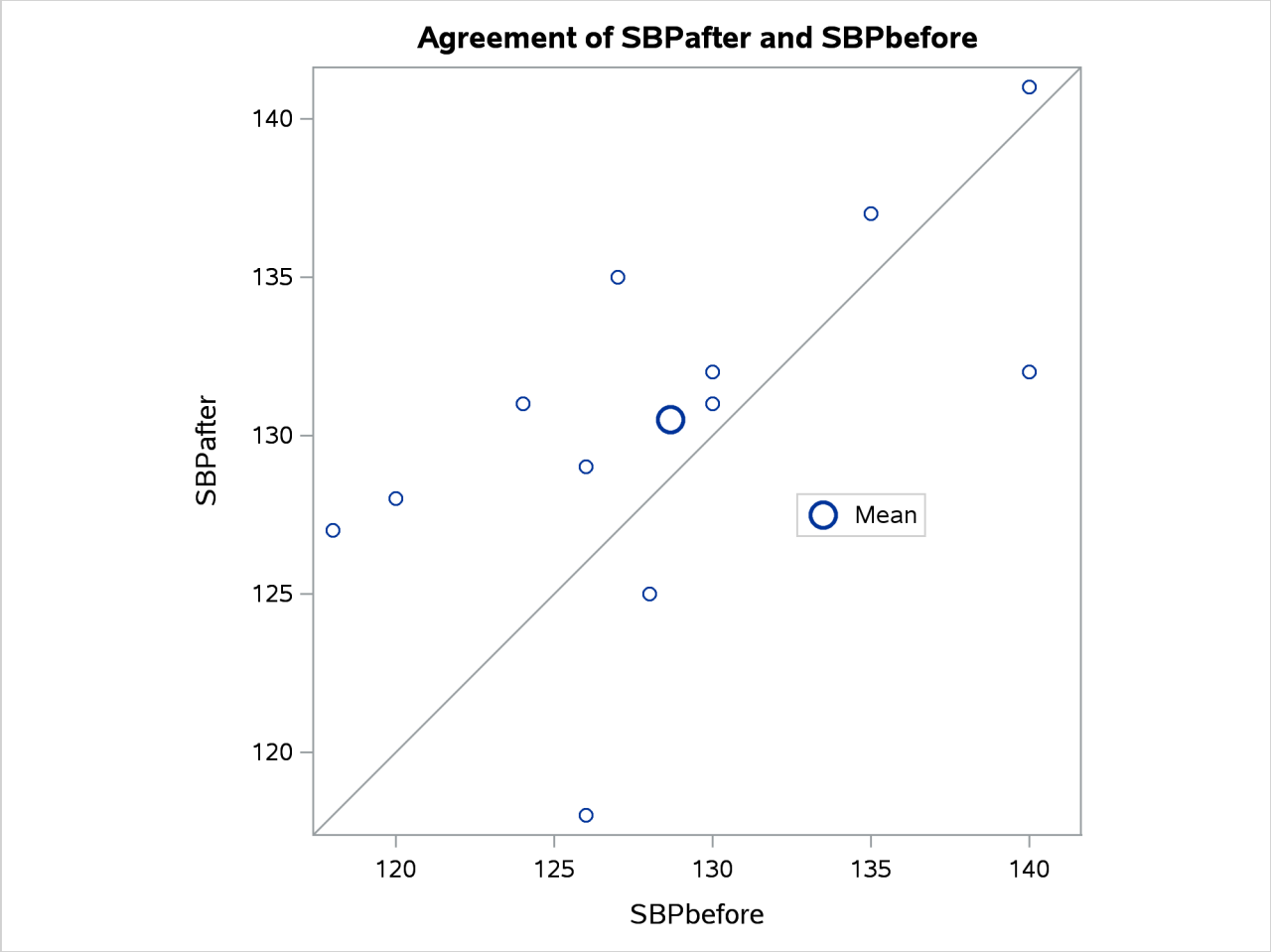

But the differences for these men are relatively large, keeping the mean difference only slightly negative.

The profiles plot in Output 128.3.4 is a different view of the same information contained in Output 128.3.3, plotting the blood pressure from before to after the stimulus.

Output 128.3.4 Profiles over Treatments

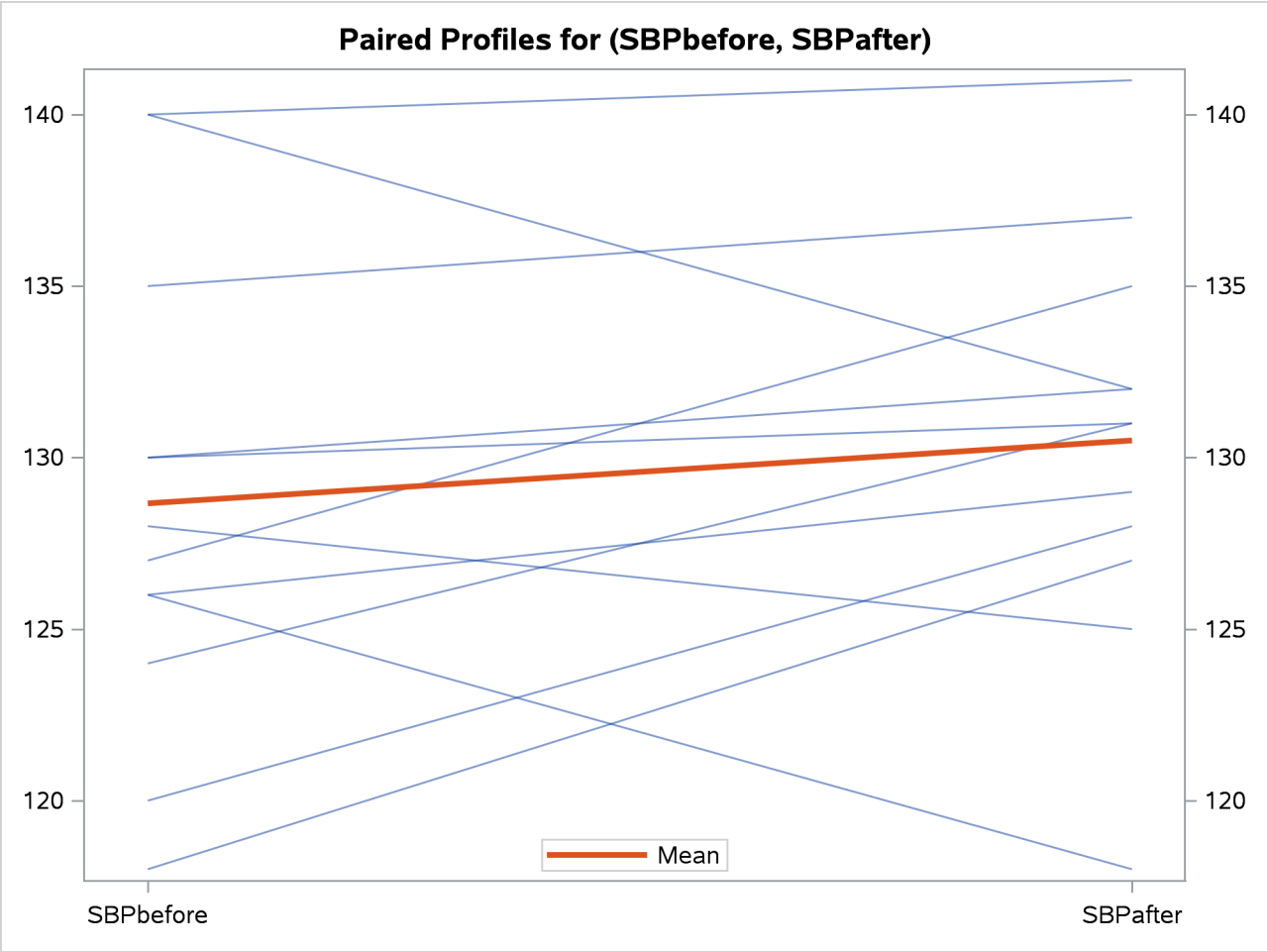

The Q-Q plot in Output 128.3.5 assesses the normality assumption.

Output 128.3.5 Q-Q Plot

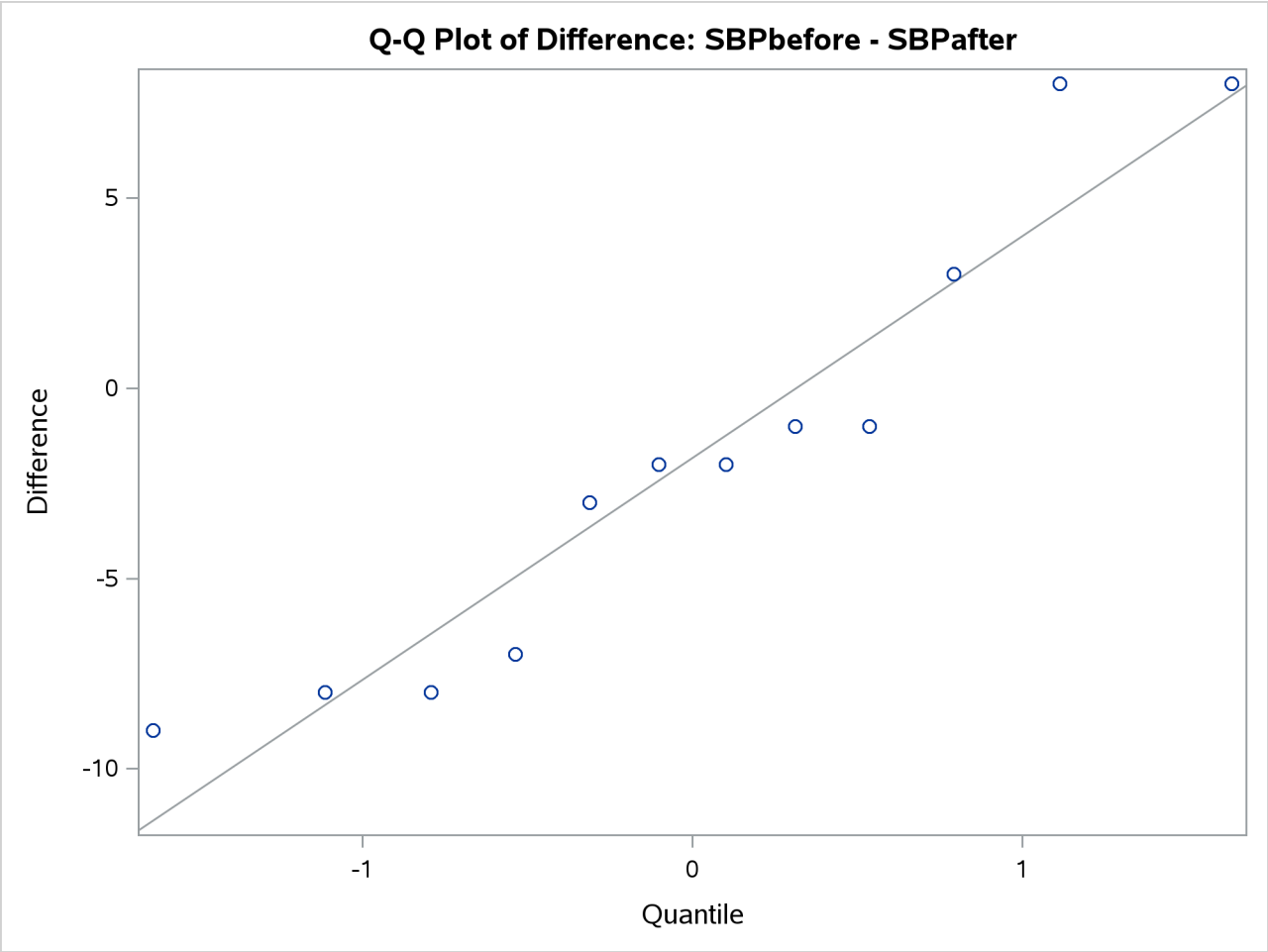

The Q-Q plot shows no obvious deviations from normality. You can check the assumption of normality more rigorously by using PROC UNIVARIATE with the NORMAL option.

Example 128.4: AB/BA Crossover Design

(View the complete code for this example (tteex04.sas) in the example repository.)

Senn (2002, Chapter 3) discusses a study comparing the effectiveness of two bronchodilators, formoterol ("for") and salbutamol ("sal"), in the treatment of childhood asthma. A total of 13 children are recruited for an AB/BA crossover design. A random sample of 7 of the children are assigned to the treatment sequence for/sal, receiving a dose of formoterol upon an initial visit ("period 1") and then a dose of salbutamol upon a later visit ("period 2"). The other 6 children are assigned to the sequence sal/for, receiving the treatments in the reverse order but otherwise in a similar manner. Periods 1 and 2 are sufficiently spaced so that no carryover effects are suspected. After a child inhales a dose of a bronchodilator, peak expiratory flow (PEF) is measured. Higher PEF indicates greater effectiveness. The data are assumed to be approximately normally distributed.

The data set is generated with the following statements:

```
data asthma;
   input Drug1 $ Drug2 $ PEF1 PEF2 @@;
   datalines;
for sal 310 270   for sal 310 260  for sal 370 300
for sal 410 390   for sal 250 210  for sal 380 350
for sal 330 365
sal for 370 385   sal for 310 400  sal for 380 410
sal for 290 320   sal for 260 340  sal for 90  220
;
```

You can display the data by using the following statements, which produce Output 128.4.1:

```
proc print data=asthma;
run;
```

Output 128.4.1 Asthma Study Data

Obs	Drug1	Drug2	PEF1	PEF2
1	for	sal	310	270
2	for	sal	310	260
3	for	sal	370	300
4	for	sal	410	390
5	for	sal	250	210
6	for	sal	380	350
7	for	sal	330	365
8	sal	for	370	385
9	sal	for	310	400
10	sal	for	380	410
11	sal	for	290	320
12	sal	for	260	340
13	sal	for	90	220

The variables PEF1 and PEF2 represent the responses for the first and second periods, respectively. The variables Drug1 and Drug2 represent the treatment in each period.

You can analyze this crossover design by using the CROSSOVER= option after a slash (/) in the VAR statement:

```
ods graphics on;

proc ttest data=asthma plots=interval;
   var PEF1 PEF2 / crossover= (Drug1 Drug2);
run;

ods graphics off;
```

With the default PROC TTEST options TEST=DIFF and DIST=NORMAL and the lack of the IGNOREPERIOD option in the VAR statement, both the treatment difference and the period difference are assessed. The PROC TTEST default options H0=0, SIDES=2, and ALPHA=0.05 specify a two-sided analysis with 95% confidence limits comparing treatment and period differences to a default difference of zero. The default CI=EQUAL option in the PROC TTEST statement requests equal-tailed confidence intervals for standard deviations. The PLOTS=INTERVAL option produces TYPE=TREATMENT confidence inter-

vals, in addition to the default plots AGREEMENT(TYPE=TREATMENT), BOX, HISTOGRAM, PROFILES(TYPE=TREATMENT), and QQ.

Output 128.4.2 summarizes the response and treatment variables for each period.

Output 128.4.2 Crossover Variable Information

The TTEST Procedure

Response Variables: PEF1, PEF2

Crossover Variable Information		
Period	**Response**	**Treatment**
1	PEF1	Drug1
2	PEF2	Drug2

Output 128.4.3 displays basic summary statistics (sample size, mean, standard deviation, standard error, minimum, and maximum) for each of the four cells in the design, the treatment difference within each treatment sequence, the overall treatment difference, and the overall period difference.

Output 128.4.3 Statistics

Sequence	Treatment	Period	Method	N	Mean	Std Dev	Std Err	Minimum	Maximum
1	**for**	**1**		7	337.1	53.7631	20.3206	250.0	410.0
2	**for**	**2**		6	345.8	70.8814	28.9372	220.0	410.0
2	**sal**	**1**		6	283.3	105.4	43.0245	90.0000	380.0
1	**sal**	**2**		7	306.4	64.7247	24.4636	210.0	390.0
1	**Diff (1-2)**			7	30.7143	32.9682	12.4608	-35.0000	70.0000
2	**Diff (1-2)**			6	62.5000	44.6934	18.2460	15.0000	130.0
Both	**Diff (1-2)**		**Pooled**		46.6071	19.3702	10.7766		
Both	**Diff (1-2)**		**Satterthwaite**		46.6071		11.0475		
Both		**Diff (1-2)**	**Pooled**		-15.8929	19.3702	10.7766		
Both		**Diff (1-2)**	**Satterthwaite**		-15.8929		11.0475		

The treatment difference “Diff (1–2)” corresponds to the “for” treatment minus the “sal” treatment, because “for” appears before “sal” in the output, according to the ORDER=MIXED default PROC TTEST option. Its mean estimate is 46.6071, favoring formoterol over salbutamol.

The standard deviation (Std Dev) reported for a “difference” is actually the pooled standard deviation across both treatment sequence (for/sal and sal/for), assuming equal variances. The standard error (Std Err) is the standard deviation of the mean estimate.

The top half of the table in Output 128.4.4 shows 95% two-sided confidence limits for the means for the same criteria addressed in the table in Output 128.4.3.

Output 128.4.4 Confidence Limits

Sequence	Treatment	Period	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev	
1	for	1		337.1	287.4	386.9	53.7631	34.6446	118.4
2	for	2		345.8	271.4	420.2	70.8814	44.2447	173.8
2	sal	1		283.3	172.7	393.9	105.4	65.7841	258.5
1	sal	2		306.4	246.6	366.3	64.7247	41.7082	142.5
1	Diff (1-2)			30.7143	0.2238	61.2048	32.9682	21.2445	72.5982
2	Diff (1-2)			62.5000	15.5972	109.4	44.6934	27.8980	109.6
Both	Diff (1-2)		Pooled	46.6071	22.8881	70.3262	19.3702	13.7217	32.8882
Both	Diff (1-2)		Satterthwaite	46.6071	21.6585	71.5558			
Both		Diff (1-2)	Pooled	-15.8929	-39.6119	7.8262	19.3702	13.7217	32.8882
Both		Diff (1-2)	Satterthwaite	-15.8929	-40.8415	9.0558			

For the mean differences, both pooled (assuming equal variances for both treatment sequences) and Satterthwaite (assuming unequal variances) intervals are shown. For example, the pooled confidence limits for the overall treatment mean difference (for – sal) assuming equal variances are 22.8881 and 70.3262.

The bottom half of Output 128.4.4 shows 95% equal-tailed confidence limits for the standard deviations within each cell and for the treatment difference within each sequence. It also shows confidence limits for the pooled common standard deviation assuming equal variances. Note that the pooled standard deviation of 19.3702 and associated confidence limits 13.7217 and 32.8882 apply to both difference tests (treatment and period), since each of those tests involves the same pooled standard deviation.

Output 128.4.5 shows the results of *t* tests of treatment and period differences.

Output 128.4.5 *t* Tests

Treatment	Period	Method	Variances	DF	t Value	Pr > |t|
Diff (1-2)		Pooled	Equal	11	4.32	0.0012
Diff (1-2)		Satterthwaite	Unequal	9.1017	4.22	0.0022
	Diff (1-2)	Pooled	Equal	11	-1.47	0.1683
	Diff (1-2)	Satterthwaite	Unequal	9.1017	-1.44	0.1838

Both pooled and Satterthwaite versions of the test of treatment difference are highly significant ($p = 0.0012$ and $p = 0.0022$), and both versions of the test of period difference are insignificant ($p = 0.1683$ and $p = 0.1838$).

The folded F test of equal variances in each treatment sequence is shown in Output 128.4.6.

Output 128.4.6 Equality of Variances Test

Equality of Variances				
Method	Num DF	Den DF	F Value	Pr > F
Folded F	5	6	1.84	0.4797

The insignificant result ($p = 0.48$) implies a lack of evidence for unequal variances. However, it does not demonstrate equal variances, and it is not very robust to deviations from normality.

Output 128.4.7 shows the distribution of the response variables PEF1 and PEF2 within each of the four cells

(combinations of two treatments and two periods) of the AB/BA crossover design, in terms of histograms and normal and kernel density estimates.

Output 128.4.7 Comparative Histograms

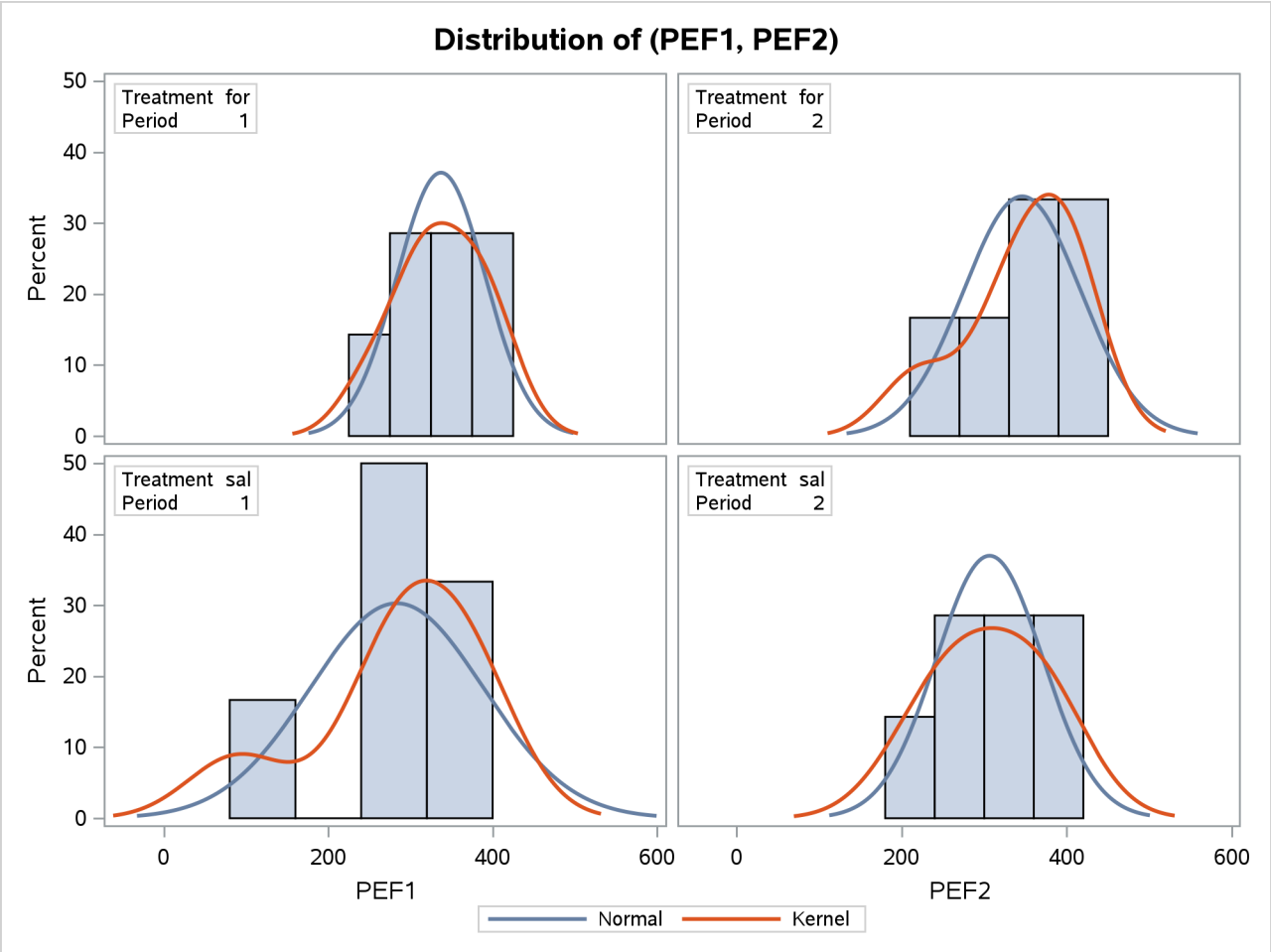

The distributions for the first treatment sequence (for/sal) appear to be somewhat symmetric, and the distributions for the sal/for sequence appear to be skewed to the left.

Output 128.4.8 shows a similar distributional summary but in terms of box plots.

Output 128.4.8 Comparative Box Plots

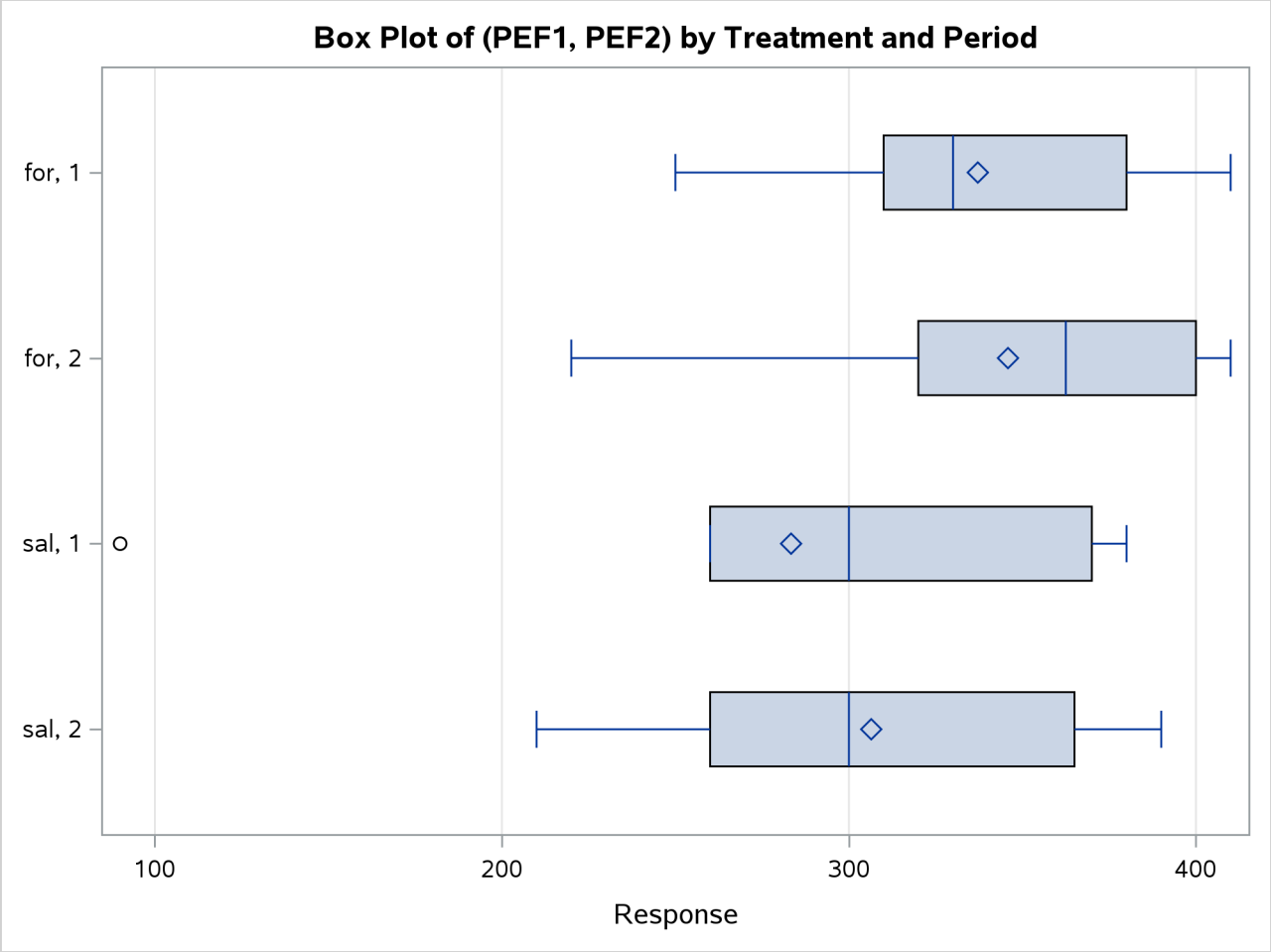

The relative locations of means and medians in each box plot corroborate the fact that the distributions for the sal/for sequence are skewed to the left. The distributions for the for/sal sequence appear to be skewed slightly to the right. The box plot for the salbutamol treatment in the first period shows an outlier (the circle on the far left side of the plot).

The treatment agreement plot in Output 128.4.9 reveals that only a single observation has a higher peak expiratory flow for salbutamol.

Output 128.4.9 Agreement of Treatments Plot

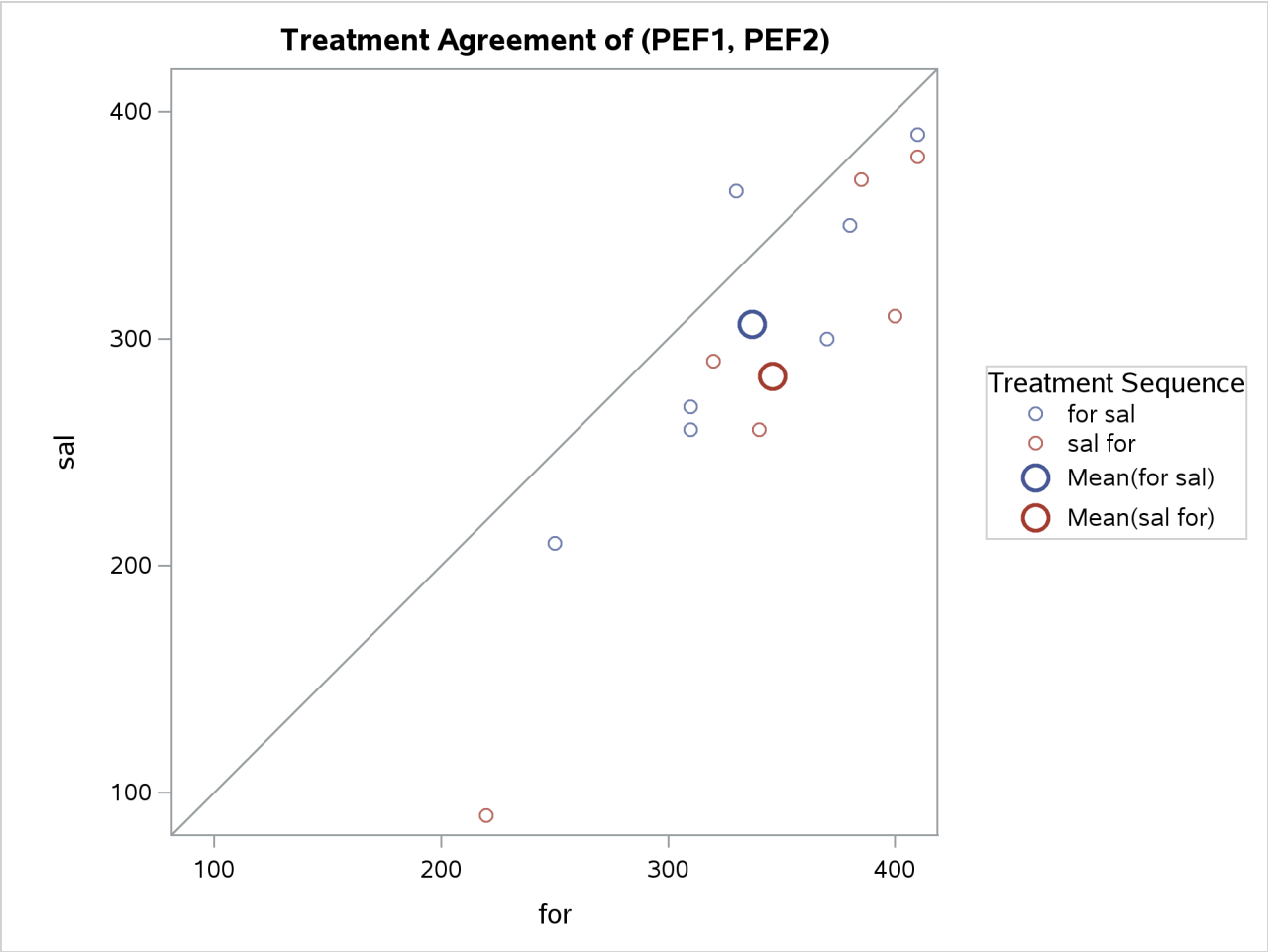

The mean for the sal/for treatment sequence is farther from the diagonal equivalence line, revealing that the treatment difference is more pronounced for the 6 observations in the sal/for sequence than for the 7 observations in the for/sal sequence. This fact is also seen numerically in Output 128.4.3 and Output 128.4.4, which show within-sequence treatment differences of 30.7 for for/sal and 62.5 for sal/for.

The profiles over treatment plot in Output 128.4.10 is a different view of the same information contained in Output 128.4.9, plotting the profiles from formoterol to salbutamol treatments. The lone observation for which the peak expiratory flow is higher for salbutamol appears as the only line with negative slope.

Output 128.4.10 Profiles over Treatment

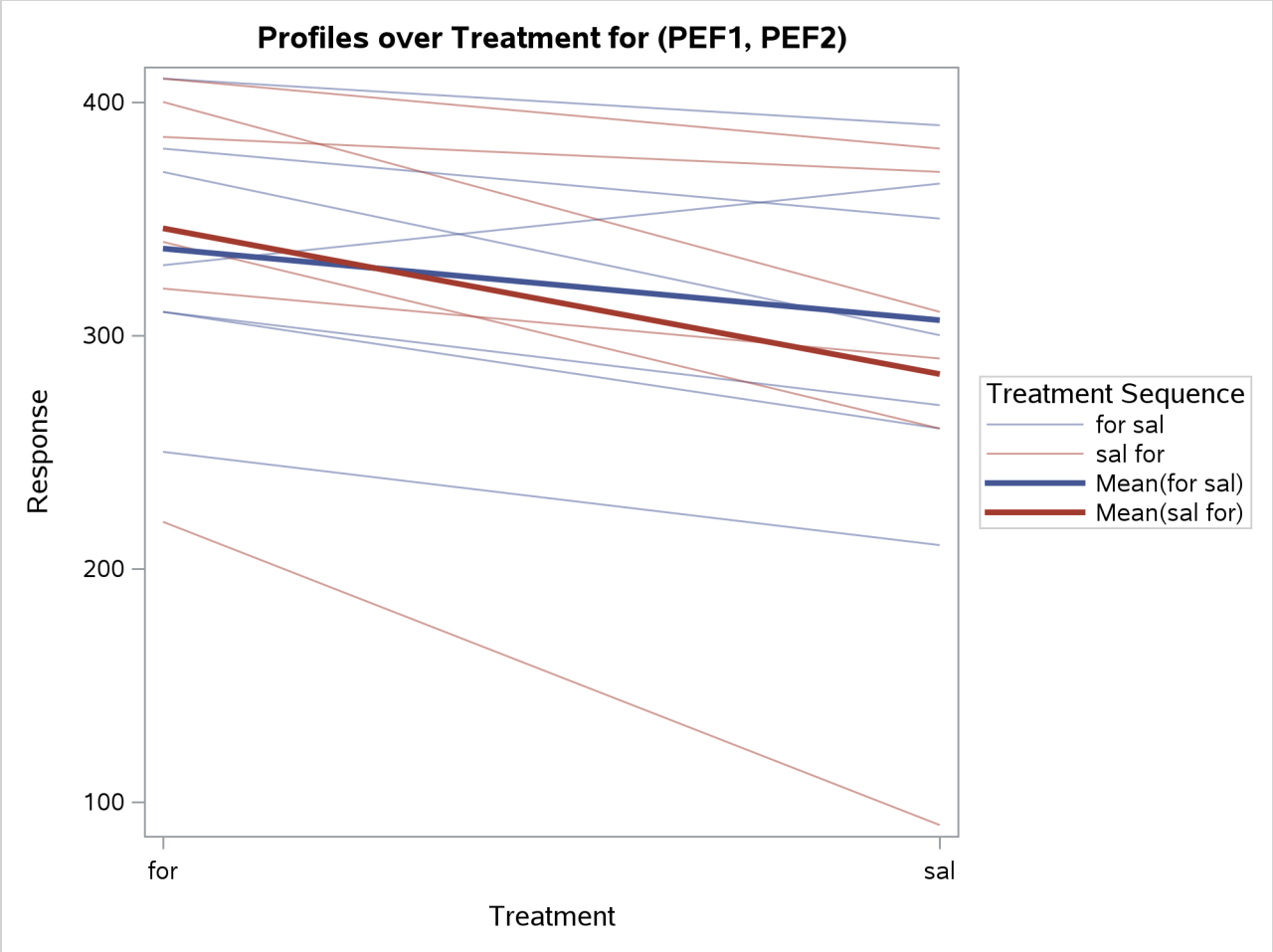

The Q-Q plots in Output 128.4.11 assess normality assumption within each of the four cells of the design.

Output 128.4.11 Q-Q Plots

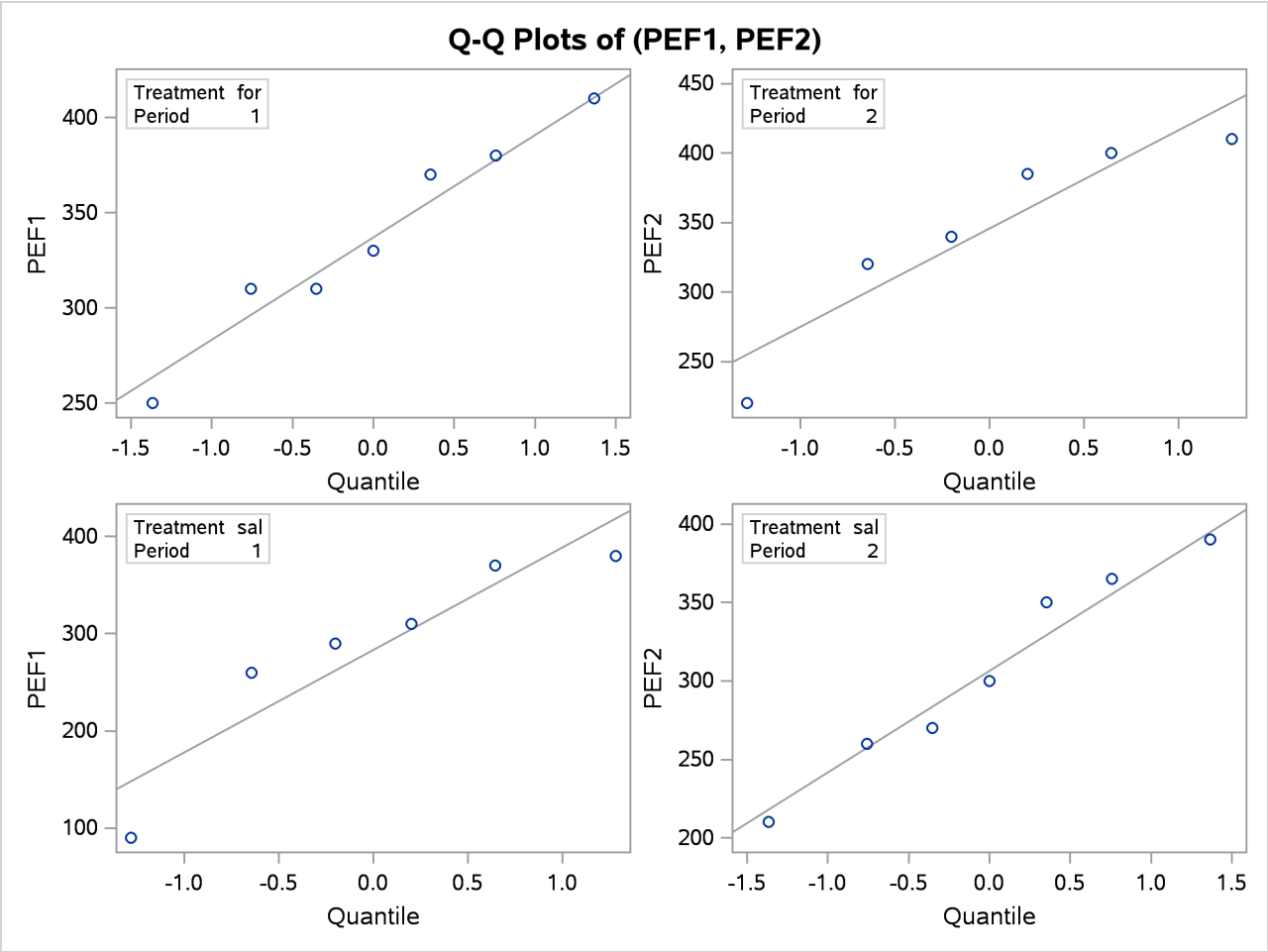

The two Q-Q plots for the sal/for sequence (lower left and upper right) suggest some possible normality violations in the tails, but the sample size is too small to make any strong conclusions. You could use the UNIVARIATE procedure with the NORMAL option to numerically check the normality assumptions.

Finally, Output 128.4.12 shows both pooled and Satterthwaite two-sided 95% confidence intervals for the treatment difference.

Output 128.4.12 Confidence Intervals for Treatment Difference

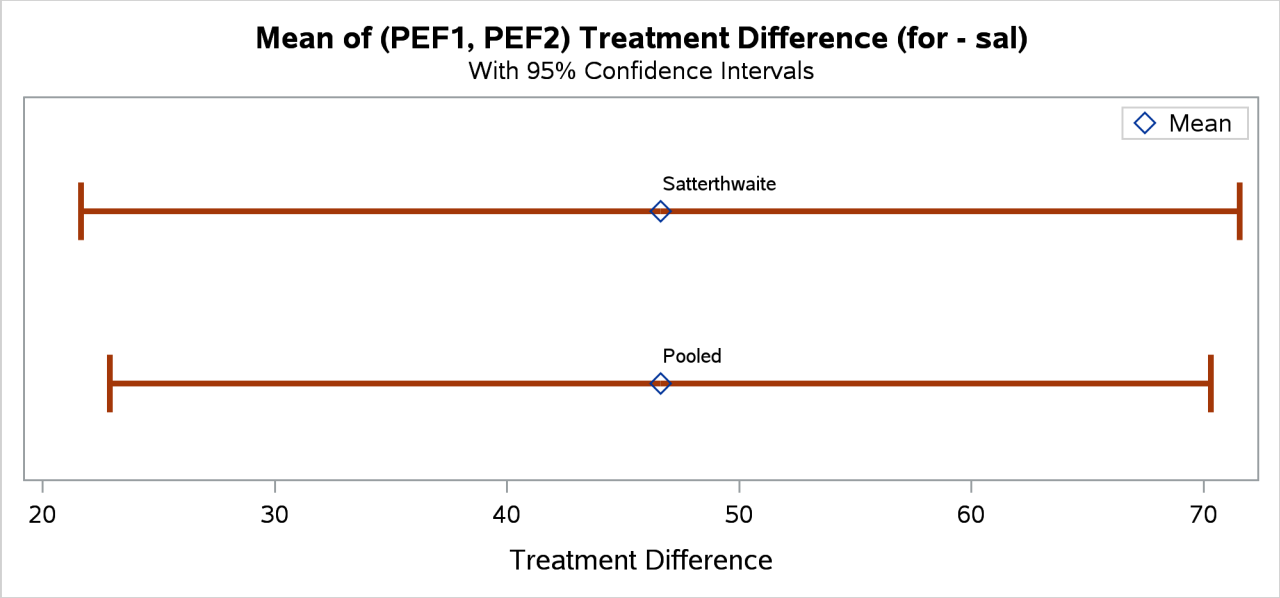

The pooled interval is slightly smaller than the Satterthwaite interval. (This is not always the case.)

Example 128.5: Equivalence Testing with Lognormal Data

(View the complete code for this example (tteex05.sas) in the example repository.)

Wellek (2003, p. 212) discusses an average bioequivalence study comparing the AUC (area under serum-concentration curve) measurements for two different drugs, denoted "Test" and "Reference," over a period of 20 hours. This example looks at a portion of Wellek's data, conducting an equivalence analysis with a paired design that uses AUC values on the original scale (assumed to be lognormally distributed). Each subject in the study received the Test drug upon one visit and then the Reference drug upon a later visit, sufficiently spaced so that no carryover effects would occur.

The goal is to test whether the geometric mean AUC ratio between Test and Reference is between 0.8 and 1.25, corresponding to the traditional FDA (80%, 125%) equivalence criterion. See the section "Arithmetic and Geometric Means" on page 10829 for a discussion of the use of geometric means for lognormal data.

The following SAS statements generate the data set:

```
data auc;
   input TestAUC RefAUC @@;
   datalines;
103.4 90.11  59.92 77.71  68.17 77.71  94.54 97.51
69.48 58.21  72.17 101.3  74.37 79.84  84.44 96.06
96.74 89.30  94.26 97.22  48.52 61.62  95.68 85.80
;
```

You can display the data by using the following statements, which produce Output 128.5.1:

```
proc print data=auc;
run;
```

Output 128.5.1 AUC Data for Test and Reference Drugs

Obs	TestAUC	RefAUC
1	103.40	90.11
2	59.92	77.71
3	68.17	77.71
4	94.54	97.51
5	69.48	58.21
6	72.17	101.30
7	74.37	79.84
8	84.44	96.06
9	96.74	89.30
10	94.26	97.22
11	48.52	61.62
12	95.68	85.80

The TestAUC and RefAUC variables represent the AUC measurements for each subject under the Test and Reference drugs, respectively. Use the following SAS statements to perform the equivalence analysis:

```
ods graphics on;

proc ttest data=auc dist=lognormal tost(0.8, 1.25);
   paired TestAUC*RefAUC;
run;

ods graphics off;
```

The DIST=LOGNORMAL option specifies the lognormal distributional assumption and requests an analysis in terms of geometric mean and coefficient of variation. The TOST option specifies the equivalence bounds 0.8 and 1.25.

Output 128.5.2 shows basic summary statistics for the ratio of TestAUC to RefAUC.

Output 128.5.2 Summary Statistics

The TTEST Procedure

Ratio: TestAUC / RefAUC

N	Geometric Mean	Coefficient of Variation	Minimum	Maximum
12	0.9412	0.1676	0.7124	1.1936

The geometric mean ratio of 0.9412 is the sample mean of the log-transformed data exponentiated to bring it back to the original scale. So the plasma concentration over the 20-hour period is slightly lower for the Test drug than for the Reference drug. The CV of 0.1676 is the ratio of the standard deviation to the (arithmetic) mean.

Output 128.5.3 shows the $100(1-\alpha)\% = 95\%$ confidence limits for the geometric mean ratio (0.8467 and

1.0462) and CV (0.1183 and 0.2884).

Output 128.5.3 Confidence Limits

Geometric Mean	95% CL Mean		Coefficient of Variation	95% CL CV	
0.9412	0.8467	1.0462	0.1676	0.1183	0.2884

Output 128.5.4 shows the $100(1 - 2\,\alpha)\% = 90\%$ confidence limits for the geometric mean ratio, 0.8634 and 1.0260.

Output 128.5.4 Equivalence Limits

Geometric Mean	Lower Bound		90% CL Mean			Upper Bound	Assessment
0.9412	0.8	<	0.8634	1.0260	<	1.25	Equivalent

The assessment of "Equivalent" reflects the fact that these limits are contained within the equivalence bounds 0.8 and 1.25. This result occurs if and only if the p-value of the test is less than the α value specified in the ALPHA= option in the PROC TTEST statement, and it is the reason that $100(1 - 2\,\alpha)\%$ confidence limits are shown in addition to the usual $100(1 - \alpha)\%$ limits.

Output 128.5.5 shows the p-values for the two one-sided tests against the upper and lower equivalence bounds.

Output 128.5.5 TOST Equivalence Test

Test	Null	DF	t Value	P-Value
Upper	0.8	11	3.38	0.0031
Lower	1.25	11	-5.90	<.0001
Overall				0.0031

The overall p-value of 0.0031, the larger of the two one-sided p-values, indicates significant evidence of equivalence between the Test and Reference drugs.

The summary panel in Output 128.5.6 shows a histogram, kernel density, box plot, and $100(1 - 2\,\alpha)\% = 90\%$ confidence interval of the Test-to-Reference ratio of AUC, along with the equivalence bounds.

Output 128.5.6 Summary Panel

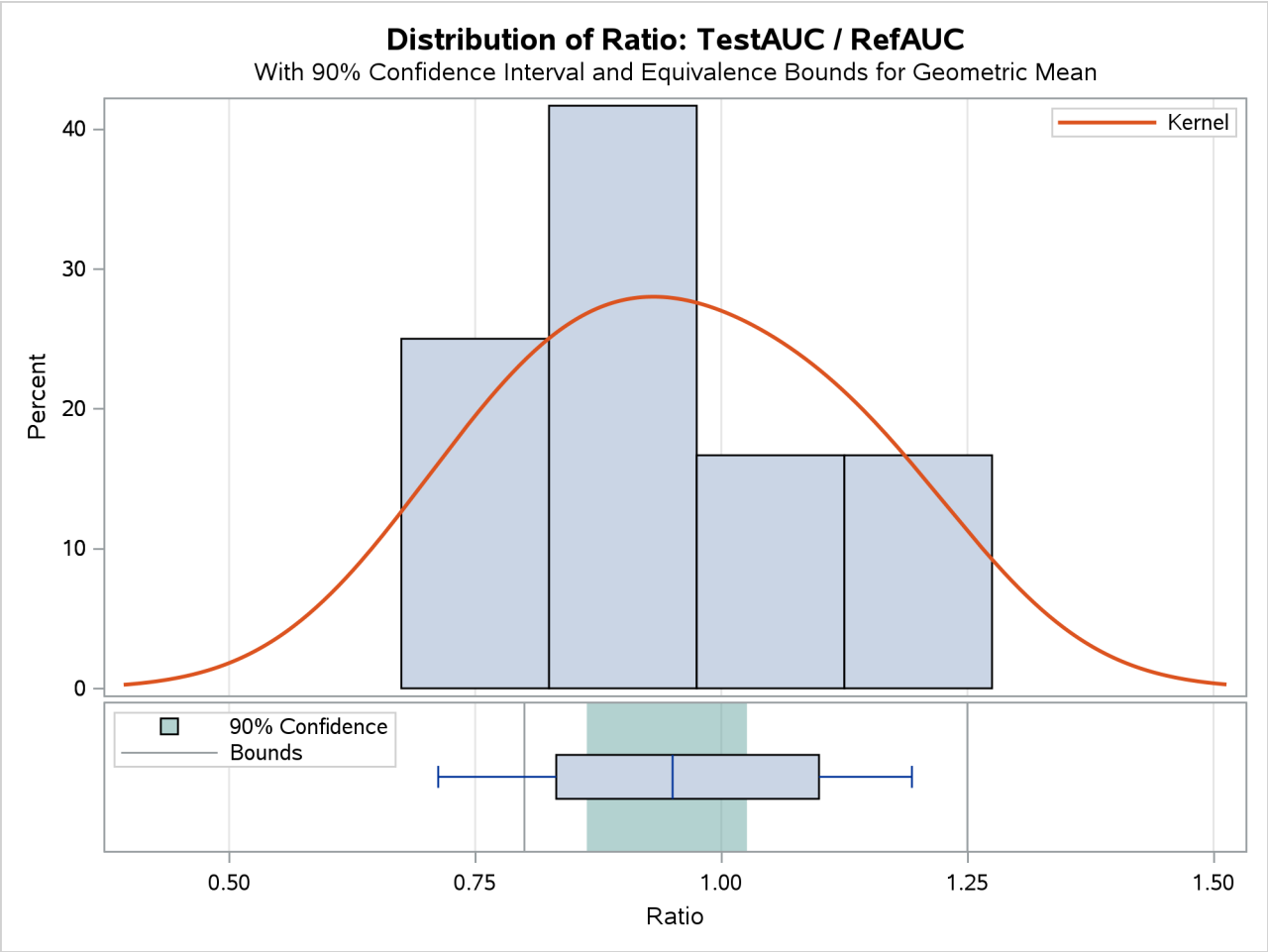

The confidence interval is closer to the lower equivalence bound than the upper bound and contained entirely within the bounds.

The agreement plot in Output 128.5.7 reveals that the only four subjects with higher AUC for the Test drug are at the far lower or far upper end of the AUC distribution. This might merit further investigation.

Output 128.5.7 Agreement Plot

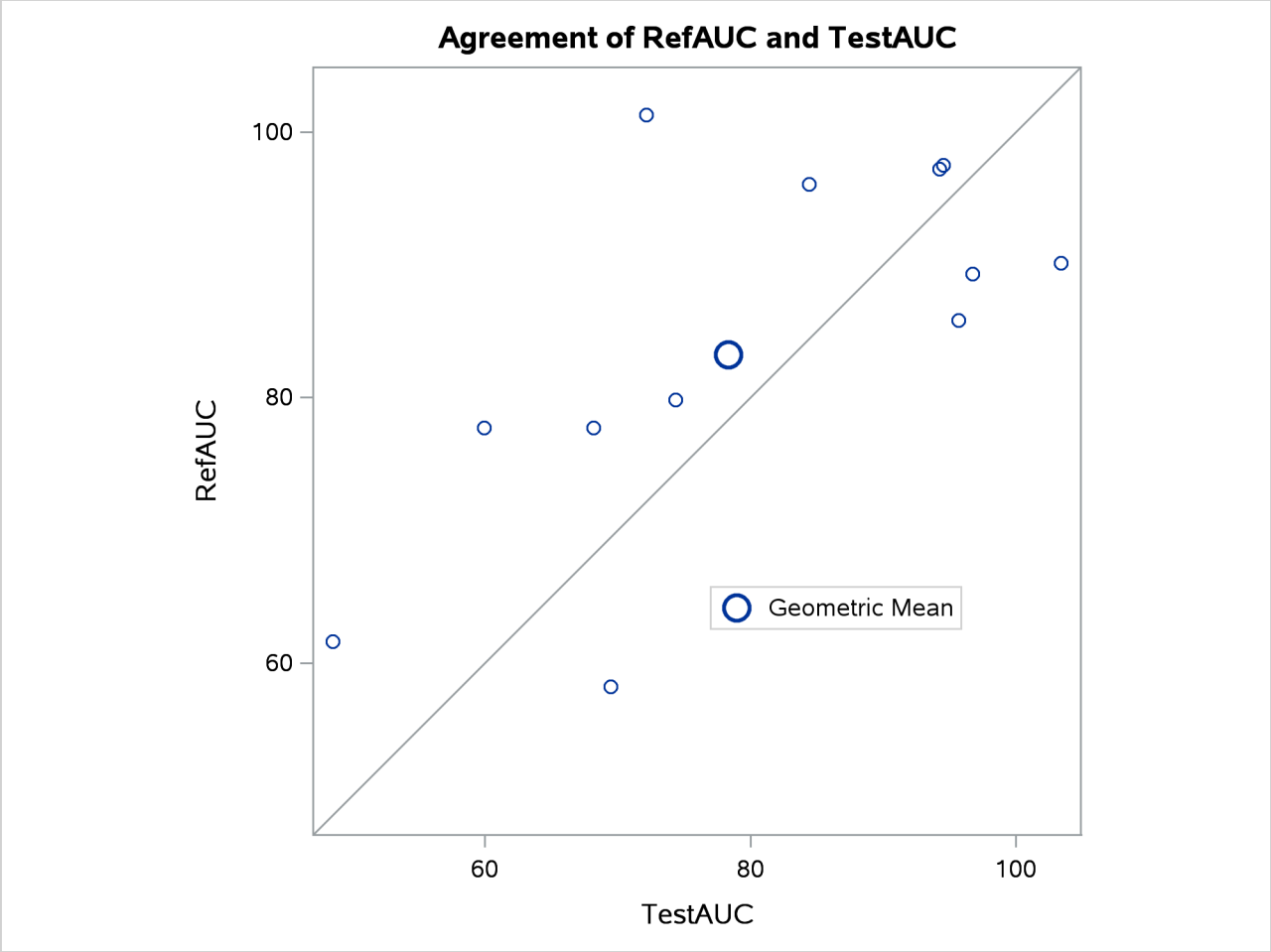

The profiles plot in Output 128.5.8 is a different view of the same information contained in Output 128.5.7, plotting the AUC from Test to Reference drug.

Output 128.5.8 Profiles Plot

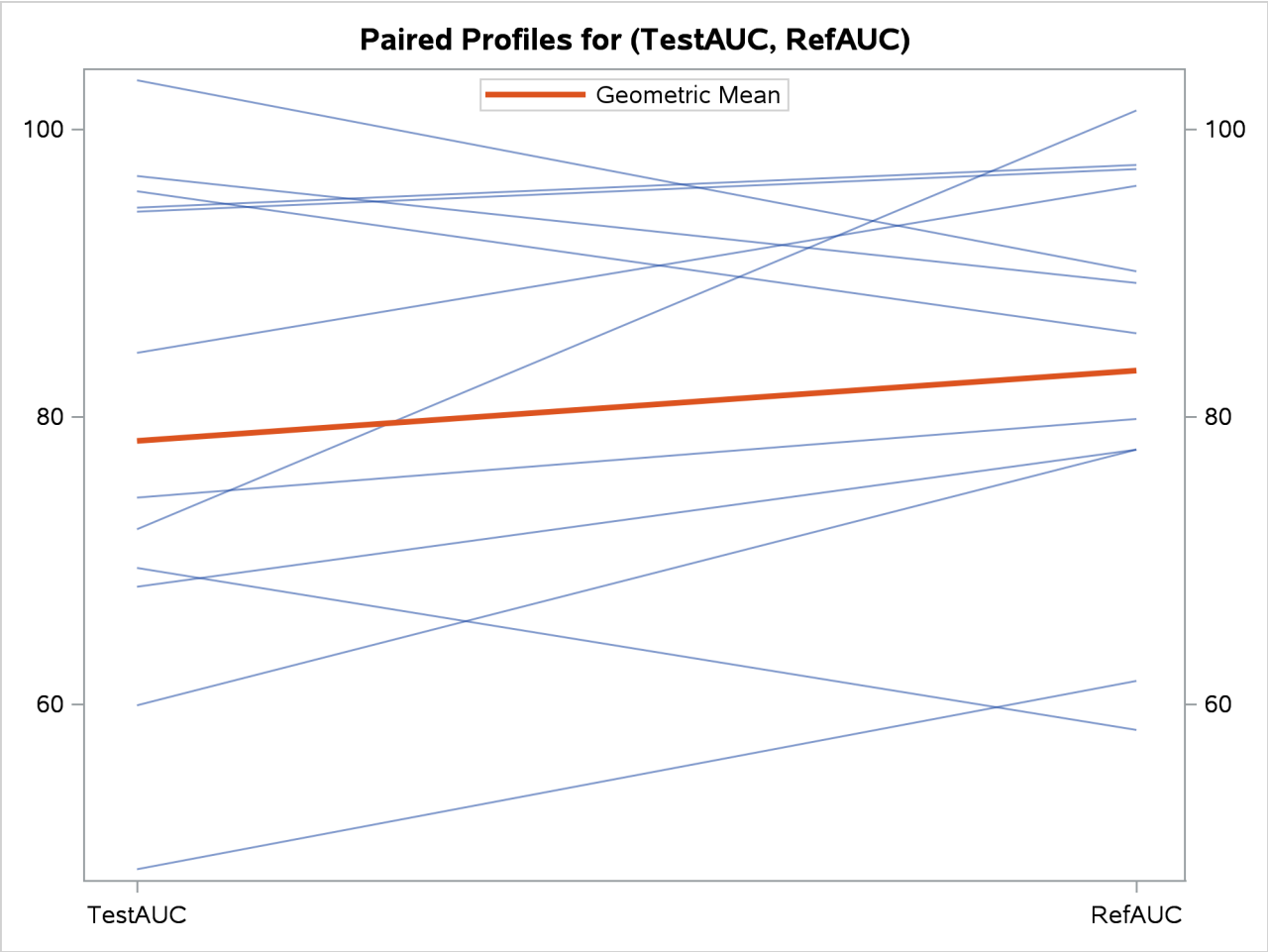

Example 128.6: Bootstrap with Two-Sample Design

(View the complete code for this example (tteex06.sas) in the example repository.)

As discussed in the section "Bootstrap Methods" on page 10840, the bootstrap is a useful technique for producing improved estimates of standard error and bias and improved confidence intervals that adjust for narrowness bias and median bias. The following example illustrates the use of the bootstrap for the same golf score data as are analyzed in the Getting Started example in the section "Comparing Group Means" on page 10808.

The golf scores for males and females in a physical education class are contained in the scores data set, which is created by the following SAS statements:

```
data scores;
   input Gender $ Score @@;
   datalines;
f 75  f 76  f 80  f 77  f 80  f 77  f 73
m 82  m 80  m 85  m 85  m 78  m 87  m 82
;
```

The addition of the BOOTSTRAP statement to the SAS statements specified in the section "Comparing Group Means" on page 10808 requests bootstrap results for the comparison of female and male golf scores, as follows:

```
ods graphics on;

proc ttest ci=equal umpu plots=bootstrap(interval);
   class Gender;
   var Score;
   bootstrap / seed=837;
run;
```

The SEED= option in the BOOTSTRAP statement is specified to enable reproducibility of the results. Because the NSAMPLES= option is not included, PROC TTEST uses 10,000 bootstrap samples by default. Because the BOOTCI= option is not specified, PROC TTEST uses bootstrap bias-corrected percentile intervals by default.

The PLOTS=BOOTSTRAP(INTERVAL) option produces bootstrap interval plots in addition to the default summary panel, Q-Q, and correlation plots.

Output 128.6.1 shows simple statistics for the female and male golf scores being compared, as well as for the difference of the female and male means. The Gender column indicates the population that corresponds to the statistics in that row, and the Method column indicates the method for estimating the standard deviation, either pooled (assuming equal variances for females and males) or Satterthwaite (assuming unequal variances).

Output 128.6.1 Simple Statistics

The TTEST Procedure

Variable: Score

Gender	Method	N	Mean	Std Dev	Std Err	Minimum	Maximum
f		7	76.8571	2.5448	0.9619	73.0000	80.0000
m		7	82.7143	3.1472	1.1895	78.0000	87.0000
Diff (1-2)	Pooled		-5.8571	2.8619	1.5298		
Diff (1-2)	Satterthwaite		-5.8571		1.5298		

Let y_{f} and y_{m} denote the random variables for female and male scores, respectively. The mean of –5.8571 in Output 128.6.1 is computed as $\bar{y}_{\mathrm{f}} - \bar{y}_{\mathrm{m}}$, the estimated mean of the difference $y_{\mathrm{f}} - y_{\mathrm{m}}$. The standard deviation of 2.8619 is the pooled estimate of the standard deviation of either y_{f} or y_{m}, which are assumed in the pooled method to have equal variances. The standard error of 1.5298 is the estimated standard deviation of $\bar{y}_{\mathrm{f}} - \bar{y}_{\mathrm{m}}$. This standard error is identical for both pooled and Satterthwaite methods because the group sample sizes are equal ($n_{\mathrm{f}} = n_{\mathrm{m}}$).

Output 128.6.2 displays both pooled and Satterthwaite 95% intervals for the mean of the difference $y_{\mathrm{f}} - y_{\mathrm{m}}$. It also shows equal-tailed and uniformly most powerful unbiased confidence limits for the assumed-common standard deviation of y_{f} and y_{m} under the assumption of equal variances.

Output 128.6.2 Confidence Limits

Gender	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev		95% UMPU CL Std Dev	
f		76.8571	74.5036	79.2107	2.5448	1.6399	5.6039	1.5634	5.2219
m		82.7143	79.8036	85.6249	3.1472	2.0280	6.9303	1.9335	6.4579
Diff (1-2)	Pooled	-5.8571	-9.1902	-2.5241	2.8619	2.0522	4.7242	2.0019	4.5727
Diff (1-2)	Satterthwaite	-5.8571	-9.2064	-2.5078					

Output 128.6.3 shows the bootstrap results for the golf score data.

Output 128.6.3 Bootstrap Analysis with Bias-Corrected Percentile Intervals

Bootstrap Statistics and Confidence Limits						
Gender	Method	Parameter	Std Error	Bias	95% CL	
Diff (1-2)		Mean	1.4184	0.00440	-8.8571	-3.2857
Diff (1-2)	Pooled	Std Dev	0.5776	-0.3472	3.2071	5.2372
Diff (1-2)	Satterthwaite	Std Dev	0.5776	-0.3472	3.2071	5.2372

Each of the 10,000 bootstrap samples in the bootstrap analysis consists of a collection of $n_{\mathrm{f}} = 7$ random draws with replacement from the female scores and $n_{\mathrm{m}} = 7$ random draws with replacement from the male scores. For each of these samples, three statistics are computed:

- the sample mean estimate of the difference $y_{\mathrm{f}} - y_{\mathrm{m}}$
- the pooled standard deviation estimate of $y_{\mathrm{f}} - y_{\mathrm{m}}$, assuming equal variances
- the Satterthwaite standard deviation estimate of $y_{\mathrm{f}} - y_{\mathrm{m}}$, assuming unequal variances

It is these three statistics that correspond to the three rows in Output 128.6.3, identified by the Method and Parameter columns.

The bootstrap estimate of the standard error of the mean difference, 1.4184, is slightly lower than the estimate of 1.5298 from the original sample. The bootstrap estimate of the bias of $\bar{y}_1 - \bar{y}_2$ is 0.0044, very small compared to either estimate of the standard error.

There is only one row for the mean difference parameter in Output 128.6.3 because the bias-corrected percentile interval is invariant to the choice between pooled and Satterthwaite methods for estimating the standard deviation. The 95% bootstrap bias-corrected confidence interval for the mean difference is substantially narrower than either of the t-based confidence intervals shown in Output 128.6.2. This is not too surprising because the bootstrap bias-corrected interval corrects only for *median* bias—that is, a difference between the parameter θ and the median of its distribution—and not for *narrowness* bias. Narrowness bias in this context is the tendency of uncorrected bootstrap percentile intervals to be too narrow, especially for small sample sizes such as in this case ($n_{\mathrm{f}} = n_{\mathrm{m}} = 7$).

The second and third rows in Output 128.6.3 show bootstrap results for the pooled and Satterthwaite estimates, respectively, of the standard deviation of $y_{\mathrm{f}} - y_{\mathrm{m}}$. These estimates are identical to each other—both in the original sample and in the bootstrap results—because the group sample sizes are equal. The bootstrap standard error estimate, 0.5776, has no straightforward basis for comparison among the estimates that are computed

directly from the original sample in Output 128.6.1 and Output 128.6.2 because there is no reasonable closed-form estimate for the standard deviation of a standard deviation estimate. This demonstrates one of the advantages of the bootstrap—that it can produce standard error estimates for any statistical estimator.

The estimated bias of the standard deviation of the difference is relatively large compared to the standard error and implies that the standard deviation of $y_{\mathrm{f}} - y_{\mathrm{m}}$ is being underestimated in the original sample.

The bootstrap bias-corrected percentile interval for the standard deviation of $y_{\mathrm{f}} - y_{\mathrm{m}}$, (3.2071, 5.2372), cannot be directly compared with the estimate and confidence limits of the pooled standard deviation that are computed from the original sample in Output 128.6.2. However, you can transform the statistics for the pooled standard deviation into statistics for the standard deviation of $y_{\mathrm{f}} - y_{\mathrm{m}}$ simply by multiplying them by $\sqrt{2}$, because the pooled estimate of $\mathrm{Var}(y_{\mathrm{f}} - y_{\mathrm{m}})$ is $2s_p^2$. The resulting point estimate from the original sample is $2.8619 \times \sqrt{2} = 4.0473$. The transformed equal-tail and UMPU confidence intervals are, respectively, (2.9022, 6.6810) and (2.8311, 6.4668).

The bootstrap summary panels in Output 128.6.4 through Output 128.6.6 show histograms, normal and kernel densities, box plots, and $100(1-\alpha)\% = 95\%$ bootstrap bias-corrected percentile intervals for the mean, pooled standard deviation, and Satterthwaite standard deviation of the difference between female and male scores.

Output 128.6.4 Summary Panel of Bootstrap Mean

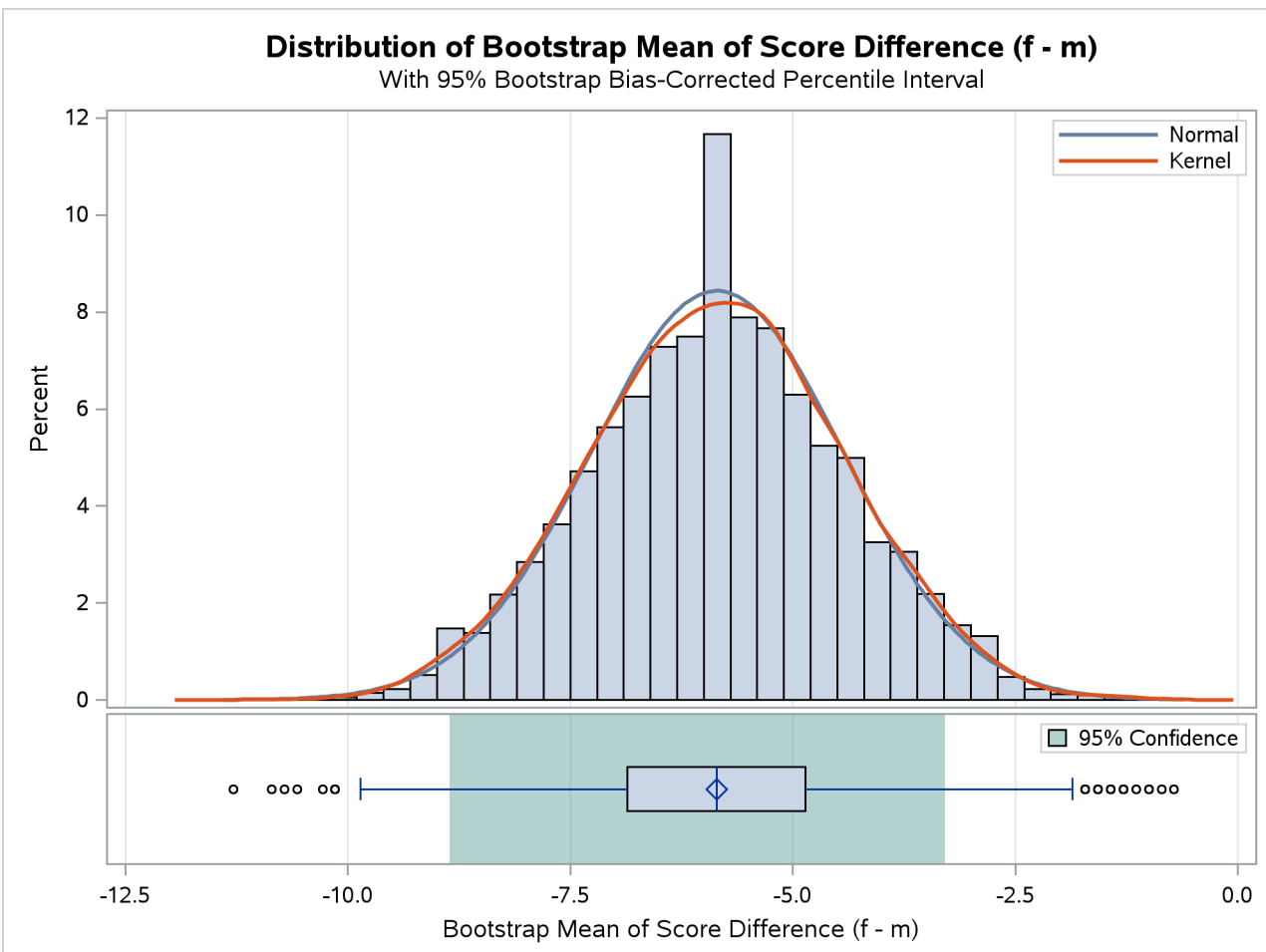

Output 128.6.5 Summary Panel of Bootstrap Pooled Standard Deviation

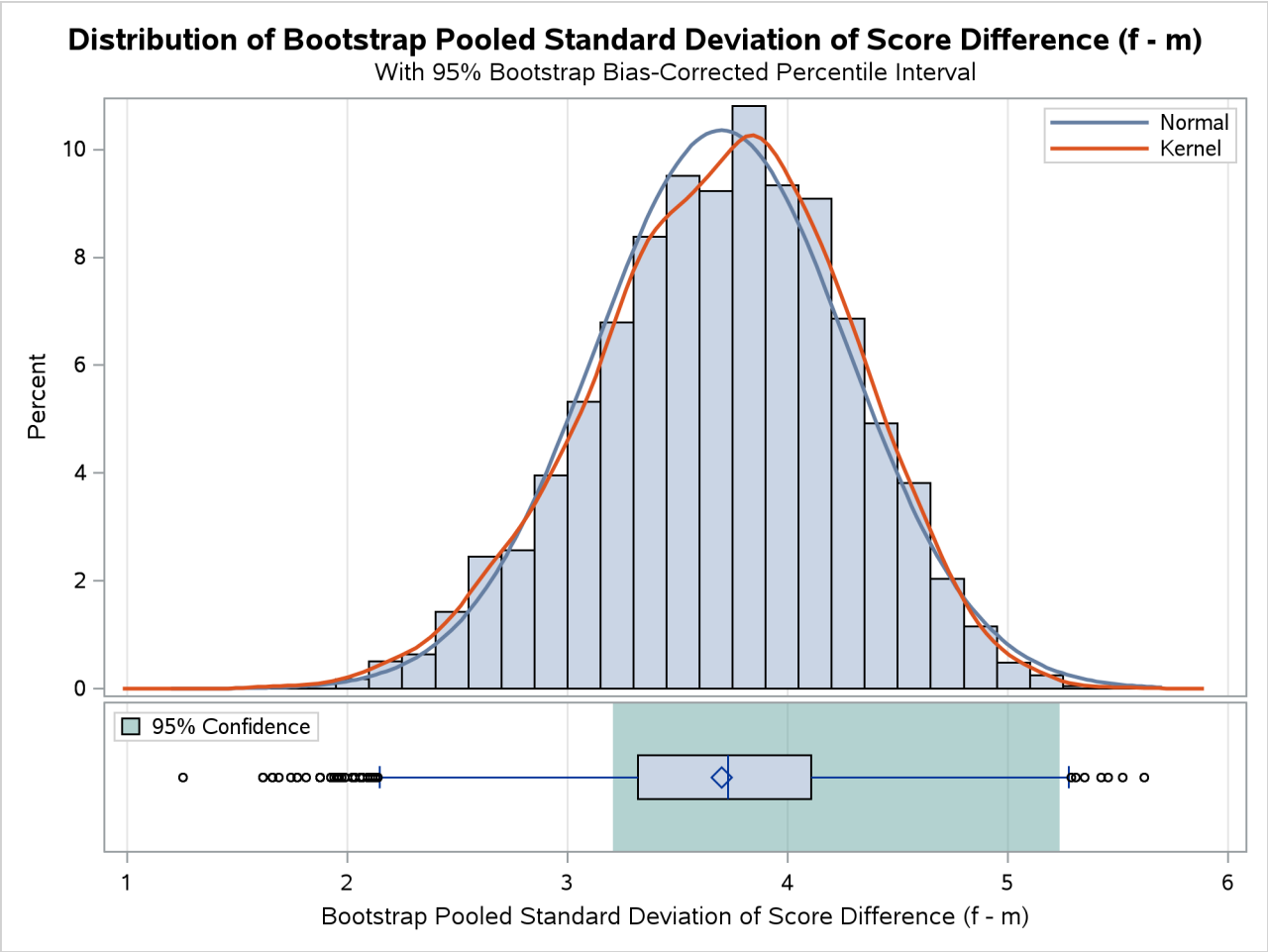

Output 128.6.6 Summary Panel of Bootstrap Satterthwaite Standard Deviation

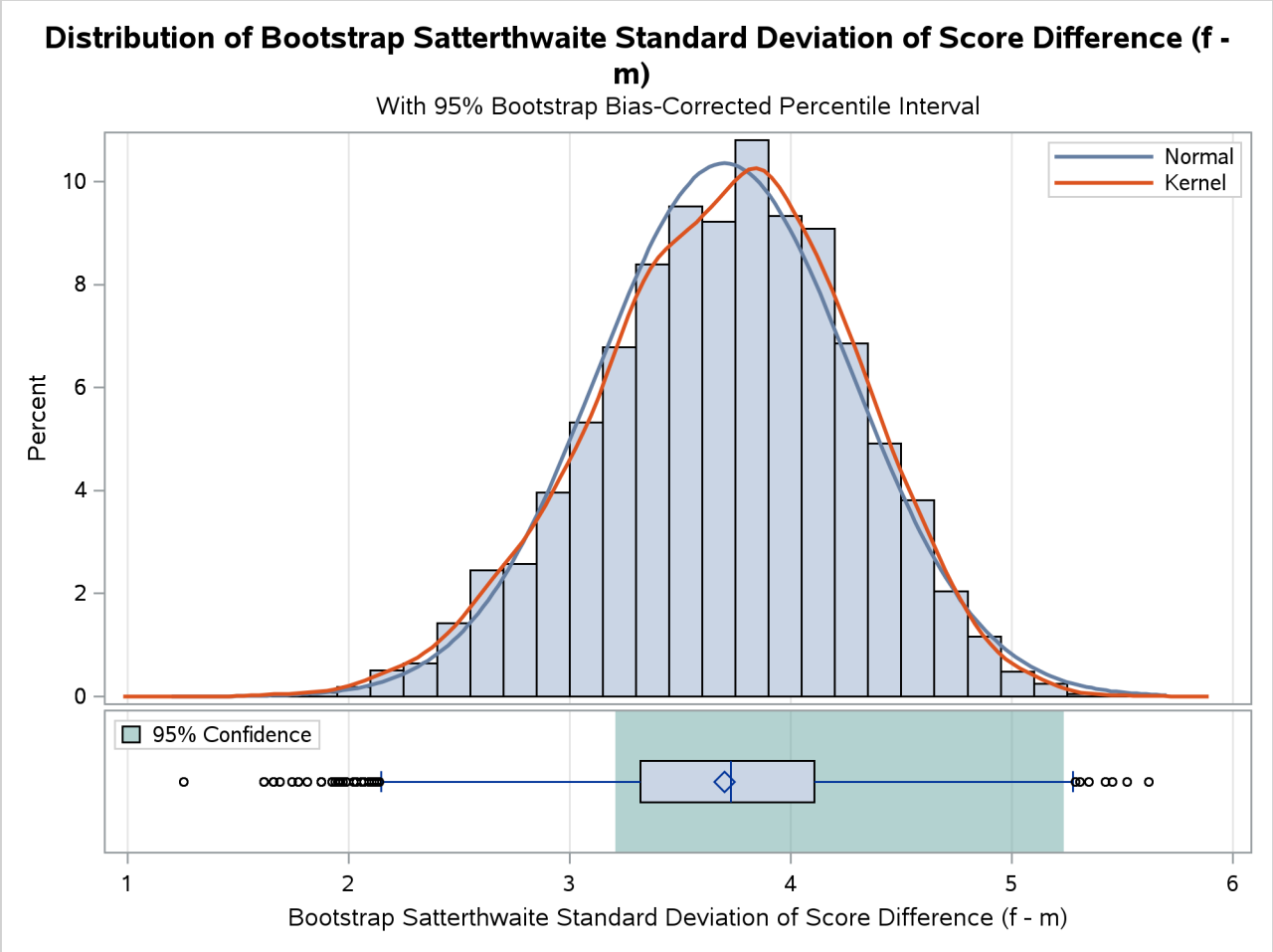

The bootstrap Q-Q plots in Output 128.6.7 through Output 128.6.9 assess the normality of the distributions of the bootstrap mean, pooled standard deviation, and Satterthwaite standard deviation of the difference between female and male scores. The deviations from normality in the upper tail of the standard deviations are not surprising, because they have approximate chi-square distributions for normal data.

Output 128.6.7 Q-Q Plot of Bootstrap Mean

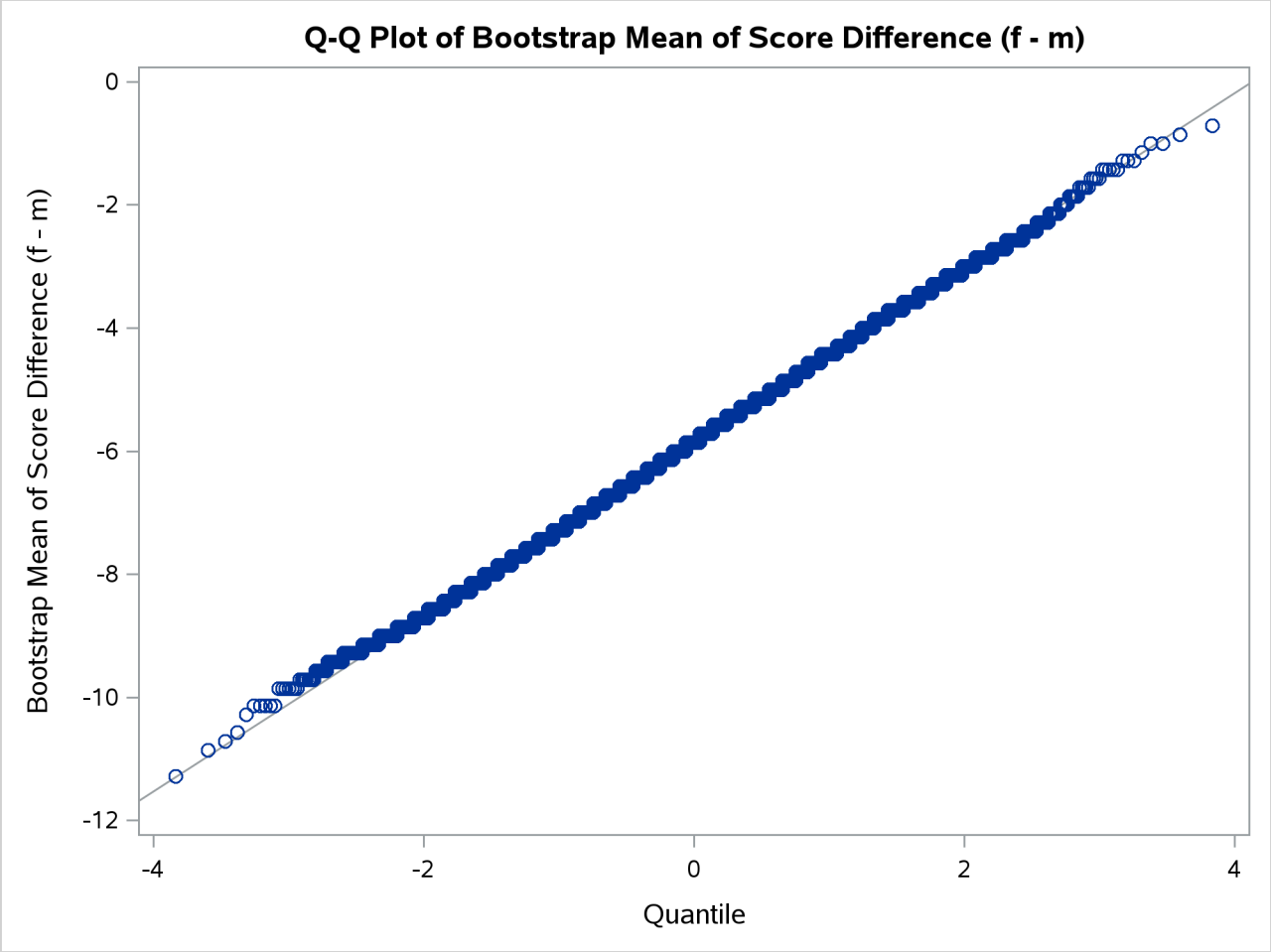

Output 128.6.8 Q-Q Plot of Bootstrap Pooled Standard Deviation

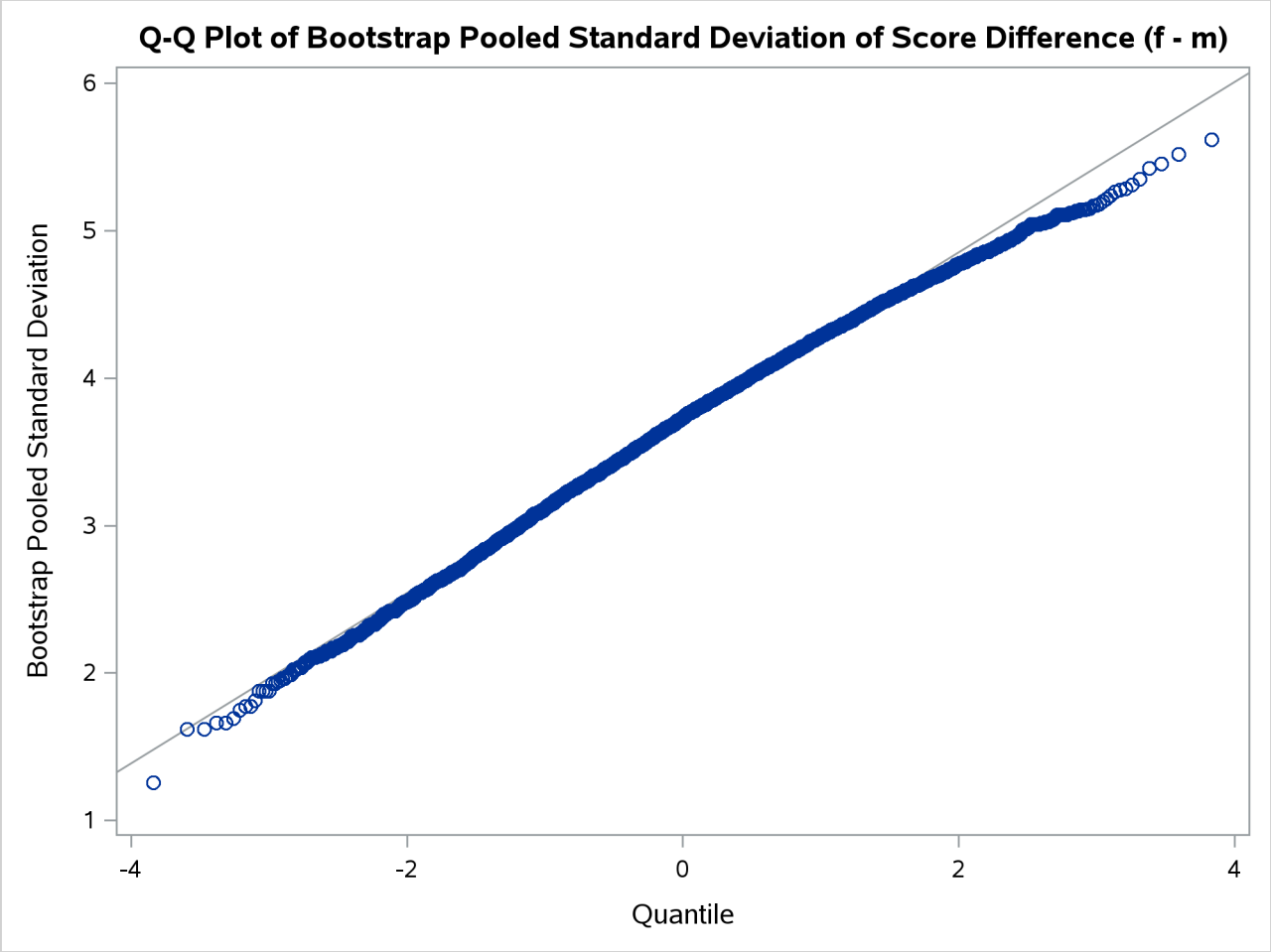

Output 128.6.9 Q-Q Plot of Bootstrap Satterthwaite Standard Deviation

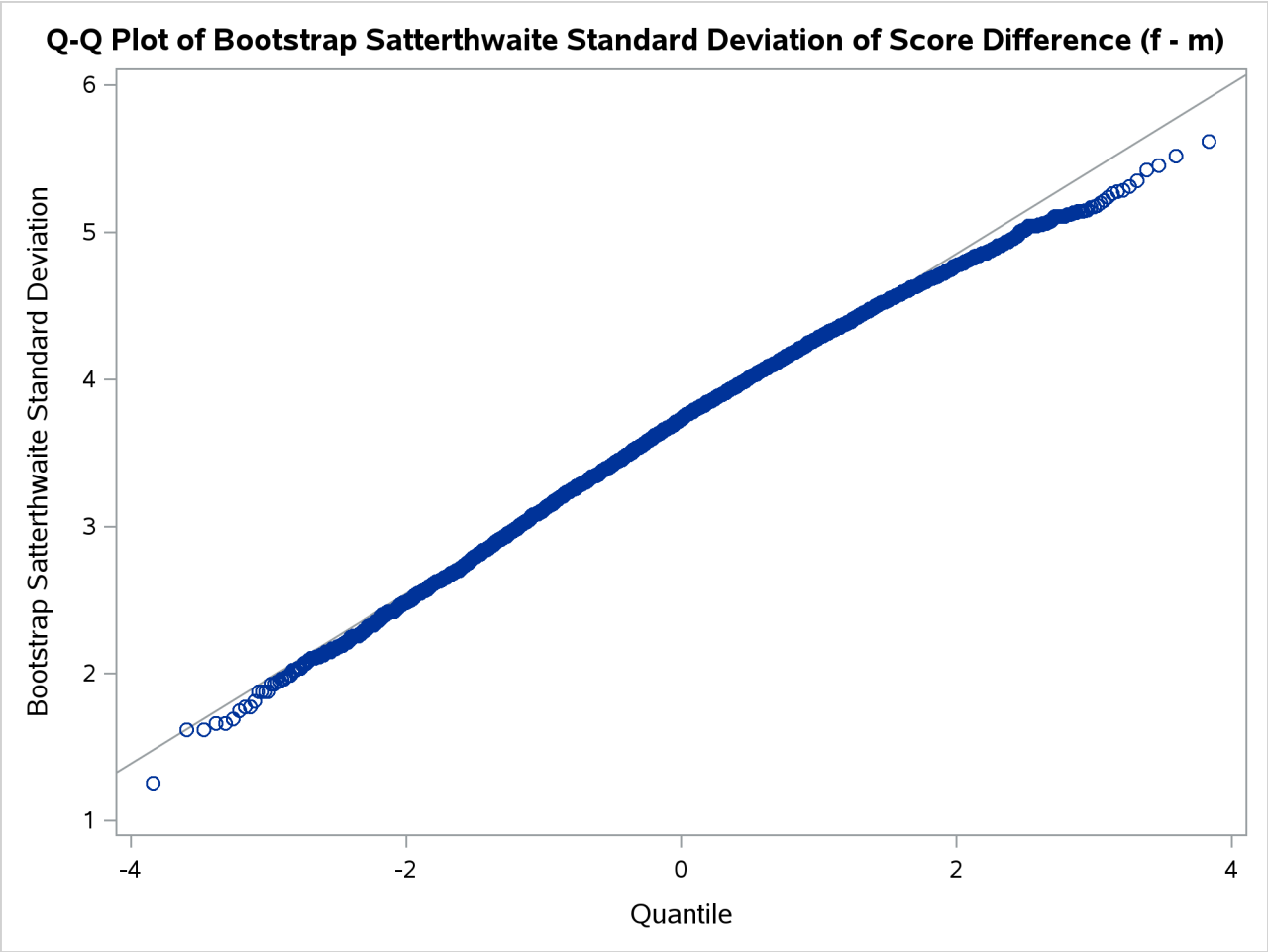

The bootstrap correlation plots in Output 128.6.10 and Output 128.6.11 show the relationship between the bootstrap mean and the bootstrap pooled and Satterthwaite standard deviations of the difference between female and male scores. The approximately circular (as opposed to elongated elliptical) shapes of the prediction regions suggest unusually low correlations between the mean and standard deviation.

Output 128.6.10 Correlation Plot of Bootstrap Mean and Pooled Standard Deviation

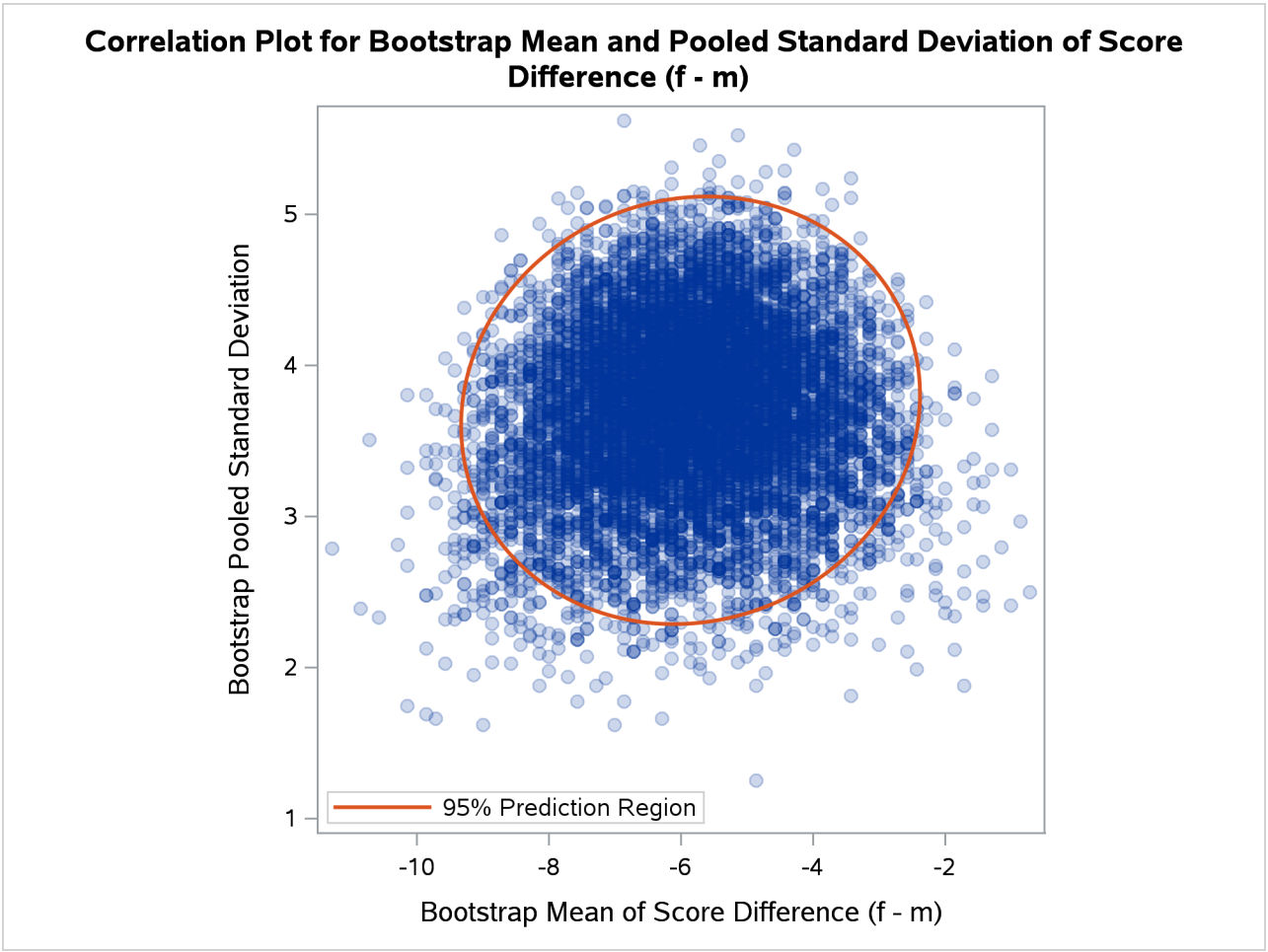

Output 128.6.11 Correlation Plot of Bootstrap Mean and Satterthwaite Standard Deviation

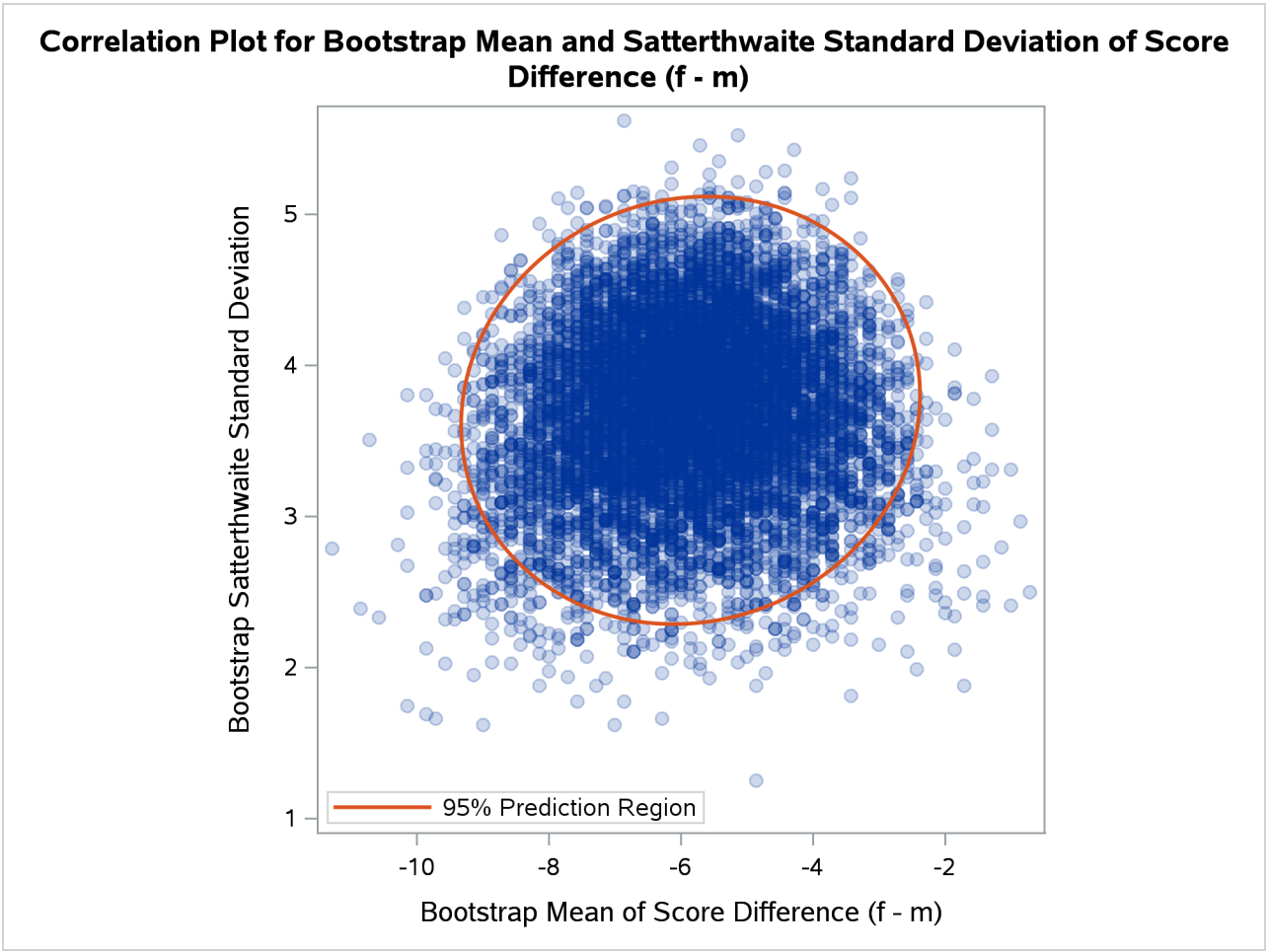

The bootstrap interval plots in Output 128.6.12 and Output 128.6.13 show $100(1-\alpha)\% = 95\%$ bootstrap bias-corrected percentile intervals of the mean, pooled standard deviation, and Satterthwaite standard deviation of the difference between female and male scores. The distance and direction between the sample and bootstrap means that are marked on the interval for each statistic reflect the size and sign of the bootstrap-estimated bias of the sample mean of the statistic.

Output 128.6.12 Bootstrap Bias-Corrected Percentile Interval Plot of Mean

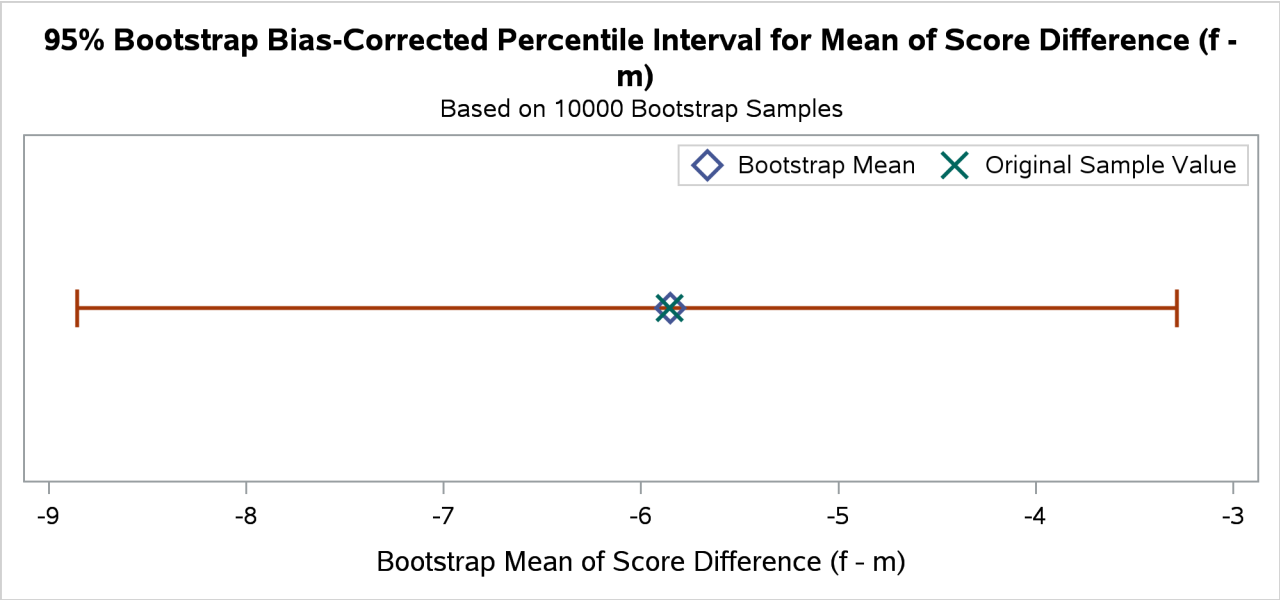

Output 128.6.13 Bootstrap Bias-Corrected Percentile Interval Plot of Pooled and Satterthwaite Standard Deviations

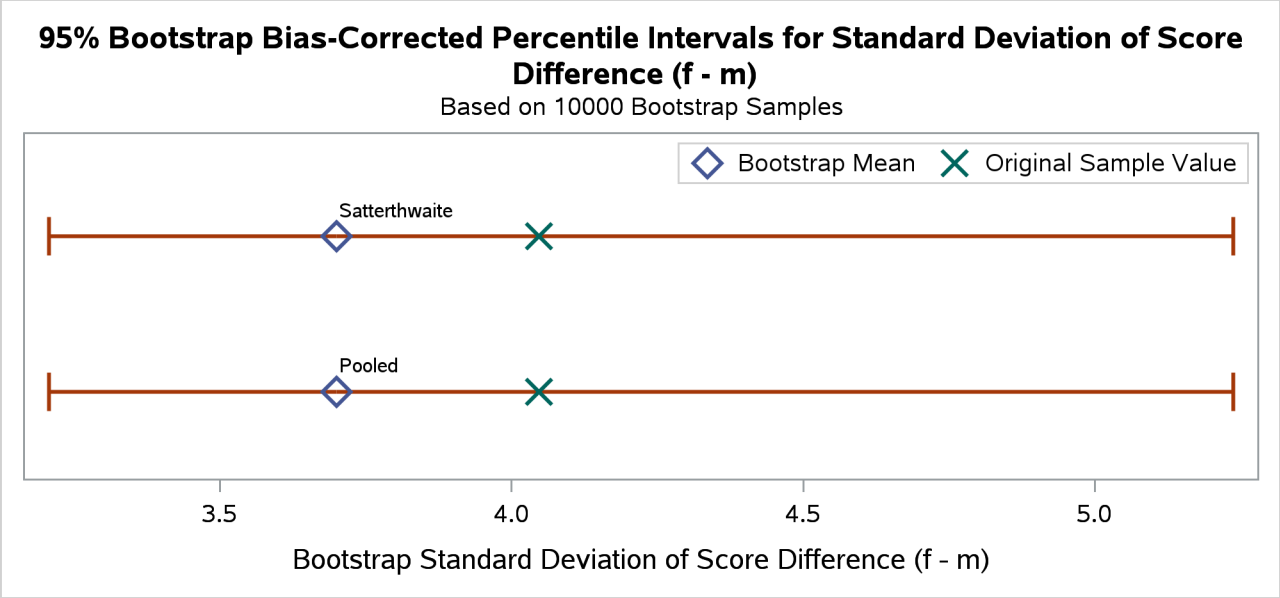

For the purpose of illustration, it is informative to compare the bootstrap bias-corrected percentile intervals to both unadjusted and expanded percentile intervals. The following SAS statements request unadjusted bootstrap percentile intervals:

```
proc ttest plots(only)=bootstrap(interval);
   class Gender;
   var Score;
   bootstrap / seed=810 bootci=percentile;
```

```
run;
```

Output 128.6.14 through Output 128.6.16 show very similar confidence limits for the mean difference but substantially lower limits for the standard deviation of the difference, compared to the bootstrap bias-corrected confidence limits in Output 128.6.3, Output 128.6.12, and Output 128.6.13. This is consistent with both the magnitudes and directions of the bootstrap bias estimates. The minor differences in the bootstrap standard error and bias estimates between Output 128.6.3 and Output 128.6.14 are due only to random variation in the bootstrap resampling.

Output 128.6.14 Bootstrap Analysis with Percentile Intervals

The TTEST Procedure

Variable: Score

Bootstrap Statistics and Confidence Limits

Gender	Method	Parameter	Std Error	Bias	95% CL	
Diff (1-2)		Mean	1.4046	-0.00934	-8.5714	-3.1429
Diff (1-2)	Pooled	Std Dev	0.5757	-0.3373	2.5355	4.7809
Diff (1-2)	Satterthwaite	Std Dev	0.5757	-0.3373	2.5355	4.7809

Output 128.6.15 Bootstrap Percentile Interval Plot of Mean

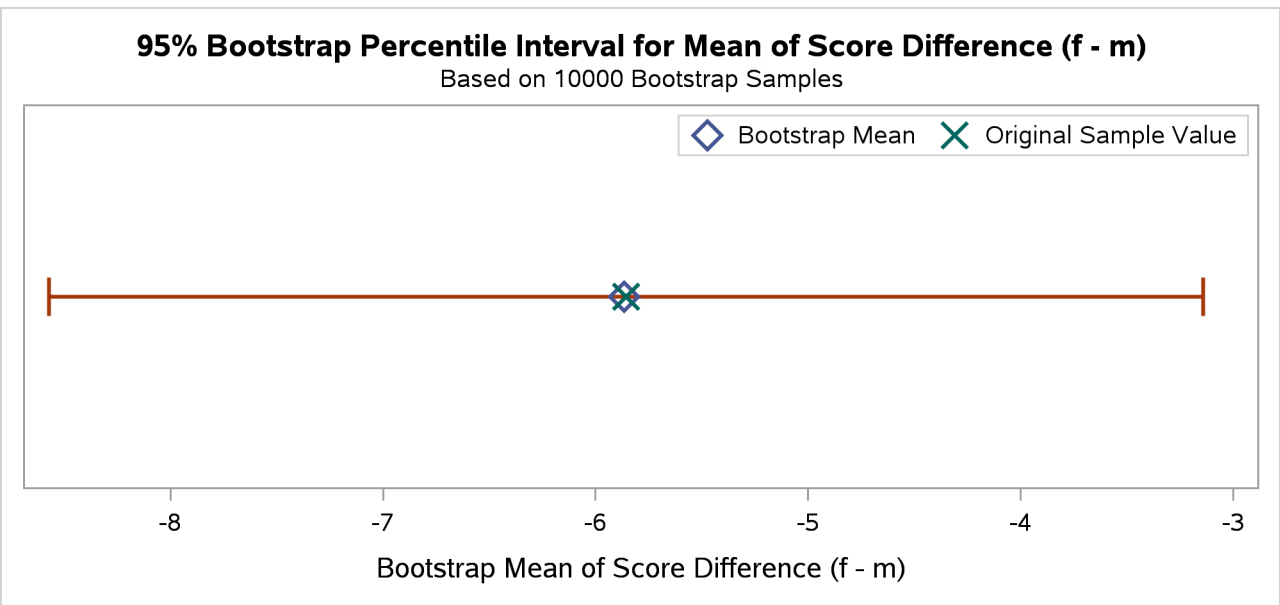

Output 128.6.16 Bootstrap Percentile Interval Plot of Pooled and Satterthwaite Standard Deviations

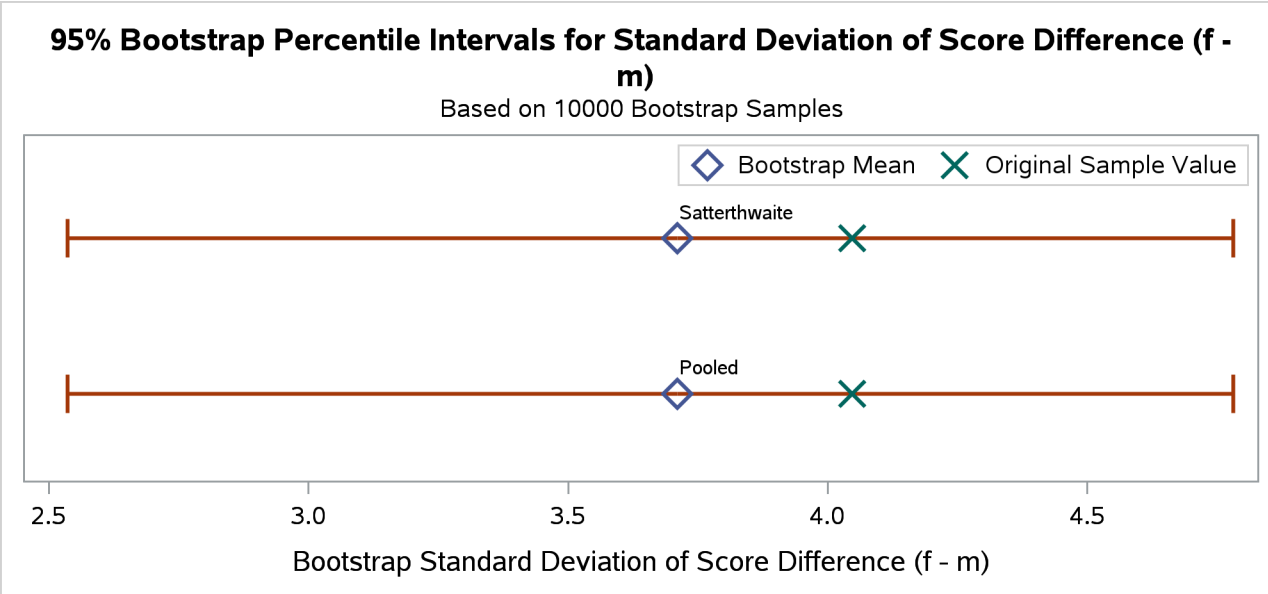

The following SAS statements request bootstrap expanded percentile intervals:

```
proc ttest plots(only)=bootstrap(interval);
   class Gender;
   var Score;
   bootstrap / seed=249 bootci=expandedperc;
run;

ods graphics off;
```

Output 128.6.17 and Output 128.6.18 show a substantially wider confidence interval for the mean difference when compared to Output 128.6.14 through Output 128.6.16. This makes sense because the expanded percentile interval, unlike the bias-corrected confidence interval, corrects for narrowness bias, which is especially problematic for small sample sizes.

Output 128.6.17 Bootstrap Analysis with Expanded Percentile Intervals

The TTEST Procedure

Variable: Score

Bootstrap Statistics and Confidence Limits

Gender	Method	Parameter	Std Error	Bias	95% CL	
Diff (1-2)	Pooled	Mean	1.4024	-0.0265	-9.0000	-2.5714
Diff (1-2)	Satterthwaite	Mean	1.4024	-0.0265	-9.1429	-2.5714
Diff (1-2)	Pooled	Std Dev	0.5817	-0.3515		
Diff (1-2)	Satterthwaite	Std Dev	0.5817	-0.3515		

Output 128.6.18 Bootstrap Expanded Percentile Interval Plot of Mean

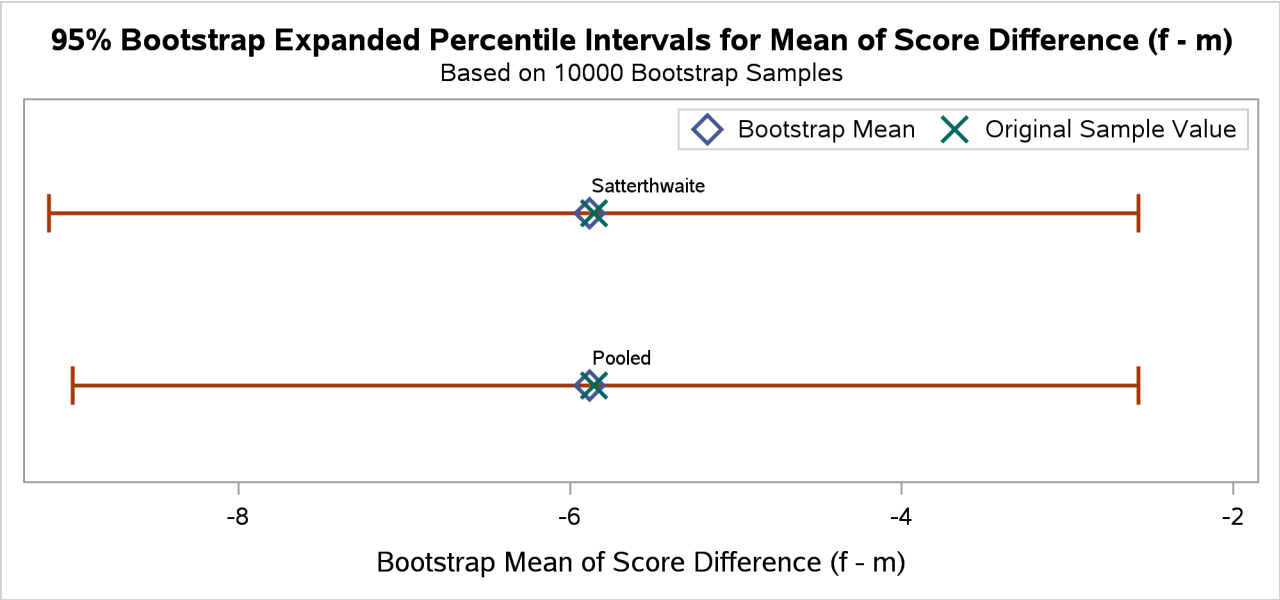

There are distinct pooled and Satterthwaite versions of the expanded percentile intervals for the mean in Output 128.6.17 and Output 128.6.18, because the two methods compute degrees of freedom differently and the degrees of freedom are involved in the "expanded" adjustment.

Expanded percentile intervals are not computed for the standard deviation parameters because the method that is specified in the BOOTCI=EXPANDEDPERC option is based on the assumption of approximate normality of the bootstrapped statistic $\hat{\theta}$. However, both the pooled and Satterthwaite standard deviation estimates have approximate chi-square distributions under normality. Because they are not sample mean estimates, they are not subject to the central limit theorem.

References

Best, D. I., and Rayner, C. W. (1987). "Welch's Approximate Solution for the Behrens-Fisher Problem." *Technometrics* 29:205–210.

Carpenter, J., and Bithell, J. (2000). "Bootstrap Confidence Intervals: When, Which, What? A Practical Guide for Medical Statisticians." *Statistics in Medicine* 19:1141–1164.

Chow, S.-C., and Liu, J.-P. (2000). *Design and Analysis of Bioavailability and Bioequivalence Studies*. 2nd ed. New York: Marcel Dekker.

Cochran, W. G., and Cox, G. M. (1950). *Experimental Designs*. New York: John Wiley & Sons.

Dilba, G., Schaarschmidt, F., and Hothorn, L. A. (2007). "Inferences for Ratios of Normal Means." *R News* 7:20–23.

Diletti, D., Hauschke, D., and Steinijans, V. W. (1991). "Sample Size Determination for Bioequivalence Assessment by Means of Confidence Intervals." *International Journal of Clinical Pharmacology, Therapy, and Toxicology* 29:1–8.

Efron, B., and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. New York: Chapman & Hall.

Fieller, E. C. (1954). "Some Problems in Interval Estimation." *Journal of the Royal Statistical Society, Series B* 16:175–185.

Hauschke, D., Kieser, M., Diletti, E., and Burke, M. (1999). "Sample Size Determination for Proving Equivalence Based on the Ratio of Two Means for Normally Distributed Data." *Statistics in Medicine* 18:93–105.

Hesterberg, T. C. (2015). "What Teachers Should Know about the Bootstrap: Resampling in the Undergraduate Statistics Curriculum." *American Statistician* 69:371–386.

Huntsberger, D. V., and Billingsley, P. (1989). *Elements of Statistical Inference*. Dubuque, IA: Wm. C. Brown.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1994). *Continuous Univariate Distributions.* 2nd ed. Vol. 1. New York: John Wiley & Sons.

Jones, B., and Kenward, M. G. (2003). *Design and Analysis of Cross-Over Trials*. 2nd ed. Boca Raton, FL: Chapman & Hall/CRC.

Lee, A. F. S., and Gurland, J. (1975). "Size and Power of Tests for Equality of Means of Two Normal Populations with Unequal Variances." *Journal of the American Statistical Association* 70:933–941.

Lehmann, E. L. (1986). *Testing Statistical Hypotheses*. New York: John Wiley & Sons.

Moore, D. S. (1995). *The Basic Practice of Statistics*. New York: W. H. Freeman.

Phillips, K. F. (1990). "Power of the Two One-Sided Tests Procedure in Bioequivalence." *Journal of Pharmacokinetics and Biopharmaceutics* 18:137–144.

Politis, D. (2016). "t and χ^2: Revisiting the Classical Tests for the 21st Century Classroom." *IMS Bulletin* 45:10–11.

Posten, H. O., Yeh, Y. Y., and Owen, D. B. (1982). "Robustness of the Two-Sample t Test under Violations of the Homogeneity of Variance Assumption." *Communications in Statistics—Theory and Methods* 11:109–126.

Ramsey, P. H. (1980). "Exact Type I Error Rates for Robustness of Student's t Test with Unequal Variances." *Journal of Educational Statistics* 5:337–349.

Robinson, G. K. (1976). "Properties of Student's t and of the Behrens-Fisher Solution to the Two Mean Problem." *Annals of Statistics* 4:963–971.

SAS Institute Inc. (1986). *SUGI Supplemental Library User's Guide, Version 5 Edition*. Cary, NC: SAS Institute Inc.

Sasabuchi, S. (1988a). "A Multivariate Test with Composite Hypotheses Determined by Linear Inequalities When the Covariance Matrix Has an Unknown Scale Factor." *Memoirs of the Faculty of Science, Kyushu University, Series A* 42:9–19.

Sasabuchi, S. (1988b). "A Multivariate Test with Composite Hypotheses When the Covariance Matrix Is Completely Unknown." *Memoirs of the Faculty of Science, Kyushu University, Series A* 42:37–46.

Satterthwaite, F. E. (1946). "An Approximate Distribution of Estimates of Variance Components." *Biometrics Bulletin* 2:110–114.

Scheffé, H. (1970). "Practical Solutions of the Behrens-Fisher Problem." *Journal of the American Statistical Association* 65:1501–1508.

Schuirmann, D. J. (1987). "A Comparison of the Two One-Sided Tests Procedure and the Power Approach for Assessing the Equivalence of Average Bioavailability." *Journal of Pharmacokinetics and Biopharmaceutics* 15:657–680.

Senn, S. (2002). *Cross-Over Trials in Clinical Research*. 2nd ed. New York: John Wiley & Sons.

Steel, R. G. D., and Torrie, J. H. (1980). *Principles and Procedures of Statistics*. 2nd ed. New York: McGraw-Hill.

Tamhane, A. C., and Logan, B. R. (2004). "Finding the Maximum Safe Dose Level for Heteroscedastic Data." *Journal of Biopharmaceutical Statistics* 14:843–856.

Wang, Y. Y. (1971). "Probabilities of the Type I Error of the Welch Tests for the Behrens-Fisher Problem." *Journal of the American Statistical Association* 66:605–608.

Wellek, S. (2003). *Testing Statistical Hypotheses of Equivalence*. Boca Raton, FL: Chapman & Hall/CRC.

Yuen, K. K. (1974). "The Two-Sample Trimmed *t* for Unequal Population Variances." *Biometrika* 61:165–170.

Chapter 129
The VARCLUS Procedure

Contents

Overview: VARCLUS Procedure **10902**
Getting Started: VARCLUS Procedure **10904**
Syntax: VARCLUS Procedure **10908**
- PROC VARCLUS Statement 10908
- BY Statement 10916
- FREQ Statement 10917
- PARTIAL Statement 10917
- SEED Statement 10917
- VAR Statement 10917
- WEIGHT Statement 10918

Details: VARCLUS Procedure **10918**
- Missing Values 10918
- Using the VARCLUS procedure 10918
- Output Data Sets 10919
- Computational Resources 10921
- Interpreting VARCLUS Procedure Output 10922
- Displayed Output 10922
- ODS Table Names 10924
- ODS Graphics 10924

Example: VARCLUS Procedure **10925**
- Example 129.1: Correlations among Physical Variables 10925

References **10932**

Overview: VARCLUS Procedure

The VARCLUS procedure divides a set of numeric variables into disjoint or hierarchical clusters. Associated with each cluster is a linear combination of the variables in the cluster. This linear combination can be either the first principal component (the default) or the centroid component (if you specify the CENTROID option). The first principal component is a weighted average of the variables that explains as much variance as possible. See Chapter 98, "The PRINCOMP Procedure," for further details. Centroid components are unweighted averages of either the standardized variables (the default) or the raw variables (if you specify the COVARIANCE option). PROC VARCLUS tries to maximize the variance that is explained by the cluster components, summed over all the clusters.

The cluster components are oblique, not orthogonal, even when the cluster components are first principal components. In an ordinary principal component analysis, all components are computed from the same variables, and the first principal component is orthogonal to the second principal component and to every other principal component. In PROC VARCLUS, each cluster component is computed from a set of variables that is different from all the other cluster components. The first principal component of one cluster might be correlated with the first principal component of another cluster. Hence, the PROC VARCLUS algorithm is a type of oblique component analysis.

As in principal component analysis, either the correlation or the covariance matrix can be analyzed. If correlations are used, all variables are treated as equally important. If covariances are used, variables with larger variances have more importance in the analysis.

PROC VARCLUS displays a dendrogram (tree diagram of hierarchical clusters) by using ODS Graphics. PROC VARCLUS can also can create an output data set that can be used by the TREE procedure to draw the dendrogram. A second output data set can be used with the SCORE procedure to compute component scores for each cluster.

PROC VARCLUS can be used as a variable-reduction method. A large set of variables can often be replaced by the set of cluster components with little loss of information. A given number of cluster components does not generally explain as much variance as the same number of principal components on the full set of variables, but the cluster components are usually easier to interpret than the principal components, even if the latter are rotated.

For example, an educational test might contain 50 items. PROC VARCLUS can be used to divide the items into, say, five clusters. Each cluster can then be treated as a subtest, with the subtest scores given by the cluster components. If the cluster components are centroid components of the covariance matrix, each subtest score is simply the sum of the item scores for that cluster.

The VARCLUS algorithm is both divisive and iterative. By default, PROC VARCLUS begins with all variables in a single cluster. It then repeats the following steps:

1. A cluster is chosen for splitting. Depending on the options specified, the selected cluster has either the smallest percentage of variation explained by its cluster component (using the PROPORTION= option) or the largest eigenvalue associated with the second principal component (using the MAXEIGEN= option).

2. The chosen cluster is split into two clusters by finding the first two principal components, performing an orthoblique rotation (raw quartimax rotation on the eigenvectors; Harris and Kaiser 1964), and assigning each variable to the rotated component with which it has the higher squared correlation.

3. Variables are iteratively reassigned to clusters to try to maximize the variance accounted for by the cluster components. You can require the reassignment algorithms to maintain a hierarchical structure for the clusters.

The procedure stops splitting when either of the following conditions holds:

- The number of clusters is greater than or equal to the maximum number of clusters as specified by the MAXCLUSTERS= option is reached.
- Every cluster satisfies the stopping criteria specified by the PROPORTION= option (percentage of variation explained) or the MAXEIGEN= option (second eigenvalue) or both.

By default, VARCLUS stops splitting when every cluster has only one eigenvalue greater than one, thus satisfying the most popular criterion for determining the sufficiency of a single underlying dimension.

The iterative reassignment of variables to clusters proceeds in two phases. The first is a nearest component sorting (NCS) phase, similar in principle to the nearest centroid sorting algorithms described by Anderberg (1973). In each iteration, the cluster components are computed, and each variable is assigned to the component with which it has the highest squared correlation. The second phase involves a search algorithm in which each variable is tested to see if assigning it to a different cluster increases the amount of variance explained. If a variable is reassigned during the search phase, the components of the two clusters involved are recomputed before the next variable is tested. The NCS phase is much faster than the search phase but is more likely to be trapped by a local optimum.

If principal components are used, the NCS phase is an alternating least squares method and converges rapidly. The search phase can be very time-consuming for a large number of variables. But if the default initialization method is used, the search phase is rarely able to substantially improve the results of the NCS phase, so the search takes few iterations. If random initialization is used, the NCS phase might be trapped by a local optimum from which the search phase can escape.

If centroid components are used, the NCS phase is not an alternating least squares method and might not increase the amount of variance explained; therefore it is limited, by default, to one iteration.

You can have VARCLUS do the clustering hierarchically by restricting the reassignment of variables such that the clusters maintain a tree structure. In this case, when a cluster is split, a variable in one of the two resulting clusters can be reassigned to the other cluster that results from the split but not to a cluster that is not part of the original cluster (the one that is split).

Getting Started: VARCLUS Procedure

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/varclugs.sas.)

This example demonstrates how you can use PROC VARCLUS to cluster variables.

The following data are job ratings of police officers. The officers were rated by their supervisors on 13 job skills on a scale from 1 to 9. There is also an overall rating that is not used in this analysis. The following DATA step creates the SAS data set JobRat:

```
data JobRat;
   input
      (Communication_Skills
       Problem_Solving
       Learning_Ability
       Judgement_under_Pressure
       Observational_Skills
       Willingness_to_Confront_Problems
       Interest_in_People
       Interpersonal_Sensitivity
       Desire_for_Self_Improvement
       Appearance
       Dependability
       Physical_Ability
       Integrity
       Overall_Rating)
      (1.);
   datalines;
26838853879867
74758876857667
56757863775875
67869777988997
99997798878888

   ... more lines ...

99997899799799
99899899899899
76656399567486
;
```

The following statements cluster the variables:

```
proc varclus data=JobRat maxclusters=3;
   var Communication_Skills--Integrity;
run;
```

The DATA= option specifies the SAS data set JobRat as input.

The MAXCLUSTERS=3 option specifies that no more than three clusters be computed. By default, PROC VARCLUS splits and optimizes clusters until all clusters have a second eigenvalue less than one. In this example, the default setting would produce only two clusters, but going to three clusters produces a more

interesting result.

The VAR statement lists the numeric variables (Communication_Skills -- Integrity) to be used in the analysis. The overall rating is omitted from the list of variables.

Although PROC VARCLUS displays output for one cluster, two clusters, and three clusters, the following figures display only the final analysis for three clusters.

For each cluster, Figure 129.1 displays the number of variables in the cluster, the cluster variation, the total explained variation, and the proportion of the total variance explained by the variables in the cluster. The variance explained by the variables in a cluster is similar to the variance explained by a factor in common factor analysis, but it includes contributions only from the variables in the cluster rather than from all variables.

The line labeled "Total variation explained" in Figure 129.1 gives the sum of the explained variation over all clusters. The final "Proportion" represents the total explained variation divided by the sum of cluster variation. This value, 0.6715, indicates that about 67% of the total variation in the data can be accounted for by the three cluster components.

Figure 129.1 Cluster Summary for Three Clusters from PROC VARCLUS

Oblique Principal Component Cluster Analysis

Cluster Summary for 3 Clusters

Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained	Second Eigenvalue
1	6	6	3.771349	0.6286	0.7093
2	5	5	3.575933	0.7152	0.5035
3	2	2	1.382005	0.6910	0.6180

Total variation explained = 8.729286 Proportion = 0.6715

Figure 129.2 shows how the variables are clustered. Figure 129.2 also displays the R-square value of each variable with its own cluster and the R-square value with its nearest cluster. The R-square value for a variable with the nearest cluster should be low if the clusters are well separated. The last column displays the ratio of $(1 - R^2_{own})/(1 - R^2_{nearest})$ for each variable. Small values of this ratio indicate good clustering.

Figure 129.2 R-Square Values from PROC VARCLUS

3 Clusters		R-squared with		
Cluster	Variable	Own Cluster	Next Closest	1-R**2 Ratio
Cluster 1	Communication_Skills	0.6403	0.3599	0.5620
	Problem_Solving	0.5412	0.2895	0.6458
	Learning_Ability	0.6561	0.1692	0.4139
	Observational_Skills	0.6889	0.2584	0.4194
	Willingness_to_Confront_Problems	0.6480	0.3402	0.5335
	Desire_for_Self_Improvement	0.5968	0.3473	0.6177
Cluster 2	Judgement_under_Pressure	0.6263	0.3719	0.5950
	Interest_in_People	0.8122	0.1885	0.2314
	Interpersonal_Sensitivity	0.7566	0.1387	0.2826
	Dependability	0.6163	0.4419	0.6875
	Integrity	0.7645	0.2724	0.3237
Cluster 3	Appearance	0.6910	0.3047	0.4444
	Physical_Ability	0.6910	0.1871	0.3801

Figure 129.3 displays the standardized scoring coefficients that are used to compute the first principal component of each cluster. Since each variable is assigned to one and only one cluster, each row of the scoring coefficients contains only one nonzero value.

Figure 129.3 Standardized Scoring Coefficients from PROC VARCLUS

Standardized Scoring Coefficients			
Cluster	1	2	3
Communication_Skills	0.212170	0.000000	0.000000
Problem_Solving	0.195058	0.000000	0.000000
Learning_Ability	0.214781	0.000000	0.000000
Judgement_under_Pressure	0.000000	0.221313	0.000000
Observational_Skills	0.220086	0.000000	0.000000
Willingness_to_Confront_Problems	0.213452	0.000000	0.000000
Interest_in_People	0.000000	0.252025	0.000000
Interpersonal_Sensitivity	0.000000	0.243245	0.000000
Desire_for_Self_Improvement	0.204848	0.000000	0.000000
Appearance	0.000000	0.000000	0.601493
Dependability	0.000000	0.219544	0.000000
Physical_Ability	0.000000	0.000000	0.601493
Integrity	0.000000	0.244507	0.000000

Figure 129.4 displays the cluster structure and the intercluster correlations. The structure table displays the correlation of each variable with each cluster component. The table of intercorrelations contains the correlations between the cluster components.

Figure 129.4 Cluster Correlations and Intercorrelations from PROC VARCLUS

Cluster Structure			
Cluster	**1**	**2**	**3**
Communication_Skills	0.800169	0.599909	0.427341
Problem_Solving	0.735630	0.538017	0.425463
Learning_Ability	0.810014	0.411316	0.376333
Judgement_under_Pressure	0.609876	0.791401	0.345399
Observational_Skills	0.830021	0.407807	0.508305
Willingness_to_Confront_Problems	0.805002	0.362927	0.583265
Interest_in_People	0.434138	0.901225	0.387770
Interpersonal_Sensitivity	0.372371	0.869826	0.287658
Desire_for_Self_Improvement	0.772554	0.589334	0.494842
Appearance	0.552003	0.393759	0.831266
Dependability	0.664778	0.785073	0.574460
Physical_Ability	0.432590	0.416070	0.831266
Integrity	0.521876	0.874342	0.477885

Inter-Cluster Correlations			
Cluster	**1**	**2**	**3**
1	1.00000	0.60851	0.59223
2	0.60851	1.00000	0.48711
3	0.59223	0.48711	1.00000

PROC VARCLUS next displays the summary table of statistics for the cluster history (Figure 129.5). The first three columns give the number of clusters, the total variation explained by clusters, and the proportion of variation explained by clusters, respectively.

As displayed in the first row of Figure 129.5, the variation explained by the first principal component of all the variables is 6.547402, and the proportion of variation explained is 0.5036.

When the number of clusters is two, the total variation explained is 7.96775 and the proportion of variation explained by the two clusters is 0.6129. The larger second eigenvalue of the clusters is 0.937902; so by default, PROC VARCLUS would stop splitting clusters at this point. But because the MAXCLUSTERS=3 option was specified in this example, PROC VARCLUS continues to the three-cluster solution.

When the number of clusters increases to three, the total variation explained is 8.729286 and the proportion of variation explained by the two clusters is 0.6715. The largest second eigenvalue of the clusters is 0.709323. The statistical improvement from increasing the number of clusters from two to three seems modest, but the interpretability of the three clusters argues for the three-cluster solution.

Figure 129.5 also displays the minimum proportion of variance explained by a cluster, the minimum R square for a variable, and the maximum $(1 - R^2)$ ratio for a variable. The last quantity is the maximum ratio of the value $1 - R^2$ for a variable's own cluster to the value $1 - R^2$ for its nearest cluster.

Figure 129.5 Final Cluster Summary Table from PROC VARCLUS

Number of Clusters	Total Variation Explained by Clusters	Proportion of Variation Explained by Clusters	Minimum Proportion Explained by a Cluster	Maximum Second Eigenvalue in a Cluster	Minimum R-squared for a Variable	Maximum 1-R**2 Ratio for a Variable
1	6.547402	0.5036	0.5036	1.772715	0.2995	
2	7.967753	0.6129	0.5475	0.937902	0.3123	0.8026
3	8.729286	0.6715	0.6286	0.709323	0.5412	0.6875

Syntax: VARCLUS Procedure

The following statements are available in the VARCLUS procedure:

PROC VARCLUS < *options* > **;**
  VAR *variables* **;**
  SEED *variables* **;**
  PARTIAL *variables* **;**
  WEIGHT *variables* **;**
  FREQ *variables* **;**
  BY *variables* **;**

Usually you need only the VAR statement in addition to the PROC VARCLUS statement. The following sections give detailed syntax information about each of the statements, beginning with the PROC VARCLUS statement. The remaining statements are listed in alphabetical order.

PROC VARCLUS Statement

PROC VARCLUS < *options* > **;**

The PROC VARCLUS statement invokes the VARCLUS procedure. By default, VARCLUS clusters the numeric variables in the most recently created SAS data set, starting with one cluster and splitting clusters until all clusters have at most one eigenvalue greater than one.

Table 129.1 summarizes the *options* available in the PROC VARCLUS statement.

Table 129.1 Options Available in the PROC VARCLUS Statement

Option	Description
Data Sets	
DATA=	Specifies the input SAS data set
OUTSTAT=	Specifies the output SAS data set to contain statistics
OUTTREE=	Specifies the output SAS data set for use with PROC TREE
Input Data Processing	
COVARIANCE	Uses the covariance matrix instead of the correlation matrix

Table 129.1 *continued*

Option	Description
NOINT	Omits the intercept
VARDEF=	Specifies the divisor for variances
Number of Clusters	
MAXCLUSTERS=	Specifies the maximum number of clusters
MINCLUSTERS=	Specifies the minimum number of clusters
MAXEIGEN=	Specifies the maximum second eigenvalue in a cluster
PROPORTION=	Specifies the minimum proportion of variance explained by a cluster component
Clustering Methods	
CENTROID	Uses centroid components instead of principal components
HIERARCHY	Clusters hierarchically
INITIAL=	Specifies the initialization method
MAXITER=	Specifies the maximum iterations during the alternating least squares phase
MAXSEARCH=	Specifies the maximum iterations during the search phase
MULTIPLEGROUP	Performs a multiple group component analysis
RANDOM=	Specifies the random number seed
Control Displayed Output	
CORR	Displays the correlation matrix
NOPRINT	Suppresses displayed output
PLOTS=	Specifies ODS Graphics details
SHORT	Suppresses display of large matrices
SIMPLE	Displays means and standard deviations
SUMMARY	Suppresses all default displayed output except the final summary table
TRACE	Displays the cluster to which each variable is assigned during the iterations

VARCLUS chooses which cluster to split based on the MAXEIGEN= and PROPORTION= options.

1. If you specify *either* or *both* of these two options, then *only* the specified options affect the choice of the cluster to split.
2. If you specify *neither* of these options, the criterion for choice of cluster to split depends on the CENTROID option:
   a) If you specify CENTROID, VARCLUS splits the cluster with the smallest percentage of variation explained by its cluster component, as if you had specified the PROPORTION= option.
   b) If you do not specify CENTROID, VARCLUS splits the cluster with the largest eigenvalue associated with the second principal component, as if you had specified the MAXEIGEN= option.

The final number of clusters is controlled by three options: MAXCLUSTERS=, MAXEIGEN=, and PROPORTION=.

1. If you specify *any* of these three options, then *only* the options you specify affect the final number of clusters.

2. If you specify *none* of these options, VARCLUS continues to split clusters until the default splitting criterion is satisfied. The default splitting criterion depends on the CENTROID option:

   a) If you specify CENTROID, the default splitting criterion is PROPORTION=0.75.

   b) If you do not specify CENTROID, splitting is based on the MAXEIGEN= criterion, with a default depending on the COVARIANCE option:

      i. For analyzing a correlation matrix (no COVARIANCE option), the default value for MAXEIGEN= is one.

      ii. For analyzing a covariance matrix (using the COVARIANCE option), the default value for MAXEIGEN= is the average variance of the variables being clustered.

VARCLUS continues to split clusters until any of the following conditions holds:

- The number of cluster equals the value specified for MAXCLUSTERS=.
- No cluster qualifies for splitting according to the MAXEIGEN= or PROPORTION= criterion.
- A cluster was chosen for splitting, but after iteratively reassigning variables to clusters, one of the cluster has no members.

The following list gives details about the *options*.

CENTROID

uses centroid components rather than principal components. You should specify centroid components if you want the cluster components to be unweighted averages of the standardized variables (the default) or the unstandardized variables (if you specify the COVARIANCE option). It is possible to obtain locally optimal clusterings in which a variable is not assigned to the cluster component with which it has the highest squared correlation. You cannot specify both the CENTROID and MAXEIGEN= options.

CORR

C

displays the correlation matrix.

COVARIANCE

COV

analyzes the covariance matrix instead of the correlation matrix. The COVARIANCE option causes variables with a large variance to have more effect on the cluster components than variables with a small variance.

DATA=*SAS-data-set*

specifies the input data set to be analyzed. The data set can be an ordinary SAS data set or TYPE=CORR, UCORR, COV, UCOV, FACTOR, or SSCP. If you do not specify the DATA= option, the most recently created SAS data set is used. See Appendix A, "Special SAS Data Sets," for more information about types of SAS data sets.

HIERARCHY

HI

requires the clusters at different levels to maintain a hierarchical structure. To draw a tree diagram, enable ODS Graphics or use the OUTTREE= option and the TREE procedure.

INITIAL=GROUP

INITIAL=INPUT

INITIAL=RANDOM

INITIAL=SEED

specifies the method for initializing the clusters. If the INITIAL= option is omitted and the MINCLUSTERS= option is greater than 1, the initial cluster components are obtained by extracting the required number of principal components and performing an orthoblique rotation (raw quartimax rotation on the eigenvectors; Harris and Kaiser 1964). The following list describes the values for the INITIAL= option:

GROUP obtains the cluster membership of each variable from an observation in the DATA= data set where the _TYPE_ variable has a value of 'GROUP'. In this observation, the variables to be clustered must each have an integer value ranging from one to the number of clusters. You can use this option only if the DATA= data set is a TYPE=CORR, UCORR, COV, UCOV, or FACTOR data set. You can use a data set created either by a previous run of PROC VARCLUS or in a DATA step.

INPUT obtains scoring coefficients for the cluster components from observations in the DATA= data set where the _TYPE_ variable has a value of 'SCORE'. You can use this option only if the DATA= data set is a TYPE=CORR, UCORR, COV, UCOV, or FACTOR data set. You can use scoring coefficients from the FACTOR procedure or a previous run of PROC VARCLUS, or you can enter other coefficients in a DATA step.

RANDOM assigns variables randomly to clusters.

SEED initializes each cluster component to be one of the variables named in the SEED statement. Each variable listed in the SEED statement becomes the sole member of a cluster, and the other variables are initially unassigned. If you do not specify the SEED statement, the first MINCLUSTERS= variables in the VAR statement are used as seeds.

MAXCLUSTERS=*n*

MAXC=*n*

specifies the largest number of clusters desired. The default value is the number of variables. VARCLUS stops splitting clusters after the number of clusters reaches the value of the MAXCLUSTERS= option, regardless of what other splitting options are specified.

MAXEIGEN=*n*

specifies that when choosing a cluster to split, VARCLUS should choose the cluster with the largest second eigenvalue, provided that its second eigenvalue is greater than the MAXEIGEN= value. The MAXEIGEN= option cannot be used with the CENTROID or MULTIPLEGROUP options.

If you do not specify MAXEIGEN=, the default behavior depends on other options as follows:

- If you specify PROPORTION=, CENTROID, or MULTIPLEGROUP, cluster splitting does not depend on the second eigenvalue.
- Otherwise, if you specify MAXCLUSTERS=, the default value for MAXEIGEN= is zero.
- Otherwise, the default value for MAXEIGEN= is either 1.0 if the correlation matrix is analyzed or the average variance if the COVARIANCE option is specified.

If you specify both MAXEIGEN= and MAXCLUSTERS=, the number of clusters will never exceed the value of the MAXCLUSTERS= option.

If you specify both MAXEIGEN= and PROPORTION=, VARCLUS first looks for a cluster to split based on the MAXEIGEN= criterion. If no cluster meets that criterion, VARCLUS then looks for a cluster to split based on the PROPORTION= criterion.

MAXITER=*n*

specifies the maximum number of iterations during the NCS phase. The default value is 1 if you specify the CENTROID option; the default is 10 otherwise.

MAXSEARCH=*n*

specifies the maximum number of iterations during the search phase. The default is 1,000 divided by the number of variables.

MINCLUSTERS=*n*

MINC=*n*

specifies the smallest number of clusters desired. The default value is 2 for INITIAL=RANDOM or INITIAL=SEED; otherwise, VARCLUS begins with one cluster and tries to split it in accordance with the PROPORTION= option or the MAXEIGEN= option or both.

MULTIPLEGROUP

MG

performs a multiple group component analysis (Harman 1976). You specify which variables belong to which clusters. No clusters are split, and no variables are reassigned to a different cluster. The input data set must be TYPE=CORR, UCORR, COV, UCOV, FACTOR, or SSCP and must contain an observation with _TYPE_='GROUP' that defines the variable groups. Specifying the MULTIPLEGROUP option is equivalent to specifying all of the following options: INITIAL=GROUP, MINC=1, MAXITER=0, MAXSEARCH=0, PROPORTION=0, and MAXEIGEN=large number.

NOINT

requests that no intercept be used; covariances or correlations are not corrected for the mean. If you specify the NOINT option, the OUTSTAT= data set is TYPE=UCORR.

NOPRINT

suppresses displayed output. This option temporarily disables the Output Delivery System (ODS). For more information, see Chapter 23, "Using the Output Delivery System."

OUTSTAT=*SAS-data-set*

creates an output data set to contain statistics including means, standard deviations, correlations, cluster scoring coefficients, and the cluster structure. The OUTSTAT= data set is TYPE=UCORR if the NOINT option is specified. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*. For information about types of SAS data sets, see Appendix A, "Special SAS Data Sets."

OUTTREE=*SAS-data-set*

creates an output data set to contain information about the tree structure that can be used by the TREE procedure to display a tree diagram. The OUTTREE= option implies the HIERARCHY option. See Example 129.1 for use of the OUTTREE= option. If you want to create a SAS data set in a permanent library, you must specify a two-level name. For more information about permanent libraries and SAS data sets, see *SAS Programmers Guide: Essentials*.

PLOTS < (*global-plot-options***) > < = *plot-request* >**

PLOTS < (*global-plot-options***) > < = (***plot-request* **< . . . *plot-request* >) >**

controls the plots produced through ODS Graphics.

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc varclus plots=dendrogram(height=ncl);
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

By default, PROC VARCLUS produces a dendrogram.

The *global-plot-options*, UNPACK and ONLY, that are commonly used in the PLOTS= option in other procedures are accepted in PROC VARCLUS, but they currently have no effect since PROC VARCLUS produces only a dendrogram.

The following *plot-requests* can be specified:

ALL

produces all plots, which for PROC VARCLUS is only a dendrogram.

MAXPOINTS=*n*

MAXPTS=*n*

suppresses the dendrogram when the number of variables (clusters) exceeds the *n* value. This prevents an unreadable plot from being produced. The default is MAXPOINTS=200.

DENDROGRAM < (*dendrogram-options* **) >**

requests a dendrogram and specifies *dendrogram-options*.

Unlike most graphs, the size of the dendrogram can vary as a function of the number of objects that appear in the dendrogram. You can specify the following *dendrogram-options* to control the size and appearance of the dendrogram:

COMPUTEHEIGHT=*a b*

CH=*a b*

specifies the constants for computing the height of the dendrogram. For *n* points being clustered, intercept *a*, and slope *b*, the height is based in part on $a + bn$. For a horizontal dendrogram, the default (given in pixels) is COMPUTEHEIGHT=100 12, the default height

in pixels is max($100 + 12n$, 480), the default height in inches is max($1.04167 + 0.125n$, 5), and the default height in centimeters is max($2.64583 + 0.3175n$, 12.7). For a vertical dendrogram, the default height is 480 pixels. The default unit is pixels, and you can use the UNIT= *dendrogram-option* to change the unit to inches or centimeters for this option. Inches equals pixels divided by 96, and centimeters equals inches times 2.54.

COMPUTEWIDTH=*a b*

CW=*a b*

specifies the constants for computing the width of the dendrogram. For n points being clustered, intercept *a*, and slope *b*, the width is based in part on $a + bn$. For a vertical dendrogram, the default (given in pixels) is COMPUTEWIDTH=100 12, the default width in pixels is max($100 + 12n$, 640), the default width in inches is max($1.04167 + 0.125n$, 6.66667), and the default width in centimeters is max($2.64583 + 0.3175n$, 16.933). For a horizontal dendrogram, the default width is 640 pixels. The default unit is pixels, and you can use the UNIT= *dendrogram-option* to change the unit to inches or centimeters for this option. Inches equals pixels divided by 96, and centimeters equals inches times 2.54.

HEIGHT=PROPORTION | NCL | VAREXP

H=P | N | V

specifies the method for drawing the height of the dendrogram. HEIGHT=PROPORTION is the default.

HEIGHT=PROPORTION specifies that the total proportion of variance explained by the clusters at the current level of the tree is used.

HEIGHT=NCL specifies that the number of clusters is used.

HEIGHT=VAREXP specifies that the total variance explained by the clusters at the current level of the tree is used.

HORIZONTAL | VERTICAL

specifies either a horizontal dendrogram with the objects on the vertical axis (HORIZONTAL) or a vertical dendrogram with the objects on the horizontal axis (VERTICAL). The default is HORIZONTAL.

SETHEIGHT=*height*

SH=*height*

specifies the height of the dendrogram. By default, the height is based on the COMPUTEHEIGHT= option. The default unit is pixels, and you can use the UNIT= *dendrogram-option* to change the unit to inches or centimeters for this *dendrogram-option*.

SETWIDTH=*width*

SW=*width*

specifies the width of the dendrogram. By default, the width is based on the COMPUTEWIDTH= option. The default unit is pixels, and you can use the UNIT= *dendrogram-option* to change the unit to inches or centimeters for this *dendrogram-option*.

UNIT=PX | IN | CM
specifies the unit (pixels, inches, or centimeters) for the SETHEIGHT=, SETWIDTH=, COMPUTEHEIGHT=, and COMPUTEWIDTH= *dendrogram-options*.

NONE
suppresses all plots.

The names of the graphs that PROC VARCLUS generates are listed in Table 129.4, along with the required statements and options.

PROPORTION=*n*

PERCENT=*n*
specifies that when choosing a cluster to split, VARCLUS should choose the cluster with the smallest proportion of variation explained, provided that the proportion of variation explained is less than the PROPORTION= value. Values greater than 1.0 are considered to be percentages, so PROPORTION=0.75 and PERCENT=75 are equivalent.

However, if you specify both MAXEIGEN= and PROPORTION=, VARCLUS first looks for a cluster to split based on the MAXEIGEN= criterion. If no cluster meets that criterion, VARCLUS then looks for a cluster to split based on the PROPORTION= criterion.

If you do not specify PROPORTION=, the default behavior depends on other options as follows:

- If you specify MAXEIGEN=, cluster splitting does not depend on the proportion of variation explained.
- Otherwise, if you specify CENTROID and MAXCLUSTERS=, the default value for PROPORTION= is 1.0.
- Otherwise, if you specify CENTROID without MAXCLUSTERS=, the default value is PROPORTION=0.75 or PERCENT=75.
- Otherwise, cluster splitting does not depend on the proportion of variation explained.

If you specify both PROPORTION= and MAXCLUSTERS=, the number of clusters will never exceed the value of the MAXCLUSTERS= option.

RANDOM=*n*
specifies a positive integer as a starting value for use with REPLACE=RANDOM. If you do not specify the RANDOM= option, the time of day is used to initialize the pseudorandom number sequence.

SHORT
suppresses display of the cluster structure, scoring coefficient, and intercluster correlation matrices.

SIMPLE

S
displays means and standard deviations.

SUMMARY
suppresses all default displayed output except the final summary table.

TRACE
displays the cluster to which each variable is assigned during the iterations.

VARDEF=DF

VARDEF=N

VARDEF=WDF

VARDEF=WEIGHT | WGT
specifies the divisor to be used in the calculation of variances and covariances. The default value is VARDEF=DF. The values and associated divisors are displayed in the following table.

Value	Divisor	Formula
DF	Degrees of freedom	$n - i$
N	Number of observations	n
WDF	Sum of weights minus one	$(\sum_j w_j) - 1$
WEIGHT | WGT	Sum of weights	$\sum_j w_j$

In the preceding table, $i = 0$ if the NOINT option is specified, and $i = 1$ otherwise.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC VARCLUS to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the VARCLUS procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

FREQ Statement

FREQ *variable* **;**

If a variable in your data set represents the frequency of occurrence for the other values in the observation, include the variable's name in a FREQ statement. The procedure then treats the data set as if each observation appears *n* times, where *n* is the value of the FREQ variable for the observation. If the value of the FREQ variable is less than 1, the observation is not used in the analysis. Only the integer portion of the value is used. The total number of observations is considered equal to the sum of the FREQ variable.

PARTIAL Statement

PARTIAL *variables* **;**

If you want to base the clustering on partial correlations, list the variables to be partialed out in the PARTIAL statement.

SEED Statement

SEED *variables* **;**

The SEED statement specifies variables to be used as seeds to initialize the clusters. It is not necessary to use INITIAL=SEED if the SEED statement is present, but if any other INITIAL= option is specified, the SEED statement is ignored.

VAR Statement

VAR *variables* **;**

The VAR statement specifies the variables to be clustered. If you do not specify the VAR statement and do not specify TYPE=SSCP, all numeric variables not listed in other statements (except the SEED statement) are processed. The default VAR variable list does not include the variable INTERCEPT if the DATA= data set is TYPE=SSCP. If the variable INTERCEPT is explicitly specified in the VAR statement with a TYPE=SSCP data set, the NOINT option is enabled.

WEIGHT Statement

WEIGHT *variables* **;**

If you want to specify relative weights for each observation in the input data set, place the weights in a variable in the data set and specify the name in a WEIGHT statement. This is often done when the variance associated with each observation is different and the values of the weight variable are proportional to the reciprocals of the variances. The WEIGHT variable can take nonintegral values. An observation is used in the analysis only if the value of the WEIGHT variable is greater than zero.

Details: VARCLUS Procedure

Missing Values

Observations that contain missing values are omitted from the analysis.

Using the VARCLUS procedure

Default options for PROC VARCLUS often provide satisfactory results. If you want to change the final number of clusters, use one or more of the MAXCLUSTERS=, MAXEIGEN=, or PROPORTION= options. The MAXEIGEN= and PROPORTION= options usually produce similar results but occasionally cause different clusters to be selected for splitting. The MAXEIGEN= option tends to choose clusters with a large number of variables, while the PROPORTION= option is more likely to select a cluster with a small number of variables.

Execution Time

PROC VARCLUS usually requires more computer time than principal factor analysis, but it can be faster than some of the iterative factoring methods. If you have more than 30 variables, you might want to reduce execution time by one or more of the following methods:

- Specify the MINCLUSTERS= and MAXCLUSTERS= options if you know how many clusters you want.
- Specify the HIERARCHY option.
- Specify the SEED statement if you have some prior knowledge of what clusters to expect.

If computer time is not a limiting factor, you might want to try one of the following methods to obtain a better solution:

- If the clustering algorithm has not converged, specify larger values for MAXITER= and MAXSEARCH=.

- Try several factoring and rotation methods with PROC FACTOR to use as input to PROC VARCLUS.
- Run PROC VARCLUS several times, specifying INITIAL=RANDOM.

Output Data Sets

OUTSTAT= Data Set

The OUTSTAT= data set is TYPE=CORR, and it can be used as input to the SCORE procedure or a subsequent run of PROC VARCLUS. The OUSTAT= data set contains the following variables:

- BY variables
- _NCL_, a numeric variable that gives the number of clusters
- _TYPE_, a character variable that indicates the type of statistic the observation contains
- _NAME_, a character variable that contains a variable name or a cluster name, which is of the form CLUS*n*, where *n* is the number of the cluster
- the variables that are clustered

The values of the _TYPE_ variable are listed in the following table.

Table 129.2 _TYPE_

TYPE	**Contents**
'MEAN'	Means
'STD'	Standard deviations
'USTD'	Uncorrected standard deviations, produced when the NOINT option is specified
'N'	Number of observations
'CORR'	Correlations
'UCORR'	Uncorrected correlation matrix, produced when the NOINT option is specified
'MEMBERS'	Number of members in each cluster
'VAREXP'	Variance explained by each cluster
'PROPOR'	Proportion of variance explained by each cluster
'GROUP'	Number of the cluster to which each variable belongs
'RSQUARED'	Squared multiple correlation of each variable with its cluster component
'SCORE'	Standardized scoring coefficients
'USCORE'	Scoring coefficients to be applied without subtracting the mean from the raw variables, produced when the NOINT option is specified
'STRUCTUR'	Cluster structure
'CCORR'	Correlations between cluster components

The observations with _TYPE_='MEAN', 'STD', 'N', and 'CORR' have missing values for the _NCL_ variable. All other values of the _TYPE_ variable are repeated for each cluster solution, with different

solutions distinguished by the value of the _NCL_ variable. If you want to specify the OUTSTAT= data set with the SCORE procedure, you can use a DATA step to select observations with the _NCL_ variable missing or equal to the desired number of clusters as follows:

```
data Coef2;
   set Coef;
   if _ncl_ = . or _ncl_ = 3;
   drop _ncl_;
run;

proc score data=NewScore score=Coef2;
run;
```

PROC SCORE standardizes the new data by subtracting the original variable means that are stored in the _TYPE_='MEAN' observations and dividing by the original variable standard deviations from the _TYPE_='STD' observations. Then PROC SCORE multiplies the standardized variables by the coefficients from the _TYPE_='SCORE' observations to get the cluster scores.

OUTTREE= Data Set

The OUTTREE= data set contains one observation for each variable clustered plus one observation for each cluster of two or more variables—that is, one observation for each node of the cluster tree. The total number of output observations is between n and $2n - 1$, where n is the number of variables clustered.

The OUTTREE= data set contains the following variables:

- BY variables, if any
- _NAME_, a character variable that gives the name of the node. If the node is a cluster, the name is CLUSn, where n is the number of the cluster. If the node is a single variable, the variable name is used.
- _PARENT_, a character variable that gives the value of _NAME_ of the parent of the node. If the node is the root of the tree, _PARENT_ is blank.
- _LABEL_, a character variable that gives the label of the node. If the node is a cluster, the label is CLUSn, where n is the number of the cluster. If the node is a single variable, the variable label is used.
- _NCL_, the number of clusters
- _VAREXP_, the total variance explained by the clusters at the current level of the tree
- _PROPOR_, the total proportion of variance explained by the clusters at the current level of the tree
- _MINPRO_, the minimum proportion of variance explained by a cluster component
- _MAXEIG_, the maximum second eigenvalue of a cluster

Computational Resources

Let

$$
\begin{aligned}
n &= \text{number of observations} \\
v &= \text{number of variables} \\
c &= \text{number of clusters}
\end{aligned}
$$

It is assumed that, at each stage of clustering, the clusters all contain the same number of variables.

Time

The time required for PROC VARCLUS to analyze a given data set varies greatly depending on the number of clusters requested, the number of iterations in both the alternating least squares and search phases, and whether centroid or principal components are used.

The time required to compute the correlation matrix is roughly proportional to nv^2.

Default cluster initialization requires time roughly proportional to v^3. Any other method of initialization requires time roughly proportional to cv^2.

In the alternating least squares phase, each iteration requires time roughly proportional to cv^2 if centroid components are used or

$$\left(c + 5\frac{v}{c^2}\right) v^2$$

if principal components are used.

In the search phase, each iteration requires time roughly proportional to v^3/c if centroid components are used or v^4/c^2 if principal components are used. The HIERARCHY option speeds up each iteration after the first split by as much as $c/2$.

Memory

The amount of memory, in bytes, needed by PROC VARCLUS is approximately

$$v^2 + 2vc + 20v + 15c$$

Interpreting VARCLUS Procedure Output

Because the VARCLUS algorithm is a type of oblique component analysis, its output is similar to the output from the FACTOR procedure for oblique rotations. The scoring coefficients have the same meaning in both PROC VARCLUS and PROC FACTOR; they are coefficients applied to the standardized variables to compute component scores. The cluster structure is analogous to the factor structure that contains the correlations between each variable and each cluster component. A cluster pattern is not displayed because it would be the same as the cluster structure, except that zeros would appear in the same places in which zeros appear in the scoring coefficients. The intercluster correlations are analogous to interfactor correlations; they are the correlations among cluster components.

PROC VARCLUS also displays a cluster summary and a cluster listing. The cluster summary gives the number of variables in each cluster and the variation explained by the cluster component. The latter is similar to the variation explained by a factor but includes contributions from only the variables in that cluster rather than from all variables, as in PROC FACTOR. The proportion of variance explained is obtained by dividing the variance explained by the total variance of variables in the cluster. If the cluster contains two or more variables and the CENTROID option is omitted, the second largest eigenvalue of the cluster is also displayed.

The cluster listing gives the variables in each cluster. Two squared correlations are calculated for each cluster. The column labeled “Own Cluster” gives the squared correlation of the variable with its own cluster component. This value should be higher than the squared correlation with any other cluster unless an iteration limit has been exceeded or the CENTROID option has been used. The larger the squared correlation is, the better. The column labeled “Next Closest” contains the next-highest squared correlation of the variable with a cluster component. This value is low if the clusters are well separated. The column labeled “1–R**2 Ratio” gives the ratio of one minus the “Own Cluster” R square to one minus the “Next Closest” R square. A small “1–R**2 Ratio” indicates a good clustering.

Displayed Output

The following items are displayed for each cluster solution unless the NOPRINT or SUMMARY option is specified. The CLUSTER SUMMARY table includes the following columns:

- the Cluster number
- Members, the number of members in the cluster
- Cluster Variation of the variables in the cluster
- Variation Explained by the cluster component. This statistic is based only on the variables in the cluster rather than on all variables.
- Proportion Explained, the result of dividing the variation explained by the cluster variation
- Second Eigenvalue, the second largest eigenvalue of the cluster. This is displayed if the cluster contains more than one variable and the CENTROID option is not specified

PROC VARCLUS also displays the following:

- Total variation explained, the sum across clusters of the variation explained by each cluster
- Proportion, the total explained variation divided by the total variation of all the variables

The cluster listing includes the following columns:

- Variable, the variables in each cluster
- R square with Own Cluster (the squared correlation of the variable with its own cluster component), and R square with Next Closest (the next highest squared correlation of the variable with a cluster component). Own Cluster values should be higher than the R square with any other cluster unless an iteration limit is exceeded or you specify the CENTROID option. Next Closest should be a low value if the clusters are well separated.
- 1–R**2 Ratio, the ratio of one minus the value in the Own Cluster column to one minus the value in the Next Closest column. The occurrence of low ratios indicates well-separated clusters.

If the SHORT option is not specified, PROC VARCLUS also displays the following tables:

- Standardized Scoring Coefficients, standardized regression coefficients for predicting cluster components from variables
- Cluster Structure, the correlations between each variable and each cluster component
- Inter-Cluster Correlations, the correlations between the cluster components

If the analysis includes partitions for two or more numbers of clusters, a final summary table is displayed. Each row of the table corresponds to one partition. The columns include the following:

- Number of Clusters
- Total Variation Explained by Clusters
- Proportion of Variation Explained by Clusters
- Minimum Proportion (of variation) Explained by a Cluster
- Maximum Second Eigenvalue in a Cluster
- Minimum R square for a Variable
- Maximum 1–R**2 Ratio for a Variable

ODS Table Names

PROC VARCLUS assigns a name to each table it creates. You can use this name to refer to the table when using the Output Delivery System (ODS) to select tables and create output data sets. These ODS table names are listed in Table 129.3. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 129.3 ODS Tables Produced by PROC VARCLUS

ODS Table Name	Description	Option
ClusterQuality	Cluster quality	Default
ClusterStructure	Cluster structure	Default
ClusterSummary	Cluster summary	Default
ConvergenceStatus	Convergence status	Default
Corr	Correlations between variables	CORR
DataOptSummary	Data and options summary table	Default
InterClusterCorr	Correlations between cluster components	Default
IterHistory	Iteration history	TRACE
RSquare	R squares between variables and clusters	Default
SimpleStatistics	Means and standard deviations	SIMPLE
StdScoreCoef	Standardized scoring coefficients	Default

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

By default, PROC VARCLUS produces a dendrogram.

You can refer to every graph produced through ODS Graphics with a name. The name of the graph that PROC VARCLUS generates is listed in Table 129.4, along with the statement and option required to produce it.

Table 129.4 Graphs Produced by PROC VARCLUS

ODS Graph Name	Plot Description	Statement and Option
Dendrogram	Dendrogram (tree diagram)	PROC VARCLUS PLOTS=DENDROGRAM

Example: VARCLUS Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 129.1: Correlations among Physical Variables

(View the complete code for this example (varcluex.sas) in the example repository.)

The data in this example are correlations among eight physical variables as given by Harman (1976). The first PROC VARCLUS run clusters on the basis of principal components. The second run clusters on the basis of centroid components. The third analysis is hierarchical, and the TREE procedure is used to display a tree diagram. The following statements create the data set and perform the analysis:

```
data phys8(type=corr);
   title 'Eight Physical Measurements on 305 School Girls';
   title2 'Harman: Modern Factor Analysis, 3rd Ed, p22';
   label ArmSpan='Arm Span'             Forearm='Length of Forearm'
         LowerLeg='Length of Lower Leg' BitDiam='Bitrochanteric Diameter'
         Girth='Chest Girth'            Width='Chest Width';
   input _Name_ $ 1-8
         (Height ArmSpan Forearm LowerLeg Weight BitDiam
          Girth Width)(7.);
   _Type_='corr';
   datalines;
Height     1.0    .846   .805   .859   .473   .398   .301   .382
ArmSpan    .846   1.0    .881   .826   .376   .326   .277   .415
Forearm    .805   .881   1.0    .801   .380   .319   .237   .345
LowerLeg   .859   .826   .801   1.0    .436   .329   .327   .365
Weight     .473   .376   .380   .436   1.0    .762   .730   .629
BitDiam    .398   .326   .319   .329   .762   1.0    .583   .577
Girth      .301   .277   .237   .327   .730   .583   1.0    .539
Width      .382   .415   .345   .365   .629   .577   .539   1.0
;

proc varclus data=phys8;
run;
```

The PROC VARCLUS statement invokes the procedure. By default, PROC VARCLUS clusters using principal components.

As displayed in Output 129.1.1, when there is only one cluster, the cluster component (by default, the first principal component) explains 58.41% of the total variation of the eight variables.

The cluster is split because the second eigenvalue is greater than 1 (the default value of the MAXEIGEN option).

The two resulting cluster components explain 80.33% of the variation in the original variables. The cluster summary table shows that the variables Height, ArmSpan, Forearm, and LowerLeg have been assigned to the first cluster, and that the variables Weight, BitDiam, Girth, and Width have been assigned to the second cluster.

The standardized scoring coefficients in Output 129.1.1 show that each cluster component has similar scores for each of its associated variables. This suggests that the principal cluster component solution should be similar to the centroid cluster component solution, which follows in the next PROC VARCLUS run.

The cluster structure table displays high correlations between the variables and their own cluster component. The correlations between the variables and the opposite cluster component are all moderate.

The intercluster correlation table shows that the two cluster components have a moderate correlation of 0.44513.

Output 129.1.1 Principal Component Clusters

Eight Physical Measurements on 305 School Girls
Harman: Modern Factor Analysis, 3rd Ed, p22

Oblique Principal Component Cluster Analysis

Observations	10000	**Proportion**	0
Variables	8	**Maxeigen**	1

Clustering algorithm converged.

Cluster Summary for 1 Cluster

Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained	Second Eigenvalue
1	8	8	4.67288	0.5841	1.7710

Total variation explained = 4.67288 Proportion = 0.5841

Cluster 1 will be split because it has the largest second eigenvalue, 1.770983, which is greater than the MAXEIGEN=1 value.

Clustering algorithm converged.

Cluster Summary for 2 Clusters

Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained	Second Eigenvalue
1	4	4	3.509218	0.8773	0.2361
2	4	4	2.917284	0.7293	0.4764

Total variation explained = 6.426502 Proportion = 0.8033

2 Clusters		R-squared with			
Cluster	Variable	Own Cluster	Next Closest	1-R**2 Ratio	Variable Label
Cluster 1	**ArmSpan**	0.9002	0.1658	0.1196	Arm Span
	Forearm	0.8661	0.1413	0.1560	Length of Forearm
	LowerLeg	0.8652	0.1829	0.1650	Length of Lower Leg
	Height	0.8777	0.2088	0.1545	
Cluster 2	**BitDiam**	0.7386	0.1341	0.3019	Bitrochanteric Diameter
	Girth	0.6981	0.0929	0.3328	Chest Girth
	Width	0.6329	0.1619	0.4380	Chest Width
	Weight	0.8477	0.1974	0.1898	

Output 129.1.1 *continued*

Standardized Scoring Coefficients			
Cluster		**1**	**2**
ArmSpan	Arm Span	0.270377	0.000000
Forearm	Length of Forearm	0.265194	0.000000
LowerLeg	Length of Lower Leg	0.265057	0.000000
BitDiam	Bitrochanteric Diameter	0.000000	0.294591
Girth	Chest Girth	0.000000	0.286407
Width	Chest Width	0.000000	0.272710
Height		0.266977	0.000000
Weight		0.000000	0.315597

Cluster Structure			
Cluster		**1**	**2**
ArmSpan	Arm Span	0.948813	0.407210
Forearm	Length of Forearm	0.930624	0.375865
LowerLeg	Length of Lower Leg	0.930142	0.427715
BitDiam	Bitrochanteric Diameter	0.366201	0.859404
Girth	Chest Girth	0.304779	0.835529
Width	Chest Width	0.402430	0.795572
Height		0.936881	0.456908
Weight		0.444281	0.920686

Inter-Cluster Correlations		
Cluster	**1**	**2**
1	1.00000	0.44513
2	0.44513	1.00000

No cluster meets the criterion for splitting.

Number of Clusters	Total Variation Explained by Clusters	Proportion of Variation Explained by Clusters	Minimum Proportion Explained by a Cluster	Maximum Second Eigenvalue in a Cluster	Minimum R-squared for a Variable	Maximum 1-R**2 Ratio for a Variable
1	4.672880	0.5841	0.5841	1.770983	0.3810	
2	6.426502	0.8033	0.7293	0.476418	0.6329	0.4380

In the following statements, the CENTROID option in the PROC VARCLUS statement specifies that cluster centroids be used as the basis for clustering:

```
proc varclus data=phys8 centroid;
run;
```

The first cluster component, which in the centroid method is an unweighted sum of the standardized variables, explains 57.89% of the variation in the data. This value is near the maximum possible variance explained, 58.41%, which is attained by the first principal component shown previously in Output 129.1.1.

The default behavior in the centroid method is to split any cluster with less than 75% of the total cluster variance explained by the centroid component. Since the centroid component for the one-cluster solution explains only 57.89% of the variation as shown in Output 129.1.2, the variables are split into two clusters.

The resulting clusters are the same two clusters created by the principal component method. Recall that this outcome was suggested by the similar standardized scoring coefficients in the principal cluster component solution.

In the two-cluster solution, the centroid component of the second cluster explains only 72.75% of the total variation of the cluster. Since this percentage is less than 75%, the second cluster is split.

In the R-square table for two clusters, the Width variable has a weaker relation to its cluster than any other variable. In the three-cluster solution this variable is in a cluster of its own.

Each cluster component is an unweighted average of the cluster's standardized variables. Thus, the coefficients for each of the cluster's associated variables are identical in the centroid cluster component solution.

The centroid method stops at the three-cluster solution. The three centroid components account for 86.15% of the variability in the eight variables, and all cluster components account for at least 79.44% of the total variation in the corresponding cluster. Additionally, the smallest squared correlation between the variables and their own cluster component is 0.7482.

If the PROPORTION= option were set to a value between 0.5789 (the proportion of variance explained in the one-cluster solution) and 0.7275 (the minimum proportion of variance explained in the two-cluster solution), PROC VARCLUS would stop at the two-cluster solution, and the centroid solution would find the same clusters as the principal components solution, although the cluster components would be slightly different.

Output 129.1.2 Centroid Component Clusters

Eight Physical Measurements on 305 School Girls
Harman: Modern Factor Analysis, 3rd Ed, p22

Oblique Centroid Component Cluster Analysis

Observations	10000	**Proportion**	0.75
Variables	8	**Maxeigen**	0

Clustering algorithm converged.

Cluster Summary for 1 Cluster

Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained
1	8	8	4.631	0.5789

Total variation explained = 4.631 Proportion = 0.5789

Cluster 1 will be split because it has the smallest proportion of variation explained, 0.578875, which is less than the PROPORTION=0.75 value.

Clustering algorithm converged.

Cluster Summary for 2 Clusters

Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained
1	4	4	3.509	0.8773
2	4	4	2.91	0.7275

Total variation explained = 6.419 Proportion = 0.8024

Output 129.1.2 *continued*

2 Clusters		R-squared with			
Cluster	Variable	Own Cluster	Next Closest	1-R**2 Ratio	Variable Label
Cluster 1	**ArmSpan**	0.8994	0.1669	0.1208	Arm Span
	Forearm	0.8663	0.1410	0.1557	Length of Forearm
	LowerLeg	0.8658	0.1824	0.1641	Length of Lower Leg
	Height	0.8778	0.2075	0.1543	
Cluster 2	**BitDiam**	0.7335	0.1341	0.3078	Bitrochanteric Diameter
	Girth	0.6988	0.0929	0.3321	Chest Girth
	Width	0.6473	0.1618	0.4207	Chest Width
	Weight	0.8368	0.1975	0.2033	

Standardized Scoring Coefficients

Cluster		1	2
ArmSpan	Arm Span	0.266918	0.000000
Forearm	Length of Forearm	0.266918	0.000000
LowerLeg	Length of Lower Leg	0.266918	0.000000
BitDiam	Bitrochanteric Diameter	0.000000	0.293105
Girth	Chest Girth	0.000000	0.293105
Width	Chest Width	0.000000	0.293105
Height		0.266918	0.000000
Weight		0.000000	0.293105

Cluster Structure

Cluster		1	2
ArmSpan	Arm Span	0.948361	0.408589
Forearm	Length of Forearm	0.930744	0.375468
LowerLeg	Length of Lower Leg	0.930477	0.427054
BitDiam	Bitrochanteric Diameter	0.366212	0.856453
Girth	Chest Girth	0.304821	0.835936
Width	Chest Width	0.402246	0.804574
Height		0.936883	0.455485
Weight		0.444419	0.914781

Inter-Cluster Correlations

Cluster	1	2
1	1.00000	0.44484
2	0.44484	1.00000

Cluster 2 will be split because it has the smallest proportion of variation explained, 0.7275, which is less than the PROPORTION=0.75 value.

Clustering algorithm converged.

Output 129.1.2 *continued*

Cluster Summary for 3 Clusters				
Cluster	Members	Cluster Variation	Variation Explained	Proportion Explained
1	4	4	3.509	0.8773
2	3	3	2.383333	0.7944
3	1	1	1	1.0000

Total variation explained = 6.892333 Proportion = 0.8615

3 Clusters		R-squared with			
Cluster	Variable	Own Cluster	Next Closest	1-R**2 Ratio	Variable Label
Cluster 1	**ArmSpan**	0.8994	0.1722	0.1215	Arm Span
	Forearm	0.8663	0.1225	0.1524	Length of Forearm
	LowerLeg	0.8658	0.1668	0.1611	Length of Lower Leg
	Height	0.8778	0.1921	0.1513	
Cluster 2	**BitDiam**	0.7691	0.3329	0.3461	Bitrochanteric Diameter
	Girth	0.7482	0.2905	0.3548	Chest Girth
	Weight	0.8685	0.3956	0.2175	
Cluster 3	**Width**	1.0000	0.4259	0.0000	Chest Width

Standardized Scoring Coefficients				
Cluster		1	2	3
ArmSpan	Arm Span	0.26692	0.00000	0.00000
Forearm	Length of Forearm	0.26692	0.00000	0.00000
LowerLeg	Length of Lower Leg	0.26692	0.00000	0.00000
BitDiam	Bitrochanteric Diameter	0.00000	0.37398	0.00000
Girth	Chest Girth	0.00000	0.37398	0.00000
Width	Chest Width	0.00000	0.00000	1.00000
Height		0.26692	0.00000	0.00000
Weight		0.00000	0.37398	0.00000

Cluster Structure				
Cluster		1	2	3
ArmSpan	Arm Span	0.94836	0.36613	0.41500
Forearm	Length of Forearm	0.93074	0.35004	0.34500
LowerLeg	Length of Lower Leg	0.93048	0.40838	0.36500
BitDiam	Bitrochanteric Diameter	0.36621	0.87698	0.57700
Girth	Chest Girth	0.30482	0.86501	0.53900
Width	Chest Width	0.40225	0.65259	1.00000
Height		0.93688	0.43830	0.38200
Weight		0.44442	0.93196	0.62900

Inter-Cluster Correlations			
Cluster	1	2	3
1	1.00000	0.41716	0.40225
2	0.41716	1.00000	0.65259
3	0.40225	0.65259	1.00000

Output 129.1.2 *continued*

No cluster meets the criterion for splitting.

Number of Clusters	Total Variation Explained by Clusters	Proportion of Variation Explained by Clusters	Minimum Proportion Explained by a Cluster	Minimum R-squared for a Variable	Maximum 1-R**2 Ratio for a Variable
1	4.631000	0.5789	0.5789	0.4306	
2	6.419000	0.8024	0.7275	0.6473	0.4207
3	6.892333	0.8615	0.7944	0.7482	0.3548

In the following statements, the MAXC= option computes all clustering solutions, from one to eight clusters, and the SUMMARY option suppresses all output except the final cluster quality table:

```
ods graphics on;

proc varclus data=phys8 maxc=8 summary;
run;
```

The results from PROC VARCLUS are shown in Output 129.1.3.

Output 129.1.3 Hierarchical Clusters and the SUMMARY Option

Eight Physical Measurements on 305 School Girls
Harman: Modern Factor Analysis, 3rd Ed, p22

Oblique Principal Component Cluster Analysis

Observations	10000	**Proportion**	1
Variables	8	**Maxeigen**	0

Clustering algorithm converged.

Number of Clusters	Total Variation Explained by Clusters	Proportion of Variation Explained by Clusters	Minimum Proportion Explained by a Cluster	Maximum Second Eigenvalue in a Cluster	Minimum R-squared for a Variable	Maximum 1-R**2 Ratio for a Variable
1	4.672880	0.5841	0.5841	1.770983	0.3810	
2	6.426502	0.8033	0.7293	0.476418	0.6329	0.4380
3	6.895347	0.8619	0.7954	0.418369	0.7421	0.3634
4	7.271218	0.9089	0.8773	0.238000	0.8652	0.2548
5	7.509218	0.9387	0.8773	0.236135	0.8652	0.1665
6	7.740000	0.9675	0.9295	0.141000	0.9295	0.2560
7	7.881000	0.9851	0.9405	0.119000	0.9405	0.2093
8	8.000000	1.0000	1.0000	0.000000	1.0000	0.0000

The principal component method first separates the variables into the same two clusters that were created in the first PROC VARCLUS run. In creating the third cluster, the principal component method identifies the variable Width. This is the same variable that is put into its own cluster in the preceding centroid method example. The tree diagram in Output 129.1.4 displays the cluster hierarchy.

Output 129.1.4 Dendrogram

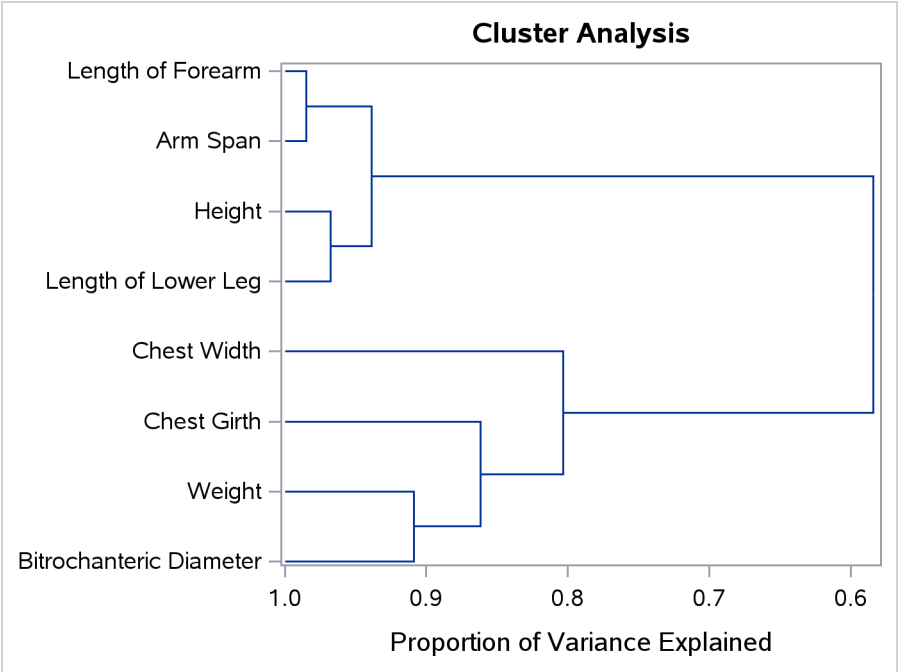

It appears from the diagram that there are two, or possibly three, clusters present. However, the MAXC=8 option forces PROC VARCLUS to split the clusters until each variable is in its own cluster.

References

Anderberg, M. R. (1973). *Cluster Analysis for Applications*. New York: Academic Press.

Harman, H. H. (1976). *Modern Factor Analysis*. 3rd ed. Chicago: University of Chicago Press.

Harris, C. W., and Kaiser, H. F. (1964). "Oblique Factor Analytic Solutions by Orthogonal Transformation." *Psychometrika* 32:363–379.

Chapter 130
The VARCOMP Procedure

Contents

Overview: VARCOMP Procedure **10934**
Getting Started: VARCOMP Procedure **10935**
Analyzing the Cure Rate of Rubber 10935
Syntax: VARCOMP Procedure **10937**
PROC VARCOMP Statement 10938
BY Statement 10939
CLASS Statement 10940
MODEL Statement 10940
Details: VARCOMP Procedure **10941**
Missing Values 10941
Fixed and Random Effects 10941
Negative Variance Component Estimates 10942
Computational Methods 10942
Gauge Repeatability and Reproducibility Analysis 10944
Confidence Limits 10946
Displayed Output 10948
ODS Table Names 10949
Relationship to PROC MIXED 10949
Examples: VARCOMP Procedure **10950**
Example 130.1: Using the Four General Estimation Methods 10950
Example 130.2: Using the GRR Method 10954
References **10957**

Overview: VARCOMP Procedure

The VARCOMP procedure handles general linear models that have random effects. Random effects are classification effects with levels that are assumed to be randomly selected from an infinite population of possible levels. PROC VARCOMP estimates the contribution of each of the random effects to the variance of the dependent variable.

A single MODEL statement specifies the dependent variables and the effects: main effects, interactions, and nested effects. The effects must be composed of classification variables; no continuous variables are allowed on the right side of the equal sign.

You can specify certain effects as fixed (nonrandom) by putting them first in the MODEL statement and indicating the number of fixed effects with the FIXED= option. An intercept is always fitted and assumed fixed. Except for the effects specified as fixed, all other effects are assumed to be random. Their contribution to the model can be thought of as an observation from a distribution that is normally and independently distributed.

The dependent variables are grouped based on the similarity of their missing values. Each group of dependent variables is then analyzed separately. The columns of the design matrix **X** are formed in the same order in which the effects are specified in the MODEL statement. A singular parameterization involving just 0–1 dummy variables is used, as in the GLM procedure.

You can specify four general methods of estimation in the PROC VARCOMP statement by using the METHOD= option. They are TYPE1 (based on computation of Type I sum of squares for each effect), MIVQUE0, maximum likelihood (METHOD=ML), and restricted maximum likelihood (METHOD=REML). A fifth method, METHOD=GRR, provides a specialized analysis for gauge repeatability and reproducibility (R&R) studies. See the section "Gauge Repeatability and Reproducibility Analysis" on page 10944 for further details. Note that this method, along with the CL option in the MODEL statement for confidence limits, applies only to certain designs, namely balanced one-way or two-way designs. The other four general methods apply to any random-effects model and design.

Other procedures, such as PROC GLM, PROC MIXED, and PROC GLIMMIX, fit similar random effects models. The VARCOMP procedure is usually more computationally efficient for certain special designs and models. See the section "Relationship to PROC MIXED" on page 10949 for a more precise comparison with the MIXED procedure in particular.

The GAUGE application in SAS/QC software provides a graphical interface for computing many of the same statistics as METHOD=GRR in PROC VARCOMP.

Getting Started: VARCOMP Procedure

Analyzing the Cure Rate of Rubber

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/varcgs1.sas.)

This example, using data from Hicks (1973), concerns an experiment to determine the sources of variability in cure rates of rubber. The goal of the experiment was to find out if the different laboratories contributed more to the variance of cure rates than did the different batches of raw materials. This information would be useful in trying to control the cure rate of the final product because it would provide insight into the sources of the variability in cure rates. The rubber used was cured at three temperatures, which were taken to be fixed. Three laboratories were chosen at random, and three different batches of raw material were tested at each combination of temperature and laboratory. The following statements read the data into the SAS data set Cure.

```
data Cure;
   input Lab Temp Batch $ Cure @@;
   datalines;
1 145 A 18.6   1 145 A 17.0   1 145 A 18.7   1 145 A 18.7
1 145 B 14.5   1 145 B 15.8   1 145 B 16.5   1 145 B 17.6
1 145 C 21.1   1 145 C 20.8   1 145 C 21.8   1 145 C 21.0
1 155 A  9.5   1 155 A  9.4   1 155 A  9.5   1 155 A 10.0
1 155 B  7.8   1 155 B  8.3   1 155 B  8.9   1 155 B  9.1
1 155 C 11.2   1 155 C 10.0   1 155 C 11.5   1 155 C 11.1
1 165 A  5.4   1 165 A  5.3   1 165 A  5.7   1 165 A  5.3
1 165 B  5.2   1 165 B  4.9   1 165 B  4.3   1 165 B  5.2
1 165 C  6.3   1 165 C  6.4   1 165 C  5.8   1 165 C  5.6
2 145 A 20.0   2 145 A 20.1   2 145 A 19.4   2 145 A 20.0
2 145 B 18.4   2 145 B 18.1   2 145 B 16.5   2 145 B 16.7
2 145 C 22.5   2 145 C 22.7   2 145 C 21.5   2 145 C 21.3
2 155 A 11.4   2 155 A 11.5   2 155 A 11.4   2 155 A 11.5
2 155 B 10.8   2 155 B 11.1   2 155 B  9.5   2 155 B  9.7
2 155 C 13.3   2 155 C 14.0   2 155 C 12.0   2 155 C 11.5
2 165 A  6.8   2 165 A  6.9   2 165 A  6.0   2 165 A  5.7
2 165 B  6.0   2 165 B  6.1   2 165 B  5.0   2 165 B  5.2
2 165 C  7.7   2 165 C  8.0   2 165 C  6.6   2 165 C  6.3
3 145 A 19.7   3 145 A 18.3   3 145 A 16.8   3 145 A 17.1
3 145 B 16.3   3 145 B 16.7   3 145 B 14.4   3 145 B 15.2
3 145 C 22.7   3 145 C 21.9   3 145 C 19.3   3 145 C 19.3
3 155 A  9.3   3 155 A 10.2   3 155 A  9.8   3 155 A  9.5
3 155 B  9.1   3 155 B  9.2   3 155 B  8.0   3 155 B  9.0
3 155 C 11.3   3 155 C 11.0   3 155 C 10.9   3 155 C 11.4
3 165 A  6.7   3 165 A  6.0   3 165 A  5.0   3 165 A  4.8
3 165 B  5.7   3 165 B  5.5   3 165 B  4.6   3 165 B  5.4
3 165 C  6.6   3 165 C  6.5   3 165 C  5.9   3 165 C  5.8
;
```

The variables Lab, Temp, and Batch contain levels of laboratory, temperature, and batch, respectively. The Cure variable contains the response values.

The following SAS statements perform a restricted maximum likelihood variance component analysis.

```
title 'Analyzing the Cure Rate of Rubber';
proc varcomp method=reml data=cure;
   class temp lab batch;
   model cure=temp|lab batch(lab temp) / fixed=1;
run;
```

The FIXED=1 option indicates that the first factor, Temp, is fixed. The effect specification Temp|Lab is equivalent to putting the three terms Temp, Lab, and Temp*Lab in the model. Batch(Lab Temp) is equivalent to putting Batch(Temp*Lab) in the MODEL statement. The results of this analysis are displayed in Figure 130.1 through Figure 130.4.

Figure 130.1 Class Level Information

Analyzing the Cure Rate of Rubber

Variance Components Estimation Procedure

Class Level Information		
Class	**Levels**	**Values**
Temp	3	145 155 165
Lab	3	1 2 3
Batch	3	A B C

Number of Observations Read	108
Number of Observations Used	108

Dependent Variable: Cure

Figure 130.1 provides information about the variables used in the analysis and the number of observations and specifies the dependent variable.

Figure 130.2 Iteration History

REML Iterations					
Iteration	**Objective**	**Var(Lab)**	**Var(Temp*Lab)**	**Var(Batch(Temp*Lab))**	**Var(Error)**
0	13.4500060254	0.5094464340	0	2.4004888633	0.5787185225
1	13.0898262160	0.3194348317	0	2.0869636935	0.6016005334
2	13.0893125570	0.3176048001	0	2.0738906134	0.6026217204
3	13.0893125555	0.3176017115	0	2.0738685461	0.6026234568

Convergence criteria met.

The "REML Iterations" table in Figure 130.2 displays the iteration history, which includes the value of the objective function associated with REML and the values of the variance components at each iteration.

Figure 130.3 REML Estimates

REML Estimates	
Variance Component	**Estimate**
Var(Lab)	0.31760
Var(Temp*Lab)	0
Var(Batch(Temp*Lab))	2.07387
Var(Error)	0.60262

Figure 130.3 displays the REML estimates of the variance components.

Figure 130.4 Covariance Matrix for REML Estimates

Asymptotic Covariance Matrix of Estimates				
	Var(Lab)	**Var(Temp*Lab)**	**Var(Batch(Temp*Lab))**	**Var(Error)**
Var(Lab)	0.32452	0	-0.04998	1.0259E-12
Var(Temp*Lab)	0	0	0	0
Var(Batch(Temp*Lab))	-0.04998	0	0.45042	-0.0022417
Var(Error)	1.0259E-12	0	-0.0022417	0.0089668

The "Asymptotic Covariance Matrix of Estimates" table in Figure 130.4 displays the asymptotic covariance matrix of the REML estimates.

The results of the analysis show that the variance attributable to Batch(Temp*Lab) (with a variance component of 2.0739) is considerably larger than the variance attributable to Lab (0.3176). Therefore, attempts to reduce the variability of cure rates should concentrate on improving the homogeneity of the batches of raw material used rather than standardizing the practices or equipment within the laboratories. Also, note that since the Batch(Temp*Lab) variance is considerably larger than the experimental error (Var(Error)=0.6026), the Batch(Temp*Lab) variability plays an important part in the overall variability of the cure rates.

Syntax: VARCOMP Procedure

The following statements are available in the VARCOMP procedure:

```
PROC VARCOMP <options> ;
    CLASS variables ;
    MODEL dependent = <effects> </options> ;
    BY variables ;
```

Only one MODEL statement is allowed. The BY, CLASS, and MODEL statements are described after the PROC VARCOMP statement.

PROC VARCOMP Statement

PROC VARCOMP < *options* > **;**

The PROC VARCOMP statement invokes the VARCOMP procedure. Table 130.1 summarizes the *options* available in the VARCOMP statement.

Table 130.1 PROC VARCOMP Statement Options

Option	Description
DATA=	Specifies the input SAS data set to use
EPSILON=	Specifies the convergence value of the objective function
MAXITER=	Specifies the maximum number of iterations for METHOD=ML or METHOD=REML
METHOD=	Specifies which of the five methods to use
SEED=	Specifies an unsigned integer used to start the pseudo-random number generator

You can specify the following *options* in the PROC VARCOMP statement.

DATA=*SAS-data-set*
specifies the input SAS data set to use. If this option is omitted, the most recently created SAS data set is used.

EPSILON=*number*
specifies the convergence value of the objective function for METHOD=ML or METHOD=REML. By default, EPSILON=1E–8.

MAXITER=*number*
specifies the maximum number of iterations for METHOD=ML or METHOD=REML. By default, MAXITER=50.

METHOD=TYPE1 | MIVQUE0 | ML | REML | GRR < (*options***) >**
specifies which of the five methods (TYPE1, MIVQUE0, ML, REML, or GRR) you want to use. By default, METHOD=MIVQUE0. METHOD=GRR provides a specialized analysis only for certain designs, whereas the other four methods apply to any random-effects model and design. You can specify the following *options* in parentheses after METHOD=GRR.

SPECLIMITS=(*LSL,USL,*< *k* >**)**

SL=(*LSL,USL,*< *k* >**)**
specifies the specification limits for the first random factor, which is regarded as the product being tested in the gauge R&R study. The lower limit (*LSL*) must be smaller than the upper limit (*USL*). The value *k* is optional. The default value is 6, which corresponds to the number of standard deviations between the "natural" tolerance limits containing the middle 99.73% of a normal process. SPECLIMITS=(*LSL,USL,k*) requests the estimates of the parameters PTR(LSL, USL, k) and Cp(LSL, USL, k) to be displayed.

RATIO

specifies that certain additional ratios of variance components should also be computed and displayed, such as proportion of total variance due to the process. These ratios are listed in Table 130.5.

For more information see the section "Computational Methods" on page 10942.

SEED=*n*

specifies an unsigned integer used to start the pseudo-random number generator. If you do not specify a seed or if you specify zero, the seed is generated from reading the time of day from the computer clock. You can use a SAS date as a seed. The random number generation is used in the computation of generalized confidence limits; see the section "Confidence Limits" on page 10946.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC VARCOMP to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the VARCOMP procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

CLASS Statement

CLASS *variables* **;**

The CLASS statement specifies the classification variables to be used in the analysis. All effects in the MODEL statement must be composed of effects that appear in the CLASS statement. Classification variables can be either numeric or character; if they are character, only the first 16 characters are used.

Numeric classification variables are not restricted to integers since a variable's format determines the levels. For more information, see the discussion of the FORMAT statement in *SAS Formats and Informats: Reference*.

MODEL Statement

MODEL *dependent* **=** < *effects* > < / *options* > **;**

The MODEL statement gives the dependent variables and independent effects. If you specify more than one dependent variable, a separate analysis is performed for each one. The independent effects are limited to main effects, interactions, and nested effects; no continuous effects are allowed. All independent effects must be composed of effects that appear in the CLASS statement. Effects are specified in the VARCOMP procedure in the same way as described for the ANOVA procedure. Only one MODEL statement is allowed.

The following *options* are available in the MODEL statement.

FIXED=*n*

specifies that the first *n* effects in the MODEL statement are fixed effects. The remaining effects are assumed to be random. By default, PROC VARCOMP assumes that all effects are random in the model. Keep in mind that if you use bar notation and, for example, specify Y=A|B / FIXED=2, then A*B is considered a random effect.

CL=MLS | GCL< **(***options***)** >

specifies that confidence limits for all of the parameters of interest be computed and displayed. It also optionally specifies the method to use for computing the confidence limits. There are two methods: the modified large-sample (MLS) method and the generalized confidence limits (GCL) method. The default method is MLS. For more information about these two methods, see the section "Confidence Limits" on page 10946.

You can specify the following *options* in parentheses after CL=GCL.

NSAMPLE=*n*

specifies the sample size for generalized pivot quantities (GPQ) sampling. The default value is 12,605.

EPSILON=*number*

specifies a small positive value used in some GPQ computations. The default value is 0.001.

The CL option applies only to balanced one-way or two-way designs for METHOD=TYPE1 or GRR.

ALPHA=α

specifies the level of significance α for $(1-\alpha)100\%$ two-sided confidence limits. The value of α must be between 0 and 1. By default, α is equal to 0.05.

Details: VARCOMP Procedure

Missing Values

If an observation has a missing value for any variable used in the independent effects, then the analyses of all dependent variables omit this observation. An observation is deleted from the analysis of a given dependent variable if the observation's value for that dependent variable is missing. Note that a missing value in one dependent variable does not eliminate an observation from the analysis of the other dependent variables.

During processing, PROC VARCOMP groups the dependent variables on their missing values across observations so that sums of squares and crossproducts can be computed in the most efficient manner.

Fixed and Random Effects

Central to the idea of variance components models is the idea of fixed and random effects. Each effect in a variance components model must be classified as either a fixed or a random effect. Fixed effects arise when the levels of an effect constitute the entire population in which you are interested. For example, if a plant scientist is comparing the yields of three varieties of soybeans, then Variety would be a fixed effect, providing that the scientist was concerned about making inferences about only these three varieties of soybeans. Similarly, if an industrial experiment focused on the effectiveness of two brands of a machine, Machine would be a fixed effect only if the experimenter's interest did not go beyond the two machine brands.

On the other hand, an effect is classified as a random effect when you want to make inferences about an entire population, and the levels in your experiment represent only a sample from that population. Psychologists comparing test results between different groups of subjects would consider Subject as a random effect. Depending on the psychologists' particular interest, the Group effect might be either fixed or random. For example, if the groups are based on the sex of the subject, then Sex would be a fixed effect. But if the psychologists are interested in the variability in test scores due to different teachers, then they might choose a random sample of teachers as being representative of the total population of teachers, and Teacher would be a random effect. Note that, in the soybean example presented earlier, if the scientists are interested in making inferences about the entire population of soybean varieties and randomly choose three varieties for testing, then Variety would be a random effect.

If all the effects in a model (except for the intercept) are considered random effects, then the model is called a *random-effects model*; likewise, a model with only fixed effects is called a *fixed-effects model*. The more common case, where some factors are fixed and others are random, is called a *mixed model*. In PROC VARCOMP, by default, effects are assumed to be random. You specify which effects are fixed by using the FIXED= option in the MODEL statement. In general, if an interaction or nested effect contains any effect that is random, then the interaction or nested effect should be considered a random effect as well.

In the linear model, each level of a fixed effect contributes a fixed amount to the expected value of the dependent variable. What makes a random effect different is that each level of a random effect contributes an amount that is viewed as a sample from a population of normally distributed variables, each with mean 0, and an unknown variance, much like the usual random error term that is a part of all linear models. The estimate of the variance associated with the random effect is known as the *variance component* because it measures the part of the overall variance contributed by that effect. Thus, PROC VARCOMP estimates the variance of the random variables that are associated with the random effects in your model, and the variance components tell you how much each of the random factors contributes to the overall variability in the dependent variable.

Negative Variance Component Estimates

The variance components estimated by PROC VARCOMP should theoretically be nonnegative because they are assumed to represent the variance of a random variable. Nevertheless, when you are using METHOD=MIVQUE0, TYPE1, or GRR, some estimates of variance components might become negative. (Due to the nature of the algorithms used for METHOD=ML and METHOD=REML, negative estimates are constrained to zero.) These negative estimates might arise for a variety of reasons:

- The variability in your data might be large enough to produce a negative estimate, even though the true value of the variance component is positive.
- Your data might contain outliers. See Hocking (1983) for a graphical technique for detecting outliers in variance components models by using the SAS System.
- A different model for interpreting your data might be appropriate. Under some statistical models for variance components analysis, negative estimates are an indication that observations in your data are negatively correlated. See Hocking (1985) for further information about these models.

Assuming you are satisfied that the model that PROC VARCOMP is using is appropriate for your data, it is common practice to treat negative variance components as if they are zero.

Computational Methods

Four methods of estimation can be specified in the PROC VARCOMP statement by using the METHOD= option. They are described in the following sections.

The Type I Method

This method (METHOD=TYPE1) computes the Type I sum of squares for each effect, equates each mean square involving only random effects to its expected value, and solves the resulting system of equations (Gaylor, Lucas, and Anderson 1970). The $\mathbf{X}'\mathbf{X} \mid \mathbf{X}'\mathbf{Y}$ matrix is computed and adjusted in segments whenever memory is not sufficient to hold the entire matrix.

The MIVQUE0 Method

Based on the technique suggested by Hartley, Rao, and LaMotte (1978), the MIVQUE0 method (METHOD=MIVQUE0) produces unbiased estimates that are invariant with respect to the fixed effects of the model and that are locally best quadratic unbiased estimates given that the true ratio of each component to the residual error component is zero. The technique is similar to TYPE1 except that the random effects are adjusted only for the fixed effects. This affords a considerable timing advantage over the TYPE1 method; thus, MIVQUE0 is the default method used in PROC VARCOMP. The $\mathbf{X}'\mathbf{X}|\mathbf{X}'\mathbf{Y}$ matrix is computed and adjusted in segments whenever memory is not sufficient to hold the entire matrix. Each element (i, j) of the form

$$\mathrm{SSQ}(\mathbf{X}_i'\mathbf{M}\mathbf{X}_j)$$

is computed, where

$$\mathbf{M} = \mathbf{I} - \mathbf{X}_0(\mathbf{X}_0'\mathbf{X}_0)^{-}\mathbf{X}_0'$$

and where $\mathbf{X}_0$ is part of the design matrix for the fixed effects, $\mathbf{X}_i$ is part of the design matrix for one of the random effects, and SSQ is an operator that takes the sum of squares of the elements. For more information see Rao (1971, 1972) and Goodnight (1978).

The Maximum Likelihood Method

The maximum likelihood method (METHOD=ML) computes maximum likelihood estimates of the variance components; see Searle, Casella, and McCulloch (1992). The computing algorithm makes use of the W-transformation developed by Hemmerle and Hartley (1973) and Goodnight and Hemmerle (1979). The procedure uses a Newton-Raphson algorithm, iterating until the log-likelihood objective function converges.

The objective function for METHOD=ML is $\ln(|\mathbf{V}|) + \mathbf{r}'\mathbf{V}^{-1}\mathbf{r}$, where

$$\mathbf{V} = \sigma_0^2\mathbf{I} + \sum_{i=1}^{n_r} \sigma_i^2\mathbf{X}_i\mathbf{X}_i'$$

and where σ_0^2 is the residual variance, n_r is the number of random effects in the model, σ_i^2 represents the variance components, $\mathbf{X}_i$ is part of the design matrix for one of the random effects, and

$$\mathbf{r} = \mathbf{y} - \mathbf{X}_0(\mathbf{X}_0'\mathbf{V}^{-1}\mathbf{X}_0)^{-}\mathbf{X}_0'\mathbf{V}^{-1}\mathbf{y}$$

is the vector of residuals.

The Restricted Maximum Likelihood Method

The restricted maximum likelihood method (METHOD=REML) is similar to the maximum likelihood method, but it first separates the likelihood into two parts: one that contains the fixed effects and one that does not (Patterson and Thompson 1971). The procedure uses a Newton-Raphson algorithm, iterating until convergence is reached for the log-likelihood objective function of the portion of the likelihood that does not contain the fixed effects. Using notation from earlier methods, the objective function for METHOD=REML is $\ln(|\mathbf{V}|) + \mathbf{r}'\mathbf{V}^{-1}\mathbf{r} + \ln(|\mathbf{X}_0'\mathbf{V}^{-1}\mathbf{X}_0|)$. See Searle, Casella, and McCulloch (1992) for additional details.

The GRR Method

Based on the technique suggested by Burdick, Borror, and Montgomery (2005), the GRR method (METHOD=GRR) produces minimum variance unbiased estimators.

Gauge Repeatability and Reproducibility Analysis

In a typical gauge R&R experiment, each operator (O_j) makes multiple observations on each of several similar parts (P_i) from a monitored process. The statistical model used to describe the response variable is the balanced two-factor crossed random model with interaction

$$y_{ijk} = \mu_y + P_i + O_j + (PO)_{ij} + E_{ijk}$$

where $i = 1, \ldots, p,\ j = 1, \ldots, o,\ k = 1, \ldots, r,\ \mu_y$ is an unknown constant, and $P_i, O_j, (PO)_{ij}, E_{ijk}$ are jointly independent normal random variables with means of zero and variances $\mathrm{Var}(P), \mathrm{Var}(O), \mathrm{Var}(PO)$, and $\mathrm{Var}(E)$, respectively. The corresponding SAS statements are as follows:

```
proc varcomp method=grr;
   class P O;
   model y = P|O;
run;
```

The first random effect in the MODEL statement is assumed to be the "Part" effect and the second is "Operator."

The ANOVA table for the preceding model is shown in Table 130.2.

Table 130.2 GRR Analysis of Variance

Source	DF	Mean Square	Expected Mean Square
Parts(P)	$p-1$	S_P^2	$\mathrm{Var}(E) + r\mathrm{Var}(PO) + or\mathrm{Var}(P)$
Operators(O)	$o-1$	S_O^2	$\mathrm{Var}(E) + r\mathrm{Var}(PO) + pr\mathrm{Var}(O)$
P×O	$(p-1)(o-1)$	S_{PO}^2	$\mathrm{Var}(E) + r\mathrm{Var}(PO)$
Error(E)	$po(r-1)$	S_E^2	$\mathrm{Var}(E)$

The gauge R&R parameters of interest are given in Table 130.3 in terms of $\mathrm{Var}(P), \mathrm{Var}(O), \mathrm{Var}(PO)$, and $\mathrm{Var}(E)$.

Table 130.3 Gauge R&R Parameters

Parameter	Formula
Mean of population of measurements	$\mu_y = \bar{y}_{\ldots} = \Sigma_{ijk} y_{ijk}/por$
Variance of the monitored process	$\gamma_P = \mathrm{Var}(P)$
Variance of the measurement system	$\gamma_M = \mathrm{Var}(O) + \mathrm{Var}(PO) + \mathrm{Var}(E)$
Total variance of the response variable	$\gamma_y = \mathrm{Var}(y) = \gamma_P + \gamma_M$
Ratio of process variance to measurement variance	$\gamma_R = \gamma_P/\gamma_M$

Table 130.3 (continued)

Parameter	Formula
Proportion of total variance due to the process	$\rho_P = \gamma_P/\gamma_y = \frac{\gamma_R}{1+\gamma_R}$
Proportion of total variance due to the measurement	$\rho_M = \gamma_M/\gamma_y = 1 - \rho_P$
Signal-to-noise ratio	$\mathrm{SNR} = \sqrt{2 \times \gamma_R}$
Discrimination ratio	$\mathrm{DR} = 1 + 2\gamma_R$

For a one-way model, $\gamma_M = \mathrm{Var}(E)$, and for a two-way model with no interaction, $\gamma_M = \mathrm{Var}(O)+\mathrm{Var}(E)$.

If you use the SPECLIMITS option to give specification limits, the two parameters in Table 130.4 will also be estimated and displayed.

Table 130.4 Gauge R&R Parameters Related to Specification Limits

Parameter	Formula
Precision-to-tolerance ratio	$\mathrm{PTR}(\mathrm{LSL}, \mathrm{USL}, \mathrm{k}) = k\sqrt{\gamma_M}/(\mathrm{USL} - \mathrm{LSL})$
Process capability ratio	$\mathrm{Cp}(\mathrm{LSL}, \mathrm{USL}, \mathrm{k}) = (\mathrm{USL} - \mathrm{LSL})/(k\sqrt{\gamma_P})$

Here, USL and LSL are the specification limits, and the value k corresponds to the number of standard deviations between the "natural" tolerance limits of a normal process.

If you use the RATIO option, the ratios in Table 130.5 will also be estimated and displayed.

Table 130.5 Gauge R&R Ratios

Ratio	Formula
Ratio of process variance to total variance	$\mathrm{Var}(P)/\gamma_y$
Ratio of operator variance to total variance	$\mathrm{Var}(O)/\gamma_y$
Ratio of process by operator variance to total variance	$\mathrm{Var}(PO)/\gamma_y$
Ratio of process variance to residual variance	$\mathrm{Var}(P)/\mathrm{Var}(E)$
Ratio of operator variance to residual variance	$\mathrm{Var}(O)/\mathrm{Var}(E)$
Ratio of process by operator variance to residual variance	$\mathrm{Var}(PO)/\mathrm{Var}(E)$

Confidence Limits

When no exact confidence limits exist, it is common practice to use approximate confidence limits. Two such approximations are the modified large-sample (MLS) method and the generalized confidence limit (GCL) method as discussed in Burdick, Borror, and Montgomery (2005). When analyzing a balanced one-way or two-way design, if you specify the CL= option with METHOD=TYPE1 or GRR, the VARCOMP procedure computes confidence limits by using either the MLS method (the default) or the GCL method. Generalized confidence limits are obtained by specifying the CL=GCL option in the MODEL statement.

MLS Confidence Limits

The method of MLS confidence limits was first introduced by Graybill and Wang (1980). It starts with approximate large-sample confidence limits; then it modifies the limits to be exact under certain parameter conditions.

For a balanced two-way crossed random model with interaction, formulas for the MLS method are given in Table 130.6. See Burdick, Borror, and Montgomery (2005) for the formulas for one-way or balanced two-way with no interaction models.

Confidence limits for parameters such as variances and their ratios might not contain the corresponding point estimates, because negative confidence bounds are increased to zero.

Table 130.6 100(1 − α)% MLS Confidence Limits

Parameter	Lower Bound	Upper Bound
μ_y	$\bar{y}_{...} - C\sqrt{\frac{K}{por}}$	$\bar{y}_{...} + C\sqrt{\frac{K}{por}}$
γ_P	$\hat{\gamma}_P - \sqrt{V_{LP}}/(or)$	$\hat{\gamma}_P + \sqrt{V_{UP}}/(or)$
γ_M	$\hat{\gamma}_M - \sqrt{V_{LM}}/(pr)$	$\hat{\gamma}_M + \sqrt{V_{UM}}/(pr)$
γ_y	$\hat{\gamma}_y - \sqrt{V_{LT}}/(por)$	$\hat{\gamma}_y + \sqrt{V_{UT}}/(por)$
γ_R	L_R	U_R
ρ_P	$L_R/(1 + L_R)$	$U_R/(1 + U_R)$
ρ_M	$1/(1 + U_R)$	$1/(1 + L_R)$

The terms in Table 130.6 are defined as follows:

$$
\begin{aligned}
V_{LP} &= G_1^2 S_P^4 + H_3^2 S_{PO}^4 + G_{13} S_P^2 S_{PO}^2 \\
V_{UP} &= H_1^2 S_P^4 + G_3^2 S_{PO}^4 + H_{13} S_P^2 S_{PO}^2 \\
V_{LM} &= G_2^2 S_O^4 + G_3^2 (p-1)^2 S_{PO}^4 + G_4^2 p^2 (r-1)^2 S_E^4 \\
V_{UM} &= H_2^2 S_O^4 + H_3^2 (p-1)^2 S_{PO}^4 + H_4^2 p^2 (r-1)^2 S_E^4 \\
V_{LT} &= G_1^2 p^2 S_P^4 + G_2^2 o^2 S_O^4 + G_3^2 (po-p-o)^2 S_{PO}^4 + G_4^2 (po)^2 (r-1)^2 S_E^4 \\
V_{UT} &= H_1^2 p^2 S_P^4 + H_2^2 o^2 S_O^4 + H_3^2 (po-p-o)^2 S_{PO}^4 + H_4^2 (po)^2 (r-1)^2 S_E^4 \\
L_R &= \frac{p(1-G_1)(S_P^2 - F_1 S_{PO}^2)}{po(r-1)S_E^2 + o(1-G_1)F_3 S_O^2 + o(p-1)S_{PO}^2} \\
U_R &= \frac{p(1+H_1)(S_P^2 - F_2 S_{PO}^2)}{po(r-1)S_E^2 + o(1+H_1)F_4 S_O^2 + o(p-1)S_{PO}^2} \\
G_1 &= 1 - F_{\alpha/2:\infty,p-1} \\
G_2 &= 1 - F_{\alpha/2:\infty,o-1} \\
G_3 &= 1 - F_{\alpha/2:\infty,(p-1)(o-1)} \\
G_4 &= 1 - F_{\alpha/2:\infty,po(r-1)} \\
H_1 &= F_{1-\alpha/2:\infty,p-1} - 1 \\
H_2 &= F_{1-\alpha/2:\infty,o-1} - 1 \\
H_3 &= F_{1-\alpha/2:\infty,(p-1)(o-1)} - 1 \\
H_4 &= F_{1-\alpha/2:\infty,po(r-1)} - 1 \\
F_1 &= F_{1-\alpha/2:p-1,(p-1)(o-1)} \\
F_2 &= F_{\alpha/2:p-1,(p-1)(o-1)} \\
F_3 &= F_{1-\alpha/2:p-1,o-1} \\
F_4 &= F_{\alpha/2:p-1,o-1} \\
G_{13} &= \frac{(F_1-1)^2 - G_1^2 F_1^2 - H_3^2}{F_1} \\
H_{13} &= \frac{(1-F_2)^2 - H_1^2 F_2^2 - G_3^2}{F_2} \\
K &= s_P^2 + s_O^2 - s_{PO}^2 \\
C &= \frac{s_P^2 \sqrt{F_{1-\alpha:1,p-1}} + s_O^2 \sqrt{F_{1-\alpha:1,o-1}} - s_{PO}^2 \sqrt{F_{1-\alpha:1,(p-1)(o-1)}}}{K}
\end{aligned}
$$

The symbol $F_{\alpha:df1,df2}$ represents the percentile of an F distribution with *df1* and *df2* degrees of freedom and area α to the left.

Generalized Confidence Limits

The method of generalized confidence limits was first introduced by Weerahandi (1993). The 100(1-α)% generalized confidence limits are determined as follows:

1. Initialize the random number generator with the seed. The seed value is specified by the SEED= option.

2. Sample N generalized pivot quantities (GPQ), defined to have a distribution that is independent of the parameters under study. The value N is specified by the NSAMPLE= option.

3. Define the lower and upper limits as the $\alpha/2$ and $1-\alpha/2$ quantiles of the sampled GPQ values.

Formulas for generalized confidence limits are given in Table 130.7, where Z denotes a standard normal random variable and W_1, W_2, W_3, and W_4 denote jointly independent chi-square random variables that are independent of Z with degrees of freedom $p-1, o-1, (p-1)(o-1)$ and $po(r-1)$, respectively. The value of ϵ in Table 130.7 is specified by the EPSILON= option.

Table 130.7 100(1 − α)% Generalized Confidence Limits

Parameter	GPQ
μ_y	$\bar{y}_{...} - Z\sqrt{\max\left[\epsilon, \frac{(p-1)s_P^2}{porW_1} + \frac{(o-1)s_O^2}{porW_2} - \frac{(p-1)(o-1)s_{PO}^2}{porW_3}\right]}$
γ_P	$\max\left[0, \frac{(p-1)s_P^2}{orW_1} - \frac{(p-1)(o-1)s_{po}^2}{prW_3}\right]$
γ_M	$\frac{(o-1)s_o^2}{prW_2} + \frac{(p-1)^2(0-1)s_p^2 o}{prW_3} + \frac{po(r-1)^2 s_E^2}{rW_4}$
γ_y	$\frac{(p-1)s_P^2}{orW_1} + \frac{(o-1)s_O^2}{prW_2} + \frac{(po-p-o)(p-1)(o-1)s_{PO}^2}{porW_3} + \frac{po(r-1)^2 s_E^2}{rW_4}$
γ_R	$\frac{\mathrm{GPQ}(\gamma_P)}{\mathrm{GPQ}(\gamma_M)}$

In general, the GCL method provides a more accurate confidence interval with a shorter interval width than the MLS method. However, the greater accuracy comes at the cost of being somewhat nondeterministic, because of the reliance on simulation.

Displayed Output

PROC VARCOMP displays the following items:

- Class Level Information for verifying the levels in your data
- Number of observations read from the data set and number of observations used in the analysis
- for METHOD=TYPE1, an analysis-of-variance table with Source, DF, Type I Sum of Squares, Type I Mean Square, and Expected Mean Square, and a table of Type I variance component estimates
- for METHOD=MIVQUE0, the SSQ Matrix containing sums of squares of partitions of the $\mathbf{X}'\mathbf{X}$ crossproducts matrix adjusted for the fixed effects
- for METHOD=ML and METHOD=REML, the iteration history, including the objective function, a table of variance component estimates, and the estimated Asymptotic Covariance Matrix of the variance components
- for METHOD=GRR, an analysis-of-variance table with Source, DF, GRR Sum of Squares, GRR Mean Square, and Expected Mean Square, and a table of GRR parameter estimates. If the CL option is specified, confidence limits for each parameter estimate will also be displayed.

ODS Table Names

PROC VARCOMP assigns a name to each table it creates. You can use these names to reference the table when using the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 130.8. For more information about ODS, see Chapter 23, "Using the Output Delivery System."

Table 130.8 ODS Tables Produced by PROC VARCOMP

ODS Table Name	Description	Statement
ANOVA	Type 1 analysis of variance	METHOD=TYPE1 or GRR
AsyCov	Asymptotic covariance matrix of estimates	METHOD=ML or REML
ClassLevels	Class level information	default
ConvergenceStatus	Convergence status	METHOD=ML or REML
DepVar	Dependent variable	METHOD=TYPE1, REML, ML, or GRR
DependentInfo	Dependent variable info (multiple variables)	
Estimates	Variance component estimates	default
IterHistory	Iteration history	METHOD=ML or REML
NObs	Number of observations	default
SSCP	Sum of squares matrix	METHOD=MIVQUE0

In situations where multiple dependent variables are analyzed that differ in their missing value pattern, separate names for ANOVA*n*, AsyCov*n*, Estimates*n*, IterHistory*n*, and SSCP*n* tables are no longer required. The results are combined into a single output data set. For METHOD=TYPE1, ML, or REML, the variable Dependent in the output data set identifies the dependent variable. For METHOD=MIVQUE0, a variable is added to the output data set for each dependent variable.

Relationship to PROC MIXED

The MIXED procedure effectively performs the same analyzes as PROC VARCOMP and many others, including Type I, Type II, and Type III tests of fixed effects, confidence limits, customized contrasts, and least squares means. Furthermore, continuous variables are permitted as both fixed and random effects in PROC MIXED, and numerous other covariance structures besides variance components are available. The VARCOMP procedure is more computationally efficient for some special designs and models.

To translate PROC VARCOMP code into PROC MIXED code, move all random effects to the RANDOM statement in PROC MIXED. For example, the syntax for the example in the section "Getting Started: VARCOMP Procedure" on page 10935 is as follows:

```
proc mixed;
   class Temp Lab Batch;
   model Cure = Temp;
   random Lab Temp*Lab Batch(Lab Temp);
run;
```

REML is the default estimation method in PROC MIXED, and you can specify other methods by using the METHOD= option.

Examples: VARCOMP Procedure

The examples in this chapter are available in the GitHub repository located at https://github.com/sascommunities/doc-supplement-statug.

Example 130.1: Using the Four General Estimation Methods

(View the complete code for this example (varcex1.sas) in the example repository.)

In this example, a and b are classification variables and y is the dependent variable. a is declared fixed, and b and a*b are random. Note that this design is unbalanced because the cell sizes are not all the same. PROC VARCOMP is invoked four times, once for each of the general estimation methods. The data are from Hemmerle and Hartley (1973). The following statements produce Output 130.1.1.

```
data a;
   input a b y @@;
   datalines;
1 1 237   1 1 254   1 1 246   1 2 178   1 2 179
2 1 208   2 1 178   2 1 187   2 2 146   2 2 145   2 2 141
3 1 186   3 1 183   3 2 142   3 2 125   3 2 136
;

proc varcomp method=type1 data=a;
   class a b;
   model y=a|b / fixed=1;
run;
```

Output 130.1.1 VARCOMP Procedure: Method=TYPE1

Variance Components Estimation Procedure

Class Level Information		
Class	**Levels**	**Values**
a	3	1 2 3
b	2	1 2

Number of Observations Read	16
Number of Observations Used	16

Dependent Variable:	y

Output 130.1.1 *continued*

Type 1 Analysis of Variance				
Source	**DF**	**Sum of Squares**	**Mean Square**	**Expected Mean Square**
a	2	11736	5868.218750	Var(Error) + 2.725 Var(a*b) + 0.1 Var(b) + Q(a)
b	1	11448	11448	Var(Error) + 2.6308 Var(a*b) + 7.8 Var(b)
a*b	2	299.041026	149.520513	Var(Error) + 2.5846 Var(a*b)
Error	10	786.333333	78.633333	Var(Error)
Corrected Total	15	24270		

Type 1 Estimates	
Variance Component	**Estimate**
Var(b)	1448.4
Var(a*b)	27.42659
Var(Error)	78.63333

The "Class Level Information" table in Output 130.1.1 displays the levels of each variable specified in the CLASS statement. You can check this table to make sure the data are input correctly.

The Type I analysis of variance in Output 130.1.1 consists of a sequential partition of the total sum of squares. The mean square is the sum of squares divided by the degrees of freedom, and the expected mean square is the expected value of the mean square under the mixed model. The "Q" notation in the expected mean squares refers to a quadratic form in parameters of the parenthesized effect.

The Type I estimates of the variance components in Output 130.1.1 result from solving the linear system of equations established by equating the observed mean squares to their expected values.

The following statements are the same as before, except that the estimation method is MIVQUE0 instead of the default TYPE1. They produce Output 130.1.2.

```
proc varcomp method=mivque0 data=a;
   class a b;
   model y=a|b / fixed=1;
run;
```

Output 130.1.2 VARCOMP Procedure: Method=MIVQUE0

Variance Components Estimation Procedure

MIVQUE(0) SSQ Matrix				
Source	**b**	**a*b**	**Error**	**y**
b	60.84000	20.52000	7.80000	89295.4
a*b	20.52000	20.52000	7.80000	30181.3
Error	7.80000	7.80000	13.00000	12533.5

MIVQUE(0) Estimates	
Variance Component	**y**
Var(b)	1466.1
Var(a*b)	-35.49170
Var(Error)	105.73660

The MIVQUE0 estimates in Output 130.1.2 result from solving the equations established by the MIVQUE0 SSQ matrix. Note that the estimate of the variance component for the interaction effect, Var(a*b), is negative for this example.

The following statements use METHOD=ML to invoke maximum likelihood estimation. They produce Output 130.1.3.

```
proc varcomp method=ml data=a;
   class a b;
   model y=a|b / fixed=1;
run;
```

Output 130.1.3 VARCOMP Procedure: Method=ML

Variance Components Estimation Procedure

Maximum Likelihood Iterations

Iteration	Objective	Var(b)	Var(a*b)	Var(Error)
0	78.3850371200	1031.49070	0	74.3909717935
1	78.2637043807	732.3606453635	0	77.4011688154
2	78.2635471161	723.6867470850	0	77.5301774839
3	78.2635471152	723.6658365289	0	77.5304926877

Convergence criteria met.

Maximum Likelihood Estimates

Variance Component	Estimate
Var(b)	723.66584
Var(a*b)	0
Var(Error)	77.53049

Asymptotic Covariance Matrix of Estimates

	Var(b)	Var(a*b)	Var(Error)
Var(b)	537826.1	0	-107.33905
Var(a*b)	0	0	0
Var(Error)	-107.33905	0	858.71104

The "Maximum Likelihood Iterations" table in Output 130.1.3 shows that the Newton-Raphson algorithm used by PROC VARCOMP requires three iterations to converge.

The ML estimate of Var(a*b) is zero for this example, and the other two estimates are smaller than their Type I and MIVQUE0 counterparts.

One benefit of using likelihood-based methods is that an approximate covariance matrix is available from the matrix of second derivatives evaluated at the ML solution. This covariance matrix is valid asymptotically and can be unreliable in small samples.

Here the variance component estimates for B and the Error are negatively correlated, and the elements for Var(a*b) are set to zero because the estimate equals zero. Also, the very large variance for Var(b) indicates a lot of uncertainty about the estimate for Var(b), and one contributing explanation is that B has only two levels in this data set.

Finally, the following statements use the restricted maximum likelihood (REML) for estimation. They produce Output 130.1.4.

```
proc varcomp method=reml data=a;
   class a b;
   model y=a|b / fixed=1;
run;
```

Output 130.1.4 VARCOMP Procedure: Method=REML

Variance Components Estimation Procedure

REML Iterations

Iteration	Objective	Var(b)	Var(a*b)	Var(Error)
0	63.4134144942	1269.52701	0	91.5581191305
1	63.0446869787	1601.84199	32.7632417174	76.9355562461
2	63.0311530508	1468.82932	27.2258186561	78.7548276319
3	63.0311265148	1464.33646	26.9564053003	78.8431476502
4	63.0311265127	1464.36727	26.9588525177	78.8423898761

Convergence criteria met.

REML Estimates

Variance Component	Estimate
Var(b)	1464.4
Var(a*b)	26.95885
Var(Error)	78.84239

Asymptotic Covariance Matrix of Estimates

	Var(b)	Var(a*b)	Var(Error)
Var(b)	4401703.8	1.29359	-273.39651
Var(a*b)	1.29359	3559.1	-502.85157
Var(Error)	-273.39651	-502.85157	1249.7

The "REML Iterations" table in Output 130.1.4 shows that the REML optimization requires four iterations to converge.

The REML estimates in Output 130.1.4 are all larger than the corresponding ML estimates (adjusting for potential downward bias) and are fairly similar to the Type I estimates.

The "Asymptotic Covariance Matrix of Estimates" table in Output 130.1.4 shows that the Error variance component estimate is negatively correlated with the other two variance component estimates, and the estimated variances are all larger than their ML counterparts.

Example 130.2: Using the GRR Method

(View the complete code for this example (varcex2.sas) in the example repository.)

In this example from Houf and Burman (1988), the response variable is the thermal performance of a module measured in Celsius degrees per watt. Each of three operators measures 10 parts three times. It is assumed that parts and operators are selected at random from larger populations. The following statements produce Output 130.2.1.

```
data Houf;
   input a b y @@;
   datalines;
1  1 37    1  1 38    1 1 37
1  2 41    1  2 41    1 2 40
1  3 41    1  3 42    1 3 41
2  1 42    2  1 41    2 1 43
2  2 42    2  2 42    2 2 42
2  3 43    2  3 42    2 3 43
3  1 30    3  1 31    3 1 31
3  2 31    3  2 31    3 2 31
3  3 29    3  3 30    3 3 28
4  1 42    4  1 43    4 1 42
4  2 43    4  2 43    4 2 43
4  3 42    4  3 42    4 3 42
5  1 28    5  1 30    5 1 29
5  2 29    5  2 30    5 2 29
5  3 31    5  3 29    5 3 29
6  1 42    6  1 42    6 1 43
6  2 45    6  2 45    6 2 45
6  3 44    6  3 46    6 3 45
7  1 25    7  1 26    7 1 27
7  2 28    7  2 28    7 2 30
7  3 29    7  3 27    7 3 27
8  1 40    8  1 40    8 1 40
8  2 43    8  2 42    8 2 42
8  3 43    8  3 43    8 3 41
9  1 25    9  1 25    9 1 25
9  2 27    9  2 29    9 2 28
9  3 26    9  3 26    9 3 26
10 1 35   10  1 34   10 1 34
10 2 35   10  2 35   10 2 34
10 3 35   10  3 34   10 3 35
;

proc varcomp data=Houf method=grr (speclimits=(18,58) ratio);
   class a b;
   model y=a|b/cl;
run;
```

You specify METHOD=GRR in this example to drive the VARCOMP procedure to produce a gauge repeatability and reproducibility analysis. With the option speclimits=(18 58), the parameters PTR and Cp are estimated and displayed. With the RATIO option, certain additional ratios of variance components are also estimated and displayed. Finally, the CL= option in the MODEL statement specifies that estimates of

GRR quantities should have the corresponding confidence limits.

Output 130.2.1 Class Level Information Using Method=GRR

Variance Components Estimation Procedure

Class Level Information		
Class	**Levels**	**Values**
a	10	1 2 3 4 5 6 7 8 9 10
b	3	1 2 3

Number of Observations Read	90
Number of Observations Used	90

Dependent Variable:	y

The "Class Level Information" table in Output 130.2.1 displays the levels of each variable specified in the CLASS statement.

Output 130.2.2 Analysis of Variance Using Method=GRR

GRR Analysis of Variance				
Source	**DF**	**Sum of Squares**	**Mean Square**	**Expected Mean Square**
a	9	3935.955556	437.328395	Var(Error) + 3 Var(a*b) + 9 Var(a)
b	2	39.266667	19.633333	Var(Error) + 3 Var(a*b) + 30 Var(b)
a*b	18	48.511111	2.695062	Var(Error) + 3 Var(a*b)
Error	60	30.666667	0.511111	Var(Error)
Corrected Total	89	4054.400000		

The GRR analysis of variance in Output 130.2.2 is the same as for the Type I analysis when the design is balanced.

Finally, the estimates of the GRR parameters of interest and their confidence limits are displayed in Output 130.2.3.

Output 130.2.3 Parameter Estimates Using Method=GRR

GRR Estimates			
		95%	
Parameter	Estimate	Confidence Limits	
Mu Y	35.80000	30.49477	41.10523
Var(a)	48.29259	22.69452	161.63918
Var(b)	0.56461	0.07296	25.75077
Var(a*b)	0.72798	0.33273	1.79272
Var(Error)	0.51111	0.36816	0.75754
Gamma Y	50.09630	24.48844	166.22217
Gamma P	48.29259	22.69452	161.63918
Gamma M	1.80370	1.20623	27.01724
Gamma R	26.77413	1.69168	105.60895
SNR	7.31767	1.83939	14.53334
PTR(18,58,6)	0.20145	0.16474	0.77967
Cp(18,58,6)	0.95933	0.52437	1.39942
DR	54.54825	4.38336	212.21791
Rho P	0.96400	0.62848	0.99062
Rho M	0.03600	0.0093801	0.37152
Var(a)/Gamma Y	0.96400	0.62848	0.99062
Var(b)/Gamma Y	0.01127	0.0008700	0.34151
Var(a*b)/Gamma Y	0.01453	0.0027083	0.04744
Var(a)/Var(Error)	94.48551	40.19199	327.32469
Var(b)/Var(Error)	1.10467	0.13662	50.37744
Var(a*b)/Var(Error)	1.42432	0.55232	3.74691

You can draw the following inferences from the results of the analysis. Most of the variation is due to differences between parts because of the relative larger value of Gamma R. The measurement system is nearly inadequate because the PTR exceeds 20%. However, the measurement system is of value in monitoring the process since the SNR is greater than five. See Burdick, Borror, and Montgomery (2003) for more information about interpreting gauge R&R studies.

The confidence limits in Output 130.2.3 are based on large-sample asymptotic approximation. You can alternatively compute more accurate and usually smaller confidence intervals by using CL=GCL for generalized confidence limits. The following statements produce Output 130.2.4:

```
proc varcomp data=Houf method=grr (speclimits=(18,58) ratio) seed=104;
   class a b;
   model y=a|b/cl=gcl;
run;
```

Output 130.2.4 Generalized Confidence Limits

Variance Components Estimation Procedure

GRR Estimates			
Parameter	Estimate	95% Generalized Confidence Limits	
Mu Y	35.80000	30.23607	41.39887
Var(a)	48.29259	22.46186	160.90542
Var(b)	0.56461	0.06936	29.93347
Var(a*b)	0.72798	0.33182	1.79558
Var(Error)	0.51111	0.36816	0.75754
Gamma Y	50.09630	25.20454	175.29591
Gamma P	48.29259	22.46186	160.90542
Gamma M	1.80370	1.18137	31.03229
Gamma R	26.77413	1.66841	84.92030
SNR	7.31767	1.82670	13.03229
PTR(18,58,6)	0.20145	0.16304	0.83560
Cp(18,58,6)	0.95933	0.52556	1.40665
DR	54.54825	4.33682	170.84061
Rho P	0.96400	0.62524	0.98836
Rho M	0.03600	0.01164	0.37476
Var(a)/Gamma Y	0.96400	0.62524	0.98836
Var(b)/Gamma Y	0.01127	0.0009352	0.35629
Var(a*b)/Gamma Y	0.01453	0.0032977	0.04258
Var(a)/Var(Error)	94.48551	40.62919	325.51792
Var(b)/Var(Error)	1.10467	0.13051	56.94205
Var(a*b)/Var(Error)	1.42432	0.55232	3.74691

Note that the generalized confidence interval widths from Output 130.2.4 for parameters γ_R and DR are 85.7 and 171.4, respectively. These widths are much shorter than the MLS-based widths, which are 103.9 and 207.8 from Output 130.2.3.

In general, the GCL method provides a more accurate confidence interval with a shorter interval width than the MLS method. However, as discussed in the section "Generalized Confidence Limits" on page 10947, they are computationally intensive and somewhat nondeterministic, because they are based on an underlying Monte Carlo simulation.

References

Burdick, R. K., Borror, C. M., and Montgomery, D. C. (2003). "A Review of Methods for Measurement Systems Capability Analysis." *Journal of Quality Technology* 35:342–354.

Burdick, R. K., Borror, C. M., and Montgomery, D. C. (2005). *Design and Analysis of Gauge R&R Studies: Making Decisions with Confidence Intervals in Random and Mixed ANOVA Models*. Philadelphia, PA, and Alexandria, VA: SIAM and ASA.

Gaylor, D. W., Lucas, H. L., and Anderson, R. L. (1970). "Calculation of Expected Mean Squares by the Abbreviated Doolittle and Square Root Methods." *Biometrics* 26:641–655.

Goodnight, J. H. (1978). *Computing MIVQUE0 Estimates of Variance Components*. Technical Report R-105, SAS Institute Inc., Cary, NC.

Goodnight, J. H., and Hemmerle, W. J. (1979). "A Simplified Algorithm for the W-Transformation in Variance Component Estimation." *Technometrics* 21:265–267.

Graybill, F. A., and Wang, C. M. (1980). "Confidence Intervals on Nonnegative Linear Combinations of Variances." *Journal of the American Statistical Association* 75:869–873.

Hartley, H. O., Rao, J. N. K., and LaMotte, L. R. (1978). "A Simple Synthesis-Based Method of Variance Component Estimation." *Biometrics* 34:233–244.

Hemmerle, W. J., and Hartley, H. O. (1973). "Computing Maximum Likelihood Estimates for the Mixed AOV Model Using the W-Transformation." *Technometrics* 15:819–831.

Hicks, C. R. (1973). *Fundamental Concepts in the Design of Experiments*. New York: Holt, Rinehart & Winston.

Hocking, R. R. (1983). "A Diagnostic Tool for Mixed Models with Application to Negative Estimates of Variance Components." In *Proceedings of the Eighth Annual SAS Users Group International Conference*, 711–716. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings-archive/SUGI83/Sugi-83-138%20Hocking.pdf.

Hocking, R. R. (1985). *The Analysis of Linear Models*. Monterey, CA: Brooks/Cole.

Houf, R. E., and Burman, D. B. (1988). "Statistical Analysis of Power Module Thermal Test Equipment Performance." *IEEE Transactions on Components, Hybrids, and Manufacturing Technology* 11:516–520.

Patterson, H. D., and Thompson, R. (1971). "Recovery of Inter-block Information When Block Sizes Are Unequal." *Biometrika* 58:545–554.

Rao, C. R. (1971). "Minimum Variance Quadratic Unbiased Estimation of Variance Components." *Journal of Multivariate Analysis* 1:445–456.

Rao, C. R. (1972). "Estimation of Variance and Covariance Components in Linear Models." *Journal of the American Statistical Association* 67:112–115.

Searle, S. R., Casella, G., and McCulloch, C. E. (1992). *Variance Components*. New York: John Wiley & Sons.

Weerahandi, S. (1993). "Generalized Confidence Intervals." *Journal of the American Statistical Association* 88:899–905.

Chapter 131
The VARIOGRAM Procedure

Contents

Overview: VARIOGRAM Procedure . . . **10960**
- Introduction to Spatial Prediction . . . 10961

Getting Started: VARIOGRAM Procedure . . . **10962**
- Preliminary Spatial Data Analysis . . . 10962
- Empirical Semivariogram Computation . . . 10967
- Autocorrelation Analysis . . . 10970
- Theoretical Semivariogram Model Fitting . . . 10971

Syntax: VARIOGRAM Procedure . . . **10975**
- PROC VARIOGRAM Statement . . . 10976
- BY Statement . . . 10983
- COMPUTE Statement . . . 10984
- COORDINATES Statement . . . 10989
- DIRECTIONS Statement . . . 10990
- ID Statement . . . 10990
- MODEL Statement . . . 10990
- NLOPTIONS Statement . . . 11002
- PARMS Statement . . . 11003
- STORE Statement . . . 11007
- VAR Statement . . . 11007

Details: VARIOGRAM Procedure . . . **11008**
- Theoretical Semivariogram Models . . . 11008
  - Characteristics of Semivariogram Models . . . 11009
  - Nested Models . . . 11013
- Theoretical and Computational Details of the Semivariogram . . . 11013
  - Stationarity . . . 11015
  - Ergodicity . . . 11015
  - Anisotropy . . . 11016
  - Pair Formation . . . 11016
  - Angle Classification . . . 11017
  - Distance Classification . . . 11019
  - Bandwidth Restriction . . . 11020
  - Computation of the Distribution Distance Classes . . . 11021
  - Semivariance Computation . . . 11025
  - Empirical Semivariograms and Surface Trends . . . 11026
- Theoretical Semivariogram Model Fitting . . . 11027
  - Parameter Initialization . . . 11029

Parameter Estimates 11031
Quality of Fit 11031
Fitting with Matérn Forms 11034
Autocorrelation Statistics 11035
Autocorrelation Weights 11035
Autocorrelation Statistics Types 11036
Interpretation 11038
The Moran Scatter Plot 11038
Computational Resources 11040
Output Data Sets 11040
Displayed Output 11044
ODS Table Names 11045
ODS Graphics 11046
Examples: VARIOGRAM Procedure **11047**
Example 131.1: Aspects of Semivariogram Model Fitting 11047
Example 131.2: An Anisotropic Case Study with Surface Trend in the Data 11058
Analysis with Surface Trend Removal 11062
Example 131.3: Analysis without Surface Trend Removal 11071
Example 131.4: Covariogram and Semivariogram 11080
Example 131.5: A Box Plot of the Square Root Difference Cloud 11084
References **11087**

Overview: VARIOGRAM Procedure

The VARIOGRAM procedure computes empirical measures of spatial continuity for two-dimensional spatial data. These measures are a function of the distances between the sample data pairs. When the data are free of nonrandom (or systematic) surface trends, the estimated continuity measures are the empirical semivariance and covariance. The procedure also fits permissible theoretical models to the empirical semivariograms, so that you can use them in subsequent analysis to perform spatial prediction. You can produce plots of the empirical semivariograms in addition to plots of the fitted models. Both isotropic and anisotropic continuity measures are available.

PROC VARIOGRAM also provides the Moran's I and Geary's c spatial autocorrelation statistics, in addition to the Moran scatter plot to visualize spatial associations within a specified neighborhood around observations. The procedure produces the OUTVAR=, OUTPAIR=, and OUTDISTANCE= data sets that contain information about the semivariogram analysis. Also, the OUTACWEIGHTS= and the OUTMORAN= output data sets contain information about the autocorrelation analysis.

The VARIOGRAM procedure uses ODS Graphics to create graphs as part of its output. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For more information about the graphics available in PROC VARIOGRAM, see the section "ODS Graphics" on page 11046.

Introduction to Spatial Prediction

Many activities in science and technology involve measurements of one or more quantities at given spatial locations, with the goal of predicting the measured quantities at unsampled locations. Application areas include reservoir prediction in mining and petroleum exploration, in addition to modeling in a broad spectrum of fields (for example, environmental health, environmental pollution, natural resources and energy, hydrology, and risk analysis). Often, the unsampled locations are on a regular grid, and the predictions are used to produce surface plots or contour maps.

The preceding tasks fall within the scope of *spatial prediction*, which, in general, is any prediction method that incorporates spatial dependence. The study of these tasks involves naturally occurring uncertainties that cannot be ignored. Stochastic analysis frameworks and methods are often used to account for these uncertainties. Hence, the terms *stochastic spatial prediction* and *stochastic modeling* are also used to characterize this type of analysis.

A popular method of spatial prediction is *ordinary kriging*, which produces both predicted values and associated standard errors. Ordinary kriging requires the complete specification (the form and parameter values) of the spatial dependence that characterizes the spatial process. For this purpose, models for the spatial dependence are expressed in terms of the distance between any two locations in the spatial domain of interest. These models take the form of a covariance or semivariance function.

Spatial prediction, then, involves two steps. First, you model the covariance or semivariance of the spatial process. These measures are typically not known in advance. This step involves computing an empirical estimate, in addition to determining both the mathematical form and the values of any parameters for a theoretical form of the dependence model. Second, you use this dependence model to solve the kriging system at a specified set of spatial points, resulting in predicted values and associated standard errors.

SAS/STAT software has two procedures that correspond to these steps for spatial prediction of two-dimensional data. The VARIOGRAM procedure is used in the first step (that is, calculating and modeling the dependence model), and the KRIGE2D procedure performs the kriging operations to produce the final predictions.

This introduction concludes with a note on terminology. You might commonly encounter the terms *estimation* and *prediction* used interchangeably by experts in different fields; this could be a source of confusion. A precise statistical vernacular uses the term *estimation* to refer to inferences about the value of fixed but unknown parameters, whereas *prediction* concerns inferences about the value of random variables—see, for example, Cressie (1993, p. 106). In light of these definitions, kriging methods are clearly predictive techniques, since they are concerned with making inferences about the value of a spatial random field at observed or unobserved locations. The SAS/STAT suite of procedures for spatial analysis and prediction (VARIOGRAM, KRIGE2D, and SIM2D) follows the statistical vernacular in the use of the terms *estimation* and *prediction*.

Getting Started: VARIOGRAM Procedure

PROC VARIOGRAM uses your data to compute the empirical semivariogram. This computation refers to the steps you take to derive the empirical semivariance from the data, and then to produce the corresponding semivariogram plot.

You can proceed further with the semivariogram analysis if the data are free of systematic trends. In that case, you can use the empirical outcome to determine a theoretical semivariogram model by using the automated methods provided by the VARIOGRAM procedure. The model characterizes the type of theoretical semivariance function you use to describe spatial dependence in your data set.

Graphical displays are requested by enabling ODS Graphics. For general information about ODS Graphics, see Chapter 24, "Statistical Graphics Using ODS." For specific information about the graphics available in the VARIOGRAM procedure, see the section "ODS Graphics" on page 11046.

Preliminary Spatial Data Analysis

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/r-z/vrogs1.sas.)

The thick data set is available from the Sashelp library. The data set simulates measurements of coal seam thickness (in feet) taken over an approximately square area. The Thick variable has the thickness values in the thick data set. The coordinates are offsets from a point in the southwest corner of the measurement area, with the north and east distances in units of thousands of feet.

It is instructive to see the locations of the measured points in the area where you want to perform spatial prediction. It is desirable to have the sampling locations scattered evenly throughout the prediction area. If the locations are not scattered evenly, the prediction error might be unacceptably large where measurements are sparse.

You can run PROC VARIOGRAM in this preliminary analysis to determine potential problems. In the following statements, the NOVARIOGRAM option in the COMPUTE statement specifies that only the descriptive summaries and a plot of the raw data be produced.

```
title 'Spatial Correlation Analysis with PROC VARIOGRAM';

ods graphics on;

proc variogram data=sashelp.thick plots=pairs(thr=30);
   compute novariogram nhc=20;
   coordinates xc=East yc=North;
   var Thick;
run;
```

PROC VARIOGRAM produces the table in Figure 131.1 that shows the number of Thick observations read and used. This table provides you with useful information in case you have missing values in the input data.

Figure 131.1 Number of Observations for the thick Data Set

Spatial Correlation Analysis with PROC VARIOGRAM

The VARIOGRAM Procedure

Dependent Variable: Thick

Number of Observations Read	75
Number of Observations Used	75

Then, the scatter plot of the observed data is produced as shown in Figure 131.2. According to the figure, although the locations are not ideally spread around the prediction area, there are not any extended areas lacking measurements. The same graph also provides the values of the measured variable by using colored markers.

Figure 131.2 Scatter Plot of the Observations Spatial Distribution

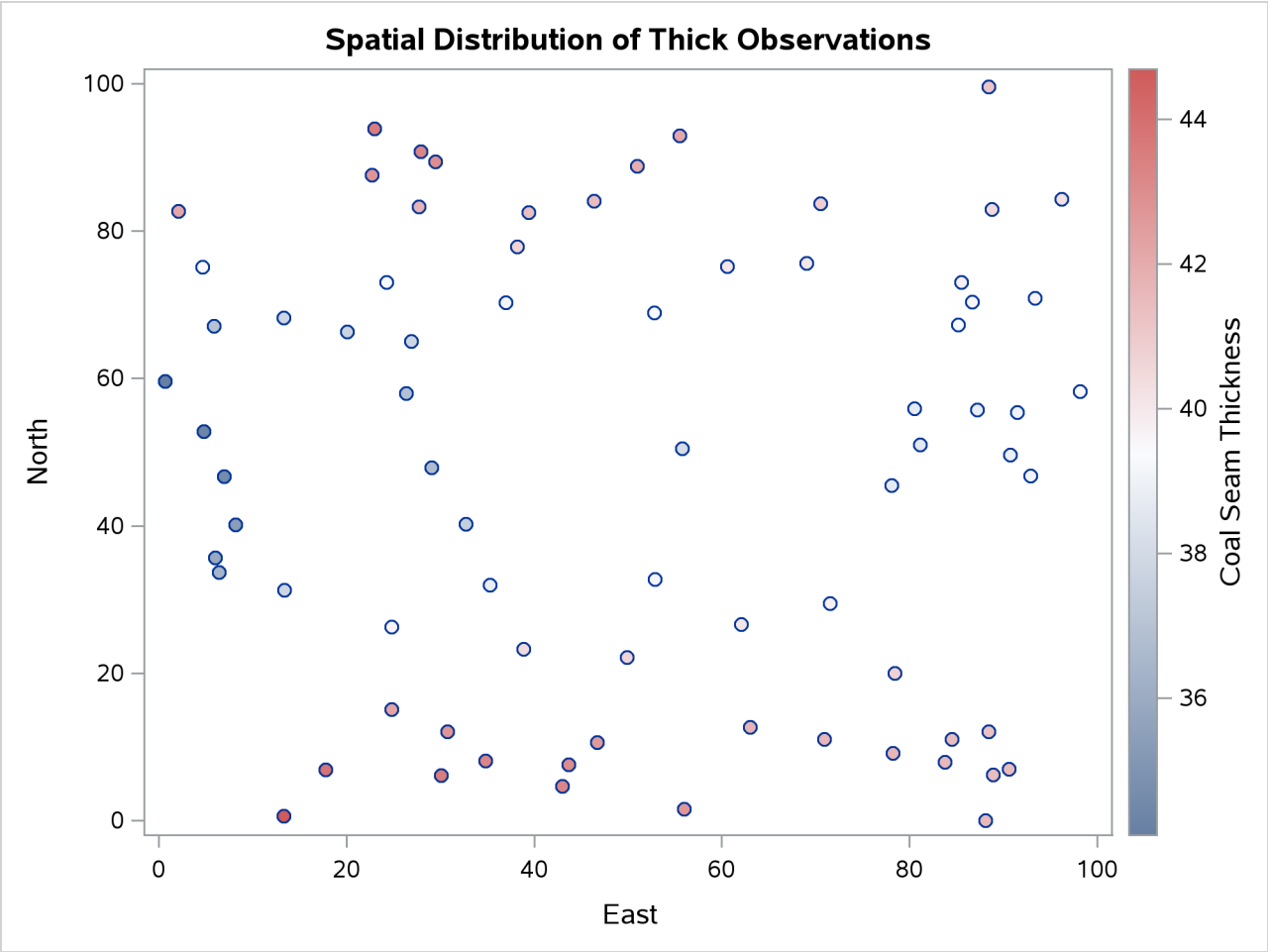

The following is a crucial step. Any obvious surface trend must be removed before you compute the empirical semivariogram and proceed to estimate a model of spatial dependence (the theoretical semivariogram model). You can observe in Figure 131.2 the small-scale variation typical of spatial data, but a first inspection indicates no obvious major systematic trend.

Assuming, therefore, that the data are free of surface trends, you can work with the original thickness rather than residuals obtained from a trend removal process. The following analysis also assumes that the spatial characterization is independent of the direction of the line that connects any two equidistant pairs of data; this is a property known as isotropy. See "Example 131.2: An Anisotropic Case Study with Surface Trend in the Data" on page 11058 for a more detailed approach to trend analysis and the issue of anisotropy.

Following the previous exploratory analysis, you then need to classify each data pair as a member of a distance interval (lag). PROC VARIOGRAM performs this grouping with two required options for semivariogram computation: the LAGDISTANCE= and MAXLAGS= options. These options are based on your assessment of how to group the data pairs within distance classes.

The meaning of the required LAGDISTANCE= option is as follows. Classify all pairs of points into intervals according to their pairwise distance. The width of each distance interval is the LAGDISTANCE= value. The meaning of the required MAXLAGS= option is simply the number of intervals you consider. The problem is that given only the scatter plot of the measurement locations, it is not clear what values to give to the LAGDISTANCE= and MAXLAGS= options.

Ideally, you want a sufficient number of distance classes that capture the extent to which your data are correlated and you want each class to contain a minimum of data pairs to increase the accuracy in your computations. A rule of thumb used in semivariogram computations is that you should have at least 30 pairs per lag class. This is an empirical arbitrary threshold; see the section "Choosing the Size of Classes" on page 11023 for further details.

In the preliminary analysis, you use the option NHCLASSES= in the COMPUTE statement to help you experiment with these numbers and choose values for the LAGDISTANCE= and MAXLAGS= options. Here, in particular, you request NHCLASSES=20 to preview a classification that uses 20 distance classes across your spatial domain. A zero lag class is always considered; therefore the output shows the number of distance classes to be one more than the number you specified.

Based on your selection of the NHCLASSES= option, the NOVARIOGRAM option produces a pairwise distances table from your observations shown in Figure 131.3, and the corresponding histogram in Figure 131.4. For illustration purposes, you also specify a threshold of minimum data pairs per distance class in the PAIRS option as THR=30. As a result, a reference line appears in the histogram so that you can visually identify any lag classes with pairs that fall below your specified threshold.

Figure 131.3 Pairwise Distance Intervals Table

Pairwise Distance Intervals				
Lag Class	Bounds		Number of Pairs	Percentage of Pairs
0	0.00	3.48	7	0.25%
1	3.48	10.45	81	2.92%
2	10.45	17.42	138	4.97%
3	17.42	24.39	167	6.02%
4	24.39	31.36	204	7.35%
5	31.36	38.33	210	7.57%
6	38.33	45.30	213	7.68%
7	45.30	52.27	253	9.12%
8	52.27	59.24	237	8.54%
9	59.24	66.20	280	10.09%
10	66.20	73.17	252	9.08%
11	73.17	80.14	230	8.29%
12	80.14	87.11	217	7.82%
13	87.11	94.08	154	5.55%
14	94.08	101.05	71	2.56%
15	101.05	108.02	41	1.48%
16	108.02	114.99	14	0.50%
17	114.99	121.96	5	0.18%
18	121.96	128.93	1	0.04%
19	128.93	135.89	0	0.00%
20	135.89	142.86	0	0.00%

The NOVARIOGRAM option also produces a table with useful facts about the pairs and the distances between the most remote data in selected directions, shown in Figure 131.5. In particular, the lag distance value is calculated based on your selection of the NHCLASSES= option. The last three table entries report the overall maximum distance among your data pairs, in addition to the maximum distances in the main axes directions—that is, the vertical (N–S) axis and the horizontal (E–W) axis. This information is also provided in the inset of Figure 131.4. When you specify a threshold in the PAIRS suboption of the PLOTS option, as in this example, the threshold also appears in the table. Then, the line that follows indicates the highest lag class with the following property: each one of the distance classes that lie farther away from this lag features a pairs population below the specified threshold.

With the preceding information you can determine appropriate values for the LAGDISTANCE= and MAXLAGS= options in the COMPUTE statement. In particular, the classification that uses 20 distance classes is satisfactory, and you can choose LAGDISTANCE=7 after following the suggestion in Figure 131.5.

Figure 131.4 Distribution of Pairwise Distances

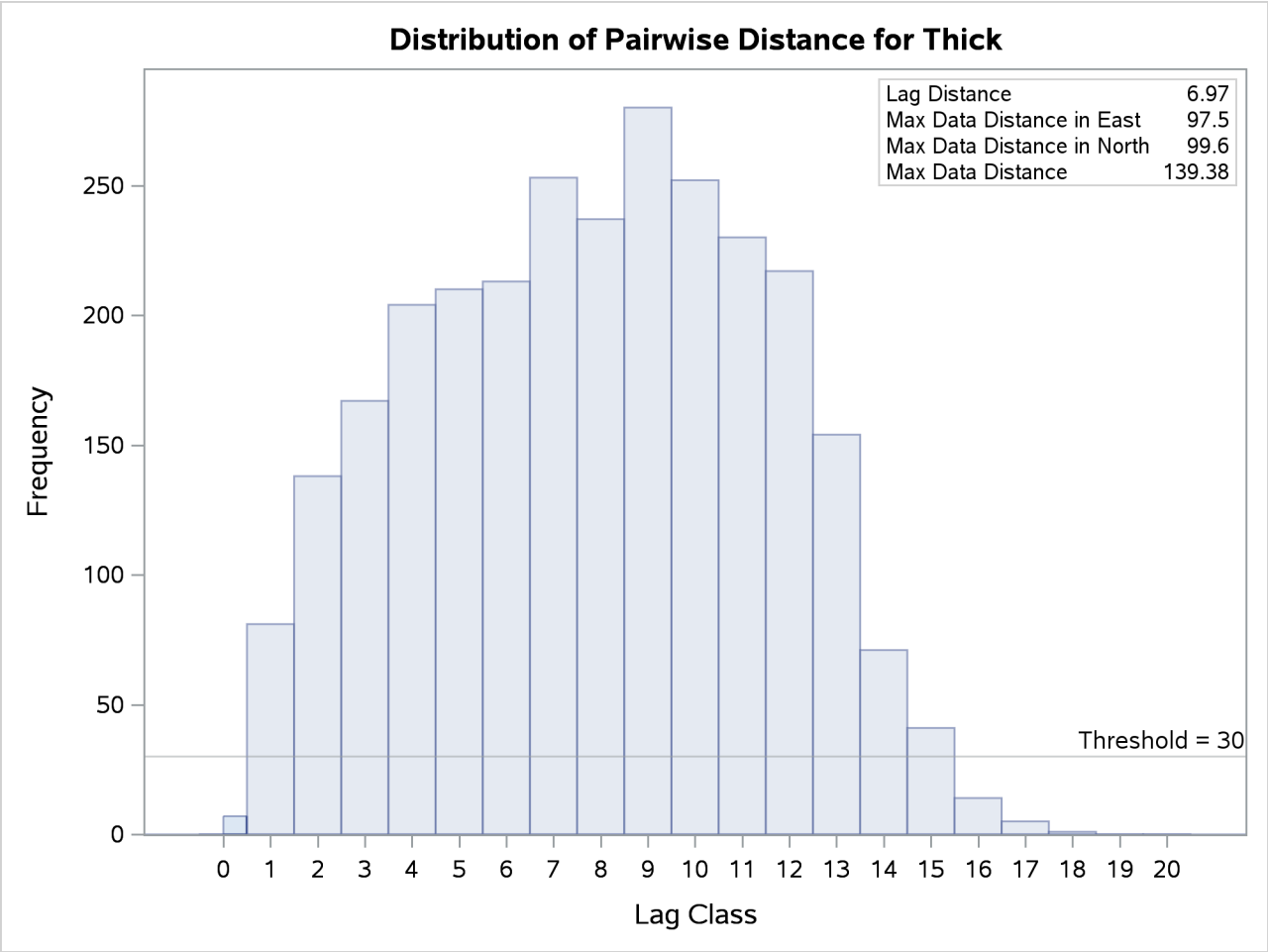

Figure 131.5 Pairs Information Table

Pairs Information	
Number of Lags	21
Lag Distance	6.97
Minimum Pairs Threshold	30
Highest Lag With Pairs > Threshold	15
Maximum Data Distance in East	97.50
Maximum Data Distance in North	99.60
Maximum Data Distance	139.38

The MAXLAGS= option needs to be specified based on the spatial extent to which your data are correlated. Unless you know this size, in the present omnidirectional case you can assume the correlation extent to be roughly equal to half the overall maximum distance between data points.

The table in Figure 131.5 suggests that this number corresponds to 139,380 feet, which is most likely on or close to a diagonal direction (that is, the northeast–southwest or northwest–southeast direction). Hence, you can expect the correlation extent in this scale to be around $139.4/2 = 69{,}700$ feet. Consequently, consider

lag classes up to this distance for the empirical semivariogram computations. Given your lag size selection, Figure 131.3 indicates that this distance corresponds to about 10 lags; hence you can set MAXLAGS=10.

Overall, for a specific NHCLASSES= choice of class count, you can expect your choice of MAXLAGS= to be approximately half the number of the lag classes (see the section "Spatial Extent of the Empirical Semivariogram" on page 11024 for more details).

After you have starting values for the LAGDISTANCE= and MAXLAGS= options, you can run the VARIOGRAM procedure multiple times to inspect and compare the results you get by specifying different values for these *options*.

Empirical Semivariogram Computation

Using the values of LAGDISTANCE=7 and MAXLAGS=10 computed previously, rerun PROC VARIOGRAM without the NOVARIOGRAM option in order to compute the empirical semivariogram. You specify the CL option in the COMPUTE statement to calculate the 95% confidence limits for the classical semivariance. The section "COMPUTE Statement" on page 10984 describes how to use the ALPHA= option to specify a different confidence level.

Also, you can request a robust version of the semivariance with the ROBUST option in the COMPUTE statement. PROC VARIOGRAM produces a plot that shows both the classical and the robust empirical semivariograms. See the details of the PLOTS option to specify different instances of plots of the empirical semivariogram. The following statements implement the preceding requests:

```
proc variogram data=sashelp.thick outv=outv;
   compute lagd=7 maxlag=10 cl robust;
   coordinates xc=East yc=North;
   var Thick;
run;
```

Figure 131.6 displays the PROC VARIOGRAM output empirical semivariogram table for the preceding statements. The table displays a total of eleven lag classes, even though you specified MAXLAGS=10. The VARIOGRAM procedure always includes a zero lag class in the computations in addition to the MAXLAGS classes you request with the MAXLAGS= option. Hence, semivariance is actually computed at MAXLAGS+1 lag classes; see the section "Distance Classification" on page 11019 for more details.

Figure 131.6 Output Table for the Empirical Semivariogram Analysis

Spatial Correlation Analysis with PROC VARIOGRAM

The VARIOGRAM Procedure

Dependent Variable: Thick

Empirical Semivariogram							
			Semivariance				
Lag Class	Pair Count	Average Distance	Robust	Classical	Standard Error	95% Confidence Limits	
0	7	2.64	0.028	0.034	0.018	0	0.069
1	82	7.29	0.210	0.394	0.061	0.273	0.514
2	138	14.16	1.008	1.179	0.142	0.901	1.458
3	169	21.08	3.018	2.799	0.304	2.202	3.396
4	205	27.93	4.811	4.602	0.455	3.711	5.493
5	213	35.17	5.990	5.928	0.574	4.802	7.054
6	214	42.20	8.104	7.518	0.727	6.094	8.943
7	250	48.78	7.533	7.221	0.646	5.955	8.487
8	247	56.16	8.066	7.195	0.647	5.926	8.464
9	281	62.89	8.279	6.845	0.577	5.713	7.976
10	250	69.93	8.144	6.358	0.569	5.243	7.472

Figure 131.7 shows both the classical and robust empirical semivariograms. In addition, the plot features the approximate 95% confidence limits for the classical semivariance. The figure exhibits a typical behavior of the computed semivariance uncertainty, where in general the variance increases with distance from the origin at Distance=0.

Figure 131.7 Classical and Robust Empirical Semivariograms for Coal Seam Thickness Data

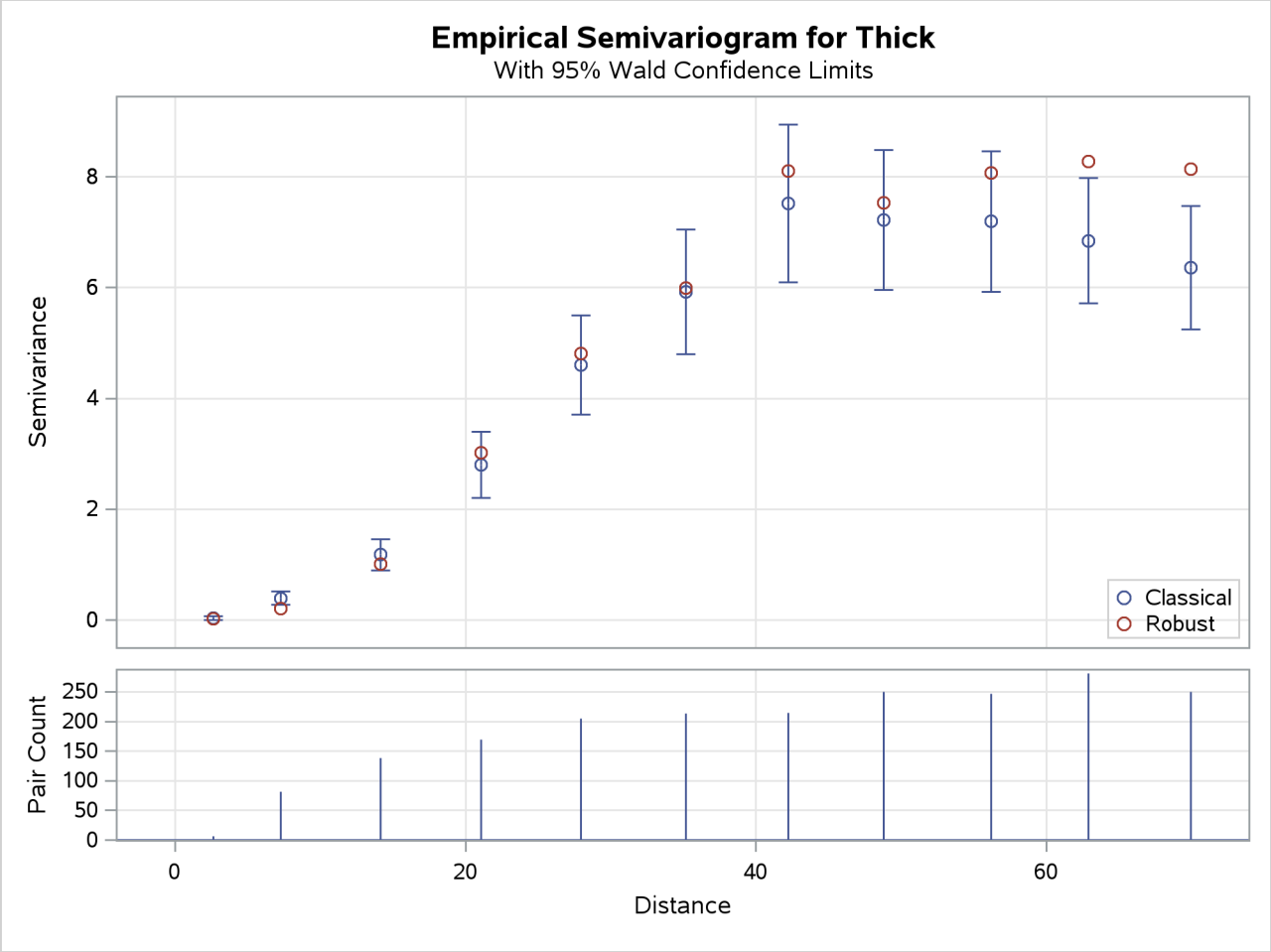

The needle plot in the lower part of the Figure 131.7 provides the number of pairs that were used in the computation of the empirical semivariance for each lag class shown. In general, this is a pairwise distribution that is different from the distribution depicted in Figure 131.4. First, the number of pairs shown in the needle plot depends on the particular criteria you specify in the COMPUTE statement of PROC VARIOGRAM. Second, the distances shown for each lag on the Distance axis are not the midpoints of the lag classes as in the pairwise distances plot, but rather the average distance from the origin Distance=0 of all pairs in a given lag class.

Autocorrelation Analysis

You can use the autocorrelation analysis features of PROC VARIOGRAM to compute the autocorrelation Moran's I and Geary's c statistics and to obtain the Moran scatter plot. In the following statements, you ask for the Moran's I and Geary's c statistics under the assumption of randomization using binary weights, in addition to the Moran scatter plot:

```
proc variogram data=sashelp.thick outv=outv plots(only)=moran;
   compute lagd=7 maxlag=10 autocorr(assum=random);
   coordinates xc=East yc=North;
   var Thick;
run;
```

For the autocorrelation analysis with binary weights and the Moran scatter plot, the LAGDISTANCE= option indicates that you consider as neighbors of an observation all other observations within the specified distance from it.

Figure 131.8 shows the output from the requested autocorrelation analysis. This includes the observed (computed) Moran's I and Geary's c coefficients, the expected value and standard deviation for each coefficient, the corresponding Z score, and the p-value in the Pr >| Z | column. The low p-values suggest strong autocorrelation for both statistics types. A two-sided p-value is reported, which is the probability that the observed coefficient lies farther away from | Z | on either side of the coefficient's expected value—that is, lower than $-Z$ or higher than Z. The sign of Z for both Moran's I and Geary's c coefficients indicates positive autocorrelation in the Thick data values; see the section "Interpretation" on page 11038 for more details.

Figure 131.8 Output Table for the Autocorrelation Statistics

Spatial Correlation Analysis with PROC VARIOGRAM

The VARIOGRAM Procedure

Dependent Variable: Thick

Autocorrelation Statistics

Assumption	Coefficient	Observed	Expected	Std Dev	Z	Pr > |Z|
Randomization	Moran's I	0.9240	-0.0244	0.145	6.53	<.0001
Randomization	Geary's c	0.0162	1.0000	0.175	-5.62	<.0001

The requested Moran scatter plot is shown in Figure 131.9. The plot includes all nonmissing observations that have neighbors within the specified LAGDISTANCE= distance. The horizontal axis displays the standardized Thick values, and the vertical axis displays the corresponding weighted average of their neighbors. The plot data points are concentrated in the upper right and lower left quadrants defined by the lines $x = 0$ and $y = 0$, and clearly around the axes' diagonal reference line $y = x$ of slope 1. This fact indicates strong positive spatial association in the thick data set observations. Therefore, for each observation its neighbors within the specified LAGDISTANCE= distance have overall similar Thick values to that observation. The plot also displays the linear regression slope, whose value is the Moran's I coefficient when the binary weights are row-averaged. See the section "The Moran Scatter Plot" on page 11038 for more details about the Moran scatter plot.

Figure 131.9 Moran Scatter Plot for Coal Seam Thickness Data

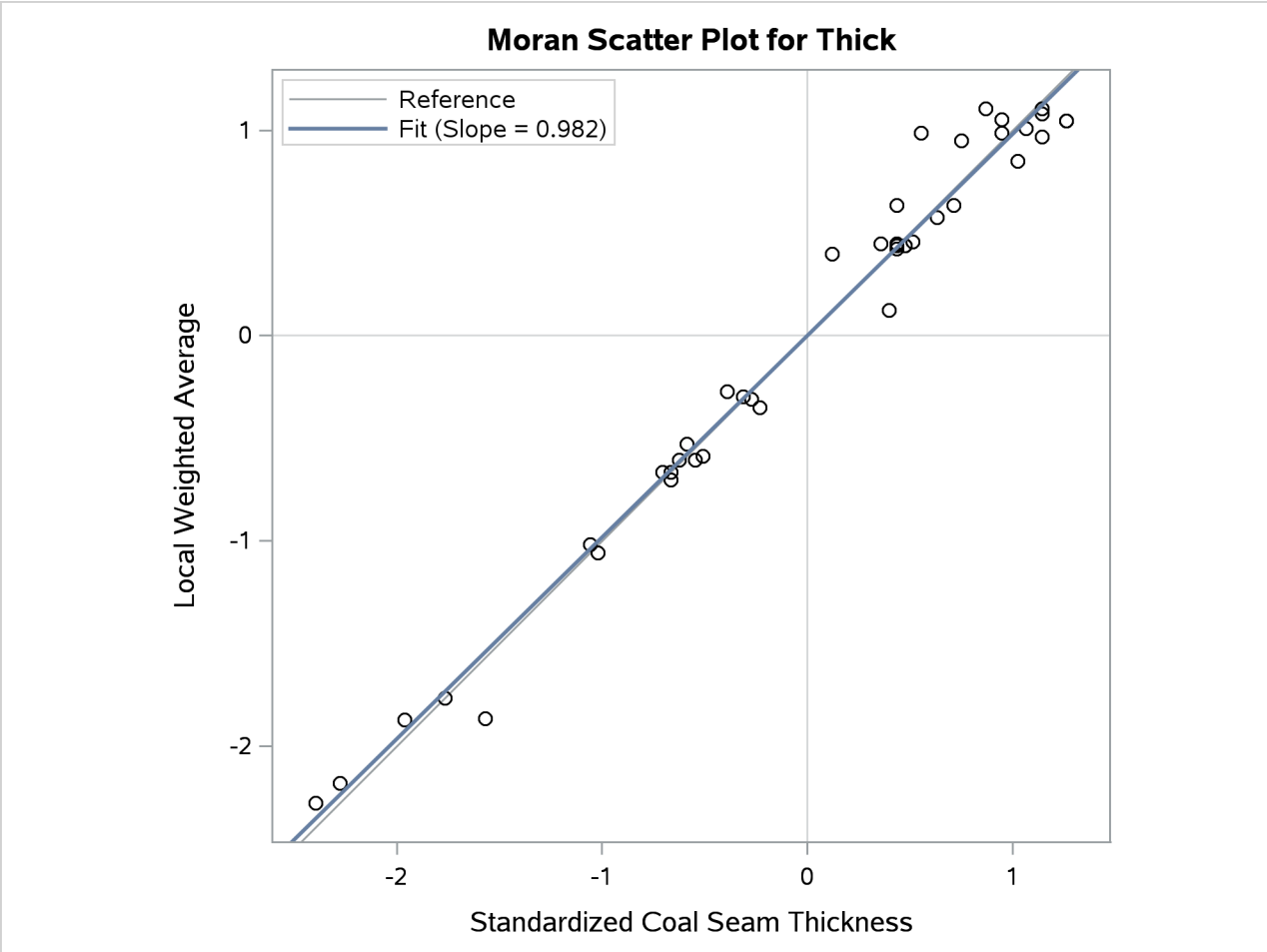

Theoretical Semivariogram Model Fitting

PROC VARIOGRAM features automated semivariogram fitting. In particular, the procedure selects a theoretical semivariogram model to fit the empirical semivariance and produces estimates of the model parameters in addition to a fit plot. You have the option to save these estimates in an item store, which is a binary file format that is defined by the SAS System and that you cannot modify. Then, you can retrieve this information at a later point from the item store for future analysis with PROC KRIGE2D or PROC SIM2D.

The coal seam thickness empirical semivariogram in Figure 131.7 shows first a slow, then rapid, rise from the origin. This behavior suggests that you can approximate the empirical semivariance with a Gaussian-type form

$$\gamma_z(h) = c_0 \left[1 - \exp\left(-\frac{h^2}{a_0^2} \right) \right]$$

as shown in the section "Theoretical Semivariogram Models" on page 11008. Based on this remark, you choose to fit a Gaussian model to your classical semivariogram. Run PROC VARIOGRAM again and specify

the MODEL statement with the FORM=GAU option. By default, PROC VARIOGRAM uses the weighted least squares (WLS) method to fit the specified model, although you can explicitly specify the METHOD= option to request the fitting method. You want additional information about the estimated parameters, so you specify the CL option in the MODEL statement to compute their 95% confidence limits and the COVB option of the MODEL statement to produce a table with their approximate covariances. You also specify the STORE statement to save the fitting outcome into an item store file with the name SemivStoreGau and a desired label. You run the following statements:

```
proc variogram data=sashelp.thick outv=outv;
   store out=SemivStoreGau / label='Thickness Gaussian WLS Fit';
   compute lagd=7 maxlag=10;
   coordinates xc=East yc=North;
   model form=gau cl / covb;
   var Thick;
run;

ods graphics off;
```

After you run the procedure you get a series of output objects from the fitting analysis. In particular, Figure 131.10 shows first a model fitting table with the name and a short label of the model that you requested to use for the fit. The table also displays the name and label of the specified item store.

Figure 131.10 Semivariogram Model Fitting General Information

Spatial Correlation Analysis with PROC VARIOGRAM

The VARIOGRAM Procedure

Dependent Variable: Thick
Angle: Omnidirectional
Current Model: Gaussian

Semivariogram Model Fitting	
Name	Gaussian
Label	Gau
Output Item Store	WORK.SEMIVSTOREGAU
Item Store Label	Thickness Gaussian WLS Fit

If you specify no parameters, as in the current example, then PROC VARIOGRAM initializes the model parameters for you with default values based on the empirical semivariance; for more details, see the section "Theoretical Semivariogram Model Fitting" on page 11027. The initial values provided by the VARIOGRAM procedure for the Gaussian model are displayed in the table in Figure 131.11.

Figure 131.11 Semivariogram Fitting Model Information

Model Information	
Parameter	**Initial Value**
Nugget	0
Scale	6.7992
Range	34.9635

Otherwise, in PROC VARIOGRAM you can specify initial values for parameters with the PARMS statement. Alternatively, you can specify fixed values for the model scale and range with the SCALE= and RANGE= options, respectively, in the MODEL statement. A nugget effect is always used in model fitting. Unless you explicitly specify a fixed nugget effect with the NUGGET= option in the MODEL statement or initialize the nugget parameter in the PARMS statement, the nugget effect is automatically initialized to zero. See the section "Syntax: VARIOGRAM Procedure" on page 10975 for more details about how the MODEL statement and the PARMS statement handle model parameters.

The output in Figure 131.12 comes from the optimization process that takes place during the model parameter estimation. The optimizer produces an optimization information table, information about the optimization technique that is used, optimization-related results, and notification about the optimization convergence.

Figure 131.12 Fitting Optimization Information

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	3
Lower Boundaries	3
Upper Boundaries	0
Starting Values From	PROC

Spatial Correlation Analysis with PROC VARIOGRAM

The VARIOGRAM Procedure

Dependent Variable: Thick
Angle: Omnidirectional
Current Model: Gaussian

Dual Quasi-Newton Optimization

Dual Broyden - Fletcher - Goldfarb - Shanno Update (DBFGS)

Hessian Computed by Finite Differences (Using Analytic Gradient)

Optimization Results			
Iterations	13	**Function Calls**	44
Gradient Calls	0	**Active Constraints**	1
Objective Function	11.433894152	**Max Abs Gradient Element**	4.316397E-6
Slope of Search Direction	-1.032129E-9		

Convergence criterion (GCONV=1E-8) satisfied.

The fitting process is successful, and the parameters converge to the estimated values shown in Figure 131.13. For each parameter, the same table also displays the approximate standard error, the degrees of freedom, the t value, the approximate p-value, and the requested 95% confidence limits.

Figure 131.13 Semivariogram Fitting Parameter Estimates

Parameter Estimates							
			Approximate 95% Confidence Limits				
Parameter	Estimate	Approx Std Error	Lower	Upper	DF	t Value	Approx Pr > |t|
Nugget	0	0	0	0	8	.	.
Scale	7.4599	0.2621	6.8555	8.0643	8	28.46	<.0001
Range	30.1111	1.1443	27.4724	32.7498	8	26.31	<.0001

The approximate covariance matrix of the estimated parameters is displayed in Figure 131.14.

Figure 131.14 Approximate Covariance Matrix of Parameter Estimates

Approximate Covariance Matrix			
Parameter	Nugget	Scale	Range
Nugget	0.0000	0.0000	0.0000
Scale	0.0000	0.0687	0.2326
Range	0.0000	0.2326	1.3094

The fitting summary table in Figure 131.15 displays statistics about the quality of the fitting process. In particular, the table shows the weighted error sum of squares in the Weighted SSE column and the Akaike information criterion in the AIC column. See more information about the fitting criteria in section "Quality of Fit" on page 11031.

Figure 131.15 Semivariogram Model Fitting Summary

Fit Summary		
Model	Weighted SSE	AIC
Gau	11.43389	6.42556

Figure 131.16 demonstrates the fitted theoretical semivariogram against the empirical semivariance estimates with the weighted least squares method. The fit seems to be more accurate closer to the origin $\boldsymbol{h} = 0$, and this is explained as follows: A smaller $\boldsymbol{h}$ corresponds to smaller semivariance; in turn, this corresponds to smaller semivariance variance, as shown in the section "Theoretical and Computational Details of the Semivariogram" on page 11013. By definition, the WLS optimization weights increase with decreasing variance, which leads to a more accurate fit for smaller distances $\boldsymbol{h}$ in the WLS fitting results.

Figure 131.16 Fitted Theoretical and Empirical Semivariogram for Coal Seam Thickness

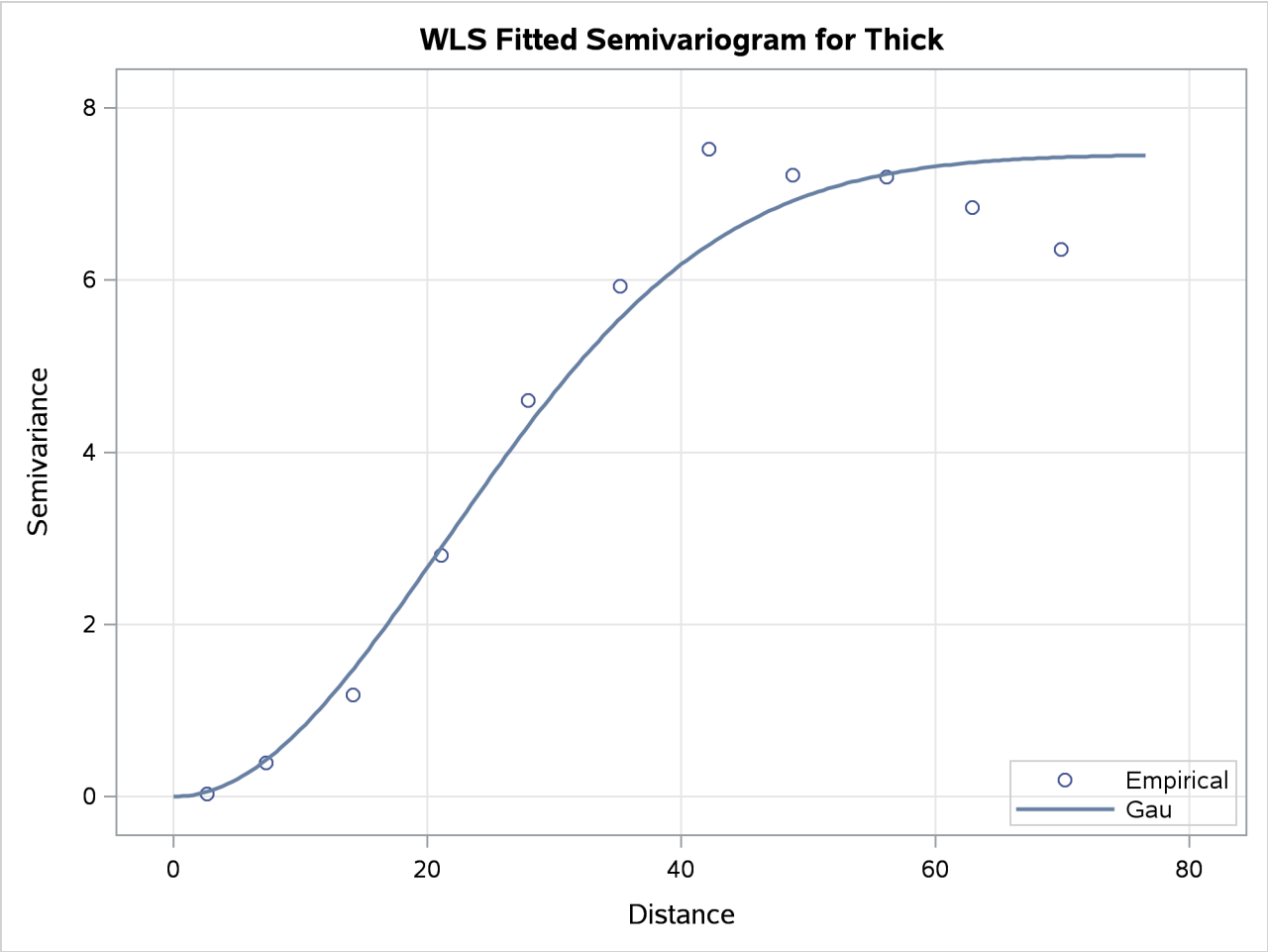

Syntax: VARIOGRAM Procedure

The following statements are available in the VARIOGRAM procedure:

PROC VARIOGRAM *options* ;
  BY *variables* ;
  COMPUTE *computation-options* ;
  COORDINATES *coordinate-variables* ;
  DIRECTIONS *directions-list* ;
  ID *variable* ;
  MODEL *model-options* ;
  PARMS *parameters-list* < / *parameters-options* > ;
  NLOPTIONS < *options* > ;
  STORE *store-options* ;
  VAR *analysis-variables-list* ;

The COMPUTE and COORDINATES statements are required. The MODEL and PARMS statements are hierarchical. If you specify a PARMS statement, it must follow a MODEL statement.

PROC VARIOGRAM Statement

PROC VARIOGRAM *options* **;**

The PROC VARIOGRAM statement invokes the VARIOGRAM procedure. Table 131.1 summarizes the *options* available in the PROC VARIOGRAM statement.

Table 131.1 PROC VARIOGRAM Statement Options

Option	Description
DATA=	Specifies the input data set
IDGLOBAL	Labels observations across BY groups using ascending observation numbers
IDNUM	Labels observations using the observation number
NOPRINT	Suppresses normal display of results
OUTACWEIGHTS=	Specifies a data set to store autocorrelation weights information
OUTDISTANCE=	Specifies a data set to store summary distance information
OUTMORAN=	Specifies a data set to store Moran scatter plot information
OUTPAIR=	Specifies a data set to store pairwise point information
OUTVAR=	Specifies a data set to store spatial continuity measures
PLOTS	Specifies the plot display and options

You can specify the following *options* in the PROC VARIOGRAM statement.

DATA=*SAS-data-set*
specifies a SAS data set that contains the x and y coordinate variables and the VAR statement variables.

IDGLOBAL
specifies that ascending observation numbers be used across BY groups for the observation labels in the appropriate output data sets and the OBSERVATIONS plot, instead of resetting the observation number in the beginning of each BY group. The IDGLOBAL option is ignored if no BY variables are specified. Also, if you specify the ID statement, then the IDGLOBAL option is ignored unless you also specify the IDNUM option in the PROC VARIOGRAM statement.

IDNUM
specifies that the observation number be used for the observation labels in the appropriate output data sets and the OBSERVATIONS plot. The IDNUM option takes effect when you specify the ID statement; otherwise, it is ignored.

NOPRINT
suppresses the normal display of results. The NOPRINT option is useful when you want only to create one or more output data sets with the procedure.

NOTE: This option temporarily disables the Output Delivery System (ODS); see the section "ODS Graphics" on page 11046 for more information.

OUTACWEIGHTS=*SAS-data-set*
OUTACW=*SAS-data-set*
OUTA=*SAS-data-set*

specifies a SAS data set in which to store the autocorrelation weights information for each pair of points in the DATA= data set. Use this option with caution when the DATA= data set is large. If n denotes the number of observations in the DATA= data set, then the OUTACWEIGHTS= data set contains $[n(n-1)]/2$ observations.

See the section "OUTACWEIGHTS=*SAS-data-set*" on page 11040 for details.

OUTDISTANCE=*SAS-data-set*
OUTDIST=*SAS-data-set*
OUTD=*SAS-data-set*

specifies a SAS data set in which to store summary distance information. This data set contains a count of all pairs of data points within a given distance interval. The number of distance intervals is controlled by the NHCLASSES= option in the COMPUTE statement. The OUTDISTANCE= data set is useful for plotting modified histograms of the count data for determining appropriate lag distances. See the section "OUTDIST=*SAS-data-set*" on page 11041 for details.

OUTMORAN=*SAS-data-set*
OUTM=*SAS-data-set*

specifies a SAS data set in which to store information that is illustrated in the Moran plot, namely the standardized value of each observation in the DATA= data set and the weighted average of its local neighbors. You must also specify the LAGDISTANCE= and AUTOCORRELATION options in the COMPUTE statement; otherwise, the OUTMORAN= data set request is ignored.

The OUTMORAN= data set is useful when you want to save the information that is illustrated in the Moran scatter plot. The data set can also contain entries of missing observations with neighbors, although these observations are not displayed in the Moran plot. However, if the only observations with neighbors in your input data set are observations with missing values, then the OUTMORAN= output data set is empty.

See the section "OUTMORAN=*SAS-data-set*" on page 11041 for details.

OUTPAIR=*SAS-data-set*
OUTP=*SAS-data-set*

specifies a SAS data set in which to store distance and angle information for each pair of points in the DATA= data set.

Use this option with caution when your DATA= data set is large. Assume that your DATA= data set has n observations. When you specify the NOVARIOGRAM option in the COMPUTE statement, the OUTPAIR= data set is populated with all $[n(n-1)]/2$ pairs that can be formed with the n observations.

If the NOVARIOGRAM option is not specified, then the OUTPAIR= data set contains only pairs of data that are located within a certain distance away from each other. Specifically, it contains pairs whose distance between observations belongs to a lag class up to the specified MAXLAGS= option in the COMPUTE statement. Then, depending on your specification of the LAGDISTANCE= and MAXLAGS= options, the OUTPAIR= data set might contain $[n(n-1)]/2$ or fewer pairs.

Finally, you can restrict the number of pairs in the OUTPAIR= data set with the OUTPDISTANCE= option in the COMPUTE statement. The OUTPDISTANCE= option in the COMPUTE statement excludes pairs of points when the distance between the pairs exceeds the OUTPDISTANCE= value.

See the section "OUTPAIR=*SAS-data-set*" on page 11042 for details.

OUTVAR=*SAS-data-set*
OUTVR=*SAS-data-set*
specifies a SAS data set in which to store the continuity measures.

See the section "OUTVAR=*SAS-data-set*" on page 11043 for details.

PLOTS < (*global-plot-options*) > < = *plot-request*< (*options*) > >
PLOTS < (*global-plot-options*) > < = (*plot-request*< (*options*) > < . . . *plot-request*< (*options*) > >) >
controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request. Here are some examples:

```
plots=none
plots=observ
plots=(observ semivar)
plots(unpack)=semivar
plots=(semivar(cla unpack) semivar semivar(rob))
```

ODS Graphics must be enabled before plots can be requested. For example:

```
ods graphics on;

proc variogram data=sashelp.thick;
   compute novariogram;
   coordinates xc=East yc=North;
   var Thick;
run;

ods graphics off;
```

For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

If ODS Graphics is enabled but you omit the PLOTS option or have specified PLOTS=ALL, then PROC VARIOGRAM produces a default set of plots, which might be different for different COMPUTE statement options, as discussed in the following.

- If you specify NOVARIOGRAM in the COMPUTE statement, the VARIOGRAM procedure produces a scatter plot of your observations spatial distribution, in addition to the histogram of the pairwise distances of your data. For an example of the observations plot, see Figure 131.2. For an example of the pairwise distances plot, see Figure 131.4.
- If you omit NOVARIOGRAM in the COMPUTE statement, the VARIOGRAM procedure computes the empirical semivariogram for the specified LAGDISTANCE= and MAXLAGS= options. The observations plot appears by default in this case too. The VARIOGRAM procedure also produces a plot of the classical empirical semivariogram. If you also specify ROBUST in the COMPUTE statement, then the VARIOGRAM procedure instead produces a plot of both the classical and robust empirical semivariograms, in addition to the observations plot. For an example of the empirical semivariogram plot, see Output 131.7. Moreover, if you specify the MODEL statement and perform model fitting, then PROC VARIOGRAM also produces a fit plot of the fitted semivariogram. An example of the fit plot is shown in Figure 131.16.

The following *global-plot-options* are available:

ONLY
suppresses the default plots. Only plots that are specifically requested are displayed.

UNPACKPANEL
UNPACK
suppresses paneling. By default, multiple plots can appear in some output panels. Specify UNPACKPANEL to get each plot in a separate panel. You can specify PLOTS(UNPACKPANEL) to unpack the default plots. You can also specify UNPACKPANEL as a suboption with the SEMIVAR option.

The following individual *plot-requests* and plot *options* are available:

ALL
produces all appropriate plots. You can specify other *options* with ALL. For example, to request all default plots and an additional classical empirical semivariogram, specify PLOTS=(ALL SEMIVAR(CLA)).

EQUATE
specifies that all appropriate plots be produced in a way that the coordinates of the axes have equal size units.

FITPLOT < (*fitplot-options*) >
FIT < (*fitplot-options*) >
requests a plot that shows the model fitting results against the empirical semivariogram. By default, FITPLOT displays one plot of the fitted model (or a panel of plots for different angles in the anisotropic case).

If you specify the FORM=AUTO option in the MODEL statement, then each class of equivalent fitted models is displayed with a different curve on the plot. The best fitting model class is chosen based on the criteria that you specify in the CHOOSE option of the MODEL statement, and a thicker line on top of any other curve is shown for it. The plot legend shows the ranked classes by displaying the label of the representative model of each class in the plot. If appropriate, the number of additional models in the same equivalence class also shows within parentheses.

You can specify the following *fitplot-options*:

NCLASSES=*number*
NCLASSES=ALL
specifies the maximum number of classes to display on the fit plot, where *number* is a positive integer. The default is NCLASSES=5 for nonpaneled plots and NCLASSES=3 for paneled plots. The option takes effect when you specify the FORM=AUTO option in the MODEL statement, and it is ignored when you fit one single model. If you specify NCLASSES=ALL or a larger number than the available classes, then all available classes are shown on the fit plot. If you specify multiple instances of the NCLASSES= option, then only the last specified instance is honored.

UNPACK
suppresses paneling in paneled fit plots. By default, fit plots appear in a panel, when appropriate.

MORAN < (*moran-options*) >

MOR < (*moran-options*) >
produces a Moran scatter plot of the observations with nonmissing values. For more details about this plot, see the section "The Moran Scatter Plot" on page 11038. In addition to the Moran scatter plot points, the plot also displays the fit line for the linear regression of the weighted average on the standardized observation values, the regression fit line slope, and a reference line with slope equal to 1. The MORAN plot has the following *moran-options*:

LABEL < (*label-options*) >
labels the observations. The label is the ID variable if the ID statement is specified; otherwise, it is the observation number. The *label-options* can be one or more of the following:

HH
specifies that labels show for observations in the upper right (high-high) plot quadrant of positive spatial association.

HL
specifies that labels show for observations in the lower right (high-low) plot quadrant of negative spatial association.

LH
specifies that labels show for observations in the upper left (low-high) plot quadrant of negative spatial association.

LL
specifies that labels show for observations in the lower left (low-low) plot quadrant of positive spatial association.

If you specify multiple instances of the MORAN option and you specify the LABEL suboption in any of those, then the resulting Moran scatter plot displays the observations labels. By default, when you specify none of the *label-options*, the PLOTS=MORAN(LABEL) request puts labels in all observations.

ROWAVG=*rowavg-option*
specifies the flag value for row-averaging of weights in the computation of the weighted average. The *rowavg-option* can be either of the following:

OFF
specifies that autocorrelation weights not be row-averaged.

ON
specifies that row-averaged autocorrelation weights be used.

The default behavior is ROWAVG=ON. If you specify the ROWAVG= option more than once in the same MORAN plot request, then the behavior is set to ROWAVG=ON unless any of the instances is ROWAVG=OFF.

When you specify the PLOTS=MORAN option, you must specify both the AUTOCORRELATION and the LAGDISTANCE= options in the COMPUTE statement to produce the Moran scatter plot. For more information about the plot, see the section "The Moran Scatter Plot" on page 11038.

NONE
suppresses all plots.

OBSERVATIONS < (*observations-plot-options*) >
OBSERV < (*observations-plot-options*) >
OBS < (*observations-plot-options*) >
produces the observed data plot. Only one observations plot is created if you specify the OBSERVATIONS option more than once within a PLOTS option.

The OBSERVATIONS option has the following *suboptions*:

GRADIENT
specifies that observations be displayed as circles colored by the observed measurement.

LABEL < (*label-option*) >
labels the observations. The label is the ID variable if the ID statement is specified; otherwise, it is the observation number. The *label-option* can be one of the following:

EQ=*number*
specifies that labels show for any observation whose value is equal to the specified *number*.

MAX=*number*
specifies that labels show for observations with values smaller than or equal to the specified *number*.

MIN=*number*
specifies that labels show for observations with values equal to or greater than the specified *number*.

If you specify multiple instances of the OBSERVATIONS option and you specify the LABEL suboption in any of those, then the resulting observations plot displays the observations labels. If more than one *label-option* is specified in multiple LABEL suboptions, then the prevailing *label-option* in the resulting OBSERVATIONS plot emerges by adhering to the choosing order: MIN, MAX, EQ.

OUTLINE
specifies that observations be displayed as circles with a border but with a completely transparent fill.

OUTLINEGRADIENT
is the same as OBSERVATIONS(GRADIENT) except that a border is shown around each observation.

SHOWMISSING

specifies that observations with missing values be displayed in addition to the observations with nonmissing values. By default, missing values locations are not shown on the plot. If you specify multiple instances of the OBSERVATIONS option and you specify the SHOWMISSING suboption in any of those, then the resulting observations plot displays the observations with missing values.

If you omit any of the GRADIENT, OUTLINE, and OUTLINEGRADIENT suboptions, the OUTLINEGRADIENT is the default suboption. If you specify multiple instances of the OBSERVATIONS option or multiple suboptions for OBSERVATIONS, then the resulting observations plot honors the last specified GRADIENT, OUTLINE, or OUTLINEGRADIENT suboption.

PAIRS < (*pairs-plot-options*) >

specifies that the pairwise distances histogram be produced. By default, the horizontal axis displays the lag class number. The vertical axis shows the frequency (count) of pairs in the lag classes. Notice that the zero lag class width is half the width of the other classes.

The PAIRS option has the following *suboptions*:

MIDPOINT

MID

specifies that the plot that is created with the PAIRS option display the lag class midpoint value on the horizontal axis, rather than the default lag class number. The midpoint value is the actual distance of a lag class center from the assumed origin point at distance zero. See also the illustration in Figure 131.22.

NOINSET

NOI

specifies that the plot created with the PAIRS option be produced without the default inset that provides additional information about the pairs distribution.

THRESHOLD=*minimum pairs*

THR=*minimum pairs*

specifies that a reference line appear in the plot that is created with the PAIRS option to indicate the *minimum pairs* frequency of data pairs. You can use this line as an exploratory tool when you want to select lag classes that contain at least THRESHOLD point pairs. The option helps you to identify visually any portion of the PAIRS distribution that lies below the specified THRESHOLD value.

Only one pairwise distances histogram is created if you specify the PAIRS option within a PLOTS option. If you specify multiple instances of the PAIRS option, the resulting plot has the following features:

- If the MIDPOINT or NOINSET suboption has been specified in any of the instances, it is activated in the resulting plot.
- If you have specified the THRESHOLD= suboption more than once, then the THRESHOLD= value specified last prevails.

SEMIVARIOGRAM < (*semivar-plot-options*) >

SEMIVAR < (*semivar-plot-options*) >

specifies that the empirical semivariogram plot be produced. You can specify the SEMIVAR option multiple times in the same PLOTS option to request instances of plots with the following *semivar-plot-options*:

ALL | CLASSICAL | ROBUST

ALL | CLA | ROB

specifies a single type of empirical semivariogram (classical or robust) to plot, or specifies that all the available types be included in the same plot. The default is ALL.

UNPACKPANEL

UNPACK

specifies that paneled semivariogram plots be displayed separately. By default, plots appear in a panel, when appropriate.

BY Statement

BY *variables* **;**

You can specify a BY statement in PROC VARIOGRAM to obtain separate analyses of observations in groups that are defined by the BY variables. When a BY statement appears, the procedure expects the input data set to be sorted in order of the BY variables. If you specify more than one BY statement, only the last one specified is used.

If your input data set is not sorted in ascending order, use one of the following alternatives:

- Sort the data by using the SORT procedure with a similar BY statement.
- Specify the NOTSORTED or DESCENDING option in the BY statement in the VARIOGRAM procedure. The NOTSORTED option does not mean that the data are unsorted but rather that the data are arranged in groups (according to values of the BY variables) and that these groups are not necessarily in alphabetical or increasing numeric order.
- Create an index on the BY variables by using the DATASETS procedure (in Base SAS software).

For more information about BY-group processing, see the "Grouping Data" section of *SAS Programmers Guide: Essentials*. For more information about the DATASETS procedure, see the discussion in the *Base SAS Procedures Guide*.

COMPUTE Statement

COMPUTE *computation-options* **;**

The COMPUTE statement provides a number of options that control the computation of the semivariance, the robust semivariance, and the covariance.

Table 131.2 summarizes the *options* available in the COMPUTE statement.

Table 131.2 COMPUTE Statement Options

Option	Description
ALPHA=	Specifies the confidence level
ANGLETOLERANCE=	Specifies the tolerance
AUTOCORRELATION	Calculates autocorrelation statistics
BANDWIDTH=	Specifies the bandwidth
CL	Requests confidence limits
DEPSILON=	Specifies the distance value for declaring that two distinct points are zero distance apart
LAGDISTANCE=	Specifies the basic distance unit that defines the lags
LAGTOLERANCE=	Specifies the tolerance around the LAGDISTANCE= value
MAXLAGS=	Specifies the maximum number of lag classes to be used
NDIRECTIONS=	Specifies the number of angle classes to be used
NHCLASSES=	Specifies the number of distance classes to be used
NOVARIOGRAM	Prevents the computation of the continuity measures
OUTPDISTANCE=	Specifies the cutoff distance for writing observations to the OUTPAIR= data set
ROBUST	Calculates a robust version of the semivariance

ALPHA=*number*

specifies a parameter to obtain the confidence level for constructing confidence limits in the classical empirical semivariance estimation. The value of *number* must be in $(0, 1)$, and the confidence level is $1-$*number*. The default is ALPHA=0.05, which corresponds to the default confidence level of 95%. If the CL option is not specified, ALPHA= is ignored.

ANGLETOLERANCE=*angle-tolerance*

ANGLETOL=*angle-tolerance*

ATOL=*angle-tolerance*

specifies the tolerance, in degrees, around the angles determined by the NDIRECTIONS= specification. The default is $180°/(2n_d)$, where n_d is the NDIRECTIONS= specification. If you do not specify the NDIRECTIONS= option or the DIRECTIONS statement, ANGLETOLERANCE= is ignored.

See the section "Theoretical and Computational Details of the Semivariogram" on page 11013 for further information.

AUTOCORRELATION < (*autocorrelation-options*) >

AUTOCORR < (*autocorrelation-options*) >

AUTOC < (*autocorrelation-options*) >

specifies that autocorrelation statistics be calculated. You can further specify the following *autocorrelation-options* in parentheses following the AUTOCORRELATION option.

ASSUMPTION < = *assumption-options* >

ASSUM < = *assumption-options* >

specifies the type of autocorrelation assumption to use. The *assumption-options* can be one of the following:

NORMALITY | NORMAL | NOR

specifies use of the normality assumption.

RANDOMIZATION | RANDOM | RAN

specifies use of the randomization assumption.

The default is ASSUMPTION=NORMALITY.

STATISTICS < = (*stats-options*) >

STATS < = (*stats-options*) >

specifies the autocorrelation statistics in detail. The *stats-options* can be one or more of the following:

ALL

applies all available types of autoregression statistics.

GEARY | GEA

specifies use of the Geary's c statistics.

MORAN | MOR

specifies use of the Moran's I statistics.

The default is STATISTICS=ALL.

WEIGHTS < = *weights-options* >

WEI < = *weights-options* >

specifies the scheme used for the computation of the autocorrelation weights. You can choose one of the following *weights-options*:

BINARY < (*binary-option*) >

specifies that binary weights be used. You also have the following *binary-option*:

ROWAVERAGING | ROWAVG | ROW

specifies that asymmetric autocorrelation weights be assigned to data pairs. For each observation, if there are nonzero weights, the ROWAVG option standardizes those weights so that they sum to 1. No row averaging is performed by default.

DISTANCE <(*distance-options*)>
specifies that autocorrelation weights be assigned based on the point pair distances. You also have the following *distance-options*:

NORMALIZE | NORMAL | NOR
specifies that normalized pair distances be used in the distance-based weights expression. The distances are normalized with respect to the maximum pairwise distance h_b, as it is defined in the section "Computation of the Distribution Distance Classes" on page 11021. By default, nonnormalized values are used in the computations.

POWER=*number*
POW=*number*
specifies the power to which the pair distance is raised in the distance-based weights expression. POWER is a nonnegative number, and its default value is POWER=1.

ROWAVERAGING | ROWAVG | ROW
specifies that asymmetric autocorrelation weights be assigned to data pairs. For each observation, if there are nonzero weights, the ROWAVG option standardizes those weights so that they sum to 1. No row averaging is performed by default.

SCALE=*number*
SCA=*number*
specifies the scaling factor in the distance-based weights expression. SCALE is a nonnegative number, and its default value is SCALE=1.

The default is WEIGHTS=BINARY. See the section "Autocorrelation Statistics" on page 11035 for further details about the autocorrelation weights.

When you specify the AUTOCORRELATION option with no *autocorrelation-options*, PROC VARIOGRAM computes by default both the Moran's I and Geary's c statistics with p-values computed under the normality assumption with binary weights.

If you specify more than one ASSUMPTION in the *autocorrelation-options*, all but the last specified ASSUMPTION are ignored. The same holds if you specify more than one POWER= or SCALE= parameter in the WEIGHT=DISTANCE *distance-options*.

If you specify the WEIGHT=BINARY option in the AUTOCORRELATION option and the NOVARIOGRAM option at the same time, then you must also specify the LAGDISTANCE= option in the COMPUTE statement. See the section "Autocorrelation Weights" on page 11035 for more information.

BANDWIDTH=*bandwidth-distance*
BANDW=*bandwidth-distance*
specifies the bandwidth, or perpendicular distance cutoff for determining the angle class for a given pair of points. The distance classes define a series of cylindrically shaped areas, while the angle classes radially cut these cylindrically shaped areas. For a given angle class $(\theta_1 - \delta\theta_1, \theta_1 + \delta\theta_1)$, as you proceed out radially, the area encompassed by this angle class becomes larger. The BANDWIDTH= option restricts this area by excluding all points with a perpendicular distance from the line $\theta = \theta_1$ that is greater than the BANDWIDTH= value. See Figure 131.23 for a visual representation of the bandwidth.

If you omit the BANDWIDTH= option, no restriction occurs. If you omit the NDIRECTIONS= option or the DIRECTIONS statement, BANDWIDTH= is ignored.

CL

requests confidence limits for the classical semivariance estimate. The lower bound of the confidence limits is always nonnegative, adhering to the behavior of the theoretical semivariance. You can control the confidence level with the ALPHA= option.

DEPSILON=*distance-value*

DEPS=*distance-value*

specifies the distance value for declaring that two distinct points are zero distance apart. Such pairs, if they occur, cause numeric problems. If you specify DEPSILON=$\Delta\varepsilon$, then pairs of points P_1 and P_2 for which the distance between them $\mid P_1 P_2 \mid < \Delta\varepsilon$ are excluded from the continuity measure calculations. The default value of the DEPSILON= option is 100 times the machine precision; this product is approximately 1E–10 on most computers.

LAGDISTANCE=*distance-unit*

LAGDIST=*distance-unit*

LAGD=*distance-unit*

specifies the basic distance unit that defines the lags. For example, a specification of LAGDISTANCE=x results in lag distance classes that are multiples of x. For a given pair of points P_1 and P_2, the distance between them, denoted $\mid P_1 P_2 \mid$, is calculated. If $\mid P_1 P_2 \mid = x$, then this pair is in the first lag class. If $\mid P_1 P_2 \mid = 2x$, then this pair is in the second lag class, and so on.

For irregularly spaced data, the pairwise distances are unlikely to fall exactly on multiples of the LAGDISTANCE= value. In this case, a distance tolerance of δx accommodates a spread of distances around multiples of x (the LAGTOLERANCE= option specifies the distance tolerance). For example, if $\mid P_1 P_2 \mid$ is within $x \pm \delta x$, you would place this pair in the first lag class; if $\mid P_1 P_2 \mid$ is within $2x \pm \delta x$, you would place this pair in the second lag class; and so on.

You can experiment and determine the candidate values for the LAGDISTANCE= option by plotting the pairwise distance histogram for different numbers of histogram classes, using the NHCLASSES= option.

A LAGDISTANCE= value is required for the semivariance and the autocorrelation computations. However, when you specify the NOVARIOGRAM option without the AUTOCORRELATION option, you need not specify the LAGDISTANCE= option.

See the section "Theoretical and Computational Details of the Semivariogram" on page 11013 for more information.

LAGTOLERANCE=*tolerance-number*

LAGTOL=*tolerance-number*

LAGT=*tolerance-number*

specifies the tolerance around the LAGDISTANCE= value for grouping distance pairs into lag classes. See the description of the LAGDISTANCE= option for information about the use of the LAGTOLERANCE= option, and the section "Theoretical and Computational Details of the Semivariogram" on page 11013 for more details.

If you omit the LAGTOLERANCE= option, a default value of $\frac{1}{2}$ times the LAGDISTANCE= value is used.

MAXLAGS=*number-of-lags*

MAXLAG=*number-of-lags*

MAXL=*number-of-lags*

specifies the maximum number of lag classes to be used in constructing the continuity measures in addition to a zero lag class; see also the section "Distance Classification" on page 11019. This option excludes any pair of points P_1 and P_2 for which the distance between them, $| P_1 P_2 |$, exceeds the MAXLAGS= value times the LAGDISTANCE= value.

You can determine candidate values for the MAXLAGS= option by plotting or displaying the OUTDISTANCE= data set.

A MAXLAGS= value is required unless you specify the NOVARIOGRAM option.

NDIRECTIONS=*number-of-directions*

NDIR=*number-of-directions*

ND=*number-of-directions*

specifies the number of angle classes to use in computing the continuity measures. This option is useful when there is potential anisotropy in the spatial continuity measures. Anisotropy is a field property in which the characterization of spatial continuity depends on the data pair orientation (or angle between the N–S direction and the axis defined by the data pair). Isotropy is the absence of this effect; that is, the description of spatial continuity depends only on the distance between the points, not the angle.

The angle classes formed from the NDIRECTIONS= option start from N–S and proceed clockwise. For example, NDIRECTIONS=3 produces three angle classes. In terms of compass points, these classes are centered at 0° (or its reciprocal, 180°), 60° (or its reciprocal, 240°), and 120° (or its reciprocal, 300°). For irregularly spaced data, the angles between pairs are unlikely to fall exactly in these directions, so an angle tolerance of $\delta\theta$ is used (the ANGLETOLERANCE= option specifies the angle tolerance). If NDIRECTIONS=n_d, the base angle is $\theta = 180°/n_d$, and the angle classes are

$$(k\theta - \delta\theta, k\theta + \delta\theta) \quad k = 0, \ldots, n_d - 1$$

If you omit the NDIRECTIONS= option, no angles are formed. This is the omnidirectional case where the spatial continuity measures are assumed to be isotropic.

The NDIRECTIONS= option is useful for exploring possible anisotropy. The DIRECTIONS statement, described in the section "DIRECTIONS Statement" on page 10990, provides greater control over the angle classes.

See the section "Theoretical and Computational Details of the Semivariogram" on page 11013 for more information.

NHCLASSES=*number-of-histogram-classes*

NHCLASS=*number-of-histogram-classes*

NHC=*number-of-histogram-classes*

specifies the number of distance classes to consider in the spatial domain in the exploratory stage of the empirical semivariogram computation. The actual number of classes is one more than the NHCLASSES= value, since a special lag zero class is also computed. The NHCLASSES= option is used to produce the distance intervals table, the histogram of pairwise distances, and the OUTDISTANCE= data set. See the OUTDISTANCE= option, the section "OUTDIST=*SAS-data-set*"

on page 11041, and the section "Theoretical and Computational Details of the Semivariogram" on page 11013 for more information.

The default value is NHCLASSES=10.

NOVARIOGRAM
prevents the computation of the continuity measures. This option is useful for preliminary analysis, or when you require only the OUTDISTANCE= or OUTPAIR= data sets.

OUTPDISTANCE=*distance-limit*

OUTPDIST=*distance-limit*

OUTPD=*distance-limit*
specifies the cutoff distance for writing observations to the OUTPAIR= data set. If you specify OUTPDISTANCE=d_{max}, the distance $\mid P_1 P_2 \mid$ between each pair of points P_1 and P_2 is checked against d_{max}. If $\mid P_1 P_2 \mid > d_{max}$, the observation for this pair is not written to the OUTPAIR= data set. If you omit the OUTPDISTANCE= option, all distinct pairs are written. This option is ignored if you omit the OUTPAIR= data set.

ROBUST
requests that a robust version of the semivariance be calculated in addition to the classical semivariance.

COORDINATES Statement

COORDINATES *coordinate-variables* **;**

The following two options give the names of the variables in the DATA= data set that contains the values of the *x* and *y* coordinates of the data.

Only one COORDINATES statement is allowed, and it is applied to all the analysis variables. In other words, it is assumed that all the VAR variables have the same *x* and *y* coordinates.

XCOORD=(*variable-name***)**

XC=(*variable-name***)**

X=(*variable-name***)**
gives the name of the variable that contains the *x* coordinate of the data in the DATA= data set.

YCOORD=(*variable-name***)**

YC=(*variable-name***)**

Y=(*variable-name***)**
gives the name of the variable that contains the *y* coordinate of the data in the DATA= data set.

DIRECTIONS Statement

DIRECTIONS *directions-list* **;**

You use the DIRECTIONS statement to define angle classes. You can specify angle classes as a list of angles, separated by commas, with optional angle tolerances and bandwidths within parentheses following the angle. You must specify at least one angle.

If you do not specify the optional angle tolerance, the default value of 45° is used. If you do not specify the optional bandwidth, no bandwidth is checked. If you specify a bandwidth, you must also specify an angle tolerance.

For example, suppose you want to compute three separate semivariograms at angles $\theta_1 = 0°$, $\theta_2 = 60°$, and $\theta_3 = 120°$, with corresponding angle tolerances $\delta\theta_1 = 22.5°$, $\delta\theta_2 = 12.5°$, and $\delta\theta_3 = 22.5°$, with bandwidths 50 and 40 distance units on the first two angle classes and no bandwidth check on the last angle class.

The appropriate DIRECTIONS statement is as follows:

```
directions 0.0(22.5,50), 60.0(12.5,40),120(22.5);
```

ID Statement

ID *variable* **;**

The ID statement specifies which variable to include for identification of the observations in the OUTPAIR= and the OUTACWEIGHTS= output data sets. The ID statement variable is also used for the labels and tool tips in the OBSERVATIONS plot.

In the VARIOGRAM procedure you can specify only one ID variable in the ID statement. If no ID statement is given, then PROC VARIOGRAM uses the observation number in the data sets and the OBSERVATIONS plot.

MODEL Statement

MODEL *fitting-options* < / *model-options* > **;**

You specify the MODEL statement if you want to fit a theoretical semivariogram model to the empirical semivariogram data that are produced in the COMPUTE statement. You must have nonmissing empirical semivariogram estimates at a minimum of three lags to perform model fitting.

Table 131.3 summarizes the *options* available in the MODEL statement.

Table 131.3 MODEL Statement Options

Option	Description
Fitting Options	
ALPHA=	Specifies confidence level
CHOOSE=	Ranks the fitted models and chooses the optimally fit model

Table 131.3 *continued*

Option	Description
CL	Constructs a *t*-type confidence interval
EQUIVTOL=	Specifies a positive upper value tolerance
FIT=	Specifies which type of empirical semivariogram to fit
FORM=	Specifies the functional form (type) of the semivariogram model
MDATA=	Specifies the input data set
METHOD=	Fits a theoretical model to the empirical semivariance
NEPSILON=	Adds a minimal nugget effect
NUGGET=	Specifies the nugget effect
RANGE=	Specifies the range parameter
RANGELAG=	Uses consecutive nonmissing empirical semivariance lags to fit model
RANKEPS=	Specifies the minimum threshold to compare fit quality
SCALE=	Specifies the scale parameter in semivariogram models
SMOOTH=	Specifies the positive smoothness parameter ν
Model Options	
COVB	Requests the covariance matrix
CORRB	Requests the approximate correlation matrix
DETAILS	Produces different levels of output
GRADIENT	Displays the gradient of the objective function
MTOGTOL=	Specifies a threshold value for the smoothness parameter
NOFIT	Suppresses the model fitting process
NOITPRINT	Suppresses the display of the iteration history table

You can choose to perform a fully automated fitting or to fit one model with specific forms. In the first case you simply specify a list of forms or no forms at all. All suitable combinations are tested, and the result is the model that produces the best fit according to specified criteria. In the second case you specify one theoretical semivariogram model, and you have more control over its parameters for the fitting process.

Furthermore, you can specify a theoretical semivariogram model in two ways:

- You explicitly specify the FORM option and any of the options SCALE, RANGE, and NUGGET in the MODEL statement.
- You can specify an MDATA= data set. This data set contains variables that correspond to the FORM option and to any of the options SCALE, RANGE, NUGGET, and SMOOTH. You can also use an MDATA= data set to request a fully automated fitting.

The two methods are exclusive; either you specify all parameters explicitly, or they all are read from the MDATA= data set.

The MODEL statement has the following *fitting-options*:

ALPHA=*number*

requests that a t-type confidence interval be constructed for each of the fitting parameters with confidence level $1 - \mathit{number}$. The value of *number* must be in $(0, 1)$; the default is 0.05 which corresponds to the default confidence level of 95%. If the CL option of the MODEL statement is not specified, then ALPHA= is ignored.

CHOOSE=*criterion* | (*criterion1 . . . criterionk*)

specifies that if the fitting task has more than one model to fit, then PROC VARIOGRAM ranks the fitted models and chooses the optimally fit model according to one or more available criteria.

If you want to use multiple fitting criteria, then the order in which you specify them in the CHOOSE= option defines how they are applied. This feature is useful when fitting suggests that two or more models perform equally well according to a certain criterion. For example, if two models are equivalent according to *criterioni*, then they are further ranked in the list based on the next criterion, *criterionj*, where $j = i + 1$.

Each *criterion* can be one of the following:

AIC

specifies Akaike's information criterion.

SSE

specifies the weighted sum of squares error for each fitted model when METHOD=WLS, and the residual sum of squares error for each fitted model when METHOD=OLS.

STATUS

classifies models based on their fitting process convergence status. CHOOSE=STATUS places on top models for which the fitting process is successful.

By default, the models are ranked in the fit summary table with the best fitted model at the top of the list, based on the criteria that you specify in the CHOOSE= option. This model is the fit choice of PROC VARIOGRAM for the particular fitting task. If you omit the CHOOSE= option, then the default behavior is CHOOSE=(SSE AIC).

Regardless of the specified fitting criteria, models for which the fitting process is unsuccessful always appear at the bottom of the fit summary table. For more details about the fitting criteria, see the section "Fitting Criteria" on page 11032. After multiple models are ranked, they are further categorized in classes of equivalence depending on whether any two models calculate the same semivariance value at the same distance for a series of different distances. For more details, see the section "Classes of Equivalence" on page 11034.

If you specify the same criterion multiple times in the CHOOSE= option, then only the first instance is used for the ranking process and any additional ones are ignored. If you specify only one model to fit in the MODEL statement and you specify the CHOOSE= option, then the option is ignored.

CL

requests that t-type confidence limits be constructed for each of the fitting parameters estimates. The confidence level is 0.95 by default; this can be changed with the ALPHA= option of the MODEL statement.

EQUIVTOL=*etol-value*

ETOL=*etol-value*

specifies a positive upper value tolerance to use when categorizing multiple models in classes of equivalence. For this categorization, the VARIOGRAM procedure computes the sum of absolute differences of semivariances for pairs of consecutively ranked models. If the sum is lower than the EQUIVTOL= value for any such model pair, then these two models are deemed to be equivalent. As a result, the EQUIVTOL= option can affect the number and size of classes of equivalence in the fit summary table. Smaller values of the EQUIVTOL= parameter result in a more strict model comparison and can lead to a higher number of classes of equivalence. For more details, see the section "Classes of Equivalence" on page 11034.

The default value for the EQUIVTOL= parameter is 10^{-3}. The EQUIVTOL= option applies when you fit multiple models with the FORM=AUTO option of the MODEL statement; otherwise, it is ignored.

The EQUIVTOL= option is independent of the ranking results from the RANKEPS= option of the MODEL statement. This means that you could possibly have models listed but not ranked in the fit summary table, and still have equivalence classes assigned according to the order in which the models appear in the table.

FIT=*fit-type-options*

specifies which type of empirical semivariogram to fit. You can choose between the following *fit-type-options*:

CLASSICAL

CLA

fits a model for the classical empirical semivariance.

ROBUST

ROB

fits a model for the robust empirical semivariance. This option can be used only when the ROBUST option is specified in the COMPUTE statement.

The default value is FIT=CLASSICAL.

FORM=*form* | (*form1, . . . , formk*) | **AUTO** (*auto-options*)

specifies the functional form (type) of the semivariogram model. The supported structures are two-parameter models that use the sill and range as parameters. The Matérn model is an exception that makes use of a third smoothing parameter ν.

The FORM= option is required when you specify the MODEL statement. You can perform fitting of a theoretical semivariogram model either explicitly or in an automated manner. For the explicit specification you specify suitable model forms in the FORM= option. For an automated fit you specify the FORM=AUTO option which has the AUTO(MLIST=) and AUTO(NEST=) suboptions. You can read more details in the following two subsections.

Explicit Model Specification

You can explicitly specify a theoretical semivariogram model to fit by using any combination of one, two, or three forms. Use the syntax with the single *form* to specify a non-nested model. Use the syntax with k structures *formi*, $i = 1, \ldots, k$, to specify up to three nested structures ($k \leq 3$) in a semivariogram model. Each of the forms can be any of the following:

CUBIC | EXPONENTIAL | GAUSSIAN | MATERN |
PENTASPHERICAL | POWER | SINEHOLEEFFECT | SPHERICAL
CUB | EXP | GAU | MAT | PEN | POW | SHE | SPH
specify a form.

All of these forms are presented in more detail in the section "Theoretical Semivariogram Models" on page 11008. In addition, you can optionally specify a nugget effect for your model with the NUGGET option in the MODEL statement.

For example, the syntax

```
FORM=GAU
```

specifies a model with a single Gaussian structure. Also, the syntax

```
FORM=(EXP,SHE,MAT)
```

specifies a nested model with an exponential, a sine hole effect, and a Matérn structure. Finally

```
FORM=(EXP,EXP)
```

specifies a nested model with two structures both of which are exponential.

NOTE: In the documentation, models are named either by using their full names or by using the first three letters of their structures. Also, the names of different structures in a nested model are separated by a hyphen (-). According to this convention, the previous examples illustrate how to specify a GAU, an EXP-SHE-MAT, and an EXP-EXP model, respectively, with the FORM= option.

When you explicitly specify the types of structures, you can fix parameter values or ask PROC VARIOGRAM to select default initial values for the forms parameters by using the SCALE, RANGE, NUGGET, and SMOOTH options. You can set your own, non-default initial parameter values by using the PARMS statement in combination with an explicitly specified semivariogram model in the MODEL statement.

Automated Model Selection

Use the FORM=AUTO option to request the highest level of automation in the best fit selection of the parameters. If you specify FORM=AUTO, any of the SCALE, RANGE, or SMOOTH options that are also specified are ignored. When you specify the FORM=AUTO option, you cannot specify the PARMS statement for the corresponding MODEL statement. As a result, when you use the FORM=AUTO option, you cannot fix any of the model parameters and PROC VARIOGRAM sets initial values for them.

The AUTO option has the following *auto-options*:

MLIST=*mform* | (*mform1, . . . , mformp*)

specifies one or more different model forms to use in combinations during the model fitting process. If you omit the MLIST= suboption, then combinations are made among all available model types. The *mform* can be any of the following eight forms:

CUBIC | EXPONENTIAL | GAUSSIAN | MATERN |
PENTASPHERICAL | POWER | SINEHOLEEFFECT | SPHERICAL
CUB | EXP | GAU | MAT | PEN | POW | SHE | SPH
specify a form.

If you use more than one *mform*, then each *mformi*, $i = 1, \ldots, p$ must be different from the others in the group of $p \leq 8$ forms that you specify.

NEST=*nest-list*

specifies the number of nested structures to use for the fitting. You can choose between the following to specify the *nest-list*:

n	a single value
m TO *n*	a sequence in which *m* equals the starting value and *n* equals the ending value

For example,

```
NEST=1
```

produces the best fit with one single model among all model types specified in the MLIST= suboption. Also,

```
NEST=2 TO 3
```

produces the best fit among all combinations of the model types specified in the MLIST= suboption that result in nested models with two or three structures. The combinations that are tested include repetitions. Hence, if you specify, for example,

```
MODEL FORM=AUTO(MLIST=(EXP,SPH) NEST=1 TO 2)
```

then the different models that are tested are equivalent to the specifications FORM=EXP, FORM=SPH, FORM=(EXP,EXP), FORM=(EXP,SPH), FORM=(SPH,SPH) and FORM=(SPH,EXP). **NOTE:** The models EXP-SPH and SPH-EXP are taken as two separate models. Although they are mathematically equivalent (see the section "Nested Models" on page 11013), PROC VARIOGRAM assigns different initial values to the model structures in each case, which can lead to different fitting results. (See the section "Example 131.1: Aspects of Semivariogram Model Fitting" on page 11047.)

If you omit the NEST suboption, then by default PROC VARIOGRAM searches for the best fit with up to three nested structures in a model. The default behavior is equivalent to

```
NEST=1 TO 3
```

In the VARIOGRAM procedure you can use a maximum of three nested structures to fit an empirical semivariogram; that is, $n \leq 3$.

You can use the AUTO value for the form in the MDATA= data set, and also in the FORM= option. However, in the former case the automation functionality is limited compared to the latter case and the *auto-options* of the FORM=AUTO option. In particular, when you specify the form to be AUTO in the MDATA= data set, then PROC VARIOGRAM follows only the default behavior and searches among all available forms for the best fit with up to three nested structures in a model.

MDATA=*SAS-data-set*

specifies the input data set that contains parameter values for the covariance or semivariogram model. The MDATA= data set must contain a variable named FORM, and it can optionally include any of the variables SCALE, RANGE, NUGGET, and SMOOTH.

The FORM variable must be a character variable. It accepts only the AUTO value or the *form* values that can be specified in the FORM= option in the MODEL statement. The RANGE, SCALE, NUGGET, and SMOOTH variables must be numeric or missing.

The number of observations present in the MDATA= data set corresponds to the level of nesting of the semivariogram model. Each observation line describes a structure of the model you submit for fitting.

If you specify the AUTO value for the FORM variable in an observation, then you cannot specify additional nested structures in the same data set, and any parameters you specify in the same structure are ignored. In that case, PROC VARIOGRAM performs a crude automated search among all available forms to obtain the best fit with up to three nested structures in a model. You can refine this type of search with additional suboptions when you perform it with the FORM=AUTO option instead of the MDATA= option in the MODEL statement.

When you have a nested model, you might want to specify parameter values for only some of the nested structures. In this case, you must specify the corresponding parameter values for the remaining model structures as missing values.

For example, you can use the following DATA step to specify a non-nested model that uses a spherical covariance within an MDATA= data set:

```
data md1;
   input scale range form $;
   datalines;
25 10 SPH
;
```

Then, you can use the md1 data in the MODEL statement of PROC VARIOGRAM as shown in the following statements:

```
proc variogram data=...;
   compute ...;
   model mdata=md1;
run;
```

This is equivalent to the following explicit specification of the semivariance model parameters:

```
proc variogram data=...;
   compute ...;
   model form=sph scale=25 range=10;
run;
```

The following data set md2 is an example of a nested model:

```
data md2;
   input form $ scale range nugget smooth;
   datalines;
SPH 20 8 5 .
MAT 12 3 5 0.7
GAU .  1 5 .
;
```

This specification is equivalent to the following explicit specification of the semivariance model parameters:

```
proc variogram data=...;
   compute ....;
   model form=(sph,mat,gau)
         scale=(20,12,.) range=(8,3,1) smooth=0.7 nugget=5;
run;
```

Use the SMOOTH variable column in the MDATA= data set to specify the smoothing parameter ν in the Matérn semivariogram models. The SMOOTH variable values must be positive and no greater than 1,000,000. PROC VARIOGRAM sets this upper limit for numerical and performance reasons. In any case, if the fitting process leads the smoothness value to exceed the default threshold value 10,000, then the VARIOGRAM procedure converts the Matérn form into a Gaussian form and repeats the model fitting. To adjust the switching threshold value, you can use the MTOGTOL= option in the MODEL statement.

If you specify a SMOOTH column in the MDATA= data set, then its elements are ignored except for the rows in which the corresponding FORM is Matérn.

The NUGGET variable value is the same for all nested structures. This is the way to specify a nugget effect in the MDATA= data set. If you specify more than one nugget value for different structures, then the last nugget value specified is used.

METHOD=*method-options*

must be specified in the MODEL statement to fit a theoretical model to the empirical semivariance. The METHOD option has the following *suboptions*:

OLS

specifies that ordinary least squares be used for the fitting.

WLS
specifies that weighted least squares be used for the fitting.

The default is METHOD=WLS.

NEPSILON=*min-nugget-factor*
NEPS=*min-nugget-factor*
specifies that a minimal nugget effect be added to the theoretical semivariance in the unlikely occasion that the theoretical semivariance becomes zero during fitting with weighted least squares. As explained in the section "Theoretical and Computational Details of the Semivariogram" on page 11013, the theoretical semivariance is always positive for any distance larger than zero. If a conflicting situation emerges as a result of numerical fitting issues, then the NEPSILON= option can help you alleviate the problem by adding a minimal variance at the distance lag where the issue is encountered. For more details, see the section "Parameter Initialization" on page 11029.

If you omit the NEPSILON= option, then PROC VARIOGRAM sets a default value of 10^{-6}. If a minimal nugget effect is used, its value is case-specific and is based on the *min-nugget-factor*. Specifically, its value is defined as *min-nugget-factor* times the sample variance of the input data set, or as *min-nugget-factor* when the sample variance is equal to zero.

NUGGET=*number*
specifies the nugget effect for the model. The nugget effect is due to a discontinuity in the semivariogram as determined by plotting the sample semivariogram; see "Theoretical Semivariogram Models" on page 11008 for more details. The NUGGET= parameter is a nonnegative number. If you specify a nonmissing value, then it is used as a fixed parameter in the fitting process.

PROC VARIOGRAM assigns a default initial value for the nugget effect in the following cases:

- if you specify a missing value.
- if you omit the NUGGET= option and you do not specify an associated PARMS statement with initial values for the nugget.

The NUGGET= option is incompatible with the specification of the PARMS statement for the corresponding MODEL statement.

RANGE=*range* | **(***range1, . . . , rangek***)**
specifies the range parameter in semivariogram models. The RANGE= option is optional. However, if you specify the RANGE= option, then you must provide range values for all structures that you have specified explicitly in the FORM= option. All nonmissing range values are considered as fixed parameters. PROC VARIOGRAM assigns a default initial value to any of the model structures for which you specify a missing range value. PROC VARIOGRAM assigns default initial values to all model structures if you omit the RANGE= option, unless you specify an associated PARMS statement and initial values for the range in it.

The range parameter is a positive number, has the units of distance, and is related to the correlation scale of the underlying spatial process.

NOTE: If you specify this parameter for a power model, then it does not correspond to a range. For power models, the parameter you specify in the RANGE option is a dimensionless power exponent whose value must range within [0,2) so that the power model is a valid semivariance function.

The RANGE= option is ignored when you specify the FORM=AUTO option. The RANGE= option is incompatible with the specification of the PARMS statement for the corresponding MODEL statement.

RANGELAG=*rlag-list*

RLAG=*rlag-list*

specifies that you prefer to use the range of consecutive nonmissing empirical semivariance lags in the *rlag-list* for the semivariogram fitting process, instead of using all MAXLAGS+1 lag classes by default. You can specify *rlag-list* in either of the following forms:

k	a single value that designates the width of the selected lag range by starting at lag zero. You must use at least three lags to perform model fitting, so you can specify k within [3, . . . , MAXLAGS+1].
m TO n	a sequence in which m equals the starting lag and n equals the ending lag. The parameters m and n must be nonnegative integer numbers to designate lag classes between zero and MAXLAGS. Use at least three lags for model fitting; hence it holds that $n - m \geq 2$.

The following two brief examples exhibit the use of the RANGELAG option. These examples assume that you have set the MAXLAGS= option to 9 or higher to indicate nonmissing empirical semivariance estimates at 10 lags or more.

In the first example,

```
RANGELAG=8
```

uses the empirical semivariance in the first eight lags to fit a theoretical model. Hence, RANGELAG=8 uses only the lag classes zero to seven. This approach enables you to account only for the correlation behavior described by the first k empirical semivariogram lag classes.

In the second example,

```
RANGELAG=2 TO 9
```

specifies that the empirical semivariance values at lag classes zero, one, and after lag class nine are excluded from the model fitting process.

RANKEPS=*reps-value*

REPS=*reps-value*

specifies the minimum threshold to compare fit quality of two models for a specific criterion. Beyond this threshold the criterion values become insensitive to comparison. In particular, when you fit multiple models, PROC VARIOGRAM computes for each one the value of the fitting criterion specified in the CHOOSE= option of the MODEL statement. These values are examined in pairs at the sorting stage. If the difference of a given pair exceeds the *reps-value*, then the sort order of the corresponding models is reversed; otherwise, the two models retain their relative order in the rankings. Hence, the RANKEPS= option can affect model ranking in the fit summary table.

The default value for the RANKEPS= parameter is 10^{-6} and accounts for the default optimization convergence tolerance at the fitting stage prior to model ranking. The convergence tolerance itself limits the accuracy that you can use to compare two models under a given criterion. As a result, smaller values of the RANKEPS= parameter might not lead to a sensible and more strict model comparison because for a smaller *reps-value*, ranking could depend on digits beyond the accuracy limit.

In the opposite end, if the specified *reps-value* turns out to be large compared to the criterion value differences, then it can make the sorting process insensitive to the specified sorting criterion. When this happens, the fit summary table ranking reflects only the order in which different models are examined in the procedure flow. You can tell whether the criterion is bypassed; if it is, then one or more values of the specified criterion might not appear to be sorted in the fit summary table.

The RANKEPS= parameter must be a positive number. The RANKEPS= option applies when you fit multiple models with the FORM=AUTO option of the MODEL statement; otherwise, it is ignored.

SCALE=*scale* | (*scale1, . . . , scalek*)

specifies the scale parameter in semivariogram models. The SCALE= option is optional. However, if you specify the SCALE= option, then you must provide sill values for all structures that you have specified explicitly in the FORM= option. All nonmissing scale values are considered as fixed parameters. PROC VARIOGRAM assigns a default initial value to any of the model structures for which you specify a missing scale value. PROC VARIOGRAM assigns default initial values to all model structures if you omit the SCALE= option, unless you specify an associated PARMS statement with initial values for scale.

The scale parameter is a positive number. It has the same units as the variance of the variable in the VAR statement. The scale of each structure in a semivariogram model represents the variance contribution of the structure to the total model variance.

In power models the SCALE= parameter does not correspond to a sill because the power model has no sill. Instead, PROC VARIOGRAM uses the SCALE= option to designate the slope (or scaling factor) in power model forms. The power model slope has the same variance units as the variable in the VAR statement.

The SCALE= option is ignored when you specify the FORM=AUTO option. The SCALE= option is incompatible with the specification of the PARMS statement for the corresponding MODEL statement.

SMOOTH=*smooth* | (*smooth1, . . . , smoothm*)

specifies the positive smoothness parameter ν in the Matérn type of semivariance structures. The special case $\nu = 0.5$ is equivalent to the exponential model, whereas the theoretical limit $\nu \to \infty$ gives the Gaussian model.

The SMOOTH= option is optional. When you specify an explicit model in the FORM= option with m Matérn structures, you can provide up to m smoothness values. You can specify a value for *smoothi*, $i = 1, \ldots, m$ that is positive and no greater than 1,000,000. PROC VARIOGRAM sets this upper limit for the SMOOTH= option values for numerical and performance reasons. In any case, if the fitting process leads the smoothness value to exceed the default threshold value 10,000, then the VARIOGRAM procedure converts the Matérn form into a Gaussian form and repeats the model fitting. To adjust the switching threshold value, you can use the MTOGTOL= option in the MODEL statement.

If you specify fewer than m values, then the remaining Matérn structures have their smoothness parameters initialized to missing values. If you specify more than m values, then values in excess are ignored.

All nonmissing smoothness values are considered as fixed parameters of the corresponding Matérn structures. PROC VARIOGRAM assigns a default initial value to any of the model Matérn structures, if any, for which you specify a missing smoothness value. PROC VARIOGRAM assigns default initial values to all model Matérn structures if you omit the SMOOTH= option, unless you specify an associated PARMS statement and initial values for smoothness in it.

The SMOOTH= option is ignored when you specify the FORM=AUTO option. The SMOOTH= option is incompatible with the specification of the PARMS statement for the corresponding MODEL statement.

In addition to the *fitting-options*, you can specify the following *model-options* after a slash (/) in the MODEL statement.

COVB

requests the approximate covariance matrix for the parameter estimates of the model fitting. The COVB option is ignored when you also specify the DETAILS=ALL option.

When you specify an explicit model with the FORM= option in the MODEL statement, the COVB option produces the requested approximate covariance matrix. When you specify the FORM=AUTO option in the MODEL statement, by default the COVB option produces output only for the selected model, where the choice is based on the criteria that you specify in the CHOOSE= option of the MODEL statement. If you specify the DETAILS option in addition to FORM=AUTO in the MODEL statement, then the COVB option produces output for each one of the fitted models.

CORRB

requests the approximate correlation matrix for the parameter estimates of the model fitting. The CORRB option is ignored when you also specify the DETAILS=ALL option.

When you specify an explicit model with the FORM= option in the MODEL statement, the CORRB option produces the requested approximate correlation matrix. When you specify the FORM=AUTO option in the MODEL statement, by default the CORRB option produces output only for the selected model, where the choice is based on the criteria that you specify in the CHOOSE= option of the MODEL statement. If you specify the DETAILS option in addition to FORM=AUTO in the MODEL statement, then the CORRB option produces output for each one of the fitted models.

DETAILS <= *detail-level* **>**

requests different levels of output to be produced during the fitting process. You can specify any of the following *detail-level* arguments:

MOD

specifies that the default output for all candidate models be produced when the FORM=AUTO option is specified in the MODEL statement. If you fit only one explicit model, then the DETAILS=MOD option has no effect and is ignored.

ITR

requests that a complete iteration history be produced in addition to the default output. The output for DETAILS=ITR includes the current values of the parameter estimates, their gradients, and additional optimization statistics.

ALL

requests the most detailed level of output when fitting a model. Specifically, except for the default output, the DETAILS=ALL option produces optimization statistics in addition to the combined output of the DETAILS=ITR, COVB, and CORRB options.

When you fit multiple models with the FORM=AUTO option in the MODEL statement, only the selected model default output is produced. The model selection is based on the criteria that you specify in the CHOOSE= option of the MODEL statement. With the DETAILS option you can produce ODS

tables with information about the fitting process of all the models that you fit. Moreover, you can produce output at different levels of detail that you can specify with the *detail-level* argument.

Omitting the DETAILS option or specifying the DETAILS option without any argument is equivalent to specifying DETAILS=MOD.

GRADIENT
displays the gradient of the objective function with respect to the parameter estimates in the "Parameter Estimates" table.

MTOGTOL=*number*
MTOL=*number*
specifies a threshold value for the smoothness parameter of the Matérn form. Above this threshold, a Matérn form in a model switches to the Gaussian form. The *number* value must be positive and no greater than 1,000,000, which is the smoothness upper bound set by the VARIOGRAM procedure.

By default, if the fitting process progressively increases the Matérn smoothness parameter ν without converging to a smoothness estimate, then PROC VARIOGRAM converts the Matérn form into a Gaussian form when smoothness exceeds the default value 10,000. If you specify the *number* value to be greater than the 1,000,000 boundary value, then it is ignored and reset to the default threshold value. For more details about the Matérn-to-Gaussian form conversion, see the section "Fitting with Matérn Forms" on page 11034.

NOFIT
suppresses the model fitting process.

NOITPRINT
suppresses the display of the iteration history table when you have also specified the DETAILS=ITR or DETAILS=ALL option in the MODEL statement. Otherwise, the NOITPRINT option is ignored.

NLOPTIONS Statement

NLOPTIONS < *options* > **;**

By default, PROC VARIOGRAM uses the technique TECH=NRRIDG, which corresponds to Newton-Raphson optimization with ridging. For more information about the NLOPTIONS, see the section "NLOPTIONS Statement" on page 533 in Chapter 20, "Shared Concepts and Topics."

PARMS Statement

PARMS (*value-list***)**. . . *< / options >* **;**

The PARMS statement specifies initial values for the semivariance parameters of a single specified model in the MODEL statement. Alternatively, the PARMS statement can request a grid search over several values of these parameters. You must specify the values by starting with the nugget effect parameter. You continue in the order in which semivariogram forms are specified in the FORM= option of the MODEL statement by specifying for each structure the values for its scale, range, and any other parameters as applicable.

The PARMS statement is optional and must follow the associated MODEL statement.

The *value-list* specification can take any of several forms:

m	a single value
$m_1, m_2, \ldots, m_n$	several values
m to n	a sequence in which m equals the starting value, n equals the ending value, and the increment equals 1
m to n by i	a sequence in which m equals the starting value, n equals the ending value, and the increment equals i
m_1, m_2 to m_3	mixed values and sequences

You can use the PARMS statement to input fixed values for parameters and also initial values that you want to optimize.

Suppose that you want to fit a semivariogram model with a Matérn component of scale 3, range 20, smoothing parameter 4.5, and an exponential component of unspecified scale and range 15. Assume that you also want to fix all the specified parameter values for the optimization. Including the nugget effect, you have a model with six parameters.

In terms of the PARMS statement, your specifications mean that you have initial values for the second, third, fourth, and sixth parameter in the parameter list. Also, the same specifications imply that you provide no initial values for the first parameter (which corresponds to the nugget effect) and the fifth parameter (which corresponds to the exponential model scale). For these parameters you prefer that PROC VARIOGRAM selects initial values, instead. Since you must specify values for all model parameters in the PARMS statement, you simply specify missing values for the first and fifth parameter. This is the way to request that PROC VARIOGRAM assigns default initial values to parameters. The SAS statements to implement these specifications are as follows:

```
proc variogram data=FirstData;
   < other VARIOGRAM statements >
   model form=(mat,exp);
   parms (.) (3) (20) (4.5) (.) (15) / hold=(2 to 4,6);
run;
```

NOTE: The preceding statements are equivalent to the following ones in which the PARMS statement is omitted:

```
proc variogram data=FirstData;
   < other VARIOGRAM statements >
   model form=(mat,exp) scale=(3,.) range(20,15) smooth=4.5;
run;
```

This example might suggest that you can always use either the PARMS or the MODEL statement to specify the same fitting parameters in the VARIOGRAM procedure. However, the PARMS statement gives you more flexibility in two ways:

- You can set non-default initial parameter values by using the PARMS statement, whereas in the MODEL statement you can request default initial values only by setting parameters to missing values. For this reason the PARMS statement cannot be specified when the FORM=AUTO option is specified in the associated MODEL statement. As an example, the following statements do not have an equivalent without using the PARMS statement, because the first parameter in the PARMS statement list (which corresponds to the NUGGET parameter) is set to the specific initial value of 2.1 and the fifth parameter (which corresponds to the exponential structure scale) is set to the specific initial value of 0.3.

```
proc variogram data=FirstData;
   < other VARIOGRAM statements >
   model form=(mat,exp);
   parms (2.1) (3) (20) (4.5) (0.3) (15) / hold=(2 to 4,6);
run;
```

- In the MODEL statement all the nonmissing parameter values that you specify remain fixed. Instead, the PARMS statement considers all values in the specified parameter sets to be subjected to optimization unless you force values to be fixed with the HOLD= option. In the previous example, you can specify that you want to optimize all of your parameters by skipping the HOLD= option as shown in the following modified statements:

```
proc variogram data=FirstData;
   < other VARIOGRAM statements >
   model form=(mat,exp);
   parms (2.1) (3) (20) (4.5) (1) (15);
run;
```

When you omit the PARMS statement list and the PDATA= data set in a PARMS statement, the specification is equivalent to a PARMS statement list where all the parameters have missing initial values. However, if you specify no other option in the PARMS statement, then the PARMS statement is ignored.

In order to avoid ambiguity, you cannot specify the PARMS statement if any of the scale, range, nugget, or smoothness parameters has been specified in the associated MODEL statement either explicitly or in the MDATA= data set. This condition is in effect even when you specify an empty PARMS statement.

If you specify more than one set of initial values, a grid of initial values sets is created. PROC VARIOGRAM seeks among the specified sets for the one that gives the lowest objective function value. Then, the procedure uses the initial values in the selected set for the fitting optimization.

The results from the PARMS statement are the values of the parameters on the specified grid. The ODS name of the “Parameter Search” table is ParmSearch.

You can specify the following *options* after a slash (/) in the PARMS statement:

HOLD=*value-list*

EQCONS=*value-list*

specifies which parameter values be constrained to equal the specified values. For example, the following statement constrains the first and third semivariance parameters to equal 0.5 and 12, respectively. The fourth parameter is fixed to the default initial value that is assigned to it by PROC VARIOGRAM.

```
parms (0.5) (3) (12) (.) / hold=1,3,4;
```

The HOLD= option accepts only nonmissing values in its list. If you specify more than the available parameters in the HOLD= option list, then the ones in excess are ignored. If the HOLD= option list has integer values that do not correspond to variables in the PARMS list, then they are also ignored. Noninteger values are rounded to the closest integer and evaluated accordingly.

When you specify more than one set of parameter initial values, the HOLD= option list applies to the set that gives the lowest objective function value before this set is sent to the optimizer for the fitting.

LOWERB=*value-list*

specifies lower boundary constraints on the semivariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that PROC VARIOGRAM uses for the semivariance parameters, and each number corresponds to the lower boundary constraint. A missing value instructs PROC VARIOGRAM to use its default constraint.

If you do not specify lower bounds for all of the semivariance parameters, then PROC VARIOGRAM assumes that the remaining parameters are not bounded. If you specify more lower bounds in the *value-list* than the available parameters, then the numbers in excess are ignored. If you specify lower bounds for parameters with missing initial values, then the VARIOGRAM procedure enforces the specified bounds in the fitting process. By default, the lower bound for all parameters is zero.

When you specify the HOLD= option together with the LOWERB= option, the lower bounds in the LOWERB= option *value-list* that correspond to fixed parameters are ignored. When you specify the NOBOUND option together with the LOWERB= option, the LOWERB= option is ignored.

MAXSCALE=*maxscale*

specifies a positive upper threshold for the fitted semivariogram sill. This option imposes a linear constraint on the optimization of the nonfixed semivariogram scale and nugget parameters so that the sum of all scale and nugget parameters does not exceed the specified MAXSCALE= value. The MAXSCALE= constraint is ignored if all the semivariogram scale and nugget parameters are fixed.

NOBOUND

requests the removal of boundary constraints on semivariance parameters. For example, semivariance parameters have a default zero lower boundary constraint since they have a physical meaning only for positive values. The NOBOUND option enables the fitting process to derive negative estimates; hence, you need to be cautious with the outcome when you specify this option.

The NOBOUND option has no effect on the power model exponent parameter. The exponent must range within [0,2) so that the model is a valid semivariance function. Also, the NOBOUND option has no effect on the Matérn smoothness parameter. The options LOWERB= and UPPERB= are ignored if either of them is specified together with the NOBOUND option in the PARMS statement.

PARMSDATA=*SAS-data-set*

PDATA=*SAS-data-set*

specifies that semivariance parameters values be read from a SAS data set. The data set should contain the values in the sequence required by the PARMS statement in either of the following two ways:

- Specify one single column under the variable Estimate (or Est) that contains all the parameter values.
- Use one column for each parameter, and place the *n* columns under the Parm1–Parm*n* variables.

For example, the following two data sets are valid and equivalent ways to specify initial values for the nugget effect and the parameters of the Matérn and exponential structures that have been used in the previous examples in the PARMS statement section:

```
data parData1;
   input Estimate @@;
   datalines;
. 3 20 4.5 . 15
;
```

```
data parData2;
   input Parm1 Parm2 Parm3 Parm4 Parm5 Parm6;
   datalines;
. 3 20 4.5 . 15
;
```

If you have the parData1 data set, then you can import this information into the PARMS statement as follows:

```
proc variogram data=FirstData;
   < other VARIOGRAM statements >
   model form=(mat,exp);
   parms / pdata=parData1 hold=(2 to 4,6);
run;
```

You can specify more than one set of initial values in the PDATA= data set by following the preceding guidelines. PROC VARIOGRAM seeks among the specified sets for the one that gives the lowest objective function value. Then, the procedure uses the initial values in the selected set for the fitting optimization.

You can explicitly specify initial parameter values in the PARMS statement or use the PDATA= option, but you cannot use both at the same time.

UPPERB=*value-list*

specifies upper boundary constraints on the semivariance parameters. The *value-list* specification is a list of numbers or missing values (.) separated by commas. You must list the numbers in the order that PROC VARIOGRAM uses for the semivariance parameters, and each number corresponds to the upper boundary constraint. A missing value instructs PROC VARIOGRAM to use its default constraint.

If you do not specify upper bounds for all of the semivariance parameters, then PROC VARIOGRAM assumes that the remaining parameters are not bounded. If you specify more upper bounds in the *value-list* than the available parameters, then the numbers in excess are ignored. If you specify upper bounds for parameters with missing initial values, then the VARIOGRAM procedure enforces the specified bounds in the fitting process. By default, the scale, range, nugget, and Matérn smoothness parameters have no upper bounds, whereas the power model exponent parameter is lower than two.

When you specify the HOLD= option together with the UPPERB= option, the upper bounds in the UPPERB= option *value-list* that correspond to fixed parameters are ignored. When you specify the NOBOUND option together with the UPPERB= option, the UPPERB= option is ignored.

STORE Statement

STORE OUT=*store-name* < / *option* > **;**

The STORE statement requests that the procedure save the context and results of the semivariogram model fitting analysis in an item store. An item store is a binary file defined by the SAS System. You cannot modify the contents of an item store. The contents of item stores produced by PROC VARIOGRAM can be processed only with the KRIGE2D or the SIM2D procedure. After you save results in an item store, you can use them at a later time without having to fit the model again.

The *store-name* is a usual one- or two-level SAS name, as for SAS data sets. If you specify a one-level name, then the item store resides in the Work library and is deleted at the end of the SAS session. Since item stores are often used for postprocessing tasks, typical usage specifies a two-level name of the form *libname.membername*. If an item store by the same name as specified in the STORE statement already exists, the existing store is replaced.

You can specify the following *option* in the STORE statement after a slash (/):

LABEL=*store-label*
specifies a custom label for the item store that is produced by PROC VARIOGRAM. When another procedure processes an item store, the label appears in the procedure's output along with other identifying information.

VAR Statement

VAR *analysis-variables-list* **;**

Use the VAR statement to specify the analysis variables. You can specify only numeric variables. If you omit the VAR statement, all numeric variables in the DATA= data set that are not in the COORDINATES statement are used.

Details: VARIOGRAM Procedure

Theoretical Semivariogram Models

The VARIOGRAM procedure computes the empirical (also known as *sample* or *experimental*) semivariogram from a set of point measurements. Semivariograms are used in the first steps of spatial prediction as tools that provide insight into the spatial continuity and structure of a random process. Naturally occurring randomness is accounted for by describing a process in terms of the *spatial random field* (SRF) concept (Christakos 1992). An SRF is a collection of random variables throughout your spatial domain of prediction. For some of them you already have measurements, and your data set constitutes part of a single realization of this SRF. Based on your sample, spatial prediction aims to provide you with values of the SRF at locations where no measurements are available.

Prediction of the SRF values at unsampled locations by techniques such as ordinary kriging requires the use of a theoretical semivariogram or covariance model. Due to the randomness involved in stochastic processes, the theoretical semivariance cannot be computed. Instead, it is possible that the empirical semivariance can provide an estimate of the theoretical semivariance, which then characterizes the spatial structure of the process.

The VARIOGRAM procedure follows a general flow of investigation that leads you from a set of spatial observations to an expression of theoretical semivariance to characterize the SRF continuity. Specifically, the empirical semivariogram is computed after a suitable choice is made for the LAGDISTANCE= and MAXLAGS= options. For computations in more than one direction you can further use the NDIRECTIONS= option or the DIRECTIONS statement. Potential theoretical models (which can also incorporate nesting, anisotropy, and the nugget effect) can be fitted to the empirical semivariance by using the MODEL statement, and then plotted against the empirical semivariogram. The flow of this analytical process is illustrated in Figure 131.17. After a suitable theoretical model is determined, it is used in PROC KRIGE2D for the prediction stage. The prediction analysis is presented in detail in the section "Details of Ordinary Kriging" on page 5594 in Chapter 74, "The KRIGE2D Procedure."

Figure 131.17 Flowchart for Semivariogram Selection

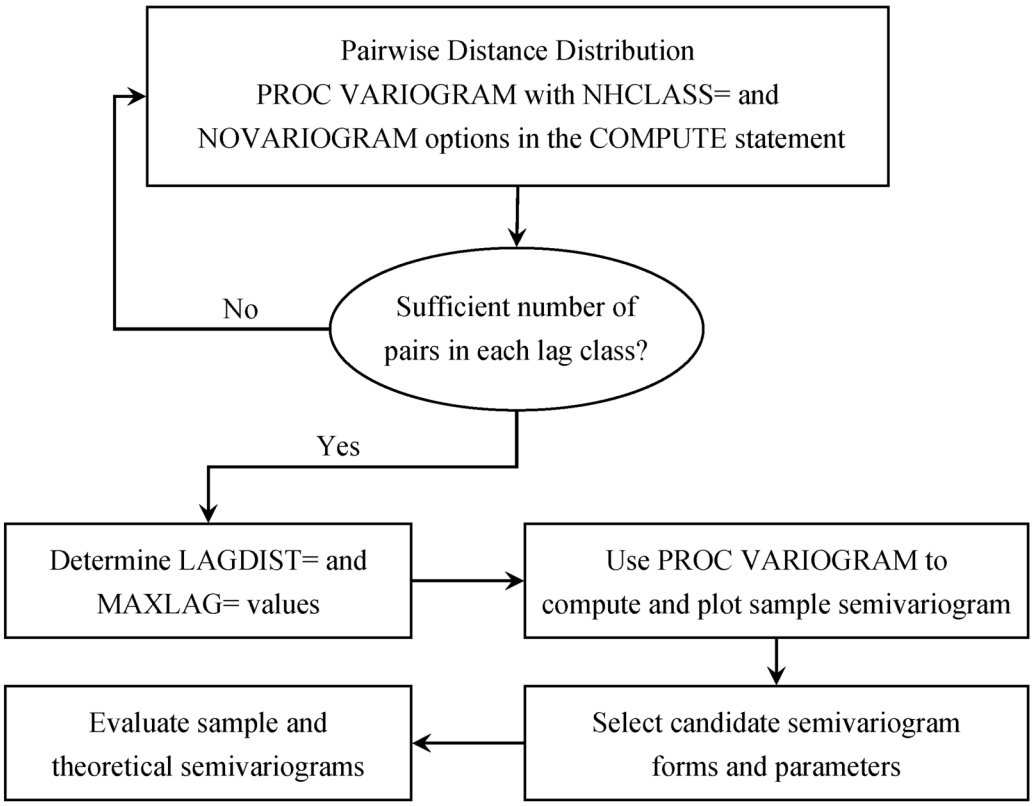

It is critical to note that the empirical semivariance provides an estimate of its theoretical counterpart only when the SRF satisfies stationarity conditions. These conditions imply that the SRF has a constant (or zero) expected value. Consequently, your data need to be sampled from a trend-free random field and need to have a constant mean, as assumed in "Getting Started: VARIOGRAM Procedure" on page 10962. Equivalently, your data could be residuals of an initial sample that has had a surface trend removed, as portrayed in "Example 131.2: An Anisotropic Case Study with Surface Trend in the Data" on page 11058. For a closer look at stationarity, see the section "Stationarity" on page 11015. For details about different stationarity types and conditions see, for example, Chilès and Delfiner (1999, section 1.1.4).

Characteristics of Semivariogram Models

When you obtain a valid empirical estimate of the theoretical semivariance, it is then necessary to choose a type of theoretical semivariogram model based on that estimate. Commonly used theoretical semivariogram shapes rise monotonically as a function of distance. The shape is typically characterized in terms of particular parameters; these are the *range* a_0, the *sill* (or *scale*) c_0, and the *nugget effect* c_n. Figure 131.18 displays a theoretical semivariogram of a spherical semivariance model and points out the semivariogram characteristics.

Figure 131.18 A Theoretical Semivariogram of Spherical Type and Its Characteristics

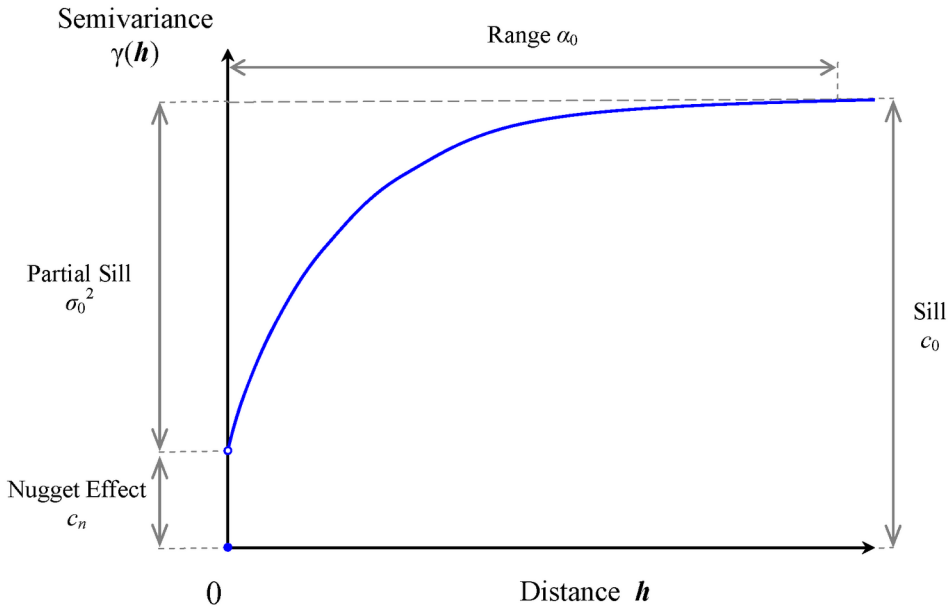

Specifically, the sill is the semivariogram upper bound. The range a_0 denotes the distance at which the semivariogram reaches the sill. When the semivariogram increases asymptotically toward its sill value, as occurs in the exponential and Gaussian semivariogram models, the term *effective* (or *practical*) range is also used. The effective range r_ϵ is defined as the distance at which there is essentially no lingering spatial correlation, which is usually considered to be the distance at which the semivariance value achieves 95% of the sill. This is based on the interpretation of effective range as provided in Banerjee, Carlin, and Gelfand (2014, pp. 26–27). In particular, for these models the relationship between the range and effective range is $r_\epsilon = 3a_0$ (exponential model) and $r_\epsilon = \sqrt{3}a_0$ (Gaussian model).

The nugget effect c_n represents a discontinuity of the semivariogram that can be present at the origin. It is typically attributed to microscale effects or measurement errors. The semivariance is always 0 at distance $\boldsymbol{h} = 0$; hence, the nugget effect demonstrates itself as a jump in the semivariance as soon as $\boldsymbol{h} > 0$ (note in Figure 131.18 the discontinuity of the function at $\boldsymbol{h} = 0$ in the presence of a nugget effect).

The sill c_0 consists of the nugget effect, if present, and the *partial sill* ${\sigma_0}^2$; that is, $c_0 = c_n + {\sigma_0}^2$. If the SRF $Z(\boldsymbol{s})$ is second-order stationary (see the section "Stationarity" on page 11015), the estimate of the sill is an estimate of the constant variance $\mathrm{Var}[Z(\boldsymbol{s})]$ of the field. Nonstationary processes have variances that depend on the location $\boldsymbol{s}$. Their semivariance increases with distance; hence their semivariograms have no sill.

Not every function is a suitable candidate for a theoretical semivariogram model. The semivariance function $\gamma_z(\boldsymbol{h})$, as defined in the following section, is a so-called *conditionally negative-definite* function that satisfies (Cressie 1993, p. 60)

$$\sum_{i=1}^{m}\sum_{j=i}^{m} q_i q_j \gamma_z(\boldsymbol{s}_i - \boldsymbol{s}_j) \le 0$$

for any number m of locations $\boldsymbol{s}_i, \boldsymbol{s}_j$ in $\mathcal{R}^2$ with $\boldsymbol{h} = \boldsymbol{s}_i - \boldsymbol{s}_j$, and any real numbers q_i such that $\sum_{i=1}^{m} q_i = 0$. PROC VARIOGRAM can use a variety of permissible theoretical semivariogram models. Specifically,

Table 131.4 shows a list of such models that you can use for fitting in the MODEL statement of the VARIOGRAM procedure.

Table 131.4 Permissible Theoretical Semivariogram Models ($a_0 > 0$, unless noted otherwise)

Model Type	Semivariance
Exponential	$\gamma_z(\boldsymbol{h}) = \begin{cases} 0 & \text{if } \mid \boldsymbol{h} \mid = 0 \\ c_n + {\sigma_0}^2 \left[1 - \exp\left(-\frac{\lvert\boldsymbol{h}\rvert}{a_0}\right)\right] & \text{if } 0 < \mid \boldsymbol{h} \mid \end{cases}$
Gaussian	$\gamma_z(\boldsymbol{h}) = \begin{cases} 0 & \text{if } \mid \boldsymbol{h} \mid = 0 \\ c_n + {\sigma_0}^2 \left[1 - \exp\left(-\frac{\lvert\boldsymbol{h}\rvert^2}{a_0^2}\right)\right] & \text{if } 0 < \mid \boldsymbol{h} \mid \end{cases}$
Power	$\gamma_z(\boldsymbol{h}) = \begin{cases} 0 & \text{if } \mid \boldsymbol{h} \mid = 0 \\ c_n + {\sigma_0}^2 \boldsymbol{h}^{a_0} & \text{if } 0 < \mid \boldsymbol{h} \mid, 0 \le a_0 < 2 \end{cases}$
Spherical	$\gamma_z(\boldsymbol{h}) = \begin{cases} 0 & \text{if } \mid \boldsymbol{h} \mid = 0 \\ c_n + {\sigma_0}^2 \left[\frac{3}{2}\frac{\lvert\boldsymbol{h}\rvert}{a_0} - \frac{1}{2}(\frac{\lvert\boldsymbol{h}\rvert}{a_0})^3\right] & \text{if } 0 < \mid \boldsymbol{h} \mid \le a_0 \\ c_0 & \text{if } a_0 < \mid \boldsymbol{h} \mid \end{cases}$
Cubic	$\gamma_z(\boldsymbol{h}) = \begin{cases} 0 & \text{if } \mid \boldsymbol{h} \mid = 0 \\ c_n + {\sigma_0}^2 \left[7(\frac{\lvert\boldsymbol{h}\rvert}{a_0})^2 - \frac{35}{4}(\frac{\lvert\boldsymbol{h}\rvert}{a_0})^3 + \frac{7}{2}(\frac{\lvert\boldsymbol{h}\rvert}{a_0})^5 - \frac{3}{4}(\frac{\lvert\boldsymbol{h}\rvert}{a_0})^7\right] & \text{if } 0 < \mid \boldsymbol{h} \mid \le a_0 \\ c_0 & \text{if } a_0 < \mid \boldsymbol{h} \mid \end{cases}$
Pentaspherical	$\gamma_z(\boldsymbol{h}) = \begin{cases} 0 & \text{if } \mid \boldsymbol{h} \mid = 0 \\ c_n + {\sigma_0}^2 \left[\frac{15}{8}\frac{\lvert\boldsymbol{h}\rvert}{a_0} - \frac{5}{4}(\frac{\lvert\boldsymbol{h}\rvert}{a_0})^3 + \frac{3}{8}(\frac{\lvert\boldsymbol{h}\rvert}{a_0})^5\right] & \text{if } 0 < \mid \boldsymbol{h} \mid \le a_0 \\ c_0 & \text{if } a_0 < \mid \boldsymbol{h} \mid \end{cases}$
Sine hole effect	$\gamma_z(\boldsymbol{h}) = \begin{cases} 0 & \text{if } \mid \boldsymbol{h} \mid = 0 \\ c_n + {\sigma_0}^2 \left[1 - \frac{\sin(\pi\lvert\boldsymbol{h}\rvert/a_0)}{\pi\lvert\boldsymbol{h}\rvert/a_0}\right] & \text{if } 0 < \mid \boldsymbol{h} \mid \end{cases}$
Matérn class	$\gamma_z(\boldsymbol{h}) = \begin{cases} 0 & \text{if } \mid \boldsymbol{h} \mid = 0 \\ c_n + {\sigma_0}^2 \left[1 - \frac{2}{\Gamma(\nu)}\left(\frac{\lvert\boldsymbol{h}\rvert\sqrt{\nu}}{a_0}\right)^{\nu} K_{\nu}\left(2\frac{\lvert\boldsymbol{h}\rvert\sqrt{\nu}}{a_0}\right)\right] & \text{if } 0 < \mid \boldsymbol{h} \mid, \nu > 0 \end{cases}$

All of these models, except for the power model, are transitive. A transitive model characterizes a random process whose variation reaches the sill value c_0 within a specific range from any location in the field.

The power model is nontransitive and applies to processes whose variance increases with distance. It has no scale and range; instead, it quantifies the process variation by using a positive slope parameter and a dimensionless power exponent α that indicate how fast the variance increases. The expression for the power model is a valid semivariogram only when the exponent parameter ranges within $0 \le \alpha < 2$. For convenience, PROC VARIOGRAM registers the power model slope parameter under the SCALE= option parameters in the MODEL statement. For the same reason, the scale and power slope parameters are represented with the common symbol ${\sigma_0}^2$ in Table 131.4. Also for convenience, PROC VARIOGRAM registers the power model exponent parameter under the RANGE= option parameters. The range and the power exponent parameters

are represented with the common symbol a_0 in Table 131.4.

The power model is a generalized case of the linear model, which is not included explicitly in the model set of PROC VARIOGRAM. The linear model is derived from the power model when you specify the exponent $\alpha = 1$.

Among the models displayed in Table 131.4, the Matérn (or K-Bessel) class is a class of semivariance models that distinguish from each other by means of the positive smoothing parameter ν. Different values of ν correspond to different correlation models. Most notably, for $\nu = 0.5$ the Matérn semivariance is equivalent to the exponential model, whereas $\nu \to \infty$ gives the Gaussian model. Also, Table 131.4 shows that the Matérn semivariance computations use the gamma function $\Gamma(\nu)$ and the second kind Bessel function K_ν.

In PROC VARIOGRAM, you can input the model parameter values either explicitly as arguments of options, or as lists of values. In the latter case, you are expected to provide the values in the order the models are specified in the SAS statements, and furthermore in the sequential order of the scale, range, and smoothing parameter for each model as appropriate, and always starting with the nugget effect. If the parameter values are specified through an input file, then the total of n parameters should be provided either as one variable named Estimate or as many variables with the respective names Parm1–Parm$n$.

You can review in further detail the models shown in Table 131.4 in the section "Theoretical Semivariogram Models" on page 5577 in Chapter 74, "The KRIGE2D Procedure."

The theoretical semivariogram models are used to describe the spatial structure of random processes. Based on their shape and characteristics, the semivariograms of these models can provide a plethora of information (Christakos 1992, section 7.3):

- Examination of the semivariogram variation in different directions provides information about the isotropy of the random process. (See also the discussion about isotropy in the following section.)
- The semivariogram range determines the zone of influence that extends from any given location. Values at surrounding locations within this zone are correlated with the value at the specific location by means of the particular semivariogram.
- The semivariogram behavior at large distances indicates the degree of stationarity of the process. In particular, an asymptotic behavior suggests a stationary process, whereas either a linear increase and slow convergence to the sill or a fast increase is an indicator of nonstationarity.
- The semivariogram behavior close to the origin indicates the degree of regularity of the process variation. Specifically, a parabolic behavior at the origin implies a very regular spatial variation, whereas a linear behavior characterizes a nonsmooth process. The presence of a nugget effect is additional evidence of irregularity in the process.
- The semivariogram behavior within the range provides description of potential periodicities or anomalies in the spatial process.

A brief note on terminology: In some fields (for example, geostatistics) the term homogeneity is sometimes used instead of stationarity in spatial analysis; however, in statistics homogeneity is defined differently (Banerjee, Carlin, and Gelfand 2004, section 2.1.3). In particular, the alternative terminology characterizes as homogeneous the stationary SRF in $\mathcal{R}^n, n > 1$, whereas it retains the term stationary for such SRF in $\mathcal{R}^1$ (SRF in $\mathcal{R}^1$ are also known as *random processes*). Often, studies in a single dimension refer to temporal processes; hence, you might see time-stationary random processes called "temporally stationary" or

simply stationary, and stationary SRF in $\mathcal{R}^n, n > 1$, characterized as "spatially homogeneous" or simply homogeneous. This distinction made by the alternative nomenclature is more evident in spatiotemporal random fields (S/TRF), where the different terms clarify whether stationarity applies in the spatial or the temporal part of the S/TRF.

Nested Models

When you try to represent an empirical semivariogram by fitting a theoretical model, you might find that using a combination of theoretical models results in a more accurate fit onto the empirical semivariance than using a single model. This is known as model nesting. The semivariance models that result as the sum of two or more semivariance structures are called *nested* models.

In general, a linear combination of permissible semivariance models produces a new permissible semivariance model. Nested models are based on this premise. You can include in a sum any combination of the models presented in Table 131.4. For example, a nested semivariance $\gamma_z(\boldsymbol{h})$ that contains two structures, one exponential $\gamma_{z,\mathrm{EXP}}(\boldsymbol{h})$ and one spherical $\gamma_{z,\mathrm{SPH}}(\boldsymbol{h})$, can be expressed as

$$\gamma_z(\boldsymbol{h}) = \gamma_{z,\mathrm{EXP}}(\boldsymbol{h}) + \gamma_{z,\mathrm{SPH}}(\boldsymbol{h})$$

If you have a nested model and a nugget effect, then the nugget effect c_n is a single parameter that is considered jointly for all the nested structures.

Nested models, anisotropic models, and the nugget effect increase the scope of theoretical models available. You can find additional discussion about these concepts in the section "Theoretical Semivariogram Models" on page 5577 in Chapter 74, "The KRIGE2D Procedure."

Theoretical and Computational Details of the Semivariogram

Let $\{Z(\boldsymbol{s}), \boldsymbol{s} \in D \subset \mathcal{R}^2\}$ be a spatial random field (SRF) with n measured values $z_i = Z(\boldsymbol{s}_i)$ at respective locations $\boldsymbol{s}_i, i = 1, \ldots, n$. You use the VARIOGRAM procedure because you want to gain insight into the spatial continuity and structure of $Z(\boldsymbol{s})$. A good measure of the spatial continuity of $Z(\boldsymbol{s})$ is defined by means of the variance of the difference $Z(\boldsymbol{s}_i) - Z(\boldsymbol{s}_j)$, where $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ are locations in D. Specifically, if you consider $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ to be spatial increments such that $\boldsymbol{h} = \boldsymbol{s}_j - \boldsymbol{s}_i$, then the variance function based on the increments $\boldsymbol{h}$ is independent of the actual locations $\boldsymbol{s}_i, \boldsymbol{s}_j$. Most commonly, the continuity measure used in practice is one half of this variance, better known as the *semivariance* function,

$$\gamma_z(\boldsymbol{h}) = \frac{1}{2}\mathrm{Var}[Z(\boldsymbol{s}+\boldsymbol{h}) - Z(\boldsymbol{s})]$$

or, equivalently,

$$\gamma_z(\boldsymbol{h}) = \frac{1}{2}\left(\mathrm{E}\{[Z(\boldsymbol{s}+\boldsymbol{h}) - Z(\boldsymbol{s})]^2\} - \{\mathrm{E}[Z(\boldsymbol{s}+\boldsymbol{h})] - \mathrm{E}[Z(\boldsymbol{s})]\}^2\right)$$

The plot of semivariance as a function of $\boldsymbol{h}$ is the *semivariogram*. You might also commonly see the term *semivariogram* used instead of the term *semivariance*.

Assume that the SRF $Z(\boldsymbol{s})$ is free of nonrandom (or systematic) surface trends. Then, the expected value $\mathrm{E}[Z(\boldsymbol{s})]$ of $Z(\boldsymbol{s})$ is a constant for all $\boldsymbol{s} \in \mathcal{R}^2$, and the semivariance expression is simplified to the following:

$$\gamma_z(\boldsymbol{h}) = \frac{1}{2}\mathrm{E}\{[Z(\boldsymbol{s}+\boldsymbol{h}) - Z(\boldsymbol{s})]^2\}$$

Given the preceding assumption, you can compute an estimate $\hat{\gamma}_z(\boldsymbol{h})$ of the semivariance $\gamma_z(\boldsymbol{h})$ from a finite set of points in a practical way by using the formula

$$\hat{\gamma}_z(\boldsymbol{h}) = \frac{1}{2 \mid N(\boldsymbol{h}) \mid} \sum_{N(\boldsymbol{h})} [Z(\boldsymbol{s}_i) - Z(\boldsymbol{s}_j)]^2$$

where the sets $N(\boldsymbol{h})$ contain all the neighboring pairs at distance $\boldsymbol{h}$,

$$N(\boldsymbol{h}) = \{i, j : \boldsymbol{s}_i - \boldsymbol{s}_j = \boldsymbol{h}\}$$

and $\mid N(\boldsymbol{h}) \mid$ is the number of such pairs (i, j).

The expression for $\hat{\gamma}_z(\boldsymbol{h})$ is called the *empirical semivariance* (Matheron 1963). This is the quantity that PROC VARIOGRAM computes, and its corresponding plot is the *empirical semivariogram*.

The empirical semivariance $\hat{\gamma}_z(\boldsymbol{h})$ is also referred to as *classical*. This name is used so that it can be distinguished from the *robust semivariance* estimate $\bar{\gamma}_z(\boldsymbol{h})$ and the corresponding *robust semivariogram*. The robust semivariance was introduced by Cressie and Hawkins (1980) to weaken the effect that outliers in the observations might have on the semivariance. It is described by Cressie (1993, p. 75) as

$$\bar{\gamma}_z(\boldsymbol{h}) = \frac{\Psi^4(\boldsymbol{h})}{2[0.457 + 0.494/N(\boldsymbol{h})]}$$

In the preceding expression the parameter $\Psi(\boldsymbol{h})$ is defined as

$$\Psi(\boldsymbol{h}) = \frac{1}{N(\boldsymbol{h})} \sum_{P_i P_j \in N(\boldsymbol{h})} [Z(\boldsymbol{s}_i) - Z(\boldsymbol{s}_j)]^{\frac{1}{2}}$$

According to Cressie (1985), the estimate $\hat{\gamma}_z(\boldsymbol{h})$ has approximate variance

$$\mathrm{Var}[\hat{\gamma}_z(\boldsymbol{h})] \simeq \frac{2[\gamma_z(\boldsymbol{h})]^2}{N(\boldsymbol{h})}$$

This approximation is possible by assuming $Z(\boldsymbol{s})$ to be a Gaussian SRF, and by further assuming the squared differences in empirical semivariances to be uncorrelated for different distances $\boldsymbol{h}$. Typically, semivariance estimates are correlated because of the underlying spatial correlation among the observations, and also because the same observation pairs might be used for the estimation of more than one semivariogram point, as described in the following subsections. Despite these restrictive assumptions, the approximate variance provides an idea about the semivariance estimate variance and enables fitting of a theoretical model to the empirical semivariance; see the section "Theoretical Semivariogram Model Fitting" on page 11027 for more details about the fitting process.

NOTE: If your data include a surface trend, then the empirical semivariance $\hat{\gamma}_z(\boldsymbol{h})$ is not an estimate of the theoretical semivariance function $\gamma_z(\boldsymbol{h})$. Instead, rather than the spatial increments variance, it represents a different quantity known as *pseudo-semivariance*, and its corresponding plot is a *pseudo-semivariogram*. In principle, pseudo-semivariograms do not provide measures of the spatial continuity. They can thus lead to misinterpretations of the $Z(\boldsymbol{s})$ spatial structure, and are consequently unsuitable for the purpose of spatial prediction. For further information, see the detailed discussion in the section "Empirical Semivariograms and Surface Trends" on page 11026. Under certain conditions you might be able to gain some insight about the spatial continuity with a pseudo-semivariogram. This case is presented in "Example 131.3: Analysis without Surface Trend Removal" on page 11071.

Stationarity

In the combined presence of the previous two assumptions—that is, when $\mathrm{E}[Z(\boldsymbol{s})]$ is constant and spatial increments define $\gamma_z(\boldsymbol{h})$—the SRF $Z(\boldsymbol{s})$ is characterized as *intrinsically stationary* (Cressie 1993, p. 40).

The expected value $\mathrm{E}[Z(\boldsymbol{s})]$ is the first statistical moment of the SRF $Z(\boldsymbol{s})$. The second statistical moment of the SRF $Z(\boldsymbol{s})$ is the *covariance* function between two points $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ in $Z(\boldsymbol{s})$, and it is defined as

$$C_z(\boldsymbol{s}_i, \boldsymbol{s}_j) = \mathrm{E}\left([Z(\boldsymbol{s}_i) - \mathrm{E}[Z(\boldsymbol{s}_i)]]\left[Z(\boldsymbol{s}_j) - \mathrm{E}[Z(\boldsymbol{s}_j)]\right]\right)$$

When $\boldsymbol{s}_i = \boldsymbol{s}_j = \boldsymbol{s}$, the covariance expression provides the variance at $\boldsymbol{s}$.

The assumption of a constant $\mathrm{E}[Z(\boldsymbol{s})] = m$ means that the expected value is invariant with respect to translations of the spatial location $\boldsymbol{s}$. The covariance is considered invariant to such translations when it depends only on the distance $\boldsymbol{h} = \boldsymbol{s}_i - \boldsymbol{s}_j$ between any two points $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$. If both of these conditions are true, then the preceding expression becomes

$$C_z(\boldsymbol{s}_i, \boldsymbol{s}_j) = C_z(\boldsymbol{s}_i - \boldsymbol{s}_j) = C_z(\boldsymbol{h}) = \mathrm{E}\left([Z(\boldsymbol{s}) - m][Z(\boldsymbol{s} + \boldsymbol{h}) - m]\right)$$

When both $\mathrm{E}[Z(\boldsymbol{s})]$ and $C(\boldsymbol{s}_i, \boldsymbol{s}_j)$ are invariant to spatial translations, the SRF $Z(\boldsymbol{s})$ is characterized as *second-order stationary* (Cressie 1993, p. 53).

In a second-order stationary SRF the quantity $C(\boldsymbol{h})$ is the same for any two points that are separated by distance $\boldsymbol{h}$. Based on the preceding formula, for $\boldsymbol{h} = 0$ you can see that the variance is constant throughout a second-order stationary SRF. Hence, second-order stationarity is a stricter condition than intrinsic stationarity.

Under the assumption of second-order stationarity, the semivariance definition at the beginning of this section leads to the conclusion that

$$\gamma_z(\boldsymbol{h}) = C(\boldsymbol{0}) - C(\boldsymbol{h})$$

which relates the theoretical semivariance and covariance. Keep in mind that the empirical estimates of these quantities are not related in exactly the same way, as indicated in Schabenberger and Gotway (2005, section 4.2.1).

Ergodicity

In addition to the constant $\mathrm{E}[Z(\boldsymbol{s})]$ and the assumption of intrinsic stationarity, *ergodicity* is a necessary third hypothesis to estimate the empirical semivariance. Assume that for the SRF $Z(\boldsymbol{s})$ you have measurements z_i whose sample mean is estimated by $\bar{Z}$. The hypothesis of ergodicity dictates that $\bar{Z} = \mathrm{E}[Z(\boldsymbol{s})]$.

In general, an SRF $Z(\boldsymbol{s})$ is characterized as ergodic if the statistical moments of its realizations coincide with the corresponding ones of the SRF. In spatial analysis you are often interested in the first two statistical moments, and consequently a more relaxed ergodicity assumption is made only for them. See Christakos (1992, section 2.12) for the use of the ergodicity hypothesis in SRF, and Cressie (1993, p. 57) for a more detailed discussion of ergodicity.

The semivariogram analysis makes implicit use of the ergodicity hypothesis. The VARIOGRAM procedure works with the residual centered values $V(\boldsymbol{s}_i) = v_i = z_i - \bar{Z}$, $i = 1, \ldots, n$, where it is assumed that the sample mean $\bar{Z}$ is the constant expected value $\mathrm{E}[Z(\boldsymbol{s})]$ of $Z(\boldsymbol{s})$. This is equivalent to using the original values, since $V(\boldsymbol{s}_i) - V(\boldsymbol{s}_j) = Z(\boldsymbol{s}_i) - Z(\boldsymbol{s}_j)$, which shows the property of the semivariance to filter out the mean. See the section "Semivariance Computation" on page 11025 for the exact expressions PROC VARIOGRAM uses to compute the empirical classical $\hat{\gamma}_z(\boldsymbol{h})$ and robust $\bar{\gamma}_z(\boldsymbol{h})$ semivariances.

Anisotropy

Semivariance is defined on the basis of the spatial increment vector $\boldsymbol{h}$. If the variance characteristics of $Z(\boldsymbol{s})$ are independent of the spatial direction, then $Z(\boldsymbol{s})$ is called *isotropic*; if not, then $Z(\boldsymbol{s})$ is called *anisotropic*. In the case of isotropy, the semivariogram depends only on the length h of $\boldsymbol{h}$ and $\gamma_z(\boldsymbol{h}) = \gamma_z(h)$. Anisotropy is characterized as *geometric*, when the range a_0 of the semivariogram varies in different directions, and *zonal*, when the semivariogram sill c_0 depends on the spatial direction. Either type or both types of anisotropy can be present.

In the more general case, an SRF can be anisotropic. For an accurate characterization of the spatial structure it is necessary to perform individual analyses in multiple directions. Goovaerts (1997, p. 98) suggests an initial investigation in at least one direction more than the working spatial dimensions—for example, at least three different directions in $\mathcal{R}^2$. Olea (2006) supports exploring as many directions as possible when the data set allows.

You might not know in advance whether you have anisotropy or not. If the semivariogram characteristics remain unchanged in different directions, then you assume the SRF is isotropic. If your directional analysis reveals anisotropic behavior in particular directions, then you proceed to focus your analysis on these directions. For example, in an anisotropic SRF in $\mathcal{R}^2$ you should expect to find two distinct directions where you observe the *major axis* and the *minor axis* of anisotropy. Typically, these two directions are perpendicular, although they might be at other than right angles when zonal anisotropy is present.

If you can distinguish a maximum and a minimum sill in different directions, then you have a case of zonal anisotropy. The SRF exhibits strongest continuity in the direction of the lowest sill, which is the direction of the major anisotropy axis. If the sill does not change across directions, then the major axis direction of strongest continuity is the one in which the semivariogram has maximum range. See "Example 131.2: An Anisotropic Case Study with Surface Trend in the Data" on page 11058 for a detailed demonstration of a case with anisotropy when you use PROC VARIOGRAM.

You can find additional information about anisotropy analysis in the section "Anisotropic Models" on page 5586 in Chapter 74, "The KRIGE2D Procedure."

Pair Formation

The basic starting point in computing the empirical semivariance is the enumeration of pairs of points for the spatial data. Figure 131.19 shows the spatial domain D and the set of n measurements $z_i, i = 1, \ldots, n$, that have been sampled at the indicated locations in D. Two data points P_1 and P_2, with coordinates $\boldsymbol{s}_1 = (x_1, y_1)$ and $\boldsymbol{s}_2 = (x_2, y_2)$, respectively, are selected for illustration.

A vector, or directed line segment, is drawn between these points. If the length

$$| P_i P_j | = | \boldsymbol{s}_2 - \boldsymbol{s}_1 | = (x_2 - x_1)^2 + (y_2 - y_1)^2$$

of this vector is smaller than the specified DEPSILON= value, then the pair is excluded from the continuity measure calculations because the two points P_1 and P_2 are considered to be at zero distance apart (or *collocated*). Spatial collocation might appear due to different scales in sampling, observations made at the same spatial location at different time instances, and errors in the data sets. PROC VARIOGRAM excludes such pairs from the pairwise distance and semivariance computations because they can cause numeric problems in spatial analysis.

If this pair is not discarded on the basis of collocation, it is then classified—first by orientation of the directed line segment $\boldsymbol{s}_2 - \boldsymbol{s}_1$, and then by its length $| P_i P_j |$. For example, it is unlikely for actual data that the

distance $| P_i P_j |$ between any pair of data points P_i and P_j located at $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$, respectively, would exactly satisfy $| P_i P_j | = | \boldsymbol{h} | = h$ in the preceding computation of $\hat{\gamma}_z(\boldsymbol{h})$. A similar argument can be made for the orientation of the segment $\boldsymbol{s}_2 - \boldsymbol{s}_1$. Consequently, the pair $P_1 P_2$ is placed into an angle and distance class.

The following subsections give more details about the nature of these classifications. You can also find extensive discussions about the size and the number of classes to consider for the computation of the empirical semivariogram.

Figure 131.19 Selection of Points P_1 and P_2 in Spatial Domain D

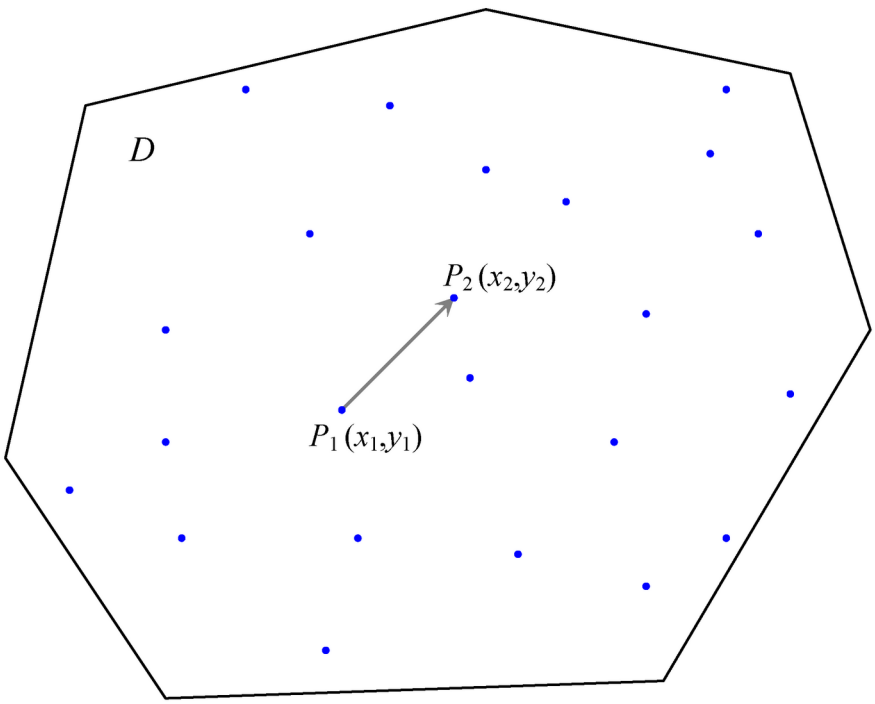

Angle Classification

Suppose you specify NDIRECTIONS=3 in the COMPUTE statement in PROC VARIOGRAM. This results in three angle classes defined by midpoint angles between 0° and 180°: $0° \pm \delta\theta$, $60° \pm \delta\theta$, and $120° \pm \delta\theta$, where $\delta\theta$ is the angle tolerance. If you do not specify an angle tolerance by using the ANGLETOLERANCE= option in the COMPUTE statement, the following default value is used:

$$\delta\theta = \frac{180°}{2 \times \mathrm{NDIR}}$$

For example, if NDIRECTIONS=3, the default angle tolerance is $\delta\theta = 30°$. When the directed line segment $P_1 P_2$ in Figure 131.19 is superimposed on the coordinate system that shows the angle classes, its angle is approximately 45°, measured clockwise from north. In particular, it falls within $[60° - \delta\theta, 60° + \delta\theta) = [30°, 90°)$, the second angle class (Figure 131.20).

NOTE: If the designated points P_1 and P_2 are labeled in the opposite order, the orientation is in the opposite direction—that is, approximately 225° instead of approximately 45°. This does not affect angle class selection; the angle classes $[60° - \delta\theta, 60° + \delta\theta)$ and $[240° - \delta\theta, 240° + \delta\theta)$ are the same.

Figure 131.20 Selected Pair $P_1 P_2$ Falls within the Second Angle Class

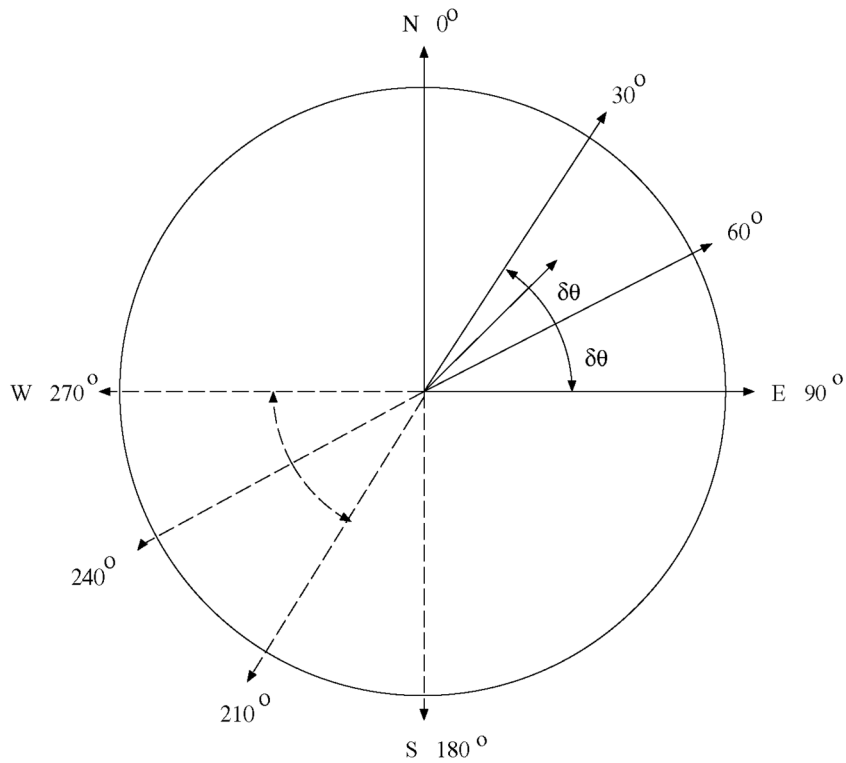

If you specify an angle tolerance less than the default, such as ATOL=15°, some point pairs might be excluded. For example, the selected point pair $P_1 P_2$ in Figure 131.20, while closest to the 60° axis, might lie outside $[60 - \delta\theta, 60 + \delta\theta) = [45°, 75°)$. In this case, the point pair $P_1 P_2$ would be excluded from the semivariance computation. This setting can be desirable if you want to reduce interference between neighboring angles. An angle tolerance that is too small might result in too few point pairs in some distance classes for the empirical semivariance estimation (see also the discussion in the section "Choosing the Size of Classes" on page 11023).

On the other hand, you can specify an angle tolerance *greater* than the default. This can result in a point pair being counted in more than one angle classes. This has a smoothing effect on the variogram and is useful when only a small amount of data is present or the available data are sparsely located. However, in cases of anisotropy the smoothing effect might have the side effect of amplifying weaker anisotropy in some direction and weakening stronger anisotropy in another (Deutsch and Journel 1992, p. 59).

Changes in the values of the BANDWIDTH= option have a similar effect. See the section "Bandwidth Restriction" on page 11020 for an explanation of how BANDWIDTH= functions.

An alternative way to specify angle classes and angle tolerances is with the DIRECTIONS statement. The DIRECTIONS statement is useful when angle classes are not equally spaced. When you use the DIRECTIONS statement, consider specifying the angle tolerance too. The default value of the angle tolerance is 45° when a DIRECTIONS statement is used instead of the NDIRECTIONS= option in the COMPUTE statement. This might not be appropriate for a particular set of angle classes. See the section "DIRECTIONS Statement" on page 10990 for more details.

Distance Classification

The distance class for a point pair $P_1 P_2$ is determined as follows. The directed line segment $P_1 P_2$ is superimposed on the coordinate system that shows the distance or lag classes. These classes are determined by the LAGDISTANCE= option in the COMPUTE statement. Denoting the length of the line segment by $| P_1 P_2 |$ and the LAGDISTANCE= value by Δ, the lag class L is determined by

$$L(P_1 P_2) = \left\lfloor \frac{| P_1 P_2 |}{\Delta} + 0.5 \right\rfloor$$

where $\lfloor x \rfloor$ denotes the largest integer $\leq x$.

When the directed line segment $P_1 P_2$ is superimposed on the coordinate system that shows the distance classes, it is seen to fall in the first lag class; see Figure 131.21 for an illustration for $\Delta = 1$.

Figure 131.21 Selected Pair $P_1 P_2$ Falls within the First Lag Class

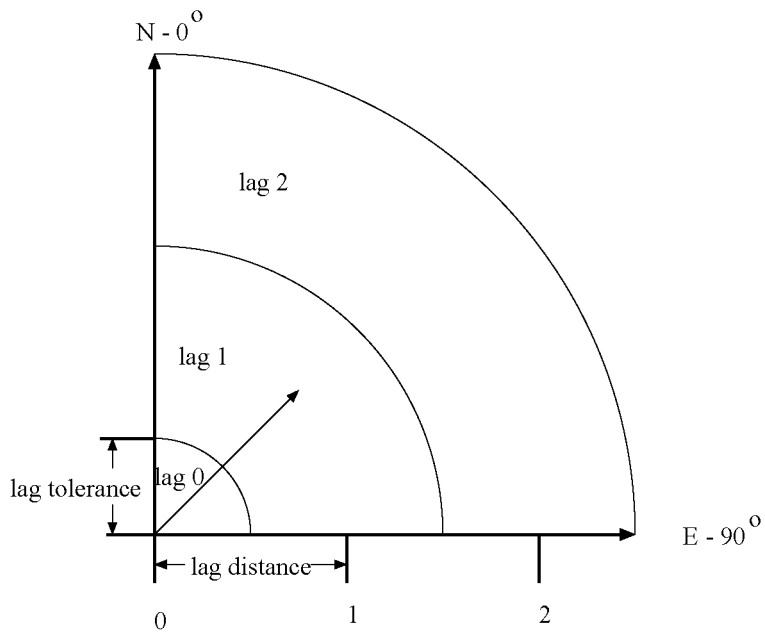

Pairwise distances are positive. Therefore, the line segment $| P_1 P_2 |$ might belong to one of the MAXLAG lag classes or it could be shorter than half the length of the LAGDISTANCE= value. In the last case the segment is said to belong to the lag class zero. Hence, lag class zero is smaller than lag classes $1, \ldots,$ MAXLAGS. The definition of lag classes in this manner means that when you specify the MAXLAGS= parameter, PROC VARIOGRAM produces a semivariogram with a total of MAXLAGS+1 lag classes including the zero lag class. For example, if you specify LAGDISTANCE=1 and MAXLAGS=10 and you do not specify a LAGTOLERANCE= value in the COMPUTE statement in PROC VARIOGRAM, the 11 lag classes generated by the preceding equation are

$$[0, 0.5), [0.5, 1.5), [1.5, 2.5), \ldots, [9.5, 10.5)$$

The preceding lag classes description is correct under the assumption of the default lag tolerance, which is half the LAGDISTANCE= value. Using the default lag tolerance results in no gaps between the distance class intervals, as shown in Figure 131.22.

Figure 131.22 Lag Distance Axis Showing Lag Classes

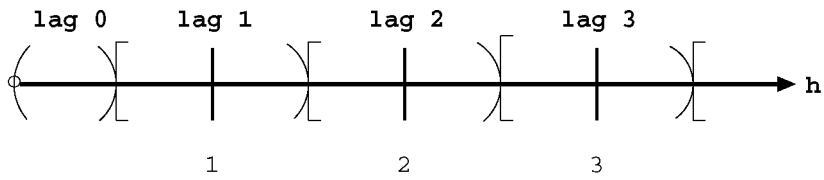

On the other hand, if you do specify a distance tolerance with the LAGTOLERANCE= option in the COMPUTE statement, a further check is performed to see whether the point pair falls within this tolerance of the nearest lag. In the preceding example, if you specify LAGDISTANCE=1 and MAXLAGS=10 (as before) and also specify LAGTOLERANCE=0.25, the intervals become

$$[0, 0.25), [0.75, 1.25), [1.75, 2.25), \ldots, [9.75, 10.25)$$

You might want to avoid this specification because it results in gaps in the lag classes. For example, if a point pair $P_1 P_2$ falls in an interval such as

$$| P_1 P_2 | \in [1.25, 1.75)$$

then it is excluded from the semivariance calculation. The maximum LAGTOLERANCE= value allowed is half the LAGDISTANCE= value; no overlap of the distance classes is allowed.

See the section "Computation of the Distribution Distance Classes" on page 11021 for a more extensive discussion of practical aspects in the specification of the LAGDISTANCE= and MAXLAGS= options.

Bandwidth Restriction

Because the areal segments that are generated from the angle and distance classes increase in area as the lag distance increases, it is sometimes desirable to restrict this area (Deutsch and Journel 1992, p. 45). If you specify the BANDWIDTH= option in the COMPUTE statement, the lateral, or perpendicular, distance from the axis that defines the angle classes is fixed.

For example, suppose two points P_3, P_4 are picked from the domain in Figure 131.19 and are superimposed on the grid that defines distance and angle classes, as shown in Figure 131.23.

The endpoint of vector $P_3 P_4$ falls within the angle class around 60° and the fifth lag class; however, it falls outside the restricted area that is defined by the bandwidth. Hence, it is excluded from the semivariance calculation.

Figure 131.23 Selected Pair $P_3 P_4$ Falls outside Bandwidth Limit

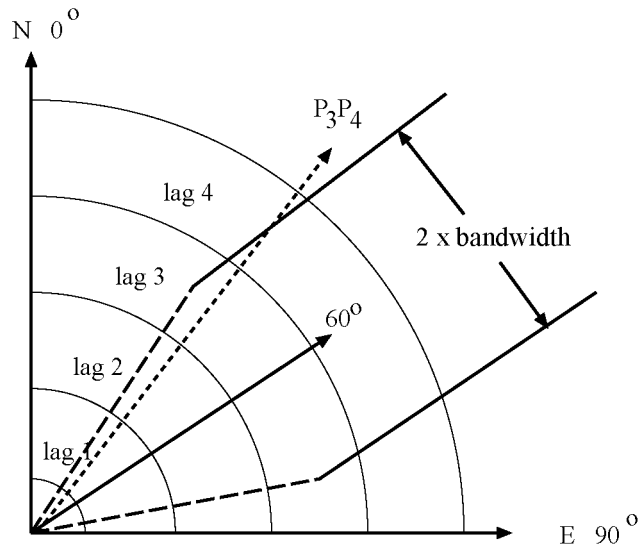

Finally, a pair $P_i P_j$ that falls in a lag class larger than the value of the MAXLAGS= option is excluded from the semivariance calculation.

The BANDWIDTH= option complements the angle and lag tolerances in determining how point pairs are included in distance classes. Clearly, the number of pairs within each angle/distance class is strongly affected by the angle and lag tolerances and whether BANDWIDTH= has been specified. See also the section "Angle Classification" on page 11017 for more details about the effects these rules can have, since BANDWIDTH= operates in a manner similar to the ANGLETOLERANCE= option.

Computation of the Distribution Distance Classes

This section deals with theoretical considerations and practical aspects when you specify the LAGDISTANCE= and MAXLAGS= options. In principle, these values depend on the amount and spatial distribution of your experimental data.

The value of the LAGDISTANCE= option regulates how many pairs of data are contained within each distance class. In effect, this information defines the pairwise distance distribution (see the following subsection). Your choice of MAXLAGS= specifies how many of these lags you want to include in the empirical semivariogram computation. Adjusting the values of these parameters is a crucial part of your analysis. Based on your observations sample, they determine whether you have sufficient points for a descriptive empirical semivariogram, and they can affect the accuracy of the estimated semivariance, too.

The simplest way of determining the distribution of pairwise distances is to determine the maximum distance h_{max} between any pair of points in your data, and then to divide this distance by some number N of intervals to produce distance classes of length $\delta = h_{max}/N$. The distance $\mid P_1 P_2 \mid$ between each pair of points P_1, P_2 is computed, and the pair $P_1 P_2$ is counted in the kth distance class if $\mid P_1 P_2 \mid \in [(k-1)\delta, k\delta)$ for $k = 1, \ldots, N$.

The actual computation is a slight variation of this. A bound, rather than the actual maximum distance, is computed. This bound is the length of the diagonal of a bounding rectangle for the data points. This bounding rectangle is found by using the maximum and minimum x and y coordinates, $x_{max}, x_{min}, y_{max}, y_{min}$, and forming the rectangle determined by the following points:

(x_{min}, y_{max})	(x_{max}, y_{max})
(x_{min}, y_{min})	(x_{max}, y_{min})

See Figure 131.24 for an illustration of the bounding rectangle applied to the data of the domain D in Figure 131.19. PROC VARIOGRAM provides you with the sizes of $x_{max} - x_{min}$, $y_{max} - y_{min}$, and h_b. For example, in Figure 131.4 in the preliminary analysis, the specified parameters named "Max Data Distance in East," "Max Data Distance in North," and "Max Data Distance" correspond to the lengths $x_{max} - x_{min}$, $y_{max} - y_{min}$, and h_b, respectively.

Figure 131.24 Bounding Rectangle to Determine Maximum Pairwise Distance in Domain D

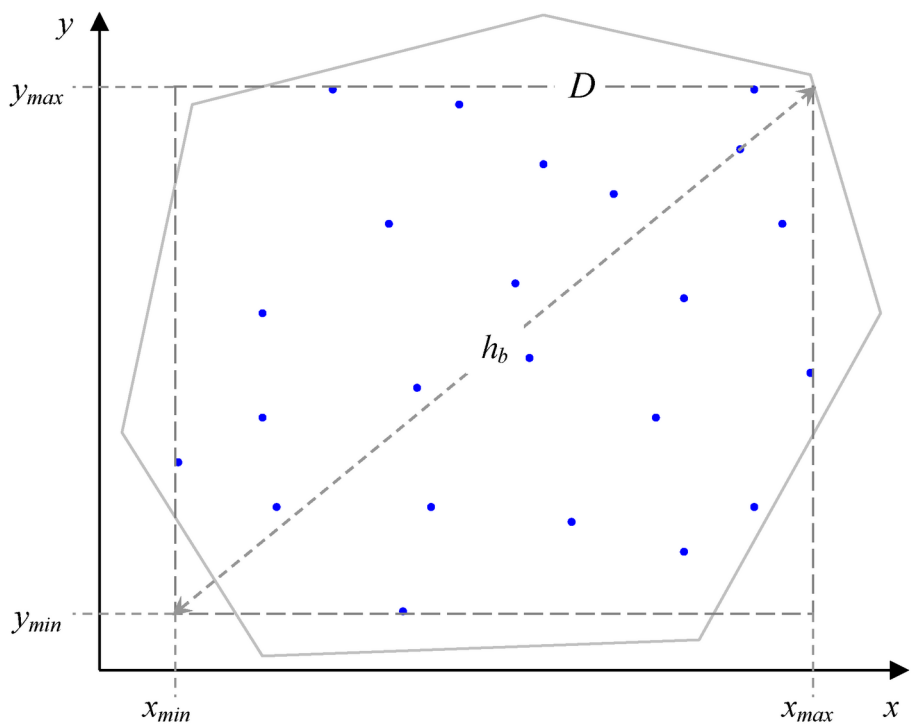

The pairwise distance bound, denoted by h_b, is given by

$$h_b = \sqrt{(x_{max} - x_{min})^2 + (y_{max} - y_{min})^2}$$

Using h_b, the interval $(0, h_b]$ is divided into $N + 1$ subintervals, where N is the value of the NHCLASSES= option specified in the COMPUTE statement, or $N = 10$ (default) if the NHCLASSES= option is not specified. The basic distance unit is $h_0 = \frac{h_b}{N}$; the distance intervals are centered on $h_0, 2h_0, \ldots, Nh_0$, with a distance tolerance of $\pm\frac{h_0}{2}$. The extra subinterval is $(0, h_0/2)$ and corresponds to lag class zero. It is half the length of the remaining subintervals, and it often contains the smallest number of pairs. Figure 131.22 shows an example where the lag classes correspond to $h_0 = 1$. This method of partitioning the interval $(0, h_b]$ is used in the empirical semivariogram computation.

Choosing the Size of Classes

When you start with a data sample, the VARIOGRAM procedure computes all the distinct point pairs in the sample. The OUTPAIR= output data set, described in the section "OUTPAIR=*SAS-data-set*" on page 11042, contains information about these pairs. The point pairs are then categorized in classes. The size of each class depends on the common distance that separates consecutive classes. In PROC VARIOGRAM you need to provide this distance value with the LAGDISTANCE= option. Practically, you can define the distance between classes to be about the size of the average sampling distance (Olea 2006).

Under a more scrutinized approach, before you specify a value for the LAGDISTANCE= option, it is helpful to be aware of two issues. First, estimate how many classes of data pairs you might need. Each class contributes one point to the empirical semivariogram. Therefore, you need enough classes for an adequate number of points, so that your empirical semivariogram can suggest a suitable theoretical model shape for the description of the spatial continuity. Second, keep in mind that a larger number of data pairs in a class can contribute to a more accurate estimate of the corresponding semivariogram point.

The first consideration is a more general issue, and both this and the following subsection address it in detail. Based on the second consideration, the class size problem translates into having a sufficient number of data pairs in each class to produce an accurate semivariance estimate. However, only empirical rules of thumb exist to guide you with this choice. Examples of minimum-pairs empirical rules include the suggestion by Journel and Huijbregts (1978, p. 194) to use at least 30 point pairs for each lag class. Also, in a different approach, Chilès and Delfiner (1999, p. 38) increase this number to 50 point pairs.

Obviously, smaller data samples provide fewer data pairs in the sample. According to Olea (2006), it is difficult to properly estimate a semivariogram with fewer than 50 measurements. The preceding minimum-pairs practical rules are useful in cases where small samples are involved. When you work with a relatively small sample, the key is to specify the value of LAGDISTANCE= such that you can strike a balance between the number of the classes you can form and their pairs count. In the coal seam thickness example of the section "Preliminary Spatial Data Analysis" on page 10962, it is not possible to create a desirable large number of classes and maintain an adequate size for each one. On the other hand, there is no practical need to invoke these rules in the case of the much larger sample of ozone concentrations in "Example 131.2: An Anisotropic Case Study with Surface Trend in the Data" on page 11058.

The spatial distribution of the sample might also affect the grouping of pairs into classes. For example, data that are sampled in clusters might prove difficult to classify according to the preceding practical rules. One strategy to address this problem is to accept fewer than 30 pairs for the underpopulated distance classes. Then, at the stage when you determine what theoretical semivariogram model to use, either disregard the corresponding empirical semivariogram points or use them and accept the increased uncertainty.

The VARIOGRAM procedure can help you decide on a suitable class size before you proceed with the empirical semivariogram computation. First, provide a number for the class count by specifying the NHCLASSES= value. Run the procedure with the option NOVARIOGRAM in the COMPUTE statement and examine the distribution data pairs. Use different values of NHCLASSES= to investigate how this parameter affects the data pairs distribution in each distance class. The pairwise distance intervals table (for example, Figure 131.3) shows the number of pairs in each distance class in the "Number of Pairs" column, and you can use the preceding rule of thumb to adjust the NHCLASSES= value accordingly.

PROC VARIOGRAM displays a rounded value of the distance between the lag bounds as the "Lag Distance" parameter in the pairs information table (see Figure 131.5) or the pairwise distances histogram (see Figure 131.4), which you can use for the LAGDISTANCE= specification. However, this is only one tool. For the semivariogram computation you can specify your own LAGDISTANCE= value based on your experience. Smaller LAGDISTANCE= values result in fewer data pairs in the classes. In that sense, you might find

smaller values useful when you work with large samples so that you obtain more semivariogram points. Also, if the LAGDISTANCE= value is too large, you might end up "wasting" too many point pairs in fewer classes at the expense of computing fewer semivariogram points and no significant accuracy gains in the estimation.

As explained earlier, depending on the sample size and its spatial distribution you might have classes with fewer points than what the practical rules advise. Most commonly, the deficient distance classes are the limiting ones close to the origin $\boldsymbol{h} = 0$ and the most remote ones at large $\boldsymbol{h}$. The classes near the origin correspond to lags 0 and 1. These lags are crucial because the empirical semivariogram in small distances $\boldsymbol{h}$ characterizes the process smoothness and can help you detect the presence of a nugget effect. However, as discussed in the section "Distance Classification" on page 11019, lag zero is half the size of the rest of the classes by definition, so it can be expected to violate the rule of thumb for the number of pairs in a class.

The classes located at higher and extreme distances within a spatial domain are often not accounted for in the empirical semivariogram. The fewer pairs that can be formed in these distances do not allow for an accurate assessment of the spatial correlation, as is explained in the following section.

Spatial Extent of the Empirical Semivariogram

Given your choice for the LAGDISTANCE= value in your spatial domain, the following paragraphs provide guidelines on how many classes to consider when you compute the empirical semivariogram.

Obviously, you want to include no more classes beyond the limit where the pairs count falls below the minimum-pairs empirical rule threshold, as discussed in the preceding subsection. PROC VARIOGRAM provides you with a visual way to inspect this upper limit, if you decide to make use of the minimum-pairs empirical rule. In particular, specify your threshold choice for the minimum pairs per class by using the THRESHOLD= parameter for the PLOTS=PAIRS option.

Then, the procedure produces in the pairwise distances histogram a reference line at the specified THRESHOLD= value, which leaves below the line all lags whose pairs count is lower than the threshold value; see, for example, Figure 131.4. The last lag class whose pair population is above the THRESHOLD= value is reported in the pairs information table as "Highest Lag With Pairs > Threshold." This value is not a recommendation for the MAXLAGS= option, but rather is an upper limit for your choice. Detailed information about the pairs count in each class is displayed in the corresponding pairwise distance intervals table, as Figure 131.3 demonstrates.

The preceding suggests that you have an upper limit indication, but you still need some criterion to decide how many lags to include in the semivariogram estimation. The criterion is the extent of spatial dependence in your domain.

Spatial dependence can exist beyond your domain limits. However, you have no data past your domain scale to define a range for larger-scale spatial dependencies. As you look for pairs of data that are gradually farther apart, the number of pairs naturally decreases with distance. The pairs at the more distant classes might be so few that they are likely to be independent with respect to the spatial dependence scale that you can detect. If you include the largest distances in your empirical semivariogram plot, then these pairs only contribute added noise. In the same sense, you cannot explore in detail spatial dependencies in scales smaller than an average minimum distance between your data. The nugget effect represents then microscale correlations whose effect is evident in your working scale.

You specify the spatial dependence extent with commonly used measures such as the *correlation range* (or *correlation length*) ϵ and the *correlation radius* h_c. Both are defined in a similar manner. The correlation range ϵ is the distance at which the covariance is 5% of its value at $\boldsymbol{h} = 0$, and shows that beyond ϵ the covariance is considered to be negligible. The correlation radius h_c is the distance at which the covariance

is about half the variance at $\boldsymbol{h} = 0$, and indicates the distance over which significant correlations prevail (Christakos 1992, p. 76). The physical meanings of these measures are similar to that of the semivariogram range. Also, the effective range r_ϵ used in asymptotically increasing semivariance models has essentially the same definition as the correlation range ϵ (see the section "Theoretical Semivariogram Models" on page 11008).

A rough estimate of the correlation extent measures might be available from previous studies of a similar site, or from prior information about related measurements. In such an event, you typically want to consider a maximum pairwise distance that does not exceed the length of two or three correlation radii, or one and a half correlation ranges. You can then specify the MAXLAGS= value on the basis of the lags that fit in that distance.

When you have no estimates of correlation extent measures, you can use first use a crude measure to get started with your analysis: you can typically expect MAXLAGS= to be about half of the lag classes shown in the pairwise distances histogram.

Then, if necessary, you can refine your MAXLAGS= choice by using the following maximum lags rule of thumb: Journel and Huijbregts (1978, p. 194) advise considering lags up to about half of the extreme distance between data in the direction of interest. The VARIOGRAM procedure assists you in this task by providing the overall extreme data distance h_b, in addition to the extreme data distances in the vertical and horizontal axes directions. For example, h_b is reported in the pairs information table as "Maximum Data Distance" (see Figure 131.5), and in the pairwise distances histogram as "Max Data Distance" (see Figure 131.4).

Overall, avoid significant deviations from the maximum lags rule of thumb. As was stated earlier, a MAXLAGS= value that takes you well beyond the half-extreme distance between data in a given direction might give you limited accuracy in the empirical semivariance estimates at higher distances. At the other end, a value of MAXLAGS= that is too small might lead you to omit important information about the spatial structure that potentially lies within the range of distances you skipped.

Semivariance Computation

With the classification of a point pair $P_i P_j$ into an angle/distance class, as shown earlier in this section, the semivariance computation proceeds as follows.

Denote all pairs that $P_i P_j$ belong to angle class $[\theta_k - \delta\theta_k, \theta_k + \delta\theta_k)$ and distance class $L = L(P_i P_j)$ as $N(\theta_k, L)$. For example, based on Figure 131.20 and Figure 131.21, $P_1 P_2$ belongs to $N(60°, 1)$.

Let $\mid N(\theta_k, L) \mid$ denote the *number* of such pairs. The component of the standard (or method of moments) semivariance that correspond to angle/distance class $N(\theta_k, L)$ is given by

$$\hat{\gamma}(h_k) = \frac{1}{2 \mid N(\theta_k, L) \mid} \sum_{P_i P_j \in N(\theta_k, L)} [V(\boldsymbol{s}_i) - V(\boldsymbol{s}_j)]^2$$

where h_k is the average distance in class $N(\theta_k, L)$; that is,

$$h_k = \frac{1}{\mid N(\theta_k, L) \mid} \sum_{P_i P_j \in N(\theta_k, L)} \mid P_i P_j \mid$$

The robust version of the semivariance is given by

$$\bar{\gamma}(h_k) = \frac{\Psi^4(h_k)}{2[0.457 + 0.494/N(\theta_k, L)]}$$

where

$$\Psi(h_k) = \frac{1}{N(\theta_k, L)} \sum_{P_i P_j \in N(\theta_k, L)} [V(\boldsymbol{s}_i) - V(\boldsymbol{s}_j)]^{\frac{1}{2}}$$

This robust version of the semivariance is computed when you specify the ROBUST option in the COMPUTE statement in PROC VARIOGRAM.

PROC VARIOGRAM computes and writes to the OUTVAR= data set the quantities h_k, θ_k, L, $N(\theta_k, L)$, $\hat{\gamma}(h)$, and $\bar{\gamma}(h)$.

Empirical Semivariograms and Surface Trends

It was stressed in the beginning of the section "Theoretical and Computational Details of the Semivariogram" on page 11013 that if your data are not free of nonrandom surface trends, then the empirical semivariance $\hat{\gamma}_z(\boldsymbol{h})$ you obtain from PROC VARIOGRAM represents a pseudo-semivariance rather than an estimate of the theoretical semivariance $\gamma_z(\boldsymbol{h})$.

In practice, two major difficulties appear. First, you might have no knowledge of underlying surface trends in your SRF $Z(\boldsymbol{s})$. It can be possible to have this information when you deal with a repetitive phenomenon (Chilès and Delfiner 1999, p. 123), or if you work within a subdomain of a broader region with known characteristics; often, though, this is not the case. Second, even if you suspect the existence of an underlying nonrandom trend, its precise nature might be unknown (Cressie 1993, p. 114, 162).

Based on the last remark, the criteria to define the exact form of a surface trend can be subjective. However, statistical methods can identify the presence and remove an estimate of such a trend. Different trend forms can be estimated in your SRF depending on the trend estimation model that you choose. This choice can lead to different degrees of smoothing in the residual random fluctuations. It might also have an effect on the residuals spatial structure characterization, because trend removals with different models are essentially different operations acting upon the values of your original observations. Following the comment by Chilès and Delfiner (1999, section 2.7.3), there are as many semivariograms of residuals as there are ways of estimating the trend. The same source also examines the introduction of bias in the semivariance of the residuals as a side effect of trend removal processes. This bias is small when you examine distances close to the origin $\boldsymbol{h} = 0$, and it can increase with distance.

Keeping in mind the preceding remarks, an approach you can take is to use one of the many predictive modeling tools in SAS/STAT software to estimate the unknown trend. Then you use PROC VARIOGRAM to analyze the residuals after you remove the trend. If the resulting model does not require too many degrees of freedom (such as if you use a low-order polynomial), then this approach might be sufficient. The section "Analysis with Surface Trend Removal" on page 11062 demonstrates how to use PROC GLM (see Chapter 53, "The GLM Procedure") for that purpose.

Apart from the standard semivariogram analysis, you can attempt to fit a theoretical semivariogram model to your empirical semivariogram if (a) either the analysis itself or your knowledge of the SRF does not clearly suggest the presence of any surface trend, or (b) the analysis can indicate a potentially trend-free direction, along which your data have a constant mean.

For example, you might observe overall similar values in your data. This can be an indication that your data are free of nonrandom trends, or that a very mild trend is present. The case falls under the preceding option (a). A very mild trend still allows a good determination of the semivariance at short distances according to Chilès and Delfiner (1999, p. 125), and this can be sufficient for your spatial prediction goal. An analysis of this type is assumed in the section "Preliminary Spatial Data Analysis" on page 10962.

If you observe similar values locally across a particular direction, this an instance of option (b). Olea (2006) suggests recognizing a trend-free direction as being perpendicular to the axis of the maximum dip in the values of $Z(\boldsymbol{s})$. If you suspect that at least one such direction exists for your data, then run PROC VARIOGRAM for a series of directions in the angular vicinity. The trend-free direction, if it exists, coincides with the one whose pseudo-semivariogram exhibits minimal increase with distance; see "Example 131.3: Analysis without Surface Trend Removal" on page 11071 for a demonstration of this approach. However, you cannot test $Z(\boldsymbol{s})$ for anisotropy in this case, because you can investigate the semivariogram only in the single trend-free direction (Olea 1999, p. 76). Chilès and Delfiner (1999, section 2.7.4) suggest fitting a theoretical model in a trend-free direction only if the hypothesis of an isotropic semivariogram appears reasonable in your analysis.

As a result, you need to be very cautious when you choose to perform semivariogram analysis on data you have not previously examined for surface trends. In this event, both of the options (a) and (b) that were reviewed in the preceding paragraphs rely mostly on empirical and subjective criteria. As noted in this section, a degree of subjectivity exists in the selection of the surface trend itself. This fact suggests that a significant part of the semivariogram analysis is based on metastatistical decisions and on your understanding of your data and the physical considerations that govern your study. In any case, as shown in the section "Theoretical and Computational Details of the Semivariogram" on page 11013, your semivariogram analysis relies fundamentally on the use of trend-free data.

Theoretical Semivariogram Model Fitting

You can choose between two approaches to select a theoretical semivariogram model and fit the empirical semivariance. The first one is manual fitting, in which a theoretical semivariogram model is selected based on visual inspection of the empirical semivariogram. For example, see Hohn (1988, p. 25) and comments from defendants of this approach in Olea (1999, p. 82). The second approach is to perform model fitting in an automated manner. For this task you can use methods such as least squares, maximum likelihood, and robust methods (Cressie 1993, section 2.6).

The VARIOGRAM procedure features automated semivariogram model fitting that uses the weighted least squares (WLS) or the ordinary least squares (OLS) method. Use the MODEL statement to request that specific model forms or an array of candidate models be tested for optimal fitting to the empirical semivariance.

Assume that you compute first the empirical semivariance $\gamma_z^*(\boldsymbol{h})$ at MAXLAGS=k distance classes, where $\gamma_z^*(\boldsymbol{h})$ can be either the classical estimate $\hat{\gamma}_z(\boldsymbol{h})$ or the robust estimate $\bar{\gamma}_z(\boldsymbol{h})$, as shown in the section "Theoretical and Computational Details of the Semivariogram" on page 11013. In fitting based on least squares, you want to estimate the parameters vector $\boldsymbol{\theta}$ of the theoretical semivariance $\gamma_z(\boldsymbol{h})$ that minimizes the sum of square differences $R(\boldsymbol{\theta})$ given by the expression

$$R(\boldsymbol{\theta}) = \sum_{i=1}^{k} {w_i}^2 \left[\gamma_z^*(\boldsymbol{h}_i) - \gamma_z(\boldsymbol{h}_i;\boldsymbol{\theta})\right]^2$$

For $i = 1, \ldots, k$, the weights are ${w_i}^2 = 1/\mathrm{Var}[\gamma_z^*(\boldsymbol{h}_i)]$ in the case of WLS and ${w_i}^2 = 1$ in the case of OLS. Therefore, the parameters $\boldsymbol{\theta}$ are estimated in OLS by minimizing

$$R(\boldsymbol{\theta})_{OLS} = \sum_{i=1}^{k} \left[\gamma_z^*(\boldsymbol{h}_i) - \gamma_z(\boldsymbol{h}_i;\boldsymbol{\theta})\right]^2$$

For WLS, Cressie (1985) investigated approximations for the variance of both the classical and robust empirical semivariances. Then, under the assumptions of normally distributed observations and uncorrelated squared differences in the empirical semivariance, the approximate weighted least squares estimate of the parameters $\boldsymbol{\theta}$ can be obtained by minimizing

$$R(\boldsymbol{\theta})_{WLS} = \frac{1}{2}\sum_{i=1}^{k} N(\boldsymbol{h}_i)\left[\frac{\gamma_z^*(\boldsymbol{h}_i)}{\gamma_z(\boldsymbol{h}_i;\boldsymbol{\theta})} - 1\right]^2$$

where $N(\boldsymbol{h}_i)$ is the number of pairs of points in the ith distance lag.

PROC VARIOGRAM relies on nonlinear optimization to minimize the least squares objective function $R(\boldsymbol{\theta})$. The outcome is the model that best fits the empirical semivariogram according to your criteria. The fitting process flow is displayed in Figure 131.25. Goovaerts (1997, section 4.2.4) suggests that fitting a theoretical model should aim to capture the major spatial features. An accurate fit is desirable, but overfitting does not offer advantages, because you might find yourself trying to model possibly spurious details of the empirical semivariogram. At the same time, it is important to describe the correlation behavior accurately near the semivariogram origin. As pointed out by Chilès and Delfiner (1999, pp. 104–105), a poor description of spatial continuity at small lags can lead to loss of optimality in kriging predictions and erroneous reproduction of the variability in conditional simulations.

The significance of achieving better accuracy near the semivariogram origin is an advantage of the WLS method compared to OLS. In particular, the semivariance variance decreases when you get closer the origin $\boldsymbol{h} = 0$, as suggested in the section "Theoretical and Computational Details of the Semivariogram" on page 11013. The WLS weights are expressed as the inverse of this variance; as a result, WLS fitting is more accurate for distances $\boldsymbol{h}$ near the semivariogram origin. In contrast, the OLS approach performs a least squares overall best fit because it assumes constant variance at all distances $\boldsymbol{h}$. Another advantage of WLS over OLS is that OLS falsely assumes that the differences in the optimization process are normally distributed and independent. However, WLS has the disadvantage that the weights depend on the fitting parameters.

Depending on your application, you can use WLS or OLS with PROC VARIOGRAM to fit classical semivariance. Other fitting methods include maximum likelihood approaches that rely crucially on the normality assumption for the data distribution, and the generalized least squares method that offers better accuracy but is computationally more demanding. You can find extensive discussions about these issues in Cressie (1993, section 2.3), Jian, Olea, and Yu (1996), Stein (1988), and Schabenberger and Gotway (2005).

The sections "Parameter Initialization" on page 11029 and "Quality of Fit" on page 11031 provide details and insight about semivariogram fitting, in addition to ways to cope with poor fits or no fit at all. These strategies can help you reach a meaningful description of spatial correlation in your problem.

Figure 131.25 Semivariogram Fitting Process Flowchart

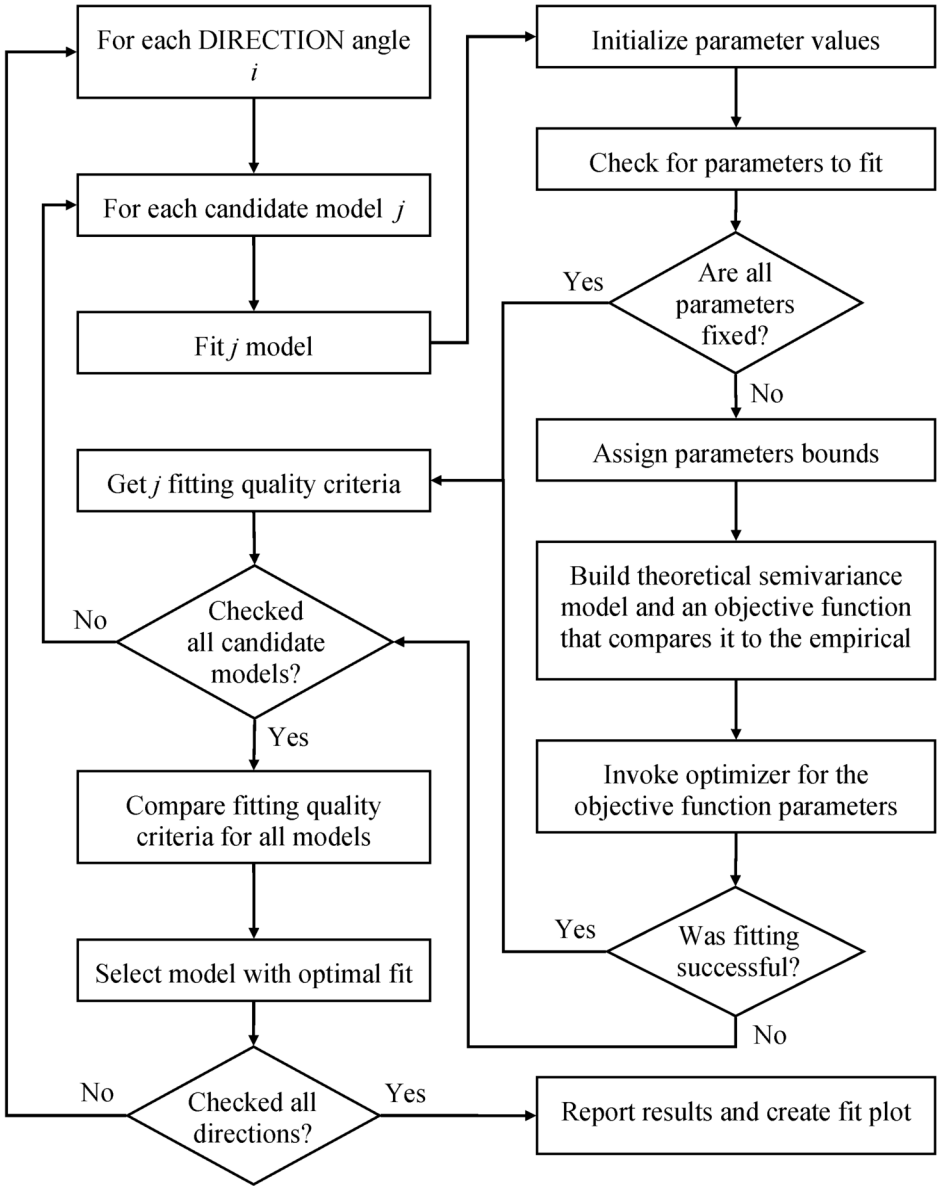

Parameter Initialization

An important stage when you prepare for the model fitting process is initialization of the model parameters. As stated earlier, nonlinear optimization techniques are used in the fitting process. These techniques assist in the estimation of the model parameters, and being nonlinear means they can be very sensitive to selection of the initial values.

You can specify initial values close to the expected estimates when you have a relatively simple problem, such as in the example of the section "Getting Started: VARIOGRAM Procedure" on page 10962. In the case of nested models the selection of initial values can be more challenging because you have to assess the level of contribution for each one of the nested components.

The VARIOGRAM procedure features automatic selection of initial values based on the recommendations in Jian, Olea, and Yu (1996). Specifically, if you compute the estimated empirical semivariogram $\hat{\gamma}_z(\boldsymbol{h})$ at k lags, then:

- The default initial nugget effect $c_{n,0}$ is

$$c_{n,0} = \text{Max}\left[0, \hat{\gamma}_z(\boldsymbol{h}_1) - \frac{\boldsymbol{h}_1}{\boldsymbol{h}_2 - \boldsymbol{h}_1}\left[\hat{\gamma}_z(\boldsymbol{h}_2) - \hat{\gamma}_z(\boldsymbol{h}_1)\right]\right]$$

- The default initial slope ${\sigma_{0,0}}^2$ and initial exponent $a_{0,0}$ for the power model are

$${\sigma_{0,0}}^2 = \frac{\left[\hat{\gamma}_z(\boldsymbol{h}_{k-2}) + \hat{\gamma}_z(\boldsymbol{h}_{k-1}) + \hat{\gamma}_z(\boldsymbol{h}_k)\right]/3 - c_{n,0}}{\boldsymbol{h}_k - \boldsymbol{h}_1}$$

$$a_{0,0} = 1$$

- The default initial scale ${\sigma_{0,0}}^2$ and initial range $a_{0,0}$ for all other models are

$${\sigma_{0,0}}^2 = \frac{\left[\hat{\gamma}_z(\boldsymbol{h}_{k-2}) + \hat{\gamma}_z(\boldsymbol{h}_{k-1}) + \hat{\gamma}_z(\boldsymbol{h}_k)\right]}{3} - c_{n,0}$$

$$a_{0,0} = 0.5\boldsymbol{h}_k$$

When you use the Matérn form, PROC VARIOGRAM sets the default initial value for the Matérn smoothness to $\nu_0 = 1$.

These rules are observed in the case of single, non-nested model fitting, and they are slightly modified to apply for nested model fitting as follows: Assume that you want to fit a nested model composed of m structures. As stated in the section "Nested Models" on page 11013, the nugget effect is a single parameter and is independent of the number of nested structures in a model. Also, the sum of the nested structure scales and the nugget effect, if any, must be equal to the total variance. For this reason, PROC VARIOGRAM simply divides the initial scale value it would assign to a non-nested model into m components ${\sigma_{0,0,1}}^2, \ldots, {\sigma_{0,0,m}}^2$. For the range parameter, the VARIOGRAM procedure sets the initial range $a_{0,0,1}$ of the first nested structure equal to the value it would assign to a non-nested model initial range. Then, the initial range $a_{0,0,m}$ of the m-component is set recursively to half the value of the initial range $a_{0,0,m-1}$ of the $(m-1)$-component.

Your empirical semivariogram must have nonmissing estimates at least at three lags so that you can use the automated fitting feature in PROC VARIOGRAM. Overall, if you specify a model form with q parameters to fit to an empirical semivariogram with nonmissing estimates at k lags, then the fitting problem is well-defined only when the degrees of freedom are $\text{DF} = k - q \geq 0$.

A potential numerical issue is that fitting could momentarily lead the fitting parameters to near-zero semivariance values at lags away from zero distance. The theoretical semivariance is always positive for any distance larger than zero, and this is also a requirement for the numerical computation of $R(\boldsymbol{\theta})_{WLS}$ in weighted least squares fitting. Such numerical issues are unlikely but possible, depending on the data set you use and the parameter initial values. If an event of nonpositive semivariance at a given lag occurs during an iteration, then PROC VARIOGRAM transparently adds a minimal amount of variance at that lag for the specific iteration. You can control this amount of variance with the NEPSILON= option of the MODEL statement. It is recommended that you leave this parameter at its default value.

The section concludes with a reminder of the fitting process sensitivity to the initial parameter values selection. The VARIOGRAM procedure facilitates this selection for you by using the simple rules shown

earlier. However, the suggested initial values might not always be the best choice. In simple cases, such as the introductory example in the section "Getting Started: VARIOGRAM Procedure" on page 10962, this approach is very convenient and effective.

In principle, you are strongly encouraged to experiment with initial values. You want to make sure that the fitting process leads the model parameters to converge to estimates that make sense for your problem. When a parameter estimate seems unreasonable on the basis of your problem specification (for example, a model scale might be estimated to be 10 times the size of your sample variance, or the estimate of a range might be zero), PROC VARIOGRAM produces a note to let you know about a potentially ambiguous fit. These issues are examined in more detail in the section "Quality of Fit" on page 11031.

Parameter Estimates

When the fit process is complete, the VARIOGRAM procedure produces the "Parameter Estimates" table with information about the fitted model parameters. The table includes estimates of the parameters, their approximate standard error, the statistical degrees of freedom DF, the corresponding t statistic, and its approximate p-value. For a model with q parameters that fits an empirical semivariogram of k nonmissing lags, $\mathrm{DF} = k - q$.

NOTE: Parameter estimates might have nonzero standard errors even in the rather extreme case where $\mathrm{DF} = 0$. This can typically occur when there are active optimization constraints in the fitting process.

You can request the confidence intervals for the parameter estimates of a fitted model by specifying the CL option of the MODEL statement. These confidence intervals are computed using the Wald-based formula

$$\widehat{\beta}_i \pm \mathrm{stderr}_i \times t(k-q, 1-\alpha/2)$$

where $\widehat{\beta}_i$ is the ith parameter estimate, stderr_i is its estimated approximate standard error, $t(k-q, 1-\alpha/2)$ is the t statistic with $\mathrm{DF} = k - q$ degrees of freedom. The confidence intervals are only asymptotically valid. The significance level α used in the construction of these confidence limits can be set with the ALPHA= option of the MODEL statement; the default value is $\alpha = 0.05$.

Specify the COVB and the CORRB options in the MODEL statement to request the approximate covariance and approximate correlation matrices of the fitted parameters, respectively. These matrices are based on the optimization process results. In agreement with reporting similar optimization output in SAS/STAT software, parameters with active restraints have zeros in the corresponding rows and columns in the covariance and correlation matrices, and display 1 in the correlation matrix diagonal.

Quality of Fit

The VARIOGRAM procedure produces a fit summary table to report about the goodness of fit. When you specify multiple models to fit with the FORM=AUTO option in the MODEL statement, the VARIOGRAM procedure uses two processes to rank the fitted models: The first one depends on your choice among available fitting criteria. The second one is based on an operational classification of equivalent models in classes. The two processes are described in more detail in the following subsections.

Overall, no absolutely correct way exists to rank and classify multiple models. Your choice of ranking criteria could depend on your study specifications, physical considerations, or even your personal assessment of fitting performance. The VARIOGRAM procedure provides you with fitting and comparison features to facilitate and help you better understand the fitting process.

Fitting Criteria

The fit summary table ranks multiple models on the basis of one or more fitting criteria that you can specify with the CHOOSE= option of the MODEL statement, as explained in the section "Syntax: VARIOGRAM Procedure" on page 10975. Currently, the VARIOGRAM procedure offers two numerical criteria (for which a smaller value indicates a better fit) and a qualitative criterion:

- The residual sum of squares error (SSE) is based on the objective function of the fitting process. When the specified method is weighted least squares, the sum of squares of the weighted differences (WSSE) is computed according to the expression

  $$\mathrm{WSSE} = \sum_{i=1}^{k} {w_i}^2 \left[\gamma_z^*(\boldsymbol{h}_i) - \gamma_z(\boldsymbol{h}_i;\boldsymbol{\theta})\right]^2$$

  where $\gamma_z^*(\boldsymbol{h}_i)$ can be either the classical or robust semivariance estimate of the theoretical semivariance $\gamma_z(\boldsymbol{h}_i;\boldsymbol{\theta})$ at the ith lag and the weights ${w_i}^2$ are taken at lags $i = 1,\ldots,k$. When you specify the METHOD=OLS option in the MODEL, the weights ${w_i}^2 = 1$ for $i = 1,\ldots,k$, and the SSE is expressed as

  $$\mathrm{SSE} = \sum_{i=1}^{k} \left[\gamma_z^*(\boldsymbol{h}_i) - \gamma_z(\boldsymbol{h}_i;\boldsymbol{\theta})\right]^2$$

- Akaike's information criterion (AIC) is included in the fit summary table when there is at least one nonfixed parameter. In its strict definition, AIC assumes that the model errors are normally and independently distributed. This assumption is not correct in the semivariance fitting analysis. However, the AIC can be also defined in an operational manner on the basis of the weighted squared error sum WSSE as

  $$\mathrm{AIC} = k \ln\left(\frac{\mathrm{WSSE}}{k}\right) + 2q$$

  for k lags and q model parameters; see, for example, Olea (1999, p. 84). The operational definition of the AIC is provided as an additional criterion for the comparison of fitted models in PROC VARIOGRAM.

  The AIC expression suggests that when you specify multiple models with the FORM=AUTO option in the MODEL statement, all models with the same number of parameters are ranked in the same way by the AIC and the WSSE criteria. Among models with the same WSSE value, AIC ranks higher the ones with fewer parameters.

- The third qualitative criterion enables you to classify multiple models based on their convergence status. A model is sent to the bottom of the ranking table if the parameter estimation optimization fails to converge or fitting is unsuccessful due to any other issue. These two cases are distinguished by the different notes they produce in the fit summary ODS table. If you specify the STORE statement to save the fitting output in an item store, then models that have failed to fit are not passed to the item store.

With respect to convergence status, PROC VARIOGRAM ranks higher those models that have successfully completed the fitting process. It might occur that the selection of parameter initial values, physical considerations about the forms that are used for the fit, or numerical aspects of the nonlinear optimization could result in ambiguous fits. For example, you might see that model parameters converge at or near their boundary

values, or that parameters have unreasonably high estimates when compared to the empirical semivariogram characteristics. Then, the fit summary table designates such fits as questionable.

You might not need to take any action if you are satisfied with the fitting results and the selected model. You can investigate questionable fits in one or more of the following ways:

- If a form in a nested model makes no contribution to the model due to a parameter at or near its boundary value, then you could have a case of a degenerate fit. When you fit multiple models, a model with degenerate fit can collapse to the more simple model that does not include the noncontributing form. The VARIOGRAM procedure includes in its fit summary all models that are successfully fit. In such cases you can ignore degenerate fits. You can also try subsequent fits of individual models and exclude noncontributing forms or use different initial values.
- Unreasonably low or high parameter estimates might be an indication that the current initial values are not a good guess for the nonlinear optimizer. In most cases, fitting an empirical semivariogram gives you the advantage of a fair understanding about the value range of your parameters. Then, you can use the PARMS statement to specify a different set of initial values and try the fit again.
- Try replacing the problematic form with another one. A clear example is that you can expect a very poor fit if you specify an exponential model to fit an empirical semivariogram that suggests linear behavior.

Eventually, if none of the aforementioned issues exist, then a model is ranked in the highest positions of the fit summary table. You can combine two or more of the fitting criteria to manage classification of multiple fitted models in a more detailed manner.

In some cases you might still experience a poor quality of fit or no fit at all. If none of the earlier suggestions results in a satisfactory fit, then you could decide to re-estimate the empirical semivariogram for your same input data. The following actions can produce different empirical semivariograms to fit a theoretical model to:

- If you compute the semivariogram for different angles and you experience optimization failures, try specifying explicitly the same direction angles with different tolerance or bandwidth value in the DIRECTIONS statement.
- Modify slightly the LAGDISTANCE= option in the COMPUTE statement to obtain a different empirical semivariogram.

Finally, it is possible to have models in the fitting summary table ranked in a way that seemingly contradicts the specification in the CHOOSE= option of the MODEL statement. Consider an example with the default behavior CHOOSE=(SSE AIC), where you might observe that models have the same SSE values but are not ranked further as expected by the AIC criterion. A closer examination of such cases typically reduces this issue to a matter of the accuracy shown in the table. That is, the displayed accuracy of the SSE values might hide additional decimal digits that justify the given ranking.

In such scenarios, discrimination of models at the limits of numerical accuracy might suggest that you choose a model of questionable fit or a nested structure over a more simple one. You can then review the candidate models and exercise your judgment to select the model that works best for you. If all values of a criterion are equal, then the ranking order is simply the order in which models are examined unless more criteria follow that can affect the ranking.

Classes of Equivalence

The fit summary that is produced after fitting multiple models further categorizes the ranked models in classes of equivalence. Equivalence classification is an additional investigation that is unrelated to the ranking criteria presented in the previous subsection; it is an operational criterion that provides you with a qualitative overview of multiple model fit performance under given fitting conditions.

To examine model equivalence, the VARIOGRAM procedure computes the semivariances for each one of the fitted models at a set of distances. For any pair of consecutively ranked models, if the sum of their semivariance absolute differences at all designated distances is smaller than the tolerance specified by the EQUIVTOL= parameter, then the two models are deemed equivalent and placed in the same class; otherwise, they are placed in different classes. Equivalence classification depends on the existing ranking; hence the resulting classes can differ when you specify different ranking criteria in the CHOOSE= option of the MODEL statement.

The equivalence class numbers start at 1 for the top-ranked model in the fit summary table. You can consider the top model of each equivalence class to be a representative of the class behavior. When you specify that fit plots be produced and there are equivalence classes, the plot displays the equivalence classes and the legend designates each one by its representative model.

Consequently, if an equivalence class contains multiple members after a fit, then all of its members produce in general the exact same semivariogram. A typical reason could be that the fitting process estimates of scale parameters are at or close to their zero boundaries in one or more nested forms in a model. In such cases, the behavior of this model reduces to the behavior of its nested components with nonzero parameters. When one or more models share this situation or have the same contributing nested forms, they could end up as members of the same equivalence class depending on the ranking criteria.

It is not necessary for all models in the same equivalence class to produce the exact same semivariogram. If a fit of two obviously different forms involves semivariance values that are small enough for the equivalence criterion to be satisfied by the default value of the EQUIVTOL= option, then you might need to specify an even smaller value in the EQUIVTOL= option to rank these two models in separate equivalence classes.

Fitting with Matérn Forms

When you use a Matérn form in the fitting process, it is possible that the fitting optimizer might encounter numerical difficulties if it tries to push the smoothing parameter ν towards increasingly high values. The VARIOGRAM procedure addresses this issue by imposing an amply elevated upper bound of 1,000,000 on the smoothness values it processes. The section "Characteristics of Semivariogram Models" on page 11009 mentions that $\nu \to \infty$ gives the Gaussian model. In the scenario of progressively increasing smoothness values, PROC VARIOGRAM acknowledges that the Matérn form behavior tends asymptotically to become Gaussian and replaces automatically the Matérn with a Gaussian form in the model. Subsequently, fitting resumes with the resulting model.

If you explore fitting of multiple models, then any duplicate models that might occur due to Matérn-to-Gaussian form conversions are fitted only once. Also, if a nested model has more than one Matérn form, then the fitting process checks one of them at a time about whether they need to be replaced by a Gaussian form. Consequently, following the switch of one Matérn form, the fitting process starts anew with the resulting model before any decisions for additional form conversions are made.

Replacement of the Matérn form with the Gaussian form occurs by default when $\nu > 10{,}000$. However, you can control this threshold value with the MTOGTOL= parameter of the MODEL statement. Practically, the Matérn form starts to resemble the Gaussian behavior for ν values that are about $\nu > 10$. If you encounter

such conversions of the Matérn form into Gaussian and you prefer to set a lower ν threshold for the conversion than the default, you might experience improved code performance because computation of the Matérn semivariance can be numerically demanding.

Autocorrelation Statistics

Spatial autocorrelation measures offer you additional insight into the interdependence of spatial data. These measures quantify the correlation of an SRF $Z(\boldsymbol{s})$ with itself at different locations, and they can be very useful whether you have information at exact locations (point-referenced data) or measurements that characterize an area type such as counties, census tracts, zip codes, and so on (areal data).

As in the semivariogram computation, a key issue for the autocorrelation statistics is that you work with a set z_i of measurements, $i = 1, \ldots, n$, that are free of nonrandom surface trends and have a constant mean.

Autocorrelation Weights

In general, the choice of a weighting scheme is subjective. You can obtain different results by using different schemes, options, and parameters. PROC VARIOGRAM offers you considerable flexibility in choosing weights that are appropriate for prior considerations such as different hypotheses about neighboring areas, definition of the neighborhood structure, and accounting for natural barriers or other spatial characteristics; see the discussion in Cliff and Ord (1981, p. 17). As stressed for all types of spatial analysis, it is important to have good knowledge of your data. In the autocorrelation statistics, this knowledge can help you avoid spurious correlations when you choose the weights.

The starting point is to assign individual weights to each one of the n data values z_i, $i = 1, \ldots, n$, with respect to the rest. An $n \times n$ matrix of weights is thus defined, such that for any two locations $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$, the weight w_{ij} denotes the effect of the value z_i at location $\boldsymbol{s}_i$ on the value z_j at location $\boldsymbol{s}_j$. Depending on the nature of your study, the weights w_{ij} need not be symmetric; that is, it can be true that $w_{ij} \neq w_{ji}$.

Binary and Nonbinary Weights

The weights w_{ij} can be either binary or nonbinary values. Binary values of 1 or 0 are assigned if the SRF $Z(\boldsymbol{s}_i)$ at one location $\boldsymbol{s}_i$ is deemed to be connected or not, respectively, to its value $Z(\boldsymbol{s}_j)$ at another location $\boldsymbol{s}_j$. Nonbinary values can be used in the presence of more refined measures of connectivity between any two data points P_i and P_j. PROC VARIOGRAM offers a choice between a binary and a distance-based nonbinary weighting scheme.

In the binary weighting scheme the weight $w_{ij} = 1$ if the data pair at $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ is closer than the user-defined distance that is defined by the LAGDISTANCE= option, and $w_{ij} = 0$ if $i = j$ or in any other case. For that reason, in the COMPUTE statement, if you specify the WEIGHTS=BINARY suboption of the AUTOCORRELATION option when the NOVARIOGRAM option is also specified, then you must also specify the LAGDISTANCE= option.

The nonbinary weighting scheme is based on the pair distances and is invoked with the WEIGHTS=DISTANCE suboption of the AUTOCORRELATION option. PROC VARIOGRAM uses a variation of the Pareto form functional to set the weights. Namely, the autocorrelation weight for every point pair P_i and P_j located at $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$, respectively, is defined as

$$w_{ij} = s\frac{1}{1+ \mid \boldsymbol{h} \mid^p}$$

where $\boldsymbol{h} = \boldsymbol{s}_i - \boldsymbol{s}_j$ and $p \geq 0$ and $s \geq 0$ are user-defined parameters for the adjustment of the weights.

In particular, the power parameter p is specified in the POWER= option of the DISTANCE suboption within the AUTOCORRELATION option. The default value for this parameter is $p = 1$. Also, the scaling parameter s is specified by the SCALE= option in the DISTANCE suboption of the AUTOCORRELATION option. The default value for the scaling parameter is $s = 1$. You can use the p and s parameters to adjust the actual values of the weights according to your needs. Variations in the scaling parameter s do not affect the computed values of the Moran's I and Geary's c autocorrelation coefficients that are introduced in the section "Autocorrelation Statistics Types" on page 11036.

Nonbinary Weights with Normalized Distances

PROC VARIOGRAM offers additional flexibility in the DISTANCE weighting scheme through an option to use normalized pair distances. You can invoke this feature by specifying the NORMALIZE option in the DISTANCE suboption of the AUTOCORRELATION option. In this case, the distances used in the definition of the weights are normalized by the maximum pairwise distance h_b (see the section "Computation of the Distribution Distance Classes" on page 11021 and Figure 131.24); the weights are then defined as $w_{ij} = s/[1 + (|\boldsymbol{h}|/h_b)^p]$.

Most likely, h_b has a different value for different data sets. Hence, it is suggested that you avoid using the weights you obtain from the preceding equation and one data set for comparisons with the weights you derive from different data sets.

Symmetric and Asymmetric Weights

The weighting schemes presented in the preceding paragraphs are symmetric; that is, $w_{ij} = w_{ji}$ for every data pair at locations $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$. However, you can also define asymmetric weights w'_{ij} such that

$$\sum_{j \in J} w'_{ij} = 1$$

for $i = 1, 2, \ldots, n$, where $w'_{ij} = w_{ij} / \sum_{j \in J} w_{ij}$, $i = 1, 2, \ldots, n$. In the distance-based scheme, J is the set of all locations that form point pairs with the point at $\boldsymbol{s}_i$. In the binary scheme, J is the set of the locations that are connected to $\boldsymbol{s}_i$ based on your selection of the LAGDISTANCE= option; see Cliff and Ord (1981, p. 18). The weights w'_{ij} are *row-averaged* (or *standardized* by the count of their connected neighbors). You can apply row averaging in weights when you specify the ROWAVG option within either the BINARY or DISTANCE suboptions in the AUTOCORRELATION option.

Autocorrelation Statistics Types

One measure of spatial autocorrelation provided by PROC VARIOGRAM is Moran's I statistic, which was introduced by Moran (1950) and is defined as

$$I = \frac{n}{(n-1)S^2 W} \sum_i \sum_j w_{ij} v_i v_j$$

where $S^2 = (n-1)^{-1} \sum_i v_i^2$, and $W = \sum_i \sum_{j \neq i} w_{ij}$.

Another measure of spatial autocorrelation in PROC VARIOGRAM is Geary's c statistic (Geary 1954), defined as

$$c = \frac{1}{2S^2 W} \sum_i \sum_j w_{ij} (z_i - z_j)^2$$

These expressions indicate that Moran's I coefficient makes use of the centered variable, whereas the Geary's c expression uses the noncentered values in the summation.

Inference on these two statistic types comes from approximate tests based on the asymptotic distribution of I and c, which both tend to a normal distribution as n increases. To this end, PROC VARIOGRAM calculates the means and variances of I and c. The outcome depends on the assumption made regarding the distribution $Z(\boldsymbol{s})$. In particular, you can choose to investigate any of the statistics under the *normality* (also known as *Gaussianity*) or the *randomization* assumption. Cliff and Ord (1981) provided the equations for the means and variances of the I and c distributions, as described in the following.

The normality assumption asserts that the random field $Z(\boldsymbol{s})$ follows a normal distribution of constant mean ($\bar{Z}$) and variance, from which the z_i values are drawn. In this case, the I statistics yield

$$\mathrm{E}_g[I] = -\frac{1}{n-1}$$

and

$$\mathrm{E}_g[I^2] = \frac{1}{(n+1)(n-1)W^2}(n^2 S_1 - nS_2 + 3W^2)$$

where $S_1 = 0.5\sum_i \sum_{j \neq i}(w_{ij} + w_{ji})^2$ and $S_2 = \sum_i(\sum_j w_{ij} + \sum_j w_{ji})^2$. The corresponding moments for the c statistics are

$$\mathrm{E}_g[c] = 1$$

and

$$\mathrm{Var}_g[c] = \frac{(2S_1 + S_2)(n-1) - 4W^2}{2(n+1)W^2}$$

According to the randomization assumption, the I and c observations are considered in relation to all the different values that I and c could take, respectively, if the n z_i values were repeatedly randomly permuted around the domain D. The moments for the I statistics are now

$$\mathrm{E}_r[I] = -\frac{1}{n-1}$$

and

$$\mathrm{E}_r[I^2] = \frac{A_1 + A_2}{(n-1)(n-2)(n-3)W^2}$$

where $A_1 = n[(n^2 - 3n + 3)S_1 - nS_2 + 3W^2]$, $A_2 = -b_2[n(n-1)S_1 - 2nS_2 + 6W^2]$. The factor $b_2 = m_4/(m_2{}^2)$ is the coefficient of kurtosis that uses the sample moments $m_k = \frac{1}{n}\sum_i v_i^k$ for $k = 2, 4$. Finally, the c statistics under the randomization assumption are given by

$$\mathrm{E}_r[c] = 1$$

and

$$\mathrm{Var}_r[c] = \frac{B_1 + B_2 + B_3}{n(n-2)(n-3)W^2}$$

with $B_1 = (n-1)S_1[n^2 - 3n + 3 - (n-1)b_2]$, $B_2 = -\frac{1}{4}(n-1)S_2[n^2 + 3n - 6 - (n^2 - n + 2)b_2]$, and $B_3 = W^2[n^2 - 3 - b_2(n-1)^2]$.

If you specify LAGDISTANCE= to be larger than the maximum data distance in your domain, the binary weighting scheme used by the VARIOGRAM procedure leads to all weights $w_{ij} = 1, i \neq j$. In this extreme case the preceding definitions can show that the variances of the I and c statistics become zero under either the normality or the randomization assumption.

A similar effect might occur when you have collocated observations (see the section "Pair Formation" on page 11016). The Moran's I and Geary's c statistics allow for the inclusion of such pairs in the computations. Hence, contrary to the semivariance analysis, PROC VARIOGRAM does not exclude pairs of collocated data from the autocorrelation statistics.

Interpretation

For Moran's I coefficient, $I > \mathrm{E}[I]$ indicates positive autocorrelation. Positive autocorrelation suggests that neighboring values $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ tend to have similar feature values z_i and z_j, respectively. When $I < \mathrm{E}[I]$, this is a sign of negative autocorrelation, or dissimilar values at neighboring locations. A measure of strength of the autocorrelation is the size of the absolute difference $\mid I - \mathrm{E}[I] \mid$.

Geary's c coefficient interpretation is analogous to that of Moran's I. The only difference is that $c > \mathrm{E}[c]$ indicates negative autocorrelation and dissimilarity, whereas $c < \mathrm{E}[c]$ signifies positive autocorrelation and similarity of values.

The VARIOGRAM procedure uses the mathematical definitions in the preceding section to provide the observed and expected values, and the standard deviation of the autocorrelation coefficients in the autocorrelation statistics table. The Z scores for each type of statistics are computed as

$$Z_I = \frac{I - \mathrm{E}[I]}{\sqrt{\mathrm{Var}[I]}}$$

for Moran's I coefficient, and

$$Z_c = \frac{c - \mathrm{E}[c]}{\sqrt{\mathrm{Var}[c]}}$$

for Geary's c coefficient. PROC VARIOGRAM also reports the two-sided p-value for each coefficient under the null hypothesis that the sample values are not autocorrelated. Smaller p-values correspond to stronger autocorrelation for both the I and c statistics. However, the p-value does not tell you whether the autocorrelation is positive or negative. Based on the preceding remarks, you have positive autocorrelation when $Z_I > 0$ or $Z_c < 0$, and you have negative autocorrelation when $Z_I < 0$ or $Z_c > 0$.

The Moran Scatter Plot

The Moran scatter plot (Anselin 1996) is a useful visual tool for exploratory analysis, because it enables you to assess how similar an observed value is to its neighboring observations. Its horizontal axis is based on the values of the observations and is also known as the response axis. The vertical Y axis is based on the weighted average or spatial lag of the corresponding observation on the horizontal X axis. **NOTE:** The term *spatial lag* in the current context is unrelated to the concept of the semivariogram lag presented in the section "Distance Classification" on page 11019.

The Moran scatter plot provides a visual representation of spatial associations in the neighborhood around each observation. You specify a neighborhood size with the LAGDISTANCE= option in the COMPUTE statement. The observations are represented by their standardized values; therefore only nonmissing observations are shown in the plot. For each one of those, the VARIOGRAM procedure computes the weighted average, which is the weighted mean value of its neighbors. Then, the centered weighted average is plotted against the standardized observations. As a result, the scatter plot is centered on the coordinates $(0, 0)$, and distances in the plot are expressed in deviations from the origin $(0, 0)$.

Depending on their position on the plot, the Moran plot data points express the level of spatial association of each observation with its neighboring ones. Conceptually, these characteristics differentiate the Moran plot from the semivariogram. The latter is typically used in geostatistics to depict spatial associations across the whole domain as a continuous function of a distance metric.

You can find the data points on the Moran scatter plot in any of the four quadrants defined by the horizontal line $y = 0$ and the vertical line $x = 0$. Points in the upper right (or high-high) and lower left (or low-low) quadrants indicate positive spatial association of values that are higher and lower than the sample mean, respectively. The lower right (or high-low) and upper left (or low-high) quadrants include observations that exhibit negative spatial association; that is, these observed values carry little similarity to their neighboring ones.

When you use binary, row-averaged weights for the creation of the Moran scatter plot and in autocorrelation statistics, Moran's I coefficient is equivalent to the regression slope of the Moran scatter plot. That is, when you specify

```
PLOTS=MORAN(ROWAVG=ON)
```

in the PROC VARIOGRAM statement and

```
AUTOCORR(WEIGHTS=BINARY(ROWAVERAGING))
```

in the COMPUTE statement, then the regression line slope of the Moran scatter plot is the Moran's I coefficient shown in the section "Autocorrelation Statistics Types" on page 11036. In this sense, the Moran's I coefficient has a global character, whereas the Moran scatter plot provides you with a more detailed exploratory view of the autocorrelation behavior of the individual observations.

This detailed view can reveal outliers with respect to the regression line slope of the Moran scatter plot. Outliers, if present, can function as leverage points that affect the value of Moran's I coefficient. As noted by Anselin (1996), such extremes indicate local instability in spatial association. This instability can be caused either by problems with the autocorrelation weights matrix or by fine-scale characteristics of the spatial structure, which are beneath the current observation structure.

Computational Resources

The fundamental computation of the VARIOGRAM procedure is binning: for each pair of observations in the input data set, a distance class and an angle class are determined and recorded. Let N_d denote the number of distance classes, N_a denote the number of angle classes, and N_v denote the number of VAR variables. The memory requirements for these operations are proportional to $N_d \times N_a \times N_v$. This is typically small.

The CPU time required for the computations is proportional to the number of pairs of observations, or to $N^2 \times N_v$, where N is the number of observations in the input data set.

Output Data Sets

The VARIOGRAM procedure produces four data sets: the OUTACWEIGHTS=*SAS-data-set*, the OUTDIST=*SAS-data-set*, the OUTPAIR=*SAS-data-set*, and the OUTVAR=*SAS-data-set*. These data sets are described in the following sections.

OUTACWEIGHTS=*SAS-data-set*

The OUTACWEIGHTS= data set contains one observation for each pair of points P_1, P_2 in the original data set, where P_1 is different from P_2, with information about the data distance and autocorrelation weight of each point pair.

The OUTACWEIGHTS= data set can be very large, even for a moderately sized DATA= data set. For example, if the DATA= data set has NOBS=500, then the OUTACWEIGHTS= data set has NOBS(NOBS $-$ 1)/2=124,750 observations.

When you perform autocorrelation computations, the OUTACWEIGHTS= data set is a practical way to save the autocorrelation weights for further use.

The OUTACWEIGHTS= data set contains the following variables:

- ACWGHT12, the autocorrelation weight for the pair P_1, P_2
- ACWGHT21, the autocorrelation weight for the pair P_2, P_1
- DISTANCE, the distance between the data in the pair
- ID1, the ID variable value or observation number for the first point in the pair
- ID2, the ID variable value or observation number for the second point in the pair
- V1, the variable value for the first point in the pair
- V2, the variable value for the second point in the pair
- VARNAME, the variable name for the current VAR variable
- X1, the *x* coordinate of the first point in the pair
- X2, the *x* coordinate of the second point in the pair

- Y1, the y coordinate of the first point in the pair
- Y2, the y coordinate of the second point in the pair

When the autocorrelation weights are symmetric, the pair P_1, P_2 has the same weight as the pair P_2, P_1. For this reason, in the case of symmetric weights the OUTACWEIGHTS= data set contains only the autocorrelation weights ACWGHT12.

If no ID statement is specified, then the corresponding observation number is assigned to each one of the variables ID1 and ID2, instead.

OUTDIST=*SAS-data-set*

The OUTDIST= data set contains counts for a modified histogram that shows the distribution of pairwise distances. This data set provides you with information related to the choice of values for the LAGDISTANCE= option in the COMPUTE statement.

To request an OUTDIST= data set, specify the OUTDIST= data set in the PROC VARIOGRAM statement and the NOVARIOGRAM option in the COMPUTE statement. The NOVARIOGRAM option prevents any semivariogram or covariance computation from being performed.

The following variables are written to the OUTDIST= data set:

- COUNT, the number of pairs that fall into this lag class
- LAG, the lag class value
- LB, the lower bound of the lag class interval
- UB, the upper bound of the lag class interval
- PER, the percent of all pairs that fall in this lag class
- VARNAME, the name of the current VAR variable

OUTMORAN=*SAS-data-set*

The OUTMORAN= data set contains the standardized value (or response) of each observation and the weighted average of its N neighbors, based on a neighborhood within a LAGDISTANCE= distance from the observation. To request this data set, specify the OUTMORAN= data set in the PROC VARIOGRAM statement, in addition to the AUTOCORRELATION and LAGDISTANCE= options in the COMPUTE statement.

The following variables are written to the OUTMORAN= data set:

- DISTANCE, the value of the neighborhood radius, which is specified with the LAGDISTANCE= option
- ID, the ID variable value or observation number for the current observation
- N, the number of neighbors within the specified DISTANCE from the current observation
- RESPONSE, the standardized value of the current observation

- STDWAVG, the standardized weighted average of the neighbors for the current observation
- V, the variable value of the current observation
- VARNAME, the variable name for the current VAR variable
- X, the x coordinate of the current observation
- Y, the y coordinate of the current observation
- WAVG, the weighted average of the neighbors for the current observation

For zero neighbors in the neighborhood of a nonmissing observation, the corresponding value of the variable N=0 and the variables STDWAVG and WAVG are assigned missing values. Observations with missing values are included in the OUTMORAN= data set if they have neighbors and only if nonmissing observations with neighbors also exist in the same data set.

OUTPAIR=*SAS-data-set*

When you specify the NOVARIOGRAM option in the COMPUTE statement, the OUTPAIR= data set contains one observation for each distinct pair of points P_1, P_2 in the original data set. Otherwise, the OUTPAIR= data set might have fewer observations, depending on the values you specify in the LAGDISTANCE= and MAXLAGS= options and whether you specify the OUTPDISTANCE= option in the COMPUTE statement.

If the NOVARIOGRAM option is not specified in the COMPUTE statement, then the OUTPAIR= data set contains one observation for each distinct pair of points that are up to a distance within MAXLAGS= away from each other. If you also specify the OUTPDISTANCE=D_{max} option in the COMPUTE statement, then all pairs P_1, P_2 in the original data set that satisfy the relation $\mid P_1 P_2 \mid \leq D_{max}$ are written to the OUTPAIR= data set.

Given the aforementioned specifications, note that the OUTPAIR= data set can be very large even for a moderately sized DATA= data set. For example, if the DATA= data set has NOBS=500, then the OUTPAIR= data could have up to NOBS(NOBS $-$ 1)/2=124,750 observations if no OUTPDISTANCE= restriction is given in the COMPUTE statement.

The OUTPAIR= data set contains information about the distance and orientation of each point pair, and you can use it for specialized continuity measure calculations.

The OUTPAIR= data set contains the following variables:

- AC, the angle class value
- COS, the cosine of the angle between pairs
- DC, the distance (lag) class
- DISTANCE, the distance between the data in pairs
- ID1, the ID variable value or observation number for the first point in the pair
- ID2, the ID variable value or observation number for the second point in the pair
- V1, the variable value for the first point in the pair

- V2, the variable value for the second point in the pair
- VARNAME, the variable name for the current VAR variable
- X1, the x coordinate of the first point in the pair
- X2, the x coordinate of the second point in the pair
- Y1, the y coordinate of the first point in the pair
- Y2, the y coordinate of the second point in the pair

If no ID statement is specified, then the corresponding observation number is assigned to each one of the variables ID1 and ID2, instead.

OUTVAR=*SAS-data-set*

The OUTVAR= data set contains the standard and robust versions of the sample semivariance, the covariance, and other information in each lag class.

The OUTVAR= data set contains the following variables:

- ANGLE, the angle class value (clockwise from N to S)
- ATOL, the angle tolerance for the lag or angle class
- AVERAGE, the average variable value for the lag or angle class
- BANDW, the bandwidth for the lag or angle class
- COUNT, the number of pairs in the lag or angle class
- COVAR, the covariance value for the lag or angle class
- DISTANCE, the average lag distance for the lag or angle class
- LAG, the lag class value (in LAGDISTANCE= units)
- RVARIO, the sample robust semivariance value for the lag or angle class
- STDERR, the approximate standard error of the sample semivariance estimate
- VARIOG, the sample semivariance value for the lag or angle class
- VARNAME, the name of the current VAR variable

The robust semivariance estimate, RVARIO, is not included in the data set if you omit the option ROBUST in the COMPUTE statement.

The bandwidth variable, BANDW, is not included in the data set if no bandwidth specification is given in the COMPUTE statement or in a DIRECTIONS statement.

The OUTVAR= data set contains a line where the LAG variable is –1. The AVERAGE variable in this line displays the sample mean value $\bar{Z}$ of the SRF $Z(\boldsymbol{s})$, and the COVAR variable shows the sample variance $\mathrm{Var}[Z(\boldsymbol{s})]$.

Displayed Output

In addition to the output data sets, the VARIOGRAM procedure produces a variety of output objects. Most of these are produced depending on whether you specify either NOVARIOGRAM or LAGDISTANCE= and MAXLAGS= in the COMPUTE statement. The VARIOGRAM procedure output objects are the following:

- a default "Number of Observations" table that displays the number of observations read from the input data set and the number of observations used in the analysis
- a default map that shows the spatial distribution of the observations of the current variable in the VAR statement. The observations are displayed by default with circled markers whose color indicates the VAR value at the corresponding location.
- a table with basic information about the lags and the extreme distance between data pairs, when NOVARIOGRAM is specified
- a table that describes the distribution of data pairs in distance intervals, when NOVARIOGRAM is specified
- a histogram plot of the pairwise distance distribution, when NOVARIOGRAM is specified). The plot also displays a reference line at a user-specified pairs frequency threshold when you specify the THRESHOLD= parameter in the PLOTS=PAIRS option. The option PLOTS=PAIRS(NOINSET) forces the informational inset that appears in the plot to hide.
- empirical semivariogram details, when NOVARIOGRAM is not specified and LAGDISTANCE= and MAXLAGS= are specified. This table also includes the semivariance estimate variance and confidence limits when CL is specified, and estimates of the robust semivariance when ROBUST is specified.
- plots of the appropriate empirical semivariograms, when NOVARIOGRAM is not specified and LAGDISTANCE= and MAXLAGS= are specified. If you perform the analysis in more than one direction simultaneously, the output is a panel that contains the empirical semivariogram plots for the specified angles. If the semivariograms are nonpaneled, then each plot includes in the lower part a needle plot of the contributing pairs distribution.
- a table that provides autocorrelation statistics, when the options AUTOCORRELATION and LAGDISTANCE= are specified
- the Moran scatter plot of the standardized observation values against the weighted averages of their neighbors, when the options PLOTS=MORAN, AUTOCORRELATION, and LAGDISTANCE= are specified

When you specify the MODEL statement and request a fit of a theoretical model to the empirical semivariogram, the VARIOGRAM procedure also produces the following default output:

- a table with some general fitting information, in addition to the output item store if you have specified one with the STORE statement
- a table with more specific information about the selected model's parameters and their initial values

- a table with general information about the optimization that provides the fitting parameters of the selected model
- a table with the optimization process output and a table with the convergence status of the optimization process, if you have specified a single model to fit
- a "Parameter Estimates" table with information about the fitted parameters estimates
- a "Fit Summary" table that reports the fit quality of all models you requested to fit
- plots of fitted theoretical semivariogram models. If you perform model fitting in more than one direction angle or for more than one variable in your DATA= data set, then the output is a panel that contains all fitted models for the respective directions or variables.

Additional output can be produced in model fitting if you specify a higher level of output detail with the DETAILS option in the MODEL statement. This output can be information tables for each separate model when you specify multiple models to fit, tables with more details about the optimization process, and the covariance and correlation matrices of the model parameter estimates. The complete listing of the PROC VARIOGRAM output follows in the section "ODS Table Names" on page 11045 and the section "ODS Graph Names" on page 11047.

ODS Table Names

Each table created by PROC VARIOGRAM has a name associated with it, and you must use this name to refer to the table when using ODS Graphics. These names are listed in Table 131.6.

Table 131.6 ODS Tables Produced by PROC VARIOGRAM

ODS Table Name	Description	Required Statement	Option
AutoCorrStats	Autocorrelation statistics information	COMPUTE	AUTOCORRELATION
ConvergenceStatus	Status of optimization at conclusion	MODEL	Default output
CorrB	Approximate correlation matrix of model parameter estimates	MODEL	CORRB
CovB	Approximate covariance matrix of model parameter estimates	MODEL	COVB
DistanceIntervals	Pairwise distances matrix	COMPUTE	NOVARIOGRAM
FitGenInfo	General fitting information	MODEL	Default output
FitSummary	Fitting process summary	MODEL	Default output
InputOptions	Optimization input options	MODEL	DETAILS=ALL
IterHist	Iteration history	MODEL	DETAILS=ITR
IterStop	Optimization-related results	MODEL	Default output
Lagrange	Information about Lagrange multipliers	MODEL	DETAILS=ALL

Table 131.6 *continued*

ODS Table Name	Description	Required Statement	Option
ModelInfo	Model information	MODEL	Default output
NObs	Number of observations read and used	PROC	Default output
OptInfo	Optimization information	MODEL	Default output
PairsInformation	General information about the pairs distribution in classes and data maximum distances in selected directions	COMPUTE	NOVARIOGRAM
ParameterEstimates	Model fitting solution and statistics	MODEL	Default output
ParameterEstimatesResults	Parameter estimates and gradient information	MODEL	DETAILS=ALL
ParameterEstimatesStart	More detailed model information	MODEL	DETAILS=ITR
ParmSearch	Parameter search values	MODEL	Default output
ProblemDescription	Information at the optimization start	MODEL	DETAILS=ITR
ProjGrad	Projected gradient information	MODEL	DETAILS=ALL
SemivariogramTable	Empirical semivariance classes, parameters, and estimates	COMPUTE	LAGD=, MAXLAGS=

ODS Graphics

Statistical procedures use ODS Graphics to create graphs as part of their output. ODS Graphics is described in detail in Chapter 24, "Statistical Graphics Using ODS."

Before you create graphs, ODS Graphics must be enabled (for example, by specifying the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section "Enabling and Disabling ODS Graphics" on page 687 in Chapter 24, "Statistical Graphics Using ODS."

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section "A Primer on ODS Statistical Graphics" on page 686 in Chapter 24, "Statistical Graphics Using ODS."

For additional control of the graphics that are displayed, see the PLOTS option in the section "PROC VARIOGRAM Statement" on page 10976.

ODS Graph Names

PROC VARIOGRAM assigns a name to each graph it creates by using ODS Graphics. You can use this name to refer to the graph when using ODS Graphics. You must also specify the PLOTS= option indicated in Table 131.7.

Table 131.7 Graphs Produced by PROC VARIOGRAM

ODS Graph Name	Plot Description	Statement	Option
FitPanel	Panel of one or more classes of fitted semivariograms in different angles	PROC	PLOTS=FIT
FitPlot	Plot of one or more classes of fitted semivariograms	PROC	PLOTS=FIT
MoranPlot	Scatter plot of standardized observed values against weighted averages	PROC	PLOTS=MORAN
ObservationsPlot	Scatter plot of observed data and colored markers that indicates observed values	PROC	PLOTS=OBSERV
PairDistPlot	Histogram of the pairwise distance distribution	PROC	PLOTS=PAIRS
Semivariogram	Plots of empirical classical and robust (optional) semivariograms	PROC	PLOTS=SEMIVAR
SemivariogramPanel	Panel of empirical classical and robust (optional) semivariogram plots	PROC	PLOTS=SEMIVAR

Examples: VARIOGRAM Procedure

The examples in this chapter are available in the GitHub repository located at `https://github.com/sascommunities/doc-supplement-statug`.

Example 131.1: Aspects of Semivariogram Model Fitting

(View the complete code for this example (vroex1.sas) in the example repository.)

This example helps you explore aspects of automated semivariogram fitting with PROC VARIOGRAM. The test case is a spatial study of arsenic (As) concentration in drinking water.

Arsenic is a toxic pollutant that can occur in drinking water because of human activity or, typically, due to natural release from the sediments in water aquifers. The World Health Organization has a standard that allows As concentration up to a maximum of 10 μg/lt (micrograms per liter) in drinking water.

In general, natural release of arsenic into groundwater is very slow. Arsenic concentration in water might exhibit no significant temporal fluctuations over a period of a few months. For this reason, it is acceptable to perform a spatial study of arsenic with input from time-aggregated pollutant concentrations. This example makes use of this assumption for its data set logAsData. The data set consists of 138 simulated observations from wells across a square area of 500 km $\times$ 500 km. The variable logAs in the logAsData data set is the

natural logarithm of arsenic concentration. Often, the natural logarithm of arsenic concentration (log*As*) is used as the random variable to facilitate the analysis because its distribution tends to resemble the normal distribution.

The goal is to explore spatial continuity in the logAs observations. The following statements read the logAs values from the logAsData data set:

```
title 'Semivariogram Model Fitting of Log-Arsenic Concentration';

data logAsData;
   input East North logAs @@;
   label logAs='log(As) Concentration';
   datalines;
193.0 296.6 -0.68153   232.6 479.1  0.96279   268.7 312.5 -1.02908
 43.6   4.9  0.65010   152.6  54.9  1.87076   449.1 395.8  0.95932
310.9 493.6 -1.66208   287.8 164.9 -0.01779   330.0   8.0  2.06837
225.7 241.7  0.15899   452.3  83.4 -1.21217   156.5 462.5 -0.89031
 11.5  84.4 -0.24496   144.4 335.7  0.11950   149.0 431.8 -0.57251
234.3 123.2 -1.33642    37.8 197.8 -0.27624   183.1 173.9 -2.14558
149.3 426.7 -1.06506   434.4  67.5 -1.04657   439.6 237.0 -0.09074
 36.4 175.2 -1.21211   370.6 244.0  3.28091   452.0  96.5 -0.77081
247.0  86.8  0.04720   413.6 373.2  1.78235   253.5 291.7  0.56132
129.7 111.9  1.34000   352.7  42.1  0.23621   279.3  82.7  2.12350
382.6 290.7  0.86756   188.2 222.8 -1.23308   382.8 154.5 -0.94094
304.4 309.2 -1.95158   337.5 387.2 -1.31294   490.7 189.8  0.40206
159.0 100.1 -0.22272   245.5 329.2 -0.26082   372.1 379.5 -1.89078
417.8  84.1 -1.25176   173.9 407.6 -0.24240   121.5 107.7  1.54509
453.5 313.6  0.65895   143.5 346.7 -0.87196   157.4 125.5 -1.96165
371.8 353.2 -0.59464   358.9 338.2 -1.07133     8.6 437.8  1.44203
395.9 394.2 -0.24144   149.5  58.9  1.17459   453.5 420.6 -0.63951
182.3  85.0  1.00005    21.0 290.1  0.31016    11.1 352.2 -0.88418
131.2 238.4 -0.57184   104.9   6.3  1.12054   247.3 256.0  0.14019
428.4 383.7  0.92448   327.8 481.1 -2.72543   199.2  92.8 -0.05717
453.9 230.1  0.16571   205.0 250.6  0.07581   459.5 271.6  0.93700
229.5 262.8  1.83590   370.4 228.6  2.96611   330.2 281.9  1.79723
354.8 388.3 -3.18262   406.2 222.7  2.41594   254.4 393.1  2.03221
 96.7  85.2 -0.47156   407.2 256.8  0.66747   498.5 273.8  1.03041
417.2 471.4 -1.42766   368.8 424.3 -0.70506   303.0  59.1  1.43070
403.1 264.1  1.64554    21.2 360.8  0.67094   148.2  78.1  2.15323
305.5 310.7 -1.47985   228.5 180.3 -0.68386   161.1 143.3  1.07901
 70.5 155.1  0.54652   363.1 282.6 -0.43051    86.0 472.5 -1.18855
175.9 105.3 -2.08112    96.8 426.3  1.56592   475.1 453.1 -1.53776
125.7 485.4  1.40054   277.9 201.6 -0.54565   406.2 125.0 -1.38657
 60.0 275.5 -0.59966   431.3 494.6 -0.36860   399.9 399.0 -0.77265
 28.8 311.1  0.91693   166.1 348.2 -0.49056   266.6  83.5  0.67277
 54.7 356.3  0.49596   433.5 460.3 -1.61309   201.7 167.6 -1.40678
158.1 203.6 -1.32499    67.6 230.4  1.14672    81.9 250.0  0.63378
372.0  50.7  0.72445    26.4 264.6  1.00862   300.1  91.7 -0.74089
303.0 447.4  1.74589   108.4 386.2  1.12847    55.6 191.7  0.95175
 36.3 273.2  1.78880    94.5 298.3 -2.43320   366.1 187.3 -0.80526
130.7 389.2 -0.31513    37.2 324.2  0.24489   295.5 211.8  0.41899
 58.6 206.2  0.18495   346.3 142.8 -0.92038   484.2 215.9  0.08012
451.4 415.7  0.02773    58.9  86.5  0.17652   212.6 363.9  0.17215
378.7 407.6  0.51516   265.9 305.0 -0.30718   123.2 314.8 -0.90591
```

```
 26.9 471.7  1.70285    16.5   7.1  0.51736   255.1 472.6  2.02381
111.5 148.4 -0.09658   440.4 375.0  1.23285   406.4  19.5  1.01181
321.2  65.8 -0.02095   466.4 357.1 -0.49272     2.0 484.6  0.50994
200.9 205.1  0.43543    30.3 337.0  1.60882   297.0  12.7  1.79824
158.2 450.7  0.05295   122.8 105.3  1.53936   417.8 329.7 -2.08124
;
```

First you want to inspect the log*As* data for surface trends and the pairwise distribution. You run the VARIOGRAM procedure with the NOVARIOGRAM option in the COMPUTE statement. You also request the PLOTS=PAIRS(MID) option, which prompts the pair distance plot to display the actual distance between pairs, rather than the lag number itself, in the midpoint of the lags. You use the following statements:

```
ods graphics on;

proc variogram data=logAsData plots=pairs(mid);
   compute novariogram nhc=50;
   coord xc=East yc=North;
   var logAs;
run;
```

The observations scatter plot in Output 131.1.1 shows a rather uniform distribution of the locations in the study domain. Reasonably, neighboring values of logAs seem to exhibit some correlation. There seems to be no definite sign of an overall surface trend in the logAs values. You can consider that the observations are trend-free, and proceed with estimation of the empirical semivariance.

Output 131.1.1 logAs Observation Data Scatter Plot

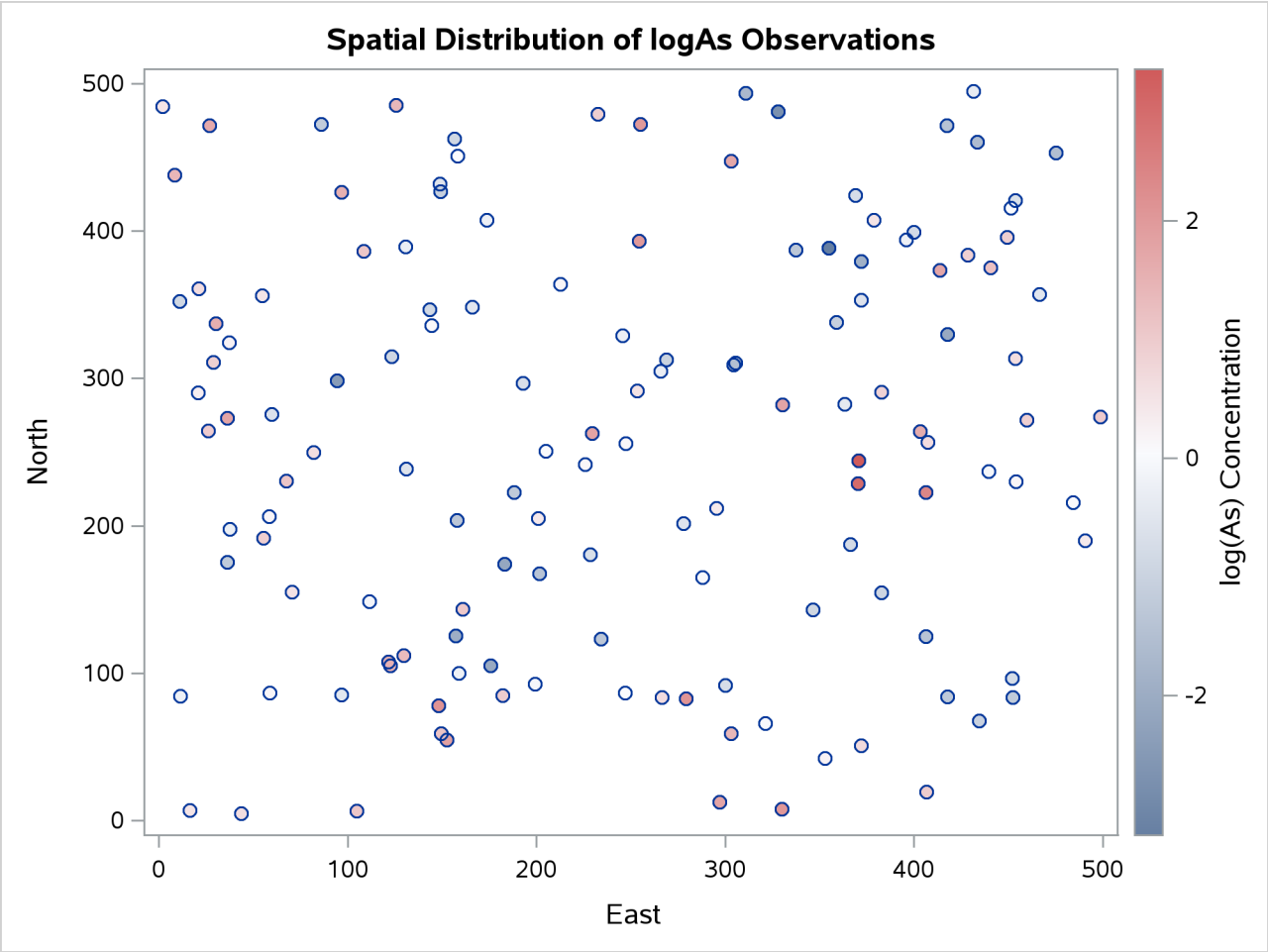

The observed logAs values go as high as 3.28091, which corresponds to a concentration of 26.6 μg/lt. In fact, only three observations exceed the health standard of 10 μg/lt (or about 2.3 in the log scale), and they are situated in relatively neighboring locations to the east of the domain center.

Based on the discussion in section "Preliminary Spatial Data Analysis" on page 10962, the pair distance plot in Output 131.1.2 suggests that you could consider pairs that are anywhere around up to half the maximum pairwise distance of about 700 km.

Output 131.1.2 Distribution of Pairwise Distances for logAs Data

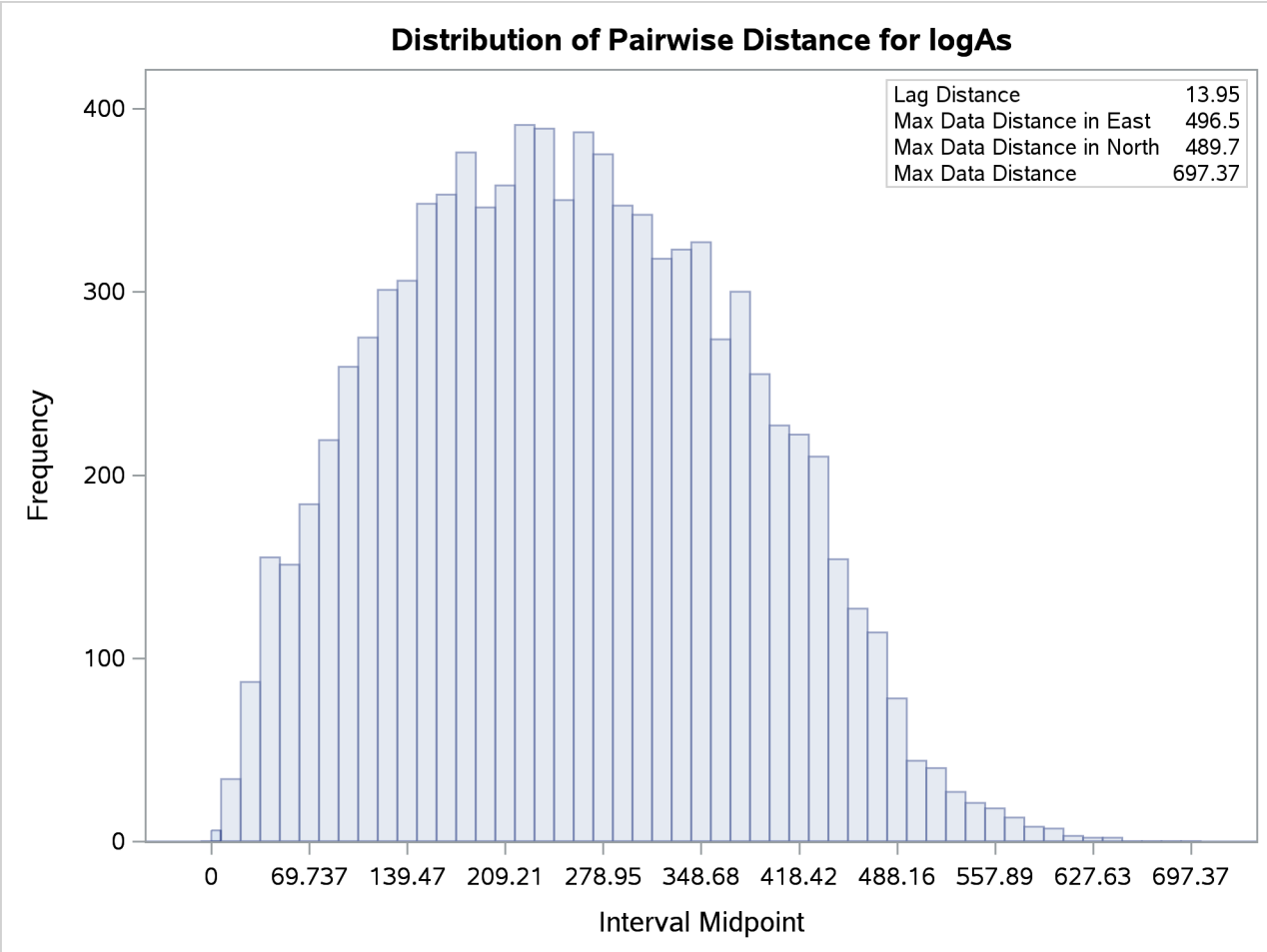

After some experimentation with values for the LAGDISTANCE= and MAXLAGS= options, you actually find that a lag distance of 5 km over 40 lags can provide a clear representation of the logAs semivariance. With respect to Output 131.1.2, this finding indicates that in the current example it is sufficient to consider pairs separated by a distance of up to 200 km. You run the following statements to obtain the empirical semivariogram:

```
proc variogram data=logAsData plots(only)=semivar;
   compute lagd=5 maxlag=40;
   coord xc=East yc=North;
   var logAs;
run;
```

The first few lag classes of the logAs empirical semivariance table are shown in Output 131.1.3.

Output 131.1.3 Partial Output of the Empirical Semivariogram Table for logAs Data

Semivariogram Model Fitting of Log-Arsenic Concentration

The VARIOGRAM Procedure
Dependent Variable: logAs

Empirical Semivariogram

Lag Class	Pair Count	Average Distance	Semivariance
0	1	1.9	0.111
1	5	4.9	0.145
2	6	9.7	0.286
3	11	14.6	0.545
4	27	20.0	0.900

Output 131.1.3 and Output 131.1.4 indicate that the logarithm of arsenic spatial correlation starts with a small nugget effect around 0.11 and rises to a sill value that is most likely between 1.4 and 1.8. The rise could be of exponential type, although the smooth increase of semivariance close to the origin could also suggest Gaussian behavior. You suspect that a Matérn form might also work, since its smoothness parameter ν can regulate the form to exhibit an intermediate behavior between the exponential and Gaussian forms.

Output 131.1.4 Empirical Semivariogram for logAs Data

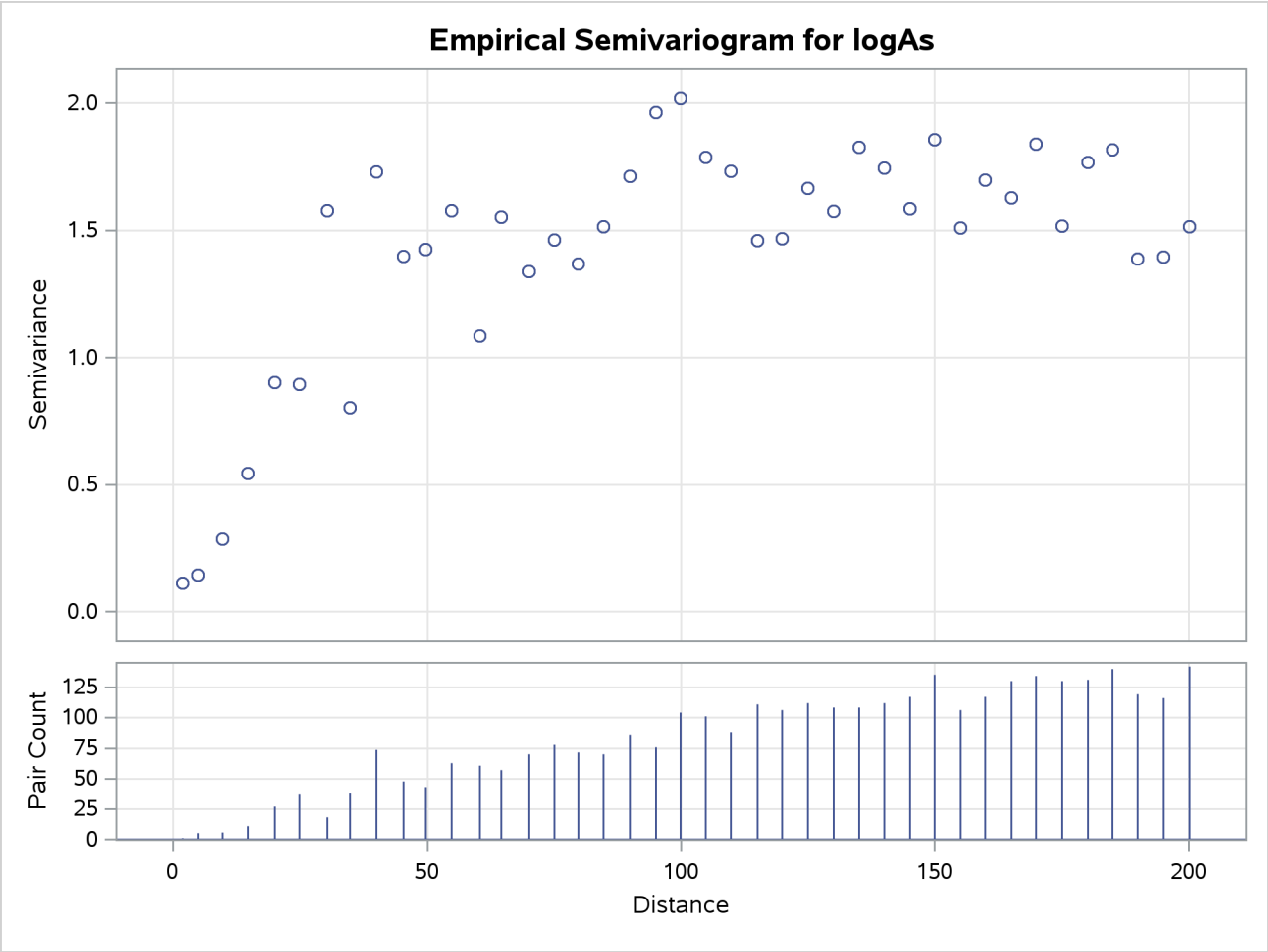

You can investigate all of the preceding clues with the model fitting features of PROC VARIOGRAM. The simplest way to fit a model is to specify its form in the MODEL statement. In this case, you have the added complexity of having more than one possible candidate. For this reason, you use the FORM=AUTO option that picks the best fit out of a list of candidates. Within this option you specify the MLIST= suboption to use the exponential, Gaussian, and Matérn forms. You also specify the NEST= suboption to request fitting of a model with up to two nested structures. Eventually, you specify the PLOTS=FIT option to produce a plot of the fitted models. The STORE statement saves the fitting output into an item store you name SemivAsStore for future use. You apply these specifications with the following statements:

```
proc variogram data=logAsData plots(only)=fit;
   store out=SemivAsStore / label='LogAs Concentration Models';
   compute lagd=5 maxlag=40;
   coord xc=East yc=North;
   model form=auto(mlist=(exp,gau,mat) nest=1 to 2);
   var logAs;
run;

ods graphics off;
```

The table of general information about fitting is shown in Output 131.1.5. The table lets you know that 12 model combinations are to be tested for weighted least squares fitting, based on the three forms that you specified.

Output 131.1.5 Semivariogram Model Fitting General Information

Semivariogram Model Fitting of Log-Arsenic Concentration

The VARIOGRAM Procedure

Dependent Variable: logAs
Angle: Omnidirectional

Semivariogram Model Fitting	
Model	Selection from 12 form combinations
Output Item Store	WORK.SEMIVASSTORE
Item Store Label	LogAs Concentration Models

The combinations include repetitions. For example, you specified the GAU form; hence the GAU-GAU form is tested, too. The model combinations also include permutations. For example, you specified the GAU and the EXP forms; hence the GAU-EXP and EXP-GAU models are fitted separately. According to the section "Nested Models" on page 11013, it might seem that the same model is fitted twice. However, in each of these two cases, each structure starts the fitting process with different parameter initial values. This can lead GAU-EXP to a different fit than EXP-GAU leads to, as seen in the fitting summary table in Output 131.1.6. The table shows all the model combinations that were tested and fitted. By default, the ordering is based on the weighted sum of squares error criterion, and you can see that the lowest values in the Weighted SSE column are in top slots of the list.

Output 131.1.6 Semivariogram Model Fitting Summary

Fit Summary

Class	Model	Weighted SSE	AIC
1	Gau-Gau	25.42435	-9.59246
	Gau-Mat	25.42482	-7.59169
	Mat-Mat	25.42970	-5.58383
2	Exp-Gau	26.36540	-8.10230
3	Exp-Mat	26.36846	-6.09754
4	Mat	26.37519	-10.08708
5	Gau	26.78629	-11.45296
6	Exp	28.01200	-9.61851
	Exp-Exp	28.01200	-5.61850
	Mat-Exp	28.01200	-3.61850
	Gau-Exp	28.01200	-5.61850

Note the leftmost Class column in Output 131.1.6. As explained in detail in section "Classes of Equivalence" on page 11034, when you fit more than one model, all fitted models that compute the same semivariance are placed in the same class of equivalence. For example, in this fitting example the top ranked GAU-GAU and GAU-MAT nested models produce indistinguishable semivariograms; for that reason they are both placed in the same class 1 of equivalence. The same occurs with the EXP, GAU-EXP, EXP-EXP, and MAT-EXP models

in the bottom of the table. By default, PROC VARIOGRAM uses the AIC as a secondary classification criterion; hence models in each equivalence class are already ordered based on their AIC values.

Another remark in Output 131.1.6 is that despite submitting 12 model combinations for fitting, the table shows only 10. You can easily see that the combinations MAT-GAU and MAT-MAT are not among the listed models in the fit summary. This results from the behavior of the VARIOGRAM procedure in the following situation: A parameter optimization takes place during the fitting process. In the present case the optimizer keeps increasing the Matérn smoothness parameter ν in the MAT-GAU model. At the limit of an infinite ν parameter, the Matérn form becomes the Gaussian form. For that reason, when the parameter ν is driven towards very high values, PROC VARIOGRAM automatically replaces the Matérn form with the Gaussian. This switch converts the MAT-GAU model into a GAU-GAU model. However, a GAU-GAU model already exists among the specified forms; consequently, the duplicate GAU-GAU model is skipped, and the fitted model list is reduced by one model. A similar explanation justifies the omission of the MAT-MAT model from the fit summary table.

In our example, the nested Gaussian-Gaussian model is the fitting selection of the procedure based on the default ranking criteria. Output 131.1.7 displays additional information about the selected model. In particular, you see the table with general information about the Gaussian-Gaussian model, the initial values used for its parameters, and information about the optimization process for the fitting.

Output 131.1.7 Fitting and Optimization Information for Gaussian-Gaussian Model

Semivariogram Model Fitting	
Name	Gaussian-Gaussian
Label	Gau-Gau

Model Information	
Parameter	**Initial Value**
Nugget	0.0903
GauScale1	0.6709
GauRange1	100.0
GauScale2	0.6709
GauRange2	50.0230

Optimization Information	
Optimization Technique	Dual Quasi-Newton
Parameters in Optimization	5
Lower Boundaries	5
Upper Boundaries	0
Starting Values From	PROC

The estimated parameter values of the selected Gaussian-Gaussian model are shown in Output 131.1.8.

Output 131.1.8 Parameter Estimates of the Fitting Selected Model

Parameter Estimates					
Parameter	Estimate	Approx Std Error	DF	t Value	Approx Pr > |t|
Nugget	0.08308	0.05097	36	1.63	0.1118
GauScale1	0.3277	0.2077	36	1.58	0.1234
GauRange1	62.3128	19.8489	36	3.14	0.0034
GauScale2	1.2615	0.2070	36	6.10	<.0001
GauRange2	21.4596	3.2722	36	6.56	<.0001

By default, when you specify more than one model to fit, PROC VARIOGRAM produces a fit plot that compares the first five classes of the successfully fit candidate models. The model that is selected according to the specified fitting criteria is shown with a thicker line in the plot.

You can modify the number of displayed equivalence classes with the NCLASSES= suboption of the PLOTS=FIT option. When you have such comparison plots, PROC VARIOGRAM displays the representative model from each class of equivalence.

The default fit plot for the current model comparison is shown in Output 131.1.9. The legend informs you there is one more model in the first class of equivalence, as the fitting summary table indicated earlier in Output 131.1.6.

Output 131.1.9 Fitted Theoretical and Empirical logAs Concentration Semivariograms for the Specified Models

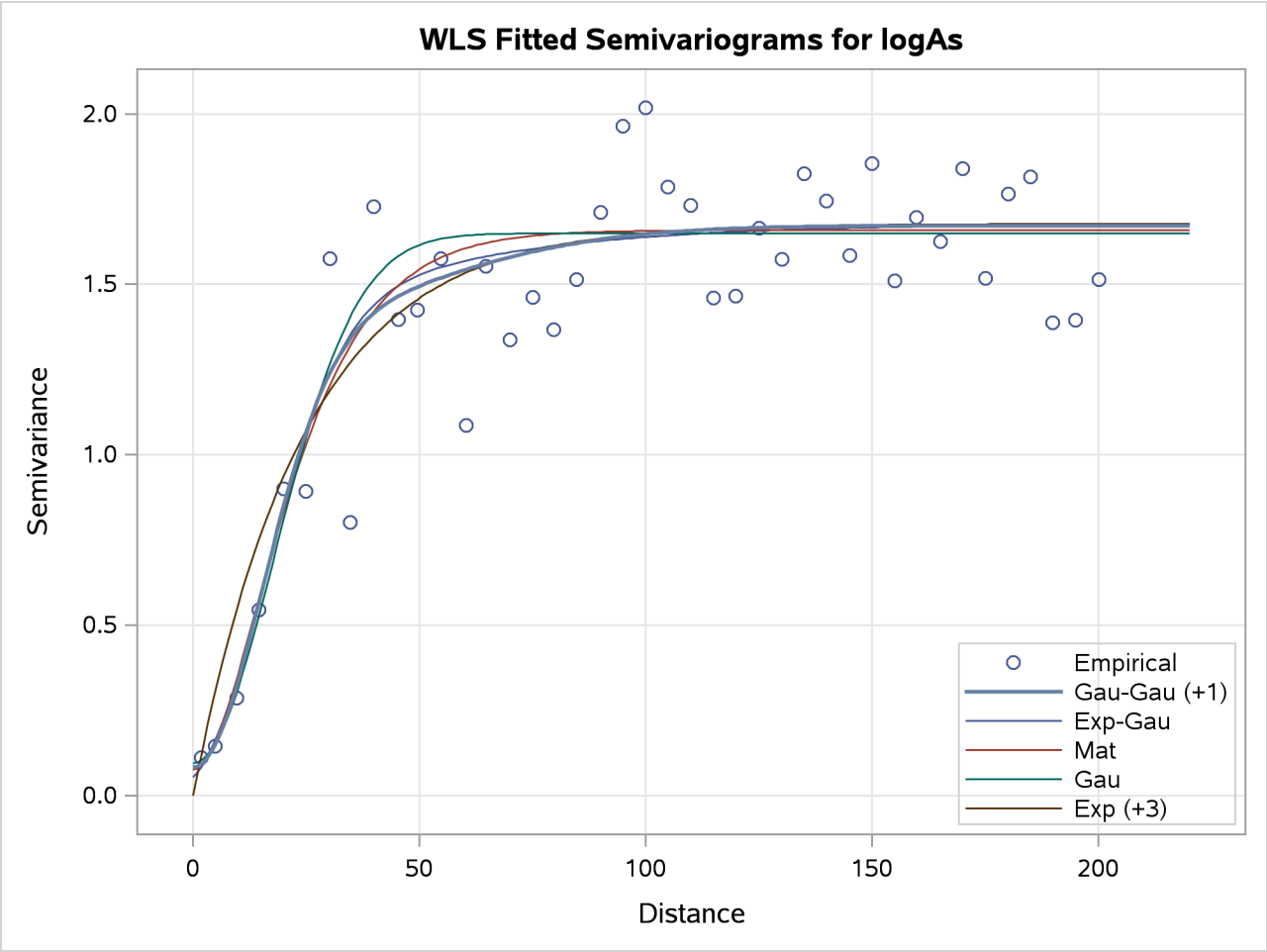

In the present example, all fitted models in the first five classes have very similar semivariograms. The selected Gaussian-Gaussian model seems to have a relatively larger range than the rest of the displayed models, but you can expect any of these models to exhibit a near-identical behavior in terms of spatial correlation. As a result, all models in the displayed classes are likely to lead to very similar output, if you proceed to use any of them for spatial prediction.

In that sense, semivariogram fitting is a partially subjective process, for which there might not exist only one single correct answer to solve your problem. In the context of the example, on one hand you might conclude that the selected Gaussian-Gaussian model is exactly sufficient to describe spatial correlation in the arsenic study. On the other hand, the similar performance of all models might prompt you to choose instead a more simple non-nested model for prediction like the Matérn or the Gaussian model.

Regardless of whether you might opt to sacrifice the statistically best fit (depending on your selected criteria) to simplicity, eventually you are the one to decide which approach serves your study optimally. The model fitting features of PROC VARIOGRAM offer you significant assistance so that you can assess your options efficiently.

Example 131.2: An Anisotropic Case Study with Surface Trend in the Data

(View the complete code for this example (vroex2.sas) in the example repository.)

This example shows how to examine data for nonrandom surface trends and anisotropy. You use simulated data where the variable is atmospheric ozone (O_3) concentrations measured in Dobson units (DU). The coordinates are offsets from a point in the southwest corner of the measurement area, with the east and north distances in units of kilometers (km). You work with the ozoneSet data set that contains 300 measurements in a square area of 100 km × 100 km.

The following statements read the data set:

```
title 'Semivariogram Analysis in Anisotropic Case With Trend Removal';

data ozoneSet;
   input East North Ozone @@;
   datalines;
34.9 68.2 286  39.2 12.5 270  44.4 37.7 275  90.5 27.0 282
91.1 40.8 285  98.6 61.6 294  61.8 26.7 281  64.0 11.5 274
22.4 26.5 274  89.3 18.3 279  32.3 28.3 274  31.1 53.1 279
43.0 17.5 272  79.3 42.3 283  99.9 57.9 291   1.8 24.1 273
81.7 73.5 294  22.9 32.0 273  64.9 67.5 292  76.5 56.3 285
78.7 11.7 276  61.8 99.3 307  49.1 86.6 299  40.0 35.8 273
69.3  3.8 278  23.4  9.3 270  66.3 94.3 304  71.3  6.5 275
 9.7 54.4 280  85.2 81.7 300  30.3 60.9 284  94.6 94.3 309
10.6 10.3 271  73.0 43.0 280   4.9 50.7 280  19.0 79.4 289
 2.4 73.1 287  77.7 25.2 278   8.4 27.1 276  93.5 19.7 279
 0.2 34.5 275  50.4 91.3 302  55.7 26.2 279  50.3  2.3 274
16.3 84.4 293  19.0  6.9 272  57.1 92.3 303  61.0  0.4 275
10.7 18.7 271  15.2 43.5 277  67.0 87.4 301  79.0 54.0 285
36.0 53.3 279  58.3 52.1 282  56.6 79.7 294  40.4 32.4 275
48.9 64.1 286  54.0 54.9 281  27.5 48.5 279  36.4 30.3 275
10.5 31.0 273  87.0 39.4 283  47.9 37.5 274  64.7 63.4 288
 0.5 90.8 294  22.8 22.4 275  31.1 78.8 291  93.6 49.8 290
 2.5 39.3 273  83.6 25.6 282  49.8 24.1 278  73.1 91.8 305
30.5 90.6 297  26.0 61.2 284  58.4 66.2 289  30.5  4.3 273
38.3 85.6 298  89.2 96.6 309  53.4  6.3 275  27.3 12.8 271
43.4 56.5 281  99.5 86.9 305  85.8 22.8 281  83.0 10.9 278
24.8 16.7 271  51.1 18.8 275  59.0 54.3 283  35.5 91.4 298
18.1 56.0 279  78.0 36.4 277  56.8  6.9 275  21.1 44.5 277
73.9 75.9 296  54.2  0.1 274  33.2 75.1 290  38.2  3.3 274
15.2 14.7 272  15.9 84.2 292  60.2 95.2 304   9.8 27.2 276
91.2 56.4 289  94.7 86.9 303  56.7 49.6 281  24.2  9.5 270
43.0 17.0 272  85.9 10.7 278  53.9 41.1 276  30.4 63.4 286
62.8 86.3 299  76.8 24.6 279  31.6 94.0 300  26.9 73.8 287
18.9 68.4 284  99.4 37.2 285  79.1  3.3 277  34.9 74.7 289
 6.4 33.8 277  48.4 82.2 294  86.0 58.0 289  92.0 60.4 293
50.2 91.6 300  12.2 38.3 275  72.7 48.9 283  82.7 34.1 279
77.0 51.0 286  86.6 15.8 278  42.0 42.7 277  99.3  8.2 278
17.4 70.6 286  11.2 92.4 295  60.2 28.8 280  92.0 73.3 297
25.3 30.6 273  36.6  8.9 274  34.2  4.4 273  26.6 54.7 278
 1.7 27.4 278  49.6  1.1 275  62.8 89.3 301  28.0 49.3 279
51.2 75.1 293  59.3 93.5 304  83.6 90.5 304  79.4 87.0 302
```

```
78.0 28.3 281  16.8 19.1 272   9.1 81.2 292  23.7 55.8 277
75.5 21.3 279  64.4 43.3 279  38.9 98.9 303  22.5 87.9 293
96.7 37.9 285  92.3 93.9 308  16.9 25.4 273  15.2 61.5 283
73.8 94.0 306  57.4 97.2 305  73.2  4.9 276  39.2 82.3 294
95.7 99.4 315  66.0 98.4 306  95.3 26.9 283  45.4 75.3 291
64.8 15.4 276  69.8 55.4 284  36.3 74.9 290   9.9 22.2 276
65.8 13.9 276  13.0 82.0 293  95.6 77.2 301  32.5 55.6 279
45.8 35.5 275  62.2  6.6 274  25.2 51.2 279  92.4  8.1 277
40.5 35.3 273   9.9  3.9 271  43.5 44.0 278  68.6 61.3 287
64.2 77.5 296  57.6 81.6 294  69.5 64.7 291  64.3 95.1 304
 2.8 62.4 283  33.2 83.3 294  10.7 71.0 285  24.3 88.2 294
94.5 32.2 283  21.0 67.6 286  20.1 71.6 286  85.2 71.3 296
94.8 30.7 283  53.4 92.0 301  81.0 50.0 287  54.6 29.9 277
71.1 90.1 303  15.2  2.9 271  83.6 17.8 278  76.0 21.8 279
55.6 37.4 275  86.7 83.7 303  43.6 83.6 295  44.2 31.7 274
90.0 83.3 300   6.2  0.5 270  42.2 87.7 298  31.7  4.3 273
91.4 41.2 285  78.0 50.6 286  27.1 56.1 278  72.6 63.9 291
29.3 49.9 281  49.0 36.9 275  13.9 53.5 280  93.1 83.2 300
73.0 61.6 289  63.1 27.5 280  38.3 72.5 287  72.7 34.2 277
 6.9 32.3 274  17.1 58.6 280  19.6 94.6 297   2.7 36.5 276
34.5  5.5 275  98.6 95.9 313   9.1 71.1 285  88.6 55.8 287
26.8 78.5 289  64.8 66.6 292  59.7 25.7 280  47.3 70.2 288
 6.1 94.4 296  50.5 82.7 296   9.1 41.6 276  86.0 71.0 296
75.2 69.8 293  73.3 84.8 300  42.5 15.9 274  56.1 76.1 292
87.9 41.2 285  65.1  9.8 274  79.0 41.2 282  44.6 65.1 287
54.7 68.3 289  57.0 26.8 279   8.7 12.3 270  33.7 61.9 286
25.0 55.8 278  69.3 94.9 306  49.2 64.6 287  78.2 93.7 307
47.9 26.6 277  96.9 51.4 292  39.6 73.4 287  37.9 66.1 285
94.5 71.4 296  51.6 18.3 276  37.6 73.2 287  68.5 10.7 274
46.7  9.6 273  87.4 38.9 282  45.6 43.9 277  70.7 76.9 296
82.8 53.6 287  82.5 55.4 286  37.8  5.1 275  89.8 96.1 309
63.9  4.9 276   2.0 11.7 270  31.3 59.2 282  93.9 65.3 296
47.9 93.0 301  29.9 36.0 274  14.6 28.3 274  17.5 70.1 286
 2.6 68.5 282  23.1 12.0 268  36.8 20.4 273  80.9  9.0 276
39.2  0.0 274  26.2 44.3 276  81.9 12.9 277   3.2 21.4 272
76.9 76.7 297  88.6  7.7 277   9.7  8.4 273  26.7 91.5 296
73.8  6.1 276  33.7 39.3 276  64.0 58.4 286   5.7 91.2 295
85.8 93.8 307  85.8 39.1 281  93.9 63.4 295  53.1 46.3 278
51.9 42.9 277  16.8 75.7 288  29.2 66.9 285  37.4 72.5 287
;
```

The initial step is to explore the data set by inspecting the data spatial distribution. Run PROC VARIOGRAM, specifying the NOVARIOGRAM option in the COMPUTE statement as follows:

```
ods graphics on;

proc variogram data=ozoneSet;
   compute novariogram nhc=35;
   coord xc=East yc=North;
   var Ozone;
run;
```

The result is a scatter plot of the observed data shown in Output 131.2.1. The scatter plot suggests an almost uniform spread of the measurements throughout the prediction area. No direct inference can be made about the existence of a surface trend in the data. However, the apparent stratification of ozone values in the

northeast–southwest direction might indicate a nonrandom trend.

Output 131.2.1 Ozone Observation Data Scatter Plot

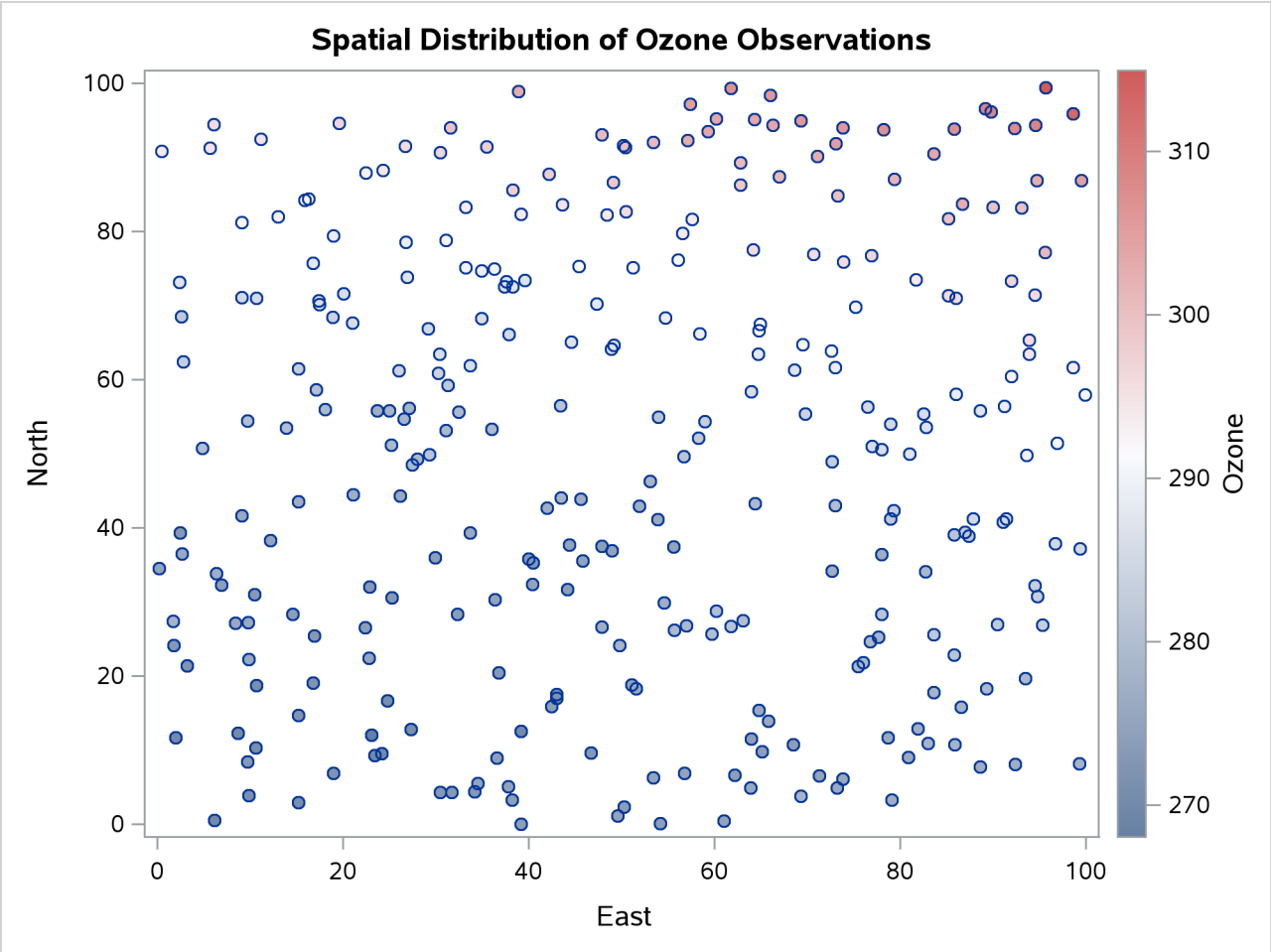

You need to define the size and count of the data classes by specifying suitable values for the LAGDISTANCE= and MAXLAGS= options, respectively. Compared to the smaller sample of thickness data used in "Getting Started: VARIOGRAM Procedure" on page 10962, the larger size of the ozoneSet data results in more densely populated distance classes for the same value of the NHCLASSES= option. After you experiment with a variety of values for the NHCLASSES= option, you can adjust LAGDISTANCE= to have a relatively small number. Then you can account for a large value of MAXLAGS= so that you obtain many sample semivariogram points within your data correlation range. Specifying these values requires some exploration, for which you might need to return to this point from a later stage in your semivariogram analysis. For illustration purposes you now specify NHCLASSES=35.

Your choice of NHCLASSES=35 yields the pairwise distance intervals table in Output 131.2.2 and the corresponding histogram in Output 131.2.3.

Output 131.2.2 Pairwise Distance Intervals Table

Pairwise Distance Intervals				
Lag Class	Bounds		Number of Pairs	Percentage of Pairs
0	0.00	2.01	52	0.12%
1	2.01	6.03	420	0.94%
2	6.03	10.06	815	1.82%
3	10.06	14.08	1143	2.55%
4	14.08	18.10	1518	3.38%
5	18.10	22.12	1680	3.75%
6	22.12	26.15	1931	4.31%
7	26.15	30.17	2135	4.76%
8	30.17	34.19	2285	5.09%
9	34.19	38.21	2408	5.37%
10	38.21	42.24	2551	5.69%
11	42.24	46.26	2444	5.45%
12	46.26	50.28	2535	5.65%
13	50.28	54.30	2487	5.55%
14	54.30	58.33	2460	5.48%
15	58.33	62.35	2391	5.33%
16	62.35	66.37	2302	5.13%
17	66.37	70.39	2285	5.09%
18	70.39	74.41	2079	4.64%
19	74.41	78.44	1786	3.98%
20	78.44	82.46	1640	3.66%
21	82.46	86.48	1493	3.33%
22	86.48	90.50	1243	2.77%
23	90.50	94.53	925	2.06%
24	94.53	98.55	710	1.58%
25	98.55	102.57	421	0.94%
26	102.57	106.59	274	0.61%
27	106.59	110.62	200	0.45%
28	110.62	114.64	120	0.27%
29	114.64	118.66	55	0.12%
30	118.66	122.68	35	0.08%
31	122.68	126.71	14	0.03%
32	126.71	130.73	11	0.02%
33	130.73	134.75	2	0.00%
34	134.75	138.77	0	0.00%
35	138.77	142.80	0	0.00%

Notice the overall high pair count in the majority of classes in Output 131.2.2. You can see that even for higher values of NHCLASSES= the classes are still sufficiently populated for your semivariogram analysis according to the rule of thumb stated in the section "Choosing the Size of Classes" on page 11023. Based on the displayed information in Output 131.2.3, you specify LAGDISTANCE=4 km. You can further experiment with smaller lag sizes to obtain more points in your sample semivariogram.

You can focus on the MAXLAGS= specification at a later point. The important step now is to investigate the presence of trends in the measurement. The following section makes a suggestion about how to remove

surface trends from your data and then continues the semivariogram analysis with the detrended data.

Output 131.2.3 Distribution of Pairwise Distances for Ozone Observation Data

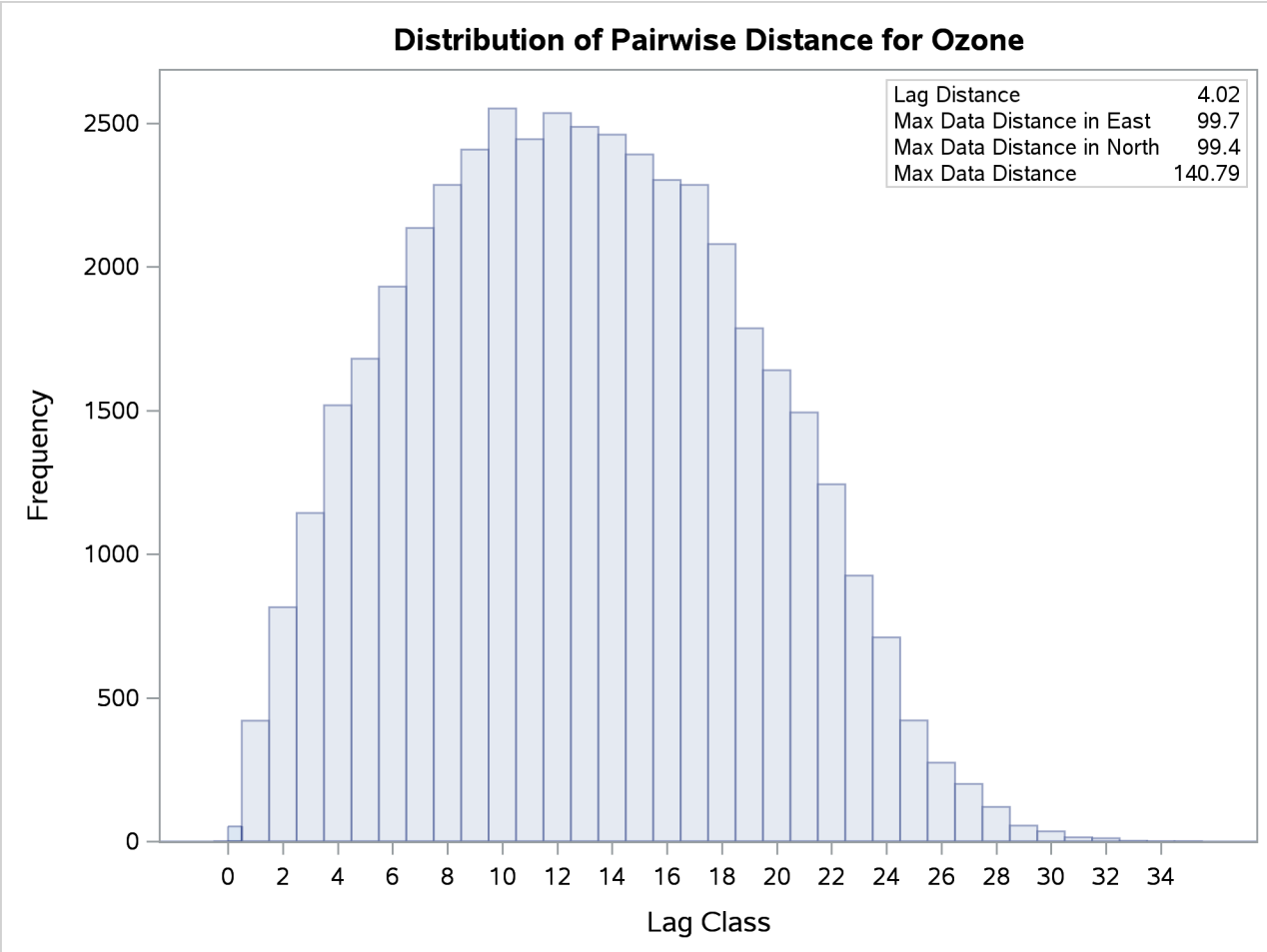

Analysis with Surface Trend Removal

You can use a SAS/STAT predictive modeling procedure to extract surface trends from your original data. If your goal is spatial prediction, you can continue processing the detrended data for the prediction tasks, and at the end you can reinstate the trend at the prediction locations to report your analysis results.

In general, the exact form of the trend is unknown, as discussed in the section "Empirical Semivariograms and Surface Trends" on page 11026. In this case, the spatial distribution of the measurements shown in Figure 131.2.1 suggests that you can use a quadratic model to describe the surface trend like the one that follows:

$$T(\text{East}, \text{North}) = f_0 + f_1\,[\text{East}] + f_2\,[\text{East}]^2 + f_3\,[\text{North}] + f_4\,[\text{North}]^2$$

The following statements show how to invoke the GLM procedure for your ozone data and how to extract the preceding trend from them:

```
proc glm data=ozoneSet plots=none;
   model ozone = East East*East North North*North;
   output out=gmout predicted=pred residual=ResidualOzone;
run;
```

Among other output, PROC GLM produces estimates for the parameters $f_0, \ldots, f_4$ in the preceding trend model. Output 131.2.4 shows the table with the parameter estimates. In this table, the coefficient f_0 corresponds to the intercept estimate, and the rest of the coefficients correspond to their matching variables; for example, the estimate in the line of "East*East" refers to f_2 in the preceding model. For more information about the syntax and the PROC GLM output, see Chapter 53, "The GLM Procedure."

Output 131.2.4 Parameter Estimates for the Surface Trend Model

Semivariogram Analysis in Anisotropic Case With Trend Removal

The GLM Procedure

Dependent Variable: Ozone

Parameter	Estimate	Standard Error	t Value	Pr > |t|
Intercept	270.6798273	0.40595731	666.77	<.0001
East	0.0065148	0.01360281	0.48	0.6323
East*East	0.0010726	0.00012987	8.26	<.0001
North	-0.0369159	0.01297491	-2.85	0.0047
North*North	0.0035587	0.00012659	28.11	<.0001

The detrending process leaves you with the GMOUT data set, which contains the ResidualOzone data residuals. This time you run PROC VARIOGRAM again with the NOVARIOGRAM option to inspect the detrended residuals, and with a request only for the observations plot, as follows:

```
proc variogram data=gmout plots(only)=observ;
   compute novariogram nhc=35;
   coord xc=East yc=North;
   var ResidualOzone;
run;
```

The requested observations plot is shown in Output 131.2.5.

Output 131.2.5 Ozone Residual Observation Data Scatter Plot

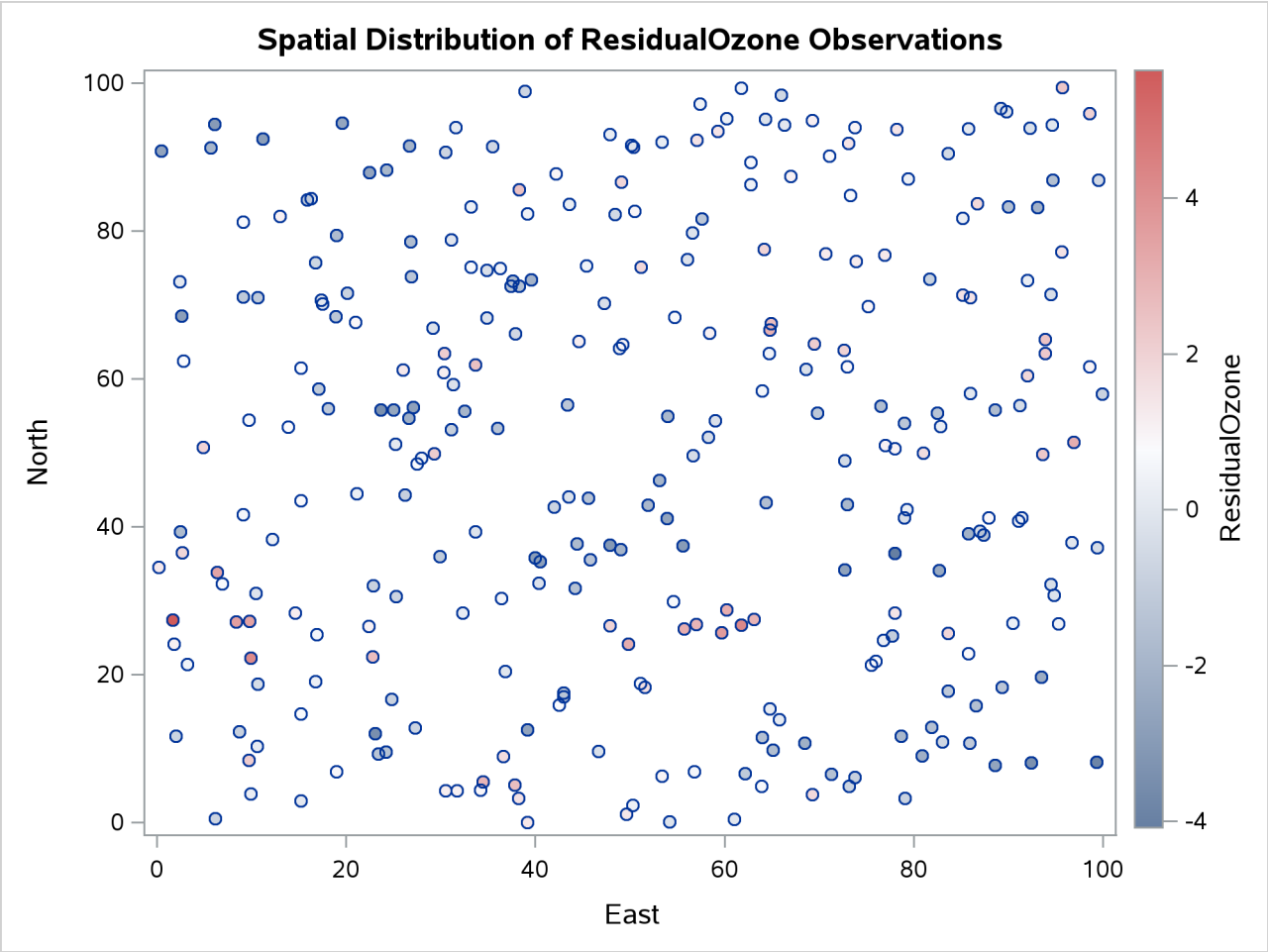

Before you proceed with the empirical semivariogram computation and model fitting, examine your data for anisotropy. This investigation is necessary to portray the spatial structure of your SRF accurately. If anisotropy exists, it manifests itself as different ranges or sills or both for the empirical semivariograms in different directions.

You want detail in your analysis, so you ask for the empirical semivariance in 12 directions by specifying NDIRECTIONS=12. Based on the NDIRECTIONS= option, empirical semivariograms are produced in increments of the base angle $\theta = 180°/12 = 15°$.

You also choose ANGLETOLERANCE=22.5 and BANDWIDTH=20. A different choice of values produces different empirical semivariograms, because these options can regulate the number of pairs that are included in a class. Avoid assigning values that are too small to these parameters so that you can allow for an adequate number of point pairs per class. At the same time, the higher the values of these parameters are, the more data pairs that come from closely neighboring directions are included in each lag. Therefore, values for the ANGLETOLERANCE= and BANDWIDTH= options that are too high pose a risk of losing information along the particular direction. The side effect occurs because you incorporate data pairs from a broader spectrum of angles; thus, you potentially amplify weaker anisotropy or weaken stronger anisotropy, as noted in the section "Angle Classification" on page 11017. You can experiment with different ANGLETOLERANCE= and BANDWIDTH= values to reach this balance with your data, if necessary.

With the following statements you ask to display only the SEMIVAR plots in the specified number of directions. Multiple empirical semivariograms are placed by default in panels, as Output 131.2.6 shows. If you want an individual plot for each angle, then you need to further specify the plot option SEMIVAR(UNPACK).

```
proc variogram data=gmout plot(only)=semivar;
   compute lagd=4 maxlag=16 ndir=12 atol=22.5 bandw=20;
   coord xc=East yc=North;
   var ResidualOzone;
run;
```

Output 131.2.6 Ozone Empirical Semivariograms with $0^\circ \leq \theta < 180^\circ$ and $\delta\theta = 15^\circ$

Output 131.2.6 *continued*

Output 131.2.6 *continued*

The panels in Output 131.2.6 suggest that in some of the directions, such as for $\theta = 0°$, the directional plots tend to exhibit a somewhat noisy structure. This behavior can be due to the pairs distribution across the particular direction. Specifically, based on the LAGDISTANCE= choice there might be insufficient pairs present in a class. Also, depending on the ANGLETOLERANCE= and BANDWIDTH= values, too many pairs might be considered from neighboring angles that potentially follow a modified structure. These are factors that can increase the variability in the semivariance estimate. A different explanation might lie in the existence of outliers in the data set; this aspect is further explored in "Example 131.5: A Box Plot of the Square Root Difference Cloud" on page 11084.

This behavior is relatively mild here and should not obstruct your goal to study anisotropy in your data. You can also perform individual computations in any direction. By doing so, you can fine-tune the computation parameters and attempt to obtain smoother estimates of the sample semivariance.

Further in this study, the directional plots in Output 131.2.6 suggest that during shifting from $\theta = 0°$ to $\theta = 90°$, the empirical semivariogram range increases. Beyond the angle $\theta = 90°$, the range starts decreasing again until the whole circle is traversed at 180° and small range values are encountered around the N–S direction at $\theta = 0°$. The sill seems to remain overall the same. This analysis suggests the presence of anisotropy in the ozone concentrations, with the major axis oriented at about $\theta = 90°$ and the minor axis situated perpendicular to the major axis at $\theta = 0°$.

The multidirectional analysis requires that for a given LAGDISTANCE= you also specify a MAXLAGS= value. Since the ozone correlation range might be unknown (as assumed here), you can apply the rule of thumb that suggests use of the half-extreme data distance in the direction of interest, as explained in the section "Spatial Extent of the Empirical Semivariogram" on page 11024. Following the information displayed in Output 131.2.3, for different directions this distance varies between $99.4/2 = 49.7$ and $140.8/2 = 70.4$ km. In turn, the pairwise distances table in Output 131.2.2 indicates that within this range of distances you can specify MAXLAGS= to be between 12 and 17 lags. In this example you specify MAXLAGS=16.

At this point you are ready to continue with fitting theoretical semivariogram models to the empirical semivariogram in the selected directions of $\theta = 0°$ and $\theta = 90°$. By trying out different models, you see that an exponential one is suitable for your empirical data:

$$\gamma_z(h) = c_0 \left[1 - \exp\left(-\frac{h}{a_0} \right) \right]$$

For the purpose of the present example, it is reasonable to assume a constant nugget effect equal to zero, based on the empirical semivariograms shown in Output 131.2.6. The same output suggests that the model scale is likely to be above 2, and that the range might be relatively small in $\theta = 0°$. You specify the PARMS statement to set initial values for the exponential model parameters and account for these considerations.

In particular, you assign an initial value of zero to the nugget effect. Then you request a grid search for the range and scale parameters, so that the optimal initial values set is selected for the parameter estimation in each of the two angles $\theta = 0°$ and $\theta = 90°$. By inspecting the empirical semivariograms in Output 131.2.6, you specify the value list 2, 2.5, and 3 for the scale, and the values from 5 to 25 with a step of 10 for the range. In addition, you specify the parameter 1 in the HOLD= option to designate the nugget effect parameter as a constant. According to these specifications, you use the following statements:

```
proc variogram data=gmout plot(only)=fit;
   compute lagd=4 maxlag=16;
   directions 0(22.5,10) 90(22.5,10);
   coord xc=East yc=North;
   model form=exp;
   parms (0.) (2 to 3 by 0.5) (5 to 25 by 10) / hold=(1);
   var ResidualOzone;
run;

ods graphics off;
```

The VARIOGRAM procedure repeats the fitting process for each one of the selected directions. First, in $\theta = 0°$ the parameter search table in Output 131.2.7 shows you which value combinations are tested initially to choose the one that gives the lowest objective function value.

Output 131.2.7 Parameter Search for the Selected Direction $\theta = 0°$

Semivariogram Analysis in Anisotropic Case With Trend Removal

The VARIOGRAM Procedure

Dependent Variable: ResidualOzone
Angle: 0
Current Model: Exponential

Parameter Search

Set	Nugget	Scale	Range	Objective Function
1	0	2	5	391.06593
2	0	2	15	1740.0
3	0	2	25	5167.5
4	0	2.5	5	64.86565
5	0	2.5	15	664.03665
6	0	2.5	25	2480.5
7	0	3	5	72.86743
8	0	3	15	305.53306
9	0	3	25	1305.0

From this search, the combination of scale equal to 2.5 and a range of size 5 is passed as initial values to the model fitting process. This result is reflected in the model information table shown in Output 131.2.8.

Output 131.2.8 Model Initial Values for the Selected Direction $\theta = 0°$

Model Information

Parameter	Initial Value	Status
Nugget	0	Fixed
Scale	2.5000	
Range	5.0000	

Fitting is successful, and among the output objects you can see the estimated parameters and the fit summary tables for the direction $\theta = 0°$ in Output 131.2.9.

Output 131.2.9 Weighted Least Squares Fitting Parameter Estimates and Summary for the Selected Direction $\theta = 0°$

Parameter Estimates

Parameter	Estimate	Approx Std Error	DF	t Value	Approx Pr > |t|
Scale	2.6657	0.03830	15	69.60	<.0001
Range	3.7277	0.5609	15	6.65	<.0001

Fit Summary

Model	Weighted SSE	AIC
Exp	43.35103	19.91399

A corresponding parameter search takes place for the direction $\theta = 90°$. The respective table and the choice of initial values for fitting in the direction $\theta = 90°$ are shown in Output 131.2.10.

Output 131.2.10 Parameter Search and Model Initial Values for the Selected Direction $\theta = 90°$

Parameter Search

Set	Nugget	Scale	Range	Objective Function
1	0	2	5	302.54551
2	0	2	15	635.93338
3	0	2	25	1996.0
4	0	2.5	5	95.09939
5	0	2.5	15	104.56776
6	0	2.5	25	662.06813
7	0	3	5	155.50670
8	0	3	15	20.48482
9	0	3	25	190.30599

Model Information

Parameter	Initial Value	Status
Nugget	0	Fixed
Scale	3.0000	
Range	15.0000	

Output 131.2.11 displays the estimated parameters and the fit summary for the direction $\theta = 90°$.

Output 131.2.11 Weighted Least Squares Fitting Parameter Estimates and Summary for the Selected Direction $\theta = 90°$

Parameter Estimates

Parameter	Estimate	Approx Std Error	DF	t Value	Approx Pr > |t|
Scale	2.9199	0.07007	15	41.67	<.0001
Range	14.7576	0.9530	15	15.49	<.0001

Fit Summary

Model	Weighted SSE	AIC
Exp	19.12246	6.00005

The fitted and empirical semivariograms for the selected directions are displayed in the panel of Output 131.2.12.

Output 131.2.12 Fitted Theoretical and Empirical Semivariogram for the Ozone Data in the $\theta = 0°$ and $\theta = 90°$ Directions

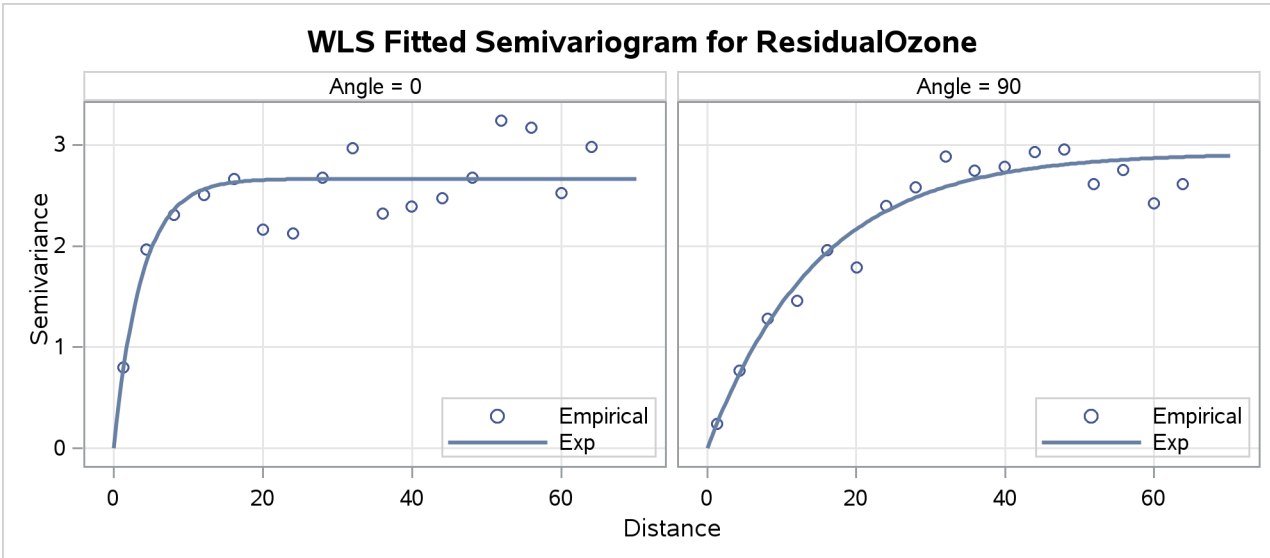

Conclusively, your semivariogram analysis on the detrended ozone data suggests that the ozone SRF exhibits anisotropy in the perpendicular directions of N–S ($\theta = 0°$) and E–W ($\theta = 90°$).

The sills in the two directions of anisotropy are similar in size. By inspecting again the empirical semivariograms in Output 131.2.6, you could make the reasonable assumption that you have a case of geometric anisotropy, where the range in the major axis is about 4.5 times larger than the minor axis range. If you would like to use these PROC VARIOGRAM results for predictions, then you would need to specify a single scale value for the geometric anisotropy sill. In this case you could choose an arbitrary value for the constant scale from the narrow interval formed by the estimated scales in the previous results. For example, you can specify the PARMS statement modified as shown in the following statement to approximate a common scale for the semivariance in all directions:

```
parms (0.) (2.7) (5 to 25 by 10) / hold=(1,2);
```

As an alternative, you can use PROC VARIOGRAM to fit an exponential model to all different angles examined in this example, and then select the constant scale value to be the mean of the scales across all directions.

Example 131.3: Analysis without Surface Trend Removal

(View the complete code for this example (vroex3.sas) in the example repository.)

This example uses PROC VARIOGRAM without removing potential surface trends in a data set in order to investigate a distinguished spatial direction in the data. In doing so, this example also serves as a guide to examine under which circumstances you might be able to bypass the effect of a trend on a semivariogram. Typically though, for theoretical semivariogram estimations you follow the analysis presented in "Example 131.2: An Anisotropic Case Study with Surface Trend in the Data" on page 11058.

As explained in the section "Details: VARIOGRAM Procedure" on page 11008, when you compute the empirical semivariance for data that contain underlying surface trends, the outcome is the pseudo-semivariance.

Pseudo-semivariograms are not estimates of the theoretical semivariogram; hence, they provide no information about the spatial continuity of your SRF.

However, in the section "Empirical Semivariograms and Surface Trends" on page 11026 it is mentioned that you might still be able to perform a semivariogram analysis with potentially non-trend-free data, if you suspect that your measurements might be trend-free across one or more specific directions. The example demonstrates this approach.

Reconsider the ozone data presented at the beginning of "Example 131.2: An Anisotropic Case Study with Surface Trend in the Data" on page 11058. The spatial distribution of the data is shown in Figure 131.2.1, and the pairwise distance distribution for NHCLASSES=35 is illustrated in Figure 131.2.3. This exploratory analysis suggested a LAGDISTANCE=4 km, and Figure 131.2.2 indicated that for this LAGDISTANCE= you can consider a value of MAXLAGS=16.

Recall from the section "Empirical Semivariograms and Surface Trends" on page 11026 that you need to investigate the empirical semivariogram of the data in a few different directions in order to identify a trend-free direction. If such a direction exists, then you can proceed with this special type of analysis. The following statements employ NDIRECTIONS=8 to examine eight directions:

```
ods graphics on;

proc variogram data=ozoneSet plot(only)=semivar;
   compute lagd=4 maxlag=16 ndirections=8 robust;
   coord xc=East yc=North;
   var Ozone;
run;
```

By default, the range of 180° is divided into eight equally distanced angles: $\theta = 0°$, $\theta = 22.5°$, $\theta = 45°$, $\theta = 67.5°$, $\theta = 90°$, $\theta = 112.5°$, $\theta = 135°$, and $\theta = 157.5°$. The resulting empirical semivariograms for these angles are shown in Output 131.3.1.

Output 131.3.1 Ozone Empirical Semivariograms with $0° \leq \theta < 180°$ and $\delta\theta = 22.5°$

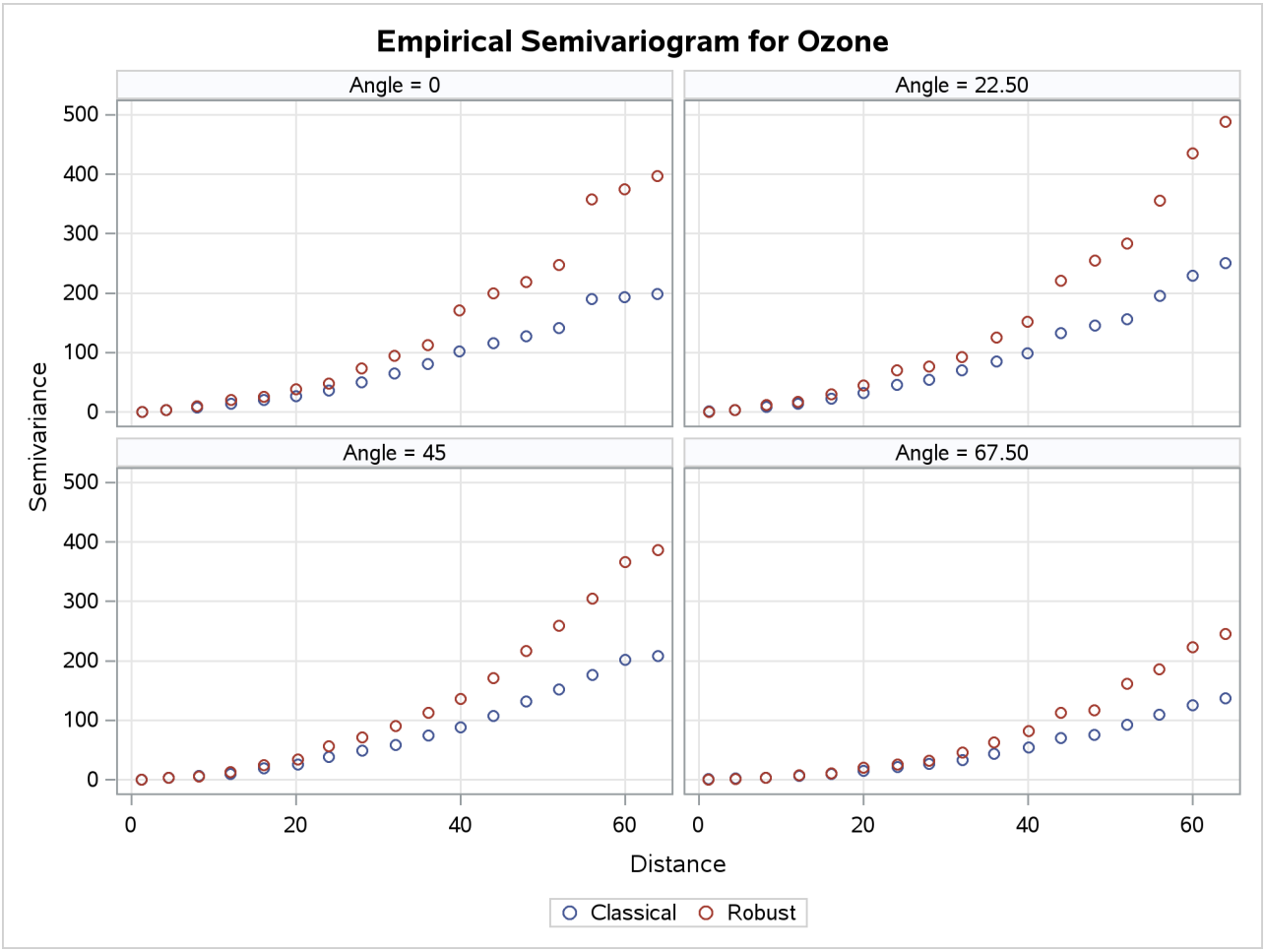

Output 131.3.1 *continued*

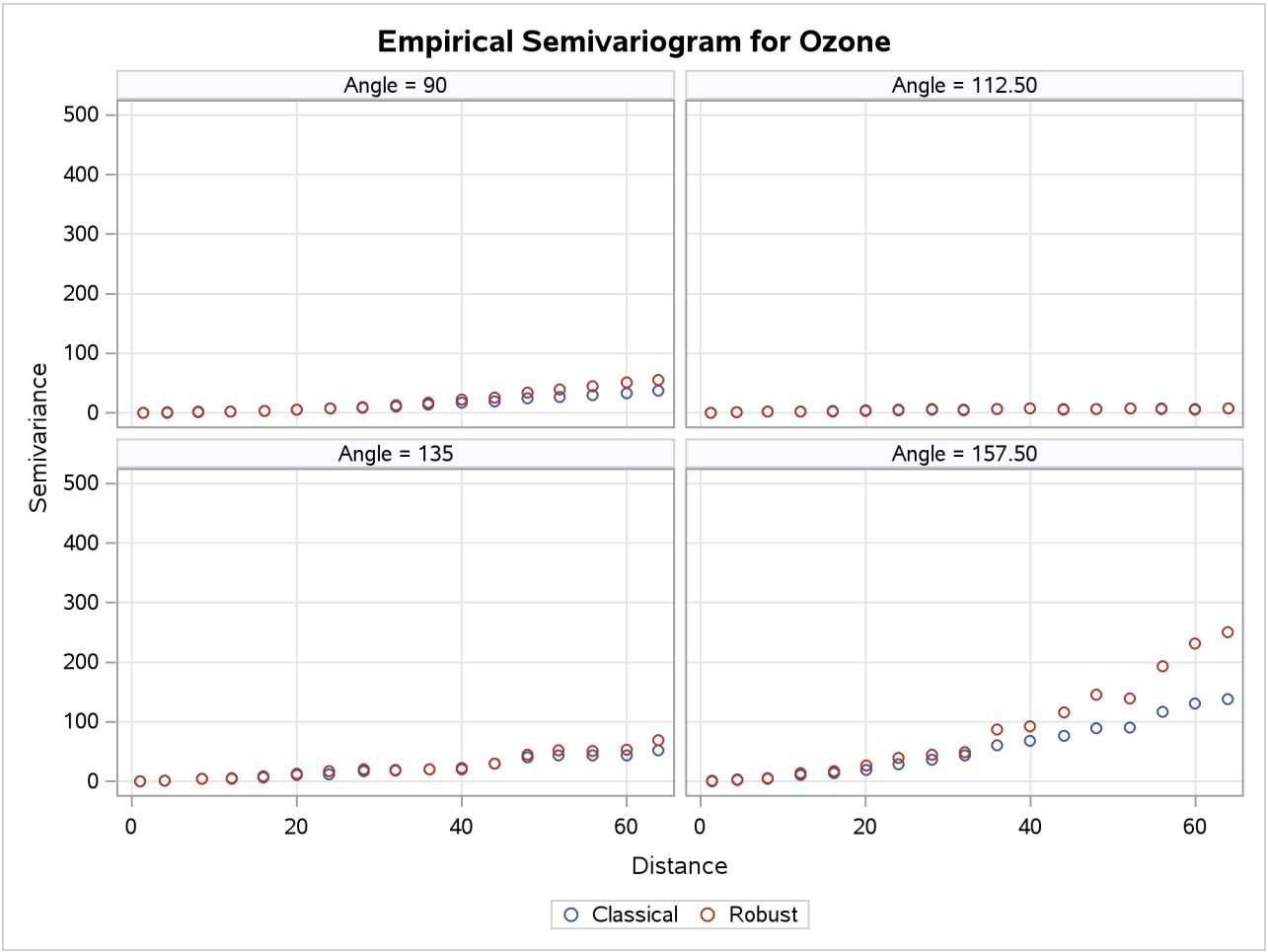

The figures in Output 131.3.1 suggest an overall continuing increase with distance of the semivariance in all directions. As explained in the section "Theoretical Semivariogram Models" on page 11008, this can be an indication of systematic trends in the data. However, the direction of $\theta = 112.5°$ clearly indicates that the increase rate, if any, is smaller than the corresponding rates across the rest of the directions. You then want to search whether there exists a trend-free direction in the neighborhood of this angle.

Run PROC VARIOGRAM again, specifying several directions within an interval of angles where you want to close in and you suspect the existence of a trend-free direction. In the following step you specify ANGLETOL=15°, which is smaller than the default value of 22.5°, and you also specify BANDWIDTH=10 km. The smaller values help with minimization of the interference with neighboring directions, as discussed in the section "Angle Classification" on page 11017.

The aforementioned considerations are addressed in the following statements:

```
proc variogram data=ozoneSet plot(only)=semivar(cla);
   compute lagd=4 maxlag=16 robust;
   directions 100(15,10) 103(15,10)
              106(15,10) 108(15,10)
              110(15,10) 112(15,10)
              115(15,10) 118(15,10);
```

```
    coord xc=East yc=North;
    var Ozone;
  run;
```

Your analysis has brought you to examine a narrow strip of angles within $\theta = 100°$ and $\theta = 118°$. The pseudo-semivariograms in Output 131.3.2 and Output 131.3.3 indicate that at the boundaries of this strip, the angles display increasing semivariance with distance. On the other hand, within this interval there are directions across which the semivariance is tentatively reaching a sill, and these are potential candidates to be trend-free directions.

Output 131.3.2 Ozone Empirical Semivariograms in 100°, 103°, 106°, and 108°

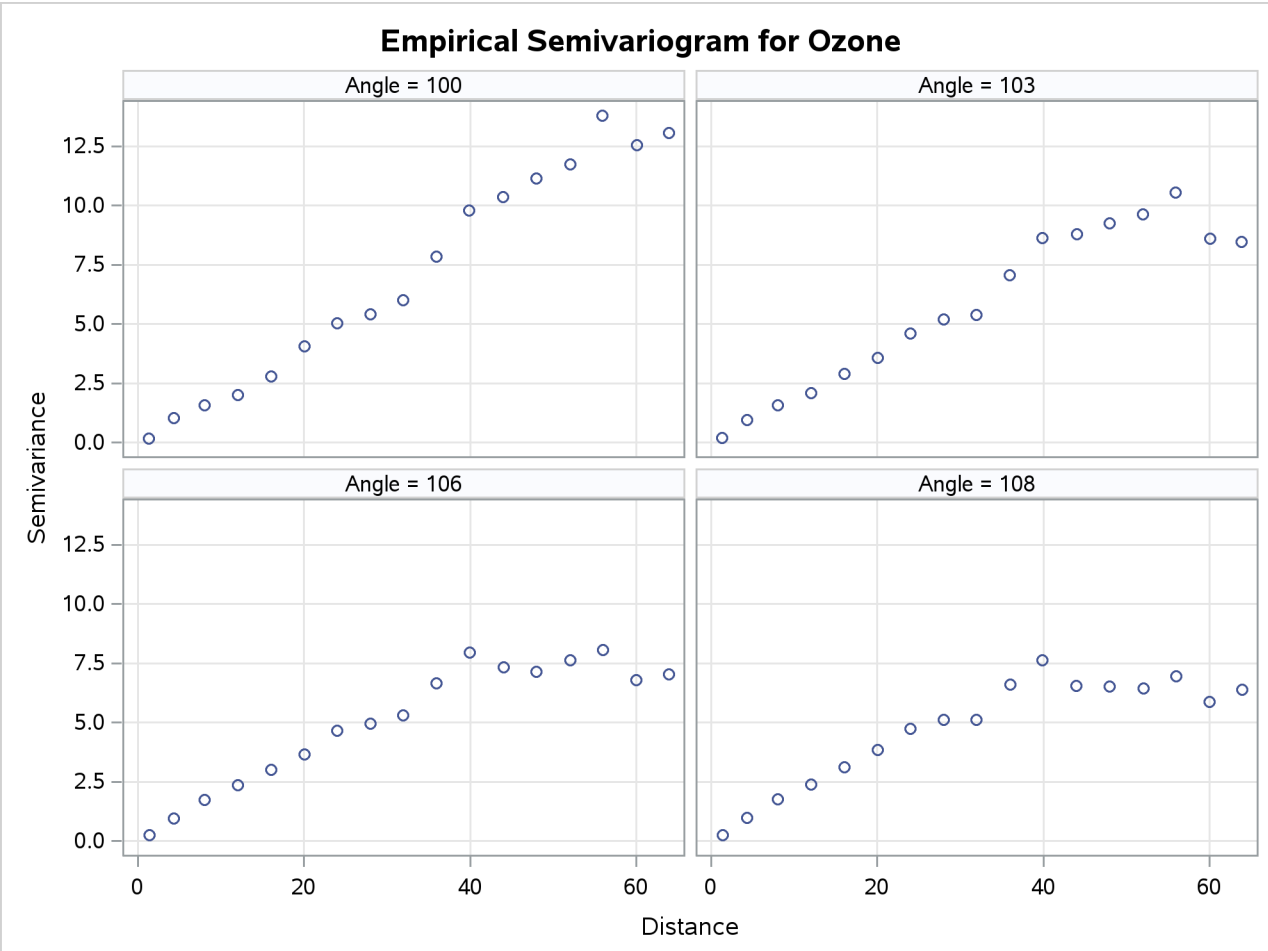

Output 131.3.3 Ozone Empirical Semivariograms in 110°, 112°, 115°, and 118°

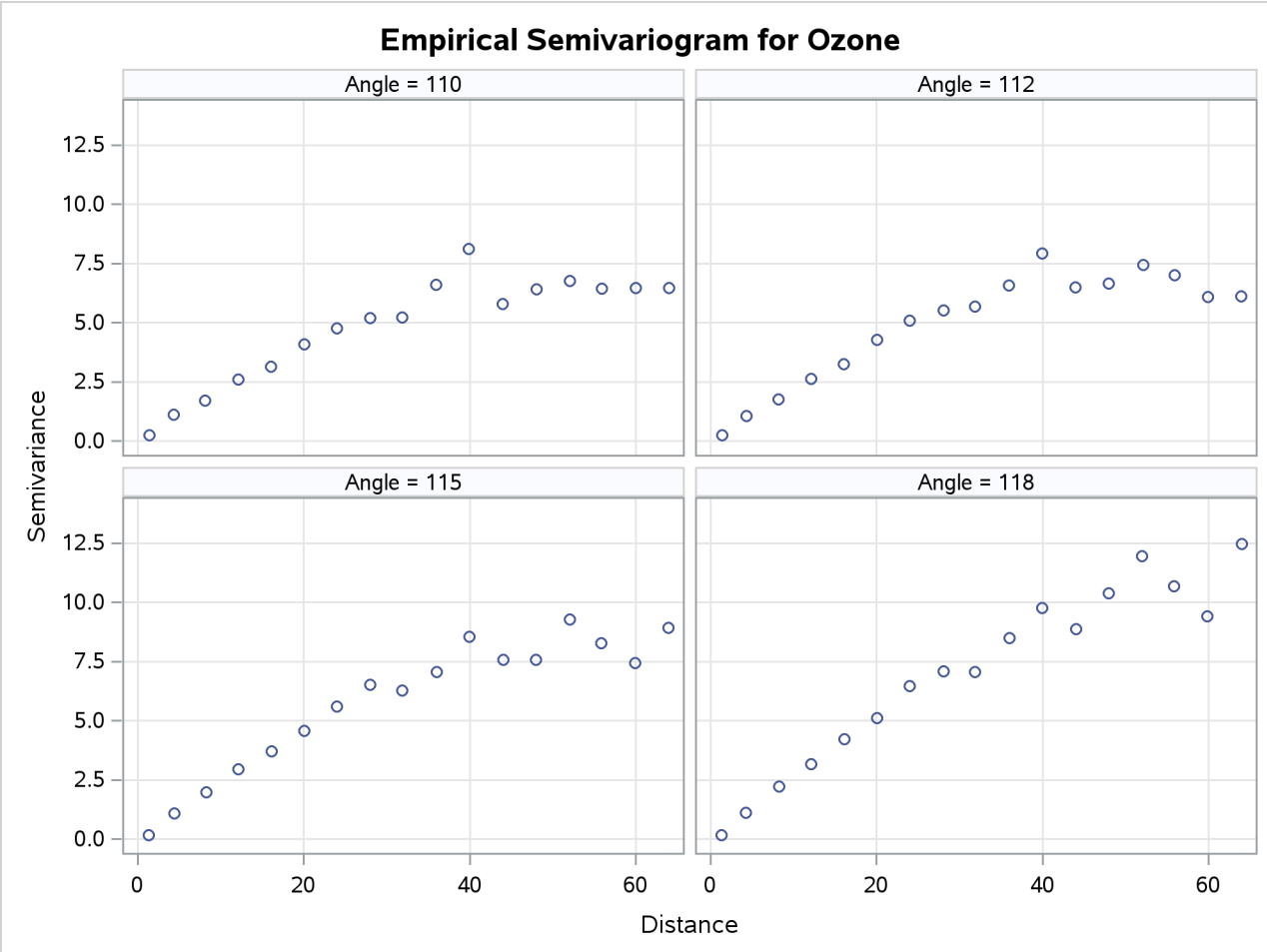

You can further investigate this angle spectrum in more detail. For example, you can monitor additional angles in between, or use a smaller LAGDISTANCE= and increased MAXLAGS= values to single out the most qualified candidate. For the purpose of this example, you can consider the direction $\theta = 108°$ to very likely be the trend-free one you are looking for.

From a physical standpoint, the trend-free direction, if it exists, is expected to be perpendicular to the direction of the maximum dip in the values of the ozone field, as mentioned in the section "Empirical Semivariograms and Surface Trends" on page 11026. If you cross-examine the ozone data distribution in Output 131.2.1, the figure suggests that this direction exists and is slightly tilted clockwise with respect to the E–W axis. This direction emerges from the mild stratification of the ozone values in your data distribution. The ozone concentrations across it are similar when compared to surrounding directions, and as such, it has been identified as a trend-free direction.

Your next step is to obtain the empirical semivariogram in the suspected trend-free direction of $\theta = 108°$ and to perform a theoretical model fit.

The semivariance in Output 131.3.2 exhibits a slow, almost linear rise at short distances and seems to be reaching the sill fast, rather than asymptotically. You can accommodate this behavior by using the spherical

model

$$\gamma_z(h) = \begin{cases} c_n + {\sigma_0}^2 \left[\frac{3}{2}\frac{h}{a_0} - \frac{1}{2}(\frac{h}{a_0})^3 \right], & \text{for } 0 < h \leq a_0 \\ c_0, & \text{for } a_0 < h \end{cases}$$

where $\gamma_z(0) = 0$ and $a_0 > 0$. The empirical semivariograms also suggest that there does not seem to be a nugget effect. Assume that in this example you are interested in what the fitting process concludes about the nugget effect, so you skip the NUGGET= option in the MODEL statement. You also let PROC VARIOGRAM provide initial values for the rest of the model parameters. Eventually, you use the PLOTS option to inspect the classical and robust empirical semivariograms in the selected direction and to produce a plot of the fitted model. The following statements implement these considerations:

```
proc variogram data=ozoneSet plot(only)=(semivar fit);
   compute lagd=4 maxlag=16 robust cl;
   directions 108(15,10);
   coord xc=East yc=North;
   model form=sph;
   var ozone;
run;

ods graphics off;
```

The classical and robust empirical semivariograms in the selected direction $\theta = 108°$ are displayed in Figure 131.3.4.

Output 131.3.4 Ozone Classical and Robust Empirical Semivariograms in $\theta = 108°$

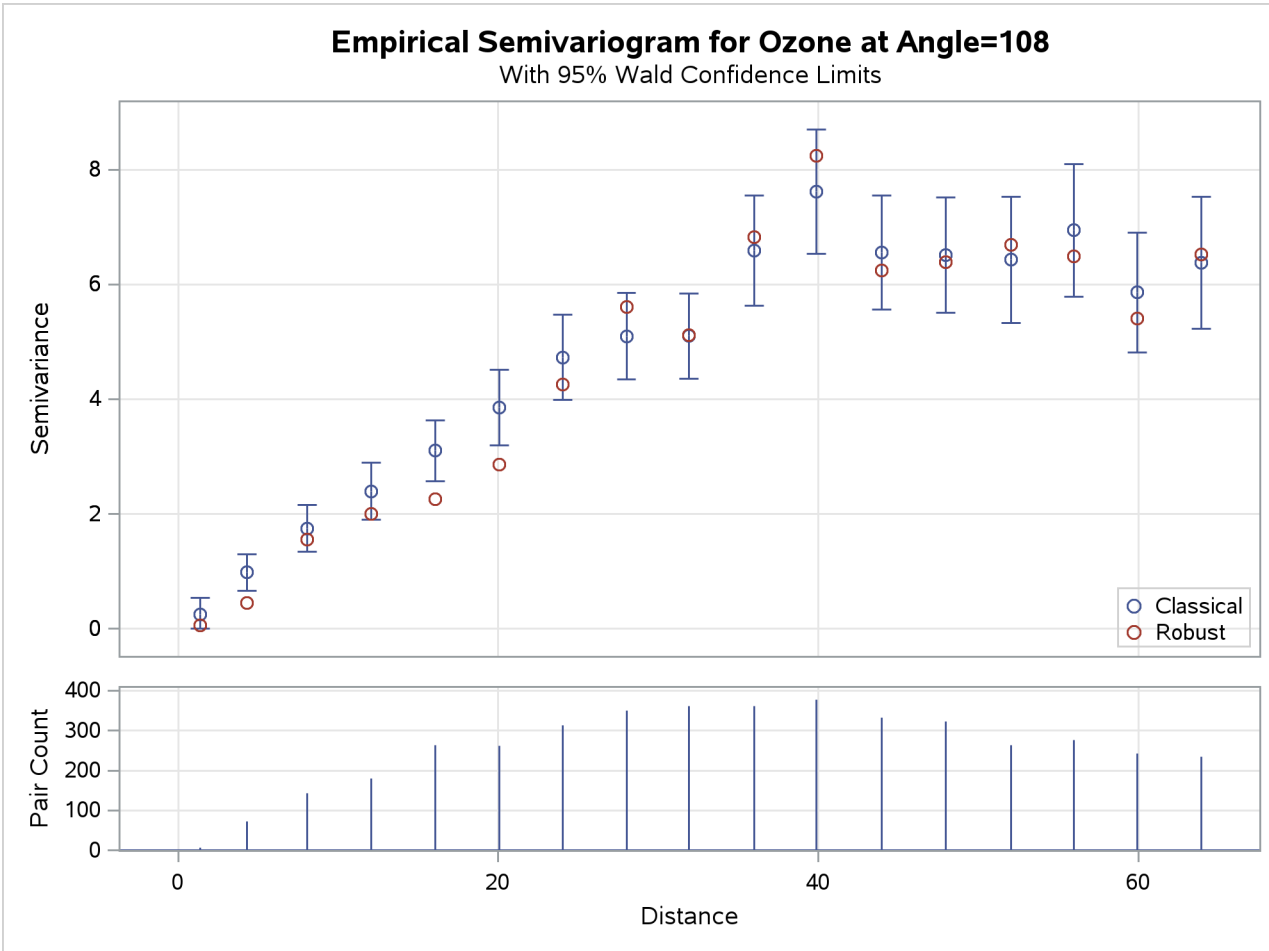

The output continues with information about the fitting process, which terminates successfully and produces the estimated parameters and the fit summary tables shown in Output 131.3.5. The near-zero nugget parameter estimate indicates that you can consider the process to be practically free of nugget effect.

Output 131.3.5 Weighted Least Squares Fitting Parameter Estimates and Summary in $\theta = 108°$

Parameter Estimates

Parameter	Estimate	Approx Std Error	DF	t Value	Approx Pr > |t|
Nugget	0.006260	0.09449	14	0.07	0.9481
Scale	6.6791	0.1741	14	38.37	<.0001
Range	47.3012	2.0776	14	22.77	<.0001

Fit Summary

Model	Weighted SSE	AIC
Sph	13.13869	1.61991

The fitted and empirical semivariograms for the selected direction $\theta = 108°$ are displayed in Output 131.3.6.

Output 131.3.6 Fitted Theoretical and Empirical Semivariogram for the Ozone Data in $\theta = 108°$

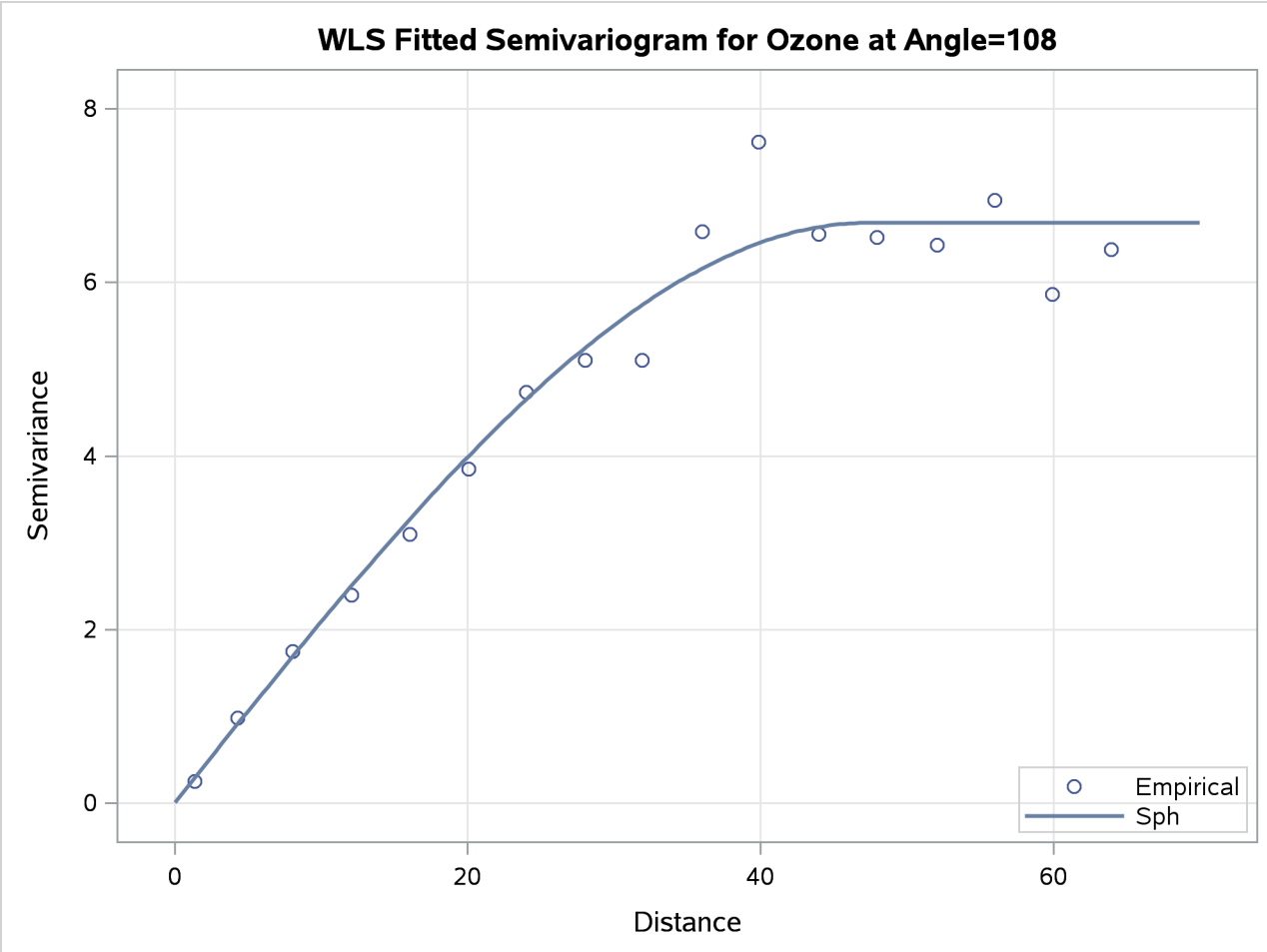

A comparative look at the empirical and fitted semivariograms in Output 131.3.6 and Output 131.2.12 suggests that the analysis of the trend-free ResidualOzone produces a different outcome from that of the original Ozone values. In fact, a more suitable comparison can be made between the semivariograms in the assumed trend-free direction $\theta = 108°$ of the current scenario and the one shown in Output 131.2.6 in the nearly identical direction $\theta = 105°$. It might seem unreasonable that these two semivariograms are produced both in the same ozone study and in a narrow band of directions free of apparent surface trends, yet they bear no resemblance. However, the lack of similarity in these plots stems from operating on two different data sets where the outcome depends on the actual data values.

More specifically, the semivariogram analysis treats the trend-free ozone set and the original ozone measurements as different quantities. The process of detrending the original Ozone values is a transformation of these values into the trend-free values of ResidualOzone. Any existing spatial correlation in the original data is not necessarily retained within the transformed data. Depending on the transformation features, the emerging data set has its own characteristics, as demonstrated in this example.

A final remark concerns the issue of isotropy. Based on the details presented in the section "Empirical Semivariograms and Surface Trends" on page 11026, your knowledge of the spatial structure of the ozoneSet data set is limited to the selected trend-free direction you indicated in the present example. You can generalize this outcome for all spatial directions only if you consider the hypothesis of isotropy in the ozone field to be

reasonable. However, you cannot infer the assumption of anisotropy in the present example based on the analysis in the section "Analysis with Surface Trend Removal" on page 11062. Again, the reason is that you currently use the observed Ozone values, whereas the ResidualOzone data in the previous example emerged from a transformation of the current data. Hence, you have essentially two data sets that do not necessarily share the same properties.

Example 131.4: Covariogram and Semivariogram

(View the complete code for this example (vroex4.sas) in the example repository.)

The covariance that was reviewed in the section "Stationarity" on page 11015 is an alternative measure of spatial continuity that can be used instead of the semivariance. In a similar manner to the empirical semivariance that was presented in the section "Theoretical and Computational Details of the Semivariogram" on page 11013, you can also compute the empirical covariance. The covariograms are plots of this quantity and can be used to fit permissible theoretical covariance models, in correspondence to the semivariogram analysis presented in the section "Theoretical Semivariogram Models" on page 11008. This example displays a comparative view of the empirical covariogram and semivariogram, and examines some additional aspects of these two measures.

You consider 500 simulations of an SRF $Z(\boldsymbol{s})$ in a square domain of 100×100 (10^6 km^2). The following DATA step defines the data locations:

```
title 'Covariogram and Semivariogram';

data dataCoord;
   retain seed 837591;
   do i=1 to 100;
      East = round(100*ranuni(seed),0.1);
      North = round(100*ranuni(seed),0.1);
      output;
   end;
run;
```

For the simulations you use PROC SIM2D, which produces Gaussian simulations of SRFs with user-specified covariance structure—see Chapter 112, "The SIM2D Procedure." The Gaussian SRF implies full knowledge of the SRF expected value $\mathrm{E}[Z(\boldsymbol{s})]$ and variance $\mathrm{Var}[Z(\boldsymbol{s})]$ at every location $\boldsymbol{s}$. The following statements simulate an isotropic, second-order stationary SRF with constant expected value and variance throughout the simulation domain:

```
proc sim2d outsim=dataSims;
   simulate numreal=500 seed=79750
            nugget=2 scale=6 range=10 form=exp;
   mean 30;
   grid gdata=dataCoord xc=East yc=North;
run;
```

Here, the SIMULATE statement accommodates the simulation parameters. The NUMREAL= option specifies that you want to perform 500 simulations, and the SEED= option specifies the seed for the simulation random number generator. You use the MEAN statement to specify the expected value $\mathrm{E}[Z(\boldsymbol{s})] = 30$ units of Z. You also specify two variance components. The first is the nugget effect, and you use the NUGGET= option to set it to $c_n = 2$. The second is the partial sill ${\sigma_0}^2 = 6$ that you specify with the SCALE= option. The

two variance components make up the total SRF variance $\mathrm{Var}[Z(\boldsymbol{s})] = c_n + {\sigma_0}^2 = 8$. You assume an exponential covariance structure to describe the field spatial continuity, where ${\sigma_0}^2$ is the sill value and its range $a_0 = 10$ km (effective range $a_\epsilon = 3a_0 = 30$ km) is specified by the RANGE= option. The option FORM= specifies the covariance structure type.

The empirical semivariance and covariance are computed by the VARIOGRAM procedure, and are available either in the ODS output semivariogram table (as variables Semivariance and Covariance, respectively) or in the OUTVAR= data set. In the following statements you obtain these variables by using the OUTVAR= data set of the VARIOGRAM procedure:

```
proc variogram data=dataSims outv=outv noprint;
   compute lagd=3 maxlag=18;
   coord xc=gxc yc=gyc;
   by _ITER_;
   var svalue;
run;
```

For each distance lag you take the average of the empirical measures over the number of simulations. PROC SORT prepares the input data for PROC MEANS, which produces these averages and stores them in the dataAvgs data set. This sequence is performed with the following statements:

```
proc sort data=outv;
   by lag;
run;

proc means data=outv n mean noprint;
   var Distance variog covar;
   by lag;
   output out=dataAvgs mean(variog)=Semivariance
                       mean(covar)=Covariance
                       mean(Distance)=Distance;
run;
```

The SGPLOT procedure creates the plot of the average empirical semivariogram and covariogram, as in the following statements:

```
proc sgplot data=dataAvgs;
   title "Empirical Semivariogram and Covariogram";
   xaxis label = "Distance" grid;
   yaxis  label = "Semivariance" min=-0.5 max=9 grid;
   y2axis label = "Covariance"   min=-0.5 max=9;
   scatter y=Semivariance x=Distance /
           markerattrs = GraphData1
           name='Semivar'
           legendlabel='Semivariance';
   scatter y=Covariance x=Distance /
           y2axis
           markerattrs = GraphData2
           name='Covar'
           legendlabel='Covariance';
   discretelegend 'Semivar' 'Covar';
run;
```

The plot of the average empirical semivariance and covariance of the preceding analysis is shown in Output 131.4.1. The high number of simulations led to averages of empirical continuity measures that

accurately approximate the simulated SRF characteristics. Specifically, the empirical semivariogram and covariogram both exhibit clearly exponential behavior. The semivariogram sill is approximately at the specified variance $\mathrm{Var}[Z(\boldsymbol{s})] = 8$ of the SRF.

The simulated SRF is second-order stationary, so you expect at each lag the sum of the empirical semivariance and covariance to approximate the field variance $\mathrm{Var}[Z(\boldsymbol{s})]$, as explained in the section "Stationarity" on page 11015. This behavior is evident in Output 131.4.1.

This example concludes with a discussion of basic reasons why the empirical semivariogram analysis is commonly preferred to the empirical covariance analysis. A first reason comes from the assumptions that are necessary to compute each of these two measures. The condition of intrinsic stationarity that is required in order to define the empirical semivariogram is less restrictive than the condition of second-order stationarity that is required in order to consider the covariance function as a parameter of the process.

Also, an empirical semivariogram can indicate whether a nugget effect is present in your data sample, whereas the empirical covariogram itself might not reveal this information. This point is illustrated in Output 131.4.1, where you expect to see that $C(\mathbf{0}) = \mathrm{Var}[Z(\boldsymbol{s})]$, but the empirical covariogram cannot have a point at exactly $\boldsymbol{h} = \mathbf{0}$. A practical way to investigate for a nugget effect when you use empirical covariograms is as follows: recall that the OUTVAR= data set provides you with the sample variance (shown in the COVAR column for LAG=–1), as the following statement shows:

```
/* Obtain the sample variance from the data set ----------------*/

proc print data=dataAvgs (obs=1);
run;
```

Output 131.4.1 Average Empirical Semivariogram and Covariogram from 500 Simulations

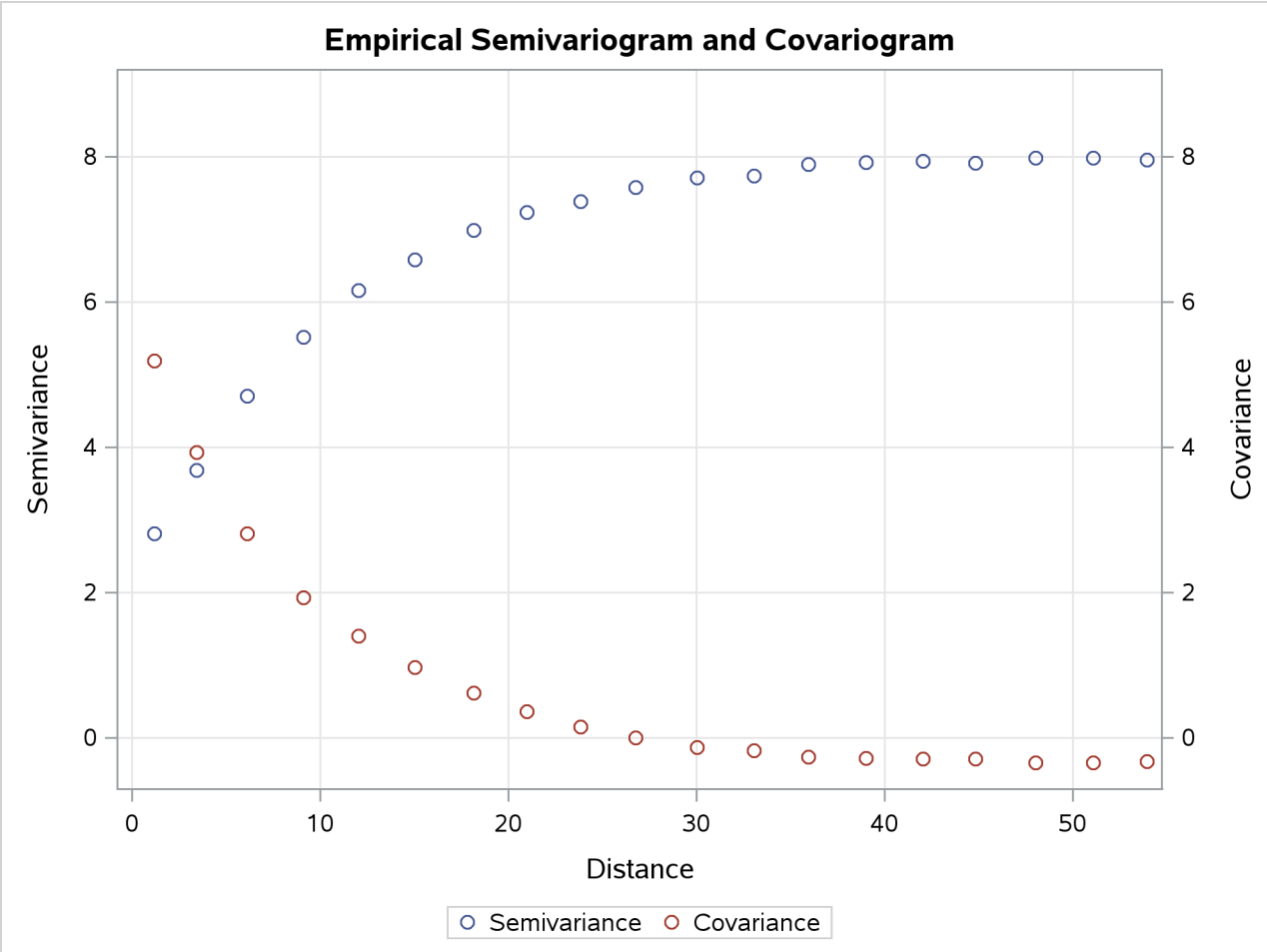

Output 131.4.2 is a partial output of the dataAvgs data set, which contains averages of the OUTVAR= data set and shows the computed average $C(\mathbf{0})$ in the Covariance column. The combination of the empirical covariogram and the $C(\mathbf{0})$ value can help you fit a theoretical covariance model that includes any nugget effect, if present. See also the discussion in Schabenberger and Gotway (2005, section 4.2.2) about the Matérn definition of the covariance function that is related to this issue. In particular, this definition provides for an additional variance component in the covariance expression at $\boldsymbol{h} = \mathbf{0}$ to account for the corresponding nugget effect in the semivariogram.

Output 131.4.2 Partial Outcome of the dataAvgs Data Set

Empirical Semivariogram and Covariogram

Obs	LAG	_TYPE_	_FREQ_	Semivariance	Covariance	Distance
1	-1	0	500	.	7.74832	.

In addition to the preceding points, if the SRF is nonstationary, the empirical semivariogram indicates that the SRF variance increases with distance $\boldsymbol{h}$, as Output 131.3.1 shows in "Example 131.3: Analysis without Surface Trend Removal" on page 11071. In that case it makes no sense to compute the empirical covariogram.

Specifically, the covariogram could provide you with an estimate of the sample variance, which is not sufficient to indicate that the SRF might not be stationary (see also Chilès and Delfiner 1999, p. 31).

Finally, the definitions of the empirical semivariance and covariance in the section "Theoretical and Computational Details of the Semivariogram" on page 11013 clearly show that the sample mean $\bar{Z}$ and the SRF expected value $\mathrm{E}[Z(\boldsymbol{s})]$ are not important for the computation of the semivariance, but either one is necessary for the covariance. Hence, the semivariance expression filters the mean, and this behavior is especially useful when the mean is unknown. On the other hand, if $\mathrm{E}[Z(\boldsymbol{s})]$ is unknown and the empirical covariance is computed based on the sample mean $\bar{Z}$, this can induce additional bias in the covariance computation.

Example 131.5: A Box Plot of the Square Root Difference Cloud

(View the complete code for this example (vroex5.sas) in the example repository.)

The Gaussian form selected for the semivariogram in the section "Getting Started: VARIOGRAM Procedure" on page 10962 is based on consideration of the plots of the sample semivariogram. For the coal thickness data, the Gaussian form appears to be a reasonable choice.

However, it can often happen that a plot of the sample variogram shows so much scatter that no particular form is evident. The cause of this scatter can be one or more outliers in the pairwise differences of the measured quantities.

A method of identifying potential outliers is discussed in Cressie (1993, section 2.2.2). This example illustrates how to use the OUTPAIR= data set from PROC VARIOGRAM to produce a square root difference cloud, which is useful in detecting outliers.

For the SRF $Z(\boldsymbol{s}), \boldsymbol{s} \in \mathcal{R}^2$, the square root difference cloud for a particular direction $\boldsymbol{e}$ is given by

$$| Z(s_i + he) - Z(s_i) |^{\frac{1}{2}}$$

for a given lag distance h. In the actual computation, all pairs $P_1 P_2$ of points P_1, P_2 within a distance tolerance around h and an angle tolerance around the direction $\boldsymbol{e}$ are used. This generates a number of point pairs for each lag class h. The spread of these values gives an indication of outliers.

Following the example in the section "Getting Started: VARIOGRAM Procedure" on page 10962, this example uses a basic LAGDISTANCE=7, with a distance tolerance of 3.5, and a direction of N–S, with an angle tolerance ATOL=30°.

First, use PROC VARIOGRAM to produce an OUTPAIR= data set. Then use a DATA step to subset this data by choosing pairs within 30° of N–S. In addition, compute lag class and square root difference variables, as the following statements show:

```
title 'Square Root Difference Cloud Example';

proc variogram data=sashelp.thick outp=outp noprint;
   compute novariogram;
   coordinates xc=East yc=North;
   var Thick;
run;
```

```
data sqroot;
   set outp;
   /*- Include only points +/- 30 degrees of N-S -------*/
   where abs(cos) < 0.5;
   /*- Unit lag of 7, distance tolerance of 3.5 --------*/
   lag_class=int(distance/7 + 0.5000001);
   sqr_diff=sqrt(abs(v1-v2));
run;

proc sort data=sqroot;
   by lag_class;
run;
```

Next, summarize the results by using the MEANS procedure:

```
proc means data=sqroot noprint n mean std;
   var sqr_diff;
   by lag_class;
   output out=msqrt n=n mean=mean std=std;
run;
title2 'Summary of Results';

proc print data=msqrt;
   id lag_class;
   var n mean std;
run;
```

The preceding statements produce Output 131.5.1.

Output 131.5.1 Summary of Results

Square Root Difference Cloud Example
Summary of Results

lag_class	n	mean	std
0	5	0.47300	0.14263
1	31	0.77338	0.41467
2	51	1.17052	0.47800
3	58	1.52287	0.51454
4	65	1.68625	0.58465
5	65	1.66963	0.68582
6	80	1.79693	0.62929
7	88	1.73334	0.73191
8	83	1.75528	0.68767
9	108	1.72901	0.58274
10	80	1.48268	0.48695
11	84	1.19242	0.47037
12	68	0.89765	0.42510
13	38	0.84223	0.44249
14	7	1.05653	0.42548
15	3	1.35076	0.11472

Finally, present the results in a box plot by using the SGPLOT procedure. The box plot facilitates the detection of outliers. The statements are as follows:

```
proc sgplot data=sqroot;
   xaxis label = "Lag Class";
   yaxis label = "Square Root Difference";
   title "Box Plot of the Square Root Difference Cloud";
   vbox sqr_diff / category=lag_class;
run;
```

Output 131.5.2 suggests that outliers, if any, do not appear to be adversely affecting the empirical semivariogram in the N–S direction for the coal seam thickness data. The conclusion from Output 131.5.2 is consistent with our previous semivariogram analysis of the same data set in the section "Getting Started: VARIOGRAM Procedure" on page 10962. The effect of the isolated outliers in lag classes 6 and 10–12 in Output 131.5.2 is demonstrated as the divergence between the classical and robust empirical semivariance estimates in the higher distances in Output 131.7. The difference in these estimates comes from the definition of the robust semivariance estimator $\bar{\gamma}_z(\boldsymbol{h})$ (see the section "Theoretical and Computational Details of the Semivariogram" on page 11013), which imposes a smoothing effect on the outlier influence.

Output 131.5.2 Box Plot of the Square Root Difference Cloud

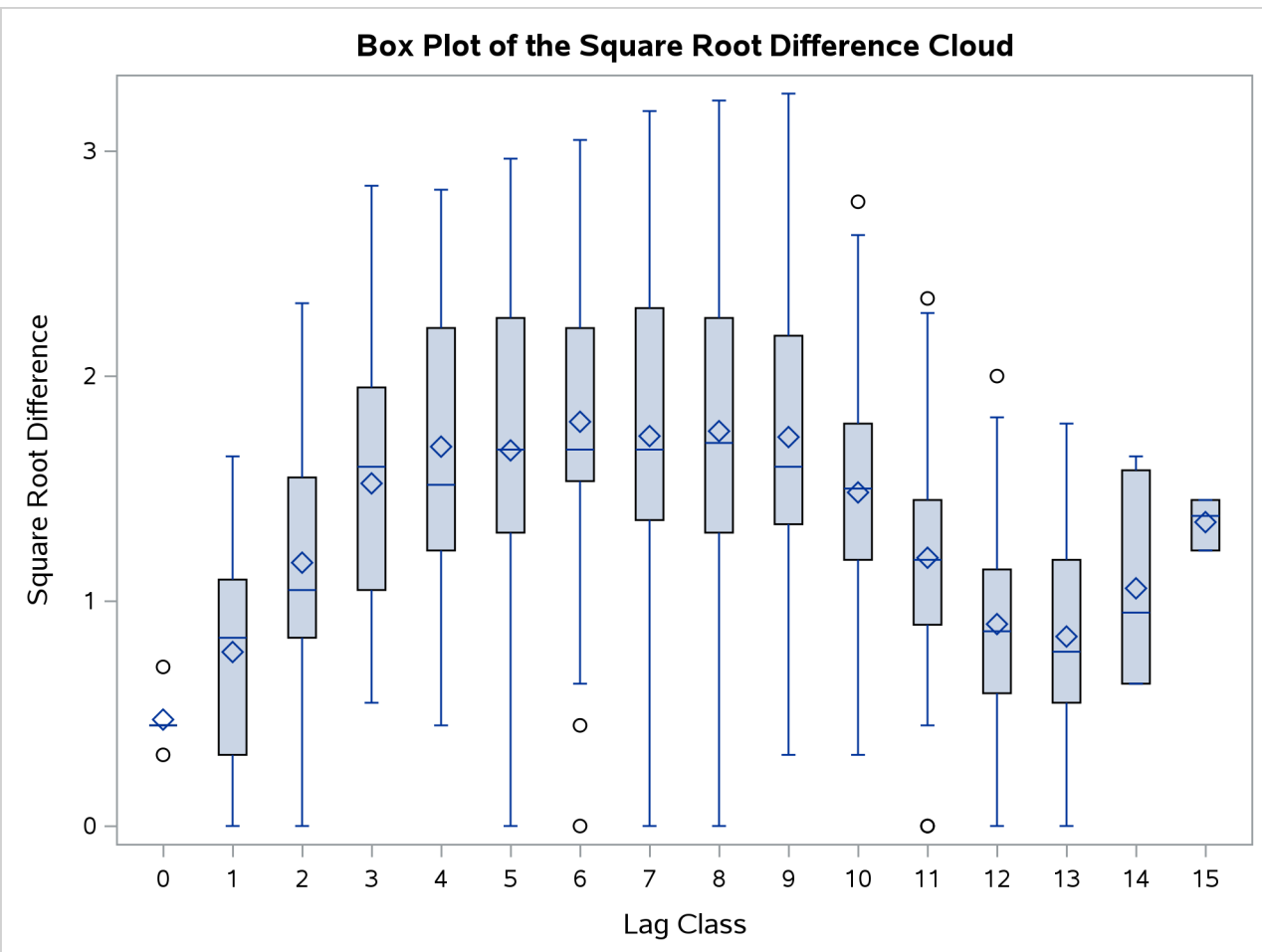

References

Anselin, L. (1996). "The Moran Scatterplot as an ESDA Tool to Assess Local Instability in Spatial Association." In *Spatial Analytical Perspectives on GIS*, edited by M. Fischer, H. Scholten, and D. Unwin, 111–125. London: Taylor & Francis.

Banerjee, S., Carlin, B. P., and Gelfand, A. E. (2004). *Hierarchical Modeling and Analysis for Spatial Data*. Boca Raton, FL: Chapman & Hall/CRC.

Banerjee, S., Carlin, B. P., and Gelfand, A. E. (2014). *Hierarchical Modeling and Analysis for Spatial Data*. 2nd ed. Boca Raton, FL: CRC Press.

Chilès, J.-P., and Delfiner, P. (1999). *Geostatistics: Modeling Spatial Uncertainty*. New York: John Wiley & Sons.

Christakos, G. (1992). *Random Field Models in Earth Sciences*. New York: Academic Press.

Cliff, A. D., and Ord, J. K. (1981). *Spatial Processes: Models and Applications*. London: Pion Ltd.

Cressie, N. (1985). "Fitting Variogram Models by Weighted Least Squares." *Mathematical Geology* 17:563–570.

Cressie, N. (1993). *Statistics for Spatial Data*. Rev. ed. New York: John Wiley & Sons.

Cressie, N., and Hawkins, D. M. (1980). "Robust Estimation of the Variogram: I." *Mathematical Geology* 12:115–125.

Deutsch, C. V., and Journel, A. G. (1992). *GSLIB: Geostatistical Software Library and User's Guide*. New York: Oxford University Press.

Geary, R. C. (1954). "The Contiguity Ratio and Statistical Mapping." *Incorporated Statistician* 5:115–145.

Goovaerts, P. (1997). *Geostatistics for Natural Resources Evaluation*. New York: Oxford University Press.

Hohn, M. E. (1988). *Geostatistics and Petroleum Geology*. New York: Van Nostrand Reinhold.

Jian, X., Olea, R. A., and Yu, Y.-S. (1996). "Semivariogram Modeling by Weighted Least Squares." *Computers and Geosciences* 22:387–397.

Journel, A. G., and Huijbregts, C. J. (1978). *Mining Geostatistics*. New York: Academic Press.

Matheron, G. (1963). "Principles of Geostatistics." *Economic Geology* 58:1246–1266.

Moran, P. A. P. (1950). "Notes on Continuous Stochastic Phenomena." *Biometrika* 37:17–23.

Olea, R. A. (1999). *Geostatistics for Engineers and Earth Scientists*. Boston: Kluwer Academic.

Olea, R. A. (2006). "A Six-Step Practical Approach to Semivariogram Modeling." *Stochastic Environmental Research and Risk Assessment* 20:307–318.

Schabenberger, O., and Gotway, C. A. (2005). *Statistical Methods for Spatial Data Analysis*. Boca Raton, FL: Chapman & Hall/CRC.

Stein, M. L. (1988). "Asymptotically Efficient Prediction of a Random Field with a Misspecified Covariance Function." *Annals of Statistics* 16:55–63.

Appendix A
Special SAS Data Sets

Contents

Introduction to Special SAS Data Sets **11090**
Special SAS Data Sets **11093**
TYPE=ACE Data Sets 11093
TYPE=BOXPLOT Data Sets 11093
TYPE=CALISFIT Data Sets 11093
TYPE=CALISMDL Data Sets 11094
TYPE=CHARTSUM Data Sets 11094
TYPE=CORR Data Sets 11094
TYPE=COV Data Sets 11098
TYPE=CSSCP Data Sets 11098
TYPE=DISTANCE Data Sets 11098
TYPE=EST Data Sets 11099
TYPE=FACTOR Data Sets 11100
TYPE=LINEAR Data Sets 11100
TYPE=LOGISMOD Data Sets 11101
TYPE=MIXED Data Sets 11101
TYPE=QUAD Data Sets 11101
TYPE=SSCP Data Sets 11101
TYPE=TREE Data Sets 11102
TYPE=UCORR Data Sets 11103
TYPE=UCOV Data Sets 11103
TYPE=WEIGHT Data Sets 11103
Definitional Formulas **11103**

Introduction to Special SAS Data Sets

All SAS/STAT procedures create SAS data sets. Any table generated by a procedure can be saved to a data set by using the Output Delivery System (ODS), and many procedures also have syntax that enables you to save other statistics to data sets. Some of these data sets are organized according to certain conventions so that they can be read by a SAS/STAT procedure for further analysis. Such specially organized data sets are recognized by the TYPE= data set attribute.

The CORR procedure (see the *Base SAS Procedures Guide: Statistical Procedures*), for example, can create a data set with the attribute TYPE=CORR containing a correlation matrix. This TYPE=CORR data set can be read by the REG or FACTOR procedure, among others. If the original data set is large, using a special SAS data set in this way can save computer time by avoiding the recomputation of the correlation matrix in subsequent analyses.

PROC REG, for example, can create a TYPE=EST data set containing estimated regression coefficients. If you need to make predictions for new observations, you can use the SCORE procedure to read both the TYPE=EST data set and a data set containing the new observations. PROC SCORE can then compute predicted values or residuals without repeating the entire regression analysis. See Chapter 109, "The SCORE Procedure," for an example.

A special SAS data set might contain different kinds of statistics. A special variable called _TYPE_ is used to distinguish the various statistics. For example, in a TYPE=CORR data set, an observation in which _TYPE_='MEAN' contains the means of the variables in the analysis, and an observation in which _TYPE_='STD' contains the standard deviations. Correlations appear in observations with _TYPE_='CORR'. Another special variable, _NAME_, is needed to identify the row of the correlation matrix. Thus, the correlation between variables X and Y is given by the value of the variable X in the observation for which _TYPE_='CORR' and _NAME_='Y', or by the value of the variable Y in the observation for which _TYPE_='CORR' and _NAME_='X'.

The special data sets created by SAS/STAT procedures can generally be used directly by other procedures without modification. However, if you create an output data set with PROC CORR and use the NOCORR option to omit the correlation matrix from the OUT= data set, you need to set the TYPE= option either in parentheses following the OUT= data set name in the PROC CORR statement or in parentheses following the DATA= option in any other procedure that recognizes the special TYPE= attribute. In either case, the TYPE= option should be set to COV, CSSCP, or SSCP according to what type of matrix is stored in the data set and what data set types are accepted as input by the other procedures you plan to use. If you do not follow these steps and you use the TYPE=CORR data set with no correlation matrix as input to another procedure, the procedure might issue an error message indicating that the correlation matrix is missing from the data set.

You can create special SAS data sets directly in a DATA step by specifying the TYPE= option in parentheses after the data set name in the DATA statement. See "Example A.2: Creating a TYPE=CORR Data Set in a DATA Step" on page 11097 for an example. If you use a DATA step with a SET statement to modify a special SAS data set, you must specify the TYPE= option in the DATA statement. The TYPE= attribute of the data set in the SET statement is *not* automatically copied to the data set being created. You can determine the TYPE= attribute of a data set by using the CONTENTS procedure (see "Example A.1: A TYPE=CORR Data Set Produced by PROC CORR" on page 11095 and the *Base SAS Procedures Guide* for details).

Table A.1 summarizes the TYPE= data sets that can be used as input to SAS/STAT procedures. Table A.2 summarizes the TYPE= data sets that are created by SAS/STAT procedures and the statements each procedure uses to create its special output data sets. Most procedures accept ordinary SAS data sets and create ordinary

output SAS data sets with no TYPE= specification in addition to the special data sets shown in the tables. When you specify a data set with a type that the procedure does not recognize, the procedure prints an error message and stops executing.

Table A.1 SAS/STAT Procedures That Accept Special Input Data Sets Types

Procedure	Special TYPE= Data Sets Accepted
ACECLUS	ACE, CORR, COV, SSCP, UCORR, UCOV
BOXPLOT	BOXPLOT, CHARTSUM
CALIS	CALISMDL, CORR, COV, FACTOR, SSCP, UCORR, UCOV, WEIGHT
CANCORR	CORR, COV, FACTOR, SSCP, UCORR, UCOV
CANDISC	CORR, COV, SSCP, CSSCP
CATMOD	EST
CLUSTER	DISTANCE
DISCRIM	CORR, COV, SSCP, CSSCP, LINEAR, QUAD, MIXED
FACTOR	ACE, CORR, COV, FACTOR, SSCP, UCORR, UCOV
LIFEREG	EST
LOGISTIC	EST LOGISMOD
MI	EST, COV, CORR
MIANALYZE	EST, COV, CORR
MODECLUS	DISTANCE
PHREG	EST
PRINCOMP	ACE, CORR, COV, EST, FACTOR, SSCP, UCORR, UCOV
PROBIT	EST
QUANTREG	EST
REG	CORR, COV, SSCP, UCORR, UCOV
ROBUSTREG	EST
SCORE	SCORE= data set can be of any type
SIMNORM	CORR, COV
SURVEYLOGISTIC	EST
STEPDISC	CORR, COV, SSCP, CSSCP
TREE	TREE
VARCLUS	CORR, COV, FACTOR, SSCP, UCORR, UCOV

Table A.2 SAS/STAT Procedures That Create Special Output Data Set Types

Procedure	TYPE=	Statement and Option Required
ACECLUS	ACE	PROC ACECLUS OUTSTAT=
BOXPLOT	BOXPLOT	PLOT / OUTBOX=
	CHARTSUM	PLOT / OUTHISTORY=

Table A.2 *continued*

Procedure	TYPE=	Statement and Option Required
CALIS	CALISFIT	PROC CALIS OUTFIT=
	CALISMDL	PROC CALIS OUTMODEL=
	CORR	PROC CALIS CORR OUTSTAT=
	COV	PROC CALIS OUTSTAT=
	EST	PROC CALIS OUTEST=
	WEIGHT	PROC CALIS OUTWGT=
CANCORR	CORR	PROC CANCORR OUTSTAT=
	UCORR	PROC CANCORR NOINT OUTSTAT=
CANDISC	CORR	PROC CANDISC OUTSTAT=
CATMOD	EST	RESPONSE / OUTEST=
CLUSTER	TREE	PROC CLUSTER OUTTREE=
DISCRIM	LINEAR	PROC DISCRIM POOL=YES OUTSTAT=
	QUAD	PROC DISCRIM POOL=NO OUTSTAT=
	MIXED	PROC DISCRIM POOL=TEST OUTSTAT=
	CORR	PROC DISCRIM METHOD=NPAR OUTSTAT=
DISTANCE	DISTANCE	PROC DISTANCE METHOD=*distance-method* OUT=
	SIMILAR	PROC DISTANCE METHOD=*similarity-method* OUT=
FACTOR	FACTOR	PROC FACTOR OUTSTAT=
LIFEREG	EST	PROC LIFEREG OUTEST=
LOGISTIC	EST	PROC LOGISTIC OUTEST=
	LOGISMOD	PROC LOGISTIC OUTMODEL=
MI	COV	EM OUTEM=
	COV	EM OUTITER=
	COV	MCMC OUTITER=
	EST	MCMC OUTEST=
NLIN	EST	PROC NLIN OUTEST=
ORTHOREG	EST	PROC ORTHOREG OUTEST=
PHREG	EST	PROC PHREG OUTEST=
PRINCOMP	CORR	PROC PRINCOMP OUTSTAT=
	COV	PROC PRINCOMP COV OUTSTAT=
	UCORR	PROC PRINCOMP NOINT OUTSTAT=
	UCOV	PROC PRINCOMP NOINT COV OUTSTAT=
PROBIT	EST	PROC PROBIT OUTEST=
QUANTREG	EST	PROC QUANTREG OUTEST=
REG	EST	PROC REG OUTEST=
	SSCP	PROC REG OUTSSCP=
ROBUSTREG	EST	PROC ROBUSTREG OUTEST=

Table A.2 *continued*

Procedure	TYPE=	Statement and Option Required
VARCLUS	CORR	PROC VARCLUS OUTSTAT=
	UCORR	PROC VARCLUS NOINT OUTSTAT=
	TREE	PROC VARCLUS OUTTREE=

Special SAS Data Sets

TYPE=ACE Data Sets

A TYPE=ACE data set is created by the ACECLUS procedure, and it contains the approximate within-cluster covariance estimate, as well as eigenvalues and eigenvectors from a canonical analysis, among other statistics. It can be used as input to the ACECLUS procedure to initialize another execution of PROC ACECLUS. It can also be used to compute canonical variable scores with PROC SCORE and as input to PROC FACTOR, specifying METHOD=SCORE, to rotate the canonical variables. See Chapter 27, "The ACECLUS Procedure," for details.

TYPE=BOXPLOT Data Sets

A TYPE=BOXPLOT data set is created by and used by the BOXPLOT procedure. The data set contains the group summary statistics and outlier values required for constructing a schematic box plot. Each observation in a TYPE=BOXPLOT data set records the value of a single feature of one group's box-and-whiskers plot, such as its mean. Consequently, a TYPE=BOXPLOT data set contains multiple observations per group. These must appear consecutively in the data set. The _TYPE_ variable identifies the feature whose value is recorded in a given observation. _TYPE_ values of 'N', 'MIN', 'Q1', 'MEDIAN', 'MEAN', 'Q3', and 'MAX 'are required for each group. See Chapter 32, "The BOXPLOT Procedure," for details.

TYPE=CALISFIT Data Sets

PROC CALIS creates a TYPE=CALISFIT data set. This data set contains the names of the model fit indices and their values. A TYPE=CALISFIT data set is intended to save all the fit index values for future use, especially when the customized fit summary table shows only a small number of fit indices. See Chapter 33, "The CALIS Procedure," for details.

TYPE=CALISMDL Data Sets

PROC CALIS creates and accepts as input a TYPE=CALISMDL data set. This data set contains the model specification and the computed parameter estimates. A TYPE=CALISMDL data set is intended to be reused as an input data set to specify good initial values in subsequent analyses by PROC CALIS. See Chapter 33, "The CALIS Procedure," for details.

TYPE=CHARTSUM Data Sets

A TYPE=CHARTSUM data set is created by and used by the BOXPLOT procedure. The data set contains group summary statistics associated with box-and-whiskers plots. See Chapter 32, "The BOXPLOT Procedure," for details.

TYPE=CORR Data Sets

A TYPE=CORR data set usually contains a correlation matrix and possibly other statistics including means, standard deviations, and the number of observations in the original SAS data set from which the correlation matrix was computed. Using PROC CORR with an output data set option (OUTP=, OUTS=, OUTK=, OUTH=, or OUT=) produces a TYPE=CORR data set. (For a complete description of the CORR procedure, see the *Base SAS Procedures Guide: Statistical Procedures*.) The CALIS, CANCORR, CANDISC, DISCRIM, PRINCOMP, and VARCLUS procedures can also create a TYPE=CORR data set with additional statistics (the CORR option is needed in PROC CALIS). A TYPE=CORR data set containing a correlation matrix can be used as input for the ACECLUS, CALIS, CANCORR, CANDISC, DISCRIM, FACTOR, PRINCOMP, REG, SCORE, STEPDISC, and VARCLUS procedures. The variables in a TYPE=CORR data set are as follows:

- the BY variable or variables, if a BY statement is used with the procedure
- _TYPE_, a character variable of length eight with values identifying the type of statistic in each observation, such as 'MEAN', 'STD', 'N', and 'CORR'
- _NAME_, a character variable with values identifying the variable with which a given row of the correlation matrix is associated
- other variables that were analyzed by the CORR procedure or other procedures

The usual values of the _TYPE_ variable are as follows:

TYPE	Contents
MEAN	mean of each variable analyzed
STD	standard deviation of each variable
N	number of observations used in the analysis. PROC CORR records the number of nonmissing values for each variable unless the NOMISS option is used. If the NOMISS option is specified, or if the CALIS, CANCORR, CANDISC, PRINCOMP, or VARCLUS procedure is used to create the data set, observations with one or more missing values are omitted from the analysis, so this value is the same for each variable and provides the number of observations with no missing values. If a FREQ statement is used with the procedure that creates the data set, the number of observations is the sum of the relevant values of the variable in the FREQ statement. Procedures that read a TYPE=CORR data set use the smallest value in the observation with _TYPE_='N' as the number of observations in the analysis.
SUMWGT	sum of the observation weights if a WEIGHT statement is used with the procedure that creates the data set. The values are determined analogously to those of the _TYPE_='N' observation.
CORR	correlations with the variable named by the _NAME_ variable

There might be additional observations in a TYPE=CORR data set depending on the particular procedure and options used.

If you create a TYPE=CORR data set yourself, the data set need not contain the observations with _TYPE_='MEAN', 'STD', 'N', or 'SUMWGT', unless you intend to use one of the discriminant procedures. Procedures assume that all of the means are 0.0 and that the standard deviations are 1.0 if this information is not in the TYPE=CORR data set. If _TYPE_='N' does not appear, most procedures assume that the number of observations is 10,000; significance tests and other statistics that depend on the number of observations are, of course, meaningless. In the CALIS and CANCORR procedures, you can use the EDF= option instead of including a _TYPE_='N' observation.

A correlation matrix is symmetric; that is, the correlation between X and Y is the same as the correlation between Y and X. The CALIS, CANCORR, CANDISC, CORR, DISCRIM, PRINCOMP, and VARCLUS procedures output the entire correlation matrix. If you create the data set yourself, you need to include only one of the two occurrences of the correlation between two variables; the other can be given a missing value.

If you create a TYPE=CORR data set yourself, the _TYPE_ and _NAME_ variables are not necessary except for use with the discriminant procedures and PROC SCORE. If there is no _TYPE_ variable, then all observations are assumed to contain correlations. If there is no _NAME_ variable, the first observation is assumed to correspond to the first variable in the analysis, the second observation to the second variable, and so on. However, if you omit the _NAME_ variable, you will not be able to analyze arbitrary subsets of the variables or list the variables in a VAR or MODEL statement in a different order.

Example A.1: A TYPE=CORR Data Set Produced by PROC CORR

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/appssds.sas.)

See Figure A.1 for an example of a TYPE=CORR data set produced by the following SAS statements. Figure A.2 displays partial output from PROC CONTENTS, which indicates that the "Data Set Type" is 'CORR'.

```
title 'Five Socioeconomic Variables';
title2 'Harman (1976), Modern Factor Analysis, Third Edition';

data SocEcon;
   input Pop School Employ Services House;
   datalines;
5700     12.8      2500      270       25000
1000     10.9      600       10        10000
3400     8.8       1000      10        9000
3800     13.6      1700      140       25000
4000     12.8      1600      140       25000
8200     8.3       2600      60        12000
1200     11.4      400       10        16000
9100     11.5      3300      60        14000
9900     12.5      3400      180       18000
9600     13.7      3600      390       25000
9600     9.6       3300      80        12000
9400     11.4      4000      100       13000
;

proc corr noprint out=corrcorr;
run;

proc print data=corrcorr;
run;

proc contents data=corrcorr;
run;
```

Figure A.1 A TYPE=CORR Data Set Produced by PROC CORR

Five Socioeconomic Variables
Harman (1976), Modern Factor Analysis, Third Edition

Obs	_TYPE_	_NAME_	Pop	School	Employ	Services	House
1	MEAN		6241.67	11.4417	2333.33	120.833	17000.00
2	STD		3439.99	1.7865	1241.21	114.928	6367.53
3	N		12.00	12.0000	12.00	12.000	12.00
4	CORR	Pop	1.00	0.0098	0.97	0.439	0.02
5	CORR	School	0.01	1.0000	0.15	0.691	0.86
6	CORR	Employ	0.97	0.1543	1.00	0.515	0.12
7	CORR	Services	0.44	0.6914	0.51	1.000	0.78
8	CORR	House	0.02	0.8631	0.12	0.778	1.00

Figure A.2 Contents of a TYPE=CORR Data Set

Five Socioeconomic Variables
Harman (1976), Modern Factor Analysis, Third Edition

The CONTENTS Procedure

Data Set Name	WORK.CORRCORR	**Observations**	8
Member Type	DATA	**Variables**	7
Engine	SASE7	**Indexes**	0
Created	DDMMMYY:00:00:00	**Observation Length**	56
Last Modified	DDMMMYY:00:00:00	**Deleted Observations**	0
Protection		**Compressed**	NO
Data Set Type	CORR	**Sorted**	NO
Label	Pearson Correlation Matrix		
Data Representation	Native		
Encoding	Session		

Example A.2: Creating a TYPE=CORR Data Set in a DATA Step

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/appssds.sas.)

This example creates a TYPE=CORR data set by reading a correlation matrix in a DATA step. Figure A.3 shows the resulting data set.

```
title 'Five Socioeconomic Variables';

data datacorr(type=corr);
   infile cards missover;
   _type_='corr';
   input _Name_ $ Pop School Employ Services House;
   datalines;
Pop        1.00000
School     0.00975   1.00000
Employ     0.97245   0.15428   1.00000
Services   0.43887   0.69141   0.51472   1.00000
House      0.02241   0.86307   0.12193   0.77765   1.00000
;

proc print data=datacorr;
run;
```

Figure A.3 A TYPE=CORR Data Set Created by a DATA Step

Five Socioeconomic Variables

OBS	_type_	_Name_	Pop	School	Employ	Services	House
1	corr	Pop	1.00000	.	.	.	.
2	corr	School	0.00975	1.00000	.	.	.
3	corr	Employ	0.97245	0.15428	1.00000	.	.
4	corr	Services	0.43887	0.69141	0.51472	1.00000	.
5	corr	House	0.02241	0.86307	0.12193	0.77765	1

TYPE=COV Data Sets

A TYPE=COV data set is similar to a TYPE=CORR data set except that it has _TYPE_='COV' observations containing covariances instead of or in addition to _TYPE_='CORR' observations containing correlations. The CALIS and PRINCOMP procedures create a TYPE=COV data set (the COV option is needed in PROC PRINCOMP). You can also create a TYPE=COV data set by using PROC CORR with the COV and NOCORR options and specifying the data set option TYPE=COV in parentheses following the name of the output data set. You can use only the OUTP= or OUT= option to create a TYPE=COV data set with PROC CORR. Another way to create a TYPE=COV data set is to read a covariance matrix in a data set, in the same manner as shown in "Example A.2: Creating a TYPE=CORR Data Set in a DATA Step" on page 11097 for a TYPE=CORR data set. TYPE=COV data sets are used by the same procedures that use TYPE=CORR data sets.

TYPE=CSSCP Data Sets

A TYPE=CSSCP data set contains a corrected sum of squares and crossproducts (CSSCP) matrix. TYPE=CSSCP data sets are created by using the CORR procedure with the CSSCP option and specifying the data set option TYPE=CSSCP in parentheses following the name of the OUTP= or OUT= data set. You can also create TYPE=CSSCP data sets in a DATA step; in this case, TYPE=CSSCP must be specified as a data set option. The variables in a TYPE=CSSCP data set are the same as those found in a TYPE=SSCP data set, except that there is not a variable called Intercept or a row with _NAME_='Intercept'. TYPE=CSSCP data sets are read by only the CANDISC, DISCRIM, and STEPDISC procedures. Formulas useful for illustrating differences between corrected and uncorrected matrices in some special SAS data sets are shown in the section "Definitional Formulas" on page 11103.

TYPE=DISTANCE Data Sets

PROC DISTANCE creates a TYPE=DISTANCE or TYPE=SIMILAR data set, depending on the METHOD= option. TYPE=DISTANCE can be used as an input data set to PROC MODECLUS or PROC CLUSTER, but TYPE=SIMILAR cannot be used as an input to any procedures. The proximity measures are stored as a lower triangular matrix or a square matrix in the OUT= data set (depending on the SHAPE= option). See Chapter 43, "The DISTANCE Procedure," for details. You can also create a TYPE=DISTANCE data set in a DATA step by reading or computing a lower triangular or symmetric matrix of dissimilarity values, such as a chart of mileage between cities. The number of observations must be equal to the number of variables used in the analysis. This type of data set is used as input by the CLUSTER and MODECLUS procedures. PROC CLUSTER ignores the upper triangular portion of a TYPE=DISTANCE data set and assumes that all main diagonal values are zero, even if they are missing. PROC MODECLUS uses the entire distance matrix and does not require the matrix to be symmetric. See Chapter 40, "The CLUSTER Procedure," and Chapter 85, "The MODECLUS Procedure," for examples and details.

TYPE=EST Data Sets

A TYPE=EST data set contains parameter estimates. The CALIS, CATMOD, LIFEREG, LOGISTIC, NLIN, ORTHOREG, PHREG, PROBIT, and REG procedures create TYPE=EST data sets when the OUTEST= option is specified. A TYPE=EST data set produced by PROC LIFEREG, PROC ORTHOREG, or PROC REG can be used with PROC SCORE to compute residuals or predicted values. The variables in a TYPE=EST data set include the following:

- the BY variables, if a BY statement is used
- _TYPE_, a character variable of length eight, that indicates the type of estimate. The values depend on which procedure created the data set. Usually a value of 'PARM' or 'PARMS' indicates estimated regression coefficients, and a value of 'COV' or 'COVB' indicates estimated covariances of the parameter estimates. Some procedures, such as PROC NLIN, have other values of _TYPE_ for special purposes.
- _NAME_, a character variable that contains the values of the names of the rows of the covariance matrix when the procedure outputs the covariance matrix of the parameter estimates
- variables that contain the parameter estimates, usually the same variables that appear in the VAR statement or in any MODEL statement. See Chapter 33, "The CALIS Procedure," Chapter 36, "The CATMOD Procedure," and Chapter 88, "The NLIN Procedure," for details on the variable names used in output data sets created by those procedures.

Other variables can be included depending on the particular procedure and options used.

Example A.3: A TYPE=EST Data Set Produced by PROC REG

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/appssds.sas.)

Figure A.4 shows the TYPE=EST data set produced by the following statements:

```
proc reg data=SocEcon outest=regest covout;
   full:   model house=pop school employ services / noprint;
   empser: model house=employ services / noprint;
quit;

proc print data=regest;
run;
```

Figure A.4 A TYPE=EST Data Set Produced by PROC REG

Five Socioeconomic Variables

OBS	_MODEL_	_TYPE_	_NAME_	_DEPVAR_	_RMSE_	Intercept	Pop	School	Employ	Services	House
1	full	PARMS		House	3122.03	-8074.21	0.65	2140.10	-2.92	27.81	-1
2	full	COV	Intercept	House	3122.03	109408014.44	-9157.04	-9784744.54	20612.49	102764.89	.
3	full	COV	Pop	House	3122.03	-9157.04	2.32	852.86	-6.20	-5.20	.
4	full	COV	School	House	3122.03	-9784744.54	852.86	907886.36	-2042.24	-9608.59	.
5	full	COV	Employ	House	3122.03	20612.49	-6.20	-2042.24	17.44	6.50	.
6	full	COV	Services	House	3122.03	102764.89	-5.20	-9608.59	6.50	202.56	.
7	empser	PARMS		House	3789.96	15021.71	.	.	-1.94	53.88	-1
8	empser	COV	Intercept	House	3789.96	5824096.19	.	.	-1915.99	-1294.94	.
9	empser	COV	Employ	House	3789.96	-1915.99	.	.	1.15	-6.41	.
10	empser	COV	Services	House	3789.96	-1294.94	.	.	-6.41	134.49	.

TYPE=FACTOR Data Sets

A TYPE=FACTOR data set is created by PROC FACTOR when the OUTSTAT= option is specified. The CALIS, CANCORR, FACTOR, PRINCOMP, SCORE, and VARCLUS procedures can use TYPE=FACTOR data sets as input. The variables are the same as in a TYPE=CORR data set. The statistics include means, standard deviations, sample size, correlations, eigenvalues, eigenvectors, factor patterns, residual correlations, scoring coefficients, and others depending on the options specified. See Chapter 44, "The FACTOR Procedure," for details. When the NOINT option is used with the OUTSTAT= option in PROC FACTOR, the value of the _TYPE_ variable is set to 'USCORE' instead of 'SCORE' to indicate that the scoring coefficients have not been corrected for the mean. If this data set is used with PROC SCORE, the value of the _TYPE_ variable tells PROC SCORE whether or not to subtract the mean from the scoring coefficients.

TYPE=LINEAR Data Sets

A TYPE=LINEAR data set contains the coefficients of a linear function of the variables in observations with _TYPE_='LINEAR'. PROC DISCRIM stores linear discriminant function coefficients in a TYPE=LINEAR data set when you specify METHOD=NORMAL (the default method), POOL=YES, and an OUTSTAT= data set; the data set can be used in a subsequent invocation of PROC DISCRIM to classify additional observations. Many other statistics can be included depending on the options used. See Chapter 42, "The DISCRIM Procedure," for details.

TYPE=LOGISMOD Data Sets

A TYPE=LOGISMOD data set contains information about a logistic regression model fit by PROC LOGISTIC. PROC LOGISTIC both creates and reads TYPE=LOGISMOD data sets. See Chapter 79, "The LOGISTIC Procedure," for details.

TYPE=MIXED Data Sets

A TYPE=MIXED data set contains coefficients of either a linear or a quadratic function, or both if there are BY groups. PROC DISCRIM produces a TYPE=MIXED data set when you specify METHOD=NORMAL (the default method), POOL=TEST, and an OUTSTAT= data set. See Chapter 42, "The DISCRIM Procedure," for details.

TYPE=QUAD Data Sets

A TYPE=QUAD data set contains the coefficients of a quadratic function of the variables in observations with _TYPE_='QUAD'. PROC DISCRIM stores quadratic discriminant function coefficients in a TYPE=QUAD data set when you specify METHOD=NORMAL (the default method), POOL=NO, and an OUTSTAT= data set; the data set can be used in a subsequent invocation of PROC DISCRIM to classify additional observations. Many other statistics can be included depending on the options used. See Chapter 42, "The DISCRIM Procedure," for details.

TYPE=SSCP Data Sets

A TYPE=SSCP data set contains an uncorrected sum of squares and crossproducts (SSCP) matrix. TYPE=SSCP data sets are produced by PROC REG when the OUTSSCP= option is specified in the PROC REG statement. You can also create a TYPE=SSCP data set by using PROC CORR with the SSCP option and specifying the data set option TYPE=SSCP in parentheses following the name of the OUTP= or OUT= data set. You can also create TYPE=SSCP data sets in a DATA step; in this case, TYPE=SSCP must be specified as a data set option.

The variables in a TYPE=SSCP data set include those found in a TYPE=CORR data set. In addition, there is a variable called Intercept that contains crossproducts for the intercept (sums of the variables). The SSCP matrix is stored in observations with _TYPE_='SSCP', including a row with _NAME_='Intercept'. PROC REG also outputs an observation with _TYPE_='N'. PROC CORR includes observations with _TYPE_='MEAN' and _TYPE_='STD' as well. TYPE=SSCP data sets are used by the same procedures that use TYPE=CORR data sets.

Example A.4: A TYPE=SSCP Data Set Produced by PROC REG

(View the complete code for this example at https://github.com/sascommunities/doc-supplement-statug/tree/main/Examples/a-c/appssds.sas.)

The following statements create a TYPE=SSCP data set from the SocEcon input data set created in "Example A.1: A TYPE=CORR Data Set Produced by PROC CORR" on page 11095:

```
proc reg data=SocEcon outsscp=regsscp;
   model house=pop school employ services / noprint;
quit;

proc print data=regsscp;
run;
```

The data set is created by PROC REG and is displayed in Figure A.5.

Figure A.5 A TYPE=SSCP Data Set Produced by PROC REG

Five Socioeconomic Variables

OBS	_TYPE_	_NAME_	Intercept	Pop	School	Employ	Services	House
1	SSCP	Intercept	12.0	74900	137.30	28000	1450	204000
2	SSCP	Pop	74900.0	597670000	857640.00	220440000	10959000	1278700000
3	SSCP	School	137.3	857640	1606.05	324130	18152	2442100
4	SSCP	Employ	28000.0	220440000	324130.00	82280000	4191000	486600000
5	SSCP	Services	1450.0	10959000	18152.00	4191000	320500	30910000
6	SSCP	House	204000.0	1278700000	2442100.00	486600000	30910000	3914000000
7	N		12.0	12	12.00	12	12	12

TYPE=TREE Data Sets

Some clustering procedures produce TYPE=TREE data sets. For example, in PROC CLUSTER, a TYPE=TREE data set contains one observation for each observation in the input data set, plus one observation for each cluster of two or more observations (that is, one observation for each node of the cluster tree). The total number of output observations is usually $2n - 1$, where n is the number of input observations. The density methods might produce fewer output observations when the number of clusters cannot be reduced to one.

In PROC VARCLUS, the OUTTREE= data set contains one observation for each variable clustered plus one observation for each cluster of two or more variables—that is, one observation for each node of the cluster tree. The total number of output observations is between n and $2n - 1$, where n is the number of variables clustered. See Chapter 40, "The CLUSTER Procedure," and Chapter 129, "The VARCLUS Procedure," for details.

TYPE=UCORR Data Sets

A TYPE=UCORR data set is almost identical to a TYPE=CORR data set, except that the correlations are uncorrected for the mean. The corresponding value of the _TYPE_ variable is 'UCORR' instead of 'CORR'. Uncorrected standard deviations are in observations with _TYPE_='USTD'. A TYPE=UCORR data set can be used as input for every SAS/STAT procedure that uses a TYPE=CORR data set, except for the CANDISC, DISCRIM, and STEPDISC procedures. TYPE=UCORR data sets can be created by the CANCORR, PRINCOMP, and VARCLUS procedures.

TYPE=UCOV Data Sets

A TYPE=UCOV data set is similar to a TYPE=COV data set, except that the covariances are uncorrected for the mean. Also, the corresponding value of the _TYPE_ variable is 'UCOV' instead of 'COV'. A TYPE=UCOV data set can be used as input for every SAS/STAT procedure that uses a TYPE=COV data set, except for the CANDISC, DISCRIM, and STEPDISC procedures. TYPE=UCOV data sets can be created by the PRINCOMP procedure.

TYPE=WEIGHT Data Sets

The CALIS procedure creates and accepts as input a TYPE=WEIGHT data set. This data set contains the weight matrix used in generalized, weighted, or diagonally weighted least squares estimation. See Chapter 33, "The CALIS Procedure," for details.

Definitional Formulas

This section contrasts corrected and uncorrected SSCP, COV, and CORR matrices by showing how these matrices can be computed. In the following formulas, assume that the data consist of two variables, X and Y, with n observations.

$$\text{SSCP} = \begin{bmatrix} n & \sum X & \sum Y \\ \sum X & \sum X^2 & \sum XY \\ \sum Y & \sum XY & \sum Y^2 \end{bmatrix}$$

$$\text{CSSCP} = \begin{bmatrix} \sum(X-\bar{X})^2 & \sum(X-\bar{X})(Y-\bar{Y}) \\ \sum(X-\bar{X})(Y-\bar{Y}) & \sum(Y-\bar{Y})^2 \end{bmatrix}$$

$$\text{COV} = \frac{\text{CSSCP}}{n-1} = \frac{1}{n-1}\begin{bmatrix} \sum(X-\bar{X})^2 & \sum(X-\bar{X})(Y-\bar{Y}) \\ \sum(X-\bar{X})(Y-\bar{Y}) & \sum(Y-\bar{Y})^2 \end{bmatrix}$$

$$\text{UCOV} = \frac{1}{n}\begin{bmatrix} \sum X^2 & \sum XY \\ \sum XY & \sum Y^2 \end{bmatrix}$$

$$\text{CORR} = \begin{bmatrix} 1 & \dfrac{\sum(X-\bar{X})(Y-\bar{Y})}{\sqrt{\sum(X-\bar{X})^2 \sum(Y-\bar{Y})^2}} \\ \dfrac{\sum(X-\bar{X})(Y-\bar{Y})}{\sqrt{\sum(X-\bar{X})^2 \sum(Y-\bar{Y})^2}} & 1 \end{bmatrix}$$

$$\text{UCORR} = \begin{bmatrix} 1 & \dfrac{\sum XY}{\sqrt{\sum X^2 \sum Y^2}} \\ \dfrac{\sum XY}{\sqrt{\sum X^2 \sum Y^2}} & 1 \end{bmatrix}$$

Appendix B
Sashelp Data Sets

Contents

Overview of Sashelp Data Sets 11106
Baseball Data 11107
Tropical Rain Forest Tree Data 11109
2003 Birth Weight Data Set 11111
Body Mass Index for Men 11112
Isopod Burrows Data 11113
Bone Marrow Transplant Data 11115
1997 Birth Weight Data 11116
Class Data 11117
Comet Data 11118
El Niño–Southern Oscillation Data 11119
Finland's Lake Laengelmaevesi Fish Catch Data 11120
Exhaust Emissions Data 11121
Fisher (1936) Iris Data 11122
Junk Email Data 11124
Leukemia Data Sets 11126
Margarine Data 11128
Flying Mileages between 10 US Cities Data 11129
Earthquake Locations in the United States 11131
Hot-Spring Locations in the United States 11132
Coal Seam Thickness Data 11133
1980 US Presidential Election Data 11134
References 11134

Overview of Sashelp Data Sets

SAS provides more than 200 data sets in the Sashelp library. These data sets are available for you to use for examples and for testing code. For example, the following step uses the Sashelp.Class data set:

```
proc reg data=sashelp.Class;
   model weight = height;
quit;
```

You do not need to provide a DATA step to use Sashelp data sets.

The following steps list all the data sets that are available in Sashelp:

```
ods select none;
proc contents data=sashelp._all_;
   ods output members=m;
run;
ods select all;

proc print;
   where memtype = 'DATA';
run;
```

The results of these steps (more than 200 data set names) are not displayed.

The following steps provide detailed information about the Sashelp data sets:

```
proc contents data=sashelp._all_;
run;
```

The results of this step (hundreds of pages of PROC CONTENTS output) are not displayed.

Twenty-two Sashelp data sets are used in SAS/STAT documentation, and the following sections describe these data sets:

Sashelp.Baseball	"Baseball Data" on page 11107
Sashelp.BEI	"Tropical Rain Forest Tree Data" on page 11109
Sashelp.BirthWgt	"2003 Birth Weight Data Set" on page 11111
Sashelp.BMIMen	"Body Mass Index for Men" on page 11112
Sashelp.BMT	"Bone Marrow Transplant Data" on page 11115
Sashelp.Burrows	"Isopod Burrows Data" on page 11113
Sashelp.BWeight	"1997 Birth Weight Data" on page 11116
Sashelp.Class	"Class Data" on page 11117
Sashelp.Comet	"Comet Data" on page 11118
Sashelp.ENSO	"El Niño–Southern Oscillation Data" on page 11119
Sashelp.Fish	"Finland's Lake Laengelmaevesi Fish Catch Data" on page 11120
Sashelp.Gas	"Exhaust Emissions Data" on page 11121
Sashelp.Iris	"Fisher (1936) Iris Data" on page 11122
Sashelp.JunkMail	"Junk Email Data" on page 11124
Sashelp.LeuTest	"Leukemia Data Sets" on page 11126
Sashelp.LeuTrain	"Leukemia Data Sets" on page 11126
Sashelp.Margarin	"Margarine Data" on page 11128

Sashelp.Mileages	"Flying Mileages between 10 US Cities Data" on page 11129
Sashelp.Quakes	"Earthquake Locations in the United States" on page 11131
Sashelp.Springs	"Hot-Spring Locations in the United States" on page 11132
Sashelp.Thick	"Coal Seam Thickness Data" on page 11133
Sashelp.Vote1980	"1980 US Presidential Election Data" on page 11134

Baseball Data

The Sashelp.Baseball data set contains salary and performance information for Major League Baseball players (excluding pitchers) who played at least one game in both the 1986 and 1987 seasons (Time Inc. 1987). The salaries are for the 1987 season, and the performance measures are from the 1986 season. The following steps display information about the Sashelp.Baseball data set and create Figure B.1:

```
title 'Baseball Data';
proc contents data=sashelp.Baseball varnum;
   ods select position;
run;

title 'The First Five Observations Out of 322';
proc print data=sashelp.Baseball(obs=5);
run;
```

Figure B.1 Baseball Data

Baseball Data

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	Name	Char	18	Player's Name
2	Team	Char	14	Team at the End of 1986
3	nAtBat	Num	8	Times at Bat in 1986
4	nHits	Num	8	Hits in 1986
5	nHome	Num	8	Home Runs in 1986
6	nRuns	Num	8	Runs in 1986
7	nRBI	Num	8	RBIs in 1986
8	nBB	Num	8	Walks in 1986
9	YrMajor	Num	8	Years in the Major Leagues
10	CrAtBat	Num	8	Career Times at Bat
11	CrHits	Num	8	Career Hits
12	CrHome	Num	8	Career Home Runs
13	CrRuns	Num	8	Career Runs
14	CrRbi	Num	8	Career RBIs
15	CrBB	Num	8	Career Walks
16	League	Char	8	League at the End of 1986
17	Division	Char	8	Division at the End of 1986
18	Position	Char	8	Position(s) in 1986
19	nOuts	Num	8	Put Outs in 1986
20	nAssts	Num	8	Assists in 1986
21	nError	Num	8	Errors in 1986
22	Salary	Num	8	1987 Salary in $ Thousands
23	Div	Char	16	League and Division
24	logSalary	Num	8	Log Salary

The First Five Observations Out of 322

Obs	Name	Team	nAtBat	nHits	nHome	nRuns	nRBI	nBB	YrMajor	CrAtBat	CrHits	CrHome
1	Allanson, Andy	Cleveland	293	66	1	30	29	14	1	293	66	1
2	Ashby, Alan	Houston	315	81	7	24	38	39	14	3449	835	69
3	Davis, Alan	Seattle	479	130	18	66	72	76	3	1624	457	63
4	Dawson, Andre	Montreal	496	141	20	65	78	37	11	5628	1575	225
5	Galarraga, Andres	Montreal	321	87	10	39	42	30	2	396	101	12

Obs	CrRuns	CrRbi	CrBB	League	Division	Position	nOuts	nAssts	nError	Salary	Div	logSalary
1	30	29	14	American	East	C	446	33	20	.	AE	.
2	321	414	375	National	West	C	632	43	10	475.0	NW	6.16331
3	224	266	263	American	West	1B	880	82	14	480.0	AW	6.17379
4	828	838	354	National	East	RF	200	11	3	500.0	NE	6.21461
5	48	46	33	National	East	1B	805	40	4	91.5	NE	4.51634

NOTE: Multiple versions of the baseball data set, including several revisions, are available from other sources.

Tropical Rain Forest Tree Data

The Sashelp.BEI data set contains the locations of 3,604 trees in a number of tropical rain forests (Condit 1998; Hubbell and Foster 1983; Condit, Hubbell, and Foster 1996). A study window of 1,000 × 500 square kilometers is used. The data set also contains covariates, represented by the variables Gradient and Elevation, which were collected at over 20,301 locations on a regular grid across the study region. The variable Trees distinguishes the event observations in the data set. The following steps display information about the data set Sashelp.BEI and create Figure B.2:

```
title 'BEI Data';
proc contents data=sashelp.bei varnum;
   ods select position;
run;

title 'The First Five Observations Out of 23,905';
proc print data=sashelp.bei(obs=5);
run;

title 'The Trees Variable';
proc freq data=sashelp.bei;
   tables Trees;
run;
```

Figure B.2 Tropical Rain Forest Tree Data

BEI Data

Variables in Creation Order			
#	**Variable**	**Type**	**Len**
1	X	Num	8
2	Y	Num	8
3	Elevation	Num	8
4	Gradient	Num	8
5	Aluminum	Num	8
6	Boron	Num	8
7	Calcium	Num	8
8	Copper	Num	8
9	Iron	Num	8
10	Potassium	Num	8
11	Magnesium	Num	8
12	Manganese	Num	8
13	Sodium	Num	8
14	Phosphorus	Num	8
15	Sulphur	Num	8
16	Zinc	Num	8
17	Ammonia	Num	8
18	Nitrate	Num	8
19	TotalNitro	Num	8
20	MinAmmonia	Num	8
21	MinNitrate	Num	8
22	MinNitrogen	Num	8
23	pH	Num	8
24	Trees	Num	8

The First Five Observations Out of 23,905

Obs	X	Y	Elevation	Gradient	Aluminum	Boron	Calcium	Copper	Iron	Potassium	Magnesium	Manganese
1	11.7	151.1	.	.	.	.	.	.	.	.	.	.
2	998.9	430.5	.	.	.	.	.	.	.	.	.	.
3	980.1	433.5	.	.	.	.	.	.	.	.	.	.
4	986.5	425.8	.	.	.	.	.	.	.	.	.	.
5	944.1	415.1	.	.	.	.	.	.	.	.	.	.

Obs	Sodium	Phosphorus	Sulphur	Zinc	Ammonia	Nitrate	TotalNitro	MinAmmonia	MinNitrate	MinNitrogen	pH	Trees
1	.	.	.	.	.	.	.	.	.	.	.	1
2	.	.	.	.	.	.	.	.	.	.	.	1
3	.	.	.	.	.	.	.	.	.	.	.	1
4	.	.	.	.	.	.	.	.	.	.	.	1
5	.	.	.	.	.	.	.	.	.	.	.	1

Figure B.2 *continued*

The Trees Variable

Trees	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	20601	85.11	20601	85.11
1	3604	14.89	24205	100.00

2003 Birth Weight Data Set

The Sashelp.BirthWgt data set contains 100,000 random observations about infant mortality in 2003 from the US National Center for Health Statistics. Each observation records infant death within one year of birth, birth weight, maternal smoking and drinking behavior, and other background characteristics of the mother. The following steps display information about the Sashelp.BirthWgt data set and create Figure B.3:

```
title '2003 Birth Weight Data';
proc contents data=sashelp.BirthWgt varnum;
   ods select position;
run;

title 'The First Five Observations Out of 100,000';
proc print data=sashelp.BirthWgt (obs=5);
run;
```

Figure B.3 2003 Birth Weight Data

2003 Birth Weight Data

Variables in Creation Order			
#	**Variable**	**Type**	**Len**
1	LowBirthWgt	Char	3
2	Married	Char	3
3	AgeGroup	Num	8
4	Race	Char	9
5	Drinking	Char	3
6	Death	Char	3
7	Smoking	Char	3
8	SomeCollege	Char	3

The First Five Observations Out of 100,000

Obs	LowBirthWgt	Married	AgeGroup	Race	Drinking	Death	Smoking	SomeCollege
1	No	No	3	Asian	No	No	No	Yes
2	No	No	2	White	No	No	No	No
3	Yes	Yes	2	Native	No	Yes	No	No
4	No	No	2	White	No	No	No	No
5	No	No	2	White	No	No	No	Yes

Body Mass Index for Men

The Sashelp.BMIMen data set was collected from the 1999–2000 and 2001–2002 surveys of men that are published by the National Center for Health Statistics. This data set contains the variables BMI and Age. Body mass index (BMI) is defined as the ratio of weight in kilograms to the square of height in meters. BMI is widely used to categorize individuals as overweight or underweight. The following steps display information about the data set Sashelp.BMIMen and create Figure B.4:

```
title 'BMIMen Data';
proc contents data=sashelp.bmimen varnum;
   ods select position;
run;

title 'The First Five Observations Out of 3,264';
proc print data=sashelp.bmimen(obs=5);
run;
```

Figure B.4 Body Mass Index for Men

BMIMen Data

Variables in Creation Order

#	Variable	Type	Len
1	Age	Num	8
2	BMI	Num	8

The First Five Observations Out of 3,264

Obs	Age	BMI
1	2.0	18.6
2	2.0	17.1
3	2.0	19.0
4	2.0	16.8
5	2.1	19.0

Isopod Burrows Data

The Sashelp.Burrows data set contains the locations of 2,089 burrows, which are the habitat of isopods ("pill bugs") in the Negev desert in Israel (Banerjee and Gelfand 2002). Each burrow has a Status variable that records whether a burrow still contains live isopods, has been emptied by an isopod predator, or has failed for some other environmental reason. The data set also contains covariates, which are measured on a square-meter grid over the study region. These covariates include topographical characteristics such as Elevation, Slope, Aspect, and PercentRock (the percentage of rocks). The data set also contains environmental characteristics such as PercentShrub (the percentage of shrubs) and Dew (the time in hours since midnight required to evaporate the dew). The following steps display information about the data set Sashelp.Burrows and create Figure B.5:

```
title 'Burrows Data';
proc contents data=sashelp.burrows varnum;
   ods select position;
run;

title 'The First Five Observations Out of 24,591';
proc print data=sashelp.burrows(obs=5);
run;

proc format;
   value bf -1 = "Killed by Predator" 0 = "Failed" 1 = "Successful";
run;

title 'The Status Variable';
proc freq data=sashelp.burrows;
   tables status;
   format status bf.;
run;
```

Figure B.5 Isopod Burrows Data

Burrows Data

Variables in Creation Order			
#	Variable	Type	Len
1	ID	Num	8
2	X	Num	8
3	Y	Num	8
4	Z	Num	8
5	Status	Num	8
6	Watershed	Num	8
7	BurrowCount	Num	8
8	Elevation	Num	8
9	Aspect	Num	8
10	Slope	Num	8
11	Dew	Num	8
12	PercentRock	Num	8
13	PercentShrub	Num	8
14	Burrow	Num	8

The First Five Observations Out of 24,591

Obs	ID	X	Y	Z	Status	Watershed	BurrowCount	Elevation	Aspect
1	448	128324.66	1031421.00	531.75	1	5	.	.	.
2	679	128344.23	1031391.63	530.79	0	5	.	.	.
3	680	128344.77	1031391.31	530.86	1	5	.	.	.
4	683	128343.09	1031402.81	531.24	0	5	.	.	.
5	686	128343.84	1031415.25	532.11	-1	5	.	.	.

Obs	Slope	Dew	PercentRock	PercentShrub	Burrow
1	.	.	.	.	1
2	.	.	.	.	1
3	.	.	.	.	1
4	.	.	.	.	1
5	.	.	.	.	1

The Status Variable

Status	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Killed by Predator	88	4.21	88	4.21
Failed	1443	69.04	1531	73.25
Successful	559	26.75	2090	100.00
Frequency Missing = 22501				

Bone Marrow Transplant Data

The Sashelp.BMT (bone marrow transplant) data set is used to illustrate survival analysis methods (Klein and Moeschberger 1997). At the time of transplant, each patient is classified into one of three risk categories: ALL (acute lymphoblastic leukemia), AML-Low Risk (acute myelocytic leukemia, low risk), and AML-High Risk. The endpoint of interest is the disease-free survival time, which is the time in days to death, relapse, or the end of the study. In this data set, the variable Group represents the patient's risk category, the variable T represents the disease-free survival time, and the variable Status is the censoring indicator such that the value 1 indicates an event time and the value 0 indicates a censored time. The following steps display information about the Sashelp.BMT data set and create Figure B.6:

```
title 'Bone Marrow Transplant Data';
proc contents data=sashelp.BMT varnum;
   ods select position;
run;

title 'The First Five Observations Out of 137';
proc print data=sashelp.BMT(obs=5);
run;

title 'The Risk Group Variable';
proc freq data=sashelp.BMT;
   tables group;
run;
```

Figure B.6 Bone Marrow Transplant Data

Bone Marrow Transplant Data

Variables in Creation Order

#	Variable	Type	Len	Label
1	Group	Char	13	Disease Group
2	T	Num	8	Disease-Free Survival Time
3	Status	Num	8	Event Indictor: 1=Event 0=Censored

The First Five Observations Out of 137

Obs	Group	T	Status
1	ALL	2081	0
2	ALL	1602	0
3	ALL	1496	0
4	ALL	1462	0
5	ALL	1433	0

Figure B.6 *continued*

The Risk Group Variable

Disease Group				
Group	Frequency	Percent	Cumulative Frequency	Cumulative Percent
ALL	38	27.74	38	27.74
AML-High Risk	45	32.85	83	60.58
AML-Low Risk	54	39.42	137	100.00

1997 Birth Weight Data

The Sashelp.BWeight data set provides 1997 birth weight data from National Center for Health Statistics (Koenker and Hallock 2001; Abreveya 2001). The data record live, singleton births to mothers between the ages of 18 and 45 in the United States who were classified as black or white. The following steps display information about the Sashelp.BWeight data set and create Figure B.7:

```
title '1997 Birth Weight Data';
proc contents data=sashelp.BWeight varnum;
   ods select position;
run;

title 'The First Five Observations Out of 50,000';
proc print data=sashelp.BWeight(obs=5);
run;
```

Figure B.7 Birth Weight Data

1997 Birth Weight Data

Variables in Creation Order				
#	Variable	Type	Len	Label
1	Weight	Num	8	Infant Birth Weight
2	Black	Num	8	Black Mother
3	Married	Num	8	Married Mother
4	Boy	Num	8	Baby Boy
5	MomAge	Num	8	Mother's Age
6	MomSmoke	Num	8	Smoking Mother
7	CigsPerDay	Num	8	Cigarettes Per Day
8	MomWtGain	Num	8	Mother's Pregnancy Weight Gain
9	Visit	Num	8	Prenatal Visit
10	MomEdLevel	Num	8	Mother's Education Level

Figure B.7 *continued*

The First Five Observations Out of 50,000

Obs	Weight	Black	Married	Boy	MomAge	MomSmoke	CigsPerDay	MomWtGain	Visit	MomEdLevel
1	4111	0	1	1	-3	0	0	-16	1	0
2	3997	0	1	0	1	0	0	2	3	2
3	3572	0	1	1	0	0	0	-3	3	0
4	1956	0	1	1	-1	0	0	-5	3	2
5	3515	0	1	1	-6	0	0	-20	3	0

Class Data

The Sashelp.Class data set provides information about a small fictitious class of students. Variables include Sex, Age, Height, and Weight. This data set is frequently used in SAS documentation to illustrate basic SAS coding. The following steps display information about the Sashelp.Class data set and create Figure B.8:

```
title 'Class Data';
proc contents data=sashelp.Class varnum;
   ods select position;
run;

title 'The Full Data Set';
proc print data=sashelp.Class;
run;
```

Figure B.8 Class Data

Class Data

Variables in Creation Order			
#	**Variable**	**Type**	**Len**
1	Name	Char	8
2	Sex	Char	1
3	Age	Num	8
4	Height	Num	8
5	Weight	Num	8

Figure B.8 *continued*

The Full Data Set

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5
5	Henry	M	14	63.5	102.5
6	James	M	12	57.3	83.0
7	Jane	F	12	59.8	84.5
8	Janet	F	15	62.5	112.5
9	Jeffrey	M	13	62.5	84.0
10	John	M	12	59.0	99.5
11	Joyce	F	11	51.3	50.5
12	Judy	F	14	64.3	90.0
13	Louise	F	12	56.3	77.0
14	Mary	F	15	66.5	112.0
15	Philip	M	16	72.0	150.0
16	Robert	M	12	64.8	128.0
17	Ronald	M	15	67.0	133.0
18	Thomas	M	11	57.5	85.0
19	William	M	15	66.5	112.0

Comet Data

The Sashelp.Comet data set provides information from the following experiment. Twenty-four male rats were divided into four groups. Three groups received a daily oral dose of a 1,2-dimethylhydrazine dihydrochloride in three dose levels (low, medium, and high, respectively); the fourth group was a control group. Three additional animals received a positive control. Cell suspensions for each animal were scored for DNA damage by using a comet assay (Ghebretinsae et al. 2013). The following steps display information about the Sashelp.Comet data set and create Figure B.9:

```
title 'Comet Data';
proc contents data=sashelp.Comet varnum;
   ods select position;
run;

title 'The First Five Observations Out of 4,050';
proc print data=sashelp.Comet(obs=5);
run;
```

Figure B.9 Comet Data

Comet Data

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	Dose	Num	8	1,2 Dimethylhydrazine dihydrochloride Dose Level
2	Rat	Num	8	Rat Index
3	Sample	Num	8	Slide Index of Grouped Cells from a Rat
4	Length	Num	8	Tail Length of the Comet

The First Five Observations Out of 4,050

Obs	Dose	Rat	Sample	Length
1	0	1	1	15.3527
2	0	1	1	16.1826
3	0	1	1	14.9378
4	0	1	1	12.4481
5	0	1	1	12.8631

El Niño–Southern Oscillation Data

The Sashelp.ENSO (El Niño–Southern Oscillation) data set contains measurements of monthly averaged atmospheric pressure differences between Easter Island and Darwin, Australia, for a period of 168 months (National Institute of Standards and Technology 1998). These pressure differences drive the southern trade winds. This data set is used to illustrate fitting nonlinear functions to a scatter plot by using methods such as loess and penalized B-splines. These data show both seasonal variations and variations due to El Niño. The following steps display information about the Sashelp.ENSO data set and create Figure B.10:

```
title 'El Nino Southern Oscillation Data';
proc contents data=sashelp.ENSO varnum;
   ods select position;
run;

title 'The First Five Observations Out of 168';
proc print data=sashelp.ENSO(obs=5);
run;
```

Figure B.10 El Niño–Southern Oscillation Data

El Nino Southern Oscillation Data

Variables in Creation Order			
#	**Variable**	**Type**	**Len**
1	Month	Num	8
2	Year	Num	8
3	Pressure	Num	8

Figure B.10 *continued*

The First Five Observations Out of 168

Obs	Month	Year	Pressure
1	1	0.08333	12.9
2	2	0.16667	11.3
3	3	0.25000	10.6
4	4	0.33333	11.2
5	5	0.41667	10.9

Finland's Lake Laengelmaevesi Fish Catch Data

The Sashelp.Fish catch data set contains measurements of 159 fish that were caught in Finland's Lake Laengelmaevesi (Puranen 1917); it is used to illustrate discriminant analysis. For each of the seven species (bream, roach, whitefish, parkki, perch, pike, and smelt), the weight, length, height, and width of each fish are tallied. Three different length measurements are recorded: from the nose of the fish to the beginning of its tail, from the nose to the notch of its tail, and from the nose to the end of its tail. The height and width are recorded as percentages of the third length variable. The following steps display information about the Sashelp.Fish data set and create Figure B.11:

```
title 'Finland''s Lake Laengelmaevesi Fish Catch Data';
proc contents data=sashelp.Fish varnum;
   ods select position;
run;

title 'The First Five Observations Out of 159';
proc print data=sashelp.Fish(obs=5);
run;

title 'The Fish Species Variable';
proc freq data=sashelp.Fish;
   tables species;
run;
```

Figure B.11 Finland's Lake Laengelmaevesi Fish Catch Data

Finland's Lake Laengelmaevesi Fish Catch Data

Variables in Creation Order			
#	**Variable**	**Type**	**Len**
1	Species	Char	9
2	Weight	Num	8
3	Length1	Num	8
4	Length2	Num	8
5	Length3	Num	8
6	Height	Num	8
7	Width	Num	8

Figure B.11 *continued*

The First Five Observations Out of 159

Obs	Species	Weight	Length1	Length2	Length3	Height	Width
1	Bream	242	23.2	25.4	30.0	11.5200	4.0200
2	Bream	290	24.0	26.3	31.2	12.4800	4.3056
3	Bream	340	23.9	26.5	31.1	12.3778	4.6961
4	Bream	363	26.3	29.0	33.5	12.7300	4.4555
5	Bream	430	26.5	29.0	34.0	12.4440	5.1340

The Fish Species Variable

Species	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Bream	35	22.01	35	22.01
Parkki	11	6.92	46	28.93
Perch	56	35.22	102	64.15
Pike	17	10.69	119	74.84
Roach	20	12.58	139	87.42
Smelt	14	8.81	153	96.23
Whitefish	6	3.77	159	100.00

Exhaust Emissions Data

The Sashelp.Gas data set contains data from an experiment about gasoline engine exhaust emissions (Brinkman 1981). Nitrogen oxide emissions from a single-cylinder engine are measured for various combinations of fuel, compression ratio, and equivalence ratio. This data set is used to illustrate how to fit models by using nonlinearly transformed data. The following steps display information about the Sashelp.Gas data set and create Figure B.12:

```
title 'Exhaust Emissions Data';
proc contents data=sashelp.Gas varnum;
   ods select position;
run;

title 'The First Five Observations Out of 171';
proc print data=sashelp.Gas(obs=5);
run;

title 'The Fuel Type Variable';
proc freq data=sashelp.Gas;
   tables fuel;
run;
```

Figure B.12 Exhaust Emissions Data

Exhaust Emissions Data

Variables in Creation Order				
#	Variable	Type	Len	Label
1	Fuel	Char	8	
2	CpRatio	Num	8	Compression Ratio
3	EqRatio	Num	8	Equivalence Ratio
4	NOx	Num	8	Nitrogen Oxide

The First Five Observations Out of 171

Obs	Fuel	CpRatio	EqRatio	NOx
1	Ethanol	12	0.907	3.741
2	Ethanol	12	0.761	2.295
3	Ethanol	12	1.108	1.498
4	Ethanol	12	1.016	2.881
5	Ethanol	12	1.189	0.760

The Fuel Type Variable

Fuel	Frequency	Percent	Cumulative Frequency	Cumulative Percent
82rongas	9	5.26	9	5.26
94%Eth	25	14.62	34	19.88
Ethanol	90	52.63	124	72.51
Gasohol	13	7.60	137	80.12
Indolene	22	12.87	159	92.98
Methanol	12	7.02	171	100.00

Fisher (1936) Iris Data

The Sashelp.Iris data set (Fisher 1936) is widely used for examples of discriminant analysis and cluster analysis. The data are measurements in millimeters of the sepal length, sepal width, petal length, and petal width of 50 iris specimens from each of three species: *Iris setosa*, *I. versicolor*, and *I. virginica*. The following steps display information about the Sashelp.Iris data set and create Figure B.13:

```
title 'Fisher (1936) Iris Data';
proc contents data=sashelp.Iris varnum;
   ods select position;
run;

title 'The First Five Observations Out of 150';
proc print data=sashelp.Iris(obs=5);
run;

title 'The Iris Species Variable';
proc freq data=sashelp.Iris;
```

```
   tables species;
run;
```

Figure B.13 Fisher (1936) Iris Data

Fisher (1936) Iris Data

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	Species	Char	10	Iris Species
2	SepalLength	Num	8	Sepal Length (mm)
3	SepalWidth	Num	8	Sepal Width (mm)
4	PetalLength	Num	8	Petal Length (mm)
5	PetalWidth	Num	8	Petal Width (mm)

The First Five Observations Out of 150

Obs	Species	SepalLength	SepalWidth	PetalLength	PetalWidth
1	Setosa	50	33	14	2
2	Setosa	46	34	14	3
3	Setosa	46	36	10	2
4	Setosa	51	33	17	5
5	Setosa	55	35	13	2

The Iris Species Variable

Iris Species				
Species	**Frequency**	**Percent**	**Cumulative Frequency**	**Cumulative Percent**
Setosa	50	33.33	50	33.33
Versicolor	50	33.33	100	66.67
Virginica	50	33.33	150	100.00

Junk Email Data

The Sashelp.JunkMail data set comes from a study that classifies whether an email is junk email (coded as 1) or not (coded as 0). The data were collected in Hewlett-Packard labs and donated by George Forman. The data set contains 4,601 observations with 59 variables. The response variable is a binary indicator of whether an email is considered spam or not. There are 57 predictor variables that record frequencies of some common words and characters and lengths of uninterrupted sequences of capital letters in emails. The following steps display information about the Sashelp.JunkMail data set and create Figure B.14:

```
title 'Junk Email Data';
proc contents data=sashelp.JunkMail varnum;
   ods select position;
run;

title 'The First Five Observations Out of 4,601';
proc print data=sashelp.JunkMail(obs=5);
run;
```

Figure B.14 Junk Email Data

Junk Email Data

Variables in Creation Order				
#	Variable	Type	Len	Label
1	Test	Num	8	0 - Training, 1 - Test
2	Make	Num	8	
3	Address	Num	8	
4	All	Num	8	
5	_3D	Num	8	3D
6	Our	Num	8	
7	Over	Num	8	
8	Remove	Num	8	
9	Internet	Num	8	
10	Order	Num	8	
11	Mail	Num	8	
12	Receive	Num	8	
13	Will	Num	8	
14	People	Num	8	
15	Report	Num	8	
16	Addresses	Num	8	
17	Free	Num	8	
18	Business	Num	8	
19	Email	Num	8	
20	You	Num	8	
21	Credit	Num	8	
22	Your	Num	8	
23	Font	Num	8	
24	_000	Num	8	000
25	Money	Num	8	

Figure B.14 *continued*

Junk Email Data

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
26	HP	Num	8	
27	HPL	Num	8	
28	George	Num	8	
29	_650	Num	8	650
30	Lab	Num	8	
31	Labs	Num	8	
32	Telnet	Num	8	
33	_857	Num	8	857
34	Data	Num	8	
35	_415	Num	8	415
36	_85	Num	8	85
37	Technology	Num	8	
38	_1999	Num	8	1999
39	Parts	Num	8	
40	PM	Num	8	
41	Direct	Num	8	
42	CS	Num	8	
43	Meeting	Num	8	
44	Original	Num	8	
45	Project	Num	8	
46	RE	Num	8	
47	Edu	Num	8	
48	Table	Num	8	
49	Conference	Num	8	
50	Semicolon	Num	8	
51	Paren	Num	8	
52	Bracket	Num	8	
53	Exclamation	Num	8	
54	Dollar	Num	8	
55	Pound	Num	8	
56	CapAvg	Num	8	Capital Run Length Average
57	CapLong	Num	8	Capital Run Length Longest
58	CapTotal	Num	8	Capital Run Length Total
59	Class	Num	8	0 - Not Junk, 1 - Junk

Figure B.14 *continued*

The First Five Observations Out of 4,601

Obs	Test	Make	Address	All	_3D	Our	Over	Remove	Internet	Order	Mail	Receive	Will	People	Report
1	1	0.00	0.64	0.64	0	0.32	0.00	0.00	0.00	0.00	0.00	0.00	0.64	0.00	0.00
2	0	0.21	0.28	0.50	0	0.14	0.28	0.21	0.07	0.00	0.94	0.21	0.79	0.65	0.21
3	1	0.06	0.00	0.71	0	1.23	0.19	0.19	0.12	0.64	0.25	0.38	0.45	0.12	0.00
4	0	0.00	0.00	0.00	0	0.63	0.00	0.31	0.63	0.31	0.63	0.31	0.31	0.31	0.00
5	0	0.00	0.00	0.00	0	0.63	0.00	0.31	0.63	0.31	0.63	0.31	0.31	0.31	0.00

Obs	Addresses	Free	Business	Email	You	Credit	Your	Font	_000	Money	HP	HPL	George	_650	Lab	Labs
1	0.00	0.32	0.00	1.29	1.93	0.00	0.96	0	0.00	0.00	0	0	0	0	0	0
2	0.14	0.14	0.07	0.28	3.47	0.00	1.59	0	0.43	0.43	0	0	0	0	0	0
3	1.75	0.06	0.06	1.03	1.36	0.32	0.51	0	1.16	0.06	0	0	0	0	0	0
4	0.00	0.31	0.00	0.00	3.18	0.00	0.31	0	0.00	0.00	0	0	0	0	0	0
5	0.00	0.31	0.00	0.00	3.18	0.00	0.31	0	0.00	0.00	0	0	0	0	0	0

Obs	Telnet	_857	Data	_415	_85	Technology	_1999	Parts	PM	Direct	CS	Meeting	Original	Project	RE	Edu
1	0	0	0	0	0	0	0.00	0	0	0.00	0	0	0.00	0	0.00	0.00
2	0	0	0	0	0	0	0.07	0	0	0.00	0	0	0.00	0	0.00	0.00
3	0	0	0	0	0	0	0.00	0	0	0.06	0	0	0.12	0	0.06	0.06
4	0	0	0	0	0	0	0.00	0	0	0.00	0	0	0.00	0	0.00	0.00
5	0	0	0	0	0	0	0.00	0	0	0.00	0	0	0.00	0	0.00	0.00

Obs	Table	Conference	Semicolon	Paren	Bracket	Exclamation	Dollar	Pound	CapAvg	CapLong	CapTotal	Class
1	0	0	0.00	0.000	0	0.778	0.000	0.000	3.756	61	278	1
2	0	0	0.00	0.132	0	0.372	0.180	0.048	5.114	101	1028	1
3	0	0	0.01	0.143	0	0.276	0.184	0.010	9.821	485	2259	1
4	0	0	0.00	0.137	0	0.137	0.000	0.000	3.537	40	191	1
5	0	0	0.00	0.135	0	0.135	0.000	0.000	3.537	40	191	1

Leukemia Data Sets

The Sashelp.LeuTrain and Sashelp.LeuTest data sets provide microarray data from (Golub et al. 1999; Zou and Hastie 2005). The Sashelp.LeuTrain data set consists of 7,129 genes and 38 training samples, and the Sashelp.LeuTest data set consists of the same 7,129 genes and 34 testing samples. Among the 38 training samples, 27 are type 1 leukemia (acute lymphoblastic leukemia, coded in the data as 1) and 11 are type 2 leukemia (acute myeloid leukemia, coded in the data as –1).

The following steps display information about Sashelp.LeuTrain data set and create Figure B.15:

```
title 'Leukemia Training Data';
proc contents data=sashelp.LeuTrain varnum;
   ods select position;
run;

title 'The First Five Observations and 11 Variables';
proc print data=sashelp.LeuTrain(obs=5);
```

```
   var y x1-x10;
run;

title 'Leukemia Type Variable';
proc freq data=sashelp.LeuTrain;
   tables y;
run;
```

Figure B.15 Leukemia Training Data

The First Five Observations and 11 Variables

Obs	y	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10
1	1	-1.46240	-0.64514	-0.83593	-1.47040	-0.91997	-1.58430	0.71239	-0.54229	1.05090	0.23649
2	1	-0.66480	0.20615	-0.36857	0.25822	-0.47567	-0.35497	-1.11940	-0.29251	-0.37542	-0.38760
3	1	-0.20049	0.37994	-2.38280	0.43960	-1.22700	-1.76220	0.10464	-1.80750	0.49292	-1.67000
4	1	-0.25776	0.27994	1.83920	-1.62950	-1.28750	-1.26510	0.76334	-0.61645	-0.31578	-0.32193
5	1	-0.56457	-0.39588	-0.98372	-0.83741	-0.41477	0.14834	-0.03550	-0.10022	-0.75753	0.37068

Leukemia Type Variable

y	Frequency	Percent	Cumulative Frequency	Cumulative Percent
-1	11	28.95	11	28.95
1	27	71.05	38	100.00

The results of the PROC CONTENTS step are not displayed. The results show that there are 7,130 variables, y and x1–x7129.

The following steps display information about Sashelp.LeuTest data set and create Figure B.16:

```
title 'Leukemia Test Data';
proc contents data=sashelp.LeuTest varnum;
   ods select position;
run;

title 'The First Five Observations and 11 Variables';
proc print data=sashelp.LeuTest(obs=5);
   var y x1-x10;
run;

title 'Leukemia Type Variable';
proc freq data=sashelp.LeuTest;
   tables y;
run;
```

Figure B.16 Leukemia Test Data

The First Five Observations and 11 Variables

Obs	y	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10
1	1	-1.38240	0.06288	0.62252	1.61210	0.52179	0.11516	-1.85270	-0.39956	0.88007	-0.86565
2	1	0.65192	-0.35476	2.29630	1.64980	0.50211	-0.37315	1.76820	-1.74270	1.63080	0.60171
3	1	0.65409	1.41340	0.22593	-0.06719	0.30015	0.76964	-0.26212	0.94481	-0.51884	-0.60999
4	1	1.07220	0.01959	0.16875	0.84779	0.24533	0.79682	0.41442	0.35122	-0.70177	1.85410
5	1	2.12480	1.66370	-0.35986	1.15850	0.89379	0.56310	-0.92476	0.56790	-0.56039	-2.12400

Leukemia Type Variable

y	Frequency	Percent	Cumulative Frequency	Cumulative Percent
-1	14	41.18	14	41.18
1	20	58.82	34	100.00

The results of the PROC CONTENTS step are not displayed. The results show that there are 7,130 variables, y and x1–x7129.

Margarine Data

The Sashelp.Margarin data set is a scanner panel data set that lists purchases of margarine (Rossi, Allenby, and McCulloch 2005). There are 313 households and a total of 3,405 purchases. The variable HouseID represents the household ID; each household made at least five purchases, which are defined by the choice set variable Set. The variable Choice represents the choice that households made among the six margarine brands for each purchase or choice set. The variable Brand has the value PPK for Parkay stick, PBB for Blue Bonnet stick, PFL for Fleischmann's stick, PHse for the house brand stick, PGen for the generic stick, and PSS for Shedd's Spread tub. The variable LogPrice is the logarithm of the product price. The variables LogInc and FamSize provide information about household income and family size, respectively. The following steps display information about the Sashelp.Margarin data set and create Figure B.17:

```
title 'Margarine Data';
proc contents data=sashelp.Margarin varnum;
   ods select position;
run;

title 'The First Six Observations Out of 20,430';
proc print data=sashelp.Margarin(obs=6);
run;
```

Figure B.17 Margarine Data

Margarine Data

Variables in Creation Order			
#	**Variable**	**Type**	**Len**
1	HouseID	Num	8
2	Set	Num	8
3	Choice	Num	8
4	Brand	Char	8
5	LogPrice	Num	8
6	LogInc	Num	8
7	FamSize	Num	8

The First Six Observations Out of 20,430

Obs	HouseID	Set	Choice	Brand	LogPrice	LogInc	FamSize
1	2100016	1	1	PPk	-0.41552	3.48124	2
2	2100016	1	0	PBB	-0.40048	3.48124	2
3	2100016	1	0	PFl	0.08618	3.48124	2
4	2100016	1	0	PHse	-0.56212	3.48124	2
5	2100016	1	0	PGen	-1.02165	3.48124	2
6	2100016	1	0	PSS	-0.16252	3.48124	2

Flying Mileages between 10 US Cities Data

The Sashelp.Mileages data set contains a table of flying mileages between 10 US cities. This data set is frequently used to illustrate cluster analysis and multidimensional scaling. The following steps display information about the Sashelp.Mileages data set and create Figure B.18:

```
title 'Flying Mileages between 10 US Cities Data';
proc contents data=sashelp.Mileages varnum;
   ods select position;
run;

title 'The Full Data Set';
proc print data=sashelp.Mileages noobs;
   id city;
run;
```

Figure B.18 Flying Mileages between 10 US Cities Data

Flying Mileages between 10 US Cities Data

Variables in Creation Order			
#	**Variable**	**Type**	**Len**
1	Atlanta	Num	8
2	Chicago	Num	8
3	Denver	Num	8
4	Houston	Num	8
5	LosAngeles	Num	8
6	Miami	Num	8
7	NewYork	Num	8
8	SanFrancisco	Num	8
9	Seattle	Num	8
10	WashingtonDC	Num	8
11	City	Char	15

The Full Data Set

City	Atlanta	Chicago	Denver	Houston	LosAngeles	Miami	NewYork	SanFrancisco	Seattle	WashingtonDC
Atlanta	0	.	.	.	.	.	.	.	.	.
Chicago	587	0	.	.	.	.	.	.	.	.
Denver	1212	920	0	.	.	.	.	.	.	.
Houston	701	940	879	0	.	.	.	.	.	.
Los Angeles	1936	1745	831	1374	0	.	.	.	.	.
Miami	604	1188	1726	968	2339	0	.	.	.	.
New York	748	713	1631	1420	2451	1092	0	.	.	.
San Francisco	2139	1858	949	1645	347	2594	2571	0	.	.
Seattle	2182	1737	1021	1891	959	2734	2408	678	0	.
Washington D.C.	543	597	1494	1220	2300	923	205	2442	2329	0

Earthquake Locations in the United States

The Sashelp.Quakes data set provides locations and measurements of earthquakes in the United States. These data are provided courtesy of the United States Geological Service. The following steps display information about the Sashelp.Quakes data set and create Figure B.19:

```
title 'Earthquake Locations in the United States';
proc contents data=sashelp.Quakes varnum;
   ods select position;
run;

title 'The First Five Observations Out of 15,578';
proc print data=sashelp.Quakes(obs=5);
run;
```

Figure B.19 Earthquake Data

Earthquake Locations in the United States

Variables in Creation Order			
#	Variable	Type	Len
1	Latitude	Num	8
2	Longitude	Num	8
3	Depth	Num	8
4	Magnitude	Num	8
5	dNearestStation	Num	8
6	RootMeanSquareTime	Num	8
7	Type	Char	10

The First Five Observations Out of 15,578

Obs	Latitude	Longitude	Depth	Magnitude	dNearestStation	RootMeanSquareTime	Type
1	35.6880	-121.129	6.2500	2.75	0.02121	0.0700	earthquake
2	41.9009	-119.622	0.0000	4.59	0.50700	0.0742	earthquake
3	41.8836	-119.641	0.5329	3.19	0.49300	0.0000	earthquake
4	41.8897	-119.640	0.5077	3.12	0.49400	0.0119	earthquake
5	40.8783	-123.270	33.5700	2.60	0.22160	0.0600	earthquake

Hot-Spring Locations in the United States

The Sashelp.Springs data set provides locations and measurements of hot springs in the United States. These data are provided courtesy of the National Oceanic and Atmospheric Administration. The following steps display information about the Sashelp.Springs data set and create Figure B.20:

```
title 'Hot Spring Locations in the United States';
proc contents data=sashelp.Springs varnum;
   ods select position;
run;

title 'The First Five Observations Out of 1,587';
proc print data=sashelp.Springs(obs=5);
run;
```

Figure B.20 Hot Spring Data

Hot Spring Locations in the United States

Variables in Creation Order			
#	**Variable**	**Type**	**Len**
1	Latitude	Num	8
2	Longitude	Num	8
3	Name	Char	34
4	Area	Char	14
5	Type	Char	9
6	Farenheit	Num	8
7	Celsius	Num	8

The First Five Observations Out of 1,587

Obs	Latitude	Longitude	Name	Area	Type	Farenheit	Celsius
1	51.925	-177.160	Fumaroles on Kanaga Island	Adak	Hotspring	219	104
2	65.858	-164.710	Serpentine Hot Springs	Bendeleben	Hotspring	171	77
3	53.851	-166.918	Hot Springs Near Makushin Volcano	Unalaska	Hotspring	H	H
4	53.877	-166.448	Hot Springs Near Summer Bay	Unalaska	Hotspring	H	H
5	52.270	-174.042	Hot Springs on Atka Island	Atka	Hotspring	H	H

Coal Seam Thickness Data

The Sashelp.Thick data set simulates measurements of coal seam thickness (in feet) taken over an approximately square area. The variable Thick contains the thickness values. The coordinates are offsets from a point in the southwest corner of the measurement area, where the unit for the north and east distances is 1,000 feet. The following steps display information about the Sashelp.Thick data set and create Figure B.21:

```
title 'Coal Seam Thickness Data';
proc contents data=sashelp.Thick varnum;
   ods select position;
run;

title 'The First Five Observations Out of 75';
proc print data=sashelp.Thick(obs=5);
run;
```

Figure B.21 Coal Seam Thickness Data

Coal Seam Thickness Data

Variables in Creation Order				
#	**Variable**	**Type**	**Len**	**Label**
1	East	Num	8	
2	North	Num	8	
3	Thick	Num	8	Coal Seam Thickness

The First Five Observations Out of 75

Obs	East	North	Thick
1	0.7	59.6	34.1
2	2.1	82.7	42.2
3	4.7	75.1	39.5
4	4.8	52.8	34.3
5	5.9	67.1	37.0

1980 US Presidential Election Data

The Sashelp.Vote1980 data set contains US county votes-cast proportions and demographic and geographic characteristics for 3,107 US counties in the 1980 presidential election (Pace and Barry 1997). The six explanatory variables are as follows: the population 18 years of age or older (Pop), the population with 12th-grade or higher education (Edu), the number of owned housing units (Houses), the aggregate income (Income), and scaled longitude and latitude of geographic centroids (Longitude, Latitude). The dependent variable LogVoteRate is the logarithm of the proportion of votes cast divided by the variable Pop. The following steps display information about the data set Sashelp.Vote1980 and create Figure B.22:

```
title 'US 1980 Presidential Election Data';
proc contents data=sashelp.vote1980 varnum;
   ods select position;
run;

title 'The First Five Observations Out of 3,107';
proc print data=sashelp.vote1980(obs=5);
run;
```

Figure B.22 US 1980 Presidential Election Data

US 1980 Presidential Election Data

Variables in Creation Order

#	Variable	Type	Len	Label
1	LogVoteRate	Num	8	Log Votes Cast per County
2	Pop	Num	8	Population of 18 Years and Older
3	Edu	Num	8	Population with 12th Grade and Higher
4	Houses	Num	8	Number of Owned Housing Units
5	Income	Num	8	Aggregate Income
6	Longitude	Num	8	Scaled Longitude
7	Latitude	Num	8	Scaled Latitude

The First Five Observations Out of 3,107

Obs	LogVoteRate	Pop	Edu	Houses	Income	Longitude	Latitude
1	-0.66156	9.9729	9.2463	9.00405	12.1349	-0.86641	0.32542
2	-0.65086	10.9033	10.2212	9.96576	13.0566	-0.87755	0.30655
3	-0.61711	9.7222	8.7535	8.70765	11.6306	-0.85389	0.31863
4	-0.63907	9.2737	8.1831	8.27741	11.2437	-0.87127	0.32997
5	-0.70027	10.1515	9.2077	9.24068	12.1551	-0.86566	0.33980

References

Abreveya, J. (2001). "The Effects of Demographics and Maternal Behavior on the Distribution of Birth Outcomes." *Journal of Economics* 26:247–257.

Banerjee, S., and Gelfand, A. E. (2002). "Prediction, Interpolation, and Regression for Spatially Misaligned Data." *Sankhyā* 64:227–245.

Brinkman, N. D. (1981). "Ethanol Fuel: A Single-Cylinder Engine Study of Efficiency and Exhaust Emissions." *Society of Automotive Engineers Transactions* 90:1410–1424.

Condit, R. (1998). *Tropical Forest Census Plots: Methods and Results from Barro Colorado Island, Panama, and a Comparison with Other Plots*. Berlin: Springer-Verlag.

Condit, R., Hubbell, S. P., and Foster, R. B. (1996). "Changes in Tree Species Abundance in a Neotropical Forest: Impact of Climate Change." *Journal of Tropical Ecology* 12:231–256.

Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems." *Annals of Eugenics* 7:179–188.

Ghebretinsae, A. H., Faes, C., Molenberghs, G., De Boeck, M., and Geys, H. (2013). "A Bayesian, Generalized Frailty Model for Comet Assays." *Journal of Biopharmaceutical Statistics* 23:618–636.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., et al. (1999). "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression." *Science* 286:531–537.

Hubbell, S. P., and Foster, R. B. (1983). "Diversity of Canopy Trees in a Neotropical Forest and Implications for the Conservation of Tropical Trees." In *Tropical Rain Forest: Ecology and Management*, edited by S. J. Sutton, T. C. Whitmore, and A. C. Chadwick, 25–41. Oxford: Blackwell.

Klein, J. P., and Moeschberger, M. L. (1997). *Survival Analysis: Techniques for Censored and Truncated Data*. New York: Springer-Verlag.

Koenker, R., and Hallock, K. (2001). "Quantile Regression." *Journal of Economic Perspectives* 15:143–156.

National Institute of Standards and Technology (1998). "Statistical Reference Data Sets." Accessed June 6, 2011. http://www.itl.nist.gov/div898/strd/general/dataarchive.html.

Pace, R. K., and Barry, R. (1997). "Quick Computation of Spatial Autoregressive Estimators." *Geographical Analysis* 29:232–247.

Puranen, J. (1917). "Fish Catch data set (1917)." Journal of Statistics Education Data Archive. Accessed February 17, 2022. http://jse.amstat.org/jse_data_archive.htm.

Rossi, P. E., Allenby, G. M., and McCulloch, R. (2005). *Bayesian Statistics and Marketing*. Chichester, UK: John Wiley & Sons.

Time Inc. (1987). "What They Make." *Sports Illustrated* (April 20): 54–81.

Zou, H., and Hastie, T. (2005). "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society, Series B* 67:301–320.

Appendix C
Known Issues in Using CAS Tables with SAS/STAT Procedures

Contents

Summary of CAS Table Issues **11137**
SAS/STAT Procedures Incompatible with Input CAS Tables **11138**
Options Incompatible with Input CAS Tables **11138**
Special Data Sets Incompatible with CAS Tables **11139**
Special Requirements for Processing Some CAS Tables **11140**
Statistical Results Not Reliably Reproducible with Input CAS Tables **11141**
Observations Might Not Be Identified Reliably **11142**
Using observation numbers to identify observations is not reliable 11142
There is no definite mapping of output data set observations to input CAS table observations 11144
Procedure options that refer to observation numbers cannot be applied 11145

Summary of CAS Table Issues

This chapter lists known issues in using CAS tables with SAS/STAT procedures. For more information about the origin of some of these issues, as well as detailed descriptions and remedial actions that you can take to address them, see the section "Using CAS Tables" on page 420 of Chapter 20, "Shared Concepts and Topics."

SAS/STAT Procedures Incompatible with Input CAS Tables

The following SAS/STAT procedures do not support CAS table input:

- BCHOICE
- BOXPLOT
- CALIS
- INBREED
- KRIGE2D
- MCMC
- MDS
- NESTED
- NLMIXED
- SIM2D
- SPP
- VARIOGRAM

Options Incompatible with Input CAS Tables

Table C.1 summarizes known procedure options or specifications that are incompatible with input CAS tables. In these situations, you must instead input SAS data sets that have the intended order of observations.

Table C.1 Procedure Options Incompatible with Input CAS Tables

Procedures	Options or Specifications
Any applicable procedures	ORDER=DATA, ORDER=FREQDATA, and RORDER=DATA options in the PROC statement or other statements
ANOVA, GLM	ABSORB option
DISTANCE	ORDER=DSORDER option for ordinal variables in the VAR statement
GLIMMIX	Using the internal variable _N_ in programming statements; RESIDUAL option and _RESIDUAL_ keyword in the RANDOM statement

Table C.1 *continued*

Procedures	Options or Specifications
GLIMMIX, MIXED	Continuous effect specifications for the SUBJECT= and GROUP= options in the RANDOM statement
HPMIXED	REPEATED statement specifications
MIXED	Continuous effect specifications for the SUBJECT= and GROUP= options in the REPEATED statement; an empty effect specification in the REPEATED statement
QUANTREG	OBS= option in the CONDDIST statement

For all options, except the OBS= option in the QUANTREG procedure, the corresponding procedure exits with an error message. The OBS= option is ignored and a warning message is issued.

Special Data Sets Incompatible with CAS Tables

Table C.2 summarizes known special data sets in procedures that are incompatible with input or output CAS tables. In these cases, you must instead use SAS data sets.

Table C.2 Special Data Sets Incompatible with CAS Tables

Procedures	Special Data Sets
Any applicable procedures	Item stores output
CLUSTER, MODECLUS	DATA= data set of TYPE=DISTANCE
CORRESP	DATA= data set for a Burt table
DISTANCE	OUT= data set of TYPE=DISTANCE
FREQ	TESTP= and TESTF= data sets for the CHISQ option in the TABLES statement
GEE, GENMOD	ZDATA= data set in the REPEATED statement
GLIMMIX	KNOTMETHOD=DATA data set in the RANDOM statement
GLIMMIX, HPMIXED, MIXED	PARMSDATA= data set in the PARMS statement (except in some limited special data situations)
MIANALYZE	DATA= data set that uses two or more observations to store estimates and standard errors for a single imputation
MIXED	LOCAL=POM data set in the REPEATED statement

Table C.2 *continued*

Procedures	Special Data Sets
SURVEYFREQ, SURVEYIMPUTE, SURVEYLOGISTIC, SURVEYMEANS, SURVEYPHREG, SURVEYREG	JKCOEFS= and REPCOEFS= data sets in the REPWEIGHTS statement
TRANSREG	OUT= data set when there is more than one dependent variable

Special Requirements for Processing Some CAS Tables

Table C.3 summarizes requirements that must be specified when you use certain CAS tables as input for SAS/STAT procedures. If these requirements are not satisfied, the procedures exit with an error message. To proceed, you can either ensure that the requirements are met or input a SAS data set that has the required data order.

Table C.3 Special Requirements for Using Some CAS Tables as Input Data

Procedures	Special Requirements
GEE	The WITHINSUBJECT= option is required for ALR models, weighted GEE models, and GEE models with working correlation structure when you use a CAS table as the input data.
GENMOD, LIFEREG, PHREG	The _NAME_ variable must be specified when you use a CAS table as the INPUT= data set for the normal priors in the BAYES statement.
GLIMMIX	The PARMS option must be specified when you use a CAS table as the TESTDATA= data set in the COVTEST statement. The ID statement must not be specified together with the SUBGRAD= option when you use a CAS table as the input data set. The NOVAR suboption of the OUTDESIGN= option must not be specified when you use a CAS table as the input data set.
PRINQUAL	The ID variables must be specified in the ID statement when you use a CAS table as the input data set or write an OUT= data set to a CAS table.

Statistical Results Not Reliably Reproducible with Input CAS Tables

Table C.4 summarizes known nonreproducibility issues of analysis methods in SAS/STAT procedures when you use a CAS table as input in the DATA= option. In these situations, the procedure issues a note about nonreproducibility or suppresses the nonreproducible statistical results.

Table C.4 Statistical Methods That Do Not Reproduce Results Reliably with Input CAS Tables

Procedures	Statistical Methods
ADAPTIVEREG, GLMSELECT, HPGENSELECT, HPLOGISTIC, HPPLS, HPQUANSELECT, HPREG, HPSPLIT, QUANTSELECT	Partitioning of observations (for model training, validation, and testing) based on random assignments of roles via the FRACTION= option in the PARTITION statement
BGLIMM	Missing data that require imputations
BGLIMM, PHREG	Bayesian methods with random effect specifications and/or imputations of missing responses
CAUSALMED, CAUSALTRT, QUANTLIFE, QUANTREG, TTEST	Bootstrap or resampling methods
CLUSTER	All cluster methods when there are tied distances
FMM, HPFMM	Data augmentation methods
GLIMMIX	Estimation of the **V** matrix and its inverse, the Cholesky root of the **V** matrix and its inverse, and the correlation matrix corresponding to the estimated **V** matrix when the SUBJECT= option is not specified in the RANDOM statement
GLM	Autocorrelation statistics and Durbin-Watson *D* (results are suppressed)
GLMSELECT, HPREG	Model averaging
MI	All multiple imputation methods
PHREG	Standard errors of Uno's concordance statistics
PLS	Cross validation methods and tests

Table C.4 *continued*

Procedures	Statistical Methods
PSMATCH	Greedy matching when there are tied propensity scores
SURVEYSELECT	Probability sampling methods
SURVEYFREQ, SURVEYIMPUTE, SURVEYLOGISTIC, SURVEYMEANS, SURVEYPHREG, SURVEYREG	Balanced repeated replication (BRR) and bootstrap methods
TREE	Tree diagrams

The nonreproducibility issue is not limited to CAS table input in the DATA= option. An additional known issue that occurs is when you input a HADAMARD= data set as a CAS table for the SURVEYFREQ, SURVEYIMPUTE, SURVEYLOGISTIC, SURVEYMEANS, SURVEYPHREG, or SURVEYREG procedure, the numerical results of the balanced repeated replication (BRR) method are not reliably reproducible.

Observations Might Not Be Identified Reliably

Three issues related to using observation numbers in SAS/STAT procedures can arise when your input data set is a CAS table:

- Using observation numbers to identify observations is not reliable.
- There is no definite mapping of output data set observations to input CAS table observations.
- Procedure options that refer to observation numbers cannot be applied.

The next three sections discuss these issues in detail.

Using observation numbers to identify observations is not reliable

In regression diagnostics, you typically want to identify extreme observations in the input data. For example, the QUANTREG and ROBUSTREG procedures produce display output that shows outlying or influential observations. If you do not specify ID variables in the ID statement for these procedures, the numerical or graphical output labels these outlying or influential observations by using their observation numbers. However, these observation numbers might not correspond to the rows in the input CAS table. As a result, you cannot rely on these observation numbers to identify the observations.

Table C.5 lists the outputs of SAS/STAT procedures that are known to exhibit the unreliable observation number issue when you use a CAS table as input data, which refer to the raw data input that you specify by using the DATA= option unless noted otherwise. In these cases, the procedures send a note to the log.

Some procedures also recommend that you specify ID variables in the ID statement so that you can use these variable values to aid in identifying observations.

Table C.5 Numerical and Graphical Output Displaying Unreliable Observation Numbers

Procedures	Displayed Output
ANOVA, GEE, GENMOD, GLIMMIX, GLM, ORTHOREG, PROBIT, TTEST	Box plots
CLUSTER	Clustering history, cluster results, and dendrogram
LOESS	Table of output statistics
LOGISTIC, MIXED, PLS, REG	Box plots, diagnostic plots, fit plots, and other plots
NLIN	Leverage and local influence plots
PRINQUAL	Multidimensional preference plot
QUANTREG, ROBUSTREG	Outlier and leverage diagnostics and the corresponding plots
QUANTREG	Conditional estimates when you specify the SHOWOBS option and input a CAS table for the TESTDATA= data set in the CONDDIST statement
TRANSREG	Preference mapping vector plot and preference mapping ideal point plot

A related issue is the display of observation numbers in tooltips of graphical plots, such as box plots, diagnostic plots, needle plots, scatter plots, and so on. Again, you cannot rely on using these observation numbers to identify the observations or rows in the input CAS table. Fortunately, in most cases other information in the tooltips is available to label the observations in these plots. Some procedures also display the ID variable values in tooltips when you specify the ID statement.

There is no definite mapping of output data set observations to input CAS table observations

When a SAS/STAT procedure produces an output data set that contains observational-level data, you cannot assume that the order of observations (or rows) in the output data set and the original data set is the same if one of the data sets is (or both are) a CAS table. In this situation, the mapping of observations between the data sets might not be clear.

For example, the STDIZE procedure standardizes variables by transforming the original (unstandardized) observations in the DATA= data set into standardized observations in the OUT= data set. By default, the OUT= data set contains only the standardized variables. If you use a CAS table as either the DATA= or OUT= data set (or both), the observations in the DATA= and OUT= data sets might not be mapped in a predictable way. Sometimes this might not be an issue, because such a mapping of observations has no effect on subsequent analyses. Problems might arise when you must establish a correspondence of observations between the data sets.

Table C.6 summarizes known mapping issues of the observations in the OUT= data set when you use a CAS table as the input (or output, in the case of PROC STDIZE) data set. In these cases, SAS/STAT procedures issue a note to the log and recommend remedial actions.

Table C.6 Mapping Issues of the Observations in the OUT= Data Set

Procedure	Issue	Solution
CLUSTER	‘OB1’, ‘OB2’, and so on, as the _NAME_ variable values cannot be used to identify the observations	Specify ID variables
MODECLUS	The _OBS_ variable values cannot be used to identify observations	Specify ID variables
PRINQUAL	‘ROW1’, ‘ROW2’, and so on, as the _NAME_ variable values cannot be used to identify the observations	Specify ID variables
STDIZE	No predictable mapping between the DATA= and OUT= observations (also applicable when the OUT= data set is a CAS table)	Use the OPREFIX= option to ensure unstandardized and standardized variables are both available in the OUT= data set

Procedure options that refer to observation numbers cannot be applied

The OBS= option in the CONDDIST statement of the QUANTREG procedure specifies observation numbers to compute some specialized statistics for the corresponding individuals. This option cannot be applied when you input a CAS table, because the specified observation numbers might not correspond to the intended observations. In this case, the procedure disables the option and issues a warning message to the log.

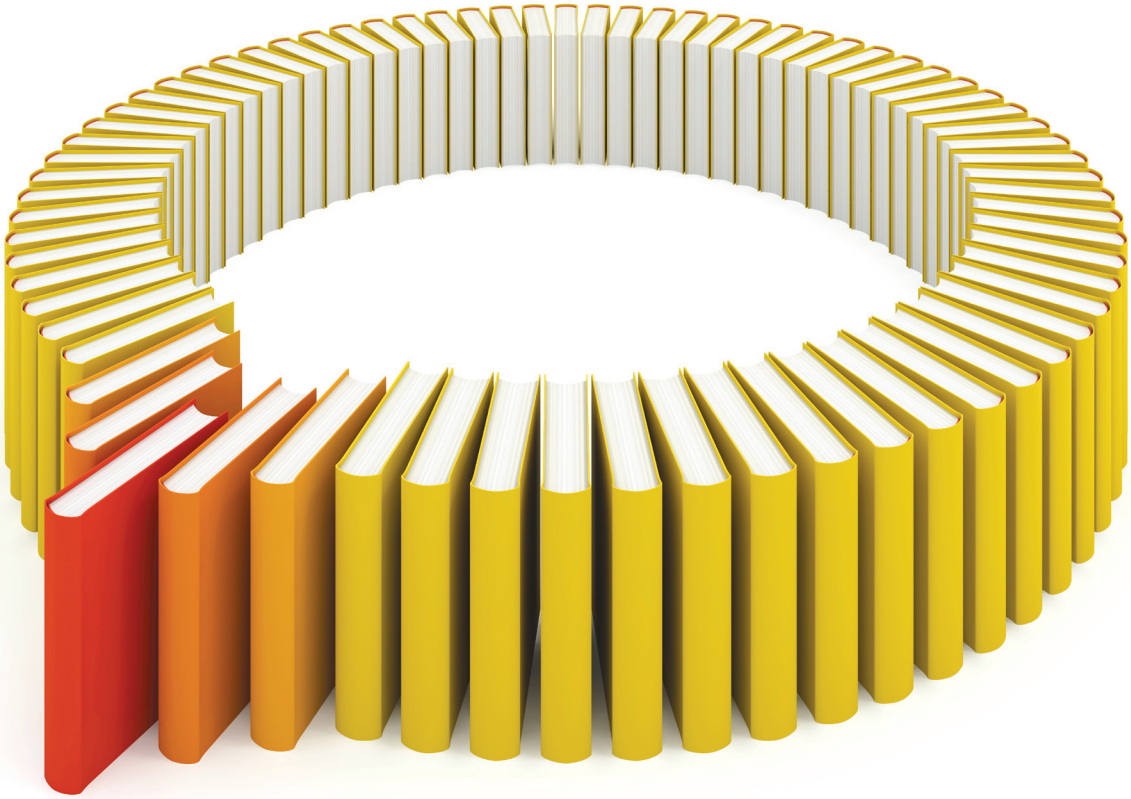

Gain Greater Insight into Your SAS® Software with SAS Books.

Discover all that you need on your journey to knowledge and empowerment.

support.sas.com/bookstore
for additional books and resources.

sas
THE POWER TO KNOW.®

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are trademarks of their respective companies. © 2013 SAS Institute Inc. All rights reserved. S107969US.0613